Sebastian Longoria
Professor Ashton
ISTA 421
May 13, 2025

Final Project: How Worldwide Events Affect Tech Company Stock Prices

Problem Formulation and Significance

- **Problem Statement**

The goal of this project is to understand how different real-world events shape stock behavior, and whether we can model those reactions in a meaningful way that offers insight into how the market will react in specific situations, ultimately allowing everyday people to have investment opportunities not often shared with the public. For this project I am investigating two types of major real-world events we have seen in the past few decades: technological advancements/innovations and US government policy-driven changes and determining if there is an impact in the predictability of two massive publicly traded companies, Apple and Microsoft.

Rather than trying to predict the price of the stock based off historical trends and how healthy the market currently is, the focus with this project was to evaluate how predictable the prices can be (if prices will go up or down) under these certain conditions. At the heart of it I am trying to answer this central question: "Do major real-world events cause the stock market to behave more chaotically or more predictably in the short term?" To answer this question, I trained a logistic regression model on "normal" (non-event) trading days and then tested it in specific time windows (6 trading days after the day a specified event takes place) marked as post-tech event or post-policy event and observed how stable the predictive pattern stayed.

The logistic regression model is made to predict on if the stock price will go up or down the following day. For example, after training the model using "normal" days and it performs poorly during a specified event window, that may suggest the event disrupted the market's typical trends, however, if it performs better than the event didn't have a major effect on the market as it followed the expected trend. This makes the model more than a prediction/forecasting tool, but a way to analyze how real-world events affect the market's behavior.

- **Significance**

Understanding whether certain types of real-world events introduce chaos or clarity into the market is valuable for anyone trying to interpret financial trends. Rather than assuming all events have the same type of impact, this model distinguishes between those that create uncertainty and those that may reinforce investor confidence. For example, most of the population may make the assumption that technological innovation may lead to more predictable upward behavior, while unexpected policy shifts might cause erratic or reactionary movement across the whole market.

The results could help investors and analysts better anticipate how the market might behave following similar events in the future. If a model trained on normal data fails to perform

well after a tariff announcement, that reveals something deeper about the unpredictability such events create. On the other hand, if the model performs well during technological advancement event windows, it suggests those types of events result in a more expected behavior.

- **Dataset Descriptions**

To carry out this experiment, I used two datasets: daily stock data for Apple (AAPL) from December 1980 through January 2025 and Microsoft (MSFT) from March 1986 through February 2025. Each dataset includes the following features for each trading day (weekdays): Opening price, High price for the day, Low price for the day, Closing price, Adjusted Closing price, and Volume of stocks traded that day. I also created a few other features to help in the analysis: Daily Return (percentage change in adjusted closing price from the previous day), a rolling 5-day Volatility column (standard deviation of the Daily returns for the past 5 days) and an Event type column (numerical value given for each event type: tech vs policy).

To measure the impact of specific events I wanted to look at, I manually annotated each "event_type" column in each dataset.

- 0 = normal market periods
- 1 = Technological Advancement/Innovation-related event windows
- 2 = Government policy-related event windows

Each event window included the day of the specified event and the next 5 trading days following that date, and I choose a variety of different events for each company to test:

- Apple Technological Advancement/Innovation-related events - announcement of iPhone, announcement of iPad, and announcement of M1 chip
- Apple Government policy-related events - 2018 US-China tariffs and US-EU antitrust probing
- Microsoft Technological Advancement/Innovation-related events - release of Windows 95, announcement of Microsoft Azure, and the announcement of the Xbox
- Microsoft Government policy-related events - US antitrust lawsuit being filed and when the antitrust was settled

All together, these datasets contain thousands of trading days, with ease of access to add a lot of rows flagged as belonging to any event window. This large sample allows for robust training on non-event periods and meaningful comparisons of model performance across different event types.

Exploratory Data Analysis

- **Data Exploration**

To begin understanding the data, I loaded and cleaned the daily stock data for both Apple and Microsoft. Each dataset includes numerical features such as Open, High, Low, Close, Adjusted Close, and Volume. I then created three new columns for each stock: Daily Return (the percentage change in Adjusted Close compared to the previous day), Volatility_5 (the standard

deviation of the Daily Return over the past five trading days), event_type (numerical value for classing what type of day that date fell into).

