AI Based Stock Market Predictions

Affiliated with the Erdős Institute

Yoshihiro Shirai[1] Xu Zhuang[2] Yiting Zhang[1]

[1]Department of Applied Mathematics, University of Washington
[2]Department of Mathematics, UC Irvine

December 1, 2024

Introduction

- **Goal:** Test the capability of different machine learning methods in predicting the close of business (COB) mid price S_t of 3 stocks, given the previous 10 days of COB prices $S_{t_{10}}, \ldots, S_{t_1}$. The three stocks are SPY, AAPL, and IBM.
- **Data Source:** Wharton Research Dataset.
- **Methods:** Linear Regression, ARIMA, RNN, LSTM.
- **Input:** $\{X_{t-h}\}_{h=1,\ldots,10}$, where $X_{t-h} = \log(\frac{S_{t-h}}{S_{t-h-1}})$, $h = 1, \ldots, 10$.
- **Objective:** Compare these approaches and refine our modeling pipeline for improved stock price prediction accuracy.

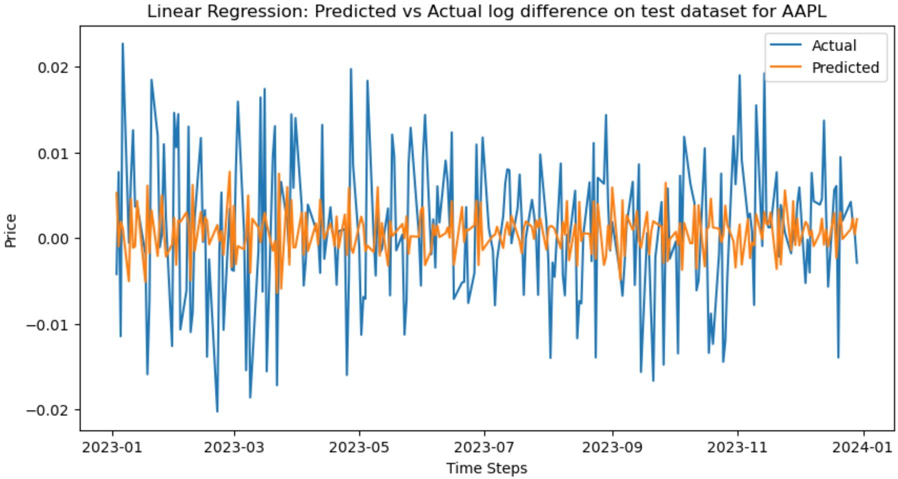
Linear Regression

- **Overview:** Supervised learning algorithm predicting a continuous target Y using features $X_1, X_2, \ldots, X_n$.
- **Model:** $Y = \mathbf{X}\boldsymbol{\beta} + \epsilon$, where:
  - $\mathbf{X} = [1, X_1, X_2, \ldots, X_n]$: Feature vector (including intercept)
  - $\boldsymbol{\beta} = [\beta_0, \beta_1, \ldots, \beta_n]^\intercal$: Coefficient vector
  - ϵ: Error term
- **Training:** Estimate $\boldsymbol{\beta}$ by minimizing mean squared error (MSE):

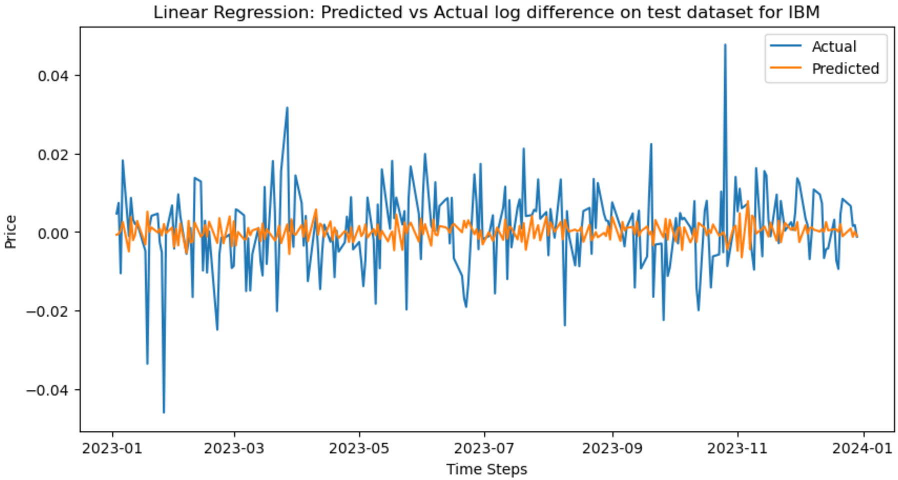
$$\text{MSE} = \frac{1}{m}\sum_{i=1}^{m}(Y_i - \hat{Y}_i)^2, \quad \hat{Y}_i = \mathbf{X}_i\boldsymbol{\beta},$$

  where m is the number of training samples.
- **Evaluation:** Apply $\boldsymbol{\beta}$ to the test sets to predict $\hat{Y}$ and assess performance using MSE.

