



# CERTIFICATE

This is to certify that Mr. \_\_\_\_\_ student of  
M.Sc(C.S.) Semester III at Suryadatta College of Management  
Information Research & Technology (SCMIRT), Pune, has successfully  
completed the assigned practical journal in Machine Learning prescribed  
by the Savitribai Phule Pune University during the academic year 2022-  
2023.

Internal Examiner

External Examiner

HOD

Principal

Place: Pune

Date:

## INDEX

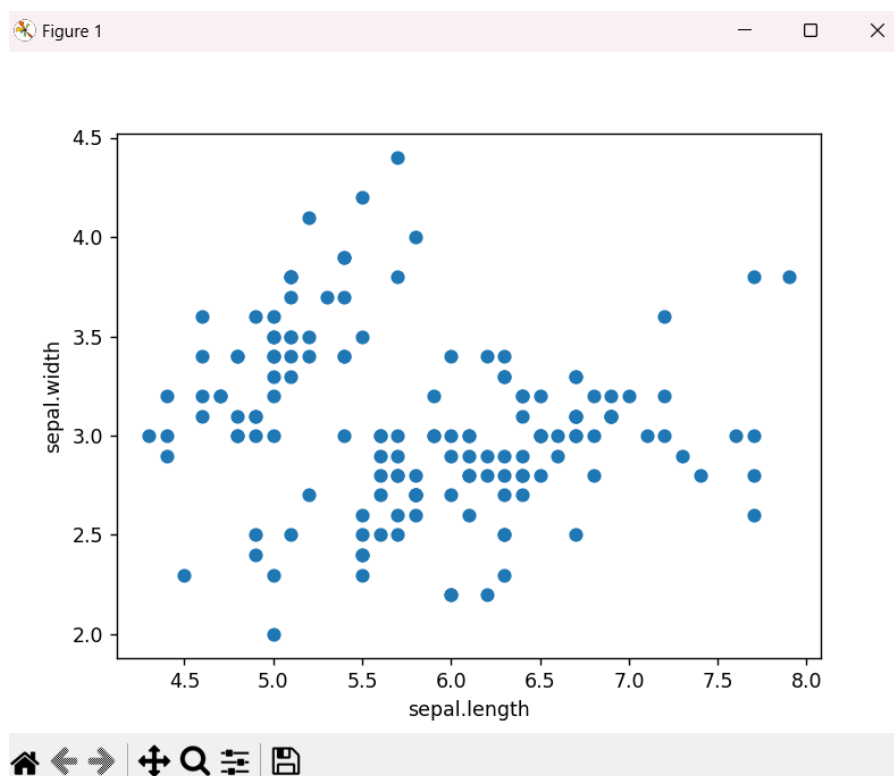
Sr.No.	Name	Page No.	Remark	Sign
1)	Write a python program to Prepare Scatter Plot (Use Forge Dataset / Iris Dataset)	3		
2)	Write a python program to find all null values in a given data set and remove them.	4 - 6		
3)	Write a python program the Categorical values in numeric format for a given dataset.	7		
4)	Write a python program to implement simple Linear Regression for predicting house price.	8 - 9		
5)	Write a python program to implement multiple Linear Regression for a given dataset.	10 - 12		
6)	Write a python program to implement Polynomial Regression for given dataset.	13 - 15		
7)	Write a python program to Implement Naïve Bayes.	16 - 17		
8)	Write a python program to Implement Decision Tree whether or not to play tennis.	18 - 19		
9)	Write a python program to implement linear SVM.	20 - 22		
10)	Write a python program to implement k-nearest Neighbors ML algorithm to build prediction model (Use Forge Dataset)	23 - 24		

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

**Ans:**

```
import pandas as pd
df = pd.read_csv('iris.csv')
print(df)
import matplotlib.pyplot as plt
plt.scatter(df['sepal.length'], df['sepal.width'])
plt.xlabel('sepal.length')
plt.ylabel('sepal.width')
plt.show()
```

**Output:**

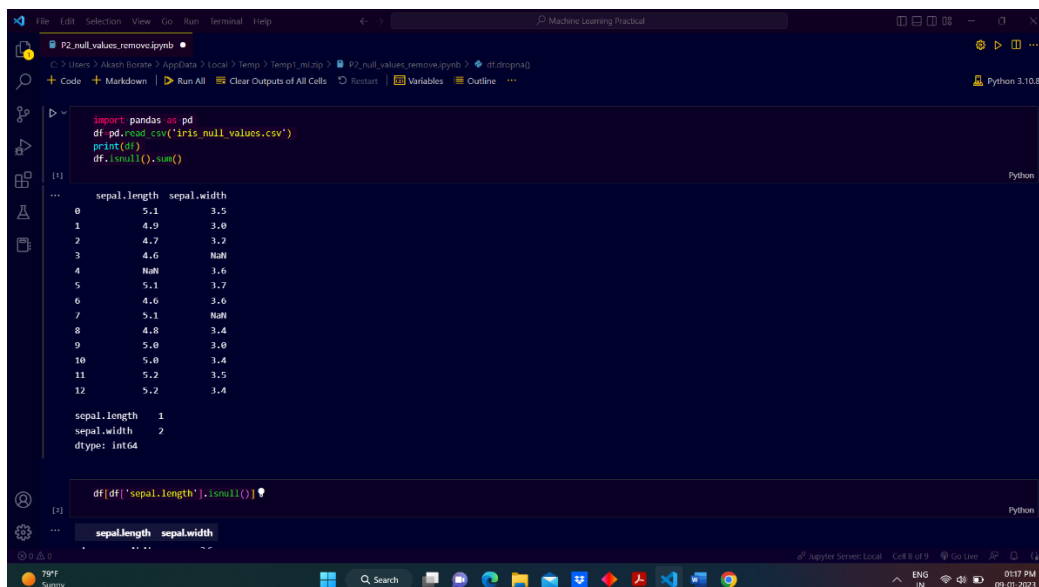


2) Write a python program to find all null values in a given data set and remove them.

