Predict 422 Final Project: Charity

Tyler Violillo, Morgan Kisselburg, Junlan Zhou, Bivek Adhikari

Northwestern University

Contents

INTRODUCTION

The purpose of this analysis was to develop a machine learning model to assist in the marketing efforts of a charitable organization. The organization wanted help deciding where to focus their marketing efforts in order to maximize the dollars brought into the charity. Recent mailing records suggested that simply mass-mailing out ads was a not a cost-effective process, and was actually losing money for them. Each mailing costs the company $2.00 and each response to these mailings averages out to a $14.50 donation to the charity. However, the response rate to the mass mailing system is only 10%, which results in an expected profit from each mailing of -$0.55. A more efficient, and profitable process would stop the organization from losing so much money on these mailings, and transform it into realizing positive results.

In order to help the organization, we were provided with historical mailing records from the charitable organization. The approach to this problem was to create two separate models: the first model would predict *who* would donate and the second model would predict *how much* those individuals would donate. By focusing on not only the individuals who are likely to donate, but also those who are likely to donate the most, the organization can ensure that they are taking all steps possible to try and prevent losses on their mailing campaign. In addition to preventing losses, the expected result will produce more money for the charity.

Our team attacked this problem in several steps, which are defined more thoroughly in the analysis section below. Furthermore, we familiarized ourselves with the data by performing some preliminary Exploratory Data Analysis and, based off that, transformed and cleaned the data to reflect best common practices. Next, we created several different types of models for both the classification and predictive purposes of the assignment. Finally, we chose the best classification model based on the maximum profit, and the best predictive model based on the lowest MSE to use as our final models. These two best models were used to predict the test data given to us by the organization for comparison against their actual results.

ANALYSIS

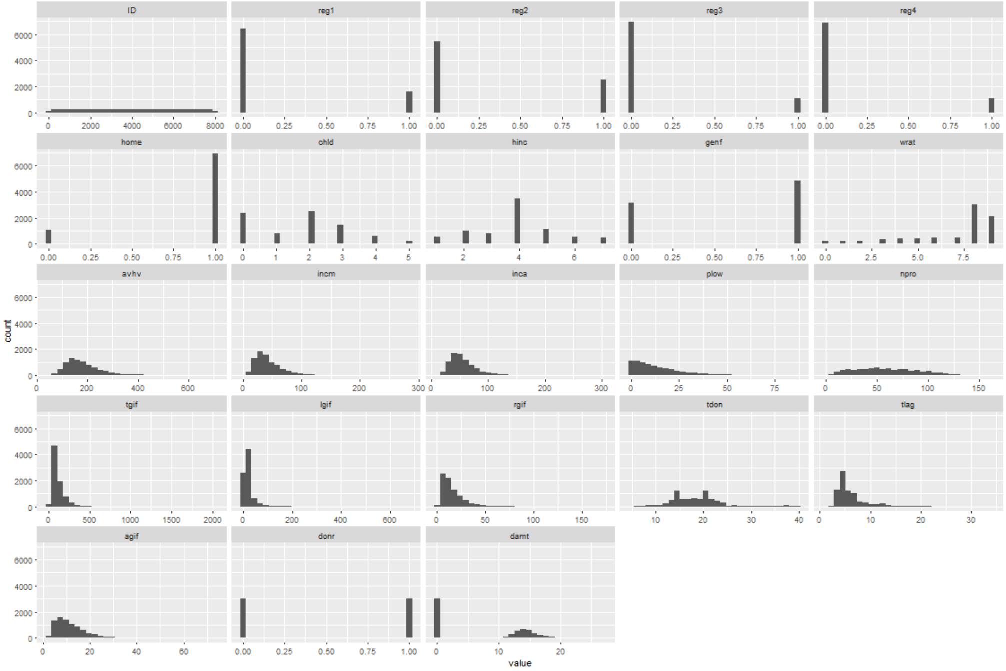
Exploratory Data Analysis and Variable Transformations

Initial EDA

The first step we took after receiving the data from the charity organization was to assess what we were working with. We had been given 8009 observations and 24 variables as part of the charity data set. Of these 8009 observations, we used 3984 as the training set to build the models, 2018 as the validation set to validate the models, and the final 2007 observations were the test set. The test set did not provide the “answers” of the correct donor (donr), and donor amount (damt) so this will be how the charity organization evaluates our model. Of the 24 variables, only 20 were used to train the models as the ID, partition variable, and the two response variables were not used in model creation.

