# Stock Performance Predictions based on News Analytics

Fan Gao*, Connor Gatlin†, and Ruisi Wang‡

Columbia University, New York, NY

## Abstract

*Historically, news and stock performance are highly correlated. Our goal here is to understand the predictive power of news. In this exercise, we not only interact with historical data, but also encounter synthetic market and news data that is designed to simulate the volume, timeline, and the computational burden that real future data will introduce. Two tree-based models and one neural network-based model are used for prediction. We conclude that news data indeed add value to stock performance predictions and in conjunction with market data, it shows great improvement over the benchmark.*

**Keywords:** News Analytics, Stock Performance Predictions, Finance, Big Data Analytics.

## 1. Introduction

The Efficient Market Hypothesis (EMH) in financial economics states that asset prices fully reflect all available information. Many researchers believe that the market is weak-form or semi-strong-form efficient. Nevertheless, you get a sense that the stock market has become more efficient in the past decade due to cheaper and faster information flow. The moment a company makes an investment decision or releases its earnings, everyone in the world has access to that information. Investors quickly act upon that information with their bank accounts. Within days or even hours, the stock price of the company reflects all the information.

Over the years, many researchers and investors ask whether the market is predictable given market data and news analytics? The ubiquity of market and news data today enables investors at any scale to make better investment decisions. The challenge is ingesting and interpreting the data to determine which data is useful, finding the signal in this sea of information.

*fg2432@columbia.edu
†ceg2195@columbia.edu
‡rw2720@columbia.edu

By analyzing news data to predict stock prices, we hope to advance the state of research in understanding the predictive power of the news. This power, if harnessed, could help predict financial outcomes and generate significant economic impact, making the market even more efficient.

Data for this analysis comes from the following sources:

- Market data provided by Intrinio.
- News data provided by Thomson Reuters.

## 2. Related Work

Liu [5] proposed a Attention-based LSTM model(At-LSTM) to predict the directional movements of Standard & Poors 500 index and individual companies stock price using financial news titles. Experimental results suggests that his model is promising and competitive with the state-of-the-art model which incorporate knowledge graph into the learning process of event embeddings.

Andres Garcia-Medina et al. [1] used Random Matrix Theory (RMT) and information theory to analyze the correlations and flow of information between 64,939 news from The New York Times and 40 world financial indices during 10 months along the period 2015-2016. The set of news was quantified and transformed into daily polarity time series using tools from sentiment analysis. Results from RMT shows that a common factor lead the world indices and news, and even share the same dynamics. Furthermore, the global correlation structure has found preserved when adding white noise, which indicate that correlations are not due to sample size effects. Likewise, they found a lot of information flowing from news to world indices for specific delay, being of practical interest for trading purpose. Their results suggest a deep relationship between news and world indices, and show a situation where news drive world market movements, giving a new evidence to support behavioral finance as the current economic paradigm.

Joshi, Bharathi, Rao [2] used polarity detection algorithm for initially labelling news and making the train set. They created dictionaries for positive and negative

words using general and finance specific sentiment carrying words. They also created their own dictionary for stop words removal which also includes finance specific stop words. Based on this data, they implemented three classification models and tested under different test scenarios. They showed that Random Forest worked very well for all test cases ranging from 88% to 92% accuracy. Accuracy followed by SVM is also considerable around 86%. Naive Bayes algorithm performance is around 83%. Given any news article, it would be possible for the model to arrive on a polarity which would further predict the stock trend.

# 3. Data Exploration

## 3.1. Market Data

### 3.1.1 Description

The data [3] includes a subset of US-listed instruments. The set of included instruments changes daily and is determined based on the amount traded and the availability of information. This means that there may be instruments that enter and leave this subset of data. There may therefore be gaps in the data provided, and this does not necessarily imply that that data does not exist (those rows are likely not included due to the selection criteria).

The market data contains a variety of returns calculated over different timespans. All of the returns in this set of market data have these properties:

- Returns are always calculated either open-to-open (from the opening time of one trading day to the open of another) or close-to-close (from the closing time of one trading day to the open of another).
- Returns are either raw, meaning that the data is not adjusted against any benchmark, or market-residualized (Mktres), meaning that the movement of the market as a whole has been accounted for, leaving only movements inherent to the instrument.
- Returns can be calculated over any arbitrary interval. Provided here are 1 day and 10 day horizons.
- Returns are tagged with 'Prev' if they are backwards looking in time, or 'Next' if forwards looking.

All of the fields in Market Data are shown in Table 3. The data covers period between 2007 to 2018. In total, there are 4,072,956 samples in the training data.

### 3.1.2 Preprocessing

From the price history of 5 randomly picked assets in Figure 1, we learn that securities could have dramatically different lengths of history in this dataset. Learning from securities with longer history is probably more meaningful.

