Using Gradient Boosting Tree algorithms on Porto-Seguro's safe drive competition

Binh Nguyen

Université Paris–Saclay & UEVE

January 16, 2018

Overview

Introduction

- Hosted on Kaggle, by Porto Seguro - a car insurance company.
- Objective: build a model that predicts the probability that a driver will initiate an auto insurance claim in the next year.
- Done on Python & Google Cloud Platform

Problem settings

- Supervised learning
- Binary classification.
- Performance metrics: Normalized Gini Index.

Data Analysis

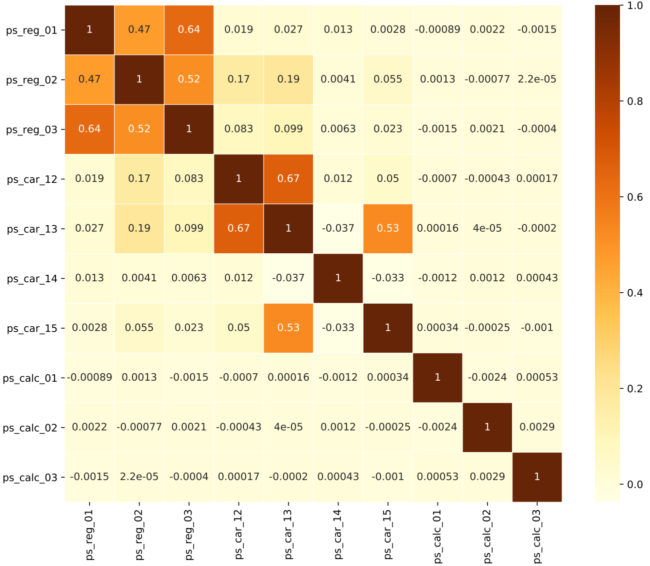
General info

- train.csv & test.csv
- For train: n = 595212; p = 59. For test: n = 892816

Interesting findings

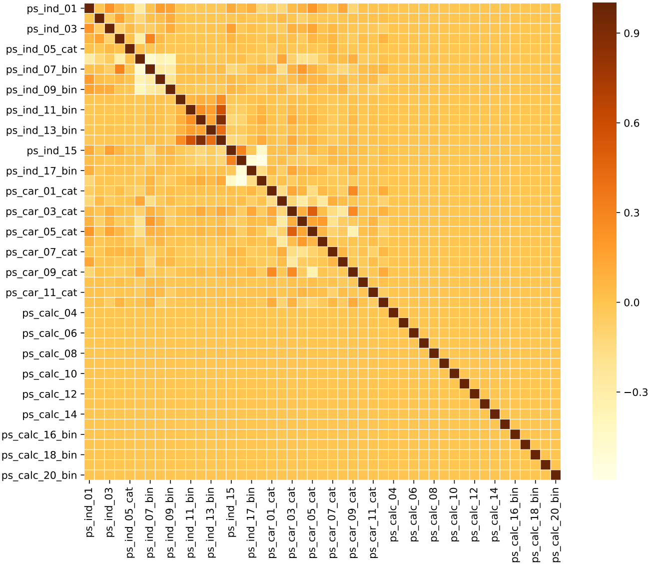
- Number of n in test is much larger than train.
- Features are encoded under generic names (for example ps_car_01, ps_reg_02) that give no specific indication on the meaning.
- There are 3 types: continuous features, categorical features and binary features.
- Values of -1 indicate that the feature was missing from the observation, or NaN value as normally seen in many datasets.

Correlation matrix of continuous features (`float64`)

