# Program 8: Implement simple linear regression

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

data\_set= pd.read\_csv(r"C:\Users\BASAPPA SIR\Desktop\Dataset\Salary\_Data.csv")

x= data\_set.iloc[:, :-1].values

y= data\_set.iloc[:, 1].values

print(x)

print(y)

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

x\_train, x\_test, y\_train, y\_test= train\_test\_split(x, y, test\_size= 1/3, random\_state=0)

from sklearn.linear\_model import LinearRegression

regressor= LinearRegression()

regressor.fit(x\_train, y\_train)

y\_pred= regressor.predict(x\_test)

x\_pred= regressor.predict(x\_train)

x\_pred

plt.scatter(x\_train, y\_train, color="green")

plt.plot(x\_train, x\_pred, color="red")

plt.title("Salary vs Experience (Training Dataset")

plt.xlabel("Years of Experience")

plt.ylabel("Salary(In Rupees)")

plt.show()

plt.scatter(x\_test, y\_test, color="blue")

plt.plot(x\_train, x\_pred, color="red")

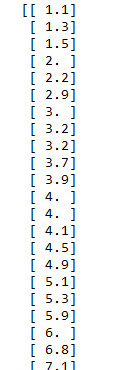
plt.title("Salary vs Experience (Test Dataset)")

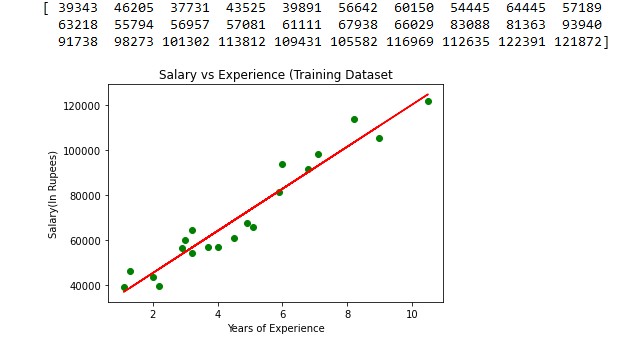
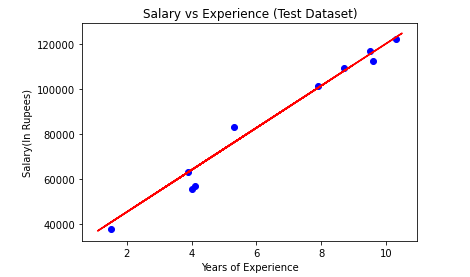
plt.xlabel("Years of Experience")

plt.ylabel("Salary(In Rupees)")

plt.show()

**output:**

****

** **

**Program 9: Implement multiple regression**

**#Multiple Regression**

import numpy as np

import matplotlib.pyplot as lpt

import pandas as pd

data\_set= pd.read\_csv("C:\\Users\BASAPPA SIR\\Desktop\\Dataset\\50\_Startups.csv")

x= data\_set.iloc[:, :-1].values

y= data\_set.iloc[:,1].values

from sklearn.preprocessing import LabelEncoder, OneHotEncoder

labelencoder\_x= LabelEncoder()

x[:,2]= labelencoder\_x.fit\_transform(x[:,2])

onehotencoder= OneHotEncoder()

x= onehotencoder.fit\_transform(x).toarray()

print(x)

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

x\_train, x\_test, y\_train, y\_test= train\_test\_split(x, y, test\_size= 0.2, random\_state=0)

from sklearn.linear\_model import LinearRegression

regressor= LinearRegression()

regressor.fit(x\_train, y\_train)

#Predicting the Test set result;

y\_pred= regressor.predict(x\_test)

print('Train Score: ', regressor.score(x\_train, y\_train))

print('Test Score: ', regressor.score(x\_test, y\_test))

fig = plt.figure(figsize=(10, 6))

plt.scatter(y\_test, y\_pred)

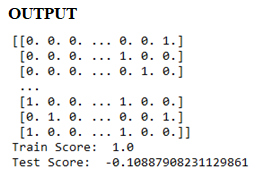
plt.xlabel('Position Levels')

