T.C.
DÜZCE ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

MAKİNE ÖĞRENMESİ TABANLI GERÇEK ZAMANLI MEDİKAL NESNELERİN İNTERNETİ ÇERÇEVESİNİN GELİŞTİRİLMESİ

EMRE YILDIRIM

DOKTORA TEZİ
ELEKTRİK-ELEKTRONİK VE BİLGİSAYAR MÜHENDİSLİĞİ
ANABİLİM DALI

DANIŞMAN
DOÇ. DR. ALİ ÇALHAN

DÜZCE, 2022

T.C.
DÜZCE ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

MAKİNE ÖĞRENMESİ TABANLI GERÇEK ZAMANLI MEDİKAL NESNELERİN İNTERNETİ ÇERÇEVESİNİN GELİŞTİRİLMESİ

Emre YILDIRIM tarafından hazırlanan tez çalışması aşağıdaki jüri tarafından Düzce Üniversitesi Lisansüstü Eğitim Enstitüsü Elektrik-Elektronik ve Bilgisayar Mühendisliği Anabilim Dalı'nda **DOKTORA TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Doç. Dr. Ali ÇALHAN

Düzce Üniversitesi

Jüri Üyeleri

Doç. Dr. Ali ÇALHAN
Düzce Üniversitesi _____

Doç. Dr. Devrim AKGÜN
Sakarya Üniversitesi _____

Doç. Dr. Muhammed Enes BAYRAKDAR
Düzce Üniversitesi _____

Doç. Dr. Murtaza CİCİOĞLU
Bursa Uludağ Üniversitesi _____

Dr. Öğr. Üyesi Arafat ŞENTÜRK
Düzce Üniversitesi _____

Tez Savunma Tarihi: 07/06/2022

BEYAN

Bu tez çalışmasının kendi çalışmam olduğunu, tezin planlanmasından yazımına kadar bütün aşamalarda etik dışı davranışımın olmadığını, bu tezdeki bütün bilgileri akademik ve etik kurallar içinde elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara kaynak gösterdiğimi ve bu kaynakları da kaynaklar listesine aldığımı, yine bu tezin çalışılması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını beyan ederim.

07 Haziran 2022

Emre YILDIRIM

TEŞEKKÜR

Yıllar boyunca büyük emek ve özenle hazırladığım doktora tezimi tamamlamanın mutluluğunu yaşamaktayım. Bu bölümü doktora öğrenimim boyunca bana desteklerini esirgememiş ve teşvik etmiş insanlara teşekkür etmek için kullanacağım.

Öncelikle doktora öğrenimimde ve bu tezin hazırlanmasında gösterdiği her türlü destek, özveri ve yardımlarından dolayı çok değerli hocam Doç. Dr. Ali ÇALHAN'a danışmanlığı için en içten dileklerimle teşekkür ederim.

Tez çalışmam boyunca tez izleme komitemde yer alarak değerli görüşleri ile araştırmanın şekillenmesini sağlayan Doç. Dr. Devrim AKGÜN'e ve Doç. Dr. Muhammed Enes BAYRAKDAR hocalarıma teşekkürlerimi sunarım. Bunun yanında, tez çalışmam boyunca bilgi ve tecrübelerini esirgemeyerek çalışmalarıma çok değerli katkılar yapan ve tez savunmama gelerek tezin son şeklini almasını sağlayan değerli hocam Doç. Dr. Murtaza CİCİOĞLU'na teşekkürü bir borç bilirim.

Tez çalışması süresince fedakarlık gösteren ve bu süreçte bana her zaman destek olan sevgili eşim Tuba SAYGILI YILDIRIM'a sonsuz teşekkürlerimi sunarım.

Tüm çalışmalarım süresince beni cesaretlendiren ve yetiştiren çok kıymetli annem Hatice YILDIRIM'a ve babam Eyüp YILDIRIM'a ithaf olunur.

Özellikle çalışmalarım dolayısıyla ona daha fazla vakit ayıramadığım canım kızım Masal'a bana sevgisiyle bu süreçte destek olduğu için bu çalışmayı ona hediye etmek isterim.

07 Haziran 2022 **Emre YILDIRIM**

İÇİNDEKİLER

Sayfa No

ŞEKİL LİSTESİ

Sayfa No

ÇİZELGE LİSTESİ

Sayfa No

KISALTMALAR

AI	Yapay Zeka
ANFIS	Uyarlanabilir Ağ Tabanlı Bulanık Çıkarım Sistemi
AODV	Geçici İsteğe Bağlı Mesafe Vektör Yönlendirme
AP	Erişim Aşaması
CAP	Çekişmeli Erişim Aşamaları
CART	Sınıflandırma ve Regresyon Ağacı
ÇP	Çatışma Penceresi
DAG	Yönlendirilmiş Döngüsel Grafik
DF	Data Frame
DVM	Destek Vektör Makinesi
EAP	Acil Erişim Aşaması
ECG	Elektrokardiyogram
EEG	Elektroensefalogram
GA	Gradyan Artırma
GUI	Grafik Kullanıcı Arayüzü
HDFS	Hadoop Dağıtık Dosya Sistemi
IoMT	Medikal Nesnelerin İnterneti
IoT	Nesnelerin İnterneti
IP	İnternet Protokolü
KA	Karar Ağacı
k-NN	K-En Yakın Komşu
KVAA	Kablosuz Vücut Alan Ağı
LR	Lojistik Regresyon
NB	Naive Bayes
PC	Kişisel Bilgisayar
RAP	Rastgele Erişim Aşaması
RDD	Esnek Dağıtılmış Dosya Sistemi
RO	Rastgele Orman
RREP	Rota Cevabı
RREQ	Rota Talebi
SA	Sinir Ağı
VR	Sanal Gerçeklik
YSA	Yapay Sinir Ağı

SİMGELER

$\sum$	Sayı Dizisi Toplamı
f	Fonksiyon
log	Logaritma
p	Olasılık
S	Öznitelik
T	Hedef

ÖZET

MAKİNE ÖĞRENMESİ TABANLI GERÇEK ZAMANLI MEDİKAL NESNELERİN İNTERNETİ ÇERÇEVESİNİN GELİŞTİRİLMESİ

Emre YILDIRIM
Düzce Üniversitesi
Lisansüstü Eğitim Enstitüsü, Elektrik-Elektronik ve Bilgisayar Mühendisliği Anabilim Dalı
Doktora Tezi
Danışman: Doç. Dr. Ali ÇALHAN
Haziran 2022, 84 sayfa

Yeni koronavirüs hastalığı (COVID-19), birçok alanda olduğu gibi sağlık sektöründe de Medikal Nesnelerin İnterneti (IoMT), Kablosuz Vücut Alan Ağları (KVAA) ve Bulut bilişim gibi yeni teknolojilere olan ihtiyacı arttırmıştır. Bu teknolojiler ayrıca milyarlarca cihazın internete bağlanmasını, birbirleriyle iletişim kurmasını ve her yerden ve her zaman erişilebilen yeni servisleri kullanmasını mümkün kılmıştır. Bu çalışmada, KVAA'lardan oluşan yeni bir IoMT çerçevesi önerilmiştir. KVAA'lardan gelen sağlık büyük verileri ile Sis (Fog) ve Bulut (Cloud) bilişim teknolojileri kullanılarak analizler gerçekleştirilmiştir. Hızlı ve kolay analizler için Sis bilişim, zaman alan ve karmaşık analizler için ise Bulut bilişim kullanılmıştır. Önerilen IoMT çerçevesi, bir diyabet ve bir kalp hastalığı tahmin senaryosu ile sunulmuştur. Hastalıkların tahmin sürecinde, bulanık mantık karar verme ile Sis bilişim üzerinde tahminler gerçekleştirilmiştir. Bulut bilişimde ise gerçek zamanlı veri akışında Node-Red ve Apache Kafka, gerçek zamanlı veri analizinde Apache Spark ve büyük verileri kolaylıkla depolayabilen MongoDB yapıları kullanılmıştır. Bununla birlikte, Apache Spark makine öğrenmesi kütüphanesi olan MLlib'in içerisindeki Karar Ağacı (KA), Rastgele Orman (RO), Gradyan Artırma (GA), Lojistik Regresyon (LR) ve Destek Vektör Makinesi (DVM) makine öğrenmesi sınıflandırma algoritmaları ile gerçek zamanlı veri analizleri yapılarak tahminler gerçekleştirilmiştir. Ayrıca, kullanılan sınıflandırma algoritmalarının performans karşılaştırmaları yapılmıştır. Sonuçlar incelendiğinde, bulanık mantık kullanılan Sis bilişimde diyabet için %64 doğruluk performansı ve Bulut bilişim de ise KA, RO, GA, LR ve DVM algoritmaları diyabet hastalığı için sırasıyla %78,77, %77,40, %84,93, %80,14 ve %79,45 doğruluk, kalp hastalığı için ise sırasıyla %89,74, %92,31, %94,87, %88,46 ve %93,60 doğruluk ile tahmin performansları sunmuştur. Ayrıca IEEE 802.15.6 standardı ve AODV yönlendirme protokolü kullanılarak oluşturulan KVAA senaryosundaki farklı önceliklere sahip heterojen düğümlerin iş çıkarım ve gecikme sonuçları da analiz edilmiştir. Bununla birlikte, veri analitiğinde kullanılan Apache Spark gerçek zamanlı veri işleme performansı incelenmiştir. Elde edilen tüm sonuçlar, önerilen yaklaşımın yüksek verimliliği ve fizibilitesini kanıtlamıştır.

Anahtar sözcükler: Apache spark, Bulut bilişim, Makine öğrenmesi, Medikal nesnelerin interneti (IoMT), Sis bilişim, Veri analitiği

ABSTRACT

DEVELOPMENT OF MACHINE LEARNING-BASED REAL-TIME MEDICAL INTERNET OF THINGS FRAMEWORK

Emre YILDIRIM
Düzce University
Institute of Graduate Studies, Department of Electric-Electronic and Computer Engineering
Doctoral Thesis
Supervisor: Assoc. Prof. Dr. Ali ÇALHAN
June 2022, 84 pages

The novel coronavirus disease (COVID-19) has increased the need for new technologies such as the Internet of Medical Things (IoMT), Wireless Body Area Networks (WBANs) and cloud computing in the healthcare industry as well as in many areas. These technologies have also made it possible for billions of devices to connect to the internet, communicate with each other, and use new services that can be accessed from anywhere and anytime. In this study, a new IoMT framework consisting of WBANs is proposed. Analyzes are carried out using health big data from WBANs and Fog and Cloud computing technologies. Fog computing is used for fast and easy analysis, and Cloud computing is used for time-consuming and complex analyses. The proposed IoMT framework is presented with a heart disease and a diabetes prediction scenario. In the disease prediction process, predictions are made on Fog computing with fuzzy logic decision making. In Cloud computing, Node-Red and Apache Kafka are used in real-time data flow, Apache Spark in real-time data analysis and MongoDB structures that can easily store big data are used. However, the Apache Spark machine learning library MLlib's Decision Tree (DT), Random Forest (RF), Gradient Boosting (GB), Logistic Regression (LR) and Support Vector Machine (SVM) machine learning classification algorithms are combined with real-time classification algorithms. Estimates are made by performing data analysis. In addition, the performance comparisons of the classification algorithms used are done.

When the results are examined, it is seen that 64% accuracy performance for diabetes in Fog computing using fuzzy logic and DT, RF, GB, LR and SVM algorithms in cloud computing have accuracy as 78,77%, 77,40%, 84,93%, 80,14%, and 79,45% for diabetes, respectively. In addition to that offers 89,74%, 92,31%, 94,87%, 88,46%, and 93,60% accuracy for heart disease, respectively. In addition, the throughput and delay results of heterogeneous nodes with different priorities in the WBAN scenario created using the IEEE 802.15.6 standard and AODV routing protocol are also analyzed. However, Apache Spark real-time data processing performance, which is used in data analytics, is examined. All the results obtained proved the high efficiency and feasibility of the proposed approach.

Keywords: Apache spark, Cloud computing, Data analytics, Fog computing, Internet of medical things (IoMT), Machine learning

EXTENDED ABSTRACT

DEVELOPMENT OF MACHINE LEARNING-BASED REAL-TIME MEDICAL INTERNET OF THINGS FRAMEWORK

Emre YILDIRIM
Düzce University
Institute of Graduate Studies, Department of Electric-Electronic
and Computer Engineering
Doctoral Thesis
Supervisor: Assoc. Prof. Dr. Ali ÇALHAN
June 2022, 84 pages

1. INTRODUCTION

New technologies can offer fast and effective solutions in health as well as in every field. As it is known, the COVID-19 pandemic, along with the rapidly increasing number of cases around the world, has changed our lives, communication, and business models as well as the health threat. In particular, this change affected the health sector the most. The rapidly increasing number of patients is reflected as a serious capacity burden on health systems around the world. This shows that new technological approaches in the field of health are needed more than ever. Augmented reality (VR), artificial intelligence (AI), sensor technologies, data analytics, cloud communication, machine learning and internet of things (IoT) are some of them.

IoMT is one of the sub-branches of IoT as remote health systems. IoMT includes wireless communication technologies, wearable health devices, sensor/actuator circuits, remote health monitoring, and IoT software and tools. IoMT aims to create an intelligent framework for healthcare. Moreover, the IoMT framework is a complex architecture and interconnects various components that interact with each other, offering many solutions to end users. IoMT enhances human-machine interaction and real-time health monitoring solutions that improve patient participation in decision making. IoMT applications have important advantages such as cost reduction in healthcare systems, real-time interventions in emergencies, and remote monitoring in pandemic environments.

The IoMT concept evolves with existing medical devices, IoT peripherals, remote healthcare software and hardware. As an IoT concept, IoMT devices feature internet protocol (IP) to communicate with medical professionals and healthcare personnel via online servers. Another key issue for IoMT is that it generates massive health data. Large

health data from IoMT devices are stored in local or cloud databases. For the analysis of big data, advanced algorithms should be used with the help of machine learning, deep learning and artificial intelligence for classification, correlation, or prediction.

In this study, we propose an IoMT framework with the components and functions mentioned above. For this purpose, first of all, Riverbed Modeler simulation software performs communication within/between WBAN with all necessary design parameters. Next, a gateway with socket programming capabilities transmits the generated health data from the Riverbed Modeler to other IoMT peripherals. Predictions of diabetes and heart disease are then performed in the fog and cloud nodes. In fog computing, predictions are made with fuzzy logic. In cloud computing, real-time predictions are made by using Apache Spark data processing engine and machine learning algorithms over the developed data flow platforms. In addition, data sent from simulation scenarios and disease prediction results are presented in real time in the visualization component and stored in the storage component.

2. MATERIAL AND METHODS

The proposed IoMT framework are developed for real-time prediction of diabetes and heart disease. The proposed IoMT framework consists of three layers: Data source layer, Fog, and Cloud computing. In the first layer, the data source layer, health data for diabetes and heart are produced in real-time in WBAN scenarios created on the Riverbed Modeler simulation program. In the second layer Fog computing, real-time diabetes disease prediction is made with the fuzzy logic prediction system by using the health data produced in the simulation scenarios. In the third layer Cloud computing layer, first of all, health data produced from simulation scenarios are taken with the help of data flow components (Node-Red and Apache Kafka). Real-time health data is transferred to the Apache Spark-based data analytics layer, where machine learning algorithms (MLlib) are used. In the Apache Spark data analytics layer, firstly, a prediction model is developed for disease prediction using Decision Tree (DT), Random Forest (RF), Gradient Boosting (GB), Logistic Regression (LR) and Support Vector Machine (SVM) classification algorithms. Then, Apache Spark's Streaming feature data is taken in real-time and disease predictions are made with the developed prediction model. Disease predictions are performed on Apache Spark with a distributed computing architecture. Also, in the data visualization and storage component, the health data sent from the simulation scenarios and the disease prediction results are presented in real time on the Live Web Panel via

Apache Kafka. However, these data are stored in the database created in the MongoDB NoSQL database management system to be used for later analysis.

3. RESULTS AND DISCUSSIONS

We propose an IoMT framework with the help of WBAN, Cloud and Fog computing, fuzzy logic and machine learning algorithms (DT, RF, GB, LR and SVM) in diabetes and heart disease prediction scenario. We discussed the proposed IoMT framework with all the components. Fuzzy logic in the Fog computing layer provides 64% accuracy performance in the diabetes system. In Cloud computing, DT, RF, GB, LR and SVM have 78,77%, 77,40%, 84,93%, 80,14%, and 79,45% accuracy performance for diabetes prediction, respectively. For heart disease prediction, DT, RF, GB, LR and SVM have 89,74%, 92,31%, 94,87%, 88,46%, ve 93,60% accuracy performances, respectively. Considering the diabetes and heart disease prediction performances, it is seen that the GB algorithm has the best performance in diabetes and heart disease prediction. However, RF algorithm has the lowest performance in diabetes prediction, while LR algorithm has the lowest performance in heart disease prediction. The reason for these estimation values to be different may be the suitability of the data set used. If the data set has appropriate data and parameters for that disease, the performance of the algorithms will have much better predictive values.

In the proposed IoMT framework, diabetes and heart disease simulation scenarios are created in Riverbed Modeler using CSMA-CA-based IEEE 802.15.6 protocol and AODV routing algorithm. The end-to-end latency and throughput results of heterogeneous nodes with various priorities are comparatively examined. In the light of the results obtained, it has been determined that vital data is transmitted to the target with lower delay and high throughput rate. In addition, it has been seen that the Node-Red and Apache Kafka structures used in the data flow component transmit all the data produced in the simulation scenarios to the data analytics component in a complete and fast manner. Apache Spark real-time forecasting model provides effective and efficient performance especially in real-time applications such as IoMT frameworks.

4. CONCLUSION AND OUTLOOK

Remote health monitoring and diagnosis services are gaining importance day by day. Increasing health costs and the number of intensive care patients, the decrease in empty hospital beds and the number of health workers per capita, especially in pandemic

situations, prompt governments to take precautions. The proposed IoMT framework is a promising solution for the aforementioned problems.

In the light of the results obtained, it has been determined that vital data is transmitted to the target with lower delay and high throughput. Apache Spark real-time prediction model provides effective and efficient performance especially in real-time applications such as IoMT frameworks. The developed IoMT framework can be used in hospitals, nursing homes, etc. allows it to be used in communities such as for future studies, various machine learning algorithms and different diseases can be considered to create an IoMT framework for various pandemic and health applications. In addition, the Apache Spark data processing engine is comparable in performance to similar applications.

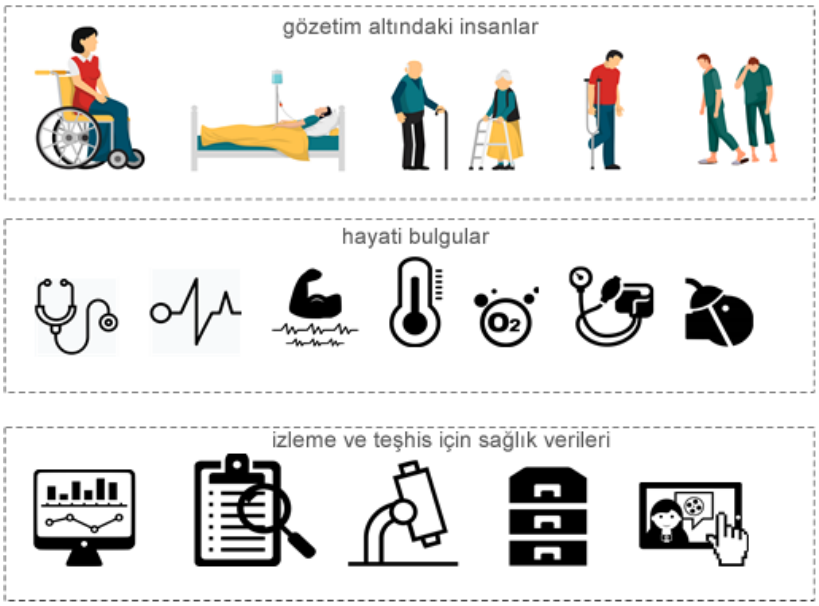
1. GİRİŞ

1.1. AMAÇ VE KAPSAM

Yeni teknolojiler her alanda olduğu gibi sağlıkta da hızlı ve etkili çözümler sunabilmektedir [1], [2]. Bilindiği üzere COVID-19 pandemisi, dünya genelinde hızla artan vaka sayısı ile birlikte hayatımızı, iletişimimizi ve iş modellerimizi değiştirdiği gibi sağlık tehdidini de değiştirmiştir [3], [4]. Özellikle bu değişim en çok sağlık sektörünü etkiledi. Hızla artan hasta sayısı, dünyadaki sağlık sistemlerine ciddi bir kapasite yükü olarak yansımaktadır. Bu da sağlık alanında yeni teknolojik yaklaşımlara her zamankinden daha fazla ihtiyaç duyulduğunu göstermektedir. Artırılmış gerçeklik (VR) [5], yapay zeka (AI) [6], sensör teknolojileri [7], [8], veri analitiği [9], [10], bulut iletişimi [11]–[13], makine öğrenimi [14]–[18] ve nesnelerin interneti (IoT) [19], [20] bunlardan bazılarıdır.

IoMT, uzak sağlık sistemleri olarak IoT'nin alt dallarından biridir [21]. IoMT, kablosuz iletişim teknolojileri, giyilebilir sağlık cihazları, sensör/aktüatör devreleri, uzaktan sağlık izleme ve IoT yazılım ve araçlarını içermektedir [22], [23]. IoMT, sağlık hizmetleri için akıllı bir çerçeve oluşturmayı amaçlar. Ayrıca, IoMT çerçevesi karmaşık bir mimaridir ve birbiriyle etkileşime giren çeşitli bileşenleri birbirine bağlar ve son kullanıcılara birçok çözüm sunar [24]. IoMT, karar vermede hasta katılımını, gerçek zamanlı sağlık izleme çözümlerini ve insan-makine etkileşimini geliştirir [25]. IoMT uygulamalarının sağlık sistemlerinde maliyet düşürme, acil durumlarda gerçek zamanlı müdahaleler ve pandemi ortamlarında uzaktan izleme gibi önemli avantajları vardır [26].

IoMT konsepti, mevcut tıbbi cihazlar, IoT çevre birimleri, uzaktan sağlık yazılımı ve donanımının birlikte kullanımı ile gelişmektedir. Bir IoT konsepti olarak IoMT cihazları, tıp uzmanları ve sağlık personeli ile çevrimiçi sunucular aracılığıyla iletişim kurmak için internet protokolü (IP) özelliklerine sahiptir [27]. IoMT için bir diğer önemli konu, büyük sağlık verileri oluşturmasıdır. IoMT cihazlarından gelen büyük sağlık verileri, yerel veya bulut veri tabanlarında depolanır [28]. Büyük verilerin analizi için, sınıflandırma, korelasyon veya tahmin için makine öğrenmesi, derin öğrenme ve yapay zeka yardımıyla gelişmiş algoritmalar kullanılmalıdır [29], [30].



Şekil 1.1. IoMT bileşenleri.

IoMT'nin temel bileşenleri Şekil 1.1.'de gösterilmiştir. Gözetim altındaki kişiler vücutlarının üzerinde/içinde/etrafında çeşitli sensörlerle donatılmıştır. Sensörler, yetenekleri sayesinde yaşamsal belirtileri algılarlar ve kablolu veya kablosuz iletişim araçlarına sahiptirler. Ayrıca sensörler, yaşamsal belirtilerin sürekli veya olaya dayalı algılama görevleriyle programlanmıştır. Bir koordinatör düğüm, bir merkezi düğüm olarak vücutta bulunan sensörlerin koordinasyonundan sorumludur. IoMT'nin diğer bileşenlerine veri göndermek için bir ağ geçidi olarak sensörlerden gelen verileri toplar. Temel olarak, koordinatör bir kişisel bilgisayar (PC), akıllı telefon, tablet veya normal bir sensör düğümü olabilir [31]. IoMT'nin özel ve önemli sensör ağı bileşeni olan IEEE 802.15.6 standardı, koordinatörü bir HUB düğümü olarak adlandırır [32]. Koordinatör 4G/5G, Wi-Fi, Bluetooth vb. bağlantılar gibi bazı üstün yeteneklere sahiptir. Bir koordinatörün en önemli görevi, toplanan verileri uzak merkezlere göndermektir [31]. Uzak merkez, ikincil bir depolama, veri tabanı, izleme cihazı, Sis veya Bulut bilgi işlem düğümü olabilir. Bu aşamada gerekirse büyük veri analizi başlamaktadır.

Bu çalışmada, yukarıda bahsedilen bileşen ve fonksiyonlara sahip bir IoMT çerçevesi önermekteyiz. Bu amaçla öncelikle Riverbed Modeler benzetim yazılımı, gerekli tüm tasarım parametreleri ile KVAA içi/arası haberleşmeyi gerçekleştirmektedir. Ardından, soket programlama özelliklerine sahip bir ağ geçidi, oluşturulan sağlık verilerini Riverbed Modeler'dan diğer IoMT çevre birimlerine iletmektedir. Sonrasında, Sis ve Bulut bilişim katmanlarında diyabet ve kalp hastalığına dair hastalık tahminleri yapılmaktadır. Sis

bilişimde bulanık mantık sistemi ile diyabet tahmini yapılmaktadır. Bulut bilişimde ise geliştirilen veri akış platformları üzerinden Apache Spark veri işleme motoru ile makine öğrenmesi algoritmaları kullanılarak gerçek zamanlı olarak diyabet ve kalp hastalığı tahminleri gerçekleştirilmektedir. Ayrıca, benzetim senaryolarından gönderilen sağlık verileri ve hastalık tahmin sonuçları görselleştirme bileşeninde gerçek zamanlı olarak sunulmakta ve depolama bileşeninde depo edilmektedir.

Önerilen çalışmanın katkıları şu şekilde özetlenebilir:

- Sis ve Bulut bilişim yetenekleri gibi önemli katmanlarla yetenekli bir IoMT çerçevesi tasarlanmıştır.
- IoMT'nin daha gerçekçi bir performans analizi için Riverbed Modeler'daki KVAA'lardan gerçek zamanlı veriler elde edilmiştir.
- Diyabet ve kalp hastalığı tahminleri için Sis bilişimde bulanık mantık ve Bulut bilişimde makine öğrenmesi algoritmaları (KA, RO, GA, LR ve DVM) kullanılmıştır.
- Apache Spark MLlib makine öğrenmesi algoritmaları ile yapılan hastalık tahmini performansları karşılaştırılmıştır.
- Diyabet ve kalp hastalığının tahmin sonuçları gerçek zamanlı Canlı Web Panel'de görüntülenmiş ve MongoDB veri tabanında depolanmıştır.
- Apache Spark veri analitiği sistemi ile dağıtık hesaplamalar yapılmıştır.
- IEEE 802.15.6 protokolü ve AODV yönlendirme algoritması kullanılarak geliştirilen benzetim senaryoların uçtan uca gecikme ve iş çıkarım performansları incelenmiştir.

1.2. LİTERATÜRDEKİ BENZER ÇALIŞMALAR

Son yıllarda sağlık hizmetlerinde kullanılan teknolojilerde üretilen yüksek hacimli veriler, büyük veri analitiği, veri madenciliği ve makine öğrenmesi gibi araştırma alanları önemli bir konu haline gelmiştir. Sağlık hizmetlerinin iyileştirilmesi, maliyetlerin düşürülmesi ve bunun sonucunda erken teşhis gibi insan hayatını kolaylaştıracak çalışmalar yapılmaktadır. Literatürde IoMT, veri analizi, bulanık mantık ve makine öğrenmesi algoritmaları ile ilgili çeşitli çalışmalar bulunmaktadır. Ancak, yapılan

çalışmalar incelendiğinde bu konuların birlikte ele alınmadığı görülmüştür.

Bu çalışmada, yukarıda bahsedilen tüm konuların yanı sıra Sis ve Bulut kavramlarını da içeren bir IoMT çerçevesi önerilmektedir. Bu bölümde ise sağlık hizmetlerine yönelik çeşitli tahmin ve öneri sistemleri incelenmektedir.

Niswati vd. [33] bir kişinin diyabet riskini beş değişken kullanarak teşhis eden bulanık mantık temelli bir karar destek sistemi önerdiler: glikoz, diyastolik kan basıncı, vücut kitle indeksi ve yemekten 2 saat sonra ailede diyabet. Karar destek sistemi, MATLAB'da Mamdani çıkarım yöntemine göre programlanmıştır. Ayrıca sistemin kullanımı için bir Grafik Kullanıcı Arayüzü (GUI) geliştirilmiştir. Deneysel sonuçlar, karar destek sisteminin %100 doğru teşhis ile çalıştığını göstermektedir.

Bressan vd. [34], Diabetes Mellitus tip 2 glisemik indeks seviyesini gösteren otomatik bir bulanık sınıflandırma sistemi önermektedir. Bulanık tabanlı sistem, glisemik indeks seviyesini dört farklı şekilde gösterir: çok azalt, azalt, sabit tut ve artır. Ayrıca KA algoritması, bulanık sınıflandırma yapılırken karar sisteminde destek olarak kullanılmaktadır. Sonuçlar, bulanık sınıflandırma sisteminin umut verici olduğunu ve doktorların karar verme sürecine yardımcı olduğunu göstermektedir.

Zou vd. [35] Luzhou, Çin'de bulunan bir hastanede fizik muayene verilerini kullanarak Diabetes Mellitus'u öngören analizler yaptılar. Diabetes Mellitus'u tahmin etmek için RO, KA ve Sinir Ağı (SA) sınıflandırma algoritmalarını kullandılar. 5 kat çapraz doğrulama kullanılarak yapılan değerlendirmede, RO algoritması %80 doğrulukla en iyi tahmin sonucuna sahip olduğu görülmüştür.

Muhammed vd. [36], Tip 2 diyabet verilerini kullanarak denetimli makine öğrenimi algoritmalarıyla bir tahmin modeli geliştirmek için LR, DVM, K-en yakın komşu (k-NN), RO, Naive Bayes (NB) ve GB kullandılar. Çalışmada, Nijerya'daki Uzman Dr. Murtala Muhammed Kano Hastanesi'nde yapılan tahminlerde RO algoritması %88,76 doğruluk oranı ile en iyi performansa sahiptir.

Yuvaraj ve SriPreethaa [37] diyabetin sınıflandırılması için Hadoop tabanlı bir uygulama geliştirdi. Uygulamada, sınıflandırma için üç farklı makine algoritması kullanan bir model önerdiler: NB, KA ve RO. Pima Kızılderili'lerinin diyabet verilerinin kullanıldığı çalışmada RO algoritması %94 doğrulukla en iyi sınıflandırmayı yapmıştır.

Ramesh vd. [38], kişisel sağlık cihazlarından ve akıllı giyilebilir cihazlardan gelen sağlık verilerini birleştirerek diyabetin erken teşhisi için IoT tabanlı bir çerçeve önerdi.

Çalışmalarında diyabet tahmini için DVM sınıflandırma algoritmasının kullanıldığı bir model bulunmaktadır. Önerdikleri çerçevede, 10 kat çapraz doğrulama yöntemi ile doğruluk, duyarlılık ve özgüllük performans ölçümlerinin sonuçları sırasıyla %83,20, %87,20 ve %79 olarak elde edilmiştir.

Tan ve Halim [39], çalışmalarında diyabet ve böbrek hastalığını öngören IoT tabanlı bir uzaktan sağlık izleme sistemi önerdiler. Önerilen sistem mimarisinde bazı sensörler vücut ısısı, nabız ve kan basıncı gibi üç temel hayati değeri ölçer. Sensör'lerden alınan veriler sistemin bulut yapısında saklanmakta ve hastalık tahmini yapılmaktadır. Hastalık tahmini için Yapay Sinir Ağı (YSA) algoritması kullanılır. Diyabet ve böbrek hastalığı tahminlerinin doğruluk oranları sırasıyla %90,54 ve %87,88 idi.

Devarajan vd. [40] kullanıcının diyabet riskini tahmin etmek için Sis destekli gerçek zamanlı bir sağlık sistemi önerdiler. Önerilen sağlık sistemlerinde fizyolojik değerler, sürekli gerçek zamanlı izleme için IoT akıllı telefonlar aracılığıyla bulut sunucusuna iletilir. Tahmin, Sis katmanında kan şekerinin risk seviyesinde olup olmadığını belirlemek için J48Graft karar ağacı algoritması ile yapılır. Deneysel sonuçlar incelendiğinde J48Graft karar ağacı diyabetin risk düzeyini %98,56 gibi yüksek bir sınıflandırma doğruluğu ile tahmin etmektedir. Ayrıca önerilen sistemin enerji verimliliği, tahmin doğruluğu, hesaplama karmaşıklığı ve gecikme süresi açısından başarılı olduğu görülmektedir.

Abdel-Basset vd. [41], tip 2 diyabet hastalarını tespit etmek ve izlemek için yeni bir IoT tabanlı çerçeve önerdi. Önerilen çerçeve üç ana katmandan oluşuyor: giyilebilir IoT sensörleri, Sis katmanı ve Bulut katmanı. KVAA ve mobil uygulamalar, kullanıcıların fizyolojik değişikliklerini izlemek ve değişen verileri toplamak için kullanılmaktadır. Kullanıcının diyabetli olup olmadığını tespit etmek için önerilen çerçevede yeni bir hibrit karar destek sistemi geliştirilmiş ve karar destek sistemi ile analizler yapılmıştır. Analiz sonucunda önerilen sistemin olumlu sonuçlar verdiği görülmüştür.

Khan vd. [42], COVID-19'a maruz kalan herkes tarafından kullanılabilecek bir IoMT tabanlı akıllı izleme sistemi önerdi. Sistem MATLAB aracı ile Mamdani bulanık çıkarım yöntemi kullanılarak geliştirilmiştir. Önerilen sistem, doğrudan Covid-19 ile ilgili kullanıcı verilerini toplayacak, sağlık durumunu değerlendirecek ve karantina için bir uzmana danışma ihtiyacı olup olmadığını bildirecektir. Sistemlerinin doğruluğu yaklaşık %83'tür.

Otom vd. [43] gerçek zamanlı bir Covid-19 tespit ve izleme sistemi önerdi. Önerilen sistem, şüpheli Covid-19 vakalarını erken tespit etmek, tedaviyi izlemek ve hastalığı anlamak için geliştirilmiştir. Ayrıca, gerçek zamanlı belirti verilerini toplamak için IoT çerçevesi kullanılır. Sistem, belirti verisi toplama ve yükleme, karantina/izolasyon merkezi, veri analiz merkezi, sağlık hekimleri ve bulut altyapısı olmak üzere beş bileşenden oluşmaktadır. Sistemde COVID-19 vakalarının tespiti için DVM, NN, NB, k-NN, Decision Table (DeT), Decision Stump, OneR ve ZeroR sınıflandırma algoritmaları kullanılmaktadır. Sonuçlar, bu sekiz algoritmadan beşinin %90'ın üzerinde bir doğruluk elde ettiğini göstermektedir.

