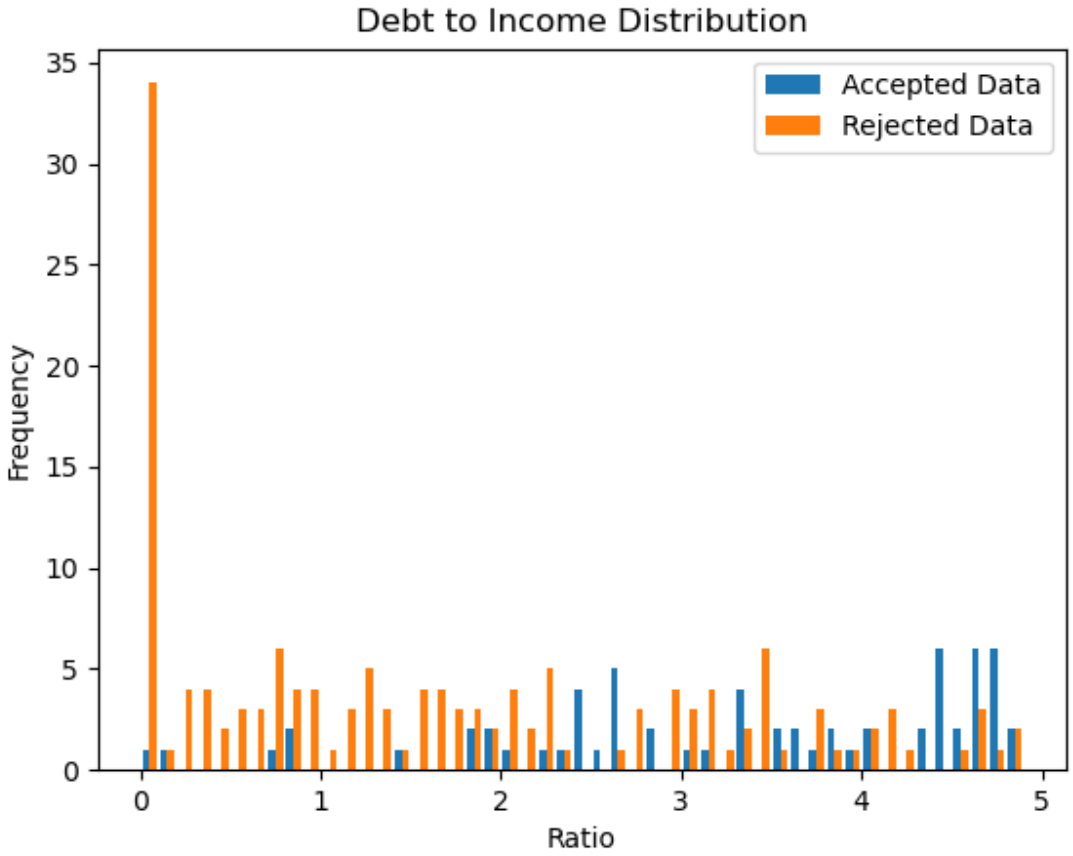
Team 25 Report