With the exception of the partition variable (factor variable which was only used to split the data into the train, validation, and test sets), the other variables were all numeric. This made it relatively easy to create a histogram for each of the variable's distributions as shown in the figure below:

Variable Distribution Histograms



Looking at the figure above, a few observations immediately become obvious:

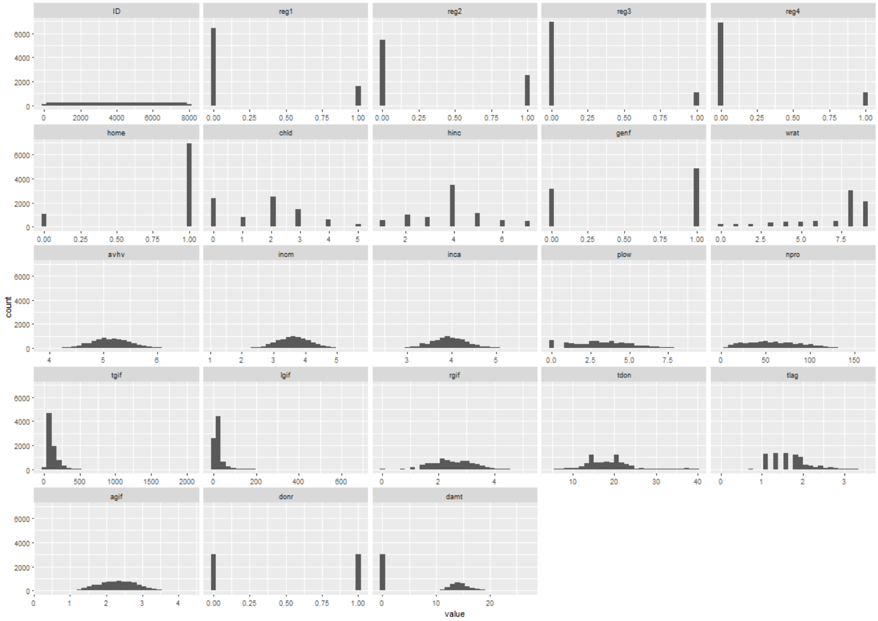
- The first 2 rows of variables (with the exception of ID) appear as if they could possibly be converted to factors/dummy variables due to the fact that they are discrete. This is discussed in more detail later in this section.
- The majority of the continuous variables are right skewed with some being more extreme than others. This makes them prime candidates for a log transformation.
- It can be seen in the histograms as well as the boxplots (which are not shown) that all of the outliers in the variables are in the right skews previously mentioned. It is assumed that the majority of these will be adequately controlled by the transformations.

Transformations

After viewing the distributions for each variable, it was clear that some obvious transformations were essential. 11 of the variables above were candidates for transformations due to their distributions:

- avhv, incm, inca, rgif, tlag, and agif were log transformed
- plow was considered for a log transformation, but the presence of 0 values made this difficult. Instead a square root transformation was used
- npro and tdon did not need a transformation
- tgif and lgif were considered for transformations as well as for making dummy variables, but the group ultimately decided to exclude them from being log transformed

After all transformations were performed, the distributions appeared normal and much easier to analyze.

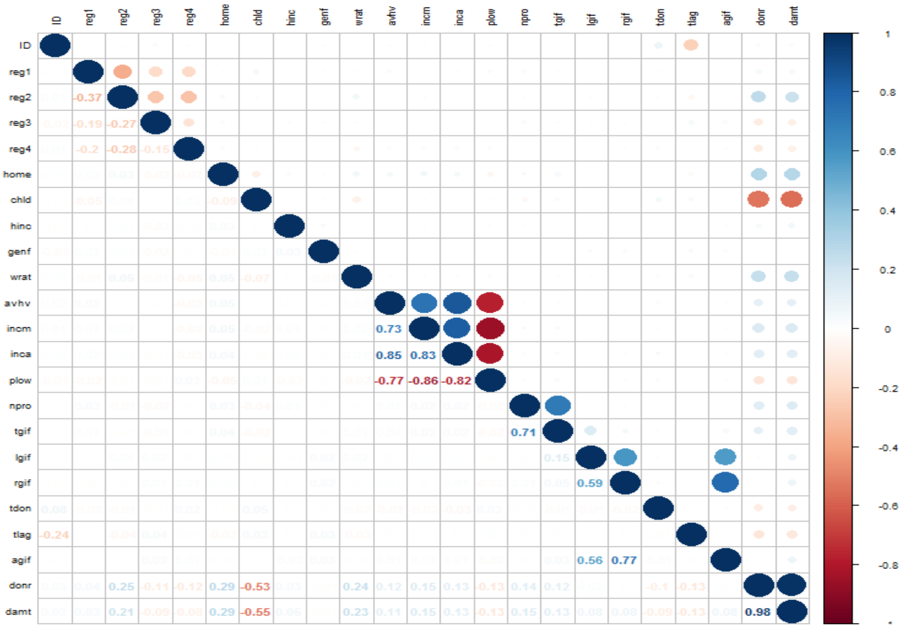


During the transformation process, there were 7 variables we thought would work better as factors. However, we still wanted to keep an all numeric dataset for use in certain models. We decided it would be best to create two different, yet consistent datasets to train models: one with all numeric, and one with seven of the numeric variables converted to factors. The factor dataset consisted of the same transformations used in the numeric dataset, with the addition of reg1, reg2, reg3, reg4, home, genf, and wrat converted to factors. During the model building process, we observed considerable improvement in some models when using the factor dataset over the numeric dataset.

