Predicting The Stock Market Using Machine Learning Techniques.

Ryan Taylor, Christian Okey-Ezeh, and Nasor Clough

Georgia State University

Abstract — In this study, we investigate the application of Support Vector Machines (SVM), Linear Regression, Long Short-Term Memory (LSTM), and Random Forest Regressor for predicting stock prices of five prominent companies traded on the Nasdaq Stock Exchange: Amazon, American Airlines, Apple, Microsoft, and NVIDIA. Utilizing historical stock market data for model training, we employ root mean squared error (RMSE) as the evaluation metric to gauge the performance of each model. Our objective is to identify the most accurate model for stock price prediction and assess the viability of these methods in forecasting market trends. The findings reveal that while Random Forest and LSTM demonstrate superior performance on the training dataset, Linear Regression and SVM often outperform both when faced with unseen data. This study offers valuable insights into the effectiveness of various machine learning techniques in predicting stock prices and their potential applications in financial markets.

I. Introduction

The stock market is a complex and dynamic system that is influenced by a variety of factors, such as economic indicators, company news and world events. Precise stock price predictions can provide valuable insights for investors, traders and financial analysts. Machine learning techniques have been widely used to analyze stock data and to make predictions based on historical trends. In this project, we focus on four prominent machine learning models, namely SVM, Linear Regression, LSTM and Random Forest, which have demonstrated promising results in prior studies. We selected five companies with diverse backgrounds and market performance, to create a robust and comprehensive analysis of our models' performance. Our project aims to provide a comparative study of these models and to determine their strengths and weaknesses. We acquired our data for our training set from the NASDAQ data on kaggle. Overall, our study has important implications for investors and financial analysts who rely on accurate stock price forecasts. The findings can potentially inform investment decisions, risk management strategies, and financial forecasting practices in the ever-changing landscape of financial markets.

II. Dataset, Preprocessing, and Training

While the dataset[1] chosen contains data on stocks from the NASDAQ, NYSE, and S&P500, we chose to hand pick individual stocks from the NASDAQ to train the models on stocks that were similar to each other. Our code loads stock price data from the respective CSV files for each chosen stock, preprocesses it, and generates a windowed dataset. Subsequently, the data is divided into training, validation, and test sets, with the designated model trained on the training set. The model is then evaluated on the validation and test sets, and the predictions are plotted alongside the actual values.

Each CSV file contains extensive data for a specific stock spanning multiple years. To enhance the dataset[1], we performed preprocessing by adding a feature representing the 30-day moving average of closing prices, as well as a label column indicating the closing price 30 days from any given day. The script also computes the root mean squared error (RMSE) between the predicted and actual values on the training set and outputs the result. Our visualizations display the date on the X-axis and the closing price on the Y-axis, providing an intuitive representation of model performance.

III. SVM

Support Vector Machines (SVMs) are a powerful class of supervised learning algorithms used for both classification and regression tasks. The basic idea behind SVMs is to find a hyperplane in a high-dimensional space that can effectively separate different classes or predict the value of a continuous output variable. SVMs aim to maximize the margin between the hyperplane and the closest data points of each class, which are called support vectors.

In other words, SVMs try to find the decision boundary that separates the data into two classes with the largest margin, which is the distance between the hyperplane and the closest data points. SVMs can also handle non-linearly separable data by using kernel functions to transform the input data into a higher-dimensional space, where a linear hyperplane can separate the data.

The training process of SVMs involves finding the optimal hyperplane that maximizes the margin while minimizing the classification error. This is typically done by solving a quadratic optimization problem, which involves minimizing the norm of the weight vector subject to the constraint that all data points are correctly classified.

SVMs have several advantages, such as their ability to handle high-dimensional data and their robustness to overfitting. They are widely used in many applications, such as image classification, text classification, and bioinformatics. However, SVMs can be computationally expensive and may require careful tuning of the hyperparameters, such as the choice of kernel function and the regularization parameter.

For our stock price prediction project, we opted for an SVM with a linear kernel and a regularization parameter (C) of 1. The linear kernel was chosen as it best suits continuous output prediction, and a C value of 1 offers a balance between maximizing the margin and minimizing the classification error.