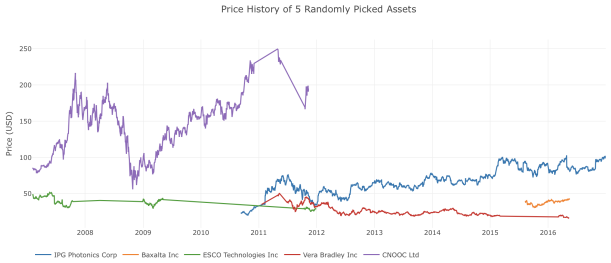


Figure 1. Price History of 5 Randomly Picked Assets

From Figure 2, we see the impact of major news on the market. Some events only left a small dent on stock prices, while others had more profound impact on the market.

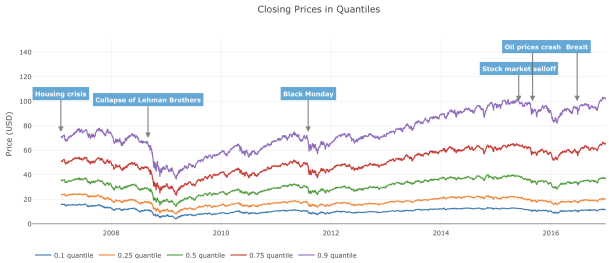


Figure 2. Closing Prices in Quantiles

We calculate 1-day price change by subtracting open price from close price and plot the standard deviations of 1-day price change by month. In figure 3, the largest bubble shows that the price of a stock dropped by $9948.99 in a day. The other large bubbles and their scales also look suspicious. We show the information of these securities in Figure 4.

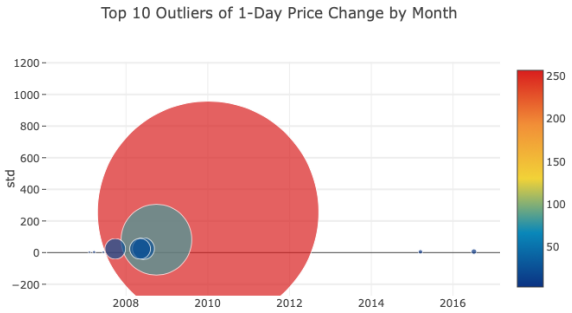


Figure 3. Top 10 Outliers of 1-Day Price Change (Before)

Upon verifying the open prices of these securities via other source, we conclude that the data is incorrect. Looking further into the data, we found 38 securities whose prices doubled and 16 securities whose prices fell by more then 50% in one day. These outliers are likely to distort our model. Our solution is to replace them with the mean. We

| | time | assetCode | assetName | volume | close | open |
|---|---|---|---|---|---|---|
| 1127598 | 2010-01-04 22:00:00+00:00 | TW.N | Towers Watson & Co | 223136.0 | 50.00 | 9998.9900 |
| 627547 | 2008-09-29 22:00:00+00:00 | BK.N | Bank of New York Mellon Corp | 18718479.0 | 26.50 | 3288.1136 |
| 502997 | 2008-06-05 22:00:00+00:00 | AHG.N | Apria Healthcare Group Inc | 801892.0 | 17.29 | 999.9900 |
| 471381 | 2008-05-06 22:00:00+00:00 | CEPH.O | Cephalon Inc | 4846.0 | 61.04 | 999.9900 |
| 242847 | 2007-09-27 22:00:00+00:00 | EXH.N | Archrock Inc | 490100.0 | 79.99 | 999.9900 |

Figure 4. Suspicious Data Points

show the adjusted dataset in Figure 5.

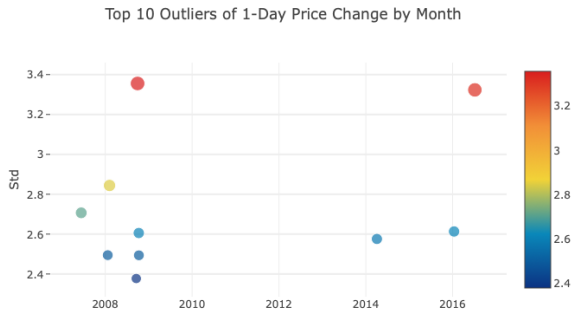


Figure 5. Top 10 Outliers of 1-Day Price Change (After)

Now let's take a look at the return variables. As shown in Figure 6, we see wide dispersions during the financial crisis. Other times, the returns fluctuate around long-term mean.

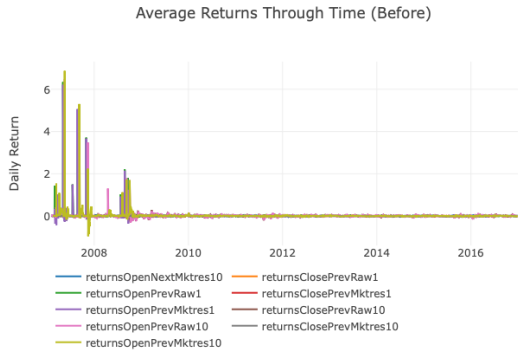


Figure 6. Returns Through Time (Before)

If these influential observations are ignored, our results are likely dominated by them, which are not reflective of the general trend. We notice that some teams simply trim off data prior to 2009 because returns were extremely volatile during that period. In our opinion, a better way to mitigate the impact of these extreme data points without losing any valuable data is to censor the data. In this case, we think a +/- 20% movement in 10 days is representative of reactions to market events. Therefore, the returns are clipped at +/- 20%.

