STAT 27410 Final Project Proposal - Comparions between Frequentist and Bayesian Approach to SalesForce Stock Price Prediction

Andrew Su, Scarlett He

1. Introduction

Stock price prediction presents unique challenges that demand sophisticated statistical approaches for effective modeling under uncertainty. This study seeks to explore innovative approaches to enhance investment decision-making by comparing traditional frequentist methods with Bayesian techniques in the context of stock price prediction.

Over a three-year period from January 1st, 2022, to January 1st, 2025, we analyze historical stock prices for:

- Salesforce Inc. Common Stock (CRM) (Nasdaq, n.d.b)

We use sector specific ETF:

- Invesco QQQ Trust (QQQ) (Nasdaq, n.d.a)

As well as S&P 500 index (SPX) (Nasdaq, n.d.c) as a baseline for comparison.

All data used is from the 3-year timespan of January 1st, 2022 to January 1st, 2025 (753 trading days) and obtained directly from the Nasdaq stock exchange. Data was sourced directly from the Nasdaq stock exchange, ensuring a robust empirical basis for our analysis.

For each security, the data set include the following variables:

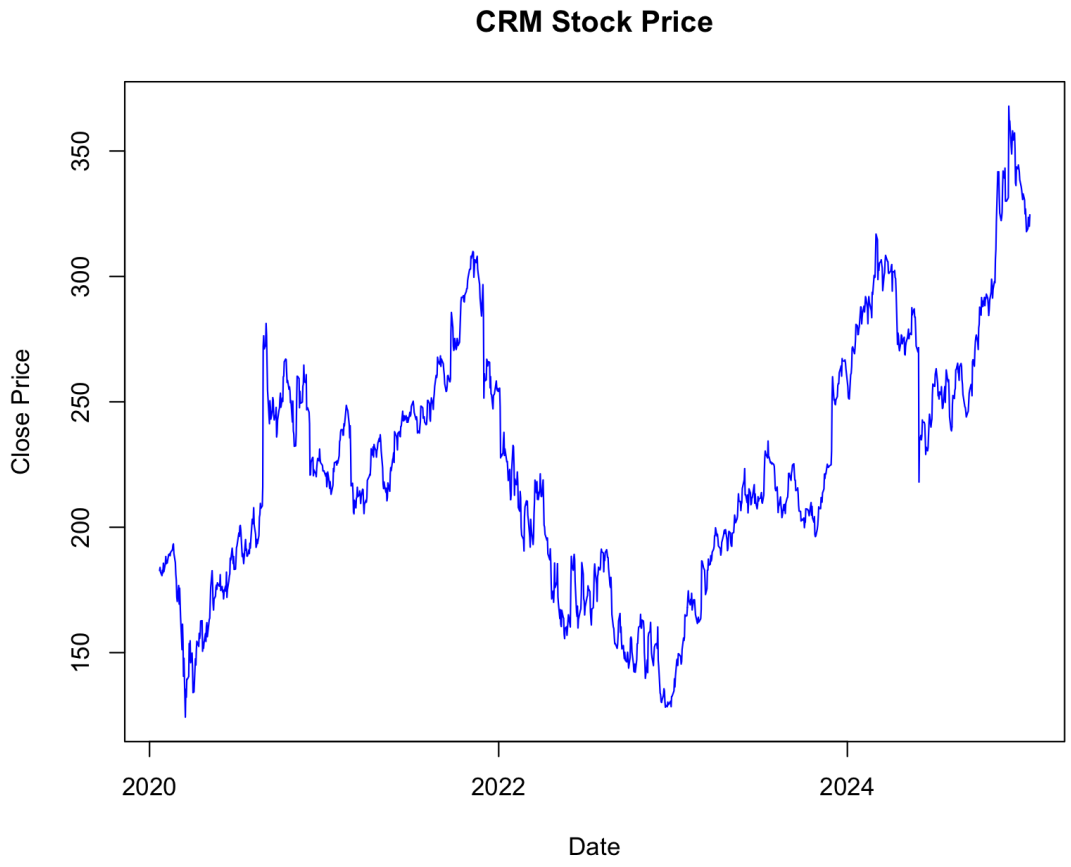
- **Date**: The date of the trading day
- **Close/Last**: The price of the security at the end of the trading day
- **Volume**: The total number of shares traded during the trading day
- **Open**: The price of the security at the start of the trading day
- **High**: The highest price of the security during the entire trading day
- **Low**: The lowest price of the security during the entire trading day

For the purpose of our analysis, we will only be looking at the Date, Close/Last categories across each of the securities to build different models to capture essential market dynamics and trends.

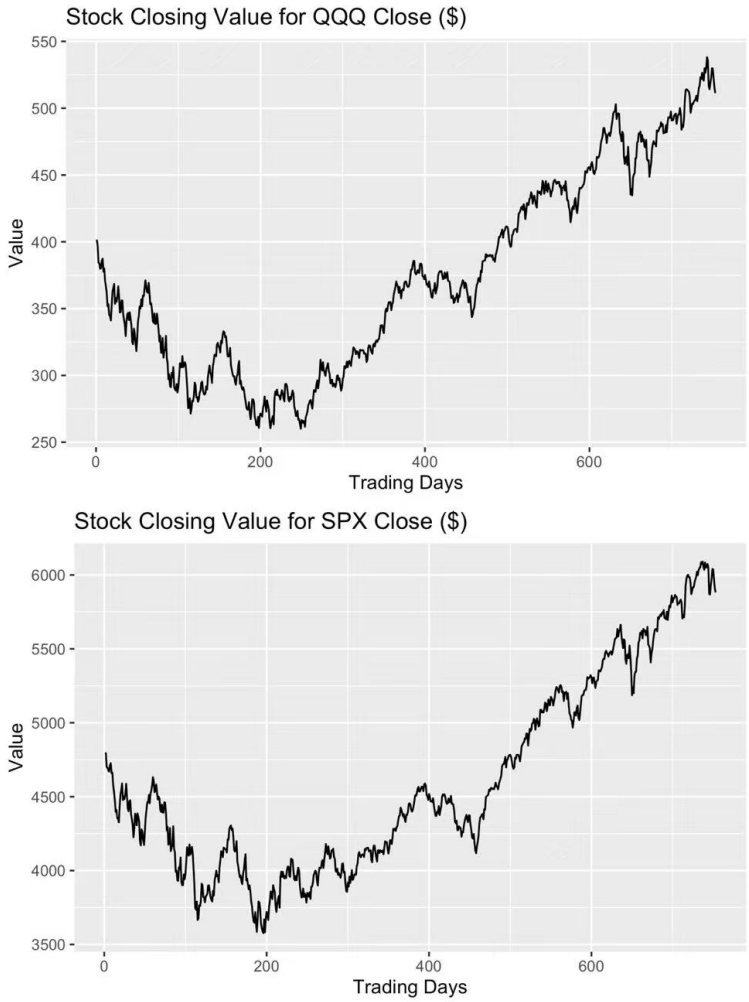
This study aims to provide insights into the important features to capture comprehensive market dynamics, and various uses of different models.

2. Exploratory Data Analysis

We will start by looking the trends and changes in CRM over this 3 year period.



The stock demonstrates significant price volatility, fluctuating between approximately 130 dollars and 350 dollars, with multiple distinct trading ranges and trend reversals.The stock displays cyclical behavior with distinct bull and bear phases.



We explored the index price, and found that CRM and the index prices show similar patterns. Through further analysis, we see that CRM exhibits an exceptionally strong positive correlation with QQQ (0.93), demonstrating very tight co-movement with the tech sector.Similarly high correlation with SPX (0.91) indicates that CRM closely tracks the broader market performance.

These high correlations suggest that:

- CRM is highly integrated with both tech sector and broader market movements
- While company-specific factors exist, macro market forces significantly influence CRM's performance

Thus, we will incorporate market index in our prediction to create more accurate prediction.

