A General Overview of Predictive Modeling with Logistic Regression

Harisingh Khedar

December 16, 2016

Hello! Welcome back. In the previous document, we discussed about modeling process in general. We learnt about typical set of steps taken before a model is deployed in the production environment. In this exercise, we will continue our discussion in similar fashion but with another technique - Logistic Regression.

**What is logistic regression?**

Logistic regression is used when you want to estimate class probabilities i.e the probability that an object is in a given class. Logistic regression has very useful application like detecting whether a transaction is fraudulent or legitimate, assigning appropriate categories to products etc.

**What do we achieve with logistic regression?**

Logistic regression predicts the probability y that an instance belongs to a specific category - for instance, the probability that a flight will be delayed. When x[i,] is a row of inputs (for example, a flight's origin and destination, the time of year, the weather, the air carrier), logistic regression finds a fit function f(x) such that

P[y[i] in class] ~ f(x[i,]) = s(a+b[1] x[i,1] + ... b[n] x[i,n])

Here, s(z) is the so-called sigmoid function, defined as s(z) = 1/(1+exp(z)). If the y[i] are the probabilities that the x[i,] belong to the class of interest (in our example, that a flight with certain characteristics will be delayed), then the task of fitting is to find the b[1], ..., b[n] such that f(x[i,]) is the best possible estimate of y[i]. R supplies a one-line command to find these coefficients: glm(). Note that we don't need to supply y[i] that are probability estimates to run glm(); the training method only requires y[i] that say whether a given training example is in the target class. The sigmoid function maps real numbers to the interval (0,1)-that is, to probabilities. The inverse of the sigmoid is the logit, which is defined as log(p/(1-p)), where p is a probability. The ratio p/(1-p) is known as the odds, so in the flight example, the logit is the log of the odds (or log-odds) that a flight will be delayed. In other words, you can think of logistic regression as a linear regression that finds the log-odds of the probability that you're interested in. In particular, logistic regression assumes that logit(y) is linear in the values of x. Like linear regression, logistic regression will find the best coefficients to predict y, including finding advantageous combinations and cancellations when the inputs are correlated.

**Dataset Information**

The data is related to direct marketing campaigns of a Portuguese Banking Institution. The classification goal is to predict if the client will subscribe (Yes/No) a term deposit (Bank product and outcome variable). For more information on this dataset attributes, please access [<http://archive.ics.uci.edu/ml/datasets/Bank+Marketing>]

Just to remind, we organize our work around below set of steps -

* Data collection
* Data exploration and preparation
* Build model
* Evaluate model
* Validate model
* Deploy model

## 1. Data Collection

*Load the dataset in R*

*Note: The input file needs to exist locally to use below command. Please download the same.*

BankData <- read.table(file.choose(), header = T, sep = ";")  
dim(BankData)

# [1] 41188 21

*Getting to know the dataset*

str(BankData)

# 'data.frame': 41188 obs. of 21 variables:  
# $ age : int 56 57 37 40 56 45 59 41 24 25 ...  
# $ job : Factor w/ 12 levels "admin.","blue-collar",..: 4 8 8 1 8 8 1 2 10 8 ...  
# $ marital : Factor w/ 4 levels "divorced","married",..: 2 2 2 2 2 2 2 2 3 3 ...  
# $ education : Factor w/ 8 levels "basic.4y","basic.6y",..: 1 4 4 2 4 3 6 8 6 4 ...  
# $ default : Factor w/ 3 levels "no","unknown",..: 1 2 1 1 1 2 1 2 1 1 ...  
# $ housing : Factor w/ 3 levels "no","unknown",..: 1 1 3 1 1 1 1 1 3 3 ...  
# $ loan : Factor w/ 3 levels "no","unknown",..: 1 1 1 1 3 1 1 1 1 1 ...  
# $ contact : Factor w/ 2 levels "cellular","telephone": 2 2 2 2 2 2 2 2 2 2 ...  
# $ month : Factor w/ 10 levels "apr","aug","dec",..: 7 7 7 7 7 7 7 7 7 7 ...  
# $ day\_of\_week : Factor w/ 5 levels "fri","mon","thu",..: 2 2 2 2 2 2 2 2 2 2 ...  
# $ duration : int 261 149 226 151 307 198 139 217 380 50 ...  
# $ campaign : int 1 1 1 1 1 1 1 1 1 1 ...  
# $ pdays : int 999 999 999 999 999 999 999 999 999 999 ...  
# $ previous : int 0 0 0 0 0 0 0 0 0 0 ...  
# $ poutcome : Factor w/ 3 levels "failure","nonexistent",..: 2 2 2 2 2 2 2 2 2 2 ...  
# $ emp.var.rate : num 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 ...  
# $ cons.price.idx: num 94 94 94 94 94 ...  
# $ cons.conf.idx : num -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 ...  
# $ euribor3m : num 4.86 4.86 4.86 4.86 4.86 ...  
# $ nr.employed : num 5191 5191 5191 5191 5191 ...  
# $ y : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...

