

ızzet

by ızzet 1

Submission date: 06-Jul-2023 03:05PM (UTC+0300)

Submission ID: 2127227285

File name: ızzet_ozdemir_223739370_VA598.docx (2.18M)

Word count: 3377

Character count: 22995



4
T.C.

ALTINBAŞ ÜNİVERSİTESİ

Lisansüstü Eğitim Enstitüsü

Veri Analitiği Anabilim Dalı

Diyabet Hastalığı Tahmini

İzzet Özdemir

Yüksek Lisans PROJESİ

Danışman

Dr. Oğuz KARAN

İstanbul, 2023

Diyabet Hastalığı Tahmini

İzzet ÖZDEMİR

1

Veri Analitiği Anabilim Dalı

Yüksek Lisans Projesi

ALTINBAŞ ÜNİVERSİTESİ

2023

Bu projedeki tüm bilgilerin akademik kurallara ve etik davranışlara uygun olarak edinildiğini ve sunulduğunu beyan ederim. Ayrıca, çalışmanın tamamında bu kuralların ve davranışların gerektirdiği şekilde, hareket ederek orijinal olmayan tüm materyalleri ve sonuçları tamamen alıntı yaptığımı ve referans gösterdiğim beyan ederim.

İzzet ÖZDEMİR

İmzası

ÖZET

DİYABET HASTALIĞI TAHMİNİ

ÖZDEMİR, İzzet

1

Yüksek Lisans Projesi, Veri Analitiği Anabilim Dalı, Altınbaş Üniversitesi

Danışman: Dr. Oğuz KARAN

Tarih: 06 / 2023

Sayfa: 28

Yaşamakta olduğumuz bilgi çağında teknoloji hızla ilerlemektedir. Gelişen teknoloji ile birlikte birçok meslekte gözle görülür gelişmeler yaşanmaktadır. Özellikle son yıllarda yapay zekanın hayatımıza girmesiyle, gelişim gösteren sektörlerden biri olan sağlık sektöründe, makine öğrenimi ve derin öğrenme algoritmaları ile hastalık tahminlemesine yönelik ciddi çalışmalar yürütülmektedir. Bu projedelığımızın en sık görülen hastalıklarından biri olan diyabet yani halk dilindeki adıyla şeker hastalığının makine öğrenimi modelleri ile analizi yapılarak hastalık teşhisini konulmaya çalışılmıştır.

Diyabet hastalığı, vücutta insülin hormonunun yetersiz olduğu ya da hiç olmadığı ve yahut diğer dokuların insüline karşı duyarsız hale gelmesi durumlarında ortaya çıkmaktadır. İnsülin hormonu pankreas tarafından üretilir. Bu hormon kan içerisindeki şekerin dolaşım esnasında hücreler tarafından emilmesini sağlar. Böylece vücut ihtiyacı olan enerjiyi üretecek yakıt bulabilir. Ancak bazen de kandaki şeker olması gerekenden daha fazla olabilir. Kanda bulunan yüksek orandaki şekerin karaciğer, kas gibi çeşitli organlarda depolanması için vücutu uyarır.

⁵
Projede modelleme için “Pima Indians Diabets” veri seti kullanılmıştır. Bu veri seti, ABD’ nin Arizona Eyaletinde Phoenix yakınlarında yaşayan Pima Kızılderili popülasyonunun incelenmesi sonucu elde edilmiştir. ⁶ “Amerikan Ulusal Diyabet ve Sindirim ve Böbrek Hastalıkları Enstitüsü” kaynaklarındanandır. Veri seti, belirli teşhis ölçümlerini kapsamaktadır. Hastanın diyabet olup olmadığıının tahminlenmesi amacına uygun olması amacıyla bu projede kullanılmıştır.

Anahtar Kelimeler: Diyabet, Makine Öğrenimi, Pima Kızılderilileri

ABSTRACT

DIABETES PREDICTION

Ozdemir, Izzet

Master Project, Data Analytics Department, Altınbaş University,

Supervisor: Dr. KARAN, Oguz

Date: 06 / 2023

Pages: 28

In the age of information we live in, technology is advancing rapidly. With the development of technology, many professions are experiencing noticeable advancements. Especially in recent years, with the introduction of artificial intelligence into our lives, significant efforts are being made in the healthcare sector, one of the most evolving sectors, towards disease prediction using machine learning and deep learning algorithms. In this project, an analysis of diabetes, which is one of the most common diseases of our time and also known as sugar disease, has been attempted using machine learning models to make a diagnosis.

Diabetes occurs when there is insufficient insulin hormone in the body or when other tissues become insensitive to insulin. Insulin is produced by the pancreas. This hormone enables the absorption of sugar in the blood by the cells during circulation. Thus, the body can find the necessary fuel to produce the energy it needs. However, sometimes there can be more sugar in the blood than necessary. This prompts the body to store the excess sugar in various organs, such as the liver and muscles.

The "Pima Indians Diabetes" dataset was used for modeling in this project. This dataset was obtained through the examination of the Pima Native American population living near Phoenix, Arizona, in the United States. The data is sourced from the "National Institute of Diabetes and

Digestive and Kidney Diseases" in America. The dataset encompasses specific diagnostic measurements. It has been used in this project for the purpose of predicting whether a patient has diabetes or not.

Keywords: Diabetes, Machine Learning, Pima Indians

İÇİNDEKİLER

	<u>Sayfa</u>
ÖZET	iv
ABSTRACT	vi
⁴ TABLO LİSTESİ.....	ix
ŞEKİL LİSTESİ	x
KISALTMALAR	xii
1. GİRİŞ	1
2. MATERİYAL VE METOD.....	3
2.1 VERİ SETİ ANALİZİ	3
2.2 VERİ SETİ DÜZENLEMESİ	4
2.2.1 Veri Ön İşleme:	4
3. MODELLEME	19
3.1.1 Veri Setini Bölme:	19
3.1.2 Algoritmalar:	19
3.1.3 Değerlendirme:	21
4. SONUÇ	22
REFERANSLAR	23

TABLO LİSTESİ

5
Sayfa

Tablo 2.1: Pima Indians Diabetes veri seti özellikleri.....4

ŞEKİL LİSTESİ

	<u>Sayfa</u>
Şekil 2.1: İlk yüklenen Python kütüphaneleri.	5
Şekil 2.2: CSV formatındaki veri setine erişim.	5
Şekil 2.3: Veri seti nitelik isimlerini Türkçe yapılması.	6
Şekil 2.4: Veri seti satır, sütun bilgisi.	6
Şekil 2.5: Veri seti nitelikleri hakkında bilgiler.	6
Şekil 2.6: Veri tipleri değiştirilmesi.	7
Şekil 2.7: Temel istatistiksel değerlerin gösterilmesi.	7
Şekil 2.8: Veri seti boş değer sorgulaması.	8
Şekil 2.9: Bazı alanların sıfır değerlerinin “NaN” yapılması.	9
Şekil 2.10: Veri seti niteliklerinin boş değer sayıları.	10
Şekil 2.11: Boş değer içeren niteliklerin doldurulması.	10
Şekil 2.12: Veri seti niteliklerinin güncellenmiş boş değer sayıları.	11
Şekil 2.13: Veri seti nitelikleri arasındaki korelasyon değerleri.	11
Şekil 2.14: Korelasyon değerlerinin heatmap ile görselleştirilmesi ve çıktısı.	12
Şekil 2.15: Histogram görselinin oluşturulması ve çıktısı.	13
Şekil 2.16: Boxplot görselinin oluşturulması ve çıktısı.	14
Şekil 2.17: Baskılama metodunun kodlaması ve çıktısı.	15
Şekil 2.18: Baskılama metodunun kontrol edilmesi.	16
Şekil 2.19: Glikoz kategorik özellik eklemesi.	17
Şekil 2.20: İnsülin kategorik özellik eklemesi.	17
Şekil 2.21: Beden Kitle Endeksi kategorik özellik eklemesi.	18
Şekil 2.22: Label Encoder ve Standard Scaler fonksiyonlarının yüklenmesi.	18
Şekil 2.23: Label Encoder işlemi ve çıktısı.	18
Şekil 3.1: Hedef değişkenimizin veri setinden ayrılması.	19
Şekil 3.2: Standard Scaler çalışması ve çıktısı.	19
Şekil 3.3: train_test_split fonksiyonunun yüklenmesi ve veri bölme işlemi.	20
Şekil 3.4: Algoritma kütüphanelerinin yüklemesi.	20