3. Frequentist Analysis

3.1 Proposed Frequentist Model(s)

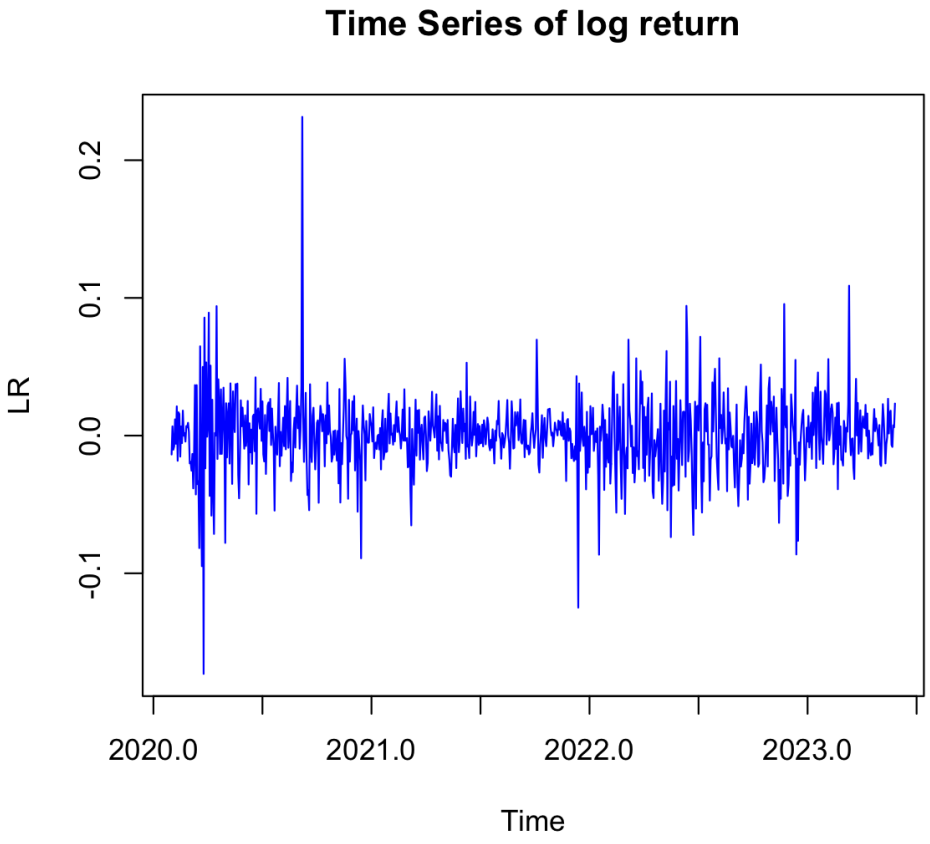
For all models, We choose to predict daily log returns. The percentage change in stock price from time t to $t+1$, denoted as $R_{i,t}$, is given by:

$$R_{i,t} = \frac{P_{i,t+1} - P_{i,t}}{P_{i,t}}$$

We first train and test the frequentist models using around $\frac{2}{3}$ of the data. Then turn $\ln R_{i,t}$ back to price to show the price prediction.

We made this decision primarily for 2 reasons:

1. Log returns tend to be more normally distributed than simple returns, which facilitates statistical modeling and analysis.
2. Log transformation helps stabilize variance in the time series, reducing heteroscedasticity and making the data more suitable for statistical analysis.



We see the graph of log return shows like white noise around mean 0.

3.1.1 Time Series Models

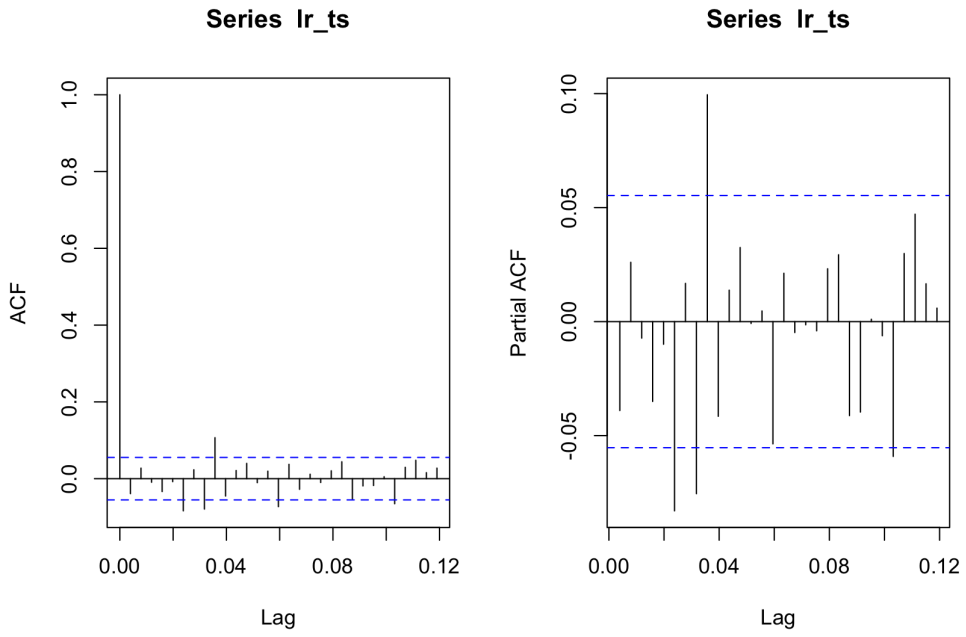
Time series analysis emerges as a fundamental approach for financial data forecasting due to financial data's natrual of time dependency. It offers ways to locate patterns from historical fianncial data and form predictions to provide insights on investment decisions.

The strength of time series analysis lies in its capacity to clean data and remove confounding variables and white noises. It also handles the non-independency between data in a sequence of time.

The initial phase involved testing for stationarity in the financial time series data. The log transformation of the return is stationary according to the Augmented Dickey-Fuller (ADF) test.

```
        Augmented Dickey-Fuller Test

data:  lr_ts
Dickey-Fuller = -11.089, Lag order =
10, p-value = 0.01
alternative hypothesis: stationary
```

Following the establishment of stationarity, we examined Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots. They provide insights in determining the presence and order of autoregressive (AR) and moving average (MA) components. The patterns observed in these plots served as preliminary indicators for model specification.



The PACF suggests a potential AR component due to significant spikes at low lags. Since there is small spikes with some lag on the ACF plot, the graph does not give strong indication of MA component selection. We may explore different MA components.

We chose not to incorporate seasonality because the spike lags do not share a common divisor. Additionally, the period would be 252 trading days, which is too large for the code to efficiently optimize the model.

To ensure optimal model selection, we fitted multiple ARIMA models to each financial time series, varying the orders of AR or MA each time, and compare them. Model performance was evaluated using established criteria such as AIC (Akaike Information Criterion) and BIC (Bayesian Information Criterion).

We compare between two selections: one focusing on AR component, and another model focusing on MA component.

First, We propose model ARIMA(6,0,1).

```
Call:
arima(x = lr_ts, order = c(6, 0, 1))

Coefficients:
         ar1     ar2     ar3      ar4      ar5      ar6     ma1  intercept
      -0.7494  0.0115  0.0172  -0.0408  -0.0264  -0.1059  0.7358      1e-04
s.e.   0.0842  0.0431  0.0431   0.0430   0.0430   0.0368  0.0788      8e-04
```

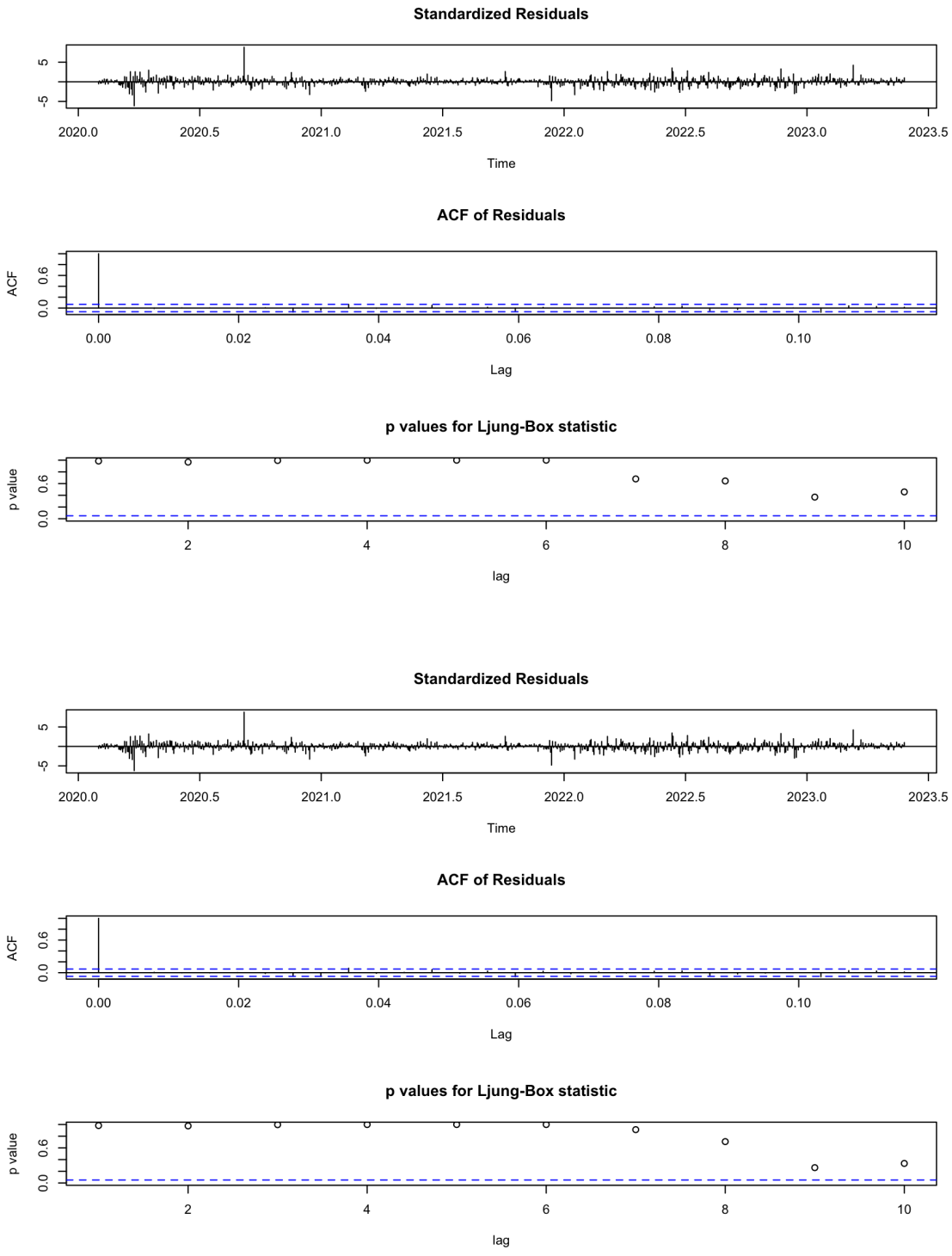
The summary shows that the ar6 component have an coefficient absolute value greater than 0.05, meaning that this components perform some effect in the model.

