**CARDANO (ADA) PRICE PREDICTION**

1. **INTRODUCTION**

Cryptocurrencies are a digital means of exchange which rely on the transmission of digital information, utilizing cryptographic methods to ensure legitimate, unique transactions. The cryptocurrency market has evolved erratically and at unprecedented speed over the course of its short lifespan. The popularity of cryptocurrencies has skyrocketed in 2017 due to several consecutive months of super exponential growth of their market capitalization (El Bahrawy et al., 2017), which peaked at more than $800 billion in Jan. 2018. Today, there are more than 1500 actively traded cryptocurrencies.

The Binance Exchange is an exchange founded in 2017 offering trade in more than 500 cryptocurrencies and virtual tokens. It provides a crypto wallet for its traders where they can store their electronic funds. Binance offers services around the trading, listing, fundraising, and de-listing, or withdrawal of cryptocurrencies (*Binance Exchange Definition*, n.d.).

The aim of the project is to predict the closing price of Cardano (ADA).

Our project uses the Python-Binance API to fetch the data for the cryptocurrency Cardano (ADA). The Binance API is a method that allows us to connect to the Binance servers via Python or several other programming languages (*Binance Python API – A Step-by-Step Guide - AlgoTrading101 Blog*, 2020). With it, trading can be automated. More specifically, Binance has a RESTful API that uses HTTP requests to send and receive data. Further, there is also a WebSocket available that enables the streaming of data such as price quotes and account updates (*Binance Python API – A Step-by-Step Guide - AlgoTrading101 Blog*, 2020).

1. **METHODOLOGY**

*Libraries:*  We are using pip to install the necessary libraries. In the project we are using pip to install python-binance, pandas, scikit-learn, matplotlib, keras, tensorflow, plotly, mplfinance libraries. Below code snippet shows the code to pip install the necessary libraries.

Graphical user interface, text

Description automatically generated

When we run the above code to pip install the necessary, we get below result as the libraries are already installed, else the libraries will be installed:

Text

Description automatically generated

*Main Function*: The program starts from here in the main it will call all the requires functions that are needed to extract the data, visualize it, run analysis on it and to train and perform predictions on the data.

Text

Description automatically generated

*Data Extraction*: Data extraction is the process of collecting or retrieving disparate types of data from a variety of sources. Data extraction makes it possible to consolidate, process, and refine data so that it can be stored in a centralized location to be transformed (What Is Data Extraction? Definition and Examples, 2020). The sources from which data is retrieved may be highly unorganized and unstructured. Extracting data makes it possible to store the data in a neat and organized manner. For this project, the data is extracted from the Binance exchange website using the Binance API. Below code snippet shows the data is collected from the Binance API with interval as 1Day using get\_Klines function, then we are setting the open time as index and transforming the data into float types from string. Below code snippet illustrates the above specified functionality.

Text

Description automatically generated

The data extracted consists of the following: -

Table

Description automatically generated

*Data Preprocessing:* This is the step in which data is converted into a form which makes it ready to be used for analysis. It consists of data cleaning, data transformation and data reduction. To perform these steps, the Pandas library is imported, and the data is imported into a dataframe. Data cleaning involves handling missing data and noisy data. Data transformation includes normalization, attribute selection etc. For our project, we just need the open time, open price, close price, high price, low price, and the volume.

Open Price: The Stock at which opens at the start of market  
High price: The particular stock which made high during that particular day  
Low Price: The Particular stock which made Low during that particular day  
Close Price: The stock closing at the end of the Market hours  
Volume: The number of crypto being traded

Text

Description automatically generated

*Data Visualization:* The graphical representation of information and data is known as data visualization. For this project, the plotly library was imported to plot the candlestick chart. A candlestick is a type of price chart used in technical analysis that displays the high, low, open, and closing prices of a security for a specific period.

