flo

June 18, 2024

[27]:
```
# Abdurrahman Bulut
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import scipy.cluster.hierarchy as sch
from sklearn.cluster import AgglomerativeClustering
from sklearn.metrics import silhouette_score
```

Adım 1

[2]:
```
df = pd.read_csv("flo_data_20K.csv")
```

[3]:
```
df.head()
```

[3]:
```
                              master_id order_channel last_order_channel  \
0  cc294636-19f0-11eb-8d74-000d3a38a36f   Android App            Offline
1  f431bd5a-ab7b-11e9-a2fc-000d3a38a36f   Android App             Mobile
2  69b69676-1a40-11ea-941b-000d3a38a36f   Android App        Android App
3  1854e56c-491f-11eb-806e-000d3a38a36f   Android App        Android App
4  d6ea1074-f1f5-11e9-9346-000d3a38a36f       Desktop            Desktop

  first_order_date last_order_date last_order_date_online  \
0       2020-10-30      2021-02-26             2021-02-21
1       2017-02-08      2021-02-16             2021-02-16
2       2019-11-27      2020-11-27             2020-11-27
3       2021-01-06      2021-01-17             2021-01-17
4       2019-08-03      2021-03-07             2021-03-07

  last_order_date_offline  order_num_total_ever_online  \
0              2021-02-26                         4.0
1              2020-01-10                        19.0
2              2019-12-01                         3.0
3              2021-01-06                         1.0
4              2019-08-03                         1.0

   order_num_total_ever_offline  customer_value_total_ever_offline  \
```

```
0                           1.0                            139.99
1                           2.0                            159.97
2                           2.0                            189.97
3                           1.0                             39.99
4                           1.0                             49.99

   customer_value_total_ever_online       interested_in_categories_12
0                            799.38                            [KADIN]
1                           1853.58  [ERKEK, COCUK, KADIN, AKTIFSPOR]
2                            395.35                  [ERKEK, KADIN]
3                             81.98              [AKTIFCOCUK, COCUK]
4                            159.99                      [AKTIFSPOR]
```

[4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19945 entries, 0 to 19944
Data columns (total 12 columns):
 #   Column                             Non-Null Count  Dtype
---  ------                             --------------  -----
 0   master_id                          19945 non-null  object
 1   order_channel                      19945 non-null  object
 2   last_order_channel                 19945 non-null  object
 3   first_order_date                   19945 non-null  object
 4   last_order_date                    19945 non-null  object
 5   last_order_date_online             19945 non-null  object
 6   last_order_date_offline            19945 non-null  object
 7   order_num_total_ever_online        19945 non-null  float64
 8   order_num_total_ever_offline       19945 non-null  float64
 9   customer_value_total_ever_offline  19945 non-null  float64
 10  customer_value_total_ever_online   19945 non-null  float64
 11  interested_in_categories_12        19945 non-null  object
dtypes: float64(4), object(8)
memory usage: 1.8+ MB
```

Adım 2

[5]:
```
# Tarih değişkenlerini datetime formatına çevirelim
date_columns = ['first_order_date', 'last_order_date',␣
 ↪'last_order_date_online', 'last_order_date_offline']
df[date_columns] = df[date_columns].apply(pd.to_datetime)
```

[7]:
```
# Tenure ve Recency değişkenlerini oluşturalım
# Referans tarih olarak veri setindeki en güncel tarih seçilebilir
df['today_date'] = pd.to_datetime('2024-06-18')
```

[8]:
```
# Müşterinin yaşı
df['tenure'] = (df['today_date'] - df['first_order_date']).dt.days
```

[9]:
```
# En son alışveriş yaptığı gün sayısı
df['recency'] = (df['today_date'] - df['last_order_date']).dt.days
```

[10]:
```
# Toplam alışveriş sayısı ve toplam harcama miktarı
df['order_num_total'] = df['order_num_total_ever_online'] +␣
 ↪df['order_num_total_ever_offline']
df['customer_value_total'] = df['customer_value_total_ever_online'] +␣
 ↪df['customer_value_total_ever_offline']
```

[12]:
```
# Kullanılacak değişkenler
selected_columns = ['order_num_total', 'customer_value_total', 'recency',␣
 ↪'tenure']
df_selected = df[selected_columns]

