

Python ile Makine Öğrenmesi

Bilal Kurban

Klasik Makine
Öğrenmesi

```
graph LR; A[Klasik Makine Öğrenmesi] --> B[Gözetimli (Supervised) Öğrenme]; A --> C[Gözetimsiz (Unsupervised) Öğrenme];
```

Gözetimli
(Supervised)
Öğrenme

Gözetimsiz
(Unsupervised)
Öğrenme

Gözetimli Öğrenme

Görev güdümlü
(bir sonraki
değeri tahmin et)

Önceden
kategorize
edilmiş veriler

Gözetimsiz Öğrenme

Veri güdümlü
(kümeleri
tanımla)

Etiketlenmemiş
veriler

Gözetimli Öğrenme

Regresyon

Sınıflandırma

Gözetimsiz Öğrenme

Kümeleme

Boyut
indirgeme

Gözetimli Öğrenme

Regresyon

- Risk değerlendirme
- Skor tahmini

Sınıflandırma

- Sahtekarlık tespiti
- Spam e-posta tespiti
- Tanılama
- İmaj sınıflandırması

Gözetimsiz Öğrenme

Kümeleme

- Hedef Pazar, Müşteri segmentasyonu
- Şehir Planlama
- Biyoloji

Boyut indirgeme

- Veri madenciliği
- Yüz tanıma

Gözetimli Öğrenme

Regresyon

- Doğrusal

Sınıflandırma

- KNN

Gözetimsiz Öğrenme

Kümeleme

- K-Means

Neden Python



```
graph LR; A[Neden Python] --- B[›Kolay<br/>›Güçlü<br/>›Popüler]; A --- C[›Sınırsız yetenek<br/>›Eşsiz hız]
```

›Kolay
En ›Güçlü
›Popüler

›Sınırsız yetenek
›Eşsiz hız

Gözetimli Öğrenme

Regresyon

- Doğrusal

Sınıflandırma

- KNN

Gözetimsiz Öğrenme

Kümeleme

- K-Means

Python ile Çoklu Doğrusal Regresyon

- › Emlak veri setimiz ([emlak fiyat boyut metro.csv](#))
- › Bu veri setini kullanarak
 - » Emlağın boyutu ve metroya yakınlığı verildiğinde fiyatını tahmin edebilecek bir Çoklu Doğrusal Regresyon modeli oluştur.
 - » Sabit (intercept) ve katsayıları (coefficients) göster
 - » R-kare (R-squared) ve düzeltilmiş R-kareyi (Adjusted R-squared) bul ve karşılaştıır
 - » Modeli kullanarak 120 metrekarelik ve metroya 10dk yakınlıkta bir dairenin fiyatı hakkında bir tahmin yap
 - » İki değişkenin p-değerlerini bul ve yorumla.

İlgili kütüphaneleri içe aktar

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
from sklearn.linear_model import LinearRegression
```

Veriyi yükle

```
data = pd.read_csv('emlak_fiyat_boyut_metro.csv')  
data.head()
```

	fiyat	boyut	metro_yakinlik
0	234314.144	98	15
1	228581.528	100	16
2	281626.336	74	13
3	401255.608	229	5
4	458674.256	194	4

Özelliklerin şekli

