Case Study 2

Logistic Regression, Random Forest, and SVM Application to a Real World Data Set

Submitted by

Md Salman Rahman

Graduate Student

University of Texas Rio Grande Valley

Submitted to

Dr. Tamer Oraby

November 28, 2021

Contents

1 Data Preparation **2**

1.1 Descriptive Analysis 2

1.2 Missing Value Checking 2

1.3 Training and Testing Data 2

2 Information Value (IV) **2**

2.1 Multicollinearity 3

3 Classification **3**

3.1 Logistic Regression (LR) 4

3.2 Random Forest (RF) 6

3.3 Support Vector Machine (SVM) 8

4 Comparing Logistic Regression(LR), Random Forest (RF), SVM(Polynomial Kernel, and SVM(Gaussian Radial Kernel) **11**

5 Conclusion **13**

6 Data and Code **13**

1 Data Preparation

1.1 Descriptive Analysis

This data set is about marketing example from the insurance industry. The data contains information on customer responses to a historical direct mail marketing campaign. The goal is to find the best possible target audience going forward. The data set contains 68 predictive variables and 20k records. In the R code, the histogram, pie chart, and correlation analysis is done to find the information about the given data-sets.

1.2 Missing Value Checking

There are no missing value in the training and validation data set.

1.3 Training and Testing Data

We split our data set into 10k records for training 10k records for testing. Here the model success is depends on predicting whether customer will take the purchase offer based on validation data-set. We will use ROC curve to evaluate the performance of the model. We will use same set of variables for each model and the variables were selected based on section 2.

2 Information Value (IV)

Information value depicts which columns in a particular data set have high predictive power on the value of dependent variable. IV provides a great framework for explanatory analysis and variable screening for binary classifiers. Also, Information value widely use to access credit risk for a long time.

Data Type	**Sources**
< 0.02	Useless
$0.02 - 0.1$	Weak
$0.1 - 0.3$	Medium
$0.3 - 0.5$	Strong
> 0.5	Suspiciously good; too good to be true

Here, we will remove all the variable with an information value(IV) less than 0.05. I use R “Information” package to calculate information values. By removing the useless variable, we define a new training and validation set which consist of 34 predictive variables and 20k records.

2.1 Multicollinearity

I use R “ClustOfVar” variable clustering library to eliminate highly correlated variable from the training and validation data-set. In this process, I generate 20 cluster and picked the variable with highest Information Value within each class. In this process, I prepare 20 variables as the inputs for all the three classification methods. Also, a new response variable called NewPurchase created from the PURCHASE variable in the training data. If PURCHASE=1, I make NewPurchase=1, otherwise NewPurchase=-1 using ifelse command in R.

3 Classification

I will discuss three commonly use classification model called logistic regression, random forest, and support vector machine. Also, We will see their application on marketing data-set.

3.1 Logistic Regression (LR)

In linear regression based classification task, if the output is close to zero then we predict a negative probability of the output; if the output become very large, then linear regression gives a value bigger than 1. So this is problem, especially fitting a straight line to a binary response that is coded as 0 and 1, in principle we can always predict probability of input $p(X) < 0$ for some values and $p(X) > 1$ for others unless the range of input variable is limited.
Logistic regression tackle this problem using logistic function in which the output of the model is always between 0 and 1. Logistic function is given by,

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$$

We have to use multiple logistic regression for our data-set as for our problem there are multiple predictors. For multiple predictors logistic functions becomes

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

The above equation can be rewritten as

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_p X_p}}$$

We will use maximum likelihood method to estimate $\beta_0, \beta_1, \cdots, \beta_p$. Step by step procedure to deploy logistic regression model is discussed below.

- **Step 1:** I use R glm function to build logistic regression model. I use validation data set for making prediction for logistic regression model.
- **Step 2:** Summary and coefficient of the model is calculated and given in the R code.
- **Step 3:** For logistic regression model, I calculate confusion matrix, sensitivity and specificity . Also, I calculate the accuracy which means the percentage of

all correct predictions that were made by the model.

