Divorce in America

Predictive Modeling Project

Samantha Hair & Ezequiel Carrillo

MA 5790 February 25, 2022

Abstract:
Divorce is a social issue which involves the legal dissolution of a matrimonial union between two people. The result of divorce has repercussions beyond the confines of the home. However, marriage contributes positively to society by building and strengthening human relationships within the home (among spouses and children) and beyond (involving extended family, friends, and workplaces). Marriage therefore is the bedrock of a prosperous society, from which the future generations are brought forth and introduced to society as the next generation of productive and responsible citizens. On the contrary, children who experience the separation of their parents are not equally equipped with the tools and resources to achieve success in all aspects of their lives. This can be due to having less emotional and financial support than their peers whose parents are not divorced. It has long been recognized that children from divorced families are disadvantaged in many regards from their peers who have not experienced divorce in the home. It is then, in the noble interest of this project to accurately capture future samples in the hopes of recommending a remedial marriage therapy to couples at risk of divorce. Therefore, it is the goal of this project to build statistical models that accurately predict divorce based on a dataset of fifty-four predictors.

Table of Contents

1 Background

Divorce rates in America have increased three-fold in the past three decades alone, with no clear signs of slowing down. As discussed previously, divorce deteriorates the fabric of society by ungraciously interrupting the rhythm of a household's daily routine. The ever-rippling effects of divorce can be felt not only by children but also by communities, workplaces, and society at large. Therefore, at the heart of this study, our main goal is to build statistical models that are properly trained and tuned, such that the predictive ability of our models can capture and accurately predict whether a relationship will end in divorce. A potential use case of our project could be to integrate our models into a system where a couple can assess their likelihood of divorce and go to counseling if the response variable predicts divorce.

2 Variable Introduction and Definitions:

The data has been provided by Kaggle (https://www.kaggle.com/csafrit2/predicting-divorce). The response variable is a factor ('Y/N') which denotes whether or not a marriage will result in divorce based on the predictor values. There are 54 predictors and 170 samples. The predictors are all categorical in nature, ranging from values between (0 - 4) with 0 being the lowest and 4 being the highest. Below is a compiled list of the variable names (both response and predictors) along with their descriptions. The data was collected in 2014 by Dr. John Gottam through his couples therapy research.

Variable Name	***Description***
Sorry_end	If one of us apologizes when our discussion deteriorates, the discussion ends.
Ignore_diff	I know we can ignore our differences, even if things get hard sometimes.
begin_correct	When we need it, we can take our discussions with my spouse from the beginning and correct it.
Contact	When I discuss with my spouse, contacting him will eventually work.
Special_time	The time I spent with my wife is special for us.
No_home_time	We don't have time at home as partners.
X2_strangers	We are like two strangers who share the same environment at home rather than family.
enjoy_travel	I enjoy our travel with my spouse.
enjoy_holiday	I enjoy our holidays with my wife.
common_goals	Most of our goals are common to my spouse.
harmony	My spouse and I have been in harmony with each other.
freeom_value	My spouse and I have similar values in terms of personal freedom.
entertain	My spouse and I have a similar sense of entertainment.
people_goals	Most of our goals for people (children, friends, etc.) are the same.
dreams	Our dreams with my spouse are similar and harmonious.
love	We're compatible with my spouse about what love should be.
happy	We share the same views about being happy in our life with my spouse
marriage	My spouse and I have similar ideas about how marriage should be
roles	My spouse and I have similar ideas about how roles should be in marriage
trust	My spouse and I have similar values in trust.
likes	I know exactly what my spouse likes.

care_sick	I know how my spouse wants to be taken care of when she/he is sick.
fav_food	I know my spouse's favorite food.
stresses	I can tell you what kind of stress my spouse is facing in her/his life.
inner_world	I have knowledge of my spouse's inner world.
anxieties	I know my spouse's basic anxieties.
current_stress	I know what my spouse's current sources of stress are.
hopes_wishes	I know my spouse's hopes and wishes.
know_well	I know my spouse very well.
Aggro_argue	I feel aggressive when I argue with my spouse.
Always_never	When discussing with my spouse, I usually use expressions such as 'you always' or 'you never' .
negative_personality	I can use negative statements about my spouse's personality during our discussions.
offensive_expressions	I can use offensive expressions during our discussions.
insult	I can insult my spouse during our discussions.
humiliate	I can be humiliating when we discussions.
not_calm	My discussion with my spouse is not calm.
hate_subjects	I hate my spouse's way of opening a subject.
sudden_discussion	Our discussions often occur suddenly.
idk_what.s_going_on	We're just starting a discussion before I know what's going on.
calm_breaks	When I talk to my spouse about something, my calm suddenly breaks.
argue_then_leave	When I argue with my spouse, ı only go out and I don't say a word.
silent_for_calm	I mostly stay silent to calm the environment a little bit.
good_to_leave_home	Sometimes I think it's good for me to leave home for a while.
silent_instead_of_discussion	I'd rather stay silent than discuss with my spouse.
silent_for_harm	Even if I'm right in the discussion, I stay silent to hurt my spouse.
silence_fear_anger	When discussing, I stay silent because I am afraid of not being able to control my anger.
I.m_right	I feel right in our discussions.
accusations	I have nothing to do with what I've been accused of.
I.m_not_guilty	I'm not actually the one who's guilty about what I'm accused of.
I.m_not_wrong	I'm not the one who's wrong about problems at home.
no_hesitancy_inadequate	I wouldn't hesitate to tell my spouse about her/his inadequacy.
you.re_inadequate	When I discuss, I remind my spouse of her/his inadequacy.
incompetence	I'm not afraid to tell my spouse about her/his incompetence.
friends_social	I know my spouse's friends and their social relationships.
Divorce_Y_N	The variable that is the outcome, is Divorce (Y/N)

