

ASSIGNMENT - 4

MAHENDRA ENGINEERING COLLEGE FOR WOMEN

NAME : SHANMUGAPRIYA R

CLASS : 4th YEAR ECE

SUBJECT : IBM

REGISTER NO : 611419106058

1.importlibraries

#importlibrary

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

2.Load the dataset

#load dataset

```
from google.colab import
filesupload=files.upload()
```

<IPython.core.display.HTMLObject>

Saving Mall_Customers.xlsx to Mall_Customers

(1).xlsxcustomer=pd.read_excel("Mall_Customers.xlsx")

3.Univariate Analysisdf=pd.read_excel("Mall_Customers.xlsx")

#view first five rows of DataFrame

df.head()

	CustomerID	Gender	Age	AnnualIncome(k\$)	SpendingScore(1-100)
0	1.0	Male	19.0		15.0
39.0					
1	2.0	Male	21.0		15.0
81.0					
2	3.0	Female	20.0		16.0
6.0					
3	4.0	Female	23.0		16.0
77.0					
4	5.0	Female	31.0		17.0
40.0					

```
#calculate mean of Annual Income(k$)
df["Annual Income (k$)"].mean()60.56
```

```
#calculate median of Annual Income(k$)
df["Annual Income (k$)"].median()61.5
```

```
#calculate standard deviation of Annual Income(k$)
df["Annual Income(k$)"].std()26.2647211
```

```
65271244
```

```
#calculate mode of Annual Income(k$)
df["Annual Income(k$)"].mode()
```

```
0      54.0
1      78.0
dtype:float64
```

```
#create frequency table for 'Annual Income(k$)'
df["Annual Income (k$)"].value_counts()
```

```
54.0      12
78.0      12
48.0       6
71.0       6
63.0       6
..
58.0       2
59.0       2
16.0       2
64.0       2
137.0      2
```

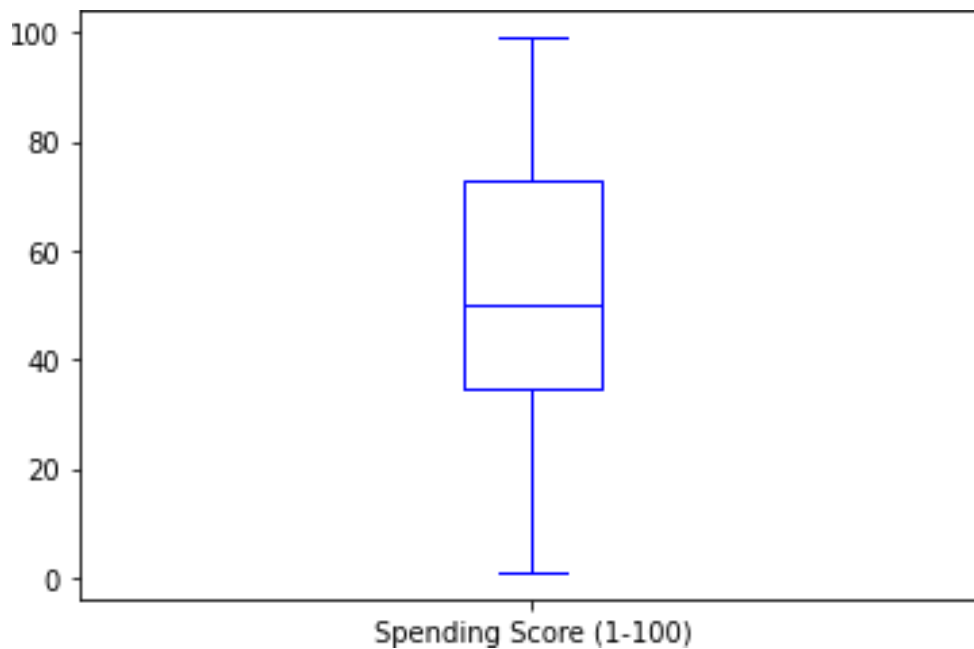
```
Name: Annual Income(k$), Length: 64, dtype:int64
```

```
#view last five rows of DataFrame
df.tail()
```

	CustomerID	Gender	Age	Annual	Income (k\$)	Spending	Score(1-
100)							
195	196.0	Female	35.0		120.0		
79.0							
196	197.0	Female	45.0		126.0		
28.0							
197	198.0	Male	32.0		126.0		
74.0							
198	199.0	Male	32.0		137.0		
18.0							
199	200.0	Male	30.0		137.0		
83.0							

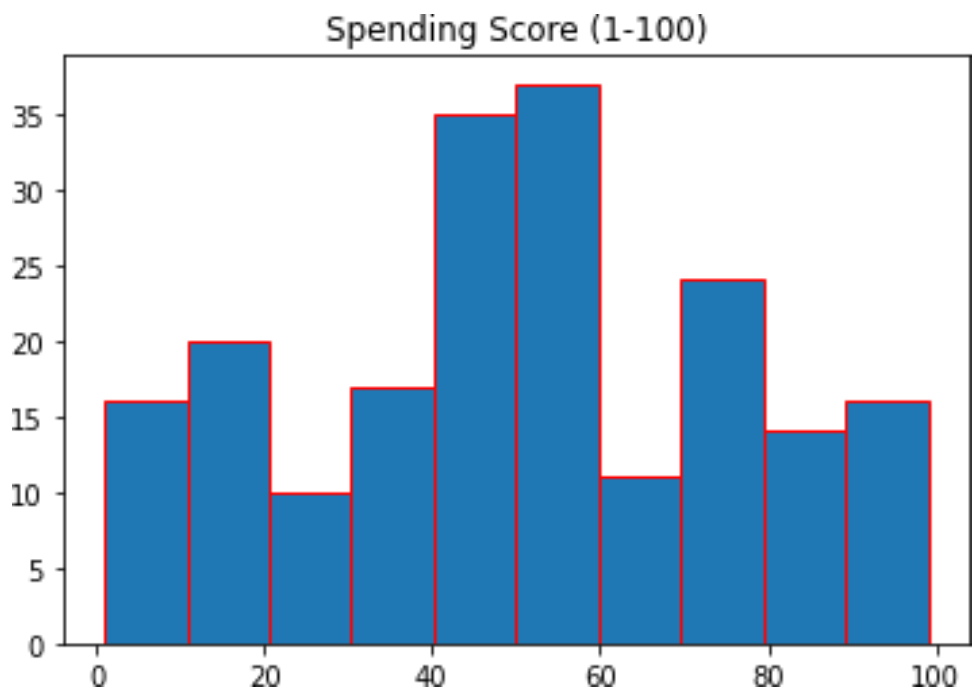
```
#create a boxplot for the
'Spending Score' variableimport matplotlib.pyplot as pltcustomer.boxplot(col
umn=['Spending Score(1-100)'],grid=False,color='blue')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc3b924e850>
```



#to create histogram for the 'Spending Score' variable
`customer.hist(column='Spending Score(1-100)',grid=False,edgecolor='red')`

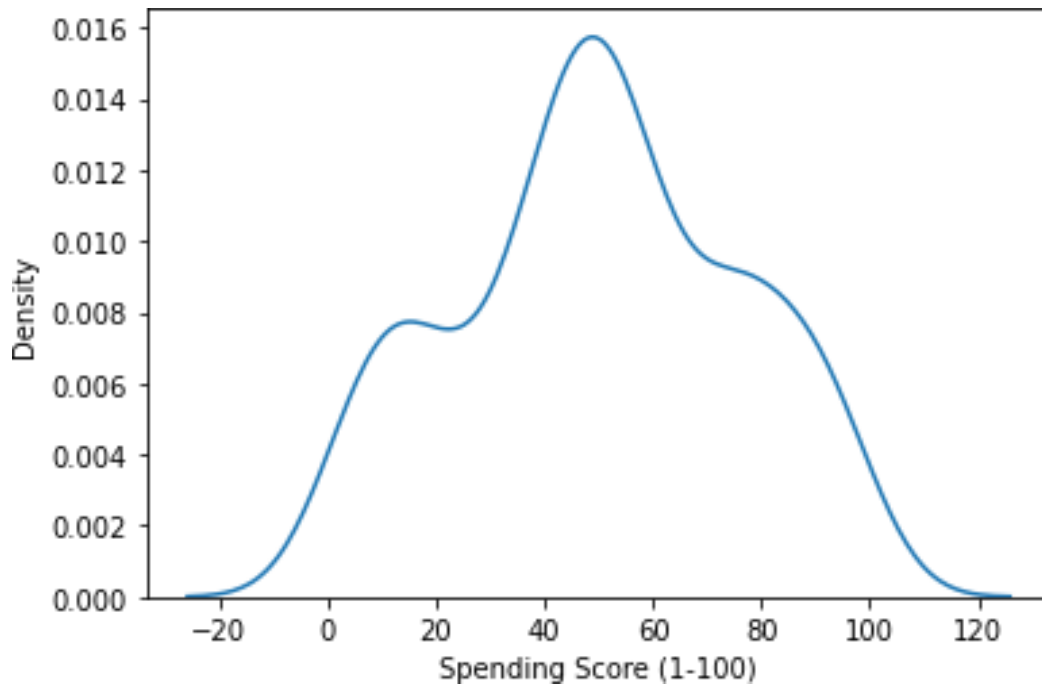
`array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7fc3b982a490>]],
dtype=object)`



```
#to create a density curve for the 'Spending Score' variable
```

```
sns.kdeplot(customer['Spending Score(1-100)'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc3b9255f10>
```



```
#information of dataset
```

```
customer.info()
```

```
<class
```

```
'pandas.core.frame.DataFrame'>RangeIndex:200e
```

```
ntries,0 to 199 Data columns (total 5 columns):
```

#	Column	Non-Null Count	Dtype
0	CustomerID	200 non-null	float64
1	Gender	200 non-null	object
2	Age	200 non-null	float64
3	Annual Income (k\$)	200 non-null	float64
4	Spending Score (1-100)	200 non-null	float64

```
dtypes: float64(4),
```

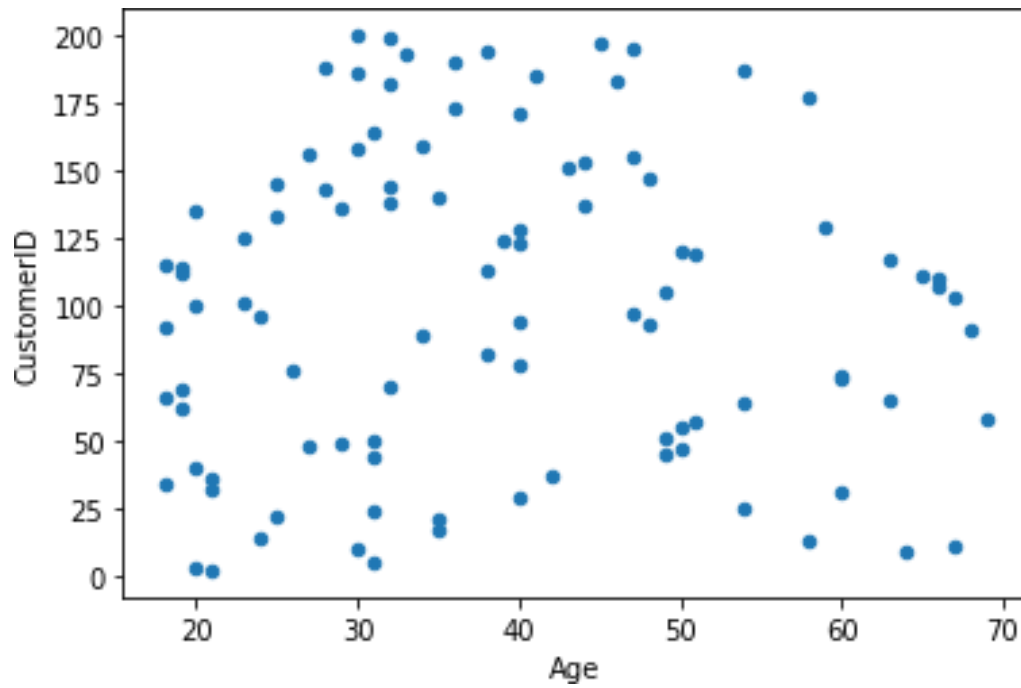
```
object(1) memory usage: 7.9+KB
```

4. Bi-Variate Analysis

```
#Scatter Plot
```

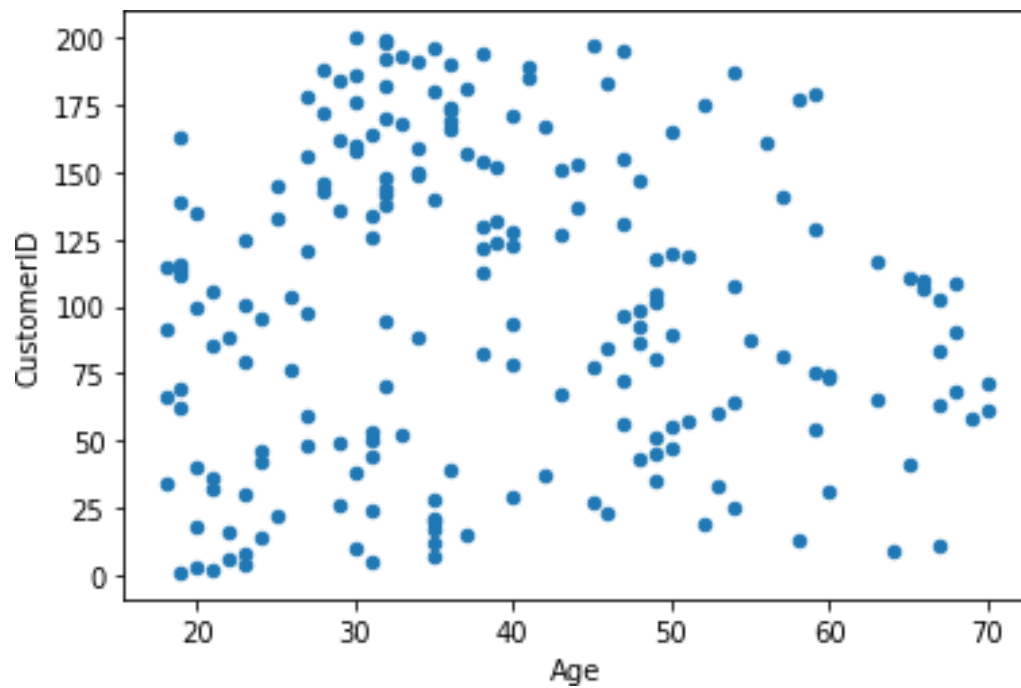
```
customer[customer['Spending Score(1-100)']<100].sample(100).plot.scatter(x='Age',y='CustomerID')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc3b8f1e4d0>
```



```
customer[customer['Spending Score(1-100)']<100].plot.scatter(x='Age',y='CustomerID')
```

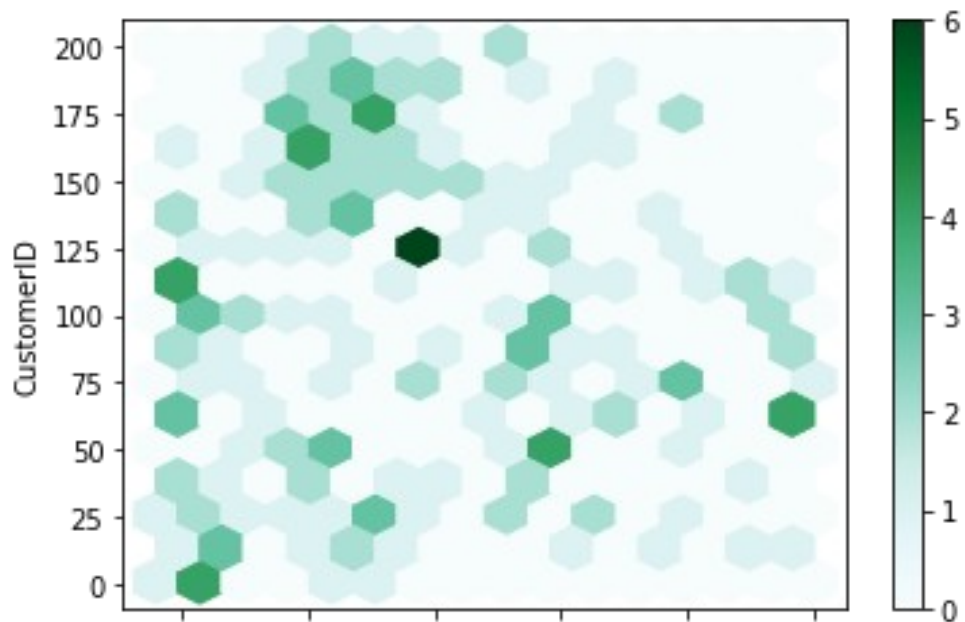
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc3b8eb2b10>
```



#HexPlot

```
customer[customer['Spending Score(1-100)']<100].plot.hexbin(x='Age',y='CustomerID', gridsize=15)
```

```
<matplotlib.axes._subplots.AxesSubplotat0x7fc3b8dc0310>
```



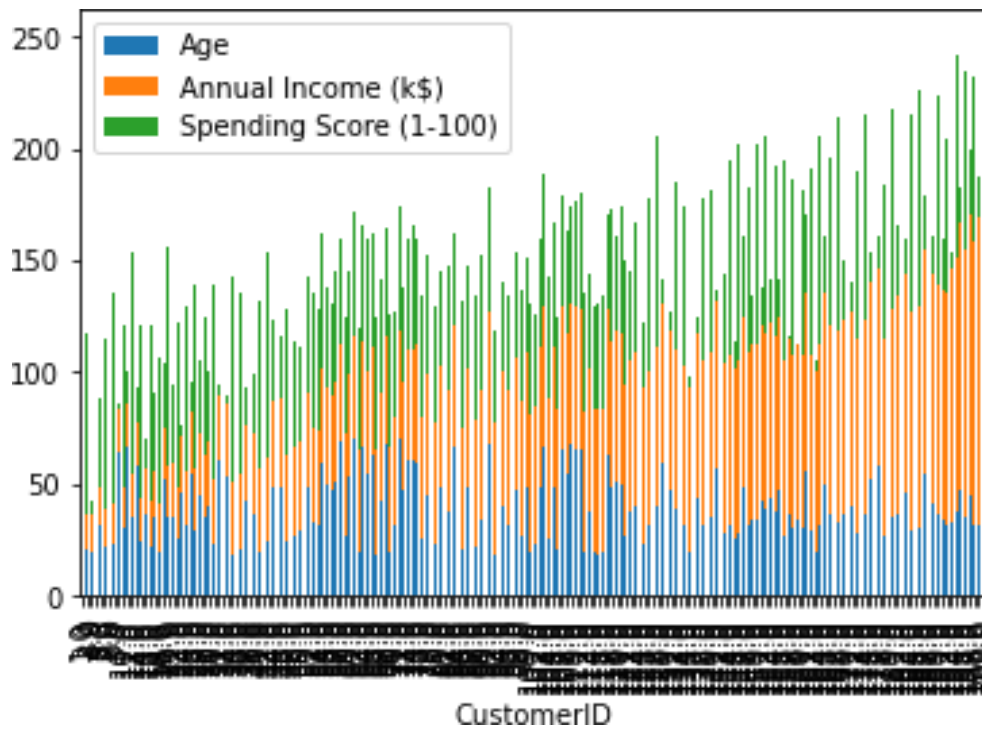
#stackedplot

```
customer_count=pd.read_excel("Mall_Customers.xlsx",index_col=0)customer_count.head()
```

CustomerID	Gender	Age	Annual Income(k\$)	Spending	Score	(1-100)
1.0	Male	19.0	15.0			39.0
2.0	Male	21.0	15.0			81.0
3.0	Female	20.0	16.0			6.0
4.0	Female	23.0	16.0			77.0
5.0	Female	31.0	17.0			40.0

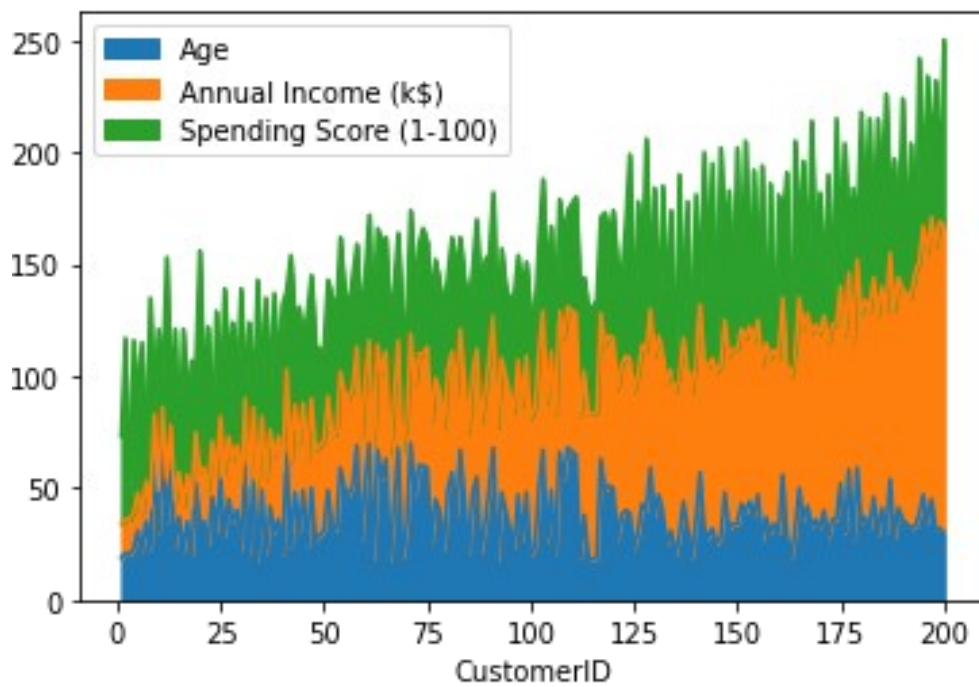
```
customer_count.plot.bar(stacked=True)
```