Then, We propose model ARIMA(1,0,6).

```
Call:
arima(x = lr_ts, order = c(1, 0, 6))

Coefficients:
         ar1     ma1      ma2     ma3      ma4      ma5      ma6  intercept
      -0.7025  0.6829  -0.0041  0.0101  -0.0358  -0.0253  -0.0943      1e-04
s.e.   0.1040  0.1086   0.0420  0.0424   0.0461   0.0408   0.0369      8e-04
```

Similarly, the ma6 component have an coefficient absolute value greater than 0.05, meaning that this components perform some effect in the model.

We conducted model diagnostic by checking the residuals. We ensured model residuals fell within acceptable bounds and exhibited properties consistent with white noise processes. This validation process confirmed the adequacy of our selected models and their suitability for forecasting purposes.



The diagnostics show that the residuals are not correlated with each other, meaning that both of our proposed model works.

When we compare the AIC and BIC, though the two values are similar for both graph, the model ARIMA(6,0,1) performs slightly better than ARIMA(1,0,6).

```
> AIC(crm_106)
[1] -3669.223
> BIC(crm_106)
[1] -3626.654
> AIC(crm_601)
[1] -3673.835
> BIC(crm_601)
[1] -3631.267
```

Thus, we choose our proposed model ARIMA(6,0,1) as our model for prediction for CRM price. The model parameters indicate significant autoregressive and moving average components, suggesting the stock price exhibits both momentum and mean-reverting characteristics.

3.1.2 Multiple Linear Regress (MLR) Models

Linear regression serves as a fundamental forecasting method. Multiple Linear Regression extends the basic model by incorporating various market factors, enabling deeper analysis of market behaviors.

Our methodology implements recursive prediction, with the initial model expressed as:

$$lnR_{i,t+1} = \beta_1 lnR_{i,t} + \beta_2 lnR_{i,t-1} + \beta_3 lnR_{\mathrm{QQQ}_t} + \beta_4 lnR_{\mathrm{SPX}_t}$$

where $lnR_{i,t+1}$ represents the predicted log return, $lnR_{i,t}$ and $lnR_{i,t-1}$ represent lagged log return, and lnR_{QQQ_t}, lnR_{SPX_t} represent log return of market indices at time t.

Model validation employs multiple diagnostic tests. We examine R-squared values and adjusted R-squared to assess model fit, F-statistics for overall significance, and t-statistics for individual variable significance. Residual analysis confirms assumptions of normality, homoscedasticity, and independence.

The recursive prediction process follows these steps:

1. Predict future price using lagged values
2. Update the dataset with predicted values
3. Adjust lagged variables for subsequent predictions
4. Iterate for next time period prediction

This recursive approach enables continuous updating of predictions while maintaining the temporal structure of the data.

3.1.3 Long Short Term Memory Models

Long Short-Term Memory networks represent an advanced deep learning architecture specifically designed for sequential data analysis. Unlike traditional neural networks, LSTM's unique memory cell structure enables effective capture of long-term dependencies in financial time series.

Our LSTM implementation utilizes a sequential architecture with hyperparameter tuning across multiple configurations. The model explores various hyperparameter combinations:

- **LSTM units**: 50, 100, 200
- **Activation functions**: ReLU, tanh
- **Learning rates**: 0.001, 0.01, 0.1

Training configuration employs:

- Batch size of 32
- 300 epochs
- Adam optimizer
- Mean Squared Error loss function

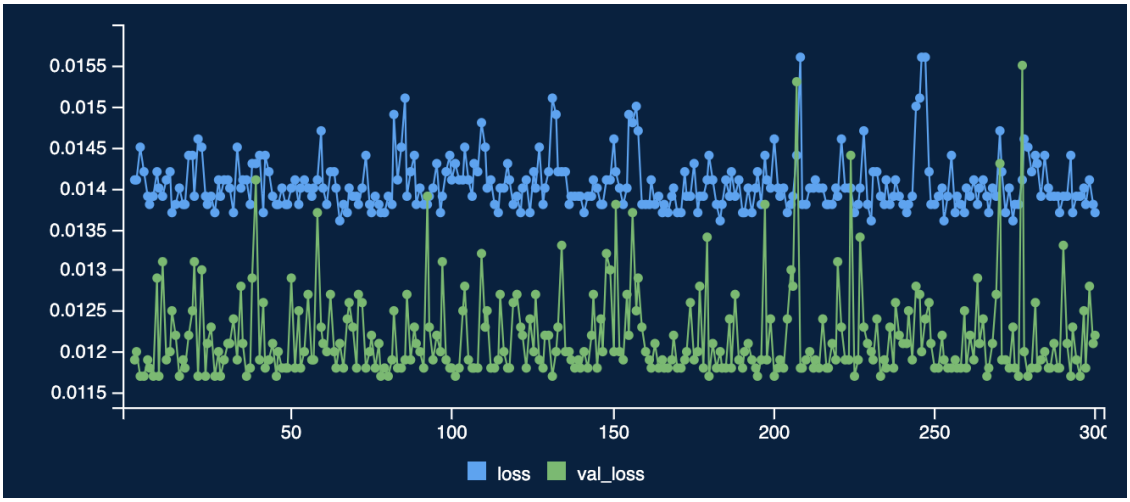
The input features are structured as a 3D array (samples $\times$ timesteps $\times$ features), with normalized log returns as the primary input. Using the same methodology of MLR model, we perform recursive prediction method. We first perform a single-step prediction for model evaluation. Then, we feed the prediction back into the next step prediction. We perform the same method iteratively for 25 times. Each time, we use the last value in the period as our predicted value for 1 step ahead prediction.

Model performance is evaluated using Root Mean Square Error (RMSE) as the primary metric, with the best-performing model selected based on minimum RMSE on the test set.

The best model has the features, and hyperparameters in the following graph:

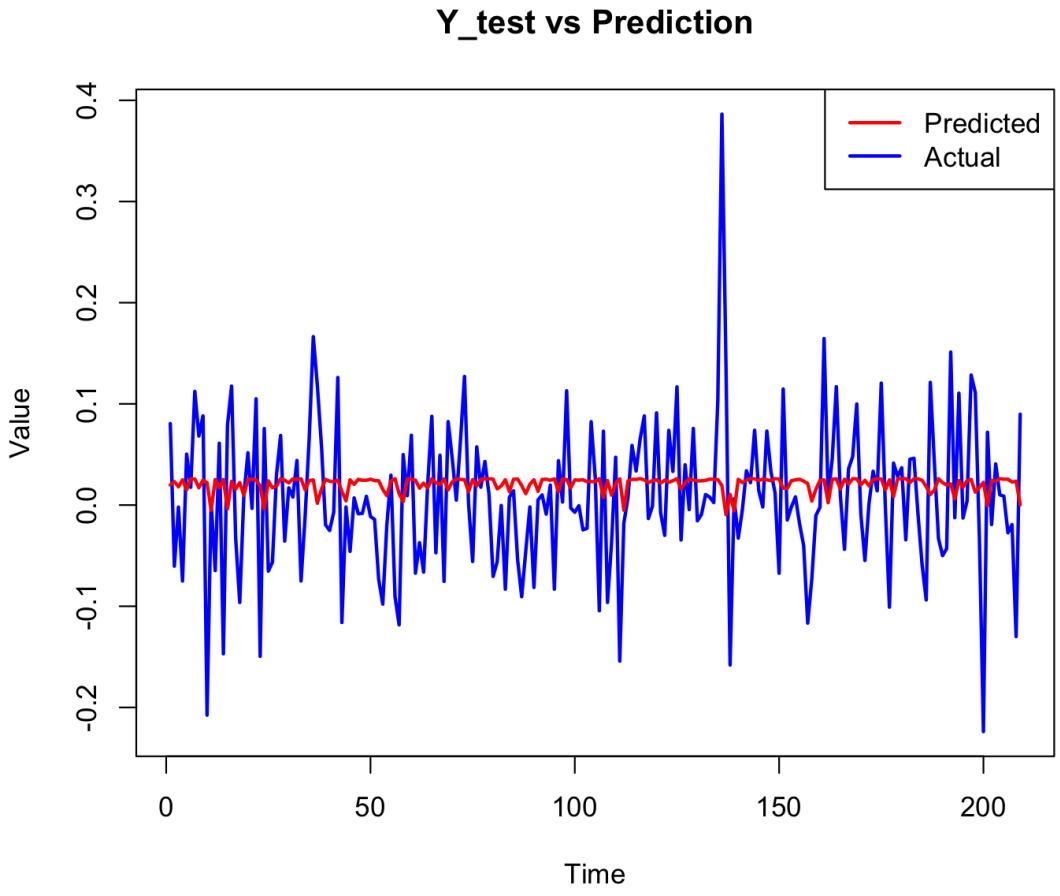
```
> best_learning_rate
[1] 0.001
> best_activation
[1] "tanh"
> best_lstm_model
Model: "sequential_303"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_303 (LSTM)             (None, 200)               161600
 dense_303 (Dense)           (None, 1)                 201
=================================================================
Total params: 161801 (632.04 KB)
Trainable params: 161801 (632.04 KB)
Non-trainable params: 0 (0.00 Byte)
```

During the training of the model, the loss plot indicates several issues.