head(BankData)

# age job marital education default housing loan contact month  
# 1 56 housemaid married basic.4y no no no telephone may  
# 2 57 services married high.school unknown no no telephone may  
# 3 37 services married high.school no yes no telephone may  
# 4 40 admin. married basic.6y no no no telephone may  
# 5 56 services married high.school no no yes telephone may  
# 6 45 services married basic.9y unknown no no telephone may  
# day\_of\_week duration campaign pdays previous poutcome emp.var.rate  
# 1 mon 261 1 999 0 nonexistent 1.1  
# 2 mon 149 1 999 0 nonexistent 1.1  
# 3 mon 226 1 999 0 nonexistent 1.1  
# 4 mon 151 1 999 0 nonexistent 1.1  
# 5 mon 307 1 999 0 nonexistent 1.1  
# 6 mon 198 1 999 0 nonexistent 1.1  
# cons.price.idx cons.conf.idx euribor3m nr.employed y  
# 1 93.994 -36.4 4.857 5191 no  
# 2 93.994 -36.4 4.857 5191 no  
# 3 93.994 -36.4 4.857 5191 no  
# 4 93.994 -36.4 4.857 5191 no  
# 5 93.994 -36.4 4.857 5191 no  
# 6 93.994 -36.4 4.857 5191 no

The dataset has 41188 instances and 21 variables, consisting good mix of numerical and categorical variables in the dataset. If you check the attribute info on aforementioned link, the variables cover client data, previous campaign data, social and economic contextual data among others.

## 2. Data Exploration and Preparation

### *Fetching the summary statistics*

summary(BankData)

# age job marital   
# Min. :17.00 admin. :10422 divorced: 4612   
# 1st Qu.:32.00 blue-collar: 9254 married :24928   
# Median :38.00 technician : 6743 single :11568   
# Mean :40.02 services : 3969 unknown : 80   
# 3rd Qu.:47.00 management : 2924   
# Max. :98.00 retired : 1720   
# (Other) : 6156   
# education default housing   
# university.degree :12168 no :32588 no :18622   
# high.school : 9515 unknown: 8597 unknown: 990   
# basic.9y : 6045 yes : 3 yes :21576   
# professional.course: 5243   
# basic.4y : 4176   
# basic.6y : 2292   
# (Other) : 1749   
# loan contact month day\_of\_week  
# no :33950 cellular :26144 may :13769 fri:7827   
# unknown: 990 telephone:15044 jul : 7174 mon:8514   
# yes : 6248 aug : 6178 thu:8623   
# jun : 5318 tue:8090   
# nov : 4101 wed:8134   
# apr : 2632   
# (Other): 2016   
# duration campaign pdays previous   
# Min. : 0.0 Min. : 1.000 Min. : 0.0 Min. :0.000   
# 1st Qu.: 102.0 1st Qu.: 1.000 1st Qu.:999.0 1st Qu.:0.000   
# Median : 180.0 Median : 2.000 Median :999.0 Median :0.000   
# Mean : 258.3 Mean : 2.568 Mean :962.5 Mean :0.173   
# 3rd Qu.: 319.0 3rd Qu.: 3.000 3rd Qu.:999.0 3rd Qu.:0.000   
# Max. :4918.0 Max. :56.000 Max. :999.0 Max. :7.000   
#   
# poutcome emp.var.rate cons.price.idx cons.conf.idx   
# failure : 4252 Min. :-3.40000 Min. :92.20 Min. :-50.8   
# nonexistent:35563 1st Qu.:-1.80000 1st Qu.:93.08 1st Qu.:-42.7   
# success : 1373 Median : 1.10000 Median :93.75 Median :-41.8   
# Mean : 0.08189 Mean :93.58 Mean :-40.5   
# 3rd Qu.: 1.40000 3rd Qu.:93.99 3rd Qu.:-36.4   
# Max. : 1.40000 Max. :94.77 Max. :-26.9   
#   
# euribor3m nr.employed y   
# Min. :0.634 Min. :4964 no :36548   
# 1st Qu.:1.344 1st Qu.:5099 yes: 4640   
# Median :4.857 Median :5191   
# Mean :3.621 Mean :5167   
# 3rd Qu.:4.961 3rd Qu.:5228   
# Max. :5.045 Max. :5228   
#

