Stock Market Prediction Using Machine Learning

Introduction/Background

There are two main ways for investors to analyze a stock. The first way is a fundamental analysis which takes into account “the intrinsic value of stocks, and performance of the industry, economy, political climate etc” [1]. The other way is through a technical analysis by viewing the market activity, including the previous prices and volumes. Due to the volatility of the stock market, it is difficult to make accurate predictions [2]. While it may be difficult, there is plenty of research attempting to predict price changes using a variety of combinations of preprocessing and ML methods [2]. The most commonly used methods fall into neural networks, support vector machines or genetic algorithms [3].

Dataset

We will use the following dataset from Kaggle: Stock Market Dataset. It includes the date, opening price, closing price, maximum price, minimum price, adjusted close price, and volume for each stock up until 04/01/2020. These features will help analyze price changes for different stocks.

Problem Statement

The stock market is inherently volatile and non-linear, making stock price prediction a difficult task. The goal of this project is to use machine learning techniques to increase the accuracy of stock price prediction. This project is motivated by the potential benefits for investors and traders, allowing them to make more informed decisions.

Methods

Preprocessing

- **Z-score Standardization**: To account for varying stock prices across different stocks as well as remove outliers, the data for the stock prices will be normalized using Z-score standardization. The

StandardScaler class in scikit-learn will be used.

- **Feature Engineering**: New features such as moving averages and rolling statistics will be created to enhance model performance. Moving averages will help smooth out price data to identify trends more easily while rolling statistics will capture the short-term variability in stock prices.
- **Train-Test Split**: To evaluate the model's effectiveness, the dataset will be split into training and testing sets. We will use a time-based split, where earlier data is used for training and more recent data is reserved for testing. This will help mimic real-world scenarios where future prices are predicted based on past data.

ML Algorithms/Models

- **Random Forest**: Random forest will be used as it is resistant to the noise and volatility of the market. Since it averages out the results of multiple decision trees made with random data points, it also reduces potential overfitting. The RandomForestRegressor class in sci-kit learn will be used.
- **Long Short-Term Memory (LSTM)**: Stock market data is often noisy and filled with random fluctuations. LSTM's architecture allows it to ignore irrelevant short-term spikes (through the forget gate) while focusing on long-term patterns that are more indicative of future behavior.
- **Support Vector Machine (SVM)**: SVM is well-suited for stock market prediction due to its ability to model complex, nonlinear relationships between input features by projecting them into higher-dimensional spaces. This helps in capturing subtle trends and dependencies in the data that simpler models might miss. The SVR class in scikit-learn will be used.

Evaluation Metrics

We will use the following quantitative metrics to evaluate model performance:

- **Mean Squared Error (MSE)**: Measures the average squared difference between the predicted and actual stock prices.
- **Root Mean Squared Error (RMSE)**: Provides error in the same units as the target variable.
- **Mean Absolute Percentage Error (MAPE)**: Expresses accuracy as a percentage, making it easier to interpret.

Goals and Expectations

We aim to develop a model that can accurately predict future stock movements by not overfitting our model to the training data. To make our model resilient, it will need to be trained on markets with varying

levels of implied volatility and directions. By only using publicly available stock information, we are in adherence to ethical trading standards and regulations [4].

References

- [1] K. Vanukuru, “Stock Market Prediction Using Machine Learning,” International Research Journal of Engineering and Technology, vol. 5, no. 10, pp. 1032-35, 2018. [Online]. Available: https://doi.org/10.13140/RG.2.2.12300.77448.
- [2] A. Gupta, Akansha, K. Joshi, M. Patel and V. Pratap, "Stock Market Prediction using Machine Learning Techniques: A Systematic Review," 2023 International Conference on Power, Instrumentation, Control and Computing (PICC), Thrissur, India, 2023, pp. 1-6, doi: 10.1109/PICC57976.2023.10142862.
- [3] T. Strader, J. Rozycki, T. ROOT, and Y.-H. (John) Huang, “Machine Learning Stock Market Prediction Studies: Review and Research Directions,” Journal of International Technology and Information Management, vol. 28, no. 4, pp. 63–83, Jan. 2020, Available: https://scholarworks.lib.csusb.edu/jitim/vol28/iss4/3/
- [4] G. Lawson, “The Ethics of Insider Trading,” Harvard Journal of Law and Public Policy, no. 3, p. 727, Jul. 1988, Available: https://scholarship.law.bu.edu/faculty_scholarship/2435/

Contributions

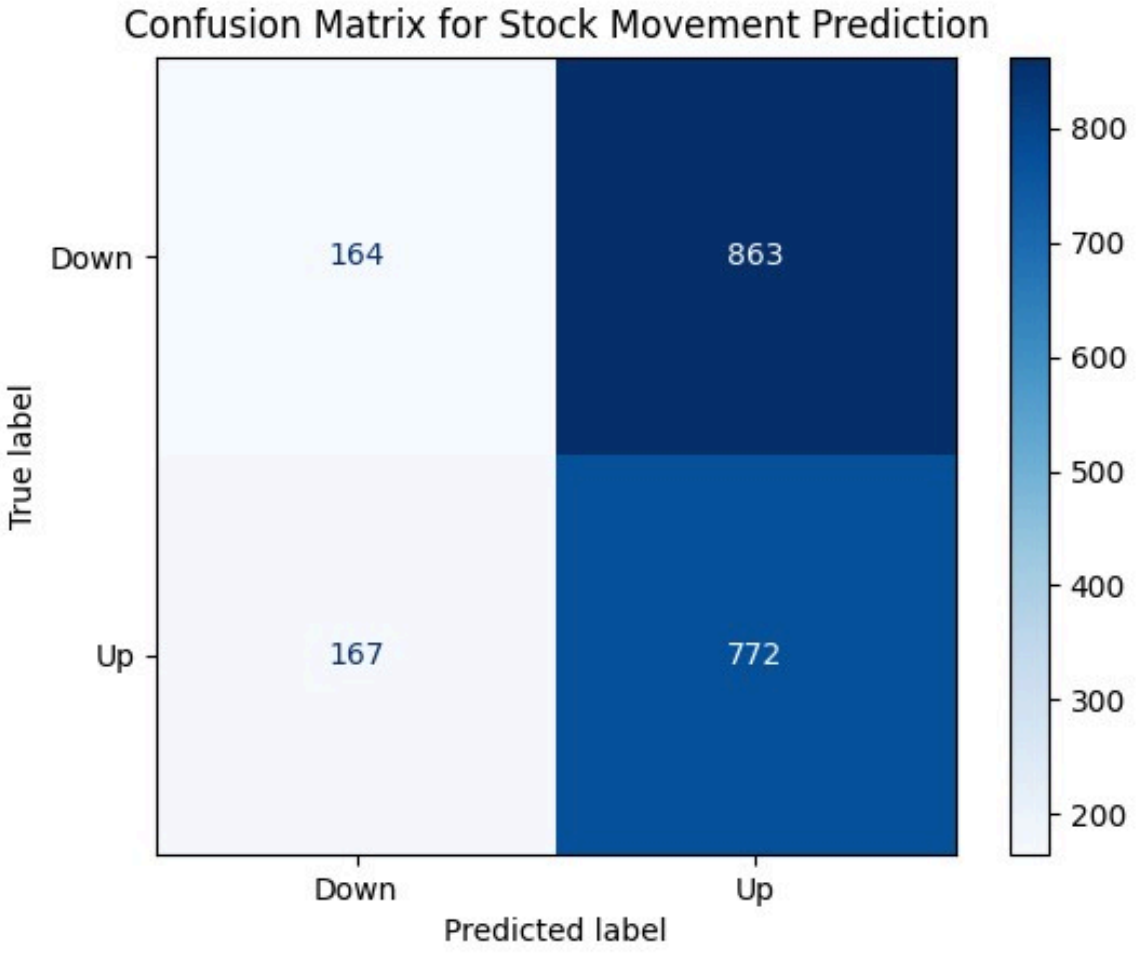
- **Jake Wang**: Created introduction/background, 1 preprocessing method, 1 ML model.
- **Yashman Singh**: 2 ML models, 2 preprocessing methods, 1 quantitative metric, streamlit project.
- **Manya Jain**: Collecting sources for the literature review, technical content and visuals for the powerpoint presentation, recording and posting the video, gantt chart.
- **Jonathan Marto**: Collect 2 additional sources and contribute to literature review. Added project goals, sustainability and ethical considerations.
- **Swapnil Mittal**: Recording and posting the video, quantitative metrics

