Stock Price Performance with Market and News Analytics

Brendan McGivern
November, 2018

Overview

Can we use news analytics and market data to predict stock price performance?

Let’s not be naive

> “It will fluctuate” ~ J. P. Morgan

Why is this important?

- Ubiquity of data today enables investors at any scale to make better investment decisions
- Distinguish signal from noise.

Kaggle: Two Sigma

Kaggle competitions

- Very challenging
- Real world problems using real world data
- Out of the box thinking

Two datasets:

- Market data (2007 to present)
- News data (2007 to present)

Kaggle: Two Sigma

Predict a signed confidence value: $\hat{y}_{ti} \in [-1, 1]$

Multiply by the market-adjusted return of a given *assetCode* over a ten day window

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

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

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

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 that controls whether a particular asset is included in scoring on a particular day.

Your submission score is then calculated as the mean divided by the standard deviation of your daily x_t values

Challenges

The DataFrames were large

- Market data - market_train_df shape: (4072956, 16)
- News data - news_train_df shape: (9328750, 35)

These are loaded into the Kaggle environment - feather format

7GB of RAM eliminated instantly - Python doesn’t release memory back to the OS

Garbage collecting only does so much

A number of features came already formatted - category, dates, etc…

Can only be loaded once per kernel instance

EDA

EDA

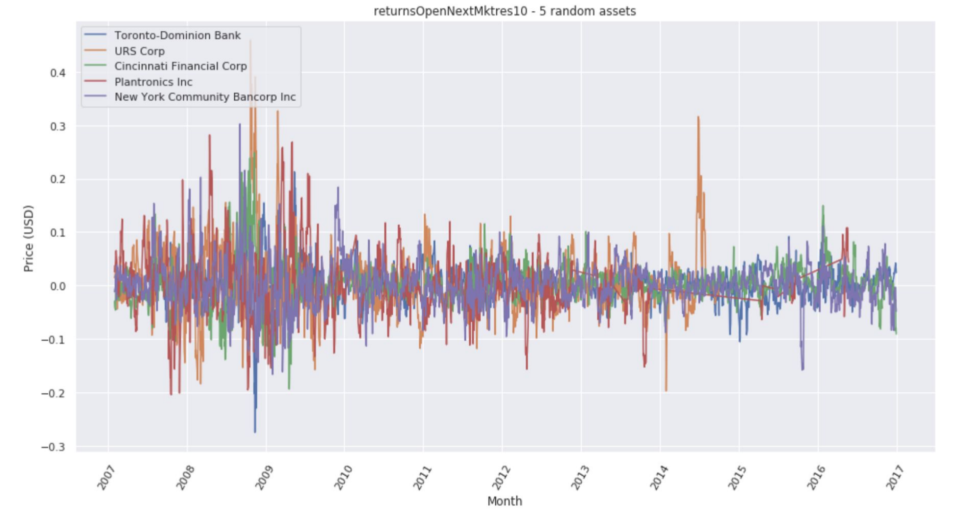
Domain knowledge

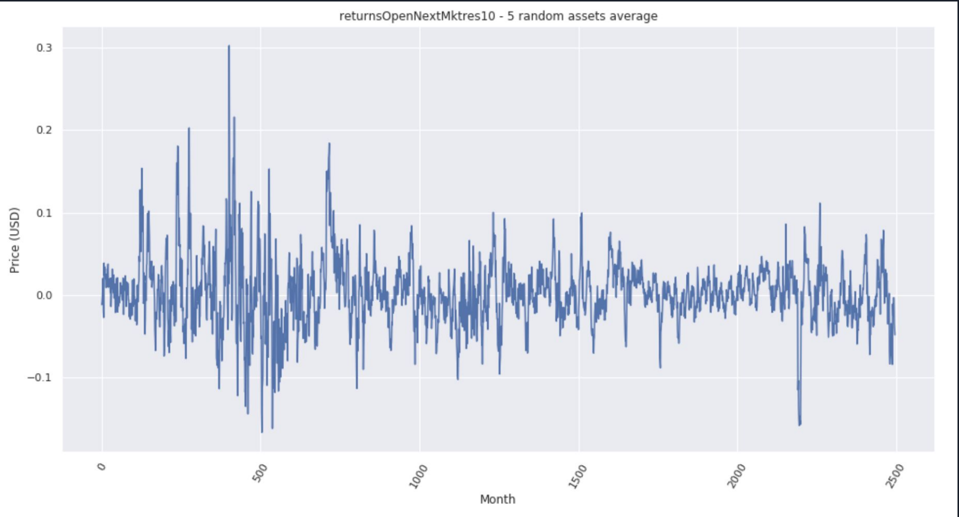
Stock splits, market crashes, flash crashes, etc...

EDA

The returnsOpenNextMktres10 feature represents a 10 day, market-residualized return

I don’t see anything immediately useful. It fluctuates.

