Improving Stock Prediction Based on News Data

Once upon a time -- yzhan417, yzhan420, qwang52, ftong

Goal

Predicting how the stock market will perform is one of the most difficult things to do, since there are so many factors and data involved in the prediction. Especially in this information age, stock prediction just based on quantitative data is not enough. Our goal is to use the supervised learning algorithms with news data to predict the stock movement with the past dataset so as to analyze how data affects the result. Since it's really profitable and effective, individuals, institutions and even government can hire people to build and improve this model so as to make money, monitor or even manipulate the stock market.

Data

We selected data of 44 out of 100 companies from early 2016 to late 2017, based on the amount of news generated in a year (selected company are with over 200 days of news data). For each company each day:

- Price data (clean but raw): originally downloaded from Yahoo Finance (example with Apple). Based on the closing price, we calculated useful price data: the increase both in number and ratio ("prive_change" and "pchange" in data file).
- News data (pretty dirty): scraped headlines from MarketWatch and Reuters. Headlines were queried using the symbol or name of the company as the key word and were collected for articles with the same dates as the stock data. Headlines were then lowercased, filtered for stopwords, and lemmatized.
- Sentiment Score: we use VADER (Valence Aware Dictionary and Sentiment Reasoner) sentiment analysis, which is a dictionary and rule-based sentiment analysis tool that specifically targets at sentiments expressed in social media. (Using vaderSentiment package in Python)

Model+Evaluation Setup

-Overview:
We trained the model with both KNN and SVM to compare the result which SVM turns out to be more accurate. So inside the SVM model, we also compare the accuracy with 3 different situations: (a) only input the baseline (quantitative data); (b) input both baseline and news data; (c) input baseline, news data, and sentimental score. By comparison, we can find the most effective and accurate way to predict the stock movement and learn about how data affects the test set which is exactly what our goal is.

-Specific Steps:

1. Input preprocess:
   For each company and each day, we turn the news data into 100 dimensional vectors with doc2vec. By combining the stock data, news data and sentiment score, the input for each company each day would be a corresponding (6+100+1=) 107-dimensional vector.

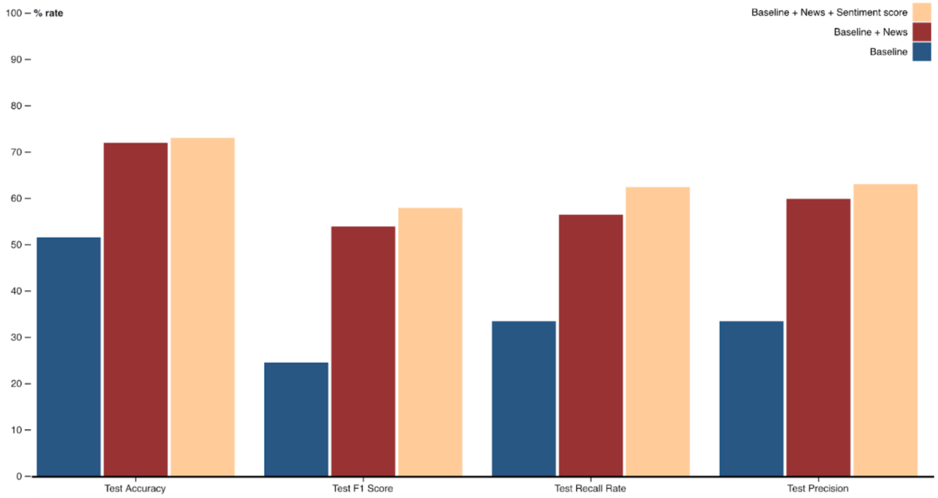
2. Train/test:
   We split dataset into train set (80%) and test set (20%). With each 107-dimensional vector in training dataset, we trained our model. In the case when the company has insufficient news, the vector will be filled with the news from the previous day.
3. Output:
   If the model predicts that the price would be flat in the next day, output would be 0; if fall, output would be -1 and if raise, output would be 1. Then we test on the test dataset.
4. Evaluation:
   This is a kind of classification problem, so we choose the classic four evaluation indexes to measure results: accuracy, recall, F1 score, and precision of the test set.

Results and Analysis

Claim #1: The classifier trained using all features outperforms baseline models by a significant margin.

	baseline	**baseline + news data**	**baseline + news data + sentiment score**
Accuracy of test set	0.5147	0.7194	0.7305
F1 of test set	0.2448	0.5386	0.5792
recall of test set	0.3337	0.5636	0.6243
precision of test set	0.3348	0.5982	0.6290

Support for Claim #1: After figuring out that SVM performs better in this case, we trained it with different kinds of input.



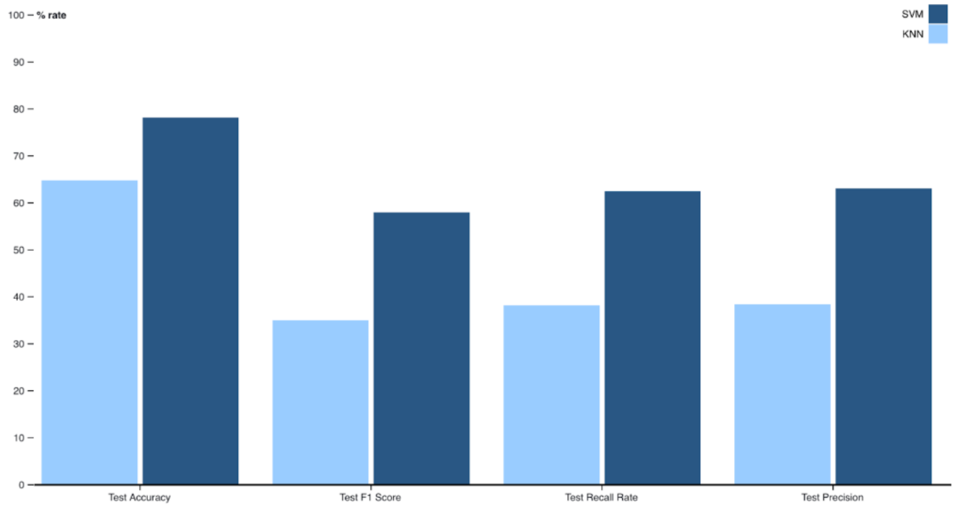
Test score for different input combinations

Claim #2: SVM outperforms KNN classifier.

Support for Claim #2: Both models are trained with all features, including baseline (price data), news data and sentiment score.

	KNN	**SVM**

Accuracy of test set	0.6473	0.7305
F1 of test set	0.3482	0.5792
recall of test set	0.3801	0.6243
precision of test set	0.3829	0.6290



Test score for different models

Claim #3: The model is performing badly with sudden jump or fall.
Support for Claim #3: Prediction accuracy are only 49.5328% for the days with more than ±5% increasing ratio of baseline, and can reach up to 79.0000% for those with minor changes (changing percentage less than ±5%).