# Seçilen değişkenlerin genel istatistiklerine bakalım
df_selected.describe()
```

[12]:

	order_num_total	customer_value_total	recency	tenure
count	19945.000000	19945.000000	19945.000000	19945.000000
mean	5.024768	751.244287	1247.458360	1914.302833
std	4.742707	895.402173	103.281149	523.396883
min	2.000000	44.980000	1115.000000	1118.000000
25%	3.000000	339.980000	1156.000000	1630.000000
50%	4.000000	545.270000	1224.000000	1764.000000
75%	6.000000	897.780000	1315.000000	1949.000000
max	202.000000	45905.100000	1480.000000	4173.000000

1 Task 2

Adım 1

[14]:
```
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df_selected)
```

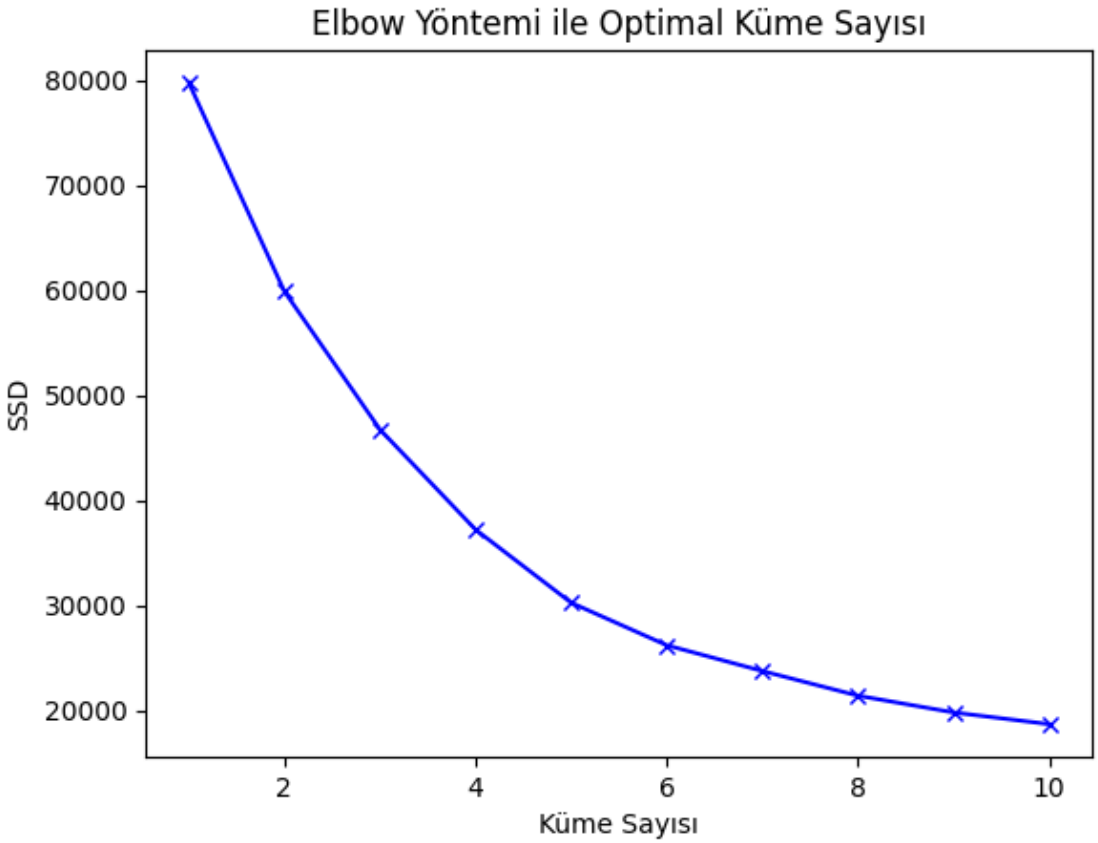
Adım 2

[16]:
```
# Elbow yöntemini kullanarak en uygun küme sayısını bulma
ssd = []
K = range(1, 11)
for k in K:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(df_scaled)
    ssd.append(kmeans.inertia_)
```

