Using Statistical Learning Techniques to Optimize the Marketing Campaign of a Charity Organization

1. Introduction

A charity organization is interested in running a more effective marketing campaign and maximizing their profit by identifying and targeting the possible donors. This project investigates the optimum classification and prediction models that can best predict the probability of a household to be donor and the donation amount.

Various classification and prediction models are investigated to find the most suitable classification and prediction models. The considered classification models are Logistic Regression, Linear discriminant analysis (LDA), Quadratic discriminant analysis (QDA), K-nearest neighbors (KNN), classification trees, support vector machines. The discussed prediction models include least squares regression (LSR), best subset selection, best subset selection using 10-fold cross-validation, Lasso regression using 10-fold cross-validation, principal component regression, PLS, GAM, and regression tree.

2. Results and Discussion

The entire dataset has been split into 3 parts: Training data: 3984 observations; Validation data: 2018 observations; Test data: 2007 observations. Each data set has 22 variables which their summary can be found in Table 4, Appendix I. The training data set was used to perform all aspects of model-fitting. Table 4 in Appendix I shows a descriptive summary of the dependent and independent variables in training data set. In order to get a sense of the data set and find possible correlations between variables, a pairwise scatterplot of all the variables is generated (Figure 10 in Appendix II). The pairwise scatterplot showed that some of the variables might have correlations, in which case, their interaction should also be considered in model development. Checking the histogram of these variables indicated that “avhv”, “incm”, and “inca” variables needed to be treated before running any model development which was addressed with a log transformation. More details of the EDA analysis can be found in Appendix II.

2.1. Classification Models

For each model discussed in this project, the classification model was developed using the training data. Then the validation data set is used to calculate the predicted profit using each model. The results of each model investigated in this project are presented in the following sections (2.1.1-2.1.5). The R codes associated with model can be found in Appendix III.

2.1.1. Logistic Regression

First, a logistic regression model was developed using all the variables as predictors in the model, without including any interaction terms. This model predicted the charity campaign profit to be $11402.5. However, after inspecting the coefficient estimates of this logistics regression model and their associated p-values, some of the predictors in the model are insignificant. Having p-values larger than 5% significance level, “reg3”, “reg4”, “genf”, “inca”, “plow”, “lgif”,

"rgif", and "agif" seemed to be not significant and possibly degrade the model performance. Therefore, another Logistic Regression model was developed with only the significant predictors in the model. The revised model predicted the profit to be $11410.5 which is slightly higher than the original logistic model. Interestingly, the performance of a logistic regression model could be improved by adding a quadratic order of "hinc" variable to the model (predicted profit of $11651.5). Adding more interaction terms to the model did not help to improve the profit prediction. Adding a quadratic term for "wrat", "chld" continued to improve the profit prediction to $ 11672 and 11687. The best performance in terms of profit prediction ($11772.5) was achieved when the following variables were considered as predictors in the model;
reg1, reg2, home, chld, hinc, I(hinc^2), I(wrat^2), wrat, avhv, I(avhv^2), incm, I(chld^2), npro, I(npro^2), tgif, I(tgif^2), tdon, I(tdon^2), tlag, I(tlag^2).

2.1.2. Linear discriminant analysis (LDA)

First a LDA model was developed with only the significant predictors found from the Logistic Regression model and the quadratic term of "hinc". The predicted profit was $11615 which is lower than that of the logistic regression model with similar predictors. Adding the quadratic term of all the predictors to the LDA model improved its predicted profit to $11627.5. A LDA model with simply all the variables included as predictors was also tried which resulted in poor profit prediction of $11350.5. As expected, including the nonsignificant variables in the model would only deteriorate the model performance.
It is notable that for every subset of the considered predictors, the Logistic Regression model had a superior performance than the LDA model.

2.1.3. Quadratic discriminant analysis (QDA)

The QDA model with all the variables included as predictors in the model, had profit prediction of $11266. Despite LDA and Logistic Regression models, adding a quadratic term of "hinc" variable lowered the predicted profit to $11225.5. This is likely because QDA inherently assumes the quadratic form, and so the addition of the term obscures the relationship. Performance of the QDA is noticeably poorer than LDA and Logistic regression. This suggests that the linear form assumed by logistic regression and LDA captures the true relationship more accurately than the quadratic form assumed by QDA.

2.1.4. K-nearest neighbors with k chosen using leave-one-out cross validation

First a KNN model was developed with K=3 to determine the significant predictors and then the optimal K was found. KNN model with K=3 with all predictors resulted in profit prediction of $11172. Running the KNN model with K=3 with only significant predictors reduced the predicted profit to $11045.5. Since excluding the nonsignificant predictors did not improve the model, finding the optimal K value for KNN model with all the predictors was attempted. In developing the KNN model, K was chosen using leave-one-out cross validation. Figure 1 shows the misclassification error rates based on leave-one-out cross-validation. Minimum misclassification error rate occurs at k=10. The random number seed of 2014 was used for running the cross validation loop and also for KNN model fitting purposes. The profit predicted by the KNN mode with k=10 is $11257 which is higher than the two other KNN models with K=3. However, even the best performing KNN classifier has a lower performance than the Logistic Regression model.

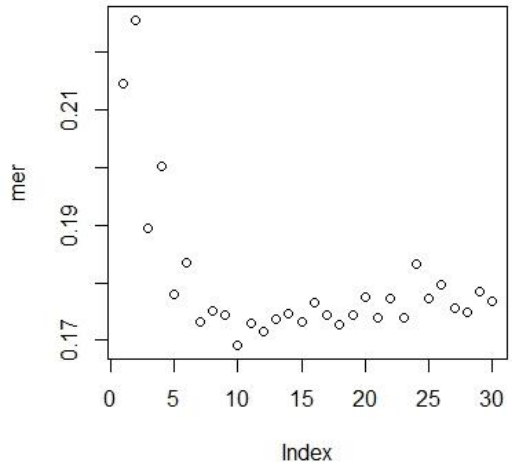

Figure 1. Misclassification error rates based on leave-one-out cross-validation. Minimum misclassification error rate occurs at k=10.

2.1.5. Classification Trees

Both unpruned and pruned classification tree models were developed. The unpruned classification tree structure can be seen in Figure 2. This model has 0.14 misclassification error rate and predicts the profit amount of $11140.5. A pruned tree model with seed value of 3 was developed. This model has 0.25 misclassification error rate and its predicted profit is $7256. which is siginificantly low. Unpruned tree model has better peroformance than pruned model, but still does not performs as well as the other discussed classification models.

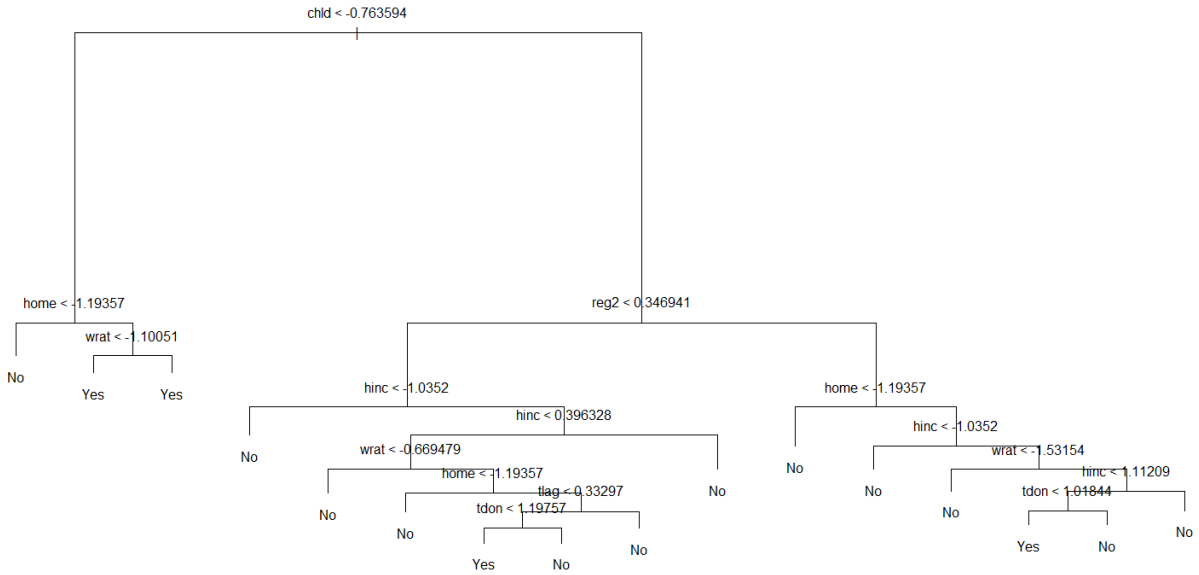

Figure 2. Unpruned classification tree structure of donors and non-donors.

2.1.6. Support Vector Machines (SVM)

The support vector machines with linear kernel and cost value of 0.01predicted the charity campaign profit of $10437.5. Using the radial kernel with similar cost value of 0.01 reduced the predicted profit to $9246. A polynomial kernel with the same cost value could improve the performance of the SVM model to $11159.5 profit prediction. Among the SVM models, the one with polynomial kernel had the best performance followed by the linear kernel. The SVM with the radial kernel had the lowest performance in terms of profit prediction.

