Text Analytics and Deal Prediction Analytics

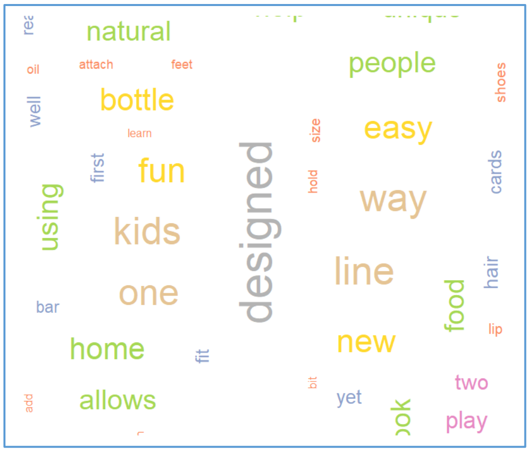
Amit KULKARNI

Table of Contents

Text Analytics:

```
> library(SnowballC)
> library(tm)
> library(ggplot2)
> library(RColorBrewer)
> library(wordcloud)
> library(data.table)
> library(dplyr)
> library(syuzhet)
> library(plyr)
> library(topicmodels)
> library(stringi)
> library(gridExtra)
> setwd("C:/Users/Amit Kulkarni/Documents/R Programming/text analytics")
> Senti<- read.csv("Dataset.csv")
> #Senti$description<- as.date(Senti$description, format="%d-%m-%y")
> Senti$description<- as.character(Senti$description)
> #str(tweets.df)
> mycorpus<- Corpus(VectorSource(Senti$description))
> mycorpus<- tm_map(mycorpus, content_transformer(stri_trans_tolower))
Warning message:
In tm_map.SimpleCorpus(mycorpus, content_transformer(stri_trans_tolower)) :
  transformation drops documents
> removeURL<-function(x)gsub("http[^[:space:]]*","", x)
> mycorpus<-tm_map(mycorpus, content_transformer(removeURL))
Warning message:
In tm_map.SimpleCorpus(mycorpus, content_transformer(removeURL)) :
  transformation drops documents
> removeusername<-function(x) gsub("@[^[:space:]]*","",x)
> mycorpus<- tm_map(mycorpus, content_transformer(removeusername))
Warning message:
In tm_map.SimpleCorpus(mycorpus, content_transformer(removeusername)) :
  transformation drops documents
> removeNumPunct<- function(x) gsub("[^[:alpha:][:space:]]*","",x)
> mycorpus<-tm_map(mycorpus, content_transformer(removeNumPunct))
Warning message:
In tm_map.SimpleCorpus(mycorpus, content_transformer(removeNumPunct)) :
  transformation drops documents
> mystopwords<- c((stopwords('english')), c("use","company","just","even","made","without","make
","like","get","makes","products","product","can","used", "via", "amp","also","dont","thats","will","since",
"ever"))
> mycorpus<-tm_map(mycorpus,removeWords, mystopwords)
Warning message:
In tm_map.SimpleCorpus(mycorpus, removeWords, mystopwords) :
  transformation drops documents
> mycorpus<- tm_map(mycorpus, stripWhitespace)
Warning message:
In tm_map.SimpleCorpus(mycorpus, stripWhitespace) :
  transformation drops documents
> writeLines(strwrap(mycorpus[[15]]$content,60))
state capitals fun minutes efficient entertaining method
learn us geography set flash cards combines phonetics
cartoons associations keep kids interest drive longterm
learning retention author ken bradford worked closely
public private school teachers develop fun satisfying study aide
```

```
> #wordcloud(mycorpus, min.freq=5)
> #wordcloud(mycorpus, min.freq=2, color = brewer.pal(8, "Set2"), random.order=F, rot.per =.30)
> wordcloud(mycorpus, min.freq=5, color = brewer.pal(8, "Set2"), random.order=F, rot.per =.30)
```



Description column from the dataset has been used to create the word cloud. This wordcloud typically gives a fair idea of what products dominated the shows. The bigger the font size in the cloud the most frequent was the word used and in our case the most number of ideas.

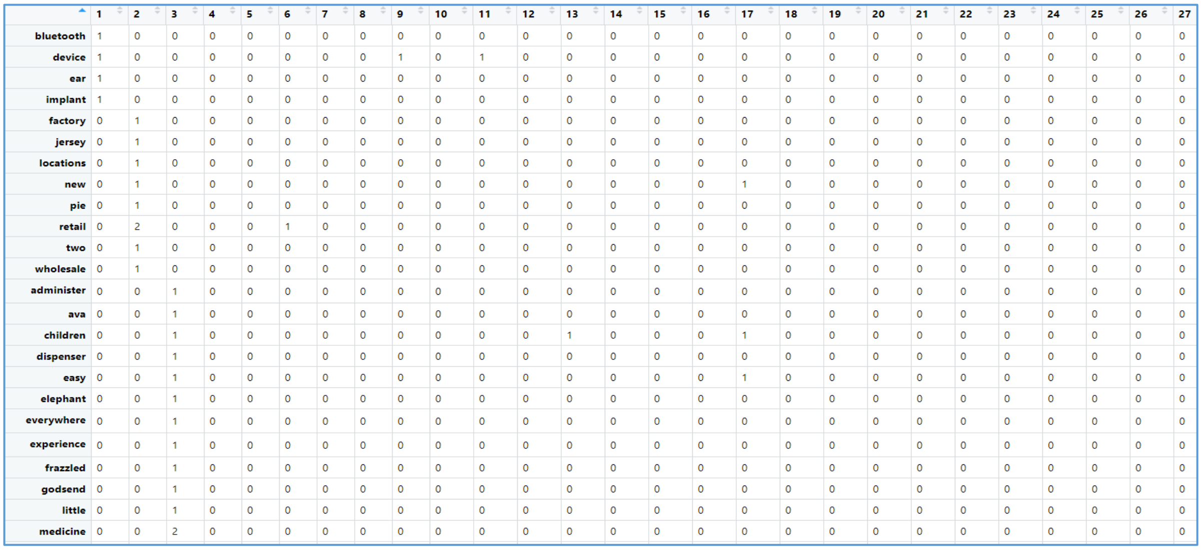
If we look at it we can see some clear outstanding product lines pertained to kids, fun loving, natural, bottled products, somethings to do with hair, shoes etc. can be identified. In our case it means ideas pertaining to these categories were presented the most

Creating a document matrix.

To create a document term matrix we need the package "tm". Once loaded below command will provide the results

```
> DTMatrix<- as.matrix(TermDocumentMatrix(mycorpus))
```

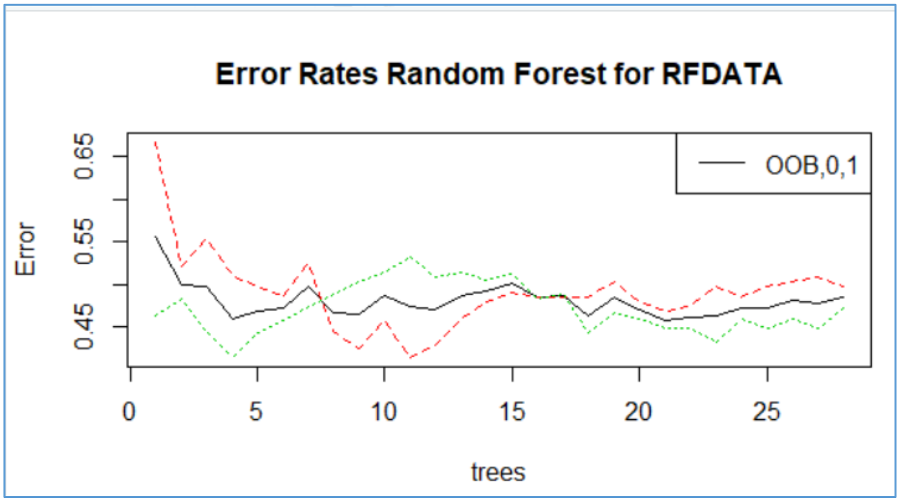
```
> View(DTMatrix)
```
A sample matrix detailed level is provided below



