STAT – 615 – FINAL PRESENTATION

Marketing Analytics: To develop a scientific model for the prediction of the Response of the customers

Chaytanya Kumar & Pape Theodore Seye

1. Introduction

1.1 About Client

A leading FMCG company in India, which manufactures and sells a host of products like Biscuits, Coffee, Juice, Chocolates, and many more, has launched a new flavor variant of a coffee brand in January 2020.

They conducted a marketing campaign with their **channel Partners** (resellers, service providers, vendors, retailers, or agents) before the actual product launch.
For that campaign, the company introduced the product and a reward system (discount and tokens) through SMS, email, and call.

The campaign was run in January 2020 for the new variant of coffee and 1228 channel partners were contacted. The responses of the partners were tracked and saved to contact them again in the future.

1.2 Questions of interest

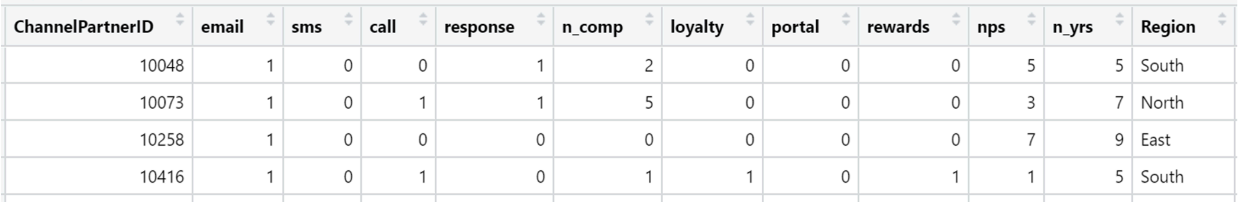
- Predict whether Channel partners will respond or not (0 or 1) based on various independent variables.
- What factors (variables) impact the chances of getting a response the most?
- Identifying the most effective communication channel/s for them.
- Who to target first in the next planned campaign?

1.3 Source of Data

It is a real-case dataset collected from one of the largest IT service providers that provide IT services for one of the largest FMG companies in India.

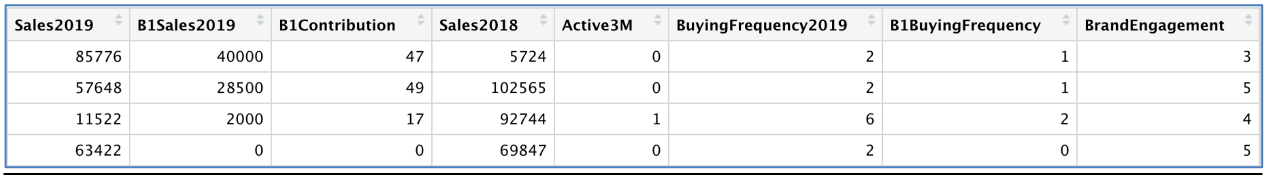
1.4 Channel Partner Information

Summary of data:



ChannelPartnerID	email	sms	call	response	n_comp	loyalty	portal	rewards	nps	n_yrs	Region
10048	1	0	0	1	2	0	0	0	5	5	South
10073	1	0	1	1	5	0	0	0	3	7	North
10258	1	0	0	0	0	0	0	0	7	9	East
10416	1	0	1	0	1	1	0	1	1	5	South

- **Email / sms / call** : method for contacting the channel partner and the number of times contacted through that method.
- **Response** : whether the channel partner responded or not
- **n_comp** : number of complaints made by the channel partner in the last three months
- **loyalty** : is the channel partner member of the loyalty program or not
- **portal** : is the channel partner active on the web portal
- **rewards** : did the channel partner redeem any reward points last month
- **nps** : Net Promoter Score (score that represents the partners' satisfaction with products)
- **n_yrs** : number of years in business with the company
- **Region** : Region in which the channel partner is located.



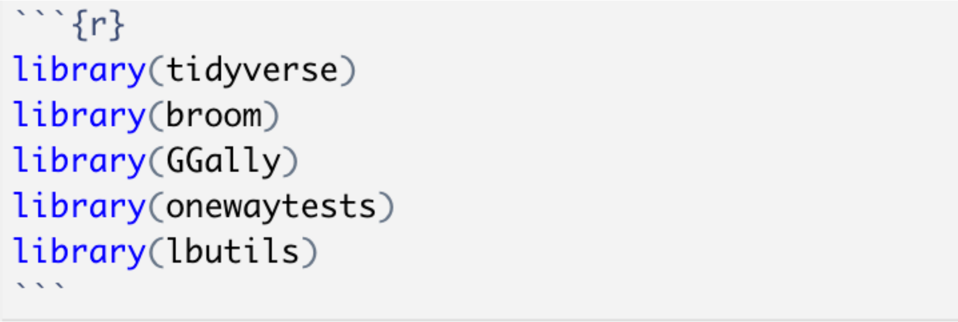
Sales2019	B1Sales2019	B1Contribution	Sales2018	Active3M	BuyingFrequency2019	B1BuyingFrequency	BrandEngagement
85776	40000	47	5724	0	2	1	3
57648	28500	49	102565	0	2	1	5
11522	2000	17	92744	1	6	2	4
63422	0	0	69847	0	2	0	5

- **Sales2019 :** Annual sales for 2019
- **B1Sales2019 :** Annual sales for brand B1 in 2019

- **B1Contribution :** percentage of B1 in the total sales
- **Sales2018 :** Annual sales for 2018
- **Active3M :** whether or not the partner purchased any product in the last 3 months
- **BuyingFrequenciy2019 :** Number of unique months for which products were purchased
- **B1BuyingFrequency :** Same but only for B1 product
- **BrandEngagement :** Number of brands the channel partner is engaged with

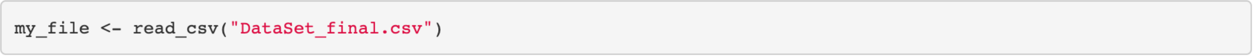
2. Data visualization and exploratory analysis

- load the needed libraires :



~~~
```{r}
library(tidyverse)
library(broom)
library(GGally)
library(onewaytests)
library(lbutils)
```
~~~

- load the data :



```
my_file <- read_csv("DataSet_final.csv")
```

- Compute the response rates (rate for ‘response’ and rate for ‘no response’)

```
response <- table(my_file$response)
Responserate <- (response[2]/(response[1]+response[2]))*100
Responserate[2] <- (response[1]/(response[1]+response[2]))*100
names(Responserate) <- c(1,0)
Responserate
```

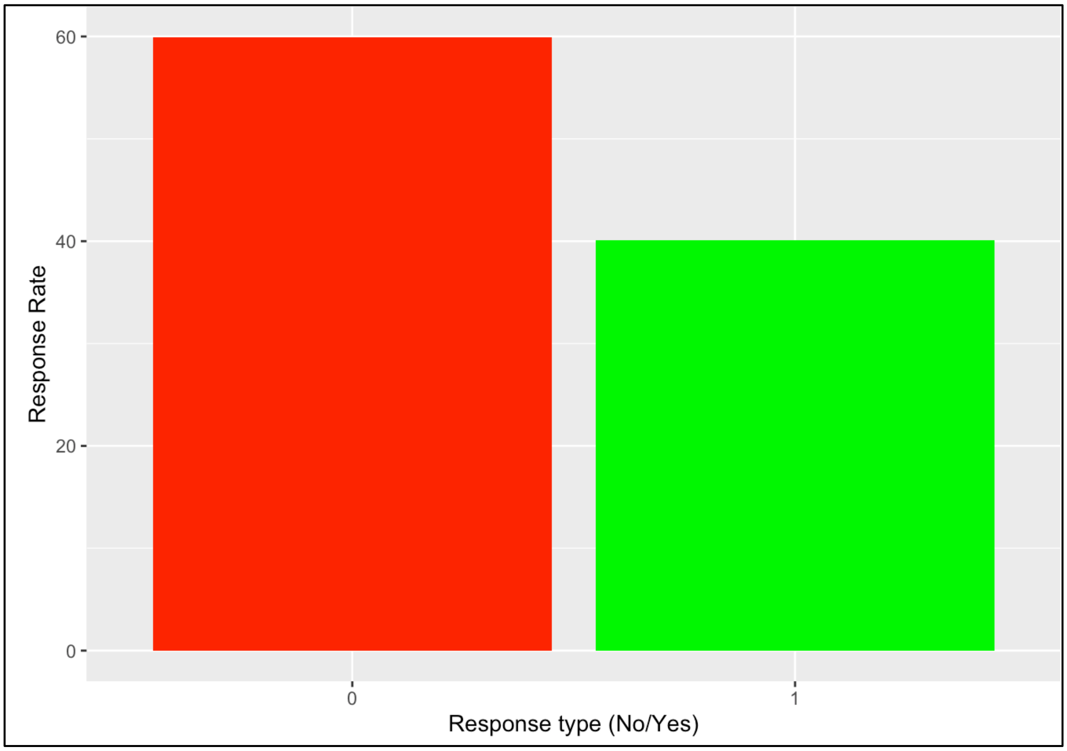
```
##        1        0
## 40.06515 59.93485
```

```
Res <- data.frame(response=c(1,0),
                  rate=c(40.06515,59.93485))
Res
```

```
##   response     rate
## 1        1 40.06515
## 2        0 59.93485
```