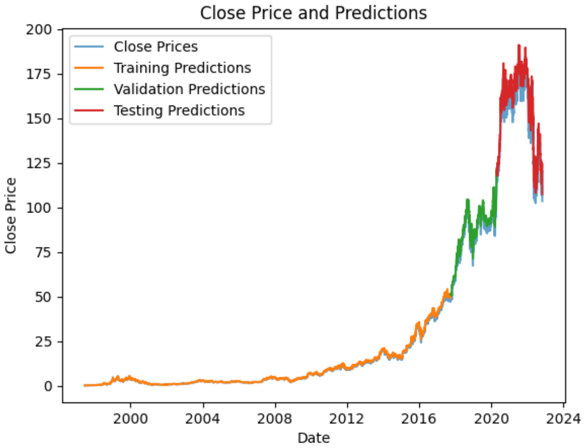


Fig. 1. Graph of SVM model's performance on Amazon Dataset.

IV. Linear regression

Linear Regression is a widely used statistical modeling technique that seeks to establish a linear relationship between a dependent variable and one or more independent variables. In the context of our project, the dependent variable is the label of the closing stock price 30 days ahead, while the independent variables include the historical stock data of the Open, Volume, Low, High, Close, Adjusted Close and the 30-day moving average. The primary goal of linear regression is to fit a line that best represents the relationship between the dependent and independent variables, minimizing the difference between predicted and actual values.

In our study, we employed linear regression to predict stock prices, leveraging its simplicity and interpretability. Linear regression assumes that the relationship between the dependent and independent variables is linear, which may not always hold true for complex systems such as the stock market. Nevertheless, linear regression can serve as a valuable baseline for comparison against more sophisticated machine learning models like SVMs, LSTMs, and Random Forest Regressors.

To fit the linear regression model, we used the least squares method, which minimizes the sum of the squared differences between the actual and predicted values. Once the model is trained, it can be used to make predictions on unseen data, allowing for the evaluation of its performance in predicting stock prices. In our analysis, we compared the performance of the linear regression model with other machine learning techniques to identify the most accurate and reliable method for stock price prediction.

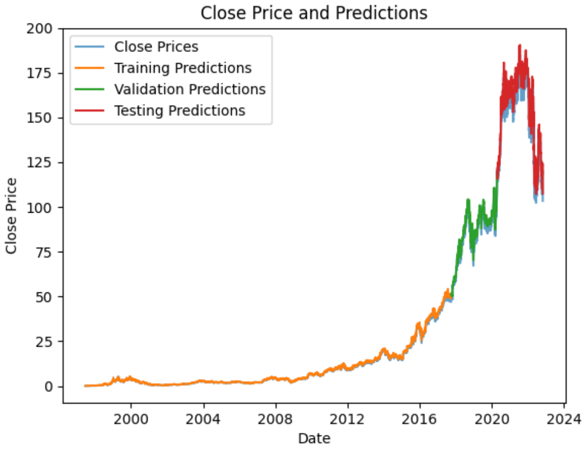


Fig. 2. Graph of Linear Regression model's performance on Amazon Dataset.

V. LSTM

LSTM (Long Short-Term Memory) is a type of Recurrent Neural Network (RNN) that is commonly used for time-series data. It is designed to overcome the limitation of traditional RNNs by avoiding the vanishing gradient problem that occurs when training a neural network over a long period of time. The LSTM model consists of several memory cells, each with three gates: input, output, and forget gates. These gates help control the flow of information within the cell, enabling the model to selectively remember or forget past information.

Our code uses a window of the previous three days' stock prices to predict the stock price for the next day. The code preprocesses the raw stock price data by creating a label indicating what the stock price will be in the next 30 days, and also calculates the 30-day moving average of the stock price.

The code uses several libraries, including NumPy, Pandas, Matplotlib, Scikit-learn, and Keras/TensorFlow. It first loads the stock price data from a CSV file and preprocesses it by dropping any rows with missing values, shifting the "Close" column by 30 days to create the labels, and calculating the 30-day moving average.

The code then converts the preprocessed data into a windowed format suitable for LSTM training, by creating windows of size 4 (3 input days and 1 output day) and converting them into arrays of shape (num_samples, window_size, num_features=1). It splits the windowed data into training, validation, and testing sets, using a 80%/10%/10% split.

The LSTM model architecture in the code consists of a single LSTM layer with 64 units, followed by two dense layers with 32 units and ReLU activation, and a final dense layer with 1 unit. The model is compiled with mean squared error loss and Adam optimizer with a learning rate of 0.001.

The code trains the model for 100 epochs on the training set, and evaluates the model's performance on the validation and test sets using mean squared error and mean absolute error metrics. Finally, it plots the predicted and actual stock prices for the training, validation, and test sets using Matplotlib.

