06/03/23

Experiment - I

the probability that it is Friday and 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.

Aim:

To find the probability that a student is absent given that is Friday.

Description:

Machine Learning is a method of data analysis that automates analytical model building of dataset Using the implemented algorithms that iteratively learn from data, machine learning allows computers to find hidden insights without being explicitly programmed where to look. Naive Bayes algorithm is one of the most popular machines learning technique.

Baye's Theorem is based on Conditional Probabality and Baye's Rule.

Conditional Probability is just what is the probability that something will happen, given that something else has already happened.

Let us say we have a collection of people. Some of them are singers. They are either male or female. If we select a random sample, what is the probability that this person is a

: Output:

15.0

male and singers? Conditional Probability is the best option here.
We can calculate probability like:

P(Singers | Male) = P(Male) * P(Singers) Male)

Baye's Rule: We can simplify define Bayes Rule like this. Let $A_1, A_2, \ldots, A_n$ be a set of manually exclusive events that together from the same space 'S'. Let 'B' be any event from the same sample space, such that $P(B) > 0$. Then, $P(A_K|B) = \frac{P(A_K \cap B)}{P(A_1 \cap B) + \cdots + P(A_n \cap B)}$

Bayes Classifier: Navies Bayes classifiers are a family of simple probabilistic classifiers based on applying Baye's Theorem with strong (navie) independence assumptions between the features in machine learning.

Basically we can use above theories and equations for classification Problem.

Source Code:

:: Program:

```
prob_abs_friday = 0.03
prob_friday = 0.2
per_bayes_result = (prob_abs_friday)/(prob_friday) * 100
print(per_bayes_result)
```

CMR GROUP OF INSTITUTIONS

	Time	Weather	Temperature	Company
0	Morning	Sunny	Warm	Yes
1	Evening	Rainy	Cold	No
2	Morning	Sunny	Moderate	Yes
3	Evening	Sunny	Cold	Yes

	Humidity	Wind	Goes
0	Mild	Strong	Yes
1	Mild	Normal	No
2	Normal	Normal	Yes
3	High	Strong	Yes

The attributes are : [['Morning' 'Sunny' 'Warm' 'Yes' 'Mild' 'Strong']
['Evening' 'Rainy' 'Cold' 'No' 'Mild' 'Normal']
['Morning' 'Sunny' 'Moderate' 'Yes' 'Normal' 'Normal']
['Evening' 'Sunny' 'Cold' 'Yes' 'High' 'Strong']]

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

Implementation of 'Find-S' algorithm using Python

: dataset [dF.csv]

```
Time,Weather,Temperature,Company,Humidity,Wind,Goes
Morning,Sunny,Warm,Yes,Mild,Strong,Yes
Evening,Rainy,Cold,No,Mild,Normal,No
Morning,Sunny,Moderate,Yes,Normal,Normal,Yes
Evening,Sunny,Cold,Yes,High,Strong,Yes
```

: Program [Find-S.py]

```
import pandas as pd
import numpy as np
#df = pd.csv
df = pd.read_csv("dF.csv")
print (df)
attr = np.array(df)[:,:-1]
print ("The attributes are :", attr)
target = np.array(df)[:,-1]
print ("The target is: ", target)
```

CMR GROUP OF INSTITUTIONS
STORE TO INVENT

```
def train_Find_S(a,t):
    for i,v in enumerate(t):
        if(v == "Yes"):
            specific_hypothesis = a[i].copy()
            break
    for i,v in enumerate(a):
        if(t[i] == "Yes"):
            for x in range(len(specific_hypothesis)):
                if(val[x] != specific_hypothesis[x]):
                    specific_hypothesis[x] = '?'
                else:
                    pass
    return specific_hypothesis
print("The final hypothesis is:", train_Find_S(attx, target))
```

CMR GROUP OF INSTITUTIONS EXPLORE TO INVENT

Experiment - 2

Extract the data from database using python.

Aim:

To extract the data from database using Python.

Description:

i. Connect to MySQL from Python.

- Refer to Python MySQL database connection to connect to MySQL database from Python using MySQL connector module.

ii. Define a SQL 'Select' Query.

- Next, prepare a SQL 'Select' query to fetch rows from a table. You can select all or limited rows based on the requirement. If the where condition is used, then it decides the no' of rows to fetch. For example, 'select' col1, col2, ..., colN from MySQL_table where ... id=10;". This will return row number 10.

iii. Get Cursor object from Connection.

- Next, use a 'connection.cursor()' method to create a cursor object. This method create a new 'MySQLCursor' object. Cursor.

iv. Execute the 'Select' query using execute() method:

- Execute the Select query using the 'cursor.execute()' method.

v. Extract all rows from a result.

