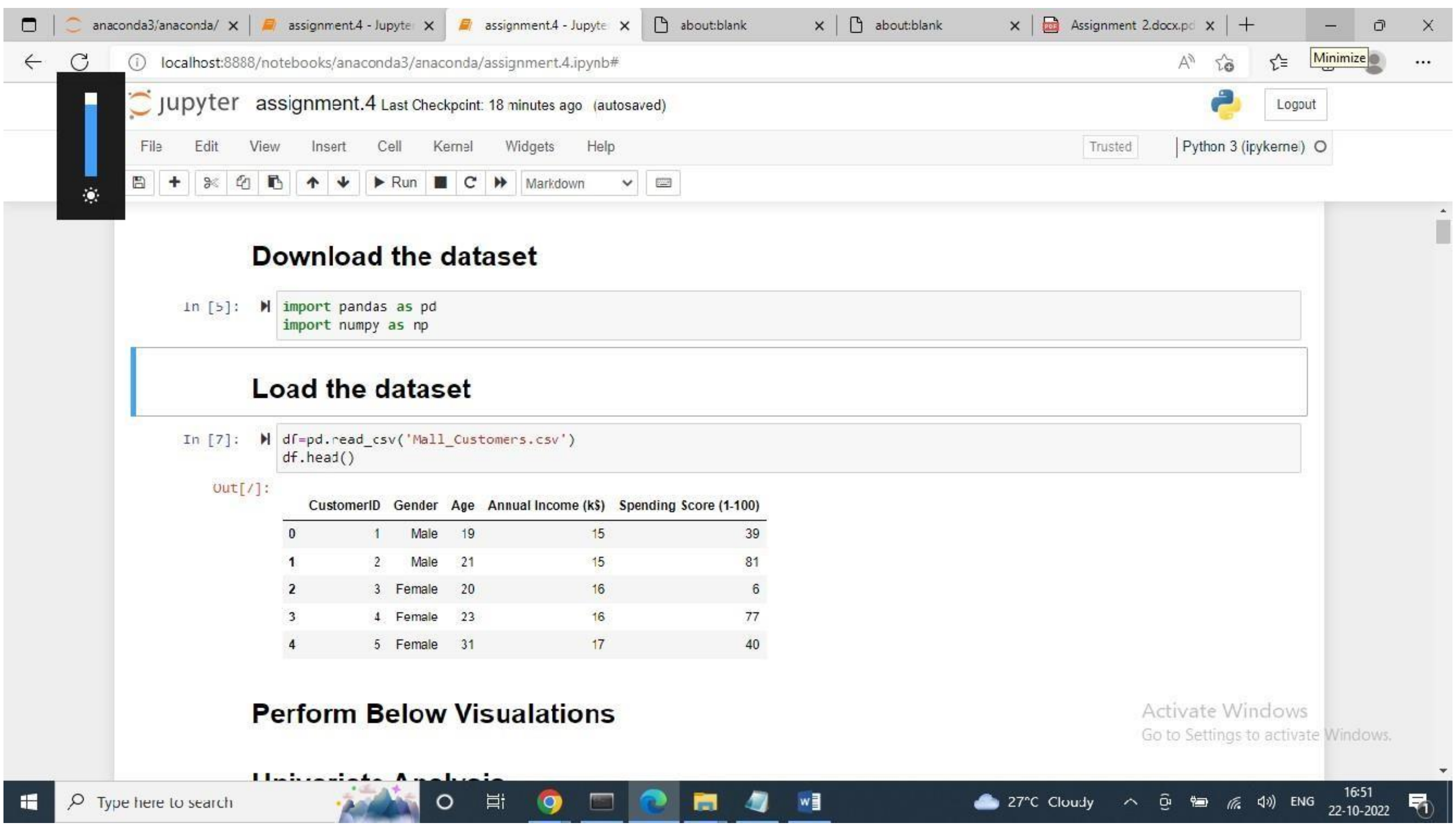
ASSIGNMENT-4

APPLIED DATA SCIENCE

Assignment date	22 October 2022
Student Name	**Abdul Quadir**
Student Roll Number	7309730919104001
Maximum Marks	2 Marks



Download the dataset

In [5]:
```
import pandas as pd
import numpy as np
```

Load the dataset

In [7]:
```
df=pd.read_csv('Mall_Customers.csv')
df.head()
```

Out[7]:

	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

Perform Below Visualations

localhost:8888/notebooks/anaconda3/anaconda/assignment.4.ipynb#

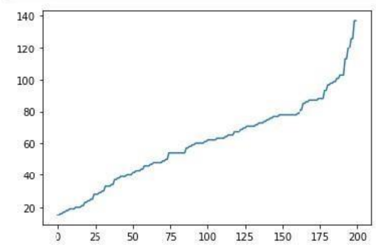
jupyter assignment.4 Last Checkpoint: 19 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help | Trusted | Python 3 (ipykernel)

Univariate Analysis

In [8]:
```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [11]:
```
plt.plot(df['Annual Income (k$)'])
plt.show()
```



In [10]:
```
data=np.array(df['Age'])
plt.plot(data,linestyle = 'dotted')
```

Out[10]: [<matplotlib.lines.Line2D at 0x26f3e956f10>]

Activate Windows
Go to Settings to activate Windows.

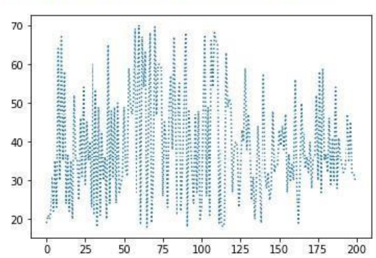
Type here to search 27°C Cloudy ENG 16:52 22-10-2022

localhost:8888/notebooks/anaconda3/anaconda/assignment.4.ipynb#

jupyter assignment.4 Last Checkpoint: 19 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help | Trusted | Python 3 (ipykernel)

Out[10]: [<matplotlib.lines.Line2D at 0x26f3e956f10>]



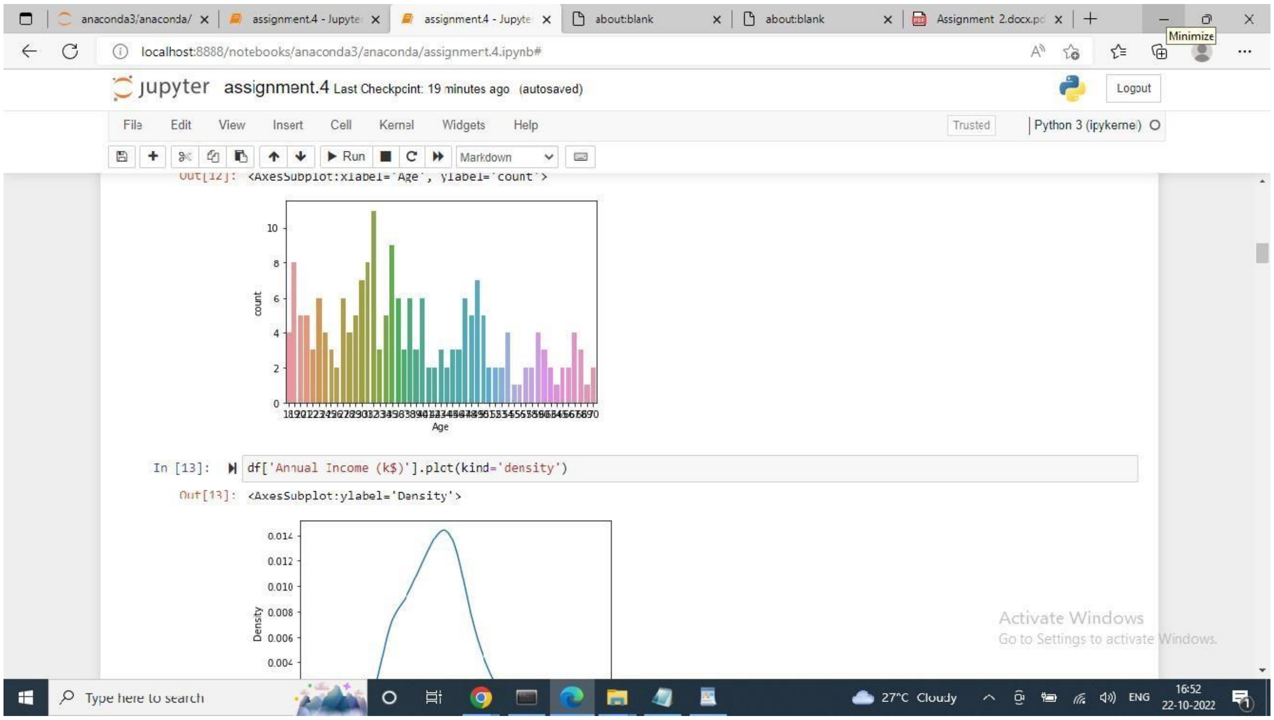
In [12]:
```
sns.countplot(df['Age'])
```

