

# Conda kurulumu

<https://www.anaconda.com/docs/getting-started/miniconda/install#windows-installation>

Doğru dizinde olduğunuzdan emin olun

```
cd Users\User  
  
curl https://repo.anaconda.com/miniconda/Miniconda3-latest-Windows-x86_64.exe  
--output .\Downloads\Miniconda3-latest-Windows-x86_64.exe
```

Yükleme tamamlanınca %100 olur.

```
C:\Users\User>curl https://repo.anaconda.com/miniconda/Miniconda3-latest-Windows-x86_64.exe --output .\Downloads\Miniconda3-latest-Windows-x86_64.exe  
% Total    % Received % Xferd  Average Speed   Time     Time      Time  Current  
          Dload  Upload   Total   Spent    Left  Speed  
90 88.8M  90 80.1M    0       0  3895k      0  0:00:23  0:00:21  0:00:02 4011k
```

Yüklenen exe'nin çalıştırılması.

```
.\Downloads\Miniconda3-latest-Windows-x86_64.exe
```

**License Agreement**

Please review the license terms before installing Miniconda3 py313\_25.9.1-3 (64-bit).

Press Page Down to see the rest of the agreement.

OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE , AND NON-INFRINGEMENT ARE DISCLAIMED. IN NO EVENT SHALL ANACONDA BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF MINICONDA(R), EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Miniconda3 py313\_25.9.1-3 (64-bit).

Anaconda, Inc. —

< Back

I Agree

Cancel

**Select Installation Type**

Please select the type of installation you would like to perform for Miniconda3 py313\_25.9.1-3 (64-bit).

Install for:

- Just Me (recommended)
- All Users (requires admin privileges)

Anaconda, Inc. —

< Back

Next >

Cancel

**Choose Install Location**

Choose the folder in which to install Miniconda3 py313\_25.9.1-3 (64-bit).

Setup will install Miniconda3 py313\_25.9.1-3 (64-bit) in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

**Destination Folder**

Space required: 475.5 MB

Space available: 13.1 GB

Anaconda, Inc. —

**Advanced Installation Options**

Customize how Miniconda3 integrates with Windows

 Create shortcuts (supported packages only). Add installation to my PATH environment variable

NOT recommended. This can lead to conflicts with other applications. Instead, use the Command Prompt and Powershell menus added to the Windows Start Menu.

 Register Miniconda3 as my default Python 3.13

Allows other programs, such as VSCode, PyCharm, etc. to automatically detect Miniconda3 as the primary Python 3.13 on the system.

 Clear the package cache upon completion

Recommended. Recovers some disk space without harming functionality.

Anaconda, Inc.

&lt; Back

Install

Cancel

İndirme işleminin bitmesinin ardından cmd yi aç kapa ve conda kurulmuş mu diye bak.

C:\Users\User&gt;conda --version

conda 25.9.1

Harika conda kurulumu gerçekleşti.

## Environment kurulumu

Aşağıdaki komutu **aynen** çalıştır:

```
conda config --add channels https://repo.anaconda.com/pkgs/main
```

Bu, **resmî Anaconda deposunun** kullanım şartlarını kabul eder.

Enveriment kuralım cmd ye yazalım

```
conda create -n env-01 python=3.10 numpy scipy
```

```
C:\Users\User>
C:\Users\User>conda create -n yapayzeka_env python=3.10 numpy scipy
Do you accept the Terms of Service (ToS) for https://repo.anaconda.com/pkgs/r? [(a)ccept/(r)eject/(v)iew]: a
Do you accept the Terms of Service (ToS) for https://repo.anaconda.com/pkgs/msys2? [(a)ccept/(r)eject/(v)iew]: a
3 channel Terms of Service accepted
Retrieving notices: done
Channels:
- defaults
Platform: win-64
Collecting package metadata (repodata.json): -
```

a ya basıp kabul et çıkan bildirimleri a enter.

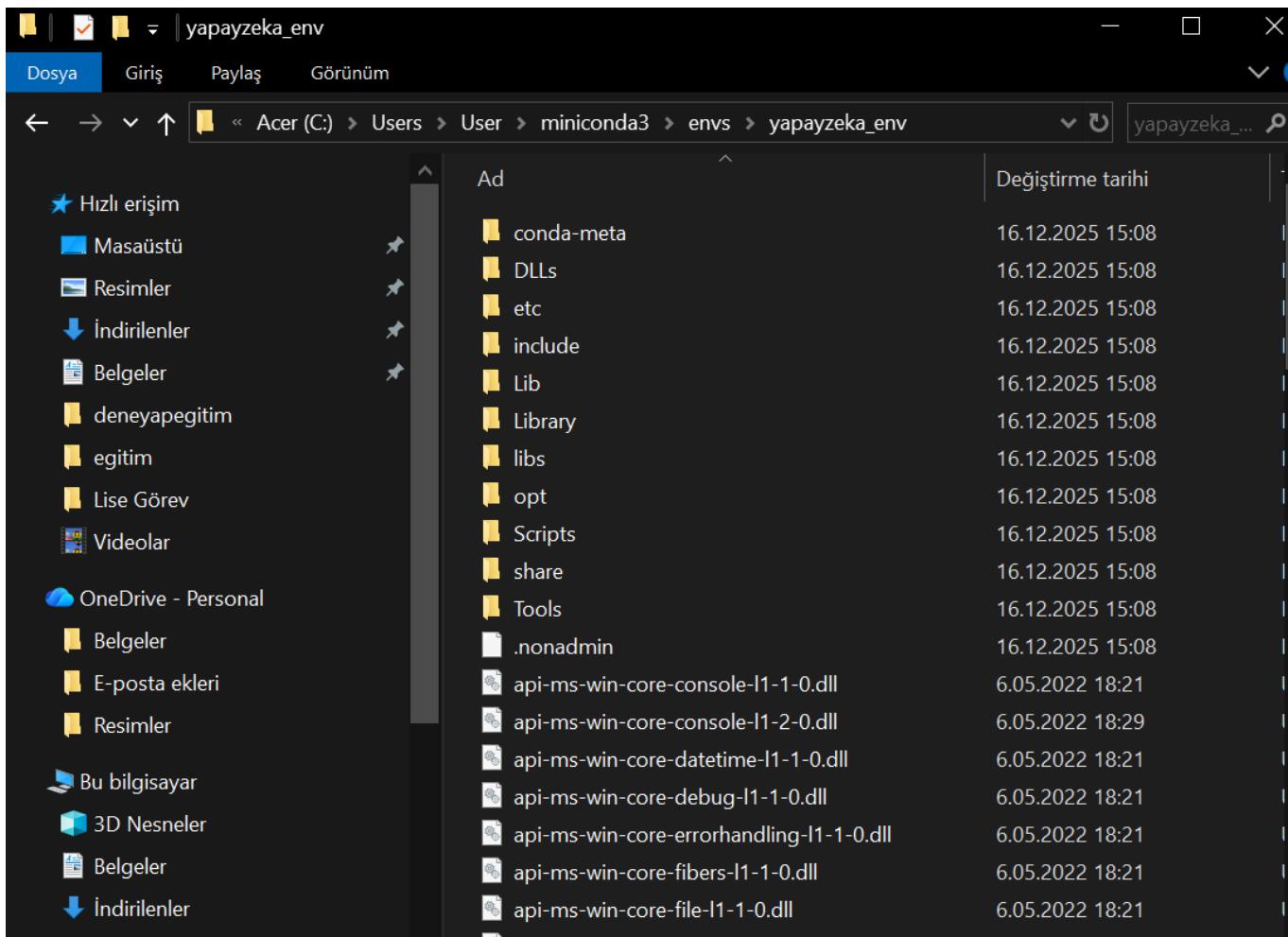
ilelemede y yaz ve entere bas.

```
Proceed ([y]/n)? y

Downloading and Extracting Packages:
mkl-2025.0.0      | 104.7 MB | #####| 71%   100%
scipy-1.15.3       | 26.0 MB  | #####| 100%  100%
python-3.10.19     | 15.3 MB  | #####| 100%  100%
numpy-base-2.2.5    | 9.2 MB   | #####| 100%  100%
icc_rt-2022.1.0     | 6.5 MB   | #####| 100%  100%
libhwloc-2.12.1     | 2.3 MB   | #####| 100%  100%
intel-openmp-2025.0. | 2.1 MB   | #####| 100%  100%
setuptools-80.9.0    | 1.4 MB   | #####| 100%  100%
pip-25.3             | 1.1 MB   | #####| 100%  100%
tbb-devel-2022.3.0    | 1.1 MB   | #####| 100%  100%
mkl_random-1.3.0     | 245 KB   | #####| 100%  100%
tbb-2022.3.0          | 162 KB   | #####| 100%  100%
mkl_fft-2.1.1          | 146 KB   | #####| 100%  100%
wheel-0.45.1            | 145 KB   | #####| 100%  100%
ca-certificates-2025 | 125 KB   | #####| 100%  100%
libexpat-2.7.3           | 119 KB   | #####| 100%  100%
mkl-service-2.5.2        | 72 KB    | #####| 100%  100%
expat-2.7.3              | 31 KB    | #####| 100%  100%
numpy-2.2.5                | 11 KB    | #####| 100%  100%
blas-1.0                  | 6 KB     | #####| 100%  100%
```

