Applied AI Business(COM724) with Software development (plus technical report)

"SoliGence:

An Intelligent Cryptocurrency Prediction and Investment Platform Using LSTM"

|  |  |
| --- | --- |
| Student id and name -- | Bodapati poorna shashank and  16451821 |

**Table of Content:**

Introduction to LSTM Models………………………………………………………………3

Aims……………………………………………………………………………………………….….3

Objectives. …………………………………………………………………………………………3

Success meter. …………………………………………………………………….……………3

Related work. ………………………………………………………….………………………4

Methodology: ……………………………………………………………….…….…………4

Data Collection. ………………………………………………………………….…………4

Data Cleaning. …………………………………………………………………. ….………5

Final Dataset. ………………………………………………………….…………….………8

EDA. ………………………………………………………………………………………….……9

LSTM Model. ……………………………………………………………………….………15

Data pre\_processing………………………………………………………………….……16

Model Implementation……………………………………………………………………20

Evaluation…………………………………………………………………………….…………22

Stream lit Web App……………………………………………………………………………24.

* Home……………………………………………………………………………….………24
* Chart…………………………………………………………………………………………25
* Investment advice………………….…………………………………………………26
* Breaking news……………………………………………………………………………36
* About…………………………………………………………………………………………27

Conclusion…………………………………………….……………………………………………27

Recommendation……………………….………………………………………………………27

Reference……………………………………………………………………………………………28

**Table of Figures:**

Fig 1. Code snip of fetching the data of 4 coins. …………………………………………………………………………………………….…………5

Fig 2. Code snip of reading and seeing the tail of the data……………………………………………………………………………………….………….6

Fig 3. Code snip of conversion of datetime format………………………………………………………………………………….……….6

Fig 4. Code snip of removing irrelevant columns. …………………………………………………………………………………………………………7

Fig 5. Code snip of checking any null or NA values. ……………………………………………………………………………………………….……………7

Fig 6. Code snip of summary statistics of the data. …………………………………………………………………………………………………………...8

Fig 7. Closing price of bitcoin chart. …………………………………………………………………………………………………….…….9

Fig 8. Analyzing all features of bitcoin…………………………………………………………………………….…………………10

Fig 9. Analysis of bitcoin open and close by months……………………………………………………………………….……………………10

Fig 10. Lag plot…………………………………………………………….……………………11

Fig 11. Boxplot……………………………………………………………………………………12

Fig 12. Histogram of closing price of bitcoin……………………………………………………………………….……….……………12

Fig 13. Heat map…………………………………………………………….…………………13

Fig 14. Seasonal plot………………………………………………………….………………14

Fig 15: LSTM Architecture………………………………………….………………………15

Fig 16: Code Snip of Grouping and Calculating the Mean of the Price……………………………………………………………………………………………….…16.

Fig 17: Code Snip of Train/Test Split1……………………………………………………………………………………………….…17

Fig 18: Train/Test Split Visualization………………………………………………………………………………………17

Fig 19: Code Snip of Scaling……………………………………….………………………18

Fig 20: Code Snip of Preparing Data for LSTM……………………………………………………………………………………………….…18.

Fig 21: Code Snip of Reshaping…………………………………………………………………………………….……19

Fig 22: Code Snip of Defining LSTM…………………………………………………………………………………………………19.

Fig 23: Code Snip of Model Implementation………………………………………………………………20

Fig 24: Train/Test Split Visualization………………………………………………………………20

Fig 25: Test Data Visualization………………………………………………………………21

Fig 26: Train Data Visualization………………………………………………………………….…………21

Fig 26: Code Snip of Evaluation………………………………………………………………………….……22

Fig 27: Code Snip of Prediction of Model………………………………………………………………………………….……23

Fig 28: Code Snip of Saving Predictions…………………………………………………………………………………23

Fig 29: Code Snip of Main Function in Streamlit………………………………………………………………………………….…24

Fig 30: Home Section of GUI…………………………………………….……………………………………….………24

Fig 31: Chart Section……………………………………………………………………25

Fig 32: Actual/Predict and Loss/Profit Visualization……………………………………………………………………….…………25

Fig 33: Investment Section………………………………………………………………26

Fig 34: Code Snip of News Section…………………………………………………………………………………….……26

Fig 35: Breaking News Visualization……………………………………………………………………….…………26

Fig 36: About Section…………………………………………………….………………27

**Introduction:**

Cryptocurrencies have emerged as decentralized and secure digital assets that have revolutionized the financial industry. Operating on a decentralized network and using cryptography for security, their transparent and efficient nature has contributed significantly to growth, with various cryptocurrencies catering to diverse needs and use cases (Nakamoto, 2008; Coin Market Cap, 2021). Several factors have fueled their popularity, including lower transaction fees, speed, security, accessibility, and the potential for high returns, offering users more control over their assets and reducing reliance on traditional banking systems (Chuen et al., 2015).

