Loan default prediction

Dakshin Kannaan, Siddharth Iyer, Santino Luppino, Zheng Li, Dimitri Niles
I pledge my honor that I have abided by the Stevens Honor System.

1. Overview

Loan default is a critical problem to banks and other financial institutions since it has a huge effect on profit. In 2018, the US Banks report the share of non-performing loans to be 1.02%. This is still a really high number considering the numerous number of loans issued every year. Hence, loan default prediction is necessary. It can minimize risk by being able to predict which borrower can pay back the loans, and it is particularly useful for people without a credit history.

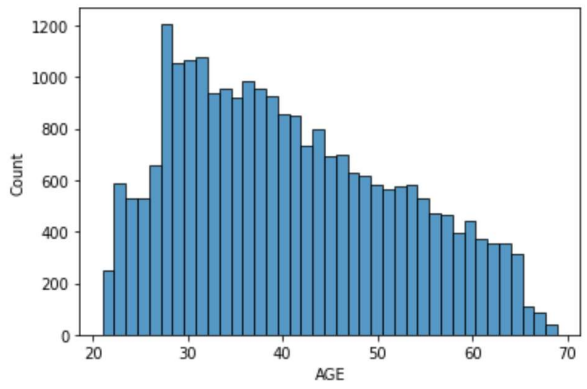
2. Description of data, data cleaning

Our data is a subset of competition data provided by Home Credit International. It had data for 307,512 loans and 122 columns of factors for each loan. Due to the size of the data set, we sampled 10,000 of the loans to use in the analysis. Many columns had data irrelevant to our analysis (for example SK_ID_CURR just held the IDs of the loans in the sample, which says nothing about the chances of the loan defaulting). In the end, we had to cut down about 100 columns because some factors are based on the evaluation that is not open to us and also any number of factors over 20 will just take too long to run for each model. For the models, we used the most theoretically important 15 factors as inputs. They were target (whether or not the loan defaulted), contract type, gender, whether they owned a car, how many children they have, income, credit, annuity, who the client was accompanied by when applying for the loan, income type, highest education, family status, housing type, relative region and age (at the time they applied for the loan. We converted the original data reported in days to years). A few factors used string to describe their status so we converted those to numbers (for example, gender used M and F so we used 0's and 1's to represent them).

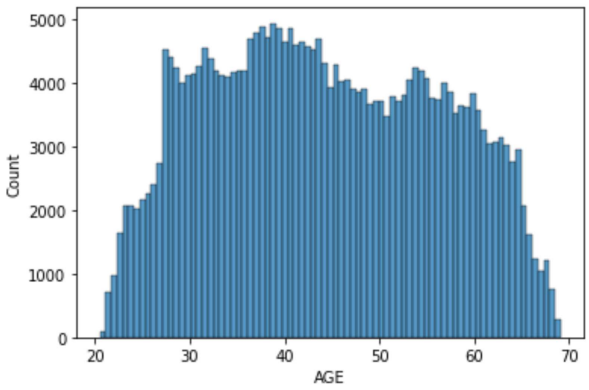
Exploratory Data Analysis:

Looking for trends in the data and possible explanations for defaults before applying the machine learning methods.

Age Distribution of Defaulted Loans

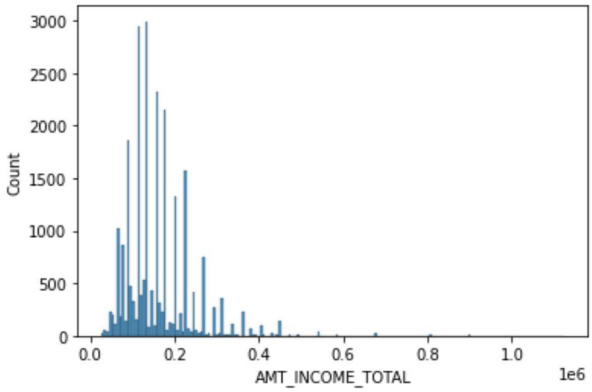


Age Distribution of Non-defaulted loans



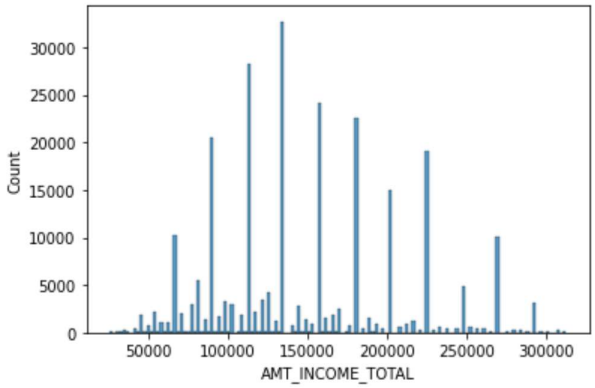
Here we can see that the ages for defaulted loans skews younger than non-defaulted loans. This shows that age is likely to be an important predictor in whether a loan defaults. It is also worth noting that mid-20's and below and mid-60's and above are much less likely to have loans

Income Distribution of Defaulted Loans



The spikes are from people rounding up to the nearest $10k when collecting data.