Şekil 3.5: Algoritmaların Model dizinine eklenmesi.....	21
Şekil 3.6: Modelin çalıştırılması.....	21
Şekil 3.7: Model çıktısı.....	22

KISALTMALAR

T2DM	:	Type 2 Diabet Mellius
EDA	:	Exploratory Data Analysis
NaN	:	Not a Number
CSV	:	Comma Separated Values
GBM	:	Gradient Boosting Machine (Ligh GBM)
XGBoost	:	¹³ eXtreme Gradient Boosting
SVM	:	Support Vector Machine
KNN	:	K-Nearest Neighbors
CART	:	Classification and Regression Tree
LR	:	Logistic Regression

1. GİRİŞ

Pima, ABD'nin Arizona eyaletindeki Phoenix yakınlarındaki Gila ve Salt nehirleri kıyılarında yaşan Kuzey Amerika Kızılderilileridir. "Nehir insanları" olarak kendilerini adlandırırlar. Pima'lar Hohokam soyundan gelmektedir. Onlar ataları gibi tek odalı evlerde yaşayan ve nehir suyu ile sulama yaparak tarım yapan çiftçilerdi. Nehir kıyısında yaşamalarına rağmen ortalama beş yılda bir kuraklık ile karşıya kaldılar. Bu ağır koşulların yanı sıra 1900'lü yıllarda itibaren beyaz yerleşimcilerin nüfusu artış gösterdi ve nehir suyunun yönünde değişiklikler oldu. Bu dönemlerde mahsul kıtlığı onları avcılık ve toplayıcılık yapmaya zorladı. Bu da onların geleneksel ekonomilerine derin darbeler vurdu ve çiftçiliğin gerilemesine yol açtı. Çevresel faktörlerin etkili olmasıyla yaşam koşulları değişen Pima yerlileri, yeni hayat tarzına adapte olsalarda ciddi sağlık sorunları ile karşıya kaldılar. Diğer Kızılderili kabilelere nazaran daha homojen bir yapıya sahip olan Pima Kızılderilileri, dünyadaki en yüksek tip2 diyabet prevalansına sahiptirler. Bu durum oldukça dikkat çekicidir. 1984' te antropolog Robert Ferrell tarafından ortaya atılan bir hipoteze göre Amerikan yerlileri arasında diyabet prevalansının artmasının nedeni olarak genetik yatkınlık ile birlikte, bu popülasyonların tarımsal ürünler yerine işlenmiş gıdalar ile beslenmeleri ve hareketsiz yaşam tarzları olduğu ileri sürülmüştür.

Pima Kızılderilileri üzerinde iki yılda bir aralıklarla incelemeler yapıldı. Diyabetin başlangıcı ve gelecekteki komplikasyonları ile ilgili bulgular değerlendirilmeye çalışıldı. Bu projede kullanılan "Pima Indians Diabetes" veri seti, "Amerikan Ulusal Diyabet ve Sindirim ve Böbrek Hastalıkları Enstitüsü" kaynaklarında yer almaktadır. Bu veri seti, Pima Kızılderili ırkından 21 yaş ve üzeri toplam 768 Pima kadının araştırma verilerini içermektedir. Amaç, dünyanın en yüksek tip2 diyabet prevalansına sahip topluluğundan elde edilen araştırma verilerini kullanarak insanların diyabet hastası olup olmadığı sonucunu makine öğrenimi algoritmaları yardımıyla tahminlemektir. Bu projede oluşturulacak model yardımıyla eğer yüksek doğruluk oranında bir tahminleme elde edilebilirse, hastalığın teşhisinde önemli bir adım teknolojik unsurlar ile tavsiye niteliğinde sağlanmış olacak ve insanlığın bu hastalığın pençesinden kurtulması için belki de küçük ama önemli bir adım atılmış olacaktır.

Projenin en önemli handikabı veri sayısının düşüklüğüdür. Veri setinde 768 kayıt mevcuttur. Bu veriler, teknik olarak eğitim ve test verisi olarak bölünecektir. Dolayısıyla makine öğrenimi için yeterli bir eğitim verisi olmayabilir. Ancak veri kaynağının orijinalliği bu veri setindeki çalışmayı önemli kılmaktadır.

Veri seti üzerinde yapılan keşifsel veri analiz (EDA) bulgularında, bazı niteliklerin ölçümle
2
ribben değerlerin sıfır olamayacağı halde sıfır girildiği tespit edilmiştir. Bu alanlar glikoz, kan basıncı, deri kalınlığı, insülin ve beden kitle endeksi nitelikleridir. Bu nitelikler için çalışma yapılmış ve bu değerlerin en anlamlı şekilde doldurulması için özel çaba sarfedilmiştir.

2. MATERİYAL VE METOD

2.1 VERİ SETİ ANALİZİ

Proje çalışmasında kullanılan “Pima Indian Diabetes” veri seti 768 kayıttan ve 9 özellik (nitelik, alan, değişken) den oluşmaktadır. (Tablo 2.1)’ de görüldüğü üzere 8 farklı bağımsız değişken ve 1 bağımlı değişken yer almaktadır. “Outcome” alanı kişilerin diyabet olup olmadığını gösteren bağımlı değişkendir. Aldığı sıfır (0) değeri kişinin diyabet olmadığını, 1 değerler ise diyabet olduğunu ifade etmektedir. Diğer özellikler ise bağımsız değişkenlerdir.

Tablo 2.1: Pima Indians Diabetes veri seti özellikleri.

Veri Nitelikleri	Açıklama	Veri Tipleri	Veri Türü
Pregnancies	Hamile Kalma Sayısı	integer	Sayısal
Glucose	Glikoz	integer	Sayısal
Blood Pressure	Kan basıncı	integer	Sayısal
Skin Thickness	Deri Kalınlığı	integer	Sayısal
Insulin	İnsülin	integer	Sayısal
BMI	Beden Kitle endeksi (Body Mass Index)	double	Sayısal
Diabetes Pedigree Function	Soyumuzdaki kişilere göre diyabet olma ihtimalimizi hesaplayan bir fonksiyon	double	Sayısal
Age	Yaş	integer	Sayısal
Outcome	Diyabet olup olmadığı bilgisi. (0 : Hayır / 1:Evet)	integer	Kategorik

2.2 VERİ SETİ DÜZENLEMESİ

“Pima Indians Diabetes” veri setini modelleme çalışmasında Python programlama dili kullanılmıştır. Bu dilin seçilmesindeki en önemli unsurlardan biri veri analizi, makine öğrenimi ve derin öğrenme gibi algoritmaların kolay kullanımını sağlayan birçok kütüphanesinin bulunmasıdır. Bu projede Python’ının os, pandas, numpy, sklearn, matplotlib ve seaborn kütüphaneleri etkin bir şekilde kullanılmıştır. Diğer taraftan bir başka neden de Python ile yapılmış birçok çalışma olmasıdır. Böylelikle araştırma ve geliştirmede kolaylık sağlanmıştır.

```
[ ] import os  
import pandas as pd  
import numpy as np
```

Şekil 2.1: İlk yüklenen Python kütüphaneleri

Proje kodlaması Google CoLab üzerinde gerçekleştirilmiştir. Google CoLab, kolay kullanımı ve her yerden ulaşılabilir yapısı ile etkili bir Online IDE’dir.

Veri seti, Kaggle platformundan elde edilmiştir. [3] Kaggle, Google bünyesinde olan online bir platformdur. Veri bilimcileri ve makine öğrenimi uygulayıcılarını bir araya getiren çevrimiçi bir topluluğa ev sahipliği yapar.