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
bluetooth	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
device	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ear	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
implant	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
factory	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jersey	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
locations	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
new	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
pie	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
retail	0	2	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
two	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
wholesale	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
administer	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ava	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
children	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
dispenser	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
easy	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
elephant	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
everywhere	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
experience	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
frazzled	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
godsend	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
little	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
medicine	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Code for Random Forest with the ratio

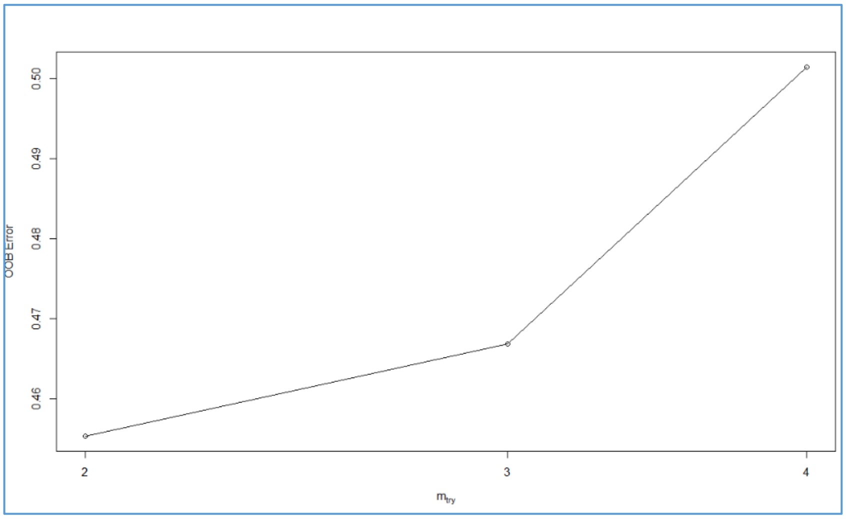
```
> library(randomForest)
randomForest 4.6-14
Type rfNews() to see new features/changes/bug fixes.
Warning message:
package ‘randomForest’ was built under R version 3.5.3
> setwd("C:/Users/Amit Kulkarni/Documents/R Programming/text analytics")
> RFdata= read.csv("DatasetR.csv", header=TRUE)
> #View(RFdata)
> deal.d = model.matrix(~deal -1, data=RFdata)
> RFdata1 = data.frame(RFdata, deal.d)
> View(RFdata1)
> #str(RFdata1)
> set.seed(123)
> split = sample.split(RFdata1$dealTRUE, SplitRatio = 0.70)
Error in sample.split(RFdata1$dealTRUE, SplitRatio = 0.7) :
  could not find function "sample.split"
> library(caTools)
Warning message:
package ‘caTools’ was built under R version 3.5.2
> #str(RFdata1)
> set.seed(123)
> split = sample.split(RFdata1$dealTRUE, SplitRatio = 0.70)
> RFtrain= subset(RFdata1, split==T)
> RFtest= subset(RFdata1, split==F)
> RFtrain=subset(RFtrain, select=-c(1,2,4,5,6,7,11,17,18,19,21))
> #View(CARTtrain)
> RFtest=subset(RFtest, select=-c(1,2,4,5,6,7,11,17,18,19,21))
> RF<- randomForest(as.factor(dealTRUE)~., data=RFtrain,
+                 ntree=28, mtry = 3, nodesize = 10, importance= TRUE)
> print(RF)

Call:
 randomForest(formula = as.factor(dealTRUE) ~ ., data = RFtrain,      ntree = 28, mtry = 3, no
desize = 10, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 28
No. of variables tried at each split: 3
```



```
        OOB estimate of  error rate: 48.41%
Confusion matrix:
   0  1 class.error
0 86 85   0.4970760
1 83 93   0.4715909
> plot(RF, main="")
> legend("topright", c("OOB,0,1"), text.col = 1:6, lty = 1:3, col=1:3)
> title(main="Error Rates Random Forest for RFDATA")

> RF$err.rate
            OOB         0         1
 [1,] 0.5564516 0.6666667 0.4626866
 [2,] 0.5000000 0.5208333 0.4824561
 [3,] 0.4980989 0.5546875 0.4444444
 [4,] 0.4600000 0.5106383 0.4150943
 [5,] 0.4687500 0.4967742 0.4424242
 [6,] 0.4725610 0.4875000 0.4583333
 [7,] 0.4984985 0.5246914 0.4736842
 [8,] 0.4672619 0.4451220 0.4883721
 [9,] 0.4649123 0.4251497 0.5028571
[10,] 0.4868805 0.4583333 0.5142857
[11,] 0.4738372 0.4142012 0.5314286
[12,] 0.4695652 0.4294118 0.5085714
[13,] 0.4869565 0.4588235 0.5142857
[14,] 0.4927954 0.4795322 0.5056818
[15,] 0.5014409 0.4912281 0.5113636
[16,] 0.4841499 0.4853801 0.4829545
[17,] 0.4870317 0.4853801 0.4886364
[18,] 0.4639769 0.4853801 0.4431818
[19,] 0.4841499 0.5029240 0.4659091
[20,] 0.4697406 0.4795322 0.4602273
[21,] 0.4582133 0.4678363 0.4488636
[22,] 0.4610951 0.4736842 0.4488636
[23,] 0.4639769 0.4970760 0.4318182
[24,] 0.4726225 0.4853801 0.4602273
[25,] 0.4726225 0.4970760 0.4488636
[26,] 0.4812680 0.5029240 0.4602273
[27,] 0.4783862 0.5087719 0.4488636
[28,] 0.4841499 0.4970760 0.4715909

> impvar<- round(randomForest::importance(RF),2)
> impvar[order(impvar[,3], decreasing =TRUE),]

                    0     1 MeanDecreaseAccuracy MeanDecreaseGini
exchangeForStake  1.02  0.31                 1.72             9.51
Ratio            -2.12  1.79                 0.43            10.58
shark4            0.90 -0.33                 0.37             3.50
shark5           -0.94  0.85                 0.02             3.43
shark1           -1.39  0.79                -0.50             1.91
valuation        -2.78  2.30                -0.65            18.85
shark3           -0.10 -0.60                -0.93             1.39
askedFor         -1.61  0.16                -0.99            16.44
episode          -1.22 -0.51                -1.02            15.84
shark2           -1.58 -0.54                -1.45             2.56
```

```
> tRF<- tuneRF(x=RFtrain[, -c(11)],
+             y=as.factor(RFtrain$dealTRUE),
+             mtrystart=3,
+             ntreeTry = 101,
+             stepFactor = 1.5,
+             improve=0.001,
+             trace= TRUE,
+             plot=TRUE,
+             doBest = TRUE,
+             nodesize=10,
+             importance=TRUE,
+ )
```

```
mtry = 3  OOB error = 46.69%
Searching left ...
mtry = 2   OOB error = 45.53%
0.02469136 0.001
Searching right ...
mtry = 4   OOB error = 50.14%
-0.1012658 0.001
```