Microsoft data frame head after cleaning:

	date	Open	High	Low	Close	Adj Close	Volume	Daily Return	Volatility_5
5	1986-03-20	0.098090	0.098090	0.094618	0.095486	0.058758	58435200	-2.654749	2.816513
6	1986-03-21	0.095486	0.097222	0.091146	0.092882	0.057156	59990400	-2.727116	1.893905
7	1986-03-24	0.092882	0.092882	0.089410	0.090278	0.055554	65289600	-2.803508	0.432563
8	1986-03-25	0.090278	0.092014	0.089410	0.092014	0.056622	32083200	1.922941	2.016293
9	1986-03-26	0.092014	0.095486	0.091146	0.094618	0.058224	22752000	2.830012	2.814910

Apple data frame head after cleaning:

	date	Adj Close	Close	High	Low	Open	Volume	Daily Return	Volatility_5
5	1980-12-19	0.097116	0.126116	0.126674	0.126116	0.126116	48630400	6.102858	5.758344
6	1980-12-22	0.101842	0.132254	0.132813	0.132254	0.132254	37363200	4.867013	5.317928
7	1980-12-23	0.106140	0.137835	0.138393	0.137835	0.137835	46950400	4.219907	1.474397
8	1980-12-24	0.111726	0.145089	0.145647	0.145089	0.145089	48003200	5.262747	1.202232
9	1980-12-26	0.122039	0.158482	0.159040	0.158482	0.158482	55574400	9.230927	1.963757

Basic descriptive statistics confirmed wide ranges and high standard deviations, which reflect the decades-long time span and exponential growth of both companies. For example, Apple's Daily Return ranged from -51.87% to +33.23%, while Microsoft's ranged from -30.11% to +19.57%. All while, rolling volatility values revealed trends in price stability, particularly during economic crises or tech booms. I also confirmed each columns datatype to be either float64 or int64 and that there were no missing values that may mess up the analysis.

Microsoft statistical breakdown:

```
Microsoft Summary Stats:

              Open         High          Low        Close    Adj Close  \
count  9800.000000  9800.000000  9800.000000  9800.000000  9800.000000
mean     63.173728    63.812493    62.513818    63.187157    57.451679
std      98.682750    99.575073    97.725918    98.697881    98.714771
min       0.088542     0.092014     0.088542     0.090278     0.055554
25%       5.898438     5.976563     5.794922     5.880860     3.618860
50%      27.436250    27.770000    27.200001    27.490000    19.266710
75%      47.597813    48.145000    47.062500    47.592500    40.066001
max     467.000000   468.350006   464.459991   467.559998   465.786438

             Volume  Daily Return  Volatility_5
count  9.800000e+03   9799.000000   9795.000000
mean   5.630897e+07      0.112386      1.756862
std    3.812127e+07      2.102810      1.206792
min    2.304000e+06    -30.115887      0.068665
25%    3.141062e+07     -0.910068      0.964287
50%    4.946235e+07      0.037737      1.479103
75%    7.027870e+07      1.119693      2.207315
max    1.031789e+09     19.565212     18.018816
```

Apple statistical breakdown:

```
Apple Summary Stats:

              Open          High           Low         Close     Adj Close  \
count  11107.000000  11107.000000  11107.000000  11107.000000  11107.000000
mean      24.339076     24.598169     24.092608     24.357607     23.522229
std       50.166818     50.691902     49.682631     50.217498     49.767881
min        0.049665      0.049665      0.049107      0.049107      0.037815
25%        0.300090      0.306362      0.292411      0.300290      0.243402
50%        0.542679      0.553393      0.534598      0.542411      0.446682
75%       21.367679     21.569285     21.115715     21.397143     18.260086
max      258.190002    260.100006    257.630005    259.019989    259.019989

             Volume  Daily Return  Volatility_5
count  1.110700e+04  11106.000000  11102.000000
mean   3.154341e+08      0.109604      2.304649
std    3.348735e+08      2.778277      1.562694
min    0.000000e+00    -51.869200      0.075523
25%    1.111164e+08     -1.260233      1.266991
50%    2.036944e+08      0.000000      1.962168
75%    3.960418e+08      1.437862      2.946773
max    7.421641e+09     33.228009     24.495311
```

