Assignment 2 Report
K NEAREST NEIGHBOR
Data Mining

CSE 5334 Section 001

Student Name:
Ahir Kunjal (1001877263)
Sheth Devanshi (1001959019)
Gope Pankaj (1001990547)

Professor: Dr Elizabeth D Diaz

Pre-Processing :-

```
#Module for Pre-Processing
#partitioning of the training dataset
p_india_train_dataset = pima_india_read_dataset.drop(axis=1,columns='class')
#creating the test data set
p_india_testing_dataset = pima_india_read_dataset['class']
#The value of columns belong to different range and hence transforming the values into training dataset
p_india_transform_val = (p_india_train_dataset-p_india_train_dataset.mean())/p_india_train_dataset.std()
#The processed dataset is not joined with variable
p_india_process_dataset = pd.concat([p_india_transform_val,p_india_testing_dataset],axis=1)
print(p_india_process_dataset)

#refs
#https://www.malicksarr.com/data-standardization-with-python/
```

First, the columns except the output variables are separated and converted. The values in all columns are integers, so you should put them all in the same range for better results. Then connect the output features and the converted features.

Parameters of K Nearest Neighbours :-

1. Number of Nearest Neighbours: K-Nearest Neighbors uses a specified number of nearest neighbors to determine the class of an existing data point. This is the most important parameter.
2. Distance: There are several ways to measure the distance of a particular data point from adjacent data points. Euclid, Mahana Robis, Minkowski, Manhattan, etc.
3. Weight: This parameter indicates which other parameters should take precedence in determining which neighbor's majority vote is taken into account.

Criteria for Attribute Selection :-

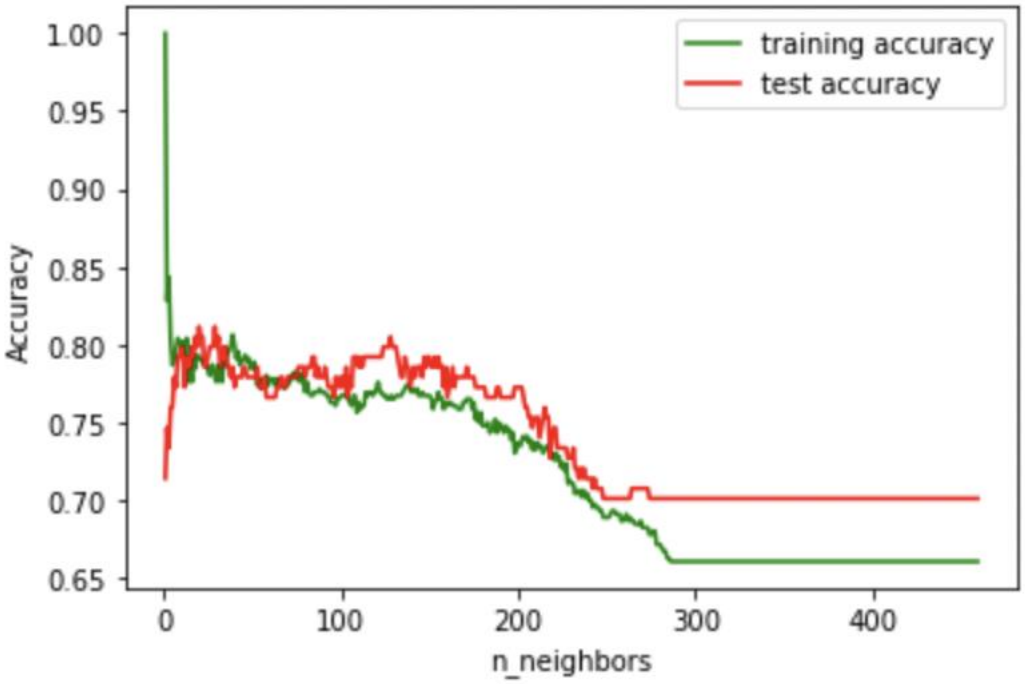
```
#Finding the best 3 attributes using univariate selection
select_univariate_facility = SelectKBest(score_func=f_classif, k='all')
fit = select_univariate_facility.fit(p_india_transform_val,p_india_testing_dataset)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(p_india_transform_val.columns)

#Adding the important scores and attribute name to a dataframe to use the available panda methods
attribute_in_dataframe = pd.concat([dfcolumns,dfscores],axis=1)
attribute_in_dataframe.columns = ['Attribute Name','Importance'] #Providing the column names
#The below command prints the best 3 features based on importance score
print(attribute_in_dataframe.nlargest(3,'Importance'))
print("\nThe best 3 univariate selection attributes are%s: Plas, mass and age.")

#refs
#https://towardsdatascience.com/feature-selection-techniques-in-machine-learning-with-python-f24e7da3f36e
```

To determine the relationship between the characteristics and the output variable, we employed the univariate methodology. To identify the relationship between the characteristics and the output class, statistical approaches are applied. Once this is determined, we choose the top three attributes with the greatest scores to further forecast the class.

