Tweet Sentiment's Impact on Stock Price

Erin Amoueyan

June 2023

THE PROBLEM

- The challenge of quantifying the impact of tweet sentiment on stock returns due to the complex and dynamic nature of financial markets.

- Without a clear understanding of the influence of tweet sentiment on stock returns, investors may miss valuable opportunities or make suboptimal investment decisions.

THE SOLUTION

- Natural Language Processing (NLP) techniques and LSTM (Long Short-Term Memory) model to:
    - Analyze and extract sentiment from a large corpus of tweets related to specific stocks or financial events
    - Predict the impact of tweet sentiment's labels on stock return

DATA - Tweet Sentiments

14 Columns

1,395,450 Rows

	Unnamed: 0	TWEET	STOCK	DATE	LAST_PRICE	1_DAY_RETURN	2_DAY_RETURN	3_DAY_RETURN	7_DAY_RETURN
0	0	RT @robertoglezcano: @amazon #Patents Show Fl...	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	NaN	Amazon	31/01/2017	823.48	0.008379	0.014924	0.014924	-0.001263	3.137196e+06
2	1	@FAME95FM1 Jamaicans make money with @Payoneer...	PayPal	31/01/2017	39.780000	0.002011	0.012318	0.012318	5.480141e-02
[illegible]	[illegible]	@CBSi Jamaicans [illegible] @Pay...	[illegible]	[illegible]1/2017	39.780000	0.002011	0.012318	0.012318	5.480141e-02
4	3	@Hitz92fm Jamaicans make money with @Payoneer ...	PayPal	31/01/2017	39.780000	0.002011	0.012318	0.012318	5.480141e-02

to expand output; double click to hide output

Data Source: https://www.kaggle.com/datasets/thedevastator/tweet-sentiment-s-impact-on-stock-returns

DATA - Stock Prices

8 Columns

13,470 Rows

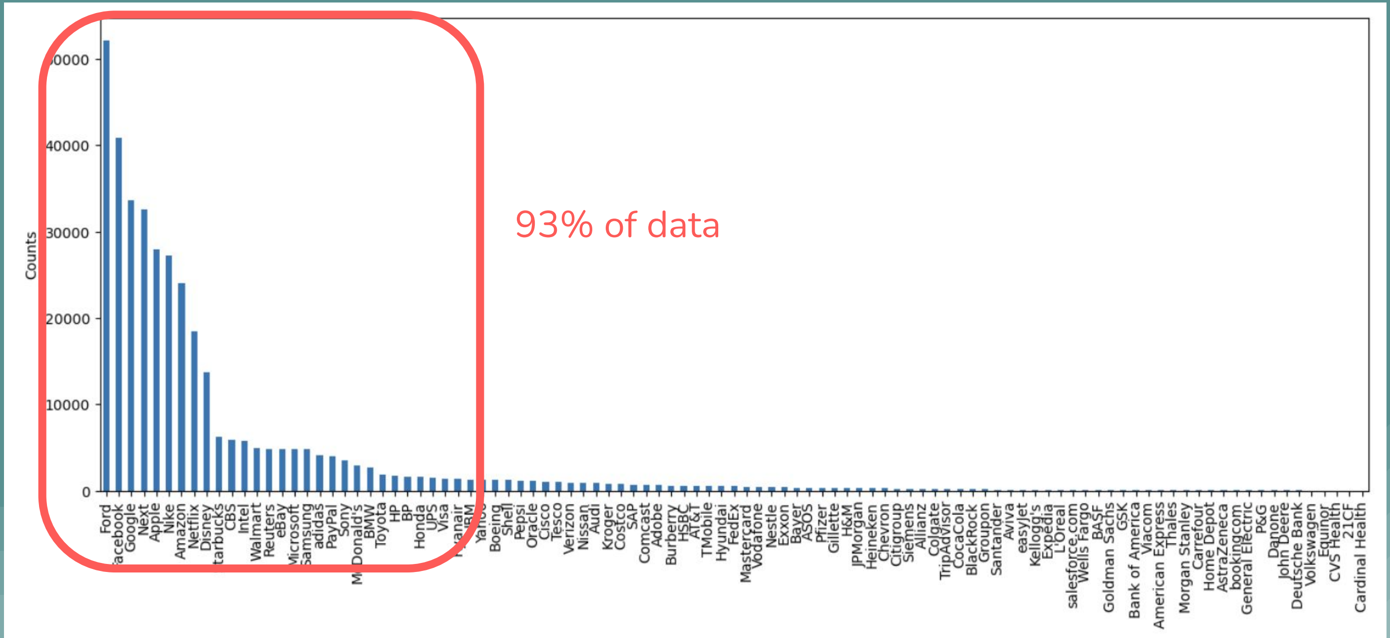
	Date	Open	High	Low	Close	Adj Close	Volume	ticker
0	2017-01-20	12.45	12.48	12.31	12.36	9.159223	29270000	F
1	2017-01-23	12.35	12.38	12.22	12.31	9.122169	31670700	F
2	2017-01-24	12.35	12.61	12.34	12.61	9.344479	34625200	F
3	2017-01-25	12.71	12.80	12.64	12.79	9.477868	46747800	F
4	2017-01-26	12.65	12.68	12.35	12.37	9.166630	55672800	F
5	2017-01-27	12.48	12.54	12.38	12.49	9.255554	34613900	F
6	2017-01-30	12.46	12.46	12.28	12.37	9.166630	39254200	F
7	2017-01-31	12.31	12.39	12.19	12.36	9.159223	46974500	F
8	2017-02-01	12.45	12.58	12.22	12.32	9.129580	44396800	F
9	2017-02-02	12.30	12.37	12.23	12.28	9.099936	29035400	F