active ve deactivate olması:

```
conda activate yapayzeka_env
conda deactivate
```



## VS Code'u environment ile ilişkilendirelim

- VS Code'u açt
- Ctrl + Shift + P
- **Python: Select Interpreter**
- Listeden: Python 3.10.x (yapayzeka\_env) seç

## Sonuç:

- VS Code'un sağ alt köşesinde: Python 3.10.x (yapayzeka\_env) yazdı

---

## Terminal ile interpreter farkını anladık

### Önemli kavram:

Interpreter seçmek ≠ terminalde environment aktif etmek

- Sağ alttaki seçim:
  - Run / Debug için
- Terminal:

- Manuel komutlar için
- 

## Terminalde env otomatik aktif olsun diye ayar yaptık

- Terminal açılınca (yapayzeka\_env) otomatik gelsin

### VS Code Ayarı:

Python > Terminal: Activate Environment Açıtık

### Sonuç:

- Yeni terminal açınca: (yapayzeka\_env) PS ...
- 

## Neden “Run” ile hata aldık?

Çünkü:

- [Running] python -u ...
- **Code Runner eklentisi**
- Sistem Python'u kullandı

### Çözüm:

- Code Runner kullanma
- **Run Python File in Terminal** kullan

## Yapay zekaya adım

### NumPy (Numerical Python)

NumPy, Python'da **sayısal hesaplamalar** yapmak için kullanılan en temel ve en hızlı kütüphanelerden biridir.

Özellikle **çok boyutlu diziler (array)** üzerinde çalışmak için tasarlanmıştır. Normal Python listelerine göre **daha az bellek kullanır ve çok daha hızlı işlem yapar**.

NumPy;

- Vektör ve matris işlemleri
- Lineer cebir hesaplamaları
- İstatistiksel işlemler
- Matematiksel fonksiyonlar

gibi konularda yoğun olarak kullanılır. Yapay zekâ ve makine öğrenmesi algoritmalarının büyük çoğunluğu **NumPy dizileri** üzerinden çalışır.

## NumPy Örnek

```
import numpy as np  
dizi = np.array([1, 2, 3, 4, 5])  
print(dizi * 2)
```

Bu örnekte NumPy dizisindeki tüm elemanlar **tek satırda** 2 ile çarpılmaktadır.

## SciPy

SciPy, **NumPy üzerine inşa edilmiş**, bilimsel ve mühendislik hesaplamaları için kullanılan **çok yönlü bir Python kütüphanesidir**.

Özellikle veri analizi, sinyal işleme, optimizasyon ve istatistiksel hesaplamalarda yaygın olarak kullanılır.

SciPy; **kümeleme, regresyon (tahmin), veri işleme, sayısal integrasyon, diferansiyel denklemler** ve **optimizasyon** gibi birçok ileri seviye işlemi gerçekleştirebilecek hazır fonksiyonlar sunar. Bu sayede karmaşık matematiksel işlemler, düşük seviyeli kod yazmaya gerek olmadan hızlı ve güvenilir bir şekilde uygulanabilir.

Makine öğrenmesi ve yapay zekâ projelerinde SciPy;

- Verilerin **ön işlenmesi**,
- Matematiksel modelleme,
- Optimizasyon problemlerinin çözümü gibi aşamalarda destekleyici bir rol üstlenir.

## Pandas

Pandas, **tablosal veriler** (Excel, CSV, veri tabloları) üzerinde işlem yapmak için kullanılan temel Python kütüphanesidir.

Verileri **satır ve sütun** mantığıyla ele alır ve veri analizini oldukça kolaylaştırır.

Pandas ile;

- Veri okuma (CSV, Excel, JSON)
- Eksik veri temizleme
- Filtreleme ve sıralama
- İstatistiksel analiz

İşlemleri rahatlıkla yapılabilir. Yapay zekâ projelerinde veri genellikle Pandas ile **ön işleme (preprocessing)** aşamasından geçirilir.

## Pandas Örnek

```
import pandas as pd
veri = { "Yaş": [20, 22, 21],
          "Not": [85, 90, 88] }
df = pd.DataFrame(veri) print(df)
```

Bu örnekte bir tablo oluşturulmuş ve satır–sütun yapısında ekrana yazdırılmıştır.

---

## Matplotlib

Matplotlib, verileri **grafik ve görseller** ile ifade etmek için kullanılan bir çizim kütüphanesidir. Sayısal verilerin daha kolay anlaşılabilmesi için **çizgi grafikleri, sütun grafikleri, dağılım grafikleri** gibi görseller üretir.

Yapay zekâ ve veri analizinde;

- Veri dağılımlarını incelemek
- Model sonuçlarını görselleştirmek
- Eğitim sürecindeki hata (loss) grafiklerini çizmek

amacıyla sıkça kullanılır.

## Matplotlib Örnek

```
import matplotlib.pyplot as plt
x = [1, 2, 3, 4]
y = [10, 20, 15, 25]
plt.plot(x, y)
plt.xlabel("X Değeri")
plt.ylabel("Y Değeri")
plt.title("Basit Grafik Örneği")
plt.show()
```

Bu örnekte verilen x ve y değerleri kullanılarak basit bir çizgi grafiği oluşturulmuştur.

## Scikit-Learn

Scikit-Learn, **makine öğrenmesi algoritmalarını kolay ve hızlı bir şekilde uygulamak** için kullanılan en yaygın Python kütüphanelerinden biridir.

Veri ön işleme, model eğitimi, test ve değerlendirme adımlarını tek bir yapı altında sunar.

