ENM421 – INTRODUCTION TO DATA SCIENCE

Summary Animation: https://www.powtoon.com/s/dPcNhMt49A9/1/m/s

HEART DISEASE DETECTION

Erhan ŞİMŞEKER

OVERVIEW

03

PROBLEM

04

EXPLORATORY DATA ANALYSIS

09

DATA CLEANING AND PREPROCESSING

10

MACHINE LEARNING

11

INTERPRETATION OF RESULTS

13

CLOSURE

PROBLEM

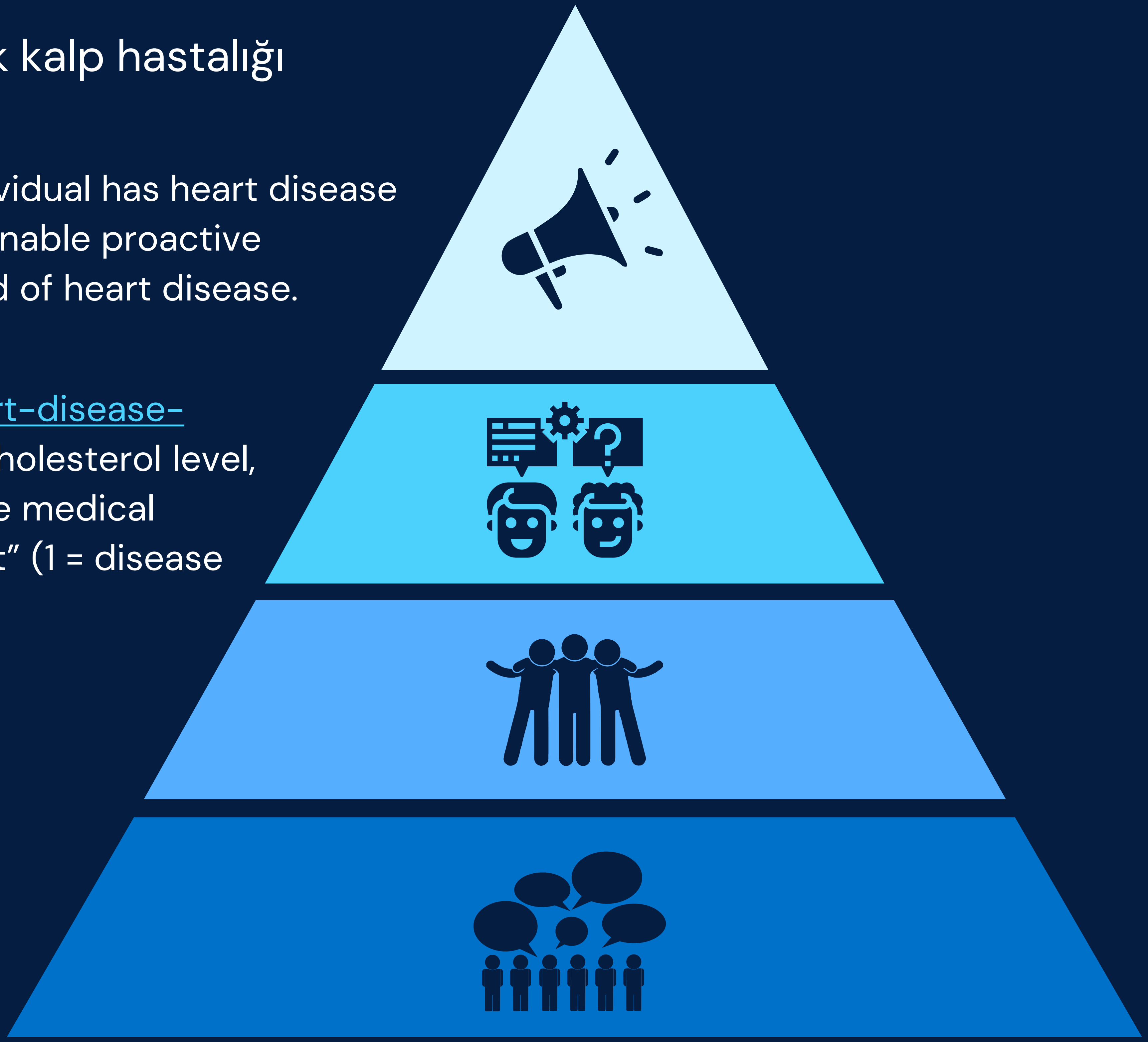
Bu projede, bireylerin tıbbi verilerinden yola çıkarak kalp hastalığı taşıyıp taşımadığını tahmin etmek hedeflenmiştir.

01

In this project, the goal is to predict whether an individual has heart disease based on their medical data. This problem aims to enable proactive healthcare interventions by estimating the likelihood of heart disease.

02

Using the heart.csv file—obtained from https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset are made based on features such as age, cholesterol level, and maximum heart rate. The dataset includes these medical attributes along with a target variable named “target” (1 = disease present, 0 = disease absent).



EXPLORATORY DATA ANALYSIS

General Information About the Data

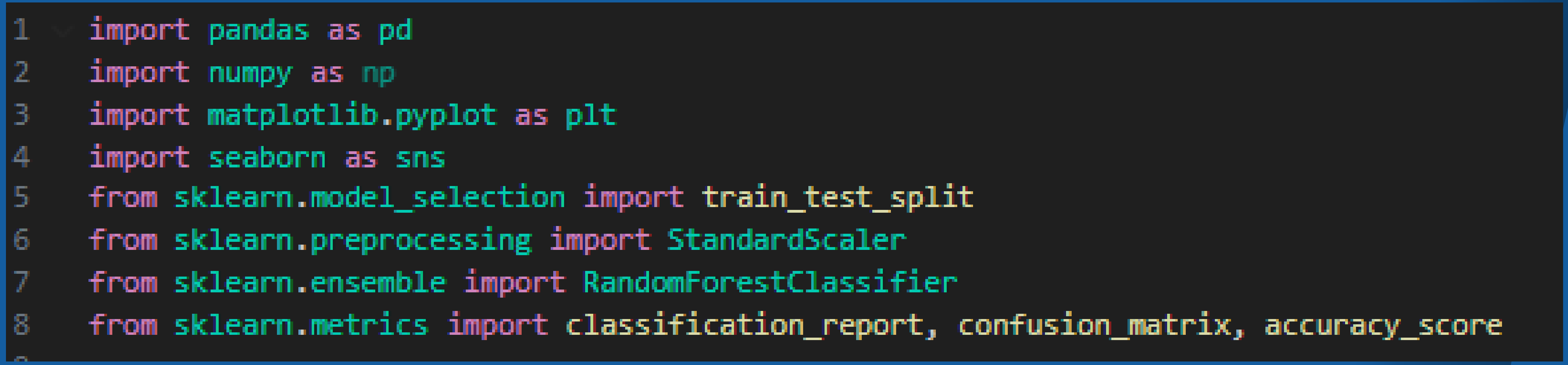
- The dataset was inspected with the info() function, and it was confirmed that there are no missing values.
- The basic statistics of numerical variables were analyzed using describe().

