Stock Market Movement Prediction

Byungsu Jung
Department of Mathematics
bjung@uw.edu

Seoyoung Park
Department of ACMS(Data Sci&Stat)
syp1017@uw.edu

1 Summary of research questions and Results

Question Summary
In a brief sentence, our research is trying several NLP techniques to predict the stock market and get words that perform as important features.
The aim of our study is to roughly predict stock market price movement based on top 25 News headlines for each day. Considering this task involves NLP, we will implemented several NLP feature extraction techniques such as N-gram, TF-IDF and others. For each of feature extraction techniques, we will implement different machine learning models to evaluate which feature extraction showed highest prediction. After successfully identifying best feature extraction technique and machine learning model, we will further provide report on "Why selected feature extraction technique and machine learning model showed best performance", and we will propose future improvements,
Question list

1. Is it possible to roughly predict stock market price movement solely based on NLP?
2. What is the best performing NLP feature extraction technique?
3. After feature extraction, identify the word or words that has most impact on prediction.
4. What is the best correlating machine learning model to the selected NLP feature extraction technique?
5. If possible, calculate the optimized hyper-parameters that produces highest prediction.

Results

1. Yes, the best accuracy we can get is 0.5883905013192612
2. Tf-idf with bi-gram
3. First five are the most significant words that indicate the increase in DIJA Adj close value, and the others indicate the decrease in it.

	features	coefficient
411	set to	1.264377
31	and other	1.259669
391	right to	1.216202
276	likely to	1.166895
14	after the	1.128663
131	fire on	-1.098193
40	around the	-1.128193
366	phone hacking	-1.144952
597	up in	-1.146688
452	the country	-1.393351

4. Logistic Regression

5. `LogisticRegression(C=0.4910000000000003, class_weight=None, dual=False, fit_intercept=True, intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1, penalty='l2', random_state=None, solver='liblinear', tol=0.0001, verbose=0, warm_start=False)`

2 Motivation and Background

The core context of our research questions is "Does new headlines have a measurable significant effect on movement of Stock Price?". Within scope of this question, we are especially interested in verifying if "News Headlines" does have impact on the movement of stock price, what machine learning model shows best prediction. Also, we are interest in finding out the word or words that have most effect on such analysis.

With the result of this research through computing, we can verify the effect of news on stock price. With an answer to this question, we can understand at least few aspect that could effect the stock price and we can use this information to continuously look for other aspects that could affect stock price. Furthermore, we can possibly build a model that could be used in fields such as financial engineering and so on.

3 Data set

We will use dataset from the **Kaggle**. This dataset consist of three different CSV files with each containing different type of data. The following is URL to our dataset.

`https://www.kaggle.com/aaron7sun/stocknews#DJIA_table.csv`

News data: Crawled historical news headlines from Reddit WorldNews Channel (`https://www.reddit.com/r/worldnews?hl`), which are ranked by Reddit users' votes. Only the top 25 headlines are considered for a single date. (Range: 2008-06-08 to 2016-07-01)

Stock data: The Dow Jones Industrial Average, or simply the Dow, is a stock market index that indicates the value of 30 large, publicly owned companies based in the United States, and how they have traded in the stock market during various periods of time. (Range: 2008-08-08 to 2016-07-01)

- **RedditNews.csv**: The first column is the "date", and the second column is the "news headlines".

- **DJIA table.csv**: Gained from Yahoo
  - "Date": YYYY-MM-DD format
  - "Open": Opening weighted average stock value in USD
  - "High": All day high in USD
  - "Low": All day low in USD
  - "Close": Closing weighted average stock value in USD
  - "Volume": Number of trades
  - "Adj Close": Adjusted closing prices - adjusted for both dividends and splits - in USD

- **Combined News DJIA.csv**: The first column is "Date", the second is "Label", and the others are news headlines("Top 1" to "Top 25"). "Label" indicates whether the DJIA Adj Close value rose or not - Binary classification. If it rose or stayed the same, it's 1, and if it decreased, it's 0.