```
data.shape
```

```
(100, 3)
```

Tanımlayıcı istatistikler

```
data.describe()
```

	fiyat	boyut	metro_yakinlik
count	100.000000	100.000000	100.000000
mean	292289.470160	129.960000	12.980000
std	77051.727525	45.389876	6.485384
min	154282.128000	73.000000	1.000000
25%	234280.148000	98.000000	8.000000
50%	280590.716000	106.000000	12.500000
75%	335723.696000	157.000000	17.000000
max	500681.128000	281.000000	30.000000

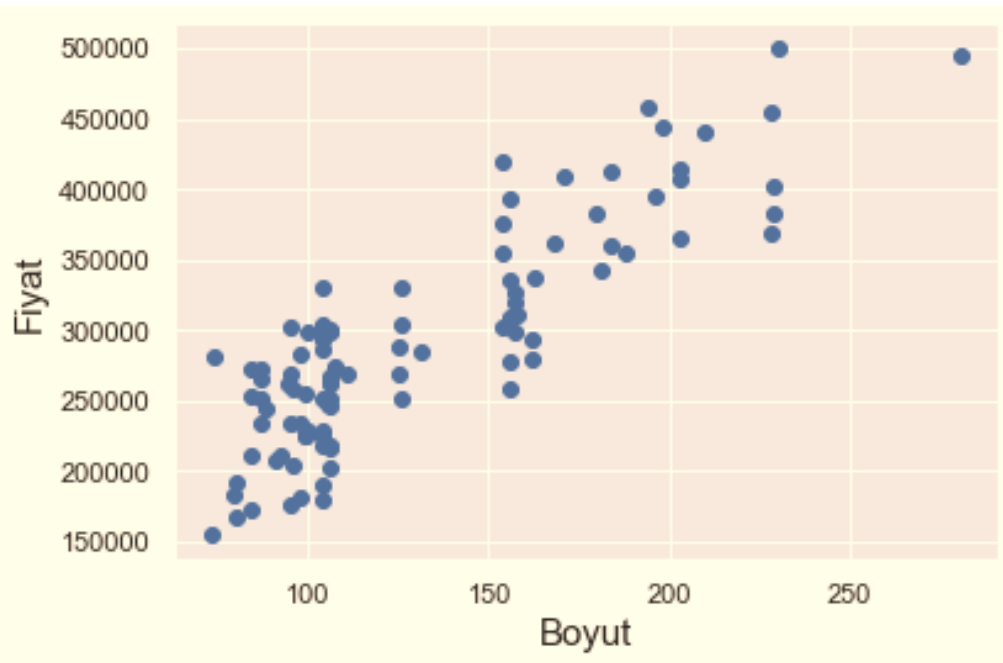
Bağımlı, bağımsız değişkenleri bildir

```
x = data[['boyut','metro_yakinalik']]
```

```
y = data['fiyat']
```

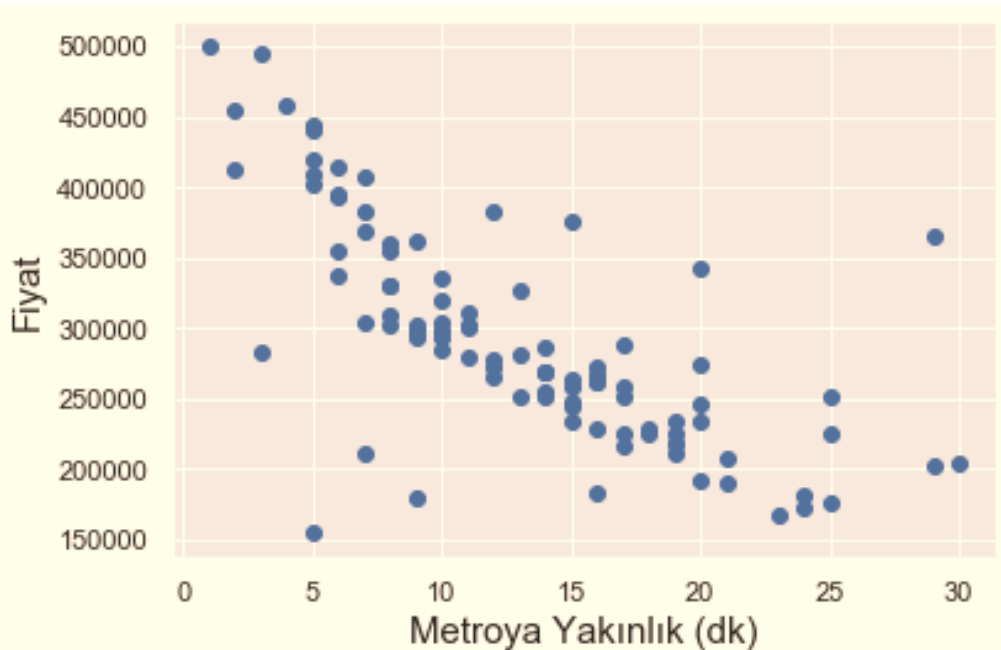
Saçılım Grafiği (Boyut, Fiyat)

```
plt.scatter(x['boyut'],y)  
plt.xlabel('Boyut', fontsize = 15)  
plt.ylabel('Fiyat', fontsize = 15)  
plt.show()
```



Saçılım Grafiği (Metroya yakınlık, Fiyat)