The confusion matrix is given below:

		Reference		
		(−1)	(1)	**total**
	(−1)	TP (7787)	FP (1598)	P′
Predict outcome	(1)	FN (236)	TN (379)	N′
	total	P	N	

TP = True Positive, TN = True Negative,

FP = False Positive, and FN = False Negative.

For logistic regression model the sensitivity is 0.9706, and specificity is 0.1917, and accuracy of the model is 0.8166. Here, sensitivity indicate the proportion of the customers that were predicted to take the offer out of those who actually took the offer and specificity indicates the proportion of customers that were predicted not to take the offer out of those who actually did not take the offer. And, the accuracy is the percentage of all correct predictions that were made by the logistic regression model.

- **Step 4:** Finally, I create the ROC curve and evaluate the AUC value. The AUC (Area under the curve) is 0.5811 for the logistic regression model and the ROC plot is given in Figure 1.

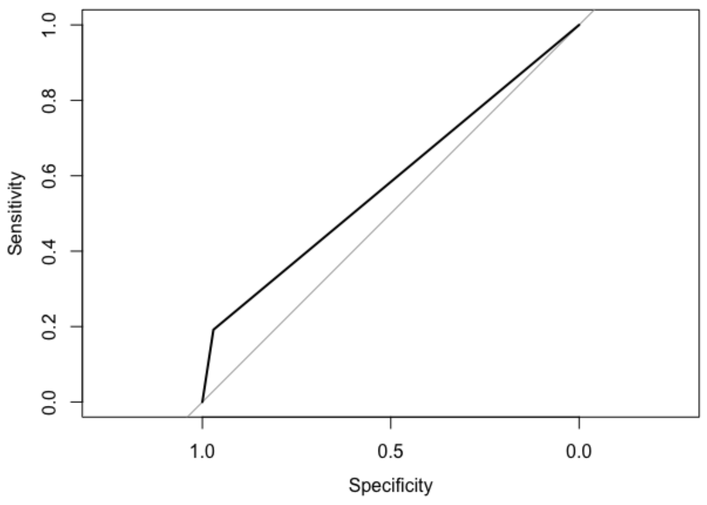


Figure 1: ROC Plot for Logistic Regression

3.2 Random Forest (RF)

Ensemble learning such as bagging methods get success for reducing the variance of an essential prediction function. Bagging especially works better for trees where variance is high and bias is low. Ranfom forest is modification of the bagging technique that builds a large collection of de-correlated trees. The principle difference between bagging and random forest is the choice of predictor of subset size say m. For example, if for random forest model m=p then the model is simply to bagging. In this case study, to deploy random forest I use randomForest library and use the set.seed(101) to make the result reproducible. Here step by step procedure of implementing random forest is explained:

- **Step 1:** I build a random forest model using 10001 trees using the training data. I try different mtry (number of variables randomly sampled as candidates at each split) values varying from 1 to 13 and calculate the Out of Bag(OOB) error for each model.

- **Step 2:** I consider the lowest OOB error as the best model and mtry = 9

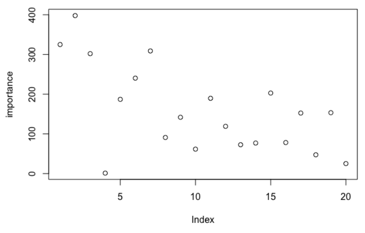
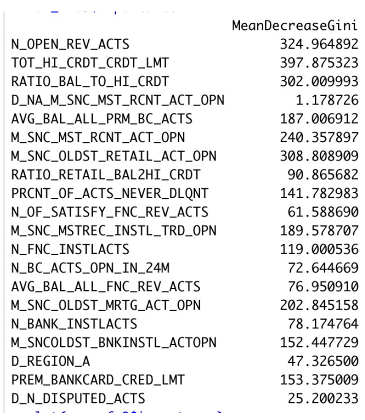