Target Variable Distribution

- The distribution of individuals with and without heart disease was visualized using sns.countplot. This analysis demonstrated whether the dataset is balanced.

EXPLORATORY DATA ANALYSIS

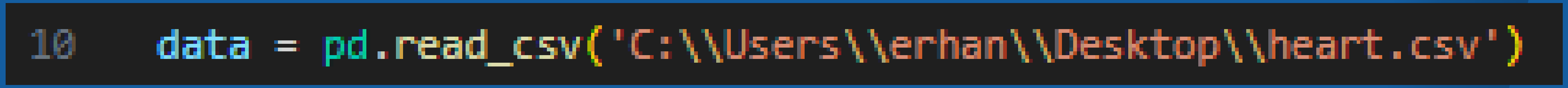
REQUIRED LIBRARIES



```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
```

- pandas: Data manipulation and analysis
- numpy: Numerical computations and array operations
- matplotlib.pyplot: Plotting
- seaborn: Advanced data visualization tools
- sklearn: Machine learning algorithms and utilities

DATA LOADING

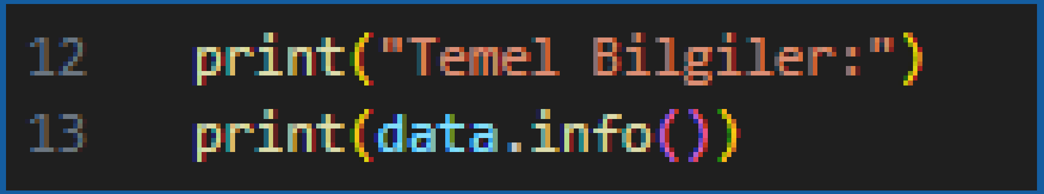


```python
data = pd.read_csv('C:\\Users\\erhan\\Desktop\\heart.csv')
```

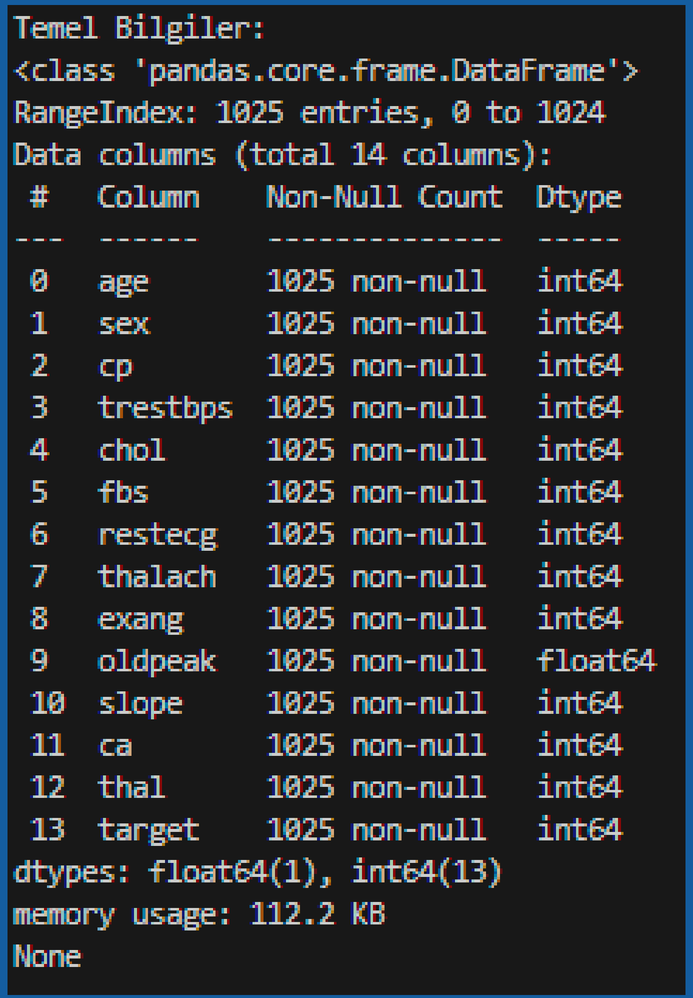
EXPLORATORY DATA ANALYSIS

Data Overview

- The dataset's column types, and any missing values were displayed.



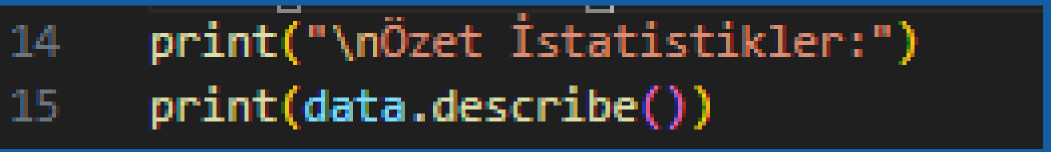
```
print("Temel Bilgiler:")
print(data.info())
```



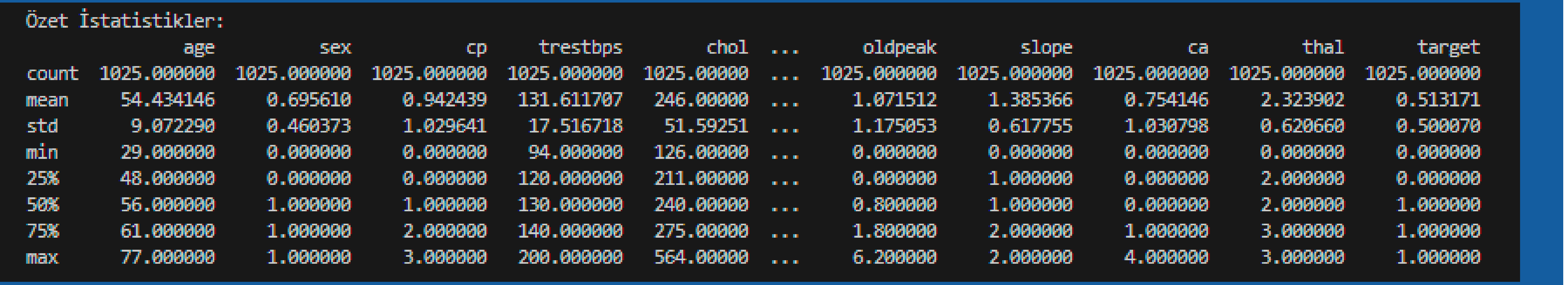
```
Temel Bilgiler:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1025 non-null   int64
 1   sex       1025 non-null   int64
 2   cp        1025 non-null   int64
 3   trestbps  1025 non-null   int64
 4   chol      1025 non-null   int64
 5   fbs       1025 non-null   int64
 6   restecg   1025 non-null   int64
 7   thalach   1025 non-null   int64
 8   exang     1025 non-null   int64
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64
 11  ca        1025 non-null   int64
 12  thal      1025 non-null   int64
 13  target    1025 non-null   int64
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
None
```

