Northwestern University
Loan Default Prediction
Team Checkpoint 1

Shatabdi Choudhury
Dan Rubach
Rohan Mehta
Roza Salazar
Stephen Young
Justin Yun

NORTHWESTERN UNIVERSITY | SCHOOL OF PROFESSIONAL STUDIES

PREDICT 454: Advanced Modeling Techniques **Winter 2017**

Jennifer Wightman, PhD

Overview

The focus of this work is on the prediction of loan default followed by a loss amount which is conditional upon a default event. This work falls into the machine learning category of "supervised learning" as our data set includes a large number of features from which we seek to classify unseen out-of-sample loans as those that will and will not default. Default is a binary classifier which while not included in the data set is created based on loss values whereby we set default equal to one for those cases where a loss amount is greater than zero. We follow the Cross-Industry Standard Process for Data Mining ("CRISP-DM") life-cycle which starts with an understanding of the business problem, check of data quality and exploratory analysis, data preparation which may include transformations, imputations, and other modifications, followed by modeling, evaluation, and deployment.

It is important to note that this document is part of a larger project deliverable and is therefore limited to a description of the modeling problem, data inventory and quality check, and initial Exploratory Data Analysis ("EDA"). Future work will delve deeper into EDA, model fitting, and evaluation of the various approaches on out-of-sample test data.

The Modeling Problem

The modeling problem which we are faced with is to predict defaults and loss amounts on future unseen data records. Modeling defaults and losses is of paramount importance for banks, finance companies, and those that may purchase loan securitizations made up of a large number of individual loans. Loan default modeling is also of concern to regulators that seek to understand the quality of a bank's loan portfolio as these financial institutions are overseen by various agencies whose goal is to ensure the safety and soundness of individual entities and the financial system as a whole.

In contrast to unsupervised learning which may be focused on broad categorizations and will often have no desired particular output, supervised learning is used to learn-by-example with the goal of generalization to future cases. For example, supervised learning may be focused on classifying an email as "spam" or "not spam", a financial transaction as "fraudulent" or "not fraudulent", and a loan as likely to "default" or "not default." Supervised learners focused on classification include such modeling methods as logistic regression, linear discriminant analysis, decision trees, ensembles of decision trees which come together to form so-called random forests, and other approaches. Whichever approach, the overarching goal is the generalization of the model from in-sample fitting to out-of-sample test records and "real world" future use upon deployment. What generally varies from model-to-model is the modeling approaches ability to determine a decision boundary, or boundaries for classification problems which go beyond binary, that serve to separate the data into a respective category where in our case it is default or no default.

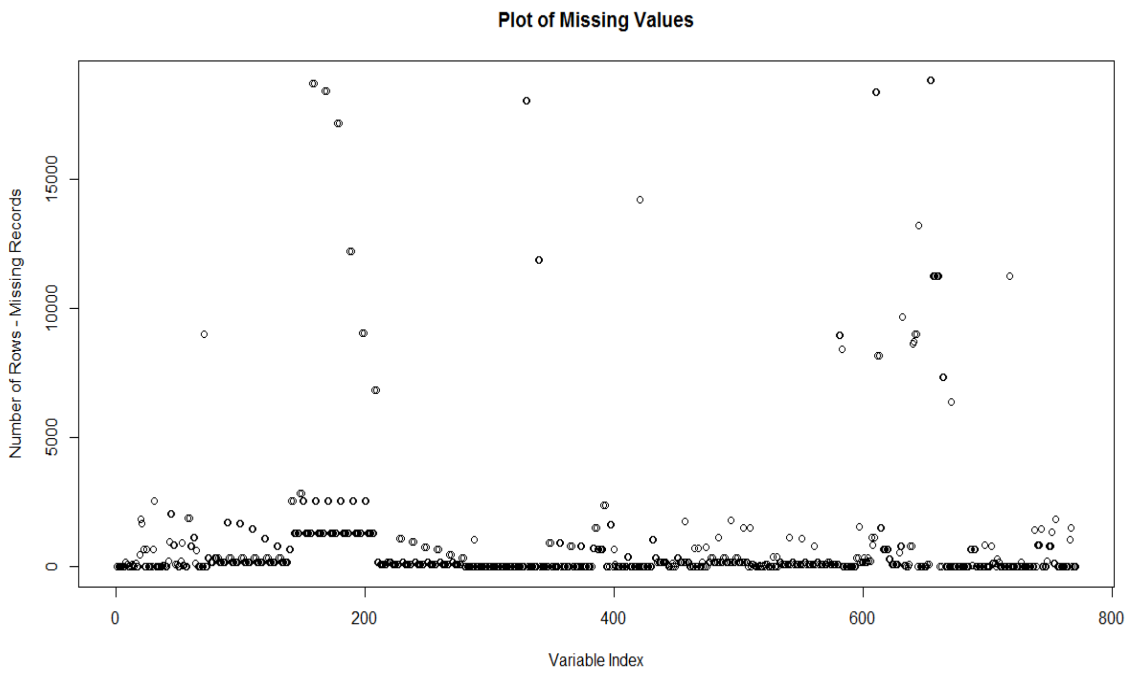
Thus, our modeling problem is well defined and is fully focused on the classification of loans to a binary outcome which includes default and no default. Further, beyond default we are concerned with the loss or severity of the outcome which is typically a fraction of the overall value of any loan as there is typically collateral (e.g. property) that may be liquidated and therefore result in recovery. When one combines the classification and subsequent loss the modeling effort is one of frequency and severity which are characteristic of default modeling. It is important to note that while frequency and even the outcome from select models (e.g. logistic regression which models log odds which are translated into a probability) may be continuous, in our modeling a loan will either be classified as one that will or will not default.

