Agency Loan Default Probability Prediction

Guanjie Huang, Zhening Liu, Jiada Chen
Department of Computer Science
Columbia University
e-mail: {gh2396, zl2578, jc4730}@columbia.edu

Abstract—**This paper evaluates the performance across a number of machine learning algorithms for agency loan default probability prediction. Specifically, Gradient Boosting, Logistic Regression, Random Forest, GLMNET and Neural Network are used to train the Freddie Mac single family loan dataset. The paper will detail the data preprocessing steps in order to handle such large dataset, and the model selection criteria when dealing with highly imbalanced class. In this particular dataset, GLMNET has shown the most favored result among all other classification methods via various measures. For further business value, the ROC curve could be a tool used for banks or regulators to to maximize model true positive rate for an acceptable false positive rate level given their risk appetite.**

Keywords -- Agency Loan; Default; Machine Learning; Severely Imbalanced Class

I. Introduction

After 2009, the mortgage market has dramatically changed. New issued loans have a significantly higher credit quality, and present different default pattern.

Traditional mortgage approval processes are based on hard-coded rules on loan attributes such as FICO scores, Loan-to-Value (LTV) ratio, Debt-to-Income (DTI) ratio, application document completeness etc. In addition, the widely used default rate prediction methodology relies heavily on dynamically changing macro data and future projection of variables based on a monthly transition rate analysis.

In response to the above two points, our goal is to develop a flexible non-linear classifier to predict the possibility that a borrower will default within the first 5 years since the origination of the loan. In addition, to be able to predict such probability at loan origination date or even in the application process to reduce projection difficulty. The algorithm could help mortgage issuance agencies to decide whether to write a new loan the borrower.

This paper will begin with Section II to review previous works related to topics in loan default prediction, Section III to describe the system design and dataset overview for this project, Section IV to show the machine learning algorithm and model selection criteria used, Section V to demonstrate the usage of our model in other application, Section VI to discuss the results from the machine learning algorithm, and Section VII to draw a conclusion and envision the future works from this project.

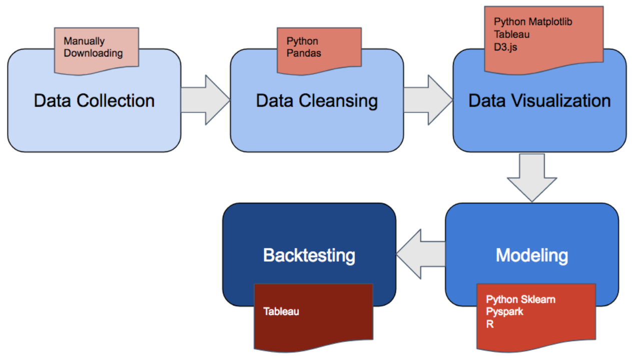
II. Related Works

The mortgage loan performance has long been studied using various machine learning and data mining methods, from very old and simple logistic regression [1], to newer and more complex models such as neural networks and deep learning [2]. Also, metrics such as AUC under ROC curve has been shown as effective model performance measure [3].

In our work, we would like to apply additional machine learning methods of our choice, such as GLMNET, and examine the results using innovative model performance metrics, such as KS Test and Actual vs Model Probability, and backtest the result.

III. System & Data Overview

3.1 System Diagram



3.2 Data Elements
The Freddie Mac Single Family FIX 30 Loan-Level Dataset is the main component of our original dataset. Mortgages originated between Q4 2009 and Q4 2011 are our target population for the post-crisis loan default rate analysis purpose. The Freddie Mac data for each quarter is separated into two files: Acquisitions and History. The acquisition file is the metadata of individual loans, including fields such as original unpaid balance, maturity date, LTV ratio etc. It also includes borrowers' characteristic, such as FICO score, DTI ratio etc. The history data includes the amortization schedule of the loan, and variables that we could use to derive the default flag. We mainly extract loan's attribution information from the acquisition dataset and derive the default flag from the history dataset. In our model, we define default event as loans' ever delaying 5 months payment or going through the

foreclosure, REO or repurchased process during the first 5 years after origination.

In addition, the Federal House Price Index (HPI) quarterly data at MSA level is used for supplement. We were able to derive the max HPI up move and max HPI down move at loan level. For loans has missing or unavailable MSA, we are using state level HPI as proxy. Since house price change is one of the main reasons of default, HPI is an important feature to the our dataset.