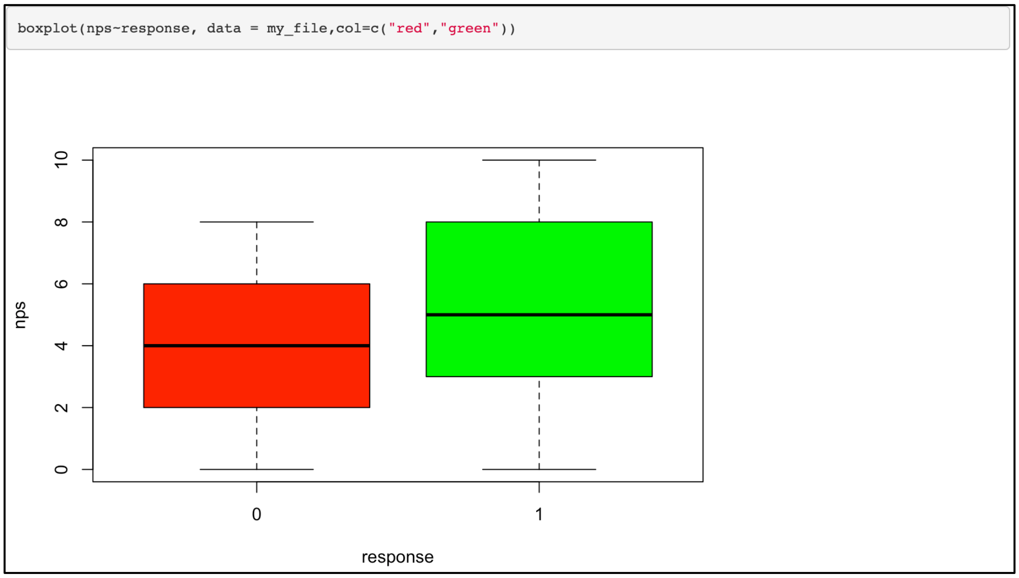
- Visual comparison of the two response rates :

```
Res$response =as.factor(Res$response)


ggplot(Res,aes(x=response,y=rate))+
  geom_bar(stat = "identity",fill = c("green", "red"))+
  labs(x="Response type (No/Yes)",y="Response Rate")
```



A) Boxplot analysis: boxplots for the different responses (0 and 1) over different relevant variables.

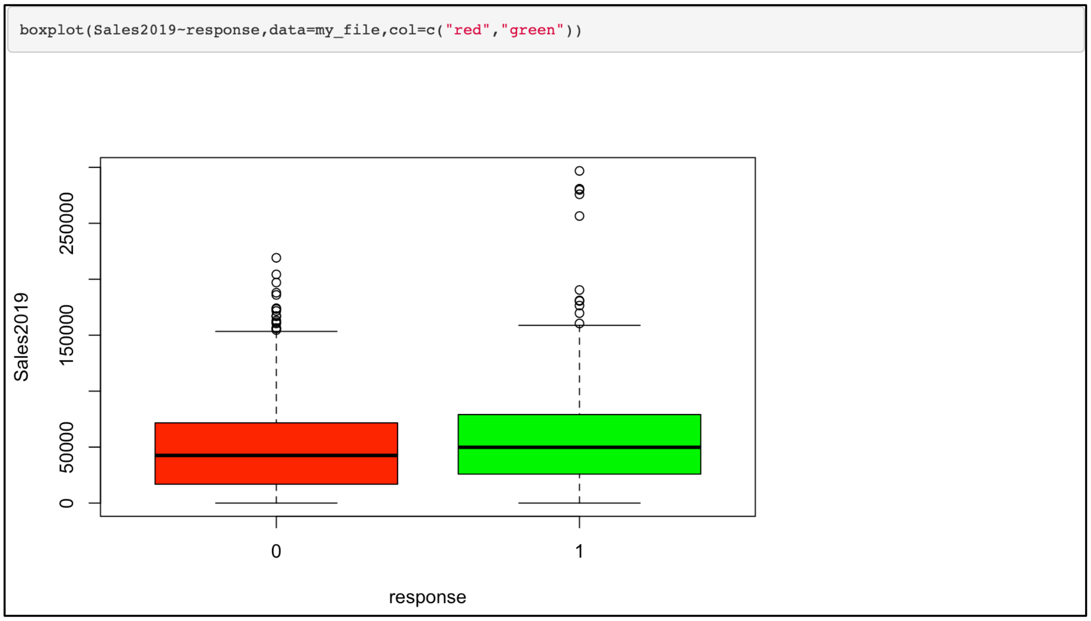
- **For NPS**



```
boxplot(nps~response, data = my_file,col=c("red","green"))
```

➢ We see that channel partners with higher NPS, responded more on average ; which is not surprising.

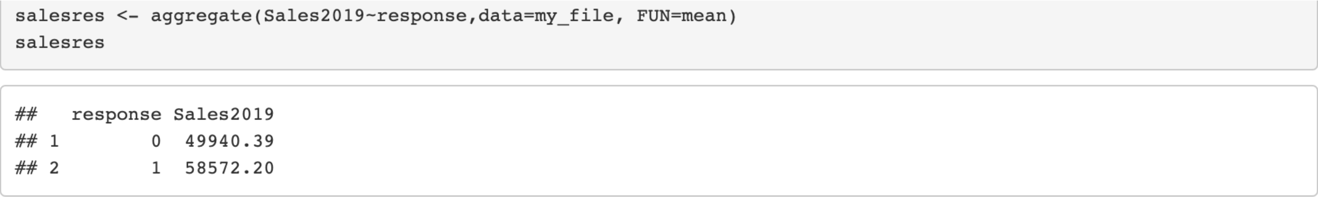
- **For Sales 2019**



```
boxplot(Sales2019~response,data=my_file,col=c("red","green"))
```

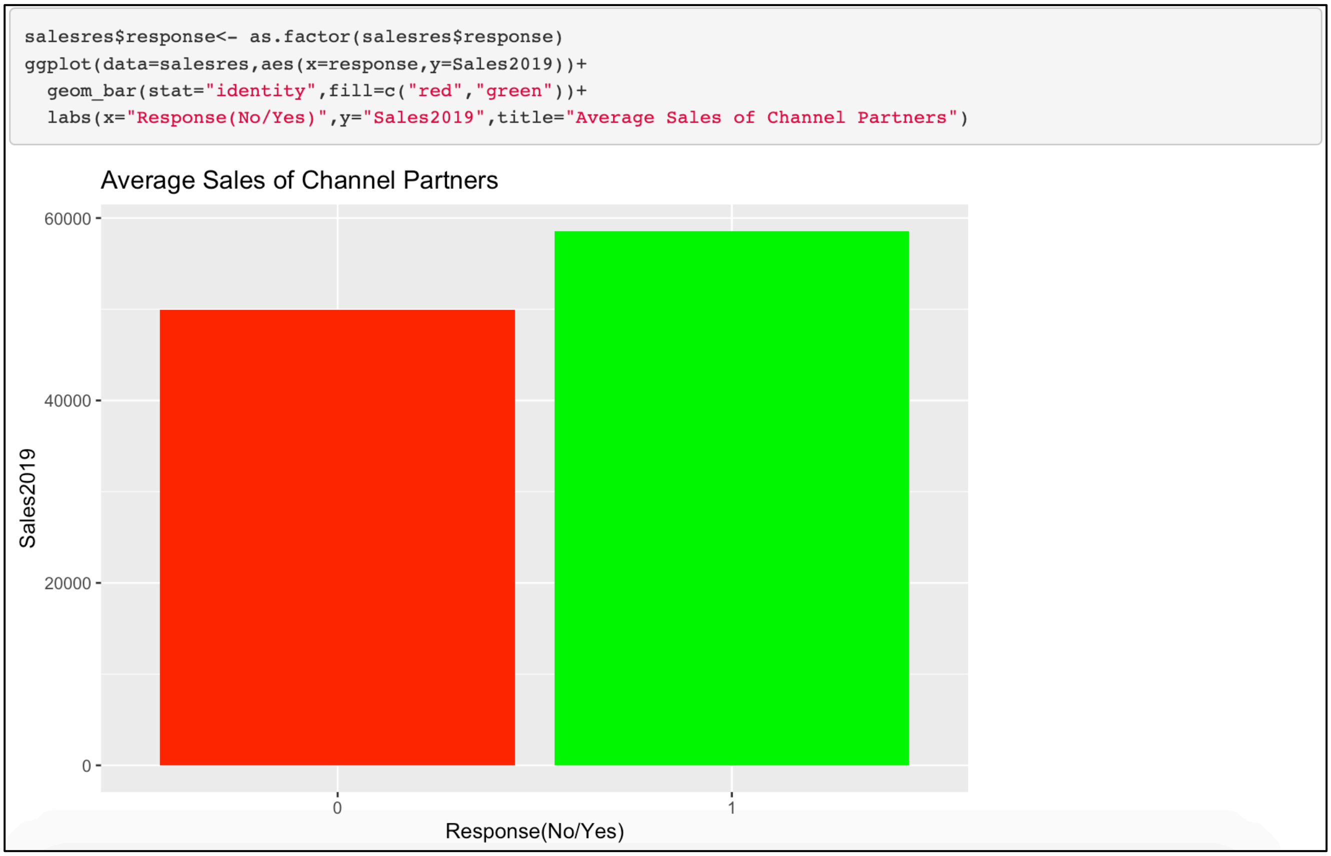
➢ Presence of many outliers. But for the core of the data, it seems like the respondents and the non respondents had on average similar Sales2019.
So does the value of Sales2019 has great impact on the probability of getting a response ?

