T.C.
Fırat Üniversitesi
Fen Bilimleri Enstitüsü

Büyük Veri ve Makine Öğrenmesi Kullanılarak Elektrik Tüketim Örüntülerinin Çıkartılması

Fatih ÜNAL

Doktora Tezi

Enerji Sistemleri Mühendisliği Anabilim Dalı
Enerji Planlaması ve Verimliliği Bilim Dalı

Haziran 2022

T.C.
Fırat Üniversitesi
Fen Bilimleri Enstitüsü

Enerji Sistemleri Mühendisliği Anabilim Dalı

Doktora Tezi

Büyük Veri ve Makine Öğrenmesi Kullanılarak Elektrik Tüketim Örüntülerinin Çıkartılması

Tez Yazarı

Fatih ÜNAL

Danışman

Prof.Dr. Sami EKİCİ

Haziran 2022
Elazığ

T.C.
Fırat Üniversitesi
Fen Bilimleri Enstitüsü

Enerji Sistemleri Mühendisliği Anabilim Dalı

Doktora Tezi

Başlığı:	Büyük Veri ve Makine Öğrenmesi Kullanılarak Elektrik Tüketim Örüntülerinin Çıkartılması
Yazarı:	Fatih ÜNAL
İlk Teslim Tarihi:	10.05.2022
Savunma Tarihi:	13.06.2022

Tez Onayı

Fırat Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına göre hazırlanan bu tez aşağıda imzaları bulunan jüri üyeleri tarafından değerlendirilmiş ve akademik dinleyicilere açık yapılan savunma sonucunda OYBİRLİĞİ ile kabul edilmiştir.

		İmza
Danışman:	Prof.Dr. Sami EKİCİ Fırat Üniversitesi, Teknoloji Fakültesi	Onayladım
Başkan:	Prof. Dr. Mehmet Salih MAMİŞ İnönü Üniversitesi, Mühendislik Fakültesi	Onayladım
Üye:	Prof. Dr. Muhsin Tunay GENÇOĞLU Fırat Üniversitesi, Mühendislik Fakültesi	Onayladım
Üye:	Prof. Dr. Resul ÇÖTELİ Fırat Üniversitesi, Teknoloji Fakültesi	Onayladım
Üye:	Doç. Dr. Murat UYAR Bursa Uludağ Üniversitesi, Mühendislik Fakültesi	Onayladım

Bu tez, Enstitü Yönetim Kurulunun ...../...../20.... tarihli toplantısında tescillenmiştir.

Prof. Dr. Kürşat Esat ALYAMAÇ
Enstitü Müdürü

Beyan

Fırat Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırladığım "Büyük Veri ve Makine Öğrenmesi Kullanılarak Elektrik Tüketim Örüntülerinin Çıkartılması" Başlıklı Doktora Tezimin içindeki bütün bilgilerin doğru olduğunu, bilgilerin üretilmesi ve sunulmasında bilimsel etik kurallarına uygun davrandığımı, kullandığım bütün kaynakları atıf yaparak belirttiğimi, maddi ve manevi desteği olan tüm kurum/kuruluş ve kişileri belirttiğimi, burada sunduğum veri ve bilgileri unvan almak amacıyla daha önce hiçbir şekilde kullanmadığımı beyan ederim.

13/06/2022

Fatih ÜNAL

Önsöz

Tez çalışmamızın hazırlanmasıyla sınırlı kalmayarak, kendisini tanıdığım ilk günden itibaren benimle hayata dair ve bilime dair bilgi ve tecrübesini cömertçe paylaşan, özgün akademik bakış açısıyla bu yolculukta bana rehberlik yaparken kritik düşünme, kavramları titizlikle inceleme ve sebep-sonuç ilişkilerini anlamamda tükenmez bir sabır göstererek meslek anlayışıma birçok kazanım sağlayan değerli danışman hocam *Sayın Prof. Dr. Sami EKİCİ*'ye sonsuz teşekkürlerimi sunarım.

Tez çalışması süresince, sağladığı imkanlarla ve gösterdiği anlayışla daha huzurlu bir çalışma ortamında bulunmamızda büyük emeği olan başta *Sayın Prof. Dr. Mehmet ESEN*'e ve Enerji Sistemleri Mühendisliği bölümü öğretim üyelerine de gönülden teşekkürlerimi sunarım. Tez çalışmalarımız esnasında tanıştığımız ve değerli katkılarını esirgemeyen Deggendorf Teknoloji Enstitüsünden *Sayın Prof. Patrick GLAUNER*'e ve Hail Üniversitesinden *Sayın Dr. Abdulaziz ALMALAQ*'a çok teşekkür ederim. Tez çalışmasının LaTeX ile yazılmasında verdiği desteklerden dolayı değerli öğretim üyesi *Sayın Prof. Dr. Resul ÇÖTELİ*'ye ve değerli meslektaşım *Dr. Öğr. Üyesi Ferhat UÇAR*'a da çok teşekkür ederim.

Tez kapsamında kullandığımız veriyi elde etme aşamasında yardımcı olan, Eksim Holding CEO Ofisi Direktörü *Sayın Mahmut Sami SARAÇ*'a, DİCLE EDAŞ Otomatik Sayaç Okuma ve İzleme ekibine ve *Sayın Tolga KOÇ*'a teşekkürlerimi sunarım.

Tez çalışmasında kullanılan donanım ekipmanlarının temininde bize maddi destek sağlayan ve bu tez çalışması ile tamamlanan *FÜBAP TEKF.20.25* numaralı doktora araştırma projesi süresince yardımlarını esirgemeyen FÜBAP Koordinasyon Birimine teşekkür ederim.

Meslek hayatım boyunca her an yanımda olan başta *Arş. Gör. Cihangir KALE* ve *Arş. Gör. Sercan Gülce GÜNGÖR* olmak üzere, tüm saygıdeğer dostlarıma ve yükümü hafifleten tüm çalışma arkadaşlarıma da teşekkürlerimi sunarım.

Tezimi, hayatın tüm zorluklarına rağmen pozitif kalmayı başararak yanımda olan, varlıklarından güç aldığım ve derin bir huzur duyduğum kız kardeşlerim Rüveyda'ya ve Sümeyye'ye, abim Yasin'e, erkek kardeşim Furkan'a ve annem ile babama ithaf ederim.

Fatih ÜNAL

Elazığ, 2022

İçindekiler

Sayfa

Özet

Büyük Veri ve Makine Öğrenmesi Kullanılarak Elektrik Tüketim Örüntülerinin Çıkartılması

Fatih ÜNAL

Doktora Tezi

Fırat Üniversitesi
Fen Bilimleri Enstitüsü

Enerji Sistemleri Mühendisliği Anabilim Dalı
Haziran 2022, Sayfa: xii + 139

Enerji sektöründe günümüze kadar gerçekleştirilen faaliyetlerin büyük bir kısmı temel olarak enerjinin üretimine ve iletimine odaklanmaktadır. Bununla birlikte son yıllarda özellikle, akıllı şebeke ve Gelişmiş Sayaç Altyapısı (Advanced Metering Infrastructure-AMI) gibi teknolojilerin hayatımıza girmesiyle dağıtım ve tüketici seviyesinde de gerçekleştirilen uygulamaların sayısında ciddi bir artış gözlenmektedir. Elektrik dağıtım şirketleri açısından değerlendirildiğinde AMI sistemler ve akıllı sayaçlar, yerinde sayaç okuması ihtiyacını ortadan kaldırarak güç akışının daha hızlı ve güvenilir bir şekilde izlenmesine ve kontrol edilmesine yardımcı olmaktadır. Son kullanıcı açısından değerlendirildiğinde ise akıllı sayaçlar, tüketicinin dağıtım şirketinden aldığı hizmetin kalitesinin artmasını sağlayarak enerjinin etkin ve verimli kullanımı hakkında bir farkındalık oluşturmaktadır.

Bu tez çalışmasında, Büyük Veri analitiği ve makine öğrenmesi yaklaşımları kullanılarak tüketici seviyesinde kısa dönemli yük tahmini ve kayıp-kaçak tespiti uygulaması gerçekleştirilmiştir. Kısa dönemli yük tahmini uygulamasında akıllı sayaç verileri kullanılarak tüketicilerin yük profilleri ve tüketim örüntüleri çıkarılmıştır. Tüketici seviyesinde yük tahmini yüksek değişkenlik, belirsizlik ve veri gizliliği gibi sorunlar nedeniyle dağıtım hattı seviyesinde gerçekleştirilen toplam yük tahminine göre daha karmaşık süreçlerin tasarlanmasını gerektirmektedir. Bu nedenle, kısa dönemli yük tahmini uygulaması için hibrit bir derin öğrenme modeli oluşturulmuştur. Önerilen model, 1-Boyutlu (1-B) evrişimsel sinir ağlarını (Convolutional Neural Network-CNN), uzun kısa süreli bellek (Long Short Term Memory-LSTM) ağlarını ve gelişmiş veri ön işleme yöntemlerini bütünleşik bir çerçevede kullanmaktadır. Gelişmiş veri ön işleme aşamasında yük profillerinde yoğunluk tabanlı aykırılık analizi, parametre optimizasyonu, 1-B CNN ağları kullanılarak öznitelik çıkarma işlemleri gerçekleştirilmiştir. Önerilen hibrit derin öğrenme modeli hem dağıtım şirketinden hem de halka açık akıllı sayaç veri kümelerinde test edilmiştir. Kayıp-kaçak tespit uygulamasında altı farklı sahte veri yerleştirme (False Data Injection-FDI) senaryosu yerel bir dağıtım şirketinin AMI seviyesindeki gözlemci sayaç ve akıllı sayaç verileri kullanılarak incelenmiştir. Aynı zamanda, Highly Comparative Time-Series Analysis (HCTSA) yazılım paketi ve Komşuluk bileşen analizi (Neighborhood Components Analysis-NCA) kullanılarak anormal yük profillerinden kapsamlı istatistiksel öznitelikler hesaplanmış ve derin öğrenme tabanlı bir sınıflandırıcı model oluşturulmuştur. Önerilen derin öğrenme modeli Büyük Veri teknolojileri ile entegre edilerek farklı kayıp-kaçak senaryoları kullanılarak test edilmiştir.

Anahtar Kelimeler: Akıllı Sayaç, Büyük Veri, Gelişmiş Sayaç Altyapısı, Kayıp-Kaçak Tespiti, Yük Tahmini.

Abstract

Extracting Electricity Consumption Patterns Using Big Data and Machine Learning

Fatih ÜNAL

Ph.D. Thesis

Firat University
Graduate School of Natural and Applied Sciences

Department of Energy Systems Engineering
June 2022, Page: xii + 139

Most of the activities carried out in the energy industry so far mainly focused on the generation and transmission of energy. Besides, especially with the introduction of technologies such as smart grid and Advanced Metering Infrastructure (AMI) into our daily lives in recent years, a significant increase has been observed in the number of applications carried out at the distribution and consumer levels. In terms of electricity utilities, AMI systems, and smart meters help to monitor and control the power flow more quickly and reliably by eliminating the need for on-site meter reading. As opposite, in terms of the end-user, smart meters create awareness about the effective and efficient use of energy by increasing the quality of the service received by the consumer from the electricity utility.

In this thesis, short-term load forecasting and non-technical loss (NTL) detection application at the consumer level is carried out by using Big Data analytics and machine learning approaches. In the short-term load forecasting application, the load profiles and consumption patterns of the consumers are analyzed using smart meter data. Load forecasting at the consumer level requires more complex processes to be designed compared to aggregated load estimation at the distribution level due to the issues resulting from the high variability, uncertainty, and data privacy. Therefore, a hybrid deep learning model is designed for the short-term load forecasting application. The proposed model consists of 1-Dimensional (1-D) Convolutional Neural Network (CNN), Long Short Term Memory (LSTM) networks, and advanced data preprocessing methods in an integrated framework. In the advanced data preprocessing stage, density-based outlier analysis, parameter optimization, and feature extraction using 1-D CNN network operations are performed on the load profiles of consumers. The proposed hybrid deep learning model has been tested on both private and public smart meter datasets. In the NTL detection application, six different False Data Injection (FDI) scenarios were examined using the AMI-level observer meter and smart meter data of a regional electricity utility. Besides, comprehensive statistical feature extraction and selection processes are performed for abnormal load profiles using the Highly Comparative Time-Series Analysis (HCTSA) software package and Neighborhood Components Analysis (NCA) and then a deep learning-based classifier model is designed for NTL detection. Finally, the proposed deep learning model has been integrated with Big Data platforms and tested on different NTL scenarios.

Keywords: Advanced Metering Infrastructure, Big Data, Load Forecasting, Non-technical Loss Detection, Smart Meter.

Şekiller Listesi

Tablolar Listesi

Sayfa

Ekler Listesi

Sayfa

Simgeler ve Kisaltmalar

Simgeler

$b_{(\cdot)}$	: LSTM modelin bias değerleri
b_h	: RNN'nin her bir düğümündeki gizli katmanın bias değeri
b_t^l	: x_t enerji zaman serisinde t'inci nöronun l'inci katmanındaki bias değeri
b_y	: RNN'nin her bir düğümündeki çıkış katmanın bias değeri
c	: Kayıp-kaçak veri kümesindeki sınıf sayısı
c_i	: Akıllı sayaç veri kümesindeki i'inci kümenin merkezi
C	: D nesne kümesinin alt kümesi
C_i	: Akıllı sayaç veri kümesindeki i'inci küme
$conv1Dz(.,.)$	: 1-B tam evrişim işlemi
$d_{a,b}$	: Akıllı sayaç veri kümesindeki iki farklı yük profili arasındaki Öklid mesafesi
$d_{i,j}$	: i'inci ve j'inci örnek için MDS konfigürasyon uzaklıkları
D	: Sayaç veri kümesi uzaklık matrisi
$D_{[K,K]}^{(i)}$	: Akıllı sayaç veri kümesindeki i'inci tüketiciye ait uzaklık matrisi
$D_{i,j}$	: HCTSA veri matrisi
$D_w(x_i, x_j)$	: NCA algoritmasında iki örnek arasındaki ağırlıklandırılmış uzaklık
$D_R{}'$	: Bootstrap veri kümesi
e_t	: BRNN'nin geri yöndeki gizli durumu
$E(N)$	: Entropi
$Eps()$	: DBSCAN algoritmasının küme yarıçapı parametresi
EnİyiÇözüm	: Farklı parametreler için BDE-DBSCAN algoritmasının en iyi çözümü
f	: Kayıp-kaçak veri kümesinin normalize edilmemiş öznitelik vektörü
f_k	: XGBoost algoritması zayıf sınıflandırıcı model
f_t	: LSTM geçit birimlerindeki unut kapısı
$f\,(t)$	: Günlük yük profilinin $[0, 1]$ şeklindeki sıralı çıkış fonksiyonu
$f(x)$	: İkili sınıflandırma karar fonksiyonu
$f(\delta_{a,b})$	: Günlük yük profilleri arasındaki monoton uzaklık fonksiyonu
F	: $\mathbb{R}^p$ kümesindeki p boyutlu öznitelik vektörü
F_j	: HCTSA operasyon çıktısı
g_t	: LSTM modelin giriş düğümü
$g(\theta^T x)$	: $h_\theta(x)$ hipotezinde θ parametresiyle ilişkili olasılık değeri
$G(x)$	: GBM algoritması ağırlıklandırılmış çoğunluk oylama fonksiyonu
$G(N)$	: Gini indeksi
h_{t-1}	: LSTM modelin $t-1$ zaman adımındaki gizli durumu
$h(t-1)$	: RNN'nin $t-1$ zaman adımındaki gizli katmanı
$h(x)$	: İkili sınıflandırma ayırma düzlemi
$h_\theta(x)$	: Lojistik regresyon hipotezi
H	: GBM algoritması eğitim hata fonksiyonu
H_{SL}	: Hibrit derin öğrenme modeli
i_t	: LSTM geçit birimlerindeki giriş kapısı
$J(\theta)$	: Lojistik regresyonunda en büyük olabilirlik kestirimi
$k(z)$	: NCA algoritmasının çekirdek fonksiyonu
K	: Akıllı sayaç veri kümesi kullanılabilir günlük yük profili sayısı

Simgeler

$L1, L2$	: Düzenlileştirme parametresi
$L(y_i, \hat{y}_i)$	: XGBoost algoritması dışbükey türevlenebilir kayıp fonksiyonu
$MinPts$	: DBSCAN algoritmasının minimum nesne sayısı parametresi
$\mathrm{m_f}$	: f özniteliklerinin medyanı
n	: $\mathbf{y}'$ veri kümesinde gelecek dönem için tahmin edilecek örneklerin sayısı
n_i	: C_i kümesinde yer alan toplam örnek sayısı
N	: Yük profilindeki ölçüm sayısı [24, 48, 96]
$\mathrm{N_h}$	: Hata matrisindeki toplam örnek sayısı
o_t	: LSTM geçit birimlerindeki çıkış kapısı
p_1	: Hata matrisinde rastgele seçilen bir sınıf etiketinin pozitif sınıfa dahil olma olasılığı
p_2	: Sınıflandırıcının pozitif bir etiket üretme olasılığı
p_i	: NCA algoritmasında i'inci örneğin doğru sınıflandırılmış olma olasılığı
p_{ij}	: NCA algoritmasında i'inci örneğin referans noktası olarak j'inci örneği seçme olasılığı
$\text{Popülasyon}_{[i,ii]}$	: i'inci epok ve ii'inci iterasyonda oluşturulan popülasyon
$\mathrm{r_f}$	: f özniteliklerinin çeyrek sapma aralığı
Rast. doğruluk	: p_1 ve p_2'nin tesadüfen çakışma olasılığı
s	: MDS konfigürasyon alt uzayı
$\tilde{\mathrm{sf}}$	: Kayıp-kaçak veri kümesinin sağlam Sigmoid kullanılarak normalize edilmiş öznitelik değerleri
s_t	: LSTM modelin iç durumu
s_i^{l-1}	: x_t enerji zaman serisinde i'inci nöronun $l-1$'inci katmanındaki çıkış değeri
S_k	: k'ıncı boyuttaki en düşük $stress$ değeri
$S(\hat{X})$	: $\mathbb{R}^k$ uzayındaki veri kümesinin stres fonksiyonu
t	: FDI saldırısının zaman aralığı
t_1	: FDI saldırısının başlangıç zamanı
t_2	: FDI saldırısının bitiş zamanı
$tanh$	: LSTM bellek hücresindeki tanjant aktivasyon fonksiyonu
$\mathrm{V_0}$	: Gerilim algılayıcı çıkış gerilimi
$\mathrm{V_{giriş}}$	: Gerilim algılayıcı giriş gerilimi
w	: NCA algoritmasının ağırlıklandırma vektörü
$w_{(\cdot)}$	: LSTM modelin ağırlık matrisi
w_k	: F matrisindeki öznitelik ağırlıkları
$w^{h,h}$	: RNN'nin ardışık zaman adımlarında gizli katman ile kendisi arasındaki özyineleme ağırlık matrisi
$w^{h,x}$	: RNN'nin giriş katmanı ve gizli katmanı arasındaki ağırlık katsayısı matrisi
$w_{i,k}^{l-1}$	: x_t enerji zaman serisinde $l-1$ katmanındaki i'nci nöron ile l'inci katmandaki k'ncı nöron arasında bulunan çekirdek fonksiyonu
x	: Sayaç ölçüm değeri
x_t	: t zaman adımındaki gerçek sayaç ölçüm değeri
$x(t)$	: RNN'nin özyineli düğümlerinde t zaman adımındaki mevcut veri noktası
$\widetilde{x}_t$	: t zaman adımındaki değiştirilmiş sayaç ölçüm değeri
x_t^l	: x_t enerji zaman serisinin 1-B CNN katmanı giriş vektörü

Simgeler

x_i	: Giriş veri kümesi gerçek değerleri
x'_i	: Giriş veri kümesinin $[0, 1]$ aralığında normalize edilmiş değerleri
$x_{maksimum}$	: Giriş veri kümesinin en yüksek değeri
$x_{minimum}$	: Giriş veri kümesinin en düşük değeri
$\hat{X}_{n\times k}$	: $\mathbb{R}^k$ uzayındaki sayaç veri kümesi
$(X^{(i)}_{[K,N]})$	: Akıllı sayaç veri kümesi mesafe matrisi
$(X_{[K,2]})$	: Akıllı sayaç veri kümesindeki yük profillerinin 2-B altuzay izdüşümleri
y	: İkili sınıflandırmada çıkış değeri
y_i	: Kayıp-kaçak veri kümesindeki gerçek sınıf etiketleri
$\hat{y}^t$	: RNN'nin t zaman adımındaki çıkış değeri
$\mathbf{y}'$	: Giriş veri kümesi için normalize edilmiş tahmin değerleri
y'_{i_n}	: i_n'inci zaman adımındaki tahmini yük değeri
$\overline{y}$	: Giriş veri kümesinindeki tüketim değerlerinin ortalaması
$(\delta_{a,b})$	: Benzemezlik ölçüşü
(Δ^l_k)	: x_t enerji zaman serisinde k'ıncı nöronun l'inci katmanındaki hata farkı
$rev(\cdot)$	: Dizi ters çevirme operatörü
σ	: LSTM bellek hücresindeki sigmoid aktivasyon fonksiyonu
σ_N	: NCA algoritmasının çekirdek genişliği
$\odot$	: Hadamard çarpımı
$\xi(w)$	: NCA algoritmasında σ_N sıfıra yaklaştığında elde edilen gerçek LOO sınıflandırma doğruluğu
λ	: NCA algoritması düzenleme terimi
$\xi(.)$	: İkili sınıflandırma kayıp fonksiyonu
$\phi(.)$	: SVM çekirdek fonksiyonu
α_m	: GBM algoritmasında zayıf sınıflandırıcının tahmin sonucuna olan katkısının ağırlığı
$\Omega(.)$	: XGBoost algoritması model karmaşıklığı
κ	: Cohen'in Kappa Katsayısı

Kısaltmalar

AB	: Avrupa Birliği
ABD	: Amerika Birleşik Devletleri
ADC	: Analog Sinyalin Dijitale Dönüştürülmesi (Analogue to Digital Conversation)
AMI	: Gelişmiş Sayaç Altyapısı (Advanced Metering Infrastructure)
AMR	: Otomatik Sayaç Okuma (Automatic Meter Reading)
ANN	: Yapay Sinir Ağları (Artificial Neural Networks)
ANSI	: Amerikan Ulusal Standartlar Enstitüsü (American National Standards Institute)
AR	: Özbağlanımlı Model (Autoregressive Model)
ARMA	: Özbağlanımlı Kayan Ortalamalı Model (Autoregressive Moving Average)
ARIMA	: Özbağlanımlı Tümlenik Kayan Ortalamalı Model (Autoregressive Integrated Moving Average)
ASCII	: Bilgi Alışveriş Amerikan Kod Standardı (American Standard Code for Information Interchange)
ASF	: Apache Yazılım Vakfı (Apache Software Foundation)
API	: Uygulama Programlama Arayüzü (Application Programming Interface)
BDE	: İkili Diferansiyel Gelişim (Binary Differential Evolution)
CBS	: Coğrafi Bilgi Sistemi
CNN	: Evrişimsel Sinir Ağı (Convolutional Neural Network)
CNTK	: Microsoft Bilişsel Araç Seti (Microsoft Cognitive Toolkit)
CLC	: Konteyner Yaşam Döngüsü (Container Life Cycle)
CV	: Çapraz Geçerlilik Sınaması (Cross Validation)
CVP	: Çapraz Geçerlilik Bölütleme (Cross Validation Partition)
DAG	: Döngüsel Olmayan Grafik (Directed Acyclic Graph)
DBN	: Derin İnanç Ağı (Deep Belief Network)
DBSCAN	: Density-based Spatial Clustering Applications with Noise
DEPSAŞ	: Dicle Elektrik Perakende Satış Anonim Şirketi
DL	: Derin Öğrenme (Deep Learning)
DSO	: Dağıtım Sistemi Operatörü (Distribution System Operator)
DSP	: Dijital Sinyal İşlemcisi (Digital Signal Processor)
DT	: Karar Ağacı (Decision Tree)
EDA	: Keşifsel Veri Analizi (Exploratory Data Analysis)
EFB	: Özel Değişken Paketi (Exclusive Feature Bundling)
ENTSO-E	: Avrupa Elektrik İletim Sistemi Operatörleri Birliği (European Network of Transmission System Operators for Electricity)
EPDK	: Enerji Piyasası Düzenleme Kurumu
FDI	: Sahte Veri Yerleştirme (False Data Injection)
FN	: Yanlış Negatif (False Negative)
FP	: Yanlış Pozitif (False Positive)
FPR	: Yanlış Pozitif Oranı (False Positive Rate)
GAN	: Çekişmeli Üretici Ağı (Generative Adversarial Network)
GBM	: Gradyan Artırma Makinesi (Gradient Boosting Machine)
GFS	: Google Dosya Sistemi (Google File System)
GUI	: Grafik Kullanıcı Arayüzü (Graphical User Interface)

Kısaltmalar

GOSS	: Gradyan Tabanlı Tek Yönlü Örnekleme (Gradient-based One-Side Sampling)
GRU	: Geçitli Tekrarlayan Ünite (Gated Recurrent Unit)
HAN	: Ev Alan Ağı (Home Area Network)
HCTSA	: Highly Comparative Time-Series Analysis
HDFS	: Hadoop Dağıtık Dosya Sistemi (Hadoop Distributed File System)
HDP	: Hortonworks Veri Platformu (Hortonworks Data Platform)
IaaS	: Servis Olarak Altyapı (Infrastructure as a Service)
IEC	: Uluslararası Elektroteknik Komisyonu (International Electrotechnical Commission)
IEEE	: Elektrik ve Elektronik Mühendisleri Enstitüsü (Institute of Electrical and Electronics Engineers)
ILM	: Müdahaleci Yük İzleme (Intrusive Load Monitoring)
ISO	: Bağımsız Sistem İşletmesi (Independent System Operator)
JVM	: Java Sanal Makinesi (Java Virtual Machine)
LR	: Lojistik Regresyon (Logistic Regression)
LSTM	: Uzun Kısa Süreli Bellek (Long Short Term Memory)
MAE	: Ortalama Mutlak Hata (Mean Absolute Error)
MAPE	: Ortalama Mutlak Yüzde Hata (Mean Absolute Percentage Error)
MDS	: Çok Boyutlu Ölçekleme (Multidimensional Scaling)
MLlib	: Machine Learning Library
MMLSpark	: Microsoft Machine Learning for Apache Spark
MLP	: Çok Katmanlı Algılayıcı (Multilayer Perceptron)
MSE	: Ortalama Karesel Hata (Mean Square Errror)
NaN	: Sayı Olmayan Değer (Not a Number)
NAN	: Komşu Alan Ağı (Neighborhood Area Network)
NB	: Naif Bayes (Naive Bayes)
NCA	: Komşuluk Bileşen Analizi (Neighborhood Components Analysis)
NDFS	: Nutch Dağıtık Dosya Sistemi (Nutch Distributed File System)
NILM	: Müdahaleci Olmayan Yük İzleme (Non-Intrusive Load Monitoring)
NTL	: Teknik Olmayan Kayıp (Non-technical Losses)
OBİS	: Nesne Tanımlama Sistemi (Object Identification System)
OSOS	: Otomatik Sayaç Okuma Sistemi
PaaS	: Servis Olarak Platform (Platform as a Service)
PBS	: Taşınabilir Toplu İş Sistemi (Portable Batch System)
PCA	: Temel Bileşen Analizi (Principal Component Analysis)
PLC	: Güç Hattı İletişimi (Power Line Communication)
PLC	: Programlanabilir Sayısal Kontroller (Programmable Logic Controller)
PQ	: Güç Kalitesi (Power Quality)
RDD	: Resilient Distributed Dataset
RF	: Rastgele Orman (Random Forest)
RLP	: Gerçek Yük Profili (Real Load Profile)
RMS	: Karesel Ortalamanın Karekökü (Root Mean Squared)
RMSE	: Kök Ortalama Kare Hata (Root Mean Square Error)
RMSSTD	: Kök Ortalama karesel Standart Sapma (Root Mean Square Standard Deviation)

Kısaltmalar

RNN	: Özyineli Sinir Ağı (Recurrent Neural Network)
RTO	: Bölgesel İletim Organizasyonu (Regional Transmission Organization)
SaaS	: Servis Olarak Yazılım (Software as a Service)
SARIMAX	: Mevsimsel ARIMA (Seasonal Autoregressive Integrated Moving Average with Exogenous Inputs)
SGD	: Rasgele Gradyan İnişi (Stochastic Gradient Descent)
SGSC	: Akıllı Şebeke Akıllı Şehir (Smart Grid Smart City)
SLA	: Hizmet Seviyesi Anlaşması (Service Level Agreement)
SLURM	: Basit Linux Yardımcı Programı (Simple Linux Utility for Resource Management)
SMPS	: Anahtarlamalı Güç Kaynağı (Switch Mode Power Supply)
SOA	: Servis Odaklı Mimari (Service-oriented Architecture)
SSH	: Secure Shell Protocol
SVM	: Destek Vektör Makineleri (Support Vector Machines)
SVR	: Destek Vektör Regresyonu (Support Vector Regression)
TEDAŞ	: Türkiye Elektrik Dağıtım Anonim Şirketi
THD	: Toplam Harmonik Bozulma (Total Harmonic Distortion)
TSE	: Türk Standartları Enstitüsü
TSO	: İletim Sistemi Operatörü (Transmission System Operator)
TP	: Doğru Pozitif (True Positive)
TN	: Doğru Negatif (True Negative)
UDF	: Kullanıcı Tanımlı Fonksiyon (User Defined Function)
WAN	: Geniş Alan Ağı (Wide Area Network)
YARN	: Yet Another Resource Negotiator

1. Giriş

Akıllı şebeke; güç akışı, iş akışı ve veri akışının birlikte gerçekleştiği siber-fiziksel-sosyal bir yapının genel bir tanımıdır. Günümüzde akıllı sayaçlar sürekli olarak elektrik kullanıcılarına tüketimleri hakkında bilgi sağlayarak bu sistemin en önemli bileşenlerinden biri haline gelmiştir. Dünyanın dört bir yanındaki ülkeler, şebekelerin modernizasyonu ve dijitalleşmesi için önemli yatırım kararları almakla birlikte elektrik kullanıcılarının enerjinin üretim, iletim ve dağıtım aşamalarında aktif rol alabilmelerine katkı sağlayan teknolojilerin geliştirilmesine de destek vermektedir. Akıllı sayaçlar bu noktada veri analiz yöntemlerinin kullanılmasına ve makine öğrenmesi temelli yaklaşımların yaygınlaşmasına imkan sağlayan detaylı bilgi paylaşımında kritir bir öneme sahiptir. Büyük veri ve makine öğrenmesi terimleri günümüzde akıllı sayaçlar ile birlikte yaygın olarak kullanılmaktadır. Farklı sektörlerde faaliyet gösteren şirketler ve üniversitelerde araştırma yapan akademisyenler günlük yaşamın problemlerine makine öğrenmesi yöntemlerini uygulayarak çözüm getirmeye çalışmaktadır. Enerji sektörü bu teknolojilerin yaygın olarak kullanıldığı alanlardan biridir ve akıllı sayaç veri analitiği yöntemleri ile elektrik kullanıcılarının tüketim davranışlarının anlaşılmasında, arz-talep ve enerji yönetiminin sağlıklı bir şekilde gerçekleştirilmesinde ve elektrik kullanıcılarına sağlanan hizmetlerin iyileştirilebilmesi için alt yapı çalışmalarında sıklıkla tercih edilmektedir.

Enerji sektöründe günümüze kadar gerçekleştirilen faaliyetlerin çoğu temel olarak üretim ve iletim sektörlerine odaklanırken, dağıtım aşaması ve son kullanıcı son yıllarda gündeme gelmiştir. Bu durumun popülerlik kazanmasında şüphesiz akıllı sayaç kurulumunun her geçen yıl katlanarak artması ve mobil uygulamalar üzerinden kullanıcılarının tüketimleri hakkında detaylı bilgiye sahip olmak istemeleri oldukça etkili olmuştur. Ayrıca dağıtım sistemleri son yıllarda dağıtık enerji kaynakları, çoklu enerji sistemleri entegrasyonu, kontrol altyapısı ve veri toplama ekipmanları ile oldukça ilginç tasarımların uygulandığı bir ortam haline gelmiştir. Dağıtım sistemleriyle entegre yenilenebilir enerji kaynakları ve enerji verimliliği uygulamaları aynı zamanda dekarbonizasyon ve ilkim değişikliğini önlemek için kullanılabilecek etkili iki yaklaşımdır. Bu noktada akıllı sayaçlar gibi veri toplama ekipmanları mevcut durumu detaylı ve hızlı bir şekilde paylaşabilmeye imkan sağladığı ve ilk yatırım maliyetleri dikkate alındığında şebeke modernizasyonu uygulamaları içerisinde tercih edilebilecek en mantıklı seçeneklerin başında gelmektedir.

Geleneksel bir elektrik şebekesi temel olarak bir dizi elektrik jeneratörü, iletim hattı ve trafo merkezinden oluşur ve birçok farklı sınıfta yer alan tüketiciye (örneğin; mesken, ticarethane veya endüstriyel kullanım, vb.) aynı anda hizmet sağlar. Fransa'nın Paris kentinde 2015 yılında düzenlenen Paris Birleşmiş Milletler İklim Değişikliği Konferansı'ndaki iklim değişikliği anlaşmasının ardından, bu geleneksel elektrik şebekesi paradigmasını değiştirmek için adımlar atılmıştır [1]. Bu değişikliğin arkasında yatan ana

Şekil 1.1. AB Enerji Verimliliği Direktifleri [2]

nedenlerden birincisi karbon emisyonlarını azaltma ihtiyacı, ikincisi ise elektrik enerjisi üretiminin merkeziyetsizleştirilmesini (decentralization) sağlamaktır. Akıllı şebeke kavramı bu zorlukların üstesinden gelebilmek için ortaya çıkmış, çok sayıda sensörün kullanıldığı, yenilenebilir enerji kaynaklarına dayalı dağıtık üretimi teşvik eden bir sistem olarak da değerlendirilmektedir. Burada akıllı sayaçlar, geleneksel elektrik şebekesinden akıllı şebekeye geçişte sahip olduğu gelişmiş sensör teknolojilerinden faydalanarak kritik bir rol oynamaktadır. Avrupa Birliği (AB) 'nin enerji verimliliği kapsamında oluşturduğu direktifler Şekil 1.1.'de gösterilmektedir.

Elektrik dağıtım şirketleri açısından değerlendirildiğinde akıllı sayaçlar, yerinde sayaç okuma ihtiyacını ortadan kaldırarak güç akışını daha hızlı ve doğru bir şekilde izlemeye imkan sağlar. Ayrıca akıllı sayaçlar, optimize edilmiş kaynak kullanımına izin verir ve yarı gerçek zamanlı olarak veri sağlayarak elektrik kesintilerini azaltırken yük dengesizliği problemlerinin çözümüne yardımcı olur. Bunun dışında dinamik tarifelendirme seçeneği, şebeke yatırımlarında gereksiz sermaye harcamasını önler ve mevcut kaynaklarla gelirin optimize edilmesine de katkı sağlar. Akıllı sayaçlar için yeni ölçüm teknolojisine ve ilgili yazılım altyapısına uzun vadeli bir finansal taahhüt gereklidir ve güç hizmeti sağlayıcıları, büyük miktardaki ölçüm verisinin yönetimini, depolanmasını, güvenliğini ve gizliliğini sağlamakla yükümlüdür.

Son kullanıcı açısından değerlendirildiğinde akıllı sayaçlar, tüketicinin dağıtım şirketinden aldığı hizmet kalitesinin artmasını sağlar, daha bilgilendirici faturalandırma işlemleri gerçekleştirir ve enerjinin etkin kullanımı hakkında farkındalık oluşturur. Ayrıca, son kullanıcının tüketim davranışı hakkında bilgi sahibi olması tüketim alışkanlıklarını değiştirmesine yardımcı olur. Burada dağıtım şirketleri, personel eğitimi ve ekipman geliştirme aşamasında yüksek bir maliyetle karşı karşıya kalabilir ve eski sayaçların yeni sayaçlarla değiştirilmesi aşamasında bazı olası olumsuz halk tepkileriyle de uğraşmak zorunda kalabilir. Buna ek olarak son kullanıcılar, mahremiyetlerini ve toplanan kişisel verileri koruma konusunda oluşabilecek belirsizliklerin yanı sıra doğabilecek bazı ek ücretlerle de karşı karşıya kalabilmektedir.

Akıllı sayaçlar son on yıl içerisinde başta Amerika Birleşik Devletleri (ABD), Çin ve Avrupa Birliği ülkeleri olmak üzere birçok gelişmiş ülkede hızla yaygınlaşmaya başlamıştır. Berg Insight adlı bağımsız endüstri analisti ve danışmanlık firması, Kuzey Amerika'daki akıllı sayaçların 2019-2025 tahmin dönemi boyunca yıllık yüzde 5.7'lik bir bileşik büyüme oranıyla 2019'da 110.4 milyondan 2025'te toplam 153.8 milyona çıkacağını tahmin etmektedir. Asya-Pasifik pazarında (yani Çin, Japonya, Güney Kore, Hindistan, Avustralya ve Yeni Zelanda) – akıllı sayaç kurulumunun ise 2019'da 659.3 milyon üniteden 2025'te 886.1 milyon üniteye çıkacağını tahmin etmektedir [3]. Avrupa Birliği komisyonunun 2020 yılında hazırladığı teknik raporda ise 2024 yılına kadar, AB çapında %44'lük bir kurulum ve devreya alma oranına karşılık gelen 51 milyon akıllı sayacın devreye gireceği tahmin edilmektedir [2]. Akıllı bir sayacın dakikalık aralıklarla ölçüm kaydedebilme özelliği göz önünde bulundurulduğunda, kurulum ve devreye alınma sonrasında büyük miktarlarda elektrik verisi üretilecektir. Yakın gelecekte karşılaşılması yüksek bir ihtimal olan bu Büyük Veri problemi, Büyük Veri ve Veri Analitiği alanlarının bu tür verileri keşfetmek ve analiz etmek için gerekli araçları tasarlamalarını gerektirmektedir.

Akıllı sayaç verisi, gelişmiş veri analitiği ve Büyük Veri analizi araçlarından faydalanılarak akıllı şebekede beş genel uygulama sahasında kullanılmaktadır. Akıllı sayaç verisinin en yaygın kullanıldığı uygulama alanlarından bazıları; yük eğrisi analizi (load profiling), arz-talep dengeleme (demand-response), yük tahmini (load forecasting), tüketici bölütleme (customer segmentation) ve kayıp-kaçak tespiti (non-technical loss detection) olarak gruplandırılabilir. Ayrıca, veri işleme araçlarının gelişen akıllı şebeke teknolojileriyle entegrasyonu da akıllı sayaç veri analizi uygulamaları içerisinde değerlendirilebilir. Yukarıda kısaca açıklanan akıllı sayaç uygulama alanları hakkında detaylı bilgi aşağıda verilmiştir:

Yük Eğrisi Analizi: Akıllı sayaç verilerinin en yaygın kullanım alanlarından bir tanesidir ve temelde tüketiciye özgü yük profillerinin tespitinde kullanılmaktadır [4]. Bu tür uygulamalar için yaygın kullanılan araçların başında eğiticisiz kümeleme algoritmaları gelmektedir [5]. Yük eğrisi analizi farklı yük ayrıştırma (load disaggregation) seviyelerinde tüketim örüntülerini tespit etmek için eğiticili makine öğrenmesi algoritmalarıyla da birlikte kullanılmaktadır. Bu yöntemde, Müdahaleci Yük İzleme (Intrusive Load Monitoring-ILM) ve Müdahaleci Olmayan Yük İzleme (Non-Intrusive Load Monitoring-NILM) olmak üzere iki farklı yaklaşım bulunmaktadır. ILM yaklaşımı, uygulama maliyetlerini artıran ek ölçüm ekipmanlarına ihtiyaç duyarken, NILM yaklaşımında daha düşük maliyetli donanım ekipmanları ile daha yoğun analitik yöntemler kullanılmaktadır [6].

Arz-talep Dengeleme: Tüketicileri enerji üretim ve planlama süreçlerine dahil ettiği için akıllı şebeke uygulamalarında potansiyeli en yüksek uygulamalardan biridir. Bu uygulamalardaki temel motivasyon, tüketicilerin azami talep dönemlerindeki (peak demand period) tepe yük (peak load) değerini azaltmak için tüketim modelini kasıtlı

olarak değiştirmektir [7], [8].

Yük Tahmini: Tüketicilerin gelecek dönem enerji talebini tahmin etmeye yardımcı olan uygulamalardır. Bu tür uygulamalar Dağıtım Sistemi Operatörlerine (Distribution System Operators-DSO) gelecek dönem enerji talebinin planlanmasında katkı sağlayarak iş-akış süreçlerini değiştirdiği için son yıllarda sıklıkla tercih edilmektedir. Ayrıca, akıllı şebekede tüketicilerin artan aktif katılımıyla oluşan dinamik enerji piyasası yüksek doğrulukla elde edilen tüketimi tahminine dayanmaktadır.

Kayıp-kaçak Tespiti: Literatürde teknik olmayan kayıp (Non-technical Losses-NTL) tespiti olarak da adlandırılan bu tür uygulamalar doğrudan dağıtım şirketlerinin maliyetlerini azaltmayı amaçlar. Ayrıca, büyük ölçekli kayıp-kaçak uygulamaları, güç sisteminde ciddi yük dengesizliklerine neden olabilir. Bu nedenle, karmaşık elektrik şebekelerinde kayıp-kaçak tespiti uygulamaları son yıllarda servis sağlayıcılarının öncellikli yatırım planları arasında yer almaktadır. Kayıp-kaçak tespiti uygulamaları, birçok dağıtım şirketi için izinsiz elektrik kullanımının azaltmak ve tüketicilerin yasal haklarını korumak için yeni önlemlerin alınması aşamasında önemli bir rol oynamaktadır. Kayıp-kaçak kullanımı, özellikle gelişmekte olan ülkelerde ortaya çıkan ve ekonomik göstergelerle doğrudan ilişkili bir olgudur. Kayıp-kaçak kullanımı bu tür gelişmekte olan ülkelerin toplam tüketiminin önemli bir oranıdır ve ülkeden ülkeye göre değişiklik gösterebilir. Örneğin; Brezilya'da kayıp-kaçak kullanımının dağıtılan toplam elektriğin %40'ına kadar ulaştığı tahmin edilmektedir [9]. Dünya çapındaki kayıp-kaçak kullanımının parasal değerinin yılda yaklaşık 100 Milyar dolar ($) olduğu tahmin edilmektedir. Hindistan'daki elektrik kurumları, elektrik hırsızlığı nedeniyle her yıl 4.5 Milyar dolar ($) kaybetmektedir [10]. Brezilya Elektrik Düzenleme Kurumu, Brezilya'daki kayıp-kaçak kullanımının finansal kaybının 2011 yılında 4.0 Milyar dolar olduğu bildirmiştir [11]. 2004 yılında, Malezya'nın resmi elektrik dağıtım kuruluşu olan Tenaga Nasional Berhad, arızalı sayaçlar ve elektrik hırsızlığı nedeniyle yılda 229 Milyon dolar ($) gelir kaybı bildirmiştir [12]. Türkiye'de ise 2019 yılında teknik ve teknik olmayan kayıp oranı %12.7 olarak bildirilmiştir [13]. Bu örneklerin dışında, gelişmiş ülkeler dahil olmak üzere dünya çapında çok sayıda örnek sıralanabilir [9].

Bu tez çalışmasında, gerçek akıllı sayaç veri kümeleri üzerinde yukarıda kısaca açıklanan yük tahmini ve kayıp-kaçak tespiti uygulamaları gerçekleştirilmiştir. Yük tahmini hava durumu, tüketici sınıfı, konum ve zaman bilgisi gibi birçok parametreden etkilenmektedir. Yük tahmini uygulamaları aynı zamanda tahmin ufkunun büyüklüğüne göre kısa dönemli, orta dönemli ve uzun dönemli olmak üzere üç grupta değerlendirilmektedir. Bu çalışmada, DSO'lar için potansiyel arz-talep yanıtını tahmin etmek ve gerçek zamanlı güç akışı kontrolüne ek bilgi sağlamak amacıyla tüketici seviyesinde kısa dönemli yük tahmini problemleri incelenmiştir. Dağıtım hattı seviyesinde gerçekleştirilen kısa dönemli yük tahmin uygulaması, dağıtım şirketlerinin güç akışında tepe kaydırma (peak shifting) ve yük çizelgeleme (load scheduling) gibi kısa dönemli planlama uygulamalarında kullanılabilir. Diğer yandan tüketiciler, aylık faturalarını

azaltmak için enerji tüketim davranışlarını sistemden gelen gerçek zamanlı geri bildirimleri dikkate alarak planlayabilir. Kısa dönemli yük tahmini uygulamasında yoğunluk tabanlı aykırılık analizi ile hibrit derin öğrenme (Deep Learning-DL) modelinin tasarımı birlikte gerçekleştirilmiştir. Ayrıca, aykırılık analizinde parametre optimizasyonu tüketici özelinde gerçekleştirilerek elde edilen sonuçlar karar destek sistemlerine yardımcı olmaktadır.

Kayıp-kaçak kullanımı günümüzün modern güç sistemleri üzerinde doğrudan ve dolaylı birçok olumsuz etkiye neden olmaktadır. Dağıtım şirketlerinin gelir kaybı, kayıp-kaçak kullanımının doğrudan olumsuz etkilerindendir. Bu uygulamalar aynı zamanda, planlanmamış yerinde denetim ve bakım-onarım işlemleri, arızalı sayaçların değiştirilmesi gibi ek masraflara da neden olmaktadır. Bu çalışmada, dağıtım hattı seviyesinde akıllı sayaç ölçümüne müdehale edilerek gerçekleştirilen kayıp-kaçak uygulamaları incelenmiştir. Dağıtım hattı seviyesinde gerçekleştirilen kayıp-kaçak uygulamasında aynı trafo istasyonuna bağlı tüketicilerin akıllı sayaç verileri kullanılmıştır. Kayıp-kaçak tespiti uygulamasında toplam altı farklı sayaç verisine müdehale yöntemi incelenmiş ve kayıp-kaçak durumlarını gerçek durumlardan ayırt etmeye yardımcı olabilecek istatistiksel öznitelikler araştırılmıştır. Son aşamada ise, Büyük Veri analitiği ile entegre DL tabanlı bir sınıflandırıcı model tasarlanmıştır. Elde edilen bulgulardan önerilen DL modeli birçok farklı anomali tespit uygulamasında kullanılabilme potansiyeline sahip olduğu sonucuna ulaşılmıştır.

1.1. Tezin Amacı ve Kapsamı

Bu tez çalışmasının temel motivasyonu, akıllı şebeke uygulamalarında Büyük Veri analitiğinin ve makine öğrenmesi yöntemlerinin birlikte kullanıldığı genel bir çerçeve tasarlamaktır. Geliştirilen bu genel çerçevesinin, tüketici seviyesinde yük tahmini ve kayıp-kaçak tespiti uygulamaları için kullanılması amaçlanmaktadır. Ayrıca, gerçek tüketim verilerinden elde edilecek örüntülerin, enerji piyasasında üretim, planlama, fiyatlandırma ve süreç optimizasyonu gibi farklı uygulamalar için kullanılabilmesi amaçlanmaktadır.

Yük tahmin modeli, tahmin hatalarını en aza indirebilmek amacıyla yoğunluk tabanlı aykırılık analizi ve hibrit DL yaklaşımlarını birlikte kullanmaktadır. Önerilen yük tahmin modelinin performansı, benzer amaçlar için kullanılan diğer makine öğrenme algoritmaları ile kıyaslanarak avantajları ve dezavantajları değerlendirilecektir. Gerçek sayaç verilerinde karşılaşılan bozucu etkiler dikkate alınarak farklı durum senaryoları oluşturulacak ve önerilen hibrit DL modelinin bu senaryoların hangilerinde iyi performans gösterdiği belirlenecektir. Mevcut matematiksel yaklaşımların yeterli performans gösteremediği durumlarda veri odaklı yük tahmini yapabilen bir sistem geliştirmek bu akıllı şebeke uygulamasının temel amacıdır.

Kayıp-kaçak tespit modeli, sınıflandırma hatalarını en aza indirebilmek amacıyla kaspamlı öznitelik çıkarma, seçme ve DL tabanlı sınıflandırıcı tasarlama süreçlerini Büyük Veri analiğiti ile entegre bir yapıda kullanmaktadır. Büyük Veri yöntemlerinin ve paralel

hesaplamanın kullanılması, büyük miktarlardaki akıllı sayaç verisindeki anormal davranışları, kısa sürelerde gerçekleşen hızlı değişimleri ve gerçek durumları analiz edebilmek için yeni fırsatlar sunmaktadır. Bu uygulamanın temel amacı ise, enerji piyasası, akıllı şehir uygulamaları gibi diğer birçok sektörde uygulanabilecek bir Büyük Veri çerçevesi ile entegre bir anomali tespit modeli geliştirmektir.

1.2. Literatür Çalışması

Akıllı sayaçların yaygınlaşmasıyla çok büyük miktarda ve yüksek örneklemeli elektrik tüketim verisinin elde edilmesi kolaylaşmıştır. Ayrıca, enerji sektörünün serbestleştirilmesi, özellikle elektrik dağıtım tarafında, Dünya genelinde sürekli ve artan bir şekilde devam etmektedir. Akıllı şebekede, verimliliği ve sürdürülebilirliği artırmak için büyük akıllı sayaç verilerinin hangi uygulamalarda ve nasıl kullanılacağı ise öncelikli bir konudur [14]. Bu tez çalışmasında, literatür araştırması yük tahmini, kayıp-kaçak tespiti ve Büyük Veri uygulamaları olarak belirlenen üç farklı kategoride yapılmıştır. Bu bölümde aynı zamanda mevcut durum, teknolojik gelişmeler, karşılaşılan problemler ve çözüm yolları açıklanarak mevcut çalışmanın genel yapısı literatür çalışmalarının da yardımıyla okuyucuya sunulacaktır.

Kısa Dönemli Yük Tahmini Uygulamaları: Yüksek gerilim seviyelerindeki yük profilleri ile karşılaştırıldığında, bir tüketici grubunun orta-düşük gerilim seviyesindeki yük profilleri genellikle daha değişkendir ve tüketici davranışlarına karşı hassastır. Örneğin; bir mesken kullanıcısının yükü hava koşullarına daha çok duyarlıyken, büyük bir fabrikanın yükü genellikle belirli bir çalışma programı tarafından belirlenir. Bu yük profilleri tüketicinin sınıfına göre farklılık gösterse de, yük tahmini uygulamaları piyasa etkisinin hesaplanması, meteorolojik değişkenlerin etkilerinin modellenmesi gibi ortak zorluklar ile karşı karşıya kalmaktadır.

Akıllı sayaçlar yük tahmini uygulamalarına iki şekilde olumlu katkı sağlamaktadır. İlk olarak, akıllı sayaçlar, yerel dağıtım şirketlerinin tek bir meskenin veya ticarethanenin yükünü daha iyi anlamasına ve tahmin etmesine yardımcı olur. İkincisi ise, akıllı sayaçlardan elde edilen yüksek örneklemeli yük verileri, dağıtım ve iletim hattı seviyesinde tahmin doğruluğunu artırmak için büyük bir potansiyele sahiptir. Literatürde kısa dönemli yük tahminiyle ilgili olarak birçok farklı yaklaşımlar bildirilmiştir. Ancak bu yaklaşımların çok azı tüketici seviyesinde kullanılabilecek bir yük tahmin modeli öne sürebilmiştir.

Kong'a (2019) göre, [15] ve [16] tüketici seviyesinde yük tahminine odaklanan ilk iki örnektir [17]. Chaouch [16]'da, yük tahmini için zaman serisi tahmin yaklaşımı kullanılmış ve sonuçlar günlük medyan mutlak hata (median absolute errors) cinsinden bildirilmiştir. Elde edilen sayısal sonuçlardan, yük profili kümeleme yaklaşımının dalgacık-çekirdek zaman serisi modeli ile birlikte kullanılması tahmin performansını artırdığı sonucuna varılmıştır. Ghofrani vd. [15]'da, Kalman filtresinin kullanıldığı bir tahmin modeli oluşturulmuş ve %12 ile %30 arasında değişen bir ortalama mutlak yüzde hata (Mean Absolute Percentage

Error-MAPE) değeri elde edilmiştir.

Cao vd. [18]'de, gün içi yük tahmini için özbağlanımlı tümlenik kayan ortalamalı model (Autoregressive Integrated Moving Average-ARIMA) ve benzer gün (similar day) yöntemi Çin'in Şanghay şehirdeki belirli bir bölgede kullanılmıştır. Önerilen benzer gün yönteminin mekanizması, hedeflenen günü geçmişteki meteorolojik olarak benzer günlerle gruplandırmak ve o günlerin ortalama talebine göre yükü tahmin etmektir.

Cai vd. [19]'da, bir gün önceden bina düzeyinde yük tahmini uygulaması için harici girişli mevsimsel ARIMA (Seasonal Autoregressive Integrated Moving Average with Exogenous Inputs-SARIMAX) ile özyineli sinir ağları (Recurrent Neural Networks-RNNs) ve evrişimsel sinir ağları (Convolutional Neural Network-CNN) modellerini karşılaştırmıştır. [19]'da, SARIMAX modeline kıyasla hava durumunun tüketimle güçlü kovaryansının bulunduğu durumlar için gün öncesi çok adımlı tahmin (day-ahead multi-step forecasting) hatasında ortalama %22.6'lık bir düşüş bildirmişlerdir.

Marinescu vd. [20]'de, mesken seviyesinde yük tahmini için iki farklı senaryoda altı tahmin yöntemini değerlendirilmiştir. Burada kullanılan tahmin yöntemler sırasıyla, yapay sinir ağları (Artificial Neural Networks-ANN), özbağlanımlı model (autoregressive model-AR), özbağlanımlı kayan ortalamalı model (autoregressive moving average-ARMA), ARIMA model, bulanık mantık, dalgacık sinir ağları modelidir. Elde edilen sonuçlardan, diğerlerinden açıkça daha iyi performans gösteren bir yöntem olmadığı sonucuna varılmıştır. Özbağlanımlı yaklaşımlar diğer yöntemlere kıyasla daha iyi ortalama tahmin sonuçları sağlarken, bulanık mantık, ANN ve dalgacık sinir ağları yöntemleri sabah ve akşam tüketimin fazla olduğu pik saatlerdeki tahminde daha iyi performans göstermiştir.

DL modelleri son yıllarda tüketici seviyesindeki kısa dönemli yük tahmini uygulamaları için sıklıkla tercih edilmektedir. Mocanu vd. [21]'de, bir konut binasındaki enerji tüketimini farklı zaman dilimlerinde ve farklı zaman çözünürlükleriyle tahmin etmek için aralarında RNN'nin de olduğu beş farklı yaklaşımı karşılaştırmıştır. Khatri vd. [22]'de, basit RNN, geçitli tekrarlayan ünite (Gated Recurrent Unit-GRU) ve uzun kısa süreli bellek (Long Short Term Memory-LSTM) ağlarını yük tahmin uygulamasında kullanmışlardır. İrlanda sosyal bilimler veri arşivi üzerinden temin edilen akıllı sayaç verilerinde gerçekleştirilen deneylerde, LSTM modeli diğer yöntemlere kıyasla kısa dönemli yük tahmininde daha iyi tahmin sonucu üretmiştir. Yang vd. [23]'de, olasılıksal yük tahmini uygulaması için çoklu görev Bayes DL modeli ve kümeleme tabanlı yük profili gruplandırma yaklaşımını İrlanda Enerji Düzenleme Komisyonunun ve Avustralya hükümetinin Akıllı Şebeke Akıllı Şehir (Smart Grid Smart City-SGSC) projesinden temin edilen akıllı sayaç veri kümelerine uygulamışlarıdr. Syed vd. [24]'te, öznitelik çıkarma veya boyut indirgeme yöntemlerinin performansını, ANN, LSTM ve doğrusal regresyon (Linear Regression-LR) yük tahmin modelinin kullanıldığı tahmin uygulamalarında değerlendirilmiştir. Deneysel sonuçlar, daha yüksek doğruluk ve daha hızlı eğitim süreleri için akıllı şebeke uygulamalarında veri ön işleme aşamasında uygulanan boyut indirgeme yöntemlerinin kullanışlılığını göstermektedir. Kiprijanovska vd. [25]'de, (day-ahead) gün

öncesi mesken elektrik tüketimi tahmini için HousEEC adlı ölçeklenebilir bir sistem önermişlerdir. Önerilen tahmin yöntemi, derin artıklı sinir ağına (deep residual neural network) dayanmaktadır ve hava durumu, takvim bilgisi, geçmiş döneme ait yük bilgisi gibi çoklu veri kaynaklarını kullanmaktadır. Farklı test senaryolarında, yazarlar önerilen modelden 300 mesken için kısa dönemli yük tahmininde ortalama 0.44 kWsaat'lik bir kök ortalama kare hata (Root Mean Square Error-RMSE) değeri ve 0.23 kWsaat'lik ortalama mutlak hata (Mean Absolute Error-MAE) değeri elde etmişlerdir. Kim ve Cho [26]'da, özkodlayıcıya (autoencoder) dayalı bir DL modeli kullanarak son bir saatlik tüketim verisini üzerinden 15, 30, 45 ve 60 dakikalık tüketim tahmini gerçekleştirmiştir. Shi vd. [27]'de, tüketicilerin yük profillerini tahmin uygulamasında gruplandırılmış örneklemeye (pooling) dayalı bir derin özyineli sinir ağı önermişlerdir. Önerilen DL modeli, veri çeşitliliğini ve hacmini artırarak aşırı uyumlama (overfitting) sorununu çözmeyi amaçlamıştır. Rahman vd. [28]'de ise, derin özyineli sinir ağı modelini mesken ve ticarethaneler için orta ve uzun dönemli yük tahmininde kullanmışlardır.

Jiao vd. [29]'da, LSTM özyineli sinir ağına dayalı ve zaman serisinde yer alan örnekler arasındaki kolerasyonu kullanan bir modeli sanayi tüketicilerinin yükünü tahmin etmek için kullanmışlardır. Çalışmada ayrıca, k-ortalama (k-means) kümeleme algoritması sanayi tüketicilerinin günlük yük profillerini analiz etmek ve tüketicinin enerji tüketim davranış kalıplarını sınıflandırmak için kullanılmıştır. Wang vd. [30]'da, LSTM özyineli sinir ağı ve anomali sezimine dayalı bir elektrik tüketimi tahmini yaklaşımı önermiştir. Önerilen LSTM modeli ARIMA model ile karşılaştırıldığında tahmin hatasında %22'lik bir azalma göstermektedir. Wang vd. [31]'da ise, gelecekteki yük profillerinin değişkenliğini ve belirsizliğini incelemek için LSTM tabanlı olasılıksal bir yük tahmin modeli önermişlerdir. Önerilen yöntem ile mesken tüketicileri için % 2.19 ila % 7.52 arasında, küçük-orta seviye işletmeler için ise % 3.79 ile % 25.80 arasında değişen bir iyileşme sağlanmıştır. Kong vd. [17]'de, tüketici seviyesinde yük tahmininde sıklıkla karşılaşılan oynaklık (volatility) ve belirsizlik (uncertainty) sorunlarını göz önünde bulundurarak LSTM tabanlı bir model önermişlerdir. Çalışmada önerilen LSTM yöntemiyle elde edilen en düşük tahmin hatası, % 44.06 MAPE değeriyle 12 saat geriye dönük tüketim değerinin giriş olarak kullanıldığı senaryodan elde edilmiştir. Somu vd. [32]'de, bina enerji tüketimi tahmini için iyileştirilmiş sinüs kosinüs optimizasyon algoritması (sine cosine optimization algorithm) ve LSTM özyineli sinir ağları kullanan bir model geliştirmişlerdir. Çalışmada kısa, orta ve uzun dönemli enerji tüketimini tahmin etmek için Hindistan'ın Mumbai şehrindeki Bombay Hindistan Teknoloji Enstitüsü'ndeki bir akademik binadan elde edilen enerji tüketimi verileri kullanılmıştır. Marino vd. [33]'de ise, standart LSTM ve Sequence to Sequence (S2S) mimarisini tüketici seviyesindeki yük tahmini uygulamasında karşılaştırmıştır. Her iki mimari de bir mesken tüketicisinin dakikalık ve saatlik olarak örneklenen iki farklı veri kümesinde kullanılmıştır. Çalışmada, S2S mimarisinin her iki veri kümesinde de iyi performans gösterdiği sonucu elde edilmiştir.

Vos vd. [34]'te, CNN tabanlı WaveNet mimarisinin alçak gerilim hattı seviyesinde

toplam (aggregated) elektrik yüklerinin kısa dönemli tahminleri için uygun olup olmadığını araştırmışlardır. Çalışmada yazarlar, WaveNet'in, daha yüksek model karmaşıklığına sahip olmasına karşın, bireysel tüketiciler için geleneksel makine öğrenmesi modellerine benzer veya biraz daha iyi performans gösterdiğini belirtmişlerdir. Elvers vd. [35]'de, tüketici seviyesinde kısa dönemli yük tahmini için CNN tabanlı bir tahmin modeli önermişlerdir. Önerilen model, Pecan Street veri kümesindeki 222 meskende ve farklı toplam yük seviyelerinde test edilmiştir. Çalışmada önerilen CNN tabanlı tahmin modeli, doğrusal kantil regresyon (linear quantile regression) modeline kıyasla Austin'deki sırasıyla 10, 20 ve 50 meskenin toplam yükleri üzerinde %21 ile %29 arasında değişen oranlarda daha iyi performans göstermiştir. Kim vd. [36]'da, 1-Boyutlu (1-B) CNN ve RNN modellerinin birlikte kullanıldığı bir yük tahmin modeli önermişlerdir. Önerilen model, Güney Kore'deki üç büyük dağıtım kompleksinin güç tüketim verileri üzerinde uygulanmıştır. Çalışmada önerilen modelin günlük elektrik tüketim tahmininde çok katmanlı algılayıcı (multilayer perceptron), RNN ve 1-B CNN modellerinden daha iyi performans gösterdiği belirtilmiştir. Kaligambe ve Fujita ise [37]'de, ticarethanelerin kısa dönemli yük tahmini için WaveNet mimarisi ile 1-B CNN modelini birlikte kullanan bir model önermişlerdir. Amarasinghe vd. [38]'de, tüketici seviyesindeki yük tahmini uygulamalarında CNN modeli kullanmanın etkinliğini incelemişlerdir. Önerilen model, tek bir mesken kullanıcısının elektrik tüketim verileri üzerinde test edilmiştir. Aynı veri kümesi için CNN'den elde edilen sonuçlar, S2S LSTM, faktörlü kısıtlı Boltzmann Makineleri, sığ (shallow) ANN modeli ve destek vektör makineleri (Support Vector Machines-SVM) ile elde edilen sonuçlarla karşılaştırılmıştır. Çalışmada önerilen CNN modeli, ANN ve DL metodolojileriyle yakın sonuçlar üretirken, SVM yönteminden daha iyi performans göstermiştir. Eneyew vd. [39]'da, sadece elektrik tüketim verisinin kullanıldığı veri odaklı bir tahmin modeli önermişlerdir. Bu çalışmada, artıklı dalgacık dönüşümü (Stationary Wavelet Transform) kullanılarak elde edilen öznitelikleri ve CNN modelini birleştiren derin bir sinir ağı mimarisi önerilmiştir. Önerilen yaklaşımda, dalgacık ayrıştırma (wavelet decomposition), evrişim (convolution) ve örnekleme (pooling) işlemleri uygulanarak akıllı sayaç okumalarından otomatik olarak çıkarılan öznitelikler kullanılmaktadır. Bu çalışmadan elde edilen bulgular, öznitelik çıkarımını otomatikleştirmek ve tahmin doğruluğunu iyileştirmek için dalgacık özelliklerini CNN modeliyle birleştirmenin avantajını göstermektedir.

Kim ve Cho [40]'ta, mesken enerji tüketimini etkin bir şekilde tahmin etmek için mekansal ve zamansal özellikleri çıkarabilen bir CNN-LSTM sinir ağı önermiştir. Deneyler, CNN-LSTM sinir ağının enerji tüketiminin karmaşık özelliklerini çıkarabildiğini göstermiştir. Önerilen CNN-BLSTM modelinin kısa dönemli yük tahmin uygulamasında MAPE değeri % 34.84'dir. Alhussein vd. [41]'de, CNN-LSTM hibrit modeline dayanan bir DL çerçevesi önermiştir. Çalışmada yazarlar, [17]'de elde edilen %44.06'lık bir ortalama MAPE değerine sahip LSTM tabanlı modele kıyasla, bireysel elektrik tüketimi tahmini için ortalama %40.38'lik bir MAPE değeri elde ettiklerini bildirmiştir.

Fan vd. [42]'de, kısa dönemli bina enerji tahmini için aktarmalı öğrenme (transfer learning) yaklaşımını kullanmışlardır. Çalışmada yazarlar, özerk modellerle karşılaştırıldığında, aktarmalı öğrenme tabanlı yaklaşımların 24 saatlik gün öncesi bina enerji tüketimi tahmin hatalarını yaklaşık %15 ila %78'ini azaltabildiğini bildirmişlerdir. Tian vd. [43]'de ise, aktarmalı öğrenme yoluyla halihazırda eğitilmiş modellerden yararlanarak birçok akıllı sayaç için sinir ağı tabanlı modeller oluşturmayı amaçlayan bir benzerlik tabanlı zincirleme aktarım öğrenimi (similarity-based chained transfer learning) modeli önermişlerdir. İlk model geleneksel bir şekilde oluşturularak, diğer tüm modeller ise zincir benzeri bir şekilde eğitim aşamasında elde edilen ağırlık katsayılarını aktarmalı öğrenme yoluyla bir sonraki akıllı sayaç verisinin eğitimi aşamasında kullanmaktadır. Gao vd. [44]'te, sayaçlarda eksik veri olması durumlarını da dikkate alarak S2S, dikkat katmanına (attention layer) sahip iki boyutlu (2-B) CNN ve aktarmalı öğrenme yaklaşımlarını birlikte kullanan iki farklı DL modeli önermiştir. Önerilen model, eskik veriyle eğitilen LSTM ağının sonuçlarıyla karşılaştırıldığında, S2S modelinde az miktarda veriye sahip bir mesken için tahmin doğruluğunu MAPE cinsinde % 19.69 oranında ve 2-B CNN modelinde ise ortalama % 20.54 artırmıştır. Önerilen yöntemin aynı işlevsel yapıya ve iklime sahip üç farklı bina üzerinde test edilmesi geniş ölçekli uygulama imkanını kısıtlamaktadır. Ma vd. [45]'de, sayaçlarda eksik veri olması durumları için çift yönlü veri yakıştırma (bi-directional imputation), LSTM ağı ve aktarımlı öğrenme yaklaşımlarının birlikte kullanıldığı bir model önermişlerdir. Önerilen model, bir kampüs laboratuvar binasının elektrik tüketimi verileri üzerinde test edilmiştir. Çalışmada yazarlar, önerilen modelin farklı veri kayıp oranlarındaki tahmin sonuçlarında % 4.24 ila % 47.15 daha düşük RMSE değeri elde ettiklerini belirtmişlerdir. Liu vd. [46]'da ise, bina enerji tüketimini tahmin etmek için yaygın olarak kullanılan üç derin pekiştirmeli öğrenme (deep reinforcement learning) yöntemini (asenkron avantaj aktör-kritik (A3C), derin deterministik politika gradyan (deep deterministic policy gradient) ve özyineli deterministik politika gradyan (recurrent deterministic policy gradient)) bir ofis binasının elektrik tüketim verisine uygulamışlardır. Çalışmada yazarlar, MAE cinsinden hesaplanan tahmin performanslarının tek adımlı tahmin (single-step ahead prediction) için %16-24 oranında ve çok adımlı tahmin (multi-step ahead prediction) için ise %19-32 oranında iyileştirebildiğini bildirmişlerdir.