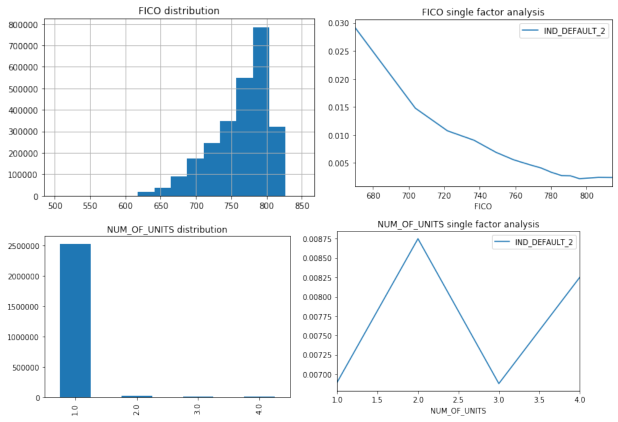
All raw data files combined is around 20GB. After merging, data cleansing, the remaining dataset is 250MB with 2.5 million single family loans.

3.3 Exploratory Data Analysis and Variable Selection

Exploratory data analysis is done using the cleansed dataset. Various data visualization techniques are used to help us understand the data.

3.3.1 Single Factor Analysis

We have done single factor analysis for each feature variable. In this way we are able to quickly identify any outliers and decide whether further data preprocessing such as data normalization is needed. Visualization was done using matplotlib in Python. For example, below are the single factor analysis graphs for FICO and Number of Unit, respectively.



One action, for example, we have done after inspecting above graph is to shrink the number of units down to either 1 or more than 1, since including other values adds little information.

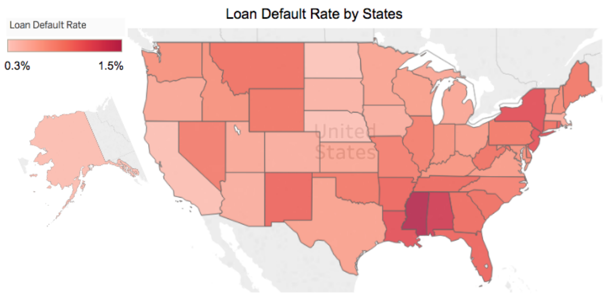
Also by binning continuous variables into smaller buckets, then calculating the average default rate for each one, we could get some intuition if default events is correlated with this variable, if no monotonicity or trend was found, we probably will not feed this variable into the model.

To see the complete single factor analysis results, please refer to the ‘variable_selection.zip’ file.

All the records are splitted into training and test data set and the ratio is accordingly 80% and 20%. And all the feature variables are normalized before the model training step. This is important for algorithms not based on decision trees such as logistic regression.

3.3.2 Default Rate Analysis

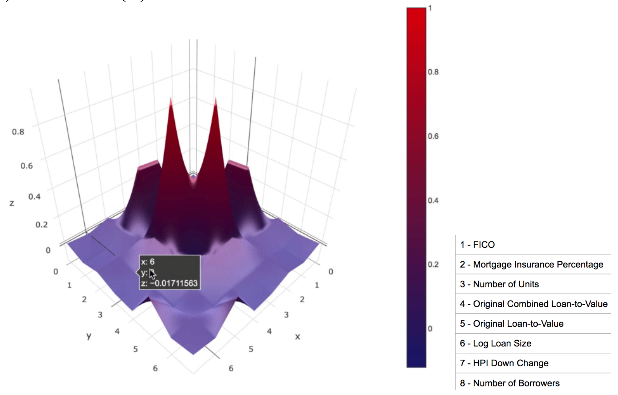
In addition to all the feature variables, we also took a look at the Y variable - default flag. The overall default rate is 0.7%. A choropleth map with default rate by state is done using Tableau. We can see the default rate for each state varies.



