

# WSJ [ Python ]

## WSJ NLP S&P500 MARKET PREDICTOR

---

By: Gabe Blatstein

CAPSTONE PROJECT

# TABLE OF CONTENTS

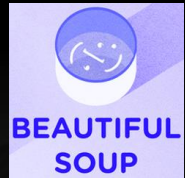
---

|    |                      |
|----|----------------------|
| 3  | Introduction         |
| 4  | Approach/Methodology |
| 5  | EDA                  |
| 12 | Modeling             |
| 15 | Conclusion           |

# INTRODUCTION

WSJ NLP S&P500 Market Predictor

WSJ NLP S&P500 Market Predictor Capstone Project



- **Capstone idea:**

- Originally the first idea was to create a classification model for newspaper/media websites to decide whether the article or headline was right leaning or left leaning
- Instead, only one newspaper was selected to predict something different, the direction the market went on a given day

- **Sources:**

- The WSJ was selected as the newspaper to be used
- The WayBackMachine was used to obtain the historical newspaper data.
- Yahoo Finance was used for the financial data (S&P500 Closing & Opening prices)

- **Main Python Packages:**

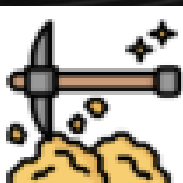
- BeautifulSoup: Web scraping tool which easily takes down and organizes the html from web page in a digestible and Python friendly way
- Selenium: Automated Web Driver, used to click through the WayBackMachine to switch between WSJ homepages

# APPROACH & METHODOLOGY

## WSJ NLP S&P500 Market Predictor

### OSEMN Framework

- **Obtain:** The text data needed was scraped from the WSJ homepage, and the financial data Yahoo Finance was used
- **Scrub:** The data needed the traditional Natural Language Processing cleaning
  - This consisted of lower casing the text data, and eliminating stop words and any other English language words or symbols
  - All of the text data was put into a DataFrame along with the corresponding label of a positive or negative day represented as 1 or -1 respectively
- **Explore:** A little bit of exploratory data analysis was conducted before the modeling to gain insight into the extracted data
- **Model:** For each of the different NLP strategies, I decided to run a very basic logistic regression model.
  - Advanced Modeling: Neural networks and pretrained word embeddings
- **Interpret:** Interpret all models along with all EDA results to form a conclusion of the data