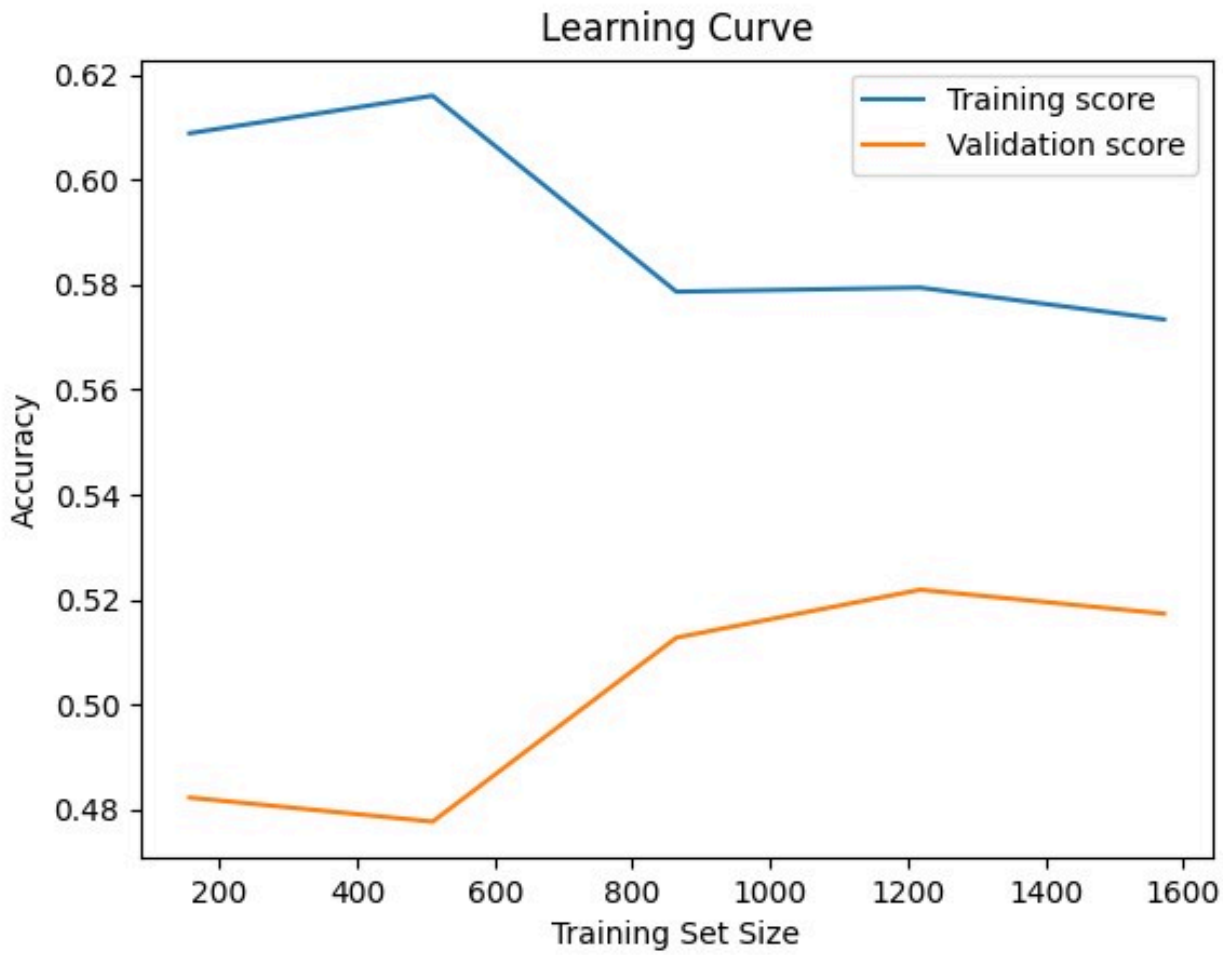
Gantt Chart

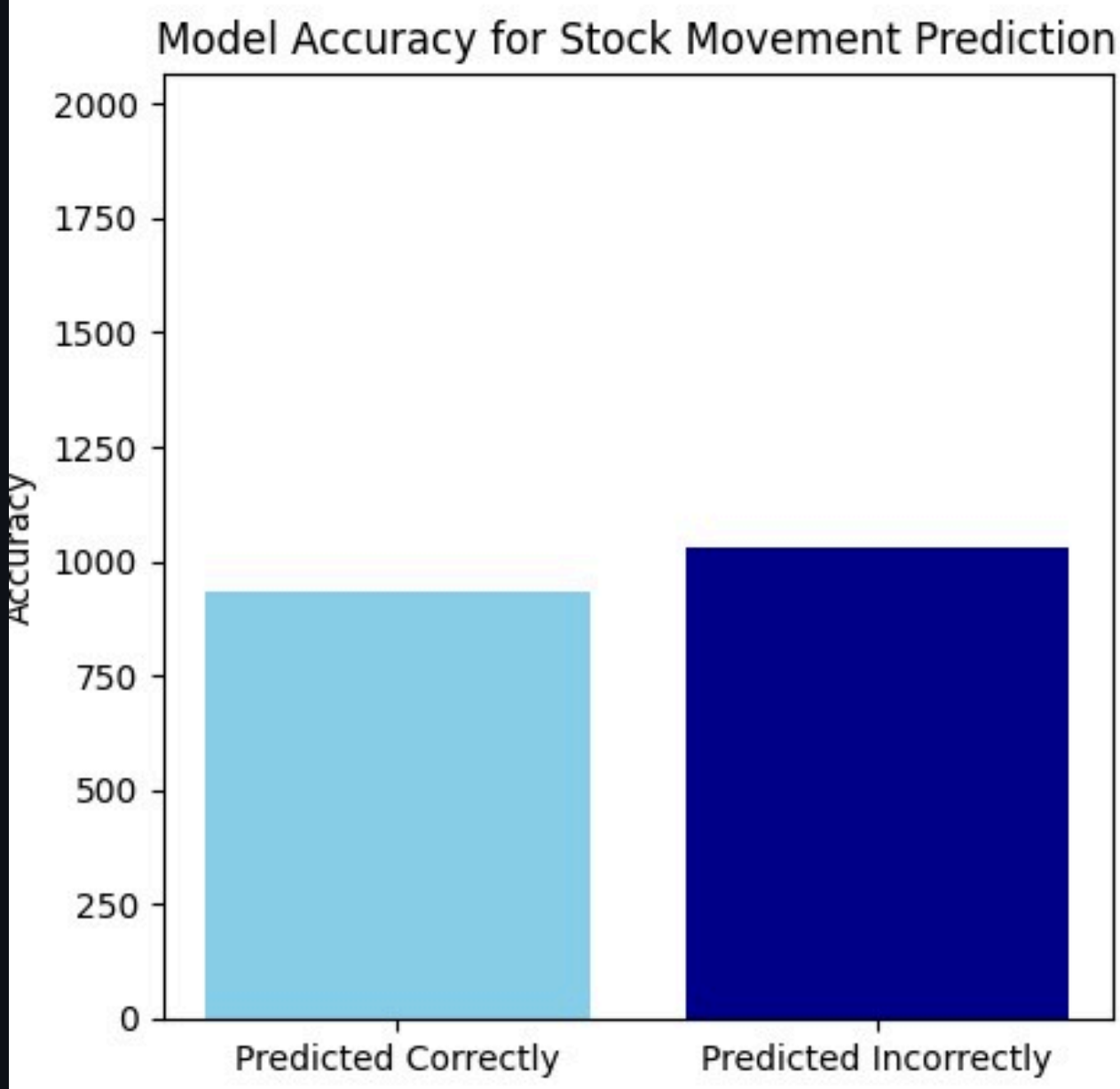
Results and Discussion

Basic SVM

Visualizations




Learning Curve
Training score
Validation score
Accuracy
0.62
0.60
0.58
0.56
0.54
0.52
0.50
0.48
200
400
600
800
1000
1200
1400
1600
Training Set Size



Quantitative Metrics

Confusion Matrix:

- Our initial model struggles to accurately distinguish between "Up" and "Down" movements. The confusion matrix indicates a high number of misclassifications, with a noticeable bias toward predicting "Up" movements. This may be due to an inherent class imbalance, limitations in the feature set, or the model's inability to capture the complex, non-linear relationships in stock movement data.

- This bias toward "Up" predictions suggests that the model may not be receiving enough information to reliably differentiate market directions, highlighting the need for more sophisticated feature engineering.

Learning Curve:

- The gap between training and validation accuracy shows signs of overfitting. As training set size increases, validation accuracy improves slightly but remains below training accuracy, implying the model might not generalize well to unseen data.
- Given this overfitting trend, our current feature set may not be robust enough to provide consistent predictive signals, which suggests that exploring additional or more meaningful features could help reduce this gap.

Analysis of Basic SVM

- The algorithm provides a foundational approach to stock price movement prediction using machine learning. By starting with basic features and progressively incorporating advanced technical indicators and sequential data, the code demonstrates the iterative process of model improvement.