Statistical Summary

- Basic statistics (mean, standard deviation, etc.) were computed for the numerical columns.



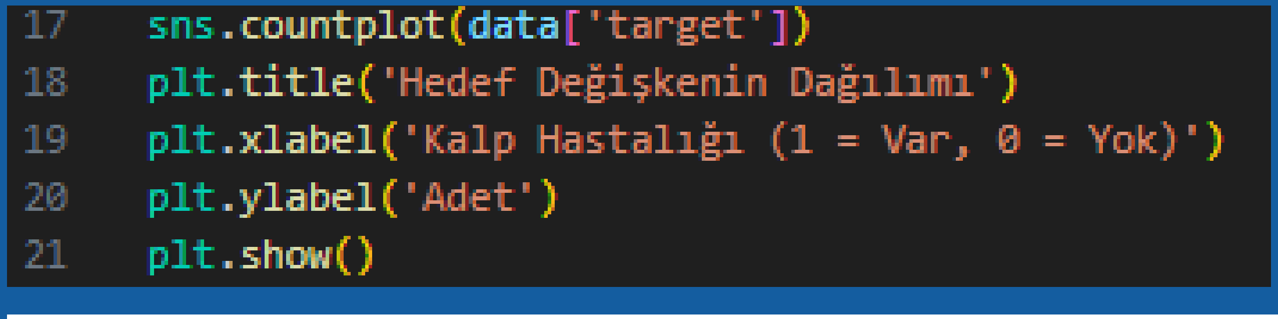
```
print("\nÖzet İstatistikler:")
print(data.describe())
```



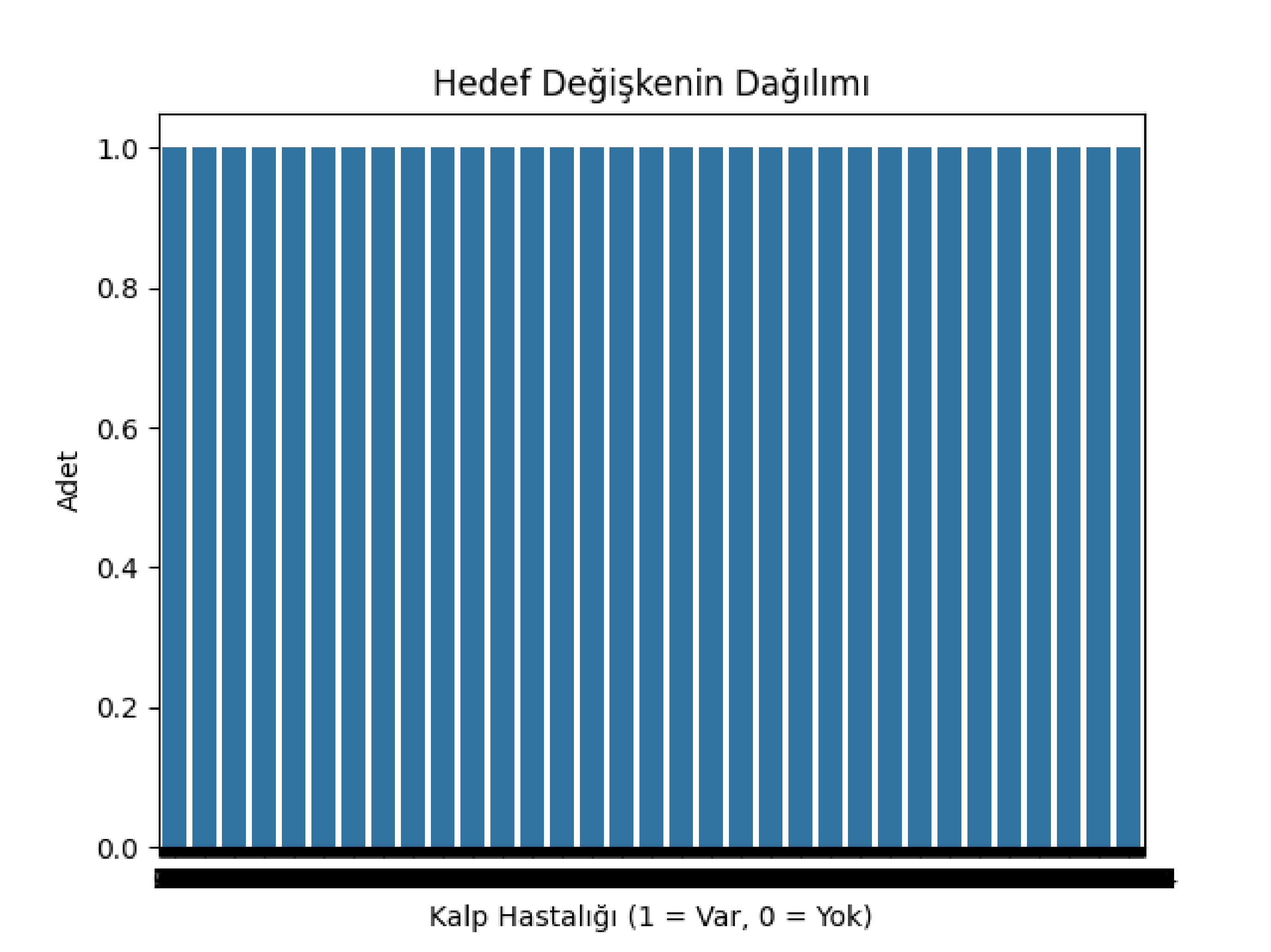
```
Özet İstatistikler:
               age          sex           cp     trestbps        chol  ...      oldpeak        slope           ca         thal       target
count  1025.000000  1025.000000  1025.000000  1025.000000  1025.00000  ...  1025.000000  1025.000000  1025.000000  1025.000000  1025.000000
mean     54.434146     0.695610     0.942439   131.611707   246.00000  ...     1.071512     1.385366     0.754146     2.323902     0.513171
std       9.072290     0.460373     1.029641    17.516718    51.59251  ...     1.175053     0.617755     1.030798     0.620660     0.500070
min      29.000000     0.000000     0.000000    94.000000   126.00000  ...     0.000000     0.000000     0.000000     0.000000     0.000000
25%      48.000000     0.000000     0.000000   120.000000   211.00000  ...     0.000000     1.000000     0.000000     2.000000     0.000000
50%      56.000000     1.000000     1.000000   130.000000   240.00000  ...     0.800000     1.000000     0.000000     2.000000     1.000000
75%      61.000000     1.000000     2.000000   140.000000   275.00000  ...     1.800000     2.000000     1.000000     3.000000     1.000000
max      77.000000     1.000000     3.000000   200.000000   564.00000  ...     6.200000     2.000000     4.000000     3.000000     1.000000
```