Correlations and Frequencies

After all transformations were completed. We looked at the correlation of all the numeric variables (in the numeric dataset) as well the frequencies of the factors (from the factor dataset).

Correlation



The correlation plot shows the correlations of the predictor, and the response variables. Some noteworthy observations include:

- With a few exceptions, there is not a large amount of correlation among the predictor variables themselves. Avhv, incm, inca, and plow are all highly correlated with each other. This is expected because they are all focused on income/wealth. There are no other correlations among the predictor variables that are above .8.
- The response variables seem to have at least some correlation with the majority of the predictor variables. The largest correlation seen with both response variables in a negative correlation of chld. This correlation is only about -0.55.

Frequencies

	0	1	2	3	4	5	6	7	8	9
Reg1	6404	1605								
Reg2	5454	2555								
Reg3	6938	1071								
Reg4	6892	1117								
Home	1069	6940								
Genf	3161	4848								
Wrat	222	198	250	333	443	408	516	480	3021	2138

The frequency table illustrates a few large disparities in the different levels of some variables. As a group, we felt that they should all be kept as factors and used in the factor dataset. We decided that leaving all of the variables in both datasets, and to use variable selection methods in order to exclude variables with minimal interaction during the model building process.

Classification Models

To develop a method that would determine which potential mail recipients would be likely to donate, we built several classification models. We fit all prospective models using the training data, and evaluated those fitted models using the validation data. The goal is to develop a classification model for the ‘donr’ variable. In this section we will give a brief overview of each model and later, in the results section, we will compare the best models as well as choose our final model based on maximum profit.

Logistic Regression Model

The first and most basic classification model we built was the logistic regression model. In theory, this model is very similar to a normal linear regression model with the main difference being that instead of predicting a quantitative value, it calculates the probability that a particular observation belongs to a certain group. Logistic regression is used on responses that are binary, which fits our purpose well (with 1 being someone who is likely to respond to a mailing, and 0 being someone who is not likely to respond). For our particular model, 4 methods were used. Initially, we included all of the variables we had available. Then, we used each of the 3 selection methods (forward, backward, and stepwise) to determine if better results could be obtained. The best model based on maximum profit turned out to be the full model that is outlined below:

Logistic Full Model Accuracy Rate:	**85.3%**
Logistic Full Model Error Rate:	**14.7%**
Logistic Full Model - Predicted Number of Mailings:	**1,251**
Logistic Full Model - Predicted Profit:	**$11,650**

Linear Discriminant Analysis Model

Linear Discriminant Analysis is very similar to logistic regression except now we model the distribution of the predictors separately in each of the response classes. Then we use Bayes' theorem to incorporate them into our predictions. There are 3 main reasons to use LDA over logistic regression: 1) Logistic regression can be unstable when the classes are well separated, LDA is not, 2) if *n* is small and distributions are normal, LDA tends to be more stable than logistic, and 3) LDA is more widely used when there are more than two response classes. Since we have a rather large *n* and only two response classes, LDA was not expected to be one of our better models. For comparison's sake, we leveraged the same variables used in logistic regression and utilized them in the LDA model:

LDA Full Model Accuracy Rate:	**82.8%**
LDA Full Model Error Rate:	**17.2%**
LDA Full Model - Predicted Number of Mailings:	**1,103**
LDA Full Model - Predicted Profit:	**$11,646.50**

The LDA model performed similarly to the logistic model. It had a slightly lower accuracy rate and a slightly lower profit.

Quadratic Discriminant Analysis Model

An additional classification model that was evaluated is the Quadratic Discriminant Analysis (QDA) model. The qda() function is included in the MASS library of the R environment, and has nearly identical syntax with the LDA, LM, and GLM family of regression models. It assumes a quadratic decision boundary, and is primarily used to separate measurements of multiple classes into a quadric surface. The inner workings of the Quadratic Discriminant Model incorporate the likelihood ratio algorithm, and the objective is to determine if the quadratic form taken by the QDA model can, with higher fidelity, effectively capture the authentic relationship the linear approaches used by other relevant classification models. For this report, we analyzed and compared two separate QDA models.

