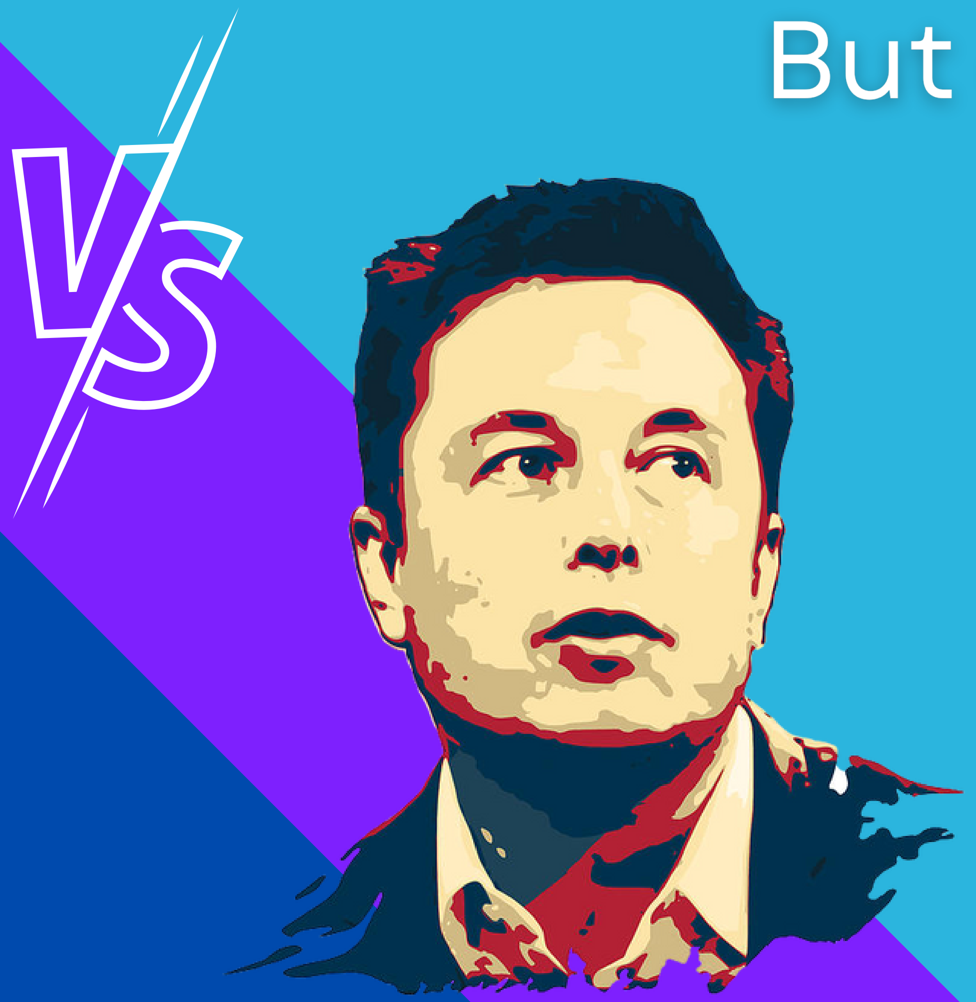
ELON TWEET ANALYSIS

Elon Musk, the richest man on Earth with over 107M followers on twitter, uses the platform to share his opinions, often controversial.

But does he wield pricing power?



Presented by:

Ben Gunnels, Greg Richardson,
Manisha Lal , Quianna Rolston,
Smruthi Danda & Zehra Vahidy

QUESTIONS WE HOPE TO ANSWER:

- Does a correlation exist between Elons tweets and stock market reactions ?
- Is it possible to predict whether or not a stock with rise/fall/remain with the sentiment of his tweets?
- Can we design a stock purchase method based on Elons tweets?

VS

QUESTIONS WE HOPE TO ANSWER:

BONUS

- Do Elons tweets affect companies other than TSLA? (TEST: 'DOGE' + 'TWTR')
- Should we consider Elons tweet history on a certain company before investing?

TWEET DATA

pulled from

kaggle™

*Timeframe:
December 1 2011 - April 13, 2021

STOCKS ANALYZED

Tesla (TSLA) *Timeframe: June 29 2010 - April 13, 2021

Dogecoin (DOGE) *Timeframe: November 9 2017 - April 13, 2021

SP500 (SPY) *Timeframe: June 29 2010 - April 13, 2021

STOCK DATA

pulled from

yahoo! finance

TWEETS IMPORTED

Elon_Tweet_Sentiment_Analysis

[214…

```
import pandas as pd
from pathlib import Path
import warnings
warnings.filterwarnings("ignore")
import yfinance as yf
```

[215…

```
elon_tweets_reply = pd.read_csv(Path("../Resources/tweets_and_replies.csv"))
display(elon_tweets_reply.head())
display(elon_tweets_reply.tail())
```

	Id	Date	Text	ConversationId
0	1575021541103874048	2022-09-28 07:16:12+00:00	https://t.co/mEBAgBCCkj	1575021541103874048
1	1574958348163612672	2022-09-28 03:05:06+00:00	I guess this joke is a slow burn ...	1574895951973449729
2	1574957722415398912	2022-09-28 03:02:36+00:00	@WholeMarsBlog Big improvement in high speed c...	1574940528520536064
3	1574956999938256896	2022-09-28 02:59:44+00:00	@chrispavlovski @dbongino @rustyrockets Maybe ...	1574861160502927385
4	1574901832622612480	2022-09-27 23:20:31+00:00	Make "hair on fire" not just a metaphor	1574901832622612480

	Id	Date	Text	ConversationId
7248	1433137351203561474	2021-09-01 18:39:05+00:00	@Kristennetten @StianWalgermo @Tesla @ARKInves...	1433080556376530952
7249	1433123220643717120	2021-09-01 17:42:56+00:00	@thesheetztweetz They can shake their fist at ...	1433081862918975496
7250	1433122554156257280	2021-09-01 17:40:17+00:00	@StianWalgermo @Tesla @ARKInvest @WholeMarsBlo...	1433080556376530952
7251	1433121450446127106	2021-09-01 17:35:54+00:00	@Max9907826460 @TeslaratiTeam Our new crane!	1433001281753337856
7252	1433115031940440065	2021-09-01 17:10:24+00:00	@AaronS5_ @ashleevance 2021 has been the year ...	1433110569100333061

[216…

```
elon_tweets_reply.info()
```

TWEETS 'CLEANED'

[219…

```python
def preprocess_tweet(sen):
    '''Cleans text data up, leaving only 2 or more char long non-stepwords composed of A-Z & a-z only
    in lowercase'''

    sentence = sen.lower()

    # Remove RT
    sentence = re.sub('RT @\w+: '," ", sentence)

    # Remove special characters
    sentence = re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)"," ", sentence)

    # Single character removal
    sentence = re.sub(r"\s+[a-zA-Z]\s+", ' ', sentence)  # When we remove apostrophe from the word "Mark's", the apostrophe is replaced by an empty

    # Remove multiple spaces
    sentence = re.sub(r'\s+', ' ', sentence)  # Next, we remove all the single characters and replace it by a space which creates multiple spaces i

    return sentence
```

