

Applying Predictive Analytics to Financial Markets

COREA KITTII¹, ARON SAMAYOA², AND DANIEL CASCO³

¹Placeholder

²Placeholder

³Placeholder

Placeholder

Placeholder

ABSTRACT Within this paper we present a price prediction application that utilizes tick-level price data from financial markets in order to perform predictive analytics by way of machine learning models. The application includes a data importation tool written Python that makes use of both the PyArrow and Pandas libraries in order to import data from both CSV files and Parquet files, which is then both stored in and managed by a MySQL database. Two types of machine learning models are implemented: Long-Short-Term Memory models and transformer models. These models can be trained on specific securities in order to perform predictive analysis. Instances of fully trained models can be saved for each security in the database, where they can be retrieved for future use. This application provides two interfaces: a user interface and an administrator interface. Users are capable of viewing candlestick charts that are formed out of the tick-level price data, and can view the results of the predictive analysis performed by the machine learning models. Administrators can do the same, while also having the ability to initiate training sessions for the models, as well as any required maintenance of the database. With this application prediction future price movements of financial markets can be done simply, quickly, and accurately.

INDEX TERMS TODO

I. INTRODUCTION

IT is generally understood that accurately predicting the future is extraordinarily difficult, if not impossible; however, that hasn't stopped people from trying, especially in regards to financial markets. Of all the research that has been performed on these markets, predicting future price movements has proven to be one of the most challenging problems due to the variables involved, such as market volatility, investor sentiment, and factors external to the financial market, which include the economy, climate, politics, and more. Where simpler models often fall short due to the complexities of financial time series data, machine learning techniques, particularly deep learning, have shown promise. Of the many different types of deep learning models, Gated Recurrent Units (GRU), Long-Short-Term Memory (LSTM), and Transformer models have displayed promising performance, with our application utilizing the latter two. Extensive research has been performed on this subject, which has explored a wide variety of machine learning techniques in order to find those suitable for price prediction. These range from the more classic support vector machines (SVM) all

the way to the more advanced neural networks. Studies have proven that using deep learning models for price prediction is effective, while determining which models are the most effective is an ongoing process [1]. Precision price prediction has had its importance proven via advancements made in algorithmic trading, especially in high-frequency trading, where decisions must be made based on real-time analysis of the market [2]. Further refining the accuracy of price predictions can be done by utilizing tick-level price data, which can capture even the smallest market movements due to its granularity [3].

Past research has performed comparisons on the performance of GRU, LSTM, and Transformer models in regards to price prediction, with the dataset consisting of Tesla (TSLA) price data ranging from 2015 to 2024. The dataset underwent preprocessing in order to correct for missing values and to normalize price fluctuations, along with other adjustments that allowed for the optimization of training each model. The results of the experiments that were performed showed that LSTM models performed better than both GRU and Transformer models, while Transformers performed the worst out

of the three [1].

II. PROJECT OVERVIEW

Our application will utilize historical tick-level price data from the financial market in order to perform predictive analytics using machine learning models, with the application also providing a tool to import said data. Users will be provided an interface that allows them to view the price data in the form of candlestick charts, as well as view the results of the predictive analysis produced by the machine learning models. Administrators are capable of doing that on top of having the ability to both initiate model training and perform necessary maintenance.

A. SYSTEM DESIGN

The system is designed in a modular fashion, and features the data importation tool, the database, the two types of machine learning models, an interface for administrators, and an interface for normal users. A more detailed description is as follows:

1) Data Importation Tool

The data importation tool is implemented using the Python programming language, which allows us to take advantage of its excellent third-party library support. We utilize both the PyArrow and Pandas libraries in order to process both CSV files and Parquet files.

2) MySQL Database

Once imported, data is stored within a MySQL database for easier access. Candlestick charts are generated using this data, and it is also fed into machine learning models during their training. Each instance of a fully trained model is also stored within the database, allowing for easy retrieval. The database also stores the results of predictive analysis so that they can be viewed by users.

Data within the database is stored according to the following schema:

a: Tick-Level Data

Price data for each security is stored in a table whose name is in the format `TICK_DATA-[security]`.

- `trade_id`: A unique ID is assigned to each trade
- `trade_time`: Unix-timestamp of when the trade took place
- `price`: The value of the security at the time of the trade
- `volume`: How much of the security was traded
- `side`: Whether the trade was a buy or sell order

b: Candlestick Charts

Candlestick charts for each security is stored in a table whose name is in the format `CANDLE_[timeframe]-[security]`.

