MSBD 5001 Final Project: Using News to Predict Stock Movements

Haojun Chen Qian Chen Zihao Peng Cong Wang Shen Wang
The Hong Kong University of Science and Technology

Abstract
This project is involved in using news to predict stock movements via multiple machine learning approaches. The goal is to utilize the content of news analytics to predict stock price performance. In this way, we will be able to provide more professional and sharp insights for investors to make better investment decisions. To achieve our goal, we will implement integrated data preprocessing and multiple machine learning algorithms to achieve a model with higher accuracy.

1. Introduction & Motivations

Machine learning has had fruitful applications in finance well before the advent of mobile banking apps, proficient chat bots, or search engines. Due to the given high volume, accurate historical records and quantitative nature of the finance world, it is also better suited for our objective. Thus, we chose a financial problem as our main topic and believe that it will be more useful, interesting, as well as challengeable.

Nowadays, the ubiquity of data enables investors at any scale to make better investment decisions. However, distinguishing the useful data and the noise in this sea of information could be very hard, especially surrounded by news from all over the world every day. Our target is to overcome this difficulty so that we will be able to help investors more easily and accurately discover key information in the market, then use the information to predict stock movements in a more reliable and scientific way.

Meanwhile, by analyzing news data to predict stock prices, we will get a deeper insight on applying machine learning algorithms to real life problems and contribute to the advancement of research in understanding the predictive power of news. In essence, we hope to create significant economic impact worldwide.

2. Data Overview

There are two separate datasets: market data with 16 features provided by Intrinio, and news data with 35 features provided by Thomson Reuters. All together there are 4,072,956 rows of data for the market dataset and 9,328,750 rows of data for the news dataset. The datasets are divided into two parts, 2007/2/1-2016/12/31 is for training, and 2017/1/1-2018/7/31 is for testing.

Market dataset

The market dataset contains financial market information such as opening price, closing price, trading volume, and calculated returns. It has the following columns:

[Time, assetCode, assetName, universe, volume, close, open, returnsClosePrevRaw1, returnsOpenPrevRaw1, returnsClosePrevMktres1, returnsOpenPrevMktres1, returnsClosePrevRaw10, returnsOpenPrevRaw10, returnsClosePrevMktres10, returnsOpenPrevMktres10, returnsOpenNextMktres10]

The returns columns consists of two types:

- Raw, meaning that the data is not adjusted against any benchmark.
- Market-adjusted(Mktres), meaning that the movement of the market as a whole has been accounted for, leaving only movements inherent to the instrument interval.

Return intervals provided here are 1 day and 10 day horizons.

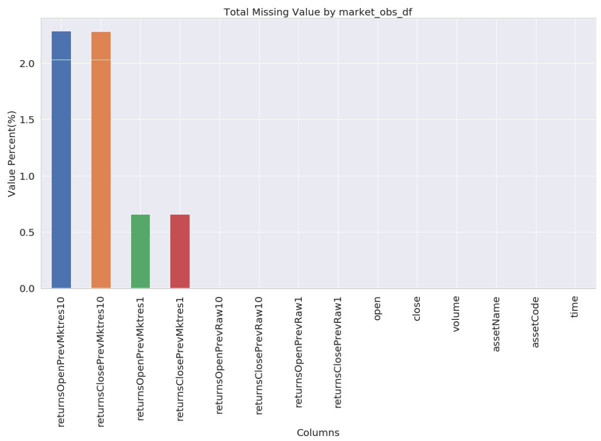
News dataset
The news dataset contains information about news articles/alerts published about assets, such as article details, sentiment, and other commentary. It has the following columns:
[time, sourceTimestamp, firstCreated, sourceId, headline, urgency, takeSequence, provider, subjects, audiences, bodySize, companyCount, headlineTag, marketCommentary, sentenceCount, wordCount, assetCodes, assetName, firstMentionSentence, relevance, sentimentClass, sentimentNegative, sentimentNeutral, sentimentPositive, sentimentWordCount, noveltyCount12H, noveltyCount24H, noveltyCount3D, noveltyCount5D, noveltyCount7D, volumeCounts12H, volumeCounts24H, volumeCounts3D, volumeCounts5D, volumeCounts7D]

