MACHINE LEARNING LAB-1:

Code:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report
d1=pd.read_excel('C:\\Users\\SAI REVANTH\\Downloads\\Lab Session1 Data.xlsx',sheet_name='Purchase data')
d1.drop(d1.iloc[:,5:22],inplace=True,axis=1)
A=d1.iloc[:,1:-1].values
C=d1.iloc[:,-1].values
A=np.array(A)
C=np.array(C)
print("Matrix of A:")
print(A)
print("Matrix of C:")
print(C)
rank=np.linalg.matrix_rank(A)
print("Rank of Matrix A:", rank)
inverse=np.linalg.pinv(A)
print("Inverse of A: ",inverse)
Pseudo_inv=np.matmul(inverse,C)
print("Pseudo inverse is ie actual cost of each product is : ",Pseudo_inv)
t=np.array(d1['Payment (Rs)'])
number=len(t)
New_cat=[]
for i in range(0,number):
    if t[i]>200:
        New_cat.append('RICH')
```

```
    else:
        New_cat.append('POOR')
d1.insert(loc = 5,column = 'Label',value = New_cat)
print("New Data Excel Sheet for Purchase Data is: ")
print(d1)
X = d1.drop(['Customer', 'Payment (Rs)', 'Label'], axis=1)
y = d1['Label']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
scaler = StandardScaler() # Feature scaling
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
model = RandomForestClassifier(random_state=42)
model.fit(X_train_scaled, y_train)
y_pred = model.predict(X_test_scaled)
print(classification_report(y_test, y_pred))
```

In [26]:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report
d1=pd.read_excel('C:\\Users\\SAI REVANTH\\Downloads\\Lab Session1 Data.xlsx',sheet_name='Purchase data')
d1.drop(d1.iloc[:,5:22],inplace=True,axis=1)
A=d1.iloc[:,1:-1].values
C=d1.iloc[:,-1].values
A=np.array(A)
C=np.array(C)
print("Matrix of A:")
print(A)
print("Matrix of C:")
print(C)
rank=np.linalg.matrix_rank(A)
print("Rank of Matrix A:", rank)
inverse=np.linalg.pinv(A)
print("Inverse of A: ",inverse)
Pseudo_inv=np.matmul(inverse,C)
print("Pseudo inverse is ie actual cost of each product is : ",Pseudo_inv)
t=np.array(d1['Payment (Rs)'])
number=len(t)
New_cat=[]
for i in range(0,number):
    if t[i]>200:
        New_cat.append('RICH')
    else:
        New_cat.append('POOR')
d1.insert(loc = 5,column = 'Label',value = New_cat)
print("New Data Excel Sheet for Purchase Data is: ")
print(d1)
X = d1.drop(['Customer', 'Payment (Rs)', 'Label'], axis=1)
y = d1['Label']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
scaler = StandardScaler() # Feature scaling
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
model = RandomForestClassifier(random_state=42)
model.fit(X_train_scaled, y_train)
y_pred = model.predict(X_test_scaled)
print(classification_report(y_test, y_pred))
```

```
Matrix of A:
[[20  6  2]
 [16  3  6]
 [27  6  2]
 [19  1  2]
 [24  4  2]
 [22  1  5]
 [15  4  2]
 [18  4  2]
 [21  1  4]
 [16  2  4]]
Matrix of C:
[386 289 393 110 280 167 271 274 148 198]
Rank of Matrix A: 3
Inverse of A:  [[-0.01008596 -0.03124505  0.01013951  0.0290728   0.0182907   0.01161794
  -0.00771348  0.00095458  0.01743623 -0.00542016]
 [ 0.09059668  0.07263726  0.03172933 -0.09071908 -0.01893196 -0.06926996
   0.05675464  0.03152577 -0.07641966  0.00357352]
 [ 0.00299878  0.15874243 -0.05795468 -0.06609024 -0.06295043  0.03348017
   0.01541831 -0.01070461  0.00029003  0.05938755]]
Pseudo inverse is ie actual cost of each product is :  [ 1. 55. 18.]
New Data Excel Sheet for Purchase Data is:
  Customer  Candies (#)  Mangoes (Kg)  Milk Packets (#)  Payment (Rs) Label
0      C_1           20             6                 2           386  RICH
1      C_2           16             3                 6           289  RICH
2      C_3           27             6                 2           393  RICH
3      C_4           19             1                 2           110  POOR
4      C_5           24             4                 2           280  RICH
5      C_6           22             1                 5           167  POOR
6      C_7           15             4                 2           271  RICH
7      C_8           18             4                 2           274  RICH
8      C_9           21             1                 4           148  POOR
9     C_10           16             2                 4           198  POOR
              precision    recall  f1-score   support

        POOR       1.00      1.00      1.00         1
        RICH       1.00      1.00      1.00         1

    accuracy                           1.00         2
   macro avg       1.00      1.00      1.00         2
weighted avg       1.00      1.00      1.00         2
```

