MARMARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BÜYÜK VERİ TEKNOLOJİLERİ VE VERİ MADENCİLİĞİ YÖNTEMLERİ İLE MEDİKAL VERİ ANALİZİ

SÜLEYMAN ÖZER

YÜKSEK LİSANS TEZİ
Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Programı

DANIŞMAN
Dr. Öğr. Üyesi Zehra Aysun ALTIKARDEŞ

İSTANBUL, 2019

MARMARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BÜYÜK VERİ TEKNOLOJİLERİ VE VERİ MADENCİLİĞİ YÖNTEMLERİ İLE MEDİKAL VERİ ANALİZİ

SÜLEYMAN ÖZER
(523616012)

YÜKSEK LİSANS TEZİ
Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Programı

DANIŞMAN
Dr. Öğr. Üyesi Zehra Aysun ALTIKARDEŞ

İSTANBUL, 2019

MARMARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Marmara Üniversitesi Fen Bilimleri Enstitüsü Yüksek Lisans Öğrencisi Süleyman ÖZER'in "Büyük Veri Teknolojileri ve Veri Madenciliği Yöntemleri ile Medikal Veri Analizi" başlıklı tez çalışması, 28 Ağustos 2019 tarihinde savunulmuş ve jüri üyeleri tarafından başarılı bulunmuştur.

Jüri Üyeleri

Dr. Öğr. Üyesi Zehra Aysun ALTIKARDEŞ (Danışman)
Marmara Üniversitesi Teknik Bilimler Meslek Yüksekokulu Bilgisayar Teknolojileri Bölümü

(İMZA)...

Doç. Dr. Tugay Turgay BİLGİN (Üye)
Bursa Teknik Üniversitesi Mühendislik ve Doğa Bilimleri Fakültesi Bilgisayar Mühendisliği Bölümü

(İMZA)...

Dr. Öğr. Üyesi Önder DEMİR (Üye)
Marmara Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği Bölümü

(İMZA)...

ONAY

Marmara Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 04/09/2019 tarih ve 2019/18-08 sayılı kararı ile Süleyman ÖZER'in Bilgisayar Mühendisliği (Türkçe) Programında Yüksek Lisans derecesi alması onanmıştır.

Fen Bilimleri Enstitüsü Müdürü

Prof. Dr. Bülent EKİCİ

ÖNSÖZ

Bu çalışmanın tamamlanmasında bolca emeği geçen tez danışmanım Dr. Öğr. Üyesi Zehra Aysun Altıkardeş'e,
Marmara Üniversitesi Hipertansiyon ve Ateroskleroz Eğitim, Uygulama ve Araştırma Merkezi Müdürü Prof. Dr. Ali Serdar Fak' a ve
Türk Kalp Vakfı çalışanlarına
Teşekkürlerimi sunarım.

Ağustos,2019 Süleyman Özer

İÇİNDEKİLER/

ÖZET
BÜYÜK VERİ TEKNOLOJİLERİ VE VERİ MADENCİLİĞİ YÖNTEMLERİ İLE MEDİKAL VERİ ANALİZİ

Her canlının günün yorgunluğunu atabilmeleri için sağlıklı uykuya ihtiyacı vardır. Uyku, canlıların hem zihinsel hem bedenen dinlenmesini sağlamaktadır. Uyku kalitesi hem daha kaliteli yaşam sürdürmelerine olanak tanır hem de fiziki ve zihni sağlıklarının korunmasına katkı sağlamaktadır. Öyle ki kaliteli uyku uyuyamayan bireylerde algıda yavaşlık, zihin yorgunluğu ve fiziki yorgunluklar baş göstermektedir. Bunlar kalitesiz uykunun canlılar üzerindeki gözle görülür etkileridir. Bir de sağlık açısından etkileri bulunmaktadır. Zihin ve kasların dinlenmeye ihtiyacı olduğu gibi iç organların da dinlenmeye ihtiyaçları vardır. Nitekim insan metabolizması bile uyku halinde yavaşlamakta ve organların dinlenmesine olanak tanımaktadır. Uyku sırasındaki bu yavaşlamanın olup olmadığını 24 saat boyunca alınan tansiyon verileriyle teşhis edilmektedir. Bunu da gezici kan basıncı izleme cihazındaki manşon adı verilen bir parçayı bireylerin kollarına takarak 24 saat boyunca hem uyku hem uyanıklık halinde sağlık verileri ölçülerek elde edilmektedir. Gezici kan basıncı izleme cihazı çalışırken manşon kolu sıkıştırmakta ve ölçümü bu sayede yapabilmektedir. Bu durum uyku halinde olan hastayı uyandırmakta ve doğru sonuçların alınamamasına yol açmaktadır.

Yapılan çalışmada gezici kan basıncı izleme cihazına ihtiyaç duymaksızın kişilerin tahlil verilerinden dipper/non-dipper olup olmadığını veri madenciliği yöntemleriyle karar verilmesini sağlayacak altyapı oluşturulmuştur. Bu bağlamda birey verilerini muhafaza etmek ve işlemek için büyük veri ortamı kurulmuş ve kullanılmıştır. Bu ortamın kullanımı sayesinde büyüyen veri miktarını karşılama kapasitesi sağlanmış, aynı zamanda real-time projelere Hipertansiyon ve Ateroskleroz Eğitim, Uygulama ve Araştırma Merkezi (HİPAM) bünyesi için altyapı oluşturulmuştur.

Giriş başlığı altında yapılan çalışmaya ait ön bilgilendirmeler ve literatürdeki seçilen 13 adet benzer çalışmalar, materyal ve yöntem başlığı altında işletim sistemi konfigürasyon ayarlarının yapılması, büyük veri ortamının kurulması, modem ayarlarının yapılması, veritabanı işlemleri, veri temizleme işlemleri, tahminleme ve raporlamaya ait bilgilendirmelere yer verilmiştir. Tezin en sonunda ise çalışmada elde edilen bulgular

değerlendirilmiştir. Sonuç olarak iki adet farklı Veriseti için Naive Bayes ve Random Forest algoritmaları kullanılarak sırasıyla Veriseti-1 için %70,37, %81,48 ve Veriseti-2 için %73,07, %80,76 doğruluk oranlarıyla tahminleme modelleri oluşturulmuş ve bunlar yorumlanarak çalışmanın ileriki aşamaları için öneriler sunulmuştur.

ABSTRACT
BIG DATA TECHNOLOGIES AND MEDICAL DATA ANALYSIS USING DATA MINING METHODS

Every living thing needs a healthy sleep in order to relieve the tiredness of the day. Sleep provides both mental and physical rest for living beings. Sleep quality not only allows them to lead a better quality of life, but also contributes to the protection of their physical and mental health. So much so that people who can not sleep quality sleep slows in perception, mental fatigue and physical fatigue. These are the visible effects of poor quality sleep on living things. There are also health effects. The mind and muscles need rest, as well as the internal organs need rest. As a matter of fact, even human metabolism slows down during sleep and allows organs to rest. The presence of this slowdown during sleep is diagnosed by blood pressure readings taken for 24 hours. This is achieved by measuring the health data in both sleep and wake state for 24 hours by attaching a piece of sleeve in the mobile blood pressure monitoring device to the arms of individuals. The mobile blood pressure monitoring device clamps the sleeve arm while it is in operation and is thus able to make the measurement. This situation awakens the dormant patient and leads to the inability to obtain the correct results.

In this study, it is provided an infrastructer whether or not dipper / non-dipper is the data of the people without the need for mobile blood pressure monitoring device by using data mining methods, and for this purpose, a large data environment has been established and used to maintain and process individual data. Thanks to the use of this environment, the capacity to meet the growing amount of data ensured and real-time projects will be provided with infrastructure for Hipertansiyon ve Ateroskleroz Eğitim, Uygulama ve Araştırma Merkezi (HİPAM).

Preliminary information about the study conducted under the title of introduction and 13 similar studies selected in the literature, configuration of operating system configuration under the title of material and method, setting up of the big data environment, setting up the modem, database operations, data cleaning operations, information on forecasting and reporting it is given. At the end of the thesis, the findings obtained in the study were evaluated. As a result, two different Veriseti algorithms using Naive Bayes and Random Forest algorithms, respectively, for Veriseti-1 70.37%, 81.48% and 73.07% for Veriseti-2, 80.76% accuracy and estimation models were formed suggestions for the next stages of the study are presented.

ŞEKİL LİSTESİ

TABLO LİSTESİ

KISALTMALAR/ABBREVIATIONS

DB : Database

JDK : Java Development Kit

BAPKO : Bilimsel Araştırma Projeleri Koordinasyon Birimi

TKV : Türk Kalp Vakfı

HİPAM : Hipertansiyon ve Ateroskleroz Eğitim, Uygulama ve Araştırma Merkezi

DHCP : Dynamic Host Configuration Protocol

WHO : World Health Organization

BOH : Bulaşıcı Olmayan Hastalıklar

ABPM : 24-saat gezici kan basıncı izleme - 24-hour ambulatory blood pressure monitoring

THC : Tansiyon Holter Cihazı

SSH : Secure Shell

NTP : Network Time Protocol

ETL : Extract, Transform, Load

HDFS : Hadoop Distributed File System

IoT : Nesnelerin İnterneti - Internet of Things

TN : Doğru Olumsuz - True Negative

TP : Doğru Olumlu - True Positive

FN : Yanlış Olumsuz - True Negative

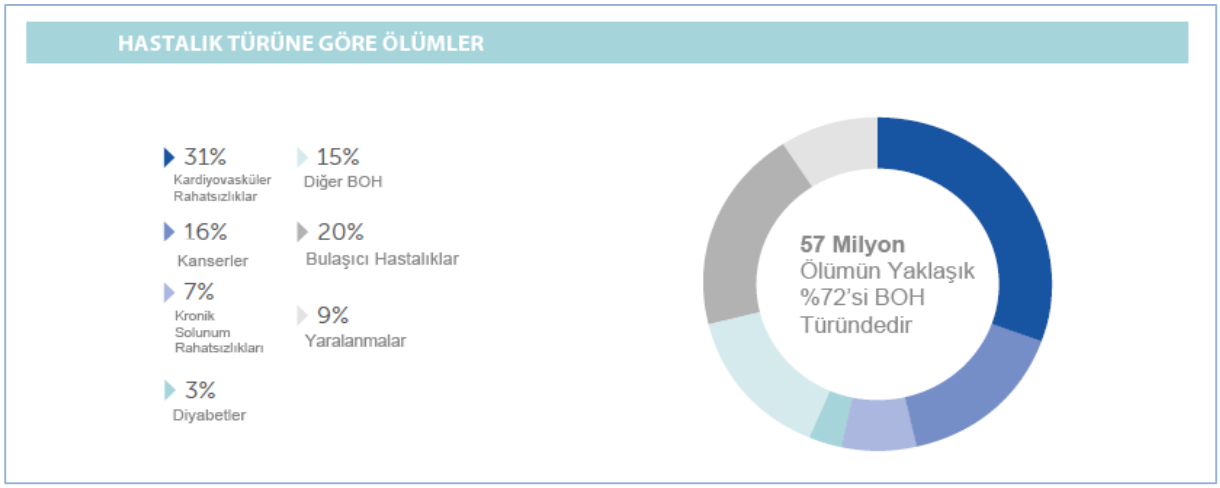
FP : Yanlış Olumlu - True Positive

UML : Birleşik Modelleme Dili – Unified Modeling Language

1. GİRİŞ

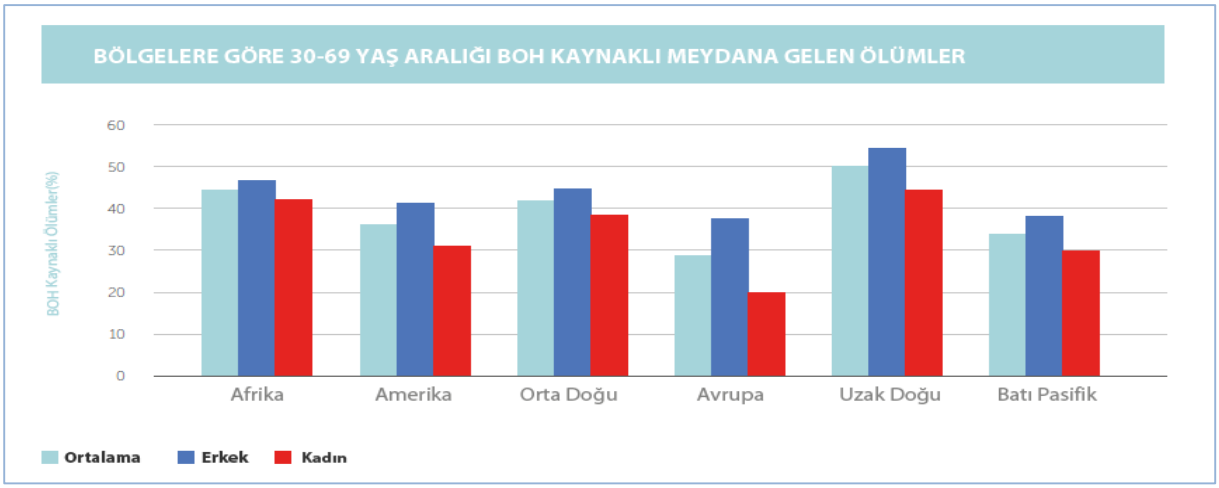
Her canlının günün yorgunluğunu atabilmesi için sağlıklı uykuya ihtiyacı vardır. Uyku, canlıların hem zihinsel hem bedenen dinlenmesini sağlamaktadır. Uyku kalitesi daha kaliteli yaşam sürdürmelerine, fiziki ve zihni sağlıklarının korunmasına katkı sağlamaktadır. Öyle ki kaliteli uyku uyuyamayan bireylerde algıda yavaşlık, zihin ve fiziki yorgunluklar baş göstermektedir. Bunlar kalitesiz uykunun canlılar üzerindeki gözle görülür etkileridir. Zihin ve kasların olduğu kadar iç organların da sağlık açısından dinlenmeye ihtiyaçları vardır. Bundan dolayı insan metabolizması uyku halinde yavaşlamakta ve organların dinlenmesine olanak tanımaktadır. Örneğin insan vücudu için önemli organlardan biri olan kalp, derin uyku halinde uyanık haline göre yaklaşık %10 daha yavaş çalışmaktadır. Uyku problemi yaşayan bireylerde derin uykuya geçememeleri halinde kalp yavaşlamamakta bu da beraberinde sağlık problemlerini getirmektedir. Yapılan bir çalışmada hipertansiyonu olan bireylerden derin uykuya geçemeyenlerin, diğer bireylere göre kardiyovasküler risk faktörlerinin üç kat daha fazla olduğu gösterilmiştir [1]. Kardiyovasküler hastalıkların önemini anlamak adına bu hastalıktan ölenlerin istatistiksel durumu Dünya Sağlık Örgütü-World Health Organization (WHO) verileriyle aşağıda sunulmaktadır.

2016 yılında gerçekleşen 57 milyon ölümün 41 milyonu bulaşıcı olmayan hastalıklar (BOH)'dan dolayıdır [2]. Ölümler nedenlerine göre incelendiğinde, en fazla ölüm %31 ile kardiyovasküler hastalıklardan dolayı gerçekleşmektedir. Bunu %20 ile bulaşıcı hastalıklar, %16 ile kanser, %15 ile diğer bulaşıcı olmayan hastalıklar, %9 yaralanma sonucu, %7 kronik solunum rahatsızlıkları ve %3 diyabet izlemektedir [2] (Şekil 1.1).



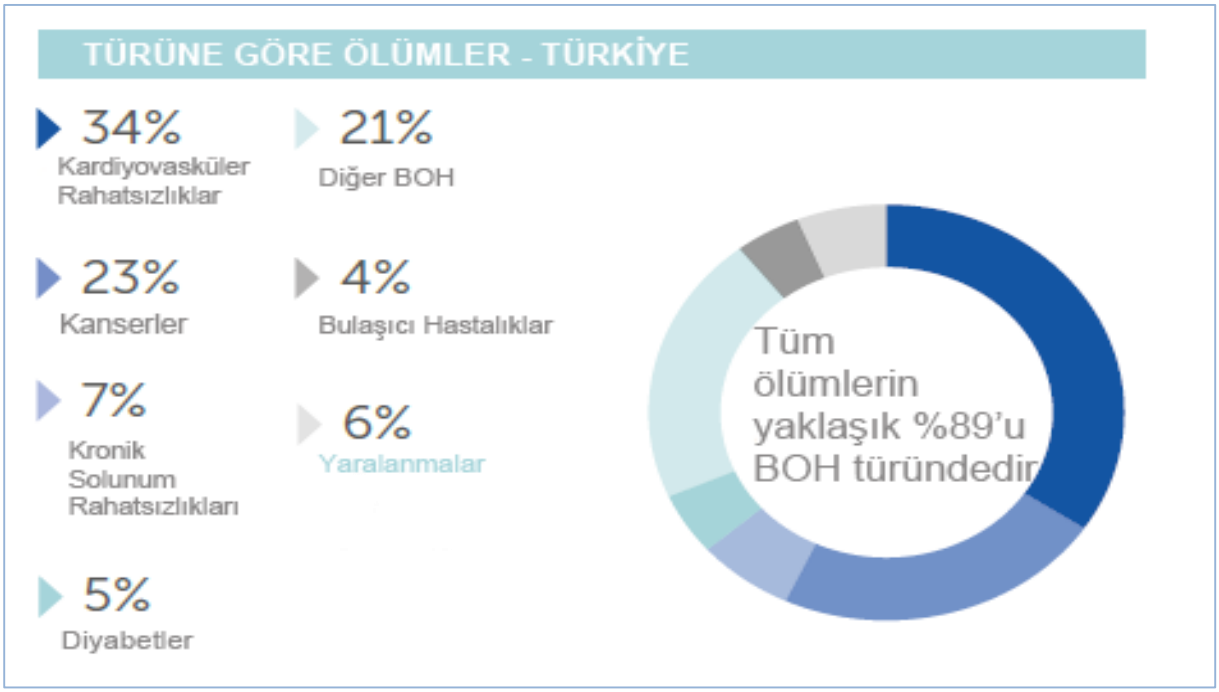
Şekil 1.1. Dünya Genelinde Hastalık Türüne Göre Ölüm Oranı

Ölümlerin bölgelere göre kırılımlarına bakıldığında %24 Doğu Akdeniz, %23 Güneydoğu Asya, %22 Afrika, %15 Amerika, %17 Avrupa, %16 Batı Pasifik şeklindedir [2] (Şekil 1.2).



Şekil 1.2. Bölgelere Göre Ölüm Oranları

Dünyadaki durum böyle iken Türkiye'de hastalık türüne göre ölüm oranları %34 kardiyovasküler hastalıklar, %23 kanser, %21 diğer BOH, %7 kronik solunum rahatsızlığı, %6 yaralanma sonucu, %5 diyabet ve %4 bulaşıcı hastalıklar şeklindedir [2] (Şekil 1.3).



Şekil 1.3. Türkiye'de Hastalık Türüne Göre Ölüm Oranı

Bu istatistiksel verilerden de görüleceği üzere dünya genelinde kardiyovasküler hastalıklar BOH'lar arasında en fazla ölüm oranına sahip hastalıkları teşkil etmekte ve bu yüksek oranın azaltılması için çalışmaların yapılması gerekmektedir. Bu amaca yönelik olarak ülkemizde Türk Kalp Vakfı (TKV) bu konu üzerine ciddiyetle eğilmekte ve hem farkındalık yaratma hem de tıbbi takiplerle ilgili birçok çalışmaya imza atmaktadır. Aynı amaçla Marmara Üniversitesi Hipertansiyon ve Ateroskleroz Eğitim, Uygulama ve Araştırma Merkezi (HİPAM)'da çeşitli projeler yürütmektedir. Bu iki kurum arasında 17.04.2017 tarihinde, "tıbbi datanın HİPAM bünyesinde yapay zekâ ve akıllı sistemler çerçevesinde analiz edilmesi ve bilimsel araştırmalar yapılması" amacıyla ortak protokol imzalanmıştır [http://hipam.marmara.edu.tr/notice/hipamin-yeni-yerinin-acilis-toreni/]. Bahsi geçen protokol gereğince ve alınan Marmara Üniversitesi Tıp Fakültesi Klinik Araştırmalar Etik Kurul onayı (07.12.2018 tarih/Protokol kodu 09.2018.876) kapsamında ortak çalışma başlatılmıştır.