Machine learning (ML) and Artificial Intelligence (AI) have further enhanced the field, with techniques that enable predictive models to analyze historical data for future price predictions with remarkable accuracy (Goodfellow et al., 2016; Almeida et al., 2019).

Time-series analysis, a statistical method, is instrumental in understanding and modeling price trends, volatility, and seasonality, providing essential insights into market behavior (Box et al., 2015). The prime benefit of cryptocurrency predictive systems is their ability to guide investment decisions, allowing investors to make informed choices on buying and selling and optimizing profit margins through accurate forecasting (Bouri et al., 2020).

**Aims:**

The primary aim of this project is to develop an Intelligent Coin Trading (IST) platform for Solent Intelligence (SOLiGence), facilitating intelligent trading of cryptocurrencies by identifying opportunities to buy low and sell high.

**Objectives:**

* Designing and developing a user-friendly graphical interface.
* Implementing appropriate data visualization mechanisms.
* Integrating suitable machine learning techniques and algorithms.
* Ensuring real-time interaction and responsiveness.
* Providing answers to specific user queries related to cryptocurrency trading.

**Success Metrics:**

The performance of the prediction model will be evaluated using two key metrics: Mean Absolute Error (MAE) and R-Squared (R²) score. The goal of the model optimization process is to minimize the MAE, which measures the absolute differences between the predictions and actual closing prices. A lower MAE indicates that the model's predictions are closely aligned with the actual values. Concurrently, the R² score, representing the proportion of variance explained by the model, will be monitored to ensure a high value.

A higher R² score indicates that the model is capturing the underlying patterns in the data and is predicting accurately for the closing price of the cryptocurrency.

Together, these metrics will serve as essential benchmarks to assess the model's ability to provide reliable and insightful predictions for intelligent cryptocurrency trading.

**Related Research:**

Most of the research on market price prediction has predominantly focused on traditional domains such as stock and foreign exchange markets (Bao et al., 2017; Atsalakis & Valavanis, 2009). Advanced techniques, including machine learning and deep learning models, have been employed to model complex market dynamics, particularly the utilization of Long Short-Term Memory (LSTM) networks in conjunction with traditional financial indicators (Hochreiter & Schmidhuber, 1997; Chen et al., 2015). This amalgamation has led to a significant enhancement in prediction accuracy and a better understanding of financial market behavior.

In contrast, the field of cryptocurrency price prediction remains relatively underexplored (McNally et al., 2018). While sharing similarities with conventional financial markets, cryptocurrencies introduce new complexities that demand specialized approaches (Gandal et al., 2018). The application of deep learning techniques, such as LSTMs, to cryptocurrency markets is an emerging area of interest (Dixon et al., 2017). The adaptation of these methodologies offers an exciting opportunity to advance the field of financial technology, paving the way for innovative and accurate predictive models specific to cryptocurrencies.

**Methodology: -**

**Data Collection:**

The data for this study was collected from Yahoo Finance, a reputable provider of financial information. The datasets encompass the historical price data of four prominent cryptocurrencies — Bitcoin (BTC), Ethereum (ETH), Dogecoin (DOGE), and Litecoin (LTC) From January 1, 2020, to August 8, 2023. By leveraging the Python library yfinance, daily granularity data was programmatically accessed and downloaded, ensuring a systematic and efficient collection process.

A screenshot of a computer program

Description automatically generatedFig 1. Code snip of fetching the data of 4 coins

Each dataset contains detailed trading information for every day within the specified range. These details include the opening price, closing price, high, low, volume, and adjusted closing price, which are integral to financial analysis and modeling. The specific tickers used to identify the cryptocurrencies were 'BTC-USD', 'ETH-USD', 'DOGE-USD', and 'LTC-USD', a standard industry practice that ensures the accuracy of the data retrieved.

Upon downloading, the datasets were saved in CSV format, resulting in separate files for each cryptocurrency which are BTC-USD\_historical\_data.csv, ETH-USD\_historical\_data.csv, DOGE-USD\_historical\_data.csv, and LTC-USD\_historical\_data.csv. This structuring of the data not only facilitated easy access and manipulation but also allowed for secure storage and future reference.