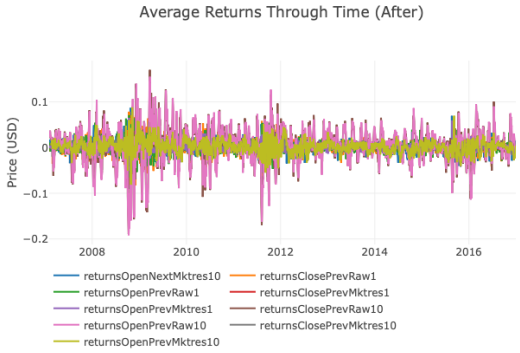


Figure 7. Returns Through Time (After)

The adjusted returns are shown in Figure 7. They look much more reasonable after the treatment.

## 3.2. News Data

### 3.2.1 Description

The news data contains information at both the news article level and asset level (in other words, the table is intentionally not normalized).

All of the fields in News Data are shown in Table 4. The data covers period between 2007 and 2018. In total, there are 9,328,750 samples in the training data.

### 3.2.2 Preprocessing

Headline is the only text field in in the news dataset. Figure 8 shows the word cloud of the first 100,000 headlines. As you can see, they are words often used to describe companies' operational performance and media appearance.

In Figure 9, we see that the urgency code is disproportionate – two thirds of its mass is in bucket #3, one third of its mass is in bucket #1 and nothing is in bucket #2.

In Figure 10 and Figure 11, we see that most of the articles have less than 2500 words and 200 sentences. There exist, however, outliers with more than 20,000 words and 1200 sentences.

Each article is mapped to multiple headline tags. After we separate and bucket them, we have counts for all the tags shown in Figure 12. Lots of articles are not even tagged. This is another unbalanced categorical variable.

The distributions of 3 sentiment categories – positive, neutral, and negative – are shown in Figure 13. The sentiments are bipolar, with a large portion concentrated around neutrality and a small portion clustered around the apex. This indicates that the confidence on these sentiment scores is low and only a small number of articles show strong sentiments.

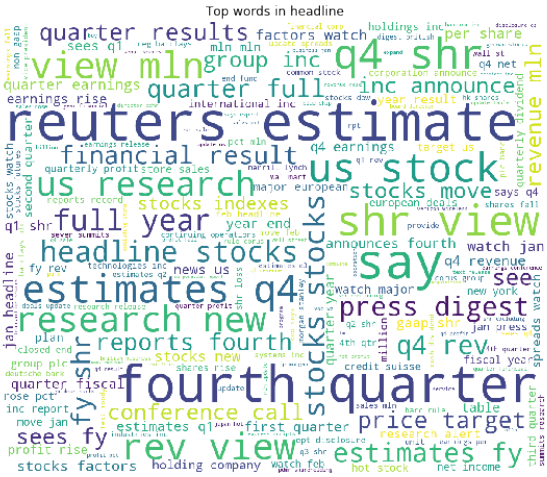


Figure 8. Top Words in Headline

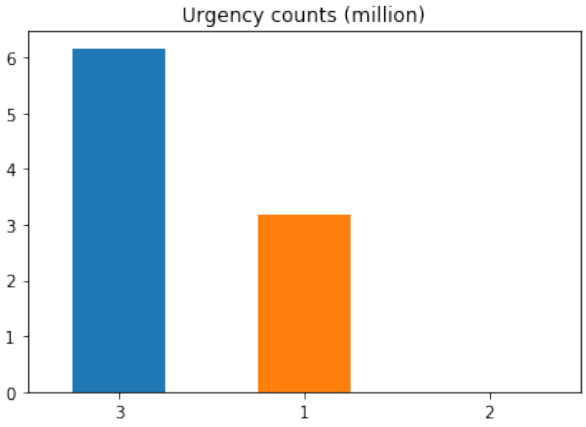


Figure 9. urgency Counts (million)

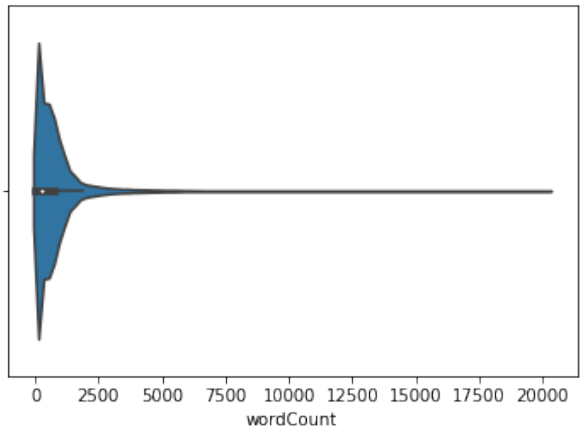


Figure 10. Distribution of wordCount

We also see that a number of companies such as Apple, Citigroup, Barclays show up in more than one sentiment

