



ENM421 – VERİ BİLİMİNE GİRİŞ

Özet Animasyon: <https://www.powtoon.com/s/dPcNhMt49A9/1/m/s>

# KALP HASTALIĞI TESPİTİ

Erhan ŞİMŞEKER



# GENEL BAKIŞ

03

---

PROBLEM

04

---

KEŞİFSEL VERİ ANALİZİ

09

---

VERİ TEMİZLEME VE ÖN  
İŞLEME

10

---

MAKİNE ÖĞRENMESİ

11

---

SONUÇLARIN  
YORUMLANMASI

13

---

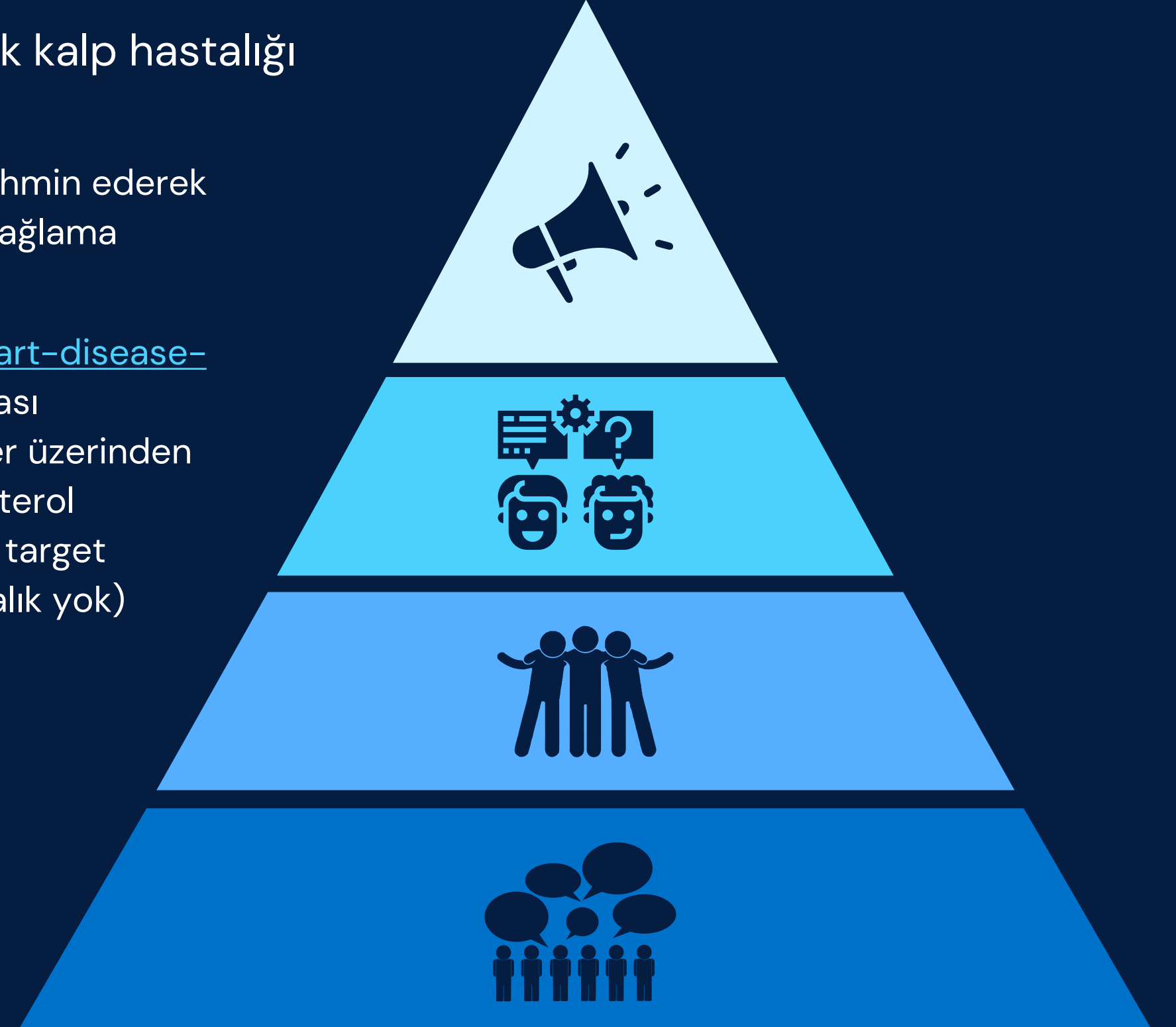
KAPANIŞ

# PROBLEM

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

**01** Bu problem, kalp hastalığına sahip olma olasılığını tahmin ederek sağlık hizmetleri için proaktif müdahalelere olanak sağlama amacı taşımaktadır.

