Estd.2001

Sri Indu
College of Engineering & Technology
UGC Autonomous Institution
Recognized under 2(f) & 12(B) of UGC Act 1956,
NAAC, Approved by AICTE &
Permanently Affiliated to JNTUH

NAAC
NATIONAL ASSESSMENT AND ACCREDITATION COUNCIL

MACHINE LEARNING(R22CSM3126)

LAB MANUAL

III Year IT –Semester I

DEPARTMENTOFINFORMATION TECHNOLOGY

ACADEMIC YEAR 2024-25

DEPARTMENT OF INFORMATIONTECHNOLOGY

HANDOUT- INDEX

S. No	Contents
1	Vision, Mission, PEOs, POs, PSOs & COs
2	Institution Academic Calendar
3	Department Academic Calendar
4	Syllabus Copy
i)	Index
ii)	Programs
iii)	Model Paper For Lab Internal-1
iv)	Model Paper For Lab Internal-2

SRI INDU COLLEGE OF ENGINEERING & TECHNOLOGY

B. TECH –INFORMATION TECHNOLOGY

INSTITUTION VISION

To be a premier Institution in Engineering & Technology and Management with competency, values and social consciousness.

INSTITUTION MISSION

IM1 Provide high quality academic programs, training activities and research facilities.

IM2 Promote Continuous Industry-Institute Interaction for Employability,Entrepreneurship, Leadership and Research aptitude among stakeholders.

IM3 Contribute to the Economical and technological development of the region, state and nation.

DEPARTMENT VISION

To be a recognized knowledge center in the field of Information Technology with self -motivated, employable engineers to society.

DEPARTMENT MISSION

The Department has following Missions:

DM1 To offer high quality student centric education in Information Technology.

DM2 To provide a conducive environment towards innovation and skills.

DM3 To involve in activities that provide social and professional solutions.

DM4 To impart training on emerging technologies namely cloud computing and IOT with involvement of stake holders.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO 1: Higher Studies: Graduates with an ability to apply knowledge of Basic sciences and programming skills in their career and higher education.

PEO 2: Lifelong Learning: Graduates with an ability to adopt new technologies for ever changing IT industry needs through Self-Study, Critical thinking and Problem solving skills.

PEO 3: Professional skills: Graduates will be ready to work in projects related to complex problems involving multi-disciplinary projects with effective analytical skills.

PEO 4: Engineering Citizenship: Graduates with an ability to communicate well and exhibit social, technical and ethical responsibility in process or product.

PROGRAM OUTCOMES (POs) & PROGRAM SPECIFIC OUTCOMES (PSOs)

PO	Description
PO 1	**Engineering Knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, andan engineering specialization to the solution of complex engineering problems.
PO 2	**Problem Analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO 3	**Design / development of Solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO 4	**Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO 5	**Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO 6	**The engineer and Society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO 7	**Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO 8	**Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice
PO 9	**Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO 10	**Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO 11	**Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO 12	**Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological Change.
Program Specific Outcomes	
PSO 1	**Software Development:** To apply the knowledge of Software Engineering, Data Communication, Web Technology and Operating Systems for building IOT and Cloud Computing applications.
PSO 2	**Industrial Skills Ability:** Design, develop and test software systems for world-wide network of computers to provide solutions to real world problems.
PSO 3	**Project implementation:** Analyze and recommend the appropriate IT Infrastructure required for the implementation of a project.

DEPARTMENT OF INFORMATION TECHNOLOGY

COURSE OUTCOMES (CO'S)

COURSE NAME: MACHINE LEARNING LAB

Course Name	Course outcomes
C31L2.1	Understand complexity of machine learning algorithms and their limitations
C31L2.2	Understand modern notions in data analysis-oriented computing
C31L2.3	Be capable of confidently applying common machine learning algorithms in practice and implementing their own
C31L2.4	Be capable of performing experiments in machine learning using real-world data

COURSE ARTICULATION MATRIX

CO	**PO1**	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**	**PSO3**
C31L2.1	2	2	3	2	-	-	-	-	2	-	-	-	-	2	-
C31L2.2	2	1	3	2	-	-	-	-	2	-	-	-	2	2	-
C31L2.3	2	2	3	1	-	-	-	-	2	-	-	-	2	2	-
C31L2.4	2	3	2	2	-	-	-	-	-	-	-	-	2	2	-
C31L1	**2**	**2**	**1.3**	**0.8**	**0.6**	-	-	-	0.7	-	-	-	**0.6**	**2**	**-**

SRI INDU COLLEGE OF ENGINEERING & TECHNOLOGY

(An Autonomous Institution under UGC, New Delhi)

B.Tech. - III Year – I Semester

L	T	P	C
0	**0**	**2**	**1**

(R22CSM3126) MACHINE LEARNING LAB

Course Objective: The objective of this lab is to get an overview of the various machine learning techniques and can able to demonstrate them using python.

Course Outcomes: After the completion of the course the student can able to:

- understand complexity of Machine Learning algorithms and their limitations;
- understand modern notions in data analysis-oriented computing;
- be capable of confidently applying common Machine Learning algorithms in practice and implementing their own;
- Be capable of performing experiments in Machine Learning using real-world data.

List of Experiments

1. The probability that it is Friday and that a student is absent is 3 %. Since there are 5 school days in a week, the probability that it is Friday is 20 %. What is the probability that a student is absent given that today is Friday? Apply Baye‟s rule in python to get the result. (Ans: 15%)
2. Extract the data from database using python
3. Implement k-nearest neighbours classification using python
4. Given the following data, which specify classifications for nine combinations of VAR1 and VAR2 predict a classification for a case where VAR1=0.906 and VAR2=0.606, using the result of k- means clustering with 3 means (i.e., 3 centroids)

VAR1	VAR2	CLASS
1.713	1.586	0
0.180	1.786	1
0.353	1.240	1
0.940	1.566	0
1.486	0.759	1
1.266	1.106	0
1.540	0.419	1
0.459	1.799	1
0.773	0.186	1

5. The following training examples map descriptions of individuals onto high, medium and low credit-worthiness.

medium skiing design single twenties no -> high Risk high
golf trading married forties yes -> low Risk low
speedway transport married thirties yes -> med Risk medium
football banking single thirties yes -> low Risk high
flying media married fifties yes -> high Risk low
football security single twenties no -> med Risk medium
golf media single thirties yes -> med Risk medium
golf transport married forties yes -> low Risk high

skiing banking single thirties yes -> high Risk low
golf unemployed married forties yes -> high Risk

Input attributes are (from left to right) income, recreation, job, status, age-group, home-owner. Find the unconditional probability of `golf' and the conditional probability of `single' given `med Risk' in the dataset?

6. Implement linear regression using python.
7. Implement Naïve Bayes theorem to classify the English text
8. Implement an algorithm to demonstrate the significance of genetic algorithm
9. Implement the finite words classification system using Back-propagation algorithm

GENERAL LABORATORY INSTRUCTIONS

1. Students are advised to come to the laboratory at least 5 minutes before (to the starting time), thosewho come after 5 minutes will not be allowed into the lab.
2. Plan your task properly much before to the commencement, come prepared to the lab with thesynopsis / program / experiment details.
3. Student should enter into the laboratory with:
   a) Laboratory observation notes with all the details (Problem statement, Aim, Algorithm, Procedure,Program, Expected Output, etc.,) filled in for the lab session.
   b)Laboratory Record updated up to the last session experiments and other utensils(if any) needed inthe lab.
   c) Proper Dress code and Identity card.
4. Sign in the laboratory login register, write the TIME-IN, and occupy the computer system allotted to you by the faculty.
5. Execute your task in the laboratory, and record the results / output in the lab observation note book,and get certified by the concerned faculty.
6. All the students should be polite and cooperative with the laboratory staff,must maintain thediscipline and decency in the laboratory.
7. Computer labs are established with sophisticated and high end branded systems, which should beutilized properly.
8. Students / Faculty must keep their mobile phones in SWITCHED OFF mode during the lab sessions. Misuse of the equipment, misbehaviors with the staff and systems etc., will attract severepunishment.
9. Students must take the permission of the faculty in case of any urgency to go out ; if anybody found loitering outside the lab / class without permission during working hours will be treated seriously andpunished appropriately.
10. Students should LOG OFF/ SHUT DOWN the computer system before he/she leaves the lab after completing the task (experiment) in all aspects. He/she must ensure the system / seat is kept properly.

Head of the Department

List of Experiments

S.No	Name Of The Experiment	No. of Class required	CO
1.	The probability that it is Friday and that a student is absent is 3 %. Since there are 5 school days in a week, the probability that it is Friday is 20 %. What is theprobability that a student is absent given that today is Friday? Apply Baye's rule in python to get the result. (Ans: 15%)	3	CO1
2.	Extract the data from database using python	3	CO1
3.	Implement k-nearest neighbours classification using python	3	CO1
4.	Given the following data, which specify classifications for nine combinations of VAR1 and VAR2 predict a classification for a case where VAR1=0.906 and VAR2=0.606, using the result of kmeans clustering with 3 means (i.e., 3 centroids) periments VAR1 VAR2 CLASS 1.713 1.586 0 0.180 1.786 1 0.353 1.240 1 0.940 1.566 0 1.486 0.759 1 1.266 1.106 0 1.540 0.419 1 0.459 1.799 1 0.773 0.186 1	3	CO1,CO2
5.	The following training examples map descriptions of individuals onto high, medium and low creditworthiness. medium skiing design single twenties no -> high Risk high golf trading married forties yes -> low Risk low speedway transport married thirties yes -> med Risk medium football banking single thirties yes -> low Risk high flying media married fifties yes -> high Risk low football security single twenties no -> med Risk medium golf media single thirties yes -> med Risk medium golf transport married forties yes -> low Risk high skiing banking single thirties yes -> high Risk low golf unemployed married forties yes -> high Risk Input attributes are (from left to right) income, recreation, job, status, age-group, home-owner. Find the unconditional probability of `golf' and the conditional probability of `single' given `med Risk' in the dataset?	3	CO1,C02

6.	Implement linear regression using python.	**3**	**C04**
7	Implement Naïve Bayes theorem to classify the English text	**3**	**C04**
8	Implement an algorithm to demonstrate the significance of genetic algorithm	**3**	**C03**
9	Implement the finite words classification system using Back-propagation algorithm	**3**	**C03**
2.ADDITIONAL PROGRAMS			
1.	Implement and demonstrate the FIND-S algorithm for finding the most specific hypothesis based on a given set of training data samples. Read the training data from a .CSV file.	**3**	**C01**
2.	Build an Artificial Neural Network by implementing the Backpropagation algorithm and test the same using appropriate data sets.	**3**	**C02**
3.	Write a program to implement *k*-Nearest Neighbour algorithm to classify the irisdata set. Print both correct and wrong predictions. Java/Python ML library classes can be used for this problem.	**3**	**C03**

LAB INCHARGE **HOD**

Experiment :1

1. The probability that it is Friday and that a student is absent is 3 %. Since there are 5 school days in a week, the probability that it is Friday is 20 %. What is theprobability that a student is absent given that today is Friday? Apply Baye’s rule in python to get the result. (Ans: 15%)

ALGORITHM:

Step 1: Calculate probability for each word in a text and filter the words which have a probability less than thresholdprobability. Words with probability less than threshold probability are irrelevant.
Step 2: Then for each word in the dictionary, create a probability of that word being in insincere questions and its probability in sincere questions. Then finding the conditional probability to use in naive Bayes classifier.
Step 3: Prediction using conditional probabilities.
Step 4: End.

PROGRAM:

```
PFIA=float(input(“Enter probability that it is Friday and that a student is absent=”))
PF=float(input(“ probability that it is Friday=”))
PABF=PFIA / PF
print(“probability that a student is absent given that today is Friday using conditional probabilities=”,PABF)
```

OUTPUT:

```
Enter probability that it is Friday and that a student is absent= 0.03probability that it is Friday= 0.2
probability that a student is absent given that today is Friday using conditional probabilities= 0.15
```

Viva questions:

1.What is Machine learning?
2.Define Supervised learning?
3.Define Unsupervised learning?
4.Define Semi supervised learning?
5.Define Reinforcement learning?

Experiment:2

2. Extract the data from database using python

ALGORITHM:

Step 1: Connect to MySQL from Python
Step 2: Define a SQL SELECT Query
Step 3: Get Cursor Object from Connection
Step 4: Execute the SELECT query using execute() method
Step 5: Extract all rows from a result
Step 6: Iterate each row
Step 7: Close the cursor object and database connection object
Step 8: End.