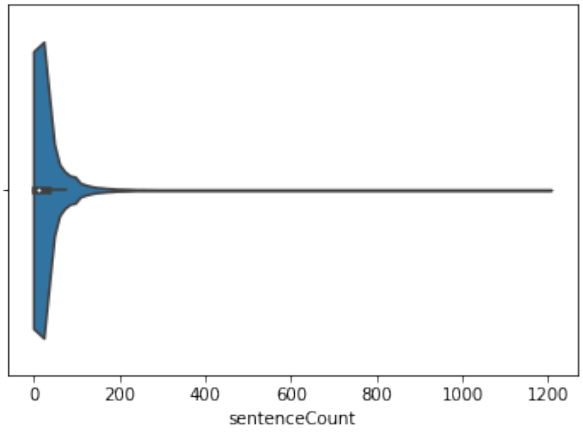


Figure 11. Distribution of sentenceCount

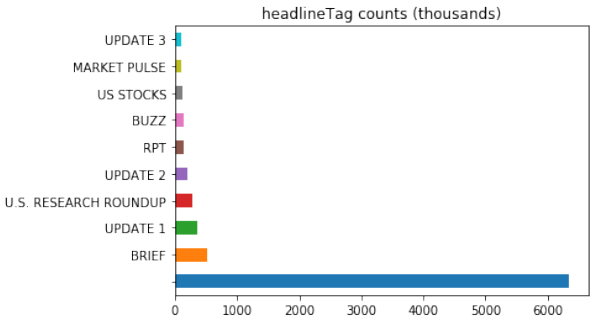


Figure 12. headlineTag Counts (thousands)

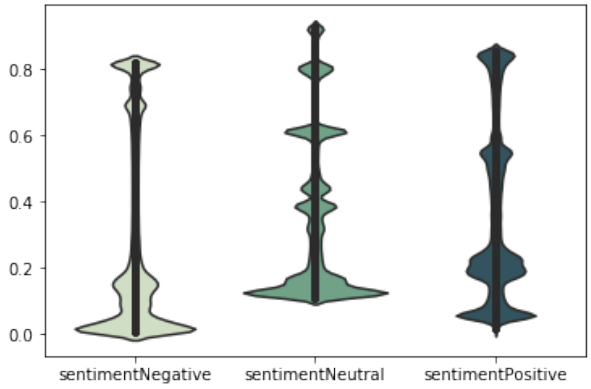


Figure 13. Distribution of Sentiments

classes. It makes sense as these large companies get more attention from the press. From time to time, they could receive very different sentiments. See details in Table 5, 6, and 7.

### 3.3. Data Visualization

In order to aid in our initial data exploration, we created a Node.js-based website to visualize the market and news data sets. Using the UI, we can query historical information

for a specific company's stock and view the D3.js generated market analysis graph which includes close prices, moving averages, volume and RSI plots as well as the pertinent news articles and their sentiment categories. The custom website visualization for the "AAPL" stock between 2007 and 2010 is shown in Figure 14. We can use this website to identify key discrepancies and outliers within the data.

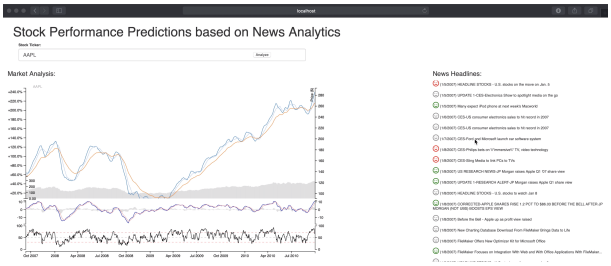


Figure 14. Website Visualization for Apple Stock

### 3.4. Feature Engineering

#### 3.4.1 Extract Market Features

First, all columns are casted from float64 to float32 to save memory. The precision difference shouldn't affect the accuracy of the model. Second, we extract more features from the original data, including close to open ratio, volume to average volume ratio, previous 1-day close to open raw return, previous 10-day close to open raw return, previous 1-day close to open market residual return, previous 10-day close to open market residual return, previous 1-day to previous 10-day open to open raw return, previous 1-day to previous 10-day close to close raw return, previous 1-day to previous 10-day open to open market residual return, previous 1-day to previous 10-day close to close market residual return.

#### 3.4.2 Extract News Features

As mentioned earlier, there is textual and categorical data in the news data. We need to transform them into numerical data for further analysis. First, "headline" is converted from string to vector representation using a TF-IDF transformer. Second, "subjects" is unpacked from lists of subjects into an one-hot representation simply by counting the top 20 subjects ordered by term frequency.