Data Source: Yahoo Finance API

DATA WRANGLING
AND
EXPLORATORY DATA ANALYSIS

TWEET SENTIMENTS

- Source of data: Kaggle
- Data duration: 01/31/2017 - 10/31/2018
- Number of unique stocks= 101
- Number of stocks considered for further evaluation = 30
- Percentage of data related to the top 30 stocks = 93%
- Checked for missing values
- Checked for duplicate values



STOCK PRICES

- Source of data= Yahoo Finance API
- Data duration: 01/31/2017 - 10/31/2018
- Added Stock Tickers

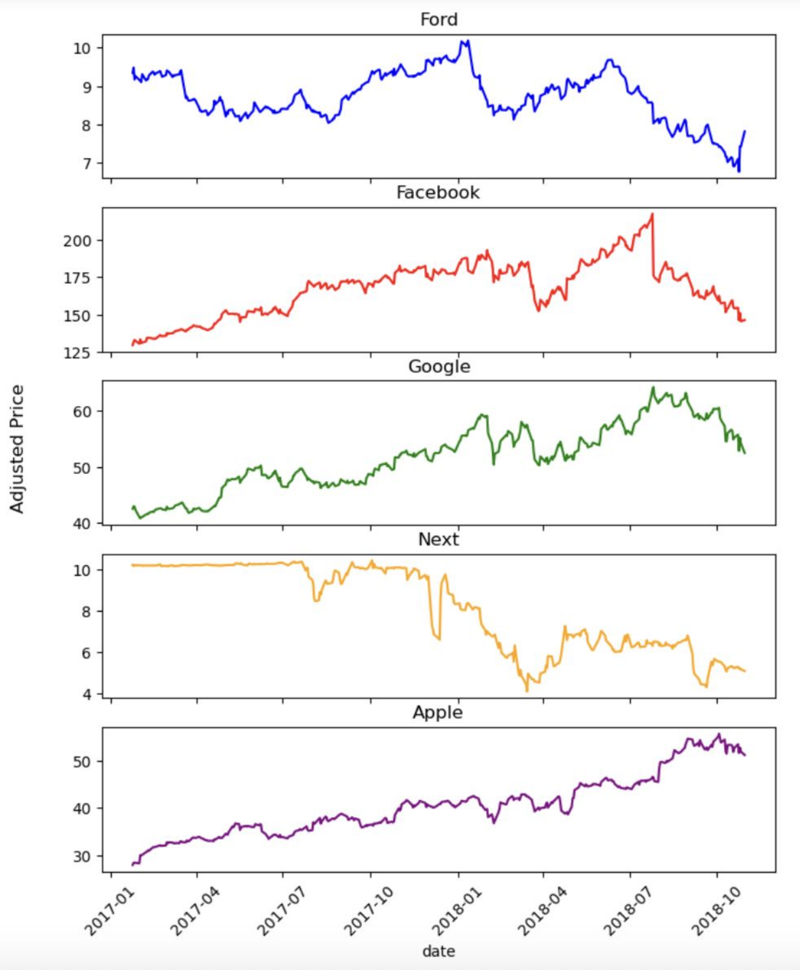
- Removed the outliers
- Calculated 1-day stock return based on "adjusted-close" price

