**1.Write a python program to Prepare Scatter Plot (Use Forge Dataset / Iris Dataset).**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

data=pd.read\_csv("iris.csv")

print(data)

x=data["SepalLengthCm"]

y=data["SepalWidthCm"]

print(x)

print(y)

plt.scatter(x,y,c="red")

plt.title("IRIS CSV")

plt.xlabel("SepalLengthCm")

plt.ylabel("SepalWidthCm")

plt.show()

**OUTPUT:**

Id SepalLengthCm ... PetalWidthCm Species

0 1 5.1 ... 0.2 Iris-setosa

1 2 4.9 ... 0.2 Iris-setosa

2 3 4.7 ... 0.2 Iris-setosa

3 4 4.6 ... 0.2 Iris-setosa

4 5 5.0 ... 0.2 Iris-setosa

.. ... ... ... ... ...

145 146 6.7 ... 2.3 Iris-virginica

146 147 6.3 ... 1.9 Iris-virginica

147 148 6.5 ... 2.0 Iris-virginica

148 149 6.2 ... 2.3 Iris-virginica

149 150 5.9 ... 1.8 Iris-virginica

[150 rows x 6 columns]

0 5.1

1 4.9

2 4.7

3 4.6

4 5.0

...

145 6.7

146 6.3

147 6.5

148 6.2

149 5.9

Name: SepalLengthCm, Length: 150, dtype: float64

0 3.5

1 3.0

2 3.2

3 3.1

4 3.6

...

145 3.0

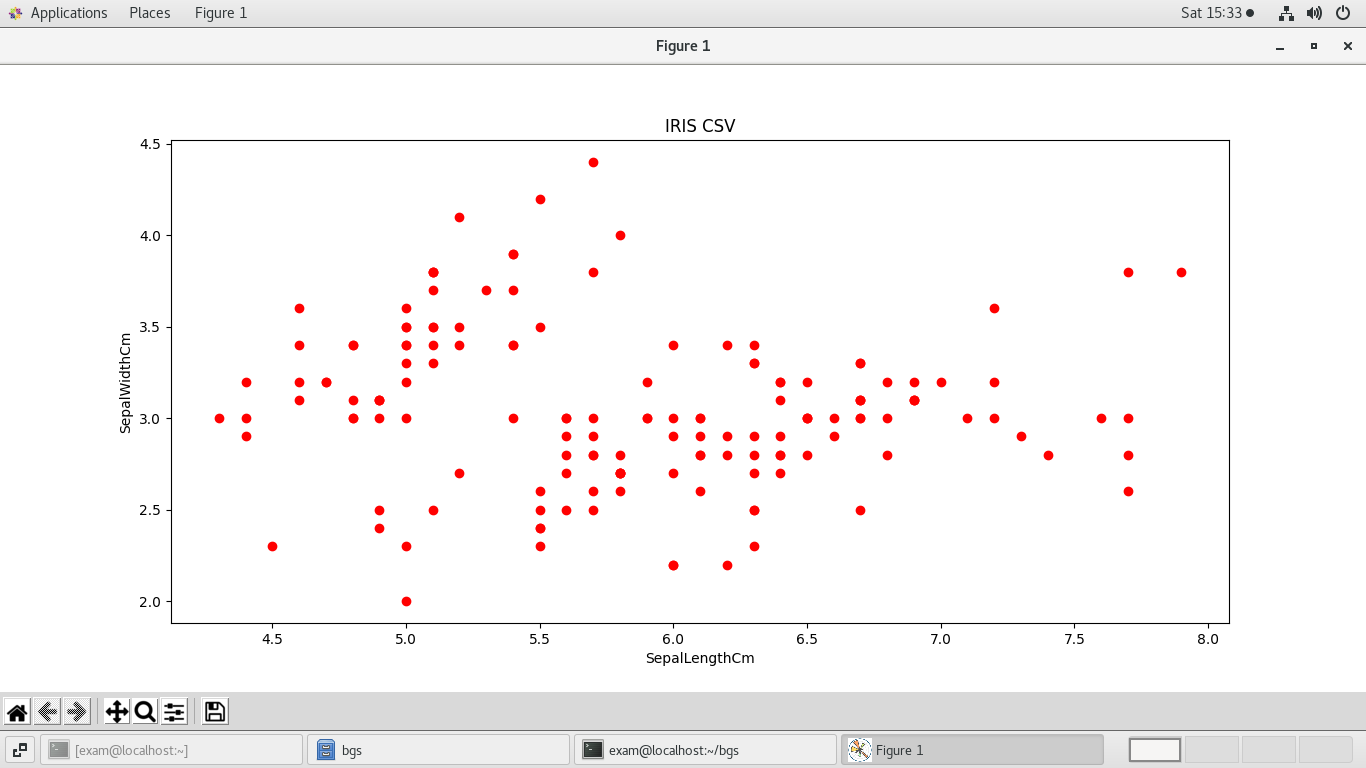
146 2.5

147 3.0

148 3.4

149 3.0

Name: SepalWidthCm, Length: 150, dtype: float64



**2. Write a python program to find all null values in a given data set and remove them.**

import pandas as pd

import numpy as np

data=pd.read\_csv("ass2\_data.csv")

print(data)

print(data.isnull())

print(data.notnull())

data1=data.dropna(axis=0,how="any")

print(data1)

data["m1"]=data["m1"].replace(np.NaN,data["m1"].mean())

data["m2"]=data["m2"].replace(np.NaN,data["m2"].mean())

data["m3"]=data["m3"].replace(np.NaN,data["m3"].mean())

print(data)

**OUTPUT:**

rollno name m1 m2 m3

0 1 bgs 10.0 NaN 30.0

1 2 cgs 20.0 30.0 40.0

2 3 ngs NaN 40.0 50.0

3 4 pgs 25.0 35.0 45.0

4 5 ppp 11.0 22.0 NaN

rollno name m1 m2 m3

0 False False False True False

1 False False False False False

2 False False True False False

3 False False False False False

4 False False False False True

rollno name m1 m2 m3

0 True True True False True

1 True True True True True

2 True True False True True

3 True True True True True

4 True True True True False

rollno name m1 m2 m3

1 2 cgs 20.0 30.0 40.0

3 4 pgs 25.0 35.0 45.0

rollno name m1 m2 m3

0 1 bgs 10.0 31.75 30.00

1 2 cgs 20.0 30.00 40.00

2 3 ngs 16.5 40.00 50.00

3 4 pgs 25.0 35.00 45.00

4 5 ppp 11.0 22.00 41.25

**3. Write a python program the Categorical values in numeric format for a given dataset.**

import pandas as pd

import numpy as np

data=pd.read\_csv("ass3\_data.csv")

print(data)

x=data.iloc[:,0:1].values

print(x)

from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

x1=le.fit\_transform(x)

print(x1)

from sklearn.preprocessing import OneHotEncoder

ohe=OneHotEncoder()

xn=ohe.fit\_transform(x).toarray()

print(xn)

**OUTPUT:**

food availbale price

0 punjabi yes 100

1 chinese yes 200

2 punjabi no 250

3 indian yes 300

[['punjabi']

['chinese']

['punjabi']

['indian']]

[2 0 2 1]

[[0. 0. 1.]