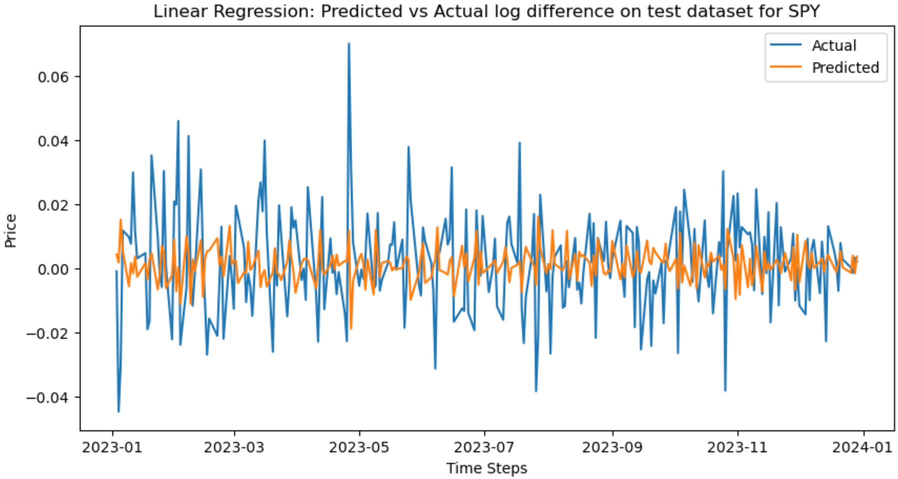
Linear Regression



Linear Regression



Linear Regression



Linear Regression

Ticker	Train MSE	Validation MSE	Test MSE	Signal
AAPL	0.0040	0.0062	0.0018	0.604
IBM	0.0054	0.0042	0.0018	0.540
SPY	0.0037	0.0064	0.0031	0.604

Table: Mean Squared Error and Signal for each ticker using Linear Regression

Linear Regression provides reasonable predictions, but performance varies across different stocks. Further refinement of the model or more complex algorithms may improve accuracy.

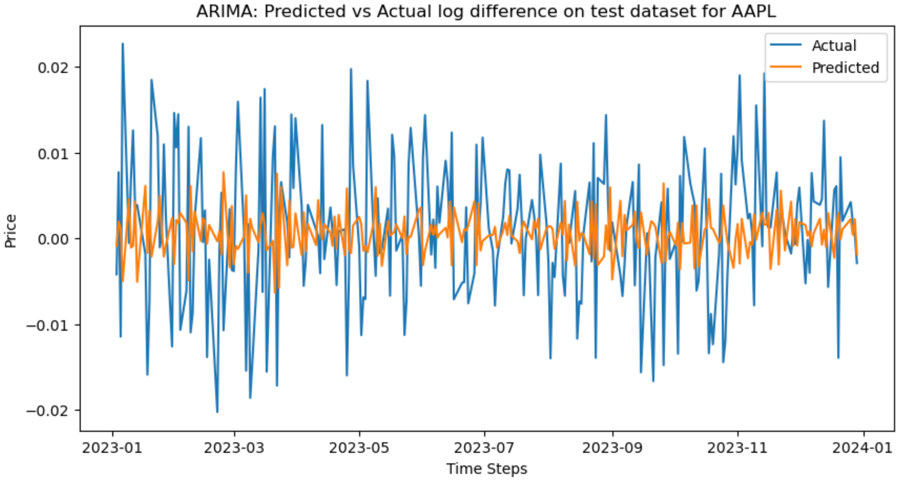
ARIMA

- **Overview:** A popular statistical model for time series forecasting, particularly suitable for univariate data.
- **Components:**
  - **AR (AutoRegressive):** Uses past values to predict future values.
  - **I (Integrated):** Differencing the data to achieve stationarity by removing trends.
  - **MA (Moving Average):** Models the relationship between current values and past error terms.
- **Model:**

