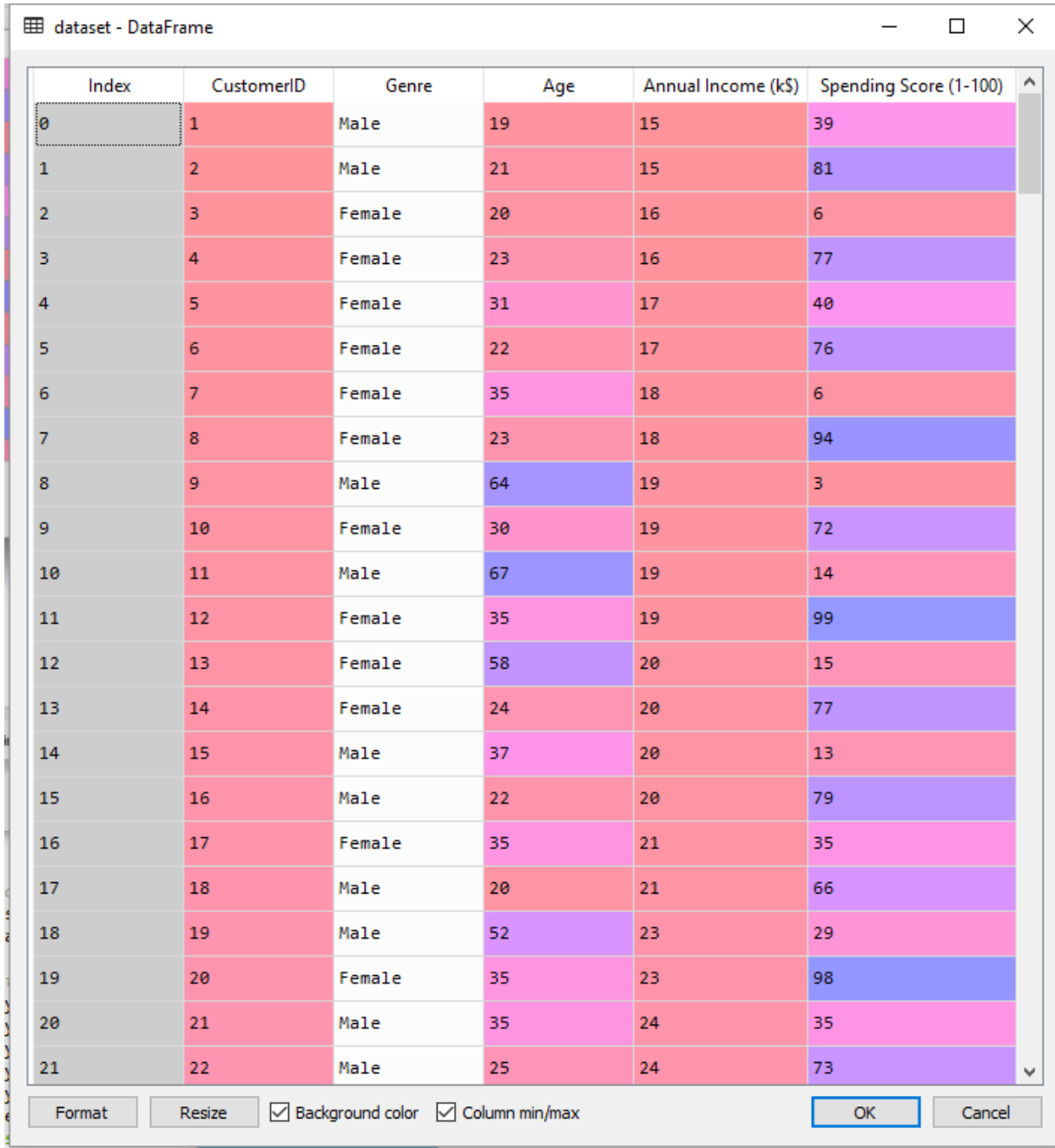


```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import os

os.chdir('Calisma_Dizniniz')
dataset = pd.read_csv('Mall_Customers.csv')

Spyder variable explorer ekranından veri setimizi görelim.
```



