UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Bachelor as Ingeniería Informática Bilingüe

BACHELOR THESIS

Develop a web application to interact with predictive algorithms about the stock market.

Autor: Juan Llamazares Ruiz
Tutor: Francisco Saiz López

June 2023

Juan Llamazares Ruiz
Develop a web application to interact with predictive algorithms about the stock market.

Juan Llamazares Ruiz

C\ Francisco Tomás y Valiente No 11

IMPRESO EN ESPAÑA – PRINTED IN SPAIN

Agradecimientos

Abstract

The interest of modern technology has made it easier to stay informed about the stock market and make investments accordingly. With growing interest among individuals and investors to actively participate in the stock market, having access to tools that help them make better decisions has become increasingly desirable. To study in more detail the stock market is a complex and dynamic asset and can be difficult for these individuals to keep up with the latest trends, company results and world news. By developing a web application that can interact with predictive algorithms, investors can access real-time data and historical trends to make informed investment decisions.

The main purpose of this Bachelor Thesis will be to demonstrate explore the possibility of obtaining a predictive algorithm on the stock market that can be interacted with through a developed web application tool to help individuals and investors to optimize their time and effort to decide the best financial asset. We will consider the utilization of Recurrent Neuronal Networks (RNNs) to support a more accurate stock prediction, and the development of a forecasting model that could be easily implemented into the web application. Finally, we will illustrate the user the strategies to measure the accuracy of the algorithm and the performance of the developed web application.

Furthermore, we go into detail the many technical aspects of web application development such as user interface design and implementation, use of web services, front-end and back-end development.

Keywords

Web application, predictive algorithms, stock market, stock prediction, machine learning, neuronal networks, LSTM, GRU, data analysis, market trends, historical data.

Resumen

El interés de la tecnología moderna ha hecho más fácil mantenerse informado sobre el mercado de valores y realizar inversiones en consecuencia. Con el creciente interés de particulares e inversores por participar activamente en el mercado bursátil, cada vez es más deseable tener acceso a herramientas que les ayuden a tomar mejores decisiones. El mercado bursátil es un activo complejo y dinámico y puede resultar difícil para estos particulares mantenerse al día de las últimas tendencias, los resultados de las empresas y las noticias mundiales. Mediante el desarrollo de una aplicación web capaz de interactuar con algoritmos predictivos, los inversores pueden acceder a datos en tiempo real y a tendencias históricas para tomar decisiones de inversión con conocimiento de causa.

El objetivo principal del Trabajo de Fin de Grado será demostrar explorar la posibilidad de obtener un algoritmo predictivo sobre el mercado de valores con el que se pueda interactuar a través de una herramienta de aplicación web desarrollada para ayudar a particulares e inversores a optimizar su tiempo y esfuerzo para decidir el mejor activo financiero. Consideraremos la utilización de Redes Neuronales Recurrentes (RNNs) para apoyar una predicción bursátil más precisa, y el desarrollo de un modelo de predicción que pueda ser fácilmente implementado en la aplicación web. Por último, ilustraremos al usuario las estrategias para medir la precisión del algoritmo y el rendimiento de la aplicación web desarrollada.

Además, entraremos en detalle en los numerosos aspectos técnicos del desarrollo de aplicaciones web, como el diseño y la implementación de la interfaz de usuario, el uso de servicios web y el desarrollo front-end y back-end.

Palabras clave

Aplicación web, algoritmos predictivos, bolsa, predicción bursátil, aprendizaje automático, redes neuronales, LSTM, GRU, análisis de datos, tendencias del mercado, datos históricos.

Table Of Contents

Figure and Table Index

1

Introduction

In recent years, machine learning has improved to the point that is a very powerful tool to analyse and improve the performance and predictions about future events. It has been used by many institutions and individuals as it has plenty of use cases. For financial purposes has heavily increased in the past years. I end up searching about the use cases of machine learning in the stock market.
The stock market is huge financial market where publicly traded companies' stocks (shares) are bought and sold. It is one of the most important sources of capital for companies, and it allows many individual investors and institutions to buy and sell ownership stakes in publicly traded companies.

One of the main advantages of using machine learning for stock market predictions is its ability to process large amounts of data and identify patterns that may not be visible for the human eye.

Many people in the world invest in this type of asset. Most of them do it based on the financial company results or sometimes based on technical analysis.