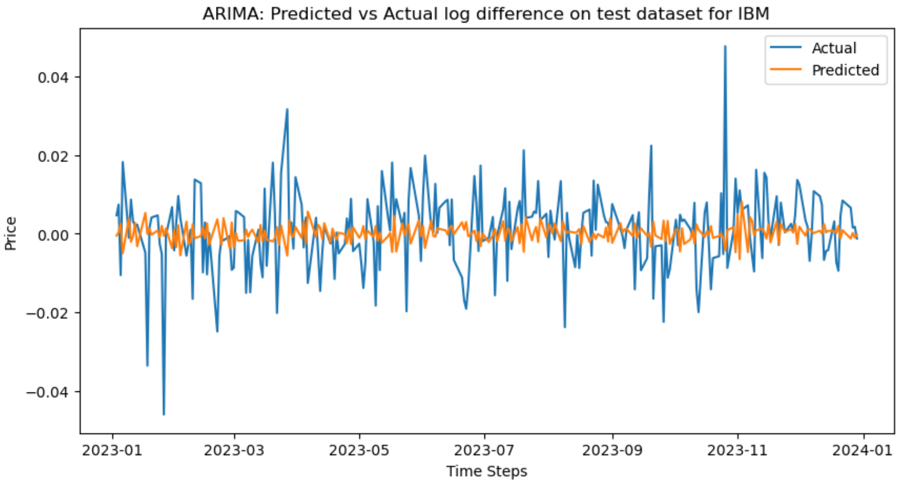
$$\mathrm{ARIMA}(p, d, q)$$

  where:
  - p: Number of autoregressive lags
  - d: Degree of differencing
  - q: Number of moving average terms
- **Usage:** Widely used for forecasting by capturing temporal structures in time series data.

ARIMA

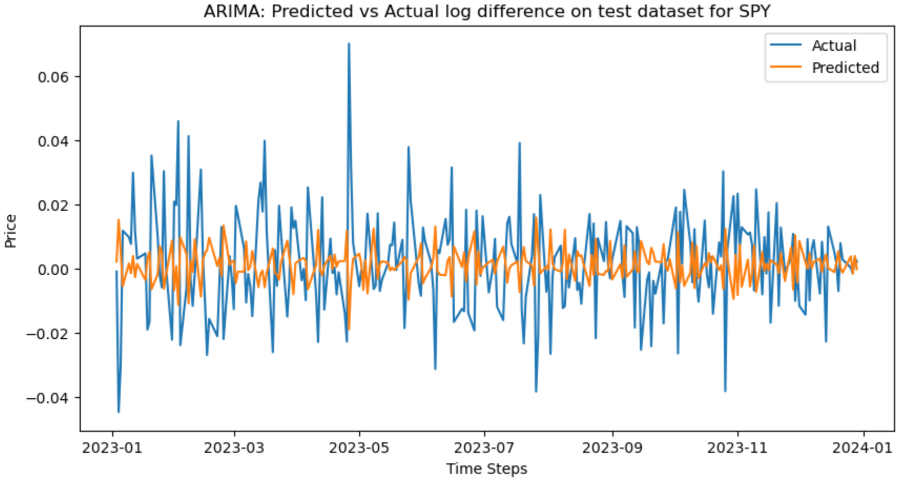

ARIMA: Predicted vs Actual log difference on test dataset for AAPL
Actual
Predicted
Price
0.02
0.01
0.00
−0.01
−0.02
2023-01
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
Time Steps

ARIMA



ARIMA: Predicted vs Actual log difference on test dataset for IBM
Actual
Predicted
Price
0.04
0.02
0.00
−0.02
−0.04
2023-01
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
Time Steps

ARIMA


ARIMA: Predicted vs Actual log difference on test dataset for SPY
Actual
Predicted
Price
0.06
0.04
0.02
0.00
−0.02
−0.04
2023-01
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
Time Steps

ARIMA

Ticker	Train MSE	Valid MSE	Test MSE	Signal
AAPL	0.0040	0.0081	0.0022	0.6
IBM	0.0054	0.0048	0.0021	0.548
SPY	0.0037	0.0096	0.0047	0.604

Table: Mean Squared Error and Signal for each ticker using ARIMA.

- **AAPL:** Linear Regression achieves a lower Validation MSE compared to ARIMA, with similar Signal values.
- **IBM:** Both models have comparable Validation MSE, but ARIMA shows a slightly higher Signal.
- **SPY:** Linear Regression outperforms ARIMA with a lower Validation MSE; Signal values are identical.

