**Modeling Test**

**Propensity Model for Up-Sell in Telecom Industry**

**Business Problem**: Company X in telecom domain has strong market share. The market share has reached a point where capturing new customers is quite expensive and has low ROI. Company X planned to focus on its existing customers for growth in its revenue. To capture growth, company decided to target its existing customers for higher value plans through an up-sell campaign. Since the company has a very huge database of existing customers, it is not possible to target each and every customer and hence marketing team proposed to develop a propensity model for identifying the customers whom they should target in this campaign.

Develop a model for identifying right set of customers with plan A to up-sell plan B.

Tasks to be performed:

1. Import data
2. Identify and develop Dependent variable
3. Prepare Uni-Variate and Bi-Variate Report
4. Prepare missing value report
5. Perform initial variable reduction and missing value imputation
6. Perform extreme value treatment
7. Prepare correlation matrix and VIF chart
8. Perform Multicollinearity check and variable reduction through Multicollinearity
9. Develop IV report
10. Perform Binning to prepare modeling dataset
11. Perform sampling to prepare training and validation dataset
12. Run the model
13. Develop report for model outcomes
14. Write the Scoring or implementation strategy

Solution:

1. Import data

Create working directory

setwd("C:/Users/babycorn/Documents/Edupristine/Telecom upsell Case")

Import the Raw data file into cust\_data file

cust\_data<-read.csv("Rawdatafile.csv")

### See the data summary (verify Data)

head(cust\_data)

tail(cust\_data)

1. Identify and develop Dependent variable

See for the variable which can be used to identify the historical responders and non-responders. This needs business understanding and data understanding. In the given case, flag for Plan change can be used to identify the customers who went for up-sell in past. Model building exercise will try to capture the characteristics of these customers based on other independent variables.

1. Prepare Uni-Variate and Bi-Variate Report

summary(cust\_data)

write.csv(summ\_cust\_data.csv, "summ\_cust\_data.csv.csv")

Check this file generated in the work folder and verify the summary details.

Identify the variables with missing value eg. Var1, Var27

Identify the cases with Extreme value eg. Var4

Identify the cases with very high or very low granularity eg. Var10









1. Prepare missing value report

The analysis from above report tells us that Var1 and Var27 will need missing value treatment. Further, the % of missing value is very high in Var27

% Missing = 4172/5000 = 83.44

As a rule, we generally drop variables with more than 50% missing values. Hence Var27 should be dropped out from the dataset.

For Var1 , % Missing = 25/5000 = 0.5% . We can impute this with either the mode value of the variable or any meaning value we can think of.

1. Perform initial variable reduction and missing value imputation

##Code to drop the variables not required

cust\_dat1<- cust\_data[,-c(12,29)]

View(cust\_dat1)

ifelse(cust\_dat1$Var1=="", "North",

ifelse(cust\_dat1$Var1=="South", "South",

ifelse(cust\_dat1$Var1=="Central", "Central", "North")))

1. Perform extreme value treatment

ifelse(cust\_dat1$Var4 >50, 50,

ifelse(cust\_dat1$Var4< 25, 20 , 30))

Verify summary again:

cust\_dat1$Var1<-ifelse(cust\_dat1$Var1=="", "North",

ifelse(cust\_dat1$Var1=="South", "South",

ifelse(cust\_dat1$Var1=="Central", "Central", "North")))

cust\_dat1$Var4<-ifelse(cust\_dat1$Var4 >50, 50,

ifelse(cust\_dat1$Var4< 25, 20 , 30))

summ\_cust\_data2.csv<-summary(cust\_dat1)

write.csv(summ\_cust\_data2.csv, "summ\_cust\_data2.csv")









1. Prepare correlation matrix and VIF chart

Corr\_data<- cust\_dat1[,-c(1,2,3,4,5,7,11)]

corr\_matrix.csv <- cor(Corr\_data,Corr\_data)

write.csv(corr\_matrix.csv, "corr\_matrix.csv")



cust\_dat1$Responder<-ifelse(cust\_dat1$Plan\_Chg\_Flag == "Yes", 1, 0)

head(cust\_dat1)

tail(cust\_dat1)

vif\_Cust\_data.csv <- vif(lm(Responder ~ Var4+ Var6+ Var7+ Var8+ Var11+ Var12+ Var13+ Var14+ Var15+ Var16+ Var17+ Var18+ Var19+ Var20+ Var21+ Var22+ Var23+ Var24+ Var25+ Var26

, data=cust\_dat1))

write.csv(vif\_Cust\_data.csv, "vif\_Cust\_data.csv")



1. Perform Multicollinearity check and variable reduction through Multicollinearity

Since the VIF for the all factors is less than 2 and correlation matrix has lower corr values, we can safely assume that the collinearity issue is not present.