**Ans:**

```
import pandas as pd
df=pd.read_csv('iris_null_values.csv')
print(df)
df.isnull().sum()
df[df['sepal.length'].isnull()]
df[df['sepal.width'].isnull()]
df.fillna('default')
df.isnull()
df.notnull()
df.isnull()
df.dropna()
```

**Output:**



The screenshot shows a Jupyter Notebook interface with the following code and output:

```
import pandas as pd
df=pd.read_csv('iris_null_values.csv')
print(df)
df.isnull().sum()
```

The output of the code is a DataFrame with two columns: 'sepal.length' and 'sepal.width'. The DataFrame contains 13 rows of data. The 'sepal.length' column has values ranging from 4.4 to 5.2, and the 'sepal.width' column has values ranging from 3.0 to 3.8. The output also shows the data types: 'sepal.length' is 'float64' and 'sepal.width' is 'float64'.

	sepal.length	sepal.width
0	5.1	3.5
1	4.9	3.0
2	4.7	3.2
3	4.6	NaN
4	NaN	3.6
5	5.1	3.7
6	4.6	3.6
7	5.1	NaN
8	4.8	3.4
9	5.0	3.0
10	5.0	3.4
11	5.2	3.5
12	5.2	3.4

The output also shows the data types: 'sepal.length' is 'float64' and 'sepal.width' is 'float64'.

```
File Edit Selection View Go Run Terminal Help
P2_null_values_remove.ipynb
C:\Users> Akash Borate > AppData > Local > Temp > temp1_m4zip > P2_null_values_remove.ipynb > df.dropna()
+ Code + Markdown | Run All | Clear Outputs of All Cells | Restart | Variables | Outline ... Python 3.10.8

[3]
df[df['sepal.length'].isnull()]
...
sepal.length sepal.width
4 NaN 3.6
+ Code + Markdown

[3]
df[df['sepal.width'].isnull()]
...
sepal.length sepal.width
3 4.6 NaN
7 5.1 NaN

[4]
df.fillna("default")
...
sepal.length sepal.width
0 5.1 3.5
1 4.9 3.0
2 4.7 3.2
3 4.6 default
4 default 3.6
5 5.1 3.7
6 4.6 3.6
7 5.1 default
8 4.8 3.4
```

```
File Edit Selection View Go Run Terminal Help
P2_null_values_remove.ipynb
C:\Users> Akash Borate > AppData > Local > Temp > temp1_m4zip > P2_null_values_remove.ipynb > df.dropna()
+ Code + Markdown | Run All | Clear Outputs of All Cells | Restart | Variables | Outline ... Python 3.10.8

[4]
df.notnull()
...
sepal.length sepal.width
0 True True
1 True True
2 True True
3 True False
4 False True
5 True True
6 True True
7 True False
8 True True
9 True True
10 True True
11 True True
12 True True

[5]
df.isnull()
...
sepal.length sepal.width
0 False False
1 False False
2 False False
```

Machine Learning Practical

P2\_null\_values\_remove.ipynb

Code | Markdown | Run All | Clear Outputs of All Cells | Restart | Variables | Outline

Python 3.10.8

```
df.isnull()
```

	sepal.length	sepal.width
0	False	False
1	False	False
2	False	False
3	False	True
4	True	False
5	False	False
6	False	False
7	False	True
8	False	False
9	False	False
10	False	False
11	False	False
12	False	False

79°F Sunny

0517 PM 09-01-2023

Machine Learning Practical

P2\_null\_values\_remove.ipynb

Code | Markdown | Run All | Clear Outputs of All Cells | Restart | Variables | Outline

Python 3.10.8

```
df.dropna()
```

	sepal.length	sepal.width
0	5.1	3.5
1	4.9	3.0
2	4.7	3.2
5	5.1	3.7
6	4.6	3.6
8	4.8	3.4
9	5.0	3.0
10	5.0	3.4
11	5.2	3.5
12	5.2	3.4

79°F Sunny

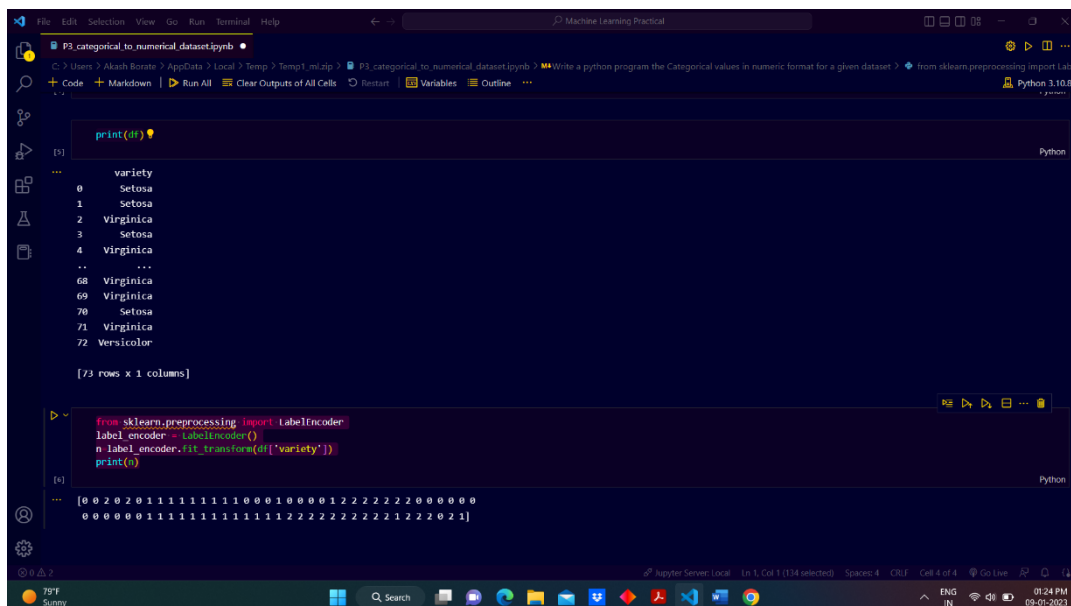
0518 PM 09-01-2023

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

**Ans:**

```
import pandas as pd
df=pd.read_csv('categorical_data.csv')
print(df)
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
n=label_encoder.fit_transform(df['variety'])
print(n)
```

**output:**



```
print(df)
```

	variety
0	Setosa
1	Setosa
2	Virginica
3	Setosa
4	Virginica
...	...
68	Virginica
69	Virginica
70	Setosa
71	Virginica
72	Versicolor

[73 rows x 1 columns]

