20MCA241-DATASCIENCE

LABORATORYRECORD

Submittedinpartialfulfilmentoftherequirementsfortheawardof Masters of

Computer Applications

At

COLLEGEOFENGINEERINGPOONJAR

ManagedbyI.H.R.D.,AGovt.ofKerala undertaking

(AffiliatedtoAPJAbdulKalamTechnologicalUniversity)

SUBMITTEDBY

SANITHA K S(PJR24MCA-2016)

Department of Computer Applications

COLLEGEOFENGINEERINGPOONJAR

COLLEGEOFENGINEERINGPOONJAR

ManagedbyI.H.R.D.,AGovt.ofKerala undertaking

(AffiliatedtoAPJAbdulKalamTechnologicalUniversity)

CERTIFICATE

Certified that this is a Bonafide record of practical work done in Data ScienceLab (20MCA241) Laboratory by **SANITHA K S** Reg No. **PJR24MCA-2016** of College of Engineering, Poonjar during the academic year 2024- 2026.

Dr.AnnieJulieJoseph
HeadoftheDepartment

MaryTreesaThomas
Asst.Professor ,CSE

SubmittedtotheUniversityExaminationheldon:

INTERNALEXAMINER **EXTERNALEXAMINER**

INDEX

PROGRAM-1

AIM

Createanumpyarrayandperformthefollowingoperations

1. Appendvaluestotheendofanarray.
2. 2.Insertvaluesintoanarrayataspecifiedposition.
3. Deleteelementsfromanarray.
4. Finduniqueelementsinanarray.
5. Sortan array.

SOURCECODE

```
importnumpy asnp
array = np.array([3, 2, 1, 2])
print("Originalarray:",array)
print("Append(6,7,8):",np.append(array,[6,7, 8]))
print("InsertSpecific(10,11)atthirdsecondposition:",np.insert(array,2,[10,11])) print("Delete
values (1,3): ", np.delete(array, [0, 2]))
print("Uniqueelement:",np.unique(array))
print("Sorted array: ", np.sort(array))
```

OUTPUT

RESULT

Theprogramhasbeenexecuted successfullyand output isobtained.

PROGRAM-2

AIM

YouhavetwoNumPyarrays,arr1andarr2.WriteNumPycodetoperformthefollowing operations:

1. Addarr1andarr2tocreateanewarraycalledresult_add.
2. Multiplyarr1andarr2to createanewarraycalledresult_multiply.
3. Calculatethemeanof result_add.
4. Find themaximum valuein result_multiply.

SOURCECODE

```
importnumpy asnp
arr1=np.array([1,2,3,4, 5])
arr2 = np.array([6, 7, 8, 9, 10])
result_add = arr1 + arr2
print("Sumofarray:",result_add)
result_multiply = arr1 * arr2
print("Product of array: ", result_multiply)
print("Meanofarray:",np.mean(result_add))
print("Maxvalue:",np.max(result_multiply))
```

OUTPUT

RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-3

AIM

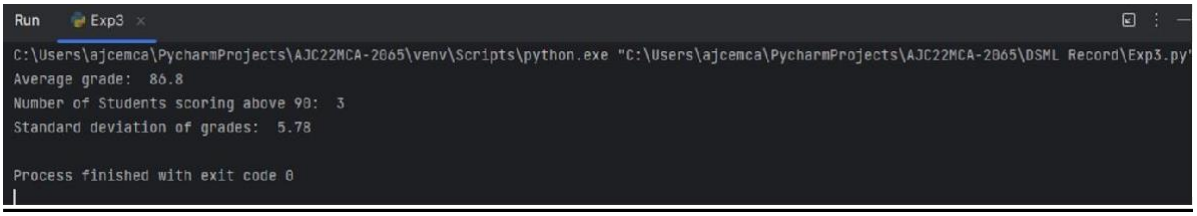
YouhaveaNumPyarraycalledgradesthatrepresentsthescoresofstudentsinaclass: Write NumPy code to answer the following questions:

1. Whatistheaverage (mean)gradeintheclass?
2. Howmanystudentsscoredabove90?
3. Calculatethestandard deviationofthegrades**.**

SOURCECODE

```
importnumpy asnp
grades=np.array([85,90, 78,92,88,76, 95,89,84, 91])
print("Averagegrade:",np.mean(grades))
filter_grade = grades[grades > 90]
print("NumberofStudentsscoringabove90:",len(filter_grade))
std_grade = np.std(grades)
print("Standarddeviationofgrades:",np.round(std_grade,decimals=2))
```

OUTPUT



RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-4

AIM

Performthefollowingmatrixoperations:

1. Addmatrix_Aandmatrix_B element-wisetocreateanewmatrix, matrix_sum.
2. Multiplymatrix_Aandmatrix_Belement-wisetocreateanewmatrix, matrix_product.
3. Calculatethematrixproductofmatrix_Aandmatrix_B(dotproduct)andstoreit in matrix_dot.
4. Transposematrix_Aandstoreitinmatrix_A_transpose.

Print the results of each operation.

SOURCECODE

```
importnumpy asnp
matrix_A=np.array([[1,2,3],[4,5,6],[7,8,9]])
matrix_B=np.array([[9,8,7],[6,5,4],[3,2,1]])
matrix_sum = matrix_A + matrix_B
print("Sum of matrices: \n", matrix_sum)
matrix_product = matrix_A * matrix_B
print("Productofmatrices:\n",matrix_product)
matrix_dot = np.dot(matrix_A, matrix_B)
print("Dot Product of matrices: \n",matrix_dot)
matrix_A_transpose = np.transpose(matrix_A)
print("TransposeofmatrixA:\n", matrix_A_transpose)
```

OUTPUT

RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-5

AIM

YouhaveaCSVfilenamed"sales_data.csv" containing salesdatawithcolumnsfor"Date", "Product","Quantity"and"Revenue"LoadthisdatausingPandasandanswerthefollowing questions:

1. Howmanyrowsandcolumnsarethereinthedataset?
2. What isthetotalrevenue forall thesales?

SOURCECODE

```
importpandasaspd
 df=pd.read_csv("sales_data.csv")
print(df.head())
print("Number of Rows:",len(df))
print("Numberofcolumns:",len(df.columns))
print("Total revenue:",sum(df['revenue']))
//sales_data.csvdate,product,quantity,revenue
2024-05-09,book,5,500
2024-07-12,pencil,5,10
2024-03-11,eraser,12,50
2024-08-23,scale,6,30
2024-11-23,pen,20,200
```

OUTPUT

```
/home/cep/PycharmProjects/pythonProject/.venv/bin/python /home/cep/Desktop/pythonProject/files.py
         date product  quantity  revenue
0  2024-05-09    book         5      500
1  2024-07-12  pencil         5       10
2  2024-03-11  eraser        12       50
3  2024-08-23   scale         6       30
4  2024-11-23     pen        20      200
Number of Rows: 5
Number of columns: 4
Total revenue: 790

Process finished with exit code 0
```

RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-6

AIM

YouhaveaDataFramecalled"student_data"withcolumns"Student_ID,""Name,""Age," and "GPA." Perform the following operations using Pandas:

1. Filteranddisplaytherowsofstudentswhoare20yearsoldor older.
2. CalculatetheaverageGPAofthestudentsintheDataFrame.
3. SorttheDataFrameindescendingorderofGPAanddisplaythetop5students with the highest GPAs.
4. Groupthestudentsbytheiragesandcalculatetheaverage GPAforeachage group.

