Loan Prediction Challenge

DN Created by Dan Nguyen
Last updated: less than a minute ago

Business Objective

- A lending company that provides small loans to individual borrowers for 3 or 5 years wants to predict whether a loan it makes will be "good" or "bad" down the road.
- The objective here is to build a predictive model that can predict whether an individual loan is "good" or "bad," using the provided sample dataset.

Data

- **Unit of analysis**: 10,000 unique customers with no duplicates, each has a single loan.
  - 476 observations have missing data across all fields other than `loan_amount` and `funded_amount`. It's appropriate to exclude them from the analysis.
- **Target variable**: `loan_status`, using the logic of a good loan having the status of either "current," "fully paid", or "in grace period" and bad loan having the status of "default," "charged off," "late (16-30 days)" or "late (31-120 days". In reality, this logic should be driven by business context and informed by domain knowledge. The outcome distribution is *highly imbalanced* with ~4% of the loans are bad vs. 96% are good.
- **Predictors**: assuming the model is used to predict loan status *before* making the loans, several variables should be excluded because they are either endogenous to the loan-making decision or simply are not available at the decision-making time (`funded_amount`, `outstanding_principal`, `total_payment`, `total_received_principal`, and `total_received_interest`). Also, while zip code is usually forbidden and `address_state` might be admissible for regulatory compliance purposes, I exclude it for now. That leaves us with 15 predictors as discussed below.

Model Screening

- How we do feature engineering and feature generation is part of the modeling process. I explore four sets of features (aka, recipes) and four models, resulting in 16 combinations.
  - For ease of comparison, each subsequent recipe includes all features in the previous recipe plus additional features. Specific steps of feature engineering and generation can be found in the code.
  - All four models use default hyper-parameter settings (no tuning) with over-sampling the minority class.

Feature sets	*Base recipe (loan-related features)* • `loan_amount` • `purpose` • `term` • `interest_rate` • `installment`.	*Mid recipe (base recipe + credit worthiness-related features)* • `delinquency_2years` • `months_since_delinquency` • `months_since_first_credit` • `revolving_balance` • `open_accounts` • `total_accounts`	*Full recipe (mid recipe + personal finance features)* • `annual_income` • `dti` (debt to income ratio) • `employment_length` • `home_ownership`	*Generative recipe (full recipe + additional generated features)* • `loan_income_ratio` • `installment_balance_ratio` • `open_account_ratio` • `total_earnings`
Classification models	Logistic regression	Random forest	XGBoost	LightGBM

- The goal of model screening is to select the most predictive feature set-model combo. The results suggest the generative recipe-logistic regression combo performs the best. (Code in R since I have done this many times before and could re-use some of the code.)
  - Generally, metrics should be business-driven, but for this highly imbalanced dataset, the key metric is the Jaccard index (specificity + sensitivity - 1) to balance the trade-off between specificity and sensitivity (i.e., out of all the truly good/bad loans, how many are correctly predicted as good/bad).
  - Logistic regression also has the important benefit of providing convenient model explainability.

Loan Classification - Model Screening

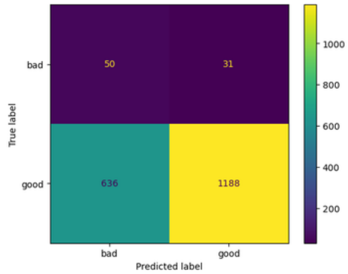
5-fold CV on training set (80%)

RECIPE/MODEL COMBO	J-INDEX	SPECIFICITY	SENSITIVITY	PRECISION	RECALL	F1 SCORE	LOG LOSS
generative_recipe_logistic_reg	0.23	0.62	0.61	0.06	0.61	0.11	0.65
full_recipe_logistic_reg	0.21	0.62	0.58	0.06	0.58	0.11	0.64
mid_recipe_logistic_reg	0.18	0.57	0.61	0.06	0.61	0.10	0.68
base_recipe_logistic_reg	0.16	0.60	0.56	0.06	0.56	0.10	0.67
base_recipe_rand_forest	0.15	0.68	0.47	0.06	0.47	0.10	0.61
base_recipe_boost_tree_4	0.11	0.74	0.37	0.06	0.37	0.10	0.56
base_recipe_boost_tree_3	0.08	0.71	0.37	0.05	0.37	0.09	0.63
mid_recipe_rand_forest	0.08	0.82	0.26	0.06	0.26	0.09	0.49
full_recipe_rand_forest	0.08	0.82	0.26	0.06	0.26	0.09	0.49
mid_recipe_boost_tree_3	0.06	0.85	0.21	0.06	0.21	0.09	0.44
generative_recipe_rand_forest	0.06	0.69	0.37	0.06	0.37	0.09	0.74
full_recipe_boost_tree_3	0.05	0.85	0.20	0.05	0.20	0.08	0.47
mid_recipe_boost_tree_4	0.04	0.87	0.17	0.05	0.17	0.08	0.41
generative_recipe_boost_tree_4	0.03	0.73	0.30	0.05	0.30	0.08	2.44
full_recipe_boost_tree_4	0.02	0.85	0.17	0.05	0.17	0.07	0.42
generative_recipe_boost_tree_3	0.02	0.72	0.30	0.05	0.30	0.07	1.60

Final Model

- A last model coded in Python use the selected feature-model combo and is trained on the training set (80%) and tested on the test set (20%).
  - The model is able to identify 62% (50 out of 81) of the bad loans (at the expense of low precision with only 50 out of 686 predicted bad loan are truly bad). The model's ability to recall/identify a clear majority of the bad loans is perhaps more important from the business perspective since making bad loans can result in direct capital loss.

- Out of all the good loans, the model correctly identifies 64% of them. The 97% precision rate is also very high, meaning if the model predicts a good loan, it's almost certain that the loan is truly good.



	precision	recall	f1-score	support
bad	0.07	0.62	0.13	81
good	0.97	0.64	0.77	1824
accuracy			0.63	1905
macro avg	0.52	0.63	0.45	1905
weighted avg	0.94	0.63	0.74	1905

- The final model is then trained on all the data and pickled as a saved model for use as part of productionization (although if we use a cloud-based machine learning platform, this can be partly automated).

Next Steps

- In projects like this, it's important to collaborate with business stakeholders to quantify the cost of making a bad loan vs. the cost of missing out on a good loan. That will enable us to **derive a custom cost function** and use it to select an optimal classification threshold rather than using the default 0.5.
- Conduct **model explainability** using feature importance (regression model coefficients could suffice) and subpopulation analysis for bias and fairness assessment.
- Conduct **causal analysis**, if experiments are impossible or prohibitively costly, to identify the causal effects of key predictors such as credit line and loan term on the probability of default/charged-off. These variables are actionable and thus could have important business impacts.