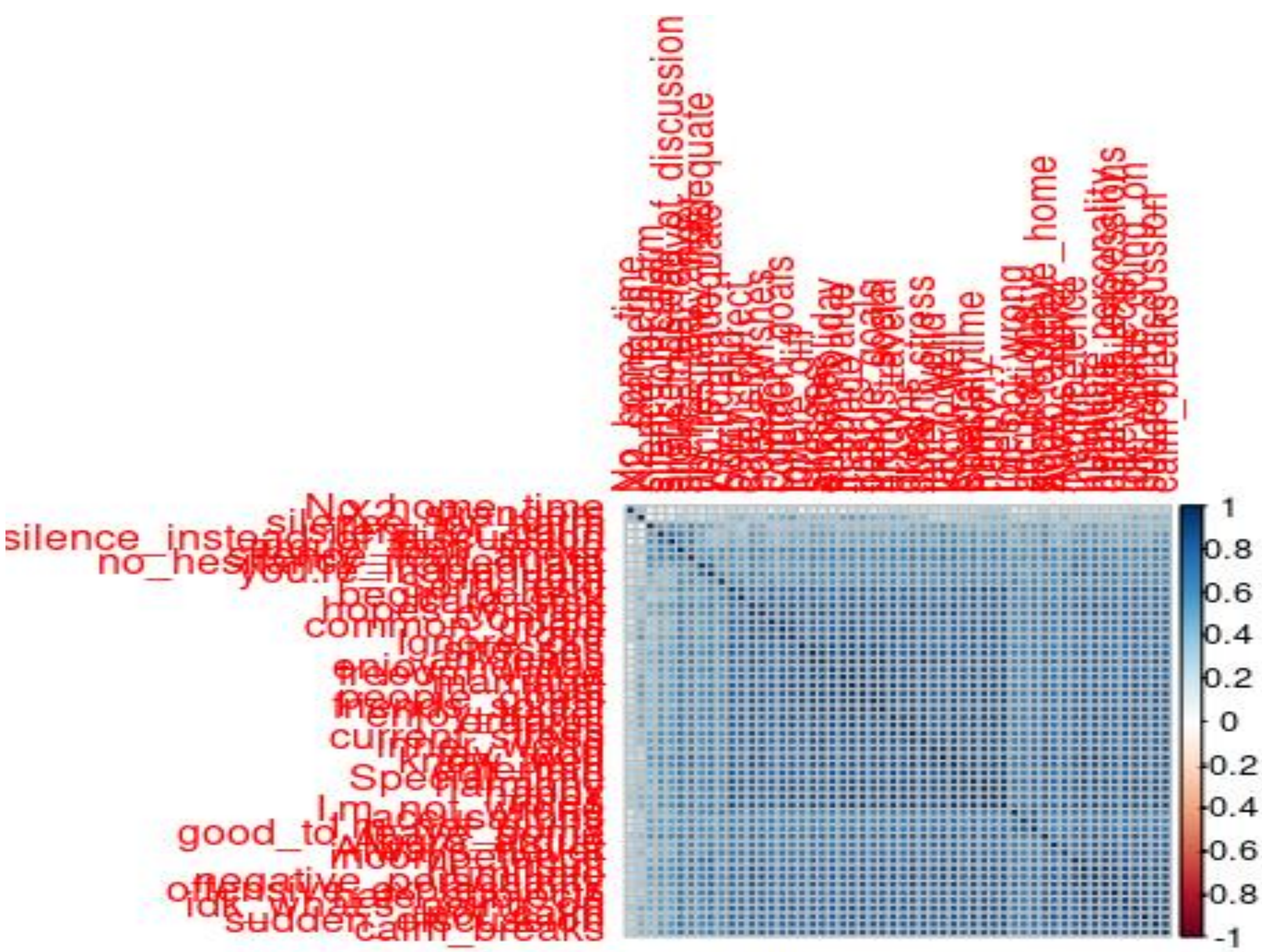
Given the predictor variables, we will perform appropriate data wrangling in order to build our models on data which is suitable for training and testing the model performance. These data cleaning techniques include predictor transformations such as adding and removing predictors, creating linear combinations of predictors, or modifying the structure of the data if need be. Consequently, we will build, train, tune, and evaluate the predictive models and make a selection based on the top two most accurate and efficient models. The classification metric we will be using is Accuracy, as it gives a general idea of how the model is correctly capturing the true negatives and true positives against the sum of all the classifications (true positive, true negative, false positives, false negatives).

3 Data Pre-processing

Before we go about building the predictive models, we must pre-process the data by performing the appropriate transformations. For example, among the most commonly used are checking if the predictors lie on the same scale, checking for correlations between predictors, and investigating if there are predictors with near zero variance. For each of these issues, if present, we will resolve by performing a transformation on the data where appropriate.

