Implementation of Neural Network Classification Model

Ex No: 6

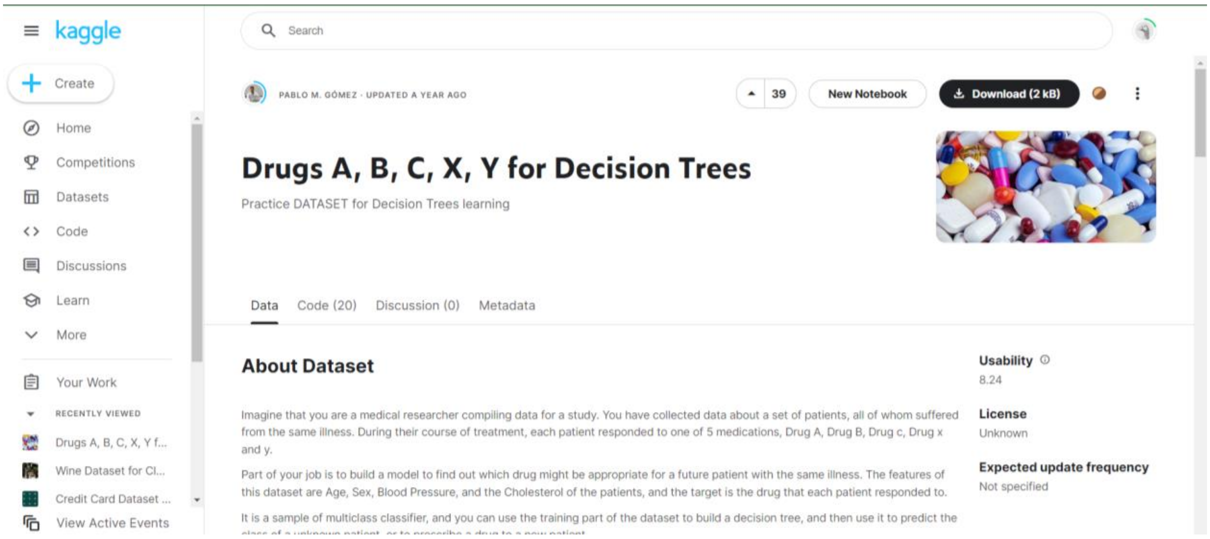
Date 15/9/22

AIM:

The aim of the experiment was to implement Neural Network Classification Model in weka, java using weka and in python using machine learning libraries

DOWNLOADING DATASET:

1. Go to Kaggle (www.kaggle.com) sign in or register new account.
2. After that download the data set you like.



Drugs A, B, C, X, Y for Decision Trees

Practice DATASET for Decision Trees learning

About Dataset

Imagine that you are a medical researcher compiling data for a study. You have collected data about a set of patients, all of whom suffered from the same illness. During their course of treatment, each patient responded to one of 5 medications, Drug A, Drug B, Drug c, Drug x and y.

Part of your job is to build a model to find out which drug might be appropriate for a future patient with the same illness. The features of this dataset are Age, Sex, Blood Pressure, and the Cholesterol of the patients, and the target is the drug that each patient responded to.

It is a sample of multiclass classifier, and you can use the training part of the dataset to build a decision tree, and then use it to predict the class of a unknown patient, or to prescribe a drug to a new patient.

