Introduction to Data Science

Finalterm Project

|  |  |  |
| --- | --- | --- |
| Name | ID | Section |
| Amit Podder | 20-42273-1 | C |
| Writhik Banik Barshan | 20-43124-1 | C |

**Topic:** Cervical Cancer Dataset

**Dataset Description:**

The dataset is about Cervical Cancer. When cancer starts in the cervix, it is called cervical cancer.

Cervical cancer is cancer that starts in the cells of the cervix. The cervix is the lower, narrow end of the uterus (womb). The cervix connects the uterus to the vagina (birth canal). Cervical cancer usually develops slowly over time. Before cancer appears in the cervix, the cells of the cervix go through changes known as dysplasia, in which abnormal cells begin to appear in the cervical tissue. Over time, if not destroyed or removed, the abnormal cells may become cancer cells and start to grow and spread more deeply into the cervix and to surrounding areas. Anyone with a cervix is at risk for cervical cancer. It occurs most often in people over age 30. Long-lasting infection with certain types of human papillomavirus (HPV) is the main cause of cervical cancer. HPV is a common virus that is passed from one person to another during sex. At least half of sexually active people will have HPV at some point in their lives, but few women will get cervical cancer.

Screening tests and the HPV vaccine can help prevent cervical cancer. When cervical cancer is found early, it is highly treatable and associated with long survival and good quality of life.

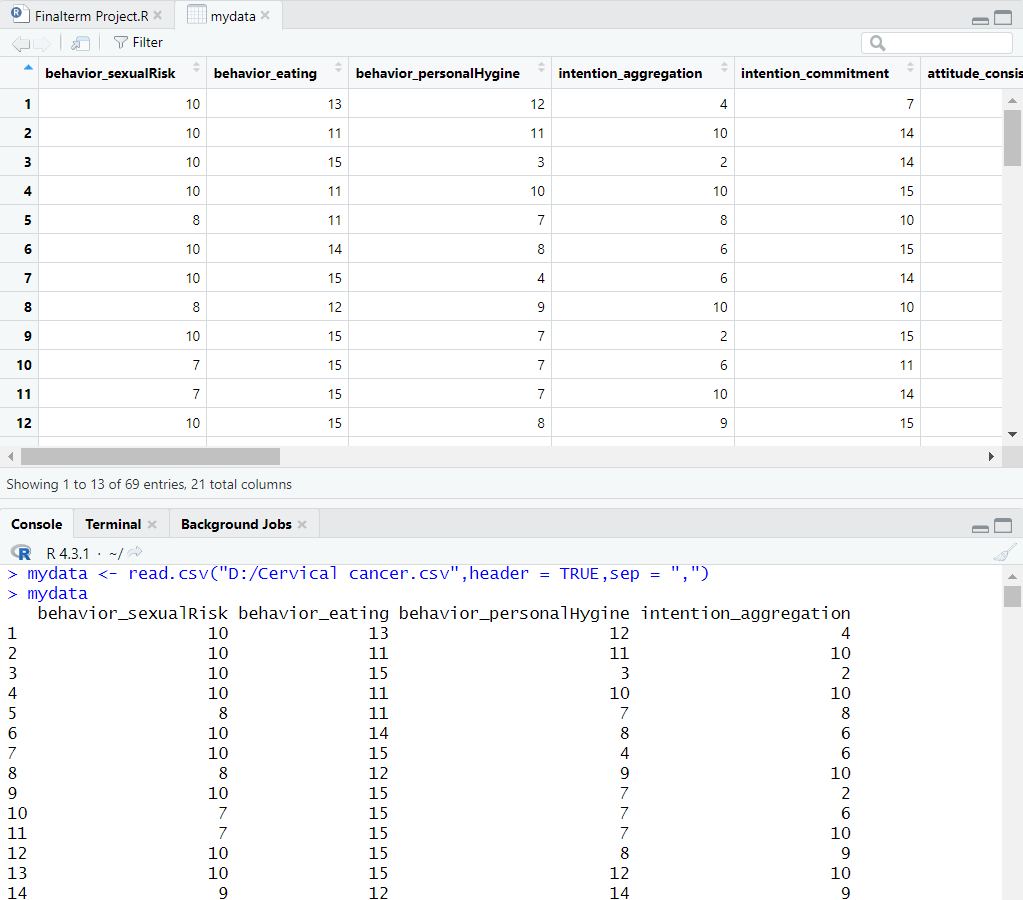
**Import the dataset as CSV and print the dataset:**

