Predicting Customer Churn

Shivani Goyal

17 October 2018

Contents

Introduction

Problem:

The increased number of telecoms are a challenge to the telecom companies and many companies are facing huge revenue losses, to keep the customers many companies invest a huge revenue in the beginning and thus it becomes very important for the customers to expand the business and get back the amount that has been invested in the business.

The increase in the number of churn customers is become the present day challenge to the telecom industry and such customers create financial burden to the company, identifying such customers is the objective of this research paper. Research indicates that the cost of developing a new customer is approximately 5 higher than retaining the new customer.

Churn Prediction

In terms of telecommunication the customers leaving the current company and moving into another is called as churn, and in the present scenario considering the surge in the number of churn customers the industry is trying its best to retain the profitable customers and this is named as churn management.

Churn Management

Since acquiring new customers is challenging it is very important to retain the current customers. Churn can be reduced by analyzing the past history of the potential customers systematically. Large data is maintained about the customers and on performing a proper analysis on the same it is possible to predict the probable customers that might churn. The information that is available can be analyzed in different ways and thereby provide various ways for the operators to envisage the churning and evade the same.

Data collection

For analysis the data that is available in the telecom dataset has been used and prediction has been done for the same.

Data understanding

Before the data can be analyzed we have to clean the data and keep it ready so that the desired results can be derived from it. Data has to be clean so that the redundancy and errors can be removed because having such data will lead to incorrect results as well.

In this paper a Churn Analysis has been applied on Telecom data, here the agenda is to know the possible customers that might churn from the service provider. R programing and Python programming is used for the same .

We have Training and testing data given:
Training with 3333 observations of 21 variables – 20 Independent and 1 Dependent variable
Testing with 1667 observations of 21 variables – 20 Independent and one target variable

Since this is a classification problem , we will apply different classification algorithms on training data and then evaluate the model using test data and then we will select the model based on accuracy,false negative rate and other parameters.

We have different variables like state , area code , phone no, international plan , voice mail plan as categorical variables and number vmail messages , total day minutes , total day charge, total day calls , total night minutes , total night charge , total night calls , total eve calls, total eve minutes , total eve calls , total intl minutes , total intl charge , total int charge , number customer service calls and Churn as dependent variable

	state	account length	area code	phone number	international plan	voice mail plan	number vmail messages	total day minutes	total day calls	total day charge	...	total eve calls	total eve charge	total night minutes	total night calls	total night charge	total intl minutes	total intl calls
2	NJ	137	415	358-1921	no	no	0	243.4	114	41.38	...	110	10.3	162.6	104	7.32	12.2	

2.Data pre-processing

Before applying any model we will apply pre-processing steps to data –
Missing values , outlier , feature selection , feature scaling some steps are required prior to data modelling

Missing value analysis :

There are no missing values in the dataframe . Hence no need to remove any variable or impute any value .

	0
state	0
account length	0
area code	0
phone number	0
international plan	0
voice mail plan	0
number vmail messages	0
total day minutes	0
total day calls	0
total day charge	0
total eve minutes	0
total eve calls	0
total eve charge	0
total night minutes	0
total night calls	0

	0
total night charge	0
total intl minutes	0

Outlier analysis:

There are some outliers in the dataframe as we can visualize the same in box plot of variables. It is necessary to remove those values from the data so that model follows the correct data or pattern and does not give any extreme results