```
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
```

```
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

[17]:
```
# Elbow grafiği
plt.plot(K, ssd, 'bx-')
plt.xlabel('Küme Sayısı')
plt.ylabel('SSD')
plt.title('Elbow Yöntemi ile Optimal Küme Sayısı')
plt.show()
```



[18]:
```
# küme sayısı belirleme
optimal_k = 4
```

Adım 3

[19]:
```
# KMeans algoritması uygulandı
kmeans = KMeans(n_clusters=optimal_k, random_state=42)
kmeans.fit(df_scaled)
df['segment'] = kmeans.labels_

print(df.groupby('segment').agg({
    'order_num_total': ['mean', 'sum'],
    'customer_value_total': ['mean', 'sum'],
```

```
    'recency': ['mean'],
    'tenure': ['mean']
}))
print(df['segment'].value_counts())
```

```
        order_num_total          customer_value_total              \
                   mean      sum                 mean         sum
segment
0              3.676700  23086.0           539.973075  3390490.94
1              4.390337  46981.0           675.832411  7232082.63
2             22.303030  13984.0          3633.743652  2278357.27
3              6.915312  16168.0           890.776933  2082636.47

             recency       tenure
                mean         mean
segment
0        1363.910655  1795.264373
1        1178.731614  1707.255677
2        1203.736842  2491.848485
3        1260.997434  3026.763473
```

```
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

Adım 4

[20]:
```
# Her bir segmentin istatistiksel özellikleri
segment_summary = df.groupby('segment').agg({
    'order_num_total': ['mean', 'sum'],
    'customer_value_total': ['mean', 'sum'],
    'recency': ['mean'],
    'tenure': ['mean']
})
print(segment_summary)
```

```
        order_num_total          customer_value_total              \
                   mean      sum                 mean         sum
segment
0              3.676700  23086.0           539.973075  3390490.94
1              4.390337  46981.0           675.832411  7232082.63
2             22.303030  13984.0          3633.743652  2278357.27
3              6.915312  16168.0           890.776933  2082636.47

             recency       tenure
                mean         mean
segment
```

```
0        1363.910655  1795.264373
1        1178.731614  1707.255677
2        1203.736842  2491.848485
3        1260.997434  3026.763473
```

[24]:
```
# Segmentlerin istatistiksel özetini daha anlaşılır kılmak için sütun␣
 ↪isimlerini düzenleme
segment_summary.columns = ['_'.join(col) for col in segment_summary.columns]
segment_summary = segment_summary.reset_index()


print(segment_summary)
```

```
   segment  order_num_total_mean  order_num_total_sum  \
0        0              3.676700              23086.0
1        1              4.390337              46981.0
2        2             22.303030              13984.0
3        3              6.915312              16168.0

   customer_value_total_mean  customer_value_total_sum  recency_mean  \
0                  539.973075                3390490.94   1363.910655
1                  675.832411                7232082.63   1178.731614
2                 3633.743652                2278357.27   1203.736842
3                  890.776933                2082636.47   1260.997434