OBTAIN



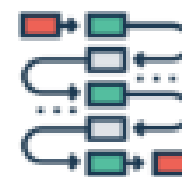
SCRUB



EXPLORE



MODEL



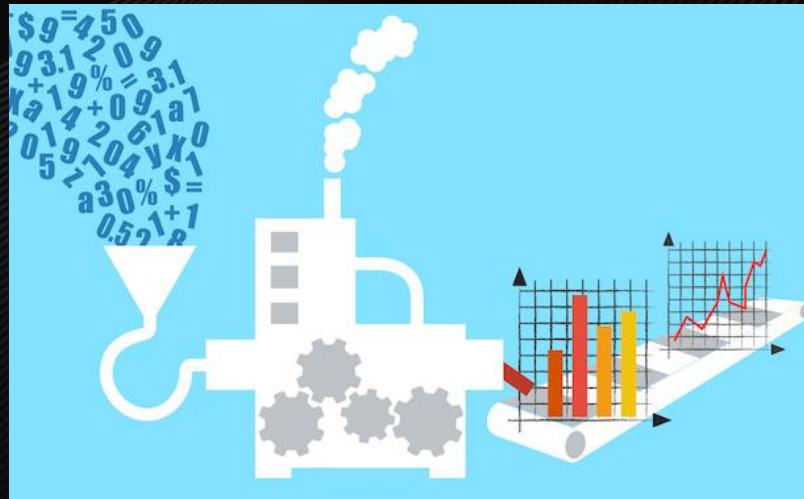
INTERPRET

# EXPLORATORY DATA ANALYSIS

WSJ NLP S&P500 Market Predictor

## EDA

- Word Frequencies
- Absolute Difference Word Frequencies
- Word Clouds

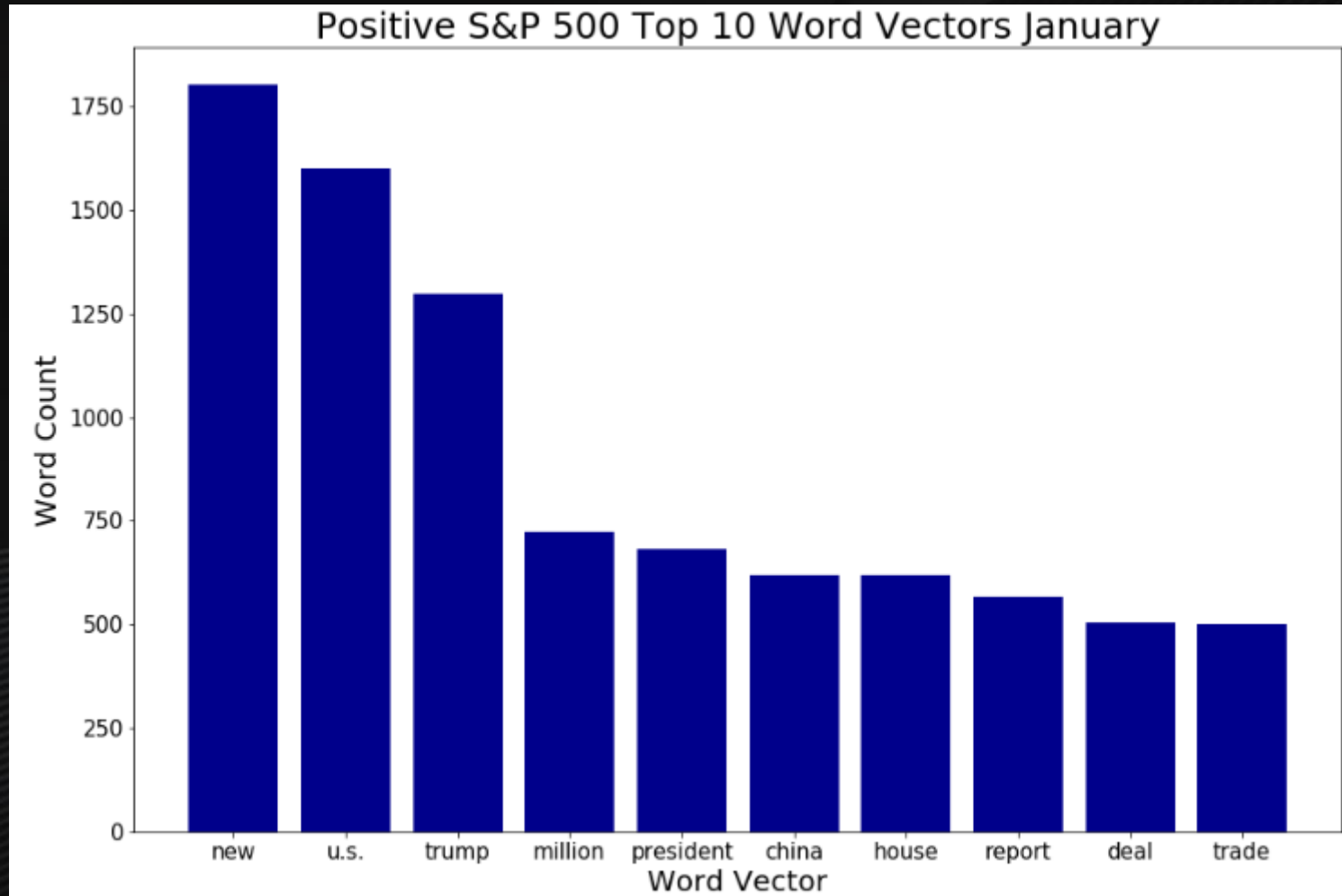


# EDA WORD FREQUENCY

## WSJ NLP S&P500 Market Predictor

Top Words on Positive  
S&P 500 Days:

1. new
2. u.s
3. trump
4. million
5. president
6. china
7. house
8. report
9. deal
10. trade



