Home Credit Default Risk

Ye Shen

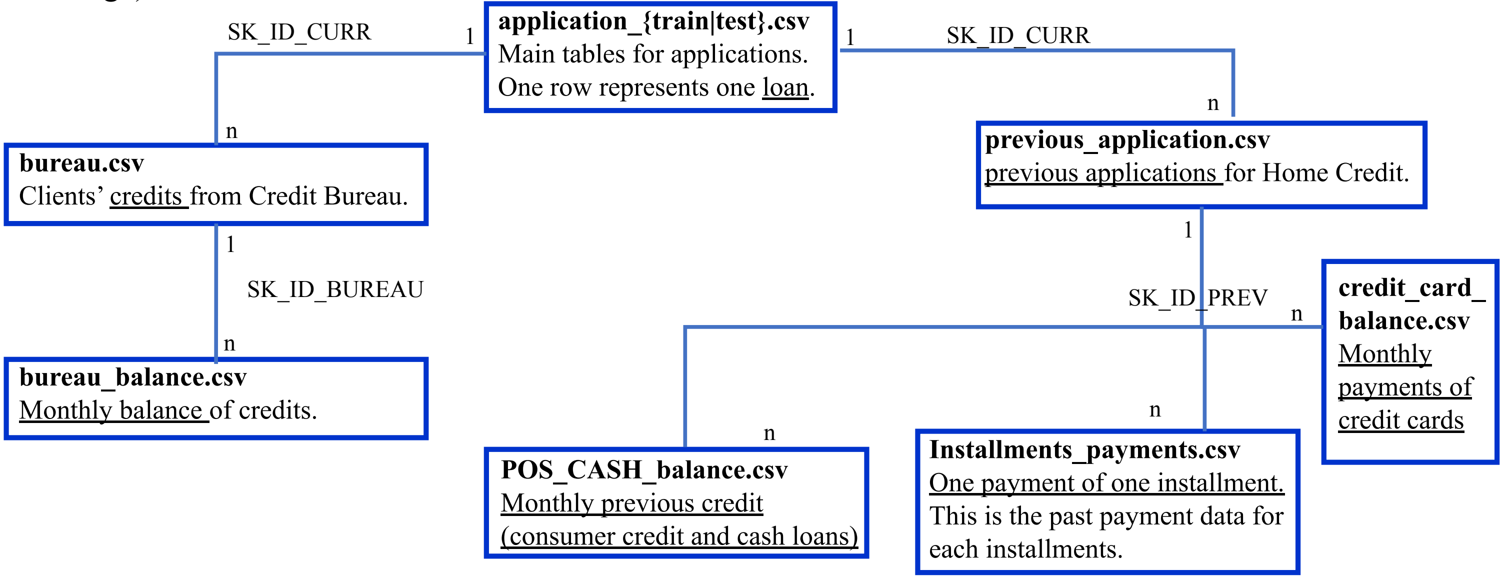
https://www.kaggle.com/c/home-credit-default-risk

Overview: Motivation

- Many people struggle to get loans due to insufficient credit histories.
- Home Credit (A loan provider) would like to broaden its market for those underserved population. But there is a risk that those population might default on their loans.
- Therefore, there is a strong motivation to develop models to predict if an applicant will default on a loan or not?
- A binary classification problem. Kaggle requires ROC AUC as the evaluation metrics.

Overview: Data

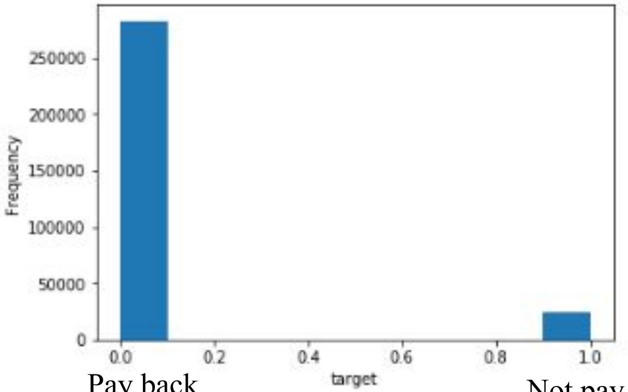
- A total of 221 columns of 2.5 G of data in 7 tables, which are related by keys. (perform aggregation and table merge)