The Data - Data Inventory and Data Quality Check

The data consists of a set of anonymized financial transactions associated with individuals. The number of total records is 129,494 records of which there are 105,471 included in a file deemed training and 24,023 entries in a file labeled test for which the latter does not include a loss amount. That is, there are 771 columns included in the training data and only 770 in the test data set which does not include a loss amount. In the training set we use the loss amount to specify a binary classifier for default which takes the value of zero or one. Based on this, we have two values to predict which include the binary classifier associated with default and, in the event of default, a loss amount. Finally, given that there is a column labeled "id" there are 769 features which we may use to fit our models in the training set.

As we treat the test data as future unseen records, we focus herein on the training data. Out of the 105,471 records in the training data, there are 9,783 records for which there are positive loss amounts which constitutes 9.28% of the overall records. Similar to most large data files, there are missing values for several predictors for which we must ultimately decide to impute or discard the feature. In Figure 1 we provide for a plot that highlights those predictors which include missing values. Given the anonymized nature of the data, we do not know the exact nature of the predictors but we know that there are two features (i.e. f662 and f663 which may be redundant as the summary statistics for each are extremely similar) for which there are 17.86% of observations missing, 25 predictors which greater than 10% of their observations missing, and 43 where the missing values exceed 5%. It remains to be seen for which predictors we may choose to impute missing values as opposed to simply discarding the feature.

Figure 1. Plot of missing values for the training data set.



Beyond missing values, there are 9 features (i.e. f33, f34, f35, f37, f38, f678, f700, f701, and f702) for which the entire columnar values are zeros and two features whose values are constants (i.e. f736, and f764). As such, these 11 features will be not be used for modeling purposes.

Finally, we look for those predictors whose observations may include so-called outliers. We look for such outliers by computing the distance of the minimum and maximum values respectively, from the mean and normalize by dividing by the predictor's standard deviation. These values are analogous to z-scores and will point us towards those features whose minimum or maximum values may appear to be outliers. There are 53, 97, and 119 predictors respectively that have minimum values which are 10, 5, and 4 standard deviations from their means for which the largest deviation is approximately 143 standard deviations in distance from its minimum to its mean value. Similarly, there are 223, 465, and 533 predictors that have maximum values which are 10, 5, and 4 standard deviations from their means for which the largest is 233 standard deviations in distance from the maximum to the mean when normalized by the standard deviation.

Based on these cursory calculations we can be highly confident that there are a large number of predictors which have outliers or the series is highly non-normal.

EDA - Initial Exploratory Data Analysis

We begin our EDA by examining the distribution of loss amounts and through its relationship to defaults the prevalence of defaults in the training data. As previously mentioned, out of the 105,471 records in the training data, there are 9,783 records for which there are positive loss amounts which constitutes 9.28% of the overall records. In Figure 2 we present box plots of the loss amounts conditioned on no default (i.e. 0) and default (i.e. 1). As can be seen in Figure 2, loss amounts as a percentage of a loan amount and so bounded by zero and one are generally below 20% with a mean of 8.62%. This suggests that there is collateral such that upon a loss the recovery represents a sizeable amount of the overall loan. It is important to note the loss amount is right-skewed with a large number of values in the 0 – 20% range but the maximum loss is 100%. In Figure 3 we present a kernel density plot of the loss amount conditioned on the loss value exceeding zero. Figure 3 shows the pronounced right-skew associated with the loss value. To further explore the loss amount in Figure 4 we provide a kernel

Figure 2. Box plots of loss amounts conditioned on no default and default.

