**PROGRAM.NO.6 (Naive bayes Classifier)**

**Write a program to implement the naïve Bayesian classifier for a sample training data set stored as a .CSV file. Compute the accuracy of the classifier, considering few test data sets.**

import pandas as pd

import numpy as np

mush = pd.read\_csv("nnbss.csv")

mush = mush.replace('?', np.nan)

mush.dropna(axis=1, inplace=True)

target = 'class'

features = mush.columns[mush.columns != target]

target\_classes = mush[target].unique()

test = mush.sample(frac=.3)

mush = mush.drop(test.index)

cond\_probs = {}

target\_class\_prob = {}

for t in target\_classes:

mush\_t = mush[mush[target] == t][features]

target\_class\_prob[t] = float(len(mush\_t) / len(mush))

class\_prob = {}

for col in mush\_t.columns:

col\_prob = {}

for val, cnt in mush\_t[col].value\_counts().iteritems():

pr = cnt/len(mush\_t)

col\_prob[val] = pr

class\_prob[col] = col\_prob

cond\_probs[t] = class\_prob

def calc\_probs(x):

probs = {}

for t in target\_classes:

p = target\_class\_prob[t]

for col, val in x.iteritems():

try:

p \*= cond\_probs[t][col][val]

except:

p = 0

probs[t] = p

return probs

def classify(x):

probs = calc\_probs(x)

max = 0

max\_class = ''

for cl, pr in probs.items():

if pr > max:

max = pr

max\_class = cl

return max\_class

b = []

for i in mush.index:

b.append(classify(mush.loc[i, features]) == mush.loc[i, target])

print(sum(b), "correct of", len(mush))

print("Accuracy:", sum(b)/len(mush))

# Test data

b = []

for i in test.index:

b.append(classify(test.loc[i, features]) == test.loc[i, target])

print(sum(b), "correct of", len(test))

print("Accuracy:", sum(b)/len(test))

Dataset file name (**nnbss.csv**)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **class** | **cap-shape** | **cap-surface** | **cap-color** | **bruises** |
| 1 | 1 | 1 | 1 | 5 |
| 1 | 1 | 1 | 2 | 5 |
| 2 | 1 | 1 | 2 | 10 |
| 3 | 2 | 1 | 1 | 10 |
| 3 | 3 | 2 | 2 | 5 |
| 2 | 2 | 2 | 2 | 10 |
| 1 | 2 | 1 | 1 | 5 |
| 1 | 3 | 2 | 1 | 10 |
| 3 | 2 | 2 | 2 | 10 |
| 1 | 2 | 2 | 2 | 10 |
| 2 | 2 | 1 | 2 | 10 |
| 2 | 1 | 2 | 1 | 10 |
| 3 | 2 | 1 | 2 | 5 |
| 1 | 2 | 1 | 2 | 10 |
| 1 | 2 | 1 | 2 | 5 |

Output

6 correct of 11

Accuracy: 0.5454545454545454

1 correct of 4

Accuracy: 0.25