Example	Sky	Temp	Humidity	Wind	Water	Forecast	Enjoyment
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

Output:

The attribute are [[1 'Sunny' 'warm' 'normal' 'strong' 'warm' 'same']
['2' 'sunny' 'warm' 'High' 'strong' 'warm' 'same']
['3' 'Rainy' 'cold' 'high' 'strong' 'warm' 'change'] ['4' 'sunny' 'warm' 'high' 'strong' 'cold' 'change']

The target is : ['yes', 'yes', 'No', 'Yes']

The final hypothesis is :- ['?' 'sunny' 'warm' '?' 'strong' '?' '?']

Expt. No. 2 Page No. 9

Expt. Name. CANDIDATE ELIMINATION ALGORITHM Date : 11.

AIM :

Implement & demonstrate the Candidate elimination algorithm to output for a description of the set of all training data example stored in a .csv file.

Algorithm

1) For each training example d, do: if d is positive example

2). Remove for G any hypothesis h inconsistent with d.

3). For each hypothesis s in S not consistent with d.

4). Remove s from S.

5). Add to S all minimal generalization of s consistent with d & having a generalization in G.

6). Remove from S any hypothesis with a more specific h is S.

7. If d is negative example. Remove from S any hypothesis h inconsistent with d

8. Remove s from G.

9. Add to G all minimal specification of S consistent with d & having a specialization in S.

Instances are

[['sunny' 'warm' 'normal' 'strong' 'warm' 'same']
['sunny' 'warm' 'high' 'strong' 'warm' 'same']
['rainy' 'cold' 'high' 'strong' 'warm' 'change']
['sunny' 'warm' 'high' 'strong' 'cool' 'change']]

Target values are: ['Yes' 'Yes' 'No' 'Yes']

Initialization of Specific -h and general -h

Specific Boundary: ['sunny' 'warm' 'normal' 'strong' 'warm' 'same']

Generic Boundary: [['?' '?' '?' '?' '?' '?'], ['?' '?' '?' '?' '?' '?'], ['?' '?' '?' '?' '?' '?']
['?' '?' '?' '?' '?' '?'] ['?' '?' '?' '?' '?' '?'] ['?' '?' '?' '?' '?' '?']]

Instance I is ['sunny', 'warm', 'normal' 'strong' 'warm' 'same']

Instance is positive :-

Specific Boundary After Instance ['sunny' 'warm' 'normal' 'strong' 'warm' 'same']

Generic Boundary After I Instances ['?' '?' '?' '?' '?' '?'] ['?' '?' '?' '?' '?' '?']
['?' '?' '?' '?' '?' '?' '?'] ['?' '?' '?' '?' '?' '?'] ['?' '?' '?' '?' '?' '?']
['?' '?' '?' '?' '?' '?']

Instance 2 is ['sunny', 'warm', 'high', 'strong', 'warm', 'same']

Instance is positive

Specific Boundary after 2 instances ['sunny', 'warm', '?', 'strong', 'warm', 'same'].

Generic Boundary After 2 Instance [['?' '?' '?' '?' '?' '?']

['?' '?' '?' '?' '?' '?'] ['?' '?' '?' '?' '?' '?'] ['?' '?' '?' '?' '?' '?']

['?' '?' '?' '?' '?' '?'] ['?' '?' '?' '?' '?' '?']

Instance 3 is ['Rainy' 'cold' 'high' 'strong' 'warm' 'change']

Instance 3 is Negative.

Specific Boundary after 3 Instance ['sunny' 'warm' '?' 'strong' 'warm', 'same']

Generic Boundary after 3 Instances [['sunny' '?' '?' '?' '?' '?']

['?' 'warm' '?' '?' '?' '?'] ['?' '?' '?' '?' '?' '?'], ['?' '?' '?' '?' '?' '?']

['?' '?' '?' '?' '?' '?'] ['?' '?' '?' '?' '?' 'same']

Instance 4 is ['sunny' 'warm' 'high' 'strong' 'cool' 'change']

Instance 4 positive.

Specific Boundary after 4 Instance is ['sunny', warm '?', 'strong', '?' '?']

Expt. No. _____ Page No. 11

Expt. Name. _____ Date : _____

10. Remove from G any hypothesis having a more general hypothesis in G.

Source code:

```
import numpy as np
import pandas as pd

data = pd.read_csv("c:/downloads/Sportnew.csv")
concepts = np.array(data.iloc[:,0:-1])
print("\n Instances are: \n", concepts)
target = np.array(data.iloc[:,-1])
print("\n Target Values are:", target)

def learn(concepts, targets):
Specific_h = concepts[0].copy()
print("\n Initialization of Specific_h and general_h")
print("\n Specific Boundary:", Specific_h)
general_h = [["?" for i in range(len(Specific_h))] for i in
                                range(len(Specific_h))]

Print("\n Generic Boundary:", general_h)

for i, h in enumerate(concepts):
    print("\n Instance", i+1, "is", h)
    if target[i] == "yes":
print("Instance is positive")
    for x in range(len(Specific_h)):
        if h[x] != Specific_h(x):
```

Expt. No. 3 Page No. 15

Expt. Name. Decision Tree Based ID3 Algorithm Date :

AIM:

To write a program to demonstrate the working of the decision tree base ID3 algorithm.

