T.C.
BEYKENT ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ BİLİM DALI

KREDİ KARTI HARCAMALARININ MAKİNE ÖĞRENMESİ YÖNTEMLERİYLE TAHMİN EDİLMESİ

Yüksek Lisans Tezi

Tezi Hazırlayan

Selviye GÜLTAŞLAR

İstanbul, 2022

T.C.
BEYKENT ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ BİLİM DALI

KREDİ KARTI HARCAMALARININ MAKİNE ÖĞRENMESİ YÖNTEMLERİYLE TAHMİN EDİLMESİ

Yüksek Lisans Tezi

Tezi Hazırlayan

Selviye GÜLTAŞLAR

Öğrenci No

1908020023

Orcid No

0000-0002-1202-6959

Danışman

Dr. Öğr. Üyesi Ediz ŞAYKOL

İstanbul, 2022

YEMİN METNİ

Yüksek Lisans Tezi olarak sunduğum “Kredi Kartı Harcamalarının Makine Öğrenmesi Yöntemleriyle Tahmin Edilmesi” başlıklı bu çalışmanın, bilimsel ahlak ve geleneklere uygun bir şekilde tarafımdan yazıldığını, bu tezdeki bütün akademik ve etik kurallar içinde elde ettiğimi, yararlandığım eserlerin tamamının kaynaklarda gösterildiğini ve çalışmanın içinde kullandıkları her yerde bunlara atıf yapıldığını, patent ve telif haklarını ihlal edici bir davranışımın olmadığını belirtir ve bunu onurumla doğrularım. 23/06/2022

Selviye GÜLTAŞLAR

T.C.
BEYKENT ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ MÜDÜRLÜĞÜ
TEZLİ YÜKSEK LİSANS SINAV TUTANAĞI

23/06/2022

Enstitümüz ***Bilgisayar Mühendisliği*** Anabilim Dalı ***Bilgisayar Mühendisliği*** Programı yüksek lisans öğrencilerinden ***1908020023*** numaralı ***Selviye GÜLTAŞLAR***'ın *"Beykent Üniversitesi Lisansüstü Eğitim – Öğretim Yönetmeliği"*nin ilgili maddesine göre hazırlayarak, Enstitümüze teslim ettiği ***"Kredi Kartı Harcamalarının Makine Öğrenmesi Yöntemleriyle Tahmin Edilmesi"*** konulu tezini, Yönetim Kurulumuzun 31/05/2022 tarih ve 2022/22 sayılı toplantısında seçilen ve On-Line toplanan biz jüri üyeleri huzurunda, Beykent Üniversitesi Lisansüstü Eğitim ve Öğretim Yönetmeliğinin 29. maddesinin 3. fıkrası gereğince 45 dakika süre ile Zoom programı aracılığıyla on-line olarak aday tarafından savunulmuş ve sonuçta adayın tezi hakkında ***"OYBİRLİĞİ"*** ile ***"KABUL"*** kararı verilmiştir.

İşbu tutanak, 2 nüsha olarak hazırlanmış ve Enstitü Müdürlüğü'ne sunulmak üzere tarafımızdan düzenlenmiştir.

DANIŞMAN
Dr. Öğr. Üyesi Ed*** ŞA***
(Beykent Üniversitesi)

ÜYE
Dr. Öğr. Üyesi At*** YI***
(Beykent Üniversitesi)

ÜYE
Dr. Öğr. Üyesi So*** SE***
(Kütahya Dumlupınar Üniversitesi)

Adı ve Soyadı : Selviye GÜLTAŞLAR
Danışmanı : Dr. Öğr. Üyesi Ediz ŞAYKOL
Türü ve Tarihi : Yüksek Lisans Tezi, 2022
Alanı : Bilgisayar Mühendisliği
Anahtar Kelimeler : Makine Öğrenmesi, Makine Öğrenmesi ile Sınıflandırma ve Tahmin, Kredi Kartı Harcama Tahmini

ÖZ

KREDİ KARTI HARCAMALARININ MAKİNE ÖĞRENMESİ YÖNTEMLERİYLE TAHMİN EDİLMESİ

Günümüzde artan harcama isteği ve teknolojik gelişmelerin sonucunca kredi kartı kullanımı oldukça yaygınlaşmıştır. Bu çalışmada kredi kartı harcamalarının makine öğrenmesinin alt dalı olan derin öğrenme yöntemleri kullanılarak tahmin edilmesi gerçekleşmiştir.

Özel bir ödeme sistemleri firmasından alınan 10000 adet örnek içeren veri setine ait veriler üzerinden incelemeler yapılmış ve LSTM, ARIMA ve PROPHET algoritmaları ile veri seti derin öğrenmeye tabi tutulmuştur. Uygulama geliştirilirken Python ortamı kullanılmıştır. Yapılan çalışma sonucunda LSTM yönteminin ARIMA ve PROPHET yöntemlerine göre daha başarılı sonuçlar vermiş olduğu görülmüştür.

Name and Surname	:	Selviye GÜLTAŞLAR
Supervisor	:	Dr. Lecturer Ediz ŞAYKOL
Degree and Date	:	Master's Thesis, 2022
Major	:	Computer Engineering
Key Words	:	Machine Learning, Classification and Prediction with Machine Learning, Estimation of Credit Card Expenditure

ABSTRACT

ESTIMATING CREDIT CARD SPENDING WITH MACHINE LEARNING METHODS

These days, as a result of the increasing desire to spend, the use of credit cards has become quite common. In this study, the estimation of credit card expenditures with deep learning methods has been realized. The data of the data set containing 10000 samples taken from a private payment systems company was examined.

The dataset was subjected to deep learning with LSTM, ARIMA and PROPHET algorithms. The python environment was used while developing the application. As a result of the study, it was seen that the LSTM method was more successful than the ARIMA and PROPHET methods.

İÇİNDEKİLER

Sayfa No.

TABLOLAR LİSTESİ

Sayfa No.

ŞEKİLLER LİSTESİ

Sayfa No.

KISALTMALAR

ARIMA	:	Birleştirilmiş Otoregresif Hareketli Ortalama
ATM	:	Otomatik Vezne Makinesi
BKM	:	Bankalararası Kart Merkezi
IMDB	:	Internet Movie Database
KNN	:	K-En Yakın Komşu
LSTM	:	Uzun Kısa Süreli Bellek
MAE	:	Ortalama Mutlak Hata
MAPE	:	Ortalama Mutlak Yüzde Hatası
ML	:	Makine Öğrenmesi
ÖKC	:	Ödeme Kaydedici Cihaz
POS	:	Satış Noktası
RMSE	:	Ortalama Karekök Sapma Hatası
XGB	:	Aşırı Gradyan Arttırma
YSA	:	Yapay Sinir Ağları

SÖZLÜK

Kredi Kartı: Tüketicilerin acil olan nakit gereksinimlerini karşılayan, alışverişlerde taksit olanağı sağlayan, kullandıkça para puan kazandırma durumu söz konusu olan, mal ve hizmet alımlarında kullanılan ve bunlardan yararlanılırken bankaya borçlanmalarının gerçekleşmesini sağlayan bir finansal araçtır.

Modelleme: Bağımlı ve bağımsız değişkenler arasındaki ilişki bulmak için model kurulur. Model kurulduktan sonra da tahminleme yapılır.

Ödeme Sistemleri: Finansal kuruluşlar aracılığıyla ödeme alınmasını ve tahsilat gerçekleştirilmesini sağlayan sistemlerin genel adıdır. Bu sistem, belirli standartlar ve yasal mevzuata göre belirlenerek dijitalleşmenin de etkisiyle alıcının ödeme yapmasındaki kanalların çeşitliliğini gösterir.

Tahminleme: Finans ve yatırımda kullanılan, bir bağımlı değişken ile bir dizi bağımsız değişken arasındaki ilişkinin gücünü ve niteliğini belirlemeye çalışan istatistiksel bir terimdir.

GİRİŞ

Günümüzde artan teknoloji ve yeni çıkan tüketim ürünlerinin gelişmesi sebebiyle insanlar harcamalarını kolay ve hızlı bir şekilde yapmak istemektedirler. Bu sebepten dolayı kredi kartları, sıklıkla kullanılan kartlı ödeme araçlarının en başında gelmektedir. Teknolojik altyapının hızla gelişmesi, kredi kartları ile verilen hizmetlerin çeşitlendirilmesine, aynı zamanda kalitelerinin artırılmasına önemli katkılarda bulunmuştur. Bireysel bankacılıkla ilgili yatırımlara hız verilmesi bu artışın en önemli nedenlerinden olmuştur (Girginer ve ark., 2008).

Kredi kartları, finansal kuruluşlar tarafından temin edilen anlaşmalı kurum ya da kuruluşlarda bulunan POS, ATM ve ÖKC üzerinden harcama yapabilen ödeme araçlarıdır. Harcamaların; kredi kartlarının alındığı finansal kuruluşlar üzerinden belirledikleri limitler dâhilinde, nakit avans çekimi; harcama-taksitlendirme; harcama-erteleme gibi işlemler ile alışverişler yapmalarına olanak sağlayan fiziksel ya da sanal olarak temin edilebilen ödeme aracıdır. Kredi kartı kavramı olarak adlandırılan bu ödeme yönteminin ülkemizde ilk kullanımları 1968 yılında gerçekleşmiştir. 1980'li yıllarda küresel otomasyon alandaki çalışmaların gelişmesiyle üzerine dikkat çekmeye başlamıştır. Özellikle 1999 yılından itibaren bankaların hizmet portföyleri içinde sayılır bir paya sahiptir (Çeker, 1992; Çavuş, 2006).

Son zamanlarda kredi kartı veren kurum ve kuruluşların da sayısı oldukça hızlı artmakta olup aynı zamanda kredi kartına farklı özellikler de getirilmektedir. Bunun en büyük nedenlerinden biri nakit taşıma ihtiyacının ortadan kaldırılması, yüksek harcamalarda taksitlendirme veya erteleme gibi seçeneklere sahip olmamız ve aynı zamanda yapılan harcamaların sonunda geri alınan cashback ya da bonus gibi kazanlarımızın olmasıdır.

2021 yılında toplam kredi kartı sayısı 83791396, Ocak 2022 itibariyle artış yaşayarak sayı 86209419 olmuştur (BKM, 2022). Banka müşterileri, kazançlarının artması ile alışveriş piyasasında sıkça bulunmak istemektedir.

Bankalar yeni müşteriler kazanabilmek ve mevcut müşterileri koruyabilmek için farklı kampanyalar ve özellikler sunmaktadır. Bankaların sunmuş olduğu teklifler, kredi kartı kullanıcılarının kredi kartları üzerinde yapılan harcama isteğini arttırmaktadır. Bundan dolayı kredi kartları sayısında gün geçtikçe artış devam etmektedir.

Aşağıda Dünya'da banka kartı, ön ödemeli kart ve kredi kartı sağlayan finansal kuruluşlar verilmiştir;

- American Express- Israacard
- Bankcard- JCB
- Card Reciprocal Agreements- Mastercard
- Cardholder Information Security Program- Meeza
- Carte Bleue- PayPak
- Cashplus- Rakuten Card
- Diners Club International- RuPay
- Discover Financial- The Everything Card
- Elo- Troy
- Eurocard- UnionPay
- Europay International- Universal Air Travel Plan
- European Payments Initiative- Visa
- Yukarıda yer alan kuruluşlar arasında;
- American Express
- Mastercard
- Troy
- Visa

Türkiye'de faaliyet gösteren bankalar bu firmalar aracılığı ile müşterilerine kredi kartları vermektedir.

Sektörde sıkça yer edinmek isteyen bankalar, yeni alanlarda öncü olmayı hedeflemektedir. İleriki dönemlerde çıkartmak istedikleri yeni ürünler için olabilecek harcama tahminlerini elde etmek istemektedirler. Böylelikle hem olabilecek problemlere karşı önlem almak hem de piyasada öncü olma hedefleri için önemlidir.

Bu çalışmada, finans sektöründeki resmi ve özel olan iki farklı bankadan elde edilen son iki yıllık dönemdeki gerçek veriler ile gözetimli makine öğrenmesi yöntemleri kullanarak gelecek dönem harcamaları tahmin edilecektir. Çalışmanın sonucunda en yüksek performans gösteren algoritmanın bulunması amaçlanmıştır.

Çalışmasının birinci bölümünde giriş yapılmıştır. İkinci bölümde farklı kaynaklar üzerinden benzer çalışmaların incelenmesi ve araştırılması yapılmıştır. Üçüncü bölümde ise gözetimli makine öğrenmesi yöntemleri anlatılmıştır. Dördüncü bölümde veri seti ve algoritmaların uygulanması anlatılmıştır. Beşinci bölümde ise uygulanan algoritmanın sonuçları özetlenmiş ve çalışmada elde edilen sonuçlar anlatılmıştır.

BİRİNCİ BÖLÜM
BENZER ÇALIŞMALAR

Tezin bu kısmında, finansal tahminlemeler için kullanılan yöntemler ve benzer çalışmalar araştırılmış ve başarı sonuçları değerlendirilmiştir.

Gökçay (2020) yapmış olduğu çalışmada, bir özel bankaya ait iki farklı ATM verileri ile günlük olarak çekilecek olan para miktarını tahmin eden bir sistem hazırlamıştır. Tahminleme için veri setinde 2 yıllık dönemi kapsayan gerçek veriler kullanılmış olup maaş günleri, hafta içi ve hafta sonu günleri gibi zaman serileri eklenmiştir. Çalışmasında Yapay Sinir Ağları, Uzun Kısa Bellekli Tahminleme ve Bütünleşik Otoregresif Hareketli Ortalama yöntemlerini kullanılmıştır. Çalışma sonucunda iki farklı ATM üzerinden farklı sonuçlar elde edilmiş ve Uzun Kısa Bellekli tahminleme yönteminin daha başarılı olduğunu gözlemlenmiştir. MAE hata değeri 13.755 olarak tespit edilmiştir.

Demiray ve Gürhanlı (2020) yapmış oldukları çalışmada, Microsoft şirketine ait 2014 ve 2019 yılları arasındaki veriler ile günlük hisse senetlerinin kapanış fiyatlarının tahmininde bulunmuşlardır. Tahminleme için Polinomsal Regresyon, Bütünleşik Otoregresif Hareketli Ortalama, Aşırı Gradyan Güçlendirme, Uzun Kısa Bellekli Tahminleme ve Facebook Prophet yöntemlerini kullanmışlardır. Tahminleme yaparken kısa vade için 5 günlük, 10 günlük ve 20 günlük; uzun vade için ise 1 yıllık veriler kullanmışlardır. 5 ve 10 günlük tahminlemeler de hata oranının yüksek, 20 günlük verilerde ise ortalama hatanın daha düşük olduğu gözlemlenmiştir. Gerçek verilere en yakın tahminleme sonuçlarını Uzun Kısa Bellekli Tahminleme ve Bütünleşik Otoregresif Hareketli Ortalama yöntemleri ile yakalamışlardır. Aşırı Gradyan Güçlendirme yönteminde ise en düşük başarıyı elde etmiştir.

Koç (2021) yapmış olduğu çalışmasında, orta vadede hisse senetlerinin tahmini için finans sektöründeki Türkiye İş Bankası A.Ş. ve ulaşım sektöründeki Pegasus Hava Yolları'na ait 5 yıllık veriler üzerinden LSTM ve Sinir Ağları yöntemleri kullanarak tahminlemeler yapmıştır. Uzun Kısa Bellekli Tahminleme yöntemi ile Türkiye İş Bankası A.Ş. ve Pegasus Hava Yolları için zıt yönlü tahminlemeler çıkarmışlardır. Pegasus Hava Yolları için başarı oranı RMSE değerini 60, gizli birimde ise oranın %80-20 arasında olduğunu tespit etmişlerdir. Türkiye İş Bankası A.Ş. için başarı oranı RMSE değerini 100, gizli birimde %90-100 arasında olduğunu tespit etmişlerdir. Çalışma sonucunda Uzun Kısa Bellekli Tahminleme yönteminin finans sektöründe yüksek başarılı sonuçlar verdiği, ulaşım sektöründe ise daha az başarılı sonuçlar verildiği tespit etmişlerdir.

Kaya (2019) hazırlamış olduğu çalışmada, kredi kartı harcamalarına dayalı alışkanlıkları incelerek uygun olan kampanya yöntemini belirlemedeki etkiyi bulmayı hedeflemiştir. Kaya çalışmasında 4497 müşteri üzerinden 3 aylık kredi kartı harcama davranışları incelenmiştir. Turi Create aracılığıyla oluşturduğu veri setlerini, benzerlik modeli üzerinden Pearson korelasyonu ve Kosinüs benzerliğine uygulanmıştır. Pearson Korelasyonunun daha başarılı sonuçlar sağladığı görülmüştür.

