Predicting Stock Movement

Jad Rayes

Abstract—An essential aspect of the utility of news in financial markets, is the ability to use the content of news analytics to predict stock price performance. By analyzing news data to predict stock prices, we can harness the predictive power of current events to help predict financial outcomes and generate a global economic impact. In this paper we present an approach to predict stock movement using tree based classifiers. We use market data provided by Intrinio and news data provided by Thompson Reuters to predict the direction that stock over the next ten days will move. Our approach performs competitively in the Two Sigma hosted "Predicting Stock Movements" competition on Kaggle, scoring in the top 11% of 2500 participants.

1 Introduction and Motivation

An interesting dimension of research, that has not been utilized as effectively in other systems, is the use of news content to predict stock movement or trading patterns. Developing an interpretable model that can effectively utilize heterogeneous types of financial news in conjunction with market data to predict stock movement, has a number of useful applications. As in [2], such systems could be used to help regulate financial markets, and could also act as recommender systems for stock trading. For instance, given some market and news data on a day, you could alert a user to certain opportune times to sell and purchase stock. In [4], it is shown how a multi-instance learner is able to combine a number of different indicators from heterogeneous news and market data to outperform state-of-the-art models in stock movement prediction. Their system is then shown to be effective in forecasting market changes and can be usefully applied to economic research tasks, or commercialized to serve as a trading recommender. Several papers have been released that involve the use of technical indicators – such as the "RSI"(Relative Strength Index) and "SO" (Stochastic Oscillator) – to predict stock movement effectively [1]. We build upon the insights of these papers and use their findings by supplementing the data from Intrinio with specific market indicators.

We use gradient boosted decision trees to predict values that indicate our confidence that the stock will move in a particular direction. We compare two different decision trees for this task, using XGBoost and LightGBM. We show that LightGBM outperforms XGBoost on the classification task and that our model achieves competitive results, using our leaderboard ranking as a baseline.

Our goal is to take market and news data, and predict the likelihood of a positive or negative return over the next ten days. This is a binary classification problem, where the class label 1 signifies an increase in the return of a particular stock over the next ten days and 0 signifies that it will not increase. A major goal of this model is to provide a high degree of interpretability by using quantitative measures – such as how often a given feature is used in the model during prediction time, or the entropy of a split using a particular feature – to understand what factors affect stock movement. Our approach also differs from most others in the literature because we supplement both our market data with technical indicators, discussed in 3, and our news data with tf-idf related attributes (3.1.3) to assess the impact of headlines on price fluctuation.

2 Related Work

I will briefly discuss work that uses news analytics to both predict stock movement, and regulate markets by detecting illegal insider trading. I will also explain where my work fits in.

2.1 Predicting Stock Movement using Tree based Classifiers

Basak et. al invent an approach to stock movement prediction, using gradient boosted decision trees to predict whether the stock price will increase or decrease based upon the price for the past n days [1]. Their approach differs from most others because they use a number of technical indicators that measure certain properties of the stock, such as it's return on investment and how much momentum the stock has gained over time. I incorporate the use of technical indicators into my model by supplementing the market data provided by Kaggle with indicators that we discuss in 3. Our approach differs from this one because we incorporate news data into our feature space. We also demonstrate in 4.3 which of these indicators are most significant in stock movement classification.

2.2 Predicting Price and Index Movement using Trend Deterministic Data

In their paper, [3] Patel et. al demonstrate the efficacy of trend deterministic indicators in market data. They show how considering the market data with the use of indicators that are highly "trend-based" improves prediction accuracy. They also evaluate the performance of four different classifiers, ANN(Artificial Neural Network), Naive Bayes, SVM, and Random Forest, and find that Random Forest achieves the best prediction accuracy. The idea of using "trend-based" indicators such as MACD, or EWMA are adopted heavily by our paper. We also find that Patel's results confirm that tree-based classifiers are especially good models for stock movement classification. We extend the work done by Patel, by showing precisely how indicators contribute to a classification model, and how news and market data can be used together to bolster prediction accuracy.