1. Develop IV report using Bi-variate analysis and variable reduction technique:

Perform the frequency analysis of each category in the data against the response variable (dependent variable).

The final analysis should prepare a report highlighting the Cumulative IV values as below.

The following code generates and saves csv for bi-variate analysis:

##Remove the 2nd variable which was used to create responder variable

cust\_dat1<-cust\_dat1[,c(-2)]

## Set the data in the new datafile

dat<-cust\_dat1

###Verify the dataset

head(dat)

## create table to identify the var type

bi\_var<-apply(dat,2,typeof)

## add var name for the type and added the flag ..default set to 1 for all var

bi\_var<-data.frame(colnames(dat),bi\_var,flag=1)

## set the row names as numbers

row.names(bi\_var)<-1:nrow(bi\_var)

## set the column names

colnames(bi\_var)<-c("variable","var\_type","flag")

## get the position for variables to set the flag as 0

bi\_var$flag[which( bi\_var$variable %in% c("Cust\_id","Responder"))]<-0

## remove those with flag as 0

bi\_var<-bi\_var[bi\_var$flag==1,]

## created an object to get bi var analysis

event\_rate<-NULL

## loop in till all the var in the table

for ( i in 1:nrow(bi\_var))

{

## get the freq table for each var..ensure that the deleted var and numbers

## are in sequence..eg responder should be at 2nd position

aa<-as.matrix(table(dat[,i+2],dat[,27]))

cc<-aa

## append var name and the categories in that variable

bb<-cbind(rep(as.character(bi\_var$variable[i]),nrow(aa)),row.names(aa))

## merge the name, cat and freq table

aa<-data.frame(cbind(bb,aa))

## calc for ER, NER, WOE, IV and cum IV

aa[,5]<-as.numeric(cc[,1])/sum(as.numeric(cc[,1]))

aa[,6]<-as.numeric(cc[,2])/sum(as.numeric(cc[,2]))

aa[,7]<-log(aa[,5]/aa[,6])

aa[,8]<-(aa[,5]-aa[,6])\*aa[,7]

aa[,9]<-sum(aa[,8])

## append everything in new dataset

event\_rate<-rbind(event\_rate,aa)

}

## give the column names for data created above ..after the for loop

colnames(event\_rate)<-c("variable","Factor","Res","Non-Res","ER","NER","WOE","IV","Cum\_IV")

##Read the eventrate file and safe for analysis

head(event\_rate)

write.csv(event\_rate, "event\_rate\_IV.csv")



1. Perform Binning to prepare modeling dataset

cust\_dat1$GRPVar1<-ifelse(cust\_dat1$Var1=="North",1,ifelse(cust\_dat1$Var1=="South",2,3))

cust\_dat1$GRPVar2<-ifelse(cust\_dat1$Var2=="Low",1,ifelse(cust\_dat1$Var2=="Medium",2,3))

cust\_dat1$GRPVar3<-ifelse(cust\_dat1$Var3=="Unemployed",1,ifelse(cust\_dat1$Var3=="Govt",2,3))

cust\_dat1$GRPVar4<-ifelse(cust\_dat1$Var4 < 25,1,ifelse(cust\_dat1$Var4< 40,2,3))

cust\_dat1$GRPVar5<-ifelse(cust\_dat1$Var5=="Male",1,2)

cust\_dat1$GRPVar6<-ifelse(cust\_dat1$Var6 < 2,1,2)

cust\_dat1$GRPVar7<-ifelse(cust\_dat1$Var7 < 500,1,ifelse(cust\_dat1$Var7< 1000,2,3))