2.2.1 Veri Ön İşleme:

Veri Seti Yükleme

Veri ön işleme işlemlerine CSV formatındaki veri setinin CoLab’da yüklenmesi ile başlanmıştır. Kaggle’dan indirilen veri seti CoLab klasörüne yüklenmiştir. Daha sonra aşağıdaki Python kodlaması ile kaynak kodun içerisinde dahil edilmiştir.

```
[ ] dataset_path = "datasets/"  
data_path = os.path.join(dataset_path, 'diabetes.csv')  
ds = pd.read_csv(data_path, header=None)
```

Şekil 2.2: CSV formatındaki veri setine erişim

Veri Nitelik Adlarının Değiştirilmesi

Veri seti niteliklerinin isimleri Türkçe olarak değiştirilerek daha anlaşılır bir kodlama gerçekleştirilemesi amaçlandı.

```
ds.columns = ["Hamile_Kalma_Sayısı", "Glikoz", "Kan_Basıncı", "Deri_Kalınlığı",
    "İnsülin", "Beden_Kitle_Endeksi", "Genetiklik", "Yaş", "Diyabet"]
```

Şekil 2.3: Veri seti nitelik isimlerini Türkçe yapılması

14

Veri setimizin kaç satır ve kaç sütundan olduğunu (Şekil 2.4)'te gösterilmektedir.

```
[ ] ds.shape  
(769, 9)
```

Şekil 2.4: Veri seti satır, sütun bilgisi

Veri Tiplerinin Değiştirilmesi

(Şekil 2.5)' de veri seti nitelikleri ile ilgili bazı detay bilgiler gösterilmiştir.

```
[ ] ds.info()  
  
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 768 entries, 1 to 768  
Data columns (total 9 columns):  
 #   Column           Non-Null Count  Dtype     
---  --  
 0   Hamile_Kalma_Sayısı  768 non-null   object  
 1   Glikoz              768 non-null   object  
 2   Kan_Basıncı         768 non-null   object  
 3   Deri_Kalınlığı     768 non-null   object  
 4   İnsülin             768 non-null   object  
 5   Beden_Kitle_Endeksi 768 non-null   object  
 6   Genetiklik          768 non-null   object  
 7   Yaş                 768 non-null   object  
 8   Diyabet              768 non-null   object  
dtypes: object(9)  
memory usage: 54.1+ KB
```

Şekil 2.5: Veri seti nitelikleri hakkında bilgiler

Veri seti niteliklerinin “object” tipinde olması dikkat çekmiştir. Numpy kütüphanesini kullanılabilmek için veri tiplerinin “integer” ya da “float” olması gerekmektedir. Bu nedenle tüm veri tipleri “float” olarak güncellenmiştir.

```
[ ] for i in range(0, len(ds.columns)):
    ds[ds.columns[i]] = ds[ds.columns[i]].astype(float)

[ ] ds.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 1 to 768
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Hamile_Kalma_Sayısı 768 non-null   float64
 1   Glikoz             768 non-null   float64
 2   Kan_Basıncı        768 non-null   float64
 3   Deri_Kalınlığı     768 non-null   float64
 4   İnsülin            768 non-null   float64
 5   Beden_Kitle_Endeksi 768 non-null   float64
 6   Genetiklik          768 non-null   float64
 7   Yaş                768 non-null   float64
 8   Diyabet             768 non-null   float64
dtypes: float64(9)
memory usage: 54.1 KB
```

Şekil 2.6: Veri tipleri değiştirilmesi

Veri Tiplerinin İstatistiksel Değerleri

```
[ ] ds.describe().T
```

	count	mean	std	min	25%	50%	75%	max
Hamile_Kalma_Sayısı	768.0	3.845052	3.369578	0.000	1.00000	3.0000	6.00000	17.00
Glikoz	768.0	120.894531	31.972618	0.000	99.00000	117.0000	140.25000	199.00
Kan_Basıncı	768.0	69.105469	19.355807	0.000	62.00000	72.0000	80.00000	122.00
Deri_Kalınlığı	768.0	20.536458	15.952218	0.000	0.00000	23.0000	32.00000	99.00
İnsülin	768.0	79.799479	115.244002	0.000	0.00000	30.5000	127.25000	846.00
Beden_Kitle_Endeksi	768.0	31.992578	7.884160	0.000	27.30000	32.0000	36.60000	67.10
Genetiklik	768.0	0.471876	0.331329	0.078	0.24375	0.3725	0.62625	2.42
Yaş	768.0	33.240885	11.760232	21.000	24.00000	29.0000	41.00000	81.00
Diyabet	768.0	0.348958	0.476951	0.000	0.00000	0.0000	1.00000	1.00

Şekil 2.7: Temel istatistiksel değerlerin gösterilmesi

“Describe” metodu ile (Şekil 2.7)’ de görüldüğü üzere veri setimizin temel istatistik sonuçlarına ulaşmaktadır. Bu değerleri şöyle açıklayabiliriz:

Count : Veri seti niteliklerini kayıt sayısını verir.

Mean : Aritmetik ortalama değerini verir.

Std : Standart sapma değerini verir.

Min: Niteliğin kayıtları arasındaki en küçük değerini verir.

%25 : Nitelik değerlerini sıralar, dört eşit parçaya ayırr, ilk dilimin en büyük değerini verir.

%50 : Nitelik değerlerini sıralar, ortanca yani medyan değerini verir.

%75 : Nitelik değerlerini sıralar, dört eşit parçaya ayırr, üçüncü dilimin en büyük değerini verir.

Max. : Niteliğin kayıtları arasındaki en büyük değerini verir.

Veri Seti Boş Değer Analizi ve İşlemleri

Projenin bu noktasında önemli bir veri ön işleme kriteri olan boş değer analizini yapmak için tüm niteliklerin kaç tane “Null” değere sahip olduğu (Şekil 2.8) da görüldüğü gibi sorgulanmıştır.

```
[ ] ds.isnull().sum()
```

Hamile_Kalma_Sayısı	0
Glikoz	0
Kan_Basıncı	0
Deri_Kalınlığı	0
İnsülin	0
Beden_Kitle_Endeksi	0
Genetiklik	0
Yaş	0
Diyabet	0

dtype: int64

Şekil 2.8: Veri seti boş değer sorgulaması

Görüldüğü üzere veri setimizde hiç “Null” değerimiz yoktur. Kısaca niteliklerimizin tüm değerleri girilmiş durumda. Ancak (Şekil 2.7) de bazı alanların min değerinin sıfır olduğu görülmektedir. Ancak glikoz, kan basıncı, deri kalınlığı, insülin ve beden kitle endeksi niteliklerinin sıfır değer alması tıbben mümkün değildir.

Bu niteliklerin değerlerinin, veri setini hazırlayan uzmanlar tarafından elde olmayan nedenlerden dolayı test ölçümleri yapılamadığı ve veri seti bütünlüğünü sağlamak amacıyla değerlerin sıfır (0) girildiği kanaatine varılmıştır.

Projede oluşturulan modelin sağlıklı bir sonuca ulaşması için tıbben sıfır olamayacak değerlere ilişkin anlamlı bir işlem uygulamamız gerekmıştır. Bunun için ya sıfır değer içeren kayıtların veri setinden çıkarılması ve geri kalan sağlıklı veriler ile yola devam edilmesi ya da sıfır değerlerin anlamlı olabilecek değerler ile güncellenmesi ihtimalleri incelenmiştir.

Veri seti analiz edildiğinde kayıt sayısının oldukça düşük olduğu ve makine öğrenimi algoritmaları için yetersiz olacağı endişesi duyulmuştur. Bu nedenle veri silme işlemi düşünülmemiştir. Eğer veri silme kararı alınsaydı veri setinin %48 lik kısmı kaybedilecekti. Bu da modelimiz için irrasyonel bir tutum olacaktı. Veri analiz işlemlerinde veri silme işleminin son çare olarak değerlendirilmesi yaklaşımı tarafımızdan savunulmaktadır. Bu nedenle veri setinde güncelleme işlemi uygulanmıştır.