   tenure_mean
0  1795.264373
1  1707.255677
2  2491.848485
3  3026.763473
```

2 Task 3

Adım 1

[28]:
```
# Farklı küme sayıları için Silhouette skorlarını hesaplayalım
silhouette_scores = []
K = range(2, 11)
```

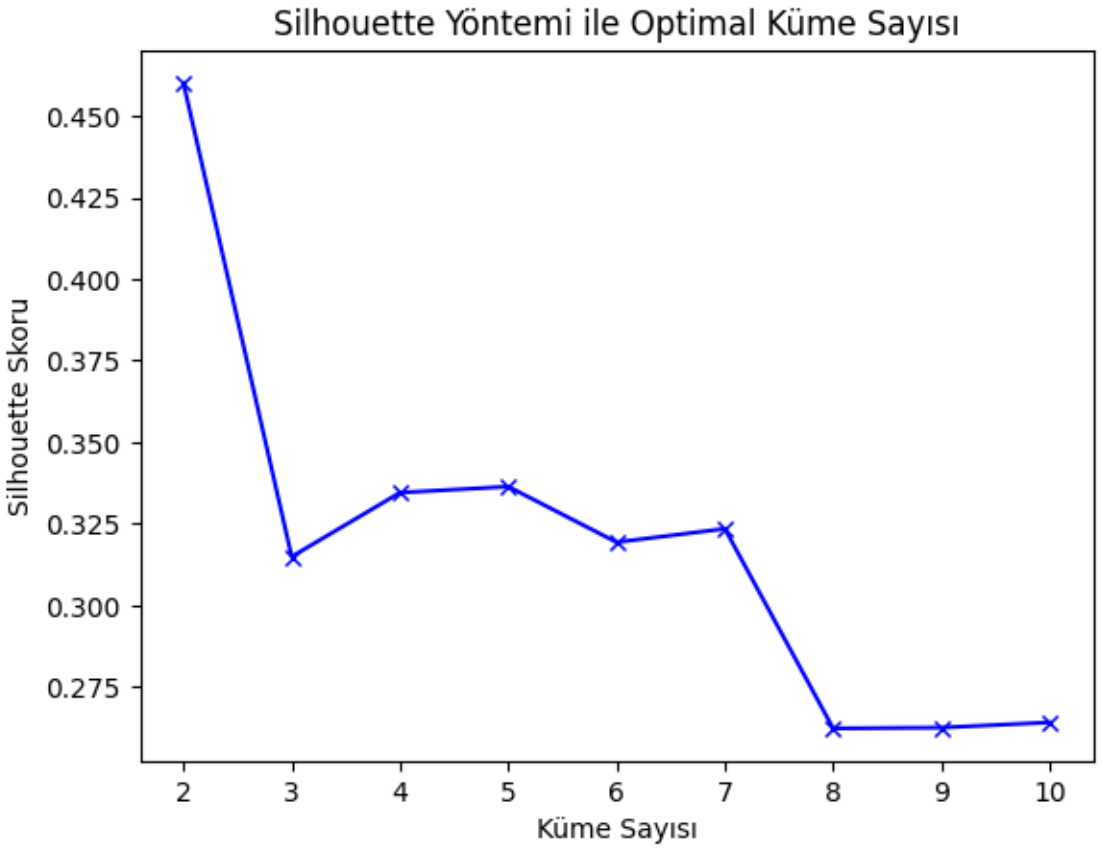
[29]:
```
for k in K:
    hc = AgglomerativeClustering(n_clusters=k, affinity='euclidean',␣
 ↪linkage='ward')
    cluster_labels = hc.fit_predict(df_scaled)
    silhouette_avg = silhouette_score(df_scaled, cluster_labels)
    silhouette_scores.append(silhouette_avg)
```

```
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_agglomerative.py:1005: FutureWarning: Attribute
```

```
`affinity` was deprecated in version 1.2 and will be removed in 1.4. Use
`metric` instead
  warnings.warn(
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_agglomerative.py:1005: FutureWarning: Attribute
`affinity` was deprecated in version 1.2 and will be removed in 1.4. Use
`metric` instead
  warnings.warn(
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_agglomerative.py:1005: FutureWarning: Attribute
`affinity` was deprecated in version 1.2 and will be removed in 1.4. Use
`metric` instead
  warnings.warn(
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_agglomerative.py:1005: FutureWarning: Attribute
`affinity` was deprecated in version 1.2 and will be removed in 1.4. Use
`metric` instead
  warnings.warn(
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_agglomerative.py:1005: FutureWarning: Attribute
`affinity` was deprecated in version 1.2 and will be removed in 1.4. Use
`metric` instead
  warnings.warn(
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_agglomerative.py:1005: FutureWarning: Attribute
`affinity` was deprecated in version 1.2 and will be removed in 1.4. Use
`metric` instead
  warnings.warn(
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_agglomerative.py:1005: FutureWarning: Attribute
`affinity` was deprecated in version 1.2 and will be removed in 1.4. Use
`metric` instead
  warnings.warn(
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_agglomerative.py:1005: FutureWarning: Attribute
`affinity` was deprecated in version 1.2 and will be removed in 1.4. Use
`metric` instead
  warnings.warn(
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_agglomerative.py:1005: FutureWarning: Attribute
`affinity` was deprecated in version 1.2 and will be removed in 1.4. Use
`metric` instead
  warnings.warn(
```

[30]:
```
# Silhouette skorlarını görselleştirelim
plt.plot(K, silhouette_scores, 'bx-')
plt.xlabel('Küme Sayısı')
```