- ## Determining the optimal K value :-

```
#finding the best value of K.
#we here select the value of K only when the number of nearest neighbors is more
train_accracy_dataset = []
testing_accracy_dataset = []
# check n neighbors from 1 to the length of the training dataset
check_neighbors = range(1, len(X_training_set))

for n_neighbors in check_neighbors:
    knn = KNeighborsClassifier(n_neighbors=n_neighbors,metric='minkowski')
    knn.fit(X_training_set, y_training_set)
    # here training dataset's accuracy is noted
    train_accracy_dataset.append(knn.score(X_training_set, y_training_set))
    # here generalization accuracy is noted
    testing_accracy_dataset.append(knn.score(X_testing_set, y_testing_set))

plt.plot(check_neighbors, train_accracy_dataset, color="green", label="training accuracy")
plt.plot(check_neighbors, testing_accracy_dataset, color="red", label="test accuracy")
plt.ylabel("Accuracy")
plt.xlabel("n_neighbors")
plt.legend()
plt.show()


print("\n The best test accuracy at the value of K =",testing_accracy_dataset.index(max(testing_accracy_dataset)))
```

We estimated the value of test accuracy for each feasible value of K. We saved the data in a list and calculated the index of the item with the greatest precision. The result was 19.



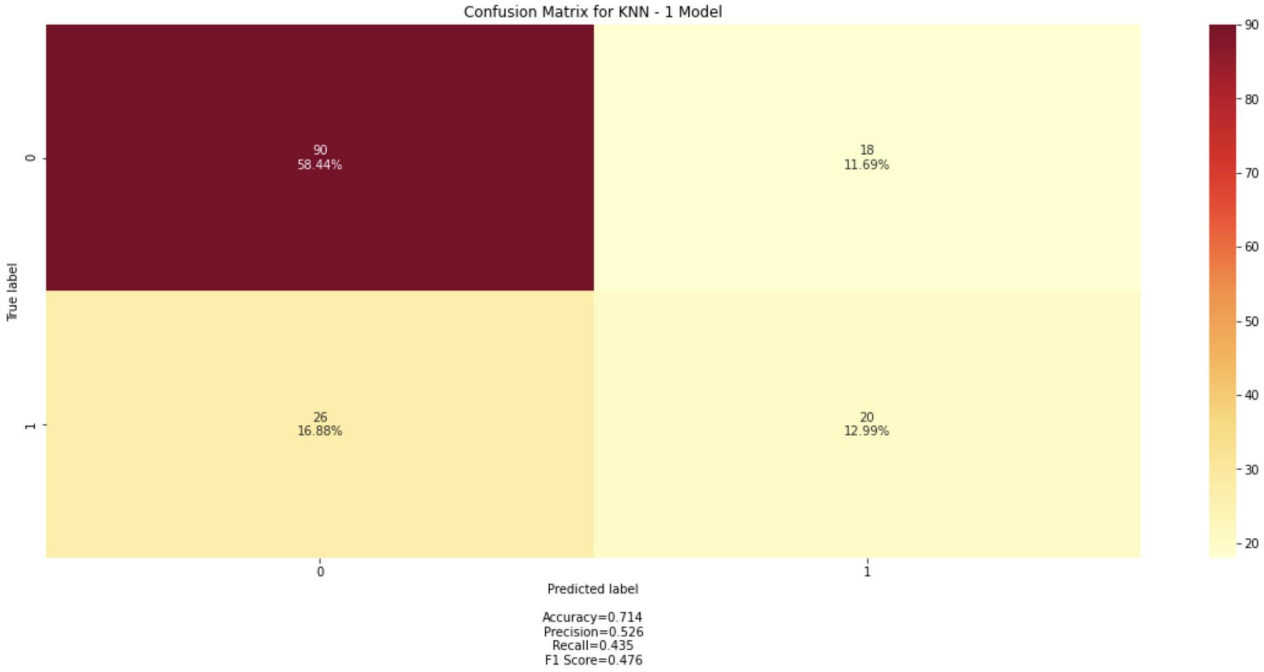

```
The best test accuracy at the value of K = 19
```

In order to properly visualize the confusion matrix, we understood and used the "make_confusion_matrix" function we found from an authoritative source.