Kumar ve Gandhi [44], giyilebilir IoT sensörlerinden gelen büyük hacimli verileri depolamak ve analiz etmek için bir sağlık izleme sistemi önerdi. Önerilen sistemleri üç katmanlı bir mimariden oluşur: veri toplama, veri depolama ve veri analitiği. İlk katman, IoT sensörleri tarafından oluşturulan verileri toplar. İkinci katman, Apache HBase kullanarak büyük hacimli sensör verilerini depolar. Son olarak, üçüncü katman, Apache Mahout kullanarak LR tabanlı modelle analiz ederek kalp hastalıklarını analiz eder. Kalp hastalıklarının tespiti için sistemde Alıcı Çalışma Karakteristiği (ROC) analizleri yapılmaktadır.

Kamarajugadda vd. [45] kalp hastalığı tahmini için Biyocoğrafya tabanlı Optimizasyon-Destek Vektör Makinesi (BBO-SVM) modelini kullanan yeni bir IoMT tabanlı sağlık sistemi önerdiler. Önerilen model, BBO algoritmasını kullanan DVM'nin parametre ayarlarını içerir. Önerilen modeli değerlendirmek için Statlog Kalp Hastalığı veri seti kullanılmıştır. Ayrıntılı analizde önerilen BBO-SVM modeli %89,26 doğruluk, %88,33 kesinlik, %87,60 duyarlılık, %87,96 F-skoru ve %78,27 Kappa değerleri ile olumlu sonuçlar elde etmiştir.

Khan ve Algarni [46], Modifiye Salp Sürü Optimizasyonu (MSSO) ve Uyarlamalı ağ tabanlı bulanık çıkarım sistemi (ANFIS) kullanarak kalp hastalığının teşhisi için bir IoMT sistemi önerdi. Önerdikleri sistemde, analizleri daha etkin hale getirmek için veri ön işleme aşamasında Levy Flight algoritması kullanılmıştır. Analiz sonuçlarına göre önerilen sistem kalp hastalığını %99,45 doğruluk ve %96,54 kesinlik ile tahmin etmektedir.

Önerilen çalışma ile birlikte ilgili çalışmaların karşılaştırmalı bir özeti Çizelge 1.1.'de verilmiştir. Diyabet ve diğer hastalıklara dayalı çalışmalar [33]-[46]'da verilmiştir. İlgili

çalışmalar incelendiğinde, çalışmamızda önerilen IoMT çerçevesine benzer bir çalışma bulunmamaktadır. Önerilen IoMT çerçevesi, özellikle pandemi durumlarında Sis ve Bulut bilişim yetenekleri ve KVAA iletişimleri ile genişletilebilir bir uygulama ortamı gibi üstün özelliklere sahiptir. Performans değerlendirmesi için hem bulanık mantık hem de makine öğrenmesi algoritmaları dikkate alınmış ve önerilen çerçevede gerçek zamanlı diyabet ve kalp hastalığı tahminleri gerçekleştirilmiştir.

Çizelge 1.1. Önerilen IoMT çerçevesinin ilgili çalışmalar ile karşılaştırılması.

No	Çalışmalar	Hastalık Tipi	Makine Öğrenmesi	Bulanık Mantık	Sis / Bulut Bilişim	IoT
1	Niswati ve ark. [33]	Diyabet	x	√	x	x
2	Bressan ve ark. [34]	Diyabet	KA	√	x	x
3	Zou ve ark. [35]	Diyabet	RO, KA, NN	x	x	x
4	Muhammad ve ark. [36]	Diyabet	LR, DVM, k-NN, RO, NB, GB	x	x	x
5	Yuvaraj ve SriPreethaa [37]	Diyabet	NB, KA, RO	x	x	x
6	Ramesh ve ark. [38]	Diyabet	DVM	x	√	√
7	Tan ve Halim [39]	Diyabet ve Böbrek	YSA	x	√	√
8	Devarajan ve ark. [40]	Diyabet	J48Graft	x	√	√
9	Abdel-Basset ve ark. [41]	Diyabet	x	x	√	√
10	Khan ve ark. [42]	Covid-19	x	√	√	x
11	Otoom ve ark. [43]	Covid-19	DVM, NN, NB, k-NN, DeT, DS, OneR, ZeroR	x	√	√
12	Kumar ve Gandhi [44]	Kalp Hastalığı	LR	x	√	√
13	Kamarajugadda ve ark. [45]	Kalp Hastalığı	BBO-DVM	x	√	√
14	Khan ve Algarni [46]	Kalp Hastalığı	x	√	√	x
15	Önerilen IoMT Çerçevesi	Diyabet ve Kalp Hastalığı	KA, RO,GA, LR, DVM	√	√	√

1.3. TEZ ORGANİZASYONU

Bu tezde yeni bir IoMT çerçevesi önerilmiştir ve organizasyonu aşağıdaki gibidir. Birinci bölümde, amaç, kapsam ve tez çalışması kapsamında ilgili çalışmalara yer verilmiştir. İkinci bölümde, önerilen IoMT çerçevesinde kullanılacak olan KVAA, Apache Spark, Apache Kafka, bulanık mantık ve makine öğrenimi hakkında ilgili materyal ve yöntemlere yer verilmiştir. Üçüncü bölümde, tez çalışması kapsamında geliştirilen IoMT çerçevesi detaylı bir şekilde sunulmuştur. Dördüncü bölümde, önerilen IoMT çerçevesinin geliştirme aşamaları detaylı bir şekilde açıklanmaktadır. Beşinci bölümde, IoMT çerçevesi için geliştirilen diyabet ve kalp hastalığı senaryoları için ayrı ayrı performans sonuçları hakkında bulgular ayrıntılı bir şekilde sunulmuştur. Son olarak geliştirilen IoMT çerçevesinin sonuçları sunulmuş ve önemi ve alana katkısı vurgulanarak, gelecek çalışmalara ait yeni önerilere yer verilmiştir.

2. MATERYAL VE YÖNTEM

Bu bölümde, önerilen IoMT çerçevesinde kullanılan materyal ve yöntemler hakkında detaylı bilgiler farklı başlıklar altında sunulmuştur.

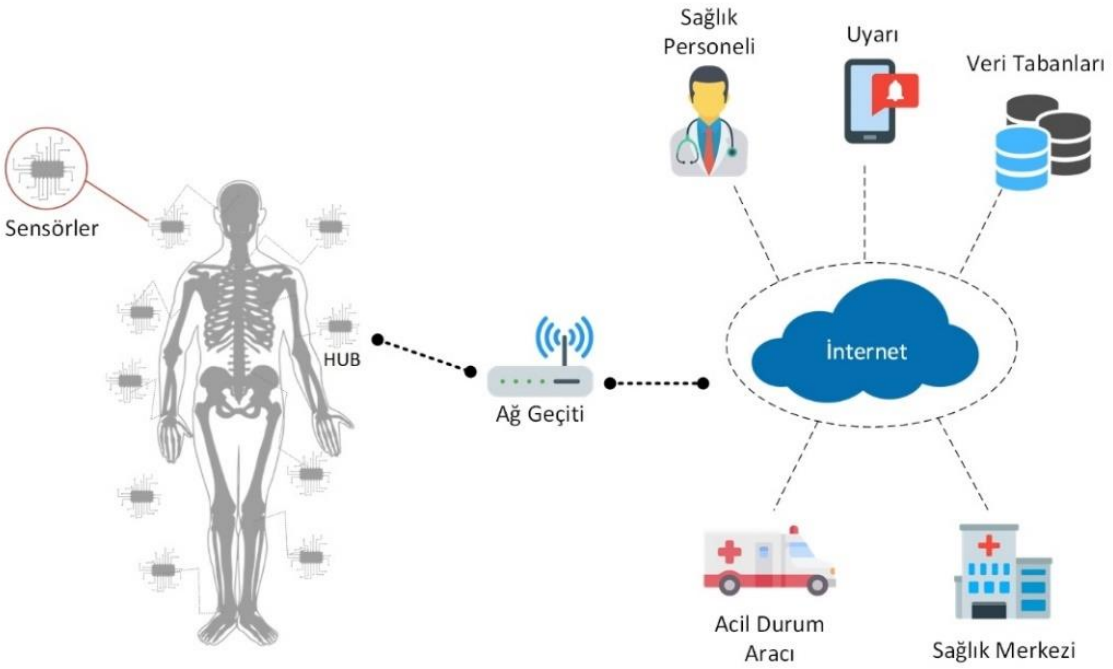
2.1. KABLOSUZ VÜCUT ALAN AĞLARI

Son yıllarda, düşük güç elektroniği, algılama cihazları, kablolu ve kablosuz teknolojilerindeki ilerleme, sağlık gibi büyük sektörlerdeki gibi gelişmiş hizmetlere olan talep arttıkça yükselişe geçmiştir. Örneğin kablosuz iletişim, birçok uygulamada kablolu iletişim yerine tercih edildiğinden son on yılda muazzam bir gelişme yaşadı. Uydu, televizyon, telefon, internet, kişisel aletler ve hatta elektrik kablosuz iletişimin uygulamalarıdır. Teknolojideki ve kablosuz iletişimdeki gelişmeler, sağlık sektörüne hitap eden ve profesyonel sağlık personeli tarafından yapılan çalışmaları etkin ve güvenilir bir şekilde tamamlayan çözümlerin geliştirilmesine yol açtı. Bunlarla sınırlı olmamakla birlikte içerisinde: doğru algılama cihazlarının gelişimini, uzaktan izlemeye izin veren kablosuz izleme cihazlarının tasarımı ve hastane ortamında tıbbi ağ yeteneklerinin geliştirilmesini barındırır. Kablosuz iletişim ve yazılım mühendisliğindeki ilerlemeyle birlikte insan vücudunun fizyolojik doğasını izleyebilen düşük güçlü, küçük tıbbi algılama cihazlarının geliştirilmesi, bu tür teknolojilerin her yerde bulunan sağlık sistemlerine entegrasyonuna yol açtı. Ucuz, güvenilir ve giyilebilir algılama cihazlarının getirilmesi, hastaları hastane yataklarında, hastane çevresinde ve hatta evlerinin rahatlığında sürekli izleyebilen küçük kablosuz ağların geliştirilmesine olanak sağlamaktadır. Bu sensörler, hastaların fizyolojik durumunu sürekli gözlemleyen KVAA adında kablosuz ağlar oluşturur.

KVAA sağlık, tıp ve sporda çok çeşitli tıbbi / tıbbi olmayan uygulamaları destekler [47], [48] İlgi çekici olan, hem tanı hem de korunma için fizyolojik sinyallerin sürekli izlenmesi için kullanıldı. Elektrokardiyogram (ECG), Elektroensefalogram (EEG), sıcaklık ve kan basıncı gibi vücuda entegre edilebilir sensörler ile hastaların yaşamsal bulgularının doğru teşhisi için vücut ölçümleri yapılır [49], [50]. Sensör ağının ve ilgili bileşenlerinin çalışması ve tasarımı tıbbi ve endüstriyel yönetmeliklere tabidir. Bu

düzenlemeler hastaya maksimum güvenlik, iletişimde güvenilirlik ve fizyolojik sinyallerin doğru ölçümlerini sağlar. Güç kaynakları, algılama cihazları, kablolu veya kablosuz alıcı vericilerin her biri bu tür düzenlemelere tabidir. Hastaların sağlığı bu tür ağların doğru şekilde çalışmasına bağlı olabileceğinden, sistem genelinde yüksek düzeyde tasarım ve uygulamanın doğruluğu korunmaktadır. Vücut alanı bağlantılarının modellenmesi ve ağ topolojisi ile eşleşen uygun bir protokol tasarımının seçilmesi de dahil olmak üzere, KVAA'ların tasarım ve uygulamasında, çeşitli faktörler göz önünde bulundurulmalıdır. Diğer taraftan, KVAA'lar kısa menzilli iletişim olarak bilindiğinden, devrelerin enerji tüketimi, aygıtların basit olmasını ve düşük güç tüketimi özelliklerine sahip olmasını gerektiren radyo frekans iletim enerjisine benzerdir. KVAA'ların tasarımında ve uygulanmasında, vücut alanı bağlantılarının modellenmesi ve ağ topolojisi ile eşleşen uygun bir protokol tasarımının seçilmesi de dahil olmak üzere çeşitli faktörler göz önünde bulundurulur.

Kablosuz izleme için sürekli olarak çözümler geliştirilir. Bu tür ağların güvenilirliğini artırmak ve ayrıca daha geniş bir tıbbi cihaz yelpazesini desteklemek için çeşitli kablosuz protokoller önerilmekte ve uygulanmaktadır [51]. Tıbbi düzenlemeler nedeniyle, KVAA'ların protokolleri maksimum iletim gücü, çalışma sıklığı ve diğer tıbbi cihazlarda ortaya çıkan minimum parazite uygun olmalıdır. Öte yandan, çoklu biyosensörlerin merkezi bir düğüme ilettiği KVAA'lardaki protokoller, insan vücudu ve merkezi düğüm arasındaki gerçekçi bir kanal modelinde yüksek güvenilirlik, gürültüye bağışıklık ve çoklu erişim yeteneklerini desteklemelidir. Bu nedenle, KVAA cihazları, istenen uzun süreli uygulamayı sağlamak için gürültüye ve parazite karşı oldukça dayanıklı iken, düşük güç tüketimi özelliklerine sahip tasarımlar için geliştirilmeye ihtiyaç duymaktadır.



Şekil 2.1. KVAA genel mimarisi.

Bir KVAA genel mimarisi Şekil 2.1.'de gösterilmiştir [52]. İnsan vücudunun üzerine veya içine yerleştirilen sensör düğümleri fiziksel veri toplar ve ön işleme gerçekleştirir. Veriler bir koordinatör (HUB) düğüm tarafından toplanır ve daha sonra internet üzerinden paylaşılmak üzere bir baz istasyonuna iletilir.

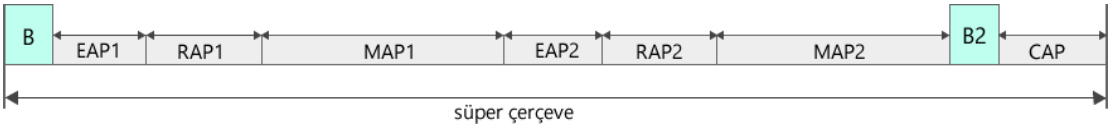
KVAA düğümleri heterojen bir ağ yapısına ve sınırlı enerji kaynaklarına sahiptir. İnsan vücuduna yerleştirilebilen bu sensör düğümleri çoğu durumda şarj edilemez ve değiştirilemez. IEEE 802.15.6 standardı, insan vücudunu çevreleyen küçük cihazlar arasındaki kısa mesafeli iletişimdeki hizmet farklılıklarını ele almak için geliştirilmiş ve kullanılmıştır.

2.1.1. IEEE 802.15.6 Standardı

IEEE 802.15.6 standardı, insan vücuduna yakın veya üzerine bağlı kısa mesafeli kablosuz iletişim için artan talebi karşılamak ve KVAA'ın çeşitli uygulamalarını hızlandırmak için 2008 yılında IEEE birliği tarafından geliştirilmiştir [53]. Standart, bir sekme ve iki sekme topolojisini sunar. Bir sekmeli topolojide veriler, düğümler ve HUB arasında doğrudan değiş tokuş edilirken, iki sekmeli topolojide ise HUB ve düğümler, veri alışverişi için röle düğümlerini kullanabilir. IEEE 802.15.6 standardı, üç farklı fiziksel katmanı destekleyen bir MAC katmanı tanımlar. Bunlar, dar bant, ultra geniş bant ve insan vücudu iletişimi katmanlarını içerir. MAC alt katmanında IEEE 802.15.6, çekişmeli erişim ve çekişmesiz

erişim dahil olmak üzere iki farklı erişim mekanizmasını destekler. Çekişmeli erişim aşaması, S-ALOHA tabanlı erişim mekanizmasını veya CSMA/CA tabanlı erişim mekanizmalarını destekler. Çekişmesiz erişim aşaması ise planlanmış bir yukarı bağlantı/aşağı bağlantı erişim şemasının yanı sıra doğaçlama bir yoklama/gönderme tabanlı erişim şemasını destekler [54].

Dünya çapında sunulan IEEE 802.15.6 standardı, ağların her birinin yalnızca bir HUB'a ve birden çok düğüme sahip olması beklenen KVAA'ları kümeler halinde düzenler. Bir HUB, işaret paketli süper çerçeve, işaret paketsiz süper çerçeve ve işaret paketsiz süper çerçevesiz yöntemlerle çalışabilmektedir. İşaret paketli süper çerçeve yöntemi, HUB ve tüm KVAA düğümleri arasında senkronizasyon sunar. Ayrıca, ağı daha güvenilir ve daha fazla enerji verimliliği sağlar. Düğümler uyku durumuna geçerek, gönderilecek verileri olduğunda periyodik olarak aktif hale gelir. Bu nedenle tıbbi uygulamalarda en çok kullanılan yöntemdir. Şekil 2.2.'de gösterildiği gibi, işaret paketli süper çerçeve yapısında iki acil erişim aşaması (EAP), iki rastgele erişim aşaması (RAP), iki yönetim erişim aşaması (MAP) ve bir çekişmeli erişim aşaması (CAP) olmak üzere yedi erişim aşaması bulunmaktadır. HUB, CAP başlamadan önce başka bir isteğe bağlı işaret ve her erişim aşamasının boyutunu tanımlamak için süper çerçevenin başlangıcında bir işaret gönderir. Erişim aşamaları EAP1, EAP2, RAP1, RAP2 ve CAP, CSMA/CA veya S-ALOHA erişim mekanizmasını kullanarak iletişimi başlatmak için düğümler tarafından çekişmeli tahsis için kullanılır. EAP, acil durum ve tıbbi olaylar için yalnızca en yüksek öncelikli veriler seviyesinde kullanılır. MAP erişim aşamasında, HUB programlanmış bağlantı tahsis aralıklarını düzenleyebilir, programlanmamış çift bağlantı tahsis aralıkları sağlayabilir ve tip I acil sorgulamalı tahsis aralıklarını sağlayabilir [55], [56].



Şekil 2.2. İşaret paketli süper çerçeve yapısı.

2.1.2. Geçici İsteğe Bağlı Mesafe Vektör Yönlendirme (AODV)

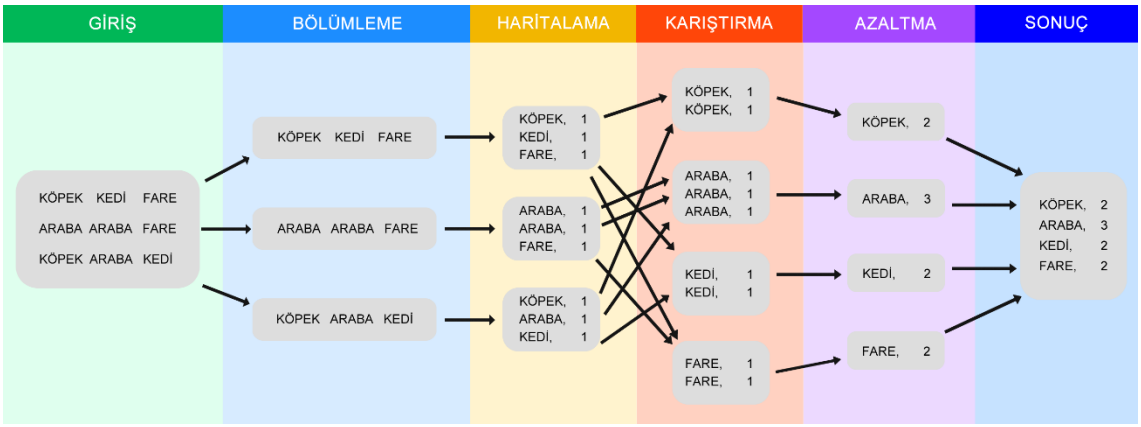
IEEE 802.15.6 standardı, KVAA içi kablosuz iletişim için fiziksel ve veri bağlantı katmanı gereksinimlerini ayrıntılı olarak tanımlamıştır, ancak her düğüm grubu arasında (veya HUB düğümleri arasında) KVAA arası kablosuz iletişim için herhangi bir öneride bulunmamıştır. Bu aşamada, HUB düğümleri arasında geçici olarak kablosuz iletişimin

sağlanması için her bir HUB düğümlerin paketlerini AODV yönlendirme protokolü yardımıyla hedefe gönderilir.

AODV yönlendirme protokolü, topoloji oluşturmak için düğümler arasında kendini başlatabilen dinamik ve çok sekmeli bir yönlendirme mekanizmasına sahiptir [57]. AODV, ağ topolojisi değişikliklerinde döngüsüz olarak Bellman-Ford'un sonsuza kadar sayma probleminden kaçınan bir yaklaşımdır [58]. Yönlendirme işlemi, düğümlere geçersiz yollar da bildirebilir. AODV'nin bir özelliği, her rota için hedef sıra numarasının kullanılmasıdır. Hedef sıra numarası, eklenecek hedefle birlikte talep eden düğümlere gönderdiği tüm rota bilgileri ile oluşturulur. Hedef sıra numaralarının kullanılması, döngü oluşturmamasını sağlar ve yönlendirme protokolünü tasarlamayı kolaylaştırır. Bir rota talep eden bir düğüm, iki rota arasında seçim yapmak için en büyük sıra numarasına sahip olanı seçer. Başka bir deyişle; bir kaynak düğümde bilinmeyen bir hedefe iletilecek veri olduğunda, o hedef için bir Rota Talebi (RREQ) yayınlar. Her ara düğümde, bir RREQ alındığında kaynağa bir yol oluşturulur. Alıcı düğüm bu RREQ'i daha önce almadıysa ve hedef değilse ve hedefe giden geçerli bir rotası yoksa, RREQ'yi yeniden yayınlıyor. Alıcı düğüm bir hedef veya hedefe giden geçerli bir yolsa, bir Rota Cevabı (RREP) oluşturur. RREP yayıldıkça, her ara düğüm hedefe giden bir yol oluşturur. Kaynak RREP'i aldığında hedefe giden yolu kaydeder ve veri göndermeye başlar. Kaynak düğüm tarafından birden fazla RREP alınırsa, en kısa atlama sayısına sahip rota seçilir.

2.2. APACHE SPARK

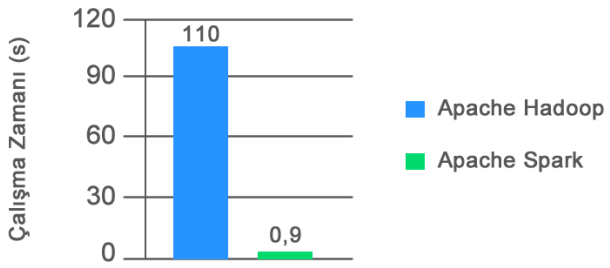
Apache Hadoop, Apache Software Foundation tarafından yönetilen açık kaynaklı bir yazılım platformudur. Büyük hacimli verilerin verimli ve uygun maliyetli depolanması ve yönetimi için en yaygın olarak bilinen platformdur. Apache Hadoop, Hadoop Dağıtılmış Dosya Sistemi (HDFS) ve MapReduce olmak üzere iki ana bileşenden oluşmaktadır. HDFS, verilerin esnek ve hataya dayanıklı bir şekilde depolanmasını sağlar. MapReduce, dosya sisteminden verileri okur ve girdi verilerini Map (Haritalama) işlevini kullanarak anahtar/değer çiftlerine dönüştürür ve ara sonuçlara aktarır. Reduce (Azaltma) işlevini kullanarak ara sonuçları azaltır ve sonuçları diske geri depolar. Şekil 2.3.'te örnek bir MapReduce işlemi gösterilmiştir.



Şekil 2.3. MapReduce işlemi.

Apache Hadoop, ölçeklenebilir, esnek ve hataya dayanıklı bilgi işlem çözümleri sağlayan programlama modeline (MapReduce) dayandığından, birçok sektörde veri analizleri için Apache Hadoop yoğun bir şekilde kullanılmaktadır. Apache Hadoop'un sınırlılığı, işlemlerin doğru ve hatasız bir şekilde yapılması için büyük boyutlu verilerin işlenmesi zaman almaktadır. Ayrıca, veriler üzerinde tekrarlı hesaplamaların yapılabilmesi için her işlemde HDFS'e gidip veri alınması gerektiğinden veri erişim süreleri artmaktadır.

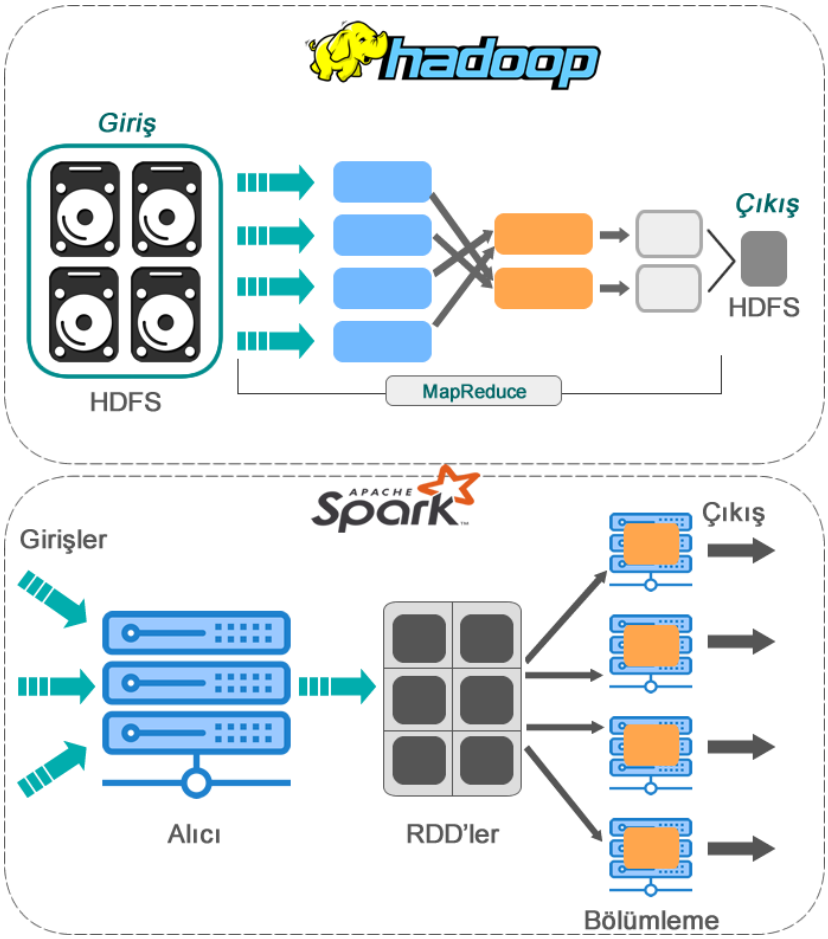
Apache Spark, büyük ve çeşitli verilerin toplu, gerçek zamanlı, etkileşimli ve yinelemeli bir şekilde işlenmesini birleştiren açık kaynaklı bir büyük veri işleme motorudur [59], [60]. Apache Spark, Apache Hadoop'taki MapReduce'un verimliliğini artırmak ve avantajlarını korumak için oluşturuldu. Apache Spark kendi dosya sistemine sahip olmasa da birçok farklı depolama sistemlerindeki verilere erişebilmektedir. Apache Spark'ın kullandığı veri yapısına Esnek Dağıtılmış Veri Kümesi veya RDD adı verilir. RDD, birkaç makineye bölünmüş, salt okunur bir nesneler topluluğudur. Bir bölüm yanlışlıkla çökse bile yeniden oluşturulabilmektedir. RDD'nin en büyük avantajı, fiziksel bellekte kalıcı olarak depolanma zorunluluğu yoktur. RDD'nin bu avantajı, özellikle yinelemeli uygulamalar için bazı belirli uygulama türlerinde daha iyi performans göstermektedir. Ayrıca, Spark, Apache Hadoop'a göre disk tabanlı uygulamalarda 10 kat, bellek içi uygulamalarda ise 100 kat daha iyi performans gösterdiği belirtilmektedir [60]. Bu bağlamda, Apache Hadoop ve Apache Spark'ın bir veri seti üzerindeki lojistik regresyon analizlerinin çalışma zamanı karşılaştırması Şekil 2.4.'te gösterilmiştir.



Şekil 2.4. Hadoop ve Apache Spark'ın lojistik regresyon karşılaştırması.

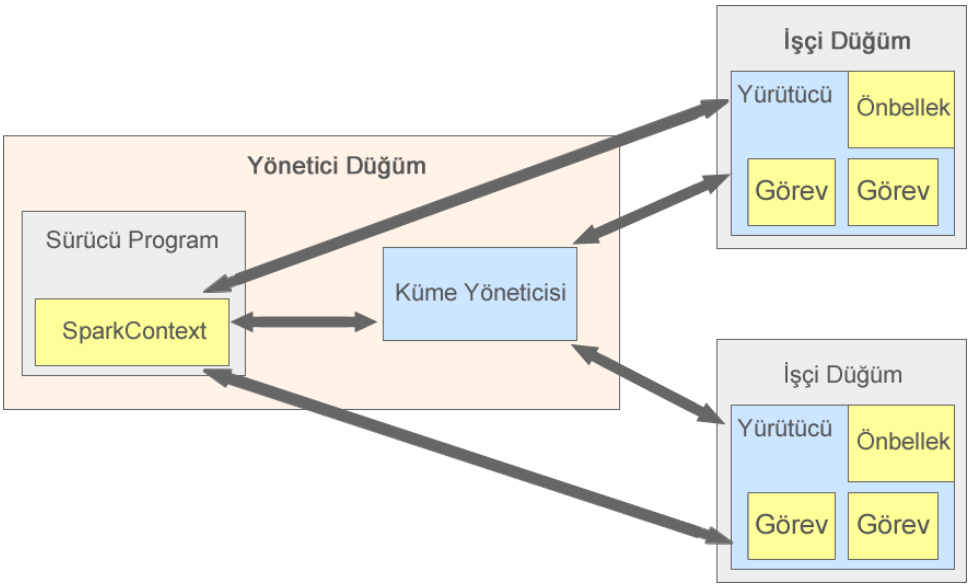
Apache Spark'ın en büyük avantajı, yinelemeli uygulamaların yürütülmesini hızlandıran bellek içi küme hesaplama işlemini sağlamasıdır. Giriş verileri, birçok işlemin aynı anda yapıldığı ana belleğe yüklenir. Veriler ana bellekte bulunduğundan, verilere erişim daha az zaman almaktadır. Bu nedenle disk'e G/Ç süresini tamamen ortadan kaldırmaktadır.

İki büyük veri işleme çerçevesi, verileri oldukça farklı şekillerde işlemektedir. Hem MapReduce özellikli Apache Hadoop hem de RDD'li Apache Spark verileri dağıtık bir ortamda işlese de Apache Hadoop toplu işleme için daha uygun olabilmektedir. Apache Spark ise gerçek zamanlı olarak gerçekleştirilen uygulamalarda öne çıkmaktadır. Apache Hadoop'un amacı, verileri disklerde depolamak ve ardından dağıtık bir ortamda toplu olarak paralel analiz etmektir. Apache Spark, RDD yapısı ile çalıştığından, RDD'leri en yakın düğümlere böler ve işlemleri paralel olarak gerçekleştirir. Sistem, Yönlendirilmiş Döngüsel Grafik (DAG) kullanılarak bir RDD üzerinde gerçekleştirilen tüm eylemleri izler. Bellek içi hesaplamalar ve üst düzey API'ler ile Apache Spark, yapılandırılmamış verilerin gerçek zamanlı olarak akışını etkin bir şekilde işleyebilmektedir. Apache Hadoop ve Apache Spark'ın veri işleme süreçleri Şekil 2.5.'te gösterilmiştir.



Şekil 2.5. Apache Hadoop ve Apache Spark'ın veri işleme süreçleri.

Apache Spark, Master (Yönetici) / Worker (İşçi) modeliyle dağıtık olarakta çalışmaktadır. Apache Spark programındaki ana işlem, işçi düğümlerini yöneten ve yönetici olarak da adlandırılan SparkContext'tir [60]. Apache Spark paketi, kümedeki tüm işçi düğümlerde arasında kurulmalıdır. Her Apache Spark programı, nesne yönelimli terminolojideki yöntemlere benzer dönüşümler ve eylemler içerir. Dönüşümler tembel bir şekilde değerlendirilir ve eylem çağrıldığında eylemler hemen hesaplanır. Program toplama dönüşümü içeriyorsa, SparkContext sürücüsü, kümedeki görevle ilişkili tüm işçilerden sonuçları toplar ve bunu Master'a üretir. SparkContext'in ana sorumluluğu, işçilere periyodik olarak görevler atayarak boşta kalmadıklarından emin olmaktır. Ölçeklenebilirlik elde etmek için işçiler, Apache Spark ve Scala yüklü olarak kümeye eklenerek dinamik olarak artırılabilir. Bir Apache Spark uygulamasının ana bileşenleri, sürücü, işçi ve küme yöneticisidir. Apache Spark'ın ayrıntılı mimarisi Şekil 2.6.'da gösterilmiştir.



Şekil 2.6. Apache Spark standalone yöntemi ile dağıtık hesaplama mimarisi.

Apache Spark'ın son derece popüler olmasının başlıca nedeni, onu veri işleme, makine öğrenimi ve veri akışı uygulamalarındaki kullanmanın çok kolay olmasıdır. Ayrıca, bellek içi hesaplama yaptığı için nispeten çok daha hızlıdır. Apache Spark genel bir veri işleme motoru olduğundan, Apache Kafka, MongoDB, HBase, Cassandra, Amazon S3, HDFS vb. gibi çeşitli veri kaynaklarıyla kolayca kullanılabilmektedir. Apache Spark, kullanıcılara üzerinde işlem yapabilmesi için Java, Python, Scala ve R programlama dili olmak üzere dört farklı dil seçeneği sunmaktadır.

Şekil 2.7. Apache Spark katmanları.

Apache Spark'ın katmanlarına bakıldığında (Şekil 2.7.) en altta Hadoop'un temel altyapını kullanır ve ardından Spark Core (çekirdek motoru) bulunmaktadır. Spark Core katmanın üstünde ise Spark SQL, Spark Streaming, MLlib ve GraphX bulundurur.