[220…

```python
cleaned_tweets = []

for tweet in elon_tweets_reply['Text']:
  cleaned_tweet = preprocess_tweet(tweet)
  cleaned_tweets.append(cleaned_tweet)
```

[221…

```python
elon_tweets_reply['cleaned'] = pd.DataFrame(cleaned_tweets)
elon_tweets_reply.head(5)
```

[221…

	Id	Date	Text	ConversationId	cleaned
0	1575021541103874048	2022-09-28 07:16:12+00:00	https://t.co/mEBAgBCCkj	1575021541103874048	
1	1574958348163612672	2022-09-28 03:05:06+00:00	I guess this joke is a slow burn …	1574895951973449729	i guess this joke is slow burn
2	1574957722415398912	2022-09-28 03:02:36+00:00	@WholeMarsBlog Big improvement in high speed c…	1574940528520536064	big improvement in high speed cross traffic v…
3	1574956999938256896	2022-09-28 02:59:44+00:00	@chrispavlovski @dbongino @rustyrockets Maybe …	1574861160502927385	maybe worth talking at some point
4	1574901832622612480	2022-09-27 23:20:31+00:00	Make "hair on fire" not just a metaphor	1574901832622612480	make hair on fire not just metaphor

CALCULATING NEGATIVE, POSITIVE, NEUTRAL AND COMPOUND VALUES

```python
#Calculating Negative, Positive, Neutral and Compound values

elon_tweets_reply[['polarity', 'subjectivity']] = elon_tweets_reply['cleaned'].apply(lambda Text: pd.Series(TextBlob(Text).sentiment))
for index, row in elon_tweets_reply['cleaned'].iteritems():
    score = SentimentIntensityAnalyzer().polarity_scores(row)
    neg = score['neg']
    neu = score['neu']
    pos = score['pos']
    comp = score['compound']
    if comp <= -0.05:
        elon_tweets_reply.loc[index, 'sentiment'] = "negative"
    elif comp >= 0.05:
        elon_tweets_reply.loc[index, 'sentiment'] = "positive"
    else:
        elon_tweets_reply.loc[index, 'sentiment'] = "neutral"
    elon_tweets_reply.loc[index, 'neg'] = neg
    elon_tweets_reply.loc[index, 'neu'] = neu
    elon_tweets_reply.loc[index, 'pos'] = pos
    elon_tweets_reply.loc[index, 'compound'] = comp

elon_tweets_reply.head(5)[['polarity', 'subjectivity']] = elon_tweets_reply['cleaned'].apply(lambda Text: pd.Series(TextBlob(Text).sentiment))
for index, row in elon_tweets_reply['cleaned'].iteritems():
    score = SentimentIntensityAnalyzer().polarity_scores(row)
    neg = score['neg']
    neu = score['neu']
    pos = score['pos']
    comp = score['compound']
    if comp <= -0.05:
        elon_tweets_reply.loc[index, 'sentiment'] = "negative"
    elif comp >= 0.05:
        elon_tweets_reply.loc[index, 'sentiment'] = "positive"
    else:
        elon_tweets_reply.loc[index, 'sentiment'] = "neutral"
    elon_tweets_reply.loc[index, 'neg'] = neg
    elon_tweets_reply.loc[index, 'neu'] = neu
    elon_tweets_reply.loc[index, 'pos'] = pos
    elon_tweets_reply.loc[index, 'compound'] = comp

elon_tweets_reply.head(5)
```

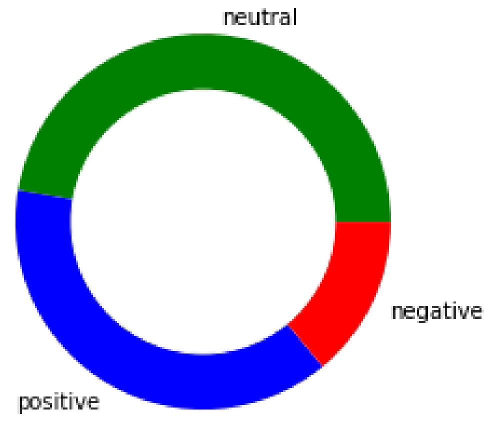
	Total	Percentage
neutral	3430	47.29
positive	2807	38.70
negative	1016	14.01

	Id	Date	Text	ConversationId	cleaned	polarity	subjectivity	sentiment	neg	neu	pos	compound
0	1575021541103874048	2022-09-28 07:16:12+00:00	https://t.co/mEBAgBCCkj	1575021541103874048		0.00	0.00	neutral	0.000	0.000	0.000	0.0000
1	1574958348163612672	2022-09-28 03:05:06+00:00	I guess this joke is a slow burn … 😈	1574895951973449729	i guess this joke is slow burn	-0.30	0.40	positive	0.000	0.694	0.306	0.2960
2	1574957722415398912	2022-09-28 03:02:36+00:00	@WholeMarsBlog Big improvement in high speed c...	1574940528520536064	big improvement in high speed cross traffic v...	0.04	0.16	positive	0.000	0.786	0.214	0.4588
3	1574956999938256896	2022-09-28 02:59:44+00:00	@chrispavlovski @dbongino @rustyrockets Maybe ...	1574861160502927385	maybe worth talking at some point	0.30	0.10	positive	0.000	0.725	0.275	0.2263
4	1574901832622612480	2022-09-27 23:20:31+00:00	Make "hair on fire" not just a metaphor	1574901832622612480	make hair on fire not just metaphor	0.00	0.00	negative	0.286	0.714	0.000	-0.3400

ELON HAS MORE TO SAY WHEN THINGS AREN'T GOING HIS WAY....

```
size=pichart["Percentage"]

# Create a circle for the center of the plot
my_circle=plt.Circle( (0,0), 0.7, color='white')
plt.pie(size, labels=names, colors=['green','blue','red'])
p=plt.gcf()
p.gca().add_artist(my_circle)
plt.show()
```



In [227…

```
#Calculating tweet's lenght and word count
elon_tweets_reply['text_len'] = elon_tweets_reply['cleaned'].astype(str).apply(len)
elon_tweets_reply['text_word_count'] = elon_tweets_reply['cleaned'].apply(lambda x: len(str(x).split()))
```