Öncelikle bu niteliklerin sıfır(0) değerleri “Null” değerlere çekmiştir. Null değerler daha sonra anlamlı değerler ile doldurulmuştur. (Şekil 2.9)’da görüldüğü üzere sıfır değerler “NaN” olarak güncellenmiştir.

```
[ ] ds_NaN = ["Glikoz","Kan_Basinci","Deri_Kalinligi","insulin","Beden_Kitle_Endeksi"]
[ ] for i in ds_NaN:
[ ]     ds[i] = ds[i].replace(to_replace=0, value=np.NaN)

[ ] ds.head()
```

	Hamble_Kalma_Sayisi	Glikoz	Kan_Basinci	Deri_Kalinligi	insulin	Beden_Kitle_Endeksi	Genetiklik	Yas	Diyabet
1	6.0	148.0	72.0	35.0	NaN	33.6	0.627	50.0	1.0
2	1.0	85.0	66.0	29.0	NaN	26.6	0.351	31.0	0.0
3	8.0	183.0	64.0	NaN	NaN	23.3	0.672	32.0	1.0
4	1.0	89.0	66.0	23.0	94.0	28.1	0.167	21.0	0.0
5	0.0	137.0	40.0	35.0	168.0	43.1	2.288	33.0	1.0

Şekil 2.9: Bazı alanların sıfır değerlerinin “NaN” yapılması

Bu işlemin ardından veri setimizdeki “Null” değerler sorgulandığında (Şekil 2.10) deki sonuç ile karşılaşılmıştır.

	[] ds.isnull().sum()
Hamile_Kalma_Sayısı	0
Glikoz	5
Kan_Basıncı	35
Deri_Kalınlığı	227
İnsülin	374
Beden_Kitle_Endeksi	11
Genetiklik	0
Yaş	0
Diyabet	0
dtype:	int64

Şekil 2.10: Veri seti niteliklerinin boş değer sayıları

Null değerleri doldurmak için (Şekil 2.11)' de görüldüğü üzere glikoz, kan basıncı, deri kalınlığı, insülin ve beden kitle endeksi alanları işleme alınmıştır. İşlemde null olan değerin diyabet özelliğindeki verisi kontrol edilmiştir. Null kaydın diyabet değeri karşılığı sıfır (0) ise veri setindeki diyabet değeri sıfır olan kayıtların ilgili nitelik değerlerinin aritmetik ortalaması alınarak null değer doldurulmuştur. Aynı şekilde diyabet değeri 1 olan null değerleri de yine diyabet değeri 1 olan kayıtların ilgili nitelik değerlerinin aritmetik ortalaması ile doldurulmuştur.

```
[ ] ds_NaN = ["Glikoz","Kan_Basıncı","Deri_Kalınlığı","İnsülin","Beden_Kitle_Endeksi"]
for i in ds_NaN:
    ds[i][(ds[i].isnull()) & (ds["Diyabet"] == 0)] = ds[i][(ds[i].isnull()) & (ds["Diyabet"] == 0)].fillna(ds[i][ds["Diyabet"] == 0].mean())
    ds[i][(ds[i].isnull()) & (ds["Diyabet"] == 1)] = ds[i][(ds[i].isnull()) & (ds["Diyabet"] == 1)].fillna(ds[i][ds["Diyabet"] == 1].mean())
```

Şekil 2.11: Boş değer içeren niteliklerin doldurulması

Böylece değeri null olan niteliklerin doldurulması esnasında bağımlı değişken olan diyabet alanı ile ilişkilendirilmiş ve kişinin diyabet olup olmaması dikkate alınmıştır. Veri setindeki boş değerler dengeli ve anlamlı bir şekilde doldurulmuştur.

Sonuç itibarıyle veri seti niteliklerinin null değer sayısını tekrar sorgulayarak yapılan doldurma işleminin ne derece gerçekleştiği (Şekil 2.12)' de sorgulanmıştır.

```
[ ] ds.isnull().sum()

Hamile_Kalma_Sayıısı      0
Glikoz                      0
Kan_Basıncı                  0
Deri_Kalınlığı                0
İnsülin                      0
Beden_Kitle_Endeksi          0
Genetiklik                   0
Yaş                          0
Diyabet                      0
dtype: int64
```

Şekil 2.12: Veri seti niteliklerinin güncellenmiş boş değer sayıları

Veri Seti Nitelikleri Arasındaki İlişki

Veri setinin nitelikleri arasındaki ilişkiyi Pearson korelasyon metodu ile gösterilmiştir. Böylece veri setimiz üzerinde ilişkisel olarak zayıf kalan ve model kapsamının dışında tutulan bir alan olup olmadığına karar verilmiştir.

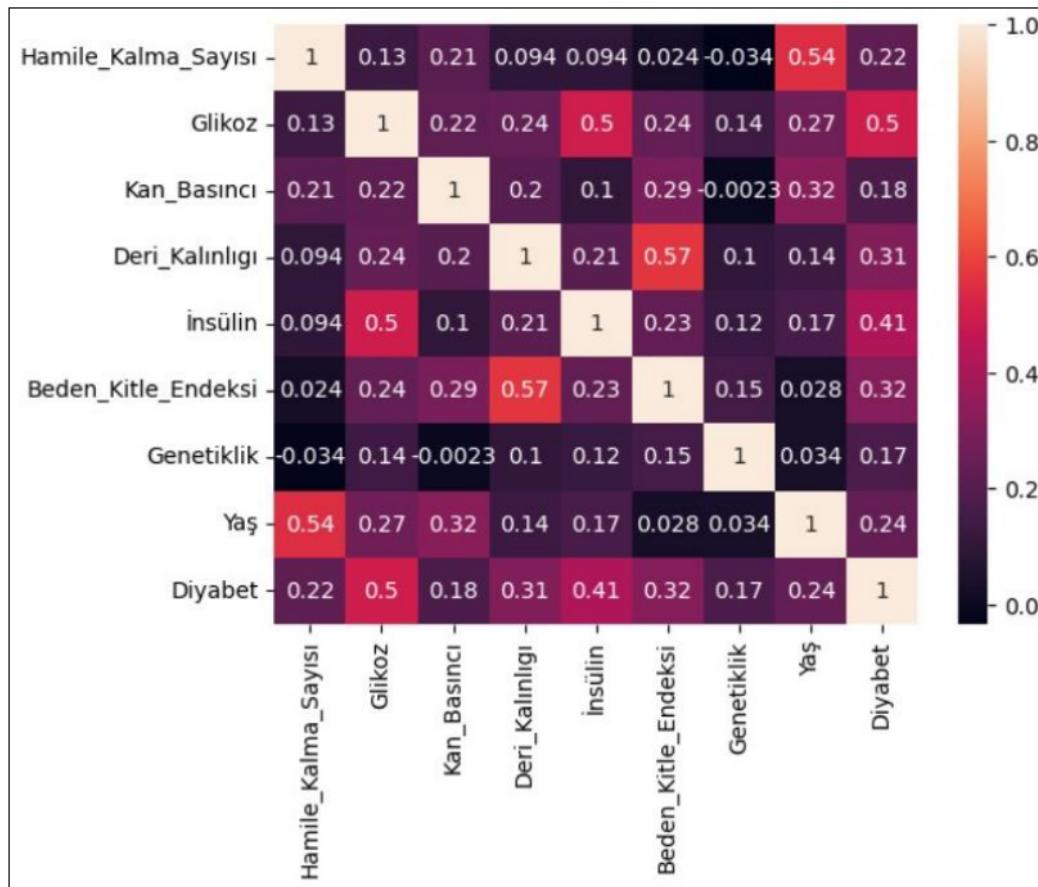
```
[19] cor = ds.corr()
cor
```

	Hamile_Kalma_Sayıısı	Glikoz	Kan_Basıncı	Deri_Kalınlığı	İnsülin	Beden_Kitle_Endeksi	Genetiklik	Yaş	Diyabet
Hamile_Kalma_Sayıısı	1.000000	0.129867	0.208935	0.094172	0.093546	0.024127	-0.033523	0.544341	0.221898
Glikoz	0.129867	1.000000	0.224356	0.235040	0.497789	0.235253	0.138125	0.268566	0.495954
Kan_Basıncı	0.208935	0.224356	1.000000	0.203453	0.099996	0.286518	-0.002264	0.324439	0.175087
Deri_Kalınlığı	0.094172	0.235040	0.203453	1.000000	0.212573	0.565443	0.102426	0.135916	0.308094
İnsülin	0.093546	0.497789	0.099996	0.212573	1.000000	0.231533	0.121716	0.165149	0.410918
Beden_Kitle_Endeksi	0.024127	0.235253	0.286518	0.565443	0.231533	1.000000	0.152530	0.027578	0.315271
Genetiklik	-0.033523	0.138125	-0.002264	0.102426	0.121716	0.152530	1.000000	0.033561	0.173844
Yaş	0.544341	0.268566	0.324439	0.135916	0.165149	0.027578	0.033561	1.000000	0.238356
Diyabet	0.221898	0.495954	0.175087	0.308094	0.410918	0.315271	0.173844	0.238356	1.000000