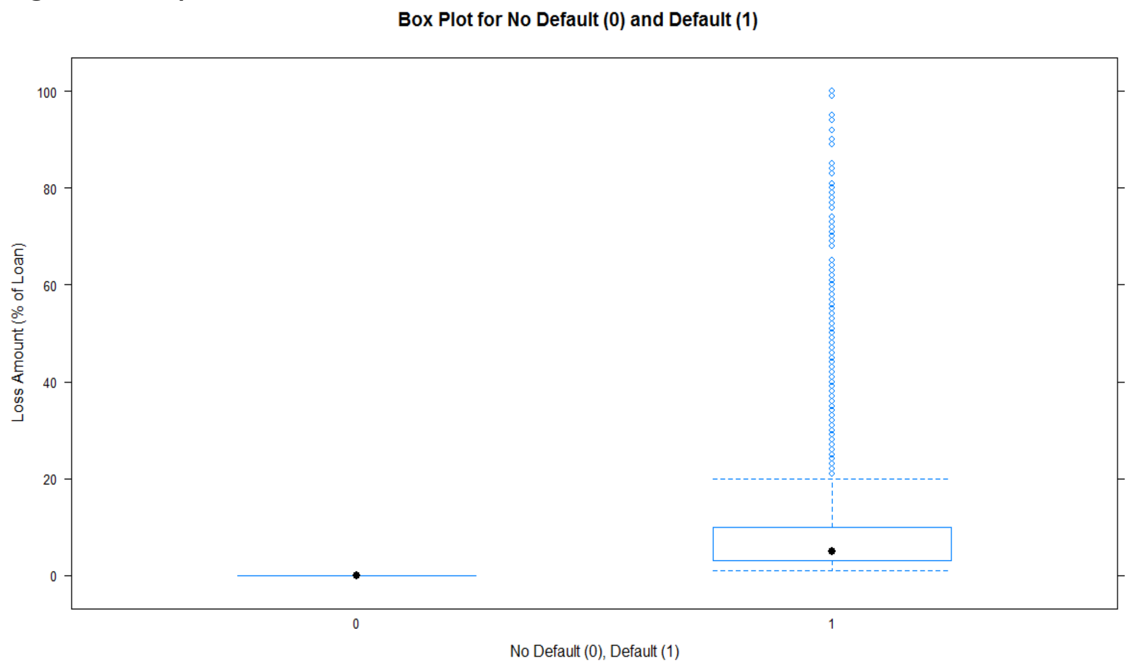
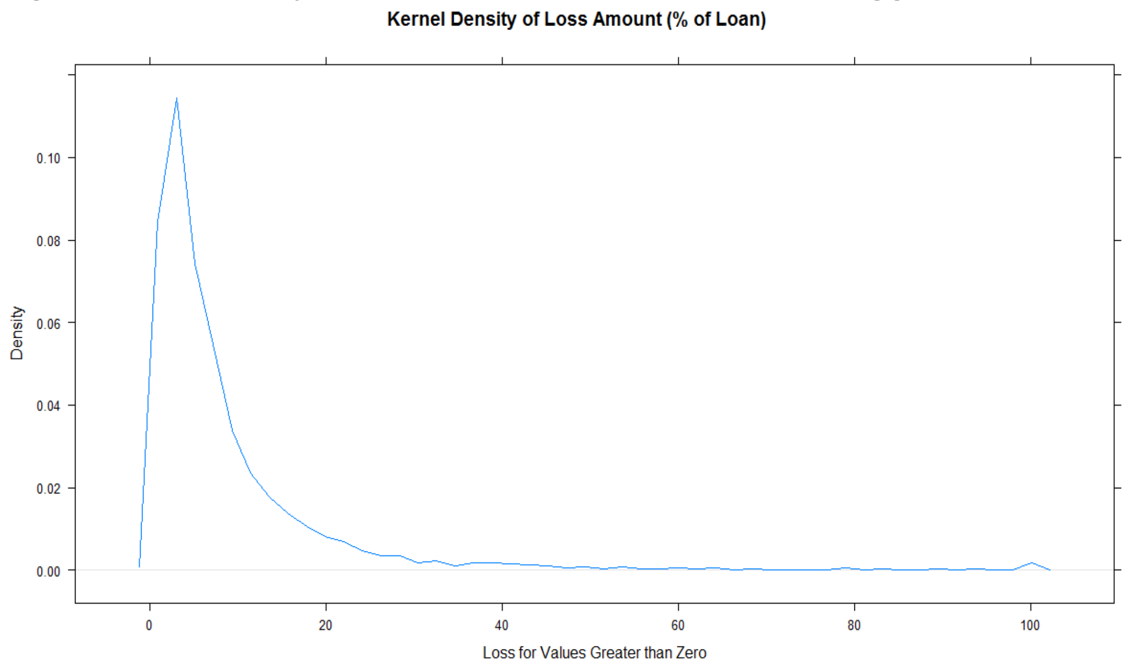
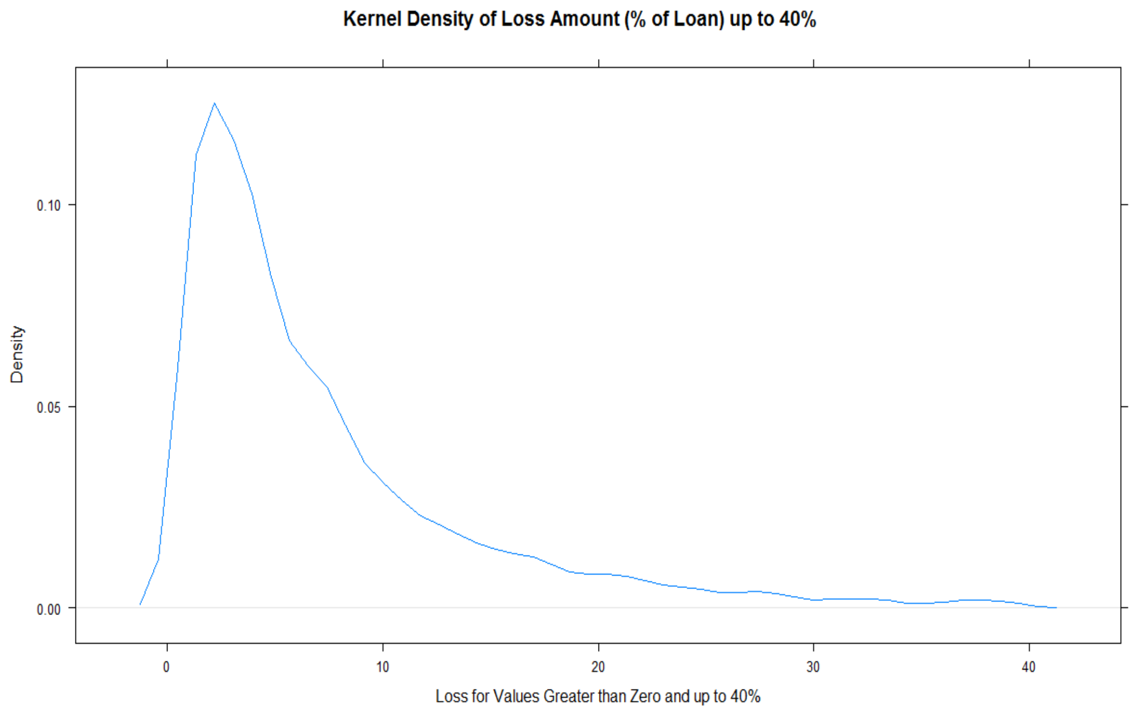