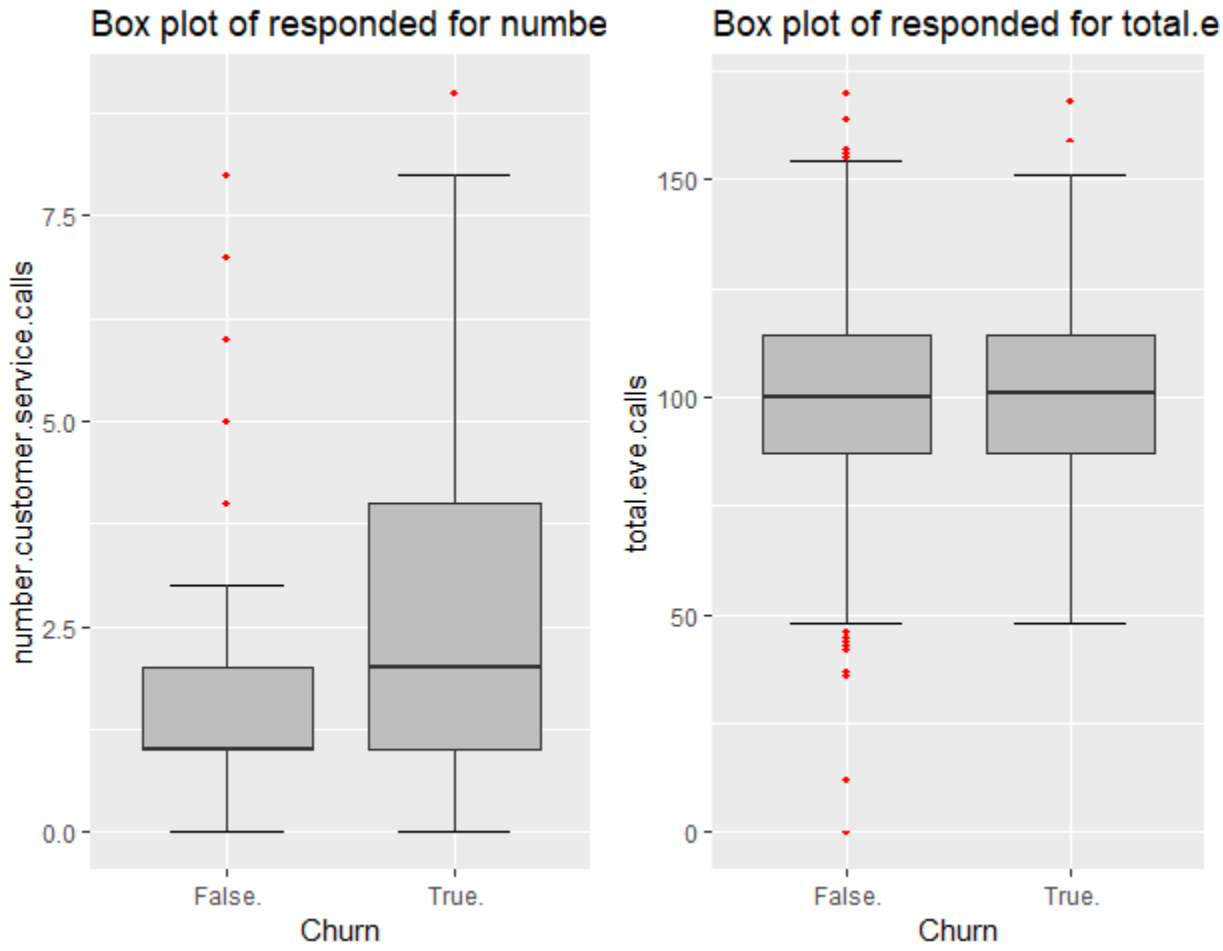

Fig 2.1

From the above figure Fig2.1 we can evidently visua

lize the points which are lying beyond the upper and lower bounds.Also from the box plot we can visualize that customers who churned out have made more no of customer calls than those who did not.

It is important to remove these variables as data pre-processing steps. Hence we will remove the data by box plot technique in R and calculating interquartile ranges for Python

Feature Selection :

For feature selection **we** will use correlation analysis among numerical independent variables and chi-square test among dependent and independent categorical variables

Correlation analysis :