In [228…

```
round(pd.DataFrame(elon_tweets_reply.groupby("sentiment").text_len.mean()),2)
```

	text_len
sentiment	
negative	93.00
neutral	30.66
positive	73.40

Tweets are longer when the sentiment has feeling...especially negative

VERIFYING TWEET SENTIMENT TO MARKET HAPPENINGS

```python
import time

# Load env variables in from .env
load_dotenv()

# Set base URL for meaningcloud API
url = "https://api.meaningcloud.com/sentiment-2.1"

# Create payload to send to meaningcloud API
payload={
    'key': os.getenv("MEANINGCLOUD_API_KEY"),
    'txt': '',
    'lang': 'en'
}

# Read the CSV of yesterday's tweets and gather the sentiments from meaningcloud
yesterdays_elon_tweets_df = pd.read_csv('yesterday.csv')

# Iterate through yesterdays_tweets dataframe and get sentiment from meaningcloud API
for index, row in yesterdays_elon_tweets_df.iterrows():
    time.sleep(1)
    # Set the payload text to the raw tweet text
    payload['txt'] = row['Tweet']

    # Call meaningcloud API and store response as JSON
    response = requests.post(url, data=payload).json()

    # # Update the dataframe with the new columns for score_tag and confidence
    yesterdays_elon_tweets_df.at[index,'score_tag'] = response['score_tag']
    yesterdays_elon_tweets_df.at[index,'confidence'] = response['confidence']

# Create a column 'datetime' by concatenating the 'date' and 'time' columns
yesterdays_elon_tweets_df['datetime'] = pd.to_datetime(yesterdays_elon_tweets_df['date'] + ' ' + yesterdays_elon_tweets_df['time'])

# Set the 'datetime' column as the index
yesterdays_elon_tweets_df = yesterdays_elon_tweets_df.set_index('datetime')

# Localize the times to UTC
yesterdays_elon_tweets_df = yesterdays_elon_tweets_df.tz_localize('utc')

# Output the resulting dataframe to CSV
yesterdays_elon_tweets_df.to_csv('yesterdays_elon_tweets_df.csv')
```

Tweet sentiment is determined

Dataframe now includes tweet and sentiment

df to csv...

DATASET NOW INCLUDES SENTIMENT

README.md | sentiment_df.csv | sentiment_analyzer.ipynb | ElonTweetAnalysis Deck.pdf

Delimiter: ,

	datetime	id	conversation_id	created_at	date	time	timezone	user_id	username
9	2021-04-11 01:09:02+...	1381006333864861696	1381006333864861696	2021-04-11 01:09:02 ...	2021-04-11	01:09:02	300	44196397	elonmusk
10	2021-04-10 23:22:20...	1380979481309945856	1380949316215451649	2021-04-10 23:22:20 ...	2021-04-10	23:22:20	300	44196397	elonmusk
11	2021-04-10 22:49:26...	1380971204299747328	1380873129535418369	2021-04-10 22:49:26 ...	2021-04-10	22:49:26	300	44196397	elonmusk
12	2021-04-10 22:46:36...	1380970491356225536	1380860614990499840	2021-04-10 22:46:36 ...	2021-04-10	22:46:36	300	44196397	elonmusk
13	2021-04-10 11:32:35+...	1380800869382320134	1379310700011479043	2021-04-10 11:32:35 ...	2021-04-10	11:32:35	300	44196397	elonmusk
14	2021-04-10 11:23:03+...	1380798469862957057	1379310700011479043	2021-04-10 11:23:03 ...	2021-04-10	11:23:03	300	44196397	elonmusk
15	2021-04-10 11:16:58+...	1380796939151704071	1379310700011479043	2021-04-10 11:16:58 ...	2021-04-10	11:16:58	300	44196397	elonmusk
16	2021-04-10 10:46:16...	1380789209582034944	1379310700011479043	2021-04-10 10:46:16 ...	2021-04-10	10:46:16	300	44196397	elonmusk
17	2021-04-10 10:45:17...	1380788962558496768	1380788934905503744	2021-04-10 10:45:17 ...	2021-04-10	10:45:17	300	44196397	elonmusk
18	2021-04-10 10:45:10...	1380788934905503744	1380788934905503744	2021-04-10 10:45:10 ...	2021-04-10	10:45:10	300	44196397	elonmusk
19	2021-04-10 10:29:45...	1380785053542666242	1380784868729122818	2021-04-10 10:29:45 ...	2021-04-10	10:29:45	300	44196397	elonmusk
20	2021-04-10 10:29:01...	1380784868729122818	1380784868729122818	2021-04-10 10:29:01 ...	2021-04-10	10:29:01	300	44196397	elonmusk
21	2021-04-10 10:25:11+...	1380783907096850433	1380419366169153537	2021-04-10 10:25:11 ...	2021-04-10	10:25:11	300	44196397	elonmusk
22	2021-04-10 10:15:47...	1380781539647053826	1380781539647053826	2021-04-10 10:15:47 ...	2021-04-10	10:15:47	300	44196397	elonmusk
23	2021-04-10 10:12:20...	1380780669895208961	1380777569964883969	2021-04-10 10:12:20 ...	2021-04-10	10:12:20	300	44196397	elonmusk
24	2021-04-10 09:51:12...	1380775355502317568	1380138845145038858	2021-04-10 09:51:12 ...	2021-04-10	09:51:12	300	44196397	elonmusk
25	2021-04-10 09:44:23...	1380773639038111751	1380772975733923841	2021-04-10 09:44:23 ...	2021-04-10	09:44:23	300	44196397	elonmusk
26	2021-04-10 09:41:45...	1380772975733923841	1380772975733923841	2021-04-10 09:41:45 ...	2021-04-10	09:41:45	300	44196397	elonmusk
27	2021-04-10 04:07:04...	1380688750179586049	1380388899193712642	2021-04-10 04:07:04 ...	2021-04-10	04:07:04	300	44196397	elonmusk
28	2021-04-10 04:04:26...	1380688087697612802	1380388899193712642	2021-04-10 04:04:26 ...	2021-04-10	04:04:26	300	44196397	elonmusk
29	2021-04-10 03:33:12...	1380680228230426625	1380332424467341315	2021-04-10 03:33:12 ...	2021-04-10	03:33:12	300	44196397	elonmusk
30	2021-04-10 01:24:01...	1380647716959645700	1380647716959645700	2021-04-10 01:24:01 ...	2021-04-10	01:24:01	300	44196397	elonmusk
31	2021-04-10 01:21:50...	1380647165597339650	1380313600187719682	2021-04-10 01:21:50 ...	2021-04-10	01:21:50	300	44196397	elonmusk
32	2021-04-09 22:41:08...	1380606725380706304	1380388899193712642	2021-04-09 22:41:08 ...	2021-04-09	22:41:08	300	44196397	elonmusk
33	2021-04-09 22:41:01...	1380606694816800768	1380388899193712642	2021-04-09 22:41:01 ...	2021-04-09	22:41:01	300	44196397	elonmusk
34	2021-04-09 22:39:26...	1380606296945135617	1380575287516176384	2021-04-09 22:39:26 ...	2021-04-09	22:39:26	300	44196397	elonmusk
35	2021-04-09 22:14:21...	1380599983481577472	1380599009874743298	2021-04-09 22:14:21 ...	2021-04-09	22:14:21	300	44196397	elonmusk
36	2021-04-09 22:13:12...	1380599694154362882	1380388899193712642	2021-04-09 22:13:12 ...	2021-04-09	22:13:12	300	44196397	elonmusk

BUILDING THE TWEET SENTIMENT LAB...