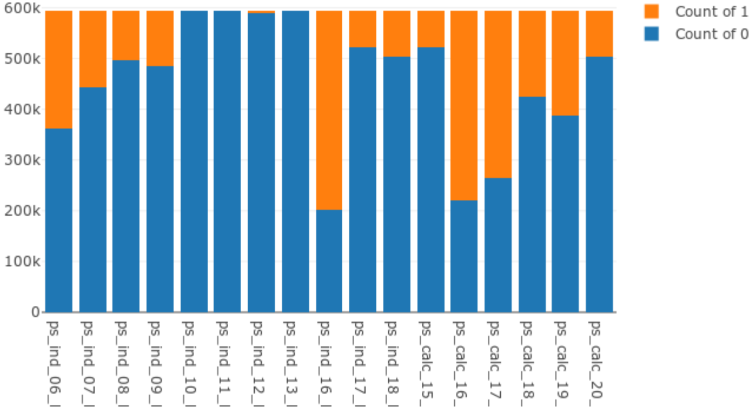


	ps_reg_01	ps_reg_02	ps_reg_03	ps_car_12	ps_car_13	ps_car_14	ps_car_15	ps_calc_01	ps_calc_02	ps_calc_03
ps_reg_01	1	0.47	0.64	0.019	0.027	0.013	0.0028	-0.00089	0.0022	-0.0015
ps_reg_02	0.47	1	0.52	0.17	0.19	0.0041	0.055	0.0013	-0.00077	2.2e-05
ps_reg_03	0.64	0.52	1	0.083	0.099	0.0063	0.023	-0.0015	0.0021	-0.0004
ps_car_12	0.019	0.17	0.083	1	0.67	0.012	0.05	-0.0007	-0.00043	0.00017
ps_car_13	0.027	0.19	0.099	0.67	1	-0.037	0.53	0.00016	4e-05	-0.0002
ps_car_14	0.013	0.0041	0.0063	0.012	-0.037	1	-0.033	-0.0012	0.0012	0.00043
ps_car_15	0.0028	0.055	0.023	0.05	0.53	-0.033	1	0.00034	-0.00025	-0.001
ps_calc_01	-0.00089	0.0013	-0.0015	-0.0007	0.00016	-0.0012	0.00034	1	-0.0024	0.00053
ps_calc_02	0.0022	-0.00077	0.0021	-0.00043	4e-05	0.0012	-0.00025	-0.0024	1	0.0029
ps_calc_03	-0.0015	2.2e-05	-0.0004	0.00017	-0.0002	0.00043	-0.001	0.00053	0.0029	1

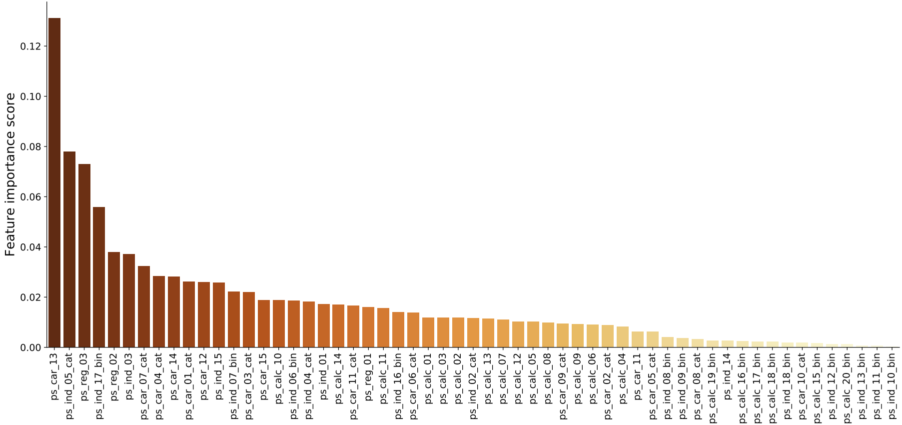
Correlation matrix of continuous features (`int32`)



Count of number of 0 and 1 in binary features



Variable important ranking using Random Forest Classifier



Feature Engineering

Eliminating unimportant features

Top 35 variables which have highest feature important score are kept.

Dealing with missing values

One could possibly either set `NaN` to some extreme value (e.g. 10^6) or use the value that represented the property of the particular variable ($\max(X_i)$, $\text{median}(X_i)$). In this case, we choose to keep the default value -1 to represent `NaN`.

