Honghui Wang, honghuiw
Shi Heng Zhang, shz1
Haopeng Wang, hwa125
Chengxi Yang, cya91

Team Quartet

CMPT 732 – BIG DATA PROGRAMMING I

Final Project - Home Credit Default Risk Analysis

Contents

Honghui Wang, honghuiw
Shi Heng Zhang, shz1
Haopeng Wang, hwa125
Chengxi Yang, cya91

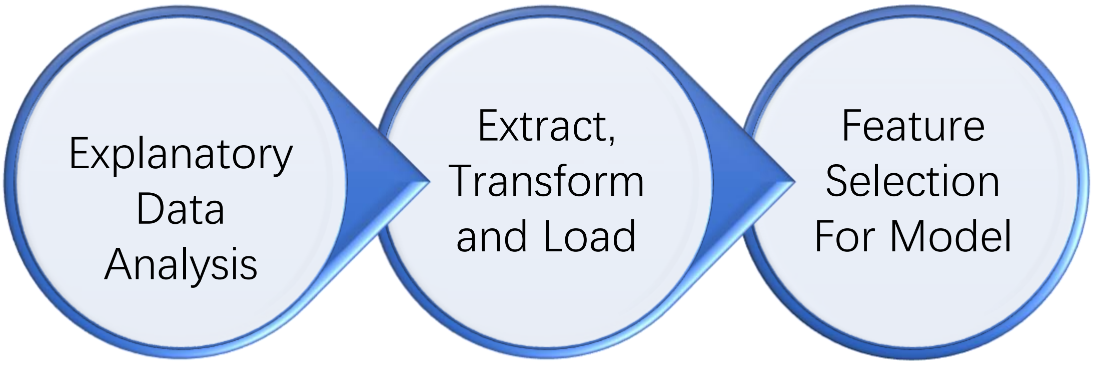
1. Problem Definition and Project Outline

We conducted a Kaggle challenge (https://www.kaggle.com/c/home-credit-default-risk) in the financial domain to help financial institutions identify whether a loan applicant has the ability to repay the loan. We decided to make the best use out of the Big Data Tools that we learned in class to solve practical problems, while learning new technology along the way. The data provided by the Home Credit Group composes of seven csv forms with a total of 2.49 GB.

We used big data tools throughout our project to achieve great scalability. We first performed a thorough Explanatory Data Analysis (EDA) to understand the structure of the dataset, and discovered some valuable patterns that could facilitate financial institution's decision making. Then we conducted the Extract, Transform and Load (ETL) process to properly combine and encode the relational dataset, and generated new features based on EDA results. Finally, we used multiple techniques for feature selection to expedite the model training and avoid overfitting.

The biggest challenge of this problem is that the abundance and sparsity of the information at the start. There are over 300 features in total across all the files, so it requires a certain amount of background knowledge to identify them and make the best use out of the features.

2. Methodology



Explanatory Data Analysis

- Loading data to Cassandra
- Finding outliers and missing values
- Various feature comparison and finding the ones with the most noticeable trends

Extract, Transform and Load

- Transform into one table without losing information
- Aggregating numerical feature
- Counting non-numerical feature

Feature Selection For Model

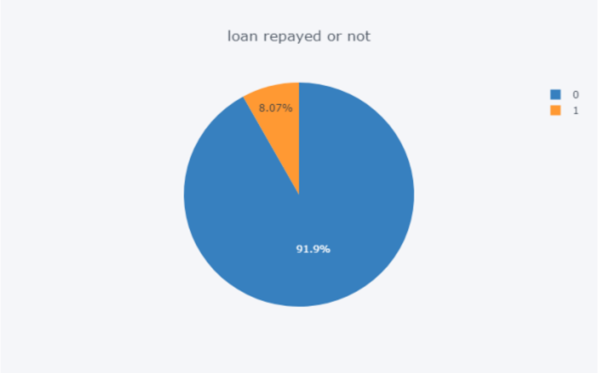
- Get rid of columns with collinearity, high missing value percentages, and zero importance
- Tuning the model with different parameters

2.1 Explanatory Data Analysis

The data is composed of seven csv files with a total of 2.49 GB. We first created the tables in Cassandra, one table for each file, to simulate a situation where a banking/loan agency stores massive financial data in a no-relational database. There is a total of more than 300 columns, so we had to manually mark column data types and then write a Python script to generate CQL for table creation.

