Team Name: Ria Chakrabarti

Introduction

This lab asked us to participate via late submission for Porto Seguro's Safe Driver Prediction competition on Kaggle. As introduced on Kaggle, Porto Seguro is one of Brazil's largest car insurance companies. This competition and ultimately a model from this competition is meant to improve our predictions of the likelihood that a customer will make an insurance claim. Exploration of feature importance can also improve our human understanding of what can cause car accidents. In the grand scheme of things this can help many people save lives and money.

Data exploration

Three files were provided as part of this project. First, a file of over 600 thousand data points called "train.csv". Then a file was provided called "test.csv" which contained similar data to the training file without "target" values. A final model is meant to be used on this file to collect data for submission. Lastly, "sample-submission.csv" was meant to show how the submission file was meant to be formatted.

The format of data provided for this competition was a particular challenge. Aside from the columns "id" and "target", which respectively meant the id of a given customer and whether or not that customer made a claim, all other columns which represent features for the data had little indication of what they actually were meant to represent. The only information given about other columns is the "type" of data it is and a generic grouping of the data. Data types were binary, categorical, and just a generic integer/float value. Values of -1 were meant to represent missing data. Similar groupings of features were indicated by the inclusion of *ind, reg, car,* or *calc* in their name.

I first explored data by group creating histograms for each of the features. This made it possible to visualise which data points had differentiating features. For example as one can see from this distribution plot, feature ps_reg_03 had a high amount of missing data.

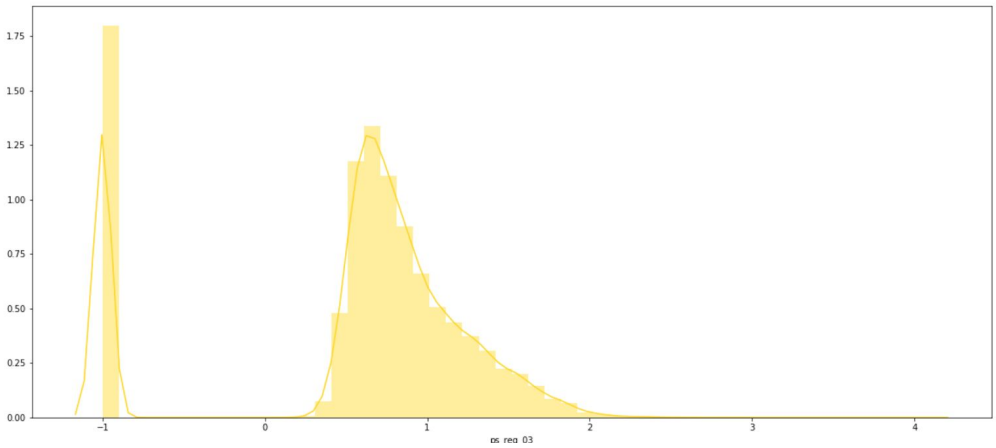
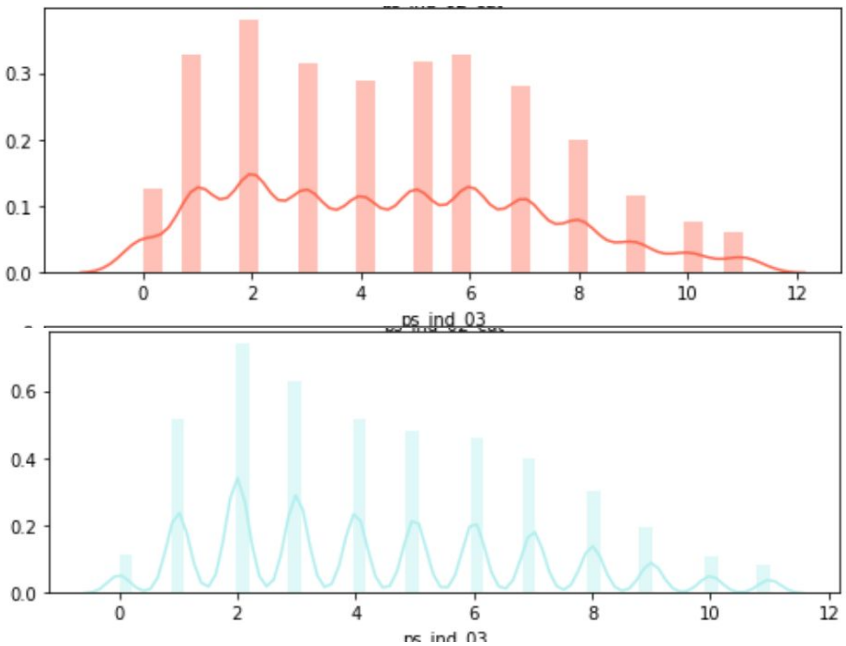


Figure 1: Distribution plot of ps_reg_13 feature.

Some inconsequential insights from this data can be read about in the mark-downs of the attached ipynb notebook. In order to gain insights on differences in features between cases in

which a claim was filed and no-claim cases I separated the data by target values ("0" or "1") and looked at the distribution plots again.



The red plot on the left shows the distribution plot of feature ps_ind_03 using only data of people that made a claim. The blue plot underneath shows the distribution plot of the same feature for people who did not make a claim. As one can see, more people who filed claims had ps_ind_03 values of 5 and 6 for that feature.

One of the biggest attributes of the training data set is the distribution of target values. About 96% of the customers represented by this data never filed a claim.

Data "manipulation"

Since over 96% of data in the training set has a target value of "0", any model that classified all rows as "0" on this training set or a sample of this training set would have 96% accuracy. One of the ways to build a better model with this problem is to undersample the data. I chose this undersampling route and created a new data set that was half made up of claim data and half no-claim data. This has the consequence of a far smaller training set.

Model Choice and Evaluation

I chose to first start off with different versions of an XGBoost model which is a version of a Decision Tree model that also focuses on "gradient boosting" using an extreme error gradient optimizer. I evaluated each type of XGBoost model I made (one with a training set, one with an eval set, and one with SelectKBest feature selection) by F1 score and with a confusion matrix for each. I also used a regular Decision Tree model as well. The XGBoost model with an eval set performed with the best F1 score of 0.556.

Results and Conclusion

After submission on Kaggle I got a public score of 0.15384. If given more time I would have attempted using different models, tried to see if the features I pointed out visually matched the most important features discovered by the models, and attempted replacing -1 value categories with the mode/median value categories instead of treating it as its own.

References:

https://www.kaggle.com/headsortails/steering-wheel-of-fortune-porto-seguro-eda/report

https://www.kaggle.com/rafjaa/resampling-strategies-for-imbalanced-datasets

https://www.kaggle.com/arthurtok/interactive-porto-insights-a-plot-ly-tutorial

https://xgboost.readthedocs.io/en/latest/tutorials/model.html

https://xgboost.readthedocs.io/en/latest/python/python_api.html