- After successfully executing a Select operation, use the 'fetchall()' method of a cursor object to get all rows from a query

result, it returns a list of rows

vi. Iterate each row.

- Iterate a row list using a for loop and access each row individually. (Access each row column data using a column name or index number)

vii. Close the cursor object and database connection object.

- use 'cursor.close()' and 'connection.close()' method to close open connections after your work completes.

Source Code:

: Program :

```
import mysql.connector
from mysql.connector import Error.
try:
    mydb = mysql.connector.connect(host='localhost',user="root",
    ...password="root@mysql")
    cur = mydb.cursor()
    cur.execute("Create DataBase If Not Exists ml_exp02")
    cur.execute("Create Table If Not Exists ml_exp02
    ... (Id Int Auto_Increment PrimaryKey, Name Int Varchar(255),
    ... Admission_No Varchar(255), Department Varchar(255))")
    cur.execute("Insert Into ml_exp02
    ... (Name, Admission_No, Department) Values (%s, %s, %s)",
    ... ('ABC', '010203', 'CSE(DS)'))
    cur.execute("Insert Into ml_exp02
    ... (Name, Admission_No, Department) Values (%s, %s, %s)",
    ... ('LMN', '121314', 'CSE(DS)'))
```

:: Output :

```
------
Id : 1
Name : ABC
Admission_No : 010203
Department : CSE(DS)
------
Id : 2
Name: LMN
Admission_No : 121314
Department : CSE(DS)
------
Id : 3
Name: XYZ
Admission_No : 242526
Department : CSE(DS)
------
```



```
cur.execute("Insert Into ml_expo2
... (Name, Admission_No, Deparment) Values (%s, %s, %s)",
... ('XYZ', '242526', 'CSE(DS)'))
cur.excute("Select * From ml_expo2")
qry = cur.fetchall()
for i in qry:
    print('------')
    print(f'Id : {i[0]}')
    print(f'Name: {i[1]}')
    print(f'Admission : {i[2]}')
    print(f'Deparment: {i[3]}')
    print('------')
cur.execute("Drop Table ml_expo2")
cur.execute("Drop DataBase ml_expo2")
except Error as e:
    print("oops!! something went wrong...")
    print(f'log :: {e}')
finally:
    if mydb.is_connected():
        mydb.close()
```

CMR GROUP OF INSTITUTIONS
WELCOME TO INVENT

Experiment - 3

Implement K-Nearest neighbors classification using Python.

Aim:

To implement K-Nearest Neighbors Classification using Python.

Theory:

i) K-Nearest Neighbors is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining and intrusion detection.

ii) It is widely disposable in real-life scenarios since it is non-parametric, meaning it doesn't make any underlying assumptions about the distribution of data.

iii) Algorithm:

↳ Input: Let 'm' be the number of training data samples. Let 'p' be an unknown point.

Method:

i) Store the training samples in an array of data points arr[]. This means each element of this array represents a tuple (x,y)

ii) for i=0 to m

Calculate Euclidean distance d(arr[i], p)

iii) Make set S of K smallest distances obtained. Each of these distances correspond to an already classified data limit point.

iv) Return the majority label among 'S'.

Output:

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

Program:

```
From sklearn.neibors import KNeibors
from sklearn.model_selection import train_test_split
From sklearn.datasets import load_iris
irisData = load_iris()
X = irisData.data
y = irisData.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
... random_state = 42)
Knn = KNeighborsClassifier(n_neighbors=7)
Knn.fit(X_train, y_train)
print(Knn.predict(X_test))
```

CMR
GROUP OF INSTITUTIONS
EXPLORE TO INVENT

Experiment - 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)

::ml_exp04.txt

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

Aim:

To predict a classification for a case where VAR1 = 0.906 and VAR2 = 0.606, using the result of Kmeans clustering with 3 means and given data.

CMR GROUP OF INSTITUTIONS EXPLORE TO INVENT

: Output:

[2 1 1 2 0 2 0 1 0]

[0]

Theory:

i) Python code snippet demonstrates the implementation of a simple K-Means clustering to automatically divide input data into groups based on given features.

ii) ";" seperated csv file is loaded first, which contains three corresponding input columns

iii) K-Means clustering model is created from this input data. Afterwards, new data can be classified using the predict() method based on the learned model

iv) The skit-learn and the Pandas library to be installed.

... pip install sklearn

... pip install pandas

Program:

```
from sklearn.cluster import KMeans
import pandas as pd
input_data = pd.read_csv("mlexp04.txt",sep=";")
#print(input_data.to_string())
Knmeans = KMeans(n_clusters=3,n_init='auto')
Kmeans.fit(input_data.values)
print(Kmeans.labels_)
predicted_class = Kmeans.predict([[0.906,0.606,1]])
print(predicted_class)
```