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
n=label_encoder.fit_transform(df['variety'])
print(n)
```

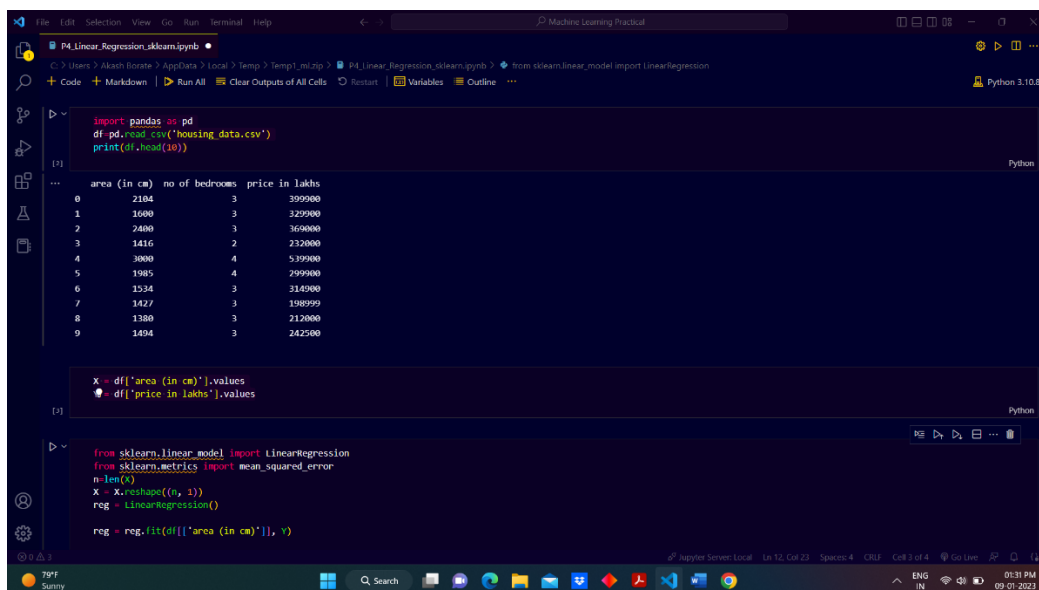
```
[0 2 0 2 0 1 1 1 1 1 1 1 1 1 0 0 0 1 0 0 0 0 1 2 2 2 2 2 2 0 0 0 0 0 0
 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 1 2 2 0 2 1]
```

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

**Ans:**

```
import pandas as pd
df=pd.read_csv('housing_data.csv')
print(df.head(10))
X = df['area (in cm)'].values
Y = df['price in lakhs'].values
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
n=len(X)
X = X.reshape((n, 1))
reg = LinearRegression()
reg = reg.fit(df[['area (in cm)']], Y)
Y_pred = reg.predict(X)
print(Y_pred)
# Calculating R2 Score
r2_score = reg.score(X, Y)
print(r2_score)
```

**Output:**



```
import pandas as pd
df=pd.read_csv('housing_data.csv')
print(df.head(10))
```

	area (in cm)	no of bedrooms	price in lakhs
0	2104	3	399900
1	1600	3	329900
2	2400	3	369900
3	1416	2	232900
4	3800	4	539900
5	1985	4	299900
6	1534	3	314900
7	1427	3	198999
8	1380	3	212000
9	1494	3	242500

```
x = df['area (in cm)'].values
y = df['price in lakhs'].values
```

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
n=len(X)
X = X.reshape((n, 1))
reg = LinearRegression()

reg = reg.fit(df[['area (in cm)']], y)
```



```
File Edit Selection View Go Run Terminal Help
Machine Learning Practical

P4_Linear_Regression_sklearn.ipynb
> Users > Akash Borate > AppData > Local > temp > temp1_ml.zip > P4_Linear_Regression_sklearn.ipynb > from sklearn.linear_model import LinearRegression
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Variables Outline ... Python 3.10.8

[P1]
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
n=len(X)
X = X.reshape((n, 1))
reg = LinearRegression()

reg = reg.fit(df[['area (in cm)']], y)

y_pred = reg.predict(X)
print(y_pred)

# Calculating R2 Score
r2_score = reg.score(X, y)

print(r2_score)

[P2]
...
[354311.69781212 286510.95280112 394131.18297731 261758.29986059
474846.35560945 338103.18857341 277632.28311158 263238.07802551
256915.18950266 272251.27230277 332249.550626 340321.06788921
325523.28623999 673674.73889997 241848.557278 380678.65420528
248843.87223945 237543.74807095 422246.96811084 479016.63952878
308076.6758504 325254.23566454 287049.053952 335209.10695584
594573.86168047 219248.30894099 267408.36194484 411081.36923006
367226.12543326 426013.67616701 318662.49656625 205795.78016897
345702.07938802 493276.32002713 314895.78851009 264583.33090272
237947.32393411 358078.40586828 638294.58018955 362114.16449989
295120.57121521 172338.08636663 416596.90602659 232700.83771302
105886.03758637 320411.32530662 233104.41357618]
0.7310037839755306

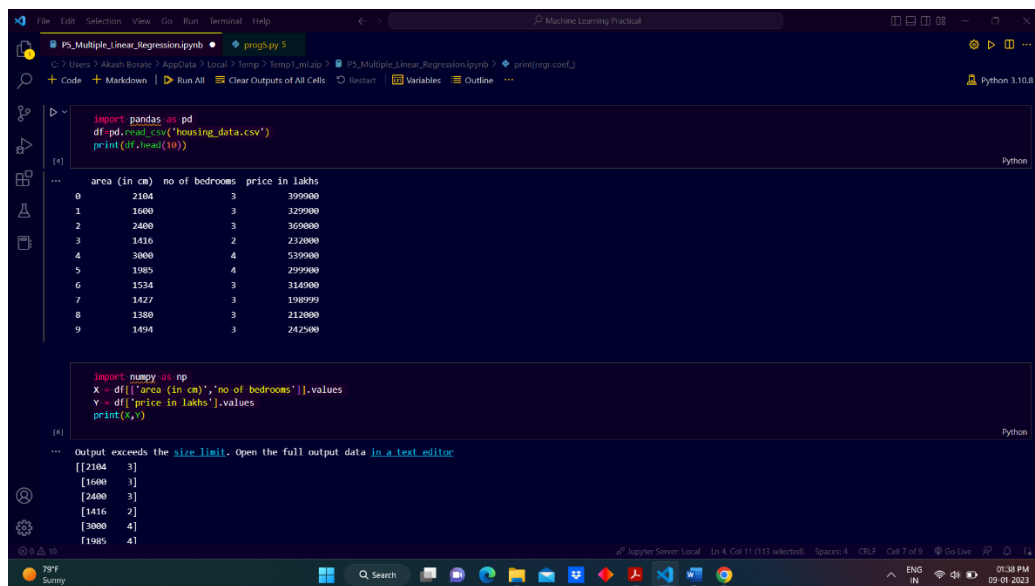
Jupyter Server: Local In 12, Col 23 Spaces: 4 CRLF Cell 3 of 4 Go Live Python
```