```
<matplotlib.axes._subplots.AxesSubplotat0x7fc3b8ead250>
```



```
customer_count.plot.area()
```

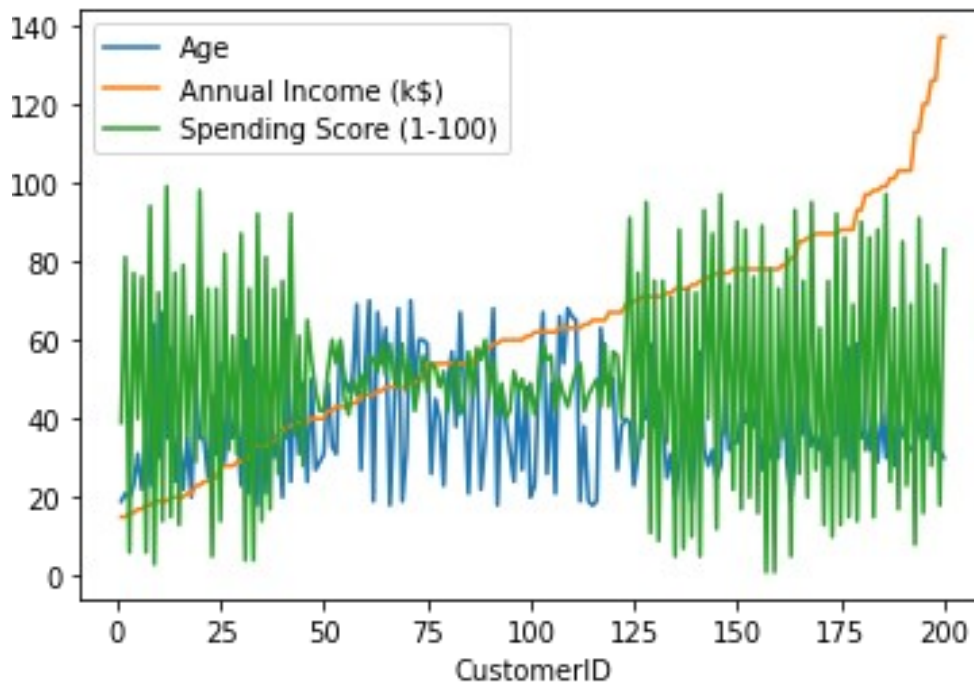
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc3b832b8d0>
```



#Bivariate linechart

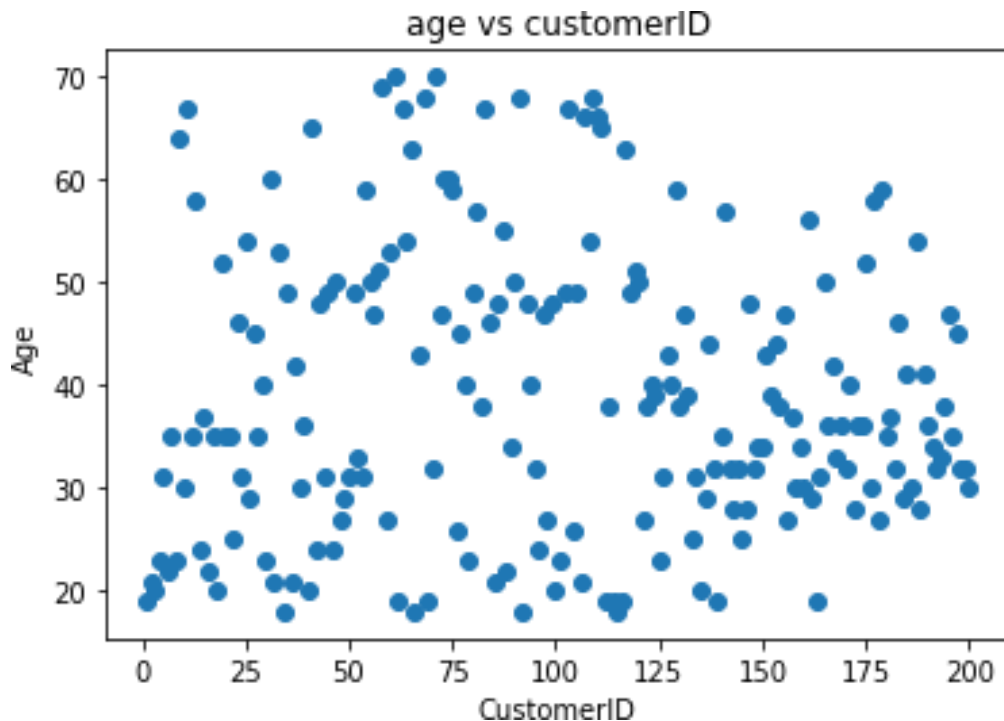
```
customer_count.plot.line()
```

<matplotlib.axes._subplots.AxesSubplot at 0x7fc3b8338290>



#createscatterplot

```
ofAnnualIncomevsSpendingScoreplt.scatter(customer.CustomerID,custo  
mer.Age)plt.title('agevscustomerID')plt.xlabel('CustomerID')  
plt.ylabel('Age')Text(0,0.5,  
'Age')
```

```
#createcorrelationmatrix
customer.corr()
```

	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	
SpendingScore(1-100)	0.013835	-0.327227		0.009903	

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

```
importstatsmodels.apiassm
```

```
#defineresponsevariable
y=customer['CustomerID']
```

```
#defineresponsevariable
x=customer['Age']
```

```
#add constanttopredictorvariables
x=sm.add_constant(x)
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142:FutureWarning: In
a future version of pandasallargumentsofconcat exceptfor theargument 'objs'will bekeyword
```

```
-
```

```
only
x= pd.concat(x[:,order],1)
```

```
#fit linearregressionmodel
model=sm.OLS(y,x).fit()
```

```
#viewmodelsummary
print(model.summary())
```

OLSRegression Results

```
=====
=====
```

Dep.Variable:	CustomerID	R-squared:
0.001		
Model:	OLS	Adj. R-squared:
-0.004		
Method:	LeastSquares	F-statistic:
0.1419		
Date:	Sat, 22 Oct 2022	Prob (F-statistic):
0.707		
Time:	14:53:57	Log-Likelihood:
-1094.9		
No.Observations:	200	AIC:
2194.		
DfResiduals:	198	BIC:
2200.		
DfModel:	1	

Covariance Type: nonrobust

```
=====
=====
```

	coef	std err	t	P> t	[0.025
0.975]					

const	104.8081	12.149	8.627	0.000	80.850
128.766					
Age	-0.1109	0.294	-0.377	0.707	-0.691
0.470					

```
=====
=====
```

Omnibus:	84.500	Durbin-Watson:
0.002		
Prob(Omnibus):	0.000	Jarque-Bera(JB):
11.691		
Skew:	-0.014	Prob(JB):
0.00289		
Kurtosis:	1.816	Cond.No.

122.

=====

Notes:

[1]StandardErrors assumethat thecovariancematrixofthe errorsiscorrectlyspecified.

5. Multi-Variate Analysis

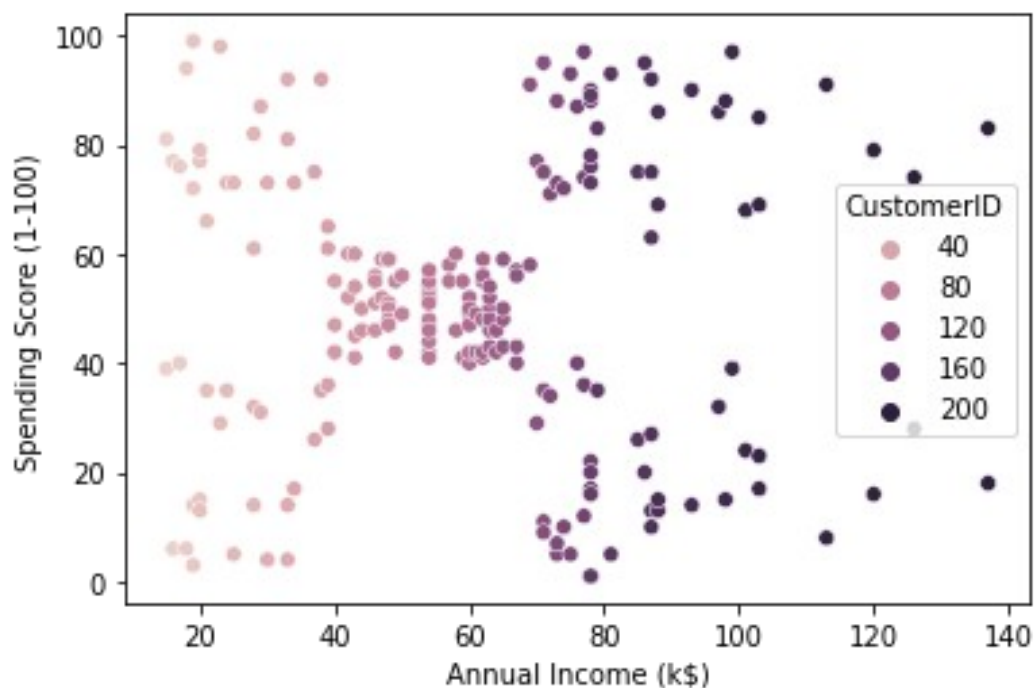
```
sns.scatterplot(customer["Annual Income (k$)"],customer["SpendingScore(1-100)"],hue=customer["CustomerID"])
```

/usr/local/lib/python3.7/dist-

packages/seaborn/_decorators.py:43:FutureWarning:Passthefollowingvariablesaskeyworddargs: x,y.Fromversion0.12,theonlyvalidpositionalargumentwillbe`data`,andpassing otherargumentswithout an explicitkeyword willresult inan errorormisinterpretation.

FutureWarning

<matplotlib.axes._subplots.AxesSubplotat0x7fc3ac87c410>



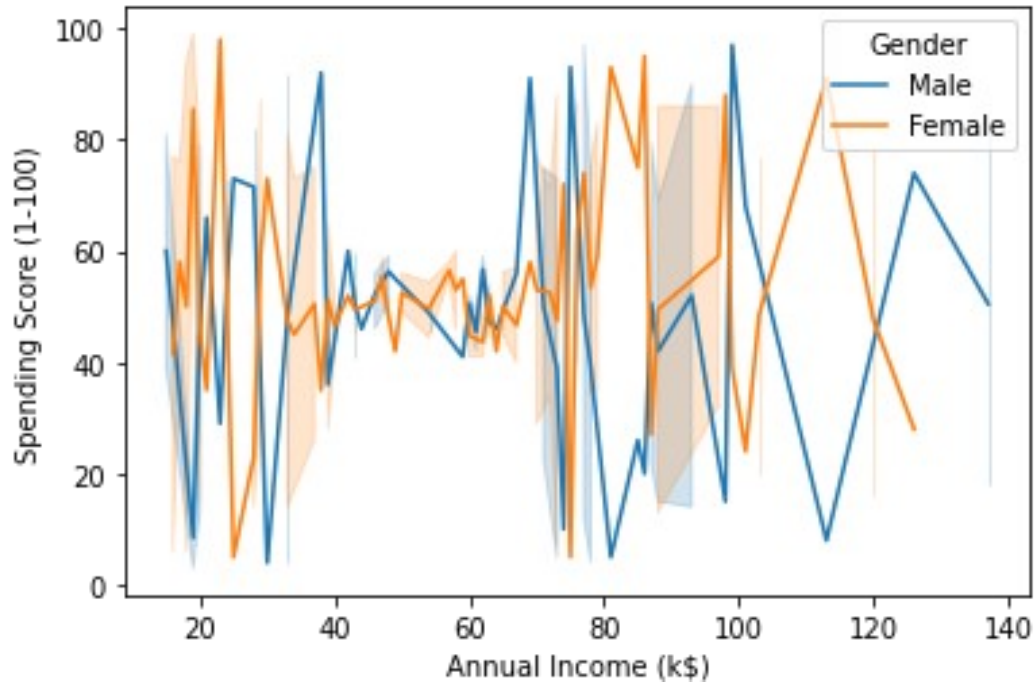
```
sns.lineplot(customer["Annual Income (k$)"],customer["Spending Score(1-100)"],hue=customer["Gender"])
```

/usr/local/lib/python3.7/dist-

packages/seaborn/_decorators.py:43:FutureWarning:Passthefollowingvariablesaskeyworddargs: x,y.Fromversion0.12,theonlyvalidpositionalargumentwillbe`data`,andpassing otherargumentswithout an explicitkeyword willresult in

an error or misinterpretation.
FutureWarning

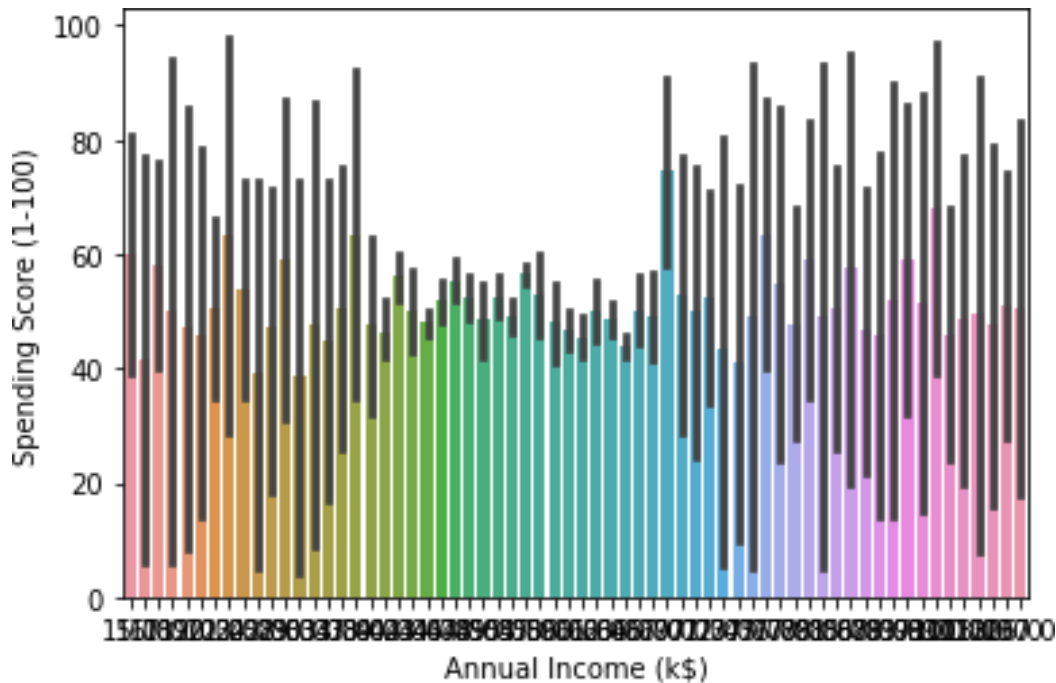
<matplotlib.axes._subplots.AxesSubplot at 0x7fc3ac4aee90>



```
sns.barplot(customer["Annual Income (k$)"],customer["Spending Score(1-100)"])
```

/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:FutureWarning:Passthefollowingvariablesaskeywordargs: x,y.Fromversion0.12,theonlyvalidpositionalargumentwillbe`data`,andpassing otherargumentswithout an explicitkeyword willresult inan errorormisinterpretation.
FutureWarning

<matplotlib.axes._subplots.AxesSubplot at 0x7fc3ac102f10>



```
customer.skew()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1:FutureWarning: Dropping of
nuisance columns in DataFrame reductions(with'numeric_only=None')isdeprecated;in
afutureversionthiswillraiseTypeError.Selectonly validcolumnsbeforecallingthereduction.
"""Entry point forlaunching anIPython kernel.
```

```
CustomerID          0.000000
Age                 0.485569
AnnualIncome(k$)    0.321843
SpendingScore(1-100) -
0.047220dtype:float64
```

```
label=df.CustomerID.value_counts().indexcount=df.CustomerID.value_counts().values
```

```
plt.pie(count,labels=label)([<matplotlib.patches.Wedgeat0x7fc3a615
```

```
b150>,
<matplotlib.patches.Wedgeat0x7fc3a615b610>,
<matplotlib.patches.Wedgeat0x7fc3a615bd50>,
<matplotlib.patches.Wedgeat0x7fc3a61643d0>,
<matplotlib.patches.Wedgeat0x7fc3a6164cd0>,
<matplotlib.patches.Wedgeat0x7fc3a6176450>,
<matplotlib.patches.Wedgeat0x7fc3a6176890>,
<matplotlib.patches.Wedgeat0x7fc3a6176210>,
<matplotlib.patches.Wedgeat0x7fc3a6164a10>,
```

<matplotlib.patches.Wedgeat0x7fc3a6176e90>,
<matplotlib.patches.Wedgeat0x7fc3a615b110>,
<matplotlib.patches.Wedgeat0x7fc3a61f5650>,
<matplotlib.patches.Wedgeat0x7fc3a61f5c90>,
<matplotlib.patches.Wedgeat0x7fc3a61fd310>,
<matplotlib.patches.Wedgeat0x7fc3a61fd950>,
<matplotlib.patches.Wedgeat0x7fc3a61fdf90>,
<matplotlib.patches.Wedgeat0x7fc3a620e610>,
<matplotlib.patches.Wedgeat0x7fc3a620ec50>,
<matplotlib.patches.Wedgeat0x7fc3a611d2d0>,
<matplotlib.patches.Wedgeat0x7fc3a611d910>,
<matplotlib.patches.Wedgeat0x7fc3a611df50>,
<matplotlib.patches.Wedgeat0x7fc3a61255d0>,
<matplotlib.patches.Wedgeat0x7fc3a6125c10>,
<matplotlib.patches.Wedgeat0x7fc3a612e290>,
<matplotlib.patches.Wedgeat0x7fc3a612e8d0>,
<matplotlib.patches.Wedgeat0x7fc3a612ef10>,
<matplotlib.patches.Wedgeat0x7fc3a6136590>,
<matplotlib.patches.Wedgeat0x7fc3a6136bd0>,
<matplotlib.patches.Wedgeat0x7fc3a613f250>,
<matplotlib.patches.Wedgeat0x7fc3a613f890>,
<matplotlib.patches.Wedgeat0x7fc3a613fed0>,
<matplotlib.patches.Wedgeat0x7fc3a6148550>,
<matplotlib.patches.Wedgeat0x7fc3a6148b90>,
<matplotlib.patches.Wedgeat0x7fc3a6150210>,
<matplotlib.patches.Wedgeat0x7fc3a6150850>,
<matplotlib.patches.Wedgeat0x7fc3a6150e90>,
<matplotlib.patches.Wedgeat0x7fc3a6159510>,
<matplotlib.patches.Wedgeat0x7fc3a6159b50>,
<matplotlib.patches.Wedgeat0x7fc3a60e21d0>,
<matplotlib.patches.Wedgeat0x7fc3a60e2810>,
<matplotlib.patches.Wedgeat0x7fc3a60e2e50>,
<matplotlib.patches.Wedgeat0x7fc3a60ea4d0>,
<matplotlib.patches.Wedgeat0x7fc3a60eab10>,
<matplotlib.patches.Wedgeat0x7fc3a60f5190>,
<matplotlib.patches.Wedgeat0x7fc3a60f57d0>,
<matplotlib.patches.Wedgeat0x7fc3a60f5e10>,
<matplotlib.patches.Wedgeat0x7fc3a60fe490>,
<matplotlib.patches.Wedgeat0x7fc3a60fead0>,
<matplotlib.patches.Wedgeat0x7fc3a610a150>,
<matplotlib.patches.Wedgeat0x7fc3a610a790>,
<matplotlib.patches.Wedgeat0x7fc3a610add0>,
<matplotlib.patches.Wedgeat0x7fc3a6112450>,
<matplotlib.patches.Wedgeat0x7fc3a6112a90>,
<matplotlib.patches.Wedgeat0x7fc3a609f110>,
<matplotlib.patches.Wedgeat0x7fc3a609f750>,
<matplotlib.patches.Wedgeat0x7fc3a609fd90>,
<matplotlib.patches.Wedgeat0x7fc3a60aa410>,
<matplotlib.patches.Wedgeat0x7fc3a60aaa50>,
<matplotlib.patches.Wedgeat0x7fc3a60b40d0>,

