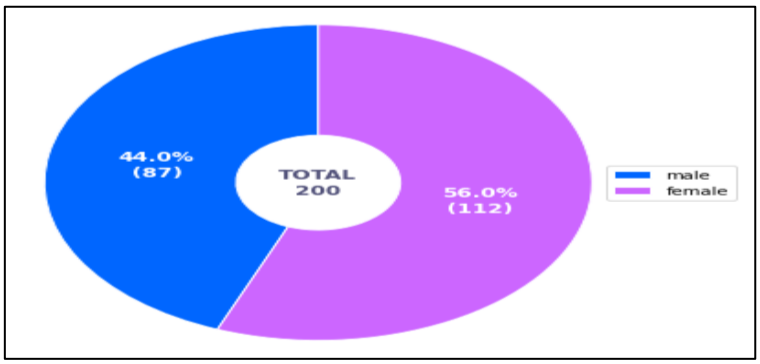
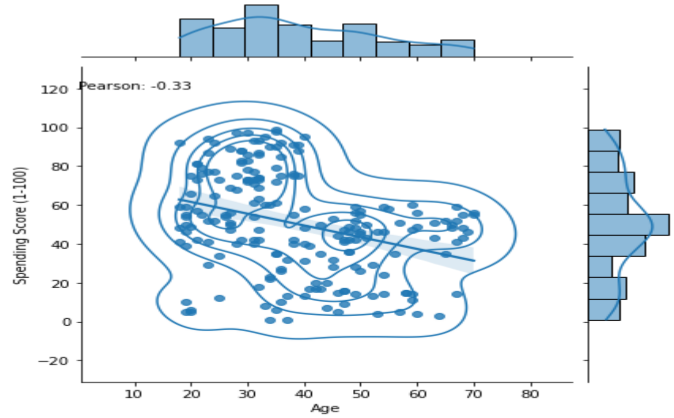
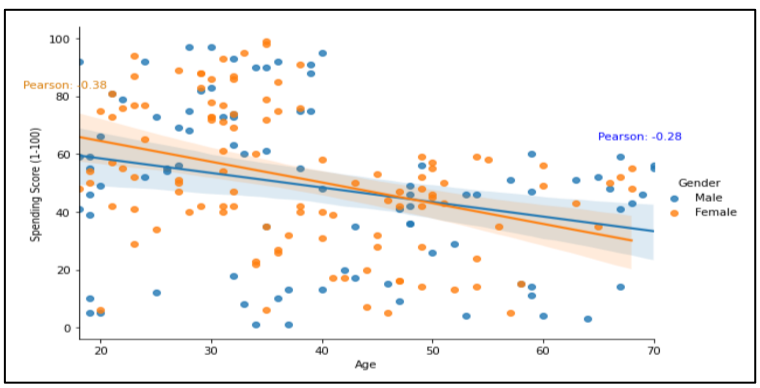
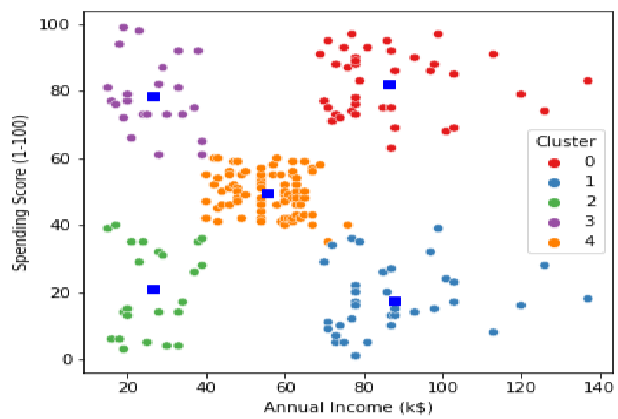
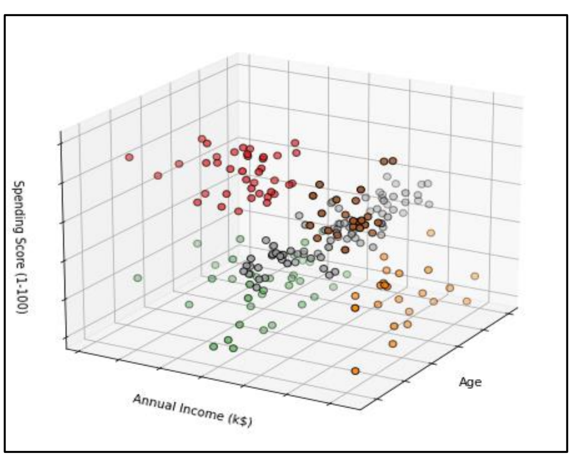
Implementation of 4 Clustering Algorithms: K-Means, DBSCAN, MeanShift and Agglomerative on a customer dataset and their comparison

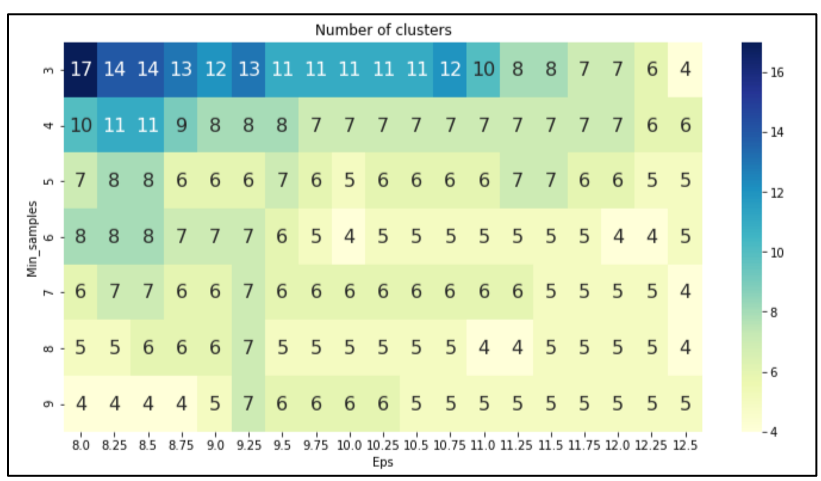












```
#Importing Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats

import warnings
warnings.filterwarnings("ignore")
```

```
from sklearn.metrics import silhouette_score

from scipy.cluster.hierarchy import linkage
from scipy.cluster.hierarchy import dendrogram
from scipy.cluster.hierarchy import cut_tree
```

```
#Importing dataset
data = pd.read_csv('Customers.csv')
print("There are {} rows and {} columns in the dataset".format(data.shape[0], data.shape[1]))
```

```
There are 200 rows and 5 columns in the dataset
```

Exploratory Data Analysis (EDA)

```
data.head()
```

	CustomerID	Gender	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

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column                  Non-Null Count  Dtype 
---  ------                  --------------  ----- 
 0   CustomerID              200 non-null    int64 
 1   Gender                  200 non-null    object
 2   Age                     200 non-null    int64 
 3   Annual Income (k$)      200 non-null    int64 
 4   Spending Score (1-100)  200 non-null    int64 
dtypes: int64(4), object(1)
memory usage: 7.9+ KB
```

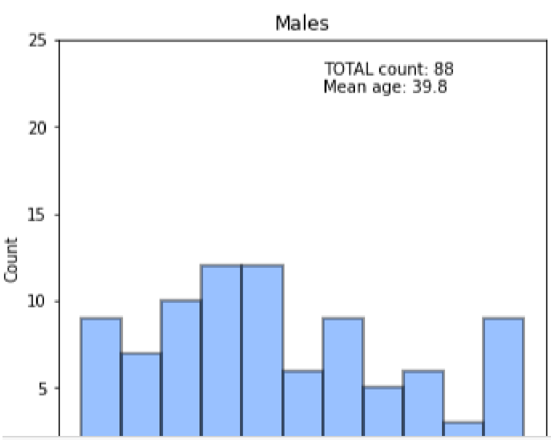
```
data.describe()
```

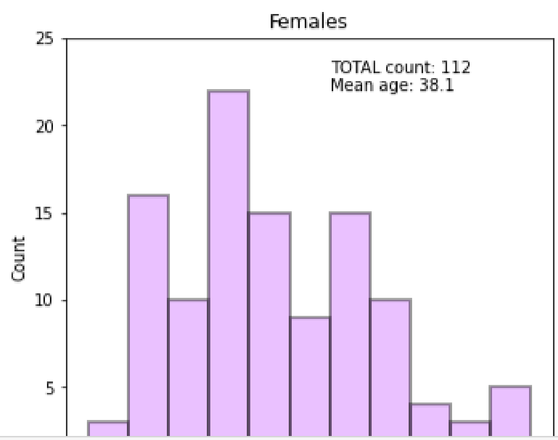
	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
count	200.000000	200.000000	200.000000	200.000000
mean	100.500000	38.850000	60.560000	50.200000
std	57.879185	13.969007	26.264721	25.823522
min	1.000000	18.000000	15.000000	1.000000
25%	50.750000	28.750000	41.500000	34.750000
50%	100.500000	36.000000	61.500000	50.000000
75%	150.250000	49.000000	78.000000	73.000000
max	200.000000	70.000000	137.000000	99.000000

```
data.isnull().sum()
```

```
CustomerID                0
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

```python
males_age = data[data['Gender']=='Male']['Age']
females_age = data[data['Gender']=='Female']['Age']

age_bins = range(15,75,5)

#males histogram
fig2, (ax1,ax2) = plt.subplots(1,2, figsize=(12,5), sharey=True)
sns.distplot(males_age, bins=age_bins, kde=False, color='#0066ff', ax=ax1, hist_kws=dict(edgecolor="k",linewidth=2))
ax1.set_xticks(age_bins)
ax1.set_ylim(top=25)
ax1.set_title('Males')
ax1.set_ylabel('Count')
ax1.text(45,23, 'TOTAL count: {}'.format(males_age.count()))
ax1.text(45,22, 'Mean age: {:.1f}'.format(males_age.mean()))

#females histogram
fig2, (ax1,ax2) = plt.subplots(1,2, figsize=(12,5), sharey=True)
sns.distplot(females_age, bins=age_bins, kde=False, color='#cc66ff', ax=ax1, hist_kws=dict(edgecolor="k",linewidth=2))
ax1.set_xticks(age_bins)
ax1.set_ylim(top=25)
ax1.set_title('Females')
ax1.set_ylabel('Count')
ax1.text(45,23, 'TOTAL count: {}'.format(females_age.count()))
ax1.text(45,22, 'Mean age: {:.1f}'.format(females_age.mean()))

plt.show()
```



