Driverless Car Sentiment Analysis and Impact

Kyle Markwardt

Objective

Question:

What are the downstream impacts of paper-of-record headlines on the burgeoning autonomous vehicle industry?

Supposition is that the stock market reacts to headlines.

Goals:

Conduct sentiment analysis on Autonomous Vehicle headlines

Create a model to predict impact on self-driving company stock prices based on volume and sentiment of headlines

Judge various company sensitivities to headlines about self driving

Hurdles and Limitations

Stock data:

- Many main players owned by much larger companies (Cruise:GM, Waymo:GOOG)
- Many failed or never went public
- Others are partial plays (TSLA, NVDA, APTV)
- Pure plays have short public history (AUR, TSPH)

News:

- Tech Crunch has no API
- NewsAPI.org includes sources like TC and full text, but is limited to 1 month search
- NYT limited to headline and snippet

Data Sources

Stock data

- Yfinance

News

- NYT API
    - 500 articles for Autonomous Driving back to Jan 2016

Labeled Training Data [Twitter self driving sentiment]

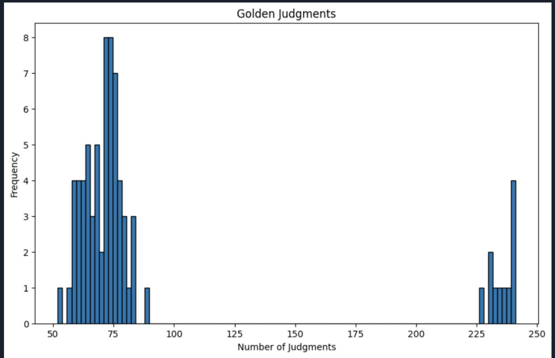
- https://data.world/crowdflower/sentiment-self-driving-cars
- 7000+ rows, created 2015

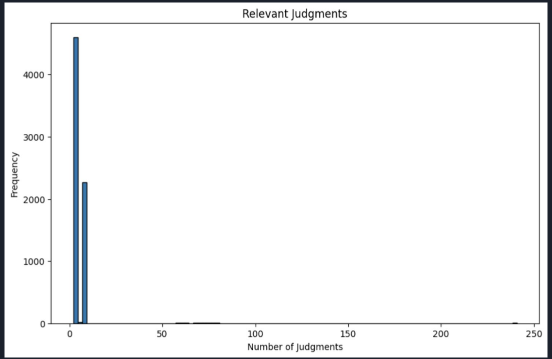
Plan A - Create Sentiment Model

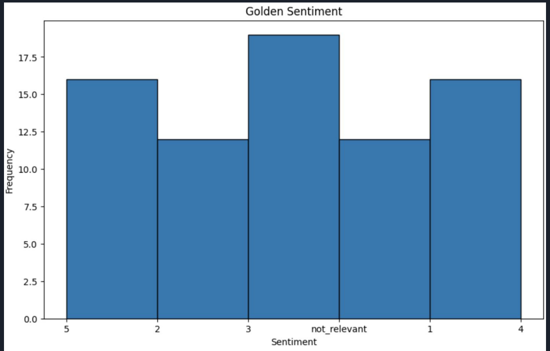
1. Create sentiment model on labeled twitter data
    a. Convert 1-5 ratings to pos/neg
    b. (drop 3s)

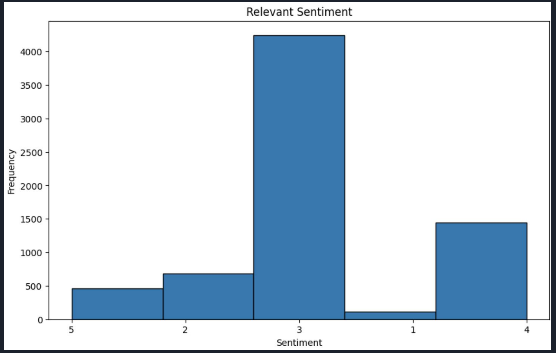
2. Predict headline sentiment with that model
    a. Label smaller (n=20) set of the headlines and evaluate model performance

Examining the twitter data:

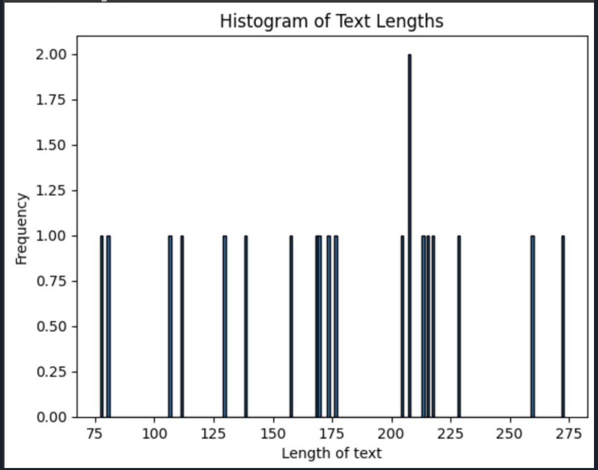








```
Maximum length: 272
Minimum length: 77
```



Result of Plan A:

Tokenized texts, max words 250, Embedding

- Dense Neural Network
- Bidirectional
- Dropout (overfitting was a huge problem with the small dataset
- About 73%, not terrible…

```
Epoch 10/10
54/54 [==============================] - 65s 1s/step - loss: 0.0647 - accuracy: 0.9693 - val_loss: 1.4321 - val_accuracy: 0.7292
17/17 [==============================] - 2s 134ms/step - loss: 1.4294 - accuracy: 0.7296
Test accuracy: 0.729629635810852
```

But when I ran the headlines evaluation set through it… 35%

```
1/1 [==============================] - 0s 158ms/step - loss: 4.2869 - accuracy: 0.3500
Test accuracy: 0.3499999940395355
```

Plan B: Pre-trained Sentiment Model

Decided to use a tried and trusted sentiment analysis model and work on the stock price correlation

```
sentiment_model = pipeline('sentiment-analysis', model='distilbert-base-uncased-finetuned-sst-2-english')
```

Voila!

```
Accuracy: 0.8
Classification Report:
              precision    recall  f1-score   support

          -1       0.80      0.80      0.80        10
           1       0.80      0.80      0.80        10

    accuracy                           0.80        20
   macro avg       0.80      0.80      0.80        20
weighted avg       0.80      0.80      0.80        20
```

Plan B continued

Features: (1) Sum of article sentiment aggregated by day (net sentiment per day)

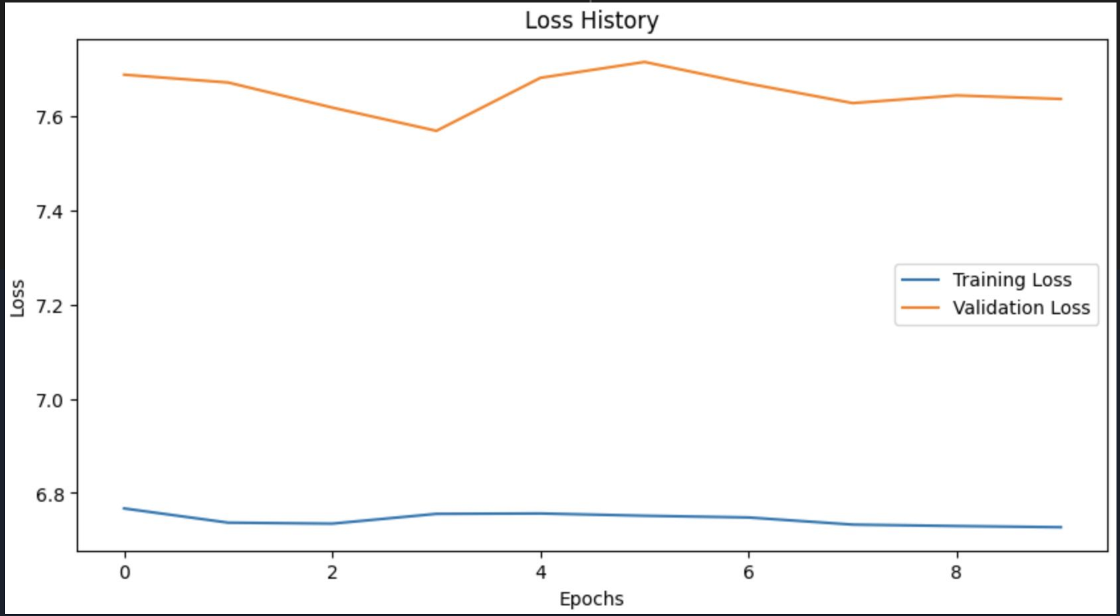
- e.g. [(0.9564) x (-1)] + [0.7854 x (1)]