If you notice, we actually don't have all the information about all clients. For example, we don't have credit status info, housing loan info, personal loan info etc. about many clients. All these values are indicated by 'Unknown' level for categorical variables. *In normal scenario, we should go back and try to find out why we don't have missing information for so many clients.* Was it because the interviewer failed to collect such information? Was it because data was lost for some reason? It is worth investigating!

For this exercise, we will consider these records as missing data and treat accordingly. Lets reload same file again, but considering 'Unknown' as NA's.

BankData <- read.table(file.choose(), header = T, sep = ";", na.strings = "unknown")  
summary(BankData$housing)

# no yes NA's   
# 18622 21576 990

***Removing*** *missing data*

dim(BankData)

# [1] 41188 21

BankData.v1 <- na.omit(BankData)  
dim(BankData.v1)

# [1] 30488 21

summary(BankData.v1$housing)

# no yes   
# 13967 16521

*Lets* ***rename*** *variables for readability*

colnames(BankData.v1) <- c("Age", "Job", "Marital\_status", "Education\_level",   
 "Default\_credit", "Housing\_loan", "Personal\_loan", "Contact\_type", "Contact\_month",   
 "Contact\_weekday", "Duration", "Campaign", "Contact\_gapdays", "Previous\_contacts",   
 "Previous\_outcome", "Employment\_var\_rate", "Consumer\_price\_idx", "Consumer\_confi\_idx",   
 "Euribor\_3M\_rate", "Num\_of\_employees", "Term\_deposit")  
colnames(BankData.v1)

# [1] "Age" "Job" "Marital\_status"   
# [4] "Education\_level" "Default\_credit" "Housing\_loan"   
# [7] "Personal\_loan" "Contact\_type" "Contact\_month"   
# [10] "Contact\_weekday" "Duration" "Campaign"   
# [13] "Contact\_gapdays" "Previous\_contacts" "Previous\_outcome"   
# [16] "Employment\_var\_rate" "Consumer\_price\_idx" "Consumer\_confi\_idx"   
# [19] "Euribor\_3M\_rate" "Num\_of\_employees" "Term\_deposit"

*Lets* ***reencode*** *the outcome as logical - TRUE/FALSE*

BankData.v1$Term\_deposit <- ifelse(BankData.v1$Term\_deposit == "yes", TRUE,   
 FALSE)

*Lets also* ***reset*** *levels for categorical variables and set* ***reference levels****. For other categorical variables, we will just keep the default reference level.*

# Job  
Joblist <- list(unemployed = c("unemployed", "student", "retired"), self\_employed = c("self-employed",   
 "entrepreneur"), employed = c("admin.", "blue-collar", "housemaid", "management",   
 "services", "technician"))  
for (i in 1:length(Joblist)) levels(BankData.v1$Job)[levels(BankData.v1$Job) %in%   
 Joblist[[i]]] <- names(Joblist)[i]  
BankData.v1$Job <- relevel(BankData.v1$Job, "unemployed")  
  
# Marital  
BankData.v1$Marital\_status <- relevel(BankData.v1$Marital\_status, "single")  
  
# Education level  
levels(BankData.v1$Education\_level)[levels(BankData.v1$Education\_level) %in%   
 c("basic.4y", "basic.6y", "basic.9y")] <- "basic"  
BankData.v1$Education\_level <- relevel(BankData.v1$Education\_level, "illiterate")

It would make more sense to **categorize few numeric variables** -

Campaign\_breaks <- c(0, 2, 50)  
BankData.v1$Campaign1 <- cut(BankData.v1$Campaign, breaks = Campaign\_breaks,   
 labels = c("<=2", ">2"), include.lowest = T)  
summary(BankData.v1$Campaign1)

# <=2 >2   
# 21119 9369

Gap\_breaks <- c(0, 3, 30)  
BankData.v1$Contact\_gapdays0 <- ifelse(BankData.v1$Contact\_gapdays == 999, 0,   
 BankData.v1$Contact\_gapdays)  
BankData.v1$Contact\_gapdays1 <- cut(BankData.v1$Contact\_gapdays0, breaks = Gap\_breaks,   
 labels = c("<=3", ">3"), include.lowest = T)  
summary(BankData.v1$Contact\_gapdays1)