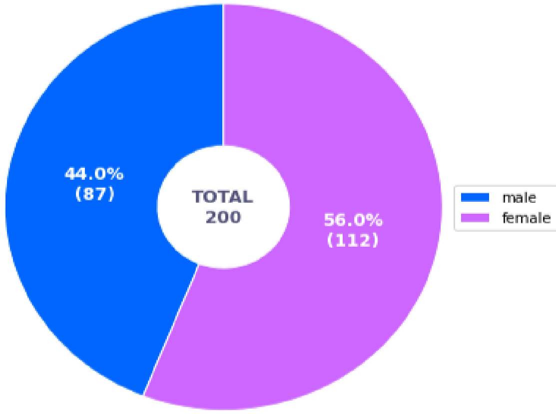



```python
print('Kolgomorov-Smirnov test p-value: {:.2f}'.format(stats.ks_2samp(males_age, females_age)[1]))
```

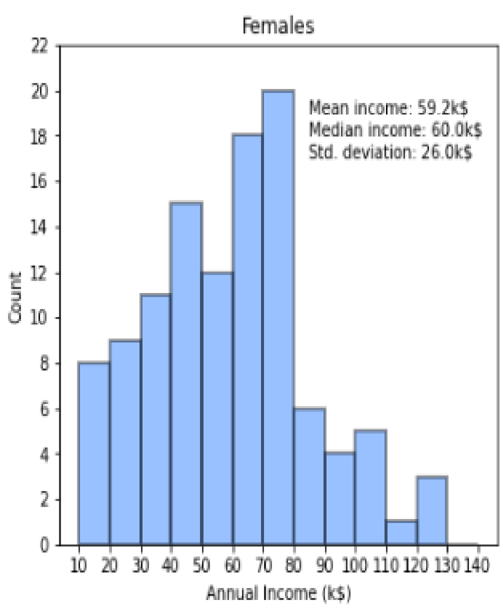
```
Kolgomorov-Smirnov test p-value: 0.49
```

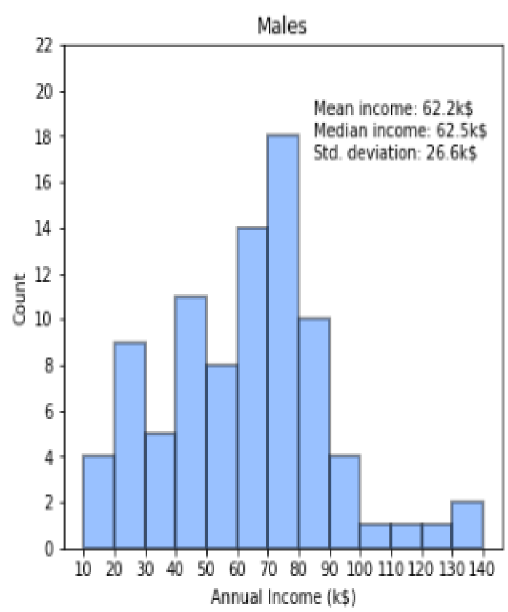
Kolgomorov-Smirnov test shows that the differences between these two groups are statistically insignificant

```python
def labeler(pct, allvals):
    absolute = int(pct/100.*np.sum(allvals))
    return "{:.1f}%\n({:d})".format(pct,absolute)

sizes = [males_age.count(), females_age.count()]
fig0, ax1 = plt.subplots(figsize =(6,6))
wedges, texts, autotexts = ax1.pie(sizes,
                                  autopct=lambda pct: labeler(pct, sizes),
                                  radius=1,
                                  colors=['#0066ff', '#cc66ff'],
                                  startangle=90,
                                  textprops=dict(color='w'),
                                  wedgeprops=dict(width=0.7, edgecolor='w'))
ax1.legend(wedges, ['male','female'],
          loc='center right',
          bbox_to_anchor=(0.7,0,0.5,1))
plt.text(0,0,'TOTAL\n{}'.format(data['Age'].count()),
        weight='bold', size=12, color='#52527a',
        ha='center', va='center')
plt.setp(autotexts, size=12, weight='bold')
ax1.axis('equal')
plt.show()
```



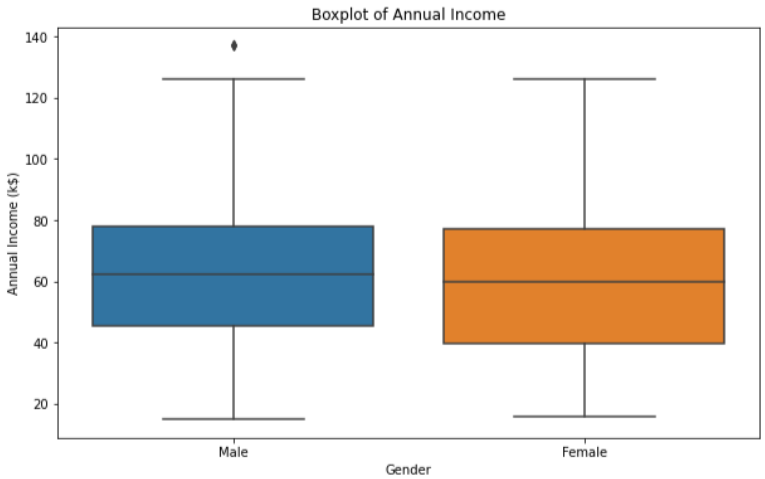

```python
males_income = data[data['Gender']=='Male']['Annual Income (k$)']
females_income = data[data['Gender']=='Female']['Annual Income (k$)']

my_bins = range(10,150,10)

#males histogram
fig2, (ax1,ax2,ax3) = plt.subplots(1,3, figsize=(18,5))
sns.distplot(males_income, bins=my_bins, kde=False, color='#0066ff', ax=ax1, hist_kws=dict(edgecolor="k",linewidth=2))
ax1.set_xticks(my_bins)
ax1.set_yticks(range(0,24,2))
ax1.set_ylim(0,22)
ax1.set_title('Males')
ax1.set_ylabel('Count')
ax1.text(85,19, 'Mean income: {:.1f}k$'.format(males_income.mean()))
ax1.text(85,18, 'Median income: {:.1f}k$'.format(males_income.median()))
ax1.text(85,17, 'Std. deviation: {:.1f}k$'.format(males_income.std()))

#females histogram
fig2, (ax1,ax2,ax3) = plt.subplots(1,3, figsize=(18,5))
sns.distplot(females_income, bins=my_bins, kde=False, color='#0066ff', ax=ax1, hist_kws=dict(edgecolor="k",linewidth=2))
ax1.set_xticks(my_bins)
ax1.set_yticks(range(0,24,2))
ax1.set_ylim(0,22)
ax1.set_title('Females')
ax1.set_ylabel('Count')
ax1.text(85,19, 'Mean income: {:.1f}k$'.format(females_income.mean()))
ax1.text(85,18, 'Median income: {:.1f}k$'.format(females_income.median()))
ax1.text(85,17, 'Std. deviation: {:.1f}k$'.format(females_income.std()))


plt.show()
```





```
#boxplot
sns.boxplot(x='Gender',y='Annual Income (k$)', data=data, ax=ax3)
ax3.set_title('Boxplot of Annual Income')
plt.show()
```

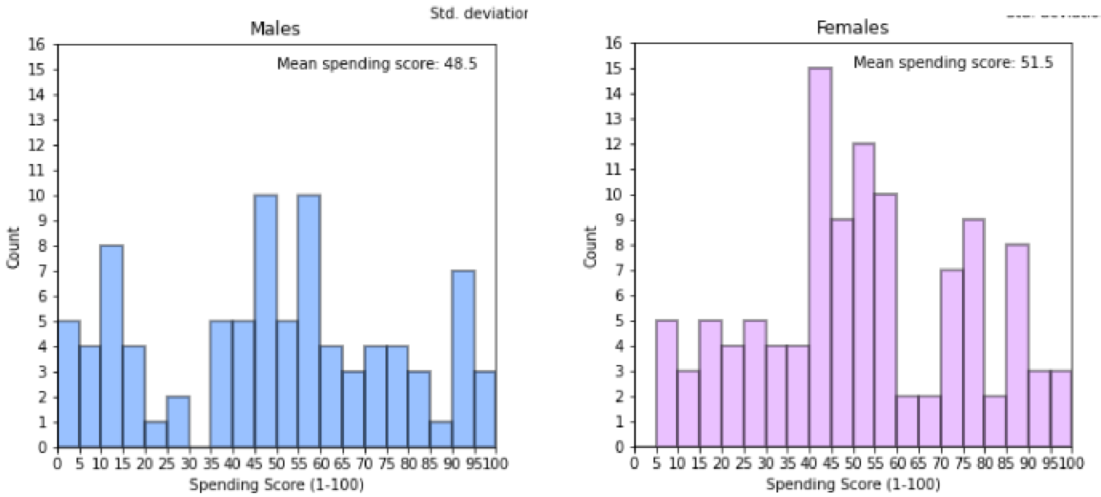
```
plt.figure(figsize=(10, 6))  # Adjust the size as needed
sns.boxplot(x='Gender', y='Annual Income (k$)', data=data)
plt.title('Boxplot of Annual Income')
plt.show()
```



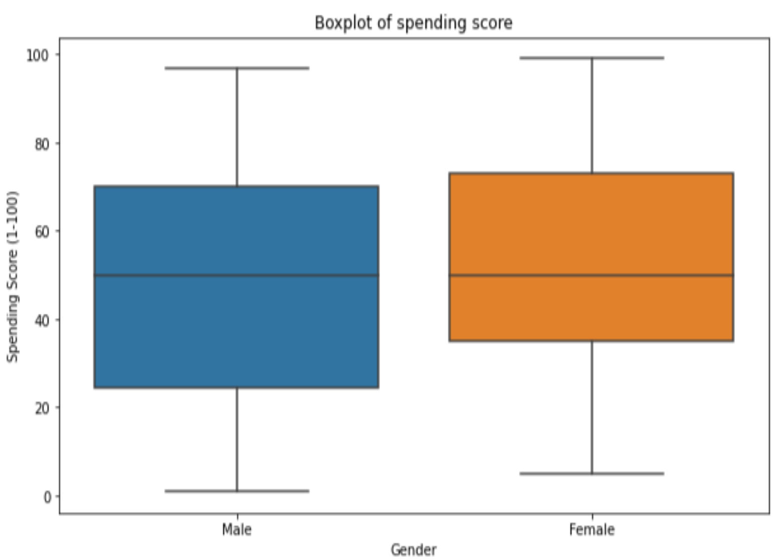

```
print('Kolgomorov-Smirnov test p-value: {:.2f}'.format(stats.ks_2samp(males_income, females_income)[1]))
```

```
Kolgomorov-Smirnov test p-value: 0.78
```

```
males_spending = data[data['Gender']=='Male']['Spending Score (1-100)']
females_spending = data[data['Gender']=='Female']['Spending Score (1-100)']

spending_bins = range(0,105,5)

#males histogram
fig2, (ax1,ax2,ax3) = plt.subplots(1,3, figsize=(18,5))
sns.distplot(males_spending, bins=spending_bins, kde=False, color='#0066ff', ax=ax1, hist_kws=dict(edgecolor="k",linewidth=2))
ax1.set_xticks(spending_bins)
ax1.set_xlim(0,100)
ax1.set_yticks(range(0,17,1))
ax1.set_ylim(0,16)
ax1.set_title('Males')
ax1.set_ylabel('Count')
ax1.text(50,15, 'Mean spending score: {:.1f}'.format(males_spending.mean()))
ax1.text(85,18, 'Median spending score: {:.1f}'.format(males_spending.median()))
ax1.text(85,17, 'Std. deviation spending score: {:.1f}'.format(males_spending.std()))

#females histogram
fig2, (ax1,ax2,ax3) = plt.subplots(1,3, figsize=(18,5))
sns.distplot(females_spending, bins=spending_bins, kde=False, color='#cc66ff', ax=ax1, hist_kws=dict(edgecolor="k",linewidth=2))
ax1.set_xticks(spending_bins)
ax1.set_xlim(0,100)
ax1.set_yticks(range(0,17,1))
ax1.set_ylim(0,16)
ax1.set_title('Females')
ax1.set_ylabel('Count')
ax1.text(50,15, 'Mean spending score: {:.1f}'.format(females_spending.mean()))
ax1.text(85,18, 'Median spending score: {:.1f}'.format(females_spending.median()))
ax1.text(85,17, 'Std. deviation spending score: {:.1f}'.format(females_spending.std()))

plt.show()
```




```
plt.figure(figsize=(10, 6))  # Adjust the size as needed
sns.boxplot(x='Gender', y='Spending Score (1-100)', data=data)
plt.title('Boxplot of spending score')
plt.show()
```

