CS 229 Final Report: Predicting Insurance Claims in Brazil

Matthew Millican
millimat@stanford.edu

Laura Zhang
lzhang96@stanford.edu

Dixee Kimball
dkimball@stanford.edu

December 16, 2017

1 Introduction

Improving the accuracy of insurance claims benefits both customers and insurance companies. Incorrect predictions effectively raise insurance costs for safe drivers and lower costs for risky drivers, and can be costly to insurance companies. Better predictions increase car-ownership accessibility for safer drivers and allow car insurance companies to charge fair prices to all customers. Better predictions also lead to improved profits for insurance companies.

The problem is as follows: *given a series of unlabeled features collected by an insurance company about a customer, can we predict whether the customer will file an insurance claim during a period of interest?*

The input to our algorithm is set of 595,213 labeled records, one per customer. Each record consists of $n = 57$ features with unknown meaning and a label indicated whether the customer filed an insurance claim. We then use least squares ridge regression, least squares lasso regression, logistic regression, Naive Bayes, random forests, gradient boosting, one-layer perceptron, and two layer-perceptron to predict whether a customer filed an insurance claim.

2 Related Work

Insurance claim prediction has rarely been studied in past 229 projects. The project that is most similar to this one was conducted by Diveesh Singh in 2016 using images of drivers while operating their vehicle to predict driver behavior (Singh 2016).

However, insurance is a topic that has been researched more broadly in the field. Smith et al. (2000) use several techniques such as decision trees and neural networks to study which customer characteristics affect retention in their policy. Yeo et al. (2001) employ clustering techniques to group customers according to their risk classification and then regression methods to model the expected claim costs within a risk group. Outside of automobile insurance, researchers have also studied health insurance (Browne 1992), non-life insurance (Salcedo-Sanz et al. 2005), and many other situations where risk plays a substantial factor in pricing (Lee & Urrutia 1996, Cummins & Phillips 1999).

3 Dataset and Features

Data was provided by Porto Seguro (Kaggle 2017). The training dataset contains labeled records $(x^{(i)}, y^{(i)})$ for 595,213 customers. Each record $x^{(i)}$ consists of $n = 57$ features with unknown meaning, and a label $y^{(i)} \in \{0, 1\}$ indicating whether customer i filed a claim. 21,694 examples have label 1, and the remaining 573,518 have label 0.

The test set contains 892,816 records without labels. To prevent fitting to the test data, Porto Seguro makes test labels y_{test} available only implicitly, by displaying $\mathcal{G}_n(\hat{y}_{\text{test}}, y_{\text{test}})$ for a submitted prediction $\hat{y}_{\text{test}}$.

Of the 57 features available for each customer, 26 represent continuous or ordinal values, and 31 represent categorical values.

This project faces the unique challenge of fitting a dataset with unlabeled features. To protect the identities of car insurance policyholders in the data provided by Porto Seguro, one of Brazil's largest auto and homeowner insurance companies, all features are unlabeled. However, some information on the type of data is provided, and features are grouped by their type. Because further information on the features was not provided, we cannot rely on intuition to feature-select.

3.1 PCA - Dimensionality Reduction

We attempted to conduct dimensionality reduction on our dataset by using PCA. However, during training, we decided against doing so after retrieving the variance ratios outputted by PCA. The result showed that 90% of the variance in the training set can be explained by a single principal component. We found this result suspicious, and when we attempted to train models on this single principal component, they performed measurably worse under our evaluation metric than models trained on the full dataset (results not shown). This suggests the dataset is incompatible with PCA, which may be due to the large number of categorical and binary features it contains.

3.2 Imputation of Missing Values

Some values in the input datasets were missing (marked with -1). We preprocessed the dataset to replace these missing values with the mean of the feature over all examples in the dataset (for continuous features) or with the mode of the feature value over the dataset (for categorical and binary features). This imputation afforded minor improvements to our evaluation metric relative to models trained on data without imputation.