	date	open	high	low	close	adj_close	volume	ticker	company_name	1_day_return
0	2017-01-20	30.112499	30.112499	29.932501	30.000000	27.993986	130391600	AAPL	Apple	NaN
1	2017-01-23	30.000000	30.202499	29.942499	30.020000	28.012650	88200800	AAPL	Apple	NaN
2	2017-01-24	29.887501	30.025000	29.875000	29.992500	27.986988	92844000	AAPL	Apple	-0.000917
3	2017-01-25	30.105000	30.525000	30.070000	30.469999	28.432562	129510400	AAPL	Apple	0.015671
4	2017-01-26	30.417500	30.610001	30.400000	30.485001	28.446554	105350400	AAPL	Apple	0.000492
5	2017-01-27	30.535000	30.587500	30.400000	30.487499	28.448895	82251600	AAPL	Apple	0.000082
6	2017-01-30	30.232500	30.407499	30.165001	30.407499	28.374241	121510000	AAPL	Apple	NaN
7	2017-01-31	30.287500	30.347500	30.155001	30.337500	28.308924	196804000	AAPL	Apple	-0.002307
8	2017-02-01	31.757500	32.622501	31.752501	32.187500	30.035215	447940000	AAPL	Apple	0.057476
9	2017-02-02	31.995001	32.347500	31.945000	32.132500	29.983891	134841600	AAPL	Apple	-0.001712
10	2017-02-03	32.077499	32.297501	32.040001	32.270000	30.112206	98029200	AAPL	Apple	0.004261

EXPLORATORY DATA ANALYSIS

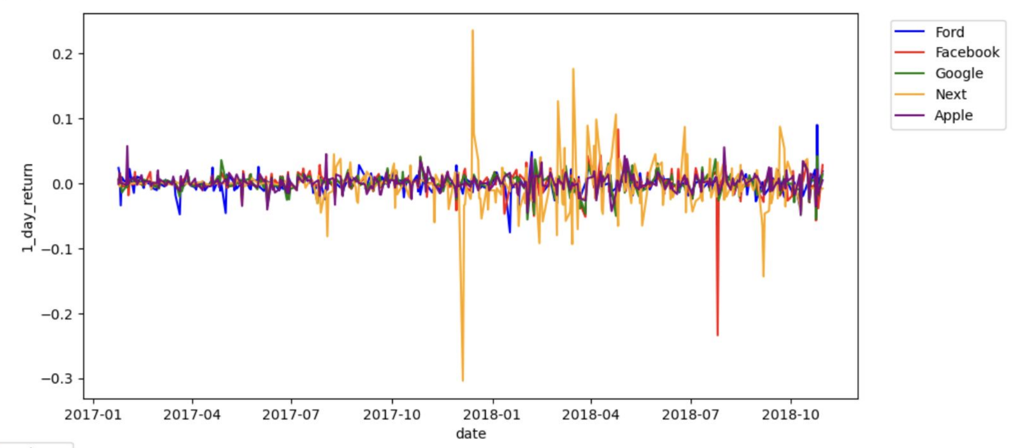
- Source of data= Yahoo Finance API
- Top 5 companies:
  - Ford
  - Facebook
  - Google
  - Next
  - Apple