Şekil 2.13: Veri seti nitelikleri arasındaki korelasyon değerleri

Bu değerleri daha anlaşılır bir çerçeveden bakmak için seaborn kütüphanesi yüklenerek veri görselleştirilmiştir.

```
[22] import seaborn as sns
sns.heatmap(cor, annot = True)
```



Şekil 2.14: Korelasyon değerlerinin heatmap ile görselleştirilmesi ve çıktısı

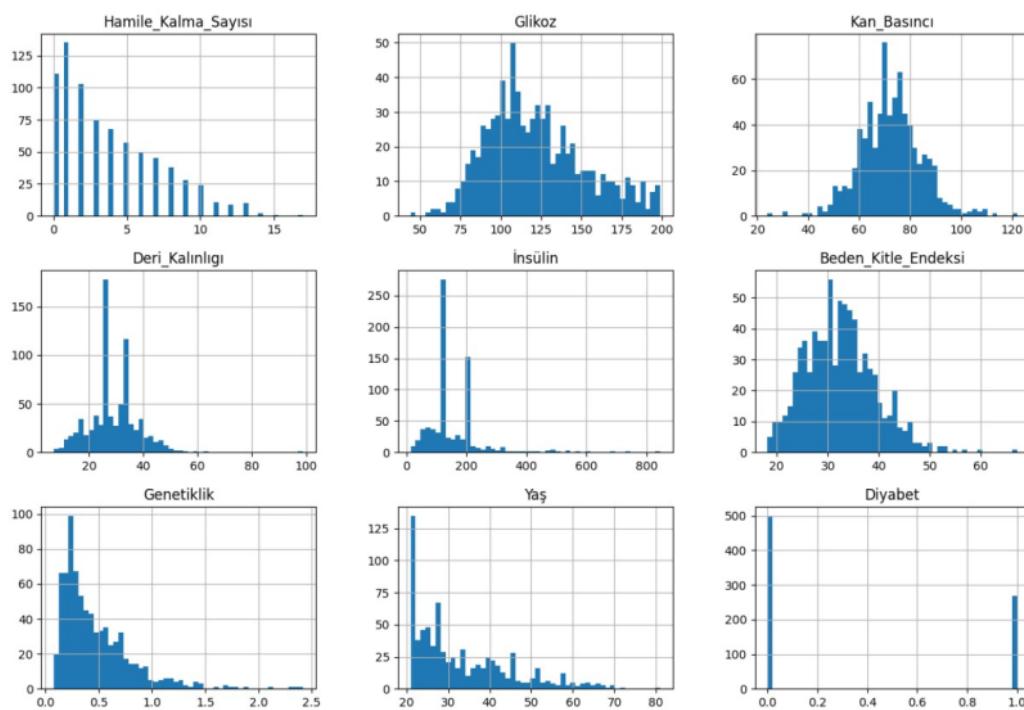
Bağımlı değişkenimiz “Diyabet” alanının bağımsız değişkenler ile korelasyonu incelendiğinde ısı haritasına göre en yüksek korelasyonun diyabet ile glikoz arasında olduğu görülmüştür. Daha sonra sırasıyla “İnsülin”, “Beden Kitle Endeksi”, “Deri Kalınlığı”, “Yaş”, “Hamile Kalma Sayısı”, “Kan basıncı” ve “Genetiklik” gelmektedir. Daha önce Pima Kızılderili popülasyonu üzerinde yapılan araştırmalarda T2DM nedenlerinin başında “genetiklik” varsayımları vardı. Ancak mevcut veri setimizdeki korelasyon “Genetiklik” özelliğini test sonuçlarında arka planda bırakmış gözükmemektedir.

Eldeki veriler ışığında tüm alanların modele dahil edilmesinde bir sakınca görülmemiştir.

Veri Setindeki Akyırı Değerlerin Tespiti

Aykırı değerler kurulan modelde sapmaya yol açan ve doğruluk değerini düşüren bir unsurdur. Dolayısıyla mutlaka aykırı değerler üzerinde işlem yapılması gereklidir. Veri seti üzerindeki boş değerler için yapılan öngöründe kayıt eksiltmesine karşı takınılan tutum, aynı şekilde aykırı değerler için de devam ettirilmiştir. Tespit edilecek aykırı değerlere ait kayıtları silmek yerine o değerler üzerinde işlem yapılması daha uygundur. Aykırı değerleri tespit için öncelikle veri seti niteliklerinin histogram görsellerini incelemekte faydalı görülmüştür. Histogram ile görselleştirme için Matplotlib kütüphanesi kullanılmıştır.

```
[ ] import matplotlib.pyplot as plt  
ds.hist(bins=50, figsize=(15, 10))  
plt.show()
```

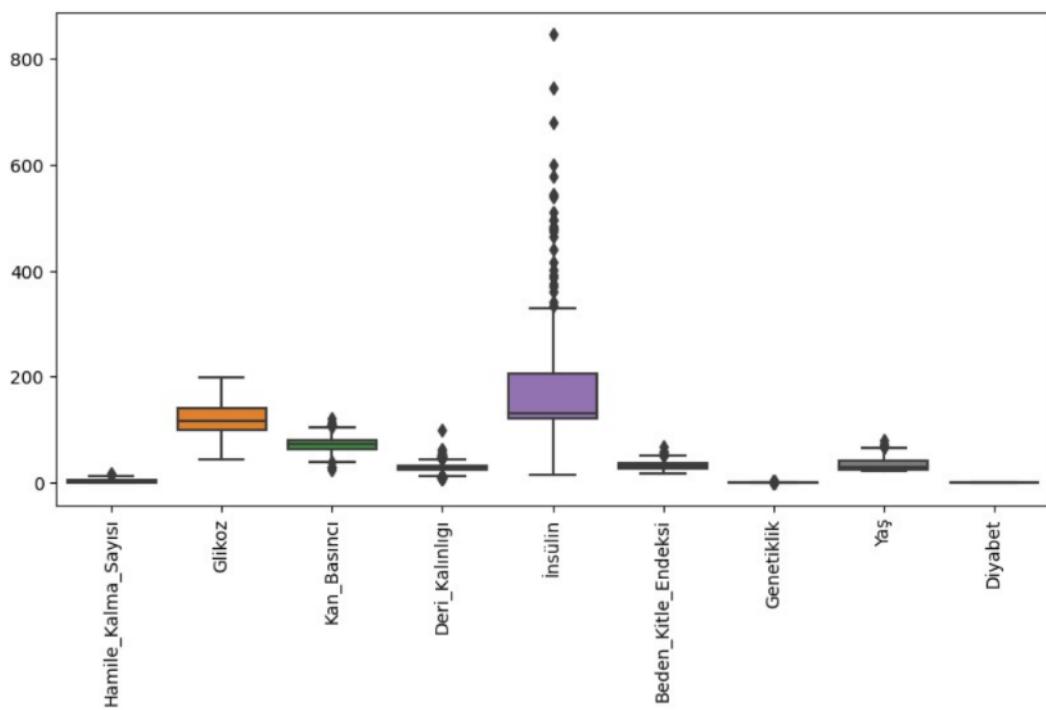


Şekil 2.15: Histogram görselinin oluşturulması ve çıktısı

Histogram değerleri incelendiğinde bazı nitelikler göze çarpmaktadır. Barların dağılımında dengesizlik olan niteliklerin başında insülin gelmektedir. Bu dikkat çekicidir. Peşinden onu deri kalınlığı ve yaş nitelikleri izlemektedir.