- Classifier with value of K = 1 :-

```
#Testing the classifier on three nearest neighbors - 1
knn_for_1 = KNeighborsClassifier(n_neighbors=1,metric='minkowski')
knn_for_1.fit(X_training_set, y_training_set)
predict_knn_for_1 = knn_for_1.predict(X_testing_set)
matrix_for_knn_1 = confusion_matrix(y_testing_set, predict_knn_for_1)
categories = ['0','1']
func_confuse_matrix(matrix_for_knn_1,
                    figsize = (20,8),
                    categories=categories,
                    cmap='YlOrRd',
                    cbar =True,
                    title= 'Confusion Matrix for KNN - 1 Model')
print('\nThe classification report is provided below: \n',classification_report(y_testing_set,predict_knn_for_1))
```

```
The classification report is provided below:
              precision    recall  f1-score   support

           0       0.78      0.83      0.80       108
           1       0.53      0.43      0.48        46

    accuracy                           0.71       154
   macro avg       0.65      0.63      0.64       154
weighted avg       0.70      0.71      0.71       154
```



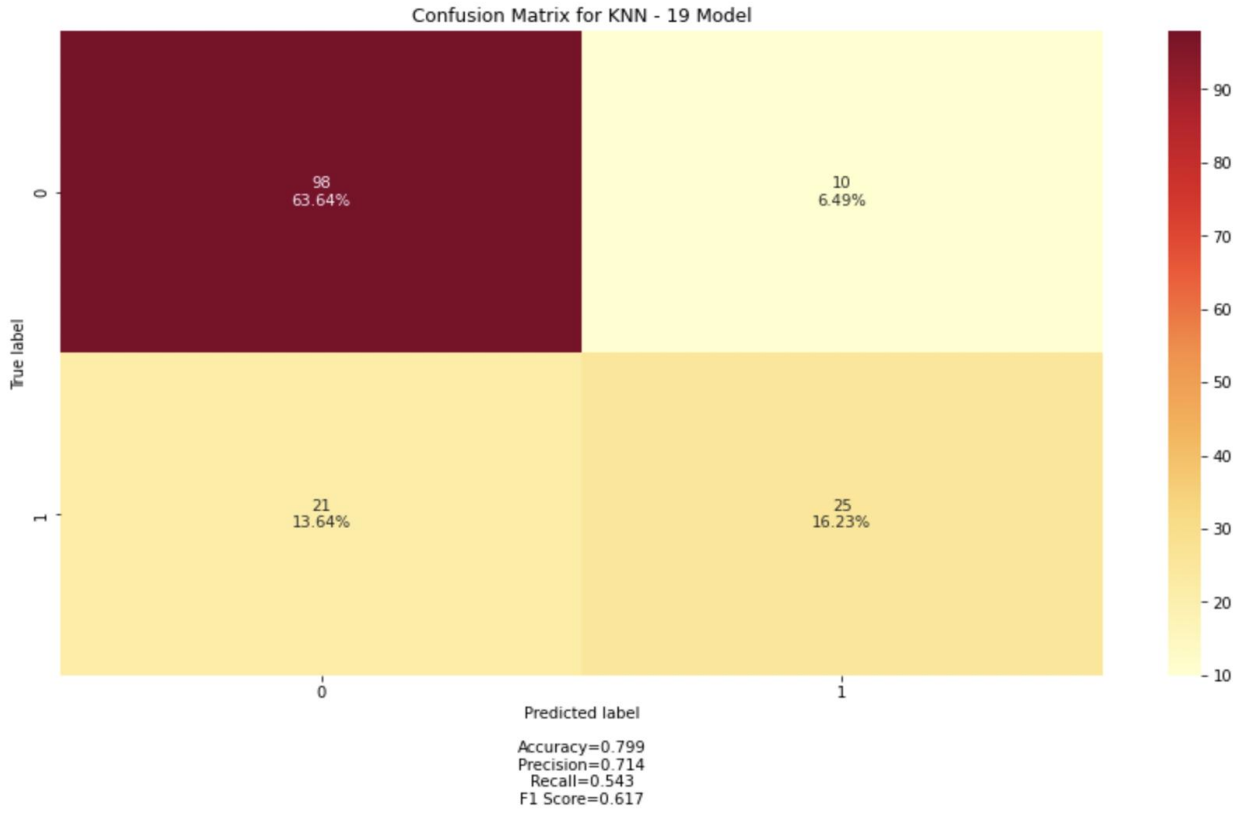
It can be seen that the model has an accuracy of about 71%, which can be well estimated with the current Confusion Matrix. If the diagonal blocks are dark red, the accuracy is higher.