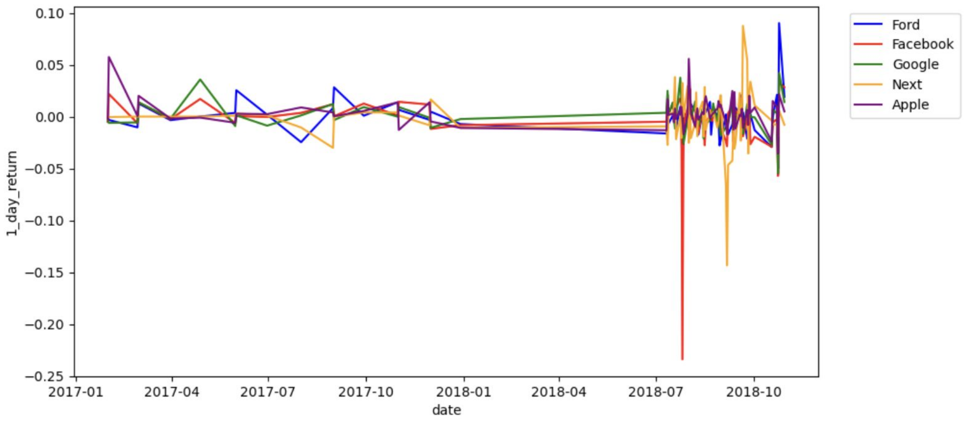
Adjusted Price



EXPLORATORY DATA ANALYSIS

1-day Stock Return
before merging Yahoo Finance
Data and Tweet Data





1-day Stock Return
after merging Yahoo Finance
Data and Tweet Data

PREPROCESSING AND MODELING

1. Analyzing Stock Sentiments

2. Time-Series Forecasting:

Predicting Stock Prices

3. Time-Series Forecasting:

Predicting Stock Returns

Analyzing Stock Sentiments

Data Preprocessing

- Natural Language Processing (NLP) techniques
  - **tokenization** : to split texts into individual tokens (words or characters)
  - removing **stop words**
  - handling **emoticons**
  - **lemmatization** to reduce the dimensionality of the text data
  - **sequency** : converts the text data into sequences of integers.
  - **padding**: to make all sequences the same length

Data Preprocessing

- Assign sentiment labels (e.g., positive, negative, neutral) to each tweet
- “SentimentIntensityAnalyzer”, built-in tool in the Natural Language Toolkit (NLTK) library to find sentiment labels
- Sentiment scores >= 0.5 —> “positive”
- Sentiment scores <= -0.5 —> “negative”
- -0.5 < Sentiment scores < 0.5 —> “neutral”

Check for imbalance data:

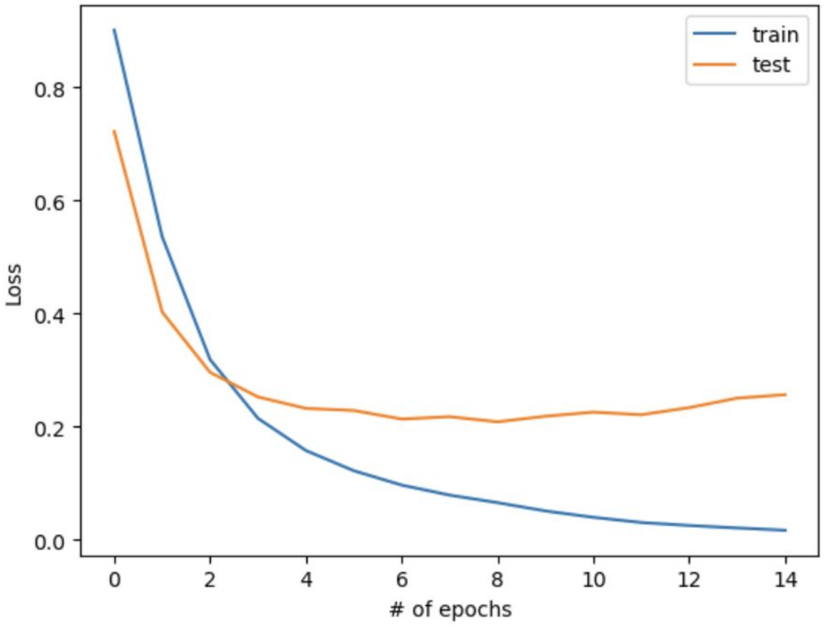
- Positive = 46%,
- Negative = 22%
- Neutral = 32%