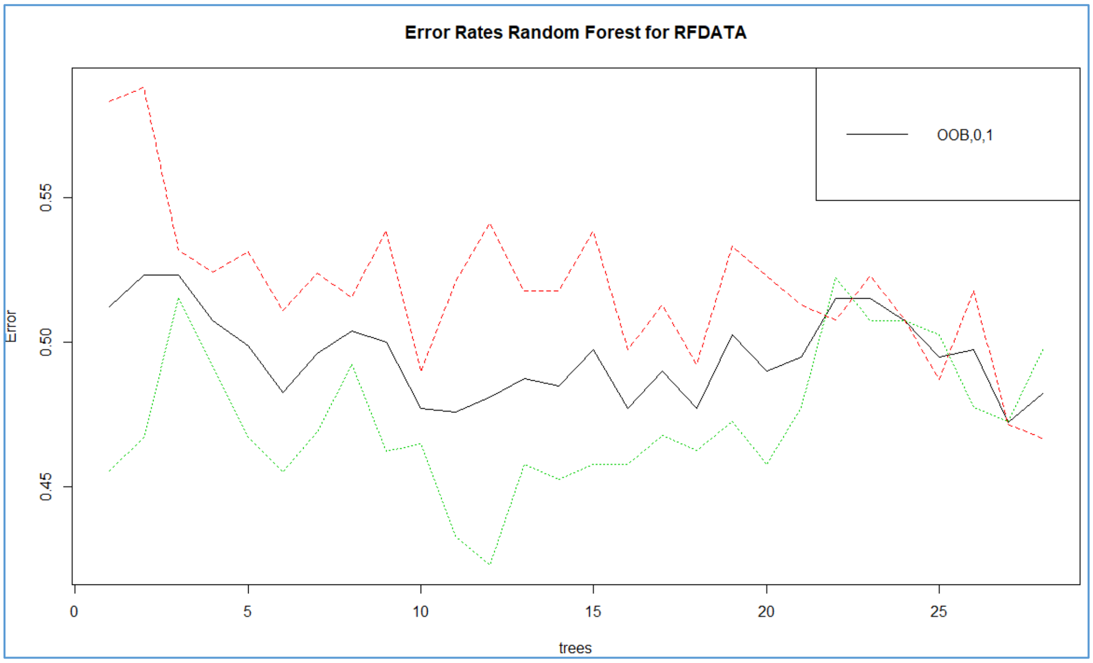
```
> RFtrain$predict.class=predict(tRF, RFtrain, type="class")
> RFtrain$predict.score=predict(tRF, RFtrain, type="prob")
> RF<- randomForest(as.factor(dealTRUE)~., data=RFtrain,
+                 ntree=28, mtry = 3, nodesize = 10, importance= TRUE)
```

Random Forest without the Ratio

```
> library(randomForest)

> library(caTools)

> setwd("C:/Users/Amit Kulkarni/Documents/R Programming/text analytics")

> RFdata= read.csv("Dataset.csv", header=TRUE)
> deal.d = model.matrix(~deal -1, data=RFdata)

> RFdata1 = data.frame(RFdata, deal.d)

> set.seed(3000)
> split = sample.split(RFdata1$dealTRUE, SplitRatio = 0.80)
> RFtrain= subset(RFdata1, split==T)
> RFtest= subset(RFdata1, split==F)
> View(RFtrain)
> RFtrain=subset(RFtrain, select=-c(1,2,4,5,6,7,11,17,18,19,20))
> RFtest=subset(RFtest, select=-c(1,2,4,5,6,7,11,17,18,20))
> RF<- randomForest(as.factor(dealTRUE)~., data=RFtrain,
+                  ntree=28, mtry = 3, nodesize = 10, importance= TRUE)
> print(RF)
```

```
Call:
 randomForest(formula = as.factor(dealTRUE) ~ ., data = RFtrain,      ntree = 28, mtry = 3, nodesize = 10, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 28
No. of variables tried at each split: 3

        OOB estimate of  error rate: 48.74%
Confusion matrix:
    0  1 class.error
0 111 84   0.4307692
1 109 92   0.5422886
```

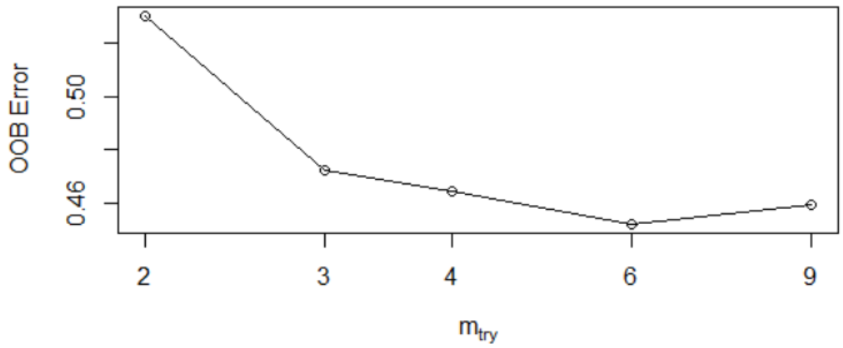
```
> plot(RF, main="")
> legend("topright", c("OOB,0,1"), text.col = 1:6, lty = 1:3, col=1:3)
> title(main="Error Rates Random Forest for RFDATA")
```



```
> RF$err.rate
           OOB         0         1
 [1,] 0.4899329 0.4705882 0.5061728
 [2,] 0.5346939 0.6371681 0.4469697
 [3,] 0.5264901 0.6180556 0.4430380
 [4,] 0.5236686 0.5398773 0.5085714
 [5,] 0.5335196 0.5086705 0.5567568
 [6,] 0.5240642 0.4918033 0.5549738
 [7,] 0.5145119 0.4782609 0.5487179
 [8,] 0.5025641 0.4397906 0.5628141
 [9,] 0.5127551 0.4635417 0.5600000
[10,] 0.4860051 0.4248705 0.5450000
[11,] 0.5101523 0.4639175 0.5550000
[12,] 0.4924242 0.4461538 0.5373134
[13,] 0.5050505 0.4564103 0.5522388
[14,] 0.4772727 0.4307692 0.5223881
[15,] 0.4873737 0.4615385 0.5124378
[16,] 0.5176768 0.4871795 0.5472637
[17,] 0.5000000 0.4820513 0.5174129
[18,] 0.5126263 0.4923077 0.5323383
[19,] 0.5151515 0.4717949 0.5572139
[20,] 0.5101010 0.4820513 0.5373134
[21,] 0.5075758 0.4717949 0.5422886
[22,] 0.5101010 0.4769231 0.5422886
[23,] 0.5000000 0.4717949 0.5273632
[24,] 0.4924242 0.4769231 0.5074627
[25,] 0.4873737 0.4564103 0.5174129
[26,] 0.4823232 0.4512821 0.5124378
[27,] 0.4898990 0.4461538 0.5323383
[28,] 0.4873737 0.4307692 0.5422886
```

```
> impvar<- round(randomForest::importance (RF),2)
> impvar[order(impvar[,3], decreasing =TRUE),]

                   0     1 MeanDecreaseAccuracy MeanDecreaseGini
shark1           2.10  0.62                 2.21             4.26
shark3          -0.62  1.55                 0.49             1.42
askedFor        -0.59  0.88                 0.11            20.34
valuation        0.93 -1.27                -0.16            22.70
shark5          -0.89  0.78                -0.20             2.87
episode         -0.21 -0.59                -0.60            22.28
exchangeForStake -0.71 -0.55               -1.02            12.28
shark4          -0.41 -1.27                -1.30             2.55
shark2           0.86 -1.77                -1.41             2.04
```

```
> tRF<- tuneRF(x=RFtrain[, -c(10)],
+              y=as.factor(RFtrain$dealTRUE),
+              mtrystart=3,
+              ntreeTry = 101,
+              stepFactor = 1.5,
+              improve=0.001,
+              trace= TRUE,
+              plot=TRUE,
+              doBest = TRUE,
+              nodesize=10,
+              importance=TRUE,
+ )
mtry = 3  OOB error = 47.22%
Searching left ...
mtry = 2   OOB error = 53.03%
-0.1229947 0.001
Searching right ...
mtry = 4   OOB error = 46.46%
0.01604278 0.001
mtry = 6   OOB error = 45.2%
0.02717391 0.001
mtry = 9   OOB error = 45.96%
-0.01675978 0.001
```



```
> RF6<- randomForest(as.factor(dealTRUE)~., data=RFtrain,
+                    ntree=28, mtry = 6, nodesize = 10, importance= TRUE)
> print(RF6)

Call:
 randomForest(formula = as.factor(dealTRUE) ~ ., data = RFtrain,      ntree = 28, mtry = 6, nodesize
= 10, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 28
No. of variables tried at each split: 6

        OOB estimate of  error rate: 48.23%
Confusion matrix:
    0   1 class.error
0 104  91   0.4666667
1 100 101   0.4975124

> plot(RF6, main="")
> legend("topright", c("OOB,0,1"), text.col = 1:6, lty = 1:3, col=1:3)
> title(main="Error Rates Random Forest for RFDATA")
> RF6$err.rate
            OOB         0         1
 [1,] 0.5123457 0.5833333 0.4555556
 [2,] 0.5234375 0.5882353 0.4671533
 [3,] 0.5231788 0.5319149 0.5155280
 [4,] 0.5074184 0.5240964 0.4912281
 [5,] 0.4985994 0.5314286 0.4670330
 [6,] 0.4825737 0.5108696 0.4550265
 [7,] 0.4960836 0.5238095 0.4690722
 [8,] 0.5038560 0.5156250 0.4923858
 [9,] 0.5000000 0.5388601 0.4623116
[10,] 0.4771574 0.4896907 0.4650000
[11,] 0.4759494 0.5206186 0.4328358
[12,] 0.4810127 0.5412371 0.4228856
[13,] 0.4873737 0.5179487 0.4577114
[14,] 0.4848485 0.5179487 0.4527363
[15,] 0.4974747 0.5384615 0.4577114
[16,] 0.4772727 0.4974359 0.4577114
[17,] 0.4898990 0.5128205 0.4676617
[18,] 0.4772727 0.4923077 0.4626866
[19,] 0.5025253 0.5333333 0.4726368
[20,] 0.4898990 0.5230769 0.4577114
[21,] 0.4949495 0.5128205 0.4776119
[22,] 0.5151515 0.5076923 0.5223881
[23,] 0.5151515 0.5230769 0.5074627
[24,] 0.5075758 0.5076923 0.5074627
[25,] 0.4949495 0.4871795 0.5024876
[26,] 0.4974747 0.5179487 0.4776119
[27,] 0.4722222 0.4717949 0.4726368
[28,] 0.4823232 0.4666667 0.4975124
```

```
> impvar<- round(randomForest::importance(RF),2)
> impvar[order(impvar[,3], decreasing =TRUE),]

                    0     1 MeanDecreaseAccuracy MeanDecreaseGini
shark1           2.10  0.62                 2.21             4.26
shark3          -0.62  1.55                 0.49             1.42
askedFor        -0.59  0.88                 0.11            20.34
valuation        0.93 -1.27                -0.16            22.70
shark5          -0.89  0.78                -0.20             2.87
episode         -0.21 -0.59                -0.60            22.28
exchangeForStake -0.71 -0.55               -1.02            12.28
shark4          -0.41 -1.27                -1.30             2.55
shark2           0.86 -1.77                -1.41             2.04
```