All data is loaded

```
# elon_tweets = Path("./TweetsElonMusk.csv")
doge = Path("./doge.csv")
sp500 = Path("./sp500.csv")
tsla = Path("./tsla.csv")
sentiment = Path("./sentiment_df.csv")
```

Calculating (+) or (-) returns

```
pct_chgs = doge_df["Close"].pct_change()
neg = 0
pos = 0
for i in pct_chgs:
    if i < 0:
        neg += 1
    else:
        pos += 1
display(neg)
display(pos)
```

```
906
877
```

```
# Shift the percent change outcome up one day because we are trying to predict the price change for the next day
merged_df["pct_chg"] = merged_df["pct_chg"].shift(-1)
merged_df.dropna(inplace=True)
```

merged_df includes tsla data and sentiment data

```
X = merged_df.iloc[:, :-1]
y = merged_df.iloc[:, -1]
```

```
X = X.drop("Date", axis=1)
```

```
# change to -1 for negative and 1 for positive
for i, v in y.items():
    if v > 0:
        y[i] = 1
    elif v < 0:
        y[i] = -1
    else:
        y[i] = -1
```

BUILDING THE TWEET SENTIMENT LAB...

Run merged_df thru StandardScaler

```python
from sklearn.preprocessing import StandardScaler
```

```python
scaler = StandardScaler()
scaled_data = scaler.fit_transform(merged_df)
```

```python
scaled_df = pd.DataFrame(
    scaled_data,
    columns=merged_df.columns
)
merged_df = pd.concat([scaled_df, encoded_df], axis=1)
```

Training our data

```python
from sklearn.svm import SVC
# reg = LogisticRegression(random_state=1)
reg = SVC()
X_train, X_test, y_train, y_test = train_test_split(X, y)
```

```python
reg.fit(X_train, y_train)
```

```
SVC()
```

```python
predictions = reg.predict(X_test)
```

```python
predictions_x_train = reg.predict(X_train)
```

```python
y_test.value_counts()
```

```
-1.0    199
 1.0    158
Name: pct_chg, dtype: int64
```

```python
predictions
```

```
array([ 1., -1.,  1.,  1.,  1., -1., -1., -1., -1., -1.,  1., -1.,  1.,
        1.,  1.,  1., -1.,  1., -1., -1., -1.,  1.,  1., -1.,  1., -1.,
       -1., -1.,  1.,  1., -1., -1., -1., -1., -1., -1., -1., -1.,  1.,
       -1.,  1.,  1.,  1., -1., -1.,  1.,  1.,  1.,  1., -1.,  1., -1.,
       -1., -1., -1.,  1., -1., -1.,  1.,  1.,  1., -1.,  1.,  1., -1.,
        1., -1., -1., -1.,  1., -1., -1., -1., -1., -1., -1.,  1.,  1.,
        1.,  1., -1., -1., -1.,  1.,  1., -1., -1.,  1.,  1., -1., -1.,
        1.,  1.,  1., -1., -1., -1.,  1., -1., -1., -1., -1., -1., -1.,
       -1., -1.,  1., -1., -1., -1.,  1.,  1.,  1., -1., -1., -1., -1.,
       -1., -1.,  1.,  1.,  1., -1.,  1.,  1., -1., -1., -1., -1.,  1.,
        1., -1., -1., -1., -1.,  1., -1., -1.,  1.,  1., -1.,  1.,  1.,
        1.,  1.,  1., -1.,  1.,  1., -1., -1., -1.,  1., -1., -1.,  1.,
        1., -1., -1., -1., -1.,  1., -1.,  1., -1., -1.,  1., -1., -1.,
       -1.,  1., -1., -1.,  1., -1., -1.,  1., -1., -1., -1.,  1., -1.,
        1., -1., -1., -1., -1., -1., -1.,  1.,  1.,  1., -1., -1.,  1.,
       -1., -1., -1., -1., -1., -1.,  1.,  1., -1., -1., -1., -1., -1.,
       -1.,  1.,  1.,  1., -1., -1.,  1., -1., -1.,  1., -1.,  1.,  1.,
       -1.,  1., -1.,  1., -1., -1., -1., -1.,  1., -1.,  1., -1., -1.,
        1.,  1.,  1., -1., -1., -1.,  1., -1.,  1.,  1.,  1., -1., -1.,
        1., -1., -1., -1., -1., -1., -1.,  1., -1., -1.,  1.,  1., -1.,
       -1., -1., -1.,  1., -1., -1., -1., -1., -1., -1.,  1.,  1.,  1.,
       -1., -1., -1., -1., -1., -1.,  1., -1., -1., -1., -1., -1.,  1.,
       -1., -1., -1., -1., -1., -1.,  1.,  1.,  1., -1.,  1.,  1., -1.,
       -1., -1.,  1., -1.,  1., -1.,  1.,  1.,  1., -1., -1.,  1., -1.,
```

```python
from sklearn.metrics import classification_report
report = classification_report(y_test, predictions)
print(report)
```

```
              precision    recall  f1-score   support

        -1.0       0.67      0.73      0.70       199
         1.0       0.62      0.54      0.58       158

    accuracy                           0.65       357
   macro avg       0.64      0.64      0.64       357
weighted avg       0.65      0.65      0.65       357
```

```python
report = classification_report(y_train, predictions_x_train)
print(report)
```

```
              precision    recall  f1-score   support

        -1.0       0.72      0.81      0.76       617
         1.0       0.69      0.58      0.63       451

    accuracy                           0.71      1068
   macro avg       0.71      0.69      0.70      1068
weighted avg       0.71      0.71      0.71      1068
```

Our trained data is now scoring higher than the original test data

DIFFERENT MACHINE LEARNING MODELS

Logistic Regression

```
lr Classification Report for Tesla
-------------------------
              precision    recall  f1-score   support

         0.0       0.57      0.89      0.70       198
         1.0       0.54      0.16      0.25       158

    accuracy                           0.57       356
   macro avg       0.56      0.53      0.47       356
weighted avg       0.56      0.57      0.50       356
```

SVC

```
SVC Classification Report for Tesla
-------------------------
              precision    recall  f1-score   support

         0.0       0.70      0.80      0.75       198
         1.0       0.70      0.57      0.63       158

    accuracy                           0.70       356
   macro avg       0.70      0.69      0.69       356
weighted avg       0.70      0.70      0.69       356
```

Decision Tree Classifier

```
DTC Classification Report for Tesla
--------------------------
              precision    recall  f1-score   support

         0.0       0.64      0.69      0.66       198
         1.0       0.57      0.52      0.54       158

    accuracy                           0.61       356
   macro avg       0.61      0.60      0.60       356
weighted avg       0.61      0.61      0.61       356
```

Random Forest Classifier

```
rf Classification Report for Tesla
--------------------------
              precision    recall  f1-score   support

         0.0       0.69      0.75      0.72       198
         1.0       0.65      0.58      0.61       158

    accuracy                           0.67       356
   macro avg       0.67      0.66      0.67       356
weighted avg       0.67      0.67      0.67       356
```