Income Distribution of Non-defaulted loans

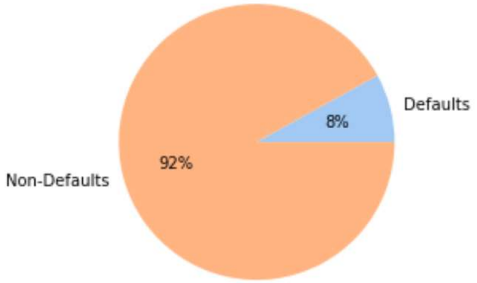


The income for the defaulted loans is heavily skewed towards the right while the non-defaults are more evenly spread. Unsurprisingly, this shows that those with less income may have a harder time paying back a loan.

Men are more likely to default

```
Male Default Rate:
0.10166590161426058
Female Default Rate:
0.07004736280903613
```

Important to note that there were more females than males in the data set (about a 2:1 ratio) so gender may be an important predictor as well.



Far more non-default loans than defaults in this data set. Unbalanced datasets are bad for ML classification.

3. Machine learning methods

We used R and Python and their respective machine learning libraries such as Keras. Methods used were random forest, logistic regression, support vector machine, neural network.

Clustering (K-means clustering and Hierarchical clustering): The data will be clustered based on their reported values by using the kmeans() function. The parameters for this function will be all of the independent variables in our study, which excludes the target variable, 6 clusters, and 20 attempts. The best cluster will then be displayed. We choose 6 clusters as the appropriate amount of clusters because we do not want to make an extraneous amount of clusters. Our data that we have is not too diverse considering we mostly have binary data with the exception of a few data points. Moreover, restricting us to smaller amounts of clusters will group the variables in clusters together that do not necessarily belong to each other. Therefore, 6 clusters should be appropriate so we do not underfit and overfit our data. Moreover, the same data will now be clustered by using hierarchical clustering with complete distance metric, single

distance metric, and average distance metric. The dendrogram will be computed and displayed for each of the clusters. We will then cut 6 clusters using the cutree() function and display the results, so it is easier to compare the performance to that of the K-means clustering.

Models: Logistic, SVM, Random Forest, Neural Network

These four models were chosen to find the model with the best performance and the lowest complexity. To test the performance of these models in different scenarios, we trained and tested the models in three different ways: unbalanced train with unbalanced test, balanced train with unbalanced test, and balanced train with balanced test. The split is 7500 training data and 2500 testing data. "Unbalanced" represents the raw data's ratio of default to paid loans which is about 1 to 9. "Balanced" represents the modified data's ratio of default to paid loans which is about 1 to 1. In general the "unbalanced" ratio is closer to real world situations with less than 5% of default possibility. The reason we modified the raw data to be balanced is to feed the models more data about the default people' information. (We used the radial method for the SVM, 1000 trees and 5 variables for each tree for the Random Forest model, and 1 hidden layer with 4 nodes and 300 epochs for the Neural Network model.)

4. Results and recommendation

K-means Clustering:

1) Not scaled

data.NAME_CONTRACT_TYPE	data.CODE_GENDER	data.FLAG_OWN_CAR
1	1	1
data.FLAG_OWN_REALTY	data.CNT_CHILDREN	data.NAME_FAMILY_STATUS
1	1	1
data.NAME_HOUSING_TYPE	data.REGION_POPULATION_RELATIVE	data.AMT_CREDIT
1	1	2
data.AMT_INCOME_TOTAL	data.AMT_ANNUITY	data.AGE
3	4	5
data.NAME_TYPE_SUITE	data.NAME_INCOME_TYPE	data.NAME_EDUCATION_TYPE
6	6	6