```
plt.scatter(x['metro_yakinalik'],y)
plt.xlabel('Metroya Yakınlık (dk)', fontsize = 15)
plt.ylabel('Fiyat', fontsize = 15)
plt.show()
```



Regresyonu oluşturun

```
reg = LinearRegression()  
reg.fit(x,y)
```

```
LinearRegression(copy_X=True, fit_intercept=True,  
n_jobs=None, normalize=False)
```

Sabit (kesim noktası) bul

reg.intercept_

194304.12416550127

Katsayıları bul

```
reg.coef_
```

```
array([ 1156.04485385, -4025.75063265])
```

R^2 yi hesapla

```
reg.score(x,y)
```

0.8257742530054536

Düzeltilmiş R^2 yi hesapla

```
r2 = reg.score(x,y)
```

```
n = x.shape[0]
```

```
p = x.shape[1]
```

```
duzeltilmis_r2 = 1-(1-r2)*(n-1)/(n-p-1)
```

```
duzeltilmis_r2
```

```
In [4]: data.shape
```

```
Out[4]: (100, 3)
```

$$R_{adj.}^2 = 1 - (1 - R^2) * \frac{n-1}{n-p-1}$$

```
0.8221819695622671
```

Tahmin yapma

(120 metrekarelik ve metroya 10dk
mesafede bir dairenin tahmini fiyatını bul)

```
reg.predict([[120,10]])
```

```
array([292772.00030096])
```

Çoklu tahmin için veri seti

```
yeni_data=pd.DataFrame({'boyut': [100,150,200],  
    'metro_yakinlik': [30,10,5]})
```

yeni_data

	boyut	metro_yakinlik
0	100	30
1	150	10
2	200	5

Çoklu tahmin değerleri

```
reg.predict(yeni_data).round(1)
```

```
array([189136.1, 327453.3, 405384.3])
```


Çoklu tahmini veri setine ekle

```
yeni_data['Tahmini_Fiyat'] = reg.predict(yeni_data)
yeni_data
```

	boyut	metro_yakinlik	Tahmini_Fiyat
0	100	30	189136.090571
1	150	10	327453.345916
2	200	5	405384.341772

Değişkenlerin p değerlerini hesapla

```
from sklearn.feature_selection import f_regression  
f_regression(x,y)
```

```
(array([284.65801738, 96.07823108]),  
array([9.55713521e-31, 3.25674570e-16]))
```

Değişkenlerin p değerlerini hesapla