Our model shows non-decreasing loss, and high variablity, which indicates that the training is not as effective.

This is also shown in the test, where predicted captures most of the flutations of values, but the variance is lower.

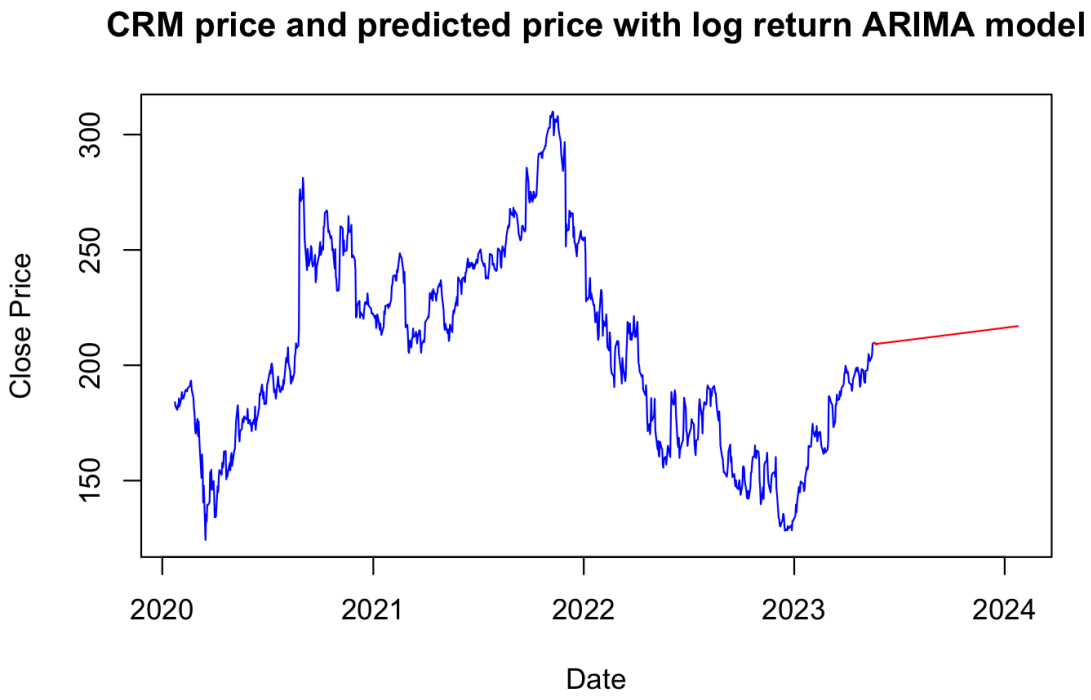


3.2 Fitting the Frequentist Model(s)

3.2.1 Fitting Time Series Models

We found that time series data is more effective for fitting short-term future data. However, as the forecasting period extends, the variance of the predictions increases, limiting the model's ability to provide meaningful insights.

Additionally, we observed that the ARIMA model predicts a slight increase in the mean stock price.



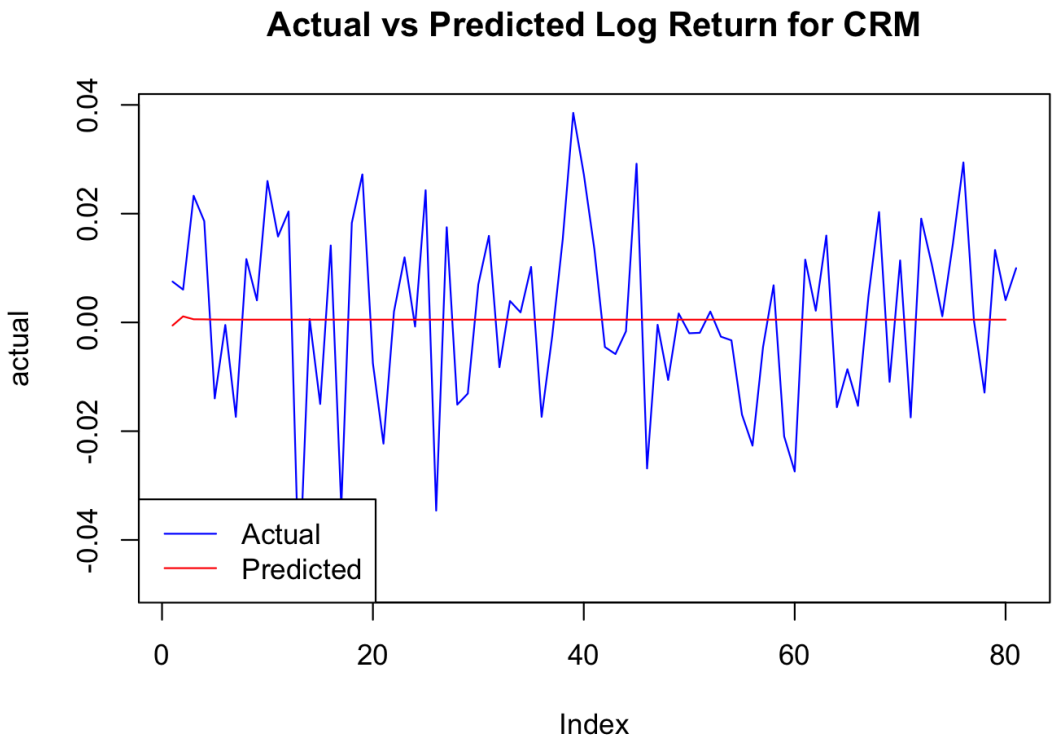
3.2.2 Fitting the MLR Models

Our Multiple Linear Regression model encounters same challenges in forecast accuracy. As predictions extend further from known data points, subsequent forecasts increasingly rely on previously predicted values rather than actual data. This dependency creates a compounding effect, leading to expanded variance and prediction intervals over time.

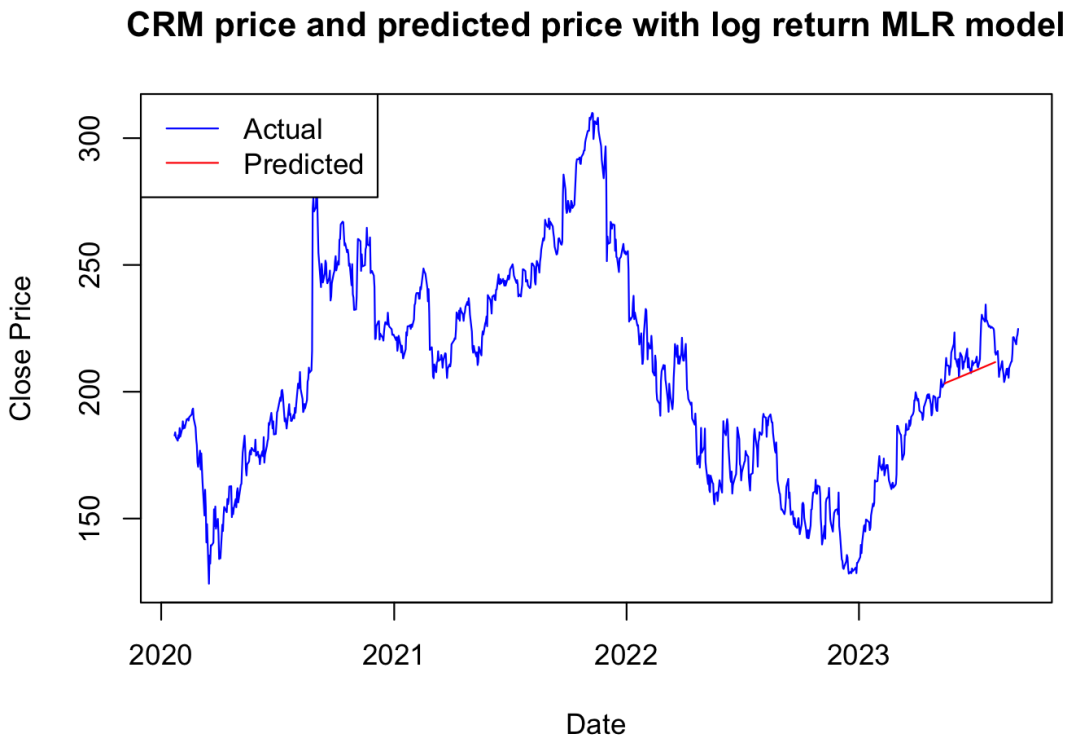
Analysis of the model coefficients, as shown in the regression summary, reveals that the most significant predictor variable is the immediate lagged price, with the highest t-value and lowest p-value. This finding aligns with market efficiency principles, suggesting that recent price information captures the most relevant market signals.

```
Call:
lm(formula = crm_lr ~ CRM_lag1 + CRM_lag2 + QQQ_lag1 + SPX_lag1,
    data = crm_train)

Residuals:
      Min        1Q    Median        3Q       Max
-0.139530 -0.011442  0.000001  0.012566  0.228017

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.0001748  0.0008447   0.207 0.836067
CRM_lag1     0.1079351  0.0495223   2.180 0.029549 *
CRM_lag2     0.0177332  0.0328564   0.540 0.589522
QQQ_lag1     0.1865537  0.1444613   1.291 0.196902
SPX_lag1    -0.5916403  0.1554658  -3.806 0.000151 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.02552 on 908 degrees of freedom
  (2 observations deleted due to missingness)
Multiple R-squared:  0.03435,   Adjusted R-squared:  0.03009
F-statistic: 8.074 on 4 and 908 DF,  p-value: 2.132e-06
```

However, we can see from the table that the R-squared value are small, meaning that only about 3% of the variance in crm_lr is explained by the predictors. This shows that the MLR approach is not as effective.



