# KNN Classifier

**Objective:** To implement KNN classification algorithm on iris dataset as part of Lab Migration Project.

**Methods:**

(i) Load the iris dataset and view information about it.
(ii) Split dataset to training and testing data.
(iii) Perform feature scaling of training and testing data.
(iv) Fit KNN model to training data and plot it.
(v) Compute accuracy of model and perform model evaluation using different values of K (3, 5, 7, 15).
(vi) Conclusion

```
#To clear the environment
rm(list=ls())

#Importing the libraries
library(class)

#Load and view information about the dataset
mydata=iris
head(mydata)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```
str(mydata)
```

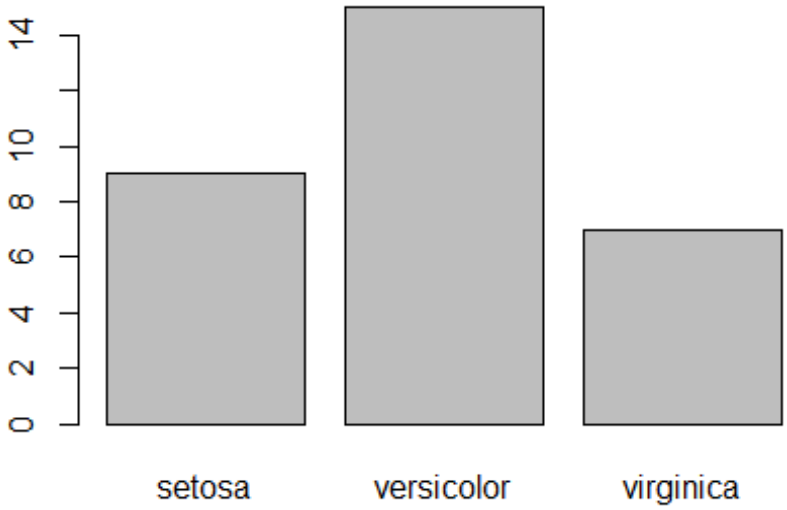
```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1
1 1 1 1 ...
```

```
#Split into training and testing data
index=sample(2,nrow(mydata),replace=TRUE,prob=c(0.7,0.3))
training=mydata[index==1,]
testing=mydata[index==2,]
```

Inference: To split iris data to training data and testing data. Among the 150 observations available, approximately 70% of the dataset is used for training and the remaining for testing.

```
#Feature scaling
training_scale <- scale(training[,1:4])
testing_scale <- scale(testing[,1:4])

#Fitting KNN Model to training data
KNNM <- knn(train=training_scale,test=testing_scale,cl=training$Species,k=1)
plot(KNNM)
```




```
#Confusion matrix
CFM <- table(testing$Species,KNNM)
CFM

##             KNNM
##              setosa versicolor virginica
##   setosa          9          0         0
##   versicolor      0         13         1
##   virginica       0          2         6

#Accuracy of model
misClassError <- mean(KNNM!=testing$Species)
print(paste('Accuracy =',1-misClassError))

## [1] "Accuracy = 0.903225806451613"

#When k=3
KNNM <- knn(train=training_scale,test=testing_scale,cl=training$Species,k=3)
```

```
misClassError <- mean(KNNM!=testing$Species)
print(paste('Accuracy =',1-misClassError))

## [1] "Accuracy = 0.967741935483871"

#When k=5
KNNM <- knn(train=training_scale,test=testing_scale,cl=training$Species,k=5)
misClassError <- mean(KNNM!=testing$Species)
print(paste('Accuracy =',1-misClassError))

## [1] "Accuracy = 0.935483870967742"

#When k=7
KNNM <- knn(train=training_scale,test=testing_scale,cl=training$Species,k=7)
misClassError <- mean(KNNM!=testing$Species)
print(paste('Accuracy =',1-misClassError))

## [1] "Accuracy = 0.935483870967742"

#When k=15
KNNM <- knn(train=training_scale,test=testing_scale,cl=training$Species,k=15)
misClassError <- mean(KNNM!=testing$Species)
print(paste('Accuracy =',1-misClassError))

## [1] "Accuracy = 0.935483870967742"
```

**Conclusion:**

The K-Nearest Neighbor is a supervised non-linear classification algorithm. It is dependent on its k value (neighbors) and finds use in a variety of industries, including banking and healthcare.

On applying the algorithm in the iris dataset, we can see that the model shows high accuracy of 96.77% when K=3.