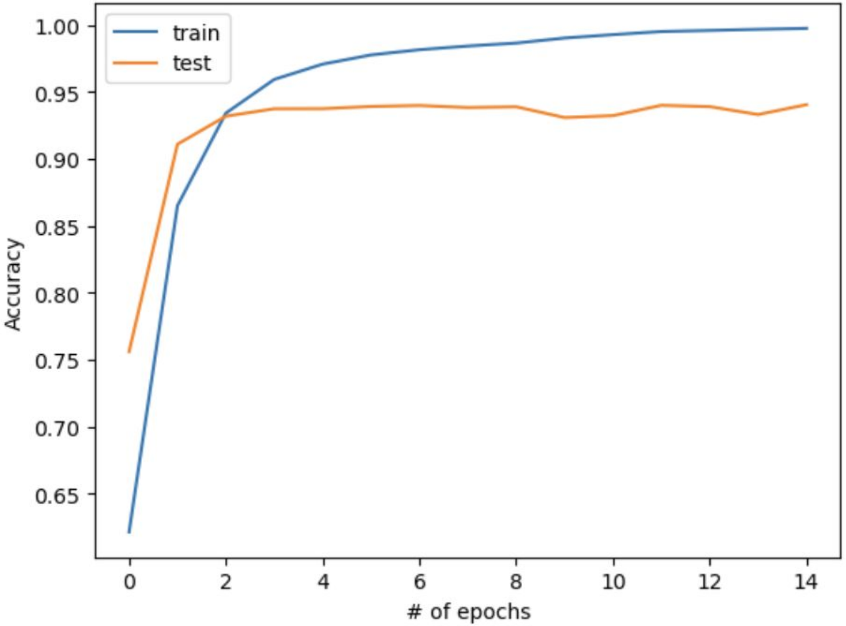
Modeling

- **Train/Test split**
  - 80% for training
  - 20% for testing
- **Model :** LSTM
- Keras sequential model
- **loss**: sparse categorical cross-entropy loss
- **optimizer**: adam
- **performance metric**: accuracy
- **# of epochs** = 15
- **Batch size** = 256

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_4 (Embedding)     (None, 26, 100)           13261300

 bidirectional (Bidirectiona  (None, 128)              84480
 l)

 dense_6 (Dense)             (None, 32)                4128

 dense_7 (Dense)             (None, 3)                 99

=================================================================
Total params: 13,350,007
Trainable params: 13,350,007
Non-trainable params: 0
_________________________________________________________________
```

Model Training and Evaluation





Accuracy of test set: 94.05%

Predicted labels

0 1 2