Code:

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
df = pd.read_excel("C:\\Users\\SAI REVANTH\\Downloads\\Lab Session1 Data.xlsx",sheet_name="IRCTC Stock Price")
price_data=df['Price']
mean=np.mean(price_data)
var=np.var(price_data)
print('The Mean is:', mean)
print('The Variance is:', var)
wednesday_data = price_data[df['Day'] == 'Wed']
sample_mean = np.mean(wednesday_data)
print('Sample mean is:', sample_mean)
apdata = price_data[df['Month'] == 'Apr']
apmean = np.mean(apdata)
print('April mean is:', apmean)
chg_data = df['Chg%']
is_loss = np.where(chg_data > 0, False, True)
lossprob = np.mean(is_loss)
print('Probability of making a loss:', lossprob)
weddata = df[df['Day'] == 'Wed']
wedprofit = np.mean(weddata['Chg%'] > 0)
print('Wednesday profit probability is:', wedprofit)
wedprob = np.mean(df['Day'] == 'Wed')
cdprob = (wedprofit / wedprob)
print('Conditional Probability on Wednesday is:', cdprob)
plt.scatter(df['Day'], df['Chg%'])
plt.xlabel('Day')
plt.ylabel('Chg%')
plt.title('Chg% vs Day')
```

plt.show()

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
df = pd.read_excel("C:\\Users\\SAI REVANTH\\Downloads\\Lab Session1 Data.xlsx",sheet_name="IRCTC Stock Price")
price_data=df['Price']
mean=np.mean(price_data)
var=np.var(price_data)
print('The Mean is:', mean)
print('The Variance is:', var)
wednesday_data = price_data[df['Day'] == 'Wed']
sample_mean = np.mean(wednesday_data)
print('Sample mean is:', sample_mean)
apdata = price_data[df['Month'] == 'Apr']
apmean = np.mean(apdata)
print('April mean is:', apmean)
chg_data = df['Chg%']
is_loss = np.where(chg_data > 0, False, True)
lossprob = np.mean(is_loss)
print('Probability of making a loss:', lossprob)
weddata = df[df['Day'] == 'Wed']
wedprofit = np.mean(weddata['Chg%'] > 0)
print('Wednesday profit probability is:', wedprofit)
wedprob = np.mean(df['Day'] == 'Wed')
cdprob = (wedprofit / wedprob)
print('Conditional Probability on Wednesday is:', cdprob)
plt.scatter(df['Day'], df['Chg%'])
plt.xlabel('Day')
plt.ylabel('Chg%')
plt.title('Chg% vs Day')
plt.show()
```

```
The Mean is: 1560.6634538152612
The Variance is: 58496.49239931618
Sample mean is: 1550.7060000000001
April mean is: 1698.9526315789474
Probability of making a loss: 0.5020080321285141
Wednesday profit probability is: 0.42
Conditional Probability on Wednesday is: 2.0916
```