****

**Practical Machine Learning**

**Assignment 1**

**Student Id: R0018250*5***

**Name: D**heeraj **A**limchandani

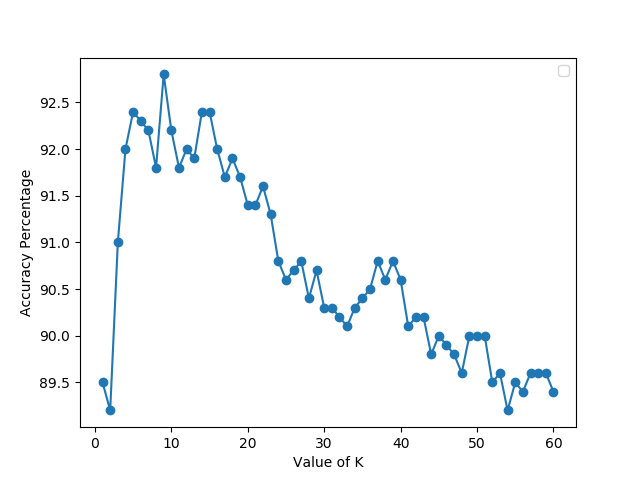
**Part 1: k-NN Algorithm**

A basic model of K-NN algorithm has been used in this part of the assignment. The problem set here is a multi-class classification problem. The model is trained on the provided classification training data and the accuracy of the model is checked against the provided test dataset.

The approach followed in order to predict the classes for the test data is as follows:

1. The test and the training dataset are read using NumPy
2. The test and training dataset are filtered based on features and the class
3. The training dataset with all features is used against every query instance present in the test dataset in order to calculate the Euclidean Distance
4. The Euclidean distances are sorted in the ascending order. The first K distances selected are shortest distances between the training instance and the query instance. The classes for these training instances are the set of predicted values for the query instance.
5. In case K = 1, the very first class is the predicted value that is to be considered.
6. In case K > 1, the mode of all the K predicted values is the predicted class of the query instance
7. The predicted class values for each query instance are compared with the actual test dataset class values and the accuracy of the model is computed.

The Basic KNN model was run against several K values and the results are as below:



**Conclusion:**

It can be seen from the above graph that the performance of the Basic KNN model in our case with the data provided degrades with increasing K value after a certain point.

The best accuracy is observed at **K = 9**, the accuracy achieved is **92.8%.**

It can be concluded that the basic KNN model is best accurate at K =9.

**Part 2: k-NN variants and hyper-parameters**

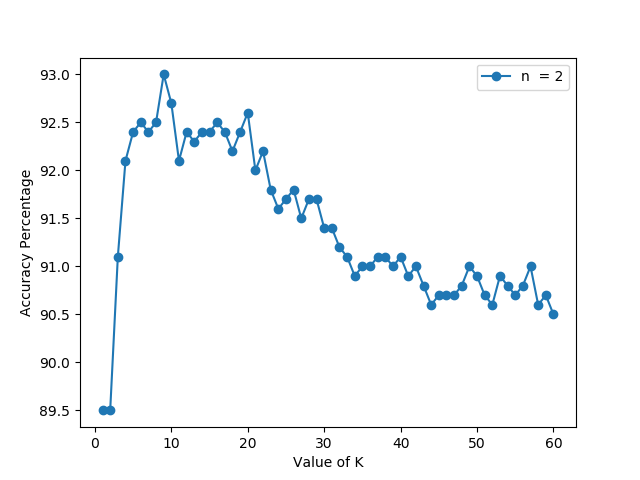
**a) Distance Weighted k-NN Algorithm for Classification**

A distance-weighted model of K-NN algorithm has been used in this part of the assignment. The problem set here is a multi-class classification problem. The model is trained on the provided classification training data and the accuracy of the model is checked against the provided test dataset.