PROCEDURE

CREATING A DATABASE IN MYSQL AS FOLLOWS:

```
CREATE DATABASE myDB;SHOW DATABASES;
USE myDB
CREATE TABLE MyGuests (id INT, name VARCHAR(20), email VARCHAR(20));
SHOW TABLES;
INSERT INTO MyGuests (id,name,email) VALUES(1,"sairam","xyz@abc.com");
…
SELECT * FROM authors;
```

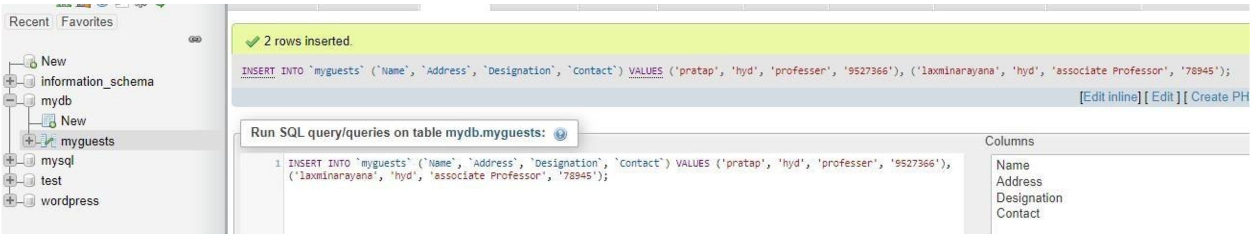
We need to install mysql-connector to connect Python with MySQL.
You can use the below command toinstall this in your system.

```
pip install mysql-connector-python-rf
```

PYTHON SOURCE CODE:

```
import mysql.connector

mydb = mysql.connector.connect(host="localhost",
user="root", password="", database="myDB")
    mycursor = mydb.cursor()
    mycursor.execute("SELECT * FROM
    MyGuests")

    myresult =

    mycursor.fetchall()for x

    in myresult:
         print(x)
```

OUTPUT:

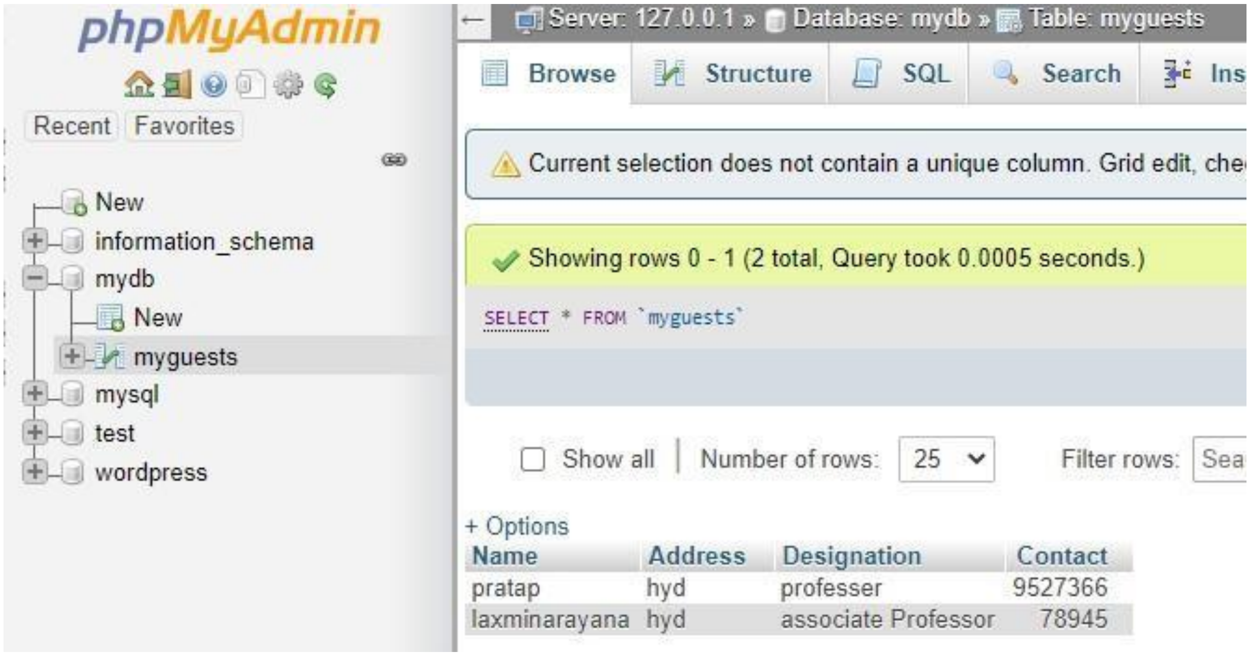


✓ 2 rows inserted.

```
INSERT INTO `myguests` (`Name`, `Address`, `Designation`, `Contact`) VALUES ('pratap', 'hyd', 'professer', '9527366'), ('laxminarayana', 'hyd', 'associate Professor', '78945');
```

Run SQL query/queries on table mydb.myguests:

```
INSERT INTO `myguests` (`Name`, `Address`, `Designation`, `Contact`) VALUES ('pratap', 'hyd', 'professer', '9527366'),
('laxminarayana', 'hyd', 'associate Professor', '78945');
```



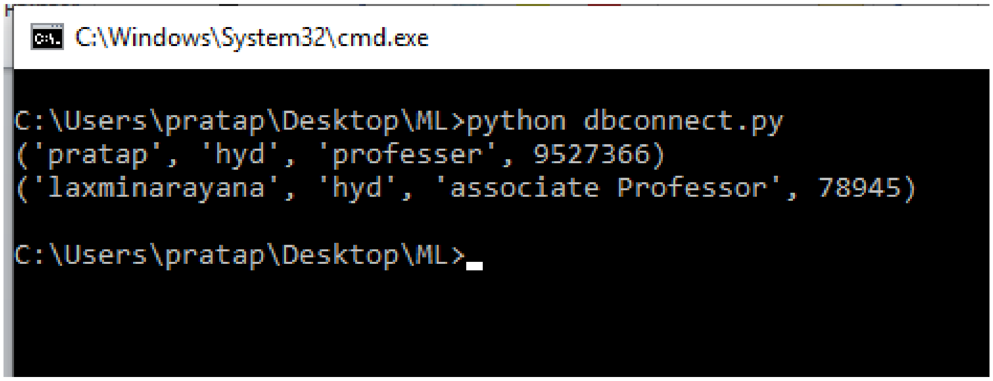
Current selection does not contain a unique column. Grid edit, che

✓ Showing rows 0 - 1 (2 total, Query took 0.0005 seconds.)

```
SELECT * FROM `myguests`
```

□ Show all | Number of rows: 25 Filter rows: Sea

Name	Address	Designation	Contact
pratap	hyd	professer	9527366
laxminarayana	hyd	associate Professor	78945



```
C:\Users\pratap\Desktop\ML>python dbconnect.py
('pratap', 'hyd', 'professer', 9527366)
('laxminarayana', 'hyd', 'associate Professor', 78945)

C:\Users\pratap\Desktop\ML>_
```

Extracting data from Excel sheet using Python

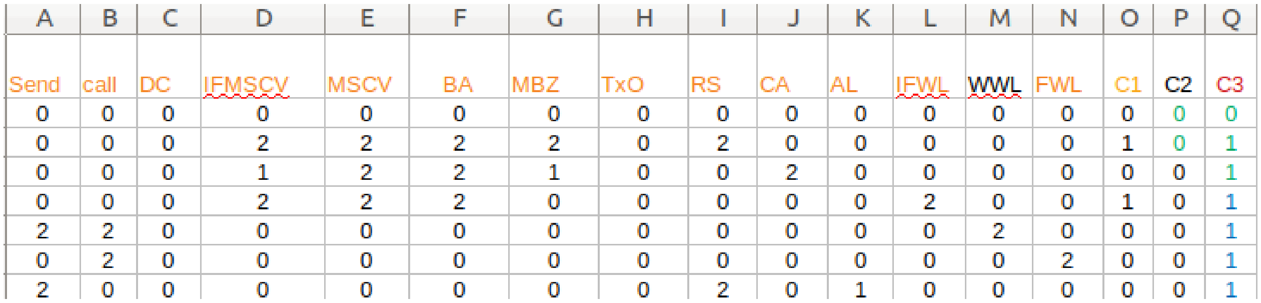
Step1: First convert dataset present in excel to CSV file using online resources, then execute following program:

consider dataset excel consists of 14 input columns and 3 output columns (C1, C2, C3)as follows:Python Souce Code:

```
import pandas as pd
dataset=pd.read_csv("Mul_Label_Dataset.csv",
delimiter=',') print(dataset)                    #Print entire
datasetX =
dataset[['Send','call','DC','IFMSCV','MSCV','BA','MBZ','TxO','RS','CA','AL','IFWL','WWL','FWL']].values
Y = dataset[['C1','C2','C3']].values
print(Y)                        #Prints output values
print(X)                        #Prints intput values
X1 = dataset[['Send','call','DC','IFMSCV','MSCV']].values
print(X1)                       #Prints first 5 columns of intput
values
print(X[0:5])                   # Prints only first 5 rows of input values
```

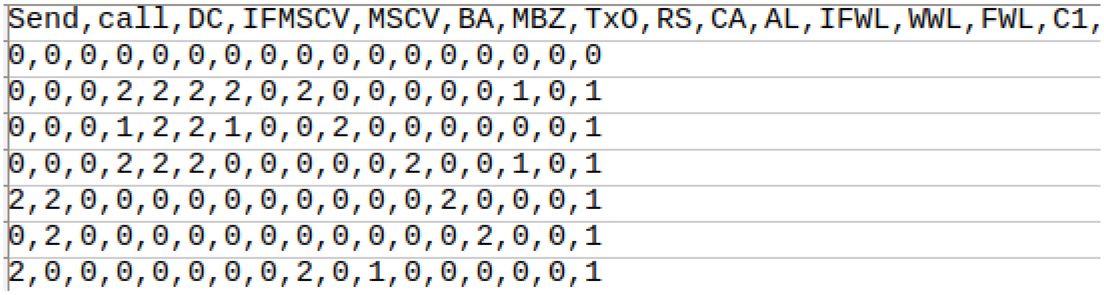
OUTPUT SCREENS:

Excel Format: CSV



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
Send	call	DC	IFMSCV	MSCV	BA	MBZ	TxO	RS	CA	AL	IFWL	WWL	FWL	C1	C2	C3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	2	2	2	2	0	2	0	0	0	0	0	1	0	1
0	0	0	1	2	2	1	0	0	2	0	0	0	0	0	0	1
0	0	0	2	2	2	0	0	0	0	0	2	0	0	1	0	1
2	2	0	0	0	0	0	0	0	0	0	0	2	0	0	0	1
0	2	0	0	0	0	0	0	0	0	0	0	0	2	0	0	1
2	0	0	0	0	0	0	0	2	0	1	0	0	0	0	0	1

Format:



```
Send,call,DC,IFMSCV,MSCV,BA,MBZ,TxO,RS,CA,AL,IFWL,WWL,FWL,C1,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0,0,0,2,2,2,2,0,2,0,0,0,0,0,1,0,1
0,0,0,1,2,2,1,0,0,2,0,0,0,0,0,0,1
0,0,0,2,2,2,0,0,0,0,0,2,0,0,1,0,1
2,2,0,0,0,0,0,0,0,0,0,0,2,0,0,0,1
0,2,0,0,0,0,0,0,0,0,0,0,0,2,0,0,1
2,0,0,0,0,0,0,0,2,0,1,0,0,0,0,0,1
```

Viva questions:

1. Define Entropy?
2. Define Regression?
3. How is KNN different from k-means clustering?
4. What is Concept Learning?
5.Specific Boundary & General Boundary?

Experiment:3

3.Implement k-nearest neighbours classification using python

ALGORITHM:

Step 1: Load the data
Step 2: Initialize the value of k
Step 3: For getting the predicted class, iterate from 1 to total number of training data points

i) Calculate the distance between test data and each row of training data. Here we will use Euclidean distance as our distance metric since it's the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.

ii) Sort the calculated distances in ascending order based on distance values 3. Get top k rows from thesorted array

iii) Get the most frequent class of these rows i.e. Get the labels of the selected K entries

iv) Return the predicted class If regression, return the mean of the K labels If classification, return the mode of the K labels

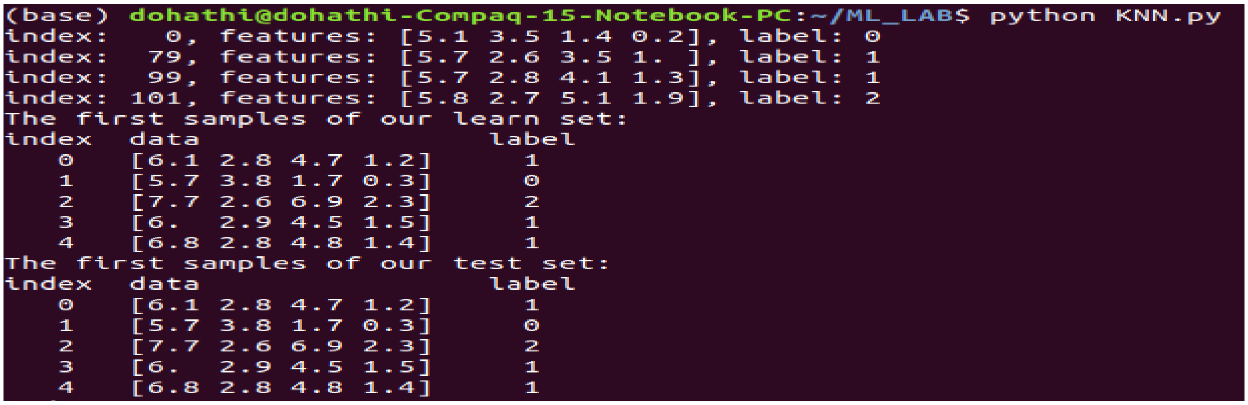
Step 4: End.