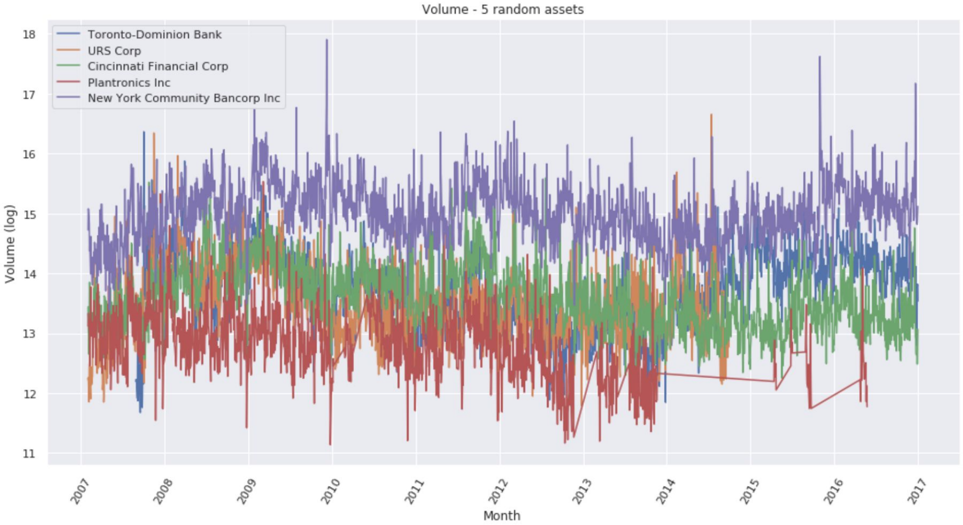




EDA

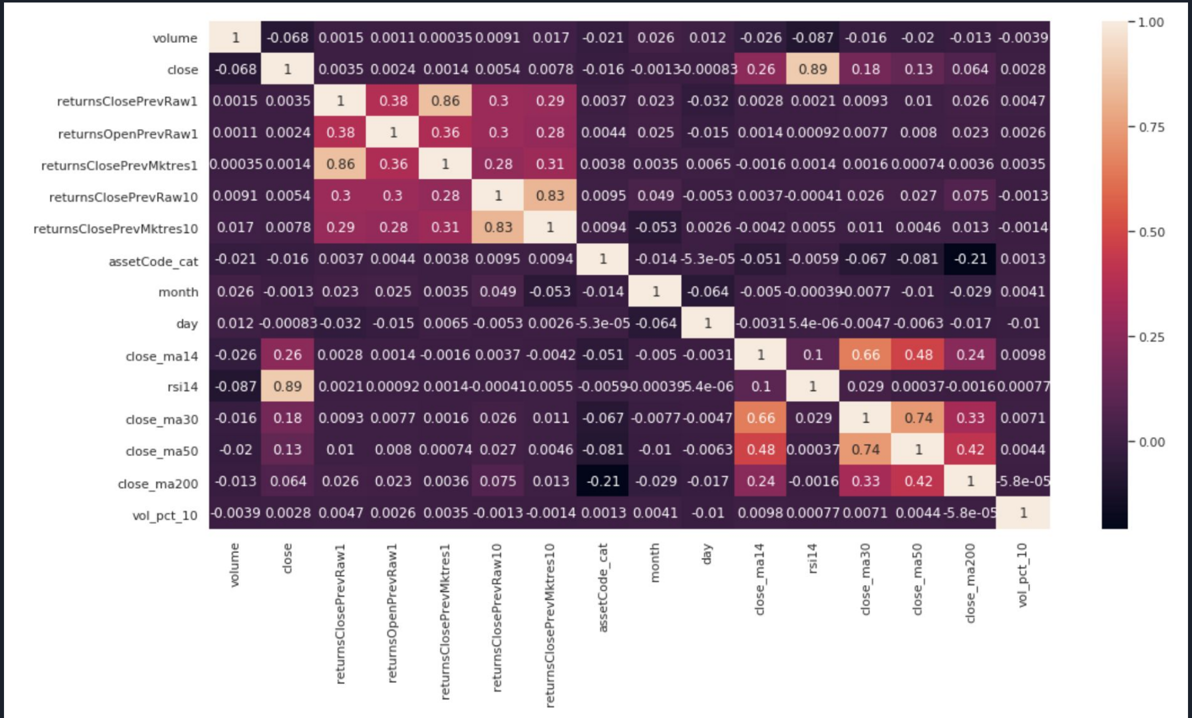
Notice New York Community Bancorp:

- highest / most variance in volume
- lowest / least variance in closing prices



EDA

We can see there is no correlation



EDA

Predicting Future Prices from Historical Prices - Apple

- We can look into the future

EDA

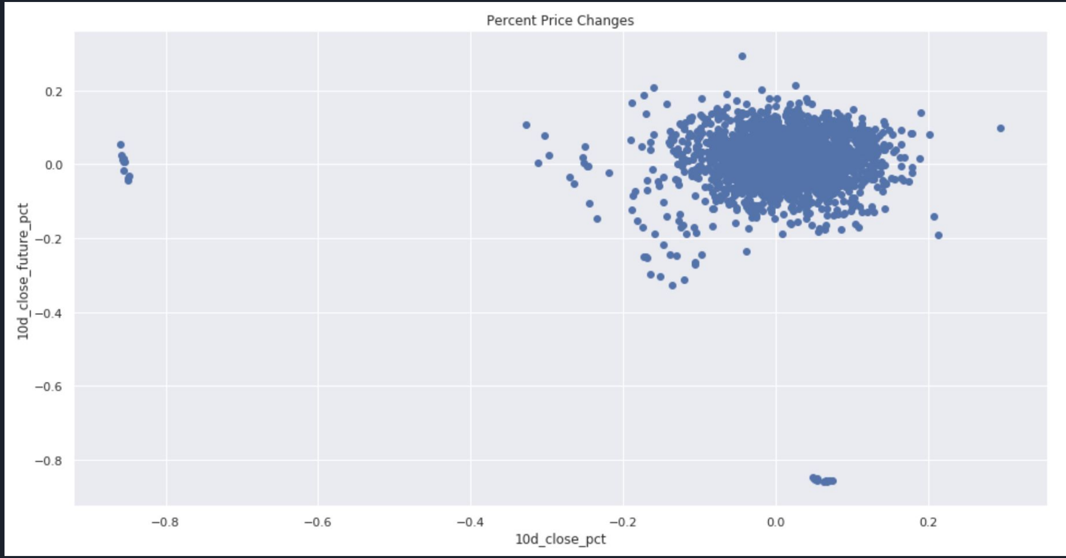
While past prices seem to be highly correlated with future prices, this is somewhat of a mirage!

- The range of future prices compared to current prices is simply too large

EDA

Percent Price Changes

- This looks a lot different than the correlation between closing price and future closing price



EDA

	10d_close_pct	10d_close_future_pct
10d_close_pct	1.000000	-0.090782
10d_close_future_pct	-0.090782	1.000000

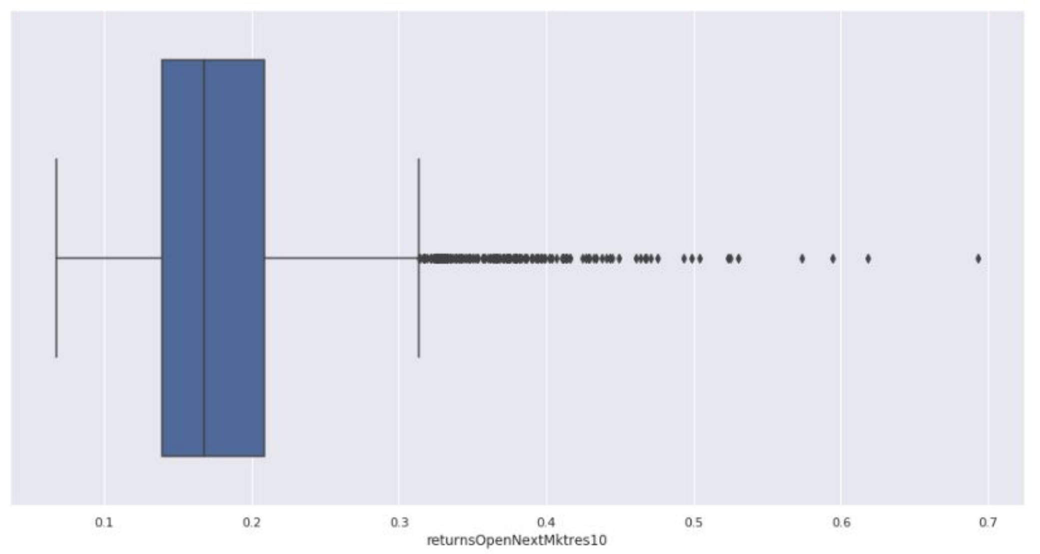
EDA

	10d_close_pct	10d_close_future_pct
10d_close_pct	1.000000	0.211207
10d_close_future_pct	0.211207	1.000000

EDA

Outliers

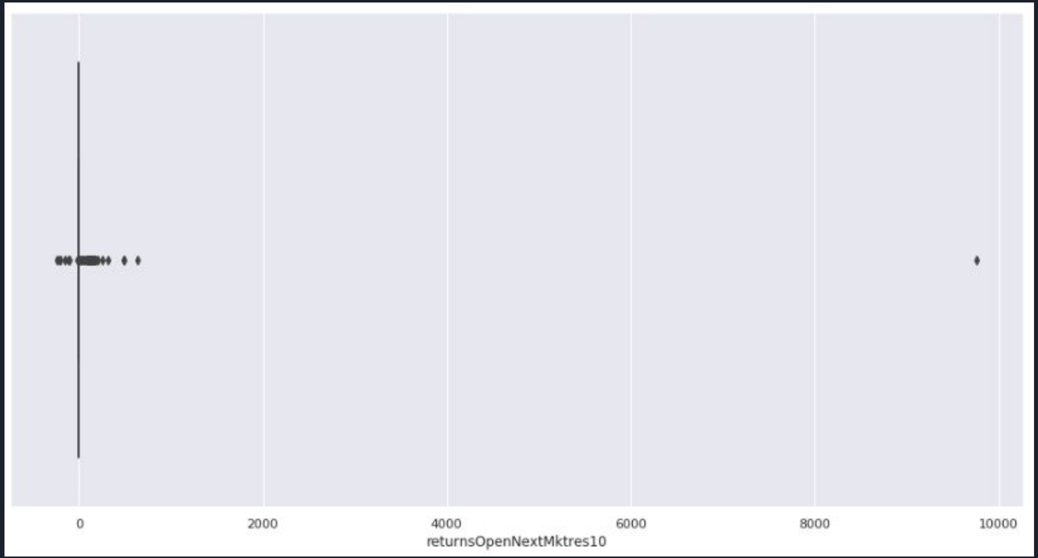
- Here we can see the 99th percentile of data
- Mean: 0.18



