Kaggle Competition: 2sigma Using News to Predict Stock Movements

Ziwen Wang
Department of Mathematical Sciences

Yufan Liu
School of Life Sciences

Jingyu Zhang
Department of Physics

Abstract

With the rapid accumulation of financial data including changes in news and public opinion, thus it's possible to use these data to make better investment decisions. In this project, we use datasets from the 2sigma competition at Kaggle, which aims at improving our understanding of how news fluctuation might influence the performance of stock prices. For this purpose, we implement boosting methods and a fully connected neural network to investigate the stock market returns based on datasets provided by 2sigma competition, using stock data and merged data of news data and stock data, respectively. Gradually, we find that there's a slight improvement with merged data.

1 Introduction

As a scientifically driven investment manager, 2sigma has been applying technology and data science to financial forecasts for over 17 years. Now, there eager to predict stock prices by analyzing news data.

1.1 Setting of the problem

By the emergency of natural language processing technology, it's difficult to quantify news sentiment. Song et al.[1] propose that long lasting sentiment change tends to bring market impact, which can be used for prediction. By understanding the predictive power of news, it would be helpful to predict financial outcomes and generate significant economic impact all over the world [2]. In this project, we use a large set of daily market and news data provided for US-listed financial instruments. This data shall be used to predict future stock market returns. Specially, the news data have been preprocessed in advance, so it's not necessary to do NLP tasks.

The competition comprises two stages with two stages with two different evaluation periods. In the first stage, predictions are tested against historical data of the period 1/1/2017 to 7/31/2018, and then the model will be evaluated from true market data in the subsequent period. Since this competition has disclosed on 7/15/2019 and the submission will no longer accepted, so we evaluate the prediction performance by cross validation instead of future data.

1.2 Evaluation criteria

In this competition, the target of prediction output is a signed confidence value $\hat{y}_{ti} \in [-1, 1]$, which is multiplied by the market-adjusted return of a given 'assetCode' over a ten day window. If a stock is expected to have a large positive return over the next days, the value need to be set a large, positive 'confidenceValue' near 1.0, and if the stock is expected to have a negative return, the value need to be set a large, negative 'confidenceValue' near -1.0. And if it is unsure to predict the status of this asset, the value might be assigned near zero.

For each day in the evaluation time period, we calculate:

$$x_t = \sum_i \hat{y}_{ti} \tau_{ti} u_{ti}, \tag{1}$$

where τ_{ti} is the 10-day market-adjusted leading return for day t for instrument i, and u_{ti} is a binary value of $0/1$ universe variable that controls whether a particular assets is included in scoring on a particular day. Finally, the score which determines the position in the competition is composed of the mean the standard deviation of the daily value x_t:

$$score = \frac{\bar{x}_t}{\sigma(x_t)} \tag{2}$$

if $\sigma(x_t) = 0$, the score will be set to 0.

2 Exploratory Data Analysis

The datasets are provided, one for market data and one for news data, for about 3700 assets from 2007 to 2016. We read datasets with vaex[3], and finally merge two datasets on ‘date‘, ‘assetCode‘ and ‘assetName‘ after data cleaning for model preprocessing.

The market data provided by Intrinio with more than 4 million rows has 15 features, this dataset contains financial market information such as opening price, closing price, trading volume, calculated returns, etc. Each asset is identified by an ‘assetcode‘, and a single company may have multiple ‘asset Code’s. The set of included instruments changes daily and is determined based on the amount traded and the availability of information. The number of all tradable asset at a given day ranges from roughly 1300 to1800. ‘returnsOpenNextMktres10‘ is a 10 day market-adjusted residualized return as the target variable used in competition scoring for the prediction task.

The news data provided by Thomson Reuters with more than 9 million rows has 37 features, this data set contains.