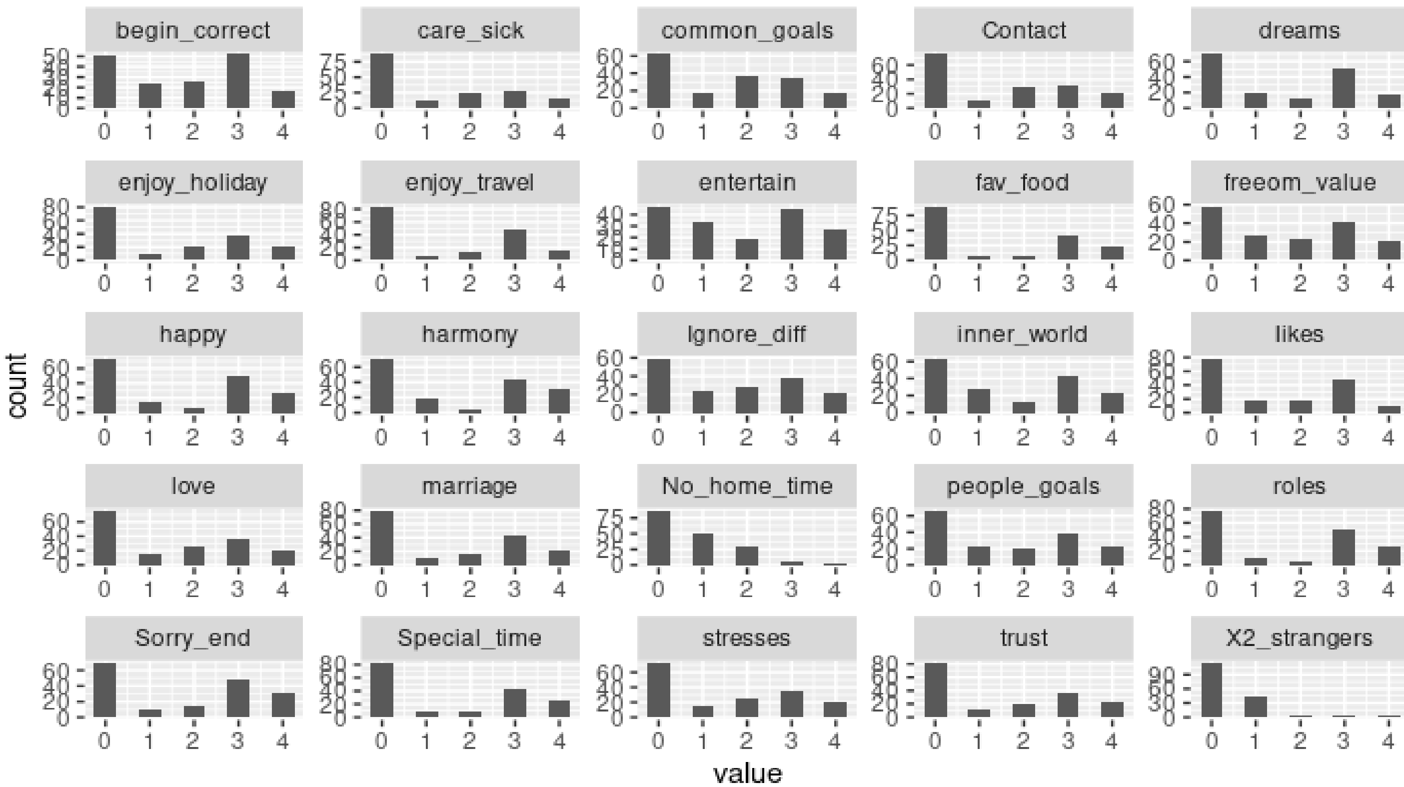
a. Correlation

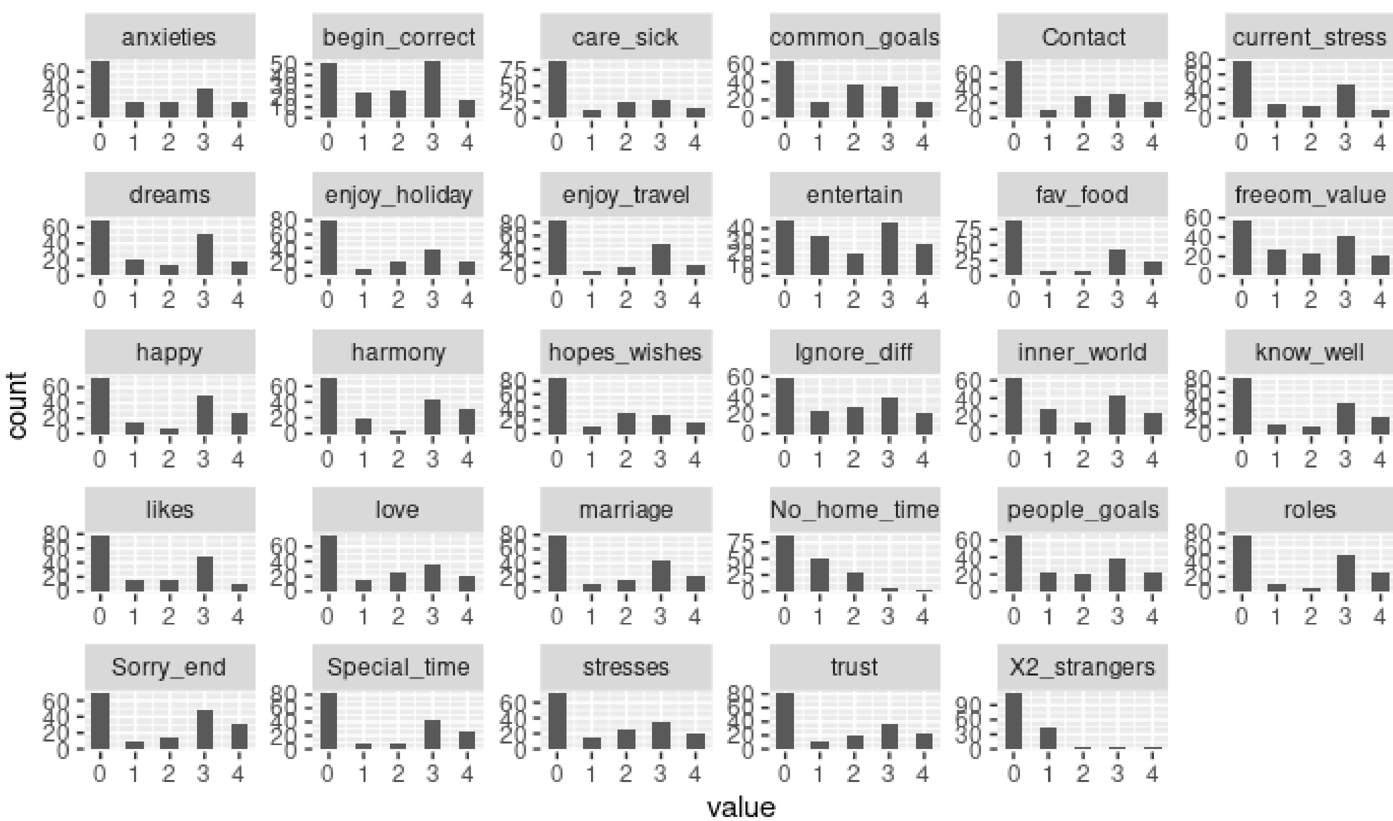
Prior to checking for the correlation structure of the data, we found that no predictors were classified as near zero variance, and there was no missing data for all predictors. Consequently, we plotted pairwise correlations for all the 54 predictors. As shown in the legend to the right, the positive correlation coefficient values are shown as blue and the negative correlation coefficients are shown as red. Given our relatively small sample size, we did not want to drop too many predictors, so we specified a cutoff value of 0.90, this resulted in predictors with greater than 0.90 pairwise correlations to be dropped. From the 54 predictors, there are 21 that fall within the 0.90 - 1.00 correlation coefficient range, resulting in 33 total predictors. The list of the dropped predictors can be found in the highcor variable in the R Code attached to this file.



