customer-segmentation-analysis

April 13, 2023

1 Importing Necessary Libraries

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

2 Data Ingestion

[2]:
```
df=pd.read_csv(r'C:\Users\PS4Z\Downloads\archive\Mall_Customers.csv')
```

[3]:
```
#seeing how the looks like
df.head()
```

[3]:

	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

3 Understanding the Data

[4]:
```
#seeing the shape of data
print('Data Shape: ',df.shape)
```

```
Data Shape:  (200, 5)
```

[5]:
```
#finding null values in data
df.isnull().sum()
```

[5]:
```
CustomerID                0
Genre                     0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

[6]:
```python
#Getting information about data; null counts and data types of data columns
df.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   Genre                   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
```

[7]:
```python
#list of column names
df.columns
```

[7]:
```
Index(['CustomerID', 'Genre', 'Age', 'Annual Income (k$)',
       'Spending Score (1-100)'],
      dtype='object')
```

[8]:
```python
#getting data types of each column header
df.dtypes
```

[8]:
```
CustomerID                 int64
Genre                     object
Age                        int64
Annual Income (k$)         int64
Spending Score (1-100)     int64
dtype: object
```

[9]:
```python
#checking for duplicate values
df.duplicated().sum()
```

[9]: 0

[10]:
```python
#getting five point summary of the data
df.describe()
```

[10]:

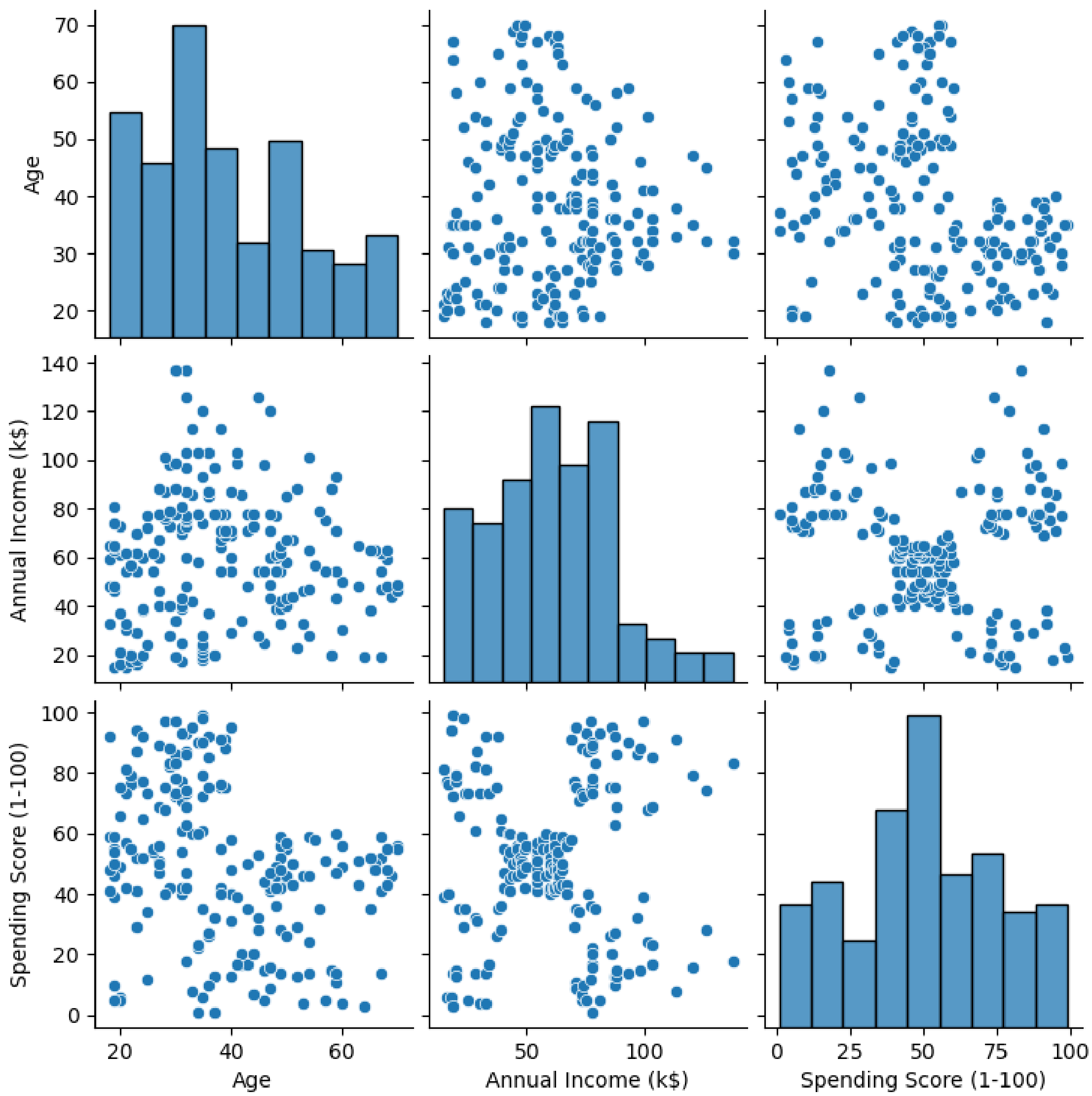
	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
count	200.000000	200.000000	200.000000	200.000000
mean	100.500000	38.850000	60.560000	50.200000
std	57.879185	13.969007	26.264721	25.823522
min	1.000000	18.000000	15.000000	1.000000