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

**Ans:**

```
import pandas as pd
df=pd.read_csv('housing_data.csv')
print(df.head(10))
import numpy as np
X = df[['area (in cm)','no of bedrooms']].values
Y = df['price in lakhs'].values
print(X,Y)
from sklearn import linear_model
regr = linear_model.LinearRegression()
regr.fit(X, Y)
predicted_y = regr.predict([[1300, 4]])
print(predicted_y)
print(regr.coef_)
import pandas as pd
df=pd.read_csv('student.csv')
print(df)
from sklearn import linear_model
X = df[['IQ','Study hours']].values
Y = df['Test score'].values
regr = linear_model.LinearRegression()
regr.fit(X, Y)
predicted_y = regr.predict([[110, 40]])
print(predicted_y)
print(regr.coef_)
print(regr.intercept_)
```

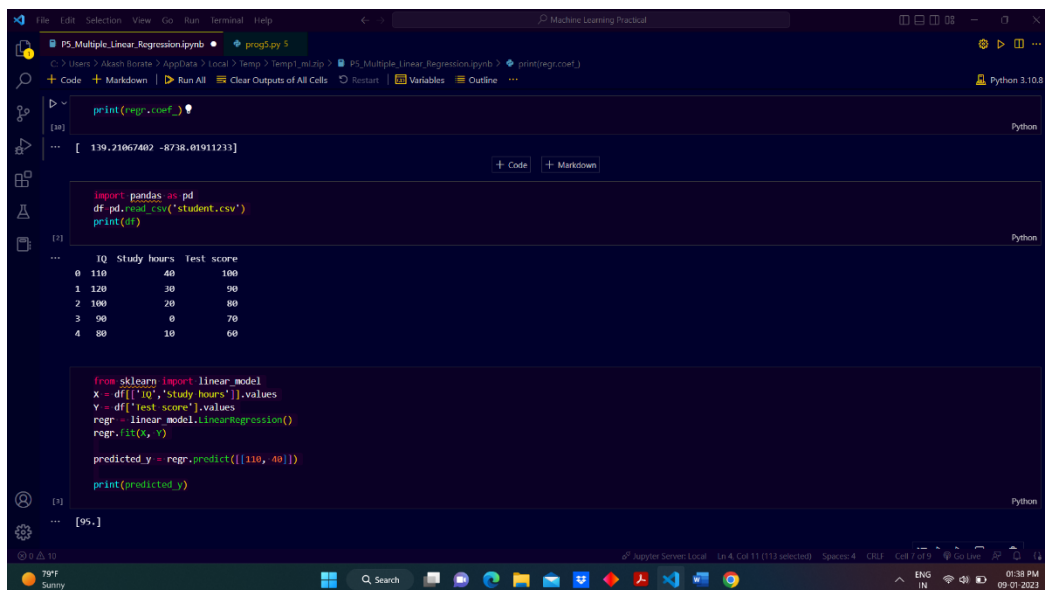
## Output:



```
import pandas as pd
df=pd.read_csv('housing_data.csv')
print(df.head(10))
```

	area (in cm)	no of bedrooms	price in lakhs
0	2104	3	399900
1	1600	3	329900
2	2400	3	369000
3	1416	2	232000
4	3800	4	539900
5	1985	4	299900
6	1534	3	316000
7	1427	3	198999
8	1380	3	212000
9	1494	3	242500

```
import numpy as np
X=df[['area (in cm)','no of bedrooms']].values
Y=df['price in lakhs'].values
print(X,Y)
```



```
print(regr.coef_)
```

```
import pandas as pd
df=pd.read_csv('student.csv')
print(df)
```

	IQ	Study hours	Test score
0	110	40	90
1	120	30	90
2	100	20	80
3	90	0	70
4	80	10	60

```
from sklearn import linear_model
X = df[['IQ','study hours']].values
Y = df['test score'].values
regr = linear_model.LinearRegression()
regr.fit(X,Y)

predicted_y = regr.predict([[110, 40]])
print(predicted_y)
```

```
File Edit Selection View Go Run Terminal Help
Machine Learning Practical
PS_Multiple_Linear_Regression.ipynb • progress 5
C:\Users> Akash Borate > AppData > Local > Temp > Temp1\ml\ip > PS_Multiple_Linear_Regression.ipynb > print(reg.coef_)
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Variables Outline ... Python 3.10.8

[1940 4]
[2080 3]
[1890 3]
[4478 5]
[1268 3]
[2380 4]
[1328 2]
[1236 3]
[2689 4]
[3031 4]
[1767 3]
[1888 2]
[1684 3]
[1962 4]
[1890 3]
...
239999 347000 329999 609900 259900 449900 299900 199900 499998 599000
252000 255000 242900 259900 572900 249900 464500 469000 475000 299900
349900 169900 314900 179900 285900 249900 229900 345000 549000 287000
368500 329900 314000 299000 179900 299900 239500]

from sklearn import linear_model
regr = linear_model.LinearRegression()
regr.fit(X, y)

predicted_y = regr.predict([[1300, 4]])
print(predicted_y)

[235619.7893164]
```

```
File Edit Selection View Go Run Terminal Help
Machine Learning Practical
PS_Multiple_Linear_Regression.ipynb • progress 5
C:\Users> Akash Borate > AppData > Local > Temp > Temp1\ml\ip > PS_Multiple_Linear_Regression.ipynb > print(reg.coef_)
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Variables Outline ... Python 3.10.8

2 100 20 80
3 90 0 70
4 80 10 60

from sklearn import linear_model
X = df[['IQ', 'Study hours']].values
Y = df[['Test score']].values
regr = linear_model.LinearRegression()
regr.fit(X, y)

predicted_y = regr.predict([[110, 40]])
print(predicted_y)

[95.]

print(regr.coef_)

[0.5 0.5]

print(regr.intercept_)

20.000000000000004
```

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

**Ans:**

```
import pandas as pd
df=pd.read_csv('employees.csv')
print(df)
X=df.iloc[:,1:2].values
y=df.iloc[:,2].values
print(X,y)
#fitting the polynomial regression model to the dataset
from sklearn.preprocessing import PolynomialFeatures
poly_reg=PolynomialFeatures(degree=4)
X_poly=poly_reg.fit_transform(X)
poly_reg.fit(X_poly,y)
lin_reg2=LinearRegression()
lin_reg2.fit(X_poly,y)
#Visualising the polynomial regression model results
import numpy as np
X_grid=np.arange(min(X),max(X),0.1)
X_grid=X_grid.reshape((len(X_grid),1))
plt.scatter(X,y,color='red')
plt.plot(X,lin_reg2.predict(poly_reg.fit_transform(X)),color='blue')
plt.title('(Polynomial Regression)')
plt.xlabel('Position Level')
plt.ylabel('Salary')
plt.show()
from sklearn.linear_model import LinearRegression
lin_reg=LinearRegression()
lin_reg.fit(X,y)
import matplotlib.pyplot as plt
plt.scatter(X,y,color='red')
plt.plot(X,lin_reg.predict(X),color='blue')
plt.title('(Linear Regression)')
plt.xlabel('Position Level')
plt.ylabel('Salary')
plt.show()
```

```
import numpy as np
lin_reg.predict(np.array([ [6.5] ]))
import numpy as np
lin_reg2.predict(poly_reg.fit_transform(np.array([ [6.5] ])))
```

