**WEEKLY PROJECT STATUS REPORT**

|  |  |  |  |
| --- | --- | --- | --- |
| PROJECT NAME | **Prediction model for AMZN stock using LSTM** | Course CODE | **BDM 2053** |

|  |  |  |  |
| --- | --- | --- | --- |
| PROJECT  MEMBERS | DATE OF  STATUS ENTRY | PERIOD  COVERED | PROJECTED DATE  OF COMPLETION |
| Pranesh Balaji Vnekatachalamurthy  Sunny D Souza  Jay Savjbhai Davra  Keyvan Amini  Nitin Ullas | 04/06/2023 | 3 weeks | 04/13/2023 |

PROJECT STATUS THIS WEEK

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OVERALL PROJECT STATUS | ON TRACK | Zone |  |  | ON TRACK |

SUMMARY

|  |
| --- |
| We loaded and preprocessed the data, produced input sequences, specified and assembled the LSTM model, trained and assessed the model on the test data, and predicted the closing price for the following day. All of these tasks were completed successfully. As a result of the effective data preprocessing and training procedure, we estimate that around 80% of the project is completed.  The remaining 20% of the project would be spent optimising the model architecture, fine-tuning the hyperparameters, and enhancing the model's performance on fresh, untested data. To further improve the model's performance we will be adding more features which are likely to impact the value of Amazon stock. This would help improve the efficiency of predictions and help predict future value of the stock for the next 10 days. |

MILESTONES

|  |
| --- |
| 1. Data loading: The stock price data has been successfully loaded into a Pandas dataframe. 2. Data preprocessing: The data has been normalized using the MinMaxScaler function and split into training and test sets. 3. Input sequence creation: The create\_sequences function has been defined to create input sequences of a specified window size. 4. Model definition: The LSTM model has been defined using the Keras API, with two LSTM layers and one dense layer. 5. Model training: The model has been trained on the training data for 100 epochs with a batch size of 32. 6. Model evaluation: The model's performance has been evaluated on the test data using the root mean squared error (RMSE), Mean Average Error (MAE), R squared. 7. Prediction: The model has been used to predict the next day's closing price based on the latest available data. |

WORK ACCOMPLISHED

|  |  |  |
| --- | --- | --- |
| **TASK NO.** | **DESCRIPTION** | **IMPLEMENTATION** |
| 1 | Data preprocessing: One of the biggest challenges in working with financial data is that it can be noisy and highly variable. We overcame this challenge by normalizing the data using the MinMaxScaler function, which scales the data to a fixed range of 0 to 1. This helps to reduce the impact of outliers and make the data more manageable. | 1.Data filtering: We filtered the input data to only include the 'Close' price column. This is because stock prices can be very volatile and can have sudden spikes or dips due to news events, market conditions, or other factors. By only using the closing price data, we can smooth out some of this volatility and focus on long-term trends in the market. Also the correlation between OHL (Open, high, low) was 1. This carried a risk of collinearity.  2.Data normalization: We used the MinMaxScaler function from the sklearn.preprocessing library to normalize the data. Normalization scales the data to a fixed range of 0 to 1, which helps to reduce the impact of outliers and make the data more manageable for the LSTM model. This makes sure to smoothen the data irrespective of its scale.  3.Data splitting: We split the normalized data into training and test sets. The training set contains 80% of the data and is used to train the LSTM model, while the test set contains the remaining 20% of the data and is used to evaluate the model's performance.  4.Sequence creation: We created sequences of a specified window size from the normalized data. The window size is the number of time steps that the LSTM model uses to make its predictions. We used a window size of 60, which means that the model uses the previous 60 days of closing price data to predict the next day's closing price. |
| 2 | Model training: Training an LSTM model can be computationally intensive and time-consuming, especially when working with large datasets. We overcame this challenge by using the fit method in Keras to train the model on the training data for a specified number of epochs with a batch size of 32. | We used the Adam optimizer and mean squared error loss function for model training. The Adam optimizer is a popular optimization algorithm that adjusts the learning rate adaptively during training. The mean squared error loss function is commonly used for regression problems, such as predicting stock prices.  We trained the model for 100 epochs, with a batch size of 32. During training, we used the training set of data to update the model's weights and minimize the loss function.  To prevent the model from overfitting the training data, we monitored the loss function on the test set during training. This allowed us to determine when the model was no longer improving on the test set and to stop training at an appropriate time. Setting the patience as 20 and calling the EarlyStopping() class to prevent overfitting of the model. |
| 3 | Prediction: Finally, using the trained LSTM model to make predictions on new data is a key task in this project. We overcame this challenge by using the predict method in Keras to predict the next day's closing price based on the latest available data. | We used the trained LSTM model to predict the closing prices of the stock on the test set. The predictions were made using the last 60 days of data as input.  We rescaled the predicted values back to their original scale using the MinMaxScaler object that we fit to the data during preprocessing. This allowed us to compare the predicted values to the actual values on the test set.  We calculated the root mean squared error (RMSE) between the predicted and actual values on the test set. The low RMSE value indicates that the LSTM model was able to make accurate predictions. We achieved a 64% accuracy on R squared.  We visualized the predicted and actual values on the test set using a line chart. The visualization allowed us to visually inspect the model's performance and see how well it predicted the stock prices. |