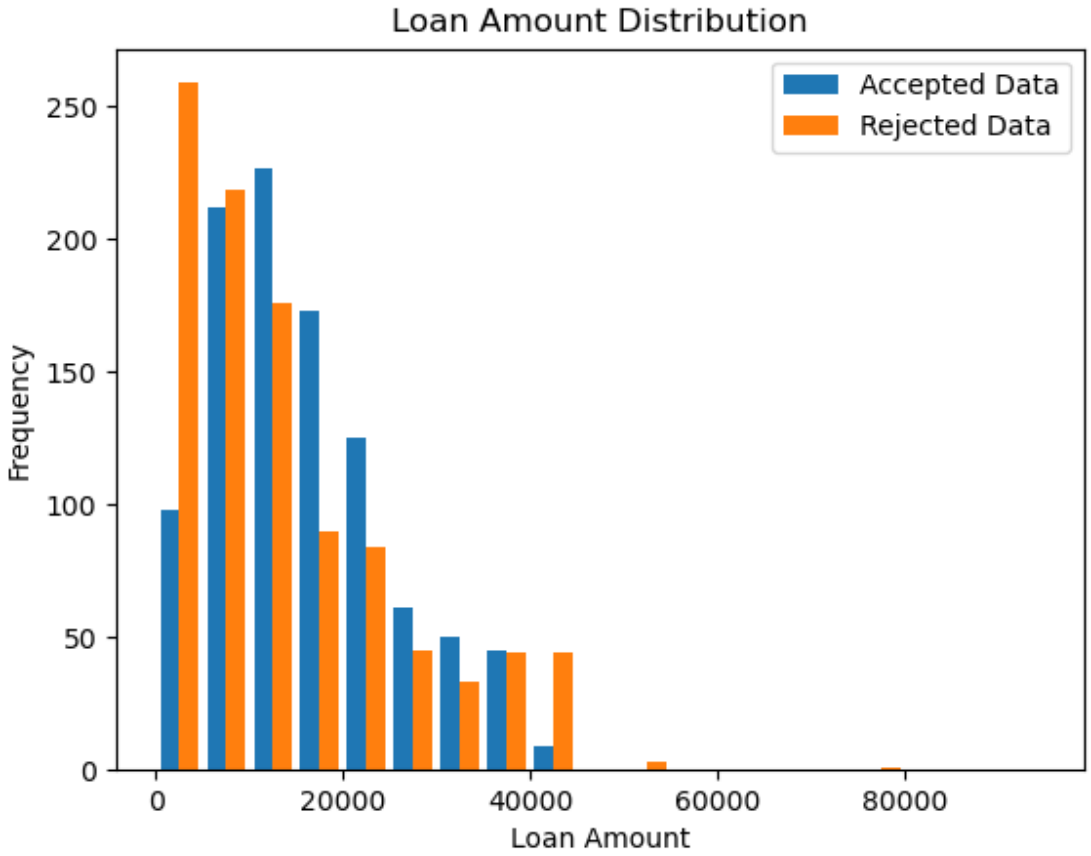
Ellia Yang, Anh Nguyen, Divya Shyamal, Michael Chen

