Telecom Report

Sairam Sasupathi

1 October 2016

Objective

To increase the revenue of a Telecom company using the dataset provided by them.

Introduction

The data we have acquired is from the Telecom company who wants their revenue to be improved. Before we dive into making the model, we have to understand the data that we have.

```
summary(telecom)
```

```
##      region          tenure           age           marital
##  Min.   :1.000   Min.   : 1.00   Min.   :18.00   Min.   :0.000
##  1st Qu.:1.000   1st Qu.:17.00   1st Qu.:32.00   1st Qu.:0.000
##  Median :2.000   Median :34.00   Median :40.00   Median :0.000
##  Mean   :2.022   Mean   :35.53   Mean   :41.68   Mean   :0.495
##  3rd Qu.:3.000   3rd Qu.:54.00   3rd Qu.:51.00   3rd Qu.:1.000
##  Max.   :3.000   Max.   :72.00   Max.   :77.00   Max.   :1.000
##     address          income              ed            employ
##  Min.   : 0.00   Min.   :   9.00   Min.   :1.000   Min.   : 0.00
##  1st Qu.: 3.00   1st Qu.:  29.00   1st Qu.:2.000   1st Qu.: 3.00
##  Median : 9.00   Median :  47.00   Median :3.000   Median : 8.00
##  Mean   :11.55   Mean   :  77.53   Mean   :2.671   Mean   :10.99
##  3rd Qu.:18.00   3rd Qu.:  83.00   3rd Qu.:4.000   3rd Qu.:17.00
##  Max.   :55.00   Max.   :1668.00   Max.   :5.000   Max.   :47.00
##      retire          gender          reside         tollfree
##  Min.   :0.000   Min.   :0.000   Min.   :1.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:1.000   1st Qu.:0.000
##  Median :0.000   Median :1.000   Median :2.000   Median :0.000
##  Mean   :0.047   Mean   :0.517   Mean   :2.331   Mean   :0.474
##  3rd Qu.:0.000   3rd Qu.:1.000   3rd Qu.:3.000   3rd Qu.:1.000
##  Max.   :1.000   Max.   :1.000   Max.   :8.000   Max.   :1.000
##      equip          callcard        wireless        longmon
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   : 0.900
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.: 5.200
##  Median :0.000   Median :1.000   Median :0.000   Median : 8.525
##  Mean   :0.386   Mean   :0.678   Mean   :0.296   Mean   :11.723
##  3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:14.412
##  Max.   :1.000   Max.   :1.000   Max.   :1.000   Max.   :99.950
##     tollmon         equipmon        cardmon         wiremon
##  Min.   :  0.00   Min.   : 0.00   Min.   :  0.00   Min.   :  0.00
##  1st Qu.:  0.00   1st Qu.: 0.00   1st Qu.:  0.00   1st Qu.:  0.00
##  Median :  0.00   Median : 0.00   Median : 12.00   Median :  0.00
##  Mean   : 13.27   Mean   :14.22   Mean   : 13.78   Mean   : 11.58
```

```
##  3rd Qu.: 24.25   3rd Qu.:31.48   3rd Qu.: 20.50   3rd Qu.: 24.71
##  Max.   :173.00   Max.   :77.70   Max.   :109.25   Max.   :111.95
##     longten           tollten          equipten         cardten
##  Min.   :   0.90   Min.   :   0.0   Min.   :   0.0   Min.   :   0.0
##  1st Qu.:  90.14   1st Qu.:   0.0   1st Qu.:   0.0   1st Qu.:   0.0
##  Median : 285.48   Median :   0.0   Median :   0.0   Median : 332.5
##  Mean   : 574.05   Mean   : 551.3   Mean   : 465.6   Mean   : 605.8
##  3rd Qu.: 755.02   3rd Qu.: 846.9   3rd Qu.: 579.5   3rd Qu.: 910.0
##  Max.   :7257.60   Max.   :5916.0   Max.   :5028.6   Max.   :7515.0
##     wireten          multline         voice           pager
##  Min.   :   0.0   Min.   :0.000   Min.   :0.000   Min.   :0.000
##  1st Qu.:   0.0   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000
##  Median :   0.0   Median :0.000   Median :0.000   Median :0.000
##  Mean   : 442.7   Mean   :0.475   Mean   :0.304   Mean   :0.261
##  3rd Qu.: 316.5   3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:1.000
##  Max.   :7856.9   Max.   :1.000   Max.   :1.000   Max.   :1.000
##     internet         callid         callwait        forward
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000