➢ Since the last boxplots had many outliers and since boxplot indicate the median and not the mean let’s compute the mean of Sales2019 of each group (respondents and non-respondents).



```
salesres <- aggregate(Sales2019~response,data=my_file, FUN=mean)
salesres
```

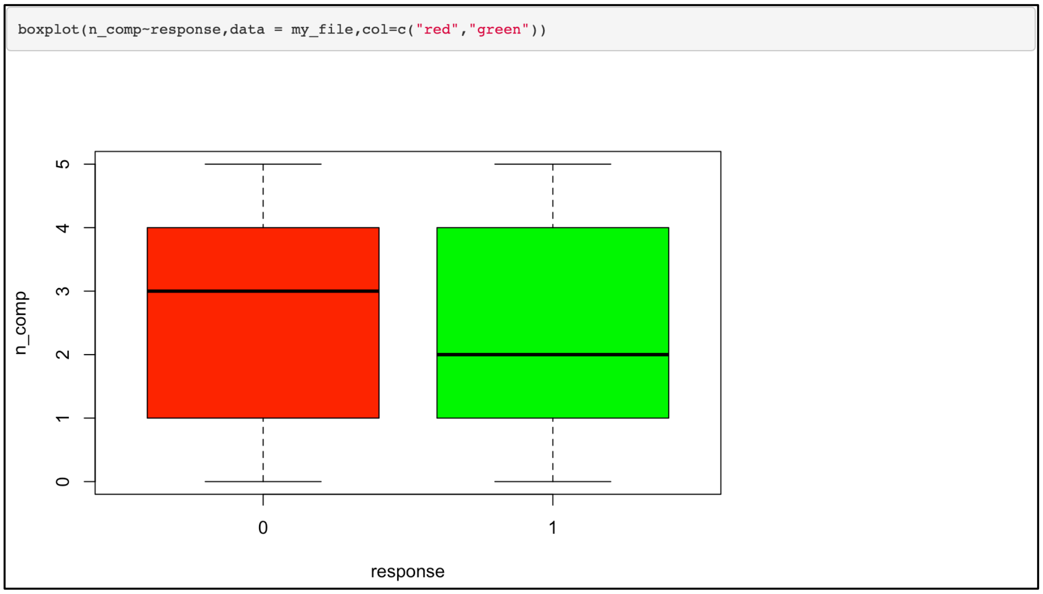
```
##   response Sales2019
## 1        0  49940.39
## 2        1  58572.20
```



```
salesres$response<- as.factor(salesres$response)
ggplot(data=salesres,aes(x=response,y=Sales2019))+
  geom_bar(stat="identity",fill=c("red","green"))+
  labs(x="Response(No/Yes)",y="Sales2019",title="Average Sales of Channel Partners")
```

- We see that those who responded have a higher mean for Sales2019 than those who did not. And the difference between the two is of 14.7%

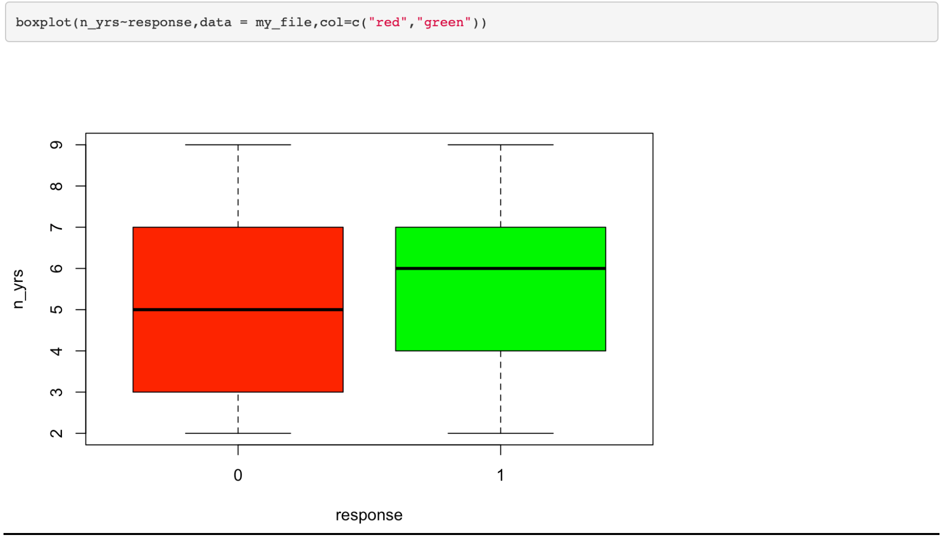
- **<u>For n-comp</u>**



```
boxplot(n_comp~response,data = my_file,col=c("red","green"))
```

- ➢ Even though the two ranges are similar, we observe that those who responded have a much lower median value of number of complains in the last 3 months. Which again, is not surprising.

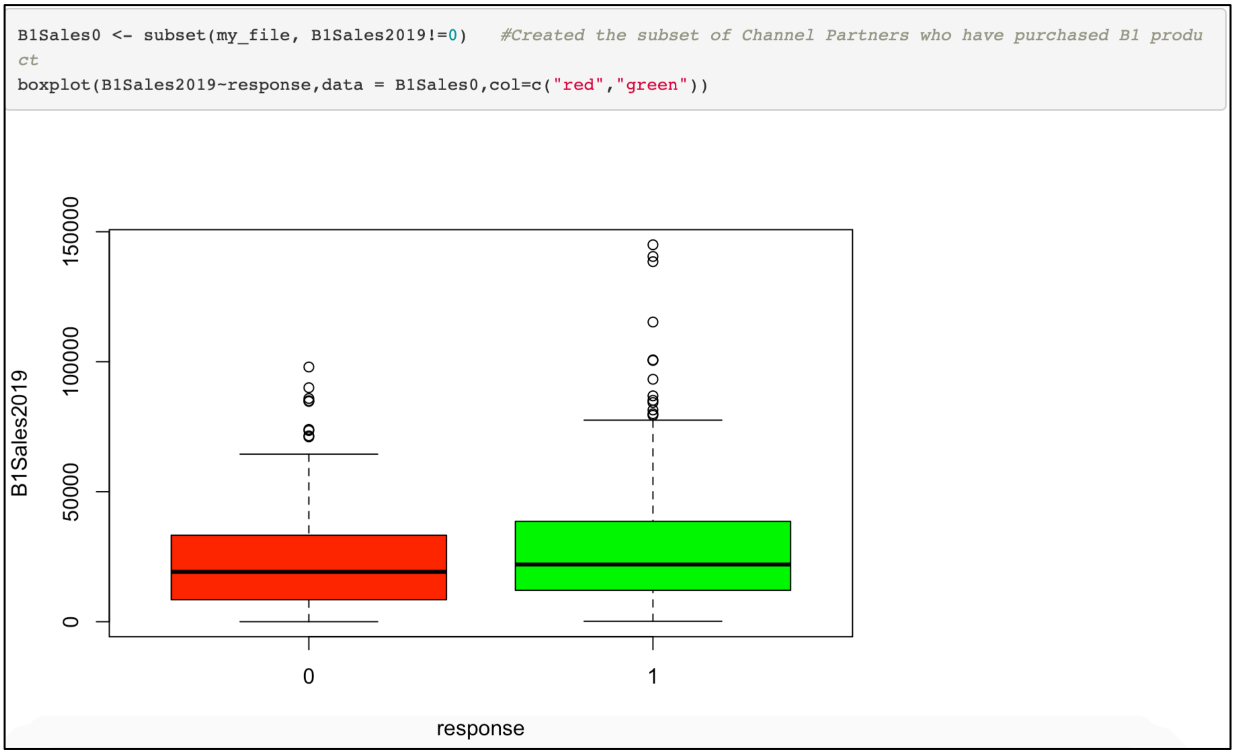
- **<u>For n-years</u>**



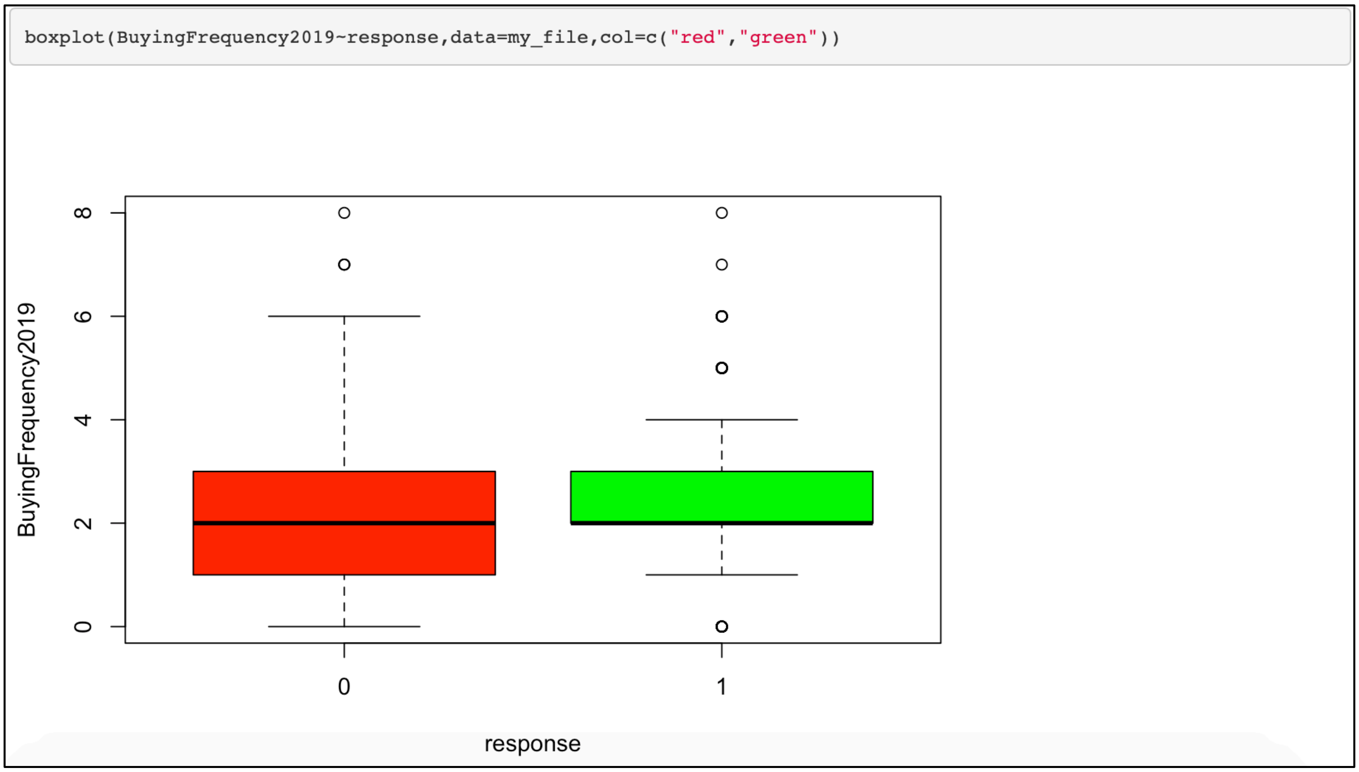
```
boxplot(n_yrs~response,data = my_file,col=c("red","green"))
```

The group who responded has a higher median value for n_yrs (number of years in business with the company.