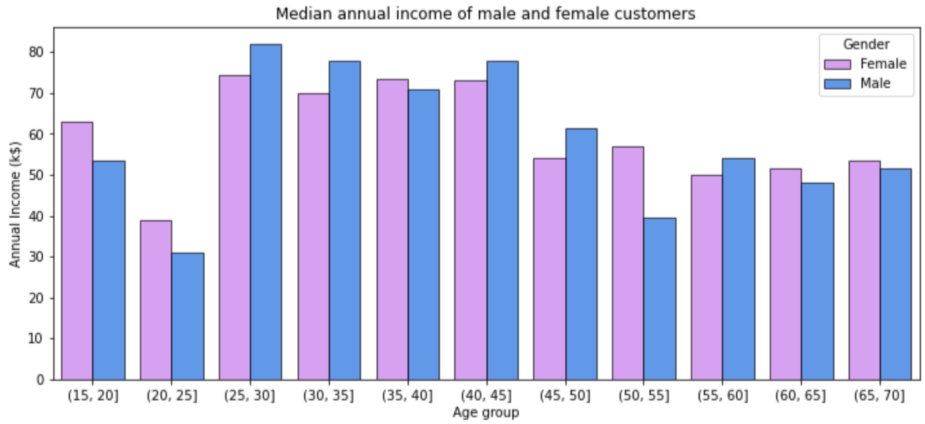



```
print('Kolgomorov-Smirnov test p-value: {:.2f}'.format(stats.ks_2samp(males_spending, females_spending)[1]))
```

```
Kolgomorov-Smirnov test p-value: 0.29
```

```
medians_by_age_group = data.groupby(["Gender", pd.cut(data['Age'],age_bins)]).median()
medians_by_age_group.index = medians_by_age_group.index.set_names(['Gender', 'Age_group'])
medians_by_age_group.reset_index(inplace=True)
```

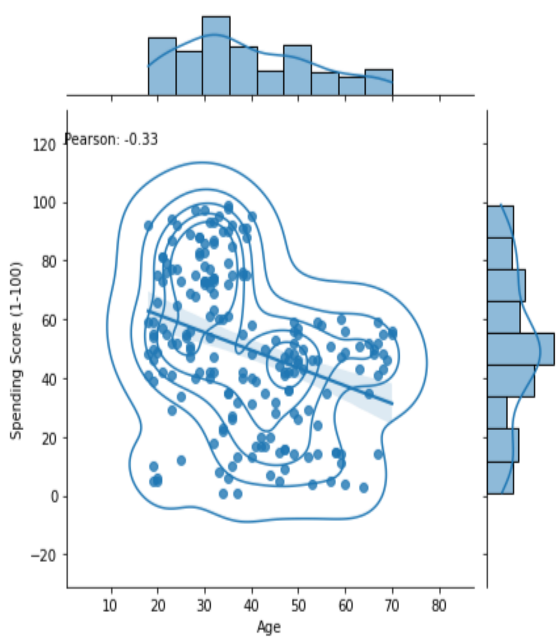
```
fig, ax = plt.subplots(figsize=(12,5))
sns.barplot(x='Age_group', y='Annual Income (k$)', hue='Gender',data=medians_by_age_group,
           palette=['#cc66ff','#0066ff'],
           alpha=0.7, edgecolor='k',
           ax=ax)
ax.set_title('Median annual income of male and female customers')
ax.set_xlabel('Age group')
plt.show()
```



Correlations

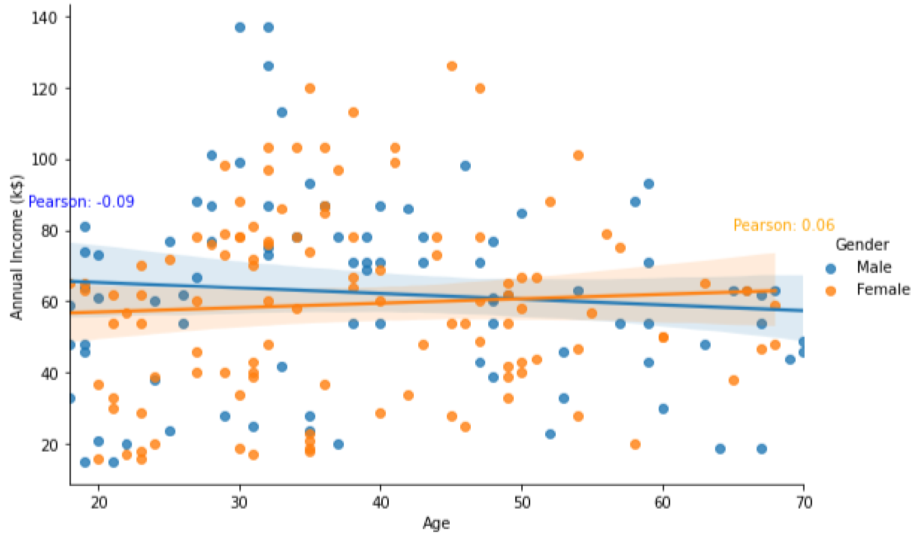
```
from scipy.stats import pearsonr
```

```
#Calculating Pearson's correlation
corr, _ = pearsonr(data['Age'], data['Spending Score (1-100)'])

jp = (sns.jointplot('Age','Spending Score (1-100)',data=data,
                    kind='reg')).plot_joint(sns.kdeplot, zorder=0, n_levels=6)
plt.text(0,120, 'Pearson: {:.2f}'.format(corr))
plt.show()
```




```
#Calculating Pearson's correlations
corr1, _ = pearsonr(males_age.values, males_income.values)
corr2, _ = pearsonr(females_age.values, females_income.values)

sns.lmplot('Age', 'Annual Income (k$)', data=data, hue='Gender',
          aspect=1.5)
plt.text(15,87, 'Pearson: {:.2f}'.format(corr1),color ='blue')
plt.text(65,80, 'Pearson: {:.2f}'.format(corr2),color ='orange')
```

```
Text(65, 80, 'Pearson: 0.06')
```

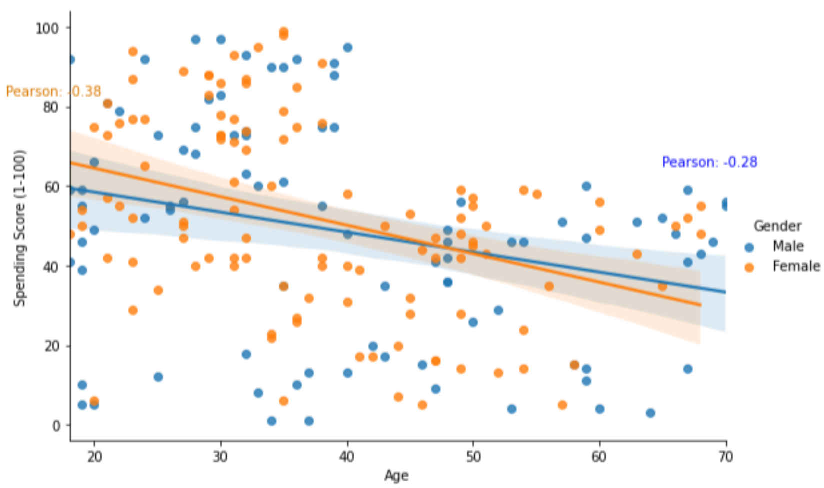


There is a negligible correlation between age and annule income of customers for both sex

Slope is almost parallel to x-axis

```
#Calculating Pearson's correlations
corr1, _ = pearsonr(males_age.values, males_spending.values)
corr2, _ = pearsonr(females_age.values, females_spending.values)

sns.lmplot('Age', 'Spending Score (1-100)', data=data, hue='Gender',
          aspect=1.5)
plt.text(65,65, 'Pearson: {:.2f}'.format(corr1),color ='blue')
plt.text(13,83, 'Pearson: {:.2f}'.format(corr2),color ='#d97900')
```

```
Text(13, 83, 'Pearson: -0.38')
```

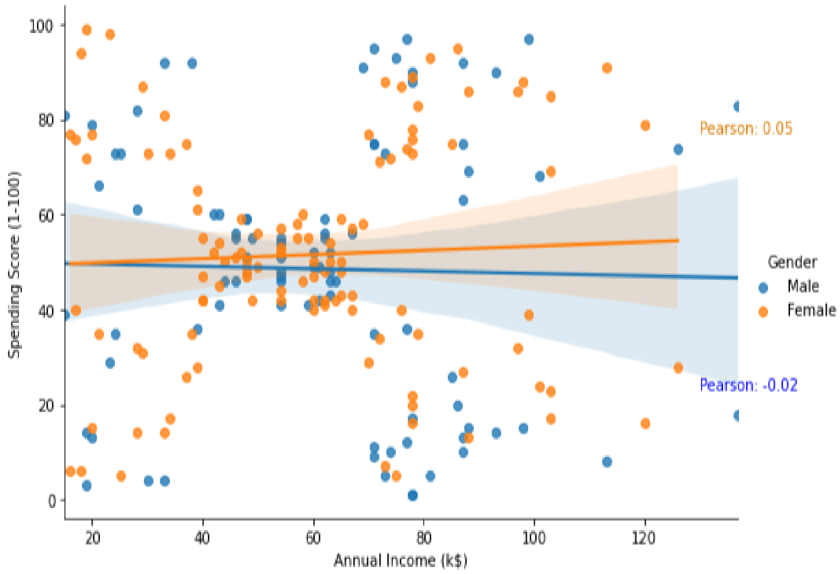


- There are weak negative corrleations (<0.5) between age and spending score of customers for both sex groups