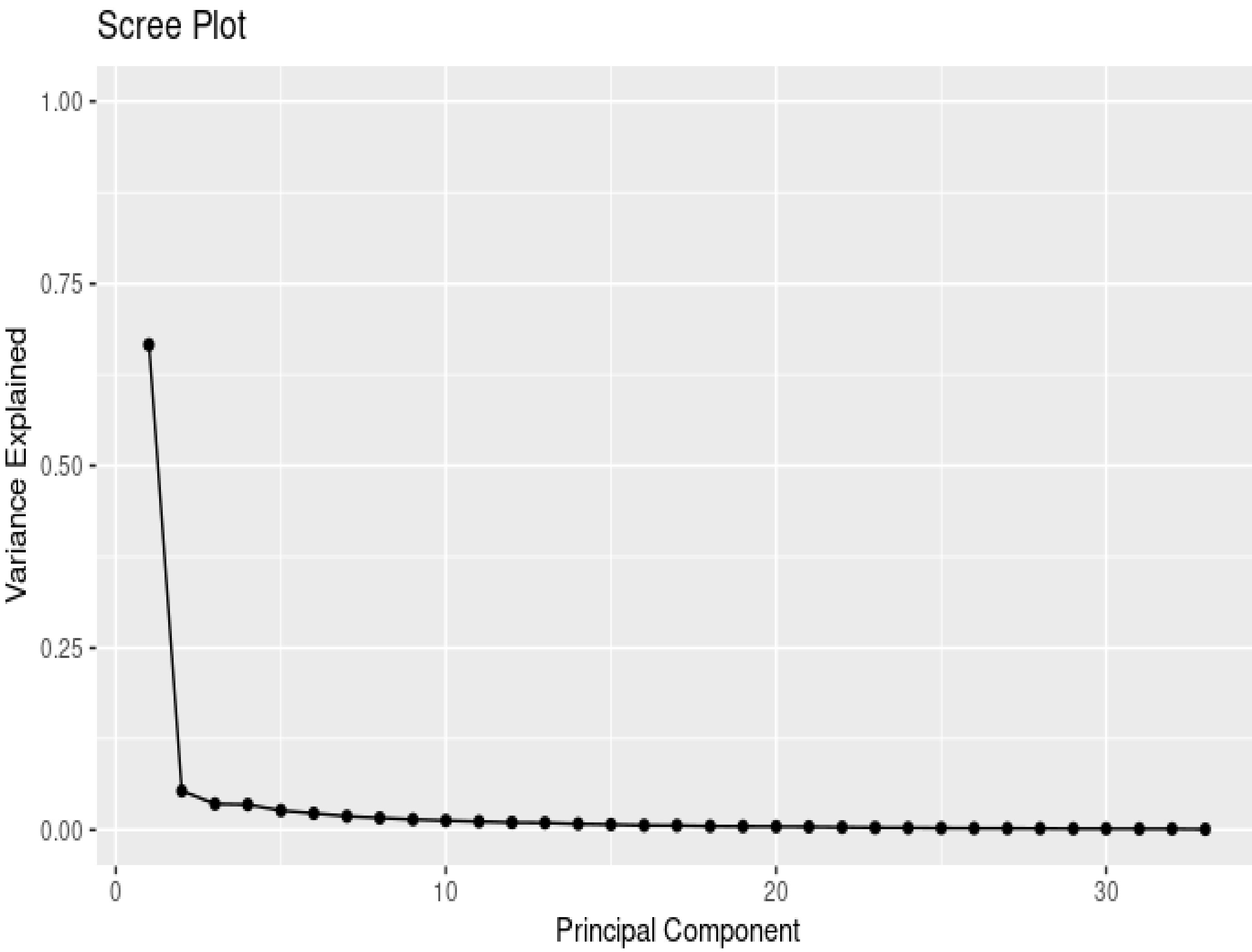
b. Transformation

Within the pre-processing phase, we generated histogram plots to get a rough idea of the distributional characteristics of the data. Below are the histograms for all predictors.



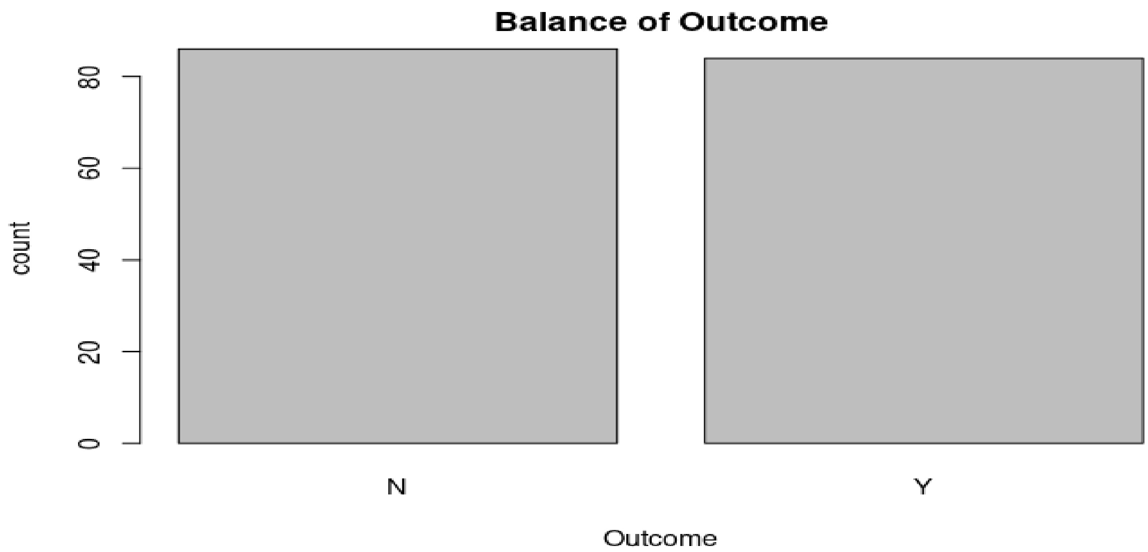


Moreover, given that our predictors are categorical in nature, and they all lie on the same scale, we do not require centering and scaling. It also didn’t make sense from a pre-processing perspective, to apply box cox given that the predictors are categorical. Furthermore, to reduce the dimensionality of the data even further, we performed Principal Component Analysis (PCA) and found the most optimal number of Principal Components (PC’s) to be 15, given that these first fifteen components explain 95% of the variance.



4 Splitting of the Data

It is after the data pre-processing stage where we specify the parameters for the data split, such as the percentage of data for training and the percentage of data for testing. Then, prior to performing the data split, it is useful to get an understanding on how the outcome variable is spread out among the samples (balanced or unbalanced). Below is a plot showing the outcome variable as roughly balanced (N=86, Y=84). Therefore, given the balanced design, we decided to use stratified random sampling (but simple random sampling would have worked fine as well) to split the data into two parts: 80% for training and 20% for testing.



5 Model Fitting

The models we selected can be split into two categories: Linear and Non-linear classification statistical models. The linear models are Logistic Regression, Linear Discriminant Analysis (LDA), Partial Least Squares Linear Discriminant Analysis (PLSDA), and Lasso and Elastic-Net Regularized Generalized Linear Models (GLMNET). The non-linear classification models are Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Naïve Bayes. We chose these models for our project for their robustness and wide applicability while also keeping in mind the nature and underlying data structure of our dataset. Below is a table summarizing the models. The models have been tuned using the training data and the accuracy and kappa values below are calculated using LGOCV cross validation in order to obtain the optimal results. The top two models were found to be SVM and KNN due to their high predictive accuracy which we can be defined as the sum of the true positives and true negatives divided by the sum of all classifications (TP+TN/(TP+TN+FP+FN).

Type	**Model**	**Best Tuning Parameters**	**Accuracy**	**Kappa**
Linear	Logistic Regression	- - - - - -	0.9381	0.9411
	Linear Discriminant Analysis	- - - - - -	0.9718	0.9529
	PLS Discriminant Analysis	Ncomp = 1	0.9788	0.9588
	GLMNET	$\alpha = 0.1$, $\lambda = 0.2$	0.9753	0.9437
Non-Linear	Support Vector Machine	$\sigma = 0.0049$, C = 2	0.9788	0.7617
	K-Nearest Neighbors	K = 3	0.9859	0.9315
	Naïve Bayes	Constant of fL = 2	0.9659	0.9201

The confusion matrices for each of the predictive models are summarized below. Given that our testing accuracy metric was very high, we can expect to see a large number of predictions lie in the true positive and true negative slots.