##  Median :0.000   Median :0.000   Median :0.000   Median :0.000
##  Mean   :0.368   Mean   :0.481   Mean   :0.485   Mean   :0.493
##  3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:1.000
##  Max.   :1.000   Max.   :1.000   Max.   :1.000   Max.   :1.000
##      confer           ebill          custcat          churn
##  Min.   :0.000   Min.   :0.000   Min.   :1.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:1.000   1st Qu.:0.000
##  Median :1.000   Median :0.000   Median :3.000   Median :0.000
##  Mean   :0.502   Mean   :0.371   Mean   :2.487   Mean   :0.274
##  3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:3.000   3rd Qu.:1.000
##  Max.   :1.000   Max.   :1.000   Max.   :4.000   Max.   :1.000
```

```
str(telecom)
```

```
## 'data.frame':    1000 obs. of  36 variables:
##  $ region  : int  2 3 3 2 2 2 3 2 3 1 ...
##  $ tenure  : int  13 11 68 33 23 41 45 38 45 68 ...
##  $ age     : int  44 33 52 33 30 39 22 35 59 41 ...
##  $ marital : int  1 1 1 0 1 0 1 0 1 1 ...
##  $ address : int  9 7 24 12 9 17 2 5 7 21 ...
##  $ income  : int  64 136 116 33 30 78 19 76 166 72 ...
##  $ ed      : int  4 5 1 2 1 2 2 2 4 1 ...
##  $ employ  : int  5 5 29 0 2 16 4 10 31 22 ...
##  $ retire  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ gender  : int  0 0 1 1 0 1 1 0 0 0 ...
##  $ reside  : int  2 6 2 1 4 1 5 3 5 3 ...
##  $ tollfree: int  0 1 1 0 0 1 0 1 1 0 ...
##  $ equip   : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ callcard: int  1 1 1 0 0 1 1 1 1 1 ...
##  $ wireless: int  0 1 0 0 0 0 0 1 0 0 ...
##  $ longmon : num  3.7 4.4 18.15 9.45 6.3 ...
##  $ tollmon : num  0 20.8 18 0 0 ...
##  $ equipmon: num  0 0 0 0 0 0 0 50.1 0 0 ...
##  $ cardmon : num  7.5 15.2 30.2 0 0 ...
##  $ wiremon : num  0 35.7 0 0 0 0 0 64.9 0 0 ...
```

```
##  $ longten : num  37.5 42 1300.6 288.8 157.1 ...
##  $ tollten : num  0 211 1247 0 0 ...
##  $ equipten: num  0 0 0 0 0 ...
##  $ cardten : num  110 125 2150 0 0 ...
##  $ wireten : num  0 380 0 0 0 ...
##  $ multline: int  0 0 0 0 0 0 1 1 1 1 ...
##  $ voice   : int  0 1 0 0 0 0 0 1 0 0 ...
##  $ pager   : int  0 1 0 0 0 0 0 1 0 0 ...
##  $ internet: int  0 0 0 0 0 0 1 1 0 0 ...
##  $ callid  : int  0 1 1 0 1 1 0 1 1 0 ...
##  $ callwait: int  0 1 1 0 0 1 1 1 1 0 ...
##  $ forward : int  1 1 0 0 1 0 0 1 1 0 ...
##  $ confer  : int  0 1 1 0 1 0 0 1 1 0 ...
##  $ ebill   : int  0 0 0 0 0 0 1 1 0 0 ...
##  $ custcat : int  1 4 3 1 3 3 2 4 3 2 ...
##  $ churn   : int  1 1 0 1 0 0 1 0 0 0 ...
```

Data Cleaning

the data needs to be cleaned as there are many variable whose data type is not in the right format or it doesn't comply with the values that those variable hold. Variables that need to be converted from int to factor are region, marital, ed, retire, gender, reside, tollfree, equip, callcard, wireless, multiline, voice, pager, internet, callid, callwait, forward, confer, ebill, custcat, churn.