4 Evaluation Metric

Since claim prediction is a binary classification problem, a conventional method of model evaluation might involve computation of accuracy, precision, recall, or F1 score. However, Porto Seguro's own model evaluation system involves computation of the *normalized Gini coefficient*, a rank-based metric that encourages an estimator to assign continuous *risk scores* to customers to quantify how confident the estimator is that they will file a claim.

The normalized Gini coefficient is computed as follows. Given a vector $\hat{y} \in \mathbf{R}^m$ of risk score predictions for customers $1, 2, ..., m$, the customers are assigned ranks $r = (r_1, ..., r_m)$, where $r_i < r_j$ if and only if $\hat{y}_i < \hat{y}_j$, i.e. customer i has a lower risk score than customer j. The ordering established by r is used to compute a piecewise-linear Lorenz curve over the true vector of claim labels, $y \in \{0,1\}^m$. For any $z \in [0, m]$, the value of the Lorenz curve is given by

$$\mathcal{L}_{\hat{y}}(z) = \sum_{k=1}^{\lfloor z \rfloor} y_{r_k} + (z - \lfloor z \rfloor) y_{r_{\lceil z \rceil}}.$$

The (non-normalized) Gini coefficient is proportional to the area between L and the line segment connecting $(0,0)$ and (m,m), which would be the Lorenz curve for a homogeneous population of individuals who all made insurance claims. This area is then scaled to the interval $[0,1]$:

$$\mathcal{G}(\hat{y}) = 1 - \frac{2}{m^2} \int_0^m \mathcal{L}_{\hat{y}}(z)\, dz.$$

A higher Gini coefficient indicates that the model generally assigned lower risk scores to customers who turned out not to file claims. The best possible Gini coefficient is obtained by a model that ranks every negative example lower than every positive example. This means the maximum possible Gini coefficient for a dataset is given by

$$\mathcal{G}^* = \mathcal{G}(y) = 1 - \frac{1}{m^2} \sum_{i=1}^{m} \mathbf{1}\{y^{(i)} = 1\}.$$

Thus, the *normalized* Gini coefficient is a function of $\hat{y}$ and y, defined as the ratio of the Gini coefficients for the predicted and actual labels:

$$\mathcal{G}_n(\hat{y}, y) = \frac{\mathcal{G}(\hat{y})}{\mathcal{G}(y)}.$$

This metric is difficult to optimize directly, since it is a function of the ordering established by $\hat{y}$ rather than a continuous function of $\hat{y}$ itself. However, it does allow us to use both classification methods and regression methods to approach the problem: either can theoretically achieve a high normalized Gini score, as long as the outputs for true positives are generally larger than those for true negatives.

5 Methods and Results

5.1 Baseline

We began our analysis by formulating a baseline model to compare our results to. Our baseline generates labels from a Bernoulli random process with the probability of filing a claim calculated as the mean percentage of customers with a claim:

$$p = \frac{1}{m} \sum_{i=1}^{m} y^{(i)}$$

This model achieves a *normalized Gini score* of $4.14 \cdot 10^{-4}$. This is expected since for the Bernoulli process, the generated labels are essentially random, and so our Gini Score should be around 0.

5.2 Exploratory data analysis

To gauge the complexity of the dataset and search for any obvious dependencies between feature values and claim-making, we selected a handful of random pairs of features and plotted all positive and negative examples in the training set in two-dimensional feature spaces (see Figure 1). These plots were not particularly helpful in exposing any simple relationships between the values of small feature sets and customers' claim-making tendencies. However, they do illustrate considerable overlap between positive and negative examples in low-dimensional space, suggesting that no small subset of features Porto Seguro collects about its customers is easily able to predict future claim-making.

5.3 Linear models

Although our exploratory plots suggested that claim-making is likely a nonlinear function of the collected features, we trained a few linear models on the data to get a baseline for the performance of more sophisticated algorithms.

In linear models, we model the output y as a function of a linear combination of the features x. Linear models are characterized by their loss function and trained using various gradient descent models. We used three linear models for our project.