Karagöz (2020) yapmış olduğu çalışmada, farklı sektörler içerisinden seçmiş olduğu firmaların kapanış fiyatlarını tahmin etmeye çalışmıştır. Çalışmasında 2 yıllık verileri günlük ve pay bazlı olarak ayrıştırmıştır. 15 farklı veri seti hazırlamış olup içerisinde altın, döviz ve tahvil endeksi faiz bilgilerine de yer verilmiştir. Makine öğrenmesi algoritmaları içerisinden; Karar Ağacı Regresyonu, Sinir Ağı Regresyonu, Bayes Ağı Regresyonu, Doğrusal Regresyon, Poisson Regresyonu ve Karar Ormanı Regresyonunu kullanmıştır. Karagöz çalışması için hazırladığı veri setinin %70 eğitim, %30'luk kısmını ise test için kullanmıştır. Karar Ağacı Regresyonunda %0.00005, Doğrusal Regresyonda en iyi değer %99.9914, Sinir Ağı Regresyonunda %99.9893, Bayes Ağı Regresyonunda %99.9487 ve Possion Regresyonunda ise %86.4988 sonuçlarını elde etmiştir. Sonuç olarak ise pay bazındaki tahminler birden fazla pay içeren tahminlemelere göre daha fazla başarı göstermiştir.

Gençalp (2020) çalışmasında ATM nakit çekme miktarını makine öğrenmesi yöntemleri ile tahminlemeye çalışmıştır. Veri setinde 1 yıllık gerçek verileri kullanmış ve datasında; işlem tarihi, hafta içi, hafta sonu, resmî tatiller, dini tatiller ve festival gibi değişkenleri ekleyerek ilerlemiştir. Çalışması için Rastgele Orman Regresyonu, Gradyan Güçlendirme, Destek Vektör Regresyonu ve K En Yakın Komşu algoritmaları kullanmıştır. Çalışma sonucunda KNN algoritması ile ham veriler üzerinden MAPE %27 ile en iyi sonucu elde etmiştir.

Çelikel (2018) çalışmasında, makine öğrenmesi ve veri madenciliği ile hisse senetlerinin kapanış fiyatlarını tahminlemeye yönelik çalışma yapmıştır. Çalışmasında Çelikel, twitter üzerinden elde ettiği son 6 aylık veriler ile borsa tahminlemesi yapmıştır. Pegasus Hava Yolları ve Türk Hava Yolları şirketleri için twitter üzerindeki elde edilen tweetler ile borsalar arasındaki değer korelasyonlarını incelemiştir. Tahminleme için hazırlanan veri setinin %60'ı eğitim için kullanmıştır. Elde etmiş olduğu verileri negatif, pozitif ve nötr olacak şekilde duygu analizi ile ayrıştırılmasını yapılmıştır. Tahminleme için günlük ve aylık olacak şekilde Çoklu Doğrusal Regresyon, Karar Ağacı Regresyonu, Rastgele Orman Regresyonu ve Destek Vektör Algoritmalarını kullanmıştır. Çalışmanın sonucunda, Destek Vektör Algoritması için doğruluk %93,92, Karar Ağacı için doğruluk %91,41, Rastgele Orman için %91,37, Doğrusal Çoklu Regresyon için ise doğruluk %95,98 değerinde olmuştur.

Hasan (2020) yapmış olduğu çalışmada hem derin öğrenme yöntemlerini hem de makine öğrenmesi yöntemlerini kullanarak borsa endeks yönünü tahmin etmeye çalışmıştır. Çalışmasında kullanmış olduğu sınıflandırma teknikleri ise Derin Sinir Ağları, Destek Vektör Makinesi, Rastgele Orman ve Lojistik Regresyon'dur. Veri setinde kullanmış olduğu veriler Borsa İstanbul'un 2008 ve 2016 yıllarına ait verilerden oluşmaktadır. Hasan çalışmasında tahminleme yaparken günlük, aylık ve yıllık olacak şekilde 3 farklı zaman periyodlarını tercih etmiştir. Çalışmasının sonucunda, Derin Sinir Ağları'nın daha iyi performans sağladığını tezinde savunmuştur.

Aydın (2019) yapmış olduğu çalışmada, tahminlemelerde en çok kullanılan yöntemlerden farklı olarak Uyarlamalı Sinirsel Bulanık Çıkarım Sistemi'ni ve K Ortalama Kümeleme algoritmasını kullanmıştır. Veri setinin 30%'unu test için %70'ini ise eğitim için ayırmıştır. Yapılan çalışmada veriler üzerinde özellik ölçeklendirme uygulanmıştır. Aydın çalışmasında, K Ortalama Kümeleme yönteminin daha başarılı sonuçlar sağladığını ve hata oranının daha düşük olduğunu ortaya koymuştur.

Nacar ve Erdebilli (2021) yapmış oldukları çalışmada, makine öğrenmesi ve metin madenciliği ile satış tahminlemesini ele almıştır. Çalışmada 20 aylık veri seti içerisinde farklı veri tiplerine ait değişkenlerde yer almaktadır. Doğrusal Regresyon, Rastgele Orman, K En Yakın Komşu, Ridge, Lasso ve ElasticNet yöntemleri kullanılmışlardır. Elde edilen sonuçlar arasında en iyi tahminlemenin %83,9980 ile Rastgele Orman'a ait olduğunu tespit etmiştirler.

Selimoğlu ve Yılmaz (2021) yapmış oldukları çalışmada, kredi kartı ile yapılan normalin dışındaki harcamaları Çok Katmanlı Yapay Sinir Ağları ve Naive Bayes ile tahminlemeye çalışmıştır. Çalışmada kullanılan veriler Kaggle veri tabanından elde edilmiş ve 2013 yılına aittir. Veri setinin %70'lik kısmı eğitim için kullanılmıştırlar. KNIME programı ile elde edilen tahminleme sonuçlarında Çok Katmanlı Yapay Sinir Ağları modeliyle %99.943 değeri ile sağlamıştırlar. Naive Bayes modelinde ise %98.207 değeri elde edilmiştir.

Namlı ve Özcan (2017) yapmış oldukları çalışmada, Sinema sektöründeki yeni gelişmelerin fiyat artışını peşinde getirmelerine bağlı olarak yapılacak olan yeni yapıtların sonuçlarında hasılatın tahminlemesini amaçlamışlardır. Çalışmalarında, Yapay Sinir Ağları, Rastgele Orman, Karar Ağacı ve Destek Vektör Makinesi Algoritmalarını kullanmışlardır. Veri setinde 12 farklı özellik kullanmış olup veri seti IMDB'e ait bir web sitesi üzerinden alınmıştır. Yapılan tahminleme de en iyi sonuç %91,6166 ile Rastgele Orman Algoritmasına aittir.

Ustalı ve ark. (2021) hazırlamış oldukları çalışmada, 9 yıllık bir dönemde ve içlerinde Arçelik, Aselsan ve Sabancı Holding'in de bulunduğu 22 farklı firma üzerinden BIST30 endeksi firmalarının hisse senetlerinin kapanış fiyatları tahminleme çalışmaları yapmışlardır. Veri setinin %70'i eğitim için kullanmış olup Aşırı Gradyan Güçlendirme, Yapay Sinir Ağları ve Rastgele Orman Algoritması yöntemlerini tahminlemeler için kullanmışlardır. Öncelikle olarak finansal tablolar üzerinden finansal hesaplamaları ve sonrasında ise bu hesaplamaların 3 aylık ortalamalarını bulmuşlardır. Yapılan çalışmada kullanılmış olan 3 farklı algoritmada en başarılı sonucu Aşırı Gradyan Güçlendirme ile elde etmiştirler. Yapay Sinir Ağlarında ise tek ve çift mimari yapısının kullanılmış olmasına rağmen başarı oranını düşük olarak tespit etmiştirler.

Akay ve ark. (2019) yapmış oldukları çalışmada, konut fiyat artışlarını tahminleme de Bütünleşik Otoregresif Hareketli Ortalama, Rastgele Orman ve Hibrit Rastgele Orman yöntemlerini kullanmışlardır. Çalışmalarında doğrusal olan ve doğrusal olmayan yöntemleri kıyaslayarak en iyi başarı oranını veren algoritmayı tespit etmeye çalışmışlardır. Veri seti Türkiye Cumhuriyet Merkez Bankası tarafından temin edilmiş olup uzun bir dönemi içermektedir. Rastgele Orman yöntemi ile elde ettikleri model ile en başarılı performansını MAE %0.140 ile elde etmişlerdir.

Yiğit ve ark. (2021) yapmış oldukları çalışmada, petrol fiyatlarının tahmini için 8267 örnekli bir veri seti üzerinden günlük petrol fiyatlarını almışlardır. Çalışmalarında Uzun-Kısa Süreli Bellekli Tahminleme ve Bütünleşik Otoregresif Hareketli Ortalama yöntemlerini kullanarak ileriye dönük 180 günlük tahminle yapılmışlardır. Araştırmalarında iyi sonuçların elde edilmesi için veri setinin %80'lik kısmı eğitim için kullanılmış olup en iyi başarıyı Uzun-Kısa Süreli Bellekli Tahminleme yönteminde MAE %0.886 ile %99 oranı ile yakalamışlardır.

Kocabaş ve ark. (2019) yapmış oldukları çalışmada, kripto paralara olan talep sonrasında gelecek fiyat tahminlemesini gerçeğe en yakın şekilde yapılabilmesi için bir çalışma gerçekleştirmişlerdir. Uzun-Kısa Süreli Bellekli Tahminleme ve Yapay Sinir Ağları yöntemlerini çalışmalarında kullanmışlar ve sonrasında ise elde edilen sonuçlar ile kıyaslama yapmışlardır. Çalışma sonucunda MAE ve RMSE değerleri ile kıyaslama yapılmıştır. Yapay Sinir Ağlarının Uzun-Kısa Süreli Bellekli tahminleme yöntemine göre daha iyi sonuçlar vermiş, gerçek verilere daha yakın tahminlemeler yapabildiğini gözlemlemişlerdir.

Sevli ve Gülsoy (2020) yapmış oldukları çalışmada, makine öğrenmesi modellerinden olan Facebook Prophet modelini kullanarak 2019 yılında tüm dünya üzerinden yaygınlaşan Covid 19 vakasının tahminlemesi için çalışmıştır. Çalışmalarında 3 aylık dönemi içeren ocak ve mart ayları verileri kullanılmış olup veriler haftalık olacak şekilde tahminlemelerde kullanılmıştır. Çalışmada Python programlama dili tercih edilmiştir. Çalışmasının %20'lik kısmı test için kullanılırken kalan %80'lik kısım ise eğitim için kullanılmıştır. Bu çalışmada hata değeri olarak MAPE (Mutlak hata metriği) dikkate alınmıştır. Çalışmanın sonucunda elde edilen değerler gerçek veriler ile karşılaştırılarak ilerlenmiş olup gerçek veriler ile tahminlemeler yapılan verilerin uyumluluğunun büyük ölçüde olduğu gözlemlenmiştir. Bu çalışmada kullanılan Prophet algoritmasının başarılı sonuçlar verdiği gözlemlenmiştir.

Akdağ ve Bozma (2021) hazırlamış oldukları çalışmalarında Prophet yöntemini kullanarak bitcoin piyasa fiyatlarını tahminlemeye çalışmışlardır. 7 yıllık dönemi içeren veriler alınarak toplamda 2804 veri çalışmada kullanılmıştır. Çalışmada stok akış oranı bulunularak karşılaştırmalar yapılmıştır. Çalışmasının sonucunda RMSE değeri 0,1551, MAE değeri 0,1138 ve MAPE değeri ise 0,015 olarak sonuçlar verdiği gözlemlenmiş, en iyi MAE değerinin stok akış modeli üzerinde daha başarılı sonuçları getirdiği görülmüştür.

Yurttabir ve Şen (2021) yaptıkları çalışmalarında, borsa da işlem gören 173 şirket üzerinden 11 yıllık bir dönem üzerinden veriler alınarak kâr-zarar tahminlemesinin yapılmadı için Prophet makine öğrenmesi algoritmalarını kullanmışlardır. Veri setinde cari oran, kâr marjı, hasılat gibi birçok farklı değişken kullanılmıştır. 48 dönemi içeren verilerin 46'sı eğitim için kullanılırken 2 dönemi test için verilmiştir. Çalışmada sonuçların karşılaştırılması için RMSE, MAPE, MAE değerleri dikkate alınmıştır. Çalışmada kullanılan iki dönemlik test verisinde hata metrikleri üzerinde en iyi sonucu MSE, 0,0185 değerine sahiptir. Çalışmasının sonucunda 0 değerine yakınlık başarı oranını gösterirken 0'dan uzaklaşan değerler başarısız tahminleri göstermektedir. Bu çalışmada başarısız sonuçlarda gözlemlenmiştir.

İKİNCİ BÖLÜM
YÖNTEM VE MATERYALLER

2.1. Makine Öğrenmesine Genel Bakış

Makine öğrenmesine yönelik ilk kavramlar 1950 yılların başında ortaya çıkmaya başlamıştır. Makine öğrenmesi geçmiş veriler üzerinden öğrenim sağlayarak geleceğe dair öngörüler sunarak tahminlemeler yapabilen yapay zekâ bilimidir.

Makine öğrenimi, çeşitli algoritmalar kullanarak yenilenen veriler üzerinden öğrenim sağlar. Makinelerin, matematiksel ve istatiksel yeni çıkarımlarda bulunmasını, tahminlemeler yapmasını sağlar ve sonrasında bu çıkarımlara bağlı sonuçlar üretilmesini hedefler (Hurwitz ve Kirsch, 2018).

Makine öğrenmesi, insanlar gibi geçmişteki verilerden öğrenen ve öğrenmiş olduğu bilgilerden çıkarımlar yaparak geleceğe dair çözümler üretebilen, tahminler yapabilen insan zekâsı ürünlerinden biridir. Makine öğrenmesi zekâ kazanılmasına dayalı bir kuramdır (Ereken ve Tarhan, 2018).

Her gün gelişmekte olan makine öğrenmesinin; maliyet azaltma, veri bütünlüğü sağlama, kullanıcı deneyimi elde etme, risk azaltma, yenilikler oluşturma gibi birçok ileriye dönük avantajları bulunmaktadır (Microsoft, 2022).

2.1.1. Yapay Zekâ

Gerçek anlamıyla "Zekâ", dışarıdan gelen tüm uyarıların algılanarak kullanılmak üzere bilgiye dönüştürülmesidir. Zekâ üzerine birçok yaklaşımlarda bulunulmuş ve tanımlamalar yapılmış olmasına karşılık zekâ, birden çok ürünün şekillendirilerek çözümlendirilmesidir. Yapay zekâ, insanların zekâ yapısına benzer şekilde programlar ve makineler yapma bilimi olarak ifade edilmiştir. Akıl yürütme, olgulardan anlam çıkarma ve genelleme yapabilen bilinçsel becerilerin kullanılmasıdır (Arslan, 2020).

Yapay zekâ kavramına dair ilk temeller 1950'li yılların başında M. Turing tarafından "Makineler düşünebilir mi?" soru ile atılmıştır. Turing zekânın test edilebilen bir kuram olduğunu "Turing Testi" ile ortaya koymuştur (Ereken ve Tarhan, 2018).

Yapay zekânın kullanım alanı her geçen gün genişlemekte ve savunma, sanayi, hava araçları, eğitim, tıp ve eğlence gibi birçok farklı alanda yaygın olarak kullanılmaktadır.

Yapay zekâ üzerine yapılan tanımlamalardan en çok literatürde karşılaşılanlar aşağıdaki gibidir;

- Yapay zekâ, insan tarafından yaptığında zeki olarak adlandırılan davranışların makine tarafından yapılmasıdır.
- Yapay zekânın amacı, insan zekasını bilgisayar aracılıyla taklit etmektir.
- Düşünme, anlama, faaliyete geçirmeyi sağlayacak bilgi işlemle çalışmasıdır."

Makineleri daha işlevli daha akıllı, daha kullanılabilir ve daha faydalı olarak tüm alanları kapsayacak şekilde kullanılması yapay zekanın ilk amaçları arasında sıralanabilir.

Yapay zekanın önemini Fredkin (1992) şu şekilde vurgulamıştır: "Tarihte üç büyük olay vardır. Bunlardan ilki kâinatın oluşumudur. İkincisi yaşamın başlangıcının olmasıdır. Üçüncüsü ile yapay zekanın ortaya çıkışıdır."