**Code:**

mydata <- read.csv("D:/Cervical cancer.csv",header = TRUE,sep = ",")

mydata

**Output:**



**Description:**

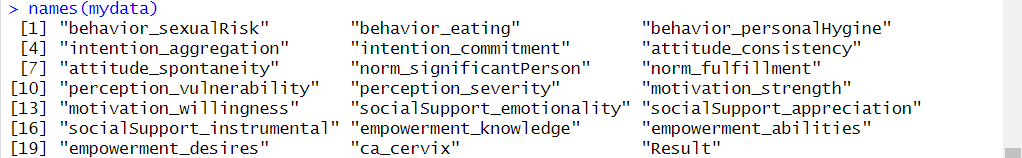
Here we have imported the code of the dataset as a csv file. We can also see the output of the dataset imported in RStudio.

**To see the column name of the dataset:**

**Code:**

names(mydata)

**Output:**



**Description:**

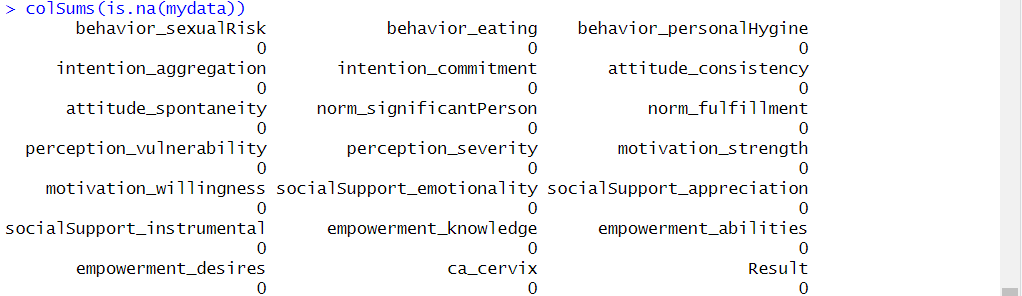
In this code, we can see the column name of the dataset. Here, with the help of the code names(), we can see all the attribute names present in the dataset.

**Finding the Missing(Null) values:**

**Code:**

colSums(is.na(mydata))

**Output:**



**Description:**

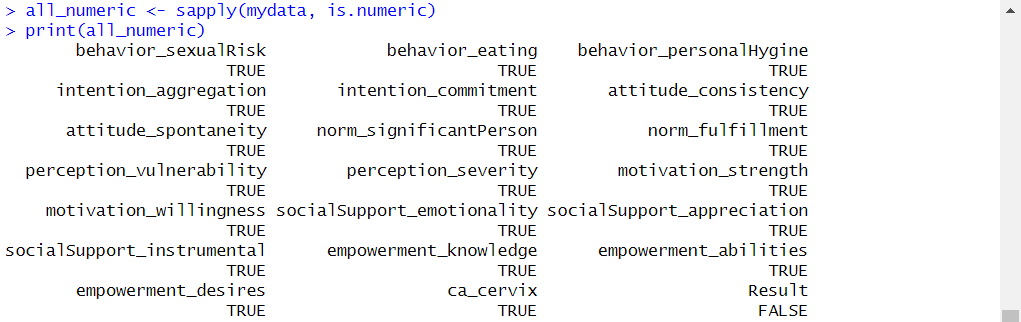
In this code, we can see all the null values of the dataset. Here, with the help of the code colSums(is.na()), we can check missing values in each column.

**Check whether all the data is numeric or not:**

all\_numeric <- sapply(mydata, is.numeric)

print(all\_numeric)

**Output:**



**Description:**

Here, we are using this code to check whether all the data is numeric or not. After implementing the code, we can see that all the data is showing true which means all the data is numeric.