Kısa dönemli yük tahmini uygulamalarında farklı veri ön işleme yöntemleri tahmin performansını artmak için sıklıkla tercih edilmektedir. En sık kullanılan yöntemlerin başında yük profili kümeleme yöntemleri gelmektedir [47]–[53]. Bu kümeleme yöntemleri sayesinde tüketici seviyesindeki yük tahmini performansını etkileyen dışsal faktörler ve insan aktivitelerinden kaynaklanan belirsizlikler gruplandırılarak tüketim örüntüleri tespit edilmektedir [54]–[57]. Enerji zaman serisi tüketimini tahmin etmek için mevcut makine öğrenme yöntemlerinin kapsamlı bir incelemesini [58]'da sunulmuştur.

Kayıp-kaçak Tespiti Uygulamaları: Literatür, kayıp-kaçak kullanımının hem dağıtım şirketi tarafında hem de tüketiciler tarafında meydana gelen önemli ekonomik ve

toplumsal çeşitli sonuçlarını bildirmektedir [59], [60]. Kayıp-kaçak kullanımının ekonomik sonuçlarının en önemlileri arasında, dağıtım şirketinin yıllık gelir kaybı ve normal tüketiciler için elektrik fiyatlarındaki artışlar yer almaktadır. Ayrıca, dağıtım sistemlerinde sıklıkla meydana gelen güç kesintileri veya şebekede kararsızlık sorunları da gözlenmektedir. Dağıtım şirketleri açısından, kayıp-kaçak kullanımı, altyapı iyileştirme çalışmaları yoluyla ele alınabilecek teknik bir tasarım sorunudur. Akıllı sayaçlar gibi akıllı şebekede meydana gelen son teknolojiler, bu sorunların üstesinden gelmek için yeni fırsatlar oluşturmuştur. Akıllı şebeke, geleneksel ve yeni teknolojiler aracılığıyla elektrik tedarik zincirinin akıllı kontrolünü sağlarken, akıllı sayaçlar gelişmiş yüksek çözünürlüklü veri toplama yeteneğini kullanarak kayıp-kaçak tespitinde anlık durum bilgisi paylaşmaktadır.

Siber saldırılar ve fiziksel olarak gerçekleştirilen sayaç müdehaleleri (physical meter tampering), sayaç okumalarını ve diğer hizmetleri birçok farklı şekilde manipüle edebilmektedir. Bunlardan bazıları, sayaç yazılımına yapılan müdehaleler, sahte veri yerleştirme (False Data Injection-FDI), tüketicilerin şebekeyle bağlantısını kesme veya yardımcı sistem operasyonlarını kesintiye uğratma gibi farklı yaklaşımlardır. Fiziksel sayaç müdehaleleri, akıllı sayaçların fiziksel bileşenlerini doğrudan manipüle etmeyi amaçlar ve bu müdehaleler, sayaçta yer alan bilgilendirme kodları vasıtasıyla uzman teknisyenler tarafından ancak yerinde inceleme ve tespit ile belirlenebilmektedir. Fiziksel müdehaleler, tüketiciler tarafından kolayca gerçekleştirilebildiği için literatürde karşılaşılan çalışmaların büyük çoğunluğu bu konuları incelemektedir. Ancak FDI türü siber saldırılar, tüketiciler tarafından gerçekleştirilmesi daha zor uygulamalar olduğu için literatürde konuyla alakalı olarak karşılaşılan çalışmaların sayısı daha azdır.

Günümüzde kayıp-kaçak tespitinde kullanılan son teknolojik gelişmeler, [9] ve [60] dahil olmak üzere çok sayıda araştırmada sunulmuştur. Literatürde kayıp-kaçak tespit uygulamalarındaki geleneksel yaklaşım, şebeke topolojisine dayalı yük dengesi ölçümlerini gerektirmektedir. Ancak şebeke topolojisi, günümüzde iletim ve dağıtım sistemlerinin genişlemesi nedeniyle sürekli ve düzenli olarak gelişmektedir. Son yıllarda ise akıllı şebekede kayıp-kaçak tespitinde kullanılan uygulamalar, büyük hesaplama kapasiteleri kullanarak yapay zeka ve makine öğrenmesi modelleri geliştirmeye odaklanmaktadır [61]–[63]. Literatürde, dolandırıcılık tespiti (fraud detection), olağandışılık sezimi (anomaly detection), izinsiz şebeke bağlantıları (illicit connections) tespiti gibi farklı yöntemleri belirleyebilmek için farklı makine öğrenmesi stratejilerine (denetimli, denetimsiz ve yarı denetimli öğrenme) dayalı çeşitli çalışmalar incelenmiştir. Makine öğrenmesine dayalı yaklaşımlar, kayıp-kaçak kullanımı ile yüksek oranda ilişkili olduğu bilinen anormal tüketim davranışlarını tespit etmek için tüketicilerin yük profilleri hakkındaki bilgileri kullanmaktadır. SVM, ikili sınıflandırma (binary classification) için yaygın olarak kayıp-kaçak problemlerine uygulanmaktadır [64]–[67]. Ayrıca ANN modeli de, kayıp-kaçak tespiti için ikili sınıflandırıcı olarak yaygın olarak kullanılmaktadır [68]–[70]. Son yıllarda ise DL modelleri, kayıp-kaçak tespiti uygulamalarında umut verici

sınıflandırma performansı ve yüksek tahmin doğruluğu elde etmektedir [71], [72].

Akıllı şebekede haberleşme ağı üzerinden veri aktarımına olan yüksek bağımlılık, sistemin siber saldırılara karşı savunmasızlığını artırmaktadır. Siber saldırılar, veriler şebeke haberleşme ağı üzerinden iletilirken, akıllı sayaç hafızasında veya merkez veri tabanına kaydedilirken meydana gelebilir. Saldırıların bazıları hizmet kesintisine, bazıları altyapı hasarına, bazıları ise enerji ve bilgi hırsızlığına neden olabilmektedir. Kayıp-kaçak kullanımına yol açan siber saldırılar, akıllı sayaç okumalarına olan güveni tehlikeye atarak dağıtım şirketinin yanlış faturalandırma oluşturmasına yol açmaktadır. Kayıp-kaçak kullanımında siber tabanlı kullanılan birincil yöntem, sayaç verilerine FDI yöntemlerinin uygulanmasıdır. Bu yöntemde, bir saldırgan veya birden çok saldırgan, enerji tüketimini azaltmak veya artırmak için sisteme saldırı vektörlerini ekleyerek akıllı sayaç verilerini manipüle edebilmektedir. FDI tabanlı kayıp-kaçak kullanımının uygulamaları, matematiksel altyapısı ve akıllı şebekede meydana gelen olumsuz sonuçları [73]–[76] arasındaki literatür araştırmalarında detaylı olarak incelenmiştir.

McLaughlin vd. [77]'de, Gelişmiş Sayaç Altyapısında (Advanced Metering Infrastructure-AMI) kayıp-kaçak kullanımını tespit etmek ve sensör bilgilerini akıllı sayaçtan gelen tüketim verileri ile birleştirmek için bilgi tümleştirme (information fusion) yaklaşımını kullanmışlardır. Önerilen sistem, enerji hırsızlığı ile ilgili davranışları daha doğru bir şekilde modellemek ve tespit etmek için fiziksel ve siber olayların sayaç kontrol tarihçesini (audit log) tüketim verileriyle birleştirmektedir. Zhang vd. [78]'de, dağıtım sistemlerinde FDI yöntemlerini tespit etmek için veri odaklı ve DL tabanlı bir algoritma önermektedir. Çalışmada, akıllı sayaç veri kümelerinde boyut indirgeme (dimension reduction) ve öznitelik çıkarma (feature extraction) işlemleri için çekişmeli üretici ağı (Generative Adversarial network-GAN) tabanlı bir özkodlayıcı (autoencoder) model geliştirilmiştir. Ayrıca çalışmada önerilen model, gerçek durumda güç sistemlerinden toplanan veri kümelerinde sıklıkla karşılaşılan eksik veya kısmen etiketleme (labeling) sorununu dikkate alınarak, etiketlenmemiş veri kümeleriyle de eğitilebilmektedir. Çalışmada yazarlar önerilen modelin, Elektrik ve Elektronik Mühendisleri Enstitüsü'nün (Institute of Electrical and Electronics Engineers-IEEE) 13 ve 123 baralı iki farklı güç sisteminde FDI senaryolarını sırasıyla % 96.25 ve % 97.85 doğrulukta tespit ettiğini bildirmişlerdir. Li vd. [79]'da ise, benzer şekilde FDI saldırılarını IEEE'nin 30 ve 118 baralı iki farklı güç sisteminde durum kestirimi (state estimation) amacıyla gerçekleştirmişlerdir. Çalışmada önerilen siber-fiziksel model, ideal ölçümleri yakalayan fiziksel bir modeli, ideal ölçümlerdeki sapmaları yakalayan GAN tabanlı bir veri modeliyle birleştirmektedir. Pu vd. [80]'de, FDI tabanlı siber saldırılara ait öznitelikler çıkarmak için derin inanç ağı'na (Deep Belief Network-DBN) dayalı saldırı tanımlama mekanizması önermişlerdir. Leite ve Mantovani [81]'de, ölçülen varyansı izlemek için güvenilir bir bölge oluşturan çok değişkenli (multivariate) bir kontrol çizelgesi kullanarak kayıp-kaçak tespiti için bir strateji önermiştir. Çalışmada önerilen yöntem, kayıp-kaçak tespiti sonrasında A-Star (A*) arama algoritmasına dayalı bir yol bulma (pathfinding) prosedürü ile

kayıp-kaçak noktası ile tüketim noktasını belirleyebilmektedir. Ayrıca bir coğrafi bilgi sistemi (CBS) uygulaması sayesinde, siber saldırının hedefi olan tüketim noktasını görüntülenebilmektedir. Liu ve Hu [82]'de ise, dağıtım şebekesine bağlı akıllı evlerden oluşan bir toplulukta elektrik hırsızlığı amacıyla gerçekleştirilen siber saldırıları tespit etmek amacıyla Bollinger bandı ve kısmen gözlenebilir Markov karar süreci (partially observable Markov decision process) yaklaşımının birlikte kullanıldığı bir model önermiştir. Çalışmada yazarlar aynı zamanda, Markov sürecinin karmaşıklığını azaltmak ve siber saldırı tespitinin verimliliğini artırmak amacıyla kanı uzayı (belief state) azaltma tabanlı uyarlamalı bir dinamik programlama yöntemi geliştirmişlerdir. Yazarlar simülasyon sonuçlarını dikkate alarak, önerilen yöntemin ortalama olarak %92.55 oranında kayıp-kaçak kullanımını başarılı bir şekilde tespit ettiğini bildirmişlerdir. Liu vd. [83]'te, akıllı bir sayaçtaki birimler arasındaki bilgi akışını tanımlamak için renkli Petri ağını kullanırken, aynı zamanda akıllı sayaçlar için bir tehdit modeli önermişlerdir. Çalışmada yazarlar, bir akıllı sayacın kısıtlı hesaplama ve depolama kaynaklarını göz önünde bulundurarak FDI tabanlı kayıp-kaçak kullanımına karşı bir saldırı tespit mekanizması sunmuşlardır.

Büyük Veri Uygulamaları: Akıllı şebekede kısa dönemli yük tahmini, enerji yönetimi, sistemin işletilmesi ve piyasa analizi için oldukça önemlidir [6]. Akıllı şebekede tüketicilerin artan aktif rolü ile yüksek verimli dinamik elektrik piyasası da aynı zamanda iyi bir elektrik tüketimi tahmini performansına dayanmaktadır. Ayrıca, büyük ölçekli kayıp-kaçak kullanımı, güç sistemlerinde ciddi dengesizlik sorunlarına neden olmaktadır. Bu nedenle, karmaşık iletim ve dağıtım sistemlerinde kayıp-kaçak kullanımını tespit etmek için etkili bir Büyük Veri çerçevesi oluşturmak dağıtım şirketlerinin öncelikli amaçlarından birisidir. Bu bölümde kısa dönemli yük tahmini ve kayıp-kaçak tespiti uygulamaları hesaplama paradigması dikkate alınarak incelenmiştir. Bu açıdan değerlendirildiğinde literatürde önerilen yöntemlerin ölçeklenebilirlik kriteri dikkate alınarak incelenmesi geniş ölçekli akıllı sayaç uygulamaları için önemlidir. Akıllı şebeke verileri yalnızca büyük miktarda akıllı sayaç okuma verilerini, tüketici verilerini, haberleşme ağı verilerini değil, aynı zamanda internet, hava durumu, CBS, nüfus ve akıllı şehir verileri gibi farklı kaynaklardan gelen büyük miktardaki verileri de içermektedir. Zhou vd.'nin [84]'te de belirtildiği üzere, enerji yönetimindeki büyük veri zorlukları, bilişim teknolojileri altyapısı, veri toplama ve yönetim, veri entegrasyonu ve paylaşımı, veri işleme ve analizi, güvenlik ve akıllı enerji yönetimi olmak üzere altı temel konuya odaklanmaktadır [85].

Liu vd. [86]'da, yük tahmini yapılacak coğrafi alanı yerel hava durumu bilgilerine göre bölerek dağıtık yük tahmini uygulaması için bir Map-Reduce programlama çerçevesi önermişlerdir. Zhang vd. [87]'de ise, Büyük Veri teknolojilerine dayalı bir kısa dönemli yük tahmini çerçevesi önermiştir. Önerilen kısa dönemli yük tahmini çerçevesi, 1.2 milyon tüketiciden oluşan gerçek bir dağıtım sisteminden toplanan akıllı sayaç verileri kullanılarak doğrulanmıştır. Bu çalışmada önerilen kısa dönemli yük tahmini çerçevesi, 5 adet veri işleme sunucusu kullanılarak Hadoop platformu üzerinde uygulanmıştır. Oprea

ve Bâra [88]'de, akıllı sayaçlardan ve hava durumu sensörlerinden gelen verileri toplayarak veri ön işleme yapabilen ve daha sonra ise büyük hacimli heterojen verileri bir NoSQL veritabanında depolayabilen ölçeklenebilir bir Büyük Veri çerçevesi önermişlerdir. Alemazkoor vd. [89]'da, yük tahmini uygulamasında 3.7 milyon tüketici için yüksek boyutlu akıllı sayaç verilerinin entegrasyonunu sağlamak amacıyla hiyerarşik boyut indirgeme (dimension reduction) algoritmasından yararlanan bir indirgenmiş model (reduced model) yaklaşımı önerilmiştir. Deng vd. [90]'da, Hadoop MapReduce hesaplama paradigmasını kullanarak hibrit bir kısa dönemli yük tahmin modelinin nasıl oluşturulabileceğini açıklamaktadır. Önerilen modelde ilk aşamada, orijinal geçmiş yük verilerinden farklı özniteliklere sahip bir dizi modal fonksiyon bileşeni elde etmek için varyasyonel kip ayrıştırıcı (variational mode decomposition) yöntemi kullanılmış ve daha sonra ise, Hadoop MapReduce mimarisinden yararlanılarak, varyasyonel kip ayrıştırıcı tarafından oluşturulan her bir yük bileşenini tahmin etmek için dağıtık DBN modeli oluşturulmuştur. Zainab vd. [91]'de, yük tahmini uygulaması için 8 düğüme (node) kadar iş yükünü paralelleştirebilen ve Apache Spark kütüphanesinde yer alan regresyon yöntemlerini kullanabilen bir Büyük Veri çerçevesi oluşturmuşlardır. Çalışmada yazarlar, hesaplama süresinin yalnızca veri boyutuna bağlı olmadığı, aynı zamanda hesaplama düğümlerinin sayısına ve işi yürütmek için atanan çekirdek sayısına da bağlı olduğunu belirtmektedir. Dong vd. ise [92]'de, Amazon Web Servisi (AWS S3), Amazon EMR, MongoDB ve Apache Spark kullanarak mesken tüketicilerinden elde edilen toplam 10 GB'lık bir akıllı veri kümesinde, Rastgele Orman (Random Forest-RF) makine öğrenmesi algoritmasını kullanarak gelecek dönem enerji tüketimini tahmin etmiştir. Çalışmada yazarlar, büyük ölçekli veri kümeleri üzerinde makine öğrenmesi algoritmalarını dağıtık sistemler ile birlikte kullanmanın önemli hesaplama avantajları sağladığı sonucunu çıkarırken, işlenen veri miktarı dağıtık sistemler tarafından sağlanan hesaplama verimliliklerinden yararlanabileceği bir eşiğin altında olduğunda, dağıtık sistemlerin hesaplama açısından külfetli olabileceği sonucuna varmıştır.

Nagi vd. [93]'te, Malezya'daki en büyük elektrik kuruluşu olan Tenaga Nasional Berhad için elektrik hırsızlığını azaltmak amacıyla SVM tabanlı bir kayıp-kaçak modeli geliştirmişlerdir. Önerilen sistem, Microsoft Visual Basic 6.0 kullanılarak bir grafik kullanıcı arayüzü (Graphical User Interface-GUI) olarak geliştirilmiş ve hesaplamalar, yerel olarak bir Dell PowerEdge iş istasyonunda çalıştırılmıştır. Önerilen model ile kayıp-kaçak sınıflandırma sonuçlarının elde edilmesi için geçen süre tüketici başına yaklaşık 1.8 saniyedir ve iş istasyonunun yapılandırma ayarlarına göre değişkenlik göstermektedir. Jokar vd. [94]'de, İrlanda Akıllı Enerji Girişimi (Irish Smart Energy Trial) tarafından oluşturulan 5000'den fazla İrlandalı ev ve işyerinin yer aldığı halka açık bir veri kümesini kullanmışlardır. Çalışmada FDI örnekleri dağıtım transformatörleri kullanılarak sentetik olarak oluşturulmuştur. Önerilen algoritma, düşük örnekleme oranlarında çalıştırılmak üzere tasarlanmıştır. Buzau vd. [95]'te, akıllı sayaçlardaki anormallikleri ve kayıp-kaçağı tespit etmeye yönelik olarak hibrit bir DL modeli önermişlerdir. Çalışmada

yazarlar, İspanya'nın en büyük elektrik kuruluşu olan Endesa'nın gerçek akıllı sayaç verilerini kullanmışlardır. PySpark, tüm veri ön işleme adımları için dağıtık hesaplama seçeneği olarak tercih edilmiştir. Yazarlar, önerilen metodolojinin dağıtım şirketinde kayıp-kaçak tespitine yardımcı bir program olarak yaklaşık % 47'lik bir doğrulukla yeni sayaç müdehalelerini tespit etmek için kullanıldığını bildirmiştir. [96]'de yazarlar, kayıp-kaçak tespit uygulamasının iş yükünü paralelleştirmek için ağ sınırında bilişim (edge computing) çerçevesi geliştirmişlerdir. Çalışmada yazarlar, önerilen çerçeve için Eşle-İndirge (Map-Reduce) fonksiyonları oluşturmuş ve daha sonra her bir alt işlem, yerel olarak çözülmek üzere her akıllı sayacın mikroişlemcisine atanmıştır. Önerilen çerçevede ağ sınır aygıtı ana düğüm (master node) olarak, akıllı sayaçlar ise işçi düğüm (slave node) olarak çalışmaktadır. Lipčák vd. ise [97]'de, akıllı sayaç verilerinde anomali sezimi (anomaly detection) için Büyük Veri çerçevesi oluşturmuşlardır. Önerilen çerçeve, veri sıklaştırma (data densification) seçeneklerine sahip bir veri giriş katmanını, akış işleme için Apache Flink ve veri depolama katmanı olarak da Hadoop dağıtık dosya sistemi (Hadoop Distributed File System-HDFS) ve KairosDB altyapısını kullanmaktadır.

Bu bölümde bahsedilen literatür çalışmalarından da anlaşılacağı üzere, son yıllarda AMI sistemler için geliştirilen yük tahmini ve kayıp-kaçak tespiti uygulamalarının çoğunluğu makine öğrenmesi ve veri analitiği temelli yaklaşımlarını kullanmaktadır. Ayrıca, tüketici seviyesinde gerçekleştirilen yük tahmini ve anomali tespiti gibi işlemlerin Büyük Veri teknolojileri ile birlikte kullanılması geniş ölçekli olarak modellenebilmesine imkan sağlamaktadır.

1.3. Hedefler ve Literatüre Sağlanan Katkılar

Bu tez çalışmasında geliştirilen yük tahmini ve kayıp-kaçak tespit modelinin AMI sistemlerde gerçekleştirilen günlük operasyonel işlemlere katkı sağlaması öncelikli hedef olarak belirlenmiştir. Akıllı şebeke uygulamalarında karşılaşılan ölçeklenebilirlik probleminin çözümüne katkı sağlamak amacıyla Büyük Veri teknolojilerini kullanan modellerin araştırılması ve uygulanması ise ikinci hedeftir. Bu kapsamda geliştirilen modellerden elde edilecek elektrik tüketim örüntülerini dağıtım şirketlerinin araştırma-geliştirme (Ar-Ge) faaliyetlerinde referans bir bilgi kaynağı olarak kullanabilmesini sağlamak amacıyla tasarım ve uygulama detaylarına herkesin ulaşabilmesi ise üçüncü hedeftir.

Bu tez çalışmasında geliştirilen kısa dönemli yük tahmin modelinin literatüre sağladığı katkılar aşağıda listelenmiştir:

- Kısa dönemli yük tahmin uygulamalarında yoğunluk tabanlı aykırılık analizinin tahmin performansına olan katkısı kaspamlı bir şekilde incelenmiştir.
- Gelişmiş veri ön işleme yöntemleri, optimizasyon ve hibrit DL modelinin birlikte kullanıldığı bir çerçeve tasarlanmıştır.

- Analizler hem yerel dağıtım şirketinden elde edilen akıllı sayaç veri kümeleriyle hem de halka açık büyük hacimli veri kümeleriyle doğrulanmıştır.
- Yük tahmin uygulamasından elde edilen çıktılar, kullanıcı ihtiyaçları göz önünde bulundurularak kompakt ve yeniden üretilebilir bir formatta sunulmuştur.

Bu tez çalışmasında geliştirilen kayıp-kaçak tespit modelinin literatüre sağladığı katkılar aşağıda listelenmiştir:

- Tüketici yük profillerinin parametrelerini ve özniteliklerini dikkate alan yeni, kapsamlı bir kayıp-kaçak tespit modeli geliştirilmiştir.
- Geliştirilen DL tabanlı sınıflandırıcı model Büyük Veri teknolojileri ile entegre edilmiştir.
- Kayıp-kaçak tespitinde siber saldırıların kullanıldığı senaryoların kapsamlı bir uygulaması gerçek dağıtım şebekesinde belirlenen pilot bir trafo istasyonunda gerçekleştirilmiştir.
- Kayıp-kaçak tespitinde kullanılabilecek önemli ve sağlam istatistiksel özniteliklerin kapsamlı bir araştırması ve uygulaması gerçekleştirilmiştir.

Tez kapsamında gerçekleştirilen yük tahmini ve kayıp-kaçak tespit uygulamalarında gerçek hayatta karşılaşılan kısıtlamalar dikkate alınarak az sayıda değişken kullanılmıştır. Bu tez çalışmasında, akıllı sayaç veri kümelerinin temin edilmesi ve analiz edilmesi aşamasında karşılaşılan veri gizliliği meseleleri için çözüm önerileri geliştirilmiş ve genel bir veri toplama (data acquisition) çerçevesi tasarlanmıştır. Ayrıca, sayaç veri kümelerini yönetme, uygun formata dönüştürme ve depolama işlemleri için açık kaynaklı ve ticari olarak ölçeklendirilebilen sistemlerin uygulanabilirliği incelenmiştir.

1.4. Tezin Organizasyon Yapısı

Bu tez çalışması beş bölümden oluşmaktadır. Bölüm 1.'de akıllı şebekede yük tahmini, kayıp-kaçak tespiti ve Büyük Veri konsepti hakkında genel bir değerlendirme yapılmıştır. Bölüm 2.'de AMI sistemlerin temel bileşenleri ve tez kapsamında kullanılan Büyük Veri teknolojileri detaylı olarak açıklanmıştır. Bölüm 3.'te tez kapsamında kullanılan veri kümeleri, dağıtık hesaplama ortamının kurulumu ve veri analiz yöntemlerinin matematiksel altyapısı sunulmuştur. Bölüm 4.'te tez çalışmasında vurgulanan en önemli bulgu ve sonuçlarla bu tezin yayınlarının bir özeti bulunmaktadır. Bölüm 5.'te ise bulgular ve tartışma bölümünde verilen sonuçların genel bir değerlendirmesi ile gelecekteki olası araştırma konuları hakkında öneriler sunulmuştur.

2. Akilli Şebekeler ve Büyük Veri Teknolojisi

Enerji sektöründe meydana gelen liberalleşme süreci elektrik dağıtım şirketlerinin özelleştirilmesinden tedarik ve dağıtım işlemlerinin birbirinden ayrışmasına kadar birçok süreci etkileyen zincirleme bir reaksiyonu başlatmıştır. Bu süreçte elektrik şebekesinin en önemli unsurlarından olan sayaçlar ve diğer ölçüm sistemleri de bu gelişmelerden payını almıştır.

Akıllı Şebeke terimi, günümüzde teknik bir tanımdan ziyade pazarlama terimi olarak kullanılmaktadır. Bu nedenle, *akıllı* olarak tanımlanan şeylerin ne olup ne olmadığı konusunda net ve genel kabul görmüş bir kapsam bulunmamaktadır. Uluslararası Elektroteknik Komisyonu (International Electrotechnical Commission-IEC)'nun alt komitesi olan SyC Smart Energy, akıllı şebekeleri elektrik şebekesini modernize etme çabalarının bir bütünü olarak tanımlamaktadır [98]. Bu açıdan değerlendirildiğinde, akıllı şebekeler, elektrik ve bilgi teknolojilerini herhangi bir üretim noktası ile herhangi bir tüketim noktası arasında bir araya getirmeye katkı sağlamaktadır. Bu bölümde, akıllı şebekede AMI sistemler ve alt bileşenleri ile Büyük Veri teknolojileri detaylı olarak açıklanmıştır.

2.1. Akıllı Şebekelerde Gelişmiş Sayaç Altyapısı

Akıllı sayaç sistemleri tüketim yapılan noktalardaki ölçümleri merkezi veri sistemine aktaran yazılım, donanım ve iletişim kanallarından oluşan bir altyapı olarak tanımlanmaktadır. Ayrıca tek ve çift yönlü haberleşme protokollerine bağlı olarak da iki farklı şekilde tanımlanabilir. Tek yönlü haberleşme protokolünü kullanan Otomatik Sayaç Okuma Sistemleri (OSOS) yalnızca sayaç verilerinin merkezi sisteme iletildiği sistemlerdir. Çift yönlü haberleşme protokolünü kullanan AMI sistemler ise tüketim değerlerini merkezi sistem ile paylaşmanın yanı sıra uzaktan erişim, gelişmiş tarifelendirme seçenekleri ve ev-içi gösterim cihazları vasıtasıyla tüketim verilerini görselleştirme gibi ek imkanlar sunan sistemlere verilen genel bir tanımlamadır. AMI genel olarak akıllı sayaçlardan, veri yönetim sistemlerinden ve iletişim ağlarından oluşan bütünleşik bir sistemdir.

Tek yönlü haberleşme kullanan sistemlerde; sayaca yapılan dış müdahale bilgisi, elektrik kesintisi, sayaç karakteristik bilgisi, endeks, yük profili gibi veriler, öncelikle sayaç veri merkezine daha sonra ise merkezi veri sistemine gönderilmektedir. Çift yönlü haberleşme kullanan sistemlerde ise, sayaçların verileri periyodik olarak gönderilebilmesi önceden programlanabilmekte ve aynı şekilde sayaçtan veri çekilebilmesi için komutlar verilebilmektedir. Çift yönlü haberleşme kullanan sistemlerde veri okuma işlemlerinin dışında ayrıca uzaktan elektrik kesme-açma işlemleri de yapılabilmektedir.

AMI sistemleri akıllı şebeke tasarımında önemli bir temel bileşendir. AMI sistemi, çift yönlü haberleşme teknolojisini kullanarak akıllı sayaçlardan veri toplayan ve analiz eden daha sonra ise bu verilere dayalı olarak dağıtım seviyesinde çeşitli uygulama ve hizmetlerin akıllı yönetimine fayda sağlamaktadır. AMI uygulamaları ise, elektrik şebekesi kontrol

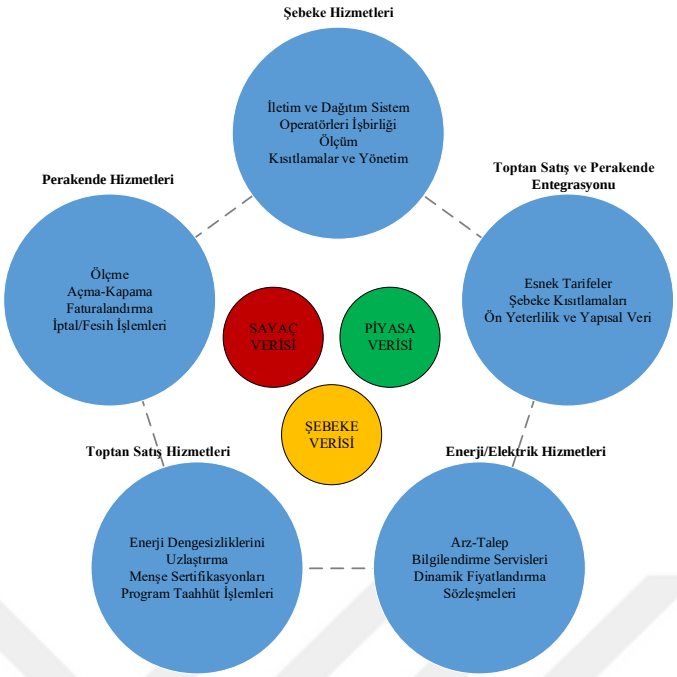
sistemlerinin dijitalleştirilmesinin ilk aşaması olarak değerlendirilmektedir. AMI sistemlerde elektrik sayaçlar kritik önem taşıyan erişim noktalarında ve hizmet sağlayıcıların iletişim sistemleri ağında verileri ölçmek, toplamak, yönetmek ve analiz etmek için veri yönetim sistemleri gibi ortamlarda gaz ve ısı sayaçları ile birlikte çalışmaktadır. AMI sistemlerde kullanılan veri türleri ve elektrik piyasası modelleme süreçleri Şekil 2.1.'de verilmiştir.

AMI sistemlerin öne çıkan bazı avantajları aşağıda listelenmiştir:

- Şebeke bağlantıları içerisinde en kapsamlı ağ bağlantısına sahip son teknoloji ekipmanlar içermektedir.
- Akıllı sayaçlar sayesinde, sistem koşullarını (örneğin Güç Kalitesi (Power Quality-PQ) olayları dahil) tüketici düzeyine kadar kapsamlı bir şekilde ölçmeye imkan sağlar.
- Tüketici seviyesinde yapılan analizler yerel olarak kontrol edilirken (örneğin; gerçek zamanlı tarifelendirme seçenekleri gibi), dağıtım seviyesinde gerçekleştirilen operasyonlar, AMI bilgilerini işleyerek sistem ve bölgesel düzeyde kontrol işlemlerini gerçekleştirmektedir.
- AMI sistemler, dağıtık enerji kaynakları ile kolay entegre olabilen bir yapıya sahip olduğu için yenilenebilir enerji sistemleriyle uyumlu çalışır ve haberleşme ağı maliyetlerini azaltmaya yardımcı olur.
- AMI tüketici portalları, ev alanı ağları (Home Area Network-HAN) ve ev içi gösterim ekranları, kullanıcılar için kullanışlı arabirimler oluşturarak tüketici karar verme sürecini desteklemektedir. Dağıtım şirketlerinin operasyon merkezlerindeki karar verme süreçlerinde AMI tarafından ek bilgiler sağlanır.

Yukarıda kısaca özetlenen avantajların dışında AMI sistemlerinde karşılaşılan dezavantajlar ise aşağıda listelenmiştir:

- AMI sistemlerin sağladığı faydaları değerlendirirken yalnızca dağıtım şirketleri operasyonlarıyla sınırlandırmak, iş tanımının ve gerekçelerinin daha geniş perspektifte ele alınmasını önyargılı bir hale getirmektedir.
- AMI haberleşme ağı modernizasyonu, veri yönetim sistemi entegrasyonu ve diğer sabit kurulum maliyetleri, ortalama olarak toplam sayaç başına maliyetin yarısından fazlasını oluşturmaktadır. Şehir düzeyinde ve kısmi ölçekli uygulamalar genellikle pilot ölçekli uygulamalardan sayaç başına daha düşük bir toplam maliyete sahiptir. Bu maliyetler, uygulanan projenin kapsamına da bağlı olarak her dağıtım şirketi için değişiklik göstermektedir.
- Haberleşme ağı modernizasyonu sadece AMI sistemleri için değil aynı zamanda akıllı şebeke konseptindeki diğer işlevleride destekleyecek farklı tasarım problemlerini de beraberinde getirmiştir. Sürece sonrasında dahil edilen bu gereksinimler bazı dağıtım



Şekil 2.1. Veri Türleri ve Elektrik Piyasası Modelleme Süreçleri

şirketleri için toplam maliyeti artırsa da, şirketlerin yatırım hisselerini de artırmış ve gelecekteki şebeke modernizasyonu çalışmaları için zemin hazırlanmasına yardımcı olmuştur.

- AMI sistemlerden en uygun seviyede performans elde edebilmek için akıllı sayaçlarda, kullanıcı cihazlarında, haberleşme ve bilgi sistemlerinde kullanılan veri formatlarında uluslararası düzeyde belirlenmiş ve kabul görmüş standartlara ihtiyaç duyulmaktadır.
- Güçlü siber güvenlik sistemlerine duyulan ihtiyaç ve müşteri gizliliği gibi meseleler, AMI sistemlerin uygulama kapsamı arttıkça dağıtım şirketleri için önemli bir odak noktası haline gelmektedir.

2.1.1. Akıllı Sayaçlar

Akıllı sayaçlar tüketici ile sistem operatörleri arasındaki siber, fiziksel ve sosyal etkileşimi sağlayan elektrik şebesinin temel bir ekipmanıdır. AMI sistemlerin veri toplama yetkinliğine katkı sağlayan ve temel işlevi tüketim noktasında kullanılan enerjiyi ölçmek olan bir ölçüm aletidir. Akıllı sayaçlar fiziksel özelliklerine göre farklı şekillerde sınıflandırabilmektedir. Teknolojik gelişmeler dikkate alındığında mekanik ve elektronik sayaçlar, bağlantı yapıldığı tüketim noktası dikkate alındığında ise tez fazlı veya üç fazlı sayaçlar olarak sınıflandırılmaktadır. Bunların yanısıra yüksek güç çeken tüketicilerin kullandığı sayaçlar akım ve gerilim trafoları ile birlikte kullanılabilmektedir.

Akıllı sayaçlar tüketicilerin ve sistem operatörlerinin ihtiyaçları dikkate alınarak farklı şekillerde tasarlanmaktadır. Akıllı sayaçları geleneksel mekanik sayaçlardan ayıran

en önemli tasarım detaylarından birkaçı ise veri kaydetmek için kullandığı bellek ünitesi ve haberleşme portları vasıtasıyla sistem dışına veri aktarabilme yeteniğidir.

Tek fazlı bir akıllı sayaç tüketilen aktif enerjiyi hesaplayabilmek için akım ve gerilim değerlerini farklı kanallar üzerinden eş zamanlı olarak ve yüksek bir örnekleme frekansı kullanarak sayısal işaretlere dönüştürür. Sonraki aşamda ise yüksek hızlı bir mikrokontrolör bu sayısal işaretlerden tüketilen enerji değerlerini hesaplar ve sonuçları sayaç ekranına aktarır. Bu işlem geleneksel elektromekanik sayaçlarda ise tüketilen toplam enerji miktarını hesaplabilmek için ayda bir defa yapılmaktadır. Akıllı sayaçların öne çıkan diğer özellikleri ise: dinamik fiyatlandırma yapmak, dağıtım şirketi ve tüketici için veri kaydetmek, en az hatayla ölçüm yapmak, güç kaybı veya sistem bakım-onarım gibi durumlar hakkında bildirim göndermek, uzaktan kesme-açma işlemleri yapmak, arz-talep veya gecikmiş ödemeler gibi durumlar için yük sınırlaması getirmek, erken ön ödeme işlemleri yapmak, güç kalitesini izlemek, sayaca yapılan dış müdahale veya enerji hırzılğı tespiti için güvenlik protokolleri kullanmak ve binalardaki diğer akıllı cihazlarla haberleşebilmek, vb. şeklinde özetlenebilir.

Ülkemizde kullanılan akıllı sayaçların yukarıda bahsedilen hizmetleri sorunsuz bir şekilde yerine getirebilmesi için Türkiye Elektrik Dağıtım Anonim Şirketi (TEDAŞ) Strateji Daire Başkanlığı tarafından hazırlanan TEDAŞ-MLZ/2017-062.A numaralı Elektronik Elektrik Sayaçları Teknik Şartnamesi'ne ve Enerji Piyasası Düzenleme Kurumu'nun (EPDK) Elektrik Piyasasında Kullanılacak Sayaçlar Hakkında Tebliğine uygun olarak tasarlanması gerekmektedir. Bu şartname ve tebliğ dağıtım sistemi seviyesinde kullanılacak elektronik elektrik sayaçların teknik detayları, yazılım gereksinimleri, haberleşme protokolleri ve veri formatları hakkında kapsamlı açıklamalar içermektedir. Akıllı sayaçların çalışma şartlarına uygun olacak şekilde tasarlanması için Tablo 2.1.'de belirtilen genel özelliklere sahip olması gerekmektedir.

Bu tez çalışmasında oluşması muhtemel anlam karmaşıklığını gidermek amacıyla yukarıda bahsedilen elektronik elektrik sayaçları terimi yerine literatürde kabul görmüş şekli olan akıllı sayaç terimi kullanılmıştır.

Son yıllarda özellikle AB direktifleri ve enerji verimliliği konularının daha sık gündeme gelmesiyle birlikte mesken ve endüstriyel kullanıcıları için geleneksel elektromekanik ve elektronik sayaçlardan akıllı sayaçlara geçiş hızlanmıştır. Elektromekanik sayaçlar özellikle 1970'lerde ve öncesinde yaygın olarak kullanılırken, 1970 ile 2000 yılları arasında yaşanan teknolojik gelişmeler ile elektronik sayaçların kullanımı artmış ve gerçek zamanlı otomatik sayaç okuma gibi hizmetler kullanılmaya başlanmıştır. Elektronik sayaçlar her ne kadar elektromekanik sayaçlara kıyasla birçok yenilikçi fikrin uygulamaya geçmesine yardımcı olsa da, tek yönlü haberleşmeye imkanı sağladığından tüm ihtiyaçları karşılayamamaktadır. Akıllı sayaçlar çift yönlü haberleşme dahil daha birçok özelliği kullanabilme imkanı sağlayarak tüketicinin ihtiyaçlarına uygun tasarlanabilmektedir. Şekil 2.2.'de modern bir akıllı sayacın temel fonksiyonları ve donanım ekipmanlarının şematik gösterimi verilmiştir.

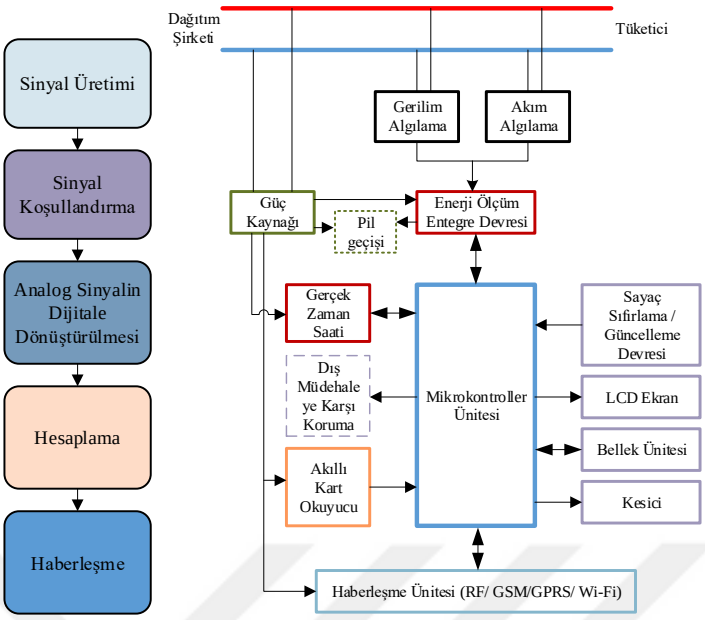
Tablo 2.1. Akıllı sayaçların genel özellikleri [99].

No	Özellik	Değer
1	Çalışma frekansı	50 Hz
2	Yükselti (Rakım)	2000 m
3	Kullanım yeri	Bina içi/Bina dışı
4	Ortam sıcaklığı (°C)	Bina içi (−25°C ile +55°C arası) Bina dışı (−40°C ile +70°C arası)
5	Azami bağıl nem (%)	95
6	Raf ömrü	en az 4 yıl
7	Sayaç pil adeti	2 adet
8	Sayaç pil ömrü	en az 10 yıl
9	Optik port haberleşme hızı	19200 baud rate
10	Sayaç koruma sınıfları	Bina içi (IP51) / Bina dışı (IP54)
11	Haberleşme portu	RS 485
12	Zaman saatinin sapma değeri	Nominal sıcaklıkta en fazla (0.5sn/gün)
13	Hafıza özellikleri	Silinmez hafıza ve her ayın sonundaki tüketim bilgilerini bir yıl boyunca hafızada saklama
14	Sayacın çalışması için gerekli besleme	Anahtarlamalı güç kaynağı (Switch Mode Power Supply-SMPS)
15	Elektriksel koruma sınıfı	Sınıf II
16	Sayaç nominal gerilimi	Tek fazlı sayaçlarda 230V Üç fazlı direkt ve akım trafosuna bağlı sayaçlarda 3x230/400V Gerilim trafosuna bağlı sayaçlarda 3x57.7/100V
17	Darbe gerilim dayanımı	en az 6 kV (R_{kaynak}=2 ohm)
18	Elektrostatik boşalma dayanımı	havada 15 kV, temaslı boşalmada 8 kV
19	Tarife bilgileri	en az 4 tarifeli ve bir günü 8 ayrı zaman dilimine bölebilme
20	Yük profili	15-30-60 dakikalık ayarlanabilir aralıklarla ve saat başı denk gelecek şekilde kaydedilebilme

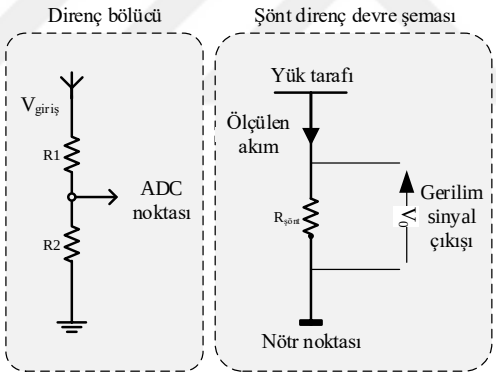
Akıllı sayacın tüketim verisini oluşturma ve kaydetme süreçi temel olarak beş aşamadan oluşmaktadır. Bu aşamalar sırasıyla sinyalin üretilmesi, sinyalin koşullandırılması, analog sinyalin dijitale dönüştürülmesi (Analogue to Digital Conversation-ADC), hesaplama işlemi ve haberleşmedir.

Modern akıllı sayaçlar, giriş sinyallerini elde etmek için gerilim ve akım algılayıcıları kullanmaktadır. Sinyal koşullandırma, ADC ve hesaplama işlemleri mikrokontroller birimi içinde gerçekleştirilmektedir. Sistem operatörleri ile haberleşme, saat ve tarih ölçümleri, veri yedekleme ve depolama gibi diğer işlemler için ek donanım bileşenleri gerekmektedir. Akıllı sayaç tipik olarak şu altı temel donanım bileşeninden oluşmaktadır: gerilim ve akım algılama ünitesi, güç kaynağı ve yedek pil, enerji ölçüm ünitesi, mikrokontroller, gerçek zaman saati ve haberleşme sistemi.

Gerilim Algılama Ünitesi: Basit direnç bölücüler, düşük maliyetleri nedeniyle akıllı sayaçlarda gerilim algılayıcı olarak yaygın olarak kullanılmaktadır. Şekil 2.3.'te, basit bir direnç bölücü tipi gerilim algılayıcısının devre şeması verilmiştir. Şekil 2.3.'deki R_1 ve R_2 değerleri AC şebeke gerilimini enerji ölçüm çipinin ADC'sinin giriş aralığına uygun şekilde bölebilecek şekilde seçilmektedir. Şekil 2.3.'te R_1 direncine AC gerilimi uygulanırken R_2 direnci topraklanır ve ayırıcının orta noktasından çıkış sinyali alınır. Ayrıcının ADC'ye göre çıkış gerilimi ise denklem 2.1.1. kullanılarak hesaplanır.



Şekil 2.2. Modern bir akıllı sayacın donanım altyapısı [100].



Şekil 2.3. Akıllı sayaçlarda akım ve gerilim algılama ünitesi elemanları [100].

$$V_0 = \frac{R_1}{R_1 + R_2} V_{giriş} \qquad (2.\ 1)$$

Denklem 2.1.1.'de belirtilen $V_{giriş}$ ve V_0 terimleri sırasıyla giriş ve çıkış gerilimidir. Genellikle R_1 ve R_2 dirençlerinin değeri kΩ seviyesinde şeçilir ve R_1 direcinin değeri R_2 direncinin değerinden çok daha büyüktür ($R_1 \geq 500R_2$). Gerilim algılayıcılarda yüksek direnç değerleri kullanılarak yüksek güç kaybının oluşması önlenmektedir.

Akım Algılama Ünitesi: Akım algılama ünitesi yaygın olarak akım

algılayıcılardan ve örtüşüm önleyici (anti-aliasing) filtrelerden oluşmaktadır. Akıllı sayaçlarda en sık kullanılan dört farklı akım algılayıcı ise şunlardır: Hall etkili doğrusal akım algılayıcıları, akım trafoları, Şönt dirençleri ve Rogowski bobinleri.

Güç Kaynağı Ünitesi: Güç kaynağı ünitesi, akıllı sayacın tasarımına göre değişiklik gösterebilmektedir. Güç kaynağı ünitesi temel olarak düşürücü trafolardan, doğrultuculardan, AC-DC dönüştürücülerden, DC-DC dönüştürücülerden ve düzenleyicilerden oluşmaktadır. Enerji ölçüm çipi tasarımcıları kendi referans güç kaynağı çıkış şemalarını kullandıklarından akıllı sayaçtaki diğer donanım bileşenlerini çalıştırmak için fazladan enerjiye sahip olmayabilir. Bu durumda güç kaynağı ünitesi akıllı sayaç tasarımı aşamasında enerji ölçüm entegre devresine, mikrodenetleyiciye, LCD ekrana, pil şarj cihazına ve haberleşme ünitesine göre farklılık gösterebilmektedir. Şekil 2.4.'te, akıllı sayaçlarda kullanılan tipik bir güç kaynağının temel elemanları göstermektedir.

Şekil 2.4. Akıllı sayaçlarda kullanılan tipik bir güç kaynağı ünitesi ve elemanları [100].

Güç kaynağı ünitesinde ilk aşamada AC şebeke gerilimi bir diyot köprüsü üzerinden doğrultulurken bazı durumlarda ise doğrultulma işleminden önce devre çalışma gerilimine kadar düşürülmektedir. Daha sonra ise düzensiz gerilim çıkışı bir DC-DC dönüştürücüyü veya bir düzenleyici entegre devresini beslemektedir. Son aşamada ise bir pil şarj ünitesi, şarj edilebilir bir pili kontrol etmektedir.

Enerji Ölçüm Ünitesi: Sinyal koşullandırma, ADC ve hesaplama işlemleri bu entegre ölçüm devresi içerisinde gerçekleştirilmektedir. Enerji ölçüm ünitesinin içerisinde standart bir enerji ölçümü çipi veya sayaç sisteminin mikrodenetleyicisi bulunabilmektedir. Günümüzde kullanılan modern enerji ölçüm çipleri gerekli hesaplamaları yapabilmek için bir dijital sinyal işlemcisine (digital signal processor-DSP) intiyaç duymaktadır. Bu enerji ölçüm entegre devreleri, tek fazlı veya üç fazlı akıllı sayaçlar için farklı şekilde tasarlanmış enerji ölçüm çipleri içermektedir. Bu çipler veri veya frekans (darbe) çıkışı olarak aktif, reaktif ve görünür güç gibi enerji bilgilerini sağlamaktadır. Ayrıca bazı enerji ölçüm entegre devreleri gerilimin ve akımın etkin değerini (Root Mean Square-RMS) hesaplama, frekans ve sıcaklık ölçümü, dış müdehale algılama, güç yönetimi, toplam harmonik bozulma (Total Harmonic Distortion-THD) ve gerilim çökmesi tespiti gibi ek özellikler içerebilmektedir. Enerji ölçüm çipleri IEC ve Amerikan Ulusal Standartlar Enstitüsü (American National Standards Institute-ANSI) doğruluk standartlarına göre tasarlanmaktadır. Enerji ölçüm çipleri tasarlayan tanınmış üreticilerden bazıları ise Microchip, STMicroelectronics, Teridian, Analog Devices and NXP'dir [100].

Mikrodenetleyici Ünitesi: Akıllı sayaç içersinde bulunan tüm fonksiyonlar mikrodenetleyici tarafından gerçekleştirilmektedir. Akıllı sayacın çekirdeği olarak kabul edilebilir. Mikrodenetleyicilerin kontrol ettiği temel fonksiyonlarından bazıları şunlardır: enerji ölçüm çipi ile haberleşme, üretilen verilere dayalı hesaplamalar, elektrik parametrelerini, piyasa tarifelerini ve tüketim maliyetini görüntülemeye imkan sağlama, akıllı kart okuma, sayaca yapılan dış müdehale tespiti, bellek ünitesi ile birlikte veri yönetimi, diğer akıllı cihazlar ile haberleşme ve güç yönetimidir.

Modern akıllı sayaçlar genellikle bir ekran (örneğin; Liquid Crystal Display-LCD ekran) ile birlikte tasarlanmaktadır. Bu sayede tüketiciye tarife, elektrik kesintisi veya diğer işlemler hakkında bilgilendirme yapılabilmektedir. Bazı durumlarda ise akıllı sayaçlardaki mikrodenetleyici ünitesi tüketiciye bir üst tarife hakkında veya yüksek tüketim talepleri konusunda gerekli bilgilendirmeleri alabilmesi için merkezi sistemden gelen uyarı sinyallerini ulaştırır. Akıllı sayaç bu işlevleri yerine getirebilmek için bazı durumlarda harici bir mikrodenetleyiciye de ihtiyaç duyulabilir. Bu gibi durumlarda ise çoklu görev (multi-tasking) veya yüksek dereceli paralellik gibi ek özellikler gerekmektedir. Başka bir ifadeyle, akıllı sayaç tek bir veri kümesine aynı anda birkaç farklı işlemi yapılabilecek kapasitede ve özellikte bir mikrodenetleyiciye gereksinim duyulabilir.

Gerçek Zaman Saati: Gerçek zaman saati tüm akıllı sayaçlarda mevcut ve yerel saati takip etmekle sorumlu önemli bir donanım bileşenidir. Tarih ve saat bilgisi, uyarı sinyalleri gibi önemli bilgileri sağlar. Bazı enerji ölçüm çiplerinde gömülü bir gerçek zamanlı saat aygıtı da bulunmaktadır. Akıllı sayaçların çoğunluğunda ise mikrodenetleyici birimi üzerinden erişilebilen ve ayrı bir şekilde çalışan gerçek zaman saati bulunmaktadır. Gerçek zaman saatlerinin çoğunluğunda zaman saati kayması kabul edilebilir aralıklarda kalacak şekilde tasarlanmaktadır. Bazı akıllı sayaçlar gerçek zaman saatindeki zaman kaymalarını önlemek için gerçek zamanlı olarak mevcut ağ bağlantısı üzerinden periyodik olarak senkronize edilmektedir. Böyle bir ağ bağlantısına bağlı olmayan sayaçların ise yüksek doğrulukta ölçüm yapan bir gerçek zaman saati kullanması veya düzenli aralıklarla zaman kaymalarının düzeltilmesi gerekmektedir.

Haberleşme Ünitesi: AMI sistemler akıllı sayaçlardan, haberleşme ağ geçidinden, akıllı kontrol ve veri yönetiminden oluşmaktadır. AMI sistemlerde çeşitli haberleşme protokolleri kullanılmaktadır. AMI sistemlerde yaygın olarak kullanılan haberleşme protokollerinden bazıları ise HAN, Komşu Alan Ağı (Neighborhood Area Network-NAN) ve Geniş Alan Ağı (Wide Area Network-WAN)'dır. Tüketici seviyesinde kullanılan elektrikli cihazlar, akıllı sayaçlar, elektrikli araçlar veya yerel generatörler ile haberleşme HAN bağlantısı üzerinden gerçekleştirilmektedir. Sayaçlardaki HAN bağlantısı merkezi bir enerji yönetimi, hizmetleri ve tesisleri arasında bilgi akışına imkan sağlar. Akıllı sayaçlar tüketim bilgisini ilk aşamada NAN bağlantısı ve ardından WAN bağlantısı kullanarak merkezi sisteme aktarmaktadır. Elektrik enerjisinin toplu üretimi, iletimi ve dağıtımı aşamalarında toplanan veriler WAN ile iletilmektedir. Dağıtım sistemi operatörleri bu verileri toplamak için WAN bağlantısını kullanmaktadır. Enerji piyasaları, dağıtım sistemi

operatörleri ve hizmet sağlayıcılar ise bilgileri işlemek için internet bağlantısını kullanmaktadır.

HAN bağlantı protokolü veriyi kablolu veya kablosuz ortamlar üzerinden aktarabilir. ZigBee, Z-wave, Wi-Fi ve güç hattı iletişimi (Power Line Communication-PLC), HAN bağlantılarında yaygın olarak kullanılan protokollerdir. PLC bağlantı protokolü mevcut enerji dağıtım hattının altyapısını kullandığı için uygun maliyetli bir yaklaşım olarak değerlendirilse de, gerek şebeke topolojisinin değiştirilemez oluşu gerekse şebeke hattının bozucu etkilerinin veri iletimini olumsuz etkilemesi uygulanmasını ve yaygınlaşması zorlaştırmaktadır. ZigBee bağlantı protokolü HAN uygulamaları için uygun maliyetli, daha az karmaşıklık, düşük güç tüketen ve güvenilir bir iletim ortamı sunmaktadır [100].

NAN bağlantı protokolü, komşu akıllı sayaçlar arasında veri aktarımı için kullanılmaktadır. Bilgilendirme mesajlarını, donanım ekipmanları için yazılımı güncellemeleri ve gerçek zamanlı bilgi aktarımı gibi operasyonel işlemleri kolaylaştırmaktadır. ZigBee haberleşme protokolü, yüksek veri aktarım hızı ve düşük maliyeti nedeniyle NAN bağlantılarında da yaygın olarak kullanılmaktadır.

Bazı durumlarda akıllı sayaçlar, bir WAN bağlantısı aracılığıyla doğrudan uzak bir sunucuya bağlanabilir. Bu durumda herhangi bir NAN bağlantısına ihtiyaç duyulmadan veriler kablosuz bir ağ kullanılarak sunucuya aktarılmaktadır. Sayaç ve uzak sunucu arasında faturalama işlemleri, elektrik kesintilerinin bildirilmesi, uzaktan açma-kesme işlemleri, sayaç dış müdehale tespiti ve uzaktan yapılandırmalar gibi işlemler için bir veri toplayıcı vasıtasıyla iletişim kurulur. Akıllı sayacın WAN ile bağlantısı yapabilmesi için ise çeşitli haberleşme teknolojileri (örneğin; Global System for Mobile Communications (GSM), General Packet Radio Service (GPRS), 3G ve WiMax) kullanılmaktadır.

2.1.2. Çift Yönlü İletişim Kanalları

Merkezi sistem ile haberleşme tüm enerji ölçüm sistemlerinde bulunması gereken zorunlu bir fonksiyondur. Sistemin mevcut durum bilgisi (örneğin; akım, gerilim, frekans, enerji, güç ve güç kalitesi vs.) ve ölçümleri dahil olmak üzere enerji ölçüm çipi içerisinde işlenen tüm verilerin harici bir mikrodenetleyiciye iletilmesi günlük operasyonların devamı için kritik bir işlemdir. AMI sistemlerde yaygın olarak tercih edilen haberleşme bağlantılarından bazıları ise PLC, geniş bant kablolar, Wi-Fi, ZigBee, GPRS, 3G ve radyo frekanslarıdır.

Akıllı şebekenin günlük ve rutin operasyonları enerji akışının yönetimini ve kontrolünü sağlamaktadır. Şebeke izleme ve takip, kontrol, raporlama, denetleme ve alış-şatış gibi işlemler akıllı şebekenin günlük operasyonlarından bazılarıdır. Bu işlemleri gerçekleştirebilmek için tüm trafo merkezleri, tüketici tesisleri ve akıllı saha ekipmanları çift yönlü bir haberleşme ağı kullanılarak birbirine bağlanmaktadır. Akıllı şebeke uygulamalarının çoğunluğu ise bölgesel iletim organizasyonunun (Regional Transmission-Organization-RTO) bağımsız sistem işletmesine (Independent System Operator-ISO) bağlı iletim ve dağıtım SCADA (Supervisory Control and Data Acquisition) sistemlerinin

fiziksel donanım ekipmanlarında ve yönetim sistemlerinde gerçekleştirilmektedir.

Akıllı sayaçta haberleşmenin çoğunluğu TS EN 62056-21 haberleşme protokollerini sağlayan bir optik port ve izole beslemeli RS 485 bağlantısı kullanılarak yapılmaktadır. Ülkemizde kullanılan akıllı sayaçların çoğunluğunda RS 485 seri haberleşme katmanı kullanılmaktadır [101]. Bu haberleşme protokolünün fiziksel katmanı optik port üzerinden gerçekleştirilen seri ve yarı çift yönlü (half-dublex) bir şekildedir. Aynı zamanda RS-232, GSM, GPRS gibi iletişim yöntemleri de kullanılmaktadır. Protokol paket yapısı mantığı ile çalışmaktadır. Ana sunucu (master server) ilk aşamada iletişimi başlattıktan sonra sorgu yapar ve iletişimi sonlandırır. Akıllı sayaç (slave client) dinleme konumundadır ve harici bir iletişim başlatamaz.

Ülkemizde kullanılan akıllı sayaçların haberleşme protokolleri Türk Standartları Enstitüsü (TSE), Avrupa Elektroteknik Standart Komitesi (EN) ve IEC tarafından belirlenmekte ve en son yayımlanan baskılarına uygun olarak tasarlanmaktadır. Akıllı sayaçların tasarım ve üretim süreçlerinde kullanılan standartlar Bölüm 5.'teki Tablo 5.1.'de listelenmiştir.

EN ve IEC tarafından geliştirilen standartlar teknolojik gelişmeler dikkate alınarak sürekli olarak güncellenmektedir. Bu standartlar gelişen teknolojiyle birlikte artan veri türleri ve ölçüm parametrelerini bir tanımlama altyapısı üzerinden gruplandırarak ortak bir platform oluşturmaktadır. Ortak Asgari Kodlama Yapısı, diğer ismiyle Nesne Tanımlama Sistemi (Object Identification System-OBİS) tüm sayaçlarda ortak bulunan parametrelerin ve bu parametrelerin veri formatlarının tanımlandığı kod sistemidir [102].

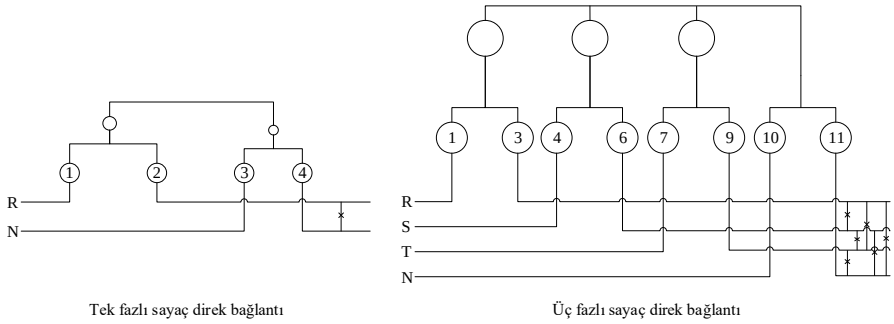
OBIS kodları, A'dan F'ye kadar olan altı değer grubunu kullanarak hiyerarşik bir şekilde enerji ölçüm ekipmanında kullanılan veri öğelerini tanımlamaktadır.

Grup A, sayacın hangi enerji ölçüm türü için kullanıldığı belirtmektedir ve bir sayaç yalnızca tek bir enerji türünden ölçüm almak üzere tasarlanmaktadır. Grup A'nın değerleri 0 ile 15 arasında değişmektedir (örneğin; 0: genel bilgi, 1: elektrik ile ilgili, 5: soğutma ile ilgili, 9: sıcak su ile ilgili vs.). Grup A'nın 9 üzerindeki diğer değerleri oluşabilecek diğer olası enerji ölçümleri için ayrılmıştır.

Grup B, sayacın aldığı ölçümleri fiziksel bir kanal numarası üzerinden aktarmak için kullandığı numaraları ifade etmektedir. Ölçüm yapılan noktadaki parametrelerin aynı olup olmadığı dikkate alınarak farklı türlerdeki veriler farklı kanallar üzerinden iletilmektedir. Grup B'nin değer aralığı 1 ile 64 arasında değişmektedir ve 65'ten 127'ye kadar olan değerler ise gelecekteki uygulamalar için belirlenmiştir.

Grup C, daha önce A grubundaki enerji türüne bağlı olarak farklı tanımlanmış değer türlerini ifade etmektedir. Grup C genel olarak akım, gerilim, güç, hacim, sıcaklık gibi referans veri kaynağıyla ilişkili soyut veya fiziksel veri öğelerini tanımlamaktadır.

Grup D, genel ölçümlerin diğer kodlardan daha ayrıntılı alt bölümleri için veya bu enerji ölçümlerinin önceden tanımlanmış bazı algoritmalara göre işlenmiş sonuçlarını ifade etmektedir. Algoritmalar enerji ve talep gibi niceliklerin yanı sıra diğer fiziksel nicelikleri de ifade edebilmektedir.



Şekil 2.5. Tek fazlı ve üç fazlı sayaç bağlantı şeması

Grup E, A'dan D'ye kadar olan değer gruplarındaki değerlerle tanımlanan niceliklerin özel tarife seçeneklerindeki farklı oranları ifade etmektedir. Grup F ise farklı fatura dönemlerine göre A'dan E'ye kadar tanımlanmış gruplarındaki değerlerinin daha ileri alt bölümlerinin bilgisini farklı zaman periyotlarına bölerek saklamaktadır. Genellikle tarihsel parametreler için kullanılmaktadır.

Sayaç bağlantılarının hatalı ölçüme yol açmaması için ölçüm yapılacağı devreye doğru bir şekilde yapılması gerekmektedir. Şekil 2.5.'te örnek bir tek fazlı ve üç fazlı sayacın bağlantı şeması verilmiştir.

Ayrıca sayaç numaratörleri tarafından kaydedilen değerlerden tüketim değerini elde edebilmek ve ölçüm hatalarını önlemek amacıyla bir sayaç çarpan katsayısı hesaplanmaktadır. Bu işlem sonrasında bir sayaç çarpan katsayısı elde edilerek sayaçtan okunan değerler bu değer ile çarpılır. Sayaç çarpan katsayısı denklem 2.1.2.'de verilen formül ile hesaplanmaktadır.

$$\text{Sayaç Çarpan Katsayısı} = \frac{\text{ATDO} \times \text{GTDO}}{\text{Sayaç İç Çarpanı}} \tag{2. 2}$$

Burada ATDO ve GTDO değişkenleri sırasıyla akım ve gerilim trafolarının dönüştürme oranlarını temsil etmektedir.

Sayaç haberleşmesinde yaygın olarak kullanılan protokoller seri haberleşme şeklini kullanmaktadır. Seri haberleşmede dört farklı parametre kullanılmaktadır. Bu parametreler sırasıyla veri haberleşme hızı, karakter olarak kodlanan veri bitlerinin sayısı, eşlik biti seçimi ve durdurma bitidir. Bu veri iletim çerçevesinde, ilk aşamada tek başlangıç biti daha sonra ise iletilecek olan veri biti ve son olarak ise eşlik biti ve durdurma biti veya bitleri bulunmaktadır.

RS 232 haberleşme protokolünde yalnızca iki gerilim seviyesi kullanılmaktadır. Bu iki farklı gerilim seviyesi sırasıyla iletilen işareti (mark) ve boşluğu (space) temsil etmektedir. İletilen sinyalde negatif gerilim seviyesi işareti, pozitif gerilim seviyesi ise boşluğu ifade etmektedir. Sinyalin +3V'dan büyük olması halinde (sinyal $>$ +3V) çıkış 0, −3V'dan

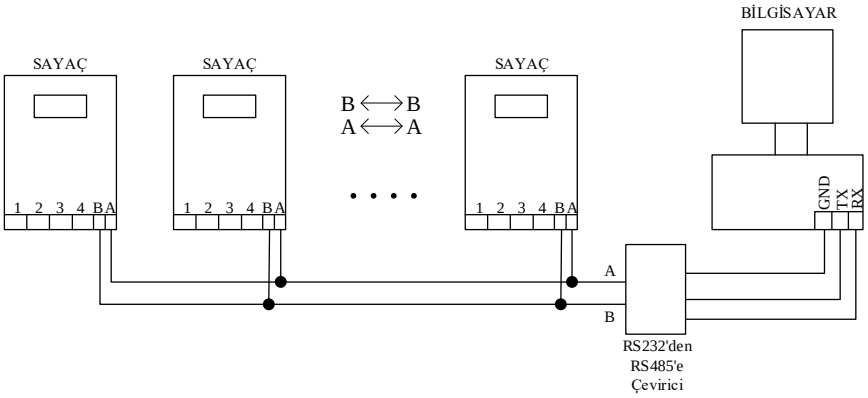
küçük olması halinde ise (sinyal $< -3V$) çıkış 1'dir.