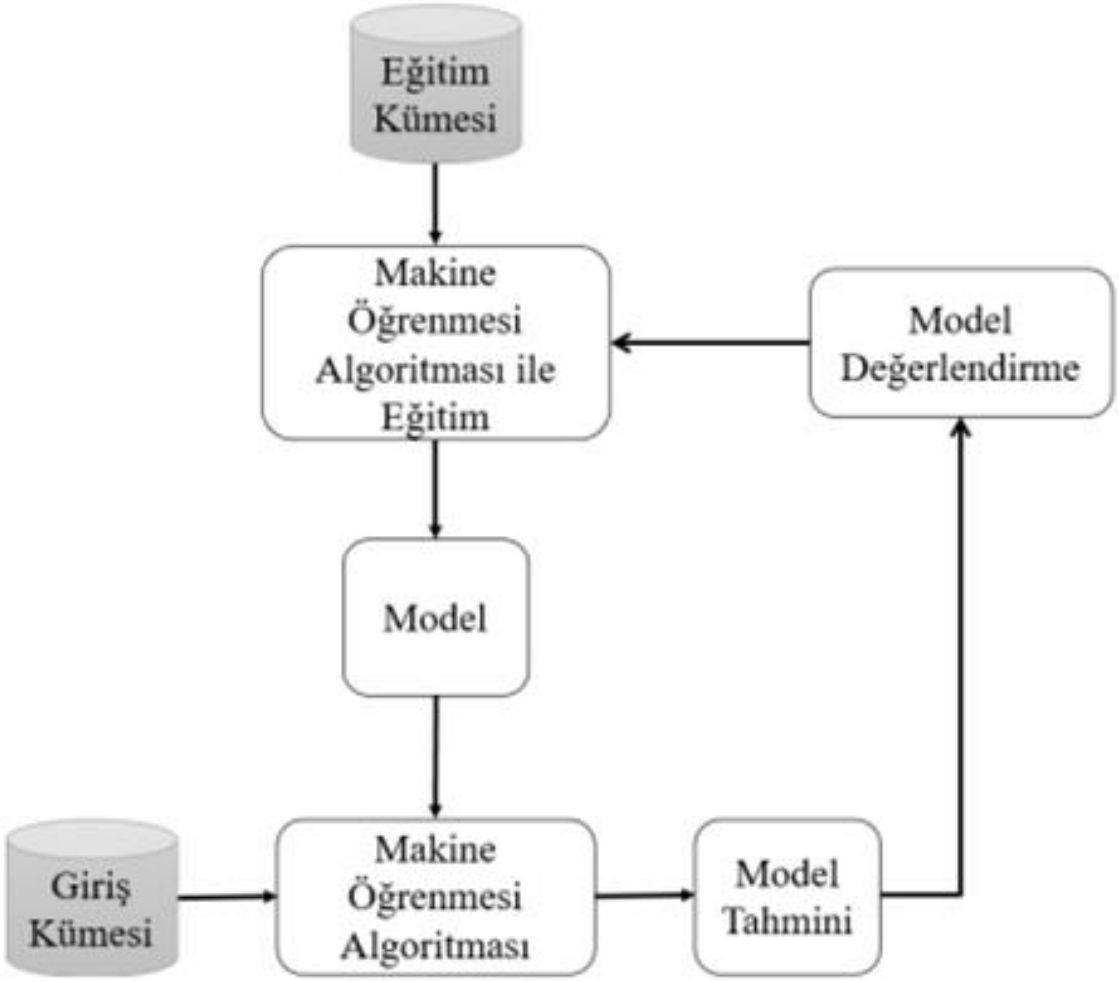
Yapay zekâ, tahminlemelerin dışında kümeleme ve sınıflandırma için de kullanılmaktadır. Yapay sinir ağları, mantık, genetik algoritmalar ve bulanık mantık yapay zekâ teknolojilerinden bazılarıdır. Makine öğrenmesi, insanlar gibi davranan bilgilerden çıkarımlar yapan bilgisayar sistemidir (Atalay ve Çelik, 2017).

2.1.2. Makine Öğrenmesi ve Yöntemleri

Makine öğrenmesi kavramının daha anlaşılır olabilmesi için, öğrenme kavramının ne olduğunun bilinmesi gerekir. Öğrenme, zaman içerisinde yeni bilgilerin keşfedilmesiyle davranışların iyileşme sürecidir (Simon, 1993).

Bu öğrenme işini bilgisayar üslendiği zaman makine öğrenmesi kavramı ortaya çıkmaktadır. Makine öğrenmesi, geçmiş ve yeni elde edilen deneyimlerden öğrenerek mevcut olanı daha da iyileştirmek, çözümleyebilmek kararlar verebilmek için modellemeler kullanmasıdır (Pulat ve Kocakoç, 2021).

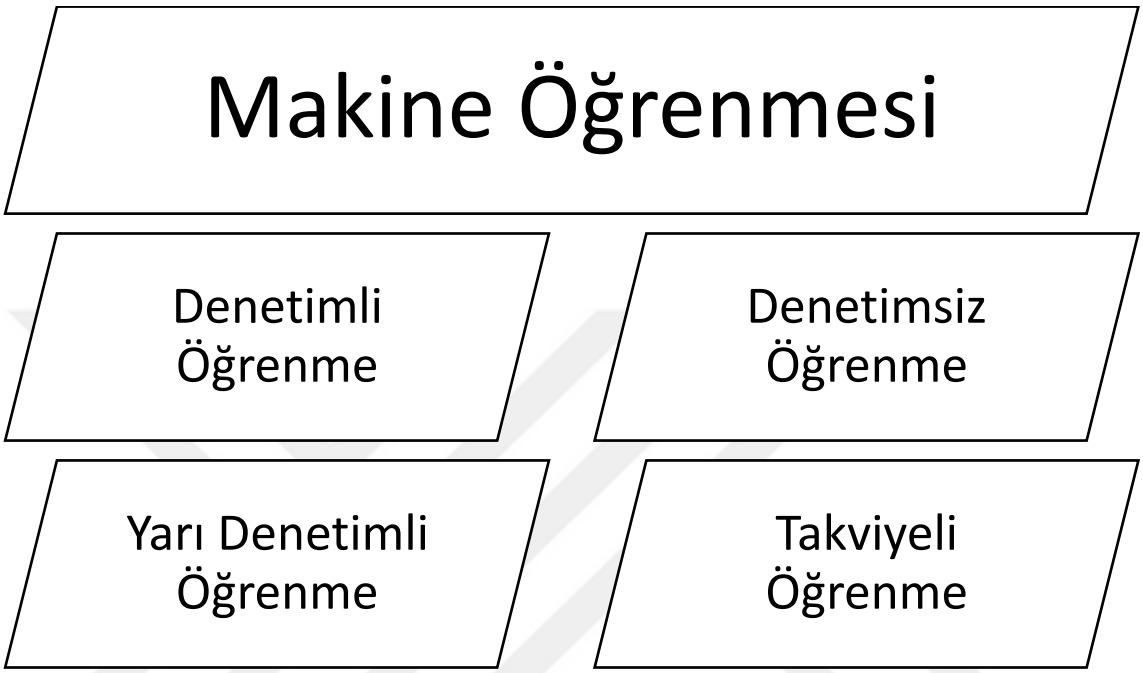
Şekil 1'de Makine öğrenmesine ait akış süreci belirtilmiştir. Yukarıda yer alan ifadelere göre genel bir tanımda bulunmak istenildiğinde makine öğrenmesi; farklı modeller kullanarak insanın öğrenme biçimi taklit ederek tahminleme yapabilen, çözüm üretebilin bilgisayar programı ve yapay zekânın alt bilimidir.



Şekil 1 Makine Öğrenmesi Genel Akış Şeması

Arthur Lee Samuel tarafından 1959 yılında makine öğrenmesine dayalı ilk kavramlar dama oynayan programı tasarlamasıyla oluşmaya başlamıştır. Bu oyun ile makine öğrenmesinin tanımları oluşmaya başlamıştır, dama hatalarından öğrenerek yeni tecrübeleriyle kendini geliştirmesi üzerine yapılmıştır (Ereken ve Tarhan, 2018).

Makine öğrenmesi performansını iyileştirmek, tahmin ve analizleri başarılı kılmak için denetimli öğrenme, denetimsiz öğrenme, yarı denetimli öğrenme ve pekiştirmeli öğrenme modelleri kullanılır. Şekil 2'de Makine Öğrenmesi yöntemleri gösterilmiştir.

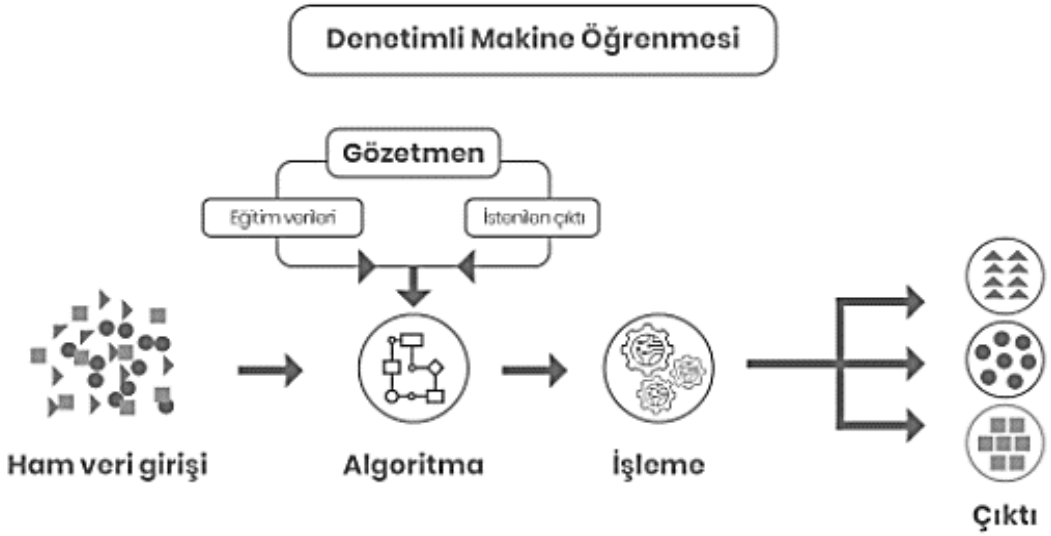


Şekil 2 Makine Öğrenmesi Yöntemleri

2.1.2.1. Denetimli Öğrenme

Denetimli öğrenme, sistemin öğrenebilmesi için girdi ve çıktı değerlerinin birlikte verildiği makine öğrenmesinin bir alt dalıdır. Çıkış değerlerini tahmin etmeye çalışmak için giriş değerleri ile eğitilir. Sonuçları bilinmeyen veriler için, girdi olarak verilen ve sonuçları bilinen veriler üzerinden sınıflandırma ve kümeleme çalışması yapılır (Kara ve Şamlı, 2021).

Bu yöntem, çıktı verilerini girdi verilerine karşı tahmin ederek sonuçlara ulaşmayı amaçlar. Denetimli öğrenme, regresyon ve sınıflandırma yöntemlerini kullanır. Regresyon mevcut olan özellikler üzerinden verinin farklı özelliklerinin tahminlemesini yapar, sınıflandırma ise tanımlanan veri seti üzerinden belirtilen özelliklerin gruplandırılmasını sağlar (Çelik, 2018). Şekil 3'te denetimli öğrenmeye ait akış verilmiştir.



Şekil 3 Denetimli Makine Öğrenmesi Akışı

Denetimli öğrenmede en çok kullanılan yöntemler;

- Lineer Regresyon
- Lojistik Regresyon,
- Rastgele Orman,
- K En Yakın Komşu
- Navie Bayes
- Karar Ağaçları
- Sinir Ağları

şeklindedir.

2.1.2.2. Denetimsiz Öğrenme

Denetimsiz öğrenme, denetimli makine öğrenmesinden farklı olarak çıktı verilerini almaz. Bu yöntemde eğitim verisi kullanılmaz. Verilen veri seti üzerinden ilişkiler ve bağlantılar kullanarak tahminelemeler yapar ve çıktıya ulaşır. Bu öğrenme yönteminde kullanıcının sistem üzerinde hiçbir kontrolü yoktur ve sistem keşifler yapar, daha sonra ilişkilendirir ve sonuçlara ulaşır. Denetimsiz öğrenmede, veri kümeleri algoritmalar tarafından tanımlanır ve ilişkilendirilir. Bir sistemdeki veri değerlendirme sayısı arttıkça karar verme ve başarı gücü de artar. Denetimsiz öğrenme yöntemlerinde sınıflandırma ve ilişkilendirme algoritmaları kullanılabilir.

Sınıflandırma, verileri benzerliğe göre gruplandırır. İlişkilendirme ise tek bir veri kümesindeki verilerin benzer özelliklerinin ortak yönlerinin belirlenmesidir (Çelik, 2018).

Denetimsiz öğrenmede en çok kullanılan yöntemler;

- K-Kümeleme
- Temel Bileşen Analizi

Şeklindedir.

2.1.2.3. Yarı Denetimli Öğrenme

Yarı denetimli öğrenme, az miktarda işlenmiş veri ile büyük miktardaki ham veriyi tahmin eder. Veriler hakkında bilgi sağlamak için bir sınıflandırma gerçekleştirir. Sistemin hatalarından ders alarak daha iyi sonuçlar vermesi sağlanır. Denetimli öğrenme, tahmin edilecek eğitim verilerinin miktarını azaltırken, yarı denetimli öğrenme hafife alınacak ham veri miktarını artırır. Bu yöntem sıklıkla tavsiye edilir.

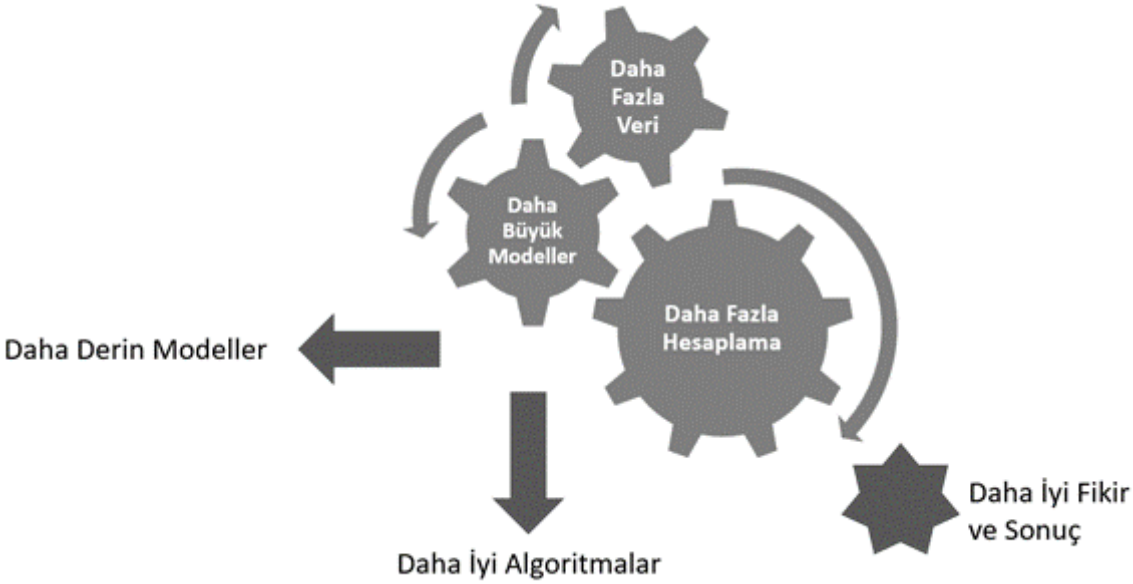
Yarı denetimli makine öğrenmesi yönteminde verilerin durumuna bağlı olarak üç farklı öğrenme durumu mevcuttur. Bunlar; ortak eğitim, kendi kendine eğitim ve aktif eğitimdir (Köksoy, 2021).

2.1.2.4. Takviyeli Öğrenme

Bu öğrenme yöntemi bir ödül sistemini kullanır. Temsilciler, hedefe giden doğru yolda ödüllendirilecektir. Takviyeli öğrenme, başlangıç ve bitiş noktalarını bilir ancak asıl amaç en kısa yolları kullanarak bitiş noktasına ulaşmaktır. Sistem doğru adımları atarsa ödüllendirilir, yanlış adımlar atarsa cezalandırılır. Temsilci, eğitim hakkında geri bildirim sağlar. Bu bildirim, sistem ile veriler arasındaki iletişimi sağlar ve sistemin başarı oranını artırır. Bazı literatür de bu yönteme pekiştirmeli öğrenme de denir (Köksoy, 2021).

2.1.3. Derin Öğrenme ve Yöntemleri

Derin öğrenme, doğal dil işleme, metin okuma, konuşma tanıma ve duygu tanıma gibi karmaşık sorunları çözmek için çok katmanlı yapay sinir ağlarını kullanarak karmaşık ve büyük ölçekli yapılandırılmış verileri analiz etmek için kullanılan bir makinedir. Derin öğrenme, makine öğrenmesi yöntemlerine karşılık olarak kodlanmış kuralları kullanmaz ve metin, ses gibi veri simgelerinden otomatik olarak öğrenir (Umut ve ark., 2019). Şekil 4'te derin öğrenmeye genel bir bakış yapılmıştır.



Şekil 4 Derin Öğrenmeye Genel Bir Bakış

Derin öğrenme yöntemleri esnektir ve ham verileri analiz ederek öğrenmeyi sağlar. Verinin boyutuna bağlı olarak tahminlemelerin başarı oranı da yüksek olacaktır. Derin öğrenme, çok katmanlı mimarilerde ve çok boyutlu verilerle çalışma imkânı sunmaktadır. Karmaşık problemlerin çözümlenmesi için kullanılan yöntemlerde önemli bir yere sahiptir (Çalışkan ve Demir, 2022). Derin öğrenme yönteminde, problemdeki hatanın düzeltilmesi için basit komutlar kullanılarak sistemin öğrenme sürecini takip etmek yeterlidir. Zaman içerisinde yapay sinir ağları gelişerek karmaşık problemleri çözümleyebilmek için derin öğrenme yöntemlerine evirilmiştir (Umut ve ark., 2019).