25%	50.750000	28.750000	41.500000	34.750000
50%	100.500000	36.000000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

[11]:
```
#making a separate data frame for selected features
feat_df=df[['Age','Annual Income (k$)','Spending Score (1-100)']]
feat=feat_df.columns
feat
```

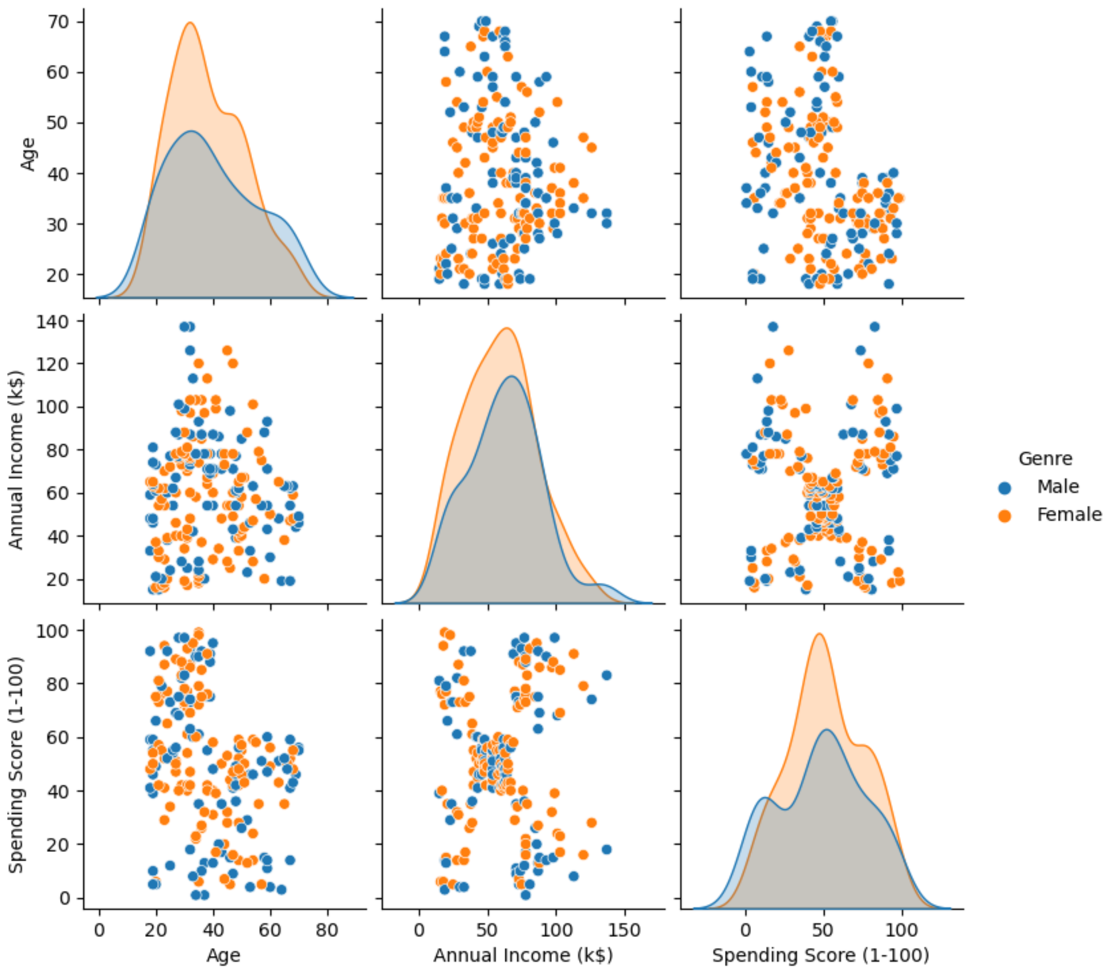
[11]:
```
Index(['Age', 'Annual Income (k$)', 'Spending Score (1-100)'], dtype='object')
```

4 Visualizing the Data

[12]:
```
#checking the relations of each such features with one another
sns.pairplot(data=df,vars=feat_df);
```

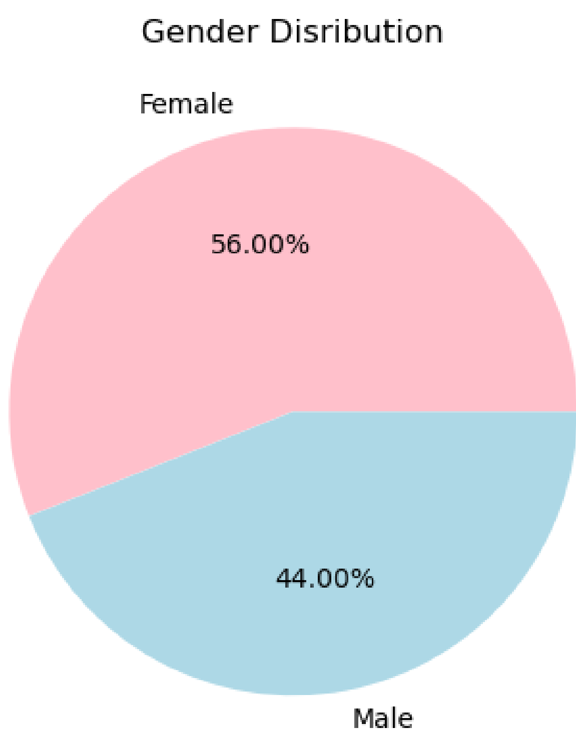


[13]:
```
#checking relations with Genre as hue
sns.pairplot(data=df,vars=feat_df,hue='Genre');
```

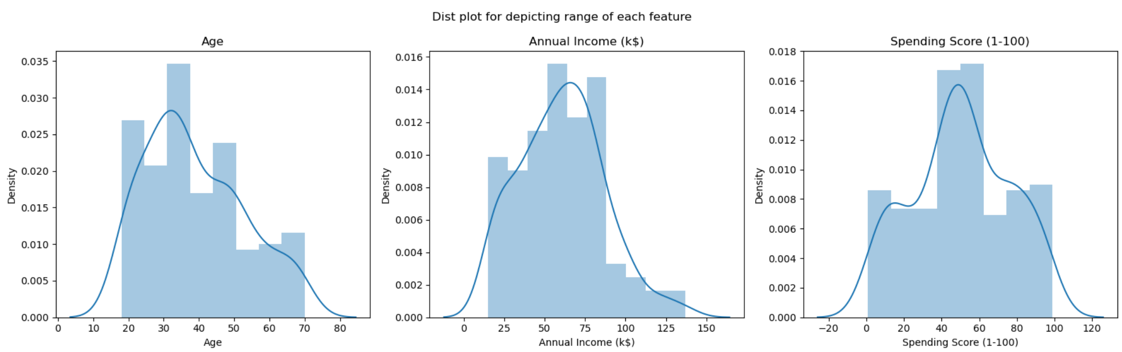


[14]:

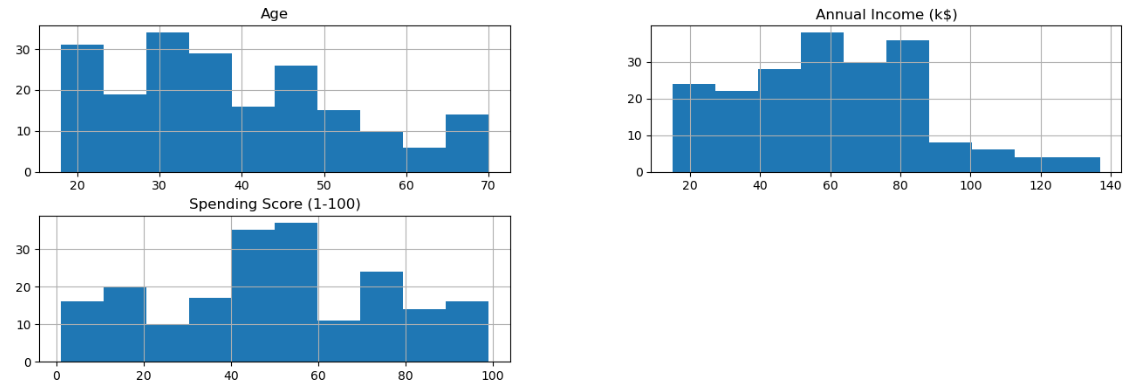
```
#Pie chart for Gender Distribution
df1=df.groupby('Genre').size()

