**Kaggle example 1**

#Machine Learning model to identify bad car purchases

newdata<-read.csv(file.choose())

newdata

head(newdata)

summary(newdata)

#columns PRIMEUNIT And AUCGUART are almost completely null they can be completely removedcolnam

colnames<-(names(newdata) %in% c("PRIMEUNIT","AUCGUART"))

newdata=newdata[,!(names(newdata) %in% c("PRIMEUNIT","AUCGUART"))]

names(newdata)

summary(newdata)

newdata<-na.omit(newdata)

newdata<-newdata[newdata==NULL,]

#exploringthe data to understand what drives bad buys

library(ggplot2)

ggplot(numdata,aes(x=newdata$Make,y=as.factor(newdata$IsBadBuy),fill=as.factor(newdata$IsBadBuy)))+geom\_bar(stat="identity")

ggplot(newdata,aes(x=newdata$SubModel,y=as.factor(newdata$IsBadBuy),fill=as.factor(newdata$IsBadBuy)))+geom\_bar(stat="identity")

ggplot(newdata,aes(x=newdata$Color,y=as.factor(newdata$IsBadBuy),fill=as.factor(newdata$IsBadBuy)))+geom\_bar(stat="identity")

ggplot(newdata,aes(x=newdata$Transmission,y=as.factor(newdata$IsBadBuy),fill=as.factor(newdata$IsBadBuy)))+geom\_bar(stat="identity")

ggplot(newdata,aes(x=newdata$WheelType,y=as.factor(newdata$IsBadBuy),fill=as.factor(newdata$IsBadBuy)))+geom\_bar(stat="identity")

ggplot(newdata,aes(x=newdata$Nationality,y=as.factor(newdata$IsBadBuy),fill=as.factor(newdata$IsBadBuy)))+geom\_bar(stat="identity")

#treat remaining missing values with medians

attach(newdata)

newdata[is.na(MMRAcquisitionAuctionAveragePrice)]<-median(as.numeric(newdata$MMRAcquisitionAuctionAveragePrice),na.rm=TRUE)

newdata[is.na(MMRAcquisitionRetailAveragePrice)]<-median(as.numeric(newdata$MMRAcquisitionRetailAveragePrice),na.rm=TRUE)

newdata[is.na(MMRAcquisitonRetailCleanPrice)]<-median(as.numeric(MMRAcquisitonRetailCleanPrice),na.rm=TRUE)

newdata[is.na(MMRCurrentAuctionAveragePrice)]<-median(as.numeric(MMRCurrentAuctionAveragePrice),na.rm=TRUE)

newdata[is.na(MMRCurrentRetailCleanPrice)]<-median(as.numeric(newdata$MMRCurrentRetailCleanPrice),na.rm=TRUE)

#exploring the data to understand which are the cars which are faulty

with(newdata,table( IsBadBuy,Size ))

with(newdata,table( IsBadBuy,VehicleAge ))

with(newdata,table( IsBadBuy,Make ))

with(newdata,table(IsBadBuy,Size))

#Remove Remaining null values

newdata<-na.omit(newdata)

summary(newdata)

names(newdata)

attach(newdata)

library(randomForest)

#split the data into training and test set

d<-sample(nrow(newdata),0.7\*nrow(newdata))

train<-newdata[d,]

test<-newdata[-d,]

#check for data balance

table(newdata$IsBadBuy)

#Data is heavily in terms of majority cases being of a good buy this can be handled with wieights in the random forest algorithm

attach(newdata)

fit<-randomForest(as.factor(IsBadBuy)~Make+Transmission+VehOdo+as.numeric(MMRAcquisitionAuctionAveragePrice)+as.numeric(MMRAcquisitionAuctionCleanPrice)+as.numeric(MMRAcquisitionRetailAveragePrice)+as.numeric(MMRAcquisitonRetailCleanPrice)+as.numeric(MMRCurrentAuctionAveragePrice)+as.numeric(MMRCurrentAuctionCleanPrice)+as.numeric(MMRCurrentRetailAveragePrice)+as.numeric(MMRCurrentRetailCleanPrice)+VehBCost+IsOnlineSale+WarrantyCost,data=train,Classwt=c(0.3,0.7))

fit$importance

attach(train)

#Predict on test data and check for accuracy

pred=predict(fit,test)

table(pred,test$IsBadBuy)

install.packages("pROC")

plot(roc(pred,test$IsBadBuy, direction="<"),col="yellow", lwd=3, main="ROC curve")

simp\_roc <- roc(pred,test$IsBadBuy)

auc(simp\_roc)