EDA

Outliers

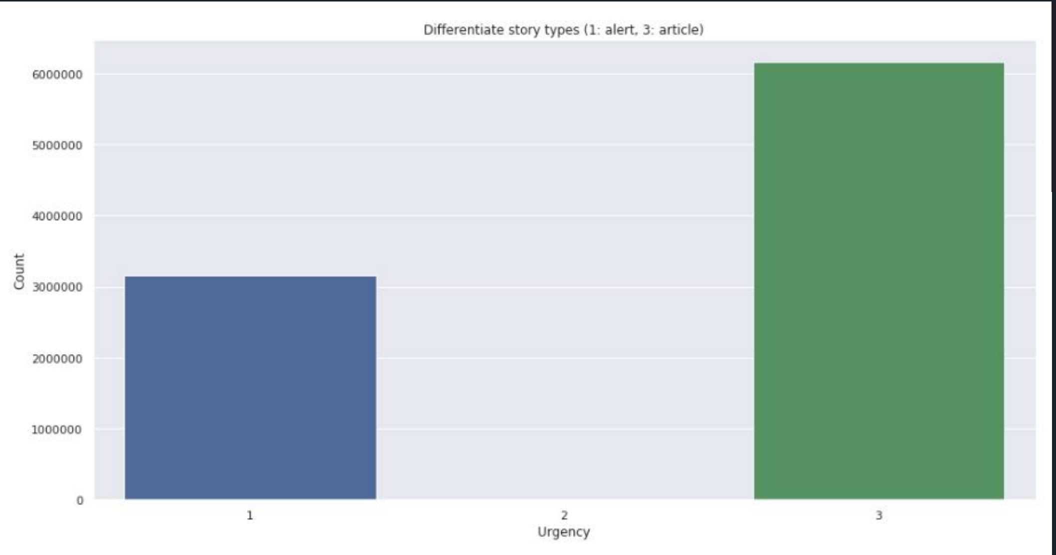
- Let’s see what falls outside of the 99th percentile
- Here is a company with returnsOpenNextMktres10 outside of the 99th percentile
- Mean: 2.9 (Petroleo Brasileiro SA Petrobras)



EDA: News

The majority of news data are classified as articles, with about 50% of that number being classified as alerts