Index	CustomerID	Genre	Age	Annual Income (k\$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40
5	6	Female	22	17	76
6	7	Female	35	18	6
7	8	Female	23	18	94
8	9	Male	64	19	3
9	10	Female	30	19	72
10	11	Male	67	19	14
11	12	Female	35	19	99
12	13	Female	58	20	15
13	14	Female	24	20	77
14	15	Male	37	20	13
15	16	Male	22	20	79
16	17	Female	35	21	35
17	18	Male	20	21	66
18	19	Male	52	23	29
19	20	Female	35	23	98
20	21	Male	35	24	35
21	22	Male	25	24	73

Veriyi Anlamak

Yukarıda görülen veri seti bir markete ait olsun. Market, müşterilerine dağıttığı üyelik kartları ile müşteri bilgileri ile satın alma bilgilerini kaydetmiş olsun. Niteliklerimiz sırasıyla şöyle: Müşteri Numarası (CustomerID), Cinsiyet (Gender), Yaş (Age), Yıllık Gelir (Annual Income) ve Harcama Skoru (Spending Score). Harcama Skoru müşterilerin geçmiş alış-veriş kayıtlarına dayanarak market tarafından 1 ile 100 arasında belirlenmiş bir puandır. Puan 1'e yaklaşması müşterinin daha az harcama yapan bir müşteri olduğunu gösterir. Market elindeki müşterileri segmentlere (kümelere) ayırmak ister. Kim bilir belki de her segmentteki müşteriyi ayrı ele alacak ve ona göre satış arttırma politikaları üretecektir. Kaç segment oluşacağı belli değildir.

Nitelikleri Seçmek

Yukarıda gördüğümüz niteliklerden bağımsız değişken olarak sadece yıllık geliri ve harcama skorunu kullanacağız.

```
X = dataset.iloc[:,[3,4]].values
```

Elimizde set var şimdi bunu K-Ortalamalar ile kümelemeye tabi tutacağız ancak algoritma bizden küme sayısı isteyecek. Bir [önceki yazımızda](#) bahsettiğimiz gibi küme sayısını bulmak için dirsek yönteminden (elbow method) yardım alacağız. Bunun için önce WCSS'i hesaplayıp küme sayısı ile birlikte WCSS deki değişimin çizgi grafiğini çizmeliyiz. Çizmeden rakamlara bakarak da bir karar verebiliriz ancak yine de çizelim. Öncelikle scikit-learn kütüphanesinden KMeans sınıfını indirelim

```
from sklearn.cluster import KMeans
```

Boş bir liste oluşturalım. Bu listeye for döngüsünde her bir küme sayısı için WCSS değerlerini ekleyeceğiz. Küme sayısı için range() fonksiyonu ile 1'den 10'a kadar birer artan bir liste oluşturalım.

```
wcss = []
```

```
kume_sayisi_listesi = range(1, 11)
```

```
for i in kume_sayisi_listesi :
```

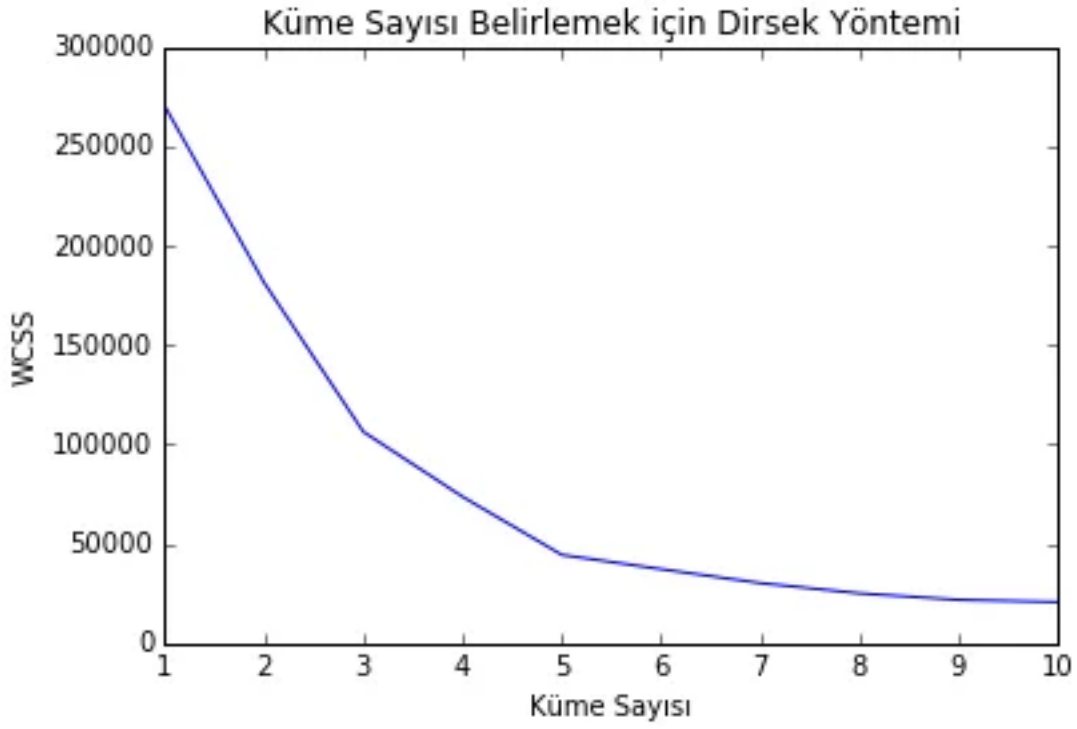
```
kmeans = KMeans(n_clusters = i, init = 'k-means++', max_iter = 300, n_init = 10,  
random_state = 0)  
kmeans.fit(X)  
wcss.append(kmeans.inertia_)
```