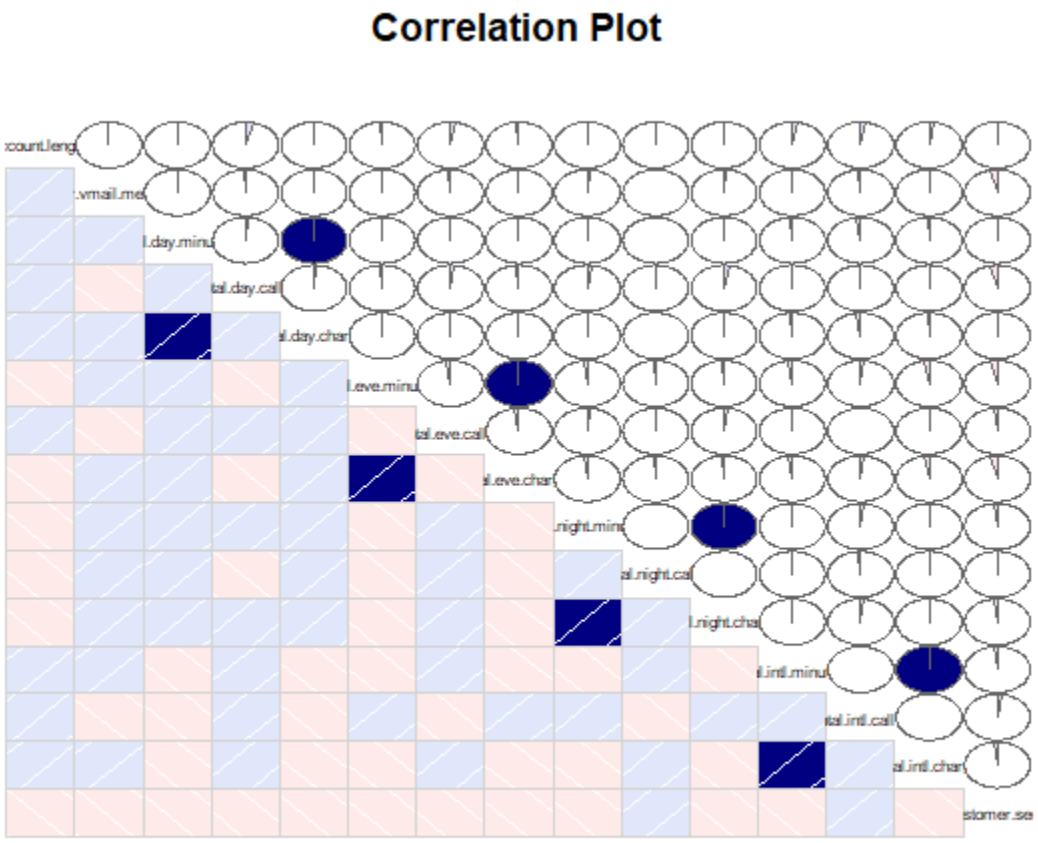

Fig2.2

As we can see from the plot that total day minutes and total day charge , total eve minutes and total eve charge , total night minutes and total night charge , total int minutes and total intl charge are completely co-related.Hence we can remove one of them from all

We will remove total day charge , total night charge , total eve charge , total intl charge from numeric variables

Chisquare test :

This test is to check dependency between categorical variables. We will check p value if it is less than 0.05 it means variables are dependent (We are rejecting the null hypothesis that variables are independent)

state	0.0207434729945
area code	0.697176291596
phone number	0.491109912426
international plan	1.68607692707e-53
voice mail plan	2.64389444987e-07

We will remove area code and phone no as their values are greater than of 0.05 . Also we will remove state variable as its value is not significant and it contains 51 levels which is not good for Model building

After removing removing outliers and redundant features we have 2797 observations of 14 variables

Feature Scaling :

Feature scaling is important in case we are using KNN algorithm as it uses distance method so it is important to scale the data so that some features dominance effect can be removed

In our data we have scaled the data using normalization technique.

3.Modelling

Since we have a classification problem here we will try different classification algorithms and choose the one which is giving maximum accuracy

1. Decision Trees

Decision trees is a good method to work on classification problem . We have worked here on C5.0 model which works on Information gain. It will select the node with high information gain and downwards make the tree with the same selection criteria.
We have different patterns here which form the rules

Some of the rules I am pasting here from R code:

Rule 7/15: (110.6/7, lift 1.4)
account.length > 44
international.plan = no
total.day.minutes <= 221.8
total.day.calls <= 77
total.eve.minutes > 168.7
total.eve.calls <= 125
total.intl.calls > 1
-> class 1 [0.929]

