Final Project Report

Team 6: Zinan Chen, Qiuhao Chengyong, Qiaoling Huang , Melissa Putur, Weifu Shi.

Project Goals:

- Identify unique customer segments through cluster analysis.
- Predict whether or not a customer will be able to pay off their loan through dimensionality reduction and supervised machine learning.

Dataset:

The dataset we analyzed contained over 800,000 rows and 30 columns. Each row in the table represented an individual loan and the columns described information both about the loan and the customer that took out the loan including; current status of the loan (Good vs. Bad), reason for loan, salary, housing status (rent, own), and employment length at current job.

The dataset can be found here: https://www.kaggle.com/mrferozi/loan-data-for-dummy-bank

Initial Data Cleaning:

As part of our data cleaning process we converted all categorical data columns to dummy variables. We converted the following columns:

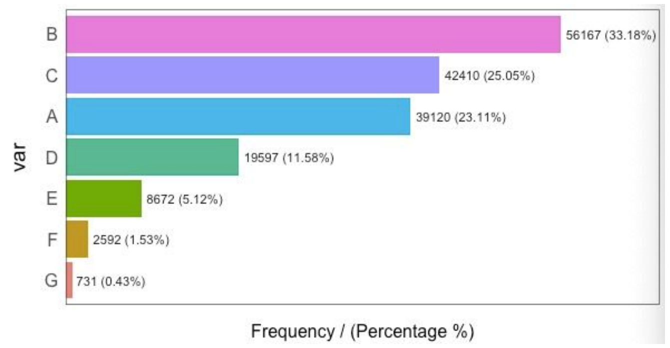
- Region: Converted to 5 dummy region columns; Munster, Leinster, Cannught, Ulster and Northern Ireland
- Home Ownership: Converted to 3 dummy columns; Rent, Own, and Mortgage
- Purpose: Converted to 5 dummy columns; Credit Card, Medical, House, Small Business and Vacation
    - Note: The original dataset contained 14 purpose categories, we knew we did not want to keep every category because the original dataset was so large we were not able to successfully upload it to github. When choosing which purposes to keep, we chose the categories that were both interesting to us and were included in at least a couple thousand rows. We removed categories like education, renewable energy and wedding that did not make up a large portion of the total loan reasons.
- Loan Grade: Converted values “A”,”B”,”C” etc. to 7,6,5 etc.

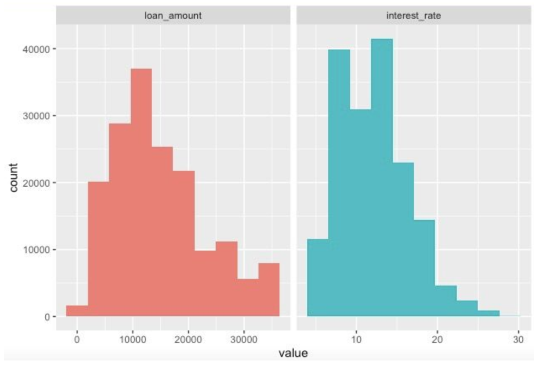
Our data did not have any missing values or columns we did not understand so we did not need to complete any further cleaning. Our clean dataset contains 8 numerical columns, 20 binary columns, and over 242,000 rows.

High-Level Exploratory Data Analysis

- The data contains ~11,000 (6%) loans that are in a “good” condition while the rest are in a “bad” condition.
- The majority of the loans are grade A, B or C.

- The distribution of the loan amounts and interest rates are approximately the same, both distributions are fairly normal with a slight skew to the right.



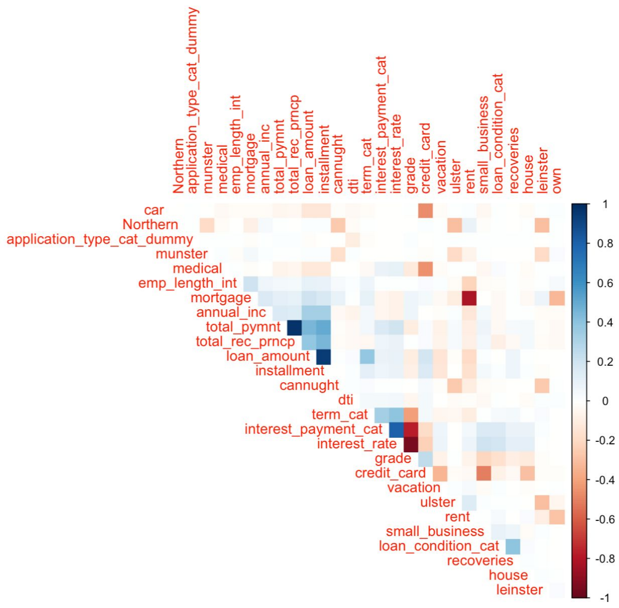


In examining the distributions of other variables we noticed that a couple numeric variables have outliers, like debt to income. We have not decided how to manage our outliers yet, we may end up deleting outliers over a specific standard deviation threshold if we feel like they are impacting or skewing our analysis as we get further along.