```
p_values = f_regression(x,y)[1]  
p_values.round(3)
```

```
array([0., 0.])
```

Gözetimli Öğrenme

Regresyon

- Doğrusal

Sınıflandırma

- KNN

Gözetimsiz Öğrenme

Kümeleme

- K-Means

Python ile KNN (K-en yakın komşu)

- › Veri setimiz (iris)
- › Bu veri setini kullanarak
 - » Çiçek ölçülerinden çiçek türünü sınıflandıran bir model geliştir.
 - » Modeli kullanarak sepal uzunluk:4.8, sepal genişlik:2.9, petal uzunluk:1.3, petal genişlik:0.3 olan bir çiçeğin hangi türe ait olduğunu tahmin et.
 - » Modelin başarı oranını bul.

Veriyi yükle

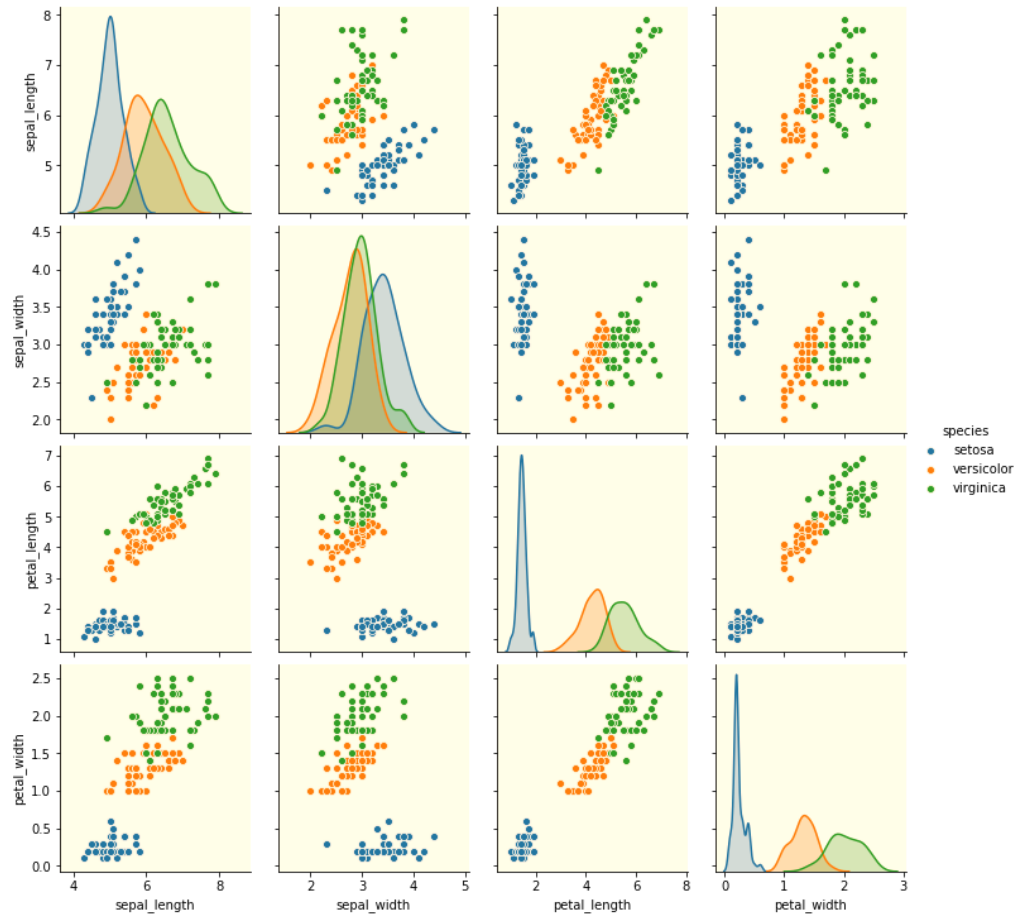
```
import matplotlib.pyplot as plt
import seaborn as sns
iris=sns.load_dataset('iris')
iris.head(5)
```

	sepal_len gth	sepal_wi dth	petal_len gth	petal_wid th	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

İkili Çizimler

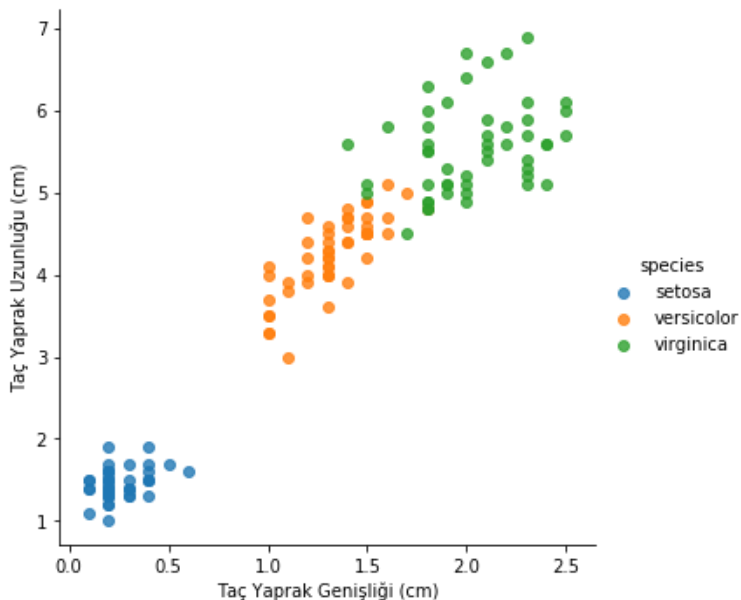
```
sns.pairplot(iris,hue='species')
```

```
plt.show()
```



Taç Yaprak Genişliği ve Uzunluğu

```
sns.Implot(x='petal_width',y='petal_length',data=iris,hue='species',fit_reg=False)  
plt.xlabel('Taç Yaprak Genişliği (cm)')  
plt.ylabel('Taç Yaprak Uzunluğu (cm)')  
plt.show()
```



Modeli oluřtur

```
from sklearn.neighbors import KNeighborsClassifier  
en_yakin_k=KNeighborsClassifier(n_neighbors=5)  
en_yakin_k.fit(iris[['sepal_length','sepal_width','petal_length','petal_width']],iris['species'])
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30,  
metric='minkowski', metric_params=None,  
n_jobs=1, n_neighbors=5, p=2, weights='uniform')
```

Tahmin