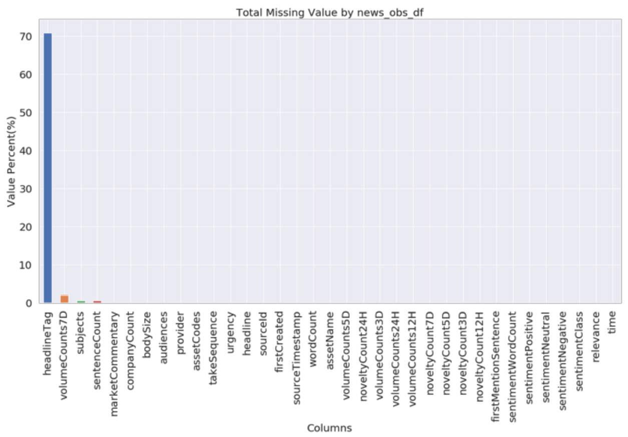
3. Data Preprocessing
Since we used 3 different models for evaluation, there were some modifications to the preprocessing steps for each of the models. However, we used the following preprocessing steps as a base for all our models.

1) Missing Data
There is around 0.5% and 2% of the data missing for different entries. We could just drop them all together, or we can fill in some reasonable values. We need to determine if zero is a legitimate value to fill in for our missing entries.

Since we have two datasets, we will explore them separately. For market data, we have a really interesting dataset which contains stock prices for many companies over a decade. When we have a look at the data itself and not think about the competition, we can see long-term trends such as appearing and declining companies. Higher quantile prices have increased with time and lower quantile prices have decreased. This could be because the rich-poor gap has increased or many smaller companies are ready to go to market so their share prices aren't very high yet.

When we check for missing values, we can see that for the market dataset, most of the missing values lie in the returnsOpenPrevMktres10 and returnsClosePrevmktres10 columns. Since these are all market-adjusted columns, we will fill in the missing values with the raw values from the same row. For the news dataset, most of the missing values lie in the headlineTag column. We will simply leave these values as NaN.