2.3 Stock Market Prediction via Multi-Source Multiple Instance Learning

Some recent work on predicting stock movement with news content has involved the use of multiple instance learning. Multiple instance learning is a type of supervised learning where instead of receiving a set of instances which are individually labeled, the learner receives a set of labeled bags, each containing many instances. A bag may be labeled negative if all the instances in it are negative. On the other hand, a bag is labeled positive if there is at least one instance in it which is positive. The task of the labeller is to try and label either the bags, with multiple instances, or the instances themselves. The task of the paper in [4] is to use indicators that come from multiple sources to predict stock market composite index movements. They use a multiple instance model to combine news events, sentiments and market data into a comprehensive framework. Essentially treating a "bag" as a collection of sentiment, event and market data, in order to predict stock movements 14. The key difference between the approach of this paper, and my work, is the use of multiple sources to build the model. A direction of future work might be to try and incorporate different news and market indicators, by placing them into labelled bags and training a Multi-instance learner on those bags.

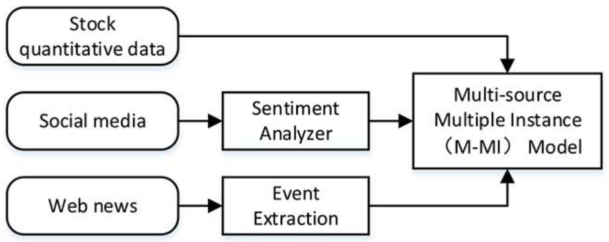

Figure 1. Overview of the model in [4]

3 Methodology

We perform a considerable amount of feature engineering to try and understand both our market and news features. Summarized below (3.1) is the full size of the training data we are given by Kaggle. We begin by trying to find outliers and price points which may serve as noise to the model's input. For example, Two Sigma claims that they have purposefully injected artifical data into the feature space to make engineering resilient models more difficult, and also to force us to look more closely at the behavior of the market over time. Thus in 3.1.2 we plot both news and market data over time to try and observe any patterns in the frequency of news or the behavior of price. We have 16 features for market, and 35 features for news data. We add an additional six technical indicators to our market data for 22 features, and add an additional two features to our news data for 37 features per sample. We join the market to the news data with a total of 3,339,381 rows. These are less than the original dataset, because as we will show in 3.1.2 we remove outliers and noise, and also only consider market and news data from mid 2009 onward.

3.1 Preprocessing and feature extraction

News Data samples	Market Data samples	Joined
4,072,956	9,328,750	3,339,381

3.1.1 Source of data

For this project, I will be using Two Sigma's Kaggle hosted dataset. The dataset consists of two components: market and news data. In the market data are basic metrics, such as opening and closing price, volume of shares traded, and the name of stock. In the news data we have a large number of features, including the headline of an article, what stocks it discusses, when it was published, and its sentiment. A complete description of our features are given in 6. Note that the features which are bold, are those which I have added to the data, and were not supplied to me. I describe in greater detail how I supplemented the market and news data below.

3.1.2 Market data

We have millions of price points for all stocks in the competition, but we have by proportion only a very small percentage of outliers. Statistically, outliers can be quite meaningful, but our goal is to generate as resilient and robust a model as possible. From the following time series plots (2 and 3), we observe that the market behaves quite homogeneously from mid 2009 onward. We know that because of the financial crisis in 2008, there was a large fluctuation in price points for stock across sectors. If our model considered only the data from mid-2009 to present, it would be able to follow this relative stability in market trends more easily. If, we add more radical fluctuations in pricing, by including data from 2008, we could force our model to over or underestimate confidence values. This is of course assumes, that the market will continue to behave as it has been (there won't be an impending financial crisis). In addition, Two Sigma has randomly injected noise into the feature space by including either values that don't make sense, or stocks (asset codes) associated with prices that are unknown; So it helps to perform some extensive cleaning before we try to engineer new features. The plot in 3 shows how the market data behaves more consistently from 2009 onwards. The colored lines represent the different quantiles in market return over a ten day window. For example, the pink line is the 95th percentile in returns, which means that the returns of the market as a whole on these days were higher than 95% of all other values. What we can observe from this plot, is that besides the trends themselves, the really oscillating return fluctuations happen in the 2008-2009 period. The following table summarizes how many outliers we remove from the data. We remove any values, in any feature, that are higher than 98% of all others, or less than 2% of all others – the 98th and 2nd percentiles respectively.