```
import pandas as pd
import numpy as np
cicek=np.array([[4.8,2.9,1.3,0.3]])
tahmin=en_yakin_k.predict(cicek)
tahmin_olasilik=en_yakin_k.predict_proba(cicek)
print('Çiçek Türü Tahmini:',tahmin)
print('Tahmin Olasılığı:',tahmin_olasilik)
```

Çiçek Türü Tahmini: ['setosa'] Tahmin Olasılığı: [[1. 0. 0.]]

Başarı oranı

```
x=iris[['sepal_length','sepal_width','petal_length','petal_width']]
y=iris[['species']]
data=y.copy()
tahmin1=en_yakin_k.predict(x)
data['Tahmin y']=tahmin1
data['Aynımı']=data['species']==data['Tahmin y']
print('Başarı Sayısı',data.Aynımı.value_counts())
print('Başarı Oranı',data.Aynımı.value_counts('%'))
```

```
Başarı Sayısı True 145 False 5 Name: Aynımı, dtype:
int64 Başarı Oranı True 0.966667 False 0.033333
Name: Aynımı, dtype: float64
```

Gözetimli Öğrenme

Regresyon

- Doğrusal

Sınıflandırma

- KNN

Gözetimsiz Öğrenme

Kümeleme

- K-Means

Python ile K-means kümeleme

- › Market veri setimiz
([market tatmin sadakat.csv](#))
- › Bu veri setini kullanarak
 - › Verilerimizi çizelim
 - › Müşterileri önce iki kümeye ayıralım.
 - › Değişkenleri standartlaştıralım
 - › K'nin seçimi
 - › Dört küme

İlgili kütüphaneleri içe aktar

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
from sklearn.cluster import KMeans
```

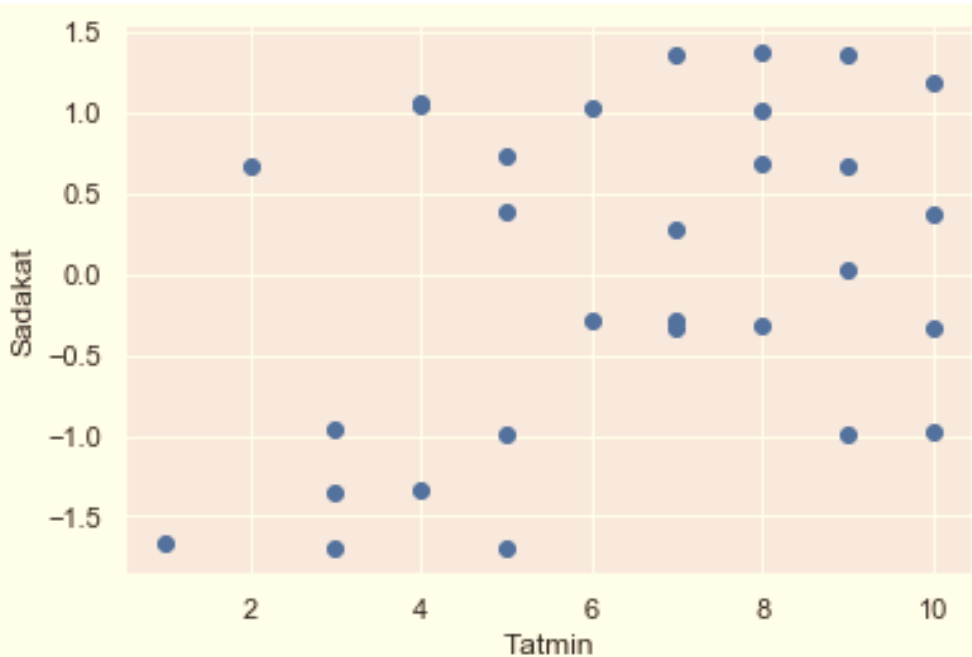
Veriyi yükle

```
data = pd.read_csv('market_tatmin_sadakat.csv')  
data.head()
```

	tatmin	sadakat
0	4	-1.33
1	6	-0.28
2	5	-0.99
3	7	-0.29
4	4	1.06

Verileri çiz

```
plt.scatter(data['tatmin'],data['sadakatk'])  
plt.xlabel('Tatmin')  
plt.ylabel('Sadakat')
```