## Output:

```
import pandas as pd
df=pd.read_csv('housing_data.csv')
print(df.head(10))
```

	area (in sq)	no of bedrooms	price in lakhs
0	2104	3	399900
1	1600	3	329900
2	2400	3	369900
3	1416	2	232000
4	3800	4	539900
5	1985	4	299900
6	1534	3	314000
7	1427	3	198999
8	1800	3	212000
9	1404	3	242500

```
import numpy as np
X = df[['area (in sq)','no of bedrooms']].values
Y = df['price in lakhs'].values
print(X,Y)
```

```
[[2104 3]
 [1600 3]
 [2400 3]
 [1416 2]
 [3800 4]
 [1985 4]
 [1534 3]
 [1427 3]
 [1800 3]
 [1404 3]]
```

```
...
[1304 4]
[2000 3]
[1800 3]
[4478 5]
[1268 3]
[2300 4]
[1328 2]
[1236 3]
[2609 4]
[3031 4]
[1767 3]
[1888 2]
[1604 3]
[1962 4]
[3830 3]
....
[220999 347000 320999 699900 259900 449900 299900 199900 499998 599000
 252900 255900 242900 259900 573900 249900 464500 469000 475900 299900
 349900 169900 114900 579900 285900 249900 229900 345000 549000 287000
 368500 129900 314000 299000 179900 299900 239500]
```

```
from sklearn import linear_model
regr = linear_model.LinearRegression()
regr.fit(X, Y)

predicted_y = regr.predict([[1300, 4]])

print(predicted_y)
```

```
[235619.7093164]
```

```
File Edit Selection View Go Run Terminal Help
P5_Multiple_Linear_Regression.ipynb • prog5.py 3
C:\Users\Akash Borate> AppData\Local\Temp\Temp1_m1.zip P5_Multiple_Linear_Regression.ipynb print(regr.coef_)
+ Code + Markdown Run All Clear Outputs of All Cells Restart Variables Outline Python 3.10.8

[39] Python
... [ 130.21067402 -8738.01911233]

+ Code + Markdown

import pandas as pd
df=pd.read_csv("student.csv")
print(df)

[42] Python
...
   IQ  Study hours  Test score
0  110           40          100
1  120           30           90
2  100           20           80
3   90            0           70
4   80           10           60

from sklearn import linear_model
X = df[["IQ", "Study hours"]].values
Y = df["Test score"].values
regr = linear_model.LinearRegression()
regr.fit(X, Y)

predicted_y = regr.predict([[110, 40]])
print(predicted_y)

[43] Python
... [95.]
```

```
File Edit Selection View Go Run Terminal Help
P5_Multiple_Linear_Regression.ipynb • prog5.py 5
C:\Users\Akash Borate> AppData\Local\Temp\Temp1_m1.zip P5_Multiple_Linear_Regression.ipynb print(regr.coef_)
+ Code + Markdown Run All Clear Outputs of All Cells Restart Variables Outline Python 3.10.8

[42] Python
...
   IQ  Study hours  Test score
0  110           40          100
1  120           30           90
2  100           20           80
3   90            0           70
4   80           10           60

from sklearn import linear_model
X = df[["IQ", "Study hours"]].values
Y = df["Test score"].values
regr = linear_model.LinearRegression()
regr.fit(X, Y)

predicted_y = regr.predict([[110, 40]])
print(predicted_y)

[43] Python
... [95.]

[44] Python
... [0.5 0.5]

print(regr.intercept_)

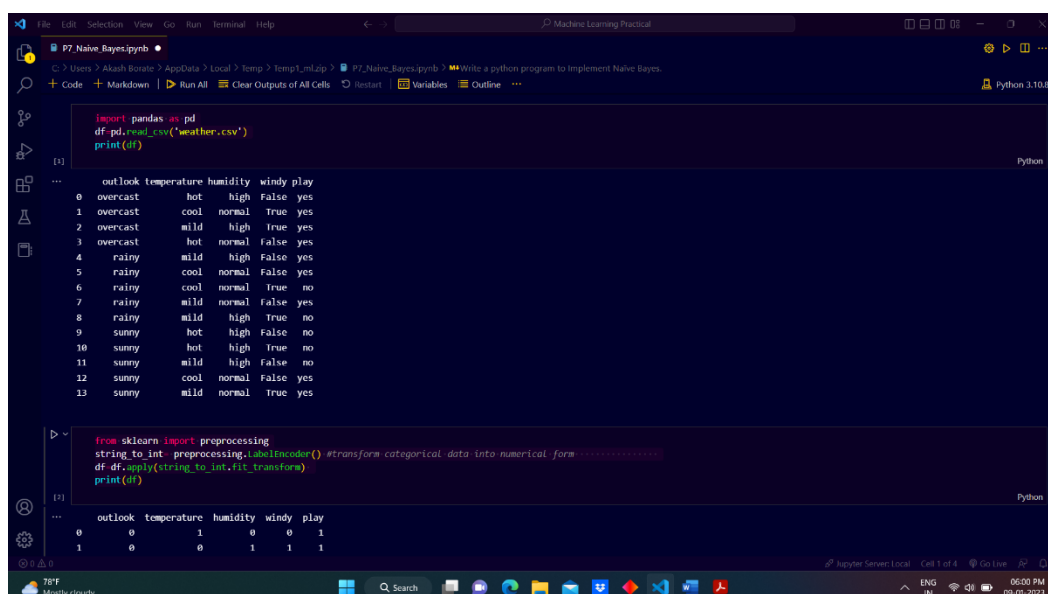
[45] Python
... 20.000000000000014
```

## 7) Write a python program to Implement Naïve Bayes.

**Ans:**

```
import pandas as pd
df=pd.read_csv('weather.csv')
print(df)
from sklearn import preprocessing
string_to_int= preprocessing.LabelEncoder() #transform categorical data into
numerical form
df=df.apply(string_to_int.fit_transform)
print(df)
from sklearn.naive_bayes import GaussianNB
X = df[['outlook','temperature','humidity','windy']]
y= df['play']
#Create a Gaussian Classifier
model = GaussianNB()
# Train the model using the training sets
model.fit(X,y)
#Predict Output
predicted= model.predict(X) # 2:sunny, 0:cool, 1:normal, 1:true
print("Predicted Value:", predicted)
```