Since only 21 different entries had a 10 day return that was positive or negative of over 50%, we can consider any returns at or above this to be outliers, with respect to the millions of records. As an important point, we treat outliers as artificial noise (stemming from inaccurate measurement), and instead of removing them, we replace them with values in a more reasonable range.

Below we categorize our outliers to show how much data was removed for what reason.

Abnormally high return	Assets with no pricing data
211	531

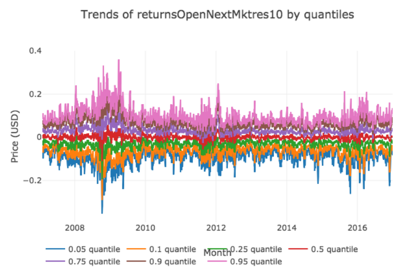

Figure 2. 2008-

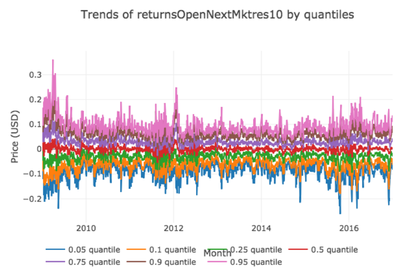

Figure 3. Post 2009

3.1.3 News data

We plot the news data in 4, and interestingly observe that the amount of news released is always greater during the fiscal quarters. This information could help us organize price points by quarter, and see if there is any sharper increase or decrease in return by quarter. Additionally, we vectorize the headlines of each news article, and use a logistic regression model to try and predict market return, given only the headline. We map the output of the model to a probability value by using a sigmoid function:

$$h_\theta(x) = \frac{1}{1 + e^- \theta^T x}$$

θ are the learned weights for each headline and x is the feature vector for each sample. Note that the inner product

$$\theta^T x$$

can be any real number, which will then be mapped to a number in the reals between 0 and 1 – hence a probability. This probability, which may be denoted as $P(S|H)$, is the probability that the 10 day return, S, increased given a particular news headline H.

We also use another metric to compare headlines in terms of relative rarity. For example, if two headlines contain several rare words (not necessarily the same words) then they might be articles describing a less frequent occurrence, and could have a useful baring on the stock price. In order to capture this property with a single number, we can compute the tf-idf vectors and take the mean of all elements (see Figure 6). This won't give us similarity between headlines, but instead can tell us which headlines contain rarer words, since headlines with rarer words will have higher weights. Since most headlines contain only a few words, this is reasonable. As a general strategy, this is not advisable. The wordcloud below shows the top 10,000 words across all headlines, larger words are most common, with lower tf-idf weights, whereas smaller ones are less common, with higher weights (see 6).

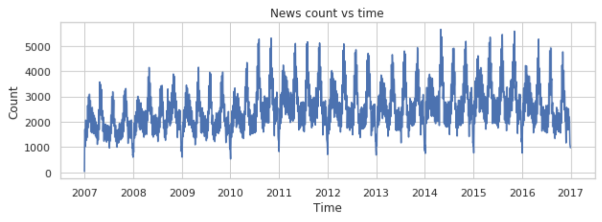

Figure 4. The count of news articles over time

$$v_t = \text{idf}(t) = log \frac{1 + n_d}{1 + \text{df}(d,t)} + 1$$

$$tfscore = \sum_{t=1}^{n} \frac{v_t}{n * \sqrt{v_{1t}^2 + v_{1t+1}^2 + \cdots + v_{1n}^2}}$$

Figure 5. The mean of all tf-idf vector components for a headline

Figure 6. A wordcloud of 10,000 news headlines

3.1.4 Market Indicators

Market indicators are used successfully in the literature to predict stock movement. Using market indicators bolstered the accuracy of the model more than most other features according to 4.3. A useful exercise in understanding the utility of indicators can be to take the RSI (Relative Strength Index) for example. Stocks may be over-bought which could drive up metrics like the opening price. Based only upon these metrics, one may wrongfully conclude that the price will continue to rise based upon a streak of high opening prices. However, since the RSI takes into account that demand may unjustly raise the price of stock higher than it's actual value, a low RSI value will be able to predict that the stock will price will soon fall. Therefore, using market indicators allows us to distinguish between external factors which may be driving the price of a stock up, and the actual market value. This will allow us to make more stable predictions in the long-term.