# <=3 >3   
# 29647 841

# Rounding number of employees  
BankData.v1$Num\_of\_employees <- round(BankData.v1$Num\_of\_employees)

**NOTE: We create new variables, in case if it is required to use original variables for analysis.**

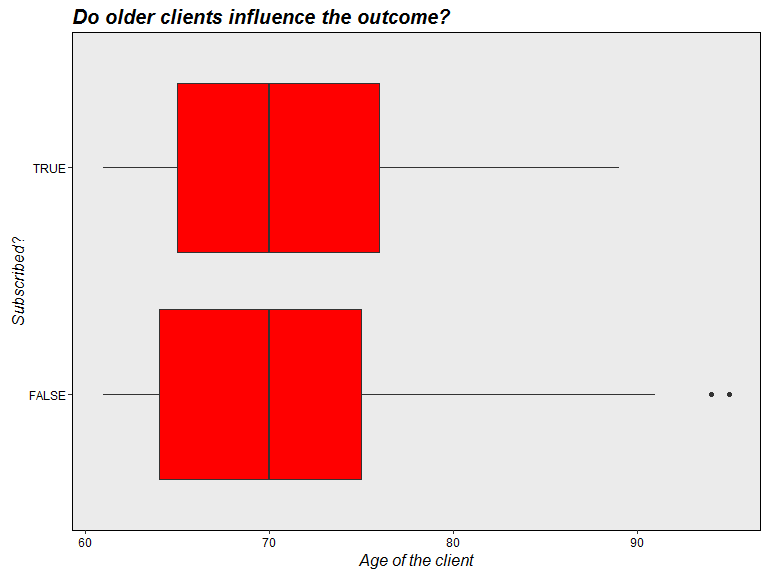
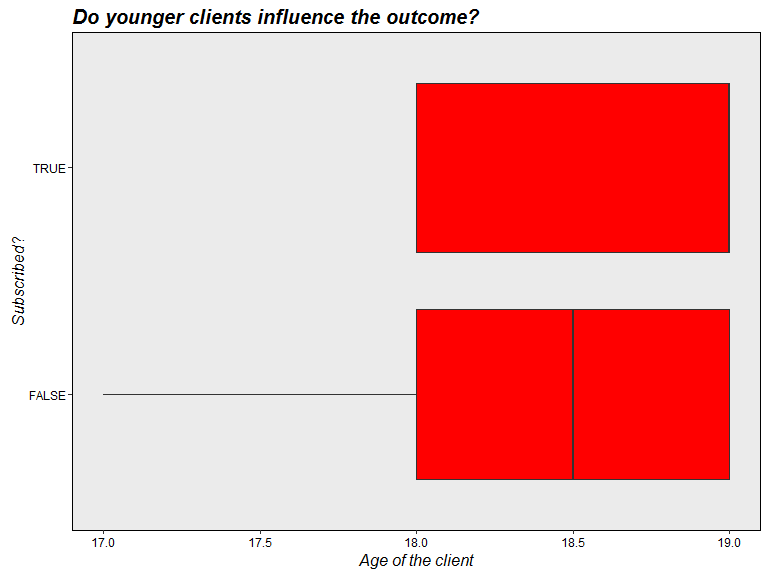
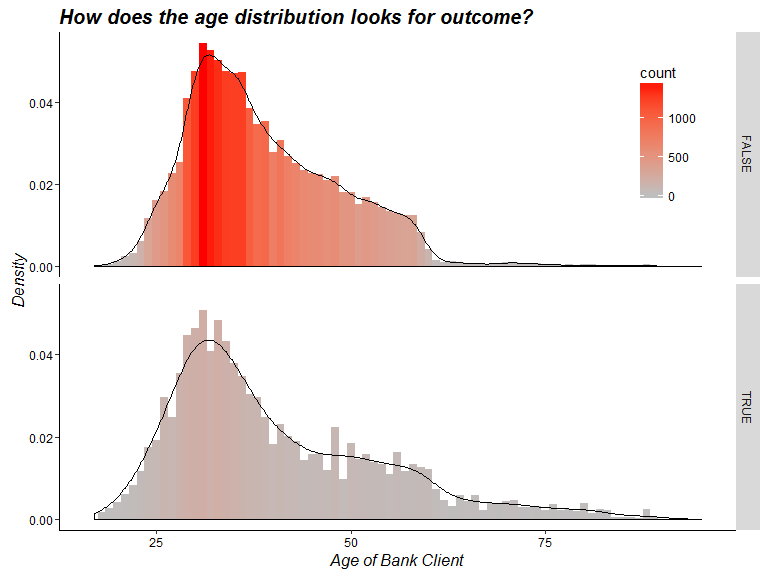
But when should one convert numerical variables into categorical? Please check [<http://www.theanalysisfactor.com/3-situations-when-it-makes-sense-to-categorize-a-continuous-predictor-in-a-regression-model/>]. I chose it because I didn't see good spread of values across the range. The values seem concentrated in certain areas. Converting numerical variables into categorical is subjective and your call !

