CS598 - Project 3

Xiaoming Ji

Computer System

Hardware

- Dell Precision Tower 5810
- CPU: Intel Xeon E5-1607 @ 3.10GHz
- Memory: 32GB
- GPU: Nvidia GeForce GTX 1080 (2 cards)

Software

- OS: Windows 10 Professional 64bit
- R: 3.5.1
- R Packages:
  - catboost_0.11.1
  - xgboost_0.71.2
  - randomForest_4.6-14
  - glmnet_2.0-16
  - kernlab_0.9-26

Preprocessing and Feature Engineering

Several approaches are taken to pre-process the data.

- Response label: merge *Charged Off* to *Default* and convert the label value to 0 or 1.
- Build new predictors to help training/prediction:
  - `earliest_cr_line_mon`: derived from *earliest_cr_line* that indicates how many months has elapsed till *2019-1-1* when the borrower's earliest reported credit line was opened.
  - `fico_score`: consolidate *fico_range_high* and *fico_range_low* using formula: *(fico_range_high + fico_range_low) / 2*.
- Level grouping:
  - `zip_code`: it has more than 900 levels, I group these values to 10 new levels to reduce memory usage and improve performance.
- Remove predictors: remove less useful and redundant predictors.
  - `emp_title` (too many levels), `title` (redundant with *purpose*), `grade` (redundant with *sub_grade*) ,`earliest_cr_line`, `fico_range_high`, `fico_range_low`.

Models

For testing purpose, I build 7 models,

- Dumb model: this is the simplest model that predict 0.2 for every sample.
- Logistic Regression
- Boosting (XGBoost, CatBoost)
- RandomForest
- Lasso

- liner SVM

Suprisingly, Dumb model can achieve `0.504` logloss score. kernlab *ksvm()* fails to build the model (hang forever). lasso and random forest don't give me significant improvement than logistic regression and they take much longer time to build. Thus, I will pick `dumb`, `Logistic Regression` and `Boosting` as my final models.

Note: My testing shows CatBoost performs at least 10x faster than XGBoost (with GPU, CatBoost can do even better). In case catboost library is not installed, xgboost will be used.

Evaluation

I tested all 3 test datasets against these models with the parameters,

- Dumb: None.
- Logistic Regression: liner combination of all available predictors.
- Boosting: One-hot encoding on train/test data then train with the following parameters,
  - CatBoost: loss_function = "Logloss", learning_rate = 0.09, iterations = 1200
  - XGBoost: objective = "binary:logistic", eval_metric = "logloss", eta = 0.09, nrounds = 1200

The LogLoss scores are,

	Dumb	LogisticRegression	Boosting
Test1	0.5038353	0.4539925	0.4445856
Test2	0.5038353	0.4548663	0.4462146
Test3	0.5038353	0.4539786	0.4448837
Average	0.5038353	0.4542791	0.4452280

Computation time: 1237 seconds