Variables that need to be converted from factor to numeric are income, longten, tollten, equipten, cardten, wireten.

```
# Integer to Factor conversions.
a <- match(c("region", "marital", "ed", "retire", "gender", "reside", "tollfree", "equip",
             "callcard", "wireless", "multline", "voice", "pager", "internet", "callid",
             "callwait", "forward", "confer", "ebill", "custcat", "churn"), names(telecom))
for(i in a)
  telecom[,i] = as.factor(telecom[,i])


# Factor to integer conversions
a <- match(c("income", "longten", "tollten", "equipten", "cardten", "wireten"), names(telecom))
```

The below conversion of data was introducing many NA's. The introduction of NAs are because of some of the values having commas. So the format of the data is changed in the excel so that no NAs are introduced by coercion. It was not shown for the sake of convenience.

```
# Factor to integer conversions (continuation)
for(i in a)
  telecom[,i] = as.numeric(as.character(telecom[,i]))
```

Problem Identification

The business problem has to be understood first before we can suggest what can be improved to increase the revenue. We see that we've different customer categories. Lets analyse and try to understand the different

customer categories and the revenue generated by them. Here we're considering the revenue generated by them over the whole tenure, since that will be the only way we can understand the importance of them.

```
# Mean revenue for different customer categories
avr_custcat = aggregate(cbind(longten, tollten, equipten, cardten, wireten) ~ custcat,
                data = telecom, mean)
avr_custcat # Mean revenue
```

```
##   custcat  longten   tollten  equipten  cardten    wireten
## 1       1 260.3818   54.9047  126.5235 274.9352   24.33158
## 2       2 745.6919  115.5636  621.8069 699.2396  122.52074
## 3       3 737.5358  961.6263  138.7349 741.6050  223.74093
## 4       4 575.1093 1022.7100 1093.4788 730.9958 1469.51907
```

```
# Median revenue for different customer categories
med_custcat = aggregate(cbind(longten, tollten, equipten, cardten, wireten) ~ custcat,
                        data = telecom, median)
med_custcat # Median revenue
```

```
##   custcat longten tollten equipten cardten  wireten
## 1       1  133.00    0.00    0.000     0.0    0.000
## 2       2  488.15    0.00  121.550   465.0    0.000
## 3       3  393.55  702.05    0.000   490.0    0.000
## 4       4  274.35  610.15  693.525   527.5 1080.475
```

```
# Total revenue for different customer categories
tot_custcat = aggregate(cbind(longten, tollten, equipten, cardten, wireten) ~ custcat,
                        data = telecom, sum)
tot_custcat # Total revenue
```

