Lab_5

October 31, 2023

Aim: Implement K-Nearest Neighbors algorithm on diabetes.csv dataset. Compute confusion matrix, accuracy, error rate, precision and recall on the given dataset.
Name: Yash Ghorpade
Div: BE-A
Roll No: B211046

[1]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neighbors  import  KNeighborsClassifier
from sklearn.metrics import confusion_matrix, classification_report,␣
 ↪accuracy_score
```

[2]:
```
df=pd.read_csv("diabetes.csv")  #Reading  the  Dataset
df.head()
```

[2]:
```
   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0            6      148             72             35        0  33.6
1            1       85             66             29        0  26.6
2            8      183             64              0        0  23.3
3            1       89             66             23       94  28.1
4            0      137             40             35      168  43.1

   Pedigree  Age  Outcome
0     0.627   50        1
1     0.351   31        0
2     0.672   32        1
3     0.167   21        0
4     2.288   33        1
```

[3]:
```
df.dtypes
```

[3] :
```
Pregnancies        int64
Glucose            int64
BloodPressure      int64
SkinThickness      int64
Insulin            int64
```

```
BMI                 float64
Pedigree            float64
Age                   int64
Outcome               int64
dtype: object
```

[4] :
```
df["Glucose"].replace(0,df["Glucose"].mean(), inplace=True)
df["BloodPressure"].replace(0,df["BloodPressure"].mean(),    inplace=True)
df["SkinThickness"].replace(0,df["SkinThickness"].mean(),    inplace=True)
df["Insulin"].replace(0,df["Insulin"].mean(), inplace=True)
df["BMI"].replace(0,df["BMI"].mean(), inplace=True)
df.head()
```

[4]:
```
   Pregnancies  Glucose  BloodPressure  SkinThickness     Insulin   BMI  \
0            6    148.0           72.0      35.000000   79.799479  33.6
1            1     85.0           66.0      29.000000   79.799479  26.6
2            8    183.0           64.0      20.536458   79.799479  23.3
3            1     89.0           66.0      23.000000   94.000000  28.1
4            0    137.0           40.0      35.000000  168.000000  43.1

   Pedigree  Age  Outcome
0     0.627   50        1
1     0.351   31        0
2     0.672   32        1
3     0.167   21        0
4     2.288   33        1
```

[5] :
```
X = df.iloc[:, :8]
Y = df.iloc[:, 8:]
X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.
 ↪46,random_state=46)
```

[6] :
```
def apply_model(model):#Model to print the scores of various models
    model.fit(X_train,Y_train)
    print("Training score = ",model.score(X_train,Y_train))
    print("Testing score = ",model.score(X_test,Y_test))
    print("Accuracy = ",model.score(X_test,Y_test))
    Y_pred = model.predict(X_test)
    print("Predicted values:\n",Y_pred)
    print("Confusion Matrix:\n",confusion_matrix(Y_test,Y_pred))
    print("Classification Report:\n",classification_report(Y_test,Y_pred))
```

[7] :
```
knn = KNeighborsClassifier(n_neighbors=5) #KNN Model
apply_model(knn)
```

```
Training score =  0.7946859903381642
Testing score =  0.7175141242937854
Accuracy =  0.7175141242937854
Predicted values:
 [0 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 1 1 1 0 0 1 0 1 0 1 0 0 1 0
 0 0 0 1 1 0 1 0 0 0 1 0 0 1 0 0 0 1 0 1 1 0 0 0 1 0 1 0 0 1 0 0 0 0 0 1 0
 1 0 1 0 0 0 0 0 1 1 0 1 0 1 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
 1 1 1 1 0 0 1 0 0 0 1 0 0 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 0 1 0 1 1 0 1
 1 1 1 0 1 1 1 1 1 1 0 0 0 0 0 1 0 1 1 0 0 1 0 1 1 0 1 1 1 0 1 1 1 1 0 0 0
 1 0 1 0 0 1 1 0 0 0 1 0 0 0 1 0 0 0 1 0 1 0 1 0 0 0 0 0 0 0 0 1 0 1 0 1 0
 0 0 1 0 0 1 1 0 1 0 1 1 0 0 1 0 0 1 0 1 0 0 1 0 1 1 1 0 0 0 0 0 0 1 0 0 1
 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1 0 0 0 0 0 0 0 1 1 0 1 0 0 1 0 1 0 0 1 1 0 0
 1 0 1 1 0 0 1 1 0 1 1 1 0 1 0 0 0 1 0 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0
 0 1 1 0 0 0 0 0 0 1 1 1 0 0 0 0 1 0 1 0 0]
Confusion Matrix:
 [[173  48]
 [ 52  81]]
Classification Report:
              precision    recall  f1-score   support

           0       0.77      0.78      0.78       221
           1       0.63      0.61      0.62       133

    accuracy                           0.72       354
   macro avg       0.70      0.70      0.70       354
weighted avg       0.72      0.72      0.72       354

C:\ProgramData\anaconda3\lib\site-
packages\sklearn\neighbors\_classification.py:215: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape
of y to (n_samples,), for example using ravel().
  return self._fit(X, y)
```

```
[ ]:
```