Multiple models:

- Stock price on same day
- Stock price on next day
- Aggregate stock and sentiment by week
- Loop through multiple stocks

RNN with LSTM
Loss: MSE b/c of continuous nature of pct_change

```python
# Build and train LSTM model
def train_lstm(X_train, y_train, X_test, y_test):
    model = Sequential()
    model.add(LSTM(128, return_sequences=True, input_shape=(X_train.shape[1], 1)))
    model.add(LSTM(128))
    model.add(Dropout(0.2))
    model.add(Dense(1))
    model.compile(optimizer=Adam(learning_rate=0.001), loss='mean_squared_error')
    history = model.fit(X_train, y_train, batch_size=32, epochs=10, validation_data=(X_test, y_test), verbose=1)
    loss = model.evaluate(X_test, y_test)
    print(f'Test Loss: {loss}')
    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)
    print(f'Mean Squared Error: {mse}')
    print(f'R^2 Score: {r2}')
    return history, model, mse, r2
```



Plan B.2 - Skip sentiment analysis

What if I skipped the 0/1 or -1/1 step and embedded the headline text but otherwise did the same thing?

Keep more information in the input, plus more features, NN makes more sense…

Atrocious r^2 values

	ticker	shift_days	mse	r2
0	TSLA	0	0.904176	-0.073068
1	TSLA	1	5.381370	-3.256683
2	TSLA	weekly	59.469277	-0.004637
3	AUR	weekly	38.998601	0.010772
4	GOOG	0	2.808688	-0.679575
5	GOOG	1	2.193855	-0.200059
6	GOOG	weekly	14.118953	-0.011923
7	GM	0	2.423245	-0.324621
8	GM	1	7.339129	-0.813074
9	GM	weekly	21.313594	-0.024720
10	NVDA	0	14.398336	-0.060567
11	NVDA	1	6.202351	-0.032551
12	NVDA	weekly	29.647777	-0.018149
13	APTV	0	4.369776	-0.951300
14	APTV	1	3.861646	-5.950820

Plan B.3 - Random Forest

Models not returning anything useful

- Models are bad?
- Data is bad?
- No correlation?

Not having any luck with NN, try random forest model

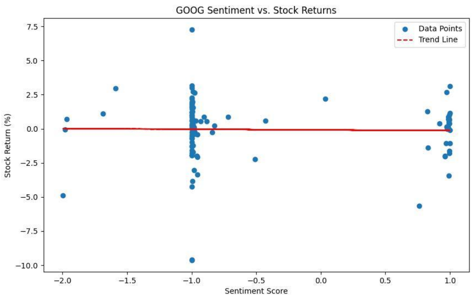
```
def train_random_forest(X_train, y_train, X_test, y_test):
    model = RandomForestRegressor(n_estimators=100, random_state=777)
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)
    print(f'Mean Squared Error: {mse}')
    print(f'R^2 Score: {r2}')
    return model, mse, r2
```

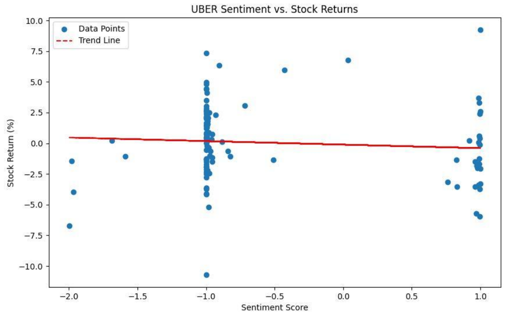
	ticker	shift_days	mse	r2
0	TSLA	0	0.871382	-0.034149
1	TSLA	1	5.434484	-3.298697
2	TSLA	weekly	62.870552	-0.062096
3	AUR	weekly	27.995865	0.289865
4	GOOG	0	2.096480	-0.253680
5	GOOG	1	1.789170	0.021307
6	GOOG	weekly	14.382935	-0.030843
7	GM	0	2.458829	-0.344072
8	GM	1	9.256998	-1.286869
9	GM	weekly	22.055687	-0.060398
10	NVDA	0	12.602407	0.071719
11	NVDA	1	6.509131	-0.083623
12	NVDA	weekly	31.669994	-0.087594
13	APTV	0	3.905757	-0.744095
14	APTV	1	3.478999	-5.262069
15	APTV	weekly	19.823030	-0.022911
16	TSPH	weekly	236.701643	-3.039999