- slope is slight negative here

- It says that with increase in age there is decrease in spending score

```
#Calculating Pearson's correlations
corr1, _ = pearsonr(males_income.values, males_spending.values)
corr2, _ = pearsonr(females_income.values, females_spending.values)

sns.lmplot('Annual Income (k$)', 'Spending Score (1-100)', data=data, hue='Gender',
           aspect=1.5)
plt.text(130,23, 'Pearson: {:.2f}'.format(corr1),color ='blue')
plt.text(130,77, 'Pearson: {:.2f}'.format(corr2),color ='#d97900')
plt.show()
```



- There is a negligible correlation between annual income and spending score of customers for both sex groups

CLUSTERING - 1.K Means, 2.DBSCAN, 3.MeanShift, 4.Agglomerative

K-Means

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

```
#subset with numeric variables only
X_numerics = data[['Age','Annual Income (k$)', 'Spending Score (1-100)']]
```

```
X_numerics.head()
```

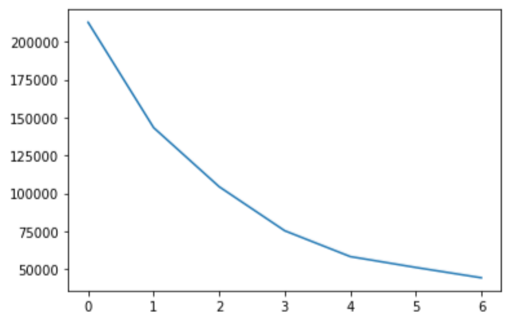
	Age	Annual Income (k$)	Spending Score (1-100)
0	19	15	39
1	21	15	81
2	20	16	6
3	23	16	77
4	31	17	40

```
# k-means with some arbitrary k
kmeans = KMeans(n_clusters=4, max_iter=50)
kmeans.fit(X_numerics)
kmeans.labels_
```

```
array([3, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3,
       0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3,
       0, 3, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 3,
       0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 1, 2, 1, 2, 1, 2, 1, 2,
       1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2,
       1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2,
       1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2,
       1, 2])
```

```
# elbow-curve/SSD
ssd = []
range_n_clusters = [2, 3, 4, 5, 6, 7, 8]
for num_clusters in range_n_clusters:
    kmeans = KMeans(n_clusters=num_clusters, max_iter=50)
    kmeans.fit(X_numerics)

    ssd.append(kmeans.inertia_)

# plot the SSDs for each n_clusters
# ssd
plt.plot(ssd)
```

```
[<matplotlib.lines.Line2D at 0x14573124070>]
```

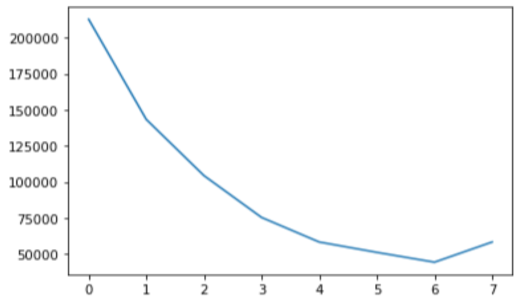



```
# silhouette analysis
range_n_clusters = [2, 3, 4, 5, 6, 7, 8]

for num_clusters in range_n_clusters:

    # intialise kmeans
    kmeans = KMeans(n_clusters=num_clusters, max_iter=50)
    kmeans.fit(X_numerics)

    cluster_labels = kmeans.labels_

    # silhouette score
    silhouette_avg = silhouette_score(X_numerics, cluster_labels)
    print("For n_clusters={0}, the silhouette score is {1}".format(num_clusters, silhouette_avg))
```

```
For n_clusters=2, the silhouette score is 0.293166070535953
For n_clusters=3, the silhouette score is 0.3839349967742105
For n_clusters=4, the silhouette score is 0.40546302077733304
For n_clusters=5, the silhouette score is 0.44428597560893024
For n_clusters=6, the silhouette score is 0.4523443947724053
For n_clusters=7, the silhouette score is 0.43978902692261157
For n_clusters=8, the silhouette score is 0.4280971079746162
```

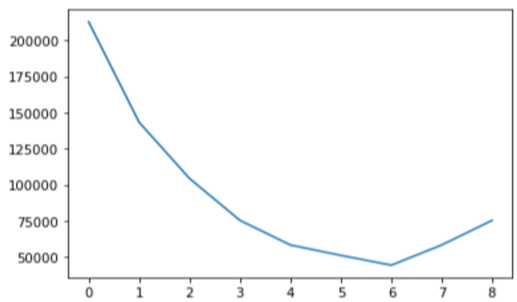
```
# final model with k=6
kmeans = KMeans(n_clusters=6, max_iter=50)
kmeans.fit(X_numerics)
ssd.append(kmeans.inertia_)
plt.plot(ssd)
```

[<matplotlib.lines.Line2D at 0x145732ab250>]



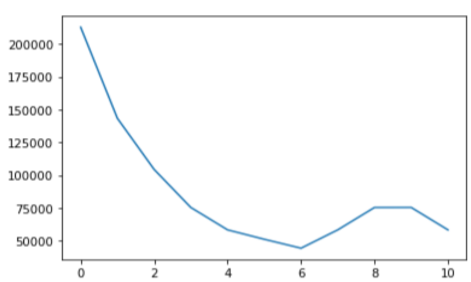
```
# final model with k=5
kmeans = KMeans(n_clusters=5, max_iter=50)
kmeans.fit(X_numerics)
ssd.append(kmeans.inertia_)
plt.plot(ssd)
```

[<matplotlib.lines.Line2D at 0x14573300970>]



```
KM_6_clusters = KMeans(n_clusters=6, init='k-means++').fit(X_numerics)
ssd.append(KM_6_clusters.inertia_)
plt.plot(ssd)
```

[<matplotlib.lines.Line2D at 0x145734ae340>]

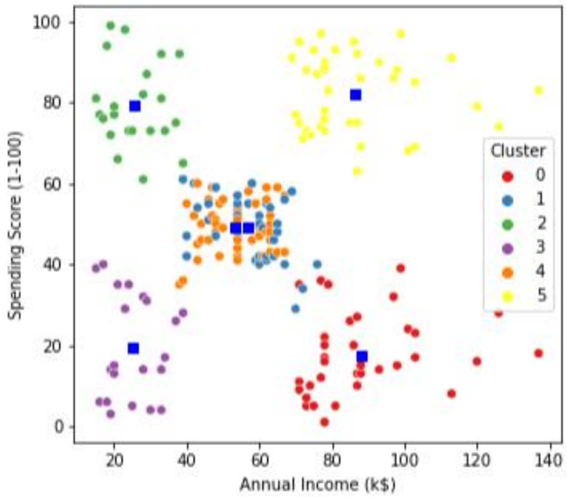


```
KM_6_clustered = X_numerics.copy()
KM_6_clustered.loc[:,'Cluster'] = KM_6_clusters.labels_
```

fig1, (axes) = plt.subplots(1,2,figsize=(12,5)) scat_1 = sns.scatterplot('Annual Income (k$)','Spending Score (1-100)', data=KM_6_clustered, hue='Cluster', ax=axes[0], palette='Set1', legend='full') sns.scatterplot('Age','Spending Score (1-100)', data=KM_6_clustered, hue='Cluster', ax=axes[0], palette='Set1', legend='full')

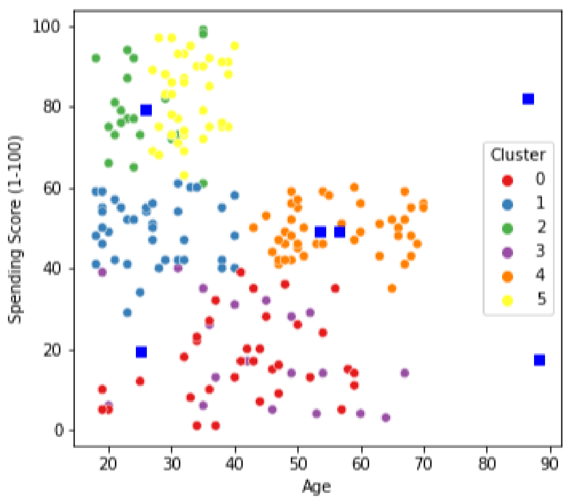
axes[0].scatter(KM_6_clusters.cluster_centers_[:,1], KM_6_clusters.cluster_centers_[:,2], marker='s',s=40, c="blue")
axes[1].scatter(KM_6_clusters.cluster_centers_[:,0], KM_6_clusters.cluster_centers_[:,2], marker='s',s=40, c="blue")

```
fig1, (axes) = plt.subplots(1,2,figsize=(12,5))
scat_1 = sns.scatterplot('Annual Income (k$)','Spending Score (1-100)', data=KM_6_clustered,
                        hue='Cluster', ax=axes[0], palette='Set1', legend='full')

axes[0].scatter(KM_6_clusters.cluster_centers_[:,1], KM_6_clusters.cluster_centers_[:,2], marker='s',s=40, c="blue")
```



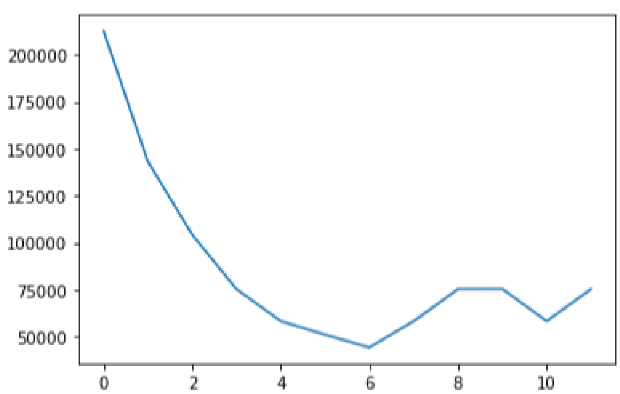

```
fig1, (axes) = plt.subplots(1,2,figsize=(12,5))
scat_1 = sns.scatterplot('Age','Spending Score (1-100)', data=KM_6_clustered,
                        hue='Cluster', ax=axes[0], palette='Set1', legend='full')

axes[0].scatter(KM_6_clusters.cluster_centers_[:,1], KM_6_clusters.cluster_centers_[:,2], marker='s',s=40, c="blue")
```

<matplotlib.collections.PathCollection at 0x1456f14c430>




```
KM_5_clusters = KMeans(n_clusters=5, init='k-means++').fit(X_numerics)
ssd.append(KM_5_clusters.inertia_)
plt.plot(ssd)
```