Scikit-Learn;

- **Regresyon** (sayısal tahmin),
  - **Sınıflandırma** (etiket tahmini),
  - **Kümeleme** (benzer verileri graplama)
- gibi temel makine öğrenmesi problemlerinde kullanılan çok sayıda algoritmayı hazır olarak içerir.

Derin öğrenme yerine, **klasik makine öğrenmesi** problemleri için tercih edilir.

## Basit Örnek – Doğrusal Regresyon

```
from sklearn.linear_model import LinearRegression
import numpy as np
X = np.array([[1], [2], [3], [4]])
y = np.array([2, 4, 6, 8])
model = LinearRegression()
model.fit(X, y)
print(model.predict([[5]])) # Tahmin`
```

## TensorFlow

TensorFlow, **Google tarafından geliştirilen**, büyük ölçekli **derin öğrenme ve yapay sinir ağları** uygulamaları için kullanılan açık kaynaklı bir kütüphanedir.

Özellikle görüntü işleme, doğal dil işleme ve büyük veri setleri üzerinde model eğitimi için kullanılır.

TensorFlow;

- Otomatik türev alma,
- GPU/CPU desteği.
- Büyük modelleri verimli şekilde eğitme gibi güçlü özellikler sunar.

## Basit Örnek – Tek Katmanlı Sinir Ağı

```
import tensorflow as tf
model = tf.keras.Sequential([tf.keras.layers.Dense(1, input_shape=(1,))])
model.compile(optimizer='adam', loss='mse')
print("TensorFlow modeli hazır")`
```

## Keras

Keras, **TensorFlow üzerinde çalışan**, sinir ağlarını **hızlı ve kolay şekilde kurmak** için geliştirilmiş yüksek seviyeli bir kütüphanedir.

Kod okunabilirliğini artırır ve karmaşık ağları birkaç satırla tanımlamaya olanak tanır.

Keras;

- Katman tabanlı yapı,
  - Hızlı prototipleme,
  - Kolay eğitim ve test süreçleri
- sunarak özellikle **öğrenme ve hızlı deneme** aşamalarında tercih edilir.

## Basit Örnek – Keras ile Sinir Ağı

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
model = Sequential()
model.add(Dense(8, activation='relu', input_shape=(4,)))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')
print("Keras modeli oluşturuldu")`
```

## PyTorch

PyTorch, **Facebook (Meta)** tarafından geliştirilen, özellikle **esnekliği ve dinamik yapısı** ile öne çıkan bir derin öğrenme kütüphanesidir.

Araştırma ve akademik çalışmalarında oldukça yaygın olarak kullanılır.

PyTorch'un en büyük avantajı;

- Dinamik hesaplama grafiği,
- Python'a çok yakın söz dizimi,
- Model üzerinde adım adım kontrol imkânı sunmasıdır.

## Basit Örnek – PyTorch Tensör Kullanımı

```
import torch
x = torch.tensor([1.0, 2.0, 3.0])
y = x * 2
print(y)
```

## Kütüphane kurulumları

VS Code terminalinde veya CMD'ye:

```
conda activate yapayzeka_env
```

Terminal başında şunu görmelisin:

```
(yapayzeka_env)
```

NumPy, Pandas ve Matplotlib Kur

```
conda install numpy
conda install pandas
conda install matplotlib
conda install scipy
conda install scikit-learn
conda install pytorch torchvision torchaudio cpuonly -c pytorch
pip install tensorflow
```

## Yapayzeka hızlı bir örnek

Eğitilebilir Yapay Zekâ Platform ile Taş Kâğıt Makas Oyununun Modellenmesi için aşağıda verilen linki tıklayın. <https://teachablemachine.withgoogle.com/>

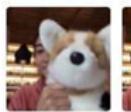
# Teachable Machine

Train a computer to recognize your own images, sounds, & poses.

A fast, easy way to create machine learning models for your sites, apps, and more – no expertise or coding required.

Get Started

Tıklayınız.



**Image Project**

Tıklayınız.



**Audio Project**

Teach based on one-second-long sounds, from files or your microphone.



**Pose Project**

Teach based on images, from files or your webcam.

## New Image Project

**Standard image model**

Best for most uses

224x224px color images

Export to TensorFlow, TFLite, and TF.js

Model size: around 5mb

Tıklayınız.

**Embedded image model**

Best for microcontrollers

96x96px greyscale images

Export to TFLite for Microcontrollers, TFLite, and TF.js

Model size: around 500kb

[See what hardware supports these models.](#)

Class isimlerini “Taş”, “Kâğıt” ve “Makas” olarak isimlendirin