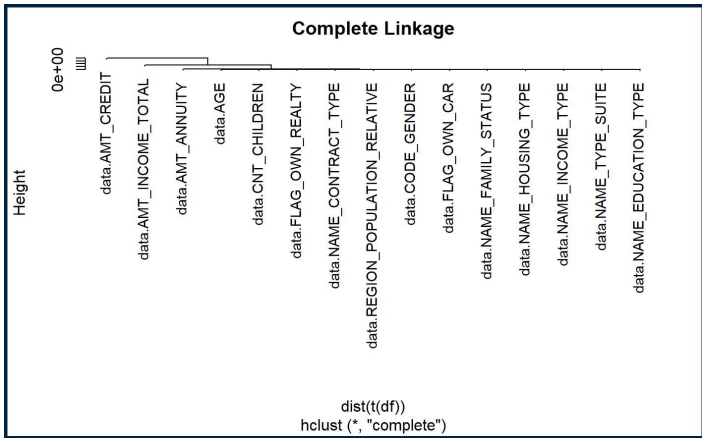
2) Scaled

data.AGE	data.AMT_INCOME_TOTAL	data.NAME_TYPE_SUITE
1	2	3
data.NAME_INCOME_TYPE	data.NAME_EDUCATION_TYPE	data.NAME_CONTRACT_TYPE
3	3	4
data.CODE_GENDER	data.FLAG_OWN_CAR	data.FLAG_OWN_REALTY
4	4	4
data.CNT_CHILDREN	data.NAME_FAMILY_STATUS	data.NAME_HOUSING_TYPE
4	4	4
data.REGION_POPULATION_RELATIVE	data.AMT_CREDIT	data.AMT_ANNUITY
4	5	6

We can see from the results of both clusters that the data has the trend of concentrating towards a singular cluster. In the case of the not scaled data, it is evident that most of the data is in cluster 1. Some of the features that are included in cluster one are Children Count, Realty Status, and Family Status which make logical sense of those features being related to each other; we would expect features of education type and income to be related to each other, but they are placed in different clusters. Regarding the scaled k-means clustering, we can see that most of the data is concentrated in cluster 4. These two clusterings are very similar to each other, most of the data that was clustered in the first observation were clustered together in the second observation as well. It is apparent that the discrepancy between the two methods of K-Means clustering is not a factor in classifying each of the variables.

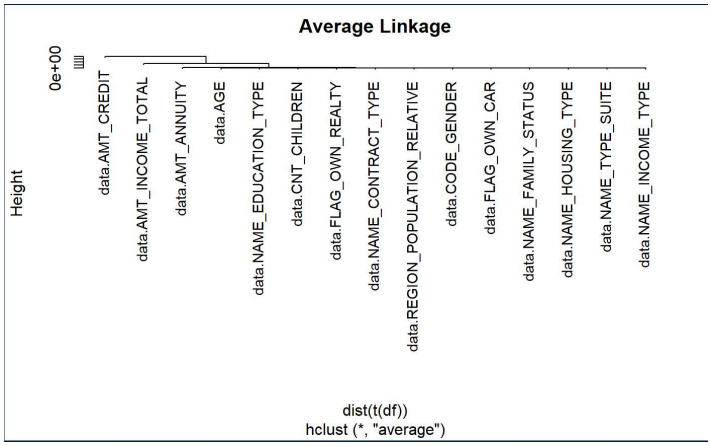
Hierarchical Clustering:

1) Complete



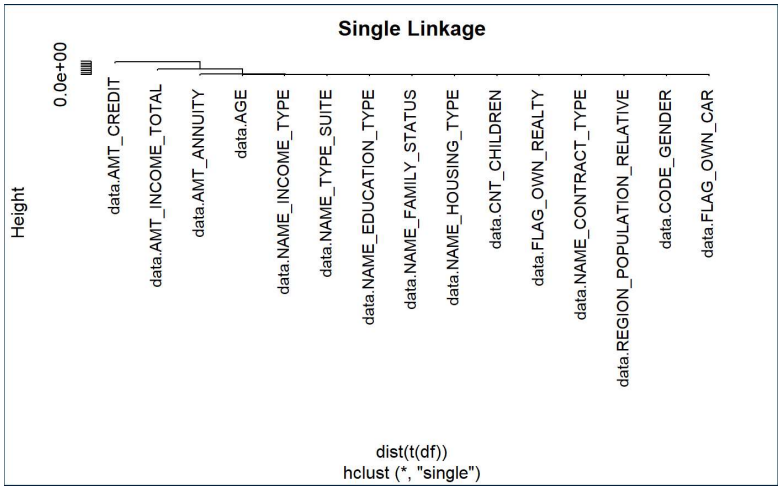
data.NAME_CONTRACT_TYPE	data.CODE_GENDER	data.FLAG_OWN_CAR
1	1	1
data.FLAG_OWN_REALTY	data.CNT_CHILDREN	data.NAME_FAMILY_STATUS
1	1	1
data.NAME_HOUSING_TYPE	data.REGION_POPULATION_RELATIVE	data.AMT_INCOME_TOTAL
1	1	2
data.AMT_CREDIT	data.AMT_ANNUITY	data.NAME_TYPE_SUITE
3	4	5
data.NAME_INCOME_TYPE	data.NAME_EDUCATION_TYPE	data.AGE
5	5	6