Usability
8.24

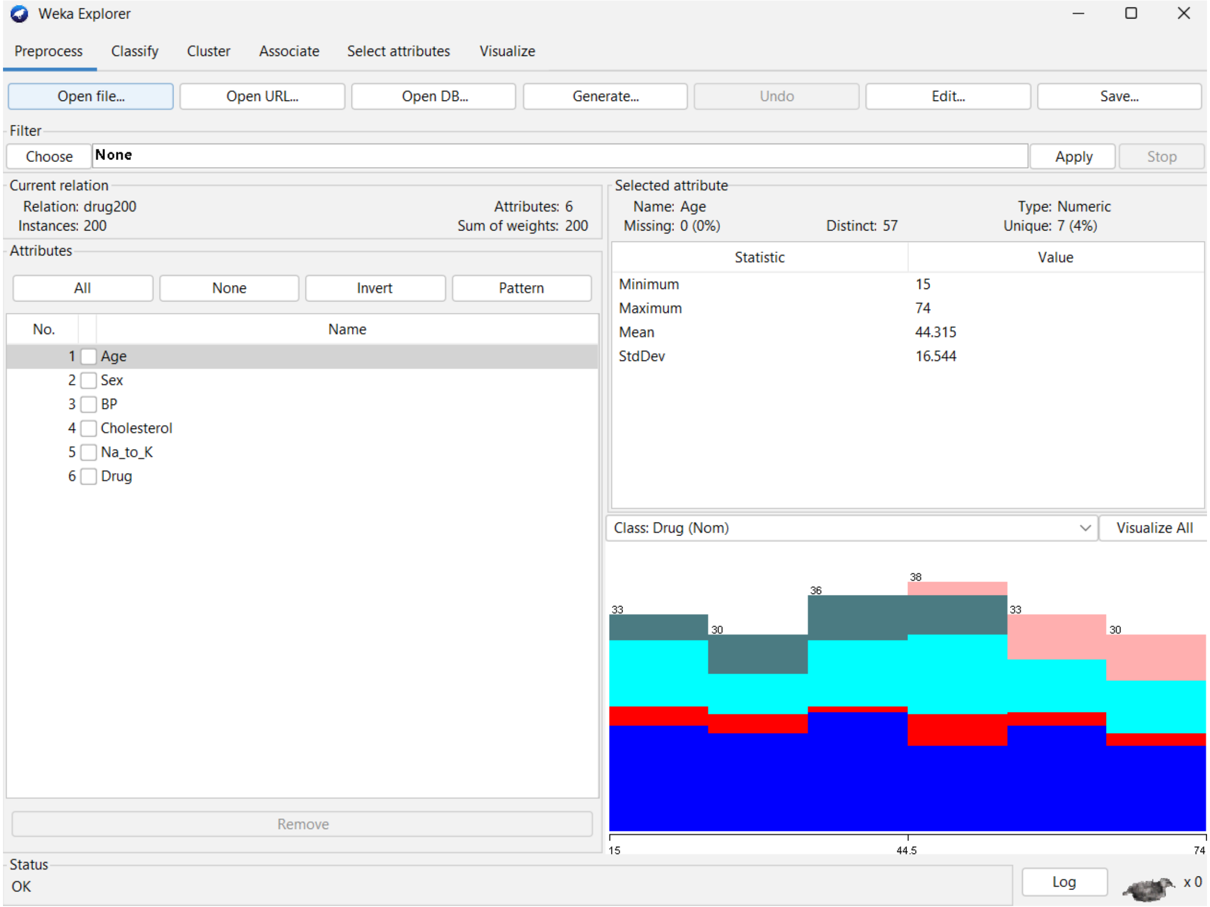
License
Unknown

Expected update frequency
Not specified

3. Click download button to download the dataset.

NEURAL NETWORK IN WEKA:

1. Open weka and click explore.
2. In preprocessing tab click open file button.
3. Select the dateset file you going to perform decision tree.
4. Then click classify tab.
5. Select the classifer by clicking the choose button.
6. Select the Multilayer Perceptron which is present under function.
7. Finally click start. Then it automatically built the model



Filter: None

Current relation
Relation: drug200 Attributes: 6
Instances: 200 Sum of weights: 200

Attributes

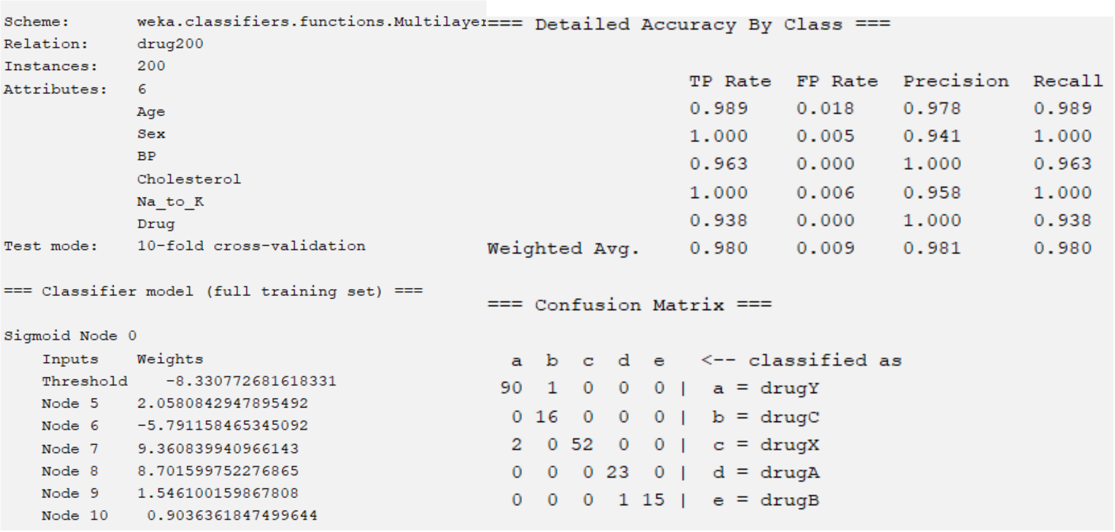
No.	Name
1	□ Age
2	□ Sex
3	□ BP
4	□ Cholesterol
5	□ Na_to_K
6	□ Drug