Başlangıç biti her veri paketinin başlangıcında yer almaktadır. Bu bit negatif ve pozitif gerilim seviyeleri arasındaki geçişdir. Eşlik biti seçimi iletilen veri bitlerini takip eden bir hata kontrol şeklidir. Veri iletiminden önce eşlik bitinin tek mi çift mi (odd-even) olduğu belirlenir. Eşlik bitinin tek olması durumunda eşlik biti ve veri bitlerinde bulunan 1'lerin toplamı tek sayı olmaktadır. Örneğin veri bitlerindeki 1'lerin toplamı 3 ise eşlik bitinin değeri 0'dır ve böylece 1'lerin toplam adeti tek sayı olur. İletilen veri paketinin son bölümünde ise durdurma biti bulunmaktadır ve daima negatif gerilim seviyesi ile ifade edilmektedir. İletilen veri paketinden sonra yeni bir paketin gelmesi durumunda ise yeni durumun habercisi olarak pozitif gerilim (boşluk) başlangıç biti kullanılmaktadır.

Seri haberleşmede aktarılan sinyalin veri akışını kontrol edilebilmek için saat sinyaline ve zamanlama frekansına ihtiyaç duyulmaktadır. Alıcı ve verici ekipman bu şekilde gönderilen her bir bitin ne zaman gönderileceğine ve alınacağına karar vermektedir. Saat sinyalinin kullanım yerine göre seri haberleşme senkron veya asenkron şeklinde gerçekleştirilmektedir. Senkron haberleşme yönteminde bütün cihazlar tek bir cihaz tarafından veya harici bir cihaz tarafından üretilen saat sinyalini kullanmaktadır. İletilen tüm veri paketinde bulunan sinyaller saat sinyali ile senkronizedir. Seri haberleşmenin senkron şekilde gerçekleştirilmesi 4.57 metreden daha az ve tek bir devre üzerinden bağlantı gerçekleştiren ekipmanlar için uygundur. Daha uzak mesafelerde gürültüden etkilendiği ve saat sinyali için fazladan hat kurulumu gerektirdiği için uygulanabilir değildir. Asenkron bağlantılar için fazladan bir saat sinyaline ihtiyaç duyulmamaktadır. Her iki ekipmanda (alıcı ve verici) kendi saat sinyalini sağlayan donanım ekipmanı bulunduğundan yalnızca kendi içinde senkron olması yeterlidir. Bu nedenle veri paketinde iletilen her bir byte için saat sinyalini eşleştirmek üzere bir başlangıç (start) biti ve haberleşmenin bittiğini bildirmek bir durdurma (stop) biti bulunmaktadır. RS 232 ve RS 485 haberleşme bağlantıları asenkron seri haberleşmedir.

RS 485 veya diğer adıyla Electronic Industries Association (EIA) 485 half-dublex veri iletimi yapabilen ve haberleşme mesafesini 1.2 km'ye kadar genişletebilen haberleşme standartıdır. Seri haberleşmede RS 232' nin iletim mesafesinden (yaklaşık 15.24 m) daha uzak mesafelere veri iletimi gerçekleştirebildiği için RS 485 tercih edilmektedir. RS 232 haberleşmede tek sonlu hatlar tercih edilirken, RS 485 haberleşme dengelenmiş hatlar üzerinden bilgi taşıyabilmektedir. İletilen her iki sinyal de tek bir kablo üzerinden hem positif hemde negatif gerilim seviyeleri ile birlikte kablo çiftlerine aktarılmaktadır. Alıcı ekipman her iki gerilim seviyesi arasındaki farkla cevap sinyali üretmektedir. Dengelenmemiş bir haberleşme hattında bir sinyal kablosu, dengelenmiş haberleşme hattında ise iki adet sinyal kablosu kullanılmaktadır. Dengelenmiş haberleşme hatlarının gürültüden daha az etkilenmeleri en önemli özelliklerinden bir tanesidir.

Seri haberleşmede half-dublex yapı kullanan bir sistemde birden fazla verici birden fazla alıcı ile herhangi bir zaman dilimi içerisinde sadece bir tanesi aktif olacak şekilde haberleşmektedir. Bu haberleşme şeklinde her iki ekipmanda hem alıcı hemde verici



Şekil 2.6. Üç fazlı akıllı sayaç ile ana sunucu arasındaki RS 485 bağlantı şeması

olarak çalışarak komut gönderme ve alma işlemlerini gerçekleştirebilmektedir. Bir RS 485 bağlantısında ana sunucu (master server) özel olarak adreslenmiş akıllı sayaç (slave client) ile bir karakter dizisi kullanarak konuşmaya başlar ve sayaç sisteminin cevap döndürmesini bekler. Önceden tanımlanmış zaman aralıklarında sayaç cevap vermezse ana sunucu iletişimi sonlandırır. Şekil 2.6.'da üç fazlı akıllı sayaç ile AMI sistemin ana sunucusu arasındaki RS 485 bağlantısının genel gösterimi verilmiştir.

Otomatik Sayaç Okuma (Automatic Meter Reading-AMR) sistemlerinde kullanılan akıllı sayaçlarda genellikle RS485 standartı tercih edilmektedir ve 32 adet cihaza kadar bağlantı yapılabilmektedir. Bu şekilde tüm veriler tek bir bağlantı noktası üzerinden ana sunucuya gönderilebilmektedir.

Modbus ücretsiz bir protokol olarak birçok farklı türdeki cihaz için kullanılan bir seri haberleşme protokolüdür. Modicon tarafından ilk aşamada programlanabilir sayısal kontroller (Programmable Logic Controller-PLC) için tasarlanmış olsa da birçok ağ haberleşmesinde kullanımının basit olması ve seri hatlar üzerinden iki cihaz arasındaki bilgi transferini kolaylaştırması gibi avantajlarından dolayı tercih edilmektedir. Modbus bir ana sunucu (master)/ uydu (slave) seri haberleşme protokolüdür. Modbus kullanılan bir veri iletim yolunda sadece bir ana sunucu ve en fazla 247'ye kadar farklı uydu bağlantısı kurulabilmektedir. Uydu cihazlar ana sunucudan komut gelmedikçe veri veya bilgi iletişimi sağlayamazlar. Ana sunucu her bir haberleşmede bir kez işlem başlatır ve uydu cihazlar birbirleri ile iletişim kuramazlar. Modbus protokolünde veri, bilgi alışveriş Amerikan kod standardı (American Standard Code for Information Interchange-ASCII) veya uzak terminal birimi (Remote Terminal Unit-RTU) gibi bir seri haberleşme modunda tanımlı çerçeve kullanılarak temsil edilmektedir. Cihazlar Modbus-ASCII modunda yapılan haberleşmede veri paketlerindeki her bir 8 bit ikili ASCII karakteri kullanılarak iletilir. RTU modda ise her bir 8 bitlik veri paketi iki adet 4 bitlik hexadecimal karakter kullanılarak iletir ve daha büyük karakter yoğunluğu sayesinde ASCII mod ile

kıyaslandığında haberleşme hızı daha yüksektir.

IEC 62056-21 Avrupa Birliği ülkelerinde bilgisayar ile sayaç okuma gibi uygulamalar için geliştirilen ve yaygın olarak kullanılan bir haberleşme protokolüdür. IEC 62056-21 protokolü RS485 portu ve optik port kullanarak half-dublex şekilde ASCII kod göndermektedir. A, B, C, D ve E olmak üzere beş farklı protokol modu kullanılmaktadır. A, B, C ve E modları veri alışverişini çift yönlü olarak gerçekleştirmektedir. Haberleşme ilk aşamada ana sunucunun okuma istek mesajı göndermesiyle başlar. A ve C protokollerinde okuma cihazı ana sunucu (master) ve sayaçlar ise uydu (slave) gibi davranmaktadır. E protokolünde ise tam tersine okuma cihazı istemci (client), sayaç sunucu (server) gibi davranmaktadır. Bu protokoller sayesinde sayaç okuma veya programlama gibi işlemler yapılabilmektedir. D protokolünde haberleşme tek yönlüdür ve sadece sayaç okuma işlemleri gerçekleştirilmektedir. Bu protokolde bilgi sayaçtan okuma cihazına fiziksel buton kullanılarak veya sayaç üzerindeki algılayıcı kullanılarak başlatılmaktadır.

2.2. Büyük Veri

Günümüz ekonomisinde faaliyet gösteren birçok büyük şirket yeni iş fırsatları ortaya çıkarabilmek ve sürdürülebilir ekonomik büyüme planları geliştirebilmek amacıyla müşterilerinden veya kuruluşun iç yapısından gelen geri bildirimleri ve operasyonel verileri daha sık kullanmaya başlamıştır. Bu süreçlerden elde edilen büyük miktardaki verinin yönetilmesi ve analiz edilmesi için geliştirilen teknolojiler ise büyük veri analitiği olarak tanımlanmaktadır. Bunun yanısıra elektriğin üretimi, iletimi ve dağıtımı aşamasında aktif görev alan kurum ve kuruluşlar ise bu kavramlara ek olarak "enerjinin interneti" gibi yeni kavramları dahil etmektedir.

Büyük verinin tanımı günümüzde çok net ve tekdüze bir anlam taşımamaktadır. Ancak genel anlamda farklı tanımlar arasında bir fikir birliğinden bahsedilebilir. Bu ise, kurum veya kuruluş için faydalı bilgileri etkin bir şekilde ortaya çıkarabilmek amacıyla yeni çerçeve (framework) ve yöntemlere ihtiyaç duyan büyük hacimli, çeşitli kategorilerden ve karmaşık yapılardan oluşan bir veri kümesinin beraberinde getirdiği ve yeni ortaya çıkan bir teknik sorundur [103]. Bu nedenle, büyük verinin tanımı, veri madenciliği algoritmalarının ve ilgili donanım ekipmanının büyük hacimli veri kümeleriyle başa çıkma yeteneğine bağlıdır [104]. Büyük veri aynı zamanda mutlak bir tanım yerine göreceli bir kavramdır ve tanımlandığı ortama göre değişkenlik gösterebilir.

Hem enerji hem de bilgi sistemlerini bir araya getiren akıllı şebeke, elektrik üretimi, iletimi, dağıtımı ve tüketimi sürecinden elde edilen büyük miktarda veri içeren bir bilgi kaynağı olarak da düşünülebilir. Bu veriler, dağıtım istasyonlarından sayaçlara kadar olan tüm sistemlerin enerji bilgilerinin yanısıra pazarlama, meteorolojik ve bölgesel ekonomik veriler gibi enerji ile ilgili olmayan diğer bilgileri de içermektedir [105].

Akıllı şebekede, hem dağıtım şirketine (DSO) hem de tüketicilere enerji ile ilgili bilgiler akıllı sayaçlar yardımıyla toplanır ve iletilir. Toplanan bu verilerin hacmi ve

Tablo 2.2. DEPSAŞ'ın 2015-2021 yılları arasında sisteme dahil ettiği sayaç sayısı ve üretilen ortalama veri miktarı [Kaynak: DEPSAŞ].

Yıl	Sisteme Eklenen Sayaç Sayısı	Veri Miktarı
2015	23.400	2.1 TB
2016	98.000	12.994 TB
2017	128.500	35.421 TB
2018	174.300	73.489 TB
2019	168.200	126.653 TB
2020	183.900	196.320 TB
2021	142.300	278.757 TB
Toplam	**918.600**	**278.757 TB**

çeşitliliği kullanılan saha ekipmanlarına ve örnekleme sürelerine göre değişiklik gösterse de genel anlamda büyük veri olarak değerlendirilir. Örneğin; AMI sistemi verimli şekilde kullanan büyük bir dağıtım şirketinin akıllı sayaç okumalarının sayısının yılda 24 milyondan günde 220 milyona artması bir büyük veri problemidir [85]. Dicle Elektrik dağıtım şirketinin son yedi yılda her yıl sisteme dahil ettiği sayaç sayısı ve üretilen ortalama veri miktarı Tablo 2.2.'de verilmiştir.

Büyük veriyi işleyecek sistemlerin en önemli bileşeni ise, bilgi keşfinde ve karar süreçlerinde kullanılmak üzere tasarlanmış veri analizi yöntemleridir [106], [107]. Genel anlamda değerlendirildiğinde, veri analitiği veya veri madenciliği gibi uygulamalar, veri tabanı, istatistik, örüntü tanıma, makine öğrenimi vb. dahil olmak üzere birçok yöntemle değişkenler arasındaki potansiyel ilişkileri ortaya çıkarmak için oluşturulmuş hesaplama sürecini ifade etmektedir.

2.2.1. Büyük Veri Kavramına Genel Bir Bakış

Büyük verinin üç önemli özelliği hacmi, çeşitliliği ve hızıdır [108]. Adından da anlaşıldığı üzere, verinin miktarı veya hacmi büyük verinin çok önemli bir yönüne vurgu yapmaktadır. Bazı durumlarda ise verinin sermaye ve emeğe benzer şekilde bir ekonomik girdi olarak değerlendirildiği ve iş dünyasının yeni hammaddesi olduğu belirtilmektedir.

Akıllı sayaçlar ve sensörler enerji büyük verisinin oluşumuna katkı sağlayan önemli kaynaklardandır. Büyük enerji verisindeki bu artış akıllı sayaçların devreye girmesiyle birlikte neredeyse 1000 kat daha fazla verinin gelmesi ihtimalini ortaya çıkarmaktadır [109]. Veri çeşitliliği, farklı veri türlerini ifade etmektedir. Sosyal medya veya akıllı cihazlar gibi artan sayıda farklı veri kaynağı farklı veri türlerini ifade etmektedir. Hız, veri aktarım hızını veya işlem hızını ifade etmektedir. Artan veri hacmi, çeşitliliği ve hızı veri yönetimi ve analizi aşamasında zorluklara neden olabilmektedir. Büyük veriden bahsederken genellikle verinin hacmi, çeşitliliği ve hızı ön plana çıksa da, büyük verinin en önemli yönlerinden biri ondan elde edilecek değerdir. Bir veri kümesi çok büyük, çeşitliliğe sahip ve hızlı bir şekilde artıyor olsa bile, faydalı herhangi bir bilgiye veya sonuça ulaştırmayan verinin değeri azdır.

Büyük veri analiz aşamasında büyük miktarda verinin temin edilmesi tek başına

yeterli değildir. Veri analistleri genellikle karar verme ve modelleme süreçlerinde tüm veri kümesini daha uygun ifade edecek bir alt veri kümesi üzerinden işlemleri gerçekleştirmektedir. Ayrıca akıllı sayaç ve sensör gibi cihazlardan planlı ve sistematik bir şekilde veri toplama bu sürecin sağlıklı bir şekilde yürütülmesine daha fazla katkı sağlamaktadır. Bu tür bir yaklaşımla temin edilen verinin miktarı daha az olsa da planlanmamış ve kontrol edilmeyen büyük bir veri kümesinden çok daha fazla fayda sağlayabilir [110]. Son olarak verilerin nasıl toplandığına veya veri boyutunun ne kadar büyük olduğuna bakılmaksızın en önemli husus, verilerin analizi sonrasında elde edilebilecek değerdir.

Büyük veri, büyük veri kümelerini oluşturmanın ve depolamanın yanı sıra uygun veri analiz tekniklerini de ihtiyaç duymaktadır. Bu noktadan hareketle yazılım algoritmalarındaki iyileştirmeler, artan veri miktarının neden olduğu bilgi işlem zorluklarını aşabilmek için Moore yasası kadar önemlidir [111]. Ayrıca akıllı ölçüm ve sensör verilerinin hacminin üstel olarak artacağı düşünüldüğünden büyük veri teknolojisinde meydana gelen büyümenin de hızlı bir şekilde gerçekleşeceği öngörülmektedir [112].

OSOS'dan temin edilen elektrik tüketim verileri, güç sistemlerinde artan veri hacminin başlıca nedenlerinden biri olarak gösterilmektedir. Büyük veri kümelerini oluşturma ve saklama zorluklarının yanı sıra elektrik dağıtım şirketlerinin karşılaştığı en büyük zorluk, büyük verilerin analizi ve elde edilecek sonuçlar dikkate alınarak geliştirilecek iş zekası uygulamalarıdır [112], [113]. Bu yüzden veri yönetim sistemlerinde yatırım önceliği veri analitiği araçlarındadır ve güç sistemleri alanında büyük veri analizinin kullanımını yaygınlaştırmak için geniş bir kapsam bulunmaktadır. Enerji sektöründe arz-talep tahmini, tüketicilerin tüketim örüntülerinin çıkarılması, kesintileri veya kayıp-kaçak kullanımını engelleme ve enerji dengesizliklerini optimize etme gibi uygulamalar büyük verinin olası uygulamaları olarak gösterilebilir. Güç sistemlerindeki diğer uygulamaları ise güç sistemlerinin işletilmesi, kontrolü ve koruması örnek olarak verilebilir [114], [115].

Büyük verinin önemini ortaya koyan ilk örneklerinden biri Google'ın grip eğilimleri projesidir [116]. Bu proje internet arama sorgularını kullanarak Hastalık Kontrol ve Önleme Merkezlerinin raporlarını tahmin etmek için oluşturulmuştur ve büyük verinin örnek bir kullanımı olarak kabul edilmiştir. Bu projede Google'ın grip benzeri hastalıklar için doktor ziyaretlerinin oranını Hastalık Kontrol ve Önleme Merkezlerine kıyasla yaklaşık iki katı daha yüksek bir doğrulukla tahmin ettiği rapor edilmiştir. Bu analizlerin bazı sorunlardan biri ise yanıltıcı korelasyon riskiyle ilgilidir. Yanıltıcı korelasyon, veri kümelerindeki iki farklı değişkenin istatistiksel olarak anlamlı bir korelasyon göstermesi ancak değişkenler arasında temel bir nedensellik veya anlamlı bir ilişki olmadığı tüm durumları ifade etmektedir. Anlamlı ilişkiye bir örnek olarak, herhangi bir zaman serisindeki ardışık ölçümler arasındaki otokorelasyon örnek olarak gösterilebilir. Kesin bir şekilde nedensellikten bahsedilemez ancak modellemesi anlamlıdır.

Büyük veri analizinde karşılaşılan diğer bir sorun ise yanlı örneklemdir. Bu soruna örnek olarak 1936 yılında yapılan Amerikan Başkanlık seçimlerinde seçimin galibini tahmin etmek için yapılmış iki farklı anket gösterilebilir. Bu anketlerden ilki The Literary Digest tarafından 2.4 milyon geri dönüşlü bir posta anketi ve diğeri kamuoyu yoklaması öncüsü olan George Gallup'un yaklaşık 3000 röportajından oluşan daha küçük bir anketidir. O yıllar için büyük bir veri hacmine sahip daha büyük ilk anketin seçim sonuçlarını daha iyi tahmin edebileceği varsayılmış ancak gerçekte daha küçük olan ikinci anketin sonuçları daha doğru sonuçlar üretmiştir [110]. Büyük veri analizinde karşılaşılan diğer sorunlardan biri ise çok büyük veri kümeleri tüm istatistiksel popülasyonu temsil ediyor gibi görünebileceğinden, örnekleme sorunlarına neden olmasıdır. Tüm örneklem veri kümesini kullanmak büyük veri kümelerindeki olası yanlı örnekleri bulmayı zorlaştırabilir. Bu aşamada veri analistlerinin yapacağı değerlendirme veri analizini doğrudan etkiler ve bu nedenle verilerin ve sonuçların daha genel bir değerlendirmesi aşamasında matematiksel analizler daha büyük bir önem kazanır. Elde edilen sonuçların yorumlanması ve problemle ilgili genel çıkarımlarla karşılaştırılması önemli bir aşama olarak kabul edilmektedir [117].

2.2.2. Yazılım Desteği ve Platformlar

Bilgisayar sistemleri literatüründe dağıtık hesaplama (distributed computing) kavramı farklı şekillerde tanımlanabilmektedir. Dağıtık bir sistem genel anlamda kullanıcıya tek bir bilgisayar gibi görünen ancak birbirinden bağımsız bilgisayar kümelerinden oluşan sistemlere verilen genel bir tanımdır [118]. Dağıtık sistemi oluşturan bileşenler (bilgisayarlar) arasındaki farklılıklar ve bu bileşenlerin birbirleriyle olan iletişimi hakkındaki tüm detaylar mümkün olduğunca kullanıcıdan gizlenmiştir. Tek bir sistem gibi görünen bilgisayar kümelerinin birbiri arasındaki iletişimini desteklemek amacıyla dağıtık sistemlerde genellikle bir yazılım ara katmanı bulunmaktadır. Bu yazılım katmanı mantıksal olarak uygulamanın veya kullanıcının üstünde, işletim sistemi ve ağ katmanının altında yer almaktadır.

Herhangi bir işi (job) oluşturan alt işlemler için birden fazla işlemcinin gücünü kullanarak işlem yapmaya paralel hesaplama (parallel computing) denir. Paralel hesaplama dağıtık hesaplama sistemlerinin bir alt dalıdır ve öncelikli olarak karmaşık işlemlerin yürütme sürelerini (execution time) azaltmayı hedefler. Bu açıdan değerlendirildiğinde daha kısa sürelerde sonuç elde etmek ve büyük ölçekli uygulamalarda karşılaşılan problemleri çözmek amacıyla paralel hesaplama kulllanılmaktadır.

Paralel hesaplama genel anlamda verilerin paralelleştirilmesi ve işlerin paralelleştirilmesi olarak değerlendirilebilir [119]. Bu iki farklı yaklaşım kullanılarak işlemcilerde verilerin dağıtılması veya işlemlerin paylaştırılması yoluyla dağıtık hesaplama işlemleri gerçekleştirilir. Kullanıcıdan dağıtık hesaplama aşamasında izlenen karmaşık süreçlerin gizlenmesi işlemine ise saydamlık (transparency) denir. Dağıtık sistemlerde çeşitli saydamlaştırma yapıları bulunmaktadır. Bunlar kısaca Tablo 2.3.'te özetlenmiştir.

Grid hesaplama; dağıtık ve büyük ölçekli bilgisayar kümelerinden oluşan ağ

Tablo 2.3. Dağıtık sistemlerin saydamlık özellikleri ve tanımları [118].

Saydamlık	**Tanımı**
Erişim (*Access*)	Veri temsilindeki farklılıkların ve kaynağa nasıl erişildiğinin gizlenmesi
Konum (*Location*)	Kaynak konumunun gizlenmesi
Göçürme (*Migration*)	Kaynağın farklı bir konuma taşınma işleminin gizlenmesi
Yer değiştirme (*Relocation*)	Kullanım esnasında kaynağın farklı bir konuma taşınmasının gizlenmesi
Yineleme (*Replication*)	Kaynağın birden fazla kopyasının olmasının gizlenmesi
Paralel erişim (*Concurrency*)	Kaynağın birden fazla kullanıcı tarafından paylaşılıyor olmasının gizlenmesi
Hata (*Failure*)	Hataların ve kaynağın yeniden kurtarılmasının gizlenmesi

bağlantılı bir paralel işleme biçimi olarak düşünülebilir [120]. Bu sistemde kaynaklar (örneğin bellek, işlemci, bilgisayar veya servisler) grid sistemine dinamik bir şekilde bağlanabilir veya ayrılabilir. Her ne kadar kaynaklar coğrafi olarak farklı konumlarda bulunsalar da kullanıcılar bu sisteme istedikleri zaman bağlanarak işlemlerini gerçekleştirebilirler.

Servis Odaklı Mimari (Service-oriented Architecture-SOA), kaynakların ve hizmetlerin bir servis olarak sunulduğu sistemlerdir [121]. SOA mimarisinde işlemler istek-cevap prensibiyle senkron veya asenkron şekilde yürütülebilmektedir. SOA üzerinden birçok heterojen sistem birlikte kullanabilir ve işlemci, hafıza, işletim sistemi gibi farklı kaynak ve hizmetler sunulabilir.

Bulut bilişim, grid hesaplamada karşılaşılan bazı problemleri ortadan kaldırmak amacıyla geliştirilmiş ve konsept olarak grid hesaplamaya oldukça benzemektedir. Kullanıcıların grid hesaplamada coğrafi konumları birbirinden farklı olan kaynakların veya kaynak kümelerinin hesaplama kümesine ne zaman dahil edilip edilemeyeceğine karar vermeleri gerekir. Grid yapısı farklı coğrafi konumlarda bulunan donanım ekipmanları üzerinde kurulduğundan kaynak yönetimini ve işlerin zamanlanmasını dinamik olarak kontrol etmek oldukça zordur [122]. Grid sistemin bir diğer sorunu ise genel olarak ağ tabanlı her sistemde karşılaşılan güvenlik konusudur.

Bulut bilişim internet aracılığıyla birbirine bağlanan, birlikte çalışabilen ve dinamik bir şekilde yönetilebilen, izlenebilen veya bakımı yapılabilen sanal sunuculardan oluşan sistemin genel adıdır. Bulut bilişim temel olarak internettir ve ağ diyagramlarında internetin temsili bulut şeklindedir. Kullanıcılar bulut bilişim teknolojileri sayesinde kendilerine özgü sanal imajlar oluşturup kullanabilir veya kendi ihtiyaçlarını dikkate alarak bulut üzerinden mevcut bir imajı sanal makine olarak çalıştırabilirler [123]. Herhangi bir sanal makine, üzerinde kurulu olduğu kaynaktan bağımsız olarak harici bir ortam oluşturabilir. Bu sayede kaynak üzerinde oluşturulan sanal sunucular farklı işlemci kapasitesine, disk kapasitesine, bellek birimine ya da bant genişliğine göre tasarlanabilirler. Dağıtık sistemin ölçeklendirilebilmesi ile kaynak paylaşımının esnekliği

Tablo 2.4. Bulut sisteminde kullanılan kaynak ve altyapının temel katmanları

Bileşen	**Açıklama**
Veri	Uygulamanın kullandığı veri
Ağ	İnternet bağlantısı
Uygulama	Bulut bilişim teknolojisi üzerinde çalışan yazılım
Sunucu	Dağıtık hesaplama kaynağını içeren sanal makine
Çalışma anı	Uygulamanın kodlarının yürütülme anı
İşletim sistemi	Sanal makinenin sahip olduğu işletim sistemi
Sanallaştırma	Sanal makinenin imajlarının oluşturulması ve başlatılması işlemi
Ara katman yazılımı	İşletim sisteminde uygulamayı çalıştırabilmek için gerekli ara birim
Depolama	Sunucunun sabit disk kapasitesini harici bir şekilde artırmayı sağlayan birim

arasında doğrusal bir ilişki bulunmaktadır. Bu sayede bulut bilişim kaynaklarının etkin bir şekilde yönetilebilmesi mümkün olur. Bulut sistemini oluşturan kaynakların ve alt yapının ara katmanlarında yer alan önemli bileşenler Tablo 2.4.'te açıklanmıştır [124].

Bulut bilişim sağlayıcıları üç temel servis modeli kullanarak son kullanıcılarına hizmet vermektedir. Bu modeller sırasıyla Servis Olarak Altyapı (Infrastructure as a Service-IaaS), Servis Olarak Platform (Platform as a Service-PaaS) ve Servis Olarak Yazılım (Software as a Service-SaaS)'dır.

IaaS hizmet modelinde depolama, ağ, sanallaştırma ve sunucu hizmetleri bir bütün olarak sunulur. Bu hizmeti tercih eden kullanıcılar fiziksel donanım altyapısı zorunluluğundan kurtularak ihtiyaç duydukları sanal makinelerini (sunucularını) bu alt yapı üzerinden başlatabilirler. Bu şekilde kullanıcıların ihtiyaç duydukları fiziksel alt yapı bir hizmet olarak sunulmuş olur. Bu alt yapıların oluşturulması için gerekli ilk yatırım maliyeti hizmet olarak kiralanmasına kıyasla oldukça pahalıdır. Satın alınan ya da kiralanan hizmetler ihtiyaç duyulan süre boyunca kullanılır ve kullanım miktarına göre ödeme prensibine göre çalışır. Bu hizmet modelinin en çok bilinen örneklerinden bazıları ise; Amazon EC2, Microsoft Azure, Rackspace, Google Compute Engine, Digital Ocean, Cisco Metacloud ve Oracle'dır.

PaaS hizmet modelinde uygulamaların çalışabilmesi için gerekli donanımlara ek olarak işletim sistemleri (örneğin; Windows Server, Ubuntu Server) ve platform yazılımları (örneğin; SQL, JDK, .Net) bulunmaktadır. Bu hizmet modelinin en çok bilinen örneklerinden bazıları ise; AWS Elastic Beanstalk, Windows Azure, Google App Engine, Apache Stratos'tur.

SaaS hizmet modelinde kullanıcılar bulut bilişim tabanlı uygulamalardan internet aracılığıyla bağlanarak hizmet alırlar. Bu hizmet modelinde bulut bilişim hizmeti sağlayıcıları tarafından hizmetin tüm bileşenleri eksiksiz olarak kullanıcılara sunulur. Bu hizmet modeline örnek olarak Microsoft Office 365, e-posta veya takvim uygulamaları verilebilir. Bu hizmet modelinin en çok bilinen örneklerinden bazıları ise; Google Workspace, Google Docs, Dropbox, Cisco WebEx, Adobe Creative Cloud'dır.

Bulut bilişim teknolojilerinin alt yapı ve platform hizmetlerinde kullandığın kadar

öde (pay as you go) yaklaşımı kullanıcıların uygun fiyatlara hizmet alabilmelerini kolaylaştırmaktadır. Geleneksel hesaplama yaklaşımlarında hesaplama gücü, bant genişliği ve bellek alanı ihtiyaçlar dikkate alarak belirlendikten sonra kiralanmaktadır. Bu yaklaşım kaynağın kullanılmadığı zamanlarda dahil olmak üzere kaynak reserve edildiği için yüksek hesaplama maliyetlerine neden olabilmektedir. Bu noktadan yola çıkılarak geliştirilmiş ve ticari olarak hizmet sağlayan alt yapı ve platform servisleri yeni bulut bilişim teknolojilerinin de geliştirilmesine öncülük yapmaktadır. Bu sayede başta akademi olmak üzere birçok sektörün ihtiyaçlarına uygun açık kaynaklı ve bulut bilişim teknoloji kullanan platformlar geliştirilmeye başlanmıştır. Bu alanda geliştirilen uygulamalardan bazıları ise; OpenStack, Eucalyptus, Apache Mesos'tur.

Bulut bilişim hizmetleri tür olarak özel (private), genel (public) ve hibrit (hybrid) şeklinde üç genel başlıkta değerlendirilebilir. Özel bulut bilişim teknolojileri herhangi bir işletmenin veri merkezinden kurum içi kullanıcılara hizmet sağlar. Bu tür bir teknoloji ile yönetim, kontrol ve güvenlik gibi gereksinimler kolaylıkla ve sorunsuz bir şekilde gerçekleştirilebilir. Genel bulut bilişim modelinde hizmetler harici bir sağlayıcı üzerinden internet üzerinden ulaştırılır. Amazon Web Services (AWS), Microsoft Azure, IBM'in SoftLayer'ı veya Google'ın Compute Engine teknolojisi genel bulut bilişim teknolojilerine örnek olarak gösterilebilir. Hibrit bulut teknolojisi is temelde genel (public) bulut hizmetlerinin bir karışımıdır. Bu şekilde kuruluşlar önemli uygulamaları özel bulut teknolojisi üzerinde çalıştırabilirken, kritik veriler üzerindeki kontrolü de kaybetmemiş olurlar.

2.2.3. Apache Hadoop ve Temel Bileşenleri

Büyük veri ile birlikte kullanılan diğer kavramlar çoğunlukla bu alanda kullanılan teknolojik gelişmeleri ifade etmektedir. Dağıtık hesaplama ve bulut bilişim hizmetleri bu teknolojilerin birkaçıdır. Bu kavramlar tanımlanırken hem kullanıcı hem de tasarımcı açısından farklı şekilde değerlendirilmektedir. Kullanıcıların sürecin işleyişi hakkında bilgisinin olduğu sistemlerde hizmetler harici bir otorite tarafından oluşturulur ve çalıştırılır. Kullanıcılar bu şekilde veri merkezinden depolamaya, işlemeye, ağ ve yazılım altyapısı oluşturmaya ve uygulamaya kadar birçok süreci kontrol edebilir veya dışarıdan bu hizmetleri temin etmeyi seçebilir. Tasarımcılar açısından dağıtık hesaplama, bulut bilişim veya büyük veri teknolojileri her düzeyde hizmet teklifleri sunmaya imkan sağlayan yazılım teknolojisini ifade etmektedir [125]. Örneğin, sanal makineler bulut bilişimin önemli bir temel bileşenidir; ancak bir uygulamayı açık hizmet olarak uygularken kullanılamazlar. Donanım ve yazılım hizmetleri, internet üzerinden hizmet sağlayan bir tedarikçiden temin edilir. Dağıtık hesaplama ve bulut bilişim teknolojileri kuruluşların bilgi işlem altyapılarını yalnızca öz kaynak kullanarak oluşturmak zorunda kalmadan, çalışma kaynaklarını bir yardımcı program olarak kullanabilmelerine imkan vermektedir.

Apache Yazılım Vakfı (Apache Software Foundation-ASF), 1999 yılında kurulmuş olan bireysel bağışlar veya kurumsal sponsorlar tarafından finanse edilen US 501(c)(3)

türü bir kuruluştur. ASF, proje taahhütünde bulunan geliştiriciler için muhtemel yasal dağıtımları sınırlayan fikri mülkiyet veya mali destek gibi konularda kurumsal bir çerçeve sağlamaktadır. ASF aynı zamanda dünya çapında milyonlarca kullanıcıya fayda sağlayan, ücretsiz olarak temin edilebilen kurumsal düzeyde yazılım ve projeler gibi binlerce yazılım çözümünü Apache lisansı altında dağıtmaktadır.

Büyük veri analizi için Apache lisansı ile servis edilen Hadoop, ölçeklenebilir (scalability) ve güvenilir (reliability) dağıtık hesaplama çözümleri içeren birden çok açık kaynaklı yazılımın bir araya getirilmesiyle oluşturulmuş bir çerçeve (framework)'dir. Dağıtık hesaplamada güvenilirlik (reliability) kriteri olası bir hatanın meydana gelmesi durumunda çalışan işi kesintisiz olarak devam ettirebilme yeteneğine, ölçeklenebilirlik (scalability) kriteri ise çalışan işlerin sayısının artması durumunda dahi bu talebe cevap verebilme yeteneğine vurgu yapmaktadır.

Apache Hadoop yazılım kütüphanesi, basit programlama modelleri kullanarak büyük veri kümelerinin dağıtık bilgisayar kümeleri üzerinde işlenmesine izin verir. Apache Hadoop aynı zamanda her biri yerel olarak hesaplama ve depolama hizmeti sunan tek bir sunucudan binlerce makineye kadar ölçeklenebilecek bir yapıda tasarlanmıştır. Yüksek düzeyde kullanılabilirlik sağlamak amacıyla yalnızca donanıma güvenmek yerine, yazılım kütüphanesinin kendisi uygulama katmanındaki muhtemel hataları algılamak ve işlemek için özel olarak tasarlanmıştır. Bu sayede her bir sunucu hataya açık (fault-tolerant) ve yüksek düzeyde kullanılabilirlik (high availability) sağlayacak şekilde hizmet sunabilmektedir [126].

İlk versiyonu 2006 yılında kamuoyuna duyurulan Hadoop, Doug Cutting ve Mike Cafarella tarafından geliştirilmiş ve Google'ın 2003 yılındaki Google dosya sistemi (Google File System-GFS) [127] ve 2004 yılındaki MapReduce [128] yaklaşımlarını temel alan bir yazılım çerçevesidir. Apache Hadoop projesinde her iki teknoloji açık kaynaklı olarak yorumlanmış ve ASF çatısı altındaki Apache Lucene ve Nutch projeleriyle birleştirilmiştir. Bu entegrasyondan sonra 2004 yılında Nutch dağıtık dosya sistemi (Nutch Distributed File System-NDFS), 2005 yılında ise MapReduce ortaya çıkmıştır. Sonraki yıllarda NDFS, HDFS olarak yeniden güncellenerek MapReduce ile birlikte Hadoop ismiyle tek çatı altında toplanmıştır. Hadoop ekosistemine ilerleyen yıllarda pek çok proje dahil olsa da Hadoop'un temelinde HDFS ve MapReduce yer almaktadır.

Bu tez çalışmasında Hadoop ekosistemi üç genel başlık altında incelenmiştir. Bunlardan birincisi HDFS ve alt bileşenleri, ikincisi MapReduce hesaplama paradigması, üçüncüsü ise büyük veri analizinde genel orkestrasyonu sağlayan ve kaynak yöneticisi olan Yet Another Resource Negotiator (YARN)'dır.

HDFS: Birden fazla sunucuyu biribirine bağlayarak ağ üzerindeki veriyi yönetmeye imkan sağlayan dosya sistemine dağıtık dosya sistemi denilmektedir. HDFS, herhangi bir donanım üzerinde çalışmak üzere tasarlanmış bu şekilde bir dağıtık dosya sistemidir. Sunucularda kullanılan dağıtık dosya sistemleriyle birçok benzerlik taşımaktadır ancak diğer sistemlerden farklı olarak hataya son derece dayanıklıdır ve düşük maliyetli donanım

ekipmanları dikkate alınarak tasarlanmıştır. Herhangi bir veri kümesi tek bir makinenin diskinde tutulamayacak kadar büyürse bu veri kümesi alt kümelere ayrılarak birden fazla makineye dağıtılması gereksinimi ortaya çıkar. HDFS, bu gereksinimleri karşılamak için ağ tabanlı bir dosyalama sistemi kullanır ancak ağ tabanlı programlamada karşılaşılan birçok olumsuzluklara karşı da açıktır [129].

HDFS'te meydana gelebilecek muhtemel donanımsal sorunlar istisnadan daha ziyade bu tür sistemlerde bir ön kabuldür. Bu nedenle bir çok bileşene sahip olması ve herhangi bir bileşende yüksek olasılıkta hata meydana gelebilmesi, HDFS'nin bazı bileşenlerinin her zaman işlevsel olamayacağı anlamına gelmektedir. Bu yüzden HDFS'nin temel mimari yapısı hataları hızlı tespit etme ve otomatik kurtarma prensibine göre tasarlanmıştır.

HDFS'deki tipik bir dosya boyutu gigabayt ila terabayt seviyelerindeki büyük dosyaları destekleyecek şekilde ayarlanmıştır. HDFS uygulamalarında veri dosyaları bir kez yaz-çok kez oku (write-once-read-many) erişim modeli kullanılarak HDFS'e aktarılır. Bu şekilde herhangi bir dosyanın oluşturulduktan, yazıldıktan ve kapatıldıktan sonra tekrar değiştirilmesi gerekmez. Bu yaklaşım veri tutarlılığı (data coherency) sorunlarını basitleştirir ve yüksek verimli bir veri erişimine imkan sağlar.

HDFS üzerinde çalışan uygulamalar, veri kümelerine akış erişimine ihtiyaç duyar. Burada akış erişimi, veri kümesinin iletilen kısmı dışındaki geriye kalan kısım hala aktarılmaya devam ederken veri kümesi üzerinde yeni bir işlemin başlatılmasına izin veren, herhangi bir bilgisayar ağı üzerinden sabit ve sürekli bir akış olarak veri iletme veya alma yöntemini ifade etmektedir. HDFS, kullanıcılara etkileşimli bir kullanım deneyimi sağlamadan ziyade daha çok toplu işlemede (batch processing) kullanılmak amacıyla tasarlanmıştır. Ayrıca HDFS, çok büyük boyutlu veri kümelerinde okuma (streaming) işlemini kolaylıkla gerçekleştirebilir ancak rastgele erişim (random access) özelliğine sahip değildir.

Veri analizi içeren uygulamalarda herhangi bir sürecin hesaplama işlemleri, üzerinde çalıştığı verilerin bulunduğu sunucunun yerel diskinde yürütülürse çok daha verimli bir şekilde gerçekleştirilmektedir. Bu tespit, özellikle veri kümesinin boyutunun çok büyük olduğu durumlar için geçerlidir. Bu sayede, sunucuların bulunduğu düğümler (node) arasındaki ağ trafiği en aza indirildiğinden sistemin genel verimi de artar. Buradaki temel varsayım, verileri uygulamanın çalıştığı yere taşımak yerine, hesaplamayı verilerin bulunduğu yere taşımanın genellikle daha iyi bir strateji olduğu şeklindedir. Bu amaçla HDFS'te uygulamaları verilerin bulunduğu yerel disklere taşıyabilmek için tasarlanmış arabirimler bulunmaktadır. HDFS'te bulunan veri kümeleri herhangi bir platformdan diğerine kolayca aktarılabilmektedir. Bu özelliği sayesinde HDFS geniş bir uygulama ölçeğine sahiptir.

HDFS, master (NameNode)-slave (DataNode) mimarisine sahiptir. Standart bir HDFS sunucu kümesinde tek bir ana sunucu (NameNode) bulunur. NameNode dosya sistemindeki ad alan (namespace) işlemlerini (örneğin; dosya ve dosya dizinlerini açma, kapatma veya yeniden adlandırma) yönetir ve istemciler (clients) tarafından oluşturulan

dosyalara erişimi düzenler. NameNode aynı zamanda HDFS mimarisinde yer alan blokların sunucu kümeleri üzerinde oluşturulması, silinmesi ya da herhangi bir sorun meydana gelmesi durumunda blokların yeniden oluşturulmasına kadar birçok süreçten sorumludur. HDFS üzerinde tanımlanmış tüm metaveri (metadata) NameNode tarafından yönetilir ve saklanır. DataNode, genellikle her sunucu kümesinde düğüm başına bir tane olacak şekilde, üzerinde çalıştıkları düğümlere bağlı depolamayı yönetir. HDFS'e aktarılan bir veri dosyası bir veya daha fazla bloğa bölünerek bir DataNode kümesinde depolanır. DataNode bu blokların oluşturulmasından saklanmasına kadar olan tüm süreci kontrol eder ve her bir DataNode kendi sunucusunun yerel diskindeki veri dosyalarından sorumludur. DataNode'lar ayrıca NameNode'dan gelen talimat üzerine blok oluşturma, silme ve çoğaltma (replication) işlemlerini gerçekleştirir. HDFS tarafından kullanılan tipik bir blok boyutu 64 megabyte (MB) veya 128 MB'tır. Böylece, bir HDFS dosyası 64 MB veya 128 MB'lık parçalara bölünür ve mümkünse her bir parça farklı bir DataNode'da bulunur.

MapReduce: MapReduce, büyük veri kümelerini hem oluşturmak hem de işlemek için geliştirilmiş bir programlama modelidir. MapReduce uygulamasında kullanıcılar, bir ara anahtar-değer çiftleri (key-value pairs) kümesi oluşturabilmek için üretilen herhangi bir anahtar-değer çiftini işleyen bir `map` (eşleme) fonksiyonu ve aynı ara anahtarla ilişkili tüm ara değerleri birleştiren bir `reduce` (azaltma) işlevi tanımlar. Bu şekilde yazılan programlar otomatik bir şekilde paralelleştirilerek büyük sunucu kümelerinde çalıştırılabilir. MapReduce modeli, giriş verilerinin bölütlenmesi (partition), programın bir dizi sunucuda yürütülmesinin (execution) zamanlanması, sunucuda meydana gelen hataların giderilmesi ve sunucular arasındaki iletişimi yönetme gibi aşamaları kontrol eder. Bu sayede, dağıtık hesaplama konusunda deneyimi olmayan kullanıcılar büyük bir dağıtık sistemin kaynaklarını kolay bir şekilde kullanabilirler [128].

MapReduce hesaplama işleminde bir dizi giriş anahtar-değer çiftine karşılık bir dizi çıkış anahtar-değer çifti üretilir. Kullanıcılar MapReduce kütüphanesinden faydalanarak dağıtık olarak gerçekleştirilecek hesaplama işlemi için `map` ve `reduce` adında iki fonksiyon tanımlar. Kullanıcı tarafından yazılan `map` fonksiyonu, bir girdi çifti alır ve bir dizi ara anahtar-değer çifti üretir. MapReduce kütüphanesi, I şeklinde tanımlanabilecek bir ara anahtar-değer ile ilişkili tüm ara değerleri bir araya toplar ve bunları `reduce` fonksiyonuna iletir. Kullanıcı tarafından tanımlanan ikinci fonksiyon olan `reduce`, I ara anahtar-değerini ve bu anahtar değerle ilişkili tüm değerleri kabul eder. Sonraki aşamada ise `reduce` fonksiyonu, mümkün olabildiğince daha küçük değerler kümesi oluşturabilmek amacıyla bu değerleri birleştirir. Her bir `reduce` işleminde tipik olarak yalnızca sıfır veya bir çıkış değeri üretilir. MapReduce işleminde üretilen ara değerler `reduce` fonksiyonuna bir yineleyici (iterator) üzerinden aktarılır. Bu işlem sayesinde, belleğe sığmayacak kadar büyük değer listelerinin işlenmesi mümkün olur. Kullanıcı programı MapReduce işlevini çağırdığında ardışık olarak gerçekleşen işlemler maddeler halinde aşağıda listelenmiştir:

- İlk aşamada `map` fonskiyonu, giriş verilerini bir dizi M alt kümeye otomatik olarak

bölerek birden çok makineye dağıtır. Bu sayede giriş verilerinin alt kümeleri farklı makineler üzerinden paralel olarak işlenebilir.

- İkinci aşamada `reduce` fonksiyonları, bir bölütleme fonksiyonu (partitioning function) kullanılarak ara anahtar uzayında R adet parçaya bölünerek dağıtılır. Bölütleme sayısı (R) ve bölütleme fonksiyonu kullanıcı tarafından belirlenir.

- Kullanıcı programındaki MapReduce kütüphanesi önce giriş verilerini her bir alt küme tipik olarak 16 MB'tan 64 MB'a kadar değişen bir aralıkla M adet parçaya böler (Bu değer bir parametre aracılığıyla kullanıcı tarafından kontrol edilebilir). Daha sonra ise programın birçok kopyasını sunucu kümesinde başlatır.

- Programın kopyalarından biri ana (master) sunucu olarak belirlenir. Geriye kalan kopyalar ise, ana sunucu tarafından işe atanan işçi (worker) sunuculardır. Böyle bir MapReduce işleminde atanacak M adet `map` görevi ve R adet `reduce` görevi bulunmaktadır. Daha sonra ana sunucu, kullanılmayan işçi sunucuları seçerek her birine bir `map` görevi veya bir `reduce` görevi atar.

- Bir `map` görevi atanan işçi sunucu, giriş verisinde kendisine karşılık gelen alt kümenin içeriğini okur. Sonraki aşamada ise giriş verilerden elde edilen anahtar-değer çiftlerini ayrıştırarak her bir çifti `map` fonksiyonuna iletir. Bu işlem esnasında `map` fonksiyonu tarafından üretilen ara anahtar-değer çiftleri arabellekte tutulur.

- Belirli aralıklarla arabelleğe aktarılan anahtar-değer çiftleri, bölütleme fonksiyonu tarafından R adet bölgeye bölünerek yerel diske yazılır. Bu çiftlerin yerel diskteki konumları ana sunucuya geri iletilir.

- Bir `reduce` işlemine atanan işçi sunucuda ana sunucu tarafından bu konumlar iletildiğinde, `map` işlemine atanan işçi sunucu tarafından yerel disklerden arabelleğe alınan verileri okumak için uzaktan yordam çağrıları (Remote procedure calls) oluşturulur. Sonraki aşamada `map` işlemine atanan işçi sunucu tüm ara anahtar-değer çiftlerini okuduğunda, aynı ara anahtar-değer çiftiyle ilişki tüm örnekleri aynı gruba toplayabilmek amacıyla bir sıralama işlemi gerçekleştirilir. Bu aşamada sıralama işlemi gereklidir, çünkü tipik bir MapReduce işleminde aynı `reduce` fonksiyonu birçok farklı ara anahtar-değer çiftini oluşurmak için kullanılır.

- `reduce` işlemine atanan işçi sunucu, elde edilen her bir benzersiz ara anahtar değeri için aynı işlemi yineler ve anahtarı ve karşılık gelen ara değerler kümesini kullanıcının `reduce` fonksiyonuna iletir.`reduce` fonksiyonunun çıktısı, bu indirgeme işlemi için oluşturulmuş son bir çıktı dosyasına eklenir.

- Tüm `map` görevleri ve `map` görevleri tamamlandığında, ana sunucu kullanıcı programına sonucu iletir. Bu aşamada, kullanıcı programındaki MapReduce çağrısı kullanıcı koduna geri döner.

- Tüm işlemler başarılı bir şekilde tamamlandıktan sonra, MapReduce uygulamasının çıktısı R adet çıktı dosyalarında bulunur (Bu aşamada her bir `reduce` görevi için bir adet çıktı dosyası oluşur). Kullanıcıların bu aşamada elde edilen R adet çıktı dosyasını tek bir dosyada birleştirmeleri gerekmez, çünkü bu dosyalar genellikle ya başka bir MapReduce uygulamasına girdi olarak iletirler veya bu dosyalar birden çok alt dosyaya bölünmüş girdiyle başa çıkabilen başka bir dağıtık hesaplama işleminde kullanılırlar.

YARN: Hadoop'un ilk versiyonunda MapReduce, hem veri işleme hem de kaynak yönetimi işlevlerini yerine getirmekteydi. MapReduce kaynak yönetimini ilk versiyonda bir tek ana sunucuda bulunan Job Tracker olarak adlandırılan bir alt bileşen üzerinden gerçekleştirmekteydi. Job Tracker bileşeni kaynak tahsisinden, planlamaya ve veri işleme iş akış takibine kadar birçok işlemi kontrol etmekteydi. Job Tracker bileşeni sonraki aşamada Task Trackers olarak adlandırılan bir başka bileşene çalışan uygulamanın alt süreçlerinde yer alan `map` ve `reduce` görevlerini atardı. Daha sonra ise Task Trackers bileşeni çalışan uygulamanın ilerleme raporunu periyodik olarak Job Tracker'a bildirirdi.

Bu tasarımda, tüm işlemler tek bir Job Tracker tarafından kontrol edildiği için ölçeklenebilirlik problemleri ortaya çıkmıştır. Ayrıca MapReduce'un ilk versiyonunda kullanılan tasarım, hesaplama kaynaklarının kullanımı açısından verimsizlik ve Hadoop çerçevesinin yalnızca MapReduce işleme paradigması ile sınırlı kalması gibi sorunları ortaya çıkarmıştır. Tüm bu sorunların üstesinden gelebilmek amacıyla 2013 yılında Yahoo ve Hortonworks işbirliği ile Hadoop'un ikinci sürümünde (Hadoop 2.0) YARN tanıtılmıştır [130]. YARN'ın geliştirilmesinin arkasındaki temel fikir, kaynak yönetimi ve iş planlama gibi sorumlulukları üstlenecek ve MapReduce'un üzerindeki işlemleri azaltacak bir ara katman oluşturmaktır. Bu sayede YARN, Hadoop'a MapReduce olmayan işleri de çalıştırma yeteneği kazandırmıştır. YARN aynı zamanda, gerçek zamanlı veri işlemek için Spark, SQL sorguları yazabilmek için Hive, NoSQL türü veri tabanları için ise HBase ve diğer birçok teknoloji ile birlikte işlemleri gerçekleştirebilmektedir.

YARN, temelde kaynak yöneticisi (Resource Manager), düğüm yöneticisi (Node Manager), Application Master ve konteyner (Container) şeklindeki dört farklı bileşenden oluşmaktadır [131].

Kaynak yöneticisi: Tek bir ana sunucu üzerinde çalışır ve kaynak yönetimini kontrol eder. Dağıtık hesaplama işlemi için gerekli kaynak yönetiminde nihai otoritedir. Kaynak yöneticisi, veri işleme isteklerini aldıktan sonra fiziki olarak istekleri gerçekleştirecek birimlerin ilgili düğüm yöneticilerine bu istekleri iletir. Kaynak yöneticisi, aynı zamanda dağıtık kümede kaynak aktarımı sırasında meydana gelebilecek uyuşmazlıkları çözebilmek amacıyla belirlenmiş karar merciidir ve mevcut kaynakları önceden belirlenen kurallar dahilinde birlikte çalışan uygulamalar için paylaştırır. Bu sayede, herhangi bir işlem için kaynaktan kapasite garantisi sağlama, adil kullanım ve hizmet seviyesi anlaşması (Service Level Agreements-SLAs) gibi çeşitli kısıtlamalara karşı tüm kaynakları her zaman kullanımda tutmak amacıyla kümenin kaynak kullanımını

optimize eder. Kaynak yöneticisinin iki temel bileşeni bulunmaktadır: zamanlayıcı (Scheduler) ve uygulama yöneticisi (Application Manager). Zamanlayıcı: Kapasite veya kuyruk (queue) vb. gibi kısıtlamaları dikkate alarak uygulamalara kaynak tahsis etmekten sorumludur. ResourceManager'da iş akış şemasını oluşturan asıl bileşendir ve uygulamalar için herhangi bir iş akış izleme veya durum takibi işlemi gerçekleştirmez. Ayrıca, herhangi bir uygulama hatası veya donanım hatası meydana gelmesi halinde, zamanlayıcı başarısız görevlerin yeniden başlatılmasını garanti etmez. Uygulama yöneticisi: Dağıtık küme üzerinde başlatılan işleri kabul eder ve her bir uygulamaya özel olarak oluşturulan uygulama yöneticisini yürütmek amacıyla kaynak yöneticisiyle konyetner tahsisinde uzlaşıyı sağlar. Bunun dışında, dağıtık hesaplama kümesinde Application Master'ın çalışmasından ve herhangi bir hata durumunda Application Master bileşeninin yeniden başlatılmasından sorumludur.

Düğüm yöneticisi: Hadoop kümesindeki her bir düğümde bir tane bulunur ve düğümdeki kullanıcı işlerini ve iş akışını yönetir. Kaynak yöneticisi ile birlikte çalışarak düğümün mevcut durumu (health status) hakkında bilgi paylaşır. Düğüm yöneticisinin birincil amacı, kaynak yöneticisi tarafından kendisine tahsis edilen konteyner'ları yönetmektir. Kaynak yöneticisini sürekli olarak düğümün mevcut durumu hakkında bilgilendirme yaparak güncel tutar. Application Master, uygulamanın çalışması için atanan konteyner'ı ve beraberinde gerekli her şeyi içeren bir konteyner başlatma içeriği göndererek düğüm yöneticisinden talep eder. Düğüm yöneticisi, bu işlem için istenen konteyner'ı oluşturur ve başlatır. Aynı zamanda, her bir konteyner'ın kaynak kullanımını (bellek, CPU) izler, log kayıtlarını düzenler ve kaynak yöneticisinin bilgisi dahilinde konteyner'da çalışan uygulamaları sonlandırabilir.

Application Master: Dağıtık kümede oluşturulan her bir uygulama, büyük veri çerçevesinde işlenen bir iş olarak değerlendirilir. Bu açıdan değerlendirildiğinde her bir uygulama, çerçeveye özel ve uygulamayla ilişkili benzersiz bir Application Master'a sahiptir. Application Master, tıpkı düğüm yöneticisi ve kaynak yöneticisi gibi herhangi bir uygulamanın dağıtık hesaplama kümesinde yürütülmesini koordine eden ve aynı zamanda hataları yöneten süreçlerde görev alır. Application Master'ın temel görevi, kaynak yöneticisiyle gerekli hesaplama kaynakları konusunda uzlaşı sağlamak, uygulamanın çalıştığı süre boyunca alt bileşenlerin görevlerini yürütmek ve izlemek için düğüm yöneticisi ile birlikte çalışmaktır. Application Master başlatıldıktan sonra, durumunu bildirmek ve kaynak taleplerinin kaydını güncellemek amacıyla kaynak yöneticisine periyodik olarak durum sinyali gönderir.

Konteyner: Tek bir düğümdeki RAM, CPU çekirdekleri ve yerel diskler gibi fiziksel kaynakların bir bütününü ifade eder. YARN konteyner'ları, konteyner yaşam döngüsü (Container Life Cycle-CLC) olarak tanımlanan bir konteyner başlatma ve sonlandırma içeriği tarafından yönetilir. Bu içerikte, ortam değişkenlerinin (environment variables) içeriği, uzaktan erişilebilir bir yerel diskin depolama biriminde bulunan bağımlılıkları (dependencies), güvenlik karakter dizgileri (tokens), düğüm yöneticisinde

çalışan hizmetler için yararlı yük (payload) ve işlemi oluşturmak için gerekli komutlar bulunmaktadır. Konteyner aynı zamanda, herhangi bir uygulama için belirli bir ana sunucu üzerindeki kaynağı (bellek, CPU vb.) kullanma imkanını sağlar.

2.2.4. Apache Spark ve Temel Bileşenleri

Son yıllarda büyük veride karşılaşılan hesaplama zorluklarını çözebilmek amacıyla çok sayıda düğüme ölçeklenebilecek ve Hadoop'un MapReduce'una alternatif olarak sunulabilecek birçok yeni sistem tasarlanmıştır. Büyük verinin çeşitlilik ve dağınıklık gibi doğal özelliklerinden dolayı standart bir veri işleme sürecinde dahi, veriyi çalışma ortamına aktarma, SQL sorguları çalıştırma veya makine öğrenmesi algoritmaları kullanabilmek için MapReduce benzeri bir yaklaşımı temel alan kodlara ihtiyaç duyulmaktaydı. Bu amaç doğrultusunda geliştirilen sistemler hem veri işleme süreçlerinde bir verimlilik artışına hem de daha fazla karmaşıklığın oluşmasına neden olmuştur. Kullanıcılar bu aşamada farklı sistemleri bir araya getirmeye ihtiyaç duymuş ancak bu uygulamaların bazılarının doğası gereği karmaşıklık düzeyi yüksek olduğundan herhangi bir çerçevede verimli bir şekilde ifade edilememektedir.

Yukarıda kısaca özetlenen problemlere çözüm önerisi olarak, 2009 yılında dağıtık veri analizi için bütünleşik bir çerçeve olarak tasarlanan Apache Spark projesi başlatılmıştır [132]. Spark, MapReduce'a benzer bir programlama modeli ve Resilient Distributed Datasets (RDD) adı verilen bir veri-paylaşım soyutlaması (data-sharing abstraction) kullanmaktadır [133]. Spark, böyle bir veri soyutlama eklentisi sayesinde herhangi bir harici motora (engine) ihtiyaç duymaksızın SQL, gerçek zamanlı veri akış analizi (streaming processing), makine öğrenmesi ve graf analizi dahil olmak üzere birçok uygulama için kullanılabilmektedir. Bu uygulamalar, özel motorlarda kullanılan optimizasyon yöntemlerine benzer yöntemleri kullandığından dolayı (örneğin; sütun odaklı işleme (column-oriented processing) veya kademeli güncelleme (incremental update), vs.) benzer performans değerleri elde edilmektedir. Tüm bu özellikler ortak bir motor üzerinden bir kütüphane olarak çalıştırıldığında ise bu işlemler hem ara süreçleri tasarlamayı kolay bir hale getirmiş hem de daha verimli bir veri işleme şeması oluşturmaya yardımcı olmuştur. Spark'ın bu şekilde bir genelleştirmeye sahip olmasının önemli faydaları bulunmaktadır. Bu faydalar kısaca maddeler halinde aşağıda listelenmiştir:

- Spark, bütünleşik ve tek bir uygulama programlama arayüzü (Application Programming Interface-API) kullandığı için uygulamaların geliştirilmesi alternatiflerine kıyasla daha kolaydır.
- Diğer sistemlerin verileri başka bir motora aktarabilmesi için depolama birimine yazması gerekirken, Spark birçok farklı işlemi aynı veri kümesi üzerinde gerçekleştirebilmektedir. Bu sayede Spark'ta farklı veri işleme süreçlerini birleştirmek daha verimlidir.
- Spark, diğer başka sistemlerde mümkün olmayan yeni uygulamalara (örneğin; bir graf

yapı üzerinde etkileşimli sorgular oluşturma veya canlı akışa sahip makine öğrenmesi gibi) olanak sağlamaktadır.

Spark'taki temel programlama soyutlaması RDD'lerdir. RDD'ler temel olarak, paralel olarak işlenebilen bir veri kümesinin bölümlere ayrılmış nesnelerinden oluşan ve hataya dayanıklı koleksiyonlarıdır. Bu sayede kullanıcılar, veri kümelerine dönüşümler (transformations) olarak adlandırılan (örneğin; `map`, `filter` veya `groupBy` gibi) işlemleri uygulayarak RDD'ler oluşturabilir. Kullanıcılar, herhangi bir veri kümesini RDD ve Scala, Java, Python ve R gibi farklı programlama dillerinde bulunan yerel fonksiyonları kullanarak dağıtık küme üzerinde paralel olarak işleyebilir.

Spark, RDD üzerinde yapacağı işlemleri verimli bir plan dahilinde gerçekleştirmek amacıyla tembel değerlendirme (lazy evaluation) yaklaşımını kullanmaktadır. Bu yaklaşım, adından da anlaşılacağı üzere, Spark'taki dönüşüm işlemleri için herhangi bir eylem (action) komutu gönderilene kadar yürütmenin başlamayacağı anlamına gelmektedir. Dönüşümler yapısı gereği tembel işlemlerdir, diğer bir deyişle RDD'de işlemler çağırıldığında yürütme işlemi hemen başlatılmaz. Bir eylem çağrıldığında, Spark uygun bir yürütme planı oluşturmak amacıyla kullanılan tüm dönüşüm grafiğini dikkate almaktadır. Örneğin; arka arkaya birden fazla `filter` veya `map` işleminin bulunduğu bir senaryoda, Spark bu işlemleri tek bir graf yapıda birleştirebilir. Kullanıcılar bu sayede herhangi bir performans düşüşü yaşamadan modüler yapıda programlar oluşturabilmektedir. RDD'ler aynı zamanda hesaplamalar arasında veri paylaşımı için açık destek sağlamaktadır. Spark'ın veri paylaşım özelliği MapReduce yaklaşımı ile arasındaki temel farklılıktır. RDD'ler varsayılan olarak bir eylemde (örneğin, `count` gibi) her kullanıldıklarında yeniden hesaplandıkları için geçici bir yapıya sahiptir, ancak kullanıcılar seçtikleri RDD'leri bellekte hızlı yeniden kullanım amacıyla sürdürmeye de devam edebilir.

Apache Spark'ta RDD üzerinde herhangi bir eylem başlatılacağı zaman yönlendirilmiş döngüsel olmayan grafik (Directed Acyclic Graph-DAG) olarak tanımlanan süreçte başlatılır. DAG yalnızca bir eylem çalıştırıldığında oluşturulur ve temel olarak köşelerden (vertices) ve kenarlardan (edges) oluşan bir kümedir. Burada köşeler oluşturulan RDD'leri, kenarlar ise RDD'ye uygulanacak işlemi temsil etmektedir. DAG'da her bir kenar, işlem dizisindeki bir önceki görevden (task) bir sonraki göreve doğru yönlendirilir. Bir eylem çağrısı sonrasında oluşturulan DAG, grafiğin alt görevlerini DAG zamanlayıcısına (DAG Scheduler) gönderir. DAG işlemleri, MapReduce'a kıyasla daha iyi global optimizasyon yapabilir. DAG kullanımının faydaları karmaşıklığın arttığı işlerde daha net bir şekilde görülmektedir.

Apache Spark'ın DAG yapısı sayesinde, kullanıcılar atanan görevlerin içeriklerine ve herhangi bir aşamadaki (stages) görevin ayrıntılarına ulaşabilme imkanını elde eder. DAG gösteriminde bu görevlerin alt aşamaları, o aşamaya ait tüm RDD'lerin ayrıntıları ve hangi işlemlerin uygulanacağı detaylı olarak betimlenmektedir. Zamanlayıcı, oluşturulan RDD'ye uygulanacak dönüşümleri dikkate alarak çeşitli aşamalara ayırır. Her aşama, aynı hesaplamayı paralel olarak gerçekleştirecek olan RDD'nin bölümlerine bağlı görevlerden

oluşmaktadır. Apache Spark'ın DAG yapısını kullanarak gerçekleştirdiği işlem basamakları kısaca aşağıda listelenmiştir:

- İlk aşama yorumlayıcıdır (örneğin; Scala, Java, vs.). Bu aşamada Spark çalıştırılacak kodu küçük bazı değişiklikler uygulayarak yorumlar.
- Spark, sonraki aşamada bir operatör grafiği oluşturarak bu grafiği DAG zamanlayıcı'ya gönderir.
- DAG zamanlayıcı, operatörleri görevin aşamalarına ayırarak ardışık düzene sahip bir yapıya dönüştürür (örneğin; `map` operatörleri tek bir aşamada birleştirilir).
- Bu aşamalar daha sonra görev zamanlayıcına (Task Scheduler) iletilir ve küme yöneticisi (örneğin; YARN, Mesos veya özerk sistemler (Standalone Systems)) aracılığıyla görevi başlatır.
- İşçi sunucular Spark üzerinden iletilen görevleri yürütür.

Veri paylaşımı ve paralel işleme özelliklerinin yanı sıra, RDD'ler ayrıca oluşabilecek hatalardan otomatik olarak kurtarılabilmektedir. Geleneksel dağıtık bilgi işlem sistemlerinde, veri kopyalama (data replication) veya kontrol noktası (checkpointing) gibi özellikler aracılığıyla belirli bir aralıkta hata toleransı elde edilebilmektedir. Spark, hata toleransını belirli bir aralıkta tutabilmek amacıyla kök (lineage) adı verilen farklı bir yaklaşım kullanmaktadır [133]. Her RDD, kendisini oluşturan dönüşümlerin DAG yapısını takip eder ve kaybolan bölümleri yeniden oluşturmak için DAG yapısında belirlenen bu işlemleri ana veri kümesi üzerinde yeniden çalıştırır (örneğin; bellek içi hata bölümünü tutan bir düğüm başarısız olursa, Spark filtreyi HDFS dosyasının ilgili bloğuna uygulayarak yeniden oluşturabilir). Spark'ın kök tabanlı veri kurtarma yaklaşımı, veri yoğun işlerde veri çoğaltma yöntemine kıyasla daha verimlidir. Bu yaklaşım ile ağ üzerinden veri yazmak, RAM'e yazmaktan çok daha yavaş olduğu için hem zamandan hem de bellekte depolama alanından tasarruf sağlamaktadır.

Spark, kalıcı depolama için birden fazla harici sistemle kullanılmak üzere tasarlanmıştır ve HDFS, Amazon S3 veya Cassandra gibi küme dosya sistemleriyle birlikte kullanılabilmektedir. Spark'ın depolama sisteminden bağımsız bir çerçeve olarak tasarlanması, kullanıcıların mevcut veri kümeleri üzerinde hesaplamaları yürütebilmesi ve farklı veri kaynaklarının sürece dahil edilmesi işlemlerini kolaylaştırmaktadır.

RDD yapısı, yalnızca dağıtık nesne koleksiyonları ve bunlar üzerinde çalışacak işlevler için tasarlanmıştır. RDD'lerin ana fikrinde "*verilerin düğümler arasında bölütlenmesi ve üzerinde çalışan dağıtık işlemlerin kontrol edilebilmesi sağlandığında, bu işlemleri gerçekleştirebilen yürütme tekniklerinin birçoğu diğer motorlar üzerinden de gerçekleştirilebilir*" yaklaşımı yer almaktadır. Kullanıcılar, farklı kütüphanelerin birlikte kullanıldığı durumlarda ise önemli bazı faydalar ve gelişmiş performans değerleri elde edebilmektedir.