For the initial model - QDA Model 1 - we included all 20 of the predictor variables. The model produced an error for the baseline model of 16.2%, and correctly classified 1,691 of the observations. In essence, the QDA model predictions were accurate 83.8% of the time. See table below:

QDA 1 Full Model Accuracy Rate:	**83.8%**
QDA 1 Full Model Error Rate:	**16.2%**
QDA 1 Full Model - Predicted Number of Mailings:	**1,390**
QDA 1 Full Model - Predicted Profit:	**$11,241.50**

For the second model - QDA Model 2 - we reduced the number of predictor variables to only include the ones that indicated somewhat of a relationship with the DONR dependent variable. This was achieved by reviewing the correlation matrix that was developed in the EDA portion of this report. The error rate for the second model is 21.8%, and correctly classified 1,578 of the observations. The QDA model predictions were accurate 78.2% of the time. See table below:

QDA 2 Model Accuracy Rate:	**78.2%**
QDA 2 Model Error Rate:	**21.8%**
QDA 2 Model - Predicted Number of Mailings:	**1,378**
QDA 2 Model - Predicted Profit:	**$11,280**

The results of the QDA model analysis is that adjusting the number of predictor variables in the model did not necessarily improve the prediction accuracy, nor lower the classification error rate. In summary, the baseline model that included all of the predictor variables performed very well - producing predictions that were accurate 84% of the time, and an expected profit of $11,241.50.

Tree-Based Models

We used several tree-based methods in our model-building process. A decision tree is a rather simple classification model to read, and interpret. Specifically, a person can simply start at the top of the tree and follow the “stems” down until reaching a “leaf”. The stems sort the observation into one of two groups to move the observation further down the tree. Once an observation reaches the end (leaf), it is classified accordingly. Each of these stems is created when the data is separated into non-overlapping boxes called regions. Each observation falls into only one of these regions. There are several methods to determine how to split the observations into regions including recursive binary splitting, and pruning. Although they are easy to interpret, a decision tree independently is not typically a competitive model. For this reason we also created a bagging, random forest, and boosting model for the charity organization.

Bagging

Bagging uses all the variables in the model and takes repeated samples (using bootstrapping) for each observation to create a relative number of decision trees. We then use the training data on each of these trees, and then average the conclusions together to get the correct predicted class. Whichever class an observation was sorted into more is the class the Bagging Model will predict it into.

Bagging Model Accuracy Rate:	**88.7%**
Bagging Model Error Rate:	**11.3%**
Bagging Model - Predicted Number of Mailings:	**1,038**
Bagging Model - Predicted Profit:	**$11,032**

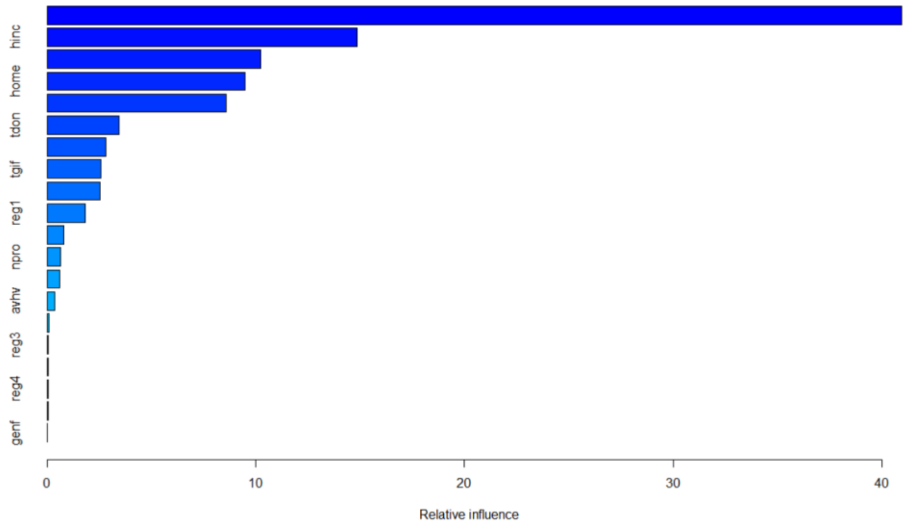
Random Forest

Random forest uses the exact same process as bagging with one exception: instead of using all variables in the trees that are created, only a random subset is used. This is to prevent the same (most important) variables from being used every time. By only using a subset, other variables get a chance to be stems higher up in the tree. After trying several different values for the number of variables, we chose 5 variables to be used.

RF Model Accuracy Rate:	**88.8%**
RF Model Error Rate:	**11.2%**
RF Model - Predicted Number of Mailings:	**1,053**
RF Model - Predicted Profit:	**$11,132.50**

Boosting

The final tree-based model we created was a boosting model. With boosting, we are again creating many trees that we can average, however this time the trees are built sequentially so that they build off of each other. This unique approach allows us to dramatically reduce the error. Our best boosting model was built using a Gaussian distribution, 5000 trees, a depth of 6, and a shrinkage value of 0.001. Along with creating the model, the gbm() function also outputs a plot of the relative influence each variable had in the tree-building process. In our case chld and hinc had the most influence. It is worth noting that chld was found to have the highest correlation with donr in our EDA.