[1. 0. 0.]

[0. 0. 1.]

[0. 1. 0.]]

**4. Write a python program to implement simple Linear Regression for predicting house price.**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear\_model import LinearRegression

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import mean\_absolute\_error

data=pd.read\_csv("house.csv")

print(data)

x=data[["sqft\_living"]]

y=data.price

print(x)

print(y)

plt.scatter(x,y)

plt.xlabel("sqft\_living")

plt.ylabel("price")

plt.show()

xtrain,xtest,ytrain,ytest=train\_test\_split(x,y,test\_size=0.2)

print(xtrain)

print(xtest)

print(ytrain)

print(ytest)

lr=LinearRegression()

lr.fit(xtrain,ytrain)

print(lr.intercept\_)

print(lr.coef\_)

print(lr.predict([[1000]]))

ypred=lr.predict(xtest)

cm=mean\_absolute\_error(ytest,ypred)

print(cm)

**OUTPUT:**

id date price ... long sqft\_living15 sqft\_lot15

0 7129300520 20141013T000000 221900.0 ... -122.257 1340 5650

1 6414100192 20141209T000000 538000.0 ... -122.319 1690 7639

2 5631500400 20150225T000000 180000.0 ... -122.233 2720 8062

3 2487200875 20141209T000000 604000.0 ... -122.393 1360 5000

4 1954400510 20150218T000000 510000.0 ... -122.045 1800 7503

... ... ... ... ... ... ... ...

4995 3583400130 20141014T000000 692500.0 ... -122.256 2290 10700

4996 7230400430 20140930T000000 322400.0 ... -122.100 1990 20359

4997 7140600190 20140905T000000 233500.0 ... -122.214 1400 10658

4998 6817801410 20140624T000000 400000.0 ... -122.034 1570 11517

4999 6430500010 20140620T000000 547000.0 ... -122.350 1300 4080

[5000 rows x 21 columns]

sqft\_living

0 1180

1 2570

2 770

3 1960

4 1680

... ...

4995 3420

4996 1710

4997 1580

4998 1230

4999 2200

[5000 rows x 1 columns]

0 221900.0

1 538000.0

2 180000.0

3 604000.0

4 510000.0

...

4995 692500.0

4996 322400.0

4997 233500.0

4998 400000.0

4999 547000.0

Name: price, Length: 5000, dtype: float64

sqft\_living

1950 1480

480 1700

2811 3915

677 1120

2589 980

... ...

4835 970

2070 3240

59 1850

3309 640

1851 2230

[4000 rows x 1 columns]

sqft\_living

2222 2300

4491 4180

1987 2820

4508 1420

950 1410

... ...

404 1820

1964 1340

477 1050

3559 1270

2788 1620

[1000 rows x 1 columns]

1950 483945.0

480 378500.0

2811 963990.0

677 188000.0

2589 134000.0

...

4835 219000.0

2070 370000.0

59 430000.0

3309 426000.0

1851 500000.0

Name: price, Length: 4000, dtype: float64

2222 453000.0

4491 673200.0

1987 1600000.0

4508 413107.0

950 435000.0

...

404 322500.0

1964 252000.0

477 438924.0

3559 540000.0

2788 545000.0

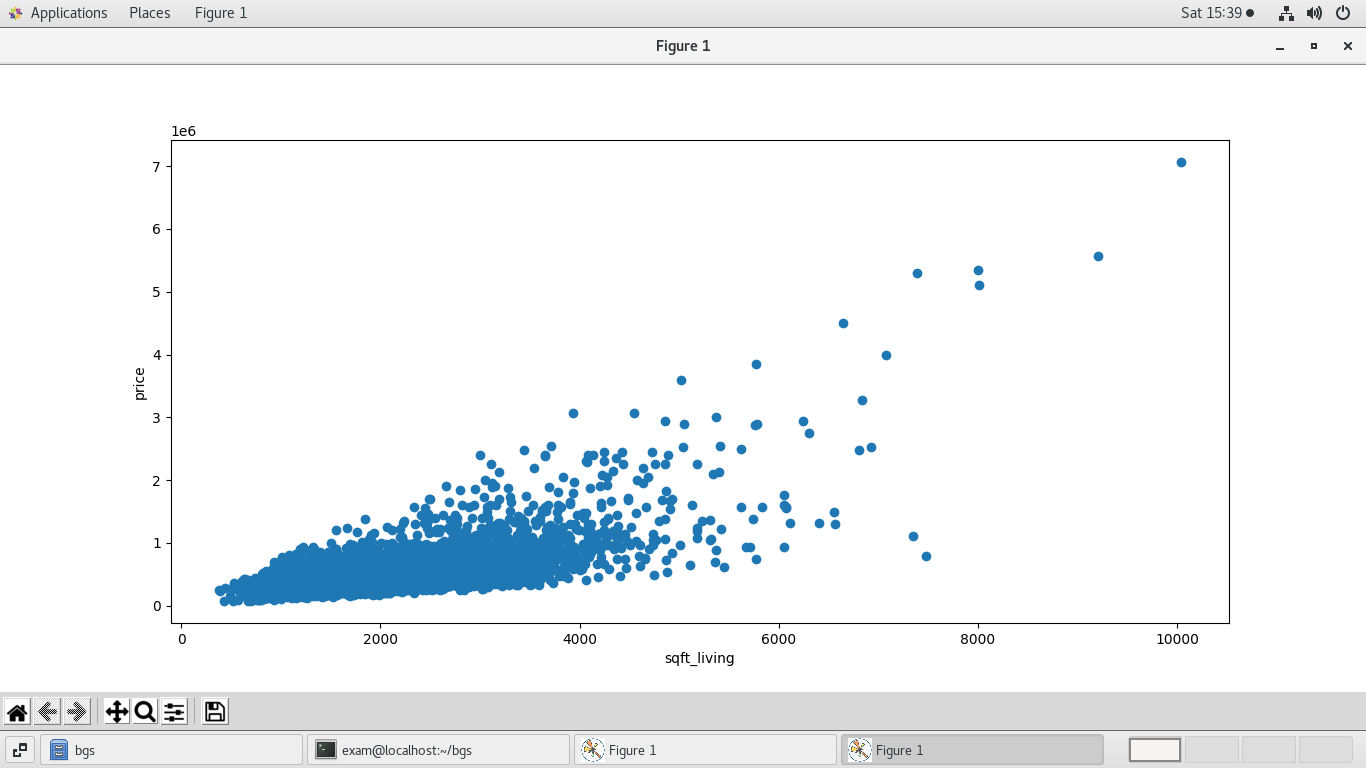
Name: price, Length: 1000, dtype: float64