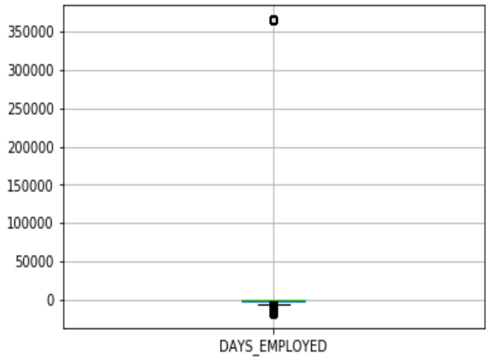
We used the following steps and interactive plotting tools to make sense of the data and support financial institutions' decision making.

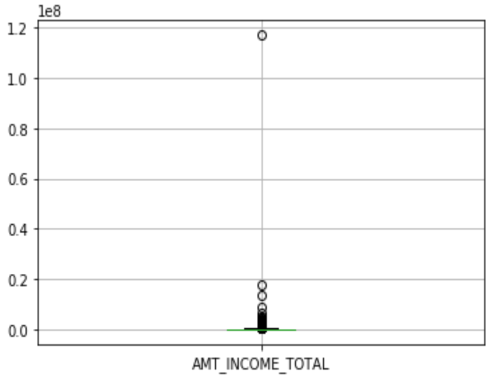
1. Target Structure:



0: repay loans
1: do not repay loans
We can clearly see it really is an extremely unbalanced dataset, and since it is also a binary classifier, we used the area under Receiver Operating Characteristic (ROC) curve as the metric of our model to evaluate the model performance.

2. Outlier Detection:

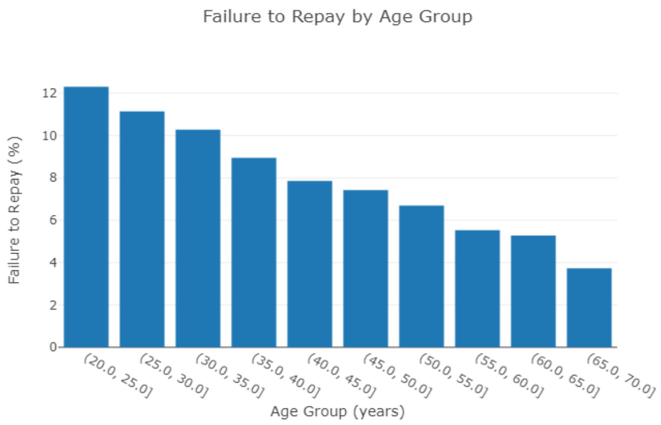




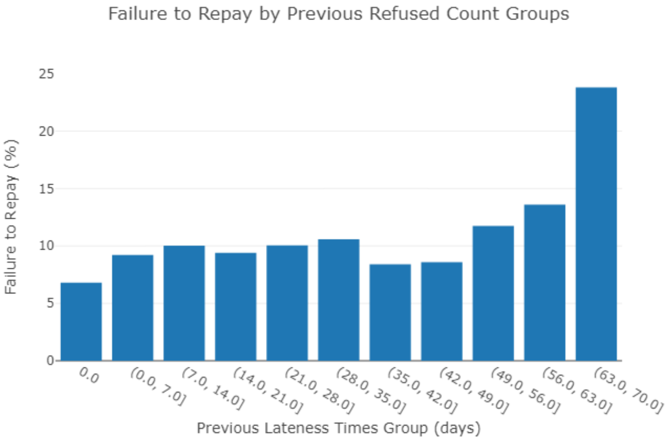
From the dataset, we discovered there are people who have an income close to one billion and there are people who are employed for about a hundred years. Obviously, those data are invalid outliers, so we have decided to ignore those data points for future analysis. Also, we have noticed that about half of the features have missing values, and there are columns have more than 75% missing values.

