

AD SOYAD = BURAK HAYRAT

NUMARA = 21040101026

BÖLÜM = BİLGİSAYAR MÜHENDİSLİĞİ

KONU = DERİN ÖĞRENME FİNAL ÖDEVİ

**1. ) Naive Bayes Sınıflandırıcısı**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.naive\_bayes import GaussianNB

from sklearn.metrics import confusion\_matrix, classification\_report, roc\_curve,roc\_auc\_score

import matplotlib.pyplot as plt

veri\_seti = pd.read\_csv("veri\_seti.csv")

X = veri\_seti.drop("sinif\_degiskeni", axis=1)

y = veri\_seti["sinif\_degiskeni"]

X\_egitim, X\_test, y\_egitim, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

nb\_model = GaussianNB()

nb\_model.fit(X\_egitim, y\_egitim)

y\_pred = nb\_model.predict(X\_test)

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

print("Confusion Matrix:\n", conf\_matrix)

class\_report = classification\_report(y\_test, y\_pred)

print("Classification Report:\n", class\_report)

tn, fp, fn, tp = conf\_matrix.ravel()

accuracy = (tp + tn) / (tp + tn + fp + fn)

sensitivity = tp / (tp + fn)

specificity = tn / (tn + fp)

f1\_score = 2 \* (sensitivity \* specificity) / (sensitivity + specificity)

print("Accuracy:", accuracy)

print("Sensitivity:", sensitivity)

print("Specificity:", specificity)

print("F1 Score:", f1\_score)

y\_proba = nb\_model.predict\_proba(X\_test)[:, 1]

fpr, tpr, \_ = roc\_curve(y\_test, y\_proba)

auc = roc\_auc\_score(y\_test, y\_proba)

plt.figure()

plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve (area = %0.2f)' % auc)

plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.05])

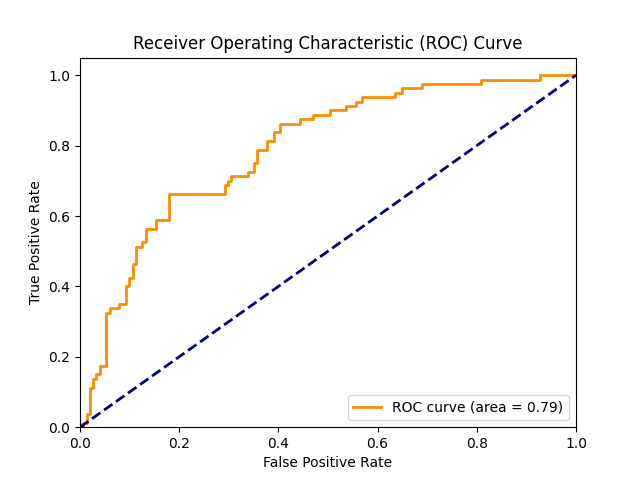
plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic (ROC) Curve')

plt.legend(loc="lower right")

plt.show()



Confusion Matrix:

[[119 32]

[ 27 53]]

Classification Report:

precision recall f1-score support

0 0.82 0.79 0.80 151

1 0.62 0.66 0.64 80

accuracy 0.74 231

macro avg 0.72 0.73 0.72 231

weighted avg 0.75 0.74 0.75 231

**2) K En Yakın Komşuluk Sınıflandırıcısı**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import classification\_report, confusion\_matrix, roc\_curve,roc\_auc\_score

import matplotlib.pyplot as plt

veri\_seti = pd.read\_csv("veri\_seti.csv")

# Özellikler (X) ve hedef değişken (y) olarak ayır

X = veri\_seti.drop('sinif\_degiskeni', axis=1)

y = veri\_seti['sinif\_degiskeni']

# Eğitim ve test setlerini oluştur

X\_egitim, X\_test, y\_egitim, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# K-en yakın komşuluk sınıflandırıcısını uygula

k\_degerleri = range(1, 21) # 1'den 20'ye kadar k değerlerini dene

dogruluklar = []

for k in k\_degerleri:

knn = KNeighborsClassifier(n\_neighbors=k)

knn.fit(X\_egitim, y\_egitim)

dogruluklar.append(knn.score(X\_test, y\_test))

en\_iyi\_k = k\_degerleri[dogruluklar.index(max(dogruluklar))]

print("En iyi k değeri:", en\_iyi\_k)

en\_iyi\_model = KNeighborsClassifier(n\_neighbors=en\_iyi\_k)

en\_iyi\_model.fit(X\_egitim, y\_egitim)

y\_tahmin = en\_iyi\_model.predict(X\_test)

print("Confusion Matrix:\n", confusion\_matrix(y\_test, y\_tahmin))

print("\nClassification Report:\n", classification\_report(y\_test, y\_tahmin))

y\_proba = en\_iyi\_model.predict\_proba(X\_test)[:, 1]

fpr, tpr, \_ = roc\_curve(y\_test, y\_proba)

auc = roc\_auc\_score(y\_test, y\_proba)

plt.figure()

plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve (area = %0.2f)' % auc)

plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.05])

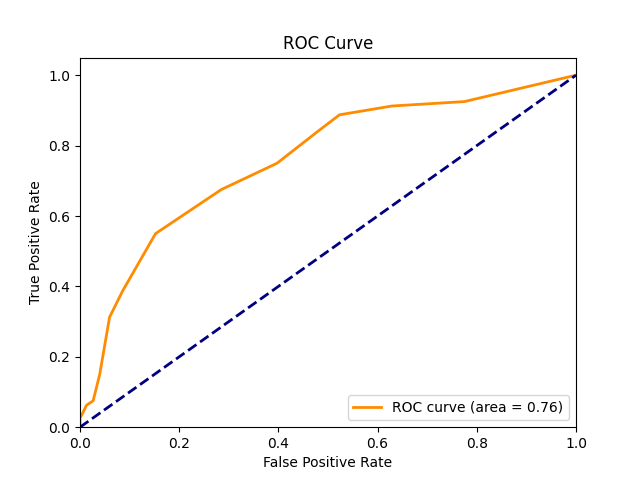
plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('ROC Curve')

plt.legend(loc="lower right")

plt.show()