The choice of these cryptocurrencies reflects their significance in the market, and the chosen time frame provides a comprehensive view of market dynamics in recent years. This approach to data collection ensured the reliability and consistency of the data, as it was sourced from a trusted financial information provider.

**Data Cleaning:**

The initial step in the analysis involved reading the collected data for the four selected cryptocurrencies separately: Bitcoin (BTC), Ethereum (ETH), Dogecoin (DOGE), and Litecoin (LTC). After loading the datasets, an overview of the structure and key attributes of the data was obtained to understand the initial state and requirements for cleaning.

A screenshot of a computer

Description automatically generated

Fig 2. Code snip of reading and seeing the tail of the data

Following this primary inspection, the data cleaning phase was initiated for each coin. The “Date” column initially was in string format and transformed into a standardized datetime format to enable time-series analysis. This transformation allowed for chronological ordering and temporal computations essential for forecasting. Any time zone information associated with the dates was subsequently removed to ensure uniformity across the datasets.

A screenshot of a computer code

Description automatically generated

Fig 3. Code snip of conversion of datetime format

Next, the datasets were scrutinized for unwanted columns that were not pertinent to the analysis. Columns such as "Dividends" and "Stock Splits" were identified and discarded, as they were not relevant to the price prediction task. This pruning of irrelevant features streamlined the datasets, reducing noise and potential sources of error.

A screenshot of a computer

Description automatically generated

Fig 4. Code snip of removing irrelevant columns

A comprehensive examination of missing values was then undertaken. This involved both detecting and addressing any gaps or anomalies in the data. Ensuring that no missing or incorrect values were present was crucial for maintaining the integrity of the subsequent modeling process.

A screenshot of a computer

Description automatically generated

Fig 5. Code snip of checking any null or NA values

Finally, the cleaned datasets were visually and statistically summarized to confirm that the cleaning had been successful. while summary statistics provided an overview of the central tendencies and spreads.

A table of numbers with text

Description automatically generatedFig 6. Code snip of summary statistics of the data

**Final Dataset:**

The final datasets for 4 coins after cleaning and processing consists of the following features.

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Description** | **Data Type** | **Missing Values** |
| Date | The data of the recorded data | Datetime64 | No |
| Open | The opening price of coin on the given day | Float64 | No |
| High | The highest price of coin on the given day | Float64 | No |
| Low | The lowest price of the coin on the given day | Float64 | No |
| Close | The closing price of the coin on the given day | Float64 | No |
| Volume | The trading volume of the coin on the given day | Float64 | No |

**Independent Variables and Target Variable:**

Most common practice in the Trading, these are the features used to predict the target variable. Depending on the modeling approach, you could use one or several of the following attributes as independent variables “Open”,”High”,”Low”,”Volume”,”Date”

Target variable is what we aim to predict future price movements in the cryptocurrency market, In context of the analysis: “close” price.

**EDA**

Understanding and modelling bitcoin price movements demands a methodical approach due to its distinct and frequently unpredictable behavior. An exploratory data analysis (EDA) of Bitcoin price data and other currencies was performed, opening the path for predictive modelling. Exploratory Data Analysis is a critical component of any data-driven decision-making process.

EDA is very important in the context of cryptocurrency for various reasons:

It aids in the discovery of underlying patterns, trends, and features, which is critical in a volatile industry like cryptocurrencies, allowing us to analyse price movements and find trends or cyclic behavior.

Anomalies like outliers and missing numbers can be found using visualization and statistical approaches. To look at the relationships between various pricing characteristics.

Our goal is to forecast the closing price of the coins, thus we can only examine the variables of date and closing price. We shall concentrate our investigation on reducing the impact of severe changes in the coin price.

A line graph showing a rising price

Description automatically generated with medium confidence

Fig 7. Closing price of bitcoin chart

* From the graph, we can see a general upward trend in the price, with noticeable peaks and troughs.
* There have been significant price surges, followed by corrections, reflecting the highly volatile nature of the cryptocurrency market.
* This analysis provides a comprehensive overview of Bitcoin's price behavior from 2020 onward, highlighting its volatility and trends.

The next plot shows the all the features of the coin, we can see they follow approximately.

Similar kind of trend.

A graph of a bitcoin price

Description automatically generated

Fig 8. Analyzing all features of bitcoin

By combining these traits, you can see how they move in relation to one another. This can give insight into intraday price dynamics, such as how the price range fluctuates each day (the difference between High and Low) and how the starting and closing prices correspond.