Derin öğrenme, beynin yapısının bir örneği olan yapay sinir ağını taklit ederek nöronlardan girdi sinyalleri alıp bunları toplayıp işleyerek, çıktıya gönderip ve daha sonra boyut olarak daha fazla veriyi depolamak suretiyle gerçekleştirilecektir.

Eğitimde kullanılan veri setinin boyutu nedeniyle makine öğrenmesi yöntemleri yeterli olmayabilir. Ancak derin öğrenme yöntemleri, daha büyük verileri daha etkin bir şekilde işledikleri ve daha iyi sonuçlar ürettikleri için giderek daha fazla tercih edilmektedir (Şişmanoğlu ve ark. 2020).

Derin öğrenme modelleri, verilerden görevleri çıkarmak için sınıflandırma işlevlerini kullanarak kendilerini eğitir. Derin öğrenmenin amaçlarından biri de öğrenme düzeyini insan düzeyinden ayırarak yükseltmektir. 2006'da Hinton'un araştırması yapay sinir ağlarına yeni bir bakış açısı getirdi ve bu yaklaşıma "Derin Öğrenme" adı verildi (Doğan ve Türkoğlu, 2019).

Derin öğrenmede en çok kullanılan yöntemler;

- Derin Sinir Ağları
- Derin Oto Kodlayıcı
- Boltzmann Makinesi,
- Yinelemeli Sinir Ağları,
- Erişimli Sinir Ağları,
- Uzun Kısa Süreli Bellek

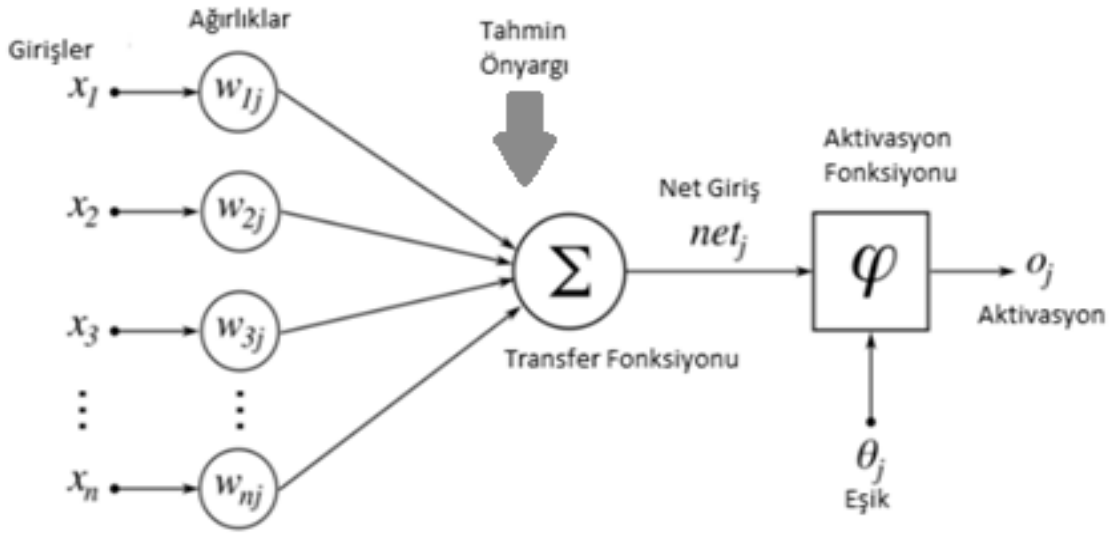
şeklindedir.

2.1.3.1. Yapay Sinir Ağları

İnsan beyninin özelliklerinden yararlanmak için araştırmacılar, beynin nörofiziksel yapısını örnek alıp kullanarak matematiksel sonuçlar çıkarmak için çok çalıştılar. Araştırmalarda birçok farklı yapay sinir ağı ve modeli geliştirilmiş ve üzerlerinde çalışılmıştır. Böylelikle mevcut hesaplama yöntemlerinden farklılaşarak yapay sinir ağlarına dayalı yeni kavramlar ortaya çıkmıştır (Ataseven, 2013).

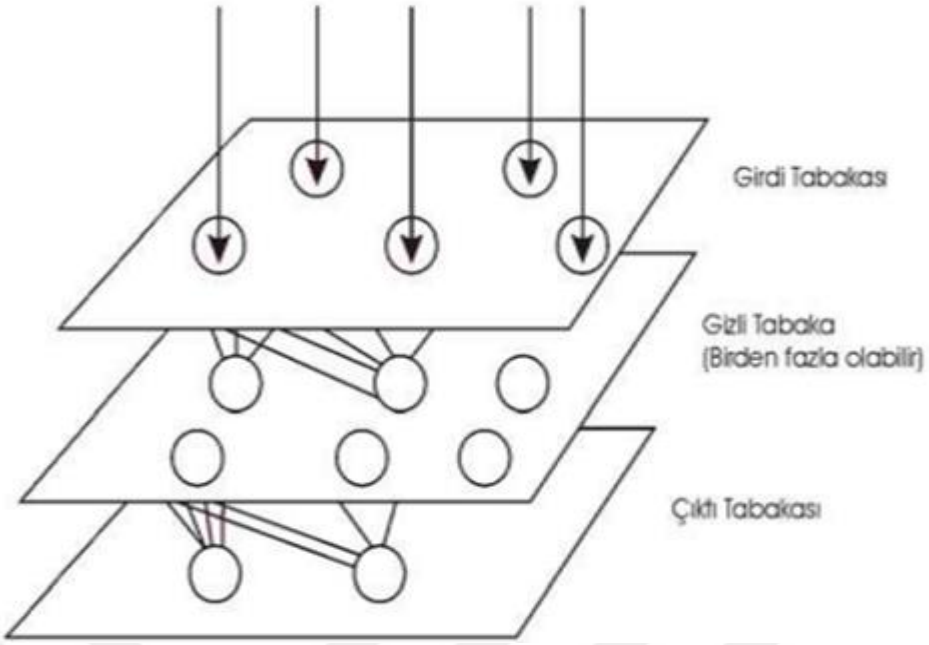
Yapay sinir ağlarının temelleri 1943 yılına dayanmaktadır. Sinir hekimi olan Warren McCulloch ve Matematikçi arkadaşı Walter Pitts “Sinir Aktivitesinde Düşüncelere Ait Bir Mantıksal Hesap” makalesinde yapay sinir ağları üzerinde düşünülmeye yöneltmiş ve temellerini atmıştır. Yapay sinir ağları farklı tabirlerle de ifade edilmektedir. Bunlardan bazıları; Nuromorfik ağlar, paralel dağıtılmış ağlar, bağlantılı ağlardır (Şahin ve Öztürk, 2018).

Yapay sinir ağları, temel olarak insan beyninin özelliklerini taklit ederek ve bu bilgilerden yeni bilgiler çıkararak öğrenilen bilgileri keşfedebilen, öğrenebilen ve depolayabilen bilgisayar sistemidir. Yapay sinir ağları nöronları üç unsurdan oluşur. Temel yapı taşları, toplayıcılar, aktivasyonlar, giriş sinapsı ağırlıklıdır. Ağırlığın çok fazla olması sonucu olumlu etkiler. Bir sonraki adım veri ağırlıklarının toplanmasıdır ve son adımı oluşturan sürecin sahibi aktivasyondur. Etkinleştirme, toplanan verilerin sonucunu bir işleve geçirerek çıktı üretir (Erdoğan ve Öztürk 2012). Şekil 5'te yapay sinir hücreleri gösterilmiştir.



Şekil 5 Yapay Sinir Hücreleri

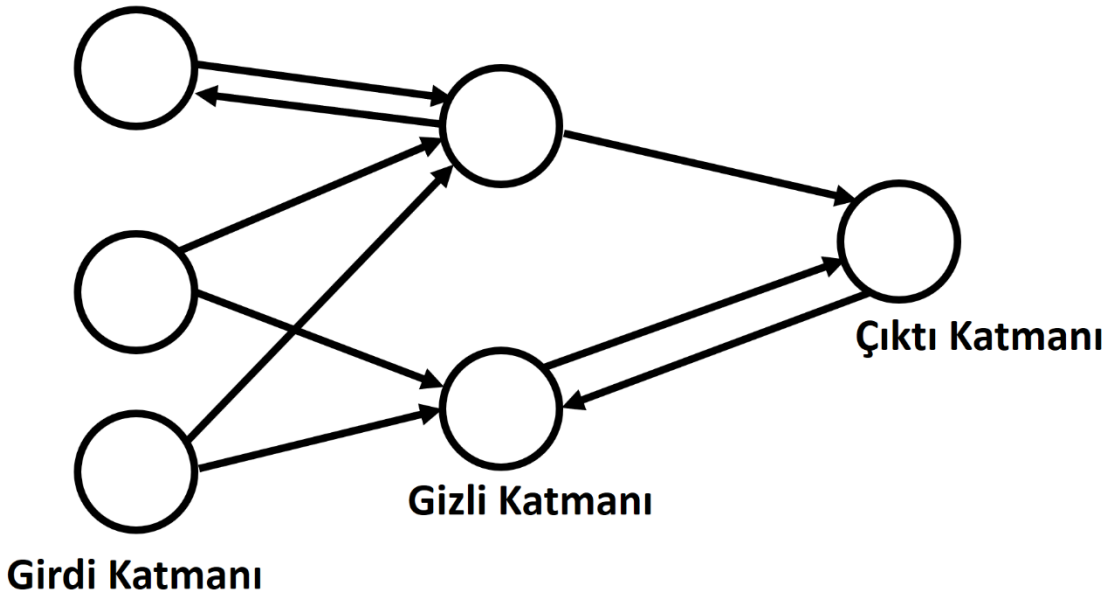
Yapay sinir ağlarının yapısında herhangi bir kısıtlama yoktur. Gizli katman sayısı az sayıda sinir ağı veriyorsa çözüm bulurken istenilen sonuç alınamayabilir. Bu nedenle gizli katmanda kaç nöron olduğu önemlidir. Gizli katmanda olması gereken nöron sayısını gösteren veya bunlarla ilgili hiçbir süreç testi yoktur (Ataseven, 2013). Şekil 6'da yapay sinir ağı genel yapısı verilmiştir.



Şekil 6 Yapay Sinir Ağı Genel Yapısı

Yapay sinir ağlarının en önemli özelliği öğrenme yetenekleridir. Öğrenme adımları denetimli öğrenme ve denetimsiz öğrenme olarak ikiye ayrılır. Çıktı değerleri eğitim için denetimli öğrenme girdi katmanına iletilir ve ortaya çıkan hata değeri beklenen değerle karşılaştırılır. Bu, sistemi öğrenmek ve sistemin öğrenme aşamasına ulaşmak için bir eğitim seti kullanan ve yaygın olarak kullanılan bir öğrenme yöntemidir. Denetimsiz öğrenmede çıkış değeri giriş katmanına iletilmez ve çıkış değeri sistem tarafından ayarlanmaktadır (Fırat ve Güngör, 2004).

Yapay sinir ağları; ileri beslemeli, geri beslemeli, çok katmanlı ve tek katmanlı olarak modellenmektedir. Tek katmanlı sinir ağları girdi ve çıktılardan oluşmaktadır ve doğrusaldırlar. Çok katmanlı sinir ağları birçok nöronun belirli bir üstünlük ile ilişki kurduğu doğrusal olmayan modeldir. İleri beslemeli sinir ağları girişten itibaren katmanların tertipli olarak çıkışa iletilmesi ile elde eden modellemelerdir ve ilerideki tabaka ile bağ kurmaktadırlar. Geri beslemede ise ileri beslemeliden değişik olarak önceki katmanla da bağ kurabilmesidir ve geri besleme modeli dinamik yapıdadır (Erdoğan ve Öztürk 2012). Şekil 7'de İleri beslemeli yapay sinir ağı modeli verilmiştir.



Şekil 7 İleri Beslemeli Yapay Sinir Ağı Modeli

İleri Beslemeli modeller; kümeleme, sınıflandırma, yorumlama, filtreleme ve tahmin YSA'nın en yaygın uygulamalarını temsil eder. Yapay sinir ağlarının kullanılmasındaki en büyük etken zor ve karmaşık problemlerin çözülmesidir. Yapay sinir ağları finans, tıp, enerji sistemleri ve sanayi gibi alanlarda yaygın olarak desteklenmekte ve farklı birçok alanda kullanılmaktadır.

2.1.3.2. Derin Sinir Ağları

Derin sinir ağları birçok alanda kullanılmakta olup karmaşık problem yapılarında iyi performans gösterebilmektedir. Eğitim verileri, birden çok gizli katmana sahip derin sinir ağları için çok önemlidir. Eğitim için çok fazla seçenek mevcuttur. Bu seçeneklere hiper parametreler denir.

Hiper Parametreler nöron sayısı, katman ve aktivasyon içermektedir. Genellikle derin sinir ağları sınıflandırma ya da regresyon için kullanılmakta olup öğrenme süreçleri geniş zaman dilimleri içerisinde olabilmektedir (Küçük ve Arıcı, 2018).

2.1.3.3. Bozaltmann Makinesi

Bozaltmann yönteminin temeli, Saraft Dinov ve Hinton tarafından önerilen derin öğrenme konularından biridir. Bu ağın tüm katmanları arasında sınırsız bağlantı vardır. Tekrarlayan sinir ağlarıdır. Bu genellikle pratik çözümler için kullanılır. Maksimum olabilirlik algoritması 1999'da Younes tarafından önerilmiş olup yukarıdan aşağıya geri bildirim sağlamaktadır ancak yüksek veri üzerinde çok az etkisi vardır.

2.1.3.4. Derin Oto Kodlayıcılar

Derin oto kodlayıcılar, girdi ve çıktı katmanında aynı verileri kullanıldığı için, eğitim bu katmanları birbirine uyarlamaya çalışır ve yayılım algoritmasını kullanır. Oto kodlayıcılarda amaç en düşük kayıpla çıktıya ulaşmaktır ancak aynı veri setinin hem girdi katmanında hem de çıktı katmanında kullanılması bazı durumlarda olumsuz sonuçlara yol açabilmektedir. Bu durumun önlenebilmesi içinde girdi katmanına farklı gürültüler eklenerek eğitim içerisinde olmayan özelliklerle ile çıkışa ulaşım sağlanır (Kaynar ve ark., 2017). Şekil 3.8'de derin oto kodlayıcı yapısı gösterilmiştir.

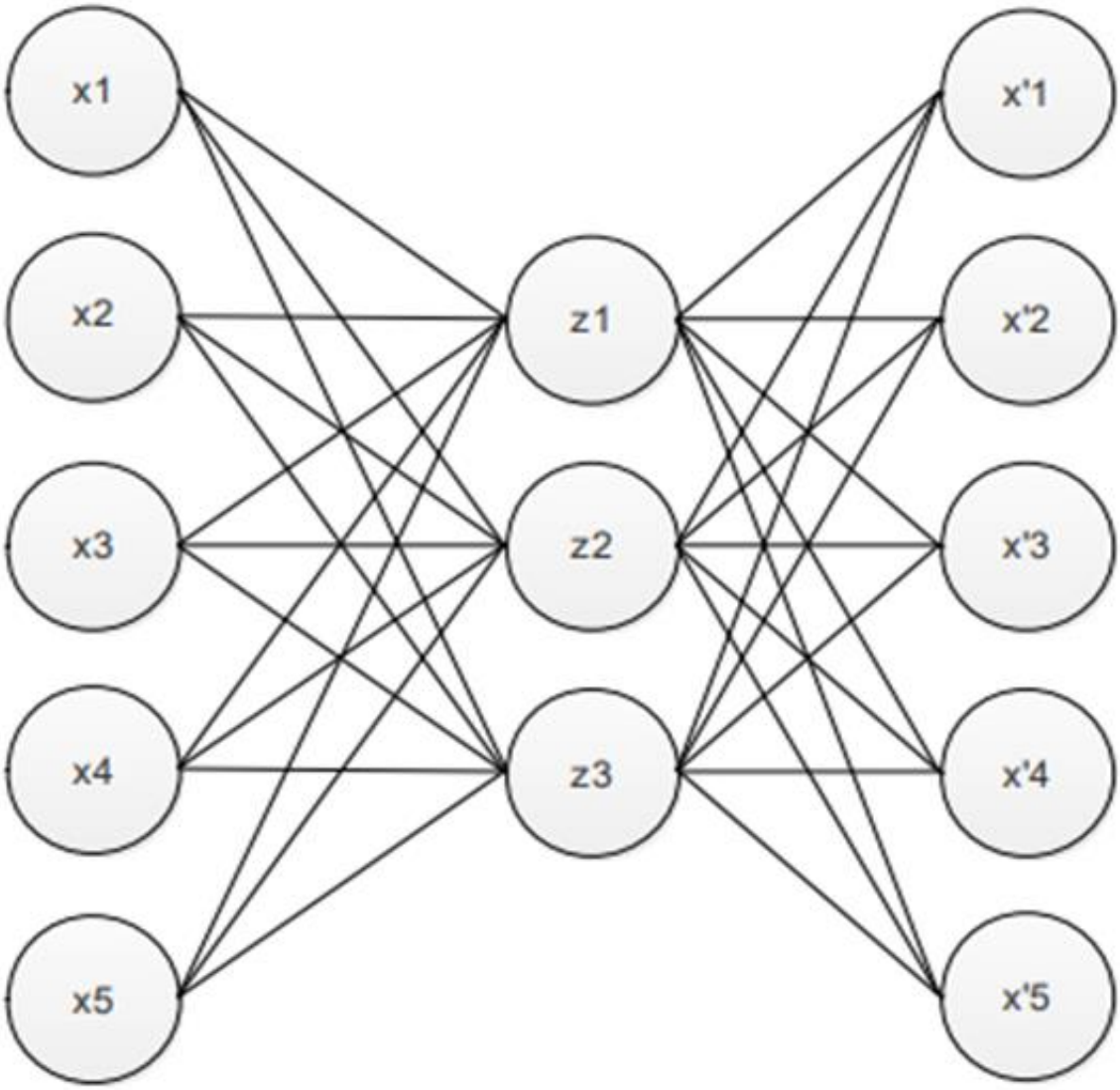
$$y_i = f\left(\sum_{i=1}^{n} xi \times w_{ij}\right) + b \qquad \text{Denklem 1}$$

Denklem 1'de katmanların aktarım formülü verilmiştir.

$$min \sum_{i=1}^{n} (xi \times w_{ij}) \qquad \text{Denklem 2}$$

Denklem 2'de minimum formülü verilmiştir.