Figure 3. Kernel density of loss amount conditioned on the value being greater than zero.



density of the loss amount conditioned on the amount being greater than zero but less than 40%. As with Figure 3, Figure 4 shows the pronounced right-skew and further highlights that loss amounts are generally less than 20% which amounts to significant recovery rates.

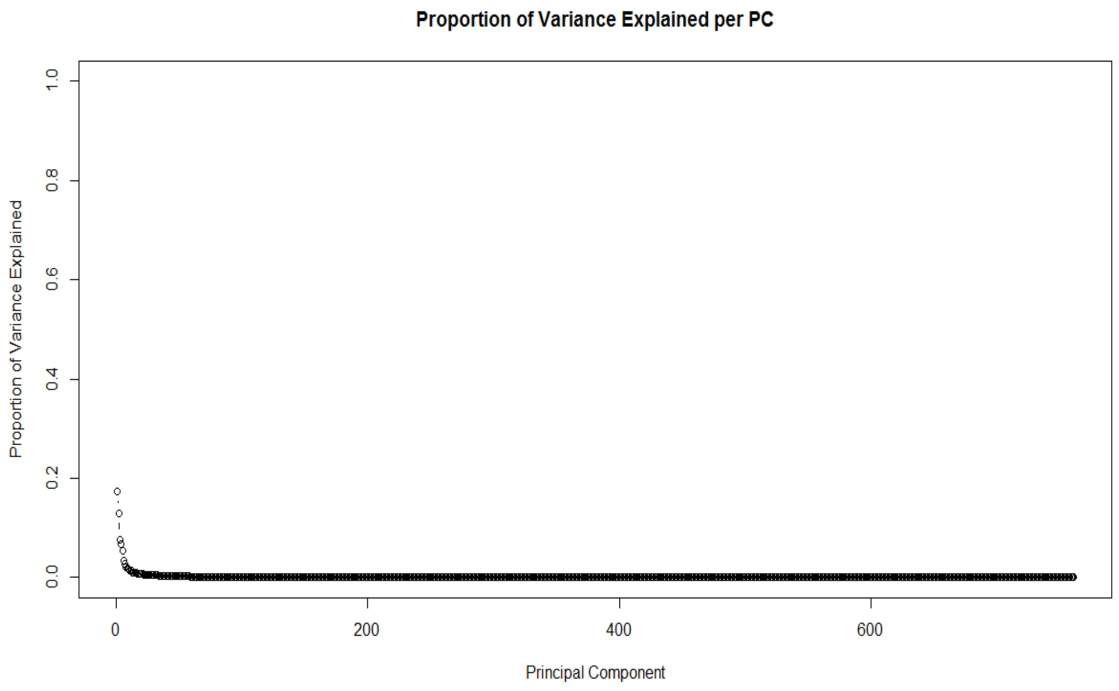
Figure 4. Kernel density of loss amount conditioned on the value being greater than zero and less than 40%.

