Porto Seguro's Safe Driver Prediction Project Status Report

Hanlin He, Mingze Xu, Su Yang, Tao Wang

Department of Computer Science, The Erik Jonsson School of Engineering and Computer Science

The University of Texas at Dallas

Email: {hxh160630, mxx160530, sxy161730, txw162630}@utdallas.edu

***Abstract*—This is the project status report of Porto Seguro's Safe Driver Prediction. Some of the content would be part of the final report.**

I. Introduction

Machine learning is emerging in the insurance industry and is being applied across multiple areas including the interpretation of data, business operations and driver safety. One key application is claim prediction. Inaccuracies in car insurance company's claim predictions raise the cost of insurance for good drivers and reduce the price for bad ones. A more accurate prediction will allow insurers to further tailor their prices, and hopefully make auto insurance coverage more accessible to more drivers.

In this report, we based on Kaggle's Featured Prediction Competition *Porto Seguro's Safe Driver Prediction* [1], conducted several experiments, compared different approaches' effectiveness to tackle the claim prediction problem, including *logistic regression*, *tensorflow estimator* and *random forest*.

The organization of the following report is as follows. In section II, we will formally define the problem to solve and discuss the theoretical principle of the algorithm we used. Then we will analyze the data feature and our method of feature engineering in section III. After that, the experimental results are shown and analyzed in section IV. Finally, we will discuss related works and conclude the report.

II. Problem Definition and Algorithm

A. Task Definition

A machine learning problem is defined as to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E [6]. The claim prediction problem can be defined as follow:

- E previous year's policy holders' information and whether or not a claim was filed for that policy holder.
- T predicting the possibility that an auto insurance policy holder will file an insurance claim next year.
- P the accuracies and Gini Coefficient was used to measure the effectiveness of models.

B. Algorithm Definition

We used logistic regression, neural network in our model building. In this section, we will discuss the theoretical foundation of these algorithms.

1) Logistic Regression: Logistic Regression is an approach to learning functions of the form $f : X \rightarrow Y$ [5] or in our case $P(Y|X)$ where Y is discrete-valued, and $X = \langle X_1 ... X_n \rangle$ is any vector containing discrete and continuous variables. The parametric model assumed by Logistic Regression in the case where Y is boolean is:

$$P(Y=1|X) = \frac{1}{1+\exp(w_0+\sum_{i=1}^{n} w_i X_i)}$$
$$P(Y=0|X) = \frac{\exp(w_0+\sum_{i=1}^{n} w_i X_i)}{1+\exp(w_0+\sum_{i=1}^{n} w_i X_i)}$$

One reasonable approach to training Logistic Regression is to choose parameter values that maximize the conditional data likelihood. We also used regularization to reduce the overfitting problem. The penalized log likelihood function is as followed:

$$W \leftarrow \arg\max_{W} \sum_{l} \ln P(Y^l|X^l, W) - \frac{\lambda}{2}||W||^2$$

where the last term is a penalty proportional to the squared magnitude of W.

In general, the algorithm used gradient ascent to repeatedly update the weights in the direction of the gradient, on each iteration changing every weight w_i, beginning with initial weights of zero, according to:

$$w_i \leftarrow w_i + \eta \sum_{l} X_i^l (Y^l - \hat{P}(Y^l=1|X^l, W)) - \eta\lambda w_i$$

where η is a small constant which determines the step size. The actual implementation of scikit learn library includes multiple solvers, such as Stochastic Average Gradient (SAG) descent, SAGA and Broyden-Fletcher-Goldfarb-Shanno (LBFGS).

III. Feature Analysis and Engineering

The data comes in the traditional Kaggle form of one training and test file each: `train.csv` and `test.csv`. Each row corresponds to a specific policy holder and the columns describe their features. The target variable is named `target` here and it indicates whether this policy holder made an insurance claim in the past.

A. Data Overview

In the train and test data, features that belong to similar groupings are tagged as such in the feature names (e.g., `ind`, `reg`, `car`, `calc`). In addition, feature names include the postfix `bin` to indicate binary features and `cat` to indicate categorical features. Features without these designations are either continuous or ordinal. Feature count in each category and type are shown in table I.

TABLE I

Feature Counts in Each Category and Type

Type	Binary	Categorical	Numeric	Total
`ind`	11	3	4	18
`reg`	0	0	3	3
`car`	0	11	5	16
`calc`	6	0	14	20

Although feature's categories are provided, the meaning of each feature remains unknown. Some participants have guessed the meaning of several features, for example the 'binary' variables `ps_ind_06-10` are one-hot encoded, and `ps_car_13` might be car's mileage. Our experiment did not take these information into consideration.