```
C:\Users\admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[12]: <AxesSubplot:xlabel='Age', ylabel='count'>

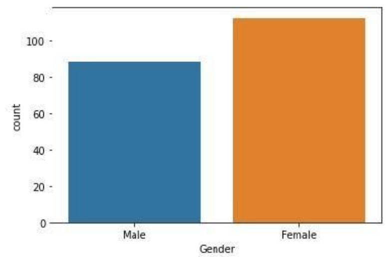
Activate Windows
Go to Settings to activate Windows.

Type here to search 27°C Cloudy ENG 16:52 22-10-2022


jupyter assignment.4 Last Checkpoint: 19 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Markdown
Out[12]: <AxesSubplot:xlabel='Age', ylabel='count'>
count
Age
In [13]: df['Annual Income (k$)'].plot(kind='density')
Out[13]: <AxesSubplot:ylabel='Density'>
Density
Activate Windows
Go to Settings to activate Windows.
Type here to search
27°C Cloudy
ENG
16:52
22-10-2022

```
d arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an expli
cit keyword will result in an error or misinterpretation.
  warnings.warn(
```

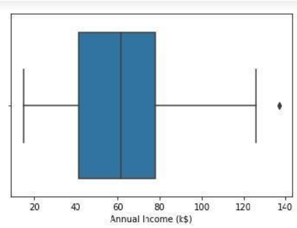
Out[14]: `<AxesSubplot:xlabel='Gender', ylabel='count'>`