- Data Quality and Feature Engineering**: Stock prediction is inherently challenging due to market volatility. The model's low performance might stem from insufficient or irrelevant features. Including technical indicators or external financial data could improve predictive capability.
- Hyperparameter Tuning: The model might benefit from fine-tuning hyperparameters to find an optimal configuration for better accuracy and generalization.
- Feature Engineering: Our initial approach used basic features (e.g., Close, Volume, 30-day Moving Average) along with some engineered indicators. While these provide a foundation, the current results suggest that they may not fully capture the complexities of stock price movements. We'll explore adding:
    - Advanced technical indicators, such as the Stochastic Oscillator or Bollinger Bands, which are commonly used to identify price momentum and reversal points.
    - External market data, like overall market indices, to provide a broader context.
    - Event-driven or sentiment data to make the model more sensitive to news and events that impact stock prices.

Next Steps

The next steps we plan to take include conducting another data preprocessing method. The next method we plan to implement will be Z-score standardization which may help the model be applied more evenly

to all different stocks. We will also continue with implementing our ML models which includes random forest and long short term memory. With each model, we plan to use our evaluation metrics which include mean squared error, root mean squared error, and mean absolute percentage error which will help us determine the effectiveness of each of our models for comparison. Currently, our SVM model is not very accurate and through the addition of more data preprocessing, we hope to improve its accuracy.