2.2. Prediction Models

The same data sets used for the classification modeling are used for the prediction model development as well. The data sets would also undergo similar treatment processes as did for classification purposes. The training data set was used to perform all aspects of model-fitting. For each model discussed in this project, the best model size was selected based on the appropriate test on the models developed from the training data. Then the test data set was used to calculate the test mean square error of the chosen model. Finally the best model of the determined size – that was found from cross-validation approach- was fit on the full data set. The results of each prediction model investigated in this project are presented in the following sections (2.2.1-2.2.5).

The R codes associated with model can be found in Appendix IV.

2.2.1. Least Squares Regression (LSR)

The least regression equation (LSR) with all predictors in the models resulted in validation data set test MSE and standard error of 1.8675 and 0.1696615. By reviewing the coefficients of the LSR model and their associated p-values, it becomes evident that some of the predictors are not significant in the model. The LSR model was then developed without the “wrat” variable to check its effect on the LSR model performance. The test MSE and standard error of the revised model became 1.867 and 0.1696. The revised LSR model has lower validation set test MSE and standard error and therefore a better performance.

2.2.2. Variables Selection

2.2.2.1. Best Subset

Figure 3 shows the MSE values for best subset selection models of different sizes fitted on the training data. As is shown in Figure 3, best subset selection model with 19 independent variables has the lowest MSE value. However, 19 variables would result in a too big model which is not desirable. So we selected 11 variables. The MSE of the selected model with 11 variables is 1.819634.

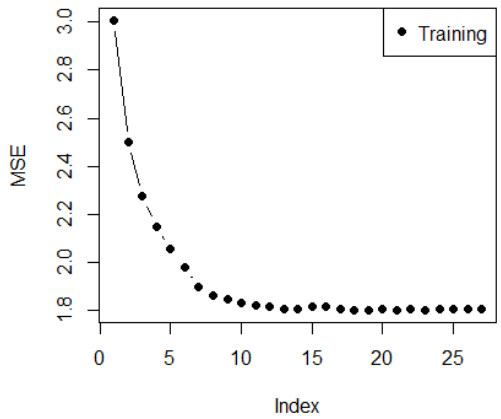

Figure 3. The model with 19 variables has the lowest MSE. However, 19 variables are too many variables which is not a desired model. So we select 11 variables.

2.2.2.2. Best Subset Selection using 10-fold Cross-Validation

Figure 4 shows the mean cross-validation (C.V.) error for best subset selection using 10-fold cross-validation models with different number of independent variables. The model with 14 predictors has the lowest mean C.V. error value of 1.6776. However the model with 14 predictors is doesn't not provide much improvement over the model with 11 variables.

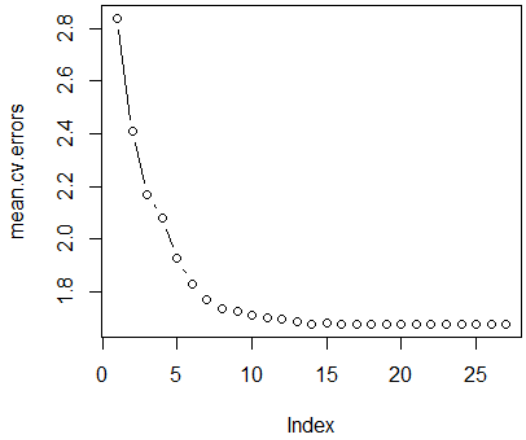

Figure 4. Mean cross-validation error for best subset selection using 10-fold cross-validation models with different number of independent variables.

2.2.3. Model Shrinkage

2.2.3.1. Lasso Regression

Figure 5 (left) shows the mean square error (MSE) versus log(lambda) for Lasso models using 10-fold cross-validation. The lambda with value of 0.0268 yields the cross-validation error within 1 standard error of the minimum. The test MSE of this model on validation data set is 1.68751.

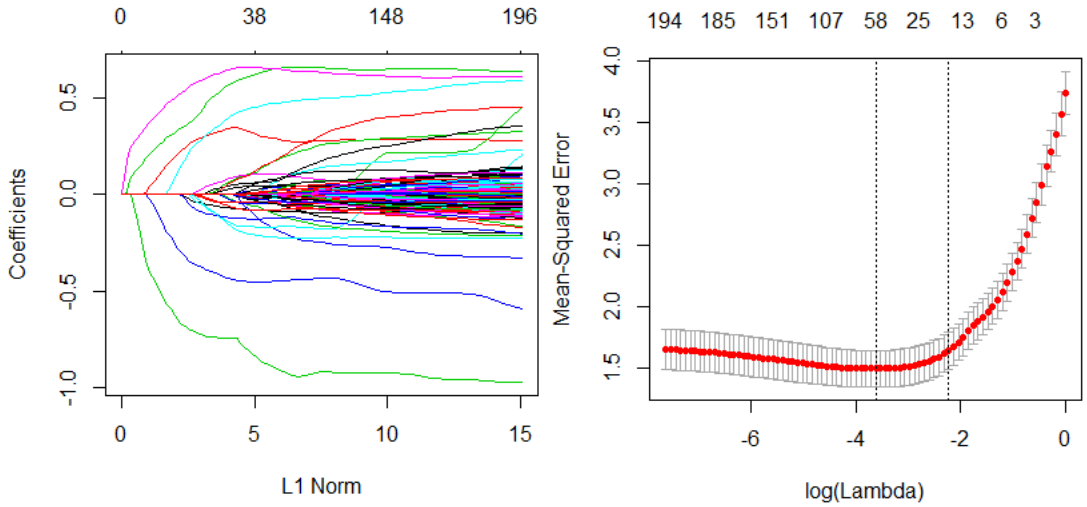

Figure 5. Lasso Model Using 10-Fold Cross-Validation: (left)coefficients versus L1 norm, (right) MSE versus log(lambda).

2.2.3.2. PCR Regression

The graph presented in Figure 6 shows that with 14 components, we can get the best results from the PCR model. The model with 25 components has the lowest MSEP. However, PCR with 25 components defeats its purpose and will have no advantage over the full regression model. So the PCR with 14 components that gives the next lowest MSEP is the proper choice.

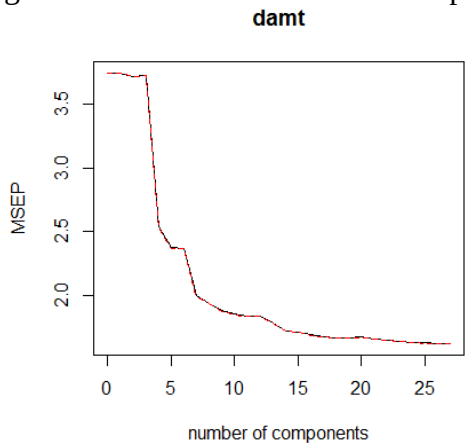

Figure 6. PCR model MSEP versus number of components.

2.2.3.3. PLS Regression

Figure 7 shows the MSEP versus the number of components which is 1.816689 at 5 number of components. From 5 to 25, adding more number of components to the model does not add the performance of the model.

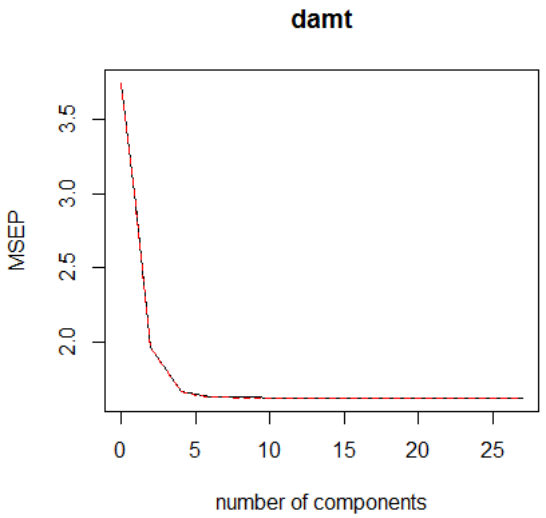

Figure 7. PLS model MSEP versus number of components.

2.2.4. Non-linear Models

2.2.4.1. GAM with spline

The reference model for the GAM analysis was set as the model obtained from the best subset model. Three different models with various combinations of the predictions and interactions were compared with the reference model. Figure 8 shows the scatterplots of the training data set variables versus damt, which hints possible interactions and curvatures in the variable which are concerned in the tested models. Table 1 summarizes the models tested against the reference models and their associated mean prediction error. As all ANOVA p-values are below the significance level of 0.05, we can reject the null hypothesis that these model are not different, and conclude that model 3 is significantly better than model 2, model 2 is better than model 1, and model 1 is better than model 0. It is noteworthy that the level of improvement from 2nd model to 3rd model is significantly higher than that of model 0 to 1st or 1st to 2nd model.

Table 1. GAM models and their associated mean prediction error.