Noting the successes in both [3] and [1], we use the following trend based indicators.

- EWMA
- Average Close
- Average Volume
- Std close/volume
- Centered Average

The exponentially weighted moving average is a reliable "trend-based" indicator. It is less sensitive to outliers (infrequently high price points), and provides an indication of the performance of stock over a period of n days. Let S_0 be the first term in the series:

$$S_0 = Y_0$$

Then the new price on day t is added to the moving average with weight α, and $0 \leq \alpha \leq 1$

$$S_t = \alpha * Y_t + (1 - \alpha) * S_{t-1}$$

The effect of this is to keep the average smooth, and only many observations will be able to force a change in the average. This average is resilient to outliers, such as single day high's in stock price.

The average close, average volume and the standard deviations of the two, give us an idea of how volatile the trend is and what the expected values of closing price or volume would be for a given asset. Finally, the centered average gives a more recent window with which to make a reasonable guess about the price. If the window is too small, it is unreliable, but the if the window is large enough – in our case, 7 - 10 days – it can often be a reasonable indicator of the next day's price. Below we show a plot of a single stock (asset's) indicators.

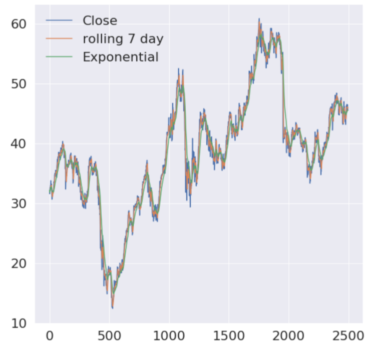

Figure 7. Graph of trend based indicators for a particular asset

3.2 Model building

3.2.1 *Gradient Boosting Machines*

Gradient Boosting Machines (GBMs) are ensemble learners that work by training many individual learners which are then combined into a single strong learner. In GBM models which are tree-based, such as LightGBM or XGBoost, the trees are trained sequentially with each tree learning from the mistakes of the previous ones. The hundreds or thousands of weak learners are combined to make a single strong ensemble learner with the contributions of each individual learned during training using Stochastic Gradient Descent. Tree-based classifiers are used in [3] and [1] successfully, and are quite successfully utilized here as well. What makes GBM's a good choice here is that we are dealing with a large number of training samples, so we can control over-fitting by using early stopping, and we also have structured data – our features are engineered and we can understand them easily by looking at a spreadsheet.

To get a sense of the intuition behind gradient boosted models we refer to the below image. In the first square on the left, a single learner(Box 1) makes a prediction, drawing it's decision region as a vertical line. Although it manages to distinguish some "+" from "-", it misclassifies three training examples. So, the next weak learner (Box 2) will penalize the misclassified samples with a higher weight (as is done in gradient descent) to ensure it classifies them correctly. In doing this, it draws a vertical line on the other side of the square, and misclassifies three "-". The third learner (Box 3) will proceed in the same way as the second and gives the misclassified "-" labels a higher weight. However, it too misclassifies two samples. Finally, we see that the strong learner (Box 4) combines each of the three learners with some weights to produce the final non-linear decision region.

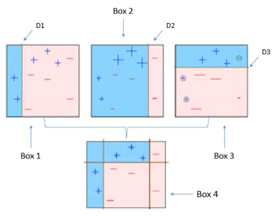

Figure 8. Shows how weak learners are progressively added to improve the loss function

3.3 Prediction workflow

We will first formalize this problem as a classification one. We takes as input to the model a feature vector x. The goal is to map $x \in \mathbb{R}^d$ to the set $\{0, 1\}$. Formally, it is a map f s.t. $\forall x \in \mathbb{R}^d$ $f(x) \mapsto \{0, 1\}$. Here the 0 signifies that the price of a particular stock will not increase, and the 1 signifies that it will.