**02** <https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset> web sitesinden elde edilen heart.csv dosyası aracılığıyla kalp hastalığı tahmini için kullanılan veriler üzerinden tahminler yapılmıştır. Bu dosya; bireylerin yaş, kolesterol seviyesi, maksimum kalp atışı gibi tıbbi özellikleri ile target adında bir hedef değişken (1 = hastalık var, 0 = hastalık yok) içeriyor olarak tanımlanmıştır.





# KEŞİFSEL VERİ ANALİZİ

## Veri Hakkında Genel Bilgi

- Veri seti info() fonksiyonu ile incelenmiştir ve herhangi bir eksik veri bulunmadığı doğrulanmıştır.
- describe() ile sayısal değişkenlerin temel istatistikleri analiz edilmiştir.

---

## Hedef Değişkenin Dağılımı

- sns.countplot ile kalp hastalığı olan ve olmayan bireylerin dağılımı görselleştirilmiştir. Bu analiz, veri setinin dengeli olup olmadığını göstermiştir.
-

# KEŞİFSEL VERİ ANALİZİ

## GEREKLİ KÜTÜPHANELER

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import seaborn as sns
5 from sklearn.model_selection import train_test_split
6 from sklearn.preprocessing import StandardScaler
7 from sklearn.ensemble import RandomForestClassifier
8 from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
```

- pandas: Veri manipülasyonu ve analizi
- numpy: Sayısal hesaplamalar ve dizi işlemleri
- matplotlib.pyplot: Grafik çizme
- seaborn: Gelişmiş veri görselleştirme araçları
- sklearn: Makine öğrenmesi algoritmaları ve araçları

---

## VERİ YÜKLEME

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



# KEŞİFSEL VERİ ANALİZİ

## Veri Bilgisi

- Veri setinin sütun tipleri ve eksik değerleri gösterilmiştir.

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

## İstatistiksel Özet

- Verideki sayısal sütunlar için temel istatistikler (ortalama, standart sapma vb.) çıkarılmıştır.

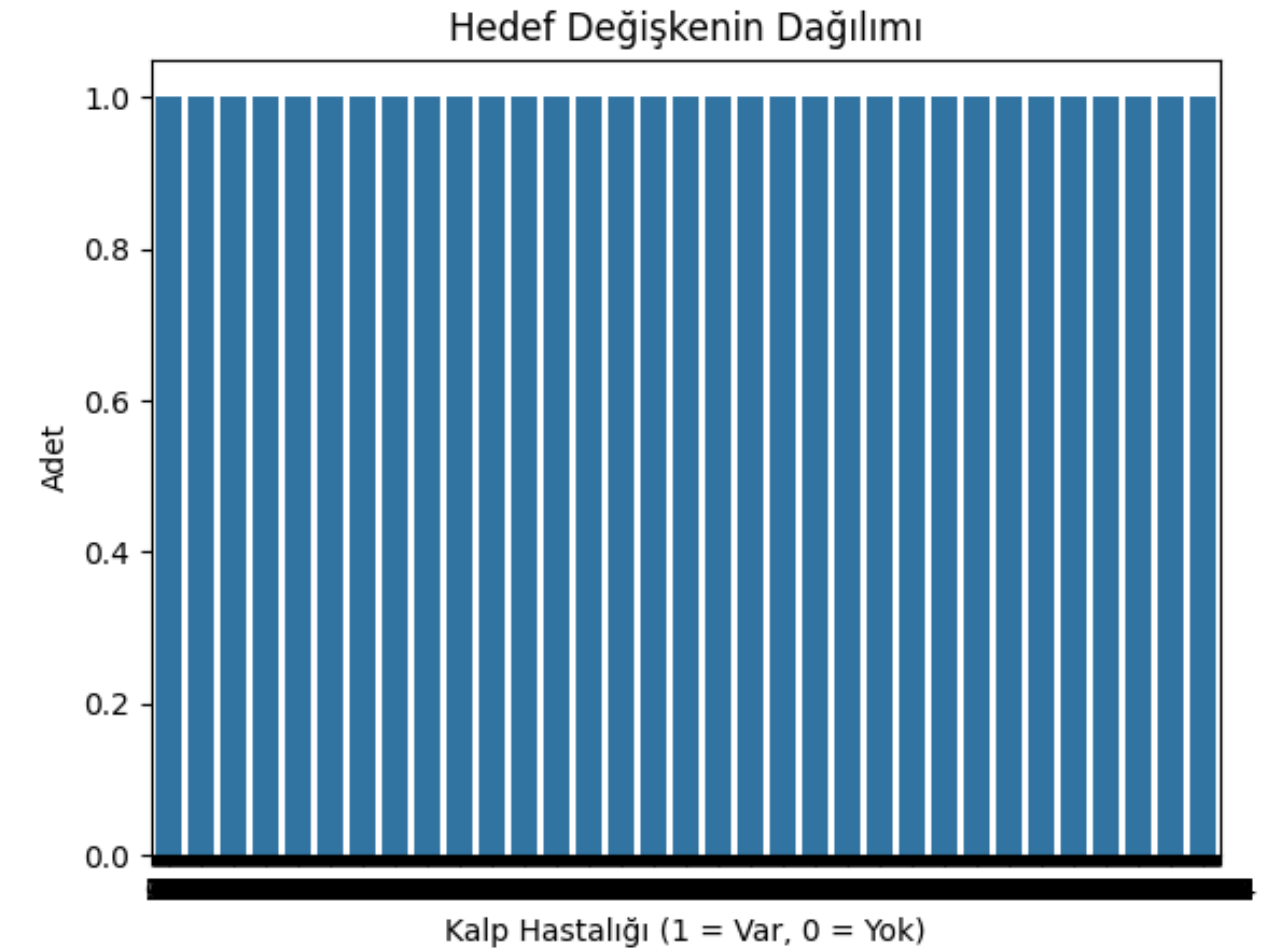
```
14 print("\nÖzet İstatistikler:")
15 print(data.describe())
```