Model	p-value of ANOVA analysis	Mean Prediction Error
Model 0: damt ~ reg3 + reg4 + home + chld + incm + npro + rgif + agif +plow:inca + hinc + plow	-	1.81963
Model 1: damt ~ reg3 + reg4 + home + chld + incm + npro + rgif + agif +plow:inca + ns(hinc,3) + ns(plow,5)	0.0006404	1.78648
Model 2: damt ~ reg3 + reg4 + home + chld + incm + npro + rgif + agif +plow:inca + s(hinc,3) + s(plow,5)	0.0006404	1.81963
Model 3: damt ~ reg3 + reg4 + home + chld + ns(incm,4)+npro + rgif + ns(agif,3) + plow:inca + ns(hinc,3) + ns(plow,5)	< 2.2e-16	1.65376

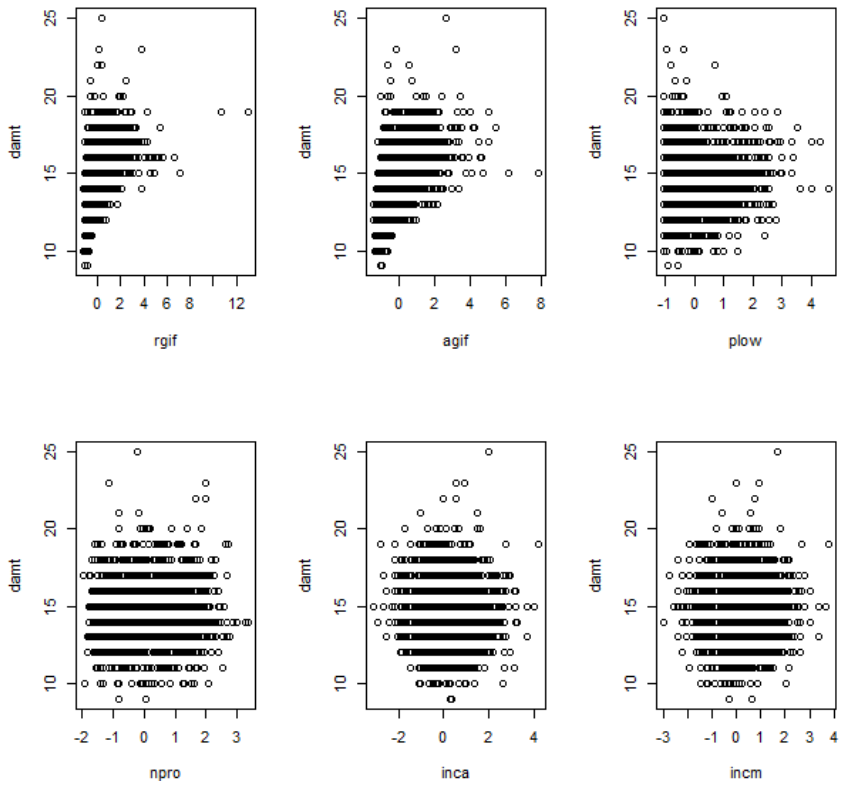

Figure 8. Scatter plots of training set variables versus damt.

2.2.5. Regression Tree

Figure 9 shows the plot with tree size on the x-axis and cross-validated classification error rate on the y-axis. This plot shows that the cross validation error is constantly decreasing with tree size up to 11. However, from 7 onward, the CV error doesn't change significantly. Therefore, 7 is selected as the optimum tree size. Mean square error of the tree model is 2.2411.

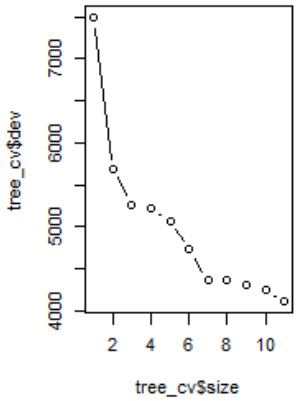

Figure 9. Regression tree model cross-validated classification error rate versus tree size.

3. Conclusion

3.1. Summary of Classification Models

In this project, the performance of Logistic Regression, LDA, QDA, KNN, classification trees, and SVM classification models on predicting the optimized profit from the charity marketing campaign were investigated. Table 2 summarizes the performance of best classification model of each type and its associated predicted profit. According to the result from the validation data set, Logistic Regression model has the best performance in classifying the target households as donor and non-donor and optimizing the predicted profit by identifying the donors with highest probability of donation. Finally the Logistic Regression model –identified as best classification model- is used to apply on the test data set and classify the cases in the test data set as whether they should be in the contact list of the charity marketing campaign or not. Application of the Logistic Regression model on the test data set resulted in identifying 327 cases -out of the total 2007 cases- which should be contacted by campaign via mail.

Table 2. Performance of best classification model of each type and its associated predicted profit

Classification Model	Predicted Profit
Logistic Regression (with significant variable and their quadratic terms)	**$11772.5**
LDA	$11627.5
QDA	$11266.0
KNN (all predictors, K=10)	$11257.0
Classification Trees (unpruned)	$11140.5
SVM (polynomial kernel, cost=0.01)	$11159.5

3.2. Summary of Prediction Models

In this project the performance of the various prediction models to best capture the charity marketing campaign success strategy is investigated. The analyzed prediction models are least squares regression (LSR), best subset selection, best subset selection using 10-fold cross-validation, Lasso regression using 10-fold cross-validation, principal component regression, PLS, GAM, and regression tree. Table 3 summarizes the test MSE on the validation data set of all the prediction models investigated in this study.

GAM model (model 3) with 1.653762 had the lowest test mean square error on the validation data set amongst all the models that were discussed in this project, followed by best subset selection using 10-fold cross-validation, and Lasso model using 10-fold cross validation. Regression tree model with 2.241075 had the highest test MSE.

Finally the best model (GAM) was applied to the test data set to predict the result for test data set cases.

Table 3. Performance of best perdition model of each type and its associated predicted profit

Model	Test MSE on Validation Set
LSR	1.867433
Best Subset selection	1.819634
Best subset Selection using 10-fold cross-validation	1.6776
Lasso model using 10-fold cross validation	1.68751
Principal Component Regression (PCR)	1.806217

PLS	1.816689
GAM (Model 3)	**1.653762**
Regression Tree	2.241075

4. Appendices

4.1. Appendix I: Tables

Table 4. Summary of variables in each data set

Variable	Definition and Possible Values
ID number	-
REG1, REG2, REG3, REG4	Region (There are five geographic regions;
HOME	(1 = homeowner, 0 = not a homeowner)
CHLD	Number of children
HINC	Household income (7 categories)
GENF	Gender (0 = Male, 1 = Female)
WRAT	Wealth Rating (Wealth rating uses median family income and population statistics from each area to index relative wealth within each state. The segments are denoted 0-9, with 9 being the highest wealth group and 0 being the lowest.)
AVHV	Average Home Value in potential donor's neighborhood in $ thousands
INCM	Median Family Income in potential donor's neighborhood in $ thousands
INCA	Average Family Income in potential donor's neighborhood in $

	thousands
PLOW	Percent categorized as “low income” in potential donor's neighborhood
NPRO	Lifetime number of promotions received to date
TGIF	Dollar amount of lifetime gifts to date
LGIF	Dollar amount of largest gift to date
RGIF	Dollar amount of most recent gift
TDON	Number of months since last donation
TLAG	Number of months between first and second gift
AGIF	Average dollar amount of gifts to date
DONR	Classification Response Variable (1 = Donor, 0 = Non-donor)
DAMT	Prediction Response Variable (Donation Amount in $)

4.2. Appendix II: Detailed Exploratory Data Analysis of Training Data Set

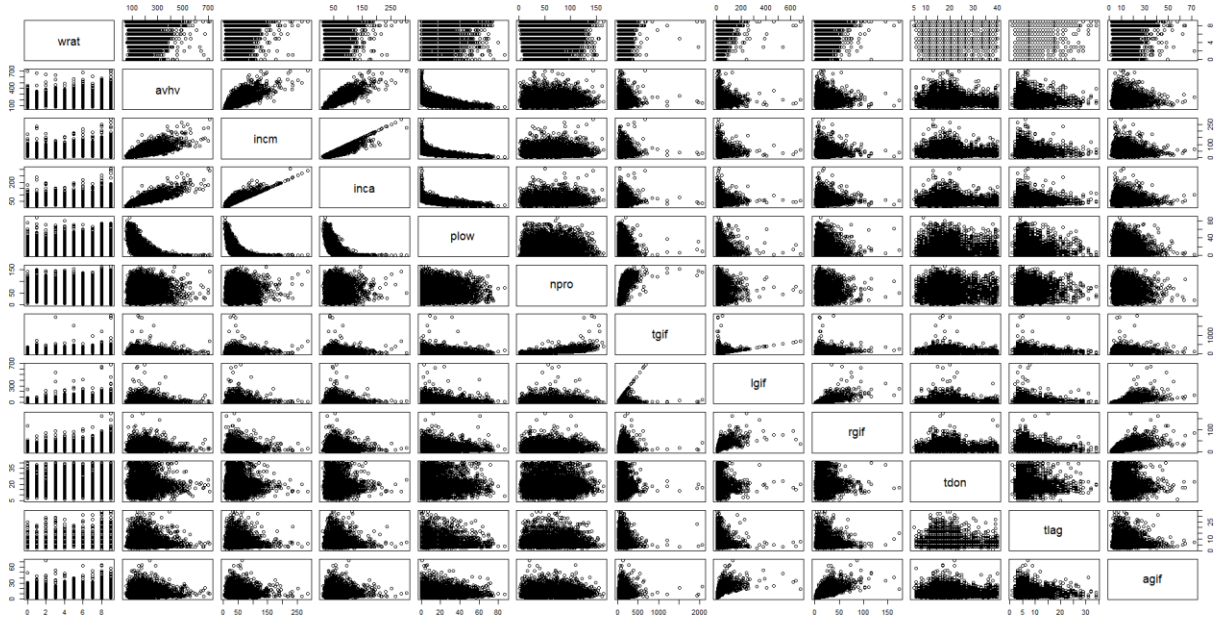

Figure 10. Pairwise scatter plots for all the variables.