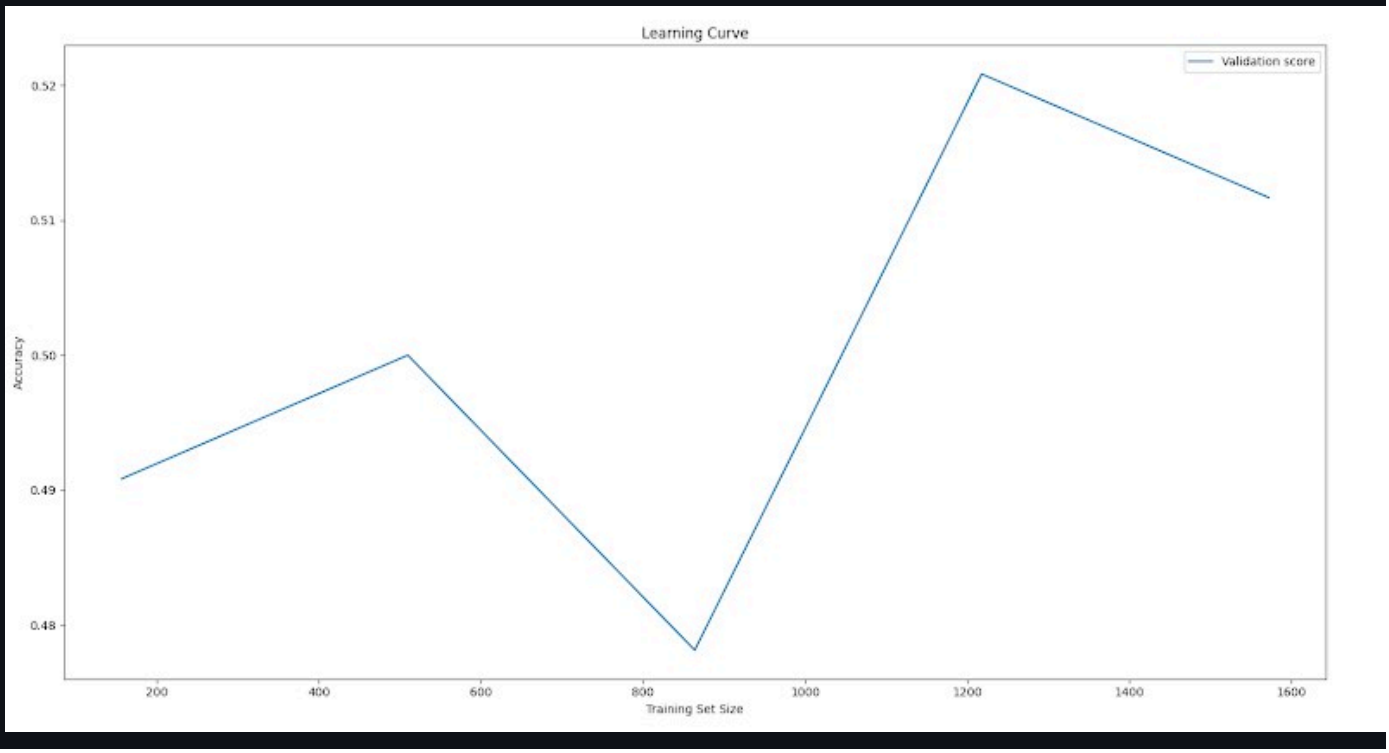
Midterm Contributions

- **Jake Wang**: Created 1 visualization, split training/testing data for pre-processing, next steps
- **Yashman Singh**: Implemented models, maintained GitHub, streamlit website
- **Manya Jain**: Helped implement models, analysis of models
- **Jonathan Marto**: Created 2 visualizations and performed feature engineering in preprocessing / preparing test and train folders
- **Swapnil Mittal**: Updated models for visualization, added quantitative metrics and analysis of algorithm

