```
>
> #https://www.edureka.co/blog/naive-bayes-in-r/
> #Zulaikha Lateef
> #Zulaikha is a tech enthusiast working as a Research Analyst at Edureka.
>
> if (FALSE)
+ {"
+ 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
+ BloodPressure: Diastolic blood pressure (mm Hg)
+ SkinThickness: Triceps skin fold thickness (mm)
+ Insulin: 2-Hour serum insulin (mu U/ml)
+ BMI: Body mass index (weight in kg/(height in m)^2)
+ DiabetesPedigreeFunction: 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.
+ "}
>
> #Loading required packages
> #install.packages('tidyverse')
> library(tidyverse)
> #install.packages('ggplot2')
> library(ggplot2)
> #install.packages('caret')
> library(caret)
> #install.packages('caretEnsemble')
> library(caretEnsemble)
> #install.packages('psych')
> library(psych)
> #install.packages('Amelia')
> library(Amelia)
> #install.packages('mice')
> library(mice)
> #install.packages('GGally')
> library(GGally)
> #install.packages('rpart')
> library(rpart)
> #install.packages('randomForest')
> library(randomForest)
```

```
> #install.packages('klaR')
> library(klaR)
>
> #Reading data into R
> data<- read.csv("C:/Users/jmard/OneDrive/Desktop/Computing and Graphics in Applied
Statistics2020/Bayes_Material/diabetes.csv")
> table(data$Outcome)

  0   1
500 268
>
> #Setting outcome variables as categorical
> data$Outcome <- factor(data$Outcome, levels = c(0,1), labels = c("False", "True"))
> table(data$Outcome)

False  True
  500   268
>
> #Studying the structure of the data
> str(data)
'data.frame':   768 obs. of  9 variables:
 $ Pregnancies             : int  6 1 8 1 0 5 3 10 2 8 ...
 $ Glucose                 : int  148 85 183 89 137 116 78 115 197 125 ...
 $ BloodPressure           : int  72 66 64 66 40 74 50 0 70 96 ...
 $ SkinThickness           : int  35 29 0 23 35 0 32 0 45 0 ...
 $ Insulin                 : int  0 0 0 94 168 0 88 0 543 0 ...
 $ BMI                     : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
 $ DiabetesPedigreeFunction: num  0.627 0.351 0.672 0.167 2.288 ...
 $ Age                     : int  50 31 32 21 33 30 26 29 53 54 ...
 $ Outcome                 : Factor w/ 2 levels "False","True": 2 1 2 1 2 1 2 1 2 2 ...
> describe(data)
                        vars   n   mean     sd median trimmed   mad   min    max  range  skew kurtosis   se
Pregnancies                1 768   3.85   3.37   3.00    3.46  2.97  0.00  17.00  17.00  0.90     0.14 0.12
Glucose                    2 768 120.89  31.97 117.00  119.38 29.65  0.00 199.00 199.00  0.17     0.62 1.15
BloodPressure              3 768  69.11  19.36  72.00   71.36 11.86  0.00 122.00 122.00 -1.84     5.12 0.70
SkinThickness              4 768  20.54  15.95  23.00   19.94 17.79  0.00  99.00  99.00  0.11    -0.53 0.58
Insulin                    5 768  79.80 115.24  30.50   56.75 45.22  0.00 846.00 846.00  2.26     7.13 4.16
BMI                        6 768  31.99   7.88  32.00   31.96  6.82  0.00  67.10  67.10 -0.43     3.24 0.28
DiabetesPedigreeFunction   7 768   0.47   0.33   0.37    0.42  0.25  0.08   2.42   2.34  1.91     5.53 0.01
Age                        8 768  33.24  11.76  29.00   31.54 10.38 21.00  81.00  60.00  1.13     0.62 0.42
Outcome*                   9 768   1.35   0.48   1.00    1.31  0.00  1.00   2.00   1.00  0.63    -1.60 0.02
>
> #Convert '0' values into NA
> data[, 2:7][data[, 2:7] == 0] <- NA
>
> #save graph(s) in pdf
> windows(7,7)
>  pdf(file="C:/Users/jmard/OneDrive/Desktop/Computing and Graphics in Applied
```

```
Statistics2020/Output/BayesClassifierExample2R_Figure.pdf")
>
> #visualize the missing data
> missmap(data)
>
> #Use mice package to predict missing values  Generates Multivariate Imputations by Chained Equations (MICE)
> mice_mod <- mice(data[, c("Glucose","BloodPressure","SkinThickness","Insulin","BMI")], method='rf')  #method is rf=Random
Forest

 iter imp variable
  1   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  1   2  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  1   3  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  1   4  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  1   5  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  2   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  2   2  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  2   3  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  2   4  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  2   5  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  3   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  3   2  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  3   3  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  3   4  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  3   5  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  4   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  4   2  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  4   3  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  4   4  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  4   5  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  5   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  5   2  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  5   3  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  5   4  Glucose  BloodPressure  SkinThickness  Insulin  BMI