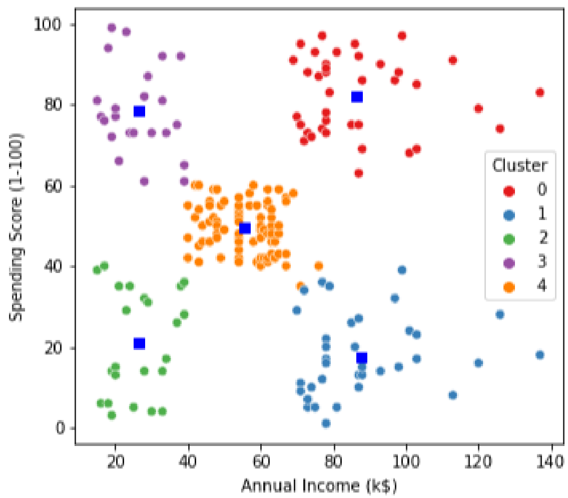
[<matplotlib.lines.Line2D at 0x14574777370>]



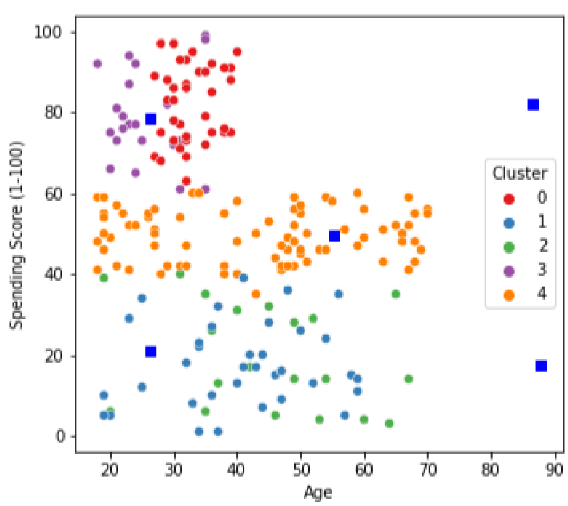
```
KM_5_clustered = X_numerics.copy()
KM_5_clustered.loc[:,'Cluster'] = KM_5_clusters.labels_
```

```
fig1, (axes) = plt.subplots(1,2,figsize=(12,5))
scat_1 = sns.scatterplot('Annual Income (k$)','Spending Score (1-100)', data=KM_5_clustered,
                        hue='Cluster', ax=axes[0], palette='Set1', legend='full')


axes[0].scatter(KM_5_clusters.cluster_centers_[:,1], KM_5_clusters.cluster_centers_[:,2], marker='s',s=40, c="blue")
```

```
<matplotlib.collections.PathCollection at 0x1457335bc10>
```




```
fig1, (axes) = plt.subplots(1,2,figsize=(12,5))
scat_1 = sns.scatterplot('Age','Spending Score (1-100)', data=KM_5_clustered,
                        hue='Cluster', ax=axes[0], palette='Set1', legend='full')


axes[0].scatter(KM_5_clusters.cluster_centers_[:,1], KM_5_clusters.cluster_centers_[:,2], marker='s',s=40, c="blue")
```



```
KM_5_clusters.labels_
```

```
array([2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3,
       2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3,
       2, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
       4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
       4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
       4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 0, 1, 0, 4, 0, 1, 0, 1, 0,
       1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 4, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0,
       1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0,
       1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0,
       1, 0])
```

```
# assign the label
X_numerics['cluster_id'] = KM_5_clusters.labels_
X_numerics.head()
```

	Age	Annual Income (k$)	Spending Score (1-100)	cluster_id
0	19	15	39	2
1	21	15	81	3
2	20	16	6	2
3	23	16	77	3
4	31	17	40	2

```
KM_6_clusters.labels_
```

```
array([3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2,
       3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 4, 2, 4, 1,
       3, 2, 4, 1, 1, 1, 4, 1, 1, 4, 4, 4, 4, 4, 1, 4, 4, 1, 4, 4, 4, 1,
       4, 4, 1, 1, 4, 4, 4, 4, 4, 1, 4, 1, 1, 4, 4, 1, 4, 4, 1, 4, 4, 1,
       1, 4, 4, 1, 4, 1, 1, 1, 4, 1, 4, 1, 1, 4, 4, 1, 4, 1, 4, 4, 4, 4,
       4, 1, 1, 1, 1, 1, 4, 4, 4, 4, 1, 1, 1, 5, 1, 5, 0, 5, 0, 5, 0, 5,
       1, 5, 0, 5, 0, 5, 0, 5, 0, 5, 1, 5, 0, 5, 0, 5, 0, 5, 0, 5, 0, 5,
       0, 5, 0, 5, 0, 5, 0, 5, 0, 5, 0, 5, 0, 5, 0, 5, 0, 5, 0, 5, 0, 5,
       0, 5, 0, 5, 0, 5, 0, 5, 0, 5, 0, 5, 0, 5, 0, 5, 0, 5, 0, 5, 0, 5,
       0, 5])
```

```
# assign the label
X_numerics['cluster_id'] = KM_6_clusters.labels_
X_numerics.head()
```

	Age	Annual Income (k$)	Spending Score (1-100)	cluster_id
0	19	15	39	3
1	21	15	81	2
2	20	16	6	3
3	23	16	77	2
4	31	17	40	3

```
KM5_clustered = X_numerics.copy()
KM5_clustered.loc[:,'Cluster'] = KM_5_clusters.labels_
```

```
KM5_clust_sizes = KM5_clustered.groupby('Cluster')
KM5_clust_sizes.columns = ["KM5_size"]
KM5_clust_sizes.describe()
```

	Age								Annual Income (k$)		...	cluster_id		Labels							
	count	mean	std	min	25%	50%	75%	max	count	mean	...	75%	max	count	mean	std	min	25%	50%	75%	max
Cluster																					
0	39.0	32.692308	3.728650	27.0	30.0	32.0	35.50	40.0	39.0	86.538462	...	1.0	1.0	39.0	2.000000	0.000000	2.0	2.0	2.0	2.0	2.0
1	36.0	40.666667	11.496583	19.0	34.0	41.5	47.25	59.0	36.0	87.750000	...	2.0	2.0	36.0	0.055556	0.232311	0.0	0.0	0.0	0.0	1.0
2	23.0	45.217391	13.228607	19.0	35.5	46.0	53.50	67.0	23.0	26.304348	...	4.0	4.0	23.0	4.000000	0.000000	4.0	4.0	4.0	4.0	4.0
3	23.0	25.521739	5.273170	18.0	21.5	24.0	30.00	35.0	23.0	26.304348	...	3.0	3.0	23.0	2.739130	0.688700	1.0	3.0	3.0	3.0	3.0
4	79.0	43.088608	16.478572	18.0	27.0	47.0	54.50	70.0	79.0	55.291139	...	0.0	2.0	79.0	0.987342	0.112509	0.0	1.0	1.0	1.0	1.0

5 rows × 40 columns

```
KM6_clustered = X_numerics.copy()
KM6_clustered.loc[:,'Cluster'] = KM_6_clusters.labels_
```

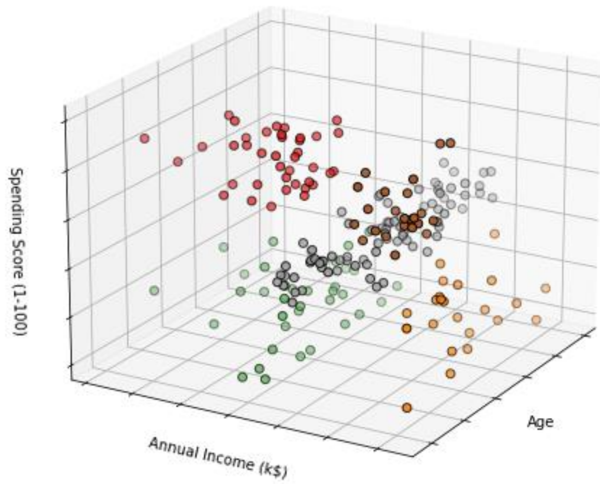
```
KM6_clust_sizes = KM6_clustered.groupby('Cluster')
KM6_clust_sizes.columns = ["KM6_size"]
KM6_clust_sizes.describe()
```

	Age								Annual Income (k$)		...	cluster_id		Labels							
	count	mean	std	min	25%	50%	75%	max	count	mean	...	75%	max	count	mean	std	min	25%	50%	75%	max
Cluster																					
0	35.0	41.685714	10.897305	19.0	35.00	43.0	47.50	59.0	35.0	88.228571	...	2.0	2.0	35.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.0
1	38.0	27.000000	7.032742	18.0	21.00	26.5	31.75	40.0	38.0	56.657895	...	0.0	3.0	38.0	1.000000	0.000000	1.0	1.0	1.0	1.0	1.0
2	22.0	25.272727	5.257030	18.0	21.25	23.5	29.75	35.0	22.0	25.727273	...	3.0	3.0	22.0	2.818182	0.588490	1.0	3.0	3.0	3.0	3.0
3	21.0	44.142857	13.089254	19.0	35.00	45.0	53.00	67.0	21.0	25.142857	...	4.0	4.0	21.0	4.000000	0.000000	4.0	4.0	4.0	4.0	4.0
4	45.0	56.155556	8.543886	43.0	49.00	54.0	65.00	70.0	45.0	53.377778	...	0.0	4.0	45.0	1.133333	0.625227	1.0	1.0	1.0	1.0	4.0
5	39.0	32.692308	3.728650	27.0	30.00	32.0	35.50	40.0	39.0	86.538462	...	1.0	1.0	39.0	2.000000	0.000000	2.0	2.0	2.0	2.0	2.0

6 rows × 40 columns

```
from mpl_toolkits.mplot3d import Axes3D

fig = plt.figure(figsize=(7,7))
ax = Axes3D(fig, rect=[0, 0, .99, 1], elev=20, azim=210)
ax.scatter(KM5_clustered['Age'],KM5_clustered['Annual Income (k$)'],
           KM5_clustered['Spending Score (1-100)'],
           c=KM_5_clusters.labels_,
           s=35, edgecolor='k', cmap=plt.cm.Set1)

ax.w_xaxis.set_ticklabels([])
ax.w_yaxis.set_ticklabels([])
ax.w_zaxis.set_ticklabels([])
ax.set_xlabel('Age')
ax.set_ylabel('Annual Income (k$)')
ax.set_zlabel('Spending Score (1-100)')
ax.set_title('3D view of K-Means 5 clusters')
ax.dist = 12
plt.show()
```