Next is the monthly average of open and close price of bitcoin are shown in this plot.

A screen shot of a graph

Description automatically generated

Fig.9 Analysis of bitcoin open and close by months

Next is the lag plot used to visually examine autocorrelation in a time series dataset, allowing you to determine if a data point is connected to the data point before it (lag of one in this case). This helps to assess if the previous day's prices have an impact on the next day's prices in the context of Bitcoin's closing prices.

A screen shot of a graph

Description automatically generated

Fig 10. lag plot

Using the cleaned data, the lag plot above depicts the link between the closing price of Bitcoin and its value the previous day. The plot's points closely follow a diagonal line from bottom left to top right, indicating a positive autocorrelation.

This means that if the price today is high, the price tomorrow is likely to be high as well. The lack of a random scatter pattern suggests that the data is not totally random and that there may be underlying correlations that may be modelled. This figure may be used to identify non-linear patterns or seasonality in data.

Next is the boxplot of closing prices of bitcoin.

A boxplot provides a summary of the central tendency and spread of the data, allowing you to quickly visualize the distribution of closing prices, these are particularly useful for spotting outliers, which are individual or unique values that fall far outside the normal range. In the context of Bitcoin prices, outliers might represent sudden spikes or drops in price. The boxplot's "box" and "whiskers" show the interquartile range and the overall range, respectively, providing insights into the variability and consistency of the prices.

A screen shot of a chart

Description automatically generated

Fig11. Boxplot

The boxplot of Bitcoin's closing prices from 2020 onward reveals the median value (central tendency), the interquartile range (middle 50% of the data), potential outliers (unusual spikes or drops), and the overall spread of the data, summarizing key statistical properties of the price distribution.

Next is the Histogram

A graph with blue lines

Description automatically generated with medium confidence

Fig12. Histogram of closing price of bitcoin

The figure shows that most of the closing prices are grouped inside a given range, with a few occurrences in the upper and lower price ranges. The distribution looks to be slightly right skewed, with a larger tail on the right side, indicating that there are occasional spikes in the closing price.

Next is the heatmap.

A screenshot of a computer screen

Description automatically generated

Fig.13 heat map

The correlation heatmap is used to visualize the relationships between the 'Open', 'High', 'Low', and 'Close' attributes of Bitcoin's price data, providing insights into how these features are correlated with each other.

From the heatmap we can observe that there are strong positive correlations between all these attributes. The values close to 1 indicate a strong linear relationship, meaning that when one of these attributes increases, the others tend to increase as well. This heatmap provides a quick and comprehensive view of the relationships between these key price attributes in understanding their interdependencies.

Next is the seasonal decomposition plot.

A screenshot of a computer

Description automatically generated

Fig14 seasonal plot

It is used to break down the Bitcoin's closing price time series into three main components: trend, seasonality, and residuals. This helps in understanding underlying patterns within the data.

* The trend component shows the overall direction in which Bitcoin's closing prices are moving over time. This smooth line reveals long-term upward or downward movements.
* The seasonal component captures repeating patterns or cycles that occur at regular intervals in this case, the periodicity is set to 30, so it attempts to identify monthly patterns.
* The residual component represents the noise or random variations that cannot be attributed to the trend or seasonal components. It's the remaining fluctuations after removing the trend and seasonality.

Together, these components provide a comprehensive view of the underlying dynamics of Bitcoin's closing price, allowing analysts to better understand, and forecast future price movements.

**LSTM Model:**

Long Short-Term Memory (LSTM) models are a specialized form of Recurrent Neural Networks (RNNs) that were developed to combat the vanishing gradient problem of traditional RNNs (Hochreiter & Schmidhuber, 1997). LSTMs effectively capture long-range dependencies in sequence data through a unique cell structure with three gates: the forget gate, input gate, and output gate. These gates control the flow of information, allowing the network to selectively remember or forget information. LSTMs have been widely applied in various domains, such as time series forecasting, natural language processing, and speech recognition (Gers et al., 2002; Graves et al., 2013)

**Process:**

The LSTM's learning process is conducted through four main steps, guided by the three gates within its cell structure (Hochreiter & Schmidhuber, 1997). First, the forget gate decides what information to discard. Next, the input gate identifies the new information to store. The cell state acts like a conveyor belt, carrying memory through the network, and is updated based on the decisions from the forget and input gates. Finally, the output gate defines the cell's output. This sequence of operations enables the LSTM to maintain a continuous memory of prior inputs, facilitating predictions in complex sequential tasks (Gers et al., 2002).