Veri işlemede yaygın olarak kullanılan paradigmalardan biri veri tabanları üzerinde çalıştırılan ilişkisel sorgulardır. Spark SQL veya bir önceki versiyonu olan Shark, analitik veritabanlarında kullanılan tekniklere benzer yaklaşımlar kullanarak bu tür sorguları Spark üzerinde uygulamaya imkan sağlar [134]. Bu sistemlerde genellikle, sütun tabanlı depolama (columnar storage), maliyete dayalı optimizasyon ve sorgu yürütme için kod oluşturma gibi özellikler bulunmaktadır. Bu sistemlerin temelindeki ana fikir, RDD'ler içinde analitik veritabanları (sıkıştırılmış sütunlu depolama) ile aynı veri yapısını kullanmaktır. Spark SQL'de, RDD'deki her bir kayıt, ikili formatta (binary format) depolanan bir dizi satırın bilgisini tutar ve sistem, doğrudan bu yapı üzerinde çalıştırılacak sorguyu üretir.

Bir SQL sorgusu, başka bir programlama dili üzerinden çalıştırıldığında sonuçlar Spark'ın çalışma ortamına `DataSet` veya `DataFrame` formatında aktarılmaktadır. `DataSet`, Spark SQL'in optimize edilmiş yürütme motorunun ve RDD'lerin avantajlarını birlikte kullanan bir arabirimdir. `DataSet`, Java sanal makinesi (Java Virtual Machine-JVM) nesneleri kullanılarak oluşturulabilir ve üzerinde birçok farklı dönüşüm işlemi (`map`, `flatMap`, `filter` vb.) uygulanabilir. `DataSet` API'si, Scala ve Java'da mevcuttur ancak Python ve R dilleri bu API'yi desteklememektedir.

`DataFrame`, adlandırılmış sütunlar halinde düzenlenen bir `DataSet`'tir. Kavramsal olarak ilişkisel bir veritabanındaki bir tabloya veya R veya Python'daki bir veri çerçevesine (dataframe) eşdeğerdir, ancak veri işleme süreçlerinde kullanılabilecek daha zengin optimizasyon seçeneklerine sahiptir. `DataFrame`'ler, yapılandırılmış veri dosyaları, Hive'daki tabloları, harici veritabanları veya mevcut RDD'ler gibi çok çeşitli ve farklı kaynaklar üzerinden oluşturulabilmektedir. `DataFrame` API'si Scala, Java, Python ve R'da bulunmaktadır. Ayrıca, SQL dilinde oluşturulan sorgulardan farklı olarak, `DataFrame` işlemleri yüksek seviyeli programlama dillerinde (Python ve R gibi) oluşturulmuş fonksiyonlar gibi çağrılmaktadır. Spark'ın DataFrame'leri, tek düğümde çalışan kütüphanelere benzer bir API hizmeti sunar, ancak Spark SQL'in sorgu planlayıcısını kullanarak hesaplamayı otomatik olarak paralelleştirir ve optimize eder. Bu sayede kullanıcının çalıştırdığı kod, Spark'ın işlevsel API'si altında mevcut olmayan optimizasyonları kullanabilir.

Spark Streaming, ayrıklaştırılmış akışlar (discretized streams) adı verilen bir model kullanarak artımlı (incremental) gerçek zamanlı akış analizi gerçekleştirebilir. Spark üzerinden gerçek zamanlı akış uygulamalarında, giriş verileri küçük gruplara bölünerek (örneğin; her 200 milisaniyede bir gibi) yeni sonuçlar üretmek amacıyla RDD'lerde depolanan durumlarla düzenli olarak birleştirilmektedir. Gerçek zamanlı akış hesaplamalarını bu şekilde çalıştırmak, geleneksel dağıtık akış sistemlerine göre çeşitli avantajlara sahiptir. Örneğin, hata kurtarma (fault recovery) işlemleri, kök yaklaşımı sayesinde daha az maliyetli ve akışı toplu ve etkileşimli sorgularla birleştirmek daha kolaydır [135].

Spark GraphX, Pregel ve GraphLab'a benzer bir graf hesaplama arayüzüne sahiptir ve oluşturduğu RDD'lerde bölütleme işlevi için bu sistemlerle aynı yerleşim optimizasyonlarını

(köşe bölütleme şemaları gibi) uygular [136].

Spark'ın makine öğrenimi kütüphanesi olan Machine Learning Library (MLlib), dağıtık şekilde model eğitimi için 50'den fazla yaygın olarak kullanılan algoritmayı içermektedir. Spark MLlib'in temel görevi, pratik makine öğrenimi uygulamalarını ölçeklenebilir ve kolay tasarlanabilir bir yapıya dönüştürmektir. Yüksek düzeyde kullanım sağlamak amacıyla aşağıda listelenen pratik araçları sağlamaktadır:

- Makine öğrenmesi algoritmaları (Machine learning algorithms): sınıflandırma, regresyon, kümeleme ve ortak filtreleme gibi yaygın olarak kullanılan öğrenme algoritmaları.
- Özellik mühendisliği (Feature engineering): özellik çıkarma, dönüştürme, boyut indirgeme (dimension reduction) ve seçme.
- Veri işleme hattı (Pipeline): makine öğrenimi için gerekli veri işleme hattı oluşturmak, performans değerlendirmesi yapmak ve ince ayarlama yapmak amacıyla geliştirilmiş araçlar.
- Süreklilik (Persistence): algoritmaları, modelleri ve veri işleme hatlarını kaydetme ve yeniden yükleme.
- Yardımcı programlar (Utilities): lineer cebir, istatistik, veri işleme vb.

Spark MLlib, yaygın olarak kullanılan makine öğrenmesi algoritmalarının hızlı ve dağıtık işlemeye uygun versiyonlarını içermektedir. Bunlardan bazıları; sınıflandırma ve regresyon problemleri için çeşitli lineer modeller, Naif Bayes (Naive Bayes-NB) ve karar ağaçları (Decision Tree-DT) toplulukları, işbirlikçi filtreleme (collaborative filtering), kümeleme ve boyut indirgeme işlemleri için çeşitli algoritmalardır (örneğin; k-ortalamalar (k-means) kümeleme algoritması, temel bileşenler analizi (Principal Component Analysis-PCA). Kütüphane ayrıca dışbükey optimizasyon, dağıtık lineer cebir işlemleri, istatistiksel analiz ve özellik çıkarma, dağıtık makine öğrenmesi ve çeşitli tahmin uygulamaları için birçok optimizasyon yaklaşımını da içermektedir. Örneğin, karar ağaçlarında düğümler arası veri paylaşımı sırasında oluşan haberleşme maliyetlerini azaltmak için veriye dayalı özellik ayrıklaştırması gibi yaklaşımlar sayesinde karar ağacı toplulukları, hem ağaçların içinde hem de ağaçlar arasında öğrenmeyi paralel hale getirebilmektedir. Ayrıca genelleştirilmiş lineer modeller, çalışan hesaplamaları için hızlı C++ tabanlı lineer cebir kütüphanelerini kullanılarak gradyan hesaplamasını paralelleştiren optimizasyon algoritmalarını kullanarak öğrenme işlemini gerçekleştirebilir. Birçok algoritma bu verimli veri paylaşım ve haberleşme ilkellerinden yararlanmaktadır. Bu sayede özellikle ağaç yapısının kullanıldığı veri birleştirme yaklaşımı sayesinde yürütücü sunucuda çalışan programda veri akışı sırasında bir darboğazın oluşması önlenir ve Spark büyük modelleri çalışan işçi sunuculara hızlı bir şekilde dağıtabilir.

Geleneksel makine öğrenimini kullanan bir veri işleme hattında bir dizi veri ön işleme, özellik çıkarma, model uydurma ve doğrulama aşamaları bulunmaktadır. Çoğu makine

öğrenimi kütüphanesinde ise, veri işleme hattını oluşturabilmek için gerekli fonksiyonlar yerel olarak desteklenmez. Özellikle bu işlem büyük ölçekli veri kümeleriyle uğraşırken, sürecin başlangıçtan sona bir veri işleme hattı üzerinden bir araya getirildiği senaryoda, hem ağ trafiği açısından hem hesaplama yoğunluğu açısından pahalı bir işlemdir. Spark MLlib aynı zamanda, bu ihtiyaçları gidermek için tasarlanmış `spark.ml` adlı bir paket de içermektedir. Bu paket, tek tip ve üst düzey bir API kümesi sağlayarak çok aşamalı veri işleme hatlarının geliştirilmesini ve ayarlanmasını basitleştirmektedir [137].

Spark MLlib, Spark ekosistemindeki çeşitli bileşenlerden yararlanmaktadır. En düşük seviyedeki Spark çekirdeği, veri temizleme ve özellik çıkarımı gibi verileri dönüştürme işlemleri için 80'den fazla operatöre sahip genel bir yürütme motoru sağlamaktadır. MLlib aynı zamanda, Spark ile entegre diğer üst düzey kütüphanelerden de yararlanmaktadır.

3. Materyal ve Metot

Enerji iletim ve dağıtım hizmetleri endüstrisinde; arz-talep dengesi, müşteri tüketim örüntülerini çıkartma, elektrik kesintilerini önleme veya birim taahhüdü (unit commitment) problemini optimize etme gibi işlemler büyük verinin olası uygulamaları olarak gösterilebilir [112], [113]. Bu kapsamda hem iletim hem de dağıtım sistemi operatörlerinin araştırma ve geliştirme faaliyetlerini birleştirme ihtiyacı, Avrupa Elektrik İletim Sistemi Operatörleri Birliği'nin (European Network of Transmission System Operators for Electricity-ENTSO-E) 2017–2026 araştırma ve geliştirme yol haritasında da açık bir şekilde belirtilmektedir [138]. Bu açıdan değerlendirildiğinde, büyük verinin birçok uygulaması hem DSO'ların hem de iletim sistemi operatörlerinin (Transmission System Operator-TSO) günlük faaliyetlerini etkileyebilme potansiyeline sahiptir.

Bu tez çalışmasının ana yapısını oluşturan bileşenler alt bölümler halinde şu şekilde özetlenebilir: Bölüm 3.1.'de, üç farklı veri kümesi üzerinde keşifsel veri analizi (Exploratory Data Analysis-EDA) gerçekleştirilerek akıllı sayaç verilerinde karşılaşılan sorunlardan ve kısıtlamalardan bahsedilmiştir. Bölüm 3.2.'de, Hortonworks veri platformu üzerinden Apache Hadoop ve Spark çerçeve yazılımlarının kurulum basamakları gösterilmiştir. Ayrıca, veri paylaşımı için gerekli protokoller ve ayarlamalar, H_2O.ai ve Microsoft Machine Learning for Apache Spark (MMLSpark) gibi sistemlerin entegrasyonları da bu aşamada gerçekleştirilmiştir. Bölüm 3.3.'te, DL tabanlı tahmin uygulaması için kullanılan yöntemlerin, Bölüm 3.4.'te ise, dağıtım hattı seviyesinde gerçekleştirilen kayıp-kaçak tespiti uygulamasında kullanılan yöntemlerin matematiksel altyapısı açıklanmıştır. Son olarak Bölüm 3.5.'te, her iki uygulama için tez kapsamında önerilen yöntemlerden ve Bölüm 3.6.'da ise, performans ölçütlerinden bahsedilmektedir.

3.1. Veri Kümeleri

Akıllı şebeke uygulamalarında veri kümesi kavramı, belirli bir dönem boyunca örneklenen elektrik verilerinin toplanmasıyla oluşan koleksiyonlar olarak tanımlanabilir. Bu koleksiyonlar akıllı şebeke uygulamasına göre farklı parametrelerin bir araya getirilmesiyle oluşturulabilir.

AMI sistemlerde yük profili, tüketilen enerji miktarının zamana karşı değişimini gösteren bir zaman serisidir. Bu zaman serilerinde örneklenen veri sayısı sayacın örnekleme süresine (15, 30, 60 dakikalık periyotlar gibi) göre farklılık gösterebilir ancak tüm yük profilleri 24 saatlik (1 günlük) bir zaman aralığını ifade etmektedir. Bu tez çalışmasında kullanılan veri kümelerinde yer alan her bir örnek, bu şekilde oluşturulmuş bir günlük yük profiline karşılık gelmektedir.

Bu tez çalışmasında üç farklı veri kümesi kullanılmıştır. Bu veri kümelerinin iki tanesi Dicle Elektrik Perakende Satış Anonim Şirketi (DEPSAŞ) üzerinden 6698 sayılı Kişisel Verilerin Korunması Kanunu ("KVKK" veya "Kanun") kapsamında tanımlanan kısıtlamalar ve gizlilik esasları dikkate alınarak temin edilmiş, üçüncü veri kümesi ise

Avustralya hükümetinin enerji verimliliği girişiminde (Energy Efficiency Initiative) yer alan SGSC programı üzerinden temin edilmiştir. SGSC, herkese açık bir veri kümesiyken (public dataset), DEPSAŞ üzerinden temin edilen veri kümeleri gizlilik sözleşmesine tabiidir.

Bu tez çalışması kapsamında gerçekleştirilecek analizlerin ortak bir eksende değerlendirilebilmesini kolaylaştırmak amacıyla, veri kümelerinde ortak parametreler kullanılmıştır. Bu ortak parametreler sırasıyla; tüketici numarası, tarih bilgisi, hangi tüketici grubunda yer aldığı (mesken, ticarethane veya resmi kurum gibi) ve Kilowatt saat (kWsa) cinsinden tüketim değerleridir. Veri kümelerinde ortak şekilde kullanılmayan parametreler ise ilgili veri kümelerinin detaylı olarak açıklandığı bölümlerde tanımlanmıştır.

3.1.1. Dicle Elektrik Perakende Satış A.Ş. (DEPSAŞ) Veri Kümesi

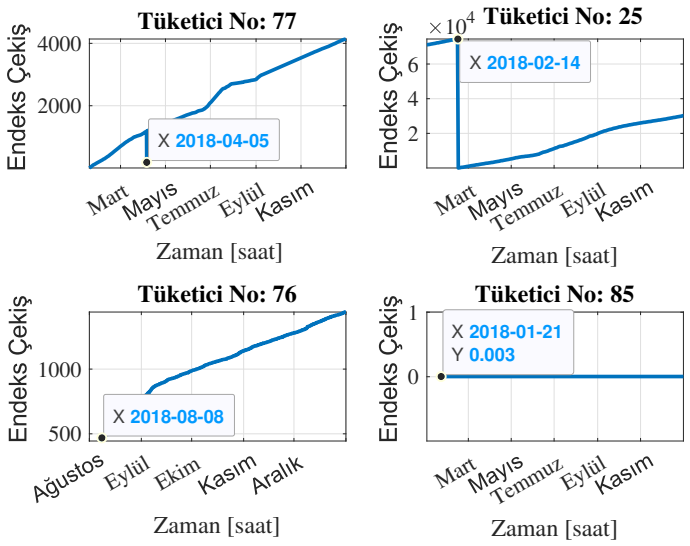
DEPSAŞ veri kümesi, rastgele şekilde seçilmiş toplamda 100 farklı tüketicinin 2018 yılına ait endeks çekiş değerlerinden oluşmaktadır. Veri kümesinde 1-Ocak-2018 00:00:00 ile 31-Aralık-2018 23:59:00 tarihleri arasında kaydedilmiş toplamda 1.872.994 ölçüm değeri bulunmaktadır. Veri kümesinde endeks çekiş değerleri dışında ayrıca tüketici numarası, çarpan katsayısı, ölçüm tarihi ve tüketici sınıfı bilgileri yer almaktadır. Veri kümesinde bulunan 100 tüketicinin 56'sı mesken, 44'ü ise ticarethane sınıfında yer almaktadır.

DEPSAŞ veri kümesinde, analizlerin daha sağlıklı gerçekleştirilebilmesi amacıyla aynı şehir içerisinde farklı konumlarda bulununan sayaçlar kullanılmıştır. Bu işlem, örneklem seçiminde sıkça karşılaşılan istatistiksel yanlılık (statistical bias) veya uzamsal otokolerasyon (spatial autocorrelation) gibi problemlerin etkilerini en aza indirebilmek amacıyla gerçekleşirilmiştir.

Veri kümesinde farklı örnekleme sürelerine (örneğin; 15, 30, 60 dakikalık) sahip sayaçlar bulunduğu için günlük yük profillerinde de farklı miktarda veri bulunmaktadır (örneğin; 24, 48, 96). Bu tez çalışmasında, sayaçların farklı örnekleme sürelerine sahip olması tüm veri kümesinin tek bir veri matrisinde birleştirilebilmesini zorlaştırdığı için her bir tüketiciye özgü tek bir matris düzenlenerek işlemler gerçekleştirilmiştir. DEPSAŞ veri kümesini ön işleme aşamasında karşılaşılan diğer zorluklar ise aşağıda listelenmiştir:

- Sayacın çeşitli nedenlerle yanlış ölçüm aldığı durumlarda endeks değerinin farklı okunması.
- Veri tabanı üzerinden yapılan sorgularda belirlenen tarih aralığında aynı sayacın iki farklı tüketicinin değerlerini içermesi.
- Sayaçtan uzun süreli ölçüm alınamaması.
- Çok düşük seviyelerde kaydedilmiş endeks çekiş değerleri.

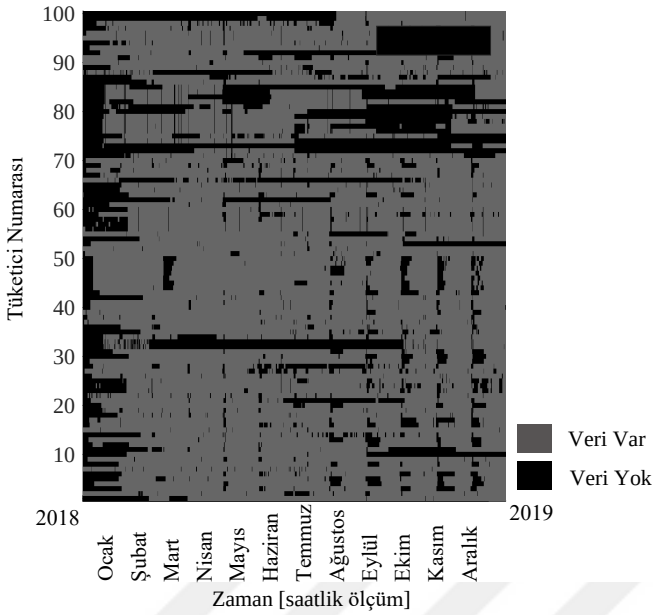
Yukarıda kısa özetlenen dört farklı sorunun veri kümesinde karşılaşılan bazı örnekleri Şekil 3.1.'de verilmiştir.



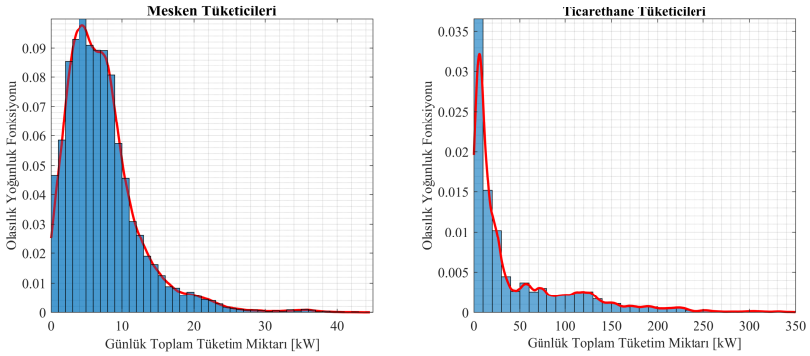
Şekil 3.1. Dicle Elektrik veri kümesinde karşılaşılan problemler

Dağıtım şirketi üzerinden akıllı sayaç veri kümesi temini aşamasında, sahada karşılaşılan en büyük zorlukların başında sayacın ölçüm alamaması veya verinin sayaç veri yönetim sistemine iletilememesi gelmektedir. Bu problem özellikle AMI sistemlerin yeni kurulduğu bölgelerde sıkça karşılaşılan bir durumdur. Bu aşamada veri analistlerinin izleyeceği stratejiler ve uygulayacağı yöntemler büyük bir önem taşımaktadır çünkü kayıp veri yerleştirme (data imputation) işlemleri sırasında atanacak varsayımsal tüketim değerleri veri kümesinde yanlılık problemine neden olabilmektedir. Şekil 3.2.'de DEPSAŞ veri kümesindeki tüm tüketicilerin 60'ar dakika olarak örneklendiği senaryoda elde edilen boşluklar gösterilmiştir.

Şekil 3.2.'den de anlaşıldığı üzere uzun dönemli kayıp veriler ve boşluklar, veri kümesinde sıkça karşılaşılan ve veri ön işlemede başlangıç şartları etkileyen önemli bir durumdur. Bu tez çalışmasında bu sebeplerden dolayı, kayıp değerlere sahip yük profilleri tamamen veri kümesinden çıkarılmıştır. Halka açık akıllı sayaç veri kümelerinin birçoğunda çok düşük seviyelerde tüketim gerçekleştiren tüketici örnekleriyle az karşılaşıldığından, bu problem literatürde çok az bahsedilen bir olgudur. Ayrıca, sıfır (0) veya sıfıra çok yakın tüketim değerlerinden oluşan yük profilleri istatistiksel olarak anlamlı sonuçlar üretemez. Bu çalışmada, tıpkı veri kümesinden kayıp değerlere sahip yük profillerinin tamamen çıkartılmasına benzer bir yaklaşım çok düşük seviyerlerde tüketim gerçekleştiren tüketiciler için de uygulanmıştır. Bu nedenle, yıllık toplam tüketim değeri 500 kW'tan düşük olan tüketiciler veri kümesinden çıkartılmıştır. Şekil 3.3.'te mesken ve ticarethane tüketicilerinin günlük toplam elektrik tüketim değerlerinin histogramı ve olasılık yoğunluk fonksiyonu verilmiştir. Akıllı sayaç veri kümeleri, ayda bir ölçüm alınarak oluşturulan veri kümelerine kıyasla daha yüksek zaman çözünürlüğüne sahiptir. Günlük yük profillerinden elde edilen toplam tüketim değerleri, farklı zaman eksenlerinde



Şekil 3.2. Dicle Elektrik veri kümesinde bulunan boşluklar

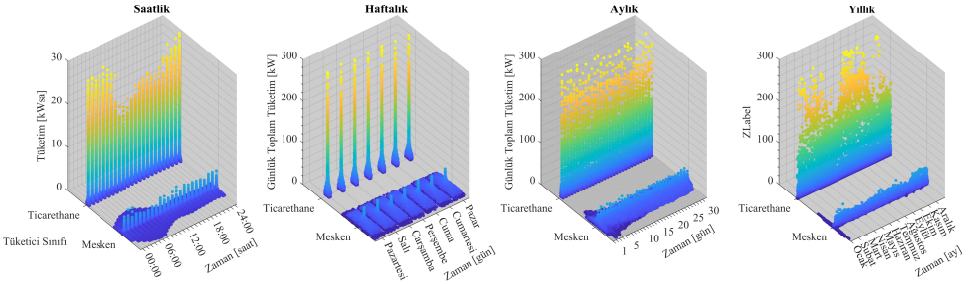


Şekil 3.3. Mesken ve ticarethane tüketicilerinin günlük toplam tüketimlerinin histogramı

incelendiğinde önemli tüketim örüntülerinin ortaya çıkarılmasına katkı sağlayabilir. Şekil 3.4.'te tüketim değerlerinin zaman eksenindeki saatlik, haftalık, aylık ve bir yıllık tüketim dağılımları gösterilmiştir. DEPSAŞ veri kümesinden eksik veri veya çok düşük seviyeli tüketim değerleri içeren yük profilleri çıkartıldıktan sonra geriye veri ön işleme ve tahmin uygulamasında kullanılmak üzere 22639 farklı yük profili kalmıştır. Bu işlemler sonrasında ise yük profillerinin $\approx$ 38'i ilk etapta veri kümesinden çıkarılmıştır.

3.1.2. Smart Grid Smart City (SGSC) Veri Kümesi

Akıllı şebekede dağıtım seviyesinden tüketici seviyesine geçişte karşılaşılan en temel problemlerden birkaçı, tüketim seviyelerinde karşılaşılan çeşitlilik ve değişkenliktir. Avustralya hükümetinin SGSC projesi, akıllı şebekede karşılaşılan bu problemlere çözüm



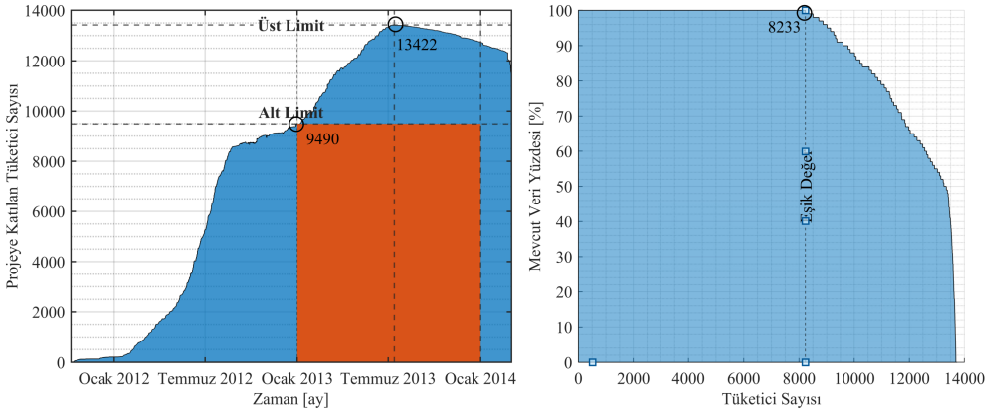
Şekil 3.4. Elektrik tüketiminin ticarethane ve mesken abonelerinde günlük, haftalık, aylık ve yıllık olarak değişimi

üretmek amacıyla geliştirilmiş ilk ticari ölçekli uygulamasıdır [139]. SGSC projesi, Avustralya hükümetinin yanısıra Ausgrid, EnergyAustralia ve konsorsiyum ortakları IBM Australia, GE Energy Australia, Sydney Water ve Newcastle City Council gibi kuruluşlar tarafından da finanse edilmiştir. Proje Ekim 2010'da başlamış ve 28 Şubat 2014'e kadar devam etmiştir. Projenin nihai raporları ile birlikte halka açık veri kümeleri, 28 Temmuz 2014'te yayınlanmıştır.

Elektrik tüketim verileri ve diğer ilişkili parametreler, projeye gönüllü olarak katılan tüketicilerden temin edilmiştir. Proje aynı zamanda, Avustralya ile bağlantılı demografik bilgileri, elektrikli ev cihazları kullanımı, iklim, perakende ve distribütör ürün teklifleri gibi farklı konular hakkında da ayrıntılı bilgiler içermektedir. Bu veri kümesini oluşturan veri kaynakları sırasıyla: belirli zaman aralıklarıyla kaydedilmiş sayaç okumaları, tüketicilerin evleri hakkındaki tanımlayıcı bilgiler, HAN üzerinden iletimi gerçekleştirilen priz okumaları, azami talep dönemlerinde (peak demand period) karşılaşılan olaylar ve sistemin yanıt süresi (response), teklif ve kabul süreçleridir. SGSC projesi, Dünya'daki akıllı şebeke altyapısının en büyük ve en kapsamlı ticari ölçekli olarak halka açık veri kümesi sağlayan girişimlerden bir tanesidir.

Veriler, büyük ölçüde Newcastle ve Sidney şehirlerinden ve belirli akıllı şebeke uygulamalarını test etmek için seçilmiş kırsal alanlardan toplanmıştır. Newcastle bölgesi, coğrafi ve iklim koşullarından, sosyoekonomik ve demografik yapısından kaynaklanan önemli bölgesel ve banliyö özelliklerini birlikte içerdiği için projenin odak noktalarından bir tanesi olarak seçilmiştir. Bu nedenle Newcastle'da yaşayan tüketicilerin demografik ve sosyoekonomik özellikleri, tipik bir Avustralya şehrinin demografik özelliklerini yakından yansıtmaktadır.

SGSC veri kümesi toplamda 13735 tüketicinin 2010-2014 yılları arasındaki tüketim değerlerini ve ilişkili diğer parametrelerini içermektedir. Veri kümesinde her tüketicinin projeye katılma ve ayrılma tarihi farklılık göstermektedir, bu yüzden veri kümesinde her bir tüketiciden dört yıllık süre boyunca veri temin edilememiştir. Bu tez çalışmasında SGSC veri kümesindeki tüketici numarası, zaman bilgisi, kWsa cinsinden tüketim miktarı parametreleri kullanılmıştır. Veri kümesinde toplam 344.518.791 adet sayaç okuması bulunmaktadır ve örnekleme süresi yarım saatlik (00:30:00) aralıklar olacak şekilde



Şekil 3.5. SGSC veri kümesi örnek seçimi için belirlenen en uygun tarih aralığı ve bu aralıkta yera alan tüketicilerin yüzde (%) cinsinden kullanılabilir veri miktarı

belirlenmiştir. Şekil 3.4.'te SGSC veri kümesinde yer alan ticarethane ve mesken abonelerinin elektrik tüketimleri günlük, haftalık, aylık ve yıllık olarak değişimi verilmiştir. Bu tez çalışmasında SGSC veri kümesinin kullanımı aşağıda kısaca özetlenen süreçlere katkı sağlamaktadır:

- Günümüzde tüketicilerin elektrik kullanım bilgisi tıpkı sosyal medyada veya diğer platformlarda muhafaza edilen bilgilere benzer bir yapıya sahip olduğu için dağıtım şirketleri geniş ölçekli tüketici verisi paylaşma konusunda tereddütler yaşamaktadır.
- Bu tez çalışmasının ana yapılarından birini oluşturan büyük hacimli sayaç verisini işleme ve analiz etme işlemleri SGSC veri kümesi sayesinde mümkün olabilmiştir.
- DEPSAŞ veri kümesi ile karşılaştırıldığında elde edilen benzerlikler ve farklılıklar ortaya çıkarılmıştır.
- Veri temini aşamasında yapılan ön kabullerin etkisi incelenerek uygun ayıklama, dönüştürme ve yükleme (Extract, Transform, Load-ETL) süreçleri tasarlanmıştır.

Şekil 3.5.'te SGSC veri kümesinde örnek seçimi için en uygun tarih aralığı ve bu aralıkta yera alan tüketicilerin % cinsinden kullanılabilir veri miktarı verilmiştir. Şekil 3.5.'ten de anlaşıldığı üzere dört yıllık program süresi içerisinde en fazla tüketicinin dahil olabileceği tarih aralığı [1 Ocak 2013 - 1 Ocak 2014] arasındaki turuncu bölgedir.

SGSC veri kümesinde DEPSAŞ veri kümesinde uygulanan benzer örnek seçimi stratejileri uygulanmıştır. Veri kümesine, çok düşük tüketimli ve % cinsinden kullanılabilir veri miktarı 2013 yılı için % 100'ün altında olan tüketiciler dahil edilmemiştir. Bu ön kabul sonrasında SGSC veri kümesinde yer alan tüketici sayısı ise 7063'tür.

3.1.3. Kayıp-Kaçak Veri Kümesi

Kayıp-kaçak işlemi için kullanılacak veri kümesi DEPSAŞ'tan temin edilen ikinci veri kümesidir. Bu veri kümesi ilk veri kümesinden farklı olarak belirli bir bölgede aynı trafo merkezine bağlı tüketicilerden oluşmaktadır. Bu tez çalışmasında veri gizliliği nedeniyle daha önceden kaydedilmiş ve etiketlenmiş kayıp-kaçak örnekleri yerine varsayımsal örnekler üretilerek analizler gerçekleştirilmiştir. Bu varsayımsal örnekler gerçek hayatta karşılaşılan durumların matematiksel denklemlerinden türetilmiştir.

AMI seviyesinde meydana gelen kayıp-kaçak işlemleri üç farklı açıdan değerlendirilebilir: sayaca yapılan fiziksel müdahaleler (physical intrusions), sayacın donanım veya yazılım altyapısına yapılan izinsiz erişimler (hardware or firmware tampering) veya iletişim altyapısına yapılan saldırılar (communication infrastructure attacks). Bu tez calışmasında, günümüzdeki dağıtım şebekelerinde sık karşılaşılan bir durum olan yetersiz iletişim altyapısı nedeniyle üçüncü kategoride yer alan senaryolar incelenmiştir. Ayrıca, kayıp-kaçak kullanan tüketicilerin tespiti ve yerinde incelenmesi maliyetli seçenekler olduğundan ve bu tüketici gruplarının veri kümelerindeki azınlık davranışı, kayıp-kaçak tespiti uygulamalarında daha düşük sınıflandırma doğruluğuna yol açtığından dolayı bu kategori tercih edilmiştir. Tüketicilerin bu kategoride işlem gerçekleştirebilmesi için donanım veya yazılım ekipmanını manipüle etmek gibi daha karmaşık becerilere sahip olmaları gerekir ve bu işlemleri geleneksel araçlarla gerçekleştirmek mümkün değildir.

Akıllı sayaçların ölçümlerini değiştiren siber saldırılar yaygın olarak FDI olarak adlandırılmaktadır. Bu yöntemler sayesinde kayıp-kaçak kullanmayı hedefleyen bir tüketici, sayaçlara yapılan fiziksel müdahalelere benzer davranışları benzeştirerek akıllı sayaçların rapor ettiği elektrik tüketim miktarını azaltır. FDI temelli kayıp-kaçak kullanımında enerji tüketiminin hesaplanması ve raporlanması sürecini etkileyen birçok müdahale bulunmaktadır. Bu işlemlerde asıl amaç, herhangi bir alarmı tetiklemeden rapor edilen güç tüketimini manipüle etmektir [94], [140]. Tablo 3.1.'de, yaygın olarak kullanılan altı FDI müdahalesinin resmi bir tanımı ve orijinal tüketime karşılık gelen matematiksel formülasyonları verilmiştir.

Tablo 3.1.'deki, $x, x_t, \widetilde{x}_t$ değişkenleri sırasıyla veri noktasını, orijinal ve manipüle edilmiş yük profilini temsil etmektedir. t, t_1, t_2 parametreleri her örnek için rastgele şekilde belirlenmektedir ve $f(t)$ fonksiyonu $[0, 1]$'li sıralı bir çıktı üretmektedir. Tablo 3.1.'de verilen altı FDI müdehalesi aşağıdaki gibi tanımlanabilir:

- FDI1: Tüm saldırı süresi boyunca aynı faktör kullanılarak yük profilinin saatlik tüketimi azaltılır.
- FDI2: Her bir farklı yük profilinde daha önceden rastgele şekilde belirlenmiş bir eşik değerin üzerindeki saatlik okumalar kırpılır ve değerlerin geri kalanı aynı kalır.
- FDI3: Her yük profilinin tepe (peak) değerinden daha düşük bir eşik değer belirlenir ve bu eşik değer rapor edilen tüm tüketim değerlerinden çıkarılır.

Tablo 3.1. Literatürde karşılaşılan altı farklı FDI saldırısının matematiksel denklemleri [14], [94], [140].

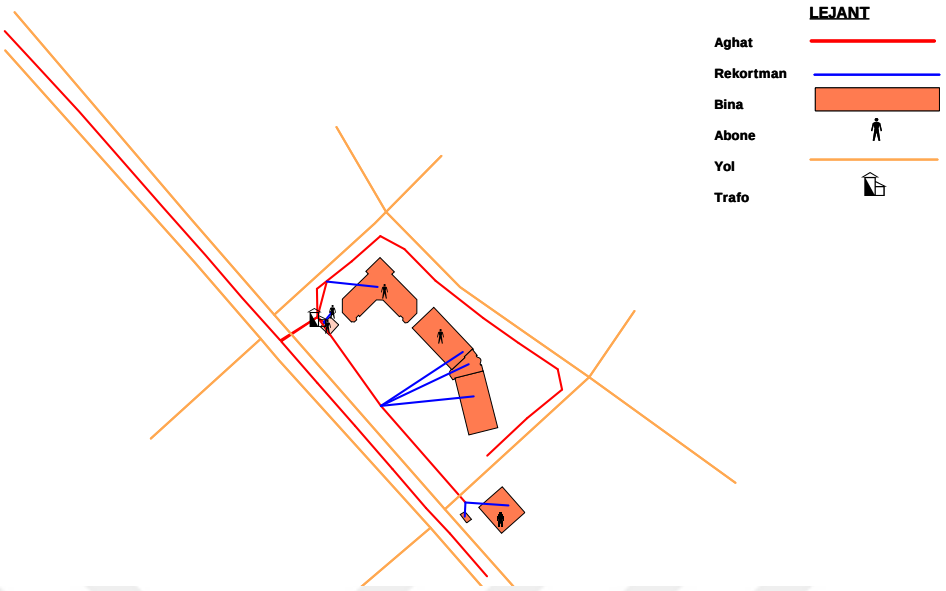
Yöntem	Değişiklik	Ön Koşullar
FDI1	$\widetilde{x}_t \leftarrow \alpha \cdot x_t$	$0.2 < \alpha < 0.8$ α : tüm saatlik okumalar için aynı değer
FDI2	$\widetilde{x}_t \leftarrow \begin{cases} x_t, & \text{eğer } x_t \leq \gamma \\ \gamma, & \text{yoksa} \end{cases}$	γ: rastgele bir kesme noktası $\gamma < \max x$
FDI3	$\widetilde{x}_t \leftarrow \max\{x_t - \gamma, 0\}$	γ rastgele bir kesme noktası $\gamma < \max x$
FDI4	$\widetilde{x}_t \leftarrow f(t) \cdot x_t$	$f(t) \leftarrow \begin{cases} 0, & \text{eğer } t_1 < t < t_2 \\ 1, & \text{yoksa} \end{cases}$ $t_1 - t_2$: dört saatten uzun rastgele bir zaman aralığı.
FDI5	$\widetilde{x}_t \leftarrow \alpha_t \cdot x_t$	$0.2 < \alpha_t < 0.8$ aralığında rastgele oluşturulmuş bir katsayı
FDI6	$\widetilde{x}_t \leftarrow \alpha_t \cdot \bar{x}$	$0.2 < \alpha_t < 0.8$ aralığında rastgele oluşturulmuş bir katsayı $\bar{x}$: günlük yük profilinin ortalama tüketimi

- FDI4: Dört saatten uzun bir zaman aralığındaki bir yük profilinin tüm tüketim değerleri sıfır ile değiştirilir.
- FDI5: Bir yük profilinin her bir tüketim değeri, 0.2 ile 0.8 arasında rastgele olarak belirlenmiş bir katsayı ile çarpılarak değiştirilir.
- FDI6: Bir yük profilinin her değeri için belirlenmiş bir katsayı, günlük yük profilinin ortalama tüketim değeri ile çarpılır.

Veri kümesi, 1 Ocak 2019 - 31 Aralık 2019 tarih aralığında kaydedilmiş 15, 30 ve 60 dakikalık zaman çözünürlüğüne sahip 31 akıllı sayaç verisi ve 1 gözlemci sayaç (observer meter) verisinden oluşmaktadır. Veri kümesi tüketici numarası, tüketici sınıfı, zaman bilgisi ve kWsa cinsinden tüketim değerlerini içermektedir. Veri kümesindeki 31 akıllı sayacın 28'i mesken, 2'si okul ve 1'i aydınlatma'dır. Akıllı sayaç veri kümesinde toplam 949.477 ölçüm, gözlemci sayaç veri kümesinde ise 16.470 ölçüm değeri bulunmaktadır. Bu çalışmada FDI tabanlı kayıp-kaçak senaryolarının oluşturulduğu trafo istasyonunun CBS çıktısı Şekil 3.6.'da verilmiştir.

Veri kümesindeki her bir tüketicinin sayaç verisi farklı zaman çözünürlüğüne sahip olduğu için saatlik toplam akıllı sayaç verisi ile aynı saatteki gözlemci sayaç verisi karşılaştırılamamaktadır. Bu çalışmada bu nedenlerden dolayı tüm sayaçlar saatlik olarak yeniden örneklenmiştir. Bu tez çalışmasında ayrıca hem dağıtım düzeyinde hem de tüketici düzeyinde tüketim örüntülerini derinlemesine analiz edebilmek için bazı zorunlu ön kabuller gerekmektedir. Bu ön kabuller, nihai veri kümesinin oluşumuna katkıda bulunan temel veri dönüşümlerini ve ön işleme adımlarını içermektedir. Bu çalışmada, veri analizi aşamasında oluşabilecek olası sorunları en aza indirmek için aşağıdaki gibi bazı ön koşullar oluşturulmuştur:

- Günlük bazda kullanılabilir veri miktarı % 95'ten fazla olmalıdır.
- Veri analizine dahil edilen her tüketicinin günlük toplam tüketimi 0.5 kW'tan yüksek olmalıdır.

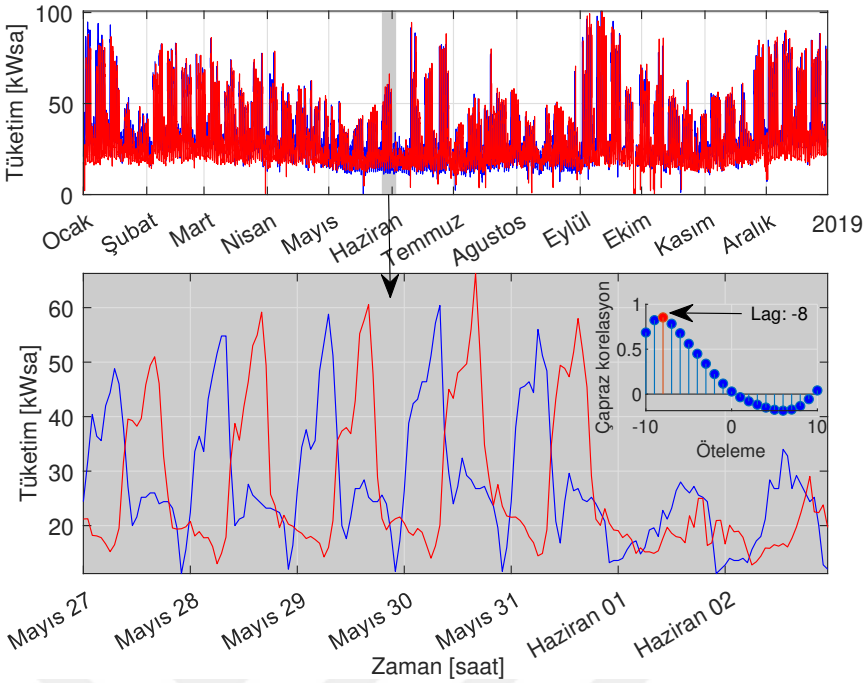


Şekil 3.6. FDI tabanlı kayıp-kaçak senaryolarının oluşturulduğu Diyarbakır ilindeki pilot bölgenin CBS çıktısı

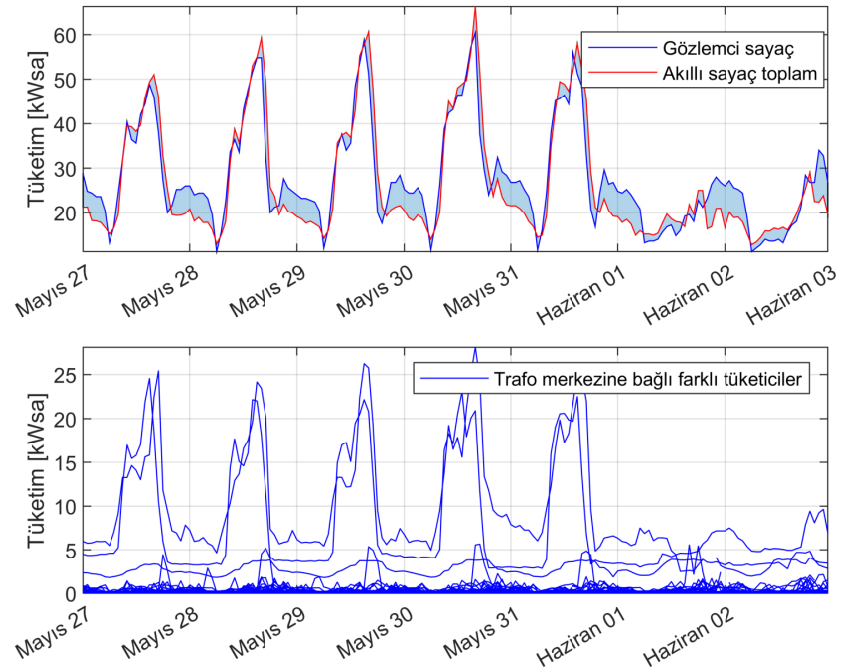
- Yük profillerindeki saatlik okumalar dakikalık olarak yeniden örneklendirilmelidir.

Yukarı kısaca açıklanan ön koşullar sağlandıktan ve eksik değerlere sahip yük profilleri veri kümesinden çıkartıldıktan sonra, veri kümesi, satırları farklı günlük yük profillerine karşılık gelen ve sütunların dakikalık zaman aralıklarına karşılık gelen 7807 $\times$1381 boyutlu bir matristir. Veri matrisinde sayı olmayan değerleri (Not a Number-NaN) içeren günlük yük profilleri çıkarıldıktan sonra matrisin satır sayısı 7807'ye eşittir. Günlük yük profillerinin [00:00:00-23:00:00] zaman aralıkları arasındaki tüketim değerleri doğrusal enterpolasyon kullanılarak yeniden örneklendikten sonra matrisin sütun sayısı ise 1381'e eşittir. Bu işlem sayesinde, kısa zaman aralıklarında meydana gelen FDI örneklerinde bilgilendirici ve saklı bulunan örüntülerin çıkartılması kolaylaşmıştır.

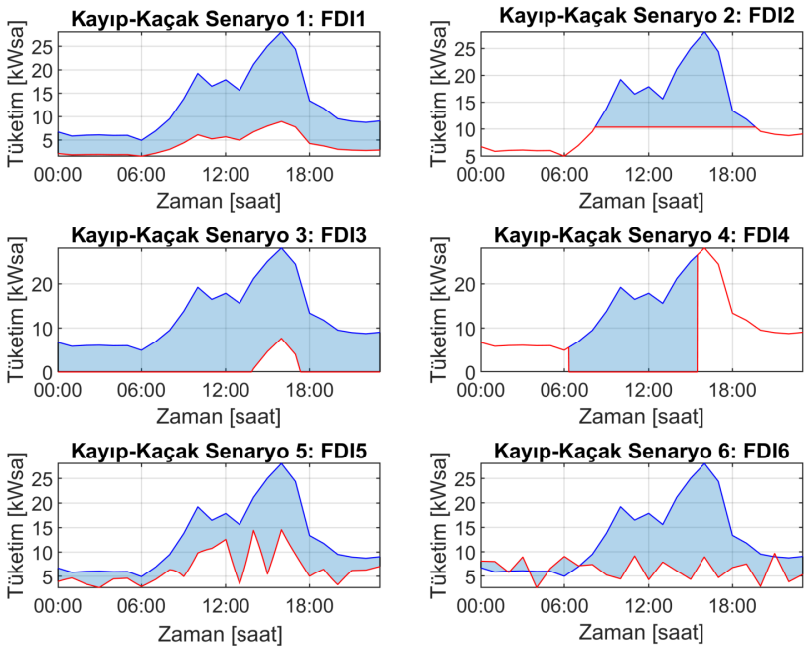
Trafo merkezi ölçümleri ve bu trafo merkezine bağlı akıllı sayaç toplam ölçümleri ideal olarak hemen hemen aynı olmalıdır, ancak teknik ve teknik olmayan kayıplar [82]'de belirtildiği gibi ölçümler arasında gözle görülür farklılıklara yol açabilir. Ayrıca, AMI altyapısının yeni gelişmekte olduğu bölgelerde akıllı sayaçlar ve gözlem sayaçlar arasında zaman senkronizasyonu problemleri de meydana gelmektedir. Şekil 3.7.'de, trafo merkezi ölçümleri ile akıllı sayaç toplam ölçümleri arasında sekiz saatlik bir gecikme gözlemlenmiştir. Şekil 3.8.'de gösterildiği gibi ölçümler arasındaki gecikme (lag=0), gözlemci sayaç verileri zaman ekseninde ötelenerek düzeltilse bile, veri kümesinde yine de farklı nedenlerden kaynaklanan, örtüşmeyen ve tutarsız boşluklar bulunmaktadır. Ayrıca, Şekil 3.8.'de gösterildiği üzere farklı tüketim seviyelerinde bulunan tüketicilerin aynı trafo merkezine bağlı olduğu durumlarda, gözlemci sayaç verilerini düşük seviyeli yüklerin kayıp-kaçak keşif süreçlerine dahil etmek kolay olmayabilir, çünkü yüksek seviyeli yükler trafo merkezi toplam yükü üzerinde daha belirleyecidir.



Şekil 3.7. Kayıp-kaçak senaryolarının oluşturulması aşamasında karşılaşılan akıllı sayaç ile gözlemci sayaç arasındaki zaman senkronizasyonu problemleri



Şekil 3.8. Kayıp-kaçak veri kümesinde farklı nedenlerden kaynaklanan akıllı sayaç ve gözlemci sayaç ölçümleri arasındaki tutarsız boşluklar



Şekil 3.9. Rastgele seçilen bir yük profiline FDI saldırılarının uygulanması

Bu bölümünde açıklanan FDI saldırıları, akıllı sayaç veri kümelerinde kayıp-kaçak örneklerini elde etmek için kullanılmıştır. FDI tabanlı kayıp-kaçak senaryolarını oluşturmak için kullanılan stratejiler aşağıda özetlenmiştir:

- Belirlenen rastgele bir tarihte rastgele bir tüketici seçilerek gerçek yük profiline rastgele seçilen bir FDI yöntemi uygulanmıştır.
- Her bir yöntemin veri kümesindeki dağılımının eşit olması ve her bir tüketici için toplam yük profili sayısının yarısını geçmemesi sağlanmıştır.
- Değiştirilen tüm yük profillerinin yüzdesi, toplam veri kümesinin % 30'unu geçmemiştir.

Rastgele seçilmiş bir yük profiline FDI tabanlı kayıp-kaçak senaryoları uygulandıktan sonra değiştirilmiş yük profilleri Şekil 3.9.'da gösterilmektedir.

3.2. Dağıtık Hesaplama Ortamı Bileşenleri ve Kurulumu

Bu tez çalışmasındaki tahmin ve sınıflandırma işlemlerinin tümü sanal Linux işletim sistemleri üzerinde gerçekleştirilmiştir. Bu işlemleri gerçekleştirebilmek için ilk olarak Oracle şirketinin VirtualBox'ı (versiyon=6.1.12) varsayılan ayarlar kullanılarak Windows 10 işletim sistemi üzerine kurulmuştur. Ayrıca, Windows işletim sistemi üzerinden uzak Linux sunucuya erişebilmek, secure shell protocol (SSH) bağlantısı kurabilmek, Linux

komut satırında (command shell) çalışabilmek, Windows makine ile sanal Linux işletim sistemi arasında dosya paylaşımı sağlayabilmek amacıyla PuTTY ve WinSCP programları da varsayılan ayarlar kullanılarak kurulmuştur. Dağıtık hesaplama ve büyük veri analizi için ise Cloudera şirketinin Hortonworks veri platformunun (Hortonworks Data Platform-HDP) Sandbox'ı (versiyon=2.6.5) daha öncesinde oluşturulan sanal Linux işletim sistemi üzerine kurulmuştur.

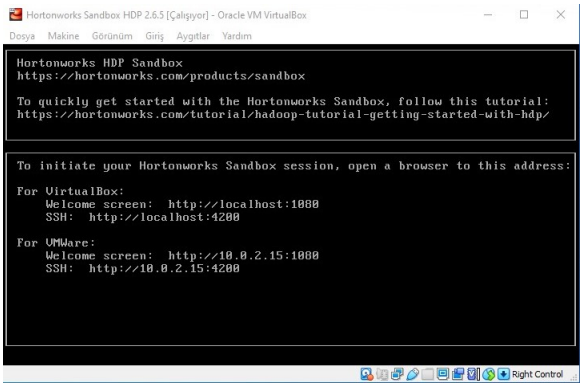
İndirme ve kurulum işlemleri tamamlandıktan sonra ise, yetkilendirme, veri paylaşım protokollerini düzenleme, büyük veri ekosistemiyle entegre farklı platformları birleştirme gibi işlemler gerçekleştirilmiştir. Büyük veri analizinde kullanılacak yöntemlerde ortaya çıkabilecek farklı ön gereksinimler ve versiyon uyumsuzlukları da göz önünde bulundurularak HDP Sandbox 2.6.5 versiyonu dışında farklı sanal Linux sunuculara Hadoop ve Spark bileşenleri ayrı ayrı da kurulmuştur.

3.2.1. Hortonworks Veri Platformu

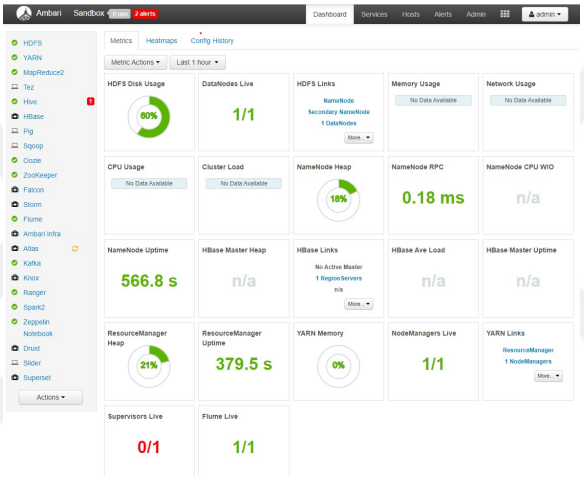
HDP, Apache Hadoop tarafından desteklenen büyük hacimli verileri depolamak, işlemek ve analiz etmek için hizmet sağlayan ölçeklenebilir ve % 100 açık kaynaklı bir platformdur. Birçok farklı kaynaktan ve formattan gelen verileri çok hızlı, kolay ve uygun maliyetli bir şekilde analiz etmek için tasarlanmıştır. HDP, büyük veri'nin depolanması ve işlenmesi amacıyla ortaya çıkan operasyonel, güvenlik ve yönetim işlerine odaklanan temel ASF projelerinden oluşmaktadır. Bu projelerin bazıları MapReduce, HDFS ve YARN ile birlikte Ambari, Falcon, Flume, HBase, Hive, Kafka, Knox, Oozie, Phoenix, Pig, Ranger, Slider, Spark, Sqoop, Storm, Tez ve ZooKeeper'dır. Bu projeler HDP sürümlerinde sürecinin bir parçası olarak entegre edilerek ve test edilerek kurulum ve yapılandırma araçları ile birlikte kullanıma sunulmuştur [141]. HDP, ayrıca sanal veya bulut platformlarında (örneğin; VMware vSphere veya AWS EC2) çalıştırılabilmektedir.

HDP'de yer alan projelerin varsayılan ayarları ve ortam değişkenleri kurulum sırasında otomatik olarak oluşturulduğu için ek ayarlamalara ihtiyaç duyulmamaktadır. HDP Sandbox, sanal makine üzerinde ilk kez çalıştırıldığında ise bir defaya özgü olarak tüm yetkili kullanıcısı (`root`) ve Ambari yöneticisi (`admin`) için şifre sıfırlama işlemi gerçekleştirilmektedir. Sonraki aşamada yerel arama motorundan Ambari arayüzüne ulaşabilmek için ise yerel sunucuda (`localhost`) daha öncesinden belirlenen IP adresi (örneğin; `127.0.0.1`) ve port numarası (örneğin; `8080`) kullanılmaktadır. Oracle VirtualBox'ta HDP Sandbox başlatıldığında elde edilen ekran çıktısı Şekil 3.10.'da verilmiştir.

HDP'de dağıtık hesaplama kümesinde yönetici Apache Ambari'dir ve küme ile etkileşim bu bileşen üzerinden gerçekleştirilmektedir. HDP Sandbox, VirtualBox üzerinden çalıştırıldıktan sonra Ambari yöneticisi (`admin`) üzerinden giriş yapılır ve tüm servisler otomatik olarak başlatılır. Apache Ambari arayüzü ve büyük veri bileşenleri başlatıldıktan sonra elde edilen ekran çıktısı Şekil 3.11.'de verilmiştir.



Şekil 3.10. Oracle VirtualBox üzerinde çalışan HDP Sandbox'ın ekran çıktısı



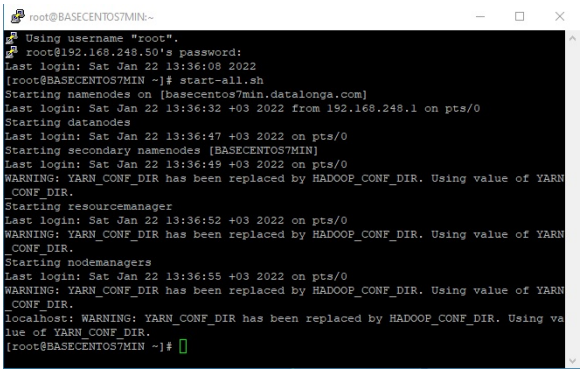
Şekil 3.11. Apache Ambari arayüzü ve büyük veri bileşenleri başlatıldıktan sonra elde edilen ekran çıktısı

3.2.2. Linux İşletim Sistemine Apache Hadoop ve Spark Kurulumu

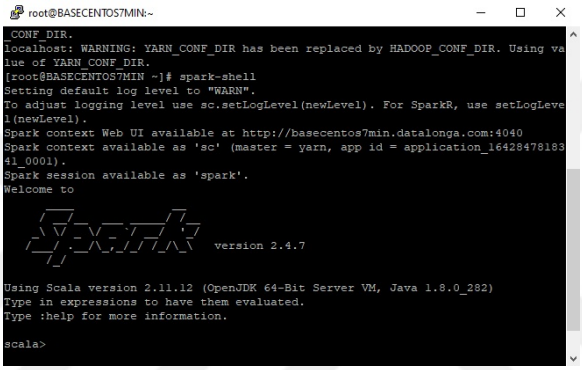
HDP Sandbox 2.6.5 sürümünde varsayılan olarak Apache Hadoop 2.7.3 ve Apache Spark 2.3.0 versiyonları bulunmaktadır. Bu versiyonların birbiri arasındaki uyumluluğu dağıtık hesaplama sürecinde gerçekleştirilen işlemlerin hatasız olarak tamamlanabilmesi için önemlidir. Özellikle Spark MLlib'de bulunmayan makine öğrenme algoritmalarının dağıtık şekilde işlem yapabilmesi için uygun Hadoop ve Spark versiyonların belirlenmesi gerekmektedir.

Bu tez çalışmasında HDP Sandbox haricinde sanal bir Linux işletim sistemine Apache Hadoop ve Apache Spark bileşenleri ayrı şekilde kurularak ikinci bir dağıtık hesaplama ortamı oluşturulmuştur. Apache Hadoop ve Apache Spark'ın sanal Linux işletim sistemi üzerine kurulumu ve gerekli ayarlamalar Bölüm 5. EK-2'de detaylı olarak verilmiştir.

Kurulumun ilk aşamasında sanal Linux Centos 7 işletim sistemi VMware Tools üzerine yüklenmiştir. Bu aşamada sanal makinede ihtiyaç duyulacak her türlü yazılımları yükleme ve ağ konfigürasyonlarını belirleme işlemleri gerçekleştirilmiştir (örneğin; yum, curl, unzip,



Şekil 3.12. Sanal Linux işletim sisteminde Hadoop ve bileşenlerinin başlatılması



Şekil 3.13. Sanal Linux işletim sisteminde Spark ve bileşenlerinin başlatılması

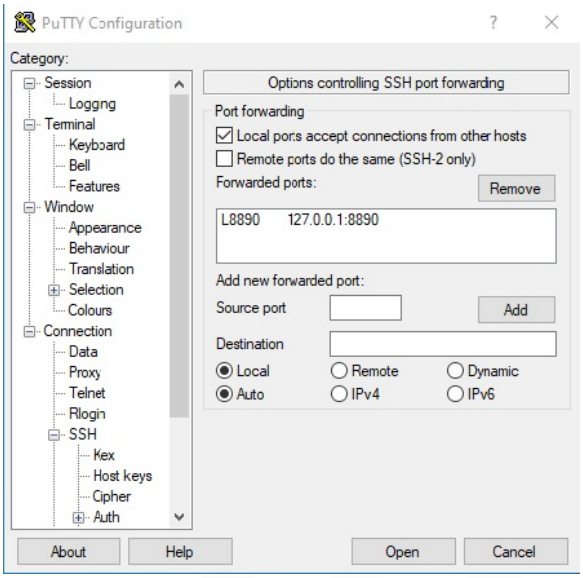
tar, wget, OpenSSL, Java (1.8 ve üzeri), ssh, ntp gibi yazılımları yükleme veya IPv4, IPv6, Nameservers'ları belirlemek). Kurulum işlemi bittikte sonra Linux işletim sistemi üzerinde çalıştırılan Hadoop ve Spark bileşenleri sırasıyla Şekil 3.12. ve Şekil 3.13.'te verilmiştir.

İkinci aşamada ise büyük veri analizi için gerekli yazılımlar yüklenmiş ve ortam değişkenleri tanımlanmıştır. Bu aşamada ayrıca sanal makinede kullanılacak programlama dili olarak Python belirlenmiş (örneğin;versiyon=2.7.X ve 3.6.X) ve kurulum işlemleri tamamlanmıştır.

3.2.3. Veri Paylaşım Protokolleri ve Gerekli Ayarlamalar

Anaconda Jupyter notebook, tüm programlama dilleriyle etkileşimli bilgi işlem için ücretsiz yazılım, açık standartlar ve web hizmetleri sunan bir platformdur. Jupyter notebook kod geliştirme dışında görselleştirme, multimedya, işbirliği ve daha farklı amaçlar için de kullanılmaktadır. Bu çalışmada, her iki sanal işletim sistemine kurulan Hadoop ve Spark bileşenleri Jupyter notebook aracılığıyla kullanılmaktadır. Veri paylaşımı için gerekli protokoller, HDP Sandbox'ta `hdfs` kullanıcısı üzerinden, VMware'de kurulu Centos 7 işletim sisteminde ise `root` kullanıcısı üzerinden tanımlanmıştır.

Jupyter notebook, sanal Linux işletim sistemlerine kurulduğundan dolayı ana



Şekil 3.14. Ana makine ile sanal Linux işletim sistemi arasında yapılan SHH tünelleme işlemi

makine (Windows 10 işletim sistemi) üzerinden çalışma ortamına erişebilmek için PuTTY üzerinden SHH tünelleme (SSH Tunneling) işlemi gerçekleştirilmiştir. Bu işlem Windows 10 ana makine ile sanal Linux makine arasında yerel bağlantı kuralabilmek ve mevcut portları yönlendirebilmek (Local Port Forwarding) amacıyla gerçekleştirilmiştir. Şekil 3.14.'te ana makine ile sanal Linux işletim sistemi arasında gerçekleştirilen SHH tünelleme işleminin ekran çıktısı verilmiştir.

Veri kümeleri, her iki sistemde de ilk olarak sanal makinenin yerel diskine daha sonra ise HDFS'e aktarılmaktadır. Sonraki aşamada ise veri kümeleri, Jupyter notebook üzerinde başlatılan bir Spark Session aracılığıyla HDFS üzerinden çalışma ortamına Spark Dataframe olarak aktarılabilmektedir.

3.2.4. H_2O.ai Entegrasyonu

H_2O.ai , yazılım desteği sağlayarak yaygın makine öğrenmesi algoritmalarının iş dünyasının pratik uygulamalarında kullanılmasına imkan sağlayan bir platformdur. Aynı zamanda H_2O.ai , finans sektöründen sağlık sektörüne kadar birçok farklı sektörün karmaşık sorunlarını çözmek için yapay zeka ve DL algoritmalarının dağıtımına liderlik yapan açık kaynaklı bir platformdur [142].

Bellek içi (in-memory) sıkıştırmayı kullanan H_2O.ai, küçük bir kümede bile bellekteki milyarlarca veri satırını işleyebilme kabiliyetine sahiptir. H_2O.ai platformu, veri analitiği iş akışları oluşturmayı kolaylaştırmak için Python, R, Java, Scala, JSON ve CoffeeScript/JavaScript gibi programlama dilleri için arabirimlere sahiptir ve yerleşik bir web arabirimi olan `Flow`'dan oluşmaktadır. H_2O.ai, özerk (standalone) modda, Hadoop'ta veya bir Spark kümesinde çalışabilecek şekilde tasarlanmıştır ve genellikle birkaç dakika içinde devreye girebilmektedir.

H_2O.ai, doğrusal regresyon, lojistik regresyon (Logistic Regression-LR), NB, PCA, k-ortalama kümeleme ve word2vec gibi birçok yaygın makine öğrenimi algoritmasını içermektedir. H2O aynı zamanda, karar ağaçları ormanı (Random Forest-RF), gradyan artırma makinesi (Gradient Boosting Machine-GBM) ve DL gibi makine öğrenme algoritmalarını dağıtık şekilde çalıştırabilmektedir. H2O ile kullanıcılar bu makine öğrenmesi algoritmaları sayesinde çok sayıda model oluşturabilir veya en iyi tahminleri elde edebilmek amacıyla bu yöntemlerin sonuçlarını karşılaştırabilir.

Sparkling Water, kullanıcıların H_2O.ai'nun hızlı, ölçeklenebilir makine öğrenimi algoritmalarını Spark'ın yetenekleriyle birleştirebilmesine olanak tanıyan bir ara birimdir. Bu iki açık kaynaklı platform entegre edildiğinde ise, Spark kullanarak veri ön işleme işlemlerini gerçekleştirmek, model oluşturmak için sonuçları H_2O.ai'ya iletmek ve tahmin sonucu üretmek gibi görevler, kullanıcılar için sorunsuz bir deneyime dönüşmektedir. Sparkling Water, H_2O.ai algoritmaları için Spark'ta oluşturulan iş-akış kurallarını takip eder, böylece Spark görevlerine aşina olan kullanıcılar için API'yi başlatmak kolay bir işlemdir. Sparkling Water, Spark veri yapıları (RDD, DataFrame, Dataset) ile H_2O.ai'nun `H2OFrame`'i arasında ve tam tersi şekilde veri dönüşümü sağlayabilmektedir. Ancak, `H2OFrame`'de depolanan veriler yüksek derecede sıkıştırılabildiğinden, verinin dönüştürme işlemi sonrasında RDD formatında korunmasına gerek yoktur [143].

Sparkling Water'ın ana bileşeni `H2OContext`'tir. `H2OContext` dağıtık kümenin mevcut durum bilgisini tutar ve veri formatlarını dönüştürme işlemi için temel özellikler sağlar. `H2OContext`, `SparkSession`, `SparkContext` veya `SQLContext` gibi Spark öğelerinin tasarım prensiplerine benzer özelliklere sahiptir. `H2OContext`'in başlatılması aşamasında aşağıda listelenen bir dizi işlem gerçekleştirilir:

- Sunucu üzerinde yürütüldüğü durumunda, görev dağıtılır ve tüm erişilebilir Spark yürütücü (executor) düğümleriyle iletişim kurulur. Daha sonra ise yürütücülerin JVM'leri içindeki H_2O.ai hizmetleri başlatılır.
- Sunucu dışında oluşturulan bir küme üzerinde çalıştırıldığı durumunda ise, YARN üzerinden H_2O.ai kümesini başlatır ve ona bağlanır veya yapılandırma ayarlarına göre mevcut H_2O.ai kümesine bağlanır.