3. Feature Exploration in terms of Repayment Ability

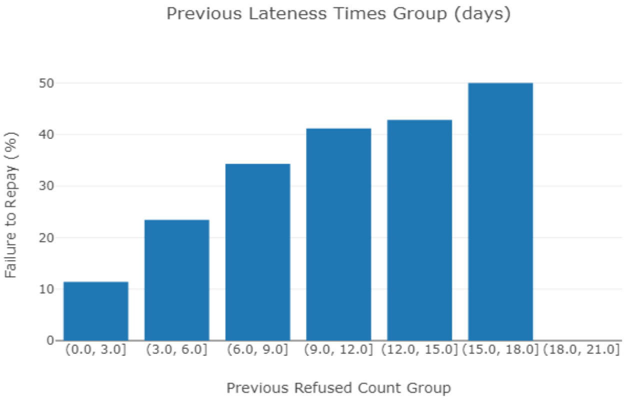
We have tried many column combinations, and we will show a few that has the most noticeable trends:



This is the breakdown of failure of payment vs age groups, we can clearly see that as age increases, the chance of not repay decreases.



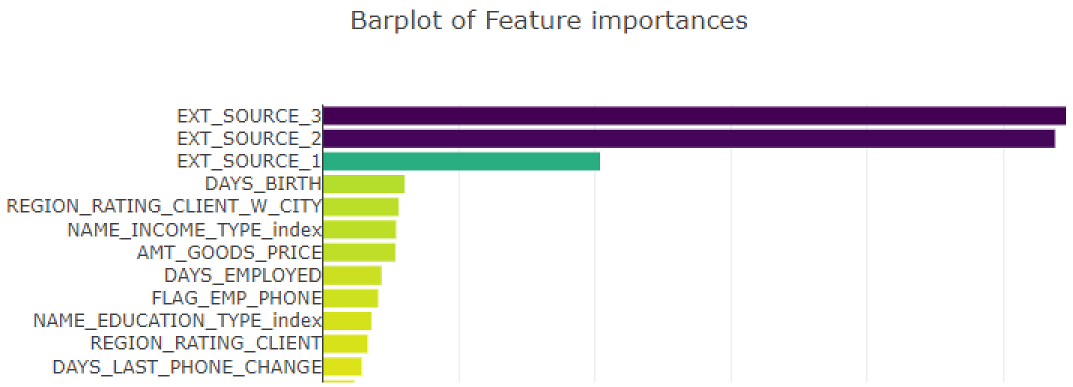
The dataset consists of information from previous applications. We can clearly see that when people constantly miss their payments for more than a month, the chance to not repay shoots straight up.



The dataset consists of information from the Credit Bureau. We can clearly see that when people start to loan money constantly from various sources within 90 days, the chance to not repay shows a steady increase. They might be "robbing Peter to pay Paul".

This is just a small glance of all the Feature Exploration we have done. Please check the EDA.ipynb/EDA.html to see all of the work with interactive graphs.

4. Feature Importance Exploration:



Full list is also in EDA.ipynb/EDA.html. This process helped us to select features for our information exploration. We explored all the features with a high importance score to discover their relationship with client's repayment ability. This is just the top twelve of the list. Note that all the "EXT_SOURCE_1/2/3" are external sources, and we have no information about what these features are. We skipped exploration for such features even though they have high importance. We guess these features might be credit reports provided by third parties.