SOURCECODE

```
importpandasaspd
df=pd.read_csv("students_data.csv")
print(df)
print(“noofstudentswhoare20yearsoldorolder”)
 age_df=df[df['age']>=20]
print(age_df)
avggpa=df['GPA'].mean()
print("Average GPA: ")
print(avggpa)
sorted_df=df.sort_values(by='GPA',ascending=False)
print("descending order of GPA: ")
print(sorted_df)
top_five=sorted_df.head(5)
print("Top 5 students with highest GPA: ")
print(top_five)
avg_grp=df.groupby('age')['GPA'].mean().reset_index()
print("Average GPA of Each age group : ")
print(avg_grp)
//student_data.csvstudent_id,name,age,GPA
1001,Amal,23,9.8
```

1002,athul,18,8.8

1003,mathew,20,7.9

1004,thoma,21,8.9

1005,aby,19,6.8

1006,john,20,6.9

1007,robin,22,9.5

OUTPUT

```
/home/cep/PycharmProjects/pythonProject/.venv/bin/python /home/cep/Desktop/pythonProject/p6_student.py
   student_id    name  age  GPA
0        1001    Amal   23  9.8
1        1002   athul   18  8.8
2        1003  mathew   20  7.9
3        1004   thoma   21  8.9
4        1005     aby   19  6.8
5        1006    john   20  6.9
6        1007   robin   22  9.5
no of students who are 20 years old or older:
   student_id    name  age  GPA
0        1001    Amal   23  9.8
2        1003  mathew   20  7.9
3        1004   thoma   21  8.9
5        1006    john   20  6.9
6        1007   robin   22  9.5
Average GPA:
8.37142857142857
descending order of GPA:
   student_id    name  age  GPA
0        1001    Amal   23  9.8
6        1007   robin   22  9.5
3        1004   thoma   21  8.9
1        1002   athul   18  8.8
2        1003  mathew   20  7.9
5        1006    john   20  6.9
4        1005     aby   19  6.8
```

```
Top 5 students with highest GPA:
   student_id    name  age  GPA
0        1001    Amal   23  9.8
6        1007   robin   22  9.5
3        1004   thoma   21  8.9
1        1002   athul   18  8.8
2        1003  mathew   20  7.9
Average GPA of Each age group :
   age  GPA
0   18  8.8
1   19  6.8
2   20  7.4
3   21  8.9
4   22  9.5
5   23  9.8

Process finished with exit code 0
```

RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-7

AIM

Createscatterplotforthebelowdata:(useScatter function)

Product	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Affordable Segment	173	153	195	147	120	144	148	109	174	130	172	131
Luxuary Segment	189	189	105	112	173	109	151	197	174	145	177	161
Super Luxury Segment	185	185	126	134	196	153	112	138	200	145	167	110

CreatescatterplotforeachSegmentwithfollowingpropertieswithinone graph

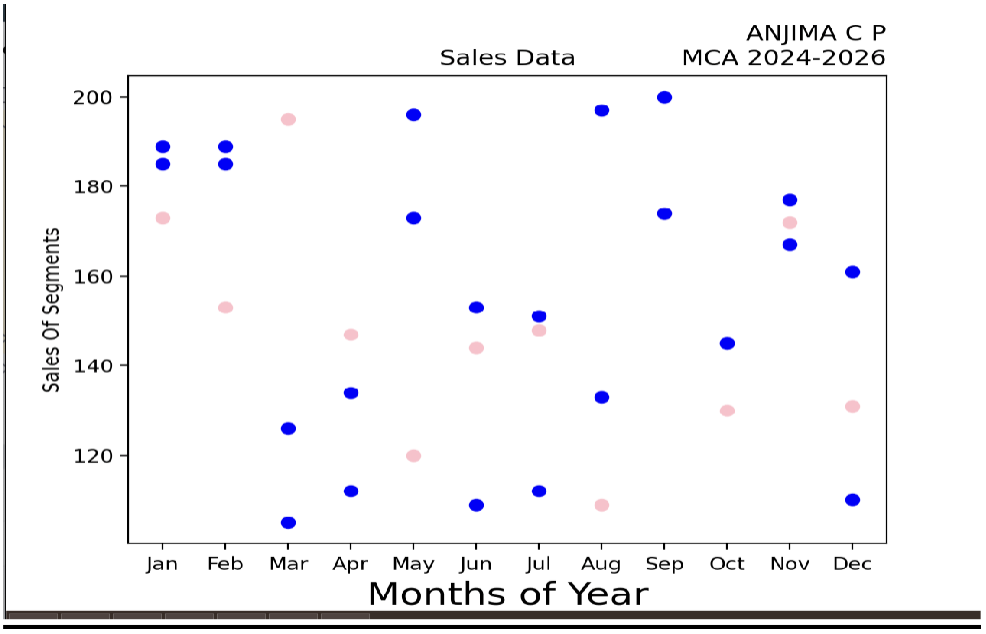
- XLabel-Month ofYear with fontsize18
- YLabel-SalesofSegments
- Title-SalesData
- ColorforAffordablesegment-pink
- ColorForLuxurysegment

SOURCECODE

```
importmatplotlib.pyplotasplt
import numpy as np
month=np.array(['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec'])
AS=np.array([173,153,195,147,120,144,148,109,174,130,172,131])
LS=np.array([189,189,105,112,173,109,151,197,174,145,177,161])
SLS=np.array([185,185,126,134,196,153,112,133,200,145,167,110])
plt.xlabel("MonthsofYear",fontsize=18)
plt.ylabel('Sales Of Segments')
plt.title('Sales Data')
plt.title('ANJIMA C P\n MCA 2024-2026',loc='right')
plt.scatter(month,AS,label='Affordable Segment',color='pink')
plt.scatter(month,LS,label=' Luxury Segment',color='blue')
plt.scatter(month,SLS,label='SuperLuxurySegment',color='blue')
```

```
plt.savefig("months.png")
plt.show()
```

OUTPUT



RESULT

Theprogramhasbeenexecutedsuccessfullyandoutputis obtained.