The approach followed in order to predict the classes for the test data is as follows:

1. The test and the training dataset are read using NumPy
2. The test and training dataset are filtered based on features and the class
3. The training dataset with all features is used against every query instance present in the test dataset in order to calculate the Euclidean Distance
4. The Euclidean distances are sorted in the ascending order. The first K distances selected are shortest distances between the training instance and the query instance. The classes for these training instances are the set of predicted values for the query instances.
5. In case K = 1, the very first class is the predicted value that is to be considered.
6. In case K > 1,
   1. For specific class, identify each instance amongst the K nearest instances that belong to that class
   2. Calculate and add the inverse distance for each of the identified instances.
7. The largest value amongst all the calculated values is the predicted value.
8. The predicted class values for each query instance are compared with the actual test dataset class values and the accuracy of the model is computed.

The Distance-Weighted KNN model was run against several K values and the results are as below:



**Conclusion:**

The above graph represents the predictions for the distance weighted KNN model where the inverse distance squared is used(n=2).

For **K=10**, the accuracy of the model is **92.7%**

It can be seen from the above graph that the performance of the distance weighted KNN model in our case with the data provided degrades with increasing K value after a certain point.

The best accuracy is observed at **K = 9**, the accuracy achieved is **93.0%.**

It can be concluded that the distance weighted KNN model is best accurate at K =9.

**b) Investigating k-NN variants and hyper-parameters**

A variety of techniques are available in order to investigate the performance of the k-NN model. For the purpose of this assignment the techniques considered are as below:

1. Selecting the value of K nearest neighbours
2. Using different distance metrics
3. Performing the scaling of the dataset
4. **Selecting the value of K nearest neighbours:**

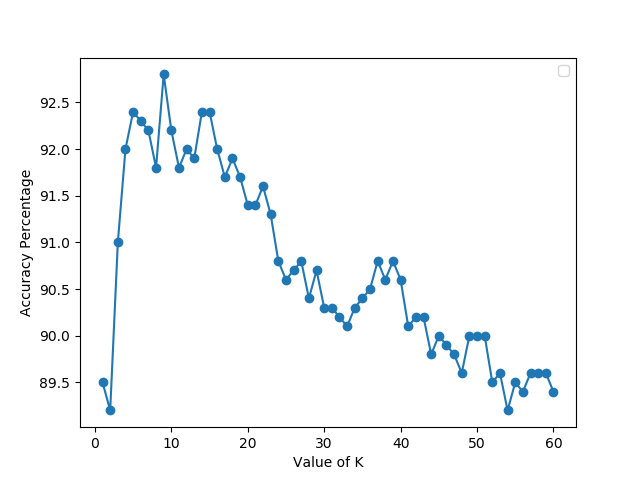
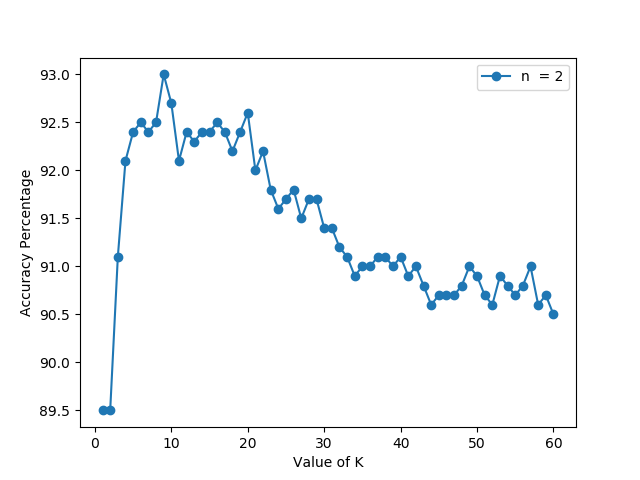
The process of selecting the appropriate value of k at which the model performs the best is not defined or has no definite process. Also, it should be noted that the selection of appropriate value of k is import in the success of the performance of the model.

Selection of a too small value of k can make the model more inclined towards noise and may lead to overfitting and it generates a very sensitive model, whereas selection of a too large value of k can reduce the impact of noise and on the same time can make the boundaries between the classes less distinct. A large value of k also come with a larger computational expense.

