**Laboratory Practice I**

**Data Analytics**

***Shiva Saran***

***BE-B 20***

***2020-2021***

**Practical 2**

*Download Pima Indians Diabetes dataset. Use Naive Bayes‟ Algorithm for classification*

* Load the data from CSV file and split it into training and test datasets.
* Summarize the properties in the training dataset so that we can calculate probabilities and make predictions.
* *Classify samples from a test dataset and a summarized training dataset.*

Dataset Link - https://www.kaggle.com/uciml/pima-indians-diabetes-database

* ***Installing Libraries and Importing Data Set***

#Installing necessary Libraries  
> install.packages('e1071')

> install.packages('caTools')

#Checking that the libraries are successfully installed

> library(caTools)

> library(e1071)

#Importing The Dataset

> mydata <- read.csv("~/Documents/BE/LP1/diabetes.csv")

> View(mydata)

* ***Spiting the Dataset into training and testing Data***

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

> #70% will b in training

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

> #30% will be in testing

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

#Checking the

> head(train)

Pregnancies Glucose BloodPressure SkinThickness Insulin BMI DiabetesPedigreeFunction Age Outcome

3 8 183 64 0 0 23.3 0.672 32 1

4 1 89 66 23 94 28.1 0.167 21 0

5 0 137 40 35 168 43.1 2.288 33 1

6 5 116 74 0 0 25.6 0.201 30 0

8 10 115 0 0 0 35.3 0.134 29 0

9 2 197 70 45 543 30.5 0.158 53 1

> head(test)

Pregnancies Glucose BloodPressure SkinThickness Insulin BMI DiabetesPedigreeFunction Age Outcome

1 6 148 72 35 0 33.6 0.627 50 1

2 1 85 66 29 0 26.6 0.351 31 0

7 3 78 50 32 88 31.0 0.248 26 1

10 8 125 96 0 0 0.0 0.232 54 1

11 4 110 92 0 0 37.6 0.191 30 0

16 7 100 0 0 0 30.0 0.484 32 1

* ***> #Using Naive Bayes Algorithm, training the train Data Set***

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

> my\_model

Naive Bayes Classifier for Discrete Predictors

Call:

naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:

Y

0 1

0.6497065 0.3502935

Conditional probabilities:

Pregnancies

Y [,1] [,2]

0 3.253012 2.975604

1 4.636872 3.662286

Glucose

Y [,1] [,2]

0 109.1928 26.20657

1 142.4916 33.87259

BloodPressure

Y [,1] [,2]

0 67.91265 18.21095

1 70.83799 21.18993

SkinThickness

Y [,1] [,2]

0 19.29819 15.03807

1 22.16201 18.07387

Insulin

Y [,1] [,2]

0 65.10542 98.29565

1 100.30168 142.80693

BMI

Y [,1] [,2]

0 30.44277 7.229345

1 34.88994 6.879959

DiabetesPedigreeFunction

Y [,1] [,2]

0 0.4342139 0.3019496

1 0.5815140 0.3794261

Age

Y [,1] [,2]

0 30.96687 11.35298

1 36.95531 11.01981

* **#Now predicting the data remaining Split data using Trained dataset**

> #predicting, try putting type="class" or type="raw" after the test data

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

> pred1

[1] 1 0 0 0 0 0 0 0 1 0 1 0 1 0 0 1 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 1 1 1 0 0 0 0 0 1 0 0 1 0 0 1 0 1 0 1

[71] 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 1 1 1 0 0 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 1

[141] 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 1 0 1 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 1 1 0 0 1 0 0 0 0 0 0 0 1 0 1 0 1 1 0 1 1 0 1 0 1 1 1 0

[211] 0 0 0 0 0 1 1 0 0 1 1 1 1 0 0 1 1 1 0 0 0 1 0 0 1 1 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 1 1 0 0 0 0

Levels: 0 1

* **#Creating Confusion Matrix.**

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

Actual

predicted 0 1

0 140 37

1 28 52

> #To save the prediction

> output<-cbind(test, pred1)

> View(output)