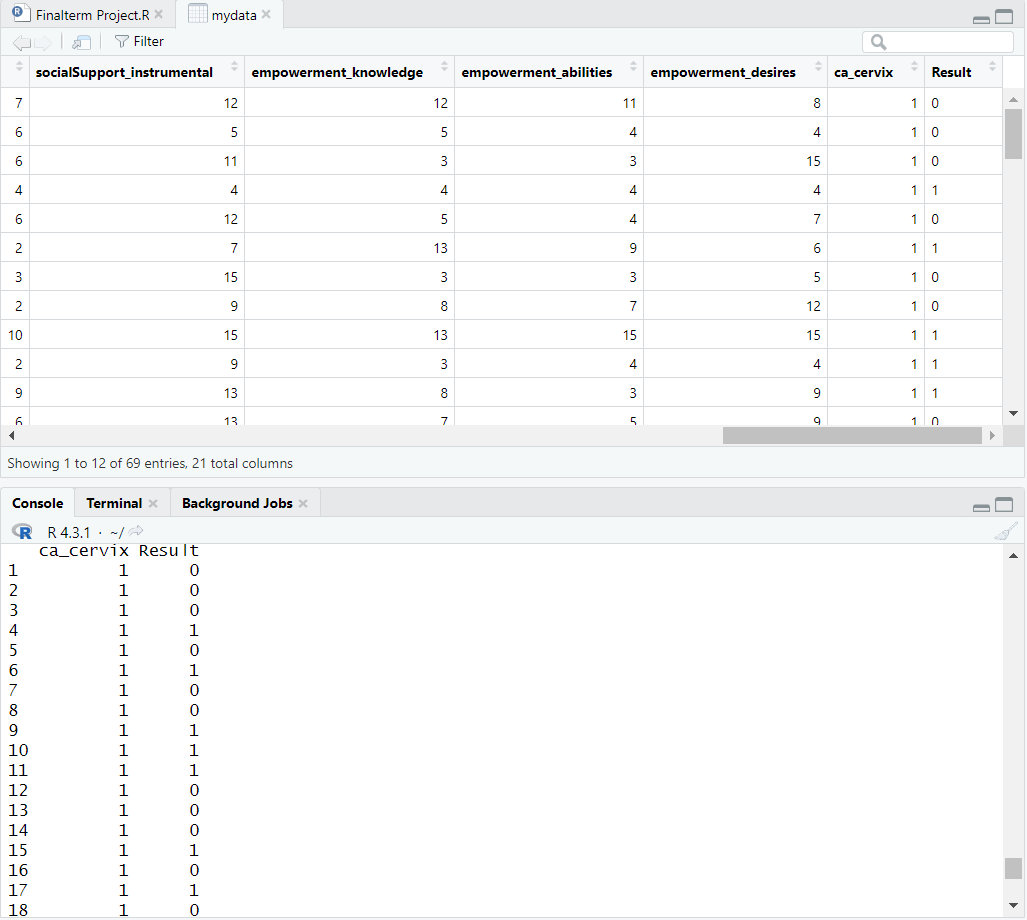
**Annotating Datasets:**

**Code:**

mydata$Result <- factor(mydata$Result, levels = c("Positive","Negetive"),labels = c("0","1"))