Bu çalışmada, H_2O.ai (`versiyon=3.20.0.1`) ve Sparkling Water (`versiyon=2.3.7`) bileşenleri HDP Sandbox üzerinde çalışan Hadoop ve Spark bileşenleri ile entegre edilmiştir. Dağıtık hesaplamada kaynak yöneticisi olarak ise YARN belirlenmiştir.

3.2.5. SynapseML Entegrasyonu

SynapseML (önceki adıyla MMLSpark), ölçeklenebilir makine öğrenimi modellerinin oluşturulmasını basitleştirmek amacıyla geliştirilmiş açık kaynaklı bir yazılım kütüphanesidir. SynapseML, Spark'ın makine öğrenme modellerine Microsoft bilişsel araç seti (Microsoft Cognitive Toolkit-CNTK), LightGBM ve OpenCV ile sorunsuz

entegrasyonu dahil olmak üzere Spark ekosistemine birçok DL ve veri bilimi aracını eklemeye imkan sağlar.

Bu çalışmada sadece SynapseML'nin LightGBM algoritması Spark ile entegre edilerek kullanılmıştır. Bu entegrasyon ise VMware Tools üzerine yüklenen Centos 7 işletim sisteminde yer alan Spark (`versiyon=2.4.7`) ve Hadoop (`versiyon=3.1.2`) bileşenleri kullanılarak gerçekleştirilmiştir. Sonraki aşamada ise Microsoft'un LightGBM için geliştirdiği projenin ".jar" uzantılı paketleri Spark Session'a dahil edilmiştir. Burada Spark Session'a dahil edilen MMLSpark'ın (`versiyon=0.18.1`) uygun Maven kordinatları belirlenmiş ve işlemler Jupyter notebook üzerinde gerçekleştirilmiştir.

3.3. Derin Öğrenme Tabanlı Yük Tahmini

Akıllı şebekelerde DL tabanlı yük tahmini, hem dağıtım seviyesinde hem de tüketici seviyesinde son yıllarda sıklıkla tercih edilen uygulamalardır. Günümüzde kullanılan güç sistemleri planlaması ve işletilmesinde yakın geçmiş zamanda gözlemlenen değerler kullanılarak yapılan ekstrapolasyon işlemi yakın gelecek tahmini için yeterli bir yöntem olarak değerlendirilmemektedir. Akıllı şebeke uygulamalarında üretim planlama, yük yönetimi ve sistem kararlılığı gibi birçok operasyonel işlem yük tahmini uygulamasına dayanmaktadır. Ayrıca geçmiş dönem tüketim verilerinin yanında, hava durumu verileri, periyodiklik içeren veya insan temelli olaylar gibi diğer değişkenler, enerji arz-talebini belirlemek amacıyla gelecekteki güç tüketimini tahmin etmeye yardımcı olmaktadır.

Geleneksel yük tahmini uygulamalarında Bölüm 3.1.'de de büyük ölçüde açıklandığı üzere veri toplama ve düzenleme işlemi sonrasında, uygun bir tahmin modeli oluşturulmaktadır. Ayrıca Bölüm 1.2.'de belirtildiği gibi bilimsel literatürde zaman serilerine dayalı tahmin uygulamaları için çok çeşitli yöntemler önerilmektedir. Tüketici seviyesinde gerçekleşen tüketim örüntüleri genellikle doğrusal olmayan zaman serileri olarak değerlendirildiğinden, makine öğrenme temelli yaklaşımlar geleneksel yöntemlere kıyasla daha başarılıdır. Bu çalışmada, tüketici seviyesinde kısa dönemli yük tahmini için DL tabanlı bir model geliştirilmiştir. DL modelinin mimarisi bu bölümün alt başlıklarında detaylı olarak açıklanmıştır. Önerilen model, denetimli bir makine öğrenme algoritması olduğu için modelin uygun özellikler kullanılarak eğitilmesi gerekmektedir. Aynı zamanda, modelin eğitim sürecine dahil edilmeyen farklı bir test veri kümesinde, modelin performansı Bölüm 3.6.1.'de detaylı olarak verilen performans değerlendirme metrikleri aracılığıyla değerlendirilmektedir.

Bu tez çalışmasında, mimari yapısı aynı olan tek bir DL modeli, her bir tüketici üzerinde ayrı şekilde eğitilerek test edilmiştir. Bu yaklaşımın tercih edilmesinin ana nedenleri aşağıda kısaca açıklanmıştır:

- Bölüm 3.1.1. ve Bölüm 3.1.2.'de kullanılan veri kümelerinden kaynaklanan problemler (örneğin; her tüketicinin kullanılabilir veriye sahip olduğu zaman aralıklarının farklı olması),

- Bu çalışma kapsamında geliştirilen DL modelinin kişiye özel uygulamalar kapsamında değerlendirilmesi (örneğin; tüketiciler için geliştirilen mobil uygulamalar veya ara yüzler),
- Ağ sınırında hesaplama (Edge Computing) teknolojilerinin akıllı sayaç cihazlarıyla uyumlu yazılım altyapısı sunması. Bu sayede akıllı sayaçların belleği ve işlemcisi kullanılarak dağıtım şirketlerindeki merkezi sunucuların üzerindeki hesaplama yükü azaltılabilir (örneğin; trafo sayacının ana düğüm (master node), akıllı sayaçların ise işçi düğüm (slave/worker node) şeklinde çalıştığı bulut bilişim hizmetleri).

3.3.1. Veri Ön İşleme

Akıllı şebeke uygulamalarında tüketici seviyesinde yük tahmini problemi, karmaşık bir zaman serisi problemidir. Bu karmaşık zaman serisi problemlerini çözmek ve tahmin hatalarını azaltmak için gelişmiş veri ön işleme yöntemleriyle hibrit bir DL modeli birlikte kullanılmıştır. Veri ön işleme yöntemlerinin amacı, tüketicilerin elektrik tüketim verilerinde herhangi bir gizlilik ihlaline neden olmadan ve tüketim davranışlarına herhangi bir dış müdahale (örneğin; yük profillerinde veri yumuşatma (data smoothing) veya veri yakıştırma (data imputation) işlemleri, vs.) uygulamadan veri temizleme (data cleansing) işlemi gerçekleştirmektir. Bu çalışmada kullanılan her bir örnek, veri kümesinde yer alan farklı bir günlük yük profilidir. Bölüm 3.1.1. ve Bölüm 3.1.2.'de detaylı olarak açıklanan işlemler gerçekleştirilerek sonra ise veri kümeleri yeniden boyutlandırılmıştır. Ardından, boyut indirgeme (dimension reduction) ve aykırı değer tespiti (outlier detection), veri ön işlemenin ikinci aşamasında gerçekleştirilmiştir. Bu aşamada kullanılan algortimalar herhangi bir ön koşul gerektirmeyen entegre ve tam otomatik bir şekilde çalışmaktadır. Veri ön işlemenin son aşamasında ise 1-B CNN kullanılarak yük profillerinden tahmin aşamasında kullanılacak öznitelikler elde edilmiştir.

3.3.2. Metrik Olmayan Çok Boyutlu Ölçeklendirme Algoritması

Çok boyutlu ölçekleme (Multidimensional Scaling-MDS), yüksek boyutlu bir uzayda temsil edilen örnek çiftleri arasındaki benzerlik (veya farklılık) ölçüsünü düşük boyutlu uzaylarda temsil eden bir yöntemdir. MDS yönteminde herhangi bir olasılıksal dağılım dikkate alınmadan belirli uzaklık ölçüm fonksiyonları kullanılarak düşük boyutlarda gösterim uzaklıkları elde edilmektedir. Burada asıl amaç, tıpkı PCA ya da t-Distributed Stochastic Neighbor Embedding (t-SNE) yöntemleri gibi yüksek boyutlu veri kümelerinin düşük boyutlu ve kolay yorumlanabilir temsillerini elde etmek ve temelde bir boyut indirgeme işlemi yapmaktır.

MDS yönteminde örnekler arasında benzerlikler (similarities) veya farklılıklar (disparities) bir uzaklık matrisi oluşturularak belirlenmektedir. Bu uzaklık matrisinin daha düşük boyutlu bir uzayda grafiksel olarak gösterililebilmesi için en az hata ile gösterim kordinatlarının belirlenmesi gerekmektedir. Bir veri kümesinde bulunan n örnek

arasında $n(n-1)/2$ çift uzaklık değeri hesaplanır. Bu orjinal uzaklık değerleri mutlak uzaklık olarak tanımlanır ve uygun düşük boyutlu alt uzaylar edebilmek amacıyla orjinal uzaklık değerlerine yakın bir gösterim kordinatları oluşturmaya çalışılır. Orjinal uzaklık değerleri ile konfigürasyon uzaklık değerleri arasındaki uygunluğu ölçen parametre ise *stress* ölçüsü olarak tanımlanmaktadır.

MDS analizinde örneklerin birbirine benzerliğini veya benzemezliğini (farklılığı) ölçmek önemlidir. Herhangi bir veri kümesinin tüm örnek çifleri arasındaki benzemezlik ölçüsü ($\delta_{a,b}$) ve toplam örnek sayısı ise n ile ifade edildiğini varsayılım. Bu verilerin temsil edildiği konfigürasyon, s boyutlu bir alt uzayda aranır. Alt uzay gösterimindeki her nokta bir önek çiftini temsil eder. Bu örnek çiftlerinin birbiri arasındaki uzaklık ise $d_{a,b}$ şeklinde gösterilebilir. MDS analizindeki asıl amaç, uzaklıklar ile benzemezlikleri eşleyen en uygun konfigürasyonu elde etmektir.

Mertik olmayan analizde, giriş verileri ordinal (sıralı) ilişkiler içerir ancak çıkış verilerinin metrik olması veya aralıklı ölçekle ölçülmüş veri olması yeterlidir. Metrik olmayan çok boyutlu ölçekleme problemlerinin çözümü için ilk genel algoritma Shepard [144], [145] tarafından oluşturulmuş, Kruskal [146], [147] ise bu algoritma için *stress* adını verdiği bir kayıp fonksiyonu tamınlayarak fonksiyonu minimize etmeyi amaçlamıştır.

Klasik çok boyutlu ölçekleme yönteminde örnek çifleri arasındaki uzaklık ($d_{a,b}$) ile benzemezlik ölçüsü ($\delta_{a,b}$) birbirine eşittir ($d_{a,b} = \delta_{a,b} \quad a = 1, 2, 3, ..., k$). Denklem (3. 1)'de gösterildiği şekilde benzemezlik ölçüsünün metrik yapısı kaybolmuş ise metrik olmayan analiz uygulanır. Böylece benzemezlik ölçümünün ank sırası bu dönüşüm ile korunur.

$$\delta_{a,b} < \delta_{a',b'} \Rightarrow f(\delta_{a,b}) \leq f(\delta_{a',b'}) \quad 1 \leq a, b, a', b' \leq k \tag{3. 1}$$

burada f monoton bir fonksiyonu ifade etmektedir. Metrik olmayan analizde, uzaklıkların nümerik değerlerinin yanı sıra büyüklük sıraları da kullanılabilmektedir. Burada konfigürasyon uzaklıklarını ($d_{i,j}$) belirleme aşamasında kullanılan tek parametre uzaklık değerlerinin sıra sayılarıdır ($\delta_{i,j}$). Bu yaklaşım ile genel algoritmanın analitik bir çözüm bulabilmesi mümkün olmadığı için, *stress* değeri iteratif bir şekilde minimize edilir. Metrik olmayan çok boyutlu ölçekleme algortimasının işlem basamakları aşağıda verilmiştir [148], [149]:

- D uzaklık matrisinin tüm elemanlarını sıralanır (köşegendeki sıfır değerleri hariç), $d_{i_1,j_1} < d_{i_2,j_2} < \cdots < d_{i_m,j_m}, \quad m = n(n-1)/2$
- $d_{i,j}$ elemanları ile monotonik olarak ilişkilendirilen $d^*_{i,j}$ değerleri tanımlanır, Burada $d_{i,j} < d_{u,v} \rightarrow d^*_{i,j} \leq d^*_{u,v}$ ($\forall \ i < j$ ve $u < v$ için) gibi bir koşul sağlanmalıdır.
- Çok boyutlu uzayda tanımlanan gerçek şekille, indirgenmiş boyutlu uzayda (örneğin; k boyutlu bir alt uzay) elde edilen şekil arasındaki farklılığı ifade eden *stress* değeri hesaplanır. Burada $\hat{X}_{n\times k}$, $\mathbb{R}^k$ uzayında $\hat{d}_{i,j}$ uzaklık değeri ile tanımlanan bir şekil

ise, $\hat{X}$'nin *stress* fonksiyonu Denklem (3. 2)'de gösterildiği gibi hesaplanır.

$$S(\hat{X}) = \min(\sqrt{\frac{\sum_{i<k}(d^*_{i,j} - \hat{d}_{i,k})^2}{\sum_{i<k}\hat{d}^2_{i,k}}}), \tag{3. 2}$$

- Her k boyut için en düşük *stress* değerine sahip olan şekil, k'ıncı boyutttaki en iyi şekil olarak adlandırılır ve k'ıncı boyuttaki en düşük *stress* değeri, $S_k = \min(S(\hat{X}))$ eşitliği ile ifade edilir. Bu eşitlikte S_k, k'nın azalan bir fonksiyonudur.
- Son aşamada, uygun boyut sayısının belirlenebilmesi için $S_1, S_1, ..., S_k$ değerleri hesaplanır ve bu işlemler en düşük *stress* değeri elde edilinceye kadar tekrarlanır.

3.3.3. DBSCAN Kümeleme Algoritması

Kümeleme, temel ve yaygın olarak kullanılan bir veri analizi yöntemidir. Bir veri kümesinde birbirine benzeyen alt kümeleri bulmak için geliştirilmiş denetimsiz bir sınıflandırma işlemi olarak değerlendirilebilir. Kümeleme yöntemleri görüntü işlemeden, örüntü tanımaya, makine öğrenmesi uygulamalarından, bilgi keşfine kadar birçok alanda uygulanmaktadır.

Density-based Spatial Clustering Applications with Noise (DBSCAN), yoğunluk tabanlı bir kümeleme algoritmasıdır. Başlangıçta, uzamsal verilerin kümelenmesi için önerilse de, daha sonraki süreçlerde bilimin farklı alanlarında uygulanmıştır. Yoğunluğa dayalı kümelemede nesneler, küme elemanlarının yoğun olduğu bölgeler dikkate alınarak birbirinden ayrılır. Geleneksel DBSCAN algortiması iki farklı giriş parametresine ihtiyaç duymaktadır. Bunların birincisi kümenin yarıçapı (Eps), ikincisi ise alt küme oluşturabilmek için gerekli minimum nesne sayısıdır (MinPts). DBSCAN algoritmasının işlem basamakları şu şekildedir [150].

- Tanım 1 (Eps: Bir nesnenin komşuluğu): D nesnelerinden oluşan bir kümede, p nesnesinin Eps komşuluğu, Eps(p) şu şekilde tanımlanır:
  $Eps(p) = \{q \in D \mid \text{mesafe}(p, q) \leq Eps\}$
- Tanım 2 (Doğrudan yoğunluk tabanlı erişim-Directly density-reachable): Bir p nesnesi, bir q nesnesine Eps ve MinPts, D nesneleri kümesinde şu iki koşulu sağlarsa doğrudan yoğunlukla erişilebilir:
  Koşul 1: $p \in Eps(q)$,
  Koşul 2: $|Eps(q)| \geq MinPts$.
- Tanım 3 (Çekirdek nesne & sınır nesnesi): Bir nesne, Tanım 2'nin 2. koşulunu karşılıyorsa çekirdek nesnedir. Sınır nesnesi ise, çekirdek nesnenin kendisi değildir, ancak başka bir çekirdek nesne üzerinden yoğunluğa erişilebilir (bkz. Tanım 4).

- Tanım 4 (Yoğunluk tabanlı erişim-Density reachable): Bir p nesnesi, ilişkili bir q nesnesine erişilebilen yoğunluktadır. Eğer Eps ve MinPts değerleri, $(p_1, p_2, ..., p_n)$ gibi bir durumda, ve $p_1 = q$'den $p_n = p$'e kadar ulaşabilen nesnelerden oluşan bir zincir oluşturabiliyorsa, $p_i + 1$ nesnesi küme yoğunluğuna p_i üzerinden doğrudan erişilebilir.

- Tanım 5 (Yoğunluk bağlantılı-Density connected): Bir o nesnesi ($o \in D$) üzerinden hem p hem de q yoğunluğa erişilebiliyorsa, bir p nesnesi bir q nesnesine yoğunlukla bağlıdır denir.

- Tanım 6 (Küme-Cluster): D, nesnelerin bir veri kümesi olduğunda, Eps ve $MinPts$ kullanılarak elde edilen C kümesi, aşağıdaki koşulları karşılayan ve boş küme olmayan D'nin bir alt kümesidir:
  Koşul 1: Her p ve q nesnesi için ($\forall\ p, q$), eğer $p \in C$ ve q, Eps ve MinPts sayesinde p üzerinden yoğunluğa ulaşılabilir ise, o zaman $q \in C$'da yoğunluğa erişebilir (Maksimumluk).
  Koşul 2: p nesnesi ($\forall\ p, q \in C$), Eps ve $MinPts$ sayesinde q'ya yoğunlukla bağlıdır.

- Tanım 7 (Aykırı Değer-Noise): $C_1, C_2, ..., C_k$, D veri kümesinin Eps ve $MinPts$ değerleri sayesinde elde edilen kümeleri olsun. Bu durumda aykırı değer, D veri kümesinden elde edilen herhangi bir C kümesine ait olmayan nesnelerin kümesidir (örneğin; aykırı deger $= \{p \in D \mid \forall\ i : p \notin C_i,\ i = 1, 2, ..., k.\}$).

3.3.4. 1-B Evrişimsel Sinir Ağları

Klasik CNN, memeli hayvanların görsel korteksinden ilham alınarak oluşturulan basit ileri beslemeli ANN modelidir. CNN modeli, ilk olarak görüntüler veya videolar gibi iki boyutlu (2-B) veriler üzerinde çalışacak şekilde tasarlanmıştır. Ayrıca CNN, günümüzde birçok görüntü ve video tanıma sisteminde kullanılan standart bir uygulama haline gelmiştir. Son yıllarda ise, 2-B CNN ağlarına alternatif olarak ise, 1-B CNN olarak adlandırılan değişik versiyonları da kullanılmaktadır.

1-B CNN modeli, klasik CNN modelinin parametrelerinden bağımsız olarak giriş katmanında her türlü boyuttaki veriyi kabul edebilir. Ayrıca, çok katmanlı algılayıcı (Multilayer Perceptron-MLP) katmanına ihtiyaç duymadan "*yalnızca CNN*" katmanında yer alan evrişim (convolution) ve alt örnekleme (sub-sampling) işlemleri kullanılarak gizli CNN katmanlarında yer alan nöronlar farklı amaçlar için genelleştirilebilir. 1-B seriler için ileri yayılım (Forward Propagation-FP), denklem (3. 3)'te şu şekilde ifade edilebilir:

$$x_t^l = b_t^l + \sum_{i=1}^{N_{l-1}} conv1D(w_{i,k}^{l-1}, s_i^{l-1}) \qquad (3.\ 3)$$

burada x_t^l zaman serisinin giriş değerini, b_t^l t'inci nöronun l'inci katmanındaki bias değerini, s_i^{l-1} ise i'inci nöronun $l-1$'inci katmanındaki çıkış değerini ifade etmektedir. Ayrıca $w_{i,k}^{l-1}$, $l-1$ katmanındaki i'nci nöron ile l'inci katmandaki k'ncı nöron arasında bulunan çekirdek

fonksiyondur. 1-B CNN'de hatanın geriye yayılımı MLP çıkış katmanından itibaren başlar. MLP'nin giriş ve çıkış katmanları sırasıyla $l = 1$ ve $l = L$ olarak tanımlandığında, çıkış katmanında elde edilen hata değeri, Denklem (3. 4)'teki gibi ifade edilebilir:

$$E = \sum_{i=1}^{N_L} \left(y_i^L - t_i \right)^2 \tag{3. 4}$$

burada E hata değerini, N_L, N'inci nöronun L'nci çıkış katmanını, y_i^L ise p gibi bir giriş vektörüne karşılık gelen çıkış değerini, t_i ise gerçek değeri ifade etmektedir. Geri yayılımda her bir nöronun ağırlığına ve bias değerine karşılık gelen en düşük hatayı bulabilmek için gradyan inişi (gradient descent) uygulanır. Her MLP katmanındaki tüm hatalar geri yayılım ile belirlendikten sonra, her bir nöronun ağırlıkları ve bias değerleri gradyan iniş yöntemi ile güncellenebilir. Örneğin; k'ıncı nöronun l'inci katmanındaki hata farkı (Δ_k^l), k'ıncı nöronun bias değerini ve bu nörona bağlı önceki katmandaki nöronların tüm ağırlıklarını güncelleyebilmek için denklem (3. 5)'deki eşitlik ile ifade edilebilir:

$$\frac{\partial E}{\partial w_{i,k}^{l-1}} = \Delta_k^l \cdot y_i^{l-1} \text{ ve } \frac{\partial E}{\partial b_k^l} = \Delta_k^l \tag{3. 5}$$

Böylece MLP giriş katmanından CNN çıkış katmanına doğru gerçekleşen normal (skaler) geri yayılım, denklem (3. 6)'daki gibi ifade edilebilir:

$$\frac{\partial E}{\partial s_k^l} = \Delta s_k^l = \sum_{i=1}^{L+1} \frac{\partial E}{\partial x_i^{l+1}} \frac{\partial x_i^{l+1}}{\partial s_k^l} = \sum_{i}^{N_{l+1}} \Delta_i^{l+1} \cdot w_{i,k}^l \tag{3. 6}$$

İlk geri yayılım işlemi $l + 1$ katmanından l katmanına doğru gerçekleştirildikten sonra, l katmanının çıkışından elde edilen hata değeri giriş katmanının delta değerine (Δ_k^l) kadar yayılabilir. Delta hatasının geri yayılım katmanları arasındaki (CNN katmanları arasındaki) ifadesi($\Delta s_k^l \overset{\sum}{\Leftarrow} \Delta_i^{l+1}$) denklem (3. 7)'deki gibi ifade edilebilir:

$$\Delta_s^k = \sum_{i=1}^{N_{l+1}} conv1Dz(\Delta_i^{l+1}, rev(w_{i,k}^l)) \tag{3. 7}$$

burada $rev(\cdot)$, diziyi tersine çevirir ve $conv1Dz(.,.)$ ise seriye $K - 1$ tane sıfır eklendikten (zero-padding) sonra uygulanan 1-B tam evrişim işlemi gerçekleştirir. Son aşamada ise, ağırlık ve bias değerleri denklem (3. 8)'deki gibi ifade edilebilir:

$$\frac{\partial E}{\partial w_{i,k}^l} = conv1D(s_k^l, \Delta_i^{l+1}) \frac{\partial E}{\partial b_k^l} = \sum_n \Delta_k^l(n) \tag{3. 8}$$

3.3.5. Özyineli Sinir Ağları

RNN modeli, zaman serilerinde ardışık zaman adımlarını nöronlar vasıtasıyla temsil edebilen ileri beslemeli sinir ağlarıdır. RNN'ler de, tıpkı ileri beslemeli ağlar gibi

geleneksel nöron diziliminde döngülere sahip olmayabilir ancak özyineli (recurrent) bir bağlantı üzerinden ardışık zaman adımlarını düğümler (edges) sayesinde birbirine bağlayarak daha uzun döngüler oluşturabilir.

RNN'nin özyineli düğümlerinde t anında, mevcut veri noktasından $x(t)$ ve ağın önceki durumundaki gizli düğümünden $h(t-1)$ girdi alınır. Her t zaman adımındaki çıkış değeri $\hat{y}^t$, t zaman adımındaki gizli düğüm değerleri $h(t)$ kullanılarak hesaplanır. RNN'lerde $t-1$ anımdaki $x(t-1)$ girişi, t adımındaki $\hat{y}^t$ değerini ve özyineli bağlantılar yoluyla üretilecek diğer çıkış değerlerini etkileyebilir. Basit bir RNN'de, ileri yönde ve her zaman adımında hesaplanan gizli düğüm ve çıkış değeri denklem (3. 9) ve (3. 10)'da verilmiştir:

$$h^{(t)} = \sigma(w^{h,x}x^{(t)} + w^{h,h}h^{t-1} + b_h) \quad (3.\ 9)$$

$$\hat{y}^t = softmax(w^{y,h}h^{(t)} + b_y) \quad (3.\ 10)$$

burada $w^{h,x}$, giriş ve gizli katman arasındaki ağırlık matrisidir ve $w^{h,h}$, ardışık zaman adımlarında gizli katman ile kendisi arasındaki özyineleme ağırlıklarının matrisidir. b_h ve b_y vektörleri ise, her bir düğümün gizli katmanının ve çıkış katmanının bias değerleridir. RNN'ler, her bir zaman adımı başına bir katmana ve ağırlıklara sahip derin ağlar olarak yorumlanabilir. RNN'ler geri yayılım kullanılarak birçok zaman adımında eğitilebilir. Ayrıca tüm RNN'ler, zaman boyunca geri yayılım (backpropagation through time) olarak adlandırılan bir yaklaşımı kullanarak eğitimlerini gerçekleştirir.

Özyineli ağlar kullanılarak uzun dönemli bağımlılıkların öğrenilmesi, uzun zamandır zor bir süreç olarak değerlendirilmektedir. Özellikle kaybolan (vanishing) veya aşırı büyüyen (exploding) gradyan sorunları, hatayı birçok zaman adımında geriye yayarken ortaya çıkmaktadır. Zaman boyunca kesilmiş geri yayılım (truncated backpropagation through time), sürekli çalışan ağlar için aşırı büyüyen gradyan problemine bir çözüm üretmektedir. LSTM ağ mimarisi ise, kaybolan gradyan problemine çözüm getirmek amacıyla, sabit birim ağırlıklı ve özyineli düğümleri kullanmaktadır [151].

Dizi öğrenimi (sequence learning) için geliştirilmiş en başarılı RNN ağ mimarileri, LSTM ve çift yönlü (bidirectional) RNN'dir. LSTM modelinde, geleneksel bir sinir ağının gizli katmanında yer alan düğümler, hesaplama birimi olarak tasarlanan bellek hücreleri (memory cell) ile değiştirilmiştir. LSTM modeline sonraki aşamada, unutma hücresi (forget cell) adı verilen ek bir ara birim dahil edilmiştir [152]. Bu bellek hücreleri sayesinde ağlar, daha önce özyineli ağların karşılaştığı zorluklar ile karşılaşmamaktadır. Çift yönlü RNN modelinde ise, dizinin herhangi bir zaman adımındaki çıktısını belirleyebilmek amacıyla dizinin hem ileri (gelecek) hem de geri (geçmiş) yönde sahip olduğu bilgileri kullanabilen bir mimari sunulmuştur.

LSTM hücresinin temel yapısı, bellek hücrelerinden ve geçit birimlerinden (gate unit) oluşur. Her bellek hücresi, kendi kendine bağlı, sabit ağırlıklı ve tekrarlayan bir düğüm içerir, bu da eğitim sırasında elde edilen en uygun ağırık katsayılarının kaybolan veya aşırı büyüyen gradyan problemlerine neden olmadan birçok zaman adımından geçmesine imkan sağlar. Bu çalışmada LSTM'in geçit birimleri, giriş kapısı i_t, unut kapısı f_t ve çıkış kapısı

o_t olarak tanımlanmış ve TM modelini kısaca formüle etmek için referans [153]'te kullanılan benzer notasyonlar tercih edilmiştir.

Bir veri kümesinde $\{x_1, x_2, x_3, \ldots, x_n\}$ gibi bir dizi, $x_n \in R^k$ $(t+n)$ zaman adımındaki k boyutlu giriş vektörünü temsil etsin. LSTM modelinin iç durumu s_t', hangi iç durum vektörünün güncellenmesi, sürdürülmesi veya silinmesi gerektiğine karar vermek için önceki gizli durum h_{t-1} ve sonraki giriş örneği x_t ile etkileşime girer. LSTM'nin bu yapısı, bellek hücre durumları arasında geçici bağlantılar kurar ve tüm süreç boyunca bu bağlantı korunur. Ayrıca, LSTM bellek hücresinde *tanh* ve *sigmoid* gibi ek aktivasyon fonksiyonları kullanılır. g_t giriş düğümünde, hem mevcut giriş örneği x_t hem de önceki gizli katmandan özyineli ağırlıklar (recurrent weights) fazladan bir *tanh* aktivasyon fonksiyonundan geçirilir. Bir LSTM hücresindeki tüm düğümlerin matematiksel ifadeleri denklem (3. 11) ile (3. 16) arasında verilmiştir:

$$g_t = \tanh\left(w_{g,x}x_t + w_{g,h}h_{t-1} + b_g\right) \quad (3.\ 11)$$

$$f_t = \sigma\left(w_{f,x}x_t + w_{f,h}h_{t-1} + b_f\right) \quad (3.\ 12)$$

$$i_t = \sigma\left(w_{i,x}x_t + w_{i,h}h_{t-1} + b_i\right) \quad (3.\ 13)$$

$$o_t = \sigma\left(w_{o,x}x_t + w_{o,h}h_{t-1} + b_o\right) \quad (3.\ 14)$$

$$s_t = g_t \odot i_t + s_{t-1} \odot f_t \quad (3.\ 15)$$

$$h_t = \tanh\left(s_t\right) \odot o_t \quad (3.\ 16)$$

burada $w_{(\cdot)}$ ve $b_{(\cdot)}$ sırasıyla ağırlık matrislerini ve bias değerlerini, $\odot$ matrislerin Hadamard çarpımını, σ (*sigmoid*) ise aktivasyon fonksiyonunu temsil etmektedir. Giriş kapısı, bellek biriminin güncelleme prosedüründen sorumluyken, çıkış kapısı ve unutma kapısı, mevcut bilginin birim içinde tutulup tutulmayacağına veya silineceğine karar verir. Giriş kapısı sıfır değerini aldığında, LSTM yapısının iç durumunda herhangi bir değişiklik olmaz ve hafıza hücresi mevcut aktivasyonu korur. Bu süreç hem giriş hem de çıkış kapıları kapalı olduğu sürece ara zaman adımları boyunca devam eder.

Çift yönlü RNN (BRNN), zaman serisinin çıkış değerlerini belirlemek için serinin hem geri (geçmiş) hem de ileri (gelecek) bilgileri kullanmaktadır [154]. Yalnızca önceki bilgilerin çıkış üzerinde bir etkiye sahip olduğu LSTM modelinden farklı olarak, bu yaklaşım, önceden tanımlanmış sabit bir giriş vektör boyutu kısıtlamaları olmaksızın eğitilebilir. BRNN mimarisinde iki farklı gizli katman düğümü bulunur ve bu katmanlar hem giriş hem de çıkış katmanlarıyla bağlantılıdır. Birinci gizli katmanın özyineli birimi, veri aktarım şemasında önceki zaman adımlarıyla bir bağlantıya sahipken, ikinci gizli katmanda ters yönde benzer bir bağlantı gerçekleştirilir. BRNN, standart bir RNN'de kullanılan aynı yöntemle eğitilebilir. Gizli katmanlar arasında herhangi bir etkileşimin olmaması nedeniyle, zaman içinde açılmış (unfolded through time) standart bir ileri beslemeli ağ da tercih edilebilir. BRNN'nin matematiksel denklemleri (3. 17) ile (3. 19)

arasında verilmiştir.

$$h_t = \sigma\left(w_{h,x}x_t + w_{h,h}h_{t-1} + b_h\right) \quad (3.\ 17)$$

$$e_t = \sigma\left(w_{e,x}x_t + w_{e,e}e_{t+1} + b_e\right) \quad (3.\ 18)$$

$$y_t = f(x)_i \times \left(w_{y,h}h_t + w_{y,e}e_t + b_y\right) \quad (3.\ 19)$$

burada h_t ve e_t, sırasıyla BRNN'nin hem ileri hem de geri yönlerdeki gizli katmanların değerlerini temsil etmektedir. LSTM ve BRNN modeli birlikte kullanılabilecek uygun yöntemlerdir. LSTM modeli gizli katmanda kullanılabilecek yeni bir birim oluştururken, BRNN modeli bu birimler arasındaki bağlantıyı kurar. Bu yaklaşım çift yönlü LSTM (BLSTM) olarak adlandırılır ve el yazısı tanıma ve fonem sınıflandırması için [155]'de kullanılmıştır.

3.4. Dağıtım Hattı Seviyesinde Kayıp-Kaçak Tespiti

Elektrik şebekesinde kayıp-kaçak kullanımı, gerek kamu hizmetleri gerekse DSO'lar üzerinde doğrudan veya dolaylı birçok olumsuz etkiye sahiptir. Kamu ve özel sektörün yıllık gelir ve kar kaybı, NTL'lerin doğrudan olumsuz etkileridir. Ayrıca, düzensiz yerinde denetimler, planlanmamış bakım-onarım faaliyetleri ve arızalı sayaçların değiştirilmesi gibi ek işlemler de ortaya çıkmaktadır.

Geleneksel kayıp-kayak tespitinde, aylık toplam tüketim değerleri ile tüketiciyi tanımlayan temel bilgiler (master informations) kullanılmaktadır. Kayıp-kaçak tespiti veri analitiği perspektifinde değerlendirildiğinde ise, tüketiciyi tanımlayan temel özellikler abonelik işleminin ilk aşamasında belirlenir ve zaman içerisinde değişiklik göstermez. Dinamik olarak değişen parametreler ise aylık toplam elektrik tüketimi ile sayaca fiziksel müdahale uygulanması durumunda akıllı sayacın sistemi bilgilendirdiği uyarı kodlarıdır.

Bu çalışmada Hadoop ve Spark çerçeveleri ile entegre bir kayıp-kaçak sınıflandırma modeli geliştirilmiştir. NTL modeli özerk (standalone) modda dört ana aşamadan oluşan bir yapıda tasarlanmıştır. Bunlar sırasıyla; yerel veri deposundan HDFS'ye veri aktarımı, dağıtık hesaplama kullanarak yük profillerinden öznitelik çıkarımı, dönüşümler-eylemler (transformations-actions) ve Spark ile entegre sınıflandırıcı modellerinin oluşturulmasıdır.

3.4.1. HCTSA Yazılım Paketi

Highly Comparative Time-Series Analysis (HCTSA) yazılım paketi, çeşitli uygulamalar için önemli istatistiksel özellikleri otomatik olarak elde edebilmek amacıyla geliştirilmiş bir özellik çıkartma uygulamasıdır. HCTSA toplamda 7764 farklı istatistiksel özellik içerir ve özellik seçimi (feature selection) ve sınıflandırma uygulamaları aşamaları ile birlikte bütünleşik bir özellik öğrenme süreci oluşturmaya katkı sağlar [156].

HCTSA yazılım paketinin büyük bir kısmı, işlemler (operations) olarak tanımlanan mevcut zaman serisi analiz yöntemlerinin (açık kaynaklı yazılım paketlerinin ve toolbox'ların da dahil olduğu) uygulanmasını içermektedir. HCTSA yazılım paketinde

7000'den fazla işlem listelense de, bu sayının bir kısmı birden fazla parametre değeri için tek bir yöntemin tekrarlandığı durumlardır (örneğin; 40 farklı zaman ötelemesi (lag) durumu için otokorelasyon fonksiyonunu hesaplamak, tek bir parametrenin bir defa değiştirmesine kıyasla 40 farklı işlem içerir). HCTSA yazılım paketinde, kavramsal olarak farklı yöntemlerin sayısı ise nispeten daha düşüktür. Pakette yaklaşık olarak 1000 farklı benzersiz istatistiksel özellik bulunmaktadır.

Bu yazılım paketinin diğer önemli bileşeni ise, büyük miktarda veri içeren zaman serilerine kapsamlı operasyonel işlemleri kolaylıkla uygulayabilmek amacıyla geliştirilmiş dağıtık hesaplama seçeneğidir. Bu özellik taşınabilir toplu iş sistemi (Portable Batch System-PBS) veya kaynak yönetimi için basit Linux yardımcı programı (Simple Linux Utility for Resource Management-SLURM) gibi programlar üzerinden Matlab kullanılarak gerçekleştirilebilmektedir. HCTSA yazılım paketi aynı zamanda, `pyopy` adlı ek bir kütüphane kullanılarak Python programlama dili ile entegre edilebilmektedir.

HCTSA yazılım paketinde hesaplama işlemleri, satırlar zaman serilerini ve sütunlar hesaplama işlemlerini temsil edecek şekilde oluşturulan bir veri matrisi olarak değerlendirilebilir. Veri matrisinde, $D_{i,j}$ matrisin her bir elemanını, x_i bir zaman serisini ve F_j ise bu zaman serisine uygulanan her bir işlemin çıktısını ifade ettiğinde, veri matrisindeki tüm operasyonlar $D_{i,j} = F_j(x_i)$ şeklinde ifade edilebilir. Veri kümesinde zaman serileri bu yapıya uygun olarak özellik vektörleri olarak temsil edilir.

3.4.2. Komşuluk Bileşen Analizi

Komşuluk bileşen analizi (Neighborhood Components Analysis-NCA), en yakın komşu (nearest neighbour) algoritmasının sınıflandırma doğruluğunu artırmak için kullanılan ve parametrik olmayan bir uzaklık öğrenme yöntemidir [157]. NCA yöntemi ile aynı zamanda, hızlı sınıflandırma ve veri görselleştirme amacıyla kullanılabilecek düşük boyutlu gösterimler de elde edilebilmektedir. NCA'da uzaklık metriğini öğrenme süreci, eğitim veri kümesinin birini dışarıda bırak (leave-one-out-LOO) sınıflandırma hatasını optimize eder ve bir düzenlileştirme (regularization) parametresi kullanarak özellik ağırlık matrisinin elde edilmesine katkı sağlar.

Bir eğitim verisi kümesinde, $F = (x_1, y_1, ..., x_i, y_i, ..., x_n, y_n)$, $x_i \in \mathbb{R}^p$ bir p boyutlu öznitelik vektörünü temsil etsin. Burada, $y_i \in 1, 2, ..., c$ gerçek sınıf etiketini, c sınıf sayısını ve n ise giriş veri kümesindeki toplam örnek sayısını temsil etsin. NCA yönteminde ağırlıklandırma (weighting) vektörü w, sınıflandırma hatasını en aza indirmeye katkıda bulunan özniteliklerin bir alt kümesidir ve burada asıl amaç en uygun ağırlık vektörlerini bulmaktır. NCA'da x_i ve x_j gibi iki farklı örnek arasındaki ağırlıklandırılmış uzaklık, D_w ise denklem (3. 20)'deki gibi ifade edilebilir:

$$D_w(x_i, x_j) = \sum_{k=1}^{p} w_k^2 |x_{ik} - x_{jk}| \qquad (3.\ 20)$$

burada w_k, k'ıncı özniteliğin ağırlığını temsil etmektedir. Geleneksel en yakın komşu yönteminde, gerçek LOO sınıflandırma doğruluğu türevlenemez (non-differentiable) bir fonksiyondur ve referans noktası olarak en yakın komşu kullanılır. NCA yönteminde bu yaklaşımın yerine olasılık dağılımına dayalı türevlenebilir bir maliyet fonksiyonunu tercih edilir. NCA yönteminde x_i'nin referans noktası olarak x_j'yı seçme olasılığı, $i \neq j$ koşulu altında denklem (3. 21)'deki gibi ifade edilir:

$$p_{ij} = \frac{k(D_w(x_i, x_j))}{\sum_{m \neq i} k(D_w(x_i, x_m))} \tag{3. 21}$$

burada $(i = j)$ gibi bir koşul varsa, $p_{ij} = 0$'dir ve $k(z) = \exp(-z/\sigma_N)$ gibi bir çekirdek fonksiyondur. Burada $sigma_N$ ise, her örneğin bir referans noktası olarak belirlenme olasılığını etkileyen çekirdek genişliğini (kernel width) ifade etmektedir. $sigma_N$ çekirdek genişliği 0'a yaklaştığında, referans noktası olarak x_i'nin en yakın komşusu seçilir. Öte yandan, $sigma_N$ çekirdek genişliği $+\infty$'a yaklaştığında ise, tüm örneklerin referans noktası olarak seçilme olasılığı aynıdır. NCA yönteminde x_i örneğinin doğru sınıflandırılmış olma olasılığı ise denklem (3. 22)'deki gibi ifade edilir:

$$p_i = \sum_j y_{ij} p_{ij} \tag{3. 22}$$

burada $y_{ij} = 1$ ise $y_i = y_j$'dir, aksi durumda $y_{ij} = 0$'dır. Yaklaşık LOO doğruluğu ise denklem (3. 23)'te verilmiştir:

$$\xi(w) = \frac{1}{n} \sum_i p_i \tag{3. 23}$$

burada $\xi(w)$, çekirdek genişliği $sigma_N$ sıfıra yaklaştığında elde edilen gerçek LOO sınıflandırma doğruluğudur. Düzenlileştirme terimiyle eşlenik nihai amaç fonksiyonu ise, öznitelik seçimi ve aşırı uyumlamayı (overfitting) azaltmak için denklem (3. 24)'teki gibi ifade edilir:

$$\xi(w) = \sum_i p_i - \lambda \sum_{k=1}^{p} w_k^2 \tag{3. 24}$$

burada λ, çapraz geçerlilik sınaması (Cross Validation-CV) yoluyla belirlenen bir düzenleme terimidir. Burada λ düzenleme parametresi yalnızca nihai çözüm vektörünü değiştirir ve dolayısıyla denklem (3. 23)'teki $1/n$ katsayısı ihmal edilebilir.

3.4.3. Lojistik Regresyon

Lojistik regresyon, verileri birbirinden ayrık (discrete) sonuçlar olarak değerlendiren bir sınıflandırma yöntemidir. Lojistik regresyon, genellikle ikili (binary) sınıflandırma problemleri için uygundur ve çıkış değeri $y \in \{0, 1\}$ gibi ikili sınıflardan

oluşur. Burada sıfır (0), negatif sınıfı ve bir (1) ise pozitif sınıfı belirtir. Sınıf etiketi keyfi olarak belirlenebilir, ancak pozitif sınıf genellikle veri kümesinde azınlık sınıfına karşılık gelir ve sınıflandırma işleminde önemli olan bir alt kümedir. Sınıflandırıcının 0 ile 1 arasındaki değerleri tahmin ettiği durumda (örneğin; $0 \leq h_\theta(x) \leq 1$ gibi), lojistik regresyon hipotezi denklem (3. 25)'teki gibi tanımlanır:

$$h_\theta(x) = g(\theta^T x), \quad g(z) = \frac{1}{1+e^{-z}} \tag{3. 25}$$

burada oluşturulan hipotez ya iki terimli (binomial) ya da bir sigmoid fonksiyondur. Hipotezin çıktısı ise, θ parametresiyle ilişkilendirilen yeni bir x giriş örneği için $y = 1$, (örneğin; $P(y = 1 \mid x; \theta)$) şeklinde belirlenen bir olasılık değeridir. Bu fonksiyon sürekli olmasına rağmen çıkış değişkeninin ayrık değerler alması gerektiğinden, hipotez eğer $h_\theta(x) \geq 0.5$ şartını sağlıyor ise $y = 1$, $h_\theta(x) < 0.5$ şartını sağlıyor ise $y = 0$ olduğu varsayılır. Ayrıca, hipotezin olasılığının eşit (örneğin; $h_\theta(x) = 0.5$) olduğu durum ise karar sınırı (decision boundary) olarak adlandırılır. Bu durumda en yüksek olasılıkla her iki sınıfı birbirinden ayıran karar sınırı, doğrusal olmayan bir fonksiyondur. Lojistik regresyon için θ_i'nin en iyi parametreleri, en büyük olabilirlik kestirimi (maximum likelihood estimation) ilkesine göre denklem (3. 26) ve (3. 27)'de tanımlanan maliyet fonksiyonunun minimize edilmesiyle elde edilir:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \text{amaç fonksiyonu}(h_\theta(x^{(i)}), y^{(i)}) \tag{3. 26}$$

$$\text{amaç fonksiyonu}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & y = 1 \\ -\log(1 - h_\theta(x)) & y = 0 \end{cases} \tag{3. 27}$$

Öğrenme algoritması ne zaman yanlış bir tahmin yaparsa, çok yüksek bir cezayla (penalty) yeniden güncellenir. LR maliyet fonksiyonu denklem (3. 28)'deki gibi yeniden düzenlendiğinde:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} \left[y^{(i)} \log(h_\theta(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right] \tag{3. 28}$$

LR'nin maliyet fonksiyonunu en aza indirmek için gradyan inişi (gradient descent) optimizasyonu kullanılır. LR, çok sınıflı (multi-class) sınıflandırma problemlerine de uygulanabilir. LR modelinde, K farklı sınıftan oluşan bir sınıflandırma probleminde, bire karşı hepsi (one vs all) sınıflandırma algoritmasının yardımıyla, herhangi bir kategori pozitif sınıfa atanır, geriye kalan sınıflar ise negatif sınıfa atanır. Bu şekilde, K farklı ikili sınıflandırma problemleri elde edilir ve LR kullanılarak sınıflandırma işlemi gerçekleştirilir. Standart bir LR sınıflandırıcısı $(h_\theta^i(x))$, daha sonra $y = i$ olma olasılığını tahmin etmek için her bir i sınıfı için eğitilebilir (örneğin; $h_\theta^i(x) = P(y = i \mid x; \theta)$ her bir $i \in \{1, 2, \cdots, K\}$). Burada her yeni giriş değeri, (x) için yapılan tahminde, $h_\theta^{(i)}(x)$ değerini maksimize edecek sınıf seçilir.

3.4.4. Naive Bayes Algoritması

NB sınıflandırıcısı, en basit Bayes sınıflandırıcısı olarak bilinir ve değişkenler arasında güçlü bağımsızlık varsayımına rağmen önemli bir olasılıksal model olarak değerlendirilmektedir.

N örnekten oluşan bir eğitim veri kümesinde (örneğin; $(x^{(i)}, y^{(i)}$ gibi), her bir $x^{(i)}$'nin d-boyutlu bir öznitelik vektörü olduğunu ve her bir $y^{(i)}$'nin ise sınıf etiketini temsil ettiğini varsayalım. Burada Y ve X gibi iki değişkenin y sınıf etiketine ve $\{x_1, x_2, ..., x_d\}$ gibi bir öznitelik vektörüne karşılık gelen iki rastgele değişken olduğunu varsayalım. Rastgele değişkendeki üst simge, $i = \{1, 2, ..., N\}$ eğitim örneklerini indekslemek için kullanılmış, alt simge ise bir vektörün her bir özniteliğine veya herhangi bir rastgele değişkene karşılık gelmektedir. Genel olarak Y, K'nın olası sınıflarından ($k \in \{1, 2, \cdots, K\}$) birine tam olarak ($C_k$) giren ayrık bir değişkendir ve $\{x_1, x_2, ..., x_d\}$ özellikleri ayrık veya sürekli olabilir.

Burada asıl amaç, olası Y değerleri için $p(Y \mid X)$ sonsal olasılığı (posterior probability) hesaplayabilecek bir sınıflandırıcı oluşturmaktır. Bayes teoremine göre bu süreç ($p(Y = C_k \mid X = x)$), denklem (3. 29)'da formulize edilmiştir [158]:

$$p(Y = C_k \mid X = x) = \frac{p(X = x \mid Y = C_k)p(Y = C_k)}{p(X = x)} \tag{3. 29}$$

burada $p(Y \mid X)$'i öğrenmenin bir yolu, $p(X \mid Y)$ ve $p(Y)$'yi tahmin etmek için eğitim verilerini kullanmaktır. Bu tahminler daha sonra herhangi bir yeni $x^{(i)}$ örneği için sonsal olasılığı, $p(Y \mid X = x^{(i)})$ belirleme aşamasında Bayes teoremi ile birlikte kullanılır.

Kesin sonucu veren Bayes sınıflandırıcılarını öğrenmek genellikle zordur. Y'nin mantıksal bir değişken (Boolean) ve X'in ise d boyutlu mantıksal özelliklerin bir vektörü olduğu göz önünde bulundurulduğunda, yaklaşık olarak 2^d parametrenin sonsal olasılığının, $p(X_1 = x_1, X_2 = x_2, ..., X_d = x_d \mid Y = C_k)$ tahmin edilmesi gerekir. Bunun nedeni ise, C_k'nın belirli herhangi bir değeri için x'in 2^d olası değeri olduğundan, toplamda $2^d - 1$ tane bağımsız parametrenin hesaplaması gerekir. Örneğin; Y için iki olası değer belirlendiğinde, toplam $2(2^d - 1)$ parametrenin tahmin edilmesi gerekir.

Bayes sınıflandırıcısını oluşturma aşamasında bu zorlu örnek karmaşıklığını çözmek için, NB sınıflandırıcısı, özellikler arasında koşullu bağımsızlık varsayımı yaparak bu karmaşıklığı azaltır. Bu sayede $\{X_1, X_2, ..., X_d\}$ örneklerinde tüm özellikler, Y verildiğinde, birbirinden koşullu olarak bağımsızdır. Bir önceki durum ile karşılaştırıldığında, bu koşullu bağımsızlık (conditional independence) varsayımı, $p(X \mid Y)$ modellemesi aşamasında tahmin edilecek parametre sayısını $2(2^d - 1)$'den sadece $2d$'ye düşürmeye yardımcı olmaktadır. Denklem (3. 29)'daki $p(X = x \mid Y = C_k)$ sonsal olasılığı bu kabule göre yeniden düzenlendiğinde ise denklem (3. 30)'daki ifade elde edilir:

$$p(X = x \mid Y = C_k) = \prod_{j=1}^{d} p(X_J = x_j \mid Y = C_k) \tag{3. 30}$$

burada olasılığın genel bir özelliği olan zincir kuralı uygulanabildiği için yukarıda bahsedilen koşullu bağımsızlık durumu her bir örnek için geçerlidir. Rasgele değişken X_j'nin değerinin diğer tüm özellik değerlerinden bağımsız olduğunu ve yalnızca Y'nin etiketine göre farklılık gösterdiği varsayımı temel olarak NB yaklaşımıdır. Burada X_j ve Y iki farklı mantıksal değişken olarak tanımlandığında, $p(X_j \mid Y = C_k)$ hesaplayabilmek için sadece $2d$ parametreye ihtiyaç duyulur. NB sınıflandırıcısının temel denklemi, denklem (3. 29) ve (3. 30) kullanılarak (3. 31)'deki gibi elde edilebilir:

$$p(Y = C_k \mid X_1, \cdots, X_d) = \frac{p(Y = C_k) \prod_j p(X_j \mid Y = C_k)}{\sum_i p(Y = y_i) \prod_j p(X_j \mid Y = y_i)} \tag{3. 31}$$

burada Y değişkeninin yalnızca en olası değeriyle ilgileniyorsak, NB sınıflandırma kuralı denklem (3. 32)'deki eşitlik kullanılarak belirlenebilir:

$$Y \leftarrow \arg\max_{C_k} \frac{p(Y = C_k) \prod_j p(X_j \mid Y = C_k)}{\sum_i p(Y = y_i) \prod_i p(X_i \mid Y = y_i)} \tag{3. 32}$$

Denklem (3. 32)'de payda kısmı C_k'ya bağlı olmadığından, yukarıdaki formülasyon denklem (3. 33)'teki gibi basitleştirilebilir:

$$Y \leftarrow \arg\max_{C_k} p(Y = C_k) \prod_j p(X_j \mid Y = C_k) \tag{3. 33}$$

3.4.5. Rastgele Orman Algoritması

RF algoritması, çok modelli bootstrap (Bagging) yaklaşımının farklı bir versiyonudur. Bu yaklaşımda, makine öğrenmesi algoritmalarının varyansını azaltmak ve kararlılığını artırmak için eğitim veri kümeleri birden fazla model kullanılarak eğitilir. Sonraki aşamada, oluşturulan model test veri kümeleri üzerinde çalıştırılır ve farklı modellerden elde edilen sonuçların ortalama değerleri hesaplanır. RF algoritması, çok modelli bootstrap'ın temel adımlarını takip eder ve temel sınıflandırıcıyı oluşturmak için karar ağacı (Decision Tree-DT) algoritmasını kullanır. RF algoritması, çok modelli bootstrap'ta kullanılan bootstrap örneklemesi (bootstrap sampling) ve çoğunluk oylamasının (majority voting) yanı sıra, RF'da kullanılan temel sınıflandırıcıların çeşitliliğini artırmak için eğitim veri kümesinin yapısına rastgele öznitelik uzay seçimini de dahil eder. RF algoritmasının genel prosedürü aşağıda açıklanmıştır [159]:

- Herhangi bir eğitim veri kümesinde ($D_R = x_i, y_i$), rastgele örnekleme yoluyla bir bootstrap veri kümesi ($D_R{}'$) oluşturulur.
- Bootstrap veri kümesi ($D_R{}'$) üzerinde *LearnDecisionTree* gibi bir fonksiyon tanımlanır ve *LearnDecisionTree*'nin parametreleri ($LearnDecisionTree(data = D_R{}', iteration = 0, ParentNode = root)$) kullanılarak bir karar ağacı oluşturulur.

LearnDecisionTree, veri kümesini, iterasyon sayısını ve üst düğüm dizinini giriş olarak alan

ve geçerli veri kümesinin bir bölümü için aynı süreci tekrarlayan özyinelemeli bir fonksiyondur.

3.4.6. Karar Ağaçları Algoritması

DT algoritması, sıradüzensel bölütleme (hierarchical partitioning) kategorisinde yer alan bir sınıflandırma yöntemidir. DT, düğüm olarak tanımlanan farklı bölümleri farklı sınıflarla ilişkilendirerek verilerin hiyerarşik bir şekilde bölümlenmesini sağlar. Veri kümesinde her düzeyde gerçekleştirilen hiyerarşik bölütleme işlemi, bir bölme kriteri (split criterion) kullanılarak oluşturulur. Bölme kriteri, bir veya birden çok öznitelik üzerinde koşul oluşturabilir. DT yönteminde genel yaklaşım, farklı düğümler üzerindeki farklı sınıflar arasındaki farklılığı en yüksek düzeye çıkarmak için eğitim verilerini özyinelemeli olarak bölmeye çalışmaktır. Bu şekilde belirli bir düğümdeki farklı sınıflar arasındaki çarpıklık (sınıflar arasında farklılık) düzeyi maksimize edildiğinde, farklı sınıflar arasındaki ayrım da maksimize edilir. Bu çarpıklığı ölçmek için Gini indeksi veya entropi gibi farklı ölçüler kullanılmaktadır. Örneğin, herhangi bir N düğümündeki k farklı sınıfa ait bölümün $(p_1, ..., p_k)$, Gini indeksi $G(N)$ denklem (3. 34)'teki gibi tanımlanır [160]:

$$G(N) = 1 - \sum_{i=1}^{k} p_i^2 \tag{3. 34}$$

burada $G(N)$'nin değeri 0 ile 1-1/k arasındadır ve bu değer ne kadar küçülürse, çarpıklık da o kadar büyük olur. Gini indeksine alternatif olarak kullanılabilecek bir ölçü de $E(N)$ şeklinde denklem (3. 35)'de tanımlanan entropi değeridir:

$$E(N) = -\sum_{i=1}^{k} p_i \log(p_i) \tag{3. 35}$$

burada entropinin değeri 0 ile $\log(k)$ arasındadır. Veri kümesindeki örnekler farklı sınıflar arasında en uygun şekilde dengelendiğinde bu değer $\log(k)$ olur. Bu durum ise, maksimum entropiye sahip en uygun senaryoya karşılık gelir. Entropi ne kadar küçükse, verilerdeki çarpıklık o kadar büyük olur. Böylece, Gini indeksi ve entropi, DT'nin herhangi bir bölütleme seviyesinde yer alan bir düğümün kalitesini farklı sınıflar arasındaki ayrım düzeyi açısından değerlendirmek için etkili bir yol sağlar.

3.4.7. Destek Vektör Makineleri

Makine öğrenmesi algoritmalarının çoğunluğunda aşırı uyumlama (overfitting) problemi ile karşılaşılmaktadır ancak maksimum sınır (margin) yaklaşımını kullanan sınıflandırıcıların aşırı uyumlama olasılığı daha düşüktür. SVM, model karmaşıklığını kontrol etmek ve örnekler arasında ayırma yüzeyini (separating surface) maksimum yapmak için destek vektör olarak adlandırılan sınır örneklerini kullanan bir sınıflandırıcıdır. SVM yönteminde veriler doğrusal olarak ayrılabilir olmadığında, hatalı

sınıflandırılan örnekler için kayıp terimiyle ilişkili bir ceza fonksiyonu (penalty function) veya doğrusal olmayan ayırma yüzeyleri oluşturmak için çekirdek hileleri (kernel tricks) kullanılır [161].

İkili sınıflandırma probleminde asıl amaç, herhangi bir eğitim veri kümesinde veriler öznitelik-etiket çiftleri şeklinde tanımlandığında (örneğin; $S = \{(x_i, y_i) \mid y_i \in \{-1, +1\}, x_i \in \mathbb{R}^n, \forall i = 1, \cdots, l\}$ gibi), bir x giriş özelliğini kullanarak y etiketini doğru bir şekilde tahmin edebilen bir $f(x)$ karar fonksiyonu bulmaktır. Bu iki sınıf, denklem (3. 36)'da, $h(x) = w^\top x + b$ şeklinde tanımlanmış doğrusal bir fonksiyon kullanılarak iki bölgeye ayrıldığında, SVM' nin ayırma yüzeyi, $H = \{x \mid h(x) = 0\}$ bu alanları pozitif ve negatif bölge olarak tanımlar [161].

$$f(x) = \begin{cases} +1 & \text{eğer} \quad w^\top x + b > 0, \\ -1 & \text{eğer} \quad w^\top x + b < 0 \end{cases} \tag{3. 36}$$

Özellik uzayında pozitif ve negatif örnekleri doğrusal şekilde ayıran sonsuz sayıda hiperdüzlem olduğundan, her iki sınıf arasında en büyük boşluğa sahip olan doğrusal fonksiyonu seçmek daha makul bir seçenektir. Ancak gerçek durumda, verilerin bu varsayımı karşılaması düşük bir olasılıktır. Bu nedenle, modelin aynı zamanda doğrusal olarak ayrılamayan veriler için de kullanılabilmesi önemlidir. Doğrusal olarak ayrılamayan durumları çözebilmek ve bu eşitsizlikleri ihlal eden durumları ölçmek için $\xi(.)$ şeklinde tanımlanmış bir kayıp fonksiyonu kullanılır. SVM yönteminde kullanılan $L1$ ve $L2$ düzenlileştirme (regularization) parametreleri denklem (3. 37) ve denklem (3. 38)'de verilmiştir:

$$\xi_{L1}(w, b; x_i, y_i) = \max(0, 1 - y_i(w^\top x_i + b)) \tag{3. 37}$$

$$\xi_{L2}(w, b; x_i, y_i) = \max(0, 1 - y_i(w^\top x_i + b))^2 \tag{3. 38}$$

burada yalnızca hiperdüzlemde tanımlanan ayrıma fonksiyonu ile destek vektörler arasındaki uzaklığı maksimum seviyeye çıkarmak dışında, her bir örnek için aynı zamanda kayıp terimlerinin de $(\xi_{L1}(w, b; x_i, y_i))$ hesaplanması gerekir. Burada hesaplanması gereken iki amaç fonksiyonu bulunmaktadır ve birden fazla hedefi aynı anda en aza indirmek daha zor olduğu için, $\|w\|^*$ eşiz norm (dual norm) ve kayıp terimleri arasında bazı ödünleşmeler (tradeoff) getirilmiştir. SVM yaklaşımda negatif olmayan bir skaler bir kayıp fonksiyonu, $\xi_i = \xi_{L1}(w, b; x_i, y_i)$ denklem (3. 39)'daki gibi ifade edilebilir:

$$\min_{w,b,\xi} \ \frac{1}{2} w^\top w + C \sum_{i=1}^{l} \xi_i \ \text{ ve } \ y_i(w^\top x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \ \ \forall i = 1, \cdots, l \tag{3. 39}$$

Çekirdek fonksiyonunun kullanıldığı ve özniteliklerin doğrusal olarak ayrılabilir altuzaylarda temsil ettiği durum için genel denklem, (3. 39)'daki eşitliğe benzerdir. Buradaki tek değişiklik, örneklerin x_i yerine $\phi(x_i)$ şeklindeki bir çekirdek fonksiyon ile temsil edilmesidir.

3.4.8. Gradient Boosting Algoritması

Gradient Boosting, gradyan tabanlı optimizasyon ve boosting yöntemi gibi iki güçlü yaklaşımı bir araya getiren bir makine öğrenme yöntemidir. Gradyan tabanlı optimizasyon, eğitim verilerini kullanarak bir modelin kayıp fonksiyonunu en aza indirmek için gradyan hesaplamalarını kullanır [162]. Boosting yöntemi ise, tahmine dayalı işlemlerde sağlam (robust) bir öğrenme sistemi oluşturmak için bir dizi zayıf (weak) modeli bir araya getirir. Bu açıdan değerlendirildiğinde boosting yöntemler, bagging ve diğer topluluk tabanlı yaklaşımlar ile benzerlik göstermektedir [163].

Çıkış değişkeninin $y \in \{-1, +1\}$ şeklinde ifade edildiği bir iki sınıflı probleminde, bir x tahmin değişkeni vektörü verildiğinde, bir $G(X)$ sınıflandırıcısı $\{-1, +1\}$ değerlerinden birini kullanarak bir tahmin üretir. Eğitim örneğindeki hata ise denklem (3. 40)'daki gibi ifade edilebilir:

$$\text{hata} = \frac{1}{N} \sum_{i=1}^{N} H\left(y_i \neq G(x_i)\right) \tag{3. 40}$$

burada H tahmin edilen etiketlerin gerçek değerler ile aynı olmadığı durumları ifade eden bir fonksiyondur. Bu aşamada zayıf bir sınıflandırıcı, hata oranı rastgele bir tahminden sadece biraz daha iyi olan sınıflandırıcıdır. Boosting yönteminin asıl amacı ise, zayıf sınıflandırma algoritmasını verilerin değiştirilmiş versiyonlarına sırayla uygulayarak, $G_m(x), m = 1, 2, 3, ..., N$ şeklinde bir dizi zayıf sınıflandırıcı üretmektir. Tüm bu zayıf sınıflandırıcılardan gelen tahminler daha sonra nihai tahmini üretmek için (weighted majority voting) ağırlıklandırılmış çoğunluk oylama kullanılarak denklem (3. 41)'deki gibi ifade edilebilir [162]:

$$G(x) = sign\left(\sum_{m=1}^{M} \alpha_m G_m(x)\right) \tag{3. 41}$$

buradaki α_m değerleri boosting algoritması ile hesaplanır ve her bir zayıf sınıflandırıcının, $G_m(x)$ tahmin sonucuna olan katkısının ağırlığını ifade eder. Her boosting adımında gerçekleşen veri değişiklikleri ise, eğitim örneklerinin her birine $(x_i, y_i), i = 1, 2, ..., N$ atanan ağırlıkların $(w_1, w_2, w_3, ..., w_N)$ değiştirilmesiyle elde edilir. Eğitimin başlangıcında tüm ağırlıklar $w_i = 1/N$ olarak ayarlanır ve ilk adımda zayıf sınıflandırıcı veri üzerinde olağan şekilde eğitilir. Her ardışık yineleme $(m = 2, 3, ..., M)$ için gözlem ağırlıkları ayrı ayrı değiştirilir ve sınıflandırma algoritması ağırlıklı gözlemlere yeniden uygulanır. Örneğin; m'inci adımında sınıflandırıcı tarafından yanlış sınıflandırılan örnekler için bir önceki adımda oluşturulan zayıf sınıflandırıcının $(G_{m-1}(x))$ ağırlıkları artarken, doğru sınıflandırılanlar için ağırlıklar azaltılır. Böylece yinelemeler ilerledikçe, doğru bir şekilde sınıflandırılması daha zor olan örneklerin eğitimde giderek artan bir etkisi oluşur ve her bir ardışık sınıflandırıcı, dizideki önceki zayıf sınıflandırıcılar tarafından yanlış sınıflandırılan eğitim örneklerine daha çok dikkat etmeye zorlanırlar.

3.4.9. XGBoost Algoritması

XGBoost algoritması, genelleştirilmiş gradyan boosting algoritmasının bir uygulamasıdır ve daha doğru modeller elde etmek için boosting ve DT yaklaşımını birlikte kullanan denetimli bir öğrenme algoritmasıdır [164]. XGBoost, temel olarak DT yapısının boosting yaklaşım ile birlikte kullanıldığı farklı bir algoritmasıdır. Bu yaklaşım genellikle tree boosting olarak adlandırılmakta ve zayıf sınıflandırıcılardan oluşan diziye eklenen her yeni model temelde bir DT'dir. DT algoritmaları, bölütlemenin etkinliğini ölçen bazı kriterleri en üst düzeye çıkarmak için tipik olarak düğümleri kökten fırsatçı (greedy) algoritma yaklaşımından faydalanarak genişletir. DT yapısını oluştururken dikkat edilmesi gereken bir diğer husus, aşırı uyumlamayı önlemek için bir tür düzenlileştirme yaklaşımı uygulamaktır [165].

XGBoost, aşırı uyumla problemine çözüm üretmek amacıyla bir düzenlileştirme terimini ve ayrıca keyfi türevlenebilir kayıp fonksiyonunu içeren genelleştirilmiş bir gradyan boosting uygulamasıdır. XGBoost algoritmasının amaç fonksiyonu, bir kayıp fonksiyonu ve modelin karmaşıklığını düzenleyen bir düzenleme terimi içeren iki parçalı bir yapıda denklem (3. 42)'deki gibi tanımlanabilir [164]:

$$\text{Amaç Fonksiyonu} = \sum_i L(y_i, \hat{y}_i) + \sum_k \Omega(f_k) \tag{3. 42}$$

burada $L(y_i, \hat{y}_i)$ belirli bir eğitim örneği için tahmin ve gerçek sınıf etiketleri arasındaki farkı ölçen herhangi bir dışbükey (convex) türevlenebilir kayıp fonksiyonunu ve $\Omega(f_k)$ ise f_k ağacının karmaşıklığını ifade eder. Modelin karmaşıklığını temsil eden $\Omega(f_k)$ terimi denklem (3. 43)'te verilmiştir:

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda w^2 \tag{3. 43}$$

burada T parametresi, f_k ağacında yer alan uç düğüm (leaf node) sayısını, w parametresi ise uç düğümlerin ağırlıklarını ifade etmektedir. γT her yeni uç düğüm için sabit bir ceza sağlar ve λw ise sınıflandırıcı dizisinde yüksek ağırlık katsayısına sahip zayıf sınıflandırıcıları cezalandırır. γ ve $\lambda\gamma$ değişkenleri kullanıcı tarafından yapılandırılabilen parametrelerdir. XGBoost algoritmasında boosting yaklaşımının yinelemeli bir şekilde tekrarlandığı durumda, m'inci yineleme için amaç fonksiyonunu, en yeni karar ağacı f_k için önceki yinelemenin tahmini cinsinden ($\hat{y}_i^{(m-1)}$) denklem (3. 44)'teki gibi ifade edilebilir:

$$\text{Amac Fonksiyonu}^m = \sum_i L(y_i, \hat{y}_i^{(m-1)} + f_k(x_i)) + \sum_k \Omega(f_k) \tag{3. 44}$$

burada amaç fonksiyonunu en aza indiren f_k zayıf sınıflandırıcısını bulmak için optimizasyon işlemi gerçekleştirilir.