<u>Principal Component Analysis</u>

Our goal when completing principal component analysis was to use the principal components as additional variables for both our cluster analysis and machine learning algorithm to predict loan status.

Before beginning the principal component analysis we created a correlation matrix of all of the variables. To the right you can see that generally the correlation matrix was very light in color which told us that the majority of variables were uncorrelated with each other. This suggested that PCA may not be the best method to use for our dataset, but we still decided to move forward to see if we could get some useful variables.



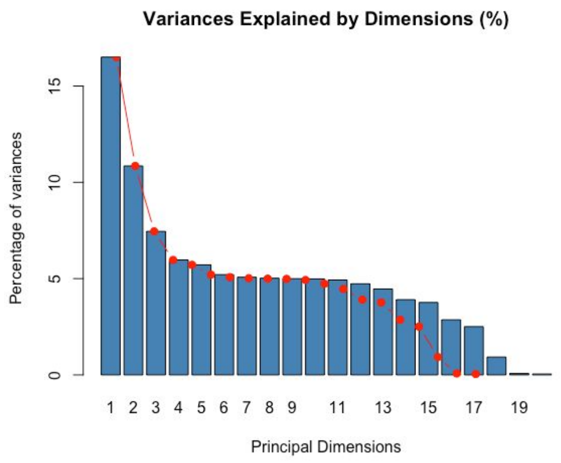
Our next challenge with PCA was figuring out how to scale the data, if we needed to scale the data, and how to manage the binary/dummy variables we had created during our cleaning phase. We proceeded by creating principal components using several different scaling methods:

- **Method 1 -** Create principal components, only using the numerical data in the dataset, and then combine the principal components with the untouched binary data for clustering. For this method, 4 principal components was optimal based on the eigan-value methodology, these principal components explain 72% of the variance in the numerical data.

- **Method 2 -** Create principal components using all of the data, with all of the data scaled. For this method, variance was very slow to be reduced and 16 principal components contain an eigan-value greater than 1. With the 16 principal components, 86% of the variance in the data was explained.
- **Method 3 -** Create principal components using all of the data, scaled with a max/min methodology. For this method, none of the principal components contained an eigan-value greater than 1, but looking at the screeplot, the elbow method suggests that maybe 7 principal components will be optimal. Using 7 principal components explains approximately 70% of the variance in the data.
- **Method 4 -** Create principle components using all of the columns, but with just the numerical variables scaled. For this method the first four principal components have an eigenvalue greater than 1, and can explain 64.37% of variance in the data.

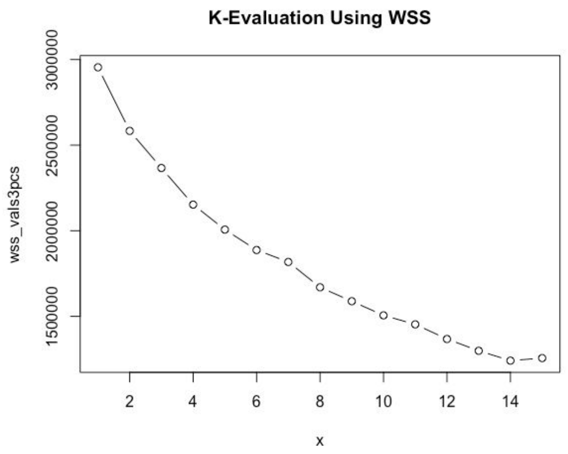
Ultimately we were not very pleased with these results because in almost every case we had to use several principal components just to explain 70% of the variability. Because of this we sought out other methods for dimensionality reduction.

We found an alternate method for dimensionality reduction called “factor analysis of mixed data” which is used when a dataset contains both numerical and categorical data. FAMD analysis uses methods from both principal component analysis and multiple correspondence analysis The article we learned of this method from can be found here: http://www.sthda.com/english/articles/22-principal-component-methods-videos/72-famd-in-r-using-factominer-quick-scripts-and-videos/



Using the FAMD method, the screeplot recommended using 4 principle components, however only choosing 4 did not explain much variance in the data. Again we evaluated the eigen-values which recommended 12 components and explained 80% of the variance in the data. We then leveraged these components for our cluster analysis.