Figure 2: Variable Importance Plot



```
                               MeanDecreaseGini
N_OPEN_REV_ACTS                      324.964892
TOT_HI_CRDT_CRDT_LMT                 397.875323
RATIO_BAL_TO_HI_CRDT                 302.009993
D_NA_M_SNC_MST_RCNT_ACT_OPN            1.178726
AVG_BAL_ALL_PRM_BC_ACTS              187.006912
M_SNC_MST_RCNT_ACT_OPN               240.357897
M_SNC_OLDST_RETAIL_ACT_OPN           308.808909
RATIO_RETAIL_BAL2HI_CRDT              90.865682
PRCNT_OF_ACTS_NEVER_DLQNT            141.782983
N_OF_SATISFY_FNC_REV_ACTS             61.588690
M_SNC_MSTREC_INSTL_TRD_OPN           189.578707
N_FNC_INSTLACTS                      119.000536
N_BC_ACTS_OPN_IN_24M                  72.644669
AVG_BAL_ALL_FNC_REV_ACTS              76.950910
M_SNC_OLDST_MRTG_ACT_OPN             202.845158
N_BANK_INSTLACTS                      78.174764
M_SNCOLDST_BNKINSTL_ACTOPN           152.447729
D_REGION_A                            47.326500
PREM_BANKCARD_CRED_LMT               153.375009
D_N_DISPUTED_ACTS                     25.200233
```

Figure 3: Variable Vs Mean Decrease Gini

gives the lowest Out of Bag(OOB) error and it is consider as the best model to create a variable importance plot to decide on the importance of the variable. Figure 2 shows the variable importance plot and we see that second variable is the most important variable in the model. Figure 3 shows the corresponding importance value of each variable. In figure 2 the Y axis indicate mean decrease Gini value for each variable.

- **Step 3:** The above selected best model is used to make prediction using the validation data.

- **Step 4:** I calculate confusion matrix, sensitivity which means the proportion of customers that were predicted to take the offer out of those who actually took the offer and specificity which means the proportion of customers that were predicted not to take the offer out of those who actually did not take the offer. Additionally, I calculate the accuracy which means the percentage of all correct predictions that were made by the model.

  The confusion matrix for random forest is given below:

Reference

		(-1)	(1)	**total**
	(-1)	TP (7721)	FP (1457)	P′
Predict outcome	(1)	FN (302)	TN (520)	N′
	total	P	N	

TP = True Positive, TN = True Negative,

FP = False Positive, and FN = False Negative.

- **Step 5:** Finally, I create the ROC curve and evaluate the AUC value. For best random forest model the AUC is 0.6127 and the ROC curve is shown in Figure 4.

3.3 Support Vector Machine (SVM)

Support Vector Machine basically construct a hyper plane or set of hyper plane in an high dimensional space, which is used for classification, regression, and outliers detection.

In the next section below, I describe the step by step procedure to implement SVM.

- **Step 1:** I use R e1071 package to build two SVM classification models, one using polynomial kernel with degree 3 and the other one using a Gaussian radial kernel using the training data. Additionally, I use cost value = 0.01 for both model and gamma = 0.000001 for the second model which use radial kernel.

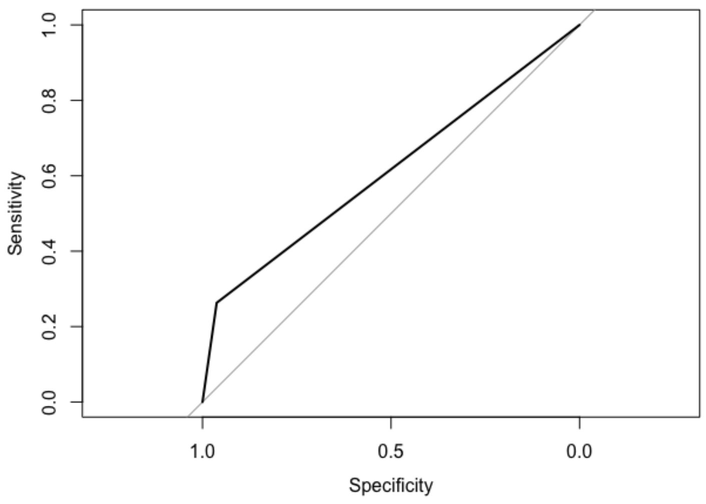


Figure 4: ROC Plot for Random Forest

- **Step 2:** I use validation data set for making prediction for both model.
- **Step 3:** For support vector machine, I calculate confusion matrix, sensitivity and specificity . Also, I calculate the accuracy which means the percentage of all correct predictions that were made by the model.