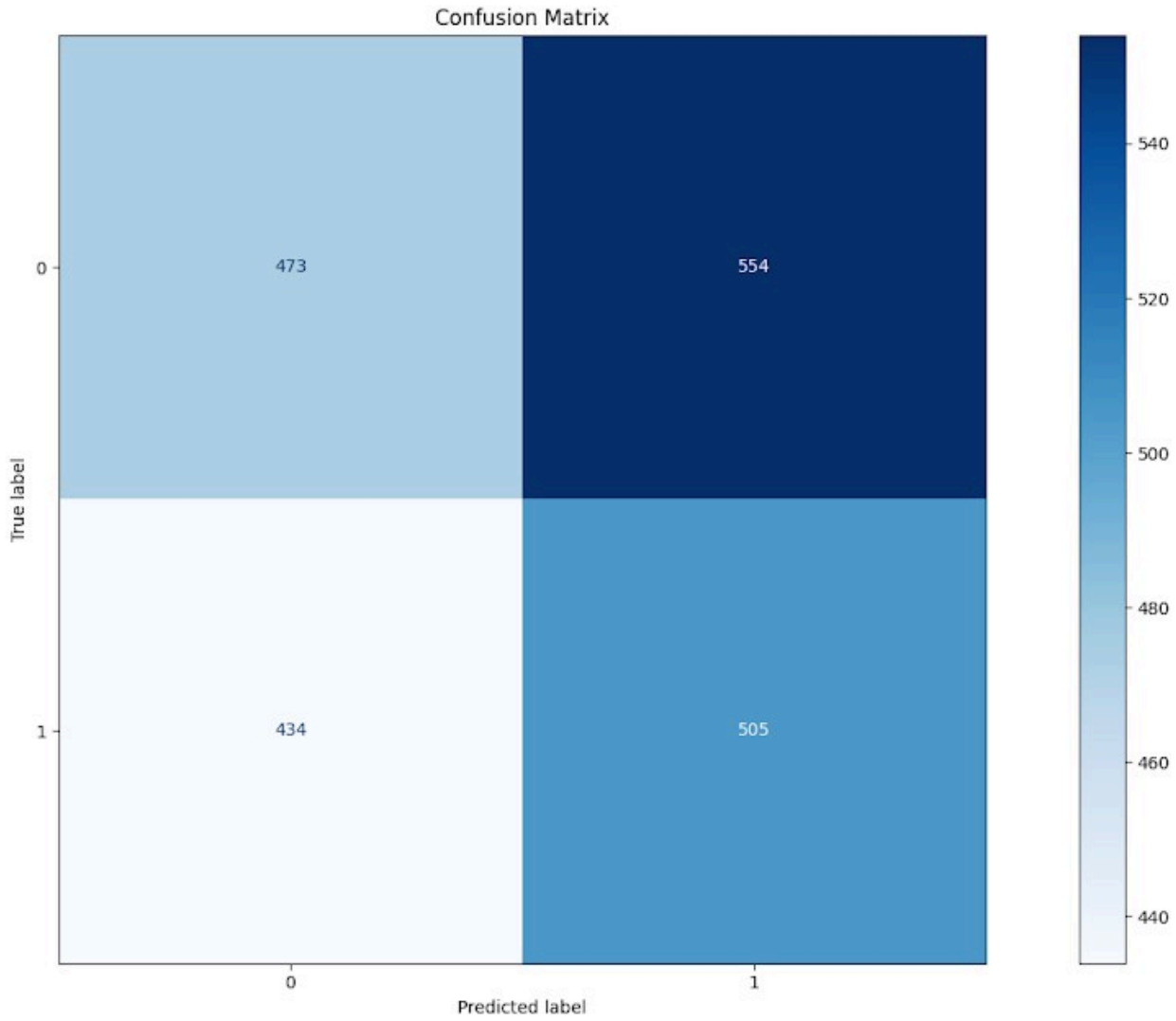
Final Report

Random Forest Classifier

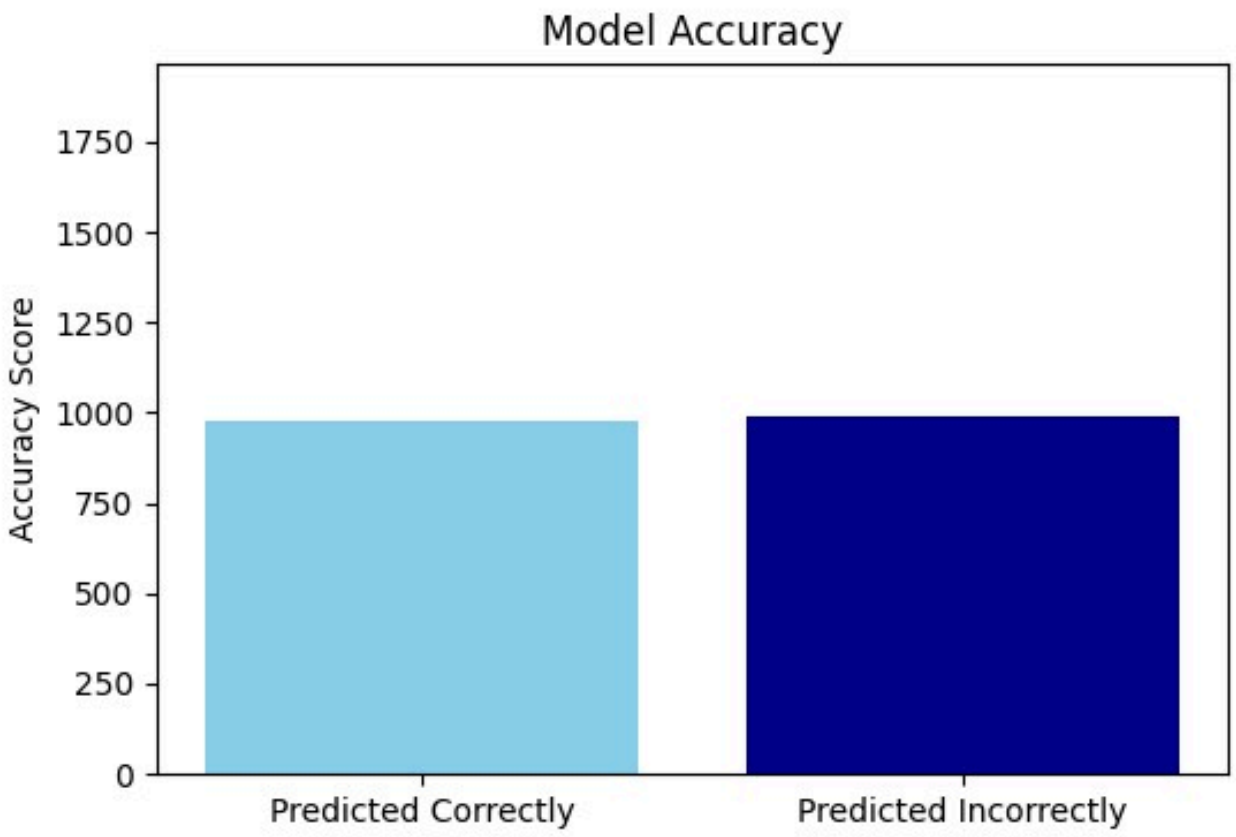
Learning Curve:



Confusion Matrix:

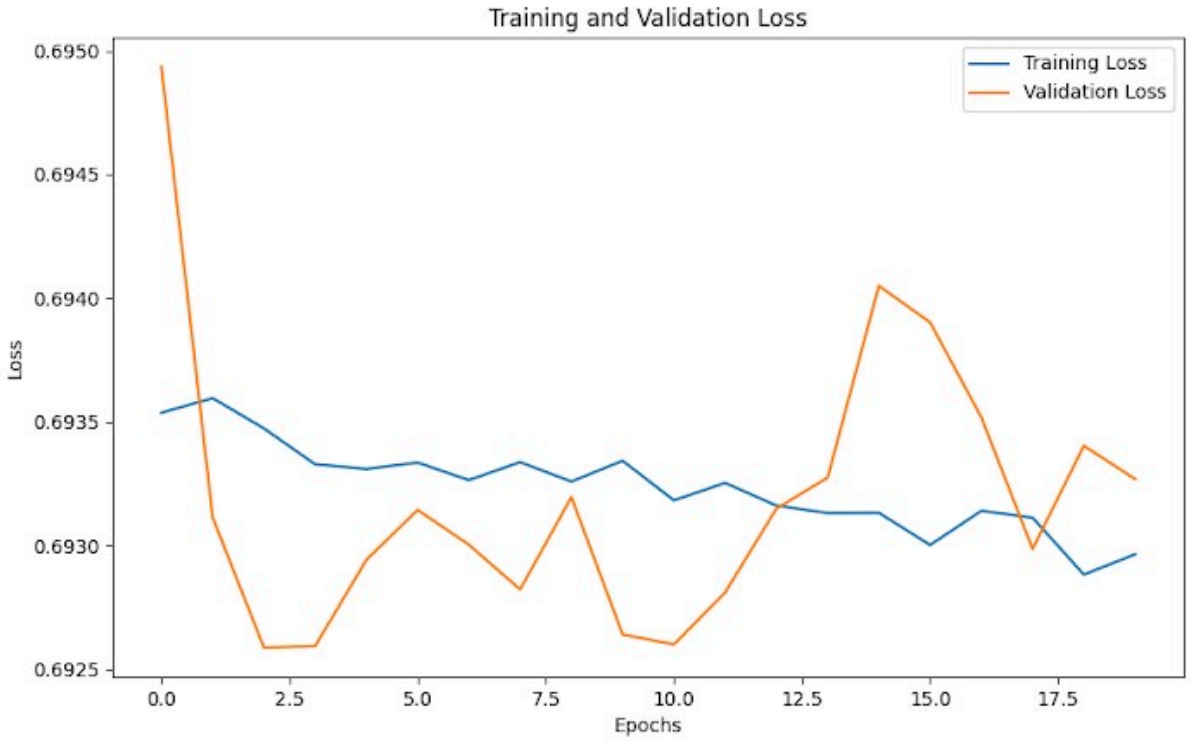


Model Accuracy:

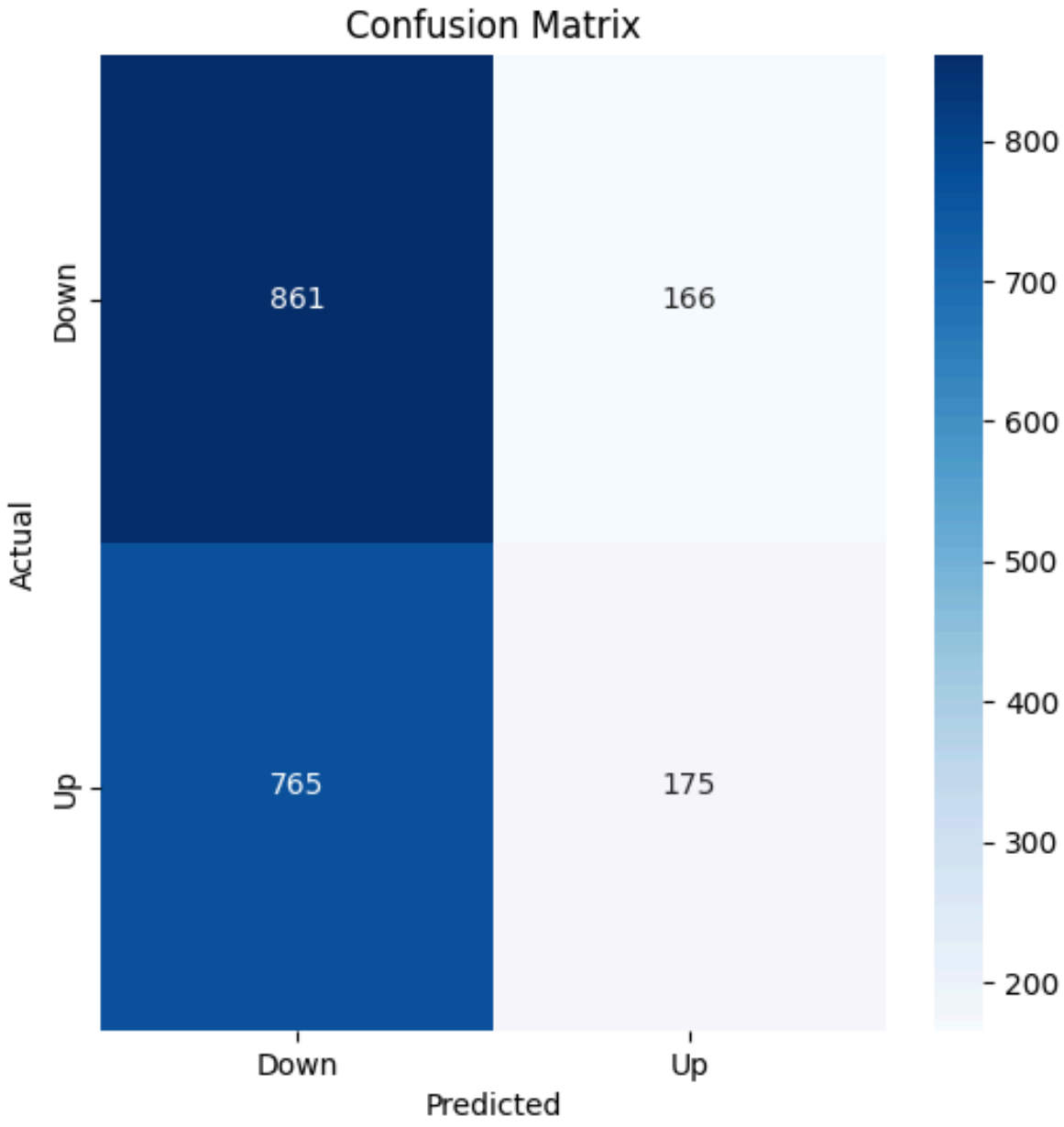


LSTM

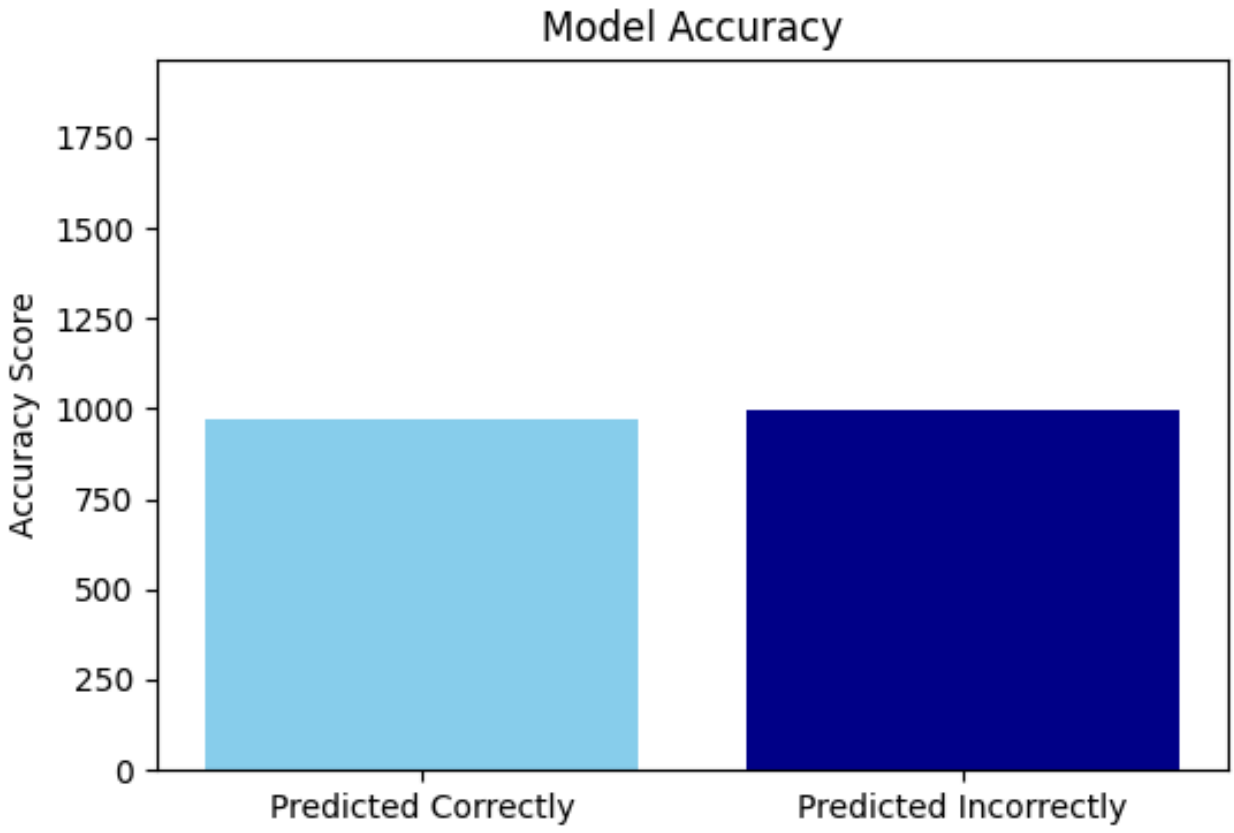
Training and Validation Loss:



Confusion Matrix:

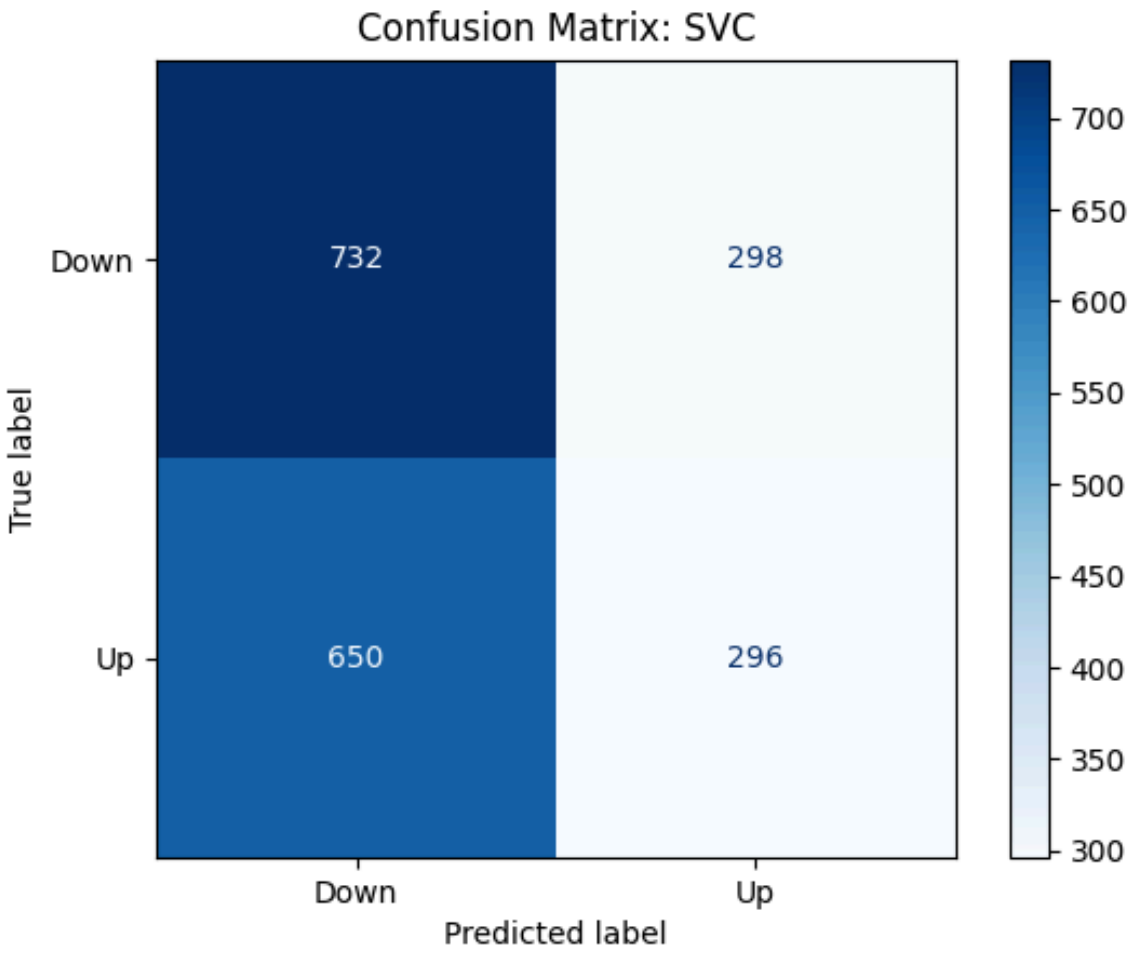


Model Accuracy:

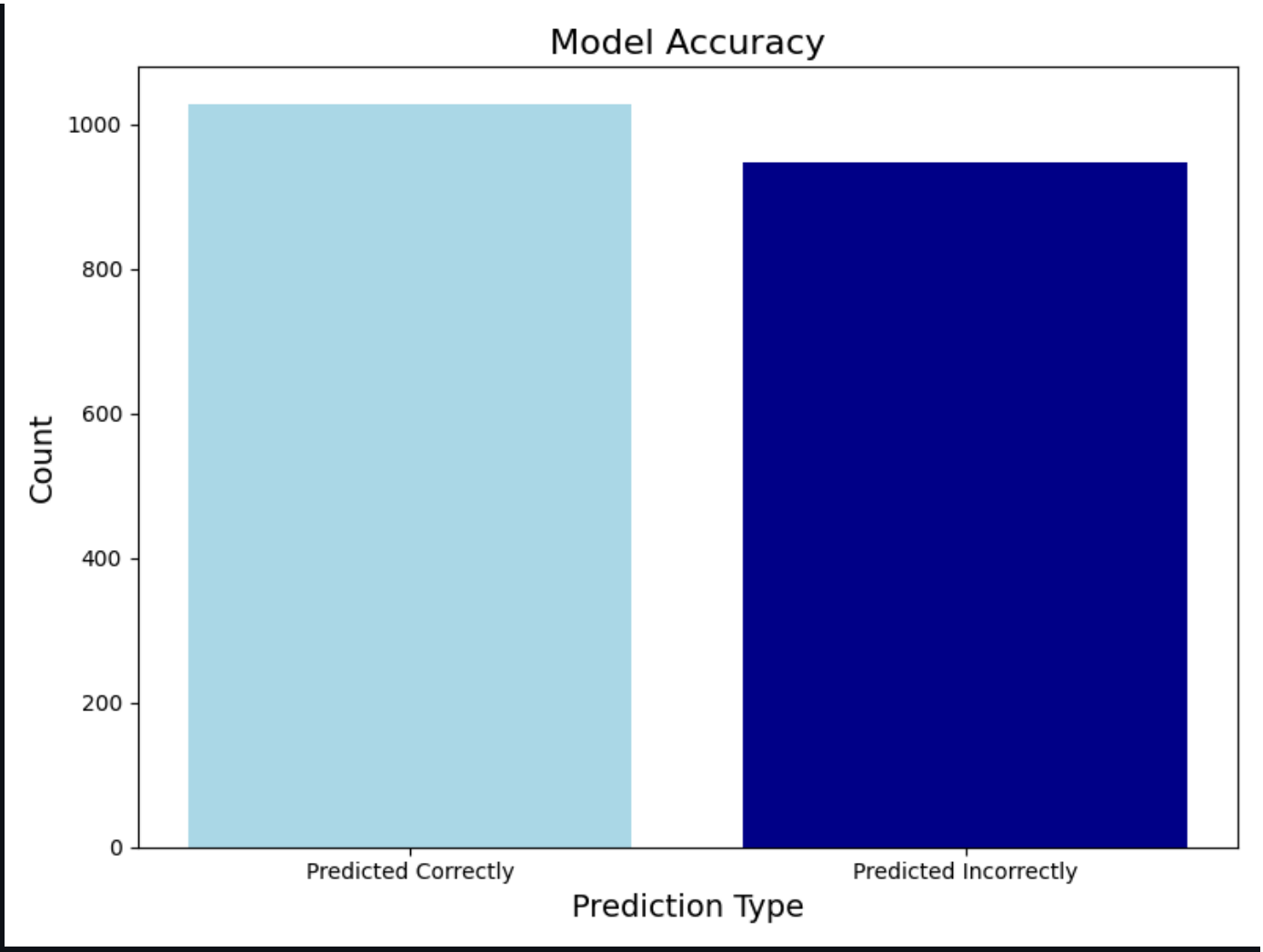


Redone SVC

Confusion Matrix:



Model Accuracy:



Random Forest Quantitative Metrics:

- The model generally will predict that the stock price will move upwards over downwards. From the confusion matrix, it was more accurate in predicting upwards movement when the stock moved upwards over downwards movement when the stock moved downwards. Even when the stock moved downwards, the model predicted upwards movement more often.
- From the validation score graph, the score begins to increase at just over 800 samples and peaks at around 1200 samples. Overall, the graph shows a large amount of movement in validation score, meaning that the model is unstable and unable to make proper generalizations to the data.

Random Forest Analysis:

- The overall accuracy of the model is 50% which is the same as randomly guessing without any information. Some potential reasons as to why Random Forest was not effective here was that it does not account for any time related complexities and focuses only on the specific moment. There is likely lots of noise in the data as well making it difficult to detect patterns with Random Forest.
- Initially, a higher number of decision trees were used. However, using a higher number caused overfitting and drastically decreased the accuracy of the model. By reducing the number of

estimators, the model's accuracy was able to improve and help reduce the impact of overfitting in its predictions.