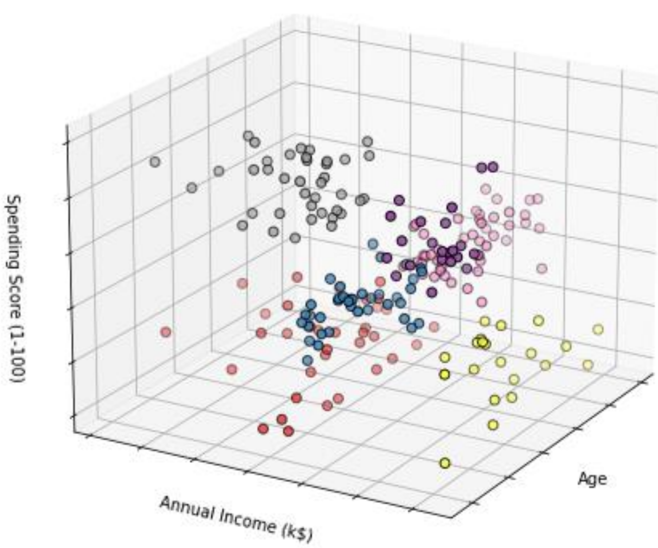
3D view of K-Means 5 clusters




```
KM_6_clustered = X_numerics.copy()
KM_6_clustered.loc[:,'Cluster'] = KM_6_clusters.labels_
```

```
from mpl_toolkits.mplot3d import Axes3D

fig = plt.figure(figsize=(7,7))
ax = Axes3D(fig, rect=[0, 0, .99, 1], elev=20, azim=210)
ax.scatter(KM_6_clustered['Age'],KM_6_clustered['Annual Income (k$)'],
           KM_6_clustered['Spending Score (1-100)'],
           c=KM_6_clusters.labels_,
           s=35, edgecolor='k', cmap=plt.cm.Set1)

ax.w_xaxis.set_ticklabels([])
ax.w_yaxis.set_ticklabels([])
ax.w_zaxis.set_ticklabels([])
ax.set_xlabel('Age')
ax.set_ylabel('Annual Income (k$)')
ax.set_zlabel('Spending Score (1-100)')
ax.set_title('3D view of K-Means 6 clusters')
ax.dist = 12
plt.show()
```

3D view of K-Means 6 clusters



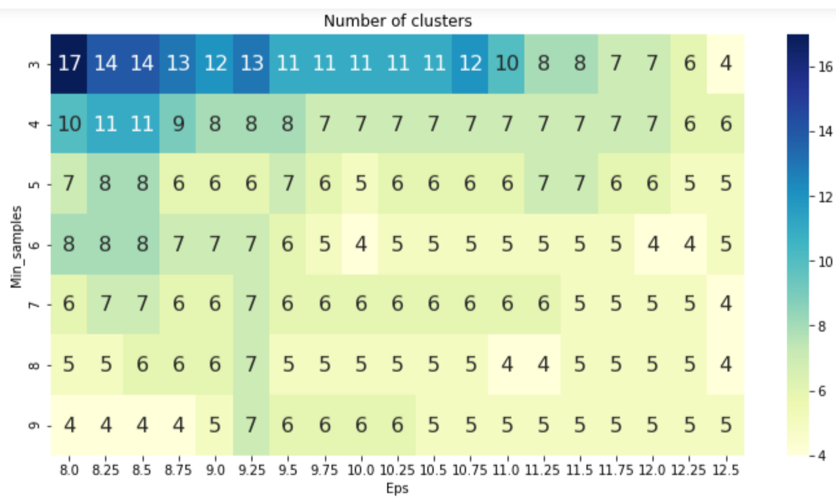
DBSCAN Clustering Algorithm

```
from sklearn.cluster import DBSCAN
```

```
from itertools import product

eps_values = np.arange(8,12.75,0.25) #eps values to be investigated
min_samples = np.arange(3,10) #min_samples values to be investigated
DBSCAN_params = list(product(eps_values, min_samples))
```

```
no_of_clusters = []
sil_score = []
for p in DBSCAN_params:
    DBS_clustering = DBSCAN(eps=p[0], min_samples=p[1]).fit(X_numerics)
    no_of_clusters.append(len(np.unique(DBS_clustering.labels_)))
    sil_score.append(silhouette_score(X_numerics, DBS_clustering.labels_))
```

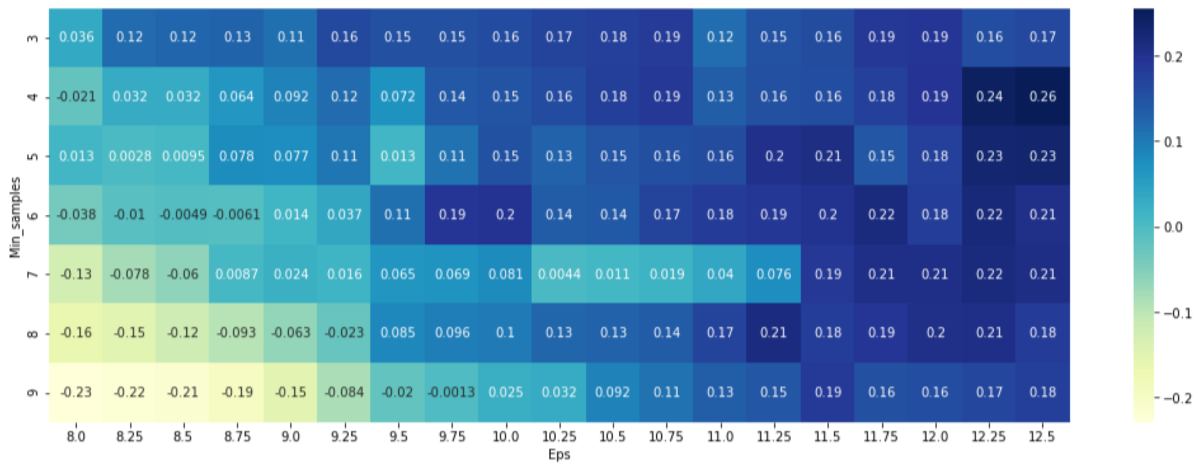
```
tmp = pd.DataFrame.from_records(DBSCAN_params, columns=['Eps','Min_samples'])
tmp['No_of_clusters'] = no_of_clusters

pivot_1 = pd.pivot_table(tmp, values='No_of_clusters', index='Min_samples', columns='Eps')

fig, ax = plt.subplots(figsize=(12,6))
sns.heatmap(pivot_1, annot=True, annot_kws={"size":16}, cmap="YlGnBu", ax=ax)
ax.set_title('Number of clusters')
plt.show()
```



From the above heatmap the number of clusters can range from 4 to 17.

Given sample size of 3 observations and eps = 8, it have 17 clusters.

```
tmp = pd.DataFrame.from_records(DBSCAN_params, columns=['Eps','Min_samples'])
tmp['Sil_score'] = sil_score

pivot_1 = pd.pivot_table(tmp, values='Sil_score', index='Min_samples', columns='Eps')

fig, ax = plt.subplots(figsize=(18,6))
sns.heatmap(pivot_1, annot=True, annot_kws={"size":10}, cmap="YlGnBu", ax=ax)

plt.show()
```



The silhouette score in the above heatmap is 0.26 for eps value=12.5 and sample size=4.

This is Global Maximum

```
DBS_clustering = DBSCAN(eps=12.5, min_samples=4).fit(X_numerics)

DBSCAN_clustered = X_numerics.copy()
DBSCAN_clustered.loc[:,'Cluster'] = DBS_clustering.labels_
```

```
DBSCAN_clust_sizes = DBSCAN_clustered.groupby('Cluster')
DBSCAN_clust_sizes.columns = ["DBSCAN_size"]
DBSCAN_clust_sizes.describe()
```

	Age								Annual Income (k$)		...	Spending Score (1-100)		cluster_id							
	count	mean	std	min	25%	50%	75%	max	count	mean	...	75%	max	count	mean	std	min	25%	50%	75%	max
Cluster																					
-1	18.0	36.944444	12.316762	20.0	32.00	34.5	36.50	67.0	18.0	74.000	...	77.75	99.0	18.0	2.611111	1.974511	0.0	0.5	3.0	4.75	5.0
0	112.0	39.142857	16.002735	18.0	24.00	37.0	50.00	70.0	112.0	48.250	...	58.25	92.0	112.0	2.491071	1.355639	0.0	1.0	2.0	4.00	4.0
1	8.0	53.250000	7.382412	42.0	48.25	53.5	58.50	64.0	8.0	27.750	...	14.25	17.0	8.0	3.000000	0.000000	3.0	3.0	3.0	3.00	3.0
2	34.0	32.882353	3.859382	27.0	30.00	32.0	36.00	40.0	34.0	82.000	...	90.75	97.0	34.0	5.000000	0.000000	5.0	5.0	5.0	5.00	5.0
3	24.0	45.583333	8.303570	34.0	39.25	44.0	52.50	59.0	24.0	85.875	...	23.25	39.0	24.0	0.000000	0.000000	0.0	0.0	0.0	0.00	0.0
4	4.0	20.750000	2.872281	19.0	19.00	19.5	21.25	25.0	4.0	76.250	...	10.50	12.0	4.0	0.000000	0.000000	0.0	0.0	0.0	0.00	0.0

6 rows × 32 columns

cluster -1 is an outlier. There are 18 outliers present

```
DBS_clustering.labels_
```

```
array([ 0,  0, -1,  0,  0,  0, -1, -1,  1,  0, -1, -1,  1,  0, -1,  0,  0,
        0,  0, -1,  0,  0,  1,  0,  1,  0,  0,  0,  0,  1,  0,  1,  0,
        1,  0,  1,  0,  0,  0, -1,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  2,  0,  2,  0,  2,  3,  2,  3,  2,  0,  2,  4,  2,
        3,  2,  4,  2,  3,  2,  0,  2,  4,  2,  0,  2,  3,  2,  3,  2,  3,
        2,  3,  2,  3,  2, -1,  2,  3,  2,  4,  2,  3,  2,  3,  2,  3,  2,
        3,  2,  3,  2,  3,  2,  3,  2,  3,  2,  3,  2,  3,  2,  3,  2,  3,
       -1,  3,  2,  3, -1, -1,  2, -1, -1, -1, -1, -1, -1], dtype=int64)
```

DBSCAN created 5 clusters plus outlier cluster [-1]. Sizes of cluster 0-4 vary significantly