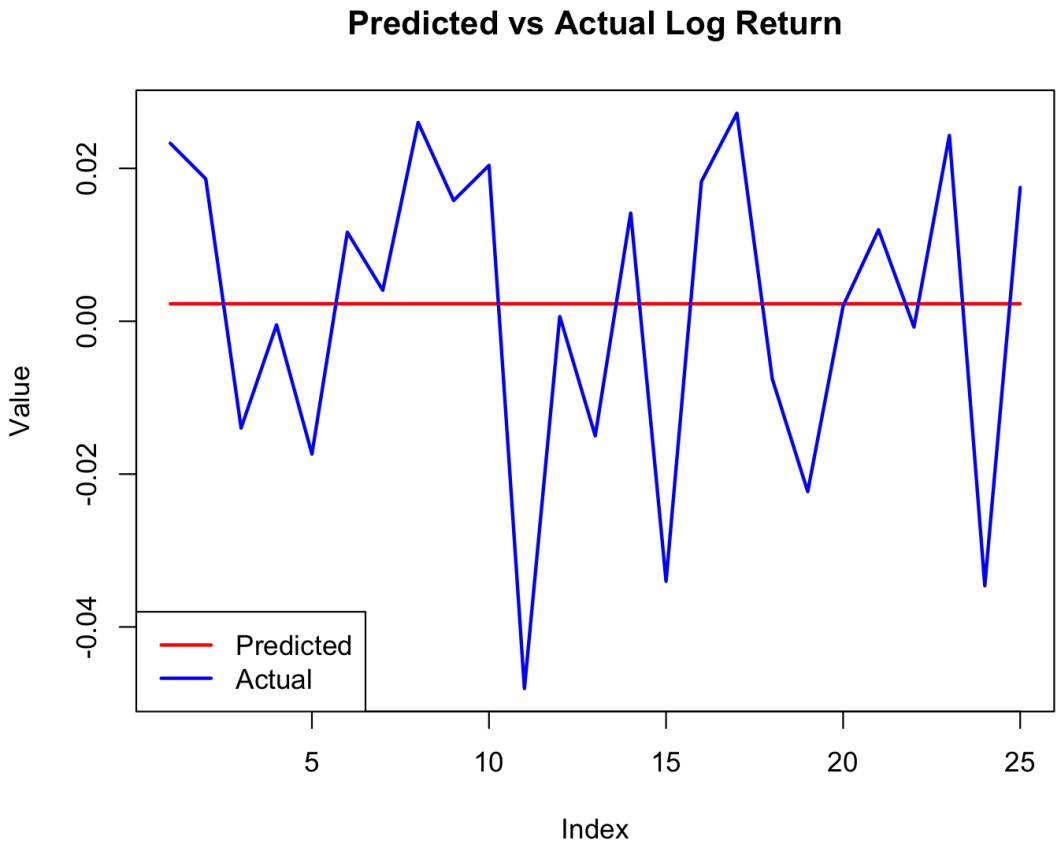
The visualization shows that the MLR prediction for log return falls in the median value of the actual log returns, but does not captures the changes at each time. This behavior underscores the limitations that the model may not be helpful on insights for short-term investment, but could only capture the long term trend.



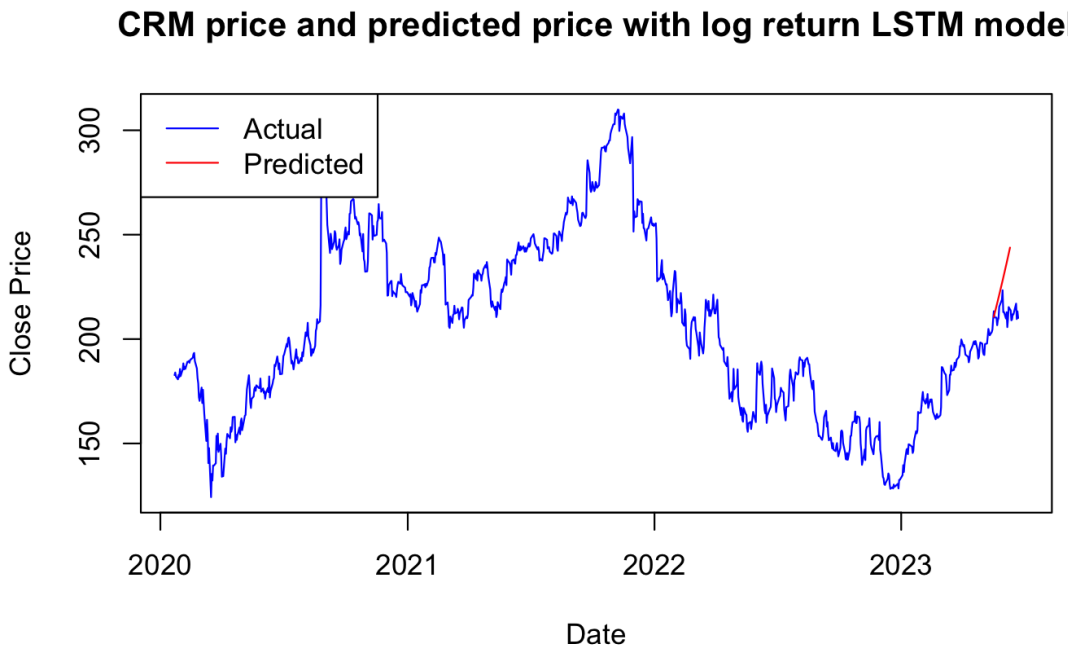
From the plot of price, we can see that the MLR prediction captures the correct trend of the stock, but does not provide details in price fluctuations.

3.2.3 Fitting LSTM Models

The prediction shows a similar pattern for LSTM model as the MLR Model.



We transform the log return back to the price to show the prediction.



We can see that same as MLR, LSTM model also captures the overall increasing trend, but does not capture the fluctuations.

4. Bayesian Analysis

4.1 Proposed Bayesian Model(s)

4.1.1 Bayesian ARMA Model

The Bayesian Model that will be explore in more depth is the Bayesian version of the frequentist ARMA model. Our model for the log return at time t is given by

$$R_t = \phi(R_{t-1}) + \epsilon_t, \epsilon_t \sim N(0, \sigma)$$

where R_t represents the log return at time t, ϕ represents the autocorrelation coefficient and σ is the volatility of the stock.

We will further make the assumptions that the returns at any time t has mean 0, and that the error term ϵ_t is not correlated with the returns at the previous time $t-1$. Additionally, we are conducting this analysis under the condition that the process is stationary, i.e $|\phi| < 1$. Given this, we can obtain the distribution for R_t by computing the variance:

$$\begin{aligned} Var[R_t] &= Var[\phi(R_{t-1}) + \epsilon_t] \\ &= \phi^2 Var[R_{t-1}] + Var[\epsilon_t] + 2\phi Cov(R_{t-1}, \epsilon_t) \\ &= \phi^2 Var[R_t] + \sigma^2 + 0 \\ &= \frac{\sigma^2}{1-\phi^2} \end{aligned}$$

Thus, we have that

$$R_1 \sim N\Big(0, \sqrt{\frac{\sigma^2}{1-\phi^2}}\Big) = N\Big(0, \frac{\sigma}{\sqrt{1-\phi^2}}\Big)$$

Then we have that for every future R_t, since we have R_{t-1}, it is distributed as

$$R_t \sim N\Big(\phi R_{t-1}, \sigma\Big)$$

For the priors on σ and ϕ, we will use the training data to compute approximate normal distributions for them. Then, to compute future returns, we will use the 20 previous closing prices prior to time t as the likelihood, and compute the posterior distribution for R_t for all t in the test data set.

4.1.2 Markov Chain Monte Carlo

This method refers to the method in the article "An alternative simulation method for stock price forecasting using Python" (Leung, n.d.).

We assume that the stock price movement can be categorized into discrete states, and the probability of moving to a new state depends only on the current state. We define a transition matrix and state-dependent price changes. The model then simulates state transitions and applies the corresponding price changes.(Leung, n.d.)