1.1. Motivation

Stock market is laborious asset that requires time a deep understanding of the financial markets. More and more people are entering this sector and because of the vast amount of data available its learning curve is increasing constantly. A web-based interactive tool could help investors and non-experience individuals to learn about the effectiveness of prediction models and make more informed decisions. Additionally, the interactive web-based tool will provide an intuitive and user-friendly platform for users to interact with the predictions making it accessible for a large range of users.

1.2. Objectives

The goal of this work is to develop a web-based application tool to provide and interactive platform that allows users to learn about stock market predictions and make informed decisions using different machine learning prediction algorithms.
The main objectives are now described:

- Obtain the minimum possible error compared to the actual closing price; it will be visualized through the web-based tool. Historical data will be used for these predictions.
- Provide different techniques to predict the stock market to improve the user decision-making with visual representations.
- Facilitate data analysis. The tool will allow users to filter, search and analyse different prediction models and compare their effectiveness based on various parameters. I will help users to understand the performance of different models and identify patterns and trends in the data that may not be immediately apparent to the human eye.
- The tool will be up to date as it uses data from financial APIs, it will allow to have an updated, providing accurate up-to-date predictions.

2

State of the art

Machine learning techniques, like Long Short-term Memory (LSTM) networks, have been widely used in recent years for stock market predictions. LSTM networks are a type of recurrent neural network that is used mainly for times series data, like stock predictions. It can learn and remember past patterns in the data, making them very useful for predicting future trends.

Apart from LSTM networks, other studies have also used other machine learning techniques such as Random Forest and Gated Recurrent Unit (GRU)

In the following section, we will see the explanation if each of the technologies involved in the development of the project.

2.1. Machine Learning

The technology used for predictions is called “Machine Learning” it is a division of artificial intelligence and computer science that uses data and algorithms to generate an imitation to an analysis improving its accuracy.

Long Short-Term Memory neuronal network

Long Short-term Memory (LSTM) networks is a type of neural network (RNN) used for a variety of tasks, including stock market predictions, machine translation, speech recognition, and more. It is particularly effective for tasks that involve long-term dependencies, where the past inputs have a significant impact on future outputs,

The LSTM architecture consists of memory cells and gates that regulate the flow of the information through the network. The memory cells are used to store information over time, allowing the network to selectively remember and forget information as needed. The gates are made up of sigmoid and tanh activation functions, they control the flow of the information into and out of the memory cells.

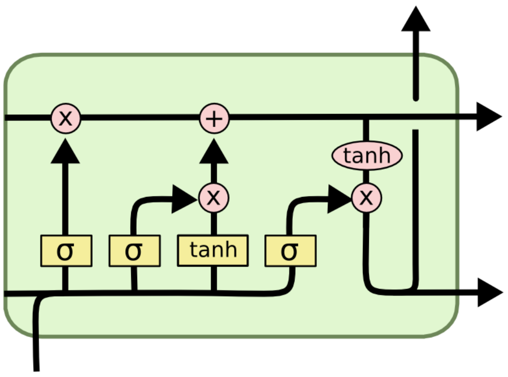


Figure 1: LSTM architecture

Gated Recurrent Units (GRU)

A Gated Recurrent Unit (GRU) is a type of recurrent neural network architecture for sequential data processing. The have been designed to handle sequential data, such a speech signals, time series data and text.

The architecture of the GRU neuronal network consists of an input layer, a hidden layer, and an output layer. The hidden layer contains a set of GRU units, each of which maintains a hidden state vector that is updated based on the input data and the previous hidden state.

The GRU network is computationally efficient and can be trained using standard backpropagation techniques.

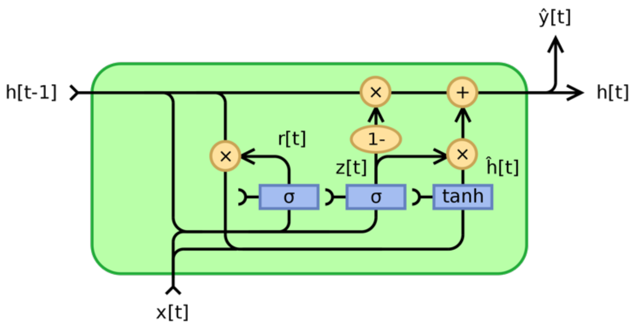


Figure 2: GRU architecture

Long Short-Term Memory (LSTM) vs Gated Recurrent Units (GRU)

The LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit) networks are both types of recurrent neural networks that have been widely used for sequential data processing. While both networks are designed to handle long-term dependencies in sequential data, there are some key differences between the two.