Bu kapsamda gerçekleştirilen bu tez çalışmasının amacı;

- Günümüzde büyük veri yapılarının kazandığı önem düşünülerek, HİPAM için buna uygun bir sistem mimarisi tasarlanarak sunucu altyapısının kurulması,
- TKV tarafından takip edilmiş 2015 yılı ve sonrasındaki 27.000 hasta verisinin bu altyapıya aktarılması,
- Retrospektif olarak yapılacak çalışma için bu veriler içinden Demografik, 24-saat gezici kan basıncı izlenmesi- Ambulatory Blood Pressure Monitoring (ABPM), ve Laboratuvar verileri olan hastaların tespit edilerek yeni verisetinin oluşturulması,
- Bu veriseti üzerinde temizleme ve ön işleme yapılarak üzerinde tıbbi amaca yönelik yapay zekâ teknikleriyle çalışılabilecek bir veriseti oluşturulması,
- Bu veriseti üzerinde dipper/non-dipper tahminleme modeli oluşturmak için Naive Bayes ve Random Forest algoritmaları ile Apache Spark uygulaması içerisinde Scala dili kullanılarak pilot çalışmanın geliştirilmesi,
- Yapılacak çalışmalara ait raporların web ve mobil platform üzerinden ulaşılabilir olmasını sağlayacak sistemin oluşturulmasıdır.

Bu proje FEN-C-YLP-200318-0119 numarası ile Marmara Üniversitesi Bilimsel Araştırma Projeleri Koordinasyon Birimi (BAPKO) tarafından desteklenmiştir.

1.1. Ön Bilgi

Bu tez çalışmasının daha iyi anlaşılabilmesi amacıyla, tıbbi ve teknik konularda iki farklı alt başlık altında ön bilgilendirme sunulmuştur.

1.1.1. Tıbbi açıdan bilgilendirme

Sağlıklı bireylerin derin uykuya geçmeleri halinde, yukarıda da bahsi geçtiği gibi kan basıncı değerlerinin %10 ve üzeri düşüş göstermesi gerekmektedir. Bu durum 'dipper', olarak adlandırılmaktadır. Kardiyovasküler hastalığı olan bireylerde ise bu düşüş yaşanmayabilmektedir. Bu durum ise 'non-dipper' olarak adlandırılmaktadır.

Dipper/non-dipper tespiti için ABPM cihazı kullanılmakta, ancak hastanın yaşam kalitesine ve maliyetlere bazı olumsuz etkileri bulunmaktadır. Bunlar aşağıda listelenmiştir.

- Bireylerin uzunca bir süre koluna takılı manşon ile yaşamaları hareketlerini kısıtlamakta,
- Kola takılan manşon yüzünden bireyler huzursuz hissetmelerinden ötürü derin uykuya girememekte,
- Ölçüm yapılabilmesi için manşonun takılı olduğu kola uyguladığı basınç uyku sırasında bireylerin uyanmasına neden olabilmekte,
- ABPM cihazı alımı ve kullanacak personelin tayin edilmesinin maliyetleri,
- Hastanın bu cihazı almak ve teslim etmek üzere hastaneye gitmesidir.

ABPM cihazı ile dipper/non-dipper tespiti aşamalarının tüm olumsuz etkilerini ortadan kaldırmak ve kardiyovasküler hastalığı olan bireylerin kolayca teşhis edilmesiyle birlikte kardiyovasküler hastalıklardan dolayı ölüm oranlarının azaltılmasının amaçlandığı TKV-HİPAM ortaklığında yürütülen büyük ölçekli proje için sistem altyapısı, tahminleme modellerinin oluşturulması ve raporların çıkarılması bu tez çalışmasında gerçekleştirilmiştir.

1.1.2. Teknik açıdan bilgilendirme

Son zamanların en değerli olgularından biri olan veri her geçen gün daha da anlamlandırılarak hayata dair önemli kararlar alınmasına, sonuçlar çıkarılmasına katkı sağlamaktadır. Verinin gücünü bilen ve bu güce hükmetmek isteyen firmalar ve kişiler veriyi saklamanın önemini bilmenin yanı sıra veriyi anlamlandırmak ve veriden değer

üretmek de istemektedirler. Çünkü verinin anlamlandırılması hem firmaya hem de bireylere katkılar sağlamaktadır. Bu nedenle bu tip projeler daha da değer kazanmaktadır. Bu değerin üç aşamalı sağlandığı görülmektedir. İlk aşaması olan veri saklama süreci, ikinci aşaması ise bu saklanan verilerin işlenerek anlamlandırılması ve üçüncü aşamasında ise raporlanmasıdır.

Bundan birkaç yıl öncesine kadar veri bilgisayara girilmiş kayıtlardan ibaret olsa da günümüzde her alandan toplanmakta ve işlenmektedir. Örneğin akıllı telefona sahip olan kullanıcılar telefonlarından ev ilanlarına baktıklarında konuyla ilgili kredi reklamlarının çıktığını görmektedirler. Bu da açık bir şekilde hayatın her anındaki verinin toplandığını, işlendiğini ve anlamlandırıldığını göstermektedir. Verilerin toplanması günlük hayatımızın içine bu kadar girmişken bunca verinin nerede ve nasıl tutulduğu ya da kitlesel toplanan verilerin nasıl işlendiği konusunda bilgi sahibi olmak gerekmektedir.

Günümüzdeki teknolojinin kullanımının artmasıyla birlikte veri miktarları da artmaktadır. 2012 rakamlarıyla dünyada bir günde 2,5 kentilyon byte veri üretilmekte. 10 yıl içinde de toplam veri büyüklüğünün şu anki zamanın 44 katına ulaşacağı tahmin edilmektedir [1]. Veri miktarının artmasıyla birlikte zaman içerisinde büyük veri kavramı gelişmiş ve bilişim alanının ilgi odağı olmuştur. Büyük veride önemli olan kavramlar Şekil 1.4'teki görselde verilmiştir.

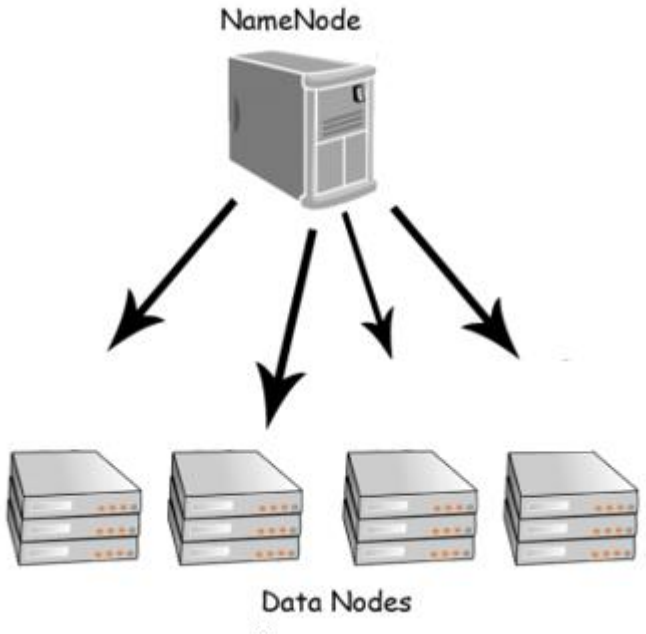
Şekil 1.4. Büyük Veri Karakteristikleri

Yukarıda tanımı yapılan büyük verinin işlenmesi, yönetilmesi ve depolanmasında kullanılabilir ürünler aşağıda başlıklar halinde ele alınmıştır.

2.1.1.1. Apache Hadoop

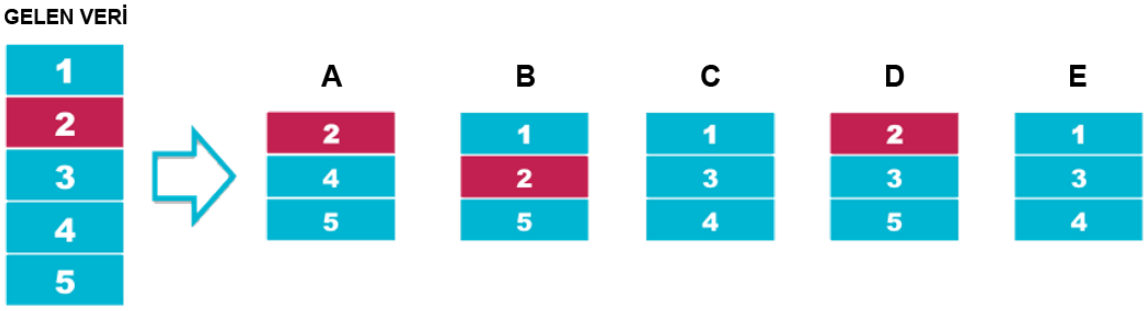
Apache Hadoop verilerin saklanması, işlenmesi, anlamlandırılması ve raporlanmasına olanak tanıyan veri platformudur. Son zamanlarda adından sıkça söz ettiren bu platform yapısal olarak performansını birbirine bağlanan çeşitli sunucuların kaynaklarını eş zamanlı ve dağıtık kullanabilmesinden almaktadır. Apache Hadoop platformunda kullanılacak sunucuların sayıca çok olması sistem performansına doğrudan etki etmektedir. Platformda kullanılacak sunucular tercih edilirken az sayıda sistemsel özellikleri çok yüksek sunuculardan ziyade çok sayıda sistemsel özellikleri yüksek olmayan sunucular kabul edilmelidir. Elde edilen bu yapı büyük veri platformunun yapıtaşı olacaktır.

Büyük veri platformunda kullanılan sunuculara kullanım şekillerine göre ad düğümü-name node (master) ya da veri düğümü-data node (slave) denmektedir. Ad düğümü büyük veri platformunun yönlendiricisi ve düzenleyicisi olarak kullanılmaktadır. İstenen taleplerin ilk gittiği ve işlendiği yer ad düğümleridir. Veri düğümleri ise ad düğümlerinden gelen talepleri yerine getiren, verileri barındıran sunuculardır. HDFS mimarisi Şekil 1.5'te görünmektedir.



Şekil 1.5. HDFS Mimarisi

Apache Hadoop sistemde tanımlanan sunucuları ortak kullanarak verileri depolar. Bu yöntem hem verileri hızlı okuma ve yazma hem de güvenilir depolama olanağı tanımaktadır. Bu mimariye Hadoop Dağıtık Dosya Sistemi-Hadoop Distributed File System (HDFS) denmektedir. HDFS konfigürasyonu yapılırken verilerin blok boyutu ve kopyalanma sayısı (replication) belirlenir. Bu ayarlar sayesinde sisteme alınacak veriler depolama alanlarına bloklanarak ve kopyası alınarak atılır. Bu da verilerin çekilmesi ya da yazılması hızına doğrudan etki etmekte ve veri güvenliğini sağlamaktadır. Verilerin kopyalar halinde farklı sunucular üzerinde tutulması veri güvenliğini sağlayan etkendir. Sunuculardan biri arızalandığında veri kaybı yaşanmayacak başka bir sunucuda olan veri kullanılabilecektir. Tüm bu işlemleri HDFS verilen konfigürasyon değerlerine göre otomatik olarak yapmaktadır.



Şekil 1.6. HDFS Veri Saklama

HDFS'e ait bazı komutlar aşağıdaki gibidir.

Klasör oluşturma: hdfs dfs -mkdir /user/data

Klasör içerisindekileri gösterme: hdfs dfs -ls /user

HDFS'e dosya koyma: hdfs dfs -put /home/suleyman/Desktop/jupyter.txt /user

HDFS'e dosya çekme: hdfs dfs -get /user/eğitim.csv /home/suleyman/Desktop

2.1.1.2. Apache Ambari

Büyük veri platformunun yönetim panelidir. Bu panel sayesinde uygulamaların konfigürasyonları, gerektiğinde servisleri durdurma, çalıştırma gibi işlemler ve HDFS kontrolü gibi işlemler kolaylıkla yapılabilmektedir. Konfigürasyonlarda yapılan değişiklikler Apache Ambari içerisinde versiyonlanarak tutulmakta, yapılan değişiklik geri alınmak istendiğinde kolayca gerçekleştirilebilmektedir.

2.1.1.3. Apache Kafka

Yapısal olmayan (unstructered) verilerin depolanabildiği platformdur. Gerçek zamanlı veri işlemede sıkça kullanılan bu platform verilen konfigürasyonlar ile kaç günlük veriyi topic içerisinde tutacağını ayarlayabilmektedir. Bu konfigürasyon sayesinde Kafka ortamına atılan veriler işlendikten sonra gereksiz yere sistemde yer kaplamayacaktır.

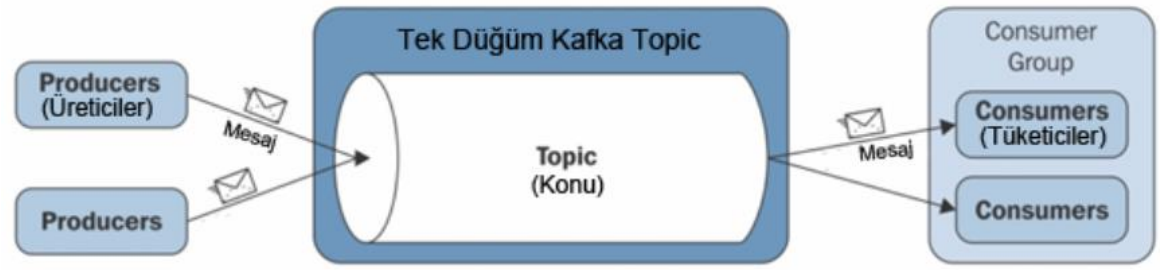
Kafka'da 3 yapı vardır;

Üreticiler-Producers: Verilerin Kafka Topic'ine gönderilmesini sağlayan yapılardır.

Konu-Topic: Kafka içerisine gönderilen verilerin tutulduğu depolama alanlarıdır.

Tüketiciler-Consumers: Verilerin Kafka topiclerinden çekilmesini sağlayan yapılardır.

Apache Kafka'nın mimari yapısı şekil 1.7'deki gibidir.



Şekil 1.7. Kafka Mimarisi

Apache Kafka'ya ait bazı komutlar aşağıdaki gibidir.

Topic Oluşturma: sudo bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic kafkaTopic

Topicleri Gösterme: sudo bin/kafka-topics.sh --list --zookeeper localhost:2181

Veri Gönderme Ekranı: sudo bin/kafka-console-producer.sh --broker-list localhost:6667 --topic kafkaTopic

Veri Çekme Ekranı: sudo bin/kafka-console-consumer.sh --bootstrap-server localhost:6667 --topic kafkaTopic --from-beginning

Topic Silme: sudo bin/kafka-topics.sh --delete --zookeeper localhost:2181 --topic kafkaTopic

2.1.1.4. Apache Storm

Dağıtık, hata toleransı olan, açık kaynaklı bir hesaplama sistemidir. Gerçek zamanlı akışları sağlayan yapılara topoloji denmektedir. Java ile kodlanarak kullanılan Storm gerçek zamanlı verilerin işlenmesinde yaygın olarak kullanılmaktadır. Çalışmasını Apache Hadoop üzerinde değil Apache Zookeper üzerinde gerçekleştirir ve iç koordinasyonunu sağlamak için kendi işçi-worker'larını kullanır.

2.1.1.5. Apache Zookeper

Apache Zookeper, servisleri koordine etmeye, konfigürasyonlarını düzenlemeye, dağıtık çalışan sistemleri senkron çalıştırmaya yarayan merkezi bir servistir.

2.1.1.6. Apache Spark

Apache Spark bellek içi-in memory çalışan bir sistemdir. İşlemleri bellek içerisinde yapmasından ötürü oldukça hızlıdır. Scala, Python, Java ve R dilleri ile geliştirme imkânı tanıyan Apache Spark, kendisine ait kütüphaneleri olmasından dolayı birçok alanda geliştirmeleri kolayca yapmaya olanak tanımaktadır. Örnek olarak makine öğrenmesi yöntemleri için MLlib, SQL sorguları için Spark SQL, ETL ve join tabanlı geliştirmeler için GraphX, gerçek zamanlı veri akışı için Spark Streaming kütüphaneleri gösterilebilir. Spark çalıştırılmak istendiğinde Scala kodlarıyla çalıştırılacaksa spark-shell --master local[2] komutu, Python kodları ile çalıştırılacaksa pyspark --master local[2] komutuyla çalıştırılmalıdır.

2.1.1.7. Apache Nifi

Gerçek zamanlı veri akışında yapısal ya da yapısal olmayan verilerin kaynaklardan çekilip işlenmesine imkân tanıyan veri işleme platformudur. Kaynaktan alınan veriler üzerinde ETL (Extract, Transform, Load) süreçlerini uygulayarak nihai verilerin işlenebilmesi ve depolanabilmesini sağlayan büyük veri uygulamasıdır. Ayrıca Apache Minifi adındaki alt ürünü sayesinde cihaz verilerinin gerçek zamanlı ve/veya gerçek zaman yakın şekilde toplanmasına ve işlenmesine olanak tanımaktadır.

Yukarıda başlıklar halinde anlatılan büyük veri uygulamaları veriyi farklı kaynaklardan alıp işlemeye olanak tanımaktadır. Kaynak olarak aldığı veriler yapısal, yapısal olmayan veya

gerçek zamanlı akan veriler olabilmektedir. Son zamanlarda kullanım alanı oldukça yaygınlaşan nesnelerin interneti-Internet of Things (IoT) farklı cihazların gerek otomatik şekilde gerekse internet üzerinden kontrol edilebilmesine denmektedir. IoT cihazlarından elde edilen verilerin gerçek zamanlı ya da gerçek zamana yakın şekilde büyük veri ortamlarına gönderilmesiyle büyük veri kümeleri oluşmaktadır. Bu veri kümelerinin işlenmesiyle hem donanım hem de yazılımı bütünleyici analitik sonuçlar elde edilmektedir. IoT teknolojisine örnek olarak yaygın olarak kullanılan oto kiralama araçları verilebilir. Yapılan bazı çalışmalar ile bu araçlara entegre edilen cihazlar araba beyninden çektiği sensör verilerini kiralama şirketlerine iletmektedir. Bu sayede kişilerin kullanım durumlarına göre kaza risk faktörleri belirlenip, kişi bazlı oto kiralama fiyatları uygulanabilmektedir.

İşlenen ve depolanan verilerin anlamlandırılması verinin değerini ve işlevselliğini arttıran bir unsurdur. Sahip olunan verilerden sonuçlar çıkarmak, çıkarılan sonuçları süreci iyileştirmede kullanmak günümüzde yaygın olarak kullanılmaktadır. Birçok firma ve kuruluş verileri analiz etmek ve anlamlandırmak için raporlar ve dashboardlar kullanmaktadır. Fakat bu yöntem ile verilerin analiz edilmesi ve anlamlandırılması kullanıcıların dikkatlerine ve iş bilme seviyelerine bırakılmıştır. Verinin anlamlandırılmasında yenilikçi ve büyük ölçekli firmalar tahminleme yapılarına başvurmuş, standart raporlama ve dashboardlar ile analizden ziyade veri madenciliği yöntemleri ile modeller oluşturmakta ve oluşturulan modellere sokulan veriler ile durum analizlerini yapabilmektedir. Ardından üretilen anlamlı sonuçları raporlar ve dashboardlarda göstermektedir. Bu durum insan faktörüne düşen iş yükünü hafifletme, hata payını azaltmakta ve daha güvenilir sonuçlar elde edilmesini sağlamaktadır. Bu nedenledir ki veri madenciliği yöntemleri ve algoritmaları şirketler ve kuruluşlar açısından oldukça öneme sahip bir olgudur.

Raporlar ve dashboardlar, verilerin daha düzenli ve anlaşılır bir şekilde kullanıcılara gösterilmesine olanak tanır. Bu sayede eldeki verilerden hızlı, kolay ve anlaşılır analizler çıkarılmasını sağlar. Raporlar verilerin karakter ve nümerik değerlerle gösterilmesi esasına dayanırken, dashboardlar daha çok görselleştirme üzerine dayalıdır. Raporlama ve dashboard ürünleri kabiliyetlerine bağlı olarak desktop, web ve mobil uygulamalarında gösterilebilmekte, ayrıca istenen dosya uzantılarına göre çıktılar verebilmektedir.

Raporlar gelişmekte olan ve gelişmiş şirketlerde aktif kullanılmakta, raporları üreten ilgili birimler genelde iş zekası olarak adlandırılmaktadır. İş zekası firmaların gelişimlerine, verilerinin kolay takibiyle beraber doğru kararlar verilmesine olanak tanır.

