AHSANULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY DHAKA-1208, BANGLADESH.

Department of Computer Science and Engineering
Spring 2019

Program: Bachelor of Science in Computer Science and Engineering
Course No: CSE 4108
Course Title: Artificial Intelligence Lab

Term Project: 03
Topic: K-Nearest Neighbor Classification and Decision Tree Classification

Date of Submission: October 6, 2019

Submitted to:
Dr. S.M. Abdullah Al-Mamun Professor, Department of CSE, AUST.

Md. Siam Ansary
Adjunct Faculty, Department of CSE, AUST.

Submitted By:
Name: Syed Sanzam
ID: 16.01.04.042
Group: A2

Preface:
In machine learning and statistics, classification is a **supervised learning** approach in which the computer program learns from the data input given to it and then uses this learning to classify new observation. It is a tool widely used in the classification in many domains such as in credit approval, medical diagnosis, target marketing etc.

Classification Algorithms:
There are a number of classification algorithms used in various sectors, depending on the nature of a specific task. In this case, the **K-Nearest Neighbor** and **Decision Tree** classification algorithms are applied in order to find a predict a certain result.

1. K-Nearest Neighbor (KNN): A supervised classification algorithm, which takes a bunch of labelled points and uses them to learn how to label other points. To label a new point, it looks at the labelled points closest to that new point (those are its nearest neighbors), and has those neighbors vote, so whichever label the most of the neighbors have is the label for the new point (the "k" is the number of neighbors it checks).

2. Decision Tree: Builds classification models in the form of a tree structure. It breaks down a data set into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node has two or more branches and a leaf node represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.

Tools Used:
- Python 3
- Sci-kit learn
- Pandas
- Numpy
- Matplotlib

Dataset:

LiverPatients.csv
Associated Task: Classification
Number of Instances: 583
Number of attributes: 11
Attribute Information:

Contents:
1. age: Age of the patient in years
2. gender: Patient Gender: Male or Female
3. TB: Total Bilirubin
4. DB: Direct Bilirubin
5. alkphos: Alkaline Phosphotase
6. sgpt: Alamine Aminotransferase
7. sgot: Aspartate Aminotransferase
8. TP: Total Proteins
9. ALB: Albumin
10. A_G: Ratio of Albumin and Globulin
11. class: Predictor Class: 1 if patient has Liver Disease and 2 if they do not

Major Steps of Processing:

1. The dataset is divided into X and Y where X is represented as following columns from the LiverPatients.csv dataset,
   'age','gender','TB','DB','alkphos','sgpt','sgot','TP','ALB','A_G'
   And Y as, *'class'*

2. The dataset is then split into 10 folds (groups) for *Cross Validation*. For each group, take the group as **train dataset** and the remaining groups as the **test dataset.**
3. Fit a model on the training set and evaluate it on the test set
4. Performance Matrices (Accuracy, Mean Absolute Error, Mean Squared Absolute Error and Root Mean Squared Error) have been calculated using the testing data and predicted data for both models.
5. Finally, results have been visualized with a *Grouped Bar Chart* to illustrate the performance of both models.

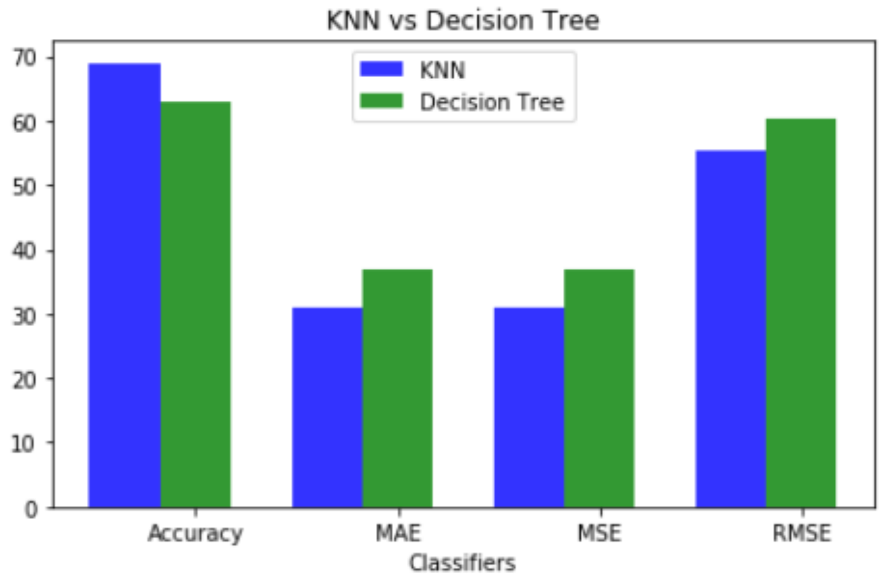
Python Implementation:
kNNVsDecisionTree.py

```python
import pandas as pd
from pandas import DataFrame as df
import numpy as np
import matplotlib.pyplot as plot
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import KFold
from sklearn import metrics
from sklearn.metrics import accuracy_score
from sklearn import tree
from sklearn.preprocessing import LabelEncoder