### 3.5. Merge Data

For news data, the challenge is that we may have multiple news articles associated with a company on a given day. Out approach is grouping the data by date and company first, and then aggregating a number of key statistics from articles in each group, including the count of urgency, the maximum of takeSequence, the mean and standard deviation of bodySize, wordCount, sentenceCount, companyCount, relevance, noveltyCount12H, noveltyCount24H, noveltyCount3D, noveltyCount5D, noveltyCount7D, volumeCounts12H, volumeCounts24H, volumeCounts3D, volumeCounts5D, volumeCounts7D, the mean of marketCommentary, headline 1-20, subject 1-20, and the maximum, minimum, mean, and standard deviation of sentimentNegative, sentimentNeutral, sentimentPositive, sentimentWordCount. We pick these metrics based on our beliefs of what information may add value. It's a balance between quantity of information and computational burden - too few features would have compromised the rigor of this study, and too many features would have drained our computing resources.

## 4. Modeling

### 4.1. Evaluation

We predict 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. For a stock that we expect to have a large positive return–compared to the broad market–over the next ten days, we assign it a large, positive confidence value (near 1.0). For a stock that we expect to have a negative return, we assign it a large, negative confidence value (near -1.0). When unsure, we assign it a value near zero.

For each day in the evaluation time period, we calculate

$$x_t = \sum_i \hat{y}_{ti} r_{ti} u_{ti}$$

where $r_{ti}$ is the 10-day market-adjusted leading return for day $t$ for instrument $i$, and $u_{ti}$ is a 0/1 universe variable (see the data description for details) that controls whether a particular asset is included in scoring on a particular day.

The final score is then calculated as the mean divided by the standard deviation of the daily $x_t$ values:

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

If the standard deviation of predictions is 0, the score is defined as 0.

### 4.2. LightGBM

Gradient Boosting Decision Tree (GBDT) is a popular machine learning algorithm, and has quite a few effective implementations such as XGBoost and pGBRT. LightGBM addresses efficiency and scalability issues in earlier implementations via two techniques: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB).

With GOSS, the algorithm excludes a significant proportion of data instances with small gradients, and only uses the rest to estimate the information gain. The authors [4] prove that, since the data instances with larger gradients play a more important role in the computation of information gain, GOSS can obtain quite accurate estimation of the information gain with a much smaller data size. With EFB, the algorithm bundles mutually exclusive features (i.e., they rarely take nonzero values simultaneously), to reduce the number of features. The authors prove that finding the optimal bundling of exclusive features is NP-hard, but a greedy algorithm can achieve quite good approximation ratio (and thus can effectively reduce the number of features without hurting the accuracy of split point determination by much). LightGBM is proven on multiple public datasets that it speeds up the training process of conventional GBDT by up to over 20 times while achieving almost the same accuracy.

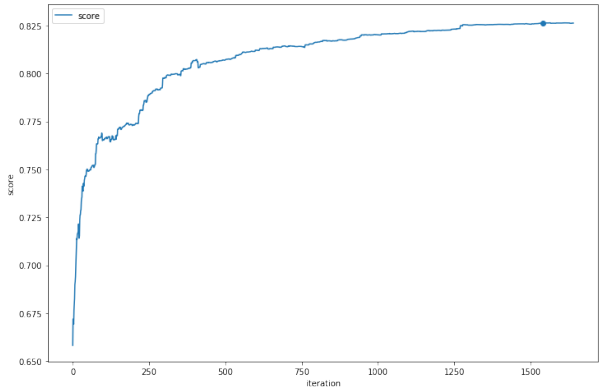


Figure 15. Score on Evaluation Data During Training

One challenge in using LightGBM is that it has more than a dozen parameters. It uses the leaf-wise tree growth algorithm, while many other popular tools use depth-wise tree growth. Compared with depth-wise growth, the leaf-wise algorithm can converge much faster. However, the leaf-wise growth may be over-fitting if not used with the appropriate parameters. To achieve fast speed, better accuracy and deal with overfitting, we use a Bayesian optimizer called skopt to optimize the hyper-parameters. skopt is a simple and efficient library to minimize (very) expensive and noisy black-box functions. We set the loss to be the negative score on validation data after 500 boost rounds. As shown in Figure 16, the loss function slowly converges.

After training, we looked at feature importance in the final model. Market data such as previous 10-day returns, volume, open to close dominate the model. Sentiments scores get median ranks among all the features, proving their value. On the other hand, text features such as headline and subjects only have minimal contributions to the model.