<matplotlib.patches.Wedgeat0x7fc3a60b4710>,
<matplotlib.patches.Wedgeat0x7fc3a60b4d50>,
<matplotlib.patches.Wedgeat0x7fc3a60be3d0>,
<matplotlib.patches.Wedgeat0x7fc3a60bea10>,
<matplotlib.patches.Wedgeat0x7fc3a60c9090>,
<matplotlib.patches.Wedgeat0x7fc3a60c96d0>,
<matplotlib.patches.Wedgeat0x7fc3a60c9d10>,
<matplotlib.patches.Wedgeat0x7fc3a60d3390>,
<matplotlib.patches.Wedgeat0x7fc3a60d39d0>,
<matplotlib.patches.Wedgeat0x7fc3a605e050>,
<matplotlib.patches.Wedgeat0x7fc3a605e690>,
<matplotlib.patches.Wedgeat0x7fc3a605ecd0>,
<matplotlib.patches.Wedgeat0x7fc3a606b350>,
<matplotlib.patches.Wedgeat0x7fc3a606b990>,
<matplotlib.patches.Wedgeat0x7fc3a606bfd0>,
<matplotlib.patches.Wedgeat0x7fc3a6076650>,
<matplotlib.patches.Wedgeat0x7fc3a6076c90>,
<matplotlib.patches.Wedgeat0x7fc3a607e310>,
<matplotlib.patches.Wedgeat0x7fc3a607e950>,
<matplotlib.patches.Wedgeat0x7fc3a607ef90>,
<matplotlib.patches.Wedgeat0x7fc3a6088610>,
<matplotlib.patches.Wedgeat0x7fc3a6088c50>,
<matplotlib.patches.Wedgeat0x7fc3a60952d0>,
<matplotlib.patches.Wedgeat0x7fc3a6095910>,
<matplotlib.patches.Wedgeat0x7fc3a6095f50>,
<matplotlib.patches.Wedgeat0x7fc3a601e5d0>,
<matplotlib.patches.Wedgeat0x7fc3a601ec10>,
<matplotlib.patches.Wedgeat0x7fc3a6029290>,
<matplotlib.patches.Wedgeat0x7fc3a60298d0>,
<matplotlib.patches.Wedgeat0x7fc3a6029ed0>,
<matplotlib.patches.Wedgeat0x7fc3a6034550>,
<matplotlib.patches.Wedgeat0x7fc3a6034b90>,
<matplotlib.patches.Wedgeat0x7fc3a603e210>,
<matplotlib.patches.Wedgeat0x7fc3a603e850>,
<matplotlib.patches.Wedgeat0x7fc3a603ee90>,
<matplotlib.patches.Wedgeat0x7fc3a604a510>,
<matplotlib.patches.Wedgeat0x7fc3a604ab50>,
<matplotlib.patches.Wedgeat0x7fc3a60541d0>,
<matplotlib.patches.Wedgeat0x7fc3a6054810>,
<matplotlib.patches.Wedgeat0x7fc3a6054e50>,
<matplotlib.patches.Wedgeat0x7fc3a5fdf4d0>,
<matplotlib.patches.Wedgeat0x7fc3a5fdfb10>,
<matplotlib.patches.Wedgeat0x7fc3a5fea190>,
<matplotlib.patches.Wedgeat0x7fc3a5fea7d0>,
<matplotlib.patches.Wedgeat0x7fc3a5feae10>,
<matplotlib.patches.Wedgeat0x7fc3a5ff4490>,
<matplotlib.patches.Wedgeat0x7fc3a5ff4ad0>,
<matplotlib.patches.Wedgeat0x7fc3a6001150>,
<matplotlib.patches.Wedgeat0x7fc3a6001790>,
<matplotlib.patches.Wedgeat0x7fc3a6001dd0>,

<matplotlib.patches.Wedgeat0x7fc3a6009450>,
<matplotlib.patches.Wedgeat0x7fc3a6009a90>,
<matplotlib.patches.Wedgeat0x7fc3a6014110>,
<matplotlib.patches.Wedgeat0x7fc3a6014750>,
<matplotlib.patches.Wedgeat0x7fc3a6014d90>,
<matplotlib.patches.Wedgeat0x7fc3a5fa0410>,
<matplotlib.patches.Wedgeat0x7fc3a5fa0a50>,
<matplotlib.patches.Wedgeat0x7fc3a5fa90d0>,
<matplotlib.patches.Wedgeat0x7fc3a5fa9710>,
<matplotlib.patches.Wedgeat0x7fc3a5fa9d50>,
<matplotlib.patches.Wedgeat0x7fc3a5fb53d0>,
<matplotlib.patches.Wedgeat0x7fc3a5fb5a10>,
<matplotlib.patches.Wedgeat0x7fc3a5fc0090>,
<matplotlib.patches.Wedgeat0x7fc3a5fc06d0>,
<matplotlib.patches.Wedgeat0x7fc3a5fc0d10>,
<matplotlib.patches.Wedgeat0x7fc3a5fca390>,
<matplotlib.patches.Wedgeat0x7fc3a5fca9d0>,
<matplotlib.patches.Wedgeat0x7fc3a5fd5050>,
<matplotlib.patches.Wedgeat0x7fc3a5fd5690>,
<matplotlib.patches.Wedgeat0x7fc3a5fd5cd0>,
<matplotlib.patches.Wedgeat0x7fc3a5f61350>,
<matplotlib.patches.Wedgeat0x7fc3a5f61990>,
<matplotlib.patches.Wedgeat0x7fc3a5f61fd0>,
<matplotlib.patches.Wedgeat0x7fc3a5f6c650>,
<matplotlib.patches.Wedgeat0x7fc3a5f6cc90>,
<matplotlib.patches.Wedgeat0x7fc3a5f75310>,
<matplotlib.patches.Wedgeat0x7fc3a5f75950>,
<matplotlib.patches.Wedgeat0x7fc3a5f75f90>,
<matplotlib.patches.Wedgeat0x7fc3a5f80610>,
<matplotlib.patches.Wedgeat0x7fc3a5f80c50>,
<matplotlib.patches.Wedgeat0x7fc3a5f8a2d0>,
<matplotlib.patches.Wedgeat0x7fc3a5f8a910>,
<matplotlib.patches.Wedgeat0x7fc3a5f8af50>,
<matplotlib.patches.Wedgeat0x7fc3a5f945d0>,
<matplotlib.patches.Wedgeat0x7fc3a5f94c10>,
<matplotlib.patches.Wedgeat0x7fc3a5f1e290>,
<matplotlib.patches.Wedgeat0x7fc3a5f1e8d0>,
<matplotlib.patches.Wedgeat0x7fc3a5f1ef10>,
<matplotlib.patches.Wedgeat0x7fc3a5f2a590>,
<matplotlib.patches.Wedgeat0x7fc3a5f2abd0>,
<matplotlib.patches.Wedgeat0x7fc3a5f35250>,
<matplotlib.patches.Wedgeat0x7fc3a5f35890>,
<matplotlib.patches.Wedgeat0x7fc3a5f35ed0>,
<matplotlib.patches.Wedgeat0x7fc3a5f3f550>,
<matplotlib.patches.Wedgeat0x7fc3a5f3fb90>,
<matplotlib.patches.Wedgeat0x7fc3a5f4a210>,
<matplotlib.patches.Wedgeat0x7fc3a5f4a850>,
<matplotlib.patches.Wedgeat0x7fc3a5f4ae90>,
<matplotlib.patches.Wedgeat0x7fc3a5f55510>,
<matplotlib.patches.Wedgeat0x7fc3a5f55b50>,

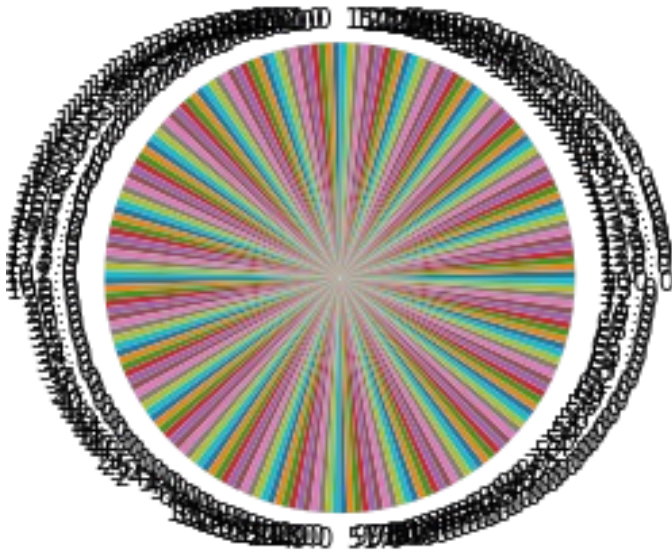
<matplotlib.patches.Wedgeat0x7fc3a5ee11d0>,
<matplotlib.patches.Wedgeat0x7fc3a5ee1810>,
<matplotlib.patches.Wedgeat0x7fc3a5ee1e50>,
<matplotlib.patches.Wedgeat0x7fc3a5eeb4d0>,
<matplotlib.patches.Wedgeat0x7fc3a5eebb10>,
<matplotlib.patches.Wedgeat0x7fc3a5ef6190>,
<matplotlib.patches.Wedgeat0x7fc3a5ef67d0>,
<matplotlib.patches.Wedgeat0x7fc3a5ef6e10>,
<matplotlib.patches.Wedgeat0x7fc3a5eff490>,
<matplotlib.patches.Wedgeat0x7fc3a5effad0>,
<matplotlib.patches.Wedgeat0x7fc3a5f0a150>,
<matplotlib.patches.Wedgeat0x7fc3a5f0a790>,
<matplotlib.patches.Wedgeat0x7fc3a5f0add0>,
<matplotlib.patches.Wedgeat0x7fc3a5f15450>,
<matplotlib.patches.Wedgeat0x7fc3a5f15a90>,
<matplotlib.patches.Wedgeat0x7fc3a5ea0110>,
<matplotlib.patches.Wedgeat0x7fc3a5ea0750>,
<matplotlib.patches.Wedgeat0x7fc3a5ea0d90>,
<matplotlib.patches.Wedgeat0x7fc3a5eaa410>,
<matplotlib.patches.Wedgeat0x7fc3a5eaaa50>,
<matplotlib.patches.Wedgeat0x7fc3a5eb50d0>,
<matplotlib.patches.Wedgeat0x7fc3a5eb5710>,
<matplotlib.patches.Wedgeat0x7fc3a5eb5d50>,
<matplotlib.patches.Wedgeat0x7fc3a5ec13d0>,
<matplotlib.patches.Wedgeat0x7fc3a5ec1a10>,
<matplotlib.patches.Wedgeat0x7fc3a5ecc090>,
<matplotlib.patches.Wedgeat0x7fc3a5ecc6d0>,
<matplotlib.patches.Wedgeat0x7fc3a5eccd10>,
<matplotlib.patches.Wedgeat0x7fc3a5ed6390>,
<matplotlib.patches.Wedgeat0x7fc3a5ed69d0>,
<matplotlib.patches.Wedgeat0x7fc3a5e61050>,
<matplotlib.patches.Wedgeat0x7fc3a5e61690>,
<matplotlib.patches.Wedgeat0x7fc3a5e61cd0>,
<matplotlib.patches.Wedgeat0x7fc3a5e6b350>,
<matplotlib.patches.Wedgeat0x7fc3a5e6b990>,
<matplotlib.patches.Wedgeat0x7fc3a5e6bfd0>,
<matplotlib.patches.Wedgeat0x7fc3a5e75650>,
<matplotlib.patches.Wedgeat0x7fc3a5e75c90>,
<matplotlib.patches.Wedgeat0x7fc3a5e82310>,
<matplotlib.patches.Wedgeat0x7fc3a5e82950>,
<matplotlib.patches.Wedge at
0x7fc3a5e82f90>],[Text(1.099864295735893,0.017278048656840017,'1.0'),
Text(1.098778862512746,0.05181709462326203,'138.0'),
Text(1.0966090672579494,0.08630500337553038,'128.0'),
Text(1.0933570512973614,0.12070773951300358,'129.0'),
Text(1.089026023977971,0.15499135169012426,'130.0'),
Text(1.0836202595006603,0.18912200612229665,'131.0'),
Text(1.0771450927020836,0.22306601997574568,'132.0'),
Text(1.0696069137898296,0.25678989460840573,'133.0'),
Text(1.0610131620360566,0.2902603486290348,'134.0'),

Text(1.0513723184358303,0.323444350741928,'135.0'),
Text(1.0406938973374056,0.35630915234481686,'136.0'),
Text(1.028988437052714,0.3888223198477845,'137.0'),
Text(1.0162674894573234,0.4209517666813023,'139.0'),
Text(1.00254360859013,0.4526657849617979,'2.0'),
Text(0.9878303382640403,0.4839330767835073,'140.0'),
Text(0.9721421986998625,0.5147227851057277,'141.0'),
Text(0.9554946721966024,0.5450045242049898,'142.0'),
Text(0.9379041878523043,0.5747484096620972,'143.0'),
Text(0.9193881053505146,0.6039250878544384,'144.0'),
Text(0.8999646978283725,0.632505764924468,'145.0'),
Text(0.8796531338432289,0.6604622351957656,'146.0'),
Text(0.8584734584555963,0.6877669090086319,'147.0'),
Text(0.8364465734470948,0.7143928399477517,'148.0'),
Text(0.8135942166929181,0.7403137514350502,'149.0'),
Text(0.789938940709176,0.7655040626615023,'127.0'),
Text(0.7655040903962849,0.7899389138323019,'126.0'),
Text(0.7403137800003703,0.8135941907004767,'125.0'),
Text(0.7143928693154189,0.8364465483647374,'124.0'),
Text(0.6877669391496638,0.8584734343080762,'103.0'),
Text(0.6604622660804166,0.8796531106543768,'104.0'),
Text(0.6325057965222588,0.8999646756210731,'105.0'),
Text(0.6039251201341856,0.9193880841466838,'106.0'),
Text(0.5747484425919446,0.9379041676728674,'107.0'),
Text(0.5450045577524398,0.9554946530614746,'108.0'),
Text(0.5147228192376729,0.9721421806279276,'109.0'),
Text(0.4839331114662637,0.9878303212731329,'110.0'),
Text(0.45266582016113777,1.002543592697018,'111.0'),
Text(0.42095180236248814,1.0162674746776919,'112.0'),
Text(0.38882235597560333,1.0289884234011486,'113.0'),
Text(0.3563091888836148,1.0406938848273783,'114.0'),
Text(0.3234443876556455,1.0513723070796874,'115.0'),
Text(0.29026038588124276,1.0610131518450052,'116.0'),
Text(0.2567899321623406,1.0696069047739272,'117.0'),
Text(0.22306605779434627,1.0771450848702275,'118.0'),
Text(0.18912204416824066,1.0836202528605794,'119.0'),
Text(0.1549913899258649,1.0890260185362188,'120.0'),
Text(0.12070777790080672,1.093357047059308,'121.0'),
Text(0.08630504187751195,1.0966090642277773,'122.0'),
Text(0.051817133201425246,1.0987788606934454,'123.0'),
Text(0.017278087273112874,1.0998642951292597,'150.0'),
Text(-0.01727801004056712,1.0998642963425254,'151.0'),
Text(-0.05181705604509893,1.0987788643320453,'152.0'),
Text(-0.08630496487354881,1.0966090702881204,'177.0'),
Text(-0.12070770112520036,1.0933570555354135,'179.0'),
Text(-0.15499131345438355,1.089026029419722,'180.0'),
Text(-0.18912196807635245,1.0836202661407397,'181.0'),
Text(-0.22306598215714474,1.0771451005339385,'182.0'),
Text(-0.2567898570544705,1.069606922805731,'183.0'),
Text(-0.29026031137682645,1.0610131722271068,'184.0'),

Text(-0.32344431382821,1.0513723297919717,'185.0'),
Text(-0.35630911580601843,1.0406939098474313,'186.0'),
Text(-0.38882228371996524,1.0289884507042784,'187.0'),
Text(-0.42095173100011607,1.0162675042369533,'188.0'),
Text(-0.45266574976245755,1.0025436244832402,'189.0'),
Text(-0.4839330421007501,0.9878303552549466,'190.0'),
Text(-0.5147227509737815,0.9721422167717966,'191.0'),
Text(-0.5450044906575389,0.9554946913317294,'192.0'),
Text(-0.5747483767322485,0.93790420803174,'193.0'),
Text(-0.6039250555746907,0.9193881265543443,'194.0'),
Text(-0.6325057333266766,0.8999647200356706,'195.0'),
Text(-0.6604622043111139,0.8796531570320797,'196.0'),
Text(-0.6877668788675995,0.8584734826031152,'197.0'),
Text(-0.7143928105800839,0.8364465985294511,'198.0'),
Text(-0.7403137228697292,0.8135942426853584,'199.0'),
Text(-0.7655040349267188,0.789938967586049,'178.0'),
Text(-0.7899388869554268,0.7655041181310663,'176.0'),
Text(-0.8135941647080343,0.7403138085656895,'153.0'),
Text(-0.836446523282379,0.7143928986830851,'175.0'),
Text(-0.858473410160555,0.6877669692906948,'154.0'),
Text(-0.8796530874655237,0.6604622969650666,'155.0'),
Text(-0.8999646534137725,0.6325058281200486,'156.0'),
Text(-0.9193880629428517,0.6039251524139322,'157.0'),
Text(-0.9379041474934295,0.5747484755217915,'158.0'),
Text(-0.9554946339263455,0.5450045912998892,'159.0'),
Text(-0.9721421625559913,0.5147228533696177,'160.0'),
Text(-0.9878303042822243,0.48393314614901967,'161.0'),
Text(-1.0025435768039053,0.45266585536047715,'162.0'),
Text(-1.016267459898059,0.42095183804367353,'163.0'),
Text(-1.0289884097495818,0.3888223921034217,'164.0'),
Text(-1.04069387231735,0.3563092254224123,'165.0'),
Text(-1.0513722957235434,0.32344442456936306,'166.0'),
Text(-1.0610131416539528,0.2902604231334502,'167.0'),
Text(-1.0696068957580231,0.25678996971627505,'168.0'),
Text(-1.07714507703837,0.2230660956129465,'169.0'),
Text(-1.0836202462204974,0.18912208221418428,'170.0'),
Text(-1.0890260130944651,0.15499142816160513,'171.0'),
Text(-1.0933570428212531,0.12070781628860958,'172.0'),
Text(-1.0966090611976036,0.08630508037949325,'173.0'),
Text(-1.0987788588741434,0.05181717177958819,'174.0'),
Text(-1.0998642945226247,0.01727812588938547,'102.0'),
Text(-1.099864296949156,-0.017277971424294453,'101.0'),
Text(-1.0987788661513433,-0.05181701746693556,'100.0'),
Text(-1.0966090733182898,-0.08630492637156696,'26.0'),
Text(-1.0933570597734645,-0.12070766273739685,'28.0'),
Text(-1.0890260348614718,-0.15499127521864245,'29.0'),
Text(-1.0836202727808175,-0.1891219300304079,'30.0'),
Text(-1.0771451083657921,-0.22306594433854343,'31.0'),
Text(-1.0696069318216308,-0.2567898195005348,'32.0'),
Text(-1.0610131824181555,-0.2902602741246177,'33.0'),