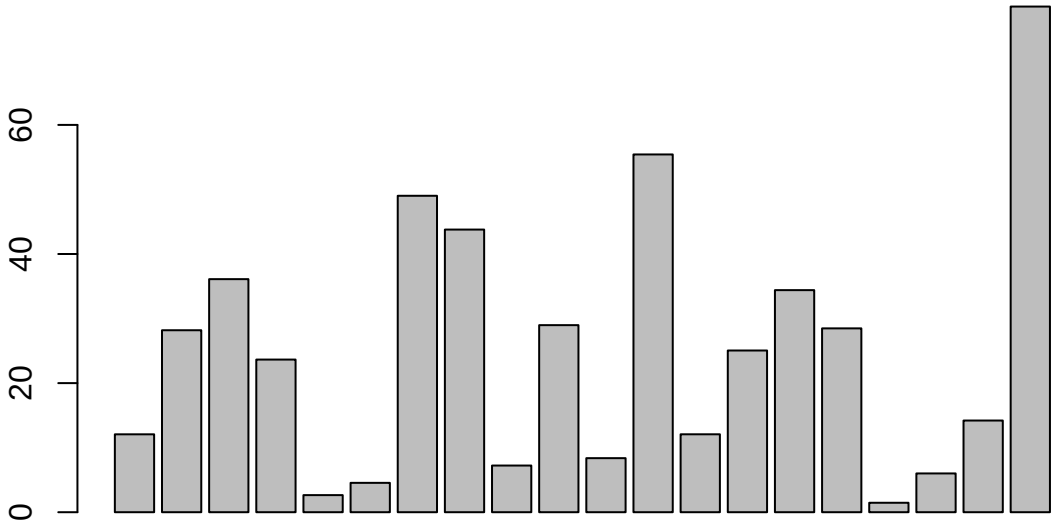
```
##   custcat   longten   tollten  equipten   cardten  wireten
## 1       1  69261.55  14604.65  33655.25  73132.75   6472.2
## 2       2 161815.15  25077.30 134932.10 151735.00  26587.0
## 3       3 207247.55 270217.00  38984.50 208391.00  62871.2
## 4       4 135725.80 241359.55 258061.00 172515.00 346806.5
```

```
# Percentage of contribution of different services for revenue for each category
tot_revenue = tot_custcat # Copy of total for each customer category
k = c()
k[1] = 10
for(i in 2:6) k[i] = sum(tot_custcat[,i])
percen_revenue = tot_revenue

tot_revenue = rbind(tot_custcat, k)
```

```
## Warning in `[<-.factor`(`*tmp*`, ri, value = 10): invalid factor level, NA
## generated
```

```
# Calculating the percentage of revenue
for(i in 2:6)
  for(j in 1:4)
```

```
    percen_revenue[j,i] = tot_revenue[j,i]*100/tot_revenue[5,i]

# Printing the data in barplot
barplot(height = c(percen_revenue$longten, percen_revenue$tollten, percen_revenue$equipten,
                   percen_revenue$cardten, percen_revenue$wireten))
```



Customer Category 4 is the highest revenue generator, so they've to be given the most importance. Also, the amount of revenue given from the Customer category 4 is completely spread out across all services. Customer Category 1 are the low revenue generators. Customer Category 2 and 3 create high revenue in some of the revenue streams but they don't use all the services provided by us. Hence, it's important for us to concentrate more on the category 2 and 3 as they're not using all the services as of now. If we could concentrate on moving those customers to the higher category, we can increase the revenue a lot.

Wireless revenue is contributed highest by Customer category 4. Having the highest revenue in only one of the category is not good.

Now, lets analyse the amount of customer churn in each category.

```
table(telecom$churn, telecom$custcat)
```

```
##
##       1   2   3   4
##   0 183 158 237 148
##   1  83  59  44  88
```

