**Data preparation, exploration, visualization**

Due to this week’s models being focused on trees, minimal changes to the data was done. I only dropped a constant column and used regex to fix some of the feature labeling issues (for LightGBM). Stratified train-test-split at 80/20 and also oversampled train/test using SMOTE on the minority class only. In last week’s models, oversampling caused significant overfitting and a high false positive rate. This week I intend to see if this will happen with tree models, and if tuning the model for the oversampled sets will fix this.

**Review research design and modeling methods**

Below are the models I plan to use.

* RandomForestClassifier
* GradientBoostingClassifier
* LGBMClassifier (from LightGBM)
* XGBClassifier (from XGBoost)
* ExtraTreesClassifier
* StackingCVClassifier (from mlxtend)

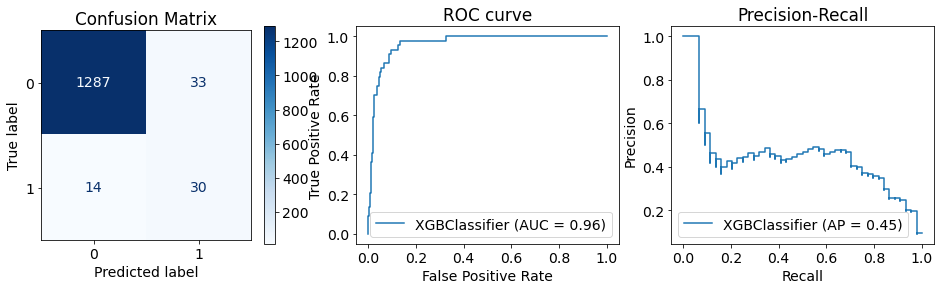
All models are tuned through Optuna, maximizing F1 score on the training set through CV. Better final F1 scores could have been achieved if the CV process was done on the test set instead, but I thought it more practical to minimize accessing the test set.

Once all models are built, I try model stacking to further improve test outcomes.

**Review results, evaluate models**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Model** | **FN** | **FP** | **TP** | **TN** | **FPR** | **TPR** | **ACC** | **PREC** | **F1** | **F1 micro** | **F1 macro** | **F1 weighted** |
| Random Forest | 26 | 22 | 18 | 1298 | 0.02 | 0.41 | 0.96 | 0.45 | 0.43 | 0.96 | 0.71 | 0.96 |
| RF smote | 17 | 33 | 27 | 1287 | 0.03 | 0.61 | 0.96 | 0.45 | 0.52 | 0.96 | 0.75 | 0.97 |
| GradientBoost | 28 | 23 | 16 | 1297 | 0.02 | 0.36 | 0.96 | 0.41 | 0.39 | 0.96 | 0.68 | 0.96 |
| GB smote | 21 | 26 | 23 | 1294 | 0.02 | 0.52 | 0.97 | 0.47 | 0.49 | 0.97 | 0.74 | 0.97 |
| LGBM | 18 | 39 | 26 | 1281 | 0.03 | 0.59 | 0.96 | 0.4 | 0.48 | 0.96 | 0.73 | 0.96 |
| LGBM smote | 18 | 27 | 26 | 1293 | 0.02 | 0.59 | 0.97 | 0.49 | 0.54 | 0.97 | 0.76 | 0.97 |
| XGBoost | 14 | 33 | 30 | 1287 | 0.03 | 0.68 | 0.97 | 0.48 | 0.56 | 0.97 | 0.77 | 0.97 |
| XGB smote | 14 | 52 | 30 | 1268 | 0.04 | 0.68 | 0.95 | 0.37 | 0.48 | 0.95 | 0.73 | 0.96 |
| Extra Trees | 16 | 38 | 28 | 1282 | 0.03 | 0.64 | 0.96 | 0.42 | 0.51 | 0.96 | 0.74 | 0.96 |
| ET smote | 17 | 35 | 27 | 1285 | 0.03 | 0.61 | 0.96 | 0.44 | 0.51 | 0.96 | 0.74 | 0.96 |

From XGBoost:



Model Stacking:

precision recall f1-score support

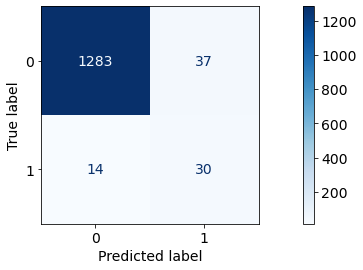
0 0.99 0.97 0.98 1320

1 0.45 0.68 0.54 44

accuracy 0.96 1364

macro avg 0.72 0.83 0.76 1364

weighted avg 0.97 0.96 0.97 1364



**Results**

As expected LightGBM and XGBoost turned out to produce the best F1 scores, and share the top 2 of every other metric.

In most cases, the SMOTE models showed better F1 scores except in XGBoost. Smote seems to reduce false negatives while increase false and true positives

**Exposition, problem description, and management recommendations**

Evaluating both weeks of models show that achieving high recall results in high false positive rates (i.e., high precision) and that models with a high F1 score are basically mediocre in both. Depending on the practical application of this model, scoring should be weighted for either costs in false positives or false negatives.

Modeling stacking also results in a strong model similar to the XGBoost result (unsurprising since XGBoost was the meta-classifier), but it has significant performance costs for limited outcome improvement. Ideally, I would have separate split datasets for base and meta classifiers.