```
array([[0.99309134, 0.00377028, 0.00313842],
       [0.00136719, 0.9937552 , 0.00487752],
       [0.00136693, 0.99375856, 0.00487448],
       ...,
       [0.0018416 , 0.00317701, 0.99498147],
       [0.9931403 , 0.0037384 , 0.00312126],
       [0.00184036, 0.00317381, 0.9949858 ]]
```

	date	adj_close	volume	ticker	1_day_return	predicted_label
0	2017-01-31	28.308922	196804000	AAPL	-0.002307	0
1	2017-01-31	28.308922	196804000	AAPL	-0.002307	1
2	2017-01-31	28.308922	196804000	AAPL	-0.002307	1
3	2017-01-31	28.308922	196804000	AAPL	-0.002307	2
4	2017-01-31	28.308922	196804000	AAPL	-0.002307	0

Time-Series Forecasting:

Predicting Stock Prices

Data Preprocessing

- 1- Framing the dataset as a supervised learning problem.
  - predicting the stock price at the current hour (t), using the stock price and sentiment label from the previous time step.
- 2- Normalizing the input variables
  - to improve the performance and convergence of the LSTM model.
- Input Feature: The input shape was 10 time steps with 2 features.
- Reshaping the inputs (X) into the 3D format required by LSTMs: [samples, timesteps, features]
- Output Feature: The output feature was set as "adj_price."

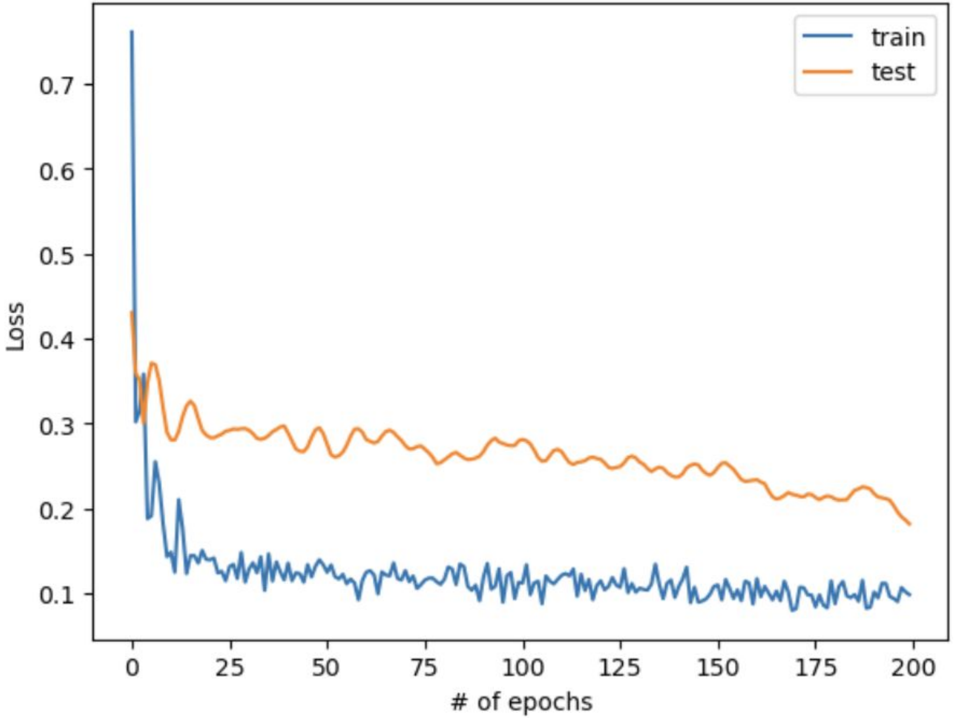
Modeling