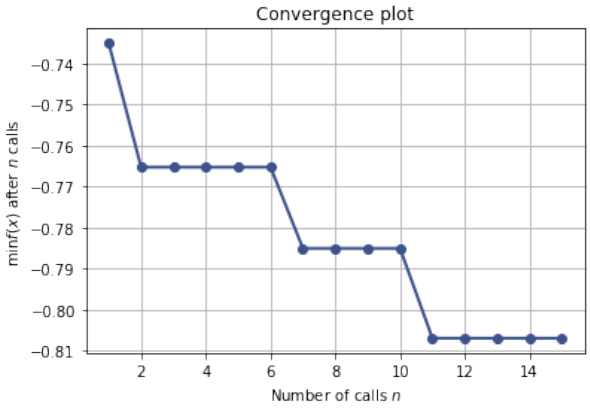


Figure 16. Hyper-Parameter Tuning Using Bayesian Optimization

### 4.3. Catboost

Similar to the LightGBM model, the CatBoost model utilizes GBDT as base predictors. The primary algorithmic advantages of CatBoost are its implementation of ordered boosting (a permutation-driven alternative to classic boosting) and its effective handling of categorical features [6]. CatBoost has been proven to provide the best results on many datasets compared to XGBoost, LightGBM, MatrixNet and H2O models. As well, several datasets have been proven to train significantly quicker with the CatBoost model than other similar boosting algorithms, thus making it more ideal for larger datasets.

In this exercise, CatBoost's Classifier model is used, which after fitting provides the probability that an object belongs to the returned class. We can categorize the merged market and news dataset into one of two labeled classes: any instance that has a positive 10 day leading market return and any instance that has a zero or negative 10 day leading market return. Because we are dealing with a 2-class classification problem (positive or negative return), the loss function that we use in the classification model is LogLoss as opposed to "MultiClass". For hyperparameter tuning of the CatBoost Classifier, we use a parameter grid search with cross-validation on the train set to find the optimal model parameters without overfitting to the dataset. Per the results of the grid search, we use a learning rate of 0.1 and a maximum tree depth of 10 in the classification model. Predicted confidence on the training dataset is shown in 17.

### 4.4. LSTM

Long short-term memory (LSTM) units are units of a recurrent neural network (RNN). It can process data sequentially and keep its hidden state through time. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. Stock market has a lot

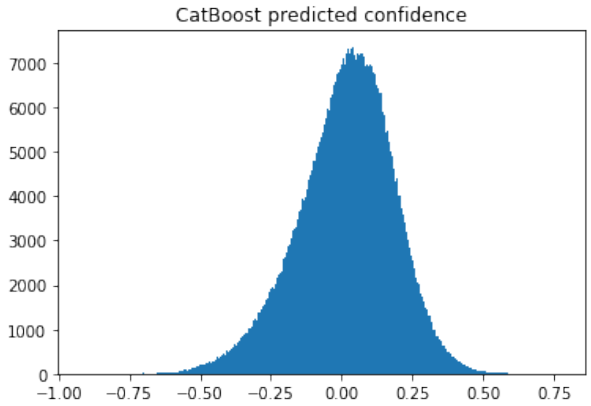


Figure 17. CatBoost Predicted Confidence

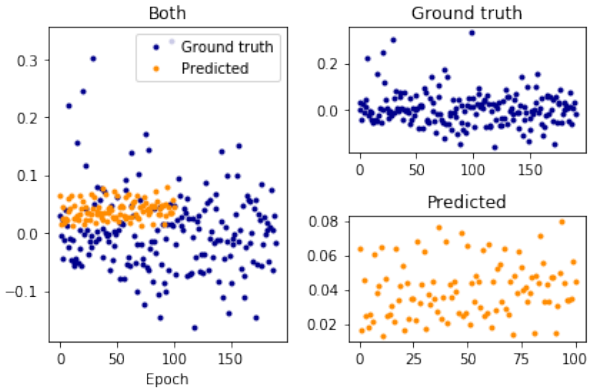


Figure 18. LSTM Predictions with 100 Assets Samples

of times series data which makes it possible to implement the LSTM framework. With the introduction of LSTM, the analysis of time dependent data become more efficient. These type of networks have the capability of holding past information. With the market data and news data, we implement 100+ features - the same as LightGBM and CatBoost above - and keep the same train-test split ratio. By applying a look back window of 90 days and look back step of 10 days, we build a time-series of historical data for each asset. In that case we got the input matrix. For implementation in keras, we demonstrate model sequence layers in 1. The first three layers are basic LSTM layers with 128, 64, 32 units. We then put in a dropout layer with a 0.5 dropout ratio for regularization. We use a sigmoid function in the dense layer as it is a binary prediction. A SGD optimizer is used with a learning rate of 0.1, momentum of 0.9, and decay of 0.0. We use a 'binary cross entropy' loss function for training. In the final prediction phase, the custom evaluation function mentioned above is applied.

| Layers(type) | Output Shape | Parameter |
|---|---|---|
| Layer1(LSTM) | (None, None, 128) | 116736 |
| Layer2(LSTM) | (None, None, 64) | 49408 |
| Layer3(LSTM) | (None, 32) | 12416 |
| Layer4(Dropout) | (None, 32) | 0 |
| Layer5(dense) | (None, 1) | 33 |

Table 1. Model Structure

And the prediction result is shown in 18.