```

```python
def readDataset():
    global df
    df = pd.read_csv("LiverPatient.csv")
    #print(df.head())

def allocateValues():
    enc = LabelEncoder()
    enc.fit(df['gender'])
    df['gender'] = enc.transform(df['gender'])
    # X = df[['age','gender','TB','DB','alkphos','sgpt','sgot','TP','ALB','A_G']]
    # Y = df['class']
    global X
    global Y
    X = df.iloc[:, [0,9]].values
    Y = df.iloc[:, 10].values

def crossValidation(classifier):
    global acc
    global mae
    global mse
    global rmse

    acc = np.array([ ])
    mae = 0
    mse = 0
    rmse = 0

    ns = 10
    crossValidation = KFold(n_splits=ns)


    for train_index, test_index in crossValidation.split(X):
        X_train, X_test = X[train_index], X[test_index]
        Y_train, Y_test = Y[train_index], Y[test_index]
        classifier.fit(X_train,Y_train)
        prediction = classifier.predict(X_test)
        acc = np.append(acc, [accuracy_score(Y_test, prediction)], axis = 0)

        mae = mae + metrics.mean_absolute_error(Y_test, prediction)
        mse  = mse + metrics.mean_squared_error(Y_test, prediction)
        rmse = rmse + np.sqrt(metrics.mean_squared_error(Y_test, prediction))


    acc = np.mean(acc)
    mae = mae/ns
    mse = mse/ns
    rmse = rmse/ns

    print('Accuracy',acc)
    print('Mean Absolute Error: ',mae)
    print('Mean Squared Error',mse)
    print('Root Mean Squared Error',rmse)
    print('\n')

def testClassifiers():
    knn = KNeighborsClassifier(n_neighbors=20)
    dct = tree.DecisionTreeClassifier()
    print('For KNN : ')
    crossValidation(knn)

    global k_acc
    global k_mae
    global k_mse
    global k_rmse

    global d_acc
    global d_mae
    global d_mse
    global d_rmse

    k_acc = acc
    k_mae = mae
    k_mse = mse
    k_rmse = rmse

    print('For Decision Tree : ')
    crossValidation(dct)

    d_acc = acc
    d_mae = mae
    d_mse = mse
    d_rmse = rmse

def illustrateResult():
    n_groups = 4
    knn_bar = (k_acc*100, k_mae*100, k_mse*100, k_rmse*100)
    dct_bar = (d_acc*100, d_mae*100, d_mse*100, d_rmse*100)

        # create plot
            fig, ax = plot.subplots()
            index = np.arange(n_groups)
            bar_width = 0.35
            opacity = 0.8

            rects1 = plot.bar(index, knn_bar, bar_width,
            alpha=opacity,
            color='b',
            label='KNN')

            rects2 = plot.bar(index + bar_width, dct_bar, bar_width,
            alpha=opacity,
            color='g',
            label='Decision Tree')

            plot.xlabel('Classifiers')
            plot.ylabel('')
            plot.title('KNN vs Decision Tree')
            plot.xticks(index + bar_width, ('Accuracy', 'MAE', 'MSE', 'RMSE'))
            plot.legend()

            plot.tight_layout()
            plot.show()

        #Main
        readDataset()
        allocateValues()
        testClassifiers()
        illustrateResult()
```

Output:

```
For KNN :
Accuracy 0.6893921683226184
Mean Absolute Error:  0.31060783167738165
Mean Squared Error 0.31060783167738165
Root Mean Squared Error 0.5542988792254294


For Decision Tree :
Accuracy 0.6310052600818234
Mean Absolute Error:  0.3689947399181766
Mean Squared Error 0.3689947399181766
Root Mean Squared Error 0.6044338437309891
```



References:

1. https://machinelearningmastery.com/k-fold-cross-validation/
2. https://towardsdatascience.com/machine-learning-classifiers-a5cc4e1b0623
3. https://medium.com/@Mandysidana/machine-learning-types-of-classification-9497bd4f2e14

My GitHub: https://github.com/sanzamsyed/Artificial-Intelligence