Logistic Regression		
Prediction	Reference	
	No	Yes
No	423	11
Yes	2	414

GLM NET		
Prediction	Reference	
	No	Yes
No	29750	1470
Yes	0	28280

Linear Discriminant Analysis		
Prediction	Reference	
	No	Yes
No	425	20
Yes	0	405

Support Vector Machine		
Prediction	Reference	
	No	Yes
No	4673	1112
Yes	2	3563

PLS Linear Discriminant Analysis		
Prediction	Reference	
	No	Yes
No	1700	60
Yes	0	1640

K-Nearest Neighbors		
Prediction	Reference	
	No	Yes
No	21247	1452
Yes	3	19798

Naïve Bayes		
Prediction	Reference	
	No	Yes
No	424	33
Yes	1	392

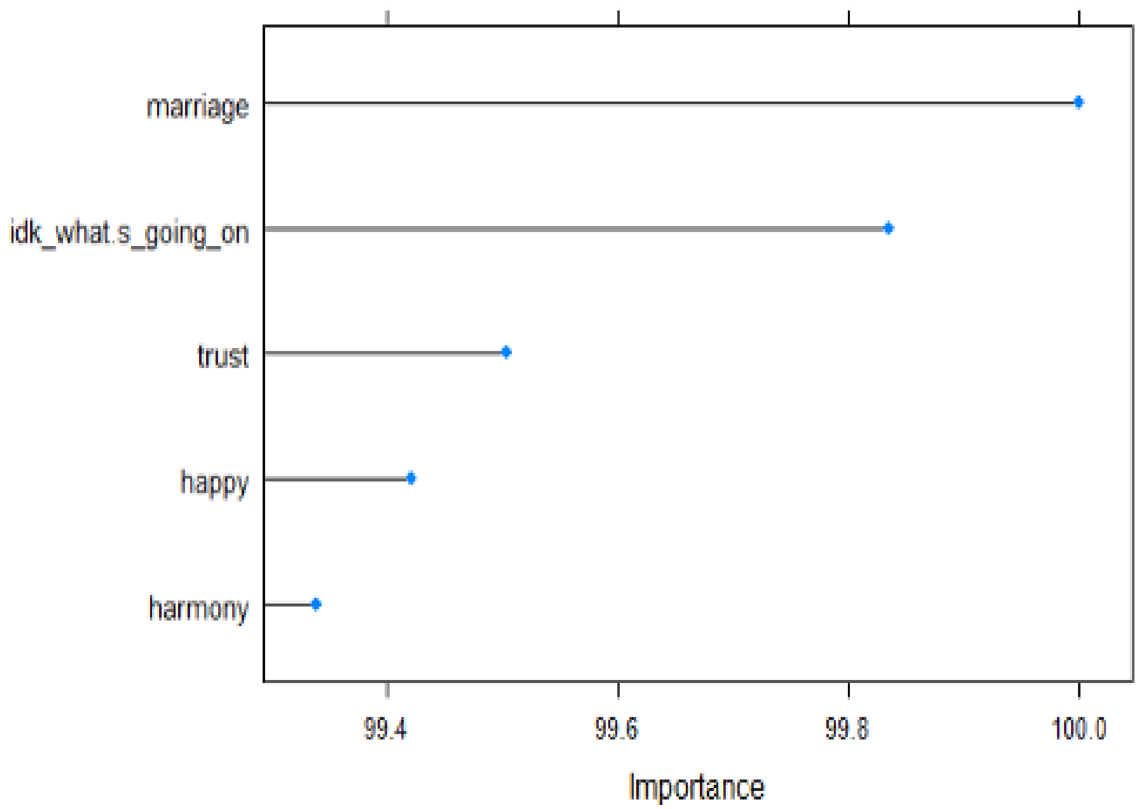
From the confusion matrices for each model, we can observe how the model performs at a more granular level. For example, the models seem to perform very well at avoiding false negatives, all while incurring in more false positive classifications. However, we denote that the true negatives and true positives significantly outnumber the false negatives/positives, which explains our high model accuracies.

Nonetheless, the more exciting part lies on how our models performed on the testing data, what the more granular classification metrics are, as well as the best tuning parameters. Such information is condensed in the table below for a clear and succinct view of our top two models' performance on the testing and training data. The best performing model is KNN, followed by SVM. Both of these models are non-linear, however, it is worth noting that the linear models performed quite well.

Furthermore, we can conclude that the most optimal models for the testing data are the same as those for the training data.

Type	**Top Two Models**	**Best Tuning Parameters**	**Train Accuracy**	**Test Accuracy**
Non-Linear	Support Vector Machine	σ = 0.0049, C = 2	0.9788	0.9697
	K-Nearest Neighbors	K = 3	0.9859	0.9697

For the K-Nearest Neighbors model, we further analyzed the data to find the most important predictors for predicting the response variable. The plot below summarizes the most important features for the logistic regression model, which was our final candidate model. As per the meaning of these variables, they are defined in the variable descriptions section above.



