we first load the library, read the data, and convert the Churn variable as factor because it is a response variable that we are interested in and since it is a binary variable so we convert it to factor:

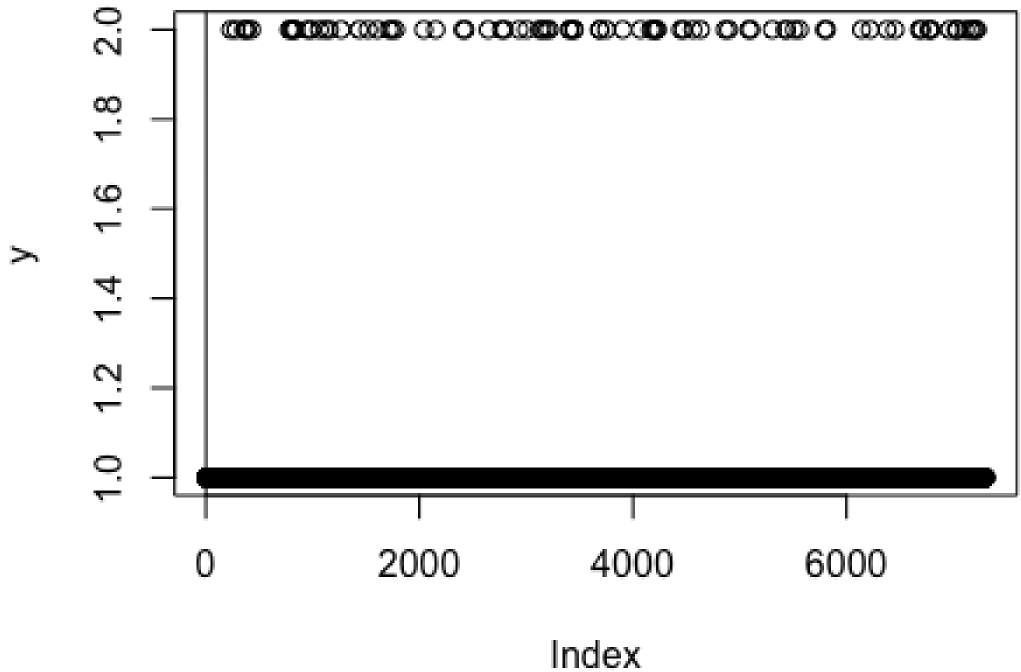
```
library(randomForest)

## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

set.seed (1)
data_df =
read.csv("Documents/Anna_Projects/company_projects/BCG_Analytics/difference.c
sv")
data_df$Churn<- as.factor(data_df$Churn)
```

Then, we split the data into training and testing set and then we train our random forest model. The output we obtain is a confusion matrix that shows the classification error for TP, FP, FP, FN as well as the prediction error rate. The argument mtry=44 indicates that all 44 predictors should be considered for each split of the tree—in other words, that bagging should be done:

```
train = sample(1:nrow(data_df), nrow(data_df)/2)
bag.data= randomForest(Churn~., data=data_df, family = binomial,
subset=train,mtry=44,importance =TRUE)
bag.data

##
## Call:
##  randomForest(formula = Churn ~ ., data = data_df, family = binomial,
mtry = 44, importance = TRUE, subset = train)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 44
##
##         OOB estimate of  error rate: 9.54%
## Confusion matrix:
##      0  1 class.error
## 0 6562 13 0.001977186
## 1  684 44 0.939560440
```

Here’s the graph that we can plot for the tree:

```
yhat.bag = predict(bag.data, newdata=data_df[-train ,])
data.test=data_df[-train, "churn"]
plot(yhat.bag, data.test)
abline(0,1)
```



We can also change the number of trees grown by using a ntree argument, in which case the error will be slightly higher:

```
bag.data=randomForest(Churn~., data=data_df,subset=train, mtry=44,ntree=15)
yhat.bag = predict(bag.data,newdata=data_df[-train ,])
bag.data

##
## Call:
##  randomForest(formula = Churn ~ ., data = data_df, mtry = 44,      ntree =
15, subset = train)
##                Type of random forest: classification
##                      Number of trees: 15
## No. of variables tried at each split: 44
##
##         OOB estimate of  error rate: 11.64%
## Confusion matrix:
##      0   1 class.error
## 0 6374 194  0.02953715
## 1  655  73  0.89972527

set.seed(1)
```

if we change the mtry argument to a smaller value, we can see that the misclassification error rate decreases as well but the classification error rate for FN and TP gets slightly higher. Here, we use importance() function to see the importance of each variable.