Selected attribute
Name: Age Type: Numeric
Missing: 0 (0%) Distinct: 57 Unique: 7 (4%)

Statistic	Value
Minimum	15
Maximum	74
Mean	44.315
StdDev	16.544

Class: Drug (Nom)

Status
OK



```
Scheme:       weka.classifiers.functions.Multilayer
Relation:     drug200
Instances:    200
Attributes:   6
              Age
              Sex
              BP
              Cholesterol
              Na_to_K
              Drug
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

Sigmoid Node 0
    Inputs    Weights
    Threshold    -8.330772681618331
    Node 5    2.0580842947895492
    Node 6    -5.791158465345092
    Node 7    9.360839940966143
    Node 8    8.701599752276865
    Node 9    1.546100159867808
    Node 10    0.9036361847499644
```

```
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall
                 0.989    0.018    0.978      0.989
                 1.000    0.005    0.941      1.000
                 0.963    0.000    1.000      0.963
                 1.000    0.006    0.958      1.000
                 0.938    0.000    1.000      0.938
Weighted Avg.    0.980    0.009    0.981      0.980

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 90  1  0  0  0 |  a = drugY
  0 16  0  0  0 |  b = drugC
  2  0 52  0  0 |  c = drugX
  0  0  0 23  0 |  d = drugA
  0  0  0  1 15 |  e = drugB
```

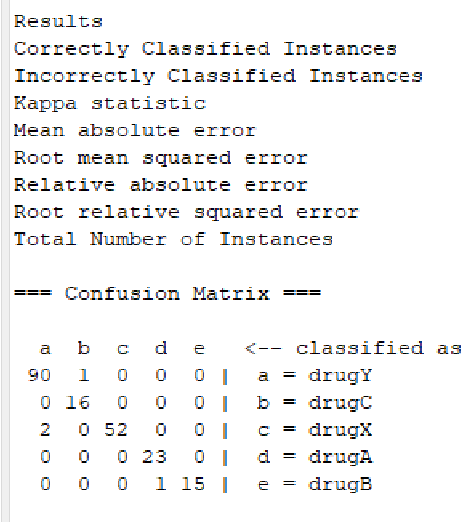
Accuracy and confusion Matrix

NEURAL NETWORK USING JAVA

Code:

```
package Javatree;
import weka.core.Instances;
import weka.classifiers.bayes.NaiveBayes;
import weka.core.converters.CSVLoader;
import java.io.File;
import weka.classifiers.evaluation.Evaluation;
import weka.classifiers.functions.MultilayerPerceptron;

import java.util.Random;
public class Main {
    public static void main(String[] argv) {
        try {
            CSVLoader loader = new CSVLoader();
            String name = "C:\\Users\\kaush\\Desktop\\drug200.csv";
            loader.setSource(new File(name));
//                      BufferedReader dataset = new
BufferedReader(new FileReader(name));
                        Instances drug = loader.getDataSet();
                        System.out.println(drug);
                        MultilayerPerceptron tree = new
MultilayerPerceptron();
                        drug.setClassIndex(
                                drug.numAttributes() - 1);
//                      tree.buildClassifier(drug);
//                      System.out.println(tree);
                        Evaluation evaluation= new
Evaluation(drug);
                        evaluation.crossValidateModel(tree, drug,
10,new Random(1));

    System.out.println(evaluation.toSummaryString("\nResults", false));

    System.out.println(evaluation.toMatrixString());
                    }
                    catch(Exception e) {}

                }
            }
```

Output:



```
Results
Correctly Classified Instances         196               98      %
Incorrectly Classified Instances         4                2      %
Kappa statistic                          0.9712
Mean absolute error                      0.0197
Root mean squared error                  0.0752
Relative absolute error                  7.0503 %
Root relative squared error             20.1696 %
Total Number of Instances              200

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 90  1  0  0  0 |  a = drugY
  0 16  0  0  0 |  b = drugC
  2  0 52  0  0 |  c = drugX
  0  0  0 23  0 |  d = drugA
  0  0  0  1 15 |  e = drugB
```