PROGRAM–8

AIM

100 students wereasked what their primary mode of transport for getting toschool was.The resultsofthissurveyarerecordedinthetablebelow.Constructabargraph representingthis information.

Modeof Transport	Frequency
Walking	29
Cycling	15
Car	35
Bus	18
Train	3

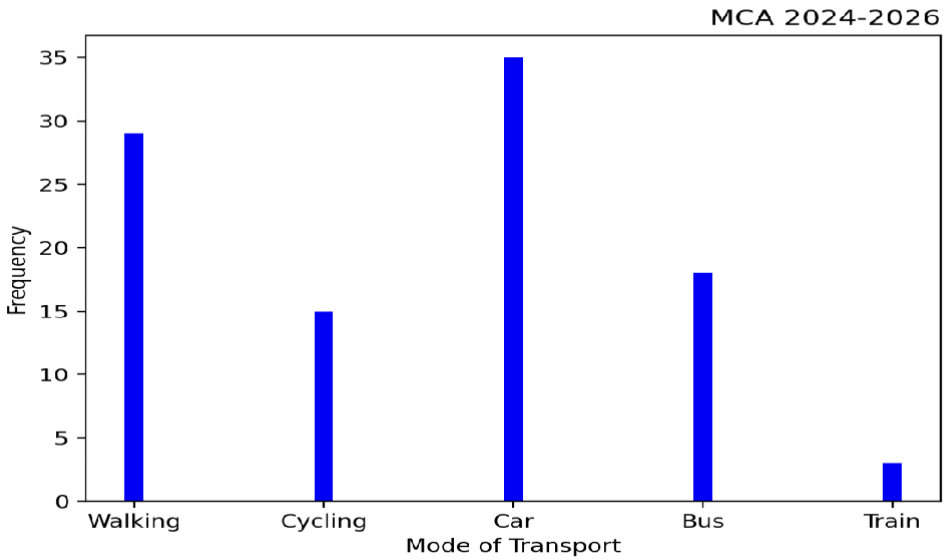
Createabargraph

- Xaxis-modeofTransportandYaxis-frequency
- Provideappropriatelabelsandtitle
- Width.1,colorblue

SOURCECODE

```
importmatplotlib.pyplotasplt
import numpy as np
mode_transport=np.array(["Walking",'Cycling','Car','Bus','Train'])
feq=np.array([29,15,35,18,3])
plt.xlabel('ModeofTransport')
plt.ylabel('Frequency')
plt.title('MCA 2024-2026',loc='right')
plt.bar(mode_transport,feq,width=0.1,color='blue')
plt.savefig("Transport.png")
plt.show()
```

OUTPUT



RESULT

Theprogramhasbeenexecutedsuccessfullyandoutputis obtained.

PROGRAM-9

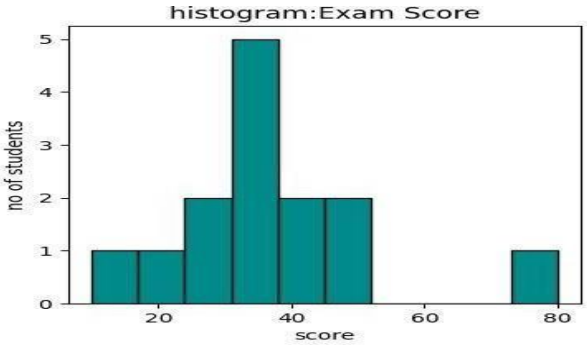
AIM

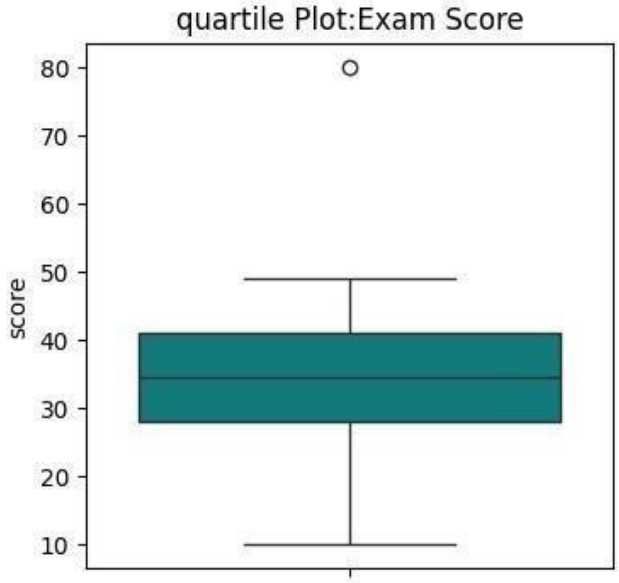
Use a dataset of your choice (e.g., exam scores of students, employee salaries, or any other numericaldata).Createahistogramtovisualizethedata'sdistribution.Afterward,plotquartiles (e.g., Q1, Q2, Q3) on the same graph.

SOURCECODE

```
importmatplotlib.pyplotasplt
import seaborn as sns
import numpy as np
marks=np.array([10,18,34,37,33,38,24,80,45,49,27,31,35,42])
fig,ax=plt.subplots(figsize=(4,4))
ax.hist(marks,color="darkcyan",ec="black",lw=1)
plt.title('histogram:Exam Score')
plt.ylabel('no of students')
plt.xlabel('score')
plt.savefig('histogram.png')
plt.figure(figsize=(4,4))
sns.boxplot(y=marks,color="darkcyan")
plt.title('quartile Plot:Exam Score')
plt.ylabel('score')
plt.savefig('quartile_plot.png')
```

OUTPUT





RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-10

AIM

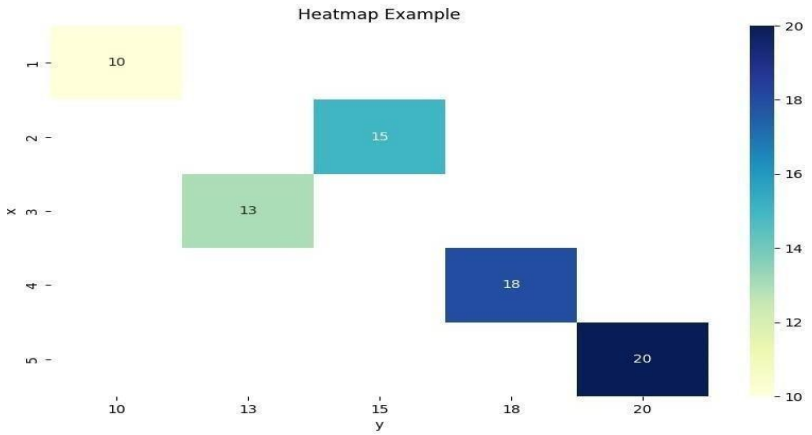
Choose a dataset that contains two numerical variables (e.g., income vs. education level, temperaturevs.icecreamsales).Create adistributionchartforeachvariableandascatterplot to visualize their relationship.