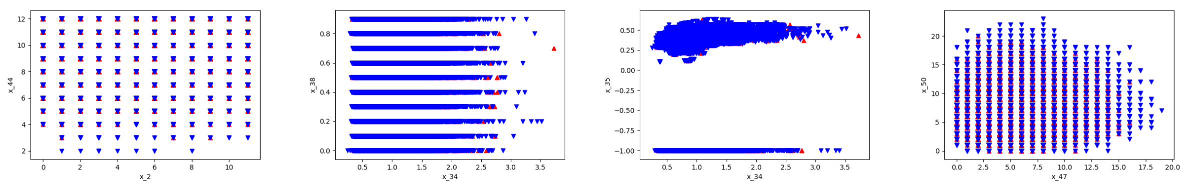

Figure 1: Plots showing distribution of positive examples (red, ▲) and negative examples (blue, ▼) across a few pairs of features. Features do not show any obvious linear dependencies between feature values and claim-making in low-dimensional space.

5.3.1 Least squares: ridge regression

In this model, we attempt to model risk scores as $\hat{y}^{(i)} = \theta_0 + \theta^T x^{(i)}$, where θ_0 is a bias term and $\theta \in \mathbf{R}^n$. The loss function is the mean squared deviation between true labels y and predicted labels $\hat{y}$, is augmented with an l_2 regularization term $\alpha \left\|\theta\right\|_2^2$ to combat overfitting.

We performed ridge regression on the training set, selecting α from a range of possible values as the one that maximized the mean value of $\mathcal{G}_n(\hat{y}, y)$ in 3-fold cross-validation. Our final value of α was $7.56 \cdot 10^2$, indicating relatively strong regularization.

5.3.2 Least squares: lasso regression

In this model, we also attempt to model risk scores as an affine function of feature vectors with a loss function based on mean squared deviation. However, our regularization term is an l_1 regularization $\alpha \left\|\theta\right\|_1$. l_1 regularization places more pressure on the model to drive feature weights θ_j towards 0, so that with large α, the model tends to incorporate a limited subset of features that contribute heavily to the loss function; the remainder of the parameters are implicitly marked as irrelevant by vanishingly small values of θ_j.

As with ridge regression, we trained several models with varying values of α, selecting the one with highest mean value of $\mathcal{G}_n(\hat{y}, y)$ under 3-fold cross-validation. This yielded an optimal α of $3 \cdot 10^{-5}$, indicating very weak regularization; in fact, we did not observe any sparsity in the resulting estimation vector θ.

5.3.3 Logistic regression

In logistic regression we use the sigmoid function to model the probability of our hypothesis $\in \{0, 1\}$. We trained a logistic regression model on the training set without regularization. For the output vector $\hat{y}$, we used the raw probabilities produced by the sigmoid function rather than converting values to binary decisions 0 or 1.

5.4 Naive Bayes

We trained a Gaussian naive Bayes classifier on the training data. This model uses the feature values from each class in the training set to estimate a Gaussian distribution for each feature conditioned on the label:

$$P(x_j^{(i)} | y^{(i)}) \sim \mathcal{N}\left((\bar{x}_j | y^{(i)}), (s_j^2 | y^{(i)})\right).$$

5.5 Ensemble methods

5.5.1 Random forests

We tested two ensemble methods on our training data, the first of which was the random forests regressor. For this problem, we can used the regressor rather than the classifier. As the Gini coefficient evaluation metric only uses the predicted $\hat{y}$ to create the rank ordering of customers, and not the value of the $\hat{y}$ in the metric, we do not need to constraint $\hat{y}$ to be in $[0, 1]$. The random forests model uses multiple decision trees on sub-samples of the training set and outputs a mean prediction of the decision trees. This ideally corrects for overfitting and improves accuracy. However, in our first attempt, we did not set a maximum depth for the tree. Our Gini coefficient on the training data was exceptionally high, around 0.99, which indicates massive overfitting. We handled the overfitting by conducting grid search with 3-fold cross validation to select the maximum tree depth that produced the highest test score. We found that the optimal maximum depth was 9, and this greatly reduced the level of overfitting as shown in the results table.