CART without the Ratio

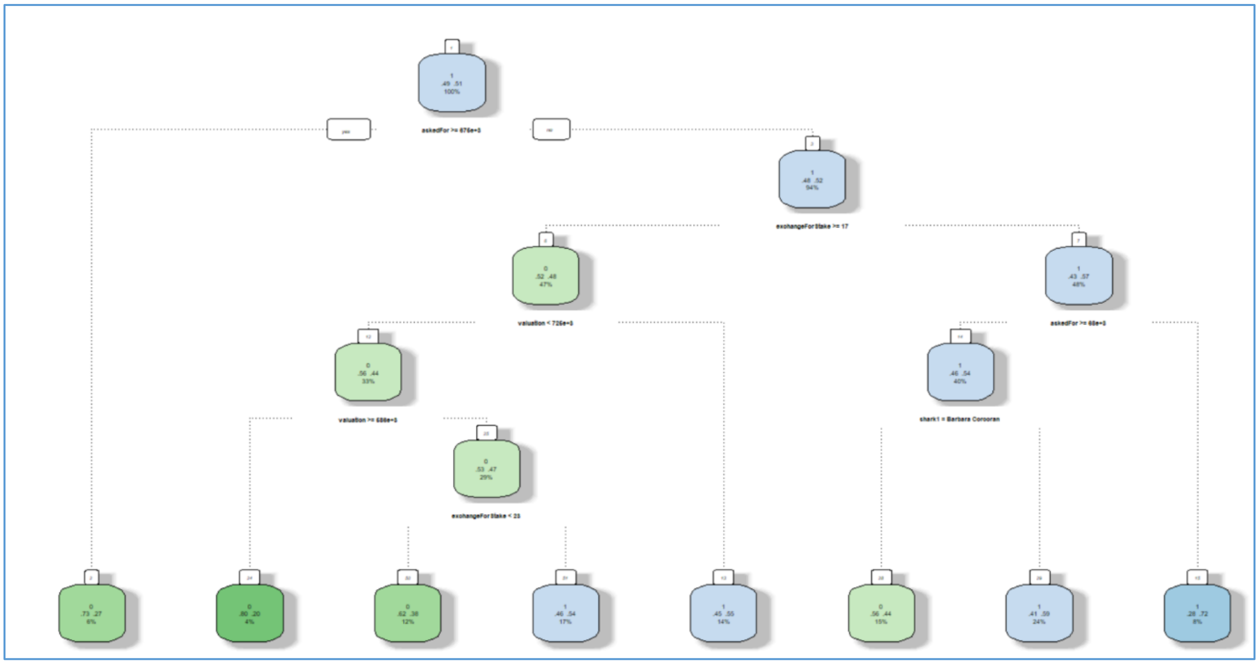
```
> setwd("C:/Users/Amit Kulkarni/Documents/R Programming/packages")
> library(SDMTools)
Warning message:
package ‘SDMTools’ was built under R version 3.5.3
> library(pROC)
Type 'citation("pROC")' for a citation.

Attaching package: ‘pROC’

The following object is masked from ‘package:SDMTools’:

    auc

The following objects are masked from ‘package:stats’:

    cov, smooth, var

Warning message:
package ‘pROC’ was built under R version 3.5.3
> library(Hmisc)
Loading required package: lattice
Loading required package: survival
Loading required package: Formula

Attaching package: ‘Hmisc’

The following objects are masked from ‘package:plyr’:

    is.discrete, summarize

The following objects are masked from ‘package:dplyr’:

    src, summarize

The following objects are masked from ‘package:base’:

    format.pval, units

Warning messages:
1: package ‘Hmisc’ was built under R version 3.5.3
2: package ‘Formula’ was built under R version 3.5.2
> library(caTools)
> library(rpart)

Attaching package: ‘rpart’

The following object is masked from ‘package:survival’:

    solder

Warning message:
package ‘rpart’ was built under R version 3.5.2
> library(rpart.plot)
Warning message:
```

```
package ‘rpart.plot’ was built under R version 3.5.2
> library(rattle)
Rattle: A free graphical interface for data science with R.
Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
Type 'rattle()' to shake, rattle, and roll your data.

Attaching package: ‘rattle’

The following object is masked from ‘package:randomForest’:

    importance

Warning message:
package ‘rattle’ was built under R version 3.5.2
> library(RColorBrewer)
> library(scales)

Attaching package: ‘scales’

The following object is masked from ‘package:syuzhet’:

    rescale

Warning message:
package ‘scales’ was built under R version 3.5.2
> library(data.table)
> setwd("C:/Users/Amit Kulkarni/Documents/R Programming/text analytics")
> CARTdata= read.csv("Dataset.csv", header=TRUE)
> View(CARTdata)
> deal.d = model.matrix(~deal -1, data=CARTdata)
> CARTdata1 = data.frame(CARTdata, deal.d)
> View(CARTdata1)
> #str(CARTdata1)
> set.seed(3000)
> split = sample.split(CARTdata1$dealTRUE, SplitRatio = 0.80)
> CARTtrain= subset(CARTdata1, split==T)
> CARTtest= subset(CARTdata1, split==F)
> CARTtrain=subset(CARTtrain, select=-c(1,2,3,4,5,6,7,11,17,18,20))
>
> View(CARTtrain)
> CARTtest=subset(CARTtest, select=-c(1,2,3,4,5,6,7,11,17,18,20))
>
> #View(CARTtest)
> #summary(CARTtrain)
> r.ctrl=rpart.control(minsplit = 100, minbucket = 10, cp=0, xval=10)
> CART_Train_Model = rpart(formula = dealTRUE~., data=CARTtrain , method="class", control = r.c
trl )
```

```
> CART_Train_Model
n= 396

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 396 195 1 (0.4924242 0.5075758)
   2) askedFor>=675000 22   6 0 (0.7272727 0.2727273) *
   3) askedFor< 675000 374 179 1 (0.4786096 0.5213904)
     6) exchangeForStake>=16.5 185  88 0 (0.5243243 0.4756757)
      12) valuation< 725000 129  57 0 (0.5581395 0.4418605)
        24) valuation>=585714.5 15   3 0 (0.8000000 0.2000000) *
        25) valuation< 585714.5 114  54 0 (0.5263158 0.4736842)
          50) exchangeForStake< 22.5 47  18 0 (0.6170213 0.3829787) *
          51) exchangeForStake>=22.5 67  31 1 (0.4626866 0.5373134) *
      13) valuation>=725000 56  25 1 (0.4464286 0.5535714) *
     7) exchangeForStake< 16.5 189  82 1 (0.4338624 0.5661376)
      14) askedFor>=67500 157  73 1 (0.4649682 0.5350318)
        28) shark1=Barbara Corcoran 61  27 0 (0.5573770 0.4426230) *
        29) shark1=Lori Greiner 96  39 1 (0.4062500 0.5937500) *
      15) askedFor< 67500 32   9 1 (0.2812500 0.7187500) *
```

```
> prp(CART_Train_Model)
```