4 Methodology

In a brief sentence, our goal is to predict stock price movement base on news headlines. The following is the steps that we are going to perform to get to the result. These following steps include algorithms, analysis and metric we will be using to predict stock price movement.

1. As mentioned, the dataset contains daily top 25 new headlines in a form of separated string. In order to use all these headlines to be used to predict single day stock movement, we are going to combine them and clean the data. Cleaning data includes steps like making all the character to lower cases and removing all the special characters.

2. With cleaned dataset, we are going to vectorize the string by using both bi-gram and Tf-Idf. We chose to use bi-gram to avoid ignoring effect of sequence of words.

3. With Tf-Idf setup to be used for analysis, we are going to use multiple classifiers such as Gradient-boosting, Ada-boost, k-NN, Stochastic Gradient Descent Classifier and more. We will also evaluate each model to find out which classifier best classify up and down movement of stock price.

4. Furthermore, we will use the highest predicting classifier to perform hyper-parameter tuning to possibly determine best hyper parameter and see why this hyper-parameter shows best result.

In a conclusion, we will propose further research topic to consider based on our conclusion.

5 Results

We used the logistic regression with n-gram(range from 1 to 3) by Tf-idf vectorizer, and the bi-gram has the highest accuracy score.
Uni-gram accuracy score: 0.46965699208443273
Bi-gram accuracy score: 0.5725593667546174
Tri-gram accuracy score: 0.5277044854881267
Combine uni, bi, and tri accuracy score: 0.5171503957783641

Confusion Matrix

—	Pred: NO	Pred: YES
Actual: NO	73	114
Actual: YES	48	144

Hyper-parameter tuning of C on Test data
Best Accuracy: 0.5883905013192612
Best Model:

```
LogisticRegression(C=0.4910000000000003, class_weight=None, dual=False,
fit_intercept=True, intercept_scaling=1, max_iter=100, multi_class='ovr',
n_jobs=1, penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
verbose=0, warm_start=False)
```

Confusion Matrix

—	Pred: NO	Pred: YES
Actual: NO	56	131
Actual: YES	25	167

	features	coefficient
1184	nigeria	1.057232
1647	state	0.890378
1552	self	0.837283
1146	mubarak	0.825880
992	korea	0.820153
423	congo	-0.943028
803	hacking	-0.958350
1518	sanctions	-1.078926
1506	run	-1.180069
445	country	-1.208424

Figure 1: Uni-gram

	features	coefficient
411	set to	1.264377
31	and other	1.259669
391	right to	1.216202
276	likely to	1.166895
14	after the	1.128663
131	fire on	-1.098193
40	around the	-1.128193
366	phone hacking	-1.144952
597	up in	-1.146688
452	the country	-1.393351

