Extending Prediction Tools by Combining Stock Price Data with Market News Sentiment Analysis

Introduction

Stock market prediction has and currently is a very active area of research, however it is quite difficult to predict the movement of stock prices across the market. They are often highly volatile and unpredictable, even if a stock has been performing well over a long period of time, it does not guarantee that this behavior will remain. There is always the chance of it, or even the market in its entirety crashing.

While the state of the economy and other economical factors such as inflation play into the unpredictability of the stock market, there is also the fact that certain events around the world can heavily impact the increase or decrease in stock returns. Geopolitical risks for example have been shown to contribute up to 40% decrease in stock returns through market analysis of 22 different countries (Egan).

Similarly, the way a company is portrayed by the media, and as such the way it is perceived by the people, can determine the future of that stock. If there is negative press surrounding a business, investors may sell their holdings which would cause the price of the stock to go down. This applies to social media, news and all forms of media that aims to communicate through the internet.

This is where sentiment analysis comes into play. By collecting data about social media posts on Twitter or Instagram, it is possible to process this information and get a prediction on whether the "sentiment" of the post or tweet is overall positive or negative with regards to how the stock may perform in the near future. Training is done as per usual using data that already has a sentiment label, taking out any unnecessary words or special characters (Gupta, 2020). After the model has been trained appropriately, we can use data that does not have a pre-existing sentiment label and get a prediction.

Rather than social media posts, I have chosen to use market news data for sentiment analysis as social media sentiment analysis has already been worked on and implemented by users researching this field. The final program aims to use news articles and history prices, then apply textual and time series processing to get a final evaluation on the price. The stock price prediction will be adjusted by a factor depending on the sentiment labeling received from market analysis.

Libraries

yfinance: This api allows market data to be downloaded from Yahoo!finance, using this we can download both history price data as well as market news data for the required stock (pypi.org, n.d.).

Textblob: This library is used for processing textual data, more specifically we can use the api to tokenize text and perform sentiment analysis as well as classification (textblobl.readthedocs.io, n.d.).

Functions

First, we will need to get the market news data. We will do this using the yfinance api.

```
def get_news(ticker):

    #get market news data for necessary ticker
    news = yf.Ticker(ticker).news
```

[{'uuid': 'aa3ca8b0-27a2-3386-8602-1fd1c998132f', 'title': 'Tesla Woes Bolster Appeal of Top China EV Maker BYD: Tech Watch', 'publisher': 'Bloomberg', 'link': 'https://finance.yahoo.com/news/tesla-woes-bolster-appeal-top-020000753.html', 'providerPublishTime': 1698638003, 'type': 'STORY', 'thumbnail': {'resolutions': [{'url': 'https://s.yimg.com/uu/api/res/1.2/3Qgv5UPR8i0wcQ1rYMpl3Q--~B/aD0xMzMzO3c9MjAwMDthcHBpZD15dGFjaHlvbg--/https://media.zenfs.com/en/bloomberg_markets_842/19cc78ed12d26f2a29732ead56dc6eae', 'width': 2000, 'height': 1333, 'tag': 'original'}, {'url': 'https://s.yimg.com/uu/api/res/1.2/ePSxvd266.WFYEui9nPlDQ--~B/Zmk9ZmlsbDtoPTE0MDtweW9mZj0wO3c9MTQwO2FwcGlkPXl0YWNoeW9u/https://media.zenfs.com/en/bloomberg_markets_842/19cc78ed12d26f2a29732ead56dc6eae', 'width': 140, 'height': 140, 'tag': '140x140'}]}, 'relatedTickers': ['TSLA', 'BYDDY']}

Data is provided in this format, we can see that this news data is related to two tickers, TSLA and BYDDY. We are interested primarily in the 'title' attribute, which holds textual data that can be analyzed for sentiment labeling and classification.

We will then need a function that utilizes TextBlob to provide sentiment analysis. For this application we will be using rule-based sentiment analysis to calculate the text sentiment. This is done by analyzing the given text and using the occurrence of certain positive or negative words to provide an overall assumption (ES). The TextBlob api provides two properties for given text, polarity and subjectivity (textblobl.readthedocs.io, n.d.).

We are interested in the polarity attribute, which is the measure of the orientation of the tokenized text.

```
def get_sentiment(newstitle):

    analysis = TextBlob(newstitle)

    return analysis.sentiment.polarity
```

The polarity value will sit within a range between -1.0 and 1.0, with a polarity closer to 1.0 indicating a more overall positive sentence.