Area under the curve: 0.8638

#check for correlcations

numdata<-as.numeric(MMRAcquisitionAuctionAveragePrice)+as.numeric(MMRAcquisitionAuctionCleanPrice)+as.numeric(MMRAcquisitionRetailAveragePrice)+as.numeric(MMRAcquisitonRetailCleanPrice)+as.numeric(MMRCurrentAuctionAveragePrice)+as.numeric(MMRCurrentAuctionCleanPrice)+as.numeric(MMRCurrentRetailAveragePrice)+as.numeric(MMRCurrentRetailCleanPrice)

numdata<-data.frame(as.numeric(MMRAcquisitionAuctionAveragePrice),as.numeric(MMRAcquisitionAuctionCleanPrice),as.numeric(MMRAcquisitionRetailAveragePrice),as.numeric(MMRAcquisitonRetailCleanPrice),as.numeric(MMRCurrentAuctionAveragePrice),as.numeric(MMRCurrentAuctionCleanPrice),as.numeric(MMRCurrentRetailAveragePrice),as.numeric(MMRCurrentRetailCleanPrice))

cor(numdata)

#drop correlated variables and check for accuracy

fit2<-randomForest(as.factor(IsBadBuy)~Make+Transmission+VehOdo+as.numeric(MMRCurrentRetailCleanPrice)+VehBCost+IsOnlineSale+WarrantyCost,data=train,Classwt=c(0.7,0.5))

pred2=predict(fit2,test)

simp\_roc <- roc(pred2,test$IsBadBuy)

auc(simp\_roc)

Area under the curve: 0.7113

plot(roc(pred2,test$IsBadBuy, direction="<"),col="green", lwd=3, main="ROC curve")

#Mckinsey Hackothon : Loan Propensity Modelling

#Load the training data

loandata<-read.csv(file.choose(),na.strings = c("", "NA"))

#Lets explore the data

summary(loandata)

#Look at the structure of the dataset

str(loandata)

#Check each value of missing variable

summary(loandata$City\_Category)

summary(loandata$City\_Category)

#City category has 814 missing values, we can replace them with the most frequently occuring city category "A"

loandata$City\_Category[which(is.na(loandata$City\_Category))]<-"A"

summary(loandata$Employer\_Category1)

# Employer category1 has 4018 missing value, Replacing this with most frequently occuring employee cateogory "A"

loandata$Employer\_Category1[which(is.na(loandata$Employer\_Category1))]<-"A"

#employer category 2 is shown as type of numeric ,it must be converted to a factor and its missing values also has to be treated

loandata$Employer\_Category2<-as.factor(loandata$Employer\_Category2)

summary(loandata$Employer\_Category2)

#employer category 2 has 4298 missing values which can be replaced by the most frequently occuring category =4

loandata$Employer\_Category2[which(is.na(loandata$Employer\_Category2))]<-4

#Primary\_bank\_type has 9391 missing values which needs to be treated as well with the most frequently occuring Primary bank type P

loandata$Primary\_Bank\_Type[which(is.na(loandata$Primary\_Bank\_Type))]<-"P"

#Replace missing values in existing EMI with the median values

loandata$Existing\_EMI[which(is.na(loandata$Existing\_EMI))]<-median(loandata$Existing\_EMI,na.rm=TRUE)

#Replace missing values in Loan Amount with median values

loandata$Loan\_Amount[which(is.na(loandata$Loan\_Amount))]<-median(loandata$Loan\_Amount,na.rm=TRUE)

#Replace missing values in Loan Period with median values

loandata$Loan\_Period[which(is.na(loandata$Loan\_Period))]<-median(loandata$Loan\_Period,na.rm=TRUE)

#Replace missing values interest rates with median values

loandata$Interest\_Rate[which(is.na(loandata$Interest\_Rate))]<-median(loandata$Interest\_Rate,na.rm=TRUE)

#Replace missing values EMI with median values

loandata$EMI[which(is.na(loandata$EMI))]<-median(loandata$EMI,na.rm=TRUE)

summary(loandata)

#We will not use variables which denote ids and codes, selecting the other variables which will be used in our modelling

frame<-with(loandata,data.frame(City\_Category,Employer\_Category1,Employer\_Category2,Monthly\_Income,Primary\_Bank\_Type,Contacted,Existing\_EMI,Loan\_Amount,Loan\_Period,Interest\_Rate,EMI,Approved))

summary(frame)