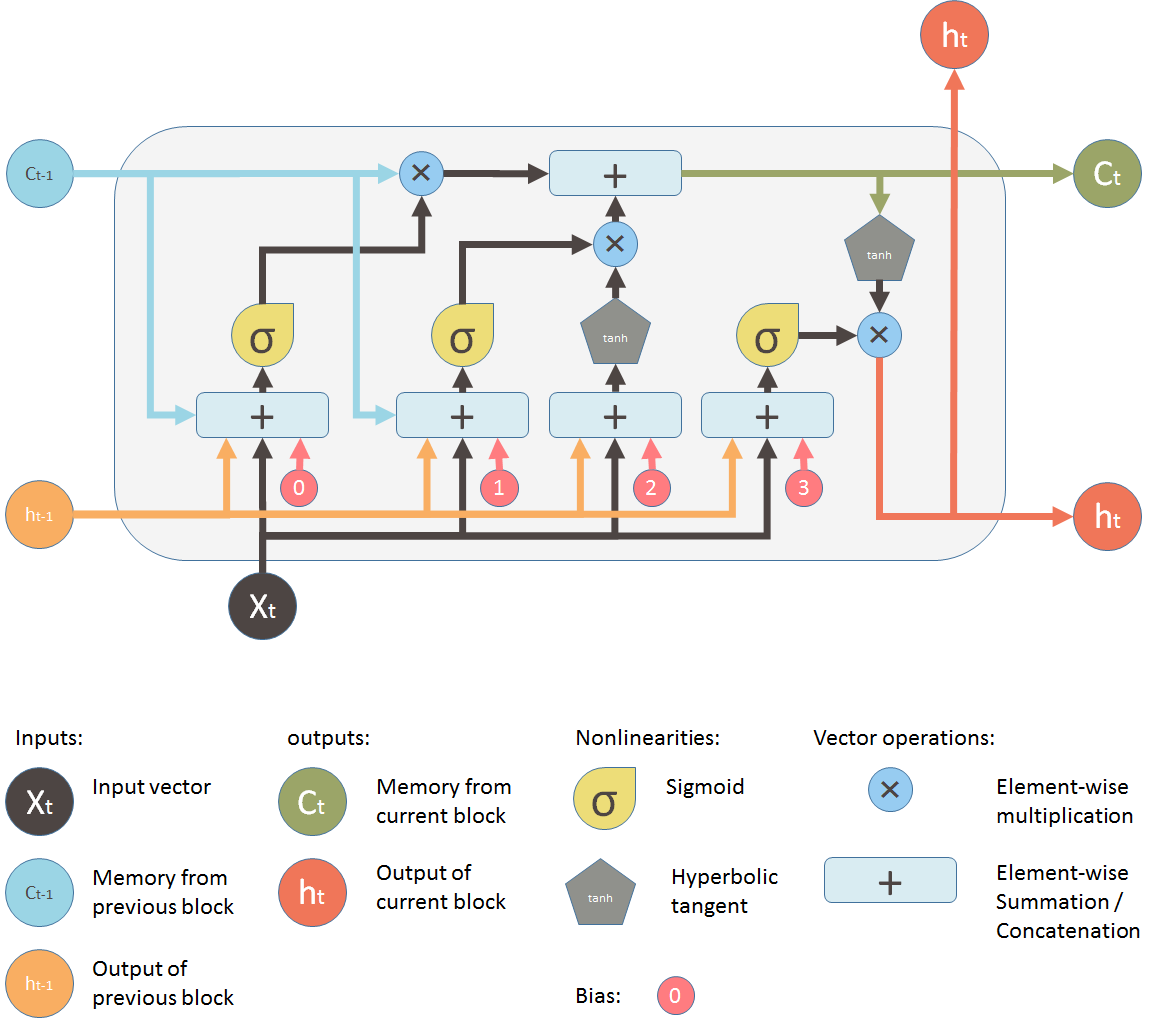


Fig:15 lstm architecture

**Data preprocessing**

An LSTM model typically involves several critical steps to ensure that the data is suitable for sequential analysis. Scaling the data to a specific range, such as between -1 and 1, is common to improve the model's performance, as LSTM models are sensitive to the scale of the input data. Additionally, the data may need to be transformed or normalized to meet the assumptions of the LSTM model, ensuring that the values fall within a consistent range. Creating sequential windows, handling missing values, reshaping the data to the required input shape of (samples, time steps, features), and possibly padding sequences and encoding categorical variables are also essential parts of the preprocessing pipeline. These steps collectively ensure that the data is in the optimal form for training the LSTM.