**Output:**



```
import pandas as pd
df=pd.read_csv('weather.csv')
print(df)
```

	outlook	temperature	humidity	windy	play
0	overcast	hot	high	false	yes
1	overcast	cool	normal	true	yes
2	overcast	mild	high	true	yes
3	overcast	hot	normal	false	yes
4	rainy	mild	high	false	yes
5	rainy	cool	normal	false	yes
6	rainy	cool	normal	true	no
7	rainy	mild	normal	false	yes
8	rainy	mild	high	true	no
9	sunny	hot	high	false	no
10	sunny	hot	high	true	no
11	sunny	mild	high	false	no
12	sunny	cool	normal	false	yes
13	sunny	mild	normal	true	yes

```
from sklearn import preprocessing
string_to_int= preprocessing.LabelEncoder() #transform categorical data into numerical form
df=df.apply(string_to_int.fit_transform)
print(df)
```

	outlook	temperature	humidity	windy	play
0	0	1	0	0	1
1	0	0	0	1	1



The screenshot shows a Jupyter Notebook interface with a file explorer on the left and a code editor on the right. The notebook is titled 'P7\_Naive\_Bayes.ipynb'. The code editor contains the following Python code:

```
from sklearn.naive_bayes import GaussianNB

x = df[['outlook', 'temperature', 'humidity', 'windy']]
y = df['play']

# Create a Gaussian classifier
model = GaussianNB()

# Train the model using the training sets
model.fit(x, y)

# Predict output
predicted = model.predict(x)
print("Predicted Value:", predicted)
```

The output of the code is displayed below the code cell:

```
Predicted Value: [1 1 1 1 1 1 1 0 0 0 0 1 1]
```

The dataset used in the code is as follows:

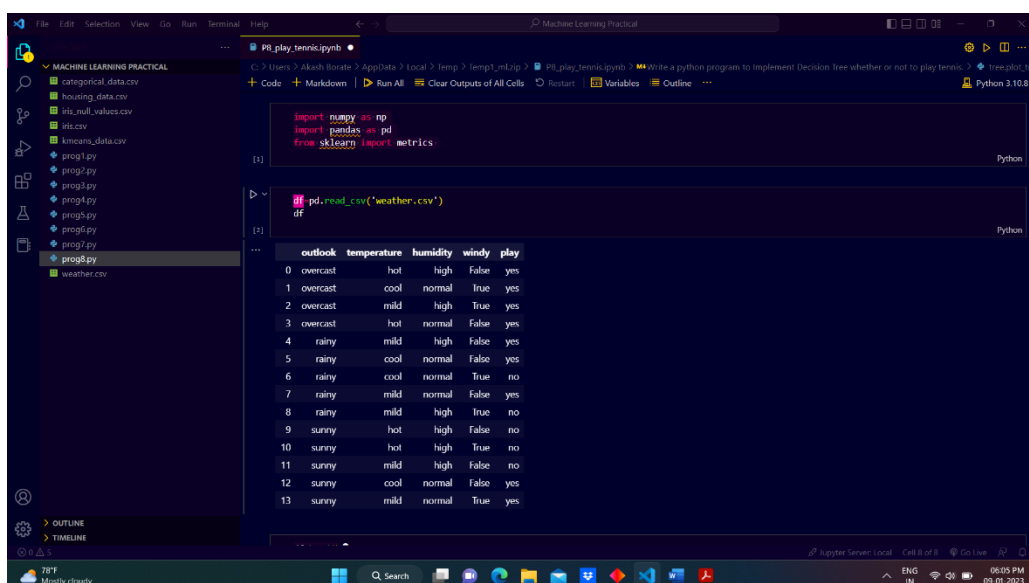
	0	1	2	0	0	1
0	0	0	1	0	0	1
1	0	0	0	1	1	1
2	0	0	2	0	1	1
3	0	1	1	1	0	1
4	1	2	2	0	0	1
5	1	0	0	1	0	1
6	1	0	0	1	1	0
7	1	2	1	0	1	1
8	1	2	0	1	0	0
9	2	1	0	0	0	0
10	2	1	0	1	0	0
11	2	2	0	0	0	0
12	2	0	1	0	1	1
13	2	2	1	1	1	1

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

**Ans:**

```
import numpy as np
import pandas as pd
from sklearn import metrics
df=pd.read_csv('weather.csv')
df
df.head()
from sklearn import preprocessing
string_to_int= preprocessing.LabelEncoder() #transform categorical data into
numerical form
df=df.apply(string_to_int.fit_transform)
df
X = df[['outlook','temperature','humidity','windy']]
y= df['play']
from sklearn import tree
clf = tree.DecisionTreeClassifier(criterion = 'entropy')
clf = clf.fit(X, y)
tree.plot_tree(clf)
```

**Output:**



```
import numpy as np
import pandas as pd
from sklearn import metrics

df = pd.read_csv('weather.csv')
df
```

	outlook	temperature	humidity	windy	play
0	overcast	hot	high	False	yes
1	overcast	cool	normal	True	yes
2	overcast	mild	high	True	yes
3	overcast	hot	normal	False	yes
4	rainy	mild	high	False	yes
5	rainy	cool	normal	False	yes
6	rainy	cool	normal	True	no
7	rainy	mild	normal	False	yes
8	rainy	mild	high	True	no
9	sunny	hot	high	False	no
10	sunny	hot	high	True	no
11	sunny	mild	high	False	no
12	sunny	cool	normal	False	yes
13	sunny	mild	normal	True	yes

Machine Learning Practical

File Edit Selection View Go Run Terminal Help

Python 3.10.8

df.head()

```

0    overcast    hot    high    False    yes
1    overcast    cool   normal    True    yes
2    overcast    mild   high    True    yes
3    overcast    hot    normal   False    yes
4    rainy       mild   high    False    yes

```

```

from sklearn import preprocessing
string_to_int = preprocessing.LabelEncoder() #transform categorical data into numerical form
df = df.apply(string_to_int.fit_transform)
df

```

```

0    0    1    0    0    1
1    0    0    1    1    1
2    0    2    0    1    1
3    0    1    1    0    1
4    1    2    0    0    1
5    1    0    1    0    1
6    1    0    1    1    0
7    1    2    1    0    1
8    1    2    0    1    0
9    2    1    0    0    0