RISKS AND ROADBLOCKS

|  |  |  |
| --- | --- | --- |
| **RISK NO.** | **DESCRIPTION** | **FIX** |
| 1 | Data preprocessing errors | We normalized using MinMaxScaler. If the input data contains NaN or infinite values, this can cause issues with the normalization process. Additionally, if the data is not properly formatted or contains outliers, it may lead to poor model performance. Since there were very few rows with NAN values, we dropped them off. |
| 2 | Sequence creation errors | The create\_sequences function creates sequences of input and output data for the LSTM model. If the window size or sequence length is too small or too large, it can lead to suboptimal model performance. Additionally, if the sequence is not properly created or indexed, it may cause issues during model training and evaluation. |
| 3 | Model configuration errors | The model is defined using the Sequential class from Keras and includes two LSTM layers and a single output layer. If the number of layers or the number of neurons in each layer is too small or too large, it can lead to poor model performance. Additionally, if the model is not properly compiled or the loss function is not appropriate for the problem, it may cause issues during model training and evaluation. |
| 4 | Model training errors | The model is trained using the fit method, with a specified number of epochs and batch size. If the number of epochs or batch size is too small or too large, it can lead to suboptimal model performance or overfitting. Additionally, if the training data is not properly shuffled or the validation data is not properly specified, it may cause issues during model training and evaluation. |

UPCOMING WORK

|  |  |  |  |
| --- | --- | --- | --- |
| TASK NO. | STATUS | DESCRIPTION | DETAILS |
| 1. | On going | Multi variate LSTM – Adding more features to increase accuracy | Since the stock market is a highly dynamic time series subject, it is crucial to experiment with multiple features that could potentially impact the closing price of the stock (AMZN). To achieve a higher level of accuracy, we aim to integrate our existing model with various technical and macroeconomic factors that have a strong correlation with the closing price.  The preliminary tests with US interest rates, RSI, and MACD have already increased our accuracy to more than 75%. By incorporating additional factors such as EMA, and other tech stocks like MSFT, AAPL, and GOOG, we expect to further improve the model's relevance and prediction capabilities.  Our approach is to continuously refine the model by merging these additional features, which will significantly enhance its predictive power. As we fine-tune our model and expand the range of factors it considers, we anticipate even greater accuracy and increased relevance in predicting the AMZN stock price. |
| 2. | Up coming | Predicting Closing on future data | Current model predicts future data based on historical closing data over a 2 year period. Once we introduce more features like USD currency, USD interest rates, MACD, RSI , predicting future values gets complicated since those features don’t exist in the future to base our predictions on. To solve this we would use ARIMA along with LSTM to predict features and then use the predicted features into our LSTM model to predict closing of future 10 days. |
| 3. | Up coming | Confidence Interval | Introducing confidence interval-based predictions is a valuable strategy to enhance the credibility of our stock price forecasts. By incorporating 90%, 95%, and 99% confidence intervals around our 10-day predictions, we present a more robust understanding of the potential stock price movement. This probabilistic approach adds an essential layer of confidence to our predictions, assisting stakeholders in making well-informed decisions based on a range of possible outcomes. With this added level of certainty, our model becomes increasingly relevant and useful in the dynamic landscape of the stock market. |

OVERALL PROJECT PROGRESS TIMELINE

MILESTONE 6

Model evaluation

**PROJECT START DATE**

**PROJECT END DATE**

**CURRENT TIMELINE POSITION**

**Milestone 7**

**Prediction**

MILESTONE 5

Model Definition

ROADBLOCK 3

Model configuration errors

MILESTONE 4

Model Definition

MILESTONE 3

Input sequence creation

MILESTONE 2

Data Preprocessing

MILESTONE 1

Data Loading

ROADBLOCK 2

Sequence Creation Errors

ROADBLOCK 1

Data preprocessing errors