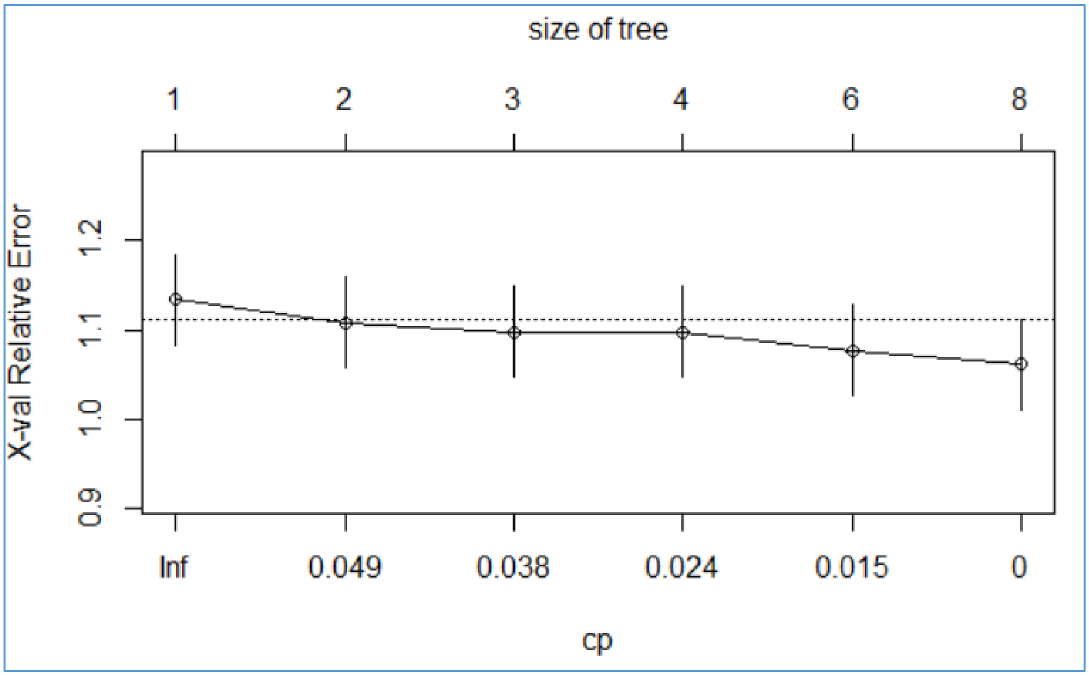
```
> fancyRpartPlot(CART_Train_Model)
```



```
> printcp(CART_Train_Model)

Classification tree:
rpart(formula = dealTRUE ~ ., data = CARTtrain, method = "class",
    control = r.ctrl)
```

```
Variables actually used in tree construction:
[1] askedFor        exchangeForStake shark1
[4] valuation

Root node error: 195/396 = 0.49242

n= 396

        CP nsplit rel error xerror     xstd
1 0.051282      0   1.00000 1.1333 0.050680
2 0.046154      1   0.94872 1.1077 0.050814
3 0.030769      2   0.90256 1.0974 0.050858
4 0.017949      3   0.87179 1.0974 0.050858
5 0.012821      5   0.83590 1.0769 0.050931
6 0.000000      7   0.81026 1.0615 0.050972
```

```
> plotcp(CART_Train_Model)
```