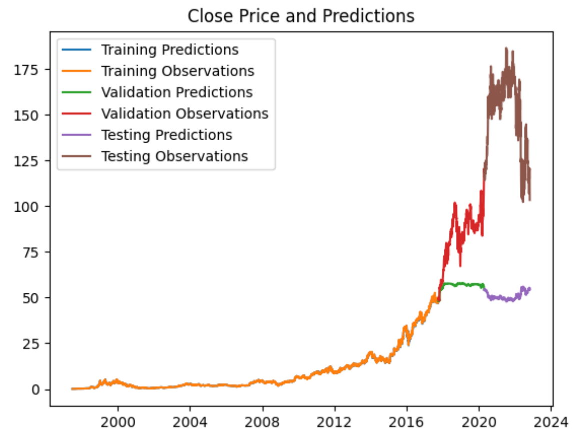


Fig. 3. Graph of LSTM model's performance on Amazon Dataset.

VI. Random Forest Regressor

Random Forest is an ensemble learning technique that constructs multiple decision trees and combines their predictions to achieve improved generalization and prediction accuracy. In the case of regression tasks, the Random Forest Regressor averages the output of multiple decision trees to predict a continuous value, such as stock prices. This technique reduces the risk of overfitting and provides a more robust model compared to single decision trees.

In our project, we employed a Random Forest Regressor to predict stock prices based on historical data. The model takes into account several parameters. We set the n_estimators parameter, which represents the number of decision trees in the forest, to 200 trees. This choice was made to create a diverse set of predictions and achieve better generalization. A higher number of trees usually results in improved performance, but it also increases the computation time. Additionally, we chose a max_depth of 15, which defines the maximum depth of each tree in the forest. This value was selected to strike a balance between model complexity and the risk of overfitting. A lower max_depth value may lead to underfitting, while a higher value might result in overfitting the model to the training data. By using these parameters, our Random Forest model aimed to provide accurate and reliable predictions of stock prices.

To evaluate the performance of the Random Forest model, we used the Root Mean Squared Error (RMSE) as our evaluation metric. RMSE is a popular metric for regression problems, as it measures the average squared difference between the predicted and actual values, penalizing larger errors more heavily than smaller ones. By comparing the RMSE values of different models, we aimed to identify the model that provided the most accurate predictions and determine the viability of these machine learning techniques for predicting stock prices.

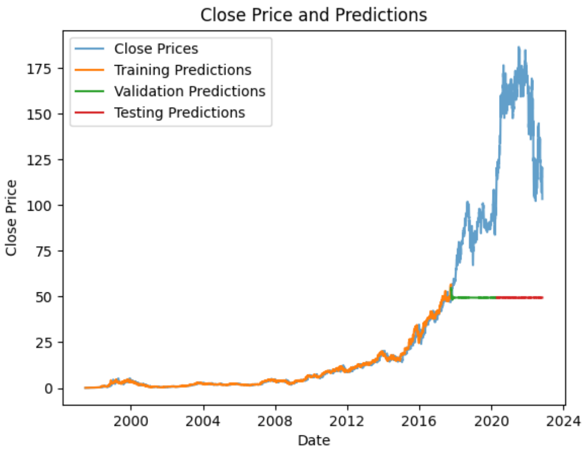


Fig. 4. Graph of SVM model's performance on Amazon Dataset.

VII. Results

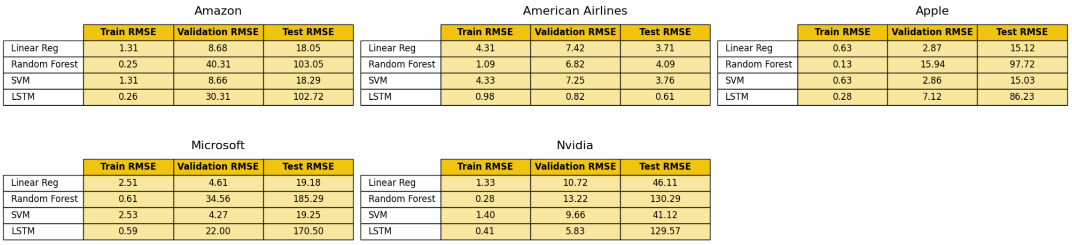
Our project evaluated the performance of four machine learning models, namely LSTM, Random Forest, Linear Regression, and SVM, for stock price prediction. We assessed the performance of these models using the root mean squared error (RMSE) metric on training, validation, and test data for five stocks. Those five companies, all traded on the Nasdaq Stock Exchange were Amazon, American Airlines, Apple, Microsoft, and NVIDIA.