df1.plot(kind='pie',subplots=True,colors=['pink','lightblue'],explode=[0,0.
 ↪001],labels=['Female','Male'],autopct='%.2f%%' )
plt.title("Gender Disribution")
plt.ylabel(" ");
```



[15]:
```
#plotting features
plt.figure(figsize=(16,5))
plt.suptitle('Dist plot for depicting range of each feature')
for a in range(0,len(feat)):
    plt.subplot(1,3,a+1)
    sns.distplot(df[feat[a]])
    plt.title(label=feat[a])
    plt.tight_layout();
```



[16]:
```
#Histogram of select features
feat_df.hist(figsize=(16,5));
```



[17]:
```
!pip install dabl
```

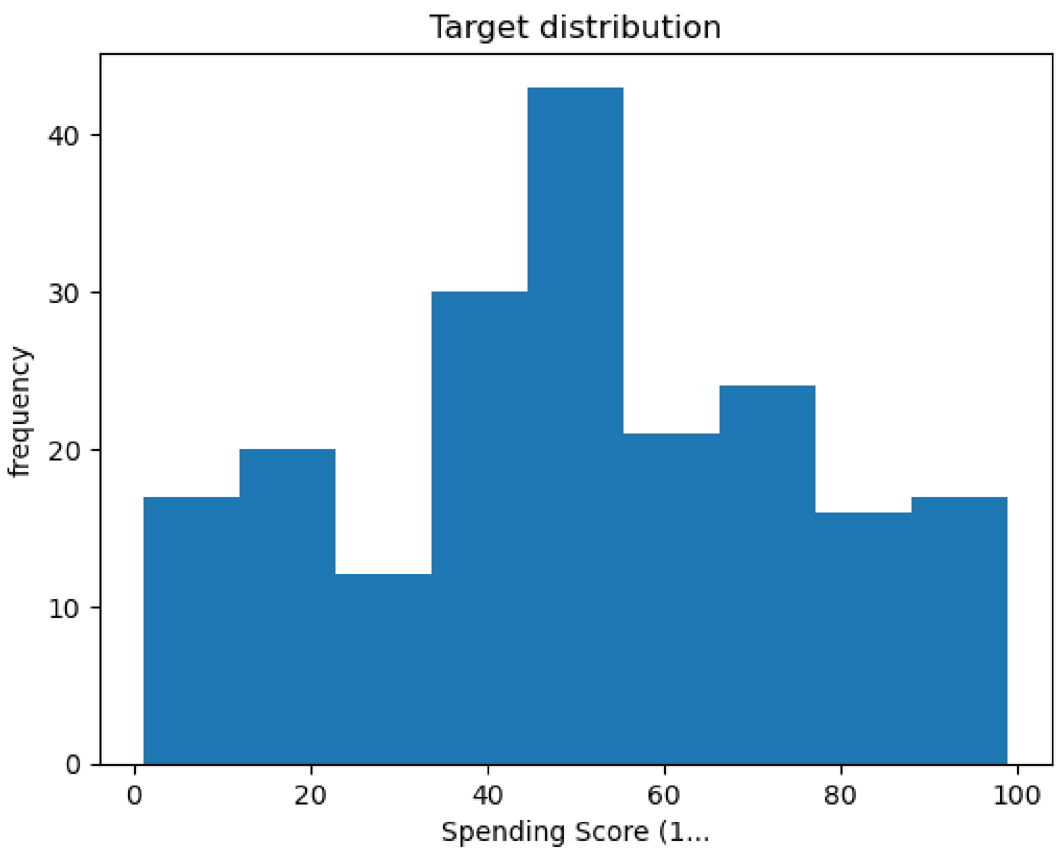
```
Requirement already satisfied: dabl in c:\users\ps4z\anaconda3\lib\site-packages
(0.2.5)
Requirement already satisfied: scikit-learn>=1.1 in
c:\users\ps4z\anaconda3\lib\site-packages (from dabl) (1.2.2)
Requirement already satisfied: pandas in c:\users\ps4z\anaconda3\lib\site-
packages (from dabl) (1.4.4)
Requirement already satisfied: seaborn in c:\users\ps4z\anaconda3\lib\site-
packages (from dabl) (0.11.2)
Requirement already satisfied: matplotlib>=3.5 in
c:\users\ps4z\anaconda3\lib\site-packages (from dabl) (3.5.2)
Requirement already satisfied: scipy in c:\users\ps4z\anaconda3\lib\site-
packages (from dabl) (1.9.1)
Requirement already satisfied: numpy in c:\users\ps4z\anaconda3\lib\site-
packages (from dabl) (1.21.5)
Requirement already satisfied: packaging>=20.0 in
c:\users\ps4z\anaconda3\lib\site-packages (from matplotlib>=3.5->dabl) (21.3)
Requirement already satisfied: kiwisolver>=1.0.1 in