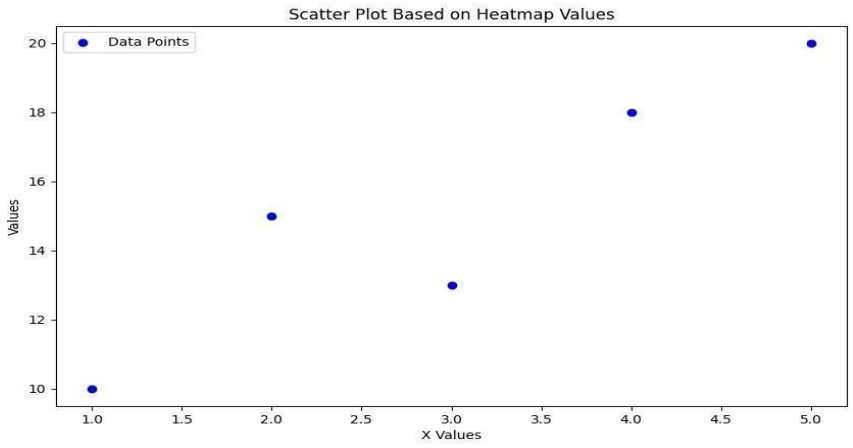
SOURCECODE

```
importmatplotlib
matplotlib.use('Agg')#Usenoninteractivebackend
import pandas as pd
importmatplotlib.pyplotasplt
import seaborn as sns
# Define the data
x_values=[1,2,3,4,5]
y_values=[10,15,13,18, 20]
data_values=[10,15,13,18,20]

#Create DataFrame
 df=pd.DataFrame({'x':x_values,'y':y_values,'value':data_values})

#Create apivot tablefor the heatmap
heatmap_data=df.pivot_table(index='x',columns='y', values='value')

# Plot theheatmap
fig,ax=plt.subplots(figsize=(10,6))
sns.heatmap(heatmap_data,annot=True,cmap='YlGnBu',cbar=True)
plt.title('Heatmap Example')
plt.savefig('heatmap_example.png')#Savetofile
plt.close()


#ScatterPlot
```

```
plt.figure(figsize=(10,6))
plt.scatter(df['x'],df['value'],marker='o',color='blue',label='DataPoints')
plt.xlabel('X Values')
plt.ylabel('Values')
plt.title('ScatterPlotBasedonHeatmapValues')
plt.legend()
plt.savefig('scatter_plot_example.png')#Savetofile
plt.close()
```

OUTPUT





RESULT

Theprogramhasbeenexecutedsuccessfullyandoutputisobtained

PROGRAM-11

AIM

Selectadatasetwithatleastthreenumericalvariables(e.g.,population,income,andeducation levelbycity).Createabubblechartthatrepresentsthedatabyusingbubblesizesandcolorsto encode information. Additionally, create a density chart (e.g., a 2D density plot) to show the concentration of data points.

SOURCECODE

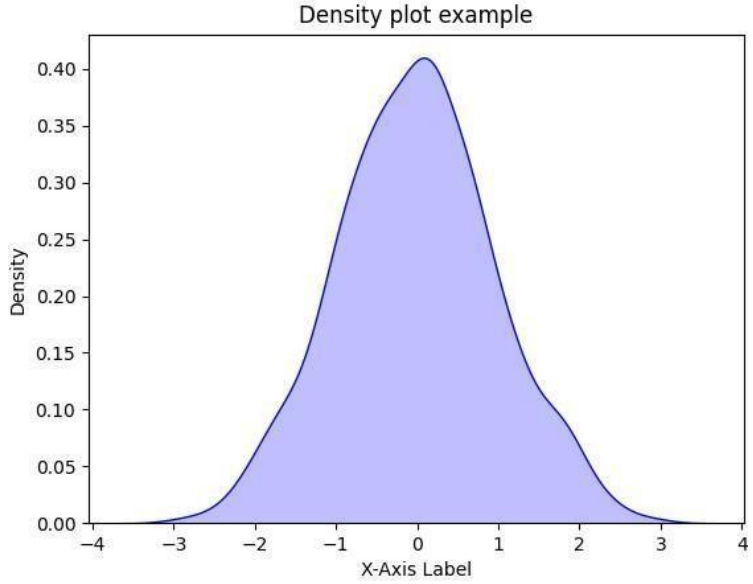
//DensityChart.py

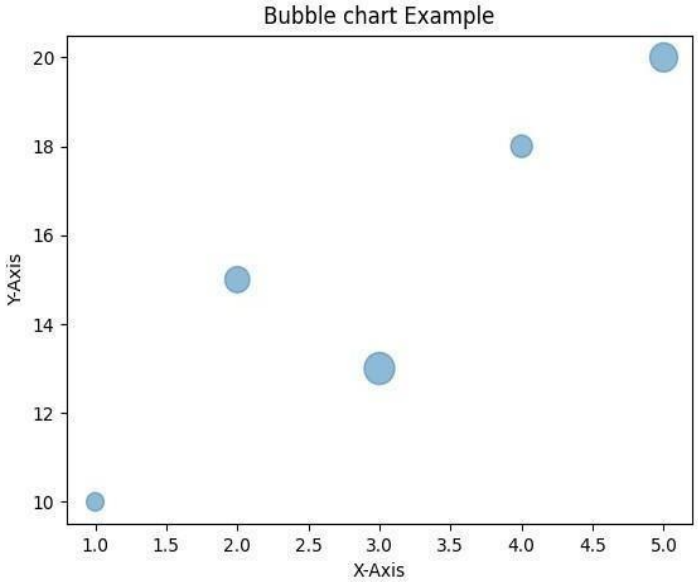
```
importseabornassns
importmatplotlib.pyplotasplt
import numpy as np
data = np.random.randn(1000)
sns.kdeplot(data,fill=True,color='blue',label='DensityPlot')
plt.xlabel('X-Axis Label')
plt.ylabel('Density')
plt.title('Densityplotexample')
plt.savefig('Density_plot.png')
```

//BubbleDiagram.pyimport

```
matplotlib.pyplot as plt
x =[1,2,3,4,5]
y =[10,15,13,18,20]
sizes=[100,200,300,150,250]
plt.scatter(x,y,s=sizes,alpha=0.5)
plt.xlabel('X-Axis')
plt.ylabel('Y-Axis')
plt.title('BubblechartExample')
plt.savefig('bubblechart.png')
```

OUTPUT





RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-12

AIM

ProgramtoimplementK-NearestNeighbourClassificationandfindtheaccuracyof algorithm.