PROGRAM

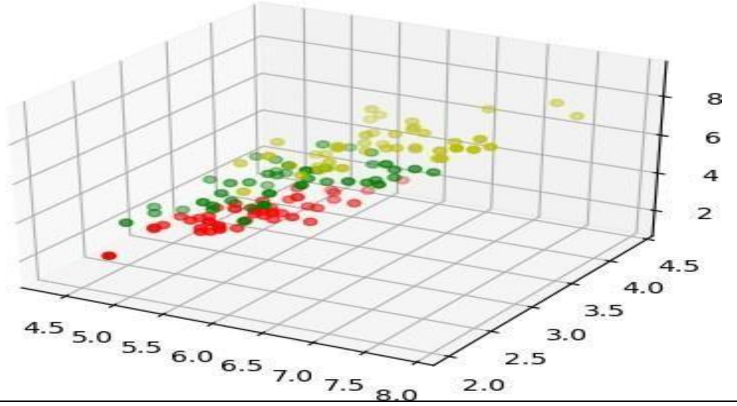
```
import numpy as np

from sklearn import datasets

iris = datasets.load_iris()data = iris.data
labels = iris.target

for i in [0, 79, 99, 101]:
print(f"index: {i:3}, features: {data[i]}, label: {labels[i]}")

np.random.seed(42)
indices = np.random.permutation(len(data))n_training_samples = 12
learn_data = data[indices[:-n_training_samples]] learn_labels = labels[indices[:-n_training_samples]]
test_data = data[indices[-n_training_samples:]] test_labels = labels[indices[-n_training_samples:]]
print("The first samples of our learn set:") print(f"{'index':7s}{'data':20s}{'label':3s}")for i in range(5):
print(f"{i:4d}  {learn_data[i]}  {learn_labels[i]:3}")

print("The first samples of our test set:") print(f"{'index':7s}{'data':20s}{'label':3s}")for i in range(5):
print(f"{i:4d}  {learn_data[i]}  {learn_labels[i]:3}")

#The following code is only necessary to visualize the data of our learnsetimport matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3Dcolours = ("r", "b")
X = []
for iclass in range(3):
X.append([[], [], []])
for i in range(len(learn_data)):if learn_labels[i] == iclass:
X[iclass][0].append(learn_data[i][0]) X[iclass][1].append(learn_data[i][1])
X[iclass][2].append(sum(learn_data[i][2:]))
```

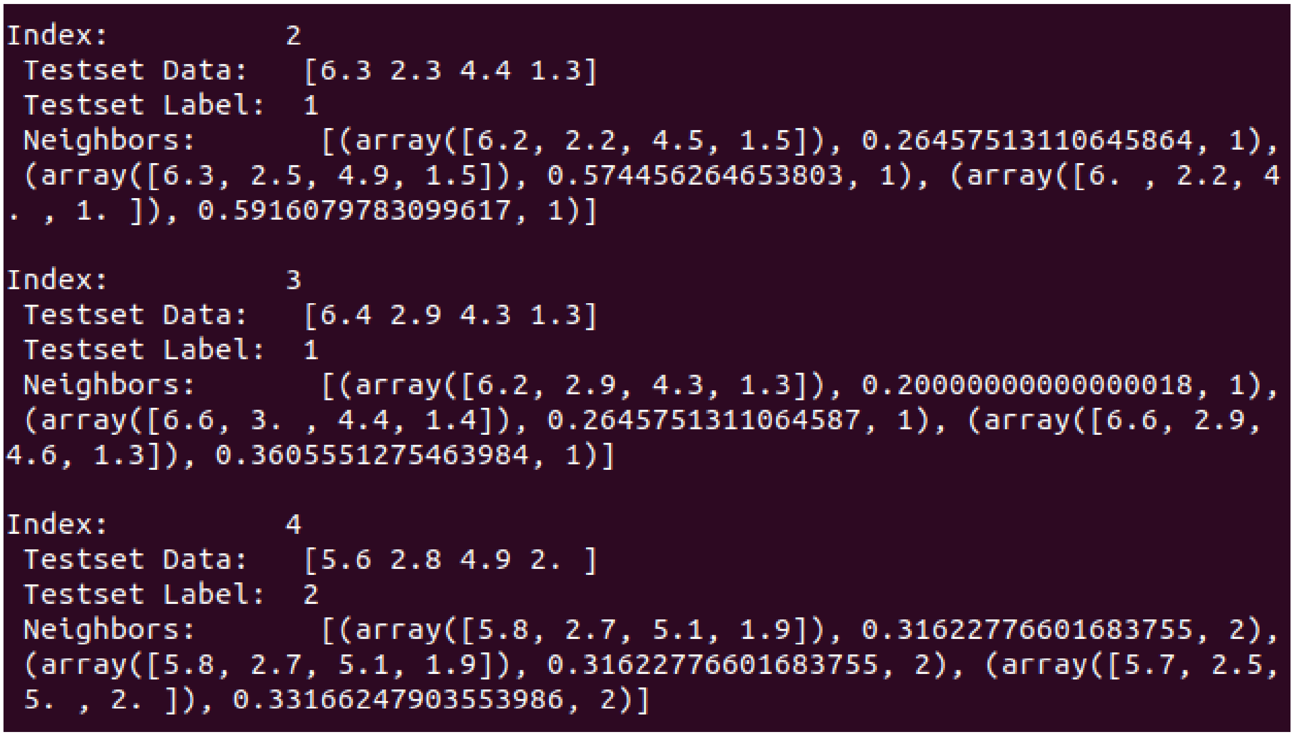
```
colours = ("r", g", "y")fig = plt.figure()
ax  =  fig.add_subplot(111,  projection='3d')for iclass in range(3):
ax.scatter(X[iclass][0], X[iclass][1], X[iclass][2], c=colours[iclass])plt.show()
#..........


def distance(instance1, instance2):
""" Calculates the Eucledian distance between two instances"""
return np.linalg.norm(np.subtract(instance1, instance2))

def get_neighbors(training_set, labels, test_instance, k, distance):
“””get_neighors calculates a list of the k nearest neighbors of an instance 'test_instance'.
The function returns a list of k 3-tuples. Each 3-tuples consists of (index, dist, label) """
distances = []
for index in range(len(training_set)):
dist = distance(test_instance, training_set[index]) d
istances.append((training_set[index],  dist,  labels[index]))
distances.sort(key=lambda x: x[1])neighbors = distances[:k]
return neighbors
for i in range(5):
neighbors = get_neighbors(learn_data, learn_labels, test_data[i], 3, distance=distance)
print("Index:      ",i,'\n',"Testset Data: ",test_data[i],'\n', "Testset Label: ",test_labels[i],'\n',"Neighbors:(",neighbors,'\n')
```

OUTPUT:



```
(base) dohathi@dohathi-Compaq-15-Notebook-PC:~/ML_LAB$ python KNN.py
index:    0, features: [5.1 3.5 1.4 0.2], label: 0
index:   79, features: [5.7 2.6 3.5 1. ], label: 1
index:   99, features: [5.7 2.8 4.1 1.3], label: 1
index:  101, features: [5.8 2.7 5.1 1.9], label: 2
The first samples of our learn set:
index   data                 label
   0    [6.1 2.8 4.7 1.2]      1
   1    [5.7 3.8 1.7 0.3]      0
   2    [7.7 2.6 6.9 2.3]      2
   3    [6.  2.9 4.5 1.5]      1
   4    [6.8 2.8 4.8 1.4]      1
The first samples of our test set:
index   data                 label
   0    [6.1 2.8 4.7 1.2]      1
   1    [5.7 3.8 1.7 0.3]      0
   2    [7.7 2.6 6.9 2.3]      2
   3    [6.  2.9 4.5 1.5]      1
   4    [6.8 2.8 4.8 1.4]      1
```





```
Index:          2
 Testset Data:   [6.3 2.3 4.4 1.3]
 Testset Label:  1
 Neighbors:       [(array([6.2, 2.2, 4.5, 1.5]), 0.26457513110645864, 1),
 (array([6.3, 2.5, 4.9, 1.5]), 0.574456264653803, 1), (array([6. , 2.2, 4
. , 1. ]), 0.5916079783099617, 1)]

Index:          3
 Testset Data:   [6.4 2.9 4.3 1.3]
 Testset Label:  1
 Neighbors:       [(array([6.2, 2.9, 4.3, 1.3]), 0.20000000000000018, 1),
 (array([6.6, 3. , 4.4, 1.4]), 0.2645751311064587, 1), (array([6.6, 2.9,
4.6, 1.3]), 0.3605551275463984, 1)]

Index:          4
 Testset Data:   [5.6 2.8 4.9 2. ]
 Testset Label:  2
 Neighbors:       [(array([5.8, 2.7, 5.1, 1.9]), 0.31622776601683755, 2),
 (array([5.8, 2.7, 5.1, 1.9]), 0.31622776601683755, 2), (array([5.7, 2.5,
 5. , 2. ]), 0.33166247903553986, 2)]
```

Viva questions:

1. Define regression?
2. How KNN is different from k-means clustering?
3. What is concept learning?
4. Define target function?
5. Define specific and generic boundary?

Experiment 4

4.Given the following data, which specify classifications for nine combinationsof VAR1 and VAR2 predict a classification for a case where VAR1=0.906 and VAR2=0.606, using the result of k- means clustering with 3 means (i.e., 3 centroids)

VAR1	VAR2	CLASS
1.713	1.586	0
0.180	1.786	1
0.353	1.240	1
0.940	1.566	0
1.486	0.759	1
1.266	1.106	0
1.540	0.419	1
0.459	1.799	1
0.773	0.186	1

Algorithm Steps

1. **Data Preparation**:
   - Create a dataset from the provided VAR1, VAR2, and CLASS values.
2. **K-means Clustering**:
   - Initialize the k-means algorithm with 3 clusters (centroids).
   - Fit the model to the dataset.
   - Retrieve the cluster centroids.
3. **Prediction**:
   - Predict the cluster for the new point (VAR1=0.906, VAR2=0.606).
   - Identify the class associated with the predicted cluster by examining the training examples that belong to that cluster.
4. **Output the Results**:
   - Print the cluster centroids.
   - Print the predicted cluster index.

PROGRAM:

```
import numpy as np

import pandas as pd

from sklearn.cluster import KMeans

import matplotlib.pyplot as plt
# Create a DataFrame with the provided data
data = {
   'VAR1': [1.713, 0.180, 0.353, 0.940, 1.486, 1.266, 1.540, 0.459, 0.773],
   'VAR2': [1.586, 1.786, 1.240, 1.566, 0.759, 1.106, 0.419, 1.799, 0.186],
   'CLASS': [0, 1, 1, 0, 1, 0, 1, 1, 1] }

df = pd.DataFrame(data)

    # Prepare the data for k-means (excluding
    the  CLASS column)

    X = df[['VAR1', 'VAR2']]
    # Initialize k-means with 3 clusters
kmeans = KMeans(n_clusters=3, random_state=42)
kmeans.fit(X)
# Get the centroids
centroids = kmeans.cluster_centers_
# Predict the classification for the new point
new_point = np.array([[0.906, 0.606]])
predicted_cluster = kmeans.predict(new_point)
# Find the class of the closest centroid based on the existing data
closest_centroid_index = predicted_cluster[0]
class_for_new_point = df[df['CLASS'] == closest_centroid_index].CLASS.mode()[0]

# Output results
print("Centroids:")
print(centroids)
print("Predicted Cluster Index:",
closest_centroid_index)
print("Predicted Class for new point (VAR1=0.906, VAR2=0.606):", class_for_new_point)
# Optional: Visualize the clusters
plt.scatter(df['VAR1'], df['VAR2'], c=df['CLASS'], label='Data Points')
plt.scatter(centroids[:, 0], centroids[:, 1], c='red', label='Centroids', marker='X', s=200)
plt.scatter(new_point[0][0], new_point[0][1], c='green', label='New Point', marker='o', s=100)
plt.title("K-Means Clustering")
plt.xlabel("VAR1")
plt.ylabel("VAR2")
plt.legend()
plt.show()
```

OUTPUT:

```
Centroids:
[[1.515 1.112]   # This is just an example; actual values will vary
 [0.385 1.252]
 [1.060 0.538]]
Predicted Cluster Index: 1
Predicted Class for new point (VAR1=0.906, VAR2=0.606): 1
```

Viva Questions

1.Define Decision Tree?
2.What is ANN?
3.Explain Gradient Descent approximation?
4.What is linear Regression?
5.What is the role of pandas library in this experiment ?

Experiment 5

The following training examples map descriptions of individuals ontohigh, medium and low credit-worthiness.

medium skiing design single twenties no -> high Risk high
golf trading married forties yes -> low Risk low
speedway transport married thirties yes -> med Risk medium
football banking single thirties yes -> low Risk high
flying media married fifties yes -> high Risk low
football security single twenties no -> med Risk medium
golf media single thirties yes -> med Risk medium
golf transport married forties yes -> low Risk high
skiing banking single thirties yes -> high Risk low
golf unemployed married forties yes -> high Risk

Input attributes are (from left to right) income, recreation, job, status, age-group, home-owner. Find the unconditional probability of `golf' and the conditional probability of `single' given `med Risk' in the dataset?