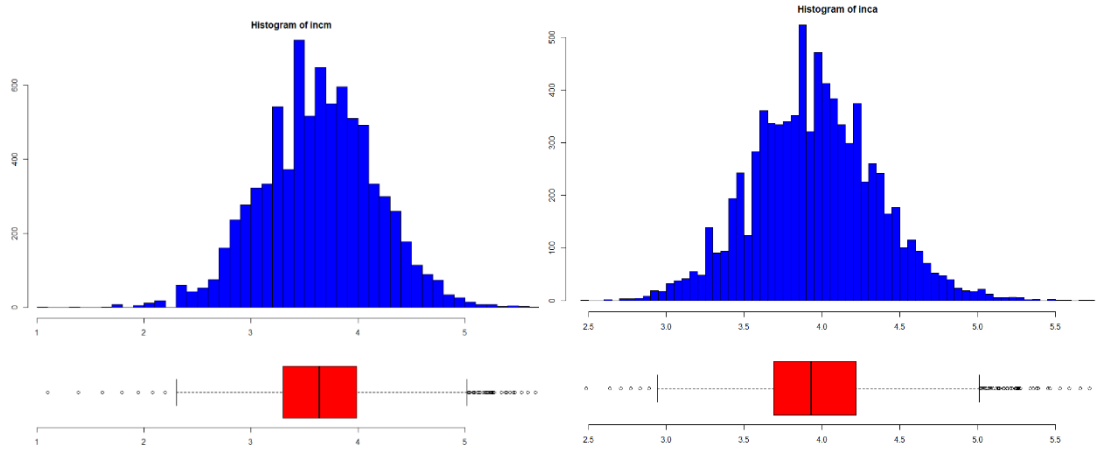

Figure 11. Histograms of incm and inca variables.

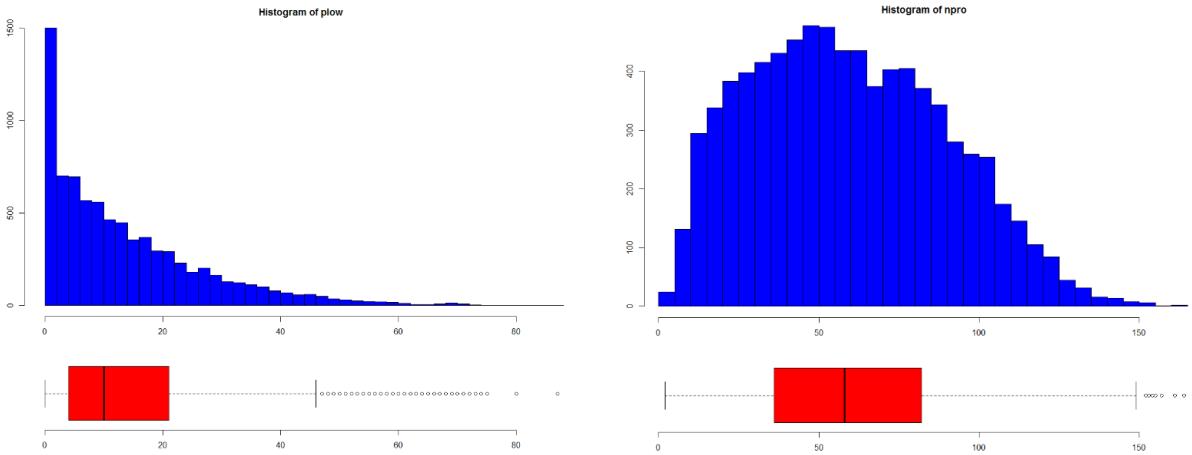

Figure 12. Histograms of plow and npro variables.

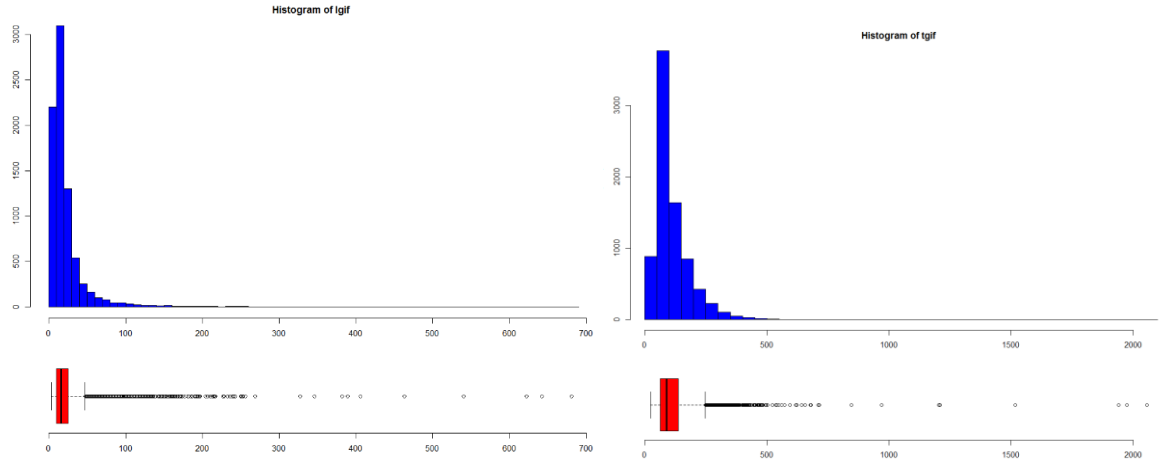

Figure 13. Histograms of lgif and tgif variables.

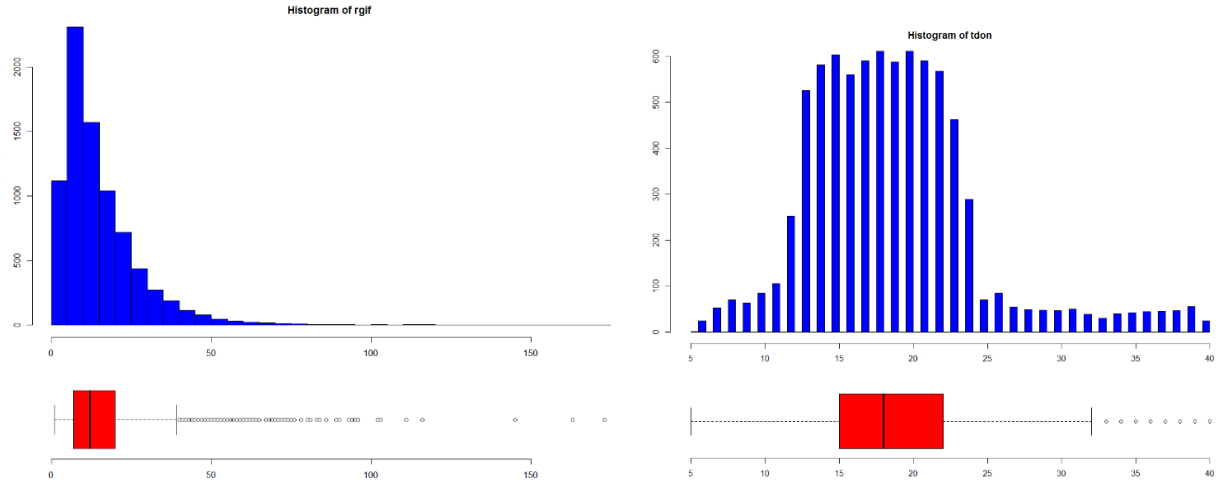

Figure 14. Histograms of rgif and tdon variables.

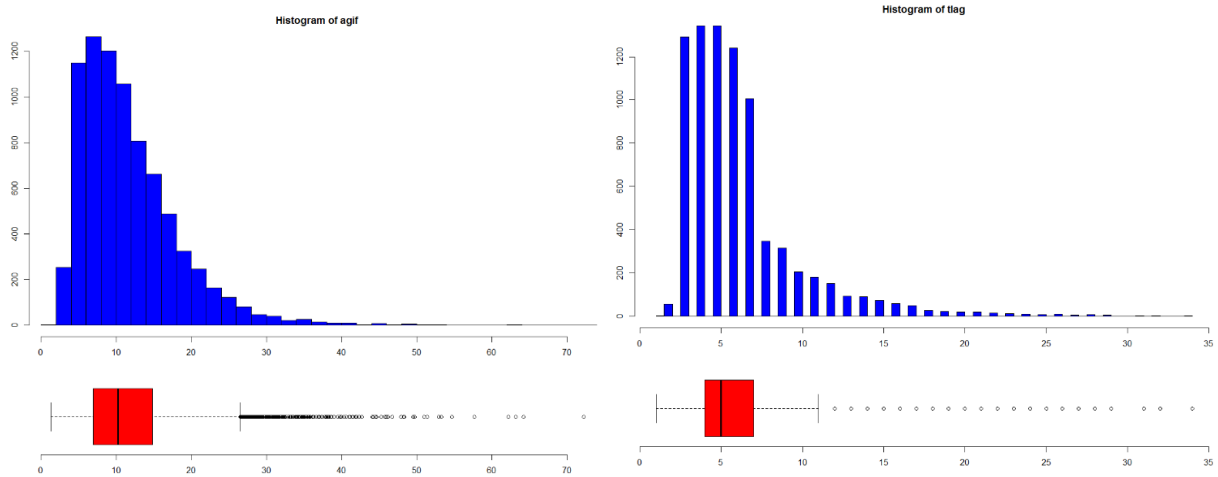

Figure 15. Histograms of agif and tlag variables.