SOURCECODE

```
importpandasaspd
fromsklearn.model_selectionimporttrain_test_split
fromsklearn.neighborsimportKNeighborsClassifier
fromsklearn.metricsimportaccuracy_score,classification_report
data = pd.read_csv("iris.csv")
X=data.drop("species",axis=1)y =data["species"]
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=42)
knn_classifier = KNeighborsClassifier(n_neighbors=3)
knn_classifier.fit(X_train,y_train)
y_pred = knn_classifier.predict(X_test)
accuracy = accuracy_score(y_test,y_pred)
print("Accuracy:",accuracy)
report=classification_report(y_test,y_pred,target_names=data["species"].unique())
print("Classification Report:\n",report)
```

OUTPUT

```
/home/cep/PycharmProjects/pythonProject/.venv/bin/python /home/cep/Desktop/pythonProject/Knearest.py
Accuracy: 1.0
Classification Report:
              precision    recall  f1-score   support

      Setosa       1.00      1.00      1.00        19
  Versicolor       1.00      1.00      1.00        13
   Virginica       1.00      1.00      1.00        13

    accuracy                           1.00        45
   macro avg       1.00      1.00      1.00        45
weighted avg       1.00      1.00      1.00        45


Process finished with exit code 0
```

RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-13

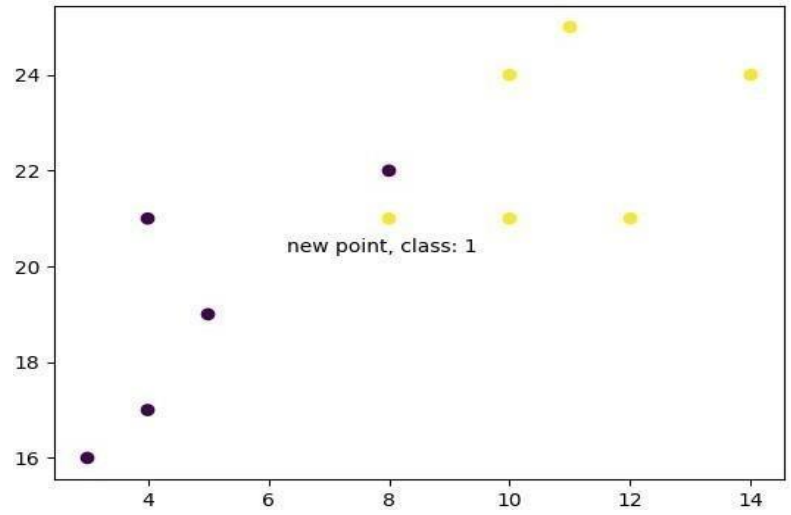
AIM

ProgramtodemonstratehowtheK-NearestNeighbors(KNN)algorithmdeterminestheclass of a new data point.

SOURCECODE

```
importmatplotlib.pyplotasplt
fromsklearn.neighborsimportKNeighborsClassifier
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))
knn=KNeighborsClassifier(n_neighbors=5)
knn.fit(data, classes) new_x = 8 new_y = 21
new_point = [(new_x, new_y)]
prediction= knn.predict(new_point)
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"newpoint,class:{prediction[0]}")
plt.savefig("knn.png")
```

OUTPUT



RESULT

Theprogramhasbeenexecutedsuccessfullyandoutputis obtained.

PROGRAM-14

AIM

ProgramtoimplementNaiveBayesClassifier.

SOURCECODE

```
fromsklearn.datasets
importload_iris
iris=load_iris()
X=iris.data
y=iris.target
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.4,random_state=)
from sklearn.naive_bayes import GaussianNB
gnb= GaussianNB()
gnb.fit(X_train,y_train)
y_pred=gnb.predict(X_test)
fromsklearnimportmetrics
print("GaussNaivebayesmodel accuracy(in%):",metrics.accuracy_score(y_test,y_pred)*100)
```

OUTPUT

```
/home/cep/PycharmProjects/pythonProject/.venv/bin/python /home/cep/Desktop/pythonProject/naivebayes.py
GaussNaive bayes model accuracy(in %): 95.0

Process finished with exit code 0
```

RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-15

AIM

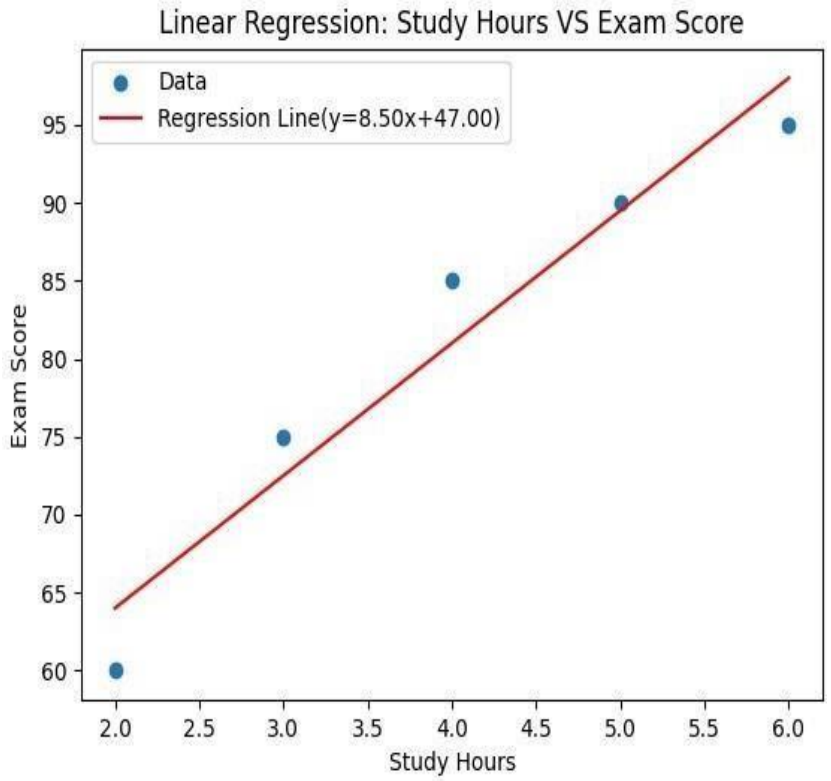
Programtoimplement SimpleLinearRegressionandfindr2score.

