Harshit Mishra(19BCE0799)

CSE4020 – MACHINE LEARNING

LAB ASSIGNMENT 2

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

Dataset Used: https://www.kaggle.com/hemanthhari/symptoms-and-covid-presence

Procedure:

- We first import the dataset into our workspace using pandas.
- We then specify the attributes to be used as independent attributes and those to be used as dependent attribute.
- Then we encode categorical variables into numbers.
- Then we split our dataset into training set and test set in order to later test the accuracy of our model.
- Then we create an instance of our decision tree classifier.
- We fit our training sets to the object of decision tree classifier in order to train it.
- We then predict the result of our test set and store it in another array.
- Finally, we create the confusion matrix and check the performance of our classifier.

Harshit Mishra(19BCE0799)

Code:

```
# Importing the libraries
import numpy as np
import pandas as pd

# Importing the dataset
data = pd.read_csv("CovidDataset.csv")
X = data.iloc[:, 0:6].values
y = data.iloc[:, 20:].values

# 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')

# 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)

# Printing the accuracy of our model
from sklearn.metrics import accuracy_score
accuracy_score(y_test, y_pred, normalize=True, sample_weight=None)

# Printing the classification error
1-accuracy_score(y_test, y_pred, normalize=True, sample_weight=None)

# Printing the Sensitivity
from sklearn.metrics import recall_score
recall_score(y_test, y_pred)
```

```
# Printing the Precision
from sklearn.metrics import precision_score
precision_score(y_test, y_pred)

# Making confusion Matrix
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test, y_pred)
```

Code Snippet and Explanation:

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

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

Instead of using all the attributes to train our model, we have used only six of them. They 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')
```

Since the attributes contain categorical values of Yes and No, we needed to encode them into numbers for our machine learning model to understand them and thus we have encoded all the categorical attributes 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)
```

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 [5]: # 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)

Out[5]: DecisionTreeClassifier(criterion='entropy', random_state=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]: # Predicting result for test set
        y_pred = classifier.predict(X_test)
```

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

Harshit Mishra(19BCE0799)

```
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]: 0.937442502299908

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

Out[8]: 0.062557497700092

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

Out[9]: 0.9921787709497206

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

Out[10]: 0.9357218124341412

In [12]: # Making confusion Matrix
         from sklearn.metrics import confusion_matrix
         confusion_matrix(y_test, y_pred)

Out[12]: array([[131,  61],
                [  7, 888]], dtype=int64)
```

Here we have printed all the required results of accuracy, classification error, sensitivity, precision and confusion matrix.

Harshit Mishra(19BCE0799)

Result and Conclusion:

- Our confusion matrix is:

131	61
7	888

Thus,
True Positives (TP) = 888
True Negatives (TN) = 131
False Positives (FP) = 61
False Negatives (FN) = 7

- Accuracy of Model = (TP + TN) / (TP + TN + FP + FN)
  = (888 + 131) / (131 + 888 + 61 + 7)
  = 1019 / 1087
  = 0.9374 (same as Out[7])

- Classification Error = 1 – Accuracy
  = 1 – 0.9374
  = 0.06257 (same as Out[8])

- Sensitivity = TP / (TP + FN)
  = 888 / (888 + 7)
  = 888 / 895
  = 0.99217 (same as Out[9])

- Specificity = TN / (TN + FP)
  = 131 / (131 + 61)
  = 131 / 192
  = 0.68229

- Precision = TP / (TP + FP)
  = 888 / (888 + 61)
  = 888 / 949
  = 0.93572 (same as Out[10])

- Final results:

  Accuracy = 0.9374
  Classification Error = 0.0625
  Sensitivity = 0.9921
  Specificity = 0.6822
  Precision = 0.9357