  5   5  Glucose  BloodPressure  SkinThickness  Insulin  BMI
> mice_complete <- complete(mice_mod)
>
> #Transfer the predicted missing values into the main data set
> data$Glucose <- mice_complete$Glucose
> data$BloodPressure <- mice_complete$BloodPressure
> data$SkinThickness <- mice_complete$SkinThickness
> data$Insulin<- mice_complete$Insulin
> data$BMI <- mice_complete$BMI
>
> missmap(data)
>
> #Data Visualization
> #Visual 1
```

```
> ggplot(data, aes(Age, colour = Outcome)) +
+ geom_freqpoly(binwidth = 1) + labs(title="Age Distribution by Outcome")
>
> #visual 2
> c <- ggplot(data, aes(x=Pregnancies, fill=Outcome, color=Outcome)) +
+ geom_histogram(binwidth = 1) + labs(title="Pregnancy Distribution by Outcome")
> c + theme_bw()
>
> #visual 3
> P <- ggplot(data, aes(x=BMI, fill=Outcome, color=Outcome)) +
+ geom_histogram(binwidth = 1) + labs(title="BMI Distribution by Outcome")
> P + theme_bw()
>
> #visual 4
> ggplot(data, aes(Glucose, colour = Outcome)) +
+ geom_freqpoly(binwidth = 1) + labs(title="Glucose Distribution by Outcome")
>
> #visual 5
> ggpairs(data)

 plot: [1,1] [>-------------------------------------------]  1% est: 0s
 plot: [1,2] [>-------------------------------------------]  2% est: 2s
 plot: [1,3] [=>------------------------------------------]  4% est: 2s
 plot: [1,4] [=>------------------------------------------]  5% est: 2s
 plot: [1,5] [==>-----------------------------------------]  6% est: 2s
 plot: [1,6] [===>----------------------------------------]  7% est: 3s
 plot: [1,7] [===>----------------------------------------]  9% est: 3s
 plot: [1,8] [====>---------------------------------------] 10% est: 2s
 plot: [1,9] [=====>--------------------------------------] 11% est: 3s
 plot: [2,1] [=====>--------------------------------------] 12% est: 3s
 plot: [2,2] [======>-------------------------------------] 14% est: 3s
 plot: [2,3] [======>-------------------------------------] 15% est: 3s
 plot: [2,4] [=======>------------------------------------] 16% est: 3s
 plot: [2,5] [========>-----------------------------------] 17% est: 3s
 plot: [2,6] [========>-----------------------------------] 19% est: 3s
 plot: [2,7] [=========>----------------------------------] 20% est: 2s
 plot: [2,8] [=========>----------------------------------] 21% est: 2s
 plot: [2,9] [==========>---------------------------------] 22% est: 2s
 plot: [3,1] [==========>---------------------------------] 23% est: 2s
 plot: [3,2] [==========>---------------------------------] 25% est: 2s
 plot: [3,3] [===========>--------------------------------] 26% est: 2s
 plot: [3,4] [============>-------------------------------] 27% est: 2s
 plot: [3,5] [============>-------------------------------] 28% est: 2s
 plot: [3,6] [=============>------------------------------] 30% est: 2s
 plot: [3,7] [=============>------------------------------] 31% est: 2s
 plot: [3,8] [==============>-----------------------------] 32% est: 2s
 plot: [3,9] [===============>----------------------------] 33% est: 2s
 plot: [4,1] [================>---------------------------] 35% est: 2s
```

```
 plot: [4,2] [=================>------------------------------] 36% est: 2s
 plot: [4,3] [==================>-----------------------------] 37% est: 2s
 plot: [4,4] [==================>-----------------------------] 38% est: 2s
 plot: [4,5] [===================>----------------------------] 40% est: 2s
 plot: [4,6] [===================>----------------------------] 41% est: 2s
 plot: [4,7] [====================>---------------------------] 42% est: 2s
 plot: [4,8] [====================>---------------------------] 43% est: 2s
 plot: [4,9] [=====================>--------------------------] 44% est: 2s
 plot: [5,1] [=====================>--------------------------] 46% est: 2s
 plot: [5,2] [=====================>--------------------------] 47% est: 2s
 plot: [5,3] [======================>-------------------------] 48% est: 2s
 plot: [5,4] [======================>-------------------------] 49% est: 2s
 plot: [5,5] [======================>-------------------------] 51% est: 2s
 plot: [5,6] [=======================>------------------------] 52% est: 2s
 plot: [5,7] [=======================>------------------------] 53% est: 2s
 plot: [5,8] [=======================>------------------------] 54% est: 2s
 plot: [5,9] [========================>-----------------------] 56% est: 1s
 plot: [6,1] [========================>-----------------------] 57% est: 1s
 plot: [6,2] [========================>-----------------------] 58% est: 1s
 plot: [6,3] [=========================>----------------------] 59% est: 1s
 plot: [6,4] [=========================>----------------------] 60% est: 1s