- **Handling of Missing and Imbalanced Data**

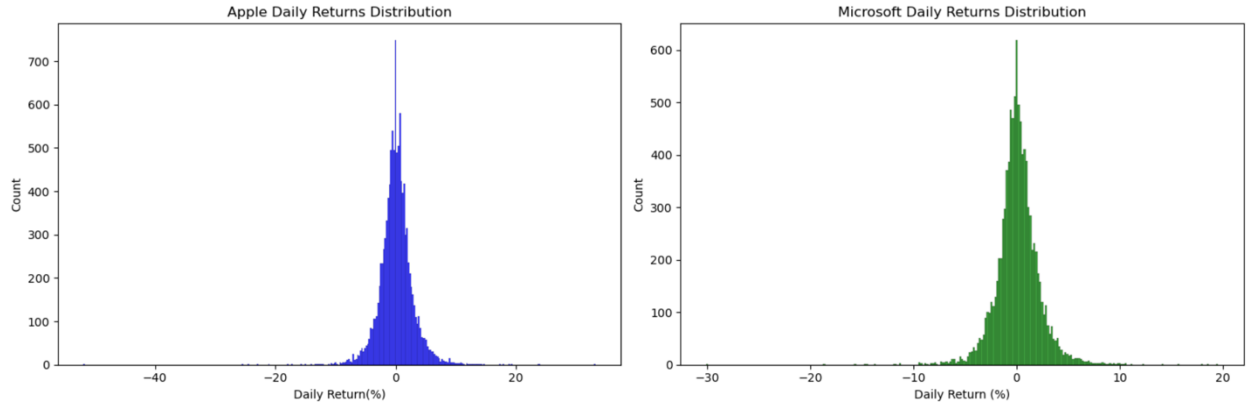
Originally there were no missing values in the original price or volume columns. However, due to the rolling calculation of volatility and the fact daily return used percentage change, the first few rows contained NaNs since nothing could be calculated before the company was publicly traded. Rather than trying to skew the calculations or results any columns that contained NaNs were dropped, making both datasets clean and 100% usable.

For classification, prior to running the model a target variable was created, signifying a 1 if the stock's daily return was positive or a 0 otherwise. Since this variable is derived from the continuous price series, it is not severely imbalanced. However, by isolating subsets for event types, the sample sizes became smaller. This was addressed not through resampling, but through evaluating the model separately on each event window to maintain clarity about performance variation.

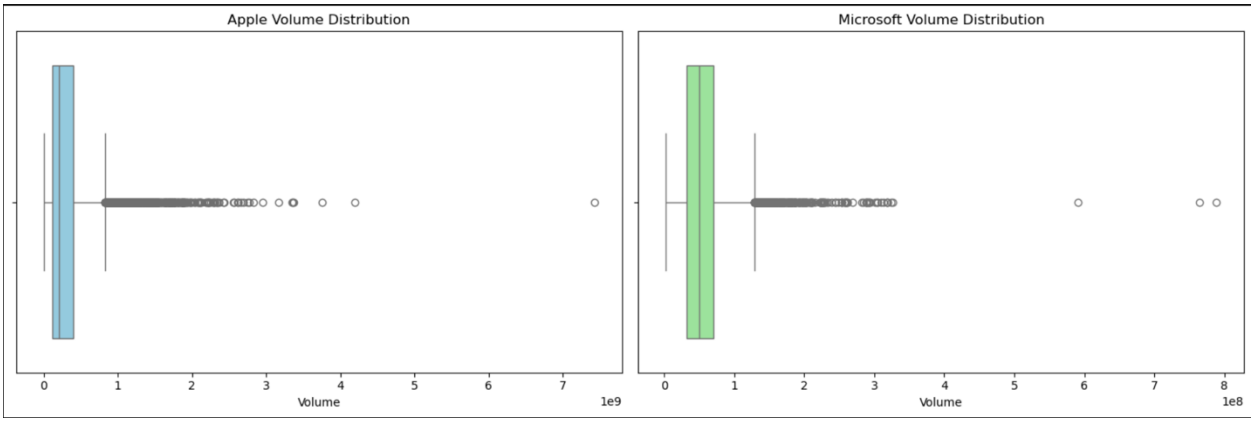
- **Data Visualizations**

I made sure to create numerous visualizations of the data to fully understand the patterns seen throughout each dataset.