Hence, it is advisable to choose the value of k keeping the above-mentioned points. There are variety of techniques for selecting the k vale:

* Square root of the number of classified instances [k = sqrt(n)]
* Selecting a range of k values and assessing the performance against each selected value to measure the accuracy.

In order to inspect the most appropriate value of k at which the model performs best, a distance weighted k-NN and Basic k-NN model was run with k ranging from 1 to 60 and the results were noted on the line graph.



Distance weighted k-NN Basic k-NN

**Observations:**

It can be observed from the above graph that the performance of the distance weighted k-NN model with distance squared measure is worst for too small values of k and even degrades for larger values of k.

The most optimal k value here is **9**, on which the model performs the best which is showing the accuracy of **93.0%.**

On further increasing the k value the accuracy is dropped in an orderly fashion.

The similar behaviour can be observed for the basic k-NN model as well, the best performance is recorded at **K=10**, when accuracy of the model is **92.7%.** The accuracy drops further with increasing k value.

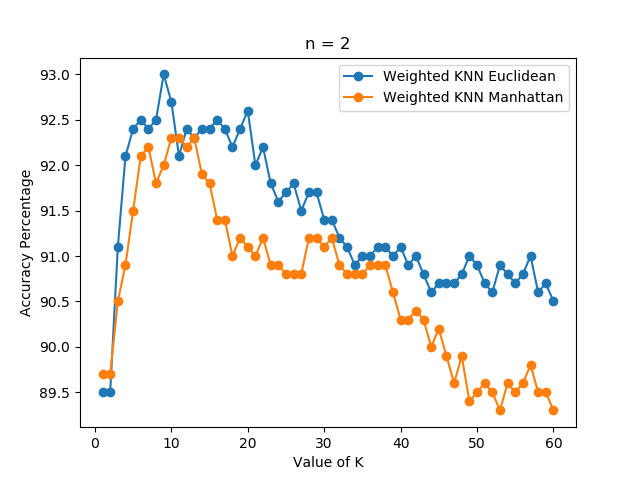
1. **Using different distance metrics**

The basic idea of the k-NN model depends on measuring the distance between the query instance and the training instance. The closer the training instance is to the query instance the probability of selecting that instance increases, hence distance measurement is the important metric in understanding the performance of k-NN model.

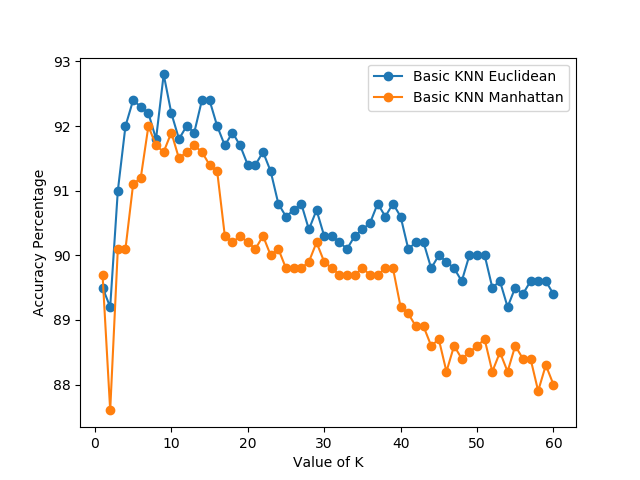
There are number of ways that can be used in order to calculate the distance. For the purpose of this assignment the below distance metrics have been used to understand the selection of distance calculation on the k-NN model

* Euclidean Distance
* Manhattan Distance

In order to inspect the impact of distance selection metric, a distance weighted k-NN and Basic k-NN model was run with k ranging from 1 to 60 for Euclidean and Manhattan distance metrics and the results were noted on the line graph.



Distance weighted k-NN



Basic k-NN

**Observations:**

It can be seen from the above graphs that the k-NN model performs well with Euclidean distance compared to Manhattan distance for a wide range of K values. Therefore, it can be said that selection of Euclidean distance as a distance metric for the k-NN model in our case is a better choice over Manhattan distance calculation.