Algorithm:

Step 1: Declare the Target_attributes is the attribute whose value is to be predicted by the tree. Attribute is a list of Other attributes that may be tested by the learning decision tree. Example attribute are returned by a decision that correctly classifies the given examples.

Step 2:

Execute the function ID3 (Example, Target_attribute, Attribute).

Step 3: Create a Root node for the tree. If all examples are positive, Return the single node tree root, with label = +. If all the examples are negative, Return the single node tree root, with label = -. If Attribute is Empty, return the single node tree root, with label = most common value of Target_attribute in example.

Step 4:

Otherwise Begin

A ← the attribute from Attributes that best classifies examples

The decision attribute for Root ← A

For each positive value, vi of A

Add a new tree branch below root, Corresponding to the test A = Vi

Let Example v_i, be the subset of Example, that have value v_i for A.

If Example v_i, is empty.

Then below their new branch add a leaf node with label = most common values of Target_attribute in Examples.

else

below this new branch and the subtree ID3 (eg : v_i, Target_Attribute, Attribute {A})

End

Steps :

Return Root.

Output :

Outlook

Overcast → ['Yes']

rain

wind

Strong → ['no']

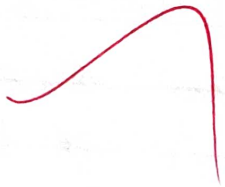
weak → ['Yes']

Sunny

humidity

high → ['no']

normal → ['yes']



Expt. No. 4 Page No. 11

Expt. Name. Back Propagation Algorithm Date :

AIM:

To build an Artificial Neural Network by implementing the back propagation algorithm and test the same using appropriate data sets.

ALGORITHM:

1). Create a feed forward network ni inputs nHidden units, Noutput units

2). Initialize all network device to small random units

3). For each training sample, do propagate the input forward through the network input x to the network and compute output of every unit is network.

4). Propagate the errors backward through the network.

For each network unit k.

Calculate ek error δ_k

$$\delta_k \leftarrow O_k(1-O_k)(t_k - O_k)$$

For each network unit h, calculate it error δ_h

$$\delta_h \leftarrow O_h(1-O_h) \sum_{k \in \text{outputs}} w_{h,k}\,\delta_k$$

5). Update each network weight $w_{j,i}$

$$w_{j,i} \leftarrow w_{j,i} + \Delta w_{j,i}$$

where

$$\Delta w_{j,i} = \eta\,\delta_j\,x_{j,i}$$

Output:

Input :

```
[[0.66666671]
 [0.33333333 0.55555556]
 [          0.66666667]]
```

Actual output

```
[[0.92]
 [0.86]
 [0.89]]
```

Predicted output :

```
[[0.83933318]
 [0.82983658]
 [0.83570317]]
```

Expt. No. 5 Page No. 35

Expt. Name. Naive Bayesian classification Date :

AIM:

Write a program to implement the naive Bayesian classifier for a sample training data set stored as a .csv file. Compute the accuracies of the classifiers, considering few data sets.

Algorithm:

Step 1: Convert the dataset into a frequency table
Step 2: Compute the 'prior' probability for each the class.
Step 3: Compute the probability of evidence
Step 4: Compute the probability of likelihood of evidence
Step 5: Substitute all the values into naive bayer formula to get the probability.

$$P(y/x) = \frac{P(x/y)}{P(x)} * P(y)$$

Step 6: The class with the highest posterior probability outcome of prediction.

Code:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
dataset = pd.read_csv("D:/NaiveBayes.csv")
```

Output :-

Age	Salary	Purchased
19	19000	0
35	20000	0
26	43000	0
.	.	.
.	.	.
.	.	.
50	20,000	1
36	33,000	0
49	36,000	1

0.91

Expt. No. 6. Page No. 39

Expt. Name. Naive Bayesian Classifier model - Calculate accuracy, precision & Recall for the dataset Date :

AIM:

By assuming a set of documents that need to be classified, use the naive Bayesian Classifier model to perform this task. Built-in Java classes / API can be used to write the program. Calculate the accuracy, precision, and recall for your dataset.

Algorithm:

Step 1: Convert the dataset into a frequency table

Step 2: Compute the 'prior' probabilities for each of the class

step 3: Compute the probability of evidence.

Step 4: Compute the probability of likelihood of evidence

Step 5: Substitute all the values into the Naive Bayes formula to get the probability.

$$P(Y/x) = \frac{P(x/y) * P(y)}{P(x)}$$

Step 6: The class with the highest posterior probability is the outcome of prediction.

Step 7: Compute the Accuracy for the given dataset

$$\text{Accuracy} = \frac{TN + TP}{TN + FP + TP + FN}$$

Step 8: Compute the precision

$$\text{precision} = \frac{TP}{TP + FP}$$

Dataset :

Age	Salary	purchased
19	19000	0
35	20000	0
26	43000	0
⋮	⋮	⋮
50	20000	1
36	33000	0
49	36000	1

Output :-

0.91

Accuracy Metrics :

Accuracy = 0.91

Recall : 0.84375

Precision : 0.8709677419354839

Confusion Matrix :

[[64 4]
[5 27]]

Step 9: Compute the Recall

$$\text{Recall} = \frac{TP}{TP + FN}$$

Step 10: Compute the confusion Matrix.

	Negative	Positive
Negative	True Negative	False positive
positive	False Negative	True Positive

Code:

```
import numpy as np.
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
dataset = pd.read_csv ("Naive Bayes.csv")

x= dataset.iloc [:,[0,1]].values
y= dataset.iloc [:,2].values
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x,y, test_si
                                   random_state=0)
from sklearn.preprocessing import StandardScaler

sc_x = StandardScaler ()
x_train = sc_x.fit_transform (x_train)
```

Expt. No. 7 Page No. 15

Expt. Name. Bayesian network model for medical data set Date :

AIM :-

Write a program to construct a Bayesian network considering medical data. Use this model to demonstrate the diagnosis of heart patients using standard Heart Disease data set.

Algorithm :

1. Import the libraries and import the model for Bayesian model
2. print the Sample instances from the dataset
3. Calculate the posterior conditional probability distribution of each of the possible unobserved causes given the observed evidence
4. Calculate probability of HeartDisease given evidence for restecg attributes
5. Calculate probability of HeartDisease given evidence for cp attributes

Dataset:

thalach	oldpeak	age	gender	cp	trestBps	Chol	Fbs	restecg	exang
150	2.3	63	1	1	145	233	1	2	0
108	1.5	67	1	4	160	286	0	2	1
129	2.6	67	1	4	120	229	0	2	1
187	3.5	37	1	3	130	250	0	0	0
172	1.4	41	0	2	130	204	0	2	0

1. Probability of Heart Disease given evidence = restecg.

HeartDisease	Phi (heart Disease)
heart disease (0)	0.1012
heart disease (1)	0.0000
heart disease (2)	0.2392
heart disease (3)	0.2015
heart disease (4)	0.4561.

2). Probability of Heart Disease given Evidence = cp

heart Disease	Phi (Lart Disease)
heart disease (0)	0.3610
heart disease (1)	0.2159
heart disease (2)	0.1373
heart disease (3)	0.1537
heart disease (4)	0.1321

Expt. No. 8 Page No. 51

Expt. Name. EM Algorithm & K-Means Algorithm. Date :

AIM:

Apply EM algorithm to cluster a set of data set for clustering using k-Means algorithm. Compare the results of the two algorithm and comment on the quality of clustering. You can add Java / Python ML Library classes / API in the program.

ALGORITHM:

Step 1: Import the libraries & include the model for K-means and gaussian mixture

Step 2: Label the attributes

Step 3: Generate the colour map

Step 4: Calculate the accuracy & confusion matrix for K-Mean

Step 5: Generate the scatter plot for gaussian mixture model

Step 6. Calculate the accuracy & confusion matrix for GMM.

Output:

The accuracy od score Od K-Mean: 0.24

The confusion matrix of k-Mean:

[[0.50 0]
 [48 0 2]
 [14 0 36]]

The accuracy score od EM: 0.36666666664

The accuracy confusion matrix od EM:

[[50 0 0]
 [0 5 45]
 [0 50 0]]

Expt. No. 9 Page No. 57

Expt. Name. K-Nearest Neighbour. Date : 18/7/22

AIM:

Write a program to implement k-Nearest Neighbour algorithm to classify the iris data set. Print both correct and wrong prediction. Java/Python ML Library classes can be used for this problem.

Algorithm:

1. Import the libraries & model for nearest neighbours.
2. Specify the name for the attributes.
3. Split the training & testing data from the dataset.
4. Specify neighbors count as 5
5. Calculate the confusion matrix, accuracy & classification report
6. Print the calculated values.

OUTPUT:

	Sepal length (cm)	sepal width(cm)	petal length(cm)	petal width(cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

Original label	predicted label	correct / wrong
0	0	correct
0	0	correct
0	0	correct
1	1	correct
2	2	correct
1	1	correct
0	0	correct
2	2	correct
0	0	correct
2	2	correct
1	1	correct
2	2	correct
0	0	correct
0	0	correct
2	2	correct
		correct

Confusion Matrix:

[[7 0 0]
[0 3 0]
[0 0 5]]

Classification Report :

	Precision	recall	f1-score	Support
0	1.00	1.00	1.00	7
1	1.00	1.00	1.00	3
2	1.00	1.00	1.00	5
accuracy			1.00	15
macro avg	1.00	1.00	1.00	15
weighted avg	1.00	1.00	1.00	15

Accuracy of the Classifier is 1.00