Rule 7/16: (28.1/1.2, lift 1.4)
international.plan = no
total.day.minutes > 221.8
total.eve.minutes <= 267
total.intl.calls <= 1
-> class 1 [0.927]

Rule 7/17: (186.7/14.8, lift 1.4)
international.plan = no
voice.mail.plan = yes
total.day.minutes <= 221.8
total.day.calls > 90
total.day.calls <= 131
total.eve.minutes > 168.7
total.night.calls <= 124
total.intl.minutes <= 13.2
-> class 1 [0.916]

Here class 1 is False in which customer will not churn
Class2 is True in which customer will churn out

Accuracy achieved through DT in R and Python are 93.34 and 90.4 respectively
FNR is 45.08 for R and 41.07 for Python.

2.Random Forest

Random forests, also known as random decision forests, are a popular ensemble method that can be used to build predictive models for both classification and regression problems. Ensemble methods use multiple learning models to gain better predictive results — in the case of a random forest, the model creates an entire forest of random uncorrelated decision trees to arrive at the best possible answer.

Random forests use CART algorithm which works on Gini index.

It uses out of bag sample .It selects a subset of feature for one tree , square root of m features where m is total features and make no of trees to reduce the error

Through random forests we have achieved accuracy of 93.34 in R and 93.10 in Python with False negative rate of 49.5 with trees 50.

```
Accuracy : 0.9334
95% CI : (0.9204, 0.9449)
No Information Rate : 0.9202
P-Value [Acc > NIR] : 0.02375

Kappa : 0.6545
Mcnemar's Test P-Value : < 2e-16

 Sensitivity : 0.9342
 Specificity : 0.9248
```

3.Logistic Regression

Logistic regression is one of the most commonly-used statistical techniques. It is used with data in which there is a binary (success-failure) outcome (response) variable, or where the outcome takes the form of a binomial proportion. Like linear regression, one estimates the relationship between predictor variables and an outcome variable. In logistic regression, however, one estimates the probability that the outcome variable assumes a certain value, rather than estimating the value itself

Coefficients:

```
                            Estimate Std. Error z value Pr(>|z|)
(Intercept)                -12.764901   1.020318 -12.511  < 2e-16 ***
account.length               0.001580   0.001829   0.864 0.387625
international.plan2          2.393734   0.181032  13.223  < 2e-16 ***
voice.mail.plan2            -2.780178   0.796376  -3.491 0.000481 ***
number.vmail.messages        0.051027   0.024693   2.066 0.038785 *
total.day.minutes            0.023450   0.001603  14.629  < 2e-16 ***
total.day.calls              0.002261   0.003670   0.616 0.537879
total.eve.minutes            0.013365   0.001567   8.528  < 2e-16 ***
total.eve.calls             -0.001058   0.003776  -0.280 0.779299
total.night.minutes          0.007045   0.001501   4.692 2.70e-06 ***
total.night.calls            0.003759   0.003757   1.001 0.317041
total.intl.minutes           0.129819   0.028411   4.569 4.89e-06 ***
total.intl.calls            -0.130279   0.035950  -3.624 0.000290 ***
number.customer.service.calls  0.020200   0.072372   0.279 0.780153
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1923.0  on 2796  degrees of freedom
Residual deviance: 1332.2  on 2783  degrees of freedom
AIC: 1360.2
```

The above is the summary of logistic model at the training level .We can judge the model at the training level in statistical models .We have different regression coefficients for all variables For international plan and voice mail plan we have coefficents assigned to different values of category

After predicting the test values from the model and comparing it with actual target values of test data

We got accuracy of model as 89.02% and False negative rate as 73.21. After this we will go for KNN and Naïve bayes classification algorithms

4. KNN

Knn is k nearest neighbours algorithm , it uses k nearest neighbours to the test data and assign the mode value of k-neighbours .It can be used for both regression and classification problems

It is a lazy learning technique because it does not save any patterns or rules , it just assess the whole data according to test data given and k value

I have implemented the KNN algorithms both in R and Python code attached and

got accuracy of 89.3 % and false negative rate of 72.3% with K =5

5. Naïve Bayes

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable. Bayes' theorem states the following relationship, given class variable y and dependent feature vector x1 through xn

In Python we use Gaussian class as it binomial classification problem from sklearn.naive_bayes import GaussianNB

In R and Python code , we are getting accuracy as 86.5 and False negative rate as 65.6 percentage

4. Conclusion

Model Evaluation

To evaluate the different classification algorithms we use confusion matrix .