```
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
```

## Hedef Değişkenin Dağılımı

- sns.countplot ile target değişkeninin (kalp hastalığı var/yok) dağılımı çubuk grafiği ile çizilmiştir. Bu analiz, veri setinin dengeli olup olmadığını göstermiştir.

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



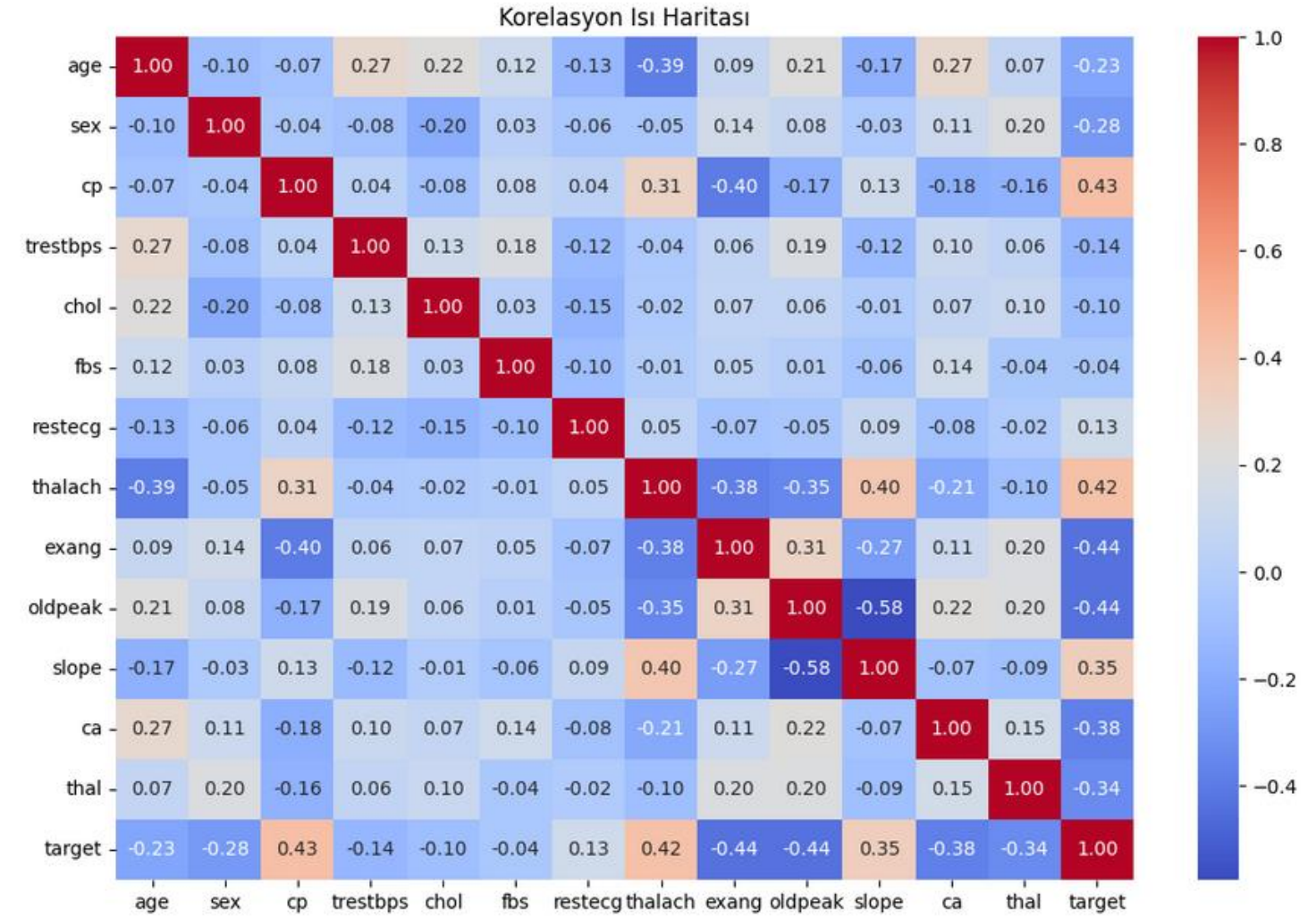
Özet İstatistikler:											
	age	sex	cp	trestbps	chol	...	oldpeak	slope	ca	thal	target
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	...	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000
mean	54.434146	0.695610	0.942439	131.611707	246.000000	...	1.071512	1.385366	0.754146	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	126.000000	...	0.000000	0.000000	0.000000	0.000000	0.000000
25%	48.000000	0.000000	0.000000	120.000000	211.000000	...	0.000000	1.000000	0.000000	2.000000	0.000000
50%	56.000000	1.000000	1.000000	130.000000	240.000000	...	0.800000	1.000000	0.000000	2.000000	1.000000
75%	61.000000	1.000000	2.000000	140.000000	275.000000	...	1.800000	2.000000	1.000000	3.000000	1.000000