Ancak aykırı değerleri bulmak için histogram tek başına yeterli değildir. Kanaatimce aykırı değerleri en iyi gösteren görselleştirme aracı “boxplot” dır. Bu nedenle bir de boxplot penceresinden aykırı değerleri görmeye çalışalım.

```
[ ] plt.figure(figsize=(10,5))
p = sns.boxplot(data=ds, orient ="v", width=0.8)
plt.xticks(rotation=90)
```



Şekil 2.16: Boxplot görselinin oluşturulması ve çıktısı

(Şekil 2.16) de çok net olarak görülmektedir ki insülin değerlerinde gerçek anlamda aykırı değerler mevcuttur. Bu nedenle bu nitelik üzerinde işlem yapılmasına karar verilmiştir. Ancak kayıt silemeyeceğimiz için “Baskılama” metodu kullanılmıştır.

```
[ ] for i in ds:
    pColumn = ds[i]
    Q1 = pColumn.quantile(0.05)
    Q3 = pColumn.quantile(0.95)
    IQR = Q3 - Q1
    lower = Q1 - 1.5 * IQR
    upper = Q3 + 1.5 * IQR

    if ds[(pColumn > upper) | (pColumn < lower)].any( axis=None):
        print(i, " : Aykırı ", ds[(pColumn > upper) | (pColumn < lower) ].shape[0], ' tane değer var!')
        print(" lower : ",lower," \n upper : ",upper)
        ds.loc[pColumn > upper,i] = upper # baskılama
        print("====")
    else:
        print(i, " : Aykırı değer yok.")
        print("====")
```

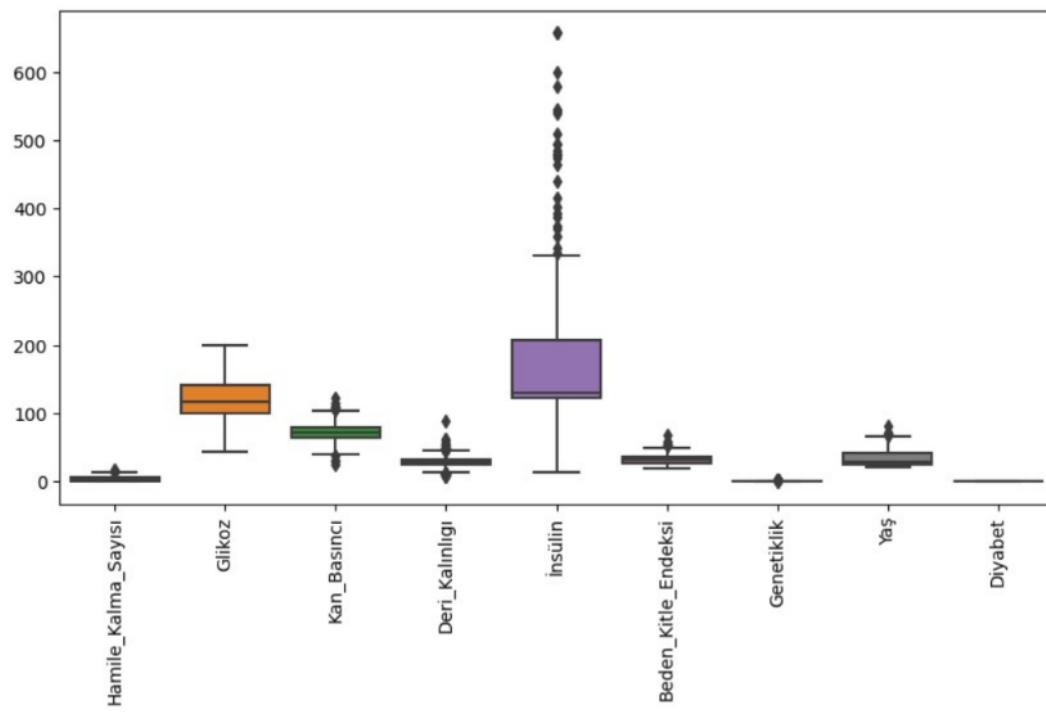
```
Hamile_Kalma_Sayısı : Aykırı değer yok.
=====
Glikoz : Aykırı değer yok.
=====
Kan_Basıncı : Aykırı değer yok.
=====
Deri_Kalınlığı : Aykırı 1 tane değer var!
lower : -30.12499999999993
upper : 88.475
=====
İnsülin : Aykırı 3 tane değer var!
lower : -314.5
upper : 657.5
=====
Beden_Kitle_Endeksi : Aykırı değer yok.
=====
Genetiklik : Aykırı değer yok.
=====
Yaş : Aykırı değer yok.
=====
Diyabet : Aykırı değer yok.
=====
```

Şekil 2.17: Baskılama metodunun kodlaması ve çıktısı

Veri seti niteliklerine ait kayıtların aykırı değerlerini bulacak kodlama için (Şekil 2.17)'i inceleyiniz. Burada belirtilen marjlar dahilindeki maksimum ve minimum değerler tespit edilmiş ve bu değerlerin dışındaki kayıtlara aykırı olarak nitelendirilmiştir.

Kandaki insülin direnci (Şekil 2.17)' de görüleceği üzere her biri 800 – 850 bandında yer alan 3 aykırı değerden oluşmaktadır. Baskılama yöntemi ile veri setinde kabul edilebilir marjlar içerisinde kalan en yüksek değer ile aykırı değerler güncellenmiştir. Aynı işlem deri kalınlığı içinde yapılmıştır.

```
[ ] plt.figure(figsize=(10,5))
p = sns.boxplot(data=ds, orient = "v", width=0.8)
plt.xticks(rotation=90)
```



Şekil 2.18: Baskılama metodunun kontrol edilmesi

(Şekil 2.16) de görüleceği üzere insülin direnci 800 -850 bandındayken (Şekil 2.18) da üst band 600 – 650 bandına gerilemiştir. Böylece baskılama yöntemi ile aykırı kayıtlar kabul edilebilir marjlar içeresine çekilmiştir.

Veri Setine Özellik Ekleme

Veri setimizde yer alan “Glikoz”, “İnsülin” ve “Beden Kitle Endeksi” özelliklerimizin değerleri, modelimizdeki istatistiksel ve matematiksel yaklaşımalar göz önüne alınarak kategorik veriye dönüştürülmüştür. Böylece modelimizin aşırı öğrenme girdabına girmesi engellenmeye çalışılmıştır.

Glikoz :

```
[ ] Score_Ranges = [0,70,99,126,200]
Score_Names = ['Düşük','Normal','Gizli','Yüksek']
ds['GLKZ'] = pd.cut(x=ds['Glikoz'], bins=Score_Ranges, labels=Score_Names)

[ ] ds.head()
```

	Hamile_Kalma_Sayısı	Glikoz	Kan_Basınçlı	Deri_Kalınlığı	İnsülin	Beden_Kitle_Endeksi	Genetiklik	Yaş	Diyabet	GLKZ
1	6.0	148.0	72.0	35.0	206.846154	33.6	0.627	50.0	1.0	Yüksek
2	1.0	85.0	66.0	29.0	130.287879	26.6	0.351	31.0	0.0	Normal
3	8.0	183.0	64.0	33.0	206.846154	23.3	0.672	32.0	1.0	Yüksek
4	1.0	89.0	66.0	23.0	94.000000	28.1	0.167	21.0	0.0	Normal
5	0.0	137.0	40.0	35.0	168.000000	43.1	2.288	33.0	1.0	Yüksek

Şekil 2.19: Glikoz kategorik özellik eklemesi

İnsülin:

```
[ ] def set_insulin(row):
    if row["İnsülin"] >= 16 and row["İnsülin"] <= 166:
        return "Normal"
    else:
        return "Anormal"

ds = ds.assign(KID = ds.apply(set_insulin, axis=1))
```

```
[ ] ds.head()
```

	Hamile_Kalma_Sayısı	Glikoz	Kan_Basınçlı	Deri_Kalınlığı	İnsülin	Beden_Kitle_Endeksi	Genetiklik	Yaş	Diyabet	GLKZ	KID
1	6.0	148.0	72.0	35.0	206.846154	33.6	0.627	50.0	1.0	Yüksek	Anormal
2	1.0	85.0	66.0	29.0	130.287879	26.6	0.351	31.0	0.0	Normal	Normal
3	8.0	183.0	64.0	33.0	206.846154	23.3	0.672	32.0	1.0	Yüksek	Anormal
4	1.0	89.0	66.0	23.0	94.000000	28.1	0.167	21.0	0.0	Normal	Normal
5	0.0	137.0	40.0	35.0	168.000000	43.1	2.288	33.0	1.0	Yüksek	Anormal