Algorithm Steps

1. **Data Preparation**:
   - Create a dataset with the provided attributes and risk classifications.
2. **Calculate Unconditional Probability of "golf"**:
   - Count the total number of entries in the dataset.
   - Count the number of entries where recreation is "golf".
   - Calculate the unconditional probability as the count of "golf" entries divided by the total number of entries.
3. **Calculate Conditional Probability of "single" given "med Risk"**:
   - Count the total number of entries with "med Risk".
   - Count the number of entries with both "med Risk" and "single".
   - Calculate the conditional probability as the count of "single" entries among "med Risk" entries divided by the total number of "med Risk" entries.
4. **Output the Results**:
   - Print the unconditional probability of "golf".
   - Print the conditional probability of "single" given "med Risk".

PROGRAM:

```
import pandas as pd

# Define the dataset
data = {
    'income': ['medium', 'high', 'low', 'medium', 'high', 'low', 'medium', 'medium', 'high', 'low'],
    'recreation': ['skiing', 'golf', 'speedway', 'football', 'flying', 'football', 'golf', 'golf', 'skiing', 'golf'],
    'job': ['design', 'trading', 'transport', 'banking', 'media', 'security', 'media', 'transport', 'banking', 'unemployed'],
    'status': ['single', 'married', 'married', 'single', 'married', 'single', 'single', 'married', 'single', 'married'],
    'age_group': ['twenties', 'forties', 'thirties', 'thirties', 'fifties', 'twenties', 'thirties', 'forties', 'thirties', 'forties'],
    'home_owner': ['no', 'yes', 'yes', 'yes', 'yes', 'no', 'yes', 'yes', 'yes', 'yes'],
    'risk': ['high Risk', 'low Risk', 'med Risk', 'low Risk', 'high Risk', 'med Risk', 'med Risk', 'low Risk', 'high Risk',
'high Risk']
}
# Create a DataFrame
df = pd.DataFrame(data)

# Calculate the unconditional probability of 'golf'
total_entries = len(df)
golf_count = len(df[df['recreation'] == 'golf'])
unconditional_probability_golf = golf_count / total_entries

# Calculate the conditional probability of 'single' given 'med Risk'
med_risk_count = len(df[df['risk'] == 'med Risk'])
single_and_med_risk_count = len(df[(df['risk'] == 'med Risk') & (df['status'] == 'single')])
conditional_probability_single_given_med_risk = single_and_med_risk_count / med_risk_count if
med_risk_count > 0
 else 0
```

Output :

```
print(f"Unconditional Probability of 'golf': {unconditional_probability_golf:.2f}")
print(f"Conditional Probability of 'single' given 'med Risk':{conditional_probability_single_given_med_risk:.2f}")
```

<u>Viva Questions:</u>

1.What does the unconditional probability of golf represent?
2.How do you calculate the unconditional probability of an event?
3.Why is the len() function used in the program?
4.What is the significance of the home_owner feature in this dataset?
5.State Bayes theorem?

Experiment 6

6.Implement linear regression using python

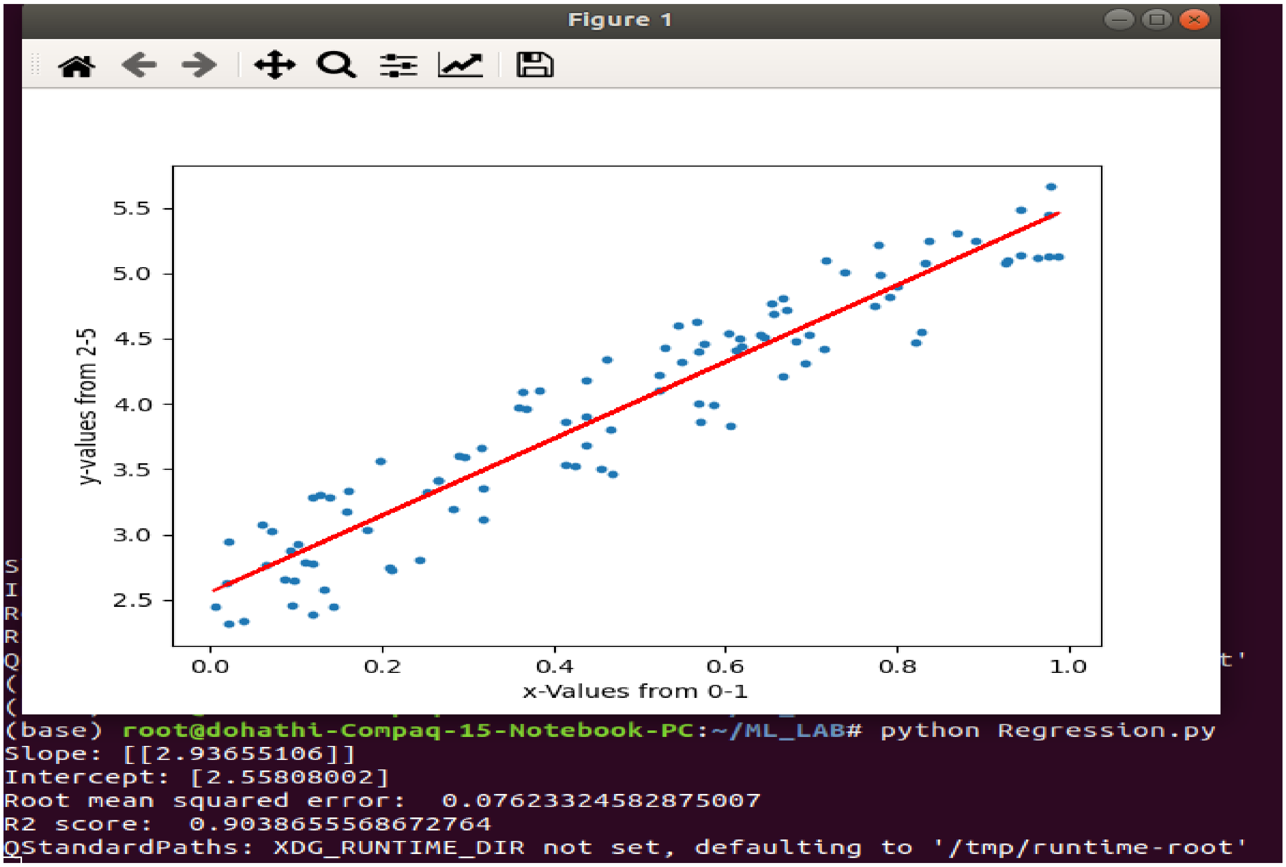
ALGORITHM:

Step 1: Create Database for Linear Regression
Step 2:Finding Hypothesis of Linear Regression
Step 3:Training a Linear Regression model
Step 4:Evaluating the model
Step 5: Scikit-learn implementation
Step 6: End

PROGRAM:

```
# Importing Necessary Libraries
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score# generate random data-set
np.random.seed(0)
x = np.random.rand(100, 1) #Generate a 2-D array with 100 rows, each row containing 1 random numbers:y = 2 +
3 * x + np.random.rand(100, 1)
regression_model = LinearRegression()        # Model initializationregression_model.fit(x, y)        # Fit the
data(train the model) y_predicted = regression_model.predict(x)      # Predict
# model evaluation
rmse = mean_squared_error(y, y_predicted)r2 = r2_score(y, y_predicted)
# printing values
print('Slope:' ,regression_model.coef_) print('Intercept:', regression_model.intercept_)
print('Root mean squared error: ', rmse)print('R2 score: ', r2)
# plotting values  # data pointsplt.scatter(x, y, s=10) plt.xlabel('x-Values from 0-1')
plt.ylabel('y-values from 2-5')# predicted values
plt.plot(x, y_predicted, color='r')plt.show() )
```

OUTPUT:



```
(base) root@dohathi-Compaq-15-Notebook-PC:~/ML_LAB# python Regression.py
Slope:  [[2.93655106]]
Intercept: [2.55808002]
Root mean squared error:   0.07623324582875007
R2 score:   0.9038655568672764
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
```

Viva Voce

1.Define Decision tree?
2.What is ANN?
3.Explain Gradient descent approximation?
4.State Bayes theorem?
5.Define Bayesian belief networks?

EXPERIMENT-7

7.Implement Naive Bayes Theorem to Classify the English Text using python

The Naive Bayes algorithm

Naive Bayes classifiers are a collection of classification algorithms based on **Bayes' Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair offeatures being classified is independent of each other.
The dataset is divided into two parts, namely, **feature matrix** and the **response/target vector**.

- The **Feature matrix** (X) contains all the vectors(rows) of the dataset in which each vector consists ofthe value of **dependent features**. The number of features is **d** i.e. **X = (x1,x2,x2, xd).**
- The **Response/target vector** (y) contains the value of **class/group variable** for **each row of featurematrix.**

Now the "naïve" conditional independence assumptions come into play: assume that all featuresin X are mutually independent, conditional on the category y:

□ **Data Preparation**

- Collect and label the text data.

□ **Text Preprocessing**

- Convert text to lowercase.
- Remove punctuation and special characters.
- Remove stop words.

□ **Feature Extraction**

- Convert the preprocessed text into numerical format (e.g., using CountVectorizer or TF-IDF).

□ **Model Training**

- Calculate the prior probabilities for each class.
- Calculate the likelihood of each feature given each class.
- Use the formula: P(C|X)=P(X|C)·P(C)P(X)P(C | X) = \frac{P(X | C) \cdot P(C)}{P(X)}P(C|X)=P(X)P(X|C)·P(C) Where:
    - P(C|X)P(C | X)P(C|X) is the posterior probability of class CCC given features XXX.
    - P(X|C)P(X | C)P(X|C) is the likelihood of features XXX given class CCC.
    - P(C)P(C)P(C) is the prior probability of class CCC.
    - P(X)P(X)P(X) is the evidence (normalization factor).

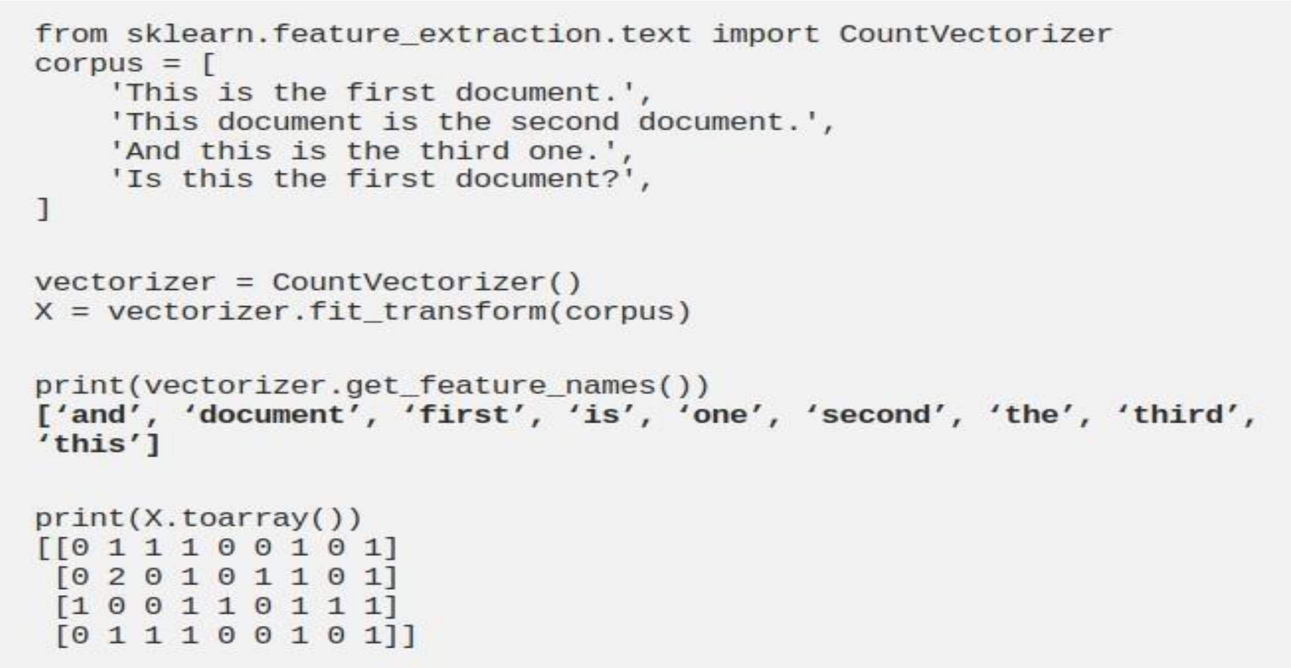
□ **Prediction**

- For a new instance, calculate the posterior probability for each class and choose the class with the highest probability.

□ **Evaluation**

- Use metrics such as accuracy, precision, recall, and F1-score to evaluate the model.

