Advances in Financial Machine Learning: Numerai's Tournament

Nitish Gade, Henry C. Ogworonjo, Yiqun Wang

***Abstract*—Numerai is a global artificial intelligence tournament aimed at making predictions of the stock market and reputed as the hardest data science competition. A competitor participates by downloading the data, making predictions and uploading the results. In this report, we describe how we approached the competition. Specifically, we explain the steps taken to develop the prediction model. This project is done as one of the tasks for the course Advances in Financial Machine Learning taught by Professor Marcos Lopez de Prado, a veteran of financial machine learning at Cornell University. Our research shows that the xgboost outperforms other tree-based ensemble methods for this dataset.**

I. Introduction

Numerai is a crowdsourced hedge fund for machine learning experts. Launched in 2015, Numerai attracts data scientists from across the globe, including those with no finance background. Competitors download the data provided by Numerai and build their own algorithm to predict regions or sectors of the stock market. It should be noted that in order to prevent the possibility of competitors incorporating their biases in the development of models, the data are obfuscated.

In this project, we applied some concepts learned in the course "Advances in Machine Learning" to solve a well-defined financial forecasting problem using the anonymized data. We applied different machine learning algorithms to the Numerai training dataset and examined their performance on the Numerai validation set. Prior to applying the machine learning algorithms, we engineered some features and selected the most suitable ones based on some defined criteria. The rest of this paper is organized as follows. In Section 2, we describe the performed feature engineering. In Section 3, we present the feature selection process. In Section 4, we describe the modeling steps and results on the train set. In section 5, we discuss the validation strategy and show the cross validation results. Finally, in section 6, we describe the submission process.

II. Feature Engineering

The Numerai training dataset used for this project has 501808 rows and 314 columns. The columns include: "id", "era", "data_type", and 310 features plus the "target". Given this large feature space, there is a high likelihood that some of these features may be linearly correlated. Therefore, for the feature engineering, we performed two main tasks to help mitigate the chance of having redundant features that may negatively impact the performance of the model that will be trained. We next discuss these two feature engineering tasks.

1) Low Mutual Information: Our first feature engineering task is to examine the features for mutual information, or substitution effects that prevents the correct selection of features resulting in overfitting. In other words, multicollinearity between features had to be taken into account. This was done by setting two correlation thresholds - between features - of 0.75 and 0.85. If any two features had an absolute correlation that exceeded the threshold, then only one of those two features would be kept. As a result, in the case of a correlation threshold of 0.75, 185 features were dropped, leaving 125 features remaining. In the case of a correlation threshold of 0.85, 156 features were dropped, leaving 154 features remaining.

2) Dimensionality Reduction: Once the low mutual information was ensured, we proceeded to do a Principal Components Analysis (PCA) on the remaining features. This is necessary to avoid curse of dimensionality and reinforce the signal by partially cancelling the noise [1] . Performing the PCA helps reduce the number of features while keeping the most important and potentially predictive principal components. We experimented with different number of principal components that are computed by setting the percentage of variance to be explained. We picked the principal components that explain 95% and 97.5% of the variance. With 95% of the variance, the number of principal components was 97, and with 97.5% of the variance, the number of principal components was 124. Therefore, we effectively reduced the feature space from 310 to 97 and 124.

III. Feature Selection

Following on from the features engineering and PCA, we proceeded to do a mean decrease impurity (MDI) analysis on the principal components' features. The MDI analysis is performed to investigate if it will select as most important the same features that PCA chooose as a confirmatory evidence that the pattern identified by the ML algorithm is not entirely overfit [2]. The MDI is a fast explanatory-importance method that is specific to tress-based classifiers [2]. With MDI, we can derive for each decision tree how much of the overfit impurity decrease can be assigned to each feature. More details of this technique can be found in 'Advances in Financial Machine Learning'. We performed the MDI analysis in accordance with the 'Snippet 8.2 MDI Feature Importance' code on page 115 of the book. The baseline classifier was a random forest classifier (with max_features = 1). The results of the MDI analysis, for when we used 97 features, are shown in TABLE I.