In [15]:
```python
sns.boxplot(df['Annual Income (k$)'])
```

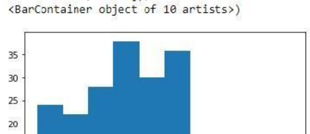
```
C:\Users\admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keywor
d arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an expli
cit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[15]: `<AxesSubplot:xlabel='Annual Income (k$)'>`



In [16]:
```python
plt.hist(df['Annual Income (k$)'])
```

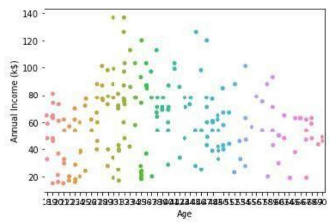
Out[16]:
```
(array([24., 22., 28., 38., 30., 36.,  8.,  6.,  4.,  4.]),
 array([ 15. ,  27.2,  39.4,  51.6,  63.8,  76. ,  88.2, 100.4, 112.6,
        124.8, 137. ]),
 <BarContainer object of 10 artists>)
```



Bivariate Analysis

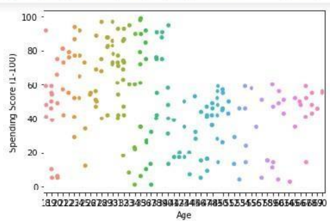
In [17]:
```python
sns.stripplot(x=df['Age'],y=df['Annual Income (k$)'])
```

Out[17]: <AxesSubplot:xlabel='Age', ylabel='Annual Income (k$)'>



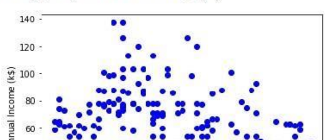
In [18]:
```python
sns.stripplot(x=df['Age'],y=df['Spending Score (1-100)'])
```

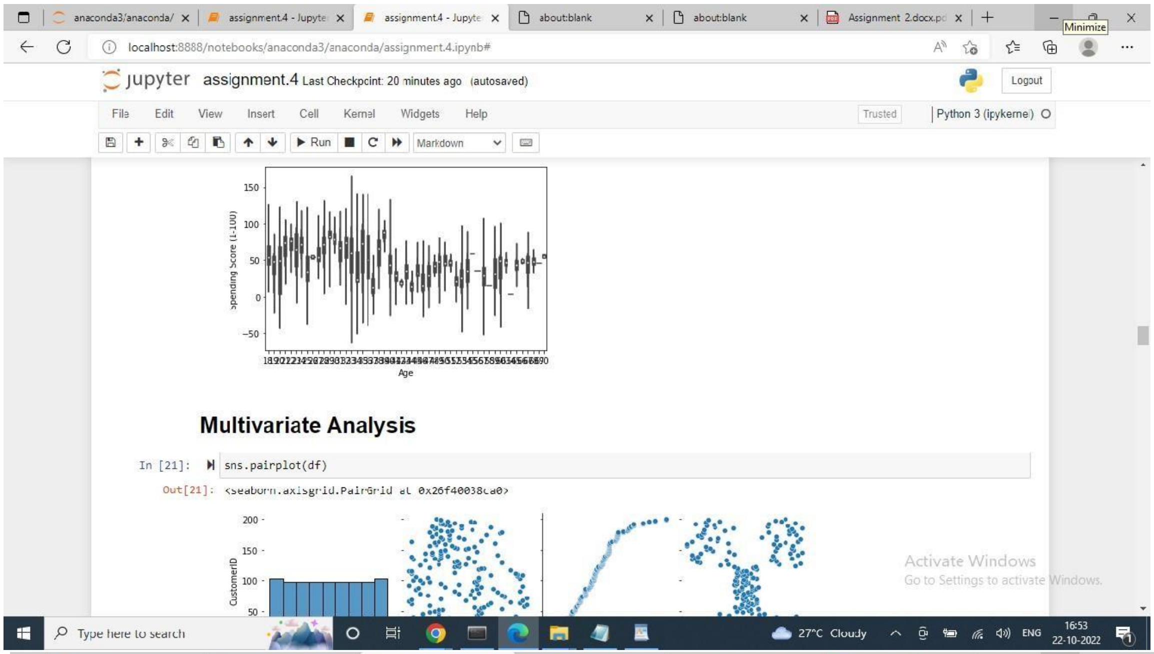
Out[18]: <AxesSubplot:xlabel='Age', ylabel='Spending Score (1-100)'>

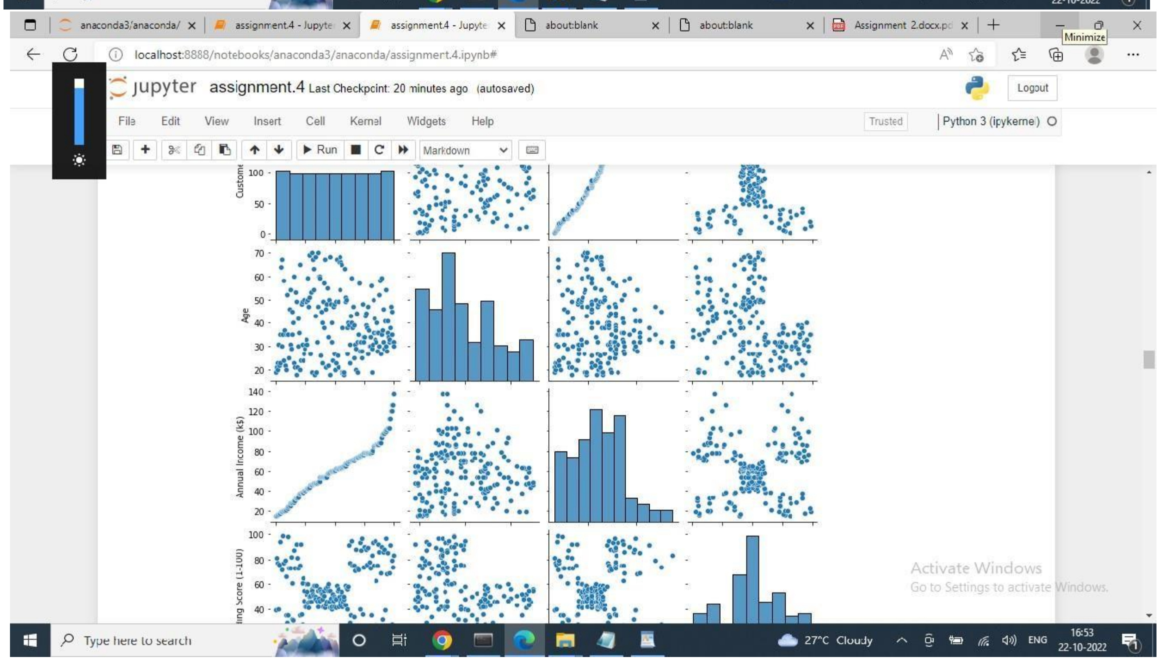


In [19]:
```python
plt.scatter(df['Age'],df['Annual Income (k$)'],color='blue')
plt.xlabel("Age")
plt.ylabel("Annual Income (k$)")
```

Out[19]: Text(0, 0.5, 'Annual Income (k$)')

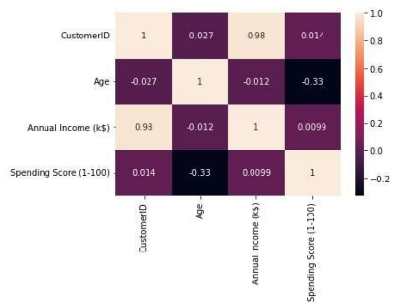



localhost:8888/notebooks/anaconda3/anaconda/assignment.4.ipynb#
jupyter assignment.4 Last Checkpoint: 20 minutes ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Spending Score (1-100)
Age
Multivariate Analysis
In [21]: sns.pairplot(df)
Out[21]: <seaborn.axisgrid.PairGrid at 0x26f40038ca0>
CustomerID
Activate Windows
Go to Settings to activate Windows.
Type here to search
27°C Cloudy
16:53
22-10-2022


localhost:8888/notebooks/anaconda3/anaconda/assignment.4.ipynb#
jupyter assignment.4 Last Checkpoint: 20 minutes ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Age
Annual Income (k$)
Activate Windows
Go to Settings to activate Windows.
Type here to search
27°C Cloudy
16:53
22-10-2022

Descriptive Statistics

In [22]:
```
sns.heatmap(df.corr(),annot=True)
```
Out[22]: <AxesSubplot:>



In [23]:
```
df.shape
```
Out[23]:
```
(200, 5)
```

In [24]:
```
df.isnull().sum()
```
Out[24]:
```
CustomerID                0
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