cust\_dat1$GRPVar8<-ifelse(cust\_dat1$Var8 < 13,1,2)

cust\_dat1$GRPVar9<-ifelse(cust\_dat1$Var9=="Cash",1,2)

cust\_dat1$GRPVar11<-ifelse(cust\_dat1$Var11 < 2,1,2)

cust\_dat1$GRPVar12<-ifelse(cust\_dat1$Var12 < 2,1,2)

cust\_dat1$GRPVar13<-ifelse(cust\_dat1$Var13 < 500,1,2)

cust\_dat1$GRPVar14<-ifelse(cust\_dat1$Var14 < 500,1,2)

cust\_dat1$GRPVar15<-ifelse(cust\_dat1$Var15 < 500,1,2)

cust\_dat1$GRPVar16<-ifelse(cust\_dat1$Var16 < 200,1,2)

cust\_dat1$GRPVar17<-ifelse(cust\_dat1$Var17 < 20,1,2)

cust\_dat1$GRPVar18<-ifelse(cust\_dat1$Var18 < 100,1,2)

cust\_dat1$GRPVar19<-ifelse(cust\_dat1$Var19 < 1000,1,2)

cust\_dat1$GRPVar20<-ifelse(cust\_dat1$Var20 < 2,1,2)

cust\_dat1$GRPVar21<-ifelse(cust\_dat1$Var21 < 2,1,2)

cust\_dat1$GRPVar22<-ifelse(cust\_dat1$Var22 < 500,1,2)

cust\_dat1$GRPVar23<-ifelse(cust\_dat1$Var23 < 500,1,2)

cust\_dat1$GRPVar24<-ifelse(cust\_dat1$Var24 < 50,1,2)

cust\_dat1$GRPVar25<-ifelse(cust\_dat1$Var25 < 100,1,2)

cust\_dat1$GRPVar26<-ifelse(cust\_dat1$Var26 < 30,1,2)

cust\_modeldata<-cust\_dat1[,c(28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53)]

write.csv(cust\_modeldata, "cust\_modeldata.csv")

1. Perform sampling to prepare training and validation dataset

## sorting the data

cust\_modeldata <- cust\_modeldata[order(cust\_modeldata$Responder,decreasing = TRUE),]

training\_data =strata(cust\_modeldata,c("Responder"),size=c(350,3150), method="srswor")

training\_data<-getdata(cust\_modeldata,training\_data)

View(training\_data)

1. Run the model

fit <- glm(Responder ~ as.factor(GRPVar2)+as.factor(GRPVar7)+as.factor(GRPVar8)+as.factor(GRPVar9)+as.factor(GRPVar11)+as.factor(GRPVar12)+as.factor(GRPVar1)+as.factor(GRPVar14)+as.factor(GRPVar15)+as.factor(GRPVar16)+as.factor(GRPVar17)+as.factor(GRPVar18)+as.factor(GRPVar19)+as.factor(GRPVar20)+as.factor(GRPVar21)+as.factor(GRPVar22)+as.factor(GRPVar23)+as.factor(GRPVar24)+ as.factor(GRPVar25), family = binomial("logit"),data=training\_data )

1. Develop report for model outcomes

summary(fit) # display results

Deviance Residuals:

Min 1Q Median 3Q Max

-1.4130 -0.4314 -0.2489 -0.0001 3.5287

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) -4.39793 0.26792 -16.415 < 2e-16 \*\*\*

as.factor(GRPVar2)2 0.13524 0.16422 0.823 0.410228

as.factor(GRPVar2)3 0.06255 0.16170 0.387 0.698897

as.factor(GRPVar7)2 0.23670 0.14187 1.668 0.095237 .

