

K En Yakın Komşu - KNN

K En Yakın Komşu - KNN nedir?

- K-en yakın komşular (KNN) algoritması, hem sınıflandırma hem de regresyon problemlerini çözmek için kullanılabilecek basit, uygulaması kolay bir denetimli makine öğrenimi algoritmasıdır.

KNN ve K means arasındaki farklar nedir?

- Her ne kadar KNN algoritması k-means algoritmasındaki benzer özellikler taşısa da büyük farklılıklar da içermektedir. KNN algoritması bir eğitim verisi içerirken k-means algoritması bir eğitim verisi içermez. Yeni bir değer geldiğinde K değerine mesafeler hesaplanır ve yeni değer bir kümeye ilave edilir. Mesafe hesaplama işleminde ise k-means ve hiyerarşik kümeleme de kullanılan öklid uzaklığı, manhattan uzaklığı gibi mesafe hesaplama yöntemleri kullanılabilir.

KNN ne yapar?

- KNN algoritması kullanarak oluşturduğumuz bir model, kendisine daha sonra sınıflandırması için verilen gözlemin, eğitim verisetindeki gözlemlerle uzaklığını hesaplayarak benzerliklerini bulur ve buna göre sınıflandırma tahmini yapar. Uzaklığı hesaplamak için çeşitli hesaplama yöntemleri olsada genellikle manhattan ve öklid uzaklıkları kullanılır. Bizde yapacağımız örneklerde öklid uzaklığını kullanacağız.

KNN ne yapar?

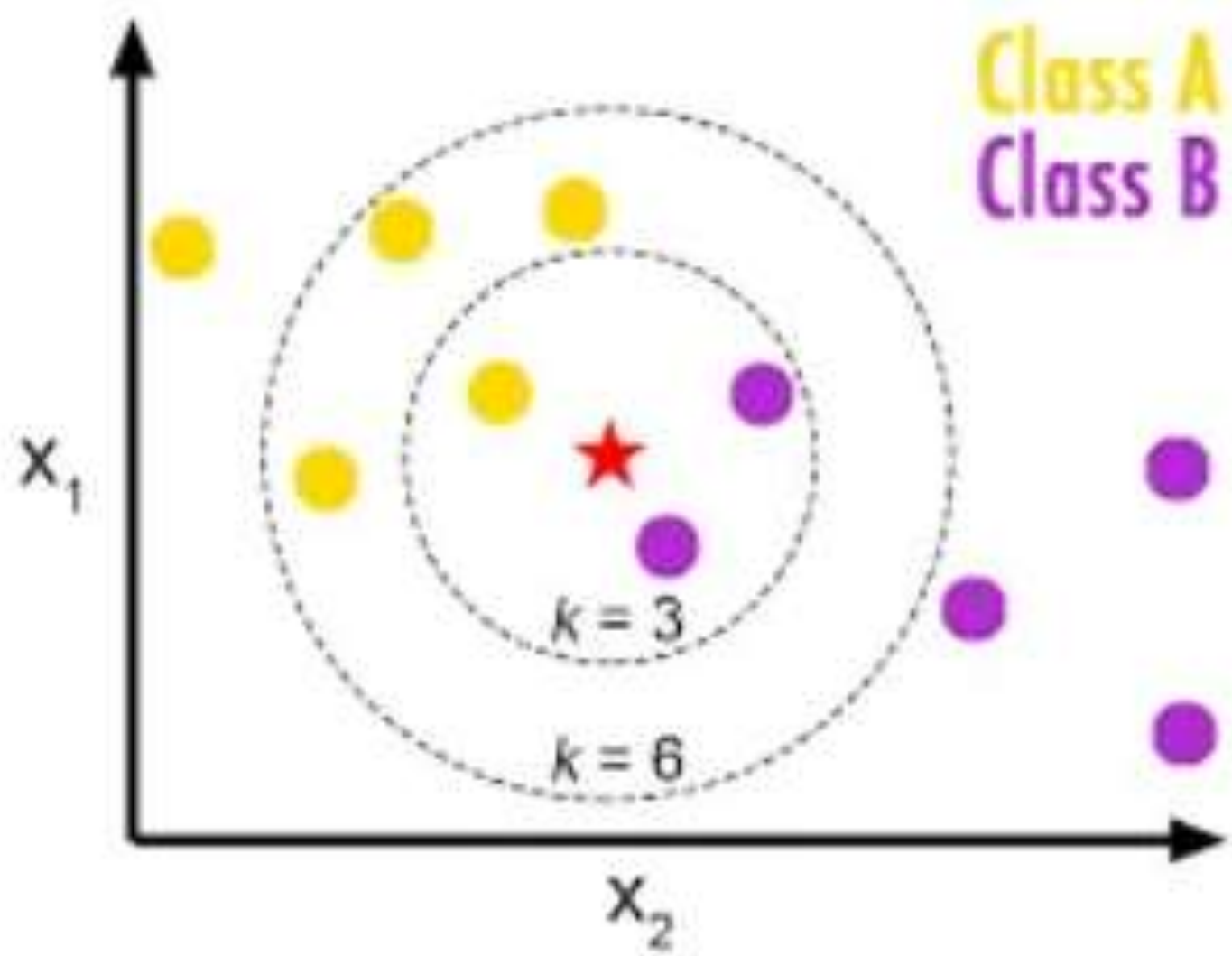
- KNN, birbirine yakın olan gözlemler birbirine benzeyen gözlemlerdir esasına dayanarak sınıflandırma gerçekleştirir. Burada ekosistemleri örnek olarak düşünebiliriz. Her canlının ait olduğu bir ekosistem, yaşamını sürdürebilmesi için ihtiyaç duyduğu koşullar vardır. KNN'de verilerimizi ekosisteme göre sınıflandırmasada, özellikleri ördeklerin habitatında bulunan özelliklere benzeyen bir gözlemi "ördek" olarak etiketleyebilir. Bu sadece mantığı anlaşılabilir kılmak için verilmiş bir örnektir, KNN bölgeleri tespit edip işaretleme, gruplama yada kümeleme yapmaz!