mydata

**Output:**



**Description:**

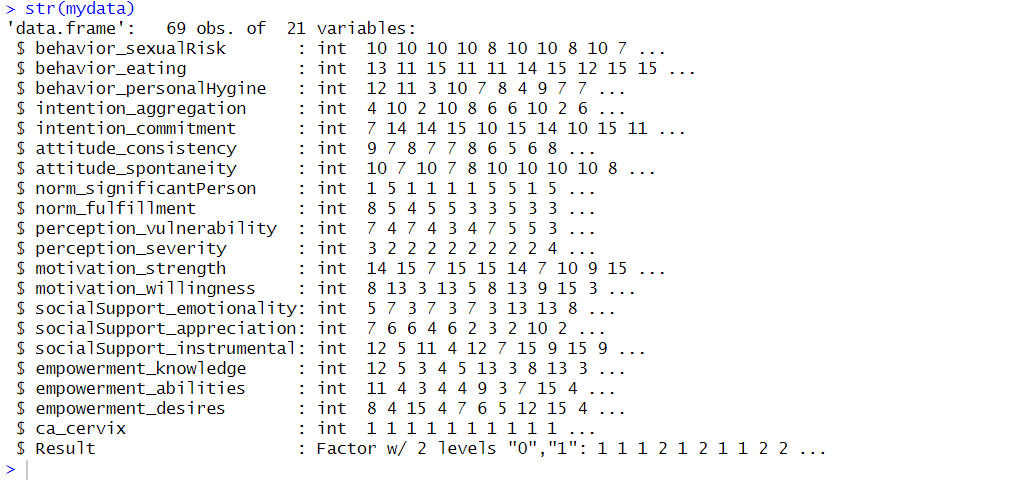
Here, the “Result” column is converted into numeric(0 and 1) where ‘0’ represents ‘Positive’ and ‘1’ represents ‘Negetive’. With the help of the code mydata$Result <- factor(mydata$Result, levels = c("Positive","Negetive"),labels = c("0","1")), we were able to successfully convert ‘Positive’ and ‘Negetive’ into numeric ‘0’ and ‘1’.

**Summary of the structure of the dataset:**

**Code:**

str(mydata)

**Output:**



**Description:**

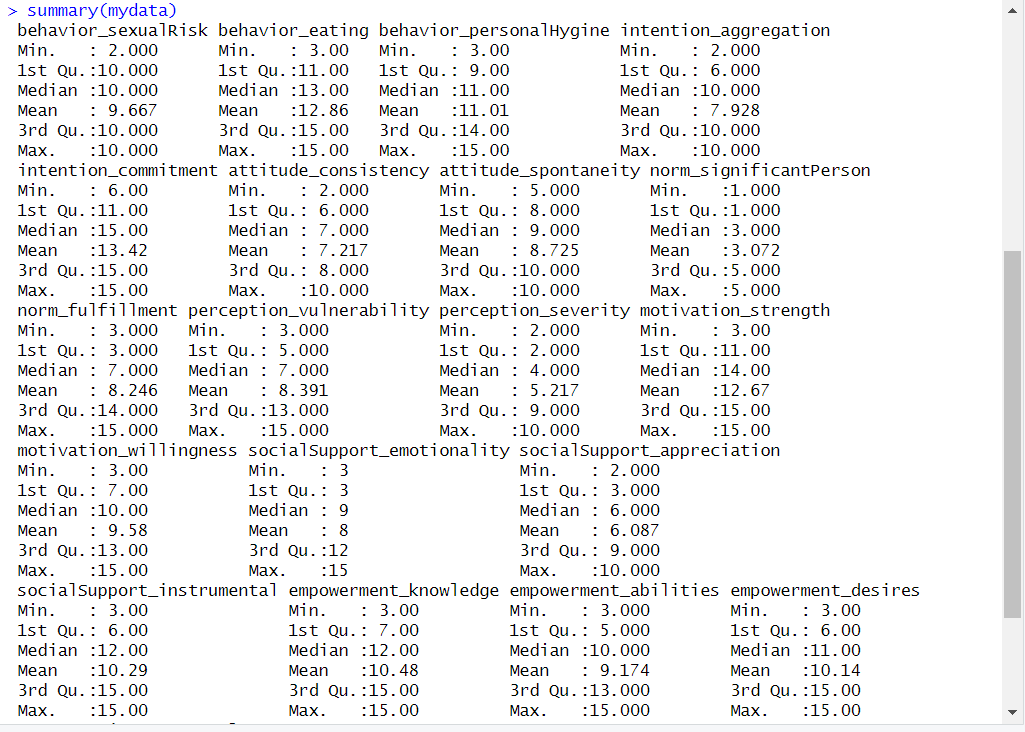
The structure of the dataset is displayed with the help of the code str().