2) Average



data.NAME_CONTRACT_TYPE	data.CODE_GENDER	data.FLAG_OWN_CAR
1	1	1
data.FLAG_OWN_REALTY	data.CNT_CHILDREN	data.NAME_EDUCATION_TYPE
1	1	1
data.NAME_FAMILY_STATUS	data.NAME_HOUSING_TYPE	data.REGION_POPULATION_RELATIVE
1	1	1
data.AMT_INCOME_TOTAL	data.AMT_CREDIT	data.AMT_ANNUITY
2	3	4
data.NAME_TYPE_SUITE	data.NAME_INCOME_TYPE	data.AGE
5	5	6

3) Single



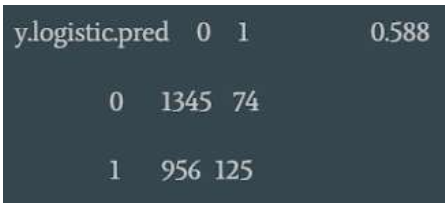
data.NAME_CONTRACT_TYPE	data.CODE_GENDER	data.FLAG_OWN_CAR
1	1	1
data.FLAG_OWN_REALTY	data.CNT_CHILDREN	data.NAME_TYPE_SUITE
1	1	1
data.NAME_EDUCATION_TYPE	data.NAME_FAMILY_STATUS	data.NAME_HOUSING_TYPE
1	1	1
data.REGION_POPULATION_RELATIVE	data.AMT_INCOME_TOTAL	data.AMT_CREDIT
1	2	3
data.AMT_ANNUITY	data.NAME_INCOME_TYPE	data.AGE
4	5	6

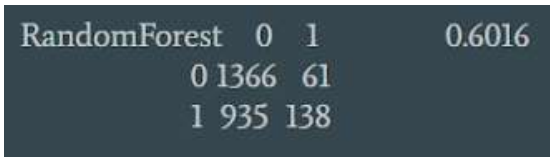
After performing the Hierarchical Clustering, it is apparent that the method of complete, average, or single distance metric is shown to be insignificant in the model. After viewing the different dendrograms, the clustering appears to be the same regarding Credit and Income being in their own branches, and the rest of the data are on its own branch. Even before we cut the tree, it is evident that the clusters for all 3 methods will be similar since the composition of the tree is the exact same. Some noticeable highlights from this cluster demonstrate that Age is its own cluster; this can be explained by the fact that it is non-binary and ranges from 0 to 100, which makes this feature unique compared to the others. Moreover, Income Type, Income Total , Credit, and Annuity are also stand alone features. All of the other features are contained within the first cluster. Overall, the Hierarchical Clustering appears to be the better model considering this model gives more consistent results regarding clustering than the k-means clustering. This idea of which features are related to each other and which features are not related to each other will influence our intuition regarding our Principal Component Analysis.

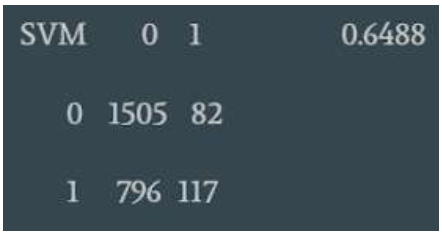
Models: Logistic, SVM, Random Forest, Neural Network

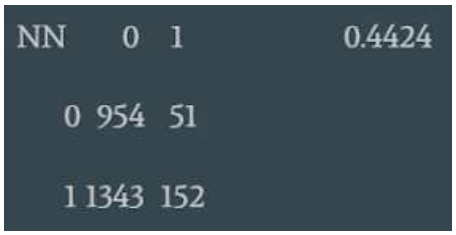
- Unbalanced train with unbalanced test
    - Logistic, SVM, Random Forest, and Neural Network all have 0.92 of accuracy and all of their predictions are 0s.
    - The results make sense since our models were fed with too little information about default people. Also, because the models were fed with unbalanced data, the models have much higher preference on 0 than 1.