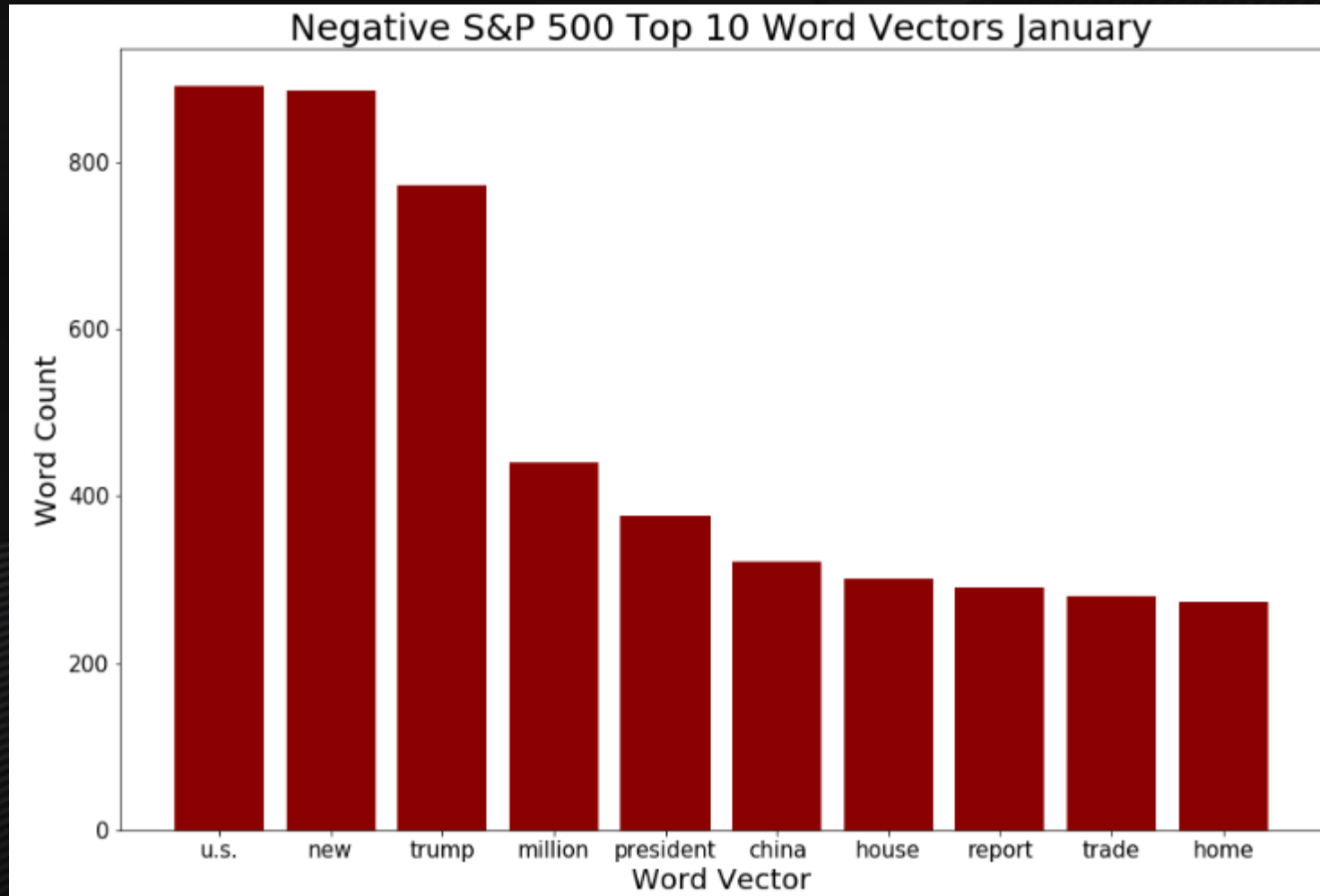


# EDA WORD FREQUENCY

## WSJ NLP S&P500 Market Predictor

Top Words on Negative  
S&P 500 Days:

1. u.s
2. new
3. trump
4. million
5. president
6. china
7. house
8. report
9. trade
10. home