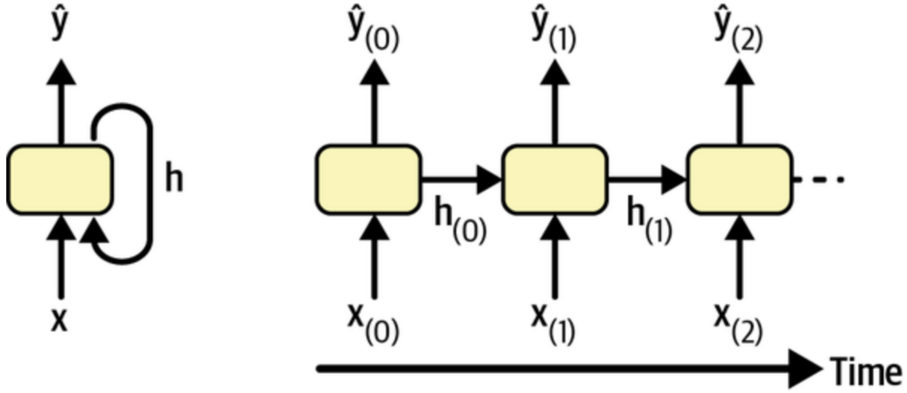
Simple RNN

- **Overview:** A fundamental neural network architecture for processing sequential data.
- **Mechanism:** Maintains a hidden state h_t that updates at each time step t based on the current input x_t and the previous hidden state h_{t-1}:

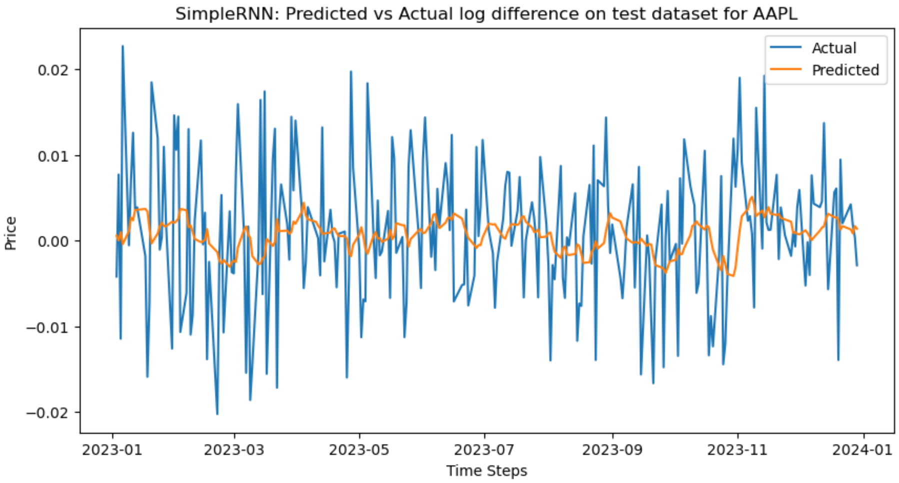
$$h_t = \phi(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$

  where:
  - x_t: Input at time step t
  - h_{t-1}: Previous hidden state
  - W_{xh}, W_{hh}: Weight matrices
  - b_h: Bias vector
  - ϕ: Activation function (e.g., tanh)
- **Applications:** Used for tasks like time series forecasting and sequence classification by capturing patterns and dependencies in sequential data.