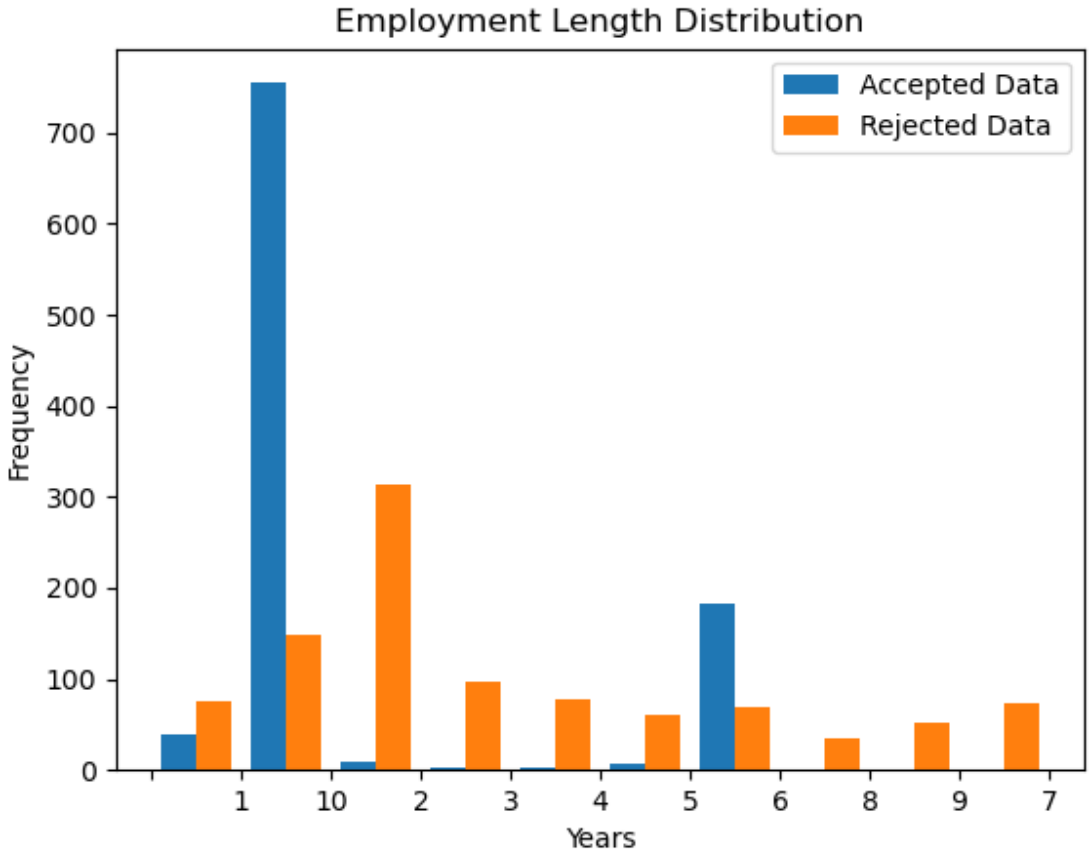
July 24, 2022

Executive Summary

Due to the constraint of time and bad planning, we did not have time to write an executive summary. Alternatively, we offer pretty graphs of questionable quality


Debt to Income Distribution
Accepted Data
Rejected Data
Frequency
Ratio
0
5
10
15
20
25
30
35
0
1
2
3
4
5


Loan Amount Distribution
Accepted Data
Rejected Data
Frequency
250
200
150
100
50
0
0
20000
40000
60000
80000
Loan Amount


Employment Length Distribution
Accepted Data
Rejected Data
Frequency
0
100
200
300
400
500
600
700
1
10
2
3
4
5
6
8
9
7
Years

Technical Exposition

Introduction

Lending club is a financial institution headquartered in San Francisco. It is a peer to peer lending platform and also offers loan trading on a secondary market.

This section delineates the components of the modeling problem and their objectives. Global assumptions applying to the entire modeling process are also listed.

Problem Restatement

In this report, we aim to build machine learning models for predicting loan grades; more specifically, we set out to solve these problems:

1) What are the most determining factors of loan grades?

2) What is the population distribution by zip code area?

3) What is the income distribution by zip code area?

4) What is the race distribution by zip code area?

5) Building machine learning models to predict loan grades

Multi-class Classification Model

Model Assumptions

1. **Data that is not quantitative and is assigned for organizational purposes is not impactful on loan grade prediction.** Variables that exemplify this include: id, member_id, and url. These variables were subsequently not included in the model.
2. **Qualitative data that was not categorized would require a degree of sentiment analysis that was beyond the scope of this model.** Variables that exemplify this include: emp_title, title, and desc. These variables were subsequently not included in the model.
3. **Data that was missing or blank for 98% or more of the sample was is not considered to be significant.** Most of the variables that satisfied this requirement were pertaining to hardship plans, resulting in the same percentage of blank rows.
    a. `hardship_end_date: 99.219%`
    b. `hardship_start_date: 99.219%`
    c. `payment_plan_start_date: 99.219%`
    d. `sec_app_earliest_cr_line: 100%`
    e. `hardship_type: 99.219%`
    f. `hardship_reason: 99.219%`
    g. `Hardship_loan_status: 99.219%`
4. **In the sample of data that was used to train the model, the loans were issued within the same timeframe.** This made the variable issue_d irrelevant in the calculations.
5. **Because of time and equipment constraints, analysis is unable to be performed on the entirety of the data set.** Since Google Colab has a limit of 12 GB RAM and the personal computers that we had access to had 8 GB RAM, many calculations were limited by the amount of memory available.

Model Development

This model was developed in Google Colab using Tensorflow and Keras.