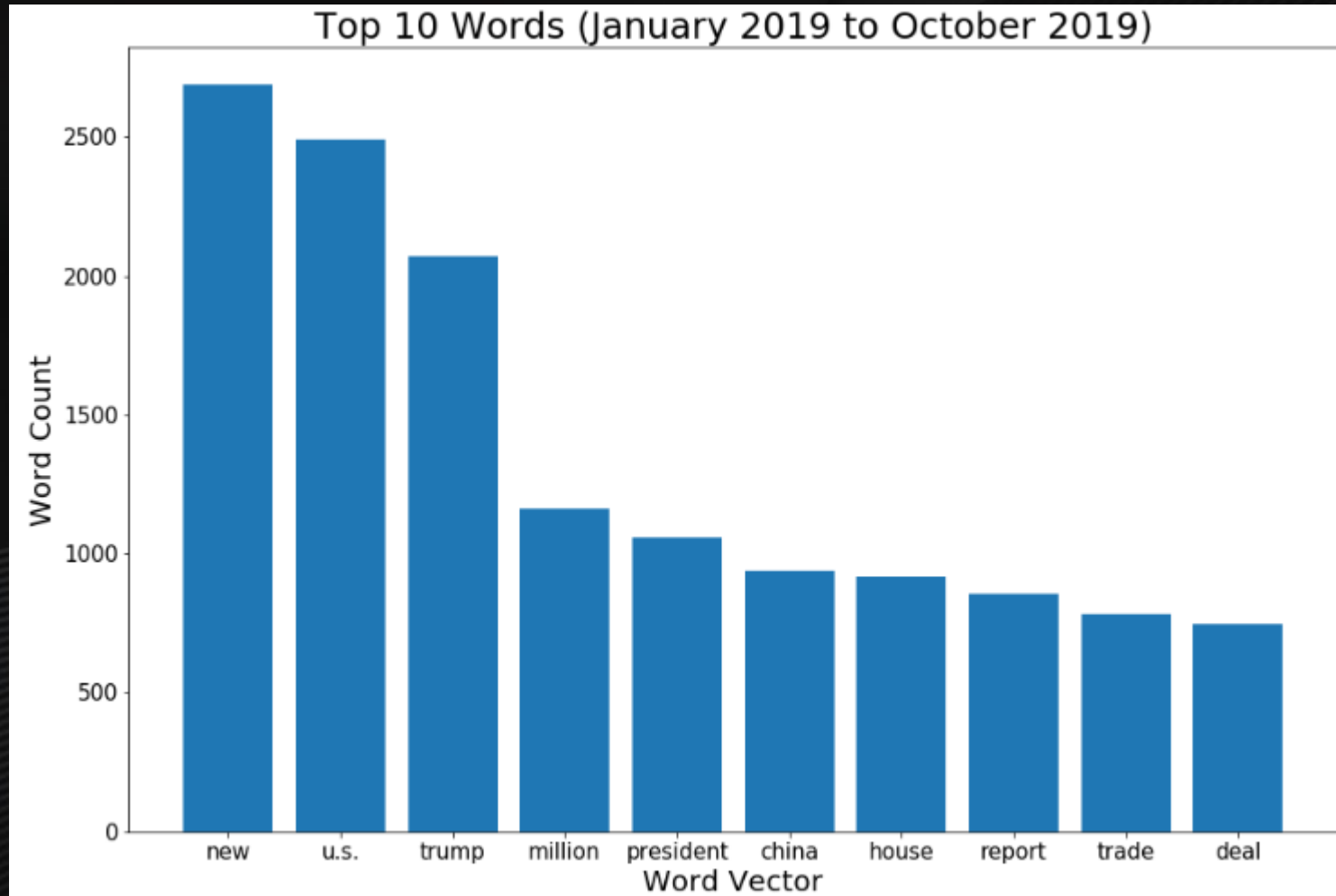


# EDA WORD FREQUENCY

## WSJ NLP S&P500 Market Predictor

Top Words:

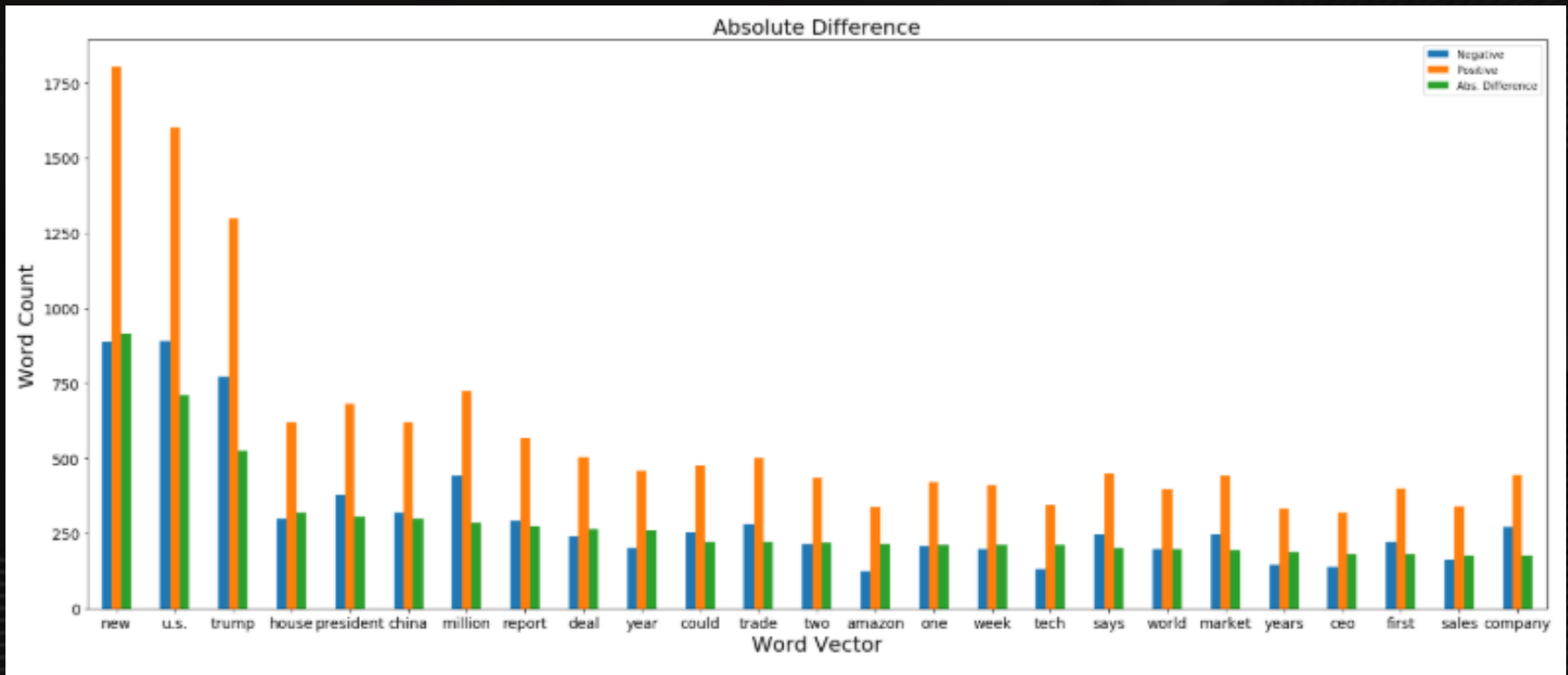
1. new
2. u.s.
3. trump
4. million
5. president
6. china
7. house
8. report
9. trade
10. deal





# EDA ABSOLUTE DIFFERENCE

## WSJ NLP S&P500 Market Predictor



# EDA WORD CLOUDS

# WSJ NLP S&P500 Market Predictor



# EDA WORD CLOUDS

# WSJ NLP S&P500 Market Predictor

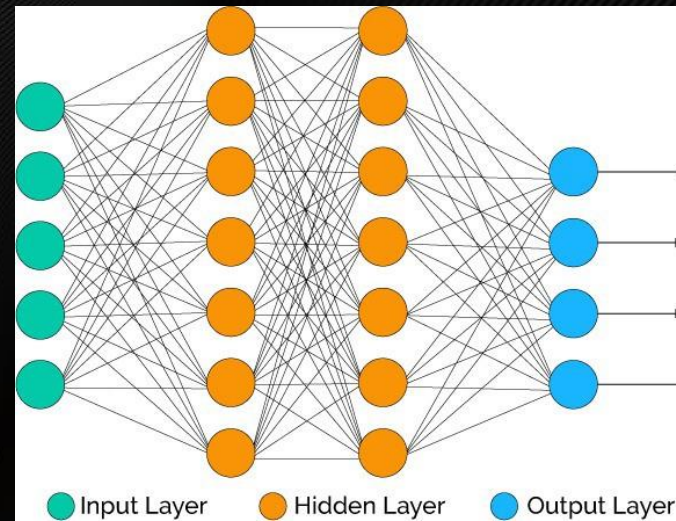


# MODELING

WSJ NLP S&P500 Market Predictor

# Modeling

- **Logistic Regression**
  - Bigrams, Lemmatization, Stemming
- **Neural Networks**