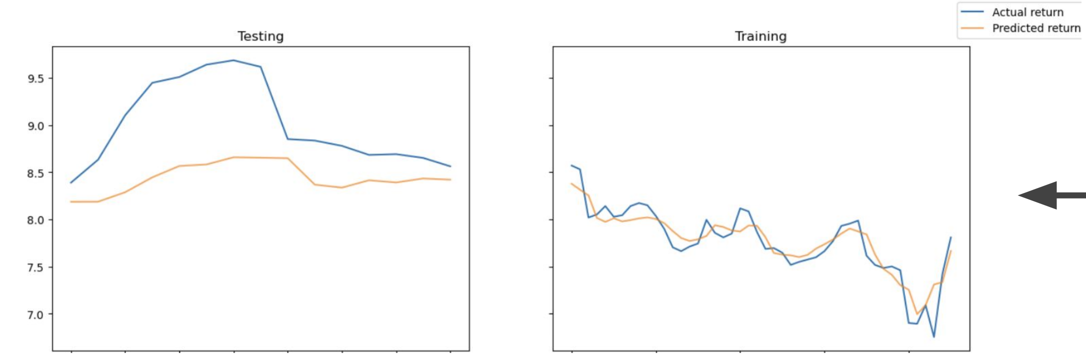
```
model = Sequential()
model.add(LSTM(400, input_shape=(train_X.shape[1], train_X.shape[2])))
model.add(Dropout(0.3))
model.add(Dense(128))
model.add(Dropout(0.3))
model.add(Dense(64))
model.add(Dropout(0.3))
model.add(Dense(1, activation ='linear'))
model.compile(loss='mse', optimizer='adam', metrics= ['mse', 'mae'])
history = model.fit(train_X, train_y, epochs=100, batch_size=16, validation_data=(test_X, test_y), verbose=0, shuffle=False)
```

Model Training and Evaluation

Error Metric:

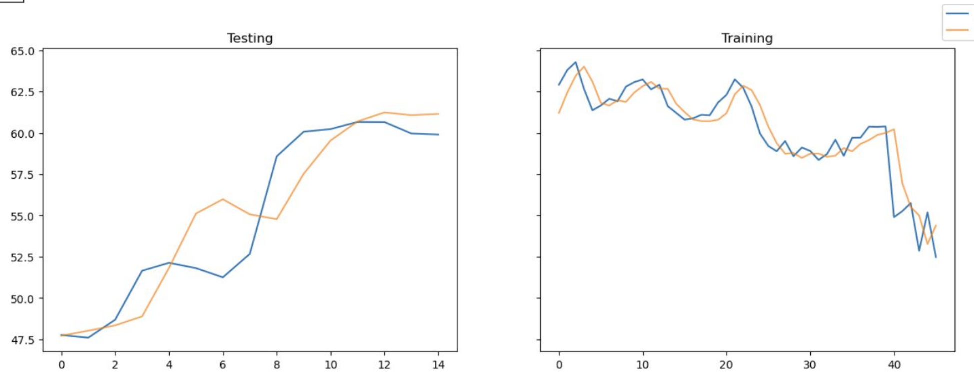
Root Mean Squared Error (RMSE)

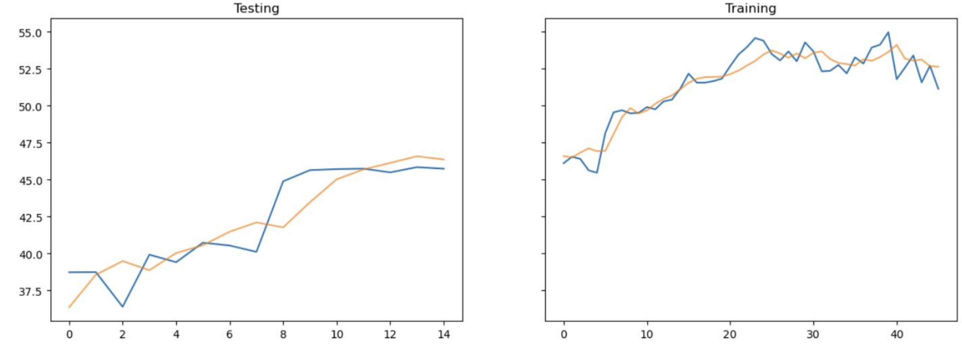




Ford

Google





Apple

Time-Series Forecasting:

Predicting Stock Returns

Data Preprocessing

- 1- Framing the dataset as a supervised learning problem.
  - predicting the stock price at the current hour (t), using the stock price and sentiment label from the previous time step.
- 2- Normalizing the input variables
  - to improve the performance and convergence of the LSTM model.
- Input Feature: The input shape was 10 time steps with 2 features.
- Reshaping the inputs (X) into the 3D format required by LSTMs: [samples, timesteps, features]
- Output Feature: The output feature was set as "adj_price."