Likelihood encoding

The central idea behind likelihood (or target) encoding technique is to estimate likelihood target variable given a particular value of a categorical variable:

$$Pr(y = 1 \mid x = \texttt{catValue}) \quad = \frac{\sum \mathbb{1}_{y=1|x=\texttt{catValue}}(y)}{n_{\text{observation}}}$$

Algorithm 1 Target encoding categorical variables

Input: `categoricalArray`, `labelArray`, k, l

data $\leftarrow$ **join** (`categoricalArray`, `labelArray`)
`targetEncodedArray` $\leftarrow$ Empty Array
`categoricalValue` $\leftarrow$ unique values of `categoricalArray`
split `data` into k parts
for each `fold[i]` in k folds **do**
    `valueEncoded` $\leftarrow 0$
    `fold[-i]` $\leftarrow$ **join** remaining $k-1$ folds except i
    split `fold[-i]` into l parts
    for each `fold[j]` in new l parts **do**
        for each `value` in `categoricalValue` **do**
            $\texttt{valueEncoded} \leftarrow \texttt{valueEncoded} + \dfrac{Pr(label \mid value)}{l}$
        end for
    end for
    `valueEncodedFold[i]` $\leftarrow$ **join** all `valueEncoded`
end for
`targetEncodedArray` $\leftarrow$ **join** k parts of `valueEncodedFold[i]`
return `targetEncodedArray`

Implementation

Model Selection

- Gradient Boosting Tree: XGBoost[a] & LightGBM[b]
- Extremely fast, can also be used with GPU

[a] https://github.com/dmlc/xgboost
[b] https://github.com/Microsoft/LightGBM

Ensembling technique

- 5-fold cross-validation
- Using `sklearn.model_selection.GridSearchCV` to find best `learning_rate`, `max_depth` parameters
- Prediction data from these 2 models was fitted by a Logistic Classifier acted as the ensembler to produce final prediction
- Google Cloud cluster with 16 CPU cores and 12GB RAM takes over 2 hours

Algorithm 2 Ensemble classifier (XGBoost + LightGBM + Logistic Regression)

Input: trainData, testData, k

`objective` ← Normalized Gini Index
split trainData, testData to k folds
for each `fold[i]` in k folds **do**
    run Algorithm 1
    fit XGBoost on `fold[-i]` using fold[i] to maximize `objective`
    fit lightGBM on `fold[-i]` using fold[i] to maximize `objective`
    `predictXGB` ← XGBoost(testData)
    `predictLGBM` ← lightGBM(testData)
end for
$\texttt{predictXGB} \leftarrow \dfrac{\texttt{predictXGB}}{k}$
$\texttt{predictLGBM} \leftarrow \dfrac{\texttt{predictLGBM}}{k}$
`ensemblePrediction` ← logisticRegression(`predictXGB`, `predictLGBM`)
return `ensemblePrediction`

Results

Model	**Public Leaderboard**	**Final Leaderboard**
XGBoost only	0.27477	0.27756
XGBoost + target encode	0.28204	0.28598
Ensemble + target encode	**0.28365**	0.28763
Best result	0.28332	**0.28849**

1800	▼ 960	**Regis**	0.28764	26	2mo
1801	▼ 257	**AntiLippasaar**	0.28764	23	2mo
1802	▲ 544	**scut_430**	0.28763	3	3mo
1803	▼ 2	**idc man**	0.28763	68	3mo
1804	▲ 606	**LightR**	0.28763	1	3mo
1805	▼ 29	**Rafael Ladeira**	0.28762	9	3mo

✔ Your submission description has been saved.

68 submissions for idc man

Sort by Private Score

All Successful Selected

Submission and Description	Private Score	Public Score	Use for Final Score
submission-lgb-only-impact-encoding-joinbin-dropcalconly-10fold-17... 3 months ago by BTNG 10 fold cv	0.28849	0.28332	☐
submission-171022-004132.csv 3 months ago by BTNG Upload Submission File submission-171022-004132 stack of 3 including gradient boosting tree with max_depth=5 all other default. No feature elimination.	0.28838	0.28225	☐
submission-lgb-only-impact-encoding-5fold-notps13square-171025-1... 3 months ago by BTNG submission-lgb-only-impact-encoding-5fold-notps13square-171025-193323.csv not create ps13square, not drop ps 13	0.28820	0.28321	☐
submission-lgb-only-impact-encoding-joinbin-dropcalconly-171024-1... 3 months ago by BTNG submission-lgb-only-impact-encoding-joinbin-dropcalconly-171024-171759.csv - create new impact join bin 06-09, maxdepth 6 on lgb learning rate 0.01	0.28806	0.28287	☐
stacking1-171019-201739.csv 3 months ago by BTNG stacking1-171019-201739.	0.28806	0.28214	☐

The End.