SOURCECODE

```
importnumpy asnp
fromfontTools.subsetimportintersect
from sklearn.feature_selection import f_regression
fromsklearn.linear_modelimportLinearRegression
import matplotlib.pyplot as plt
X=np.array([2,3,4,5,6]).reshape(-1,1)
Y=np.array([60,75,85,90,95])
model=LinearRegression()
model.fit(X,Y)
Y_pred=model.predict(X)
slope=model.coef_[0]
intercept=model.intercept_
plt.scatter(X,Y,label='Data')
plt.plot(X,Y_pred,color='red',label=f'Regression
Line(y={slope:.2f}x+{intercept:.2f})')
plt.xlabel('Study Hours')
plt.ylabel('ExamScore')
plt.legend()
plt.title('LinearRegression:StudyHoursVSExamScore')
plt.savefig('Simple_Linear_Regression.png')
new_SH=int(input(''Enter the number of hours:''))
pred_Sc=model.predict(np.array([[new_SH]]))
print(f'PredictedExamScorefor{new_SH}studyhours:{pred_Sc[0]:.2f}')
```

OUTPUT

```
/home/cep/PycharmProjects/pythonProject/.venv/bin/python /home/cep/Desktop/pythonProject/simpleLR.py
Enter the number of hours:8
Predicted Exam Score for 8 study hours: 115.00

Process finished with exit code 0
```



RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-16

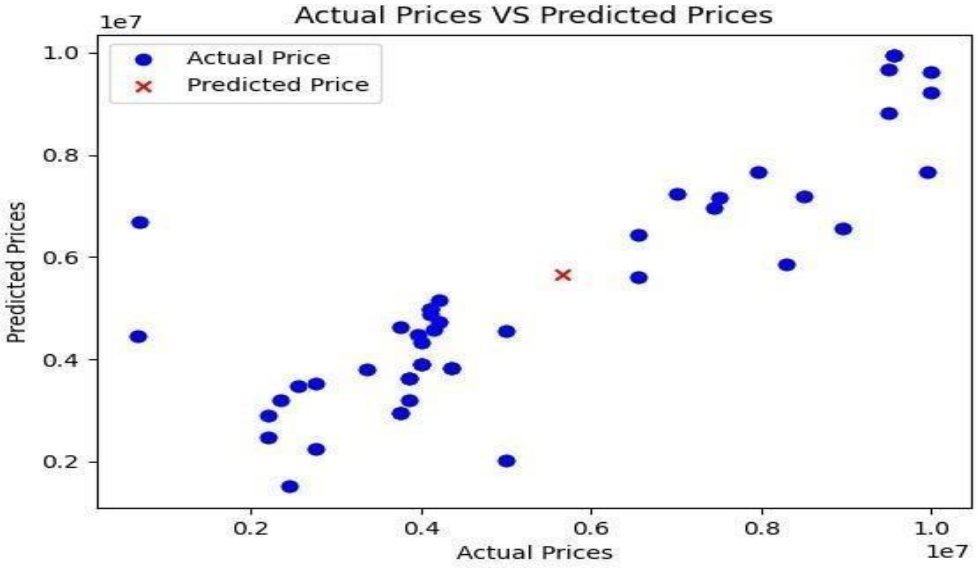
AIM

ProgramtoimplementMultipleLinearRegressionandevaluateitsperformance.

SOURCECODE

```
importpandasaspd
fromsklearn.linear_modelimportLinearRegression
import matplotlib.pyplot as plt
data=pd.read_csv("house-price.csv")
X=data[['No_Rooms','Sq_Foot','Age']].values
Y=data['Price'].valuesmodel=LinearRegression()
model.fit(X,Y)
Y_pred=model.predict(X)
plt.scatter(Y,Y_pred,color='blue')
plt.xlabel('Actual Prices')
plt.ylabel('Predicted Prices')
plt.title('ActualPricesVSPredictedPrices')
input_values=[int(input("Enterthenumberofrooms:")),
int(input("Enterthesquarefootage),int(input("EntertheAge:"))]
predicted_price=model.predict([input_values])
print("Predicted Price: ",predicted_price[0])
plt.scatter(Y,Y_pred,color='blue',label='Actual Price')
plt.scatter(predicted_price,predicted_price,color='red',marker='x',label='PredictedPrie)
plt.legend()
plt.savefig("MultipleLR.png")
```

OUTPUT



RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-17

AIM

ProgramtoimplementDecisionTreeClassification.

SOURCECODE

```
import
sysimportmatplot
lib
matplotlib.use('Agg')
import pandas as pd
fromsklearnimportDecisionTreeClassifier,plot_tree
import matplotlib.pyplot as plt

#Loadthedataset
df=pd.read_csv(‘dec_csv’)
#PrintinitialDataFrameinfo
print("Initial DataFrame:")
print(df.info())
print(df.head())

#Mapcategoricalvariablestonumericalvalues
nationality_map = {'UK': 0, 'USA': 1, 'N': 2}
df['Nationality'] =
df['Nationality'].map(nationality_map)
d={'YES':1,'NO': 0}df['Go']=df['Go'].map(d)

# Check for NaN values in the target variable
print("CheckingforNaNvaluesin'Go'column...")
print(df['Go'].isnull().sum())

ifdf['Go'].isnull().any():
   print("NaNvaluesfoundin'Go'column.DroppingrowswithNaNvalues.") df =
df.dropna(subset=['Go'])
```

```
print("DataFrameafterdroppingNaNvalues:")
print(df.info())
print(df.head())

#Checkifthere'senoughdata if
df.shape[0] == 0:
   print("Nodataavailablefortraining.Pleasecheckyourdataset.")
sys.exit(1)

# Define features and target variable
features=['Age','Experience','Rank','Nationality']
X = df[features]
 y = df['Go']
dtree=DecisionTreeClassifier()
dtree.fit(X, y)

# Plot the decision tree plt.figure(figsize=(10, 8))
plot_tree(dtree,feature_names=features,filled=Tru)
plt.title('Decision Tree Visualization')
plt.savefig('decision_tree.png')
plt.close()
sys.stdout.flush()

//decision_data.csv
"Age","Experience","Rank","Nationality","Go"
36,10,9,UK,NO
42,12,4,USA,NO
23,4,6,N,NO
52,4,4,USA,NO
43,21,8,USA,YES
44,14,5,UK,NO
66,3,7,N,YES
35,14,9,UK,YES
52,13,7,N,YES
```