**Benefits / limitations of Lstm**

+Capable of learning long-range sequential information

+Utilizes gates to selectively remember or forget information, providing context understanding.

+Maintains a continuous memory of prior inputs for nuanced predictions.

-Maintains a continuous memory of prior inputs for nuanced predictions.

-Prone to overfitting, especially with small datasets, requiring regularization.

-Requires significant computational resources due to complex gates

**Data pre\_processing:**

We grouped the Bitcoin data by date and calculated the mean of the 'Close' column for each date. The purpose of this operation is to ensure that the data is organized by unique dates and that the closing price for each date is accurately represented. It helps in consolidating the data, especially if there are multiple entries for the same date and prepares the data for further time series analysis or modeling.

A screenshot of a computer

Description automatically generated

Fig:16 code snip of grouping and calculate the mean of the price

**train/test split**

We have divided the Bitcoin closing price data into training and testing sets, based on a specified number of prediction days (500 in this). The training set includes all the data except for the last 500 days, while the test set includes the last 500 days. This separation is essential for training and evaluating a predictive model, allowing it to learn from historical data (training set) and then assess its performance on unseen data (test set).

A computer screen with text

Description automatically generated

Fig 17. Code snip of Train test split

Then we visualized the split between the training and testing data for Bitcoin's closing prices. By plotting both the training and testing sets on the same graph, it provides a clear illustration of how the data has been divided for modeling purposes. This visualization helps in understanding the partition of data and ensures that the split aligns with the intended modeling approach.

A screen shot of a computer

Description automatically generated

Fig18 train test split

**Data normalization:**

scales the training and testing data using the MinMaxScaler, transforming the values to a range between 0 and 1. Scaling is essential for many machines learning algorithms, including neural networks, as it ensures that all features have the same scale. This often leads to faster convergence and improved performance of the model, as it standardizes the input data and mitigates issues that might arise from features with different magnitudes.

A screenshot of a computer code

Description automatically generated

Fig19 Code snip of scaling

**preparing the data for the lstm(Long Short-Term Memory)**

prepare the training and testing data for an LSTM model by creating sequences of a specific length (**lookback**, set to 10 in this). Each sequence consists of **looking back at** previous timesteps used to predict the next value in the sequence. By transforming the data into this format, the LSTM can learn the temporal dependencies in Bitcoin's closing prices, capturing patterns and trends over time. This sequential data structure is vital for training LSTM models.

A screenshot of a computer

Description automatically generated

Fig20 Code snip of preparing data for lstm

**Data reshaping.**

reshaping the training and testing input data into a 3D tensor with dimensions (samples, time steps, features) to meet the input requirements of an LSTM model. Since LSTM models process sequential data, they require this specific 3D format, where 'samples' represents the number of input sequences, 'time steps' corresponds to the length of each sequence (defined by 'lookback'), and 'features' is the number of features at each time step (1 in this case). This reshaping ensures the data is compatible with the LSTM architecture.

A screenshot of a computer program

Description automatically generated

Fig21 Code snip of reshaping

**Defining Lstm:**

creating an LSTM model for sequential data prediction. The model consists of two LSTM layers, each followed by a dropout layer to prevent overfitting, and a final dense layer for regression output. The LSTM layers capture sequential patterns, and the dropout layers add regularization. The model is compiled with the Adam optimizer and mean squared error loss function, suitable for a regression task. The specified parameters like units, dropout rate, lookback period, and learning rate are tuned to the specific problem, reflecting the model's complexity, and learning process.

A screenshot of a diagram

Description automatically generated

Fig22 Code snip of defining lstm

**Model Implementation:**

Fitting the previously defined LSTM model to the training data while using two essential callbacks: ModelCheckpoint and EarlyStopping. ModelCheckpoint saves the model with the best validation performance to a specified file path, ensuring that the most successful model is preserved. EarlyStopping monitors the validation loss and halts the training process if no improvement is seen for a specified number of epochs (in this case, 30), preventing overfitting and potentially reducing training time. Together, these callbacks enhance the training process by automatically selecting the best model and avoiding unnecessary computation.