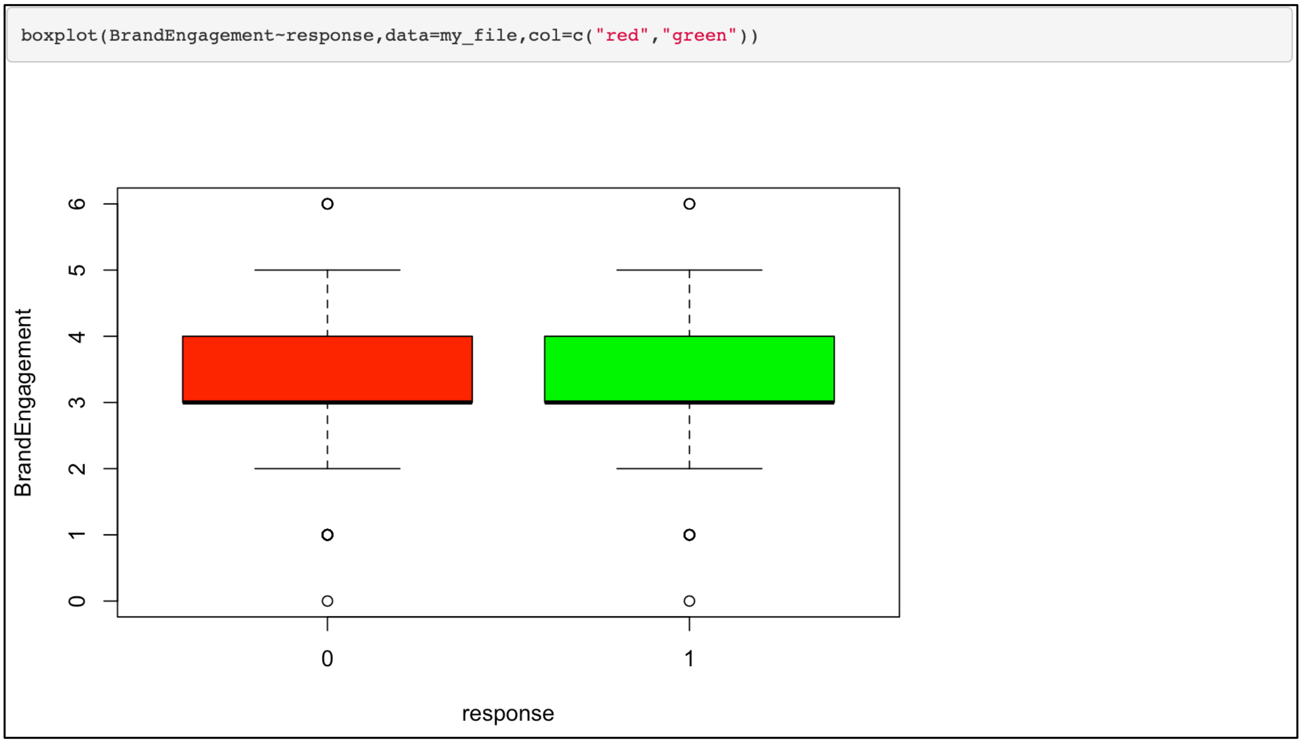
- **<u>For some other variables :</u>**



```
B1Sales0 <- subset(my_file, B1Sales2019!=0)   #Created the subset of Channel Partners who have purchased B1 produ
ct
boxplot(B1Sales2019~response,data = B1Sales0,col=c("red","green"))
```



```
boxplot(BuyingFrequency2019~response,data=my_file,col=c("red","green"))
```

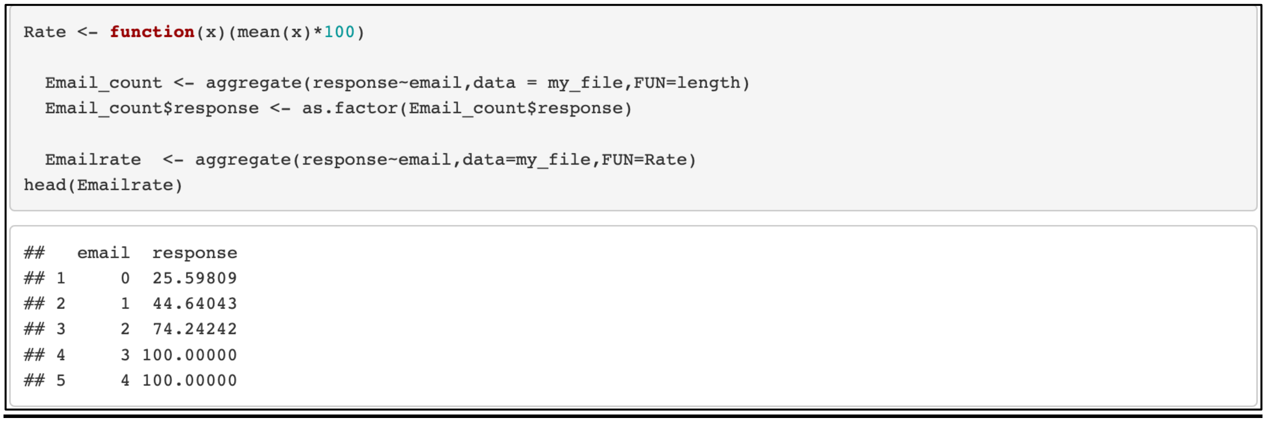


```
boxplot(BrandEngagement~response,data=my_file,col=c("red","green"))
```

B) Comparison of the response rates over the different communication channels.

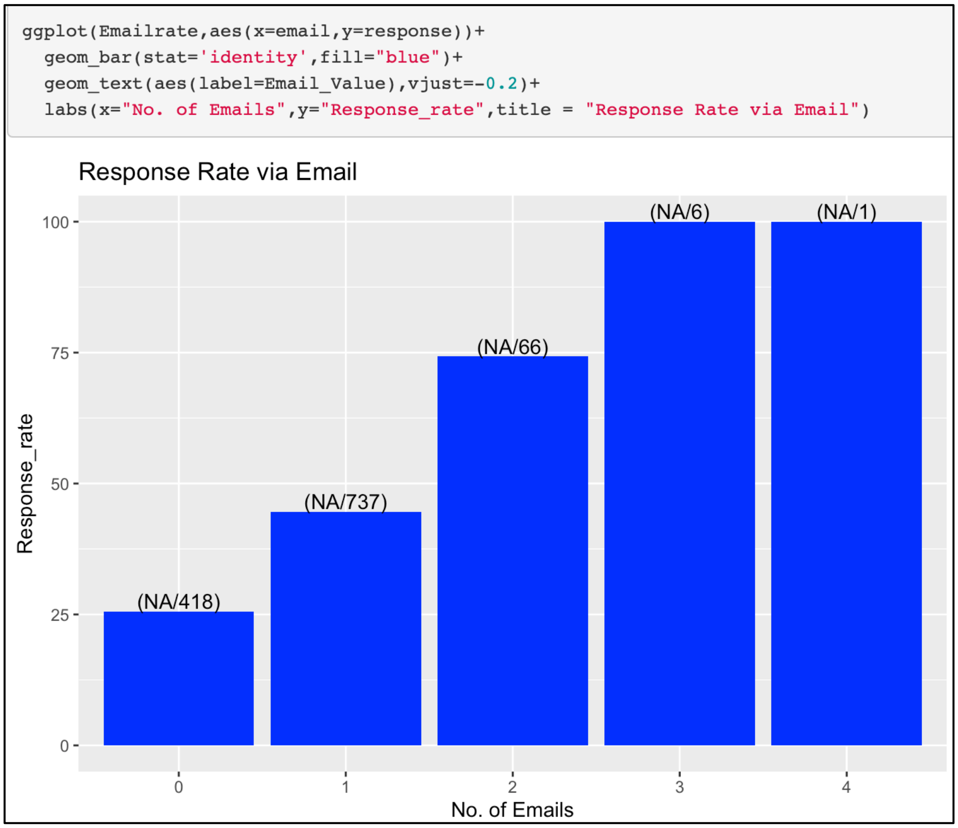
We're interested in seeing how the response rate changes when the number of messages sent changes for different communication channels.