- Type 2 - nonexistent

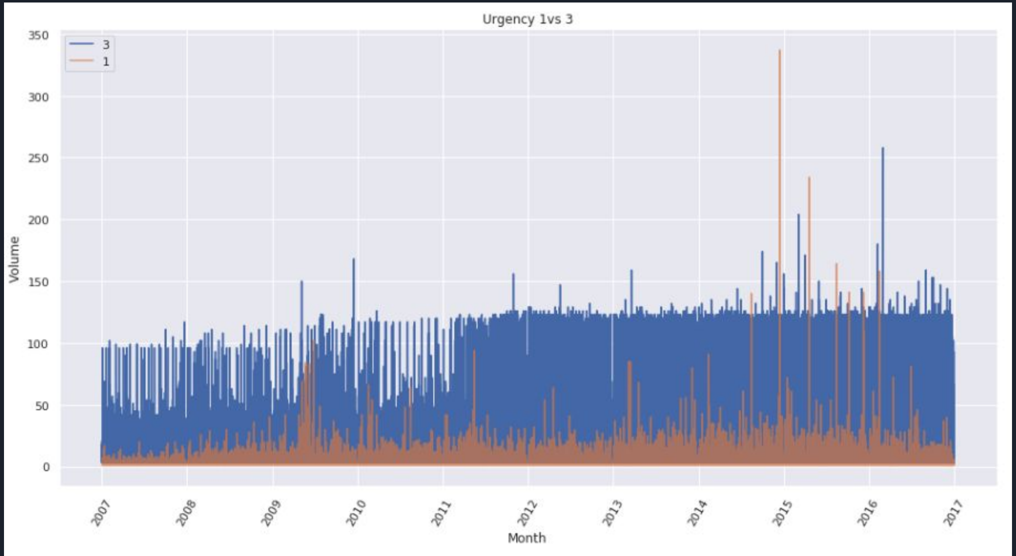


3	6162567
1	3166158
2	25

EDA: News

The plot seems to become denser in modern times

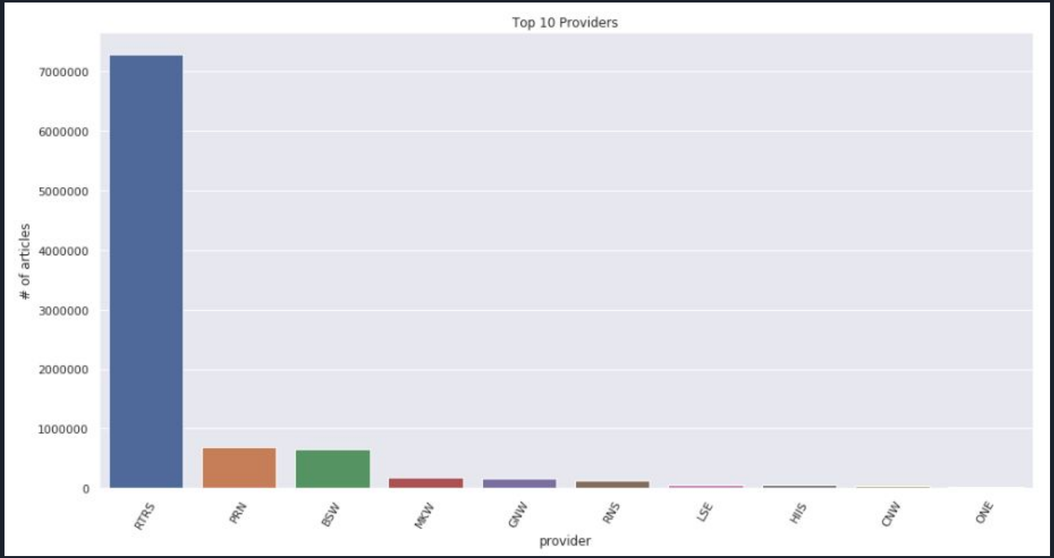
No real pattern here - I was looking for spikes in alerts around 2008



EDA: News

News providers - highly unbalanced

- Could this lead to a problem in the future?



EDA: News

Analyzing Sentiment

- Big American banks are the most negatively viewed
- Apple - Top negative and top positive sentiment

```
Top mentioned companies for negative sentiment are:
Citigroup Inc                 30823
JPMorgan Chase & Co           29129
Bank of America Corp          28197
Apple Inc                     26702
Goldman Sachs Group Inc       25044
Name: assetName, dtype: int64

Top mentioned companies for neutral sentiment are:
Barclays PLC           24898
HSBC Holdings PLC      23191
Deutsche Bank AG       20702
BHP Billiton PLC       18019
Rio Tinto PLC          16782
Name: assetName, dtype: int64

Top mentioned companies for positive sentiment are:
Barclays PLC                 22855
Apple Inc                    22770
General Electric Co          20055
Royal Dutch Shell PLC        18206
Citigroup Inc                18025
Name: assetName, dtype: int64
```

Feature Preprocessing, Exploration and Engineering

Features

Market Data

```
Data columns (total 16 columns):
time                          4072956 non-null datetime64[ns, UTC]
assetCode                     4072956 non-null object
assetName                     4072956 non-null category
volume                        4072956 non-null float64
close                         4072956 non-null float64
open                          4072956 non-null float64
returnsClosePrevRaw1          4072956 non-null float64
returnsOpenPrevRaw1           4072956 non-null float64
returnsClosePrevMktres1       4056976 non-null float64
returnsOpenPrevMktres1        4056968 non-null float64
returnsClosePrevRaw10         4072956 non-null float64
returnsOpenPrevRaw10          4072956 non-null float64
returnsClosePrevMktres10      3979946 non-null float64
returnsOpenPrevMktres10       3979902 non-null float64
returnsOpenNextMktres10       4072956 non-null float64
universe                      4072956 non-null float64
```

News Data

```
Data columns (total 35 columns):
time                    9328750 non-null datetime64[ns, UTC]
sourceTimestamp         9328750 non-null datetime64[ns, UTC]
firstCreated            9328750 non-null datetime64[ns, UTC]
sourceId                9328750 non-null object
headline                9328750 non-null object
urgency                 9328750 non-null int8
takeSequence            9328750 non-null int16
provider                9328750 non-null category
subjects                9328750 non-null category
audiences               9328750 non-null category
bodySize                9328750 non-null int32
companyCount            9328750 non-null int8
headlineTag             9328750 non-null object
marketCommentary        9328750 non-null bool
sentenceCount           9328750 non-null int16
wordCount               9328750 non-null int32
assetCodes              9328750 non-null category
assetName               9328750 non-null category
firstMentionSentence    9328750 non-null int16
relevance               9328750 non-null float32
sentimentClass          9328750 non-null int8
sentimentNegative       9328750 non-null float32
sentimentNeutral        9328750 non-null float32
sentimentPositive       9328750 non-null float32
sentimentWordCount      9328750 non-null int32
noveltyCount12H         9328750 non-null int16
noveltyCount24H         9328750 non-null int16
noveltyCount3D          9328750 non-null int16
noveltyCount5D          9328750 non-null int16
noveltyCount7D          9328750 non-null int16
volumeCounts12H         9328750 non-null int16
volumeCounts24H         9328750 non-null int16
volumeCounts3D          9328750 non-null int16
volumeCounts5D          9328750 non-null int16
volumeCounts7D          9328750 non-null int16
```