## 5. Results

We create a benchmark to effectively evaluate the performance of our models. The benchmark consists of a list of ones such that it's equivalent to buying all assets available in the market on each day.

We then divide our data into 70% training data, 20% validation data and 10% test data. The models are first trained using the training data. They are then evaluated against the validation data for hyper-parameter tuning. And the test data is reserved exclusively for final performance evaluation. The results are shown in Table 2.

| Model | Score on test data |
|---|---|
| Benchmark | 0.016 |
| CatBoost Classifier w/o News Data | 0.630 |
| CatBoost Classifier | 0.850 |
| LightGBM Regressor | 0.916 |
| LSTM | 0.402 |

Table 2. Model Performance

## 6. Conclusion

In this analysis, we worked with a large uncensored dataset in an attempt to uncover the predictive power of market and news analytics. One challenge is that we had to plow through the data carefully, weed out data errors and impute missing values. Another challenge is the computational burden as the calculation wouldn't easily fit into the memory of a VM with 32GBs of memory. We had to carefully slice the data, recycle variables and collect unused space to avoid memory errors. For a memory hungry model like LSTM, both training and inference had to be done in batches.

The results are encouraging as they show great improvements over the benchmark. Features from market data such as previous returns and prices have more significance in the models, contributing almost two thirds of the predictive power of the model. Features from news data further enhance the performance by about 35%. As for model selections, gradient boosted trees show better performance than neural network. This could be due to overfitting or suboptimal look back window in the LSTM model. Tuning the LSTM model is beyond the scope of this project. Rather than picking the best model, we think data preprocessing

and feature engineering play a bigger role in determining the final performance. Some extreme outliers could have compromised the out-of-sample performance of the models if they hadn't been treated with caution. We conclude that news data indeed add value to stock performance predictions and in conjunction with market data, it shows great improvement over the benchmark.

## References


[1] Andres Garcia-Medina et al. "Correlations and Flow of Information between The New York Times and Stock Markets". In: *arXiv:1707.05778* (2017).

[2] Kalyani Joshi, Bharathi H. N, and Jyothi Rao. "Stock Trend Prediction Using News Sentiment Analysis". In: *arXiv:1607.01958* (2016).

[3] Kaggle. "Two Sigma: Using News to Predict Stock Movements". In: *Featured Code Competition* (2018).

[4] Guolin Ke et al. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree". In: *Advances in Neural Information Processing Systems 30 (NIPS 2017)* (2017).

[5] Huicheng Liu. "Leveraging Financial News for Stock Trend Prediction with Attention-Based Recurrent Neural Network". In: *arXiv:1811.06173* (2018).

[6] Liudmila Prokhorenkova et al. "CatBoost: unbiased boosting with categorical features". In: *arXiv:1706.09516v4* (2018).

## Appendix

| Item | Description |
|---|---|
| time (datetime64[ns, UTC]) | The current time (in marketdata, all rows are taken at 22:00 UTC). |
| asset Code (object) | A unique id of an asset. |
| asset Name (category) | The name that corresponds to a group of asset Codes. These may be ”Unknown” if the corresponding assetCode does not have any rows in the news data. |
| universe (float64) | A boolean indicating whether or not the instrument on that day will be included in scoring. This value is not provided outside of the training data time period. The trading universe on a given date is the set of instruments that are available for trading (the scoring function will not consider instruments that are not in the trading universe). The trading universe changes daily. |
| volume (float64) | Trading volume in shares for the day. |
| close (float64) | The close price for the day (not adjusted for splits or dividends). |
| open (float64) | The open price for the day (not adjusted for splits or dividends). |
| returns Close Prev Raw 1 (float64) | See returns explanation above. |
| returns Open Prev Raw 1 (float64) | See returns explanation above. |
| returns Close Prev Mktres 1 (float64) | See returns explanation above. |
| returns Open Prev Mktres 1 (float64) | See returns explanation above. |
| returns Close Prev Raw 10 (float64) | See returns explanation above. |
| returns Open Prev Raw 10 (float64) | See returns explanation above. |
| returns Close Prev Mktres 10 (float64) | See returns explanation above. |
| returns Open Prev Mktres 10 (float64) | See returns explanation above. |
| returns Open Next Mktres 10 (float64) | 10 day, market-residualized return. This is the target variable used in competition scoring. The market data has been filtered such that returnsOpenNextMktres10 is always not null. |