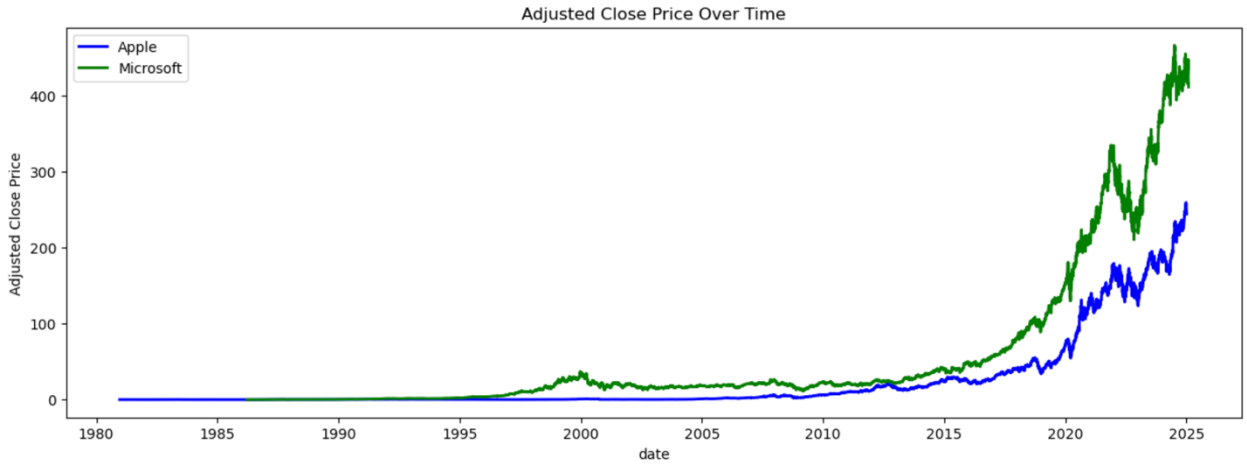
- Firstly, histograms of Daily Return for both stocks showed a roughly symmetric distribution centered near 0



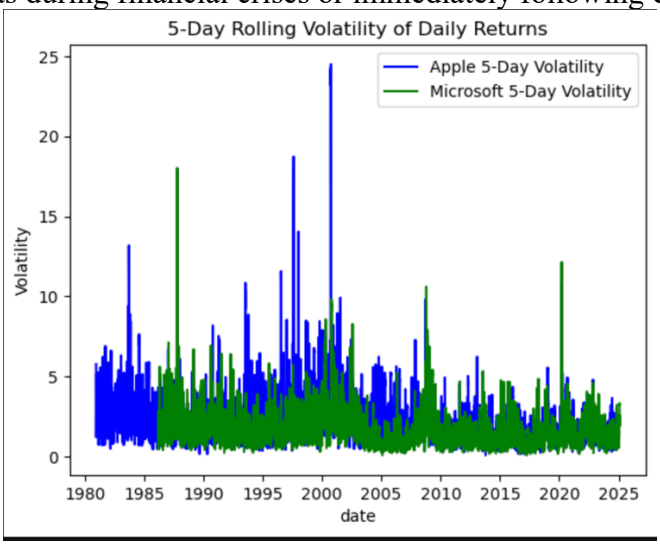
- Box plots showing trading volume, which showed his skewness, especially in the Microsoft dataset



- Line plots of Adjusted Close over time revealed the overall growth trajectory of each company and the timing of significant market jumps or drops.



- Line plots of 5-Day Rolling Volatility made it easy to spot periods of high market instability, such as during financial crises or immediately following event dates.



These visualizations and summary statistics laid the groundwork for understanding what "normal" behavior looks like in this market, helping inform the later classification model trained on non-event periods. The combination of statistical summaries, engineered features, and targeted visualizations gave a clear, interpretable foundation for analyzing how real-world events may influence stock predictability.