Feature	mean	std
feature-dexterity4	0.0108328	3.7E-05
feature-intelligence4	0.0101962	0.000109
feature-dexterity6	0.0101671	4.41E-05
feature-charisma9	0.0100187	4.75E-05
feature-dexterity12	0.0098707	3.13E-05
feature-charisma1	0.0097434	6.96E-05
feature-intelligence2	0.0095687	3.33E-05
feature-intelligence1	0.0093668	0.000175
feature-charisma43	0.0092866	4.43E-05
feature-charisma63	0.0092663	3.79E-05
feature-charisma46	0.0090733	4.34E-05
feature-charisma3	0.008994	4.97E-05
feature-constitution12	0.0089259	2.74E-05

TABLE I: MDI analysis for 97 features

Furthermore, taking the mean MDI value, and average standard deviation for all 97 features:

mean	std
0.008	3.5475E-05

TABLE II: MDI statistics for 97 features

The results of the MDI analysis, for when we used 124 features, are as below:

Feature	mean	std
feature-dexterity4	0.008874183	2.96383E-05
feature-charisma9	0.00831366	4.372336-05
feature-dexterity6	0.008263895	3.41311E-05
feature-charisma1	0.008151526	4.29688E-05
feature-dexterity3	0.007990688	2.40539E-05
feature-intelligence4	0.007950534	9.15635E-05
feature-dexterity12	0.007948761	2.74231E-05
feature-intelligence11	0.007780217	7.268894E-05
feature-dexterity1	0.007748368	2.8815E-05
feature-intelligence2	0.007688064	2.47513E-05

TABLE III: MDI analysis for 124 features

The mean MDI value, and average standard deviation of all 124 features:

mean	std
0.00649	3.0465E-05

TABLE IV: MDI statistics for 124 features

We find in these instances that the MDI analysis selects as most important the same features that PCA chose as principal, hence leading us to believe that the ML algorithm is not entirely overfit.

IV. Modelling

With the feature engineering and selection carefully done, we next fit the data to a model. We experimented with 3 tree-based methods: Random Forest, AdaBoost, and XGBoost. Our reasons for the choice of the tree-based methods are:

- Tree-based ensemble methods are known to be effective at reducing overfitting. Because of the large number of features that the data contains, we suspects that a model built on the data may be susceptible to overfitting. Therefore, since ensemble methods are known to do well are reducing overfitting, it makes natural sense to investigate a tree-based model.
- Our target is multi-class. The tree-based ensemble methods are among the algorithms that may be used for multi-class classification problems. Since this problem is treated as a multi-class classification problem, using a tree-based method is in order.
- Finally, literature shows that tree-based ensemble methods are among the best models in this problem domain [3]

To develop the model, we followed these steps:

1) Pick a features matrix generated from a specific correlation threshold (either 0.75 or 0.85) and a specific PCA explained variance threshold (either 0.95 or 0.975).
2) Train the model on the entire training data
3) Tune the hyperparameters of the model using randomized grid search.
4) Compute the performance of the model using the AUC scores as metric.

We experimented on the two feature spaces described in section II. Table V shows the AUC scores for different tree-based models when the number of features is 97. The table shows that the XGBoost outperforms the other tree-based model.

Random Forest	AdaBoost	XGBoost
0.5898	0.5974	0.6273

TABLE V: AUC scores for Correlation 0.75, PCA variance threshold 0.95

For the second case where the number of features is 124, we obtained the results shown in table VI. Once again, the XGBoost gives the best performance.

Random Forest	AdaBoost	XGBoost
0.5903	0.5984	0.6290

TABLE VI: AUC scores for Correlation 0.85, PCA variance threshold 0.975

After noticing that the XGBoost gives the best performance in both cases, the next step was to do hyperparameter tuning in order to improve out-of-sample AUC score. The hyperparameter tuning method that was followed was to do a randomized grid search with 3-fold cross-validation for each combination of hyperparameters. The parameter grid for XGBoost, was as follows:

1) learning_rate: [0.005,0.05,0.1,0.5]
2) n_estimators: [100, 150, 200, 300]
3) max_depth: [3,5,10]

The resulting best configuration of parameters and AUC score was:

learning_rate	n_estimators	max_depth	AUC
0.1	300	3	0.6302

TABLE VII: XGBoost hyper-parameter tuning

The results from the hyperparameter tuning show that out-of-sample AUC for XGBoost improved as a result of the hyperparameter tuning. Notably, the difference in AUC went from 0.6290 to 0.6302 - an increase of 0.0012.

V. Cross-validation

Having tested the three models, namely, random forest, AdaBoost, and XGBoost on the training set (with a train-test split on the training set), we next compute the models' performances on the Numerai validation set. In this section, we consider only the combination of a correlation threshold of 0.85 and PCA variance threshold of 0.975, since that combination performed best on the training set. Table VIII shows the performance of the three models on the validation set, we notice again that XGBoost performs best.