The most obvious interactions are those between avhv, incm, inca and plow. We now take a closer look at these variables.

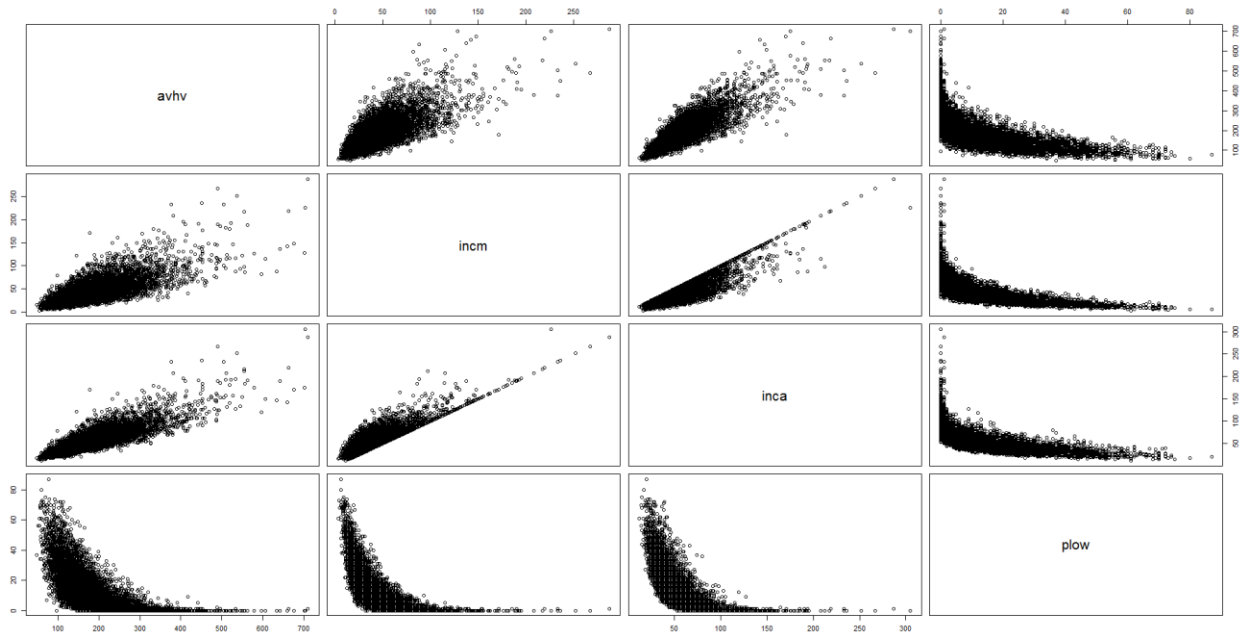

Figure 16

It looks like these variables need some transformation. We will transform the following avhv, incm and inca via a log transformation. The following plot shows the transformed variables:

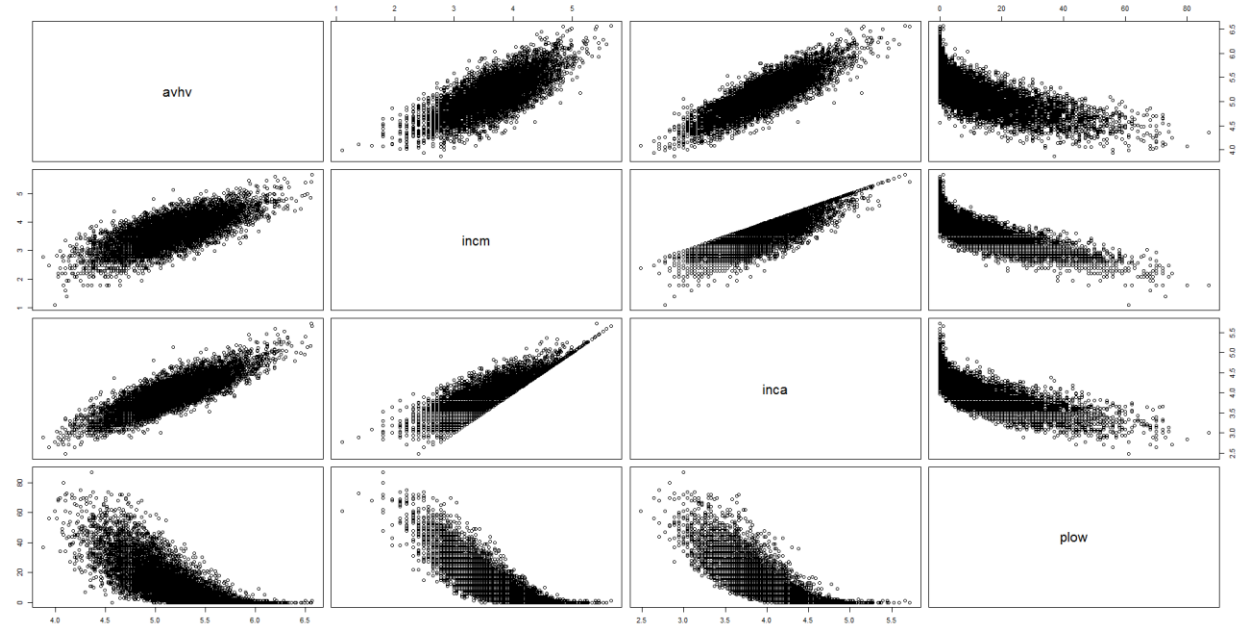

Figure 17. Pairwise scatter plot of transformed variables.

We can see that all variables have improved and look more randomly distributed than before. Next, we will check the means and standard deviations. Plot below shows that those variables with high standard deviation also have large means. This show that the data has to be standardized.

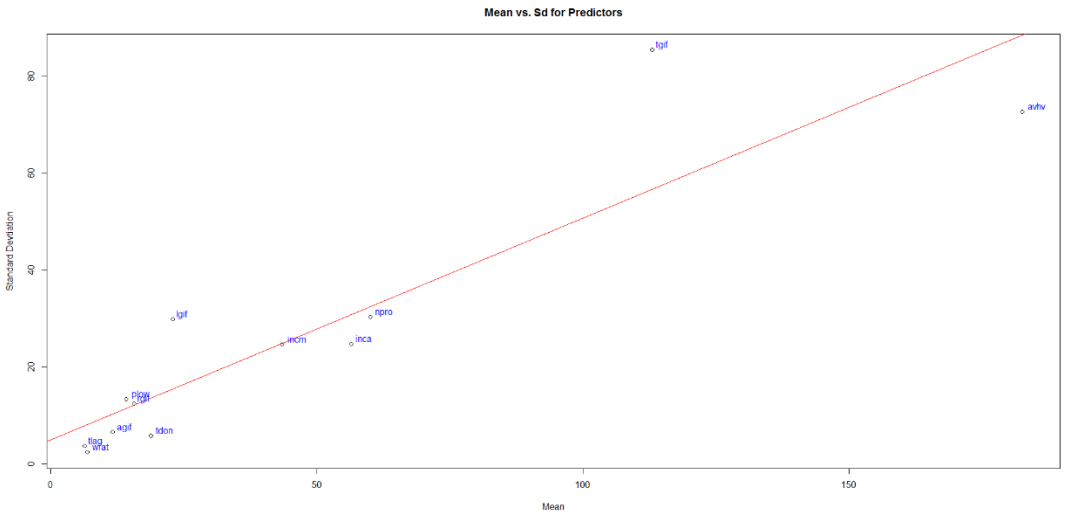

Figure 18. Standard deviation versus mean.