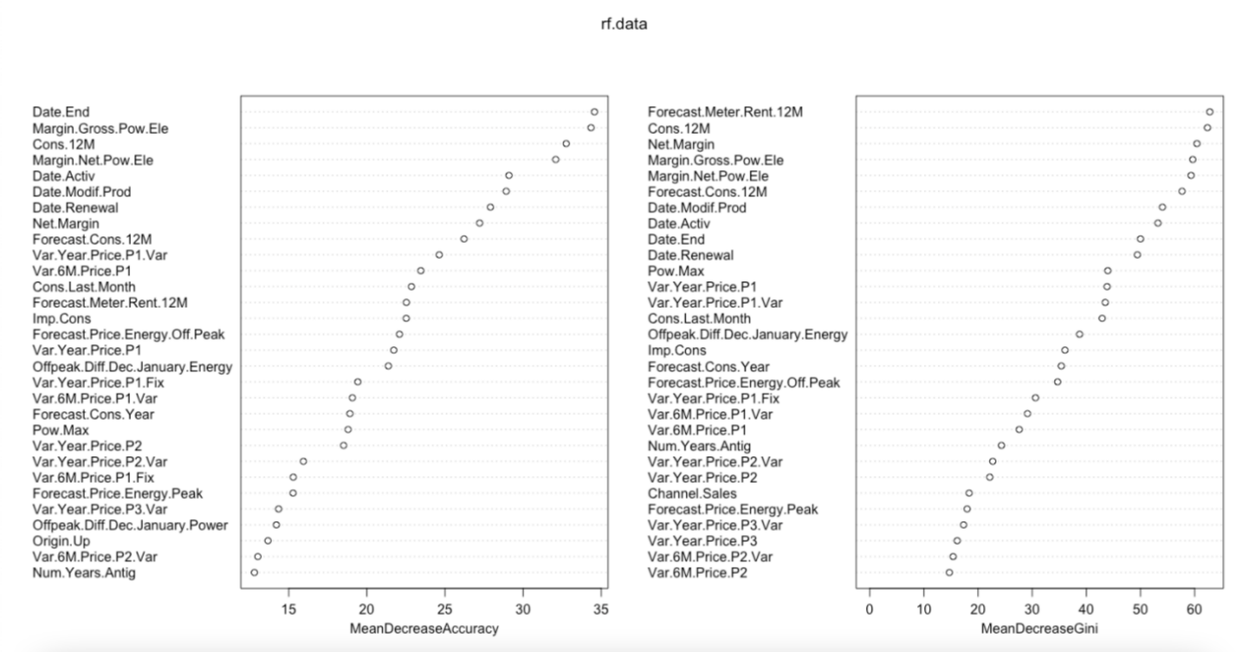
```
rf.data=randomForest(Churn~., data=data_df,subset=train,mtry=5,importance
=TRUE)
rf.data

##
## Call:
##  randomForest(formula = Churn ~ ., data = data_df, mtry = 5, importance =
TRUE,      subset = train)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##         OOB estimate of  error rate: 9.54%
## Confusion matrix:
##      0  1 class.error
## 0 6571  4 0.000608365
## 1  693 35 0.951923077

yhat.rf = predict(rf.data, newdata=data_df[-train ,])
importance(rf.data)

##                                         0           1 MeanDecreaseAccuracy
## Channel.Sales                    6.442263   6.4527769             8.113206
## Cons.Last.Month                 25.962697  -7.6158487            26.194694
## Date.Activ                      28.764675  -4.9893486            28.064088
## Date.End                        31.229139 -12.5518476            30.770916
## Date.Modif.Prod                 30.372942  -3.2826442            30.074999
## Date.Renewal                    28.739194  -7.7117329            29.155099
## Forecast.Cons.Year              24.701944 -13.5496759            24.032631
## Has.Gas                          5.854857  -2.3737600             5.946234
## Origin.Up                        9.316431  11.4913320            12.485660
## Cons.12M                        31.120415  -5.7321420            31.510171
## Cons.Gas.12M                     8.985368  -5.5271254             8.799546
## Forecast.Cons.12M               26.680717 -11.9002192            26.706232
## Forecast.Discount.Energy        10.048316  -4.3531475             9.951182
## Forecast.Meter.Rent.12M         20.927764  -3.4229480            21.373313
## Forecast.Price.Energy.Off.Peak  21.531943  -8.6170882            21.905888
## Forecast.Price.Energy.Peak      16.720356 -11.0741221            16.738033
## Forecast.Price.Pow.Off.Peak     12.614412  -4.0476415            12.893304
## Imp.Cons                        24.474244 -14.0785600            24.030030
## Margin.Gross.Pow.Ele            31.574437 -11.0941618            32.392435
## Margin.Net.Pow.Ele              32.013090 -11.3467978            32.485148
## Nb.Prod.Act                      6.268006   0.4162076             6.417510
## Net.Margin                      24.781323  -9.0289917            25.184954
## Num.Years.Antig                  8.333875   8.2466320            10.434073
## Offpeak.Diff.Dec.January.Energy 24.330731  -8.1274683            24.517394
## Offpeak.Diff.Dec.January.Power  13.066343  -7.0166019            13.037178
```

```
## Pow.Max                           21.114006  -9.4603994              21.247573
## Var.6M.Price.P1                   22.353182 -10.2184067              22.518003
## Var.6M.Price.P1.Fix               12.698582  -7.2425909              12.680883
## Var.6M.Price.P1.Var               19.425770 -11.2892109              19.641316
## Var.6M.Price.P2                   14.341952 -12.2497979              14.394719
## Var.6M.Price.P2.Fix                6.343613  -4.0222001               6.156045
## Var.6M.Price.P2.Var               12.573083  -9.2625035              12.618710
## Var.6M.Price.P3                    9.436342  -6.8017903               9.484866
## Var.6M.Price.P3.Fix                7.207128  -4.1025903               7.210247
## Var.6M.Price.P3.Var               10.403198  -4.1654483              10.837543
## Var.Year.Price.P1                 24.438889 -11.7633583              24.883716
## Var.Year.Price.P1.Fix             19.725921  -8.8568905              20.240698
## Var.Year.Price.P1.Var             25.822941 -11.1087274              26.203856
## Var.Year.Price.P2                 18.994053 -15.4821646              18.926241
## Var.Year.Price.P2.Fix             11.030335  -8.3911433              10.949803
## Var.Year.Price.P2.Var             14.962360 -10.6149502              15.025058
## Var.Year.Price.P3                 