-44954.11263401585

[283.06052059]

[238106.40795892]

190467.21975274026



**5. Write a python program to implement multiple Linear Regression for a given dataset.**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear\_model import LinearRegression

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import mean\_absolute\_error

data=pd.read\_csv("house.csv")

print(data)

x=data[["bedrooms","sqft\_living"]]

y=data.price

print(x)

print(y)

xtrain,xtest,ytrain,ytest=train\_test\_split(x,y,test\_size=0.2)

print(xtrain)

print(xtest)

print(ytrain)

print(ytest)

lr=LinearRegression()

lr.fit(xtrain,ytrain)

print(lr.intercept\_)

print(lr.coef\_)

print(lr.predict([[2,1000]]))

ypred=lr.predict(xtest)

cm=mean\_absolute\_error(ytest,ypred)

print(cm)

**OUTPUT:**

id date price bedrooms bathrooms sqft\_living ... yr\_renovated zipcode lat long sqft\_living15 sqft\_lot15

0 7129300520 20141013T000000 221900.0 3 1.00 1180 ... 0 98178 47.5112 -122.257 1340 5650

1 6414100192 20141209T000000 538000.0 3 2.25 2570 ... 1991 98125 47.7210 -122.319 1690 7639

2 5631500400 20150225T000000 180000.0 2 1.00 770 ... 0 98028 47.7379 -122.233 2720 8062

3 2487200875 20141209T000000 604000.0 4 3.00 1960 ... 0 98136 47.5208 -122.393 1360 5000

4 1954400510 20150218T000000 510000.0 3 2.00 1680 ... 0 98074 47.6168 -122.045 1800 7503

... ... ... ... ... ... ... ... ... ... ... ... ... ...

4995 3583400130 20141014T000000 692500.0 3 2.25 3420 ... 2004 98028 47.7412 -122.256 2290 10700

4996 7230400430 20140930T000000 322400.0 3 1.75 1710 ... 0 98059 47.4706 -122.100 1990 20359

4997 7140600190 20140905T000000 233500.0 3 1.50 1580 ... 0 98002 47.2903 -122.214 1400 10658

4998 6817801410 20140624T000000 400000.0 3 2.00 1230 ... 0 98074 47.6321 -122.034 1570 11517

4999 6430500010 20140620T000000 547000.0 5 2.50 2200 ... 0 98103 47.6872 -122.350 1300 4080

[5000 rows x 21 columns]

bedrooms sqft\_living

0 3 1180

1 3 2570

2 2 770

3 4 1960

4 3 1680

... ... ...

4995 3 3420

4996 3 1710

4997 3 1580

4998 3 1230

4999 5 2200

[5000 rows x 2 columns]

0 221900.0

1 538000.0

2 180000.0

3 604000.0

4 510000.0

...

4995 692500.0

4996 322400.0

4997 233500.0

4998 400000.0

4999 547000.0

Name: price, Length: 5000, dtype: float64

bedrooms sqft\_living

1583 4 2590

1267 3 1240

1816 4 1780

3375 2 1060

3844 3 1540

... ... ...

4390 1 820

4309 2 2330

3286 3 1330

2109 3 1790

3853 4 2820

[4000 rows x 2 columns]

bedrooms sqft\_living

936 4 2790

2331 3 770

2759 3 1350

1536 5 2980

3757 3 2260

... ... ...

3975 2 990

179 2 1350

1608 3 2110

1523 3 970

2446 5 3880

[1000 rows x 2 columns]

1583 1175000.0

1267 340000.0

1816 587500.0

3375 356000.0

3844 216650.0

...

4390 527550.0

4309 535000.0

3286 180000.0

2109 307000.0

3853 1075000.0

Name: price, Length: 4000, dtype: float64

936 378000.0

2331 307000.0

2759 172000.0

1536 932800.0

3757 625000.0

...

3975 210000.0

179 330000.0

1608 285000.0

1523 170000.0

2446 1126000.0

Name: price, Length: 1000, dtype: float64

85056.52244579763

[-69169.7759704 333.51889927]

[280235.86977759]

173209.9118546618

**6. Write a python program to implement Polynomial Regression for given dataset.**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear\_model import LinearRegression

data=pd.read\_csv("ass6\_data.csv")

print(data)

x=data.iloc[:,1:2].values

y=data.iloc[:,2].values

print(x)

print(y)

from sklearn.model\_selection import train\_test\_split

xtrain,xtest,ytrain,ytest=train\_test\_split(x,y,test\_size=0.25)

lr=LinearRegression()

lr.fit(xtrain,ytrain)

ypred=lr.predict(xtest)

plt.scatter(x,y,c="red")

plt.plot(x,lr.predict(x),c="green")

plt.show()

from sklearn.preprocessing import PolynomialFeatures

pr=PolynomialFeatures(degree=4)

xpoly=pr.fit\_transform(x)

poreg=LinearRegression()

poreg.fit(xpoly,y)

plt.scatter(x,y,c="red")

plt.plot(x,poreg.predict(pr.fit\_transform(x)),c="green")

plt.show()

print(lr.predict([[5.5]]))

print(poreg.predict(pr.fit\_transform([[5.5]])))