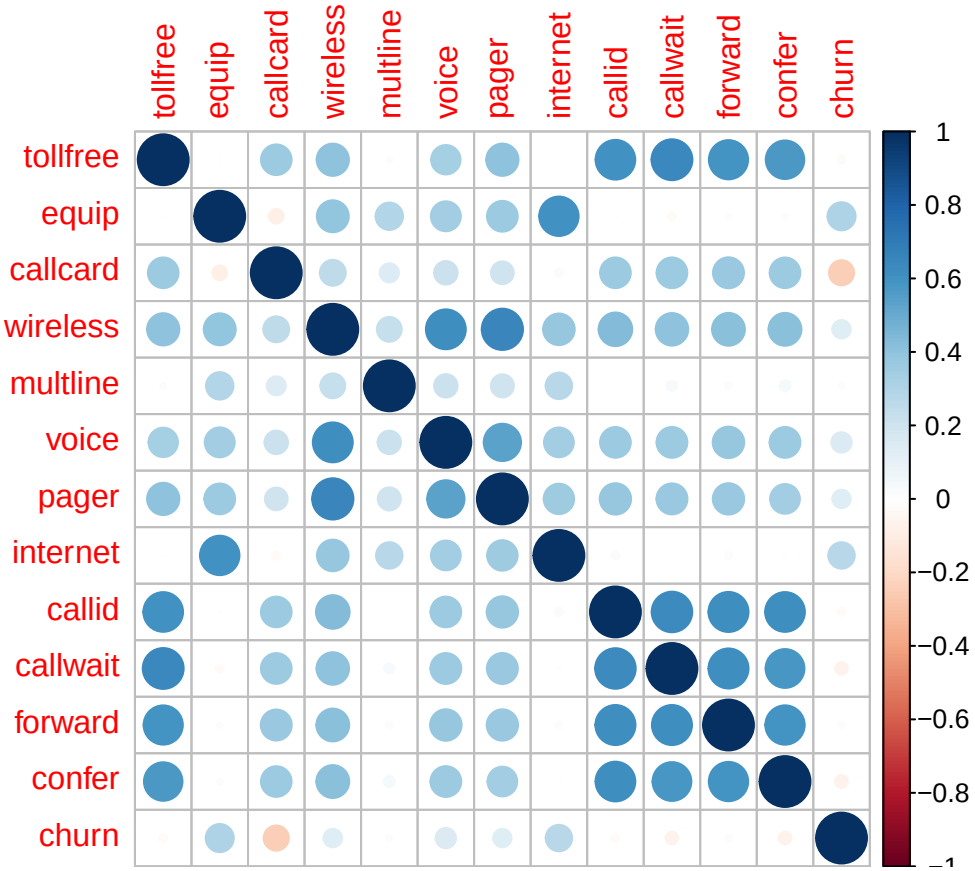
This shows that the top revenue generators are the ones who are leaving us. This is not healthy for the financial situation of the company. Hence, it is high time that we move the customers high up the category chain so that we can improve the situation. We can do two things right now. 1. Stopping the Churning customers from leaving us. 2. Move the customers up the category chain, i.e., introduce customers to use more of our other services.

Solution

Understanding Churn

Among the different Customer categories, we can find out what will be the reason for the churn. Hence, the correlation for the churn can be found.

```
corrplot::corrplot(cor(original[,c(12:15,26:33,36)]))
```



The Correlation matrix shows that we've very slight correlation for the churn with the equip and internet. Also, the internet and the equipment has good correlation between them. But correlation does not mean causation. We've to baseline our claim with some other parameter. Doing a association rule mining to base the relation between churn and these services can be one thing that can be done.

```
library(arules)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'arules'
```

```
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
# Apriori Algorithm for Churning customers according to usage of secondary services
service_matrix = as(object = as.matrix(original[,c(26:33, 36)]), Class = "itemMatrix")
sec_rules = apriori(service_matrix, parameter = list(support = 0.1, confidence = 0.4),
                    appearance = list(rhs = "churn", default = "lhs"))
```

```
## Apriori
```

```
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.4    0.1    1 none FALSE            TRUE       5     0.1      1
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 100
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[9 item(s), 1000 transaction(s)] done [0.00s].
## sorting and recoding items ... [9 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 done [0.00s].
## writing ... [1 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
sec_rules = sort(sec_rules, by = "lift")
# interestMeasure(x = sec_rules, transactions = service_matrix)
inspect(sec_rules)
```

```
##     lhs           rhs     support confidence lift
## [1] {internet} => {churn} 0.159   0.4320652  1.57688
```

```
# Apriori Algorithm for Churning customers according to usage of main services
rev_matrix = as(object = as.matrix(original[,c(12:15, 36)]), Class = "itemMatrix")
pri_rules = apriori(rev_matrix, parameter = list(support = 0.1, confidence = 0.4),
                appearance = list(rhs = "churn", default = "lhs"))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.4    0.1    1 none FALSE            TRUE       5     0.1      1
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 100
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[5 item(s), 1000 transaction(s)] done [0.00s].
## sorting and recoding items ... [5 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4
```

```
## Warning in apriori(rev_matrix, parameter = list(support = 0.1, confidence =
## 0.4), : Mining stopped (maxlen reached). Only patterns up to a length of 4
## returned!
```

```
##  done [0.00s].
## writing ... [1 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
pri_rules = sort(pri_rules, by = "lift")
# interestMeasure(x = pri_rules, transactions = rev_matrix)
inspect(pri_rules)
```

```
##     lhs        rhs     support confidence lift
## [1] {equip} => {churn} 0.172   0.4455959  1.626262
```