**Descriptive Statistics:**

**Code:**

summary(mydata)

**Output:**



**Description:**

Here, we are using this code to see the descriptive statistics. To see the descriptive statistics, we use the summary() function. We can also see the min, max, mean, and median values of the dataset.

**Using the correlation technique:**

**Code:**

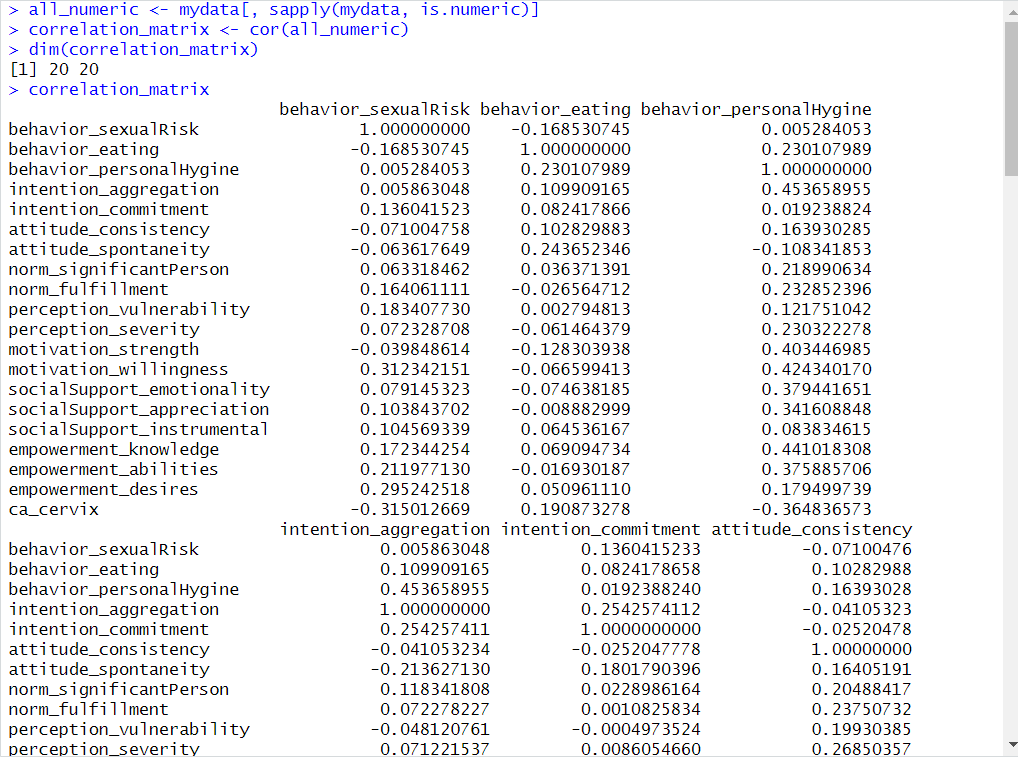
all\_numeric <- mydata[, sapply(mydata, is.numeric)]

correlation\_matrix <- cor(all\_numeric)

dim(correlation\_matrix)

correlation\_matrix

**Output:**



**Description:**

Here, we are using this code to create a correlation matrix. After implementing the code, we were able to create the correlation matrix for all numeric variables.

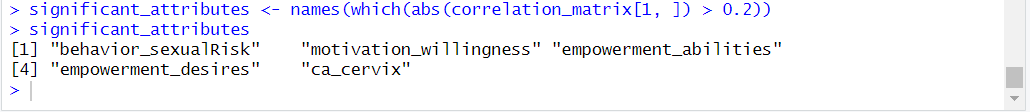
**Find the significant attributes using the correlation technique:**

**Code:**

significant\_attributes <- names(which(abs(correlation\_matrix[1, ]) > 0.2))

significant\_attributes

**Output:**



**Description:**

Here, we are using this code to find the significant attributes of the correlation matrix. After implementing the code, we were able to find the significant attributes.