Taş

Add Image Samples:

Webcam Upload

Kâğıt

Add Image Samples:

Webcam Upload

Makas

Add Image Samples:

Webcam Upload

Training

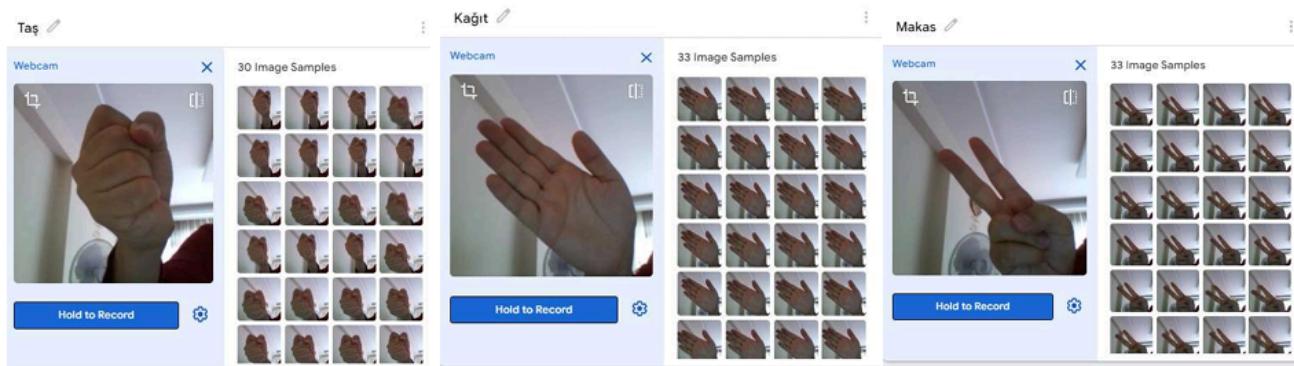
Train Model

Advanced

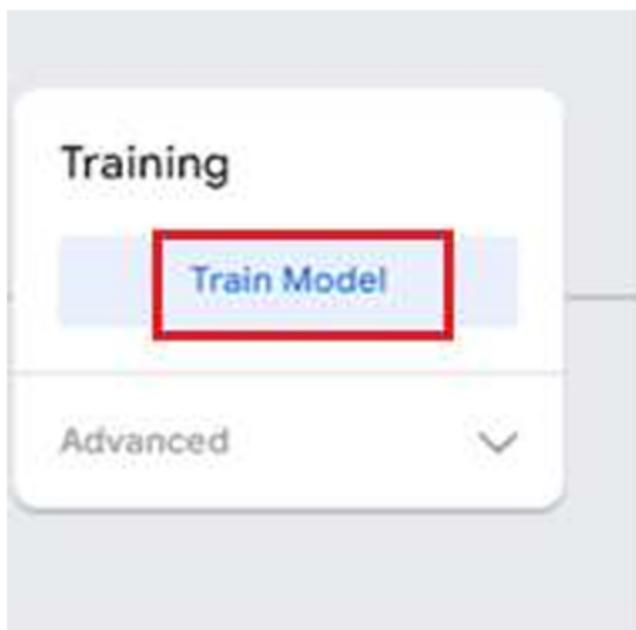
Preview Export Model

You must train a model on the left before you can preview it here.

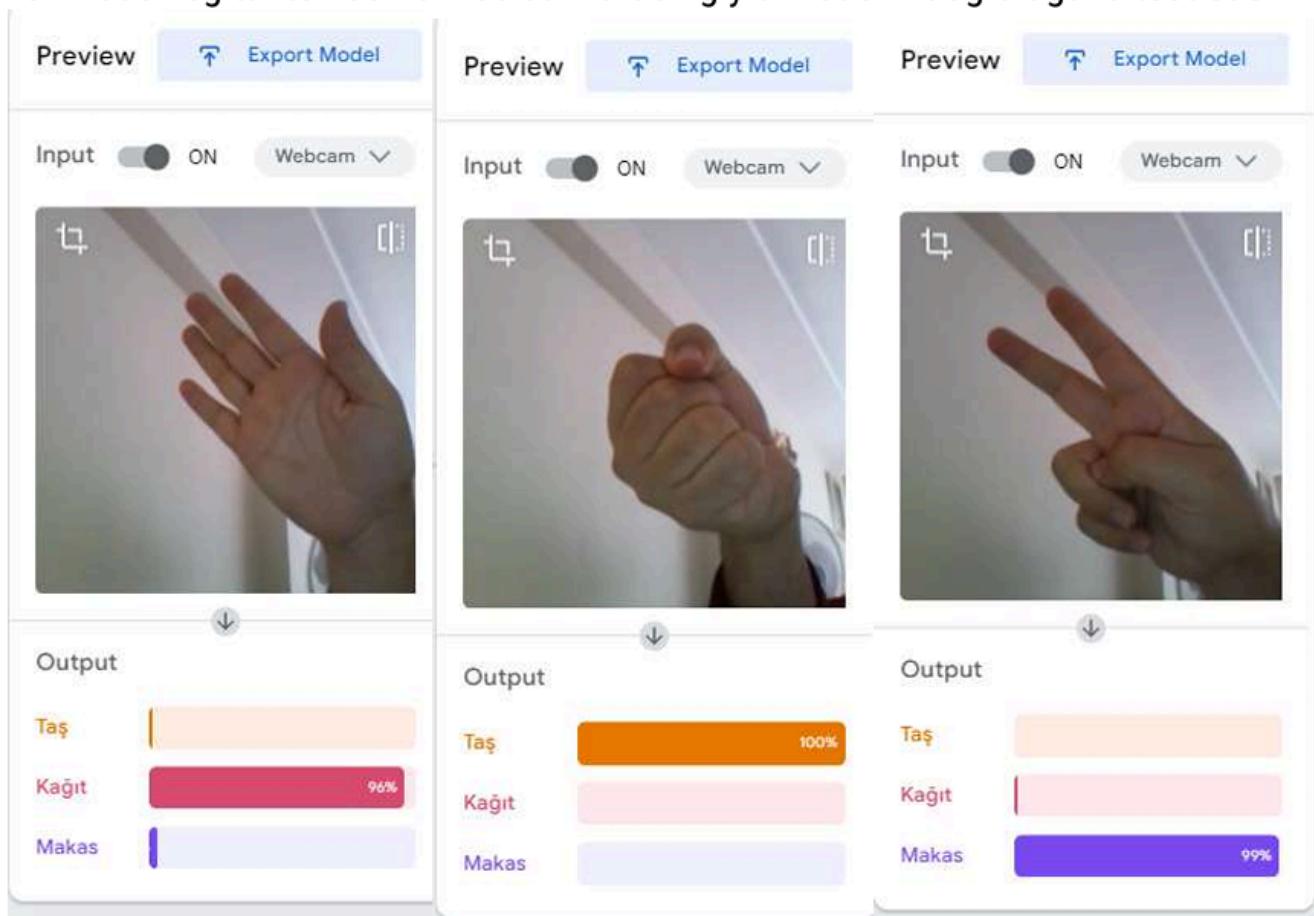
Python ile Yapay Zekâ Öğrenciler “hold to record” butonuna basarak taş, kâğıt ve makas oyunu için görüntü veri seti oluşturun



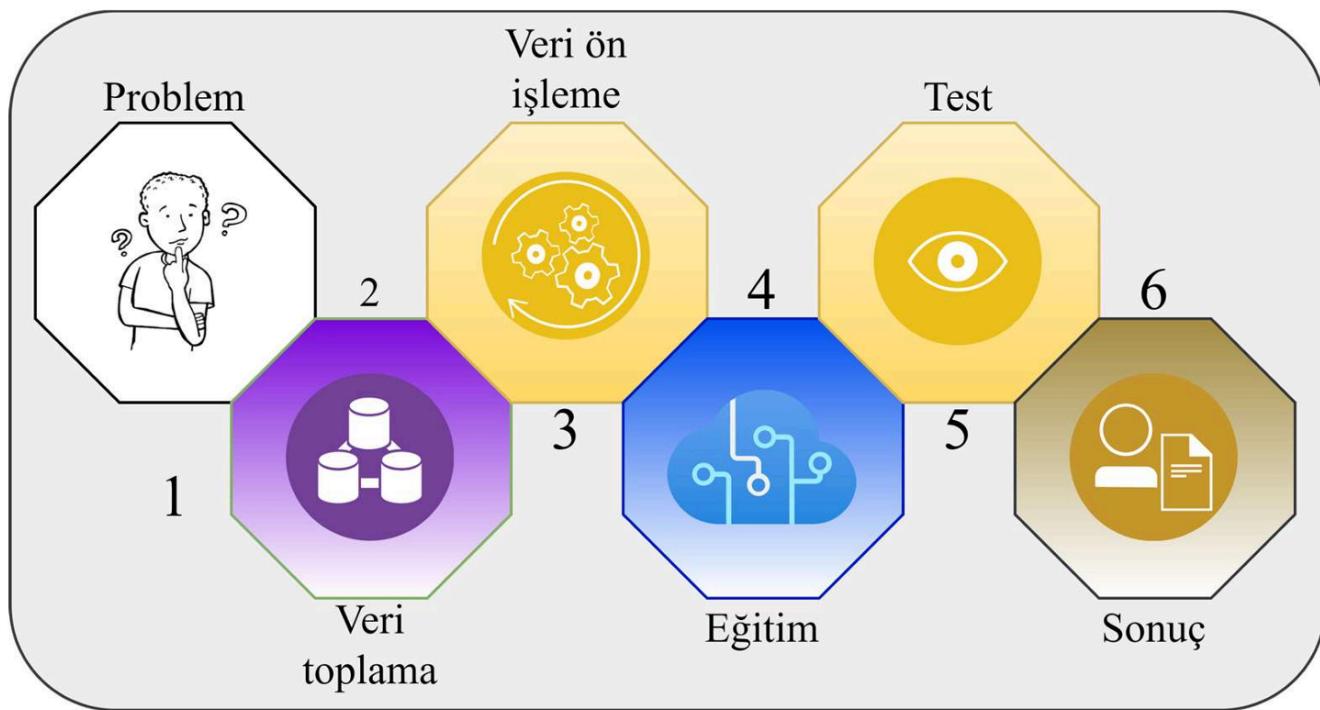
“train model” butona basarak görüntü veri setinin eğitim işlemini gerçekleştirin

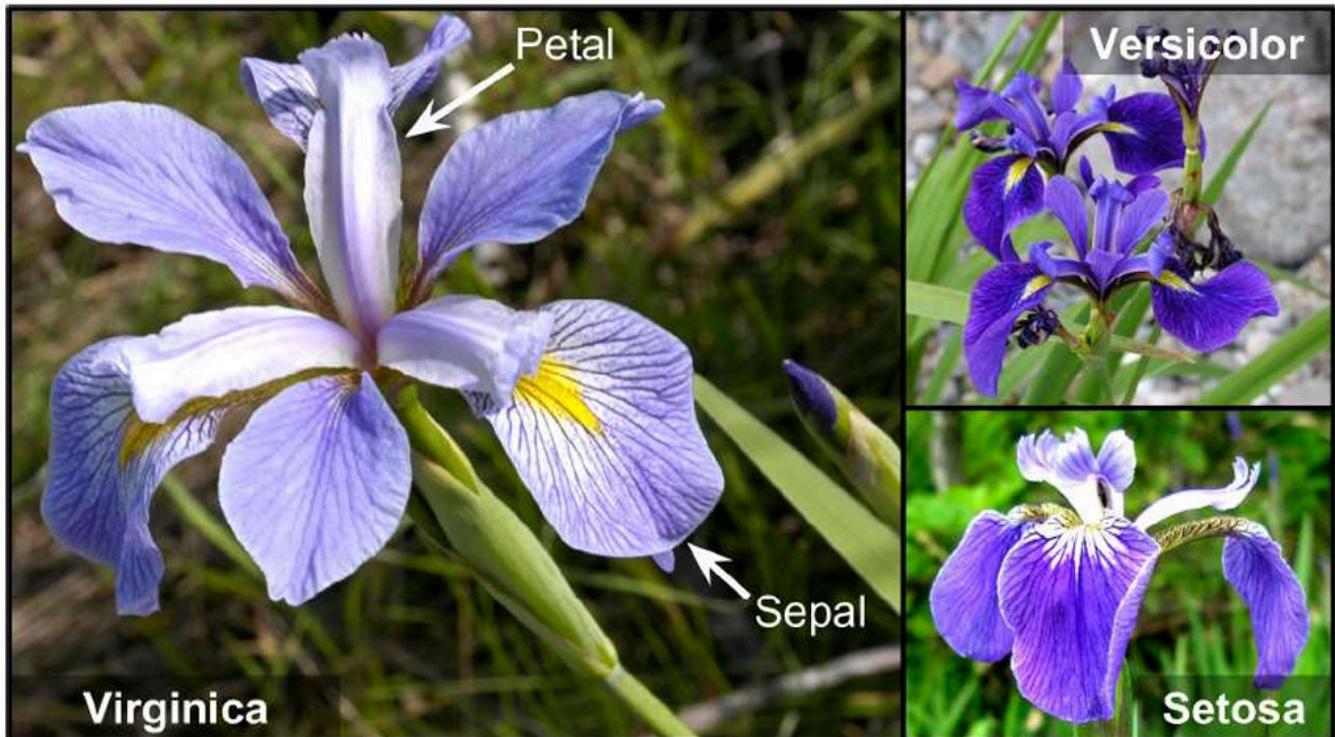
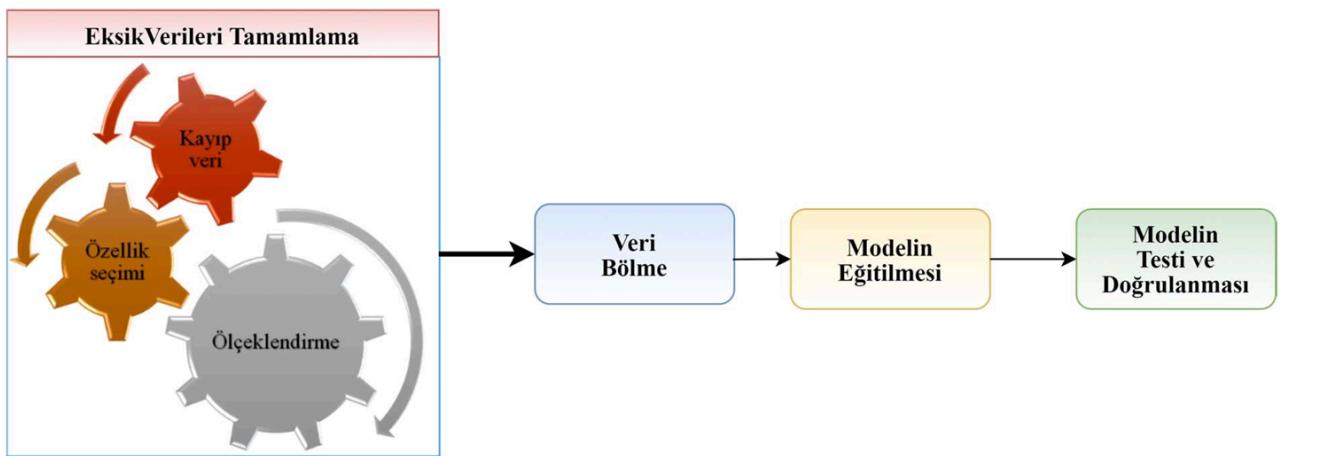


Modeli eğittiğten sonra web cam aracılığıyla modelin doğruluğunu test eder.



## Yapay Zeka kullanımı etkinliği





Şekil 1.13. İris çiçeği ve türleri (O'Reilly, 2021)

Örnek Numara	Alt yaprak uzunluğu (cm)	Alt yaprak genişliği (cm)	Üst yaprak uzunluğu (cm)	Üst yaprak genişliği (cm)	Tür
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...	...	...	...	...	...
51	7.0	3.2	4.7	1.4	Versicolor
52	6.4	3.2	4.5	1.5	Versicolor
...	...	...	...	...	...
101	6.3	3.3	6.0	2.5	Virginica
102	5.8	2.7	5.1	1.9	Virginica
...	...	...	...	...	...
150	5.9	3.0	5.1	1.8	Virginica

---

## Iris Veri Setinin Yüklenmesi