4.2. Appendix III: R-code (Classification Models)

```
rm(list=ls())
#Libraries
library(MASS)
library(class)
library(ISLR)
library(tree)
library(e1071)

# Read the data
charity = read.csv("C:/Users/pus172/Documents/copied from USB/STAT 897 - Applied Data Mining/Projects/Team Pro
ject/docs/charity.csv")

# Predictor transformations
charity.t = charity
charity.t$avhv <- log(charity.t$avhv)
charity.t$incm <- log(charity.t$incm)
charity.t$inca <- log(charity.t$inca)

# Set up data for analysis
data.train = charity.t[charity$part=="train",]
x.train = data.train[,2:21]
c.train = data.train[,22] # donr
n.train.= length(c.train) # 3984
y.train = data.train[c.train==1,23] # damt for observations with donr=1
n.train.y = length(y.train) # 1995

data.valid = charity.t[charity$part=="valid",]
```

```
x.valid = data.valid[,2:21]
c.valid = data.valid[,22] # donr
n.valid.c = length(c.valid) # 2018
y.valid = data.valid[c.valid==1,23] # damt for observations with donr=1
n.valid.y = length(y.valid) # 999

data.test = charity.t[charity$part=="test",]
n.test = dim(data.test)[1] # 2007
x.test = data.test[,2:21]

x.train.mean = apply(x.train, 2, mean)
x.train.sd = apply(x.train, 2, sd)
x.train.std = t((t(x.train)-x.train.mean)/x.train.sd) # standardize to have zero mean and unit sd
apply(x.train.std, 2, mean) # check zero mean
apply(x.train.std, 2, sd) # check unit sd
data.train.std.c = data.frame(x.train.std, donr=c.train) # to classify donr
data.train.std.y = data.frame(x.train.std[c.train==1,], damt=y.train) # to predict damt when donr=1

x.valid.std = t((t(x.valid)-x.train.mean)/x.train.sd) # standardize using training mean and sd
data.valid.std.c = data.frame(x.valid.std, donr=c.valid) # to classify donr
data.valid.std.y = data.frame(x.valid.std[c.valid==1,], damt=y.valid) # to predict damt when donr=1

x.test.std = t((t(x.test)-x.train.mean)/x.train.sd) # standardize using training mean and sd
data.test.std = data.frame(x.test.std)

train = data.train.std.c
valid = data.valid.std.c

############################################################################################################
#####                                  SOME USEFUL UTILITIES                                          ########
############################################################################################################

#Given a list of posterior probabilites, returns the cutoff by maximizing profit
cutoff = function(probs, c.valid){
  profit = cumsum(14.5*c.valid[order(probs, decreasing=T)]-2)
  number.mailings = which.max(profit) # number of mailings that maximizes profits
  cutoff.prob = sort(probs, decreasing=T)[number.mailings+1]
  return (cutoff.prob)
}

#Given a list of posterior probabilites, applies the classification using the cutoff and returns the profit de
rived
profit = function(probs, c.valid){
  cut = cutoff(probs, c.valid)
  chat = ifelse(probs>cut, 1, 0) # mail to everyone above the cutoff
  table = table(chat, c.valid) # classification table
  total.mailings = table["1","0"]+table["1","1"]
  correct.mailings = table["1","1"]
  income = 14.5*correct.mailings
  cost = 2*total.mailings
  prof = income-cost
  return(prof)
}

#Given a table, return the profit
table.profit = function(table){
```

```
  total.mailings = table["1","0"]+table["1","1"]
  correct.mailings = table["1","1"]
  income = 14.5*correct.mailings
  cost = 2*total.mailings
  prof = income-cost
  return(prof)
}



#############################################################################################################
#####                          LoGISTIC REGRESSION CLASSIFICATION                                  ########
#############################################################################################################

#First, fit an logistic regression model using all the predictors, and no interaction terms
model.log = glm(donr~., data = train, family = "binomial")
probs.log = predict(model.log, valid, type = "response")
profit.log = profit(probs.log, c.valid)
# 11402.5

#Now, inspect all p values and see if any predictors can be eliminated
summary(model.log)
#The p-values suggest eliminating reg3, reg4, genf, inca, plow, lgif, rgif, and agif

#Fit another logistic regression model, using only significant predictors
model.log = glm(donr~ reg1 + reg2 +home + chld+ hinc + wrat + avhv + incm + npro + tgif + tdon + tlag, data =
train, family = "binomial")
probs.log = predict(model.log, valid, type = "response")
profit.log = profit(probs.log, c.valid)
profit.log
#11410.5
summary(model.log)

#Now, try adding a quadratic term for hinc
model.log = glm(donr~ reg1 + reg2 +home + chld+ hinc + I(hinc^2) + wrat + avhv + incm + npro + tgif + tdon + t
lag, data = train, family = "binomial")
probs.log = predict(model.log, valid, type = "response")
profit.log = profit(probs.log, c.valid)
profit.log
#11651.5


#That improved profits significantly!

#Try some interaction terms
model.log = glm(donr~ reg1 + reg2 +home + chld+ hinc + I(hinc^2) + wrat + avhv + incm + npro + tgif + tdon + t
lag + avhv:incm + avhv:inca + avhv:plow + incm:inca +inca:plow, data = train, family = "binomial")
probs.log = predict(model.log, valid, type = "response")
profit.log = profit(probs.log, c.valid)
profit.log
#11635.5
#Not an improvement!

#Now, try adding a quadratic term for wrat
```

```
model.log = glm(donr~ reg1 + reg2 +home + chld+ hinc + I(hinc^2) + I(wrat^2)+ wrat + avhv + incm + npro + tgif
+ tdon + tlag, data = train, family = "binomial")
probs.log = predict(model.log, valid, type = "response")
profit.log = profit(probs.log, c.valid)
profit.log
#11672

#Now, try adding a quadratic term for chld
model.log = glm(donr~ reg1 + reg2 +home + chld+ hinc + I(hinc^2) + I(wrat^2)+ wrat + avhv + incm +I(chld^2) +
npro + tgif + tdon + tlag, data = train, family = "binomial")
probs.log = predict(model.log, valid, type = "response")
profit.log = profit(probs.log, c.valid)
profit.log
#11687

#Now, try adding a quadratic term for other predictors

#######BEST MODEL FROM ALL MODELS ATTEMPTED#############################
model.log = glm(donr~ reg1 + reg2 +home + chld+ hinc + I(hinc^2) + I(wrat^2)+ wrat + avhv +I(avhv^2)+ incm +I(
chld^2) + npro +I(npro^2)+ tgif +I(tgif^2)+ tdon +I(tdon^2)+ tlag +I(tlag^2), data = train, family = "binomial
")
probs.log = predict(model.log, valid, type = "response")
profit.log = profit(probs.log, c.valid)
profit.log
#11772.5 - IMPORTANT QUESTION TO CONSIDER: IS THIS OVERFITTING? Note that it is achieving higher profits on th
e validation data, not just on training.

model.best = glm(donr~ reg1 + reg2 +home + chld+ hinc + I(hinc^2) + I(wrat^2)+ wrat + avhv +I(avhv^2)+ incm +I
(chld^2) + npro +I(npro^2)+ tgif +I(tgif^2)+ tdon +I(tdon^2)+ tlag +I(tlag^2), data = train, family = "binomia
l")

#########################################################################################################
#####                                  LDA CLASSIFICATION                                       ########
#########################################################################################################

#Significant predictors, with quadratic term
model.lda = lda(donr~ reg1 + reg2 +home + chld+ hinc + I(hinc^2) + wrat + avhv + incm + npro + tgif + tdon + t
lag, data = train)
probs.lda = predict(model.lda, valid)$posterior[,2]
profit.lda = profit(probs.lda, c.valid)
profit.lda
#11615

#All predictors, with quadratic term
model.lda = lda(donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2) + genf + wrat + avhv + incm
+ inca + plow + npro + tgif + lgif + rgif + tdon + tlag + agif, data=train)
probs.lda = predict(model.lda, valid)$posterior[,2]
profit.lda = profit(probs.lda, c.valid)
profit.lda
#11627.5

#All predictors, without quadratic term
model.lda = lda(donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + genf + wrat + avhv + incm + inca + plo
w + npro + tgif + lgif + rgif + tdon + tlag + agif, data=train)
probs.lda = predict(model.lda, valid)$posterior[,2]
profit.lda = profit(probs.lda, c.valid)
```

```
profit.lda
#11350.5

#Logistic regression performs better than LDA

##############################################################################################