In [25]:
```
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   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
```

In [27]:
```
df.describe()
```
Out[27]:

	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)

In [29]:
```
df.median()
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_7908\530051474.py:1: FutureWarning: Dropping of nuisance columns in DataFrame re
ductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns
before calling the reduction.
  df.median()
```

Out[29]:
```
CustomerID                100.5
Age                        36.0
Annual Income (k$)         61.5
Spending Score (1-100)     50.0
dtype: float64
```

In [30]:
```
df.mode()
```

Out[30]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Female	32.0	54.0	42.0
1	2	NaN	NaN	78.0	NaN
2	3	NaN	NaN	NaN	NaN
3	4	NaN	NaN	NaN	NaN
4	5	NaN	NaN	NaN	NaN
...	...	...	...	...	...
195	196	NaN	NaN	NaN	NaN
196	197	NaN	NaN	NaN	NaN
197	198	NaN	NaN	NaN	NaN

Check For Missing Values

In [33]:
```
df.isna().sum()
```

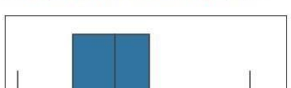
Out[33]:
```
CustomerID                0
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

Handling Outliers

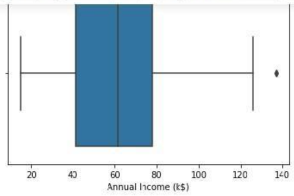
In [34]:
```
sns.boxplot(df['Annual Income (k$)'])
```

```
C:\Users\admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keywor
d arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an expli
cit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[34]:
```
<AxesSubplot:xlabel='Annual Income (k$)'>
```



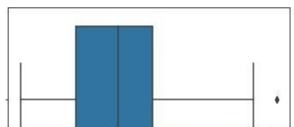
jupyter assignment.4 Last Checkpoint: 21 minutes ago (autosaved)



In [35]:
```
sns.boxplot(df['Annual Income (k$)'])
```

```
C:\Users\admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[35]: <AxesSubplot:xlabel='Annual Income (k$)'>



jupyter assignment.4 Last Checkpoint: 21 minutes ago (autosaved)

Encoding Categorial Values

In [37]:
```
numeric_data = df.select_dtypes(include=[np.number])
categorical_data = df.select_dtypes(exclude=[np.number])
print("Number of numerical variables: ", numeric_data.shape[1])
print("Number of categorical variables: ", categorical_data.shape[1])
```