For döngüsü içinde yer alan kodlar neler yapıyor biraz konuşalım. Öncelikle import ettiğimiz Kmeans sınıfından kmeans adında bir nesne oluşturuyoruz. Nesne oluştururken yapıcı fonksiyona (\_\_init\_\_) bazı parametreler gönderiyoruz. Bunlardan ilki küme sayısı olan n\_clusters. for döngüsü i değişkeniyle her dönüşünde bir artarak küme sayısını parametre olarak n\_clusters'a veriyor. init parametresi ise başlangıç noktalarını seçmek için ideal küme merkezlerini belirliyor. Hatırlarsanız rastgele başlangıç noktası tuzağından (random initialization trap) bahsetmiştik. kmeans++ parametresi bizi bu tuzaktan kurtaracak iyi başlangıç noktaları seçmemizi sağlıyor. Bir sonraki parametre max\_iter algoritmanın nihai durumuna erişmesi için en fazla kaç iterasyon yapabileceğini belirler, varsayılan 300'tür. n\_init ise küme merkezi başlangıç noktasının kaç farklı noktadan başlayabileceğini belirler. Son parametre random\_state, bu işlemleri uygulayan herkesin aynı sonuçları elde etmesini sağlar. Nesne oluştuktan sonra fit() metodu ile nesne ile veri uyumunu gerçekleştiririz. Parametre olarak daha önce oluşturduğumuz X'i veriyoruz. for döngüsünden önce oluşturduğumuz wcss listesine kmeans nesnesinin inertia\_ özelliğini ekliyoruz.

Dirsek Metodu Grafiği

Küme sayısını belirlemek için dirsek metodu grafiğimizi çizelim.