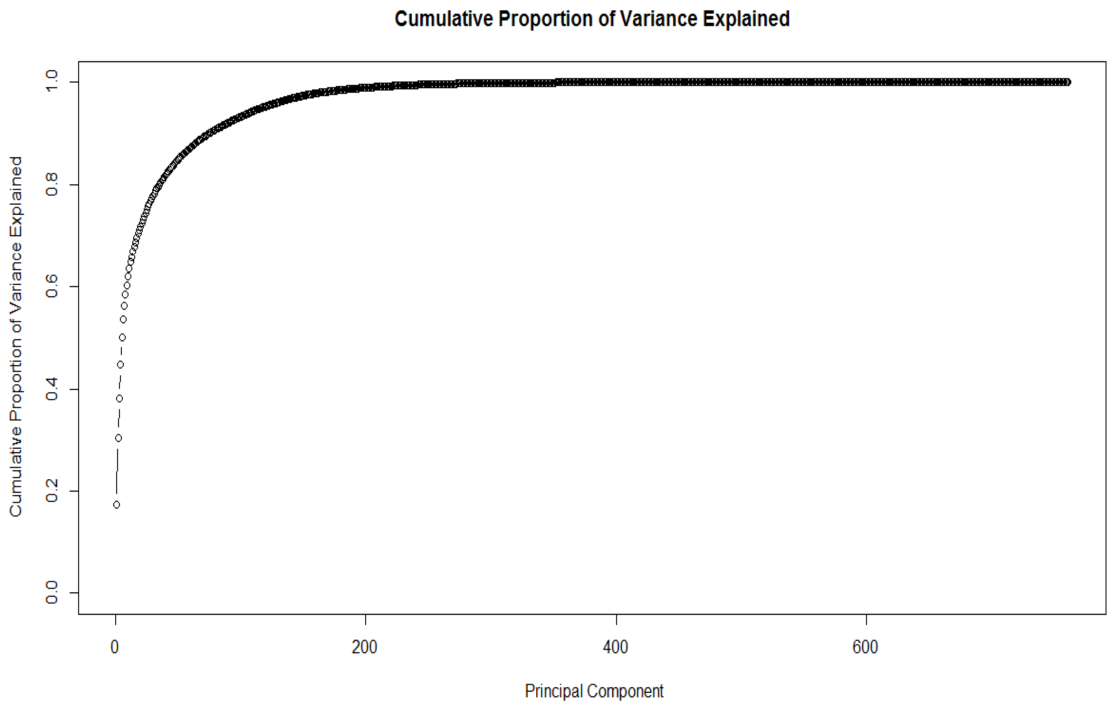


Given the large dimensionality of our feature space (i.e. 769 predictors), it is natural to explore whether this space may be reduced through Principal Components Analysis ("PCA"). PCA seeks to find the best low-dimensional representation of the variation of the data in a multivariate setting. In short, it is often possible to find a small number of principal components that may then be used to explain the variability of our response variable (i.e. in our case the loan status which we seek to classify) where by construction these components are orthogonal. We use PCA here to see if our predictor space can be reduced significantly to a few principal components that account for a sufficient amount of variability in the predictors. At times the components have an intuitive explanation but one drawback of PCA is that there may be no natural explanation. As is typical for PCA, we first standardize our predictors.

In Figure 5 we provide for a plot of the proportion of the variance explained by each Principal Component ("PC") and then the cumulative proportion of the variance which may be explained through successive accumulation of the PCs. As one can see, it may be feasible to reduce the dimensionality of the predictor space as the cumulative plot reaches 80% for a small number of components especially when compared to the 769 predictors that characterize the entire feature space.

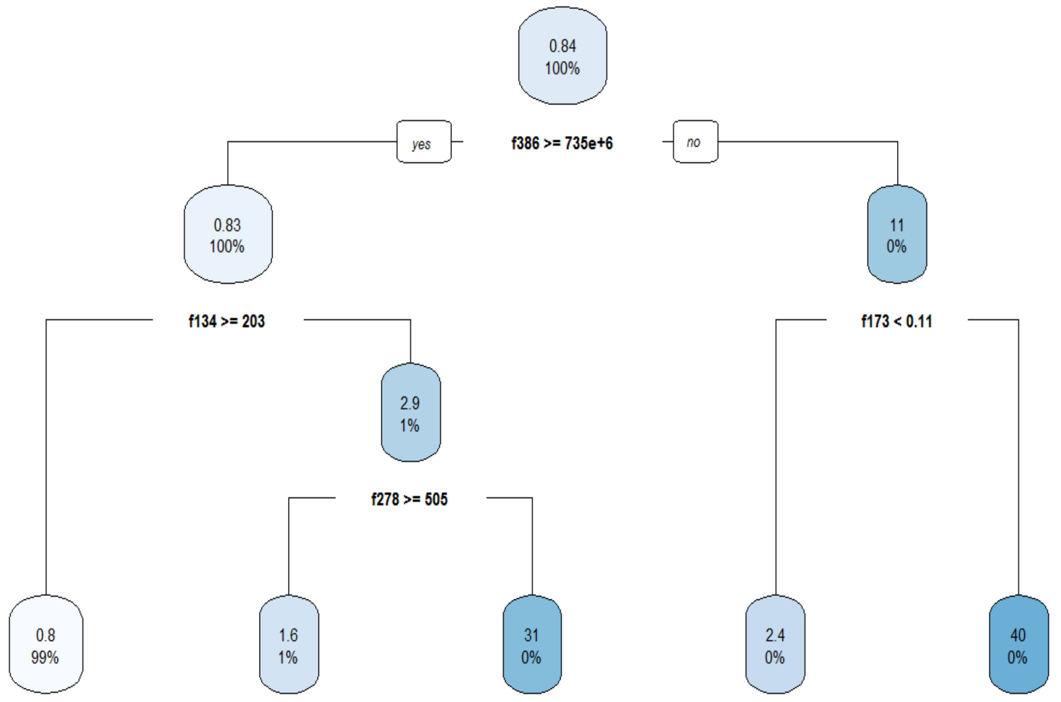
Figure 5. Proportion of variance explained by the Principal Components ("PCs") and the cumulative proportion of the variance explained through successive PCs.