Note that because we are trying to map binary class labels to real valued outputs – which will serve as our confidence values – we will have to be a bit clever in how we do this. The initial strategy is to simply take the difference of the predicted probability of each class, for each sample

$$P(y = 1|x) - P(y = 0|x)$$

Note that if they are equal, because the classifier is unsure whether the stock will go up or not, the confidence will be 0. If we are more certain that it will go down instead of up, then this difference will be negative, and hence we will have a negative confidence, resembling a negative return. Since we aim to minimize a binary logloss classification objective, in order to map class labels to probaility values we will use the sigmoid function described in 3.1.3.

3.4 Comparing performance

We compared two tree-based classifiers for this task. XGBoost had a higher training time than LightGBM, and after cross validation was performed on both models, we plotted the validation score computed by 4.1.

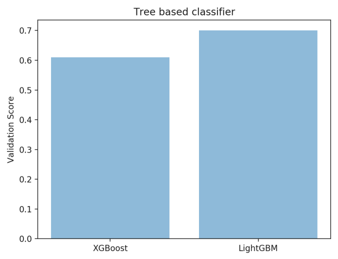

Figure 9. Comparing the performance of LightGBM and XGBoost on the test set

3.5 Cross Validation

We use stacked cross validation. We first use 5-fold "Randomized CV" to find the best range of hyper-parameters across the hyper-parameter space. We then use an exhaustive search – "Grid CV" – on the smaller range of hyper-parameters found by the randomized CV search. This process is more efficient in terms of both time and space complexity, since gradient boosted decisions trees have so many hyper-parameters to tune. The best hyper-parameters using stacked cross validation for a single LightGBM Classifier were found to be:

- number of estimators : 500
- boosting type: dart
- objective: binary
- number of leaves: 2,452
- minimum number of child samples: 212
- regularization term to prevent over-fitting: 0.01

A brief explanation on each of these is in order. Firstly, the number of estimators is simply the number of weak learners (trees) to combine to make a single strong learner (tree). The boosting type "dart" essentially picks certain trees to drop during boosting rounds, which can prevent overfitting. The number of leaves are the maximum number of leaf nodes which can exist for a single tree – not all trees! Finally, the minimum number of child samples is simply the least number of labels which must exist at any one leaf, one can think of it as the minimum leaf size.

4 Results

4.1 Evaluation metrics

The goal is to predict a signed confidence value:

$$\hat{y}_{ti} \in [-1, 1]$$

that is multiplied by the market-adjusted return of a particular stock over a ten day window.

For instance, if stock is considered to have a large increase in price–compared to the broad market–over the next ten days, a positive confidence value (near 1.0) might be appropriate. Negative confidence values (near -1.0) indicate a large decrease in price. Values near 0, indicate uncertainty in the prediction.

For each day in the evaluation time period:

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

$$\hat{y}_{ti}$$

The final score is computed as:

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

Our score on the test set, is currently 70% by the evaluation metric above. The highest public score on Kaggle is 85%, and we are at 69.254% (position 281) ranking in the top 11% of 2500 participants. We used just a single LGBM classifier (no ensemble) and the optimal hyper-paramters found in 3.5 through cross validation. As we observed, an ensemble of learners may further improve accuracy, we also show an improvement in f1 across ensemble learners 10. We are still trying to find the optimal number of learners, and have not yet found the best public score using an ensemble of learners of size n with a voting method.

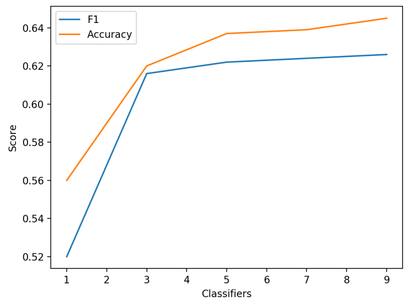

Figure 10. Increase in accuracy and f1 score as the ensemble size increases

4.2 Model Performance

We observe a decent baseline using a single LGBM classifier, with the optimal hyper-parameters found in 3.5. Below we show how our predicted confidence deviates from the actual market return (11 and 12), by plotting two histograms. We can read the graph as a distribution. For example, the mean number (most stocks) have a zero return. We can also note that since the distribution of gains are rarer than loses, or neutral values in a stock's return, we may want to use an f1 scoring metric to see how well we do at classifying increases in return for a given stock (see 4.1).