Şekil 2.20: İnsülin kategorik özellik eklemesi

Beden Kitle Endeksi:

```
[ ] Score_Ranges = [0,18.5,25,30,100]
Score_Names = ['Zayıf','Sağlıklı','Kılolu','Obez']
ds['BKE'] = pd.cut(x=ds['Beden_Kitle_Endeksi'], bins=Score_Ranges, labels=Score_Names)

[ ] ds.head()
```

	Hamile_Kalma_Sayısı	Glikoz	Kan_Basıncı	Deri_Kalınlığı	İnsülin	Beden_Kitle_Endeksi	Genetiklik	Yaş	Diyabet	GLKZ	KID	BKE
1	6.0	148.0	72.0	35.0	206.846154	33.6	0.627	50.0	1.0	Yüksek	Anormal	Obez
2	1.0	85.0	66.0	29.0	130.287879	26.6	0.351	31.0	0.0	Normal	Normal	Kılolu
3	8.0	183.0	64.0	33.0	206.846154	23.3	0.672	32.0	1.0	Yüksek	Anormal	Sağlıklı
4	1.0	89.0	66.0	23.0	94.000000	28.1	0.167	21.0	0.0	Normal	Normal	Kılolu
5	0.0	137.0	40.0	35.0	168.000000	43.1	2.288	33.0	1.0	Yüksek	Anormal	Obez

Şekil 2.21: Beden Kitle Endeksi kategorik özellik eklemesi

Veri Ölçeklendirme

Veri setimize eklediğimiz kategorik özelliklerin sayısallaştırılması elimizdeki veri setinin modelimizde kullanacağımız algoritmalarla uygun bir hale getirilmesini sağlamıştır. Sayısallaştırma işlemi için “Label Encoder” kullanılmıştır.

```
[ ] from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
```

Şekil 2.22: Label Encoder ve Standard Scaler fonksiyonlarının yüklenmesi.

```
[ ] laben = LabelEncoder()
for i in range(0, ds.shape[1]):
    if ds.dtypes[i] == "category" or ds.dtypes[i] == "object":
        ds[ds.columns[i]] = laben.fit_transform(ds[ds.columns[i]])

[ ] ds.head()
```

	Hamile_Kalma_Sayısı	Glikoz	Kan_Basıncı	Deri_Kalınlığı	İnsülin	Beden_Kitle_Endeksi	Genetiklik	Yaş	Diyabet	GLKZ	KID	BKE
1	6.0	148.0	72.0	35.0	206.846154	33.6	0.627	50.0	1.0	3	0	1
2	1.0	85.0	66.0	29.0	130.287879	26.6	0.351	31.0	0.0	2	1	0
3	8.0	183.0	64.0	33.0	206.846154	23.3	0.672	32.0	1.0	3	0	2
4	1.0	89.0	66.0	23.0	94.000000	28.1	0.167	21.0	0.0	2	1	0
5	0.0	137.0	40.0	35.0	168.000000	43.1	2.288	33.0	1.0	3	0	1

Şekil 2.23: Label Encoder işlemi ve çıktısı.

Veri Standardizasyonu

Veri setimizdeki değerler, modelimizde yer alan makine öğrenimi algoritmalarının en verimli halde çalışabilmesi için ortalaması sıfır(0) ve standart sapması 1 olacak şekilde normalize edilmiştir. Bu şekilde modelin performansının artırılması amaçlanmıştır.

Bu normalizasyon işlemi bağımsız değişkenler arasında yapılmıştır. Bu nedenle bağımlı değişkenimiz olan “Diyabet” özelliği veri setinden ayrılarak bir değişkene atanmıştır. Hedef değişkenimiz zaten sıfır(0) ve 1’ lerden oluşan sayısal bir değişkendir.

```
[ ] ds_data = ds.drop(["Diyabet"], axis=1)
ds_target = ds.Diyabet
ds_data.shape, ds_target.shape

((768, 11), (768,))

[ ] ds_target.head()

1    1.0
2    0.0
3    1.0
4    0.0
5    1.0
Name: Diyabet, dtype: float64
```

Şekil 2.24: Hedef değişkenimizin veri setinden ayrılması.

```
[ ] scaler = StandardScaler()
ds_data.iloc[:, :] = scaler.fit_transform(ds_data.iloc[:, :])

[ ] ds_data.head()
```

	Hamile_Kalma_Sayısı	Glikoz	Kan_Basıncı	Deri_Kalınlığı	İnsülin	Beden_Kitle_Endeksi	Genetiklik	Yaş	GLKZ	KID	BKE
1	0.639947	0.864020	-0.035389	0.653933	0.582285	0.167806	0.468492	1.425995	1.139446	-1.350381	0.133631
2	-0.844885	-1.205478	-0.531332	-0.026460	-0.305207	-0.850452	-0.365061	-0.190672	0.026094	0.740532	-1.469937
3	1.233880	2.013741	-0.696647	0.427135	0.582285	-1.330487	0.604397	-0.105584	1.139446	-1.350381	1.737198
4	-0.844885	-1.074081	-0.531332	-0.706854	-0.725870	-0.632253	-0.920763	-1.041549	0.026094	0.740532	-1.469937
5	-1.141852	0.502679	-2.680419	0.653933	0.131966	1.549727	5.484909	-0.020496	1.139446	-1.350381	0.133631

Şekil 2.25: Standard Scaler çalışması ve çıktısı.

3. MODELLEME

3.1.1 Veri Setini Bölme:

Oluşturduğumuz modelin en doğru şekilde çalışabilmesi için öğrenmesi, öğrenciklerinin test edilmesi ve doğruluğunun ölçülmesi gerekir. Bu nedenle elimizdeki veri seti ³ eğitim ve test verisi olarak iki parçaya ayrılmıştır. Veri setimizin kayıt sayısının düşük olması dolayısıyla eğitime bölümüne yüzde olarak daha çok yer ayrılmasına karar verilmiştir. Bu nedenle eğitim verisi %80 olarak belirlenmiş, test verisi ise %20 olarak ayrılmıştır.

```
[ ] from sklearn.model_selection import train_test_split  
  
[ ] X_train, X_test, y_train, y_test = train_test_split(ds_data, ds_target, test_size= 0.20, random_state=42)  
  
[ ] print("x_train : ", X_train.shape)  
print("y_train : ", y_train.shape)  
print("X test : ", X_test.shape)  
print("y_test : ", y_test.shape)  
  
X_train : (614, 11)  
y_train : (614,)  
X_test : (154, 11)  
y_test : (154,)
```

Şekil 3.1: train_test_split fonksiyonunun yüklenmesi ve veri bölme işlemi

(Şekil 3.1)' de veri setimizi 80-20 oranlarında bölümlemiştir.

3.1.2 Algoritmalar:

```
[43] from sklearn.model_selection import GridSearchCV, cross_val_score  
from sklearn.metrics import accuracy_score  
from sklearn.linear_model import LogisticRegression  
from sklearn.neighbors import KNeighborsClassifier  
from sklearn.svm import SVC  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.ensemble import RandomForestClassifier  
from sklearn.ensemble import GradientBoostingClassifier  
from lightgbm import LGBMClassifier  
from sklearn.model_selection import KFold  
from xgboost import XGBClassifier
```

Şekil 3.2: Algoritma kütüphanelerinin yüklemesi

(Şekil 3.2) de görüldüğü gibi algoritmalar yüklenikten sonra (Şekil.3.3) de “models” adında bir table kurgulanıp, her algoritma için bir isimlendirme yapılmış ve birinci eleman “algoritma ismi”, ikinci eleman “algoritma” şeklinde bir yapı oluşturulmuştur.

```
[44] models = []

    models.append(("LR", LogisticRegression()))
    models.append(("KNN", KNeighborsClassifier()))
    models.append(("SVR", SVC()))
    models.append(("CART", DecisionTreeClassifier()))
    models.append(("RandomForests", RandomForestClassifier()))
    models.append(("GradientBoosting", GradientBoostingClassifier()))
    models.append(("XGBoost", XGBClassifier()))
    models.append(("Light GBM", LGBMClassifier()))
```