Data Exploration: class distribution, missing values, column types

- Imbalanced class problem (under sampling, stratified cross validation)

Histogram of Labels



Pay back loans

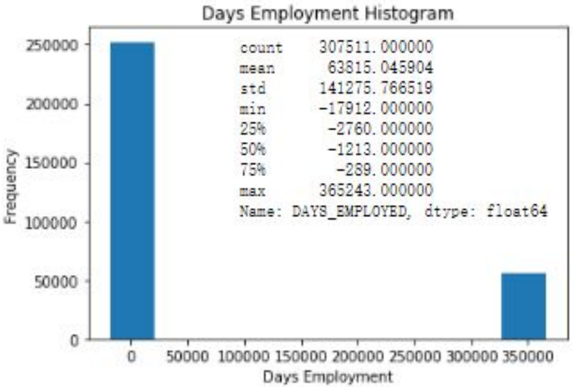
Not pay Back loans

- 67 out of the 122 columns in app_train.csv have missing values, some of the columns contain ~70% missing values.
(fill in meaningful values such as mean/median/mode or 0, or meaningful values from other features, create Boolean flag for nan)

- 16 out of the 122 columns in app_train.csv are categorical. The number of the categories vary from 2 to 58.
(label encoding and one-hot encoding, mapped to normalized frequency)

Data Exploration: anomalies, correlations

- Anomalies/Outliers: Days_employment feature has many extremely large values (350,000)



create new Boolean feature called 'flag_no_job'

- Correlations

```
Most Positive Correlations:
 NAME_EDUCATION_TYPE_Secondary / secondary special    0.049824
REG_CITY_NOT_WORK_CITY                                0.050994
DAYS_ID_PUBLISH                                       0.051457
CODE_GENDER_M                                         0.054713
DAYS_LAST_PHONE_CHANGE                                0.055218
NAME_INCOME_TYPE_Working                              0.057481
REGION_RATING_CLIENT                                  0.058899
REGION_RATING_CLIENT_W_CITY                           0.060893
DAYS_EMPLOYED                                         0.074958
TARGET                                                1.000000
Name: TARGET, dtype: float64

Most Negative Correlations:
 EXT_SOURCE_3                           -0.178919
EXT_SOURCE_2                            -0.160472
EXT_SOURCE_1                            -0.155317
DAYS_BIRTH                              -0.078239
NAME_EDUCATION_TYPE_Higher education    -0.056593
CODE_GENDER_F                           -0.054704
NAME_INCOME_TYPE_Pensioner              -0.046209
DAYS_EMPLOYED_ANOM                      -0.045987
ORGANIZATION_TYPE_XNA                   -0.045987
FLOORSMAX_AVG                           -0.044003
Name: TARGET, dtype: float64
```

Feature Engineering: within each table

- Sum: EXT_SOURCE = EXT_SOURCE_1 + EXT_SOURCE_2 + EXT_SOURCE_3
- Difference: CNT_PARENTS_MEMBERS = CNT_FAM_MEMEBR – CNT_CHILDREN
- Ratio: ANNUITY_INCOME_PERCENT = AMT_ANNUITY/AMT_INCOME_TOTAL
- Multiplication: EXT_SOURCE_1 * EXT_SOURCE_2
- Null: Fill OWN_CAR_AGE null by 100. (One does not own a car for whole life)

Feature Engineering: between different tables

- 1 to many relationship between the tables: group by key, aggregate and merge

  Numeric features: aggregate to calculate mean, max, min, sum, std
  Categorical features: aggregate to calculate count