- **Emails :**



```
Rate <- function(x)(mean(x)*100)

  Email_count <- aggregate(response~email,data = my_file,FUN=length)
  Email_count$response <- as.factor(Email_count$response)

  Emailrate  <- aggregate(response~email,data=my_file,FUN=Rate)
head(Emailrate)
```

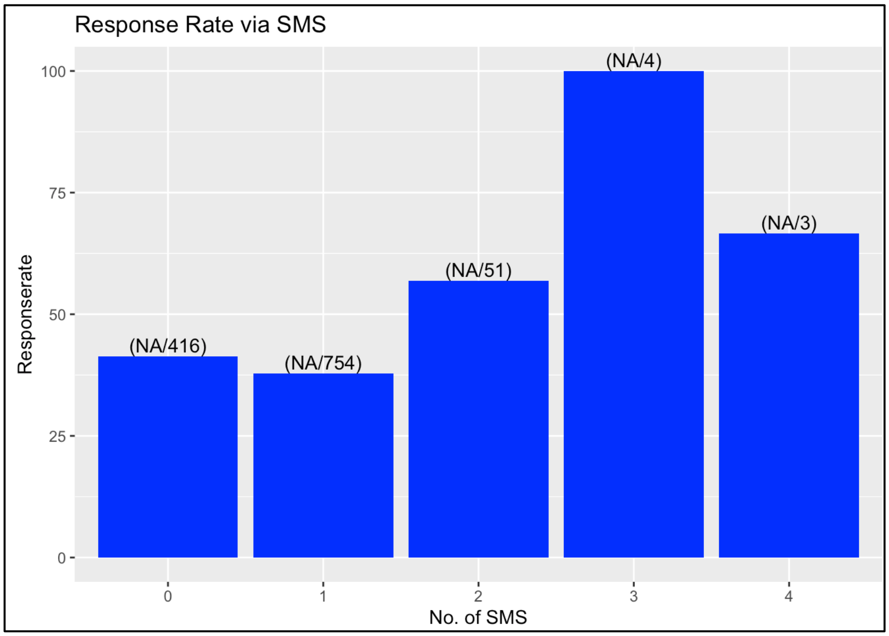
```
##   email  response
## 1     0  25.59809
## 2     1  44.64043
## 3     2  74.24242
## 4     3 100.00000
## 5     4 100.00000
```



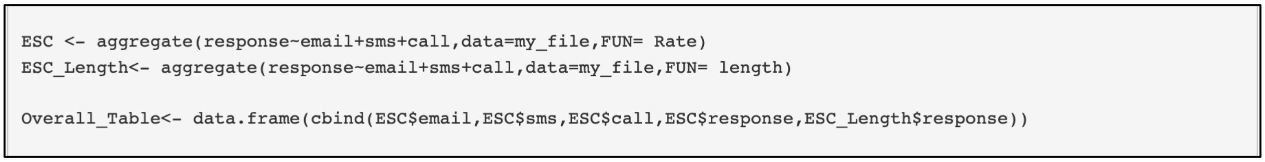
```
ggplot(Emailrate,aes(x=email,y=response))+
  geom_bar(stat='identity',fill="blue")+
  geom_text(aes(label=Email_Value),vjust=-0.2)+
  labs(x="No. of Emails",y="Response_rate",title = "Response Rate via Email")
```

Similarly,

- **<u>For SMS</u> :**

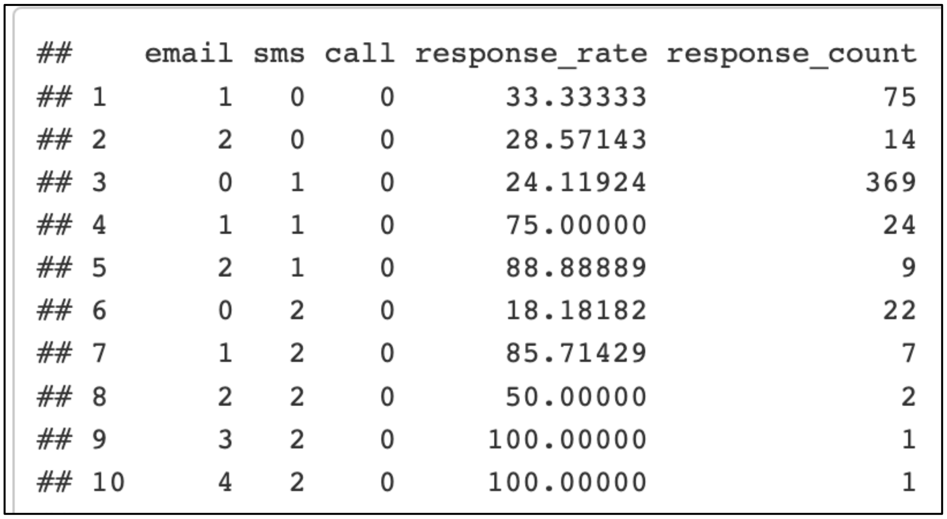


- **Now what if we put calls, SMS and emails together and see how the response rate changes for different combinations of values ?**



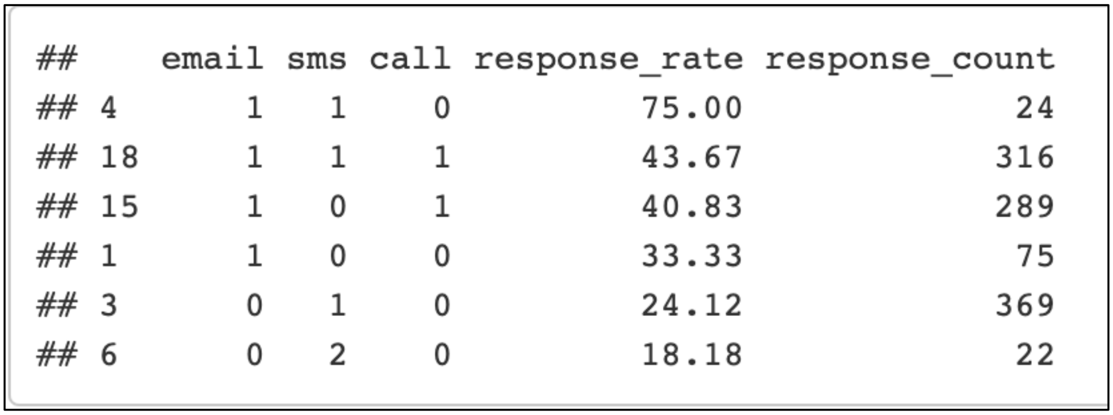
```
ESC <- aggregate(response~email+sms+call,data=my_file,FUN= Rate)
ESC_Length<- aggregate(response~email+sms+call,data=my_file,FUN= length)

Overall_Table<- data.frame(cbind(ESC$email,ESC$sms,ESC$call,ESC$response,ESC_Length$response))
```

➢ We get a table with all the combinations possible ; the table has 44 observations, here are the first 10 :



```
##    email sms call response_rate response_count
## 1      1   0    0      33.33333             75
## 2      2   0    0      28.57143             14
## 3      0   1    0      24.11924            369
## 4      1   1    0      75.00000             24
## 5      2   1    0      88.88889              9
## 6      0   2    0      18.18182             22
## 7      1   2    0      85.71429              7
## 8      2   2    0      50.00000              2
## 9      3   2    0     100.00000              1
## 10     4   2    0     100.00000              1
```

➢ We filter it to obsevations with response count higher than 20 and reorder it :



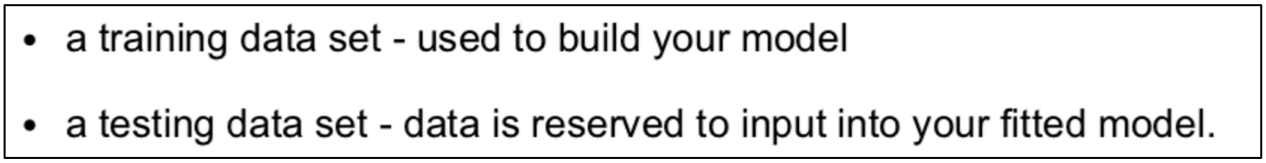
```
##    email sms call response_rate response_count
## 4      1   1    0         75.00             24
## 18     1   1    1         43.67            316
## 15     1   0    1         40.83            289
## 1      1   0    0         33.33             75
## 3      0   1    0         24.12            369
## 6      0   2    0         18.18             22
```

➢ So we get the combinations of values that gives us the highest response rates.

3. Model Building

- After having a better look at how our data looks graphically, we can now start with building of our model.

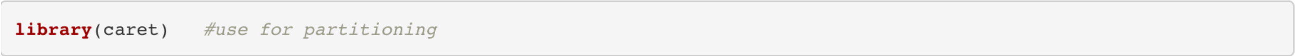
- As suggested by the class notes of model buidling, before running the model we need to partition our data into two groups for the following purpose :



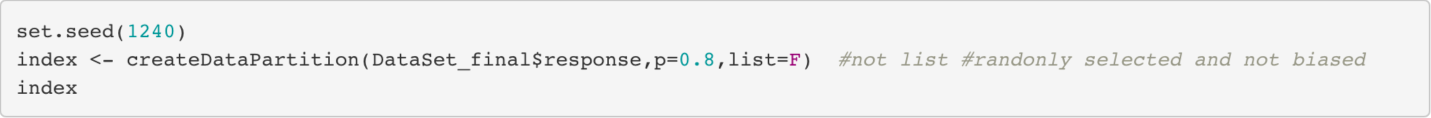
- a training data set - used to build your model
- a testing data set - data is reserved to input into your fitted model.

  - This will be used at the end to validate the quality of our model.