  The confusion matrix is given below for SVM model 1 (SVM model with polynomial kernel with degree 3):

		Reference		
		(−1)	(1)	**total**
	(−1)	TP (7815)	FP (1672)	P′
Predict outcome	(1)	FN (208)	TN (305)	N′
	total	P	N	

TP = True Positive, TN = True Negative,

FP = False Positive, and FN = False Negative.

The confusion matrix is given below for SVM model 2 (SVM model with Gaussian radial kernel:

		Reference		
		(−1)	(1)	**total**
	(−1)	TP (8023)	FP (1977)	P′
Predict outcome	(1)	FN (0)	TN (0)	N′
	total	P	N	

TP = True Positive, TN = True Negative,

FP = False Positive, and FN = False Negative.

- **Step 4:** Finally, I create the ROC curve and evaluate the AUC value for both model. And, the AUC is 0.5642 and 0.6127 and for model 1 and model 2 respectively. Also. Figure 5 and Figure 6 below shows the ROC curve for model 1 and model 2 respectively.

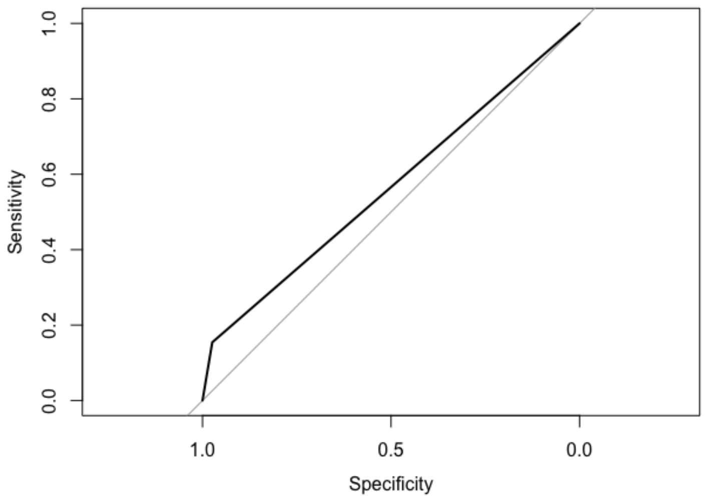


Figure 5: ROC Plot for SVM with Polynomial Kernel of Degree 3

4 Comparing Logistic Regression(LR), Random Forest (RF), SVM(Polynomial Kernel, and SVM(Gaussia Radial Kernel)

In the Table 1, I summarize the testing error, sensitivity, specificity, accuracy, and area under the curve of the four model. And all the model have testing error less than 20% but the SVM with radial kernel showing specificity equal 0 which is unconventional. Also earlier in each model section we saw ROC curve which is a graph showing the performance of the classification model at all classification

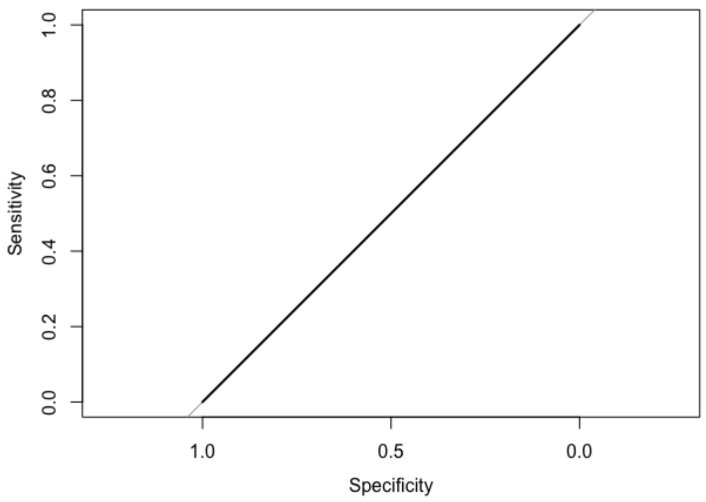


Figure 6: ROC Plot for SVM with Gaussian Radial Kernel

thresholds. Additionally, AUC (Area under the ROC curve) measures the entire two dimensional area underneath the entire ROC curve. In table 1, random forest have the smallest testing error with high AUC value. And, from the confusion matrix of random forest we saw that it performs well . So, among all the model random forest seems good compare to others.