Target Variable Distribution

- A bar chart of the target variable's distribution (heart disease present/absent) was plotted using sns.countplot. This analysis showed whether the dataset is balanced.



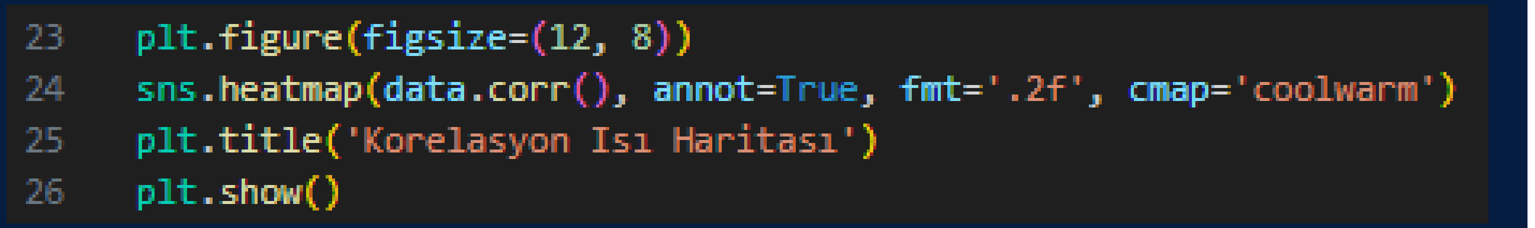
```
sns.countplot(data['target'])
plt.title('Hedef Değişkenin Dağılımı')
plt.xlabel('Kalp Hastalığı (1 = Var, 0 = Yok)')
plt.ylabel('Adet')
plt.show()
```



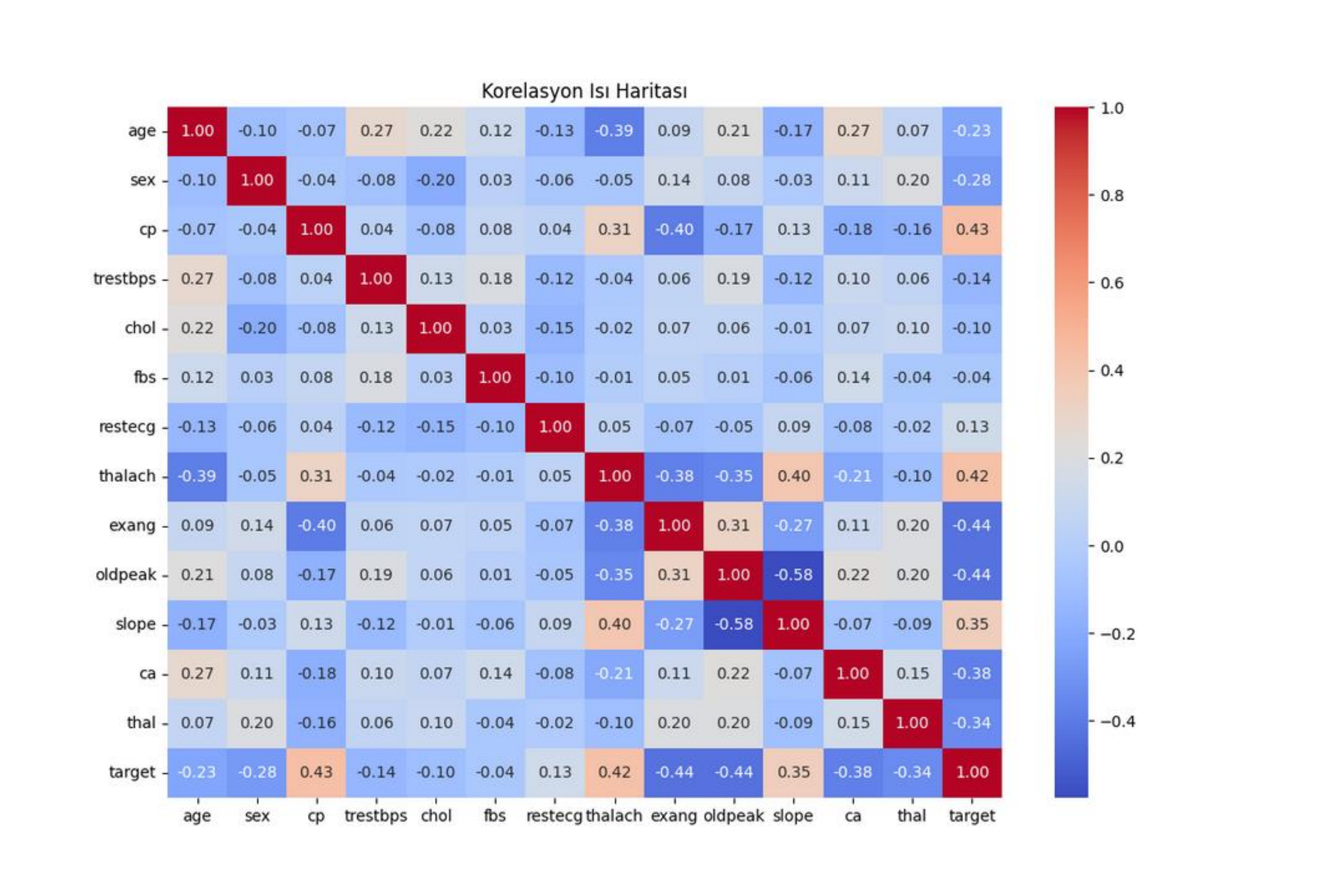
EXPLORATORY DATA ANALYSIS

Correlation Heatmap

- Correlations between features were analyzed with sns.heatmap. The heatmap visualized the relationships between the target variable (target) and the other features.



```python
plt.figure(figsize=(12, 8))
sns.heatmap(data.corr(), annot=True, fmt='.2f', cmap='coolwarm')
plt.title('Korelasyon Isı Haritası')
plt.show()
```