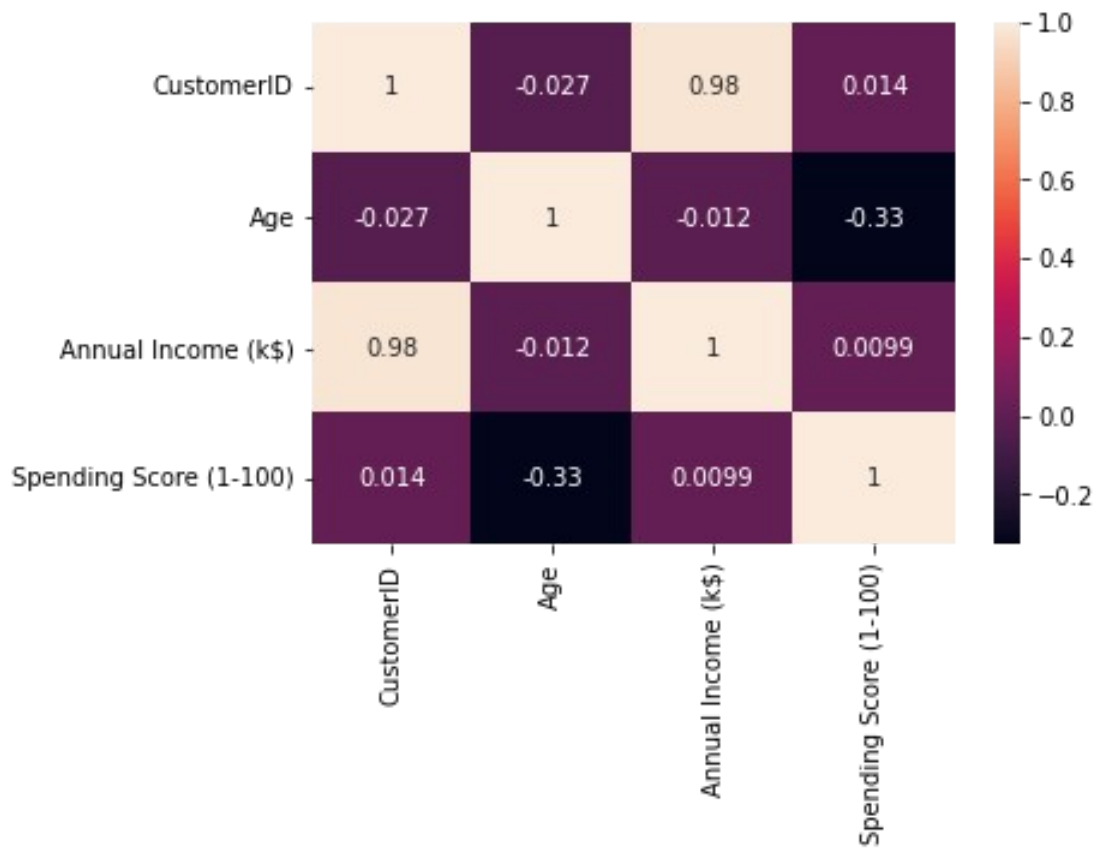
Text(-1.051372341148112,-0.32344427691449146,'34.0'),
Text(-1.0406939223574558,-0.3563090792672196,'35.0'),
Text(-1.0289884643558413,-0.38882224759214545,'36.0'),
Text(-1.0162675190165824,-0.4209516953189289,'37.0'),
Text(-1.0025436403763497,-0.4526657145631162,'38.0'),
Text(-0.9878303722458512,-0.48393300741799283,'39.0'),
Text(-0.972142234843729,-0.5147227168418353,'40.0'),
Text(-0.9554947104668547,-0.5450044571100877,'41.0'),
Text(-0.9379042282111743,-0.5747483438024,'42.0'),
Text(-0.919388147758173,-0.6039250232949418,'43.0'),
Text(-0.8999647422429681,-0.6325057017288843,'44.0'),
Text(-0.8796531802209296,-0.6604621734264611,'45.0'),
Text(-0.8584735067506332,-0.6877668487265657,'46.0'),
Text(-0.8364466236118066,-0.7143927812124149,'47.0'),
Text(-0.8135942686777979,-0.7403136943044072,'48.0'),
Text(-0.7899389944629214,-0.7655040071919342,'27.0'),
Text(-0.765504145865847,-0.7899388600785507,'25.0'),
Text(-0.7403138371310077,-0.8135941387155907,'50.0'),
Text(-0.7143929280507503,-0.8364464982000198,'24.0'),
Text(-0.687766999431725,-0.8584733860130329,'3.0'),
Text(-0.6604623278497157,-0.8796530642766697,'4.0'),
Text(-0.6325058597178379,-0.8999646312064707,'5.0'),
Text(-0.6039251846936776,-0.9193880417390187,'6.0'),
Text(-0.5747485084516377,-0.9379041273139904,'7.0'),
Text(-0.5450046248473376,-0.9554946147912153,'8.0'),
Text(-0.5147228875015618,-0.9721421444840537,'9.0'),
Text(-0.4839331808317745,-0.9878302872913146,'10.0'),
Text(-0.45266589055981604,-1.0025435609107909,'11.0'),
Text(-0.42095187372485793,-1.0162674451184253,'12.0'),
Text(-0.3888224282312395,-1.0289883960980137,'13.0'),
Text(-0.35630926196120893,-1.0406938598073205,'14.0'),
Text(-0.3234444614830799,-1.0513722843673978,'15.0'),
Text(-0.29026046038565756,-1.0610131314628988,'16.0'),
Text(-0.25679000727020984,-1.0696068867421178,'17.0'),
Text(-0.2230661334315467,-1.077145069206511,'18.0'),
Text(-0.18912212026012837,-1.083620239580414,'19.0'),
Text(-0.15499146639734543,-1.08902600765271,'20.0'),
Text(-0.12070785467641298,-1.0933570385831968,'21.0'),
Text(-0.08630511888147467,-1.0966090581674286,'22.0'),
Text(-0.051817210357750824,-1.0987788570548402,'23.0'),
Text(-0.017278164505658295,-1.0998642939159884,'49.0'),
Text(0.017277932808022013,-1.0998642975557855,'51.0'),
Text(0.051816978888772125,-1.0987788679706398,'99.0'),
Text(0.08630488786958548,-1.0966090763484582,'76.0'),
Text(0.12070762434959317,-1.093357064011514,'78.0'),
Text(0.15499123698290168,-1.08902604030322,'79.0'),
Text(0.1891218919844631,-1.0836202794208944,'80.0'),
Text(0.22306590651994235,-1.0771451161976442,'81.0'),
Text(0.25678978194659885,-1.0696069408375295,'82.0'),
Text(0.29026023687240904,-1.0610131926092028,'83.0'),

Text(0.32344424000077254,-1.0513723525042509,'84.0'),
Text(0.3563090427284208,-1.040693934867479,'85.0'),
Text(0.3888222114643252,-1.028988478007403,'86.0'),
Text(0.4209516596377419,-1.01626753379621,'87.0'),
Text(0.4526656793637746,-1.0025436562694579,'88.0'),
Text(0.4839329727352345,-0.987830389236755,'89.0'),
Text(0.5147226827098876,-0.9721422529156608,'90.0'),
Text(0.5450044235626356,-0.9554947296019792,'91.0'),
Text(0.5747483108725499,-0.9379042483906078,'92.0'),
Text(0.6039249910151923,-0.9193881689620005,'93.0'),
Text(0.6325056701310908,-0.8999647644502644,'94.0'),
Text(0.6604621425418077,-0.8796532034097785,'95.0'),
Text(0.6877668185855317,-0.8584735308981498,'96.0'),
Text(0.714392751844745,-0.8364466486941609,'97.0'),
Text(0.7403136657390846,-0.8135942946702361,'98.0'),
Text(0.7655039794571488,-0.7899390213397927,'77.0'),
Text(0.7899388332016739,-0.7655041736006266,'75.0'),
Text(0.8135941127231462,-0.7403138656963253,'52.0'),
Text(0.8364464731176593,-0.7143929574184149,'74.0'),
Text(0.8584733618655095,-0.6877670295727545,'53.0'),
Text(0.8796530410878143,-0.6604623587343643,'54.0'),
Text(0.899964608999168,-0.6325058913156264,'55.0'),
Text(0.9193880205351843,-0.6039252169734227,'56.0'),
Text(0.9379041071345502,-0.5747485413814832,'57.0'),
Text(0.9554945956560839,-0.5450046583947857,'58.0'),
Text(0.9721421264121151,-0.5147229216335053,'59.0'),
Text(0.9878302703004036,-0.4839332155145293,'60.0'),
Text(1.0025435450176754,-0.45266592575915426,'61.0'),
Text(1.01626743033879,-0.4209519094060423,'62.0'),
Text(1.0289883824464443,-0.38882246435905693,'63.0'),
Text(1.0406938472972895,-0.35630929850000553,'64.0'),
Text(1.0513722730112511,-0.32344449839679645,'65.0'),
Text(1.0610131212718434,-0.29026049763786455,'66.0'),
Text(1.0696068777262113,-0.2567900448241439,'67.0'),
Text(1.077145061374651,-0.2230661712501466,'68.0'),
Text(1.0836202329403295,-0.1891221583060708,'69.0'),
Text(1.089026002210954,-0.15499150463308553,'70.0'),
Text(1.0933570343451393,-0.1207078930642148,'71.0'),
Text(1.0966090551372523,-0.086305157383456,'72.0'),
Text(1.0987788552355353,-0.05181724893591388,'73.0'),
Text(1.0998642933093508,-0.017278203121931096,'200.0'))



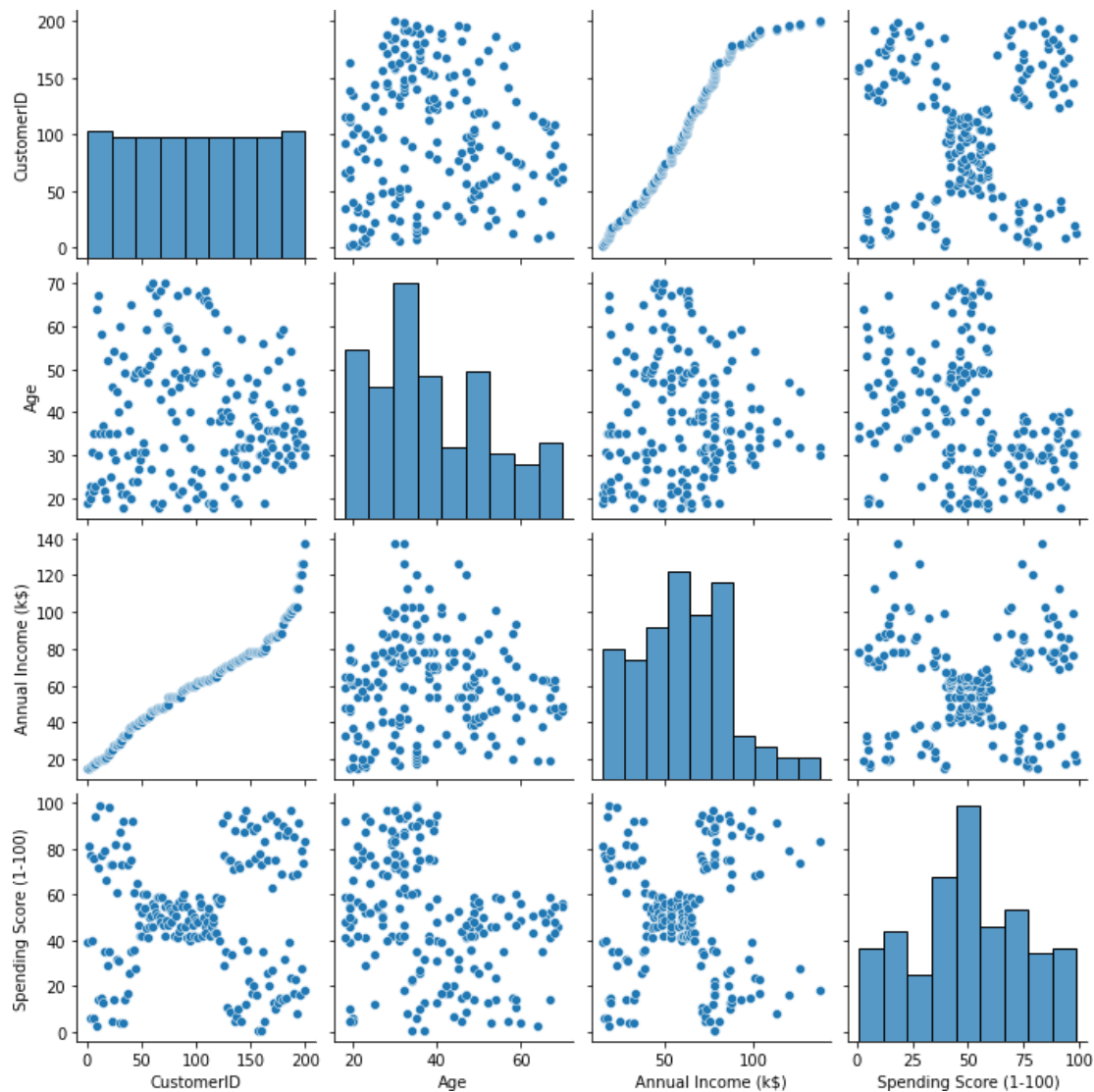
```
sns.heatmap(customer.corr(),annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc3aba09490>
```



```
sns.pairplot(customer)
```

<seaborn.axisgrid.PairGrid at0x7fc3ab9097d0>



6.Performdescriptivestatisticsonthedataset

```
#Createa DataFrame
df = pd.DataFrame(customer)df
```

	CustomerID	Gender	Age	Annual	Income	(k\$)	SpendingScore(1-
100)							
0	1.0	Male	19.0			15.0	
39.0							
1	2.0	Male	21.0			15.0	
81.0							
2	3.0	Female	20.0			16.0	
6.0							

3	4.0	Female	23.0	16.0
77.0				
4	5.0	Female	31.0	17.0
40.0				
..
...				
195	196.0	Female	35.0	120.0
79.0				
196	197.0	Female	45.0	126.0
28.0				
197	198.0	Male	32.0	126.0
74.0				
198	199.0	Male	32.0	137.0
18.0				
199	200.0	Male	30.0	137.0
83.0				

[200rows x5 columns]

#Createa DataFrame

df =

pd.DataFrame(customer)df.sum()

CustomerID2

0100.0

GenderMaleMaleFemaleFemaleFemaleFemaleFemaleFemaleMa...A

ge

7770.0

Annual Income

(k\$)12112.0

SpendingScore(1-100)

10040.0

dtype:object

#axis=1

df.sum(1)

0 74.0

1 119.0

2 45.0

3 120.0

4 93.0

...

195 430.0

196 396.0

197 430.0

198 386.0

199 450.0

Length:200,dtype: float64


```
df.mean()
```

```
CustomerID      100.50
Age              38.85
Annual Income(k$) 60.56
Spending Score (1-100)dtype:float64 50.20
```

```
df.std()
```

```
CustomerID      57.879185
Age              13.969007
Annual Income(k$) 26.264721
Spending Score (1-100)dtype:float64 25.823522
```

```
df.describe()
```

```

      CustomerID      AgeAnnual      Income (k$)      Spending      Score (1-100)
count200.000000200.000000      200.000000
mean  100.500000      38.850000      60.560000
std    57.879185      13.969007      26.264721
min     1.000000      18.000000      15.000000
25%    50.750000      28.750000      41.500000
50%    100.500000      36.000000      61.500000
75%    150.250000      49.000000      78.000000
max    200.000000      70.000000     137.000000
```

```
df.describe(include=['object'])
```

```

      Gender
count      200
unique       2
top      Female
freq       112
```

```
df.describe(include='all')
```

```

      CustomerIDGender      AgeAnnualIncome(k$)\
count      200.000000      200      200.000000      200.000000
unique      NaN      2      NaN      NaN
top      NaN      Female      NaN      NaN
freq      NaN      112      NaN      NaN
```

mean	100.500000	NaN	38.850000	60.560000
std	57.879185	NaN	13.969007	26.264721
min	1.000000	NaN	18.000000	15.000000
25%	50.750000	NaN	28.750000	41.500000
50%	100.500000	NaN	36.000000	61.500000
75%	150.250000	NaN	49.000000	78.000000
max	200.000000	NaN	70.000000	137.000000

Spending Score(1-100)

count	200.000000
unique	NaN
top	NaN
freq	NaN
mean	50.200000
std	25.823522
min	1.000000
25%	34.750000
50%	50.000000
75%	73.000000
max	99.000000

customer["Age"].mean()38.85

customer["Annual Income (k\$)"].median()61.5

customer.max()

CustomerID	200.0
Gender	Male
Age	70.0
AnnualIncome(k\$)	137.0
SpendingScore (1-100)	99.0

dtype:

objectcustomer.mi

n()

CustomerID	1.0
Gender	Female
Age	18.0
AnnualIncome(k\$)	15.0
SpendingScore (1-100)	1.0

dtype:objectcustomer.kur

tosis()

CustomerID	-1.200000
Age	-0.671573
Annual Income(k\$)	-0.098487

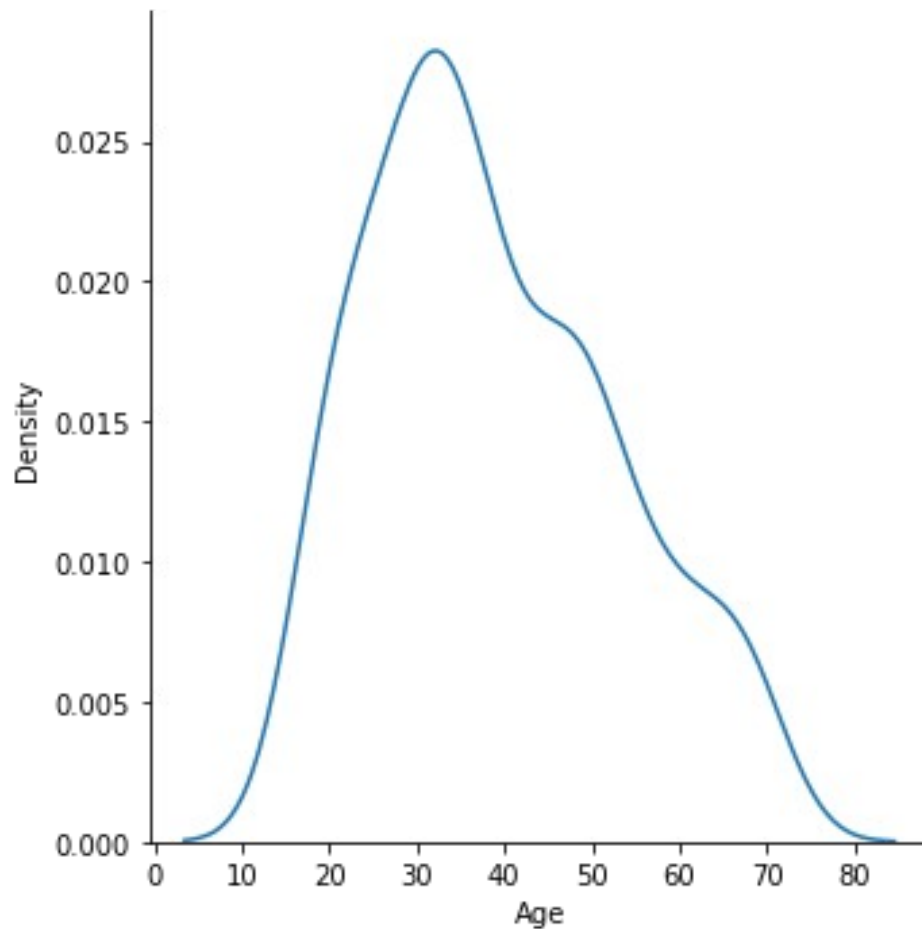
```
SpendingScore(1-100)      -  
0.826629dtype:float64
```

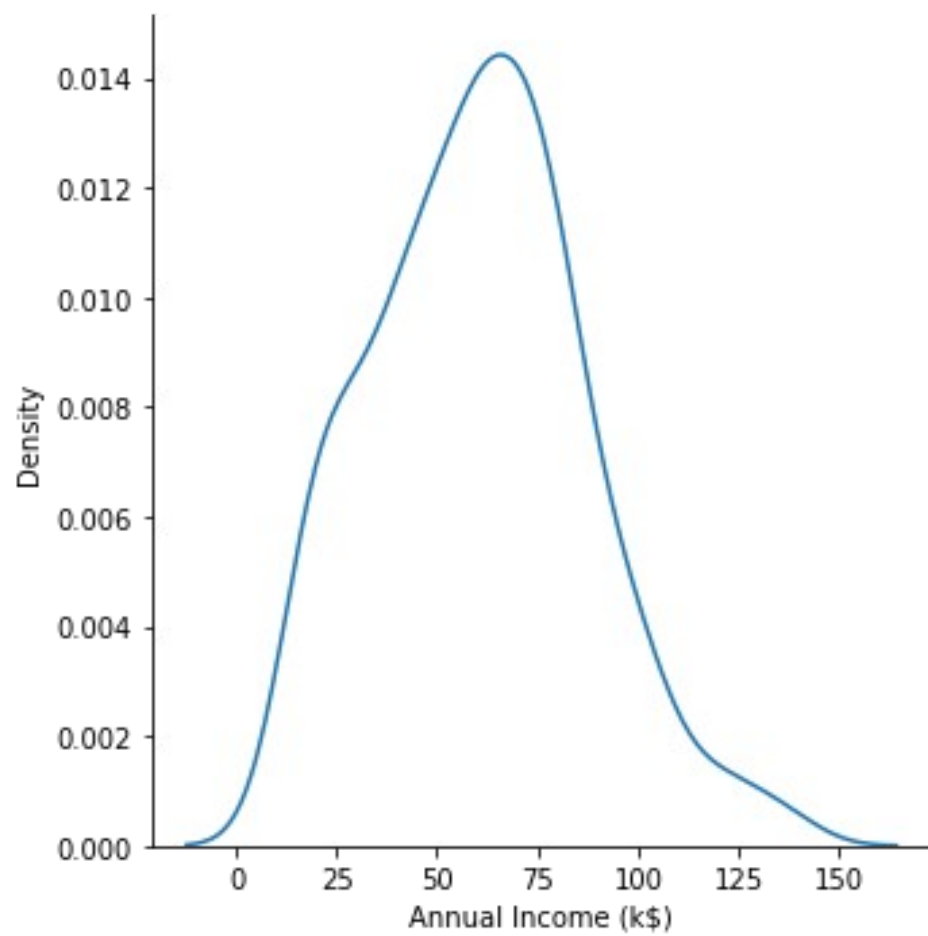
```
print(sns.displot(customer["Age"],kind="kde")),print(sns.displot(customer["Annual  
Income(k$)"],kind="kde")),print(sns.displot(customer["SpendingScore (1-100)"],kind= "kde"))
```