as.factor(GRPVar7)3 -17.19727 317.33159 -0.054 0.956781

as.factor(GRPVar8)2 0.31399 0.14478 2.169 0.030108 \*

as.factor(GRPVar9)2 0.30041 0.14080 2.134 0.032877 \*

as.factor(GRPVar11)2 -1.01584 0.14414 -7.047 1.82e-12 \*\*\*

as.factor(GRPVar12)2 -0.67231 0.14026 -4.793 1.64e-06 \*\*\*

as.factor(GRPVar1)2 -0.13777 0.16203 -0.850 0.395181

as.factor(GRPVar1)3 -0.14090 0.16094 -0.875 0.381333

as.factor(GRPVar14)2 0.49729 0.14443 3.443 0.000575 \*\*\*

as.factor(GRPVar15)2 0.40871 0.14483 2.822 0.004771 \*\*

as.factor(GRPVar16)2 0.32636 0.14490 2.252 0.024302 \*

as.factor(GRPVar17)2 0.34034 0.14502 2.347 0.018932 \*

as.factor(GRPVar18)2 0.42106 0.14363 2.932 0.003373 \*\*

as.factor(GRPVar19)2 0.35157 0.14416 2.439 0.014741 \*

as.factor(GRPVar20)2 0.55552 0.14578 3.811 0.000139 \*\*\*

as.factor(GRPVar21)2 0.38559 0.14343 2.688 0.007180 \*\*

as.factor(GRPVar22)2 0.36509 0.14402 2.535 0.011243 \*

as.factor(GRPVar23)2 0.24158 0.14353 1.683 0.092343 .

as.factor(GRPVar24)2 0.36435 0.14737 2.472 0.013424 \*

as.factor(GRPVar25)2 0.36496 0.14658 2.490 0.012781 \*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 2275.6 on 3499 degrees of freedom

Residual deviance: 1552.9 on 3477 degrees of freedom

AIC: 1598.9

Number of Fisher Scoring iterations: 18

summary\_pred\_model.csv<-predict(fit, type="response")

> write.csv(summary\_pred\_model.csv, "summary\_pred\_model.csv")

See the generated csv for predicted scores

**Model Fit Criteria:**

1. Use the Deviance or Hosmer & Lemeshow test statistics to check the validity of the model. Higher the “P” value better is the model. Proceed to next steps only if we have higher value of P.

##############################################################################

### Hosmer lemeshow goodness of fit test

hosmerlem <- function (y, yhat, g = 12)

{

cutyhat <- cut(yhat, breaks = quantile(yhat, probs = seq(0,1, 1/g)), include.lowest = T)

obs <- xtabs(cbind(1 - y, y) ~ cutyhat)

expect <- xtabs(cbind(1 - yhat, yhat) ~ cutyhat)

chisq <- sum((obs - expect)^2/expect)

P <- 1 - pchisq(chisq, g - 2)

c("X^2" = chisq, Df = g - 2, "P(>Chi)" = P)

}

##Run the above function after setting the values

R\_hat<-as.vector(fitted(**fit**))

yhat<-R\_hat

y<-**training\_data**$Responder

hosmerlem(y, yhat)

2. Test the null hypothesis for the independent variables, i.e. all β = 0. P value should be significant (i.e. p < 0.05) to reject the null hypothesis and prove that β values are not equal to 0.

3. Check the concordance and Tie. The rule of thumb test is (Concordance+ ½ Tie) should be greater than 60%.

4. Check the significance of the estimates of each of the variable. If any of the estimates are not significant, variable with highest P value will be dropped and steps i to vi are repeated with the new set of variables. This process will continue until all the variables in the model have significant estimates.

5. Frame the equation with the significant variables. Odds ratio and probability value for each of the profile is estimated.

6. Specificity and Sensitivity of the model is assessed and ROC (Receiving Operating Characteristic) graph is plotted. Area under the ROC is an indication of how well the classification of good in to good and bad to bad is decided by the identified model.

7. Coefficient Stability: Coefficient stability is checked across development and validation sample. Once the model is performing satisfactorily on development sample, we use the same set of variables to model the validation sample. A robust model should perform equally well on validation sample too. Hence, the coefficients should be in a close range and should be of same sign.

8. Concordance: Consider a set of 100 individuals out of which 10 are the responders (denoted by 1) and 90 are non-responders (denoted by 0). Now we construct pairs for each responder with every non-responder. Hence, we get 900 such pairs (10\*90 = 900). Using the model under development, we calculate the predicted response rate for each responder and non-responder in every pair. If responder’s predicted probability is greater than non-responder’s predicted probability, then the pair is concordant. If it is vice versa, then the pair is discordant and if both are equal, then the pair is tied. For a good model, the percent concordant pair lies above 65%.