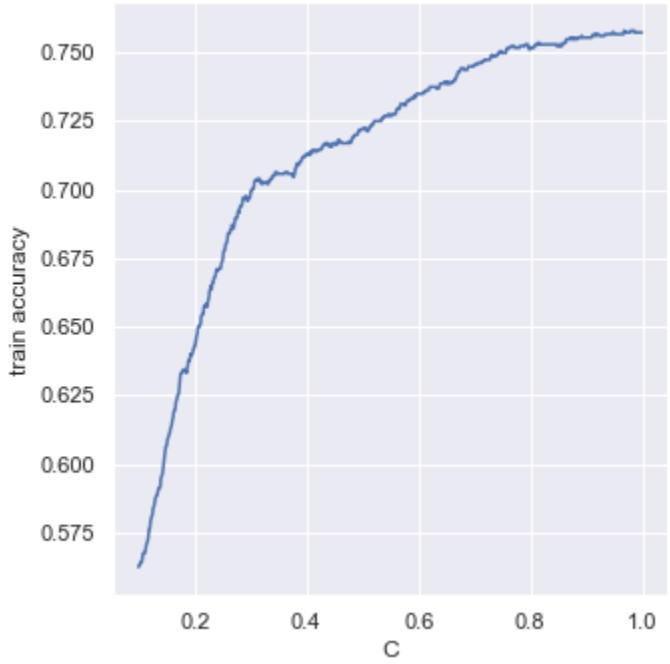
Figure 2: Bi-gram

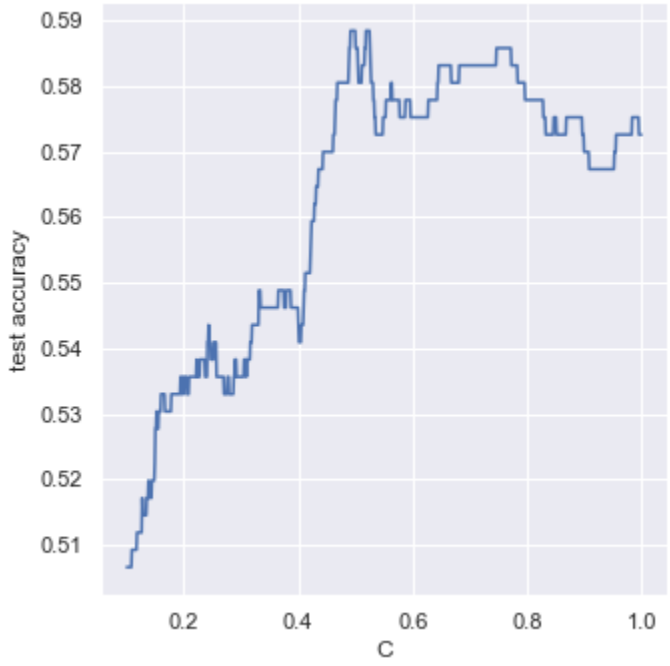
	features	coefficient
31	the right to	0.650365
27	the first time	0.525102
4	human rights watch	0.514370
23	sentenced to death	0.451972
18	of the country	0.434957
5	in order to	-0.325834
22	one of the	-0.329475
37	there is no	-0.599939
8	in the country	-0.719576
0	around the world	-0.842799

Figure 3: Tri-gram

	features	coefficient
1500	nigeria	0.949647
2096	state	0.806252
1983	self	0.778506
1455	mubarak	0.759693
156	and other	0.730263
966	hacking	-0.872009
521	congo	-0.886875
1942	sanctions	-0.997399
546	country	-1.037056
1924	run	-1.047401

Figure 4: Combine uni, bi, tri-grams





Grid Search of range from 0.4 to 0.6 for each 0.001 steps. Fitting 3 folds for each of 1001 candidates (total of 3003)
Accuracy Score : 0.575197889182058
Precision Score : 0.5543859649122806
Recall Score : 0.8229166666666666
F1 Score : 0.6624737945492662
Best model:

```
LogisticRegression(C=0.549499999999989, class_weight=None, dual=False,
```

```
fit_intercept=True, intercept_scaling=1, max_iter=100, multi_class='ovr',
n_jobs=1, penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
verbose=0, warm_start=False)
```

Confusion Matrix		
—	Pred: NO	Pred: YES
Actual: NO	60	127
Actual: YES	34	158

We can see decrease in accuracy by 0.01 but this is acceptable because this model is no longer over fit to test data. Now, let this logistic regression accuracy be our baseline for the accuracy of further prediction by other classifiers.