- `trade_id`: A unique ID is assigned to each trade
- `open_time`: Unix-timestamp of the first trade during the time frame

- `close_time`: Unix-timestamp of the last trade during the time frame
- `open_price`: Value of the security at the start of the time frame
- `high_price`: Highest value the security reached during the time frame
- `low_price`: Lowest value the security reached during the time frame
- `close_price`: Value of the security at the end of the time frame
- `volume`: How much of the security was traded during the time frame

c: Model Instances

All trained models instances are saved in the database in the `MODEL_INSTANCES` table.

- `security`: The security the model was trained on
- `model_id`: A unique ID assigned to each model saved in the database
- `model_type`: Whether the model is an LSTM or a Transformer
- `date_trained`: Timestamp of when the model was trained

d: User Information

User information is saved in the `USER_INFO` table.

- `user_id`: A unique ID is assigned to each user; no duplicate ID's are allowed, and this field cannot be null.
- `first_name`: The user's first name; this field cannot be null, but duplicate values are acceptable.
- `last_name`: The user's last name; this field cannot be null, but duplicate values are acceptable.
- `username`: Each username must be unique; this field cannot be null.
- `password`: Each password must be at least a minimum length, and feature a mix of alphanumeric and special characters. This field cannot be null.
- `admin_status`: Set to true or false depending on whether the user has been granted administrator rights.

3) Machine Learning Models

Within our application we implement two types of machine learning models:

- Long-Short-Term Memory (LSTM) models, which excel at performing predictive analysis on time-series data that spans a long period of time.
- Transformer models, which perform better at predictive analysis on time-series data that spans a shorter period of time, and are generally faster to train than LSTM's.

Models are trained on specific securities to improve the accuracy of their predictions; in other words, each instance of a fully trained model is a specialist in the security it was trained on.

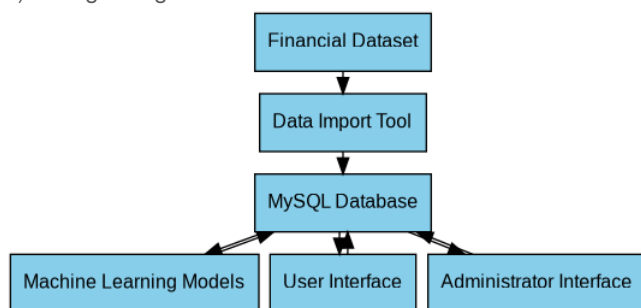
4) User Interface

We use React Native for the user interface, which is a Javascript framework that allows for the development of

cross-platform mobile applications, as well as React Native for Web, which allows for the porting of React Native applications to web browsers and greatly expands the number of platforms users can use to access our application. In regards to the UI itself, it has the following functionalities, with differences existing between normal users and administrators:

- View candlestick charts generated from imported tick-level price data.
- View the results of predictive analysis performed by trained machine learning models.
- Initiate model training sessions (Administrator Only).
- Data importation (Administrator Only).
- Database management and maintenance (Administrator Only).
- User account creation, modification, and deletion (Administrator Only).

5) Design Diagram



B. DATASET

Our choice of datasets to use for testing was limited by two main factors: time and cost. While tick-level price data can be obtained from stock exchanges, it is often at a significant cost. The stock exchange IEX does offer historical tick data for free, but it requires spending a significant amount of time parsing out unneeded data, as the files they provide contain everything that happens on their network. We ultimately decided to source our data from the crypto exchange Binance, who provides their tick data in compressed CSV files that can be readily imported into our database. From their historical data repository at data.binance.vision, we selected the five currency pairs with the most volume from their spot trading data:

- BTC/USDT (Bitcoin & Tether)
- ETH/USDT (Ethereum & Tether)
- ETH/BTC (Ethereum & Bitcoin)
- BTC/USDC (Bitcoin & USD Coin)
- ETH/USDC (Ethereum & USD Coin)

REFERENCES

- [1] J. Xiao, S. Bi, and T. Deng, "Comparative analysis of lstm, gru, and transformer models for stock price prediction," *arXiv*, vol. 2411.05790v1, 2024. [Online]. Available: <https://arxiv.org/abs/2411.05790>
- [2] Investopedia, "Algorithmic trading: Definition, how it works, pros & cons," 2023. [Online]. Available: <https://www.investopedia.com/terms/a/algorithmictrading.asp>

- [3] Intrinio, "Understanding historical tick data and why it is important," 2023. [Online]. Available: <https://intrinio.com/blog/understanding-historical-tick-data>

...