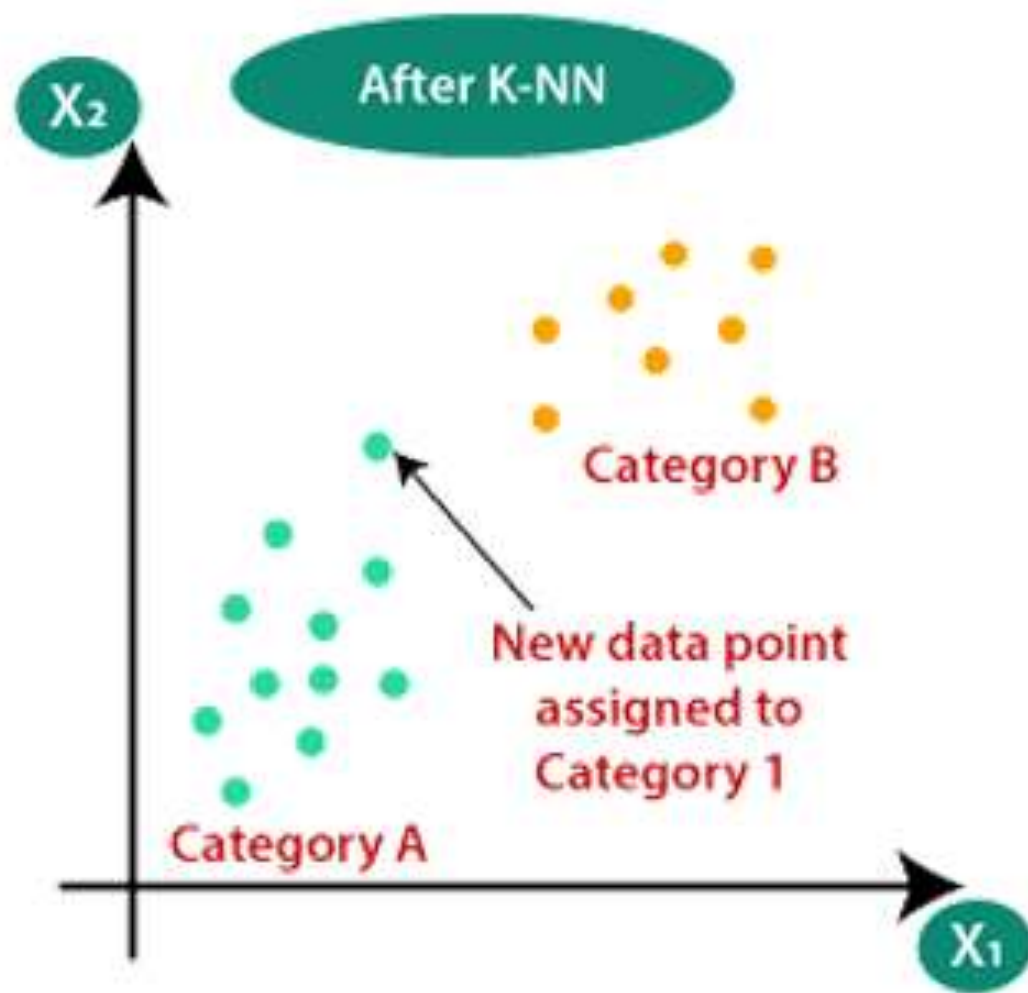
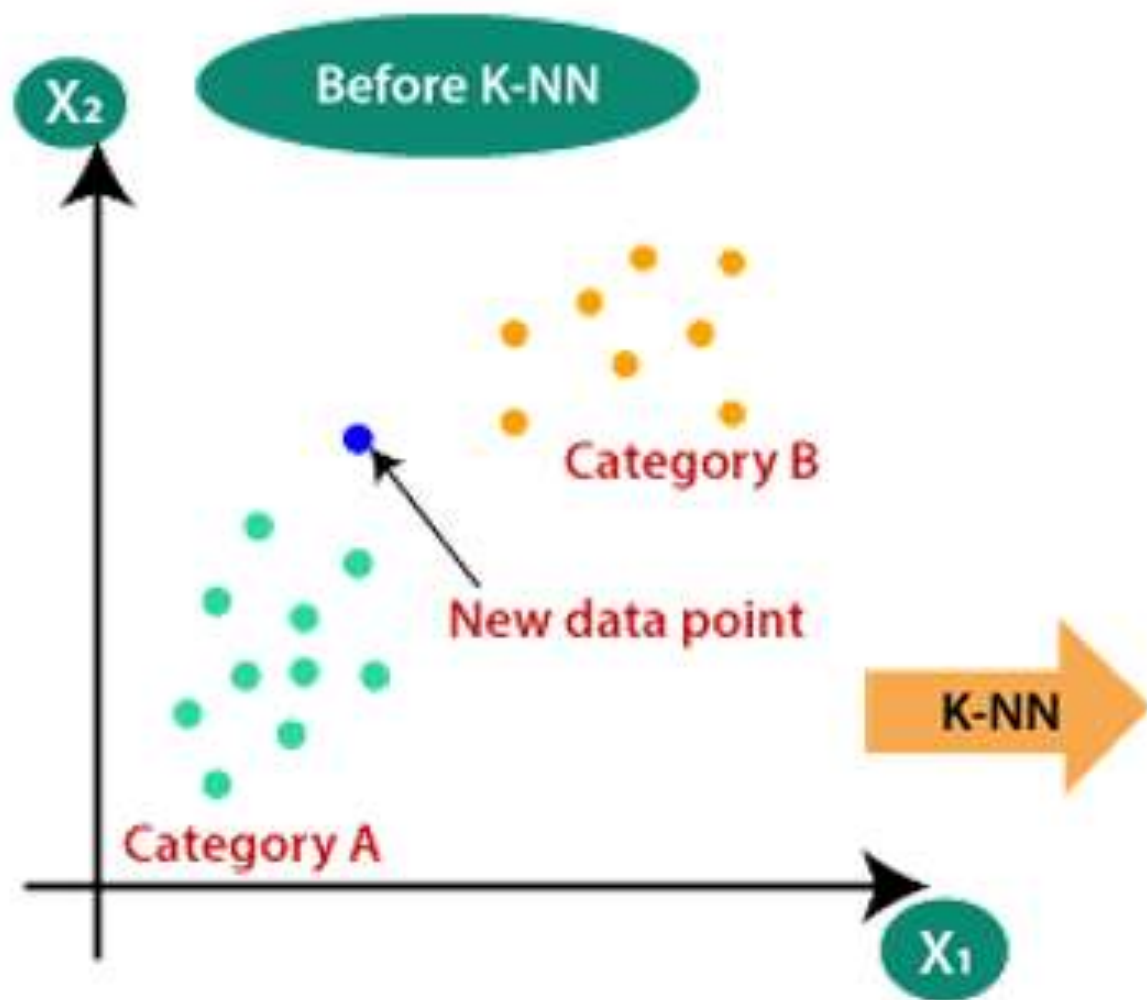
K-Nearest Neighbors algoritması 5 adımdan oluşur.