max	77.000000	1.000000	3.000000	200.000000	564.000000	...	6.200000	2.000000	4.000000	3.000000	1.000000

# KEŞİFSEL VERİ ANALİZİ

## Korelasyon Isı Haritası

- sns.heatmap ile özellikler arasındaki korelasyonlar analiz edilmiştir. Korelasyon ısı haritası, hedef değişken olan target ile diğer özellikler arasındaki ilişkiyi görselleştirmiştir.

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



# KEŞİFSEL VERİ ANALİZİ

## Haritada Yer Alan Önemli Özellikler



age

Bireyin yaşı

trestbps

Dinlenme sırasında ölçülen kan basıncı

chol

Serum kolesterol seviyesi

thalach

Egzersiz sırasında ulaşılan maksimum kalp atışı

oldpeak

Egzersiz sonrası ST segment depresyonu



# VERİ TEMİZLEME VE ÖN İŞLEME

- **Eksik Veri Analizi:** Verideki eksik değerlerin sayısı her sütun için hesaplanmıştır.

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

- **Veri standardizasyonu:** Sayısal özellikler ölçeklenmiş, böylece ortalama 0 ve standart sapma 1 olmuştur.

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

- **Veri bölme:** Veri, eğitim (%80) ve test (%20) olarak ikiye ayrılmıştır.