2.2.1. Spark Core

Spark Core, Şekil 2.7.'da gösterildiği gibi Apache Spark'ın en temel yapı taşıdır. Apache Spark'ın üstün işlevsellik özelliklerinin bel kemiğidir. Spark Core, verilerin paralel ve dağıtılmış olarak işlenmesini sağlayan bellek içi hesaplamalara olanak tanır. Apache Spark'ın tüm özellikleri, Spark Core üzerine inşa edilmiştir. Spark Core, görevleri, G/Ç işlemlerini, hata toleransını ve bellek yönetimini vb. yönetmekten sorumludur.

2.2.2. Spark SQL

Spark SQL, yapılandırılmış ve yarı yapılandırılmış veriler için yeni bir veri yapısı olan DataFrames'i sunmaktadır. DataFrames, bize Apache Spark programlarında SQL sorguları ekleme imkanı sunuyor. R ve Python'dan ödünç alınan dataframe yaklaşımını kullanır. Hem analist hem de geliştiricilere SQL benzeri sorgulama arabirimi sağlar.

2.2.3. Spark GraphX

Apache Spark, grafikler için bir API olan GraphX adlı grafik-paralel hesaplama dahil olmak üzere zengin bir üst düzey aracı ile birlikte gelir. Grafik verilerinin kullanımını kolaylaştırmak için Apache Spark API'sine bir soyutlama ekler ve bir dizi tipik grafik operatörü sağlar. Tek bir sistemde veri-paralel ve grafik-paralel hesaplama arasındaki boşluğu kapatmayı amaçlar; böylece veriler, veri hareketi veya tekrarı olmaksızın hem grafik hem de öğe koleksiyonları olarak sorunsuz bir şekilde görüntülenebilir.

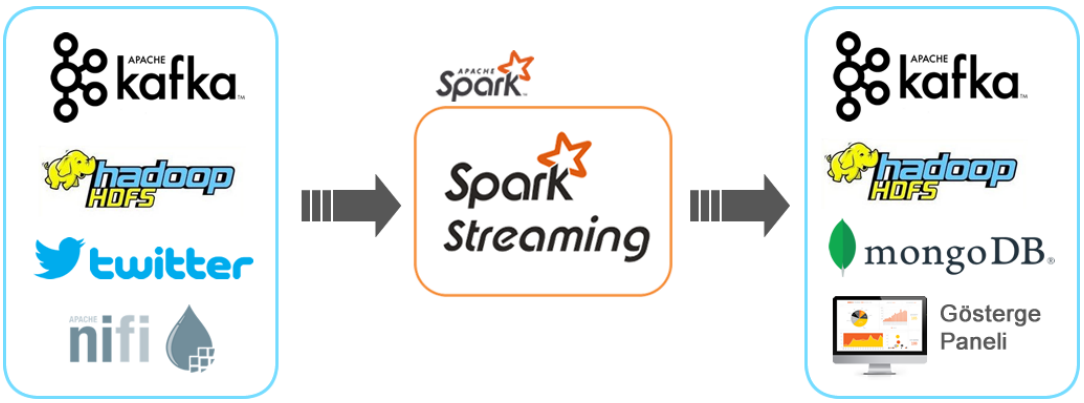
2.2.4. Spark MLlib

MLlib, makine öğrenimi tekniklerinin uygulanmasını sağlayan Apache Spark'ın makine öğrenimi kütüphanesidir. Sınıflandırma, regresyon ve kümeleme modellerini uygulayabilen ve performansa göre ölçeklenebilir bir makine öğrenimi kitaplığıdır. Apache Spark, Scala programlama dili üzerine kurulmuştur, ancak Java, Python tabanlı uygulamalar Hadoop veya bağımsız ortamlarda da çalıştırılabilir [61].

2.2.5. Spark Streaming

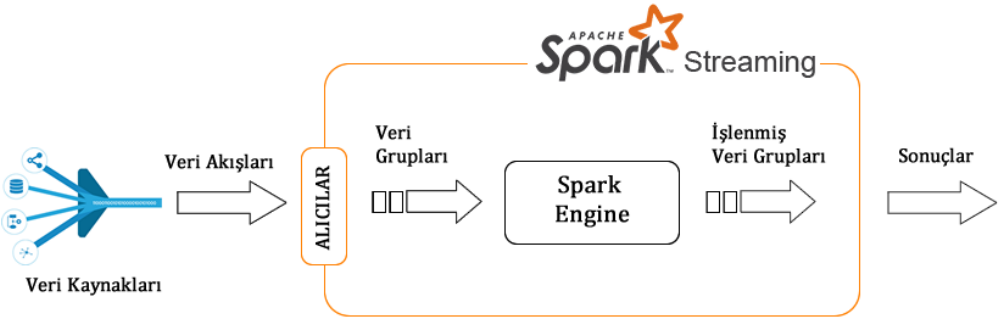
Spark streaming, gerçek zamanlı verilerin yüksek verimli, ölçeklenebilir ve hataya dayanıklı bir şekilde işlemesine olanak tanır. Apache Spark'ın bir uzantısı olarak düşünülebilir. Spark Streaming, Şekil 2.8.'de gösterildiği gibi gerçek zamanlı olarak aldığı verileri hızlı bir şekilde işleyerek diğer sistemlere aktarması için tasarlanmıştır. Spark Streaming kullanılarak Apache Kafka, NiFi, Kinesis, Twitter, Flume, HDFS veya

TCP soketleri gibi harici kaynaklardan gerçek zamanlı olarak veri alımı gerçekleştirilebilir. Harici kaynaklardan alınan veriler map, reduce, join ve window gibi üst düzey karmaşık veri hesaplama algoritmaları kullanılarak Apache Spark içinde işlenebilir. Ayrıca, gerçek zamanlı olarak alınan verileri anlamlandırmak için çeşitli makine öğrenimi ve grafik işleme algoritmaları uygulanabilir. Son olarak, işlenen büyük veriler MongoDB, Cassandra, HDFS, S3 gibi veri tabanlarına, Apache Kafka veri akışı platformuna ve gerçek zamanlı gösterge panellerine gönderilir.



Şekil 2.8. Farklı kaynaklar ile Spark Streaming entegrasyonu.

Apache Spark, alıcıları (receiver) ile gerçek zamanlı verileri alır ve her bir satır kaydı tek tek işlemek yerine birkaç küçük toplu (yığın) işlere böler. Bu küçük yığınlardan elde edilen akışa Ayrıklaştırılmış Akış (DStream) adı verilir. Özet olarak, Apache Spark verileri paralel olarak alır ve ardından işçi düğümlerinin belleğinde ara belleğe alır. Arabellek boyutunun, çalışan düğümde kullanılabilir bellekten büyük olamayacağına dikkat edilir. Bu tür her veri grubu, Apache Spark tarafından RDD olarak işlenir. Daha sonra Apache Spark motoru, küçük yığınları işlemek ve sonuçları belirtilen diğer sistemlere iletmek için kısa görevler yürütür. Spark Streaming gerçek zamanlı olarak veri işleme süreci Şekil 2.9.'da gösterilmiştir.

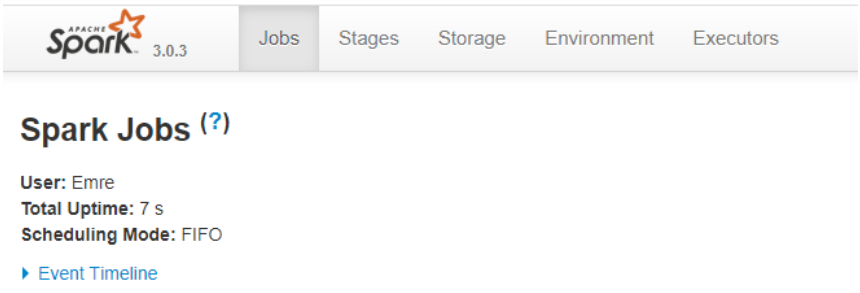


Şekil 2.9. Spark Streaming veri işleme süreci.

2.2.6. Spark UI

Spark UI, Apache Spark uygulamalarının çalışma performansının izlenmesine ve hatalarının ayıklanmasına yardımcı olmak için tasarlanmış bir web uygulamasıdır. Bir Spark uygulamasının ayrıntılı çalışma zamanı bilgilerini ve çeşitli kaynak tüketimlerini içerir. Çalışma zamanı, Apache Spark uygulamalarınızdaki performansını tanımlamada son derece yardımcı olan çeşitli ölçümler içermektedir. Unutulmaması gereken bir nokta, Spark UI yalnızca bir Apache Spark uygulaması çalışırken kullanılabilmektedir [62].

Apache Spark web kullanıcı arayüzüne http://<spark-ip>:4040 adresinden ulaşılabilmektedir. Bu URL'e yönlendirilirse, tarayıcınız Şekil 2.10.'daki gibi görüntülenecektir.



Şekil 2.10. Apache Spark web arayüzü.

2.3. APACHE KAFKA

Apache Kafka, başlangıçta LinkedIn'de geliştirilen ve ardından 2011 yılında açık kaynaklı bir Apache projesi haline gelen hızlı, ölçeklenebilir, hata toleranslı ve dağıtık bir mesajlaşma sistemidir. Apache Kafka, geleneksel mesajlaşma sistemleri ile karşılaştırıldığında yüksek verimlilik, yerleşik bölümleme ve dağıtık hesaplama özelliği ile büyük ölçekli bir mesaj işleme sistemi olarak öne çıkmaktadır [63], [64].

Dağıtık mesajlaşma sistemlerinde mesajlar, istemci uygulamaları ve mesajlaşma sistemleri arasında eş zamansız olarak sıraya alınır. Üreticiler tarafından var olan konulara gönderilen mesajlar, tüketicilerin bir veya birden fazla konuya abone olarak tüketebilecekleri mesajlardır. Apache Kafka, çevrimiçi ve çevrimdışı mesaj tüketimi için uygun bir sistemdir. Apache Kafka, Apache Zookeeper servisi tarafından koordinasyonu sağlanmaktadır. Apache Spark gibi veri işleme motorları ile entegre bir şekilde çalışabilmektedir. Düşük gecikmeli mesaj iletim özelliği ile çok sayıdaki farklı tüketiciyi kontrol edebilmekte ve saniyede 2 milyona kadar veriyi yazabilmektedir [64].

2.3.1. Apache Kafka Bileşenleri

Apache Kafka kümesi birkaç bileşenden oluşur. Apache Kafka mimarisinin farklı terminolojileri aşağıda açıklanmıştır.

Topic (Konular): Bölümlere ayrılmış belirli bir kategoriye ait mesaj akışlarıdır. Bu bölümler, Apache Kafka'ya gelen tüm veriden rastgele miktardaki veriyi işleyebilen bir dizi eşit boyutlu bölüm dosyalarıdır. Herhangi bir veri kaybını önlemek adına bu bölümlerin kopyaları tutulmaktadır [64].

Broker (Komisyoncu): Yayınlanan veriler, konu başına sıfır veya daha fazla bölüme sahip olabilen Apache Kafka'daki komisyoncular aracılığıyla korunur. Birden fazla komisyoncu, mesaj verilerinin kalıcılığını ve çoğaltılmasını yönetmek için kullanılan Apache Kafka kümesini oluşturur [64].

Producers (Üreticiler): Mesajı son segment dosyasına veya bölüme ekleyen Apache Kafka sunucularına mesaj yayınlarlar. Ayrıca, bu üreticiler mesaj göndermek için bölümleri seçebilirler [64].

Consumers (Tüketiciler): Tüketiciler bir veya daha fazla konuya abone olurlar ve komisyonculardan veri çekerek yayınlanan mesajları okurlar.

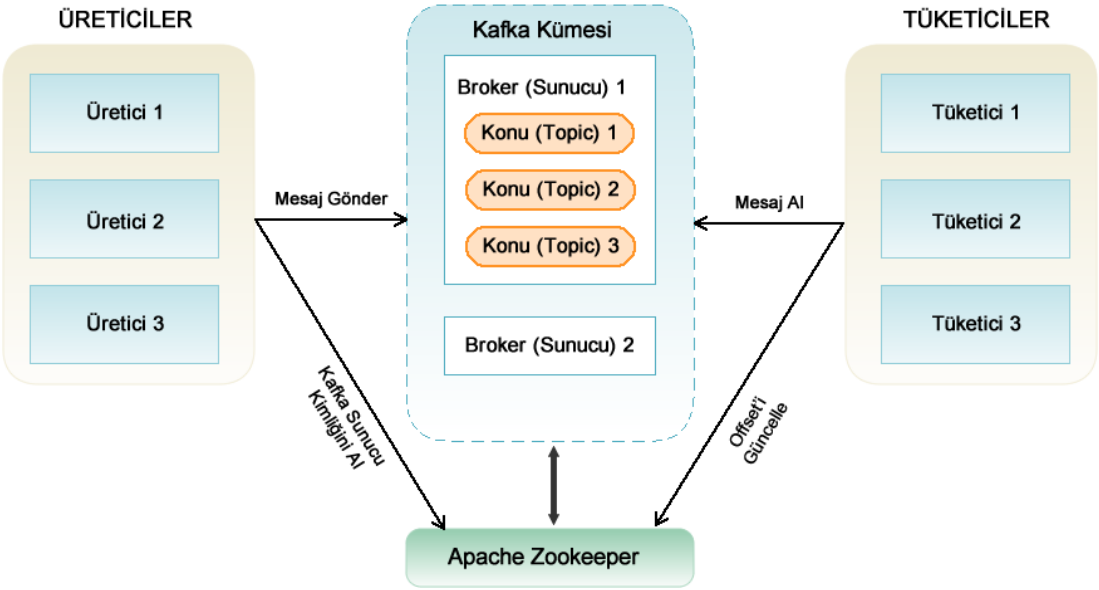
Lider ve Takipçi (Leader and Follower): Verilen bölüm için tüm okuma ve yazma işlemlerinden sorumlu olan düğüm, lider sunucu olarak bilinir ve takipçi düğümü, liderden gelen talimatları takip eder. Takipçi normal tüketici gibi davranır ve liderin başarısız olması durumunda takipçilerden biri yeni lider olur [64].

2.3.2. Apache Zookeeper

Apache ZooKeeper, çok sayıda Apache Kafka kümesini yönetmek için dağıtık bir koordinasyon hizmeti sağlar. Dağıtık bir ortamdaki bir hizmeti koordine etmek ve yönetmek karmaşık bir süreçtir. Apache ZooKeeper, basit mimarisi ve API'si ile bu sorunu çözmektedir. Apache ZooKeeper, yazılımcıların dağıtık ortamdaki uygulamalarının yönetiminden endişe duymalarına izin vermeden, onların temel uygulama mantığına odaklanmalarına olanak sağlamaktadır. Apache ZooKeeper, müşterilerine yüksek aktarım hızı, düşük gecikme süresi ve yüksek kullanılabilirlik sağlamaktadır [65].

2.3.3. Apache Kafka Çalışma Prensibi

Apache Kafka'nın Şekil 2.11.'de gösterildiği gibi çalışma prensibinde bir üretici (producer), Apache Kafka konusuna (topic) mesajlar gönderir. Apache Kafka konuları, Apache Kafka sunucusu olarak adlandırılan broker'larda yaratılır. Broker'ların oluşturduğu gruba ise Apache Kafka kümesi (kafka cluster) denir. Apache Kafka sunucuları gerektiğinde mesajları saklayabilme özelliğine sahiptir. Tüketiciler (consumer) daha sonra, sunuculardan mesaj çekerek mesajlarını almak için Apache Kafka konusuna (bir veya daha fazla) abone olurlar [63].



Şekil 2.11. Apache Kafka çalışma prensibi.

2.4. BULANIK MANTIK

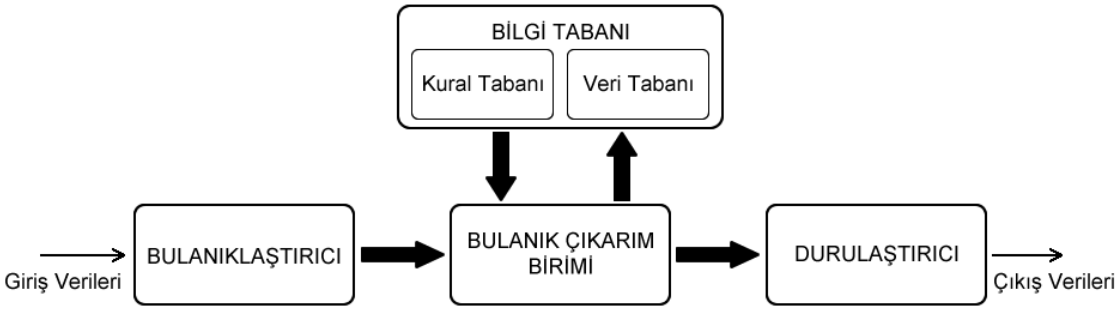
Tıbbi tedavide karar verme, doktorlar ve diğer sağlık çalışanları için devam eden bir zorluktur. Karar vermeye yardımcı olan büyük miktarda veri olmasına rağmen, yine de uzman görüşü karar vermede nihai rol oynamaktadır. Uzman görüşünün doktorlardan doktorlara değişiklik gösterdiği, karar vermedeki değişkenliğin nedeninin verilerdeki muğlaklık ve belirsizlik olduğu görülmüştür. Şüpheli ve belirsiz olan uzman bilgisi, bulanık küme tarafından yorumlanabilmektedir. Bulanık küme kavramı ilk olarak Lotfi Zadeh tarafından 1965 yılında tanıtıldı [66]. Zadeh tarafından kesin olmayan sınıflar tanımlanmış bulanık kümeler, örüntü tanıma, bilgi iletişimi ve soyutlama alanlarında önemli bir rol oynamaktadır. Zadeh, en başından beri, bulanık kümelerin doğası gereği istatistiksel olmadığını ve belirsizliğin rastgele değişkenlerle değil üyelik

fonksiyonlarıyla çözülmesi gerektiğini açıkça belirtti. Yeni ufuklar açan kavram, ilk aşamasında kötü karşılanmıştı, ancak matematikten mühendislik uygulamalarına, özellikle modelleme, karar verme optimizasyonları ve veri işleme gibi birçok bilimsel araştırma alanında olağanüstü bir yöntem haline geldi. Bununla birlikte, kavramın etkisi görüntü işleme, örüntü tanıma, yapay zeka ve bilgi sistemlerinde de fark edildi. Ayrıca, bulanık kümede olasılık teorisinin tanıtılması, belirsizlikle başa çıkmayı başardı [67].

Gerçek hayatta, bir sorunun yanıtlarında tartışmalı olan birçok durum olacaktır. Bu durum net bir yanıtın olmadığı anlamına gelmektedir. Şüpheli cevabın arkasındaki mantık, şüphe, belirsizlik ve kişisel görüş vardır. Örneğin, "bugün hava sıcak mı?" sorusunun farklı bir cevapları olabilir. Gerçek senaryo incelendiğinde, hava güneşli ve sıcaklık 26 °C'dir. Ancak, cevap kimileri için "evet sıcak", kimileri için "hayır değil", kimileri içinse "biraz sıcak" olabilmektedir. Ancak sorunun daha mantıklı bir şekilde cevaplanması gerekmektedir.

Bulanık mantığa temelde iki nedenle ihtiyaç vardır. İlki, gerçek dünya kesin şeyleri tanımlamak için açık değildir. Diğeri ise gerçek dünyadaki sistem hakkında bilgi, insan bilgisinden, sensör verilerinden ve sistemin matematiksel modelinden elde edilebilmektedir. Bu yüzden olaylar bulanık mantıkla açıklanmalı ve insan bilgisi verilerle birleştirilmelidir.

Bulanık mantık sistemi, bir dizi dilsel kuralla tanımlanan bilgisayar tabanlı bir algoritma sistemidir. Algoritma, uzman bilgilerini IF-THEN biçiminde karakterize eder. Belirsiz girdi ifadeleri ile çıktı kararlarını ilişkilendiren kuralları oluşturur. Kuralın IF kısmı öncül olarak adlandırılır ve kesin olmayan sistem durumlarını tanımlarken, kuralın THEN kısmı sonuç olarak adlandırılır ve duruma göre alınabilecek eylemleri belirtir. Bulanık bir sistem, Şekil 2.12.'de gösterildiği gibi dört bileşenden oluşur: bulanıklaştırma, bilgi tabanı, bulanık çıkarım birimi, durulaştırma.



Şekil 2.12. Bulanık mantık sistem yapısı.

2.4.1. Bulanıklaştırma

Bulanıklaştırma işlemi, sistemin ilk adımıdır. Kesin bir girdi değerinden bulanık bir girdi değerine dönüşüm, bulanıklaştırma işleminde karşılık gelen üyelik derecelerine sahip üyelik fonksiyonları seçilerek işlenir. Sistem sürecinin başlangıcında dönüşüm tamamlanır ve girdi değerleri dilsel değişkenlere çevrilir. Bulanıklaştırma işleminin temel amacı, 0 ile 1 arasında değerlere sahip girişlerin bazı giriş üyelik fonksiyonları uygulanarak kullanılmasıdır [68].

2.4.2. Bilgi Tabanı

Bilgi tabanı, bulanık mantık sisteminin bir başka adımıdır. Kural tabanı ve veri tabanı, bilgi tabanının iki bileşenidir. Kural tabanı, bulanık koşullu ifadelerin seçimini içerir ve bir veri tabanı, bulanık koşullu ifadelerdeki üyelik fonksiyonlarını tanımlar. Bilgi tabanı, bulanık bir akıl yürütme ve tümdengelim mekanizması nedeniyle bu sistemin özüdür. Ayrıca, insan beyni özel bir bilgi tabanlı sisteme sahiptir. Bilgi kaynağı, matematiksel yöntemler ve alan uzman bilgilerinden oluşmaktadır [69].

2.4.3. Bulanık Çıkarım Birimi

Bulanık çıkarım birimi, bulanık mantık sisteminin temelidir. Bulanık çıktıyı oluşturmak için çıkarım koşullarını bulanık girdiye uygulamaktan sorumludur. Özellikle, çıkarım koşulları, dilsel değerleri değerlendirmek ve bunları kesin bir değere dönüştürmek için durulaştırma sürecini gerektiren bir bulanık kümeye eşlemek için kullanılır. Bulanık sistemin hesaplama işlevselliğini sağlayan çıkarım koşullarıdır. Bu koşullar daha önceki deneyimlere, gözlemlere ve uzmanların bilgisine dayanabilir. Her bulanık çıkarım kuralı, If-Then ifadelerini ve dilsel değişkenleri içeren iki kavramdan oluşur. If-Then kuralları, gelen dilsel girdinin öncül olduğu ve sonucun girdiye dayalı dilsel çıktının olduğu öncülleri ve sonucu içerir [70].

2.4.4. Durulaştırma

Durulaştırma, bulanık çıkarım biriminden elde edilen bulanık bir kümeyi kesin bir sayı ile temsil etme işlemidir. Durulaştırma için birçok yöntem vardır, ancak bir bulanık kümeyi netleştirmek için en yüksek nokta, en yüksek nokta ortalaması, alan merkezi, maksimum ortalama, ağırlık merkezi, ağırlıklı ortalama vb. olmak üzere yaygın olarak kullanılan birkaç yöntem vardır.

Bulanık mantık sistemi, sağlık kuruluşlarında daha doğru sonuçlar elde etmek için etkili bir yöntem olduğu kanıtlanan karar destek sistemlerinden biridir. Sağlık hizmetlerinde bulanık mantığın amacı, insan karar verme sürecine yaklaşan esnek bir bilgi işlem sistemi oluşturmayı sağlamaktır. Kesin olmayan bilgilerden kesin bir çözüm belirlemenin basit bir yolunu bularak insan karar verme sürecini en aza indirir.

2.5. MAKİNE ÖĞRENİMİ

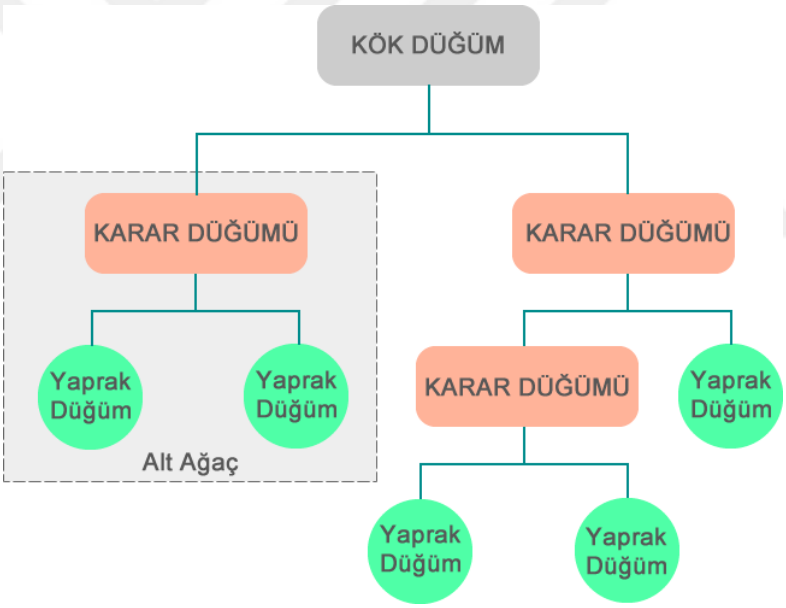
Makine öğrenimi, öncelikle büyük veri kümelerinden tahmine dayalı modeller oluşturmayı amaçlayan veri analiz teknikleridir. Karmaşık biyolojik sistemlerin daha iyi anlamlandırılması ihtiyacının artması nedeniyle makine öğrenimi, modern biyolojik araştırmaların ayrılmaz bir parçası haline gelmektedir.

Makine öğrenimi, veriye dayalı tahminler için veri analizi modeli uyarlamayı otomatikleştirmek için kullanılabilecek istatistiksel bir çerçeve olarak da tanımlanabilir. Makine öğreniminin arkasındaki ana fikir, bir algoritma ile ilişkili öz nitelikleri kullanmak ve onu bir sınıfa atamaktır. Öz nitelikler, bireylerin ölçülebilir özelliklerinin gözlemlenmesi olgusunu temsil ettiği söylenebilir. Bir makine öğrenimi modeli oluşturmak genellikle büyük ustalık ve hesaplama yeteneği gerektirir. Bu araştırmada, bazı popüler makine öğrenmesi algoritmaları kullanılarak veriye dayalı bir hesaplama yaklaşımı kullanılacaktır. Bu arada, makine öğrenimi algoritmaları, veriye dayalı bir modellemeyi gerçekleştirmek için belirli bir süreç gereklidir. Önceden, çoğu programcı algoritmaları kullanarak talimatları bilgisayarlara adım adım iletmek zorundaydı. Ancak, makine öğreniminin başlamasıyla birlikte, algoritmalar ilgili verilerden bilgi çıkararak öğrenmeyi gerçekleştirirler. 1950'lerde Alan Turing, bir bilgisayarın insanlar onunla etkileşime girdiğinde öğrendiğini iddia eden, yapay zekaya odaklanan ve kendi adını taşıyan bir kavram önerisinde bulundu. Ayrıca, insan-bilgisayar ve insan-insan etkileşimleri arasında hiçbir fark olmadığını belirtmek için çalışmalarını daha da ileriye götürdü. Makine öğrenimi dört farklı kategoride sınıflandırılabilir: denetimli, yarı denetimli, denetimsiz ve pekiştirmeli öğrenme algoritmaları.

Bu tezde, Karar Ağacı (KA), Rastgele Orman (RO), Gradyan Artırma (GA), Lojistik Regresyon (LR) ve Destek Vektör Makineleri (DVM) olmak üzere diyabet ve kalp hastalığı semptomlarını tahmin etmek için beş denetimli sınıflandırma algoritması kullanılmıştır.

2.5.1. Karar Ağaçları

Karar Ağacı algoritması, sınıflandırma ve regresyon problemlerinde kullanılan denetimli öğrenme yöntemidir. Karar ağacı sınıflandırıcısı, bir karar ağacı oluşturarak sınıflandırma modelini oluşturur. Ağaçtaki her düğüm, bir öznitelik üzerinde bir test belirtir, o düğümden inen her dal, o öznitelik için olası değerlerden birine karşılık gelir. Her yaprak, örnekle ilişkili sınıf etiketlerini temsil eder. Eğitim setindeki örnekler, yol boyunca yapılan testlerin sonucuna göre ağacın kökünden bir yaprağa kadar gezinerek sınıflandırılır. Ağacın kök düğümünden başlayarak, her düğüm, bir öznitelik test koşuluna göre örnek uzayını iki veya daha fazla alt uzaya böler. Ardından, özniteliğin değerine karşılık gelen ağaç dalından aşağı doğru hareket ettirilerek yeni bir düğüm oluşturulur. Bu işlem daha sonra yeni düğümde köklenen alt ağaç için eğitim kümesindeki tüm kayıtlar sınıflandırılana kadar tekrarlanır [71]. Karar ağacının genel yapısını gösteren diyagram Şekil 2.13.'te verilmektedir.



Şekil 2.13. Karar ağacının genel yapısı.

Karar ağacı oluşturmadaki en önemli problem, kök düğüm ve alt düğümler için en iyi özniteliğin seçilmesi işlemidir. Bu tür problemleri çözmek için Bilgi Kazanımı ve Gini Endeksi olmak üzere iki tür teknik vardır. Bu teknikler kullanılarak ağacın düğümleri için en iyi nitelik seçilmiş olur [71].

Bilgi kazanımı, bir veri setini bir öznitelik üzerinde böldükten sonra tüm entropi'den çıkarmaya dayanır. Bir karar ağacı algoritması her zaman bilgi kazanımını en yüksek değere ulaştırmaya çalışır. Bu nedenle, en yüksek bilgi kazancına sahip olan bir öznitelik

önce bölünür. Bilgi kazanımı ID3, C4.5 ve C5.0 karar ağaçları algoritmalarında kullanılır. Bilgi kazanımı aşağıdaki formül kullanılarak hesaplanabilir:

$$Bilgi\ Kazanımı(T,S) = Entropi(T) - Entropi(T,S) \tag{2.1}$$

Entropi, öznitelikteki belirsizliği ölçmek için kullanılan bir metriktir. Entropi aşağıdaki formülle hesaplanabilir:

$$Entropi(S) = \sum_{i=1}^{C} -p_i * log_2\ (p_i) \tag{2.2}$$

Gini endeksi ise CART (Sınıflandırma ve Regresyon Ağacı) algoritmasında bir karar ağacı oluştururken kullanılan belirsizlik ölçüsüdür. Aşağıdaki formül kullanılarak hesaplanabilir:

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2 \tag{2.3}$$

2.5.2. Rastgele Orman Algoritması

Rastgele orman, esnek, sağlam, karar ağaçları topluluklarına dayanan doğrusal olmayan sınıflandırıcılardır. Bir RO'da, eğitim verilerinin rastgele bir alt kümesinde çalışan birçok karar ağacı oluşturulur. Belirli bir ağacın her bir düğümünde, rastgele seçilen bir özellik verileri böler. Bir test senaryosunun tahmin sonucu, ağaçların bu duruma ilişkin tahminlerinin ortalamasıdır. RO'lar çok sayıda özelliğe sahip seyrek veri kümelerini işlemede ustadır, ancak bilgilendirici olmayan özelliklerin kaldırılması model doğruluğunu artırabilir.

Ağaçların rastgele ormanı oluşturma prosedürleri aşağıda açıklanmaktadır.

Örneğin $D = \{(x_1, y_1), (x_2, y_2) \dots (x_n, y_n)\}$ veri seti olduğu varsayılsın. Amaç, X'in girdiler ve Y'nin üretilen çıktılar olduğu $f = X \rightarrow Y$ fonksiyonunu bulmaktadır. Bununla birlikte M toplam girdi sayısı olsun.

- Rastgele orman, bir önyükleme örneği oluşturmak için D'den değiştirilmiş n gözlemi rastgele seçer.
- Her ağaç, genel M özelliklerinden bir özellik alt kümesi kullanılarak büyütülür. Regresyon için, özniteliklerin alt kümesinin M = 3 olarak ayarlanması önerilir.

Ardından her düğümde, m öznitelikler rastgele seçilir ve öznitelikler arasında en iyi performans gösteren bölüm, Gini endeksine göre seçilir.

- Ağaçlar budanmadan maksimum derinliğe kadar büyütülür.

2.5.3. Gradyan Artırma Algoritması

GA modeli, birçok temel regresyon veya sınıflandırma algoritmaları için uygulanabilecek genel bir yaklaşımdır [72]. Artırma yöntemlerinin gradyan bazlı formülasyonu ve karşılık gelen modeller GA makineleri olarak adlandırılır [73]. AdaBoost gibi, GA makineleri yanlış sınıflandırılmış tahminleri yeniden ağırlıklandırarak temel öğrenme modellerini yinelemeli olarak oluşturur. Ancak GA, AdaBoost'tan farklıdır, çünkü GA ağırlıkları her eğitim tahmininde kayıp fonksiyonunun negatif kısmi türevleri üzerinde çalışarak belirler. Bu kısmi türevler aynı zamanda sözde kalıntılar olarak da adlandırılır ve bu sözde kalıntılar kullanılarak yinelemeli olarak bir topluluk büyütülür. GA nispeten küçük veri kümeleri için verimli olabilirken, çok daha büyük veri kümeleri için ölçeklenebilir sürümlere ihtiyaç vardır. XGBoost, LightGBM ve CatBoost, GA'nın bu gereksinimi karşılamak için yakın zamanda geliştirilmiş ağaç tabanlı ölçeklenebilir sürümleridir. Temel makine öğrenimi modeli olarak karar ağaçlarını kullanan GA, ölçeklenebilir sürümlerden, gradyan destekli karar sınıflandırıcıları olarak etiketleyerek ayırt edilir [74].

GA, çok çeşitli pratik uygulamalarda kayda değer başarı gösteren güçlü makine öğrenimi teknikleri ailesidir. Farklı kayıp işlevlerine göre öğrenilmeleri gibi, uygulamanın özel ihtiyaçlarına göre son derece özelleştirilebilirler [73].