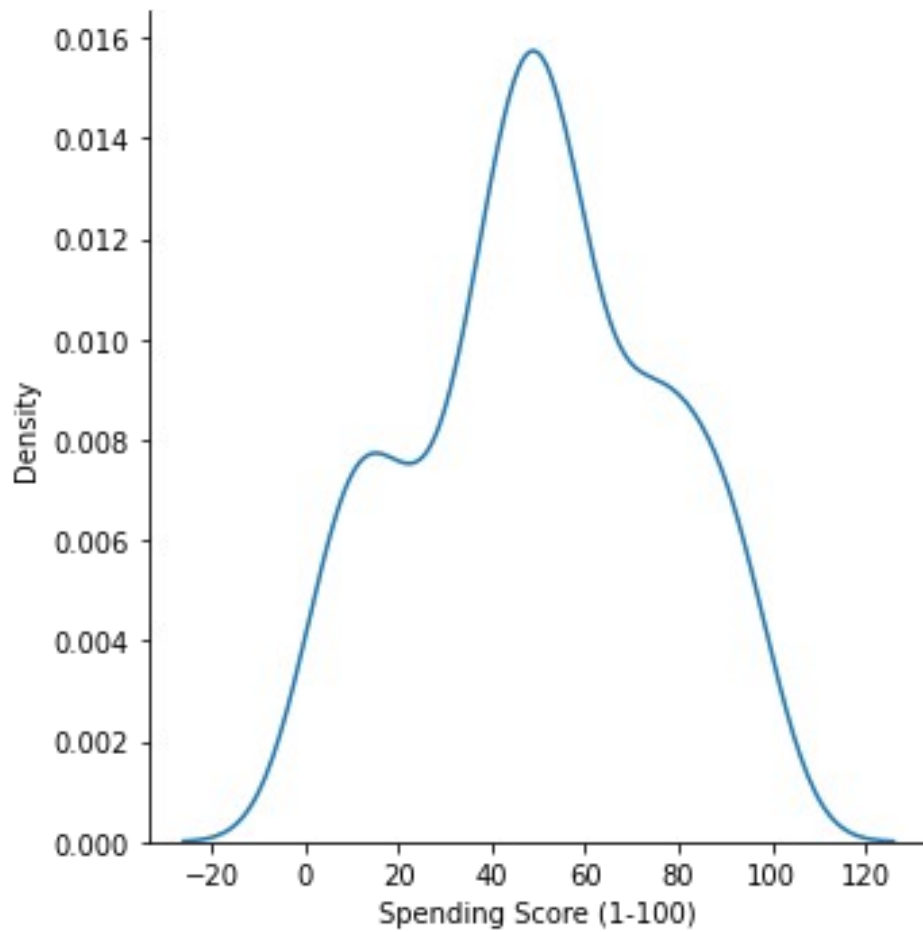
```
<seaborn.axisgrid.FacetGridobject at0x7f7e9c366c50>
```

```
<seaborn.axisgrid.FacetGridobject at0x7f7e9e0fc410>
```

```
<seaborn.axisgrid.FacetGridobject at0x7f7e9c30bf50>
```







7. Check with missing value and deal with them

`df.fillna(value=100)`

	CustomerID	Gender	Age	Annual	Income (k\$)	SpendingScore(1-100)
0	1.0	Male	19.0		15.0	
1	2.0	Male	21.0		15.0	
2	3.0	Female	20.0		16.0	
3	4.0	Female	23.0		16.0	
4	5.0	Female	31.0		17.0	
...	
195	196.0	Female	35.0		120.0	
196	197.0	Female	45.0		126.0	

28.0				
197	198.0	Male	32.0	126.0
74.0				
198	199.0	Male	32.0	137.0
18.0				
199	200.0	Male	30.0	137.0
83.0				

[200 rows x 5 columns]df

	CustomerID	Gender	Age	Annual	Income	(k\$)	Spending	Score	(1-
100)									
0	1.0	Male	19.0			15.0			
39.0									
1	2.0	Male	21.0			15.0			
81.0									
2	3.0	Female	20.0			16.0			
6.0									
3	4.0	Female	23.0			16.0			
77.0									
4	5.0	Female	31.0			17.0			
40.0									
..			
...									
195	196.0	Female	35.0			120.0			
79.0									
196	197.0	Female	45.0			126.0			
28.0									
197	198.0	Male	32.0			126.0			
74.0									
198	199.0	Male	32.0			137.0			
18.0									
199	200.0	Male	30.0			137.0			
83.0									

[200 rows x 5

columns]df["Age"].mean()

38.85

df["Age"].median()36.0

df["Age"].fillna(df["Age"].mean(),inplace= True)df

	CustomerID	Gender	Age	Annual	Income	(k\$)	Spending	Score	(1-
100)									
0	1.0	Male	19.0			15.0			
39.0									
1	2.0	Male	21.0			15.0			
81.0									
2	3.0	Female	20.0			16.0			
6.0									
3	4.0	Female	23.0			16.0			
77.0									
4	5.0	Female	31.0			17.0			
40.0									
..			
...									
195	196.0	Female	35.0			120.0			
79.0									
196	197.0	Female	45.0			126.0			
28.0									
197	198.0	Male	32.0			126.0			
74.0									
198	199.0	Male	32.0			137.0			
18.0									
199	200.0	Male	30.0			137.0			
83.0									

[200rows x5 columns]

df["Annual Income (k\$)"].fillna(df["Annual Income(k\$)"].median(),inplace=True)

df

	CustomerID	Gender	Age	Annual	Income	(k\$)	Spending	Score	(1-
100)									
0	1.0	Male	19.0			15.0			
39.0									
1	2.0	Male	21.0			15.0			
81.0									
2	3.0	Female	20.0			16.0			
6.0									
3	4.0	Female	23.0			16.0			
77.0									
4	5.0	Female	31.0			17.0			
40.0									
..			
...									
195	196.0	Female	35.0			120.0			
79.0									
196	197.0	Female	45.0			126.0			
28.0									
197	198.0	Male	32.0			126.0			

```

74.0
198      199.0      Male  32.0      137.0
18.0
199      200.0      Male  30.0      137.0
83.0

```

[200rows x5 columns]

```
df= df.replace("Male",np.nan)df
```

```

      CustomerID  Gender  Age  Annual  Income  (k$)  Spending  Score  (1-
100)
0           1.0     NaN  19.0      15.0
39.0
1           2.0     NaN  21.0      15.0
81.0
2           3.0  Female  20.0      16.0
6.0
3           4.0  Female  23.0      16.0
77.0
4           5.0  Female  31.0      17.0
40.0
..           ...     ...   ...     ...
...
195          196.0  Female  35.0      120.0
79.0
196          197.0  Female  45.0      126.0
28.0
197          198.0     NaN  32.0      126.0
74.0
198          199.0     NaN  32.0      137.0
18.0
199          200.0     NaN  30.0      137.0
83.0

```

[200rows x5 columns]

8. Find the outlier and replace them

Method to outlier detection

```
qnt = customer.quantile(q= (0.25,0.75))qnt
```

```

      CustomerID  Age  AnnualIncome(k$)  Spending  Score  (1-100)
0.25      50.75  28.75      41.5      34.75
0.75     150.25  49.00      78.0      73.00

```

```
iqr =qnt.loc[0.75]-qnt.loc[0.25]      #IQR=Q3-Q1
```

```
iqr
```


CustomerID	99.50
Age	20.25
AnnualIncome(k\$)	36.50
SpendingScore (1-100)	38.25

dtype:float64

lower=qnt.loc[0.25]-1.5* iqrlower

CustomerID	-98.500
Age	-1.625
Annual Income (k\$)-13.250	Spending Score
(1-100)-22.625	dtype:float64

upper=qnt.loc[0.75]+1.5* iqrupper

CustomerID	299.500
Age	79.375
AnnualIncome(k\$)	132.750
SpendingScore (1-100)	130.375

dtype:

float64customer.me

an()

CustomerID	100.50
Age	38.85
AnnualIncome(k\$)	60.56
SpendingScore (1-100)	50.20

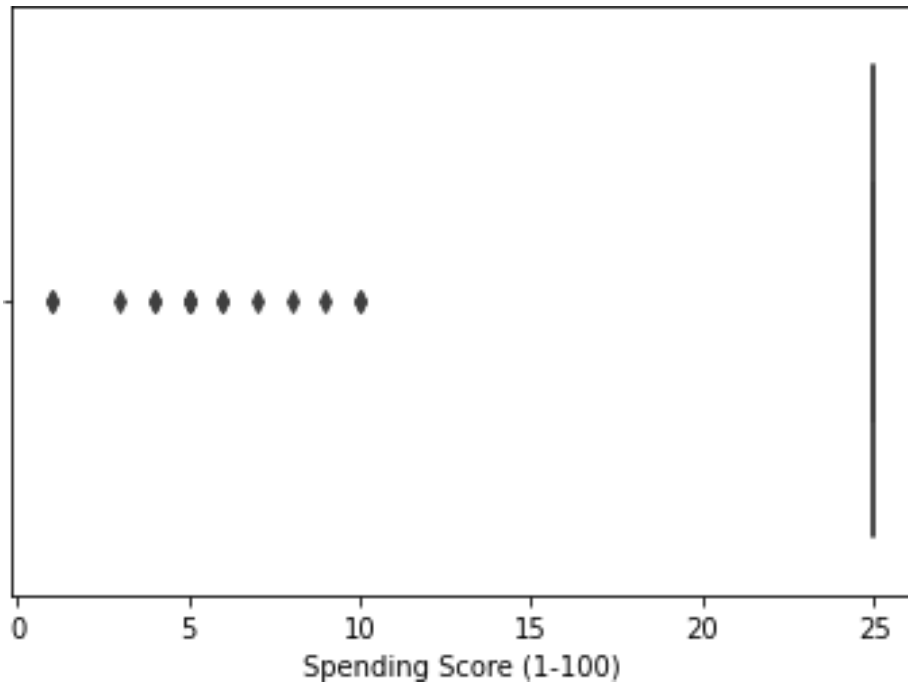
dtype:float64

###replacingoutlier

customer["Spending Score(1-100)"]= np.where(customer["Spending Score(1-100)"]>10,25,customer["SpendingScore(1-100)"])

sns.boxplot(customer["SpendingScore(1-100)"])

<matplotlib.axes._subplots.AxesSubplotat0x7f7ea0febbd0>



```
customer.isnull().sum()
```

```
CustomerID          0
Gender              0
Age                0
AnnualIncome(k$)    0
SpendingScore (1-100) 0
dtype: int64
```

```
customer= customer.dropna(axis=0)customer.isnull().sum()
```

```
CustomerID          0
Gender              0
Age                0
AnnualIncome (k$)    0
SpendingScore (1-100) 0
dtype: int64
```

9. Check for Categorical columns and perform encoding

```
customer['Gender'].unique()array(['Male','Female'],dtype
=object)
```

```
from sklearn.preprocessing import LabelEncoder
```

```
gender=
```

```
LabelEncoder()gender.fit(customer['Gen
der'])
```

```
LabelEncoder()
```

```
marry_values=gender.transform(customer['Gender'])print("Before
```

```
Encoding:",list(customer['Gender'][-10:]))
```

```
Before Encoding: ['Female', 'Female', 'Male', 'Female', 'Female', 'Female', 'Female', 'Male', 'Male', 'Male']
```

```
print("AfterEncoding:",customer[-10:])
```

After Encoding:	CustomerID	Gender	Age	AnnualIncome(k\$)	SpendingScore(1-100)
190	191.0	Female	34.0		103.0
23.0					
191	192.0	Female	32.0		103.0
69.0					
192	193.0	Male	33.0		113.0
8.0					
193	194.0	Female	38.0		113.0
91.0					
194	195.0	Female	47.0		120.0
16.0					
195	196.0	Female	35.0		120.0
79.0					
196	197.0	Female	45.0		126.0
28.0					
197	198.0	Male	32.0		126.0
74.0					
198	199.0	Male	32.0		137.0
18.0					
199	200.0	Male	30.0		137.0
83.0					

```
print("Theinversefromthe  
encodingresult:",gender.inverse_transform(marry_values[-10:]))
```

```
Theinversefromtheencodingresult:['Female','Female','Male','Female','Female','Female','Female',  
'Male',  
'Male','Male']
```

```
residence_encoder=LabelEncoder()residence_  
values=
```

```
residence_encoder.fit_transform(customer['CustomerID'])print("Before Encoding:",
```

```
list(customer['CustomerID'][:5]))BeforeEncoding:[1.0,2.0, 3.0,4.0, 5.0]
```

```
print("AfterEncoding:",residence_values[:5])AfterEncoding:[01234]
```

```
print("Theinversefrom  
theencodingresult:",residence_encoder.inverse_transform(residence_values[:5]))
```

Theinverse fromthe encodingresult:[1. 2. 3.4.

5.]fromsklearn.preprocessingimportOneHotEncoder

```
gender_encoder=OneHotEncoder()
```

```
from sklearn.preprocessing import  
OneHotEncoderimportnumpyasnp
```

```
gender_encoder=OneHotEncoder()  
gender_resaped=np.array(customer['Gender']).reshape(-1,1)gender_values  
=gender_encoder.fit_transform(gender_resaped)
```

```
print(customer['Gender'][:5])print()print(gender_  
values.toarray()[:5])print()  
print(gender_encoder.inverse_transform(gender_values)[:5])
```

```
0      Male  
1      Male  
2     Female  
3     Female  
4     Female  
Name: Gender, dtype: object
```

```
[[0.1.]  
 [0.1.]  
 [1.0.]  
 [1.0.]  
 [1.0.]]
```

```
['Male']  
['Male']  
['Female']  
['Female']  
['Female']
```

```
#Create the encoded dataframe#For  
'ever_married'column
```

```
Gender= pd.DataFrame(marry_values,columns=['Gender'])
```

```
#For'residence_type'column
```

```
Age = pd.DataFrame(residence_values,columns=['Age'])
```

```
#For'gender'column
```

```
gender=pd.DataFrame(gender_values.toarray(),columns=['Female','Male'])
```

```
#Combine all categorical columns as one data frame
df_categorical_encoded= pd.concat([Gender, Age], axis=1)
```

```
#
The preview print(df_categorical_encoded.shape)
df_categorical_encoded.head()
```

```
(200,2)
```

```
GenderAge0 1
0
1 1 1
2 0 2
3 0 3
4 0 4
```

```
df_new= pd.concat([customer, df_categorical_encoded], axis=1)
```

```
print(df_new.shape)df_new
w.head()
```

```
(200,7)
```

```
CustomerIDGender    Age AnnualIncome(k$)SpendingScore(1-100)\
0          1.0    Male   19.0                15.0
39.0
1          2.0    Male   21.0                15.0
81.0
2          3.0  Female   20.0                16.0
6.0
3          4.0  Female   23.0                16.0
77.0
4          5.0  Female   31.0                17.0
40.0
```

```
GenderAge0 1
0
1 1 1
2 0 2
3 0 3
4 0 4
```

```
df_categorical_encoded= pd.get_dummies(customer,
drop_first=True)df_categorical_encoded.head()
```

```
CustomerID    AgeAnnual Income(k$) SpendingScore (1-100)Gender_Male
```

0	1.0	19.0	15.0	39.0
1				
1	2.0	21.0	15.0	81.0
1				
2	3.0	20.0	16.0	6.0
0				
3	4.0	23.0	16.0	77.0
0				
4	5.0	31.0	17.0	40.0
0				

```
df_new= pd.concat([customer, df_categorical_encoded],axis=1)df_new.head()
```

	CustomerID	Gender	Age	Annual Income(k\$)	SpendingScore (1-100)\
0	1.0	Male	19.0		15.0
39.0					
1	2.0	Male	21.0		15.0
81.0					
2	3.0	Female	20.0		16.0
6.0					
3	4.0	Female	23.0		16.0
77.0					
4	5.0	Female	31.0		17.0
40.0					

	CustomerID	Age	Annual Income(k\$)	SpendingScore (1-100)	Gender_Male
0	1.0	19.0		15.0	39.0
1					
1	2.0	21.0		15.0	81.0
1					
2	3.0	20.0		16.0	6.0
0					
3	4.0	23.0		16.0	77.0
0					
4	5.0	31.0		17.0	40.0
0					

10. Scaling the data

```
customer.columns
```

```
Index(['CustomerID', 'Gender', 'Age', 'Annual Income (k$)', 'SpendingScore(1-100)'],
      dtype='object')x=customer[["Age", "C
```

```
ustomerID"]]x
```

	Age	CustomerID
0	19.0	1.0
1	21.0	2.0
2	20.0	3.0
3	23.0	4.0
4	31.0	5.0
..
195	35.0	196.0
196	45.0	197.0
197	32.0	198.0
198	32.0	199.0
199	30.0	200.0

[200 rows x 2

columns]x.head()

	Age	CustomerID
0	19.0	1.0
1	21.0	2.0
2	20.0	3.0
3	23.0	4.0
4	31.0	5.0

from sklearn.preprocessing import

StandardScaler
scale=StandardScaler()

st_scale = scale.fit_transform(x)
st_scale

```
array([[-1.42456879, -1.7234121],
       [-1.28103541, -1.70609137],
       [-1.3528021, -1.68877065],
       [-1.13750203, -1.67144992],
       [-0.56336851, -1.6541292],
       [-1.20926872, -1.63680847],
       [-0.27630176, -1.61948775],
       [-1.13750203, -1.60216702],
       [1.80493225, -1.5848463],
       [-0.6351352, -1.56752558],
       [2.02023231, -1.55020485],
       [-0.27630176, -1.53288413],
       [1.37433211, -1.5155634],
       [-1.06573534, -1.49824268],
       [-0.13276838, -1.48092195],
       [-1.20926872, -1.46360123],
       [-0.27630176, -1.4462805],
       [-1.3528021, -1.42895978],
       [0.94373197, -1.41163905],
       [-0.27630176, -1.39431833],
```

[-0.27630176,	-1.3769976],
[-0.99396865,	-1.35967688],
[0.51313183,	-1.34235616],
[-0.56336851,	-1.32503543],
[1.08726535,	-1.30771471],
[-0.70690189,	-1.29039398],
[0.44136514,	-1.27307326],
[-0.27630176,	-1.25575253],
[0.08253169,	-1.23843181],
[-1.13750203,	-1.22111108],
[1.51786549,	-1.20379036],
[-1.28103541,	-1.18646963],
[1.01549866,	-1.16914891],
[-1.49633548,	-1.15182818],
[0.7284319,	-1.13450746],
[-1.28103541,	-1.11718674],
[0.22606507,	-1.09986601],
[-0.6351352,	-1.08254529],
[-0.20453507,	-1.06522456],
[-1.3528021,	-1.04790384],
[1.87669894,	-1.03058311],
[-1.06573534,	-1.01326239],
[0.65666521,	-0.99594166],
[-0.56336851,	-0.97862094],
[0.7284319,	-0.96130021],
[-1.06573534,	-0.94397949],
[0.80019859,	-0.92665877],
[-0.85043527,	-0.90933804],
[-0.70690189,	-0.89201732],
[-0.56336851,	-0.87469659],
[0.7284319,	-0.85737587],
[-0.41983513,	-0.84005514],
[-0.56336851,	-0.82273442],
[1.4460988,	-0.80541369],
[0.80019859,	-0.78809297],
[0.58489852,	-0.77077224],
[0.87196528,	-0.75345152],
[2.16376569,	-0.73613079],
[-0.85043527,	-0.71881007],
[1.01549866,	-0.70148935],
[2.23553238,	-0.68416862],
[-1.42456879,	-0.6668479],
[2.02023231,	-0.64952717],
[1.08726535,	-0.63220645],
[1.73316556,	-0.61488572],
[-1.49633548,	-0.597565],
[0.29783176,	-0.58024427],
[2.091999,	-0.56292355],
[-1.42456879,	-0.54560282],
[-0.49160182,	-0.5282821],

[2.23553238,	-0.51096138],
[0.58489852,	-0.49364065],
[1.51786549,	-0.47631993],
[1.51786549,	-0.4589992],
[1.4460988,	-0.44167848],
[-0.92220196,	-0.42435775],
[0.44136514,	-0.40703703],
[0.08253169,	-0.3897163],
[-1.13750203,	-0.37239558],
[0.7284319,	-0.35507485],
[1.30256542,	-0.33775413],
[-0.06100169,	-0.3204334],
[2.02023231,	-0.30311268],
[0.51313183,	-0.28579196],
[-1.28103541,	-0.26847123],
[0.65666521,	-0.25115051],
[1.15903204,	-0.23382978],
[-1.20926872,	-0.21650906],
[-0.34806844,	-0.19918833],
[0.80019859,	-0.18186761],
[2.091999,	-0.16454688],
[-1.49633548,	-0.14722616],
[0.65666521,	-0.12990543],
[0.08253169,	-0.11258471],
[-0.49160182,	-0.09526399],
[-1.06573534,	-0.07794326],
[0.58489852,	-0.06062254],
[-0.85043527,	-0.04330181],
[0.65666521,	-0.02598109],
[-1.3528021,	-0.00866036],
[-1.13750203,	0.00866036],
[0.7284319,	0.02598109],
[2.02023231,	0.04330181],
[-0.92220196,	0.06062254],
[0.7284319,	0.07794326],
[-1.28103541,	0.09526399],
[1.94846562,	0.11258471],
[1.08726535,	0.12990543],
[2.091999,	0.14722616],
[1.94846562,	0.16454688],
[1.87669894,	0.18186761],
[-1.42456879,	0.19918833],
[-0.06100169,	0.21650906],
[-1.42456879,	0.23382978],
[-1.49633548,	0.25115051],
[-1.42456879,	0.26847123],
[1.73316556,	0.28579196],
[0.7284319,	0.30311268],
[0.87196528,	0.3204334],
[0.80019859,	0.33775413],