**OUTPUT:**

designation level salary

0 peon 1 20000

1 jr\_clerk 2 25000

2 sr\_clerk 3 35000

3 accountant 4 55000

4 os 5 85000

5 ass\_pro 6 100000

6 asso\_prof 7 140000

7 prof 8 250000

[[1]

[2]

[3]

[4]

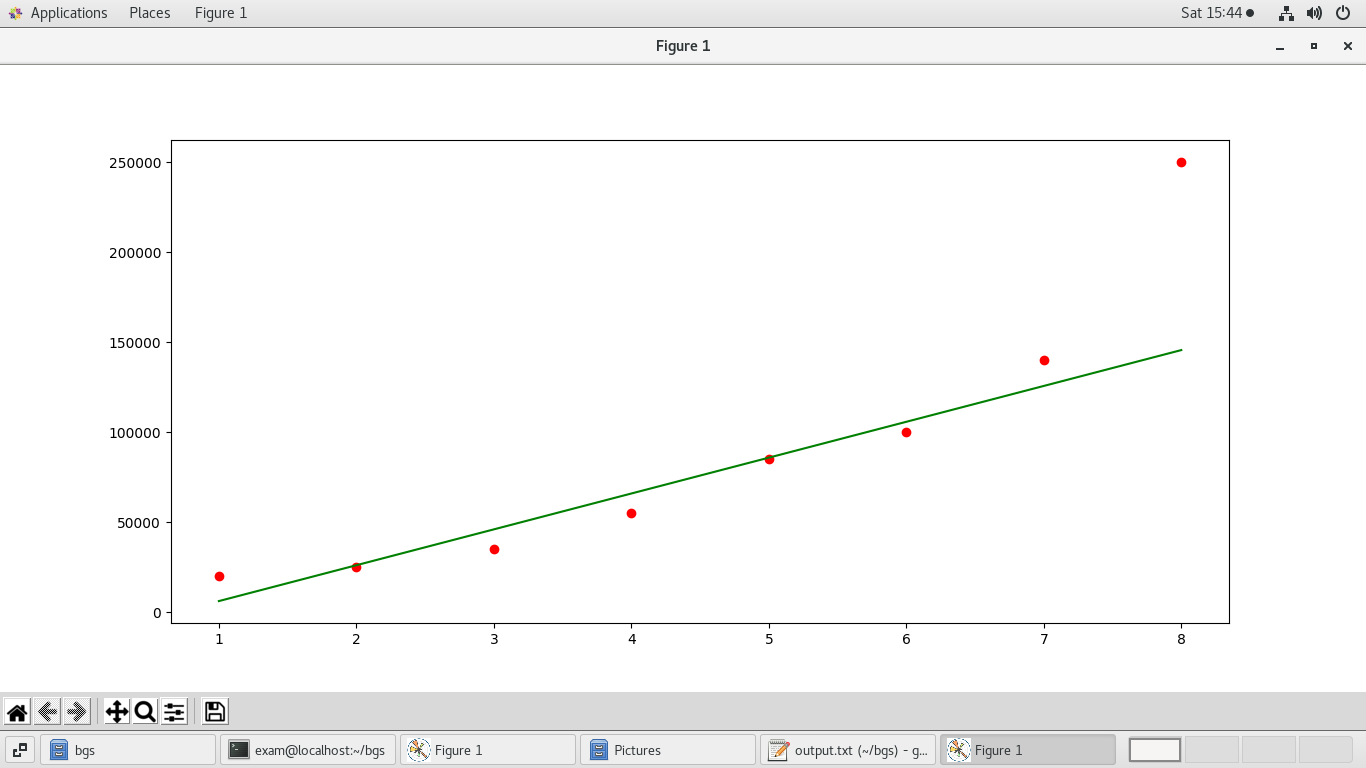
[5]

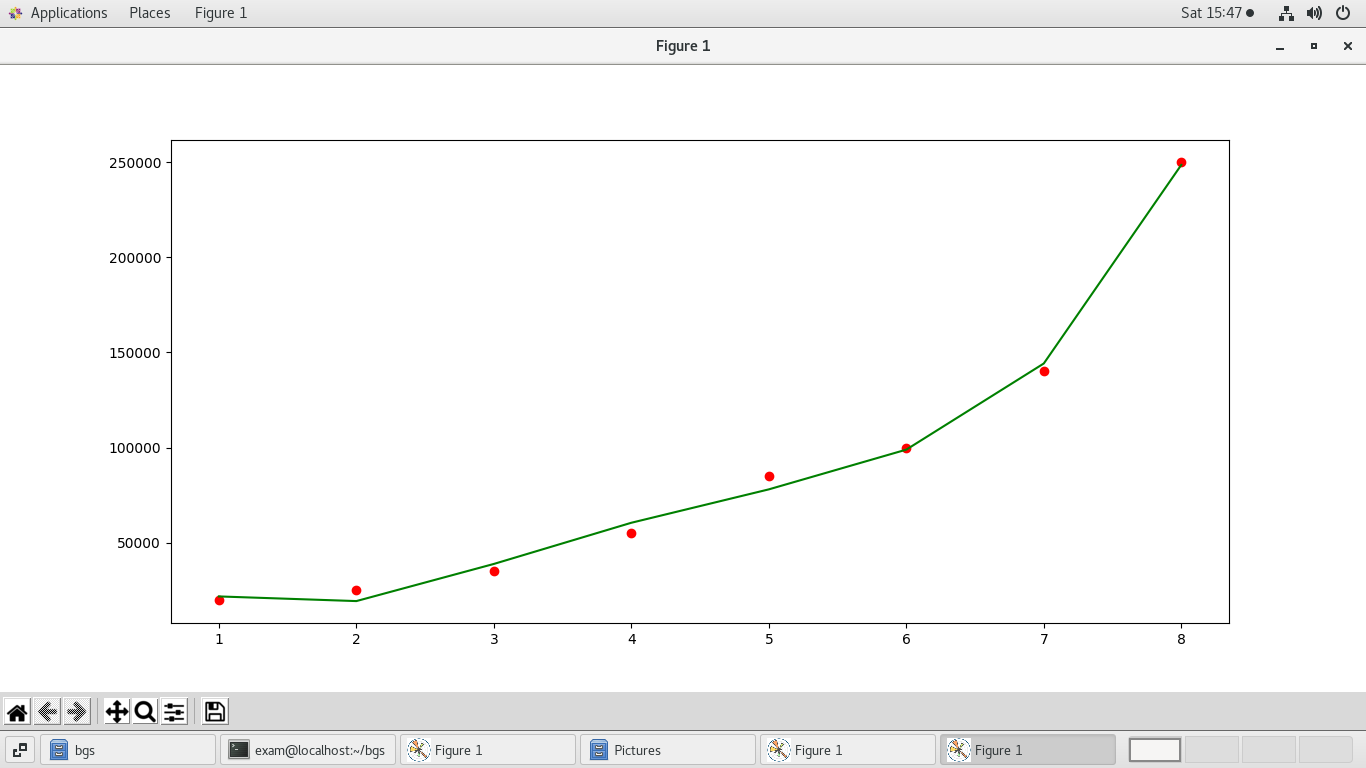
[6]