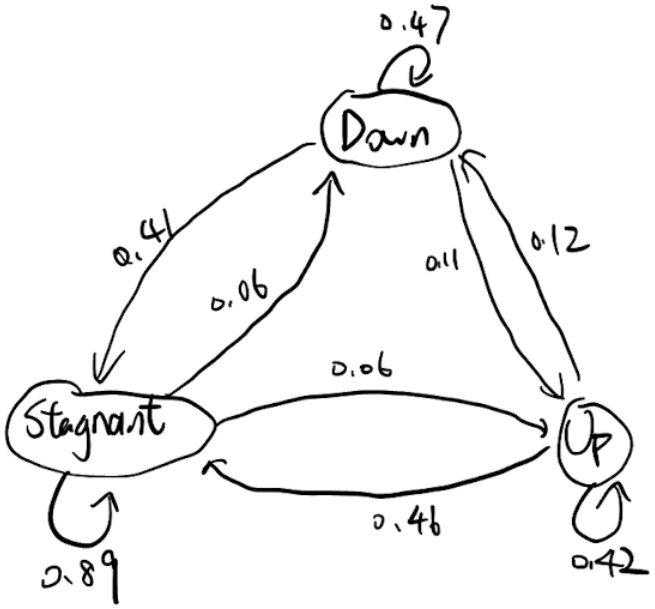
We define states of stock price going up, stagnant, or down by considering if the log return lies over, between, or below the upper threshold and lower threshold. We define the threshold as follows:

- upper threshold = mean log return + std log return
- lower threshold = mean log return - std log return

Based on the training data, our transition matrix shows as follows:

	Down	Stagnant	Up
Down	0.47407407	0.41481481	0.11111111
Stagnant	0.05633803	0.88631791	0.05734406
Up	0.12000000	0.45600000	0.42400000

For better visualization, we can view it as a weighted directed graph:



where the weight of State A -> State B can be interpreted as the probability from State A to transition to State B in the next step.

For stan model, we define a probability vector representing the transition probabilities to all other states as a parameter.

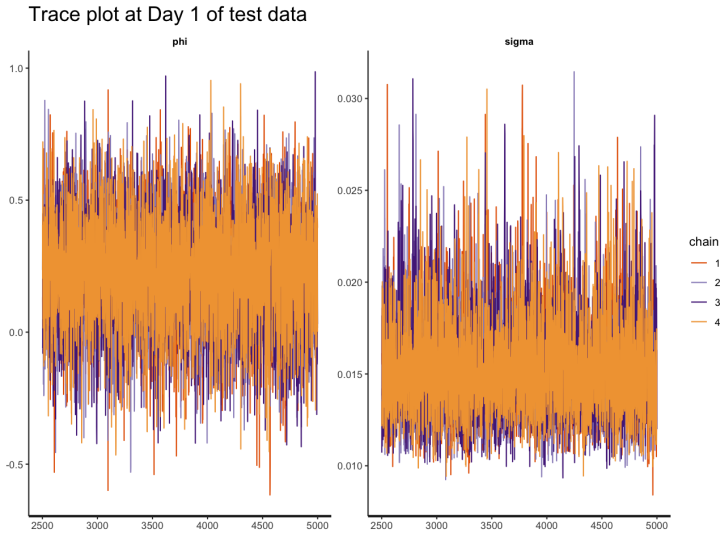
- We use the **Dirichlet Prior** for each row of the transition matrix, as the nature of this prior produces probability vectors that sum to 1, which fits the situation that each row of the transition matrix sum to 1.
- We use **Categorical Distribution** to model the probability of transitioning between states. The model fits the situation because each state transition represent a single choice from the possible next states, and the probability of all state transition sum to 1.

4.2 Fitting the Bayesian model(s)

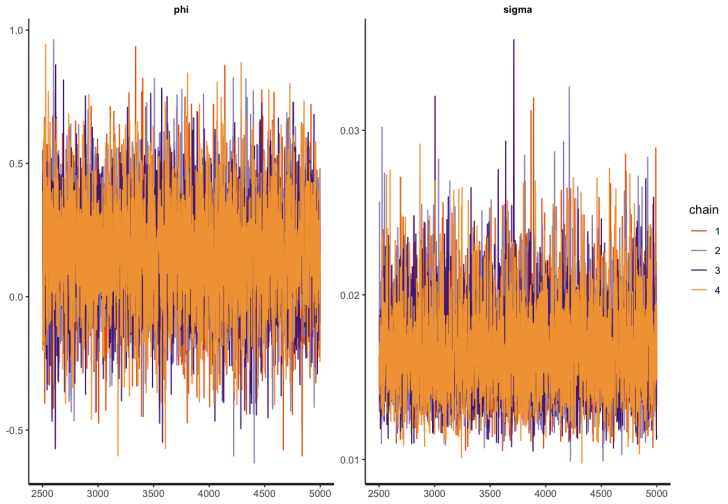
4.2.1 Fitting Bayesian ARMA Model

The models were fitted through Bayesian inference using the stan() function from the rstan package.

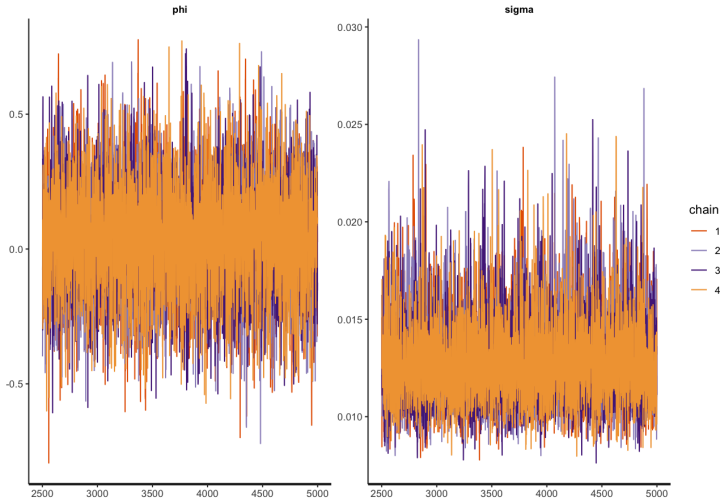
Sensitivity analysis was done through looking at the trace plots at 50 day intervals of the test data. To ensure convergence, we used 5000 iterations and 4 chains for each fit. The trace plots are as follows:



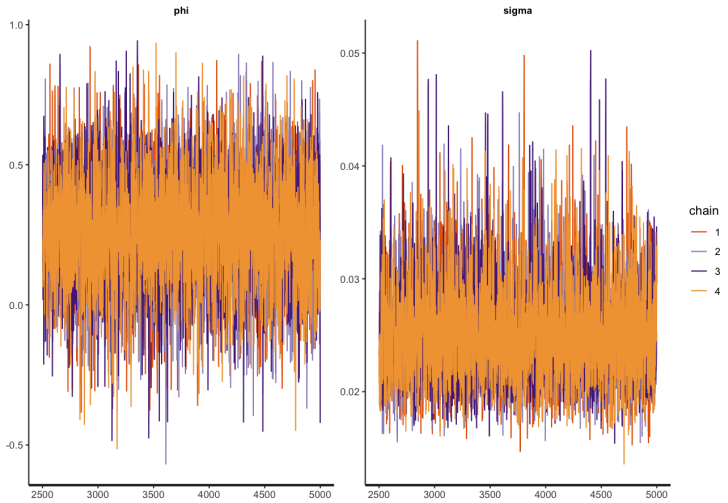
Trace plot at Day 51 of test data



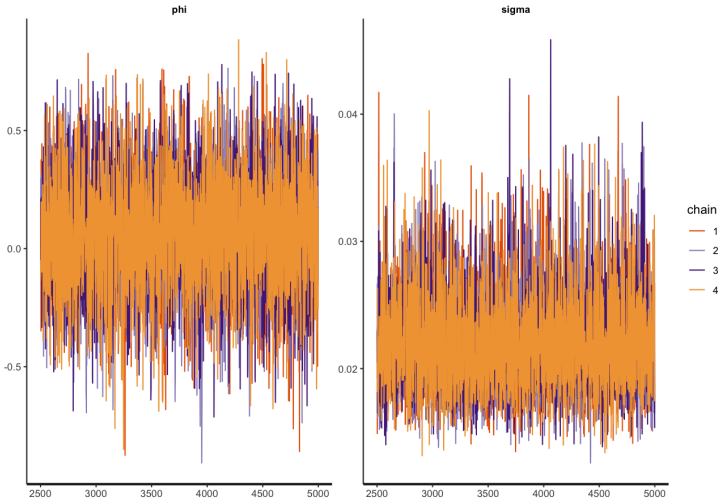
Trace plot at Day 101 of test data



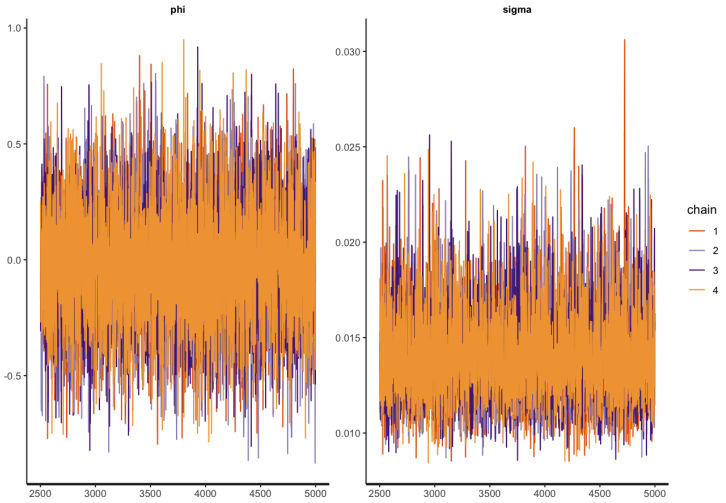
Trace plot at Day 151 of test data



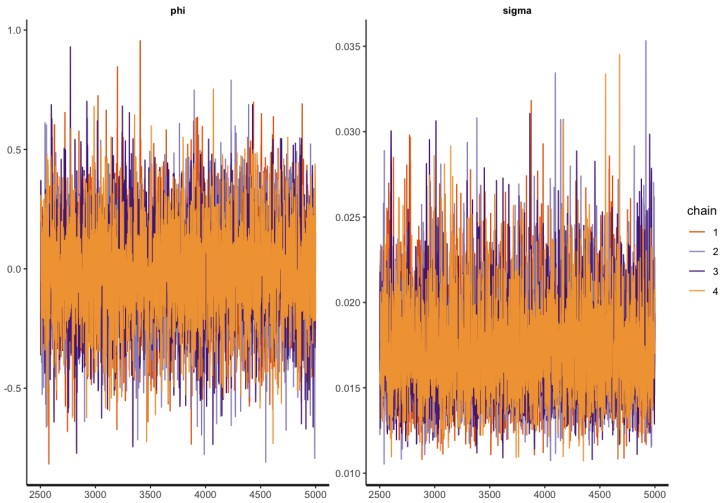
Trace plot at Day 201 of test data



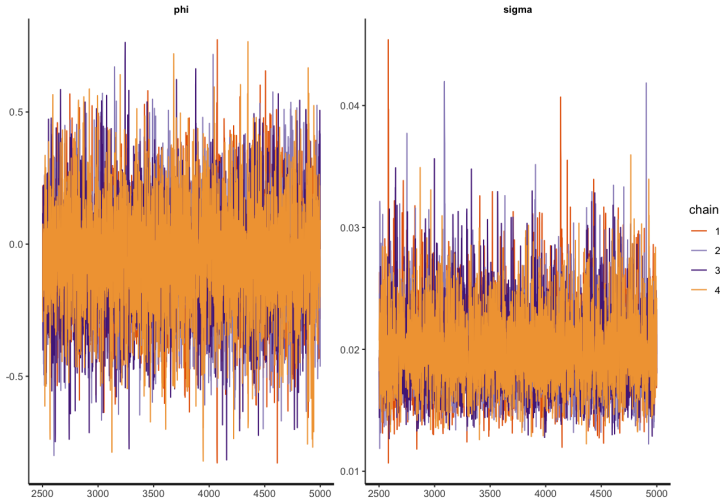
Trace plot at Day 251 of test data



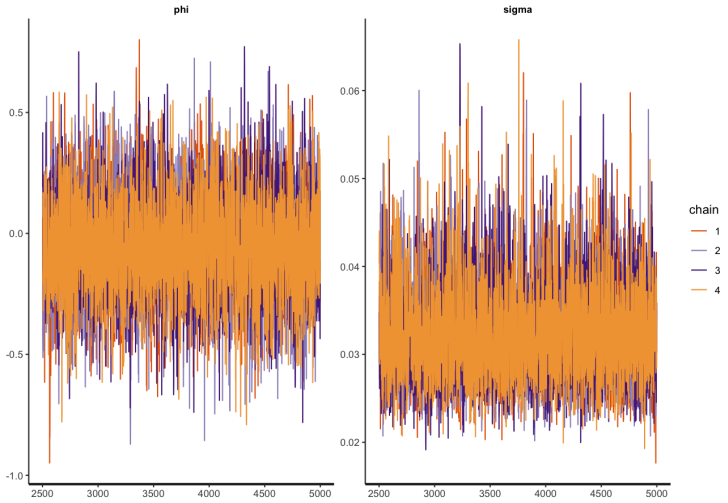
Trace plot at Day 301 of test data



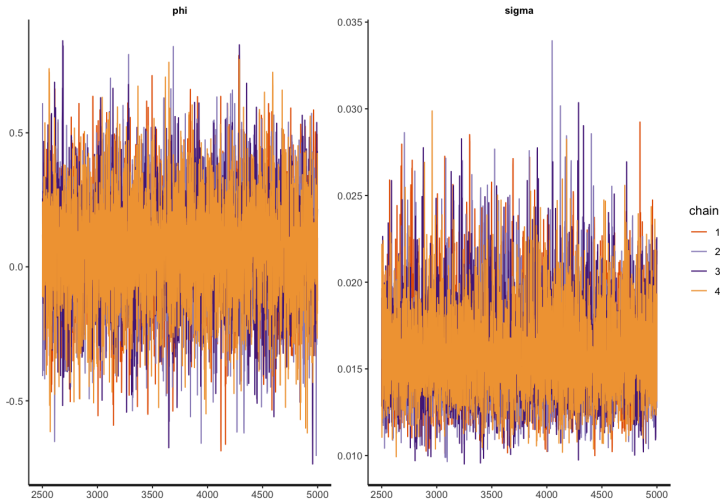
Trace plot at Day 351 of test data



Trace plot at Day 401 of test data



Trace plot at Day 418 of test data

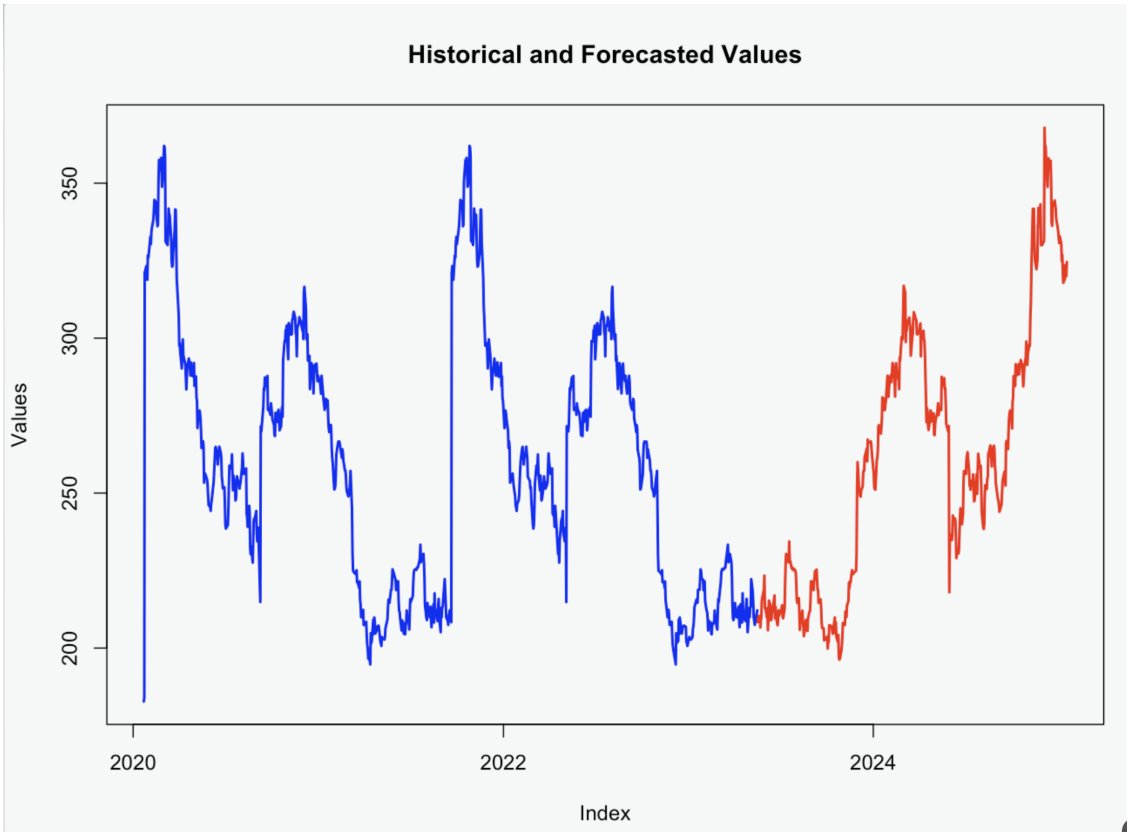


We can see from all the trace plots that there seems to be very good mix between the chains, and suggest convergence.

To additionally check MCMC convergence, we used the monitor() function from the rstan package for each fit, and confirmed that all Rhat's satisfied Rhat < 1.05.

Prediction

To make prediction on the log return, we conducted a random draw from the posterior distribution, $R_t \sim N\Big(\phi R_{t-1}, \sigma\Big)$, as derived earlier. We then did this at every test data point, and converted the log returns into a predicted price. Below is a graph showing the two predictions



From the graph we can see that the bayesian prediction captures the fluctuation of prices.

