UE17CS303

MACHINE LEARNING ASSIGNMENT

K ANJALI KAMATH

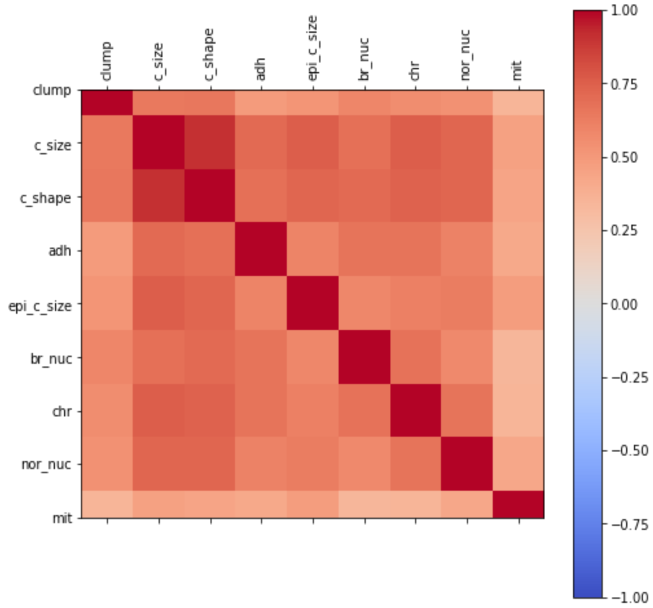
PROBLEM STATEMENT:

On the given dataset (breast-cancer.data) use any ML algorithm for classification.
Use 6 fold cross validation.

DATA CLEANING:

Each column in the dataset was of type ‘str’. All the values were converted to ‘int’ datatype before cleaning. There were 16 instances in groups 1-6 that contained a single missing attribute value.The dataset was cleaned by replacing the missing values(?) in a particular column by the mean of the column.

Correlation matrix:
Here, maroon signifies complete correlation (1.00) , while dark blue signifies no correlation between the attributes (-1.00).

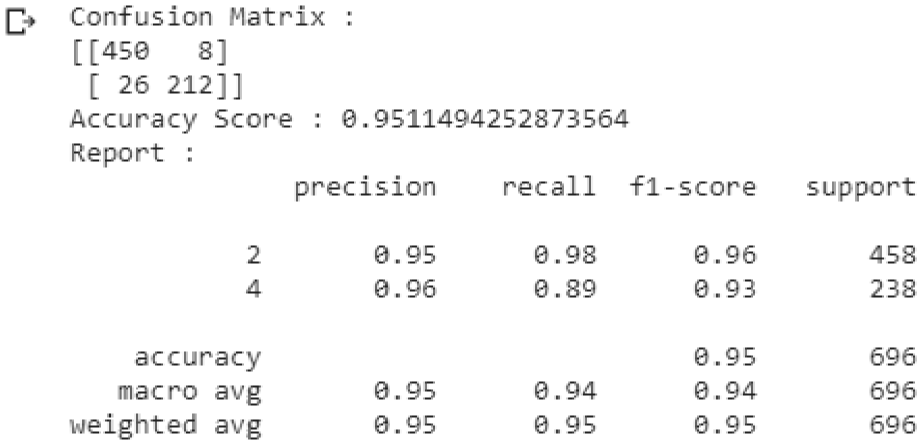


TECHNIQUES EMPLOYED:

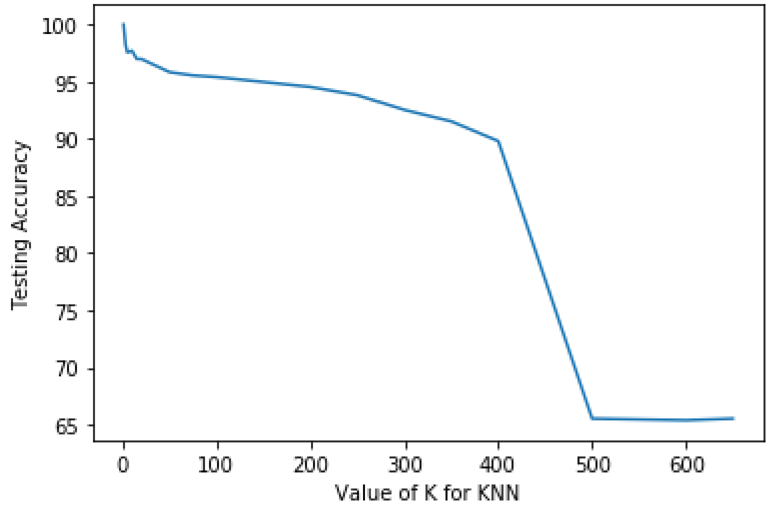
1.K NEAREST NEIGHBOURS (KNN):

KNN or k-Nearest Neighbours is an algorithm which is used for regression and classification. It classifies data based on some similarity measure. Here, we used Euclidean distance as the basis for classifying the data points.

For k=5 we got the accuracy 95.11%



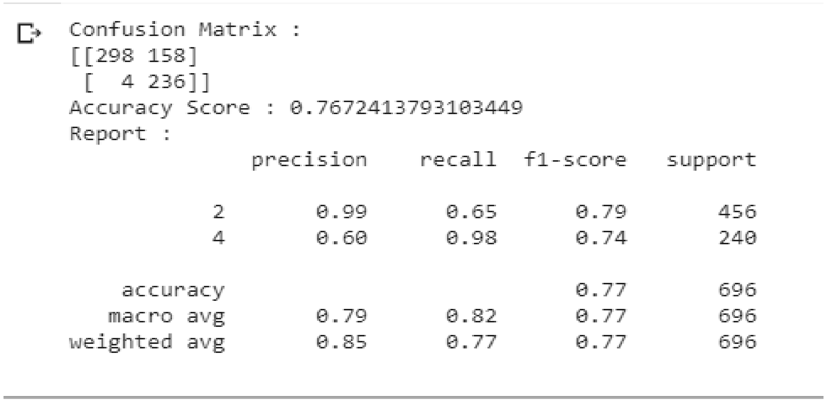
```
Confusion Matrix :
[[450   8]
 [ 26 212]]
Accuracy Score : 0.9511494252873564
Report :
              precision    recall  f1-score   support

           2       0.95      0.98      0.96       458
           4       0.96      0.89      0.93       238

    accuracy                           0.95       696
   macro avg       0.95      0.94      0.94       696
weighted avg       0.95      0.95      0.95       696
```



2.WEIGHTED KNN:

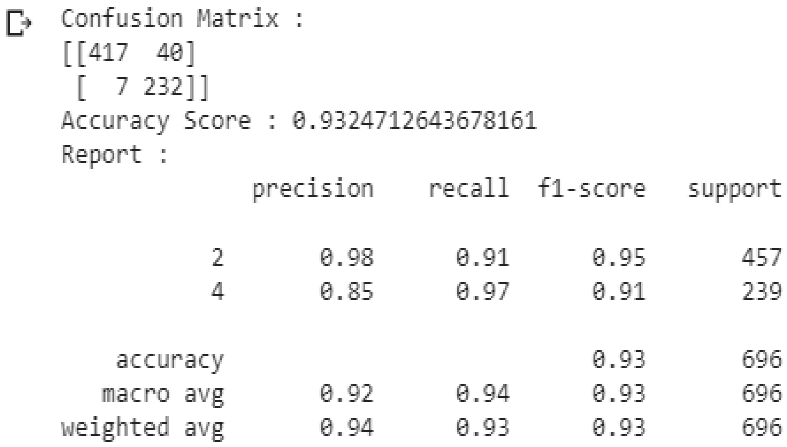
As the accuracy obtained for ordinary knn was very high,we assigned weights to each neighbour ,thereby increasing/decreasing that neighbours effect. The accuracy remains low initially for smaller k values then increases significantly.