Model Selection, Application, and Evaluation

- **Model Choice Justification**

  For this project, I implemented logistic regression entirely from scratch while also utilizing Pandas and NumPy for certain parts, but the functions and model utilized no packages. Logistic regression was chosen because it is a fundamental linear model that we studied in class and is particularly suited for binary classification problems like this one: predicting whether the stock adjusted closing price will go up or down the next day.

  The simplicity of the model helps isolate whether the drop or increase in performance is due to the event itself, rather than complexities of the model. It allows me to evaluate changes in predictability based off the targeted event type, which ultimately supports the core question I am trying to answer: "Do major real-world events cause the stock market to behave more chaotically or more predictably in the short term?"

- **Method Application**

  Quick explanation: To implement the model in jupyter notebook I implemented the sigmoid function, gradient descent updates, and logistic regression training manually using NumPy. After computing engineered features (Daily Return and Volatility_5), I defined the target variable as 1 if the next day's return was positive, otherwise 0. I then trained the model only on normal days (event_type = 0), and used it to make predictions on the two types of event windows: tech (event_type = 1) and policy (event_type = 2).

  Demo: seen in pdf or code in the Github Repository

- **Model Evaluation**

  After training on non-event periods, the model was evaluated separately on tech and policy windows for both Apple and Microsoft. I used multiple metrics to evaluate model performance: accuracy, precision, recall, F1 score, and a confusion matrix for each dataset and each event types.

  Demo: seen in pdf or code in the Github Repository

Results and Conclusions

- **Results**

  To fully understand the model's behavior, we examined both the accuracy and key evaluation metrics associated with each event type and each company. These allowed us to assess overall performance and how it was affected by the nature of the event.

The results of the logistic regression model varied significantly depending on the type of event and the company being analyzed. For Apple and Microsoft, both tech and policy events generally resulted in reduced prediction accuracy compared to normal periods. However, Microsoft's policy event windows showed a much stronger signal of predictability than the other categories. Here are the results of the validation for each company and the event type:

- Apple Technological Advancement/Innovation Events: Accuracy = 0.444, Precision = 0.444, F1 Score = 0.615
- Apple Government Policy Events: Accuracy = 0.417, Precision = 0.417, F1 Score = 0.588
- Microsoft Technological Advancement/Innovation Events: Accuracy = 0.444, Precision = 0.444, F1 Score = 0.615
- Microsoft Government Policy Events: Accuracy = 0.667, Precision = 0.667, F1 Score = 0.8

In all technology-related event windows, the model's accuracy dropped below 50%, suggesting that technological innovations introduced unpredictability or market noise that made short-term price movements harder to anticipate. Meanwhile, Microsoft's performance during policy events stood out with its accuracy and F1 score both increased, indicating that such events may introduce clearer patterns or investor responses that a simple model could capture

- **Conclusion**

This project helped confirm something I had suspected going in: that not all major events shake up the market in the same way. Technological events seem to generate a lot of noise, possible excitement, speculation, and maybe even confusion in how investors respond. That unpredictability showed up in the way the model struggled to make accurate predictions during those periods. On the other hand, policy-driven events (especially for Microsoft) made the stock a bit more predictable, as if people react quickly and decisively when governments step in.

It was also interesting to see that the model leaned heavily toward predicting stock increases. That might say something about how the market generally moves in the positive direction for both these companies, but it also tells me there's probably room to tweak things like thresholds or even rethink how the output is interpreted. Overall, it was a fun experience to use machine learning not just to predict prices, but to ask when those predictions are worth trusting.

- **Workforce Preparations and Applications**

Working on this project gave me a better sense of how machine learning can be used beyond just making predictions. It's not always about getting the perfect forecast—it's about understanding when the predictions are dependable. From a practical standpoint, knowing that different types of events affect predictability in different ways

could help people make smarter investing decisions. If someone knows that the market gets unpredictable after major tech announcements but becomes more stable during certain policy changes, that's useful info to have.

If I had more time or resources, I'd investigate improving the model by bringing in variables like news sentiment or broader economic indicators. I'd also want to try other models we learned in class, such as decision trees, to see if they respond differently to certain types of events. There's a lot of potential here to turn this into something more powerful, especially if you're trying to build tools that adjust their strategy depending on what's happening in the real world.