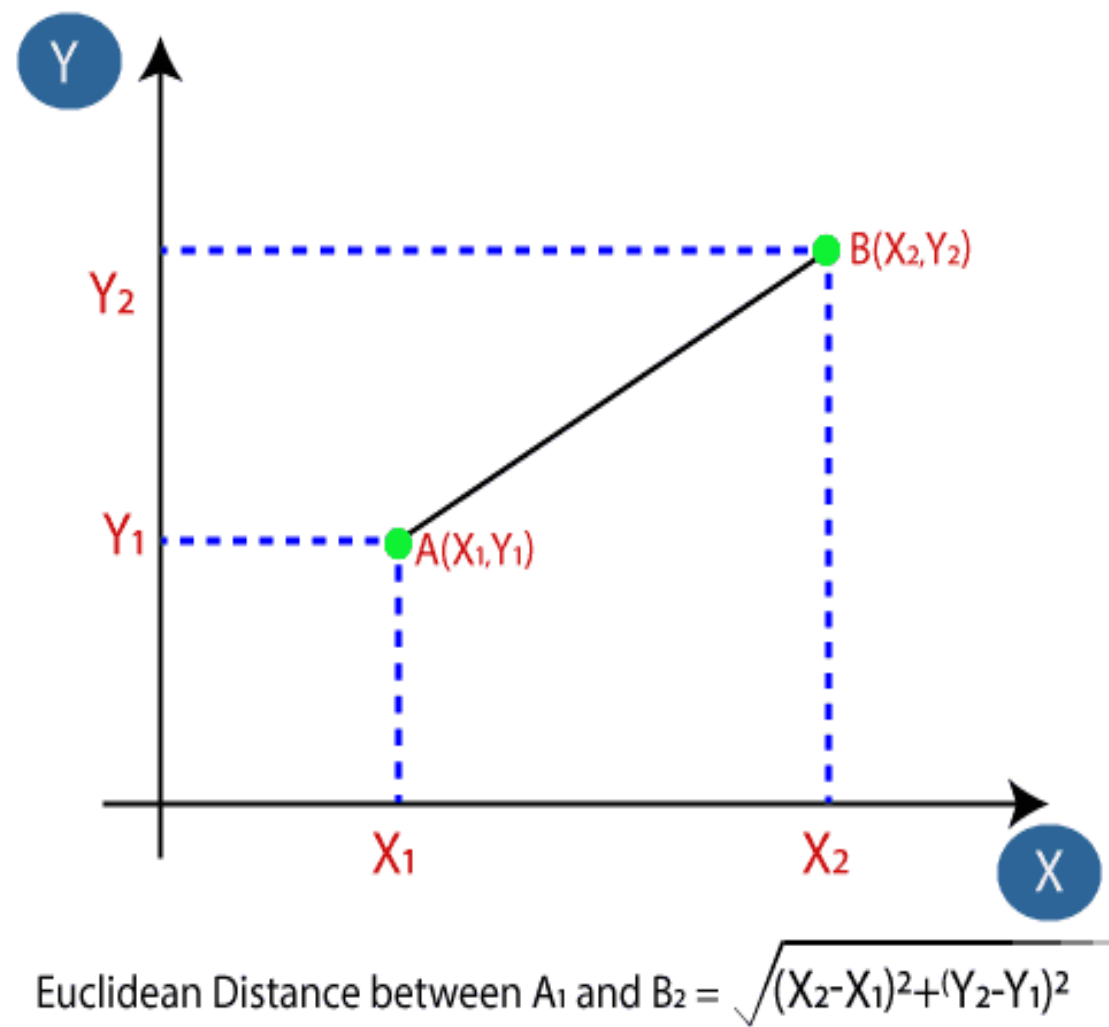
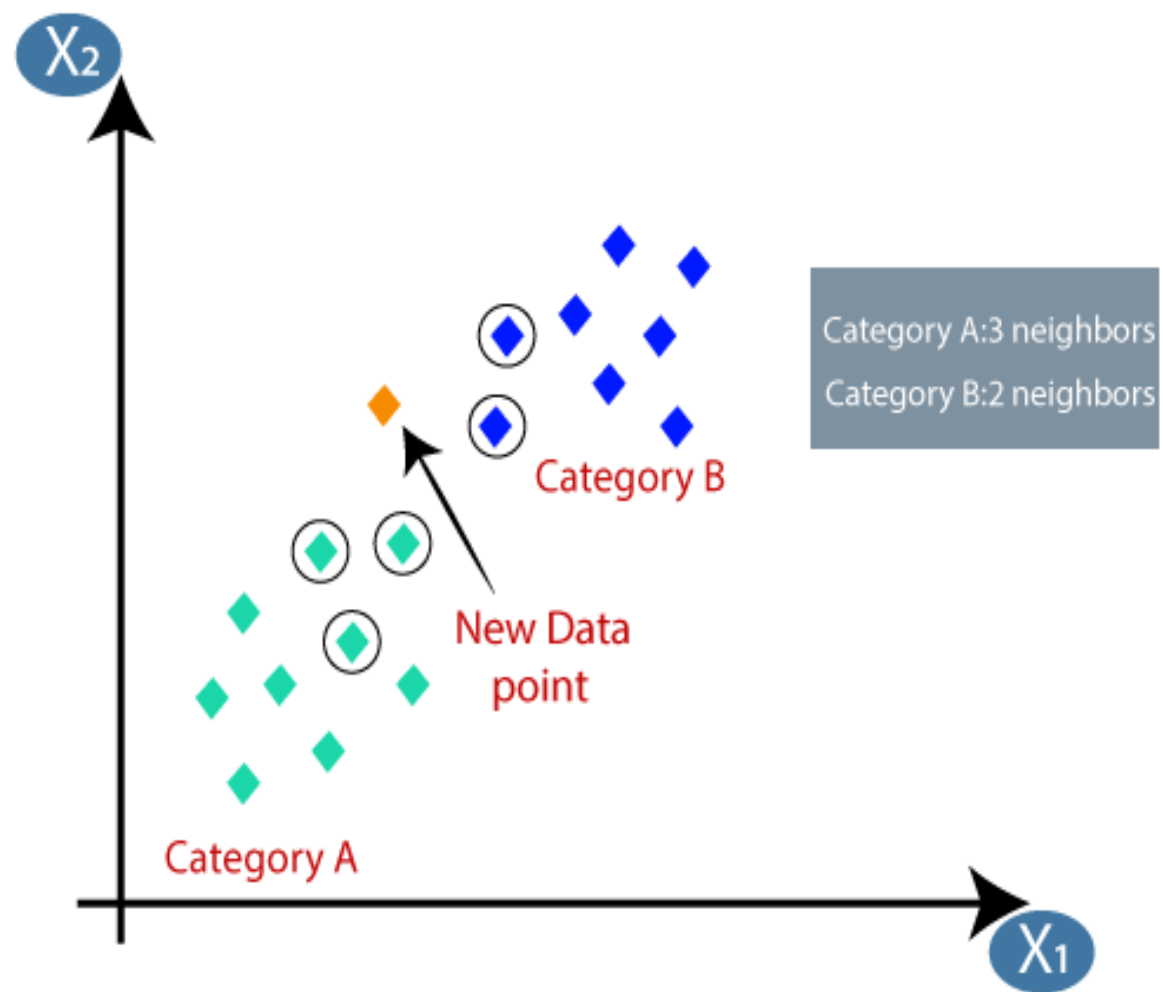
- 1- Öncelikle K değeri belirlenir.
- 2- Diğer nesnelerden hedef nesneye olan öklit uzaklıkları hesaplanır.
- 3- Uzaklıklar sıralanır ve en minimum uzaklığa bağlı olarak en yakın komşular bulunur.
- 4- En yakın komşu kategorileri toplanır.
- 5- En uygun komşu kategorisi seçilir.