- Classifier with value of K = 19 :-

```
#Testing the classifier on three nearest neighbors - 19
knn_for_19 = KNeighborsClassifier(n_neighbors=19,metric='minkowski')
knn_for_19.fit(X_training_set, y_training_set)
predict_knn_for_19 = knn_for_19.predict(X_testing_set)
matrix_for_knn_19 = confusion_matrix(y_testing_set, predict_knn_for_19)
categories = ['0','1']
func_confuse_matrix(matrix_for_knn_19,
                    figsize = (15,8),
                    categories=categories,
                    cmap='YlOrRd',
                    cbar =True,
                    title= 'Confusion Matrix for KNN - 19 Model')
print('\nThe classification report is provided below: \n',classification_report(y_testing_set,predict_knn_for_19))
```

The classification report is provided below:

	precision	recall	f1-score	support
0	0.82	0.91	0.86	108
1	0.71	0.54	0.62	46
accuracy			0.80	154
macro avg	0.77	0.73	0.74	154
weighted avg	0.79	0.80	0.79	154

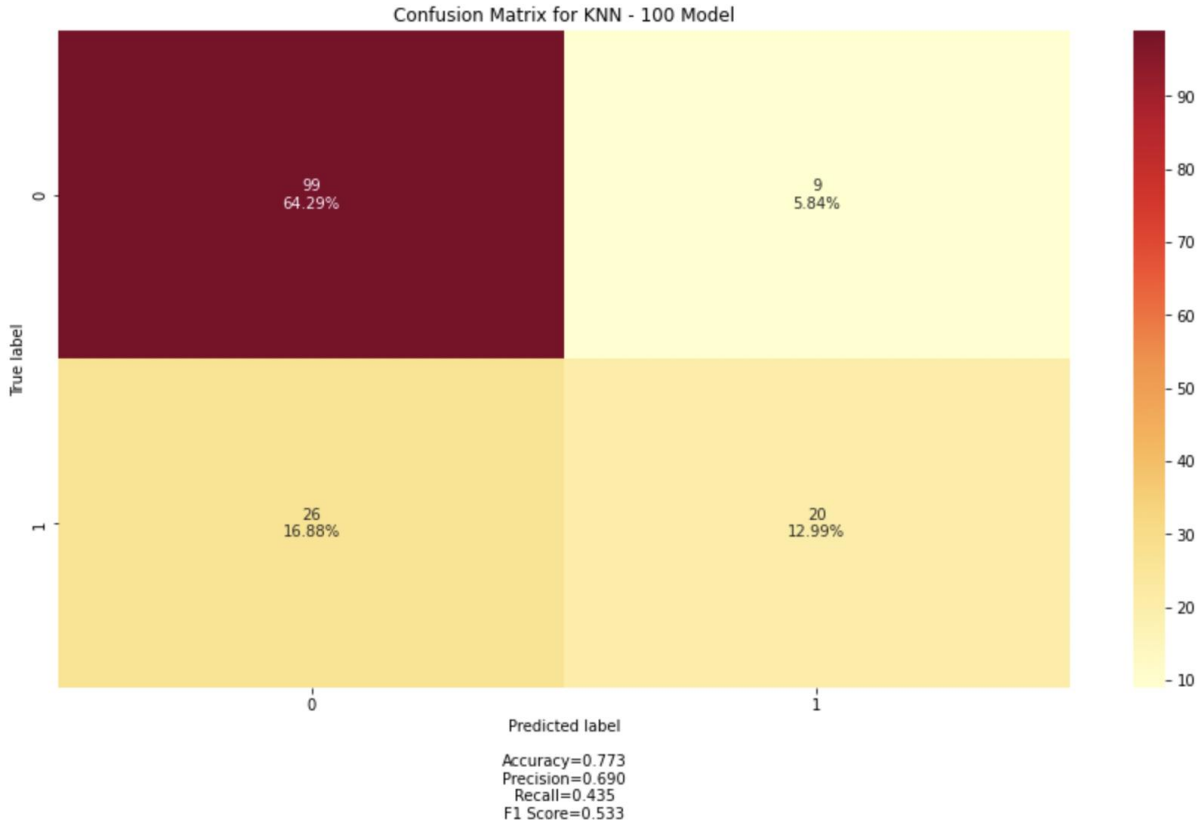


We mentioned earlier that a K value of 19 gives the highest test accuracy. Here we can see that the accuracy is almost 80%. This is the highest K value. Diagonal boxes also tend to be dark red.