[-0.85043527,	0.35507485],
[-0.06100169,	0.37239558],
[0.08253169,	0.3897163],
[0.010765,	0.40703703],
[-1.13750203,	0.42435775],
[-0.56336851,	0.44167848],
[0.29783176,	0.4589992],
[0.08253169,	0.47631993],
[1.4460988,	0.49364065],
[-0.06100169,	0.51096138],
[0.58489852,	0.5282821],
[0.010765,	0.54560282],
[-0.99396865,	0.56292355],
[-0.56336851,	0.58024427],
[-1.3528021,	0.597565],
[-0.70690189,	0.61488572],
[0.36959845,	0.63220645],
[-0.49160182,	0.64952717],
[-1.42456879,	0.6668479],
[-0.27630176,	0.68416862],
[1.30256542,	0.70148935],
[-0.49160182,	0.71881007],
[-0.77866858,	0.73613079],
[-0.49160182,	0.75345152],
[-0.99396865,	0.77077224],
[-0.77866858,	0.78809297],
[0.65666521,	0.80541369],
[-0.49160182,	0.82273442],
[-0.34806844,	0.84005514],
[-0.34806844,	0.85737587],
[0.29783176,	0.87469659],
[0.010765,	0.89201732],
[0.36959845,	0.90933804],
[-0.06100169,	0.92665877],
[0.58489852,	0.94397949],
[-0.85043527,	0.96130021],
[-0.13276838,	0.97862094],
[-0.6351352,	0.99594166],
[-0.34806844,	1.01326239],
[-0.6351352,	1.03058311],
[1.23079873,	1.04790384],
[-0.70690189,	1.06522456],
[-1.42456879,	1.08254529],
[-0.56336851,	1.09986601],
[0.80019859,	1.11718674],
[-0.20453507,	1.13450746],
[0.22606507,	1.15182818],
[-0.41983513,	1.16914891],
[-0.20453507,	1.18646963],
[-0.49160182,	1.20379036],

```

[0.08253169, 1.22111108],
[-0.77866858, 1.23843181],
[-0.20453507, 1.25575253],
[-0.20453507, 1.27307326],
[0.94373197, 1.29039398],
[-0.6351352, 1.30771471],
[1.37433211, 1.32503543],
[-0.85043527, 1.34235616],
[1.4460988, 1.35967688],
[-0.27630176, 1.3769976],
[-0.13276838, 1.39431833],
[-0.49160182, 1.41163905],
[0.51313183, 1.42895978],
[-0.70690189, 1.4462805],
[0.15429838, 1.46360123],
[-0.6351352, 1.48092195],
[1.08726535, 1.49824268],
[-0.77866858, 1.5155634],
[0.15429838, 1.53288413],
[-0.20453507, 1.55020485],
[-0.34806844, 1.56752558],
[-0.49160182, 1.5848463],
[-0.41983513, 1.60216702],
[-0.06100169, 1.61948775],
[0.58489852, 1.63680847],
[-0.27630176, 1.6541292],
[0.44136514, 1.67144992],
[-0.49160182, 1.68877065],
[-0.49160182, 1.70609137],
[-0.6351352, 1.72341211]])

```

#normalisation

```
from sklearn.preprocessing import MinMaxScaler
```

```
min_max=MinMaxScaler(feature_range=(0,1))norm=min_max.
fit_transform(x)
```

norm

```

array([[0.01923077, 0.00502513],
       [0.05769231, 0.01005025],
       [0.03846154, 0.01507538],
       [0.09615385, 0.0201005],
       [0.25, 0.02512563],
       [0.07692308, 0.03015075],
       [0.32692308, 0.03517588],
       [0.09615385, 0.04020101],
       [0.88461538, 0.04522613],
       [0.23076923, 0.05025126],
       [0.94230769, 0.05527638],
       [0.32692308, 0.05527638],

```

[0.76923077,	0.06030151],
[0.11538462,	0.06532663],
[0.36538462,	0.07035176],
[0.07692308,	0.07537688],
[0.32692308,	0.08040201],
[0.03846154,	0.08542714],
[0.65384615,	0.09045226],
[0.32692308,	0.09547739],
[0.32692308,	0.10050251],
[0.13461538,	0.10552764],
[0.53846154,	0.11055276],
[0.25 ,	0.11557789],
[0.69230769,	0.12060302],
[0.21153846,	0.12562814],
[0.51923077,	0.13065327],
[0.32692308,	0.13567839],
[0.42307692,	0.14070352],
[0.09615385,	0.14572864],
[0.80769231,	0.15075377],
[0.05769231,	0.15577889],
[0.67307692,	0.16080402],
[0. ,	0.16582915],
[0.59615385,	0.17085427],
[0.05769231,	0.1758794],
[0.46153846,	0.18090452],
[0.23076923,	0.18592965],
[0.34615385,	0.19095477],
[0.03846154,	0.1959799],
[0.90384615,	0.20100503],
[0.11538462,	0.20603015],
[0.57692308,	0.21105528],
[0.25 ,	0.2160804],
[0.59615385,	0.22110553],
[0.11538462,	0.22613065],
[0.61538462,	0.23115578],
[0.17307692,	0.2361809],
[0.21153846,	0.24120603],
[0.25 ,	0.24623116],
[0.59615385,	0.25125628],
[0.28846154,	0.25628141],
[0.25 ,	0.26130653],
[0.78846154,	0.26633166],
[0.61538462,	0.27135678],
[0.55769231,	0.27638191],
[0.63461538,	0.28140704],
[0.98076923,	0.28643216],
[0.17307692,	0.29145729],
[0.67307692,	0.29648241],
[1. ,	0.30150754],
[0.01923077,	0.30653266],

[0.94230769,	0.31155779],
[0.69230769,	0.31658291],
[0.86538462,	0.32160804],
[0.	0.32663317],
[0.48076923,	0.33165829],
[0.96153846,	0.33668342],
[0.01923077,	0.34170854],
[0.26923077,	0.34673367],
[1.	0.35175879],
[0.55769231,	0.35678392],
[0.80769231,	0.36180905],
[0.80769231,	0.36683417],
[0.78846154,	0.3718593],
[0.15384615,	0.37688442],
[0.51923077,	0.38190955],
[0.42307692,	0.38693467],
[0.09615385,	0.3919598],
[0.59615385,	0.39698492],
[0.75	0.40201005],
[0.38461538,	0.40703518],
[0.94230769,	0.4120603],
[0.53846154,	0.41708543],
[0.05769231,	0.42211055],
[0.57692308,	0.42713568],
[0.71153846,	0.4321608],
[0.07692308,	0.43718593],
[0.30769231,	0.44221106],
[0.61538462,	0.44723618],
[0.96153846,	0.45226131],
[0.	0.45728643],
[0.57692308,	0.46231156],
[0.42307692,	0.46733668],
[0.26923077,	0.47236181],
[0.11538462,	0.47738693],
[0.55769231,	0.48241206],
[0.17307692,	0.48743719],
[0.57692308,	0.49246231],
[0.03846154,	0.49748744],
[0.09615385,	0.50251256],
[0.59615385,	0.50753769],
[0.94230769,	0.51256281],
[0.15384615,	0.51758794],
[0.59615385,	0.52261307],
[0.05769231,	0.52763819],
[0.92307692,	0.53266332],
[0.69230769,	0.53768844],
[0.96153846,	0.54271357],
[0.92307692,	0.54773869],
[0.90384615,	0.55276382],
[0.01923077,	0.55778894],

[0.38461538,	0.56281407],
[0.01923077,	0.5678392],
[0.	0.57286432],
[0.01923077,	0.57788945],
[0.86538462,	0.58291457],
[0.59615385,	0.5879397],
[0.63461538,	0.59296482],
[0.61538462,	0.59798995],
[0.17307692,	0.60301508],
[0.38461538,	0.6080402],
[0.42307692,	0.61306533],
[0.40384615,	0.61809045],
[0.09615385,	0.62311558],
[0.25	0.6281407],
[0.48076923,	0.63316583],
[0.42307692,	0.63819095],
[0.78846154,	0.64321608],
[0.38461538,	0.64824121],
[0.55769231,	0.65326633],
[0.40384615,	0.65829146],
[0.13461538,	0.66331658],
[0.25	0.66834171],
[0.03846154,	0.67336683],
[0.21153846,	0.67839196],
[0.5	0.68341709],
[0.26923077,	0.68844221],
[0.01923077,	0.69346734],
[0.32692308,	0.69849246],
[0.75	0.70351759],
[0.26923077,	0.70854271],
[0.19230769,	0.71356784],
[0.26923077,	0.71859296],
[0.13461538,	0.72361809],
[0.19230769,	0.72864322],
[0.57692308,	0.73366834],
[0.26923077,	0.73869347],
[0.30769231,	0.74371859],
[0.30769231,	0.74874372],
[0.48076923,	0.75376884],
[0.40384615,	0.75879397],
[0.5	0.7638191],
[0.38461538,	0.76884422],
[0.55769231,	0.77386935],
[0.17307692,	0.77889447],
[0.36538462,	0.7839196],
[0.23076923,	0.78894472],
[0.30769231,	0.79396985],
[0.23076923,	0.79899497],
[0.73076923,	0.8040201],
[0.21153846,	0.80904523],

```

[0.01923077, 0.81407035],
[0.25, 0.81909548],
[0.61538462, 0.8241206],
[0.34615385, 0.82914573],
[0.46153846, 0.83417085],
[0.28846154, 0.83919598],
[0.34615385, 0.84422111],
[0.26923077, 0.84924623],
[0.42307692, 0.85427136],
[0.19230769, 0.85929648],
[0.34615385, 0.86432161],
[0.34615385, 0.86934673],
[0.65384615, 0.87437186],
[0.23076923, 0.87939698],
[0.76923077, 0.88442211],
[0.17307692, 0.88944724],
[0.78846154, 0.89447236],
[0.32692308, 0.89949749],
[0.36538462, 0.90452261],
[0.26923077, 0.90954774],
[0.53846154, 0.91457286],
[0.21153846, 0.91959799],
[0.44230769, 0.92462312],
[0.23076923, 0.92964824],
[0.69230769, 0.93467337],
[0.19230769, 0.93969849],
[0.44230769, 0.94472362],
[0.34615385, 0.94974874],
[0.30769231, 0.95477387],
[0.26923077, 0.95979899],
[0.28846154, 0.96482412],
[0.38461538, 0.96984925],
[0.55769231, 0.97487437],
[0.32692308, 0.9798995],
[0.51923077, 0.98492462],
[0.26923077, 0.98994975],
[0.26923077, 0.99497487],
[0.23076923, 1.]]))

```

#robustscaler

fromsklearn.preprocessingimport

RobustScalerRscale=RobustScaler()

RS = Rscale.fit_transform(x)RS

```

array([[ -0.83950617, -1.          ],
       [-0.74074074, -0.98994975],
       [-0.79012346, -0.9798995],

```

[-0.64197531,	-0.96984925],
[-0.24691358,	-0.95979899],
[-0.69135802,	-0.94974874],
[-0.04938272,	-0.93969849],
[-0.64197531,	-0.92964824],
[1.38271605,	-0.91959799],
[-0.2962963,	-0.90954774],
[1.5308642,	-0.89949749],
[-0.04938272,	-0.88944724],
[1.08641975,	-0.87939698],
[-0.59259259,	-0.86934673],
[0.04938272,	-0.85929648],
[-0.69135802,	-0.84924623],
[-0.04938272,	-0.83919598],
[-0.79012346,	-0.82914573],
[0.79012346,	-0.81909548],
[-0.04938272,	-0.80904523],
[-0.04938272,	-0.79899497],
[-0.54320988,	-0.78894472],
[0.49382716,	-0.77889447],
[-0.24691358,	-0.76884422],
[0.88888889,	-0.75879397],
[-0.34567901,	-0.74874372],
[0.44444444,	-0.73869347],
[-0.04938272,	-0.72864322],
[0.19753086,	-0.71859296],
[-0.64197531,	-0.70854271],
[1.18518519,	-0.69849246],
[-0.74074074,	-0.68844221],
[0.83950617,	-0.67839196],
[-0.88888889,	-0.66834171],
[0.64197531,	-0.65829146],
[-0.74074074,	-0.64824121],
[0.2962963,	-0.63819095],
[-0.2962963,	-0.6281407],
[0. ,	-0.61809045],
[-0.79012346,	-0.6080402],
[1.43209877,	-0.59798995],
[-0.59259259,	-0.5879397],
[0.59259259,	-0.57788945],
[-0.24691358,	-0.5678392],
[0.64197531,	-0.55778894],
[-0.59259259,	-0.54773869],
[0.69135802,	-0.53768844],
[-0.44444444,	-0.52763819],
[-0.34567901,	-0.51758794],
[-0.24691358,	-0.50753769],
[0.64197531,	-0.49748744],
[-0.14814815,	-0.48743719],
[-0.24691358,	-0.47738693],

[1.13580247,	-0.46733668],
[0.69135802,	-0.45728643],
[0.54320988,	-0.44723618],
[0.74074074,	-0.43718593],
[1.62962963,	-0.42713568],
[-0.44444444,	-0.41708543],
[0.83950617,	-0.40703518],
[1.67901235,	-0.39698492],
[-0.83950617,	-0.38693467],
[1.5308642,	-0.37688442],
[0.88888889,	-0.36683417],
[1.33333333,	-0.35678392],
[-0.88888889,	-0.34673367],
[0.34567901,	-0.33668342],
[1.58024691,	-0.32663317],
[-0.83950617,	-0.31658291],
[-0.19753086,	-0.30653266],
[1.67901235,	-0.29648241],
[0.54320988,	-0.28643216],
[1.18518519,	-0.27638191],
[1.18518519,	-0.26633166],
[1.13580247,	-0.25628141],
[-0.49382716,	-0.24623116],
[0.44444444,	-0.2361809],
[0.19753086,	-0.22613065],
[-0.64197531,	-0.2160804],
[0.64197531,	-0.20603015],
[1.03703704,	-0.1959799],
[0.09876543,	-0.18592965],
[1.5308642,	-0.1758794],
[0.49382716,	-0.16582915],
[-0.74074074,	-0.15577889],
[0.59259259,	-0.14572864],
[0.9382716,	-0.13567839],
[-0.69135802,	-0.12562814],
[-0.09876543,	-0.11557789],
[0.69135802,	-0.10552764],
[1.58024691,	-0.09547739],
[-0.88888889,	-0.08542714],
[0.59259259,	-0.07537688],
[0.19753086,	-0.06532663],
[-0.19753086,	-0.05527638],
[-0.59259259,	-0.04522613],
[0.54320988,	-0.03517588],
[-0.44444444,	-0.02512563],
[0.59259259,	-0.01507538],
[-0.79012346,	-0.00502513],
[-0.64197531,	0.00502513],
[0.64197531,	0.01507538],
[1.5308642,0.02512563],	

[-0.49382716,	0.03517588],
[0.64197531,	0.04522613],
[-0.74074074,	0.05527638],
[1.48148148,	0.06532663],
[0.88888889,	0.07537688],
[1.58024691,	0.08542714],
[1.48148148,	0.09547739],
[1.43209877,	0.10552764],
[-0.83950617,	0.11557789],
[0.09876543,	0.12562814],
[-0.83950617,	0.13567839],
[-0.88888889,	0.14572864],
[-0.83950617,	0.15577889],
[1.33333333,	0.16582915],
[0.64197531,	0.1758794],
[0.74074074,	0.18592965],
[0.69135802,	0.1959799],
[-0.44444444,	0.20603015],
[0.09876543,	0.2160804],
[0.19753086,	0.22613065],
[0.14814815,	0.2361809],
[-0.64197531,	0.24623116],
[-0.24691358,	0.25628141],
[0.34567901,	0.26633166],
[0.19753086,	0.27638191],
[1.13580247,	0.28643216],
[0.09876543,	0.29648241],
[0.54320988,	0.30653266],
[0.14814815,	0.31658291],
[-0.54320988,	0.32663317],
[-0.24691358,	0.33668342],
[-0.79012346,	0.34673367],
[-0.34567901,	0.35678392],
[0.39506173,	0.36683417],
[-0.19753086,	0.37688442],
[-0.83950617,	0.38693467],
[-0.04938272,	0.39698492],
[1.03703704,	0.40703518],
[-0.19753086,	0.41708543],
[-0.39506173,	0.42713568],
[-0.19753086,	0.43718593],
[-0.54320988,	0.44723618],
[-0.39506173,	0.45728643],
[0.59259259,	0.46733668],
[-0.19753086,	0.47738693],
[-0.09876543,	0.48743719],
[-0.09876543,	0.49748744],
[0.34567901,	0.50753769],
[0.14814815,	0.51758794],
[0.39506173,	0.52763819],

[0.09876543,	0.53768844],
[0.54320988,	0.54773869],
[-0.44444444,	0.55778894],
[0.04938272,	0.5678392],
[-0.2962963,	0.57788945],
[-0.09876543,	0.5879397],
[-0.2962963,	0.59798995],
[0.98765432,	0.6080402],
[-0.34567901,	0.61809045],
[-0.83950617,	0.6281407],
[-0.24691358,	0.63819095],
[0.69135802,	0.64824121],
[0.	, 0.65829146],
[0.2962963,	0.66834171],
[-0.14814815,	0.67839196],
[0.	, 0.68844221],
[-0.19753086,	0.69849246],
[0.19753086,	0.70854271],
[-0.39506173,	0.71859296],
[0.	, 0.72864322],
[0.	, 0.73869347],
[0.79012346,	0.74874372],
[-0.2962963,	0.75879397],
[1.08641975,	0.76884422],
[-0.44444444,	0.77889447],
[1.13580247,	0.78894472],
[-0.04938272,	0.79899497],
[0.04938272,	0.80904523],
[-0.19753086,	0.81909548],
[0.49382716,	0.82914573],
[-0.34567901,	0.83919598],
[0.24691358,	0.84924623],
[-0.2962963,	0.85929648],
[0.88888889,	0.86934673],
[-0.39506173,	0.87939698],
[0.24691358,	0.88944724],
[0.	, 0.89949749],
[-0.09876543,	0.90954774],
[-0.19753086,	0.91959799],
[-0.14814815,	0.92964824],
[0.09876543,	0.93969849],
[0.54320988,	0.94974874],
[-0.04938272,	0.95979899],
[0.44444444,	0.96984925],
[-0.19753086,	0.9798995],
[-0.19753086,	0.98994975],
[-0.2962963,	1.]]