One of the main differences between these two neuronal networks is the number of control gates to control the flow of the information, While GRU uses two gates (an update gate and a reset gate), the LSTM networks uses three gates (an input gate, an output gate and a forget gate), this last gate allows to selectively discard information that is no longer relevant, which is particularly useful for tasks such as language modelling and performance,

2.2. Python 3

Python is a popular, high-level programming language used program modern web applications. It is the most recent version of the Python programming language and has the strongest emphasis on code readability. Python3 offers a comprehensive and extensive library of pre-built modules and packages, and allows developers to create highly functional and dynamic applications. It is an easy to learn language and can be used for both small- and large-scale projects.

Framework Django

All code has been written on Django [1], it is one the most used frameworks in python 3. This framework provides a platform for robust web applications with rapid development and superior scalability.

Python libraries

As part of the development, different python libraries have been used to accomplish the scope of the project, some of the most important libraries are:

- **Tensorflow:** is an open-source library developed by Google for machine learning and artificial intelligence. It is used to train and deploy machine learning models.
- **Keras:** is an open-source software library that provides a Python interface of artificial neural networks.
- **Scikit-learn:** an open-source Python [2] library for machine learning. It provides implementations of many machine learning algorithms and convenient APIs for efficient model training and prediction.
- **Numpy:** is a Python library that supports a large collection of high-level mathematical functions to operate multi-dimensional arrays and matrices.
- **Matplot:** Matplot is a plotting library for Python. It provides many customizable plotting options. Allowing to visualize data in 2D and 3D. It is well used with other Python libraries such as Numpy and Pandas.

- **CSV File reading and writing:** Python library used to read and write the financial data used in the model prediction.

2.3. Javascript

The use of JavaScript in web development has grown exponentially in recent years due to its capability and relative ease of use. Together with HTML and CSS, JavaScript is now essential to create interactive, dynamic websites. More and more applications are being developed with it.

2.4. ChartJS

ChartJs is an open-source JavaScript library for creating charts on web pages. It allows to customize and modify various types of charts, including bar charts, pie charts and scatter plots. In this project it will be used to display the results of the stock prediction algorithms.

2.5. Bootstrap

Bootstrap is an open-source CSS framework for developing responsive and mobile-first websites. It is compatible with all major web browsers, containing HTML and CSS-based design templates. It also provides several helpful components and conventions that can help speed up the frontend development process.

3

Analysis and Design

This chapter will describe the design realised and the procedures for achieving the project goals. it will be divided in two main layers, the backend, responsible for processing and managing data, logic, and communication between the server and client-side of the web application, And the frontend which is responsible for displaying and interacting with the user interface of the web application, including layout, design, and user experience.

In the following figure:

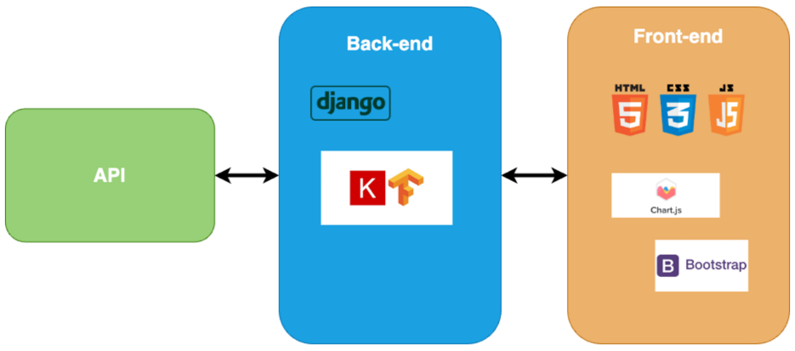


Figure 3: Distributed system architecture

Backend development

3.2 Datasets

The data provided for the development of the project as well as the data used to train the models to predict the future values of the company stock comes from an API. In this case we will use the free tier version of the Alphavantage API.

3.2.1 Alphavantage API

As we have just mentioned, the data will be provided by Alphavantage API [3]. It provides financial market data from traditional asset classes such as stocks, ETFs, or mutual funds. It also provides real-time stock data, charts, and analytics tools. It is designed as a RESful interface allowing users to easily access data from API endpoints. From its many features, we will use the free version tier for the evolution of the project.

The API endpoint used to get historic closing prices of each stock is called *TIME_SERIES_DAILY_AJUSTED*, provides daily open/high/low/close/volume rate values, end-of-day adjusted closing prices, and records of any past stock splits or dividend payments for a chosen global equity, with data spanning over 20 years.

In Figure 7.