B. Missing Value Mechanism and Data Imputation

Values of -1 indicate that the feature was missing from the observation.

IV. Experimental Evaluation

A. Methodology

We used per-class accuracy, AUC and Normalized Gini Coefficient to evaluate out model.

1) Per-Class Accuracy: Accuracy simply measures how often the classifier makes the correct prediction. Its the ratio between the number of correct predictions

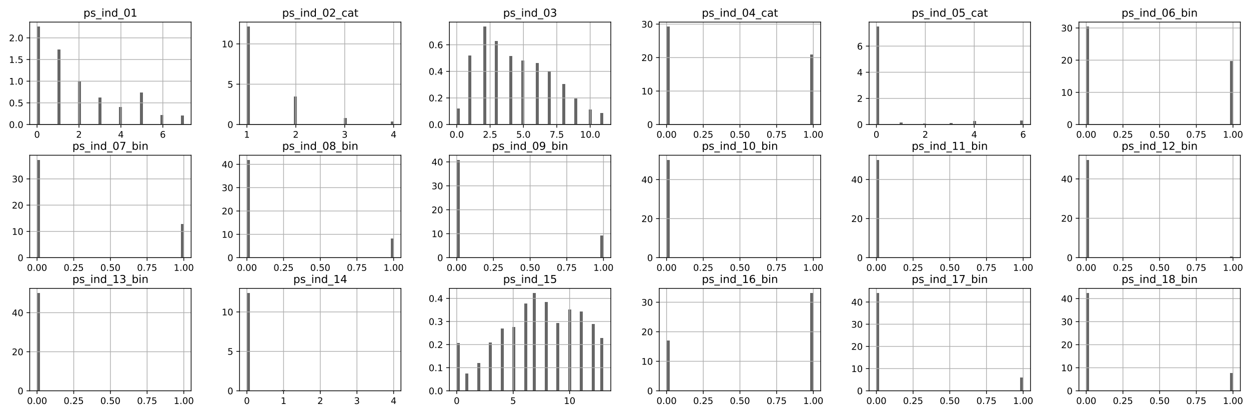

Fig. 1. Histogram for the `ind` Attributes.

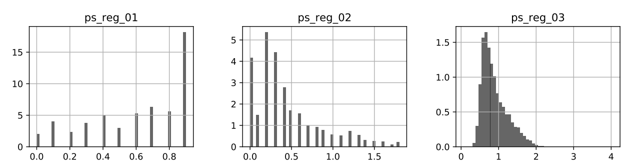

Fig. 2. Histogram for the `reg` Attributes.

and the total number of predictions (the number of data points in the test set) [7]:

$$accuracy = \frac{\#\ \text{correct predictions}}{\#\ \text{total data points}}$$

Per-class accuracy is the average of the accuracy for each class. By using per-class accuracy, we can have a better understanding of the model if the target class was dominated by one label.

2) Normalized Gini Coefficient and AUC: The Normalized Gini coefficient, (named for the similar Gini coefficient/index used in Economics, which originally developed by Italian statistician and sociologist Corrado Gini [4]), measures the inequality among values of a frequency distribution (for example, levels of income) [2]. It is most commonly defined as twice the area between the ROC curve and the diagonal (with this area being taken as negative in the rare event that the curve lies below the diagonal).

AUC stands for area under the curve.

The normalized Gini coefficient and AUC are closely related. When using normalized units, the AUC is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (assuming 'positive' ranks higher than 'negative') [3]. Elementary geometry shows that

$$Gini = 2 \times \text{AUC} - 1. \tag{1}$$

The competition used normalized Gini coefficient to measure participants' submission performance. In our experiment, we worked in terms of AUC, but the results apply equally to the Gini coefficient.

V. Coding Language

The main coding language is Python with following machine learning library:

- pandas.
- scikit-learn.
- tensorflow.

References

[1] Porto Seguros Safe Driver Prediction, 2017. [Online; accessed November 7, 2017].

[2] Wikipedia: Gini coefficient. https://en.wikipedia.org/wiki/Gini_coefficient, 2017. [Online; accessed November 7, 2017].