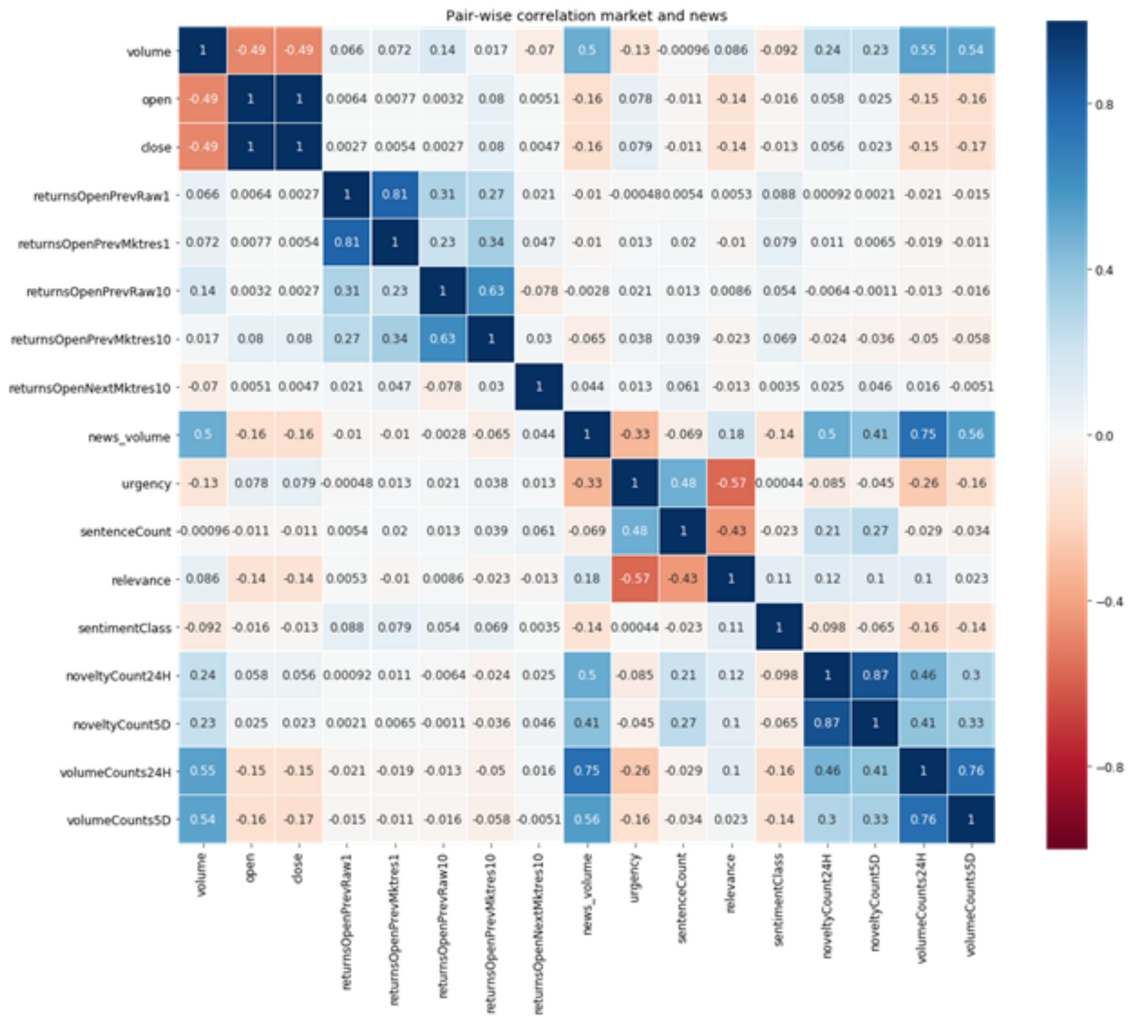




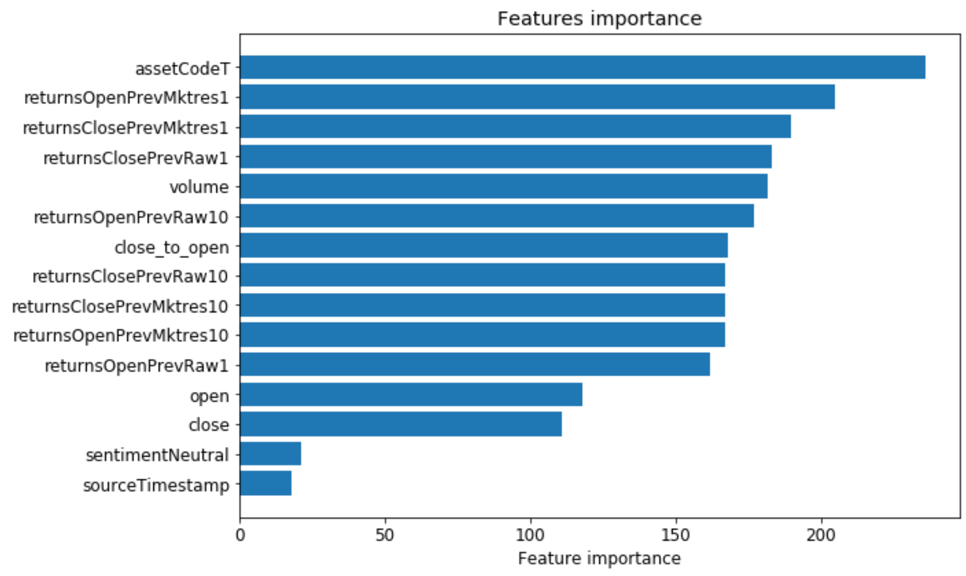
2) Outlier Analysis
In our datasets, we can observe that there are a few outliers present. We will use the interquartile range to remove all the outliers outside of the range. Otherwise, these outliers act as noise that may negatively impact our prediction later on.

For the market data, the difference between open price and close price cannot be too great, otherwise the market would be corrupt. We treat these outliers by clipping the close-to-open ratio.

3) Correlation Analysis
We are concerned with predicting the stock price ten days in the future which is the column returnsOpenNextMktres10. Therefore, we need to find out if there are any obvious correlations. To do this, we combine the market and news datasets and plot the pair-wise correlation.



4) Feature Importance
After we pre-processed the market and news datasets, we merged those data datasets to train a baseline model - LightGBM. We used this baseline model to evaluate the features importance, and chose the top-k best features as the final training features.