1.1.3. Literatür incelemesi

- 2012 yılında diyabet hastalarında 24 saatlik kan basıncı değişiminin öngörülmesine yönelik bir çalışma gerçekleştirilmiştir. Bu çalışmada bireylerden toplanan veriler üzerinde WEKA platformunda Karar Ağaçları, Naive Bayes, Yapay Sinir Ağları algoritmaları kullanılarak dipper/non-dipper patternleri tahminlenmiştir. Yapay Sinir Ağlarının doğruluk oranı en yükseksonucu veren algoritma olduğu sonucu çıkarılmıştır [4].
- 2016 yılında Apache Spark Streaming ürünü kullanılarak gerçek zamanlı (real-time) veri akışı sağlanmış, akan veriyi Apache Spark MLlib kütüphaneleri ile sınıflandırma işlemlerine sokulmuştur. Apache Spark kullanılarak Destek Vektör Makinesi algoritması ile sınıflandırma yapılmış ve hali hazırda SAS ürünü kullanılarak yapılmış olan Logistik Regresyon algoritması ile performans gözlemlemesi yapılmıştır. İki algoritmanın karşılaştırılmasında Destek Vektör Makinesi algoritması daha başarılı sonuçlar vermiştir [5].
- 2016 yılında gerçek zamanlı veri akışının Apache Spark, Storm, Flink uygulamalarındaki hızları karşılaştırılmıştır. Elde edilen Sonuçlarda Storm'un daha fazla işlemci kullandığı ve hata yakalama özelliği açıldığında neredeyse %50 performans kaybı yaşandığı, Apache Spark'ın performansında etken olarak işleme başlama süresi ve paralel çalıştırma Kafka bölümleriyle ilintisi, Flink'te ise en fazla veri işleme doygunluğunun yakalanmış olması gözlemlenmiştir [6].
- 2018 yılında yapılan bir çalışmada Apache Spark, Flink ve Apache Hadoop performansları karşılaştırılmış, Apache Spark ve Flink'in performansının Apache Hadoop'a göre daha fazla olduğu gözlemlenmiştir. Elde edilen sonuçlara göre kelime saymada, sıralama yapmada Apache Spark ve Flink Hadoop'tan çok daha kısa sürede işlemi tamamlamıştır. Okuma ve sıralamada Flink ve Apache Spark birbirlerine yakın sürelerde süreçleri tamamlarken, K-Ortalamalar (K-Means) tahminleme algoritması çalışma süresinde ise Apache Spark Flink'e göre daha hızlı sonuçlar döndürmüştür [7].

- 2018 yılında yapılan bir çalışmada sağlık turizmi ve özellikle saç ekiminde büyük verinin önemi incelenmiş bulunan sonuçlar saç ekimi merkezlerinde büyük veri ve teknolojilerinin yeterince kullanılmadığı tespit edilmiştir. Saç ekim merkezi hizmetlerini geliştirmek ve yeni müşterilerin kazanımı için Facebook analizi ve Google Adwords gibi temel veri analizi tekniklerinin kullanıldığı görülmektedir [8].
- 2018 yılında yapılan bir çalışmada Apache Spark Streaming ve MLlib kullanarak gerçek zamanlı gelen hasta verilerinden göğüs kanserini tahminleme üzerine çalışılmıştır. Bu veriler özellikle karar ağaçları algoritması ile işlenmiş ve sağlık durumları hakkında tahminleme yapılması sağlamıştır [9].
- 2018 yılında havayolu verileri üzerinde Apache Hadoop, Apache Spark ve Makine öğrenmesi algoritmaları ile bir çalışma gerçekleştirilmiştir. Makine öğrenmesinde denetimli ve denetimsiz öğrenme algoritmaları arasındaki farklılıklar çıkarılmıştır [10].
- 2018 yılında yayınlanan bir makalede Hadoop'un özelliklerinin daha da geliştirilmesini sağlayan üreticilerin açık kaynak kodlu ürünleri incelenip bu ürünler arasında karşılaştırma yapılmıştır [11].
- 2018 yılında yapılan bir çalışmada Apache Spark Streaming ve MLlib kullanarak gerçek zamanlı (real-time) gelen hasta verilerinden göğüs kanserini tahminleme üzerine çalışılmıştır. Bu veriler özellikle karar ağaçları algoritması ile işlenmiş ve sağlık durumları hakkında tahminleme yapılması sağlamıştır [12].
- 2018 yılında yapılan çalışmada bireylerin tiroid hastalığının olup olmadığı tahminlenmiştir. Bu çalışmada 747.301 kayıt kullanılmış. Bu kayıtlardan 13 öznitelik baz alınarak derin öğrenme algoritmalarıyla işlenmiştir. Derin Yapay Sinir Ağları - Deep Neural Network (DNN) algoritması Random Forest algoritmasına göre %36.63 daha fazla doğruluk üretmiştir [13].
- 2018 yılında yapılan başka bir çalışma ise gelişmiş ülkelerde yalnızca hastaların %50'si tedavilerini doğru şekilde takip ettiğini belirtmekte ve göğüs kanserinin tedavisinde ilaç tüketiminin farklı modellerinin kullanılması tavsiye edilmektedir. Onkologlarla yürütülen bu ortak çalışma neticesinde makine öğrenmesi algoritmaları ile risk faktörlerinin hesaplanması yapılmıştır [14].

- 2019 yılında yapılan bir çalışmada büyük veri platformlarında gerçek zamanlı-real time akış kullanılarak bilgi güvenliğini sağlamak adına bir çalışma gerçekleştirilmiştir. Yapılan bu çalışmada hızlı şekilde gelen VoIP (Voice Over IP) verileri tespit edilerek güvenliğin sağlanması amaçlanmıştır [15].
- 2019 yılında yapılan bir çalışmada filo araçlarının ürettiği verileri büyük veri ortamlarında tutulması ve gerçek zamanlı işlenmesi üzerine çalışılmıştır. Yapılan bu çalışmada ayrıca ilişkisel veritabanı ve büyük veri ortamındaki platformların karşılaştırılması sağlanmıştır [16].

Gerçekleştirilen bu tez çalışmanın benzer çalışmalara göre farklılıkları aşağıda listelenmiştir.

- Bu ve benzer projelerin yaygın kullanımı durumunda gelebilecek büyük veriyi kaldıracak platform HİPAM için sağlanmıştır.
- İlerleyen aşama TKV ve HİPAM arasında real-time veri akışı sağlanmak istenebileceği düşünülerek, HİPAM için bunu sağlayacak altyapı oluşturulmuştur.
- Bu çalışmada Apache Spark ürünü kullanılarak tahminleme sonuçları üretilmiştir.
- Benzer projelerde sadece sonuçlar gösterilmişken, bu çalışmada iş zekâsı ürünleriyle son kullanıcıya gösterilecek raporlar oluşturulmuştur.

Bu tez çalışmasının iş akışı, kullanılan ürünler ve yapılan çalışma, Materyal ve Yöntem bölümü altında detaylandırılarak anlatılmıştır.

2. MATERYAL VE YÖNTEM

Bu bölümde öncelikli olarak tez çalışmasının genel hatlarıyla iş akış diyagramı Şekil 2.1'de sunulmuş, peşinden kullanılan ürünler ve kullanım amaçları maddelenmiştir (Şekil 2.1).

Şekil 2.1. İş Akış Diyagramı

HİPAM ve TKV arasında imzalanan protokolden mütevellit verilerin yapısal bir veritabanı (DB)'de tutulması beklenmektedir. Gelecekte çalışmanın kapsamının genişlemesi, hasta verilerinin artması, proje kullanıcılarının artması ve sonradan büyük veri alanında kendini geliştirmek isteyen öğrencilerin bu platformu baz alarak çalışmalarını ilerletebilmesi gibi etkenlerin oluşturduğu altyapı ile projeye büyük veri makinesi dahil edilmiş olup ilgili verilerin bu makinede tutulması gerektiğine karar verilmiştir. Büyük veri makinesinin tercih

edilmesinin projenin seyrine birkaç olumlu katkısı bulunmaktadır. Bunlar aşağıda listelenmiştir.

- Büyük veri ortamının esnekliği sayesinde proje istenirse real time sonuçlar üretebilecek hale de dönüştürülebilecektir.
- Büyüyen veri miktarına hızla cevap verilebilmesi için mevcut sisteme yeni makine eklenip performans ve kapasite artırımı sağlanabilecektir.
- Sisteme real time gönderilen verilerden real time sonuçların üretilmesi mümkün kılınacaktır.
- Farklı kaynaklardan gönderilecek farklı türdeki veri tiplerinin sisteme entegrasyonu için gereken çeşitlilik sağlanabilmektedir. Örneğin yapılandırılmamış formatta gelen Excel dosyaları, csv dosya türleri ya da yapılandırılmış gelen verilerin işlenmesi mümkün olacaktır.
- Farklı projelerde kullanılacak bir altyapı sağlanmış olduğundan ilgili verilerin aynı ortamda olmasından ötürü birbirlerine entegrasyonu yapılarak verimli projelerin ortaya çıkarılması mümkün olacaktır.

Projede kullanılan ürünler ve nasıl kullanıldıkları aşağıda maddelenmiştir:

- **Microsoft AccessDB:** İlişkisel **(**Relational) DB'dir.
- **MariaDB:** Büyük veri ortamında bulunan ilişkisel DB'dir. MySQL'in büyük veri ortamındaki karşılığı da denebilir. MySQL IDE kullanılarak kolaylıkla kontrol edilebilir.
- **Java:** Dünya genelinde yaygın olarak kullanılan programlama dilidir.
- **Scala:** Kendine özgü hazır kütüphanelerinin olmasından dolayı kolay kodlama sağlayan programlama dilidir.
- **Hortonworks:** Büyük veri uygulamalarının tek çatı altında toplandığı sağlayıcı firmadır. Büyük veri sektöründe ismi fazlaca duyulan iki firmadan biridir. Yakın zamanda rakibi olan Cloudera ile yollarını birleştirerek ürünleri birlikte çıkaracaklarını açıklamışlardır. Ürünlerinin kullanımların da ücret talep etmemektedirler fakat destek istendiğinde ücrete tabii tutmaktadırlar.
- **Apache Spark:** In Memory çalışan işleme motorudur. Verileri işlerken memory içerisinde işlemesinden ötürü yüksek hızlar sunabilmektedir. R, Scala, Python ve

Java dillerini desteklemektedir. Kendine özgü SQL, Machine Learning ve görselleştirmeye yönelik kütüphaneleri vardır.

- **Microsoft PowerBI:** Raporlama ve dashboard oluşturma ve görüntüleme ürünüdür.

Büyük veri ortamında birden çok ilişkisel DB'ler olmasına karşın, MariaDB'de MySQL IDE'si kullanılabildiğinden ve farklı uygulamaların kendi içlerinde bağlantı için gerekli olan MariaDB'ye ait driver'larını bulundurmasından ötürü, bu DB tercih edilmiş ve kullanılmıştır.

Bu tez çalışmasında, tıp uzmanlarının istek ve önerileri doğrultusunda oluşturulan veriseti üzerinde tahminleme analizinin yapılması sağlanmıştır. Bu bağlamda, büyük veri platformunda bulunan Apache Spark ürünü üzerinde Scala kodlarıyla geliştirme yapılmış olup, Naive Bayes ve Random Forest algoritmaları kullanılarak tahminleme modelleri oluşturulmuştur. Elde edilen doğruluk sonuçlarının kullanıcılara gösterimi için raporlama aracına başvurulmuştur. Projede raporlama, Microsoft PowerBI iş zekası ürünü kullanılarak sağlanmıştır. Bu ürün sayesinde kullanıcılar verileri ve sonuçları daha anlaşılır bir şekilde web ve mobil platformu üzerinden görebilmekte, analiz edebilmektedirler.

Şekil 2.2. Temel Çalışma Prensibi

Projenin temel çalışma prensibi Şekil 2.2'de gösterilmiştir (Şekil 2.2). Hastalardan toplanan veriler TKV tarafından alınmakta ve HİPAM büyük veri makinesine koyulmaktadır. Sonrasında raporlar son kullanıcıya gösterilmektedir. Bu amaç için kullanılan makinenin donanım özellikleri şu şekildedir:

- AMD Ryzen 7 1800X Eight Core Processor 3.6 GHz
- 16 GB DDR4 RAM
- NVIDIA GeForce GTX 1050 Ti
- 500 GB SSD
- 2 TB HDD

2.1. Büyük Veri Platform Kurulumu

Büyük veri platformu kurulması ile ilgili yapılan çalışmalar alt başlıklar halinde sırasıyla açıklanmaya çalışılmıştır.

2.1.1. Secure shell (SSH) kurulumu

SSH bilgisayarlar arasında güvenilir iletişim kurulmasına yarayan sistemdir. Ambari Server'ın tüm cluster'lara ambari agent'ları otomatik bir şekilde yükleyebilmesi için Ambari Server Host ve cluster içindeki diğer host'lara şifresiz SSH bağlantısı kurulmalıdır. Bu sayede Ambari Server Host uzaktan erişim ve Ambari Agent'ın kurulumu için SSH public key authentication'ı kullanır.

Adımlar:

1. Umumi ve özel SSH anahtarlarının Ambari Server Host'ta oluşturulması.

```
ssh-keygen
```

2. SSH Public Key (id_rsa.pub) host'lardaki root hesabına kopyalanır.

```
.ssh/id_rsa
.ssh/id_rsa.pub
```

3. Oluşturulan key Host'lardaki authorized_keys dosyası içerisine eklenir.

```
cat id_rsa.pub >> authorized_keys
```

4. SSH versiyonuna bağlı olarak, authorized_keys'e yetkilendirme verilmesi gerekebilir.

```
chmod 700 ~/.ssh
chmod 600 ~/.ssh/authorized_keys
```

5. Ambari Server'dan clusterdaki her bir host'a SSH ile şifresiz olarak bağlanabiliyor olunduğu kontrol edilir.

```
ssh root@bigdata.marmara
```

6. SSH Private Key'in bir kopyası Browser'dan yapılacak olan Ambari Install Wizard için tutulur.

2.1.2. Network time protokol (NTP) aktifleştirme

NTP birbirleriyle paket alışverişi yapan sunucuların aynı saati kullanabilmesine yarar. Cluster içindeki tüm node'ların ve Ambari Web Interface'e bağlanmak için kullanılan makinenin saatleri birbirleriyle senkron olmalıdır. Bu sebepten dolayı NTP aktifleştirilir.

```
yum install -y ntp
systemctl enable ntpd
```

2.1.3. Host file düzenleme

Erişilmek istenen bir makineye/IP'ye, host dosyaları içerisinde isimlendirme yapılabilir. Bu sayede IP yerine tanımlanan alias ile istenen makineye erişilebilir. Localhost IP'si olan 127.0.0.1'e yani local makineye bigdata.marmara tanımı yapılmıştır.

1. Cluster içindeki tüm hostlarda Host File

```
vi /etc/hosts
```

2. İkinci cluster içindeki her bir host için kendi IP'lerine göre giriş yapılmalıdır. Tek cluster makine olduğundan local IP kullanılmıştır.

```
127.0.0.1 bigdata bigdata.marmara
```

3. Kontrol etmek için;

```
Hostname -f
```

2.1.4. Ağ ayarlarının yapılandırılması

1. Her bir host'ta network ayarları açılır.

```
vi /etc/sysconfig/network
```

2. Ayarlar aşağıdaki şekilde değiştirilir/eklenir.

```
NETWORKING=yes
HOSTNAME=bigdata.marmara
```

2.1.5. Iptables ayarlaması

Iptables Linux işletim sistemlerindeki network Firewall'dır. Firewall'lar bilgisayarların güvenliğini sağlamak amacıyla kullanılmaktadırlar. Dışarıdan gelen isteklerin filtrelenerek geçirilmesini sağlarlar. Ambari kurulumu sırasında bazı portların kesin bir şekilde açık olması gerekmektedir. Bunun için Firewall kapatılarak olası kesintilerin önüne geçilir.

```
systemctl disable Firewalld
service Firewalld stop
```

2.1.6. SELinux ve PackageKit kapatma

Selinux bilgisayardaki uygulamalara, sistem servislerine, dosyalara ve ağ kaynaklarına erişimi kısıtlayan bir protokoldür. Bu protokol inaktif edilerek olası çıkabilecek sistemsel sorunların önüne geçilir.

1. Ambari setup fonksiyonlarının sorunsuz çalışabilmesi için cluster'daki her bir host'ta SELinux deaktif edilir.

```
setenforce 0
```

2. Text editör ile /etc/yum/pluginconf.d/refresh-packagekit.conf açılır ve aşağıdaki şekilde komut değiştirilir.

```
enabled=0
```

3. UMASK yetkilendirmesi yapılır.

```
umask 0022
```

4. UMASK'ı kontrol etmek için kullanılır.

```
umask
```

5. Tüm interaktif kullanıcıları değiştirmek için aşağıdaki komut kullanılır.

```
echo umask 0022 >> /etc/profile
```

2.1.7. Ambari repository kurulumu

Kurulum sırasında gerekli dosyaların internetten indirilebilmesi için sunucunun internete bağlı olması gerekir.

1. Terminalde root hesabına geçiş yapılır.

```
Su -
```

2. Ambari repository file aşağıdaki komut ile sunucuya indirilir.

```
wget -nv http://public-repo-1.hortonworks.com/ambari/centos7/2.x/updates/2.7.3.0/ambari.repo -O /etc/yum.repos.d/ambari.repo
```

3. Repolist kontrol edilir.

```
yum repolist
```

2.1.8. Ambari server kurulum

Aşağıdaki komutla Ambari server yüklenir. Bu komut ayrıca default olarak PostgreSQL Ambari database olarak da yüklenecektir.

```
yum install ambari-server
```

2.1.9. Ambari server ayarlar

Kurulumdan sonra Ambari Server'a ait konfigürasyon ve JDK, Database gibi gerekli uygulamaların kurulması ve/veya atamalarının yapılması gerekmektedir. Kurulumda JDK Versiyonu olarak default olan JDK 1.8 versiyonu, devam edilecek kullanıcı olarak root, DB olarak mariaDB seçilmiştir.

Sunucuya uzaktan erişebilmek için sunucunun bağlı olduğu modem üzerinde ayarlamalar yapılması gerekmektedir. Bu ayarlar sayesinde lokal ağ dışından da sunucuya erişim mümkün olmaktadır. Bu konunun detayları EK-1 içerisinde verilmektedir.

2.1.10. Büyük veri makinesine erişim izninin verilmesi

Port yönlendirmeler ile internetten gelen bağlantı isteklerinin Firewall'a takılmadan alınabilmesi için CentOS üzerinden gelen isteklere yetkilendirme yapılması gerekmektedir.

Bu yetkilendirmenin yapılmaması durumunda Firewall bilgisayarın güvenliğini sağlamak adına gelen isteğe cevap vermeyecektir.

Firewall izninin belirli bir IP'ye verilmesi için aşağıdaki komut kullanılır.

```
iptables -A INPUT -i eth0 -s 192.168.1.6 -p tcp --destination-port 8888 -j ACCEPT
```

Tüm IP'lerden gelen isteklere izin vermek için ise IP yazmaksızın komut çalıştırılmalıdır.