What else we learn from summary statistics?

A **typical client** is mid-aged, employed, married, with no default credit among others. The last campaign was not that successful in terms of client subscriptions. The Bank haven't pursued same client enough between previous and current campaign.

### *Visualization*

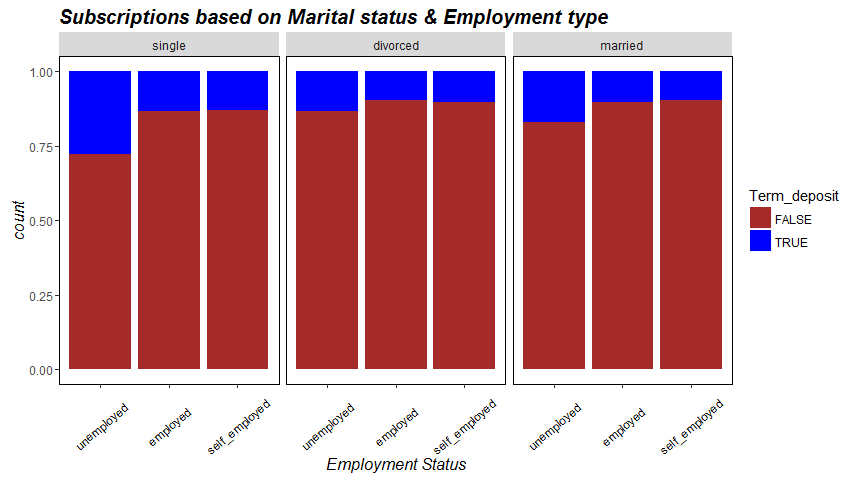
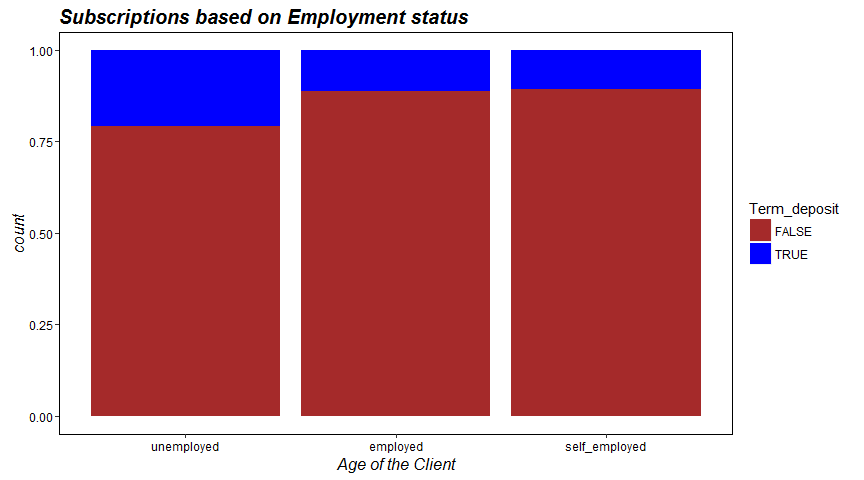
The below graph indicates that most of the people didn't opt for term deposit during last campaign. Also, mid-aged people seem to influence the outcome heavily.