**En iyi k değeri: 13**

Confusion Matrix:

[[128 23]

[ 36 44]]

Classification Report:

precision recall f1-score support

0 0.78 0.85 0.81 151

1 0.66 0.55 0.60 80

accuracy 0.74 231

macro avg 0.72 0.70 0.71 231

weighted avg 0.74 0.74 0.74 231

**3) MLP ve SVM Sınıflandırıcısı**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.neural\_network import MLPClassifier

from sklearn.svm import SVC

from sklearn.metrics import classification\_report, confusion\_matrix, roc\_curve, auc

import matplotlib.pyplot as plt

veri\_seti = pd.read\_csv('veri\_seti.csv')

# Girdi (X) ve Çıktı (y) değişkenlerini ayırma

X = veri\_seti[['kac\_hamile', 'glukoz\_test', 'kan\_basinci', 'deri\_kalinlik', 'insulin\_seviye', 'vucut\_kitle', 'aile\_diyabet', 'h\_yaş']]

y = veri\_seti['sinif\_degiskeni']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

print("Eğitim seti boyutu:", X\_train.shape)

print("Test seti boyutu:", X\_test.shape)

mlp = MLPClassifier(random\_state=42)

mlp.fit(X\_train, y\_train)

y\_pred\_mlp = mlp.predict(X\_test)

y\_prob\_mlp = mlp.predict\_proba(X\_test)[:, 1]

print("MLP Sınıflandırma Raporu:\n", classification\_report(y\_test, y\_pred\_mlp))

print("MLP Konfüzyon Matrisi:\n", confusion\_matrix(y\_test, y\_pred\_mlp))

fpr\_mlp, tpr\_mlp, \_ = roc\_curve(y\_test, y\_prob\_mlp)

roc\_auc\_mlp = auc(fpr\_mlp, tpr\_mlp)

svm = SVC(probability=True, random\_state=42)

svm.fit(X\_train, y\_train)

y\_pred\_svm = svm.predict(X\_test)

y\_prob\_svm = svm.predict\_proba(X\_test)[:, 1]

print("SVM Sınıflandırma Raporu:\n", classification\_report(y\_test, y\_pred\_svm))

print("SVM Konfüzyon Matrisi:\n", confusion\_matrix(y\_test, y\_pred\_svm))

fpr\_svm, tpr\_svm, \_ = roc\_curve(y\_test, y\_prob\_svm)

roc\_auc\_svm = auc(fpr\_svm, tpr\_svm)

plt.figure()

plt.plot(fpr\_mlp, tpr\_mlp, color='darkorange', lw=2, label='MLP ROC curve (area = %0.2f)' % roc\_auc\_mlp)

plt.plot(fpr\_svm, tpr\_svm, color='blue', lw=2, label='SVM ROC curve (area = %0.2f)' % roc\_auc\_svm)

plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.05])

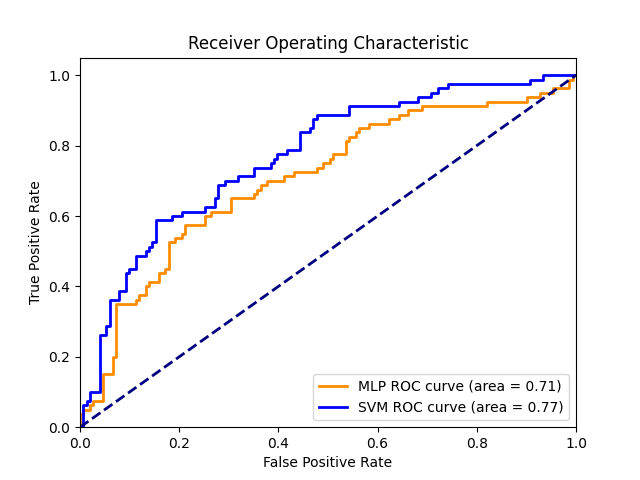
plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic')

plt.legend(loc="lower right")

plt.show()



**Eğitim seti boyutu: (537, 8)**

**Test seti boyutu: (231, 8)**

MLP Sınıflandırma Raporu:

precision recall f1-score support

0 0.76 0.81 0.78 151

1 0.59 0.53 0.56 80

accuracy 0.71 231

macro avg 0.68 0.67 0.67 231

weighted avg 0.70 0.71 0.71 231

MLP Konfüzyon Matrisi:

[[122 29]

[ 38 42]]

SVM Sınıflandırma Raporu:

precision recall f1-score support

0 0.76 0.87 0.81 151

1 0.66 0.49 0.56 80

accuracy 0.74 231

macro avg 0.71 0.68 0.69 231

weighted avg 0.73 0.74 0.72 231

SVM Konfüzyon Matrisi:

[[131 20]

[ 41 39]]

**Referanslar :**

<https://www.kaggle.com/code/shrutimechlearn/step-by-step-diabetes-classification-knn-detailed>

<https://www.kaggle.com/code/amolbhivarkar/knn-for-classification-using-scikit-learn>

<https://www.kaggle.com/code/zabihullah18/diabetes-prediction>