2.5.4. Lojistik Regresyon

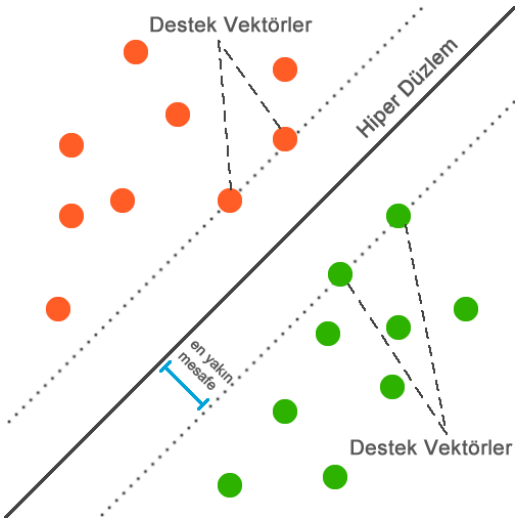
LR, her ne kadar isminde regresyon kavramı geçse de çok fazla kullanılan bir sınıflandırma modelidir. LR, genellikle iki sınıflı sınıflandırma problemlerinde kullanılmaktadır. LR, kategorik tahmin için standart çok değişkenli doğrusal regresyonda yapılan bir değişikliktir. Çıktının 0 ile 1 arasında sınırlandırılmasını ve bir olasılık olarak ele alınabilmesini sağlamak için bir lojistik fonksiyon kullanır [75]. LR, bağımlı değişkeni bir ikili değişkene dönüştürdükten sonra maksimum olabilirlik tahminini uygular. Bu şekilde, lojistik regresyon, belirli bir olayın meydana gelme olasılığını tahmin eder.

LR, sürekli ve/veya kategorik bağımsızlar bazında bir bağımlı değişkeni tahmin etmek ve bağımsızlar tarafından açıklanan bağımlı değişkendeki varyans yüzdesini belirlemek için

kullanılabilir. Bağımsızların göreli önemini sıralamak için kullanılabilir. Etkileşim etkilerini değerlendirmek ve ortak değişken kontrol değişkenlerinin etkisini anlamak için de kullanılabilir.

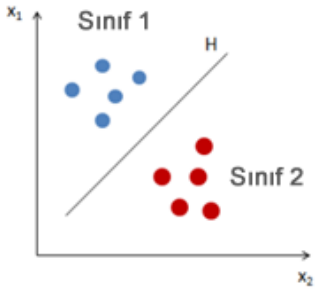
2.5.5. Destek Vektör Makinesi

DVM, verileri sınıflandırmak için kullanılan denetimli bir makine öğrenmesi algoritmasıdır. DVM, Şekil 2.14.'te gösterildiği gibi iki farklı sınıflara ait verileri ayıran ve optimal bir hiper düzlem bulan bir sınıflandırma modelidir. DVM'nin amacı, sonunda daha iyi bir sınıflandırma performansına sahip olmak için hiper düzlemden her bir düzleme en yakın noktaya olan mesafeyi temsil eden marjı optimize etmektir.



Şekil 2.14. Destek vektör makinesi sınıflandırma modeli.

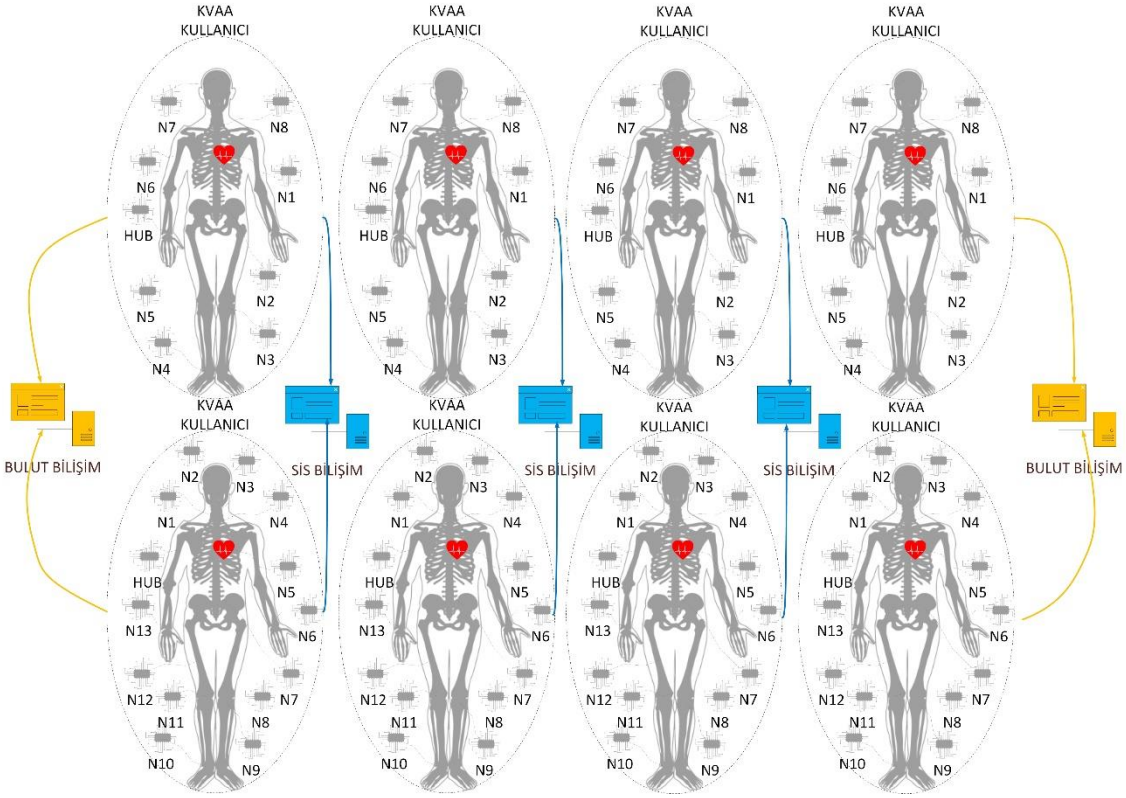
DVM basitçe şu şekilde tanımlanabilir: giriş özellikleri uzayında çizilen farklı veri noktalarından oluşan sınıflara sahip olan logaritma, Şekil 2.15.'te gösterildiği gibi noktaları farklı sınıflara ayıracak bir düzlem oluşturmalıdır. Optimal hiper düzlem bulunduktan sonra, DVM algoritmasının verdiği çözüm, onun kenarında bulunan "destek vektörleri" olarak adlandırılan noktalar ile tanımlanabilir.



Şekil 2.15. Farklı sınıflara ait olan veri noktalarının bir H düzlemi ile ayrılması.

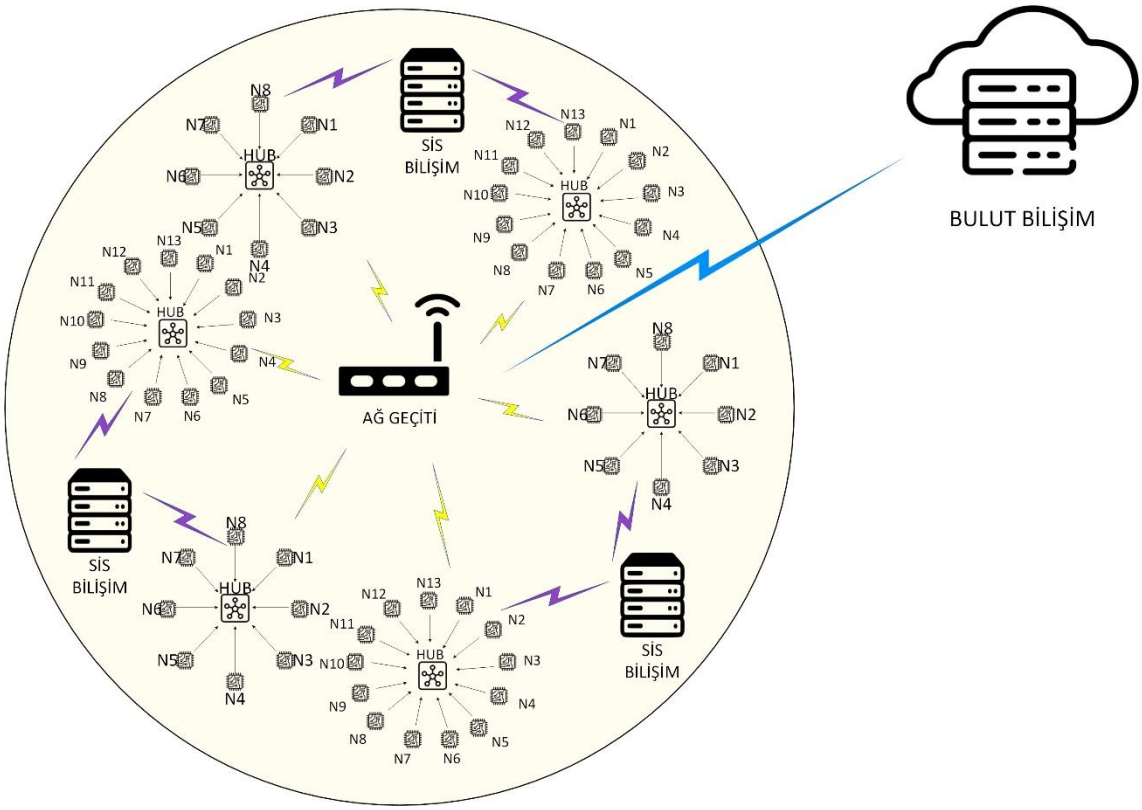
3. ÖNERİLEN MEDİKAL NESNELERİN İNTERNETİ (IOMT) ÇERÇEVESİ

Önerilen IoMT çerçevesi, Şekil 3.1.'de görülebileceği gibi KVAA içi ve KVAA'lar arası IoT yetenekleri olan IEEE 802.15.6 tabanlı veri kaynağı katmanı ve çeşitli diyabet ve kalp hastalığı tahmin yeteneklerine sahip Sis ve Bulut bilişim katmanlarından oluşmaktadır. KVAA'ları daha gerçekçi performans analizleri yapılması için Riverbed Modeler benzetim ortamında geliştirilmiştir. Riverbed Modeler benzetim ortamında KVAA'lar diyabet ve kalp hastalığı tahmini için büyük ölçekli sağlık verileri üretmektedir. Her KVAA'ın çeşitli vücut algılayıcı düğümleri ve bir koordinatör düğümü vardır. Ayrıca, KVAA'lar birbirleri üzerinden haberleşebilmektedir. Şekil 3.1.'de, her insan vücudu vücut sensörleri ile donatılmıştır. KVAA benzetim senaryolarından üretilen sağlık verileri, diyabet ve kalp hastalığı tahmini için Sis ve Bulut bilişim katmanlarına iletilir. Sis bilişim katmanı, KVAA ortamının yakınında konuşlandırılır ve bulanık mantık yardımıyla diyabet ve kalp hastalığını tahmin etme yeteneğine sahiptir. Bulut bileşeni, KVAA'lardan uzaktır. Bulut bileşeni, makine öğrenmesi algoritmaları yardımıyla diyabet ve kalp hastalığını tahmin etme yeteneğine sahiptir. Benzetim senaryomuzda, Sis bilişim KVAA'lara yakın konumlandırılırken, Bulut bilişim ise Şekil 3.2.'de görüldüğü gibi KVAA'lardan uzakta konumlandırılmıştır. Sis bilişim katmanında bulanık mantık çıkarım sistemi ile diyabet ve kalp hastalığı tahminleri yapılmaktadır. Bulut bilişim katmanı ise veri akış, veri analitiği ve görselleştirme ve depolama bileşenleri olmak üzere üç bileşenden oluşmaktadır. Veri akış bileşeninde Node-Red ve Apache Kafka veri akış platformları ile veriler diğer bileşenlere hızlı bir şekilde iletilmektedir. Veri analitiği bileşeninde ise Apache Spark veri işleme çerçevesi ile makine öğrenmesi algoritmaları kullanılarak gerçek zamanlı diyabet ve kalp hastalığı tahminleri yapılmaktadır. Son olarak görselleştirme ve depolama bileşeninde gerçek zamanlı veriler ve hastalık tahmin sonuçları Canlı Web Panel'de gösterilmekte ve MongoDB veri tabanı yönetim sisteminde oluşturulan veri tabanı içerisinde depo edilmektedir. Şekil 3.3.'te önerilen IoMT çerçeve bileşenlerinin detayları gösterilmiştir.

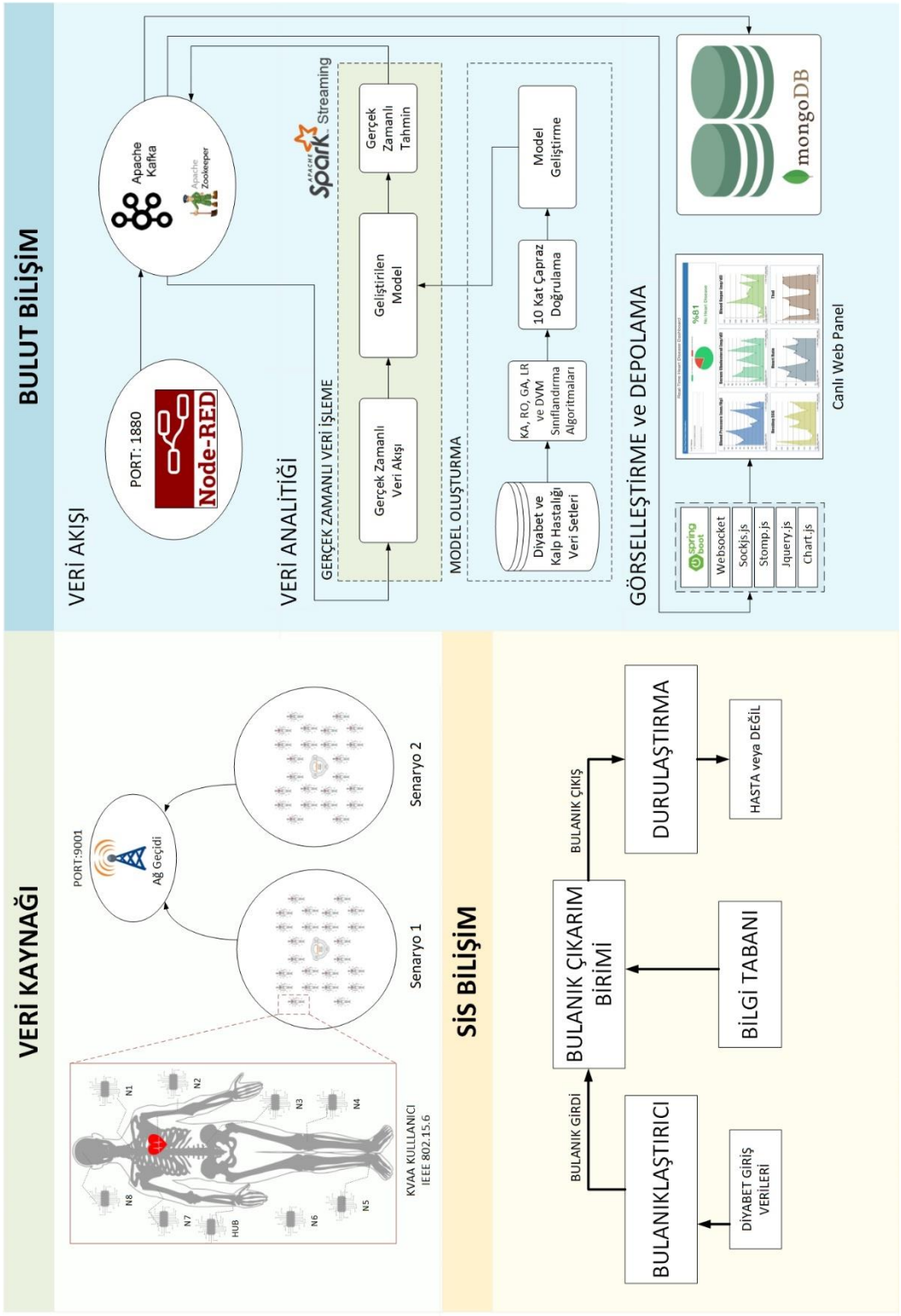


Şekil 3.1. Önerilen IoMT mimarisi.

Önerilen çerçevenin genel topolojisi Şekil 3.2.'de gösterilmiştir. Bir diyabet hastalığı KVAA sekiz sağlık sensör düğümünden (N1-N8) ve koordinatör olarak bir HUB'dan oluşurken, bir kalp hastalığı KVAA on üç sağlık düğümünden (N1-N13) ve koordinatör olan bir HUB'dan oluşmaktadır. Her HUB, sağlık sensörü verisini toplar ve düğümlerin önceliklerine göre ağ geçidine iletir. HUB'lar ayrıca bulanık mantık tabanlı diyabet tahmini için en yakın Sis bilişimle iletişim kurar. KVAA'lar, AODV yönlendirme algoritması sayesinde diğer KVAA'larla iletişim kurabilme özelliğine sahiptir. Şekil 3.2.'de görülebileceği gibi Bulut bileşeni, iletişim kapsamı nedeniyle yalnızca ağ geçidi ile iletişim kurar. Bulut bileşeni, diyabet ve kalp tahmini için makine öğrenmesi özelliklerine sahiptir. Önerilen IoMT çerçevesinde bulunan IEEE 802.15.6 tabanlı veri kaynağı, Sis bilişim ve Bulut bilişim katmanları ayrı başlıklar altında detaylı bir şekilde açıklanmaktadır.



Şekil 3.2. Önerilen IoMT çerçevesinin topolojisi.

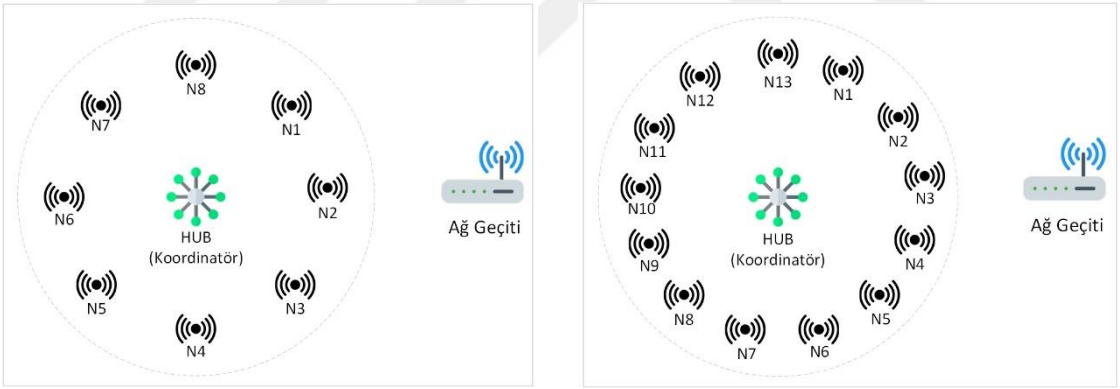


Şekil 3.3. Önerilen IoMT çerçevesi genel yapısının detaylı görünümü.

3.1. IEEE 802.15.6 TABANLI VERİ KAYNAĞI KATMANI

Veri kaynağı katmanı, IEEE 802.15.6 standardı kullanılarak geliştirilen diyabet ve kalp hastalığı benzetim senaryosundan oluşmaktadır. IEEE 802.15.6 standardı, KVAA'ları yapıları için tüm iletişim kurallarını tanımlar. KVAA'larda 2.4 GHz endüstriyel, bilimsel ve tıbbi bantlarda öncelik tabanlı bir yaklaşımla CSMA/CA tabanlı ortam erişim kontrol mekanizması kullanılmaktadır. Diyabet ve kalp hastalığına yönelik sağlık verilerinin üretimi için, IEEE 802.15.6 tabanlı KVAA'lar, Proto-C programlama dili ile Riverbed Modeler'da modellenmiştir. Riverbed Modeler, Ad-Hoc ve altyapı ağlarını tüm özellikleri ile modelleyebilen bir benzetim yazılımıdır.

KVAA'lardaki her sensör düğümü, belirli sensör değerleri üretme yeteneğine sahiptir ve koordinatör olarak kullanılan HUB, sensör düğümlerinden alınan sağlık verilerinin toplanmasından sorumludur. Diyabet hastalığı için sekiz sensör düğümü (N1-N8) ve kalp hastalığı için on üç sensör düğümü (N1-N13) olarak senaryolarda kullanılmaktadır. Diyabet ve kalp hastalığı benzetim senaryolarının gösterimi Şekil 3.4. a), b)'de sunulmuştur.



a) b)

Şekil 3.4. Benzetim senaryoları gösterimi a) Diyabet b) Kalp hastalığı.

IEEE 802.15.6 standardında erişim kategorisine göre farklı trafik öncelikleri vardır ve bu öncelikler Çizelge 3.1.'de gösterilmiştir. Bu farklı trafik türleri, acil veri, yüksek öncelikli tıbbi veri, tıbbi veri veya ağ kontrolü, ses, video, mükemmel efor, en iyi efor olarak ayrılmaktadır. Ayrıca, bu farklı veri trafikleri, CSMA/CA mekanizması için minimum ve maksimum Çatışma Penceresi (ÇP) değerleri ile kullanıcı tercihini gerçekleştirir [76].

Çizelge 3.1. Kullanıcı öncelikli haritalama (D: Veri – Y: Yönetim).

P	Trafik Türü	Paket Tipi	CSMA/CA	
			$ÇP_{min}$	$ÇP_{max}$
0	Arkaplan	D	16	64
1	En İyi Efor	D	16	32
2	Mükemmel Efor	D	8	32
3	Video	D	8	16
4	Ses	D	4	16
5	Tıbbi Veri	D/Y	4	8
6	Yüksek Öncelikli Tıbbi Veriler	D/Y	2	8
7	Acil Veri	D	1	4

3.2. IoMT ÇERÇEVESİNİN SİS BİLİŞİM KATMANI

Sis bilişim katmanı, veri kaynağı ile Bulut bilişim katmanı arasında bulunur. Bu nedenle, Sis bilişim katmanı KVAA'ların yakınında bulunur. Sis bilişim, verileri merkezi bir sunucuya aktarmanın aksine yerel olarak veri analizi ve depolama sağlar. Ayrıca, bazı uygulamalarda verilerin mümkün olduğunca çabuk işlenmesi gerekebileceğinden, Sis bilişim daha hızlı kararlar için sınırlı işlemlere sahiptir. Sis bilişim, günümüzde IoT konsepti ile ön plana çıkmaktadır.

Sis bilişim hesaplamanın yukarıda bahsedilen avantajları ışığında, bu çalışmada diyabet tahmininde hızlı kararlar için bir Sis bilişim katmanını önermekteyiz. Diyabet hastalığının pozitif veya negatif karar süreci için Sis bilişim katmanında bulanık mantık kullanılmıştır.

Çizelge 3.2. Diyabet hastalığı veri seti öznitelikleri ve değer aralıkları.

Veri	Dilsel Değişken	Aralık
Glikoz (PGC)	Abnormal	187 – 232
	Medium	141 – 186
	Normal	44 – 140
Diyastolik Kan Basıncı (mm Hg) (DBP)	Abnormal	92-122
	Medium	81-91
	Normal	30-80
Vücut Kitle Endeksi (BMI)	Abnormal	34-67
	Medium	26-33
	Normal	18-25
Diyabetin Pedigri İşlevi (DPF)	Abnormal	0,54-2,29
	Medium	0,50-0,53
	Normal	0,08-0,49
Gebelik Sayısı (NOP)	Abnormal	10-17
	Medium	5-9
	Normal	0-4

Önerilen diyabet hastalığı bulanık mantık yapısında, Çizelge 3.2'de gösterildiği gibi glikoz, kan basıncı, vücut kitle indeksi, diyabet soyağacı işlevi ve gebelik sayısı olmak üzere beş girdiye sahiptir. Bu değerler literatürdeki çalışmalardan referans alınarak seçilmiştir [33]. Her girdinin anormal, orta ve normal durumlar olmak üzere üç dilsel değişkeni vardır. Diyabet bulanık mantık sisteminde her bir değişkeni üç üyelik fonksiyonu tanımlar. Tüm üyelik fonksiyonları, Çizelge 3.2'de belirtildiği gibi aralıklar ve değerleri ile oluşturulmuştur.

Çizelge 3.3. Diyabet hastalığına ait bulanık mantık kurallarından bazıları.

PGC	DBP	BMI	DPF	NOP	Tahmin
Normal	Normal	Normal	Normal	Normal	Negatif
Abnormal	Abnormal	Abnormal	Abnormal	Abnormal	Pozitif
Normal	Normal	Normal	Medium	Medium	Negatif
Medium	Normal	Abnormal	Normal	Normal	Pozitif
Medium	Normal	Medium	Medium	Normal	Negatif
Medium	Normal	Medium	Abnormal	Abnormal	Pozitif
Medium	Medium	Medium	Normal	Normal	Negatif
Abnormal	Medium	Normal	Medium	Medium	Pozitif
Normal	Normal	Medium	Medium	Abnormal	Negatif
Normal	Abnormal	Medium	Abnormal	Abnormal	Pozitif

Çizelge 3.3'te önerilen diyabet hastalığına yönelik bulanık mantık kurallarından bazıları verilmiştir ve sistemde 243 olası kural vardır. Çizelge 3.3'teki her satırda altı sütun vardır ve ilk beş sütun girdilerdir ve son sütun diyabet hastalığını çıktı olarak teşhis eder.

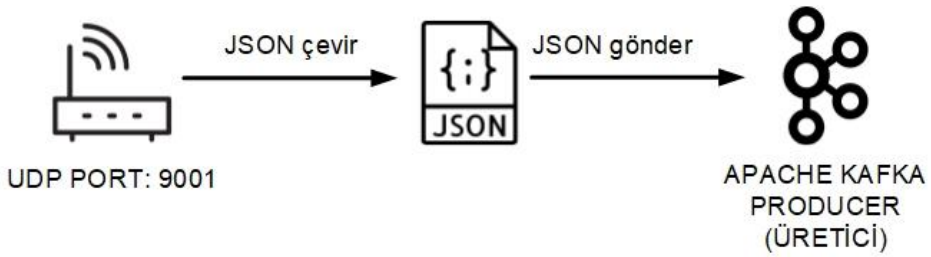
3.3. IoMT ÇERÇEVESİNİN BULUT BİLİŞİM KATMANI

Bulut bilişim, yüksek kapasiteli hesaplamalar, veri depolama ve görselleştirme yetenekleri gibi özellikleri ile Sis bilişime göre daha iyi olanaklar sunmaktadır. Ayrıca, Bulut bilişim, ağ oluşturma, veri analitiği, çeşitli yazılımlar ve yapay zeka/makine öğrenmesi tabanlı özellikler sağlamaktadır. Bulut bilişim katmanı veri kaynağından uzaktır ve çok daha fazla müşteriye hizmet verilebilmektedir.

Bulut bilişimin yukarıda bahsedilen avantajları ışığında, çalışmamızda diyabet ve kalp hastalığı tahmininde sınırsız sağlık verileri ile yüksek kapasiteli dağıtık tabanlı hesaplamalar yapabilen ve veri görselleştirme ve depolama özelliği olan bir Bulut bilişim katman yapısını önermekteyiz. Önerilen Bulut bilişim katmanı, veri akışı, veri analitiği ve görselleştirme ve depolama olmak üzere üç bileşenden oluşmaktadır. Bu bileşenler farklı başlıklar halinde ayrıntılı olarak açıklanmıştır.

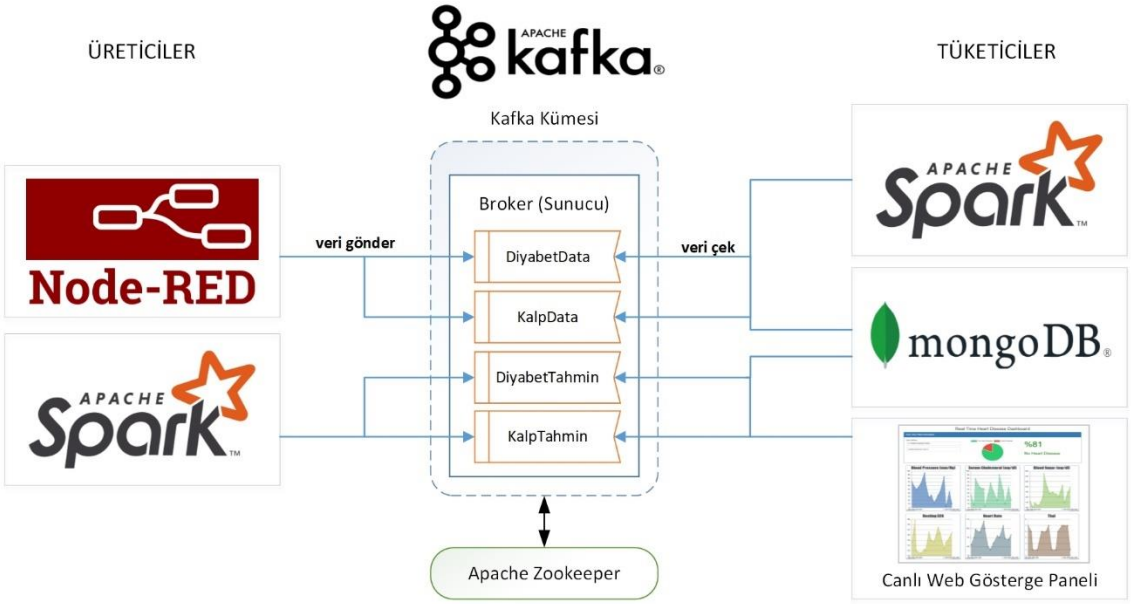
3.3.1. Veri Akışı Bileşeni

Veri akışı bileşeninde iki farklı veri akış platformu kullanılmaktadır. Birincisi Node-Red, ikincisi ise Apache Kafka veri akış platformlarıdır. Bu bileşende amaç, ilgili KVAA sağlık sensör verilerini hızlı ve hatasız bir şekilde hastalık tahminleri için veri analitiği bileşenine aktarmaktır. KVAA sağlık sensör verileri benzetim senaryolarından UDP:9001 portuna iletilmektedir. Burada, UDP protokolü gerçek zamanlı ve daha hızlı tıbbı uygulamalarda daha kullanışlı olduğundan tercih edilmektedir [77]. UDP:9001 portuna gelen KVAA sağlık sensör verileri, veri analitiği bileşeninde bulunan Apache Spark analiz motoruna ve depolama bileşenine doğrudan iletilmediği için arada veri akış bileşeni geliştirilmiştir. Bu bileşen ile KVAA sağlık sensör verileri diğer bileşenlere hızlı ve sağlıklı bir şekilde iletilmektedir.



Şekil 3.5. Node-Red veri akış platformunun çalışma şekli.

Bulut bileşeninde ilk olarak, diyabet ve kalp hastalığı benzetim senaryolarından gönderilen KVAA sensör verileri Node-Red platformunda geliştirilen veri akış sistemine gelir. Node-Red, çeşitli donanım cihazlarının ve uygulamaların çevrimiçi iletişim kurmasını sağlayan Node.js tabanlı bir programlama aracıdır [78]. Node-Red platformunda geliştirilen veri akış sisteminin çalışma şekli Şekil 3.5.'de gösterilmektedir. Burada amaç, UDP: 9001 portundan alınan KVAA sensör verileri veri analitiği bileşeni için JSON formatına dönüştürülerek bir sonraki veri akış platformu olan Apache Kafka'daki ilgili hastalık konusuna (topic) gönderilir.



Şekil 3.6. Apache Kafka veri akış platformu çalışma şekli.

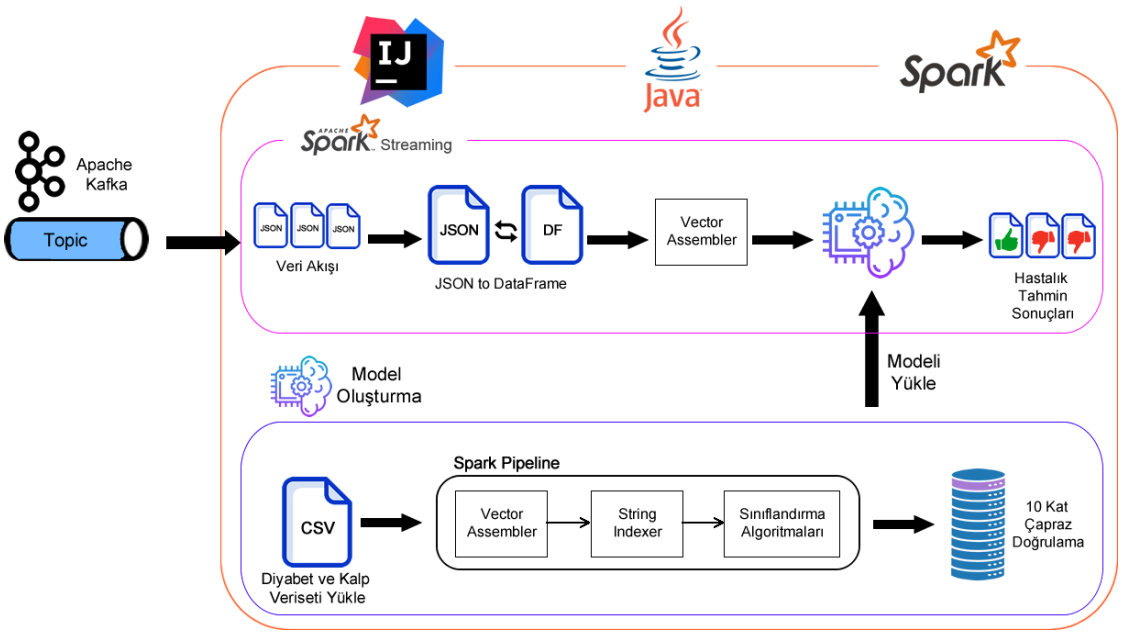
Şekil 3.6.'da çalışma şekli gösterilen Apache Kafka, ikinci veri akış platformu olarak kullanılmaktadır. Apache Kafka, Node-Red ve Apache Spark üreticileri (producers) tarafından gönderilen gerçek zamanlı verileri alarak, kendisine abone olan Apache Spark, MongoDB ve Canlı Web Gösterge Paneli tüketicilerine (consumers) iletmekten sorumludur. Apache Zookeeper ise Apache Kafka veri akış yapısının koordinasyonunu sağlamaktadır. Apache Kafka sisteminde bir Broker (komisyoncu) ve DiyabetData, KalpData, Diyabet Tahmin ve KalpTahmin olmak üzere dört farklı konu (topic) oluşturulmuştur.

Üreticiler arasında ilk olarak Node-Red veri akış platformu bulunmaktadır. Node-Red veri akış sisteminden Apache Kafka üzerinde Broker'larda bulunan DiyabetData ve KalpData konularına ilgili gerçek zamanlı KVAA sağlık sensör verileri gönderilmektedir. Ardından Apache Spark veri analitik sisteminde yapılan hastalık tahminleri sonuçları DiyabetTahmin ve KalpTahmin isimli ilgili Apache Kafka konularına gönderilmektedir.