#Feature Engineering using Random Forest

attach(frame)

library(randomForest)

model\_rf<-randomForest(as.factor(Approved)~.,data=frame)

summary(model\_rf)

model\_rf$importance

#Feature engineering using logistic regression

model\_glm <- glm(as.factor(Approved)~.,data=frame,family="binomial")

#Check the significant variables based on p values

summary(model\_glm)

#From both the feature engineering methods above we choose the following independent vairbales

#Monthly\_Income

#Existing\_EMI

##Loan\_Amount

#Loan\_Period

#EMI

#selecting the final list of variables in a finaldataframe

finalframe<-with(frame,data.frame(Monthly\_Income,Existing\_EMI,Loan\_Amount,Loan\_Period,EMI,Approved))

#check for correlations in training set

cor(train)

#EMI looks to be highky correlated with Loan amount, so we will drop EMI

finalframe<-with(frame,data.frame(Monthly\_Income,Existing\_EMI,Loan\_Amount,Loan\_Period,Approved))

#check data balance

table(finalframe$Approved)

#data is heavily unbalanced in favour of 0 or no approval

#diving the final frame into test and train

d=sample(nrow(finalframe),nrow(finalframe)\*0.70)

train=finalframe[d,]

test=finalframe[-d,]

#Fitting a random forest model with class weights 50% to both cases

rf\_fit=with(finalframe,randomForest(as.factor(Approved)~Monthly\_Income+Existing\_EMI+Loan\_Amount+Loan\_Period,data=finalframe,classwt=c(0.5,0.5)))

table(predict(rf\_fit,train))

table(predict(rf\_fit,test))

table(test$Approved,predict(rf\_fit,test))

predictions<-predict(rf\_fit,test)

install.packages("pROC")

library(pROC)

auc(test$Approved,as.numeric(predictions))

#Area under the curve is almost 0.94

#We will try to see if we can use other algorithms to improve accuracy

#We will try by using oversampling the minority class. We had 1020 minority class observations out of a total of 69713 observations. We will oversample the minority class till we have 69713 observations

install.packages("ROSE")

library(ROSE)

data\_balanced\_over <- ovun.sample(as.factor(Approved)~Monthly\_Income+Existing\_EMI+Loan\_Amount+Loan\_Period, data = finalframe, method = "over",N = 69713)$data

#Using Naive bayes to train and predict

library(naivebayes)

model\_nb=naiveBayes(as.factor(Approved)~Monthly\_Income+Existing\_EMI+Loan\_Amount+Loan\_Period,data=data\_balanced\_over)

predictions<-predict(model\_nb,test)

table(test$Approved,predict(model\_nb,test))

predictions<-predict(model\_nb,test)

library(pROC)

auc(test$Approved,as.numeric(predictions))

#Naive baiyes Area under curve 0.54

#using svm to train and predict

#Using support vector to train and predict

library(e1071)

model\_svm=svm(as.factor(Approved)~Monthly\_Income+Existing\_EMI+Loan\_Amount+Loan\_Period,data=data\_balanced\_over)

predictions<-predict(model\_svm,test)

table(test$Approved,predict(model\_svm,test))

predictions<-predict(model\_svm,test)

library(pROC)

auc(test$Approved,as.numeric(predictions))

#svm area under curve around 0.5

#From the above 3 models we choose random forest which has the highest Area under curve to score the final results

#Loading the final scoring set

loandata\_score<-read.csv(file.choose(),na.strings = c("", "NA"))

#selecting the relevant columns in the scoring set

#selecting the final list of variables in a finaldataframe

summary(loandata\_score)

str(test)

str(predictions)

finalscore<-with(loandata\_score,data.frame(Monthly\_Income,Existing\_EMI,Loan\_Amount,Loan\_Period,EMI))

names(finalscore)

#some independent variables have missing values in few observations which cannot generate predictions,such cases will be treated with medians

#Replace missing values in existing EMI with the median values

finalscore$Existing\_EMI[which(is.na(finalscoredata$Existing\_EMI))]<-median(finalscoredata$Existing\_EMI,na.rm=TRUE)

#Replace missing values in Loan Amount with median values

finalscore$Loan\_Amount[which(is.na(finalscoredata$Loan\_Amount))]<-median(finalscoredata$Loan\_Amount,na.rm=TRUE)