- Generate new features from the columns within different tables

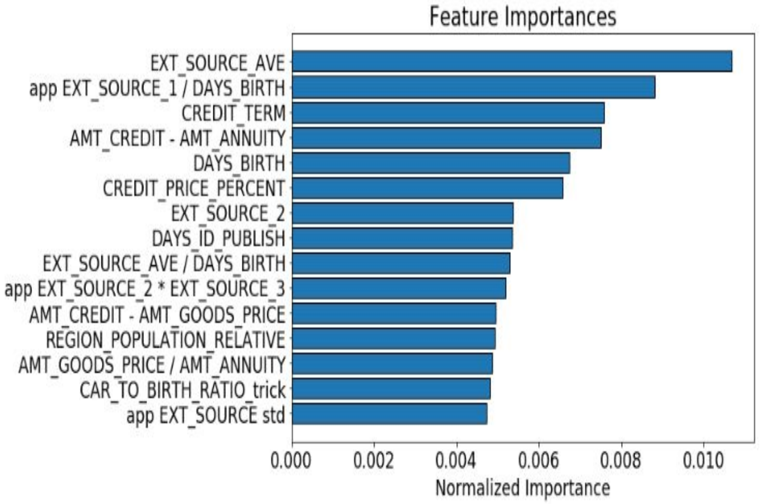
  Example: bureau_DAYS_CREDIT_max / Day_birth

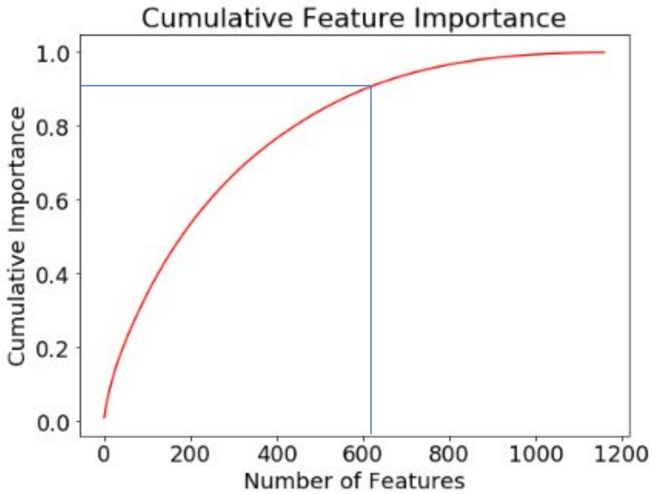
Feature Engineering: final cleaning

- Drop features: Collinear (highly correlated) features
  Features with single unique value for all the rows
  Features with high percentage of nans.

  Alignment between the training/testing dataset

Modeling: LightGBM and feature selection





603 features required for 0.9 of cumulative importance. Drop the less important features.

5 fold stratified cross validation yields average AUC = 0.794 in validation set for original data.
AUC = 0.795 in validation set for selected data.

Modeling: hyper-parameter tuning Bayesian Optimization

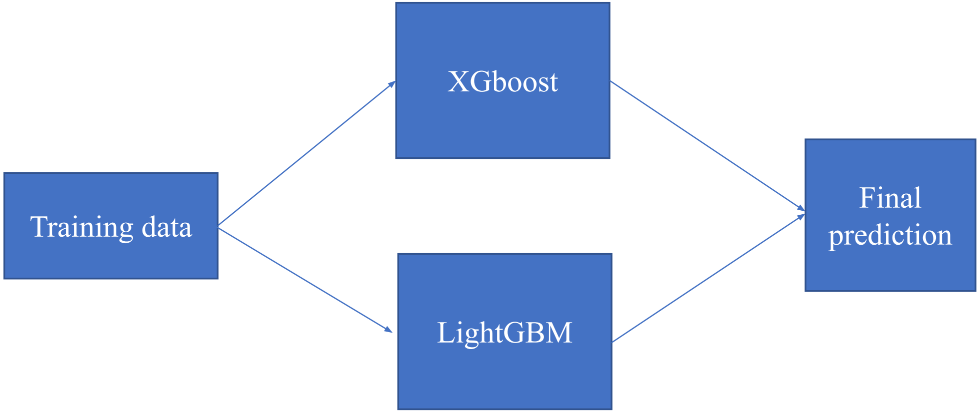
- Compared conventional hyper-parameter tuning methods, such as grid search and random search, Bayesian Optimization uses the previous evaluation results to reason about which hyper parameters perform better and uses this reasoning to choose the next values and could converge with fewer iterations.

