from \_\_future\_\_ import print\_function

import os

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

#from pandas.tools import plotting

from scipy import stats

#plt.style.use("ggplot")

import warnings

warnings.filterwarnings("ignore")

from scipy import stats

%matplotlib inline

from sklearn.ensemble import RandomForestClassifier #for the model

from sklearn.tree import DecisionTreeClassifier

from sklearn.tree import export\_graphviz #plot tree

from sklearn.metrics import roc\_curve, auc #for model evaluation

from sklearn.metrics import classification\_report #for model evaluation

from sklearn.metrics import confusion\_matrix #for model evaluation

from sklearn.model\_selection import train\_test\_split #for data splitting

Kütüphaneleri çağırdım.

# This Python 3 environment comes with many helpful analytics libraries installed

# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python

# For example, here's several helpful packages to load

import numpy as np # linear algebra

import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)

# Input data files are available in the read-only "../input/" directory

# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os

for dirname, \_, filenames in os.walk('/kaggle/input'):

    for filename in filenames:

        print(os.path.join(dirname, filename))

heart\_disease\_file\_path='/content/576697\_1043970\_bundle\_archive.zip'

heart\_disease\_data= pd.read\_csv(heart\_disease\_file\_path)

heart\_disease\_data.describe()

Kullanacağım veri setini yükledim.

print('rows, columns:', heart\_disease\_data.shape)

heart\_disease\_data.columns=["Age", "Sex", "Chest Pain Type","Resting Blood Pressure","Serum Cholestoral (mg/dl)",

 "Fasting Blood Sugar","Resting Electrocardiographic Results","Maximum Heart Rate Achieved", "Exercise Induced Angina",

"ST Depression Induced by Exercise Relative to Rest","The Slope of the Peak Exercise ST Segment","Number of Major Vessels",

"Thalium","Condition"]

Halihazırda bulunan sütunları yeniden adlandırdım.

heart\_disease\_data.info()

heart\_disease\_data

classes=heart\_disease\_data['Condition']

features =heart\_disease\_data.iloc[:,13:]

print(features)

heart\_disease\_data = pd.get\_dummies(heart\_disease\_data, drop\_first=True)

heart\_disease\_data = pd.get\_dummies(heart\_disease\_data, drop\_first=True)

heart\_disease\_data['Condition'][heart\_disease\_data['Condition'] == 0] = 'No Disease'

heart\_disease\_data['Condition'][heart\_disease\_data['Condition'] == 1] = 'Disease'

#Hedef sütun hastalık olup olmadığını gösteren sütundu.

print(classes.value\_counts())

plt.figure(figsize=(4,4))

sns.countplot(classes,label="Case Number")

plt.show()

#Hasta olan ve olmayan birey sayısını ayırdım.

#Bazı etiketleme ve grafik işlemleri yaptım.

male =len(heart\_disease\_data[heart\_disease\_data['Sex'] == 1])

female = len(heart\_disease\_data[heart\_disease\_data['Sex']== 0])

plt.figure(figsize=(8,6))

# Data to plot

labels = 'Male','Female'

sizes = [male,female]

colors = ['skyblue', 'yellowgreen']

explode = (0, 0)  # explode 1st slice

# Plot

plt.pie(sizes, explode=explode, labels=labels, colors=colors,

autopct='%1.1f%%', shadow=True, startangle=90)

plt.axis('equal')

plt.show()

plt.figure(figsize=(8,6))

# Data to plot

labels = "Chest Pain Type:0(Typical Angina)",'Chest Pain Type:1(Atypical Angina)','Chest Pain Type:2(Non-anginal pain)','Chest Pain Type:3(Asymptomatic)'

sizes = [len(heart\_disease\_data[heart\_disease\_data['Chest Pain Type'] == 0]),

         len(heart\_disease\_data[heart\_disease\_data['Chest Pain Type'] == 1]),

         len(heart\_disease\_data[heart\_disease\_data['Chest Pain Type'] == 2]),

         len(heart\_disease\_data[heart\_disease\_data['Chest Pain Type'] == 3])]

colors = ['skyblue', 'yellowgreen','orange','gold']

explode = (0, 0,0,0)  # explode 1st slice

# Plot

plt.pie(sizes, explode=explode, labels=labels, colors=colors,

autopct='%1.1f%%', shadow=True, startangle=180)

plt.axis('equal')

plt.show()

plt.figure(figsize=(8,6))

# Data to plot

labels = 'Fasting Blood Sugar < 120 mg/dl','Fasting Blood Sugar > 120 mg/dl'

sizes = [len(heart\_disease\_data[heart\_disease\_data['Fasting Blood Sugar'] == 0]),len(heart\_disease\_data[heart\_disease\_data['Chest Pain Type'] == 1])]

colors = ['skyblue', 'yellowgreen','orange','gold']

explode = (0.1, 0)  # explode 1st slice

# Plot

plt.pie(sizes, explode=explode, labels=labels, colors=colors,

autopct='%1.1f%%', shadow=True, startangle=180)

plt.axis('equal')

plt.show()

plt.figure(figsize=(8,6))