3.4.10. CatBoost Algoritması

CatBoost algoritması, temel sınıflandırıcı olarak ikili DT yapısını kullanan bir gradyan boosting uygulamasıdır. Gradyan boosting yaklaşımını kullanan uygulamalarda nümerik değerlerden oluşan özellikler için DT'leri kullanmak makul bir yaklaşımdır ancak gerçek hayatta birçok veri kümesi kategorik özellikler de içermektedir [166]. Gradyan boosting yaklaşımının kullanıldığı uygulamalarda kategorik özellikler eğitim aşamasından önce nümerik değerlere dönüştürülmektir. CatBoost algoritması ise, kategorik özelliklerle başa çıkmak için çeşitli yaklaşımlar uygular.

CatBoost, özelliklerin birbirleri arasındaki kombinasyonlarını oluşturma aşamasında fırsatçı algoritma yaklaşımına benzer bir strateji kullanır. CatBoost algoritması, ağacın her bir bölütlenmesi aşamasında mevcut ağaçtaki önceki bölütlemeler için veri kümesindeki halihazırda kullanılan tüm kategorik özellikleri (ve bunların kombinasyonlarını) birleştirir. CatBoost'ta tanıtılan diğer bir algoritmik gelişme ise, permütasyona dayalı alternatif bir sıralı (ordered) boosting yaklaşımının uygulanmasıdır.

3.4.11. LightGBM Algoritması

LightGBM, karar ağacı tabanlı öğrenme algoritmalarını kullanan bir gradyan boosting çerçevesidir. LightGBM algoritması temel olarak gradyan tabanlı tek yönlü örnekleme (Gradient-based One-Side Sampling-GOSS) ve özel değişken paketi (Exclusive Feature Bundling–EFB) olarak adlandırılan iki farklı yaklaşımı birlikte kullanmaktadır [167].

GOSS yaklaşımı: Gradyan boosting tabanlı karar ağaçlarında veri örneği için yerel ağırlık bulunmamakla birlikte, farklı gradyanlara sahip veri örnekleri bilgi kazancının (information gain) hesaplanmasında farklı etkilere sahiptir. Özellikle, daha büyük gradyanlara sahip örnekler (örneğin; az eğitilmiş örnekler) bilgi kazanımına daha fazla katkıda bulunmaktadır. Bu nedenle, LightGBM algoritmasında veri örneklerine örnek seyreltme (down-sampling) işlemi yapılırken, bilgi kazancı tahmininin doğruluğunu korumak için büyük gradyanlara sahip olan örnekler tutulur ve yalnızca küçük gradyanlara sahip örnekler rastgele olacak şekilde bırakılır. Bu işlemin veri dağılımı üzerindeki etkisini telafi etmek için ise, bilgi kazancı hesaplanırken küçük gradyanlara sahip veri örnekleri için sabit bir çarpan kullanılır. GOSS yönteminde veri örnekleri ilk aşamada gradyanlarının mutlak değerine göre sıralanır ve en üstteki a×%100 kadar örnek seçilir. Daha sonra ise, verilerin geri kalanından rastgele olacak şekilde b×%100 kadar örnekleme yapılır. Sonraki aşamada ise, bilgi kazancı hesaplanırken küçük gradyanlara sahip örneklenen verilerin sonuca olan etkileri sabit bir katsayı ($\frac{1-a}{b}$) ile artırılır. Bu sayede, eğitim aşamasında orijinal veri dağılımını fazla değiştirmeden az eğitilmiş örneklere daha fazla odaklanılır.

EFB yaklaşımı: Yüksek boyutlu veri kümeleri genellikle seyrek (sparse) özellik gösterir ve öznitelik uzayının seyrekliği (sparsity) öznitelik sayısını azaltmak için neredeyse kayıpsız bir yaklaşım tasarlama olanağı sağlayabilir. Ayrıca seyrek bir özellik uzayında, birçok özellik karşılıklı olarak özel bir örüntü taşıyabilir ve öznitelikler güvenli bir şekilde

tek bir özellikte bir araya getirilebilir. Bu öznitelik birleştirme işlemi LightGBM algoritmasında özel değişken paketi olarak adlandırılmaktadır.

3.5. Önerilen Yöntemler

3.5.1. Kısa Dönemli Yük Tahmin Yöntemi

Tüketicilerin enerji verilerini ve yük profillerini kullanarak akıllı şebekede tüketici seviyesinde yük tahmini yapmak karmaşık bir zaman serisi problemidir. Bu tez çalışmasında, karmaşık tüketim örüntülerini ortaya çıkarmak ve tahmin hatalarını azaltmak için gelişmiş veri ön işleme yöntemleriyle entegre hibrit bir DL modeli geliştirilmiştir. Bu nedenle geliştirilen büyük veri çerçevesi, gelişmiş veri ön işleme ve hibrit DL modeli olmak üzere iki ana aşamadan oluşmaktadır. Bu tez çalışması kapsamında geliştirilen kısa dönemli yük tahmin modelinin algoritma şeması Şekil 3.15.'te verilmiştir.

Veri ön işleme yöntemlerinin kullanılmasındaki asıl amaç, tüketici verisinde herhangi bir düzenleme gerçekleştirmeksizin ver kümelerini DL modelinin eğitim aşamasına hazır hale getirmektir. Bu yüzden, veri kümelerini yeniden düzenleme, boyutsallık azaltma ve aykırı değer tespiti gibi yöntemler veri ön işleme aşamasında gerçekleştirilmiş ve geliştirilen büyük veri çerçevesi herhangi bir ön kabul gerektirmeyen entegre şekilde tasarlanmıştır.

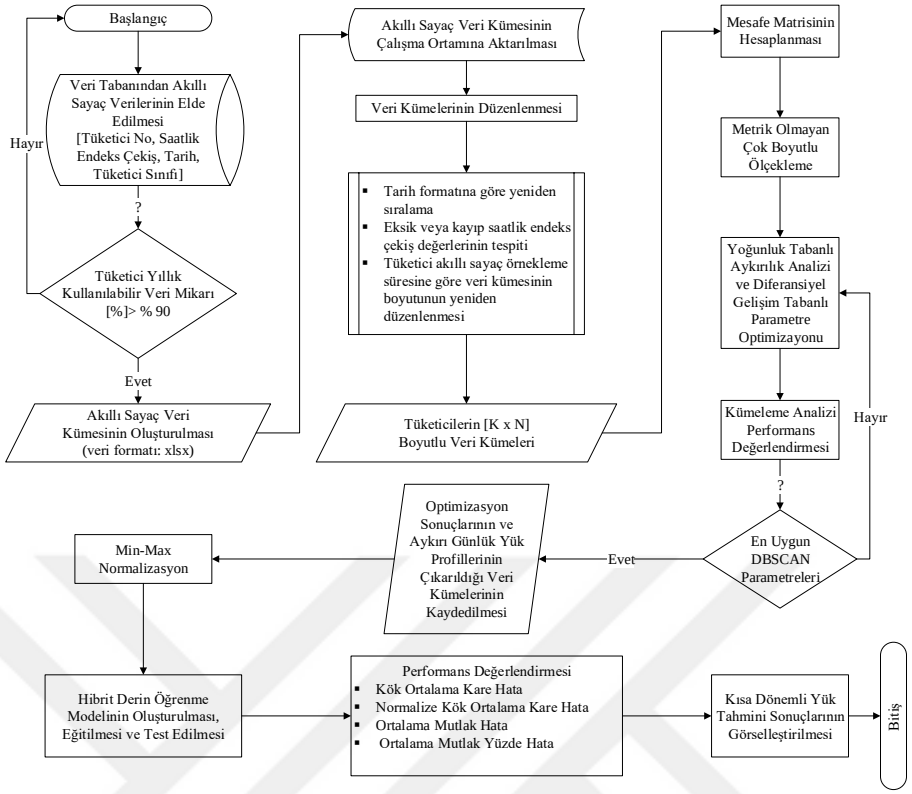
Akıllı sayaç veri kümelerindeki her bir tüketicinin mesafe matrisinin $(X^{(i)}_{[K,N]})$ şeklinde tanımlandığını varsayalım. Burada i, veri kümesindeki i'inci tüketiciyi $(i = 1, 2, 3, \cdots, m)$, K kullanılabilir günlük yük profili sayısını ve N ise yük profilindeki ölçüm sayısını $(N \in \{24, 48, 96\})$ ifade etmektedir. İki farklı yük profilinin birbiri arasındaki Öklid uzaklığı ve veri kümesinin uzaklık matrisi denklem (3. 45) ve denklem (3. 46) kullanılarak hesaplanabilir.

$$d_{a,b} = \sqrt{(d_a - d_b)(d_a - d_b)'} \tag{3. 45}$$

$$D^{(i)}_{[K,K]} = \begin{bmatrix} d_{a,a} & d_{a,b} & \cdots & d_{a,k} \\ d_{b,a} & d_{b,b} & \cdots & d_{b,k} \\ \vdots & \vdots & \ddots & \vdots \\ d_{k,a} & d_{k,b} & \cdots & d_{k,k} \end{bmatrix} \tag{3. 46}$$

burada $d_{a,b}$, iki farklı yük profili arasındaki uzaklığı, $D^{(i)}_{[K,K]}$ ise i'inci tüketiciye ait uzaklık matrisini ifade etmektedir.

Günlük yük profilleri, akıllı sayaçların örnekleme süresine bağlı olarak saatlik okumalardan oluşmaktadır. Yanlış akıllı sayaç okumaları gibi nokta aykırı değerlerden farklı olarak, günlük yük profillerinin maksimum ve minimum değerleri pik saatler arasında sürekli değişkenlik gösterir ve aykırı değer analizi aşamasında saatlik okumalardan ziyade 24 saatlik döngüleri dikkate almak daha makul bir seçenektir.



Şekil 3.15. Kısa dönemli yük tahmini için geliştirilen hibrit DL modelinin genel yapısı

Bununla birlikte, kümeleme tabanlı aykırı değer tespit algoritmaları, uzaysal koordinatlar üzerinde daha etkili çözümler üretir ve zamana bağlı temsiller yerine günlük yük profillerinin uzaysal temsili, algoritmaların işlem sürelerinin azaltılmasına katkıda bulunabilir. Bu çalışmada, yük profillerinden 2-B altuzay izdüşümleri ($X_{[K,2]}$), metrik olmayan çok boyutlu ölçekleme algoritması kullanılarak elde edilmiştir. Her tüketici için oluşturulan uzaklık matrislerine Kruskal'ın metrik olmayan *stress* kriteri [147] uygulandıktan sonra, elde edilen 2-B matrisler yoğunluk tabanlı kümeleme işleminde kullanılmıştır.

DBSCAN algoritmasının yoğunluk tabanlı benzer kümele algoritmalarına kıyasla bazı avantajları bulunmaktadır. Bu avantajlardan sırasıyla; önceden belirlenmesi gereken bir küme numarasına ihtiyaç duymaması ve veri kümesinde aykırı değerler içermesidir. Ayrıca, tüketicilerin tekrarlayan tüketim davranışları, veri kümelerinde belirgin örüntüler oluşturarak, sıradışı tüketim davranışlarını tespit etmek için önemli bilgiler sağlamaktadır. DBSCAN algoritmasında önsel (a priori) olarak tanımlanması gereken iki parametresi bulunmaktadır. Bu parametreler sırasıyla; bir kümedeki minimum nesne sayısı (*minpts*) ve nesneler arasındaki uzaklıktır (*epsilon*). DBSCAN algoritmasının kümeleme sonuçları farklı *minpts* ve *epsilon* değerlerine göre değişiklik gösterir. Bu çalışmada bu nedenle, günlük yük profillerindeki düzensizliği değerlendirmek ve aykırılık analizi için,

DBSCAN'ın ikili diferansiyel gelişim (Binary Differential Evolution-BDE) algoritması ile birleştirilmiş farklı bir versiyonunu kullanılmıştır [168].

BDE-DBSCAN algoritması, (*minpts*) parametresinin optimizasyon aşamasında bir bit dizisi şeklinde tanımlanmış popülasyonlar üretmektedir. DBSCAN algoritmasının ikinci parametresi olan *epsilon* ise, optimizasyon sürecinde hem analitik hem de yinelemeli bir şekilde yaklaşımla belirlenmektedir. DBSCAN algoritmasının parametre optimizasyonu aşamasında dahili kümeleme doğrulama (internal clustering validation) metriği olarak basitliği ve kolay yorumlanması nedeniyle kök ortalama karesel standart sapma (Root Mean Square Standard Deviation-RMSSTD) indeksi tercih edilmiştir. RMSSTD kümeleme performans indeksi denklem (3. 47)'de verilmiştir [169]:

$$\text{RMSSTD} = \sqrt{\frac{\sum_i \sum_{x \in C_i} \|x - c_i\|^2}{N \sum_i (n_i - 1)}} \tag{3. 47}$$

burada x günlük yük profilini ($\|x_i\| = \sqrt{x_i^T \cdot x_i}$), C_i ve c_i ise sırasıyla i'inci kümeyi ve bu kümenin merkezini temsil etmektedir. Burada N, yük profillerinin zaman çözünürlüğü (örneğin; 24, 48, 96) ve n_i ise C_i kümesinde yer alan toplam örnek sayısıdır. Bu tez çalışması kapsamında geliştirilen yoğunluk tabanlı aykırılık analizi algoritmasının sözde kodu (pseudo-code) Algoritma 1'de verilmiştir. Algoritma 1'de kullanılan maliyet

Algorithm 1 Önerilen BDE-DBSCAN tabanlı aykırılık analizi modelinin sahte kodu [168]

Giriş: Veri kümesi1 = $X_{[K,2]}$, Veri kümesi2 = $X_{[K,N]}$
Çıkış: EnİyiÇözüm$_{[\text{epsilon,minpts,kümesayısı,RMSSTD indeks}]}$, sınıf etiketi
Başlangıç Şartı : Referans [168]'de tanımlanan başlangıç parametrelerini kullan.
Değişiklik : Referans [168]'de kullanılan saflık (purity) kümeleme performans indeksi yerine kök ortalama karesel standart sapma (Root Mean Square Standard Deviation-RMSSTD) indeksini kullan.
1: maksimum epok = 30
2: maksimum iterasyon = 100
3: **for** $i = 1$ to maksimum epok **do**
4: **for** $ii = 1$ to maksimum iterasyon **do**
5: Veri kümesi1 ($X_{[K,2]}$) için BDE-DBSCAN algoritmasını çalıştır.
6: Veri kümesi2 ($X_{[K,N]}$) için RMSSTD indeksini denklem (3. 47)'de gibi hesapla.
7: **if** Popülasyon$_{[i,ii]}$.Maliyet < Popülasyon.Maliyet **then**
8: EnİyiÇözüm = Popülasyon$_{[i,ii]}$
9: **else if** RMSSTDindex$_{[i,ii]}$ = RMSSTDindex **and** kümesayısı$[i, ii]$ < kümesayısı **then**
10: EnİyiÇözüm = Popülasyon$_{[i,ii]}$
11: **end if**
12: **end for**
13: En iyi çözümün sonuçlarını (EnİyiÇözüm$_{[i,ii]}$) *sonuclar* adında yeni bir değişkende kaydet.
14: **end for**
15: **return** *sonuclar*

fonksiyonunun sonuçlarını azaltmak için RMSSTD indeksi hem 2-B hem de N-boyutlu veri kümesinde kullanmıştır. Bu uyarlama, algoritma 1'in 5. ve 6. satırlarında verilmiştir.

RMSSTD kümeleme indeksi, optimizasyon sürecinde elde edilen kümelerin homojenliğini değerlendirir [170]. Kümeleme işlemi öncesinde i'nci tüketicinin yeniden boyutlandırılmış veri kümesi, $X_{[K,N]}^{(i)} = \left\{x_{(1,N)}, x_{(2,N)}, x_{(3,N)}, \ldots, x_{(K,N)}\right\}$ n boyutlu uzayda $[KXN]$'lik bir matristir. Burada, K ve N sırasıyla, i'nci tüketicinin kullanılabilir

günlük yük profillerinin sayısını ve sayacın zaman çözünürlüğünü temsil etmektedir. Küme etiketi ataması işleminden sonra, veri kümesi $X_{[K,N]}$, benzersiz küme numarasına göre alt kümelere bölünür ($C = \{C_1, C_2, ..., C_i\}$) ve ardından her farklı alt küme için ortalama günlük yük profili, standart sapma ve küme içi kareler toplamı hesaplanır. Son aşamada ise, her bir alt küme için ortalama karesel toplam denklem (3. 47)'deki gibi hesaplanarak karesel ortalamanın karekökü (Root Mean Squared-RMS) değeri elde edilir. Bu tez çalışmasında kullanılan DBSCAN algoritmasının parametre optimizasyonu aşamasında, asıl amaç minimum RMSSTD indeks değerlerini bulmaktır. Bu nedenle geliştilen aykırı değer tespit modeli, parametre oprimizasyonu sonuçlarının doğruluğunu test etmek için 30 defa çalıştırılmıştır. Ayrıca, RMSSTD indeksinin monotonluk sorunlarının üstesinden gelmek için iki farklı strateji uygulanmıştır [171]. Bu stratejilerin ilki, kümeleme işlemi için 2-B alt uzayı kullanmak ve RMSSTD doğrulama indeksini hesaplamak için $[K \times N]$ boyutlu veri kümesini kullanmaktır. İkinci strateji ise, her biri 100 iterasyondan oluşan epok için elde edilen minimum RMSSTD değerini son yinelemeye kadar tutmaktır. Bu koşul sağlandığında, DBSCAN algoritmasının optimal parametreleri ve tahmin modelinin optimal eğitim veri kümesi elde edilir.

Yoğunluk tabanlı aykırılık analizi ve parametre optimizasyonu işlemi sonrasında, tüketicilerin veri kümeleri denetimli (supervised) öğrenme için eğitim (training), doğrulama (validation) ve test aşamaları için farklı alt veri kümelere ayrılmıştır. Ardından, 1-B CNN-BLSTM hibrit DL modeli kullanılarak tüketicilerin kısa dönemli yük tahmini yapılmaktadır. Hibrit DL modelinin 1-B evrişim katmanları, "*konvolüsyon*" operatörünün çalışma prensibine göre tasarlanmıştır ve bu katmanın temel amacı, elektrik tüketim zaman serisinde yer alan örnekler arasındaki korelasyonu koruyarak giriş verilerinden faydalı özellikler elde etmektir. Öznitelik matrisi, bir filtrenin 1-B enerji zaman serisinin üzerinde kaydırılmasıyla oluşturulur ve ardından öznitelik matrisindeki tüm negatif değerlerin sıfır ile değiştirilmesi için ek bir aktivasyon fonksiyonu uygulanır.

Veri kümelerinde normalizasyon işlemi, çok düşük veya çok yüksek değerlerden oluşan özelliklerin birlikte bulunuğu durumların modellerin öğrenme işlemini olumsuz şekilde etkilemesini engellemek amacıyla tüm özelliklerin giriş değerlerinin belirli bir aralıkta sınırlandırılmasına yardımcı olur. Tez kapsamında önerilen hibrit DL modelinin eğitim, doğrulama ve test aşamalarında tüketicilerin veri kümeleri min-max normalizasyon işlemi kullanılarak normalize edilmiştir. Min-max normalizasyon işlemi denklem (3. 48)'de verilmiştir:

$$x'_i = \frac{x_i - x_{minimum}}{x_{maksimum} - x_{minimum}} \tag{3. 48}$$

burada x_i giriş veri kümesinin orijinal değerlerini, x'_i ise $[0, 1]$ aralığında normalize edilmiş değerleri, $x_{maksimum}$ ve $x_{minimum}$ ise sırasıyla maksimum ve minimum tüketim değerini ifade etmektedir.

Denetimli makine öğrenme yaklaşımını kullanan modeller, eğitim aşamasında denklem

(3. 49)'da ifade edildiği şekliyle giriş ve çıkış örnek çiftlerini kullanır:

$$\mathbf{y}' = H_{SL}(\mathbf{x}') \tag{3. 49}$$

burada $\mathbf{y}'$ enerji zaman serisinin normalize edilmiş tahmin değerlerini ve H_{SL} ise hibrit DL modelini ifade etmektedir. Normalize edilmiş çıkış vektörü ise denklem (3. 50)'deki gibi ifade edilebilir:

$$\mathbf{y}' = \{y'_m, y'_{m+1}, \ldots, y'_{m+n-1}\} \tag{3. 50}$$

burada y'_{i_n}, i_n'inci zaman adımındaki ($i_n \in \{m+1, m+2, ..., m+n\}$) tahmini yük değerini, n ise $\mathbf{y}'$ veri kümesinde gelecek dönem için tahmin edilecek örneklerin sayısıdır.

Bu tez çalışmasında önerilen hibrit DL modelinin performansını uygun şekilde değerlendirmek için, test veri kümesi eğitim ve doğrulama aşaması tamamlanmış modelde kullanılan veri kümelerinden bağımsız olacak şekilde oluşturulmuştur. Bu nedenle, performans değerlendirmesinin tutarlılığını ve kesinliğini korumak için normalize edilmiş denetimli veri kümesi üç alt veri kümesine (eğitim kümesi %70, doğrulama kümesi %20 ve test kümesi %10) bölünmüştür.

Önerilen hibrit DL modeli, geleneksel 1-B CNN ve BLSTM modellerinin hibritleştirilmiş farklı bir versiyonudur. Hibrit DL modelinin ilk kısmı, giriş verisinden özellik çıkarmak ve kodlamak için kullanılan 1-B CNN katmanı, ikinci yarısı ise, elde edilen özellikleri analiz etmek için kullanılan BLSTM katmanıdır.

BLSTM modeli, ardışık (sequential) giriş ve çıkış verisinde önemli örüntüleri çıkarmak ve muhafaza etmek için ileri (forward) ve geri (backward) yönde eğitim gerçekleştirir. Bu sayede, önerilen hibrit DL modelinin eğitiminde, 1-B CNN kullanılarak elde edilen özellikler BLSTM modelinde giriş vektörü olarak kullanılmıştır. Hibrit DL modeli iki adet CNN katmanından (filtre boyutu = 64, çekirdek boyutu = 3) ve iki adet BLSTM katmanından (gizli katman1 nöron sayısı = 50, gizli katman2 nöron sayısı = 25) oluşmaktadır. Önerilen modelde kullanılan aktivasyon fonksiyonu $ReLU$ olarak belirlenmiş ve eğitim aşamasındaki toplam epok sayısı 100'dür. DL modelinin gizli katman sayısı, nöron sayısı, aktivasyon fonksiyonu, kayıp fonksiyonu ve epok sayısı gibi hiperparametreleri, ince ayarlama (fine tuning) ve karmaşık olasılıksal süreçler için bir nondeterministic polynomial (NP) optimizasyon problemi oluşturmaktadır. Bu nedenle, hibrit DL modelin hiperparametreleri literatür dikkate alınarak deneme-yanılma (trial and error) yaklaşımı kullanılarak belirlenmiştir. Bu tez çalışması kapsamında önerilen hibrit DL modelinin hiperparametreleri Tablo 3.2.'de verilmiştir.

Önerilen kısa dönemli yük tahmin modelinin veri ön işleme yöntemleri MATLAB, hibrit DL modeli ise Python 2.7 versiyonu ile uyumlu scikit-learn çerçevesi ve Keras DL çerçevesi kullanılarak geliştirilmiştir.

Tablo 3.2. Önerilen hibrit DL tahmin modelin hiperparametre değerleri

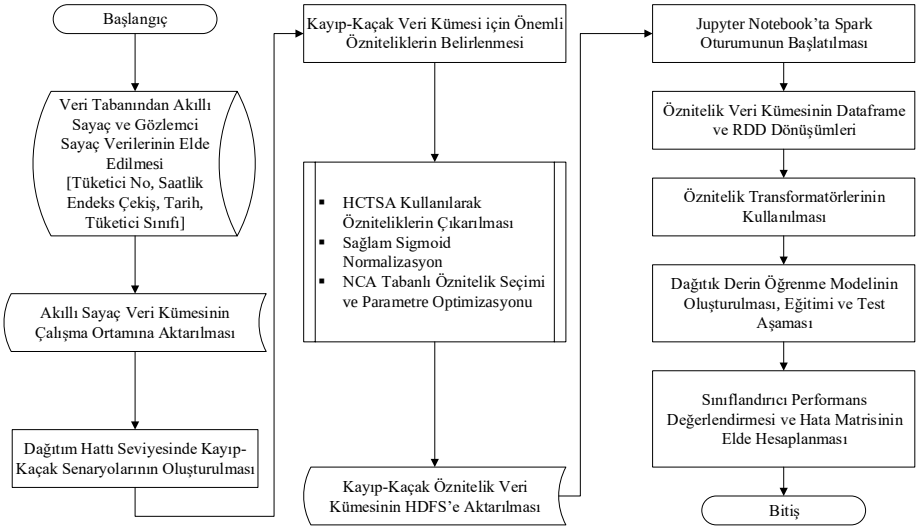
Hiperparametre	**Değer**
1-B CNN filtre boyutu	64
1-B CNN çekirdek boyutu	3
BLSTM 1. gizli katman nöron sayısı	50
BLSTM 2. gizli katman nöron sayısı	25
Aktivasyon fonksiyonu	ReLU
Kayıp fonksiyonu	Ortalama Karesel Hata (Mean Square Errror-MSE)
Öğrenme Oranı	0.001
Veri grup büyüklüğü (batch size)	120
Epok sayısı	100

3.5.2. Kayıp-Kaçak Tespiti Yöntemi

Bu tez çalışmasında geliştirilen kayıp-kaçak tespit modeli, makine öğrenmesi, DL yöntemlerini Büyük Veri çerçevesiyle birlikte kullanan bir yapıda tasarlanmıştır. Ayrıca öznitelik öğrenme sürecinde, dağıtım seviyesinde FDI tabanlı kayıp-kaçak senaryolarını tespit etmek için güçlü (robust) istatistiksel özellikler araştırılmıştır. Önerilen kayıp-kaçak tespit yöntemi Hadoop ve Spark çerçeveleri ile entegre yerel sunucuda çalışan özerk bir uygulamadır ve dört ana aşamadan oluşmaktadır. Bunlar sırasıyla; dağıtık hesaplama kullanarak öznitelik çıkarma, yerel sunucudan HDFS'ye veri aktarımı, dönüşümler-eylemler ve geleneksel ve en ileri (state of the art) makine öğrenmesi modelleri kullanarak sınıflandırma işlemidir. Bu tez çalışması kapsamında geliştirilen kayıp-kaçak sınıflandırıcı modelinin genel yapısı Şekil 3.16.'da verilmiştir.

Bu çalışmada, HDP 2.6.5 yazılım paketi kullanılarak Oracle VM VirtualBox üzerine Hadoop ve Spark bileşenleri kurulmuştur. HDP 2.6.5 paketi, varsayılan ayarlarla sanal bir Linux/Red Hat (64-bit) işletim sistemi üzerinde çalışmaktadır. Hadoop'u özerk bir uygulama olarak çalıştıran sunucu, AMD Ryzen 9 5900X 12 çekirdekli işlemciye, 32 GB DDR4 3000 MHz CORSAIR VENGEANCE LPX C16 RAM'e, Samsung SSD 970 EVO Plus 500 GB sabit diske, NVIDIA GeForce RTX 2070 SUPER grafik kartına ve MSI B450 TOMAHAWK MAX AM4 DDR4 anakartına sahiptir.

HDP'nin 2.6.5 sürümü varsayılan geliştirme ortamı olarak Apache Zeppelin notebook'u içermesine rağmen, kullanım kolaylığı ve programlama dillerinden bağımsızlığı nedeniyle geliştirilen sınıflandırıcı Anaconda Jupyter notebook üzerinde çalıştırılmaktadır. Jupyter notebook, HDFS dosya sistemindeki klasörleri kolayca yönetmek için HDFS kullanıcısının dosya dizinine (`hdfs:/user/admin`) kurulmuştur. Son olarak ise, Apache Ambari arayüzü kullanılarak gerekli yapılandırma adımları (örneğin; ortam değişkenlerini dışa aktarma (exporting environment variables), erişim izinlerini ayarlama, vb.) gerçekleştirilmiştir. Bu aşamada ayrıca, gerekli Python kütüphaneleri (örneğin; Python-Spark entegrasyonu için pyspark, makine öğrenme için Tensorflow, DL algoritmalarının paralel yürütülmesi için elephas, HCTSA yazılım kütüphanesinin Python entegrasyonu için pyopy) Jupyter notebook ortamına kurulmuş ve gerekli ayarlarlamalar



Şekil 3.16. Kayıp-kaçak tespiti için geliştirilen sınıflandırıcı modelin genel yapısı.

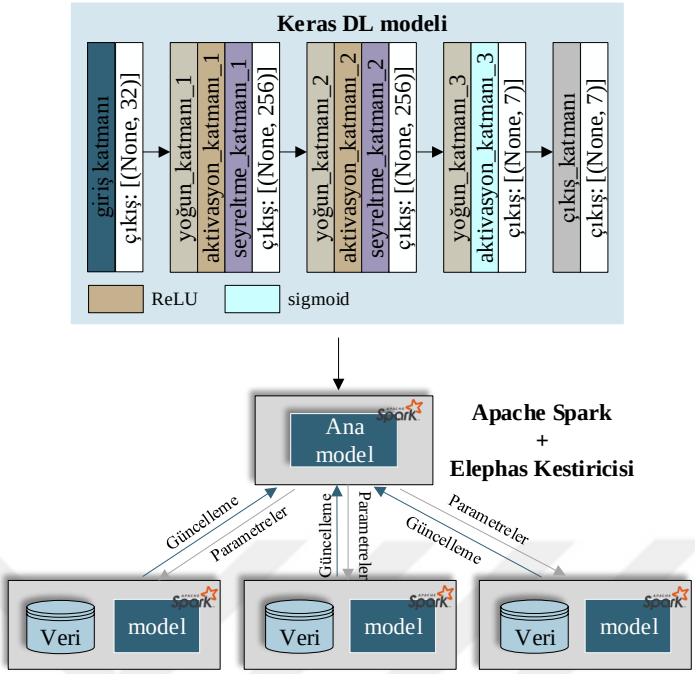
yapılmıştır.

Önerilen modelin özellik öğrenme aşaması, gerçek kayıp-kaçak uygulamalarında kullanılabilecek bilgilendirici özellikleri çıkarmak için üç farklı süreci birlikte kullanmaktadır. İlk aşamada, hctsa yazılım paketi kullanılarak 7764 farklı öznitelik hesaplanmaktadır. Kayıp-kaçak veri kümesindeki her bir örnek (yük profili) için tüm öznitelikler Matlab programının paralel hesaplama özelliği kullanılarak hesaplanmakta ve ardından veri kümesinden (NaN, Inf, -Inf, [] gibi özel değerler içeren sütunlar veya önemli hatalar oluşturan örnekler vb.) gibi örnekler çıkartılmaktadır. İkinci aşamada, elde edilen öznitelik matrisine denklem (3. 51)'deki sağlam (robust) Sigmoid normalizasyon uygulanır [156].

$$\tilde{\mathrm{sf}} = \left[1 + \exp\left(-\frac{\mathrm{f} - \mathrm{m_f}}{1.35\mathrm{r_f}}\right)\right], \tag{3. 51}$$

burada $\tilde{\mathrm{sf}}$, veri kümesinin sağlam Sigmoid normalize edilmiş öznitelik değerlerini, f normalize edilmemiş özniteliklerin vektörünü, $\mathrm{m_f}$ ise f özniteliklerinin medyanını ve $\mathrm{r_f}$ değerlerin çeyrek sapma aralığını temsil etmektedir. Son aşamada ise önemli öznitelikler, NCA tabanlı öznitelik seçme algoritması ve parametre optimizasyonu uygulanarak belirlenir.

Kayıp-kaçak veri kümesinden istatistiksel öznitelik çıkartma, normalizasyon ve öznitelik seçme işlemlerinden sonra, akıllı sayaç öznitelik veri kümesi yerel Linux sunucusundan HDFS ana sunucusuna aktarılır ve standart HDFS kullanıcılarının veri okuma-yazma işlemleri için gerekli izinler ayarlanır. Veri kümesi daha sonra Jupyter ortamında çalışan Spark oturumuna yüklenir. Spark oturumu, Spark fonksiyonlarına erişim noktasıdır ve YARN, atanan görevleri yürüten kaynak yöneticisidir. Bu tez



Şekil 3.17. Önerilen DL sınıflandırıcı modelinin katman yapısı.

çalışmasında geliştirilen kayıp-kaçak tespit modelinde, hesaplama kaynakları YARN yöneticisi tarafından tahsis edilir ve görevler paralel olarak yürütülür.

Dönüşümler ve eylemler, Spark'ın ana bileşenleridir ve çok çeşitli uygulamalar için veri kümesinin dağıtık bir şekilde oluşturulmasına yardımcı olur. Akıllı sayaç öznitelik veri kümesinin dataframe ve RDD dönüşümleri bu aşamada gerçekleştirilmiş ve (VectorIndexer, VectorAssembler) gibi özellik transformatörleri kullanılarak dağıtık veri kümesi formatı oluşturulmuştur.

Bu tez çalışması kapsamında önerilen DL sınıflandıcısının yinelemeli eylemlerini daha hızlı hale getirmek için de RDD veri formatı kullanılmıştır. Geliştirilen DL sınıflandıcısı paralel hesaplama ve Spark entegrasyonu için Elephas kütüphanesini kullanmaktadır. Bu yazılım kütüphanesi sayesinde paralel çalışan herhangi bir sinir ağı yapısı RDD formatında modellenebilmektedir. Bu tez çalışmasında önerilen kayıp-kaçak sınıflandırıcısı, Keras kütüphanesini kullanılarak geliştirilen bir YSA modelidir. Önerilen DL tabanlı sınıflandırıcı modelinin katman yapısı Şekil 3.17.'de verilmiştir.

3.6. Performans Değerlendirme Ölçütleri

3.6.1. Yük Tahmin Modeli için Performans Değerlendirme Ölçütleri

Bu tez çalışmasında geliştirilen kısa dönemli yük tahmin modelinin girdisi, tüketicilerin akıllı sayaçları tarafından kronolojik olarak toplanan elektrik tüketim

değerleridir. Önerilen kısa vadeli yük tahmin modelinin çıktısı ise, t zaman adımında ölçülen tüketim değeri için $t+1$ zaman adımındaki elektrik tüketimi tahminidir. Tez kapsamında geliştirilen hibrit DL modelinin tahmin ufku (forecasting horizon), 1 günden 21 güne kadarlık zaman aralığını kapsayabilmektedir.

Hibrit DL modelinin performansını değerlendirmek için, akıllı sayaç veri kümeleri eğitim aşaması için %70, model doğrulama için %20 ve test için %10 olacak şekilde alt kümelere bölünmüştür. Akıllı sayaç veri kümelerinde bulunan yük profilleri sıralı (sequential) bir yapıya sahip olduğu için veri kümesini alt kümelere ayırma işlemi zaman bilgisi dikkate alınarak gerçekleştirilmiştir. Bu çalışmada kullanılan DEPSAŞ ve SGSC veri kümelerinde yer alan tüketicilerin akıllı sayaç verilerinde meydana gelen periyodik değişimler de dikkate alındığı için, modelin eğitim, doğrulama ve test aşamalarının toplam süresi bir yıl olarak belirlenmiştir.

Önerilen modelin performans değerlendirmesi aşamasıda dört farklı metrik kullanılmıştır. Bu performans metrikleri sırasıyla; RMSE, RMSE'nin değişim katsayısı (coefficient of variation) olarak da bilinen normalize RMSE (NRMSE), MAE ve MAPE'dir. Bu çalışmada kullanılan performans metriklerinin matematiksel ifadeleri denklem 3. 52 ile denklem 3. 55 arasında verilmiştir:

$$\text{RMSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - \widehat{y}_i)^2} \tag{3. 52}$$

$$\text{NRMSE} = \frac{RMSE}{\overline{y}} \times 100\% \tag{3. 53}$$

$$\text{MAE} = \frac{1}{m}\sum_{i=1}^{m}|y_i - \widehat{y}_i| \tag{3. 54}$$

$$\text{MAPE} = \frac{1}{m}\sum_{i=1}^{m}\frac{|y_i - \widehat{y}_i|}{y_i} \times 100\% \tag{3. 55}$$

burada; m: zaman serisindeki toplam veri sayısını, $\overline{y}$: gerçek tüketim değerlerinin ortalamasını, y_i: veri kümesinin normalize edilmemiş gerçek ölçüm değerlerini ve $\widehat{y}_i$ ise zaman serisinin tahmini çıktısını ifade etmektedir.

3.6.2. Kayıp-Kaçak Sınıflandırıcı Modeli için Performans Değerlendirme Ölçütleri

Bu tez çalışmasında geliştirilen kayıp-kaçak sınıflandırıcı modelinin girdisi, HCTSA yazılım kütüphanesinden elde edilen istatistiksel öznitelikler, çıktısı ise FDI tabanlı iletişim alt yapısı saldırıları kullanılarak oluşturulan senaryolar ile normal durumların sınıf etiketleridir.

Önerilen sınıflandırıcı model için kullanılan degerlendirme ölçütleri, hata matrisi (confusion matrix) olarak adlandırılan bir kare matrisin oluşturulmasına dayanan ve eğiticili sınıflandırma problemlerinde sıklıkla tercih edilen ölçütler arasından seçilerek belirlenmiştir. İkili bir sınıflandırma problemin için elde edilen hata matrisinin satırları ve

sütunları sırasıyla tahmin edilen ve gerçek sınıf etiketleri temsil eder. Bu ikili sınıflandırma probleminde sınıflar pozitif ve negatif olmak üzere etiketlendiğinde, hata matrisi [2 × 2]'lik toplamda dört değerden oluşan bir matristir. Bu degerler sırasıyla; pozitif sınıfların doğru bir şekilde tahmin edildiği doğru pozitifler (True Positive–TP) ve yanlış tahmin edildigi yanlış pozitifler (False Positive–FP) ile negatif sınıfa ait örneklerin yanlış tahmin edildigi yanlış negatifler (False Negative–FN) ve dogru tahmin edildiği doğru negatiflerden (True Negative–TN) oluşmaktadır.

Bu tez çalışmasında, sınıflandırıcıların performansı 8 farklı değerlendirme metriği kullanılarak karşılaştırılmıştır. Bu değerlendirme metrikleri sırasıyla; doğruluk (accuracy), duyarlılık (sensitivity), özgüllük (specificity), kesinlik (precision), yanlış pozitif oranı (False Positive Rate-FPR), F_1 skoru, ϕ katsayısı (Matthews Correlation Coefficient-MCC) [172], Cohen'in kappa katsayısı (κ) [173]'dır. Doğruluk, duyarlılık, özgüllük, kesinlik ve FPR metriklerinin matematiksel ifadeleri denklem 3. 56 ile denklem 3. 60 aralığında verilmiştir:

$$\text{Doğruluk} = \frac{\text{TP+TN}}{\text{TP+FP+TN+FN}} \quad (3.\ 56)$$

$$\text{Duyarlılık} = \frac{\text{TP}}{\text{TP+FN}} \quad (3.\ 57)$$

$$\text{Özgüllük} = \frac{\text{TN}}{\text{FP+TN}} \quad (3.\ 58)$$

$$\text{Kesinlik} = \frac{\text{TP}}{\text{TP+FP}} \quad (3.\ 59)$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP+TN}} \quad (3.\ 60)$$

burada TP, belirli bir sınıf için doğru sınıflandırılmış örneklerin sayısına ve TN ise kalan sınıfların doğru sınıflandırılmış örneklerinin toplam sayısıdır. Buna karşılık, FP ve FN ise, sırasıyla belirli bir sınıf için yanlış sınıflandırılmış örneklerini ve kalan sınıfların toplam yanlış sınıflandırılmış örneklerini ifade etmektedir. Hata matrisindeki toplam örnek sayısı ise N_h olarak ($\text{N}_\text{h} = \text{TP} + \text{FP} + \text{TN} + \text{FN}$) temsil edilmektedir. F_1 skorunun matematiksel ifadesi denklem 3. 61'te verilmiştir:

$$F_1 = 2 \times \frac{\text{Kesinlik} \times \text{Duyarlılık}}{\text{Kesinlik} + \text{Duyarlılık}} \quad (3.\ 61)$$

Gerçek ve tahmin edilen sınıflar arasındaki korelasyon katsayısının (MCC) matematiksel ifadesi ise denklem (3. 62)'de verilmiştir:

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FP})\,(\text{TP} + \text{FN})\,(\text{TN} + \text{FP})\,(\text{TN} + \text{FN})}} \quad (3.\ 62)$$

Cohen'in Kappa Katsayısı (κ), sınıflandırıcıların performansını *rastgele bir sınıflandırıcı* performansı ile karşılaştırmamızı olanak tanır [173]. Hata matrisinde, rastgele seçilen bir sınıf etiketi p_1 olasılıkla pozitif sınıfa ve $1 - p_1$ olasılıkla negatif sınıfa dahil olur. Burada,

p_1 olasılık değeri doğru positif sınıf ile yanlış negatif sınıf örneklerinin toplam örnek sayısına oranından elde edilir (p_1 = TP+FN/$\mathrm{N_h}$). Benzer şekilde, bir sınıflandırıcı model p_2 olasılıkla pozitif bir etiket ve $1 - p_2$ olasılıkla negatif bir etiket üretir. Burada, p_2 olasılık değeri doğru pozitif ile yanlış pozitif sınıfına dahil olan örneklerin toplam örnek sayısına bölünmesiyle hesaplanır (p_2 = TP+FP/$\mathrm{N_h}$). Rastgele doğruluk, bu iki sonucun tesadüfen çakışma olasılığıdır ve denklem (3. 63)'teki gibi ifade edilir:

$$\text{Rastgele doğruluk} = \mathrm{p_1 p_2} + (1 - \mathrm{p_1})\,(1 - \mathrm{p_2}) \tag{3. 63}$$

Cohen'in Kappa Katsayısı (κ) ise, bu rastgele referans noktasından kesin sonuca doğru ilerleyen olasılıksal sürecin göreceli gelişimini temsil etmektedir ve matematiksel ifadesi denklem (3. 64)'te verilmiştir:

$$\kappa \;=\; \frac{\text{Doğruluk - Rastgele doğruluk}}{\text{1 - Rastgele doğruluk}} \tag{3. 64}$$

Bu tez çalışmasında, yukarıda bahsedilen performans degerlendirme ölçütleri ve hata matrisleri deneysel çalışmaların sonuçlarından elde edilmektedir. Ayrıca çok sınıflı hata matrisinin elemanları (TP, FP, TN, FN), iki sınıflı versiyondan farklı olarak bire karşı hepsi (One-vs-Rest) yaklaşımı kullanılarak belirlenmektedir. Bu yaklaşımın kullanıldığı çoklu sınıflandırma problemlerinde başlangıçta herhangi bir sınıfın etiketi pozitif sınıf, geri kalan tüm sınıflar ise negatif sınıf olarak etiketlenir. Bu işlem, hata matrisinde yer alan her sınıf için tekrarlanır ve hata matrisinin $[m \times m]$ kadarlık satır ve sütun elemanı hesaplanır. Çoklu sınıflandırma probleminden elde edilen hata matrisinin köşegen eksenindeki (diyagonal) elemanlar dogru sınıflandırılmış veri örneklerinin sayısını, köşegen ekseni dışında kalan (off–diagonal) tüm durumlar ise yanlış sınıflandırılmış veri örneklerinin sayısını temsil etmektedir.

4. Bulgular ve Tartışma

Bu tez çalışmasında geliştirilen modellerin performans değerlendirmesi, literatür çalışmaları dikkate alınarak oluşturulmuş konu başlıklarının detaylı olarak incelenmesini kapsamaktadır.

Bölüm 4.1. ve alt başlıklarında, tez kapsamında geliştirilen kısa dönemli yük tahmin modelinin farklı değerlendirme kriterleri kullanılarak elde edilen bulguları yer almaktadır. Bu bölümlerde, hibrit DL modelinin çıktıları ve geleneksel yöntemler ile karşılaştırılması, yük profillerinden elde edilen düşük boyutlu altuzay temsillerinin incelenmesi, ayrkırılık testi ve parametre optimizasyonu işlemleri ile tahmin ufkunun ve aykırılık analizinin tahmin sonuçları üzerindeki etkisi gibi konular incelenmiştir.

Bölüm 4.2. ve alt başlıklarında ise, kayıp-kaçak sınıflandırma problemi için geliştirilmiş DL modelinden elde edilen bulgular yer almaktadır. Bu bölümlerde, önerilen modelin geleneksel makine öğrenme algoritmaları ile karşılaştırılması, istatistiksel öznitelikleri belirleme aşamasında karşılaşılan belirsizliklerin incelenmesi, sınıf dengesizliği sorunu ve değerlendirme ölçütlerinin yorumlanması aşamasında karşılaşılan güçlükler, veri kalitesi, yanlılık ve ortak değişken kayması problemleri ile önerilen modelin ölçeklenebilirlik analizi konuları incelenmiştir.

4.1. Derin Öğrenme Tabanlı Kısa Dönemli Yük Tahmin Modelinden Elde Edilen Bulgular

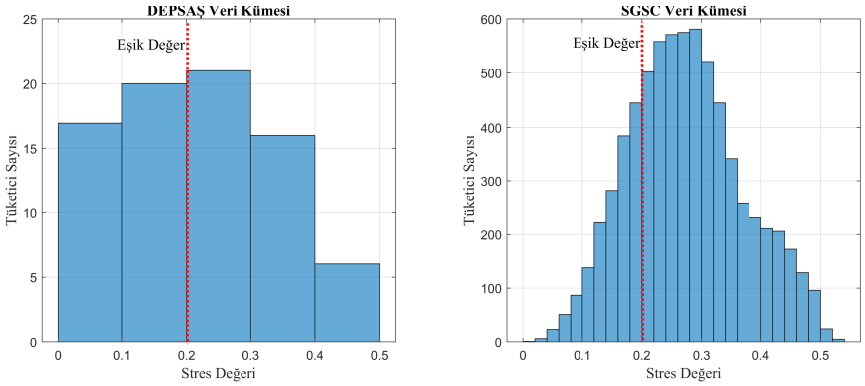
Bu tez çalışmasında önerilen hibrit DL modeli, DEPSAŞ ve SGSC veri kümeleri için tüketici seviyesinde kısa dönemli yük tahmin sonuçları üretmektedir. Bu sayede yük tahmin modeli, tüketiciye özgü enerji tüketim örüntülerini herhangi bir aşırı uyumlama (overfitting) veya yetersiz uyumlama (underfitting) sorunlarını yaşamadan veri odaklı bir yaklaşım kullanarak çıktı üretebilmektedir. Ayrıca tez kapsamında geliştirilen kısa dönemli tahmin modeli veri kümesinde yer alan tüketici sayısından bağımsız olarak düşük miktarda veri içeren kümeler için de kabul edilebilir bir aralıkta tahmin sonuçları üretebilmektedir. Bu işlem, özellikle gerçek hayatta sıkça karşılaşılan kullanılabilir veri (data availability) miktarının az olduğu durumlar için önemli bir avantaj sağlamaktadır. Bu tez çalışmasında geliştirilen kısa dönemli yük tahmin modeli, bir yıllık zaman diliminde kaydedilmiş iki farklı veri kümesine uygulanmıştır. Yük tahmin modelinden elde edilen bulgular, hem Büyük Veri hem de makine öğrenmesi perspektifinden incelenmiştir.

4.1.1. Günlük Yük Profillerinin Çok Boyutlu Ölçeklendirme Kullanılarak Düşük Boyutlu Altuzay Temsillerinin İncelenmesi

Bu çalışmada, $[K \times N]$ boyutlu akıllı sayaç veri kümelerinden metrik olmayan MDS algoritması kullanılarak 2-B alt uzay temsilleri elde edilmiştir. Metrik olmayan MDS algoritmasında uygun boyut sayısının belirlenmesi döngüsel bir süreçtir ancak bu tez

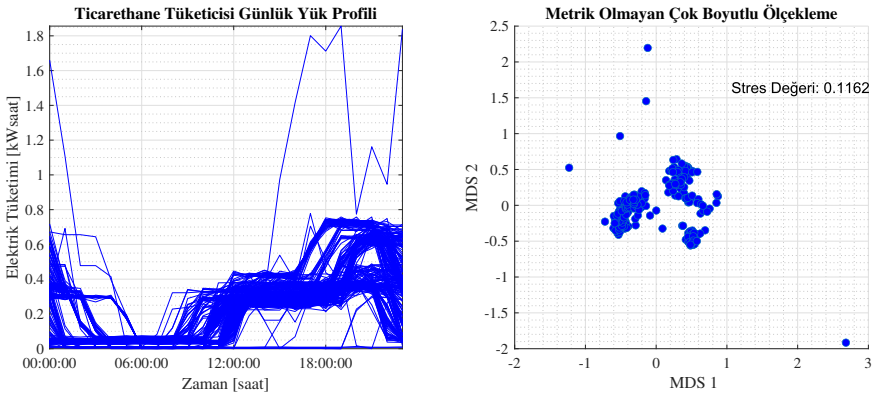
Tablo 4.1. DEPSAŞ ve SGSC veri kümeleri için metrik olmayan MDS algoritmasından elde edilen stres değerleri

MDS Parametresi/ Betimleyici İstatistikler		**DEPSAŞ Veri Kümesi** (100 Tüketici)	**SGSC Veri Kümesi** (7063 Tüketici)
Stres değeri	Minimum	0.0268	0.0153
	Maksimum	0.4932	0.5332
	Ortalama	0.2249	0.2742
	Standart Sapma	0.1211	0.0957
	Varyans	0.0147	0.0092

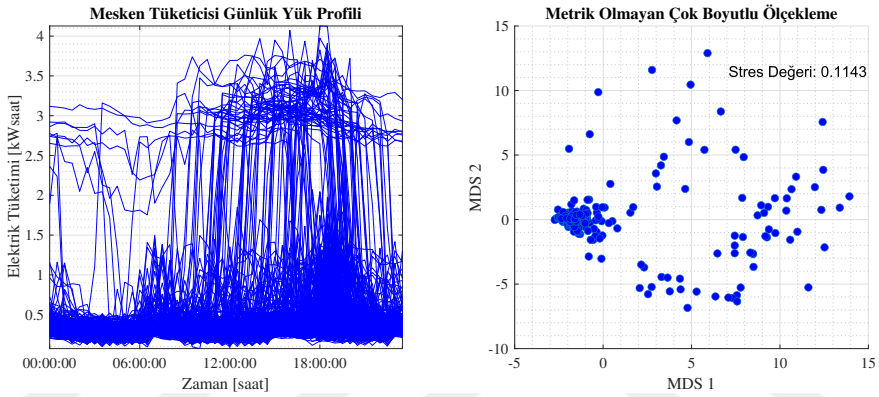


Şekil 4.1. DEPSAŞ ve SGSC veri kümelerinden metrik olmayan MDS algoritması kullanılarak elde edilen stres parametresinin histogramı

çalışmasında önerilen modelde 2-B gösterimine dayalı bir yaklaşım geliştirildiği için, algoritmada varsayılan (default) boyut sayısı iki olarak belirlenmiştir. Bu çalışmada akıllı sayaç veri kümelerinden elde edilen altuzay temsillerinin uygunluğunun değerlendirilmesi aşamasında ise *metrik stres* kriteri dikkate alınmıştır. MDS algoritmasından elde edilen altuzayların uygunluğunu belirleyen *metrik stres* kriteri, mesafe matrisinin dördüncü kuvvetlerinin toplamından elde edilen normalize edilmiş gerilimin karesidir [146], [174]. Bu kriterde ampirik kural (rule of thumb) olarak, 0.2 civarında veya üzerindeki stres değerine sahip olan bir metrik olmayan MDS koordinasyonu şüpheli olarak kabul edilmekte ve 0.3'e yaklaşan stres değeri ise, mesafe matrisinin büyükten küçüğe doğru sıralamasının keyfi olduğunu göstermektedir. Stres değeri, 0.1'e eşit veya altındaki durumlarda normal kabul edilirken 0.05'e eşit veya altındaki değerler için ise iyi uyumu olarak değerlendirilmektedir. Burada dikkat edilmesi gereken bir konu ise, algoritmanın daha yüksek boyutta veri kümesini düzenlenmesine izin vermek stresi azaltabilir ancak üçten fazla boyuttaki altuzay temsillerinin yorumlanması daha zorlu bir süreçtir. Metrik olmayan MDS algoritmasının DEPSAŞ ve SGSC veri kümelerinden elde edilen stres değerleri Tablo 4.1.'de verilmiştir. DEPSAŞ ve SGSC veri kümeleri için belirlenen stres değerlerinin histogramı ise Şekil 4.1.'de verilmiştir. Şekil 4.1.'de kırmızı çizgi ile gösterilen eşik değer yukarıda kısaca özetlenen ampirik kural dikkate alınarak belirlenmiştir. DEPSAŞ ve SGSC veri kümelerinde yer alan bir ticarethane ve mesken tüketicisinin günlük yük profilleri ile MDS altuzay gösterimi ise Şekil 4.2.'de verilmiştir.



(a) Ticarethane

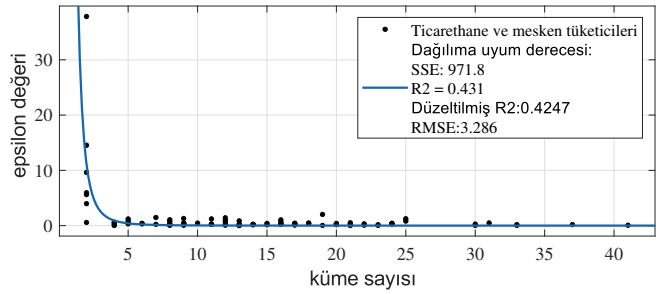


(b) Mesken

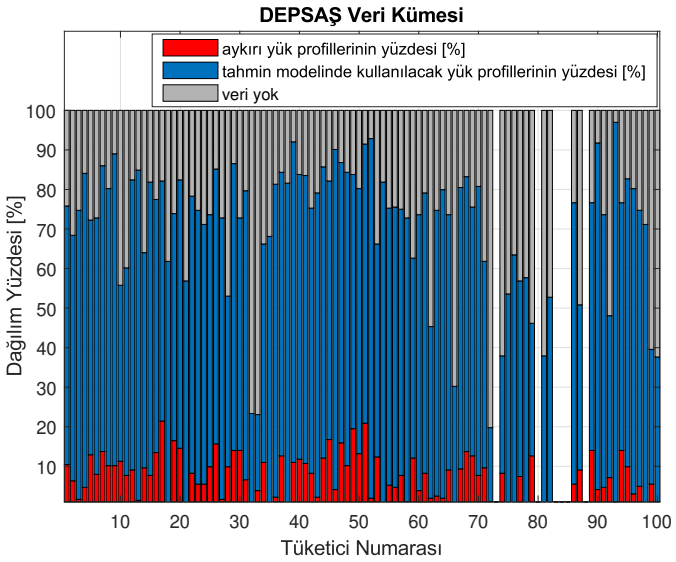
Şekil 4.2. DEPSAŞ ve SGSC veri kümesinde yer alan bir (a) ticarethane ve (b) mesken tüketicisinin günlük yük profilleri ve metrik olmayan MDS altuzay gösterimi.

4.1.2. Günlük Yük Profillerinin Yoğunluk Tabanlı Aykırılık Testi ve Parametre Optimizasyonu

Bölüm 4.1.1.'de gerçekleştirilen boyut indirgeme işleminden sonra, BDE-DBSCAN algoritması kullanılarak olağan dışı (anomalous) günlük yük profilleri tespit edilmiş ve veri kümesinden çıkarılmıştır. Kümeleme doğrulama metriği olarak RMSSTD indeksi, hem optimizasyon aşamasında maliyet fonksiyonu olarak hem de kümeleme performans değerlendirmesi ölçütü olarak kullanılmıştır. Parametre optimizasyonu sürecinde kümeleme işlemi, her tüketici için 30 kez (epok=30) tekrarlanmıştır. RMSSTD indeksinin maliyet fonksiyonu olarak kullanılması, DEPSAŞ veri kümesinde yer alan bazı gizli tüketim örüntülerini ortaya çıkarmıştır. Bu örüntülerden bir tanesi Şekil 4.3.'te gösterilmiştir. Şekil 4.3.'te, DEPSAŞ veri kümesinde yer alan çoğu mesken ve ticarethane tüketicisinin küme sayısı 5 ile 25 arasında değişmektedir ve bu aralık bireysel tüketim seviyesi dikkate alındığında kabul edilebilir bir küme sayısı aralığıdır [175]. Ayrıca, veri kümesinde yer alan tüm tüketiciler için belirlenen olağan dışı günlük yük profillerinin (aykırı günlük yük profillerinin) yüzdeleri ise Şekil 4.4.'te gösterilmektedir. Bu aralık,



Şekil 4.3. DEPSAŞ veri kümesinde yer alan ticarethane ve mesken tüketicilerinin epsilon değerleri ile küme sayısı arasındaki ilişki [176].

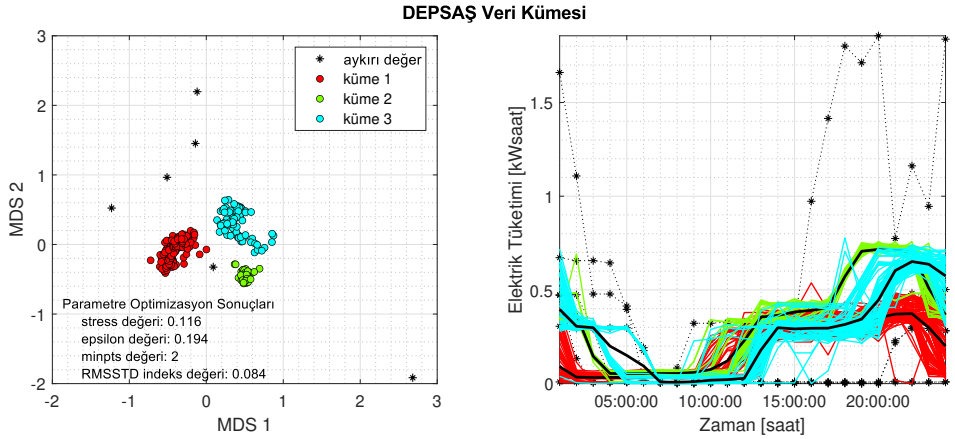


Şekil 4.4. DEPSAŞ veri kümesindeki farklı kategorilerin yüzdesi [176].

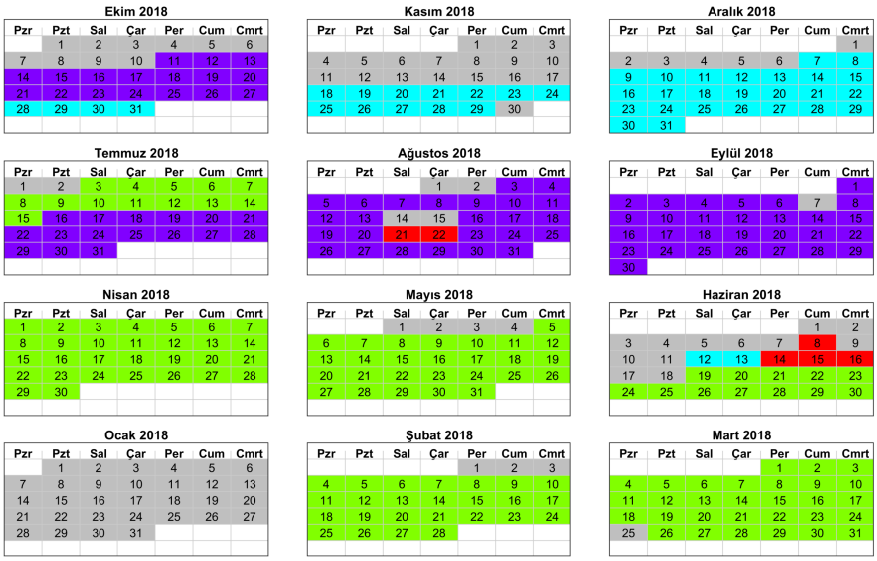
DEPSAŞ veri kümesi için ortalama olarak % 8 ile % 12 arasında değişmektedir. Burada veri kümesinde çok düşük seviyelerde tüketim gerçekleştiren tüketiciler dikkate alınmamış ve Şekil 4.4.'te beyaz-boş bölgeler olarak gösterilmiştir.

DEPSAŞ veri kümesinde yer alan bir ticarethaneye ait yük profilleri ve kümeleme sonucu Şekil 4.5.'te verilmiştir. Şekil 4.5.'teki 2-B altuzay temsilindeki her bir nokta, ticarethanenin bir yıllık dönemdeki günlük profilini temsil etmektedir. Veri kümesinde kümeleme sonuçları tarihsel olarak sıralandığında ise başka bir tüketim örüntüsü elde edilmektedir. Şekil 4.5.'teki ticarethanenin kümeleme sonuçlarının takvim gösterimi Şekil 4.6.'da verilmiştir. Takvimdeki gri ve kırmızı ile gösterilen günler, sırasıyla tüketim değerlerinin elde edilemediği durumları ve Şekil 4.5.'te siyah yıldız şeklinde etiketlenen olağan dışı yük profillerini ifade etmektedir.

Şekil 4.6.'daki kümeleme takviminde olağan dışı olarak etiketlenen günler, Hawkins'in aykırı değer tanımına "*farklı bir mekanizma tarafından oluşturulan*" güzel bir



Şekil 4.5. DEPSAŞ veri kümesindeki bir ticarethanenin metrik olmayan MDS altuzay temsili ve BDE-DBSCAN kümeleme analizi sonucu



Şekil 4.6. DEPSAŞ veri kümesindeki bir ticarethanenin kümeleme sonuçlarının tarihsel gösterimi.

örnektir [177]. Bu günler herhangi bir kümeye dahil edilemese de dikkatle incelendiğinde bu günlerin bir kısmının Türkiye'de kutlanan dini bayramlara denk geldiği gözlemlenmiştir. Diğer durumlar ise, Şekil 4.5.'te farklı renklerle temsil edilen üç kümeyi temsil etmektedir. Burada bir tüketim davranışından diğerine geçişin genellikle düzenli bir yapıya sahip olduğu Şekil 4.6.'dan anlaşılabilir. DEPSAŞ veri kümesindeki bu ticarethane için geçiş aylarında (ilkbahardan yaza veya sonbahardan kışa gibi) program veya iş yükü gibi nedenlerden dolayı herhangi bir küme değişikliği gözlenmese de, mesken tüketicilerinin yaklaşık yarısında geçiş aylarında (örneğin; Nisan-Mayıs, Eylül-Ekim, vb.) küme sayısında artış gözlenmiştir.

Bu çalışmada, SGSC veri kümesi için elde edilen en uygun epsilon değerleri stres parametresi ve regresyon karar ağaçları kullanılarak Ek-3'de yer alan Şekil E5.1.'de

kompakt bir yapıda gösterilmiştir. Şekil E5.1.'de graf yapı ile gösterilen karar ağacındaki $x1$ değişkeni, metrik olmayan MDS algoritmasının *stres* parametresini, her bir düğümün sonra erdiği noktada yer alan değerler ise kural tabanlı olarak belirlenmiş DBSCAN algoritmasının en uygun *epsilon* değerlerini ifade etmektedir. Bu sayede DBSCAN algoritmasının önsel (a priori) olarak belirlenmesi gereken *epsilon* parametresi varsayımsal ve deneme-yanılma yaklaşımları kullanılmadan belirlenebilir. Ayrıca, SGSC veri kümesinde yer alan tüketicilerin % 93.44'lük bölümü için DBSCAN algoritmasının *minpts* parametresi en uygun değer olarak sırasıyla 1 ve 2 olarak belirlenmiştir. SGSC veri kümesinde yer alan tüketicilerin yaklaşık %41.39'u için ortalama aykırı değer oranı %11-%13 aralığında değişmektedir ve geriye kalan %58.61'lik kısımın ise aykırı değer oranı %10'dan daha küçüktür. DEPSAŞ ve SGSC veri kümelerine ait RMSSTD indeksi ve küme sayısı istatistikleri Tablo 4.2.'de verilmiştir.

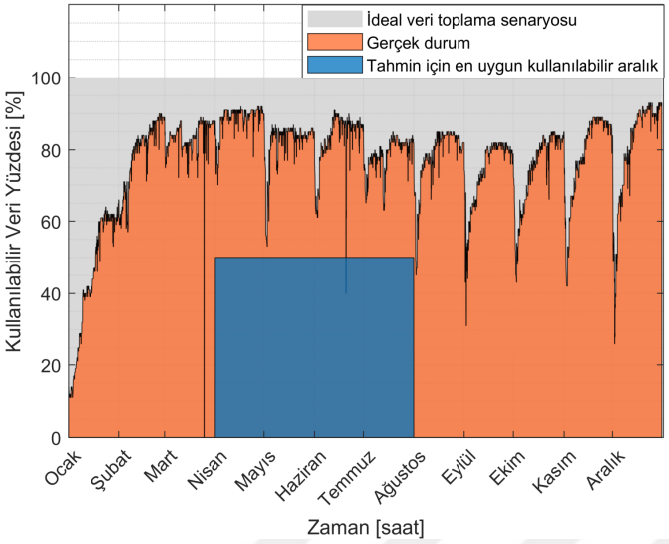
Tablo 4.2. DEPSAŞ ve SGSC veri kümelerinden elde edilen RMSSTD indeksi ve küme sayısına ait istatistikler

Optimizasyon Parametreleri/ Betimleyici İstatistikler		**DEPSAŞ Veri Kümesi** (100 Tüketici)	**SGSC Veri Kümesi** (7063 Tüketici)
RMSSTD indeksi	Minimum	0.0185	0.0103
	Maksimum	1.3700	5.7432
	Ortalama	0.2338	0.3678
	Standart Sapma	0.2222	0.3154
	Varyans	0.0494	0.0995
Küme sayısı	Minimum	2	2
	Maksimum	41	66
	Ortalama	15	20

4.1.3. Derin Öğrenme Modelinin Geleneksel Tahmin Modelleri ile Karşılaştırılması ve Elde Edilen Bulgular

Mesken tüketicilerinin elektrik tüketiminde insan davranışından kaynaklanan tutarsız ve döngüsel olmayan tüketim örüntüleri bulunmaktadır. Buna karşın ticarethane, kamu veya hizmet binaları gibi topluma hizmet veren alanlarda kaydedilen elektrik tüketim verisinde mesken tüketicilerine kıyasla daha düzenli tüketim örüntülerine sahip olsa da, iş yükü veya planlanmamış olaylar gibi farklı nedenlerden kaynaklanan belirsizlikler mevcuttur.

Tahmin modeli geliştirme aşamasında karşılaşılan diğer bir sorun ise AMI hizmetlerinde meydana gelen aksamlar ve akıllı sayaçlar ile DSO arasındaki haberleşme problemleridir. Bu tez çalışmasında kullanılan DEPSAŞ veri kümesinde bu durum belirgin bir şekilde gözlemlenmiştir. Mesken tüketicilerinin akıllı sayaçları için ideal veri toplama (data acquisition) senaryosu ve gerçek durum, Şekil 4.7.'de sırasıyla gri ve turuncu renkli alanlar olarak gösterilmektedir. Şekil 4.7.'deki turuncu alanlar, DEPSAŞ veri kümesinde yer alan akıllı sayaçların neredeyse yarısının bir yıllık süre boyunca sürekli veriye sahip olmadığını göstermektedir. Ayrıca, Şekil 4.7.'deki mavi alan, farklı mesken tüketicilerinin tüketim davranışlarının karşılaştırılması için kullanılabilir tek uygun aralıktır.