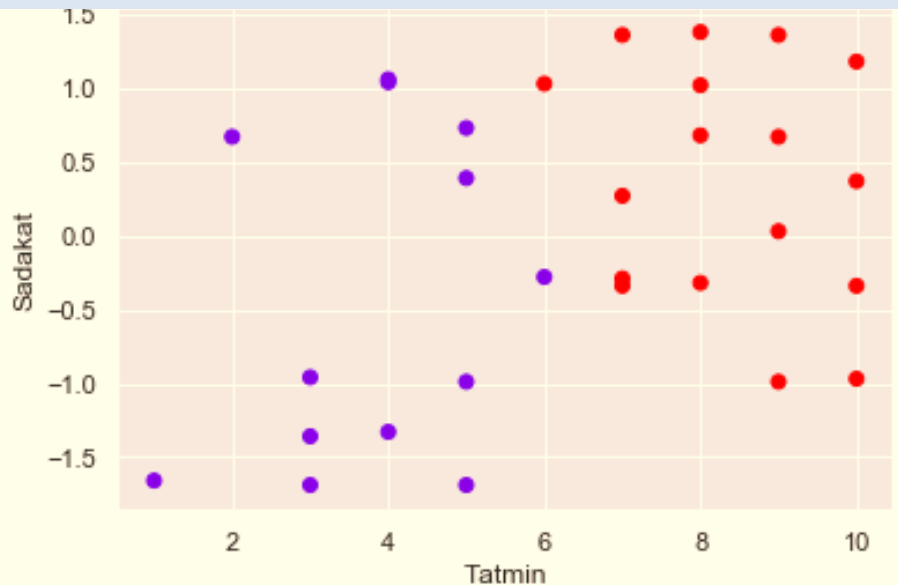
Kümeleme

```
x = data.copy()  
kmeans = KMeans(2)  
kmeans.fit(x)
```

```
KMeans(algorithm='auto', copy=True, init='k-  
means++', max_iter=300, n_clusters=2, n_init=10,  
n_jobs=None, precompute_distances='auto',  
random_state=None, tol=0.0001, verbose=0)
```

Kümeleme Sonuçları

```
clusters = x.copy()  
clusters['kume_tahmin']=kmeans.fit_predict(x)  
plt.scatter(clusters['tatmin'],clusters['sadakats'],c=clusters['kume_tahmin'],cmap='rainbow')  
plt.xlabel('Tatmin')  
plt.ylabel('Sadakat')
```



Değişkenleri standartlaştır

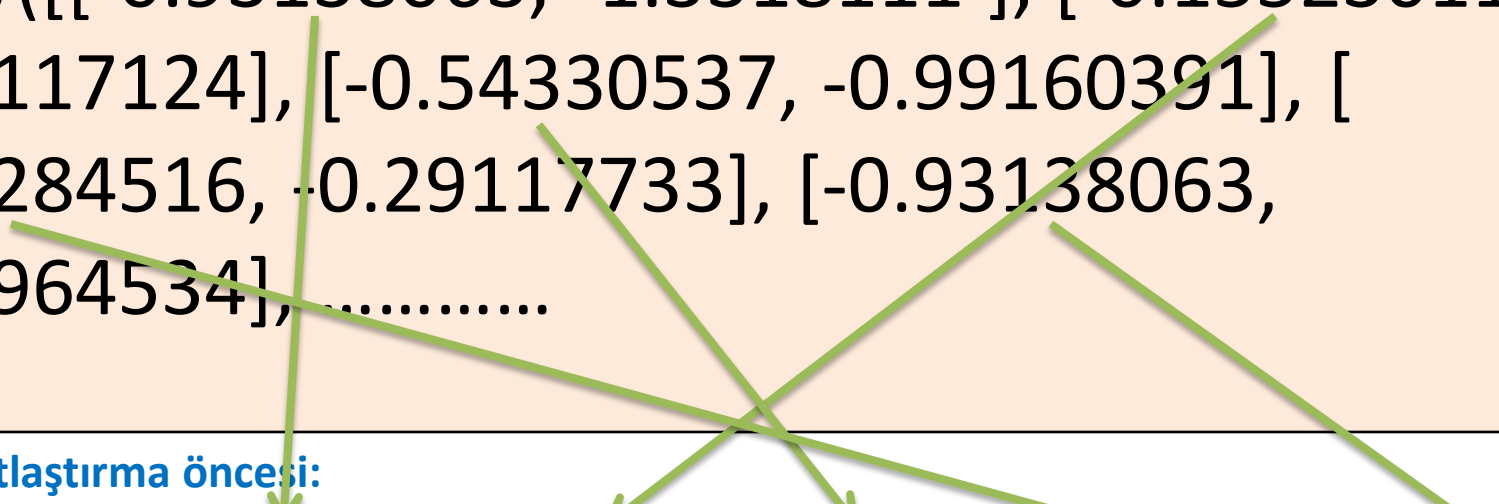
```
from sklearn import preprocessing
```

```
x_scaled = preprocessing.scale(x)
```

```
x_scaled
```

```
array([[ -0.93138063, -1.33181111 ], [-0.15523011, -  
0.28117124], [-0.54330537, -0.99160391], [  
0.23284516, -0.29117733], [-0.93138063,  
1.05964534], .....]
```