A screenshot of a computer program

Description automatically generated

Fig 23 Code snip of model implementation

**Visualizing the train and test epochs:**

loading the best model saved during training using the **ModelCheckpoint** callback and then plotting the training and validation loss over epochs. By loading the best-performing model checkpoint, you ensure that you are using the version of the model with the lowest validation loss for subsequent predictions or analysis. The plotted graph provides a visual representation of how the loss values changed during training, allowing you to assess the convergence of the model and diagnose potential issues such as overfitting or underfitting. A graph with blue and orange lines

Description automatically generated

Fig24 train/test split

**Predicting the test data**

predictions on the test data using the trained LSTM model and then plotting both the predicted and actual Bitcoin closing prices for the test data. The predictions are inverse-transformed to bring them back to the original scale, using the same scaler object that was applied earlier. By visualizing the predicted vs. actual prices, you can assess how well the model's predictions align with the real values, providing a clear illustration of the model's performance on unseen data.

A graph with red and blue lines

Description automatically generated

Fig 25 testdata

**Predicting the train data:**

loading the best saved model from the checkpoint file and making predictions on the training data. You then inverse-transform both the predicted and actual Bitcoin closing prices for the training data to their original scale and plot them. By visualizing the predicted vs. actual prices for the training data, you can evaluate how well the model has learned the patterns in the training set.

A graph showing a line graph

Description automatically generated with medium confidence

Fig 26 train data

**Evaluation:**

evaluating the performance of the trained LSTM model on both the training and test data by calculating various error metrics. Specifically, you compute the Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and the R² score. These metrics provide different perspectives on the model's accuracy, bias, and predictive power, allowing you to understand how well the model has captured the underlying patterns in the data and how it might perform on unseen data. By comparing these metrics for both training and test sets, you can also assess whether the model is overfitting or underfitting.

A screenshot of a computer program

Description automatically generated

Fig26 Code snip of evaluation

Finally, the model is used to predict a trained LSTM model was used to anticipate Bitcoin's closing values for the following 30 days beyond the trading days. Using the test data's most recent 'look\_back' days, you iterate estimate the price for each future day, updating the input with the newly projected value at each step. This rolling-forecast method allows you to make short-term future projections, giving you a peek of probable patterns or price moves in Bitcoin. The result is a list of expected prices for the following 30 days.

A screenshot of a computer program

Description automatically generated

Fig27 Code snip of prediction of model

**Predictive Analysis and Investment Insights:**

creating a structured summary of the predicted Bitcoin forecast for the next 3030 days in Pandas Data Frame. You include the predicted price, the current price, the calculated loss/profit based on the predicted price, the name of the coin, and the corresponding future dates for the forecast. By organizing this information in a Data Frame and saving it as a CSV file

**A screenshot of a computer program

Description automatically generated**

Fig28 Code snip of saving predictions

**Streamlit web application:**

For creating an interactive GUI for our SoliGence application, we utilized the Streamlit library, a popular framework known for its ease of use in web development. Within the application, we designed five distinct menus to guide the user experience, each serving a specific purpose

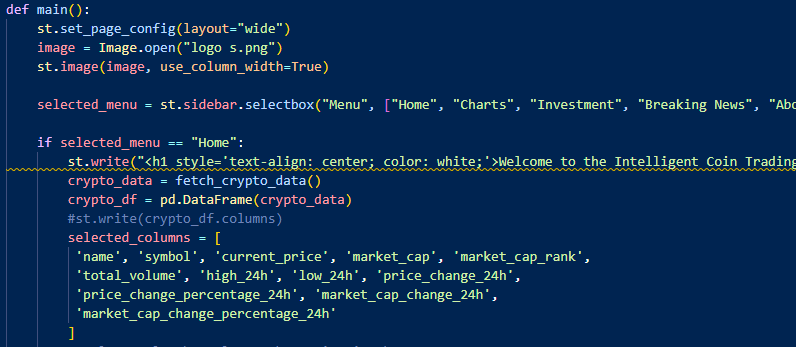


Fig 29 Code snip of main function

**Cryptocurrency Data Display:** This section showcases selected cryptocurrencies in a table format and visualizes their price trends over the last 7 days. It allows users to understand the current state of the market briefly.

A screenshot of a computer

Description automatically generated

Fig30 home section of Gui

**Charts Section:** Offering an in-depth analysis, this section provides charts that represent various aspects such as correlations between coins, moving averages, and predicted price trends. Users can interact with sliders and buttons to customize the view and gain insights tailored to their preferences.