The results indicate that LSTM and Random Forest models perform exceptionally well on the training data, with both models achieving an RMSE below 1 for all five stocks, except for American Airlines, where the Random Forest model still achieves a RMSE of 1.09. This demonstrates that these models are able to fit the training data effectively.

However, when evaluating the performance on the validation and test data, Linear Regression and SVM models seem to outperform LSTM and Random Forest in all tested stocks, except for American Airlines. In this specific case, the LSTM model demonstrates superior performance, with a test RMSE of 0.61, while Linear Regression, Random Forest, and SVM models have test RMSE values of 3.71, 4.09, and 3.76, respectively.

Overall, on the validation and test data, Linear Regression and SVM models perform, on average, approximately 5.775 times better than Random Forest and LSTM models, excluding the American Airlines case. In contrast, when considering the American Airlines case, Random Forest and LSTM models perform about 6.1 times better than Linear Regression and SVM models. On the training data, Random Forest and LSTM models perform, on average, around 3.880 times better than Linear Regression and SVM models across all five stocks.

These results provide valuable insights into the performance and reliability of the four machine learning models for stock price prediction. The differences in performance between training and validation/test data highlight the importance of evaluating models on unseen data to assess their generalization capabilities. Moreover, the varying performance across different stocks suggests that the choice of the best model may depend on the specific stock being analyzed, emphasizing the importance of a thorough comparative analysis for each stock of interest.



Amazon

	Train RMSE	Validation RMSE	Test RMSE
Linear Reg	1.31	8.68	18.05
Random Forest	0.25	40.31	103.05
SVM	1.31	8.66	18.29
LSTM	0.26	30.31	102.72

American Airlines

	Train RMSE	Validation RMSE	Test RMSE
Linear Reg	4.31	7.42	3.71
Random Forest	1.09	6.82	4.09
SVM	4.33	7.25	3.76
LSTM	0.98	0.82	0.61

Apple

	Train RMSE	Validation RMSE	Test RMSE
Linear Reg	0.63	2.87	15.12
Random Forest	0.13	15.94	97.72
SVM	0.63	2.86	15.03
LSTM	0.28	7.12	86.23

Microsoft

	Train RMSE	Validation RMSE	Test RMSE
Linear Reg	2.51	4.61	19.18
Random Forest	0.61	34.56	185.29
SVM	2.53	4.27	19.25
LSTM	0.59	22.00	170.50

Nvidia

	Train RMSE	Validation RMSE	Test RMSE
Linear Reg	1.33	10.72	46.11
Random Forest	0.28	13.22	130.29
SVM	1.40	9.66	41.12
LSTM	0.41	5.83	129.57

Fig. 5. Table of train, validation, and test performance of all models on all stocks.

VIII. Conclusion

In conclusion, our comparative study of four machine learning models - SVM, Linear Regression, LSTM, and Random Forest - for stock price prediction revealed interesting insights. Linear Regression and SVM, particularly with a linear kernel, demonstrated better performance on unseen validation and test data compared to LSTM and Random Forest. The simpler nature of Linear Regression and SVM with a linear kernel likely contributed to their ability to generalize better on unseen data, as they are less prone to overfitting than more complex models like LSTM and Random Forest. This generalization capability might be particularly useful in stock price prediction, where future trends can often differ from historical patterns. However, it is essential to consider that different hyperparameter choices, particularly for SVM, can significantly impact performance and generalization. The performance of these models varies depending on the stock being analyzed and the type of data (training, validation, or test) being used for evaluation. Factors such as model complexity, underlying assumptions, and the choice of hyperparameters also could have contributed to our findings. Our findings highlight the importance of considering the specific stock being analyzed and the evaluation of models on unseen data. Further research may explore the use of more advanced techniques, such as ensemble learning or deep learning models, as well as the impact of various hyperparameter configurations, feature engineering techniques, and the incorporation of additional features, such as news sentiment or technical indicators to improve prediction accuracy.

References and data

[1] P. Mooney, “Stock Market Data (NASDAQ, NYSE, S&P500),” *Kaggle*, 16-Jan-2023. [Online]. Available: https://www.kaggle.com/datasets/paultimothymooney/stock-market-data?resource=download. [Accessed: 02-May-2023].