Şekil 3.3: Algoritmaların Model dizinine eklenmesi

(Şekil 3.4)' da görüldüğü üzere for döngüsü sayesinde “models” table' nin içerisinde gezinerek her sınıflandırma algoritması için doğruluk (accuracy) değeri tespit edilmiştir.

```
[45] for name,model in models:
    mod = model.fit(x_train,y_train) # Trainleri modele fit etmek
    y_pred = mod.predict(x_test) # Tahmin

    acc = accuracy_score(y_test, y_pred) # RMSE hesabi
    cvscore = cross_val_score(model, ds_data,ds_target, cv = 10).mean()

    print("Holdout Method:",end=" ")
    print(name,acc) #yazdırılacak kısım

    print("Cross Val Score :",end=" ")
    print(name,cvscore)
    print("-----")
```

Şekil 3.4: Modelin çalıştırılması

```
Holdout Method: LR 0.8701298701298701
Cross Val Score : LR 0.8449760765550239
-----
Holdout Method: KNN 0.8311688311688312
Cross Val Score : KNN 0.817634996582365
-----
Holdout Method: SVR 0.8311688311688312
Cross Val Score : SVR 0.8371496924128502
-----
Holdout Method: CART 0.81818181818182
Cross Val Score : CART 0.8581339712918661
-----
Holdout Method: RandomForests 0.8571428571428571
Cross Val Score : RandomForests 0.8749829118250171
-----
Holdout Method: GradientBoosting 0.8766233766233766
Cross Val Score : GradientBoosting 0.8984791524265209
-----
Holdout Method: XGBoost 0.8766233766233766
Cross Val Score : XGBoost 0.8959159261790839
-----
Holdout Method: Light GBM 0.8961038961038961
Cross Val Score : Light GBM 0.8880724538619275
```

Şekil 3.5: Model çıktısı

“Holdout Method” olarak isimlendirilen değerler eğitim ve test verisi olarak ayırdığımız veri grupları kullanılarak oluşturulan doğruluk değerleridir. “Cross Val Score” değeri ise tüm veri setini 10 parçaaya bölüp, ortalamasını hesaplanarak tespit edilen doğruluk değerleridir.

3.1.3 Değerlendirme:

- Algoritmaların sonuçları hem “hold-out metod” hem de “Cross Validation Score” kullanılarak doğruluk değerleri karşılaştırılmıştır.
- Algoritmaların çoğunuğunda “Cross Validation Score” daha iyi sonuç vermiştir.
- Hold-Out Metod olarak en iyi sonucu %89,6 lik doğruluk orANIyla “Light GBM” algoritması sağlamıştır. Cross Val Score olarak’ ta en iyi sonucu %89,84 ile “Gradient Boosting” algoritması görülmüştür.

4. SONUÇ

Prjede Pima Kızılderililerine ait veri seti makine öğrenimi algoritmaları kullanılarak ele alınmıştır. Diyabet hastalığının teşhis edilmesi için kurulan modelde doğruluk derecesi en yüksek olan makine öğrenimi algoritmaları tespit edilmiştir. Bu çalışma amacına yönelik olarak kurduğu modelde minimum %89.6 lık bir doğruluk değerine erişmiştir. Bu değer nispeten yüksek bir değerdir. Bu doğrultuda projenin amacına ulaştığı düşünülmektedir.

Proje, keşifsel veri analizi çatısı altında yürütülen veri ön işleme adımlarından ve veri modelleme bölümlerinden oluşmaktadır. Çalışma içerisinde Python kütüphaneleri etkin biçimde kullanılmıştır. Veri görselleştirme ile analizler güçlendirilmiş ve karar noktalarında etkili olmuştur. Kullanılan teknikler güncel ve gelişmeye açıktır.

Veri analizinin önemi gelişen teknoloji ile birlikte hızla artmaktadır. Web üzerinde birçok veri analizi çalışması mevcuttur. Projede kullanılan “Pima Indians Diabetes” veri seti üzerinde de birçok çalışma yapıldığı görülmektedir. Yapılan incelemeler neticesinde takip edilen web çalışmalarının sonuçlarına göre bu projede elde edilen doğruluk oranları ile yaklaşık olarak örtüşüğü ve hakim makine öğrenimi algoritmalarının genelde boosting algoritmaları olduğu gözlemlenmiştir.

Veri setinin kayıt sayısının düşüklüğü çalışma açısından bir dezavantaj gibi görülse de veri kaynağını orijinalliği bu açığı kapatmak için yeterli bulunmuştur. Ancak yine de diyabet hastalığının teşhisini için daha büyük çaplı bir veri seti üzerinde çalışma yapılması önerilmektedir. Bu projede yapılan çalışmalar yeni projeler için referans niteliği taşıyabilir. Modelleme büyük veriler üzerinde denenebilir.

Çağımızın yaygın hastalığı diyabet için gelecekte makine öğrenimi ve derin öğrenme algoritmaları ile geliştirilecek modeller vasıtıyla, erken teşhisin yapılabileceği ve belki de zamanında müdahale ile hastanın tamamen iyileştirilebileceği bir sürecin içerisinde bu projenin de çok az da olsa bir katkısının olmasından dolayı mutluluk duymaktayım.

REFERANSLAR

- [1] Pauls, Elizabeth Prine (2007), *Pima*, Britannica
<https://www.britannica.com/topic/Pima-people> (Erişim Tarihi: 15.05.2023)
- [2] Wikipedia, *Akimel O'odhma Pima*,
https://en.wikipedia.org/wiki/Akimel_O%27odham#Customs,
(Erişim Tarihi: 15.05.2023)
- [3] ⁸ Kaggle, <https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database>
(Erişim Tarihi : 23.04.2023)
- [4] Emre Sırma, *Tahmin Modellerinin Karşılaştırılması*, Kaggle
<https://www.kaggle.com/code/emresirma/tahmin-modellerinin-kar-la-t-r-lmas#Veri-Ön-İsleme> (Erişim Tarihi : 17.05.2023)
- [5] ¹⁵ Maşide Leyla Tilki (2020), *Label Encoder ve OneHotEncoder Karşılaştırması*, Medium,
<https://medium.com/operations-management-t%C3%BCrkkiye/label-encoder-ve-onehotencoder-kar%C4%91laştırmas%C4%B1-c0983e884fc5>, (Erişim Tarihi: 18.05.2023)
- [6] Saltuk Bugra Karacan (2020), *Diyabet Veri Seti ile Hastalık Tahmini ve Accuracy Değerini Yükseltme Yolları*, Medium
<https://sbkaracan.medium.com/diyabet-veri-seti-ile-hastalik-tahmini-ve-accuracy-degerini-yükseltme-yolları-722f0bd87465> (Erişim Tarihi: 17.05.2023)
- [7] ² İsmail Bugra Bölükbaşı (2023), *Dengesiz Bir Diyabet Veri Setinde Makine Öğrenmesi Yöntemlerini Kullanarak Diyabet Hastalığının Teşhisini*, Yüksek Lisans Tezi, Bursa Uludağ Üniversitesi Fen Bilimleri Enstitüsü
- [8] ³ İsmail Bugra Bölükbaşı (2023), *Dengesiz Bir Diyabet Veri Setinde Makine Öğrenmesi Yöntemlerini Kullanarak Diyabet Hastalığının Teşhisini*, Yüksek Lisans Tezi, Bursa Uludağ Üniversitesi Fen Bilimleri Enstitüsü

PRIMARY SOURCES

1	Submitted to Altinbas University Student Paper	2%
2	sbkaracan.medium.com Internet Source	1 %
3	acikerisim.uludag.edu.tr Internet Source	1 %
4	openaccess.altinbas.edu.tr Internet Source	1 %
5	docs.neu.edu.tr Internet Source	<1 %
6	www.jove.com Internet Source	<1 %
7	www.ijert.org Internet Source	<1 %
8	rstudio-pubs-static.s3.amazonaws.com Internet Source	<1 %
9	uu.diva-portal.org Internet Source	<1 %

10	app.trdizin.gov.tr Internet Source	<1 %
11	urn.nsk.hr Internet Source	<1 %
12	www.jstage.jst.go.jp Internet Source	<1 %
13	www.mdpi.com Internet Source	<1 %
14	www.researchgate.net Internet Source	<1 %
15	www.xnovate.org Internet Source	<1 %
16	Zhang - Encyclopedia of Global Health (globalhealth) Publication	<1 %

Exclude quotes Off
Exclude bibliography Off

Exclude matches Off