Porto Seguro Kaggle Competition

Kaggle team name: Conor McGale

The Porto Seguro data set involves data collected by a car insurance firm located in South America. By sharing a comprehensive dataset about their customers, they hope the Kaggle community can apply machine learning techniques to help better inform the pricing of their insurance premiums.

The dataset is already split into test and training data. The data has a target class of whether a customer filed an insurance claim in a given year, which is given as a binary output. The format of the desired predictions is the probability that an individual customer would file a claim in a given year.

The data is very clean and only contains numerical values. It does seem, however, that a lot of the features in the data have very little to 0 correlation to one other, implying that these are randomized variables that have been engineered. I removed these features—namely all those that included the term ‘calc’. Moreover, only a few features seemed to have a significant amount of missing entries (denoted by a ‘-1’ value).

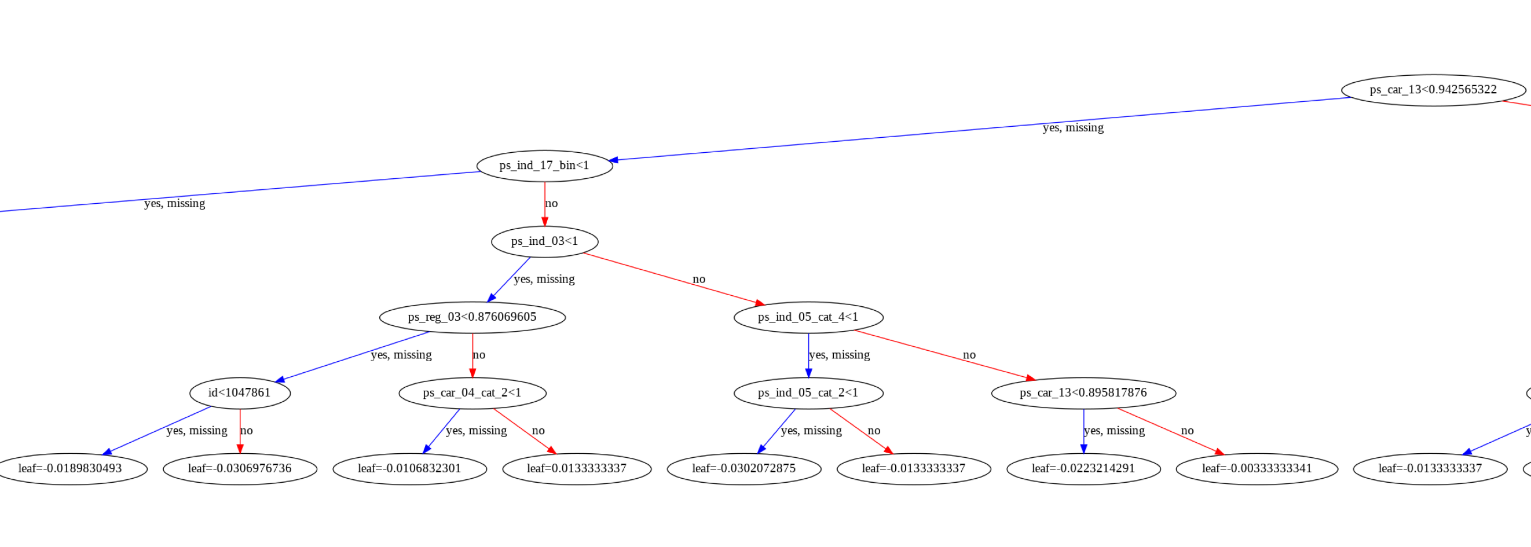
The data had very few positive cases where a customer actually made an insurance claim (under 4%). This means models would classify almost every customer as an individual that did not file a claim. In doing so, it would result in a high degree of accuracy. To combat this very imbalanced class, I under-sampled the entries with target=0. After performing this, nearly 10% of the training set contained records with a value target=1.

The model I chose is an XGBoost Classifier. After trying a few other models—namely logistic regression and a random forest—the XGBoost granted the highest score on Kaggle by a landslide. It is also a fairly ‘safe bet’ to be effective in any classification or regression problem. In comparison, it has powerful predictive abilities, and it is also fairly cheap in terms of the computational power required. This means that a more thorough investigation into the most effective hyperparameters is feasible.

XGBoost stands for ‘eXtreme Gradient Boosted trees’; a decision-tree based model. It is an ensemble method whereby multiple layers of tree models are built in order to increase the overall predictive power.

The boosting refers to the attempt to turn models from weak learners (a model that is only marginally better than randomly guessing) into strong learners (a model that has a strong correlation with the true classification). This is done by selecting errors made by previous layers of decision trees and minimizing the overall prediction error in the next layer. A large stack of layers is eventually constructed, and hopefully a strong learner is produced. A particular advantage of XGBoost is that it includes regularised boosting, which prevents overfitting.

After finding some initial success with just the standard parameters of the XGBoost Classifier, I ran a randomized search on a table of hyperparameters to fine tune the model. I chose a randomized search over a grid search to reduce computation time. The tuning process was time intensive, taking well over 2 hours to complete the randomized search. The XGBoost algorithm has a huge range of hyperparameters. The model in my final submission would most definitely have been improved by running a more comprehensive search over different hyperparameter values.

*Figure 1: This shows a subtree, from the first tree, out of the 600 included in the XGBoost model that was constructed. The root is shown in the top right. The hyperparameter ‘max depth’ determines how many decisions an individual tree would make from the root to a given leaf node. The random search performed found a max depth of 5 to be the most effective.*

My final model achieved a public score of 0.27791 and a private score of 0.28431. The score provided is a normalized gini coefficient. The maximum score is 1, and a score of 0 means the model has as much predictive power as randomly guessing. For scale, the winning public score was 0.29698.

Overall, the model built has some reasonable predictive power and is a success, although that is not to say it cannot be improved. Some areas for improvement have already been outlined in this report, but to go further, the XGBoost model could be included into an ensemble whereby several machine learning techniques are combined into one predictive model. This wasn’t performed because other models I tested had gini-coefficients of 0 or even lower- meaning they didn’t have a predictive power better than random guessing. Also, the XGBoost model is at its core is an ensemble method, meaning any gains that may be realised by further ensembling may be minimal.