Tüketiciler arasında bulunan Apache Spark, hastalık tahminleri için Apache Kafka'ya abone olarak DiyabetData ve KalpData konusundaki verileri kendi sistemine çekmektedir. Farklı bir tüketici olan MongoDB, DiyabetData, KalpData, DiyabetTahmin ve KalpTahmin konularına abone olarak bu konulara gelen verileri kendi üzerinde depolamaktadır. Son olarak ise Canlı Web Gösterge paneli, DiyabetTahmin ve KalpTahmin konularına gelen verileri alarak gerçek zamanlı olarak kullanıcılara sunmaktadır.

3.3.2. Veri Analitiği Bileşeni

Veri analitiği bileşeninde, Apache Kafka'daki ilgili hastalık konularından (topic) abone olunarak alınan KVAA sensör verileri gerçek zamanlı olarak analiz edilmiş ve hastalıklara yönelik tahminler yapılmıştır. Veri analitiği bileşeni, Intellij IDEA kod editörü üzerinde Java programlama dili kullanılarak bir Maven projesi ile geliştirilmiştir. Ayrıca, veri analitiği bileşeninde daha önce detaylı bir şekilde anlatılan Apache Spark veri işleme platformu kullanılarak hastalık tahminleri yapılmıştır. Veri analitiği bileşeninin çalışma yapısı Şekil 3.7.'de gösterilmiştir.



Şekil 3.7. Veri analitiği bileşeni çalışma yapısı.

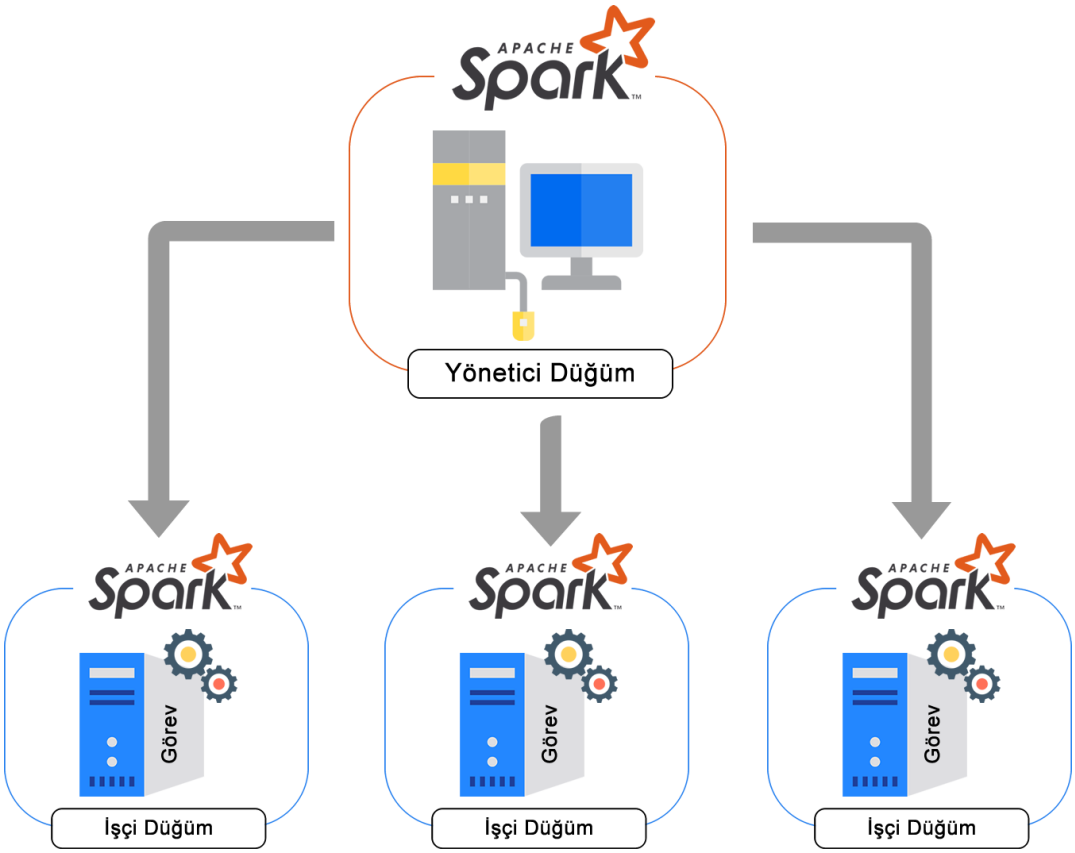
Hastalık tahminleri için öncelikle Apache Spark MLlib makine öğrenmesi kütüphanesinden çeşitli sınıflandırma algoritmalarının kullanıldığı tahmin modeli oluşturulmuştur. Şekil 3.7.'de gösterilen model oluşturma aşamasında ilk olarak veri seti bölümünde detaylandırılan diyabet veya kalp hastalığı veri seti sisteme yüklenmiştir. Çalışmada kullanılan diyabet [79] ve kalp hastalığı [80] veri setlerinin detayı Çizelge 3.4.'te verilmiştir. Sisteme yüklenen veri seti, Apache Spark Pipeline (boru hattı) bölümünde çeşitli veri ön işlemelerinden geçirilerek tahmin modeli için hazır hale getirilmiştir. Apache Spark Pipeline sınıflandırma algoritmaları aşamasında Spark MLlib kütüphanesinden KA, RO, GA, LR ve DVM sınıflandırıcıları kullanılmıştır. Son olarak tahmin modelinin en iyi hale getirilmesi için 10 kat çapraz doğrulama kullanılmış ve tahmin modelinin son hali verilmiştir. Hastalık tahmin modeli daha sonra gerçek zamanlı hastalık tahmini için kullanılmıştır.

Diyabet ve kalp hastalığının gerçek zamanlı tahmini için öncelikle Apache Kafka üzerindeki ilgili hastalık konusuna (topic) abone olunmuştur. Buradaki sağlık verileri sürekli olarak her 10 sn'de bir toplu olarak Apache Spark tarafından alınmıştır. JSON formatında toplu olarak alınan veriler, Apache Spark'ın veri analizinde kullanılan Data Frame (DF) formatına çevrilmiştir. DF formatına çevrilen veriler VectorAssembler ile vektör haline getirilmiştir. Tüm bu işlemlerden sonra hastalık tahmin modeli sisteme yüklenmiş ve gerçek zamanlı olarak hastalık tahminleri gerçekleştirilmiştir. Hastalık tahmininden alınan sonuç verileri daha sonrasında Görselleştirme ve Depolama bileşenine gönderilmiştir.

Çizelge 3.4. Diyabet ve kalp hastalığı veri setlerinin özellikleri.

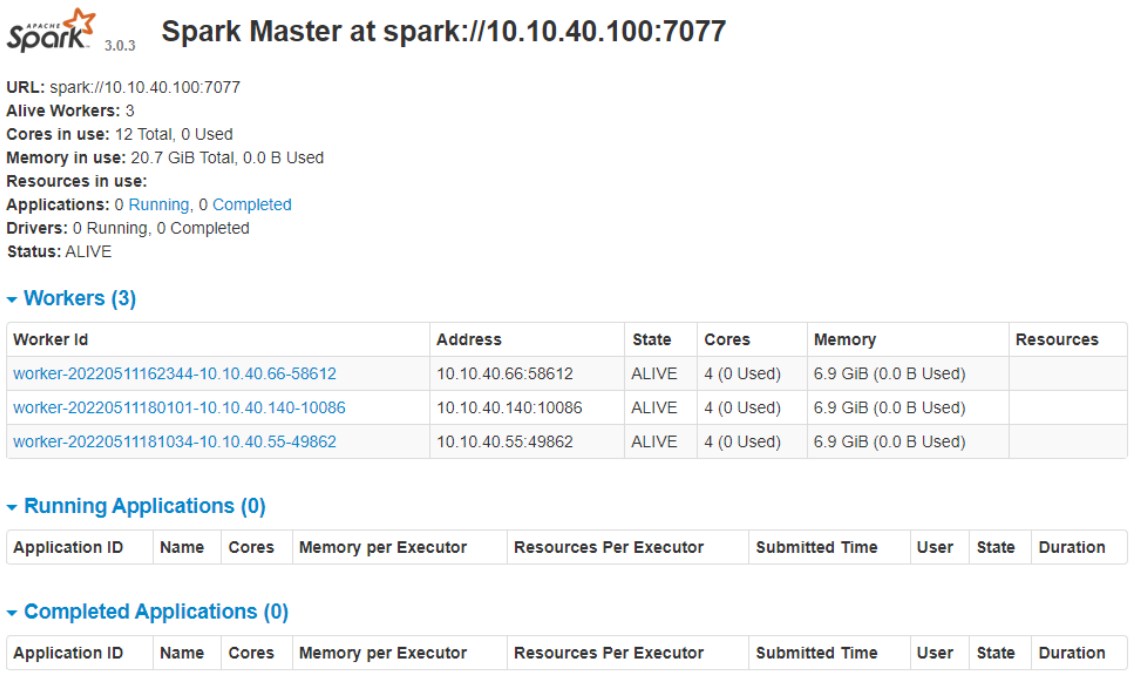
Veri Seti	**Parametreler**	**Etiket**	**Örnek Sayısı**
Diyabet Veri Seti	gebelik sayısı, glikoz, kan basıncı, insulin, cilt kalınlığı, vücut kütle endeksi, diyabet pedigri işlevi, yaş	hasta, hasta değil	768
Kalp Hastalığı Veri Seti	yaş, cinsiyet, göğüs ağrısı tipi, kan basıncı, kolestrol, kan şekeri, ekg, kalp atış hızı, egzersize bağlı anjina, dinlenmeye bağlı ST depresyonu, yoğun egzersiz ST segmentinin eğimi, renkli büyük damar sayısı, thal	hasta, hasta değil	303

Apache Spark veri işleme motorunun en önemli özelliklerden biri de dağıtık olarak hesaplamalar yapabilmesidir. Apache Spark'ın dağıtık olarak işlem yapmasındaki avantaj üzerindeki işlem yükünü farklı işçi düğümlere aktararak analizlerin daha hızlı yapılmasını sağlamaktır. Ayrıca, ölçeklenebilirlik özelliği ile bu düğümler arttırılarak sistem daha güçlü ve güvenli bir hale getirilebilmektedir. Bu çalışmada Apache Spark'ın dağıtık hesaplama özelliğini kullanılmış ve veri analizleri dağıtık olarak gerçekleştirilmiştir.



Şekil 3.8. Apache Spark dağıtık hesaplama yapısı.

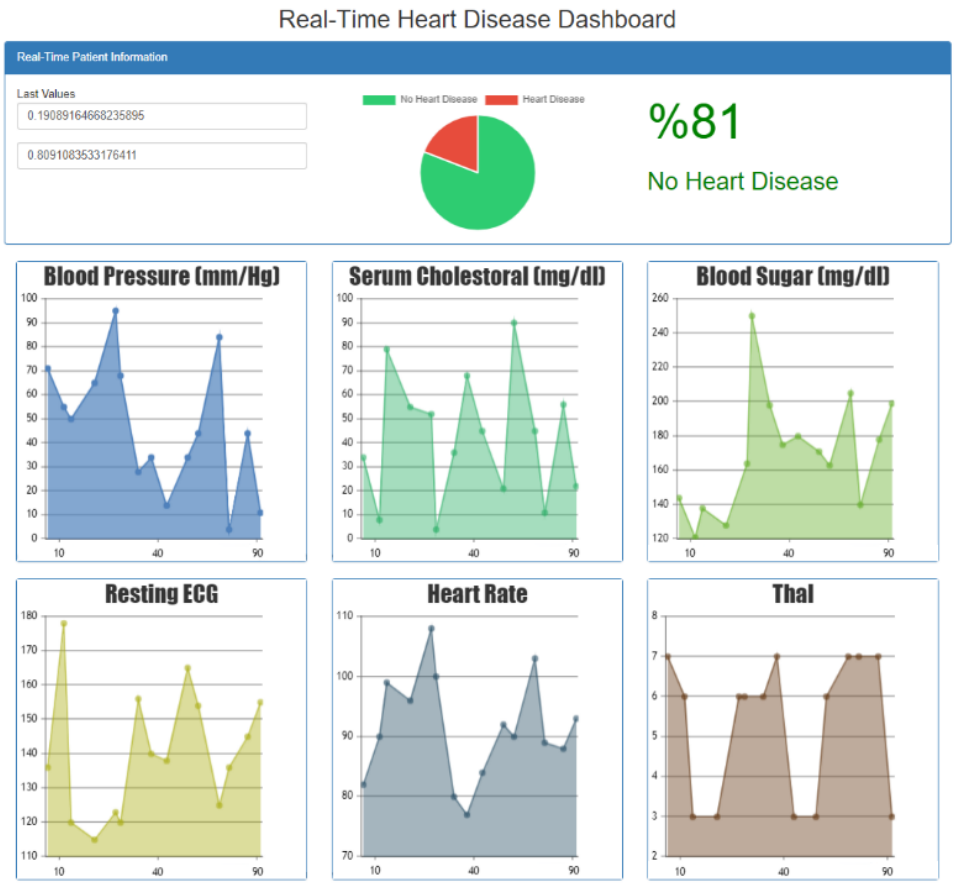
Şekil 3.8.'de çalışmada kullanılan Apache Spark veri analitiği sistemine ait dağıtık hesaplama sistem mimarisi gösterilmiştir. Bu sistemde bir yönetici düğüm ve üç işçi düğüm bulunmaktadır. Yönetici düğüm işçi düğümlere görevler vermek ve onları yönetmek ile görevlidir. İşçi düğümler ise yönetici düğümün vermiş olduğu görevleri yapmak ve sonuçlarını ona geri göndermekle görevlidir. Apache Spark dağıtık hesaplama yapan düğümlerin Spark UI ekranındaki günümü Şekil 3.9.'da gösterilmiştir.



Şekil 3.9. Apache Spark üzerinde çalışan düğümlerin Spark UI görünümü.

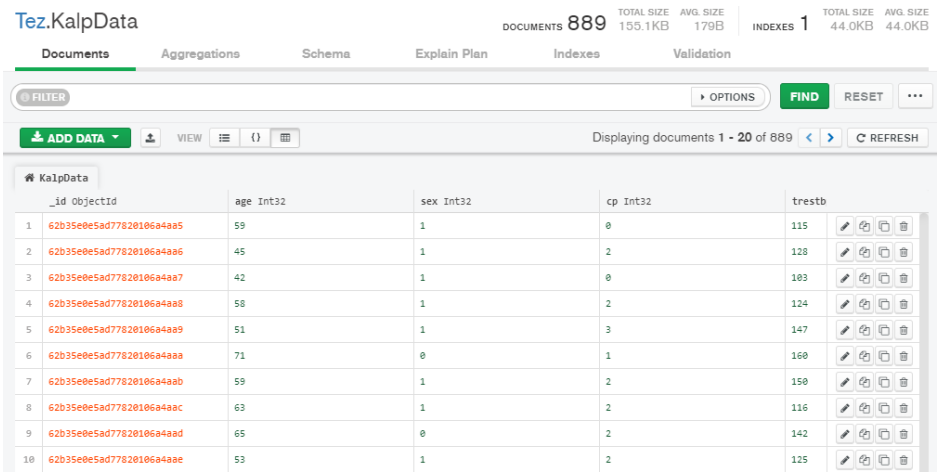
3.3.3. Görselleştirme ve Depolama Bileşeni

Bu bileşende görselleştirme ve depolama olmak üzere iki farklı sistem yapısı kullanılmıştır. Görselleştirme sistemi olarak KVAA sağlık sensör verilerinin ve Apache Spark tarafından yapılan hastalık tahmin sonuçlarının anlık olarak gösterildiği Canlı Web Panel geliştirilmiştir. Canlı Web Panel, Java tabanlı Spring Boot Framework ve çeşitli javascript uygulamaları (jquery, chart, stomp ve sock) kullanılarak geliştirilmiştir. Web Socket, gerçek zamanlı analiz sonuçlarını anında görüntülemek için kullanılmıştır. Böylece, analiz edilen veriler Web panosunda hızlı bir şekilde görüntülenmiştir. Canlı Web Panel http://10.10.40.100:5656 adresinde çalışmaktadır. Şekil 3.10.'da, benzetim senaryoları üretilen gerçek zamanlı KVAA sağlık sensör verilerinin ve hastalık tahmin sonuçlarının yayınlandığı Canlı Web Panel'inin ekran görüntüsü sunulmuştur. Hastalıkların tahminleri, senaryolardan gönderilen sağlık verilerini alan Apache Spark veri analitiği bileşeni gerçek zamanlı olarak tahminlerde bulunarak hastalıkların yüzdelik tahminini ve sonucunu Canlı Web Panel'e göndermiştir.



Şekil 3.10. Canlı Web Panel temsili görüntüsü.

Bu bileşende büyük veri depolama sistemi olarak MongoDB veri tabanı yönetim sistemi kullanılmıştır. MongoDB, açık kaynaklı bir NoSQL veri tabanıdır. Bu veri tabanı yönetim sistemi kullanılarak, büyük miktardaki veriler sistemdeki oluşturulan veri tabanlarında doküman formatında saklanabilmekte ve ölçeklenebilirlik özelliği sayesinde kolaylıkla analiz edilebilmektedir.



Şekil 3.11. MongoDB NoSQL veri tabanının temsili görüntüsü.

MongoDB veri tabanı yönetim sisteminde, diyabet ve kalp hastalığı benzetim senaryolarından gönderilen KVAA sağlık sensör verilerini ve hastalık tahmin sonuçlarını depolamak için Tez adında bir veri tabanı oluşturulmuştur. Tez veri tabanında içerisinde DiyabetData, KalpData, DiyabetTahmin ve KalpTahmin olmak üzere dört farklı koleksiyon (tablo) oluşturulmuştur. MongoDB veri tabanı yönetim sisteminde bulunan bir koleksiyonunun örnek ekran görüntüsü Şekil 3.11.'de gösterilmiştir.

3.4. MAKİNE ÖĞRENMESİ SINIFLANDIRMA ALGORİTMALARININ DEĞERLENDİRİLMESİ

Makine öğrenmesi sınıflandırıcılarının performanslarını değerlendirmek için Doğruluk, Kesinlik, Duyarlılık ve F1-Skor olmak üzere dört performans ölçütü kullanıldı. Bu performans ölçütlerini hesaplamak için Karmaşıklık Matrisi ve Çapraz Doğrulama yöntemleri kullanılmıştır. Önerilen IoMT çerçevesinde, benzetim senaryolarında üretilen sağlık verilerinin gerçek sonuçları bilinmediğinden, veri analitiğinde kullanılan algoritmaların tahmin sonuçları ile karşılaştırılamamaktadır. Bu nedenle sınıflandırma algoritmalarının performans sonuçları sistem içerisinde kullanılan eğitim veri setleri kullanılarak karşılaştırılmıştır. Diyabet ve kalp hastalığı veri setlerinin %70'i eğitim verisi ve %30'u test verisi olarak kullanılmıştır. Değerlendirme yapılırken veri setlerindeki hasta ve hasta olmayan veri sayıları eşitlenmiştir. Karmaşıklık Matrisi, 2x2 matris oluşturarak ikili sınıf için sınıflandırma algoritmalarının performansını görselleştirmek için kullanılır. Matriste her satır tahmin edilen sınıfların sayısını gösterirken, her sütun gerçek sınıftakilerin sayısını gösterir. Oluşturulan matris sonucunda Doğru Pozitif (DP), Yanlış Pozitif (YP), Doğru Negatif (DN) ve Yanlış Negatif (YN) olmak üzere dört değer oluşur. Bu değerler kullanılarak hesaplanan performans ölçütlerinin formülü aşağıdaki denklemler de sunulmaktadır (3.1)-(3.4).

$$Doğruluk = \frac{DP + DN}{DP + YP + DN + YN} \tag{3.1}$$

$$Kesinlik = \frac{DP}{DP + YP} \tag{3.2}$$

$$Duyarlılık = \frac{DP}{DP + YN} \tag{3.3}$$

$$F1 - skor = \frac{2 * Kesinlik * Duyarlılık}{Kesinlik + Duyarlılık} \tag{3.4}$$

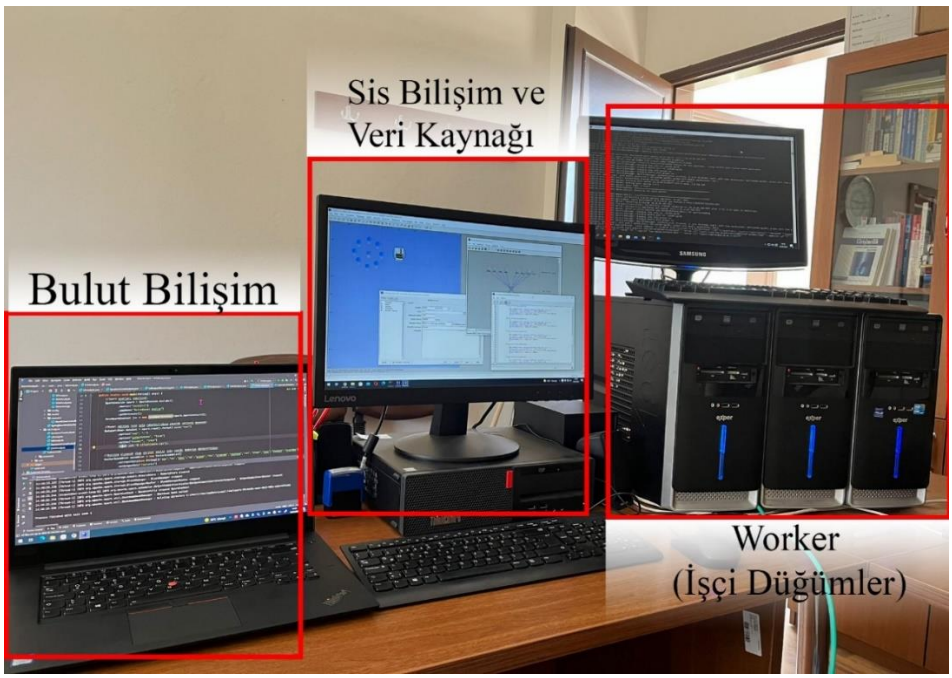
Çapraz Doğrulama ise makine öğrenmesi sınıflandırıcılarının performansını ölçebilen istatistiksel bir yöntemdir. Bu yöntemde, veri seti önceden belirlenen k değeri kadar parçaya bölünür. Bu çalışmada k değeri 10 olarak belirlenmiştir (k=10). Yöntemde sınıflandırma işlemi 10 kez yapılmaktadır ve her adımda bölünen parçalardan bir tanesi test işlemi için ayrılmaktadır. Geriye kalan 9 tanesi sınıflandırıcının eğitimi için kullanılmaktadır. 10 adım sonra elde edilen sınıflandırma sonuçlarının ortalaması alınarak genel sonuç elde edilmektedir.

4. UYGULAMA SÜRECİ

Bu bölümde tez çalışmasında önerilen IoMT çerçevesinin uygulama süreci açıklanmaktadır. Uygulama süreci, çalışma ortamı, uygulama ortamının hazırlanması ve uygulamanın gerçekleştirilmesi olmak üzere üç başlık altında sunulacaktır.

4.1. ÇALIŞMA ORTAMI

Tez çalışmasında, Riverbed benzetim ortamında diyabet ve kalp hastalığına dair KVAA sağlık sensör verilerinin üretilmesi için senaryolar oluşturulmuştur. Benzetim senaryolarından gönderilen sağlık verileri Sis bilişim katmanında bulanık mantık ve Bulut bilişim katmanında Apache Spark makine öğrenmesi algoritmaları kullanılarak analizler dağıtık olarak yapılmaktadır. Benzetim senaryoları ve Sis bilişim için bir düğüm, Bulut bilişimdeki veri akışı, veri analitiği yöneticisi, veri depolama ve canlı web panel için bir düğüm ve veri işleme dağıtık olarak yapıldığı için üç işçi düğüm olmak üzere çalışmada toplamda beş düğüm yani PC kullanılmıştır. Çizelge 4.1.'de, çalışmada kullanılan düğümlerin konfigürasyonları verilmiştir. Tüm düğümler ağ aynı ağ üzerinden iletişim kurmaktadır. Ayrıca, çalışma ortamının temsili bir görüntüsü Şekil 4.1.'de gösterilmiştir.



Şekil 4.1. Çalışma ortamı.

Çizelge 4.1. Düğüm konfigürasyonları.

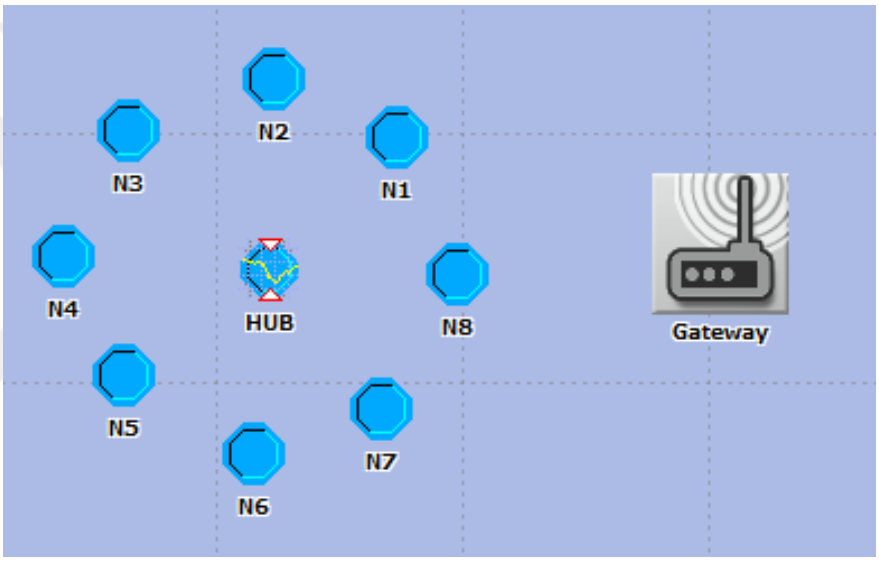
Özellik	Düğüm 1	Düğüm 2	Düğüm 3-4-5
İşlemci Türü	Intel i5	Intel i7	Intel i5
İşlemci Hızı	3.00 GHz	4.7 GHz	3.20 GHz
İşlemci Çekirdek Sayısı	4	12	4
RAM Bellek Kapasite	8 GB	32 GB	8 GB
RAM Hızı	2400 MHz	3200 MHz	2400 MHz
Sabit Disk	256 GB SSD	1TB SSD	256 GB SSD
Ağ Hızı	1 Gigabit	1 Gigabit	1 Gigabit
Kurulu Yazılımlar	Riverbed Modeler	IntelliJ IDEA 2020 MongoDB Node-Red Apache Spark 3.0.3 Apache Kafka 3.1 Apache Zookeeper Java JDK 8	Apache Spark 3.0.3 Java JDK 8
İşletim Sistemi	Windows 10 64 bit		
Açıklama	Senaryolar ve Sis bilişim bu düğümde çalışmaktadır.	Bulut Sistemindeki uygulamalar (Node-Red, Apache Kafka, Apache Spark, Canlı Görsel Panel ve MongoDB) bu düğümde çalışmaktadır.	Apache Spark dağıtık hesaplama için gerekli işçi düğümler olarak çalışmaktadırlar.

4.2. UYGULAMA GELİŞTİRME SÜRECİ

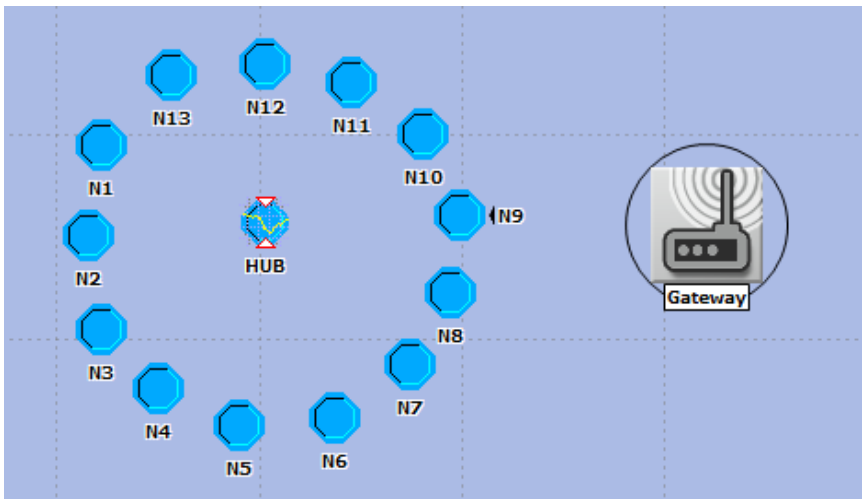
Bu bölümde önerilen IoMT çerçevesinin geliştirme aşamaları genel bir çerçeve ile anlatılmıştır.

4.2.1. Riverbed Modeler Benzetim Senaryolarının Oluşturulması

Riverbed benzetim yazılımında Diyabet ve kalp hastalığı senaryoları, gerçek zamanlı sağlık veriler üretmek için geliştirilmiştir. Diyabet ve kalp hastalığı senaryolarının Riverbed ortamındaki görünümü sırasıyla Şekil 4.2. ve Şekil 4.3.'te gösterilmiştir. Diyabet senaryosunda 8 düğüm, 1 HUB ve 1 Gateway kullanılmıştır. Kalp hastalığı senaryosunda ise 13 düğüm, 1 HUB ve 1 Gateway kullanılmıştır.

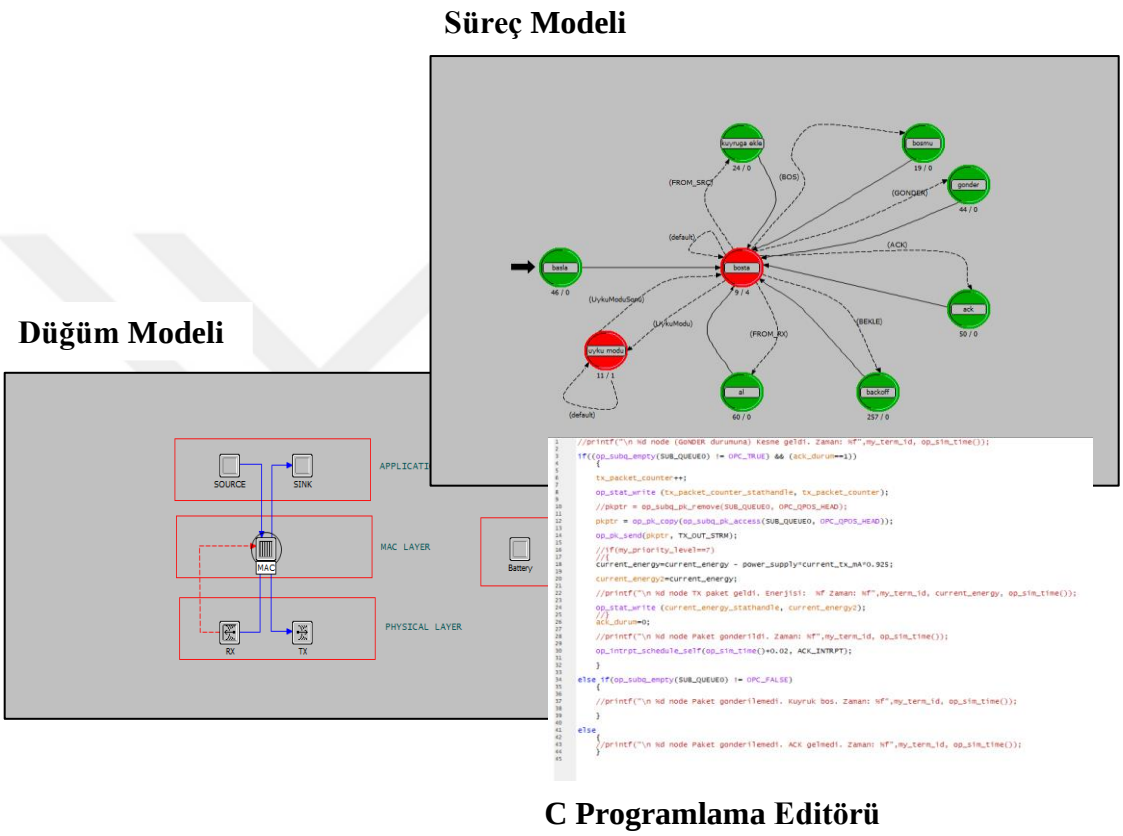


Şekil 4.2. Diyabet hastalığı KVAA senaryosu.



Şekil 4.3. Kalp hastalığı KVAA senaryosu.

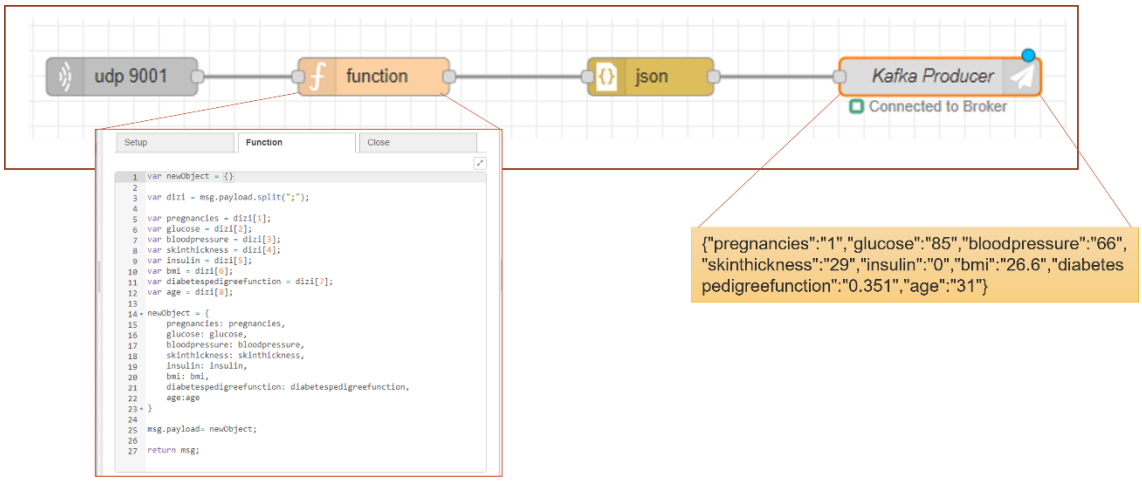
Bu senaryolarda düğümlerden gelen sağlık verileri koordinatör olarak kullanılan HUB düğümünde toplanır. Bu verilerin Bulut bilişim katmanında analiz edilebilmesi Ağ Geçiti'ne gönderilir. Şekil 4.4.'de Riverbed Modeler benzetim yazılımında CSMA/CA tabanlı Diyabet ve Kalp hastalığı senaryoları için tasarladığımız düğüm, süreç ve C tabanlı kodun ekran görüntüleri verilmiştir. Önerdiğimiz sistemin performansı tüm detaylar düşünülerek hesaplanmıştır.