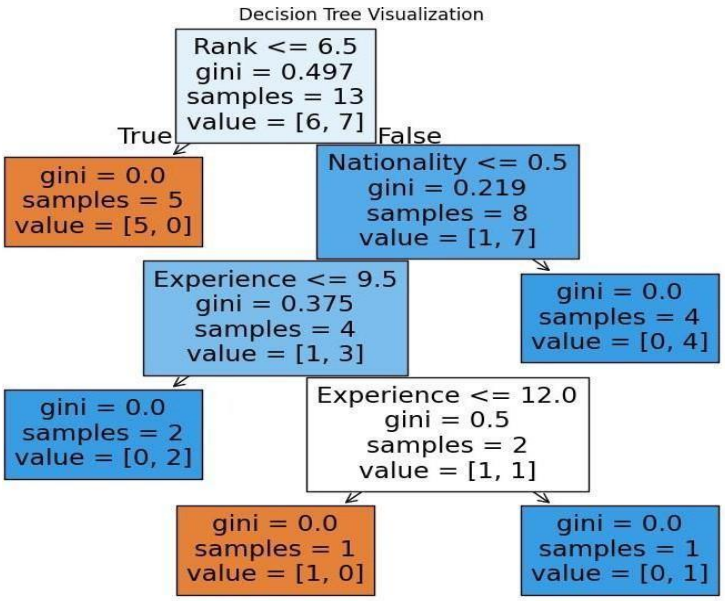
35,5,9,N,YES

24,3,5,USA,NO

18,3,7,UK,YES

45,9,9,UK,YES

OUTPUT



RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM–18

AIM

ProgramtoimplementK-MeansClassification.

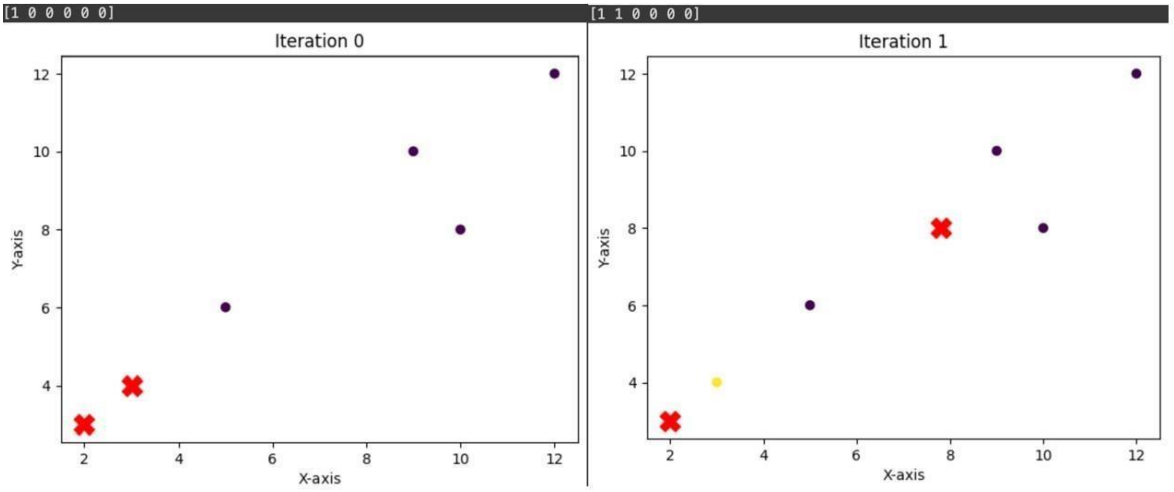
SOURCECODE

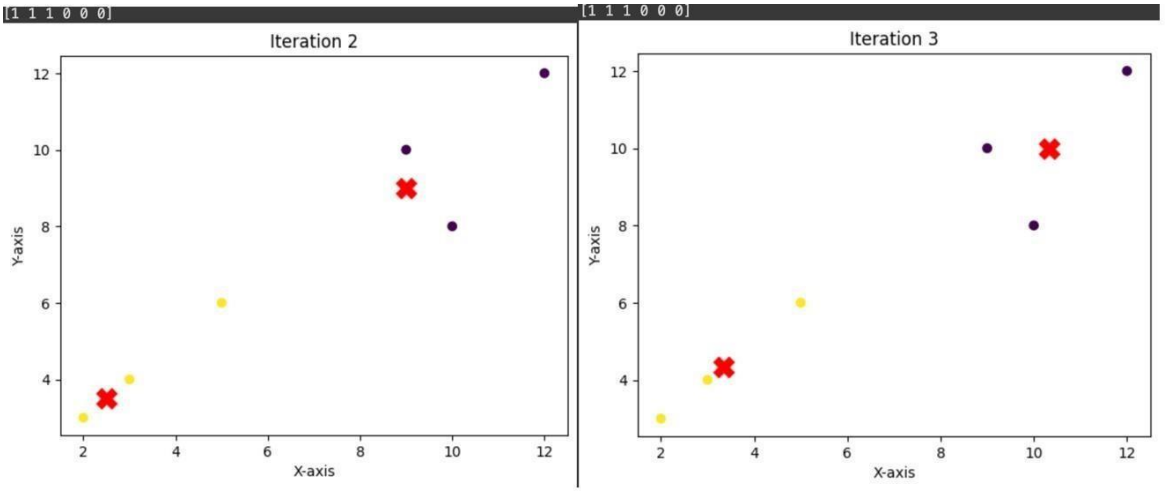
```
importnumpy asnp
importmatplotlib.pyplotasplt
data=np.array([[2,3],[3,4],[5,6],[9,10],[10,8],[12,12]])
k=2
  centroids=data[np.random.choice(data.shape[0],k,replace=
  False)]
num_iterations=100
for iteration in range(num_iterations):
distances=np.linalg.norm(data[:,np.newaxis]-centroids,axis=2)
labels = np.argmin(distances, axis=1)
print(labels)
plt.scatter(data[:,0],data[:,1],c=labels)
plt.scatter(centroids[:,0],centroids[:,1],c='red',marker='X',s=200)
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title(f"Iteration{iteration}")
plt.show()
new_centroids=np.array([data[labels==i].mean(axis=0) for
i in range(k)])
ifnp.array_equal(centroids,new_centroids):
break
centroids=new_centroids
print(distances)
print("FinalClusterAssignments:",labels)
```

print("FinalClusterCentroids:",centroids)

OUTPUT

```
[[10.88321848  1.88561808]
 [ 9.4751136   0.47140452]
 [ 6.66666667  2.3570226 ]
 [ 1.33333333  8.01387685]
 [ 2.02758751  7.60847481]
 [ 2.60341656 11.57103664]]
Final Cluster Assignments: [1 1 1 0 0 0]
Final Cluster Centroids: [[10.33333333 10.        ]
 [ 3.33333333  4.33333333]]
```





RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-19

AIM

ProgramtoimplementPart-Of-Speech(POST)Tagging.