Table 1: Performance Evaluation of Different Model

Criteria	LR	RF	SVM(Polynomial Kernel)	SVM(Radial Kernel)
Testing Error	0.1834	0.1759	0.188	0.1977
Sensitivity	0.9706	0.9624	0.9741	1.0
Specificity	0.1971	0.2630	0.1543	0.0
Accuracy	0.8166	0.8241	0.812	0.8023
AUC	0.5811	0.6127	0.5642	0.6127

5 Conclusion

In this study, I deploy logistic regression, random forest, support vector machine to a classification problem. In the earlier section I discuss confusion matrix, specificity, sensitivity, accuracy, ROC, and AUC curve of the model. Specificity which is a metric that evaluates a model's ability to predict true negative of each available category and random forest performs well in terms of this metric. And, the SVM with radial kernel cannot able to predict true negative of each available category and for this reason it have specificity equal 0. And, sensitivity evaluate a models ability to predict true positive of each available category. Accuracy is simply the proportion of individual who correctly classified the proportion of true positive and true negative.

All in all, after evaluating all the important metric, ROC, AUC, and checking the model assumption, I can conclude that among all the model random forest perform wells.

6 Data and Code

```
"
Author: Md Salman Rahman
Course: MATH 6333 Statistical Learning
Course Instructor: Dr. Tamer Oraby
Case Study 2
Goal: The main goal of the case study is to gets hands on experience in applying
the logistic regression, random forest, and Support Vector Machines to
solve a classification
problem
"
```

```
# Solution

setwd("/Users/salman/OneDrive - The University of
Texas-Rio Grande Valley/Course_video
/Statistical Learning/Homework_salman/Case Study 2")

## ### reading the data set

library("readxl")
library(ggplot2)
mar_data <- load("CaseStudy2.RData")
# checking the dimension of the data
dim(train)
dim(valid)

# checking missing value

table(is.na(train))
table(is.na(valid))



############
# exploring the data set by performing some descriptive analysis
#get means
sapply(train, mean, na.rm=TRUE)

#summary
```

```
summary(train)



# histogram
hist(train$TOT_HI_CRDT_CRDT_LMT)
hist(train$N_OF_SATISFY_FNC_REV_ACTS)



# pie chart
mytable<-table(train$PURCHASE) # 1 is Male and 2 is Female
pie(mytable)

# correlation analysis
library(corrplot)



# correlation matrixs
corrplot(cor(train[,1:69]),
         method = "number",
         type = "upper" # show only upper side
)

# scatterplot of several variable
pairs(train[,3:14])




### Step 2 Information Value ############
```

```
library(Information)
options(scipen=10)

########## Killing the weakest variables with IV
IV <- create_infotables(data=train, y="PURCHASE", ncore=2)
View(IV$Summary)

# only keeping IV value with > 0.5
train_new <- train[,c(subset(IV$Summary,IV>0.05)$Variable,"PURCHASE")]
dim(train_new)

valid_new <- valid[,c(subset(IV$Summary,IV>0.05)$Variable,"PURCHASE")]
dim(valid_new)




## Step 3: Multicollinearity check

library(ClustOfVar)
library(reshape2)
library(plyr)

tree <- hclustvar(train_new[,!(names(train_new)=="PURCHASE")])
nvars <- 20
part_init <- cutreevar(tree,nvars)$cluster
kmeans<-kmeansvar(X.quanti=train_new
[,!(names(train_new)=="PURCHASE")],init=part_init)