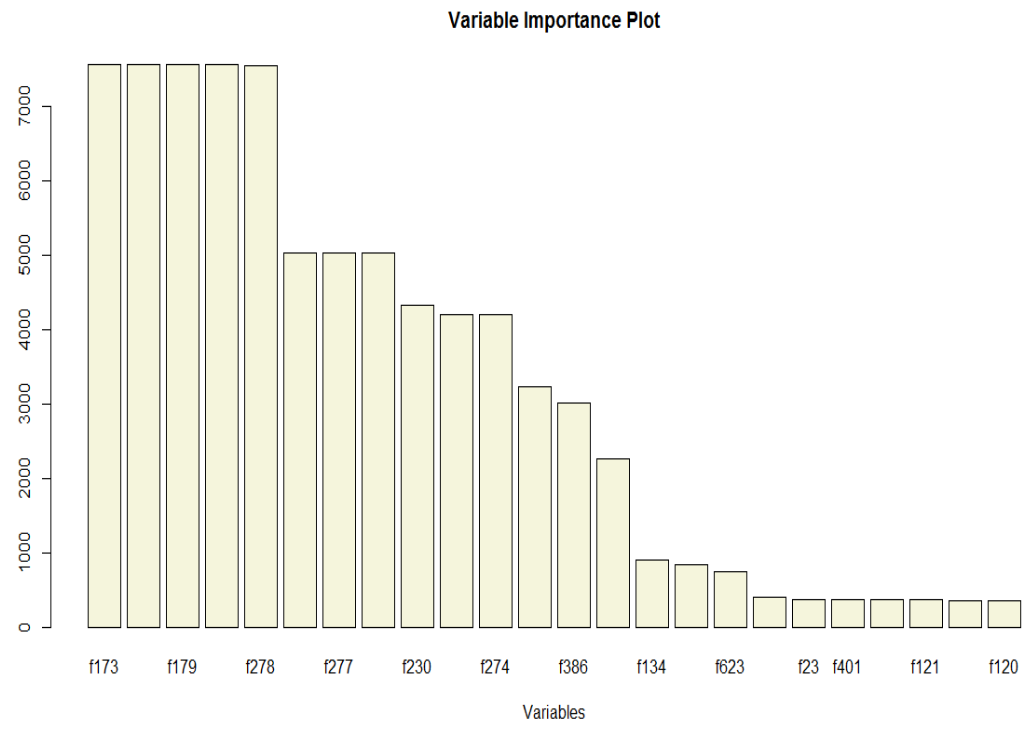
A decision tree is a multistage classification approach made up of nodes where the terminal nodes representing a class label. The splitting that occurs at a node is driven by a measure such as Entropy, Gini, or Classification error where each are values of impurity where one seeks to minimize this upon a split. The induction process proceeds based on measuring the information gained by making successive splits until all attributes have been taken account of or the increase in information is below some threshold. There are various algorithmic approaches to constructing a decision tree and we leverage those available in the rpart R package. In Figure 6 we present a plot of a decision tree that was fit on a portion of the overall training data set where we have used loss as a target or response variable. As one can see from the below, are data is first split on predictor f386 and followed by splits on f134 and f278 on one side and f173 on the other. As such, we can be reasonably confident that these predictors will prove fruitful in modeling default which is our binary target.

Figure 6. Decision tree based on a portion of the overall training data set with loss as target.



Finally, in Figure 7 we provide for a bar plot that is focused on variable importance from our predictor sp ace and model fit on the decision tree. Variables that are highlighted as important include f173, f178, f17 9, f180, f278, f276, f277, f589, f230, f273, f274, f171, f386, f622, f134, f387, f623, f139, f23, f401, f114, f121, f119, and f120.

Figure 7. Variable importance plot from a decision tree fit on a portion of the training data with loss as the target variable.



Conclusions

While our modeling task is clear, our initial analysis of the data suggests that we must further explore for outliers and non-normality and possibly truncate values through Winsorization and transform variables where applicable. It is clear that there are several features that may be removed from our data set and modeling task associated with the classification of loans and estimation of loss amounts in the event of default. Finally, as per our PCA, it is evident that the feature space may be reduced yet still account for a significant percentage of the variability of overall predictor space.