Random Forest	AdaBoost	XGBoost
0.5648	0.5591	0.5706

TABLE VIII: AUC scores on validation set

The above results were without applying the hyperparameter tuning to XGBoost. Interestingly, in this case, the configuration of parameters that performed best on the validation set is given in table IX below.

learning_rate	n_estimators	max_depth	AUC
0.05	150	5	0.5795

TABLE IX: Validation hyper-parameter tuning

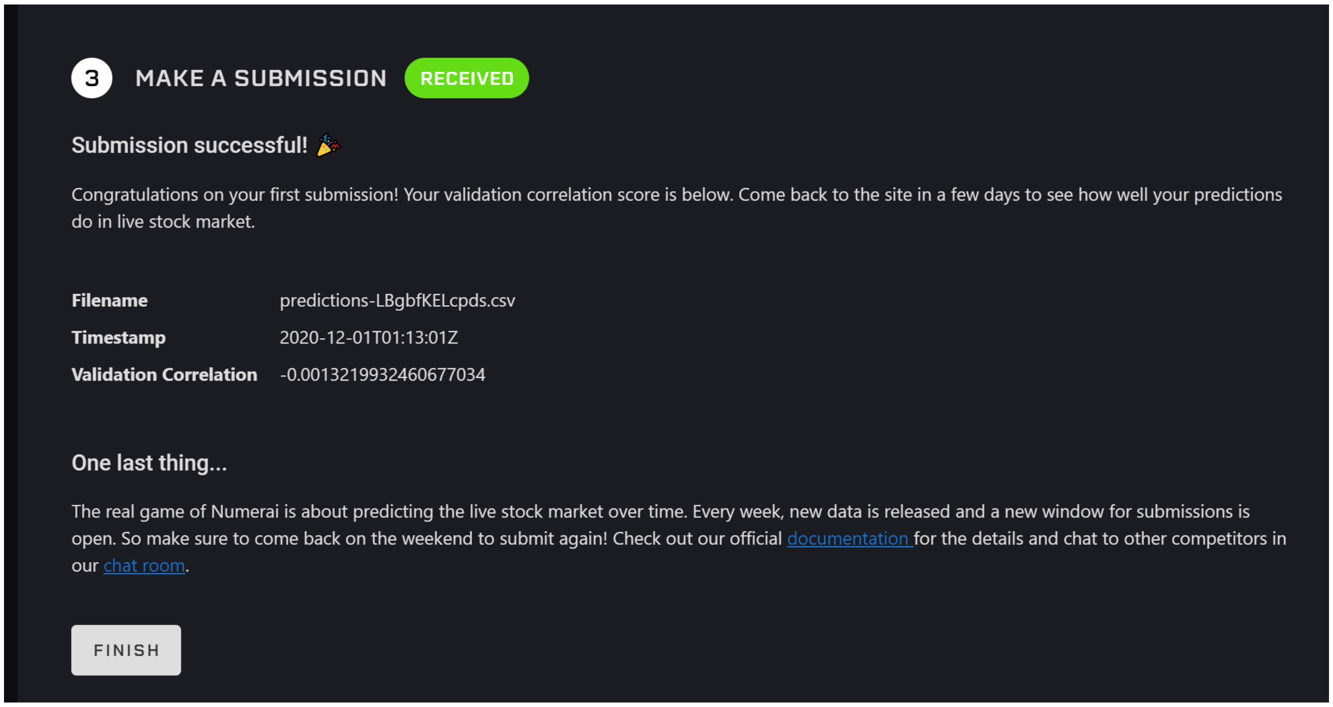
In this case the results from hyperparameter tuning again show an improvement in results. More specifically, the difference in out-of-sample AUC between the tuned XGBoost and untuned XGBoost is 0.0089.

VI. Submission

The results that we obtained on the validation set showed that the XGBoost model with tuned hyperparameters (as per Table IX) performed best. Because it had the highest AUC score, we used the tuned XGBoost model to compute forecasts on the Numerai testing set. Fig. 1 shows a screenshot of the Numerai submission page.

References

[1] M. L. De Prado, *"Advances in Financial Machine Learning:Numerai's Tournament"*, https://ssrn.com/abstract=3478927, 2019.
[2] *"Advances in financial machine learning"*, Wiley publishing, 2018.
[3] I. S. Nti, A.F. Adekoya, and B. A. W. Weyori, *"A comprehensive evaluation of ensemble learning for stock-market prediction"*, Journal of Bid Data, 2020.



Submission successful! 🎉

Congratulations on your first submission! Your validation correlation score is below. Come back to the site in a few days to see how well your predictions do in live stock market.

Filename	predictions-LBgbfKELcpds.csv
Timestamp	2020-12-01T01:13:01Z
Validation Correlation	-0.0013219932460677034

One last thing...

The real game of Numerai is about predicting the live stock market over time. Every week, new data is released and a new window for submissions is open. So make sure to come back on the weekend to submit again! Check out our official documentation for the details and chat to other competitors in our chat room.

Fig. 1: Screenshot of Numerai Submission Page