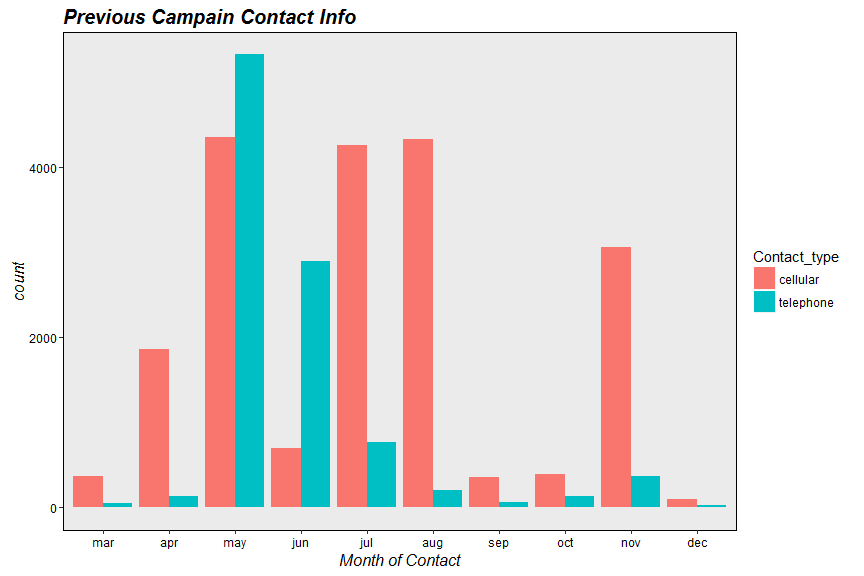
*By looking at the Boxplots, it seems the outcome is not influenced much by younger(<20) and older(>60) clients. We are not going to include them in further analysis.*

BankData.v2 <- BankData.v1[BankData.v1$Age >= 20 & BankData.v1$Age <= 60, ]  
attach(BankData.v2)

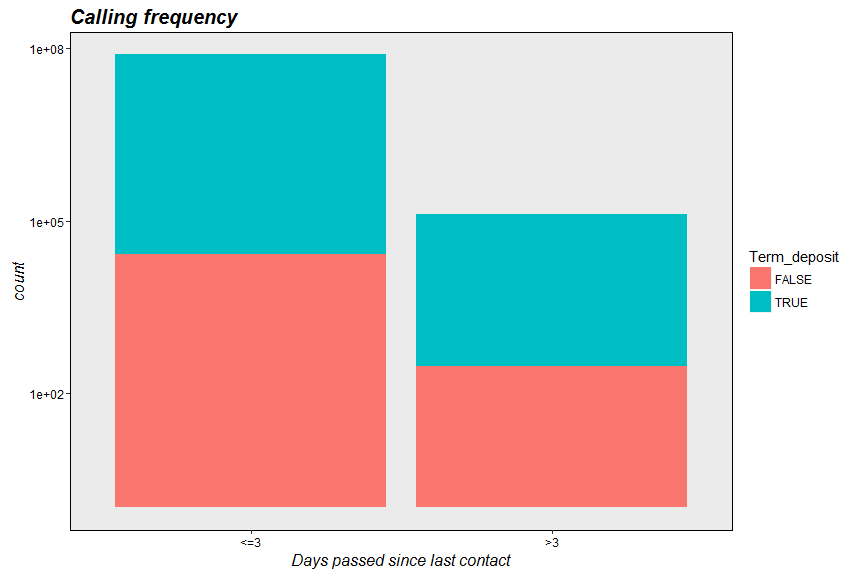
In comparison with other people, **Unemployed people** seem to have slight inclination towards term deposit.



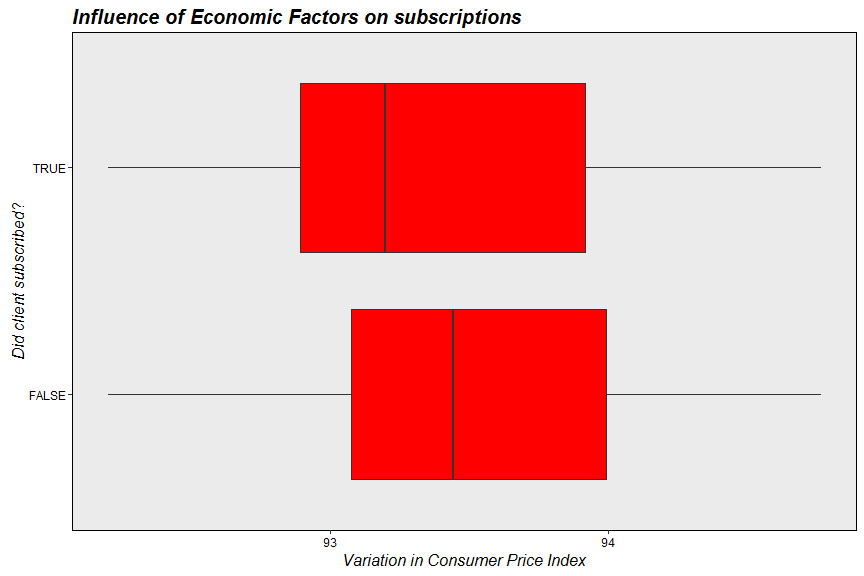
The bank seem to have contacted clients throughout the year, mostly around **May-Aug** period. Also the most common *mode of communication* was client cell phone.