```
Number of numerical variables:  4
Number of categorical variables:  1
```

In [38]:
```
print("Number of categorical variables: ", categorical_data.shape[1])
Categorical_variables = list(categorical_data.columns)
Categorical_variables
```

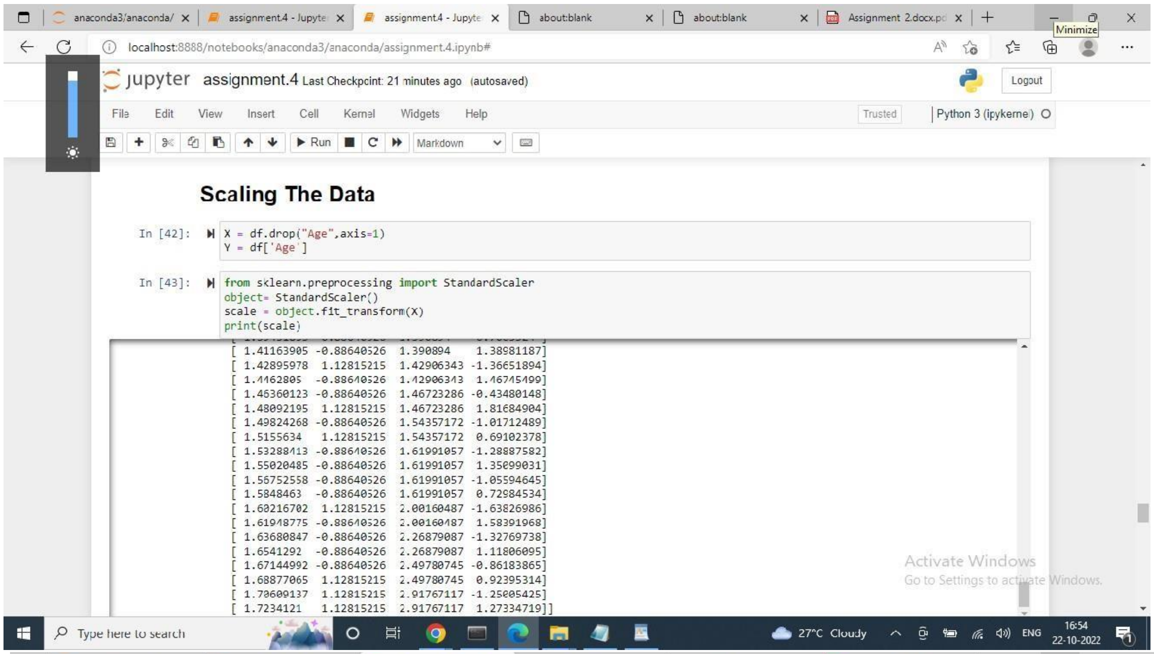
```
Number of categorical variables:  1
```

Out[38]: ['Gender']

In [39]:
```
df['Gender'].value_counts()
```

Out[39]:
```
Female    112
Male       88
Name: Gender, dtype: int64
```

In [40]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
label = le.fit_transform(df['Gender'])
df["Gender"] = label
```

