CS 514 – Applied AI Project 5: Allstate Claims Severity

Nickname – Phaedrus

Introduction

This project is on the public data science competition – Allstate Claims Severity – hosted by Kaggle.com. As the details of project are available on the website and also because all are familiar with it, I am not explaining much about it in detail. The error metric chosen here is MAE (Mean Absolute Error) as suggested by competition organizers.

Data Preprocessing

For the dataset preprocessing, I combined both the test and train data in order. Perfromed One-Hot encoding for categorical values and then separated the train the train and test data again. I also removed the 'id' and 'loss' column. The 'id' column is stored to submit the final prediction as a csv file.

Now for our project, I took the last 30% of data (47080 records) from train.csv as testing data and the remaining (141238) as training data. The removed loss column became the 'y' (or target) values of the training and testing sets.

XGBoost - Algorithm

XGBoost stands for "Extreme Gradient Boosting". XGBoost is used for supervised learning problems – where we train (or learn) the system a dataset containing multiple features to predict the value of a target or class variable. XGBoost is a tree ensembles model – a set of classification and regression trees (CART).

A CART is different from a decision tree in the sense that the leaf nodes in CART are associated with real scores that give a better and richer implementation as compared to decision trees where leaf nodes contain decision values.

A tree ensemble model sums the prediction of multiple individual trees together to give the final score. The trees usually in tree ensemble model complement each other

Tree Boosting

To learn the trees, as with all supervised learning models, an objective function is defined and is optimized. The learning process follows additive training. Since the model contains an ensemble of trees not all of them can be trained at once. Therefore an additive strategy is utilized. It fixes what has been learned and new tree is added one at a time.

The metric that we use in this problem is MAE (Mean Absolute Error). So the question of which tree has to be added next is determined by which tree maximizes the reduction of MAE.

The regularization of trees is one aspect that most tree packages trees less importantly as most usually emphasize on improving impurity. XGBoost algorithm provides a formal description and thereby a better idea of what is being learned.

XGBRegressor is a scikit-learn wrapper for XGBoost module. Following are the different paramenters that I have utilized to tune the model and obtain better results.

    max_depth = 6
    learning_rate = 0.075
    n_estimators = 1000
    seed = 0
    objective='reg:linear'
    colsample_bytree = 0.7
    subsample 0.7

Details about these parameters can be found here -
http://xgboost.readthedocs.io/en/latest/python/python_api.html