Şekil 8’de derin oto kodlayıcı yapısı verilmiştir.

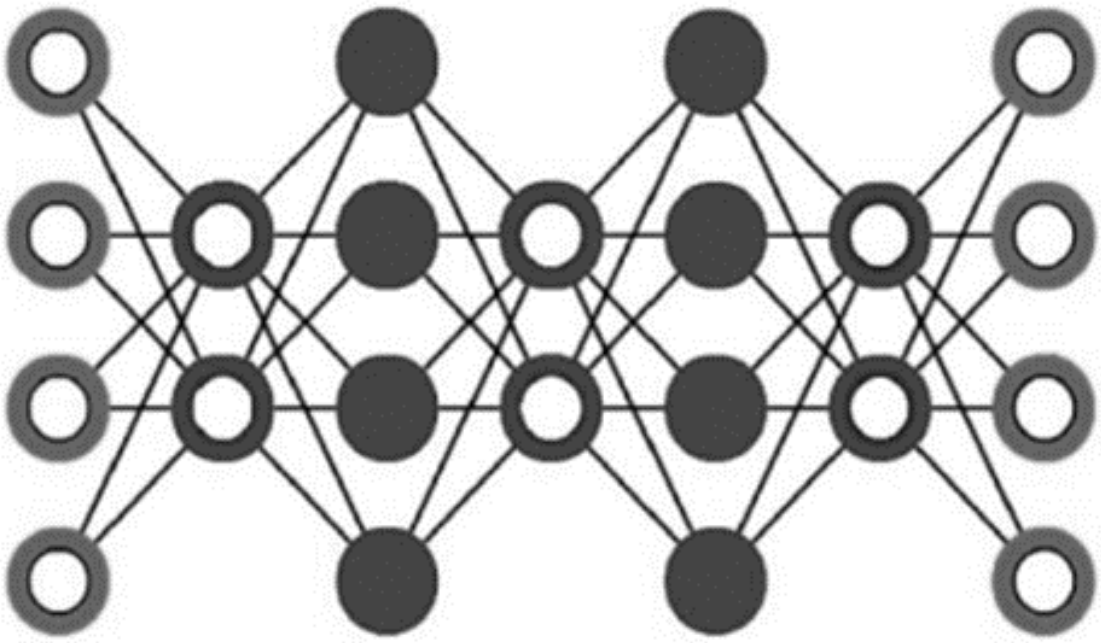


Şekil 8 Derin Oto Kodlayıcı Yapısı

2.1.3.5. Derin İnanç Ağ Modeli

Derin inanç ağları denetimsiz öğrenme yöntemleri içerisinde hem yönlü hem de yönlendirilmemiş olan grafiksel modeller içerisinden derin sinir ağı türevlerindendir. Genellikle görüntü ve üretme konularında yaygın olarak kullanılmaktadır (Doğan ve Türkoğlu 2019). Hinton çalışmalarında bu yöntem üzerine testler gerçekleştirerek çalışmalar yapmıştır.

Derin inanç ağları, alt ağlarda bağlantıları bulunmasına karşı katmanlar arasındaki düğümlerde bağlantıları yoktur. Gizli katmanın alt ağlarının bir sonraki katmanın işlevini gördüğü Kısıtlı Bozaltmann birleşimidir (Kaya ve ark. 2019). Şekil 9'de Derin inan ağ modeli verilmiştir.



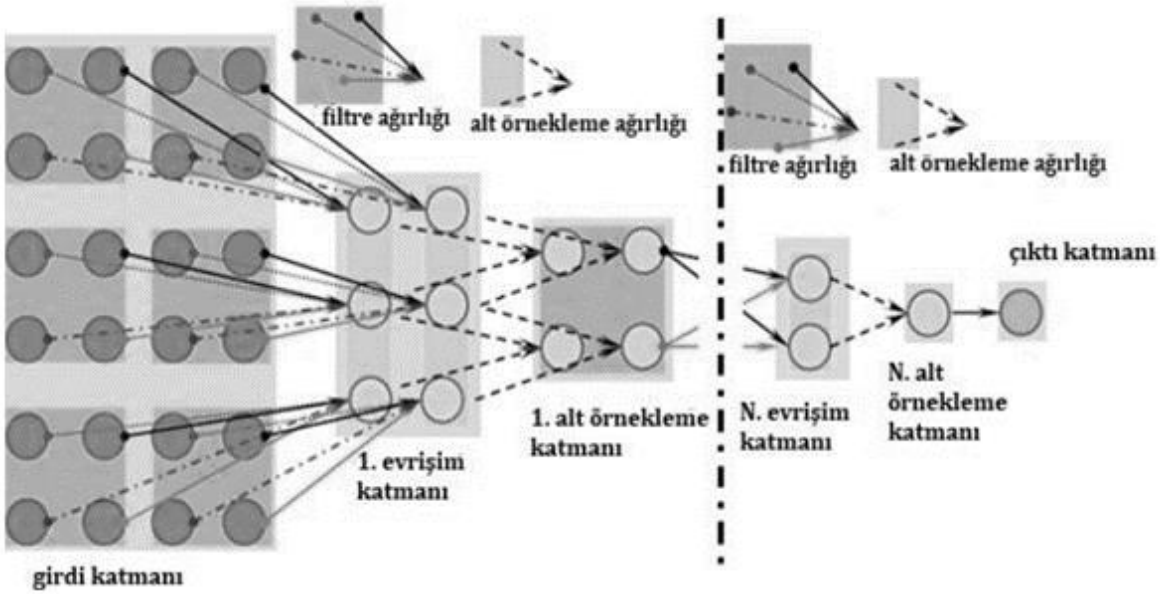
Şekil 9 Derin İnanç Ağ Modeli

2.1.3.6. Evrişimli Sinir Ağları

1998 yılında LeCun ve arkadaşları tarafından önerilen evrişimli sinir ağları, çok miktarlı veriler ile çalışabilirler. Gizli erişim fitresini göre nöronlar aktivasyonu 3 boyutlu çıktılara dönüştürürler (Kaya ve ark. 2019).

1962 yılında öne sürülen görsel koreksin nörobiyolojik modelden beslenerek geliştirilmiştir. Evrişim, havuzlama, girdi, tam bağlantılı ve çıktı katmanlarından oluşmaktadır (Küçük ve ark. 2021).

Evrişimli sinir ağlarında, parametre sayıları ağırlık paylaşımı ile yüksek oranda azalır. Buda verimliliği yüksek oranda olumlu olarak etkilemektedir (Küçük ve ark. 2021). Şekil 10'da Evrişimli sinir ağları yapısı gösterilmiştir.



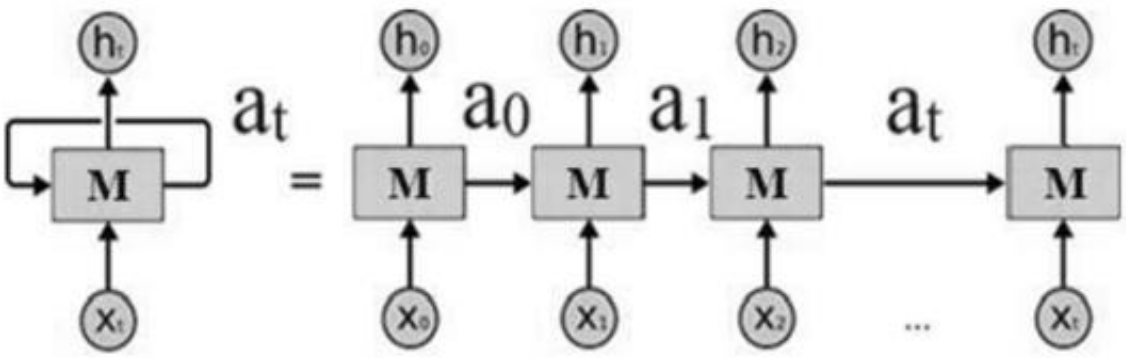
Şekil 10 Evrişimli Sinir Ağları

2.1.3.7. Yenilemeli Sinir Ağları

Yenilemeli sinir ağları (RNN), Elman tarafından tasarlanmıştır. Bu yöntem gizli yapı üzerine öğrenme süreçlerini ifade etmektedir.

Yinelemeli sinir ağlarında ağa giren veriler yeterli olmamaktadır ve daha önceki zaman serisi içerisindeki örneklerden alınmaktadır. Diğer sinir ağlarına kıyasla girişler bağımsız değildir ve veri çıktıları önce hesaplamalara dahil edilmektedir (Doğan ve ark. 2019).

Dil bilimcilerin oldukça ilgisini çeken bu yöntem genellikle dil çevirileri için kullanılmaktadır. Yenilenen ağları art arda gelen verilerle sonraki durumları tahminlemeye çalışır. Derin ve iki yönlü sinir ağları olarak ikiye ayrılmaktadır (Doğan ve ark. 2019). Şekil 11'da tekrarlanan sinir ağları yapısı gösterilmiştir.



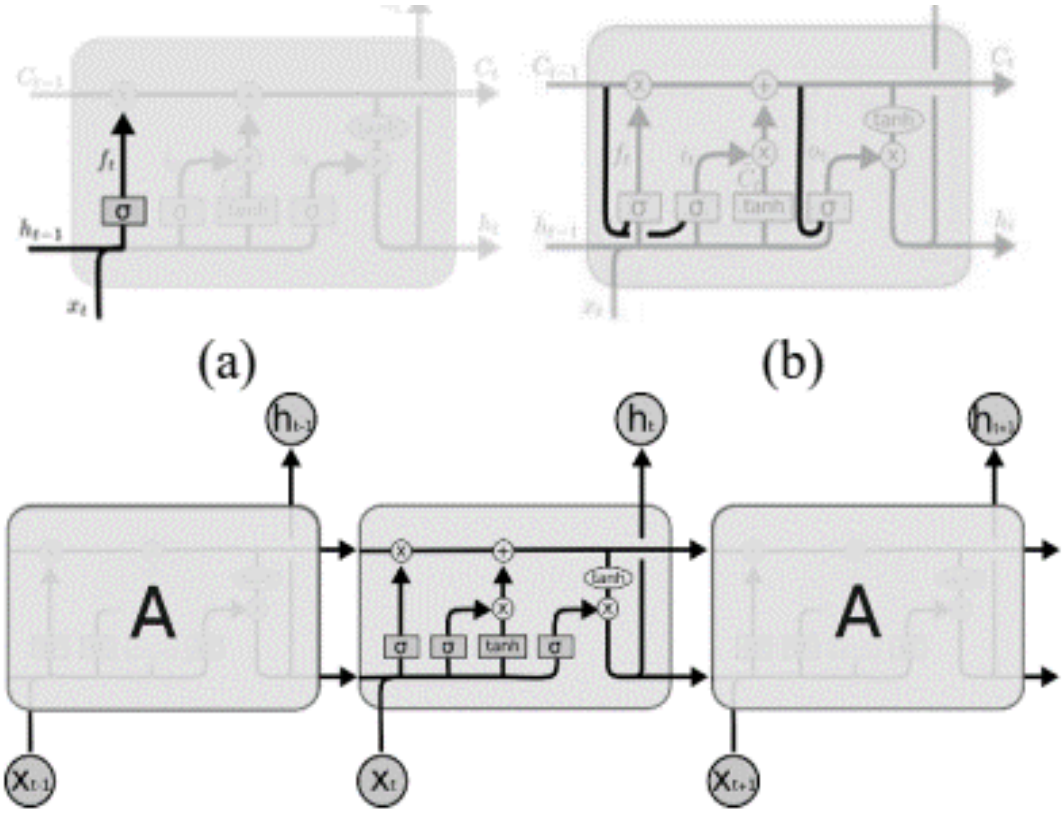
Şekil 11 Tekrarlanan Sinir Ağ Modeli

2.1.4 Uzun Kısa Süreli Hafıza

1997'de Hachreiter ve Schmidhuder, Uzun Kısa Süreli Hafızaların tarihsel verilerden toplanan bilgileri hatırlamak için önemli ve iyi fikirlere sahip olduğunu belirterek makalede tartışılan Uzun Kısa Süreli Hafıza kavramını önerdiler. Uzun Kısa Süreli Hafıza LSTM olarak bilinir. Doktora tezlerinde Hachreiter ve Schmidhuder, gürültülü sıkıştırılamaz giriş dizilerini 1000 adımdan oluşan zaman aralıklarında geçici bir ilişkiyi kaybetmeden bağlamayı öğrenebilirler. Önceki bilgileri kullanarak, bu mimariyle mükemmel performans gösterdiklerini söylediler. Hachreiter ve Schmidhuder tarafından tasarlanan LSTM mimarisi, güncellenmiş sinir ağı türevlerinden biridir. Uzun vadeli bağımlılık sorununa bir çözüm bulunmaya çalışılmıştır. Yenilenen sinir ağından tek farkı, gizli bir hesaplama yapısının varlığıdır (Keçeci, 2020).

Uzun kısa süreli hafıza, hafızasında geçmiş bilgileri tutabilmesi ve uzun vadeli bağımlıkları öğrenebilmesi özelliğiyle RNN'de yaşanan sorunlara çözüm olabilmiştir (Yıldız ve ark., 2021).

LSTM mimarisinin dört kapısı vardır; giriş, çıkış, unutma ve hafızadır. Hafıza yetenekleri, bağımlılıkları öğrenmelerine izin verir ve genellikle zaman serileri veya sıralı problemler için tercih edilir. LSTM mimarisi unutma kapısı, mevcut bir durumu silmek veya sıfırlamak için kullanılır ve öğrenmeyi etkinleştirmek için adımda alan kapısı kullanılır (Doğan ve Türkoğlu, 2019). LSTM mimarisi Şekil 12'de gösterilmiş olup "A" unutma kapısını "B" ise Alan gözetme kapısını göstermektedir.



Şekil 12 LSTM Yapısı

Uzun süreli bellek, zamanla öğrenilen bilgileri depolar ve bir sonraki adımda bir vektöre yazmanın gerekli olup olmadığına karar verir. LSTM'ler, vektöre yazılan faydalı bilgileri depolar ve diğer istenmeyen bilgileri o bellekten kaldırmak için 0 ve 1 değerlerini kullanarak performans sonuçlarını iyileştirir (Yiğit ve ark., 2021).

LSTM'de kazanılan bilgilerden hangilerinin saklanacağına ve hangi bilgilerin unutulması gerektiğine karar veren, elde edilen yeni bilgilerle geçmişteki bilgilerin hangileriyle değiştirilmesi için kullanılacağına karar veren ve son adımda ise ne şekilde aktarım yapılması gerektiğine belirten üç fonksiyon kapısını bulunur. Bu fonksiyonlar, ağlar arasındaki ilişkinin başarılı şekilde öğrenilmesi sağlamaktadır (Keçeci 2020).