[7]

[8]]

[ 20000 25000 35000 55000 85000 100000 140000 250000]





**7. Write a python program to Implement Naïve Bayes.**

import numpy as np

import pandas as pd

data=pd.read\_csv("user\_data1.csv")

print(data)

x=data.iloc[:,[2,4]].values

y=data.iloc[:,4].values

from sklearn.model\_selection import train\_test\_split

xtrain,xtest,ytrain,ytest=train\_test\_split(x,y,test\_size=0.05)

from sklearn.preprocessing import StandardScaler

sc=StandardScaler()

xtrain=sc.fit\_transform(xtrain)

xtest=sc.fit\_transform(xtest)

from sklearn.naive\_bayes import GaussianNB

gb=GaussianNB()

gb.fit(xtrain,ytrain)

ypred=gb.predict(xtest)

print(ytest)

print(ypred)

from sklearn.metrics import confusion\_matrix

cm=confusion\_matrix(ytest,ypred)

print(cm)

**OUTPUT:**

0 15624510 Male 19 19000 0

1 15810944 Male 35 20000 0

2 15668575 Female 26 43000 0

3 15603246 Female 27 57000 0

4 15804002 Male 19 76000 0

.. ... ... ... ... ...

395 15691863 Female 46 41000 1

396 15706071 Male 51 23000 1

397 15654296 Female 50 20000 1

398 15755018 Male 36 33000 0

399 15594041 Female 49 36000 1

[400 rows x 5 columns]

[0 0 0 1 0 1 0 1 0 0 0 0 0 1 0 1 1 1 1 0]

[0 0 0 1 0 1 0 1 0 0 0 0 0 1 0 1 1 1 1 0]

[[12 0]

[ 0 8]]

**8.Write a python program to Implement Decision Tree whether or not to play tennis.**

import pandas as pd

import numpy as np

from sklearn.tree import DecisionTreeClassifier

from sklearn.preprocessing import LabelEncoder

data=pd.read\_csv("tennis.csv")

print(data)

le=LabelEncoder()

data["Outlook"]=le.fit\_transform(data["Outlook"])

data["Temprature"]=le.fit\_transform(data["Temprature"])

data["Humidity"]=le.fit\_transform(data["Humidity"])

data["Wind"]=le.fit\_transform(data["Wind"])

data["Play\_Tennis"]=le.fit\_transform(data["Play\_Tennis"])

print(data)

x=data.iloc[:,1:5].values

y=data["Play\_Tennis"]

print(x)

print(y)

from sklearn.model\_selection import train\_test\_split

xtrain,xtest,ytrain,ytest=train\_test\_split(x,y,test\_size=0.4)

dc=DecisionTreeClassifier(criterion="entropy")

dc.fit(xtrain,ytrain)

ypred=dc.predict(xtest)

from sklearn.metrics import confusion\_matrix

cm=confusion\_matrix(ytest,ypred)

print(cm)

from sklearn.tree import export\_graphviz

export\_graphviz(dc,out\_file="abc.dat")

from sklearn.metrics import accuracy\_score

print("accuracy:",accuracy\_score(ytest,ypred))

**OUTPUT:**

Day Outlook Temprature Humidity Wind Play\_Tennis

0 D1 Sunny Hot High Weak No

1 D2 Sunny Hot High Strong No

2 D3 Overcast Hot High Weak Yes

3 D4 Rain Mild High Weak Yes

4 D5 Rain Cool Normal Weak Yes

5 D6 Rain Cool Normal Strong No

6 D7 Overcast Cool Normal Strong Yes

7 D8 Sunny Mild High Weak No

8 D9 Sunny Cool Normal Weak Yes

9 D10 Rain Mild Normal Weak Yes

10 D11 Sunny Mild Normal Strong Yes

11 D12 Overcast Mild High Strong Yes

12 D13 Overcast Hot Normal Weak Yes

13 D14 Rain Mild High Strong No

Day Outlook Temprature Humidity Wind Play\_Tennis

0 D1 2 1 0 1 0

1 D2 2 1 0 0 0

2 D3 0 1 0 1 1

3 D4 1 2 0 1 1

4 D5 1 0 1 1 1

5 D6 1 0 1 0 0

6 D7 0 0 1 0 1

7 D8 2 2 0 1 0

8 D9 2 0 1 1 1

9 D10 1 2 1 1 1

10 D11 2 2 1 0 1

11 D12 0 2 0 0 1

12 D13 0 1 1 1 1

13 D14 1 2 0 0 0

[[2 1 0 1]

[2 1 0 0]

[0 1 0 1]

[1 2 0 1]

[1 0 1 1]

[1 0 1 0]

[0 0 1 0]

[2 2 0 1]

[2 0 1 1]

[1 2 1 1]

[2 2 1 0]

[0 2 0 0]

[0 1 1 1]

[1 2 0 0]]

0 0

1 0

2 1

3 1

4 1

5 0

6 1

7 0

8 1

9 1

10 1

11 1

12 1

13 0

Name: Play\_Tennis, dtype: int64

[[0 1]