The Distance weighted k-NN with Euclidean Distance provides a maximum accuracy of 93% for K = 9 whereas the maximum accuracy with Manhattan is 92.3% for K=10.

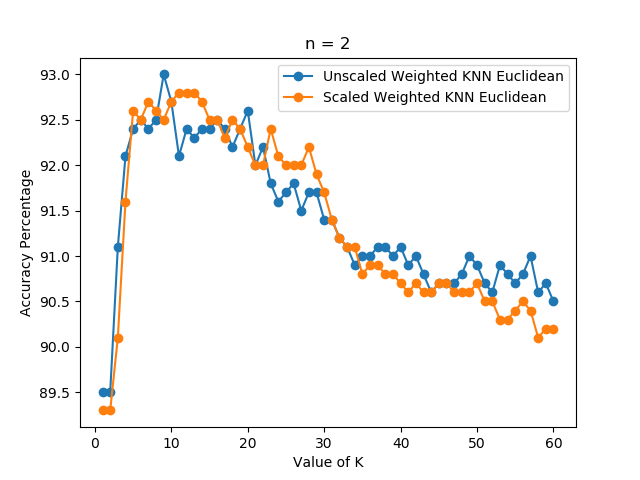
The Basic k-NN with Euclidean Distance provides a maximum accuracy of 92.8% for K=9 whereas the maximum accuracy with Manhattan is 92% for K = 7.

1. **Scaling the dataset**

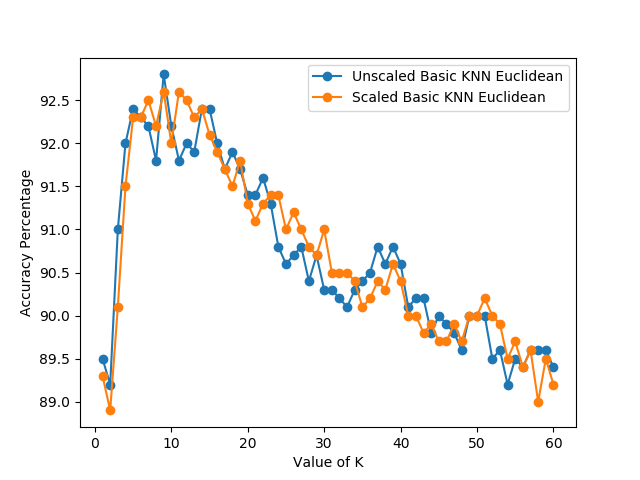
Scaling is the process of normalizing the dataset that helps in making the distance matric more meaningful. Scaling comes in handy when there is a large difference in the ranges of various features. For example, the range for the Feature 1 is from 0-100 and for Feature 2 is from1 to 10,000. This will clearly favour the distance metric in terms of Feature 2, to overcome this problem scaling is performed on the dataset to normalize the data along all the features.

For the purpose of this assignment, the performance comparison has been made between the scaled and unscaled dataset for the weighted and basic k-NN model.

A distance weighted k-NN and Basic k-NN model was run with k ranging from 1 to 60 for Euclidean distance metrics over scaled and unscaled dataset and the results were noted on the line graph.



Distance weighted k-NN



Basic k-NN

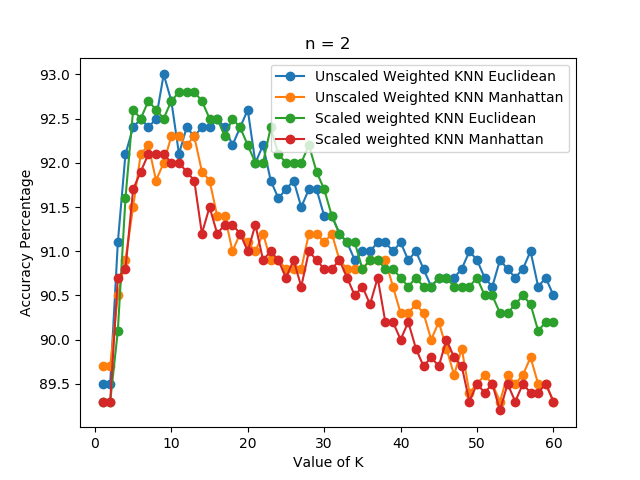
**Observations:**

From the above graphs the k-NN model when applied over a scaled dataset tends to provide better accuracy over specific k values. A general trend that can be seen here is that for certain values of k the unscaled version outperforms the scaled model but while measuring the overall performance over wide range of k values, the scaled k-NN for both the Basic and Weighted model performs better.