Bu yöntem hem basit hem de hesaplama süresi RRN'lere göre daha kısadır ve metin okuma, uzun vadeli finansal tahminlemelerde, müzik ve metin işleme gibi alanlarda yaygın olarak kullanılmaktadır (Doğan ve ark. 2019). LSTM ilk adımda, iki farklı vektör bulunur ve hücrelere iki farklı vektör olarak aktarılır.

$$\tilde{c}^{<t>} = \tanh\left(w_c[h^{<t-1>}, x^{<t>}] + b_c\right)$$

Denklem 3

Denklem 3 de iki farklı vektör gösterilmiştir.

$$g_u = \sigma\left(w_u[h^{<t-1>}, x^{<t>}] + b_u\right)$$

Denklem 4

Denklem 4'te adımda aday vektör hesaplaması yapılır.

$$g_u = \sigma\left(w_u[h^{<t-1>}, x^{<t>}] + b_u\right)$$

Denklem 5

$$g_f = \sigma\left(w_f[h^{<t-1>}, x^{<t>}] + b_f\right)$$

Denklem 6

Denklem 5 sonrasında geçmiş bilgilerin ne oranda değişmesi gerektine Denklem 6'da karar verilir.

$$g_o = \sigma\left(w_o[h^{<t-1>}, x^{<t>}] + b_o\right)$$

Denklem 7

Denklem 7'de daha önce edinilmiş olan bilgilerin ne kadarlık kısmının saklanması gerektiği karar verelir ve yeni vektör hesaplanır.

$$c^{<t>} = (g_u . \tilde{c}^{<t>}) + (g_f . c^{<t-1>})$$

Denklem 8

Son adım olarakta çıktı kapısından hücereye aktarılması gereken vektör Denklem 8'de bulunur daha sonrasında güncelleme kapısı ve unutma kapası vektörleri çarpılır ve bu çarpımlar toplanarak yeni değer Denklem 9 da elde edilir.

$$h^{<t>} = g_o . \tanh\left(c^{<t>}\right)$$

Denklem 9

2.1.5 Doğrusal Durağan Stokastik Modeller

Farklı zaman serilerinde tahminlemeler yapmayı amaçlayan ve farklı ya da çeşitli modeller kullanan ve doğrusal ilişkileri modelleyen yöntemlere Durağan stokastik modeller denmektedir. Bunlar, Otoregresif model (AR), Hareketli ortalama (MA) ve Otoregresif hareketli ortalama ARMA modelidir (Duru 2017).

2.1.5.1 Otoregresif Model

AR modeli belirli bir zaman dilimi içerisinde gözlem değerini hata değeri üzerinden ifade eden doğrusal bilişim modelidir. Bu model için geçmişte yapılmış olan gözlem değerleri oldukça önemlidir. Buna bağlı olarak geçmiş gözlem değerleri üzerinden isimlendirilmektedirler. AR model "p" ifadesi ile adlandırılmaktadır. AR model genel anlamıyla, aynı değişkene ait gözlem değerleri ile geçmiş dönemlerdeki gözlem değerlerini açıklamayı amaçlar. Bu yönü ile AR model çoklu regresyonlardan ayrıştırılmıştır (Duru 2007).

2.1.5.2. Hareketli Ortalama Modeli

Hareketli ortalama modelleri (MA), daha önceki gözlemlerinden elde etmiş oldukları hata değerleri ve mevcut dönemdeki hata değerlerinin birleşimi şeklinde açıklanmaktadır (Duru 2007).

$$x_t = \theta_0 a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - \ldots - \theta_q a_{t-q} \qquad \text{Denklem 10}$$

Denklem 10'da MA model fonksiyon bileşeni aşağıdaki formüldeki gibi ifade edilmektedir.

2.1.5.3. Otoregresif Hareketli Ortalama Modeli

Hareketli ortalama modeli olan ARMA, hareketli ortalama modeli ve Otoregresif modellerin birleşimi olarak ifade edilmektedir. ARMA modeli "p" ve "q"nin birleşimini ifade etmektedir ve belirli zamanlardaki hata gözlem değerlerinin ondan önceki gözlem değeri hatasının birleşemi olarak tanımlanır. Birden fazla ifade şekli bulunmasına istinaden en yaygın kullanılan formül aşağıda ifade edilmiştir (Duru 2007).

2.1.6. Durağan Olmayan Doğrusal Stokastik Model

ARIMA yöntemi durağan olmayan zaman serileri altında yaygın olarak kullanılan modellerden birisidir. Bu yöntemin önderleri George Box ve Gwilym Jenkis'dir. Bu nedenle bazı literatür kaynaklarında Box-Jenkis (BJ) yöntemi olarak da rastlanabilmektedir (Bars ve ark. 2018). 1970'li yıllarda yayımlamış olan kitaplarda önerilmeye başlayan bu yöntem ileriye dönük tahminlerde kullanılan matematiksel bir modeldir (Berk ve ark 2019).

ARIMA modelleri tek değişkene sahip olan değişkenlerle birlikte zaman serilerinde tahminleme yapabilen modellerdendir. Farklı modeller içerisinden uygun olanın seçilerek ve bu seçilmiş olan durumun her aşamasında incelemeye uygunluğunu denetlemek için tasarlanmış deneysel süreçtir (Duru 2007).

ARIMA yöntemi diğer yöntemlere oranla herhangi bir ön bilgi kısıtlamasına ihtiyaç duymamaktadır. Diğer yöntemlerde genellikle zaman serisi eğilimlere oldukça önem verilmektedir fakat ARIMA için bu zorunluluk olarak bulunmamaktadır (Duru 2007). Bu yöntem kendi geçmiş bilgileri ve hatalarının ortalamalarından öğrenmektedir.

ARIMA yöntemi, "p", "q" ve "d" değerleri üzerinden hesaplanmaktadır. "P" değeri Otoregresif (AR) model, "q" hareketli ortalama (MA) model derecesini "d" ise fark alma derecelerini ifade etmektedir.

$$Yt = \theta + \alpha 1 Yt_{-p} + \beta o\, \mu t + \beta 1 \mu t_{-q} \qquad \text{Denklem 11}$$

Bu doğrultuda aşağıdaki Denklem 11'de formül üzerinden ARIMA yöntemi için gösterim yapılmaktadır (Bayramoğlu ve ark. 2017). ARIMA yöntemi için dört adım bulunmaktadır. Birinci adımda değişkenlik ve durağanlık durumunun belirlenebilmesi için analiz çalışması yapılır. Kısmi ve kısmi olamayan otokorelasyon fonksiyon incelemeleri yapılır ve fark dereceleri belirlenir (Bayramoğlu ve ark. 2017). İkinci adımda, ilk adımda elde edilen "p", "q" ve "d" değerleri üzerinden ARIMA modeli için tahminleme yapılır (Bayramoğlu ve ark. 2017).

Üçüncü adımda, diğer ARIMA yöntemlerine göre tahminlemesi yapılmış olan modellemenin uygunluğu kontrol edilerek en uygun model olması diğer tabiri ile beyaz görüntü uygunluk durumu kontrol edilir (Bayramoğlu ve ark. 2017). Son adımda ise, elde edilen veriler üzerinden geleceğe dair tahminlerin yapılmasıdır.

ARIMA modeli durağan olamayan ileriye dönük zaman serileri için oldukça yaygın olarak kullanılmaktadır. Genellikle tahminlerinde zorlanılan finansal değerin tahminlenmesinde yaygın olarak kullanılarak olumlu başarıların elde edildiği görülmüştür.

2.1.7. Prophet Yöntemi

Prophet algoritması, Facebook tarafından C++ programlama dili kullanılarak geliştirilen güçlü ve hızlı bir açık kaynaklı zaman serisi modelidir. Mevsimsellik ve tatil etkileri ile doğrusal olmayan trendlere uyması için bir ek regresyon modeli kullanır. Prophet modeli, yıllık mevsimsellik için Fourier düzenini kullanırken haftalık için ardından kukla değişkenler kullanılır. Kullanımı kolaydır ve varsayılan olarak trendler ve mevsimsel yapıya sahip veriler için ustaca tahminler yapmak amacıyla model için iyi bir hiper parametre kümesini otomatik olarak bulmak üzere tasarlanmıştır.

Prophet, üç ana bileşene sahip ayrıştırılabilir bir zaman serisi modeli olarak geliştirilmiştir. Bu bileşenler zaman serisinde eğilim, mevsimsellik ve tatil günlerini saptayarak işlev görmektedirler.

$$y(t) = g(t) + s(t) + h\,(t) + \in \qquad \text{Denklem 12}$$

Denklem 12'de modelleme süreci ifade edilmiştir. "$g(t)$" zaman serisinin trendi, "$s(t)$" mevsimsellik, "$h\,(t)$" tatil günlerinin etkilerini ve $\in(t)$" hata terimidir (Taylor ve Letham, 2018). Prophet modelini Holt Winters, ARIMA, vb. geleneksel tahmin modellerinden ayıran en çarpıcı özelliği, zaman serisi verilerini yumuşatmak ve tahmin etmek için Bayes tabanlı eğri uydurma uygulamasıdır.

2.2. Veri Seti Hazırlığı

Veri seti uygulamanın en önemli aşamasını oluşturmaktadır. Kullanılacak olan veriler hem problemin doğru olarak çözümlenmesini hem de modele doğru olarak uyarlanmasını için oldukça önemlidir. Seçilmiş olan veri setinin etkin ve pozitif sonuçlar elde edebilmek için uygun verilerden oluşmalıdır.

Bu tez çalışması kapsamında, makine öğrenmesinin alt dalı olan derin öğrenme yöntemleri ile kredi kartlarının gelecek dönem içerisindeki harcamalarının tahminlemeleri yapılarak elde edilen sonuçların başarı oranları karşılaştırılmıştır.

Bu tez çalışması kapsamında, özel bir firmanın veri tabanında bulunan gerçek veriler kullanılmıştır. Bu veriler firmasının 2020-2022 yılları aralığındaki verilerden alınmıştır.

Veri seti içerisinde; cinsiyet bilgisi, tarih bilgisi, yaş bilgi, kurum kodu, işlemin nasıl ve nerden yapılmış olduğunu belirten kategori kodu (MCC) ve harcama tutarları saat bazlı olarak alınmış olup veri setinde yapılan başarılı işlemler üzerinden oluşturulmuştur. Tablo 1'de veri setinin özellikleri verilmiştir.

Tablo 1 incelendiği zaman veri seti içerisinde toplamda 7 özellik bulunmaktadır. Bu 7 özellik için 100000 adet örnek veri bulunmaktadır. Veri seti virgülle ayrılmış format ile karakter kodlaması UTF-8 olacak şekilde alınmıştır.

Tablo 1 Veri Seti Özellikleri

Banka Kodu	Veri seti 2 farklı üzerinde oluşturulduğu için 2 haneli banka kodu ile ayrıştırılmak için kullanılmıştır. (80/20)
Tarih	Harcamaların yapılmış olduğu tarih bilgisi GG/AA/YYYY şeklinde alınmıştır.
Cinsiyet	K/E erkek olacak şekilde harcama yapan kart sahiplerinin cinsiyeti belirtilmiştir.
Yaş Bilgisi	Harcama yapan kart sahiplerinin yaş bilgileri GG/AA/YYYY formatında alınarak toplamları hesaplama ile veri setine eklenmiştir.
Kategori Kodu (MCC)	Kredi kartı harcamalarının yapılmış olduğu yerin hangi alana ait olduğunu belirten alan. Market, Tekstil, Kurum gibi alanlara göre sınıflandırılması yapılmış ve 4 haneli numerik değerler verilmiştir.
İşlem Sayısı	KK ile saatlik olarak kaç adet harcama yapıldığı bilgisini içerir.
Tutar	Harcama yapılan tutar bilgisi TL cinsi üzerinden alınmıştır.

2.3. Araştırma Yöntemi

2.3.1. Uygulama Hazırlığı

Bu tez çalışması kapsamından gerçekleşen adımlar sıralı bir şekilde gerçekleştirilmiş ve aşamalandırılmıştır.

Birinci yöntem veri toplamadır. Bu çalışma ile tutarlı kullanılacak verilerin temini ve detayları açıklanmıştır. Veri kalitesi ve araştırma çalışmaları ile tekrarlama gibi hususlar göz önünde bulundurulmuştur. Veri kümesi bulunduktan sonraki adım, makine öğrenimi algoritması için veri kümesini analiz etmektir. Veri kümesinin kullanılabilir bir biçime dönüştürülmesi gereken bölümdür ve birbirleriyle karşılaştırılarak başarı oranları gözlemlenmiştir.

Uygulama ve kodlama için Python 3.8 versiyonu seçilmiştir. Kodlama ortamı ise Google Colap ve Jupiter Notebook kullanıldı. "Pandas" sayesinde veri seti okunarak ve sisteme aktarılmıştır. Çalışmanın gerçekleştirdiği bilgisayar ortamı ise Windows 10 işletim sistemine sahiptir.

Kod derleyici ortamında virgülle ayrılmış formatta tutulan veri seti okuma işlemine tabi tutulmuştur. Veri seti okunduktan sonra boş karakter içeren değerler silindi. Boş karakterlerin silinmesi işlemi gürültü veri kirliliğinin önüne geçilmek için yapılmıştır.

Veri setinin bir sonraki adımında metinsel ifadelerin sayısal değerlere dönüştürülmesi gerekir. Makine öğrenmesi algoritmalarının daha sağlıklı çalışabilmesi için bu dönüştürme işleminin yapılması gerekir. Veri setindeki Cinsiyet, Kategori Kodu alanları sayısal değerlere dönüştürüldü.

Veri setindeki tüm veriler sayısal değerler içermektedir ancak veriler üzerinde veri ölçeklendirilmesi gerekir. Veri ölçeklendirmenin amacı verilerin normal dağılamaması ve model performansının etkilememesi için gerekmektedir. Normalizasyon işleminin yapılabilmesi için çeşitli yollar bulunmaktadır. Bu çalışma içerisinde bu yollardan min-max normalizasyon yöntemi kullanılmıştır. Min-max normalizasyon yöntemi kullanılan en yaygın normalizasyon yöntemlerinden biridir. Bu normalizasyon yöntemi ile en düşük değer 0 en yüksek değer 1 olacak şekilde bütün veriler bu aralığa yerleştirilmektedir. Tarih alanı dışındaki tüm alanlar veri ölçeklendirme ile 1-0 arasına indirgenmiştir.

2.3.1.1. Derin Öğrenme Yöntemlerinin Uygulanması

Veri setindeki 100000 adet verinin %40'ı test için geri kalan %60'lık kısım ise eğitim için ayrılmıştır. Tablo 2'de veri setindeki verilerin dağılımı gösterilmiştir.

Tablo 2 Veri Setinde Test ve Eğitim Verilerinin Dağılımı

Test Verisi	Eğitim Verisi
40000	60000

2.3.1.2. LSTM Uygulanması

LSTM algoritması için kullanılan parametreler aşağıdaki gibidir;

- Units: çıktı uzayının boyutudur. Sayısal değer içerir. 100 olarak verilmiştir.
- Activation: Doğrusal etkinleştirme fonksiyonun uygulanıp uygulanmayacağını ifade eder. Yok olarak belirlenmiştir.
- Use bias: Katmanın bir sapma vektörü kullanıp kullanmadığını ifade eder. False olarak seçilmiştir.
- Kernel_initializer: Girdilerin doğrusal dönüşümü için kullanılan çekirdek ağırlıkları matrisi için başlatıcıdır. Glorot_uniform olarak seçilmiştir.
- Recurrent_initializer: Doğrusal etkinleştirme dönüşümü için kullanılan recurrent_kernel ağırlıkları matrisi için başlatıcıdır. Ortogonal olarak seçilmiştir.
- Bias_initializer: Önyargı vektörü için başlatıcıdır. Zeros olarak seçilmiştir.
- Unit_forget_bias: Eğer true olarak seçilirse, Bias_initializer değerini zero olarak seçer.
- Recurrent_regularizer: recurrent_kernel ağırlık matrisine uygulanan düzenleyici işlevini gerçekleştirir. Yok olarak seçilmiştir.
- Kernel_constraint: Çekirdek ağırlıkları matrisine uygulanan kısıtlama işlevidir. Yok olarak seçilmiştir.

- recurrent_kernel: Ağırlık matrisine uygulanan kısıtlama işlevidir. Yok olarak seçilmiştir.
- Return_sequences: Son çıktının döndürülüp döndürülmeyeceğini göstetir. Hayır olarak seçilmiştir.
- Return_state: Son çıktıya ek olarak son durumun döndürülüp döndürülmeyeceğini ifade eder. Hayır olarak seçilmiştir.
- Go_backwards: Sıranın ters olarak kullanılması için tercih edilir. False olarak seçilmiştir.
- Stateful: Veri setindeki her "i" indeksindeki her numunenin son durumu, bir sonraki serideki her "i" indeksli numune için başlangıç durumu olarak kullanılacaktır.