```

78°F Mostly cloudy

06:05 PM 09-01-2023

Machine Learning Practical

File Edit Selection View Go Run Terminal Help

Python 3.10.8

tree.plot\_tree(df)

```

[Text(133.92000000000002, 195.696, 'X[0] <= 0.5\nentropy = 0.94\nsamples = 14\nvalue = [5, 9]'),
Text(100.44000000000001, 152.208, 'entropy = 0.0\nsamples = 4\nvalue = [0, 4]'),
Text(167.40000000000003, 152.208, 'X[2] <= 0.5\nentropy = 1.0\nsamples = 10\nvalue = [5, 5]'),
Text(100.44000000000001, 108.72, 'X[0] <= 1.5\nentropy = 0.722\nsamples = 5\nvalue = [4, 1]'),
Text(66.96000000000001, 65.232, 'X[3] <= 0.5\nentropy = 1.0\nsamples = 2\nvalue = [1, 1]'),
Text(33.480000000000004, 21.744, 'entropy = 0.0\nsamples = 1\nvalue = [0, 1]'),
Text(100.44000000000001, 21.744, 'entropy = 0.0\nsamples = 1\nvalue = [1, 0]'),
Text(133.92000000000002, 65.232, 'entropy = 0.0\nsamples = 3\nvalue = [3, 0]'),
Text(234.36, 108.72, 'X[3] <= 0.5\nentropy = 0.722\nsamples = 5\nvalue = [1, 4]'),
Text(200.88000000000002, 65.232, 'entropy = 0.0\nsamples = 3\nvalue = [0, 3]'),
Text(267.84000000000003, 65.232, 'X[1] <= 1.0\nentropy = 1.0\nsamples = 2\nvalue = [1, 1]'),
Text(234.36, 21.744, 'entropy = 0.0\nsamples = 1\nvalue = [1, 0]'),
Text(301.32000000000005, 21.744, 'entropy = 0.0\nsamples = 1\nvalue = [0, 1]')]

```

78°F Mostly cloudy

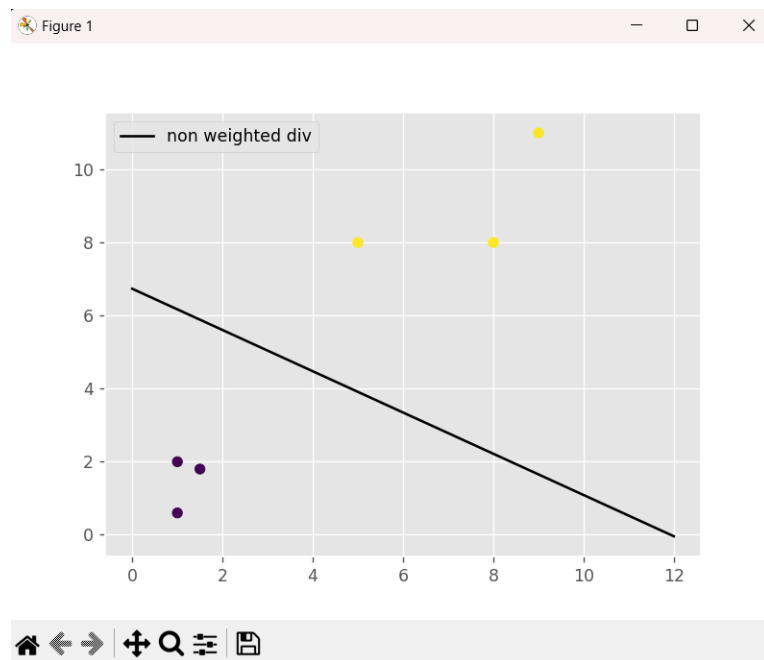
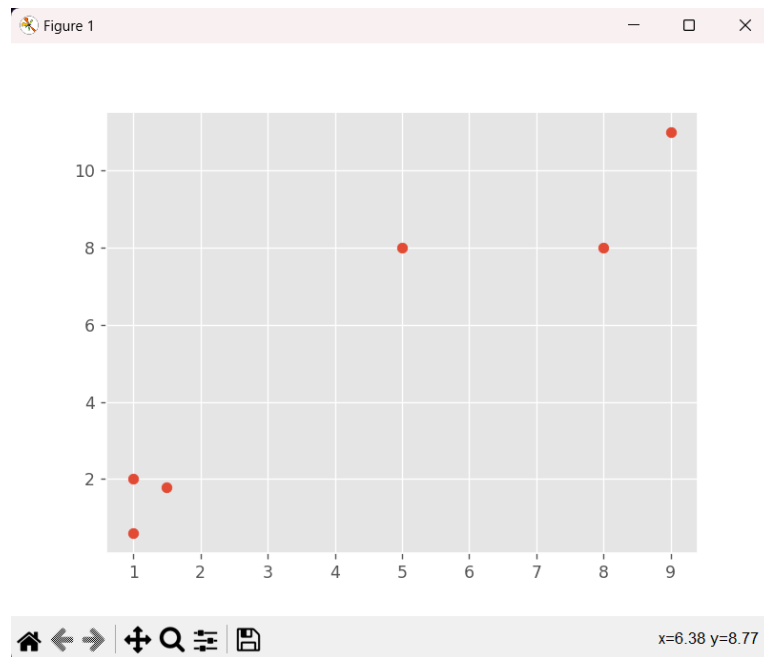
06:06 PM 09-01-2023

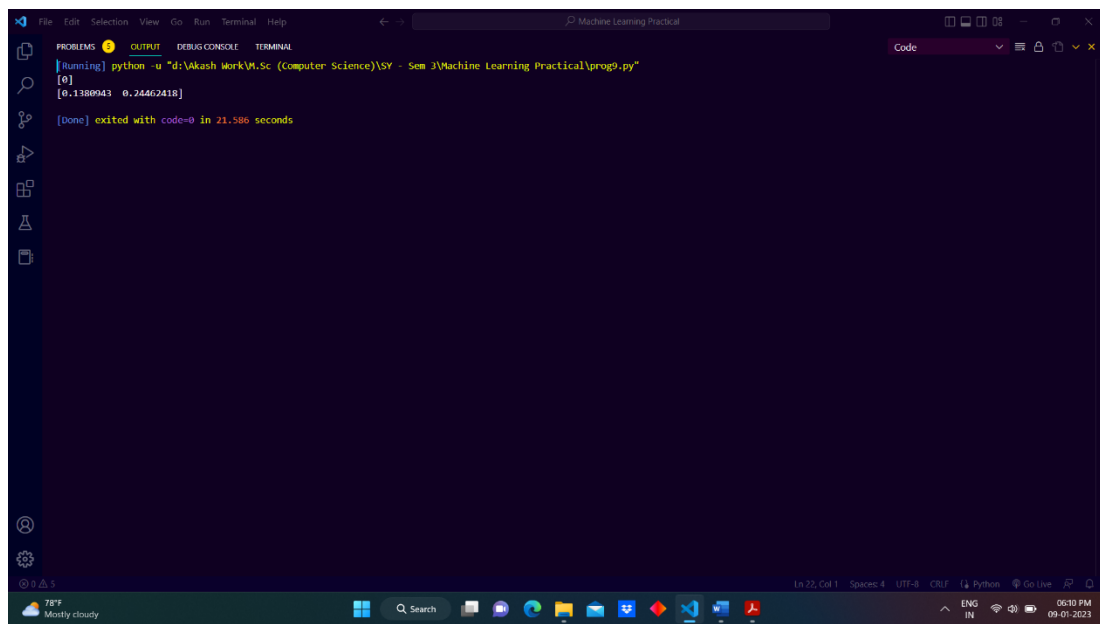
**9) Write a python program to implement linear SVM.**