[2 3]]

accuracy: 0.5

**9. Write a python program to implement linear SVM.**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

data=pd.read\_csv("user\_data1.csv")

x=data.iloc[:,2:3].values

y=data.iloc[:,3].values

print(x)

print(y)

from sklearn.preprocessing import StandardScaler

sc\_x=StandardScaler()

x=sc\_x.fit\_transform(x)

from sklearn.svm import SVR

reg=SVR()

reg.fit(x,y)

y\_pred=reg.predict(np.array([[35]]))

print(y\_pred)

x\_grid=np.arange(min(x),max(x),0.01)

x\_grid=x\_grid.reshape(len(x\_grid),1)

plt.scatter(x,y,c="red")

plt.plot(x\_grid,reg.predict(x\_grid),c="green")

plt.show()

**OUTPUT:**

[19] [26] [45] [31] [35] [26] [47] [27] [26] [20] [48] [21] [27] [32] [45] [28] [19] [18] [46] [27] [29] [47] [27] [47] [49] [32] [45] [47] [25] [46] [29] [35] [48] [31] [27] [35] [33] [30] [26] [27] [27] [33] [35] [30] [28] [23] [25] [27] [30] [31] [24] [18] [29] [35] [27] [24] [23] [28] [22] [32] [27] [25] [23] [32] [59] [24] [24] [23] [22] [31] [25] [24] [20] [33] [32] [34] [18] [22] [28] [26] [30] [39] [20] [35] [30] [31] [24] [28] [26] [35] [22] [30] [26] [29] [29] [35] [35] [28] [35] [28] [27] [28] [32] [33] [19] [21] [26] [27] [26] [38] [39] [37] [38] [37] [42] [40] [35] [36] [40] [41] [36] [37] [40] [35] [41] [39] [42] [26] [30] [26] [31] [33] [30] [21] [28] [23] [20] [30] [28] [19] [19] [18] [35] [30] [34] [24] [27] [41] [29] [20] [26] [41] [31] [36] [40] [31] [46] [29] [26] [32] [32] [25] [37] [35] [33] [18] [22] [35] [29] [29] [21] [34] [26] [34] [34] [23] [35] [25] [24] [31] [26] [31] [32] [33] [33]][31] [20] [33] [35] [28] [24] [19] [29] [19] [28] [34] [30] [20] [26] [35] [35] [49] [39] [41] [58] [47] [55] [52] [40] [46] [48] [52] [59] [35] [47] [60] [49] [40] [46] [59] [41] [35] [37] [60] [35] [37] [36] [56] [40] [42] [35] [39] [40] [49] [38] [46] [40] [37] [46] [53] [42] [38] [50] [56] [41] [51] [35] [57] [41] [35] [44] [37] [48] [37] [50] [52] [41] [40] [58] [45] [35] [36] [55] [35] [48] [42] [40] [37] [47] [40] [43] [59] [60] [39] [57] [57] [38] [49] [52] [50] [59] [35] [37] [52] [48] [37] [37] [48][41] [37] [39] [49] [55] [37] [35] [36] [42] [43] [45] [46] [58] [48] [37] [37] [40] [42] [51] [47] [36] [38] [42] [39] [38] [49] [39] [39] [54] [35] [45] [36] [52] [53] [41] [48] [48] [41] [41] [42] [36] [47] [38] [48] [42] [40] [57] [36] [58] [35] [38] [39] [53] [35] [38] [47] [47] [41] [53] [54] [39] [38] [38] [37] [42] [37] [36] [60] [54] [41] [40] [42] [43] [53] [47] [42] [42] [59] [58] [46] [38] [54] [60] [60] [39] [59] [37] [46] [46] [42] [41] [58] [42] [48] [44] [49] [57] [56] [49] [39] [47] [48] [48] [47] [45] [60] [39] [46] [51] [50] [36] [49]]

[ 19000 20000 43000 57000 76000 58000 84000 150000 33000 65000

80000 52000 86000 18000 82000 80000 25000 26000 28000 29000

22000 49000 41000 22000 23000 20000 28000 30000 43000 18000

74000 137000 16000 44000 90000 27000 28000 49000 72000 31000

17000 51000 108000 15000 84000 20000 79000 54000 135000 89000

32000 44000 83000 23000 58000 55000 48000 79000 18000 117000

20000 87000 66000 120000 83000 58000 19000 82000 63000 68000

80000 27000 23000 113000 18000 112000 52000 27000 87000 17000

80000 42000 49000 88000 62000 118000 55000 85000 81000 50000

81000 116000 15000 28000 83000 44000 25000 123000 73000 37000

88000 59000 86000 149000 21000 72000 35000 89000 86000 80000

71000 71000 61000 55000 80000 57000 75000 52000 59000 59000

75000 72000 75000 53000 51000 61000 65000 32000 17000 84000

58000 31000 87000 68000 55000 63000 82000 107000 59000 25000

85000 68000 59000 89000 25000 89000 96000 30000 61000 74000

15000 45000 76000 50000 47000 15000 59000 75000 30000 135000

100000 90000 33000 38000 69000 86000 55000 71000 148000 47000

88000 115000 118000 43000 72000 28000 47000 22000 23000 34000

16000 71000 117000 43000 60000 66000 82000 41000 72000 32000

84000 26000 43000 70000 89000 43000 79000 36000 80000 22000

39000 74000 134000 71000 101000 47000 130000 114000 142000 22000

96000 150000 42000 58000 43000 108000 65000 78000 96000 143000

80000 91000 144000 102000 60000 53000 126000 133000 72000 80000

147000 42000 107000 86000 112000 79000 57000 80000 82000 143000

149000 59000 88000 104000 72000 146000 50000 122000 52000 97000

39000 52000 134000 146000 44000 90000 72000 57000 95000 131000

77000 144000 125000 72000 90000 108000 75000 74000 144000 61000

133000 76000 42000 106000 26000 74000 71000 88000 38000 36000

88000 61000 70000 21000 141000 93000 62000 138000 79000 78000

134000 89000 39000 77000 57000 63000 73000 112000 79000 117000

38000 74000 137000 79000 60000 54000 134000 113000 125000 50000

70000 96000 50000 141000 79000 75000 104000 55000 32000 60000

138000 82000 52000 30000 131000 60000 72000 75000 118000 107000

51000 119000 65000 65000 60000 54000 144000 79000 55000 122000

104000 75000 65000 51000 105000 63000 72000 108000 77000 61000

113000 75000 90000 57000 99000 34000 70000 72000 71000 54000

129000 34000 50000 79000 104000 29000 47000 88000 71000 26000

46000 83000 73000 130000 80000 32000 74000 53000 87000 23000

64000 33000 139000 28000 33000 60000 39000 71000 34000 35000

33000 23000 45000 42000 59000 41000 23000 20000 33000 36000]