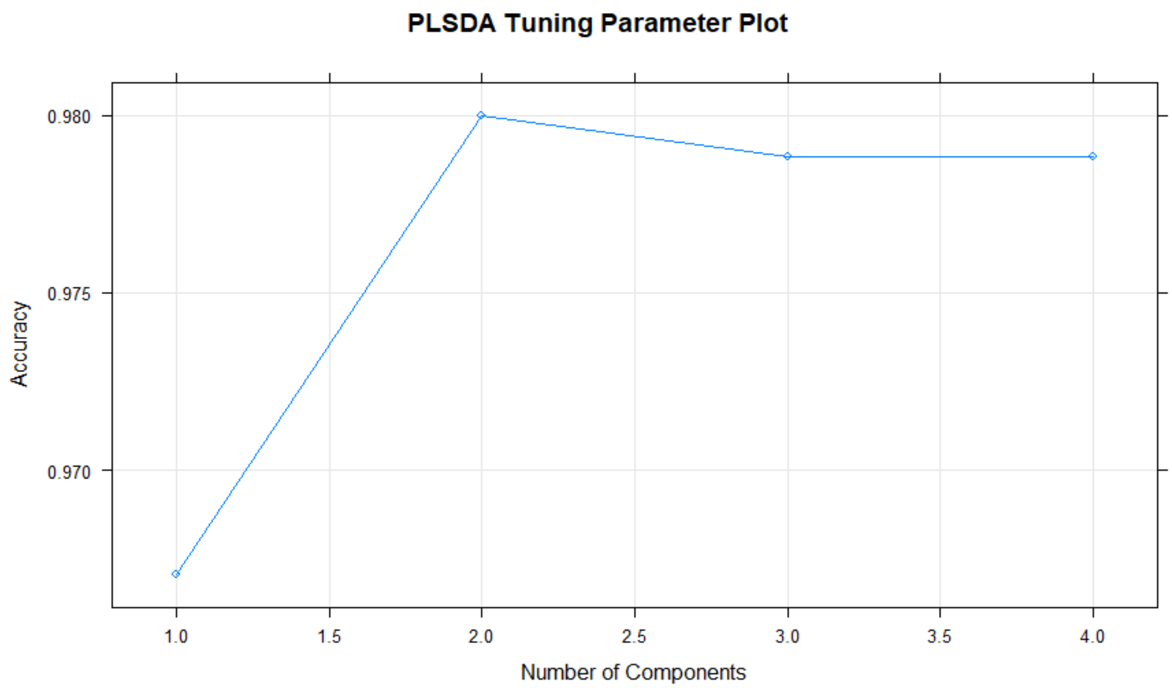
6 Summary

From our predictive model evaluations, we determined that the best model found during this analysis was the K-Nearest Neighbors model due to its high performance as well as its suitability for our data given the relatively small sample size. The resulting area under the curve was 0.9976 and the accuracy rate was 0.9697. These values are quite pleasant to report as the model is extremely suitable for our problem at hand, which was to predict whether a marriage will result in divorce based on the given set of predictors. Some final thoughts on this project would be that a larger sampling size would have been better, but overall, this was a very interesting problem from which we gained a lot of insight and value exploring.

Appendix 1: Tuning Parameters

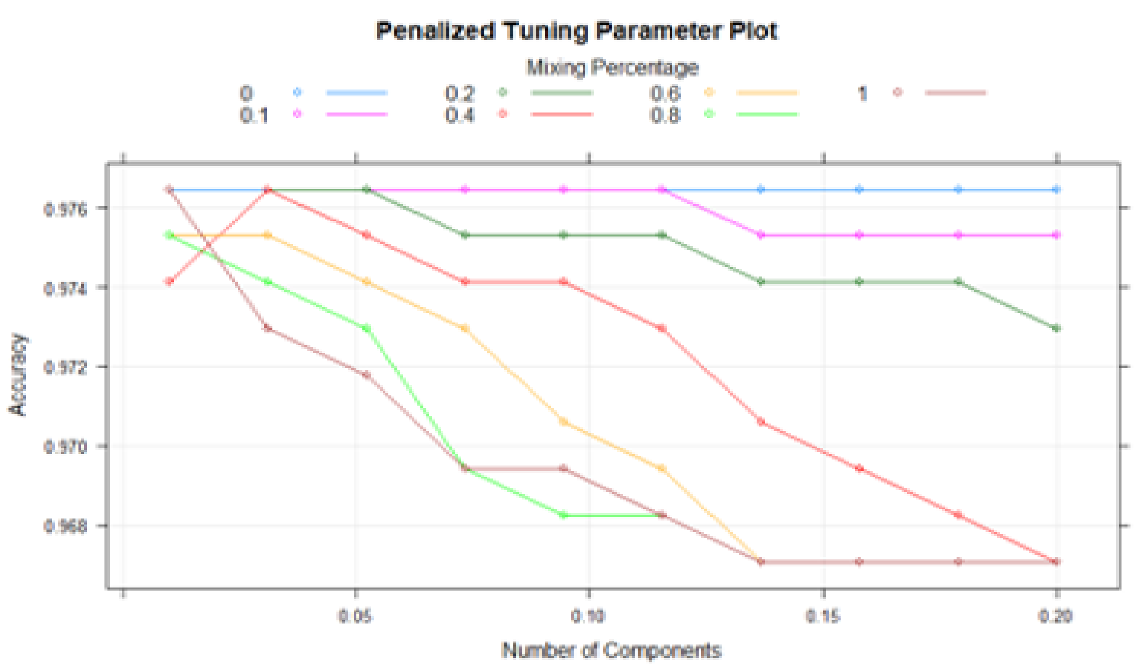
1) PLS Linear Discriminant Analysis

From the plot shown below we find that the number of components and their relationship to the ROC remain constant between 1.0 and 4.0, therefore per the trained model we set the number of components equal to 1.



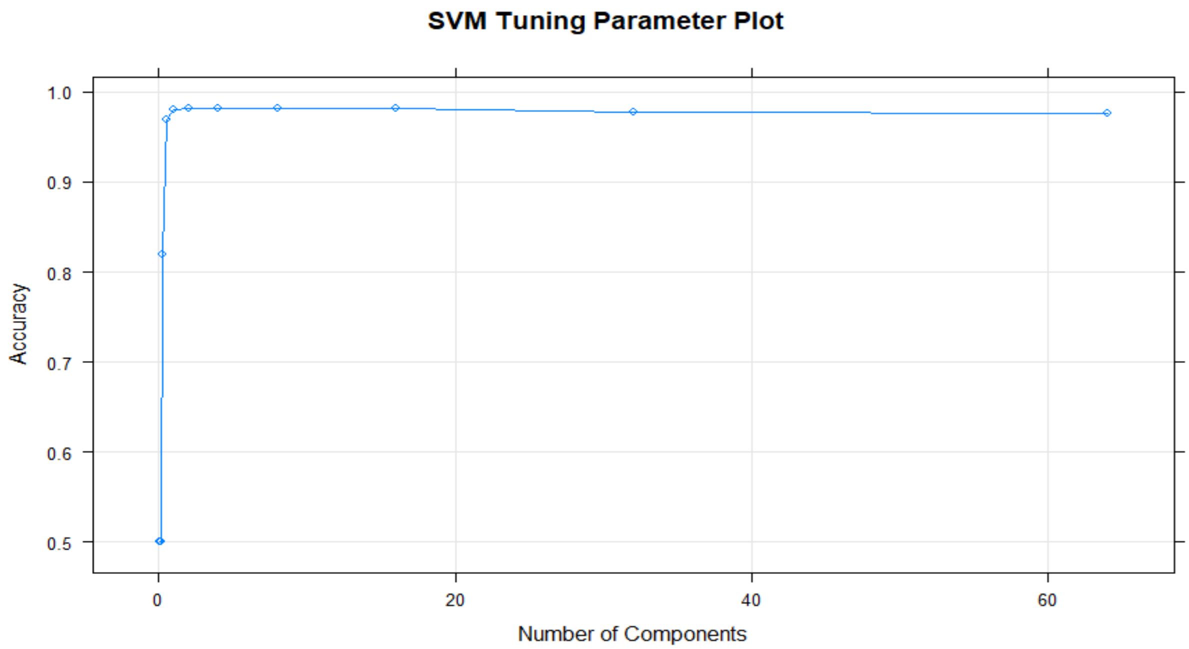
2) GLMNET

From the plot shown below, as well as the model summary output for the GLMNET model, we find that the best tuning parameters are $\alpha = 0.1$, $\lambda = 0.2$. Consequently, these parameters minimize the misclassification error rate which improves the overall predictive ability of our models.