Boosting Model Accuracy Rate:	**86.1%**
Boosting Model Error Rate:	**13.9%**
Boosting Model - Predicted Number of Mailings:	**1,271**
Boosting Model - Predicted Profit:	**$11,885.50**

Support Vector Machines Model

Support Vector Machine models classify a dataset by creating a boundary around each class of a response variable. In a perfect world, a linear boundary is created that separates all observations into the correct class. However, in most cases, this is very challenging to accomplish. Therefore, the cost parameter is used to allow a certain number of observations to be on the wrong side of this boundary in an effort of creating a better boundary for predicting new observations.

For our SVM model, we used the tune.out() function to select the best value of cost for us (5). We then built a linear SVM on the factor dataset using this value of cost. Compared to some of our other models, the error rate is similar but the profit obtained is significantly lower.

SVM Model Accuracy Rate:	**84.0%**
SVM Model Error Rate:	**16.0%**
SVM Model - Predicted Number of Mailings:	**1,822**
SVM Model - Predicted Profit:	**$10,551.50**

K-Nearest Neighbors Model

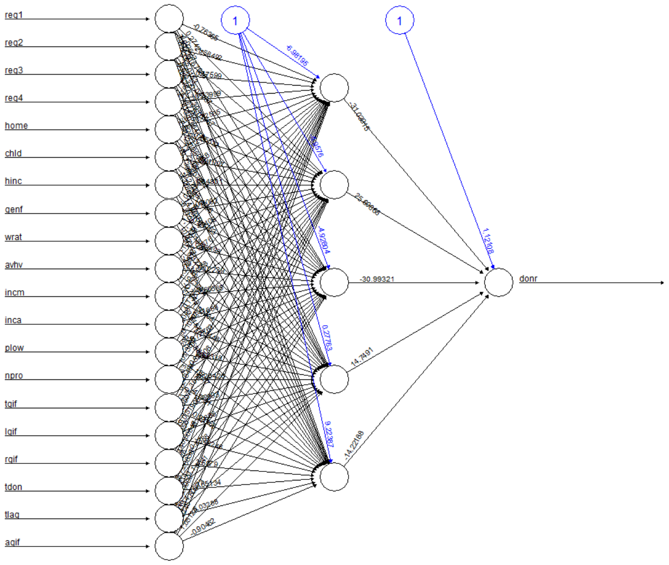
Additionally, we evaluated the K-Nearest Neighbors (KNN) classification model. The KNN model is a non-parametric approach for classification and regression. The results of the KNN model analysis revolves around the notion of identifying the optimal number of nearest neighbors to be included in the classifier. The knn() function is included in the ‘class’ library of the R environment. The KNN model is unique in that it fits the model, and creates a prediction in using just a single command. For this report, we created an R function that automated, and cross-validated the selection process of identifying the “K” value for the optimized number of nearest neighbors to be used by the classified model. The optimized number of neighbors for this data set is K=13.

For the model - KNN Model - The error for the baseline model is just below 8%, and correctly classified 1,858 of the observations. The KNN model predictions were accurate at an impressive 92.1% of the time. See table below for results:

KNN Model Accuracy Rate:	**92.1%**
KNN Model Error Rate:	**7.9%**
KNN Model - Predicted Number of Mailings:	**1,103**
KNN Model - Predicted Profit:	**$11,873.50**

Neural Network Model

An additional model that we evaluated is the Neural Network model. The neuralnet() and compute() functions are included in the 'neuralnet' library of the R environment. The neural net model uses a large collection of artificial neurons to emulate the process of how a living biological brain functions. As illustrated in the image below, each neuron (neural unit) is connected with many other neurons. For this report, our Neural Network model incorporated backpropagation to train the neural networks.



The results of the Neural Network model analysis revolves around the notion of identifying the optimal number hidden nodes, and also identifying the appropriate threshold value. We evaluated several neural network models, and decided to modify only two key model arguments to determine the model with the best fit: 1) the number of hidden neurons in each layer, 2) and threshold - numeric value specifying threshold for partial derivatives of the error function as stopping criteria. We evaluated various hidden neuron layers, including 3, 5, and 7. We also evaluated various threshold levels, including 0.1, 0.01, and 0.001.