For k=10 the accuracy obtained was 76.7%

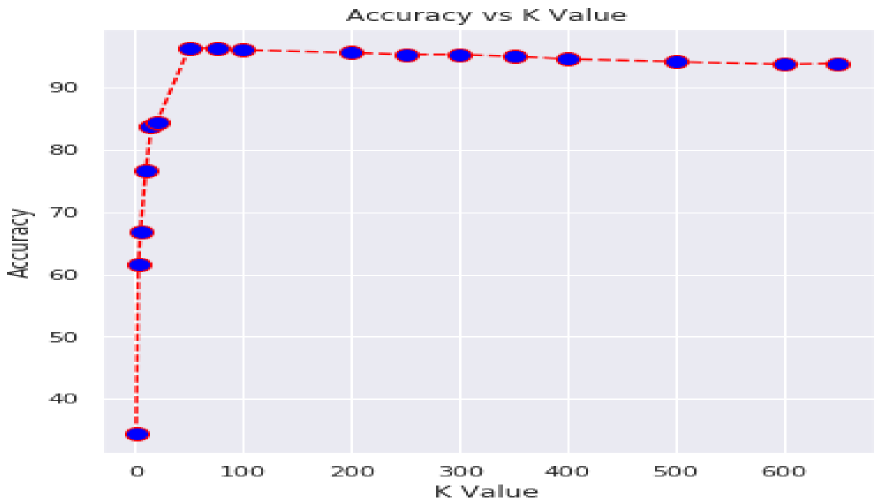


```
Confusion Matrix :
[[298 158]
 [  4 236]]
Accuracy Score : 0.7672413793103449
Report :
              precision    recall  f1-score   support

           2       0.99      0.65      0.79       456
           4       0.60      0.98      0.74       240

    accuracy                           0.77       696
   macro avg       0.79      0.82      0.77       696
weighted avg       0.85      0.77      0.77       696
```

For k=25,the accuracy obtained was 93.2%



```
Confusion Matrix :
[[417  40]
 [  7 232]]
Accuracy Score : 0.9324712643678161
Report :
              precision    recall  f1-score   support

           2       0.98      0.91      0.95       457
           4       0.85      0.97      0.91       239

    accuracy                           0.93       696
   macro avg       0.92      0.94      0.93       696
weighted avg       0.94      0.93      0.93       696
```



3.LOGISTIC REGRESSION:

Logistic model is used to model the probability of a certain class or event existing. Here we use it to predict the class (Benign/Malignant). Data was first normalised.Then the class for each row in the dataset was predicted using the formula given below.(i.e. We used sigmoid function).The predicted class was then rounded off to 2 decimal places and compared with the actual class and the error was calculated.Then the coefficients (b) for every column was updated based on the formula:
b=b+(error*learning_rate*predicted_value*(1-predicted_value)*x

The final predicted class was then compared with the actual class and accuracy was calculated.The maximum accuracy that we could achieve was 83.33% when the learning_rate was 0.93,and the number of epochs was 3000.

$$p = \frac{1}{1 + e^{-(b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_p x_p)}}$$

4.ANN:

Random weights in the range [1-100] were assigned to each input. The weights were then normalized. Then the dot product/linear combination of the inputs and weights was found for each layer in the network(we used only 1 hidden layer). The outputs of one layer become the inputs of the next layer. This is how forward propagation is implemented.The activation function used is reLu (Rectified Linear Unit). The final output is then compared with the expected output and the error is calculated using the formula error=(expected-observed)^2. The total error is found by summing the individual errors of each layer. In back propagation, this error is used to update weights accordingly. A learning rate is specified which is used to update the weights. At each epoch, the epoch number, learning rate and the total error is printed.