5.5.2 Gradient boosting

We also used the gradient boosting library XGBoost to fit our dataset. Gradient boosting uses an iterative process to combine several learning models which addresses the non-linearities in our dataset. However, there is also evidence of overfitting here as in the random forests model. The Gini coefficient for this model on the training set was .508, which is higher than most of the Gini coefficients that Porto Seguro has experienced in the past. In this case, we also used grid search with 3-fold cross-validation to tune the hyperparameters of learning rate and maximum tree depth in order to reduce over-fitting. We further responded to this by using randomized parameter search to test different values for the maximum tree depth, the learning rate, the minimum number of children, the subsample ratio of the training instance, and the subsample ratio of columns when constructing each tree. As shown in Figure 10, 11, and 12, tuning our hyper-parameters greatly reduced the difference between the training score and test score. Randomized parameter search produced the highest Kaggle submission test score while grid search performed well on our test set but did not generalize to the Kaggle submission test set.

5.6 Multilayer perceptron

We also attempted to train neural networks to make risk score predictions. We conducted parameter searches over a series of basic multilayer perceptron (MLP) architectures (number of hidden layers, number of neurons per hidden layer) and regularization strengths to select the models that would maximize the mean Gini score under cross-validation. All neurons used the sigmoid activation function, $\sigma(z) = (1 + e^{-z})^{-1}$.

5.6.1 One hidden layer

Parameter search over architectures with one hidden layer yielded an MLP with 220 hidden units, and a relatively weak regularization parameter $\alpha = 7.7 \cdot 10^{-4}$.

5.6.2 Two hidden layers

Parameter search over architectures with two hidden layers yielded an MLP with 100 units in the first hidden layer and 10 units in the second, with a very weak regularization parameter $\alpha = 1.3 \cdot 10^{-6}$. This model displayed severe overfitting behavior; see Figure 8.

6 Training

Training curves for each of the models are in Figures 2-12. We divided the training set provided by Porto Seguro further into a training and development/test set. 80% of the data was included in the training set and 20% of the data was left for the test set. The learning curves for each model demonstrate model performance as a function of training set size (linearly spaced from 0 to 476,169 examples). The test set size is kept constant (119,043 examples).The Gini performance of the model is plotted as a function of how much training data it has seen. The curves are fairly self-explanatory so discussion here is minimal. In general, test performance improved as training set size increased. Training performance significantly better than test performance indicates overfitting.

7 Conclusion and Future Work

Below is a table summarizing the performance of our models. In addition, Kaggle hosts a dataset for model evaluation (892,816 examples). We have included those normalized Gini scores in the table.

Model	Train Score	Test Score	Kaggle Test Score
Lasso	0.253	0.235	0.251
Ridge	0.253	0.235	0.251
Logistic Regression	0.250	0.233	0.249
Random Forests	0.300	**0.284**	0.245
XGBoost	**0.508**	0.263	0.276
w/ Grid Search	0.323	0.265	0.121
w/ Random Search	0.333	0.262	**0.279**
Naive Bayes	0.236	0.221	0.235
One-Layer Perceptron	0.267	0.232	0.255
Two-Layer Perceptron	0.491	0.159	0.175

Linear models performed surprisingly well given the lack of obvious linear relationships in the data. Ensemble methods were prone to overfitting the training data but still performed well on the test data, with the exception of XGBoost with grid search. One-layer perceptron performed well; two-layer perceptron overfit and performed poorly.

Given more time, there are a few things we could do to improve our models. First, we could explore more hyperparameters for our ensembling and neural net models. Second, we could impute missing feature values with sample means for the training set and all the variables. Currently missing feature values are only handled in the test set of a few of the models. Finally, we could ensemble our existing models by taking a weighted combination of their predictions.