NN Model Accuracy Rate:	**99.9%**
NN Model Error Rate:	**0.1%**
NN Model - Predicted Number of Mailings:	**1,291**
NN Model - Predicted Profit:	**$11,715**

Classification Model Performance Summary

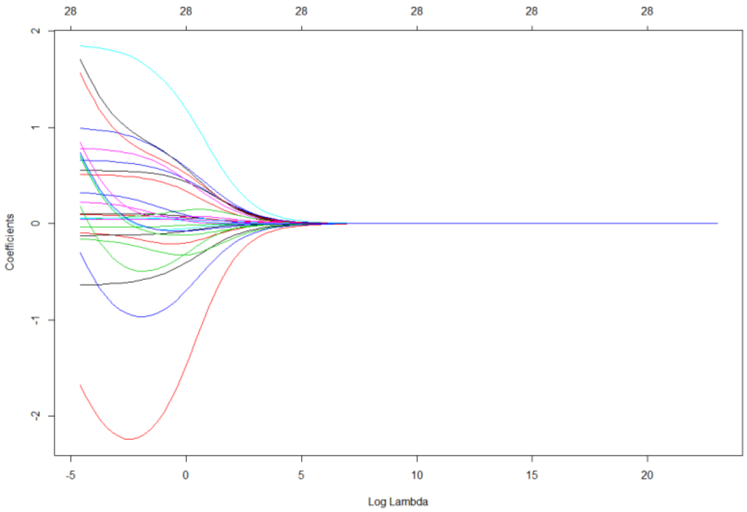
Classification Model	Accuracy	Mailings	Profits
Logistic	85.3%	1,251	$11,650.00
LDA	82.8%	1,103	$11,646.50
QDA	83.8%	1,390	$11,241.50
Bagging	88.7%	1,038	$11,032.00
Random Forest	88.8%	1,053	$11,132.50
Boost	86.1%	1,271	$11,885.50
SVM	84.0%	1,822	$10,551.50
KNN	92.1%	1,103	$11,873.50
Neural Network	99.9%	1,291	$11,715.00

Prediction Models

To develop a method that would determine of those who donated, how much they would donate, we built several prediction models. We fit all prospective models using the training data, and evaluated those fitted models using the validation data. The goal is to develop a prediction model for the 'damt' variable. In this section we will give a brief overview of each model and later, in the results section, we will compare the 3 best models as well as choose our final model based on lowest mean prediction error. In addition to the models covered in this section, we created a couple other models that we had limited experience in building: the GAM and RPART models. They performed similarly to a couple other models, and didn't show any signs of improvement over our best models; hence we ultimately decided to exclude the results from this report.

Ridge Regression Model

Ridge Regression works by shrinking the coefficients of the response variable so that they have less influence on the prediction. Ridge Regression does not perform any kind of variable selection so every variable will have at least some say in the model. The lambda (λ) value is the tuning parameter that decides how much the variables should shrink. The chart below demonstrates this:



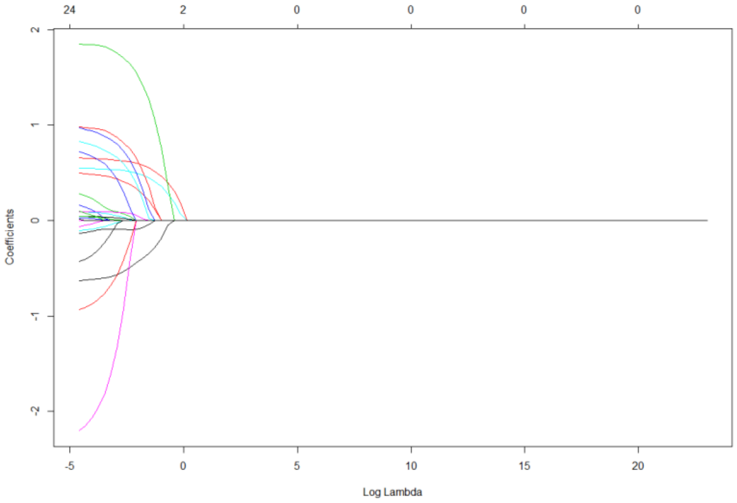
The chart shows that as lambda increases, the coefficients get smaller and smaller. It is difficult to conclude from this plot, but the coefficients never actually reach exactly 0. We attempted to build ridge models by using different λ chosen by 10-fold cross validation. We developed cross validation to determine the best value of lambda for our model (0.1224927537). The MSE and SD are shown below:

MSE: 1.521255

Std Error: 0.1571182

LASSO Model

The LASSO model is built very similarly to the Ridge Regression model. They major difference between the 2 is that LASSO allows the variable coefficients to reach 0, thus creating a form of variable selection. The chart below demonstrates this notion:



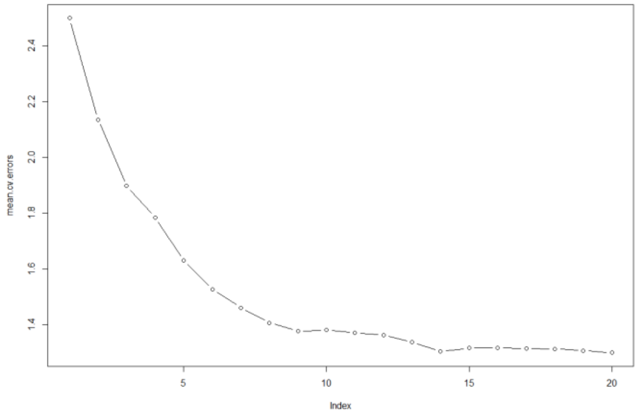
At a certain point, each variable coefficient goes to 0 and no longer has an effect on the model.
We again used cross validation for this model. Switching from Ridge Regression to LASSO in R was as simple as changing the value of the “alpha” parameter from 0 to 1. The MSE and SD are shown below:

MSE: 1.512279

Std Error: 0.1546922

Least Squares Model

For our least squares regression model, we first created a few preliminary models to get an idea of how it would perform. Next, we decided that instead of using all the variables, we would perform best subset selection. Best subset selection using 10-fold cross validation, indicated to us that the best model used all 20 variables. This is demonstrated in the plot below:



Instead of choosing the full model, we considered selecting the 14-variable model because it had a very similar MSE, but was simpler than the full model. However, when both the 14-variable and full model were run on the validation set, the full model performed better. The results below are from the full model:
MSE: 1.483712988
Std Error: 0.1533872925

When we ran a least squares regression on all of the predictor variables without using regsubsets() or cross validation, we actually ended up with a lower MSE:
MSE: 1.472403118
Std Error: 0.1541608612
Due to the lower MSE, we chose this model as our least squares model.

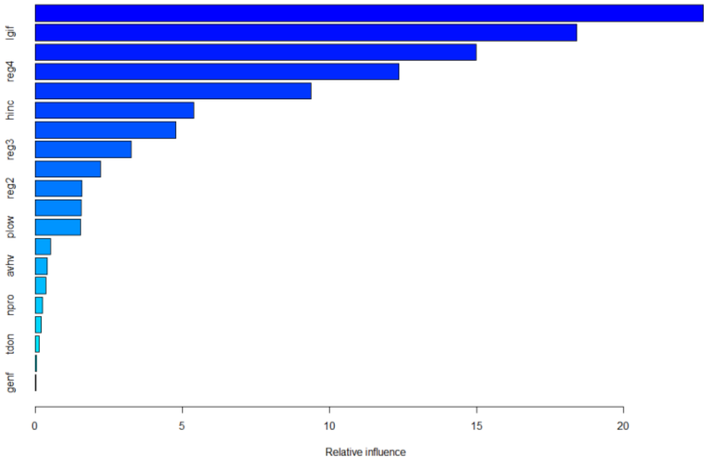
Tree-Based Models

Random Forest

This random forest model is similar to the one used for the classification model. We again used the randomForest() function included in the ‘randomForest’ library of the R environment. We tried several methods for computing the predictive random forest models, and ended up selecting 6 variables to be used at each split and 1000 trees to be built.
MSE: 1.64065585
Std Error: 0.1715645

Boosting

This boosting model is similar to the one constructed for the classification model. One noticeable difference is the influence the variables have on damt. In the classification model, chld was by far the highest influence. Looking at the chart below, we see that the top five variables are much closer in influence:

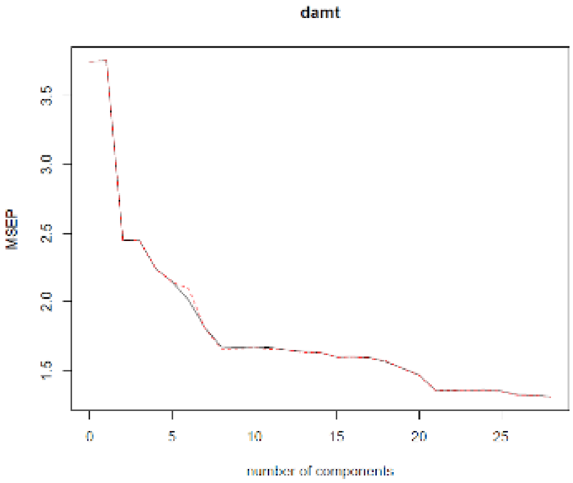


Rgif, lgif, and agif are the three top variables in the model. For this boosting model, we used 5000 trees, a Gaussian distribution, and an interaction depth of 4.