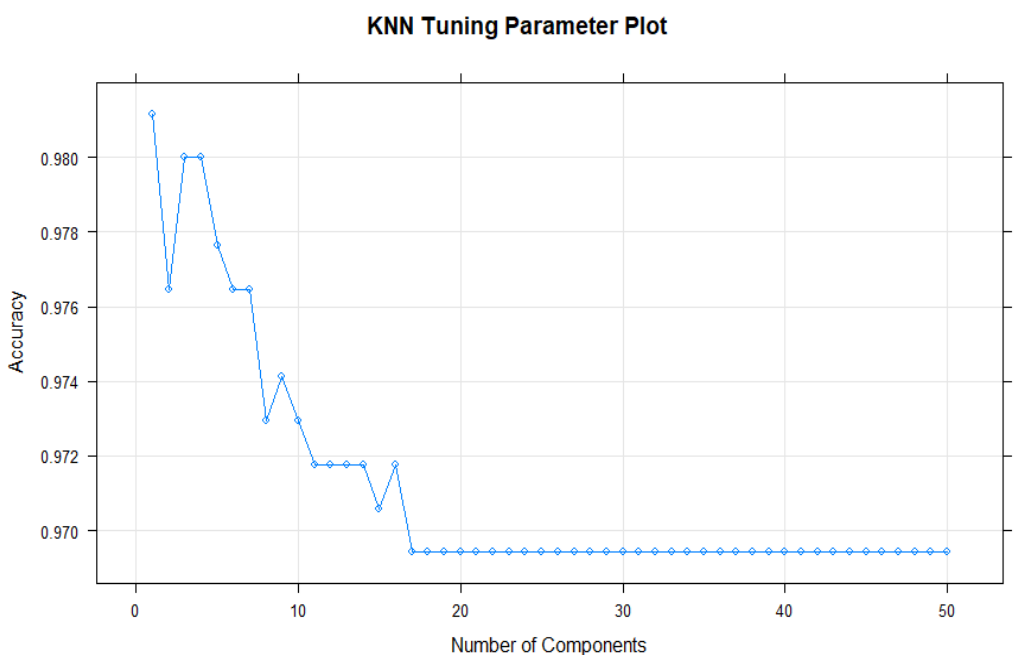
3) SVM

From the plot shown below, as well as the model summary output for the SVM model, we find that the best tuning parameters are $\sigma = 0.0049$, C = 2.



4) KNN

From the plot above, as well as the model summary output for the KNN model, we find that the best tuning parameters are K = 3.



R Code:

```
---
title: "MA 5790 Project"
author: "Ezequiel Carrillo and Samantha Hair"
date: "1/22/2022"
output: html_document
---

```{r setup}
library(purrr)
library(tidyselect)
library(dplyr)
library(tidyr)
library(ggplot2)
library(dplyr)
library(Hmisc)
library(e1071)
library(caret)
library(corrplot)
library(moments)
library(kernlab)
```
```

Data Description
For this project, we are using the dataset 'Predicting Divorce' (https://www.kaggle.com/csafrit2/predicting-divorce). The goal of using this dataset, is to predict if a couple is likely to get divorced in the future. All 54 of the predictor variables are categorical and are ranked on a scale from 0-4, with 0 being the lowest and 4 being the highest. The variable that is the outcome, is Divorce (Y/N), coded in binary. The predictor variables as described on Kaggle are as follows:

1. If one of us apologizes when our discussion deteriorates, the discussion ends.
2. I know we can ignore our differences, even if things get hard sometimes.
3. When we need it, we can take our discussions with my spouse from the beginning and correct it.
4. When I discuss with my spouse, to contact him will eventually work.
5. The time I spent with my wife is special for us.
6. We don't have time at home as partners.
7. We are like two strangers who share the same environment at home rather than family.
8. I enjoy our holidays with my wife.
9. I enjoy traveling with my wife.
10. Most of our goals are common to my spouse.
11. I think that one day in the future, when I look back, I see that my spouse and I have been in harmony with each other.
12. My spouse and I have similar values in terms of personal freedom.
13. My spouse and I have similar sense of entertainment.
14. Most of our goals for people (children, friends, etc.) are the same.
15. Our dreams with my spouse are similar and harmonious.
16. We're compatible with my spouse about what love should be.
17. We share the same views about being happy in our life with my spouse
18. My spouse and I have similar ideas about how marriage should be
19. My spouse and I have similar ideas about how roles should be in marriage
20. My spouse and I have similar values in trust.
21. I know exactly what my wife likes.
22. I know how my spouse wants to be taken care of when she/he sick.
23. I know my spouse's favorite food.
24. I can tell you what kind of stress my spouse is facing in her/his life.
25. I have knowledge of my spouse's inner world.
26. I know my spouse's basic anxieties.
27. I know what my spouse's current sources of stress are.

28. I know my spouse's hopes and wishes.
29. I know my spouse very well.
30. I know my spouse's friends and their social relationships.
31. I feel aggressive when I argue with my spouse.
32. When discussing with my spouse, I usually use expressions such as ‘you always’ or ‘you never’ .
33. I can use negative statements about my spouse's personality during our discussions.
34. I can use offensive expressions during our discussions.
35. I can insult my spouse during our discussions.
36. I can be humiliating when we discussions.
37. My discussion with my spouse is not calm.
38. I hate my spouse's way of open a subject.
39. Our discussions often occur suddenly.
40. We're just starting a discussion before I know what's going on.
41. When I talk to my spouse about something, my calm suddenly breaks.
42. When I argue with my spouse, ı only go out and I don't say a word.
43. I mostly stay silent to calm the environment a little bit.
44. Sometimes I think it's good for me to leave home for a while.
45. I'd rather stay silent than discuss with my spouse.
46. Even if I'm right in the discussion, I stay silent to hurt my spouse.
47. When I discuss with my spouse, I stay silent because I am afraid of not being able to control my anger.
48. I feel right in our discussions.
49. I have nothing to do with what I've been accused of.
50. I'm not actually the one who's guilty about what I'm accused of.
51. I'm not the one who's wrong about problems at home.
52. I wouldn't hesitate to tell my spouse about her/his inadequacy.
53. When I discuss, I remind my spouse of her/his inadequacy.
54. I'm not afraid to tell my spouse about her/his incompetence.