```
36 X = data.drop('target', axis=1)
37 y = data['target']
38 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
```

# MAKİNE ÖĞRENMESİ

- Model oluşturma ve eğitme aşamasında makine öğrenmesi algoritması olarak Rastgele Orman (Random Forest) algoritması seçilmiş ve bu sınıflandırıcı eğitilmiştir.
- Rastgele Orman algoritması, birden fazla karar ağacının çıktısını birleştirerek daha doğru ve genellenebilir tahminler yapar. Karar ağaçları, özelleşmiş kurallar seti ile veri sınıflandırır. Bu model, doğruluk oranını artırmak ve overfitting riskini azaltmak için birçok ağaç kullanır.
- Varsayılan hiperparametreler kullanılmıştır ve rastgelelik kontrolü için random\_state=42 ayarlanmıştır.

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

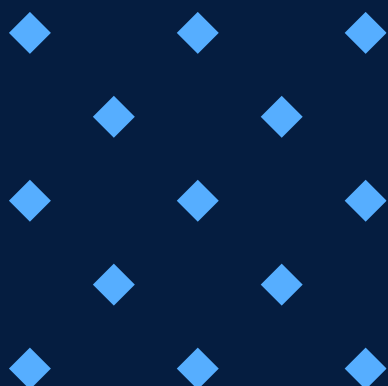
- Karışıklık matrisi ile de modelin test verisi üzerindeki tahmin performansını özetlenmiştir.

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



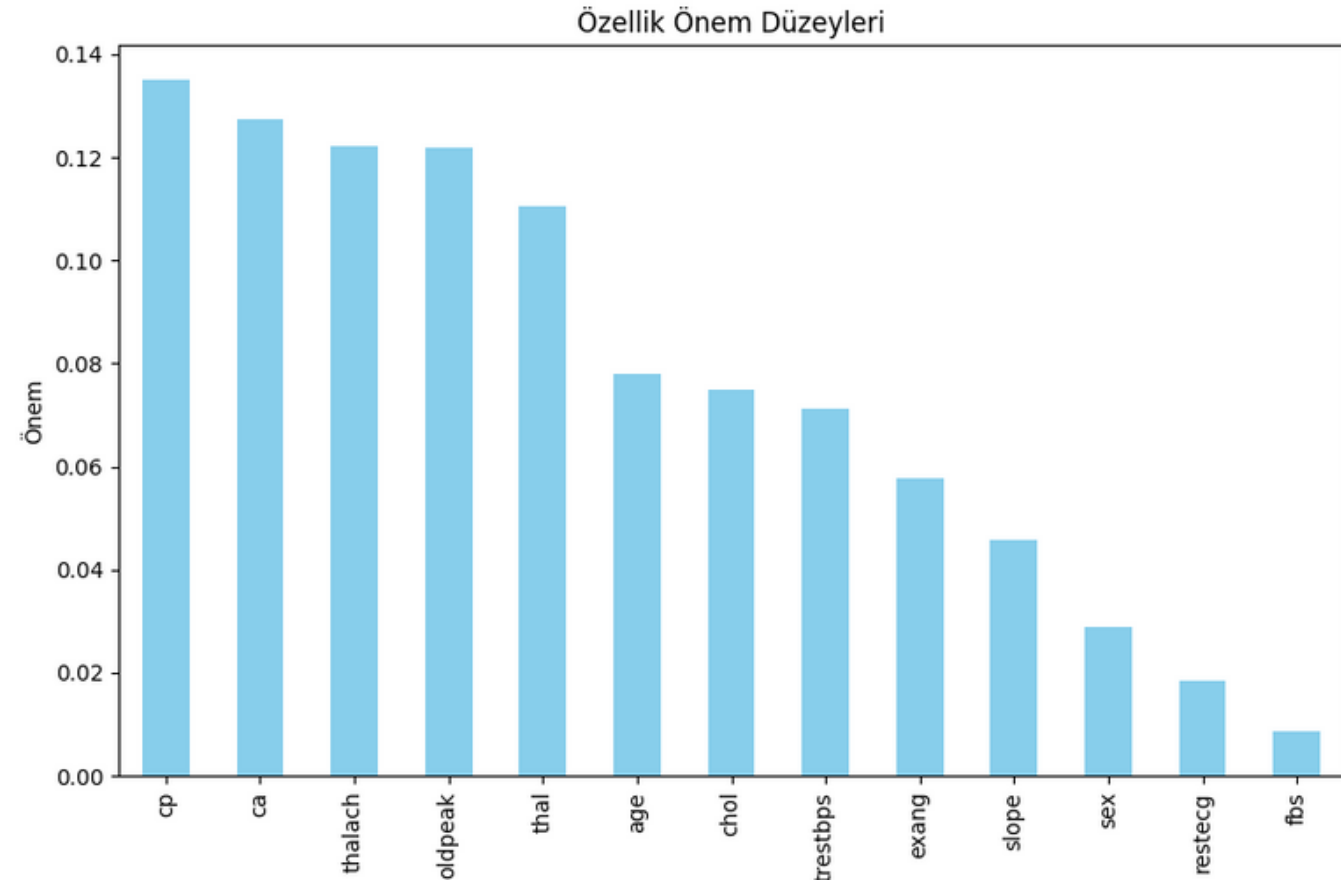
# SONUÇLARIN YORUMLANMASI

- Sınıflandırma raporu: Hassasiyet, doğruluk ve F1 skoru gibi metrikleri içerir.
- Doğruluk skoru: Modelin genel doğruluk oranını hesaplar.
- Özellik önemi görselleştirme: Modelin hangi özelliklere daha fazla önem verdiğini çubuk grafiğiyle gösterir.
- Sonuçların kaydedilmesi: Doğruluk oranı ve sınıflandırma raporu JSON formatında bir dosyaya kaydedilir.