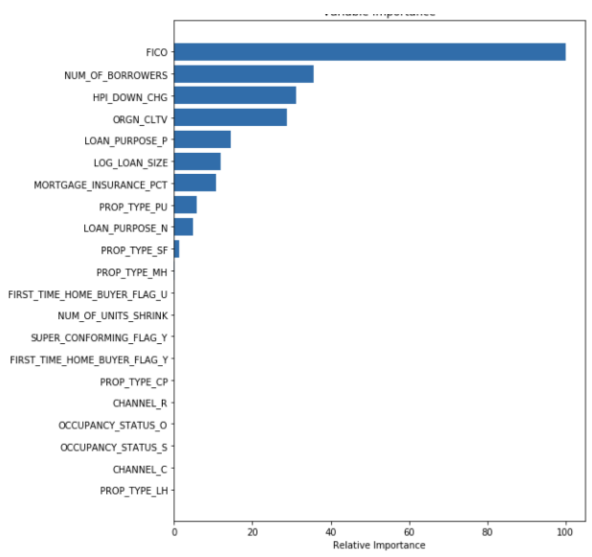
An average default rate of 0.7% is very challenging from the modeling perspective as the predicted default probability is likely to be very small. And actually in the chapter IV, the largest predicted probability among all algorithms we have tried is smaller than 50%. So the predicted value of default flag will be all zero, non-default. In such case, the classification error cannot be used as our model evaluation criteria, in chapter VI, we will discuss the tests used for model selection and evaluation.

3.3.3 Correlation Matrix and Variable Selection

A visual correlation matrix using Javascript and HTML helped us easily identify redundant features such as CLTV (4) and LTV(5).



We proceeded further with variable selection by applying Gradient Boosting algorithm and inspected the variable importance plot. The following is the result by running a GBM with depth 2. Only the top features were selected to feed into our models. Finally we chose FICO, mortgage insurance percentage, original cltv, log loan size, hpi largest down change, number of borrowers , product type and loan purpose as our feature variables.



IV. Algorithm

4.1 Logistic Regression

Logistic regression without regularization is served as our benchmark algorithm since it is fast to train and yield a linear boundary. If the competent model does not perform better than logistic regression, it is regarded as an unsuccessful one.

Logistic regression is implemented using Python sklearn package and the training time is ranking the fastest among all the other algorithms we have applied.

4.2 Gradient Boosting Method

Gradient Boosting method is applied already in the variable selection step. By tuning hyperparameters of this algorithm in model selection step, we will find the best parameter set for our classification purpose, and the according best result of the test data will be the representative of GBM's score.

Similar to logistic regression, GBM is also implemented in Python sklearn, the training time takes much longer, ranking the 4th among all. Results show some improvement compared with logistic regression and a robust result. One of the advantage of GBM is that it is resistant to overfitting.

4.3 Random Forest Classification

Random Forest algorithm is best known for its ability to deal with imbalanced datasets. Similar as in Gradient Boosting, this algorithm is based on the decision tree learning. The difference between the two is that, instead of deriving several depths of decisions from the whole dataset, the Random Forest algorithm is seeking for the best recommendation through splitted random trees.

This algorithm is a particularly suitable algorithm for the agency loan dataset for two reasons. Firstly, the potentially correlated variables won't be overfitted, since each tree learning is an independent event and final decision is averaged out, reducing the total variance. Secondly, given the extremely imbalanced dataset (0.7% default rate), the Random Forest algorithm is more likely to weight more importance to default loan sets by randomly selecting subsets with more defaults.

Random forest is implemented in pyspark. The running time ranks the third, however the test result does not perform better than logistic regression.

4.4 Neural Network

Neural Network MLP classifier is very unstable given different random seeds. Although the best result of neural network trial is ranking the second among all five. Its unstableness makes this algorithm very inappropriate for this application. The training speed of neural network using Python sklearn is surprisingly high, ranking the second among five.

4.5 GLMNET

GLMNET model is one of the newest and most advanced regression models in the market. It is in fact suitable to apply to most data models due to its flexibility in variable adjustment.

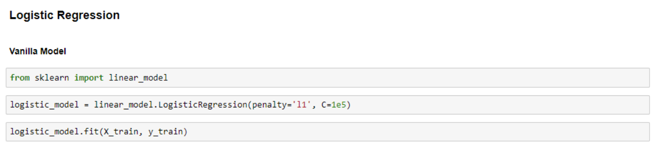
There are a few parameters to tune in this model, for example, the penalty form of l1 and l2. After careful comparison, the use of lasso model has returned the most favorable result.

Since GLMNET package is only hosted on the R repository and R is not specifically designed to process big data, the runtime is relatively expensive, ranking the least efficient algorithm among all. However, the model fitting result is arguably the best under various measures.

V. Software Package Description

5.1 scikit-learn

Sklearn is a machine learning package that is part of the Python library. The main structure of this package is built on NumPy and SciPy. We have used a combination of sklearn functions and pandas to clean our datasets, as well as to train models in **logistic regression**, **gradient boosting** and **neural network**.