We will now combine the two functions to perform the TextBlob analysis on the ticker news data. There is more than one ‘title’ to be analyzed so there must be a loop to iterate through each one, providing a result array with all the necessary polarities.

```
    # get market news data for necessary ticker
    news = yf.Ticker(ticker).news

    # get sentiment polarity for each news 'title'
    sentiments = [get_sentiment(item['title']) for item in news]

    # return average polarity of news surrounding the specified ticker
    return sum(sentiments) / len(sentiments)
```

The average polarity will then be returned.

Ensemble

We will integrate the sentiment polarity data into the ensemble function that we already have.

```
def ensemble(model, sarima_data, lstm_data, arima_data, ticker):
    lstm_prediction = predict(model, lstm_data)
    sarima_prediction = train_SARIMA(sarima_data)
    arima_prediction = train_ARIMA(arima_data)

    ensemble_prediction = (
        lstm_prediction + sarima_prediction + arima_prediction) / 3  # average the prediction values

    news_sentiment = get_news(ticker)
    adjustment = 1 + news_sentiment  # account for negative polarity

    final_prediction = ensemble_prediction * adjustment

    return final_prediction


final_df = final_dataframe(model, data)
future_price = ensemble(model, data, data, data, "TSLA")  # use TSLA ticker
```

The function is called and the ticker is passed. Adjustment variable is used, if the polarity is negative then a factor of 0.x will reduce the predicted price and if the polarity is positive, it will raise the price. The predicted price from ensemble is multiplied by the adjustment factor to get a final price.

Testing

Before we test the new function, we will get the price prediction without sentiment analysis for comparison.

Parameters: 4 layers, GRU, 15 epochs, SARIMA: 3, 2, 5 2, 1, 2, 12

ARIMA: 3, 2, 5

```
Future price after 30 days is 214.00$
```

The price after a month is predicted to be $214.

We will now test the predicted price after adjustment through sentiment polarity.

```
Epoch 7: val_loss improved from 0.00343 to 0.00342, saving model to results\2023-10-30_TSLA-sh-1-sc-1-sbd-0-mean_squared_error-rmsprop-GRU-seq-50-step-30-
units-256-b.h5
21/21 [==============================] - 11s 523ms/step - loss: 0.0071 - mean_squared_error: 0.0071 - val_loss: 0.0034 - val_mean_squared_error: 0.0034
Epoch 8/15
21/21 [==============================] - ETA: 0s - loss: 0.0066 - mean_squared_error: 0.0066
Epoch 8: val_loss did not improve from 0.00342
21/21 [==============================] - 11s 537ms/step - loss: 0.0066 - mean_squared_error: 0.0066 - val_loss: 0.0097 - val_mean_squared_error: 0.0097
Epoch 9/15
21/21 [==============================] - ETA: 0s - loss: 0.0076 - mean_squared_error: 0.0076
Epoch 9: val_loss did not improve from 0.00342
21/21 [==============================] - 11s 506ms/step - loss: 0.0076 - mean_squared_error: 0.0076 - val_loss: 0.0051 - val_mean_squared_error: 0.0051
Epoch 10/15
```

```
Machine precision = 2.220D-16
 N =           13     M =           10
 This problem is unconstrained.

At X0         0 variables are exactly at the bounds

At iterate    0    f=  3.39314D+00    |proj g|=  6.23305D-01

At iterate    1    f=  3.18981D+00    |proj g|=  3.10323D-01

At iterate    2    f=  3.11323D+00    |proj g|=  1.18267D-01
```

```
  return get_prediction_index(
Future price after 30 days is 242.58$
```

The predicted price is now $242.58 after a month. We can see that there is a significant difference in the results simply by analyzing the titles of news articles that are related to the stock that we are interested in.

Conclusion

As seen from the test results, the addition of market news sentiment leads to drastic changes in the result of stock price prediction. This not only proves that price prediction through history prices only is unreliable, but also that a far larger set of variables need to be taken into account to make sure that the predicted value can be used for real world investment applications.

References

Egan, J. (n.d.). How Are Stock Prices Determined: The Factors that Affect Share Prices of Listed Companies. *Time*. Retrieved from TIme.com

ES, S. (n.d.). Sentiment Analysis in Python: TextBlob vs Vader Sentiment vs Flair vs Building It From Scratch.

Gupta, R. (2020). *Sentiment Analysis for Stock Price Prediction.* IEEE.

pypi.org. (n.d.). Retrieved from https://pypi.org/project/yfinance/

textblobl.readthedocs.io. (n.d.). Retrieved from https://textblob.readthedocs.io/en/dev/