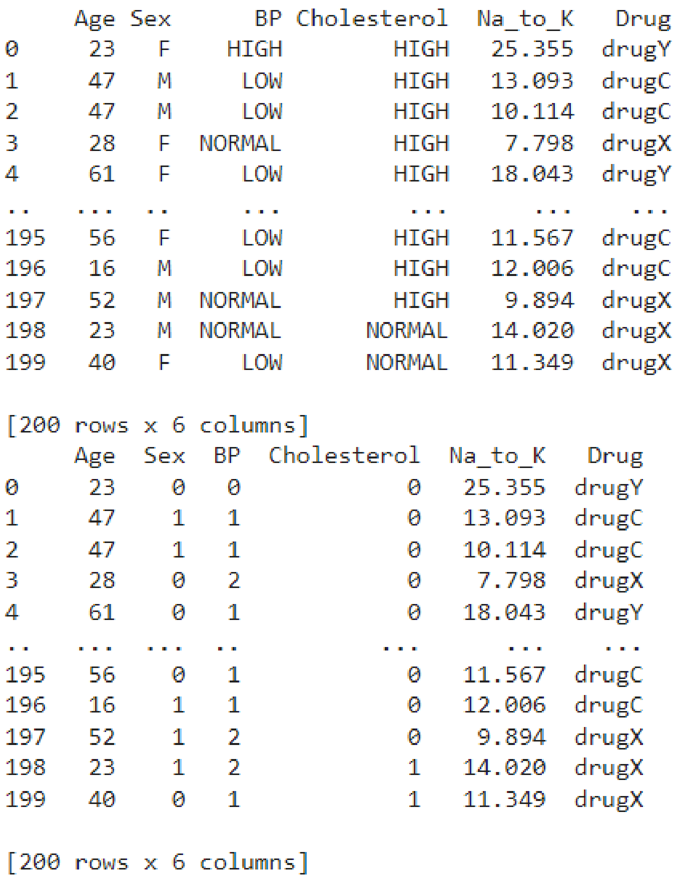
Accuracy and Confusion Matrix

NEURAL NETWORK USING PYTHON:

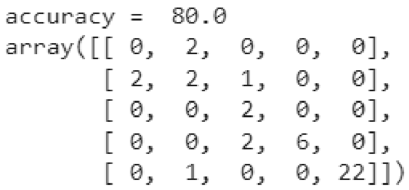
Code:

```
import pandas as pd
import numpy as np
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from matplotlib import pyplot as plt
from sklearn.neural_network import MLPClassifier
drug = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/drug200.csv")
columnName = drug.columns.tolist()
columnName.pop(0)
columnName.pop(-1)
columnName.pop(-1)
inverseLe = {}
for i in columnName:
  le = preprocessing.LabelEncoder()
  le.fit(drug[i].values)
  a = le.transform(drug[i].values)
  drug[i] = a
  inverseLe[i] = le
x = drug.values[:, 0:5]
y = drug.values[:, 5]
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.2, random_state=100)
neuralNetwork = MLPClassifier(solver="lbfgs", alpha=1e-5,
                              hidden_layer_sizes=(5, 2), random_state=1, max_iter=9999)
neuralNetwork.fit(x_train, y_train)
y_pred = neuralNetwork.predict(x_test)
print("accuracy = ", accuracy_score(y_test, y_pred)*100)
confusion_matrix(y_test, y_pred)
```

Output:



```
     Age Sex      BP Cholesterol  Na_to_K   Drug
0     23   F    HIGH        HIGH   25.355  drugY
1     47   M     LOW        HIGH   13.093  drugC
2     47   M     LOW        HIGH   10.114  drugC
3     28   F  NORMAL        HIGH    7.798  drugX
4     61   F     LOW        HIGH   18.043  drugY
..   ...  ..     ...         ...      ...    ...
195   56   F     LOW        HIGH   11.567  drugC
196   16   M     LOW        HIGH   12.006  drugC
197   52   M  NORMAL        HIGH    9.894  drugX
198   23   M  NORMAL      NORMAL   14.020  drugX
199   40   F     LOW      NORMAL   11.349  drugX

[200 rows x 6 columns]
     Age  Sex  BP  Cholesterol  Na_to_K   Drug
0     23    0   0            0   25.355  drugY
1     47    1   1            0   13.093  drugC
2     47    1   1            0   10.114  drugC
3     28    0   2            0    7.798  drugX
4     61    0   1            0   18.043  drugY
..   ...  ...  ..          ...      ...    ...
195   56    0   1            0   11.567  drugC
196   16    1   1            0   12.006  drugC
197   52    1   2            0    9.894  drugX
198   23    1   2            1   14.020  drugX
199   40    0   1            1   11.349  drugX

[200 rows x 6 columns]
```



```
accuracy =  80.0
array([[ 0,  2,  0,  0,  0],
       [ 2,  2,  1,  0,  0],
       [ 0,  0,  2,  0,  0],
       [ 0,  0,  2,  6,  0],
       [ 0,  1,  0,  0, 22]])
```

RESULT:

Successfully we implemented Neural Network classification algorithm in weka gui, java with weka library and using python.