Note that the score being used to compute these numbers is mentioned in 4.1, and it is the official competition evaluation metric.

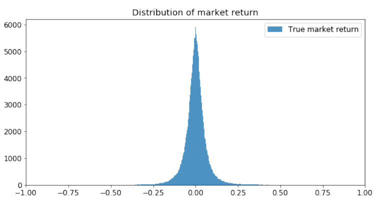

Figure 11. Graph showing the actual market return

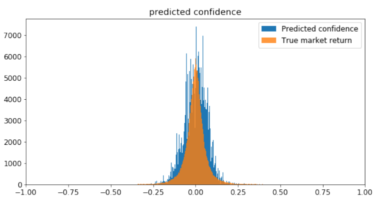

Figure 12. Graph of our predicted values against the market return

4.3 Feature Significance

In order to assess the importance of features to our model we use two main criteria. The first will be the number of times each feature is used in the model. The second is the entropy, or information gain of each feature used in the model. In other words, we measure the gini-index of each split using feature f, totalling all of them over all splits. In both cases, the market indicators are significant. They are both used the most frequently, and each split using them has a high entropy. This likely indicates that they are most responsible for the model's prediction accuracy, and if we were to remove them, we would face a reduction in accuracy.

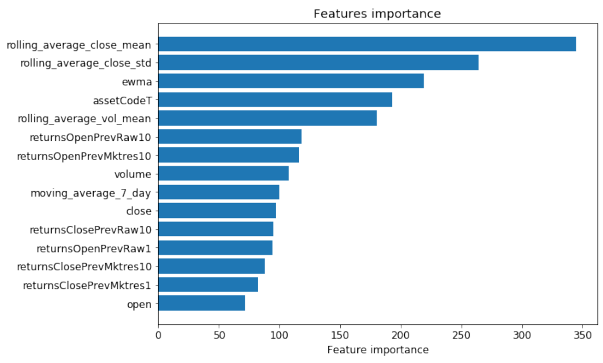

Figure 13. The importance of our features with a split criterion

In both cases, we can see that the market indicators we've added are most significant. They outdo most of the news and other market data in terms of both splits and entropy.

5 Conclusions and Future Work

We have demonstrated that a reasonable baseline can be achieved for predicting stock movement with a fairly basic methodology. Certain things however are lacking. Firstly, we have not explored in great depth the types of technical indicators which should supplement the market data. We know that the indicators we have used: EWMA, Average close/volume, standard deviation of close/volume, and a moving week to ten day average of closing price, are useful and contribute strongly to our models prediction accuracy. However, we do not know if they are the best indicators to use among all others. In addition, we should further explore the idea of an ensemble of LightGBM classifiers (4.1) with different voting techniques. We should try to determine an optimal number of learners. Thus, there remains some open questions about feature engineering and model selection. We have established a good baseline, our future work will aim to improve it.

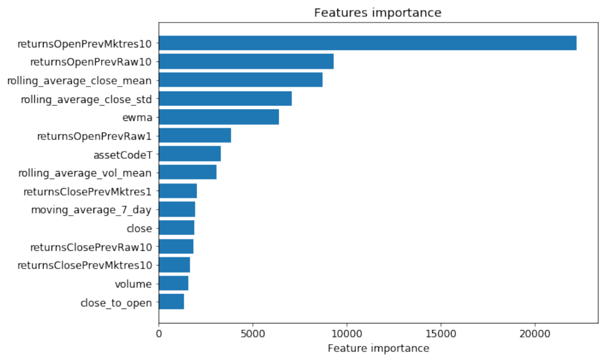

Figure 14. The importance of our features with an entropy criterion

References

[1] Suryoday Basak, Saibal Kar, Snehanshu Saha, Luckyson Khaidem, and Sudeepa Roy Dey. Predicting the direction of stock market prices using tree-based classifiers. *The North American Journal of Economics and Finance*, 2018.