KNN'nin mantığı nedir

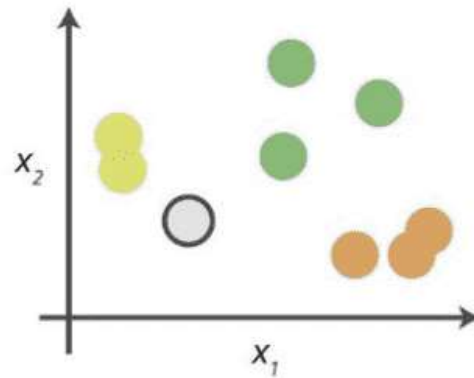
- KNNnin en temel haliyle uygulanması son derece kolaydır ve yine de oldukça karmaşık sınıflandırma görevlerini yerine getirir.
- Özel bir eğitim aşamasına sahip olmadığı için tembel bir öğrenme algoritmasıdır.
- Bunun yerine, yeni bir veri noktasını veya örneğini sınıflandırırken eğitim için tüm verileri kullanır.
- KNN, parametrik olmayan bir öğrenme algoritmasıdır, yani altta yatan veriler hakkında hiçbir şey varsaymaz. Bu son derece kullanışlı bir özelliktir, çünkü gerçek dünya verilerinin çoğu, doğrusal ayrılabilirlik, tekdüze dağılım vb. gibi herhangi bir teorik varsayımı gerçekten takip etmemektedir.





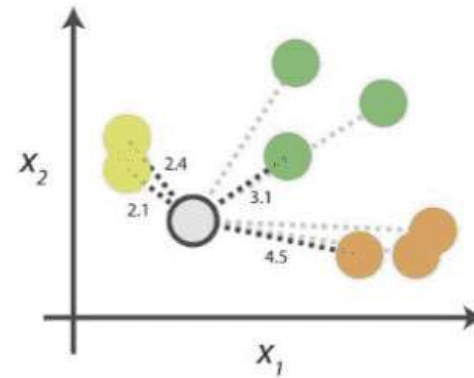


0. Look at the data











Say you want to classify the grey point into a class. Here, there are three potential classes - lime green, green and orange.

1. Calculate distances









Start by calculating the distances between the grey point and all other points.

2. Find neighbours

Point Distance			
	...	 2.1	→ 1st NN
	...	 2.4	→ 2nd NN
	...	 3.1	→ 3rd NN
	...	 4.5	→ 4th NN

Next, find the nearest neighbours by ranking points by increasing distance. The nearest neighbours (NNs) of the grey point are the ones closest in dataspace.

3. Vote on labels

Class	# of votes	
	2	➔ Class  wins the vote! Point  is therefore predicted to be of class  .
	1	
	1	

Vote on the predicted class labels based on the classes of the k nearest neighbours. Here, the labels were predicted based on the $k=3$ nearest neighbours.

File Edit Selection View Go Run Terminal Help

python

python denemeleri.py pjiopjpo.py hiyerarşik kümeleme.py KNN.py k-means.py

EXPLORER

PHYTON

hiyerarşik kümeleme.py k-means.py KNN.py python denemeleri.py pjiopjpo.py

KNN.py > _

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from sklearn.datasets import load_iris
4 from sklearn.model_selection import train_test_split
5 from sklearn.preprocessing import StandardScaler
6 from sklearn.neighbors import KNeighborsClassifier
7 from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
8
9 # Iris veri setini yükleme
10 iris = load_iris()
11 X, y = iris.data, iris.target
12
13 # Veri setini eğitim ve test olarak ayırma
14 X_train, X_test, y_train, y_test = train_test_split
15
16 # Veriyi standardize etme
17 scaler = StandardScaler()
18 X_train = scaler.fit_transform(X_train)
19 X_test = scaler.transform(X_test)
20
21 # KNN sınıflandırıcıyı oluşturma ve eğitme
22 knn = KNeighborsClassifier(n_neighbors=3)
23 knn.fit(X_train, y_train)
24
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Classification Report:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	10
1	1.00	1.00	1.00	9
2	1.00	1.00	1.00	11
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30

> OUTLINE

> TIMELINE

Figure 1

KNN Sınıflandırma Sonuçları

Phyton kodu örneği

- Örnekte de görebileceğiniz gibi, her veri kusursuz şekilde ayrışamıyor. Bu durumu önlemek için daha gelişmiş ve daha karmaşık yöntemler kullanılması gerekmektedir. Ancak bu slaytta buna değinmeyeceğiz.