Dealing with text data



```
from sklearn.feature_extraction.text import CountVectorizer
corpus = [
    'This is the first document.',
    'This document is the second document.',
    'And this is the third one.',
    'Is this the first document?',
]

vectorizer = CountVectorizer()
X = vectorizer.fit_transform(corpus)

print(vectorizer.get_feature_names())
['and', 'document', 'first', 'is', 'one', 'second', 'the', 'third',
'this']

print(X.toarray())
[[0 1 1 1 0 0 1 0 1]
 [0 2 0 1 0 1 1 0 1]
 [1 0 0 1 1 0 1 1 1]
 [0 1 1 1 0 0 1 0 1]]
```

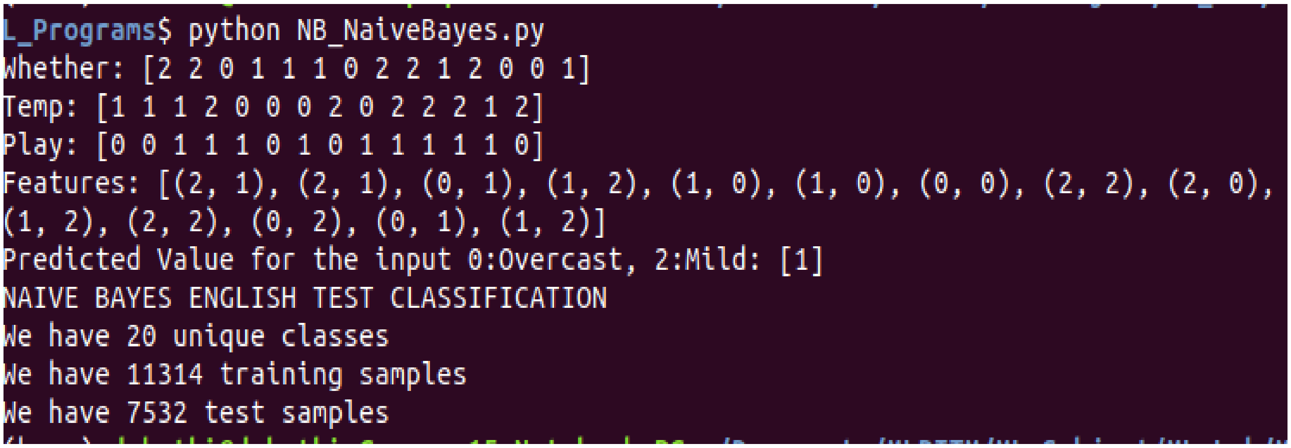
The values 0,1,2, encode the frequency of a word that appeared in the initial text data.

E.g. The first transformed row is **[0 1 1 1 0 0 1 0 1]** and the **unique vocabulary** is **[‘and’, ‘document’, ‘first’, ‘is’, ‘one’, ‘second’, ‘the’, ‘third’, ‘this’],** thus this means that the words “document”, “first”, “is”,“the” and “this” appeared 1 time each in the initial text string (i.e. ‘This is the first document.’).
In our example, we will convert the collection of text documents (train and test sets) into a matrix of token counts.
 To implement that text transformation we will use the **make_pipeline** function. This will internally transformthe text data and then the model will be fitted **using the transformed data**.

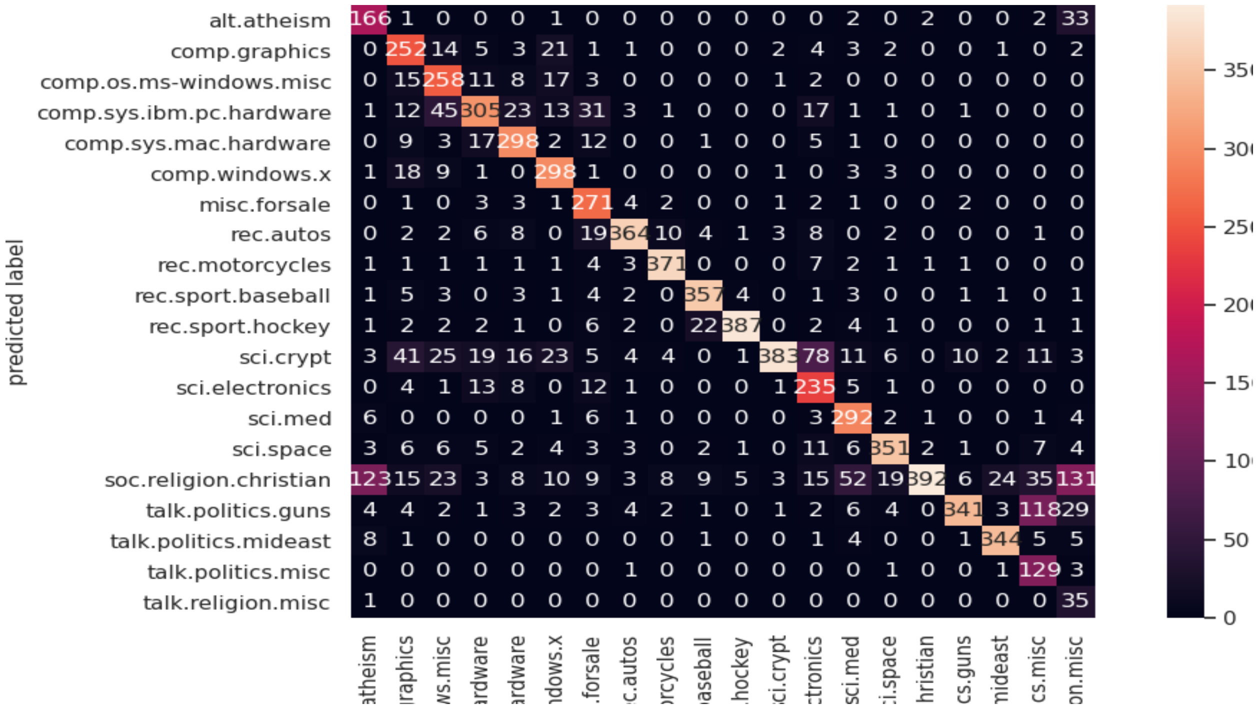
Source Code

```
print("NAIVE BAYES ENGLISH TEST CLASSIFICATION")

import numpy as np, pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt from sklearn.datasets
import fetch_20newsgroupsfrom sklearn.feature_extraction.text
 import TfidfVectorizerfrom sklearn.naive_bayes
 import MultinomialNB from sklearn.pipeline
import make_pipelinefrom sklearn.metrics
 import confusion_matrix, accuracy_scoresns.set()
# use seaborn plotting style
# Load the dataset
data = fetch_20newsgroups()
# Get the text categories text_categories = data.target_names
# define the training set
train_data = fetch_20newsgroups(subset="train", categories=text_categories)# define the test settest_data =
fetch_20newsgroups(subset="test", categories=text_categories)

print("We have {} unique classes".format(len(text_categories)))
 print("We have {} training samples".format(len(train_data.data)))
print("We have {} test samples".format(len(test_data.data)))

# let’s have a look as some training data let it 5th only#print(test_data.data[5])

# Build the model
model = make_pipeline(TfidfVectorizer(), MultinomialNB())# Train the model using the training data
model.fit(train_data.data, train_data.target)# Predict the categories of the test data predicted_categories =
model.predict(test_data.data)

print(np.array(test_data.target_names)[predicted_categories])
# plot the confusion matrix
mat = confusion_matrix(test_data.target, predicted_categories)
sns.heatmap(mat.T,square=True,annot=True,fmt="d",xticklabels=train_data.target_names,yticklabels=train_data.target_names)
plt.xlabel("true labels") plt.ylabel("predicted label")plt.show()
print("The accuracy is {}".format(accuracy_score(test_data.target, predicted_categories)))
```

OUTPUT:



```
L_Programs$ python NB_NaiveBayes.py
Whether: [2 2 0 1 1 1 0 2 2 1 2 0 0 1]
Temp: [1 1 1 2 0 0 0 2 0 2 2 2 1 2]
Play: [0 0 1 1 1 0 1 0 1 1 1 1 1 0]
Features: [(2, 1), (2, 1), (0, 1), (1, 2), (1, 0), (1, 0), (0, 0), (2, 2), (2, 0),
(1, 2), (2, 2), (0, 2), (0, 1), (1, 2)]
Predicted Value for the input 0:Overcast, 2:Mild: [1]
NAIVE BAYES ENGLISH TEST CLASSIFICATION
We have 20 unique classes
We have 11314 training samples
We have 7532 test samples
```



	atheism	raphics	ws.misc	ardware	ardware	ndows.x	.forsale	c.autos	rcycles	aseball	.hockey	ci.crypt	ctronics	sci.med	i.space	hristian	cs.guns	nideast	cs.misc	on.misc
alt.atheism	166	1	0	0	0	1	0	0	0	0	0	0	0	2	0	2	0	0	2	33
comp.graphics	0	252	14	5	3	21	1	1	0	0	0	2	4	3	2	0	0	1	0	2
comp.os.ms-windows.misc	0	15	258	11	8	17	3	0	0	0	0	1	2	0	0	0	0	0	0	0
comp.sys.ibm.pc.hardware	1	12	45	305	23	13	31	3	1	0	0	0	17	1	1	0	1	0	0	0
comp.sys.mac.hardware	0	9	3	17	298	2	12	0	0	1	0	0	5	1	0	0	0	0	0	0
comp.windows.x	1	18	9	1	0	298	1	0	0	0	0	1	0	3	3	0	0	0	0	0
misc.forsale	0	1	0	3	3	1	271	4	2	0	0	1	2	1	0	0	2	0	0	0
rec.autos	0	2	2	6	8	0	19	364	10	4	1	3	8	0	2	0	0	0	1	0
rec.motorcycles	1	1	1	1	1	1	4	3	371	0	0	0	7	2	1	1	1	0	0	0
rec.sport.baseball	1	5	3	0	3	1	4	2	0	357	4	0	1	3	0	0	1	1	0	1
rec.sport.hockey	1	2	2	2	1	0	6	2	0	22	387	0	2	4	1	0	0	0	1	1
sci.crypt	3	41	25	19	16	23	5	4	4	0	1	383	78	11	6	0	10	2	11	3
sci.electronics	0	4	1	13	8	0	12	1	0	0	0	1	235	5	1	0	0	0	0	0
sci.med	6	0	0	0	0	1	6	1	0	0	0	0	3	292	2	1	0	0	1	4
sci.space	3	6	6	5	2	4	3	3	0	2	1	0	11	6	351	2	1	0	7	4
soc.religion.christian	123	15	23	3	8	10	9	3	8	9	5	3	15	52	19	392	6	24	35	131
talk.politics.guns	4	4	2	1	3	2	3	4	2	1	0	1	2	6	4	0	341	3	118	29
talk.politics.mideast	8	1	0	0	0	0	0	0	0	1	0	0	1	4	0	0	1	344	5	5
talk.politics.misc	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	1	129	3
talk.religion.misc	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	35

Viva Questions:

1.What is Naive Bayes classifier?

2. What is the feature matrix in Naive Bayes?

3. What is a confusion matrix?

4. What is feature extraction?

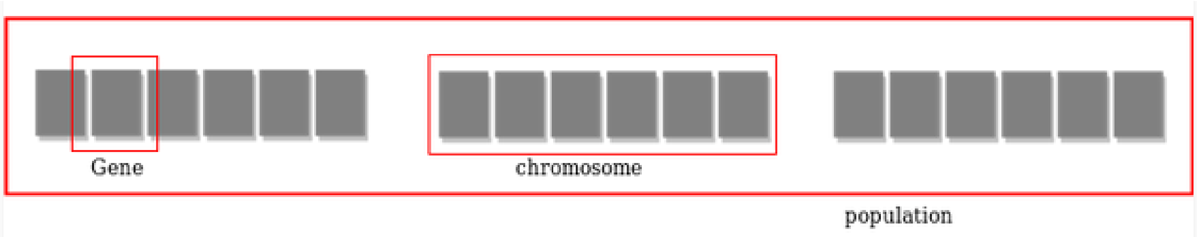
5. What is text preprocessing in machine learning?