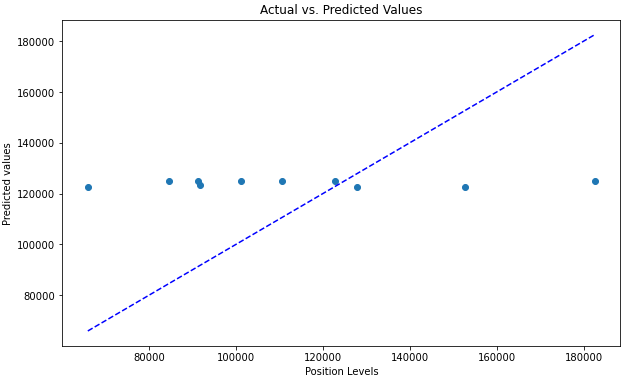
plt.ylabel('Predicted values')

plt.title('Actual vs. Predicted Values')

plt.plot([min(y\_test), max(y\_test)], [min(y\_test), max(y\_test)], linestyle='--', color='blue')

plt.show()

****

****

# Program 10: Implement polynomial regression

**#polynominal program**

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

data\_set= pd.read\_csv("C:\\Users\BASAPPA SIR\\Desktop\\Dataset\\Salary\_Data.csv")

x= data\_set.iloc[:, :-1].values

y= data\_set.iloc[:, 1].values

from sklearn.linear\_model import LinearRegression

lin\_regs= LinearRegression()

lin\_regs.fit(x,y)

from sklearn.preprocessing import PolynomialFeatures

poly\_regs= PolynomialFeatures(degree= 2)

x\_poly= poly\_regs.fit\_transform(x)

lin\_reg\_2 =LinearRegression()

lin\_reg\_2.fit(x\_poly, y)

plt.scatter(x,y,color="blue")

plt.plot(x,lin\_regs.predict(x), color="red")

plt.title("Bluff detection model(Linear Regression)")

plt.xlabel("Position Levels")

plt.ylabel("Salary")

plt.show()

plt.scatter(x,y,color="blue")

plt.plot(x, lin\_reg\_2.predict(poly\_regs.fit\_transform(x)), color="red")

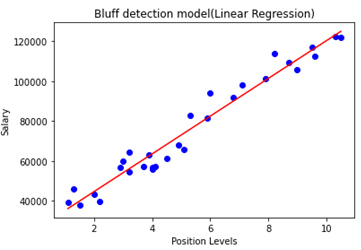
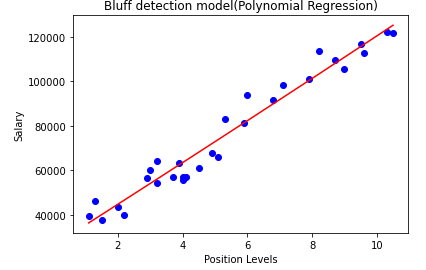
plt.title("Bluff detection model(Polynomial Regression)")

plt.xlabel("Position Levels")

plt.ylabel("Salary")

plt.show()

**OUTPUT**

** **

# Program 11: Implement Decision Tree

# Decision Tree Program

# importing libraries

import numpy as np

import matplotlib.pyplot as lpt

import pandas as pd

#importing datasets

data\_set= pd.read\_csv(r'C:\Users\BASAPPA SIR\Desktop\Dataset\user\_data.csv')

#Extracting Independent and dependent Variable

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

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

print(y)

# Splitting the dataset into training and test set.

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

x\_train, x\_test, y\_train, y\_test= train\_test\_split(x, y, test\_size= 0.25, random\_state=0)

#feature Scaling

from sklearn.preprocessing import StandardScaler

st\_x= StandardScaler()

x\_train= st\_x.fit\_transform(x\_train)

x\_test= st\_x.transform(x\_test)

#Fitting Decision Tree classifier to the training set

from sklearn.tree import DecisionTreeClassifier

classifier= DecisionTreeClassifier(criterion='entropy', random\_state=0)

classifier.fit(x\_train, y\_train)

#Predicting the test set result

y\_pred= classifier.predict(x\_test)

#Creating the Confusion matrix

from sklearn.metrics import confusion\_matrix

cm= confusion\_matrix(y\_test, y\_pred)

print(cm)

#Visulaizing the trianing set result

from matplotlib.colors import ListedColormap

x\_set, y\_set = x\_train, y\_train

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))

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

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

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

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

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

lpt.scatter(x\_set[y\_set == j, 0], x\_set[y\_set == j, 1],

c = ListedColormap(('purple', 'green'))(i), label = j)

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

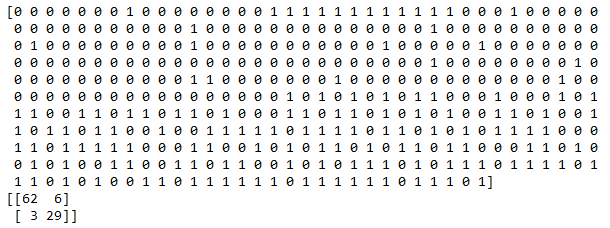
lpt.xlabel('Age')