OPTIMAL MODEL: SVC

```python
from sklearn.svm import SVC
# reg = LogisticRegression(random_state=1)
reg = SVC()
X_train, X_test, y_train, y_test = train_test_split(X, y)
```

```python
reg.fit(X_train, y_train)
```

```
SVC()
```

```python
predictions = reg.predict(X_test)
```

```python
predictions_x_train = reg.predict(X_train)
```

```python
y_test.value_counts()
```

```
-1.0    199
 1.0    158
Name: pct_chg, dtype: int64
```

```python
predictions
```

```
array([ 1., -1.,  1.,  1.,  1., -1., -1., -1., -1., -1.,  1., -1.,  1.,
        1.,  1.,  1., -1.,  1., -1., -1., -1.,  1.,  1., -1.,  1., -1.,
       -1., -1.,  1.,  1., -1., -1., -1., -1., -1., -1., -1., -1.,  1.,
       -1.,  1.,  1.,  1., -1., -1.,  1.,  1.,  1.,  1., -1.,  1., -1.,
       -1., -1., -1.,  1., -1., -1.,  1.,  1.,  1., -1.,  1.,  1., -1.,
        1., -1., -1., -1.,  1., -1., -1., -1., -1., -1., -1.,  1.,  1.,
        1.,  1., -1., -1., -1.,  1.,  1., -1., -1.,  1.,  1., -1., -1.,
        1.,  1.,  1., -1., -1., -1.,  1., -1., -1., -1., -1., -1., -1.,
       -1., -1.,  1., -1., -1., -1.,  1.,  1.,  1., -1., -1., -1., -1.,
       -1., -1.,  1.,  1.,  1., -1.,  1.,  1., -1., -1., -1., -1.,  1.,
        1., -1., -1., -1., -1.,  1., -1., -1.,  1.,  1., -1.,  1.,  1.,
        1.,  1.,  1., -1.,  1.,  1., -1., -1., -1.,  1., -1., -1.,  1.,
        1., -1., -1., -1., -1.,  1., -1.,  1., -1., -1.,  1., -1., -1.,
       -1.,  1., -1., -1.,  1., -1., -1.,  1., -1., -1., -1.,  1., -1.,
        1., -1., -1., -1., -1., -1., -1.,  1.,  1.,  1., -1., -1.,  1.,
       -1., -1., -1., -1., -1., -1.,  1.,  1., -1., -1., -1., -1., -1.,
       -1.,  1.,  1.,  1., -1., -1.,  1., -1., -1.,  1., -1.,  1.,  1.,
       -1.,  1., -1.,  1., -1., -1., -1., -1.,  1., -1.,  1., -1., -1.,
        1.,  1.,  1., -1., -1., -1.,  1., -1.,  1.,  1.,  1., -1., -1.,
        1., -1., -1., -1., -1., -1., -1.,  1., -1., -1.,  1.,  1., -1.,
       -1., -1., -1.,  1., -1., -1., -1., -1., -1., -1.,  1.,  1.,  1.,
       -1., -1., -1., -1., -1., -1.,  1., -1., -1., -1., -1., -1.,  1.,
       -1., -1., -1., -1., -1., -1.,  1.,  1.,  1., -1.,  1.,  1., -1.,
       -1., -1.,  1., -1.,  1., -1.,  1.,  1.,  1., -1., -1.,  1., -1.,
        1., -1., -1.,  1., -1.,  1., -1., -1.,  1., -1., -1.,  1., -1.,
```

```python
from sklearn.metrics import classification_report
report = classification_report(y_test, predictions)
print(report)
```

```
              precision    recall  f1-score   support

        -1.0       0.67      0.73      0.70       199
         1.0       0.62      0.54      0.58       158

    accuracy                           0.65       357
   macro avg       0.64      0.64      0.64       357
weighted avg       0.65      0.65      0.65       357
```

```python
report = classification_report(y_train, predictions_x_train)
print(report)
```

```
              precision    recall  f1-score   support

        -1.0       0.72      0.81      0.76       617
         1.0       0.69      0.58      0.63       451

    accuracy                           0.71      1068
   macro avg       0.71      0.69      0.70      1068
weighted avg       0.71      0.71      0.71      1068
```

Our top performing Machine Learning model is SVC

71% accuracy score

ATTEMPT TO FINE TUNE:

Randomized Search

```
{'kernel': 'rbf', 'gamma': 0.1, 'degree': 1, 'C': 10}
              precision    recall  f1-score   support

        -1.0       0.66      0.72      0.69       199
         1.0       0.60      0.54      0.57       158

    accuracy                           0.64       357
   macro avg       0.63      0.63      0.63       357
weighted avg       0.64      0.64      0.64       357
```

```
param_grid = {'C': [0.1, 1, 10, 100, 1000],
              'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
              'kernel': ['sigmoid', 'rbf'],
              'degree': [1, 2, 3, 4]
             }
```

Halving Grid Search:

```
{'C': 1, 'degree': 3, 'gamma': 0.1, 'kernel': 'rbf'}
              precision    recall  f1-score   support

        -1.0       0.65      0.72      0.69       199
         1.0       0.60      0.52      0.56       158

    accuracy                           0.63       357
   macro avg       0.63      0.62      0.62       357
weighted avg       0.63      0.63      0.63       357
```

ATTEMPT TO FINE TUNE CONT. :

Grid Search

```
{'C': 10, 'degree': 1, 'gamma': 0.1, 'kernel': 'rbf'}
              precision    recall  f1-score   support

        -1.0       0.66      0.72      0.69       199
         1.0       0.60      0.54      0.57       158

    accuracy                           0.64       357
   macro avg       0.63      0.63      0.63       357
weighted avg       0.64      0.64      0.64       357
```

```
param_grid = {'C': [0.1, 1, 10, 100, 1000],
              'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
              'kernel': ['sigmoid', 'rbf'],
              'degree': [1, 2, 3, 4]
             }
```

Halving Random Search:

```
{'kernel': 'rbf', 'gamma': 0.1, 'degree': 3, 'C': 1}
              precision    recall  f1-score   support

        -1.0       0.65      0.72      0.69       199
         1.0       0.60      0.52      0.56       158

    accuracy                           0.63       357
   macro avg       0.63      0.62      0.62       357
weighted avg       0.63      0.63      0.63       357
```

BUT DOES HE AFFECT MORE THAN TESLA?

SVC Model on 'DOGE'