Diagram

Description automatically generated

Technical analysts use the candlestick charts to determine when to enter and exit trades. If the candlestick is green, the price closed above where it opened and this candle will be located above and to the right of the previous one, unless it's shorter and of a different color than the previous candle. If the candlestick is red, the price closed below where it opened and this candle will be located below and to the right of the previous one, again unless it's shorter and of a different color than the previous candle (*How to Read a Candlestick Charts*, n.d.). The below code Snippet shows the implementation of the same.

Text

Description automatically generated

Application, Word

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated with medium confidence

Chart

Description automatically generated

Chart

Description automatically generatedChart

Description automatically generated

*Data Analysis:* To perform analysis, the wrapper for TA-lib was imported as TA-lib is originally written in C++. It is used to performed technical analysis on the financial market data. The RSI (Relative Strength Index) was calculated using this library.

RSI indicator: It is used to determine how a stock is performing over a certain period. By default, the RSI measures the changes in an asset's price over 14 periods. The RSI is a momentum indicator, which is a type of technical trading tool that measures the rate at which the price is changing. When momentum increases and the price is rising, it indicates that the stock is being actively bought in the market. If momentum increases to the downside, it means that the selling pressure is increasing. The RSI is also an oscillating indicator that makes it easier for traders to spot overbought or oversold market conditions (Academy, 2020).

MA(Moving Average): The MA is a tool for technical analysis that smooths out price data by creating a constantly updated average price. The direction of the moving average is used to get an idea of which way the price is moving. If it is angled up, then the price is moving up overall and vice-versa. The MA10 plot is a plot of the moving average eover 10 days and MA30 is a plot of the moving average over 30 days.

MACD: The moving average convergence divergence (MACD) is used by traders to monitor the relationship between two moving averages. It is calculated by subtracting a 26-day exponential moving average from a 12-day exponential moving average (*Moving Average (MA) Definition*, n.d.).

Below code snippet illustrates the same.

Text

Description automatically generated

*Chart, histogram

Description automatically generated*

Log10: A logarithmic price scale is a type of scale used on a chart that is plotted such that two equivalent price changes are represented by the same vertical distance on the scale. The distance between the numbers on the scale decreases as the price of the asset increases. They are used for long-term perspective analysis of price changes (*Logarithmic Price Scale Definition*, n.d.). Below code snippet illustrates the same.

Text

Description automatically generated

*Graphical user interface, application, Word

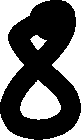
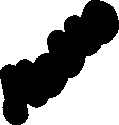
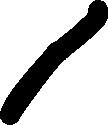
Description automatically generated*

1. **AI MODELS APPLIED**

The target variable which we aim to predict is the Closing price of the stock. For this purpose, the following models were applied-



1. *LSTM*: An LSTM layer consists of a set of recurrently connected blocks, known as memory blocks. These blocks can be thought of as a differentiable version of the memory chips in a digital computer (Brownlee, 2020). There are three gates that regulate the reading, writing and outputting values to and from the cell state-the input, output and forget gate. The input gate decides which information should enter the cell gate. The forget gate decides which information needs attention and which can be ignored. The output gate determines what output(next Hidden State) to generate from the current Internal Cell State.



Text

Description automatically generated

1. *GRU*: Recurrent Neural networks are a type of neural network in which the output from the previous step is fed to the current step as input. RNN’s face the problem of vanishing gradient. This problem is solved by GRU’s (Gated Recurrent Unit) by using the update gate and the reset gate. The update gate determines how much of the past knowledge needs to be passed along into the future. The reset gate determines how much of the past knowledge to forget. The current memory gate is incorporated into the Reset Gate just like the Input Modulation Gate is a sub-part of the Input Gate and is used to introduce some non-linearity into the input and to also make the input Zero-mean.

Text

Description automatically generated

* Below is the code snippet for the function call to initialize the models and call the respective model to predict the data. Here in the function, we are setting the window size to 10 that calculates the 10-day closing price considering the previous 9 day values.