Şekil 4.7. DEPSAŞ veri kümesinde yer alan tüketiciler için en uygun tahmin aralığı

DEPSAŞ veri kümesinde, 2018 yılı için 56 meskenin ve 44 ticarethanenin günlük yük profilleri saatlik zaman çözünürlükte bulunmaktadır ve eksik (NaN) değerlerden oluşan günlük yük profilleri veri kümesinden çıkarıldığında geriye toplamda 28.876 günlük yük profili kalmaktadır. DL modelinin oluşturulması ve test edilmesi aşamasında ise, aykırı değer tespiti sonrasında kullanılabilir olarak belirlenen toplamda 22.505 günlük yük profili bulunmaktadır.

Bu çalışmada, geliştirilen hibrit DL tabanlı kısa dönemli yük tahmin modelinin sonuçları geleneksel makine öğrenmesi yöntemlerinin (örneğin; destek vektör regresyonu (Support Vector Regression-SVR), ANN, DT, vb.) ve DL yöntemlerinin (örneğin; CNN, LSTM, BLSTM, vb.) sonuçları ile karşılaştırılmıştır. DL tabanlı yük tahmin modelleri son yıllarda hem dağıtım seviyesinde hem de tüketici seviyesinde kabul edilebilir hata oranlarıyla tahmin üretebildiği için, karşılaştırma aşamasında aynı zamanda hibrit modeller de dikkate alınmıştır (örneğin; CNN-LSTM, CNN-BLSTM, vb.) [178].

Bu çalışmada tercih edilen karşılaştırma yöntemlerinin parametreleri tekrarlanabilirlik ve yorumlanabilirlik kriterleri dikkate alınarak Tablo 4.3.'te verilmiştir. Burada kullanılan SVR ve DT modellerinin parametre ayarları, sırasıyla scikit-learn kullanıcı kılavuzlarında yer alan açıklamalar dikkate alınarak belirlenmiştir [179], [180].

Bu çalışmada önerilen hibrit DL modelinin geleneksel makine öğrenme modelleri ile karşılaştırması Tablo 4.4.'te verilmiştir. Bu çalışmada kullanılan SVR ve DT modellerinden DL modelleriyle kıyaslandığında tüm performans ölçütlerinde en düşük performans değerleri elde edilmiştir. Ayrıca, aykırılık analizi gerçekleştirilmeden ham verinin (raw data) kullanıldığı senaryoda DL modelleri, önerilen hibrit DL modeline kıyasla daha yüksek tahmin hata değerlerine sahiptir. ANN modeli, tüm değerlendirme metriklerinde SVR ve DT modellerinden daha iyi performans gösterirken, CNN, LSTM,

Tablo 4.3. Karşılaştırma modellerinin parametre ayarları

Yöntem	Karşılaştırma yöntemlerinin tasarım parametreleri*					
	Gizli Katman Nöron Sayısı	Aktivasyon fonksiyonu	Veri grup büyüklüğü	Filtre boyutu	Örnekleme boyutu	İyileştirici
ANN	500	sigmoid	500	-	-	SGD**
CNN	-	ReLU	200	64	-	ADAM
LSTM	200	sigmoid	200	-	-	SGD
BLSTM	50	-	200	-	-	ADAM
CNN-LSTM	50/25	ReLU	200	64	1	ADAM

* Tüm modeller için epok sayısı 100 ve kayıp fonksiyonu MSE olarak belirlenmiştir.
** rasgele gradyan inişi (Stochastic Gradient Descent-SGD).

Tablo 4.4. Önerilen hibrit DL modelinin farklı makine öğrenme ve DL modelleri ile karşılaştırılması

Yöntem	RMSE (kW)	NRMSE (%)	MAE (kW)	MAPE (%)
SVR	0.194	206.107	0.125	55.826
DT	0.242	102.470	0.141	141.067
ANN	0.154	82.529	0.109	54.719
CNN	0.143	61.190	0.088	57.675
LSTM	0.129	67.580	0.077	45.084
BLSTM	0.131	54.825	0.078	46.452
CNN-LSTM	0.148	63.624	0.088	46.755
Önerilen model	**0.108**	**55.432**	**0.058**	**36.748**

BLSTM, CNN–LSTM modelleri birbirine nispeten yakın performans göstermiştir. Önerilen CNN-BLSTM modeli ise tüm değerlendirme metriklerinde daha iyi performans göstermiştir. Tablo 4.3.'da gösterildiği gibi önerilen hibrit DL modeli, tüm makine öğrenme ve DL modellerine kıyasla daha düşük hata oranı sahiptir.

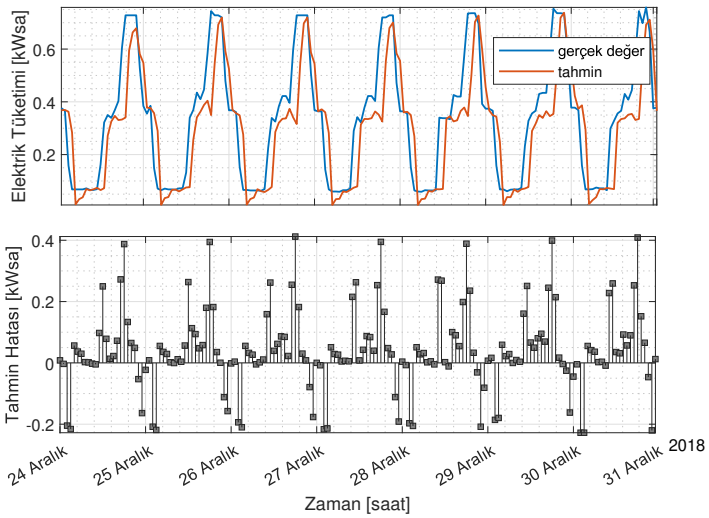
Bu çalışmada, önerilen hibrit DL modelinin performansı SGSC veri kümesinde de test edilmiştir. Burada kullanılan veri kümesi literatür çalışmaları dikkate alınarak oluşturulmuş toplamda 69 farklı mesken tüketicisinin 1 Haziran 2013 ile 31 Ağustos 2013 tarihleri arasındaki verisinden oluşmaktadır [17], [41]. Tablo 4.5.'te, önerilen hibrit DL modeli aynı test aralığında 31.481 ortalama MAPE değeri ile benzer çalışmalara kıyasla daha düşük hata değerine sahiptir. Şekil 4.6.'daki ticarethane için hibrit DL modelinin tahmini yük profili Şekil 4.8.'de verilmiştir.

Bu çalışmada kullanılan DEPSAŞ veri kümesi, mesken ve ticarethane olarak iki farklı kategoride yer alan tüketicileri içermektedir. Ayrıca, model değerlendirmesinin daha

Tablo 4.5. Önerilen modelin MAPE kriteri kullanılarak farklı çalışmalar ile karşılaştırılması

Referans	Veri kümesi	Veri kümesi boyutu*	Tahmin yöntemi	Test aralığı	MAPE (%)
Ref. [17]	SGSC	69	LSTM	23.08.2013-31.08.2013	44.060
Ref. [41]	SGSC	69	CNN-LSTM	23.08.2013-31.08.2013	40.380
Önerilen model	**SGSC**	**69**	**CNN-BLSTM**	**23.08.2013-31.08.2013**	**31.481**

* Veri kümesi boyutu, kısa dönemli yük tahmin deneyleri için seçilen alt kümeler dikkate alınarak belirlenmiştir (tüketici sayısı).



Şekil 4.8. Önerilen hibrit DL modelinin DEPSAŞ veri kümesinde yer alan ticari bir bina için ürettiği tahmini yük profili

sağlıklı bir şekilde yapılabilmesi için verilerin çok fazla eksik değer içermediği zaman aralıklarının belirlenebilmesi de oldukça önemlidir. Bölüm 4.1.3.'teki Şekil 4.7.'de gösterildiği üzere, DEPSAŞ veri kümesi için Nisan ve Ağustos 2018 ayları arasındaki zaman dilimi farklı tüketicilerin karşılaştırılabilmesi için en uygun aralıktır. Önerilen hibrit DL modelinin Nisan-Ağustos 2018 dönemi için farklı tüketici sınıflarına ait ortalama tahmin performansı Tablo 4.6.'da verilmiştir. DEPSAŞ veri kümesinde yer alan mesken ve

Tablo 4.6. Nisan-Ağustos 2018 döneminde DEPSAŞ veri kümesindeki farklı tüketici sınıfları için ortalama tahmin performansı

Tüketici sınıfı	RMSE (kW)	NRMSE (%)	MAE (kW)	MAPE (%)
Mesken	0.154	71.728	0.089	33.736
Ticarethane	0.135	26.672	0.084	27.374

ticarethane tüketicileri için ortalama RMSE değerleri sırasıyla 0.154 ve 0.135, NRMSE değerleri ise %71.73 ve %26.67 olarak hesaplanmıştır. MAE ve MAPE metrikleri kullanılarak elde edilen ortalama hata değerleri mesken sınıfında yer alan tüketiciler için sırasıyla 0.089 ve %33.74, ticarethane sınıfında yer alan tüketiciler için ise 0.084 ve %27.37'dir.

4.1.4. Değişken Zaman Çözünürlüğünün Yük Tahmin Modelinin Performansı Üzerindeki Etkisinin İncelenmesi

Bu bölümde, önerilen hibrit DL modelinin performansı değişken zaman çözünürlüğü (time resolution) senaryoları dikkate alınarak değerlendirilmiştir. Bu amaçla

Tablo 4.7. Önerilen yöntemin farklı zaman adımlarındaki ortalama hata değerleri

Veri Kümesi	Zaman Çözünürlüğü	RMSE (kW)	NRMSE (%)	MAE (kW)	MAPE (%)
DEPSAŞ Veri Kümesi	1 saat	0.108	55.432	0.058	36.748
	24 saat	0.108	56.478	0.059	36.911
	48 saat	0.108	56.144	0.059	38.008
SGSC Veri Kümesi	1 saat	0.1149	46.2344	0.0744	33.4774
	24 saat	0.1566	61.1649	0.0958	60.7743
	48 saat	0.1681	64.4710	0.1061	67.2889

önerilen modelde, gelecek dönem saatlik elektrik tüketimi tahmini için 1 saat, 24 saat ve 48 saat gibi farklı zaman çözünürleri tanımlanmıştır. Örneğin; CNN-BLSTM modelinde $t-n$ zaman adımındaki n adet giriş verisi ($n \in \{1, 24, 48\}$), $t+1$ zaman adımındaki saatlik enerji tüketim tahmini için kullanılmaktadır. Tablo 4.7.'de önerilen kısa dönemli yük tahmin modelinin test aşamasındaki farklı zaman adımlarında elde edilen ortalama hata oranları verilmiştir.

Tablo 4.7.'de, DEPSAŞ veri kümesiyle gerçekleştirilen yük tahmini uygulamasında geriye dönük farklı zaman adımlarındaki tüketim değerlerinin kullanılması (örneğin; 1 saat, 24 saat, 48 saat) performans değerlendirme metriklerinin sonuçlarda belirgin bir değişikliğe neden olmamıştır. Bununla birlikte, SGSC veri kümesinde farklı zaman adımlarının kullanılması özellikle NRMSE ve MAPE değerlerinde sırasıyla %18 ve %34 oranında belirgin bir farklılığa neden olmuştur. Her iki veri kümesinde de geriye dönük olarak saatlik tüketim değerlerinin kullanılması en düşük hata oranlarının elde edilmesine katkıda bulunmuştur. Burada önerilen modelde saatlik zaman çözünürlüğünün kullanılması, günlük ve iki günlük zaman çözünürlüklerinin kullanılmasına kıyasla daha doğru tahmin sonuçları üretmektedir.

4.1.5. Aykırılık Analizi ve Tüketici Seviyesindeki Tahmin Üzerindeki Etkisi

Bu bölümde, önerilen aykırı değer tespitinin kısa dönemli yük tahmini sonuçlarındaki etkisi incelenmiştir. DEPSAŞ ve SGSC veri kümeleri aykırı günlük yük profillerinin olduğu ve veri kümesinden çıkarıldığı durumlar dikkate alınarak yeniden düzenlenmiştir. Önerilen hibrit DL modelinin eğitim, doğrulama ve test aşamasında elde edilen ortalama hata oranları Tablo 4.8.'de verilmiştir. Burada aykırı yük profillerinin veri kümesinde bulunduğu durum *w* (with outliers) olarak, veri kümesinden çıkarıldığı durum ise *wo* (without outliers) şeklinde ifade edilmiştir.

Tablo 4.8.'den de anlaşıldığı üzere, aykırı değer tespiti hibrit DL modelinin eğitim, doğrulama ve test aşamasında elde edilen tahmin sonuçlarına olumlu yönde katkı sağlamaktadır. Burada, DEPSAŞ veri kümesi için test aşamasında NRMSE ve MAPE değerlerinde sırasıyla % 7 ve %5'lik bir azalma, SGSC veri kümesi için ise % 5 ve % 3'lük bir azalma gözlenmiştir. Bu bölümde aynı zamanda, aykırılık analizinin her bir tüketicinin tahmin sonuçlarındaki değişimi incelenerek Tablo 4.9.'da beş farklı aralıkta gruplandırılmıştır.

Tablo 4.8. Aykırılık analizinin kısa dönemli yük tahmin modeli üzerindeki etkisi

Veri Kümesi	Model	RMSE (kW)	NRMSE (%)	MAE (kW)	MAPE (%)
DEPSAŞ (w)	Eğitim	0.120	55.673	0.066	40.610
	Doğrulama	0.115	50.272	0.067	39.984
	Test	0.141	62.696	0.085	41.035
DEPSAŞ (wo)	Eğitim	0.095	50.454	0.049	37.461
	Doğrulama	0.097	46.292	0.052	35.156
	Test	0.108	55.432	0.058	36.748
SGSC (w)	Eğitim	0.147	57.759	0.091	47.391
	Doğrulama	0.120	46.684	0.082	39.105
	Test	0.132	51.243	0.087	36.324
SGSC (wo)	Eğitim	0.129	52.609	0.077	45.994
	Doğrulama	0.111	45.002	0.073	56.874
	Test	0.115	46.234	0.074	33.477

Tablo 4.9. SGSC test veri kümesindeki hata oranlarında meydana gelen iyileşmenin yüzdelik (%) dağılımı

Veri Kümesi	İyileşme Oranı	Tüketici Sınıfı (Mesken ve Ticarethaneler) *			
		RMSE (kW)	NRMSE (%)	MAE (kW)	MAPE (%)
DEPSAŞ	(<0)%	18	26	16	24
	(0-25)%	35	25	29	35
	(25-50)%	24	25	26	16
	(50-75)%	6	7	12	12
	(75-100)%	5	5	5	1
SGSC	(<0)%	1724	2362	1492	2108
	(0-25)%	3795	3420	3700	3743
	(25-50)%	1254	1027	1607	979
	(50-75)%	182	151	159	133
	(75-100)%	21	16	18	13

* Bölüm 4.1.2.'de belirtilen bazı sorunlardan dolayı iyileştirme karşılaştırması için mevcut toplam tüketici sayısı DEPSAŞ veri kümesi için 88, SGSC veri kümesi için 6976'dır.

4.2. Derin Öğrenme Tabanlı Kayıp-Kaçak Sınıflandırıcı Modelinden Elde Edilen Bulgular

Kayıp-kaçak tespit yöntemi kapsamlı istatistiksel öznitelik çıkarma, öznitelik seçme ve büyük veri analitiği ile entegre sınıflandırıcı modeli oluşturma aşamalarından oluşmaktadır. Bu tez çalışmasında kayıp-kaçak sınıflandırma işlemi dağıtım hattı seviyesinde aynı trafo istasyonuna bağlı tüketicilerden oluşan bir veri kümesine uygulanmıştır. Kayıp-kaçak veri kümesi altı farklı FDI tabanlı sayaç müdehalesi senaryosundan oluşmaktadır ve önerilen DL modelinin performansı farklı kriterler dikkate alınarak bu veri kümesi üzerinde incelenmiştir. Önerilen sınıflandırıcı model aynı zamanda, tüketici yük profili parametrelerini ve özniteliklerini dikkate alan yeni ve kapsamlı bir NTL tespit modelini ve en ileri makine öğrenme yöntemlerinin büyük veri teknolojileri ile birlikte kullanıldığı sistem tasarımlarının bir uygulamasını temsil etmektedir.

Bu çalışmada önerilen sınıflandırıcı model, kayıp-kaçak tespiti aşamasında karşılaşılan kısıtlamalar ve engeller dikkate alınarak beş farklı kategoride test edilmiştir. Bu kategoriler sırasıyla; öznitelik belirleme ve tanımlamada karşılaşılan belirsizlikler,

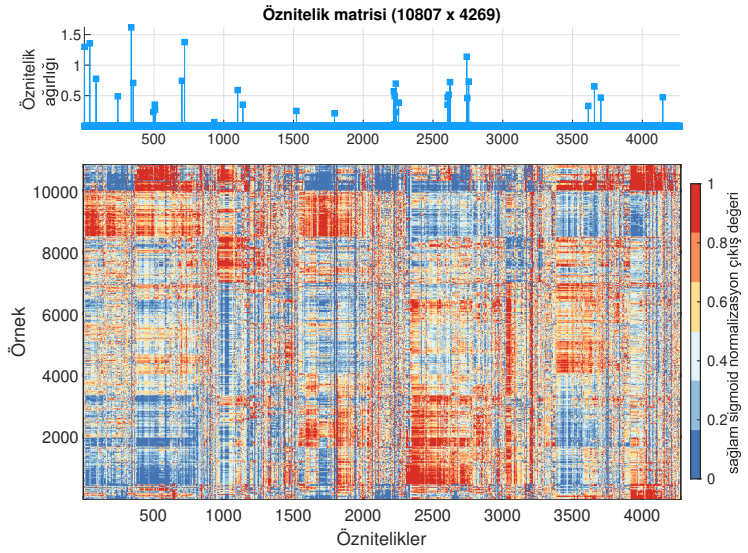
geleneksel yöntemler ile karşılaştırma, sınıf dengesizliği ve performans ölçütlerinin yorumlanmasında karşılaşılan hatalar, veri kalitesi, yanlılık ve ortak değişken kayması sorunlarının incelenmesi aşamalarından oluşmaktadır.

4.2.1. Kapsamlı İstatistiksel Öznitelik Belirleme ve Tanımlama Süreçlerindeki Belirsizliklerin İncelenmesi

Bu çalışmada önerilen öznitelik öğrenme çerçevesi HCTSA yazılım paketini ve eklentilerini kullanmaktadır. Bu bölümde, FDI tabanlı kayıp-kaçak senaryolarının tespiti için HCTSA yazılım paketindeki uygun ve sağlam (robust) istatistiksel öznitelikler detaylı olarak araştırılmış ve önemli öznitelik NCA yöntemi kullanılarak belirlenmiştir. Kayıp-kaçak veri kümesindeki yük profilleri ve FDI tabanlı kayıp-kaçak senaryolarının uygulandığı değiştirilmiş yük profilleri için HCTSA yazılım paketinde bulunan 7764 istatistiksel öznitelik hesaplanmıştır. Kayıp-kaçak veri kümesi, NaN değer içeren yük profilleri çıkarıldıktan sonra 2019 için 31 tüketiciden toplamda 7807 farklı yük profilli içermektedir. Bölüm 3.1.3.'te de açıklandığı gibi, kayıp-kaçak veri kümesinde maksimum NTL yüzdesi %30 olarak belirlenmiş ve ardından her bir FDI alt kümesinin 500 örnek içerdiği 3000 yeni değiştirilmiş yük profili veri kümesine eklenmiştir. HCTSA yazılım paketinde Varsayılan seçenek olarak, her bir örnek (yük profili) için tüm öznitelikler hesaplanır ve ardından özel değere sahip örnekler (NaN, $+\infty$, $-\infty$, [], vb.) veri kümesinden çıkarılır. Daha sonra ise, Şekil 4.9.'da gösterilen 10807×4269 boyutundaki sağlam Sigmoid normalizasyon uygulanmış veri matrisini elde edilmiştir. Son olarak, NCA tabanlı bir öznitelik seçim stratejisi kullanılarak optimal öznitelikler belirlenmiştir. Bu çalışmada NCA algoritmasının optimum düzenlileştirme parametresi λ ve öznitelik ağırlıkları, 5-kat çapraz geçerlilik sınaması (5-fold cross validation) ve stokastik gradyan iniş (Stochastic Gradient Descent-SGD) iyileştirici kullanılarak belirlenmiştir.

Çapraz geçerlilik sınamasında çok parametreli bir sınıflandırıcı model, öznitelik değişimine göre başarımın değişimini ölçmek ve kıyaslama yapabilmek amacıyla k-kat (k-fold) rastgele şekilde alt gruplara bölünmüş veri kümelerinde eğitilir ve test edilir. Burada kullanılan k-kat çapraz geçerlilik sınaması işleminde, veri kümeleri k eşit parçaya bölünür ve veri kümesinin k parçası test aşamasında kullanılırken geriye kalan $(k-1)$ parçası ise modelin eğitim aşamasına dahil edilir. Bu sayede k adet döngü sonrasında her bir döngü için bir tane olacak şekilde k kadar (%) cinsinden başarım değeri veya hata değeri elde edilir. Sınıflandırıcı modelinin genel başarımı, bu degerlerin ortalaması alınarak elde edilen tek bir başarım ya da hata değeridir.

Düzenlileştirme parametresi optimizasyonunda kayıp fonksiyonu olarak belirlenen yanlış sınıflandırma hatası (misclassification error), seçilen her bir λ değeri ve k adet bölüntülü matris (partition matrix) için hesaplanmaktadır. Bu çalışmada, SGD tabanlı iyileştirici 19. denemede λ değeri için Tablo 4.10.'da gösterildiği gibi [$\lambda = 0.0024$, ortalama kayıp $= 0.0699$] şeklinde yerel bir minimum çözüm üretmiştir. Bu çalışmada, öznitelik ağırlık katsayılarının nihai sonuçları λ için belirlenen en uygun değer,



Şekil 4.9. Sağlam sigmoid normalize edilmiş öznitelik veri kümesi ve NCA tabanlı öznitelik seçimi kullanılarak elde edilen önemli öznitelikler. Öznitelik veri kümesinde her satır örneği yük profillerini, her sütun örneği ise (NaN, $+\infty$, $-\infty$, []) gibi özel değerler veri kümesinden kaldırıldıktan sonra önceden tanımlanmış bir istatistiksel özniteliğin bir fonksiyonudur. Öznitelik veri matrisi benzer satırları yan yana getirmek için hiyerarşik bağlantı kümelemesi kullanılarak yeniden sıralanmıştır [181].

sınırlı belleğe sahip (limited memory) Broyden-Fletcher-Goldfarb-Shanno (LBFGS) algoritması ve weak Wolfe doğru üzerinde arama stratejisi kullanılarak belirlenmiştir.

Düzenlileştirme parametresi λ, Şekil 4.9.'da gösterildiği üzere sınıflandırıcı sonuçlarına herhangi bir katkı sağlamayan özniteliklerin ağırlık katsayılarını sıfır olarak belirler. Bu çalışmada ayrıca, tüm öznitelikler için tek bir düzenleme parametresi tanımlandığından, özniteliklerin ağırlıkları birbiriyle kıyaslanabilir. Bu yaklaşım sayesinde, önsel olarak belirlenmesi gereken σ gibi bir eşik değeri hakkında herhangi bir varsayımda bulunmadan en uygun öznitelikler kolaylıkla seçilebilir. Burada σ eşik değeri, son öznitelik veri kümesine sıfır ağırlıklı olmayan tüm öznitelikleri dahil edebilmek amacıyla 0.02 olarak belirlenmiştir.

Bu çalışmada, nihai öznitelik veri kümesi HCTSA yazılım paketinden elde edilen toplam 32 istatistiksel öznitelikten oluşmaktadır. Bu öznitelikler sırasıyla, özniteliğin yazılım paketindeki numarası, özniteliğin adı ve özniteliğin önem derecesi olacak şekilde Tablo 4.11.'de listelenmiştir. Seçilen özniteliklerin içeriği ve dahil olduğu kategori detaylı olarak incelediğinde, öznitelik grubunun çoğunluğunu olasılık dağılımına dayalı yaklaşımlar ve eğri uydurma tabanlı zaman serisi analizi yaklaşımları oluşturmaktadır. Buna ek olarak, "z-skor testi" ve "Pearson korelasyon katsayısının çarpıklığı" gibi aykırı değer analizinde yaygın olarak karşılaşılan bazı öznitelikler de, en uygun öznitelik veri kümesinde bulunmaktadır.

Tablo 4.10. SGD minimum bulma yöntemi ve çapraz geçerlilik bölütleme (Cross-validation Partition-CVP) ortalama kaybı kullanarak en uygun λ düzenlileştirme parametresinin belirlenmesi.

Deneme	λ	CVP (1)	CVP (2)	CVP (3)	CVP (4)	CVP (5)	Ortalama Kayıp
1	0	0.1819	0.2004	0.1863	0.1839	0.1858	0.1877
2	0.0001	0.1621	0.2362	0.1722	0.1352	0.1589	0.1729
3	0.0003	0.1704	0.1831	0.2119	0.1704	0.1883	0.1848
4	0.0004	0.2063	0.1914	0.2804	0.1947	0.2498	0.2245
5	0.0005	0.1409	0.1338	0.2356	0.2056	0.2300	0.1892
6	0.0007	0.2287	0.1536	0.1703	0.1909	0.2633	0.2014
7	0.0008	0.1685	0.1543	0.1633	0.1512	0.1922	0.1659
8	0.0009	0.1480	0.1236	0.1261	0.2511	0.2018	0.1701
9	0.0011	0.1922	0.1786	0.1517	0.1262	0.2633	0.1824
10	0.0012	0.1115	0.0832	0.0704	0.0609	0.2063	0.1065
11	0.0013	0.1525	0.1965	0.1376	0.2165	0.1903	0.1787
12	0.0015	0.0685	0.1498	0.1101	0.0673	0.1236	0.1039
13	0.0016	0.1409	0.1223	0.0704	0.0775	0.0769	0.0976
14	0.0018	0.0615	0.0730	0.0736	0.0775	0.0820	0.0735
15	0.0019	0.0922	0.0787	0.1338	0.1153	0.1108	0.1062
16	0.0020	0.0955	0.0621	0.0711	0.0833	0.0826	0.0789
17	0.0022	0.0647	0.0691	0.1498	0.0717	0.1102	0.0931
18	0.0023	0.0762	0.0615	0.1396	0.0807	0.0666	0.0849
19*	**0.0024**	**0.0743**	**0.0595**	**0.0602**	**0.0871**	**0.0685**	**0.0699**
20	0.0026	0.0955	0.1152	0.0762	0.0807	0.0871	0.0909

* Yerel minimum çözüm bulundu [İyileştirici: SGD, İterasyon sınırı=30 (her bir deneme için), Gradyan toleransı=1e-4, Hesse (Hessian) matrisi için geçmiş arabelleğinin boyutu=15, doğru üzerinde arama (line search) yöntemi=weak Wolfe].

Tablo 4.11. Öznitelik ağırlık katsayısı dikkate alınarak belirlenen özelliklerin açıklamaları.

No	Öznitelik Numarası	Öznitelik Adı	Öznitelik Önem Derecesi
1	412	IN-AutoMutualInfoStats-diff-20-kraskov1-4-ami1	1.6180
2	960	ztest	1.3771
3	41	skewness-pearson	1.3569
4	3	harmonic-mean	1.3003
5	4127	SC-FluctAnal-2-dfa-50-2-logi-ssr	1.1384
6	87	propUnique	0.7774
7	932	DN-FitKernelSmoothraw-max	0.7438
8	4156	SC-FluctAnal-2-dfa-50-3-logi-meanssr	0.7316
9	3887	MF-AR-arcov-5-a4	0.7218
10	426	IN-AutoMutualInfoStats-diff-20-kraskov1-4-ami15	0.7081
11	3245	DN-OutlierInclude-abs-001-nfexpa	0.6974
12	6511	WL-fBM-p2	0.6505
13	1454	SB-TransitionpAlphabet-40-1-maxdiagfexp-rmse	0.5916
14	3201	DN-CompareKSFit-ev-psy	0.5704
15	3878	MF-AR-arcov-4-a4	0.5166
16	3208	DN-CompareKSFit-uni-olapint	0.4921
17	258	CO-HistogramAMI-quantiles-5-2	0.4890
18	3870	MF-AR-arcov-3-a4	0.4744
19	7256	MF-armax-2-2-05-1-MA-1	0.4743
20	6561	MF-arfit-1-8-sbc-meanA	0.4682
21	4131	SC-FluctAnal-2-dfa-50-2-logi-ratsplitminerr	0.4613
22	3334	ST-LocalExtrema-l50-minmax	0.3873
23	622	SY-SlidingWindow-ent-ent5-10	0.3527
24	1502	SB-TransitionpAlphabet-20-ac-stdeigfexp-b	0.3525
25	3869	MF-AR-arcov-3-a3	0.3513
26	6460	WL-coeffs-db3-2-mean-coeff	0.3314
27	624	SY-SlidingWindow-ent-ent10-2	0.2712
28	2209	SY-StdNthDerChange-rmse	0.2501
29	607	SY-SlidingWindow-ent-s5-1	0.2333
30	3247	DN-OutlierInclude-abs-001-nfexpc	0.2321
31	2674	PH-Walker-momentum-2-sw-propcross	0.2101
32	1263	CO-AddNoise-1-gaussian-firstUnder25	0.0669

Tablo 4.12. DL ve Elephas kestiricisi model parametreleri.

Model	Parametre	Değer
DL	Nöron seyreltme oranı	0.3
	L2 düzenlileştirme	0.003
	Yitim fonksiyonu	kategorik çapraz entropi
	İyileştirici	Adam
Elephas	epok	100
	Toptan (Batch) büyüklüğü	64
	doğrulama oranı	0.10
	İyileştirici	Adam
	L1 düzenlileştirme	0.0003
	Yitim fonksiyonu	ikili çapraz entropi
	Çalışma modu	eşzamanlı
	Performans metriği	doğruluk

4.2.2. Derin Öğrenme Modelinin Geleneksel Sınıflandırma Modelleri ile Karşılaştırılması ve Elde Edilen Bulgular

Bu bölümde önerilen DL modelinin performansı toplam dokuz farklı sınıflandırıcının performansı ile 8 farklı değerlendirme ölçütü dikkate alınarak incelenmiştir. Burada yalnızca bir sınıflandırıcı ve türevlerini kullanmak yerine, Apache Spark çerçevesi ile entegre sınıflandırıcıların çeşitliliğini artırmak için farklı dağıtık hesaplama platformları birlikte kullanılmıştır. Her sınıflandırıcı için öznitelik veri kümesindeki örnekler rastgele olacak şekilde eğitim aşaması için %80, test aşaması için %20 olacak şekilde alt kümelere bölünmüş ve veri kümesinde varsayılan olarak kayıp-kaçak oranı %30 olarak belirlenmiştir. Ayrıca çapraz geçerleme (cross validation) yöntemini destekleyen sınıflandırıcılar (XGBoost, GBM) için sınıflandırma sonuçları 5-kat çapraz geçerleme kullanılarak doğrulanmıştır. Halihazırda herkese açık kayıp-kaçak veri kümeleri olmadığından, kayıp-kaçak tespit yöntemlerini kıyaslamak zordur [9]. Bu nedenle bu çalışmada kullanılan veri kümesi FDI tabanlı kayıp-kaçak senaryoları dikkate alınarak oluşturulmuş ve sınıflandırıcıların pratik uygulamalarına odaklanılmıştır.

Bu çalışmada önerilen DL modelinin performansı geleneksel ve en ileri makine öğrenmesi algoritmalarının performansı ile karşılaştırılmıştır. Tablo 4.12. ve 4.13.'te önerilen DL modeli ve karşılaştırma aşamasında kullanılan algoritmalarının ortam gereksinimleri ve model parametreleri verilmiştir.

Önerilen DL modelinin performans karşılaştırması Tablo 4.14.'te verilmiştir. Burada çoğu sınıflandırıcının doğruluğu %90'ın üzerindedir ve NB sınıflandırıcısı çoğu performans metriğinde en düşük performansı göstermiştir. Bunun nedeni ise NB yaklaşımını kullanan çoğu sınıflandırıcı güçlü varsayımlarda bulunarak belirli bir özniteliğin değerinin başka bir özniteliğin değerinden bağımsız olduğu ön kabulünü yapmasıdır. Ayrıca Tablo 4.14.'te, her bir kriter için en iyi performansı gösteren sınıflandırıcıların dağılım istatistikleri farklılık göstermektedir. Burada sınıflandırıcılar doğruluk, F_1 ve Kappa parametreleri dikkate alınarak sıralandığında ise önerilen DL tabanlı sınıflandırıcı model en iyi performansa sahiptir. Tablo 4.14.'te sınıflandırıcıların en yüksek performans gösterdiği kriterlerdeki

Tablo 4.13. Apache Spark ile entegre sınıflandırıcıların ortam gereksinimleri ve model parametreleri.

Dağıtımcı	Ortam Gereksinimleri	Sınıflandırıcı	Model Parametreleri
Spark MLLib	HDP 2.6.5 (Hadoop 2.7.3 +Spark 2.3.2)	SVM [161]	Ref. [182]
		LR	Ref. [183]
		RF [159]	Ref. [184]
		DT [160]	Ref. [185]
		NB	Ref. [186]
Elephas	Keras 2.2.5 +Spark 2.3.2	DL [187]	Şekil 3.17. ve Tablo 4.12.
H_2O.ai	Java 7 veya sonrası +H_2O 3.20.0.1 +Spark 2.3.1 +Sparkling Water 2.3.7	GBM [162]	Ref. [188]
		XGBoost [164]	Ref. [189]
Yandex	CatBoost 0.26 +Python 3.6.8 +Jupyter notebook	CatBoost [166]	Ref. [190]
MMLSpark	Centos 7 +Python 3.6 +Jupyter notebook +Spark 2.4.7 +Hadoop 3.1.2	LightGBM [167]	Ref. [191]

Tablo 4.14. FDI tabanlı kayıp-kaçak uygulamaları için önerilen DL modelin geleneksel ve en ileri sınıflandırıcılar ile karşılaştırılması.

Sınıflandırıcı	Performans Metriği*							
	Doğruluk (↑)	**Duyarlılık** (↑)	**Özgüllük** (↑)	**Kesinlik** (↑)	**FPR** (↓)	**F_1** (↑)	**MCC** (↑)	**Kappa** (↑)
DL	**0.973**	0.939	0.989	0.958	0.010	**0.948**	**0.940**	**0.892**
XGBoost	0.972	0.972	**0.995**	0.924	**0.005**	0.942	0.936	0.888
CatBoost	0.972	0.909	0.985	**0.984**	0.014	0.931	0.930	0.885
LightGBM	0.971	0.925	0.986	0.983	0.013	0.945	**0.940**	0.882
GBM	0.969	**0.979**	0.994	0.920	**0.005**	0.940	0.935	0.876
SVM	0.967	0.908	0.985	0.967	0.014	0.927	0.922	0.865
LR	0.965	0.918	0.985	0.961	0.014	0.931	0.925	0.858
RF	0.946	0.848	0.973	**0.984**	0.027	0.846	0.843	0.779
DT	0.943	0.860	0.975	0.924	0.024	0.884	0.870	0.768
NB	0.872	0.632	0.938	0.943	0.062	0.677	0.666	0.480

* (↑) : Daha büyük değer daha iyidir. (↓) : Daha küçük değer daha iyidir.

önemli değerler, kalın ve siyah renkli olarak vurgulanmıştır. Bu çalışmada en yüksek doğruluğa sahip ilk dört sınıflandırıcı, farklı kayıp-kaçak oranlarında sınıflandırıcıların performansını incelemek üzere Bölüm 4.2.3.'te kullanılmıştır.

Tablo 4.14.'de GBM ve XGBoost sınıflandırıcıları en düşük FPR oranlarına sahiptir ve Apache Spark çerçevesiyle H_2O.ai entegrasyonu sayesinde tüm boosting algoritmaları içerisindeki kurulumu ve kullanımı en kolay olanıdır. Ayrıca XGBoost sınıflandırıcısı 7.9 saniye ile boosting yöntemleri arasında en hızlı eğitim süresine sahiptir. Burada diğer üç sınıflandırıcının eğitim süresi sırasıyla 13.4 saniye ile CatBoost, 19.6 saniye ile LightGBM ve 100 epok için 100.5 saniye ile önerilen DL modelidir. Apache Spark çerçevesi ile entegrasyon için en fazla ek yapılandırma adımlarını gerektiren sınıflandırıcı, LightGBM, dağıtım şirketlerinin günlük operasyonel faaliyetleri için kullanılabilecek ikinci umut verici çözümdür. Tablo 4.14.'de, LightGBM'nin varsayılan parametreleri dengesiz kayıp-kaçak veri kümeleri için kabul edilebilir ve rekabetçi sonuçlar üretmiştir. Ayrıca, CatBoost

Tablo 4.15. Farklı kayıp-kaçak oranları için önerilen DL modeli ile boosting tabanlı sınıflandırıcılarının performansı.

Kayıp Oranı	Model	Performans Metriği*							
		Doğruluk (↑)	**Duyarlılık** (↑)	**Özgüllük** (↑)	**Kesinlik** (↑)	**FPR** (↓)	**F_1** (↑)	**MCC** (↑)	**Kappa** (↑)
<%3	DL	0.995	0.836	**0.981**	0.821	**0.019**	NaN	0.815	0.982
	XGBoost	**0.996**	0.816	**0.981**	0.856	**0.019**	NaN	0.825	**0.985**
	CatBoost	0.991	0.724	0.974	**0.880**	0.026	0.756	0.761	0.964
	LightGBM	0.994	**0.855**	0.974	0.979	0.026	**0.863**	**0.869**	0.975
%3-5	DL	**0.990**	0.819	**0.980**	0.844	**0.019**	0.828	0.815	0.962
	XGBoost	0.989	**0.877**	0.976	0.998	0.024	**0.899**	**0.903**	0.958
	CatBoost	0.987	0.817	0.960	0.954	0.039	0.824	0.821	0.947
	LightGBM	0.993	0.870	0.974	**0.999**	0.025	0.890	0.894	**0.970**
%5-10	DL	0.984	0.881	0.981	0.938	0.018	0.903	0.894	0.935
	XGBoost	0.990	0.874	0.985	**0.993**	0.015	0.890	0.900	0.960
	CatBoost	**0.994**	**0.925**	**0.991**	0.991	**0.009**	**0.948**	**0.948**	**0.978**
	LightGBM	0.988	0.882	0.984	0.952	0.016	0.903	0.900	0.951
%10-20	DL	**0.981**	**0.898**	**0.986**	0.956	**0.014**	**0.922**	0.915	**0.924**
	XGBoost	0.976	0.879	0.983	0.952	0.016	0.898	0.896	0.905
	CatBoost	0.977	0.863	0.979	0.983	0.021	0.882	0.887	0.905
	LightGBM	0.975	0.896	0.980	**0.984**	0.019	0.918	**0.917**	0.901
%20-30	DL	**0.973**	0.939	0.989	0.958	0.010	**0.948**	**0.940**	**0.892**
	XGBoost	0.972	**0.972**	**0.994**	0.924	**0.005**	0.942	0.936	0.888
	CatBoost	0.971	0.909	0.985	**0.984**	0.014	0.931	0.930	0.885
	LightGBM	0.971	0.925	0.986	**0.984**	0.013	0.945	**0.940**	0.882

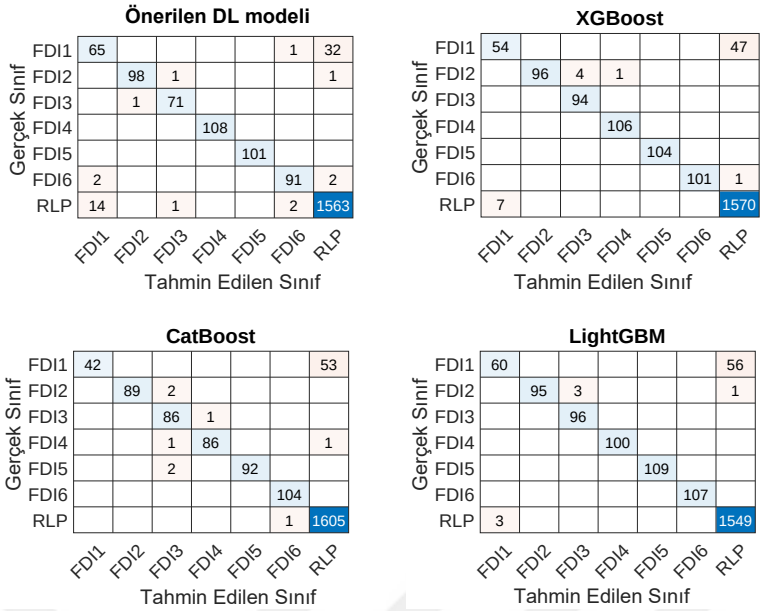
* (↑) : Daha büyük değer daha iyidir. (↓) : Daha küçük değer daha iyidir.

sınıflandırıcısı, duyarlılık parametresi dışında LightGBM sınıflandırıcısına yakın sonuçlar üretmiştir. Son olarak, GBM ve RF sınıflandırıcıları sırasıyla 0.979 ve 0.984 ile duyarlılık ve kesinlik metriklerinde iyi performans göstermiştir.

4.2.3. Kayıp-Kaçak Veri Kümesindeki Sınıf Dengesizliği ve Değerlendirme Ölçütlerinin Yorumlanması

Kayıp-kaçak veri kümelerindeki sınıf dengesizliği, her sınıf için çarpık bir olasılık dağılımına neden olduğundan sınıflandırıcılar azınlıkta olan grupları ana sınıfa dahil etme eğilimindedir. Ayrıca, değerlendirme metrikleri sınıflandırıcıların yanıltıcı sonuçlarını doğru bir şekilde göstermeyebilir ve bu da sonuçların farklı yorumlanmasına yol açmaktadır. Farklı kayıp-kaçak oranları için seçilen dört sınıflandırıcının sonuçları Tablo 4.15.'te verilmiştir. Burada kayıp-kaçak oranı, değiştirilmiş yük profillerinin sayısının, veri kümelerindeki toplam yük profili sayısına oranını ifade etmektedir. Bu çalışmada kullanılan veri kümelerindeki kayıp-kaçak oranları (<%3, %3-5, %5-10, %10-20, %20-30), veri kümelerine sonradan dahil edilen ve rastgele seçilen fazladan 210, 390, 750, 1500 ve 3000 değiştirilmiş yük profilini temsil etmektedir. Kayıp-kaçak veri kümelerinde yer alan gerçek yük profili (Real Load Profile-RLP) sayısı ise 7807'dir.

Tablo 4.15.'te gösterildiği üzere, tüm kayıp-kaçak oranları için tercih edilebilecek tek bir nihai sınıflandırıcı bulunmamaktadır. Ancak, CatBoost ve DL sınıflandırıcıları genel olarak sırasıyla %5-10 ve %10-20 NTL oranları için iyi performans göstermiştir. Ayrıca, doğruluk metriği seçilen dört sınıflandırıcının farklı kayıp-kaçak oranlarındaki performansını doğru bir şekilde yansıtmamaktadır. Bu olgu, sınıf dengesizliğinin olduğu kayıp-kaçak uygulamalarının yeterince ele alınmamış bir sorunudur [9]. Böyle bir



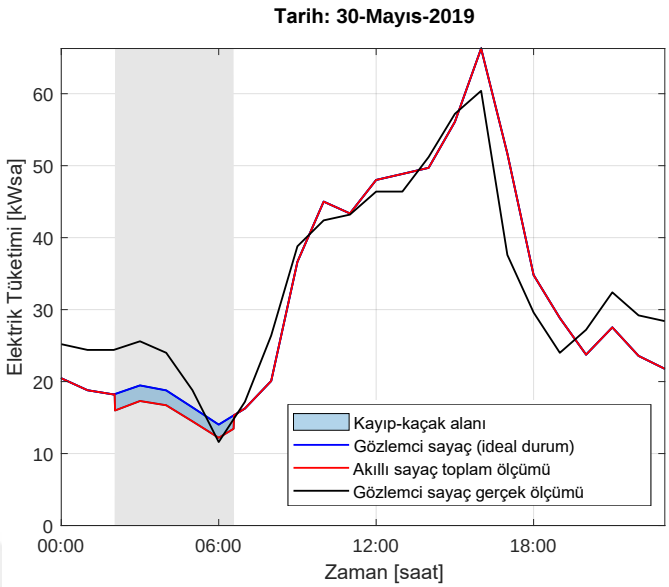
Şekil 4.10. Önerilen DL modelinin ve seçilen üç sınıflandırıcının %20-30 kayıp-kaçak oranındaki hata matrisleri [181].

durumda, FDI örneklerinin RLP sınıfına dahil olup olmadığını anlamak için FPR metriği kullanılabilir. Bu çalışmada kayıp-kaçak oranı azaldıkça, test veri kümelerinde daha fazla FDI örneği RLP olarak etiketlendiğinden FPR değerinin arttığı gözlenmiştir.

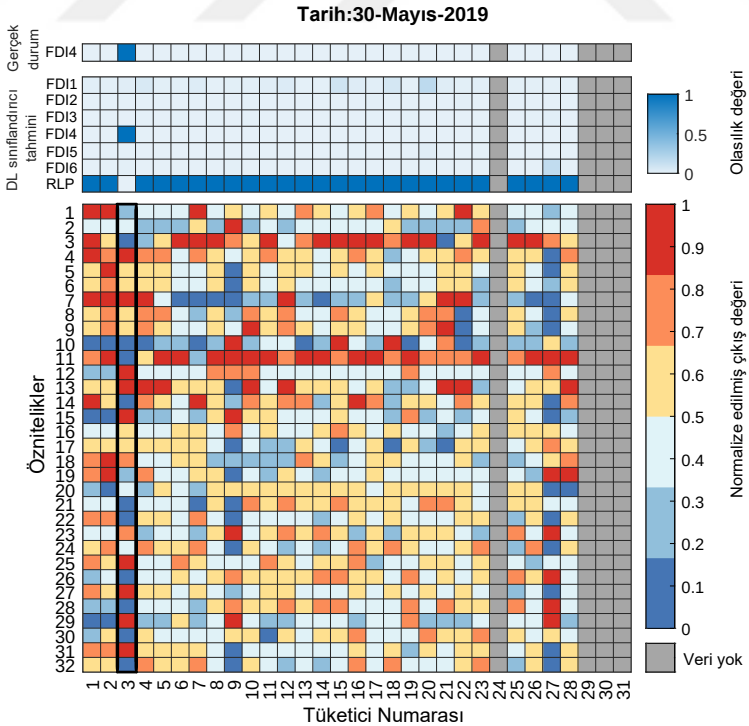
Önerilen DL modeli ile boosting tabanlı sınıflandırıcıların %20-30% kayıp-kaçak oranındaki hata matrisi Şekil 4.10.'da verilmiştir. Burada dört sınıflandırıcının tümünün FDI1 örneklerinin çoğunu yanlış bir şekilde RLP olarak etiketlemesinin bir nedeni, değiştirilmiş yük profillerinin de daha düşük bir tüketim seviyesinde RLP'ye benzemesidir.

Bu çalışmada sınıf dengesizliği problemi günlük olarak trafo seviyesinde de detaylı olarak incelenmiştir. Bu senaryoda, 2019 yılı için rastgele 10 gün seçilmiş ve ardından rastgele seçilen herhangi bir tüketicinin karşılık gelen gününe rastgele seçilmiş bir FDI saldırısı uygulanmıştır. Veri kümesinde, seçilen 10 gün için toplam 10 adet FDI yük profili ve 266 adet RLP bulunmaktadır. Bu uygulamadaki veri kümesinde sınıf dağılımı oldukça dengesizdir (highly imbalanced) ve trafo seviyesinde her gün için yalnızca bir FDI yük profili bulunmaktadır. Bu örnekler önerilen DL modelinin eğitim ve test veri kümelerinden tamamen çıkarılarak ayrı şekilde test edilmiştir. Daha sonra bu oldukça dengesiz veri kümesi üzerinde DL sınıflandırıcının performansı incelenmiştir. Önerilen DL sınıflandırıcısı, toplam 276 örnekten biri FDI ve biri RLP olan toplam 2 örneği yanlış sınıflandırmıştır. Trafo seviyesinde gerçekleştirilen kayıp-kaçak uygulaması Şekil 4.11. ve Şekil 4.12.'de gösterilmektedir.

Şekil 4.12.'de, her bir tüketici bir sayı ile temsil edilmiş ve sınıflandırma aşamasında kullanılan istatistiksel öznitelikler, daha önce Bölüm 4.2.1.'de açıklanan en uygun özniteliklerdir. Şekil 4.12.'de gösterildiği üzere, siyah dikdörtgen kutu içerisine alınan



Şekil 4.11. FDI4 tabanlı kayıp-kaçak senaryosunda gözlemci sayaç ve akıllı sayaç toplam ölçümü [30 Mayıs 2019] [181].



Şekil 4.12. Trafo merkezi seviyesinde oldukça dengesiz bir FDI4 tabanlı kayıp-kaçak senaryosu ve önerilen DL sınıflandırıcısının tahmini [181].

tüketicinin yük profili, FDI4 tabanlı bir kayıp-kaçak senaryosunu temsil etmektedir ve kalan diğer tüketicilerin yük profilleri ise normaldir. Trafo seviyesinde ölçüm alan gözlemci sayaç ile FDI4 tabanlı kayıp-kaçak uygulamasının neden olduğu akıllı sayaç toplam ölçümü arasındaki fark, Şekil 4.11.'de verilmiştir.

4.2.4. Kayıp-Kaçak Veri Kümesinin Veri Kalitesi, Yanlılık ve Ortak Değişken Kayması Sorunlarının İncelenmesi

Kayıp-kaçak tespiti uygulamalarındaki veri kalitesi sorunu, genellikle yanlış etiketlenmiş veya eksik kaydedilmiş durum bilgisinden kaynaklanmaktadır. Bu çalışmada tüketiciyi tanımlayan statik sayaç verileri (örneğin; sayaç tipi, bağlantı tipi, tarife kategorisi) kullanılmamıştır. Bunun yerine, Diyarbakır ilinde belirlenen bir trafo istasyonu ve bu trafo istasyonuna bağlı tüketicileri içeren bir pilot bölgedeki saatlik akıllı sayaç ölçümleri kullanılmıştır. Bu pilot alanda, rastgele seçilmiş tüketicilerin akıllı sayaç verisine farklı FDI saldırı senaryoları uygulanmıştır.

Akıllı sayaç ölçümleri ile gözlem sayaç ölçümleri arasındaki zaman senkronizasyonu sorunu ve bu ölçümler arasındaki tutarsız boşluklar, şehrin dağıtım hattı seviyesindeki altyapı sorunlarından kaynaklanmaktadır. Bu eksikliklerin ortadan kaldırıldığı varsayıldığında, önerilen sağlam istatistiksel öznitelikler ve Büyük Veri analitiği ile entegre DL modeli, şehir düzeyinde kullanılabilecek yaygın bir uygulama potansiyeline sahiptir.

Veri kalitesi sorunu aynı zamanda belirli bir zaman aralığında kaydedilen kullanılabilir veri miktarıyla da ilişkilidir. Kayıp-kaçak veri kümesinde, veri kullanılabilirliği 2019 için yaklaşık %70'dir ve trafo istasyonuna bağlı 31 tüketiciden 3 tanesi çok düşük tüketim seviyelerine sahip olduğundan istatistiksel öznitelik çıkarma ve sınıflandırma işlemlerine dahil edilmemiştir.

Ortak değişken kayması, farklı dağılımlara sahip eğitim ve test veri kümelerinde meydana gelen örnekleme yanlılığını ifade etmektedir. [192], [193]. Bu çalışmada, temel FDI tabanlı kayıp-kaçak senaryolarının matematiksel formülleri belli olduğu için veri kümesinde yeterli FDI örnekleri oluşturulabilmiştir. Ayrıca bu çalışmada, koşullu GAN gibi üretken modeller kullanarak da bu örnekleri oluşturmak mümkündür [194].

5. Sonuçlar

Son yıllarda, akıllı şebeke kavramının hayatımıza girmesiyle birlikte elektrik enerjisinin iletimi ve dağıtımı alanında büyük teknolojik gelişmeler meydana gelmiştir. Bu gelişmelerle birlikte tasarımcılar, aynı zamanda birçok zorlukla da karşı karşıya kalmış ve bu zorlukların üstesinden gelmek için yeni araçlar ve yaklaşımlar ortaya koymuştur. Akıllı şebekede AMI sistemler bu yeni teknolojik araçlara örnek olarak gösterilebilir. Günümüzde elektronik, otomasyon, iletişim ve veri işleme alanlarındaki yenilikler ve gelişmeler sayesinde tüketicilerden gerçek zamanlı olarak veri temin edebilen, verileri iletebilen ve güç akışını yük merkezlerinden iletilen geri bildirimleri dikkate alarak belirleyebilen bir altyapı hayata geçirilmektedir. Bu sayede merkeziyetsiz, yenilenebilir enerji kaynaklarını kullanan ve enerji verimliliğini öncelikli bir konumda değerlendiren iletim ve dağıtım sistemlerinin geliştirilmesi mümkün olabilmektedir.

AMI sistemler özellikle, planlama ve performans optimizasyonu amacıyla DSO'lar veya kamu kuruluşları için şebekenin durumu hakkında ilk elden bilgiye sahip olmalarına katkı sağlamaktadır. Ayrıca AMI sistemlerden elde edilen veriler, hem tüketici hem de sağlayıcı tarafında tüketimin düzenlenmesine yardımcı olmaktadır. AMI tarafından sağlanan teşhis ve geri bildirim araçları ise, gelecek dönem yük planlaması, tüketici segmentasyonu, siber veya fiziksel saldırıların tespiti gibi yüzlerce uygulama sayesinde milyonlarca lira tasarruf sağlayabilir.

Bu tez çalışmasında, AMI sistemler için kısa dönemli yük tahmini ve FDI tabanlı kayıp-kaçak tespiti uygulaması gerçekleştirilerek Büyük Veri teknolojilerinin uygulanabilirliği araştırılmıştır. Bu kapsamda geliştirilen tahmin ve sınıflandırıcı modellerinin performansı gerçek akıllı sayaç veri kümeleri üzerinde test edilmiştir. Tez kapsamında geliştirilen tahmin ve sınıflandırıcı modellerden elde edilen sonuçlar iki alt başlıkta aşağıda özetlenmiştir.

Kısa dönemli yük tahmini için önerilen hibrit DL modelinden elde edilen bulgular: Son yıllarda, tüketici yük profillerinin kullanıldığı kısa dönemli yük tahmini, akıllı şebekede kararlı, güvenilir ve sürdürülebilir bir sistem tasarımı oluşturmak için oldukça sık tercih edilen bir uygulama haline gelmiştir. Bu tez çalışmasında geliştirilen CNN-BLSTM tabanlı hibrit DL modeli tüketici seviyesinde kısa dönemli tahmin sonuçları üretmektedir. Önerilen kısa dönemli yük tahmin modeli gelişmiş veri ön işleme yöntemleri ile CNN-BLSTM modelinin birlikte kullanıldığı bir yapıda tasarlanmıştır. Veri ön işleme aşamasında yük profillerine yoğunluk tabanlı aykırılık analizi uygulanarak tahmin sonuçları üzerindeki etkisi incelenmiştir. Bu çalışmada geliştirilen hibrit DL modeli hem gerçek hem de halka açık akıllı sayaç veri kümelerine uygulanmıştır. Yoğunluk tabanlı aykırılık analizi aşamasında elde edilen sonuçlar aşağıda maddeler halinde açıklanmıştır:

- Metrik olmayan MDS algoritmasından DEPSAŞ ve SGSC veri kümeleri için elde edilen ortalama stres değeri sırasıyla 0.2249 ve 0.2742'dir. Bu değerlerden tüketicilerin yük profillerinin MDS koordinasyonlarının dağınık bir öznitelik uzayını temsil ettiği

sonucu çıkarılabilir.

- Her iki akıllı sayaç veri kümesi için DBSCAN algoritmasının *epsilon* ve *minpts* parametreleri optimize edilerek sonuçlar detaylı olarak Bölüm 4.1.2.'de açıklanmıştır. Burada, en uygun *minpts* değeri SGSC veri kümesinin % 93.44'ü için 1 veya 2'dir ve *epsilon* değeri Ek-3'de yer alan Şekil 5.1.'deki karar ağacı yapısı kullanılarak belirlenebilmektedir.

- DBSCAN parametre optimizasyonu sonrasında elde edilen RMSSTD kümeleme indeks değeri DEPSAŞ ve SGSC veri kümeleri için ortalama 0.2338 ve 0.3678 olarak hesaplanmıştır. Bu çalışmada RMSSTD indeksi aynı zamanda bir maliyet fonksiyonu olarak da kullanıldığı için geleneksel yaklaşımda kullanılan *dirsek noktası*'nın (elbow point) belirlenmesi optimizasyon aşamasında zorluklara neden olmaktadır. Ayrıca, her bir tüketici için gerçekleştirilen optimizasyon işlemlerindeki her bir epok'tan elde edilen RMSSTD indeks değeri artan değerden azalan değere doğru sıralansa dahi, çoğu durumda geleneksel yaklaşımdakine benzer belirgin bir *dirsek noktası* belirlenememiştir.

- Yoğunluk tabanlı aykırılık analizi işlemi sonrasında elde edilen ortalama küme sayısı DEPSAŞ ve SGSC veri kümeleri için sırasıyla 15 ve 20'dir. Her iki veri kümesinden elde edilen maksimum küme sayısı ise 41 ve 66 olarak hesaplanmıştır.

- DEPSAŞ veri kümesinde olağan dışı günlük yük profili olarak etiketlenen örneklerin yüzdesi % 8 ile % 12 arasında değişmektedir. SGSC veri kümesinde yer alan tüketicilerin ise yaklaşık % 41.39'luk kısmı için ortalama aykırı değer oranı % 11 ile % 13 arasında değişmektedir. SGSC veri kümesinde geriye kalan % 58.61'lik kısım için elde edilen aykırı değer oranı ise % 10'dan daha küçüktür.

Kısa dönemli yük tahmini için hibrit DL modelinden elde edilen sonuçlar ve performans değerlendirmesi aşağıda maddeler halinde açıklanmıştır:

- Önerilen hibrit DL modeli DEPSAŞ veri kümesi için ortalama % 36.748'lik bir MAPE değeriyle tüketici seviyesinde kısa dönemli yük tahmini gerçekleştirmektedir. Tahmin performansı karşılaştırmasında kullanılan RMSE, NRMSE VE MAE metrikleri için elde edilen ortalama değerler ise sırasıyla 0.108, % 55.432, 0.058'dir.

- Önerilen hibrit DL modeli SGSC veri kümesi için ise ortalama % 33.477'lik bir MAPE değeriyle tahmin sonucu üretmiştir. Tahmin performansı karşılaştırmasında kullanılan RMSE, NRMSE, MAE metrikleri için elde edilen ortalama değerler ise sırasıyla 0.114, % 46.234, 0.074'dir.

- Bu tez çalışmasında her iki veri kümesi için veri kullanılabilirliği dikkate alınarak iki farklı senaryo analiz edilmiştir. DEPSAŞ veri kümesi için belirlenen en uygun aralık 1 Nisan 2018 ile 31 Ağustos 2018, SGSC veri kümesi için ise 23-31 Ağustos 2013

tarihleri arasında kalan zaman dilimidir. Bu tarih aralıklarında hibrit DL modeli kullanılarak DEPSAŞ veri kümesinde yer alan mesken tüketicileri için % 33.736, ticarethane tüketicileri için ise % 27.374'lik bir MAPE değeri elde edilmiştir. SGSC veri kümesi için hesaplanan MAPE değeri ise % 31.481'dir.

- Aykırılık analizinin tahmin modeline dahil edilmesi eğitim, doğrulama ve test aşamasında elde edilen hata değerlerinin azaltmıştır. DEPSAŞ veri kümesi için test aşamasında hesaplanan NRMSE ve MAPE değerlerinde sırasıyla % 7 ve % 5'lik bir azalma, SGSC veri kümesinde ise % 5 ve % 3'lük bir azalma gözlenmiştir.

- Önerilen hibrit DL modeli, gerçek veri kümeleri üzerinde uygulanabilir çözümler üretebilen ve uygulamaya bağlı olarak yeniden düzenlenebilen esnek bir çerçeveye sahiptir. Ayrıca, tam veri temininin zor olduğu veya veri gizliliği nedeniyle daha az özniteliğin kullanılmasını gerektiren durumlar için makul bir seçenektir.

- Bu çalışmada önerilen hibrit DL modeli literatür dikkate alınarak farklı kriterlere göre değerlendirilmiştir. Önerilen model, geleneksel tahmin modellerinden daha düşük hata oranlarıyla kısa dönemli yük tahmini gerçekleştirebilmektedir. Elde edilen sonuçlar dikkate alındığında, önerilen yöntem, DEPSAŞ veri kümesinin %72'si için ve SGSC veri kümesinin %69.78'lik kısmı için MAPE değerlerini farklı oranlarda azaltmıştır.

- Önerilen hibrit DL modeli CNN-BLSTM ile entegre çalışan veri ön işleme yöntemleri, sağlam (robust) yük tahmin modelleri oluşturabilmek amacıyla birlikte kullanılmaktadır. Önerilen çerçeve, akıllı şehir veya akıllı şebeke uygulamalarında ağ sınırında bilişim ya da MapReduce gibi dağıtık hesaplama yaklaşımlar kullanılarak ölçeklendirilebilir.

Kayıp-kaçak sınıflandırma uygulaması için önerilen DL modelinden elde edilen bulgular: Son yıllarda, Dünya çapında elektrik hırsızlığının artmasıyla birlikte akıllı şebekede kayıp-kaçak tespiti önemli bir konu haline gelmiştir. Akıllı şebekede hem fiziksel hem de siber girişimler yoluyla yasadışı olarak elektrik tüketimine yönelik birçok tehdit bulunmaktadır. Bu tez çalışmasında, AMI sistemlerin siber saldırılar yoluyla manipüle edildiği FDI tabanlı kayıp-kaçak uygulamaları incelenmiştir. Ayrıca, istatistiksel öznitelik çıkarımı ve sınıflandırma işlemleri için Büyük Veri analitiği ile entegre yeni bir çerçeve önerilmiştir.

Önerilen çerçeve, kayıp-kaçak tespiti için kullanılabilecek sağlam istatistiksel öznitelik çıkarımını modelleyebilmek için çeşitli makine öğrenmesi, DL ve paralel hesaplama kütüphanelerinden yararlanmaktadır. Bu tez çalışmasından elde edilen istatistiksel öznitelikler, fiziksel sayaç müdehalelerine dayalı kayıp-kaçak tespit uygulamalarında da kullanılabilir. Öznitelik öğrenme aşamasında, HCTSA yazılım paketi ve komşuluk bileşen analizi öznitelik seçme algoritması birlikte kullanılarak iki ardışık veri ön işleme süreci birleştirilmiştir. Ayrıca, Apache Spark çerçevesinin dönüşüm ve eylem

özelliklerinden de yararlanılmaktadır. Son aşamada ise, tez çalışması kapsamında önerilen istatistiksel öznitelikler sınıflandırma uygulamasında kullanılmıştır.

Kayıp-kaçak tahmini için önerilen DL modelinden ve performans karşılaştırmasında kullanılan makine öğrenmesi yöntemlerinden edilen sonuçlar ve genel değerlendirmeler aşağıda maddeler halinde açıklanmıştır:

- Kayıp-kaçak sınıflandırma uygulamasında kullanılan sınıflandırıcıların çoğunluğunun doğruluğu % 90 'ın üzerindedir.
- Önerilen DL modeli, doğruluk, F_1 ve Kappa parametreleri dikkate alınarak sıralandığında en iyi performansı göstermiştir. Ayrıca, DL modeli Apache Spark tabanlı dağıtık hesaplama seçeneği kullanılarak eğitilmiştir. Bu işlem özellikle akıllı şehir uygulamalarında birçok parametrenin dikkate alındığı durumlar için oldukça kritir bir aşamadır.
- GBM ve XGBoost sınıflandırıcı modelleri ise FPR metriği dikkate alındığında en iyi performansı göstermişlerdir. Ayrıca, Apache Spark çerçevesiyle H_2O.ai entegrasyonu sayesinde tüm boosting algoritmaları içerisindeki kurulumu ve kullanımı en kolay olanıdır.
- Bu çalışmada en yüksek doğruluğa sahip ilk dört sınıflandırıcı model, farklı kayıp-kaçak oranlarında test edilmek üzere seçilmiştir. Karşılaştırma aşamasında kullanılan sınıflandırıcı modeller ise sırasıyla önerilen DL modeli, XGBoost, CatBoost ve LightGBM'dir.
- XGBoost sınıflandırıcı model, 7.9 saniye ile boosting yöntemleri arasındaki en düşük eğitim süresine sahiptir. Burada diğer üç sınıflandırıcının eğitim süresi sırasıyla 13.4 saniye ile CatBoost, 19.6 saniye ile LightGBM ve 100 epok için 100.5 saniye ile önerilen DL modelidir.
- Apache Spark çerçevesi ile entegrasyon aşamasında en fazla ek yapılandırma ayarları gerektiren sınıflandırıcı LightGBM algoritması olmuştur. LightGBM algoritması, elde edilen sonuçlar dikkate alındığında dağıtım şirketlerinin günlük operasyonel faaliyetleri için kullanılabilme potansiyeline sahiptir. LightGBM algoritmasının varsayılan parametreleri sınıf dengesizliğinin yüksek olduğu kayıp-kaçak senaryoları için kabul edilebilir ve rekabetçi sonuçlar üretmiştir.
- CatBoost sınıflandırıcısı, duyarlılık parametresi dışında LightGBM sınıflandırıcısına yakın sonuçlar üretmiştir. GBM ve RF sınıflandırıcıları sırasıyla 0.979 ve 0.984 ile duyarlılık ve kesinlik performans metriklerinde iyi performans göstermiştir.
- Kayıp-kaçak veri kümelerindeki sınıf dengesizliği, her sınıfa ait örnekler için çarpık bir olasılıksal dağılıma neden olmaktadır. Bu durumda ise performans değerlendirme metrikleri, sınıflandırıcıların azınlıkta olan sınıfları ana sınıfa dahil etme eğilimini doğru bir şekilde ifade edememektedir.

- Bu tez çalışmasında kayıp-kaçak veri kümelerindeki sınıf dengesizliği problemleri ayrıntılı bir şekilde incelenmiştir. Bu çalışmada kullanılan veri kümelerindeki kayıp-kaçak oranları (<%3, %3-5, %5-10, %10-20, %20-30), gerçek hayatta karşılaşılan durumlar dikkate alınarak oluşturulmuştur.

- Önerilen DL modeli tüm kayıp-kaçak oranları için en iyi performansı göstermese de, genel olarak %10-20 ve %20-30 kayıp-kaçak oranları için performans metriklerinde en iyi sonucu üretmiştir.