Through confusion matrix we will try to evaluate the accuracy and false negative rate for different algorithms and choose the appropriate one

If we Python notebook

	Accuracy	False negative rate
Decision Trees	89.8	41.5
Random Forest	93.22	49.1
Logistic Regression	89.02	73.2
KNN	88.6	75.0
Naïve Bayes	86.5	65.6

These are the values from Python notebook

Model Selection

Based on Accuracy and False negative rate , we will choose Random Forest as through this we have achieved accuracy about 94 % although false negative rate is higher than of Decision trees but difference is not much. Hence we will choose Random Forest as our model and deploy it in the production system

5. Appendix

R code

```
# remove all the objects stored
rm(list=ls())

# Installing and loading few libraries
library(ggplot2)
library(grid)
library(gtable)
install.packages("gridExtra")
library(gridExtra)
install.packages("corrgram")
library(corrgram)

# Set working directory
setwd("C:/Users/jatin/Videos/Music/Documents/project1_edwisor")

getwd()


df= read.csv("Train_data.csv",header=T, na.strings=c(" ","NA"))
str(df)

# We have 3333 observations of 21 variables

# We have area.code as int data type , converting it into factor
df$area.code = as.factor(df$area.code)
str(df)
df
# just saving a copy of dataset before pre-processing for comparison afterwards
# processing of data
df_first = df
summary(df$Churn)
# We have 3333 observations of 21 variables and churn values
```

```
# 2850 False , 483 true

## Data pre-processing
# Preparing data for model building

# Checking missing values

missing_val = data.frame(apply(df,2,function(x){
  sum(is.na(x))
}))


# There are no missing values in the dataset ,

# Outlier analysis will be done on numeric data to see if dataset is containing some exceptional
data

numeric_index = sapply(df,is.numeric)

numeric_data = df[,numeric_index]
numeric_data

cnames = colnames(numeric_data)

cnames

for (i in 1:length(cnames))
{
    assign(paste0("gn",i), ggplot(aes_string(y = (cnames[i]), x = "Churn"), data = subset(df))+
           stat_boxplot(geom = "errorbar", width = 0.5) +
           geom_boxplot(outlier.colour="red", fill = "grey" ,outlier.shape=18,
                        outlier.size=1, notch=FALSE) +
           theme(legend.position="bottom")+
           labs(y=cnames[i],x="Churn")+
           ggtitle(paste("Box plot of responded for",cnames[i])))
}

gridExtra::grid.arrange(gn1,gn9,ncol=2)

gridExtra::grid.arrange(gn15,gn7,ncol=2)
```

```
#loop to remove outliers from all variables
 for(i in cnames){
   print(i)
   val = df[,i][df[,i] %in% boxplot.stats(df[,i])$out]
   print(length(val))
   df = df[which(!df[,i] %in% val),]
 }

unique(df$number.customer.service.calls)
# We have removed a lot of observations from this variable

# Now going for feature selection
# Correlation analysis among independent variables and chisquare test among
# categorical and independent variables

corrgram(df[,numeric_index], order = F,
         upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")

# total.day.charge , total.eve.charge, total.night.charge , total.intl.charge can be removed
# as they are redundant


## Chi-squared Test of Independence

factor_index = sapply(df,is.factor)
factor_data = df[,factor_index]

# we will apply chisquare on df_first as it contains 3333 observations
#choosing statistical significance as p = 0.05
for (i in 1:ncol(factor_data))
{
  print(names(factor_data)[i])
  print(chisq.test(table(factor_data$Churn,factor_data[,i])))
}


# p>0.05 accept the null hypothesis variables are independent
# area.code, phone no  can be removed
```

```
unique(df$state)

# We will remove State variable also as it not very less than 0.05 and has too many levels
# which is not helpful in model building
# Removing 7 variables
df = subset(df, select = -
c(state,area.code,phone.number,total.day.charge,total.eve.charge,total.night.charge,total.intl.char
ge))
df

colnames(df)

#Normalisation
# for future use
# df_new frame
cnames =
c("account.length","number.vmail.messages","total.day.minutes","total.day.calls","total.eve.min
utes","total.eve.calls","total.night.calls",
          "total.night.minutes","total.intl.minutes","total.intl.calls"
,"number.customer.service.calls")

df_new = df
for(i in cnames){
  print(i)
  df_new[,i] = (df[,i] - min(df[,i]))/
    (max(df[,i]) - min(df[,i]))
}
##Data Manupulation; convert string categories into factor numeric

for(i in 1:ncol(df))
{
  if(class(df[,i]) == 'factor')
  {
    df[,i] = factor(df[,i], labels=(1:length(levels(factor(df[,i])))))
  }
}
df

# cHURN 1 = false and 2 = true in dataframe
```

```
# Implementing decision trees algorithm for classification pblm

install.packages("C50")
library(C50)

c50_model = C5.0 (Churn~.,df, trials = 10, rules = TRUE)
summary(c50_model)

# Reading the test data file and some data manipulation
test = read.csv("Test_data.csv")
# removing redudant variables from test data
test = subset(test, select = -
c(state,area.code,phone.number,total.day.charge,total.eve.charge,total.night.charge,total.intl.char
ge))

for(i in 1:ncol(test)){

  if(class(test[,i]) == 'factor'){

    test[,i] = factor(test[,i], labels=(1:length(levels(factor(test[,i])))))

  }
}


str(test)

C50_Predictions = predict(c50_model, test[,-14], type = "class")
C50_Predictions

install.packages("caret",dependencies= TRUE )
library(caret)

ConfMatrix_C50 = table(test$Churn, C50_Predictions)
confusionMatrix(ConfMatrix_C50)



# Accuracy = 93.34
# FNR = FN/FN +TP = 101/101+123
```