c:\users\ps4z\anaconda3\lib\site-packages (from matplotlib>=3.5->dabl) (1.4.2)
Requirement already satisfied: python-dateutil>=2.7 in
c:\users\ps4z\anaconda3\lib\site-packages (from matplotlib>=3.5->dabl) (2.8.2)
Requirement already satisfied: pillow>=6.2.0 in
c:\users\ps4z\anaconda3\lib\site-packages (from matplotlib>=3.5->dabl) (9.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\ps4z\anaconda3\lib\site-
packages (from matplotlib>=3.5->dabl) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
c:\users\ps4z\anaconda3\lib\site-packages (from matplotlib>=3.5->dabl) (4.25.0)
Requirement already satisfied: pyparsing>=2.2.1 in
c:\users\ps4z\anaconda3\lib\site-packages (from matplotlib>=3.5->dabl) (3.0.9)
```

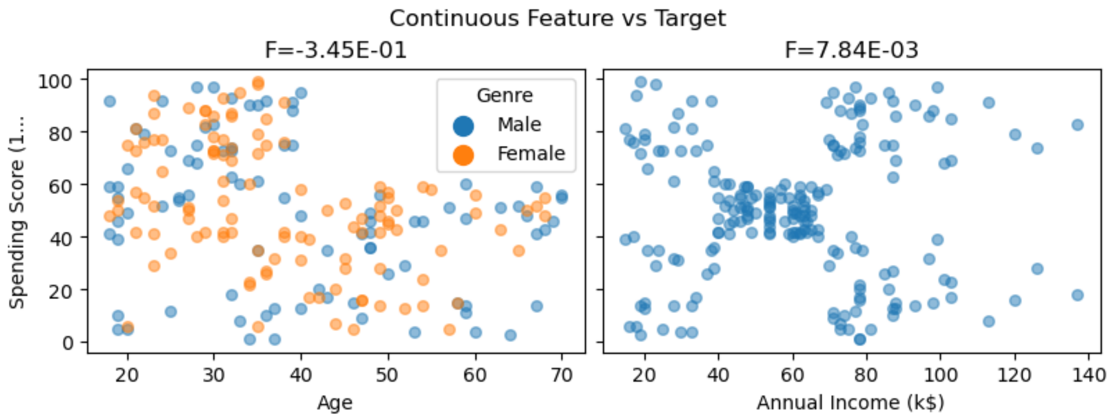
```
Requirement already satisfied: threadpoolctl>=2.0.0 in
c:\users\ps4z\anaconda3\lib\site-packages (from scikit-learn>=1.1->dabl) (2.2.0)
Requirement already satisfied: joblib>=1.1.1 in
c:\users\ps4z\anaconda3\lib\site-packages (from scikit-learn>=1.1->dabl) (1.2.0)
Requirement already satisfied: pytz>=2020.1 in c:\users\ps4z\anaconda3\lib\site-
packages (from pandas->dabl) (2022.1)
Requirement already satisfied: six>=1.5 in c:\users\ps4z\anaconda3\lib\site-
packages (from python-dateutil>=2.7->matplotlib>=3.5->dabl) (1.16.0)
```

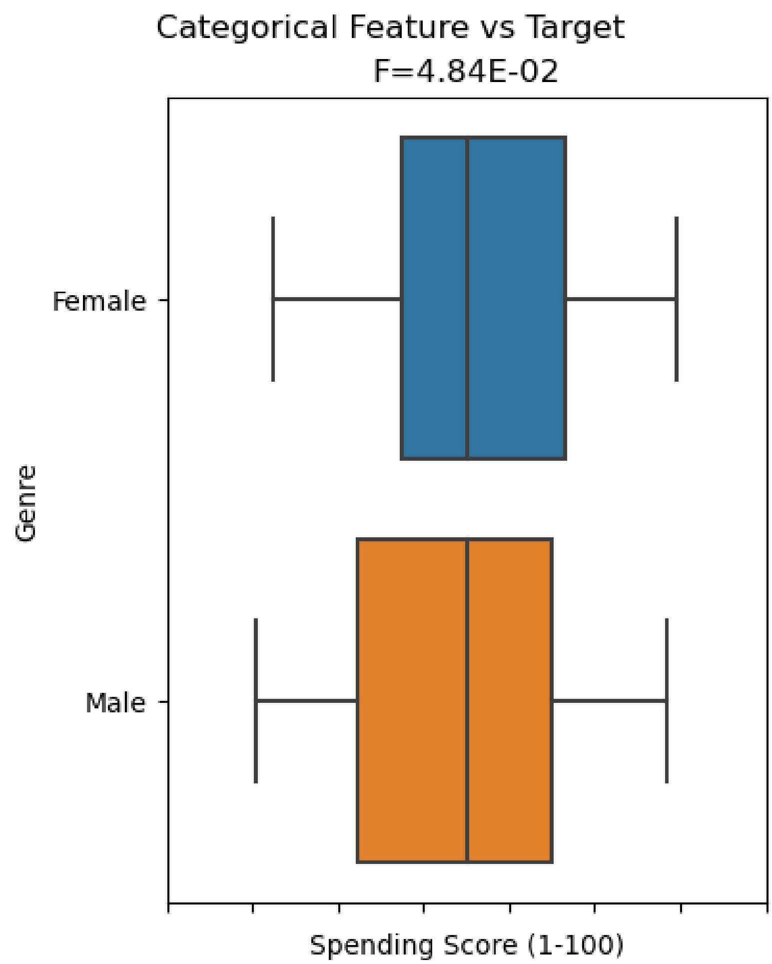
[18]:
```
import dabl
```

[19]:
```
#statistical Data Analysis
dabl.plot(df,target_col='Spending Score (1-100)');
```