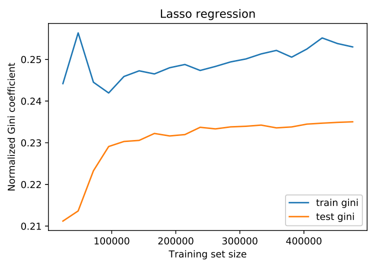

Figure 2

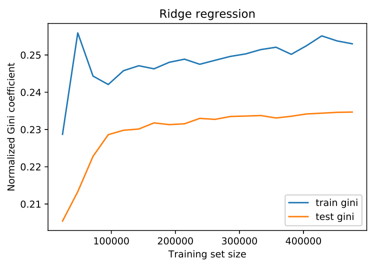

Figure 3

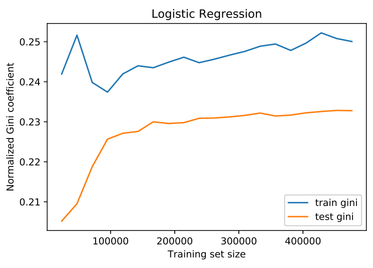

Figure 4

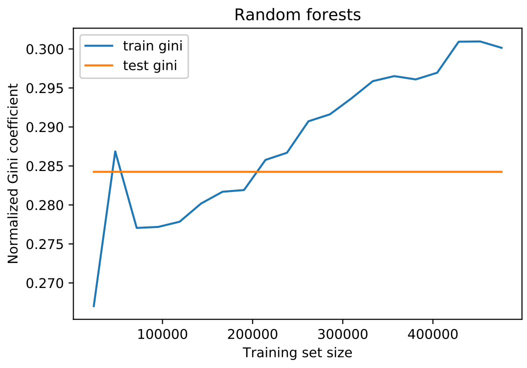

Figure 5

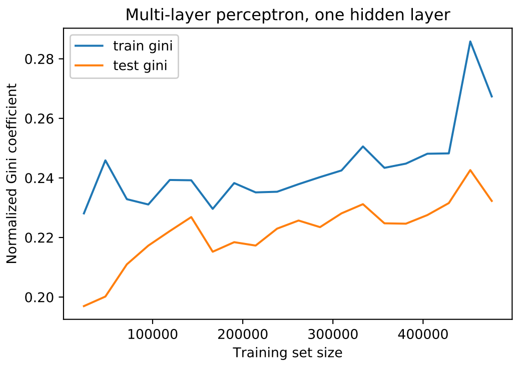

Figure 6

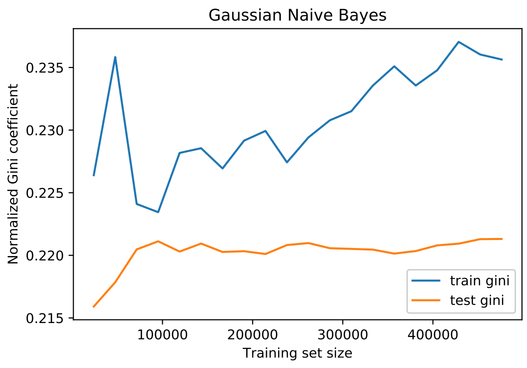

Figure 7

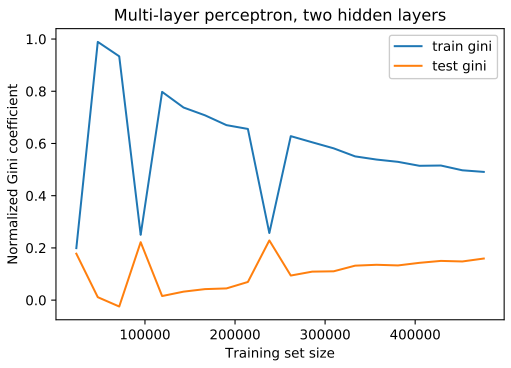

Figure 8

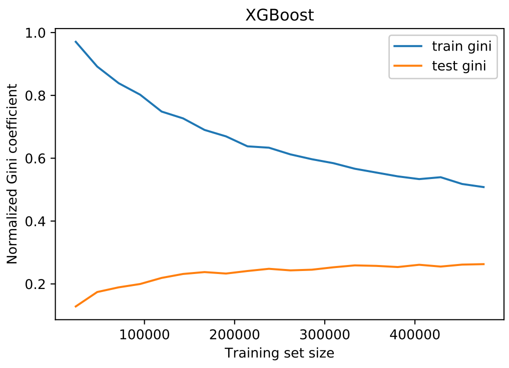

Figure 9

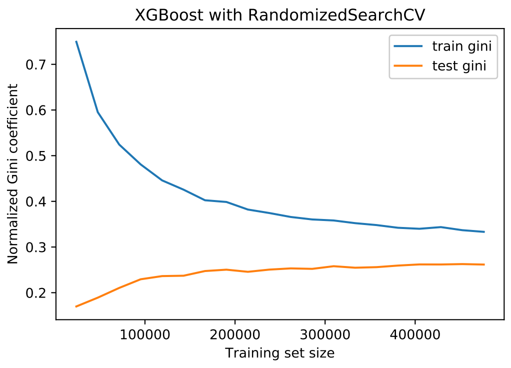

Figure 10

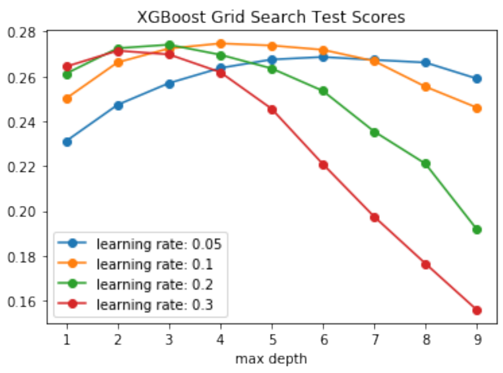

Figure 11

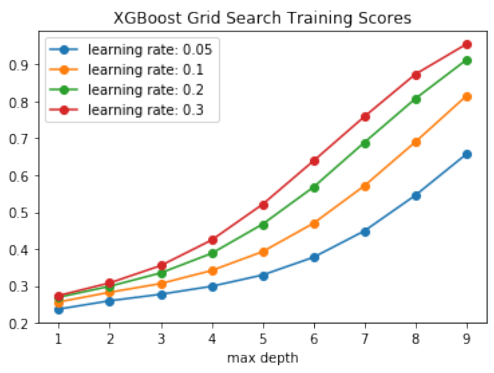

Figure 12

Contributions

Matthew Millican: python utilities, model training, plots, writeup; Laura Zhang: model training, plots, writeup; Dixee Kimball: some model training, plots, writeup

References

[1] Browne, M. (1992). Evidence of Adverse Selection in the Individual Health Insurance Market. *The Journal of Risk and Insurance,* 59(1), 13-33. doi:10.2307/253214

[2] Cummins, J., Grace, M., & Phillips, R. (1999). Regulatory Solvency Prediction in Property-Liability Insurance: Risk-Based Capital, Audit Ratios, and Cash Flow Simulation. *The Journal of Risk and Insurance,* 66(3), 417-458. doi:10.2307/253555