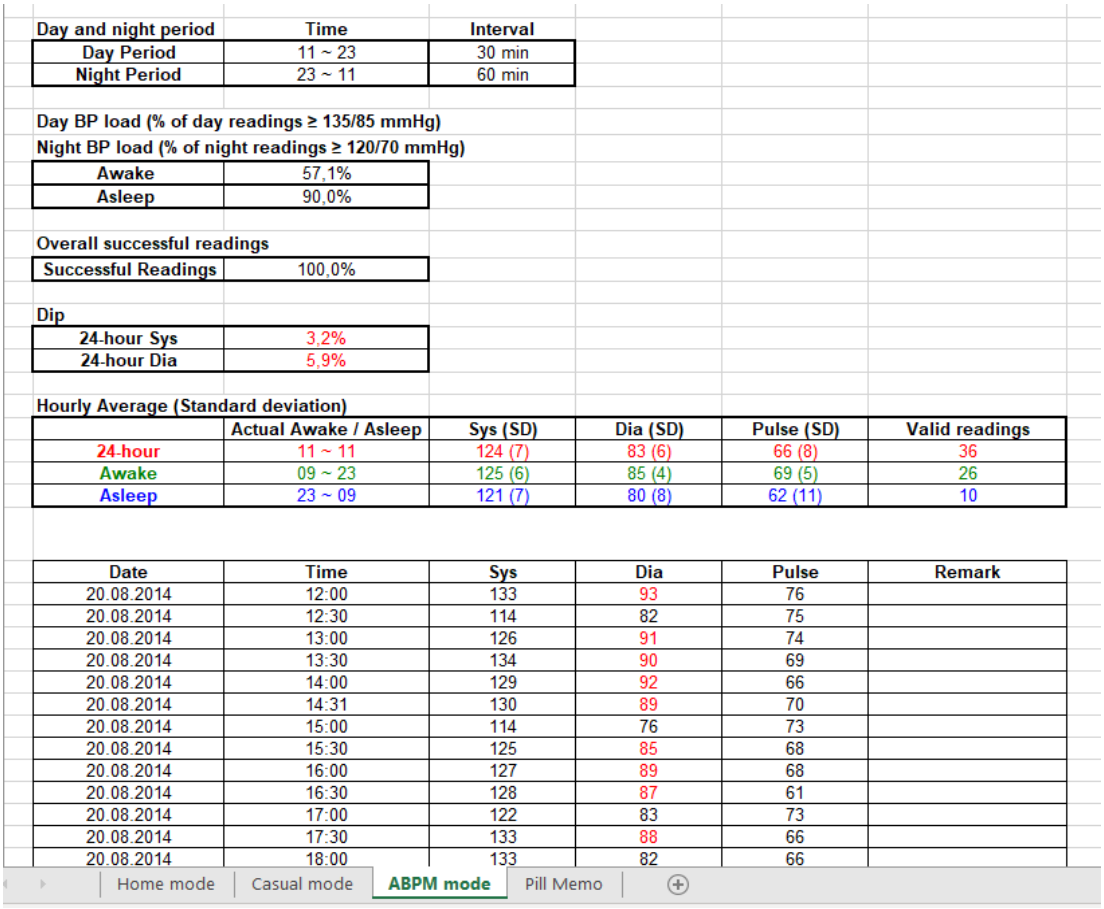
```
iptables -A INPUT -i eth0 -p tcp --destination-port 8888 -j ACCEPT
```

2.2. Harici Verinin Temizlenmesi

Proje kapsamında kullanılacak veriler hem Excel ve hem de Access platformları kullanarak alınmıştır. ABPM cihazı Excel verileri yapılandırılmamış yapıda olduğundan önce verinin temizlenmesi, ardından yapılandırılmış yapıya dönüştürülmesi gerekmektedir. Temizlenen veri Access platformuna alınmış ve veri bütünlüğü sağlandıktan sonra MariaDB veritabanına alınmıştır. Aşağıda veri temizleme ve işleme adımları yer almaktadır.

2.2.1. Java ile Excel'lerin birleştirilmesi

İşlenecek olan Excel dosyaları her bir birey için ayrı ayrı gelmektedir ve içerisinde ABPM verilerini tutmaktadır. Gelen dosyada bireye ait sağlık verileri yapılandırılmamış olarak yer almakta ve bu düzensiz verilerin işlenerek tek satırda yazılması gerekmektedir (Şekil 2.3).



Day and night period	Time	Interval
Day Period	11 ~ 23	30 min
Night Period	23 ~ 11	60 min

Day BP load (% of day readings ≥ 135/85 mmHg)
Night BP load (% of night readings ≥ 120/70 mmHg)

Awake	57,1%
Asleep	90,0%

Overall successful readings

Successful Readings	100,0%

Dip

24-hour Sys	3,2%
24-hour Dia	5,9%

Hourly Average (Standard deviation)

	Actual Awake / Asleep	Sys (SD)	Dia (SD)	Pulse (SD)	Valid readings
24-hour	11 ~ 11	124 (7)	83 (6)	66 (8)	36
Awake	09 ~ 23	125 (6)	85 (4)	69 (5)	26
Asleep	23 ~ 09	121 (7)	80 (8)	62 (11)	10

Date	Time	Sys	Dia	Pulse	Remark
20.08.2014	12:00	133	93	76	
20.08.2014	12:30	114	82	75	
20.08.2014	13:00	126	91	74	
20.08.2014	13:30	134	90	69	
20.08.2014	14:00	129	92	66	
20.08.2014	14:31	130	89	70	
20.08.2014	15:00	114	76	73	
20.08.2014	15:30	125	85	68	
20.08.2014	16:00	127	89	68	
20.08.2014	16:30	128	87	61	
20.08.2014	17:00	122	83	73	
20.08.2014	17:30	133	88	66	
20.08.2014	18:00	133	82	66	

Home mode | Casual mode | ABPM mode | Pill Memo

Şekil 2.3. Gelen ABPM Verisi

Bunun için Java dilinde bir program yazılmış olup ilgili hücredeki verilerin satır olarak yazılması sağlanmıştır. Yazılan program sayesinde veriler yapılandırılmamış yapıdan yapılandırılmış yapıya dönüştürülmesi sağlanmıştır. Ardından düzenlenen bu veriler Access veritabanına alınmıştır.

2.2.2. Access veritabanı işlemleri

Hastaneye gelen bireylere ait kimlik bilgileri, hastaneye geliş tarihi gibi bilgiler Access veritabanı ile gelmektedir. Gelen bu veriler içerisinden hastaya ait kimlik bilgileri yazılan SQL sorgusu ile alınmıştır. Alınan bireylerin kimlik bilgileri ve Excel'den Access'e import edilen ABPM verileri büyük veri ortamında bulunan MariaDB'e aktarılmıştır.

2.3. MariaDB Veritabanı İşlemleri

Aşağıdaki başlıklarda verilerin DB'ye alınması süreçleri anlatılmıştır.

2.3.1. Şema oluşturulması

MariaDB'de "Marmara" adında şema oluşturulur. İlgili tablolar bu şema altında oluşturulacaktır.

2.3.2. Nihai tabloların oluşturulması

MariaDB'e alınacak verilerin tutulacağı tablolar gelen veri yapısına uygun olacak şekilde belirlenmiş, kolon adları ve tablo adları da belirli bir standarta göre create edilmiştir. Bu standarta göre kaynaktan ilk alınan tablolara FT, kullanılacak olan nihai fact tablolara "TR", lookup tablolara "LU", view'lere "VP" ile tablo içerisindeki id alanlarına "ID", detail alanlarına "DS" ve metrik alanlarının başlarına "MT" prefix'i kullanılmıştır. Bu standart, sonradan tablolara bakacak kişilerin işlerini kolaylaştıracak, mevcutta aradığı ya da baktığı kolonun yapısı hakkında fikir sahibi olmasına yardımcı olacaktır.

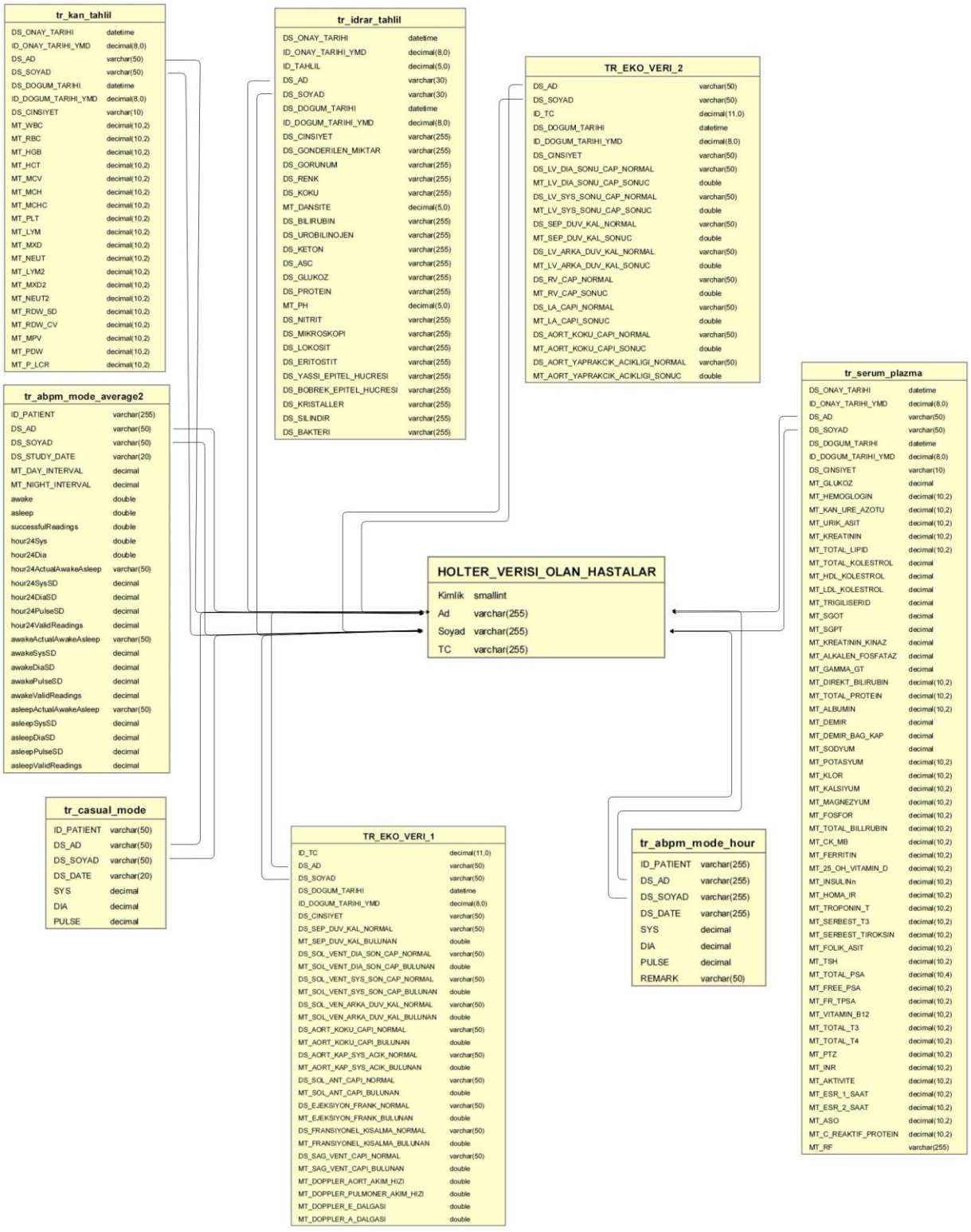
Tablo 2.1 Tablo Adları

Tablo Adı	Açıklama
ft_abpm_mode_average	24-saatlik ortalama tansiyon verisini içerir.
ft_abpm_mode_hour	Saatlik tansiyon verisini içerir.
ft_casual_mode	Gündelik tansiyon verilerini içerir.
ft_idrar_tahlil	İdrar verilerini içerir.
ft_kan_tahlil	Kan tahlili verilerini içerir.
ft_serum_plazma	Serum Plazma verilerini içerir.
ft_eko_veri1	Eko verilerini içerir.
ft_eko_veri2	Eko verilerini içerir.
lu_icd	İcd lookup tablosu
aahipam	Veriseti-2 verilerini içerir.
lu_patients	Hasta bilgilerini içeren lookup tablosu
tr_abpm_mode_average	24-saatlik ortalama tansiyon verisini içerir.
tr_abpm_mode_hour	Saatlik tansiyon verisini içerir.

Tablo 2.2. Tablo Adları Devam

Tablo Adı	Açıklama
tr_casual_mode	Gündelik tansiyon verilerini içerir.
tr_hasta_veri	Hastalara ait tüm verileri içerir.
tr_idrar_tahlil	İdrar verilerini içerir.
tr_kan_tahlil	Kan tahlili verilerini içerir.
tr_serum_plazma	Serum Plazma verilerini içerir.
tr_eko_veri1	Eko verilerini içerir.
tr_eko_veri2	Eko verilerini içerir.
tr_prediction_naive	Veriseti-1 Naive Bayes Prediction Sonuç Verilerini İçerir.
tr_prediction_rf	Veriseti-1 Random Forest Prediction Sonuç Verilerini İçerir.
tr_prediction_naive_80	Veriseti-2 Naive Bayes Prediction Sonuç Verilerini İçerir.
tr_prediction_rf_80	Veriseti-2 Random Forest Prediction Sonuç Verilerini İçerir.

Tablolar arasındaki bağlantı Birleşik Modelleme Dili – Unified Modeling Language (UML) diyagramı yardımıyla Şekil 2.4'te gösterilmiştir.



Şekil 2.4. Tablolar arasındaki ilişki (UML Diyagramı)

2.3.3. Viewlerin oluşturulması

View'ler, veritabanlarında fiziksel tablolarda bulunan verilerin istenilen formatlarda gösterilmesini sağlayan ara katmanlardır. Kendileri fiziksel bir tablo yapısında değildir, yani içlerinde herhangi bir veri barındırmazlar.

Tablo 2.3 View Adları

View Adı	Açıklama
vp_tr_hasta_veri	tr_hasta_veri tablosundaki verileri içerir.
vp_tr_select_all	Ayrı tablolarda bulunan hastalara ait tüm verileri içerir.
vp_tr_training_data	Prediction için kullanılacak ilgili kolonları içerir.
vp_tr_prediction_naive	tr_prediction_naive verilerini içerir.
vp_tr_prediction_rf	tr_prediction_rf verilerini içerir.
vp_tr_prediction_naive_2	tr_prediction_naive_80 verilerini içerir.
vp_tr_prediction_rf_2	tr_prediction_rf_80 verilerini içerir.
vp_tr_80_kisilik_veri	aahipam verilerini içerir.

2.3.4. Verilerin MariaDB'ye alınması

Veriler iki yöntem ile MariaDB veritabanına alınmıştır. Kaynaktan alınan (işlenmemiş, düzenlenmemiş veriler) FT prefix'li tablolar altında tutulmaktadır. Gerekli düzenleme ve dönüşümler yapılarak TR'lı tablolara aktarılmıştır. Bu işlemlere ait adımlar aşağıda başlıklar halinde verilmektedir.

2.3.4.1. Java uygulaması ile MariaDB'e veri aktarımı

Java uygulaması yardımıyla birleştirilen Exceller aynı zamanda MariaDB veritabanına da insert sorguları ile eş zamanlı olarak gönderilmiştir. Bu işlem neticesinde hem Excel çıktısı hem de MariaDB insert işlemi yapılmaktadır.

2.3.4.2. Access veritabanından MariaDB'e veri aktarımı

Hali hazırda tek Excel altında gelen verilerin MariaDB'e alınması işlemi ise öncelikle Access veritabanına Excel'lerin import edilmesi ardından MariaDB'e export edilmesiyle gerçekleşmiştir. MariaDB'e harici veri kaynağı olarak csv ve json formatları ile veriler

alınabilmektedir. Gelen Excel verilerinin bazılarında ondalık ayraç "." olması gerekirken "," kullanıldığı için csv formatına dönüştürülerek almak tercih edilmemiş, bunun yerine önce Access veritabanına Excel dosyaları alınıp ardından MariaDB'e aktarılması uygun görülmüştür.

2.3.4.3. Veritabanındaki veriyi temizleme işlemi

Kaynaklardan yaklaşık olarak 1.110.000 kayıt büyük veri makinesine alınmıştır. MariaDB'ye alınan bu veriler FT prefixli tablolarda tutulmaktadır. FT içerisinde kolon tiplerine ve büyüklüklerine bakmaksızın varchar (255) olarak tutulan bu verilerin, veri tip ve boyutlarına göre ayrıştırılıp kirli veriler temizlendikten sonra TR prefix'li tablolara temiz şekilde aktarılması yapılmıştır.

Veri temizleme işlemlerinde karşılaşılan zorluklardan bazı örnekler aşağıdaki gibidir:

- Numerik alanların ondalık ayrımları bazı verilerde "." bazı verilerde "," kullanılarak sağlanmıştır. MariaDB'de ondalık ayrımlar sadece nokta olacağından veri içerisindeki virgüllü alanlar nokta işareti ile replace edilmiştir.
- Bazı tarih alanlarının başlarında boşluk karakteri olduğundan bu alanlar MariaDB tarafından date alanı gibi algılanamamış bunun için TRIM fonksiyonu kullanılarak boşluklar kaldırılmıştır.
- Bazı tablolarda ad ve soyad bilgileri tek bir alan içerisinde geldiğinden, bunların SQL fonksiyonu yardımıyla ad ayrı bir kolonda, soyad ayrı bir kolonda olacak şekilde düzenlemeye gidilmiştir.
- Bazı nümerik alanlar içerisinde üst ve alt limitlerinde küçük ("<") ve büyük (">") işaretleri kullanılmış bu alanlar verilerin numerik alan içerisine alınmasına engel teşkil ettiğinden kaldırılarak ilgili alana alınmıştır.
- Bazı nümerik alanlar içerisinde gelen yazılar null olarak değiştirilerek içeri alınmıştır.
- Bazı alanlar Excel dosyasında görünüş olarak doğru ama veri içeriği olarak hatalı durmaktadır. Verisel olarak 1.01.2019 olan bir kayıt Excel içerisinde belirlenen biçimden ötürü 1.1 değerini göstermektedir fakat veritabanına alınırken tarih olarak alındığından verisel soruna neden olmaktadır. Bu alanlar tek tek Excel içerisinde yeniden düzenlenmiştir.

2.4. Tahminleme

Büyük veri platform ürünü olan Apache Spark'ın farklı yazılım dillerinde geliştirme yapma imkânı sunmaktadır. Bunlardan en bilinen ve yaygın olanları Scala, Python ve Java'dır. Bu üç programlama diliyle de geliştirme yapılabilmesine karşın kütüphanelerinin zengin olmasından ötürü Scala ve Python dilleriyle daha basit bir şekilde istenen sonuçlara ulaşılabilmektedir. Birbirine oldukça yakın olan bu iki dilden Scala dili proje genelinde kullanımı tercih edilmiştir. Projede kullanılacak verisetine ait işlemler Veriseti Tanıtımı başlığı altında model oluşturma ve algoritmalara ait detaylar Model Oluşturma başlığı altında detaylandırılmıştır.

2.4.1. Veriseti tanıtımı

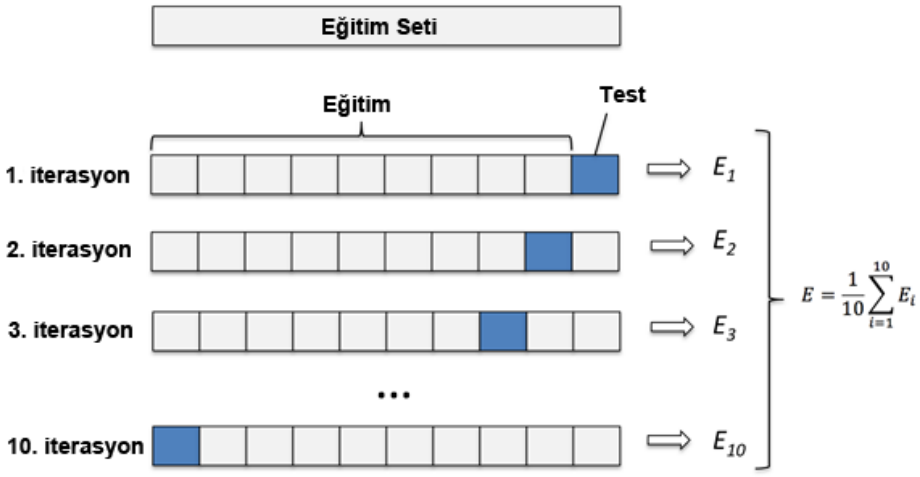
Veri madenciliği çalışmalarında gelen veri her ne kadar yapılandırılmış bir veri olsa da düzenlemelere tabiidir. Veri madenciliği çalışmasının verisel altyapısını oluşturmak için büyük veri platformundaki MariaDB veritabanına bağlanılmış, tahminlemede kullanılacak olan vp_tr_training_data view'inden ilgili veriler çekilip Scala değişkenlerinde tutulmuştur. Veri madenciliği algoritmalarına nümerik olmayan veriler girdi olarak verilememektedir. Bu sebepten ötürü yaş aralığı, cinsiyet ve dipper/non-dipper paternini gösteren alanlarda indeksleme yoluna gidilmiş nümerik değerler ile işleme alınması sağlanmıştır. İndekslemenin nasıl olduğuna dair görünüm Şekil 2.5'te gösterilmektedir.