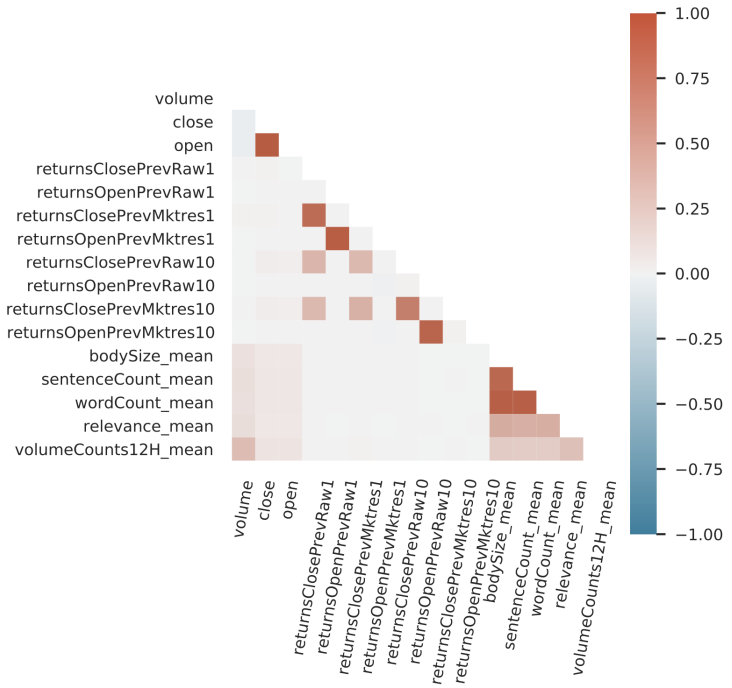

Figure 1: Correlation plot of merged data

3 Methods

After data cleaning and preprocessing, 'returnsOpenNextMktres10' in merged dataset is transformed into label for modeling. We normalize numerical data an encode codes, and split the dataset into training data and testing data for cross validation. All the methods are used for market data alone and merge data of news and market data. The following supervised learning algorithms are applied for prediction:

3.1 Logistic Regression (LR)

Logistic regression is a classic statistical model for modeling a binary dependent variable. We use this method to classify our dataset, which output is viewed as our baseline in this project for algorithms contrast, and the classifier could be described as:

$$l = log_b \frac{p}{1-p} = BX, \forall p = P(Y = 1) \tag{3}$$

3.2 Fully Connected Neural Network (FCNN)

Fully connected neural networks (FCNN) are a type of artificial neural network where the architecture is such that all the nodes, or neurons, in one layer are connected to the neurons in the next layer. We implement FCNN with Keras. After adjusting parameters, we finally determine to use 5 hidden layers. These hidden layers contain 4 Rectified Linear Unit (ReLU) activation functions, and a sigmoid function as last one layer. An embedding layer learns the encoding of the asset codes which are then concatenated to the numerical data. Binary cross-entropy is used as loss function and Adam optimizer is implemented with learning rate = 0:001; $\beta_1 = 0:9; \beta_2 = 0:999$. The model has about 50000 trainable parameters. Our activation function and loss function is described as:

$$ReLU(x) = max(x, f(x)) \tag{4}$$

$$Sigmoid(x) = \frac{1}{1+e^{-x}} \tag{5}$$

$$Loss = -\frac{1}{output_{size}} \sum_{i=1}^{output_{size}} y_i log\hat{y}_i + (1-y_i) log(1-\hat{y}_i) \tag{6}$$

3.3 Light Gradient Boosting Machine (LGBM)

Gradient boosting decision tree is a wide-used model for classification. We implement LGBM with lightGBM package[4]. LightGBM(LGBM) is an open-source gradient boosting library that has gained tremendous popularity and fondness among machine learning practitioners. It has also become one of the go-to libraries in Kaggle competitions. It can be used to train models on tabular data with incredible speed and accuracy. Binary cross-entropy is used as loss function is used with sampling rate = 0.9. The depth is set to be unlimited. And the hyper-parameters are fixed in advance described in our attched codes. Cross validation is applied for optimize process with 5 folds, and base learning rate is set to 0.19 while minimal learning rate is set to 0.01. The objective function and optimization process are described as:

$$F = \sum_k F_k(x) \tag{7}$$

$$r_k = -\frac{dL(y, F_{k-1}(x))}{dF_{k-1}(x)} \tag{8}$$

$$L = \sum_i l(y_i, \hat{y}_i) + \sum_k \Omega(F_k) \tag{9}$$

3.4 eXtreme Gradient Boosting (XGBoost)

We implement XGBoost with xgboost[5]. It is similar to LGBM, they both come from the gradient boosted trees. The difference between XGBoost and LGBM is the definition of objective function in (9). LGBM uses a novel technique of Gradient-based One-Side Sampling (GOSS) to filter out the data instances for finding a split value while XGBoost uses pre-sorted algorithm and histogram based algorithm for computing the best split. We implement XGBoost and LGBM at the same time to make a comparison between these two algorithms. According to previous research, XGBoost achieves higher accuracy rate while LGBM trains faster on a Kaggle dataset about flight delay . We are interesting in these two algorithms' performance on Two-Sigma datasets. Will LGBM have speed advantage? Will XGBoost have a accuracy advantage? When we have the exact performance data, we can make a tradeoff between those algorithms.

4 Results and Discussions

We run the model with market data only and merged data to compare results in different model. Compare the LR results as baseline, we can find that accuracy is same in two data as score is better in merged data. Generally, FCNN, LGBM and XGBoost perform better than LR. The poor results of LR is expected because the algorithm assumes linear relationships. Adding news data doesn't improve the results. This may come from the reason news data has many noises and non-related features. Besides, too much missing value in the news features in merged data after joining two datasets. Although adding news data does not have noticeable effect on the performance, the results excepted LR in merged data are about or over 0:5.