```
from sklearn.datasets import load_iris  
iris = load_iris()
```

- `load_iris()` fonksiyonu, Scikit-Learn içinde hazır gelen **Iris çiçeği veri setini** yükler.
- Bu veri seti:
  - **150 adet örnek**
  - **3 farklı çiçek türü** içerir:
    - setosa
    - versicolor
    - virginica

---

## Veri Seti İçeriğinin İncelenmesi

```
print(iris.feature_names) print(iris.target_names) print(iris.target)  
print(iris.data)
```

- `feature_names` Girdi özellikleri (bağımsız değişkenler):
  - sepal length
  - sepal width
  - petal length
  - petal width

- `target_names` Çiçek sınıfları (etiketler)
  - `target` Her veri satırının **hangi sınıfı ait olduğunu gösteren sayısal etiketler** (`0=setosa` , `1=versicolor` , `2=virginica`)
  - `data`  
Gerçek ölçüm verileri (X)
- 

## Girdi (X) ve Çıkış (Y) Değerlerinin Ayrılması

X = iris.data Y = iris.target

- X : Modelin öğreneceği **Özellikler**
  - Y : Modelin tahmin etmeye çalışacağı **sınıf etiketleri**
- 

## Eğitim ve Test Veri Setlerinin Ayrılması

```
from sklearn.model_selection import train_test_split

X_train, X_test, Y_train, Y_test =
train_test_split(X,Y,test_size=0.20,random_state=0 )
```

- Veri seti **eğitim** ve **test** olmak üzere ikiye ayrılır.
  - Toplam verinin:
    - %80 'i eğitim verisi
    - %20 'si test verisi olarak kullanılır.
  - `random_state=0` parametresi, her çalışmada **aynı veri bölünmesini** sağlar (tekrar edilebilirlik).
- 

## Karar Ağacı (Decision Tree) Modelinin Oluşturulması

```
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier()
```

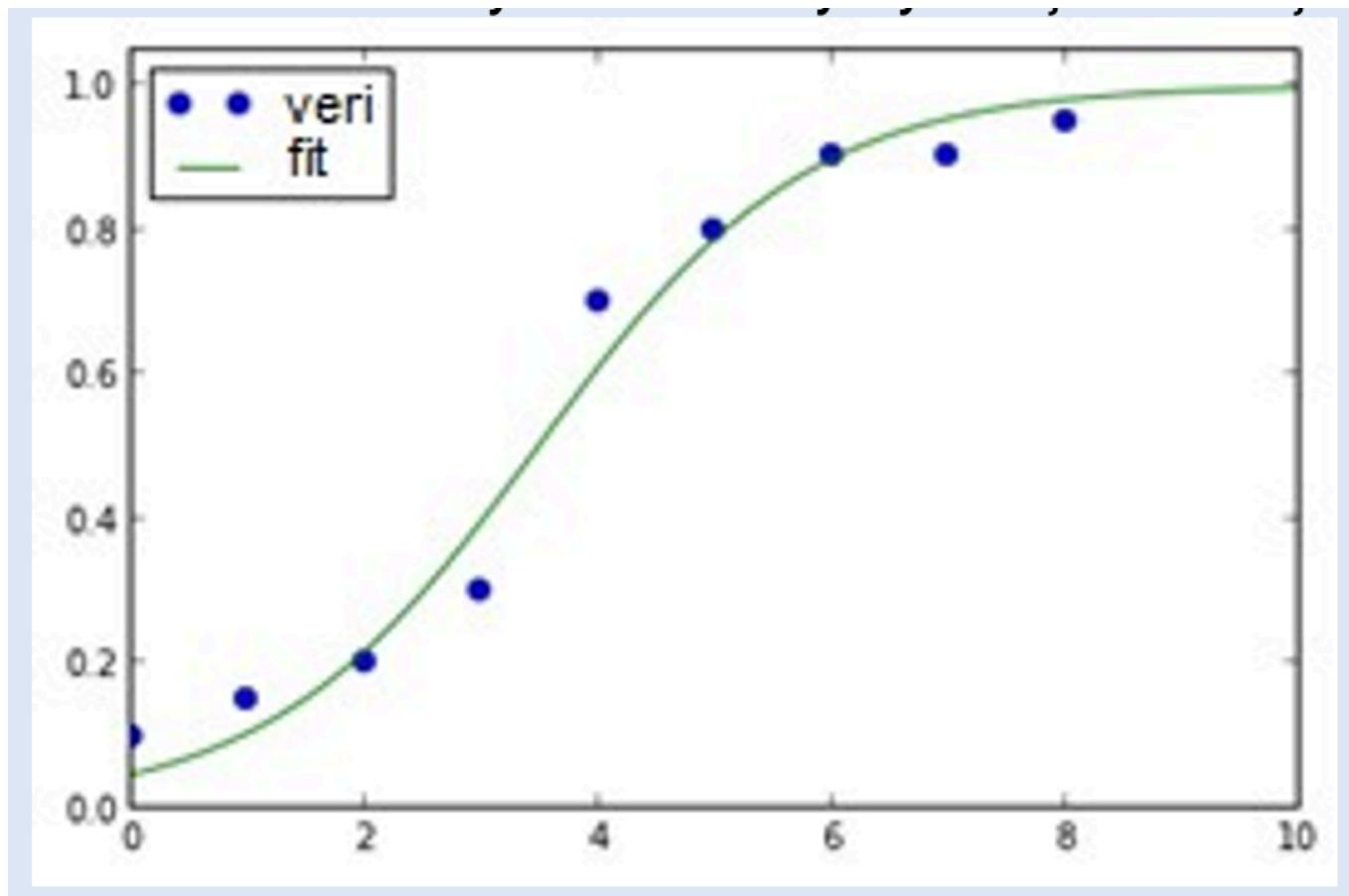
- **Decision Tree (Karar Ağacı)**, veriyi **if-else mantığıyla** dallara ayırarak sınıflandıran bir algoritmadır.
- Özellik değerlerine göre kararlar alır ve en uygun sınıfı belirler.

- Anlaşılması ve yorumlanması kolay bir makine öğrenmesi yöntemidir.
- 

## Modelin Eğitilmesi

```
model.fit(X_train, Y_train)
```

- Model, eğitim verilerini kullanarak:
  - Girdi özellikleri ( `X_train` ) ile
  - Sınıf etiketleri ( `Y_train` ) arasındaki ilişkiyi öğrenir.
- Bu aşamada model **öğrenme sürecini** tamamlar.



---

## Test Verisi Üzerinde Tahmin Yapılması

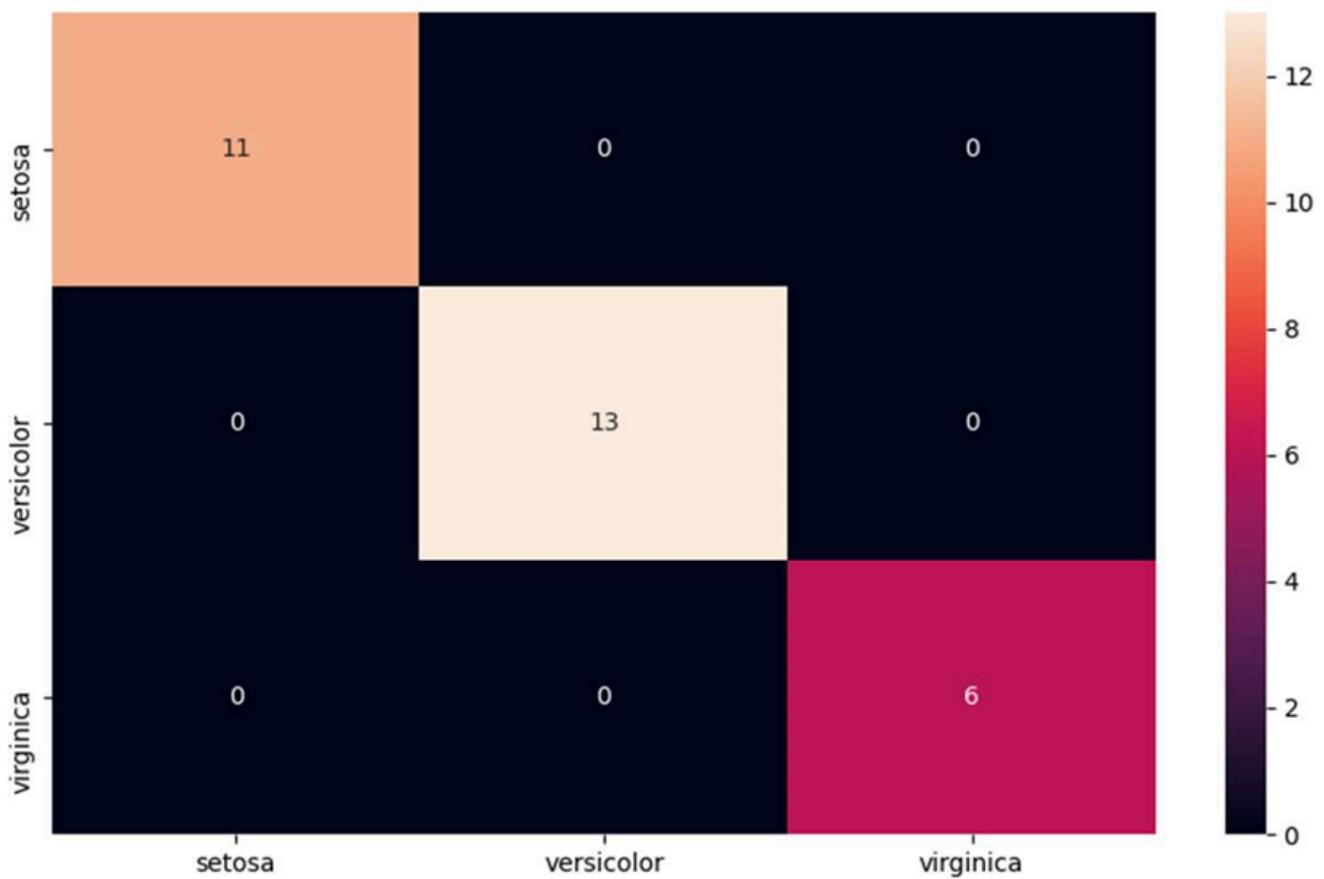
```
Y_tahmin = model.predict(X_test)
```

- Eğitilen model, daha önce **hiç görümediği test verileri** için sınıf tahmini yapar.
- Bu tahminler modelin gerçek başarısını ölçmek için kullanılır.

## Confusion Matrix (Hata Matrisi)

