CSE 4020 - MACHINE LEARNING

Lab 29+30

Lab Task1

Submitted by: Alokam Nikhitha(19BCE2555)

Question:

Build a classifier using decision tree to predict the COVID-19 severity.

Dataset Used:

https://www.kaggle.com/hemanthhari/symptomsand-covid-presence

Procedure:

I. Import the dataset using pandas.
II. Specifying if the attributes to be used as independent attributes or dependent ones
III. We have to change categorical values to numbers
IV. Split our Dataset into training set
V. Create an instance of our decision tree classifier.
VI. We fit our training sets to the object of decision tree classifier in order to train it.
VII. Result is stored in another array
VIII. We create the Confusion Matrix and check performance.

Code:

 # Importing libraries

```
import numpy as np
import pandas as pd
```

 # Importing dataset

```
data = pd.read_csv("Covid Dataset.csv")
X = data.iloc[2500:5500, 0:6].values
y = data.iloc[2500:5500, 20:].values
```

 # Encoding Categorical Attribute

```
from sklearn.preprocessing import OneHotEncoder, LabelEncoder
X[:, 0] = LabelEncoder().fit_transform(X[:, 0])
X[:, 1] = LabelEncoder().fit_transform(X[:, 1])
X[:, 2] = LabelEncoder().fit_transform(X[:, 2])
X[:, 3] = LabelEncoder().fit_transform(X[:, 3])
X[:, 4] = LabelEncoder().fit_transform(X[:, 4])
X[:, 5] = LabelEncoder().fit_transform(X[:, 5])
y[:, 0] = LabelEncoder().fit_transform(y[:, 0])
y=y.astype('int')
```

 # Splitting the dataset into the training set and test set

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=0)
```

 # Fitting Decision Tree Classification model to the training set

```
from sklearn.tree import DecisionTreeClassifier

classifier = DecisionTreeClassifier(criterion = 'entropy', random_state=5)

classifier.fit(X_train, y_train)
```

 # Predicting result for test set

```
y_pred = classifier.predict(X_test)
```

 # Accuracy of our model

```
from sklearn.metrics import accuracy_score

accuracy_score(y_test, y_pred, normalize=True, sample_weight=None)
```

#Classification error

```
1-accuracy_score(y_test, y_pred, normalize=True, sample_weight=None)
```

 # Sensitivity

```
from sklearn.metrics import recall_score
```

recall_score(y_test, y_pred)

 # Precision

from sklearn.metrics import precision_score

precision_score(y_test, y_pred)

#confusion Matrix

from sklearn.metrics import confusion_matrix

confusion_matrix(y_test, y_pred)

```
In [1]: # Importing the libraries
        import numpy as np
        import pandas as pd
```

```
In [2]: # Importing the dataset
        data = pd.read_csv("Covid Dataset.csv")
        X = data.iloc[2500:5500, 0:6].values
        y = data.iloc[2500:5500, 20:].values
```

In Cell 1:

Firstly, we import the libraries numpy as np ,pandas as pd.

In Cell 2:

Dataset is imported from Covid Dataset.csv

We take only first 6 attributes to train our model,

The attributes are Breathing Problem, Fever, Dry Cough, Sore Throat, Running Nose and Asthma. The last column is our label attribute and it tells if a person with given symptoms had COVID-19 or not.

```
In [3]: # Encoding the Categorical Attribute
        from sklearn.preprocessing import OneHotEncoder, LabelEncoder
        X[:, 0] = LabelEncoder().fit_transform(X[:, 0])
        X[:, 1] = LabelEncoder().fit_transform(X[:, 1])
        X[:, 2] = LabelEncoder().fit_transform(X[:, 2])
        X[:, 3] = LabelEncoder().fit_transform(X[:, 3])
        X[:, 4] = LabelEncoder().fit_transform(X[:, 4])
        X[:, 5] = LabelEncoder().fit_transform(X[:, 5])
        y[:, 0] = LabelEncoder().fit_transform(y[:, 0])
        y=y.astype('int')
```

In Cell 3:

We need to Encode the data as it contains yes or No, So we need to encode it into numbers

```
In [4]: # Splitting the dataset into the training set and test set
        from sklearn.model_selection import train_test_split
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

```
In [5]: from sklearn.tree import DecisionTreeClassifier
        classifier = DecisionTreeClassifier(criterion = 'entropy', random_state=5)
        classifier.fit(X_train, y_train)
```

```
Out[5]: DecisionTreeClassifier(criterion='entropy', random_state=5)
```

In Cell 4:

Here we have split our dataset into training set and test set. The training set will be used to train our decision tree classifier and the test will help us analyse the efficacy of trained model.

In Cell 5:

Here we have trained our decision tree classifier with training dataset.

Also, we have set the criterion to entropy and thus the decision tree that we constructed will be based on information gain

```
In [6]: y_pred = classifier.predict(X_test)
```

In Cell 6:

Here we are creating an array and storing the results of X_test dataset as predicted by our classifier.

```
In [7]: # Printing the accuracy of our model
        from sklearn.metrics import accuracy_score
        accuracy_score(y_test, y_pred, normalize=True, sample_weight=None)

Out[7]: 0.8926746166950597
```

In Cell 7: Accuracy of the Model

```
In [8]: # Printing the classification error
        1-accuracy_score(y_test, y_pred, normalize=True, sample_weight=None)

Out[8]: 0.10732538330494035

In [9]: # Printing the Sensitivity
        from sklearn.metrics import recall_score
        recall_score(y_test, y_pred)

Out[9]: 0.957286432160804

In [10]: # Printing the Precision
         from sklearn.metrics import precision_score
         precision_score(y_test, y_pred)

Out[10]: 0.892271662763466
```

In Cell 8,9,10: Printing Classification error,Senstivity,Precision

```
In [11]: #confusion Matrix
         from sklearn.metrics import confusion_matrix
         confusion_matrix(y_test, y_pred)

Out[11]: array([[143,  46],
                [ 17, 381]], dtype=int64)
```

In cell 11:Printing Confusion Matrix

Caluculations:

Our confusion matrix is:

TN 143	FP 46
FN 17	TP 381

- **Accuracy of Model = (TP + TN) / (TP + TN + FP + FN)**
  =(381+143)/(381+143+46+17)

  =0.8926746167(As in Cell 7)

- **Classification Error = 1 – Accuracy**
  =1-0.8926746167

  =0.1073253833(As in Cell 8)

- **Sensitivity = TP / (TP + FN)**
  =381/(381+17)

  =0.95728643216(As in Cell 9)

- **Specificity = TN / (TN + FP)**
  =143/(143+46)

  =0.75661375661

- **Precision = TP / (TP + FP)**
  =381/(381+46)

  =0.89227166276(As in Cell 10)

Final results:

Accuracy =0.8926746167

Classification Error = 0.1073253833

Sensitivity = 0.95728643216

Specificity = 0.75661375661

Precision = 0.89227166276