2.2 Extract, Transform and Load

In order to train the model, we need to integrate all the columns into one single table for the model to take in and train on. The most challenging part of our dataset is that, one current application ID (SK_ID_CURR) might be associated with several previous application IDs (SK_ID_PREV) or several bureau information (SK_ID_BUREAU). It is not really feasible to just add several thousand rows and columns into one table (number of associated ID times the number of columns for each). Therefore, after several failures and debating, we have decided to take different approaches for numerical columns and categorical columns for the sake of preserving information. For a particular categorical feature, we simply break each category into a separate column and then count the number of appearances that certain category appeared. For numerical columns, we calculate the mean, min, max, count and sum of the numbers associated with one current ID. For example, a person with the same SK_ID_CURR might have the following data for SK_ID_PREV in the previous_application table

SK_ID_CURR	SK_ID_PREV	name_contract_type	amt_annuity
331189	2370256	Cash loans	42,860.34
331189	2370282	Consumer loans	46,860.83
331189	2392282	Cash loans	27,260.92
331189	2482282	Cash loans	38,260.19

After we have applied the ETL method, this is the result

SK_ID_CURR	Count(previous_application_name_contract_type_Cash loans)	Count(previous_application_name_contract_type_Consumer loans)	Count(previous_application_name_contract_type_Revolving loans)
331189	3	1	0

This is one row but breaks into two parts(all associated with the same SK_ID_CURR)

Count (previous_application_amt_annuity)	Sum (previous_application_amt_annuity)	Max (previous_application_amt_annuity)	Min (previous_application_amt_annuity)	Avg (previous_application_amt_annuity)
4	27,260.92	46,860.83	155,242.28	38,810.57

Since the POS_CASH_balance, instalments_payments, and credit_card_balance are monthly records that are associated with the previous_application table, we applied above techniques and performed left outer joins on the previous_application in order to preserve information. We have also applied the same ETL method to the bureau information. Since bureau balance has 817,395 bureau ids, while bureau has 1,716,428 bureau ids, we performed a left outer join on the bureau table to avoid any data loss. Aside from these, we have also added several columns from our domain knowledge that would help the model in the decision-making process.

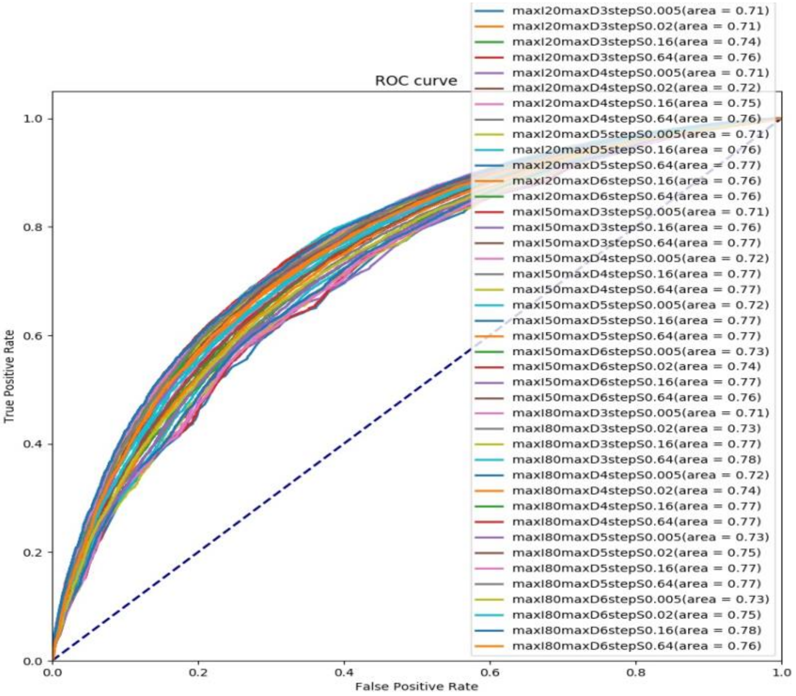
After we have completed the ETL process for Bureau data and Previous data, we performed a left outer join on the application_train/test tables to create the ready_train/test tables to be ready for training and testing.