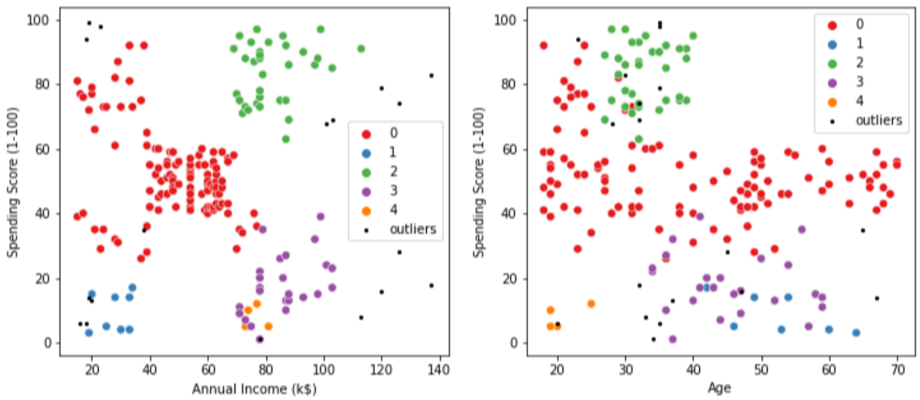
```
outliers = DBSCAN_clustered[DBSCAN_clustered['Cluster']==-1]

fig2, (axes) = plt.subplots(1,2,figsize=(12,5))

sns.scatterplot('Annual Income (k$)', 'Spending Score (1-100)',
                data=DBSCAN_clustered[DBSCAN_clustered['Cluster']!=-1],
                hue='Cluster', ax=axes[0], palette='Set1', legend='full', s=45)

sns.scatterplot('Age', 'Spending Score (1-100)',
                data=DBSCAN_clustered[DBSCAN_clustered['Cluster']!=-1],
                hue='Cluster', palette='Set1', ax=axes[1], legend='full', s=45)

axes[0].scatter(outliers['Annual Income (k$)'], outliers['Spending Score (1-100)'], s=5, label = 'outliers', c="k")
axes[1].scatter(outliers['Age'], outliers['Spending Score (1-100)'], s=5, label = 'outliers', c="k")
axes[0].legend()
axes[1].legend()

plt.setp(axes[0].get_legend().get_texts(), fontsize ='10')
plt.setp(axes[1].get_legend().get_texts(), fontsize ='10')

plt.show()
```

```
# assign the label
X_numerics['cluster_id'] = DBS_clustering.labels_
X_numerics.head(10)
```

	Age	Annual Income (k$)	Spending Score (1-100)	cluster_id
0	19	15	39	0
1	21	15	81	0
2	20	16	6	-1
3	23	16	77	0
4	31	17	40	0
5	22	17	76	0
6	35	18	6	-1
7	23	18	94	-1
8	64	19	3	1
9	30	19	72	0

```
# assign the label
X_numerics['cluster_id'] = DBS_clustering.labels_
X_numerics.tail(10)
```

	Age	Annual Income (k$)	Spending Score (1-100)	cluster_id
190	34	103	23	3
191	32	103	69	-1
192	33	113	8	-1
193	38	113	91	2
194	47	120	16	-1
195	35	120	79	-1
196	45	126	28	-1
197	32	126	74	-1
198	32	137	18	-1
199	30	137	83	-1




```
dbscan_clustered = X_numerics.copy()
dbscan_clustered.loc[:,'Cluster'] = DBS_clustering.labels_
```

```
dbscan_clust_sizes = dbscan_clustered.groupby('Cluster')
dbscan_clust_sizes.columns = ["dbscan_size"]
dbscan_clust_sizes.describe()
```

	Age								Annual Income (k$)		...	cluster_id		Labels							
	count	mean	std	min	25%	50%	75%	max	count	mean	...	75%	max	count	mean	std	min	25%	50%	75%	max
Cluster																					
-1	18.0	36.944444	12.316762	20.0	32.00	34.5	36.50	67.0	18.0	74.000	...	3.75	4.0	18.0	2.166667	1.581139	0.0	0.5	2.0	3.75	4.0
0	112.0	39.142857	16.002735	18.0	24.00	37.0	50.00	70.0	112.0	48.250	...	3.00	4.0	112.0	1.553571	1.064067	0.0	1.0	1.0	1.00	4.0
1	8.0	53.250000	7.382412	42.0	48.25	53.5	58.50	64.0	8.0	27.750	...	4.00	4.0	8.0	4.000000	0.000000	4.0	4.0	4.0	4.00	4.0
2	34.0	32.882353	3.859382	27.0	30.00	32.0	36.00	40.0	34.0	82.000	...	1.00	1.0	34.0	2.000000	0.000000	2.0	2.0	2.0	2.00	2.0
3	24.0	45.583333	8.303570	34.0	39.25	44.0	52.50	59.0	24.0	85.875	...	2.00	2.0	24.0	0.000000	0.000000	0.0	0.0	0.0	0.00	0.0
4	4.0	20.750000	2.872281	19.0	19.00	19.5	21.25	25.0	4.0	76.250	...	2.00	2.0	4.0	0.000000	0.000000	0.0	0.0	0.0	0.00	0.0

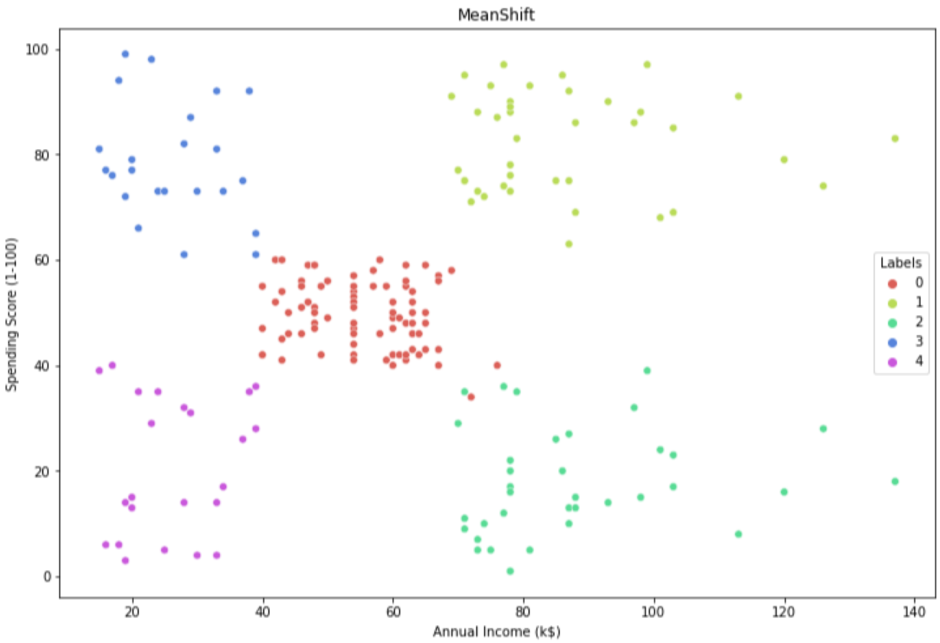
6 rows × 40 columns

MeanShift Clustering Algorithm ¶

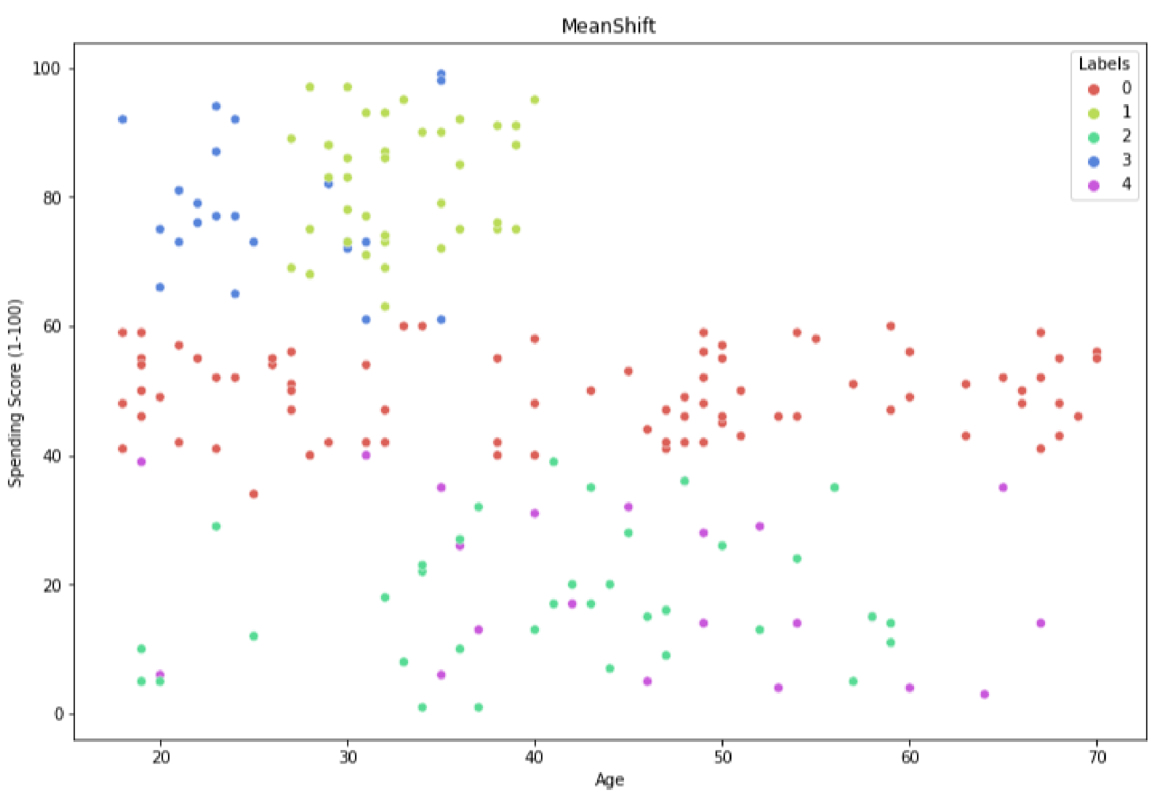
```
# k-means with some arbitrary k
ms = MeanShift(bandwidth=25)
ms.fit(X_numerics)
ms.labels_
```

```
array([4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3,
       4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3,
       4, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 1, 2, 1, 2, 1, 2, 1,
       0, 1, 2, 1, 2, 1, 2, 1, 2, 1, 0, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
       2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
       2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
       2, 1], dtype=int64)
```

```
from sklearn.cluster import MeanShift, estimate_bandwidth

ms=MeanShift(bandwidth=25).fit(X_numerics)

X_numerics['Labels'] = ms.labels_
plt.figure(figsize=(12,8))
sns.scatterplot(X_numerics['Annual Income (k$)'], X_numerics['Spending Score (1-100)'], hue=X_numerics['Labels'],
                palette = sns.color_palette('hls', np.unique(ms.labels_).shape[0]))
plt.plot()
plt.title('MeanShift')
plt.show()
```




```python
from sklearn.cluster import MeanShift, estimate_bandwidth

ms=MeanShift(bandwidth=25).fit(X_numerics)

X_numerics['Labels'] = ms.labels_
plt.figure(figsize=(12,8))
sns.scatterplot(X_numerics['Age'], X_numerics['Spending Score (1-100)'], hue=X_numerics['Labels'],
                palette = sns.color_palette('hls', np.unique(ms.labels_).shape[0]))
plt.plot()
plt.title('MeanShift')
plt.show()
```



```
MS_clustered = X_numerics.copy()
MS_clustered.loc[:,'cluster'] = ms.labels_
```

```
# assign the label
X_numerics['cluster_id'] = ms.labels_
X_numerics.head(10)
```

	Age	Annual Income (k$)	Spending Score (1-100)	cluster_id	Labels
0	19	15	39	4	4
1	21	15	81	3	3
2	20	16	6	4	4
3	23	16	77	3	3
4	31	17	40	4	4
5	22	17	76	3	3
6	35	18	6	4	4
7	23	18	94	3	3
8	64	19	3	4	4
9	30	19	72	3	3

```
ms_clustered = X_numerics.copy()
ms_clustered.loc[:,'Cluster'] = ms.labels_
```