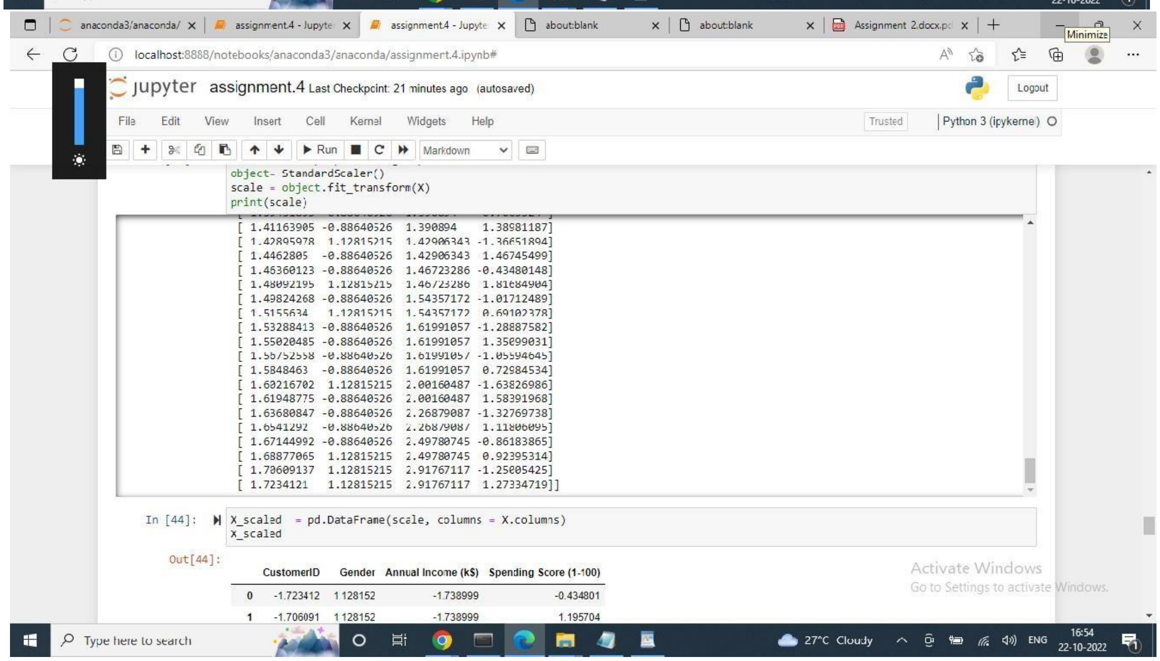


Scaling The Data

In [42]:
```
X = df.drop("Age",axis=1)
Y = df['Age']
```

In [43]:
```
from sklearn.preprocessing import StandardScaler
object= StandardScaler()
scale = object.fit_transform(X)
print(scale)
```

```
 [ 1.41163905 -0.88640526  1.390894    1.38581187]
 [ 1.42895978  1.12815215  1.42906343 -1.36651894]
 [ 1.4462805  -0.88640526  1.42906343  1.46745499]
 [ 1.46360123 -0.88640526  1.46723286 -0.43480148]
 [ 1.48092195  1.12815215  1.46723286  1.81684904]
 [ 1.49824268 -0.88640526  1.54357172 -1.01712489]
 [ 1.5155634   1.12815215  1.54357172  0.69102378]
 [ 1.53288413 -0.88640526  1.61991057 -1.28887582]
 [ 1.55020485 -0.88640526  1.61991057  1.35099031]
 [ 1.56752558 -0.88640526  1.61991057 -1.05594645]
 [ 1.5848463  -0.88640526  1.61991057  0.72984534]
 [ 1.60216702  1.12815215  2.00160487 -1.63826986]
 [ 1.61948775 -0.88640526  2.00160487  1.58391968]
 [ 1.63680847 -0.88640526  2.26879087 -1.32769738]
 [ 1.6541292  -0.88640526  2.26879087  1.11806095]
 [ 1.67144992 -0.88640526  2.49780745 -0.86183865]
 [ 1.68877065  1.12815215  2.49780745  0.92395314]
 [ 1.70609137  1.12815215  2.91767117 -1.25005425]
 [ 1.7234121   1.12815215  2.91767117  1.27334719]]
```




```
object= StandardScaler()
scale = object.fit_transform(X)
print(scale)
```

```
 [ 1.41163905 -0.88640526  1.390894    1.38581187]
 [ 1.42895978  1.12815215  1.42906343 -1.36651894]
 [ 1.4462805  -0.88640526  1.42906343  1.46745499]
 [ 1.46360123 -0.88640526  1.46723286 -0.43480148]
 [ 1.48092195  1.12815215  1.46723286  1.81684904]
 [ 1.49824268 -0.88640526  1.54357172 -1.01712489]
 [ 1.5155634   1.12815215  1.54357172  0.69102378]
 [ 1.53288413 -0.88640526  1.61991057 -1.28887582]
 [ 1.55020485 -0.88640526  1.61991057  1.35099031]
 [ 1.56752558 -0.88640526  1.61991057 -1.05594645]
 [ 1.5848463  -0.88640526  1.61991057  0.72984534]
 [ 1.60216702  1.12815215  2.00160487 -1.63826986]
 [ 1.61948775 -0.88640526  2.00160487  1.58391968]
 [ 1.63680847 -0.88640526  2.26879087 -1.32769738]
 [ 1.6541292  -0.88640526  2.26879087  1.11806095]
 [ 1.67144992 -0.88640526  2.49780745 -0.86183865]
 [ 1.68877065  1.12815215  2.49780745  0.92395314]
 [ 1.70609137  1.12815215  2.91767117 -1.25005425]
 [ 1.7234121   1.12815215  2.91767117  1.27334719]]
```


