Ahsanullah University of Science & Technology

Department of Computer Science & Engineering

AI LAB POJECT 3

Artificial Intelligence Lab

CSE-4108

Implement a Naive Bayes Classifier algorithm for a given dataset to predict data

Submitted By

Name: Ashiqur Rahman
ID: 15-02-04-057
Section: A2

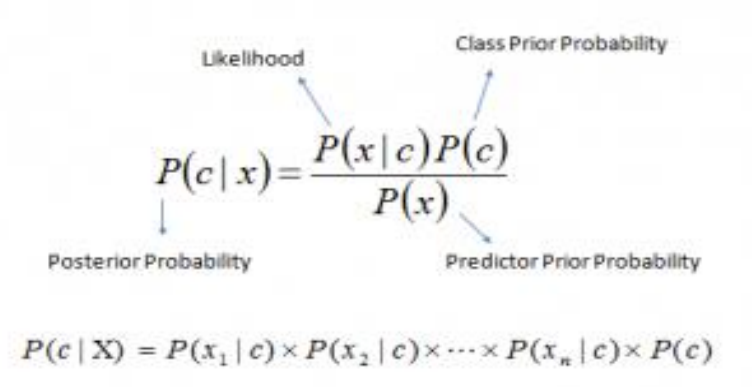
Question : Implement a Naiv Bayes Classifier algorithm for a given dataset to predict data

Answer :

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as 'Naive'.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:



$$P(c \mid x) = \frac{P(x \mid c) P(c)}{P(x)}$$

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Above,

- *P*(*c*|*x*) is the posterior probability of *class* (c, *target*) given *predictor* (x, *attributes*).
- *P*(*c*) is the prior probability of *class*.
- *P*(*x*|*c*) is the likelihood which is the probability of *predictor* given *class*.
- *P*(*x*) is the prior probability of *predictor*.

DATA: Data has been read from csv file.

age	income	student	credit_rating	com
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Input:

Enter the Class :

com = yes

com = no

Data to be Classified X : age = <=30,income = medium,student = yes,credit_rating = fair

Output :

```
P( age = <=30 | com = yes )= 0.2222222222222222

P( income = medium | com = yes )= 0.4444444444444444
```

P(student = yes | com = yes)= 0.6666666666666666

P(credit_rating = fair | com = yes)= 0.6666666666666666

P(age = <=30 | com = no)= 0.6

P(income = medium | com = no)= 0.4

P(student = yes | com = no)= 0.2

P(credit_rating = fair | com = no)= 0.4

P(X|YES)= 0.04389574759945129

P(X|NO)= 0.019200000000000002

P(YES)= 0.6428571428571429

P(NO)= 0.35714285714285715

P(YES|X)=P(X|YES)*P(YES)

= 0.028218694885361547

P(NO|X)=P(X|NO)*P(NO)

= 0.006857142857142858

P(YES|X) > P(NO|X). So X belongs to YES.

CODE :

```
import csv


def probability(data,arg,arg1):


    prm=arg.split(' = ')

    prm1=arg1.split(' = ')

    key=prm[0].strip()

    key1=prm1[0].strip()

    valu=prm[1].strip()

    valu1=prm1[1].strip()

    count=0

    c=0

    for i in range(len(data[key1])):

      if(data[key1][i]==valu1):

        count=count+1


    for i in range(len(data[key])):


      if((data[key][i])==valu and data[key1][i]==valu1):

        c=c+1

        #print('index:',i+1,' ',data[key][i],' : ',data[key1][i])
        print('P(',arg,'|',arg1,')=',c/count)
        return (c/count)



def main():

    infile=open('data.csv', 'r')

    reader = csv.DictReader(infile)

    data = {}
    for row in reader:
        for header, value in row.items():

            try:
                data[header].append(value)
            except KeyError:
                data[header] = [value]



    print('DATA:')
    print('  ',end=' ')
```

```
for i in data.keys():
    print(i, end='  ')
    clm=i
print()

for i in range(len(data[clm])):
    print(i+1,end='  ')
    for j in data.values():
        print(j[i],end='   ')
    print()

print()
while(1):
    print('Enter the Class :')

    C1=input()
    C2=input()
    C3=C1.split(' = ')
    C4=C2.split(' = ')



    bk=0
    for i in data.keys():
        if(i.strip()==C3[0] and i.strip()==C4[0]):

             bk=1

             break


    if(bk==1):

        break

    else:

         print('Class error.Give a space in both side of \'=\' or check input. ')




count_yes=0

count_no=0

total_elmnt=len(data[C3[0]])


for i in range(len(data[C3[0]])):

    if(data[C3[0]][i]==C3[1]):

            count_yes=count_yes+1


for i in range(len(data[C4[0]])):

    if(data[C4[0]][i]==C4[1]):

            count_no=count_no+1
```

```
try:
    #X =' age = <=30,income = medium,student = yes,credit_rating = fair '
    #X= ' outlook = sunny,temp = hot,humidity = normal,windy = false '
    X=input('Data to be Classified X :')
    print('X -> ',X)

    prm=X.split(',')

    s_y=1
    s_n=1

    print()
    for atr in prm:
        s_y=s_y*probability(data,atr,C1)

    print()
    for atr in prm:
        s_n=s_n*(probability(data,atr,C2))
    print()

    print('P(X|YES)=',s_y)
    print('P(X|NO)=',s_n)
    print()
```

```
        print('P(YES)=',count_yes/total_elmnt)
        print('P(NO)=',count_no/total_elmnt)
        print()
        f_yes=s_y*(count_yes/total_elmnt)
        f_no=s_n*(count_no/total_elmnt)
        print('P(YES|X)=P(X|YES)*P(YES)')
        print('        = ',f_yes)
        print()
        print('P(NO|X)=P(X|NO)*P(NO)')
        print('        = ',f_no)
        print()
        if(f_yes>f_no):
                print('P(YES|X) > P(NO|X). So X  belongs to YES.')
                print()

        else:
                 print('P(NO|X) > P(YES|X). So X  belongs to NO.')
                 print()

    except :
            print()
            print('Error in your Input (X) .Give a space in both side of \'=\' or Check  Attribute.')
main()
```