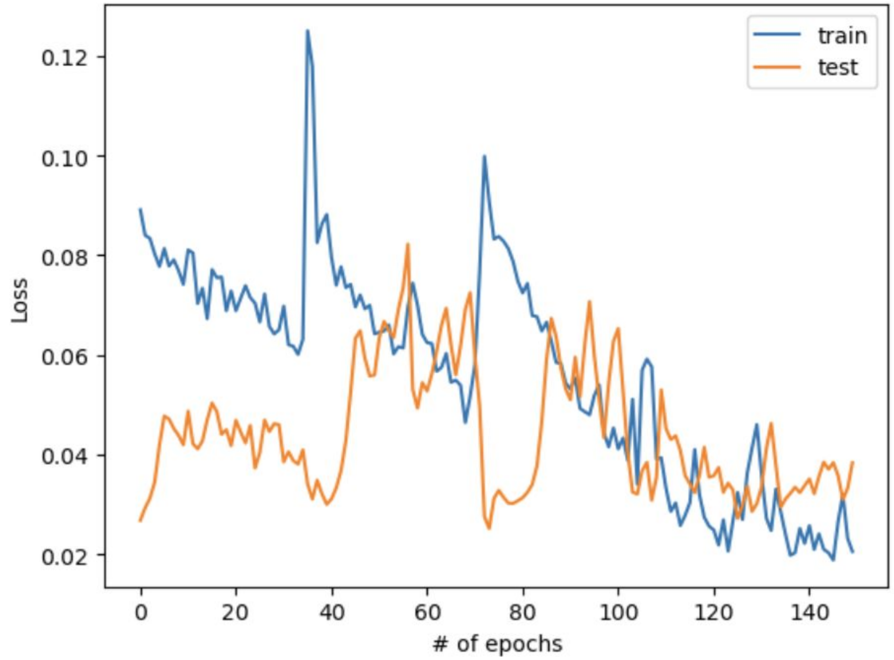
Modeling

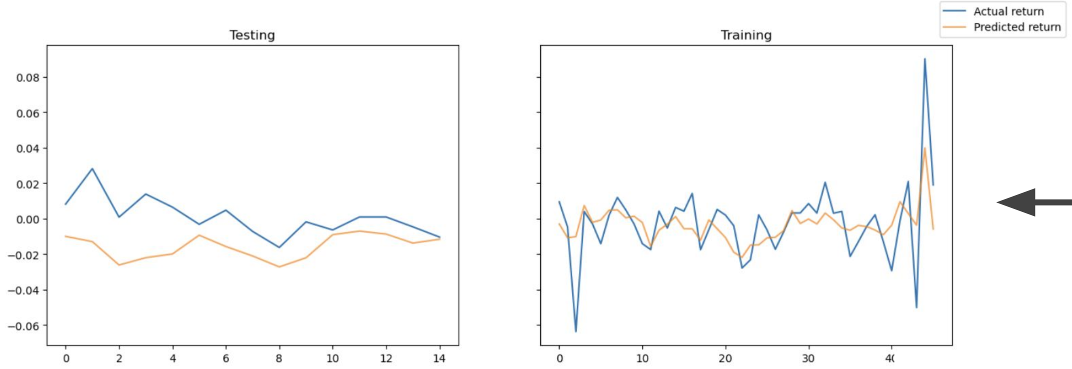
```
model = Sequential()
model.add(LSTM(300, input_shape=(train_X.shape[1], train_X.shape[2]))) model.add(Dropout(0.3))
model.add(Dense(1, activation= 'linear'))
model.compile(loss='mse', optimizer='adam', metrics= ['mse','mae'] )
history = model.fit(train_X, train_y, epochs=150, batch_size=10, validation_data=(test_X, test_y), verbose=1,
shuffle=False)
```

Model Training and Evaluation

Error Metric:

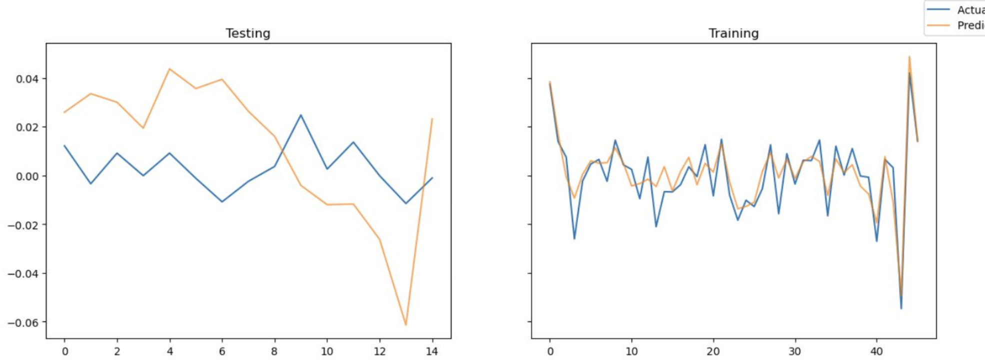
Root Mean Squared Error (RMSE)

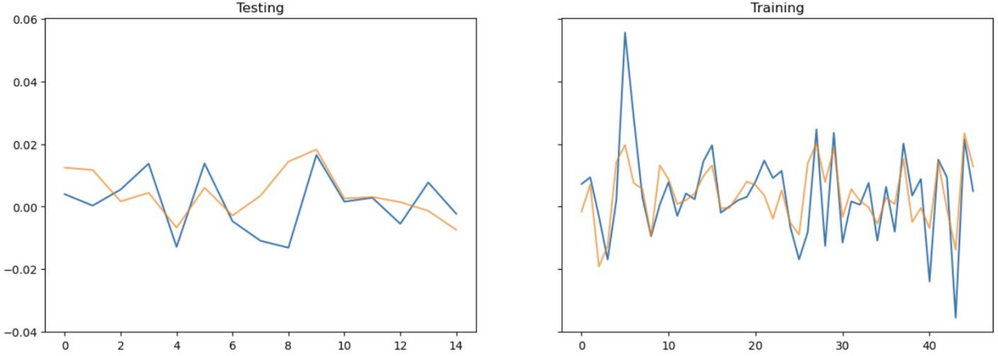




Ford

Google





Apple

Conclusion

- The LSTM models were successful in capturing the overall trend of both price and return for the majority of the evaluated stocks.
- The accuracy of the predictions varies across different stocks
- Some demonstrating higher accuracy in price prediction, others exhibiting better accuracy in return prediction.

FUTURE RESEARCH

- ❖ Further enhancement in the performance of these LSTM models:
  - ➢ gathering more data
  - ➢ refining the models through parameter adjustments
  - ➢ hyperparameter optimization
  - ➢ exploring alternative architectures

THANK YOU!