The clients from previous campaign seem to show higher subscription rate on **recontact**. '*No contact*' means that the client was not contacted during previous campaign.



The below Boxplot clearly indicates the influence of economic factors on campaign outcome. You may want to check similar behavior with other economic factors.



You may draw more graphs to understand the data further. But I hope you get the gist of this exercise step. Data exploration not only helps one to comprehend the problem context in depth but also gives an opportunity to rectify the data anomalies.

## 3. Build and Evaluate the Model

### Build the model

*Lets build the formula first..*

X <- setdiff(colnames(BankData.v2), c("Campaign", "Duration", "Previous\_outcome",   
 "Contact\_gapdays", "Contact\_month", "Term\_deposit"))  
Y <- "Term\_deposit"  
Formula <- paste(Y, paste(X, collapse = "+"), sep = "~")

*Split the dataset in training and test set..*

BankData.v2$Sampling <- runif(dim(BankData.v2)[1])  
TrainSet <- subset(BankData.v2, BankData.v2$Sampling > 0.1)  
TestSet <- subset(BankData.v2, BankData.v2$Sampling < 0.1)

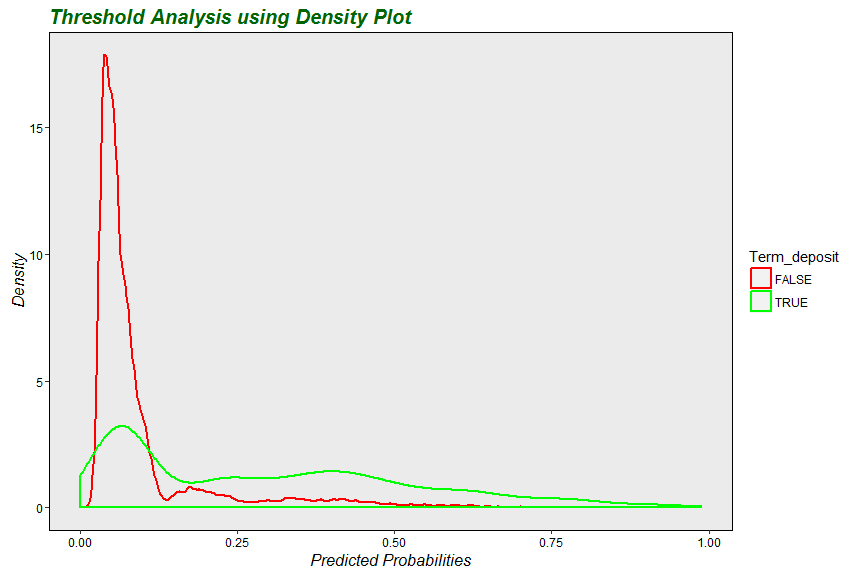
*Fitting the logistic regression model..*

# weights <- ifelse(Term\_deposit == TRUE, 1 -  
# sum(Term\_deposit)/length(Term\_deposit),sum(Term\_deposit)/length(Term\_deposit))  
# weights = 1 - sum(Term\_deposit)/length(Term\_deposit) Model <- glm(Formula,  
# family = binomial(link = 'logit'), data = TrainSet,weights=weights)  
Model <- glm(Formula, family = binomial(link = "logit"), data = TrainSet)

*Making predictions..*

TrainSet$Term\_deposit\_pred <- predict(Model, newdata = TrainSet, type = "response")  
TestSet$Term\_deposit\_pred <- predict(Model, newdata = TestSet, type = "response")

Ideally, we'd like the distribution of scores to be separated, with the scores of the negative instances (NO) to be concentrated on the left, and the distribution for the positive instances to be concentrated on the right. In the current case, both distributions are concentrated on the left, meaning that both positive and negative instances score low. *This isn't surprising, since the positive instances are few*. There are more ways to find an appropriate threshold but we would go ahead with **0.16** for demonstration purposes.