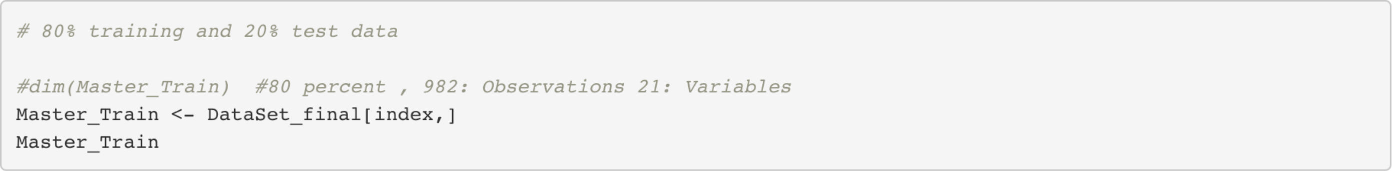
1. Partitioning of the data :



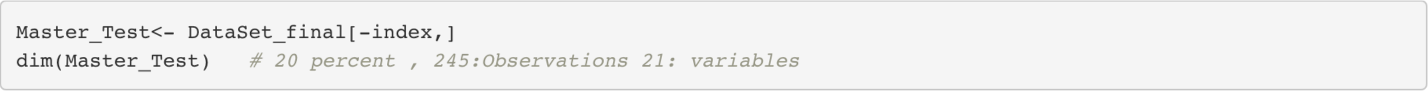
```
library(caret)   #use for partitioning
```



```
set.seed(1240)
index <- createDataPartition(DataSet_final$response,p=0.8,list=F)  #not list #randonly selected and not biased
index
```



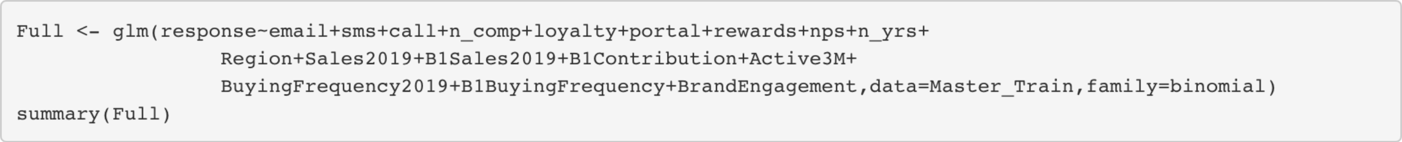
```
# 80% training and 20% test data

#dim(Master_Train)  #80 percent , 982: Observations 21: Variables
Master_Train <- DataSet_final[index,]
Master_Train
```



```
Master_Test<- DataSet_final[-index,]
dim(Master_Test)   # 20 percent , 245:Observations 21: variables
```

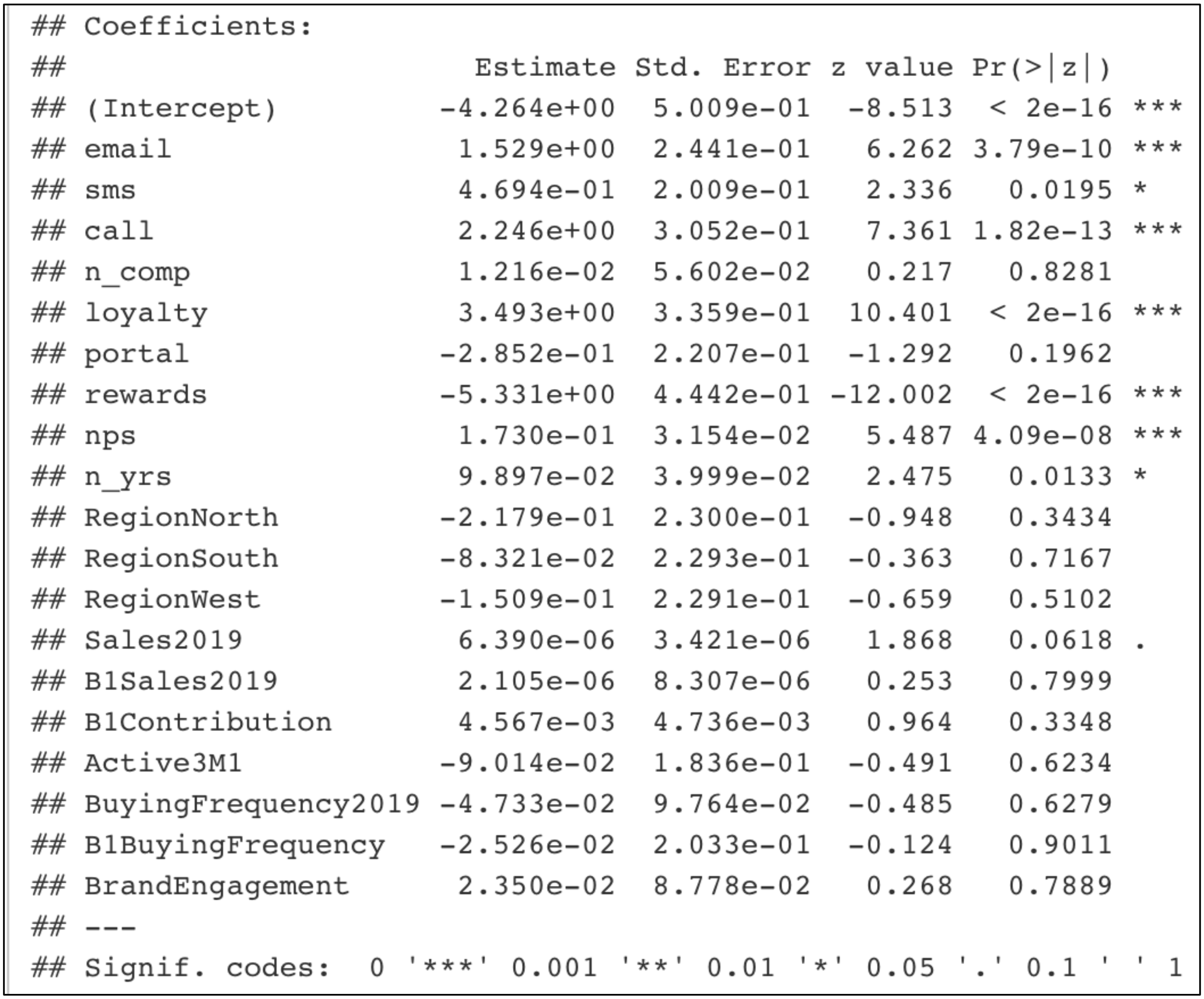
- ➢ We now have a Train data set, made of random observations from the full data set (80% of full data set size)
- ➢ And a Test data set, that is made of the rest of the observations.

2. Running the binary logistic model using the Train data and 'glm' function :



```
Full <- glm(response~email+sms+call+n_comp+loyalty+portal+rewards+nps+n_yrs+
                Region+Sales2019+B1Sales2019+B1Contribution+Active3M+
                BuyingFrequency2019+B1BuyingFrequency+BrandEngagement,data=Master_Train,family=binomial)
summary(Full)
```

- We get the following table :

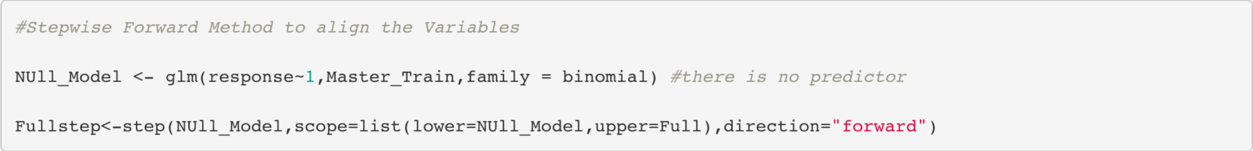


```
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)    
## (Intercept)         -4.264e+00  5.009e-01  -8.513  < 2e-16 ***
## email                1.529e+00  2.441e-01   6.262 3.79e-10 ***
## sms                  4.694e-01  2.009e-01   2.336   0.0195 *  
## call                 2.246e+00  3.052e-01   7.361 1.82e-13 ***
## n_comp               1.216e-02  5.602e-02   0.217   0.8281    
## loyalty              3.493e+00  3.359e-01  10.401  < 2e-16 ***
## portal              -2.852e-01  2.207e-01  -1.292   0.1962    
## rewards             -5.331e+00  4.442e-01 -12.002  < 2e-16 ***
## nps                  1.730e-01  3.154e-02   5.487 4.09e-08 ***
## n_yrs                9.897e-02  3.999e-02   2.475   0.0133 *  
## RegionNorth         -2.179e-01  2.300e-01  -0.948   0.3434    
## RegionSouth         -8.321e-02  2.293e-01  -0.363   0.7167    
## RegionWest          -1.509e-01  2.291e-01  -0.659   0.5102    
## Sales2019            6.390e-06  3.421e-06   1.868   0.0618 .  
## B1Sales2019          2.105e-06  8.307e-06   0.253   0.7999    
## B1Contribution       4.567e-03  4.736e-03   0.964   0.3348    
## Active3M1           -9.014e-02  1.836e-01  -0.491   0.6234    
## BuyingFrequency2019 -4.733e-02  9.764e-02  -0.485   0.6279    
## B1BuyingFrequency   -2.526e-02  2.033e-01  -0.124   0.9011    
## BrandEngagement      2.350e-02  8.778e-02   0.268   0.7889    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3. Using the ‘step’ function on the model to have the best combination of explanatory variables:

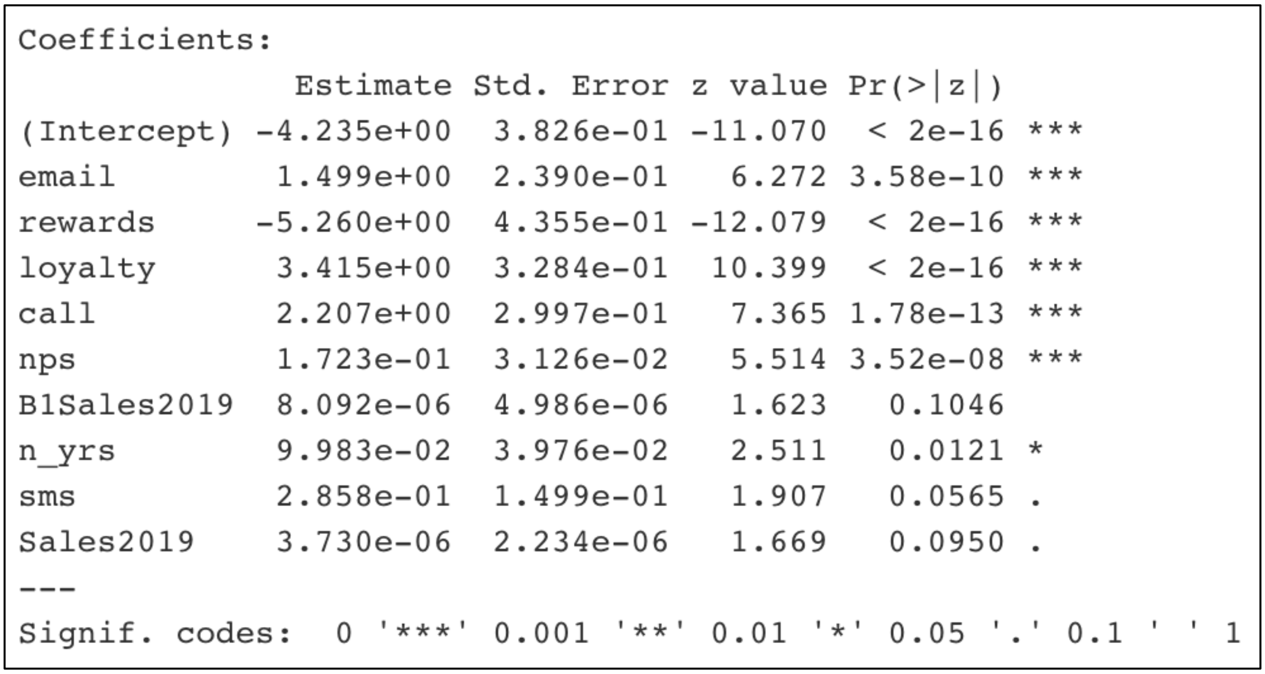
- **a.** It uses the AIC to compare the ‘quality’ of different models.
- **b.** Smaller AIC indicates a better fitting model
- **c.** So ‘step’ starts from a model with no explanatory variables and keeps on adding variables as long as the AIC is getting lower. It then provides us with the model with best AIC at the end of the procedure.

Implementing the code in R :