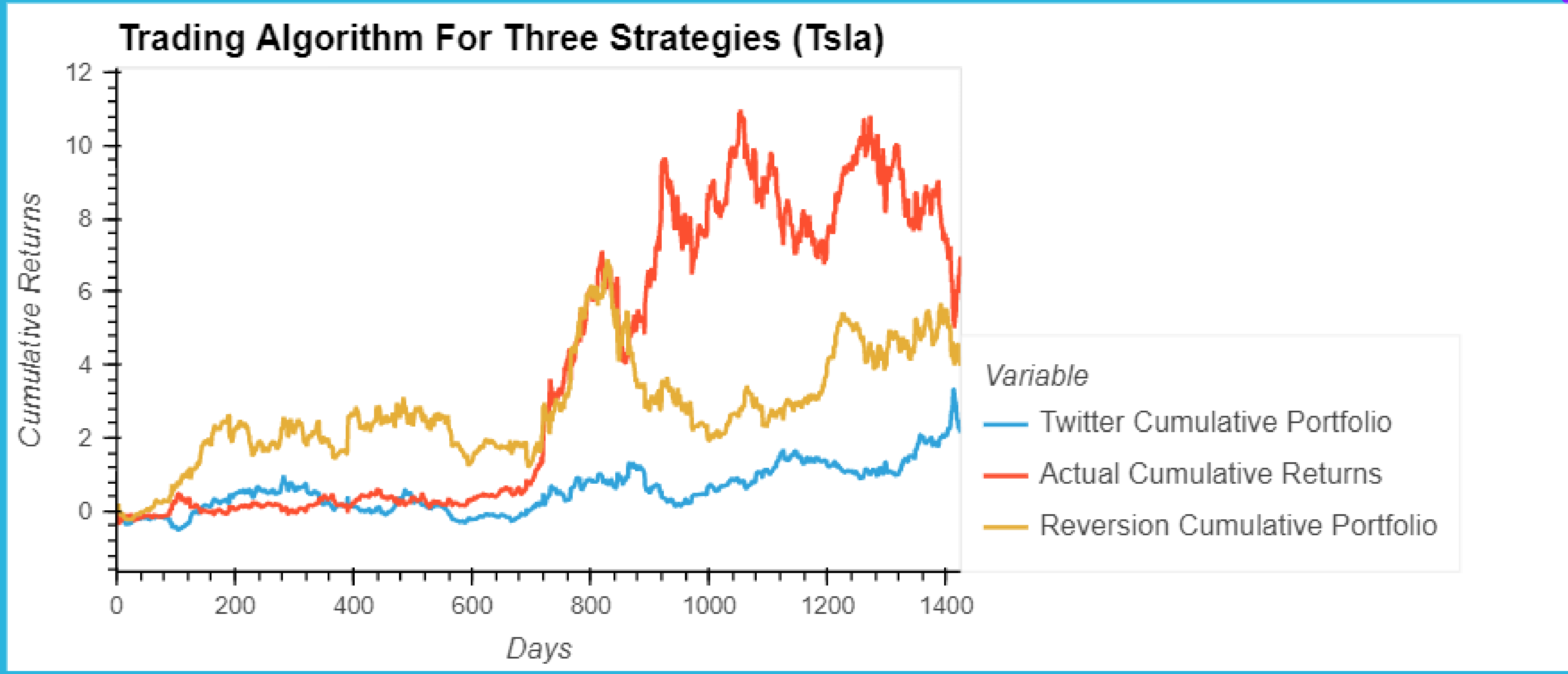
	precision	recall	f1-score	support
-1.0	0.66	0.73	0.69	163
1.0	0.54	0.46	0.50	113
accuracy			0.62	276
macro avg	0.60	0.60	0.60	276
weighted avg	0.61	0.62	0.61	276

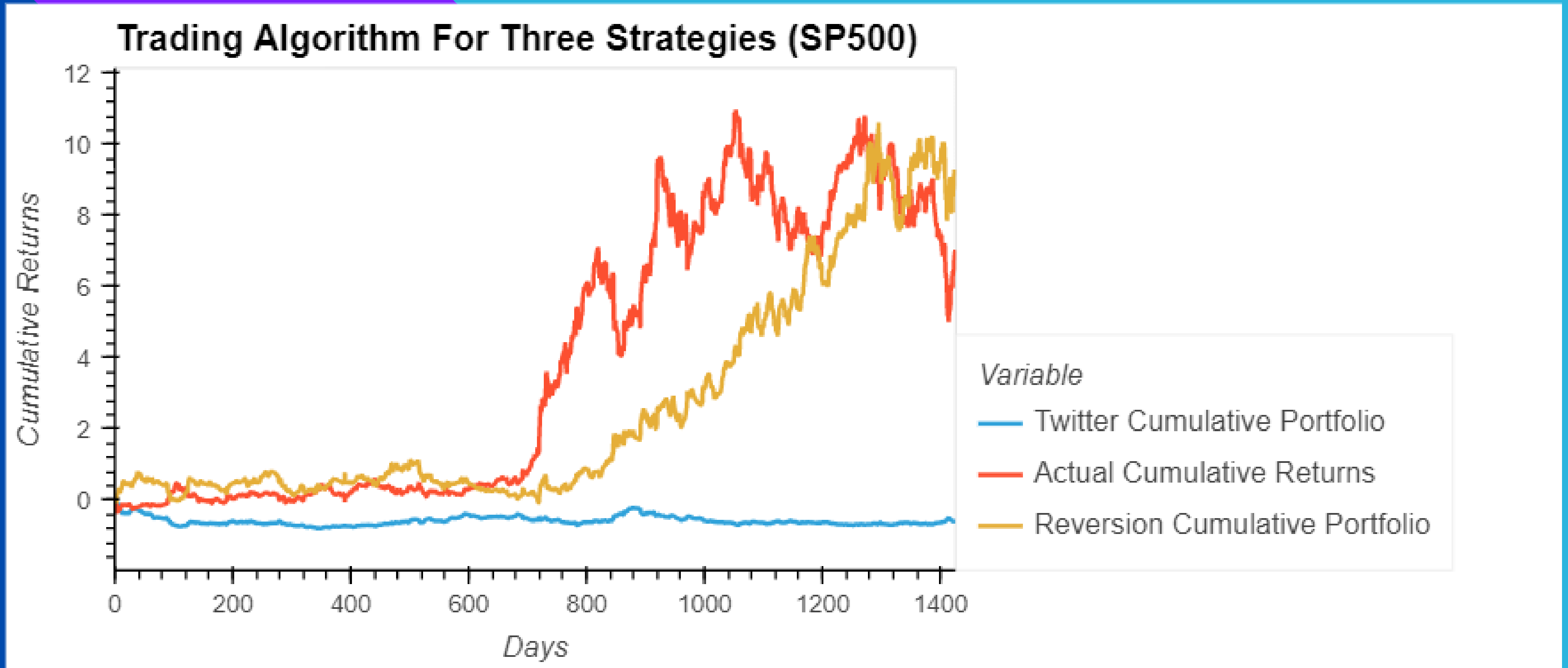
SVC Model on 'SPY'