YAS_ARALIGI	MT_YAS_ARALIGI_INDEX	DS_CINSIYET	MT_DS_CINSIYET_INDEX	SONUC	SONUC_INDEX
41-65	0.0	K	0.0	DIPPER	0.0
0-40	2.0	E	1.0	DIPPER	0.0
0-40	2.0	E	1.0	DIPPER	0.0
0-40	2.0	K	0.0	DIPPER	0.0
41-65	0.0	E	1.0	DIPPER	0.0
41-65	0.0	E	1.0	DIPPER	0.0
0-40	2.0	K	0.0	DIPPER	0.0
41-65	0.0	E	1.0	DIPPER	0.0
41-65	0.0	E	1.0	DIPPER	0.0
41-65	0.0	E	1.0	DIPPER	0.0
41-65	0.0	E	1.0	DIPPER	0.0
41-65	0.0	K	0.0	DIPPER	0.0
65+	1.0	K	0.0	NON-DIPPER	1.0
65+	1.0	K	0.0	NON-DIPPER	1.0
65+	1.0	K	0.0	NON-DIPPER	1.0
65+	1.0	K	0.0	NON-DIPPER	1.0
65+	1.0	K	0.0	DIPPER	0.0
0-40	2.0	E	1.0	NON-DIPPER	1.0
41-65	0.0	K	0.0	NON-DIPPER	1.0
41-65	0.0	E	1.0	DIPPER	0.0

Şekil 2.5. İndeksleme Sonucu

Bu tez çalışmasında 2 farklı veriseti kullanılmıştır. Bunlardan birincisi yukarıda tüm işlem adımları anlatılan ve temizleme aşamasında bahsi geçen verisetidir (veriseti-1). Diğeri ise Altıkardeş Z.A'nın çalışmasında kullanılan anomize edilmiş haldeki 80 kişilik verisetidir (veriseti-2). Çalışmaya başlarken her iki verisetinin de, %70'lik kısımları dipper/non-dipper tahminleme modeli oluşturulması amacıyla ayrılmıştır. Kalan %30'luk kısım ise modelin sağlamlık (robustness) kontrolünün yapılması amacıyla kullanılmıştır.

Sınıflandırma - tahminleme yöntemi sırsında sistem uzmanın belirlemiş olduğu etiketlere göre eğitilecek ve oluşturulan model test edilerek doğruluk oranı ortaya konmuş olacaktır. Bu bağlamda, gözetimli öğrenme metodolojisi uygulanacağı için eğitim ve test olarak verisetinin bölünmesi gerekmiştir. Ancak verisetlerindeki veri miktarının az olmasından dolayı, dipper/non-dipper tahminleme modeli oluşturulurken k-katlamalı çapraz doğrulama (k-fold cross validation) yöntemi seçilerek uygulanmıştır. Bu yöntemde veriseti belirlenen k değeri kadar eş parçaya bölünür. Bu parçalardan sadece bir tanesi test, diğerleri ise sistemin eğitilmesi amaçlı kullanılır. Parçalanan verilerden test verisi için kullanılacak küme her bir iterasyonda değiştirilerek farklı kümeler için sistem yeniden eğitilmiş ve test edilmiş olur. Bu sayede az miktardaki veri setlerinde farklı durumlar da ele alınarak elde edilen sonuçların gerçeğe daha yakın olması sağlanmıştır. Bu çalışmada k=10 olarak literatürde en çok kabul görülen şekilde uygulanmıştır. 10-katlamalı çapraz doğrulamaya ait anlatım Şekil 2.6'te gösterilmiştir.



Şekil 2.6. k-Katlamalı Çapraz Doğrulama

Tıp uzmanlarının öneri ve istekleri doğrultusunda üzerinde tahminleme modeli oluşturmak için oluşturulan veri setleri ile ilgili özellikler aşağıda başlıklar halinde anlatılmıştır.

2.4.1.1. Veriseti-1

Yukarıdaki başlıklarda anlatılan tüm işlemler neticesinde elde edilen verisetidir. Bu verisetine ait detaylar aşağıda listelenmiştir.

- Veriseti-1'de kullanılmak üzere seçilen 33 adet öz nitelik Tablo 2.4 ve Tablo 2.5'te görülmektedir.

Tablo 2.4. Veriseti-1'e Ait Öznitelik Tablosu

Demografik Veriler	Laboratuvar Verileri		Holter Verileri	
Metrik	Metrik	Açıklama	Metrik	Açıklama
Yaş	MT_WBC	lökosit	awakeSysSD	Uyanıklık Yüksek Kan Basıncı
Cinsiyet	MT_RBC	Alyuvar sayısı	awakeDiaSD	Uyanıklık Düşük Kan Basıncı
	MT_HGB	Hemoglobin	asleepSysSD	Uyku Yüksek Kan Basıncı
	MT_HCT	Hematokrit	asleepDiaSD	Uyku Düşük Kan Basıncı
	MT_MCV	Kan Hücresi Hacmi		
	MT_MCH	Ortalama Korpusküler Hemoglobin		
	MT_MCHC	Ortalama Korpusküler Hemoglobin Konsantrasyonu		
	MT_PLT	Plates, Trombosit		
	MT_LYM	Lenfosit		
	MT_NEUT	Nötrofil granülasyonu		
	MT_LYM2	Lenfosit		
	MT_MXD2	Karışık Hücre Sayısı		
	MT_NEUT2	Nötrofil granülasyonu		
	MT_RDW_SD	Kırmızı Kan Hücreleri Dağılım Genişliği		
	MT_RDW_CV	Kırmızı Kan Hücreleri Dağılım Genişliği		
	MT_MPV	ortalama trombosit hacmi		
	MT_PDW	Trombosit Dağılım Genişliği		
	MT_P_LCR	Büyük hücreli Trombosit Oranı		

Tablo 2.5. Veriseti-1'e Ait Öznitelik Tablosu Devam

Demografik Veriler	Laboratuvar Verileri		Holter Verileri	
Metrik	**Metrik**	**Açıklama**	**Metrik**	**Açıklama**
	MT_KAN_URE_AZOTU	Kan Üre Azoto		
	MT_URIK_ASIT	Ürik Asit		
	MT_KREATININ	Kreatinin		
	MT_TOTAL_LIPID	Toplam Lipid		
	MT_TOTAL_KOLESTROL	Toplam Kolestrol		
	MT_HDL_KOLESTROL	yüksek yoğunluklu lipoproteinleri		
	MT_LDL_KOLESTROL	düşük yoğunluklu lipoproteinler		
	MT_TRIGILISERID	Trigiliserid		

- 27.000 hastadan etik kurul kararında belirlenen özellikleri taşıyan hasta kayıt sayısı ise 114'dir.
- Her bireyin metebolizması kendine ait özellik taşıdığı için herhangi bir şekilde null verilerin yerine ortalama, min ya da max değer atanması doğru olmayacağından, null datası olan bireylere ait veriler verisetinden çıkartılmıştır. Yukarıda belirtilen öz niteliklerin hepsinin tam olarak ölçümlendiği hasta kayıt sayısı ise böylece 83'e düşmüştür.

2.4.1.2. Veriseti-2

- Veriseti-2'de kullanılmak üzere seçilen 35 adet öz nitelikler Tablo 2.6 ve Tablo 2.7'de görülmektedir.

Tablo 2.6. Veriseti-2'ye Ait Öznitelik Tablosu

Demografik Veriler	Laboratuvar Verileri		Holter Verileri	
Metrik	Metrik	Açıklama	Metrik	Açıklama
Yaş	bsa	Vücut Yüzeyi	dtsis	Uyanıklık Yüksek Kan Basıncı
Cinsiyet	bmi	Vücut Kitle Endeksi	dtdias	Uyanıklık Düşük Kan Basıncı
Kilo	aks	Açlık kan şekeri	ntsis	Uyku Yüksek Kan Basıncı
Belc(bel çevresi)	kreatinin	Kreatin	ntdias	Uyku Düşük Kan Basıncı

Tablo 2.7. Veriseti-2'ye Ait Öznitelik Tablosu Devam

Demografik Veriler	Laboratuvar Verileri		Holter Verileri	
Metrik	Metrik	Açıklama	Metrik	Açıklama
	v2	Valsalva 2. ölçüm	%diasD	Düşük Kan Basıncı Oranı
	vm	Valsalva manevrasına kalp hızı yanıtı		
	dsd	Derin solunumda kalp hızı değişkenliği		
	akh	Ayağa kalkınca kalp hızı değişkenliği		
	orh	Ortostotik hipotansiyon		
	hdkbf	Handgript testi		
	ph	Parasempatik hasar		
	ewing			
	LA	Sol atrium		
	Ao	Aort kökü		
	SWth	Sol ventrikül septal duvar kalınlığı		
	PWth	Sol ventrikül arka duvar kalınlığı		
	LVEDD	Diyastol sonu çapı		
	LVESD	Sistol sonu çapı		
	ME	Mitral erken doluş		
	MA	Mitral geç doluş		
	MDT	Mitral deselerasyon zamanı		
	Ao IVRT	Aortik izovolumik relaksasyon zamanı		
	LVM	Sol ventrikül kütlesi		
	LVMI	Sol ventrikül kütle indeksi		

- Toplamda bireylere ait demografik, laboratuar, otonomik test ve cihaz ölçüm verilerini içeren 80 kayıt bulunmaktadır.

2.4.2. Model oluşturma

Model oluşturmanın ilk aşamasında algoritmalara girdi olarak verilecek öznitelikler seçilirken, WEKA platformunda Selection Attributes altında yer alan InfoGainAttributeEval gibi algoritmalar ile öznitelik sıralaması yapılmıştır. Tıp uzmanlarının görüşleri de alınarak algoritmanın doğruluğuna olumlu etki eden öznitelikler

100’den fazla deney sonucu seçilmiştir. Bu deneylerden en yüksek doğruluk değerlerinin elde edildiği modeller, Naive Bayes ve Random Forest algoritmalarının çalışma prensiplerinin anlatıldığı başlıklar altında örnek olarak sunulmuştur.

2.4.2.1. Naive Bayes

Bayes Teoremine dayanan bir sınıflandırma tekniğidir. Bir Naive Bayes sınıflandırıcısı, bir sınıftaki belirli bir özelliğin varlığının diğer herhangi bir özelliğin varlığı ile ilgisiz olduğunu varsayar.

Örneğin, bir meyve turuncu, yuvarlak ve çapı yaklaşık 8 cm ise bir portakal olarak kabul edilebilir. Bu özellikler birbirine veya diğer özelliklerin varlığına bağlı olsa bile, bu özelliklerin tümü bağımsız olarak bu meyvenin bir portakal olma ihtimaline katkıda bulunur ve bu nedenle 'Naive' olarak bilinir. Naive Bayes modelinin oluşturulması kolaydır ve özellikle çok büyük veri kümeleri için kullanışlıdır ve oldukça karmaşık sınıflandırma yöntemlerinden bile daha iyi performans gösterdiği bilinmektedir.

$$P(A|B) = \frac{\mathrm{P(B|A)} * \mathrm{P(A)}}{\mathrm{P(B)}} \quad (1)$$

P (A | B) = B olayı gerçekleştiğinde A olayının gerçekleşme olasılığı

P (A) = A olayının gerçekleşme olasılığı

P (B | A) = A olayı gerçekleştiğinde B olayının gerçekleşme olasılığı

P (B) = B olayının gerçekleşme olasılığı

Algoritmanın çalıştırılmasında kullanılan Scala kodları aşağıdaki gibidir.

```
import org.apache.spark.ml.classification.Naive Bayes
import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator
import org.apache.spark.SparkContext
import org.apache.spark.sql.SQLContext
import org.apache.spark.sql.SparkSession
import org.apache.spark.ml.linalg.Vectors
import org.apache.spark.ml.feature.VectorAssembler
import org.apache.spark.ml.feature.StringIndexer
import java.util.Properties
import org.apache.spark.sql.SaveMode
```

```
val url = "jdbc:mysql://bigdata.marmara:3306"
val table = "marmara.VP_TR_TRAINING_DATA"
val properties = new Properties()
properties.put("user", "xxx")
properties.put("password", "xxx")
Class.forName("com.mysql.jdbc.Driver")

val data = spark.read.jdbc(url, table, properties)

val nFolds: Int = 10

val yasIndexer = new
StringIndexer().setInputCol("YAS_ARALIGI").setOutputCol("MT_YAS_ARALIGI_INDEX
")
val cinsiyetIndexer = new
StringIndexer().setInputCol("DS_CINSIYET").setOutputCol("MT_DS_CINSIYET_INDEX")
val sonucIndexer = new
StringIndexer().setInputCol("SONUC").setOutputCol("SONUC_INDEX")


val data2 = yasIndexer.fit(data).transform(data)
val data3 = cinsiyetIndexer.fit(data2).transform(data2)
val data4 = sonucIndexer.fit(data3).transform(data3)

val featuresMT = Array("MT_NEUT", "MT_LYM2","MT_RDW_CV",
"MT_MPV","MT_HDL_KOLESTROL", "MT_LDL_KOLESTROL","MT_TRIGILISERID")

val assembler = new VectorAssembler().setInputCols(featuresMT).setOutputCol("features")
val assemblerData = assembler.transform(data4)
val Array(trainingData, testData) = assemblerData.randomSplit(Array(0.7, 0.3), seed = 171L)

val mdl = new Naive Bayes().setLabelCol("SONUC_INDEX").setPredictionCol("prediction")

val paramGrid = new ParamGridBuilder().build()
val pipeline = new Pipeline().setStages(Array(mdl))
val evaluator = new
MulticlassClassificationEvaluator().setLabelCol("SONUC_INDEX").setPredictionCol("predic
tion").setMetricName("accuracy")
val cv = new
CrossValidator().setEstimator(pipeline).setEvaluator(evaluator).setEstimatorParamMaps(para
mGrid).setNumFolds(nFolds)

val model = cv.fit(trainingData)
val predictions = model.transform(testData)

predictions.select("SONUC", "SONUC_INDEX", "PREDICTION").show(100)
val accuracy = evaluator.evaluate(predictions)
```

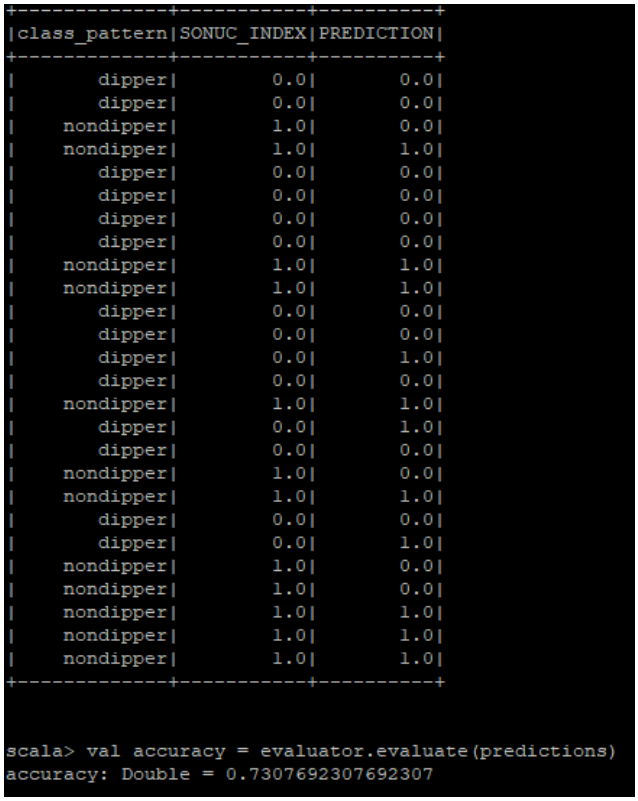
```
val insertColumns = predictions.select("YAS_ARALIGI", "DS_CINSIYET", "MT_WBC",
"MT_RBC", "MT_HGB", "MT_HCT", "MT_MCV", "MT_MCH", "MT_MCHC",
"MT_PLT", "MT_LYM", "MT_NEUT", "MT_LYM2", "MT_MXD2", "MT_NEUT2",
"MT_RDW_SD", "MT_RDW_CV", "MT_MPV", "MT_PDW", "MT_P_LCR",
"MT_GLUKOZ", "MT_KAN_URE_AZOTU", "MT_URIK_ASIT", "MT_KREATININ",
"MT_TOTAL_LIPID", "MT_TOTAL_KOLESTROL", "MT_HDL_KOLESTROL",
"MT_LDL_KOLESTROL", "MT_TRIGILISERID", "SONUC", "SONUC_INDEX",
"prediction")

insertColumns.write.mode(SaveMode.Overwrite).jdbc(url, "marmara.tr_prediction_naive",
properties)
```

Tablo 2.8. Naive Bayes Modeline Ait Öznitelikler

Veriseti	Öznitelikler
Veriseti-1	mt_neut, mt_lym2, mt_rdw_cv, mt_mpv, mt_hdl_kolestrol, mt_ldl_kolestrol, mt_trigiliserid
Veriseti-2	yas, kilo, belc, aks, dsd, orh, hdkbf, ewing

```
+----------+-----------+----------+
|     SONUC|SONUC_INDEX|PREDICTION|
+----------+-----------+----------+
|    DIPPER|        0.0|       1.0|
|    DIPPER|        0.0|       0.0|
|    DIPPER|        0.0|       1.0|
|    DIPPER|        0.0|       0.0|
|    DIPPER|        0.0|       0.0|
|    DIPPER|        0.0|       0.0|
|    DIPPER|        0.0|       1.0|
|    DIPPER|        0.0|       1.0|
|    DIPPER|        0.0|       0.0|
|    DIPPER|        0.0|       0.0|
|    DIPPER|        0.0|       0.0|
|NON-DIPPER|        1.0|       1.0|
|    DIPPER|        0.0|       0.0|
|NON-DIPPER|        1.0|       1.0|
|    DIPPER|        0.0|       0.0|
|NON-DIPPER|        1.0|       0.0|
|NON-DIPPER|        1.0|       1.0|
|    DIPPER|        0.0|       0.0|
|NON-DIPPER|        1.0|       1.0|
|    DIPPER|        0.0|       0.0|
|    DIPPER|        0.0|       1.0|
|NON-DIPPER|        1.0|       1.0|
|    DIPPER|        0.0|       0.0|
|    DIPPER|        0.0|       0.0|
|    DIPPER|        0.0|       0.0|
|NON-DIPPER|        1.0|       0.0|
|    DIPPER|        0.0|       1.0|
+----------+-----------+----------+


scala> val accuracy = evaluator.evaluate(predictions)
accuracy: Double = 0.7037037037037037
```

Şekil 2.7. Veriseti-1 Naive Bayes Modeli Sağlamlık Testi Sonucu



Şekil 2.8. Veriseti-2 Naive Bayes Modeli Sağlamlık Testi Sonucu

Kodların çalışmasıyla veriseti-1 için %70,37 doğruluk, veriseti-2 için %73,07 doğruluk sağlanmıştır, üretilen sonuçların ekran görüntüleri aşağıdaki gibidir. Bu sonuçlara göre veriseti-1'de 27 kayıttan 8 tanesi, veriseti-2'de 26 kayıttan 7 tanesi yanlış tahminlenmiştir. Sonuçlara ait doğruluk matrisi Tablo 2.9'da görülmektedir.

Tablo 2.9. Naive Bayes Doğruluk Matrisi

	Veriseti-1	Veriseti-2
TN (True Negatif)	14	11
TP (True Pozitif)	5	8
FN (False Negatif)	2	4
FP (False Pozitif)	6	3

Duyarlılık

Duyarlılık, pozitif durumların ne kadar başarılı tahmin edildiğini göstermektedir ve aşağıdaki formül ile hesaplanmaktadır.

Duyarlılık (%) = (TP / (TP+FN)) (2)

Özgüllük

Negatif durumların ne kadar başarılı tahmin edildiğini göstermektedir ve aşağıdaki formül ile hesaplanmaktadır.