EXPLORATORY DATA ANALYSIS

Key Features Displayed on the Map

age

trestbps

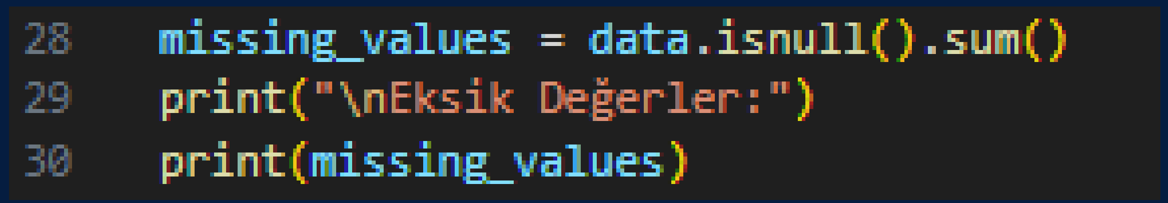
chol

thalach

oldpeak

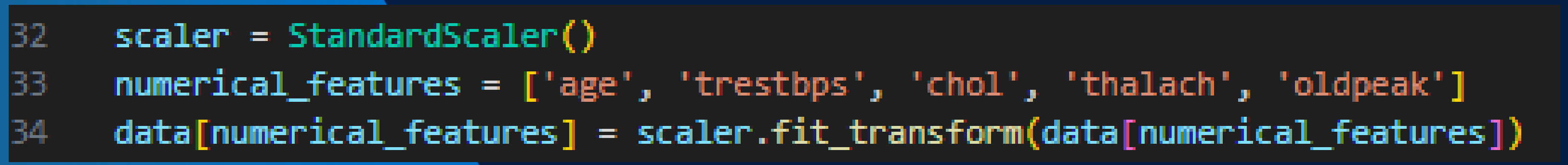
DATA CLEANING AND PREPROCESSING

- **Missing Data Analysis:** The number of missing values in each column was calculated.



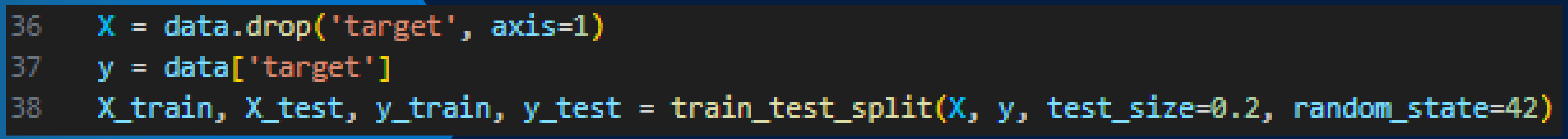
```
missing_values = data.isnull().sum()
print("\nEksik Değerler:")
print(missing_values)
```

- **Data Standardization:** Numerical features were scaled to have a mean of 0 and a standard deviation of 1.

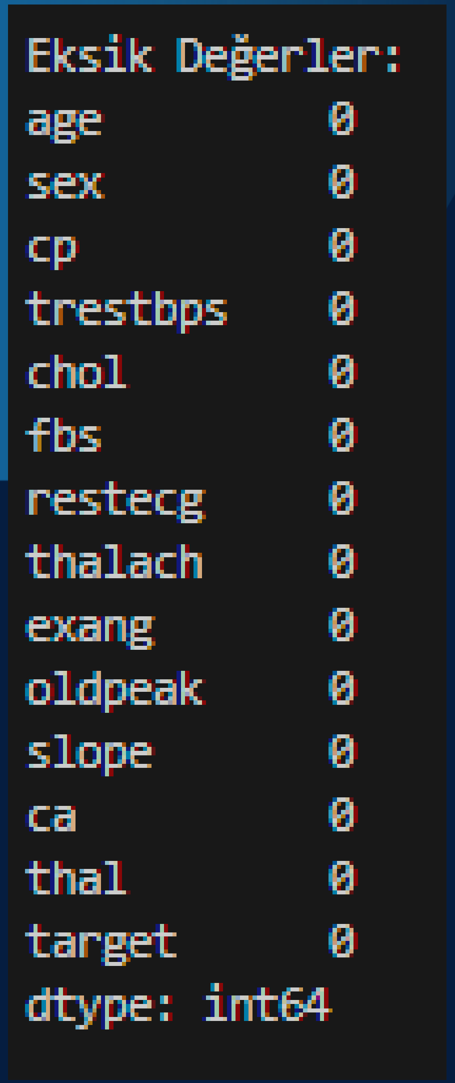


```
scaler = StandardScaler()
numerical_features = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']
data[numerical_features] = scaler.fit_transform(data[numerical_features])
```

- **Data Splitting:** The dataset was divided into training (80%) and testing (20%) sets.



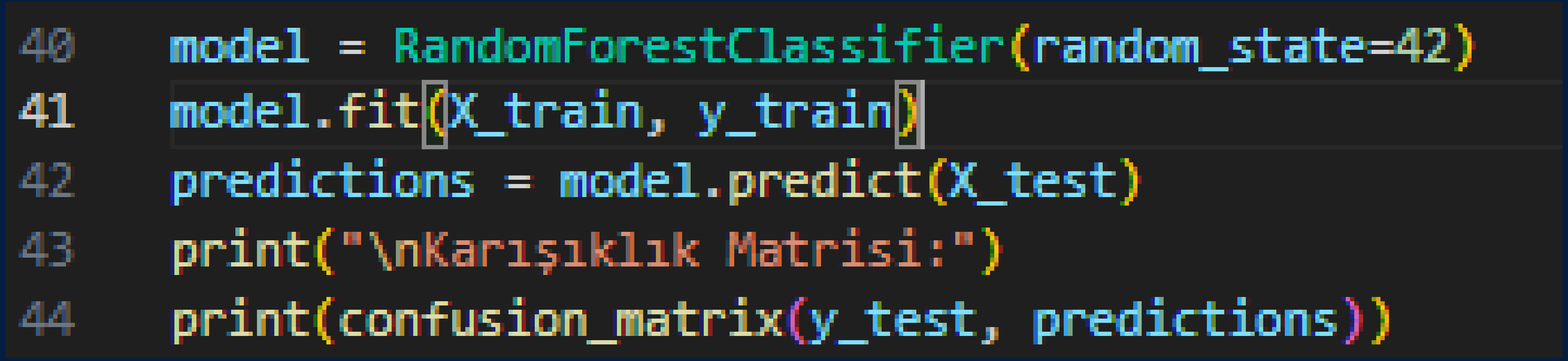
```
X = data.drop('target', axis=1)
y = data['target']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```