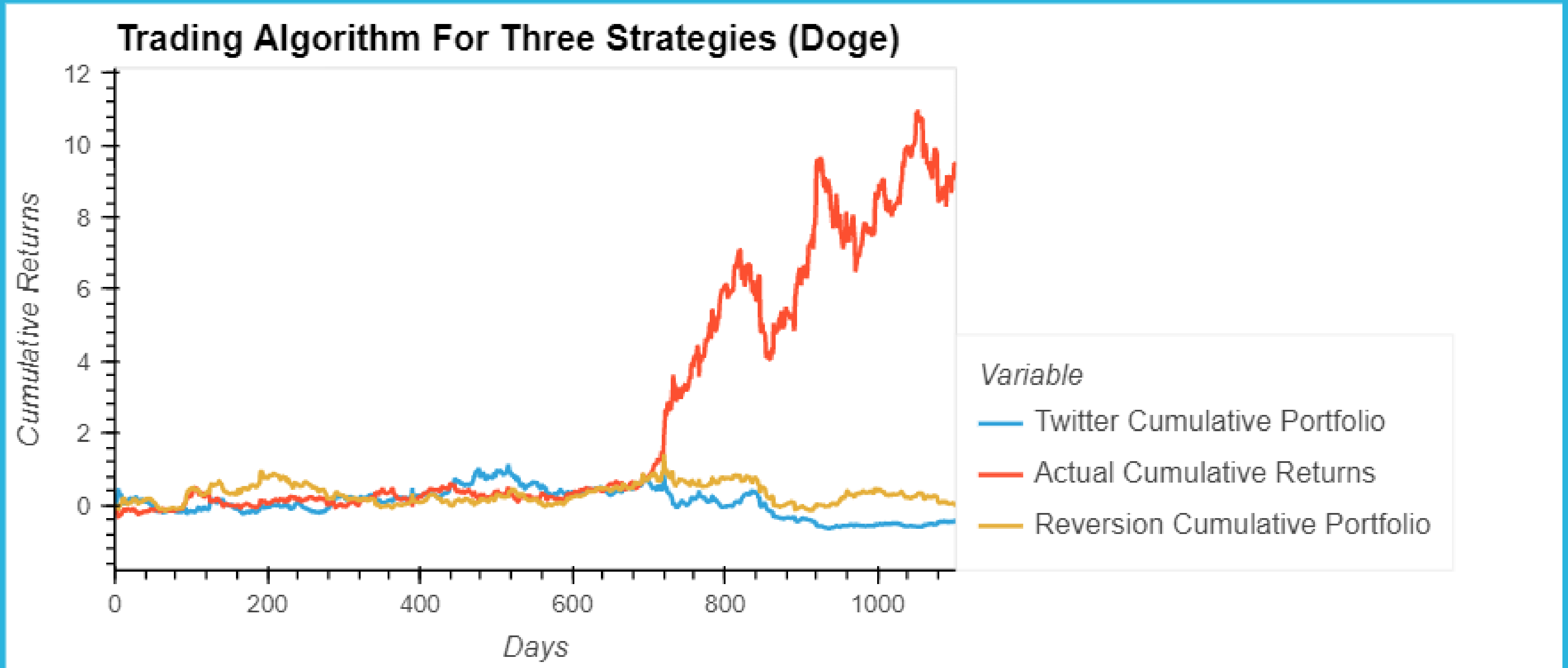
	precision	recall	f1-score	support
-1.0	0.69	0.83	0.75	224
1.0	0.57	0.38	0.46	133
accuracy			0.66	357
macro avg	0.63	0.60	0.61	357
weighted avg	0.65	0.66	0.64	357

ALGORITHMIC TRADING:

Using the Twitter Signal and Mean Reversion signal to conduct our trades for TSLA were less effective than longing TSLA over the same time period.







RESULTS:

Does a correlation exist between Elon's tweets and stock market reactions ?

- YES Elon's tweets have a greater correlation to stock prices that what a simple binary example would infer

Is it possible to predict whether or not a stock with rise/fall/remain with the sentiment of his tweets?

- YES Using our SVC machine learning model we have 70+% accuracy rate of correctly predicting the market reaction

Can we design a stock purchase method based on Elon's tweets?

- YES Using our SVC machine learning model we have 70+% accuracy rate of correctly predicting the market reaction

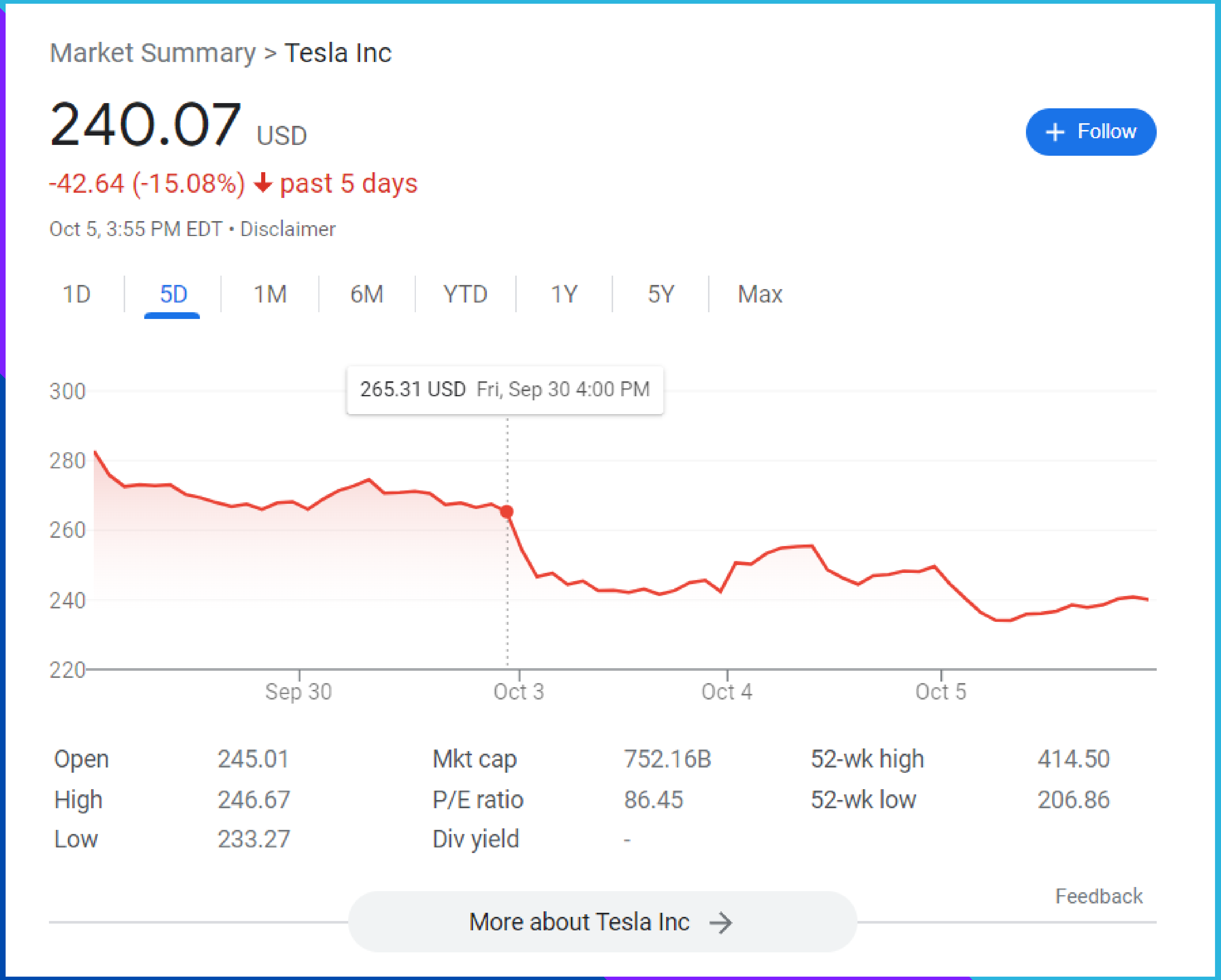
TESTING AGAINST YESTERDAY'S TWEETS...