See the results for concordance test below:

##############################################################

outcome\_and\_fitted\_col<-data.frame(training\_data$Responder, R\_hat)

> colnames(outcome\_and\_fitted\_col)<-c("Responder","fitted.values")

> Concordance = function(outcome\_and\_fitted\_col)

+

+ {

+

+ #outcome\_and\_fitted\_col = cbind(logistic1$Responder, logistic1$fitted.values)

+ # get a subset of outcomes where the event actually happened

+

+ ones = outcome\_and\_fitted\_col[outcome\_and\_fitted\_col[,1] == 1,]

+

+ # get a subset of outcomes where the event didn't actually happen

+

+ zeros = outcome\_and\_fitted\_col[outcome\_and\_fitted\_col[,1] == 0,]

+

+ # Equate the length of the event and non-event tables

+

+ if (length(ones[,1])>length(zeros[,1])) {ones = ones[1:length(zeros[,1]),]}

+

+ else {zeros = zeros[1:length(ones[,1]),]}

+

+ # Following will be c(ones\_outcome, ones\_fitted, zeros\_outcome, zeros\_fitted)

+

+ ones\_and\_zeros = data.frame(ones, zeros)

+

+ # initiate columns to store concordant, discordant, and tie pair evaluations

+

+ conc = rep(NA, length(ones\_and\_zeros[,1]))

+ disc = rep(NA, length(ones\_and\_zeros[,1]))

+ ties = rep(NA, length(ones\_and\_zeros[,1]))

+

+ for (i in 1:length(ones\_and\_zeros[,1])) {

+

+ # This tests for concordance

+

+ if (ones\_and\_zeros[i,2] > ones\_and\_zeros[i,4])

+

+ {conc[i] = 1

+ disc[i] = 0

+ ties[i] = 0}

+

+ # This tests for a tie

+

+ else if (ones\_and\_zeros[i,2] == ones\_and\_zeros[i,4])

+

+ {

+

+ conc[i] = 0

+ disc[i] = 0

+ ties[i] = 1

+

+ }

+

+ # This should catch discordant pairs.

+

+ else if (ones\_and\_zeros[i,2] < ones\_and\_zeros[i,4])

+

+ {

+

+ conc[i] = 0

+ disc[i] = 1

+ ties[i] = 0

+

+ }

+

+ }

+

+ # Here we save the various rates

+

+ conc\_rate = mean(conc, na.rm=TRUE)

+ disc\_rate = mean(disc, na.rm=TRUE)

+ tie\_rate = mean(ties, na.rm=TRUE)

+

+ return(list(concordance=conc\_rate, num\_concordant=sum(conc), discordance=disc\_rate, num\_discordant=sum(disc), tie\_rate=tie\_rate,num\_tied=sum(ties)))

+

+ }

>

> Concordance\_test<-Concordance(outcome\_and\_fitted\_col)

> Concordance\_test

$concordance

[1] 0.8571429

$num\_concordant

[1] 300

$discordance

[1] 0.1428571

$num\_discordant

[1] 50

$tie\_rate

[1] 0

$num\_tied

[1] 0

9. Gini Coefficient: The Gini coefficient is one which is used to test the model accuracy. It is calculated by using following formula. For good model the Gini coefficient should be in the range of 40-60%.

Gini=2C-1 Where C= Area under the curve (ie Concordance+1/2 of Tie)

10. Scoring: Satisfaction of the model comes when the model is doing well in terms of rank ordering, coefficient stability, Goodness of fit, Concordance and capturing both on development and validation samples.

Now, take the coefficients of variables obtained from a model run on development sample and use it to predict response rate of validation sample. This method is known as scoring of the model. Scoring provides a good idea about how the model will perform when applied to another data set. Here, we are concerned about the capturing of the responders, say in first 40 % of the population.

The model is used to predict the response rate for a set of new data is taken from a different time frame to test the validity of the rules suggested by the model. The model will be applicable to the profiles similar to the once already present in the sample data used for model development. Model validation is performed by taking the optimum threshold level of probability.