```
Target looks like regression
```



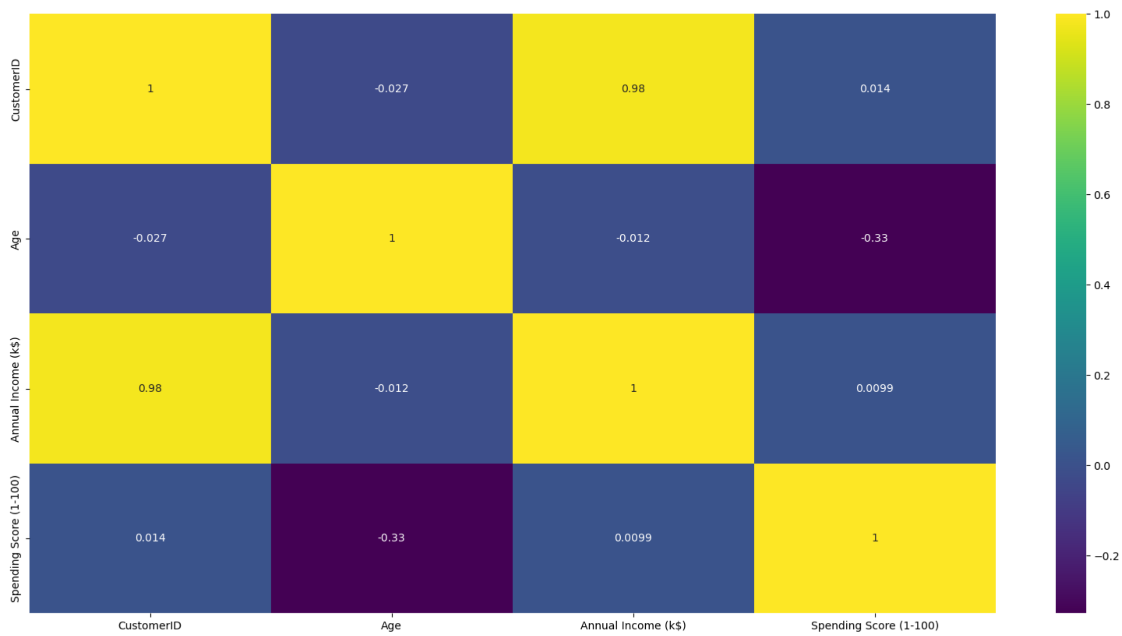




[20]:
```python
#Correlation among features
df.corr()
```

[20]:
```
                        CustomerID       Age  Annual Income (k$)  \
CustomerID                1.000000 -0.026763            0.977548
Age                      -0.026763  1.000000           -0.012398
Annual Income (k$)        0.977548 -0.012398            1.000000
Spending Score (1-100)    0.013835 -0.327227            0.009903