Şekil 4.4. Diyabet ve kalp hastalığı için düğüm, süreç ve C programlama ekranı.

4.2.2. Node-Red veri akış platformunun geliştirilmesi

Node-Red veri akış platformu, benzetim senaryolarından gelen sağlık verilerinin UDP:9001 portundan alınarak Apache Kafka platformuna gönderilmesi amacıyla geliştirilmiştir. Veri akışı platformunu geliştirmeden önce Node-Red'i çalıştıran Node.js versiyon 14 uygulaması kurulmuştur. Ardından komut istemi ekranından `npm install -g --unsafe-perm node-red` komutu ile Node-Red kuruldu. Node-Red platformu http://10.10.40.100:1880 adresinde çalışmaktadır.



Şekil 4.5. Node-Red veri akış platformu uygulaması.

Yukarıda verilen Şekil 4.5.'te UDP:9001 portundan gerçek zamanlı olarak veriler çekilmiş ve JSON formatına çevrilerek Apache Kafka Producer nesnesi ile ilgili Apache Kafka konusuna veriler gönderilmiştir.

4.2.3. Apache Kafka veri akış platformunun geliştirilmesi

Apache Kafka veri akış platformu, Node-Red platformundaki Producer nesnesi üzerinden gönderilen sağlık verilerinin Apache Kafka'ya abone olan tüketiciler'e (consumer) iletmek için geliştirilmiştir. Bu bağlamda, Apache Kafka sunucusu (broker) üzerinde toplamda dört tane konu (topic) oluşturulmuştur. Apache Kafka veri akış platformunun geliştirme için gerekli aşamalar aşağıda belirtilmiştir.

- Apache Kafka 3.1 ve yönetimi için Apache Zookeeper 3.4.6 sürümleri dosyaları C: sürücüsü altında klasörlenmiştir.
- Ortam değişkenlerine aşağıdaki değişkenler eklenmiştir.

  Değişken adı: ZOOKEEPER_HOME, Değer="C:\zookeeper"

  Değişken adı: KAFKA_HOME, Değer= "C:\kafka"

  Değişken adı: Path, Değer= "%ZOOKEEPER_HOME%\bin"

  Değişken adı: Path, Değer= "%KAFKA_HOME%\bin\windows"

- Apache Kafka sunucusu üzerinde komut istemi ile ilgili dört konu oluşturulmuştur.

```
kafka-topics.bat --create --zookeeper localhost:2181
--replication-factor 1 --partitions 1 --topic DiyabetData

kafka-topics.bat --create --zookeeper localhost:2181
--replication-factor 1 --partitions 1 --topic KalpData

kafka-topics.bat --create --zookeeper localhost:2181
--replication-factor 1 --partitions 1 --topic DiyabetTahmin

kafka-topics.bat --create --zookeeper localhost:2181
--replication-factor 1 --partitions 1 --topic KalpTahmin
```

4.2.4. Apache Spark tabanlı veri analitiği bileşeninin geliştirilmesi

Önerilen IoMT çerçevesinde, Bulut bilişim katmanındaki diyabet ve kalp hastalığına ait tahminler bu bileşende yapılmıştır. Bu bileşende tahminler Apache Spark'ın Mllib kütüphanesine ait KA, RO, GA, LR ve DVM sınıflandırma algoritmaları kullanılarak gerçekleştirilmiştir.

Veri analitiği bileşeni Intellij IDEA kod yazım editörü üzerinde, Maven projesi olarak Java programlama dili kullanılarak geliştirilmiştir. Bu tahminlerin gerçek zamanlı olarak gerçekleştirilmesi için gerekli geliştirme aşamaları bu bölümde detaylı olarak açıklanmıştır.

4.2.4.1. *Eğitim Modelinin Oluşturulması*

Diyabet ve kalp hastalığının tahminlerinin gerçekleştirilebilmesi için öncelikle tahmin modellerinin geliştirilmesi gerekmektedir. Bu tahmin modelinin geliştirme aşamaları aşağıda verilmiştir.

Diyabet ve kalp hastalığı tahmin modelinde kullanılacak olan veri setleri Şekil 4.6.'da ve Şekil 4.7.'de gösterildiği gibi belirtilen adresten okunmuştur.

```
//Model eğitimi için diyabet hastalığına yönelik veriseti okunuyor.
Dataset<Row> dataSet = spark.read().format("csv")
        .option("sep", ",")
        .option("inferSchema", "true")
        .option("header", "true")
        .load( path: "D:\\Tez\\diabetes.csv");
```

Şekil 4.6. Diyabet eğitim veri setinin okunması.

```
//Model eğitimi için kalp rahatsızlığına yönelik veriseti okunuyor.
Dataset<Row> dataSet = spark.read().format("csv")
        .option("sep", ",")
        .option("inferSchema", "true")
        .option("header", "true")
        .load( path: "D:\\Tez\\heart-disease.csv");
```

Şekil 4.7. Kalp hastalığı eğitim veri setinin okunması.

Veri setlerinden okunan değerlerin sütunları Şekil 4.8. ve Şekil 4.9.'daki gibi ön işleme aşamasından geçmiştir. StringIndexer ile veriler sayısal hale dönüştürülmüştür. VectorAssembler ile vektör haline getirilmiş tek bir sütunda toplanarak analiz için hazır hale getirilmiştir.

```
//Tahminde hangi veriler kullanılacaksa o veriler toplanıp vektör haline getiriliyor.
VectorAssembler vectorAssembler = new VectorAssembler()
        .setInputCols(new String[]{"pregnancies","glucose","bloodpressure",
                "skinthickness","insulin","bmi","diabetespedigreefunction","age"})
        .setOutputCol("features")
        .setHandleInvalid("skip");
Dataset<Row> featureDF = vectorAssembler.transform(dataSet);

//Analiz değerlerine göre sonuç hangi sütun ise o sütun label hale getiriliyor.
StringIndexer stringIndexer = new StringIndexer()
        .setInputCol("outcome")
        .setOutputCol("label")
        .setHandleInvalid("skip");
Dataset<Row> labelDF= stringIndexer.fit(featureDF).transform(featureDF);
```

Şekil 4.8. Diyabet verilerinin ön işleme aşamasından geçmesi.

```
VectorAssembler assembler = new VectorAssembler()
        .setInputCols(new String[]{"age","ca","chol","cp","exang","fbs","oldpeak",
                "restecg","sex","slope","thal","thalach","trestbps"})
        .setOutputCol("features")
        .setHandleInvalid("skip");
Dataset<Row> featureDF = assembler.transform(dataSet);

//Analiz değerlerine göre sonuç hangi sütun ise o sütun label hale getiriliyor. analiz için gerekli
StringIndexer label = new StringIndexer()
        .setInputCol("target")
        .setOutputCol("label")
        .setHandleInvalid("skip");
Dataset<Row> labelDF= label.fit(featureDF).transform(featureDF);
```

Şekil 4.9. Kalp hastalığı verilerinin ön işleme aşamasından geçmesi.

Tahmin modelinde kullanılacak olan KA, RO, GA, LR ve DVM sınıflandırma algoritmalarının MLlib kütüphanesi kullanılarak belirli parametreler ışığında Şekil 4.10.'daki gibi modeller her hastalık tahmini için tanımlanmıştır.

```
DecisionTreeClassifier dt = new DecisionTreeClassifier()
        .setLabelCol("label")
        .setFeaturesCol("features")
        .setMaxBins(10)
        .setMaxDepth(3);

RandomForestClassifier rf = new RandomForestClassifier()
        .setLabelCol("label")
        .setFeaturesCol("features")
        .setMaxBins(10)
        .setMaxDepth(3)
        .setNumTrees(20)
        .setImpurity("gini");

GBTClassifier gbt = new GBTClassifier()
        .setLabelCol("label")
        .setFeaturesCol("features")
        .setMaxBins(10)
        .setMaxDepth(3)
        .setMaxIter(20)
        .setImpurity("gini");

LogisticRegression lr = new LogisticRegression()
        .setMaxIter(100)
        .setRegParam(0.001)
        .setElasticNetParam(1)
        .setStandardization(true);

LinearSVC svc = new LinearSVC()
        .setMaxIter(100)
        .setRegParam(0.001)
        .setStandardization(true);
```

Şekil 4.10. Sınıflandırma modellerinin tanımlanması.

Yukarıda tanımlanan işlemlerin tümü Şekil 4.11.'de gösterildiği gibi bir boru hattı (pipeline) kullanılarak verilere sınıflandırma modelleri uygulanmıştır.

```
Pipeline pipeline = new Pipeline()
        .setStages(new PipelineStage[]{label, assembler, dt, labelConverter});
```

Şekil 4.11. Eğitim modelinin oluşturulması için boru hattının (pipeline) uygulanması.

Diyabet ve kalp hastalığı tahmininde en doğru modelin oluşturulması için Şekil 4.12.'deki gösterildiği gibi 10 kat çapraz doğrulama uygulanmıştır.

```
MulticlassClassificationEvaluator evaluator = new MulticlassClassificationEvaluator()
        .setLabelCol("label")
        .setPredictionCol("prediction")
        .setMetricName("accuracy");

CrossValidator cv = new CrossValidator()
        .setEstimator(pipeline)
        .setEvaluator(evaluator)
        .setEstimatorParamMaps(paramGrid)
        .setNumFolds(10);
```

Şekil 4.12. En iyi eğitim modeli için 10 kat çapraz doğrulama uygulanması.

Son olarak, Şekil 4.13.'te okunan eğitim verilerine oluşturulan model uygulanmış ve eğitim modelinin son hali kaydedilmiştir.

```
CrossValidatorModel crossValidatorModel = cv.fit(dataSet);
try {
    crossValidatorModel.write().overwrite().save( path: "D:\\Tez\\egitim-model");
} catch (IOException e) {
    e.printStackTrace();
}
```

Şekil 4.13. Eğitim verilerine oluşturulan modelinin uygulanması ve kaydedilmesi.

4.2.4.2. *Gerçek Zamanlı Tahminlerin Yapılması*

Apache Kafka üzerinden alınan verilere gerçek zamanlı olarak analizler yapılarak hastalık tahminleri yapılmıştır. Bunun için ilk olarak, Şekil 4.14.'te gösterildiği gibi 10 sn aralıklar ile verilerin çekileceğine dair Apache Spark ve Spark Streaming ayarlamaları yapılmıştır.

```
//Spark konfigürasyonları yapılıyor.
SparkConf conf = new SparkConf().setAppName("Diabet").setMaster("spark://10.10.40.100:7077")
        .setJars(new String[] {"target/DiabetAnalysis-1.0-SNAPSHOT-driver.jar"});
JavaStreamingContext jssc = new JavaStreamingContext(conf, Durations.seconds(10));
```

Şekil 4.14. Apache Spark ve Spark Streaming ayarlarının yapılması.

Diyabet sağlık verilerinin çekileceği Apache Kafka ayarları Şekil 4.15.'teki gibi yapılarak, veriler gerçek zamanlı olarak ilgili Apache Kafka konusundan çekilmiştir. Kalp hastalığı içinde benzer işlemler yapılmıştır.

```
//Kafka Stream konfigürasyonları yapılıyor.
Map<String, Object> kafkaParams = new HashMap<>();
kafkaParams.put("bootstrap.servers", "localhost:9092");
kafkaParams.put("key.deserializer", StringDeserializer.class);
kafkaParams.put("value.deserializer", StringDeserializer.class);
kafkaParams.put("group.id", timestamp.toString());
kafkaParams.put("auto.offset.reset", "latest");
kafkaParams.put("enable.auto.commit", false);

//Abone olunacak Topic seçiliyor.
Collection<String> topics = Arrays.asList("DiyabetData");
```

Şekil 4.15. Apache Kafka ayarlarının yapılması.

Şekil 4.16.'da Apache Kafka konusundan veriler, gerçek zamanlı olarak alınmıştır.

```
JavaInputDStream<ConsumerRecord<String, String>> stream =
        KafkaUtils.createDirectStream(
                jssc,
                LocationStrategies.PreferConsistent(),
                ConsumerStrategies.<~>Subscribe(topics, kafkaParams)
        );
JavaDStream<String> lines = stream.map(ConsumerRecord::value);
```

Şekil 4.16. Apache Kafka konusundan verilerin alınması.

Diyabet ve kalp hastalığına ait veriler ön işlemeden geçerek analiz için uygun formata çevrilmiştir. Şekil 4.17.'de diyabet verileri için ön işleme aşamasının kodları verilmiştir. Kalp hastalığı için de aynı işlemler yapılmıştır.

```
StructType schema = DataTypes.createStructType(new StructField[]
        {DataTypes.createStructField( name: "value", DataTypes.StringType,  nullable: true)});
StructType schemaDF = new StructType()
        .add( name: "pregnancies", dataType: "string")
        .add( name: "glucose", dataType: "string")
        .add( name: "bloodpressure", dataType: "string")
        .add( name: "skinthickness", dataType: "string")
        .add( name: "insulin", dataType: "string")
        .add( name: "bmi", dataType: "string")
        .add( name: "diabetespedigreefunction", dataType: "string")
        .add( name: "age", dataType: "string");
SparkSession spark = JavaSparkSessionSingleton.getInstance(rdd.context().getConf());
Dataset<Row> msgDataFrame = spark.createDataFrame(rowRDD, schema);
Dataset<DiabetDataModel> data = msgDataFrame.selectExpr("CAST(value AS STRING) as veri")
        .select(functions.from_json(functions.col( colName: "veri"),schemaDF)
                .as("json")).select( col: "json.*")
        .as(Encoders.bean(DiabetDataModel.class));
Dataset<Row> dataframe = data.toDF();
Dataset<Row> testData = dataframe.selectExpr("cast(pregnancies as int) pregnancies",
        "cast(glucose as int) glucose","cast(bloodpressure as int) bloodpressure",
        "cast(skinthickness as int) skinthickness","cast(insulin as int) insulin",
        "cast(bmi as double) bmi","cast(diabetespedigreefunction as double) diabetespedigreefunction",
        "cast(age as int) age");
```

Şekil 4.17. Gerçek zamanlı alınan diyabet sağlık verilerinin ön işlemeden geçmesi.

Veri ön işlemesi aşamasından sonra Şekil 4.18.'de gösterildiği gibi daha önce kaydedilen eğitim modeli yüklenmiş ve test verileri üzerinden hastalık tahminleri yapılmıştır.

```
CrossValidatorModel crossValidatorModel = CrossValidatorModel
        .read().load( path: "D:\\Tez\\egitim-model");
Dataset<Row> tahminData = crossValidatorModel.transform(testData);
```

Şekil 4.18. Hastalık tahminlerinin yüklenen eğitim modeli üzerinde yapılması.

Son olarak gerçek zamanlı olarak yapılan veriler canlı gösterge panelinde gösterilmesi için Apache Kafka konusuna yüklenmiştir. Ayrıca, daha sonra kullanılabilmesi için MongoDB veri tabanında saklanmıştır. Şekil 4.19., diyabet tahminlerinin sonuçları için kullanılan kodlardır.

```
if (tahminData.count()>0){
    Dataset<Row> dataset2 = tahminData.withColumn( colName: "value", functions.concat(
            tahminData.col( colName: "prediction"),lit( literal: ";"),
            tahminData.col( colName: "pregnancies"), lit( literal: ";"),tahminData.col( colName: "glucose"),
            lit( literal: ";"),tahminData.col( colName: "bloodpressure"), lit( literal: ";"),
            tahminData.col( colName: "skinthickness"), lit( literal: ";"),tahminData.col( colName: "insulin"),
            lit( literal: ";"),tahminData.col( colName: "bmi"), lit( literal: ";"),
            tahminData.col( colName: "diabetespedigreefunction"), lit( literal: ";"),tahminData.col( colName: "age"),
            lit( literal: ";"),tahminData.col( colName: "probability").cast("String")));
    Dataset<Row> predictDF= dataset2.select( col: "value");

    predictDF.write()
            .format("kafka")
            .option("kafka.bootstrap.servers", ":9092")
            .option("topic", "DiyabetTahmin")
            .option("checkpointLocation", "checkpointLocation-kafka2console")
            .save();
    WriteMongo writeMongo = new WriteMongo();
    writeMongo.write( topic: "DiyabetTahmin",  database: "Tez",  collection: "DiyabetTahmin",  key: "data");

}
```

Şekil 4.19. Hastalık tahmin sonuçlarının Apache Kafka ve MongoDB veri tabanına aktarılması.

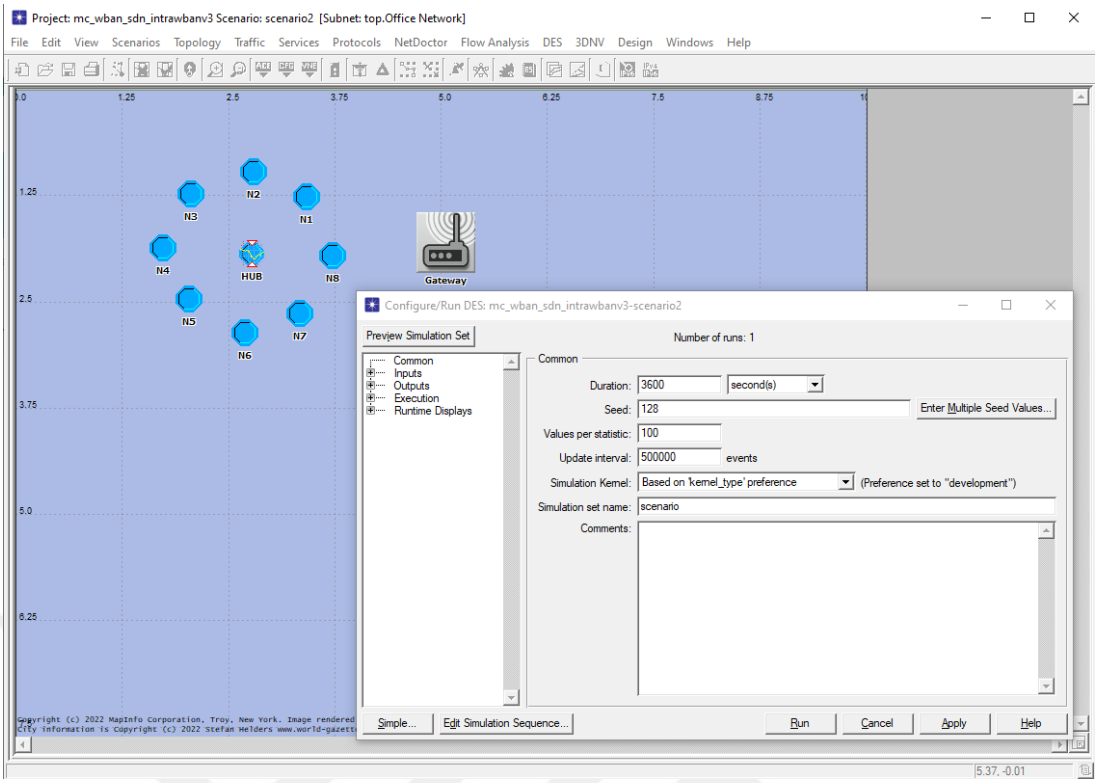
4.3. UYGULAMANIN GERÇEKLEŞTİRİLMESİ

Riverbed Modeler'da Şekil 4.2. ve Şekil 4.3.'te verilen iki farklı benzetim senaryosu hazırlanmıştır. İlk senaryoda, farklı önceliklere ve farklı paket geliş zamanlarına sahip sekiz farklı sensör düğümü (N1-N8), paketlerini CSMA/CA tabanlı IEEE 802.15.6 protokolü ile HUB'a gönderir ve ardından HUB, topladığı verileri ağ geçidine gönderir. Bu paketler ağ geçidindeki soket programlama yardımı ile dış ortama aktarılır. İkinci senaryoda sensör sayısı (N1-N13) artırılmıştır. Benzetim senaryosu parametreleri Çizelge 4.2.'de verilmiştir. Senaryomuzdaki her sensör, ilgili hastalığı tespit etmek için makine öğrenmesi algoritmalarında kullanılan veri setinin (8 ve 13 girişli) bir giriş parametresini simgelemektedir. Bu şekilde bir hastaya entegre edilmiş çeşitli sensörler tarafından ölçülen sinyaller HUB'a gönderilir, ardından HUB bu verileri önceliklerine göre sıralar ve ağ geçidine gönderir. Böylece bir kişi üzerinde ölçülen çeşitli sinyaller, uzak noktalarda Bulut bilişim gibi teknolojiler yardımıyla bir sonuca, tahmine veya teşhise dönüşebilmektedir.

Çizelge 4.2. Benzetim senaryosu parametreleri.

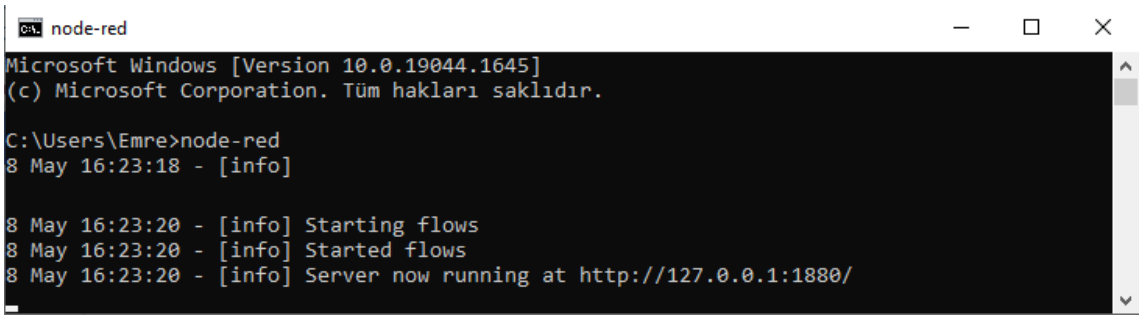
Parametre	Değer	
Benzetim Senaryosu Zamanı	3600 saniye	
Frekans	2400 to 2483.5 GHz	
Düğüm Sayısı	8 – 13 sensör	
MAC Protokolü	CSMA/CA tabanlı IEEE 802.15.6	
Çekişme Pencereleri	1-64 Alternatif İkili Üstel Geri Çekilme	
Öncelikler	0-7 (düşükten yükseğe)	
Bant Genişliği	1 MHz	
Veri Hızı	971.4 kbps	
Paket Boyutu	100 Byte	
Paketler arası varış süreleri *(Üstel dağılım işlevi kullanılarak oluşturuldu)*		
Senaryo 1 (Diyabet)	N8 = 0.2 s N7 = 0.333 s N6 = 0.5 s N5 = 1 s	N4 = 2 s N3 = 3 s N2 = 4 s N1 = 5 s
Senaryo 2 (Kalp Hastalığı)	N11=N12= N13= 0.09 s N9=N10= 0.16 s N7=N8= 0.25 s	N5=N6= 0.33 s N3=N4= 0.2 s N1=N2= 1 s

Diyabet ve kalp hastalığının tahmininde kullanılacak olan gerçek zamanlı sağlık verilerinin üretilmesi için Riverbed benzetim ortamında ilgili hastalık KVAA senaryosu Çizelge 4.2.'deki parametrelere göre ayarlanarak çalıştırılmıştır. Diyabet hastalığı benzetim senaryosunun çalıştırılması aşaması Şekil 4.20.'de gösterilmiştir.



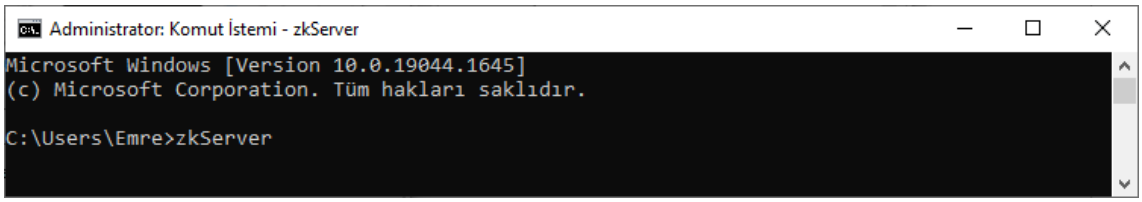
Şekil 4.20. Diyabet benzetim senaryosu çalıştırma aşaması.

Benzetim senaryolarından gönderilen sağlık verileri Bulut bilişimde makine öğrenmesi algoritmaları ile analiz edilerek tahminler gerçekleştirilir ve depolanır. Bu katmanda ilk olarak Node-Red veri akış platformu aracılığı ile UDP 9001 portundan alınır ve Apache Kafka'ya gönderilir. Şekil 4.21.'deki konsol ekranında Node-Red veri akış platformu 1880 portu üzerinden çalışması sağlanmakta ve platform süreci takip edilir.



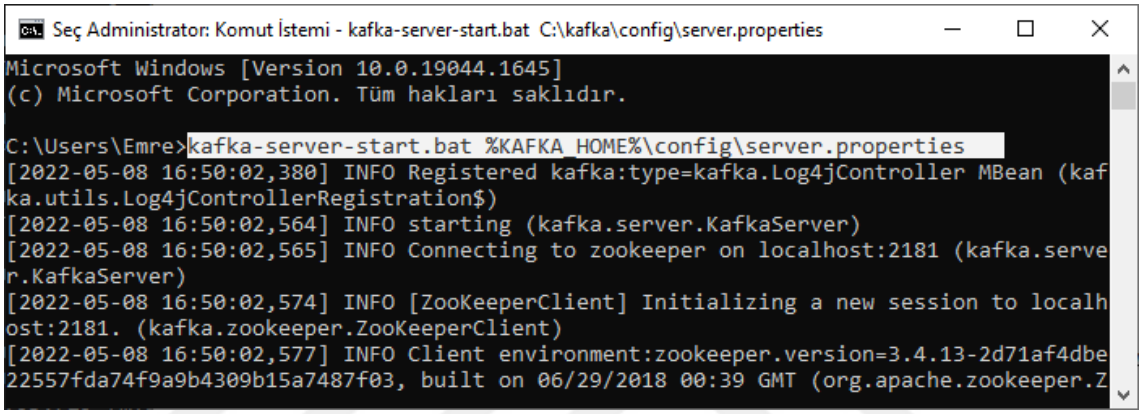
Şekil 4.21. Node-Red veri akış platformunun başlatılması ve takibi için konsol ekranı.

Bulut bilişim katmanında Node-Red'ten sonra ikinci bir veri akış platformu olan Apache Kafka çalıştırılır. Öncelikle, Apache Kafka'nın çalışmasını ve yönetimini sağlayan Apache Zookeeper çalıştırılmıştır. Şekil 4.22.'deki konsol ekranında Apache Zookeeper başlatılması gösterilmiştir.



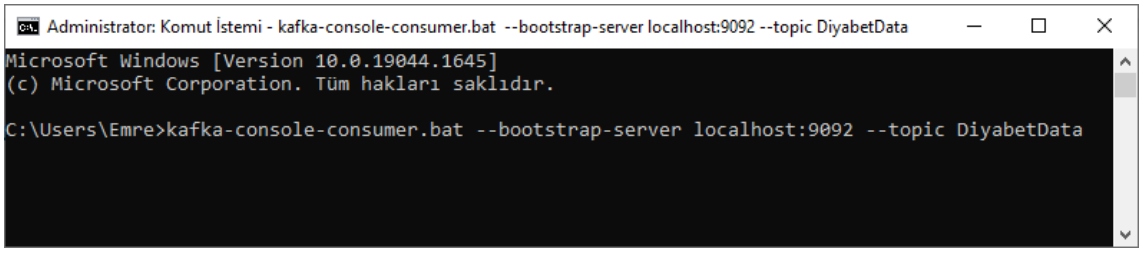
Şekil 4.22. Apache Zookeeper başlatılması.

Apache Zookeeper düzgün bir şekilde başlatıldıktan sonra Apache Kafka veri akış platformu çalıştırılmıştır. Şekil 4.23.'te Apache Kafka'nın konsol ekranında başlatılması gösterilmiştir.



Şekil 4.23. Apache Kafka'nın çalıştırılması.

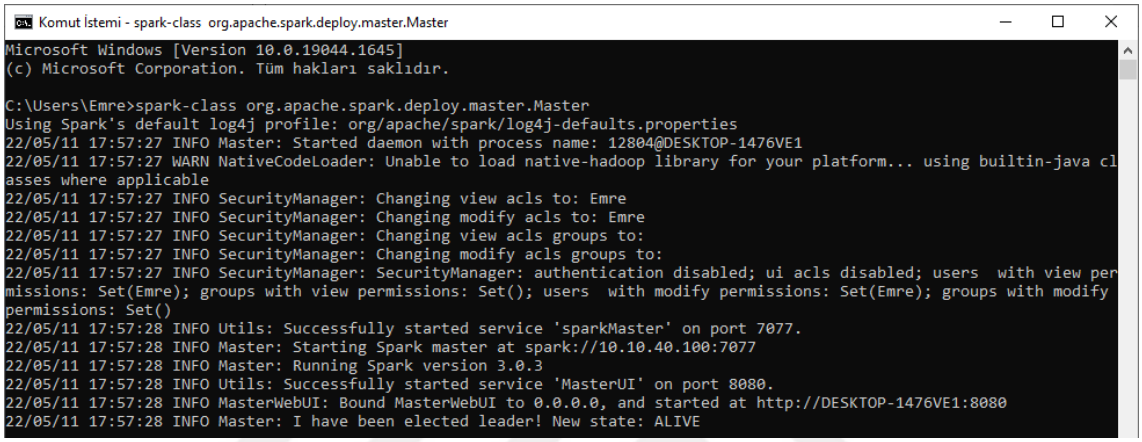
Apache Kafka konusuna gelen verileri istendiği takdirde konsol ekranından ilgili Apache Kafka konusu çalıştırılarak takip edilebilmektedir. Şekil 4.24.'te diyabet konusuna gelen sağlık verileri konsol ekranından takip edilmiştir.



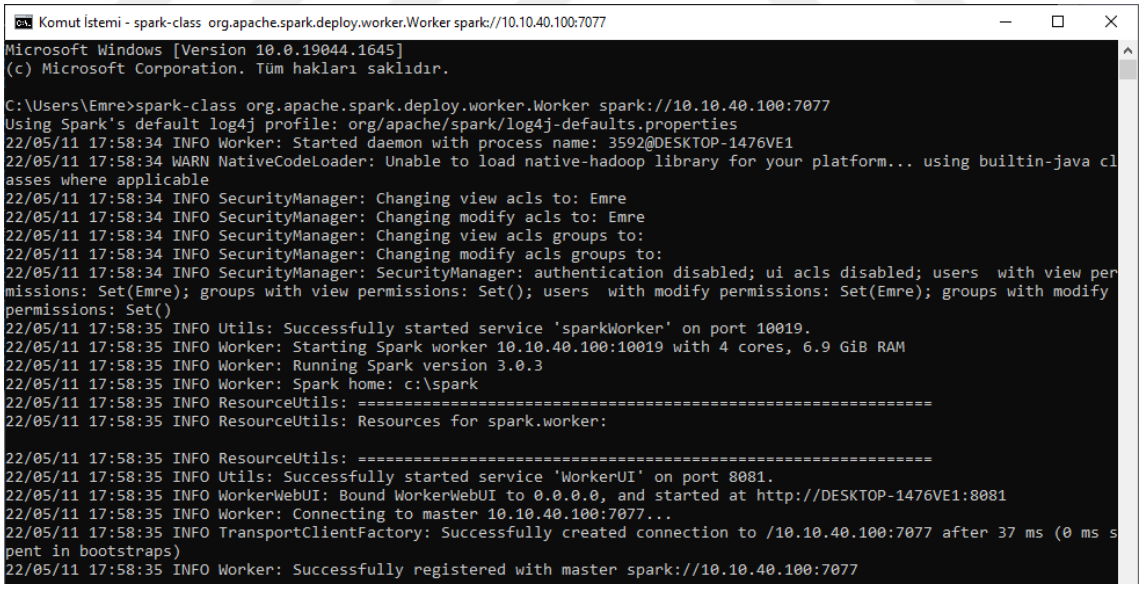
Şekil 4.24. DiyabetData Apache Kafka konusuna gelen veriler için konsol ekranı.

Diyabet ve kalp hastalığı tahminleri Bulut bilişimde Apache Spark tarafından gerçekleştirilmiştir. Öncelikle tahminlerin dağıtık olarak gerçekleşebilmesi için Yönetici ve İşçi düğümler aktif edilmiştir. Yönetici ve İşçi düğümlerin çalıştırılması Şekil 4.25. ve Şekil 4.26.'da gösterilmiştir. Ardından tahminlerin gerçek zamanlı olarak gerçekleştirilmesi için Intellij IDEA kod editörü üzerinde geliştirilen Maven projesindeki

DiabetKafkaSparkStreaming.java ve HeartDiseaseKafkaSparkStreaming.java sınıfı çalıştırılmıştır. Bu sınıflarda Apache Kafka'daki ilgili hastalık konusuna abone olunarak KVAA sağlık sensör verileri alınarak tahminler gerçekleştirilmiştir. Ayrıca, hastalık tahmin sonuçlarının görselleştirilmesi için kullanılan Canlı Web Panel'de hastalık tahminlerinin gösterilebilmesi için Apache Kafka'daki ilgili hastalık konusuna sonuçlar gönderilmiştir. Bununla birlikte hastalık tahmin sonuçları MongoDB veri tabanı yönetim sisteminde oluşturulan Tez veri tabanındaki ilgili tabloya sonuçlar kaydedilmiştir.