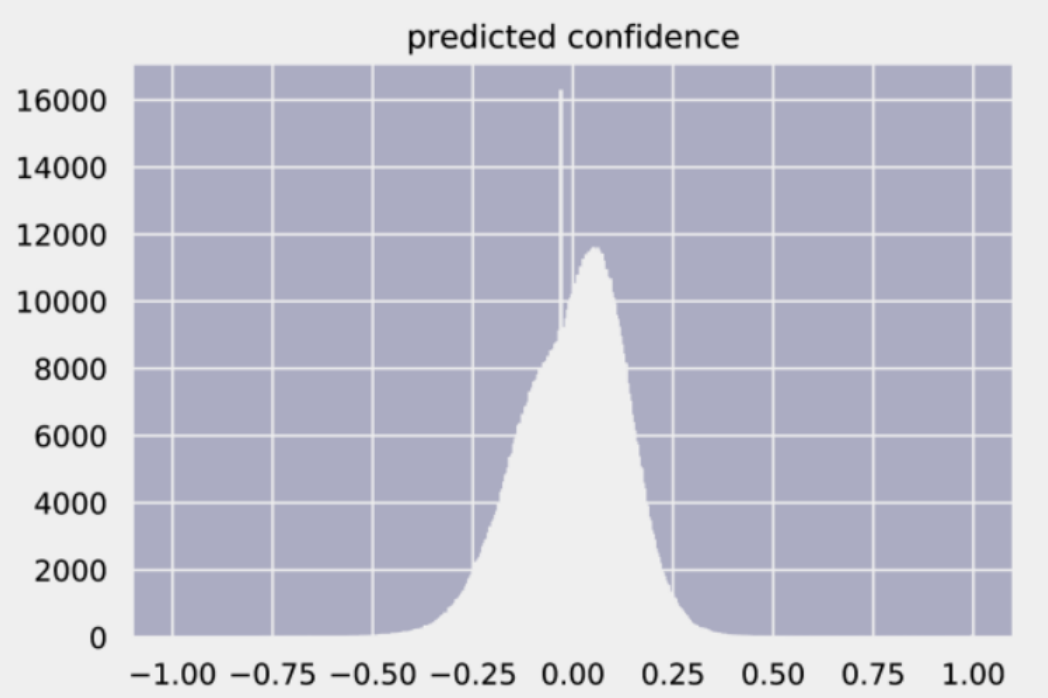

Figure 2: Prediction confidence of FCNN

As for accuracy and score, scores are less than accuracies in LR as baseline, it may be that the score calculation considers not only accuracy, but also market return and standard deviation of the prediction. In practice, we need to evaluate stocks by score instead of accuracy. Comparing LR,FCNN,LGBM and XGBoost models, we can get two conclusions:

(1) In our situation, XGBoost model performs best. However, we don't have the original test data and only split the train and test data among the original train data, so XGBoost model's performance may be over estimated.

(2) The positive impact of news data remains unknown. In LR and XGBoost models, The news data seems to be of no effect. In FCNN model, the news data seems to have negative effect. While in LGBM model, he news data seems to have positive effect.

Table 1: Model contrast with market data only

	LR	FCNN	LGBM	XGBoost
Accuracy	0.538	0.557	0.548	0.609
Score	0.325	0.671	0.403	1.957

Table 2: Model contrast with merged data

	LR	FCNN	LGBM	XGBoost
Accuracy	0.538	0.553	0.560	0.609
Score	0.327	0.623	0.499	1.957

5 Summary and Conclusions

Although we can improve the classification results in this project, the score isn't good enough. In the future, we need to implement more feature engineering to filter out outliers and useless features. Time-related model, such as LSTM, may be applied because the stock data is time series. Besides, we can also increase text information in news data by web crawler and analyze much on NLP. The fact that news may not contribute to stock movement predictions reflects market is a complicated environment. Because the time of news releasing is obscure in our dataset, the situation that using future news to prediction past stock prices could happen. There are some ways that may help improve the performance.

References

[1] Song Q., Liu A., Yang S. Y., Deane A. & Datta K. (2015) An Extreme Firm-Specific News Sentiment Asymmetry Based Trading Strategy. *2015 IEEE Symposium Series on Computational Intelligence*, pp. 898-904, doi: 10.1109/SSCI.2015.132.

[2] Shynkevich Y, McGinnity T.M., Coleman S., & Belatreche A. (2015) Stock price prediction based on stock-specific and subindustry-specific news articles. *2015 International Joint Conference on Neural Networks (IJCNN)* , pages 1–8.

[3] Breddels M. A., & Veljanoski J. (2018) Vaex: Big Data Exploration in the Era of Gaia. *Astronomy Astrophysics* **618**. doi:10.1051/0004-6361/201732493.

[4] Ke G., Meng Q., Finley T., Wang T., Chen W., Ma W., Ye Q., & Liu T. Y.. (2017)Lightgbm: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, pages 3149–3157.

[5] Chen T & Guestrin C. (2016) Xgboost: A scalable tree boosting system. *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794.