This shows that the equipment as a primary service and internet as a secondary service are leading to churn of many customers. Equipment as a service and internet as a secondary service has a lift of greater than 1.5, this shows that it did not happen by chance. If we could provide those customers some discounts on the internet service and equipment cost, they may decide to stay back. This may also stop other staying customers from churning in the future.

Cross-Selling

With the available data, we can do only the Cross-selling because to do up-selling, we need the rates of different categories. The best way to do Cross sell is to check the association mining for customer category 4 and apply the same in customer category 3.

```
# Subsets of each category of customers
category1 = subset(original, subset = custcat == 1)
category2 = subset(original, subset = custcat == 2)
category3 = subset(original, subset = custcat == 3)
category4 = subset(original, subset = custcat == 4)
```

```
# Apriori Algorithm for providing new services to the customers in Category 3
service = as(object = as.matrix(category4[,c(12:15)]), Class = "itemMatrix")
rules = apriori(service, parameter = list(support = 0.5, confidence = 0.9))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.9    0.1    1 none FALSE            TRUE       5     0.5      1
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 118
##
```

```
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[4 item(s), 236 transaction(s)] done [0.00s].
## sorting and recoding items ... [4 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [5 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rules = sort(rules, by = "lift")
# interestMeasure(x = rules, transactions = service)
inspect(rules)
```

```
##     lhs                    rhs        support   confidence lift
## [1] {tollfree,equip}    => {wireless} 0.5508475 0.9489051  1.087095
## [2] {equip,callcard}    => {wireless} 0.5805085 0.9383562  1.075010
## [3] {equip}             => {wireless} 0.6652542 0.9127907  1.045721
## [4] {tollfree,callcard} => {wireless} 0.6737288 0.9085714  1.040888
## [5] {tollfree,wireless} => {callcard} 0.6737288 0.9085714  1.035859
```

For the Customer Category 3, the above combinations will help them to move them to the next category 4.

```
# Apriori Algorithm for providing new services to the customers in Category 2
service1 = as(object = as.matrix(category3[,c(12:15)]), Class = "itemMatrix")
rules1 = apriori(service1, parameter = list(support = 0.5, confidence = 0.8))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5     0.5      1
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 140
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[4 item(s), 281 transaction(s)] done [0.00s].
## sorting and recoding items ... [2 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [3 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rules1 = sort(rules1, by = "lift")
# interestMeasure(x = rules1, transactions = service1)
inspect(rules1)
```

```
##     lhs           rhs        support   confidence lift
```

```
## [1] {callcard} => {tollfree} 0.6868327 0.8143460  1.059404
## [2] {tollfree} => {callcard} 0.6868327 0.8935185  1.059404
## [3] {}         => {callcard} 0.8434164 0.8434164  1.000000
```

For the Customer Category 2, the above combinations will help them to move them to the next category 3.

```
# Apriori Algorithm for providing new services to the customers in Category 1
service2 = as(object = as.matrix(category2[,c(12:15)]), Class = "itemMatrix")
rules2 = apriori(service2, parameter = list(support = 0.4, confidence = 0.6))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.6    0.1    1 none FALSE            TRUE       5     0.4      1
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 86
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[4 item(s), 217 transaction(s)] done [0.00s].
## sorting and recoding items ... [2 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [1 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rules2 = sort(rules2, by = "lift")
# interestMeasure(x = rules2, transactions = service2)
inspect(rules2)
```

```
##     lhs    rhs        support   confidence lift
## [1] {}  => {callcard} 0.6036866 0.6036866  1
```

For the Customer Category 1, the above combinations will help them to move them to the next category 2.