```
Eksik Değerler:
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

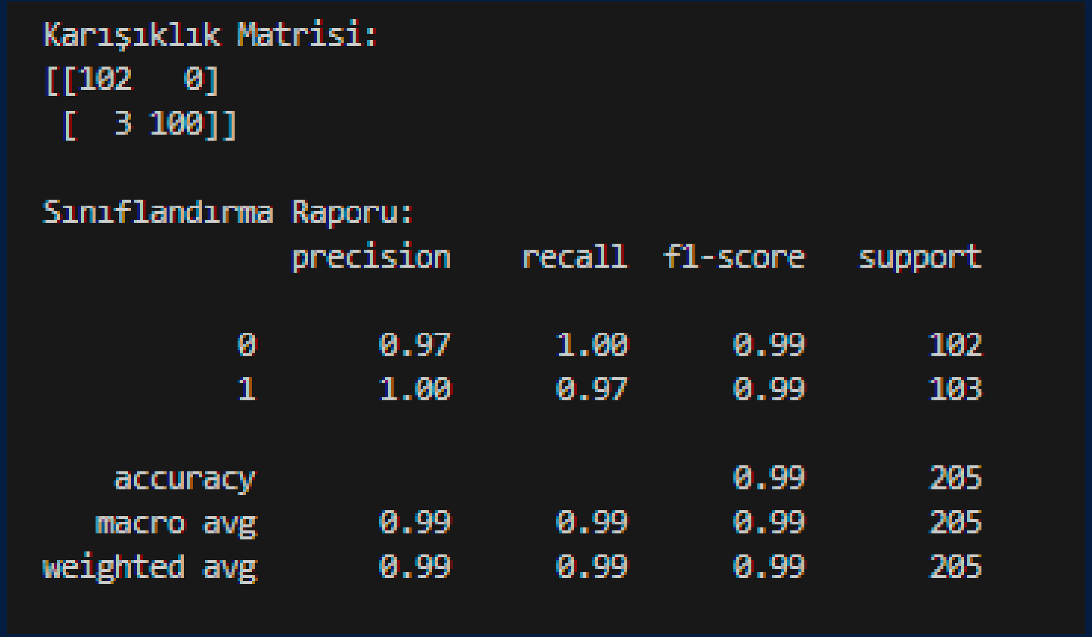
MACHINE LEARNING

- A Random Forest algorithm was selected for model building and training, and this classifier was trained on the data.
- The Random Forest algorithm combines the outputs of multiple decision trees to produce more accurate and generalizable predictions. Decision trees classify data using a specialized set of rules; by using many trees, this model increases accuracy and reduces the risk of overfitting.
- Default hyperparameters were used, with random_state=42 set to control randomness.



```python
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
predictions = model.predict(X_test)
print("\nKarışıklık Matrisi:")
print(confusion_matrix(y_test, predictions))
```

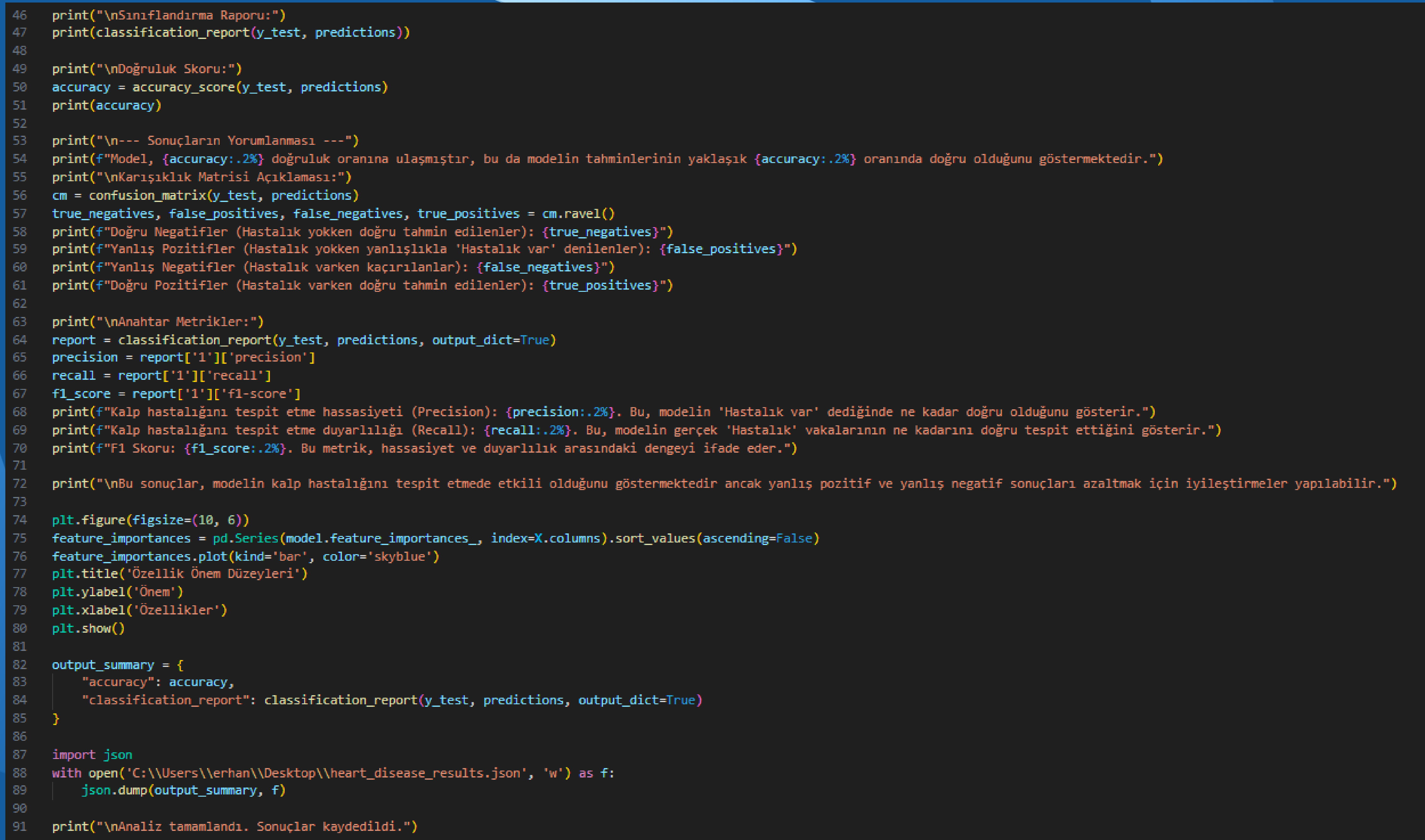
- A confusion matrix was generated to summarize the model’s prediction performance on the test dataset.



```
Karışıklık Matrisi:
[[102   0]
 [  3 100]]

Sınıflandırma Raporu:
              precision    recall  f1-score   support

           0       0.97      1.00      0.99       102
           1       1.00      0.97      0.99       103

    accuracy                           0.99       205
   macro avg       0.99      0.99      0.99       205
weighted avg       0.99      0.99      0.99       205
```