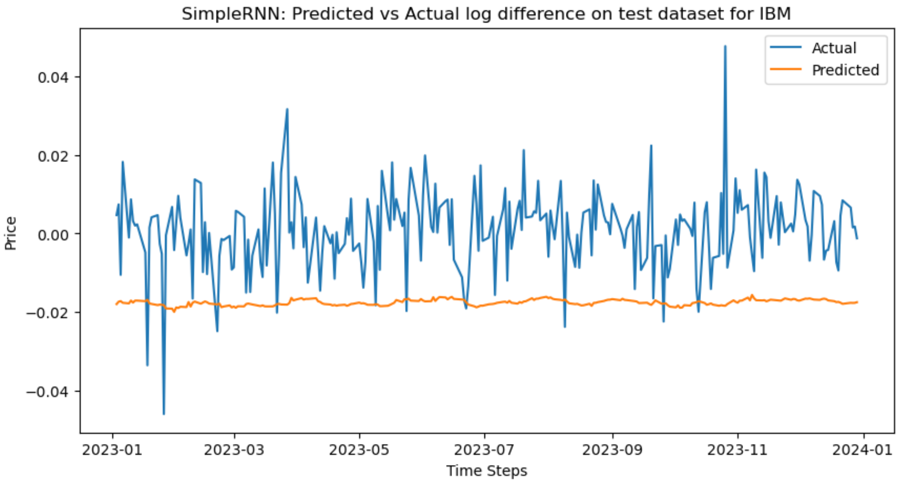
SimpleRNN



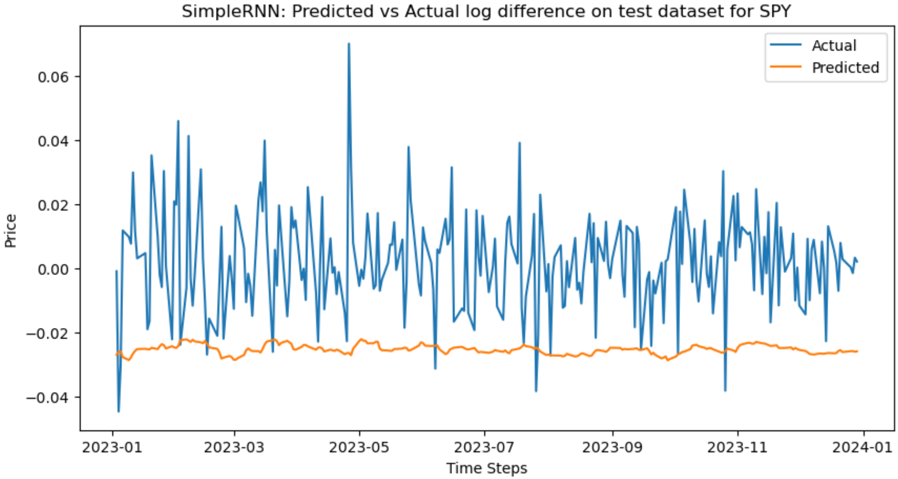
SimpleRNN


SimpleRNN: Predicted vs Actual log difference on test dataset for AAPL
Actual
Predicted
Price
0.02
0.01
0.00
−0.01
−0.02
2023-01
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
Time Steps

SimpleRNN


SimpleRNN: Predicted vs Actual log difference on test dataset for IBM
Actual
Predicted
Price
0.04
0.02
0.00
−0.02
−0.04
2023-01
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
Time Steps

SimpleRNN


SimpleRNN: Predicted vs Actual log difference on test dataset for SPY
Actual
Predicted
0.06
0.04
0.02
0.00
−0.02
−0.04
Price
2023-01
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
Time Steps

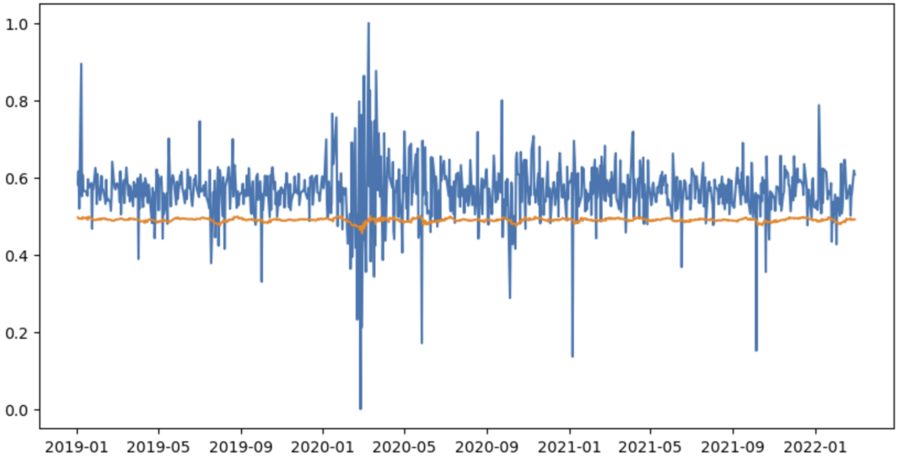
SimpleRNN

Ticker	Train MSE	Valid MSE	Test MSE	Signal
AAPL	0.0050	0.0063	0.0018	0.684
IBM	0.0110	0.0095	0.0072	0.
SPY	0.0128	0.0132	0.0115	0.