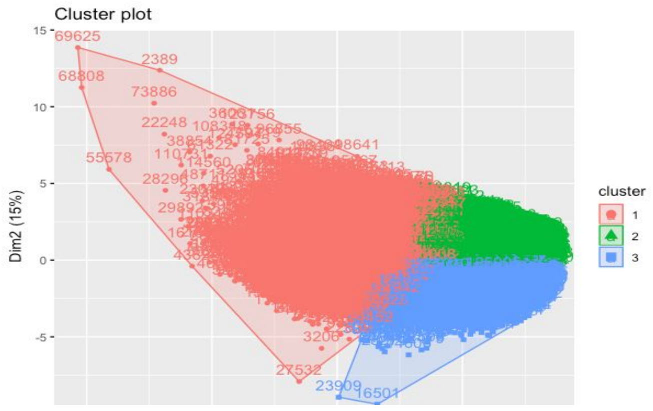
Cluster Analysis

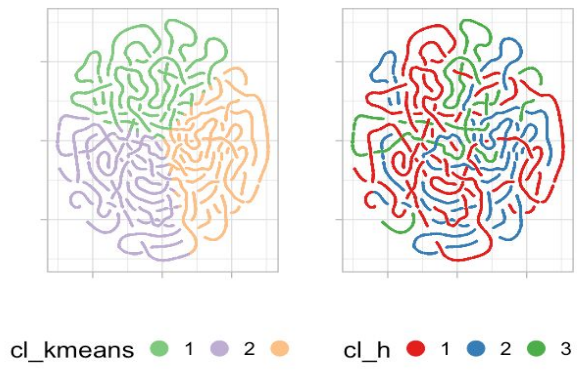


We tried three methods for cluster analysis.

- **Method 1:** First, we used the original dataset, without any principal components and applied k-means clustering. To evaluate the optimal number of clusters, we created a WSS elbow plot however, the plot was fairly smooth and there was no clear elbow. Because we are trying to segment customer data, we decided somewhere between 2 and 5 segments would be a manageable amount for the bank to work with. We tried to use K from 2 to 5 to see which K makes more sense to us.We also found there are some outliers from our plot, we went back to clean these outliers from our data

- **Method 2:** Second, we used the first 12 components from our FAMD analysis to again attempt k-means clustering. The cluster plots for method 1 and method 2 looked very similar, but with a slight improvement for the second method.
- **Method 3 and 4:** Finally, we attempted to run hierarchical clustering, but had trouble getting it to complete with our original dataset. This was not surprising to us with the size of our data and we did initially think the hierarchical clustering would not be the best fit, but we decided to sample 20,000 rows and apply T-sne dimension reduction. We used the two dimensions from T-sne to run hierarchical clustering and k-means for comparison. We can see the distribution of clusters in K-means are more clear and organized than hierarchical.





Ultimately we decided that method 2 created the tightest clusters. The clusters had the following high-level characteristics:

- **Cluster 1:** Highest earners, largest loan amounts (more than double average of other two groups), typically own a house
- **Cluster 2:** Riskiest customers, lowest earners, larger percent of renters, relatively even split between short and long term loans
- **Cluster 3:** Middle of the road customers, Middle earners, larger percent of renters, mostly 36 month loans

<u>Loan Status Prediction</u>

As stated earlier, the second goal of our project was to predict whether the status of a loan ended in “good” or “bad”.

As stated earlier, we would like to help the bank predict whether or not a customer will ultimately be able to pay off their loan. First, before fitting any models, we had to split the data into a train and test dataset. The train data was used to fit the model while the test data was used to measure the accuracy of the mode. We split the data randomly using the following function: cleandata$train <- sample(c(0, 1), nrow(cleandata), replace = TRUE, prob = c(.3, .7)).

Model 1 - Boosting with Original Data: We selected gradient boosting for our first supervised machine learning prediction method. We though boosting would be a good fit for our prediction problem based on the higher accuracy rates we have seen in previous classes. We also decided to use our original dataset, without any modifications or principal components added in. After adjusting the num_class parameter a couple of times, we were able to predict with 96% accuracy whether a loan was good or bad.

Model 2 - Boosting with Principal Components: We also tried gradient boosting but with the data updated to include principal components. The accuracy of the model was a little lower, and we were only able to predict with 94.1% accuracy whether the loan was good or bad.

Summary and Conclusions

Throughout this project we spent a lot of time exploring the best way to handle our dummy variables so that we could apply dimension reduction to our data. Despite our best efforts, we found that most methods for creating principal components were slow to explain variance in the data and did not provide much additional value over our original dataset. This was reiterated when we tried two different boosting models to predict whether or not a loan would default. We were able to get higher prediction accuracy when we trained the model with our untouched data. Despite our initial challenges, we sought out additional options for reduction techniques that dealt well with datasets with mixed categorical and numerical variables. FAMD analysis worked well with k-means clustering and we were able to uncover three unique customer segments within our data.

GitHub Repository: https://github.com/qchengyo/Unsupervised-Machine-Learning