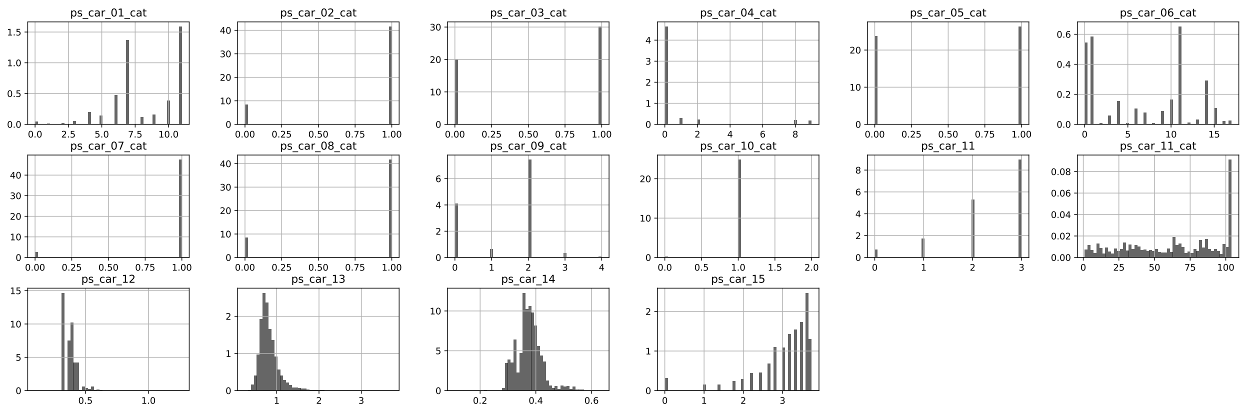

Fig. 3. Histogram for the `car` Attributes.

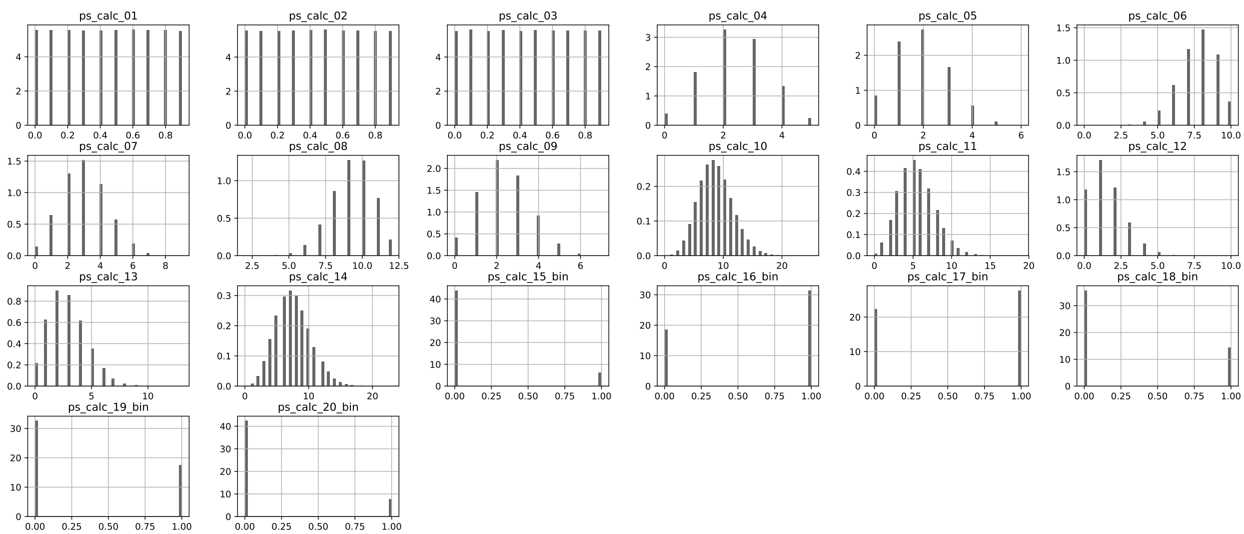

Fig. 4. Histogram for the `calc` Attributes.

[3] Tom Fawcett. An introduction to roc analysis. *Pattern Recogn. Lett.*, 27(8):861–874, June 2006.

[4] Corrado Gini. *Variabilitá e Mutuabilitá.* 1912. Contributo allo Studio delle Distribuzioni e delle Relazioni Statistiche. C. Cuppini, Bologna.

[5] Thomas M. Mitchell. *Generative and Discriminative Classifiers: Naive Bayes and Logistic Regression*. draft of February, 2016.

[6] Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

[7] Alice Zheng. *Evaluating Machine Learning Models*. OReilly Media, Inc., 1 edition, 2015.