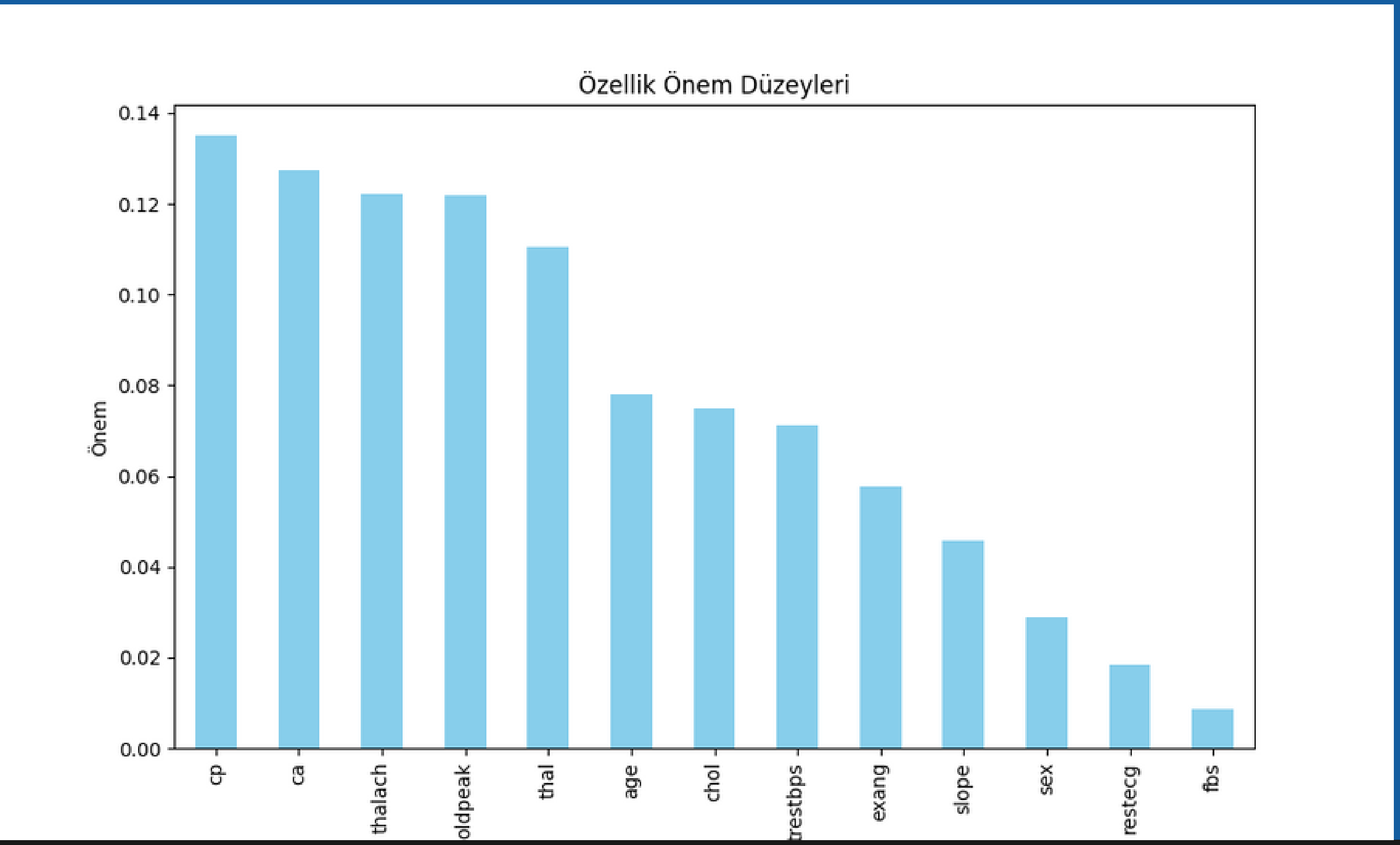
INTERPRETATION OF RESULTS

- Classification Report: Includes metrics such as precision, recall, and F1-score.
- Accuracy Score: Calculates the model’s overall accuracy.
- Feature Importance Visualization: Displays which features the model deemed most important using a bar chart.
- Saving Results: The accuracy score and classification report are saved to a JSON file.ChatGPT’ye sor