Table: Mean Squared Error and Signal for each ticker using SimpleRNN.

- **AAPL:** Simple RNN achieves similar Validation MSE as LR but with a higher Signal (0.684 vs. 0.604), indicating better correlation with actual prices. It also outperforms ARIMA in both metrics.
- **IBM and SPY:** Simple RNN underperforms compared to LR and ARIMA, as it is not able to explain the big showing higher Test MSE and negligible Signal, suggesting it did not capture the underlying patterns effectively for these stocks.

SimpleRNN


1.0
0.8
0.6
0.4
0.2
0.0
2019-01
2019-05
2019-09
2020-01
2020-05
2020-09
2021-01
2021-05
2021-09
2022-01

LSTM

- **Overview:** Long Short-Term Memory (LSTM) networks are specialized Recurrent Neural Networks (RNNs) designed to capture both short-term and long-term dependencies in sequential data.
- **Key Features:**
  - Incorporate memory cells and gating mechanisms to control the information flow.
  - Gates include:
    - **Forget Gate** (f_t): Decides what information to discard
    - **Input Gate** (i_t): Determines which new information to add
    - **Output Gate** (o_t): Controls the output based on the cell state
- **Advantages:** Effectively capture long-term dependencies and mitigate vanishing gradient problems common in traditional RNNs.
- **Applications:** Used in time series forecasting, language modeling, and speech recognition.

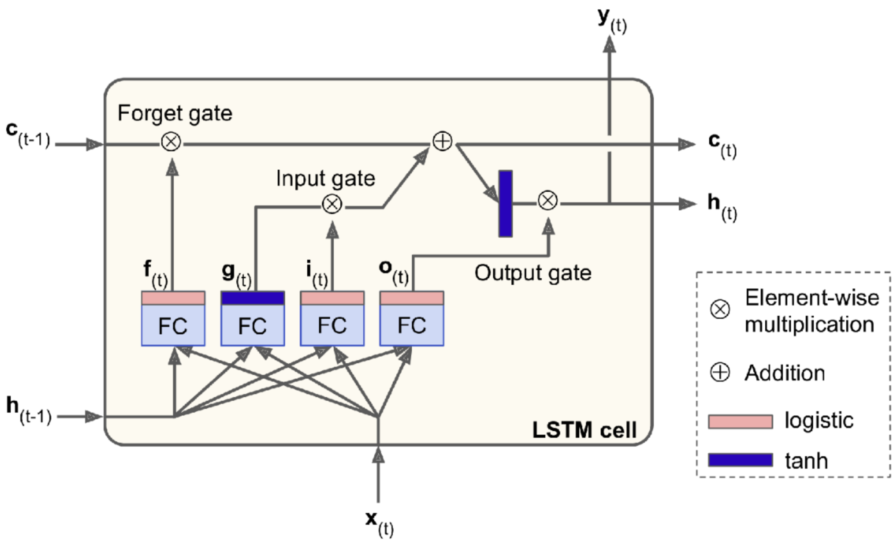
LSTM

- σ: logistic sigmoid function. The formulas for a LSTM are:

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_{x,i}^T \mathbf{x}_t + \mathbf{W}_{h,i}^T \mathbf{h}_{t-1} + \mathbf{b}_i),\ \mathbf{f}_t = \sigma(\mathbf{W}_{x,f}^T \mathbf{x}_t + \mathbf{W}_{h,f}^T \mathbf{h}_{t-1} + \mathbf{b}_f),\\
\mathbf{o}_t &= \sigma(\mathbf{W}_{x,o}^T \mathbf{x}_t + \mathbf{W}_{h,o}^T \mathbf{h}_{t-1} + \mathbf{b}_o),\\
\mathbf{g}_t &= \tanh(\mathbf{W}_{x,g}^T \mathbf{x}_t + \mathbf{W}_{h,g}^T \mathbf{h}_{t-1} + \mathbf{b}_g),\\
\mathbf{c}_t &= \mathbf{f}_t \otimes \mathbf{c}_{t-1} + \mathbf{i}_t \otimes \mathbf{g}_t,\ \mathbf{y}_t = \mathbf{h}_t = \mathbf{o}_t \otimes \tanh \mathbf{c}_t
\end{aligned}$$