                        Spending Score (1-100)
CustomerID                            0.013835
Age                                  -0.327227
Annual Income (k$)                    0.009903
Spending Score (1-100)                1.000000
```

[21]:
```
#Plotting the correlation into a Heatmap
plt.figure(figsize=(20,10))
sns.heatmap(data=df.corr(),cmap='viridis',annot=True);
```



5 Relationhip between numerical variables

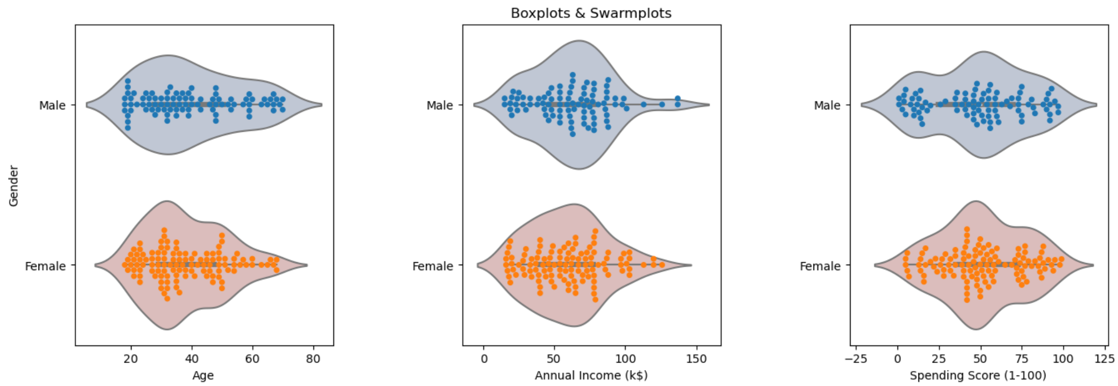
[22]:
```
#Age VS Annual Income based on Gender
plt.figure(figsize=(16,5))
for gender in ['Male','Female']:
    plt.scatter(x='Age',y='Annual Income (k$)',␣
 ↪data=df[df['Genre']==gender],s=200,alpha=0.5,label=gender)
plt.xlabel("Age")
plt.ylabel("Annual Income")
```