Özgüllük (%) = (TN) / (TN+FP) (3)

F-Measure

Duyarlılık ve özgüllüğün (kesinliğin) harmonik ortalamasına denmektedir ve aşağıdaki formül ile hesaplanmaktadır.

$$F - Measure = 2 * \frac{\text{Duyarlılık} * \text{Özgüllük}}{Duyarlılık + \text{Özg}\ddot{u}llük} \quad (4)$$

Tablo 2.10. Naive Bayes Model Başarısı

Veriseti	Duyarlılık	Özgüllük	F-Measure
Veriseti-1	71,42	70	70,70
Veriseti-2	66,66	78,57	72,12

Tablo 2.11. Veriseti-1 Naive Bayes Hata Matrisi (Confusion Matrix)

a	b	Sınıflandırma
5	6	a = Non-Dipper
2	14	b = Dipper
		Doğruluk % = (5+14)/(5+2+6+14) =70,37

Veriseti-1 Duyarlılık (%) = TP / (TP+FN) = 5 / (5+2) = 71,42

Veriseti-1 Özgüllük (%) = TN / (TN+FP) = 14 / (14+6) = 70

Tablo 2.12. Veriseti-2 Naive Bayes Hata Matrisi (Confusion Matrix)

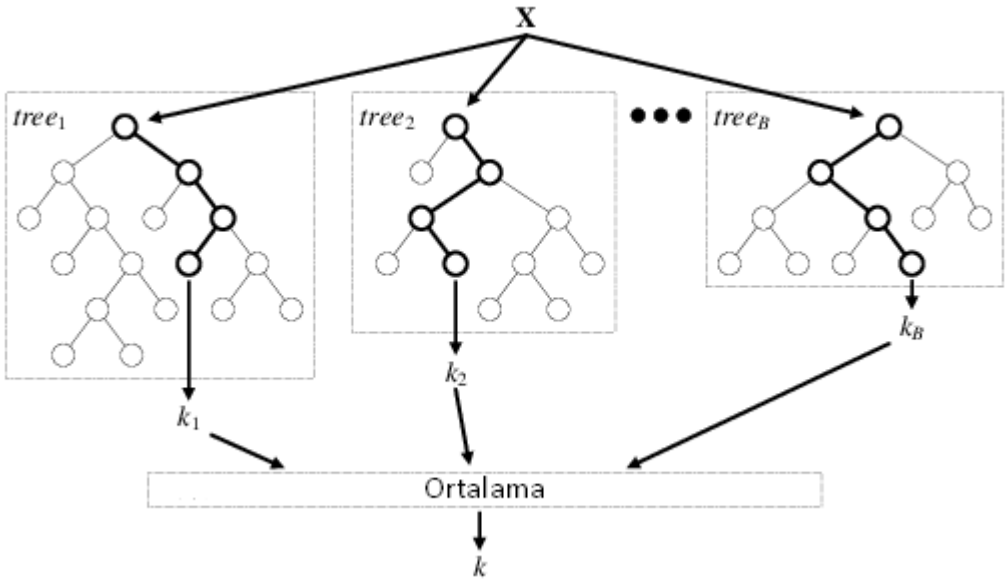
a	b	Sınıflandırma
8	3	a = Non-Dipper
4	11	b = Dipper
		Doğruluk % = (8+11)/(8+4+3+11) =73,07

Veriseti-2 Duyarlılık (%) = TP / (TP+FN) = 8/(8+4) = 66,66

Veriseti-2 Özgüllük (%) = TN / (TN+FP) = 11/(11+3) = 78,57

2.4.2.2. Random Forest(Rassal Orman)

Sınıflandırma yaparken N defa karar ağacı oluşturarak sınıflandırma değerini arttırmaya yarayan bir algoritmadır. N defa çalışması sonucunda üretilen farklı kombinasyonların tahmin ortalamaları alınarak daha doğru tahminler üretmeyi amaçlamaktadır.



Şekil 2.9. Rassal Orman

Algoritma uygulanırken oluşturulacak karar ağaçları sayısı kullanılacak kolon sayısının çok olmamasından dolayı ve deneysel olarak en fazla doğruluk elde edilen 3 değeri verilmiştir.

```
import org.apache.spark.ml.Pipeline
import org.apache.spark.ml.tuning.{ParamGridBuilder, CrossValidator}
import org.apache.spark.ml.classification.RandomForestClassifier
import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator
```

```
import org.apache.spark.ml.Pipeline
import org.apache.spark.ml.evaluation.RegressionEvaluator
import org.apache.spark.ml.feature.VectorIndexer
import org.apache.spark.ml.regression.{RandomForestRegressionModel,
RandomForestRegressor}
import org.apache.spark.ml.classification.Naive Bayes
import org.apache.spark.SparkContext
import org.apache.spark.sql.SQLContext
import org.apache.spark.sql.SparkSession
import java.util.Properties
import org.apache.spark.ml.feature.VectorAssembler
import org.apache.spark.mllib.linalg.Vectors
import org.apache.spark.ml.feature.StringIndexer
import org.apache.spark.sql.SaveMode


val nFolds: Int = 10
val numTrees: Int = 3

val url = "jdbc:mysql://bigdata.marmara:3306"
val table = "marmara.VP_TR_TRAINING_DATA"
val properties = new Properties()
properties.put("user", "xxx")
properties.put("password", "xxx")
Class.forName("com.mysql.jdbc.Driver")


val data = spark.read.jdbc(url, table, properties)

val yasIndexer = new
StringIndexer().setInputCol("YAS_ARALIGI").setOutputCol("MT_YAS_ARALIGI_INDEX
")
val cinsiyetIndexer = new
StringIndexer().setInputCol("DS_CINSIYET").setOutputCol("MT_DS_CINSIYET_INDEX")
val sonucIndexer = new
StringIndexer().setInputCol("SONUC").setOutputCol("SONUC_INDEX")


val data2 = yasIndexer.fit(data).transform(data)
val data3 = cinsiyetIndexer.fit(data2).transform(data2)
val data4 = sonucIndexer.fit(data3).transform(data3)


val featuresMT = Array("MT_YAS_ARALIGI_INDEX", "MT_DS_CINSIYET_INDEX",
"MT_WBC","MT_RBC","MT_HGB","MT_HCT","MT_MCV","MT_MCH","MT_MCHC","
MT_PLT","MT_LYM","MT_NEUT","MT_LYM2","MT_MXD2","MT_NEUT2","MT_RD
W_SD","MT_RDW_CV","MT_MPV","MT_PDW","MT_P_LCR","MT_GLUKOZ","MT_K
AN_URE_AZOTU", "MT_URIK_ASIT","MT_KREATININ",
"MT_TOTAL_LIPID","MT_TOTAL_KOLESTROL","MT_HDL_KOLESTROL",
"MT_LDL_KOLESTROL","MT_TRIGILISERID")
```

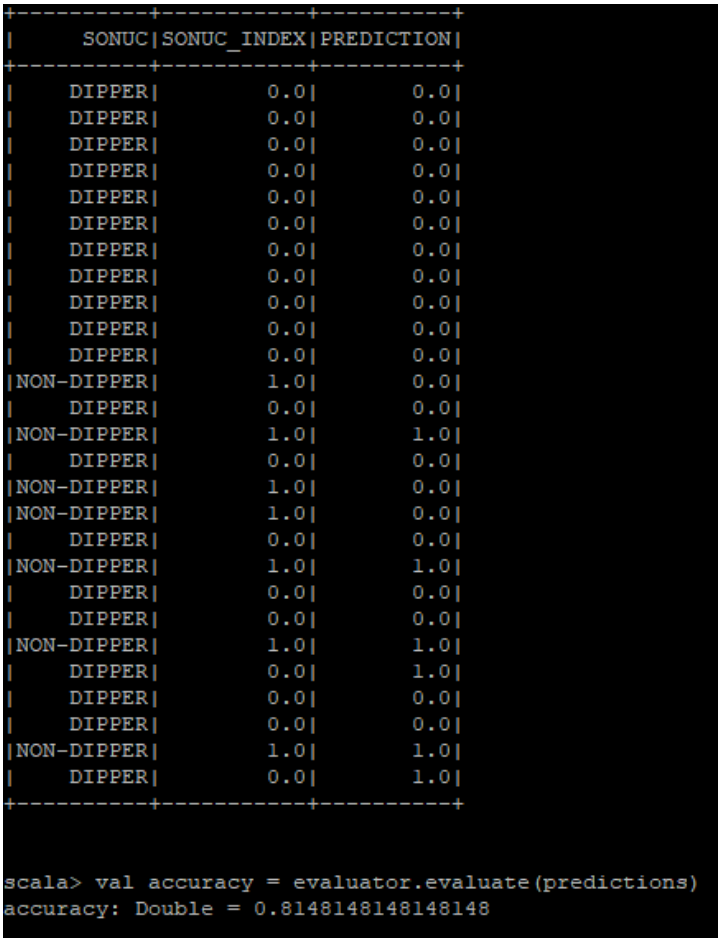
```
val assembler = new VectorAssembler().setInputCols(featuresMT).setOutputCol("features")
val assemblerData = assembler.transform(data4)
val Array(trainingData, testData) = assemblerData.randomSplit(Array(0.7, 0.3), seed = 171L)

val rf = new
RandomForestClassifier().setLabelCol("SONUC_INDEX").setFeaturesCol("features").setNu
mTrees(numTrees)
val paramGrid = new ParamGridBuilder().build()
val pipeline = new Pipeline().setStages(Array(rf))
val evaluator = new
MulticlassClassificationEvaluator().setLabelCol("SONUC_INDEX").setPredictionCol("predic
tion").setMetricName("accuracy")
val cv = new
CrossValidator().setEstimator(pipeline).setEvaluator(evaluator).setEstimatorParamMaps(para
mGrid).setNumFolds(nFolds)

val model = cv.fit(trainingData)
val predictions = model.transform(testData)

predictions.select("SONUC", "SONUC_INDEX", "PREDICTION").show(100)
val accuracy = evaluator.evaluate(predictions)

val insertColumns = predictions.select("YAS_ARALIGI", "DS_CINSIYET", "MT_WBC",
"MT_RBC", "MT_HGB", "MT_HCT", "MT_MCV", "MT_MCH", "MT_MCHC",
"MT_PLT", "MT_LYM", "MT_NEUT", "MT_LYM2", "MT_MXD2", "MT_NEUT2",
"MT_RDW_SD", "MT_RDW_CV", "MT_MPV", "MT_PDW", "MT_P_LCR",
"MT_GLUKOZ", "MT_KAN_URE_AZOTU", "MT_URIK_ASIT", "MT_KREATININ",
"MT_TOTAL_LIPID", "MT_TOTAL_KOLESTROL", "MT_HDL_KOLESTROL",
"MT_LDL_KOLESTROL", "MT_TRIGILISERID", "SONUC", "SONUC_INDEX",
"prediction")

insertColumns.write.mode(SaveMode.Overwrite).jdbc(url, "marmara.tr_prediction_rf",
properties)
```

83 kişilik Veriseti-1 ve 80 kişilik Veriseti-2'de bulunan öznitelikler farklılık göstermektedir. Algoritmalara farklı girdiler sokulmuş ayrı ayrı tahminleme sonuçları elde edilmiştir. Uygulamaya girdi olarak verilen kolonlar aşağıdaki gibidir.

Tablo 2.13. Random Forest Kullanılan Girdiler

Veriseti	Öznitelikler
Veriseti-1	mt_yas_araligi_index, mt_ds_cinsiyet_index, mt_wbc, mt_rbc, mt_hgb, mt_hct, mt_mcv, mt_mch, mt_mchc, mt_plt, mt_lym, mt_neut, mt_lym2, mt_mxd2, mt_neut2, mt_rdw_sd, mt_rdw_cv, mt_mpv, mt_pdw, mt_p_lcr, mt_glukoz, mt_kan_ure_azotu, "mt_urik_asit, mt_kreatinin, mt_total_lipid, mt_total_kolestrol, mt_hdl_kolestrol, mt_ldl_kolestrol, mt_trigiliserid
Veriseti-2	cinsiyet, yas, kilo, belc, bsa, bmi, aks, kreatinin, v1, v2, vm, dsd, akh, orh, hdkbf, ph, ewing

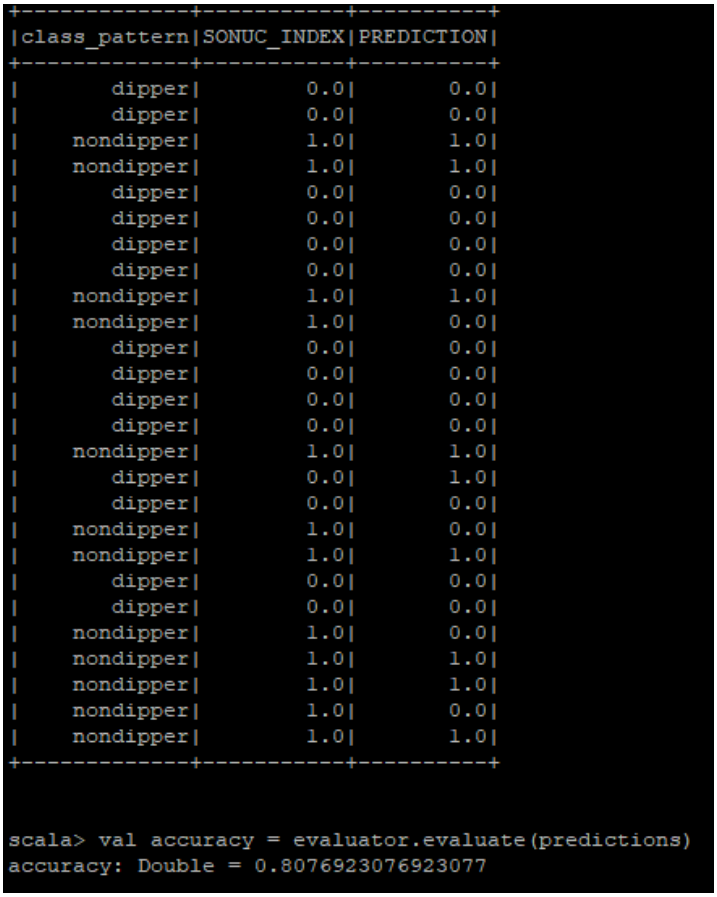
Kodların çalışmasıyla Veriseti-1 için %81,48 doğruluk, Veriseti-2 için %80,76 doğruluk sağlanmıştır, üretilen sonuçların ekran görüntüleri aşağıdaki gibidir.



SONUC	SONUC_INDEX	PREDICTION
DIPPER	0.0	0.0
DIPPER	0.0	0.0
DIPPER	0.0	0.0
DIPPER	0.0	0.0
DIPPER	0.0	0.0
DIPPER	0.0	0.0
DIPPER	0.0	0.0
DIPPER	0.0	0.0
DIPPER	0.0	0.0
DIPPER	0.0	0.0
DIPPER	0.0	0.0
NON-DIPPER	1.0	0.0
DIPPER	0.0	0.0
NON-DIPPER	1.0	1.0
DIPPER	0.0	0.0
NON-DIPPER	1.0	0.0
NON-DIPPER	1.0	0.0
DIPPER	0.0	0.0
NON-DIPPER	1.0	1.0
DIPPER	0.0	0.0
DIPPER	0.0	0.0
NON-DIPPER	1.0	1.0
DIPPER	0.0	1.0
DIPPER	0.0	0.0
DIPPER	0.0	0.0
NON-DIPPER	1.0	1.0
DIPPER	0.0	1.0

```
scala> val accuracy = evaluator.evaluate(predictions)
accuracy: Double = 0.8148148148148148
```

Şekil 2.10. Veriseti-1 ile Random Forest Modeli Sağlamlık Testi Sonucu



```
+-------------+-----------+----------+
|class_pattern|SONUC_INDEX|PREDICTION|
+-------------+-----------+----------+
|       dipper|        0.0|       0.0|
|       dipper|        0.0|       0.0|
|    nondipper|        1.0|       1.0|
|    nondipper|        1.0|       1.0|
|       dipper|        0.0|       0.0|
|       dipper|        0.0|       0.0|
|       dipper|        0.0|       0.0|
|       dipper|        0.0|       0.0|
|    nondipper|        1.0|       1.0|
|    nondipper|        1.0|       0.0|
|       dipper|        0.0|       0.0|
|       dipper|        0.0|       0.0|
|       dipper|        0.0|       0.0|
|       dipper|        0.0|       0.0|
|    nondipper|        1.0|       1.0|
|       dipper|        0.0|       1.0|
|       dipper|        0.0|       0.0|
|    nondipper|        1.0|       0.0|
|    nondipper|        1.0|       1.0|
|       dipper|        0.0|       0.0|
|       dipper|        0.0|       0.0|
|    nondipper|        1.0|       0.0|
|    nondipper|        1.0|       1.0|
|    nondipper|        1.0|       1.0|
|    nondipper|        1.0|       0.0|
|    nondipper|        1.0|       1.0|
+-------------+-----------+----------+


scala> val accuracy = evaluator.evaluate(predictions)
accuracy: Double = 0.8076923076923077
```

Şekil 2.11. Veriseti-2 ile Random Forest Modeli Sağlamlık Testi Sonucu

Tablo 2.14. Random Forest Doğruluk Matrisi

	Veriseti-1	Veriseti-2
TN (True Negatif)	18	13
TP (True Pozitif)	4	8
FN (False Negatif)	3	4
FP (False Pozitif)	2	1

Tablo 2.15. Random Forest Model Başarısı

	Duyarlılık	Özgüllük	F-Measure
Veriseti-1	57,14	90	69,90
Veriseti-2	66,66	92,85	77,60

Tablo 2.16. Veriseti-1 Random Forest Hata Matrisi (Confusion Matrix)

a	b	Sınıflandırma
4	2	a = Non-Dipper
3	18	b = Dipper
		Doğruluk % = (4+18)/(4+3+2+18) =81,48

Veriseti-1 Duyarlılık (%) = TP / (TP+FN) = 4 / (4+3) = 57,14

Veriseti-1 Özgüllük (%) = TN / (TN+FP) = 18 / (18+2) = 90

Tablo 2.17. Veriseti-2 Random Forest Hata Matrisi (Confusion Matrix)

a	b	Sınıflandırma
8	1	a = Non-Dipper
4	13	b = Dipper
		Doğruluk % = (8+13)/(8+4+1+13) =80,76

Veriseti-2 Duyarlılık (%) = TP / (TP+FN) = 8/(8+4) = 66,66

Veriseti-2 Özgüllük (%) = TN / (TN+FP) = 13/(13+1) = 92,85

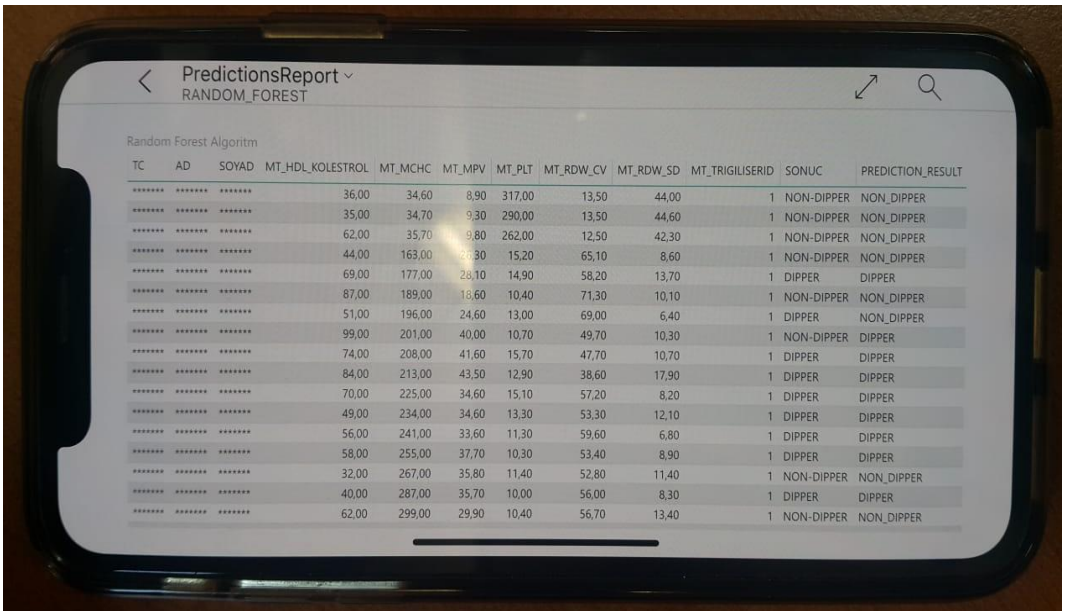
2.5. Web ve Mobil Raporlama

Yapılan çalışmada üretilen tahminleme sonuçlarının kullanıcılara daha düzenli ve anlaşılır bir şekilde gösterilebilmesi için raporların kullanılmasına karar verilmiştir. Rapor üretmek için Microsoft PowerBI ürünü kullanılmıştır. Bu ürün sayesinde MariaDB'ye bağlanılmış. İlgili view'lerden veriler çekilmiş ve çekilen veriler hem web'de hem de mobil uygulamalarda gösterilmiştir. Raporlarda gösterilen verilerde bilgilerin korunabilmesi adına bireylerin TC, Ad, Soyad bilgileri şifrelenerek gösterilmiştir ayrıca tahmin verilerinin

gerçeğe uygunluğunu karşılaştırabilmek adına konulan teşhis sonucu ile tahminleme verisi yan yana koyulmuştur. Raporlara ait görseller aşağıdaki gibidir.