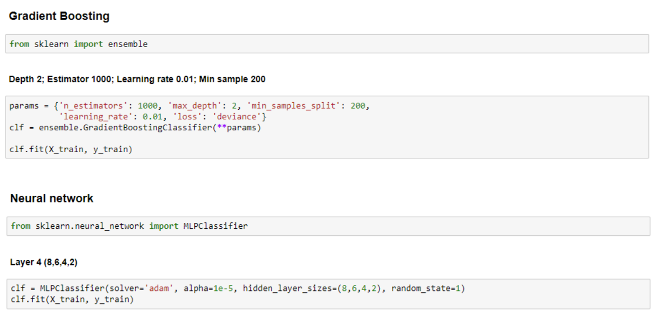

Logistic Regression

Vanilla Model

```
from sklearn import linear_model
```

```
logistic_model = linear_model.LogisticRegression(penalty='l1', C=1e5)
```

```
logistic_model.fit(X_train, y_train)
```



Gradient Boosting

```python
from sklearn import ensemble
```

Depth 2; Estimator 1000; Learning rate 0.01; Min sample 200

```python
params = {'n_estimators': 1000, 'max_depth': 2, 'min_samples_split': 200,
          'learning_rate': 0.01, 'loss': 'deviance'}
clf = ensemble.GradientBoostingClassifier(**params)

clf.fit(X_train, y_train)
```

Neural network

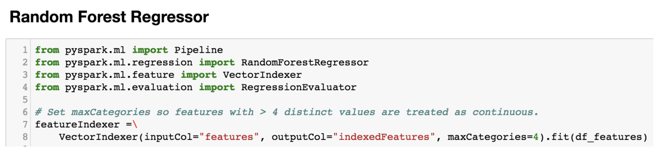
```python
from sklearn.neural_network import MLPClassifier
```

Layer 4 (8,6,4,2)

```python
clf = MLPClassifier(solver='adam', alpha=1e-5, hidden_layer_sizes=(8,6,4,2), random_state=1)
clf.fit(X_train, y_train)
```

5.2 PySpark

PySpark is a Python API that exposes to Spark built-in models. Spark is well known for its agility in computing big data. Given the size of the dataset in this project, PySpark is particularly suitable as tool. Classification methods such as Naive Bayes and **Random Forest Classifier** were trained on PySpark.



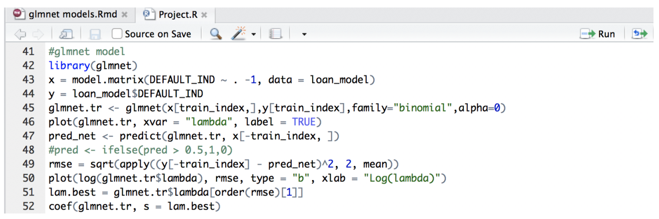
Random Forest Regressor

```python
from pyspark.ml import Pipeline
from pyspark.ml.regression import RandomForestRegressor
from pyspark.ml.feature import VectorIndexer
from pyspark.ml.evaluation import RegressionEvaluator

# Set maxCategories so features with > 4 distinct values are treated as continuous.
featureIndexer =\
    VectorIndexer(inputCol="features", outputCol="indexedFeatures", maxCategories=4).fit(df_features)
```

5.3 R

R is an open source language that is designed for statistical computing. There are various statistics research based on R and it in fact contains the well advanced models in the market. ***GLMNET*** *is a good example of matured and sophisticated models that R can offer. It turned out to be the best model in this project as well.*



```r
#glmnet model
library(glmnet)
x = model.matrix(DEFAULT_IND ~ . -1, data = loan_model)
y = loan_model$DEFAULT_IND
glmnet.tr <- glmnet(x[train_index,],y[train_index],family="binomial",alpha=0)
plot(glmnet.tr, xvar = "lambda", label = TRUE)
pred_net <- predict(glmnet.tr, x[-train_index, ])
#pred <- ifelse(pred > 0.5,1,0)
rmse = sqrt(apply((y[-train_index] - pred_net)^2, 2, mean))
plot(log(glmnet.tr$lambda), rmse, type = "b", xlab = "Log(lambda)")
lam.best = glmnet.tr$lambda[order(rmse)[1]]
coef(glmnet.tr, s = lam.best)
```