Algoritma için gerekli parametreler verildikten sonra model eğitime girmiştir. Her bir adımdaki epoch çıktısı aşağıdaki gibidir;

Epoch 1/30
5998/5998 [==========] - 56s 8ms/step - loss: 5.6440e-04
Epoch 2/30
5998/5998 [==========] - 47s 8ms/step - loss: 5.6356e-04
Epoch 3/30
5998/5998 [==========] - 48s 8ms/step - loss: 5.6397e-04
Epoch 4/30
5998/5998 [==========] - 46s 8ms/step - loss: 5.6382e-04
Epoch 5/30
5998/5998 [==========] - 48s 8ms/step - loss: 5.6496e-04
Epoch 6/30
5998/5998 [==========] - 37s 6ms/step - loss: 5.6330e-04
Epoch 7/30
5998/5998 [==========] - 22s 4ms/step - loss: 5.6799e-04
Epoch 8/30
5998/5998 [==========] - 22s 4ms/step - loss: 5.6374e-04

Epoch 9/30

5998/5998 [==========] - 23s 4ms/step - loss: 5.6639e-04

Epoch 10/30

5998/5998 [==========] - 44s 7ms/step - loss: 5.6553e-04

Epoch 11/30

5998/5998 [==========] - 48s 8ms/step - loss: 5.6216e-04

Epoch 12/30

5998/5998 [==========] - 47s 8ms/step - loss: 5.6287e-04

Epoch 13/30

5998/5998 [==========] - 48s 8ms/step - loss: 5.6104e-04

Epoch 14/30

5998/5998 [==========] - 47s 8ms/step - loss: 5.6357e-04

Epoch 15/30

5998/5998 [==========] - 47s 8ms/step - loss: 5.6232e-04

Epoch 16/30

5998/5998 [==========] - 46s 8ms/step - loss: 5.6476e-04

Epoch 17/30

5998/5998 [==========] - 47s 8ms/step - loss: 5.6369e-04

Epoch 18/30

5998/5998 [==========] - 43s 7ms/step - loss: 5.6548e-04

Epoch 19/30

5998/5998 [==========] - 42s 7ms/step - loss: 5.6584e-04

Epoch 20/30

5998/5998 [==========] - 49s 8ms/step - loss: 5.6304e-04

Epoch 21/30

5998/5998 [==========] - 48s 8ms/step - loss: 5.6348e-04

Epoch 22/30

5998/5998 [==========] - 49s 8ms/step - loss: 5.6440e-04

Epoch 23/30

5998/5998 [==========] - 13s 2ms/step - loss: 5.6523e-04

Epoch 24/30

5998/5998 [==========] - 10s 2ms/step - loss: 5.6349e-04

Epoch 25/30

5998/5998 [==========] - 11s 2ms/step - loss: 5.6159e-04

Epoch 26/30

5998/5998 [==========] - 12s 2ms/step - loss: 5.6202e-04

Epoch 27/30

5998/5998 [==========] - 12s 2ms/step - loss: 5.6143e-04

Epoch 28/30

5998/5998 [==========] - 13s 2ms/step - loss: 5.6099e-04

Epoch 29/30

5998/5998 [==========] - 15s 2ms/step - loss: 5.6317e-04

Epoch 30/30

5998/5998 [==========] - 14s 2ms/step - loss: 5.6415e-04

Model çalıştırıldığında, eğitim verisinin RSME skoru 0.0013, test verisinin RMSE 0.0178 olarak bulunmuştur. Tablo 3'te LSTM algoritması skor sonuçları verilmiştir.

Tablo 3 LSTM Algoritması Sonuçları

	RMSE
Test	0.0178
Eğitim	0.0013

3.2.1.3. ARIMA Uygulanması

ARIMA algoritması için kullanılan parametler aşağıdaki gibidir;

- endog: Gözlenen zaman serisi sürecidir. Kullanılmadı.
- exog: Dışşal regresörler dizisidir.Eğitim verisi dizisi verildi.
- order: Otoregresif, farklar ve hareketli ortalama bileşenleri için modelin (p,d,q) sırasıdır. d her zaman bir tamsayıdır, p ve q ise tamsayılar veya tamsayı listeleri olabilir. Kullanılmadı.
- seasonal_order: Yapay zeka parametreleri, farklılıklar, MA parametreleri ve periyodiklik için modelin order (P,D,Q,s) sırasıdır. Varsayılan (0, 0, 0, 0)'dır. D ve s her zaman tam sayılardır, P ve Q ise tam sayılar veya pozitif tam sayı listeleri olabilir. 5,1,0 olarak seçildi.

- Trends: ('n','c','t','ct') eğilimi kontrol eden parametre. 'c' sabit bir terimi, 't' zamandaki doğrusal bir eğilimi ve 'ct' her ikisini de içeren bir dize olarak belirtilebilir. numpy.poly1d'de olduğu gibi, bir polinomu tanımlayan yinelenebilir bir olarak da $a+bt+c^t$ belirtilebilir, burada [1,1,0,1] ifade eder. Kullanılmadı.
- enforce_stationarity: Hata teriminin varyansı, olasılık dışında konsantre edilip edilmeyeceğini gösterir.. Bu, parametre sayısını bir azaltır. Kullanılmadı.

Algoritma için gerekli parametreler verildikten sonra model eğitime girmiştir. Her bir adımdaki epoch çıktısı aşağıdaki gibidir;

predicted=0.002574, expected=0.000741
predicted=0.002529, expected=0.116633
predicted=0.019775, expected=0.001016
predicted=0.021318, expected=0.000433
predicted=0.018550, expected=0.000496
predicted=0.018702, expected=0.001736
predicted=0.023786, expected=0.000645
predicted=0.021123, expected=0.000745
predicted=0.000823, expected=0.000813
predicted=0.000788, expected=0.001033
predicted=0.000939, expected=0.004350
predicted=0.001518, expected=0.003011
predicted=0.001754, expected=0.000263
predicted=0.001654, expected=0.000965
predicted=0.001650, expected=0.001230
predicted=0.001892, expected=0.001750
predicted=0.002019, expected=0.001774
predicted=0.001483, expected=0.002466
predicted=0.001372, expected=0.001000
predicted=0.001521, expected=0.001100
predicted=0.001539, expected=0.001100
predicted=0.001539, expected=0.001500

predicted=0.001534, expected=0.001309
predicted=0.001419, expected=0.005156
predicted=0.001808, expected=0.006725
predicted=0.002764, expected=0.002316
predicted=0.002972, expected=0.000933
predicted=0.002848, expected=0.001800
predicted=0.003053, expected=0.000751
predicted=0.003143, expected=0.002010
predicted=0.002480, expected=0.005164
predicted=0.002122, expected=0.004355
predicted=0.002497, expected=0.000719
predicted=0.002398, expected=0.001084
predicted=0.002253, expected=0.001575
predicted=0.002569, expected=0.003568
.
.
.
.
predicted=0.002747, expected=0.000301
predicted=0.002676, expected=0.001175
predicted=0.002513, expected=0.001221
predicted=0.001959, expected=0.003600
predicted=0.002715, expected=0.000850

ARIMA yöntemi süresi oldukça üzün sürmüştür. Çalışan bilgisayar ortamını zorladığı gözlemlenmiştir. Tablo 4'te Arima algoritması modelinin sonuçları verilmiştir.

Tablo 4 ARIMA Algoritması Sonuçları

	RMSE
Test	0.0289
Eğitim	0.0086

3.2.1.4. PROPHET Uygulanması

Geçmiş verilere dayanarak yhat değerlerini (gelecekteki vakaları) tahmin etmek için Prophet modelini uygulandı.

Tablo 5'te, yhat'ın alt ve üst sınırlarına dayalı olarak tahmin edilen (yhat) ve doğrulanan (yhat) değerleri göstermektedir.

Tablo 5 PROPHET Modeline Ait İlk Tahminler

TREND	YHAT_LOWER	YHAT_UPPER	TREND_LOWER	TREND_UPPER	YHAT
1,64E+08	1,45E+08	1,79E+08	1,52E+08	1,71E+08	1,64E+08
1,68E+08	1,53E+08	1,72E+08	1,53E+08	1,74E+08	1,69E+08
1,75E+08	1,57E+08	1,78E+08	1,63E+08	1,86E+08	1,74E+08
1,79E+08	1,64E+08	1,84E+08	1,67E+08	1,94E+08	1,79E+08
1,84E+08	1,74E+08	1,86E+08	1,69E+08	1,96E+08	1,84E+08

Trend değerleri, Trend alt ve trend üst tarafından doğrulandı; yhat değerleri yhat alt ve yhat üst arasında olmalıdır. Model tarafından üretilen sonuçlar (yhat), yhat alt ve yhat üst aralığı arasındadır.

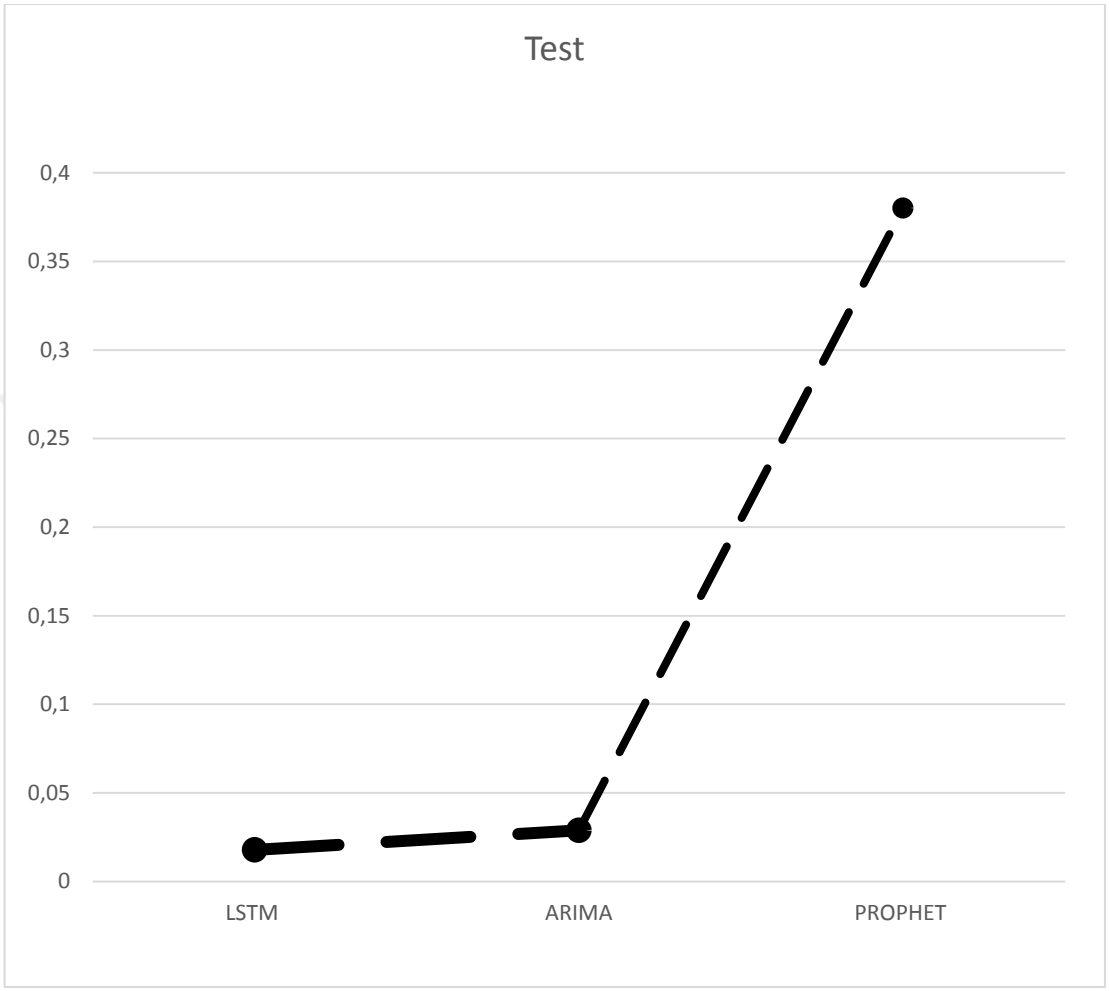
Tablo 4, Tablo 5 ve Tablo 6 kıyaslandığında LSTM yönteminin daha başarılı sonuçlar verdiği görülmüştür.

Tablo 6 PROPHET Algoritması Sonuçları

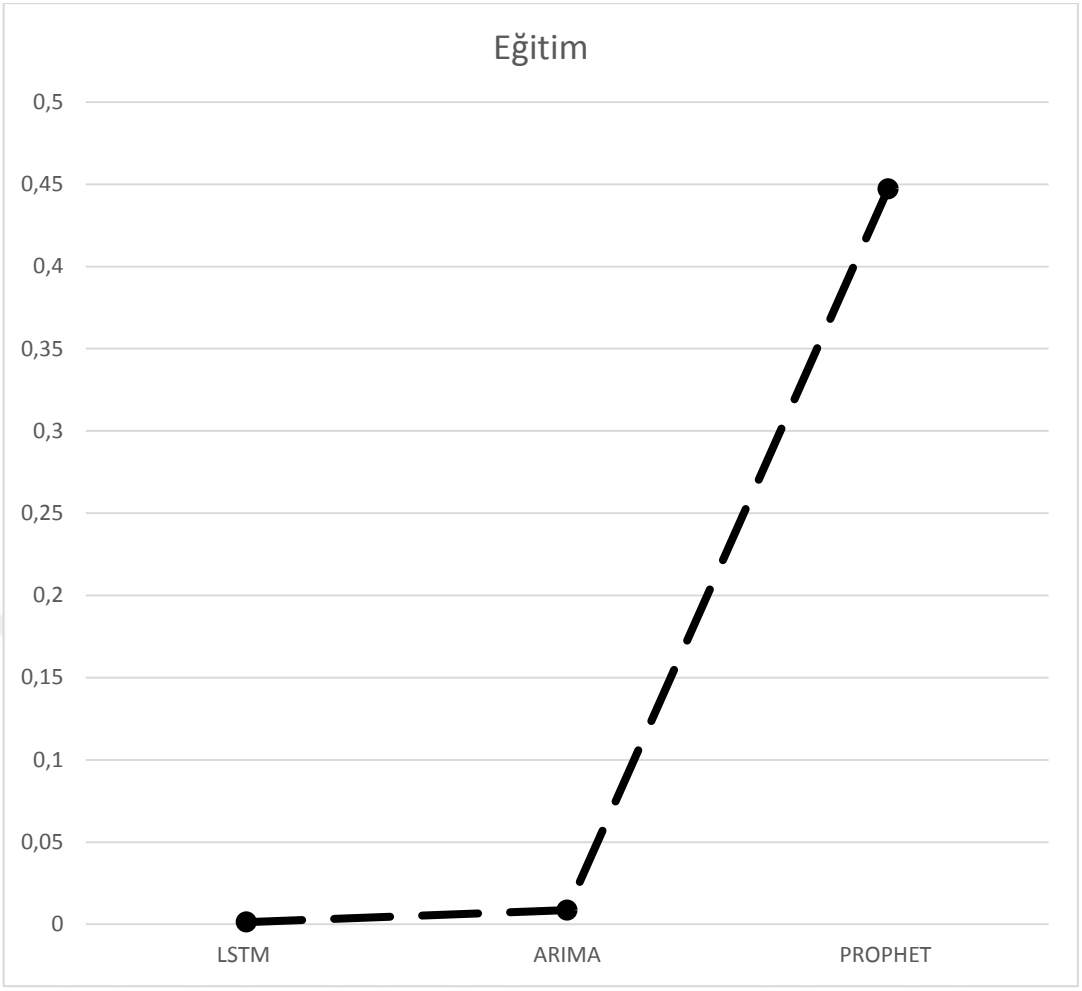
	RMSE
Test	0.3801
Eğitim	0.4472

2.3.2. Araştırma Sonuçları

Şekil 13.'de LSTM, ARIMA ve PROPHET yöntemlerindeki test skorlarının karşılaştırılması verilmiştir.



Şekil 13 LSTM, ARIMA ve PROPHET Yöntemlerindeki Test Skorlarının Karşılaştırılması



Şekil 14 LSTM, ARIMA ve PROPHET Yöntemlerindeki Eğitim Skorlarının Karşılaştırılması

Şekil 14.'te LSTM, ARIMA ve PROPHET yöntemlerindeki eğitim skorlarının karşılaştırılması verilmiştir.

SONUÇLAR VE ÖNERİLER

Bu çalışmada, kredi kartı harcamalarının makine öğrenmesi yöntemleriyle tahmin edilmesi önerilmiştir. Çalışmada özyinelemeli bir derin sinir ağı modeli olan LSTM yapısının yukarıda tanımlanan probleme müsait bir çözüm sunabileceği öne sürülmüş ve LSTM ile elde edilmiş sonuçlar geleneksel yöntemler ile karşılaştırılmıştır. Denemelerde özel bir firma tarafından elde edilen verilerden rastgele seçilen 10000 adet veri alınmıştır. 6000 adedi eğitme, 4000 adedi ise test için ayrılarak kullanılmıştır.

LSTM yönteminin verileri daha iyi kullandığı ve veri sayısı arttıkça daha iyi sonuçlar verdiği gözlemlenmiştir. LSTM, ARIMA ve PROPHET yöntemi kullanılırken veri setinde bilgi kaybı olmaması ve veri setinin boyuta bağlı uyumu göz önüne alındığında hata sonuçlarına göre LSTM'nin daha iyi sonuçlar verdiği görülmüştür.