Data Cleaning and Treatment

The data from Lending_Club_Accepted_2014_2018.csv was first cleaned of variables that were to be omitted as described above. We then assigned int values to variables that were not already of type float64. This was done by hand by running

```
trainingSet[variable].value_counts()
```

for variables that were listed on the data schema as being of type object and then making note of the possible categories in the set. Once the possible values were known, we used the `replace` function to turn the possible categories that we had found into integer values. The exact integer values were chosen based mostly on the frequency of the individual categories. For example, for the home_ownership variable, the most frequent category was MORTGAGE and there were six total categories found. Therefore, MORTGAGE was replaced with the integer value 6. Other options

were considered in order to make the categories more on equal footing, however, with the volume of variables that needed to be treated, this method seemed to be the most adaptable. Categories were never assigned the value of 0, with the exception of variables that only had options of yes or no. In those cases, no was assigned to 0 and yes was assigned to 1.

Variables that could not be treated in this matter, such as dates, were simplified to their year, which was an easily extractable integer value from the original data using the formula:

```
trainingSet[variable]=trainingSet[variable].apply(lambda x:
            int(x[-4:]) if pd.notnull(x) else 0)
```

Lastly, the large volume of blank or missing data within the set was treated using the `fillna` function, which was used to replace all blank data with a value of 0.

ML Model

Loan grade and loan subgrade were assumed to have direct correlation, so both were removed from the training set and were used as labeling data. As described above, loan grade and loan subgrade data was categorically treated in a similar manner with a slight difference in that the the categories were not based on frequency, and the lowest category was given 0 instead of 1. For example, for loan subgrades, A1 was assigned a value of 34, while G5 was assigned a value of 0. For loan grades, A was assigned a value of 6, while G was assigned 0. The reason for this was because the ML model was set up to return values [0, 35).

Attempts were made to scale the values in the dataset using `sklearn.preproccessing`, however this had no visible effect on the model, so this step was omitted. Other attempts to change the performance of the model included using the `to_categorical` Keras function and one hot encoding the labels, however these resulted in errors that were not able to be resolved. Therefore, we decided to just take the step to convert both the data and labels to NumPy arrays with values of type float32.

The final model for loan grading was a sequential model with four Dense layers and activation of ‘relu’ on the first three and ‘softmax’ on the last one:

```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dense(7, activation = ‘softmax)
])
```

It was compiled using the ‘adam’ optimizer and with a SparseCategoricalCrossentropy loss:

```
model.compile(optimizer='adam',

loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

The model was fitted with 10 epochs because of equipment limitations, with about 30% accuracy on each epoch:

```
model.fit(trainingSet, loanGrades, epochs=10)

Epoch 1/10
3125/3125 [==============================] - 15s 4ms/step - loss: 176.4264 - accuracy: 0.3409
Epoch 2/10
3125/3125 [==============================] - 9s 3ms/step - loss: 2.8567 - accuracy: 0.3145
Epoch 3/10
3125/3125 [==============================] - 9s 3ms/step - loss: 1.6759 - accuracy: 0.3053
Epoch 4/10
3125/3125 [==============================] - 9s 3ms/step - loss: 1.5820 - accuracy: 0.3049
Epoch 5/10
3125/3125 [==============================] - 9s 3ms/step - loss: 1.5853 - accuracy: 0.3053
Epoch 6/10
3125/3125 [==============================] - 8s 3ms/step - loss: 1.5774 - accuracy: 0.3037
Epoch 7/10
3125/3125 [==============================] - 9s 3ms/step - loss: 1.5777 - accuracy: 0.3035
Epoch 8/10
3125/3125 [==============================] - 9s 3ms/step - loss: 1.5779 - accuracy: 0.3047
Epoch 9/10
3125/3125 [==============================] - 9s 3ms/step - loss: 1.5777 - accuracy: 0.3052
Epoch 10/10
3125/3125 [==============================] - 9s 3ms/step - loss: 1.5776 - accuracy: 0.3049
<keras.callbacks.History at 0x7f1488840910>
```

While knowing that accuracy is supposed to improve with each epoch, we were unable to determine the cause for the steady accuracy rate through each epoch. Achieving an accuracy rate about 30% on 7 categories (for normal loan grades, which range from A-G), means that our model is about twice as accurate as if categories were randomly assigned. Random assignment would have an accuracy rate of 14.29%. The same was true when running the model using loan subgrades as the labels:

```
model.fit(trainingSet, loanGrades, epochs=10)

Epoch 1/10
3125/3125 [==============================] - 10s 3ms/step - loss: 143.5999 - accuracy: 0.0641
Epoch 2/10
3125/3125 [==============================] - 9s 3ms/step - loss: 3.4605 - accuracy: 0.0656
Epoch 3/10
3125/3125 [==============================] - 9s 3ms/step - loss: 4.2531 - accuracy: 0.0656
Epoch 4/10
3125/3125 [==============================] - 9s 3ms/step - loss: 3.1699 - accuracy: 0.0653
Epoch 5/10
3125/3125 [==============================] - 9s 3ms/step - loss: 3.1697 - accuracy: 0.0658
Epoch 6/10
3125/3125 [==============================] - 9s 3ms/step - loss: 3.1697 - accuracy: 0.0670
Epoch 7/10
3125/3125 [==============================] - 10s 3ms/step - loss: 3.1697 - accuracy: 0.0650
Epoch 8/10
3125/3125 [==============================] - 9s 3ms/step - loss: 3.1697 - accuracy: 0.0651
Epoch 9/10
3125/3125 [==============================] - 9s 3ms/step - loss: 3.1697 - accuracy: 0.0644
Epoch 10/10
3125/3125 [==============================] - 9s 3ms/step - loss: 3.1696 - accuracy: 0.0656
```

When fitting the model to loan subgrades, the number of nodes in the last layer was changed to 35, and an accuracy of about 6.5% was achieved, compared to the 2.9% from random assignment.

Scoring Model Description

Model Assumptions

1. **Data on loan amount, debt to income ratio, and employment length are parameters that determine the quality of a loan application.** These are the few factors that are available in both accepted and rejected application data that are quantitative data where an increase or decrease in value could affect the riskiness of a loan application, unlike zip code, application date, or loan ID. In the accepted application data file, these quantities are named "loan_amnt," "dti," and "emp_length." In the rejected application data file, these quantities are named "Amount_Requested," "Debt_to_Income_Ratio," and "Employment_Length," respectively.

2. **The median household income of an applicant's zip code area reflects the social economics status of the applicant.** Social economics status is commonly defined as dependent on one's education, income, and job title. The social economics status of an area typically reflects that of the individuals living in the area. (Moss et al.)

3. **Household income data in 2016 dollars (code S1901_C01_012E) from the U.S. Census Bureau's 2012-2016 American Community Survey 5-Year Estimates reflects the financial situation of the Lending Club applicants.** To get an estimate of the income level of the applicants, especially the rejected applicants, we use the first three digits of the zip code provided rather than state data because it better reflects the area of the applicant's address.

1. **The number of households per U.S. zip code from 2014 to 2018 is similar to that of the data collected by the 2010 Census (code S1901_C01_001E).** Since we only have the first three digits of each applicant's zip code, we weight the median household income per zip code by the number of households living in that zip code to determine the median household income in the area of each applicant's address.

2. **The distribution of race per zip code collected in the U.S. 2010 Census reflects that of the applicants of the Lending Club during the 2014-2018 period.** To analyze how an applicant's background may correlate with their loan application acceptance or rejection, we used the first three digits of their zip code to determine their possible race.

Model Development

Using the data available to us in Lending_Club_Accepted_2014_2018.csv, we and Lending_Club_Rejected_2014_2018.csv, we imported the selected data shown in Table 1.

Table 1: Parameters Considered in Scoring Model			
Parameters	Name in Accepted Applications Data	Name in Rejected Applications Data	Variables
Loan amount/Amount requested	loan_amnt	Amount_Requested	L
Debt to Income Ratio	dti	Debt_to_Income_Ratio	D
Zip Code	zip_code	Zip_Code	Z
State	addr_state	State	S
Employment Length	emp_length	Employment_Length	M

We formatted the zip code data to strip the trailing x's and formatted the employment length data to only report integers for ease of later computations. We also stripped the trailing percentage sign from the debt to income ratio data for rejected applicants.

Since we assumed that the loan amount, debt to income ratio, and employment length most indicative of how good a loan application, we took those factors into consideration first. We assumed that a high value in loan amount or debt to income ratio increases the risks of a loan. On the other hand, a high value in employment length decreases the risks of a loan. We could later change these assumptions.