5) Feature Normalization
Since the values in the market dataset have highly different ranges, we used scaling to normalize the variables for daily prediction.

4. Modeling
We used three different models: deep neural networks, LSTM, and LightGBM, and compared their results.

1) DNN
Deep neural networks require embedding or encoding of categorical data in the preprocessing stage, and a tuning procedure for hyperparameters of each layer. We used the following structure for our deep neural network model:

Input layer	32 units	
Hidden layer 1	128 units	relu
Hidden layer 2	64 units	relu
Hidden layer 3	64 units	relu
Output layer	1 unit	sigmoid

As part of our preprocessing for DNN, we used an encoder function to encode the categorical variables. For numerical variables, we used the standard scaler in scikit-learn.
For training, we used only the market dataset and split 75% of the data for the training set and 25% of the data for the testing set. Upon evaluation of our test set, we achieved a validation accuracy of 55.8%.

2) LSTM
LSTM networks are a special form of recurrent neural networks that are well-suited to classifying, processing and making predictions based on time series data. We used the following structure for our LSTM model:

LSTM layer	128 units	
LSTM layer	64 units	
LSTM layer	32 units	
Output layer	1 unit	sigmoid

As part of our preprocessing for LSTM, we used one-hot encoding to convert the categorical variables into numerical variables. For numerical variables, we again used the standard scaler.

For training, we combined the market and news datasets and split 75% of the data for the training set and 25% of the data for the test set. Upon evaluation of our test set, we achieved a validation accuracy of 49.7%.

3) LightGBM

LightGBM uses a novel technique of gradient-based one-side sampling to filter out the data instances for finding a split value, and also uses a special algorithm to find the split value of categorical features. Some of its advantages are a faster training speed, lower memory usage, and capability of handling large-scale data. For our hyperparameters, we used cross validation and ended up with the following:

boosting_type	Dart
learning_rate	0.1
n_estimators	500
objective	binary
reg_lambda	0.01

For training, we combined the market and news datasets and split 75% of the data for the training set and 25% of the data for the test set. In addition, we get rid of the extra unnecessary data from news data and combine multiple news reports for the same assets on one specific day. We implemented an ensemble method with multiple classifiers, and after ensemble training and achieved a validation accuracy of 56.5%.

4) Comparison with other models

To select the models that were suitable for our project, we analyzed the advantages and disadvantages of several different models. In addition to the three that we decided on, we also considered XGBoost, CatBoost, and convolutional neural networks.

XGBoost uses a pre-sorted algorithm and a histogram-based algorithm for computing the best split, but only accepts numerical values and cannot handle categorical features by itself. CatBoost uses gradient boosting on decision trees and has the flexibility of giving indices of categorical columns so that it can be encoded using one-hot encoding.

In comparison with the three models that we chose for our project, we found that XGBoost generally worked well, but was too slow for hyperparameter tuning using GridSearchCV. Similarly, CatBoost performs best when there are categorical variables in the dataset. Since there aren't many categorical features that are relevant to our prediction, we decided that it wasn't the most suitable for our project. We disregarded convolutional neural networks as it is mostly used in visual imagery.

Although the final score is not stellar, we do make some improvement from 0.63715 to 0.65518 for DNN model and this has accomplished our goal for this project – 0.60 accuracy score.

5. Contributions

Haojun Chen – data preprocessing, LightGBM model, project report
Qian Chen – data preprocessing, LSTM model
Zihao Peng – data preprocessing, neural network model
Terence Leung – data preprocessing, LightGBM model, project report
Cong Wang – data preprocessing, neural network model, project report

6. References

[1] Daniel Faggella, Machine Learning in Finance – Present and Future Applications. TechEmergence
[2] Alvira Swalin, CatBoost vs. Light GBM vs. XGBoost. Towards Data Science
[3] Jannes Klaas, XGBoost Baseline. Kaggle
[4] Nguyen Dang Minh, Complete EDA + Features Engineering + Voting LightGBM. Kaggle
[5] Dmitry Pukhov, EDA and NN for Market and News. Kaggle

[6] Sergey Kalutsky, LSTM Model on Market Data. Kaggle
[7] Hyun Woo Kim. Two Sigma News: Simple EDA + Prophet + NLP. Kaggle