#45.08

```
# Random Forest Model
install.packages("randomForest")
library(randomForest)
RF_model = randomForest(Churn ~ ., df, importance = TRUE, ntree = 50)

RF_Predictions = predict(RF_model, test[,-14])

ConfMatrix_RF = table(test$Churn, RF_Predictions)
confusionMatrix(ConfMatrix_RF)

# Accuracy = 93.34
# FNR = 109/(109 +115)
# 48.66

# Logistic regression

#Logistic Regression
logit_model = glm(Churn ~ ., data = df, family = "binomial")

#summary of the model
summary(logit_model)

test

#predict using logistic regression
logit_Predictions = predict(logit_model, newdata = test, type = "response")
logit_Predictions
#convert prob
logit_Predictions = ifelse(logit_Predictions > 0.5, 2, 1)

logit_Predictions
ConfMatrix_logit = table(test$Churn, logit_Predictions)
test$Churn

confusionMatrix(ConfMatrix_logit)

# Accuracy = 89.02
```

```
library(class)
# We will use the scaled data for KNN
#preparing the scaled test data
cnames

test_new = test
for(i in cnames){
  print(i)
  test_new[,i] = (test_new[,i] - min(test_new[,i]))/
    (max(test_new[,i]) - min(test_new[,i]))
}

test_new
KNN_Predictions = knn(df[, 1:13], test[, 1:13], df[,14], k = 5)

Conf_matrix = table(KNN_Predictions, test$Churn)

sum(diag(Conf_matrix))/nrow(test)
##Evaluate the performance of classification model
ConfMatrix_RF = table(test$Churn, KNN_Predictions)

confusionMatrix(ConfMatrix_RF)
# 156/156+68
# FNR = 72.3
#Accuraacy=89.3



#naive Bayes
install.packages("e1071")
library(e1071)

#Develop model
NB_model = naiveBayes(Churn~ ., data = df)

#predict on test cases #raw
NB_Predictions = predict(NB_model, test[,1:13], type =
```

References

https://www.statistics.com/logistic-regression/

https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

plaza.ufl.edu/sakib/docs/churn.pdf