MSE: 1.539575259
Std Error: 0.1668604649

PCR - Principal Components Regression Model

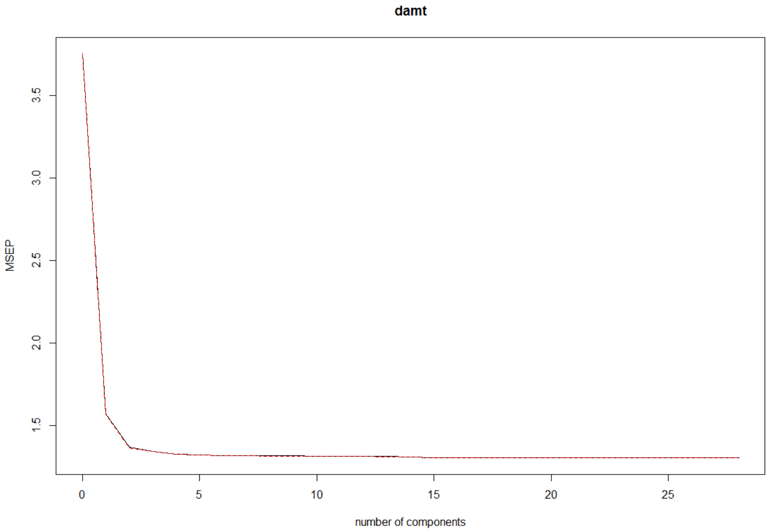
Principal Components Analysis is a dimension reduction method that looks to describe the variation contained in the variables in fewer principal components. The pcr() function is included in the 'pls' partial least squares library of the R environment. We found that the lowest cross validation error occurs when M= 28 components are used. This also gave us the lowest MSE on the validation set. Due to these factors, we decided that PCR was equivalent to the standard least squares model.



MSE: 1.472403118
Std Error: 0.1541608612

PLS - Partial Least Squares Model

Partial Least Squares is a dimension reduction method that attempts to find directions that explain not only the predictor variables, but the response as well. We performed partial least squares (PLS) using cross validation to determine the best model. We found that the lowest CV error occurred at M= 28 partial least squares directions. This means that essentially all variables are being used in the model. Looking at the plot below, however, the plot goes completely horizontal (no rate of change) at 18 components:



We used the 18 component model and observed the following results:
MSE: 1.472454
Std Error: 0.1541474.

Prediction Model Performance Summary

Prediction Model	MSE	Std Error
Ridge	1.521	0.157
LASSO	1.512	0.155
Least Squares	1.472	0.154
Random Forests	1.641	0.172
Boost	1.540	0.167
PCR	1.472	0.154
Partial Least Squares	1.472	0.154

RESULTS

The highest accuracy shown in the classification table above comes from the Neural Network model. However, since our primary objective in building these models was to maximize the profit for the charity organization, we selected the Boosting model because it produced the highest profits.

It can be seen in the predictive summary table above that our least squares, PCR, and Partial least squares all have the same MSE and Std Error. This is because they are essentially the same model. What PCR and Partial Least Squares are demonstrating is that in this case dimension reduction does not improve our predictive accuracy. Since this MSE also happens to be the lowest, we chose the least squares model as our final predictive model.

The in-depth analysis of the several classification and prediction models on the Charity data set resulted in the Boost classification model effectively capturing likely donors, and OLS predictive model effectively predicting expected gift amounts from donors. Based on our findings, a direct marketing campaign to 319 potential donors would cost $638.00, and result in profits of $3,990.80 for the charity organization. The models also concluded that the expected average donation of those predicted to be donors to be $14.51, which is a slight improvement from the previous average donation.

CONCLUSION

The Boost classification model, and the Ordinary Least Squares predictive model were the apex of this analysis, and clearly outperformed the other models.

The research, and analysis conducted in this report is not exhaustive, and focuses primarily on the models discussed in the course. However, the model development process established the fundamental framework necessary for future in-depth analysis. Nonetheless, the machine learning models, predictive model development process, and profit results can greatly benefit the direct mailing marketing campaigns for the charity organization. As future mailings are conducted, the data set can be updated, and the machine learning models can be recalibrated to better fit the charity data.

While working on this final project to satisfy the course requirement, we experienced many dynamics as a project team. We experienced some lessons learned regarding coding issues, the model development process, and team collaboration, and deepening our understanding of the R programming language. Some of the noteworthy coding issues included version control, and ensuring reproducible results for each of the models. Additionally, the model development process was unique in that each team member has unique understanding of the models, and also a varying command of the R statistical programming language. The culmination of the project experience revolved around communication, collaboration, and dedication. At times, the more experienced team members were required to mentor the less experienced team members on model expectations, and the utilization of R for machine learning. This further deepened the learning experience for the more experienced members of the team, and resulted in a practical method to apply the models to a real world problem.

SOURCES

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2015). *An introduction to statistical learning with applications in R*. New York, NY: Springer.