clusters <- cbind.data.frame(melt(kmeans$cluster), row.names(melt(kmeans$cluster)))
names(clusters) <- c("Cluster", "Variable")
clusters <- join(clusters, IV$Summary, by="Variable", type="left")
clusters <- clusters[order(clusters$Cluster),]
clusters$Rank <- ave(-clusters$IV, clusters$Cluster, FUN=rank)
View(clusters)
variables <- as.character(subset(clusters, Rank==1)$Variable)
variables




## Step 4: Adding a new response variable called Newpurchase


train_new$Newpurchase <- ifelse(train_new$PURCHASE==1,1,-1)


valid_new$Newpurchase <- ifelse(valid_new$PURCHASE==1,1,-1)



train_new$Newpurchase <- as.factor(train_new$Newpurchase)
valid_new$Newpurchase <- as.factor(valid_new$Newpurchase)



## Step 5: logistic regression model
glm.fit.case <- glm(Newpurchase ~., data=train_new[,c("Newpurchase", variables)], f
summary(glm.fit.case)
coef(glm.fit.case)
```

```
# prediction
logis.pred <- predict(glm.fit.case, newdata=valid_new[,c("Newpurchase", variables)]

logis.pred.1 <-ifelse(logis.pred>=0.5,1,-1)

# confusion matrix
library(caret)
logis.confu <- confusionMatrix(data=as.factor(logis.pred.1),reference = valid_new$N
# ROC curve

library(pROC)
roc.logis <- plot.roc(valid_new$Newpurchase, as.numeric(logis.pred.1))

#AUC
roc.1 <- roc(valid_new$Newpurchase, as.numeric(logis.pred.1))
auc(roc.1)

#ggsave("ROC_RF.png",roc.logis)



## Step 6: Random forest model
train_new$Newpurchase <- as.factor(train_new$Newpurchase)
valid_new$Newpurchase <- as.factor(valid_new$Newpurchase)

library(randomForest)
set.seed(101)
#with mtry = 1
```

```
mar_rf.1 <- randomForest(Newpurchase ~., data=train_new[,c("Newpurchase", variables
mar_rf.1$importance

rf_pred.1 <- predict(mar_rf.1, newdata=valid_new, outcome="train")

library(ROCR)
paste0("RF: ", AUC(valid_new$Newpurchase, rf_pred.1$predicted[,2])[1])

#with mtry = 2
mar_rf.2 <- randomForest(Newpurchase ~., data=train_new[,c("Newpurchase", variables
mar_rf.2$importance

rf_pred.2 <- predict(mar_rf.2, newdata=valid_new, outcome="train")



#with mtry = 3
mar_rf.3 <- randomForest(Newpurchase ~., data=train_new[,c("Newpurchase", variables
mar_rf.3$importance

rf_pred.3 <- predict(mar_rf.3, newdata=valid_new, outcome="train")

#with mtry = 4
mar_rf.4 <- randomForest(Newpurchase ~., data=train_new[,c("Newpurchase", variables
mar_rf.4$importance

rf_pred.4 <- predict(mar_rf.4,
newdata=valid_new[,c("Newpurchase", variables)], outcome="test")
library(pROC)
```

```
plot.roc(valid_new[,c("Newpurchase", variables)]$Newpurchase, as.numeric(rf_pred.4)


# #with mtry = 5
# mar_rf.5 <- randomForest(Newpurchase ~., data=train_new[,c("Newpurchase", variabl
# mar_rf.5$importance
#
# rf_pred.5 <- predict(mar_rf, newdata=valid_new, outcome="train")
#
# #with mtry = 6
# mar_rf.6 <- randomForest(Newpurchase ~., data=train_new[,c("Newpurchase", variabl
# mar_rf$importance
#
# rf_pred.6 <- predict(mar_rf.6, newdata=valid_new, outcome="train")
#
# #with mtry = 7
# mar_rf.7 <- randomForest(Newpurchase ~., data=train_new[,c("Newpurchase", variabl
# mar_rf.7$importance
#
# rf_pred.7 <- predict(mar_rf.7, newdata=valid_new, outcome="train")
#
# #with mtry = 8
# mar_rf.1 <- randomForest(Newpurchase ~.,
data=train_new[,c("Newpurchase", variables)],
ntree=10001, mtry=8, seed=2015)
# mar_rf.8$importance
#
# rf_pred.8 <- predict(mar_rf.8, newdata=valid_new, outcome="train")
```