```
#Stepwise Forward Method to align the Variables

NUll_Model <- glm(response~1,Master_Train,family = binomial) #there is no predictor

Fullstep<-step(NUll_Model,scope=list(lower=NUll_Model,upper=Full),direction="forward")
```

- We get the following model at the end of the procedure :

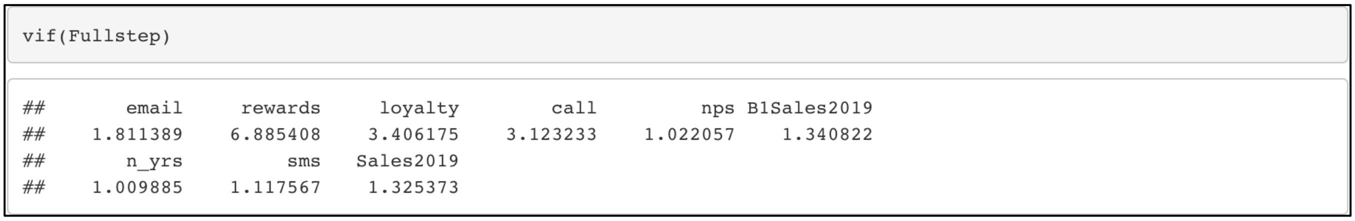


```
Coefficients:
              Estimate Std. Error z value Pr(>|z|)    
(Intercept) -4.235e+00  3.826e-01 -11.070  < 2e-16 ***
email        1.499e+00  2.390e-01   6.272 3.58e-10 ***
rewards     -5.260e+00  4.355e-01 -12.079  < 2e-16 ***
loyalty      3.415e+00  3.284e-01  10.399  < 2e-16 ***
call         2.207e+00  2.997e-01   7.365 1.78e-13 ***
nps          1.723e-01  3.126e-02   5.514 3.52e-08 ***
B1Sales2019  8.092e-06  4.986e-06   1.623   0.1046    
n_yrs        9.983e-02  3.976e-02   2.511   0.0121 *  
sms          2.858e-01  1.499e-01   1.907   0.0565 .  
Sales2019    3.730e-06  2.234e-06   1.669   0.0950 .  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Final Model Equation:

Ln(p/1-p) = - 4.235 + 1.49 (email) - 5.26 (rewards) + 3.41 (Loyalty) +2.20 (Call) + 0.17 (nps) + 0.000008092 (B1Sales 2019) + 0.09983 (n_yrs) + 0.2858 (sms) + 0.000003730 (Sales 2019)

4. **Check for Multicollinearity:**



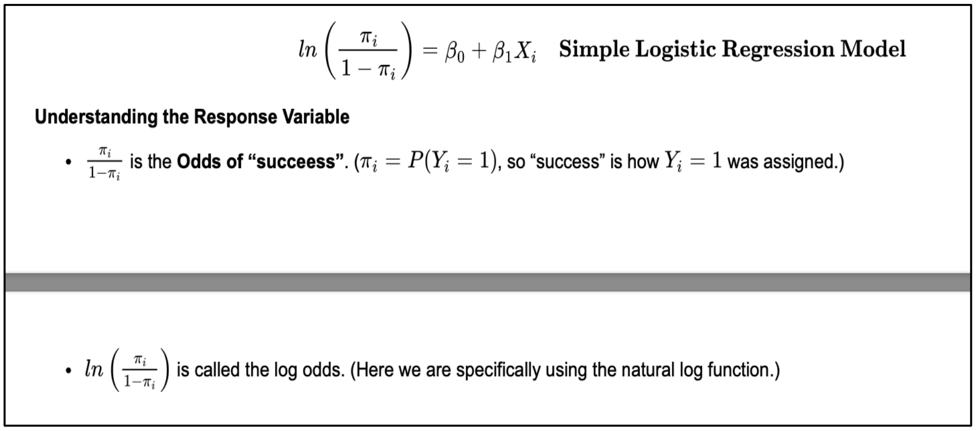
```
vif(Fullstep)
```

```
##       email     rewards     loyalty        call         nps B1Sales2019
##    1.811389    6.885408    3.406175    3.123233    1.022057    1.340822
##       n_yrs         sms   Sales2019
##    1.009885    1.117567    1.325373
```

- VIF for ‘rewards’ seems high but we decided to still keep it in the model (i.e. considering a critical value for VIF higher than usual).
- We made that choice because removing ‘rewards’ from the model made some other important variables non-significant. So removal of ‘rewards’ would make us deviate too much from the step-model.
- Other indicators of model quality (AUC) also dropped after removal of rewards.

Revised Final model :

-4.235 + 1.44 (email) - 5.26 (rewards) + 3.41 (Loyalty) +2.20 (Call) + 0.17 (nps) + 0.000008092
(B1Sales 2019) + 0.09983 (n_yrs) + 0.2858 (sms) + 0.000003730 (Sales 2019)



$$ln\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0 + \beta_1 X_i$$ **Simple Logistic Regression Model**

Understanding the Response Variable

- $\frac{\pi_i}{1-\pi_i}$ is the **Odds of "succeess"**. ($\pi_i = P(Y_i = 1)$, so "success" is how $Y_i = 1$ was assigned.)

- $ln\left(\frac{\pi_i}{1-\pi_i}\right)$ is called the log odds. (Here we are specifically using the natural log function.)

5. **Interpretation:**

Holding all other variables constant, on average, every increase in the no. of emails sent to the channel partners of the FMG company, increases the odds of getting the customer response by a factor of 4.22.

Holding all other variables constant, on average, the channel partners of the FMG company that gets a reward decreases the odds of getting the response by a factor 0.005195 compared to the channel Partner without the reward.

Holding all other variables constant, on average, the channel partners of the FMG company that are part of the loyalty program increases the odds of getting the response by a factor 30.2 compared to that of the channel that is not a part of the loyalty program.

Holding all other variables constant, on average, for every increase in the number of calls made to the channel partners of the FMG company, increases the odds of getting the response by a factor of 9.02.

Holding all other variables constant, on average, for every increase in the NPS by one score made by channel partners of the FMG company, increases the odds of getting the response by a factor of 1.18.

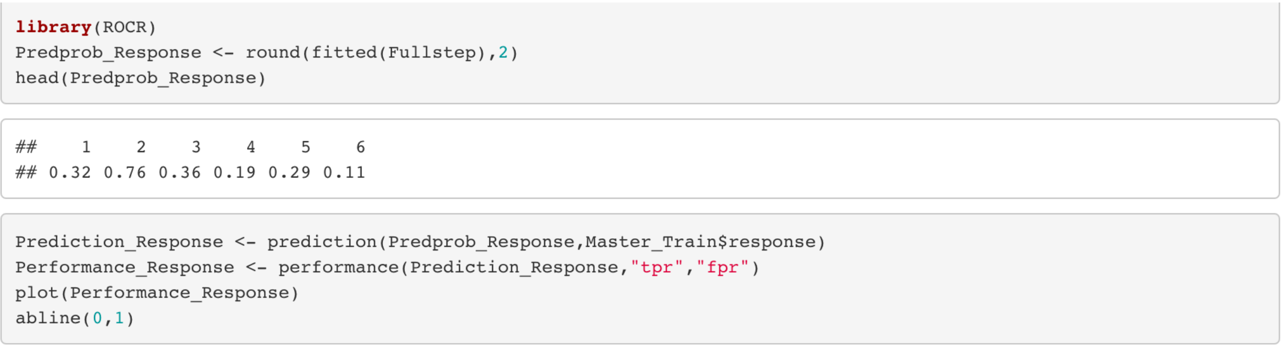
Holding all other variables constant, on average, for every increase in the no. of SMS sent to the channel partners of the FMG company, increases the odds of getting the customer response by a factor of 1.32.

6. Measuring the overall performance of the model:

The ROC curve is plotted with TPR against the FPR where TPR is on the y-axis and FPR is on the x-axis.