**Applying the Naïve Bayes algorithm:**

**Code:**

install.packages("e1071")

install.packages("caret")

install.packages("naivebayes")

library("e1071")

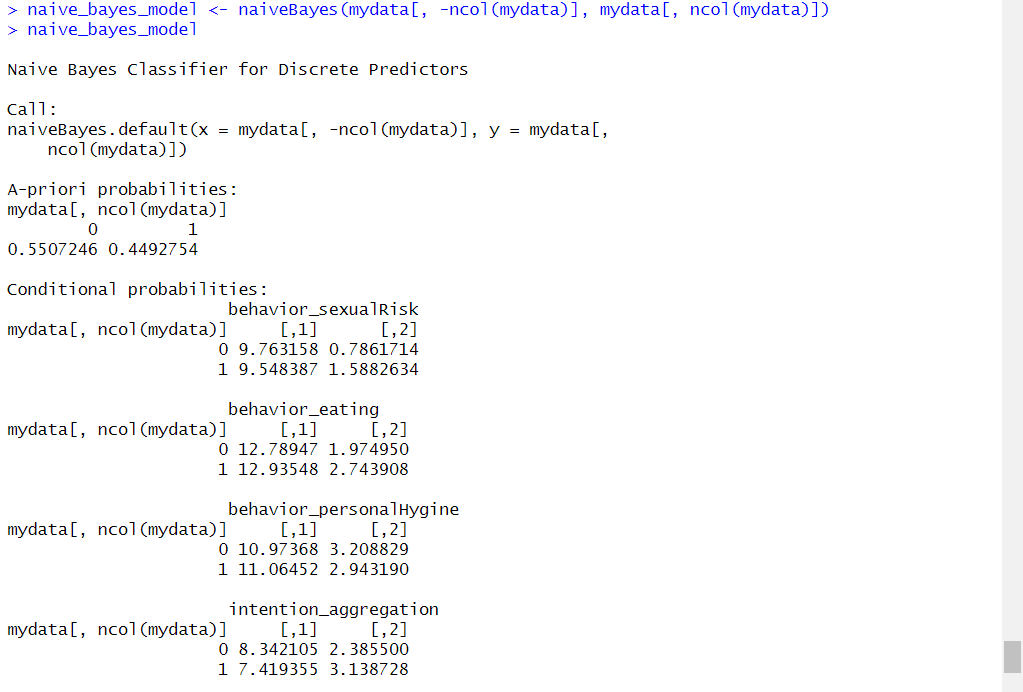
library("caret")

library("naivebayes")

naive\_bayes\_model <- naiveBayes(mydata[, -ncol(mydata)], mydata[, ncol(mydata)])

naive\_bayes\_model

**Output:**



**Description:**

Here, we have used this code to apply the naïve Bayes algorithm. After implementing the code, we were able to apply the naïve Bayes algorithm for all attributes.

**Applying the Naïve Bayes algorithm for significant attributes, dividing the data into training and test sets, finding its accuracy, and generating the confusion matrix:**

**Code:**

set.seed(123)

predictors <- colnames(mydata)[colnames(mydata) != "ca\_cervix"]

target <- "ca\_cervix"

split\_index <- sample(1:nrow(mydata), 0.7 \* nrow(mydata))

train\_mydata <- mydata[split\_index, ]

test\_mydata <- mydata[-split\_index, ]

nb\_model <- naiveBayes(train\_mydata[, predictors], train\_mydata$ca\_cervix)

unseen\_instance <- test\_mydata[1, predictors, drop = FALSE]

prediction <- predict(nb\_model, unseen\_instance)

cat("Original Class:", test\_mydata[1, "ca\_cervix"], "\n")

cat("Predicted Class:", prediction, "\n")

predictions <- predict(nb\_model, test\_mydata[, predictors])

conf\_matrix <- table(predictions, test\_mydata$ca\_cervix)