Experiment 8

8.Implement an algorithm to demonstrate the significance of Genetic Algorithm in python

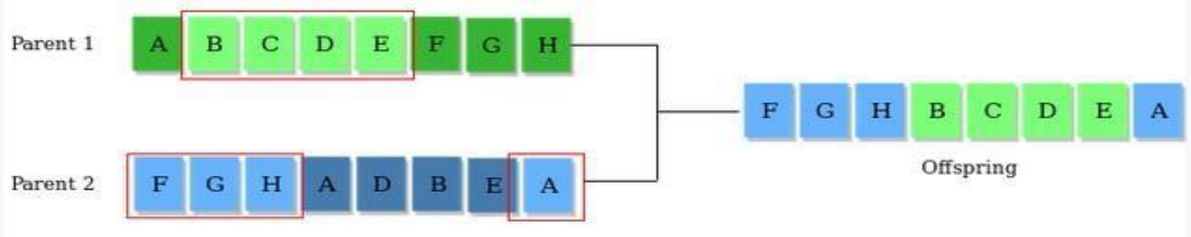
ALGORITHM:

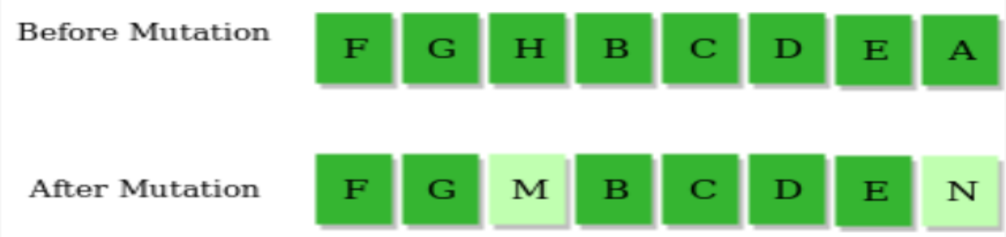
1. Individual in population compete for resources and mate
2. Those individuals who are successful (fittest) then mate to create more offspring than others
3. Genes from "fittest" parent propagate throughout the generation, that is sometimes parents createoffspring which is better than either parent.
4. Thus each successive generation is more suited for their environment.



Operators of Genetic Algorithms

Once the initial generation is created, the algorithm evolve the generation using following operators –
Selection Operator: The idea is to give preference to the individuals with good fitness scores and allow them to pass there genes to the successive generations.
Crossover Operator: This represents mating between individuals. Two individuals are selected using selection operator and crossover sites are chosen randomly. Then the genes at these crossover sites are exchanged thus creating a completely new individual (offspring).
Mutation Operator: The key idea is to insert random genes in offspring to maintain the diversity inpopulation to avoid the premature convergence.





Given a target string, the goal is to produce target string starting from a random string of the same length. Inthe following implementation, following analogies are made –
Characters A-Z, a-z, 0-9 and other special symbols are considered as genes
A string generated by these character is considered as chromosome/solution/Individual
Fitness score is the number of characters which differ from characters in target string at a particular index. So individual having lower fitness value is given more preference.

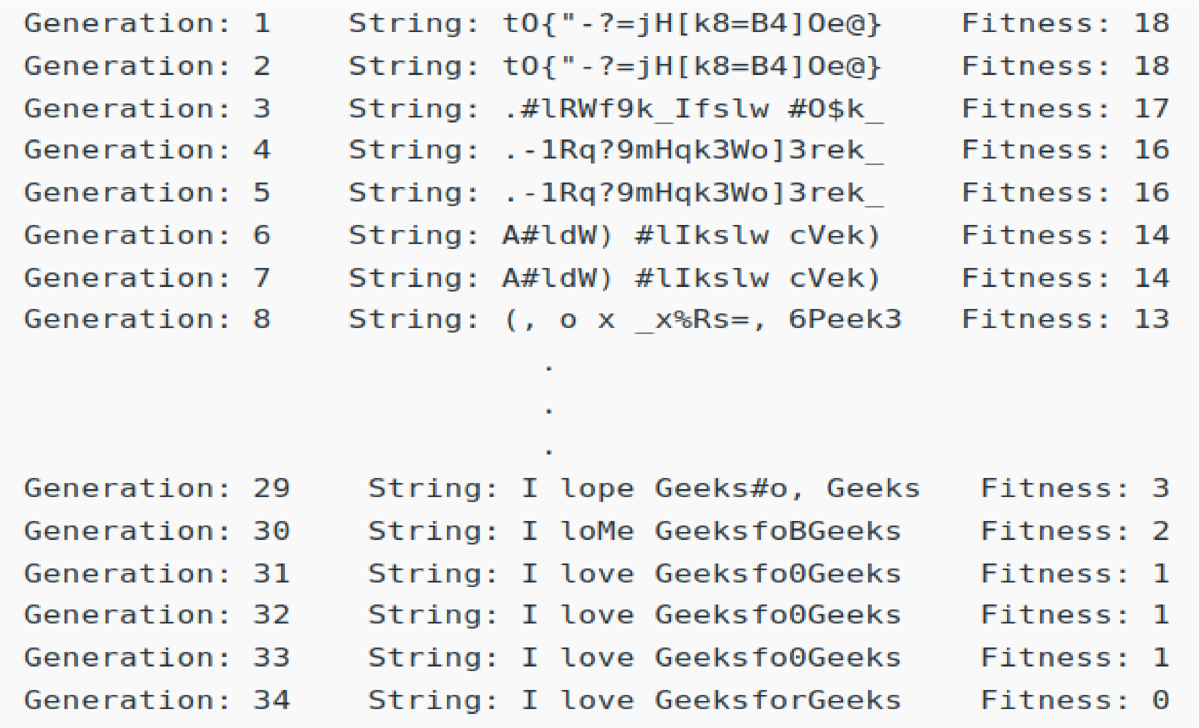
Source Code

```
# Python3 program to create target string, starting from# random string using Genetic Algorithm

import random

# Number of individuals in each generationPOPULATION_SIZE = 100

# Valid genes
GENES = '''abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890, .-
;:_!"#%&/()=?@${[]}'''

# Target string to be generated TARGET = "I love GeeksforGeeks"

class Individual(object):'''
Class representing individual in population    '''def___init_(self, chromosome):
self.chromosome = chromosomeself.fitness = self.cal_fitness()

@classmethod
def mutated_genes(self):'''
create random genes for mutation'''
global GENES
gene = random.choice(GENES)return gene

@classmethod
def create_gnome(self):'''
create chromosome or string of genes'''
global TARGET

gnome_len = len(TARGET)
return [self.mutated_genes() for _ in range(gnome_len)]
```

```
def mate(self, par2):
'''Perform mating and produce new offspring '''

# chromosome for offspringchild_chromosome = []
for gp1, gp2 in zip(self.chromosome, par2.chromosome):

# random probability prob = random.random()

# if prob is less than 0.45, insert gene# from parent 1
if prob < 0.45: child_chromosome.append(gp1)

# if prob is between 0.45 and 0.90, insert# gene from parent 2
elif prob < 0.90: child_chromosome.append(gp2)

# otherwise insert random gene(mutate),# for maintaining diversity
else:
child_chromosome.append(self.mutated_genes())

# create new Individual(offspring) using# generated chromosome for offspring return
Individual(child_chromosome)

def cal_fitness(self):
'''        Calculate fitness score, it is the number of characters in string which differ from target string.      '''global
TARGET
fitness = 0
for gs, gt in zip(self.chromosome, TARGET):if gs != gt: fitness+= 1
return fitness

# Driver codedef main():
global POPULATION_SIZE

#current generationgeneration = 1

found = False population = []

# create initial population
for _ in range(POPULATION_SIZE): gnome = Individual.create_gnome()
population.append(Individual(gnome))while not found:
# sort the population in increasing order of fitness score population = sorted(population, key = lambda x:x.fitness)

# if the individual having lowest fitness score ie.
# 0 then we know that we have reached to the target# and break the loop
if population[0].fitness <= 0:found = True
break
```

```
# Otherwise generate new offsprings for new generationnew_generation = []

# Perform Elitism, that mean 10% of fittest population# goes to the next generation
s = int((10*POPULATION_SIZE)/100)
new_generation.extend(population[:s])

# From 50% of fittest population, Individuals# will mate to produce offspring
s = int((90*POPULATION_SIZE)/100)
for _ in range(s):
parent1 = random.choice(population[:50])parent2 = random.choice(population[:50])child = parent1.mate(parent2)
new_generation.append(child)
population = new_generation print("Generation: {}\tString: {}\tFitness: {}".\
format(generation, "".join(population[0].chromosome),population[0].fitness))
generation += 1
print("Generation: {}\tString: {}\tFitness: {}".\format(generation, "".join(population[0].chromosome),
population[0].fitness))

if____________name__== ' main ':main()
```

OUTPUT:



```
Generation: 1    String: tO{"-?=jH[k8=B4]Oe@}    Fitness: 18
Generation: 2    String: tO{"-?=jH[k8=B4]Oe@}    Fitness: 18
Generation: 3    String: .#lRWf9k_Ifslw #O$k_    Fitness: 17
Generation: 4    String: .-1Rq?9mHqk3Wo]3rek_    Fitness: 16
Generation: 5    String: .-1Rq?9mHqk3Wo]3rek_    Fitness: 16
Generation: 6    String: A#ldW) #lIkslw cVek)    Fitness: 14
Generation: 7    String: A#ldW) #lIkslw cVek)    Fitness: 14
Generation: 8    String: (, o x _x%Rs=, 6Peek3   Fitness: 13
                               .
                               .
                               .
Generation: 29    String: I lope Geeks#o, Geeks   Fitness: 3
Generation: 30    String: I loMe GeeksfoBGeeks    Fitness: 2
Generation: 31    String: I love Geeksfo0Geeks    Fitness: 1
Generation: 32    String: I love Geeksfo0Geeks    Fitness: 1
Generation: 33    String: I love Geeksfo0Geeks    Fitness: 1
Generation: 34    String: I love GeeksforGeeks    Fitness: 0
```

Viva Questions:

1. What is a genetic algorithm?
2. What is a fitness score in a genetic algorithm?
3. What does the selection operator do in a genetic algorithm?
4. What is the crossover operator in genetic algorithms?
5. What is the mutation operator in genetic algorithms?

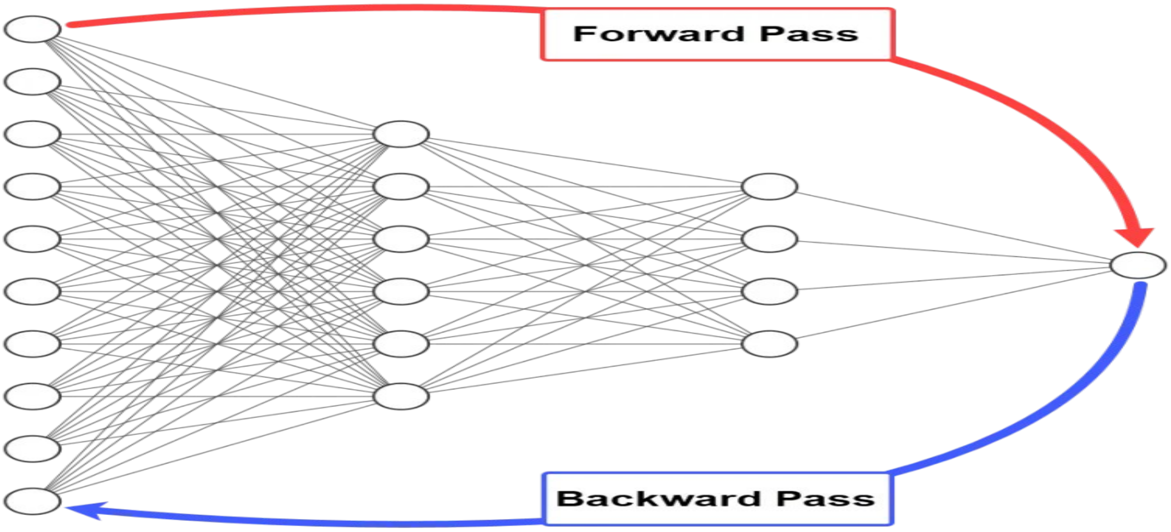
Experiment 9

9.Implement an algorithm to demonstrate Back Propagation Algorithm in python

ALGORITHM:

It is the most widely used algorithm for training artificial neural networks.
In the simplest scenario, the architecture of a neural network consists of some sequential layers, where thelayer numbered *i* is connected to the layer numbered *i+1*. The layers can be classified into 3 classes:

1. Input
2. Hidden
3. Output



Usually, each neuron in the hidden layer uses an activation function like sigmoid or rectified linear unit (ReLU). This helps to capture the non-linear relationship between the inputs and their outputs.
The neurons in the output layer also use activation functions like sigmoid (for regression) or SoftMax (for classification).
To train a neural network, there are 2 passes (phases):

- Forward
- Backward

The forward and backward phases are repeated from some epochs. In each epoch, the following occurs:

1. The inputs are propagated from the input to the output layer.
2. The network error is calculated.
3. The error is propagated from the output layer to the input layer.