VI. Experiment Results

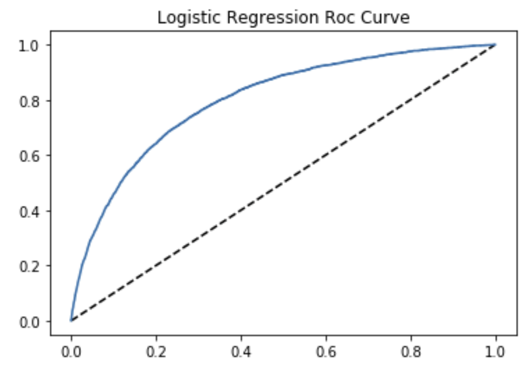
As mentioned in 3.3.2, classification error is not a good measure for this application. Below are tests implemented, the first three are used for model selection and evaluation. And the rest is used to further check model's performance. For the complete test results, please refer to the 'model_selecion.zip'.

6.1 AUC (ROC Curve)

AUC is one of the most commonly used measure for classification algorithm. The definition of the ROC curve can be viewed as a trade-off between the true positive rate and false positive rate. Below is the table of AUC for 5 algorithms with different hyperparameters. The guideline for using AUC is generally the larger the better.

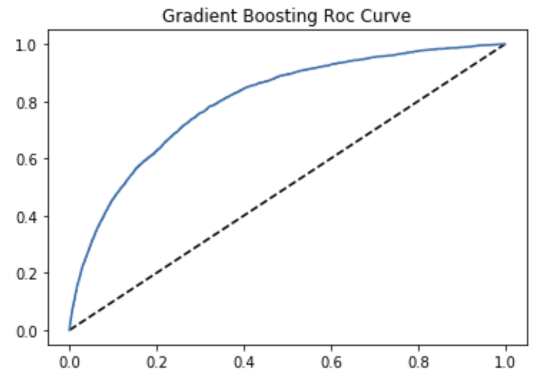
6.1.1 Logistic Regression

```
AUC is 0.7997959988562249
```



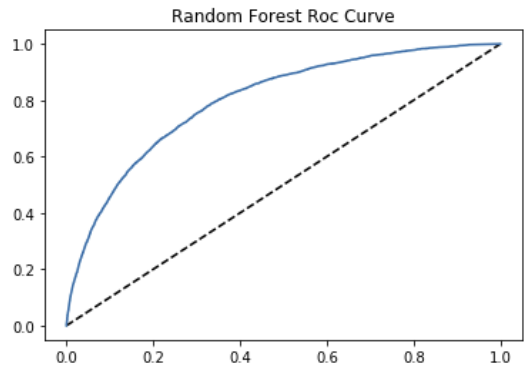
6.1.2 Gradient Boosting Method

```
AUC is 0.800396013479156
```



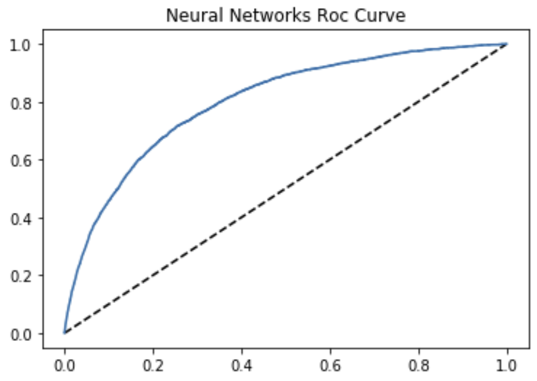
6.1.3 Random Forest

```
AUC is 0.8004125574045404
```



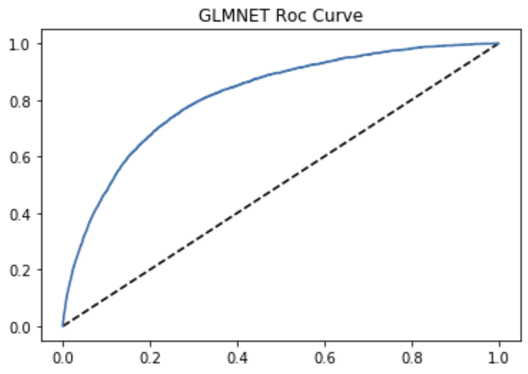
6.1.4 Neural Networks

```
AUC is 0.8007431717513441
```