- Hyper-parameters include: maximum depth, minimum child weight, minimum split gain, Number of leaves, L1 regulation, L2 regulation, bagging fraction, bagging frequency, feature fraction

- 5 fold stratified cross validation average AUC increases from 0.795 to 0.796 for Light GBM

Reference: [1] https://towardsdatascience.com/a-conceptual-explanation-of-bayesian-model-based-hyperparameter-optimization-for-machine-learning-b8172278050f
[2] https://github.com/fmfn/BayesianOptimization
[3] https://www.kaggle.com/sz8416/simple-bayesian-optimization-for-lightgbm

Modeling: stacking with XGboost



LightGBM	**XGboost**	**LightGBM, Xgboost stacking**
0.795	0.788	0.799

Summary

Data Exploration Analysis

- Class Imbalance
- Null values
- Categorical/Numerical
- Outliers
- Feature correlations

Feature Engineering

- Create new features within or between tables
- Drop features
- Merge tables, group/aggregation

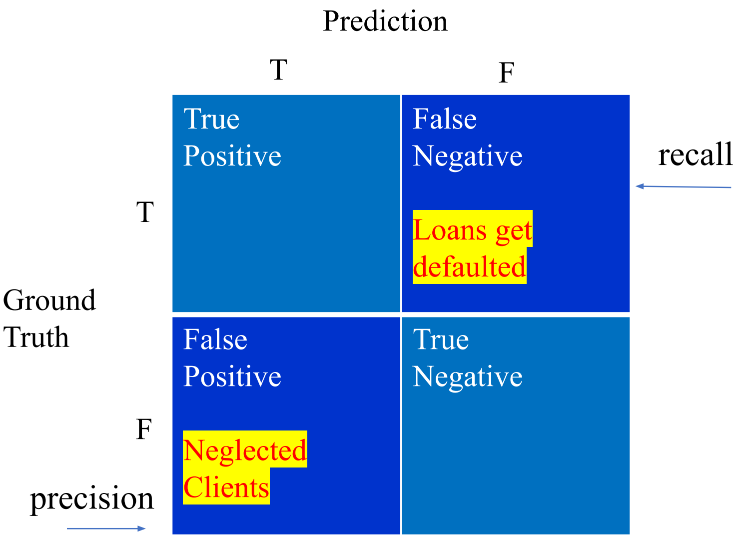
Modeling

- Feature selection
- Stratified 5 fold cross validation
- Hyper-parameter tuning through Bayesian Optimization
- Stacking LightGBM, XGBoost
- ROC AUC=0.799

Summary: future work

- Try other models (catboost)
- Continue working on feature engineering.
- Use other metrics. Because Kaggle requires ROC AUC as the metrics, Precision/Recall are more useful for unbalanced data and give more insights for the stakeholder.

Classify Loan Default True or False



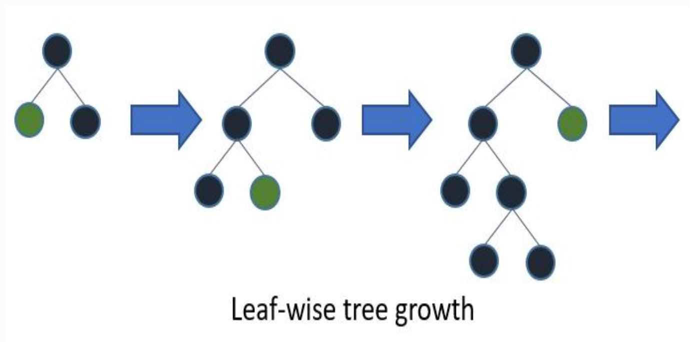
Currently working as software engineer in Hindsight-Imaging
2^{nd} year in online computer science master program in Georgia Tech
Ph.D materials science University of Wisconsin Madison
Postdoc in Harvard

In my spare time, I like go fishing with my husband.
We also have three tanks of fishes at home.

Thank you!

Modeling: LightGBM and XGBoost

- LightGBM: grows trees leaf-wise. It will choose the leaf with max delta loss to grow.



Reference:
https://lightgbm.readthedocs.io/en/latest/Features.html

- Xgboost:
  Iterate for nround times:
  grow the tree to the maximum depth
  find the best splitting point
  assign weight to two new leaves
  prune the tree to delete nodes with negative gain