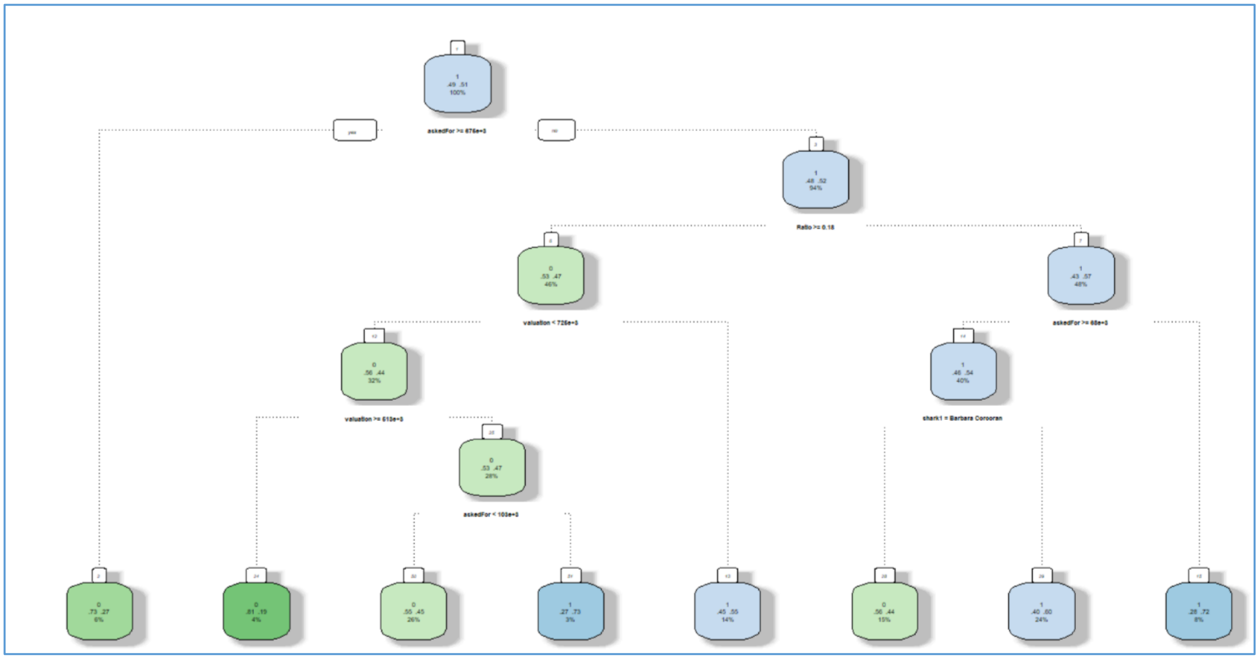


CART with the Ratio

```
> setwd("C:/Users/Amit Kulkarni/Documents/R Programming/packages")
> library(SDMTools)
> library(pROC)
> library(Hmisc)
> library(caTools)
> library(rpart)
> library(rpart.plot)
> library(rattle)
> library(RColorBrewer)
> library(scales)
> library(data.table)
> setwd("C:/Users/Amit Kulkarni/Documents/R Programming/text analytics")
> CARTdata= read.csv("DatasetR.csv", header=TRUE)
> #View(CARTdata)
> deal.d = model.matrix(~deal -1, data=CARTdata)
> CARTdata1 = data.frame(CARTdata, deal.d)
> #View(CARTdata1)
> #str(CARTdata1)
> set.seed(3000)
> split = sample.split(CARTdata1$dealTRUE, SplitRatio = 0.80)
> CARTtrain= subset(CARTdata1, split==T)
> CARTtest= subset(CARTdata1, split==F)
> CARTtrain=subset(CARTtrain, select=-c(1,2,3,4,5,6,7,11,17,18,19,21))
>
> #View(CARTtrain)
> CARTtest=subset(CARTtest, select=-c(1,2,3,4,5,6,7,11,17,18,19,21))
>
> #View(CARTtest)
> #summary(CARTtrain)
> r.ctrl=rpart.control(minsplit = 100, minbucket = 10, cp=0, xval=10)
> CART_Train_Model = rpart(formula = dealTRUE~., data=CARTtrain , method="class", control = r.c
trl )
> CART_Train_Model
```

```
n= 396

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 396 195 1 (0.4924242 0.5075758)
   2) askedFor>=675000 22   6 0 (0.7272727 0.2727273) *
   3) askedFor< 675000 374 179 1 (0.4786096 0.5213904)
     6) Ratio>=0.1774999 184  87 0 (0.5271739 0.4728261)
      12) valuation< 725000 128  56 0 (0.5625000 0.4375000)
        24) valuation>=512500 16   3 0 (0.8125000 0.1875000) *
        25) valuation< 512500 112  53 0 (0.5267857 0.4732143)
          50) askedFor< 102500 101  45 0 (0.5544554 0.4455446) *
          51) askedFor>=102500 11   3 1 (0.2727273 0.7272727) *
      13) valuation>=725000 56  25 1 (0.4464286 0.5535714) *
     7) Ratio< 0.1774999 190  82 1 (0.4315789 0.5684211)
      14) askedFor>=67500 158  73 1 (0.4620253 0.5379747)
        28) shark1=Barbara Corcoran 61  27 0 (0.5573770 0.4426230) *
        29) shark1=Lori Greiner 97  39 1 (0.4020619 0.5979381) *
```

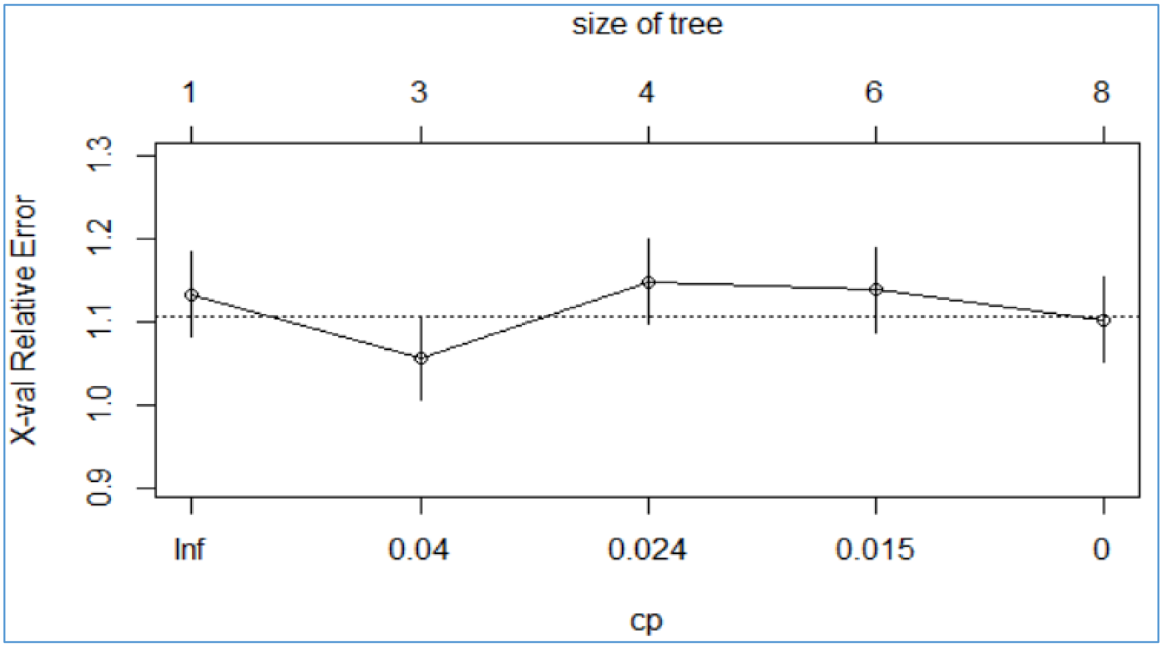
```
    15) askedFor< 67500 32   9 1 (0.2812500 0.7187500) *
```



```
> prp(CART_Train_Model)
> fancyRpartPlot(CART_Train_Model)

> printcp(CART_Train_Model)
```

```
Classification tree:
rpart(formula = dealTRUE ~ ., data = CARTtrain, method = "class",
    control = r.ctrl)

Variables actually used in tree construction:
[1] askedFor  Ratio     shark1    valuation

Root node error: 195/396 = 0.49242

n= 396

        CP nsplit rel error xerror     xstd
1 0.051282      0   1.00000 1.1333 0.050680
2 0.030769      2   0.89744 1.0564 0.050983
3 0.017949      3   0.86667 1.1487 0.050583
4 0.012821      5   0.83077 1.1385 0.050649
5 0.000000      7   0.80513 1.1026 0.050837
```

```
> plotcp(CART_Train_Model)
```