### Evaluate the model

There are several ways to evaluate the logistic regression model. We will discuss few of them.

**Confusion Matrix with test data** : This can be used to find out the model accuracy, precision, sensitivity, specificity etc. The resulting classifier is low-precision, but identifies a set of potential subscribers.

# Confusion Matrix  
cmat.test <- table(Actual = TestSet$Term\_deposit, Predicted = TestSet$Term\_deposit\_pred >   
 0.16)  
cmat.test

# Predicted  
# Actual FALSE TRUE  
# FALSE 2347 382  
# TRUE 141 248

# Model Accuracy  
sum(cmat.test[1, 1], cmat.test[2, 2])/sum(cmat.test)

# [1] 0.8322643

# Model Precision  
cmat.test[2, 2]/sum(cmat.test[, 2])

# [1] 0.3936508

# Model Recall or Sensitivity  
cmat.test[2, 2]/sum(cmat.test[2, ])

# [1] 0.6375321

# Model Specificity  
cmat.test[1, 1]/sum(cmat.test[1, ])

# [1] 0.860022

# Enrich  
(cmat.test[2, 2]/sum(cmat.test[, 2]))/mean(as.numeric(TestSet$Term\_deposit))

# [1] 3.155278

**pchisq** : Perform pchisq to test significance of observed fit.

pchisq(Model$null.deviance - Model$deviance, Model$df.null - Model$df.residual)

# [1] 1

**pseudo R-squared** : It is the measure of how much of the deviance is *explained* by the model. In our case, it is quite low and we would want it to be higher under normal circumstances.

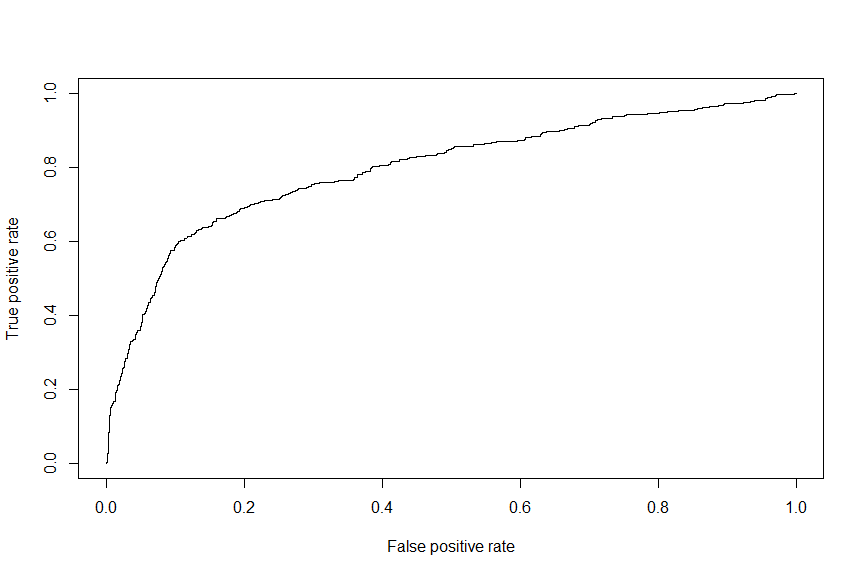
# TRAIN SET  
1 - (Model$deviance/Model$null.deviance)

# [1] 0.1980727

# TEST SET  
1 - (resid.dev.test/null.dev.test)

# [1] 0.2069353

**ROC CURVE** : This curve represents every possible trade-off between sensitivity and specificity that is available for this classifier.



**AUC** : As you can see, we would want to have much higher AUC than what we see. **A good classifier must have AUC close to 1.**

# AUC  
attributes(performance(eval, "auc"))$y.values[[1]]

# [1] 0.7968874

## 4. Model validation and Deployment

As I mentioned in my previous paper, these two steps would make more sense in real-scenario. Therefore, we will skip them for now.

## 5. Final note on the excercise!

So, we have come to the end of second exercise. Alike the last paper, this paper also covers set of steps usually taken in model-building process. We discussed logistic regression for this exercise.

Thank you again, if you have took time to read through my initial two papers. I would very happy to have your feedback and comments.

## 6. What's next?

You're probably wondering about what is next! Well, in my next paper I plan to implement other techniques on the same dataset and see if we can further improve the model. Stay tuned !