[69993.44915575]

**10. Write a python program to implement k-nearest Neighbors ML algorithm to build prediction model (Use Forge Dataset)**

import pandas as pd

import numpy as nm

import matplotlib.pyplot as mtp

data=pd.read\_csv("user\_data1.csv")

print(data)

x=data.iloc[:,2:4].values

y=data.iloc[:,4].values

from sklearn.model\_selection import train\_test\_split

xtrain,xtest,ytrain,ytest=train\_test\_split(x,y,test\_size=0.25)

from sklearn.preprocessing import StandardScaler

sc=StandardScaler()

xtrain=sc.fit\_transform(xtrain)

xtest=sc.transform(xtest)

from sklearn.neighbors import KNeighborsClassifier

reg1=KNeighborsClassifier(n\_neighbors=5)

reg1.fit(xtrain,ytrain)

ypred=reg1.predict(xtest)

from sklearn.metrics import confusion\_matrix

cs=confusion\_matrix(ytest,ypred)

print(cs)

from matplotlib.colors import ListedColormap

x\_set, y\_set = xtest, ytest

x1, x2 = nm.meshgrid(nm.arange(start = x\_set[:, 0].min() - 1, stop = x\_set[:, 0].max() + 1, step =0.01),

nm.arange(start = x\_set[:, 1].min() - 1, stop = x\_set[:, 1].max() + 1, step = 0.01))

mtp.contourf(x1, x2, reg1.predict(nm.array([x1.ravel(), x2.ravel()]).T).reshape(x1.shape),

alpha = 0.75, cmap = ListedColormap(('purple','green' )))

mtp.xlim(x1.min(), x1.max())

mtp.ylim(x2.min(), x2.max())

for i, j in enumerate(nm.unique(y\_set)):

mtp.scatter(x\_set[y\_set == j, 0], x\_set[y\_set == j, 1],c =ListedColormap(('purple', 'green'))(i), label = j)

mtp.title('Decision Tree Algorithm (Training set)')

mtp.xlabel('Age')

mtp.ylabel('Estimated Salary')

mtp.legend()

mtp.show()

**OUTPUT:**

User ID Gender Age EstimatedSalary Purchased

0 15624510 Male 19 19000 0

1 15810944 Male 35 20000 0

2 15668575 Female 26 43000 0

3 15603246 Female 27 57000 0

4 15804002 Male 19 76000 0

.. ... ... ... ... ...

395 15691863 Female 46 41000 1

396 15706071 Male 51 23000 1

397 15654296 Female 50 20000 1

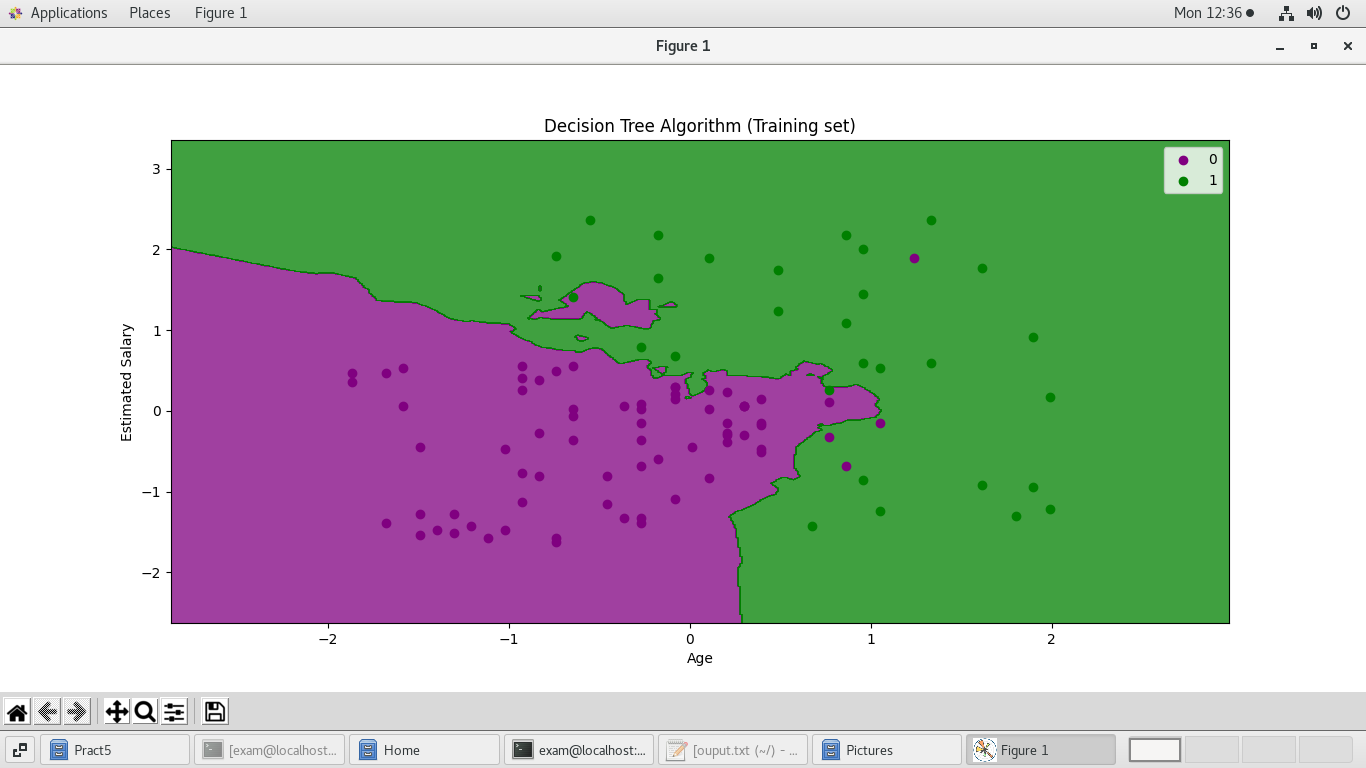
398 15755018 Male 36 33000 0

399 15594041 Female 49 36000 1

[400 rows x 5 columns]

[[67 4]

[ 2 27]]