#####                                  QDA CLASSIFICATION                                ########
##############################################################################################

#With all predictors, and no quadratic term
model.qda = qda(donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + genf + wrat + avhv + incm + inca + plo
w + npro + tgif + lgif + rgif + tdon + tlag + agif, data=train)
probs.qda = predict(model.qda, valid)$posterior[,2]
profit.qda = profit(probs.qda, c.valid)
profit.qda
#11266

#With all predictors, and quadratic term
model.qda = qda(donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2) + genf + wrat + avhv + incm
+ inca + plow + npro + tgif + lgif + rgif + tdon + tlag + agif, data=train)
probs.qda = predict(model.qda, valid)$posterior[,2]
profit.qda = profit(probs.qda, c.valid)
profit.qda
#11225.5

#Note that with QDA, performance degenerates when you include the quadratic term.
#This is likely because QDA inherently assumes the quadratic form, and so the addition of the term obscures th
e relationship

#Performance is noticeably poorer than LDA and Logistic regression. This suggests that the linear form
#assumed by logistic regression and LDA capture the true relationship more accurately than the
#quadratic form assumed by QDA

##############################################################################################
#####                                  KNN CLASSIFICATION                                ########
##############################################################################################

#First determine which terms to include, using k=3. Afterwards, we will find optimal value for k

#All predictors
train.x = train[,-21]
train.y = train$donr
test.x = valid[,-21]
test.y = valid$donr


knn.fit = knn(train=train.x, test=test.x, cl=train.y, k=3, prob=T)
knn.pred.class = c(knn.fit)-1
knn.pred.prob = attr(knn.fit, "prob") #This is the probability of the winning class
knn.prob = knn.pred.class*knn.pred.prob+(1-knn.pred.class)*(1-knn.pred.prob) # n.train post probs of Y=1
profit.knn = profit(knn.prob, test.y)
profit.knn
#11172

#Only significant predictors
```

```
train.x = cbind(train$reg1, train$reg2,  train$home, train$chld, train$hinc,  train$wrat, train$avhv, train$in
cm,  train$npro, train$tgif,  train$tdon, train$tlag)
test.x = cbind(valid$reg1, valid$reg2,  valid$home, valid$chld, valid$hinc,  valid$wrat, valid$avhv, valid$inc
m,  valid$npro, valid$tgif,  valid$tdon, valid$tlag)
knn.fit = knn(train=train.x, test=test.x, cl=train.y, k=3, prob=T)
knn.pred.class = c(knn.fit)-1
knn.pred.prob = attr(knn.fit, "prob") #This is the probability of the winning class
knn.prob = knn.pred.class*knn.pred.prob+(1-knn.pred.class)*(1-knn.pred.prob) # n.train post probs of Y=1
profit.knn = profit(knn.prob, test.y)
profit.knn
#11045.5

#Using only significant predictors does not improve results. So, let's find optimal k using all predictors
train.x = train[,-21]
test.x = valid[,-21]
#Loop to find best value...
mer = rep(NA, 30)
print(mer)
set.seed(2014)
for (i in 1:30) mer[i] = sum((train.y-(c(knn.cv(train=train.x, cl=train.y, k=i))-1))^2)/3985
plot(mer)
which.min(mer)
#Optimal k is 10
knn.fit = knn(train=train.x, test=test.x, cl=train.y, k=10, prob=T)
knn.pred.class = c(knn.fit)-1
knn.pred.prob = attr(knn.fit, "prob") #This is the probability of the winning class
knn.prob = knn.pred.class*knn.pred.prob+(1-knn.pred.class)*(1-knn.pred.prob) # n.train post probs of Y=1
profit.knn = profit(knn.prob, test.y)
profit.knn
#11257

#We see that even the best performing KNN classifier, still performs more poorly than our logistic regresison
model


#############################################################################################################
#####                                  CLASSIFICATION TREES                                          ########
#############################################################################################################

#Unpruned tree
bin.donor = ifelse(train$donr>=1, "Yes", "No")
train.tree = cbind(train, bin.donor)
tree.donors = tree(bin.donor~.-donr, train.tree)
summary(tree.donors)
plot(tree.donors)
text(tree.donors, pretty=0)
tree.donors
tree.pred = predict(tree.donors, valid, type = "class")
table(tree.pred, c.valid)
          #c.valid
#tree.pred   0   1
#       No  783  70
#       Yes 236 929
totalMailed = 236 + 929
trueDonors = 929
cost = totalMailed * 2
```

```
income = trueDonors * 14.5
profit = income-cost
profit


#Pruned tree
set.seed(3)
cv.donors = cv.tree(tree.donors, FUN=prune.misclass)
cv.donors
#Lowest error is with size of one and two!
prune.donor = prune.misclass(tree.donors, best = 2)
summary(prune.donor)
plot(prune.donor)
text(prune.donor, pretty=0)
prune.donor
tree.pred = predict(prune.donor, valid, type = "class")
table(tree.pred, c.valid)
#         c.valid
#tree.pred   0   1
#       No  922 403
#       Yes  97 596
totalMailed = 97 + 596
trueDonors = 596
cost = totalMailed * 2
income = trueDonors * 14.5
profit = income-cost
profit

#The pruned tree does really poorly as compared to our other classification methods!
#Even the unpruned tree, though, does not perform as well as the other methods.

#############################################################################################################
#####                                  SUPPORT VECTOR MACHINES                                      ########
#############################################################################################################
train$donr = as.factor(train$donr)

#Linear Kernel
svm.linear = svm(donr~., data = train, kernel = "linear", cost = 0.01, scale = FALSE)
summary(svm.linear)
ypred=predict(svm.linear,valid)
table = table(predict = ypred, truth = c.valid)
prof = table.profit(table)
prof
#10437.5


#Radial Kernel
svm.radial = svm(donr~., data = train, kernel = "radial", cost = 0.01, scale = FALSE)
summary(svm.radial)
ypred=predict(svm.radial,valid)
table = table(predict = ypred, truth = c.valid)
prof = table.profit(table)
prof
#9246

#Polynomial Kernel
```

```
svm.poly = svm(donr~., data = train, kernel = "polynomial", cost = 0.01, scale = FALSE)
summary(svm.poly)
ypred=predict(svm.poly,valid)
table = table(predict = ypred, truth = c.valid)
prof = table.profit(table)
prof
#11159.5

#The SVM does not perform as well as the Logistic Regression



#############################################################################################
#####                            FINAL CHAT AND WRITE TO FILE                          ########
#############################################################################################
post.test <- predict(model.best, data.test.std, type="response") # post probs for test data
# Oversampling adjustment for calculating number of mailings for test set
post.valid.log1 <- predict(model.best, data.valid.std.c, type="response") # n.valid post probs
profit.log1 <- cumsum(14.5*c.valid[order(post.valid.log1, decreasing=T)]-2)
n.mail.valid <- which.max(profit.log1)
tr.rate <- .1 # typical response rate is .1
vr.rate <- .5 # whereas validation response rate is .5
adj.test.1 <- (n.mail.valid/n.valid.c)/(vr.rate/tr.rate) # adjustment for mail yes
adj.test.0 <- ((n.valid.c-n.mail.valid)/n.valid.c)/((1-vr.rate)/(1-tr.rate)) # adjustment for mail no
adj.test <- adj.test.1/(adj.test.1+adj.test.0) # scale into a proportion
n.mail.test <- round(n.test*adj.test, 0) # calculate number of mailings for test set
cutoff.test <- sort(post.test, decreasing=T)[n.mail.test+1] # set cutoff based on n.mail.test
chat.test <- ifelse(post.test>cutoff.test, 1, 0) # mail to everyone above the cutoff
table(chat.test)
#chat.test
#0    1
#1680  327
#Based on this, we will mail to 327 of the test observations

length(chat.test) # check length = 2007
chat.test[1:10] # check this consists of 0s and 1s

cht <- data.frame(chat=chat.test)    # chat.test is your classification predictions (2007 0 and 1 values)
write.csv(cht, file="C:/Users/pus172/Documents/copied from USB/STAT 897 - Applied Data Mining/Projects/Team Pr
oject/docs/chat.csv", row.names=FALSE)
```