```
#
#
# #with mtry = 9
# mar_rf.9 <- randomForest(Newpurchase ~., data=train_new[,c("Newpurchase", variabl
# mar_rf.9$importance
#
# rf_pred.9 <- predict(mar_rf.9, newdata=valid_new, outcome="train")
#
# #with mtry = 10
# mar_rf.10 <- randomForest(Newpurchase ~., data=train_new[,c("Newpurchase", variab
# mar_rf.10$importance
#
# rf_pred.10 <- predict(mar_rf.10, newdata=valid_new, outcome="train")
#
# #with mtry = 11
# mar_rf.11 <- randomForest(Newpurchase ~., data=train_new[,c("Newpurchase", variab
# mar_rf.11$importance
#
# rf_pred.11 <- predict(mar_rf.11, newdata=valid_new, outcome="train")
#
# #with mtry = 12
# mar_rf.12 <- randomForest(Newpurchase ~., data=train_new[,c("Newpurchase", variab
# mar_rf.12$importance
#
# rf_pred.12 <- predict(mar_rf.12, newdata=valid_new, outcome="train")
#
# #with mtry = 13
# mar_rf.13 <- randomForest(Newpurchase ~., data=train_new[,c("Newpurchase", variab
# mar_rf.13$importance
#
# rf_pred.13 <- predict(mar_rf.13, newdata=valid_new, outcome="train")




set.seed(101)
oob_error <- function(tree =10001, mtr){
  mar.rf<- randomForest(Newpurchase ˜., data=train_new[,c("Newpurchase", variables)
  error <- tail(mar.rf$err.rate,1)[,1]
  return(error)
}
x =seq(1:13)
y=c()

for (i in x) {
  p <- oob_error(tree = 10001, mtr =i)
  y=c(y,p)

}
lowest.x <- y[x]==min(y)



#with mtry = 9
mar_rf.9 <- randomForest(Newpurchase ˜., data=train_new[,c("Newpurchase", variables
importance <- mar_rf.9$importance
plot(importance)
```

```
rf_pred.9 <- predict
(mar_rf.9, newdata=valid_new[,c("Newpurchase", variables)], outcome="test")


# confusion matrix
library(caret)
rf.confu <- confusionMatrix(data=as.factor(rf_pred.9),reference = valid_new$Newpurc


# ROC curve
library(pROC)
plot.roc(valid_new$Newpurchase, as.numeric(rf_pred.9))

#AUC
roc.2 <- roc(valid_new$Newpurchase, as.numeric(rf_pred.9))
auc(roc.2)

## Step 7: SVM classification models

set.seed(101)
library(e1071)

#support vector classifier
svm.fit.1 <- svm(Newpurchase ~ .,
data=train_new[,c("Newpurchase", variables)],
kernel = "polynomial", degree = 3, probability = TRUE)

svm.pred <- predict(svm.fit.1,
newdata=valid_new[,c("Newpurchase", variables)], outcome="test")

#support vector machine
svm.fit.2 <- svm(Newpurchase ~ .,
data=train_new[,c("Newpurchase", variables)], kernel = "radial", gamma = 0.000001,

svm.pred.2 <- predict(svm.fit.2,
newdata=valid_new[,c("Newpurchase", variables)], outcome="test")

#testing error(all four method)
df.case <- valid_new[,c("Newpurchase", variables)]

error.2 <- mean(logis.pred.1 != df.case$Newpurchase) #logistic
error.3 <- mean(rf_pred.9 != df.case$Newpurchase) # rf
error.4 <- mean(svm.pred != df.case$Newpurchase) #SVM 1
error.5 <- mean(svm.pred.2 != df.case$Newpurchase) #SVM 2



# confusion matrix
library(caret)
svm.confu.1 <- confusionMatrix(data=as.factor(svm.pred),reference = valid_new$Newpu
svm.confu.2 <- confusionMatrix(data=as.factor(svm.pred.2),reference = valid_new$New
# ROC curve

library(ROCR)
plot.roc(valid_new$Newpurchase, as.numeric(svm.pred))
plot.roc(valid_new$Newpurchase, as.numeric(svm.pred.2))
```

```
#AUC -model 1
roc.3 <- roc(valid_new$Newpurchase, as.numeric(svm.pred))
auc(roc.3)

#AUC-model 2
roc.4 <- roc(valid_new$Newpurchase, as.numeric(svm.pred.2))
auc(roc.2)
```