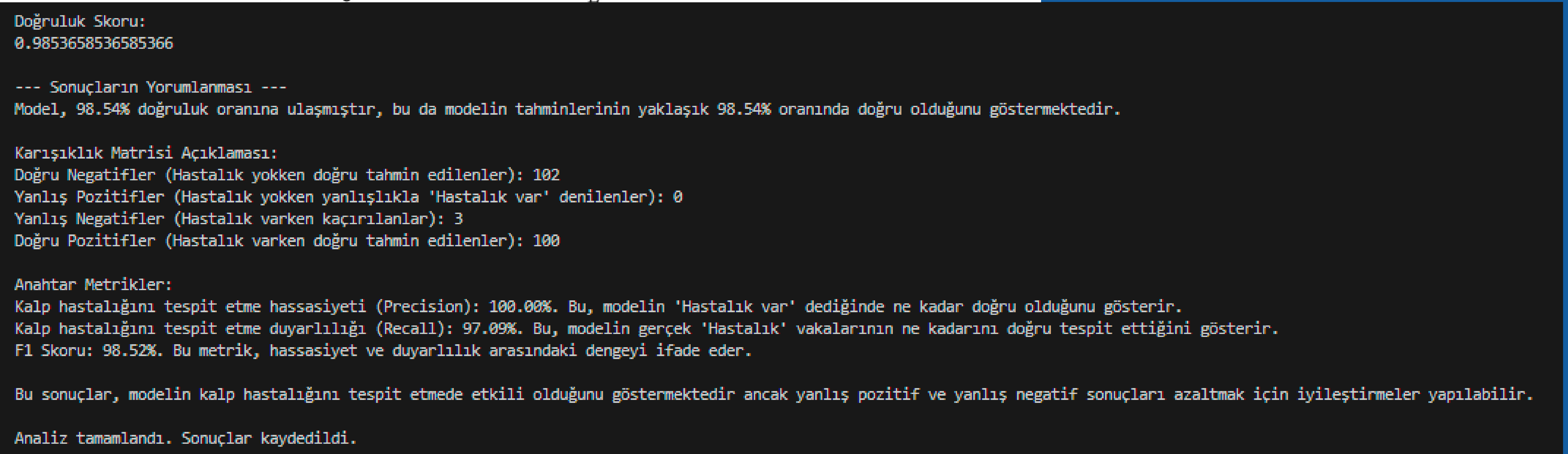


```python
print("\nSınıflandırma Raporu:")
print(classification_report(y_test, predictions))

print("\nDoğruluk Skoru:")
accuracy = accuracy_score(y_test, predictions)
print(accuracy)

print("\n--- Sonuçların Yorumlanması ---")
print(f"Model, {accuracy:.2%} doğruluk oranına ulaşmıştır, bu da modelin tahminlerinin yaklaşık {accuracy:.2%} oranında doğru olduğunu göstermektedir.")
print("\nKarışıklık Matrisi Açıklaması:")
cm = confusion_matrix(y_test, predictions)
true_negatives, false_positives, false_negatives, true_positives = cm.ravel()
print(f"Doğru Negatifler (Hastalık yokken doğru tahmin edilenler): {true_negatives}")
print(f"Yanlış Pozitifler (Hastalık yokken yanlışlıkla 'Hastalık var' denilenler): {false_positives}")
print(f"Yanlış Negatifler (Hastalık varken kaçırılanlar): {false_negatives}")
print(f"Doğru Pozitifler (Hastalık varken doğru tahmin edilenler): {true_positives}")

print("\nAnahtar Metrikler:")
report = classification_report(y_test, predictions, output_dict=True)
precision = report['1']['precision']
recall = report['1']['recall']
f1_score = report['1']['f1-score']
print(f"Kalp hastalığını tespit etme hassasiyeti (Precision): {precision:.2%}. Bu, modelin 'Hastalık var' dediğinde ne kadar doğru olduğunu gösterir.")
print(f"Kalp hastalığını tespit etme duyarlılığı (Recall): {recall:.2%}. Bu, modelin gerçek 'Hastalık' vakalarının ne kadarını doğru tespit ettiğini gösterir.")
print(f"F1 Skoru: {f1_score:.2%}. Bu metrik, hassasiyet ve duyarlılık arasındaki dengeyi ifade eder.")

print("\nBu sonuçlar, modelin kalp hastalığını tespit etmede etkili olduğunu göstermektedir ancak yanlış pozitif ve yanlış negatif sonuçları azaltmak için iyileştirmeler yapılabilir.")

plt.figure(figsize=(10, 6))
feature_importances = pd.Series(model.feature_importances_, index=X.columns).sort_values(ascending=False)
feature_importances.plot(kind='bar', color='skyblue')
plt.title('Özellik Önem Düzeyleri')
plt.ylabel('Önem')
plt.xlabel('Özellikler')
plt.show()

output_summary = {
    "accuracy": accuracy,
    "classification_report": classification_report(y_test, predictions, output_dict=True)
}

import json
with open('C:\\Users\\erhan\\Desktop\\heart_disease_results.json', 'w') as f:
    json.dump(output_summary, f)

print("\nAnaliz tamamlandı. Sonuçlar kaydedildi.")
```

INTERPRETATION OF RESULTS



- Contents of the `heart_disease_results.json` file containing the saved results:
- {"accuracy": 0.9853658536585366, "classification_report": {"0": {"precision": 0.9714285714285714, "recall": 1.0, "f1-score": 0.9855072463768116, "support": 102.0}, "1": {"precision": 1.0, "recall": 0.970873786407767, "f1-score": 0.9852216748768473, "support": 103.0}, "accuracy": 0.9853658536585366, "macro avg": {"precision": 0.9857142857142858, "recall": 0.9854368932038835, "f1-score": 0.9853644606268295, "support": 205.0}, "weighted avg": {"precision": 0.9857839721254356, "recall": 0.9853658536585366, "f1-score": 0.9853637641109759, "support": 205.0}}}



```
Doğruluk Skoru:
0.9853658536585366

--- Sonuçların Yorumlanması ---
Model, 98.54% doğruluk oranına ulaşmıştır, bu da modelin tahminlerinin yaklaşık 98.54% oranında doğru olduğunu göstermektedir.

Karışıklık Matrisi Açıklaması:
Doğru Negatifler (Hastalık yokken doğru tahmin edilenler): 102
Yanlış Pozitifler (Hastalık yokken yanlışlıkla 'Hastalık var' denilenler): 0
Yanlış Negatifler (Hastalık varken kaçırılanlar): 3
Doğru Pozitifler (Hastalık varken doğru tahmin edilenler): 100

Anahtar Metrikler:
Kalp hastalığını tespit etme hassasiyeti (Precision): 100.00%. Bu, modelin 'Hastalık var' dediğinde ne kadar doğru olduğunu gösterir.
Kalp hastalığını tespit etme duyarlılığı (Recall): 97.09%. Bu, modelin gerçek 'Hastalık' vakalarının ne kadarını doğru tespit ettiğini gösterir.
F1 Skoru: 98.52%. Bu metrik, hassasiyet ve duyarlılık arasındaki dengeyi ifade eder.

Bu sonuçlar, modelin kalp hastalığını tespit etmede etkili olduğunu göstermektedir ancak yanlış pozitif ve yanlış negatif sonuçları azaltmak için iyileştirmeler yapılabilir.

Analiz tamamlandı. Sonuçlar kaydedildi.
```

Thank you :)