12.515133 -12.4809155              12.262943
## Var.Year.Price.P3.Fix             10.711402  -8.3518868              10.665118
## Var.Year.Price.P3.Var             12.156948 -10.9022064              12.090402
##                                   MeanDecreaseGini
## Channel.Sales                            19.147491
## Cons.Last.Month                          43.005057
## Date.Activ                               53.438259
## Date.End                                 51.525905
## Date.Modif.Prod                          57.426078
## Date.Renewal                             51.023331
## Forecast.Cons.Year                       37.144320
## Has.Gas                                   5.712917
## Origin.Up                                15.767834
## Cons.12M                                 61.562875
## Cons.Gas.12M                             14.693494
## Forecast.Cons.12M                        57.900251
## Forecast.Discount.Energy                  3.205675
## Forecast.Meter.Rent.12M                  61.591472
## Forecast.Price.Energy.Off.Peak           35.646963
## Forecast.Price.Energy.Peak               19.841808
## Forecast.Price.Pow.Off.Peak               9.398551
## Imp.Cons                                 36.302877
## Margin.Gross.Pow.Ele                     60.822963
## Margin.Net.Pow.Ele                       60.010402
## Nb.Prod.Act                              11.114328
## Net.Margin                               62.665403
## Num.Years.Antig                          25.691361
## Offpeak.Diff.Dec.January.Energy          40.697741
## Offpeak.Diff.Dec.January.Power           13.632709
## Pow.Max                                  45.618434
## Var.6M.Price.P1                          26.099138
## Var.6M.Price.P1.Fix                      11.262551
## Var.6M.Price.P1.Var                      28.199710
## Var.6M.Price.P2                          13.749710
```

```
## Var.6M.Price.P2.Fix                          3.593094
## Var.6M.Price.P2.Var                         14.336819
## Var.6M.Price.P3                              8.148678
## Var.6M.Price.P3.Fix                          3.595285
## Var.6M.Price.P3.Var                         10.374127
## Var.Year.Price.P1                           46.917382
## Var.Year.Price.P1.Fix                       32.471763
## Var.Year.Price.P1.Var                       43.974712
## Var.Year.Price.P2                           23.503875
## Var.Year.Price.P2.Fix                        9.118407
## Var.Year.Price.P2.Var                       23.387800
## Var.Year.Price.P3                           17.010701
## Var.Year.Price.P3.Fix                        9.703677
## Var.Year.Price.P3.Var                       18.236016
```

We can also plot the importance of each variable by using varImpPlot function. Two measures of importance are used in this case, and they are Mean Decrease Accuracy and Mean Decrease Gini. In terms of Mean Decrease Accuracy, the two most important variables are Date_End and Margin_Gross_Power_Electricity. In terms of Mean Decrease Gini, the two most important variables are Forecast_Meter_Rent_12M and Cons_12M.

```
varImpPlot(rf.data)
```



In conclusion, you can build a random forest model depends on how you want to specify mtry and ntree, and there are some advantages and disadvantages of using a random forest as outlined below:

Advantages	**Disadvantages**

It can be used to solve regression and classification problems	It may take time for the output to print since random forest creates a lot of trees and therefore, it uses computational resources a lot
It reduces variance and hence, improves accuracy a lot	We may have to spend lots of time training a random forest since it creates a lot of trees compared to other decision tree models
It can handle non-linear relationship between independent variables well	Random forest may not handle smaller dataset or data with fewer features very well