Appendix – R Code

```
# Northwestern University

# PREDICT 454

# Loan Default Prediction



# Install packages

# Can comment out (#) if installed already

install.packages("pastecs")

install.packages("psych")

install.packages("lattice")

install.packages("plyr")

install.packages("corrplot")

install.packages("RColorBrewer")

install.packages("caTools")

install.packages("class")

install.packages("gmodels")

install.packages("C50")

install.packages("rpart")

install.packages("rpart.plot")

install.packages("modeest")

install.packages("randomForest")
```

```
# Libraries used for analysis
suppressMessages(library(pastecs))
suppressMessages(library(psych))
suppressMessages(library(lattice))
suppressMessages(library(plyr))
suppressMessages(library(corrplot))
suppressMessages(library(RColorBrewer))
suppressMessages(library(caTools))
suppressMessages(library(class))
suppressMessages(library(gmodels))
suppressMessages(library(C50))
suppressMessages(library(rpart))
suppressMessages(library(rpart.plot))
suppressMessages(library(modeest))
suppressMessages(library(randomForest))

# Get list of installed packages as check
search()

# Variable names start with Capital letters and for two words X_Y
# Examples: Path, Train_Data, Train_Data_Split1, etc.

# Set working directory for your computer
# Set path to file location for your computer
# Read in data to data file
setwd("C:/Users/Stephen D Young/Documents")
```

```
Path <- "C:/Users/Stephen D Young/Documents/Stephen D. Young/Northwestern/Predict
454/Project/Train Data/train_v2.csv"
Train_Data <- read.csv(file.path(Path,"train_v2.csv"), stringsAsFactors=FALSE)

# Get structure and dimensions of data file
str(Train_Data)
dim(Train_Data)

# Characteristics of data frame from above
# 105471 rows(records), 771 columns, 770 not including loss which is last column
# there is no indicator variable for default but can be 0 or 1 for when there is
# an observed loss.  So we can create binary default flag.

# Summary statistics for loss variable
# options used to define number and decimal places
options(scipen = 100)
options(digits = 4)
summary(Train_Data$loss)
# Get mode using mlv from modeest package
mlv(Train_Data$loss, method = "mfv")

# Results of loss are min = 0, max = 100, mean = .8, mode = 0
# Loss is not in dollars but as percent of loan amount

# Check for missing values in loss column
sum(is.na(Train_Data$loss))
```

```
# There are no missing values in loss column

# Make first working data set and add in column for validation set split
Train_Data_Split1 <- Train_Data
Train_Data_Split1$split <- round(runif(n=nrow(Train_Data_Split1)),3)

# Add in default indicator and determine number and proportion of defaults
# Use ifelse to set default to 1 if loss > 0 otherwise default is 0
# Get table of 0 and 1 values (i.e. no default, default)
Train_Data_Split1$default <- ifelse(Train_Data_Split1$loss>0,1,0)
table(Train_Data_Split1$default)

# Results are:
#     0     1
# 95688  9783
# 95,688 + 9,783 = 105,471 which is original number of records

# Get percentage of defaults
sum(Train_Data_Split1$default)/length(Train_Data_Split1$default)
# Percentage of defaults = 0.09275535

# Box plot and density plot of loss amount
# Lattice package for bwplot and densityplot
# Boxplot is for no default (0) and default (1)
# Boxplot should have zero loss for no default and range of loss upon default
bwplot(loss~factor(default), data=Train_Data_Split1,
      main = "Box Plot for No Default (0) and Default (1)",

      xlab = "No Default (0), Default (1)",

      ylab = "Loss Amount (% of Loan)")


# Density plot of loss should skew right as 100% loss is max but uncommon

densityplot(~(loss[loss>0]), data = Train_Data_Split1, plot.points=FALSE, ref=TRUE,

          main = "Kernel Density of Loss Amount (% of Loan)",

          xlab = "Loss for Values Greater than Zero",

          ylab = "Density")


# Density plot of loss should skew right even with 40% as max value for plot

densityplot(~loss[loss>0 & loss<40], data = Train_Data_Split1, plot.points=FALSE, ref=TRUE,

          main = "Kernel Density of Loss Amount (% of Loan) up to 40%",

          xlab = "Loss for Values Greater than Zero and up to 40%",

          ylab = "Density")


# Basic summary statistics for loss values when greater than zero

# Get mode using mlv from modeest package

summary(Train_Data_Split1$loss[Train_Data_Split1$loss>0])

mlv(Train_Data_Split1$loss[Train_Data_Split1$loss>0], method = "mfv")

# Results of loss are min = 1, max = 100, mean = 8.62, mode = 2


# Function to compute summary statistics

myStatsCol <- function(x,i){