4.2. Appendix IV: R-code (Prediction Models)

```
rm(list=ls())
charity <- read.csv("C:/Users/pus172/Documents/copied from USB/STAT 897 - Applied Data Mining/Projects/Team Pr
oject/docs/charity.csv")
head(charity)
names(charity)
xmat = charity[,c(-0:-8,-22,-23,-24)]
#pairs(xmat)

# predictor transformations

charity.t <- charity
charity.t$avhv <- log(charity.t$avhv)
```

```
charity.t$incm <- log(charity.t$incm)
charity.t$inca <- log(charity$inca)

xmat2 = charity.t[,c(-0:-8,-22,-23,-24)]
#pairs(xmat2) # the scatter plots look natural after these transformations

head(charity.t,10)
names(charity.t)
num_x = charity.t[!(charity.t$donr%in%c(0,NA)),]

head(num_x,10)
# add further transformations if desired
# for example, some statistical methods can struggle when predictors are highly skewed

# set up data for analysis

data.train <- charity.t[charity$part=="train",]
x.train <- data.train[,2:21]
c.train <- data.train[,22] # donr
n.train.c <- length(c.train) # 3984
y.train <- data.train[c.train==1,23] # damt for observations with donr=1
n.train.y <- length(y.train) # 1995

data.valid <- charity.t[charity$part=="valid",]
x.valid <- data.valid[,2:21]
c.valid <- data.valid[,22] # donr
n.valid.c <- length(c.valid) # 2018
y.valid <- data.valid[c.valid==1,23] # damt for observations with donr=1
n.valid.y <- length(y.valid) # 999

data.test <- charity.t[charity$part=="test",]
n.test <- dim(data.test)[1] # 2007
x.test <- data.test[,2:21]

x.train.mean <- apply(x.train, 2, mean)
x.train.sd <- apply(x.train, 2, sd)
x.train.std <- t((t(x.train)-x.train.mean)/x.train.sd) # standardize to have zero mean and unit sd
apply(x.train.std, 2, mean) # check zero mean
apply(x.train.std, 2, sd) # check unit sd
data.train.std.c <- data.frame(x.train.std, donr=c.train) # to classify donr
data.train.std.y <- data.frame(x.train.std[c.train==1,], damt=y.train) # to predict damt when donr=1

x.valid.std <- t((t(x.valid)-x.train.mean)/x.train.sd) # standardize using training mean and sd
data.valid.std.c <- data.frame(x.valid.std, donr=c.valid) # to classify donr
data.valid.std.y <- data.frame(x.valid.std[c.valid==1,], damt=y.valid) # to predict damt when donr=1

x.test.std <- t((t(x.test)-x.train.mean)/x.train.sd) # standardize using training mean and sd
data.test.std <- data.frame(x.test.std)



# Prediction modeling

## Least squares regression

model.ls1 <- lm(damt ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + genf + wrat +
```

```
                avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + tlag + agif,
              data.train.std.y)

pred.valid.ls1 <- predict(model.ls1, newdata = data.valid.std.y) # validation predictions
mean((y.valid - pred.valid.ls1)^2) # mean prediction error
# 1.867523
sd((y.valid - pred.valid.ls1)^2)/sqrt(n.valid.y) # std error
# 0.1696615

# drop wrat for illustrative purposes
model.ls2 <- lm(damt ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + genf +
                avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + tlag + agif,
              data.train.std.y)

head(data.train.std.y)
head(data.valid.std.y)
head(data.test.std)

pred.valid.ls2 <- predict(model.ls2, newdata = data.valid.std.y) # validation predictions
mean((y.valid - pred.valid.ls2)^2) # mean prediction error
# 1.867433
sd((y.valid - pred.valid.ls2)^2)/sqrt(n.valid.y) # std error
# 0.1696498

# Results

# MPE  Model
# 1.867523 LS1
# 1.867433 LS2

# select model.ls2 since it has minimum mean prediction error in the validation sample

yhat.test <- predict(model.ls2, newdata = data.test.std) # test predictions

# Save final results for both classification and regression

length(chat.test) # check length = 2007
length(yhat.test) # check length = 2007
chat.test[1:10] # check this consists of 0s and 1s
yhat.test[1:10] # check this consists of plausible predictions of damt

ip <- data.frame(chat=chat.test, yhat=yhat.test) # data frame with two variables: chat and yhat
write.csv(ip, file="~/Documents/teaching/psu/ip.csv",
          row.names=FALSE) # use group member initials for file name

# submit the csv file in Angel for evaluation based on actual test donr and damt values



######### Variable selection ########
## Best Subsets
dim(data.train.std.y)
library(leaps)
regfit.best = regsubsets(damt ~ .+avhv:incm+avhv:inca+avhv:plow+incm:inca+incm:plow+inca:plow+lgif:tgif, data
= data.train.std.y, nvmax = 27)
reg.summary = summary(regfit.best)
```

```
reg.summary$rss

test.mat = model.matrix(damt ~ .+avhv:incm+avhv:inca+avhv:plow+incm:inca+incm:plow+inca:plow+lgif:tgif, data =
data.valid.std.y)
val.errors = rep(NA,27)
for (i in 1:27) {
  coefi = coef(regfit.best, id = i)
  pred = test.mat[, names(coefi)]%*%coefi
  val.errors[i] = mean((data.valid.std.y$damt- pred)^2)
}



plot(val.errors, ylab = "MSE", pch = 19, type = "b")
legend("topright", legend = c("Training"), col = c( "black"),
       pch = 19)

val.errors
which.min(val.errors)# 19 is the best but too many. we select 11

train_val = rbind(data.train.std.y,data.valid.std.y) #combine the train and validation datasets
regfit.best.full = regsubsets(damt ~ .+avhv:incm+avhv:inca+avhv:plow+incm:inca+incm:plow+inca:plow+lgif:tgif,
data = train_val, nvmax = 27)
coef(regfit.best.full,11) #optimom model
val.errors[11] # error for optimal model



## K-fold best subset
head (train_val)
k=10
set.seed (1)
folds=sample (1:k,nrow(train_val),replace =TRUE)
cv.errors =matrix (NA ,k,27, dimnames =list(NULL , paste (1:27) ))

predict.regsubsets = function(object, newdata, id, ...) {
  form = as.formula(object$call[[2]])
  mat = model.matrix(form, newdata)
  coefi = coef(object, id = id)
  mat[, names(coefi)] %*% coefi
}

for(j in 1:k){
   best.fit =regsubsets(damt~.+avhv:incm+avhv:inca+avhv:plow+incm:inca+incm:plow+inca:plow+lgif:tgif,data=trai
n_val[folds !=j,],nvmax =27)
   for(i in 1:27) {
     pred=predict (best.fit ,train_val [folds ==j,], id=i)
     cv.errors [j,i]=mean( (train_val$damt[folds ==j]-pred)^2)
     }
}

mean.cv.errors =apply(cv.errors ,2, mean)
mean.cv.errors
par(mfrow =c(1,1))
plot(mean.cv.errors ,type="b") # still 11 is the best
```

```
regfit_pred = lm(damt~reg3+reg4+home+chld+hinc+incm+plow+npro+rgif+agif+inca:plow, data = train_val)

yhat_subset = predict(regfit_pred,data.test.std) #predicted vales for test
head(yhat_subset)

######### model shrinkage ########

# LASSO Regression
library(glmnet)
names(data.valid.std.y)
x_valid= model.matrix (damt~.*.,data.valid.std.y)
y_valid= data.valid.std.y$damt
x_train = model.matrix (damt~.*.,data.train.std.y)
y_train = data.train.std.y$damt
x_full = model.matrix (damt~.*.,train_val)
y_full = train_val$damt
x_test = model.matrix(~.*.,data.test.std )


grid =10^seq (10,-3, length =1000)
lasso.mod =glmnet (x_train,y_train,alpha =1, lambda =grid, standardize=FALSE, thresh =1e-12)

set.seed (1)
cv.out =cv.glmnet (x_train,y_train,alpha =1)
plot(cv.out)
bestlam =cv.out$lambda.min #best lambda
bestlam

lasso.pred=predict(lasso.mod ,s=bestlam ,newx=x_valid)
mean((lasso.pred-y_valid)^2)
lasso_full = glmnet(x_full,y_full,alpha =1)

plot(lasso_full)
yhat_lasso = predict (lasso_full ,s=bestlam, type = "response", newx = x_test) #yhats predecited with lasso
lasso_coef = predict (lasso_full ,s=bestlam, type = "coefficient", newx = x_test)
lasso_coef[lasso_coef!=0] # non-zero coefficients


# PCR Regression

library (pls)
set.seed (1)
pcr_fit=pcr(damt~.+avhv:incm+avhv:inca+avhv:plow+incm:inca+incm:plow+inca:plow+lgif:tgif, data=data.train.std.
y ,scale=FALSE,validation ="CV")
summary (pcr_fit )
validationplot(pcr_fit,val.type="MSEP")

pcr_valid=predict (pcr_fit ,as.data.frame(x_valid) , ncomp =25)
mean((pcr_valid-as.data.frame(y_valid))^2)


pcr_full=pcr(damt~.+avhv:incm+avhv:inca+avhv:plow+incm:inca+incm:plow+inca:plow+lgif:tgif, data=train_val,scal
e=FALSE,validation ="CV")
yhat_pcr=predict (pcr_full ,data.test.std , ncomp =14)

#PLS Regression
```

```
set.seed (1)
plsr_fit=plsr(damt~.+avhv:incm+avhv:inca+avhv:plow+incm:inca+incm:plow+inca:plow+lgif:tgif, data=data.train.st
d.y ,scale=FALSE,validation ="CV")
summary (plsr_fit )
validationplot(plsr_fit,val.type="MSEP")

plsr_valid=predict (plsr_fit ,as.data.frame(x_valid) , ncomp =5)
mean((plsr_valid-as.data.frame(y_valid))^2)


plsr_full=plsr(damt~.+avhv:incm+avhv:inca+avhv:plow+incm:inca+incm:plow+inca:plow+lgif:tgif, data=train_val,sc
ale=FALSE,validation ="CV")
yhat_plsr=predict (plsr_full ,data.test.std , ncomp =5)

######### non-linear models ########
# GAM
# we will use all the variables resulted from the shrinked model
coef(regfit_pred)
library (splines )
library(gam)
par(mfrow=c(2,3),oma = c(0, 0, 3, 0))
plot(data.train.std.y$rgif,data.train.std.y$damt, ylab="damt", xlab = "rgif")
plot(data.train.std.y$agif,data.train.std.y$damt, ylab="damt", xlab ="agif")
plot(data.train.std.y$plow,data.train.std.y$damt, ylab="damt", xlab ="plow")
plot(data.train.std.y$npro,data.train.std.y$damt, ylab="damt", xlab ="npro")
plot(data.train.std.y$inca,data.train.std.y$damt, ylab="damt", xlab ="inca")
plot(data.train.std.y$incm,data.train.std.y$damt, ylab="damt", xlab ="incm")
mtext("Scatter plots for training set", outer = TRUE, cex = 1.5)


gam0 = glm(damt~reg3+reg4+home+chld+incm+npro+rgif+agif+plow:inca+hinc+plow,data=data.train.std.y)
gam1=glm(damt~reg3+reg4+home+chld+incm+npro+rgif+agif+plow:inca+ns(hinc ,3)+ns(plow ,5),data=data.train.std.y)
gam2=glm(damt~reg3+reg4+home+chld+incm+npro+rgif+agif+plow:inca+s(hinc ,3)+s(plow ,5),data=data.train.std.y)
gam3=glm(damt~reg3+reg4+home+chld+ns(incm,4)+npro+rgif+ns(agif,3)+plow:inca+ns(hinc ,3)+ns(plow ,5),data=data.
train.std.y)

gam0_pred<- predict(gam0, newdata = data.valid.std.y)
gam1_pred<- predict(gam1, newdata = data.valid.std.y) # validation predictions
gam2_pred<- predict(gam2, newdata = data.valid.std.y) # validation predictions
gam3_pred<- predict(gam3, newdata = data.valid.std.y)
anova(gam0,gam1,gam2,gam3, test = "F") # there is enough evidence that gam3 is the full model (gam3) is the su
perior model

mean((y.valid - gam0_pred)^2) # mean prediction error 0
mean((y.valid - gam1_pred)^2) # mean prediction error 1
mean((y.valid - gam2_pred)^2) # mean prediction error 2
mean((y.valid - gam3_pred)^2) # mean prediction error 3
par(mfrow=c(2,2))
plot(gam3, se=TRUE ,col ="blue ")

yhat_gam3 = predict(gam3,newdata =data.test.std)


######### regression tree ########
library(tree)
```

```
tree_fit = tree(damt~. , data = data.train.std.y)
plot(tree_fit)
text(tree_fit,pretty = 0)
summary(tree_fit)
tree_cv = cv.tree(tree_fit)
plot(tree_cv$size ,tree_cv$dev ,type="b")
prune_tree=prune.tree(tree_fit,best =7)
plot(prune_tree)
text(prune_tree,pretty = 0)

yhat_tree=predict (tree_fit,newdata =data.valid.std.y)
tree_valid=data.valid.std.y[,21]
plot(yhat_tree,tree_valid, pch = 19)
abline (0,1)
mean((yhat_tree -tree_valid)^2)

yhat_tree_final=predict (tree_fit,newdata =data.test.std)
```