```
plt.ylabel('Silhouette Skoru')
plt.title('Silhouette Yöntemi ile Optimal Küme Sayısı')
plt.show()
```



[31]:
```
# En yüksek Silhouette skoruna sahip küme sayısını bulalım
optimal_k = K[np.argmax(silhouette_scores)]
print(f"Optimal Küme Sayısı: {optimal_k}")
```

```
Optimal Küme Sayısı: 2
```

Adım 2

[32]:
```
optimal_k = 4

# AgglomerativeClustering modelini uygulayalım
hc = AgglomerativeClustering(n_clusters=optimal_k, affinity='euclidean',␣
 ↪linkage='ward')
df['segment_hc'] = hc.fit_predict(df_scaled)
```

```
c:\Users\abdur\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\cluster\_agglomerative.py:1005: FutureWarning: Attribute
```

```
`affinity` was deprecated in version 1.2 and will be removed in 1.4. Use
`metric` instead
  warnings.warn(
```

[33]:
```
# Segment sayısını ve her bir segmentteki müşteri sayısını görelim
print(df['segment_hc'].value_counts())
```

```
segment_hc
1    10783
2     6213
3     1898
0     1051
Name: count, dtype: int64
```

Adım 3

[34]:
```
# Her bir segmentin istatistiksel özelliklerini inceleyelim
segment_summary_hc = df.groupby('segment_hc').agg({
    'order_num_total': ['mean', 'sum'],
    'customer_value_total': ['mean', 'sum'],
    'recency': ['mean'],
    'tenure': ['mean']
})

# Segmentlerin istatistiksel özetini yazdıralım
print(segment_summary_hc)
```

```
           order_num_total           customer_value_total               \
                      mean      sum                 mean         sum
segment_hc
0                 18.256898  19188.0          2836.926147  2981609.38
1                  4.559492  49165.0           707.004831  7623633.09
2                  3.465958  21534.0           496.296469  3083489.96
3                  5.443625  10332.0           682.210158  1294834.88

                recency       tenure
                   mean         mean
segment_hc
0           1193.386299  2595.109420
1           1185.726143  1708.514606
2           1363.535007  1819.827459
3           1248.145416  3015.705479
```

[35]:
```
# Segmentlerin istatistiksel özetini daha anlaşılır kılmak için sütun␣
 ↪isimlerini düzenleyelim
segment_summary_hc.columns = ['_'.join(col) for col in segment_summary_hc.
 ↪columns]
segment_summary_hc = segment_summary_hc.reset_index()
```

```python
# Segment özetini görüntüleyelim
print(segment_summary_hc)
```

```
   segment_hc  order_num_total_mean  order_num_total_sum  \
0           0             18.256898              19188.0
1           1              4.559492              49165.0
2           2              3.465958              21534.0
3           3              5.443625              10332.0

   customer_value_total_mean  customer_value_total_sum  recency_mean  \
0                2836.926147                2981609.38   1193.386299
1                 707.004831                7623633.09   1185.726143
2                 496.296469                3083489.96   1363.535007
3                 682.210158                1294834.88   1248.145416

   tenure_mean
0  2595.109420
1  1708.514606
2  1819.827459
3  3015.705479
```