Churn Prediction

Data Setup

Lets predict the churn for the existing customers. For that, we'll divide the number of customers into churn and not churning customers. Then, out of the not churning customer, we'll take 225 customers for the final prediction which can be used to extrapolate to the 1000 customers. The remaining 501 customers who will not churn and the 274 customers who will churn will be divided into 2 groups of trainset and testset.

```
churn_customers = subset(telecom, subset = churn == 1)
staying_customers = subset(telecom, subset = churn == 0)
set.seed(16027)
clearsample = staying_customers[502:726,]
trainsample = sample(1:726, 501, replace = FALSE)
total = rbind(staying_customers[1:501,],churn_customers)
trainset = total[trainsample,]
testset = total[-trainsample,]
```

Feature Selection

Selecting the variables for the program.

```
library(rpart)
library(caret)
```

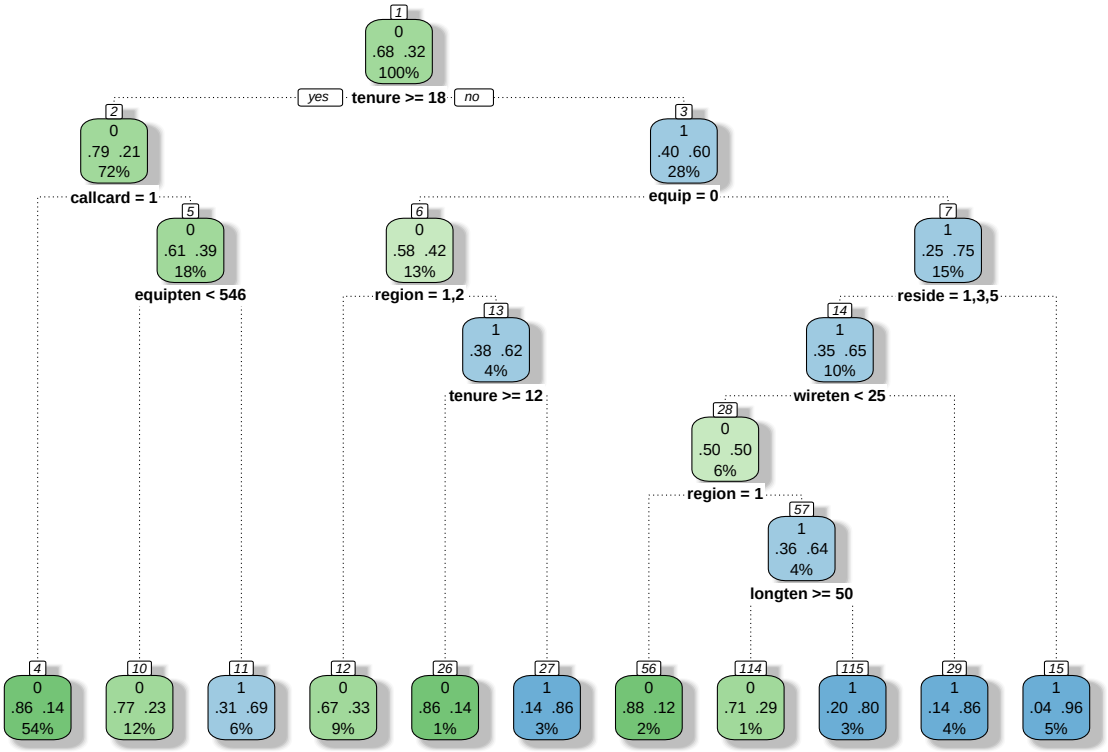
```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
library(rattle)
```

```
## Rattle: A free graphical interface for data mining with R.
## Version 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
cnt = rpart.control(xval = 10, cp = 0.0125, surrogatestyle = 1, usesurrogate = 2, maxsurrogate = 10)
model1=rpart(formula = churn~.,data = trainset, control = cnt)
pred_model1=predict(model1,newdata = testset,type="class")
rattle::fancyRpartPlot(model1)
```