6.1.5 GLMNET

```
AUC is 0.814948388533683
```

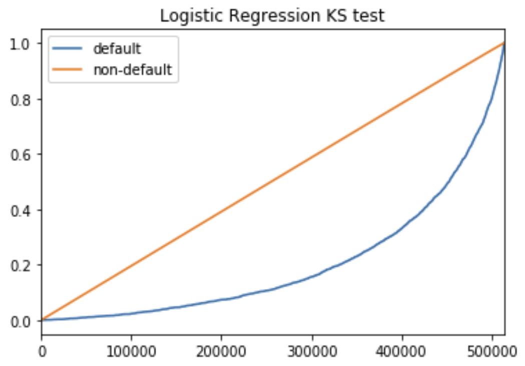


6.2 Kolmogorov-Smirnov test

KS test is very important for credit risk analysis in financial industry. It checks how good the model could differentiate default and non-default loans. The same as AUC, a larger KS stands for a better model.

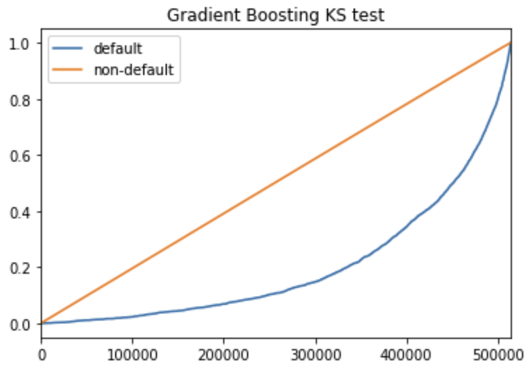
6.2.1 Logistic Regression

```
KS stat is 0.45742517900315127
```



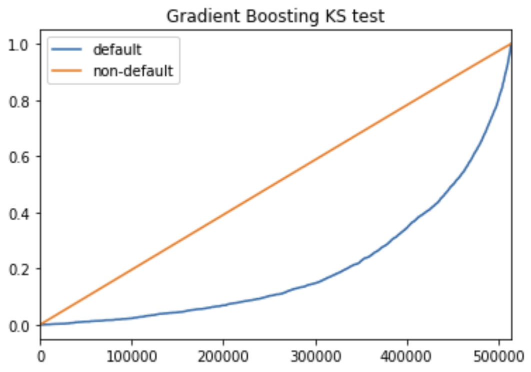
6.2.2 Gradient Boosting Method

```
KS stat is 0.4605347806880492
```



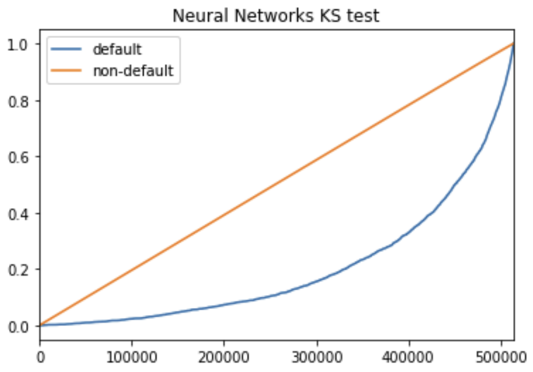
6.2.3 Random Forest

```
KS stat is 0.4605347806880492
```



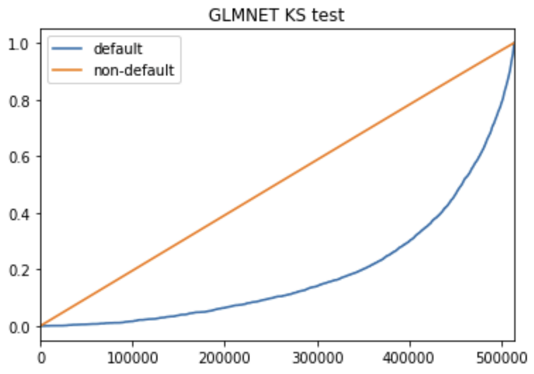
6.2.4 Neural Networks

```
KS stat is 0.4618179734225098
```