Deep Learning

There are different ways to try to predict the value of an asset. In this case, deep algorithms are used for the estimation. Inside of deep learning we find a variety of recurrent neural networks that are algorithms used for learning long term dependencies.

Methodology

The project has been developed following a lean process. Starting from a first web version that worked with all the components of the project, from the API to the frontend development.

Functional and non-functional requirements

4

Implementation

Through this section, the development of the project will be described. It will be divided in different steps.

The distributed system will be divided into two sections, the Back end and the Front-end.

Back-end development

The backend of the project is divided into two main parts. First one is the neuronal network which is the most important part since is used to make the stock predictions. The second main part is the development of the views in the Django framework.

Starting with the neuronal network, the development has followed different steps:

Import data: start by importing the necessary libraries, such as pandas and numpy. Utilize the csv library to read in the data from a CSV file. If the data is not available locally, make use of a financial API to acquire the entire database of stock information. The only metadata necessary for time series prediction is the date and the closing price of the corresponding asset.

Data normalization: once the relevant data has been collected, it is important to undertake data normalization. This step is paramount in ensuring that all input features have similar scales, which is instrumental in attaining optimal performance and successful convergence. There are multiple methods to normalize the data, and for the present study Min-Max scaling technique from a library was adopted.

Dataset split, train, and test arrays: the process of splitting the dataset into training and testing arrays is a crucial step in the machine learning and predictive modelling process. This split is necessary to accurately evaluate the performance of the model on unseen data, and to give a clear understanding of its generalizability.

```
test_ratio = 0.2
training_ratio = 1 - test_ratio
train_size = int(training_ratio * n_days)
test_size = int(test_ratio * n_days)
```

The typically used ratio is 80:20, with 80% of the dataset being allocated to the training array and the remaining 20% reserved for the testing array

Built and trained models: the following step involves utilizing deep learning libraries, such as TensorFlow and Keras, to construct and train LSTM and GRU models. The process for training the LSTM network and GRU network is similar and will now be discussed.

First, define the number of LSTM layers, hidden units, and activation functions by the specified requirements. Afterward, the model can be trained using the normalized data that has been divided into training and validation sets. Hyperparameters, including the learning rate, batch size, and epochs, must be further optimized to enhance the model's performance. Ultimately, the same steps must be repeated for a GRU (Gated Recurrent Unit) model, which is a distinct form of a recurrent neural network.

Test and predictions: Following training, predictions are made on the test dataset and are compared against the actual values. To measure the accuracy and precision of the models, several metrics are used, such as Mean Squared Error (MSE) and Root Mean Square Error (RMSE) to assess the accuracy and precision of the models. We will go into details of the results in section 5.

Calculating Metrics: To accurately evaluate the performance of the models, additional evaluation metrics should be calculated such as root mean absolute error (RMSE), mean absolute percentage error (MAPE). These metrics will provide an exhaustive insight into the model's predictive power and accuracy.

Graph the results: As part of the development of the neuronal network, we have plotted the results of the predictions versus the true values is essential to understand the performance of the model. Line plots or candlestick charts can be used to visualize the results. Through these visualizations, patterns and trends in the stock market data can be identified, and discrepancies or patterns can be identified by comparing predictions with real results.

Improve Neuronal Network: To optimize the performance of the LSTM and GRU models, analyse these models and identify areas of improvement. Try different network architectures, activation functions, regularization techniques, or hyperparameter tuning to increase model performance. Utilize the process of model building, training, and evaluation to see greater results. Iterate through these steps, and assess the results, to reach a higher level of excellence.

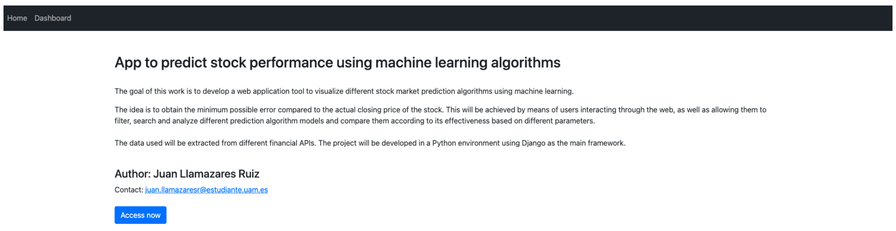
Front-end development

As we have mention in the *Analysis and design* section. In this section, we will talk about all the Front-end development of our web application. All the views, components and functions used will be described. The following figure show a diagram of the structure used in the project.

The main views are divided into different templates.