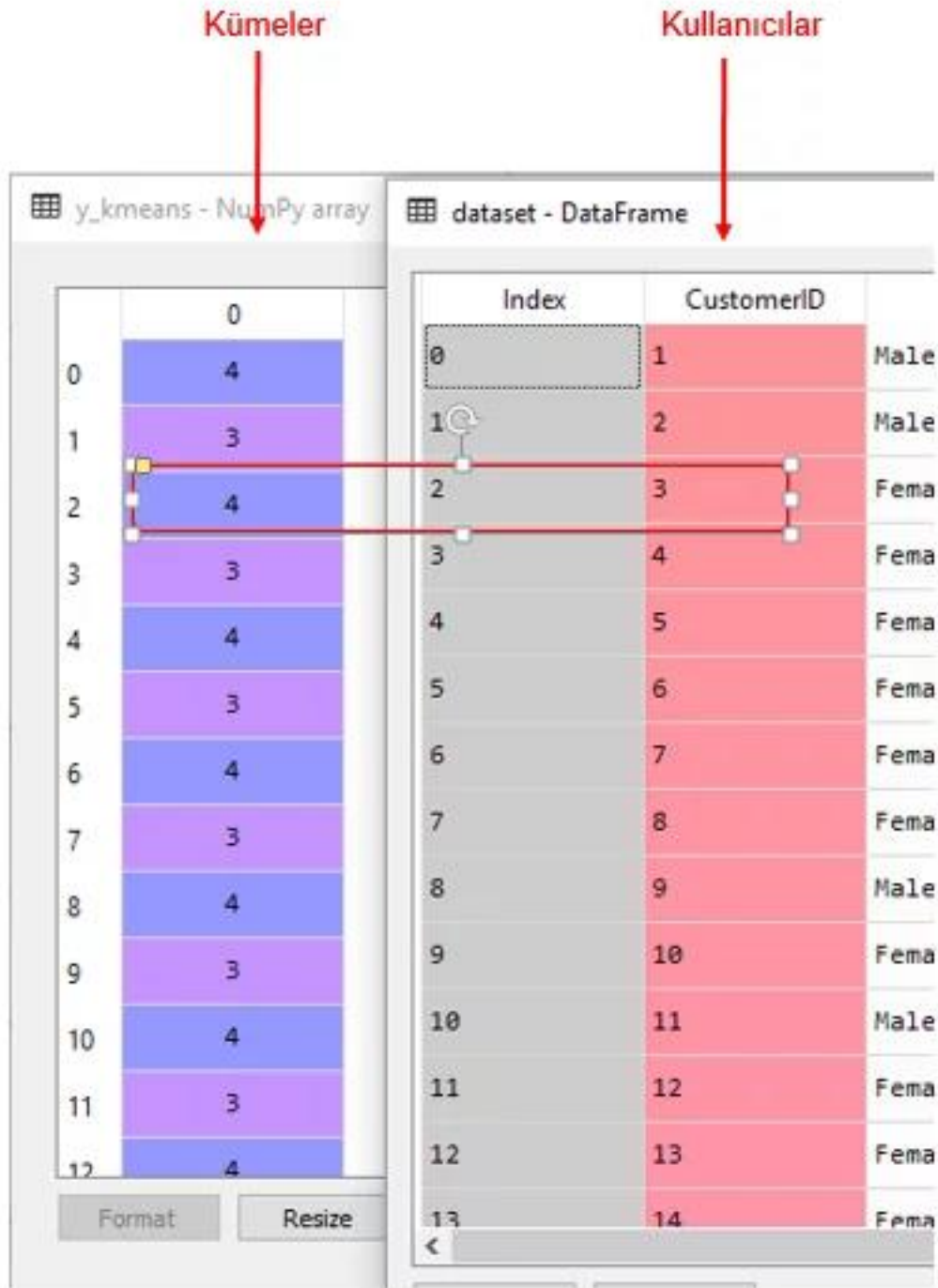
```
plt.plot(kume_sayisi_listesi, wcss)  
plt.title('Küme Sayısı Belirlemek için Dirsek Yöntemi')  
plt.xlabel('Küme Sayısı')  
plt.ylabel('WCSS')  
plt.show()
```



Grafikten ideal küme sayısının 5 olacağını görebiliriz. Şimdi for döngüsündeki küme sayısı 5 için çalışan satırı tekrarlayalım.

Belirlenen küme sayısına göre kümeleme yapmak

```
kmeans = KMeans(n_clusters = 5, init = 'k-means++', max_iter = 300, n_init = 10,  
random_state = 0)  
y_kmeans = kmeans.fit_predict(X)
```