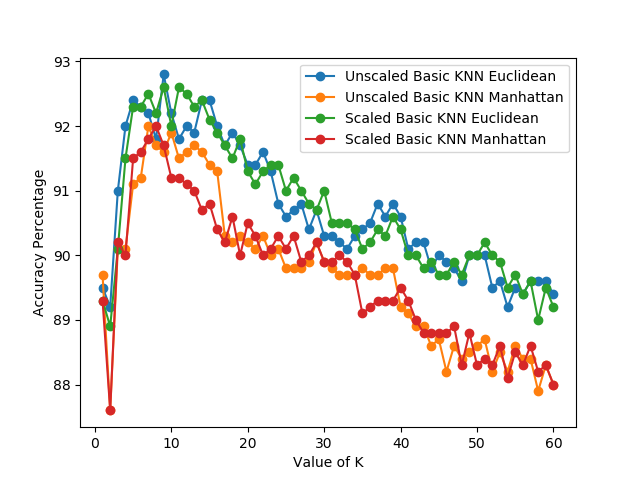
**\*\*Intentionally left blank\*\***

**3.1 Overall Performance Measures**

The overall performance of the k-NN model with various techniques is incorporated below:



Distance weighted k-NN



Basic k-NN

**Part 3: Implementation of k-NN for Regression**

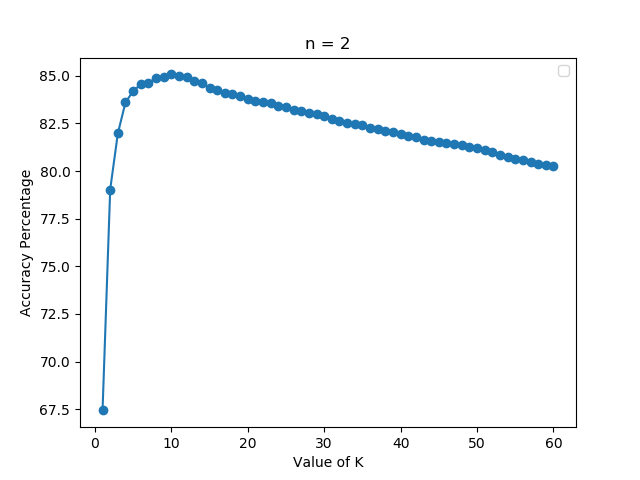
**a) Distance Weighted k-NN Algorithm for Regression**

A distance-weighted model of K-NN algorithm has been used in this part of the assignment. The problem set here is regression problem. The model is trained on the provided regression training data and the accuracy of the model is checked against the provided test dataset using the R2 coefficient metric.

The approach followed in order to predict the values for the test data is as follows:

1. The test and the training dataset are read using NumPy
2. The test and training dataset are filtered based on features and the target regression value.
3. The training dataset with all features is used against every query instance present in the test dataset in order to calculate the Euclidean Distance
4. The Euclidean distances are sorted in the ascending order. The first K distances selected are shortest distances between the training instance and the query instance. The regression values for these training instances are the set of predicted values for the query instances.
5. The selected k values are used to calculate distance weighted regression value which are the predicted values.
6. The accuracy of the model is then calculated using R2 coefficient metric.

The Distance-Weighted KNN model for regression was run against several K values and the results are as below:

****

Distance weighted k-NN

**Conclusion:**

It can be seen from the above graph that the performance of the Distance weighted KNN model in our case with the regression data provided degrades with increasing K value after a certain point.

The best accuracy is observed at **K = 10**, the accuracy achieved is **85.07%.**

It can be concluded that the basic KNN model is best accurate at K =10.

