```
> library(caret)
> library(caTools)
> library(e1071)
>
> D=read.csv("C:\\Users\\Sayali Nimhan\\OneDrive\\Desktop\\Gauri Shivale\\scores (1).csv")
> View(D)
>
> set.seed(123)
>
> S=sample.split(D, SplitRatio=0.8) # Splitting sample into train and test
> S
[1]  TRUE FALSE  TRUE
> Train=subset(D, S=="TRUE") # Train Data
> dim(Train)
[1] 307   3
> Test=subset(D, S=="FALSE")  # Test Data
> scale_train=scale(Train[,2:3])
> scale_test=scale(Test[,2:3])
>
>
> NB=naiveBayes(y~., data=Train);NB


Naive Bayes Classifier for Discrete Predictors


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


A-priori probabilities:
Y
     need h.c Don't need h.c
    0.1596091      0.8403909
```

```
Conditional probabilities:
            physical.health.score.out.of.100.
Y                [,1]     [,2]
  need h.c      79.72789 12.01536
 Don't need h.c 51.13695 15.30774

            mental.health
Y                [,1]     [,2]
  need h.c      72.44898  8.727498
 Don't need h.c 52.69380 10.164562

> names(NB)
[1] "apriori"   "tables"    "levels"    "isnumeric" "call"
> Ypred= predict(NB, newdata=Test) # Predicted Response Variable
>
> # Confusion Matrix
> cm=table(Test[,1], Ypred);cm
            Ypred
              need h.c Don't need h.c
  need h.c           15            0
 Don't need h.c       5          133
> confusionMatrix(cm)
Confusion Matrix and Statistics

            Ypred
              need h.c Don't need h.c
  need h.c           15            0
 Don't need h.c       5          133

           Accuracy : 0.9673
```

95% CI : (0.9254, 0.9893)

No Information Rate : 0.8693

P-Value [Acc > NIR] : 3.135e-05

Kappa : 0.8391

Mcnemar's Test P-Value : 0.07364

Sensitivity : 0.75000

Specificity : 1.00000

Pos Pred Value : 1.00000

Neg Pred Value : 0.96377

Prevalence : 0.13072

Detection Rate : 0.09804

Detection Prevalence : 0.09804

Balanced Accuracy : 0.87500

'Positive' Class : need h.c

>