- Balanced train with unbalanced test



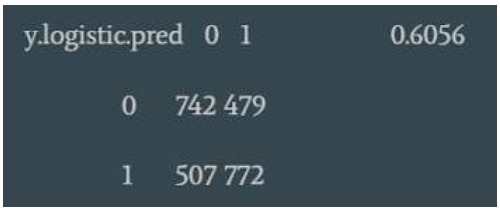


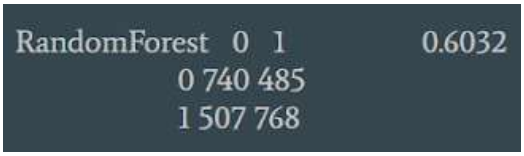


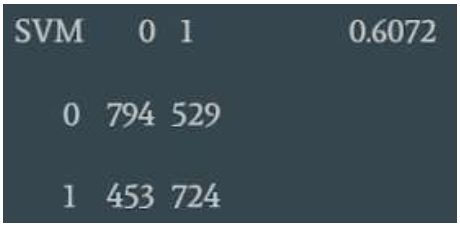


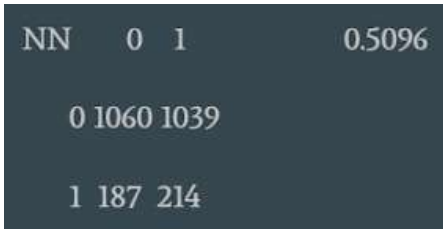
  - With balanced trains, the models now have more information about default people, however, these models tend to have a balanced prediction as well, which is not quite suitable for unbalanced situations. Clearly, SVM performs the best and NN performs the worst. The NN also is very unstable since it sometimes predicts all to be non default.

- Balanced train with Balanced test









  - With balanced trains, the models now have more information about default people. However, the performance of the models in general performed slightly worse than the previous situation. In this case, SVM still performs the best and

NN performs the worst. The NN again is very unstable since it sometimes predicts all to be non default.

5. **Next steps**

With more time we could have used more of the data and could have seen if any of the other columns had feature importance. We could also take a larger number of inputs from unbalanced dataset to see if the results of the models would differ from our current results. More complicated models are needed as well since simple models fail to give decent predictions of rare events like this.

6. **Appendix, Citations and Code**

Contributions of each team member:

Dakshin - Random Forest

Santino - Clustering

Dimitri - Data Cleaning and Analysis

Sid - Logistic Regression and SVM

Zheng - Neural Network

Data used:

https://www.kaggle.com/gauravduttakiit/loan-defaulter?select=columns_description.csv

R CODE FOR CLUSTERING

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
library(quantmod)
```

```
library("e1071")
library(randomForest)
library(keras)
library(nnet)
```

```{r}
data = read.csv("app_data_encoded.csv")
data = na.omit(data)
for (i in 1:ncol(data)){
  data[,i] = as.numeric(data[,i])
}
```

```{r}
df = data.frame(data$NAME_CONTRACT_TYPE,data$CODE_GENDER,data$FLAG_OWN_CAR,

data$FLAG_OWN_REALTY,data$CNT_CHILDREN,data$AMT_INCOME_TOTAL,data$AMT_CREDIT,data$AMT_ANNUITY,
                data$NAME_TYPE_SUITE,data$NAME_INCOME_TYPE,data$NAME_EDUCATION_TYPE,

data$NAME_FAMILY_STATUS,data$NAME_HOUSING_TYPE,data$REGION_POPULATION_RELATIVE,data$AGE)

km.equity = kmeans(t(df),6,nstart=20)

sort(km.equity$cluster)
```

```{r}
scaled_data = scale(t(df))
km.equity = kmeans(scaled_data,6,nstart=20)

sort(km.equity$cluster)
```

```{r}
hc.complete=hclust(dist(t(df)),method="complete")
plot(hc.complete,main="Complete Linkage",cex=.9)
sort(cutree(hc.complete,6))
```

```{r}
#insert r code here
hc.average=hclust(dist(t(df)),method="average")
plot(hc.average,main="Average Linkage",cex=.9)
sort(cutree(hc.average,6))
```

```{r}
#insert r code here
hc.single=hclust(dist(t(df)),method="single")
plot(hc.single,main="Single Linkage",cex=.9)
sort(cutree(hc.single,6))
```