- Bu çalışmada doğruluk metriği, önerilen DL modeli dahil seçilen dört sınıflandırıcının farklı kayıp-kaçak oranlarındaki performansını doğru bir şekilde yansıtmamaktadır. Böyle bir durumda, kayıp-kaçak senaryolarından elde edilen örneklerinin gerçek durum sınıfına dahil olup olmadığını anlamak için FPR metriği kullanılabilir.

- Bu çalışmada sınıf dengesizliği problemi günlük olarak trafo seviyesinde de detaylı olarak incelenmiştir. Önerilen DL modeli, yüksek sınıf dengesizliğinin bulunduğu kayıp-kaçak uygulamalarında kararlı ve hızlı sonuçlar üretebilmektedir. Yüksek sınıf dengesizliğine sahip durum senaryolarında elde edilen istatistiksel öznitelikler genel modelinin eğitim ve doğrulama aşamalarında meydana gelen aşırı uyumlama probleminin çözümüne katkı sağlamaktadır.

Bu çalışmada tüketiciyi tanımlayan statik sayaç verileri (örneğin; sayaç tipi, bağlantı tipi, tarife kategorisi) yerine, Diyarbakır ilinde belirlenen bir trafo istasyonu ve bu trafo istasyonuna bağlı tüketicileri içeren bir pilot bölgedeki saatlik akıllı sayaç ölçümleri kullanılmıştır. Belirlenen pilot bölgede, rastgele seçilmiş tüketicilerin akıllı sayaç verisine farklı FDI saldırı senaryoları uygulanmıştır. Önerilen sağlam istatistiksel öznitelikler ve Büyük Veri analitiği ile entegre DL modeli, akıllı sayaç ölçümleri ile gözlem sayaç ölçümleri arasında zaman senkronizasyonu problemi ortadan kaldırıldığında, şehir düzeyinde kullanılabilecek yaygın bir uygulama potansiyeline sahiptir. Ayrıca Apache Spark'ın yönlendirilmiş döngüsel olmayan grafik yapısı, sunulan işlerin yürütme planını optimize ederken, bağımsız uygulamalara kıyasla hem öğrenme sürecini hem de en uygun çözümlere yakınsamayı hızlandırdığı gözlenmiştir. Büyük Veri ile entegre DL yaklaşımı, hesaplama gereksinimleri ve hesaplama yükünün paylaştırılması açısından daha zorlu olsa da, en iyi geleneksel makine öğrenmesi modellerinden önemli ölçüde daha iyi performans göstermiştir.

Gelecek Çalışmalar ve Öneriler

Bu tez çalışmasından elde edilen bulgular dikkate alınarak gelecek çalışmalar için gerçekleştirilmesi planlanan uygulamalara ait bilgiler aşağıda listelenmiştir:

- Bu tez çalışmasında geliştirilen kısa dönemli yük tahmin modeli hem halka açık akıllı sayaç veri kümesinde hem de yerel bir dağıtım şirketinden temin edilen akıllı sayaç veri kümesinde test edilmiştir. Her ne kadar halka açık veri kümeleri geliştirilen tahmin modellerinin performans karşılaştırmasında daha nesnel bir kıyaslama imkanı sağlasa da, elektrik tüketim davranışları coğrafi konumla, meteorolojik parametrelerle ve toplumsal tüketim davranışlarıyla ilişkili olduğundan ülkemizde de halka açık akıllı sayaç veri kümelerinin paylaşılması bölgesel yük tahmin uygulamaları için önem arz etmektedir.

- Önerilen yoğunluk tabanlı aykırılık analizi ile kısa dönemli yük tahmini uygulamasının tüketiciler için GUI veya açık kaynak kodunun oluşturulması hedeflenmektedir. Ayrıca, önerilen hibrit derin öğrenme modelinin akıllı sayaç donanım ekipmanları ve ağ sınırında bilişim teknolojileri kullanılarak dağıtık hesaplama paradigmasıyla entegre olmasının sağlanması hedeflenmektedir.

- Akıllı sayaç veri kümelerinin kullanıldığı yük tahmini uygulamalarında ölçüm alınamayan durumlarla sıklıkla karşılaşılmaktadır. Geleneksel veri ön işleme sürecinde bu durum ya eksik verilerin olduğu örneklerin veri kümelerinden tamamen çıkarılmasıyla ya da veri yakıştırma yaklaşımlarının kullanılmasıyla çözümlenmektedir. Bu durum dikkate alınarak, gelecekteki uygulamalarda eksik verinin olduğu senaryolarda da eğitimini sürdürebilen hibrit derin öğrenme modellerinin geliştirilmesi planlanmaktadır.

- Kayıp-kaçak uygulamasında geliştirilen derin öğrenme tabanlı sınıflandırıcı, seçilen bir trafo merkezine ve bu trafo merkezine bağlı tüketicilerin akıllı sayaç veri kümelerinde test edilmiştir. Gelecekteki uygulamalarda farklı sayaç müdehale uygulamalarının şehir dağıtım hattı seviyesindeki etkilerinin incelenmesi planlanmaktadır. Aynı zamanda, kayıp-kaçak tespitinin etkin ve hızlı bir şekilde tespit edilebilmesi amacıyla AMI sistemlerle entegre bir erken uyarı sisteminin geliştirilmesi hedeflenmektedir.

- Kayıp-kaçak tespitinde kullanılabilecek istatistiksel özniteliklerin araştırılması ve Büyük Veri analitiği ile öznitelik belirleme sürecinin tasarımına devam edilecektir. Bu çalışmada, öznitelik çıkarma ve seçme sürecinde meydana gelen gecikmenin çoğu farklı programlama dilleri ve ek arayüzler arasındaki iletişimden kaynaklanmaktadır. Bu yüzden farklı programlama dilleri arasındaki bu gecikmeyi ve karmaşıklığı azaltmak amacıyla, Spark SQL kullanıcı tanımlı fonksiyonlar (User Defined Functions-UDF) kullanılarak ortadan kaldırılması planlanmaktadır.

Kaynaklar

[1] Paris Agreement, PARIS2015 UN Climate Change Conference COP21-CMP11, https://ec.europa.eu/clima/eu-action/international-action-climate-change/climate-negotiations/paris-agreement_en, Erişim: 11-4-2022

[2] European Commission and Directorate-General for Energy, Alaton, C., Tounquet, F. (2020). Benchmarking smart metering deployment in the EU-28 : final report, *Publications Office*

[3] Berg Insight, Smart Metering in North America and Asia-Pacific, https://www.berginsight.com/smart-metering-in-north-america-and-asia-pacific, Erişim: 13-4-2022

[4] Hayn, M., Bertsch V., Fichtner W. (2014). Electricity load profiles in Europe: The importance of household segmentation, *Energy Research & Social Science*, C. 3, 2214-6296, doi:10.1016/j.erss.2014.07.002

[5] McLoughlin, F., Duffy A., Conlon M. (2015). A clustering approach to domestic electricity load profile characterisation using smart metering data, *Applied Energy*, C. 141, 190-199, doi:10.1016/j.apenergy.2014.12.039

[6] Zhang, Y., Huang, T., Bompard E. F. (2018). Big data analytics in smart grids: a review, *Energy Informatics*, C. 1, Sayı 1, 2520-8942, doi:10.1186/S42162-018-0007-5

[7] Kwac, J., Rajagopal, R. (2013). Demand response targeting using big data analytics, *in 2013 IEEE International Conference on Big Data*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Silicon Valley, CA, USA, 683-690

[8] Tascikaraoglu, A. (2018). Chapter 11 - On Data-Driven Approaches for Demand Response, *Big Data Application in Power Systems*, Elsevier, 243-259, doi:10.1016/B978-0-12-811968-6.00011-5

[9] Glauner, P., Meira, J. A., Valtchev, P., Bettinger, F. (2017). The Challenge of Non-Technical Loss Detection Using Artificial Intelligence: A Survey, *International Journal of Computational Intelligence Systems*, C. 10, Sayı 1, 760-775, doi:10.2991/ijcis.2017.10.1.51

[10] Tanveer A., Huanxin C., Jiangyu W., Yabin G. (2018). Review of various modeling techniques for the detection of electricity theft in smart grid environment, *Renewable and Sustainable Energy Reviews*, C. 82, Sayı 3, 2916-2933, doi:10.1016/j.rser.2017.10.040

[11] Ramos, C. C. O., Rodrigues, D., de Souza, A. N., Papa, J. P. (2018). On the Study of Commercial Losses in Brazil: A Binary Black Hole Algorithm for Theft Characterization, *IEEE Transactions on Smart Grid*, C. 9, Sayı 2, 676-683, doi:10.1109/TSG.2016.2560801

[12] Nagi, J., Yap, K. S., Nagi, F., Tiong, S. K., Koh, S. P., Ahmed, S. K. (2010). NTL detection of electricity theft and abnormalities for large power consumers In TNB Malaysia, *in 2010 IEEE Student Conference on Research and Development (SCOReD)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Kuala Lumpur, Malaysia, 202-206

[13] Türkiye Elektrik Dağıtım A.Ş. Genel Müdürlüğü-Strateji Geliştirme Dairesi Başkanlığı (2021). 2020 Yılı Türkiye Elektrik Dağıtımı Sektör Raporu, https://www.tedas.gov.tr/sx.web.docs/tedas/docs/Stratejikplan/2020_Yili_Turkiye_Elektrik_Dagitimi_Sektor_Raporu.pdf, Erişim: 14-4-2022

[14] Wang, Y., Chen, Q., Kang, C. (2020). *Smart Meter Data Analytics: Electricity Consumer Behavior Modeling, Aggregation, and Forecasting*, Springer Nature, Beijing

[15] Ghofrani, M., Hassanzadeh, M., Etezadi-Amoli, M., Fadali, M. S. (2011). Smart meter based short-term load forecasting for residential customers, *in 2011 North American Power Symposium*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Boston, MA, USA, 1-5

[16] Chaouch, M. (2014). Clustering-Based Improvement of Nonparametric Functional Time Series Forecasting: Application to Intra-Day Household-Level Load Curves, *IEEE Transactions on Smart Grid*, C. 5, Sayı 1, 411-419, doi:10.1109/TSG.2013.2277171

[17] Kong, W., Dong, Z. Y., Jia, Y., Hill, D. J., Xu, Y., Zhang, Y. (2019). Short-Term Residential Load Forecasting Based on LSTM Recurrent Neural Network, *IEEE Transactions on Smart Grid*, C. 10, Sayı 1, 841-851, doi:10.1109/TSG.2017.2753802

[18] Cao, X., Dong, S., Wu, Z., Jing, Y. (2015). A Data-Driven Hybrid Optimization Model for Short-Term Residential Load Forecasting, *in 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Liverpool, UK, 283-287

[19] Cai, M., Pipattanasomporn, M., Rahman, S. (2019). Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques, *Applied Energy*, C. 236, 1078-1088, doi:10.1016/j.apenergy.2018.12.042

[20] Marinescu, A., Harris, C., Dusparic, I., Clarke, S., Cahill, V. (2013). Residential electrical demand forecasting in very small scale: An evaluation of forecasting methods, *in 2013 2nd International Workshop on Software Engineering Challenges for the Smart Grid (SE4SG)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), San Francisco, CA, USA, 25-32

[21] Mocanu, E., Nguyen, P. H., Gibescu, M., Kling, W. L. (2016). Deep learning for estimating building energy consumption, *Sustainable Energy, Grids and Networks*, C. 6, 2352-4677, doi: 10.1016/j.segan.2016.02.005

[22] Khatri, I., Dong, X., Attia, J., Qian, L. (2019). Short-term Load Forecasting on Smart Meter via Deep Learning, *in 2019 North American Power Symposium (NAPS)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Wichita, KS, USA, 1-6

[23] Yang, Y., Li, W., Gulliver, T. A., Li, S. (2020). Bayesian Deep Learning-Based Probabilistic Load Forecasting in Smart Grids, *IEEE Transactions on Industrial Informatics*, C. 16, Sayı 7, 4703-4713, doi:10.1109/TII.2019.2942353

[24] Syed, D., Refaat, S. S., Abu-Rub, H., Bouhali, O. (2020). Short-term Power Forecasting Model Based on Dimensionality Reduction and Deep Learning Techniques for Smart Grid, *in 2020 IEEE Kansas Power and Energy Conference (KPEC)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Manhattan, KS, USA, 1-6

[25] Kiprijanovska, I., Stankoski, S., Ilievski, I., Jovanovski, S., Gams, M., Gjoreski, H. (2020). HousEEC: Day-Ahead Household Electrical Energy Consumption Forecasting Using Deep Learning, *Energies*, C. 13, Sayı 10, 2672, doi:10.3390/en13102672

[26] Kim, J.Y., Cho, S.B. (2019). Electric Energy Consumption Prediction by Deep Learning with State Explainable Autoencoder, *Energies*, C. 12, Sayı 4, 739, doi:10.3390/en12040739

[27] Shi, H., Xu, M., Li, R. (2018). Deep Learning for Household Load Forecasting—A Novel Pooling Deep RNN, *IEEE Transactions on Smart Grid*, C. 9, Sayı 5, 5271-5280, doi:10.1109/TSG.2017.2686012

[28] Rahman, A., Srikumar, V., Smith, A. D. (2018). Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks, *Applied Energy*, C. 212, 372-385, doi:10.1016/j.apenergy.2017.12.051

[29] Jiao, R., Zhang, T., Jiang, Y., He, H. (2018). Short-Term Non-Residential Load Forecasting Based on Multiple Sequences LSTM Recurrent Neural Network, *IEEE Access*, C. 6, 59438-59448, doi:10.1109/ACCESS.2018.2873712

[30] Wang, X., Zhao, T., Liu, H., He, R. (2019). Power Consumption Predicting and Anomaly Detection Based on Long Short-Term Memory Neural Network, *2019 IEEE 4th International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Chengdu, China, 487-491

[31] Wang, Y., Gan, D., Sun, M., Zhang, N., Lu, Z., Kang, C. (2019). Probabilistic individual load forecasting using pinball loss guided LSTM, *Applied Energy*, C. 235, 10-20, doi:10.1016/j.apenergy.2018.10.078

[32] Somu, N. Raman M. R. R., Ramamritham, K. (2020). A hybrid model for building energy consumption forecasting using long short term memory networks, *Applied Energy*, C. 261, 114131, doi:10.1016/j.apenergy.2019.114131

[33] Marino, D. L., Amarasinghe, K., Manic, M. (2016). Building energy load forecasting using Deep Neural Networks, *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Florence, Italy, 7046-7051

[34] Vos, M., Bender-Saebelkampf, C., Albayrak, S. (2018). Residential Short-Term Load Forecasting Using Convolutional Neural Networks, *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids, SmartGridComm 2018*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Aalborg, Denmark, 1-6

[35] Elvers, A., Voß, M., Albayrak, S. (2019). Short-Term Probabilistic Load Forecasting at Low Aggregation Levels Using Convolutional Neural Networks, *2019 IEEE Milan PowerTech*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Milan, Italy, 1-6

[36] Kim, J., Moon, J., Hwang, E., Kang, P. (2019). Recurrent inception convolution neural network for multi short-term load forecasting, *Energy and Buildings*, C. 194, 328-341, doi:10.1016/j.enbuild.2019.04.034

[37] Kaligambe, A., Fujita, G. (2020). Short-Term Load Forecasting for Commercial Buildings Using 1D Convolutional Neural Networks, *2020 IEEE PES/IAS PowerAfrica*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Nairobi, Kenya, 1-5

[38] Amarasinghe, K., Marino, D. L., Manic, M. (2017). Deep neural networks for energy load forecasting, *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Edinburgh, UK, 1483-1488

[39] Eneyew, D. D., Capretz, M. A. M., Bitsuamlak, G. T., Mir, S. (2020). Predicting Residential Energy Consumption Using Wavelet Decomposition With Deep Neural Network, *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Miami, FL, USA, 895-900

[40] Kim, T. Y., Cho, S. B. (2019). Predicting residential energy consumption using CNN-LSTM neural networks, *Energy*, C. 182, 72-81, doi:10.1016/j.energy.2019.05.230

[41] Alhussein, M., Aurangzeb, K., Haider, S. I. (2020). Hybrid CNN-LSTM Model for Short-Term Individual Household Load Forecasting, *IEEE Access*, C. 8, IEEE, 180544-180557, doi:10.1109/ACCESS.2020.3028281

[42] Fan, C., Sun, Y., Xiao, F., Ma, J., Lee, D., Wang, J., Tseng, Y. C. (2020). Statistical investigations of transfer learning-based methodology for short-term building energy predictions, *Applied Energy*, C. 262, 114499, doi:10.1016/j.apenergy.2020.114499

[43] Tian, Y., Sehovac, L., Grolinger, K. (2019). Similarity-Based Chained Transfer Learning for Energy Forecasting with Big Data, *IEEE Access*, C. 7, 139895-139908, doi:10.1109/ACCESS.2019.2943752

[44] Gao, Y., Ruan, Y., Fang, C., Yin, S. (2020). Deep learning and transfer learning models of energy consumption forecasting for a building with poor information data, *Energy and Buildings*, C. 223, 110156, doi:10.1016/j.enbuild.2020.110156

[45] Ma, J., Cheng, J.C.P., Jiang, F., Chen, W., Wang, M., Zhai, C. (2020). A bi-directional missing data imputation scheme based on LSTM and transfer learning for building energy data, *Energy and Buildings*, C. 216, 109941, doi:10.1016/j.enbuild.2020.109941

[46] Liu, T., Tan, Z., Xu, C., Chen, H., Li, Z. (2020). Study on deep reinforcement learning techniques for building energy consumption forecasting, *Energy and Buildings*, C. 208, 109675, doi:10.1016/j.enbuild.2019.109675

[47] Wijaya, T. K., Vasirani, M., Humeau, S., Aberer, K. (2015). Cluster-based aggregate forecasting for residential electricity demand using smart meter data, *2015 IEEE International Conference on Big Data (Big Data)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Santa Clara, CA, USA, 879-887

[48] Quilumba, F. L., Lee, W., Huang, H., Wang, D. Y., Szabados, R. L. (2015). Using Smart Meter Data to Improve the Accuracy of Intraday Load Forecasting Considering Customer Behavior Similarities, *IEEE Transactions on Smart Grid*, C. 6, Sayı 2, 911-918, doi:10.1109/TSG.2014.2364233

[49] Bandyopadhyay, S., Ganu, T., Khadilkar, H., Arya, V. (2015). Individual and aggregate electrical load forecasting: One for all and all for one, *e-Energy 2015 - Proceedings of the 2015 ACM 6th International Conference on Future Energy Systems*, published by Association for Computing Machinery (https://dl.acm.org/), Bangalore, India, 121-130

[50] Al-Wakeel, A., Wu, J., Jenkins, N. (2017). k-means based load estimation of domestic smart meter measurements, *Applied Energy*, C. 194, 333-342, doi:10.1016/j.apenergy.2016.06.046

[51] Yıldız, B., Bilbao, J. I., Dore, J., Sproul, A. (2018). Household electricity load forecasting using historical smart meter data with clustering and classification techniques, *2018 IEEE Innovative Smart Grid Technologies - Asia (ISGT Asia)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Singapore, 873-879

[52] Teeraratkul, T., O'Neill, D., Lall, S. (2018). Shape-Based Approach to Household Electric Load Curve Clustering and Prediction, *IEEE Transactions on Smart Grid*, C. 9, Sayı 5, 5196-5206, doi:10.1109/TSG.2017.2683461

[53] Laurinec, P., Lucká, M. (2018). Clustering-based forecasting method for individual consumers electricity load using time series representations, *Open Computer Science*, C. 8, Sayı 1, 38-50, doi:10.1515/comp-2018-0006

[54] Hsiao, Y. (2015). Household Electricity Demand Forecast Based on Context Information and User Daily Schedule Analysis From Meter Data, *IEEE Transactions on Industrial Informatics*, C. 11, Sayı 1, 33-43, doi:10.1109/TII.2014.2363584

[55] Stephen, B., Tang, X., Harvey, P. R., Galloway, S., Jennett, K. I. (2017). Incorporating Practice Theory in Sub-Profile Models for Short Term Aggregated Residential Load Forecasting, *IEEE Transactions on Smart Grid*, C. 8, Sayı 4, 1591-1598, doi:10.1109/TSG.2015.2493205

[56] Lusis, P., Khalilpour, K. R., Andrew, L., Liebman, A. (2017). Short-term residential load forecasting: Impact of calendar effects and forecast granularity, *Applied Energy*, C. 205, 654-669, doi:10.1016/j.apenergy.2017.07.114

[57] Wang, Z., Wang, H. (2021). Identifying the relationship between seasonal variation in residential load and socioeconomic characteristics, *Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, published by Association for Computing Machinery (https://dl.acm.org/), Coimbra, Portugal, 160–163

[58] Deb, C., Zhang, F., Yang, J., Lee, S. E., Shah, K. W. (2017). A review on time series forecasting techniques for building energy consumption, *Renewable and Sustainable Energy Reviews*, C. 74, 902-924, doi:10.1016/j.rser.2017.02.085

[59] Depuru, S. S. S. R., Wang, L., Devabhaktuni, V. (2011). Electricity theft: Overview, issues, prevention and a smart meter based approach to control theft, *Energy Policy*, C. 39, Sayı 2, 1007-1015, doi:10.1016/j.enpol.2010.11.037

[60] Messinis, G. M., Hatziargyriou, N. D. (2018). Review of non-technical loss detection methods, *Electric Power Systems Research*, C. 158, 250-266, doi:10.1016/j.epsr.2018.01.005

[61] Figueroa, G., Chen, Y. S., Nelson, A., Chu, C. C. (2017). Improved practices in machine learning algorithms for NTL detection with imbalanced data, *2017 IEEE Power & Energy Society General Meeting*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Chicago, IL, USA, 1-5

[62] Nagi, J., Yap, K. S., Tiong, S. K., Ahmed, S. K., Mohamad, M. (2009). Nontechnical loss detection for metered customers in power utility using support vector machines, *IEEE Transactions on Power Delivery*, C. 25, Sayı 2, 1162-1171, doi:10.1109/TPWRD.2009.2030890

[63] Glauner, P., Meira, J. A., Dolberg, L., State, R., Bettinger, F., Rangoni, Y. (2016). Neighborhood features help detecting non-technical losses in big data sets, *Proceedings of the 3rd IEEE/ACM International Conference on Big Data Computing, Applications and Technologies*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Shanghai, China, 253-261

[64] Yap, K. S., Abidin, I. Z., Ahmad, A. R., Hussien, Z. F., Pok, H. L., Ismail, F. I., Mohamad, A. M. (2007). Abnormalities and fraud electric meter detection using hybrid support vector machine & genetic algorithm, *Proceedings of the third conference on IASTED International Conference: Advances in Computer Science and Technology*, published by Association for Computing Machinery (https://dl.acm.org/), Phuket, Thailand, 388-392

[65] Nagi, J., Mohammad, A. M., Yap, K. S., Tiong, S. K., Ahmed, S. K. (2008). Non-Technical Loss analysis for detection of electricity theft using support vector machines, *2008 IEEE 2nd International Power and Energy Conference*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Johor Bahru, Malaysia, 907-912

[66] Depuru, S. S. S. R., Wang, L., Devabhaktuni, V. (2011). Support vector machine based data classification for detection of electricity theft, *2011 IEEE/PES Power Systems Conference and Exposition*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Phoenix, AZ, USA, 1-8

[67] Pereira, D. R., Pazoti, M. A., Pereira, L. A. M., Rodrigues, D., Ramos, C. O., Souza, A. N., Papa, J. P. (2016). Social-Spider Optimization-based Support Vector Machines applied for energy theft detection, *Computers & Electrical Engineering*, C. 49, 25-38, doi:10.1016/j.compeleceng.2015.11.001

[68] Pereira, L. A. M., Afonso, L. C. S., Papa, J. P., Vale, Z. A., Ramos, C. C. O., Gastaldello, D. S., Souza, A. N. (2013). Multilayer perceptron neural networks training through charged system search and its application for non-technical losses detection, *2013 IEEE PES Conference on Innovative Smart Grid Technologies (ISGT Latin America)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Sao Paulo, Brazil, 1-6

[69] Ford, V., Siraj, A., Eberle, W. (2014). Smart grid energy fraud detection using artificial neural networks, *2014 IEEE Symposium on Computational Intelligence Applications in Smart Grid (CIASG)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Orlando, FL, USA, 1-6

[70] Labate, D., Giubbini, P., Chicco, G., Piglione, F. (2015). Shape: The load prediction and non-technical losses modules, *Proceedings of the Congrès International des Réseaux Electriques de Distribution (CIRED'15)*, published by CIRED (http://www.cired.net/), Lyon, France, 1428–1435

[71] Lu, X., Zhou, Y., Wang, Z., Yi, Y., Feng, L., Wang, F. (2019). Knowledge embedded semi-supervised deep learning for detecting non-technical losses in the smart grid, *Energies*, C. 12, Sayı 18, 3452, doi:10.3390/en12183452

[72] Ayub, N., Aurangzeb, K., Awais, M., Ali, U. (2020). Electricity Theft Detection using CNN-GRU and Manta Ray Foraging Optimization Algorithm, *2020 IEEE 23rd International Multitopic Conference (INMIC)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Bahawalpur, Pakistan, 1-6

[73] Gaoqi, L., Junhua, Z., Fengji, L., Steven, R. W., Zhao, Y. D. (2017). A Review of False Data Injection Attacks Against Modern Power Systems, *IEEE Transactions on Smart Grid*, C. 8, Sayı 4, 1630-1638, doi:10.1109/TSG.2015.2495133

[74] Gusrialdi, A., Qu, Z. (2019). Smart Grid Security: Attacks and Defenses, *Smart Grid Control: Overview and Research Opportunities*, Springer International Publishing, New York

[75] Gönen, S., Sayan, H. H., Yılmaz, E. N., Üstünsoy, F., Karacayılmaz, G. (2020). False data injection attacks and the insider threat in smart systems, *Computers and Security*, C. 97, 101955, doi:10.1016/j.cose.2020.101955

[76] Chuwa, M. G., Wang, F. (2021). A review of non-technical loss attack models and detection methods in the smart grid, *Electric Power Systems Research*, C. 199, 107415, doi:10.1016/j.epsr.2021.107415

[77] McLaughlin, S., Holbert, B., Fawaz, A., Berthier, R., Zonouz, S. (2013). A multi-sensor energy theft detection framework for advanced metering infrastructures, *IEEE Journal on Selected Areas in Communications*, C. 31, Sayı 7, 1319-1330, doi:10.1109/JSAC.2013.130714

[78] Zhang, Y., Wang, J., Chen, B. (2021). Detecting False Data Injection Attacks in Smart Grids: A Semi-Supervised Deep Learning Approach, *IEEE Transactions on Smart Grid*, C. 12, Sayı 1, 623-634, doi:10.1109/TSG.2020.3010510

[79] Li, Y., Wang, Y., Hu, S. (2020). Online Generative Adversary Network Based Measurement Recovery in False Data Injection Attacks: A Cyber-Physical Approach, *IEEE Transactions on Industrial Informatics*, C. 16, Sayı 3, 2031-2043, doi:10.1109/TII.2019.2921106

[80] Pu, Q., Qin, H., Han, H., Xia, Y., Li, Z., Xie, K., Wang, W. (2018). Detection mechanism of FDI attack feature based on deep learning, *Proceedings - 2018 IEEE SmartWorld, Ubiquitous Intelligence and Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People and Smart City Innovations*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Guangzhou, China, 1761-1765

[81] Leite, J. B., Mantovani, J. R. S. (2018). Detecting and Locating Non-Technical Losses in Modern Distribution Networks, *IEEE Transactions on Smart Grid*, C. 9, Sayı 2, 1023-1032, doi:10.1109/TSG.2016.2574714

[82] Liu, Y., Hu, S. (2015). Cyberthreat Analysis and Detection for Energy Theft in Social Networking of Smart Homes, *IEEE Transactions on Computational Social Systems*, C. 2, Sayı 4, 148-158, doi:10.1109/TCSS.2016.2519506

[83] Liu, X., Zhu, P., Zhang, Y., Chen, K. (2015). A Collaborative Intrusion Detection Mechanism Against False Data Injection Attack in Advanced Metering Infrastructure, *IEEE Transactions on Smart Grid*, C. 6, Sayı 5, 2435-2443, doi:10.1109/TSG.2015.2418280

[84] Zhou, K., Fu, C., Yang, S. (2016). Big data driven smart energy management: From big data to big insights, *Renewable and Sustainable Energy Reviews*, C. 56, 215-225, doi:10.1016/j.rser.2015.11.050

[85] Sagiroglu, S., Terzi, R., Canbay, Y., Colak, I. (2016). Big data issues in smart grid systems, *2016 IEEE International Conference on Renewable Energy Research and Applications (ICRERA)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Birmingham, UK, 1007-1012

[86] Dunnan, L., Long, Z., Canbing, L., Kunlong, M., Yujiao, C., Yijia, C. (2018). A Distributed Short-Term Load Forecasting Method Based on Local Weather Information, *IEEE Systems Journal*, C. 12, Sayı 1, 208-215, doi:10.1109/JSYST.2016.2594208

[87] Zhang, P., Wu, X., Wang, X., Bi, S. (2015). Short-term load forecasting based on big data technologies, *CSEE Journal of Power and Energy Systems*, C. 1, Sayı 3, 59-67, doi:10.17775/cseejpes.2015.00036

[88] Oprea, S., Bâra, A. (2019). Machine Learning Algorithms for Short-Term Load Forecast in Residential Buildings Using Smart Meters, Sensors and Big Data Solutions, *IEEE Access*, C. 7, 177874-177889, doi:10.1109/ACCESS.2019.2958383

[89] Alemazkoor, N., Tootkaboni, M., Nateghi, R., Louhghalam, A. (2022). Smart-Meter Big Data for Load Forecasting: An Alternative Approach to Clustering, *IEEE Access*, C. 10, 8377-8387, doi:10.1109/ACCESS.2022.3142680

[90] Deng, B., Wen, Y., Yuan, P. (2020). Hybrid Short-Term Load Forecasting using the Hadoop MapReduce Framework, *2020 IEEE Power Energy Society General Meeting (PESGM)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Montreal, QC, Canada, 1-5

[91] Zainab, A., Refaat, S. S., Abu-Rub, H., Bouhali, O. (2020). Distributed Computing for Smart Meter Data Management for Electrical Utility Applications, *2020 Cybernetics Informatics (KI)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Velke Karlovice, Czech Republic, 1-6

[92] Dong, C., Du, L., Ji, F., Song, Z., Zheng, Y., Howard, A., Intrevado, P., Woodbridge, D. M., Howard, A. J. (2018). *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Exeter, UK, 1293-1298

[93] Nagi, J., Yap, K. S., Tiong, S. K., Ahmed, S. K., Mohamad, M. (2009). Nontechnical loss detection for metered customers in power utility using support vector machines, *IEEE transactions on Power Delivery*, C. 25, Sayı 2, 1162-1171, doi:10.1109/TPWRD.2009.2030890

[94] Jokar, P., Arianpoo, N., Leung, V. C. M. (2016). Electricity Theft Detection in AMI Using Customers' Consumption Patterns, *IEEE Transactions on Smart Grid*, C. 7, Sayı 1, 216-226, doi:10.1109/TSG.2015.2425222

[95] Buzau, M.M., Tejedor-Aguilera, J., Cruz-Romero, P., Gómez-Expósito, A. (2019). Hybrid deep neural networks for detection of non-technical losses in electricity smart meters, *IEEE Transactions on Power Systems*, C. 35, Sayı 2, 1254-1263, doi:10.1109/TPWRS.2019.2943115

[96] Han, W., Xiao, Y. (2020). Edge computing enabled non-technical loss fraud detection for big data security analytic in Smart Grid, *Journal of Ambient Intelligence and Humanized Computing*, C. 11, Sayı 4, 1697-1708, doi:10.1007/s12652-019-01381-4

[97] Lipčák, P., Macak, M., Rossi, B. (2019). Big Data Platform for Smart Grids Power Consumption Anomaly Detection, *2019 Federated Conference on Computer Science and Information Systems (FedCSIS)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Leipzig, Germany, 771-780

[98] International Electrotechnical Commission-SyC Smart Energy, IEC TR 63097:2017 Smart grid standardization roadmap, https://webstore.iec.ch/publication/27785#additionalinfo, Erişim: 31-12-2021

[99] Türkiye Elektrik Dağıtım A.Ş. Genel Müdürlüğü-Strateji Geliştirme Dairesi Başkanlığı, Elektronik Elektrik Sayaçları Teknik Şartnamesi, https://www.tedas.gov.tr/#!tedas_sartnameler_arge, Erişim: 09-05-2022

[100] Weranga, K. S. K., Kumarawadu, S., Chandima, D. P. (2014). Smart metering design and applications, Springer, Singapore

[101] Yetişken, M. F. (2010). Elektrik Sayaçlarını Uzaktan Okuma Teknikleri Ve Prototip Geliştirilmesi, Sakarya Üniversitesi, Fen Bilimleri Enstitüsü

[102] International Electrotechnical Commission, IEC 62056-6-1:2017 Electricity metering data exchange - The DLMS/COSEM suite - Part 6-1: Object Identification System (OBIS), https://webstore.iec.ch/publication/32782#additionalinfo, Erişim: 09-05-2022

[103] Zikopoulos, P., Eaton, C. (2011). Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data, McGraw-Hill Education, New York

[104] Kaisler, S., Armour, F., Espinosa, J. A., Money, W. (2013). Big data: Issues and challenges moving forward, *2013 46th Hawaii International Conference on System Sciences*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Wailea, HI, USA, 995-1004

[105] Liu, K., Sheng, W., Zhang, D., Jia, D., Hu, L., He, K. (2015). Big data application requirements and scenario analysis in smart distribution network, *Proceedings of the CSEE*, C. 35, Sayı 2, 287-293, doi:10.13334/j.0258-8013.pcsee.2015.02.004

[106] Fan, C., Xiao, F., Li, Z., Wang, J. (2018). Unsupervised data analytics in mining big building operational data for energy efficiency enhancement: A review, *Energy and Buildings*, C. 159, 296-308, doi:10.1016/j.enbuild.2017.11.008

[107] Cheng, Y., Chen, K., Sun, H., Zhang, Y., Tao, F. (2018). Data and knowledge mining with big data towards smart production, *Journal of Industrial Information Integration*, C. 9, 1-13, doi:10.1016/j.jii.2017.08.001

[108] Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., Khan, S. U. (2015). The rise of "big data" on cloud computing: Review and open research issues, *Information Systems*, C. 47, 98-115, doi:10.1016/j.is.2014.07.006

[109] Briody, D., Economist Intelligence Unit. (2011). Big data - Harnessing a game-changing asset, https://cdn.ymaws.com/edmcouncil.org/resource/resmgr/featured_documents/Press/EDMC_Big_Data_Harn_Dec11.pdf, Erişim: 09-05-2022

[110] Harford, T., (2014). Big data: are we making a big mistake?, Financial Times, https://www.ft.com/content/21a6e7d8-b479-11e3-a09a-00144feabdc0#axzz3aEMhxJOv, Erişim: 09-05-2022

[111] The Economist, Data, data everywhere, https://www.economist.com/special-report/2010/02/27/data-data-everywhere, Erişim: 02-12-2021

[112] The Economist Intelligence Unit, Power from big data, https://impact.econ-asia.com/perspectives/technology-innovation/power-big-data, Erişim: 03-12-2021

[113] IBM Corporation, Managing big data for smart grids and smart meters, http://www.ibm.com/services/multimedia/Managing_big_data_for_smart_grids_and_smart_meters.pdf, Erişim: 31-07-2015

[114] Kezunovic, M., Xie, L., Grijalva, S. (2013). The role of big data in improving power system operation and protection, *Proceedings of IREP Symposium: Bulk Power System Dynamics and Control - IX Optimization, Security and Control of the Emerging Power Grid, IREP 2013*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Rethymno, Greece, 1-9

[115] Ukil, A., Zivanovic, R. (2014). Automated analysis of power systems disturbance records: Smart Grid big data perspective, *2014 IEEE Innovative Smart Grid Technologies - Asia, ISGT ASIA 2014*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Kuala Lumpur, Malaysia, 126-131

[116] Lazer, D., Kennedy, R., King, G., Vespignani, A. (2014). The parable of google flu: Traps in big data analysis, *Science*, C. 343, Sayı 6176, 1203-1205, doi:10.1126/science.1248506

[117] Boyd, D., Crawford, K. (2012). Critical questions for big data: Provocations for a cultural, technological, and scholarly phenomenon, *Information Communication and Society*, C. 15, Sayı 5, 662-679, doi:10.1080/1369118X.2012.678878

[118] Tanenbaum, A. S., Steen, M. (2006). Distributed Systems: Principles and Paradigms (2nd Edition), Prentice-Hall, Inc., USA

[119] Duncan, R. (1990). A survey of parallel computer architectures, *Computer*, C. 23, Sayı 2, 5-16, doi:10.1109/2.44900

[120] Foster, I. (2002). The grid: A new infrastructure for 21st century science, *Physics Today*, C. 55, Sayı 2, 51-62, doi:10.1063/1.1461327

[121] Erl, T. (2005). Service-Oriented Architecture: Concepts, Technology, and Design, Prentice Hall PTR, New York

[122] Foster, I., Kesselman, C. (2003). The grid 2: Blueprint for a new computing infrastructure, Elsevier

[123] Hoffa, C., Mehta, G., Freeman, T., Deelman, E., Keahey, K., Berriman, B., Good, J. (2008). On the Use of Cloud Computing for Scientific Workflows, *2008 IEEE Fourth International Conference on eScience*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Indianapolis, IN, USA, 640-645

[124] Hallaç, İ. R. (2014). Büyük Veri Analizinde Dağıtık Makine Öğrenmesi Algoritmalarının Kullanılması, Fırat Üniversitesi, Fen Bilimleri Enstitüsü

[125] Collins, E. (2014). Big Data in the Public Cloud, *IEEE Cloud Computing*, C. 1, Sayı 2, 13-15, doi:10.1109/MCC.2014.29

[126] Apache Software Foundation, Apache Hadoop, https://hadoop.apache.org/, Erişim: 14-12-2021

[127] Ghemawat, S., Gobioff, H., Leung, S. T. (2003). The Google File System, *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles*, published by Association for Computing Machinery (https://www.acm.org/), Bolton Landing, NY, USA, 29–43

[128] Dean, J., Sanjay, G. (2004). MapReduce: Simplified Data Processing on Large Clusters, *OSDI'04: Sixth Symposium on Operating System Design and Implementation*, published by USENIX and ACM SIGOPS (https://www.usenix.org/), San Francisco, CA, USA, 137-150

[129] Apache Software Foundation, HDFS Architecture, https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html#Large_Data_Sets, Erişim: 15-12-2021

[130] Vavilapalli , V. K., Murthy, A. C., Douglas, C., Agarwal, S., Konar, M., Evans, R., Graves, T., Lowe, J., Shah, H., Seth, S., Saha, B., Curino, C., O'Malley, O., Radia, S., Reed, B., Baldeschwieler, E. (2013). Apache hadoop YARN: Yet another resource negotiator, *Proceedings of the 4th Annual Symposium on Cloud Computing, SoCC 2013*, published by Association for Computing Machinery (https://www.acm.org/), Santa Clara, California, USA, 1-16

[131] Subramaniam, A., Hadoop YARN Tutorial – Learn the Fundamentals of YARN Architecture, https://www.edureka.co/blog/hadoop-yarn-tutorial/, Erişim: 20-12-2021

[132] Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M. J., Ghodsi, A., Gonzalez, J., Shenker, S., Stoica, I. (2016). Apache Spark: A Unified Engine for Big Data Processing, *Communications of the ACM*, C. 59, Sayı 11, 56-65, doi:10.1145/2934664

[133] Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauly, M., Franklin, M. J., Shenker, S., Stoica, I. (2012). Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing, *9th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 12)*, published by Association for Computing Machinery (https://www.acm.org/), San Jose, CA, USA, 15-28

[134] Armbrust, M., Xin, R. S., Lian, C., Huai, Y., Liu, D., Bradley, J. K., Meng, X., Kaftan, T., Franklin, M. J., Ghodsi, A., Zaharia, M. (2015). Spark SQL: Relational Data Processing in Spark, *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, published by Association for Computing Machinery (https://www.acm.org/), Melbourne, Victoria, Australia, 1383–1394

[135] Zaharia, M., Das, T., Li, H., Hunter, T., Shenker, S., Stoica, I. (2013). Discretized Streams: Fault-Tolerant Streaming Computation at Scale, *SOSP '13: Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, published by Association for Computing Machinery (https://www.acm.org/), Farminton, Pennsylvania, 423–438

[136] Gonzalez, J. E., Xin, R. S., Dave, A., Crankshaw, D., Franklin, M. J., Stoica, I. (2014). GraphX: Graph Processing in a Distributed Dataflow Framework, *11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14)*, published by USENIX (https://www.usenix.org/), Broomfield, CO, 599-613

[137] Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., Freeman, J., Tsai, D. B., Amde, M., Owen, S., Xin, D., Xin, R., Franklin, M. J., Zadeh, R., Zaharia, M., Talwalkar, A. (2016). MLlib: Machine Learning in Apache Spark, *The Journal of Machine Learning Research*, C. 17, Sayı 1, 1235–1241

[138] European Network of Transmission System Operators for Electricity-ENTSO-E, Power in Transition Research, Development & Innovation roadmap 2017–2026, https://riroadmap.entsoe.eu/, Erişim: 27-12-2021

[139] Australian Government- Energy Efficiency Initiative, Smart Grid, Smart City, https://webarchive.nla.gov.au/awa/20160615043539/http://www.industry.gov.au/Energy/Programmes/SmartGridSmartCity/Pages/default.aspx, Erişim: 10-01-2022

[140] Zanetti, M., Jamhour, E., Pellenz, M., Penna, M., Zambenedetti, V., Chueiri, I. (2019). A Tunable Fraud Detection System for Advanced Metering Infrastructure Using Short-Lived Patterns, *IEEE Transactions on Smart Grid*, C. 10, Sayı 1, 830-840, doi:10.1109/TSG.2017.2753738

[141] Hortonworks Data Platform Command Line Installation, Hortonworks, Inc., https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.6.5/bk_command-line-installation/content/index.html, Erişim: 18-01-2022

[142] Cook, D., (2016). Practical Machine Learning with H2O: Powerful, Scalable Techniques for Deep Learning and AI, O'Reilly Media, Inc.

[143] Malohlava, M., Hava, J., Mehta, N. (2022). Machine Learning with Sparkling Water: H2O + Spark, https://docs.h2o.ai/sparkling-water/3.0/latest-stable/booklet/SparklingWaterBooklet.pdf, Erişim: 23-01-2022

[144] Shepard, R. N. (1962). The analysis of proximities: Multidimensional scaling with an unknown distance function. Part I., *Psychometrika*, C. 27, Sayı 2, 125–140, doi:10.1007/BF02289630

[145] Shepard, R. N. (1962). The analysis of proximities: Multidimensional scaling with an unknown distance function. Part II., *Psychometrika*, C. 27, Sayı 3, 219–246, doi:10.1007/BF02289621

[146] Kruskal, J. B. (1964). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis, *Psychometrika*, C. 29, Sayı 1, 1–27, doi:10.1007/BF02289565

[147] Kruskal, J. B. (1964). Nonmetric multidimensional scaling: A numerical method, *Psychometrika*, C. 29, Sayı 2, 115–129, doi:10.1007/BF02289694

[148] Kruskal, J. B., Wish, M. (1978). Multidimensional Scaling, SAGE, Newbury Park

[149] Tatlıdil, H. (1992). Uygulamalı Çok Değişkenli İstatistiksel Analiz, Engin Yayınları

[150] Ester, M., Kriegel, H. P., Sander, J., Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, *KDD'96: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, published by Association for Computing Machinery (https://www.acm.org/), Portland, Oregon, USA, 226–231

[151] Hochreiter, S., Schmidhuber, J. (1997). Long Short-Term Memory, *Neural Computation*, C. 9, Sayı 8, 1735-1780, doi:10.1162/neco.1997.9.8.1735

[152] Gers, F. A., Schmidhuber, J., Cummins, F. (2000). Learning to Forget: Continual Prediction with LSTM, *Neural Computation*, C. 12, Sayı 10, 2451-2471, doi:10.1162/089976600300015015

[153] Lipton, Z. C., Berkowitz, J., Elkan, C. (2015). A Critical Review of Recurrent Neural Networks for Sequence Learning, *arXiv*, 1-38, doi:10.48550/ARXIV.1506.00019

[154] Schuster, M., Paliwal, K. K. (1997). Bidirectional Recurrent Neural Networks, *IEEE Transactions on Signal Processing*, C. 45, Sayı 11, 2673-2681, doi:10.1109/78.650093

[155] Graves, A., Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures, *Neural Networks*, C. 18, Sayı 5-6, 602-610, doi:10.1016/j.neunet.2005.06.042

[156] Fulcher, B. D., Little, M. A., Jones, N. S. (2013). Highly comparative time-series analysis: the empirical structure of time series and their methods, *Journal of The Royal Society Interface*, C. 10, Sayı 83, 1-12, doi:10.1098/rsif.2013.0048

[157] Yang, W., Wang, K., Zuo, W. (2012). Neighborhood Component Feature Selection for High-Dimensional Data, *Journal of Computers*, C. 7, Sayı 1, 161-168, doi:10.4304/jcp.7.1.161-168

[158] Aggarwal, C. C. (2014). Data Classification: Algorithms and Applications, Chapman & Hall/CRC, New York

[159] Ho, T. K. (1995). Random decision forests, *Proceedings of 3rd international conference on document analysis and recognition*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Montreal, QC, Canada, 278-282

[160] Breiman, L., Friedman, J., Stone, C.J., Olshen, R.A. (1984). Classification and Regression Trees, Taylor & Francis.

[161] Cortes, C., Vapnik, V. (1995). Support-vector networks, *Machine learning*, C. 20, Sayı 3, 273-297, doi:10.1007/BF00994018

[162] Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine, *Annals of Statistics*, C. 29, Sayı 5, 1189-1232, doi:10.1214/aos/1013203451

[163] Hastie, T., Tibshirani, R., Friedman, J. H. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer

[164] Chen, T., Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, published by Association for Computing Machinery (https://www.acm.org/), San Francisco, California, USA, 785-794

[165] Mitchell, R., Frank, E. (2017). Accelerating the XGBoost algorithm using GPU computing, *PeerJ Computer Science*, C. 2017, Sayı 7, 1-37, doi:10.7717/peerj-cs.127

[166] Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., Gulin, A. (2019). CatBoost: unbiased boosting with categorical features, *arXiv*, 1-23, doi:10.48550/arXiv.1706.09516

[167] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T. Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree, *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, 3149–3157, published by Association for Computing Machinery (https://www.acm.org/), Long Beach, California, USA, 3149–3157

[168] Karami, A., Johansson, R. (2014). Choosing DBSCAN Parameters Automatically using Differential Evolution, *International Journal of Computer Applications*, C. 91, Sayı 7, 1-11, doi:10.5120/15890-5059

[169] Sharma, S. (1995). Applied Multivariate Techniques, John Wiley and Sons, Inc., USA

[170] Gan, G., Ma, C., Wu, J. (2007). Data Clustering: Theory, Algorithms, and Applications (ASA-SIAM Series on Statistics and Applied Probability), Society for Industrial and Applied Mathematics, USA

[171] Liu, Y., Li, Z., Xiong, H., Gao, X., Wu, J. (2010). Understanding of Internal Clustering Validation Measures, *2010 IEEE International Conference on Data Mining*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), Sydney, NSW, Australia, 911-916

[172] Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme, *Biochimica et Biophysica Acta (BBA) - Protein Structure*, C. 405, Sayı 2, 442-451, doi:10.1016/0005-2795(75)90109-9

[173] Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales, *Educational and Psychological Measurement*, C. 20, Sayı 1, 37-46, doi:10.1177/001316446002000104

[174] Borg, I., Groenen, P. J. F. (2005). Modern Multidimensional Scaling Theory and Applications, Springer

[175] Ling, J., Doris, L., Alex, S., Sam, B., Kesheng, W., Spurlock, C. A., Annika, T. (2017). Comparison of Clustering Techniques for Residential Energy Behavior Using Smart Meter Data, *The AAAI-17 Workshop on Artificial Intelligence for Smart Grids and Smart Buildings*, published by the Association for the Advancement of Artificial Intelligence (AAAI) (https://www.aaai.org/), San Francisco, CA, USA, 260-266

[176] Ünal, F., Almalaq, A., Ekici, S. (2021). A Novel Load Forecasting Approach Based on Smart Meter Data Using Advance Preprocessing and Hybrid Deep Learning, *Applied Sciences*, C. 11, Sayı 6, 2742, doi:10.3390/app11062742

[177] Hawkins, D. M. (1980). Identification of Outliers, Springer

[178] Yıldız, B., Bilbao,J. I., Dore, J., Sproul, A. B. (2017). Recent advances in the analysis of residential electricity consumption and applications of smart meter data, *Applied Energy*, C. 208, 402-427, doi:10.1016/j.apenergy.2017.10.014

[179] sklearn.svm.SVR — scikit-learn 0.24.1 documentation, https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html/sklearn.svm.SVR, Erişim: 13-4-2021

[180] sklearn.tree.DecisionTreeRegressor — scikit-learn 0.24.1 documentation, https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html/sklearn.tree.DecisionTreeRegressor, Erişim: 13-4-2021

[181] Ünal, F., Almalaq, A., Ekici, S., Glauner, P. (2021). Big Data-Driven Detection of False Data Injection Attacks in Smart Meters, *IEEE Access*, C. 9, 144313-144326, doi:10.1109/ACCESS.2021.3122009

[182] pyspark.mllib.classification.SVMModel, https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.mllib.classification.SVMModel.html, Erişim: 03-08-2021

[183] pyspark.mllib.classification.LogisticRegressionModel, https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.mllib.classification.LogisticRegressionModel.html, Erişim: 03-08-2021

[184] pyspark.ml.classification.RandomForestClassificationModel, https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.classification.RandomForestClassificationModel.html, Erişim: 03-08-2021

[185] pyspark.ml.classification.DecisionTreeClassificationModel, https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.classification.DecisionTreeClassificationModel.html, Erişim: 03-08-2021

[186] pyspark.ml.classification.NaiveBayesModel, https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.classification.NaiveBayesModel.html, Erişim: 03-08-2021

[187] LeCun, Y., Bengio, Y., Hinton, G. (2015). Deep Learning, *Nature*, C. 521, Sayı 7553, 436-444, doi:10.1038/nature14539

[188] Gradient Boosting Machine (GBM), https://h2o-release.s3.amazonaws.com/h2o/rel-wright/1/docs-website/h2o-docs/data-science/gbm.html, Erişim: 03-08-2021

[189] XGBoost, https://h2o-release.s3.amazonaws.com/h2o/rel-wright/1/docs-website/h2o-docs/data-science/xgboost.html, Erişim: 03-08-2021

[190] CatBoost Classifier, https://catboost.ai/docs/concepts/python-reference-catboostclassifier.html, Erişim: 03-08-2021

[191] LightGBM, https://lightgbm.readthedocs.io/en/latest/Parameters.html, Erişim: 03-08-2021

[192] Glauner, P., Migliosi, A., Meira, J. A., Valtchev, P., State, R., Bettinger, F. (2017). Is big data sufficient for a reliable detection of non-technical losses?, *2017 19th International Conference on Intelligent System Application to Power Systems (ISAP)*, published by IEEE (https://ieeexplore.ieee.org/Xplore/home.jsp), San Antonio, TX, USA, 1-6

[193] Glauner, P., State, R., Valtchev, P., Duarte, D. (2018). On the reduction of biases in big data sets for the detection of irregular power usage, *Data Science and Knowledge Engineering for Sensing Decision Support: Proceedings of the 13th International FLINS Conference (FLINS 2018)*, published by World Scientific (https://www.worldscientific.com/), Belfast, Northern Ireland, UK, 439-445

[194] Xu, L., Skoularidou, M., Cuesta-Infante, A., Veeramachaneni, K. (2019). Modeling Tabular data using Conditional GAN, *NIPS'19: Proceedings of the 33rd International Conference on Neural Information Processing Systems*, published by Association for Computing Machinery (https://www.acm.org/), Vancouver, Canada, 1-11

Ekler

EK-1: TSE ve IEC tarafından belirlenen akilli sayaç tasarim ve üretim standartlari

Tablo 5.1. TSE ve IEC tarafından belirlenen akıllı sayaç tasarım ve üretim standartları

Standart Numarası (TS)	Uluslararası Standart Numarası (IEC, EN, ISO)	Kabul Tarihi	Standart Adı	Uygulama Durumu
TS EN 50470-1	EN 50470-1	28.01.2019	Elektrik ölçme donanımı (a.a.) - Bölüm 1: Genel kurallar, deneyler ve deney şartları - Ölçme donanımı (a, b ve c sınıfı)	Yürürlükte
TS EN 50470-3	EN 50470-3	28.01.2019	Elektrik ölçme donanımı (a.a.) - Bölüm 3: Özel kurallar - Aktif enerji için statik sayaçlar (a, b ve c sınıfı)	Yürürlükte
TS EN 62053-21	IEC 62053-21	24.04.2017	Elektrik ölçme donanımı(a.a.) - Özel şartlar - Bölüm 21: Statik sayaçlar aktif enerji için (Sınıf 1 ve sınıf 2)	Yürürlükten kaldırıldı (İptal Tarihi: 12.04.2021)
TS EN 62053-22	IEC 62053-22	19.11.2018	Elektrik ölçme donanımı (a.a.) - Özel kurallar - Bölüm 22: Statik sayaçlar - Aktif enerji için (sınıf 0,2 s ve sınıf 0,5 s)	Yürürlükten kaldırıldı (İptal Tarihi: 12.04.2021)
TS EN 62053-23	IEC 62053-23	19.11.2018	Elektrik ölçme donanımı (a.a.) - Özel kurallar bölüm 23: Statik sayaçlar - Reaktif enerji için (sınıf 2 ve sınıf 3)	Yürürlükten kaldırıldı (İptal Tarihi: 12.04.2021)
TS EN 62054-11	IEC 62054-11	19.11.2018	Elektrik sayacı (a.a) - Tarife ve yük kontrolü - Bölüm 11: Elektronik dalgacık kontrol alıcılarıyla ilgili özel gereklilikler (IEC 62054-11:2004/A1:2016/Düzeltme1:2018)	Yürürlükte
TS EN 62054-21	IEC 62054-21	19.11.2018	Elektrik sayacı (a.a.) - Tarife ve yük kontrolü - bölüm 21: Zaman anahtarları için özel kurallar	Yürürlükte
TS EN 62056-21	EN 62056-21	21.04.2004	Elektrik ölçümü - Sayaç okuma, tarife ve yük denetimi için veri değişimi - Bölüm 21: Doğrudan yerinde veri değişimi	Yürürlükte
TS EN 62052-11	IEC 62052-11	19.11.2018	Elektrik ölçme donanımı(a.a.) - Genel kurallar, deneyler ve deney şartları - Bölüm 11: Sayaç	Yürürlükten kaldırıldı (İptal Tarihi: 12.04.2021)
TS EN 62052-21	IEC 62052-21	19.11.2018	Elektrik ölçme donanımı (a.a.) - Genel kurallar, deneyler ve deney şartları - Bölüm 21: Tarife ve yük kontrol donanımı	Yürürlükte
TS EN 62056-6-1	IEC 62056-6-1	19.03.2018	Elektrik ölçüm veri değişimi - DLMS / COSEM paketi - Bölüm 6-1: Nesne Tanımlama Sistemi (OBIS)	Yürürlükte

EK-2: Linux İşletim Sistemine Apache Hadoop ve Apache Spark Kurulumu

Centos 7 işletim sistemine Apache Hadoop 3.1.2 paketinin indirilmesi ve düzenlenmesi:

```
$ wget https://archive.apache.org/dist/hadoop/common/hadoop-3.1.2/
hadoop-3.1.2.tar.gz
$ tar -xvzf hadoop-3.1.2.tar.gz
$ sudo mkdir /usr/local/hadoop
$ mv hadoop-3.1.2 /usr/local/hadoop
$ chown -R root /usr/local/hadoop
$ cd /usr/local/hadoop/hadoop-3.1.2
```

İşletim sistemindeki .bashrc dosyasında gerekli düzenlemelerin yapılması :

```
$ sudo nano ~/.bashrc
$ source ~/.bashrc
$ echo $HADOOP_HOME
$ cd $HADOOP_HOME/etc/hadoop
$ readlink -f /usr/bin/java | sed "s:bin/java::"
$ sudo nano hadoop-env.sh
```

Namenode ve Datanode sunucularının dosya dizinlerinin oluşturulması :

```
$ mkdir /usr/local/hadoop/hadoop-3.1.2/hdir/namenode
$ mkdir /usr/local/hadoop/hadoop-3.1.2/hdir/datanode
```

Apache Hadoop'un konfigürasyon dosyalarında gerekli ayarlamaların yapılması:

```
$ cd $HADOOP_HOME/etc/hadoop/
$ sudo nano core-site.xml
$ sudo nano hdfs-site.xml
$ sudo nano yarn-site.xml
$ sudo nano mapred-site.xml
```

core-site.xml dosyasında yapılan değişiklikler:

```
<configuration>
 <property>
   <name>fs.default.name</name>
   <value>hdfs://IP_Adresi:Port_Numarası</value>
  </property>
 </configuration>
```

hdfs-site.xml dosyasında yapılan değişiklikler:

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.name.dir</name>
    <value>file:///usr/local/hadoop/hadoop-3.1.2/hdir/namenode </value>
  </property>
  <property>
    <name>dfs.data.dir</name>
    <value>file:///usr/local/hadoop/hadoop-3.1.2/hdir/datanode </value>
  </property>
</configuration>
```

yarn-site.xml dosyasında yapılan değişiklikler:

```
<configuration>
 <property>
   <name>yarn.nodemanager.aux-services</name>
   <value>mapreduce_shuffle</value>
 </property>
<property>
    <name>yarn.nodemanager.pmem-check-enabled</name>
    <value>false</value>
</property>
<property>
    <name>yarn.nodemanager.vmem-check-enabled</name>
    <value>false</value>
</property>
</configuration>
```

mapred-site.xml dosyasında yapılan değişiklikler:

```
<configuration>
 <property>
   <name>mapreduce.framework.name</name>
   <value>yarn</value>
 </property>
</configuration>
```

Apache Hadoop kurulumu ve konfigurasyonu yapıldıktan sonra hdfs namenode'a format atılması:

```
$ cd ~
$ hdfs namenode -format
```

Centos 7 işletim sisteminde kurulum tamamlandıktan sonra bashrc ve bash profile dosyalarında gerekli düzenlemelerin yapılması:

```
$ sudo nano .bashrc
$ source .bashrc
$ sudo nano .bash_profile
$ source .bash_profile
$ echo $HADOOP_HOME
$ echo $JAVA_HOME
```

Apache Hadoop ve bileşenlerinin başlatılması:

```
$ start-yarn.sh
$ start-hdfs.sh
$ start-hadoop.sh
$ jps
```

Centos 7 işletim sistemine Apache Spark 2.4.7 sürümünün indirilmesi ve dosya dizininin düzenlenmesi:

```
$ wget https://archive.apache.org/dist/spark/spark-2.4.7/spark-2.4.7-bin-
without-hadoop.tgz
$ mv spark-2.4.7-bin-without-hadoop.tgz /home/MSI/
$ cd /home/MSI/
$ tar -xzf spark-2.4.7-bin-without-hadoop.tgz
$ ln -s spark-2.4.7-bin-without-hadoop spark
```

Centos 7 işletim sisteminde Apache Spark kurulumu tamamlandıktan sonra bashrc ve bash profile dosyalarında gerekli düzenlemelerin yapılması:

```
$ cd ~
$ sudo nano /etc/environment
$ sudo nano .bash_profile
$ source .bash_profile
$ sudo nano .bashrc
$ source .bashrc
```

Apache Spark'ın /home/MSI/spark-2.4.7-bin-without-hadoop/conf dosya dizininde bulunan spark-defaults.conf ve spark-env.sh dosyalarının konfigürasyon işlemleri:

```
spark.master yarn
spark.yarn.jars hdfs://IP_Adresi:Port_Numarası/user/spark/share/lib/
jars/*.jar
```

```
spark.executor.memory 1g
spark.driver.memory 512m
spark.yarn.am.memory 512m
export SPARK_DIST_CLASSPATH=$(/usr/local/hadoop/hadoop-3.1.2/bin/hadoop
classpath)
```

Apache Spark'ın konfigürasyon işlemleri:

```
$ cd $SPARK_HOME/conf
$ sudo nano spark-env.sh
$ source spark-env.sh
```

Apache Spark'ın işletim sisteminde başlatılması:

```
$ echo $SPARK_HOME
$ spark-shell
```

Sanal makinede ssh bağlantısının kurulması ve kullanıcı anahtar dizinin üretilmesi

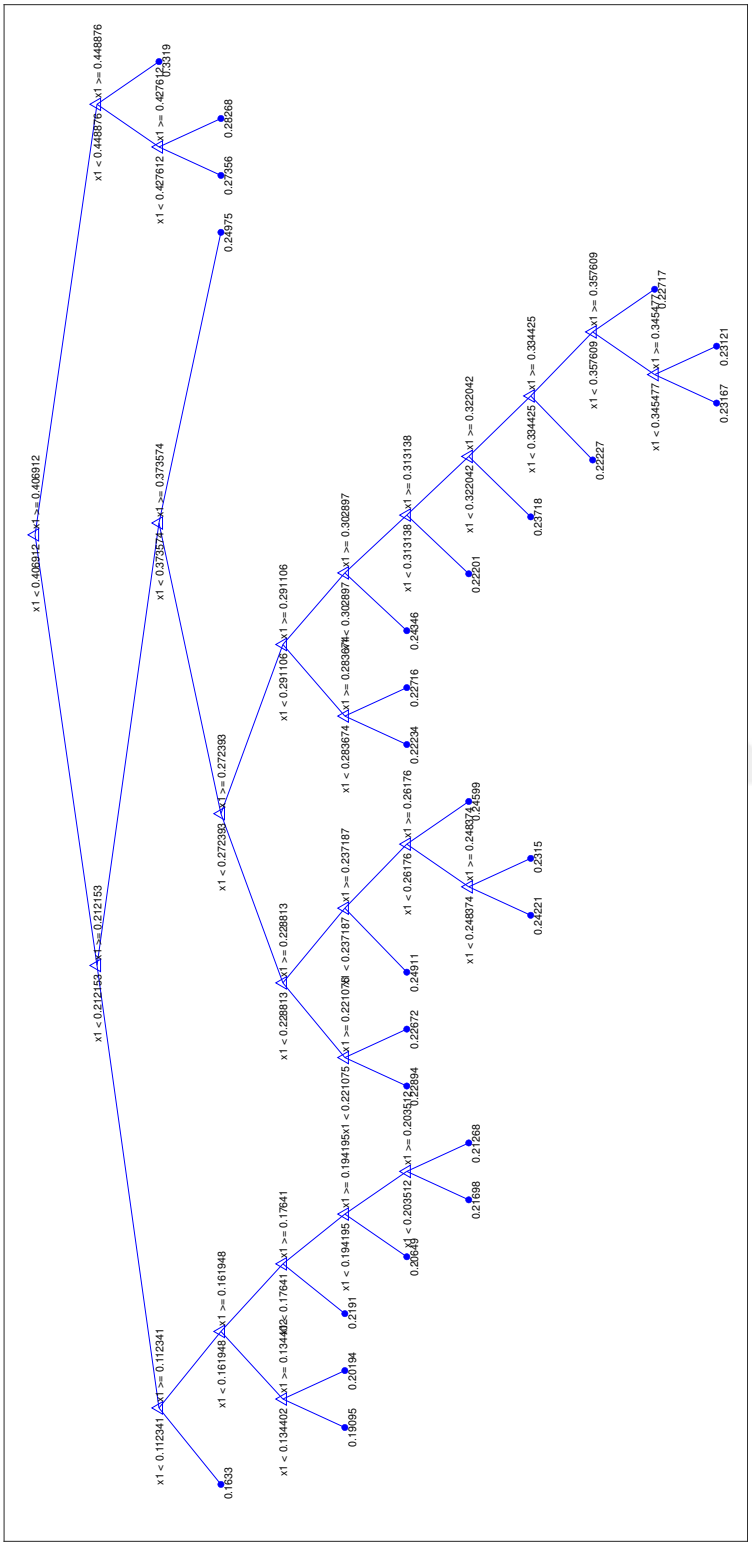
```
$ sudo yum ınstall pdsh
$ sudo yum ınstall ssh
$ ssh-keygen -t rsa
$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
$ chmod 0600 ~/.ssh/authorized_keys
$ ssh localhost
```

Apache Hadoop ve Spark kurulumu esnasında Linux işletim sistemindeki .bashrc dosyasına eklenen konfigürasyonlar

```
$ alias rm='rm -i'
$ alias cp='cp -i'
$ alias mv='mv -i'
$ alias python='/usr/local/bin/python3.6'
$ export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.282.b08-1.e17_9.
x86_64/jre/
$ export HADOOP_HOME=/usr/local/hadoop/hadoop-3.1.2
$ export HADOOP_MAPRED_HOME=$HADOOP_HOME
$ export HADOOP_COMMON_HOME=$HADOOP_HOME
$ export HADOOP_HDFS_HOME=$HADOOP_HOME
$ export YARN_HOME=$HADOOP_HOME
$ export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
$ export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin:JAVA_HOME/bin:
SPARK_HOME/bin
$ export HADOOP_INSTALL=$HADOOP_HOME
$ export SPARK_HOME="/home/MSI/spark-2.4.7-bin-without-hadoop"
```

```
$ export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
$ export YARN_CONF_DIR=$HADOOP_HOME/etc/hadoop
$ export PYSPARK_SUBMIT_ARGS="--master yarn pyspark-shell"
$ export LD_LIBRARY_PATH=/usr/local/hadoop/hadoop-3.1.2/lib/native:
$LD_LIBRARY_PATH
```

EK-3: SGSC VERI KÜMESI IÇIN STRES VE EPSILON PARAMETRELERININ KARAR AĞACI GÖSTERIMI



Şekil E5.1. SGSC veri kümesi için elde edilen stres ve epsilon parametrelerinin DT tabanlı hiyerarşik gösterimi

Özgeçmiş

Fatih ÜNAL

[redacted]

AKADEMİK FAALİYETLER

Makaleler :

1. F. Ünal, A. Almalaq, S. Ekici ve P. Glauner, "Big Data-Driven Detection of False Data Injection Attacks in Smart Meters," IEEE Access, C. 9, 144313-144326, 2021.

2. F. Ünal, A. Almalaq, ve S. Ekici, "A Novel Load Forecasting Approach Based on Smart Meter Data Using Advance Preprocessing and Hybrid Deep Learning," Applied Sciences, C. 11, Sayı 6, 2742, Mart, 2021.

AKADEMİK FAALİYETLER

Bildiriler :

1. Ünal, Fatih, ve Sami Ekici. "Extracting of Consumption Patterns in Energy Time-Series Data: A Probabilistic Suffix Tree Approach." International Applied Sciences Congress, C. 2, 17-21, 2020.

2. Ünal, Fatih, ve Sami Ekici. "A New Clustering Approach for Monthly Electricity Consumption Data." International Applied Sciences Congress, C. 2, 22-28, 2020.