# Data to plot

labels = 'No','Yes'

sizes = [len(heart\_disease\_data[heart\_disease\_data['Exercise Induced Angina'] == 0]),len(heart\_disease\_data[heart\_disease\_data['Exercise Induced Angina'] == 1])]

colors = ['skyblue', 'yellowgreen']

explode = (0.1, 0)  # explode 1st slice

# Plot

plt.pie(sizes, explode=explode, labels=labels, colors=colors,

autopct='%1.1f%%', shadow=True, startangle=90)

plt.axis('equal')

plt.show()

sns.set\_style('whitegrid')

plt.figure(figsize=(14,8))

sns.heatmap(heart\_disease\_data.corr(), annot = True, cmap='coolwarm',linewidths=.1)

plt.show()

Korelasyon incelemesi yaptım.

sns.distplot(heart\_disease\_data['Maximum Heart Rate Achieved'],kde=False,bins=30,color='violet')

Verinin daha iyi anlaşılması için farklı grafikleme işlemleri daha yaptım.

sns.distplot(heart\_disease\_data['Serum Cholestoral (mg/dl)'],kde=False,bins=30,color='red')

plt.show()

sns.distplot(heart\_disease\_data['Resting Blood Pressure'],kde=False,bins=30,color='blue')

plt.show()

plt.figure(figsize=(15,6))

sns.countplot(x='Age',data = heart\_disease\_data, hue = 'Condition',palette='GnBu')

plt.show()

plt.figure(figsize=(8,6))

sns.scatterplot(x='Serum Cholestoral (mg/dl)',y='Maximum Heart Rate Achieved',data=heart\_disease\_data,hue='Condition')

plt.show()

plt.figure(figsize=(8,6))

sns.scatterplot(x='Resting Blood Pressure',y='Maximum Heart Rate Achieved',data=heart\_disease\_data,hue='Condition')

plt.show()

X\_train, X\_test, y\_train, y\_test = train\_test\_split(heart\_disease\_data.drop('Condition', 1), heart\_disease\_data['Condition'], test\_size = .2, random\_state=10)

Model eğitilmesini sağladım.

model = RandomForestClassifier(max\_depth=5)

model.fit(X\_train, y\_train)

estimator = model.estimators\_[1]

feature\_names = [i for i in X\_train.columns]

y\_train\_str = y\_train.astype('str')

y\_train\_str[y\_train\_str == '0'] = 'no disease'

y\_train\_str[y\_train\_str == '1'] = 'disease'

y\_train\_str = y\_train\_str.values

export\_graphviz(estimator, out\_file='tree.dot',

                feature\_names = feature\_names,

                class\_names = y\_train\_str,

                rounded = True, proportion = True,

                label='root',

                precision = 2, filled = True)

from subprocess import call

call(['dot', '-Tpng', 'tree.dot', '-o', 'tree.png', '-Gdpi=600'])

from IPython.display import Image

Image(filename = 'tree.png')

y\_predict = model.predict(X\_test)

y\_pred\_quant = model.predict\_proba(X\_test)[:, 1]

y\_pred\_bin = model.predict(X\_test)

Hata matrisini belirledim.

confusion\_matrix = confusion\_matrix(y\_test, y\_pred\_bin)

confusion\_matrix

total=sum(sum(confusion\_matrix))

sensitivity = confusion\_matrix[0,0]/(confusion\_matrix[0,0]+confusion\_matrix[1,0])

print('Sensitivity : ', sensitivity )

specificity = confusion\_matrix[1,1]/(confusion\_matrix[1,1]+confusion\_matrix[0,1])

print('Specificity : ', specificity)

from sklearn.ensemble import RandomForestClassifier

start = time.time()

clf = RandomForestClassifier()

clf.fit(X\_train, y\_train)

prediction\_rf = clf.predict(X\_test)

scores = cross\_val\_score(clf, X, y, cv=5)

end = time.time()

print("Random Forest Classifier Accuracy: {0:.2%}".format(accuracy\_score(prediction\_rf, y\_test)))

print("Cross Validation Score {0:.2%} (+/- {1:.2%})".format(np.mean(scores), np.std(scores)\*2))

print("Execution Time: {0:.5} seconds \n".format(end-start))

target\_names = ['No Disease', 'Disease']

print(classification\_report(y\_test, prediction\_rf, target\_names=target\_names))

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

X\_train = sc.fit\_transform(X\_train)

X\_test = sc.transform(X\_test)

Programın doğruluk oranlarını belirledim.

from sklearn.neighbors import KNeighborsClassifier

start = time.time()

clf = KNeighborsClassifier()

clf.fit(X\_train, y\_train)

prediction\_knn\_sc = clf.predict(X\_test)

scores = cross\_val\_score(clf, X, y, cv=5)

end = time.time()

print("KNN Classifier Accuracy: {0:.2%}".format(accuracy\_score(prediction\_knn\_sc, y\_test)))

print("Cross Validation Score: {0:.2%} (+/- {1:.2%})".format(np.mean(scores), np.std(scores)\*2))

print("Execution Time: {0:.5} seconds \n".format(end-start))

target\_names = ['No Disease', 'Disease']

print(classification\_report(y\_test, prediction\_knn\_sc, target\_names=target\_names))

<https://www.kaggle.com/cherngs/heart-disease-cleveland-uci>

Bu makine öğrenmesi siteminde yukarıdaki adresteki veriyi kullandım. Kalp hastalıklarının sistemdeki

parametrelere nasıl bağlı olduğunu grafiklerle gösterdim.Doğruluk oranı %88.33 çıktı.

Rumeysa Çayır 0101190167 İstanbul Tıp Fakültesi rumeysacayir6@gmail.com