Text

Description automatically generated

* The below is the code snippet where we are initializing the LSTM and GRU models where we are creating 2 layers using Relu activation method. For GRU we are using the number of nodes in the 1st layer as 70 and in the 2nd layer as 80 with a dropout of 20% and Dense value as 1 as we are only predicting only one value. Similarly for LSTM we are specifying the nodes as 300 each in both the layers and with a dropout of 20% and Dense value as 1 as we are only predicting only one value.

Text

Description automatically generated

* Once the model is initialized, we are transforming the training data, we are trying to predict the test data which is calculating the Close Price based on the 10-window size. And we are also evaluating the models built by using RMSE and R^2 value and the graphs for the same are plotted.

Text

Description automatically generated

1. **CONCLUSION**

We tested the performance of two recurrent neural networks on the dataset- LSTM and RNN. After comparing accuracies of both the models, it was found that GRU performed better than LSTM. At a particular point of time, the R2 score of LSTM was 73.31 and for GRU it was 88.71. The RMSE for LSTM was 0.79 and the RMSE for GRU was 0.69. These are variables and their values keep changing as the model is executed at different times with different data. It is also clear from the following graphs that GRU fits better than LSTM. The better performance of GRU can be attributed to the following reasons-

* It has a less complex structure and hence easy to modify.
* It is computationally more efficient.
* GRU’s perform better on less training data.

Chart, line chart

Description automatically generated

Fig- Predicted vs Actual Price using LSTM.

Chart, line chart

Description automatically generated

Fig- Predicted vs Actual Price using GRU.

GRU can better explain the variability of closing price using Open, High, Low, and Volume compared to LSTM as the data set is very small. GRU performs better for smaller datasets and executes faster as it does not require higher memory units. LSTM would have performed better if the dataset were large.

**#NOTE: Please refer to the Installation Guide to run the code.**

1. **REFERENCES**

* ElBahrawy, A., Alessandretti, L., Kandler, A., Pastor-Satorras, R., & Baronchelli, A. (2017). Evolutionary dynamics of the cryptocurrency market. *Royal Society Open Science*, *4*(11), 170623. <https://doi.org/10.1098/rsos.170623>
* *Binance Exchange Definition*. (n.d.). Investopedia. <https://www.investopedia.com/terms/b/binance-exchange.asp#:%7E:text=Binance%20is%20an%20online%20exchange%20where%20users%20can%20trade%20cryptocurrencies.&text=Binance%20provides%20a%20crypto%20wallet,%2C%20Binance%20Coin%20(BNB)>.
* *Binance Python API – A Step-by-Step Guide - AlgoTrading101 Blog*. (2020, November 9). Quantitative Trading Ideas and Guides - AlgoTrading101 Blog. <https://algotrading101.com/learn/binance-python-api-guide/>
* *What is Data Extraction? Definition and Examples*. (2020, October 29). Talend Real-Time Open Source Data Integration Software. <https://www.talend.com/resources/data-extraction-defined/>
* *How to Read a Candlestick Charts*. (n.d.). The Balance. <https://www.thebalance.com/how-to-read-a-candlestick-chart-1031115>
* Academy, B. (2020, December 2). *What Is the RSI Indicator?* Binance Academy. <https://academy.binance.com/en/articles/what-is-the-rsi-indicator>
* *Moving Average (MA) Definition*. (n.d.). Investopedia. <https://www.investopedia.com/terms/m/movingaverage.asp>
* *Logarithmic Price Scale Definition*. (n.d.). Investopedia. <https://www.investopedia.com/terms/l/logarithmicscale.asp>
* Brownlee, J. (2020, February 19). *A Gentle Introduction to Long Short-Term Memory Networks by the Experts*. Machine Learning Mastery. <https://machinelearningmastery.com/gentle-introduction-long-short-term-memory-networks-experts/>
* Kostadinov, S. (2019, November 10). *Understanding GRU Networks - Towards Data Science*. Medium. <https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be>

**PROJECT BY: COVID RESISTANT HUSKY 3 GROUP**

* Srinidhi Karicharla
* Sai Prathap Reddy Cheluri
* Aayushi Gupta
* Atharva Shantanu Kulkarni
* Shivani Sharma