[3] Kaggle & Porto Seguro. (2017). Porto Seguro's Safe Driver Prediction. Retrieved from https://www.kaggle.com/c/porto-seguro-safe-driver-prediction/data

[4] Lee, S., & Urrutia, J. (1996). Analysis and Prediction of Insolvency in the Property-Liability Insurance Industry: A Comparison of Logit and Hazard Models. *The Journal of Risk and Insurance*, 63(1), 121-130. doi:10.2307/253520

[5] Salcedo-Sanz, J. S., JFernández-Villacañas, J., Segovia-Vargas, M. J., Bousoño-Calzón C. (2005).Genetic programming for the prediction of insolvency in non-life insurance companies. *Computers & Operations Research*, 32(4) https://doi.org/10.1016/j.cor.2003.08.015.

[6] Singh, Diveesh. (2016) Using Convolutional Neural Networks to Perform Classification on State Farm Insurance Driver Images. CS 229 Final Report.

[7] Smith, K., Willis, R., & Brooks, M. (2000). An Analysis of Customer Retention and Insurance Claim Patterns Using Data Mining: A Case Study. *The Journal of the Operational Research Society*, 51(5), 532-541. doi:10.2307/254184

[8] Yeo, A. C., Smith, K. A., Willis, R. J. and Brooks, M. (2001), Clustering technique for risk classification and prediction of claim costs in the automobile insurance industry. *Int. J. Intell. Syst. Acc. Fin. Mgmt.*, 10: 39–50. doi:10.1002/isaf.196