SOURCECODE

importnltkfromnltk import

pos_tag

fromnltk.tokenizeimportword_tokenize

sentence="part-of-speechtaggingisimportantfornaturallanguageprocessing" words

= word_tokenize(sentence)

pos_tags=pos_tag(words)

print (pos_tags)

OUTPUT

```
/home/cep/PycharmProjects/pythonProject/.venv/bin/python /home/cep/Desktop/pythonProject/p15_POSTtagging.py
[('part-of-speech', 'JJ'), ('tagging', 'NN'), ('is', 'VBZ'), ('important', 'JJ'), ('for', 'IN'), ('natural', 'JJ'), ('language', 'NN'), ('processing', 'NN')]

Process finished with exit code 0
```

RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-20

AIM

ProgramtoimplementN-GramModeling.

SOURCECODE

```
from nltk import bigrams,word_tokenize
sentence="Naturallanguageprocessingisfascinating."
words = word_tokenize(sentence)
bigrams_list=list(bigrams(words))
print (bigrams_list)
```

OUTPUT

```
/home/cep/PycharmProjects/pythonProject/.venv/bin/python /home/cep/Desktop/pythonProject/P16_NGram.py
[('Natural', 'language'), ('language', 'processing'), ('processing', 'is'), ('is', 'fascinating'), ('fascinating', '.')]

Process finished with exit code 0
```

RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM–21

AIM

ProgramtoimplementaFeed ForwardNetworkusingIrisDataSet.

SOURCECODE

```
fromscipy.signalimportimpulse
fromsklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler,
LabelEncoder
fromsklearn.datasetsimportload_iris
fromtensorflow.kerasimportmodels,layers
fromKnearestimportX_train,X_test,y_train,y_test
iris=load_iris()
X=iris.data
y=iris.target
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=42)
scaler=StandardScaler()
X_train=scaler.fit_transform(X_train)
X_test=scaler.transform(X_test)
label_encoder=LabelEncoder()
y_train=label_encoder.fit_transform(train)
y_test=label_encoder.transform(y_test)
model= models.Sequential()
model.add(layers.Dense(64,activation='relu',input_shape=(4,)))
model.add(layers.Dense(32,activation='relu'))
model.add(layers.Dense(3,activation='softmax'))
model.compile(optimizer='adam',loss='sparse_categorical_crossentropy'acy'])
model.fit(X_train,y_train,epochs=50,batch_size=8,validation_split(0.1)
test_loss,test_acc=model.evaluate(X_test,y_test)
print(f'Testaccuracy:{test_acc}')
```

OUTPUT

```
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9645 - loss: 0.0613 - val_accuracy: 0.9167 - val_loss: 0.3557
Epoch 37/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9922 - loss: 0.0458 - val_accuracy: 0.9167 - val_loss: 0.3319
Epoch 38/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9836 - loss: 0.0638 - val_accuracy: 0.9167 - val_loss: 0.3516
Epoch 39/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9588 - loss: 0.0805 - val_accuracy: 0.9167 - val_loss: 0.3342
Epoch 40/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9605 - loss: 0.0626 - val_accuracy: 0.9167 - val_loss: 0.3325
Epoch 41/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9561 - loss: 0.0700 - val_accuracy: 0.9167 - val_loss: 0.3370
Epoch 42/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9875 - loss: 0.0676 - val_accuracy: 0.9167 - val_loss: 0.3182
Epoch 43/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9814 - loss: 0.0536 - val_accuracy: 0.9167 - val_loss: 0.3574
Epoch 44/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9719 - loss: 0.0699 - val_accuracy: 0.9167 - val_loss: 0.3827
Epoch 45/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9796 - loss: 0.0570 - val_accuracy: 0.9167 - val_loss: 0.3512
Epoch 46/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9948 - loss: 0.0439 - val_accuracy: 0.9167 - val_loss: 0.3713
Epoch 47/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9838 - loss: 0.0576 - val_accuracy: 0.9167 - val_loss: 0.3596
Epoch 48/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9683 - loss: 0.0713 - val_accuracy: 0.9167 - val_loss: 0.3529
Epoch 49/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9788 - loss: 0.0555 - val_accuracy: 0.9167 - val_loss: 0.3783
Epoch 50/50
14/14 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9981 - loss: 0.0508 - val_accuracy: 0.9167 - val_loss: 0.3655
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - accuracy: 1.0000 - loss: 0.0399
Test accuracy: 1.0
```

RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.

PROGRAM-22

AIM

Forgiventext

- Perform word
- Sentencetokenization
- Removethestopwordsfromthegiventext
- Createn-grams

SOURCECODE

```
import nltk
nltk.download('punkt')
nltk.download('punkt_tab')
nltk.download('stopwords')
from nltk import ngrams
fromnltk.corpusimportstopwords
fromnltk.tokenizeimportsent_tokenize,word_tokenize
print("PJR24MCA-2005:ANJIMA C P ")
print("BATCH: MCA 2024-2026")
text1="ThedatagivensatisfiestherequirementsformodelgeneRation.ThisisusedinData Science
Lab"
print("SentenceTokenization")
print(sent_tokenize(text1))
print("Word Tokenization:")
print(word_tokenize(text1))
text=word_tokenize(text1)
text2=[wordforwordintextifwordnotinstopwords.words('english')]
print("Removing stop_words:")
print(text2)
print("")print("n
grams")
unigrams=ngrams(text2,3)
```

forgramsinunigrams:

print(grams)

OUTPUT

```
/Users/anjimacp/PycharmProjects/PythonProject5/.venv1/bin/python /Users/anjimacp/PycharmProjects/PythonProject5/ngrams.py
PJR24MCA-2005:ANJIMA C P
BATCH: MCA 2024-2026
Sentence Tokenization
['The data given satisfies the requirements for model generation.This is used in Data Science Lab']
Word Tokenization:
['The', 'data', 'given', 'satisfies', 'the', 'requirements', 'for', 'model', 'generation.This', 'is', 'used', 'in', 'Data', 'Science', 'Lab']
Removing stop_words:
['The', 'data', 'given', 'satisfies', 'requirements', 'model', 'generation.This', 'used', 'Data', 'Science', 'Lab']

n grams
('The', 'data', 'given')
('data', 'given', 'satisfies')
('given', 'satisfies', 'requirements')
('satisfies', 'requirements', 'model')
('requirements', 'model', 'generation.This')
('model', 'generation.This', 'used')
('generation.This', 'used', 'Data')
('used', 'Data', 'Science')
('Data', 'Science', 'Lab')
```

RESULT

Theprogramhasbeen executedsuccessfullyandoutputisobtained.