11. Perform any of the clustering algorithms

#K-MEANS CLUSTERING

```
yes = df.Gender
```

```
df = df.drop("Gender",axis= 1)df.head()
```

	Age	Annual Income(k\$)	SpendingScore (1-100)
0	19.0	15.0	39.0
1	21.0	15.0	81.0
2	20.0	16.0	6.0
3	23.0	16.0	77.0
4	31.0	17.0	40.0

```
fromsklearn.clusterimport KMeanskm
```

```
=KMeans(  
    n_clusters=2,random_stat  
    e=10,init="k-  
    means++",n_init=20,max_  
    iter=200  
)
```

```
importwarningswarnings.filterwarnings("ign  
ore")
```

```
km.fit(df)
```

```
KMeans(max_iter=200, n_clusters=2, n_init=20, random_state=10)km.labels_
```

```
array([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, 0,1,1,1,1,1,0,0,1,1,1,1,1, 0,1,1,0,1,1,1,  
0,  
1, 1,0,1,1,1,1,1,1,0,1,1,0,1, 1,0,1,1,0,1,1,  
0,  
0, 1,1,1,1,1,1,0,1,0,1,0,1,1, 1,0,1,1,1,1,1,  
1,  
1, 0,1,0,0,0,1,0,1,1,0,1,0,0, 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, 0,1,0,1,0,1,0,1,0,1,0,1,0, 1,0,1,0,1,0,1,  
0,  
1,0],dtype=int32)
```

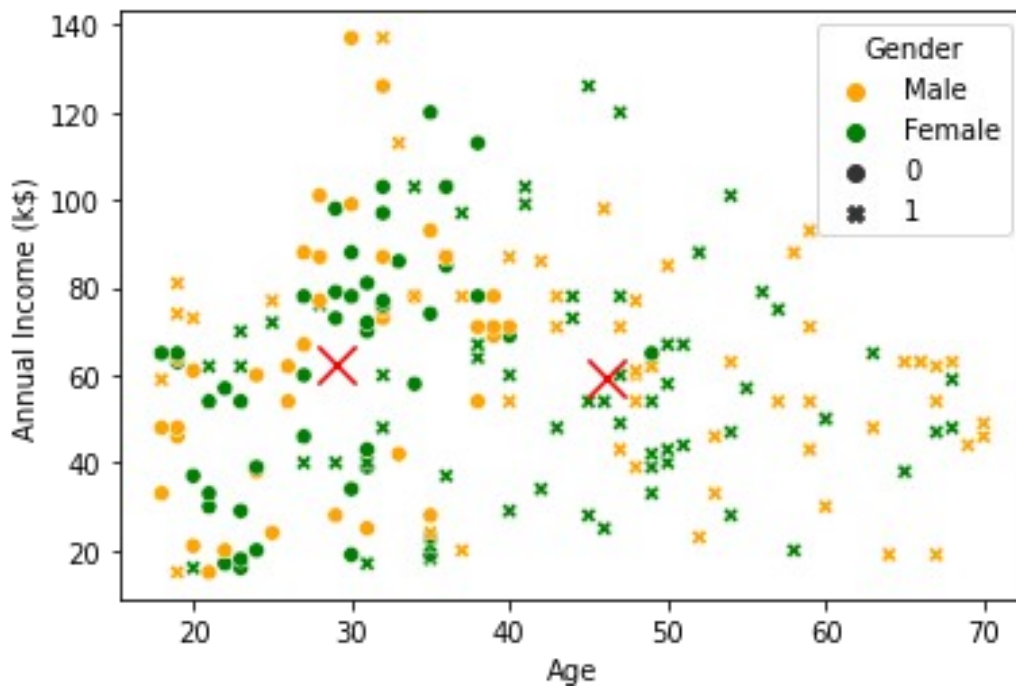
```
df.head()
```

	Age	Annual Income(k\$)	SpendingScore (1-100)
	19.0	15.0	39.0
	121.0	15.0	81.0
	220.0	16.0	6.0
	323.0	16.0	77.0
	431.0	17.0	40.0

```
sns.scatterplot(  
    x="Age",  
    y="AnnualIncome(k$)",data=df,  
    hue="yes",  
    style=km.labels_,palette=["orange","g  
    reen"]  
)
```

```
plt.scatter(  
    km.cluster_centers_[0],km.cluste  
    r_centers_[1],marker="x",  
    s=200,  
    c="red"  
)
```

```
<matplotlib.collections.PathCollection at 0x7f8402caf450>
```



```
from sklearn.metrics import  
silhouette_score
```

```
silhouette_score(df, km.labels_)
```

```
0.2931660
```

```
70535953  
  
k_means_model = cluster.KMeans(n_clusters=3, init='k-means++',  
random_state=0) k_means_model.fit(df)
```

```
k_means_model = cluster.KMeans(n_clusters=3, random_state=0)
```

```
clustered_data = k_means_model.predict(df)
```

```
#Elbow Graph
```

```
wcss = []
```

```
for k in range(1, 10):
```

```
    km = KMeans(n_clusters=k, random_state=1, init="k-means++", n_init=10)
```

```
    km.fit(df)
```

```
    error =
```

```
    km.inertia_
```

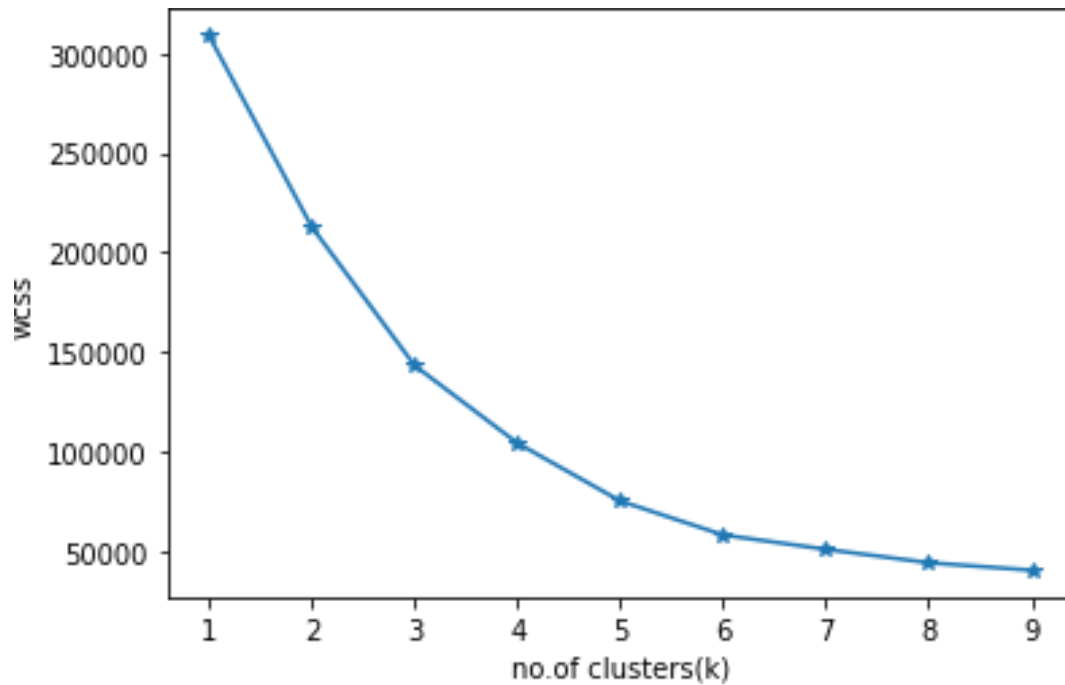
```
    wcss.append(  
    error)
```

```
plt.plot(range(1, 10), wcss, marker="*")
```

```
plt.xlabel("no. of clusters")
```

```
plt.ylabel("wcss")
```

```
plt.show()
```



12. AddClusterdatawithprimaryset

```
df['Clustered_data'] = pd.Series(clustered_data)df.head()
```

	CustomerID	Age	Annual Income(k\$)	SpendingScore(1-100)\0	
15.0	39.0				1.019.0
1	2.0	21.0		15.0	81.0
2	3.0	20.0		16.0	6.0
3	4.0	23.0		16.0	77.0
4	5.0	31.0		17.0	40.0

	Clustered_data	
0	0	
1	0	
2	0	
3	0	
4	0	

13. Splitthedataintodependentandindependentvariables

```
df.head(0)EmptyDat
```

```
aFrame
```

```
Columns: [CustomerID,Gender,Age,AnnualIncome(k$),SpendingScore  
(1-100)]
```

```
Index:[x=df.iloc[:,1:2]
```

x

	Gender
0	Male
1	Male
2	Female
3	Female
4	Female
..19	...
5	Female
196	Female
197	Male
198	Male
199	Male

[200 rows x 1 columns]

y=df.iloc[:,1:]y

	Age	Annual Income(k\$)	Spending	Score	(1-100)	Clustered_data
0	19.0	15.0			39.0	0
1	21.0	15.0			81.0	0
2	20.0	16.0			6.0	0
3	23.0	16.0			77.0	0
4	31.0	17.0			40.0	0
..
195	35.0	120.0			79.0	2
196	45.0	126.0			28.0	2
197	32.0	126.0			74.0	2
198	32.0	137.0			18.0	2
199	30.0	137.0			83.0	2

[200 rows x 4 columns]

14. Splitthedataintotraininandtesting

```
from sklearn.model_selection import train_test_split
df=df.rename(columns={'fit':'fit-feature'})
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

```
((160,1),(40,1),(160,4),(40, 4))
```

```
x_test
```

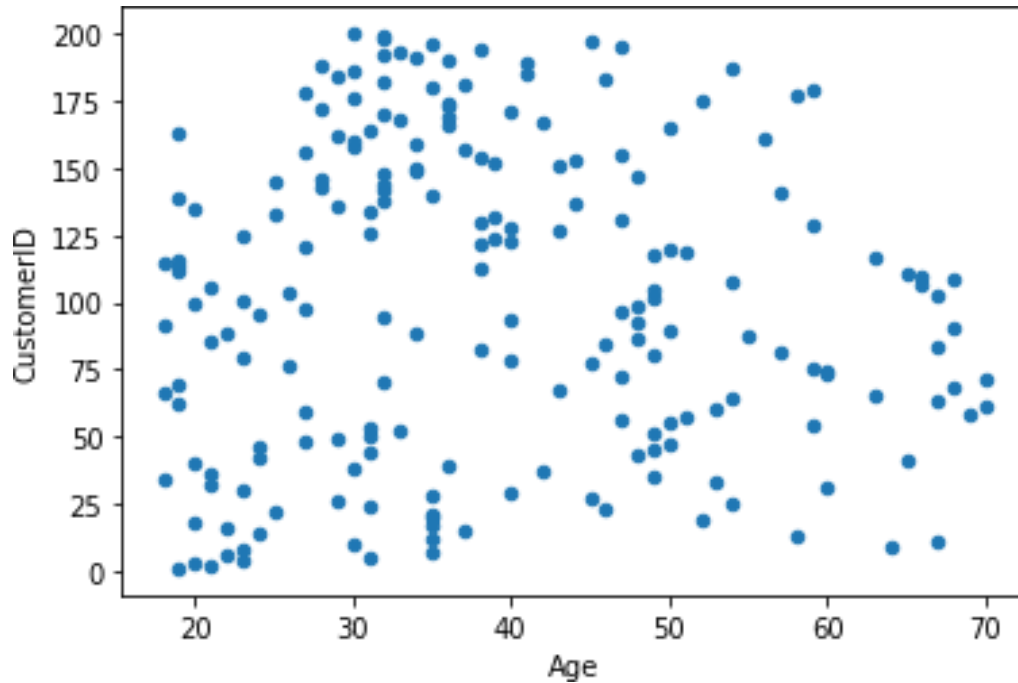

	Gender
18	Male
170	Male
107	Male
98	Male
177	Male
182	Male
5	Female
146	Male
12	Female
152	Female
61	Male
125	Female
180	Female
154	Female
80	Male
7	Female
33	Male
130	Male
37	Female
74	Male
183	Female
145	Male
45	Female
159	Female
60	Male
123	Male
179	Male
185	Male
122	Female
44	Female
16	Female
55	Male
150	Male
111	Female
22	Female
189	Female
129	Male
4	Female
83	Female
106	Female

15. Build the model

```
from sklearn.linear_model import
```

```
LinearRegressionlr=LinearRegression()df.plot.scatter("Age","CustomerID")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f46f13ccd10>
```



```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(x,y)
predict = model.predict(x)
```

```
array([[19.      , 61.02272279, 62.20768706, 1.08321414],
       [21.      , 60.97610085, 60.99784453, 1.07785252],
       [20.      , 60.99941182, 61.6027658 , 1.08053333],
       [23.      , 60.9294789 , 59.78800201, 1.07249089],
       [31.      , 60.74299113, 54.94863191, 1.05104438],
       [22.      , 60.95278987, 60.39292327, 1.0751717 ],
       [35.      , 60.64974724, 52.52894686, 1.04032113],
       [23.      , 60.9294789 , 59.78800201, 1.07249089],
       [64.      , 59.97372906, 34.98623025, 0.96257755],
       [30.      , 60.7663021 , 55.55355317, 1.0537252 ],
       [67.      , 59.90379614, 33.17146646, 0.95453511],
       [35.      , 60.64974724, 52.52894686, 1.04032113],
       [58.      , 60.11359489, 38.61575783, 0.97866243],
       [24.      , 60.90616793, 59.18308075, 1.06981008],
       [37.      , 60.6031253 , 51.31910434, 1.0349595 ],
       [22.      , 60.95278987, 60.39292327, 1.0751717 ],
       [35.      , 60.64974724, 52.52894686, 1.04032113],
       [20.      , 60.99941182, 61.6027658 , 1.08053333],
```

[52.	,	60.25346072,	42.2452854,	0.99474731],
[35.	,	60.64974724,	52.52894686,	1.04032113],
[35.	,	60.64974724,	52.52894686,	1.04032113],
[25.	,	60.88285696,	58.57815948,	1.06712926],
[46.	,	60.39332655,	45.87481297,	1.01083219],
[31.	,	60.74299113,	54.94863191,	1.05104438],
[54.	,	60.20683878,	41.03544287,	0.98938568],
[29.	,	60.78961307,	56.15847443,	1.05640601],
[45.	,	60.41663752,	46.47973424,	1.013513],
[35.	,	60.64974724,	52.52894686,	1.04032113],
[40.	,	60.53319238,	49.50434055,	1.02691706],
[23.	,	60.9294789,	59.78800201,	1.07249089],
[60.	,	60.06697295,	37.4059153,	0.9733008],
[21.	,	60.97610085,	60.99784453,	1.07785252],
[53.	,	60.23014975,	41.64036414,	0.99206649],
[18.	,	61.04603376,	62.81260832,	1.08589496],
[49.	,	60.32339364,	44.06004919,	1.00278975],
[21.	,	60.97610085,	60.99784453,	1.07785252],
[42.	,	60.48657044,	48.29449802,	1.02155544],
[30.	,	60.7663021,	55.55355317,	1.0537252],
[36.	,	60.62643627,	51.9240256,	1.03764032],
[20.	,	60.99941182,	61.6027658,	1.08053333],
[65.	,	59.95041809,	34.38130899,	0.95989673],
[24.	,	60.90616793,	59.18308075,	1.06981008],
[48.	,	60.34670461,	44.66497045,	1.00547056],
[31.	,	60.74299113,	54.94863191,	1.05104438],
[49.	,	60.32339364,	44.06004919,	1.00278975],
[24.	,	60.90616793,	59.18308075,	1.06981008],
[50.	,	60.30008266,	43.45512792,	1.00010893],
[27.	,	60.83623502,	57.36831696,	1.06176764],
[29.	,	60.78961307,	56.15847443,	1.05640601],
[31.	,	60.74299113,	54.94863191,	1.05104438],
[49.	,	60.32339364,	44.06004919,	1.00278975],
[33.	,	60.69636918,	53.73878938,	1.04568276],
[31.	,	60.74299113,	54.94863191,	1.05104438],
[59.	,	60.09028392,	38.01083656,	0.97598161],
[50.	,	60.30008266,	43.45512792,	1.00010893],
[47.	,	60.37001558,	45.26989171,	1.00815137],
[51.	,	60.27677169,	42.85020666,	0.99742812],
[69.	,	59.8571742,	31.96162394,	0.94917348],
[27.	,	60.83623502,	57.36831696,	1.06176764],
[53.	,	60.23014975,	41.64036414,	0.99206649],
[70.	,	59.83386323,	31.35670268,	0.94649267],
[19.	,	61.02272279,	62.20768706,	1.08321414],
[67.	,	59.90379614,	33.17146646,	0.95453511],
[54.	,	60.20683878,	41.03544287,	0.98938568],
[63.	,	59.99704003,	35.59115151,	0.96525836],
[18.	,	61.04603376,	62.81260832,	1.08589496],
[43.	,	60.46325947,	47.68957676,	1.01887462],
[68.	,	59.88048517,	32.5665452,	0.95185429],

[19.	,	61.02272279,	62.20768706,	1.08321414],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[70.	,	59.83386323,	31.35670268,	0.94649267],
[47.	,	60.37001558,	45.26989171,	1.00815137],
[60.	,	60.06697295,	37.4059153,	0.9733008],
[60.	,	60.06697295,	37.4059153,	0.9733008],
[59.	,	60.09028392,	38.01083656,	0.97598161],
[26.	,	60.85954599,	57.97323822,	1.06444845],
[45.	,	60.41663752,	46.47973424,	1.013513],
[40.	,	60.53319238,	49.50434055,	1.02691706],
[23.	,	60.9294789,	59.78800201,	1.07249089],
[49.	,	60.32339364,	44.06004919,	1.00278975],
[57.	,	60.13690586,	39.22067909,	0.98134324],
[38.	,	60.57981433,	50.71418307,	1.03227869],
[67.	,	59.90379614,	33.17146646,	0.95453511],
[46.	,	60.39332655,	45.87481297,	1.01083219],
[21.	,	60.97610085,	60.99784453,	1.07785252],
[48.	,	60.34670461,	44.66497045,	1.00547056],
[55.	,	60.18352781,	40.43052161,	0.98670487],
[22.	,	60.95278987,	60.39292327,	1.0751717],
[34.	,	60.67305821,	53.13386812,	1.04300194],
[50.	,	60.30008266,	43.45512792,	1.00010893],
[68.	,	59.88048517,	32.5665452,	0.95185429],
[18.	,	61.04603376,	62.81260832,	1.08589496],
[48.	,	60.34670461,	44.66497045,	1.00547056],
[40.	,	60.53319238,	49.50434055,	1.02691706],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[24.	,	60.90616793,	59.18308075,	1.06981008],
[47.	,	60.37001558,	45.26989171,	1.00815137],
[27.	,	60.83623502,	57.36831696,	1.06176764],
[48.	,	60.34670461,	44.66497045,	1.00547056],
[20.	,	60.99941182,	61.6027658,	1.08053333],
[23.	,	60.9294789,	59.78800201,	1.07249089],
[49.	,	60.32339364,	44.06004919,	1.00278975],
[67.	,	59.90379614,	33.17146646,	0.95453511],
[26.	,	60.85954599,	57.97323822,	1.06444845],
[49.	,	60.32339364,	44.06004919,	1.00278975],
[21.	,	60.97610085,	60.99784453,	1.07785252],
[66.	,	59.92710712,	33.77638773,	0.95721592],
[54.	,	60.20683878,	41.03544287,	0.98938568],
[68.	,	59.88048517,	32.5665452,	0.95185429],
[66.	,	59.92710712,	33.77638773,	0.95721592],
[65.	,	59.95041809,	34.38130899,	0.95989673],
[19.	,	61.02272279,	62.20768706,	1.08321414],
[38.	,	60.57981433,	50.71418307,	1.03227869],
[19.	,	61.02272279,	62.20768706,	1.08321414],
[18.	,	61.04603376,	62.81260832,	1.08589496],
[19.	,	61.02272279,	62.20768706,	1.08321414],
[63.	,	59.99704003,	35.59115151,	0.96525836],
[49.	,	60.32339364,	44.06004919,	1.00278975],

[51.	,	60.27677169,	42.85020666,	0.99742812],
[50.	,	60.30008266,	43.45512792,	1.00010893],
[27.	,	60.83623502,	57.36831696,	1.06176764],
[38.	,	60.57981433,	50.71418307,	1.03227869],
[40.	,	60.53319238,	49.50434055,	1.02691706],
[39.	,	60.55650335,	50.10926181,	1.02959788],
[23.	,	60.9294789,	59.78800201,	1.07249089],
[31.	,	60.74299113,	54.94863191,	1.05104438],
[43.	,	60.46325947,	47.68957676,	1.01887462],
[40.	,	60.53319238,	49.50434055,	1.02691706],
[59.	,	60.09028392,	38.01083656,	0.97598161],
[38.	,	60.57981433,	50.71418307,	1.03227869],
[47.	,	60.37001558,	45.26989171,	1.00815137],
[39.	,	60.55650335,	50.10926181,	1.02959788],
[25.	,	60.88285696,	58.57815948,	1.06712926],
[31.	,	60.74299113,	54.94863191,	1.05104438],
[20.	,	60.99941182,	61.6027658,	1.08053333],
[29.	,	60.78961307,	56.15847443,	1.05640601],
[44.	,	60.4399485,	47.0846555,	1.01619381],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[19.	,	61.02272279,	62.20768706,	1.08321414],
[35.	,	60.64974724,	52.52894686,	1.04032113],
[57.	,	60.13690586,	39.22067909,	0.98134324],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[28.	,	60.81292404,	56.7633957,	1.05908682],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[25.	,	60.88285696,	58.57815948,	1.06712926],
[28.	,	60.81292404,	56.7633957,	1.05908682],
[48.	,	60.34670461,	44.66497045,	1.00547056],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[34.	,	60.67305821,	53.13386812,	1.04300194],
[34.	,	60.67305821,	53.13386812,	1.04300194],
[43.	,	60.46325947,	47.68957676,	1.01887462],
[39.	,	60.55650335,	50.10926181,	1.02959788],
[44.	,	60.4399485,	47.0846555,	1.01619381],
[38.	,	60.57981433,	50.71418307,	1.03227869],
[47.	,	60.37001558,	45.26989171,	1.00815137],
[27.	,	60.83623502,	57.36831696,	1.06176764],
[37.	,	60.6031253,	51.31910434,	1.0349595],
[30.	,	60.7663021,	55.55355317,	1.0537252],
[34.	,	60.67305821,	53.13386812,	1.04300194],
[30.	,	60.7663021,	55.55355317,	1.0537252],
[56.	,	60.16021683,	39.82560035,	0.98402405],
[29.	,	60.78961307,	56.15847443,	1.05640601],
[19.	,	61.02272279,	62.20768706,	1.08321414],
[31.	,	60.74299113,	54.94863191,	1.05104438],
[50.	,	60.30008266,	43.45512792,	1.00010893],
[36.	,	60.62643627,	51.9240256,	1.03764032],
[42.	,	60.48657044,	48.29449802,	1.02155544],
[33.	,	60.69636918,	53.73878938,	1.04568276],