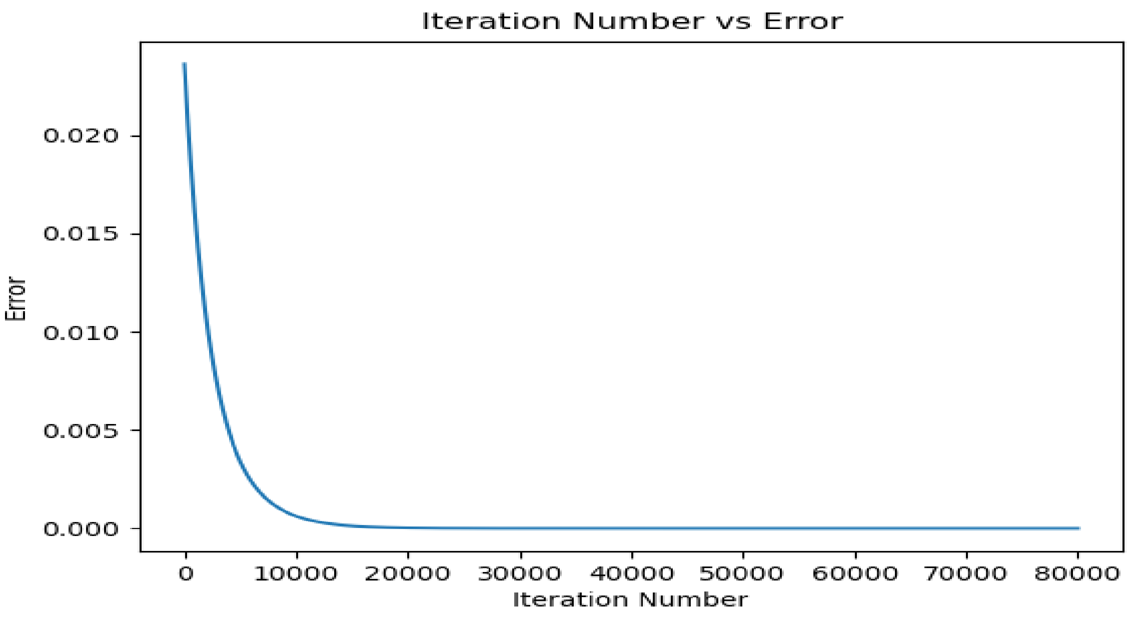
Knowing that there's an error, what should we do? We should minimize it. To minimize network error, wemust change something in the network. Remember that the only parameters we can change are the weightsand biases. We can try different weights and biases, and then test our network.

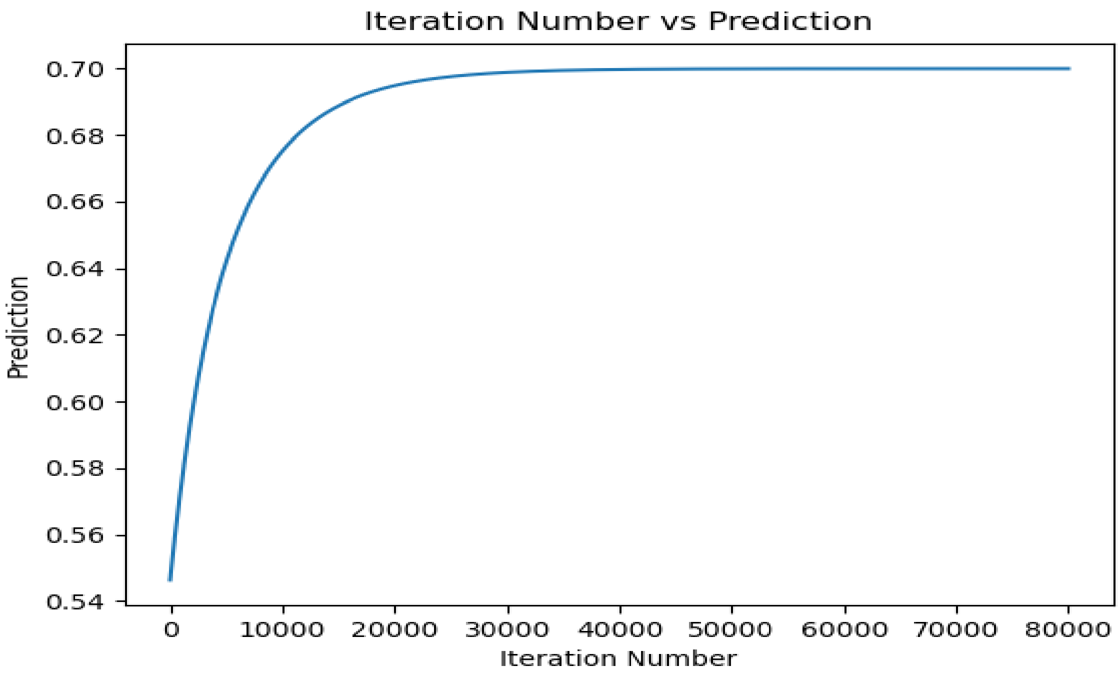
Source Code:

```
import numpy
import matplotlib.pyplot as plt
def sigmoid(sop):
return 1.0/(1+numpy.exp(-1*sop))
def error(predicted, target):
return numpy.power(predicted-target, 2)

def error_predicted_deriv(predicted, target):
return 2*(predicted-target)

def sigmoid_sop_deriv(sop):
return sigmoid(sop)*(1.0-sigmoid(sop))
def sop_w_deriv(x):return x
def update_w(w, grad, learning_rate):return w - learning_rate*grad
x1=0.1x2=0.4
target = 0.7
learning_rate = 0.01
w1=numpy.random.rand().
w2=numpy.random.rand()
print("Initial W : ", w1, w2)
predicted_output = []network_error = []
old_err = 0
for k in range(80000):# Forward Pass
y = w1*x1 + w2*x2
 predicted = sigmoid(y)
err = error(predicted, target)
predicted_output.append(predicted)network_error.append(err)
# Backward Pass
g1 = error_predicted_deriv(predicted, target)
g2 = sigmoid_sop_deriv(y)

 g3w1 = sop_w_deriv(x1)
g3w2 = sop_w_deriv(x2)

gradw1 = g3w1*g2*g1
gradw2 = g3w2*g2*g1

w1 = update_w(w1, gradw1, learning_rate)
w2 = update_w(w2, gradw2, learning_rate)
        #print(predicted)
plt.figure() plt.plot(network_error)
plt.title("Iteration Number vs Error")
plt.xlabel("Iteration Number")
plt.ylabel("Error")
plt.show()

plt.figure() plt.plot(predicted_output)
plt.title("Iteration Number vs Prediction")
plt.xlabel("Iteration Number")
plt.ylabel("Prediction")
plt.show()
```

OUTPUT:

```
Initial W :  0.08698924153243281     0.4532713230157145
```





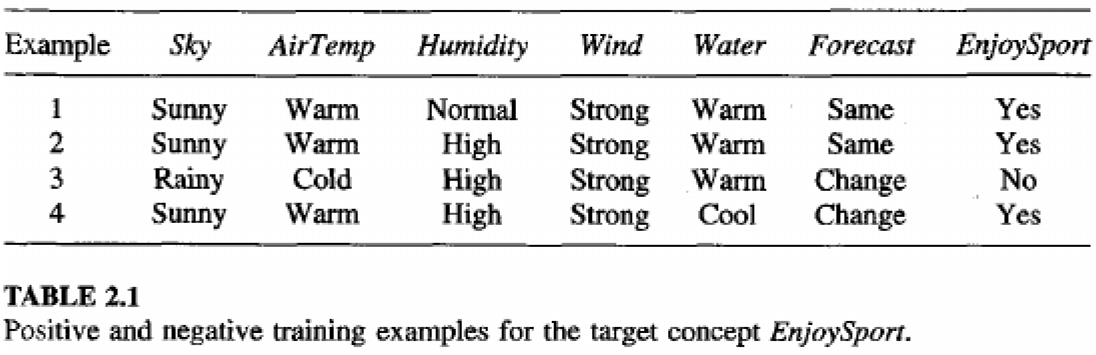
Viva Questions:

1.What is the Backpropagation algorithm used for?
2.What are the three types of layers in a neural network?
3.What is the role of weights and biases in a neural network?
4.Why do we use the `numpy` library in the Backpropagation algorithm?
5.What is the activation function in the hidden layer of a neural network?

ADDITIONAL PROGRAMS

1. Implementing FIND-S algorithm using python

Training Database



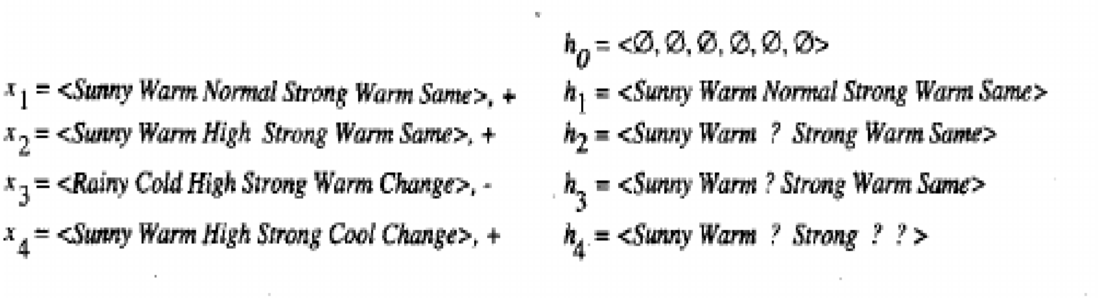
Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

TABLE 2.1
Positive and negative training examples for the target concept *EnjoySport*.

Algorithm

```
1. Initialize h to the most specific hypothesis in H
2. For each positive training instance
        x For each attribute constraint
        a, in h
            If the constraint a, is satisfied
                by xThen do nothing
            Else replace a, in h by the next more general constraint that is satisfied by x
3. Output hypothesis h
```

Hypothesis Construction

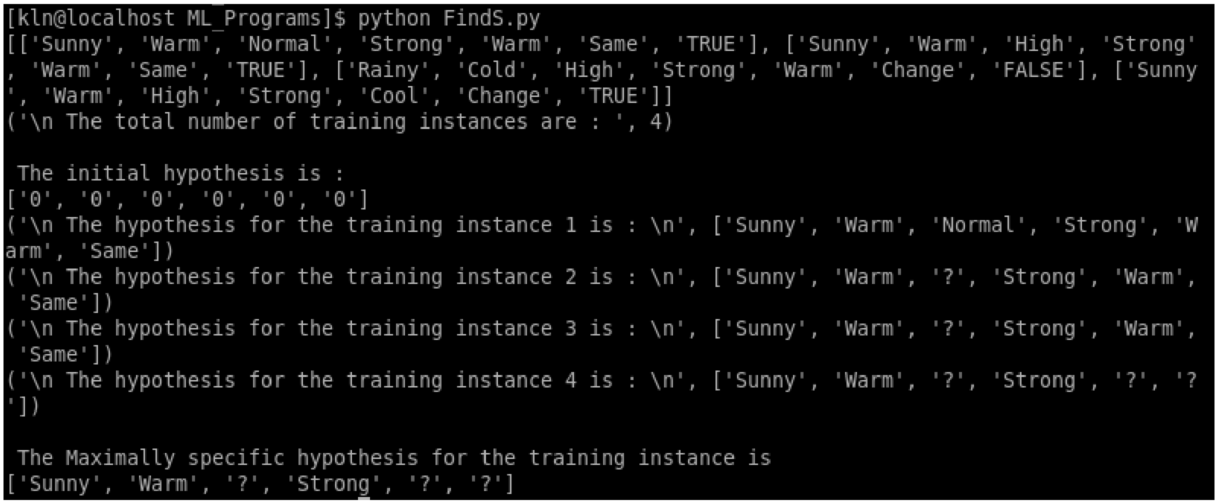


	h_0 = <∅, ∅, ∅, ∅, ∅, ∅>
x_1 = <*Sunny Warm Normal Strong Warm Same*>, +	h_1 = <*Sunny Warm Normal Strong Warm Same*>
x_2 = <*Sunny Warm High Strong Warm Same*>, +	h_2 = <*Sunny Warm ? Strong Warm Same*>
x_3 = <*Rainy Cold High Strong Warm Change*>, -	h_3 = <*Sunny Warm ? Strong Warm Same*>
x_4 = <*Sunny Warm High Strong Cool Change*>, +	h_4 = <*Sunny Warm ? Strong ? ?* >

Source Code:

```
with open('enjoysport.csv', 'r') as
    csvfile:for row in csv.reader(csvfile):
        a.append(ro
    w)print(a)
print("\n The total number of training instances are : ",len(a))num_attribute =
len(a[0])-1
print("\n The initial hypothesis is :
")hypothesis = ['0']*num_attribute
print(hypothesis)
for i in range(0, len(a)):
    if a[i][num_attribute] == 'TRUE':                    #for each positive example
         onlyfor j in range(0, num_attribute):
                if hypothesis[j] == '0' or hypothesis[j] == a[i][j]:
                     hypothesis[j] = a[i][j]
                else:
                     hypothesis[j] = '?'
    print("\n The hypothesis for the training instance {} is : \n".format(i+1),hypothesis)
print("\n The Maximally specific hypothesis for the training instance is ") print(hypothesis)
```

OUTPUT:



```
[kln@localhost ML_Programs]$ python FindS.py
[['Sunny', 'Warm', 'Normal', 'Strong', 'Warm', 'Same', 'TRUE'], ['Sunny', 'Warm', 'High', 'Strong'
, 'Warm', 'Same', 'TRUE'], ['Rainy', 'Cold', 'High', 'Strong', 'Warm', 'Change', 'FALSE'], ['Sunny
', 'Warm', 'High', 'Strong', 'Cool', 'Change', 'TRUE']]
('\n The total number of training instances are : ', 4)

 The initial hypothesis is :
['0', '0', '0', '0', '0', '0']
('\n The hypothesis for the training instance 1 is : \n', ['Sunny', 'Warm', 'Normal', 'Strong', 'W
arm', 'Same'])
('\n The hypothesis for the training instance 2 is : \n', ['Sunny', 'Warm', '?', 'Strong', 'Warm',
 'Same'])
('\n The hypothesis for the training instance 3 is : \n', ['Sunny', 'Warm', '?', 'Strong', 'Warm',
 'Same'])
('\n The hypothesis for the training instance 4 is : \n', ['Sunny', 'Warm', '?', 'Strong', '?', '?
'])

 The Maximally specific hypothesis for the training instance is
['Sunny', 'Warm', '?', 'Strong', '?', '?']
```

Viva Voce:

1. What is the purpose of the FIND-S algorithm?

2. What does the hypothesis represent in the FIND-S algorithm?

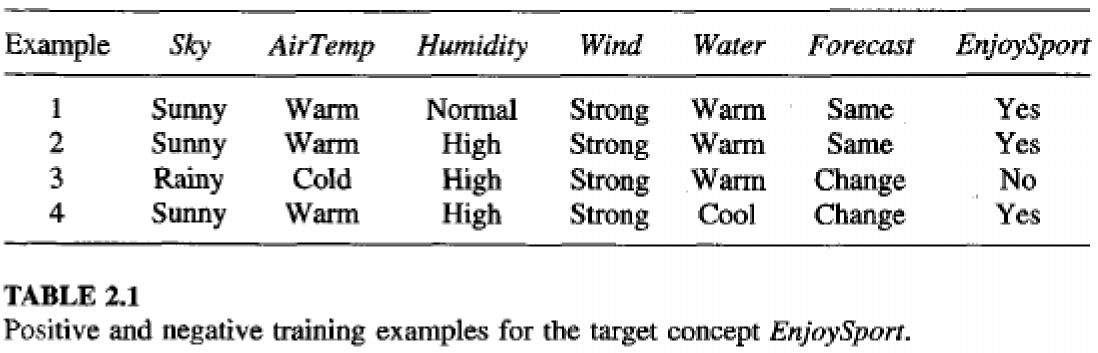
3. How does the FIND-S algorithm initialize the hypothesis?