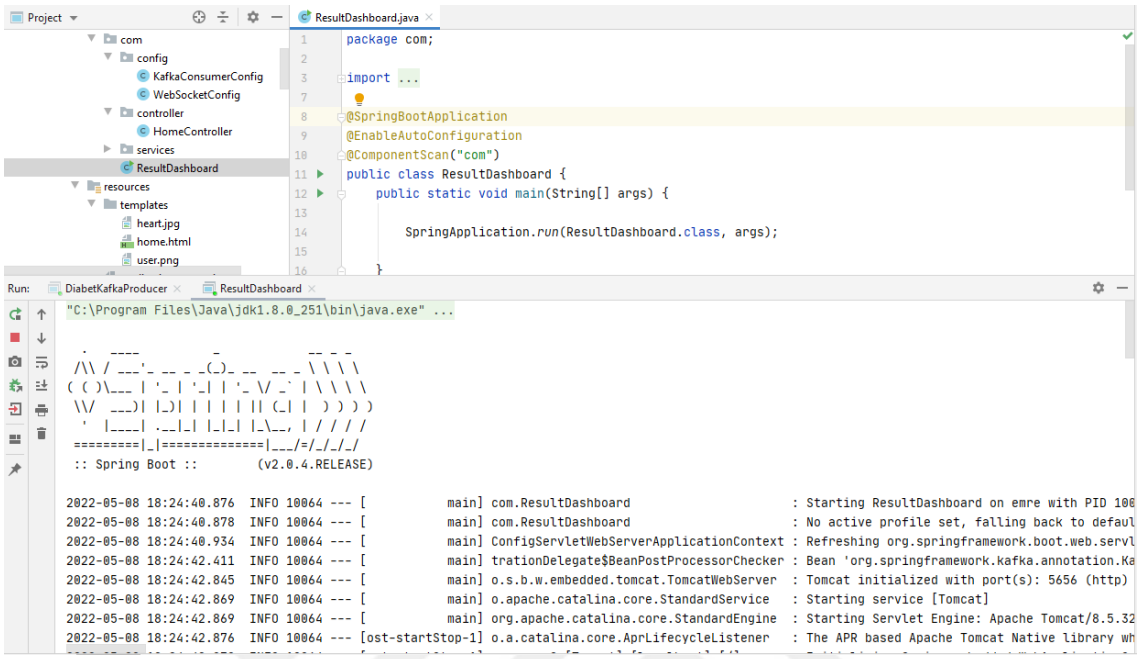
Şekil 4.25. Yönetici düğümün çalıştırılması.



Şekil 4.26. İşçi düğümlerin çalıştırılması.

Son olarak Görselleştirme bileşeni olan Canlı Web Panel çalıştırılmıştır. Canlı web panelin aktif hale gelebilmesi için öncelikle Maven projesi içerisindeki ResultDashboard.java sınıfı çalıştırılmıştır. Burada, Apache Spark tarafından Apache Kafka konusuna gönderilen hastalık tahmin sonuçları alınarak Spring Boot ve Web

Socket entegrasyonu ile Canlı Web Panel'de gerçek zamanlı olarak sunulmuştur. Şekil 4.27.'de Canlı Web Panel için ResultDashboard.java sınıfı çalıştırılmıştır. Canlı Web Panel 5656 portu üzerinden çalışmaktadır.



Şekil 4.27. Canlı Web Panel'in çalıştırılması.

5. BULGULAR VE TARTIŞMA

Bu bölümde, Önerilen IoMT çerçevesi üzerinde yapılan deneysel sonuçlar açıklanmıştır. Önerilen IoMT çerçevesi, diyabet ve kalp hastalığının gerçek zamanlı olarak tahmin edilmesi için geliştirilmiştir. Önerilen IoMT çerçevesi, veri kaynağı katmanı, Sis bilişim ve Bulut bilişim olmak üzere üç katmandan oluşmaktadır. İlk katman olan veri kaynağı katmanında, diyabet ve kalp hastalığına yönelik sağlık verileri Riverbed Modeler benzetim programı üzerinde oluşturulan KVAA senaryolarında gerçek zamanlı olarak üretilmiştir. İkinci katman Sis bilişimde, benzetim senaryolarında üretilen sağlık verileri kullanılarak bulanık mantık tahmin sistemi ile gerçek zamanlı olarak hastalık tahmini yapılmıştır. Üçüncü katman Bulut bilişim katmanında ise öncelikle benzetim senaryolarından üretilen sağlık verileri, veri akışı bileşenleri (Node-Red ve Apache Kafka) yardımı ile alınmıştır. Gerçek zamanlı sağlık verileri makine öğrenmesi algoritmalarının (MLlib) kullanıldığı Apache Spark tabanlı veri analitiği katmanına aktarılmıştır. Apache Spark veri analitiği katmanında, hastalık tahminleri için KA, RO, GA, LR ve DVM sınıflandırma algoritmaları kullanılarak bir tahmin modeli geliştirilmiştir. Ardından, Apache Spark'ın Streaming özelliği ile veriler gerçek zamanlı olarak alınmış ve geliştirilen tahmin modeli ile hastalık tahminleri yapılmıştır. Hastalık tahminleri Apache Spark üzerinde dağıtık hesaplama mimarisi ile gerçekleştirilmiştir. Ayrıca, görselleştirme ve depolama bileşeninde, benzetim senaryolarından gönderilen sağlık verileri ve hastalık tahmin sonuçları Apache Kafka üzerinden Canlı Web Panel'de gerçek zamanlı olarak sunulmuştur. Bununla birlikte, bu veriler MongoDB NoSQL veri tabanı yönetim sisteminde oluşturulan veri tabanında daha sonra kullanılmak üzere depo edilmiştir.

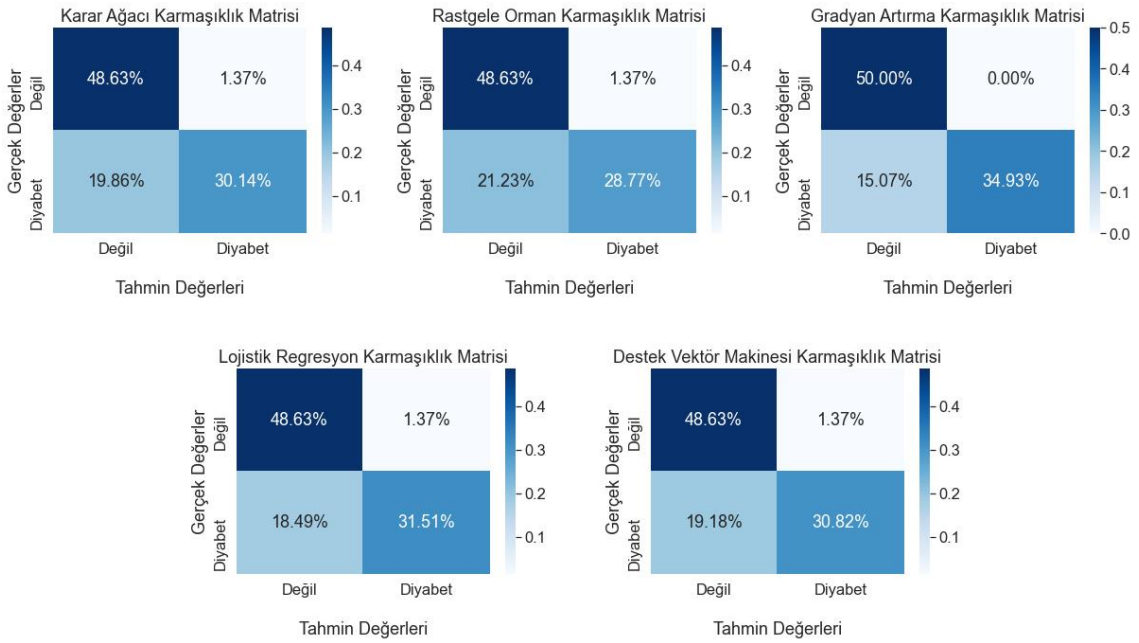
Bu bölümde, önerilen IoMT çerçevesinin çalışma performansında elde edilen bulgular, kalp ve diyabet hastalığı için ayrı başlıklar halinde sunulmaktadır.

5.1. SENARYO 1: DİYABET HASTALIĞINA AİT BULGULAR

Bu bölümde, ilk benzetim senaryosu olan diyabet hastalığına ait performans sonuçları incelenmiş ve aşağıdaki bulgular elde edilmiştir.

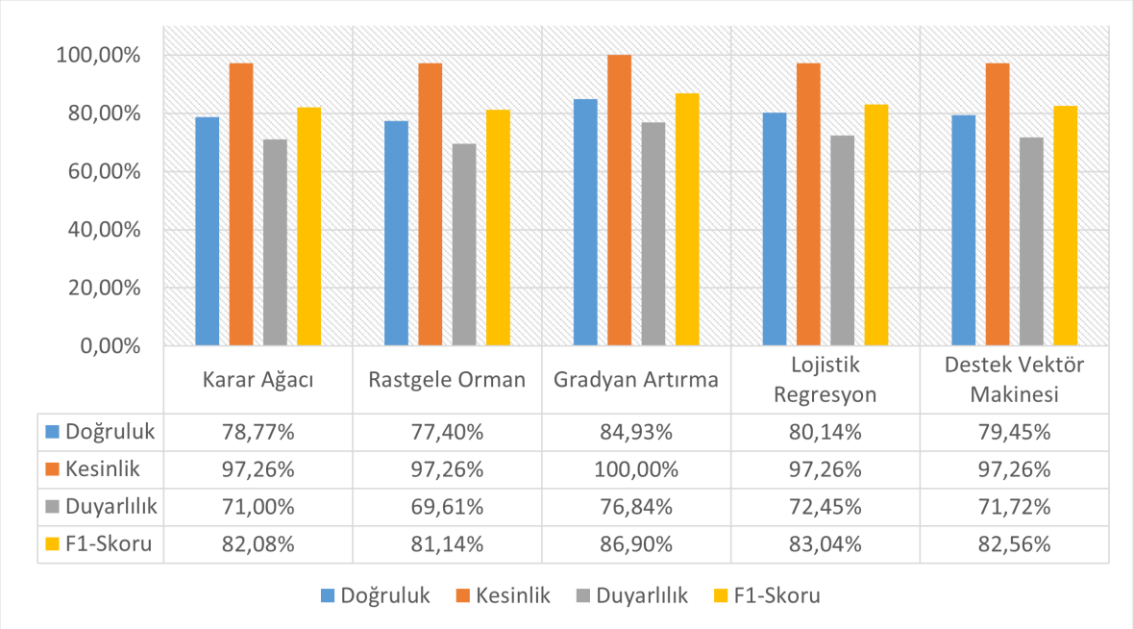
Önerilen IoMT çerçevesinin Sis bilişim katmanında bulanık mantık sistemi çalışmıştır. Yukarıda belirtilen özelliklere sahip önerilen bulanık mantık tabanlı diyabet tahmin sistemi, diyabet hastalıklarını %64 olasılıkla tahminler yapabildiği görülmüştür.

Karmaşıklık matrisi, verilerdeki gerçek sonuçlara dayalı olarak sınıflandırma modelleri tarafından yapılan doğru ve yanlış tahminlerin sayısını gösterir. Karmaşıklık matrisi, modellerin performans değerlendirmesinde en çok kullanılan metriktir. Şekil 5.1.'de KA, RO, GA, LR ve DVM sınıflandırma algoritmalarının tahmininden elde edilen karmaşıklık matrisi değerleri gösterilmiştir. Karmaşıklık matrisinde verilen DP, DN, YN ve YP değerleri kullanılarak KA, RO, GA, LR ve DVM sınıflandırma algoritmasının performans değerleri (doğruluk, kesinlik, duyarlılık, F1-skor) hesaplanmıştır. Sınıflandırma algoritmalarının karmaşıklık değerleri incelendiğinde GA algoritmasının diyabet hastası olan ve olmayan kişileri daha iyi tahmin edebildiği görülmüştür. RF algoritması ise diyabet hastası olan kişileri diğer algoritmalara göre daha düşük tahminler yapmıştır.



Şekil 5.1. Diyabet hastalığı için sınıflandırma algoritmalarının karmaşıklık matrisi değerleri.

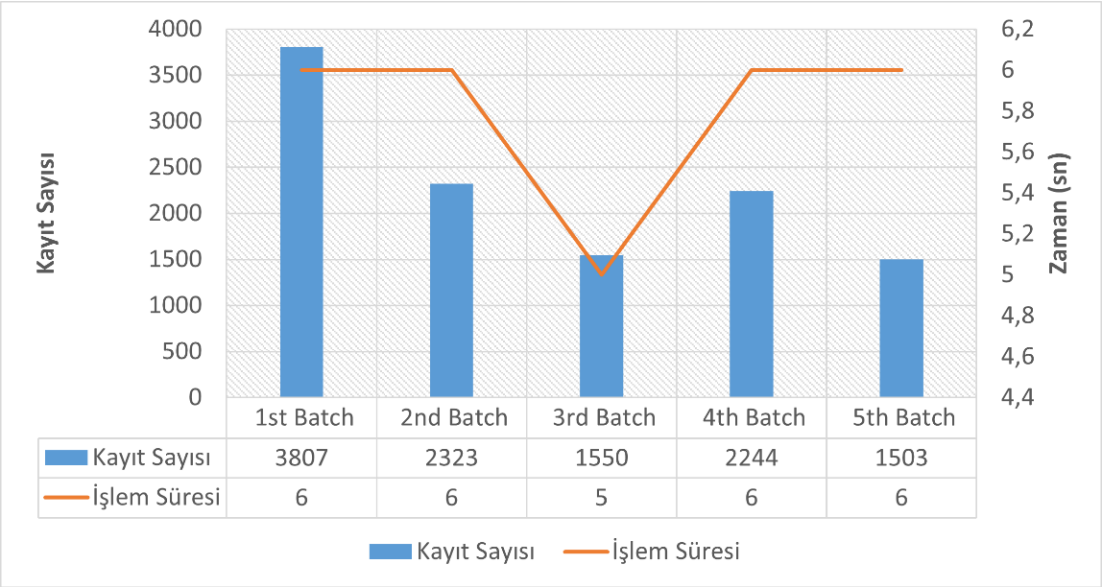
Diyabet hastalığı tahmini için 8 parametre değerlendirilmiştir. Geliştirilen model tarafından yapılan tahminler sonucunda sınıflandırma algoritmalarının doğruluk, kesinlik, duyarlılık ve F1-skor değerlerine göre performansları Şekil 5.2.'de gösterilmiştir.



	Karar Ağacı	Rastgele Orman	Gradyan Artırma	Lojistik Regresyon	Destek Vektör Makinesi
Doğruluk	78,77%	77,40%	84,93%	80,14%	79,45%
Kesinlik	97,26%	97,26%	100,00%	97,26%	97,26%
Duyarlılık	71,00%	69,61%	76,84%	72,45%	71,72%
F1-Skoru	82,08%	81,14%	86,90%	83,04%	82,56%

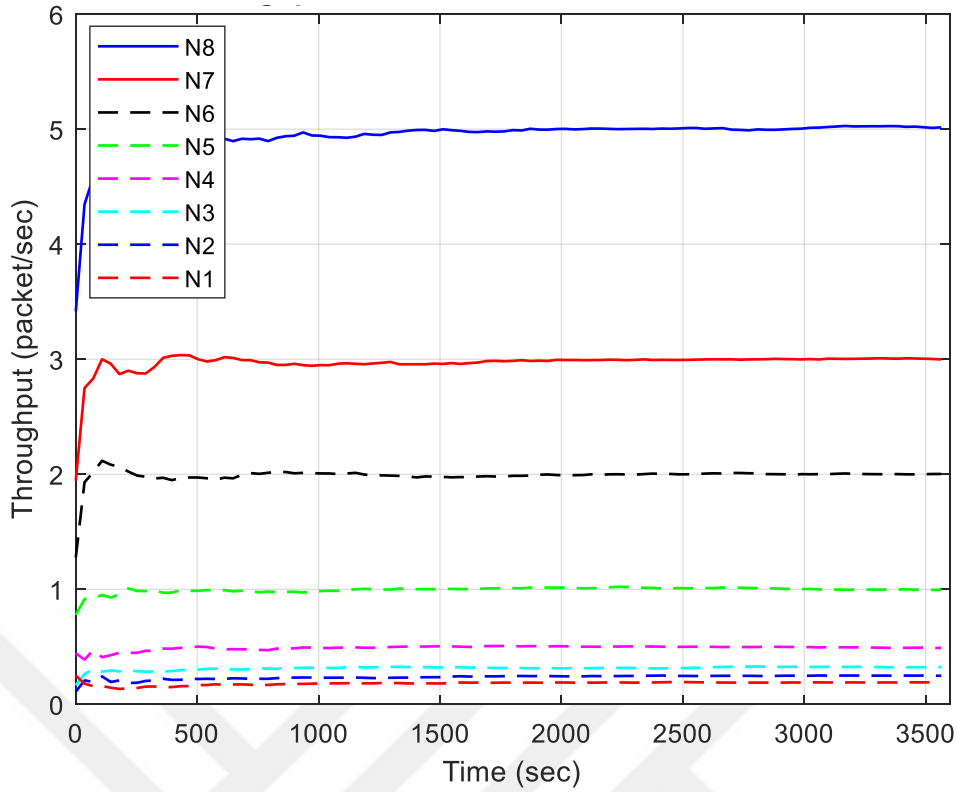
Şekil 5.2. Diyabet hastalığı için sınıflandırma algoritmalarının performans değerleri.

Şekil 5.2. incelendiğinde makine öğrenmesi sınıflandırma algoritmalarının performans sonuçları diyabet hastalığı için farklılık göstermiştir. Diyabet hastalığı tahmin performansları incelendiğinde, GA algoritmasının %84,93 doğruluk, %100,00 kesinlik, %76,84 duyarlılık ve %86,90 F1-skor değerleri ile en iyi tahmin performansına sahip olmuştur. Bununla birlikte, LR algoritması %80,14 doğruluk, %97,26 kesinlik, %72,45 duyarlılık ve %83,04 F1-skor değeri ile ikinci en iyi performansa sahip olmuştur. RF algoritması ise %77,40 doğruluk, %69,61 duyarlılık ve %81,14 F1-skor değerleri ile en düşük tahmin performansına sahip olmuştur. Bu bağlamda, GA sınıflandırma algoritması, diyabet hastalığı tahmininde en iyi performansı göstermiştir.

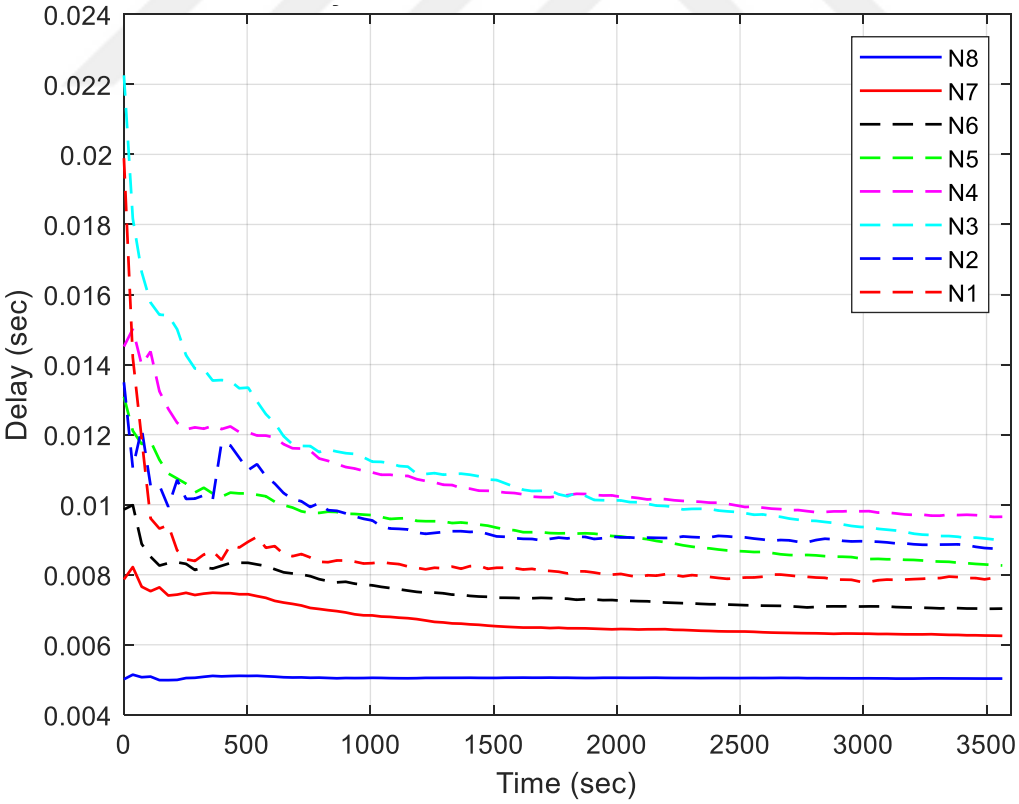


Şekil 5.3. Diyabet hastalığı için Apache Spark tarafından analiz edilen kayıtların sayısı ve işlem süreleri.

Diyabet hastalığı senaryosunda üretilen ve hastalık teşhisi için gönderilen veriler gerçek zamanlı olarak Apache Spark tahmin sistemi tarafından işlenmiştir. Diyabet hastalığı senaryosu için Apache Spark tarafından gerçek zamanlı olarak analiz edilen kayıt sayısı ve işlem süreleri Şekil 5.3.'te gösterilmiştir. Toplamda 5 adet toplu veri işleme gerçekleştirilmiştir. Bu toplu veri işlemelerin her birisi için toplam kayıt sayısı ve bu kayıtların işlenmesi için geçen süre verilmiştir. Apache Spark'ın zamana bağlı olarak işlediği kayıt sayısı incelendiğinde ortalama 6 saniye içerisinde gelen veriler işlenmiştir. Elde edilen ortalama veri işleme süreci, önerilen IoMT çerçevesinin gerçek zamanlı büyük verilerin işlenmesinde etkili olduğunu göstermiştir.



Şekil 5.4. Diyabet hastalığı benzetim senaryosu için iş çıkarım sonuçları.



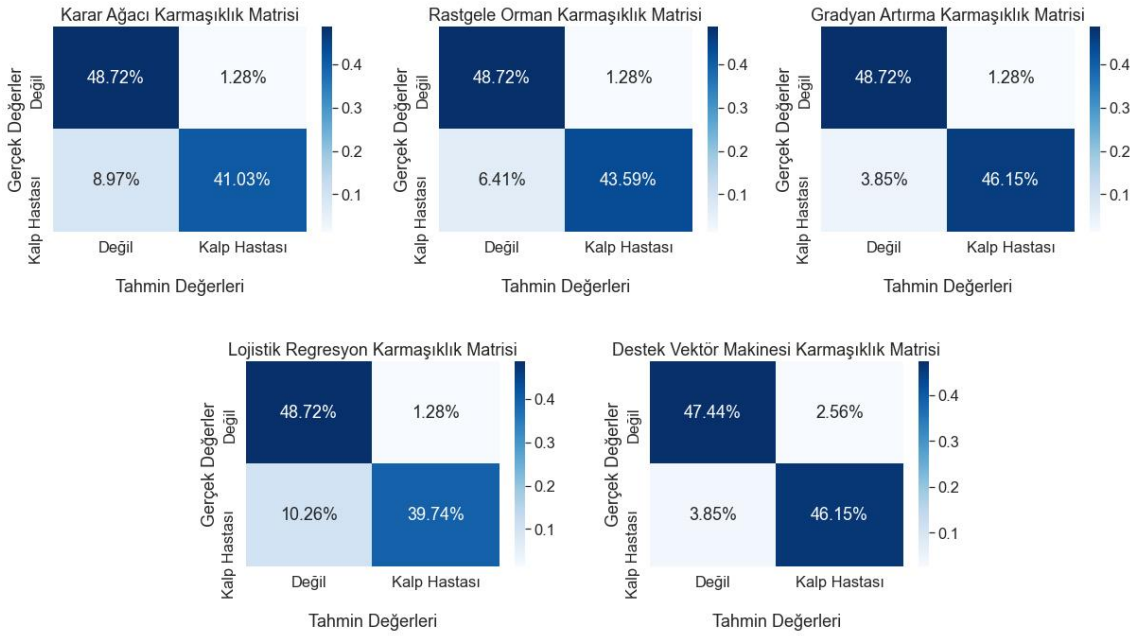
Şekil 5.5. Diyabet hastalığı benzetim senaryosu için gecikme sonuçları.

Şekil 5.4.'te diyabet hastalığı senaryonun iş çıkarım sonuçları verilmiştir. N1-N8, diyabet teşhisi için insan vücuduna yerleştirilen sensörlerdir. Bu sensörlerin diyabet teşhisi için çeşitli görevleri, öncelikleri ve paketler arası varış süreleri vardır. Sırasıyla N8 en yüksek önceliğe ve N1 en düşük önceliğe sahiptir. Öncelikler ve kablosuz iletişim yapısı arasındaki farklar, IEEE 802.15.6 standardında [76], [81] ayrıntılı olarak açıklanmıştır. Çekişme pencereleri için kullanılan aralıklar Çizelge 3.1.'de verilmiştir. Farklı paket varışları arasındaki sürelere göre elde edilen sonuçlar, tüm paketlerin ağ geçidine başarıyla ulaştığını göstermiştir. Şekil 5.5.'te, diyabet benzetim senaryosunun gecikme sonuçları gösterilmiştir. Elde edilen gecikme sonuçlarındaki farklılıklar, sensörlere atanan farklı önceliklerden ve paket varış sürelerinden kaynaklanmıştır. En yüksek öncelikli (N8) sensörün en düşük gecikmeye sahip olduğu ve öncelik sırası düştükçe sensörlerin daha yüksek gecikmelere sahip olduğu görülmüştür. Diyabet tanısında daha önemli olan ve paket geliş süreleri daha yüksek olan girdi değerleri, hedef düğüme daha hızlı ulaştırılmıştır.

5.2. SENARYO 2: KALP HASTALIĞINA AİT BULGULAR

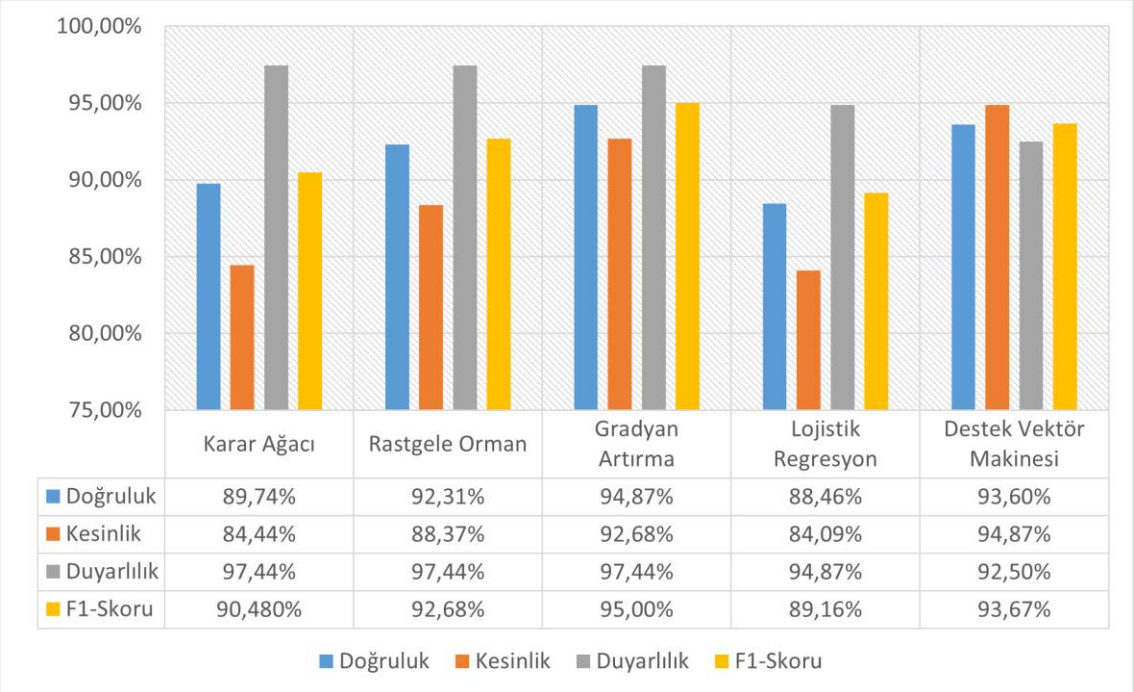
Bu bölümde, ikinci benzetim senaryosu olan kalp hastalığına ait performans sonuçları incelenmiş ve aşağıdaki bulgular elde edilmiştir.

Şekil 5.6.'da kalp hastalığı tahmininde kullanılan KA, RO, GB, LR ve DVM sınıflandırma algoritmalarının karmaşıklık matrisi değerleri gösterilmiştir. Karmaşıklık matrisi değerleri incelendiğinde GA algoritmasının kalp hastası olan kişileri daha iyi tahmin edebildiği görülmüştür. LR algoritması ise kalp hastası olan kişileri diğer algoritmalara göre daha düşük tahmin ettiği görülmüştür.



Şekil 5.6. Kalp hastalığı için sınıflandırma algoritmalarının karmaşıklık matris değerleri.

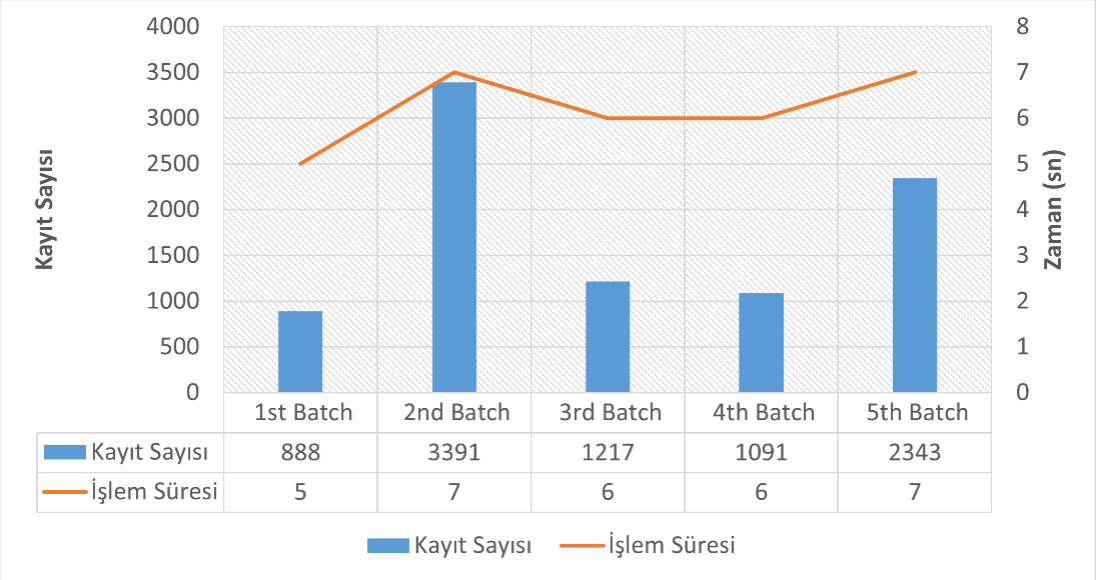
Kalp hastalığı tahmini için 13 parametre değerlendirilmiştir. Geliştirilen model tarafından yapılan tahminler sonucunda sınıflandırma algoritmalarının doğruluk, kesinlik, duyarlılık ve F1-skor değerlerine göre performansları Şekil 5.7.'de gösterilmiştir.



	Karar Ağacı	Rastgele Orman	Gradyan Artırma	Lojistik Regresyon	Destek Vektör Makinesi
Doğruluk	89,74%	92,31%	94,87%	88,46%	93,60%
Kesinlik	84,44%	88,37%	92,68%	84,09%	94,87%
Duyarlılık	97,44%	97,44%	97,44%	94,87%	92,50%
F1-Skoru	90,480%	92,68%	95,00%	89,16%	93,67%

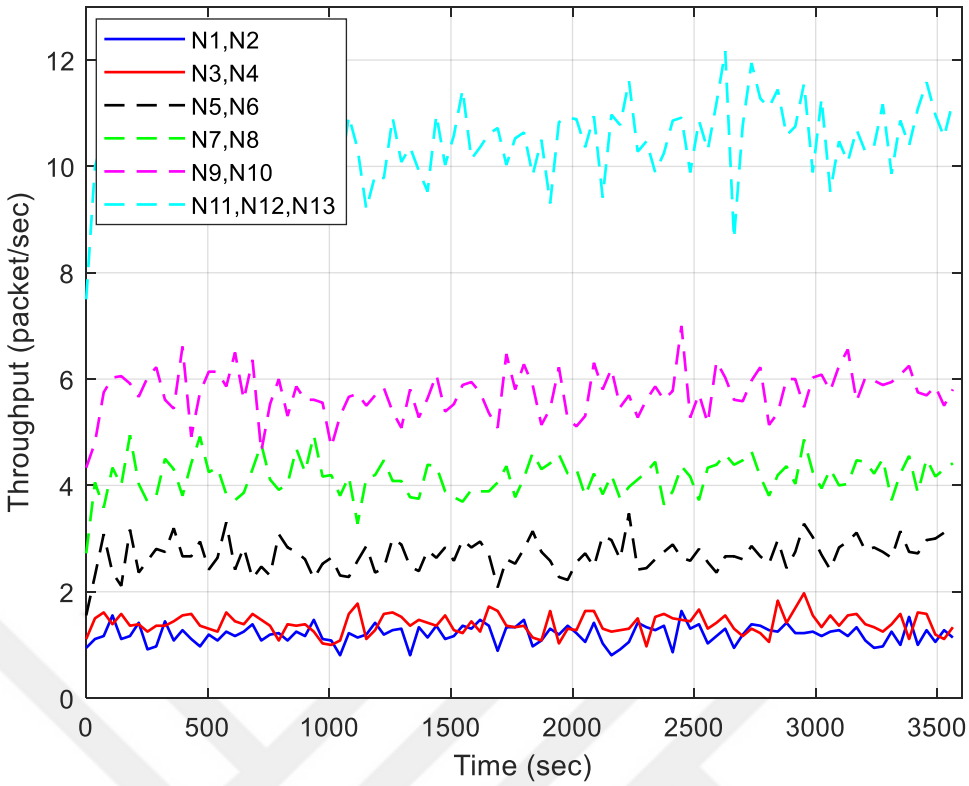
Şekil 5.7. Kalp hastalığı için sınıflandırma algoritmalarının performans değerleri.

Şekil 5.7. incelendiğinde makine öğrenmesi sınıflandırma algoritmalarının performans sonuçları gösterilmiştir. Kalp hastalığı tahminlerinde, GA sınıflandırma algoritması %94,87 doğruluk, %97,44 duyarlılık ve %95 F1-skoru ile en iyi performansa sahip olmuştur. Ayrıca, DVM algoritması %93,60 doğruluk ve %93,67 F1-skor ile en iyi ikinci tahmin değerlerine ve en iyi kesinlik (%94,87) değerine sahip olmuştur. Kalp hastalığı tahmininde en düşük tahmin performans ise en düşük doğruluk (%88,46), kesinlik (%84,09) ve F1-skor (%89,16) değerleri ile LR algoritmasına aittir.