4.2.2 Fitting Markov Chain Monte Carlo

We preprocess data by mapping and labeling the states. The result after fitting the model shows as follows:

```
Inference for Stan model: anon_model.
4 chains, each with iter=2000; warmup=1000; thin=1;
post-warmup draws per chain=1000, total post-warmup draws=4000.

                       mean se_mean   sd 2.5%  25%  50%  75% 97.5% n_eff Rhat
transition_matrix[1,1] 0.47       0 0.04 0.38 0.44 0.47 0.50  0.56  6084    1
transition_matrix[1,2] 0.41       0 0.04 0.33 0.38 0.41 0.44  0.50  5572    1
transition_matrix[1,3] 0.12       0 0.03 0.07 0.10 0.11 0.13  0.17  5262    1
transition_matrix[2,1] 0.06       0 0.01 0.04 0.05 0.06 0.06  0.07  5608    1
transition_matrix[2,2] 0.88       0 0.01 0.86 0.88 0.88 0.89  0.90  5521    1
transition_matrix[2,3] 0.06       0 0.01 0.04 0.05 0.06 0.06  0.07  5084    1
transition_matrix[3,1] 0.12       0 0.03 0.07 0.10 0.12 0.14  0.19  5459    1
transition_matrix[3,2] 0.45       0 0.04 0.37 0.42 0.45 0.48  0.54  5536    1
transition_matrix[3,3] 0.42       0 0.04 0.34 0.39 0.42 0.45  0.51  5694    1
```

We see that for all transition matrices, the effective sample size is large enough, and Rhat $= 1$, which suggests that the chain converges.

Prediction

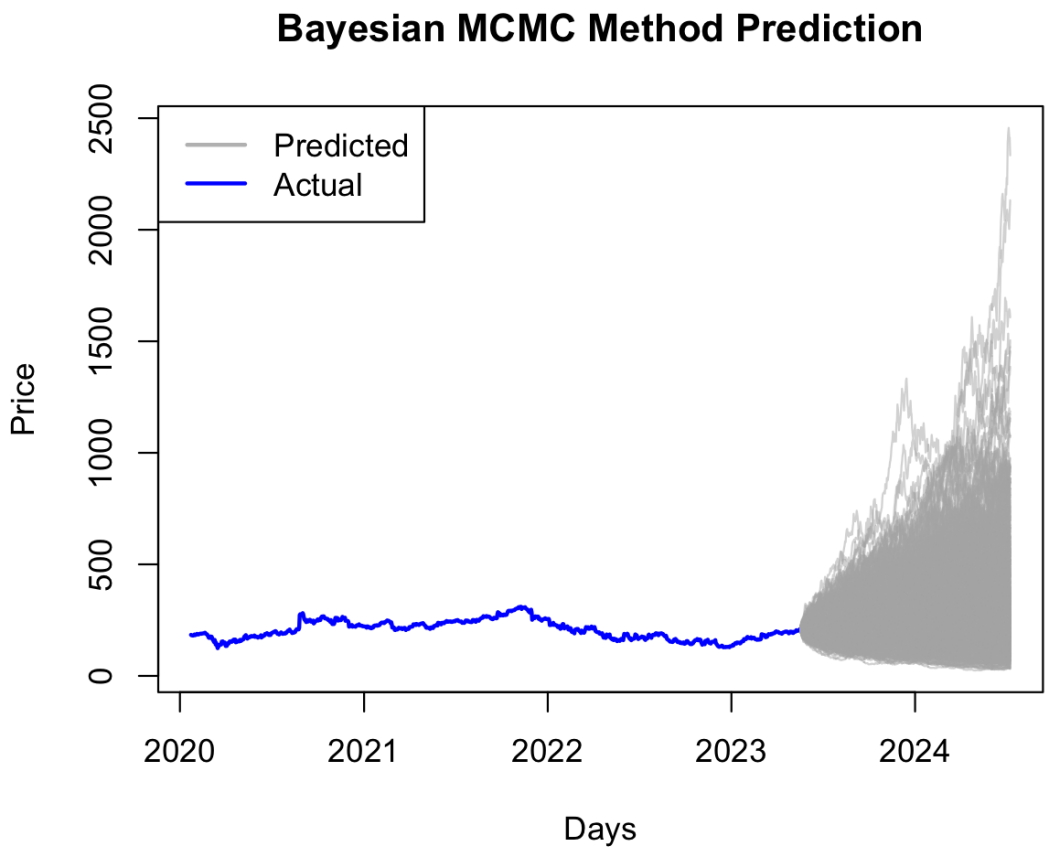
To predict the stock price, we randomly sample one posterior distribution of transition matrices. We determine the next state based on transition probabilities.

We then generate log returns using normal distributions based on each state.

- Up State $\sim N(\mu + \sigma, \sigma)$
- Down State $\sim N(\mu - \sigma, \sigma)$
- Stagnant State $\sim N(\mu, \sigma)$

where μ is the mean log return, σ is the standard deviation of the log return.

We then update the price prediction using the log return obtained.



The result shows that the chain gives a wide range of possible prices in predition, where the fan-like structure shows the increase in uncertainty of price.

5. Discussion

Our comprehensive analysis of CRM stock prediction using multiple modeling approaches reveals several key insights and limitations:

Model Performance Comparison

The three models - ARIMA(6,0,1), Multiple Linear Regression (MLR), and LSTM - each demonstrated distinct strengths and limitations:

- **ARIMA Model**: Showed better performance in short-term predictions but does not perform well in long term predictions because of the iterative approach.

- **MLR Model**: Despite its simplicity, achieved only modest predictive power with an R-squared value of approximately 3%. While it captured general trends, it failed to model the intricate price fluctuations characteristic of stock movements.
- **LSTM Model**: Demonstrated potential in capturing non-linear relationships but suffered from training instability. The model's complexity did not translate to significantly better performance compared to simpler approaches.
- **Bayesian ARMA Model**: Provides superior predictive effect compared to frequentist approaches. Captures the fluctuations instead of just showing an overall trend.
- **MCMC Model**: Generates a wide range of possible future price trajectories that capture uncertainty in the market. The fan-like spread of predictions illustrates the increasing uncertainty in price movements over longer time, which is a more realistic representation of market behavior than single-point forecasts.

Methodological Insights

1. **Data Transformation**:

- The log return transformation proved effective in achieving stationarity and normalizing the data distribution
- This transformation successfully stabilized variance across the time series, facilitating more reliable predictions

2. **Prediction Horizon**:

- All models showed degrading performance with extended prediction windows
- Short-term predictions demonstrated higher reliability than longer-term forecasts
- The compounding effect of prediction errors significantly impacts long-term forecast accuracy

3. **Market Index Integration**:

- The inclusion of market indices (QQQ, SPX) provided valuable context but did not substantially improve prediction accuracy
- The high correlation between individual stocks and market indices suggests potential redundancy in the predictors

Limitations and Future Improvements

1. **Model Enhancement Opportunities**:

- Incorporate additional technical indicators (volatility measures)
- Implement adaptive learning rates for LSTM to address training instability
- More informative prior and more accurate likelihood in bayesian analysis

2. **Data Considerations**:

- Expand the feature set to include fundamental company metrics
- Consider alternative data sources such as sentiment analysis from financial news
- Investigate the impact of different time scales on prediction accuracy

3. **Validation Methodology**:

- Implement rolling window validation to better assess model stability
- Develop more robust performance metrics for comparing model effectiveness
- Consider economic significance alongside statistical significance

Practical Implications

The analysis suggests that while sophisticated models can capture market trends, the inherent unpredictability of stock prices limits their practical application. The models serve better as tools for understanding market dynamics rather than as definitive predictive instruments for trading decisions.

6. Contributions

Both team members, Andrew Su and Scarlett He, contributed equally to all aspects of this project, with each dedicating approximately 60 hours to its completion. The work was distributed collaboratively across all sections.

The team maintained consistent communication throughout the project, meeting regularly to discuss progress, challenges, and next steps. Both team members provided valuable insights and support to each other.

References

Leung, Simon. n.d. *An Alternative Simulation Method for Stock Price Forecasting Using Python*. https://medium.com/@simonleung5jobs/an-alternative-simulation-method-for-stock-price-forecasting-using-python-1bc1649edb99.

Nasdaq. n.d.a. *Invesco QQQ Trust, Series 1 (QQQ) Historical*. https://www.nasdaq.com/market-activity/stocks/qqq/historical.

———. n.d.b. *Salesforce, Inc. Common Stock (CRM) Historical Quotes*. https://www.nasdaq.com/market-activity/stocks/crm/historical.

———. n.d.c. *S&p 500 (SPX) Historical Data*. https://www.nasdaq.com/market-activity/index/spx/historical.

Appendix

https://raw.githubusercontent.com/asu1-1/STAT27410-Final-Project/refs/heads/main/27410%20Final%20Project%20Code.R

https://raw.githubusercontent.com/asu1-1/STAT27410-Final-Project/refs/heads/main/Scarlett_part1_code.R

https://medium.com/@simonleung5jobs/an-alternative-simulation-method-for-stock-price-forecasting-using-python-1bc1649edb99