The variable that is the outcome, is Divorce (Y/N), coded in binary.

````
```{r dataset}
data_raw <- read.csv("divorce.csv")
data_raw <- data_raw %>%
  mutate(Divorce_Y_N = as.factor(if_else(Divorce_Y_N == 1, "Y", "N")))
data_predict <- data_raw[, -55]
data_out <- data_raw[, 55]
dim(data_raw)
```

```{r explore the data}
#histograms for first 20 predictors
data_predict[,seq(1:20)] %>%
  keep(is.numeric) %>%
  gather() %>%
  ggplot(aes(value)) +
  facet_wrap(~ key, scales = "free") +
  geom_histogram(bins = 9)
#histograms for predictors 21-40
data_predict[,seq(21:40)] %>%
  keep(is.numeric) %>%
  gather() %>%
  ggplot(aes(value)) +
  facet_wrap(~ key, scales = "free") +
  geom_histogram(bins = 9)
#histograms for predictors 41-55
data_predict[,seq(41:55)] %>%
  keep(is.numeric) %>%
  gather() %>%
  ggplot(aes(value)) +
  facet_wrap(~ key, scales = "free") +
  geom_histogram(bins = 9)
````
````

````
#between predictor correlations
corr <- cor(data_predict[, -55])
corrplot(corr, order = "hclust")
#frequency table
freq_tab <- lapply(data_predict, table)
#missing data
sum(!complete.cases(freq_tab))
#calculate skewness for all predictors
predictorSkew=round(skewness(data_predict),3);
#plot skewness
boxplot(predictorSkew, main='Spread of Skewness for all predictors', xlab='predictors (54)', ylab='skewness coefficient');
plot(predictorSkew, main='scatter plot of predictor and its skew',xlab = 'predictor index',ylab = 'skewness coefficient')
hist(predictorSkew)
```

#Spending the data
```{r spending the data}
set.seed(1)
train_row <- createDataPartition(data_out, p = .8, list = FALSE)
data_out_frame <- as.data.frame(data_out)
#train data
train_predict <- data_predict[train_row, ]
train_out <- data_out_frame[train_row, ]
#test data
test_predict <- data_predict[-train_row, ]
test_out <- data_out_frame[-train_row, ]
```

#Adding Predictors
Since all of our predictors are categorical, we do not need to add any predictors.

#Deleting Predictors
```{r deleting predictors}
novar <- nearZeroVar(train_predict)
corr <- cor(train_predict)
highcor <- findCorrelation(corr, cutoff = .90)
#remove predictors that are highly correlated
data_trans1 <- train_predict[,-highcor]
dim(data_trans1)
```


#Imputation
Since we do not have any missing values, we have no need to imputate any variables.

#Center and Scale
No need to center and scale since all variables are on the same categorical scale of 0-4.

#Spatial Sign
No need for spatial sign

#Box-Cox transformation
#PC
```{r transforming the data}
trans <- preProcess(data_trans1, method = c("pca"))
#scree plot
results <- prcomp(data_trans1, center = TRUE, scale = TRUE)
var_explained <- results$sdev^2 / sum(results$sdev^2)
qplot(c(1:33), var_explained) +
  geom_line() +
````

````
  xlab("Principal Component") +
  ylab("Variance Explained") +
  ggtitle("Scree Plot") +
  ylim(0, 1)
data_new <- predict(trans, data_trans1)
dim(data_new)
```


#Resampling the data w/ multiple models
```{r resampling}
ctrl <- trainControl(method = "LGOCV", classProbs = TRUE, savePredictions = TRUE)
proc <- c("center", "scale", "pca", "corr")
#Linear
#Logistic
lgfit <- train(x = train_predict, y = train_out, method = "glm", metric = "Accuracy", trControl = ctrl, preProcess = proc)
#LDA
ldafit <- train(x = train_predict, y = train_out, method = "lda", metric = "Accuracy", trControl = ctrl, preProcess = proc)
#PLSDA
plsfit <- train(x = train_predict, y = train_out, method = "pls", tuneGrid = expand.grid(.ncomp = 1:4), preProcess =
proc, metric = "Accuracy", trControl = ctrl)
#Penalized
glmgrid <- expand.grid(.alpha = c(0,.1,.2,.4,.6,.8,1), .lambda = seq(.01,.2,length = 10))
penfit <- train(x = train_predict, y = train_out, method = "glmnet", metric = "Accuracy", tuneGrid = glmgrid,
preProcess = proc, trControl = ctrl)
#Non-Linear
#SVM
sigmred <- sigest(as.matrix(train_predict))
svmgrid <- expand.grid(.sigma = sigmred[1], .C = 2^(seq(-4, 6)))
svmfit <- train(x = train_predict, y = train_out, method = "svmRadial", metric = "Accuracy", tuneGrid = svmgrid,
preProcess = proc, trControl = ctrl, fit = FALSE)
#KNN
knnfit <- train(x = train_predict, y = train_out, method = "knn", metric = "Accuracy", tuneGrid = data.frame(.k = 1:50),
preProcess = proc, trControl = ctrl)
#Naive
nbfit <- train(x = train_predict, y = train_out, method = "nb", metric = "Accuracy", tuneGrid = data.frame(.fL = 2,
.usekernel = TRUE, .adjust = TRUE), preProcess = proc, trControl = ctrl)
```

#Tuning Parameter Plot
```{r}
plot(plsfit, main = "PLSDA Tuning Parameter Plot", xlab = "Number of Components", ylab = "Accuracy")
plot(penfit, main = "Penalized Tuning Parameter Plot", xlab = "Number of Components", ylab = "Accuracy")
plot(svmfit, main = "SVM Tuning Parameter Plot", xlab = "Number of Components", ylab = "Accuracy")
plot(knnfit, main = "KNN Tuning Parameter Plot", xlab = "Number of Components", ylab = "Accuracy")
```


#Finding the best model
```{r}
#Linear
lgfit
confusionMatrix(data = lgfit$pred$pred, reference = lgfit$pred$obs)
ldafit
confusionMatrix(data = ldafit$pred$pred, reference = ldafit$pred$obs)
plsfit
confusionMatrix(data = plsfit$pred$pred, reference = plsfit$pred$obs)
penfit
confusionMatrix(data = penfit$pred$pred, reference = penfit$pred$obs)
#Non-Linear
svmfit
````

~~~
confusionMatrix(data = svmfit$pred$pred, reference = svmfit$pred$obs)
knnfit
confusionMatrix(data = knnfit$pred$pred, reference = knnfit$pred$obs)
nbfit
confusionMatrix(data = nbfit$pred$pred, reference = nbfit$pred$obs)
```

#Calculate predictions
```{r performance}
#Linear
lgpred <- predict(lgfit, newdata = test_predict)
confusionMatrix(data = lgpred, test_out)
ldapred <- predict(ldafit, newdata = test_predict)
confusionMatrix(data = ldapred, test_out)
plspred <- predict(plsfit, newdata = test_predict)
confusionMatrix(data = plspred, test_out)
penpred <- predict(penfit, newdata = test_predict)
confusionMatrix(data = penpred, test_out)
#Non-Linear
svmpred <- predict(svmfit, newdata = test_predict)
confusionMatrix(data = svmpred, test_out)
knnpred <- predict(knnfit, newdata = test_predict)
confusionMatrix(data = knnpred, test_out)
nbpred <- predict(nbfit, newdata = test_predict)
confusionMatrix(data = nbpred, test_out)
```

#Best Predictors
```{r best predictors}
aa <- varImp(knnfit)
plot(aa, top = 5, scales = list(y = list(cex = .95)))
bb <- varImp(plsfit)
plot(bb, top = 5, scales = list(y = list(cex = .95)))
``
~~~