**Ans:**

```
x = [1, 5, 1.5, 8, 1, 9]
y = [2, 8, 1.8, 8, 0.6, 11]
import matplotlib.pyplot as plt
from matplotlib import style
style.use("ggplot")
plt.scatter(x,y)
plt.show()
import numpy as np
from sklearn import svm
X = np.array([[1,2],
              [5,8],
              [1.5,1.8],
              [8,8],
              [1,0.6],
              [9,11]])
y = [0,1,0,1,0,1]
clf = svm.SVC(kernel='linear', C = 1.0)
clf.fit(X,y)
print(clf.predict([[0.58,0.76]]))
w = clf.coef_[0]
print(w)
a = -w[0] / w[1]
xx = np.linspace(0,12)
yy = a * xx - clf.intercept_[0] / w[1]
h0 = plt.plot(xx, yy, 'k-', label="non weighted div")
plt.scatter(X[:, 0], X[:, 1], c = y)
plt.legend()
plt.show()
```

## Output:





The image shows a screenshot of a Visual Studio Code (VS Code) terminal window. The terminal is running a Python script. The output of the script is displayed in the terminal window. The output shows a list of two numbers: [0] and [0.1380943 0.24462418]. The terminal also shows that the script exited with code=0 in 21.586 seconds. The VS Code interface includes a menu bar at the top with options like File, Edit, Selection, View, Go, Run, Terminal, and Help. The terminal window is titled "Machine Learning Practical". The status bar at the bottom shows the current file path, line and column numbers, and the encoding of the file.

```
File Edit Selection View Go Run Terminal Help
Machine Learning Practical

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
[Running] python -u "d:\Akash Work\W.Sc (computer Science)\SY - Sem 3\Machine Learning Practical\prog9.py"
[0]
[0.1380943 0.24462418]
[Done] exited with code=0 in 21.586 seconds

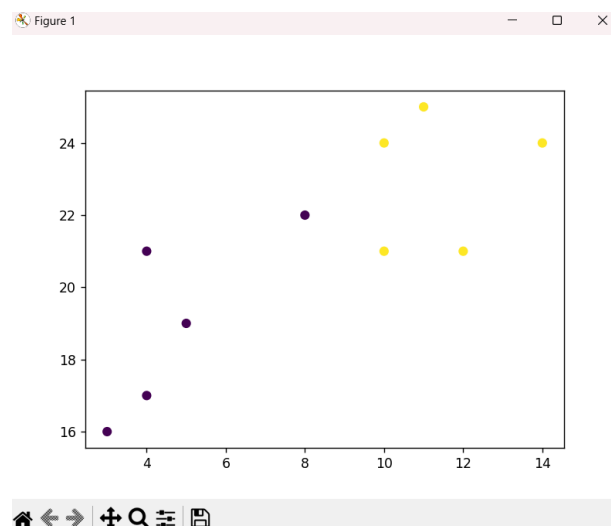
1x22, Col 1 Spaces: 4 UTF-8 CR/LF Python Go Live
```

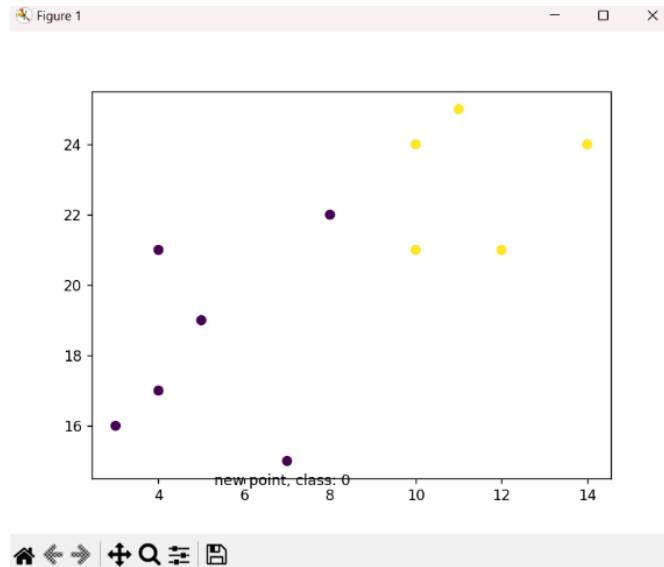
**10) Write a python program to implement k-nearest Neighbors ML algorithm to build prediction model (Use Forge Dataset)**

**Ans:**

```
x = [4, 5, 10, 4, 3, 11, 14, 8, 10, 12]
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]
classes = [0, 0, 1, 0, 0, 1, 1, 0, 1, 1]
data=list(zip(x,y))
data
import matplotlib.pyplot as plt
plt.scatter(x, y, c=classes)
plt.show()
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(data, classes)
new_x = 7
new_y = 15
new_point = [(new_x, new_y)]
prediction = knn.predict(new_point)
new_point
prediction
plt.scatter(x + [new_x], y + [new_y], c=classes + [prediction[0]])
plt.text(x=new_x-1.7, y=new_y-0.7, s=f"new point, class: {prediction[0]}")
plt.show()
```

**Output:**





```
File Edit Selection View Go Run Terminal Help
Machine Learning Practical

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
[Running] python -u "d:\akash\work\MLSc (Computer Science)\SV - Sem 3\Machine Learning Practical\prog12.py"
[Done] exited with code=0 in 12.835 seconds

[Running] python -u "d:\akash\work\MLSc (Computer Science)\SV - Sem 3\Machine Learning Practical\prog12.py"
[[4, 21], [5, 19], [10, 24], [4, 17], [3, 16], [11, 25], [14, 24], [8, 22], [10, 21], [12, 21]]
[[7, 15]]
[0]
[Done] exited with code=0 in 7.487 seconds

Apyter Server: Local Ln 5, Col 3 (155 selected) Sources 4 CTRL Col 2 of 8 Go Live
73°F Mostly cloudy 08:24 PM 08/31/2023
```