Features: My Process

- Chi-squared for *assetCode*
  - Redundant - millions or rows
- Label encoding *assetCode*
- Date features
  - *year*, *quarter*, *month*
- 4 features had null values
  - Impute using median
- Moving average and RSI (relative strength index) features
  - 14, 30, 50, 200 day features
  - RSI = 100 - (100 / 1 + RS)
  - RS = avg gain over n periods / avg loss over n periods
- Volume % change - 10 day
  - Volume and close were not correlated

- Normalize
- Temporal train / validation split
- Random Forest
- Examine feature importances
- Correlation matrix
- Drop correlated features
- Retrain the model
  - Accuracy did not change
  - Feature importances look better
- Create *coverage* feature - News data
  - determine the proportion of the article discussing the asset
- Create *position* feature - News data
  - Relative position of the first mention in the article
- TF-IDF
  - Created a *tfidf_mean* feature
- News groupby (aggregate mean) and merge

Features

Feature Importance

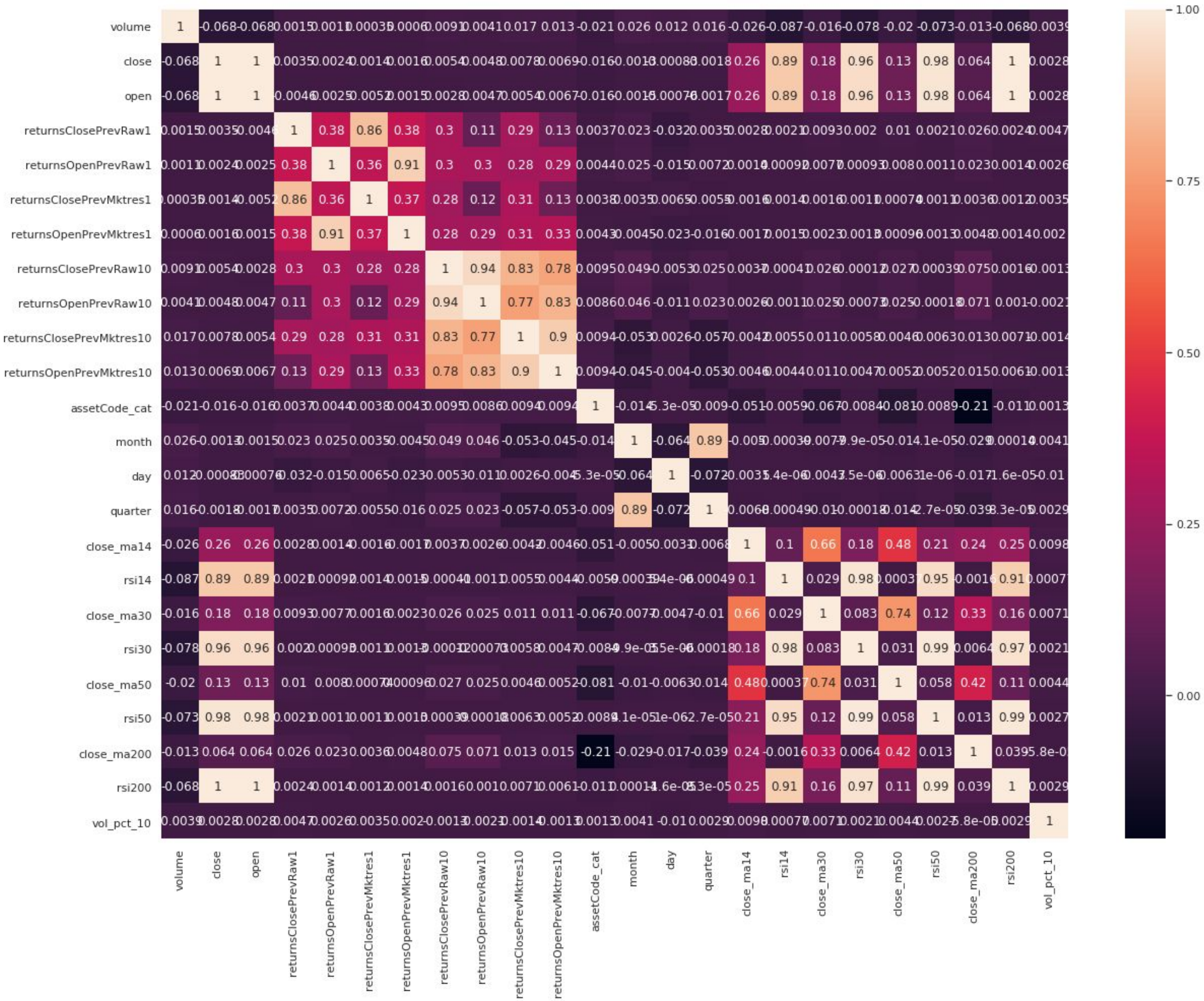
- It seems suspicious that the *month* feature is so important and *close* is near the bottom

	importance
returnsOpenPrevMktres10	0.087694
returnsOpenPrevRaw10	0.082835
month	0.065618
returnsClosePrevRaw10	0.058592
returnsClosePrevMktres10	0.056799
assetCode_cat	0.056602
volume	0.050824
close_ma200	0.048702
close_ma14	0.047997
close_ma50	0.043122
close_ma30	0.042418
rsi14	0.039586
rsi30	0.034670
returnsOpenPrevMktres1	0.033706
rsi200	0.032567
close	0.030562
rsi50	0.029960
returnsClosePrevMktres1	0.029790
open	0.028920
returnsOpenPrevRaw1	0.028608