Lift/gains chart for model case:

#### Preparing Gains/Lift chart

library(ROCR)

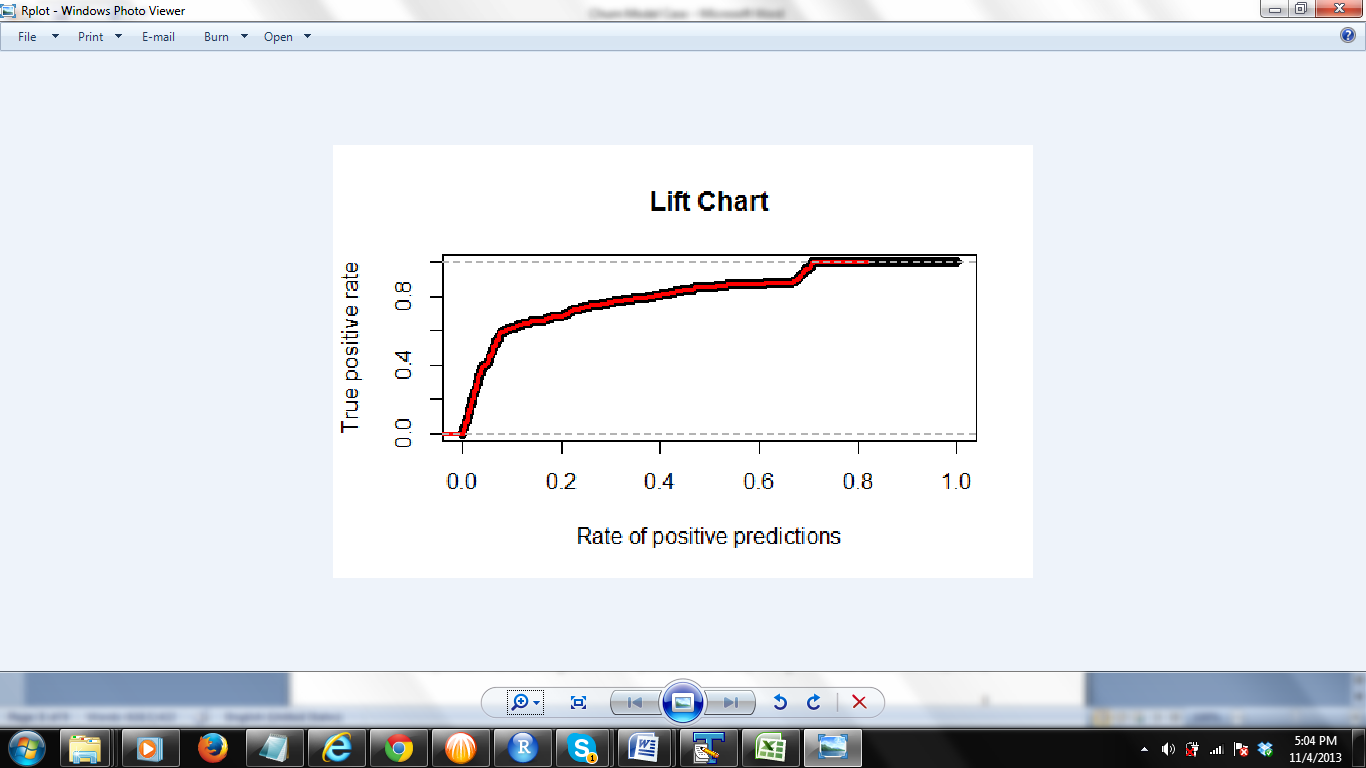
gain.chart <- function(y\_hat,y) {

plot(performance(prediction(y\_hat,y), "tpr", "rpp"),lwd = 7, main = "Lift Chart")

lines(ecdf((rank(-y\_hat)[y == T]) / length(y)),verticals = T, do.points = F, col = "red", lwd = 3)

}

gain.chart(R\_hat,training\_data$Responder)



The model is implemented and refreshed periodically to generate scores for the customers.

1. Write the Scoring or implementation strategy

The model can be used to score the existing customers to score for up-sell.

The implementation requires extracting the data of the significant variables (highlighted) and then grouping it as per model equation and generating the scores. The score are then sorted in descending order and the top few deciles will be used for up-sell marketing.