R program :-

x<-read.csv("AEB.csv",header=TRUE) // reading the csv data

head(x) //printing the first six rows

x<-x[,c(-1)] // ignoring the first column as it contains the row numbers

head(x)

library(e1071) //including the libraries

library(dplyr)

train\_index <- sample(1:nrow(x), 0.6 \* nrow(x)) // extracting the first 60% row numbers as training

test\_index <- setdiff(1:nrow(x), train\_index) // and the rest as testing data

train<-x[train\_index,]

test<-x[test\_index,]

NB=naiveBayes(class ~., data=train) // training the naïve bayes model

NB

NBpredict=predict(NB,test) //predicting the class for test data

z<-table(NBpredict,test$class) //generating confusion matrix

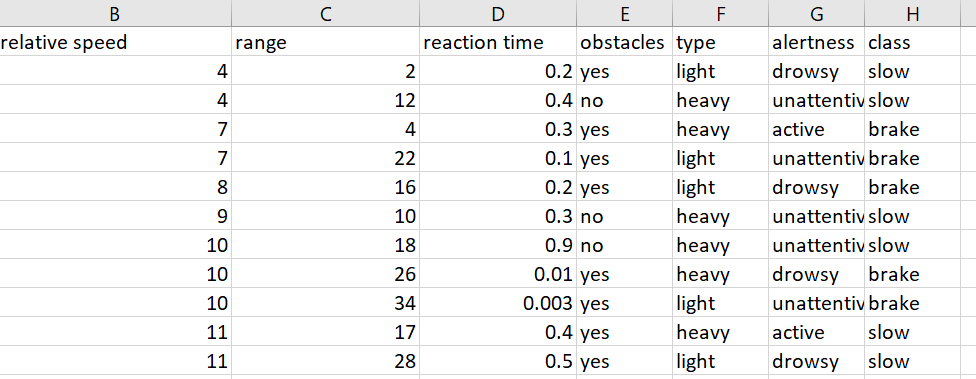
z

library(caret)

a<-confusionMatrix(z) //analyzing with confusion matrix with performance metrics

a

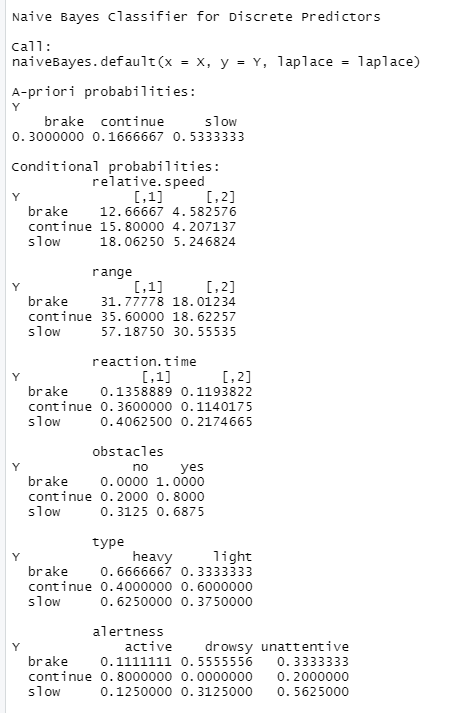
AEB.csv



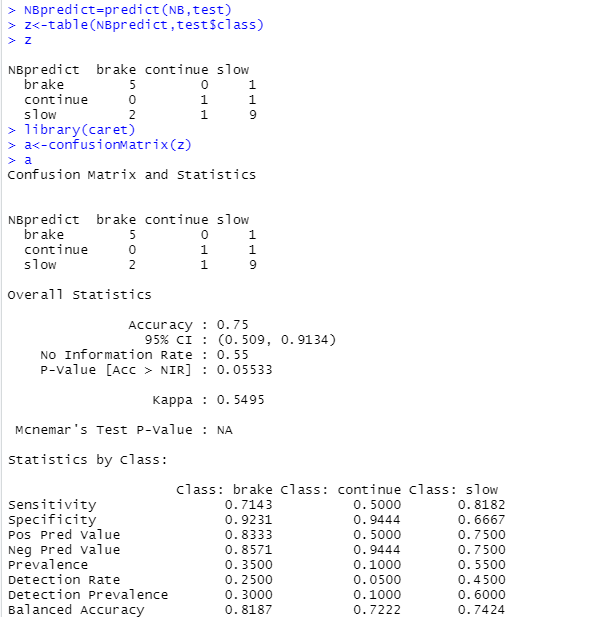
Process

* First we read the data the into the data frame .
* and then preprocessed the data by following steps
  + by removing the unnecessary attributes.
  + by replacing missing values with mean.
* Then we have split the data into train and test (60:40).
* After splitting we built the naivebayes model using the train data.
* Then from the built model we have predicted the test data.
* After that we have compared the results with the actual results and have calculated various performance metrics by making confusion matrix.

Outputs:-



The above is the output of the built naïve bayes model using train data which shows the A-priori probabilities of each class. It also depicts the conditional probabilities of each class label given the attributes. For continuous attributes it gives the mean and standard deviation for each class label.



The above output gives us the results after predicting the test data output class using the built model

It also gives the confusion matrix of the prediction results from which we got various performance metrics of the model like accuracy along with its confidence interval, sensitivity, specificity etc..

Given a situation our model is 75% percent accurate with a 95 CI value of (0.509,0.9134) that it takes correct decisions.