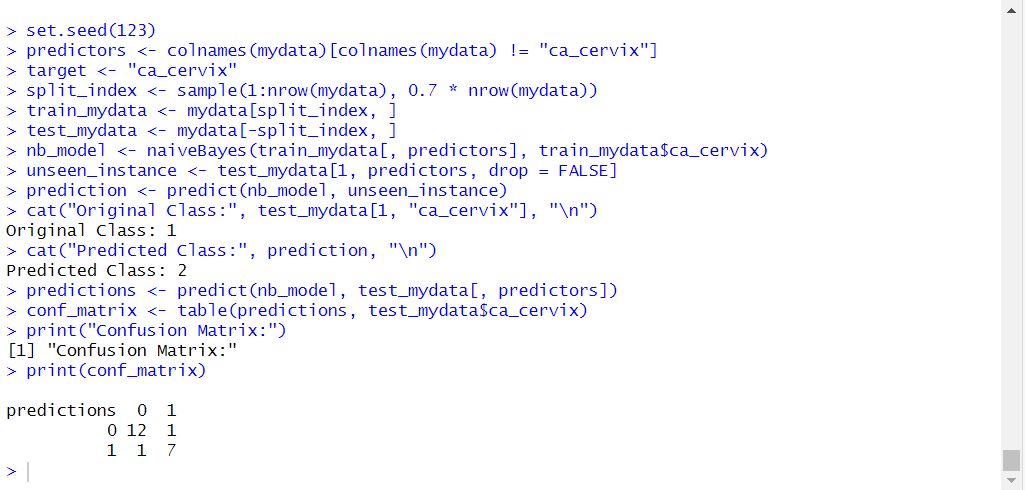
print("Confusion Matrix:")

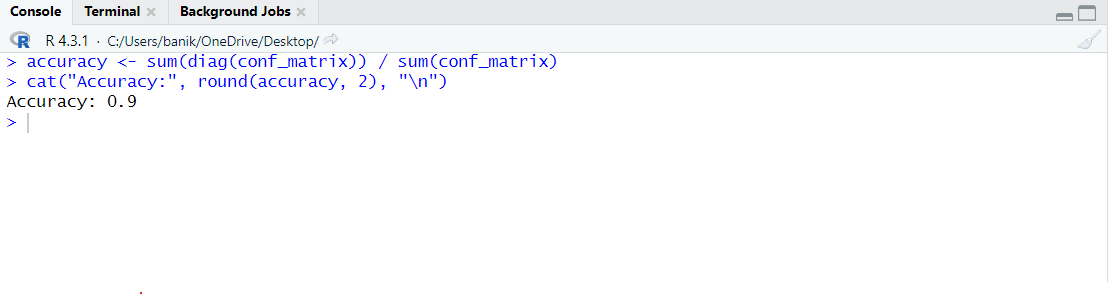
print(conf\_matrix)

accuracy <- sum(diag(conf\_matrix)) / sum(conf\_matrix)

cat("Accuracy:", round(accuracy, 2), "\n")

**Output:**





**Description:**

Here, at first, we have used the code to apply the naïve Bayes algorithm for one of the significant attributes “ca\_cervix”. We have also divided the data into training and test sets to find its accuracy. Finally, we have generated the confusion matrix for our dataset using the naïve Bayes classifier.

**Applying the 10-fold cross-validation:**

**Code:**

set.seed(123)

library("caret")