2.3 Feature Selection For Model

After the ETL above, we ended up with 1145 columns with no extra rows than the original application_train table, but it is still so many features. To avoid overfitting and achieve better performance, we applied several methods to eliminate noises and redundant features:

- Collinearity columns: we have removed all the columns but one that has an absolute value of the Pearson r correlation > 0.8
- Missing value columns: we have removed all the columns that has more than 75% missing values
- Zero weight columns: we used Embedded Method (Random Forest) to analyze feature importance and removed all the columns with zero importance

After all that feature selection, we ended up with 471 columns, and this is a much more feasible number of features for model training. As for the tuning of the model, we tried more than 40 combinations of maxIter, maxDepth, and stepSize.



We found out that maxIter affects the result the most. It has a really high positive correlation. More maxIter simply would net a larger area. From the graph, the numbers represent the area under ROC curve, and we can see that we are approaching some sort of asymptote. We choose AUC as the metric simply because that for unbalanced dataset, accuracy would lead to a biased score. After getting the best tuned hyper-parameters from plotting, we tried several combinations around them and used cross-validation to ensure we have found the best model. As a result, we achieved around 0.791 area under the ROC curve.

3. Problems & Challenges

- One of the biggest problem at the start is to make sense of the data features since we do not have much domain knowledge. We spent a long time trying to figure out which columns would have potential relationships and have tried many comparison combinations. In the final EDA, we only selected a few of them that showed clear trends when compared to the target column.

- Another challenge is during the ETL process, we failed many times on how to approach the one-to-many circumstance (one SK_ID_CURR corresponds to many IDs in the other tables). When we finally decided on the solution mentioned above in the ETL section, we found out that Spark does not have the "get_dummies" (unlike Pandas for simple one-hot_encoding), so we wrote our own get_dummy_spark to resolve the issue and tested vastly.

- Last challenge is when we try to visualize the area under the ROC curve while tuning. Since the avgMetrics from the fit only shows a number that represents the area, we had to incorporate sklearn in the process to plot the ROC curve for better visualization and to achieve a better tuning.

4. Results

We were able to achieve the area under the ROC curve to be around 0.791. We consider it to be a fair performance in predicting whether or not a person would repay the loan on time according to his or her background information. Another use for this prediction is that financial institutions can utilize the EDA results to expand business and reach potential customers who they previously do not consider as appropriate clients but usually would repay the loan on time, like students/businessman. From the data analysis, we learned many interesting aspects of the finance and banking field. We tried to use big data tools as many as possible to achieve great scalability. From the implementation, we learned the whole process from receiving the data, cleaning, analyzing, and visualizing the data, to finally tuning the model to have the best result.

5. Project Summary

Category	Points
Getting the data: Acquiring/gathering/downloading.	0
ETL: Extract-Transform-Load work and cleaning the data set. Finding outliers and columns with more than 75% missing values, as well as aggregating columns for training purposes	4
Problem: Work on defining problem itself and motivation for the analysis. It is really close to our daily lives, and we can really utilize all the Big Data Tools learned in class	2
Algorithmic work: Work on the algorithms needed to work with the data, including integrating data mining and machine learning techniques. Tuned the model with maxIter, maxDepth, stepSize, and plotted the ROC graph to find the best parameter	4.5
Bigness/parallelization: Efficiency of the analysis on a cluster, and scalability to larger data sets. Utilizes Spark and Cassandra, all of those are Big Data Tools, so it is very scalable	3
UI: User interface to the results, possibly including web or data exploration frontends. Interactive graphs for the EDA	0.5
Visualization: Visualization of analysis results. EDA visualized a lot of the combinations of features vs the target column as well as feature weights	5
Technologies: New technologies learned as part of doing the project. Used Plotly and some Pandas only for plotting, Sklearn for visualizing ROC Curve	1
Total	20