*Timeframe:
October 2, 2022

Search this file...

1	datetime	date	time	Tweet	Retweets	Quote Tweets	Likes	score_tag	confidence
2	2022-10-02 11:49:00+00:00	2022-10-02	11:49:00	Do not let ancient grudge break to new mutiny	7619	623	93000	P	92
3	2022-10-02 16:03:00+00:00	2022-10-02	16:03:00	Prufrock II emerging from the ground	3584	349	59300	NONE	100
4	2022-10-02 16:47:00+00:00	2022-10-02	16:47:00	The embarrassing hose down photos were highly motivating tbh	178	58	5817	P	100
5	2022-10-02 16:27:00+00:00	2022-10-02	16:27:00	Tunneled under the road from one property to another. This is to test the new machine	223	27	5458	NONE	100
6	2022-10-02 16:26:00+00:00	2022-10-02	16:26:00	Unfortunately, snail is still faster, but TBC might beat snail by end of 2024	179	28	4390	N	100
7	2022-10-02 15:08:00+00:00	2022-10-02	15:08:00	But between now and then, we do actually need to work hard	229	43	3985	NONE	100
8	2022-10-02 15:07:00+00:00	2022-10-02	15:07:00	I don't care about boosting the stock, but the economic implications are obvious	842	162	15600	N	92
9	2022-10-02 11:32:00+00:00	2022-10-02	11:32:00	So many talented Russians in America	139	25	3354	P+	100
10	2022-10-02 11:22:00+00:00	2022-10-02	11:22:00	Customer experience suffers when there is an end of quarter rush. Steady as she goes is the right move.	328	51	4833	NEU	94
11	2022-10-02 09:46:00+00:00	2022-10-02	9:46:00	Master Plan (mé·nage à) Trois is all about tonnage	357	43	6073	NONE	100
12	2022-10-02 08:04:00+00:00	2022-10-02	8:04:00	Thanks, we have a great team	264	22	6808	P+	100
13	2022-10-02 08:03:00+00:00	2022-10-02	8:03:00	Beautiful	494	47	15200	P	100
14	2022-10-02 04:41:00+00:00	2022-10-02	4:41:00	Exactly	250	35	7413	NONE	100

HOW WAS THE MODEL FOR OCT 2? TESLA:

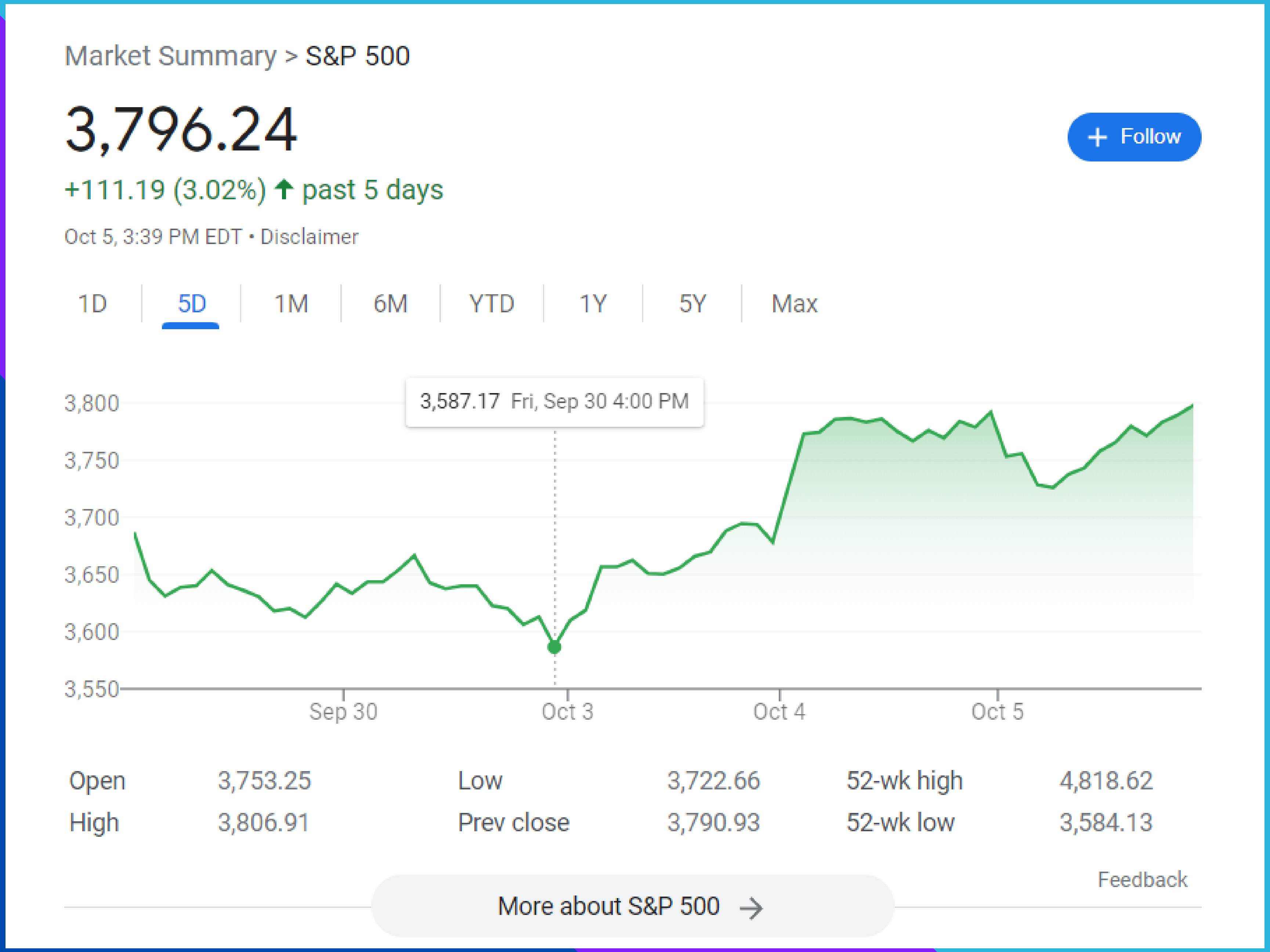


Prediction for October 2nd, 2022: Sell

(Good prediction)

HOW WAS THE MODEL FOR OCT 2?

S&P 500:



Prediction for October 2nd, 2022: Sell

(Bad prediction)

HOW WAS THE MODEL FOR OCT 2? DOGECOIN:

Prediction for October 2nd, 2022: Sell

(Bad prediction)

RESULTS:

Do Elons tweets affect companies other than TSLA? (TEST: 'DOGE' + 'TWTR')

- YES

Elon's tweets have a greater correlation to stock prices that what a simple binary example would infer -EVEN for stocks outside of TSLA

Is it possible to predict whether or not a stock with rise/fall/remain with the sentiment of his tweets?

- YES

Using our SVC machine learning model we have 60+% accuracy rate of correctly predicting the market reaction

However - this model is consistently less profitable than just maintaining a long position in the market

FIND MORE ON GITHUB

https://github.com/smruthid/ElonTweetAnalysis