```
plt.title("Age vs Annual Income")
plt.legend();
```

[23]:
```
#Annual Income VS Spending Score(1-100) based on Gender
plt.figure(1,figsize=(16,5))
for gender in ['Male','Female']:
    plt.scatter(x='Annual Income (k$)',y='Spending Score␣
↪(1-100)',data=df[df['Genre']==gender],s=200,alpha=0.5,label=gender)
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.title('Annual Income (k$) vs Spending Score (1-100)')
plt.legend();
```

[24]:
```
#Distribution of values in Age, Annual Income and Spending Score according to␣
↪Gender
plt.figure(1 ,figsize=(16,5))
n = 0
for cols in feat_df:
```

```
    n+=1
    plt.subplot(1 ,3 ,n)
    plt.subplots_adjust(hspace=0.5,wspace=0.5)
    sns.violinplot(x=cols,y='Genre',data=df,palette='vlag')
    sns.swarmplot(x=cols,y='Genre',data=df)
    plt.ylabel('Gender'if n==1 else '')
    plt.title('Boxplots & Swarmplots' if n==2 else '')
```



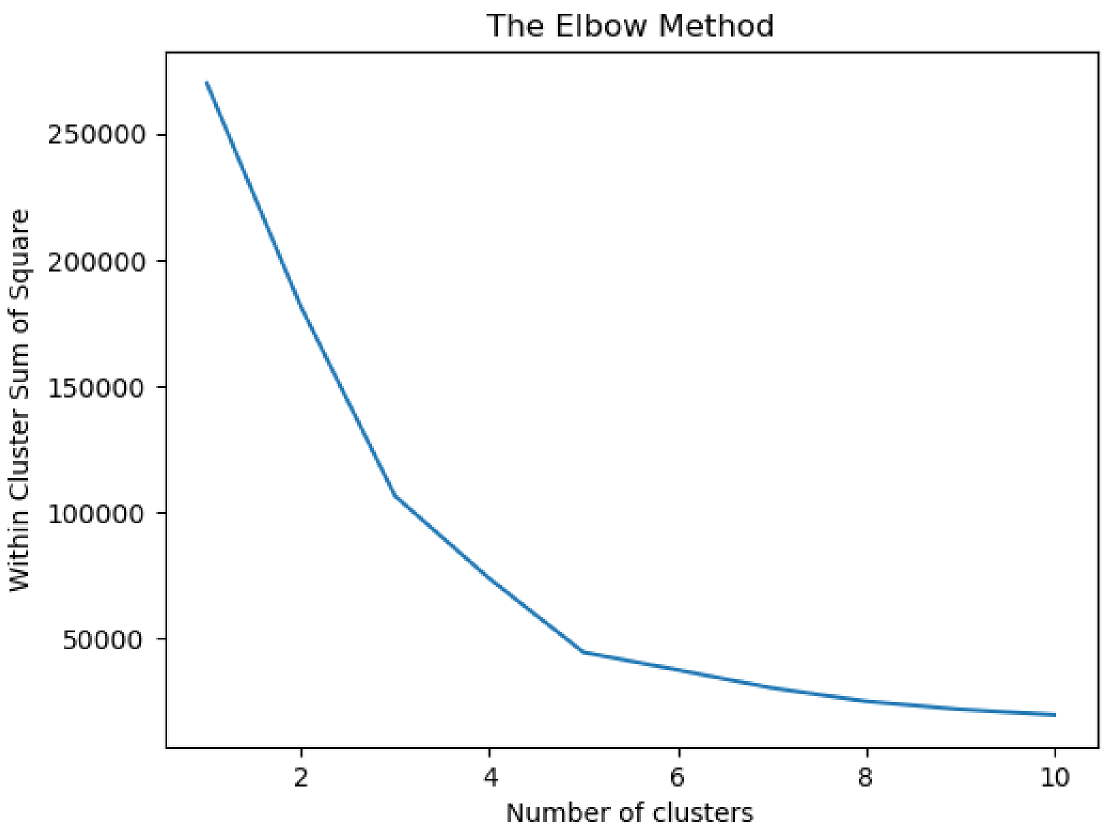
6 K-Means Clusreting

[25]:
```
from sklearn.cluster import KMeans
```

[26]:
```
#Choosing the variables Annual Income and Spending Score to cluster the data
x=df.iloc[:,[3,4]].values
```

[27]:
```
#Using the elbow method to determine the number of clusters
k = []
for i in range(1,11):
    kmeans=KMeans(n_clusters=i,init='k-means++',random_state=101)
    kmeans.fit(x)
    k.append(kmeans.inertia_)
```

[28]:
```
plt.plot(range(1, 11), k)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('Within Cluster Sum of Square');
```



Here, after 5 clusters, there is no significant decrease in the Within Cluster Sum of Square of the datapoints. Hence, we can assume value of k to be 5.

[29]:
```
#Model
model=KMeans(n_clusters=5,init='k-means++',random_state=101)
y=model.fit_predict(x)
```

[30]:
```
plt.figure(1 ,figsize=(20,10))
#First Cluster
plt.scatter(x[y==0,0],x[y==0,1],s=100,c='green',label='First Cluster')
#Second Cluster
plt.scatter(x[y==1,0],x[y==1,1],s=100,c='red',label='Second Cluster')
#Third Cluster
plt.scatter(x[y==2,0],x[y==2,1],s=100,c='yellow',label='Third Cluster')
#Forth Cluster
plt.scatter(x[y==3,0],x[y==3,1],s=100,c='blue',label='Forth Cluster')
#Fifth Cluster
plt.scatter(x[y==4,0],x[y==4,1],s=100,c='purple',label='Fifth Cluster')

plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:
 ↪,1],s=200,c='black',label='Centroids')
```

```
plt.title('K Means Clustering Algorithm')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend();
```