lab2_naive_bayes

November 15, 2022

NAIVE BAYES CLASSIFIER

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn import preprocessing
from sklearn.metrics import␣
  ↪mean_squared_error,confusion_matrix,recall_score,precision_score
```

[ ]:
```
data=pd.read_csv("C:
  ↪\Coding\ML_python\machine-learning-lab-main\datasets\play_tennis.csv")
X=data.iloc[:,:4]
y=data.iloc[:,-1]
print("the independent variables:\n",X)
print("\nthe dependent variable:\n",y)
```

```
the independent variables:
     day   outlook  temp humidity
0    D1     Sunny   Hot     High
1    D2     Sunny   Hot     High
2    D3  Overcast   Hot     High
3    D4      Rain  Mild     High
4    D5      Rain  Cool   Normal
5    D6      Rain  Cool   Normal
6    D7  Overcast  Cool   Normal
7    D8     Sunny  Mild     High
8    D9     Sunny  Cool   Normal
9   D10      Rain  Mild   Normal
10  D11     Sunny  Mild   Normal
11  D12  Overcast  Mild     High
12  D13  Overcast   Hot   Normal
13  D14      Rain  Mild     High

the dependent variable:
 0      No
1      No
2     Yes
```

```
3     Yes
4     Yes
5      No
6     Yes
7      No
8     Yes
9     Yes
10    Yes
11    Yes
12    Yes
13     No
Name: play, dtype: object
```

```
[ ]: le = preprocessing.LabelEncoder()
     for i in range(4):
       X.iloc[:,i]=le.fit_transform(X.iloc[:,i])
     y=le.fit_transform(y)
```

```
C:\Users\gpega\AppData\Local\Temp\ipykernel_24040\3345847121.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  X.iloc[:,i]=le.fit_transform(X.iloc[:,i])
C:\Users\gpega\AppData\Local\Temp\ipykernel_24040\3345847121.py:3:
FutureWarning: In a future version, `df.iloc[:, i] = newvals` will attempt to
set the values inplace instead of always setting a new array. To retain the old
behavior, use either `df[df.columns[i]] = newvals` or, if columns are non-
unique, `df.isetitem(i, newvals)`
  X.iloc[:,i]=le.fit_transform(X.iloc[:,i])
```

```
[ ]: xtrain, xtest, ytrain, ytest = train_test_split(X, y, test_size=0.3,␣
       ↪random_state=0)
     print("training dataset:\n",xtrain,"\n",ytrain)
     print("\ntesting dataset:\n",xtest,"\n",ytest)
```

```
training dataset:
     day  outlook  temp  humidity
13    5        1     2         0
9     1        1     2         1
1     6        2     1         0
7    12        2     2         0
10    2        2     2         1
3     8        1     2         0
0     0        2     1         0
5    10        1     0         1
12    4        0     1         1
```

```
 [0 1 0 0 1 1 0 0 1]

testing dataset:
     day  outlook  temp  humidity
8    13        2     0         1
6    11        0     0         1
4     9        1     0         1
11    3        0     2         0
2     7        0     1         0
 [1 1 1 1 1]
```

[ ]:

```
gaus= GaussianNB()
model= gaus.fit(xtrain, ytrain)
ypred=model.predict(xtest)
print("the predicted y values: ",ypred)
print("the test y values: ",ytest)
print("the  root mean squared error of the dataset is:␣
 ↪",mean_squared_error(ytest,ypred,squared=False))
mat=confusion_matrix(ytest,ypred)
print("the confusion matrix is:\n",mat)
```

```
the predicted y values:  [0 0 0 1 1]
the test y values:  [1 1 1 1 1]
the  root mean squared error of the dataset is:  0.7745966692414834
the confusion matrix is:
 [[0 0]
 [3 2]]
```

[ ]:

```
print("the score for the training data set is: ",model.score(xtest,ytest))
print("the recall score: ",recall_score(ytest,ypred))
print("the precision score: ",precision_score(ytest,ypred))
sensitivity = mat[1,1]/(mat[0,0]+mat[1,1])
specificity = mat[1,0]/(mat[1,0]+mat[0,1])
print("specificity: ",specificity)
print("sensitivity: ",sensitivity)
```

```
the score for the training data set is:  0.4
the recall score:  0.4
the precision score:  1.0
specificity:  1.0
sensitivity:  1.0
```