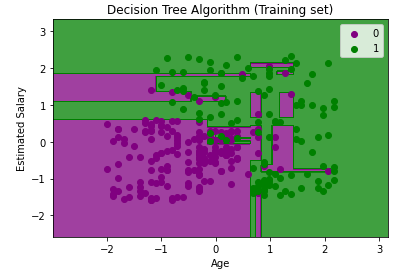
lpt.ylabel('Estimated Salary' )

lpt.legend()

lpt.show()

**OUTPUT:**

****

****

# Program 12: Implement Random Forest

**# Random Forest**

**# importing libraries**

import numpy as np

import matplotlib.pyplot as lpt

import pandas as pd

**#importing datasets**

data\_set= pd.read\_csv(r'C:\Users\BASAPPA SIR\Desktop\Dataset\user\_data.csv')

**#Extracting Independent and dependent Variable**

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

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

**# Splitting the dataset into training and test set.**

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

x\_train, x\_test, y\_train, y\_test= train\_test\_split(x, y, test\_size= 0.25, random\_state=0)

**#feature Scaling**

from sklearn.preprocessing import StandardScaler

st\_x= StandardScaler()

x\_train= st\_x.fit\_transform(x\_train)

x\_test= st\_x.transform(x\_test)

**#Fitting Decision Tree classifier to the training set**

from sklearn.ensemble import RandomForestClassifier

classifier= RandomForestClassifier(n\_estimators= 10, criterion="entropy")

classifier.fit(x\_train, y\_train)

**#Predicting the test set result**

y\_pred= classifier.predict(x\_test)

**#Creating the Confusion matrix**

from sklearn.metrics import confusion\_matrix

cm= confusion\_matrix(y\_test, y\_pred)

print(cm)

**#Visualizing the training Set resultfrom matplotlib.colors import ListedColormap**

from matplotlib.colors import ListedColormap

x\_set, y\_set = x\_train, y\_train

x1, x2 = np.meshgrid(np.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))

lpt.contourf(x1, x2, classifier.predict(np.array([x1.ravel(), x2.ravel()]).T).reshape(x1.shape),

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

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

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

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

lpt.scatter(x\_set[y\_set == j, 0], x\_set[y\_set == j, 1],

c = ListedColormap(('purple', 'green'))(i), label = j)

lpt.title('Random Forest Algorithm (Training set)')

lpt.xlabel('Age')

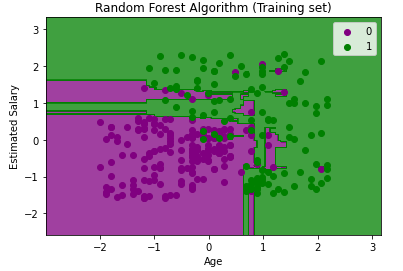
lpt.ylabel('Estimated Salary')

lpt.legend()

lpt.show()

**OUTPUT**

****

****

**Program 13: DATA VISUALIZATION USING PYTHON**

**BAR CHART**

import matplotlib.pyplot as plt

import pandas as pd

x = ['Grocery','Rent','Education of Children','Medicine','Fuel', 'Entertainment',' Miscellaneous']

y = [4,5,5,2,2,1,1]

plt.bar(x, y, color='green',edgecolor='blue',linewidth=2)

plt.title("Family Monthly plan")

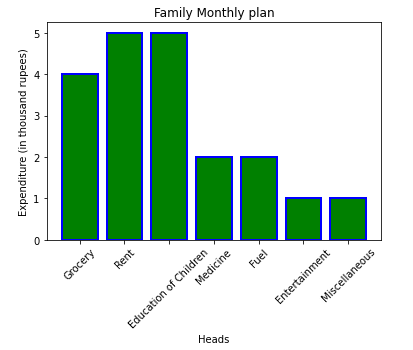
plt.ylabel('Expenditure (in thousand rupees)')

plt.xlabel('Heads')

plt.xticks(x, rotation=45)

plt.show()

**OUTPUT**

****

**HISTOGRAM**

import matplotlib.pyplot as plt

x=[22, 87, 5, 43, 56, 73, 55, 54, 11, 20, 51, 5, 79, 31, 27]

plt.hist(x, bins=[0,20,40,60,80,100], color='green', edgecolor='red', linestyle='-')

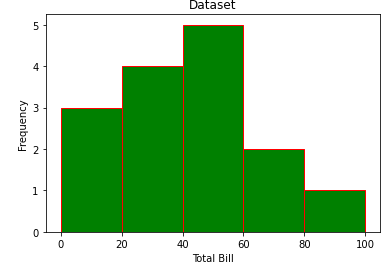
plt.title("Dataset")

plt.ylabel('Frequency')

plt.xlabel('Total Bill')

plt.show()