#Replace missing values in Loan Period with median values

finalscore$Loan\_Period[which(is.na(finalscoredata$Loan\_Period))]<-median(finalscoredata$Loan\_Period,na.rm=TRUE)

#Replace missing values interest rates with median values

finalscore$Interest\_Rate[which(is.na(finalscoredata$Interest\_Rate))]<-median(finalscoredata$Interest\_Rate,na.rm=TRUE)

#Replace missing values EMI with median values

finalscore$EMI[which(is.na(finalscoredata$EMI))]<-median(finalscoredata$EMI,na.rm=TRUE)

finalscore$Monthly\_Income[which(is.na(finalscore$Monthly\_Income))]<-median(finalscore$Monthly\_Income,na.rm=TRUE)

summary(finalscore)

#generate finalpredictions on the scoring set using the random forests method

Approved<-predict(rf\_fit,finalscore)

finalscore=data.frame(finalscore,Approved)

summary(finalscore)

#Load the final predictions into a Csv file

write.csv(data.frame(loandata\_score$ID,finalscore$Approved),"C:/Users/genus/Documents/kaggle/Loan/Finalsubmission.csv")

#Campaign Propensity Modelling

This model was based on sample data generated for cross channel modellng

#Read campaign Data,

campdata<-read.csv(file.choose())

#Read social data

socialdata<-read.csv(file.choose())

# Read internet data

website<-read.csv(file.choose())

#Read Demography data

dem<-read.csv(file.choose())

names(campdata)

names(dem)

names(website)

names(socialdata)

#Merge demography data and campaign data,website data and social data

mydata<-merge(campdata,dem)

mydata<-merge(mydata,socialdata)

mydata<-merge(mydata,website)

summary(mydata)

names(mydata)

#select numerical variables

attach(mydata)

numdata=data.frame(EMAIL\_RESPONSE\_FLAG,IncomeLevel,NumberofChildren,CreditCards,AverageInternetUseperday,Carownership,NumberofFollowers,retweets,TOTAL\_PV,GOOGLE\_AD\_CLICKS,COUPON\_PORTAL\_CLICKS,EMAIL\_SIGN\_UPS,REGISTRY\_SIGN\_UPS,CLUB\_WEDD\_REGISTRY\_SIGN\_UPS,BABY\_REGISTRY\_SIGN\_UPS,TARGET\_LISTS\_REGISTRY\_SIGN\_UPS,WEEKLY\_AD\_VIEWS,

CAT\_SPORTS\_PV)

str(numdata)

#scale numerical variables

numdata<-scale(numdata,center=TRUE,scale=TRUE)

library(randomForest)

fit<-randomForest(as.factor(EMAIL\_RESPONSE\_FLAG)~NumberofChildren+CreditCards+AverageInternetUseperday+Carownership+NumberofFollowers+retweets+TOTAL\_PV+GOOGLE\_AD\_CLICKS+COUPON\_PORTAL\_CLICKS+EMAIL\_SIGN\_UPS+REGISTRY\_SIGN\_UPS+CLUB\_WEDD\_REGISTRY\_SIGN\_UPS+BABY\_REGISTRY\_SIGN\_UPS+TARGET\_LISTS\_REGISTRY\_SIGN\_UPS+WEEKLY\_AD\_VIEWS+

CAT\_SPORTS\_PVdata=numdata)

fit$importance

names(fitdata)=c("Variable","importance")

d<-sample(nrow(mydata),0.8\*nrow(mydata))

train<-mydata[d,]

test<-mydata[-d,]

library(randomForest)

fit<-randomForest(as.factor(EMAIL\_RESPONSE\_FLAG)~NumberofFollowers+retweets+TOTAL\_PV+GOOGLE\_AD\_CLICKS +CAT\_SPORTS\_PV,classweight=c(0.1,0.9),data=train)

#predict on train data

pred<-predict(fit,train)

table(train$EMAIL\_RESPONSE\_FLAG,pred)

#predict on test data

pred<-predict(fit,test)

table(test$EMAIL\_RESPONSE\_FLAG,pred)

library(pROC)

plot(roc(pred,test$EMAIL\_RESPONSE\_FLAG,direction="<"),col="yellow", lwd=3, main="ROC curve")

simp\_roc <- roc(pred,test$IsBadBuy)

auc(simp\_roc)

#check the datatype of each column

str(mydata)

#Lets look at the segments of customers that exists in the data

names(mydata)