Random Forest Comparison With SVC:

- While both models have similar accuracies in predicting stock price movement, though Random Forest is lower, the differences in their effectiveness is highlighted in the confusion matrix. With Random Forest, there is closer to a balance between correct upwards and downwards movement. However, with SVC, it has a very skewed accuracy in predicting upwards movement and has very poor accuracy in predicting downwards movement.

Random Forest Comparison with LSTM:

- Both models again have similar accuracies in predicting stock price movement with LSTM having a higher accuracy. However, there is another large difference as the majority of the correctness from LSTM comes from predicting downwards movement when the stock moves downwards.

LSTM Comparison with SVC:

- Both models have a similar accuracy with SVC having a slightly higher accuracy. However, both models are opposite in which aspect they are correct in. LSTM often predicts downwards movement and is often correct when the stock moves downwards. However, SVC often predicts upwards movement and is often correct for upwards movement.

Next Steps:

- For the next steps, the data would be further processed in order to best reduce the amount of noise and provide a better input for each model. Tests would need to be done to help determine the optimal number of parameters to improve accuracy while still avoiding potential overfitting. Implementing other models can also be considered as there may be other models that are able to better predict stock price movement with the current data. New features could also be included into the data which may have an effect on the models' ability to make predictions. For example, there may be outside information, such as public sentiment on the stock, which may also be important in improving the accuracy of each model.

Analysis

1. Random Forest

- Accuracy: 48%

- Precision, Recall, and F1-scores are consistent at approximately 48%.
- Model performance is balanced but indicates failure to capture significant patterns in the data.
- Handles non-linear relationships well.
- Might not leverage sequential time-series relationships effectively.

2. LSTM

- Accuracy: 50%
- Slightly better F1-score (Up: 50%, Down: 51%) compared to Random Forest.
- Accuracy is better than random forest.
- Captures sequential dependencies due to its architecture.
- Fine-tuning the model will increase accuracy.
- *Weaknesses*:
- Training is computationally expensive.
- Results indicate potential overfitting or lack of sufficient data preprocessing.

3. Support Vector Classifier (SVC)

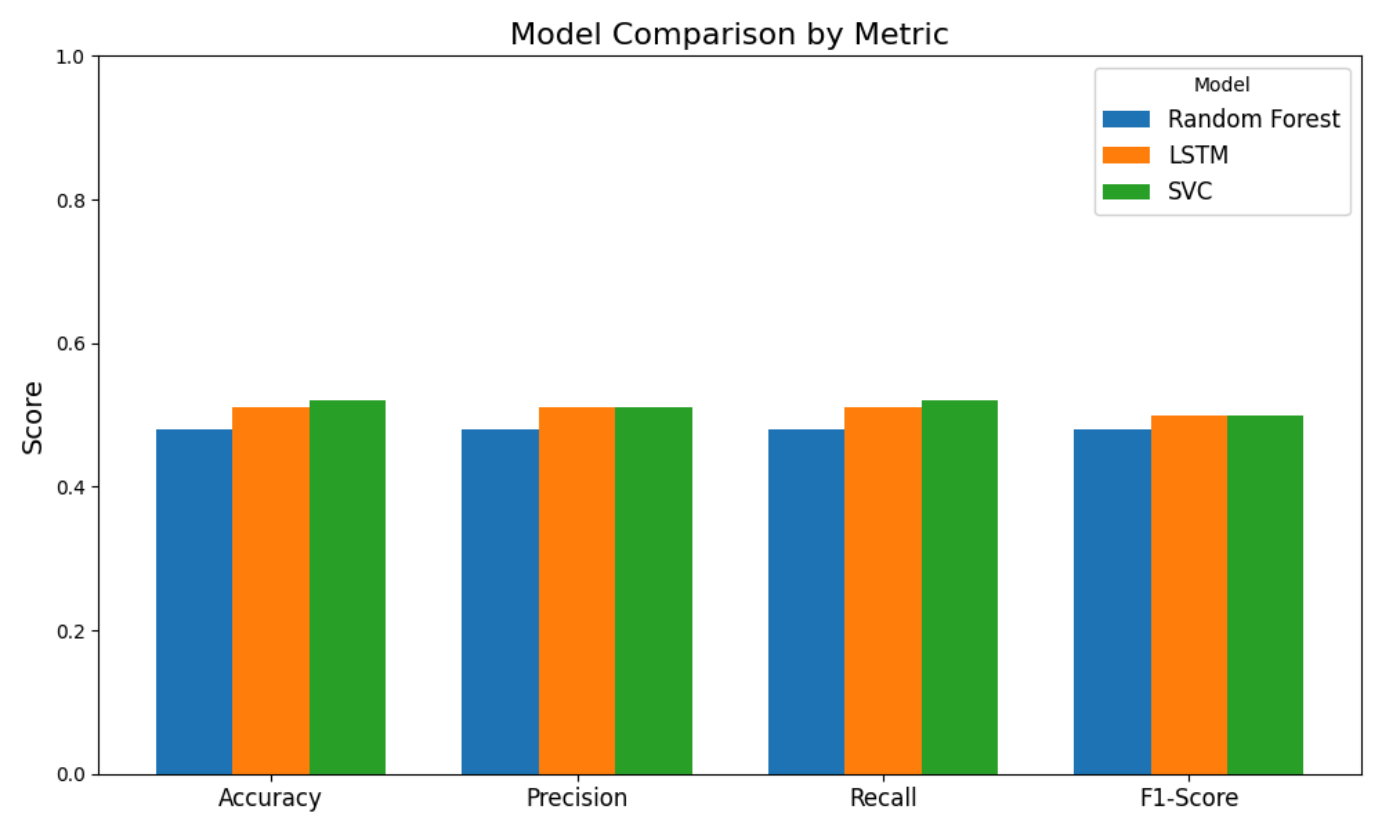
- Accuracy: 52% Classification Report:
- Best performance among the three models.
- High precision (53%) for "Down" predictions but lower recall (31%) for "Up."
- Performs pretty well with small datasets.
- Suitable for binary classification.
- Kernel-based methods can struggle with high-dimensional or noisy data.
- Imbalanced performance between the two classes.

Comparison

The graph highlights performance across four metrics (Accuracy, Precision, Recall, F1-Score):

- SVC performs slightly better than the other models in all metrics.

- LSTM captures some sequential dependencies but fails to surpass SVC in general performance.
- Random Forest demonstrates weaker results, due to the lack of temporal modeling for stock data.



Final Contributions

- **Jake Wang**: Wrote quantitative metrics for Random Forest, model analysis for Random Forest, comparisons between all 3 models, and next steps
- **Yashman Singh**: Maintained Github and Streamlit website, added code for visual analysis, visualizations, metrics, LSTMs and SVC. Also completed model comparison.
- **Manya Jain**: Created video, helped with analysis, and implemented LSTM model.
- **Jonathan Marto**: Implemented Random Forest model, created confusion matrix, prediction accuracy and learning curve visualizations for RF and predicted correctly vs incorrectly for LSTM
- **Swapnil Mittal**: Updated models for visualization, added quantitative metrics and analysis of models, comparisons between different models, and predicted correctly vs incorrectly for LSTM