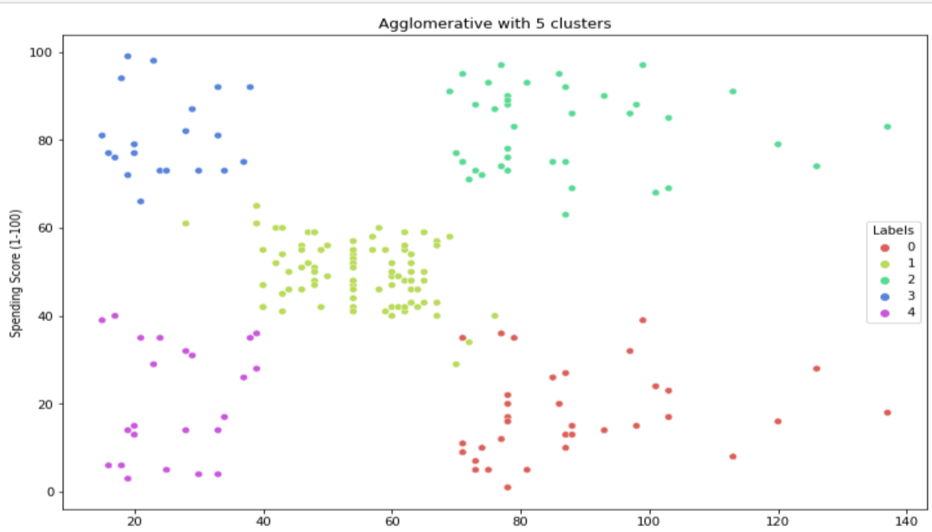
```
ms_clust_sizes = ms_clustered.groupby('Cluster')
ms_clust_sizes.columns = ["ms_size"]
ms_clust_sizes.describe()
```

	Age								Annual Income (k$)		...	cluster_id		Labels							
	count	mean	std	min	25%	50%	75%	max	count	mean	...	75%	max	count	mean	std	min	25%	50%	75%	max
Cluster																					
0	79.0	42.860759	16.603779	18.0	27.0	47.0	54.50	70.0	79.0	55.303797	...	0.0	0.0	79.0	1.000000	0.000000	1.0	1.0	1.0	1.0	1.0
1	39.0	32.692308	3.728650	27.0	30.0	32.0	35.50	40.0	39.0	86.538462	...	1.0	1.0	39.0	2.000000	0.000000	2.0	2.0	2.0	2.0	2.0
2	36.0	41.166667	11.182895	19.0	34.0	42.5	47.25	59.0	36.0	87.722222	...	2.0	2.0	36.0	0.027778	0.166667	0.0	0.0	0.0	0.0	1.0
3	23.0	25.521739	5.273170	18.0	21.5	24.0	30.00	35.0	23.0	26.304348	...	3.0	3.0	23.0	2.739130	0.688700	1.0	3.0	3.0	3.0	3.0
4	23.0	45.217391	13.228607	19.0	35.5	46.0	53.50	67.0	23.0	26.304348	...	4.0	4.0	23.0	4.000000	0.000000	4.0	4.0	4.0	4.0	4.0

5 rows × 40 columns

Agglomerative Clustering

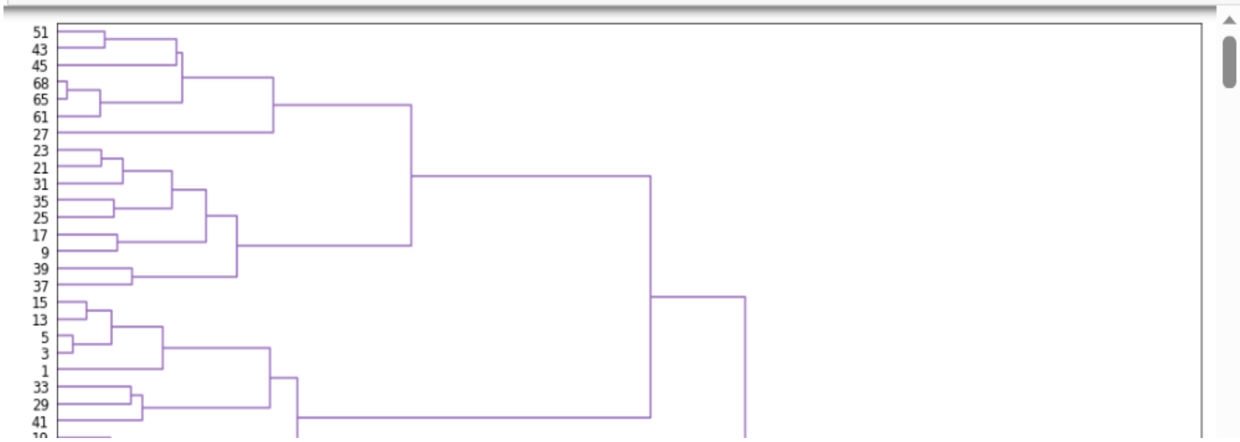
```
from sklearn.cluster import AgglomerativeClustering

agglom = AgglomerativeClustering(n_clusters=5, linkage='average').fit(X_numerics)

X_numerics['Labels'] = agglom.labels_
plt.figure(figsize=(12,8))
sns.scatterplot(X_numerics['Annual Income (k$)'], X_numerics['Spending Score (1-100)'], hue=X_numerics['Labels'],
                palette=sns.color_palette('hls',5))
plt.title('Agglomerative with 5 clusters')
plt.show()
```



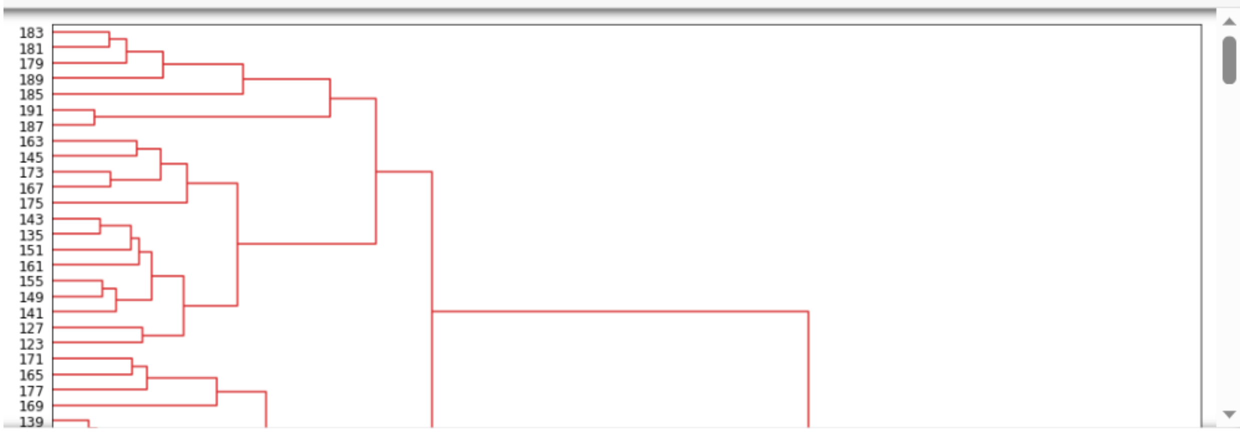
```
from scipy.cluster import hierarchy
from scipy.spatial import distance_matrix

dist = distance_matrix(X_numerics, X_numerics)
print(dist)
```

```
[[  0.          42.07136794  33.03028913 ... 117.16654813 124.55520864
  130.20752666]
 [ 42.07136794   0.          75.02666193 ... 111.7855089  137.7824372
  122.36829655]
 [ 33.03028913  75.02666193   0.         ... 129.92690253 122.26610323
  143.81585448]
 ...
 [117.16654813 111.7855089  129.92690253 ...   0.          57.11392125
   14.35270009]
 [124.55520864 137.7824372  122.26610323 ...  57.11392125   0.
   65.06919394]
 [130.20752666 122.36829655 143.81585448 ...  14.35270009  65.06919394
    0.        ]]
```

```
Z = hierarchy.linkage(dist, 'complete')
plt.figure(figsize=(18,50))
dendro = hierarchy.dendrogram(Z, leaf_rotation = 0, leaf_font_size = 12, orientation='right')
```




```
Z = hierarchy.linkage(dist, 'average')
plt.figure(figsize=(18,50))
dendro = hierarchy.dendrogram(Z, leaf_rotation = 0, leaf_font_size = 12, orientation='right')
```




```
Agg_clustered = X_numerics.copy()
Agg_clustered.loc[:,'Cluster'] = agglom.labels_
```

```
Agg_clust_sizes = Agg_clustered.groupby('Cluster')
Agg_clust_sizes.columns = ["Agg_size"]
Agg_clust_sizes.describe()
```

```
Agg_clustered = X_numerics.copy()
Agg_clustered.loc[:,'Cluster'] = agglom.labels_
```

```
Agg_clust_sizes = Agg_clustered.groupby('Cluster')
Agg_clust_sizes.columns = ["Agg_size"]
Agg_clust_sizes.describe()
```

	Age								Annual Income (k$)		...	cluster_id		Labels							
	count	mean	std	min	25%	50%	75%	max	count	mean	...	75%	max	count	mean	std	min	25%	50%	75%	max
Cluster																					
0	35.0	41.685714	10.897305	19.0	35.0	43.0	47.50	59.0	35.0	88.228571	...	2.0	2.0	35.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	83.0	42.156627	16.533397	18.0	27.0	45.0	54.00	70.0	83.0	54.759036	...	0.0	3.0	83.0	1.0	0.0	1.0	1.0	1.0	1.0	1.0
2	39.0	32.692308	3.728650	27.0	30.0	32.0	35.50	40.0	39.0	86.538462	...	1.0	1.0	39.0	2.0	0.0	2.0	2.0	2.0	2.0	2.0
3	20.0	24.850000	5.029126	18.0	21.0	23.0	29.25	35.0	20.0	24.950000	...	3.0	3.0	20.0	3.0	0.0	3.0	3.0	3.0	3.0	3.0
4	23.0	45.217391	13.228607	19.0	35.5	46.0	53.50	67.0	23.0	26.304348	...	4.0	4.0	23.0	4.0	0.0	4.0	4.0	4.0	4.0	4.0

5 rows × 40 columns

COMPARISON:

Cluster	KM5_size	KM6_size	DBSCAN_size	MS_size	Agg_size
-1	NaN	NaN	18.0	NaN	NaN
0	23.0	39.0	112.0	79.0	35.0
1	79.0	44.0	8.9	39.0	83.0
2	30.0	35.0	34.0	36.0	39.0
3	36.0	22.0	24.0	23.0	20.0
4	23.0	22.0	4.0	23.0	23.0
5	NaN	38.0	NaN	NaN	NaN