Standartlaştırma öncesi:



tatmin	4	6	5	7	4
sadakat	-1.33	-0.28	-0.99	-0.29	1.06

K'yi Seçmek için Dirsek yöntemi

```
kikt = []  
for i in range(1,10):  
    kmeans = KMeans(i)  
    kmeans.fit(x_scaled)  
    kikt.append(kmeans.inertia_)  
kikt
```

```
[60.0,  
29.818973034723147,  
17.913349527387965,  
10.247181805928422,  
7.792695153937187,  
6.571285077136385,  
5.326631124753926,  
4.380320178840311,  
3.9293801253331493]
```

Kümeleme çözümlerini keşfedin

```
kmeans_new = KMeans(4)
kmeans_new.fit(x_scaled)
clusters_new = x.copy()
clusters_new['kume_tahmin'] =
kmeans_new.fit_predict(x_scaled)
```

Kümeleme çözümlerini keşfedin

```
clusters_new.head(8)
```

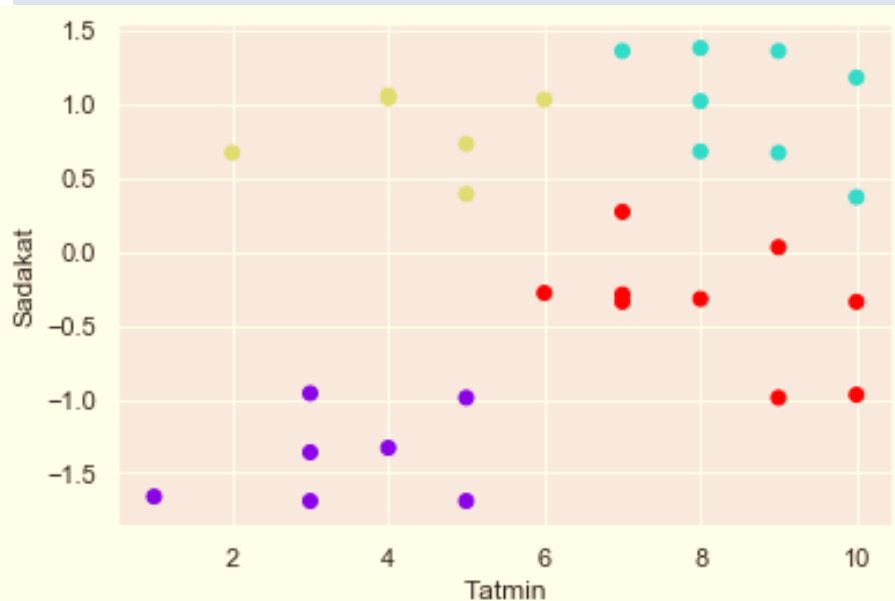
	tatmin	sadakat	kume_tahmin
0	4	-1.33	0
1	6	-0.28	3
2	5	-0.99	0
3	7	-0.29	3
4	4	1.06	2
5	1	-1.66	0
6	10	-0.97	3
7	8	-0.32	3

4'lü kümemizin çizimi

```
plt.scatter(clusters_new['tatmin'],clusters_new['sadakat'],c=clusters_new['kume_tahmin'],cmap='rainbow')
```

```
plt.xlabel('Tatmin')
```

```
plt.ylabel('Sadakat')
```



Market segmentasyonumuz (etiketli)