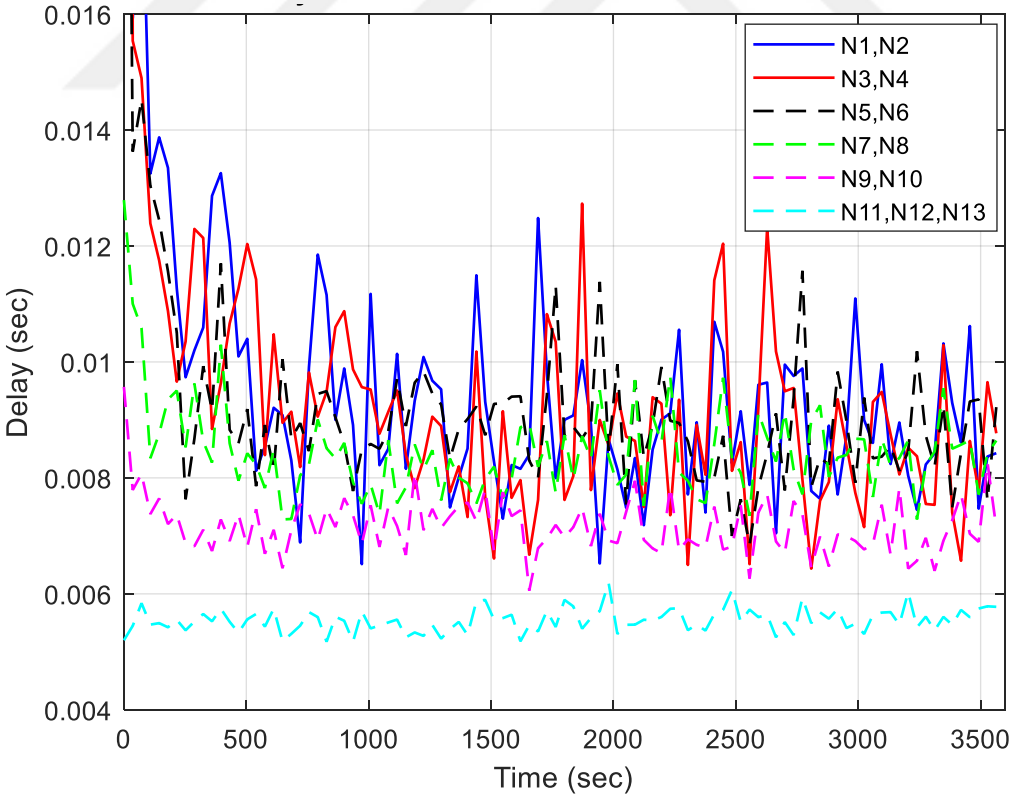


Şekil 5.8. Kalp hastalığı için Apache Spark tarafından analiz edilen kayıtların sayısı ve işlem süreleri.

Kalp hastalığı senaryosunda üretilen ve hastalık teşhisi için gönderilen veriler gerçek zamanlı olarak Apache Spark tahmin sistemi tarafından işlenmiştir. Kalp hastalığı senaryosu için Apache Spark tarafından gerçek zamanlı olarak analiz edilen kayıt sayısı ve işlem süreleri Şekil 5.8.'de gösterilmiştir. Toplamda 5 adet toplu veri işleme gerçekleştirilmiştir. Bu toplu veri işlemelerin her birisi için toplam kayıt sayısı ve bu kayıtların işlenmesi için geçen süre verilmiştir. Apache Spark'ın zamana bağlı olarak işlediği kayıt sayısı incelendiğinde ortalama 6 saniye içerisinde gelen veriler işlenmiştir. Elde edilen ortalama veri işleme süreci, önerilen IoMT çerçevesinin gerçek zamanlı büyük verilerin işlenmesinde etkili olduğunu göstermiştir.



Şekil 5.9. Kalp hastalığı KVAA senaryosu için iş çıkarım sonuçları.



Şekil 5.10. Kalp hastalığı KVAA senaryosu için gecikme sonuçları.

Kalp hastalığı senaryosu için elde edilen iş çıkarım sonuçları Şekil 5.9.'da ve gecikme sonuçları Şekil 5.10.'da gösterilmiştir. Bu senaryoda kalp hastalığını teşhis etmek veya mevcut durumu ortaya çıkarmak için 13 farklı veri kullanılmıştır. Bu veriler, makine öğrenmesinde kullanılan veri setinden elde edilen girdi kalp hastalığına ait parametrelerdir. Her sensör, veri setindeki bir giriş parametresini sembolize eder. Çizelge 3.1.'de ayrıntılı olarak açıklandığı gibi 13 farklı sensöre 8 farklı öncelik ve paket varış zamanı uygulanmıştır. Örneğin, N13, N12 ve N11 sensörleri en yüksek önceliğe ve en yüksek paket varış sürelerine sahipken, N1 ve N2 düğümleri, en düşük önceliğe ve paket varış sürelerine sahiptir. Bu varsayımlar, kalp hastalığında kullanılan veri seti ve girdi parametreleri ile ilgili olarak yapılmıştır. İş çıkarım sonuçları incelendiğinde, yüksek öncelikli sensörlerin paketler arası varış sürelerinin daha yüksek olmasına rağmen paketlerini hedefe başarıyla ilettikleri görülmektedir. Gecikme sonuçları incelendiğinde hem öncelik seviyelerine hem de paket varış süresine göre gecikmeler arasında farklılıklar olduğu görülmüştür. Bu farklılıklar, Çizelge 3.1.'de verilen çekişme penceresi aralıkları arasındaki farklar sayesinde elde edilmiştir. Buradaki temel amaç, kalp hastalığı için daha önemli bazı verilerin hedefe daha hızlı ulaşmasını sağlamaktır.

6. SONUÇLAR VE ÖNERİLER

Uzaktan sağlık izleme ve teşhis hizmetleri her geçen gün önem kazanmaktadır. Özellikle pandemi durumlarında artan sağlık maliyetleri ve yoğun bakım hasta sayısı, boş hastane yataklarının azalması ve kişi başına düşen sağlık çalışanı sayısı hükümetleri önlem almaya sevk ediyor. IoMT, yukarıda belirtilen sorunlar için umut verici bir çözümdür. Buna göre, diyabet ve kalp hastalığı tahmin senaryosunda KVAA'lar, Sis ve Bulut bilişim hesaplama, bulanık mantık ve makine öğrenme algoritmaları (KA, RO, GA, LR ve DVM) yardımıyla bir IoMT çerçevesi önerdik. Önerilen IoMT çerçevesini tüm bileşenlerle tartıştık. Sis bilişim katmanında bulanık mantık, diyabet sisteminde %64 doğruluk performansı sağlamaktadır. Bulut bilişimde ise diyabet hastalığı tahmini için KA, RO, GA, LR ve DVM sırasıyla %78,77, %77,40, %84,93, %80,14 ve %79,45 doğruluk performansına sahiptir. Kalp hastalığı tahmini için ise KA, RO, GA, LR ve DVM sırasıyla %89,74, %92,31, %94,87, %88,46 ve %93,60 doğruluk performansına sahiptir. Diyabet ve kalp hastalığı tahmin performanslarına bakıldığında diyabet ve kalp hastalığı tahmininde GA algoritmasının en iyi performansa sahip olduğu görülmektedir. Bununla birlikte, diyabet tahmininde en düşük performansa RF algoritması sahipken, kalp hastalığı tahmininde ise LR algoritması en düşük performans değerine sahiptir. Bu tahmin değerlerinin farklı çıkmasındaki sebep kullanılan veri setinin uygunluğu olabilmektedir. Veri seti o hastalık için uygun verilere ve parametrelere sahip ise algoritmaların performansları çok daha iyi tahmin değerlerine sahip olacaktır.

Önerilen IoMT çerçevesinde diyabet ve kalp hastalığı benzetim senaryoları, CSMA-CA tabanlı IEEE 802.15.6 protokolü ve AODV yönlendirme algoritması kullanılarak Riverbed Modeler'da oluşturulmuştur. Çeşitli önceliklere sahip heterojen düğümlerin uçtan uca gecikme ve iş çıkarım sonuçları karşılaştırmalı olarak incelenmiştir. Elde edilen sonuçlar ışığında, hayati verilerin hedefe daha düşük gecikme ve yüksek iş çıkarım oranı ile iletildiği tespit edilmiştir. Ayrıca, benzetim senaryolarında üretilen tüm verileri veri akış bileşeninde kullanılan Node-Red ve Apache Kafka yapıları eksiksiz ve hızlı bir şekilde veri analitiği bileşenine ilettiği görülmüştür. Apache Spark gerçek zamanlı tahmin modeli ise özellikle IoMT çerçeveleri gibi gerçek zamanlı uygulamalarda etkili ve verimli performans sağlamaktadır.

Geliştirilen IoMT çerçevesi, hastaneler, bakım evleri vb. gibi topluluklarda kullanılmasına imkan sağlar. Gelecekteki çalışmalar için, çeşitli pandemik ve sağlık uygulamaları için bir IoMT çerçevesi oluşturmak üzere çeşitli makine öğrenimi algoritmaları ve farklı hastalıklar dikkate alınabilir. Ayrıca, Apache Spark veri işleme motoru, benzer özellikteki uygulamalar ile performans olarak karşılaştırılabilir.

7. KAYNAKLAR

[1] R. P. Singh, M. Javaid, A. Haleem, R. Vaishya, ve S. Bahl, "Significance of Health Information Technology (Hit) in Context to Covid-19 Pandemic: Potential Roles and Challenges", *Journal of Industrial Integration and Management*, c. 5, sayı 4, ss. 427–440, 2020.

[2] X. Y. Zhang ve P. Y. Zhang, "Mobile Technology in Health Information Systems-A Review", *European Review for Medical and Pharmacological Sciences,* c. 20, sayı 10, ss. 2140–2143, 2016.

[3] R.Suman, M. Javaid, S.K. Choudhary, A. Haleem, R.P. Singh, D. Nandan, S. Ali ve S. Rab, "Impact of COVID-19 Pandemic on Particulate Matter (PM) Concentration and Harmful Gaseous Components on Indian Metros", *Sustainable Operations and Computers*, c. 2, sayı 2020, ss. 1–11, 2021.

[4] Bv. Reddy ve A. Gupta, "Importance of Effective Communication During COVID-19 Infodemic", *Journal of Family Medicine and Primary Care*, c. 9, sayı 8. ss. 3793–3796, 2020.

[5] T. Ong, H. Wilczewski, S. R. Paige, H. Soni, B. M. Welch, ve B. E. Bunnell, "Extended Reality for Enhanced Telehealth During and Beyond Covid-19: Viewpoint", *JMIR Serious Games*, c. 9, sayı 3, ss. 1-17, 2021.

[6] J. D. Fuhrman, N. Gorre, Q. Hu, H. Li, I. El Naqa, ve M. L. Giger, "A Review of Explainable and Interpretable AI with Applications in COVID-19 Imaging", *Medical Physics*, c. 49, sayı 1, ss. 1–14, 2022.

[7] S. Mirjalali, S. Peng, Z. Fang, C. H. Wang, ve S. Wu, "Wearable Sensors for Remote Health Monitoring: Potential Applications for Early Diagnosis of Covid-19", *Advanced Materials Technologies*, c. 7, sayı 1, ss. 1–20, 2022.

[8] M.M. Jaber, T. Alameri, M.H. Ali, A. Alsyouf, M. Al-Bsheish, B.K. Aldhmadi, S.Y. Ali, S.K. Abd, S.M. Ali, W. Albaker ve M.T. Jarrar, "Remotely Monitoring COVID-19 Patient Health Condition Using Metaheuristics Convolute Networks from IoT-Based Wearable Device Health Data", *Sensors*, c. 22, sayı 3, ss. 1205,

2022.

[9] S.J. Alsunaidi, A. M. Almuhaideb, N.M. Ibrahim, F.S. Shaikh, K.S. Alqudaihi, F.A. Alhaidari, I.U. Khan, N. Aslam, M.S. Alshahrani, "Applications of Big Data Analytics to Control Covid-19 Pandemic", *Sensors*, c. 21, sayı 7, 2021.

[10] I. Ahmed, M. Ahmad, G. Jeon, ve F. Piccialli, "A Framework for Pandemic Prediction Using Big Data Analytics", *Big Data Research*, c. 25, ss. 1-14, 2021.

[11] A. Poonia, S. Ghosh, A. Ghosh, S. B. Nath, S. K. Ghosh, ve R. Buyya, "CONFRONT: Cloud-fog-dew Based Monitoring Framework for COVID-19 Management", *Internet of Things*, c. 16, sayı Aralık, ss. 1-18, 2021.

[12] B. Aksoy ve O. K. M. Salman, "Detection of COVID-19 Disease in Chest X-Ray Images with Capsul Networks: Application with Cloud Computing", *Journal of Experimental and Theoretical Artificial Intelligence*, c. 33, sayı 3, ss. 527–541, 2021.

[13] M. Ijaz, G. Li, L. Lin, O. Cheikhrouhou, H. Hamam, ve A. Noor, "Integration and Applications of Fog Computing and Cloud Computing Based on The Internet of Things for Provision of Healthcare Services at Home", *Electronics*, c. 10, sayı 9, ss. 1077, 2021.

[14] A. S. Kwekha-Rashid, H. N. Abduljabbar, ve B. Alhayani, "Coronavirus disease (COVID-19) cases analysis using machine-learning applications", *Applied Nanoscience*, ss. 1–13, 2021.

[15] A. Bhargava, A. Bansal, ve V. Goyal, "Machine Learning-Based Automatic Detection of Novel Coronavirus (COVID-19) Disease", c. 81, sayı 10, ss. 13731–13750, 2022.

[16] X. Wang, H. Li, C. Sun, X. Zhang, T. Wang, C. Dong ve D. Guo, "Prediction of Mental Health in Medical Workers During COVID-19 Based on Machine Learning", *Frontiers in Public Health*, c. 9, sayı September, ss. 1–13, 2021.

[17] C. Lam, J. Calvert, A. Siefkas, G. Barnes, E. Pellegrini, A. Green-Saxena, J. Hoffman, Q. Mao ve R. Das, "Personalized Stratification of Hospitalization Risk Amidst COVID-19: A Machine Learning Approach", *Health Policy and Technology*, c. 10, sayı 3, ss. 100554, 2021.

[18] O. Ali, M. K. Ishak, ve M. K. L. Bhatti, "A Machine Learning Approach for Early

COVID-19 Symptoms Identification", *Computers, Materials and Continua*, c. 70, sayı 2, ss. 3803–3820, 2022.

[19] F. Ajaz, M. Naseem, S. Sharma, M. Shaba, ve G. Dhiman, "COVID-19: Challenges and its Technological Solutions using IoT", *Current Medical Imaging*, c. 18, sayı 2, ss. 113–123, 2022.

[20] M. M. Khan, S. Mehnaz, A. Shaha, M. Nayem, ve S. Bourouis, "IoT-Based Smart Health Monitoring System for COVID-19 Patients", *Computational and Mathematical Methods in Medicine*, c. 2021, ss. 1–11, 2021.

[21] M. Wazid, A. K. Das, J. J. P. C. Rodrigues, S. Shetty, ve Y. Park, "IoMT Malware Detection Approaches: Analysis and Research Challenges", *IEEE Access*, c. 7, ss. 182459–182476, 2019.

[22] F. Alsubaei, A. Abuhussein, V. Shandilya, ve S. Shiva, "IoMT-SAF: Internet of Medical Things Security Assessment Framework", *Internet of Things*, c. 8, ss. 100123, 2019.

[23] P. Hu, H. Ning, L. Chen, ve M. Daneshmand, "An Open Internet of Things System Architecture Based on Software-Defined Device", *IEEE Internet of Things Journal*, c. 6, sayı 2, ss. 2583–2592, 2019.

[24] M. Tausif, A. Jain, E. Khan, ve M. Hasan, "Memory-Efficient Achitecture for FrWF-Based DWT of High-Resolution Images for IoMT Applications", *Multimedia Tools and Applications*, c. 80, sayı 7, ss. 11177–11199, 2021.

[25] L. Haoyu, L. Jianxing, N. Arunkumar, A. F. Hussein, ve M. M. Jaber, "An IoMT cloud-based real time sleep apnea detection scheme by using the SpO2 estimation supported by heart rate variability", *Future Generation Computer Systems*, c. 98, ss. 69–77, 2019.

[26] R. De Fazio, M. De Vittorio, ve P. Visconti, "Innovative IoT Solutions and Wearable Sensing Systems for Monitoring Human Biophysical Parameters: A Review", *Electronics*, c. 10, sayı 14, ss. 1660, 2021.

[27] D. Koutras, G. Stergiopoulos, T. Dasaklis, P. Kotzanikolaou, D. Glynos, ve C. Douligeris, "Security in IoMT Communications: A Survey", *Sensors*, c. 20, sayı 17, ss. 1–49, 2020.

[28] W. Shi, J. Cao, Q. Zhang, Y. Li, ve L. Xu, "Edge Computing: Vision and

Challenges", *IEEE Internet of Things Journal*, c. 3, sayı 5, ss. 637–646, 2016.

[29] N. Bibi, M. Sikandar, I. U. Din, A. Almogren, ve S. Ali, "IOMT-Based Automated Detection and Classification of Leukemia Using Deep Learning", *Journal of Healthcare Engineering*, c. 2020, ss. 1–12, 2020.

[30] S. S. Rani, S. Selvakumar, K. P. M. Kumar, D. T. Tai, ve E. D. Chelvi, "Internet of Medical Things (IoMT) with Machine Learning–Based COVID-19 Diagnosis Model Using Chest X-Ray Images", *Data Science for COVID-19*, 1. baskı, Londra, İngiltere: Academic Press, 2021, böl. 34, ss. 627–641.

[31] T. Han, L. Zhang, S. Pirbhulal, W. Wu, ve V. H. C. de Albuquerque, "A Novel Cluster Head Selection Technique for Edge-Computing Based IoMT Systems", *Computer Networks*, c. 158, ss. 114–122, 2019.

[32] S. Savaşcı Şen, M. Cicioğlu, ve A. Çalhan, "IoT-based GPS assisted surveillance system with inter-WBAN geographic routing for pandemic situations", *Journal of Biomedical Informatics*, c. 116, sayı Mart, 2021.

[33] Z. Niswati, A. Paramita, ve F. A. Mustika, "Aplikasi Fuzzy Logic dalam Diagnosa Penyakit Diabetes Mellitus pada PUSKESMAS di Jakarta Timur", *Journal Nasional Teknologi dan Sistem Informasi*, c. 2, sayı 3, ss. 21–30, 2016.

[34] G. M. Bressan, B. C. F. de Azevedo, ve R. M. de Souza, "A Fuzzy Approach for Diabetes Mellitus Type 2 Classification", *Brazilian Archives of Biology and Technology*, c. 63, ss. 1–11, 2020.

[35] Q. Zou, K. Qu, Y. Luo, D. Yin, Y. Ju, ve H. Tang, "Predicting Diabetes Mellitus With Machine Learning Techniques", *Frontiers in Genetics*, c. 9, sayı Kasım, ss. 1–10, 2018.

[36] L. J. Muhammad, E. A. Algehyne, ve S. S. Usman, "Predictive Supervised Machine Learning Models for Diabetes Mellitus", *SN Computer Science*, c. 1, sayı 5, ss. 1–10, 2020.

[37] N. Yuvaraj ve K. R. SriPreethaa, "Diabetes Prediction in Healthcare Systems Using Machine Learning Algorithms on Hadoop Cluster", *Cluster Computing*, c. 22, sayı 1, ss. 1–9, 2019.

[38] J. Ramesh, R. Aburukba, ve A. Sagahyroon, "A Remote Healthcare Monitoring Framework for Diabetes Prediction Using Machine Learning", *Healthcare*

Technology Letters, c. 8, sayı 3, ss. 45–57, 2021.

[39] E. T. Tan ve Z. A. Halim, "Healthcare Monitoring System and Analytics Based on Internet of Things Framework", *IETE Journal of Research*, c. 65, sayı 5, ss. 653–660, 2019.

[40] M. Devarajan, V. Subramaniyaswamy, V. Vijayakumar, ve L. Ravi, "Fog-Assisted Personalized Healthcare-Support System for Remote Patients with Diabetes", *Journal of Ambient Intelligence and Humanized Computing*, c. 10, sayı 10, ss. 3747–3760, 2019.

[41] M. Abdel-Basset, G. Manogaran, A. Gamal, ve V. Chang, "Model Based on Soft Computing and IoT", *IEEE Internet of Things Journal*, c. 7, sayı 5, ss. 4160–4170, 2020.

[42] T.A. Khan, S. Abbas, A. Ditta, M.A. Khan, H. Alquhayz, A. Fatima ve M.F. Khan, "IoMT-Based Smart Monitoring Hierarchical Fuzzy Inference System for Diagnosis of Covid-19", *Computers, Materials and Continua*, c. 65, sayı 3, ss. 2591–2605, 2020.

[43] M. Otoom, N. Otoum, M. A. Alzubaidi, Y. Etoom, ve R. Banihani, "An IoT-Based Framework for Early Identification and Monitoring of COVID-19 Cases", *Biomedical Signal Processing and Control*, c. 62, ss. 102149, 2020.

[44] P. M. Kumar ve U. Devi Gandhi, "A Novel Three-Tier Internet of Things Architecture with Machine Learning Algorithm for Early Detection of Heart Diseases", *Computers and Electrical Engineering*, c. 65, ss. 222–235, 2018.

[45] K. K. Kamarajugadda, P. Movva, M. N. Raju, S. A. Kant, ve S. Thatavarti, "IoMT with Cloud-Based Disease Diagnosis Healthcare Framework for Heart Disease Prediction Using Simulated Annealing with SVM", *Smart Sensors for Industrial Internet of Things*, Cham, İsviçre: Springer, 2021, böl. 8, ss. 115–126.

[46] M. A. Khan ve F. Algarni, "A Healthcare Monitoring System for the Diagnosis of Heart Disease in the IoMT Cloud Environment Using MSSO-ANFIS", *IEEE Access*, c. 8, ss. 122259–122269, 2020.

[47] J. M. L. P. Caldeira, J. J. P. C. Rodrigues, ve P. Lorenz, "Toward Ubiquitous Mobility Solutions for Body Sensor Networks on Healthcare", *IEEE Communications Magazine*, c. 50, sayı 5, ss. 108–115, 2012.

[48] R. Schmidt, T. Norgall, J. Mörsdorf, J. Bernhard, ve T. von der Grün, "Body Area Network BAN-a Key Infrastructure Element for Patient-Centered Medical Applications", *Biomedizinische Technik*, c. 47, sayı s1a, ss. 365–368, 2002.

[49] J. Abouei, J. D. Brown, K. N. Plataniotis, ve S. Pasupathy, "Energy Efficiency and Reliability in Wireless Biomedical Implant Systems", *IEEE Transactions on Information Technology in Biomedicine*, c. 15, sayı 3, ss. 456–466, 2011.

[50] G. D. Ntouni, A. S. Lioumpas, ve K. S. Nikita, "Reliable and energy-efficient communications for wireless biomedical implant systems", *IEEE Journal of Biomedical and Health Informatics*, c. 18, sayı 6, ss. 1848–1856, 2014.

[51] S. Ullah, P. Khan, N. Ullah, S. Saleem, H. Higgins, ve K. Sup Kwak, "A Review of Wireless Body Area Networks for Medical Applications", *International Journal of Communications, Network and System Sciences*, c. 02, sayı 08, ss. 797–803, 2009.

[52] X. Lai, Q. Liu, X. Wei, W. Wang, G. Zhou, ve G. Han, "A Survey of Body Sensor Networks", *Sensors*, c. 13, sayı 5, ss. 5406-5447, 2013.

[53] A.C. Wong, M. Dawkins, G. Devita, N. Kasparidis, A. Katsiamis, O. King, F. Lauria, J. Schiff ve A.J. Burdett, "A1 V5 mA Multimode IEEE 802.15. 6/Bluetooth Low-Energy WBAN Transceiver for Biotelemetry Applications", *IEEE Journal of Solid-State Circuits*, c. 48, sayı 1, ss. 186–198, 2012.

[54] B. H. Jung, R. U. Akbar, ve D. K. Sung, "Throughput, Energy Consumption, and Energy Efficiency of IEEE 802.15.6 Body Area Network (BAN) MAC protocol", *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, Oulu, Finlandiya, 2012, ss. 584–589.

[55] T. Benmansour, T. Ahmed, ve S. Moussaoui, "Performance Evaluation of IEEE 802.15.6 MAC in Monitoring of a Cardiac Patient", *Conference on Local Computer Networks*, Dubai, Birleşik Arap Emirlikleri, 2016, ss. 241–247.

[56] S. Savaşçı Şen, M. Cicioğlu, ve A. Çalhan, "Kablosuz Vücut Alan Ağları için Coğrafi Tabanlı Yönlendirme Algoritmasının Başarım Analizi", *Düzce Üniversitesi Bilim ve Teknoloji Dergisi*, c. 8, ss. 2480–2490, 2020.

[57] C. E. Perkins ve E. M. Royer, "Ad-Hoc On-Demand Distance Vector Routing", *2nd IEEE Workshop on Mobile Computing Systems and Applications*, New

Orleans, Louisiana, ABD, 1999, ss. 90–100.

[58] J. Loo, J. Lloret Mauri, ve J. Hamilton Ortiz, "Mobile Ad Hoc Routing Protocols", *Mobile Ad Hoc Networks: Current Status and Future Trends*, Florida, ABD: CRC Press, Taylor & Francis, 2011, böl. 2, ss. 19-35.

[59] Apache Spark. (15 Nisan 2022) "Apache Spark" [Online]. Erişim: https://spark.apache.org/.

[60] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauly, M.J. Franklin, S. Shenker ve I. Stoica, "Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing", *9th USENIX Symposium on Networked Systems Design and Implementation*, Kaliforniya, ABD, 2012, ss. 15–28.

[61] A. N. M. JayaLakshmi ve K. V. Krishna Kishore, "Performance Evaluation of DNN with Other Machine Learning Techniques in a Cluster Using Apache Spark and MLlib", *Journal of King Saud University - Computer and Information Sciences*, c. 34, sayı 1, ss. 1311–1319, 2022.

[62] H. Luu, "Working with Apache Spark", *Beginning Apache Spark 2*, Kaliforniya, ABD: Apress, 2018, böl. 2, ss. 15-49.

[63] Apache Kafka. (15 Nisan 2022) "Apache Kafka" [Online]. Erişim: https://kafka.apache.org/.

[64] Tutorial Point. (15 Nisan 2022) "Apache Kafka Tutorial" [Online]. Erişim: https://www.tutorialspoint.com/apache_kafka/index.htm.

[65] F. Junqueira ve B. Reed, "Zookeeper Consepts and Basics", *ZooKeeper: Distributed Process Coordination*, Kaliforniya, ABD: O'Reilly Media, 2013, böl. 1, ss. 3-42.

[66] L. A. Zadeh, "Fuzzy Sets", *Information and Control*, c. 8, sayı 3, ss. 338–353, 1965.

[67] L. A. Zadeh, "Fuzzy Sets as a Basis for a Theory of Possibility", *Fuzzy Sets and Systems*, c. 1, sayı 1, ss. 3–28, 1978.

[68] J. S. R. Jang, C. T. Sun, ve E. Mizutani, "Neuro-Fuzzy and Soft Computing-A Computational Approach to Learning and Machine Intelligence", *IEEE*

Transactions on Automatic Control, c. 42, sayı 10, ss. 1482–1484, 2005.

[69] E. H. Mamdani ve S. Assilian, "An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller", *International Journal of Man-Machine Studies*, c. 7, sayı 1, ss. 1–13, 1975.

[70] S. Rizvi, J. Mitchell, A. Razaque, M. R. Rizvi, ve I. Williams, "A Fuzzy Inference System (FIS) to Evaluate The Security Readiness of Cloud Service Providers", *Journal of Cloud Computing*, c. 9, sayı 1, ss. 1-17, 2020.

[71] T. Pang-Ning, S. Michael, ve K. Vipin, *Introduction to data mining,* 1. baskı, Hindistan: Pearson, 2006, böl. 4, ss. 25-44.

[72] G. James, D. Witten, T. Hastie, ve R. Tibshirani, "Tree-Based Methods", *An Introduction to Statistical Learning,* 1. baskı, New York, ABD: Springer, 2021, böl. 8, ss. 327–365.

[73] A. Natekin ve A. Knoll, "Gradient Boosting Machines, a Tutorial", *Frontiers in Neurorobotics*, c. 7, ss. 21, 2013.

[74] V. A. Dev ve M. R. Eden, "Formation Lithology Classification Using Scalable Gradient Boosted Decision Trees", *Computers and Chemical Engineering*, c. 128, sayı 2019, ss. 392–404, 2019.

[75] S. Le Cessie ve J. C. Van Houwelingen, "Ridge Estimators in Logistic Regression", *Applied Statistics*, c. 41, sayı 1, ss. 191, 1992.

[76] M. Cicioğlu ve A. Çalhan, "Energy Efficiency Solutions for IEEE 802.15.6 Based Wireless Body Sensor Networks", *Wireless Personal Communications*, c. 119, sayı 2, ss. 1499–1513, 2021.

[77] G. Kokkonis, "an Open Source Architecture of a Wireless Body Area Network in a Medical Environment", *International Journal of Digital Information and Wireless Communications*, c. 6, sayı 2, ss. 63–77, 2016.

[78] M. Lekić ve G. Gardašević, "IoT Sensor Integration to Node-RED Platform", *17th International Symposium on Infoteh-Jahorina*, Batı Sarajevo, Bosna Hersek, 2018, ss. 1–5.

[79] F. Maulidina, Z. Rustam, S. Hartini, V. V. P. Wibowo, I. Wirasati, ve W. Sadewo, "Feature Optimization Using Backward Elimination and Support Vector Machines

(SVM) Algorithm for Diabetes Classification", *Journal of Physics: Conference Series*, c. 1821, sayı 1, ss. 1-7, 2021.

[80] P. Rani, R. Kumar, N. M. O. S. Ahmed, ve A. Jain, "A Decision Support System for Heart Disease Prediction Based Upon Machine Learning", *Journal of Reliable Intelligent Environments*, c. 7, sayı 3, ss. 263–275, 2021.

[81] A. Astrin, "Part 15.6: Wireless Body Area Networks", *IEEE Standards for Local and Metropolitan Area Networks,* New York, ABD: IEEE Stardards Association, 2012, böl. 6, ss. 77-148.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Emre YILDIRIM

Yabancı Dili : İngilizce

ÖĞRENİM DURUMU

Derece	Alan	Okul/Üniversite	Mezuniyet Yılı
Doktora	Elektrik-Elektronik ve Bilgisayar Mühendisliği	Düzce Üniversitesi	2022
Yüksek Lisans	Bilgisayar ve Öğretim Teknolojileri Öğretmenliği	Çanakkale Onsekiz Mart Üniversitesi	2017
Lisans	Bilgisayar ve Öğretim Teknolojileri Öğretmenliği	Çanakkale Onsekiz Mart Üniversitesi	2012
Lise	Bilgisayar	Silivri Teknik Lisesi	2004

MESLEKİ DENEYİM

Görev Dönemi	Unvan	Çalıştığı Kurum	Birim
2019- …	Öğretim Görevlisi	Osmaniye Korkut Ata Üniversitesi	Osmaniye MYO Bilgisayar Teknolojileri Bölümü
2014-2019	Öğretim Görevlisi (Ders Vermeyecek)	Kırklareli Üniversitesi	Bilgi İşlem Daire Başkanlığı
2007-2008	Yazılım Uzmanı	Tiffany	Bilgi İşlem
2005-2007	Bilgi İşlem Sorumlusu	Marin Princess Hotel	Bilgi İşlem

PROJELER

Görev Tanımı	Proje Adı	Destekleyen Kurum	Tarih	Durum
Proje Araştırmacısı	Kırklareli Üniversitesi Mobil Uygulamalarının Geliştirilmesi	Kırklareli Üniversitesi Bilimsel Araştırma Projeleri	2017-2019	Bitti

ULUSLARARASI (SCI, SCI-E) HAKEMLİ DERGİLERDE YAYIMLANAN MAKALELER

Yazarlar	Makale Başlığı	Dergi Adı	Cilt (Sayı) Sayfa	Tarih
Emre YILDIRIM, Ali ÇALHAN, Murtaza CİCİOĞLU,	Performance Analysis of Disease Diagnostic System Using IoMT and Real-Time Data Analytics	Concurrency and Computation: Practice and Experience		2022

ULUSLARARASI DİĞER HAKEMLİ DERGİLERDE YAYIMLANAN MAKALELER

Yazarlar	Makale Başlığı	Dergi Adı	Cilt (Sayı) Sayfa	Tarih
Mehmet Ali SALAHLI, Emre YILDIRIM, T.Gasimzadeh, F.Alasgarovad, A.Guliyeve	One Mobile Application for The Development of Programming Skills of Secondary School Students	Procedia Computer Science	120 502-508	2017

ULUSAL HAKEMLİ DERGİLERDE (TR DİZİN) YAYIMLANAN MAKALELER

Yazarlar	Makale Başlığı	Dergi Adı	Cilt (Sayı) Sayfa	Tarih
Emre YILDIRIM, Ali	Apache Spark ile Makine Öğrenmesi Destekli Diyabet Rahatsızlığı Tahmini	Düzce Üniversitesi Bilim ve Teknoloji		2022

ÇALHAN		Dergisi		
Emre YILDIRIM, Ali ÇALHAN	Apache Spark Tabanlı Duygu Analizi	Osmaniye Korkut Ata Üniversitesi Fen Bilimleri Enstitüsü Dergisi	4(3) 233-241	2021

ULUSLARARASI BİLİMSEL TOPLANTILARDA SUNULAN BİLDİRİLER

Yazarlar	Bildiri Başlığı	Toplantı Adı	Tarih
Mehmet Ali SALAHLI, Emre YILDIRIM	Scratch Programlama Dili Eğitimine Yönelik Bir Mobil Uygulamanın Geliştirilmesi	ULEAD 2017 7th International Congress of Reseach in Education	2017
Ömer KIRMACI, Emre YILDIRIM	Dijital Öykülemede Scratch Kullanımı	11. International Computer Instructional Technologies Symposium (ICITS2017)	2017
Bora ASLAN, Emre YILDIRIM, Veli Özcan BUDAK, Gökhan DOĞAN, Edip Serdar GÜNER, Füsun YAVUZER ASLAN	Mobil Uygulama Geliştirme Süreci Analizi: Kırklareli Üniversitesi Örneği	International Congress on Entrepreneurship, Technology, Innovation and Design	2018