Bu bilgiler göz önüne alındığında, bu çalışmada elde edilen sonuçların literatürdekilerle örtüştüğü açıktır. Daha önceki çalışmaların sonuçlarıyla karşılaştırıldığında, bu çalışmada elde edilen sonuçlar benzer sonuçlar göstermektedir.

Bu çalışmada ARIMA yöntemin çok yavaş çalıştığı görülmüştür. ARIMA yöntemi bilgisayar üzerindeki bellek kullanımı çok zorlamaktadır. Çünkü içerisindeki hesaplamalar ile diske yazıp sürekli oradan okuması zaman almaktadır. Ancak ARIMA yönteminin az sayıda örnek içeren veri setlerinde daha başarılı olduğu ortaya çıkmaktadır. ARIMA modeli, kısa vadeli tahmin için Prophet modelinden daha iyidir.

LTSM yönteminin, ARIMA ve PROPHET yöntemlerine göre daha başarılı olduğu görülmüştür. LSTM yöntemi fazla sayılı veri setlerinde başarılı olduğu ortaya çıkmıştır. Bundan sonraki çalışmalarda veri seti oluşturulurken tarih aralıkları ile mevsimler ve aylara gruplayarak bir özellik içerisindeki bağımsız değişken sayısını azaltılmayacaktır. Veri setindeki bağımsız değişken sayısının az olması kullanılan algoritma modelinin daha tutarlı olmasını sağlamaktadır.

KAYNAKÇA

Akay, E. Ç., Topal, K. H., Kizilarslan, S. ve Bulbul, H., 2019, Türkiye konut fiyat endeksi öngörüsü: ARIMA, rassal orman ve arima-rassal orman, *PressAcademia Procedia*, 10 (1), 7-11.

Akdağ, M. ve Bozma, G., 2021, Prediction of Bitcoin Price with Stock to Flow Model and Facebook Prophet Algorithm, *Uluslararası Ekonomi İşletme ve Politika Dergisi*, 5 (1), 16-30.

Arslan, K. J. B. A. E. B. D., 2020, Eğitimde yapay zekâ ve uygulamaları, 11 (1), 71-88.

Atalay, M. ve Çelik, E., 2017, Büyük veri analizinde yapay zekâ ve makine öğrenmesi uygulamalari-artificial intelligence and machine learning applications in big data analysis, 9 (22), 155-172.

Ataseven, B., 2013, Yapay sinir ağları ile öngörü modellemesi, *Öneri Dergisi*, 10 (39), 101-115.

Aydın, Ö. F., 2019, Yapay sinir ağları ile enflasyon tahmini, *Beykent Üniversitesi*.

BKM, 2022, Kart Sayıları, https://bkm.com.tr/en/tr-kart-sayilari/: [Ziyaret Tarihi: 1 Ocak 2022].

Çalışkan, D. ve Demir, Ö., 2022, Derin Öğrenme Yöntemleri ile Şüpheli Davraniş Tespiti, *International Periodical of Recent Technologies in Applied Engineering*, 3 (1), 26-41.

Çavuş, M. F., 2006, Bireysel Finansmanın Temininde Kredikartları: Türkiye'de Kredikartı Kullanımı Üzerine Bir Araştırma, *Selçuk Üniversitesi Sosyal Bilimler Enstitüsü Dergisi* (15), 173-187.

Çeker, M., 1992, Kredi kartı uygulaması ve özel hukuk açısından kredi kartının hukuka aykırı kullanımı, *Ankara Üniversitesi*.

Çelik, Ö., 2018, A research on machine learning methods and its applications, 1 (3), 25-40.

Çelikel, A. D., 2018, Stock value prediction using machine learning and text mining, *Kadir Has Üniversitesi*.

Demiray, S. ve Gürhanlı, A., 2020, Stock Closing Prediction with Machine Learning Algorithms.

Doğan, F. ve Türkoğlu, İ., 2019, Derin öğrenme modelleri ve uygulama alanlarına ilişkin bir derleme, *Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi*, 10 (2), 409-445.

Ereken, Ö. ve Tarhan, Ç., 2018, İŞ BAŞVURULARININ MAKİNE ÖĞRENMESİ YÖNTEMLERİYLE DEĞERLENDİRİLMESİ, *Yönetim Bilişim Sistemleri Dergisi*, 7 (2), 65-85.

Fırat, M. ve Güngör, M., 2004, Askı madde konsantrasyonu ve miktarının yapay sinir ağları ile belirlenmesi, *Teknik Dergi*, 15 (73).

Fredkin, E., 1992, Finite nature, 1, 345.

Gençalp, E., 2020, Makine Öğrenmesi Yöntemleri İle ATM'lerde Nakit Tahmini, Yüksek Lisans, *Galatasaray Üniversitesi*.

Girginer, N., Çelik, A. E. ve Uçkun, N. J. A. U. J. o. S. S., 2008, ESKİŞEHİR OSMANGAZİ ÜNİVERSİTESİ İKTİSADİ VE İDARİ BİLİMLER FAKÜLTESİ ÖĞRENCİLERİNİN KREDİ KARTI KULLANIMLARINA YÖNELİK BİR ARAŞTIRMA, 8 (1).

Gökçay, E. D., 2020, ATM cash stock prediction using different machine learning approaches, *Sabancı Üniversitesi*.

Hasan, A., 2020, Derin Öğrenme Ve Makine Öğrenmesi Yöntemleriyle Borsa Alım Satım Davranışlarının Modellenmesi, Doktora, *Yıldız Teknik Üniversitesi*, 91.

Hurwitz, J. ve Kirsch, D. J. I. L. E., 2018, Machine learning for dummies, 75.
Kara, Ş. E. ve Şamlı, R., 2021, Yazılım Projelerinin Maliyet Tahmini için WEKA'da Makine Öğrenmesi Algoritmalarının Karşılaştırmalı Analizi, *Avrupa Bilim ve Teknoloji Dergisi* (23), 415-426.
Karagöz, S., 2020, Payların kapanış fiyatlarının makine öğrenmesi yöntemleri ile tahmin edilmesi, Yüksek Lisans, *Beykent Üniversitesi*.
Kaya, T. S., 2019, Veri madenciliği algoritmaları ile kredi kartı kullanım alışkanlıklarının incelenmesi ve kişiye özgü kampanya teklifi, Yüksek Lisans
İstanbul Üniversitesi.
Kaynar, O., Aydın, Z. ve Görmez, Y., 2017, Sentiment analizinde öznitelik düşürme yöntemlerinin oto kodlayıcılı derin öğrenme makinaları ile karşılaştırılması, *Bilişim Teknolojileri Dergisi*, 10 (3), 319-326.
Keçeci, E., 2020, Yinelemeli Sinir Ağları ile Finansal Veri Tahmini.
Kocabaş, M., Canik, F., Yeşilyurt, C. ve Günkurt, M. Ş., 2019, Yapay Zeka Teknikleri İle Kripto Para Değer Tahmini, *Ekonomi Bilimleri Dergisi*, 14 (1), 72-101.
Koç, Y., 2021, Makine öğrenmesi ile çok terimli hisse senedi yönlü tahmini; BIST100 örneği/Multinomial direction forecast with machine learning algorithms; BIST100 example, *Kadir Has Üniversitesi*.
Köksoy, M., 2021, Makine Öğrenmesi İle SCO Kasa Sistemlerinin Temassız Alışverişlerinde Çalıntı Tahmini, Yüksek Lisans, *Beykent Üniversitesi*
Küçük, D. ve Arıcı, N., 2018, Doğal Dil İşlemede Derin Öğrenme Uygulamalari Üzerine Bir Literatür Çalişmasi, *Uluslararası Yönetim Bilişim Sistemleri ve Bilgisayar Bilimleri Dergisi*, 2 (2), 76-86.
Microsoft, 2022, Makine öğrenmesi nedir?, https://azure.microsoft.com/tr-tr/overwiew/what-is-mechine-learning-platform/: [Ziyaret Tarihi: 1 Ocak 2022].
Nacar, E. N. ve Erdebilli, B., 2021, Makine Öğrenmesi Algoritmaları ile Satış Tahmini, *Journal of Industrial Engineering*, 32 (2), 307-320.
Namlı, Ö. H. ve Özcan, T., 2017, Makine öğrenmesi Algoritmasını Kullanarak Gişe Hasılatının Tahminlemesi, *Yönetim Bilişim Sistemleri Dergisi*, 3 (2), 130-143.
Pulat, M. ve Kocakoç, İ. D. J. Y. v. E. D., 2021, Türkiye'de Makine Öğrenmesi ve Karar Ağaçları Alanında Yayınlanmış Tezlerin Bibliyometrik Analizi, 28 (2), 287-308.
Selimoğlu, M. ve Yılmaz, A., 2021, Kredi Kartı Dolandırıcılık Tespitinin Makine Öğrenmesi Yöntemleri ile Tahmin Edilmesi, *Beykent Üniversitesi Fen Ve Mühendislik Bilimleri Dergisi*, 13 (2), 28-33.
Sevli, O. ve Gülsoy, V. G. B., 2020, Covid-19 salgınına yönelik zaman serisi verileri ile Prophet model kullanarak makine öğrenmesi temelli vaka tahminlemesi, *Avrupa Bilim ve Teknoloji Dergisi* (19), 827-835.
Simon, H. A. J. S. m. j., 1993, Strategy and organizational evolution, 14 (S2), 131-142.
Şahin, M. G. ve Öztürk, N. B., 2018, Eğitim Alanında Ölçek Geliştirme Süreci: Bir İçerik Analizi Çalışması *Kastamonu Eğitim Dergisi*, 26 (1), 191-199.
Taylor, S. J. ve Letham, B., 2018, Forecasting at scale, *The American Statistician*, 72 (1), 37-45.
Umut, K., Yılmaz, A. ve Dikmen, Y., 2019, Sağlık alanında kullanılan derin öğrenme yöntemleri, *Avrupa Bilim ve Teknoloji Dergisi* (16), 792-808.
Ustalı, N. K., Tosun, N. ve Tosun, Ö. J. E. O. Ü. İ. v. İ. B. D., 2021, Makine öğrenmesi teknikleri ile hisse senedi fiyat tahmini, 16 (1), 1-16.
Yiğit, T., Aksoy, B., Ersoy, M., Şenol, R. ve Salman, O. K. M. J. U. T. B. D., 2021, Petrol fiyatlarının zaman serileri ve LSTM modeli kullanılarak tahminlenmesi, 13 (1), 34-38.

Yıldız, Z., Başkurt, F. ve Süzen, A. A., 2021, İnsan Yürüyüşünün Yapay Zekâyla Sınıflandırılması: Sistematik Bir Gözden Geçirme, *Süleyman Demirel Üniversitesi Sağlık Bilimleri Dergisi*, 12 (1), 110-116.

Yurttabir, A. ve Şen, İ. K., 2021, Finansal Performans Tahmininde Prophet Modeli: İmalat Sektörü Uygulaması, *Journal of Economics Finance Accounting*, 8 (4), 160-166.

EKLER

EK-1 LSTM ve ARIMA Yazılımı

```
import tensorflow as tf import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt from tensorflow.keras.models
import Sequential from tensorflow.keras.layers
import * from keras.layers import Dropout from tensorflow.keras.callbacks
import ModelCheckpoint from tensorflow.keras.losses
import MeanSquaredError from tensorflow.keras.metrics
import RootMeanSquaredError from tensorflow.keras.optimizers
import Adam from tensorflow.keras.models
import load_model from sklearn.metrics
import mean_squared_error as mse from sklearn.preprocessing
import MinMaxScaler

# In[3]:

df = pd.read_excel("veriseti.xlsx") df

# In[4]:

def scale_data(data, columns, scaler): for col in columns: data[col] =
scaler.fit_transform(data[col].values.reshape(-1, 1)) return data

# In[11]:

col_names =
['PRODAMON','PROISLEM','PROKATKD','PROBANKD','OZLDOGTR']

# In[12]:

from sklearn import preprocessing min_max_scaler =
preprocessing.MinMaxScaler()
```

In[13]:

scale_data(df,col_names,min_max_scaler)

In[14]:

def numerik(df, col, name): if not is_numeric_dtype(col): df[name] = col.cat.codes+1

In[18]:

df=df.replace({"K":1,"E":0})

In[19]:

df

In[21]:

train_size = int(len(df) * 0.67)

In[22]:

train_size

In[23]:

train, test = df[0:train_size,:], df[train_size:len(df),:]

In[51]:

values=df['PRODAMON'] values = values.astype('float32') dataset = np.array(values) # Birkaç Değere Bakalım dataset[0:5]

In[61]:

%60 Train % 40 Test TRAIN_SIZE = 0.60 train_size = int(len(dataset) * TRAIN_SIZE) test_size = len(dataset) - train_size train, test = dataset[0:train_size], dataset[train_size:len(dataset)] print("Gün Sayıları (training set, test set): " + str((len(train), len(test))))

In[64]:

def create_dataset(dataset, window_size = 1): data_X, data_Y = [], [] for i in range(len(dataset) - window_size - 1): a = dataset[i:(i + window_size)] data_X.append(a) data_Y.append(dataset[i + window_size]) return(np.array(data_X), np.array(data_Y))

In[65]:

Verisetlerimizi Oluşturalım window_size = 1 train_X, train_Y = create_dataset(train, window_size) test_X, test_Y = create_dataset(test, window_size) print("Original training data shape:") print(train_X.shape)

In[66]:

Yeni verisetinin şekline bakalım. train_X = np.reshape(train_X, (train_X.shape[0], 1, train_X.shape[1])) test_X = np.reshape(test_X, (test_X.shape[0], 1, test_X.shape[1])) print("New training data shape:") print(train_X.shape)

In[67]:

def fit_model(train_X, train_Y, window_size = 1): model = Sequential() # Modelin tek layerlı şekilde kurulacak. model.add(LSTM(100, input_shape = (1, window_size))) model.add(Dense(1)) model.compile(loss = "mean_squared_error", optimizer = "adam") #30 epoch yani 30 kere verisetine bakılacak. model.fit(train_X, train_Y, epochs = 30, batch_size = 1, verbose = 1) return(model) # Fit the first model. model1 = fit_model(train_X, train_Y, window_size)

In[74]:

import math from sklearn.metrics import mean_squared_error

In[86]:

def predict_and_score(model, X, Y): # Şimdi tahminleri 0-1 ile scale edilmiş halinden geri çeviriyoruz. pred = scaler.inverse_transform(model.predict(X)) orig_data = scaler.inverse_transform([Y]) # Rmse değerlerini ölçüyoruz. score = math.sqrt(mean_squared_error(orig_data[0], pred[:, 0])) return(score, pred)

In[87]:

from statsmodels.tsa.arima.model import ARIMA

In[94]:

X = dataset size = int(len(X) * 0.66) train, test = X[0:size], X[size:len(X)] history = [x for x in train] predictions = list() # walk-forward validation for t in range(len(test)): model = ARIMA(history, order=(5,1,0)) model_fit = model.fit() output = model_fit.forecast() yhat = output[0] predictions.append(yhat) obs = test[t] history.append(obs) print('predicted=%f, expected=%f' % (yhat, obs)) # evaluate forecasts rmse = sqrt(mean_squared_error(test, predictions)) print('Test RMSE: %.3f' % rmse)

In[99]:

from numpy import sqrt

In[100]:

rmse = sqrt(mean_squared_error(test, predictions)) print('Test RMSE: %.3f' % rmse)

EK-2 PROPHET Yazılımı

```
from pandas import read_csv from pandas
import to_datetime from pandas
import DataFrame from fbprophet
import Prophet from matplotlib
import pyplot

# In[5]:

from google.colab import files
import pandas as pd
import math import numpy as np

# In[10]:

# Datayı Yükleyelim\n", df = pd.read_excel('veriseti.xlsx', date_parser=[0]) # İlk 5 Satır\n", df.head()

# In[15]:

df.drop('PROISLEM',inplace=True, axis=1)

df.head()

# In[16]:

df.drop('OZLCINSI',inplace=True, axis=1) df.drop('OZLDOGTR',inplace=True, axis=1) df.drop('PROBANKD',inplace=True, axis=1) df.drop('PROKATKD',inplace=True, axis=1) df.head()

# In[17]:

df = df.rename(columns={'PROTIME': 'ds', 'PRODAMON': 'y'})
```

In[18]:

df.columns = ['ds', 'y'] df['ds']= to_datetime(df['ds'])

In[19]:

model = Prophet() model.fit(df)

In[21]:

future = list() for i in range(1, 13): date = '1968-%02d' % i future.append([date])

In[25]:

from sklearn.metrics import mean_squared_error future = DataFrame(future) future.columns = ['ds'] future['ds']= to_datetime(future['ds']) forecast = model.predict(future) y_true = df['y'][-12:].values y_pred = forecast['yhat'].values rmse = mean_squared_error(y_true, y_pred)

In[28]:

print('RMSE: %.3f' % rmse)