**OUTPUT**

****

**HISTOGRAM**

import matplotlib.pyplot as plt

cars=['AUDI','BMW','FORD','TESLA','JAGUAR']

data=[23,13,35,15,12]

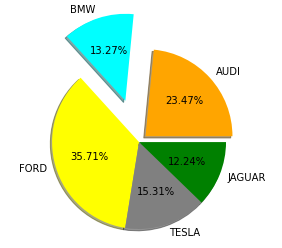
explode=[0.1,0.5,0,0,0]

colors=("orange","cyan","yellow","grey","green",)

plt.pie(data,labels=cars,explode=explode,autopct='%1.2f%%',colors=colors,shadow=True)

plt.show()

**OUTPUT**

****

**SCATTER PLOT:**

import matplotlib.pyplot as plt

price = [2.50, 1.23, 4.02, 3.25, 5.00, 4.40]

sales = [34, 62, 49, 22, 13, 19]

plt.scatter(price, sales, c='RED', s=60,marker='D', alpha=1)

plt.title("Dataset")

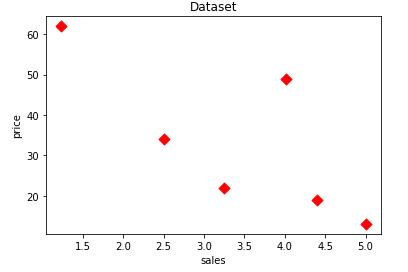
plt.ylabel('price')

plt.xlabel('sales')

plt.show()

plt.show()

**OUTPUT**

****

**Line Graph:**

import matplotlib.pyplot as plt

x = [10, 20, 30, 40]

y = [20, 25, 35, 55]

plt.title=('LINE CHART')

fig = plt.figure(figsize =(7, 5), facecolor='g',edgecolor='b', linewidth=7)

ax = fig.add\_axes([1, 1, 1, 1])

ax.plot(x, y)

plt.xticks(x, labels=["one", "two", "three", "four"])

plt.plot(x, y, color='green', linewidth=3, marker='o',markersize=15, linestyle='--')

plt.legend(["GFG"])

fig = plt.figure(figsize =(5, 4))

ax1 = fig.add\_axes([0.1, 0.1, 0.8, 0.8])

ax2 = fig.add\_axes([1, 0.1, 0.8, 0.8])

ax1.plot(x, y)

ax2.plot(y, x)

plt.ylabel('Y-Axis')

plt.xlabel('X-Axis')

plt.ylim(0,80)

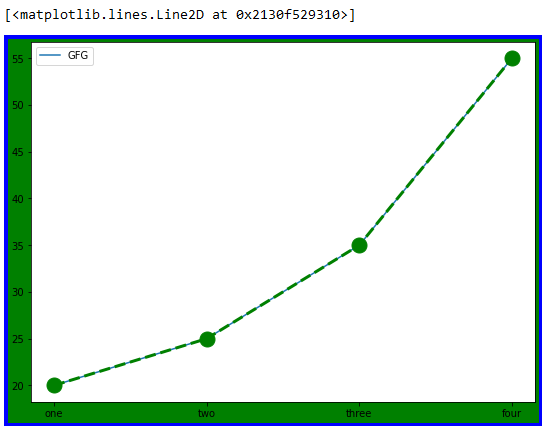
plt.subplot(122)

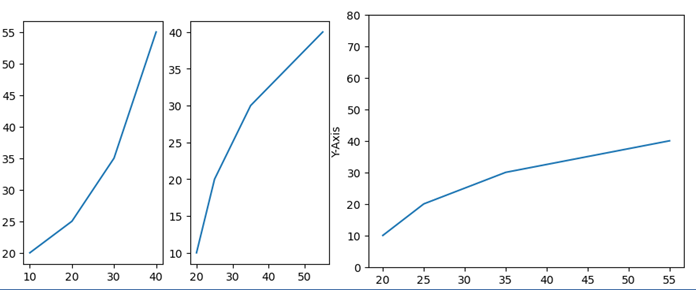
plt.plot(y,x)

plt.subplot(121)

plt.plot(x,y)

**OUTPUT**





import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

import seaborn as sns

data\_set=pd.read\_csv(r"C:\Users\BASAPPA SIR\Desktop\Dataset\salary\_Data.csv")

display(data\_set)

sns.scatterplot(x="YearsExperience",y="Salary",data=data\_set)