In [44]:
```
X_scaled  = pd.DataFrame(scale, columns = X.columns)
X_scaled
```

Out[44]:

	CustomerID	Gender	Annual Income (k$)	Spending Score (1-100)
0	-1.723412	1.128152	-1.738999	-0.434801
1	-1.706091	1.128152	-1.738999	1.195704

jupyter assignment.4 Last Checkpoint: 21 minutes ago (autosaved)

```
In [44]: X_scaled = pd.DataFrame(scale, columns = X.columns)
         X_scaled
```

Out[44]:

	CustomerID	Gender	Annual Income (k$)	Spending Score (1-100)
0	-1.723412	1.128152	-1.738999	-0.434801
1	-1.706091	1.128152	-1.738999	1.195704
2	-1.688771	-0.886405	-1.700830	-1.715913
3	-1.671450	-0.886405	-1.700830	1.040418
4	-1.654129	-0.886405	-1.662660	-0.395980
...	...	...	...	...
195	1.654129	-0.886405	2.268791	1.118061
196	1.671450	-0.886405	2.497807	-0.861839
197	1.688771	1.128152	2.497807	0.923953
198	1.706091	1.128152	2.917671	-1.250054
199	1.723412	1.128152	2.917671	1.273347

200 rows × 4 columns

```
In [45]: #train test split
         from sklearn.model_selection import train_test_split
         # split the dataset
         X_train, X_test, Y_train, Y_test = train_test_split(X_scaled, Y, test_size=0.20, random_state=0)
```

jupyter assignment.4 Last Checkpoint: 21 minutes ago (autosaved)

```
In [48]: X_train.shape
Out[48]: (160, 4)

In [49]: X_test.shape
Out[49]: (40, 4)

In [50]: Y_train.shape
Out[50]: (160,)

In [51]: Y_test.shape
Out[51]: (40,)
```

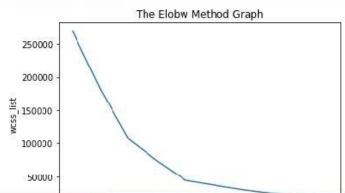
#clustering algorithm

```
In [52]: x = df.iloc[:, [3, 4]].values

In [53]: #finding optimal number of clusters using the elbow method
         from sklearn.cluster import KMeans
         wcss_list= []  #Initializing the list for the values of WCSS

         #Using for loop for iterations from 1 to 10.
         for i in range(1, 11):
             kmeans = KMeans(n_clusters=i, init='k-means++', random_state= 42)
```

jupyter assignment.4 Last Checkpoint: 21 minutes ago (autosaved)

```
from sklearn.cluster import KMeans
wcss_list= []  #Initializing the list for the values of WCSS

#Using for loop for iterations from 1 to 10.
for i in range(1, 11):
    kmeans = KMeans(n_clusters=1, init='k-means++', random_state= 42)
    kmeans.fit(x)
    wcss_list.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss_list)
plt.title('The Elobw Method Graph')
plt.xlabel('Number of clusters(k)')
plt.ylabel('wcss_list')
plt.show()
```

```
C:\Users\admin\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1036: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```



jupyter assignment.4 Last Checkpoint: 21 minutes ago (autosaved)

```
kmeans = KMeans(n_clusters=5, init='k-means++', random_state= 42)
y_predict= kmeans.fit_predict(x)
```

In [56]:

```
#visulaizing the clusters
plt.scatter(x[y_predict == 0, 0], x[y_predict == 0, 1], s = 100, c = 'blue', label = 'Cluster 1') #for first cluster
plt.scatter(x[y_predict == 1, 0], x[y_predict == 1, 1], s = 100, c = 'green', label = 'Cluster 2') #for second cluster
plt.scatter(x[y_predict== 2, 0], x[y_predict == 2, 1], s = 100, c = 'red', label = 'Cluster 3') #for third cluster
plt.scatter(x[y_predict == 3, 0], x[y_predict == 3, 1], s = 100, c = 'cyan', label = 'Cluster 4') #for fourth cluster
plt.scatter(x[y_predict == 4, 0], x[y_predict == 4, 1], s = 100, c = 'magenta', label = 'Cluster 5') #for fifth cluster
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 300, c = 'yellow', label = 'Centroid')
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```