4. What is the role of the attribute constraints in the FIND-S algorithm?

5. What does the term "more general constraint" mean in the context of the FIND-S algorithm?

2. Implementing Candidate Elimination algorithm using python

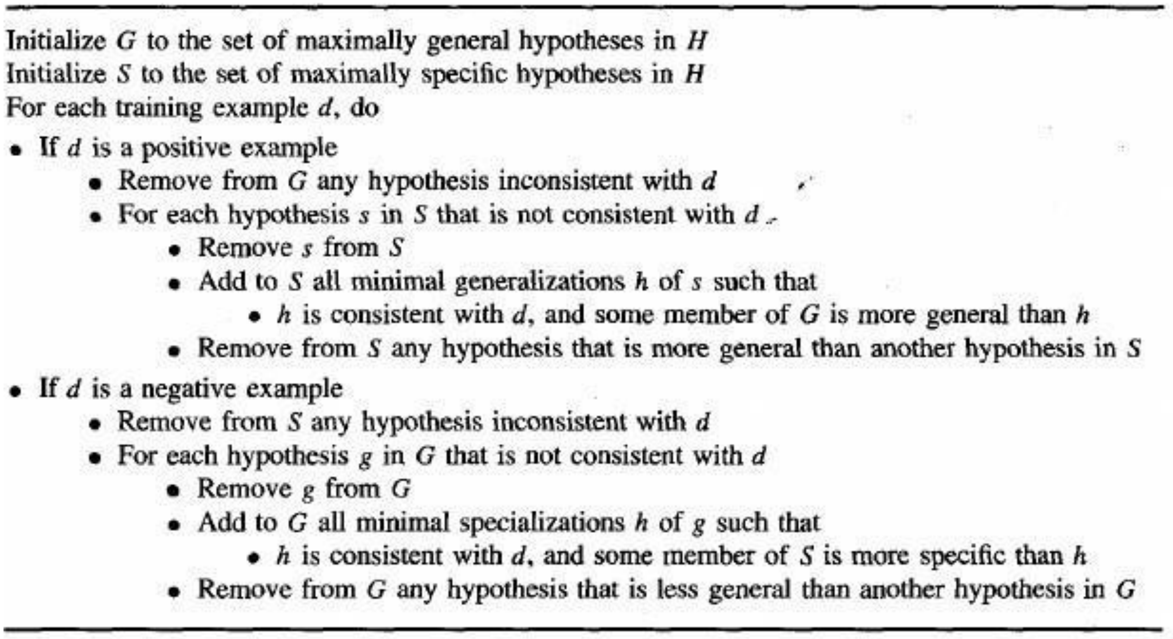
Training Database



Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

TABLE 2.1
Positive and negative training examples for the target concept *EnjoySport*.

Algorithm



```
Initialize G to the set of maximally general hypotheses in H
Initialize S to the set of maximally specific hypotheses in H
For each training example d, do
• If d is a positive example
    • Remove from G any hypothesis inconsistent with d
    • For each hypothesis s in S that is not consistent with d
        • Remove s from S
        • Add to S all minimal generalizations h of s such that
            • h is consistent with d, and some member of G is more general than h
        • Remove from S any hypothesis that is more general than another hypothesis in S
• If d is a negative example
    • Remove from S any hypothesis inconsistent with d
    • For each hypothesis g in G that is not consistent with d
        • Remove g from G
        • Add to G all minimal specializations h of g such that
            • h is consistent with d, and some member of S is more specific than h
        • Remove from G any hypothesis that is less general than another hypothesis in G
```

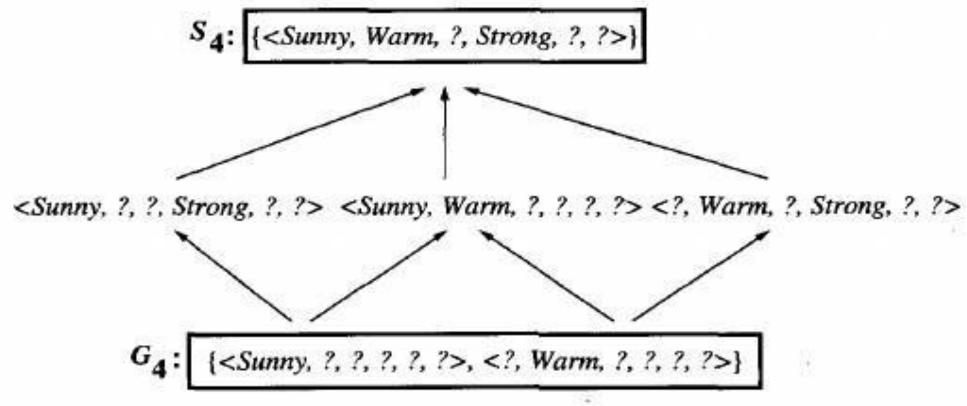


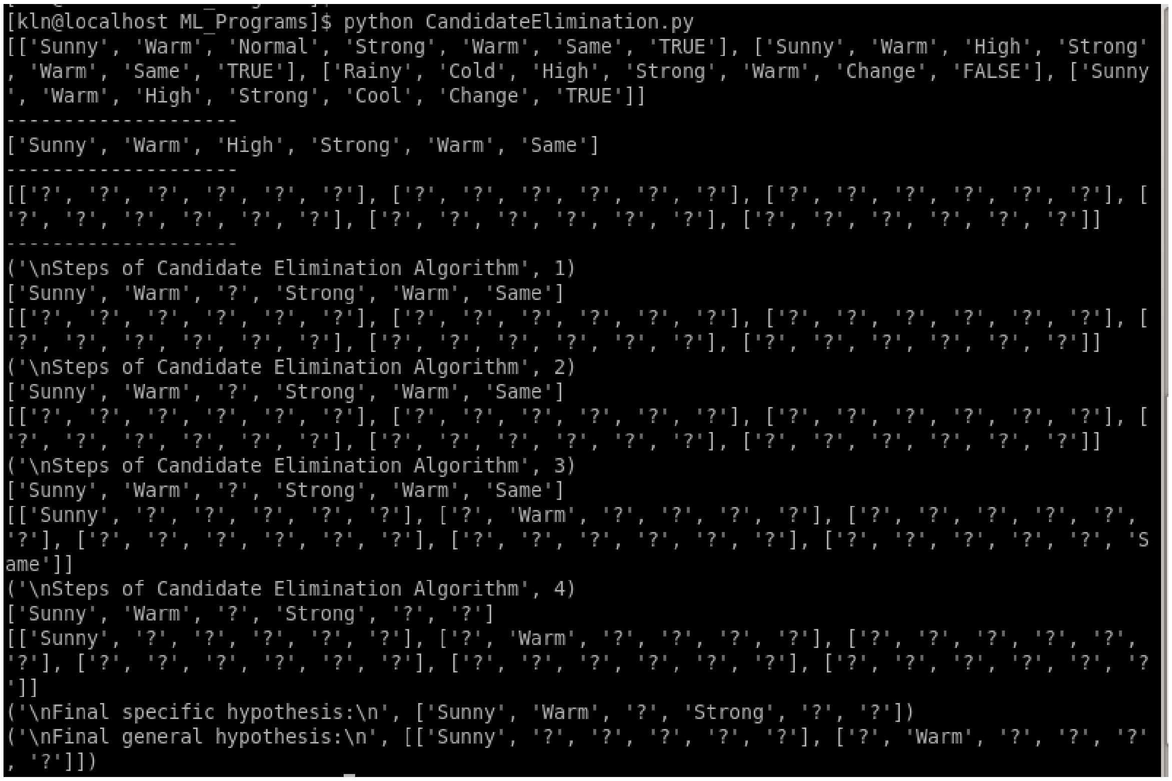
FIGURE 2.7
The final version space for the *EnjoySport* concept learning problem and training examples described earlier.

Source Code:

```
import csv

with open("enjoysport.csv")
   as f:csv_file=csv.reader(f)
   data=list(csv_file)

   print(data)
   print("..............................")
   s=data[1][:-1]     #extracting one row or instance or
   recordg=[['?' for i in range(len(s))] for j in
   range(len(s))]

   print(s)
   print("..............................")print(g)
   print("..............................")

   for i in data:
      if i[-1]=="TRUE":          # For each positive training record or
         instancefor j in range(len(s)):
               if
            i[j]!=s[j]
               :
             s[j]='?'
             g[j][j]='?'

      elif i[-1]=="FALSE":      # For each negative training record or
         examplefor j in range(len(s)):
            if i[j]!=s[j]:
               g[j][j]=s[j]
```

```
                else:
                    g[j][j]="?"
        print("\nSteps of Candidate Elimination Algorithm",data.index(i)+1)
        print(s)
        print(g
    )
    gh=[]
    for i
    in g:
        for
        in
        i:
            if j!='?':
                gh.append(i
                )break
    print("\nFinal specific
    hypothesis:\n",s)print("\nFinal
    general hypothesis:\n",gh)
```

OUTPUT:



```
[kln@localhost ML_Programs]$ python CandidateElimination.py
[['Sunny', 'Warm', 'Normal', 'Strong', 'Warm', 'Same', 'TRUE'], ['Sunny', 'Warm', 'High', 'Strong'
, 'Warm', 'Same', 'TRUE'], ['Rainy', 'Cold', 'High', 'Strong', 'Warm', 'Change', 'FALSE'], ['Sunny
', 'Warm', 'High', 'Strong', 'Cool', 'Change', 'TRUE']]
--------------------
['Sunny', 'Warm', 'High', 'Strong', 'Warm', 'Same']
--------------------
[['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], [
'?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?']]
--------------------
('\nSteps of Candidate Elimination Algorithm', 1)
['Sunny', 'Warm', '?', 'Strong', 'Warm', 'Same']
[['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], [
'?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?']]
('\nSteps of Candidate Elimination Algorithm', 2)
['Sunny', 'Warm', '?', 'Strong', 'Warm', 'Same']
[['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], [
'?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?']]
('\nSteps of Candidate Elimination Algorithm', 3)
['Sunny', 'Warm', '?', 'Strong', 'Warm', 'Same']
[['Sunny', '?', '?', '?', '?', '?'], ['?', 'Warm', '?', '?', '?', '?'], ['?', '?', '?', '?', '?',
'?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', 'S
ame']]
('\nSteps of Candidate Elimination Algorithm', 4)
['Sunny', 'Warm', '?', 'Strong', '?', '?']
[['Sunny', '?', '?', '?', '?', '?'], ['?', 'Warm', '?', '?', '?', '?'], ['?', '?', '?', '?', '?',
'?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?
']]
('\nFinal specific hypothesis:\n', ['Sunny', 'Warm', '?', 'Strong', '?', '?'])
('\nFinal general hypothesis:\n', [['Sunny', '?', '?', '?', '?', '?'], ['?', 'Warm', '?', '?', '?'
, '?']])
```

Viva Voce:

1.What is the main goal of the Candidate Elimination algorithm?

2. How does the Candidate Elimination algorithm initialize the specific and general hypotheses?

3. What role do the positive and negative training instances play in the Candidate Elimination algorithm?

4.What is the difference between the specific and general hypotheses in Candidate Elimination?

5. What is the role of the CSV file in the Candidate Elimination algorithm?