6.2.5 GLMNET

```
KS stat is 0.49098360564982396
```

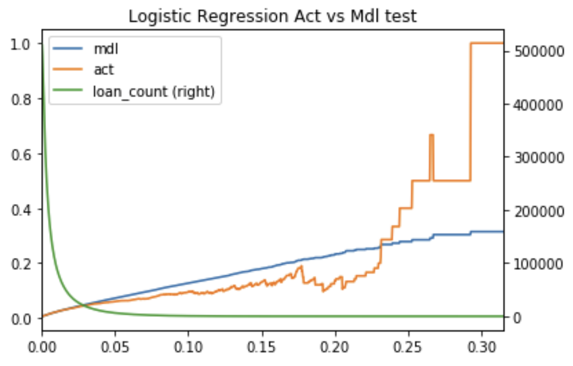


6.3 Actual vs Model Probability by Rank

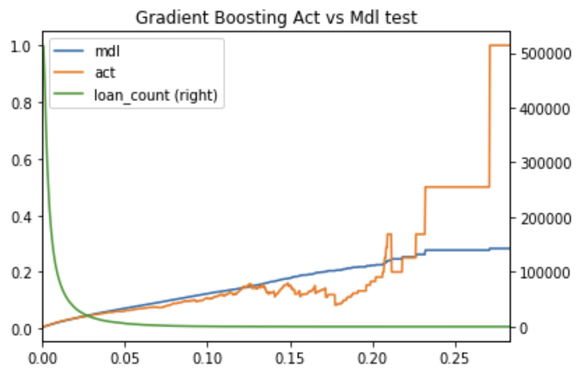
This is a test designed by ourselves, it calculates two probability curves presents E(P_act|P_mdl>p) and E(P_mdl|P_mdl>p) for 0<p<1. The closer the two curves are, the better the model is. The steps of calculation are:

(1) Calculate the range of probability that the algorithm yields. All our models produce the range between 0 and a number smaller than 50%.
(2) Bin the range into 1000 pieces with equal interval and return 1001 cut points (starts from zero) by ascending order.
(3) For p in the cut points list:
    - Find all the loans with a predicted model default probability higher than the given p
    - Calculate these loans' actual default rate and model's average default rate
    - Return (p, act_prob) and (p, mdl_prob)
(4) Plot actual probability curve with all the (p, act_prob) points and model probability curve with (p, mdl_prob) points.

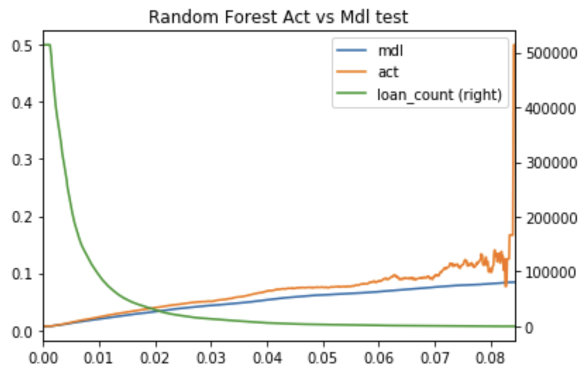
6.3.1 Logistic Regression



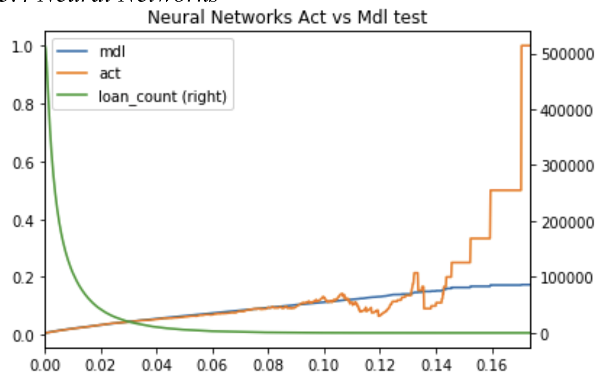
6.3.2 Gradient Boosting Method



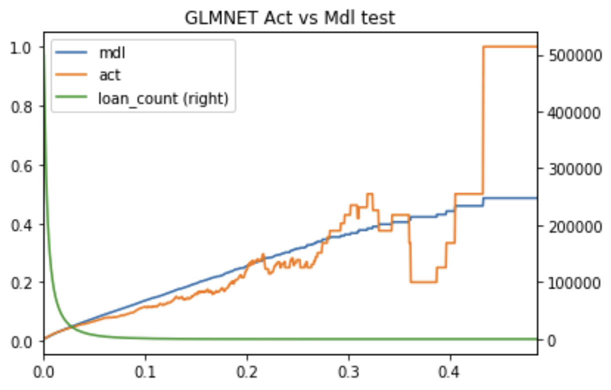
6.3.3 Random Forest



6.3.4 Neural Networks



6.3.5 GLMNET

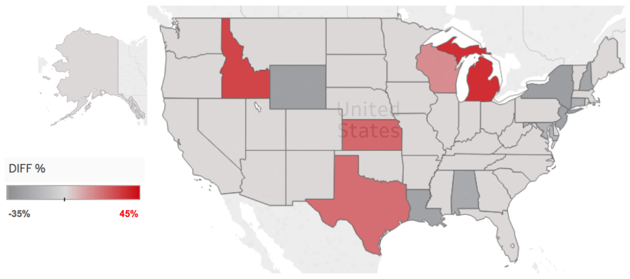


6.4 Backtest by Loan Attribution

We back-tested the model by inspecting the predicted default probability by state and other variable such as FICO, number of borrowers. The result is useful to check whether our model is sensitive enough to those loan features.