- Classifier with value of K = 100 :-

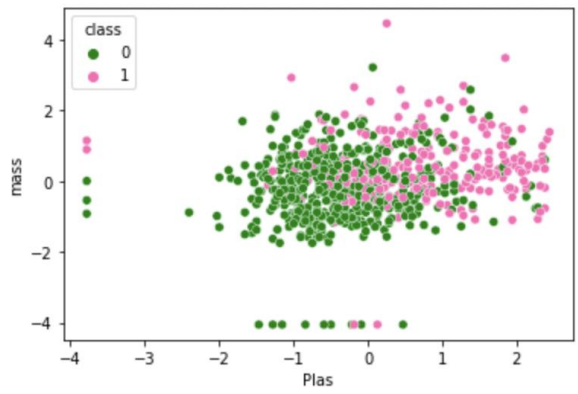
```
#Testing the classifier on three nearest neighbors - 100
knn_for_100 = KNeighborsClassifier(n_neighbors=100,metric='minkowski')
knn_for_100.fit(X_training_set, y_training_set)
predict_knn_for_100 = knn_for_100.predict(X_testing_set)
matrix_for_knn_100 = confusion_matrix(y_testing_set, predict_knn_for_100)
categories = ['0','1']
func_confuse_matrix(matrix_for_knn_100,
                    figsize = (15,8),
                    categories=categories,
                    cmap='YlOrRd',
                    cbar =True,
                    title= 'Confusion Matrix for KNN - 100 Model')
print('\nThe classification report is provided below: \n',classification_report(y_testing_set,predict_knn_for_100))
```

```
The classification report is provided below:
              precision    recall  f1-score   support

           0       0.79      0.92      0.85       108
           1       0.69      0.43      0.53        46

    accuracy                           0.77       154
   macro avg       0.74      0.68      0.69       154
weighted avg       0.76      0.77      0.76       154
```

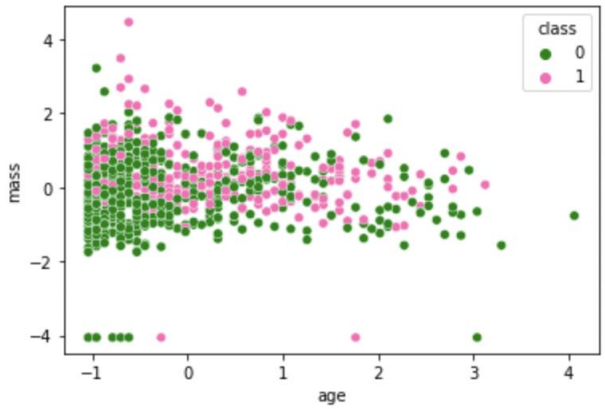


You can see that the accuracy drops significantly from 80% to 77%. This is due to the change in the number of nearest neighbors. This reinforces that K = 19 provides the highest test accuracy.

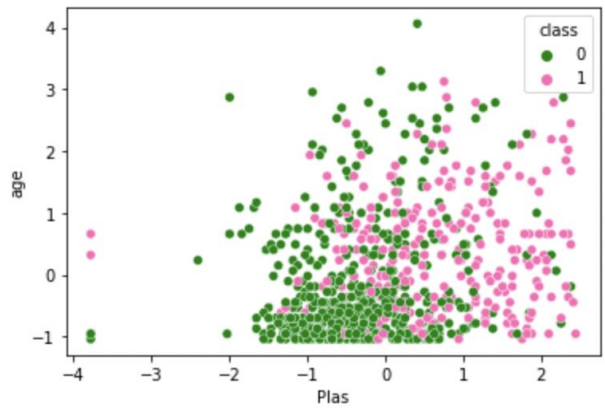
- ## Visualization of the output variable with the 3 selected features :-



If the value of plas is small, the class is usually 0. The majority of class values 1 are in the -2 to 2 age range. The dataset contains some obvious outliers.



When the age value is between -1 and 0, the class is usually 0. However, as the age value increases, the presence of pink dots increases.



Class values are usually 0 when age is between -1 and 0 and plas is between -1 and 1.

- ## Explanation :-

The value of the nearest neighbor obviously influences the accuracy of the output prediction. The three selected characteristics clearly had a major influence on the output variable classes. Confusion matrices and classification issues are also inextricably linked and draw values from one another.

References Taken:-

1. https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761
2. https://www.geeksforgeeks.org/k-nearest-neighbours/
3. Introduction to Data mining by Pang-Ning Tan, Michael Steinbach and Vipin Kumar
4. Hands-on machine Learning using Sci-kit learn.