TC	AD	SOYAD	DS_CINSIYET	YAS_ARALIGI	MT_NEUT	MT_LYM2	MT_RDW_CV	MT_MPV	MT_HDL_KOLESTROL	MT_LDL_KOLESTROL	MT_TRIGILISERID	SONUC	TAHMIN
*****	*****	*****	E	65+	10,50	32,70	49,50	32,90	52,00	63,00	79,00	NON-DIPPER	NON_DIPPER
*****	*****	*****	E	41-65	13,90	43,00	57,20	34,60	70,00	129,00	63,00	DIPPER	DIPPER
*****	*****	*****	E	41-65	15,10	46,40	56,00	35,60	48,00	222,00	177,00	DIPPER	NON_DIPPER
*****	*****	*****	E	41-65	15,70	47,00	59,60	29,60	45,00	188,00	140,00	DIPPER	NON_DIPPER
*****	*****	*****	E	41-65	16,30	47,90	54,90	38,70	50,00	187,00	175,00	DIPPER	NON_DIPPER
*****	*****	*****	E	41-65	50,20	2,80	13,40	9,40	50,00	202,00	187,00	DIPPER	DIPPER
*****	*****	*****	E	41-65	51,60	3,60	13,40	9,50	38,00	147,00	145,00	DIPPER	DIPPER
*****	*****	*****	E	65+	61,10	1,80	12,80	9,70	60,00	243,00	94,00	DIPPER	DIPPER
*****	*****	*****	E	41-65	61,30	2,20	12,70	9,60	69,00	153,00	97,00	DIPPER	DIPPER
*****	*****	*****	E	41-65	61,30	2,20	12,70	9,60	70,00	129,00	63,00	DIPPER	DIPPER
*****	*****	*****	E	65+	63,80	2,40	13,30	9,60	60,00	243,00	94,00	DIPPER	DIPPER
*****	*****	*****	K	65+	11,20	36,50	56,70	29,90	62,00	138,00	208,00	DIPPER	NON_DIPPER
*****	*****	*****	K	65+	12,20	40,00	61,70	28,30	74,00	166,00	110,00	NON-DIPPER	DIPPER
*****	*****	*****	K	65+	12,50	37,30	58,50	34,00	72,00	217,00	109,00	DIPPER	DIPPER
*****	*****	*****	K	41-65	12,80	37,60	56,70	30,80	77,00	206,00	102,00	DIPPER	DIPPER
*****	*****	*****	K	41-65	13,60	39,80	49,70	40,00	99,00	165,00	125,00	DIPPER	DIPPER
*****	*****	*****	K	65+	13,90	39,20	47,30	42,10	70,00	166,00	158,00	DIPPER	NON_DIPPER
*****	*****	*****	K	41-65	13,90	42,70	61,70	27,60	37,00	173,00	151,00	NON-DIPPER	NON_DIPPER
*****	*****	*****	K	65+	14,10	42,90	69,00	24,60	51,00	225,00	117,00	DIPPER	DIPPER
*****	*****	*****	K	41-65	14,10	42,40	56,20	36,40	40,00	153,00	292,00	NON-DIPPER	NON_DIPPER
*****	*****	*****	K	65+	14,10	42,90	69,00	24,60	51,00	117,00	82,00	NON-DIPPER	NON_DIPPER
*****	*****	*****	K	41-65	15,30	44,10	56,00	34,20	40,00	153,00	292,00	NON-DIPPER	NON_DIPPER
*****	*****	*****	K	0-40	15,40	45,30	77,50	16,70	34,00	143,00	161,00	DIPPER	NON_DIPPER
*****	*****	*****	K	41-65	52,50	2,90	13,10	9,30	65,00	132,00	96,00	DIPPER	DIPPER
*****	*****	*****	K	41-65	52,50	2,90	13,10	9,30	65,00	132,00	96,00	NON-DIPPER	DIPPER
*****	*****	*****	K	65+	59,00	2,00	12,40	9,00	72,00	217,00	109,00	DIPPER	DIPPER
*****	*****	*****	K	41-65	62,70	2,00	12,50	9,80	62,00	161,00	89,00	DIPPER	DIPPER

Şekil 2.12. Web Platformu Rapor Görüntüsü



Şekil 2.13. Mobil Platformu Rapor Görüntüsü

3. BULGULAR VE TARTIŞMA

Tez içerisinde proje detaylarının verildiği materyal ve yöntem başlığı altında anlatılan çalışmaların bulguları aşağıdaki gibidir;

- Sunucuya uzaktan erişim için SSH kurulmalıdır. Kurulum tamamlandıktan sonra Putty gibi yardımcı bir program ile güvenli bağlantı kurularak uzaktan terminal yardımıyla sunucu kontrol edilebilmektedir.
- Yapısal olmayan bir DB'den (Excel, CSV, JSON vb.) veri alındığında yapısal bir DB'ye aktarımlarda fazlaca veri düzenlenmesi gerekebilmektedir. Veriyi çıkan kaynak birimlerden verilerin en temiz şekilde çıkılması proje maliyetini azaltacaktır.
- Kullanılacak DB seçimi yapılırken IDE kullanılması verinin daha düzenli görünmesini sağlamakta anlaşılabilirliğini arttırmaktadır. Linux tabanlı terminal üzerinden DB'ye bağlanıp verilerin çekilebilmesine karşın görsel bütünlük sağlanmadığından IDE kullanımı tercih edilmiştir.
- Tahminleme yaparken kullanılması muhtemel Scala veya Python programlama dillerinin sağlamış olduğu hazır kütüphaneler Java diline göre daha fazla olmaktadır. Bu sayede yazılacak kod satır sayısı daha sade ve az olacaktır. Bu da uygulama geliştirme maliyetini azaltmakta ve süreyi kısaltmaktadır.
- Sunucu ve client'lar üzerinde host dosyası düzenlenerek sunucu IP'si yerine takma ad ile gitmek erişimlerde kolaylık sağlamaktadır. IP'lere göre takma adların hatırlanması daha kolay olmaktadır.
- Modem içerisinde sunucuya sabit bir DHCP ataması yapılması erişimi kolaylaştırmaktadır. Aksi taktirde modem her kapanıp açıldığında sunucu için tahsis edilen IP değişebilir bu da erişim sorunlarına yol açabilmektedir.
- Veriler tutulan DB'den Excel dosyasına çıkarılacağında maksimum 1,048,576 satır kayda izin verilmektedir.
- Yapısal DB'de tutulan verilerin kullanımları için viewlerin oluşturulması çalışma süreçlerine katkı sağlamaktadır. Tüm tabloyu çekmektense ilgili kolonların çekilmesine, gerekli düzenlemeler yaparak istenen biçimde verilerin çekilmesine ve kompleks SQL sorguları yerine tek seferde view ismi çağrılarak çekilmesi proje verimliliği ve maliyetlerine katkı sağlamaktadır.

- Tahminleme algoritmalarında işleme alınacak girdilerin doğru seçilmesi gerekmektedir aksi taktirde doğrulukları düşürmektedir.
- Random Forest algoritmasında oluşturulacak olan karar ağaçları sayısının belirlenmesi tahminleme doğruluğunu doğrudan etkilemektedir. Doğruluğu en fazla yapacak sayının girilmesi gerekmektedir.
- Verisetindeki veri sayısı az ise k-Katlamalı Çapraz Doğrulama yöntemi kullanılarak sistemin daha stabil çalışması sağlanabilir.
- Naive Bayes algoritması uygulanırken birbiriyle ilişkili ne kadar az değişken kalırsa o kadar doğru sonuçlar vermekteyken Random Forest algoritmasında fazla değişken kullanılması doğruluğu arttırmaktadır.
- Apache Spark'taki çalışmalar terminal penceresi üzerinden yapılırken GUI ekranı olmadığından dönen sonuçlar iç içe girebilmekte bu da analiz süreçlerini zorlaştırmaktadır.

4. SONUÇLAR

4.1. Kısıtlar

- Bilgisayarın (sunucu) alımı için verilen bütçenin az olmasından ötürü ilerleyen zamanlarda sistem yetersizliği ortaya çıkabilecektir.
- Yapılan projede gelen Excel verilerinin farklı yapıda ve formatta gelmesi mevcut çalışmanın sürdürülebilirliğine kısıt olarak yansımıştır.
- Bilgisayarın (sunucunun) bulunduğu lokasyonda statik IP olmamasından ötürü modemin gücünün kesilip yeniden gelmesi durumunda internet çıkış IP'si değişmektedir. Bu da sunucuya uzaktan erişime yeni IP girilene kadar engel olabilmektedir.
- Microsoft PowerBI lisans süresinin dolması kısıt olarak görülmektedir.

4.2. Sonuçlar

Bu çalışmada;

- Büyük veri ortamı kurulmuş bu sayede büyüyen verilerin saklanabileceği ve işlenebileceği platform sağlanmıştır.
- Gerçek zamanlı veri işlemek üzerine çalışmalar yürütülmek istenirse gerekli ortam ve alt yapı sağlanmıştır.
- Makineye uzaktan erişim ile kolay erişim ve müdahale kolaylığı sağlanmıştır.
- Yapısal olmayan veri tabanlarında saklanan verilerin yapısal veritabanında tutulması sağlanmış, bu sayede istenen verilere düzenli bir formatta ulaşma imkanı sağlanmıştır. Ayrıca SQL sorguları ile verilerin kolaylıkla manipüle edilebilmesinin önü açılmıştır.
- 2 farklı veriseti ile tahminleme algoritmaları kullanılmış, farklı girdiler kullanılmasına rağmen yakın sonuçlar elde edilmiştir. Bu da dipper/non-dipper patternine ulaşmak için farklı girdilerin kullanılabileceğini göstermiştir.

Tablo 4.1. Algoritma Model Sonuçları

Algoritmalar	Veriseti-1				Veriseti-2			
	Doğruluk	Duyarlılık	Özgüllük	F-Measure	Doğruluk	Duyarlılık	Özgüllük	F-Measure
Naive Bayes	%70,37	%71,42	%70	%70,70	%73,07	%66,66	%78,57	%72,12
Random Forest	%81,48	%57,14	%90	%69,90	%80,76	%66,66	%92,85	%77,60

- Naive Bayes algoritması ile tahminleme modeli oluşturulurken Veriseti-1'e ait 33 adet öznitelik 7 adet özniteliğe indirgenmiştir. Bu öznitelikler mt_neut, mt_lym2, mt_rdw_cv, mt_mpv, mt_hdl_kolestrol, mt_ldl_kolestrol, mt_trigiliserid şeklindedir.
- Random Forest algoritmasında tahminleme modeli oluşturulurken Veriseti-1'e ait 33 adet öznitelik 29 adet özniteliğe indirgenmiştir. Bu öznitelikler mt_yas_araligi_index, mt_ds_cinsiyet_index, mt_wbc, mt_rbc, mt_hgb, mt_hct, mt_mcv, mt_mch, mt_mchc, mt_plt, mt_lym, mt_neut, mt_lym2, mt_mxd2, mt_neut2, mt_rdw_sd, mt_rdw_cv, mt_mpv, mt_pdw, mt_p_lcr, mt_glukoz, mt_kan_ure_azotu, mt_urik_asit, mt_kreatinin, mt_total_lipid, mt_total_kolestrol, mt_hdl_kolestrol, mt_ldl_kolestrol, mt_trigiliserid şeklindedir.
- Tablo 4.1'de görüldüğü üzere Veriseti-1'de Naive Bayes ile %70,37 doğruluğa karşın Random Forest algoritması ile %81,48 doğruluk elde edilmiştir.
- Veriseti-1 için duyarlılık ve özgüllük değerleri Tablo 4.1'de incelendiğinde duyarlılığın Random Forest algoritmasında %57,14 olmasına karşın Naive Bayes algoritmasında %71,42 değeri ile daha yüksek çıktığı görülmüştür. Bu da hastaların daha hassas şekilde yakalandığını göstermektedir. Yani hastaların sağlıklı olarak nitelendirilme oranı düşüktür.
- Tablo 4.1'de görüldüğü üzere Veriseti-1'de Naive Bayes algoritması için özgüllük oranı %70 Random Forest algoritması için özgüllük oranı %90 çıkmıştır. Bu da Random Forest algoritmasında sağlıklı insanı hasta olarak nitelendirme oranının düşük çıktığını göstermektedir.
- Tablo 4.1'de görüldüğü üzere Veriseti-1'de F-Measure değeri iki algoritma için birbirine oldukça yakın değerde çıkmıştır.

- Naive Bayes algoritması ile tahminleme modeli oluşturulurken Veriseti-2'e ait 35 adet öznitelik 8 adet özniteliğe indirgenmiştir. Bu öznitelikler yas, kilo, belc, aks, dsd, orh, hdkbf, ewing şeklindedir.
- Random Forest algoritmasında tahminleme modeli oluşturulurken Veriseti-2'e ait 35 adet öznitelik 17 adet özniteliğe indirgenmiştir. Bu öznitelikler cinsiyet, yas, kilo, belc, bsa, bmi, aks, kreatinin, v1, v2, vm, dsd, akh, orh, hdkbf, ph, ewing şeklindedir.
- Tablo 4.1'de Veriseti-2 için Naive Bayes algoritması ile %73,07 doğruluğa karşın Random Forest algoritması ile %80,76 doğruluk elde edilmiştir.
- Tablo 4.1'de Veriseti-2 için duyarlılık değerlerinin iki algoritma için de aynı olup bu değer %66,66 olarak görülmektedir.
- Tablo 4.1'de Veriseti-2 için özgüllük değerleri incelendiğinde ise, Random Forest algoritmasının %92,85 değeriyle Naive Bayes algoritmasının %78,57 olan değerine göre daha yüksek çıktığı görülmektedir. Bu bağlamda bakıldığında sağlam bireylere hasta deme oranı Random Forest algoritmasında oldukça düşüktür.
- Tablo 4.1'de görüldüğü üzere Veriseti-2 için kullanılan iki algoritmadaki F-Measure değerleri Random Forest algoritmasında %77,6 değeriyle Naive Bayes algoritmasındaki %72,12 değerine göre daha yüksek çıktığı görülmüştür.
- Tablo 4.1'de her iki veriseti için doğruluk oranlarının yanı sıra duyarlılık, özgüllük ve F-Measure değerleride incelendiğinde, Veriseti-1 için Random Forest algoritmasının duyarlılığının düşük olduğunu görülmektedir. Veriseti-2 için böyle bir durum oluşmamamıştır. Dolayısıyla Veriseti-1 için tüm bu kriterler göz önüne alındığında Naive Bayes algoritması daha güvenilir sonuç üretmiş görünmektedir. Veriseti-2 için ise duyarlılık değerleri her iki algoritma için aynı olmasına rağmen özgüllük değeri olarak Random Forest algoritması daha yüksek sonuç ortaya koyarak güvenilir model üretmiştir.
- Bu çalışma kapsamında veri madenciliği teknikleri ile farklı verisetleri üzerinde çalışma yapılabilecek şekilde bir altyapı oluşturulduğu ortaya konmuştur. Yapılan pilot çalışmalar ile veriler üzerinde dipper/non-dipper patern tahminlemesi için laboratuvar değerlerinin de, EKO değerlerinin de farklı algoritmalar ve hybrid modeller oluşturmak üzere çalışılabileceği sonucuna varılmıştır.

4.3. Öneriler

- Verilerin daha sağlıklı çekilebilmesi için kaynaktan alınan veriler harici dosyalardan ziyade yapılandırılmış bir DB'den alınması daha uygulanabilir bir Extract-Transform-Load (ETL) sürecinden geçmeye olanak tanıyacaktır.
- Veriler yapılandırılmış bir DB'den alınamıyorsa, gelen harici dosyaların belirli bir standarda oturtularak gönderilmesi daha uygulanabilir bir ETL sürecine imkan tanıyacaktır.
- Projenin real-time sonuçlar üretmesi projeyi daha değerli kılacaktır. Bunun için kaynaktan gelen verilerin yukarıdaki önerilerde belirtildiği şekilde alınması gerekmektedir. Harici dosya olarak gönderimler devam edecekse Apache Kafka'nın kullanımı yararlı olacaktır. Ardından Apache Nifi gibi real-time'a elverişli bir ETL tool'u yardımıyla veriler işlenecek ve istenen sonuçlar üretilebilecektir.
- İleriki çalışmalarda, tıp uzmanlarının istek ve önerileri ile oluşturulan veriseti üzerinde Scala diliyle çeşitli makine öğrenmesi algoritmaları kullanılarak başarılı bir tahmin modeli oluşturulması planlanmaktadır. Örneğin EKO parametreleri eklenerek yapay sinir ağları gibi farklı algoritmalar ile çalışma genişletilebilir.

KAYNAKLAR

[1] Yilmaz, M. B., Yalta, K., Turgut, O. O., Yilmaz, A., Yucel, O., Bektasoglu, G., & Tandogan, I. (2007). Sleep quality among relatively younger patients with initial diagnosis of hypertension: Dippers versus non-dippers. Blood pressure, 16(2), 101-105.

[2] World Health Organization. (2014). Noncommunicable diseases country profiles 2014.

[3] Dirin B.: "Big Data Nedir", https://www.bilginc.com/tr/egitim-haber/buyuk-veri-big-data-nedir/31, (Temmuz 2019).

[4] Altıkardeş, Z. A. (2012). Tip 2 Diabetes Mellituslu Hastalarda

24 Saatlik Kan Basinci Değişiminin Öngörülmesine Yönelik Uzman Sistem Tasarimi (Doktora Tezi, Elektronik-Bilgisayar Eğitimi Anabilim Dali).

[5] Akgün, B. (2016). Apache Spark Tabanlı Destek Vektör Makıneleri İle Akan Büyük Veri Sınıflandırma (Doctoral dissertation, Fen Bilimleri Enstitüsü).

[6] Alayyoub M. (2016). A comparative study of big data stream processing frameworks: spark, storm, and flink (Yüksek Lisans Tezi, The Graduate School of Natural and Applied Science of Atilim University).

[7] Çelikel N. (2018). big data frameworks performance evaluation (yüksek lisans tezi, graduate school of natural and applied sciences).

[8] Awlaqi I. A. A., (2018). Medikal turizm sektöründe büyük veri uygulamalari üzerine nitel bir araştirma (yüksek lisans tezi, sosyal bilimler enstitüsü işletme anabilim dali).

[9] Ed-daoudy, A., & Maalmi, K. (2018, November). Application of machine learning model on streaming health data event in real-time to predict health status using spark. In 2018 International Symposium on Advanced Electrical and Communication Technologies (ISAECT) (pp. 1-4). IEEE.

[10] Keskin, M. V., & Yıldız, D. Büyük veri araçları ve r kullanarak Amerikan havayolu firmalarinin sorunlarinin keşfedilmesi. Uygulamalı Sosyal Bilimler Dergisi, 2(2), 1-19.

[11] Correia, R., Spadon, G., De Andrade Gomes, P., Eler, D., Garcia, R., & Olivete Junior, C. (2018). Hadoop Cluster Deployment: A Methodological Approach. Information, 9(6), 131.

[12] Ed-daoudy, A., & Maalmi, K. (2018, November). Application of machine learning model on streaming health data event in real-time to predict health status using spark. In 2018 International Symposium on Advanced Electrical and Communication Technologies (ISAECT) (pp. 1-4). IEEE.

[13] Attiga, Y., Chen, S. Y., LaGue, J., Ovalle, A., Stott, N., Brander, T., ... & Francis-Lyon, P. (2018, November). Applying Deep Learning to Public Health: Using Unbalanced Demographic Data to Predict Thyroid Disorder. In 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON) (pp. 851-858). IEEE.

[14] Janssoone, T., Bic, C., Kanoun, D., Hornus, P., & Rinder, P. (2018). Machine learning on electronic health records: Models and features usages to predict medication non-adherence. arXiv preprint arXiv:1811.12234.

[15] Yılmaz, A. Büyük Veride Hadoop Mimarisi ile VoIP Güvenliği Önerisi. Avrupa Bilim ve Teknoloji Dergisi, (16), 267-274.

[16] Koçer F. (2019). İlişkisel veri tabanlarinin uygulandiği araç takip sisteminin büyük veriye uyarlanmasi (Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü).