Features

Feature Importance

- We can see a lot of correlated features



Features

Feature Importance

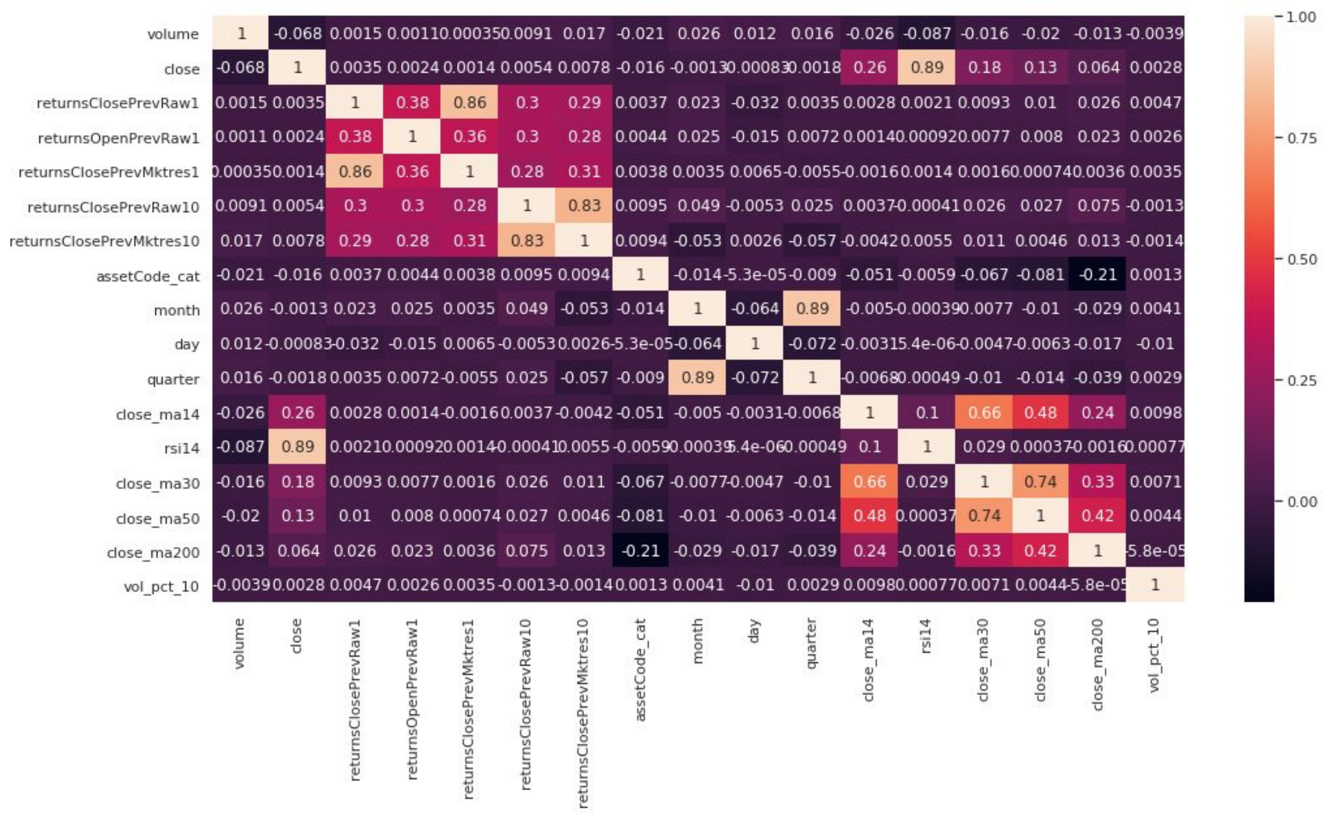
- After dropping features with greater than 90% correlation

	importance
returnsClosePrevRaw10	0.119597
returnsClosePrevMktres10	0.114182
close	0.076727
rsi14	0.073290
assetCode_cat	0.072896
month	0.064383
close_ma14	0.063179
close_ma200	0.061431
close_ma30	0.060869
volume	0.057028
close_ma50	0.055539
returnsClosePrevMktres1	0.047485
returnsClosePrevRaw1	0.036277
returnsOpenPrevRaw1	0.036137
vol_pct_10	0.028148
day	0.020215
quarter	0.012616

Features

Feature Importance

- After dropping features with greater than 90% correlation



Model Tuning and Selection

Modeling

A compilation of everything:

- Combined data processing into one function
- GridSearchCV
  - Not a good idea
- Random Search
- Random Forest
- XGBoost
- LightGBM
  - Much faster

Modeling

I performed a random search to find the optimal parameters

I ended up with a test accuracy of 60%

Discrepancy in validation accuracy and test score is due to the evaluation function used by Two Sigma

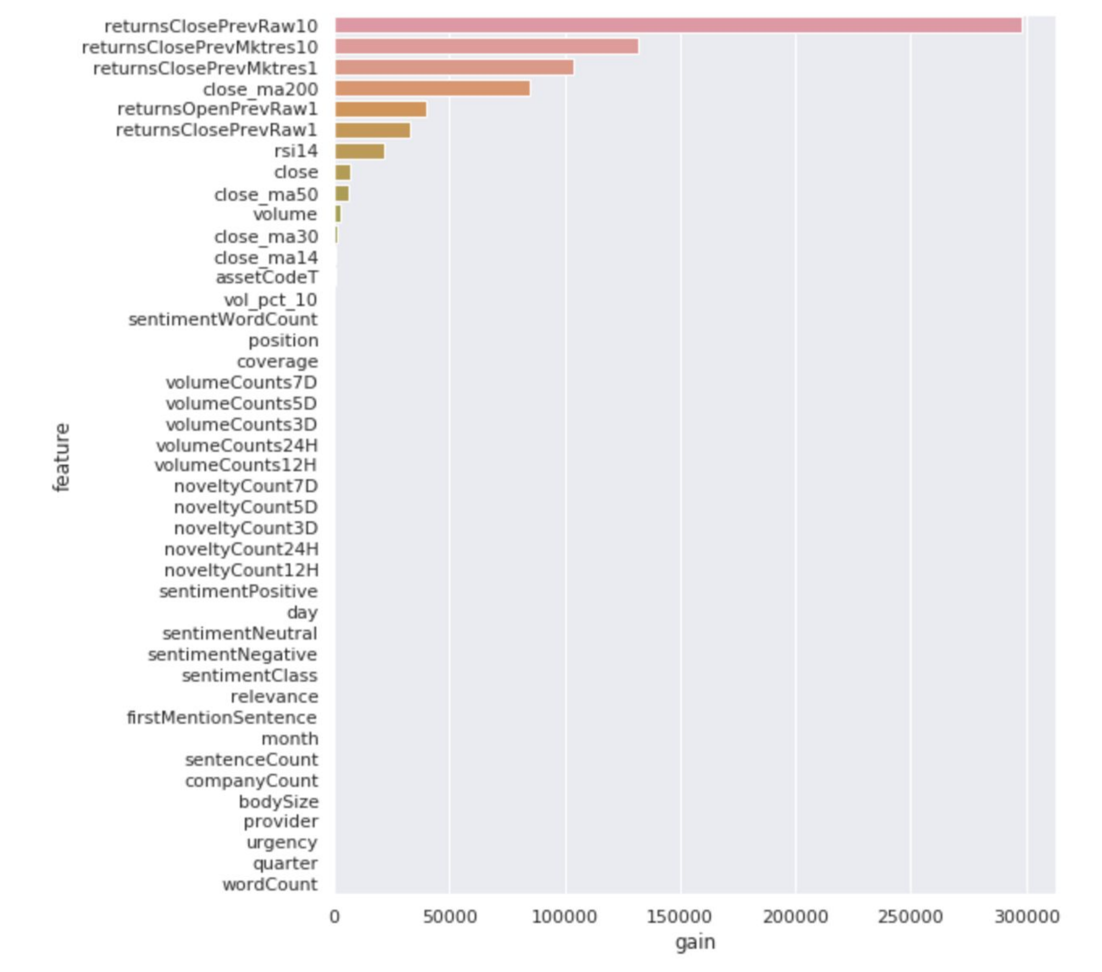
```
param_grid = {
    'learning_rate': [0.15, 0.1, 0.05, 0.02, 0.01],
    'num_leaves': [i for i in range(12, 90, 6)],
    'n_estimators': [50, 200, 400, 600, 800],
    'min_child_samples': [i for i in range(10, 100, 10)],
    'colsample_bytree': [0.8, 0.9, 0.95, 1],
    'subsample': [0.8, 0.9, 0.95, 1],
    'reg_alpha': [0.1, 0.2, 0.4, 0.6, 0.8],
    'reg_lambda': [0.1, 0.2, 0.4, 0.6, 0.8],
}
```

```
Best Score: 0.5382
Best Paramas: {'learning_rate': 0.01, 'num_leaves': 12, 'n_estimators': 200, 'min_child_sample
s': 30, 'colsample_bytree': 1.0, 'subsample': 1.0, 'reg_alpha': 0.8, 'reg_lambda': 0.4}
```

Modeling

Feature importance

News data is hard



Next Steps

- Engineering more features for the news dataset
- Experiment with stacking models
- Experiment with a custom loss function (the function used to calculate the score on the test set)

Thank You

Brendan McGivern

https://www.linkedin.com/in/brendanmcgivern1/

openroadcloud.com