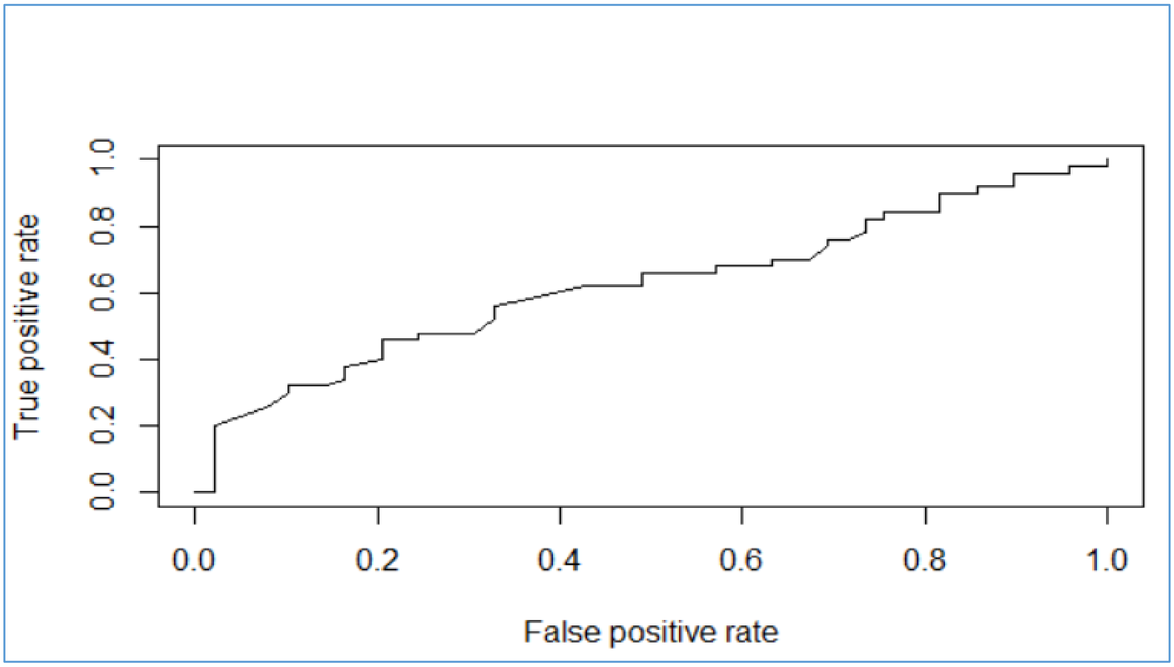


Logistic Regression without the ratio

```
> setwd("C:/Users/Amit Kulkarni/Documents/R Programming/packages")
> #install.packages(c("SDMTools", "pROC", "Hmisc"))
> library(SDMTools)
> library(pROC)
> library(Hmisc)
> library(ROCR)
> setwd("C:/Users/Amit Kulkarni/Documents/R Programming/text analytics")
> LRdata= read.csv("Dataset.csv", header=TRUE)
> #View(LRdata)
> deal.d = model.matrix(~deal -1, data=LRdata)
> LRdata1 = data.frame(LRdata, deal.d)
> View(LRdata1)
> #str(LRdata1)
> set.seed(123)
> split = sample.split(LRdata1$dealTRUE, SplitRatio = 0.80)
> LRtrain= subset(LRdata1, split==T)
> LRtest= subset(LRdata1, split==F)
> View(LRtrain)
> LRtrain=subset(LRtrain, select=-c(1,2,3,4,5,6,7,11:20))
> LRtest=subset(LRtest, select=-c(1,2,3,4,5,6,7,11:20))
> LRtrain=subset(LRtrain, select=-c(4,5,6,7,8,9))
> LRtest=subset(LRtest, select=-c(4,5,6,7,8,9))
> View(LRtrain)
> View(LRtest)
> #All Variables
> LRModel = glm(dealTRUE ~ ., data=LRtrain, family= binomial)
> summary(LRModel)
```

```
Call:
glm(formula = dealTRUE ~ ., family = binomial, data = LRtrain)

Deviance Residuals: 
   Min      1Q  Median      3Q     Max  
-1.322  -1.212   1.020   1.138   1.758  

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)  
(Intercept)      5.660e-01  2.561e-01   2.209   0.0271 *
askedFor         1.189e-07  3.588e-07   0.331   0.7404  
exchangeForStake -2.252e-02  1.195e-02  -1.885   0.0595 .
valuation       -7.776e-08  4.972e-08  -1.564   0.1178  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 548.88  on 395  degrees of freedom
Residual deviance: 542.10  on 392  degrees of freedom
AIC: 550.1

Number of Fisher Scoring iterations: 4
```

```
> predTest<-predict(LRModel, newdata=LRtest, type="response")
> table(LRtest$dealTRUE, predTest>0.5)

   FALSE TRUE
 0    16   33
 1    15   35

> (35+15)/nrow(na.omit(LRtest))

[1] 0.5050505
> ROCRpred=prediction(predTest, LRtest$dealTRUE)

> as.numeric(performance(ROCRpred,"auc")@y.values)

[1] 0.6230612

> perf= performance(ROCRpred, "tpr","fpr")
> plot(perf)
```

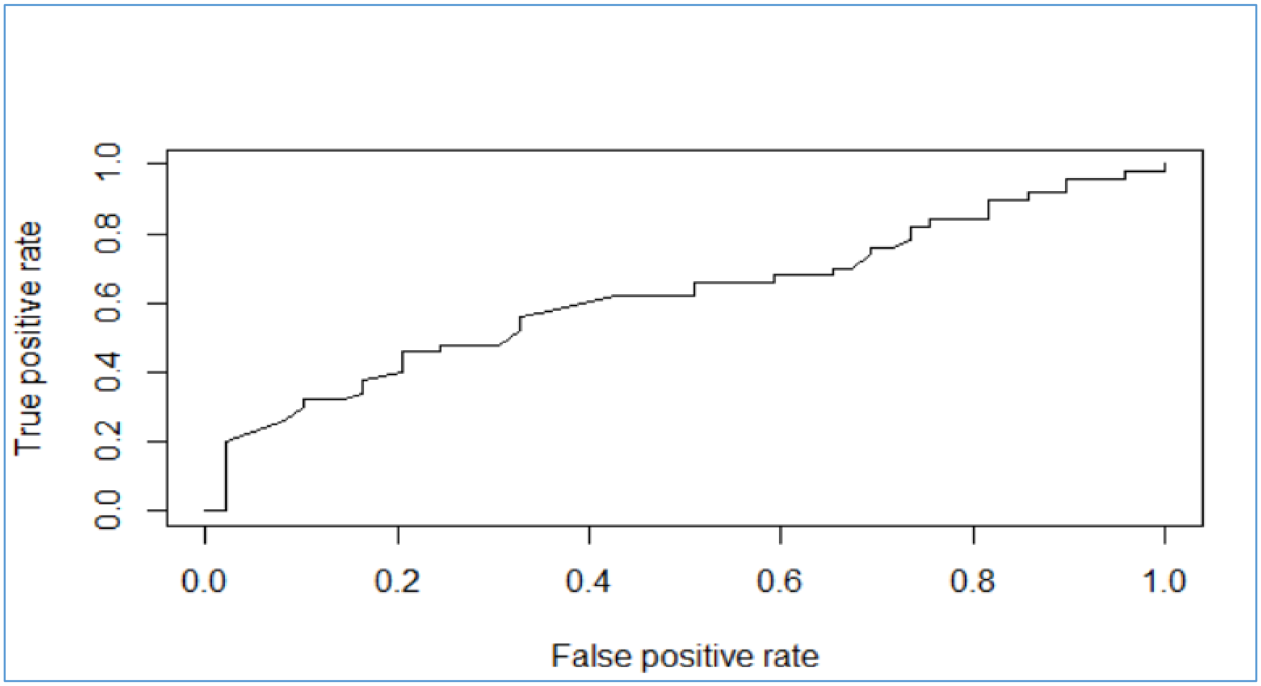


Logistic Regression with Ratio

```
> library(SDMTools)
> library(pROC)
> library(Hmisc)
> library(ROCR)
> setwd("C:/Users/Amit Kulkarni/Documents/R Programming/text analytics")
> LRdata= read.csv("DatasetR.csv", header=TRUE)
> #View(LRdata)
> deal.d = model.matrix(~deal -1, data=LRdata)
> LRdata1 = data.frame(LRdata, deal.d)
> #View(LRdata1)
> #str(LRdata1)
> set.seed(123)
> split = sample.split(LRdata1$dealTRUE, SplitRatio = 0.80)
> LRtrain= subset(LRdata1, split==T)
> LRtest= subset(LRdata1, split==F)
> View(LRtrain)
> LRtrain=subset(LRtrain, select=-c(1,2,3,4,5,6,7,11:19,21))
> LRtest=subset(LRtest, select=-c(1,2,3,4,5,6,7,11:19,21))
>
> View(LRtrain)
> View(LRtest)
> #All Variables
> LRModel = glm(dealTRUE ~ ., data=LRtrain, family= binomial)
> summary(LRModel)
```

```
Call:
glm(formula = dealTRUE ~ ., family = binomial, data = LRtrain)

Deviance Residuals:
   Min       1Q   Median       3Q      Max
-1.322  -1.212   1.021   1.138   1.760

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)       5.652e-01  2.565e-01   2.204   0.0275 *
askedFor          1.185e-07  3.590e-07   0.330   0.7414
exchangeForStake  1.062e-01  2.053e+00   0.052   0.9587
valuation        -7.789e-08  4.979e-08  -1.564   0.1177
Ratio            -1.287e+01  2.052e+02  -0.063   0.9500
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 548.88  on 395  degrees of freedom
Residual deviance: 542.10  on 391  degrees of freedom
AIC: 552.1

Number of Fisher Scoring iterations: 4
```

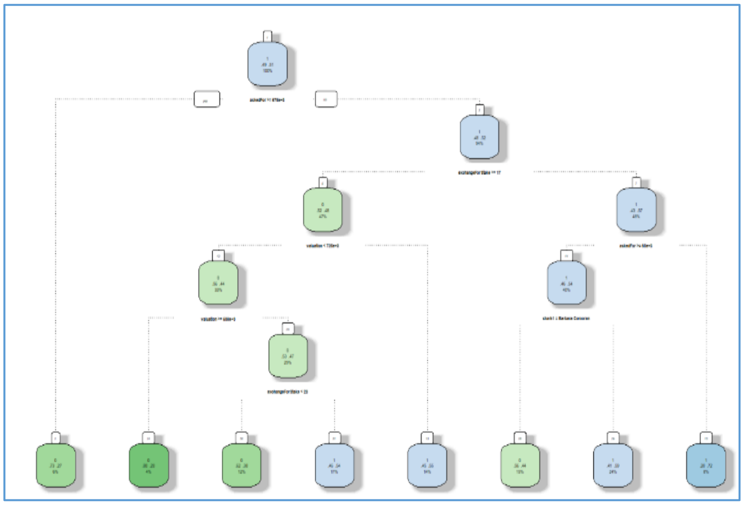
```
> predTest<-predict(LRModel, newdata=LRtest, type="response")
> table(LRtest$dealTRUE, predTest>0.5)

   FALSE TRUE
 0    16   33
 1    15   35

> (35+15)/nrow(na.omit(LRtest))

[1] 0.5050505
> ROCRpred=prediction(predTest, LRtest$dealTRUE)

> as.numeric(performance(ROCRpred,"auc")@y.values)

[1] 0.6214286

> perf= performance(ROCRpred, "tpr","fpr")

> plot (perf)
```



