**Practical Implementation of Naive Bayes In R**

**Problem Statement:** To study a Diabetes data set and build a Machine Learning model that predicts whether or not a person has Diabetes.

**Data Set Description:** The given data set contains 768 observations of patients along with their health details. Here’s a list of the predictor variables that will help us classify a patient as either Diabetic or Normal:

* Pregnancies: Number of pregnancies so far
* Glucose: Plasma glucose concentration
* Blood Pressure: Diastolic blood pressure (mm Hg)
* Skin Thickness: Triceps skin fold thickness (mm)
* Insulin: 2-Hour serum insulin (mu U/ml)
* BMI: Body mass index (weight in kg/(height in m)^2)
* Diabetes Pedigree Function: Diabetes pedigree function
* Age: Age (years)

The response variable or the output variable is:

* Outcome: Class variable (0 or 1)

Logic: To build a Naive Bayes model in order to classify patients as either Diabetic or normal by studying their medical records such as Glucose level, age, BMI, etc.

Library() is used parameter as name of the library to attach.

#install "caTools" go to tools menu. Give name caTools and click intall this is required for the splitting of data

Contains several basic utility functions including: moving (rolling, running) window statistic functions, read/write for GIF and ENVI binary files, fast calculation of AUC, LogitBoost classifier, base64 encoder/decoder, round-off-error-free sum and cumsum, etc.

library(caTools)

#for naive bayes install "e1071" package in the same way as caTools

library(e1071)

#open a dataset in the format of csv file

#this is not built in dataset of r

#Header= true means then the first row will be treated as the row names.

# CSV file can be comma delimited or tab or any other delimiter specified by parameter "sep=".

mydata <-read.csv(file = "/Users/Desktop/diabetes.csv", header=TRUE, sep=",")

View(mydata)

dim(mydata)

[1] 768 9

range(mydata$Age)

[1] 21 81

range(mydata$BloodPressure)

[1] 0 122

mean(mydata$Age)

[1] 33.24089

#split the data : First parameter is dataset on which split is to be done, second parameter is splitratio which is used to split the data used during classification into train and test subsets. By using the sample.split() we are actually creating a vector with two values TRUE and FALSE. By setting the SplitRatio to 0.7, you are splitting the original dataset to 70% training and 30% testing data.

temp\_field<-sample.split(mydata, SplitRatio = 0.7)

#keep 70% for training

train<-subset(mydata,temp\_field==TRUE)

View(train)

#summarizing the property of training data set #calculating the dimension of training

dim(train)

[1] 513 9

range(train$Pregnancies)

[1] 0 15

range(train$Glucose)

[1] 0 197

mean(train$Insulin)

[1] 77.73294

#keep 30% for testing

test<-subset(mydata,temp\_field==FALSE)

View(test)

#display few samples

head(mydata)

head(train)

head(test)

tail(train)

tail(test)

# use of naive bayes algorithm #create a learning model

# in train data outcome field indicates class value (0/1)

# naive bayes wants the output not in numeric form but in the category form so use "as.factor"

# use of "~." indicates the outcome to be learned against all other fields of dataset

my\_model <-naiveBayes(as.factor(train$Outcome)~.,train)

my\_model

View(my\_model)

#prediction class

#from test data exclude 9th column which is outcome column

pred1<-predict(my\_model,test[,-9])

pred1

# to display the probability of outcome use type="raw" If "raw", the conditional a-posterior probabilities for each class are returned and the class with maximal probability else.

pred1<-predict(my\_model,test[,-9],type="raw")

pred1

# to display the prediction class of outcome use type="class"

pred1<-predict(my\_model,test[,-9],type="class")

pred1

#generate the confusion matrix

dim(test)

dim(train)

table(pred1,test$Outcome,dnn=c("actual","predicted"))

#save the prediction

output<-cbind(test,pred1)

View(output)

dim(output)