Süleyman ÖZER

Tel 0 (555) 375 63 64
0 (216) 324 93 85

Adres Yenisahra Mahallesi Mimar Sinan Caddesi No:22/2 Ataşehir/İstanbul

E-Posta s.ozer91@gmail.com / s.ozer91@hotmail.com

Doğum Tarihi : 03.11.1991
Doğum Yeri : İstanbul / Üsküdar
Uyruğu : T.C.
Medeni Hali : Bekâr

Eğitim Durumu :

2016 - **Marmara Üniversitesi(İstanbul)**
Yüksek Lisans, Bilgisayar Mühendisliği(Örgün Öğretim)

2011 - 2015 **Süleyman Demirel Üniversitesi(Isparta)** **74.8/100**
Lisans, Bilgisayar Mühendisliği(Örgün Öğretim)

2010 - 2011 **Süleyman Demirel Üniversitesi(Isparta)** **75/100**
Sertifika, İngilizce Hazırlık(Örgün Öğretim)

2006 – 2009 **Habire Yahşi Lisesi (İstanbul)**
Fen

Deneyimler :

2019 - **SunExpress**
İş Analisti

2017 - 2019 **Türk Hava Yolları A.O.**
Büyük Veri ve İş Zekası Uzmanı

2016 - 2017 **Itelligence Analytics**
SAP BO/BI Consultant

2015 - 2015 **Isparta 112 Acil Çağrı Merkezi Müdürlüğü**
Yazılım stajı

2014 - 2014 **Bilgi Birikim Sistemleri San. Tic Ltd Şti.**
Ağ güvenliği stajı

Sertifikalar :

*** Başarı Sertifikası**
SDÜ Yabancı Diller Yüksek Okulu – *14.06.2011*
*** Etkili İletişim Eğitim Sertifikası**
Dünya Akademi – *Mart 2013*
*** Hitabet Eğitim Sertifikası**
Dünya Akademi – *Mart 2013*
*** Diksiyon Eğitim Sertifikası**
Dünya Akademi – *Mart 2013*

EKLER

EK-1

Modem Ayarlarının Yapılması

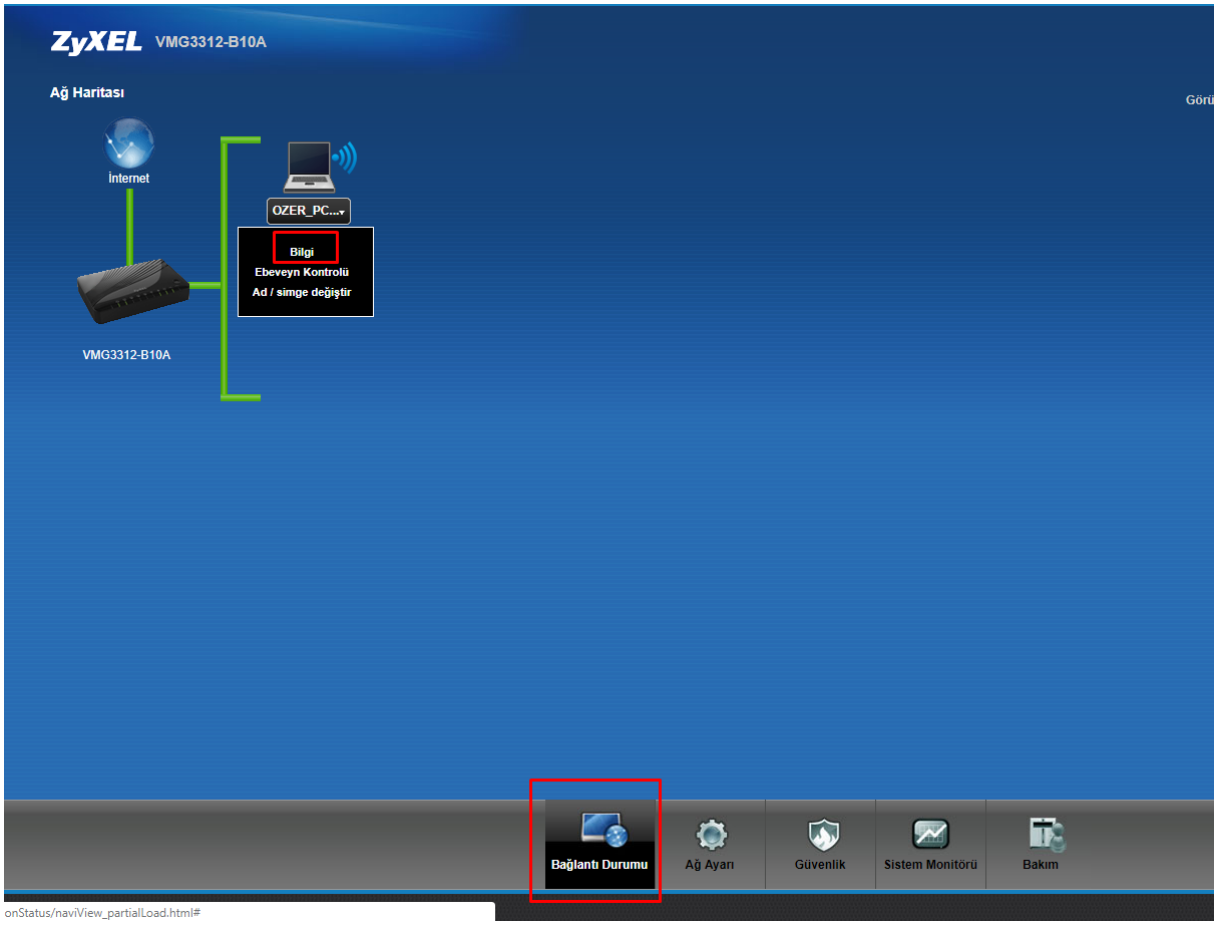
IP sabitleme

IP sabitleme, modeme bağlı cihaz için sabit bir IP tahsis etmeye denilmektedir. Bu işlem neticesinde modem ya da bilgisayar kapansa dahi, tahsis edilen IP Dynamic Host Configuration Protocol (DHCP)'de tutulacak ve başka cihazlara verilmeyecektir.

Büyük veri platformunun kurulu olduğu bilgisayara da local IP'sinin değişmemesi için modem üzerinden sabit bir IP verilmiştir. Bu sayede bilgisayar da modem de yeniden başlatılmış olsa da local IP'si değişmeyecektir. Uzaktan bağlantı için hazır olacaktır. Tahsis edilen sabit IP büyük veri makinesi için 192.168.1.7'dir.

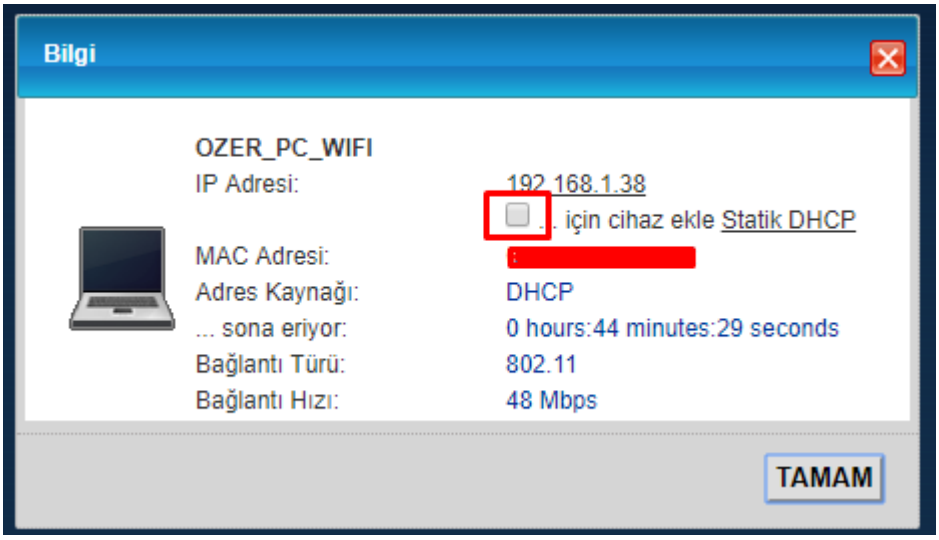
IP sabitleme adımları;

1- Bağlantı durumu seçeneğinde görünen cihazlardan, IP'sini sabitlemek istenilene tıklanır ve açılan pencerede bilgi seçeneği seçilir (Şekil 1).



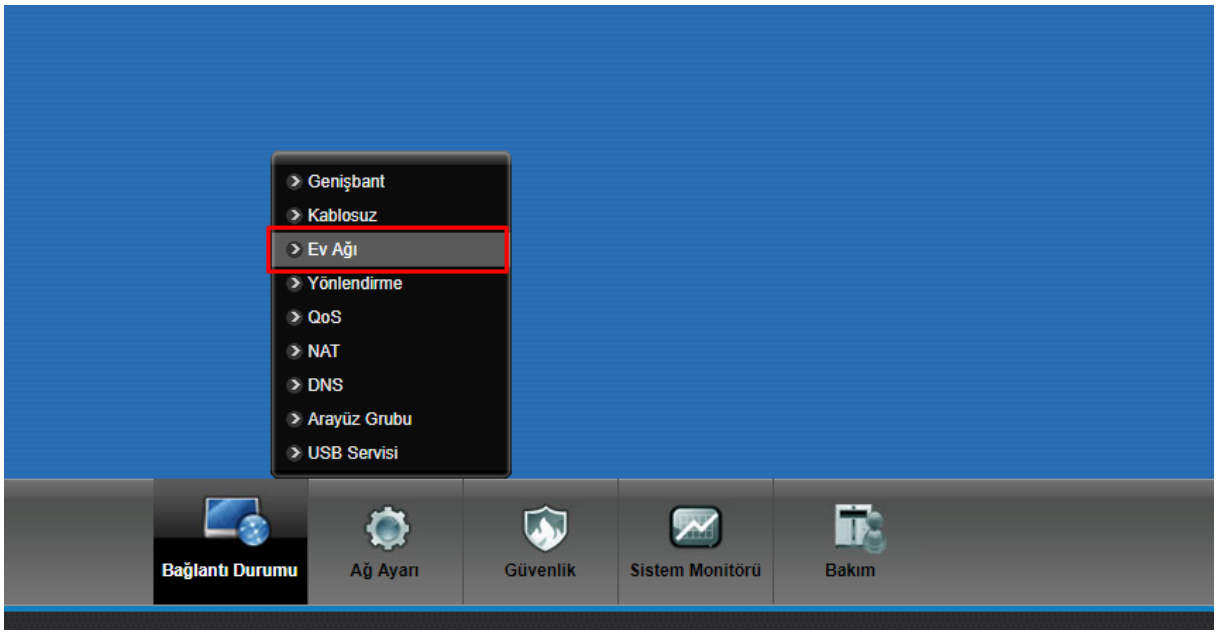
Şekil 1. Modem Arayüzü

2- Açılan pencerede statik DHCP ekle kutucuğu işaretlenir. Bilgi güvenliğinin sağlanması amacıyla bilgisayarın MAC adresi gizlenmiştir (Şekil 2).



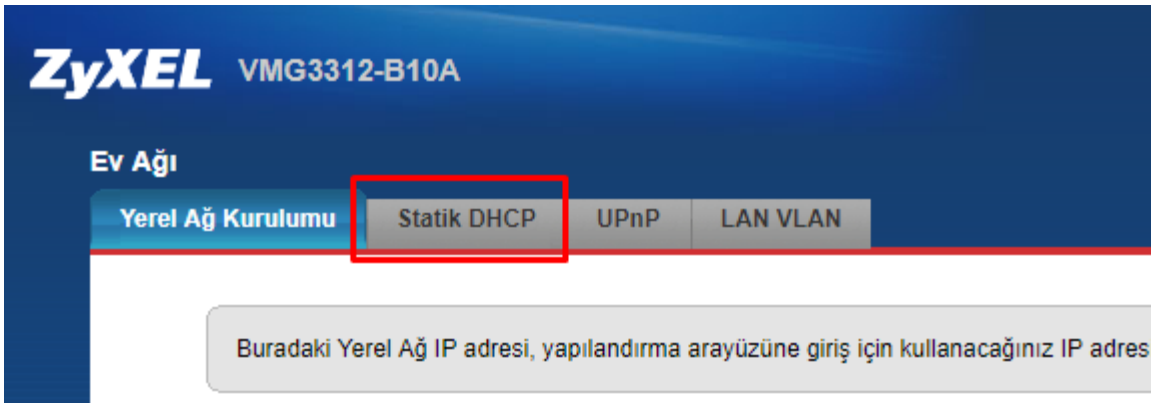
Şekil 2. Statik DHCP

3- Kontrolün sağlanması için Ağ Ayarı – Ev ağı seçeneğine girilir. Burada adı geçen Ev ağı networksel olarak Wide Area Network (WAN)'a tekabül etmektedir. Yani modemin sağladığı network ortamıdır (Şekil 3).



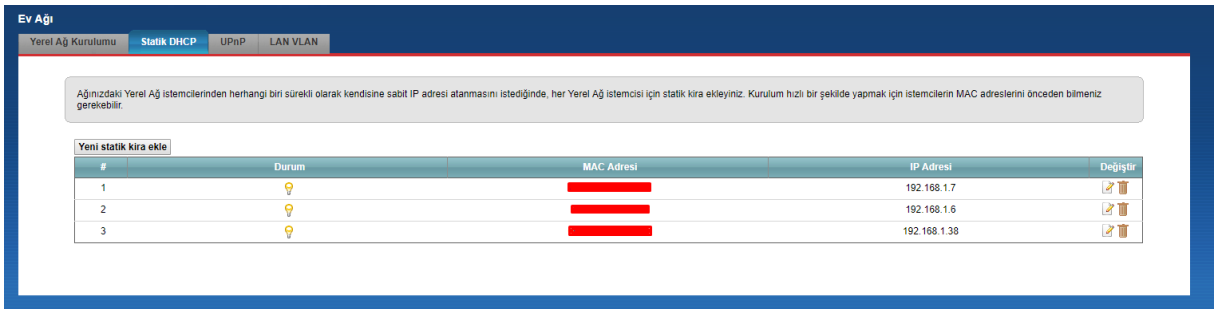
Şekil 3. Ev Ağı

4- Üst sekmelerden Statik DHCP sekmesine tıklanır (Şekil 4).



Şekil 4. Statik DHCP Sekmesi

5- Açılan pencerede statik IP verilen cihazlar listesi görünmektedir (Şekil 5).



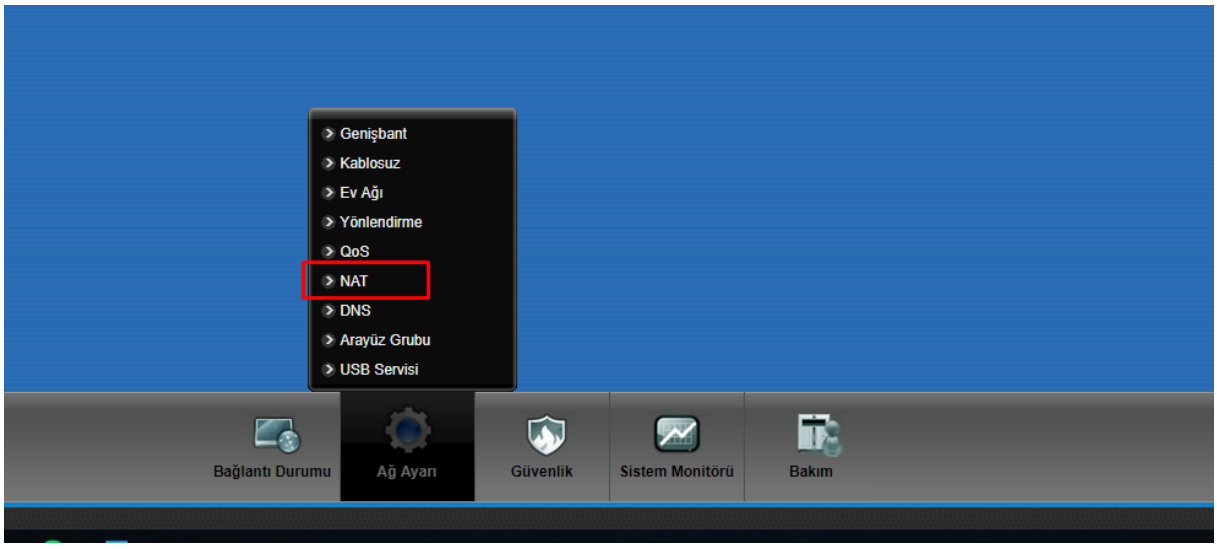
Şekil 5. Statik DHCP Kontrol

Port yönlendirme

Port yönlendirme, modemin internete açılan IP'sine belirli bir port numarası ile gelindiğinde yönlendirilecek IP ve portu belirlemeye yarar. Modemin internet çıkış IP'sine büyük veri platformunda kullanılacak portlar ile erişim için gelindiğinde modem gelen isteği otomatik olarak büyük veri makinesine yönlendirme yapacaktır. Bu sayede büyük veri makinesindeki servislere modeme bağlı olmadan uzaktan erişim açılmış olacaktır.

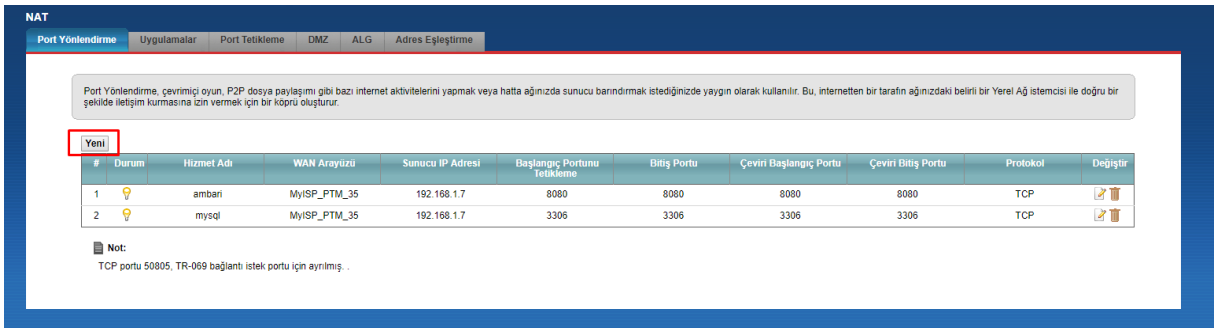
Port yönlendirme adımları;

1- ZyXEL marka modemler için port yönlendirme yapmak için ağ ayarları menüsünden NAT seçilir (Şekil 6).



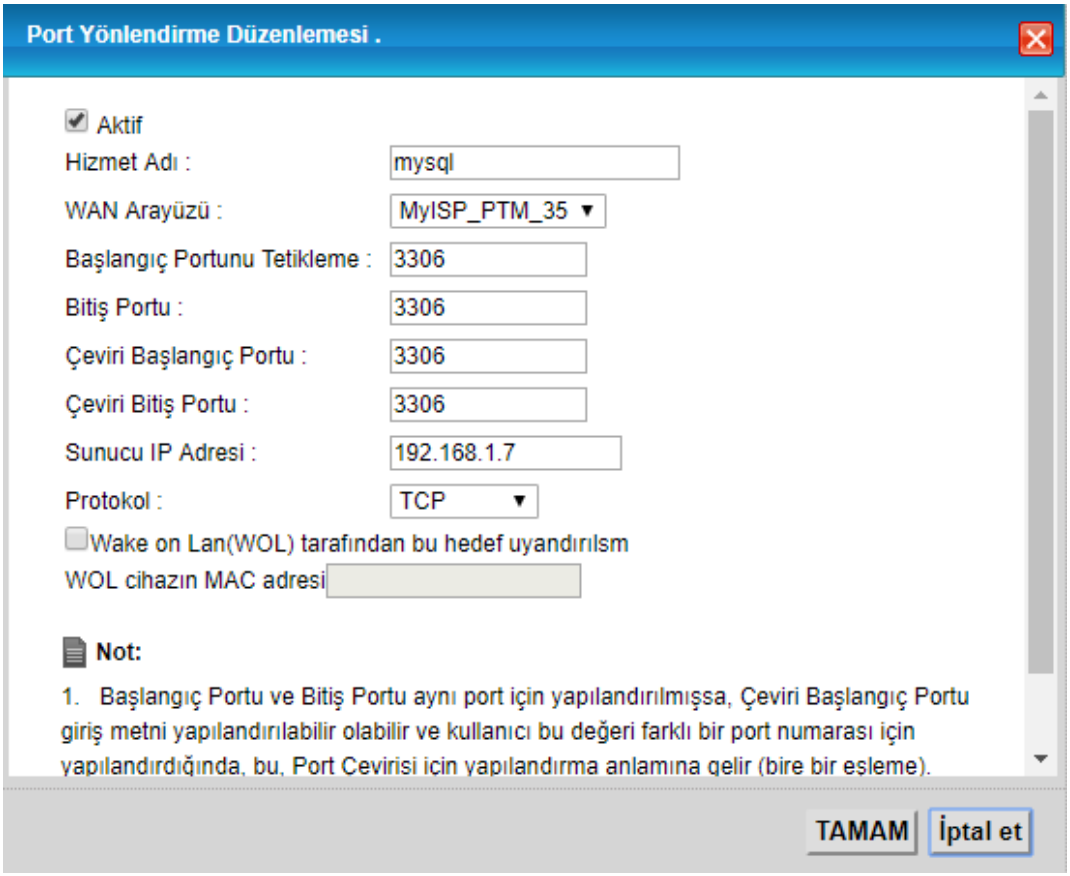
Şekil 6. NAT Seçim

2- Açılan panelden yeni butonuna basılır (Şekil 7).



Şekil 7. NAT Ekranı

3- Açılan pencerede gerekli bilgiler doldurulur ve ardından "tamam" butonuna basılır. Burada seçilen 3306 portu MySQL ve MariaDB'ye default erişim portudur. Standardın bozulmaması adına port yönlendirmede de 3306 kullanılmıştır (Şekil 8).



Şekil 8. Port Yönlendirme