We assign an initial set of weights to the variables as shown in Table 2. Just like the assumptions about how each factor may change the risks of loan, we can later shift the weights.

Table 1: Parameters Considered in Scoring Model		
Parameters	Weight variable names	Weight
Loan amount/Amount requested	l	$1/3$

Debt to Income Ratio	d	1/3
Employment Length	m	1/3

Because a better or worse loan application is relative, we decided to normalize the data with a cumulative sum scaling normalization where we first use numpy's histogram function to find the probability distribution function then use the cumsum function to find the cumulative value. To invert the normalized value for the loan amount and debt to income ratio parameters, we took the difference between 1 and the cumulative value.

Alternatives cumulative sum scaling normalization include using the standard score or min-max feature scalings.

We defined the score as

$$normalized(L) * l + normalized(D) * d + normalized(M) * m$$

Strength and Weaknesses

This model is not made to predict one's likelihood of getting a loan from the Lending Club, nor is it made to simulate the Lending Club's grading of a loan application. It is simply assigning a score to a loan application, and its simplicity gives versatility.

Sensitivity Analysis

Unfortunately, we did not have enough time to conduct a sensitivity analysis of the model. However, if we were to have more time and hardware capability, we would vary the weights and analyze how that would change the score distribution. Then we would try the different ways of normalizations and how we assume each factor would impact the risk of a loan application.

Significant Variables & Random Forest Model to Predict Loan Grades

Feature Selection

In this model, many fewer features were used. To deem which features were to be included in the model, we determined whether each variable was significant in a multivariate linear regression model that predicted loan grades. Specifically, we computed the coefficients of the multivariate linear regression, then assuming that the noise is Gaussian, estimated the variance of the noise. We ran a T-test to determine if the variable was significant to the regression. If the p-value was below .05, we determine that the variable is significant to multivariate linear regression.
We ran this analysis several times, each time recording if a variable is significant. If a variable was determined to be significant at least 70% of the time, we included it in our model.

The features that were included in the model are given in the following list:

- `annual_inc`
- `loan_status`
- `zip_code` (first three digits)
- `earliest_cr_line`
- `fico_range_low`
- `out_prncp_inv`
- `total_pymnt_inv`
- `total_rec_int`
- `total_rec_late_fee`
- `dti_joint`
- `verification_status_joint`
- `num_sats`
- `num_tl_120dpd_2m`
- `total_il_high_credit_limit`

It is worth noting that we tried a multivariate linear regression model with these features, as that was the natural next step. However, a random forest model yielded better accuracy.

On the topic of statistical testing, we performed a Chi-Squared test to check whether certain pairs of variables were correlated - we found that the number of 30+ days past-due incidences of delinquency in the borrower's credit file for the past 2 years (`delinq_2yrs`) and the interest rate assigned to the loan (`int_rate`). While this was very intriguing, we did not have a chance to pursue this further but think it should receive attention.

Model Assumptions

Model assumptions are the same as those of the Multi-class Classification model. We decided to use a Random Forest model as the construction of trees with splits lends itself well to loan data.

Model Development

This model was developed in Google Colab using sklearn.

Using the selected features, we then built a random forest with maximum depth 10 and 100 trees. The accuracy of the model was **43%**.

Data cleaning and treatment are similar to that of the Multi-class Classification model - features were encoded in the same fashion, and grades were converted to numbers using the function `f(x)=10*(int(ord(x[0])-65),` where `int(ord(x[0])` denotes the ASCII value of the grade.

References

Moss, Jennifer L., et al. "Comparisons of individual- and area-level socioeconomic status as proxies for individual-level measures: evidence from the Mortality Disparities in American Communities study." Population Health Metrics, 07 January 2021, https://pophealthmetrics.biomedcentral.com/articles/10.1186/s12963-020-00244-x. Accessed 24 July 2022.