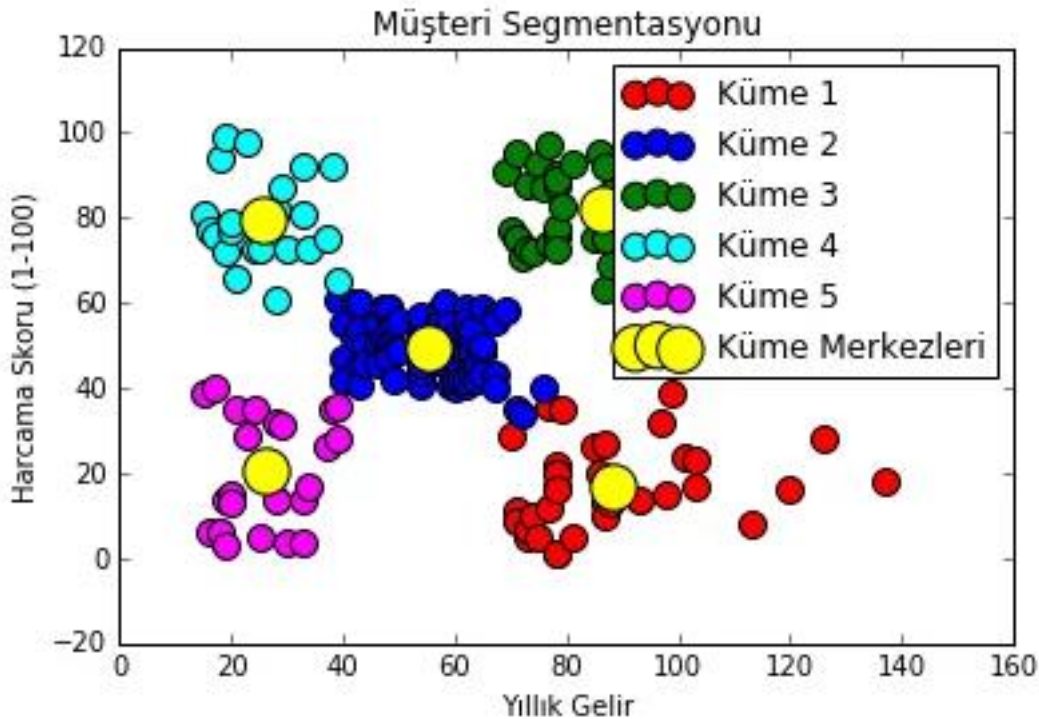


Yukarıdaki kodlarla toplam 200 ayrı kullanıcıyı 5 farklı kümeye yerleştirdik. Aşağıdaki resimde eşleşmenin belli bir kısmını görebiliyoruz.

Kümeleri grafikte göstermek

```
plt.scatter(X[y_kmeans == 0, 0], X[y_kmeans == 0, 1], s = 100, c = 'red', label = 'Küme 1')
```

```
plt.scatter(X[y_kmeans == 1, 0], X[y_kmeans == 1, 1], s = 100, c = 'blue', label =  
'Küme 2')  
plt.scatter(X[y_kmeans == 2, 0], X[y_kmeans == 2, 1], s = 100, c = 'green', label =  
'Küme 3')  
plt.scatter(X[y_kmeans == 3, 0], X[y_kmeans == 3, 1], s = 100, c = 'cyan', label =  
'Küme 4')  
plt.scatter(X[y_kmeans == 4, 0], X[y_kmeans == 4, 1], s = 100, c = 'magenta', label =  
'Küme 5')  
plt.scatter(kmeans.cluster_centers_[0], kmeans.cluster_centers_[1], s = 300, c =  
'yellow', label = 'Küme Merkezleri')  
plt.title('Müşteri Segmentasyonu')  
plt.xlabel('Yıllık Gelir')  
plt.ylabel('Harcama Skoru (1-100)')  
plt.legend()  
plt.show()
```



Grafiğimizi oluşturduk. Şimdi daha iyi görebiliyoruz. Kümelerimizi tek tek inceleyelim. Küme 1'de yer alan müşteriler (kırmızı noktalar) Yıllık Geliri yüksek ancak harcama skoru düşük müşteriler. Market sahibi bu

müşterilerin daha fazla harcamasını sağlayacak tedbirler düşünebilir. Küme 2'deki müşteriler (mavi noktalar) ortalama gelir ve ortalama harcama skoruna sahipler ve birbirine çok benzeşiyor. Muhtemelen küme içi noktaların merkeze uzaklığının kareler ortalaması (wcsc) bu kümede en yüksektir. Küme 3 (yeşil noktalar) yüksek gelire birlikte yüksek harcama skoruna sahip müşteriler. Her market işletmesinin sahip olmak isteyeceği müşteri segmetidir. Market bu müşterileri elinde tutmak ve bu kümeye müşteri eklemek için gerekli politikaları üretip uygulamaya koyabilir. Küme 4'teki müşteriler (turkuaz mavi) düşük gelire sahip olmasına rağmen yüksek harcama skoruna sahip. Bu müşteriler muhtemelen kredi kartı batağında olan dikkatsiz müşterilerdir. Küme 5 (pembe) düşük gelire sahip ve harcama skoru düşük müşteriler. Bu müşterilere dikkatli ve tutumlu müşteriler olarak adlandırabiliriz.