cv\_results <- train(

x = train\_data[, predictors],

y = train\_data[, target],

method = "rf",

trControl = trainControl(

method = "cv",

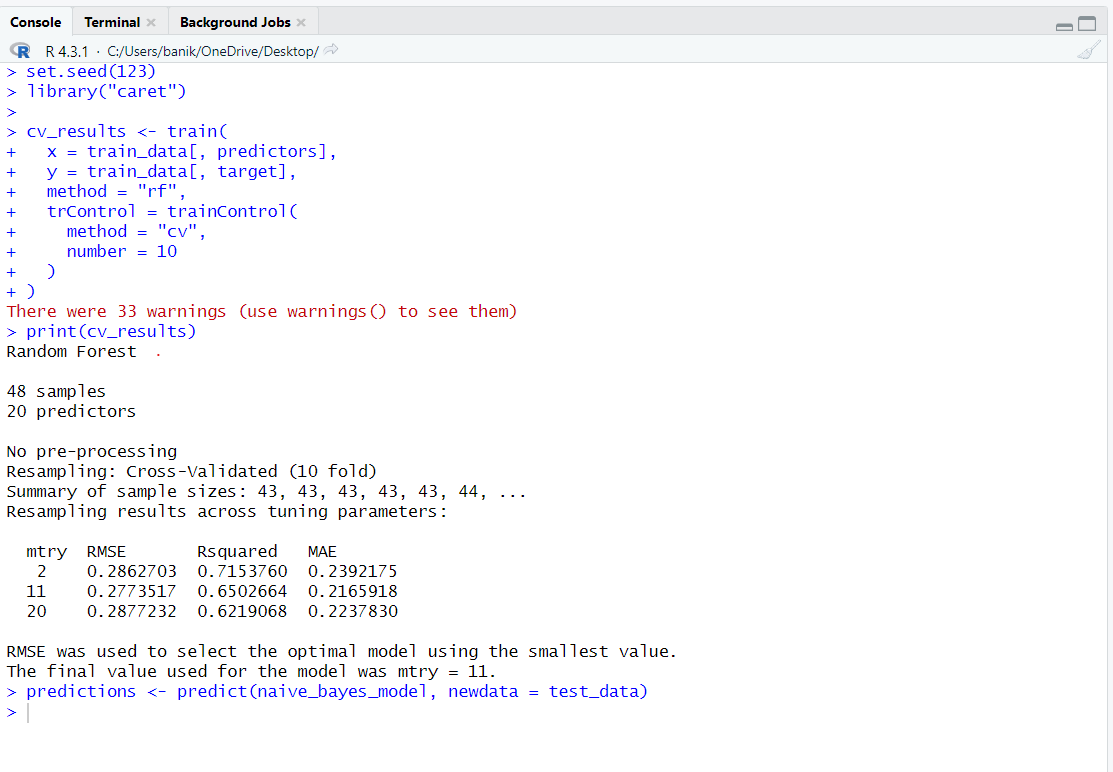
number = 10

)

)

print(cv\_results)

**Output:**



**Description:**

Here, we are displaying the result of cross-validation for the random forest model. The performance of a random forest model of our dataset is demonstrated by this output, which uses various values of mtry. We may get insight into the performance of the model using the metrics RMSE, Rsquared, and MAE. With lower RMSE and MAE values higher Rsquared values indicate greater performance. We may assess and pick the ideal hyperparameter configuration for the random forest model using the values of these metrics for each model setup.

**Calculate the** **Precision, Recall, and F-measure value of the confusion matrix:**

**Code:**

precision <- conf\_matrix[2, 2] / sum(conf\_matrix[, 2])

recall <- conf\_matrix[2, 2] / sum(conf\_matrix[2, ])

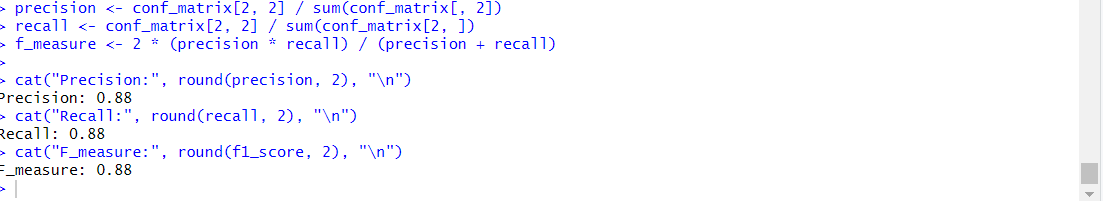
f\_measure <- 2 \* (precision \* recall) / (precision + recall)

cat("Precision:", round(precision, 2), "\n")

cat("Recall:", round(recall, 2), "\n")

cat("F\_measure:", round(f1\_score, 2), "\n")

**Output:**



**Description:**

Here, we are using this code to calculate the recall, precision, and f-measure values of the confusion matrix. After implementing the code, we were able to calculate the recall, precision, and f-measure values of the confusion matrix.