**b) Feature selection techniques**

Feature selection plays a vital role in deciding the performance of the model. The features that are selected in order to train the model have a huge impact on how the model performs.

Considering each feature is good but sometimes certain features that are not relevant or are not important if taken in consideration can impact the prediction of the model. Selection of such features can negatively impact the model. The features which hardly help in improving the model also brings with them increased computational time as the model take more time to get trained. Also, certain features can act as noise which can hamper the accuracy of the model, therefore it is very important to perform feature selection on the dataset. The idea behind feature selection is to select the most relevant and important features that can lead to a better performing model.

A basic k-NN model that weights the contribution of each feature equally may contain features that contribute very little or nothing to the performance of the model, also they may add nose to the model and overfit the model resulting in producing less accurate results. Therefore, feature selection must be the very first step in building the model.

**Feature Selection Methods:**

1. Univariate Selection
2. Random Forest
3. Correlation Matrix

**1. Univariate Selection**

The univariate selection process for selecting the feature uses statistical tests to select those features that have a strong relationship with the output variable.

The process of univariate feature selection is to look at each feature and decide if it is relevant or not. It can be described as examining the importance of individual feature in making a prediction and examining the relationship of the feature with the response variable.

There are multiple methods that come under univariate feature selection

* **Removing features with low variance**

Low variance implies that the feature is not varying much in the entire dataset as compared to other features where the variance can be very large. The problem here is that it means the data is almost same in all the instances. This means that these features are not benefitting the prediction. Removing such features can be a good choice.

* **Pearson Correlation**

In this case the aim is to find the correlation between the target variables and the selected feature and to see if there is any relationship or not, if there is no relationship drop that feature.

**2. Random Forest**

Random forest feature selection is one of the popular techniques for selecting the relevant features. The method creates hundreds of decision trees each of them is built using the features of the dataset. The selection of the feature depends on by how much that feature decreases the impurity of the tree. The reduction of impurity in decision tree by the feature is directly proportional to the selection of that feature. The impurity decrease by each feature across all the decision tree is then averaged in order to determine the selected features.

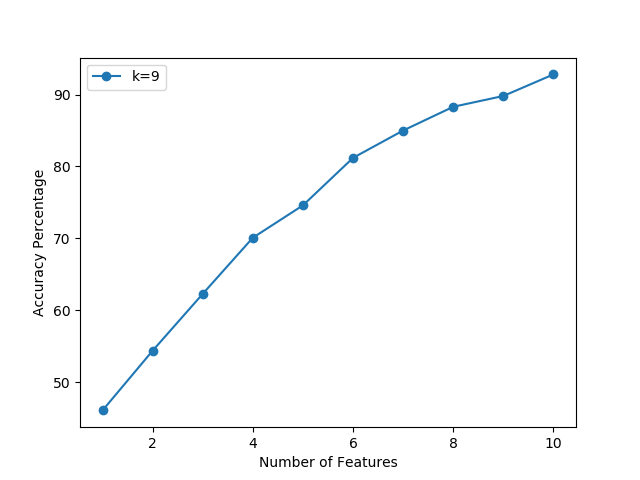
**3. Correlation Matrix**

Correlation is defined as the measure of linear relationship between two variables. In case of feature selection correlation comes in handy. Features with high correlation value are more linearly dependent and therefore they should have a similar effect on the target variable. Hence dropping one of the two features is a good idea.

**Implementation**

For the purpose of this assignment one of the above-mentioned feature selection method, Univariate Method is implemented in python using scikit learn. The feature selection is implemented using ‘SelectKBest’ class which selects features according to k highest scores and the chi2 method which computes chi-squared stats between each non-negative feature and class.

The accuracy of the basic k-NN model has been tested against f features for the classification dataset. The distances are calculated using Euclidean method and the value of k is 9 in this case.



**Conclusion:**

The graph depicts that for the classification method the model performs best for all the 10 available features with an accuracy of 92.8%. It implies that all the features in this case are equally important in predicting the class.