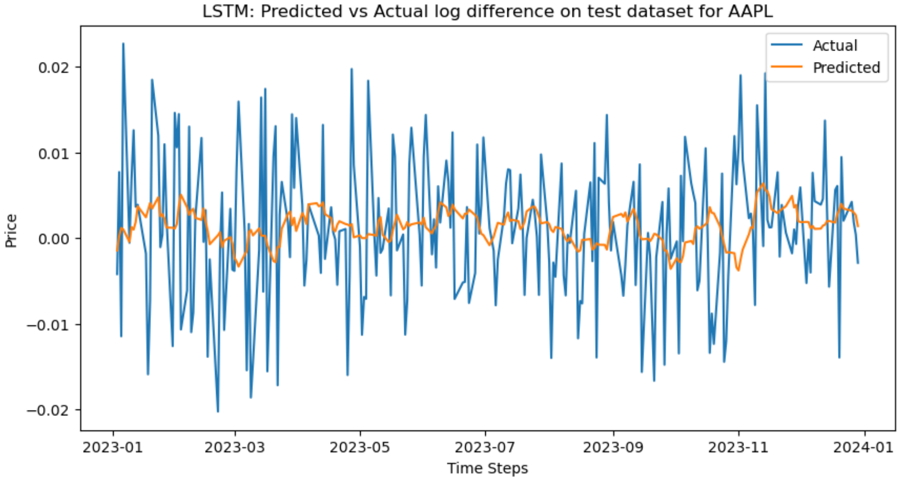
- $\mathbf{g}_t$ analyzes current input $\mathbf{x}_t$ and previous short-term memory state $\mathbf{h}_t$.
- $\mathbf{f}_t$, $\mathbf{i}_t$, $\mathbf{o}_t$ are numbers between 0 and 1.
- $\mathbf{f}_t$ is the *forget gate*, i.e. amount of past memory $\mathbf{c}_{t-1}$ to forget.
- $\mathbf{i}_t$ is the *input gate* i.e. amount of new memory $\mathbf{g}_t$ to write into $\mathbf{c}_{t-1}$.
- $\mathbf{o}_t$ is the *output gate* i.e. amount of output $\tanh(\mathbf{c}_t)$ to $\mathbf{y}_t$ and $\mathbf{h}_t$.
- $\mathbf{W}_{x,i}$, $\mathbf{W}_{x,f}$, $\mathbf{W}_{x,o}$, $\mathbf{W}_{x,g}$ are weight matricies for $\mathbf{x}_t$.
- $\mathbf{W}_{h,i}$, $\mathbf{W}_{h,f}$, $\mathbf{W}_{h,o}$, $\mathbf{W}_{h,g}$ are weight matricies for $\mathbf{h}_{t-1}$
- $\mathbf{b}_i$, $\mathbf{b}_f$, $\mathbf{b}_o$, $\mathbf{b}_g$ are bias terms.

LSTM



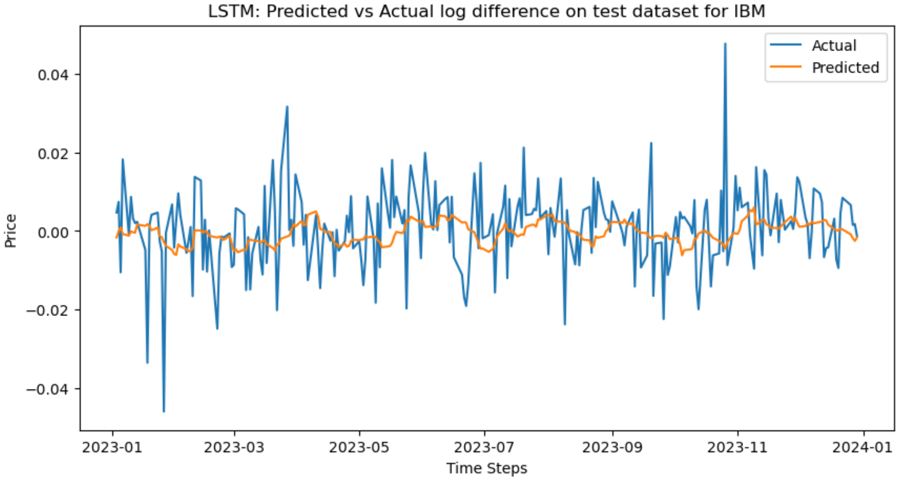
y(t)
Forget gate
c(t-1)
c(t)
Input gate
h(t)
f(t)
g(t)
i(t)
o(t)
Output gate
FC
FC
FC
FC
h(t-1)
LSTM cell
x(t)
⊗ Element-wise multiplication
⊕ Addition
logistic
tanh