```
from sklearn.metrics import confusion_matrix
hata_matrisi = confusion_matrix(Y_test, Y_tahmin)
print(hata_matrisi)
```

- Confusion Matrix, modelin:
  - **Doğru tahminlerini**
  - **Yanlış tahminlerini**
  - **Satırlar → Gerçek sınıflar**
  - **Sütunlar → Tahmin edilen sınıflar**
- sınıf bazında göstermeyi sağlar.
- Modelin hangi sınıflarda hata yaptığına açıkça görmemize yardımcı olur.



## Hata Matrisinin Görselleştirilmesi

```
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt

index = ['setosa', 'versicolor', 'virginica']
columns = ['setosa', 'versicolor', 'virginica']
hata_goster = pd.DataFrame(hata_matrisi, columns=columns, index=index )
```

- Hata matrisi, **Pandas DataFrame** formatına dönüştürülür.
  - Satırlar: **gerçek sınıflar**
  - Sütunlar: **tahmin edilen sınıflar**
- 

### Heatmap ile Görsel Gösterim

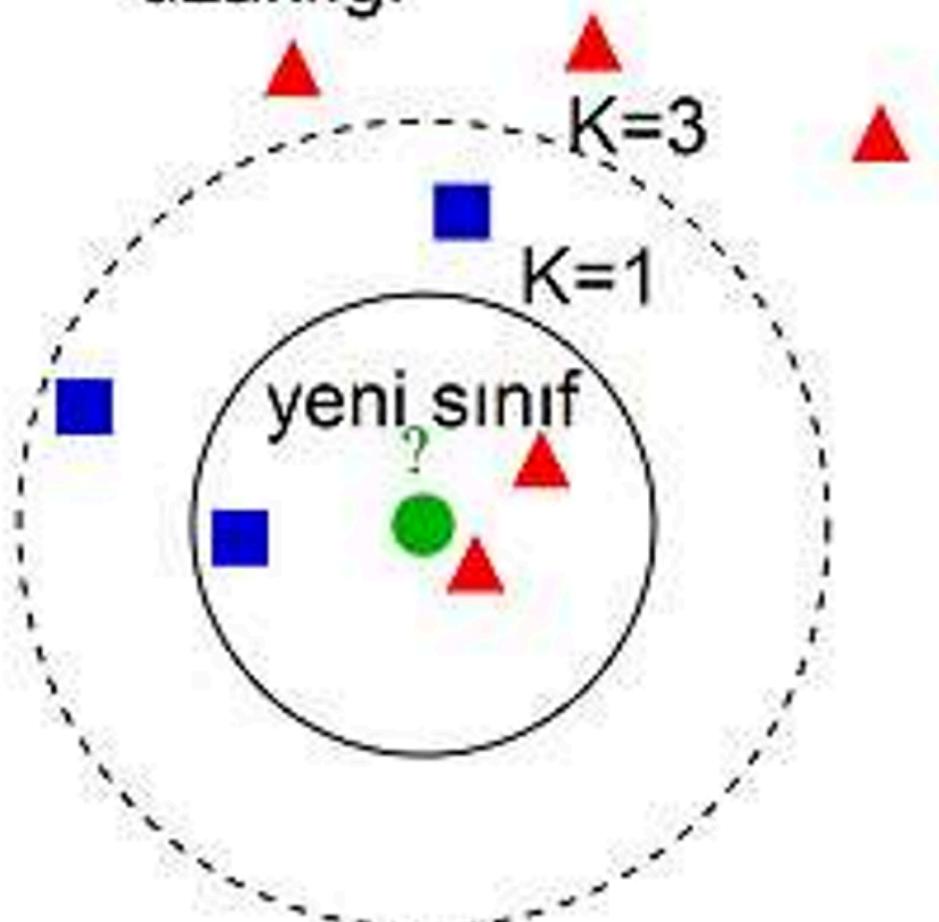
```
plt.figure(figsize=(10,6))