  # Nine statistics from min to na
  mi <- round(min(x[,i], na.rm = TRUE),4)

  q25 <- round(quantile(x[,i],probs = 0.25,  na.rm=TRUE),4)

  md <- round(median(x[,i], na.rm = TRUE),4)

  mn <- round(mean(x[,i], na.rm = TRUE),4)

  st <- round(sd(x[,i], na.rm = TRUE),4)

  q75 <- round(quantile(x[,i], probs = 0.75, na.rm = TRUE),4)

  mx <- round(max(x[,i], na.rm = TRUE))

  ul <- length(unique(x[complete.cases(x[,i]),i]))

  na <- sum(is.na(x[,i]))


  # Get results and name columns

  results <- c(mi, q25, md, mn, st, q75, mx, ul, na)

  names(results) <- c("Min.", "Q.25", "Median", "Mean",

                      "Std.Dev.", "Q.75", "Max.",

                      "Unique", "NA's")

  results

}


# Call to summary statistics function

# There are 9 summary statistics and 769 predictors excluding id and loss

Summary_Statistics <- matrix(ncol = 9, nrow = 769)

colnames(Summary_Statistics) <- c("Min.", "Q.25", "Median", "Mean",

                   "Std.Dev.", "Q.75", "Max.",

                   "Unique", "NA's")

row.names(Summary_Statistics) <- names(Train_Data_Split1)[2:770]
```

```
# Loop for each variable included in summary statistics calculation
for(i in 1:769){
   Summary_Statistics[i,] <- myStatsCol(Train_Data_Split1,i+1)
}
# Create data frame of results
Summary_Statistics <- data.frame(Summary_Statistics)
# View select records for reasonableness
head(Summary_Statistics,20)

# Output file of predictor variable summaries
write.csv(Summary_Statistics, file = file.path(Path,"myStats_Data1.csv"))

# Creates summary table from which we get n which is number of missing
# records (i.e. Summary$n)
Summary <- describe(Train_Data, IQR = TRUE, quant=TRUE)

# Calculate number of missing records for each variable
Missing_Var <- nrow(Train_Data) - Summary$n
plot(Missing_Var, main = "Plot of Missing Values", xlab = "Variable Index",
    ylab = "Number of Rows - Missing Records")

# Get variable names missing more than 5000 records as we may want to remove
# those with n missing > 5000 as imputation could be problematic
Lot_Missing <- colnames(Train_Data)[Missing_Var >= 5000]
print(Lot_Missing)
```

```
# Variables for which n missing > 5,000

# [1] "f72"  "f159" "f160" "f169" "f170" "f179" "f180" "f189" "f190" "f199"

# [11] "f200" "f209" "f210" "f330" "f331" "f340" "f341" "f422" "f586" "f587"

# [21] "f588" "f618" "f619" "f620" "f621" "f640" "f648" "f649" "f650" "f651"

# [31] "f653" "f662" "f663" "f664" "f665" "f666" "f667" "f668" "f669" "f672"

# [41] "f673" "f679" "f726"


# Find columns where all values are the same as we can remove these

Zero_Cols <- rownames(Summary)[Summary$range==0 & Summary$sd == 0]

print(Zero_Cols)

# [1] "f33"  "f34"  "f35"  "f37"  "f38"  "f678" "f700" "f701" "f702" "f736"

# [11] "f764"


# Get number of records that have no missing predictor variables

Complete_Records <- Train_Data[complete.cases(Train_Data),]

print(Complete_Records)

# If one creates a data set that has complete records for all predictors it

# would have 51,940 rows or 49% of total records (51,940/105,471 = .49)


# Principal Components Analysis (PCA) for Dimensionality Reduction

# Remove constant, columns with all zeros, and loss for PCA rescale

Train_Data_Adj <- Train_Data_Split1[,!names(Train_Data_Split1) %in% c("loss",Zero_Cols)]


# Run PCA removing observations with missing data as first pass and scaling

# which is to unit variance

PCs <- prcomp(Train_Data_Adj[complete.cases(Train_Data_Adj),], scale=TRUE)
```

```
#Variance explained by each principal component

PCs_Var <- PCs$sdev^2


#Proportion of variance explained

Prop_Var_Expl <- PCs_Var / sum(PCs_Var)


# Plot of proportion of variance explained and cumulative proportion of variance explained

plot(Prop_Var_Expl, main = "Proportion of Variance Explained per PC", xlab="Principal Component",

ylab="Proportion of Variance Explained", ylim=c(0,1),type='b')

plot(cumsum(Prop_Var_Expl), main = "Cumulative Proportion of Variance Explained",xlab=" Principal

Component", ylab ="Cumulative Proportion of Variance Explained", ylim=c(0,1),type='b')

# Based on PCA may be able to reduce features to ~ 200


# Decision tree using rpart, rpart.plot and variable importance

# Decision tree on data but without standardization

# Note that loss is response but we should convert to default and predict

# loss amounts post prediction of defaults which are binary classifier

multi.class.model <- rpart(loss~., data=Complete_Records[1:20000, 1:771])

rpart.plot(multi.class.model)

summary(multi.class.model)

multi.class.model$variable.importance


barplot(multi.class.model$variable.importance, main="Variable Importance Plot",

    xlab="Variables", col= "beige")
```