It is worth mentioning that the property state is not one of the features used in training machine learning models. Our goal is to check how large the gap between actual and model's probability of each state is. This would be helpful to provide more insight of how to improve our model.

The percentage difference of actual probability and model probability is calculated for each state and visualized in the following choropleth graph. The red color indicates severe overestimation of default rate (more conservative), while the dark grey color indicates severe underestimation.



Most states, such as California and Nevada, exhibit a small gap between the actual and the model. It is an evidence that our model has good performance in most states and can be easily generalized. For the states that are off, such as New York and Michigan, there are likely municipal factors that need to take into account. Some possible actions are to add Property State as one of the new features, and to include some municipal related factors, such as tax treatment on mortgages, into our analysis.

VII. Conclusion

This project adopted five machine learning algorithms on Freddie Mac single family 30 year fixed rate mortgages issued after Q4 2009 to classify loans which default in 5 years after origination. The best result we expect is that the algorithm could directly make classification by checking whether the probability is greater than 0.5. However, given the good macro economic scenario and strict mortgage issuance policy, the average default rate is only 0.7%, and our probability from models are all smaller than 0.5. Therefore, instead of classification error, we use three other tests (AUC, KS, actual vs model probability) to evaluate our models, and choose the best algorithm GLMNET according to the test results as our final model. During the process, we compare the training time cost, results robustness and model accuracy to understand the advantage and shortfall of these algorithms for the mortgage default analysis. Even though the model does not yield classification results default or non-default, it still provides high-quality business values. The bank or the other mortgage issuance agency could determine a threshold of probability of default and reject the borrowers whose model probability is higher than the threshold. And the ROC curve will compare the true positive rate and false positive rate, which are the percentage of default loans are rejected and the percentage of non-default loans are rejected given a threshold p. Therefore, mortgage issuer could choose a satisfying true positive and false positive pair according to their risk appetite and estimate the portfolio's default rate using the formula below:

$$r = r_u\,(1-TP)\,/\,(r_u\,(1-TP) + (1-r_u)(1-FP))$$

Given a threshold p, $TP$is true positive rate, $FP$is false positive rate, r is the portfolio's default rate and r_u is the universal default rate of the current market.

The contribution of this projects from each team member is equal. All our team members spent many hours in brainstorming, discussion of details to improve our model, presentations and report.

Guanjie was responsible for data preprocessing and model development. Zhening expanded our model library and found the best algorithm for this task. Jiada visualized our model input and output, made draft for our presentations. And this report is finished by three of us together.

According to the backtest results, we still need to improve our model by adding more municipal factors such as tax rate and more granular property information into the model since the current model severely underestimates default rate in some states like New York while overestimates that in Texas.

ACKNOWLEDGMENT

THE AUTHORS WOULD LIKE TO THANK PROFESSOR CHING-YUNG LIN AND OUR TEACHING ASSISTANTS GONGQIAN LI, VISHAL ANAND, ET AL, FOR THEIR GUIDANCE AND HELP RELATED TO THIS PROJECT.

APPENDIX

FOR MORE INFORMATION REGARDING THE SOURCE CODE AND DATA PROCESSING RELATED TO THIS PROJECT, PLEASE REFER TO: HTTPS://GITHUB.COM/SAPPHIRINE/AGENCY-LOAN-PERFORMANCE-PREDICTION.GIT

REFERENCES

[1] M. McDonald, "Financing the American Dream: Using Logistic Regression and Principal Component Analysis to Identify the Probability of Default in Mortgage Lending," University of Washington Department of Mathematics. June 2015.

[2] J. Sitignano, A. Sadhwani, K. Giesecke. "Deep Learning for Mortgage Risk," SSRN, July 2016.

[3] A. Bagherpour, "Predicting Mortgage Loan Default with Machine Learning Methods," UCR Economics, 2017.