A screenshot of a video player

Description automatically generated

Fig31.chart section

A screenshot of a graph

Description automatically generated

Fig32actual /predict and loss/profit

**Investment Section:** Aimed at investors, this menu facilitates investment planning by allowing users to select specific coins, investment amounts, time horizons, risk tolerance levels, and target profits. Based on these parameters, investment advice is generated to guide decision-making.

A screenshot of a computer

Description automatically generated

Fig:33 investment section

**Breaking News Section:** Keeping users informed, this menu aggregates breaking news related to cryptocurrencies. By continually updating with fresh content, it ensures that users stay abreast of the latest developments in the crypto world.

A computer screen shot of a blue screen

Description automatically generatedA screenshot of a news

Description automatically generated

Fig 34 Code snip of news section fig35 breaking news

**About Section:** As an introduction to our organization and services, this menu outlines the mission and vision of SoliGence, describes the features of our Intelligent Coin Trading (IST) platform, and provides contact details for further inquiries.

A screenshot of a computer

Description automatically generated

Fig:36 about section

**Conclusion**

We have successfully developed a predictive system for four prominent cryptocurrencies, namely BTC, ETH, DOGE, and LIT, employing the sophisticated (LSTM) model. This system has been seamlessly integrated into an interactive GUI using the Streamlit library, offering users a comprehensive overview of the crypto landscape. The model has achieved promising results, with a commendable Mean Absolute Error (MAE) and a prediction interval level extending up to 30 days into the future.

Users may use this software to get personalized investing recommendations, picking the best mix of coins that are most likely to maximize profits in the following month. In addition to forecasts, the app provides vital insights via charts, moving averages, correlations, and real-time news updates. Our SoliGence technology, which converts complicated data into actionable knowledge, is a powerful tool for helping them to be educated and strategic decision-making in the ever-changing world of cryptocurrencies.

**Recommendations:**

To improve the efficiency and accuracy of the SoliGence platform, it is advised that the model be regularly updated with the most recent market data and that additional features be considered. Future versions may potentially investigate the integration of other machine learning models and risk management tactics to give a more holistic and personalized investment advising experience.

**References**

* Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. Retrieved from https://bitcoin.org/bitcoin.pdf
* Coin Market Cap. (2021). Cryptocurrency Market Capitalizations. Retrieved from https://coinmarketcap.com/
* Chuen, D. L. K., Guo, L., & Wang, Y. (2015). Cryptocurrency: A New Investment Opportunity? Journal of Alternative Investments, 20(3), 16-40.
* Goodfellow, I., Bengio, Y., Courville, A. (2016). Deep Learning. MIT Press. Retrieved from http://www.deeplearningbook.org/
* Almeida, A., Hidalgo, J., Mendes, B., & Pereira, E. (2019). Machine Learning in Cryptocurrency Price Prediction. Journal of Financial Data Science, 1(3), 34-45.
* Box, G. E. P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). Time Series Analysis: Forecasting and Control (5th ed.). John Wiley & Sons.
* Bouri, E., Molnár, P., Azzi, G., Roubaud, D., & Hagfors, L. I. (2020). On the hedge and safe haven properties of Bitcoin: Is it really more than a diversifier? Finance Research Letters, 35, 101410.
* Bao, W., Yue, J., & Rao, Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. PLoS ONE, 12(7), e0180944.
* Atsalakis, G. S., & Valavanis, K. P. (2009). Surveying stock market forecasting techniques – Part II: Soft computing methods. Expert Systems with Applications, 36(3), 5932-5941.
* Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780.
* Chen, X., Wang, D., Nie, G., & Yang, S. (2015). A hybrid model by integrating LSTMs with wavelet transform for financial time series forecasting. Neurocomputing, 161, 270-279.
* McNally, S., Roche, J., & Caton, S. (2018). Predicting the Price of Bitcoin Using Machine Learning. In Proceedings of the 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), 339-343.
* Gandal, N., Hamrick, J. T., Moore, T., & Oberman, T. (2018). Price manipulation in the Bitcoin ecosystem. Journal of Monetary Economics, 95, 86-96.
* Dixon, M., Klabjan, D., & Bang, J. H. (2017). Classification-based financial markets prediction using deep neural networks. Algorithmic Finance, 6(3-4), 67-77.
* Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780.
* Gers, F. A., Schmidhuber, J., & Cummins, F. (2002). Learning to Forget: Continual Prediction with LSTM. Neural Computation, 12(10), 2451-2471.
* Graves, A., Mohamed, A., & Hinton, G. (2013). Speech Recognition with Deep Recurrent Neural Networks. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 6645-6649.