Table 3. Market Data Fields

| Item | Description |
|---|---|
| time (datetime64[ns, UTC]) | UTC timestamp showing when the data was available on the feed (second precision) |
| source Timestamp (datetime64[ns, UTC]) | UTC timestamp of this news item when it was created |
| first Created (datetime64[ns, UTC]) | UTC timestamp for the first version of the item |
| source Id (object) | An Id for each news item |
| headline (object) | The item's headline |
| urgency (int8) | Differentiates story types (1: alert, 3: article) |
| take Sequence (int16) | The take sequence number of the news item, starting at 1. For a given story, alerts and articles have separate sequences. |
| provider (category) | Identifier for the organization which provided the news item (e.g. RTRS for Reuters News, BSW for Business Wire) |
| subjects (category) | Topic codes and company identifiers that relate to this news item. Topic codes describe the news item's subject matter. These can cover asset classes, geographies, events, industries/sectors, and other types. |
| audiences (category) | Identifies which desktop news product(s) the news item belongs to. They are typically tailored to specific audiences. (e.g. "M" for Money International News Service and "FB" for French General News Service) |
| body Size (int32) | The size of the current version of the story body in characters |
| company Count (int8) | The number of companies explicitly listed in the news item in the subjects field |
| headline Tag (object) | The Thomson Reuters headline tag for the news item |
| market Commentary (bool) | Boolean indicator that the item is discussing general market conditions, such as "After the Bell" summaries |
| sentence Count (int16) | The total number of sentences in the news item. Can be used in conjunction with firstMentionSentence to determine the relative position of the first mention in the item. |
| word Count (int32) | The total number of lexical tokens (words and punctuation) in the news item assetCodes(category) - list of assets mentioned in the item assetName(category) - name of the asset |
| first Mention Sentence (int16) | The first sentence, starting with the headline, in which the scored asset is mentioned. 1: headline 2: first sentence of the story body 3: second sentence of the body, etc 0: the asset being scored was not found in the news item's headline or body text. As a result, the entire news item's text (headline + body) will be used to determine the sentiment score. |
| relevance (float32) | A decimal number indicating the relevance of the news item to the asset. It ranges from 0 to 1. If the asset is mentioned in the headline, the relevance is set to 1. When the item is an alert (urgency == 1), relevance should be gauged by firstMentionSentence instead. |
| sentiment Class (int8) | Indicates the predominant sentiment class for this news item with respect to the asset. The indicated class is the one with the highest probability. |
| sentiment Negative (float32) | Probability that the sentiment of the news item was negative for the asset |
| sentiment Neutral (float32) | Probability that the sentiment of the news item was neutral for the asset |
| sentiment Positive (float32) | Probability that the sentiment of the news item was positive for the asset |
| sentiment Word Count (int32) | The number of lexical tokens in the sections of the item text that are deemed relevant to the asset. This can be used in conjunction with wordCount to determine the proportion of the news item discussing the asset. |
| novelty Count 12H (int16) | The 12 hour novelty of the content within a news item on a particular asset. It is calculated by comparing it with the asset-specific text over a cache of previous news items that contain the asset. |
| novelty Count 24H (int16) | Same as above, but for 24 hours |
| novelty Count 3D (int16) | Same as above, but for 3 days |
| novelty Count 5D (int16) | Same as above, but for 5 days |
| novelty Count 7D (int16) | Same as above, but for 7 days |
| volume Counts 12H (int16) | The 12 hour volume of news for each asset. A cache of previous news items is maintained and the number of news items that mention the asset within each of five historical periods is calculated. |
| volume Counts 24H (int16) | Same as above, but for 24 hours |
| volume Counts 3D (int16) | Same as above, but for 3 days |
| volume Counts 5D (int16) | Same as above, but for 5 days |
| volume Counts 7D (int16) | Same as above, but for 7 days |

Table 4. News Data Fields

| Name | Count |
|---|---|
| Citigroup Inc | 30823 |
| JPMorgan Chase & Co | 29129 |
| Bank of America Corp | 28197 |
| Apple Inc | 26702 |
| Goldman Sachs Group Inc | 25044 |

Table 5. Top 5 Companies With Negative Sentiment

| Name | Count |
|---|---|
| Barclays PLC | 24898 |
| HSBC Holdings PLC | 23191 |
| Deutsche Bank AG | 20702 |
| BHP Billiton PLC | 18019 |
| Rio Tinto PLC | 16782 |

Table 6. Top 5 Companies With Neutral Sentiment

| Name | Count |
|---|---|
| Barclays PLC | 22855 |
| Apple Inc | 22770 |
| General Electric Co | 20055 |
| Royal Dutch Shell PLC | 18206 |
| Citigroup Inc | 18025 |

Table 7. Top 5 Companies With Positive Sentiment

| Item | Name |
|---|---|
| Data Preprocessing (gradient boosted trees) | Fan Gao |
| Data Preprocessing (LSTM) | Ruisi Wang |
| Feature Engineering (gradient boosted trees) | Fan Gao |
| Feature Engineering (LSTM) | Ruisi Wang |
| Visualization (website) | Connor Gatlin |
| Modeling - LightGBM | Fan Gao |
| Modeling - CatBoost | Connor Gatlin |
| Modeling - LSTM | Ruisi Wang |
| Presentation Slides | Connor Gatlin |
| Youtube Videos | Connor Gatlin |
| Report | Fan Gao |

Table 8. Individual Contribution