TeslaToTheMoon

Jason Ho (jho18), Jeremy Chen (jchen208), Steven Cheung (scheung9), Neal Mahajan (nmahaja1)

Goal

The price of a stock is said to follow the expectations of company performance in the future. Being able to predict that change is vital for the financial industry to make profit and control their risk.

<u>Our goal is to use Twitter data and previous stock information to predict the next time step's Tesla stock (TSLA) price</u>. Our hope is that the popularity and sentiment of Tesla related tweets can be an estimate of investor expectations of Tesla performance. Particularly, Tesla has a large twitter following and is quite volatile, meaning an accurate prediction model reduces investor risk.

Data

We used the yfinance package and Twitter API to gather stock and tweet data from 2/6/20 to 3/6/20 and use the TextBlob package to calculate each tweet's sentiment. To fit the tweets to the stock data, we calculate an overall sentiment score for each trading hour, using equation **[1]**. For non trading hours, such as 4PM to 9:30AM or weekends, we take the average of all hours' scores within the time period which becomes the next trading hour's aggregated sentiment score. The sentiment scores are mainly grouped around zero and upon initial inspection there does not seem to be an apparent relation between it and stock close price.

$$Aggregated\ Sentiment\ =\ \sum_{tweet\ in\ hour} (tweet\ sentiment) * (favorites + retweets + replies + quotes) \quad \textbf{(1)}$$

Model+Evaluation Setup

Because our goal is to predict future time step prices, we ordered our data chronologically and made the first 80% the training data and the last 20% testing data. We remove the last three training points to create a gap between train and test sets to remove the possibility of bias. We lag our independent variables by one time step so that we do not get direct correlation.

Because share price behavior is complex, we use multiple regression to factor in several independent variables. We explored linear, polynomial degree 2, and polynomial degree 3 regression to account for varying degrees of complexity. We also test a deep learning model because it may capture hidden complexity where regression cannot. While not entirely intuitive, multiple papers cite convolutional neural networks (CNN) as one of the best models for predicting stock price[1].

To measure performance, we wrote a script that mimics trading on the testing set using the model's prediction of the next price, simulating how it would be deployed and judged in real life.

---

[1] Gudelek, Ugur & Boluk, Arda & Ozbayoglu, Murat. (2017). A deep learning based stock trading model with 2-D CNN trend detection. 1-8. 10.1109/SSCI.2017.8285188.

We ran each model with and without Twitter data and ran them against a random actor and an optimal actor (that knows the next price). In doing so, we also hope to gain an understanding of how the addition of tweet data helps or hurts our prediction accuracy.

Results and Analysis

Claim #1: In multiple linear regression, aggregated sentiment is less relevant to the prediction of stock price.
Support for Claim #1: When we run multiple linear regression on all independent variables, we found that aggregated sentiment had a coefficient close to zero (-0.015). In addition, the aggregated sentiment had one of the highest p-values of independent variables (0.64). These both suggest that the aggregated sentiment had less relevance to the calculation of predicted close price in multiple linear regression.

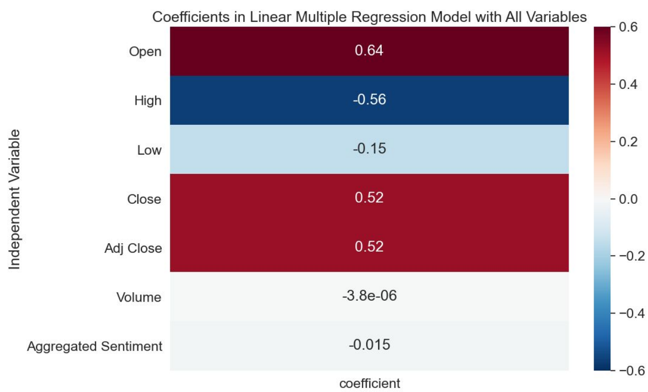

Figure 1: *Coefficient heat map*

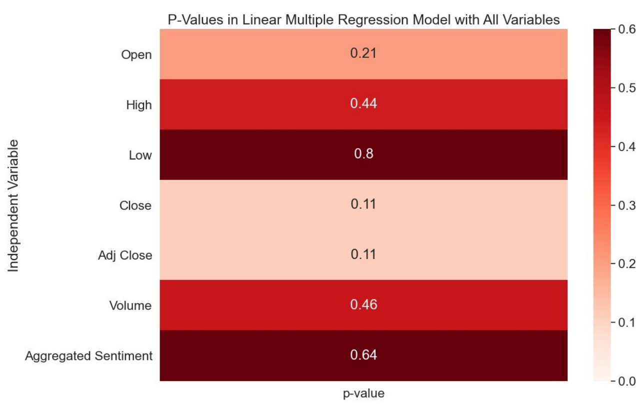

Figure 2: *P-value heat map*

Claim #2: The CNN prediction model outperformed all other models.
Support for Claim #2: Figure 3 shows each model's test MSE while table 1 shows each model's returns. Both CNN models (with and without aggregate sentiment) averaged over 5 runs had the smallest test MSE and largest returns on the test data after training on 2000 epochs. It came close to the optimal actor. We believe this is due to the CNN's increased ability to capture hidden complexity and use stock/sentiment data multiple time steps in the past.

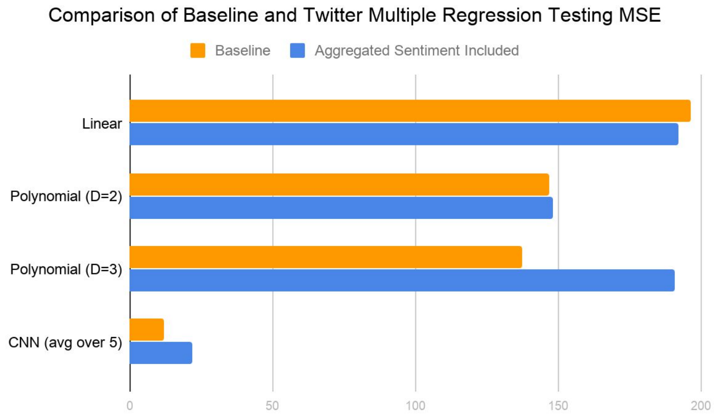

Figure 3: *Comparison of model test MSEs*

Model	**Returns Without Aggregate Sentiment Included**	**Returns With Aggregate Sentiment Included**
Linear	17.58%	12.71%
Polynomial (D=2)	15.78%	11.18%
Polynomial (D=3)	15.12%	11.06%
CNN (avg over 5)	18.93%	18.38%
Random Actor	~0%	N/A
Optimal Actor	19.97%	N/A

Table 1: *Each model's returns on the test data from the trading script*

Claim #3: All models performed better without Twitter data.
Support for Claim #3: In Table 1, we found that all of our machine learning models made larger percentage returns when aggregated sentiment was not included. We believe that this may be attributed to the models learning unnecessary/nonexistent relationships between sentiment and Tesla Close Price.

Claim #4: All models were able to predict the next time step's price to some degree of success.
Support for Claim #4: According to Table 1, all models made more returns than a random actor, meaning they had some degree of success when predicting the next time step's price. However, we worked with a small time frame so our results may not generalize. Furthermore, our data is from a period of increased volatility in TSLA. In the future, we plan on increasing the time frame of our data so our results could be more conclusive.