sns.heatmap(hata_goster, annot=True)
plt.show()
```

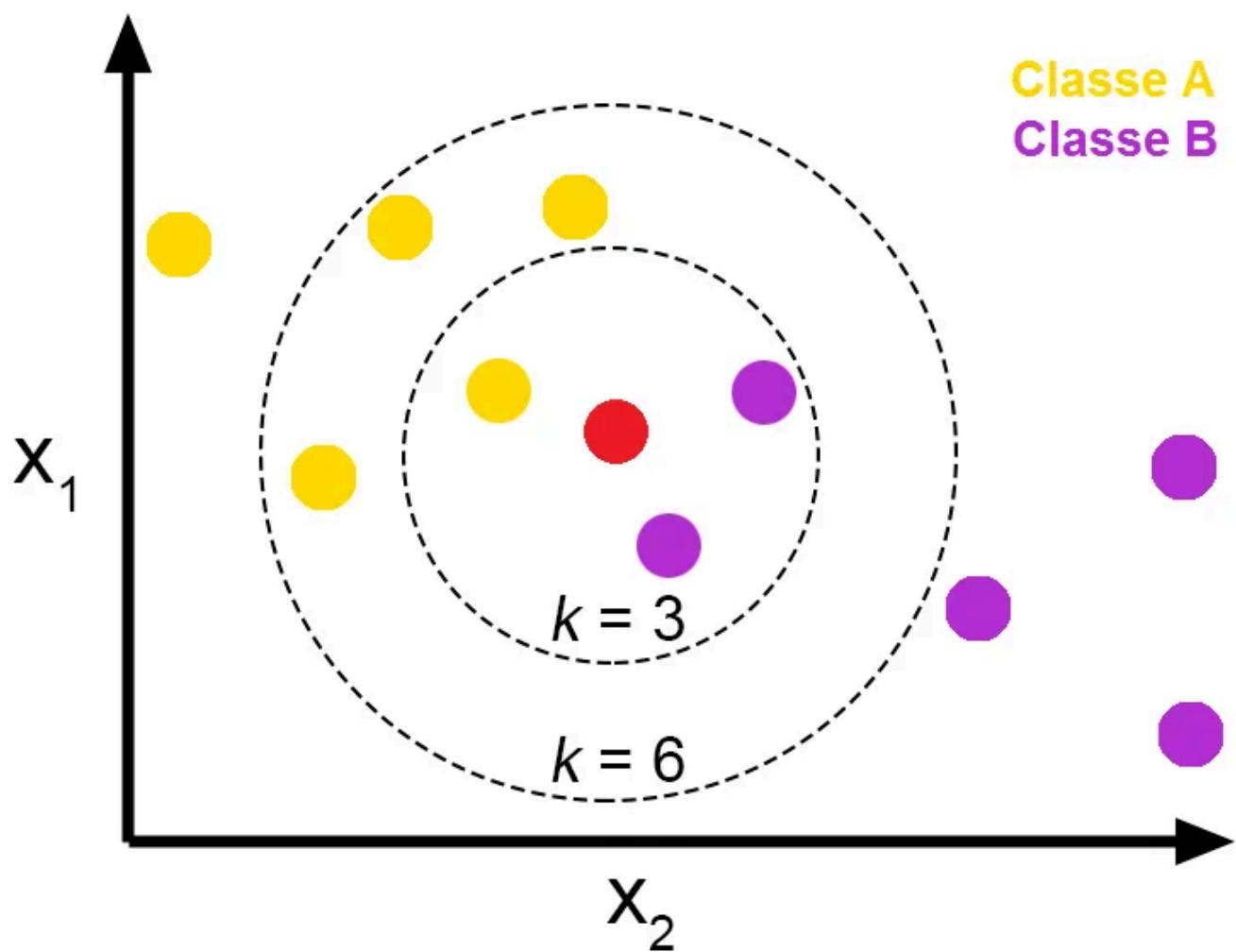
- **Heatmap**, hata matrisini renkli bir şekilde gösterir.
- Büyük değerler koyu, küçük değerler açık renkle ifade edilir.
- `annot=True` parametresi hücrelerin içine sayısal değerleri yazar.

## KNN

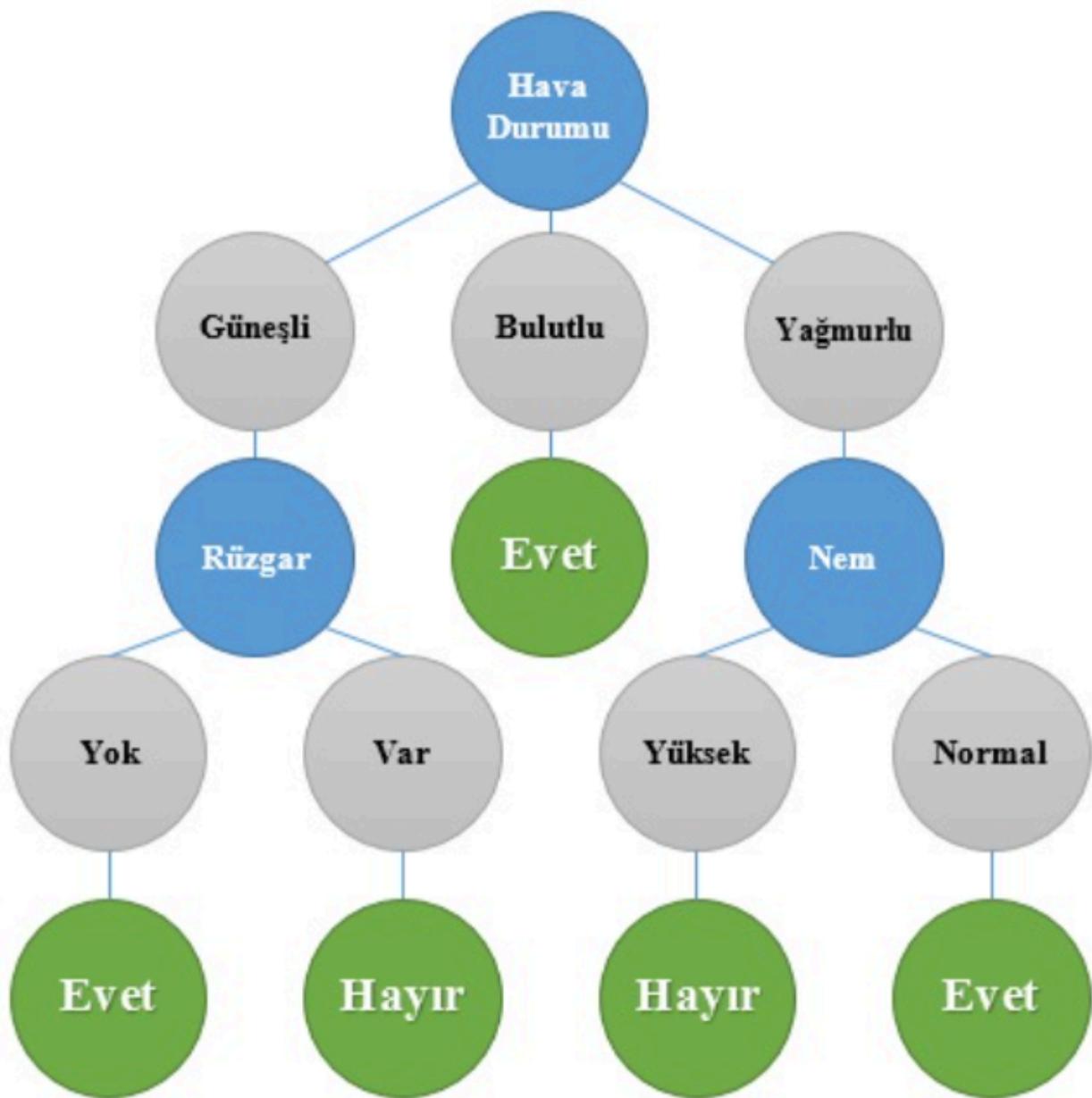
## Eğitim verisinin uzaklığı

Sınıf





Karar ağaçları



Özellikler				Hedef
Hava Durumu	Sıcaklık	Nem	Rüzgar	Futbol Oyna
Yağmurlu	Sıcak	Yüksek	Yok	Hayır
Yağmurlu	Sıcak	Yüksek	Var	Hayır
Bulutlu	Sıcak	Yüksek	Yok	Evet
Güneşli	İllik	Yüksek	Yok	Evet
Güneşli	Soğuk	Normal	Yok	Evet
Güneşli	Soğuk	Normal	Var	Hayır
Bulutlu	Soğuk	Normal	Var	Evet
Yağmurlu	İllik	Yüksek	Yok	Hayır
Yağmurlu	Soğuk	Normal	Yok	Evet
Güneşli	İllik	Normal	Yok	Evet
Yağmurlu	İllik	Normal	Yok	Evet
Bulutlu	İllik	Yüksek	Var	Evet
Bulutlu	Sıcak	Normal	Yok	Evet
Güneşli	İllik	Yüksek	Var	Hayır

## EKLER