[2] Henry G Goldberg, J Dale Kirkland, Dennis Lee, Ping Shyr, and Dipak Thakker. The nasd securities observation, new analysis and regulation system (sonar). In *IAAI*, pages 11–18, 2003.

[3] Jigar Patel, Sahil Shah, Priyank Thakkar, and K Kotecha. Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Systems with Applications*, 42(1):259–268, 2015.

[4] Xi Zhang, Siyu Qu, Jieyun Huang, Binxing Fang, and Philip Yu. Stock market prediction via multi-source multiple instance learning. *IEEE Access*, 2018.

6 Appendix

time	Current time of record
assetCode	Code for a stock: Unique stock ID
assetName	Name of stock
universe	Whether stock should be included in scoring on a given day, see 4.1
volume	Volume of shares traded
close	Closing price on the day for a given stock
open	Open price on the day for a given stock
returnsClosePrevRaw1	Previous day raw return of stock, calculated from the market close
returnsOpenPrevRaw1	Previous day raw return of stock, calculated from the market open
returnsClosePrevMktres1	Same as above but market residualized
returnsOpenPrevMktres1	Same as above but market residualized
returnsClosePrevRaw10	Ten day window instead of 1
returnsOpenPrevRaw10	Ten day window instead of 1
returnsOpenPrevMktres10	Same as above but market residualized
returnsOpenNextMktres10	Same as above but market residualized
returnsOpenNextMktres10	**Target variable: the next ten day market residualized return**
Average_Close	Average closing price for an asset
Std_Close	Standard deviation of closing price for an asset
Average_Vol	Average volume for an asset
Std_Vol	Standard deviation of volume for an asset
Rolling_Avg_7	Seven day rolling average for an asset
EWMA	Exponentially weighted moving average for closing price of an asset

time	Current time when data was first available
sourceTimestamp	UTC timestamp of when news item was created
firstCreated	UTC timestamp for the first version of the item
sourceId	An Id for each news item
headline	Items Headline
urgency	Type of news on scale of 1 - 3 (1: alert - 3 : article)
takeSequence	Sequence number of a given article, i.e. if article appears as second in series of articles
provider	Provider of the news item
subjects	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	How to categorize the articles in terms of audiences, i.e. financial audiences, general news
bodySize	the size of the current version of the story body in characters
companyCount	Count of companies mentioned in news item
headlineTag	the Thomson Reuters headline tag for the news item
marketCommentary	boolean indicator that the item is discussing general market conditions
sentenceCount	the total number of sentences in the news item.
wordCount	the total number of lexical tokens (words and punctuation) in the news item
assetCodes	list of assets mentioned in the item
assetName	name of the asset
firstMentionSentence	the first sentence, starting with the headline, in which the scored asset is mentioned.
relevance	a decimal number indicating the relevance of the news item to the asset.
sentimentClass	indicates the predominant sentiment class(pos, neg, neutral) for this news item with respect to the asset. The indicated class is the one with the highest probability.
sentimentNegative	probability that the sentiment of the news item was negative for the asset
sentimentNeutral	probability that the sentiment of the news item was neutral for the asset
sentimentPositive	probability that the sentiment of the news item was positive for the asset
sentimentWordCount	the number of tokens in the sections of the item text that are relevant to the asset.
noveltyCount12H	The 12 hour novelty of the content within a news item on a particular asset.
noveltyCount24H	same as above, but for 24 hours
noveltyCount3D	same as above, but for 3 days
noveltyCount5D	same as above, but for 5 days
noveltyCount7D	same as above, but for 7 days
volumeCounts12H	the 12 hour volume of news for each asset.
volumeCounts24H	same as above, but for 24 hours
volumeCounts3D	same as above, but for 3 days
volumeCounts5D	same as above, but for 5 days
volumeCounts7D	same as above, but for 7 days
HeadlineCoefficient	The probability that the stock will increase given a headline: formally, $P(S \mid H)$
TF_Score	The mean tf-idf score for a headline