 plot: [6,5] [=========================>----------------------] 62% est: 1s
 plot: [6,6] [==========================>---------------------] 63% est: 1s
 plot: [6,7] [==========================>---------------------] 64% est: 1s
 plot: [6,8] [==========================>---------------------] 65% est: 1s
 plot: [6,9] [===========================>--------------------] 67% est: 1s
 plot: [7,1] [===========================>--------------------] 68% est: 1s
 plot: [7,2] [===========================>--------------------] 69% est: 1s
 plot: [7,3] [============================>-------------------] 70% est: 1s
 plot: [7,4] [============================>-------------------] 72% est: 1s
 plot: [7,5] [============================>-------------------] 73% est: 1s
 plot: [7,6] [=============================>------------------] 74% est: 1s
 plot: [7,7] [=============================>------------------] 75% est: 1s
 plot: [7,8] [=============================>------------------] 77% est: 1s
 plot: [7,9] [==============================>-----------------] 78% est: 1s
 plot: [8,1] [==============================>-----------------] 79% est: 1s
 plot: [8,2] [==============================>-----------------] 80% est: 1s
 plot: [8,3] [===============================>----------------] 81% est: 1s
 plot: [8,4] [===============================>----------------] 83% est: 1s
 plot: [8,5] [================================>---------------] 84% est: 1s
 plot: [8,6] [================================>---------------] 85% est: 0s
 plot: [8,7] [================================>---------------] 86% est: 0s
 plot: [8,8] [=================================>--------------] 88% est: 0s
 plot: [8,9] [=================================>--------------] 89% est: 0s
 plot: [9,1] [==================================>-------------] 90% est: 0s `stat_bin()` using `bins = 30`. Pick better
value with `binwidth`.

 plot: [9,2] [==================================>-------------] 91% est: 0s `stat_bin()` using `bins = 30`. Pick better
```

```
value with `binwidth`.

 plot: [9,3] [=============================================>----] 93% est: 0s `stat_bin()` using `bins = 30`. Pick better
value with `binwidth`.

 plot: [9,4] [==============================================>---] 94% est: 0s `stat_bin()` using `bins = 30`. Pick better
value with `binwidth`.

 plot: [9,5] [==============================================>--] 95% est: 0s `stat_bin()` using `bins = 30`. Pick better
value with `binwidth`.

 plot: [9,6] [==============================================>--] 96% est: 0s `stat_bin()` using `bins = 30`. Pick better
value with `binwidth`.

 plot: [9,7] [==============================================>-] 98% est: 0s `stat_bin()` using `bins = 30`. Pick better
value with `binwidth`.

 plot: [9,8] [==============================================>-] 99% est: 0s `stat_bin()` using `bins = 30`. Pick better
value with `binwidth`.

 plot: [9,9] [==============================================]100% est: 0s

>
> #Building a model
> #split data into training and test data sets
> indxTrain <- createDataPartition(y = data$Outcome,p = 0.75,list = FALSE)
> training <- data[indxTrain,]
> testing <- data[-indxTrain,]
>
> #Check dimensions of the split
>
> prop.table(table(data$Outcome)) * 100

   False     True
65.10417 34.89583
>
> prop.table(table(training$Outcome)) * 100

   False     True
65.10417 34.89583
>
> prop.table(table(testing$Outcome)) * 100

   False     True
65.10417 34.89583
>
> #create objects x which holds the predictor variables and y which holds the response variables
> x = training[,-9]
```

```
> y = training$Outcome
>
> #create Naive Bayes model by using the training data set
> library(e1071)
> model = train(x,y,'nb',trControl=trainControl(method='cv',number=10))
There were 50 or more warnings (use warnings() to see the first 50)
>
> #Model Evaluation
> #Predict testing set
> Predict <- predict(model,newdata = testing )
There were 50 or more warnings (use warnings() to see the first 50)
> #Get the confusion matrix to see accuracy value and other parameter values
> confusionMatrix(Predict, testing$Outcome )
Confusion Matrix and Statistics

          Reference
Prediction False True
     False   105   16
     True     20   51

               Accuracy : 0.8125
                 95% CI : (0.75, 0.8651)
    No Information Rate : 0.651
    P-Value [Acc > NIR] : 6.369e-07

                  Kappa : 0.593

 Mcnemar's Test P-Value : 0.6171

            Sensitivity : 0.8400
            Specificity : 0.7612
         Pos Pred Value : 0.8678
         Neg Pred Value : 0.7183
             Prevalence : 0.6510
         Detection Rate : 0.5469
   Detection Prevalence : 0.6302
      Balanced Accuracy : 0.8006

       'Positive' Class : False

>
> #draw a plot that shows how each predictor variable is independently responsible for predicting the outcome
> #Plot Variable performance
> X <- varImp(model)
> plot(X)
>
> ##-----------------------------------------------------------##
> dev.off()
```

```
null device
          1
>
```