False-positive – when my output value (actual) is 0 but my predicted value is 1. (Type 1error)
True positive – when my output value is 1 and my predicited values is 1.
True Negative – when my output value is 0 and my predictied value is 0.
False Negative- when my output value is 1 but my predicted value is 0. (Type 2 error)

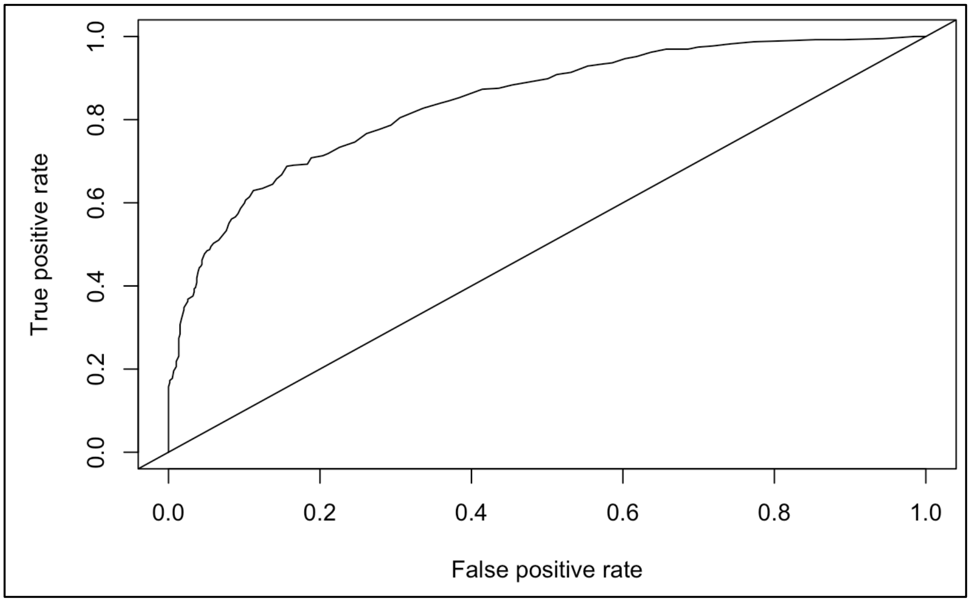
- ➢ For that we use the ROC graph and AUC value. AUC is based on the ratio of tpr : true positive rate and fpr : false positive rate.



```
library(ROCR)
Predprob_Response <- round(fitted(Fullstep),2)
head(Predprob_Response)
```

```
##    1    2    3    4    5    6
## 0.32 0.76 0.36 0.19 0.29 0.11
```

```
Prediction_Response <- prediction(Predprob_Response,Master_Train$response)
Performance_Response <- performance(Prediction_Response,"tpr","fpr")
plot(Performance_Response)
abline(0,1)
```



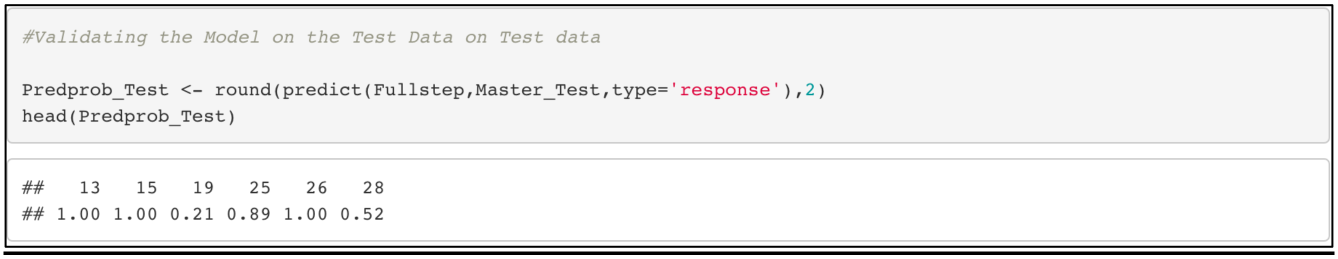
```
Auc_Response <- performance(Prediction_Response,"auc")
Auc_Response@y.values  # [1] 0.84 = 84.27%
```

```
## [[1]]
## [1] 0.8427322
```

- We, therefore, get a value of 84.27% for our model’s AUC , which is closer to value 1 and much greater than .5. Which is pretty good.
- means there is a 84.27% chance that the model will be able to distinguish between response and no response from the channel partners.

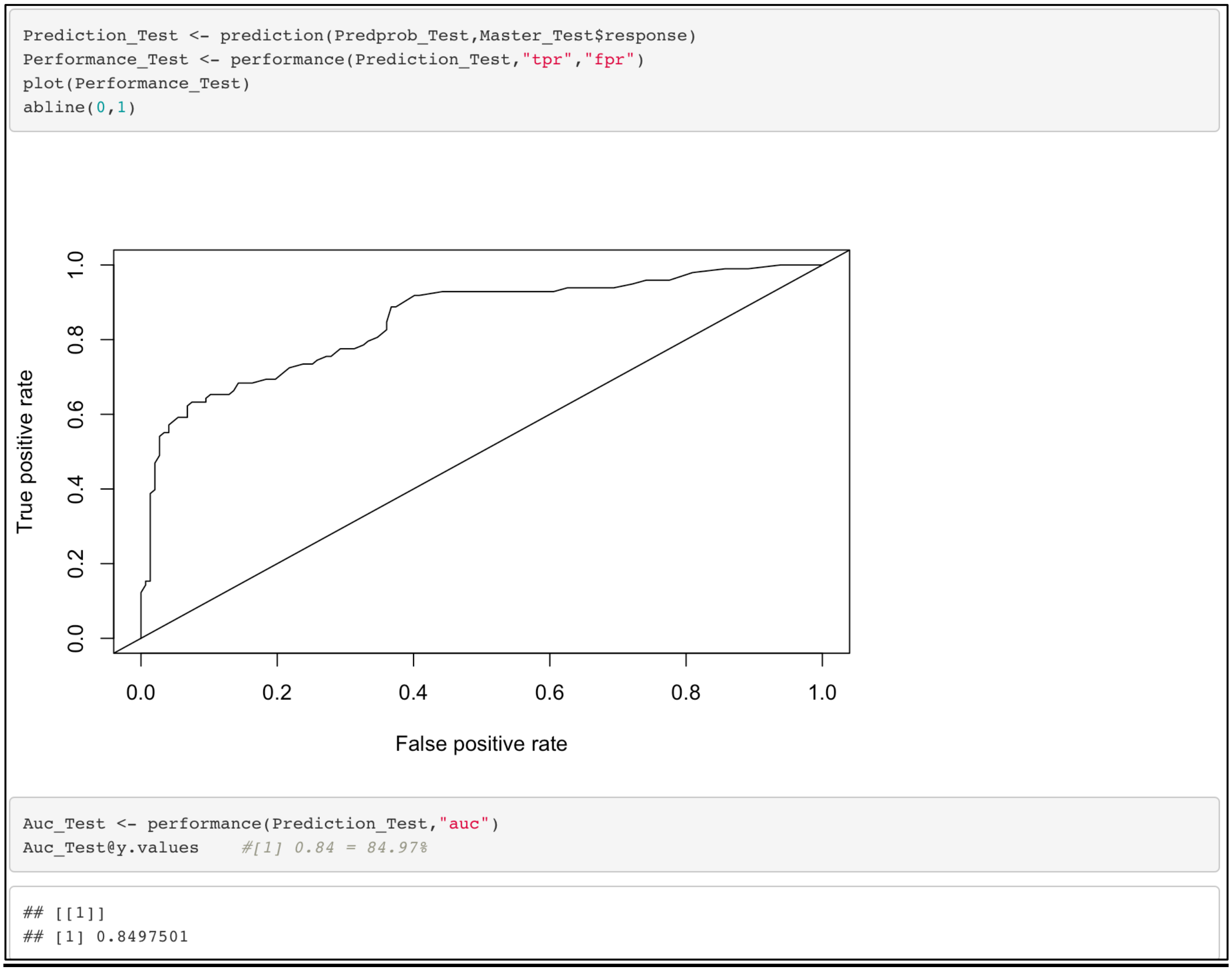
7. Validating our model with the help of the test data :

- We will compute relevant statistics (here AUC) for the test data and see if its value is close to the one of the train data.



```
#Validating the Model on the Test Data on Test data

Predprob_Test <- round(predict(Fullstep,Master_Test,type='response'),2)
head(Predprob_Test)
```

```
##   13   15   19   25   26   28
## 1.00 1.00 0.21 0.89 1.00 0.52
```



```
Prediction_Test <- prediction(Predprob_Test,Master_Test$response)
Performance_Test <- performance(Prediction_Test,"tpr","fpr")
plot(Performance_Test)
abline(0,1)
```

```
Auc_Test <- performance(Prediction_Test,"auc")
Auc_Test@y.values    #[1] 0.84 = 84.97%
```

```
## [[1]]
## [1] 0.8497501
```

- We, therefore, get an AUC value of 84.97% for our test data. Which is nearly the same as the AUC value for the master data, which is a good indicator of the quality of our model.

We now have a model that we're confident about, we can start using it to predict outcomes and derive relevant information about our data.

8. Future scope of the study:

- Having a validation dataset to further validate the fitness and robustness of our model.
- Using different classification models for our regression analysis and comparing it with our model to determine the best model for our dataset.
- Who to target first in the next planned campaign

9. References :

- (McGraw-Hill Irwin Series Operations and Decision Sciences) Michael H Kutner, Christopher J. Nachtsheim, John Neter, William Li - Applied Linear Statistical Models 5th Edition-McGraw-Hill Irwin (2004)
- https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5
- https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62
- https://www.statisticssolutions.com/binary-logistic-regression/
- https://www.dataversity.net/what-is-binary-logistic-regression-and-how-is-it-used-in-analysis/
- https://cran.r-project.org/web/packages/ROCR/ROCR.pdf
- https://www.rdocumentation.org/packages/caret/versions/6.0-92/topics/createDataPartition