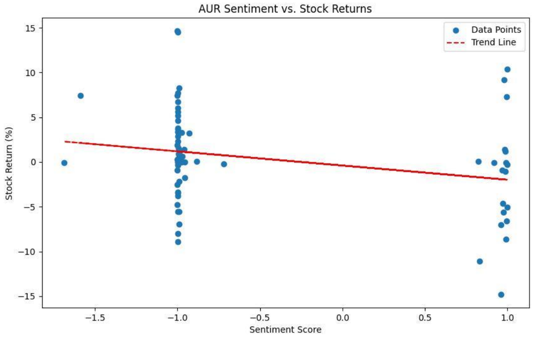
Hang on…

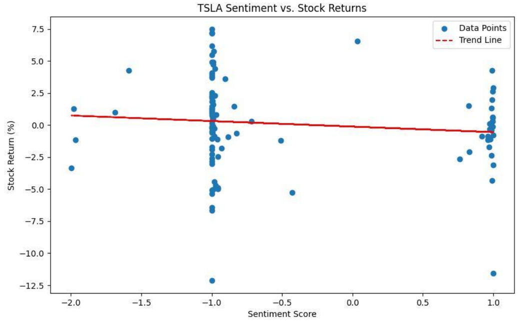
No Correlation !!

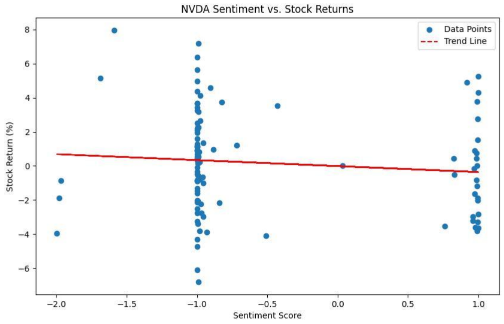
Forgot simple check

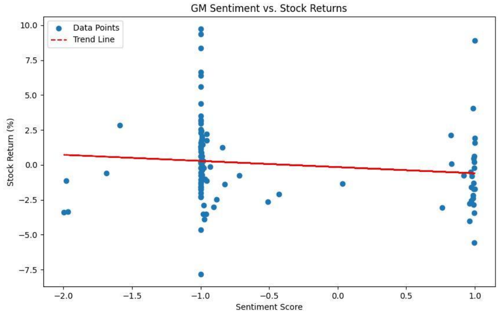


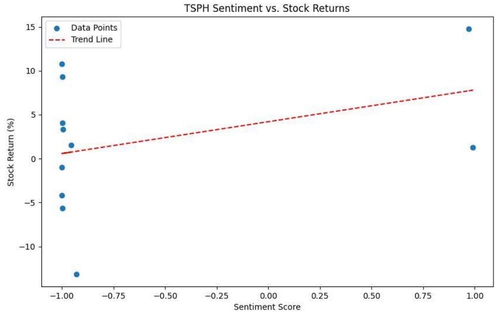










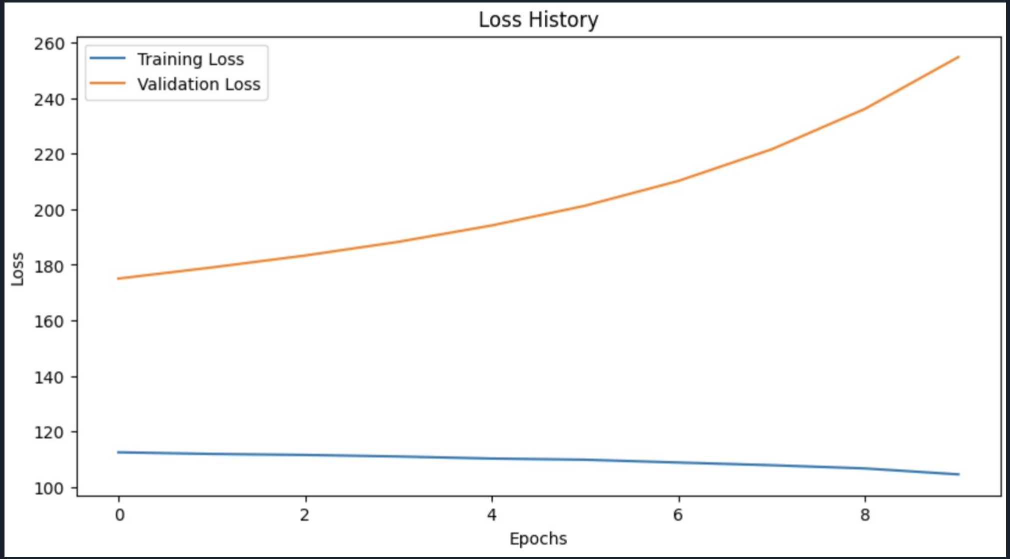


Plan B.3 - Sentiment Lagged in Time LSTM

Because the no correlation graphs were on same day, I attempted to engineer features which were sentiment lagging by up to 7 days