Rattle 2016−Nov−18 22:16:09 saiviki

```
confusionMatrix(pred_model1,testset$churn)
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction   0   1
##          0 145  73
##          1  14  42
##
##                Accuracy : 0.6825
##                  95% CI : (0.6238, 0.7372)
##     No Information Rate : 0.5803
##     P-Value [Acc > NIR] : 0.0003215
##
##                   Kappa : 0.2983
##  Mcnemar's Test P-Value : 5.027e-10
##
##             Sensitivity : 0.9119
##             Specificity : 0.3652
##          Pos Pred Value : 0.6651
##          Neg Pred Value : 0.7500
##              Prevalence : 0.5803
##          Detection Rate : 0.5292
##    Detection Prevalence : 0.7956
##       Balanced Accuracy : 0.6386
##
##        'Positive' Class : 0
```

```
##
```

From the model generated above, the variables selected are the ones at the top of the tree. They are longten, equipten, cardten, tollten, address, age and income.

Model Generation using caret

```
cntrl=trainControl(method = "cv",number=10)
model2=train(churn ~ longten + equipten + cardten + tollten + address + age + income
             ,data=trainset,method="rpart",trControl=cntrl,metric="Kappa")
pred_model2=predict(model2,newdata=testset)
confusionMatrix(pred_model2,testset$churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 145  77
##          1  14  38
##
##                Accuracy : 0.6679
##                  95% CI : (0.6087, 0.7234)
##     No Information Rate : 0.5803
##     P-Value [Acc > NIR] : 0.001819
##
##                   Kappa : 0.2623
##  Mcnemar's Test P-Value : 8.066e-11
##
##             Sensitivity : 0.9119
##             Specificity : 0.3304
##          Pos Pred Value : 0.6532
##          Neg Pred Value : 0.7308
##              Prevalence : 0.5803
##          Detection Rate : 0.5292
##    Detection Prevalence : 0.8102
##       Balanced Accuracy : 0.6212
##
##        'Positive' Class : 0
##
```

The model generated has accuracy of 72% with a very good sensitivity to find the people who might churn in the future. Hence we can use this model to predict in the dataset we have taken.

```
pred = predict(model2, newdata = clearsample)
confusionMatrix(pred, clearsample$churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
```

```
##          0 212   0
##          1  13   0
##
##                Accuracy : 0.9422
##                  95% CI : (0.9032, 0.9689)
##     No Information Rate : 1
##     P-Value [Acc > NIR] : 1.0000000
##
##                   Kappa : 0
##  Mcnemar's Test P-Value : 0.0008741
##
##             Sensitivity : 0.9422
##             Specificity :     NA
##          Pos Pred Value :     NA
##          Neg Pred Value :     NA
##              Prevalence : 1.0000
##          Detection Rate : 0.9422
##    Detection Prevalence : 0.9422
##       Balanced Accuracy :     NA
##
##        'Positive' Class : 0
##
```

We find that out of the 225 customers that we have taken, 31 will be churning. Extrapolating this to the 1000 customers, we'll get 138 customers or 13.8% of your customer base. This will greatly help in reducing the revenue loss for the company.

Conclusion

Based on the understanding that we have created about the reason for the churning of the customers, the services of Internet and the Equipment has to be improved to retain the customers who are churning. The different cross selling ideas are generated for different customer categories using the higher customer category, so that we can move the customer to the higher customer category. After understanding the reason why the customers are churning, the model for churn of the existing customers is built. Based on the association algorithm used and the model built, we can largely assume that if we apply the strategy of moving the customers to higher category, we can, to an extent, reduce churn. Luring customers with right offers and products will help increase our revenue.