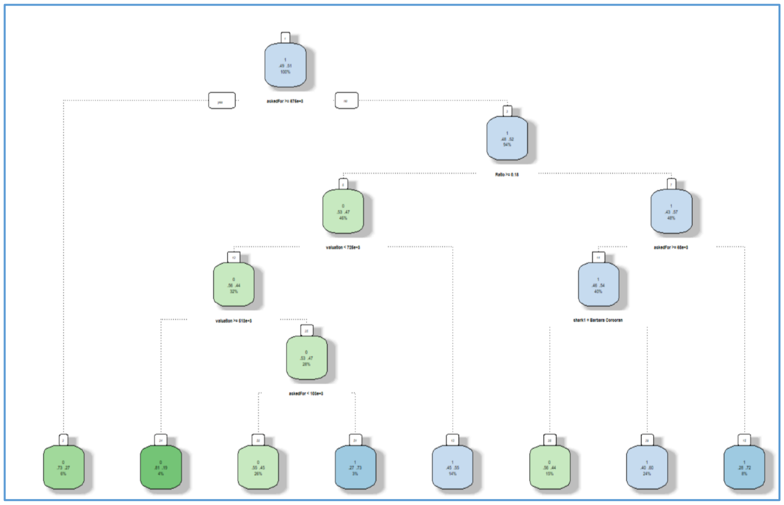
Questions

1. CART TREE Before and After

The coding for the same is provided above. The tree diagrams are depicted below



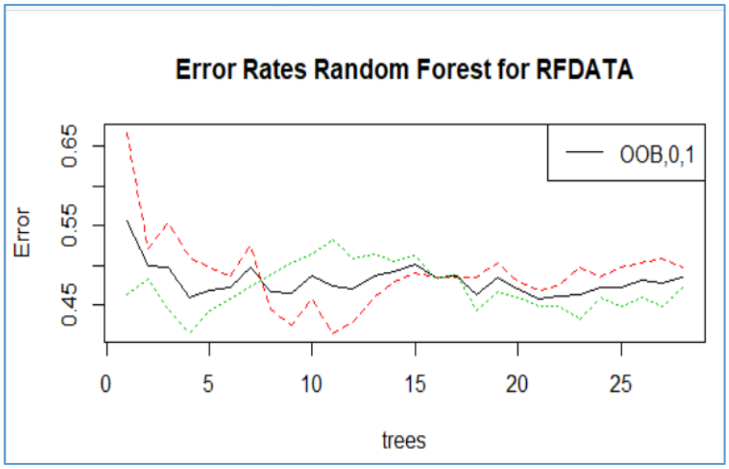
Before the ratio of asked for to valuation was introduced

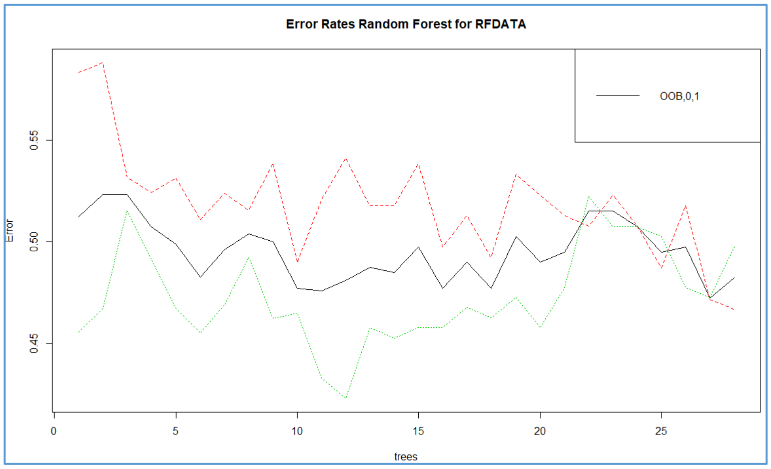


After the ratio of asked for to valuation was introduced

There has been no much change in the tree prepared after the ratio was introduced except that the tree at level 2 is broken at ratio level when it was introduced and at valuation level when the ratio was not introduced

2. *Random Forest Plot Before and After*





Random Forest model after introducing the ratio

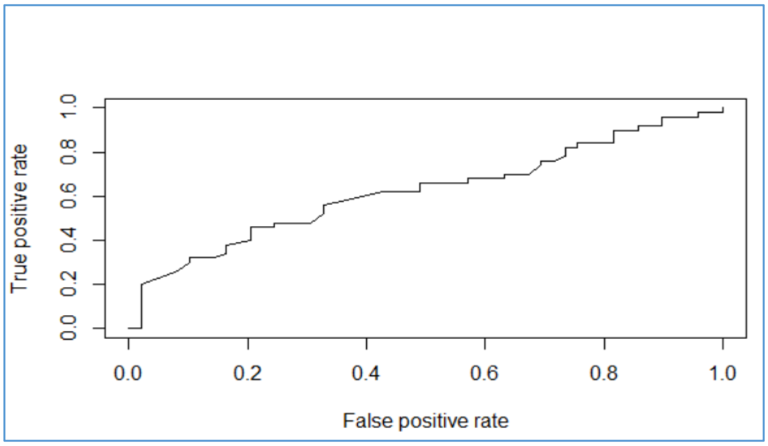
Random Forest model before introducing the ratio

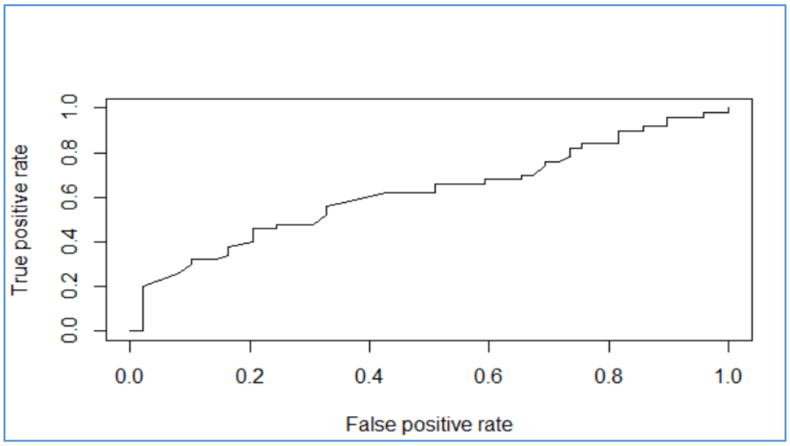
The confusion matrices arrived at for both with the ratio and without the ratio

Without the ratio		
Random Forest Confusion Matrix		
	0	1
0	86	85
1	83	93

With the ratio		
Random Forest Confusion Matrix		
	0	1
0	111	84
1	109	92

Confusion Matrices before and after





Performance plot after the ratio was introduced

Performance plot before the ratio was introduced

Confusion matrices depicting the before and after the introduction of the ratio. It can be seen that there is no difference in the matrices arrived at introducing the ratio variable

Without the ratio		
Logistic Regression Confusion Matrix		
	0	1
0	16	33
1	15	35

With the ratio		
Logistic Regression Confusion Matrix		
	0	1
0	16	33
1	15	35

Concluding remarks.

1. A full-fledged text analytics was not done but only looked at in the form of the wordcloud . Generic ideas of the business ideas in the form of their categories were identified.
2. 3 Supervised learning models were used to predict if the deal was struck or not basis the given data set
3. First CART was used to arrive the TREE diagram and see what were the significant variables at which they are broken down. It was seen that after introducing the ratio, it became an important variable in the analysis. However this did not change the tree much
4. Then Random Forest was used to check and the overall accuracy rates of the prediction was at close to 54% with the ratio and without the ratio. The mtry did have a different number of trees when the ratio was introduced but it did not change the accuracy much A 54% chance would not make the model strong.
5. Lastly logistic regression was used and the confusion matrices arrived. The true positives did not change and the confusion matrix were constant. The performance plot as well did not change. The overall accuracy of the model was close t0 51%. At this %age the model can be considered a weak model