```
46 print("\nSınıflandırma Raporu:")
47 print(classification_report(y_test, predictions))
48
49 print("\nDoğruluk Skoru:")
50 accuracy = accuracy_score(y_test, predictions)
51 print(accuracy)
52
53 print("\n--- Sonuçların Yorumlanması ---")
54 print(f"Model, {accuracy:.2%} doğruluk oranına ulaşmıştır, bu da modelin tahminlerinin yaklaşık {accuracy:.2%} oranında doğru olduğunu göstermektedir.")
55 print("\nKarışıklık Matrisi Açıklaması:")
56 cm = confusion_matrix(y_test, predictions)
57 true_negatives, false_positives, false_negatives, true_positives = cm.ravel()
58 print(f"Doğru Negatifler (Hastalık yokken doğru tahmin edilenler): {true_negatives}")
59 print(f"Yanlış Pozitifler (Hastalık yokken yanlışlıkla 'Hastalık var' denilenler): {false_positives}")
60 print(f"Yanlış Negatifler (Hastalık varken kaçırılanlar): {false_negatives}")
61 print(f"Doğru Pozitifler (Hastalık varken doğru tahmin edilenler): {true_positives}")
62
63 print("\nAnahtar Metrikler:")
64 report = classification_report(y_test, predictions, output_dict=True)
65 precision = report['1']['precision']
66 recall = report['1']['recall']
67 f1_score = report['1']['f1-score']
68 print(f"Kalp hastalığını tespit etme hassasiyeti (Precision): {precision:.2%}. Bu, modelin 'Hastalık var' dediğinde ne kadar doğru olduğunu gösterir.")
69 print(f"Kalp hastalığını tespit etme duyarlılığı (Recall): {recall:.2%}. Bu, modelin gerçek 'Hastalık' vakalarının ne kadarını doğru tespit ettiğini gösterir.")
70 print(f"F1 Skoru: {f1_score:.2%}. Bu metrik, hassasiyet ve duyarlılık arasındaki dengeyi ifade eder.")
71
72 print("\nBu sonuçlar, modelin kalp hastalığını tespit etmede etkili olduğunu göstermektedir ancak yanlış pozitif ve yanlış negatif sonuçları azaltmak için iyileştirmeler yapılabilir.")
73
74 plt.figure(figsize=(10, 6))
75 feature_importances = pd.Series(model.feature_importances_, index=X.columns).sort_values(ascending=False)
76 feature_importances.plot(kind='bar', color='skyblue')
77 plt.title('Özellik Önem Düzeyleri')
78 plt.ylabel('Önem')
79 plt.xlabel('Özellikler')
80 plt.show()
81
82 output_summary = {
83     "accuracy": accuracy,
84     "classification_report": classification_report(y_test, predictions, output_dict=True)
85 }
86
87 import json
88 with open('C:\\Users\\erhan\\Desktop\\heart_disease_results.json', 'w') as f:
89     json.dump(output_summary, f)
90
91 print("\nAnaliz tamamlandı. Sonuçlar kaydedildi.")
```



# SONUÇLARIN YORUMLANMASI



- Kaydedilen sonuçları içeren heart\_disease\_results.json dosyasındakiler:
- {"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.



**Teşekkürler :)**