**11. Write a python program to implement k-means algorithm on a synthetic dataset.**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.cluster import KMeans

data=pd.read\_csv("Mall\_data.csv")

print(data)

x=data.iloc[:,[3,4]].values

print(x)

wcss=[]

for i in range(1,11):

km=KMeans(n\_clusters=i)

km.fit(x)

wcss.append(km.inertia\_)

plt.plot(range(1,11),wcss)

plt.show()

km=KMeans(n\_clusters=5)

y\_pred=km.fit\_predict(x)

plt.scatter(x[y\_pred==0,0],x[y\_pred==0,1],s=100,c="blue",label="cluster1")

plt.scatter(x[y\_pred==1,0],x[y\_pred==1,1],s=100,c="red",label="cluster2")

plt.scatter(x[y\_pred==2,0],x[y\_pred==2,1],s=100,c="green",label="cluster3")

plt.scatter(x[y\_pred==3,0],x[y\_pred==3,1],s=100,c="pink",label="cluster4")

plt.scatter(x[y\_pred==4,0],x[y\_pred==4,1],s=100,c="black",label="cluster5")

plt.scatter(km.cluster\_centers\_[:,0],km.cluster\_centers\_[:,1],s=300,c="yellow",label="centroid")

plt.title("kmenas")

plt.xlabel("Salary")

plt.ylabel("")

plt.show()

**OUTPUT:**

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

.. ... ... ... ... ...

195 196 Female 35 120 79

196 197 Female 45 126 28

197 198 Male 32 126 74

198 199 Male 32 137 18

199 200 Male 30 137 83

[200 rows x 5 columns]

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

.. ... ... ... ... ...

195 196 Female 35 120 79

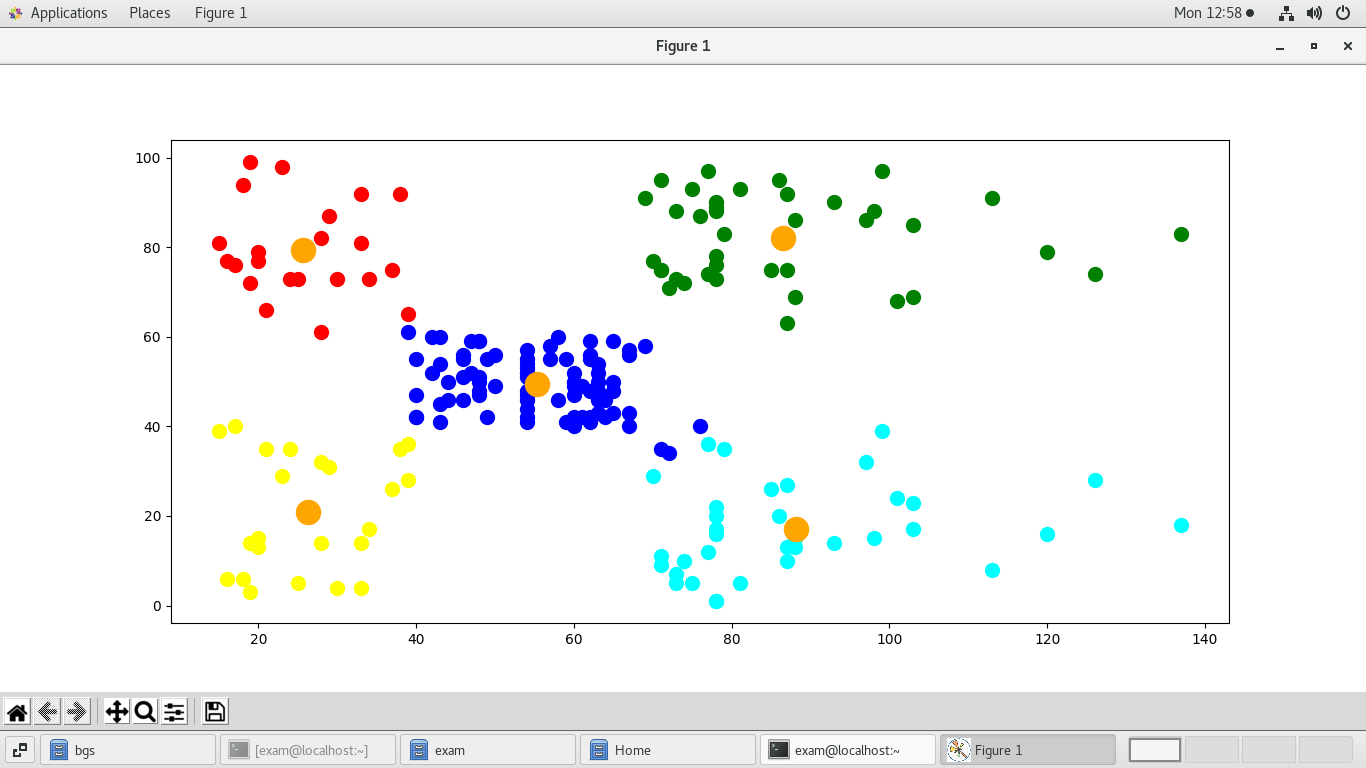
196 197 Female 45 126 28

197 198 Male 32 126 74

198 199 Male 32 137 18

199 200 Male 30 137 83

[200 rows x 5 columns]



**12. Write a python program to implement Agglomerative clustering on a synthetic dataset.**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import scipy.cluster.hierarchy as shc

data=pd.read\_csv("Mall\_data.csv")

print(data)

x=data.iloc[:,[3,4]].values

den=shc.dendrogram(shc.linkage(x,method="ward"))

plt.title("dendogram")

plt.xlabel("cluster")

plt.ylabel("ecludian distance")

plt.show();

from sklearn.cluster import AgglomerativeClustering

ag=AgglomerativeClustering(n\_clusters=3)

y\_pred=ag.fit\_predict(x)

plt.scatter(x[y\_pred==0,0],x[y\_pred==0,1],s=100,c="red",label="Cluster1")

plt.scatter(x[y\_pred==1,0],x[y\_pred==1,1],s=100,c="yellow",label="Cluster2")

plt.scatter(x[y\_pred==2,0],x[y\_pred==2,1],s=100,c="green",label="Cluster3")

plt.show()

**OUTPUT:**

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

.. ... ... ... ... ...

195 196 Female 35 120 79

196 197 Female 45 126 28

197 198 Male 32 126 74

198 199 Male 32 137 18

199 200 Male 30 137 83

[200 rows x 5 columns]