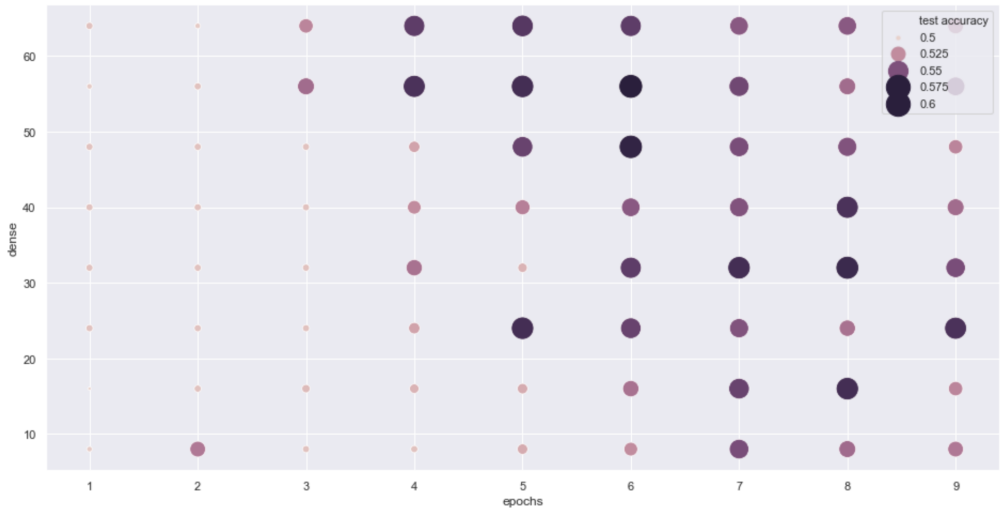
Base Random Forest Model
Base Random Forest Accuracy: 0.5224274406332454
Simple random forest classifier has 0.52 accuracy, which is still worse than our baseline, so we did random search to narrow down the hyper parameter space to improve the accuracy by hyper parameter tuning in Grid Search.
Accuracy Score: 0.5356200527704486
Best Model:

```
RandomForestClassifier(bootstrap=True, class_weight=None,
criterion='gini', max_depth=30, max_features='sqrt', max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=4,
min_samples_split=10, min_weight_fraction_leaf=0.0, n_estimators=500,
n_jobs=1, oob_score=False, random_state=1, verbose=0, warm_start=False)
```

Grid Search Best Model Accuracy: 0.5329815303430079
Grid Search Best Model:

```
RandomForestClassifier(bootstrap=True, class_weight=None,
criterion='gini', max_depth=60, max_features='sqrt', max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=3,
min_samples_split=12, min_weight_fraction_leaf=0.0, n_estimators=500,
n_jobs=1, oob_score=False, random_state=1, verbose=0, warm_start=False)
```

model 1: Hyperparameter tuning on Test data(No validation set)
model 2: Hyperparameter tuning using validation set
model 3: Base RandomForest
model 4: RandomForest using RandomSearch
model 5: RandomForest using GridSearch



Dense	Epochs	Test Accuracy
45	6	0.575198

6 Reproducing the Results

In order to reproduce the result, it is sufficient to run the Jupyter notebook from top to bottom. If there are Libraries that is not installed on your computer, open up terminal or CMD and install the Libraries using Pip install "Library name" or in case of using anaconda environment, type conda install "Library name".

7 Work Plan Evaluation

- Import data (< 1 hour) ✓
- Feature extraction (1-2 hour) → Took more than 2 hours
- Perform several NLP techniques and get accuracy score (4-6 hour) → Some took a long time to run due to hyperparameter tuning
- + We tried deep learning(Neuron Network). We researched on the use of the *Tensorflow* library and tried to get lower test error by hyperparameter tuning.
- Make some plots to compare different techniques' accuracy scores (1 hour) ✓

We were going to use git, but since we used jupyter notebook, it was hard to use git to share the code, so we shared it by messenger or email. We did meet regularly to work on the project together, and the project went pretty well.

8 Testing

Since our topic is on "Predicting up and down of stock price using machine learning", we have splitted our data into training set and the test set to test our methods. Also, in case of hyperparameter tuning, we have further divided our training data set into validation set and training set and used 3-fold cross validation technique to further improve our regularity of the models.

9 Live Presentation or Video

Live Presentation

10 Collaboration

No one