# MODELING – LOGISTIC REGRESSION

WSJ NLP S&P500 Market Predictor

- **Logistic Regression**

- **Baseline:**

- **Accuracy: 64.55%**

- **Lemmatization:**

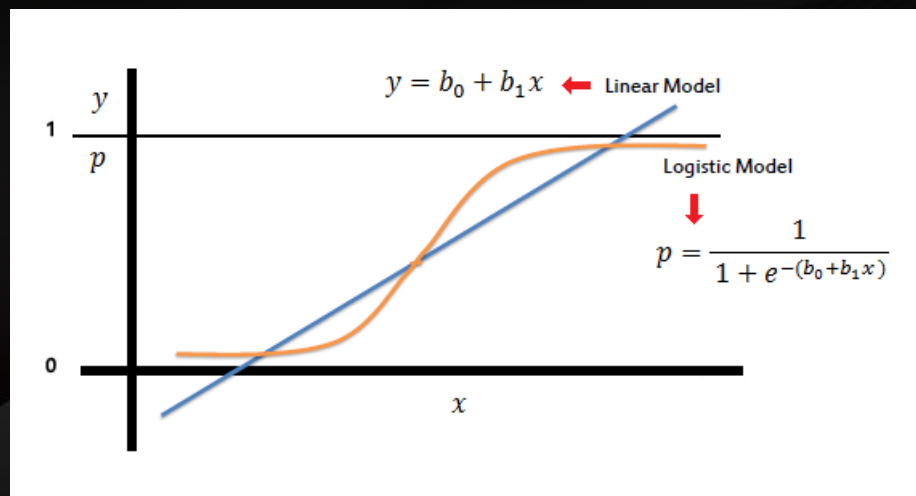
- **Accuracy: 64.92%**

- **Stemming:**

- **Accuracy: 64.92%**

- **Bigrams:**

- **Accuracy: 69.08%**



```
[('president', 'trump'), 0.0017477505802716874),  
 (('new', 'york'), 0.0016830190772986618),  
 (('logistics', 'report'), 0.0009216533042349815),  
 (('today', 'logistics'), 0.0009216533042349815),  
 (('trump', 'administration'), 0.0009154883991899314),  
 (('hong', 'kong'), 0.0008939112315322563),  
 (('white', 'house'), 0.0008939112315322563),  
 (('wall', 'street'), 0.0006473150297302546),  
 (('morning', 'risk'), 0.0006010782418923793),  
 (('risk', 'report'), 0.0006010782418923793),  
 (('737', 'max'), 0.0005579239065770289),  
 (('morning', 'download'), 0.0004901099510814785),  
 (('york', 'city'), 0.0004746976884688534),
```

Top Bigrams

# MODELING – NEURAL NETWORK

WSJ NLP S&P500 Market Predictor

- **Neural Networks**

- **Baseline:**

- **Accuracy: 62.80%**

- **Loss: .744**

- **Optimal Model:**

- **Accuracy: 65.10%**

- **Loss: .650**

Model: "sequential\_2"

| Layer (type)            | Output Shape     | Param # |
|-------------------------|------------------|---------|
| embedding_2 (Embedding) | (None, 250, 100) | 250000  |
| lstm_2 (LSTM)           | (None, 100)      | 80400   |
| dense_2 (Dense)         | (None, 8)        | 808     |
| dense_3 (Dense)         | (None, 2)        | 18      |

Total params: 331,226  
Trainable params: 331,226  
Non-trainable params: 0

None

Model: "sequential\_8"

| Layer (type)            | Output Shape     | Param # |
|-------------------------|------------------|---------|
| embedding_8 (Embedding) | (None, 250, 100) | 250000  |
| lstm_8 (LSTM)           | (None, 120)      | 106080  |
| dense_19 (Dense)        | (None, 24)       | 2904    |
| dense_20 (Dense)        | (None, 16)       | 400     |
| dense_21 (Dense)        | (None, 8)        | 136     |
| dense_22 (Dense)        | (None, 2)        | 18      |

Total params: 359,538  
Trainable params: 359,538  
Non-trainable params: 0

None

# CONCLUSION

## WSJ NLP S&P500 Market Predictor

### • **Key Takeaways:**

#### • **Best Model:**

- **Bigram Logistic Regression:** This model achieved around 69 percent accuracy in predicting the direction of the S&P 500 for the following day.
- **Neural networks** were not sufficient and due to the black box component, the optimal number of layers were not found

#### • **NLP Takeaways from WSJ:**

- **Trump's name** was mentioned the most both in negative and positive days
- **Word clouds** are a great visually appealing way to represent any type of text data
- **Amazon and Apple** were the two companies mentioned the most
- **The model accuracy** is very impressive considering the most mentioned words are very similar for positive and negative days

#### • **Further Analysis:**

- **Time series modeling** of Amazon and Apple to see whether the stock market went up or down when mentioned
- **Stacked modeling**





THANK YOU, QUESTIONS?

---