```
[36.      , 60.62643627, 51.9240256, 1.03764032],
[32.      , 60.71968016, 54.34371065, 1.04836357],
[40.      , 60.53319238, 49.50434055, 1.02691706],
[28.      , 60.81292404, 56.7633957, 1.05908682],
[36.      , 60.62643627, 51.9240256, 1.03764032],
[36.      , 60.62643627, 51.9240256, 1.03764032],
[52.      , 60.25346072, 42.2452854, 0.99474731],
[30.      , 60.7663021, 55.55355317, 1.0537252],
[58.      , 60.11359489, 38.61575783, 0.97866243],
[27.      , 60.83623502, 57.36831696, 1.06176764],
[59.      , 60.09028392, 38.01083656, 0.97598161],
[35.      , 60.64974724, 52.52894686, 1.04032113],
[37.      , 60.6031253, 51.31910434, 1.0349595],
[32.      , 60.71968016, 54.34371065, 1.04836357],
[46.      , 60.39332655, 45.87481297, 1.01083219],
[29.      , 60.78961307, 56.15847443, 1.05640601],
[41.      , 60.50988141, 48.89941929, 1.02423625],
[30.      , 60.7663021, 55.55355317, 1.0537252],
[54.      , 60.20683878, 41.03544287, 0.98938568],
[28.      , 60.81292404, 56.7633957, 1.05908682],
[41.      , 60.50988141, 48.89941929, 1.02423625],
[36.      , 60.62643627, 51.9240256, 1.03764032],
[34.      , 60.67305821, 53.13386812, 1.04300194],
[32.      , 60.71968016, 54.34371065, 1.04836357],
[33.      , 60.69636918, 53.73878938, 1.04568276],
[38.      , 60.57981433, 50.71418307, 1.03227869],
[47.      , 60.37001558, 45.26989171, 1.00815137],
[35.      , 60.64974724, 52.52894686, 1.04032113],
[45.      , 60.41663752, 46.47973424, 1.013513],
[32.      , 60.71968016, 54.34371065, 1.04836357],
[32.      , 60.71968016, 54.34371065, 1.04836357],
[30.      , 60.7663021, 55.55355317, 1.0537252]])
```

16. Train the

`model``train=df.sample(frac=0.8,random_state=200)train`

	CustomerID	Age	Annual	Income (k\$)	Spending	Score	(1-100)	\
121	122.0	38.0		67.0			40.0	
169	170.0	32.0		87.0			63.0	
194	195.0	47.0		120.0			16.0	
125	126.0	31.0		70.0			77.0	
36	37.0	42.0		34.0			17.0	
..	
90	91.0	68.0		59.0			55.0	
162	163.0	19.0		81.0			5.0	
3	4.0	23.0		16.0			77.0	
120	121.0	27.0		67.0			56.0	
95	96.0	24.0		60.0			52.0	

	Clustered_data
121	1
169	2
194	2
125	1
36	0
..	...
90	1
162	2
3	0
120	1
95	1

[160rows x5 columns]

pred_train = model.predict(x_train)pred_train

```
array([[20.      , 60.99941182, 61.6027658, 1.08053333],
 [43.      , 60.46325947, 47.68957676, 1.01887462],
 [45.      , 60.41663752, 46.47973424, 1.013513],
 [19.      , 61.02272279, 62.20768706, 1.08321414],
 [36.      , 60.62643627, 51.9240256, 1.03764032],
 [54.      , 60.20683878, 41.03544287, 0.98938568],
 [64.      , 59.97372906, 34.98623025, 0.96257755],
 [26.      , 60.85954599, 57.97323822, 1.06444845],
 [51.      , 60.27677169, 42.85020666, 0.99742812],
 [32.      , 60.71968016, 54.34371065, 1.04836357],
 [47.      , 60.37001558, 45.26989171, 1.00815137],
 [23.      , 60.9294789, 59.78800201, 1.07249089],
 [41.      , 60.50988141, 48.89941929, 1.02423625],
 [27.      , 60.83623502, 57.36831696, 1.06176764],
 [34.      , 60.67305821, 53.13386812, 1.04300194],
 [54.      , 60.20683878, 41.03544287, 0.98938568],
 [60.      , 60.06697295, 37.4059153, 0.9733008],
 [56.      , 60.16021683, 39.82560035, 0.98402405],
 [65.      , 59.95041809, 34.38130899, 0.95989673],
 [51.      , 60.27677169, 42.85020666, 0.99742812],
 [39.      , 60.55650335, 50.10926181, 1.02959788],
 [47.      , 60.37001558, 45.26989171, 1.00815137],
 [32.      , 60.71968016, 54.34371065, 1.04836357],
 [35.      , 60.64974724, 52.52894686, 1.04032113],
 [38.      , 60.57981433, 50.71418307, 1.03227869],
 [48.      , 60.34670461, 44.66497045, 1.00547056],
 [50.      , 60.30008266, 43.45512792, 1.00010893],
 [31.      , 60.74299113, 54.94863191, 1.05104438],
 [33.      , 60.69636918, 53.73878938, 1.04568276],
 [55.      , 60.18352781, 40.43052161, 0.98670487],
 [35.      , 60.64974724, 52.52894686, 1.04032113],
```

[68.	,	59.88048517,	32.5665452,	0.95185429],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[49.	,	60.32339364,	44.06004919,	1.00278975],
[25.	,	60.88285696,	58.57815948,	1.06712926],
[50.	,	60.30008266,	43.45512792,	1.00010893],
[66.	,	59.92710712,	33.77638773,	0.95721592],
[37.	,	60.6031253,	51.31910434,	1.0349595],
[35.	,	60.64974724,	52.52894686,	1.04032113],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[28.	,	60.81292404,	56.7633957,	1.05908682],
[50.	,	60.30008266,	43.45512792,	1.00010893],
[19.	,	61.02272279,	62.20768706,	1.08321414],
[35.	,	60.64974724,	52.52894686,	1.04032113],
[68.	,	59.88048517,	32.5665452,	0.95185429],
[67.	,	59.90379614,	33.17146646,	0.95453511],
[20.	,	60.99941182,	61.6027658,	1.08053333],
[53.	,	60.23014975,	41.64036414,	0.99206649],
[44.	,	60.4399485,	47.0846555,	1.01619381],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[31.	,	60.74299113,	54.94863191,	1.05104438],
[67.	,	59.90379614,	33.17146646,	0.95453511],
[47.	,	60.37001558,	45.26989171,	1.00815137],
[60.	,	60.06697295,	37.4059153,	0.9733008],
[45.	,	60.41663752,	46.47973424,	1.013513],
[59.	,	60.09028392,	38.01083656,	0.97598161],
[30.	,	60.7663021,	55.55355317,	1.0537252],
[43.	,	60.46325947,	47.68957676,	1.01887462],
[40.	,	60.53319238,	49.50434055,	1.02691706],
[38.	,	60.57981433,	50.71418307,	1.03227869],
[34.	,	60.67305821,	53.13386812,	1.04300194],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[49.	,	60.32339364,	44.06004919,	1.00278975],
[19.	,	61.02272279,	62.20768706,	1.08321414],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[65.	,	59.95041809,	34.38130899,	0.95989673],
[24.	,	60.90616793,	59.18308075,	1.06981008],
[63.	,	59.99704003,	35.59115151,	0.96525836],
[33.	,	60.69636918,	53.73878938,	1.04568276],
[24.	,	60.90616793,	59.18308075,	1.06981008],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[31.	,	60.74299113,	54.94863191,	1.05104438],
[29.	,	60.78961307,	56.15847443,	1.05640601],
[48.	,	60.34670461,	44.66497045,	1.00547056],
[24.	,	60.90616793,	59.18308075,	1.06981008],
[29.	,	60.78961307,	56.15847443,	1.05640601],
[31.	,	60.74299113,	54.94863191,	1.05104438],
[54.	,	60.20683878,	41.03544287,	0.98938568],
[29.	,	60.78961307,	56.15847443,	1.05640601],
[35.	,	60.64974724,	52.52894686,	1.04032113],
[22.	,	60.95278987,	60.39292327,	1.0751717],

[23.	,	60.9294789,	59.78800201,	1.07249089],
[49.	,	60.32339364,	44.06004919,	1.00278975],
[31.	,	60.74299113,	54.94863191,	1.05104438],
[23.	,	60.9294789,	59.78800201,	1.07249089],
[45.	,	60.41663752,	46.47973424,	1.013513],
[23.	,	60.9294789,	59.78800201,	1.07249089],
[63.	,	59.99704003,	35.59115151,	0.96525836],
[50.	,	60.30008266,	43.45512792,	1.00010893],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[35.	,	60.64974724,	52.52894686,	1.04032113],
[19.	,	61.02272279,	62.20768706,	1.08321414],
[21.	,	60.97610085,	60.99784453,	1.07785252],
[38.	,	60.57981433,	50.71418307,	1.03227869],
[27.	,	60.83623502,	57.36831696,	1.06176764],
[28.	,	60.81292404,	56.7633957,	1.05908682],
[37.	,	60.6031253,	51.31910434,	1.0349595],
[18.	,	61.04603376,	62.81260832,	1.08589496],
[30.	,	60.7663021,	55.55355317,	1.0537252],
[35.	,	60.64974724,	52.52894686,	1.04032113],
[50.	,	60.30008266,	43.45512792,	1.00010893],
[67.	,	59.90379614,	33.17146646,	0.95453511],
[21.	,	60.97610085,	60.99784453,	1.07785252],
[69.	,	59.8571742,	31.96162394,	0.94917348],
[18.	,	61.04603376,	62.81260832,	1.08589496],
[21.	,	60.97610085,	60.99784453,	1.07785252],
[27.	,	60.83623502,	57.36831696,	1.06176764],
[19.	,	61.02272279,	62.20768706,	1.08321414],
[48.	,	60.34670461,	44.66497045,	1.00547056],
[21.	,	60.97610085,	60.99784453,	1.07785252],
[25.	,	60.88285696,	58.57815948,	1.06712926],
[36.	,	60.62643627,	51.9240256,	1.03764032],
[20.	,	60.99941182,	61.6027658,	1.08053333],
[36.	,	60.62643627,	51.9240256,	1.03764032],
[31.	,	60.74299113,	54.94863191,	1.05104438],
[59.	,	60.09028392,	38.01083656,	0.97598161],
[30.	,	60.7663021,	55.55355317,	1.0537252],
[59.	,	60.09028392,	38.01083656,	0.97598161],
[49.	,	60.32339364,	44.06004919,	1.00278975],
[40.	,	60.53319238,	49.50434055,	1.02691706],
[18.	,	61.04603376,	62.81260832,	1.08589496],
[39.	,	60.55650335,	50.10926181,	1.02959788],
[21.	,	60.97610085,	60.99784453,	1.07785252],
[42.	,	60.48657044,	48.29449802,	1.02155544],
[40.	,	60.53319238,	49.50434055,	1.02691706],
[58.	,	60.11359489,	38.61575783,	0.97866243],
[53.	,	60.23014975,	41.64036414,	0.99206649],
[28.	,	60.81292404,	56.7633957,	1.05908682],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[32.	,	60.71968016,	54.34371065,	1.04836357],
[23.	,	60.9294789,	59.78800201,	1.07249089],

```

[20.      , 60.99941182, 61.6027658, 1.08053333],
[67.      , 59.90379614, 33.17146646, 0.95453511],
[49.      , 60.32339364, 44.06004919, 1.00278975],
[19.      , 61.02272279, 62.20768706, 1.08321414],
[34.      , 60.67305821, 53.13386812, 1.04300194],
[38.      , 60.57981433, 50.71418307, 1.03227869],
[60.      , 60.06697295, 37.4059153, 0.9733008],
[40.      , 60.53319238, 49.50434055, 1.02691706],
[29.      , 60.78961307, 56.15847443, 1.05640601],
[36.      , 60.62643627, 51.9240256, 1.03764032],
[38.      , 60.57981433, 50.71418307, 1.03227869],
[41.      , 60.50988141, 48.89941929, 1.02423625],
[52.      , 60.25346072, 42.2452854, 0.99474731],
[34.      , 60.67305821, 53.13386812, 1.04300194],
[20.      , 60.99941182, 61.6027658, 1.08053333],
[27.      , 60.83623502, 57.36831696, 1.06176764],
[57.      , 60.13690586, 39.22067909, 0.98134324],
[34.      , 60.67305821, 53.13386812, 1.04300194],
[70.      , 59.83386323, 31.35670268, 0.94649267],
[22.      , 60.95278987, 60.39292327, 1.0751717],
[42.      , 60.48657044, 48.29449802, 1.02155544],
[25.      , 60.88285696, 58.57815948, 1.06712926],
[30.      , 60.7663021, 55.55355317, 1.0537252],
[26.      , 60.85954599, 57.97323822, 1.06444845],
[68.      , 59.88048517, 32.5665452, 0.95185429],
[33.      , 60.69636918, 53.73878938, 1.04568276],
[49.      , 60.32339364, 44.06004919, 1.00278975],
[27.      , 60.83623502, 57.36831696, 1.06176764],
[36.      , 60.62643627, 51.9240256, 1.03764032]]

```

17. TesttheModel

y_test

	Age	Annual	Income	(k\$)	SpendingScore(1-100)	Clustered_data
18	52.0			23.0	29.0	0
170	40.0			87.0	13.0	2
107	54.0			63.0	46.0	1
98	48.0			61.0	42.0	1
177	27.0			88.0	69.0	2
182	46.0			98.0	15.0	2
5	22.0			17.0	76.0	0
146	48.0			77.0	36.0	2
12	58.0			20.0	15.0	0
152	44.0			78.0	20.0	2
61	19.0			46.0	55.0	0
125	31.0			70.0	77.0	1
180	37.0			97.0	32.0	2
154	47.0			78.0	16.0	2
80	57.0			54.0	51.0	1
7	23.0			18.0	94.0	0

33	18.0	33.0	92.0	0
130	47.0	71.0	9.0	1
37	30.0	34.0	73.0	0
74	59.0	54.0	47.0	1
183	29.0	98.0	88.0	2
145	28.0	77.0	97.0	2
45	24.0	39.0	65.0	0
159	30.0	78.0	73.0	2
60	70.0	46.0	56.0	0
123	39.0	69.0	91.0	1
179	35.0	93.0	90.0	2
185	30.0	99.0	97.0	2
122	40.0	69.0	58.0	1
44	49.0	39.0	28.0	0
16	35.0	21.0	35.0	0
55	47.0	43.0	41.0	0
150	43.0	78.0	17.0	2
111	19.0	63.0	54.0	1
22	46.0	25.0	5.0	0
189	36.0	103.0	85.0	2
129	38.0	71.0	75.0	1
4	31.0	17.0	40.0	0
83	46.0	54.0	44.0	1
106	66.0	63.0	50.0	1

pred_test=model.predict(x_test)pred_test

```
array([[52.      , 60.25346072, 42.2452854, 0.99474731],
       [40.      , 60.53319238, 49.50434055, 1.02691706],
       [54.      , 60.20683878, 41.03544287, 0.98938568],
       [48.      , 60.34670461, 44.66497045, 1.00547056],
       [27.      , 60.83623502, 57.36831696, 1.06176764],
       [46.      , 60.39332655, 45.87481297, 1.01083219],
       [22.      , 60.95278987, 60.39292327, 1.0751717],
       [48.      , 60.34670461, 44.66497045, 1.00547056],
       [58.      , 60.11359489, 38.61575783, 0.97866243],
       [44.      , 60.4399485, 47.0846555, 1.01619381],
       [19.      , 61.02272279, 62.20768706, 1.08321414],
       [31.      , 60.74299113, 54.94863191, 1.05104438],
       [37.      , 60.6031253, 51.31910434, 1.0349595],
       [47.      , 60.37001558, 45.26989171, 1.00815137],
       [57.      , 60.13690586, 39.22067909, 0.98134324],
       [23.      , 60.9294789, 59.78800201, 1.07249089],
       [18.      , 61.04603376, 62.81260832, 1.08589496],
       [47.      , 60.37001558, 45.26989171, 1.00815137],
       [30.      , 60.7663021, 55.55355317, 1.0537252],
       [59.      , 60.09028392, 38.01083656, 0.97598161],
       [29.      , 60.78961307, 56.15847443, 1.05640601],
       [28.      , 60.81292404, 56.7633957, 1.05908682],
       [24.      , 60.90616793, 59.18308075, 1.06981008],
```

```

[30.      ,      60.7663021, 55.55355317, 1.0537252],
[70.      ,      59.83386323, 31.35670268, 0.94649267],
[39.      ,      60.55650335, 50.10926181, 1.02959788],
[35.      ,      60.64974724, 52.52894686, 1.04032113],
[30.      ,      60.7663021, 55.55355317, 1.0537252],
[40.      ,      60.53319238, 49.50434055, 1.02691706],
[49.      ,      60.32339364, 44.06004919, 1.00278975],
[35.      ,      60.64974724, 52.52894686, 1.04032113],
[47.      ,      60.37001558, 45.26989171, 1.00815137],
[43.      ,      60.46325947, 47.68957676, 1.01887462],
[19.      ,      61.02272279, 62.20768706, 1.08321414],
[46.      ,      60.39332655, 45.87481297, 1.01083219],
[36.      ,      60.62643627, 51.9240256, 1.03764032],
[38.      ,      60.57981433, 50.71418307, 1.03227869],
[31.      ,      60.74299113, 54.94863191, 1.05104438],
[46.      ,      60.39332655, 45.87481297, 1.01083219],
[66.      ,      59.92710712, 33.77638773, 0.95721592]])

```

```

from sklearn.linear_model import LinearRegressionlr

```

```

=LinearRegression()

```

18. Measure the performance using evaluation metrics

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")

customer = pd.read_excel("Mall_Customers.xlsx")
x = df.iloc[:, 1:]
```

	Age	Annual Income(k\$)	Spending	Score	(1-100)
0	19.0	15.0			39.0
1	21.0	15.0			81.0
2	20.0	16.0			6.0
3	23.0	16.0			77.0
4	31.0	17.0			40.0
..
195	35.0	120.0			79.0
196	45.0	126.0			28.0
197	32.0	126.0			74.0
198	32.0	137.0			18.0
199	30.0	137.0			83.0

[200 rows x 3 columns]

```
y = df.iloc[:, 1:]
```

	Age	Annual Income(k\$)	Spending	Score	(1-100)
0	19.0	15.0			39.0
1	21.0	15.0			81.0
2	20.0	16.0			6.0
3	23.0	16.0			77.0
4	31.0	17.0			40.0
..
195	35.0	120.0			79.0
196	45.0	126.0			28.0
197	32.0	126.0			74.0
198	32.0	137.0			18.0
199	30.0	137.0			83.0

[200 rows x 3 columns]

```
from sklearn.model_selection import train_test_split
```

```
df=df.rename(columns={'fit':'fit-feature'})
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

```
x_train.shape,x_test.shape,y_train.shape,y_test.shape((160,3),(40,  
3),(160,3),(40, 3))
```

```
x_test
```

	Age	Annual	Income	(k\$)	Spending	Score	(1-100)
18	52.0			23.0			29.0
170	40.0			87.0			13.0
107	54.0			63.0			46.0
98	48.0			61.0			42.0
177	27.0			88.0			69.0
182	46.0			98.0			15.0
5	22.0			17.0			76.0
146	48.0			77.0			36.0
12	58.0			20.0			15.0
152	44.0			78.0			20.0
61	19.0			46.0			55.0
125	31.0			70.0			77.0
180	37.0			97.0			32.0
154	47.0			78.0			16.0
80	57.0			54.0			51.0
7	23.0			18.0			94.0
33	18.0			33.0			92.0
130	47.0			71.0			9.0
37	30.0			34.0			73.0
74	59.0			54.0			47.0
183	29.0			98.0			88.0
145	28.0			77.0			97.0
45	24.0			39.0			65.0
159	30.0			78.0			73.0
60	70.0			46.0			56.0
123	39.0			69.0			91.0
179	35.0			93.0			90.0
185	30.0			99.0			97.0
122	40.0			69.0			58.0
44	49.0			39.0			28.0
16	35.0			21.0			35.0
55	47.0			43.0			41.0
150	43.0			78.0			17.0
111	19.0			63.0			54.0
22	46.0			25.0			5.0
189	36.0			103.0			85.0
129	38.0			71.0			75.0
4	31.0			17.0			40.0

83	46.0	54.0	44.0
106	66.0	63.0	50.0

```
from sklearn.metrics import r2_score
```

```
from sklearn.linear_model import LinearRegression
```

```
lr = LinearRegression()
```

```
df
```

```
df.replace("Male",2).fit(x_train,y
```

```
_train).coef_,
```

```
lr.intercept_
```

```
(array([[1.00000000e+00, 1.32312315e-17,-7.16567384e-18],[-1.26527940e-16,
1.00000000e+00,-3.33066907e-16],[3.03558876e-
17,0.00000000e+00,1.00000000e+00]]),
array([-1.42108547e-14,4.26325641e-14,-1.42108547e-14]))
```

```
y_pred = lr.predict(x_test)
```

```
array([[ 52.,  23.,  29.],
[ 40.,  87.,  13.],
[ 54.,  63.,  46.],
[ 48.,  61.,  42.],
[ 27.,  88.,  69.],
[ 46.,  98.,  15.],
[ 22.,  17.,  76.],
[ 48.,  77.,  36.],
[ 58.,  20.,  15.],
[ 44.,  78.,  20.],
[ 19.,  46.,  55.],
[ 31.,  70.,  77.],
[ 37.,  97.,  32.],
[ 47.,  78.,  16.],
[ 57.,  54.,  51.],
[ 23.,  18.,  94.],
[ 18.,  33.,  92.],
[ 47.,  71.,   9.],
[ 30.,  34.,  73.],
[ 59.,  54.,  47.],
[ 29.,  98.,  88.],
[ 28.,  77.,  97.],
[ 24.,  39.,  65.],
[ 30.,  78.,  73.],
[ 70.,  46.,  56.],
[ 39.,  69.,  91.],
[ 35.,  93.,  90.]])
```

```
[ 30., 99., 97.],  
[ 40., 69., 58.],  
[ 49., 39., 28.],  
[ 35., 21., 35.],  
[ 47., 43., 41.],  
[ 43., 78., 17.],  
[ 19., 63., 54.],  
[ 46., 25., 5.],  
[ 36., 103., 85.],  
[ 38., 71., 75.],  
[ 31., 17., 40.],  
[ 46., 54., 44.],  
[ 66., 63., 50.]])
```

```
score = r2_score(y_test,y_pred)score
```

```
1.0
```