LSTM

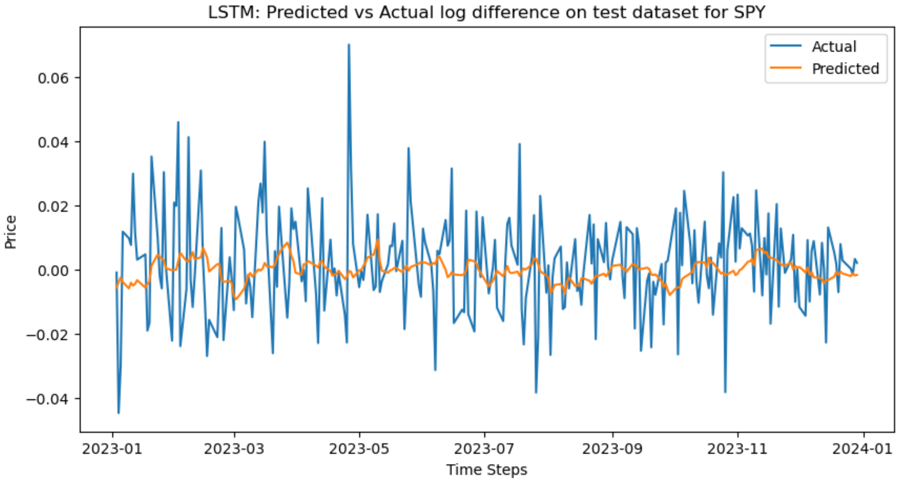


LSTM: Predicted vs Actual log difference on test dataset for AAPL
Actual
Predicted
Price
0.02
0.01
0.00
−0.01
−0.02
2023-01
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
Time Steps

LSTM


LSTM: Predicted vs Actual log difference on test dataset for IBM
Actual
Predicted
Price
0.04
0.02
0.00
−0.02
−0.04
2023-01
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
Time Steps

LSTM


LSTM: Predicted vs Actual log difference on test dataset for SPY
Actual
Predicted
Price
0.06
0.04
0.02
0.00
−0.02
−0.04
2023-01
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
Time Steps

LSTM

Ticker	Train MSE	Valid MSE	Test MSE	Signal
AAPL	0.0049	0.0063	0.0017	0.76
IBM	0.0060	0.0043	0.0018	0.436
SPY	0.0046	0.0062	0.0030	0.428

Table: Mean Squared Error and Signal for each ticker using LSTM.

- **AAPL:** LSTM outperforms Linear Regression and ARIMA with lower Test MSE (0.0017) and higher Signal (0.760), indicating improved predictions.
- **IBM and SPY:** LSTM shows similar Test MSE to Linear Regression but with lower Signal, suggesting marginal improvement.

Signal for LSTM

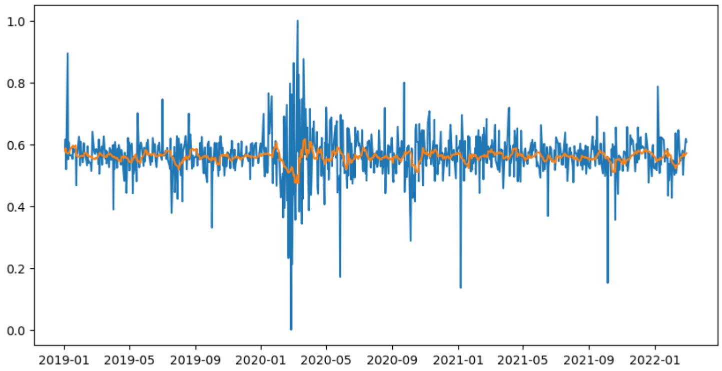


Figure: LSTM Signal Plot from 2019 to 2022

- **COVID-19 Impact:** Notable deviation below the trend in 2020 due to market disruptions caused by the COVID-19 pandemic.

Conclusion

- **LSTM Outperforms Other Models:** LSTM demonstrates better predictive performance compared to Linear Regression, ARIMA, and Simple RNN, especially for AAPL stock.
- **Challenges with Volatility:** Despite its superior performance, LSTM struggles to fully capture the volatility in log price changes, particularly during significant market events like the COVID-19 pandemic in 2020.
- **Evaluation Metric Consideration:** Mean Squared Error (MSE) may not be the most suitable performance measure for this task, as it does not adequately reflect the model's ability to capture market volatility and dynamics.