Same types of results:

	ticker	mse	r2
0	TSLA	6.678150	-0.033874
1	AUR	24.779505	0.013853
2	GOOG	3.032027	-0.012493
3	GM	4.464870	-0.051762
4	NVDA	7.637391	0.023503
5	APTV	3.667235	-0.051548
6	TSPH	254.823822	NaN



Plan C - “Blood in the Water”

Last ditch effort to see if the stock prices led articles sentiment instead.

Engineered lagging price pct_change features by 2 weeks for 5 stocks

LSTM model

```
Accuracy: 0.6428571428571429
Confusion Matrix:
[[54  0]
 [30  0]]
Classification Report:
              precision    recall  f1-score   support

           0       0.64      1.00      0.78        54
           1       0.00      0.00      0.00        30

    accuracy                           0.64        84
   macro avg       0.32      0.50      0.39        84
weighted avg       0.41      0.64      0.50        84
```

```
         date                                               text  \
18 2016-04-12  Facebook’s Developer Conference Kicks Off On T...
19 2016-04-13  Ford’s Planned New Headquarters Borrow Some Si...
20 2016-04-22  Despite Strong Earnings, G.M. Has Much to Prov...
21 2016-04-27  Ford and Google Team Up to Support Driverless ...
22 2016-05-03  Google to Get Fiat Chrysler Minivans for Self-...

   sentiment_label  sentiment_score  sentiment  sentiment_binary      APTV  \
18        POSITIVE         0.686108          1                 1 -0.371802
19        NEGATIVE         0.677211         -1                 0  5.625432
20        NEGATIVE         0.997127         -1                 0 -0.355260
21        NEGATIVE         0.871993         -1                 0 -1.091558
22        POSITIVE         0.997987          1                 1 -0.732601

          GM     GOOGL      NVDA  ...  APTV_lag_5  APTV_lag_6  APTV_lag_7  \
18  0.609544  0.895004 -0.111485  ...    2.290081   -2.219791   -0.943398
19  3.601486  0.993043  2.511165  ...    3.284344    2.290081   -2.219791
20 -1.469686 -5.414102 -0.384508  ...    2.793622    3.284344    2.290081
21 -0.093193 -0.539034  2.275224  ...   -3.881121    2.793622    3.284344
22 -1.574803 -0.835655 -1.194445  ...   -3.881121   -3.881121    2.793622

   APTV_lag_8  APTV_lag_9  APTV_lag_10  APTV_lag_11  APTV_lag_12  \
18  -2.808986   -1.040630    -1.173528    -1.945627    -5.598187
19  -0.943398   -2.808986    -1.040630    -1.173528    -1.945627
20  -2.219791   -0.943398    -2.808986    -1.040630    -1.173528
21   2.290081   -2.219791    -0.943398    -2.808986    -1.040630
22   3.284344    2.290081    -2.219791    -0.943398    -2.808986

   APTV_lag_13  APTV_lag_14
18    -2.657364    -2.774135
19    -5.598187    -2.657364
20    -1.945627    -5.598187
21    -1.173528    -1.945627
22    -1.040630    -1.173528
```

Lessons Learned

<u>LARGER DATA SETS</u>

- Most of my problems really seem to have stemmed from datasets being too small. It's hard to do ML projects on a time crunch with "go and get it" data

Clear Plan

- Genuinely went in with what I thought was a clear plan. But what exactly I wanted models to predict was not concrete and that led to some confusion on what models or parameters to select/try.

Don't forget early data checks

- By the time I realized how futile my effort was, it was too late to switch horses. Could have saved myself a lot of time and compute.

Loose ends

Would still like to see the downstream impact of headline sentiment on SOCIAL sentiment

- Didn’t have time + Twitter ‘pull’ access API is $100/mo

Would have liked to pull articles/headlines/text from more sources to build a much larger and more diverse corpus in the same time window