Home view

The first template it's the home view in Figure 3. It's the first view the user will access, it contains a short description about the project and some contact information.



App to predict stock performance using machine learning algorithms

The goal of this work is to develop a web application tool to visualize different stock market prediction algorithms using machine learning.

The idea is to obtain the minimum possible error compared to the actual closing price of the stock. This will be achieved by means of users interacting through the web, as well as allowing them to filter, search and analyze different prediction algorithm models and compare them according to its effectiveness based on different parameters.

The data used will be extracted from different financial APIs. The project will be developed in a Python environment using Django as the main framework.

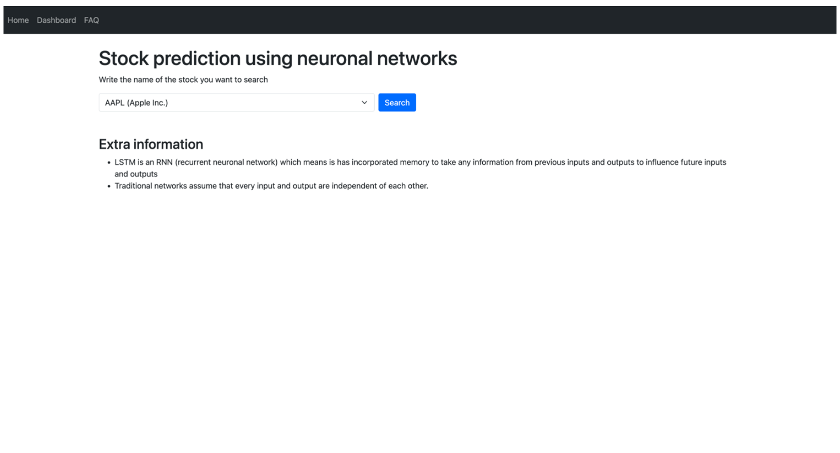
Author: Juan Llamazares Ruiz

Contact: juan.llamazaresr@estudiante.uam.es

Figure 4: Home view

Dashboard view

The dashboard is the most important view, it will contain all the information the user will need to study and analyse the predictions. The view will receive to types of requests, plain get request will only show the search input box and some extra information.



Stock prediction using neuronal networks

Write the name of the stock you want to search

AAPL (Apple Inc.)

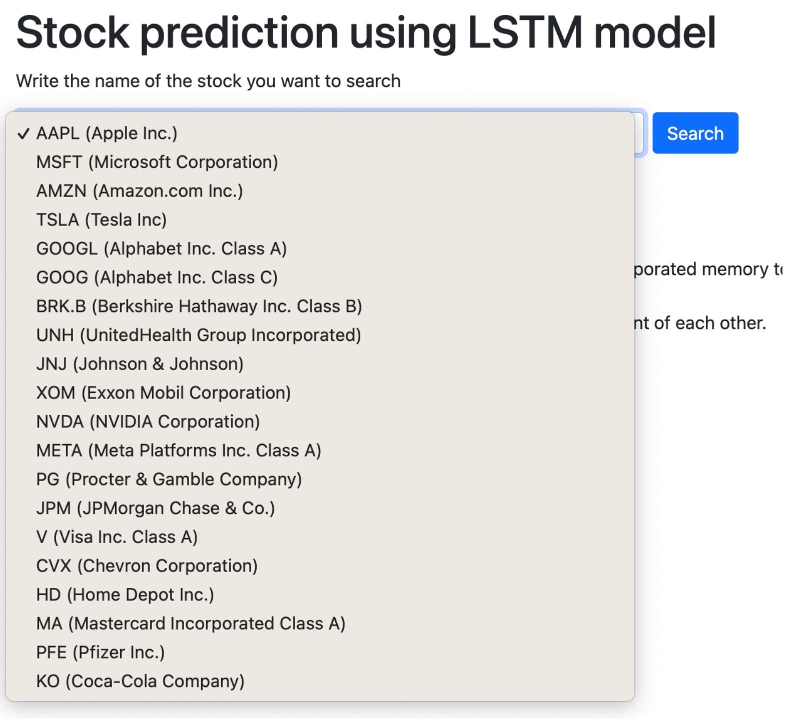
Extra information

- LSTM is an RNN (recurrent neuronal network) which means is has incorporated memory to take any information from previous inputs and outputs to influence future inputs and outputs
- Traditional networks assume that every input and output are independent of each other.

Figure 5: Dashboard view

The search input box will have a list of all the possible assets, in section 5 we will go into detail of the decision to list the assets instead of allowing it all.

Once the user decides what is the asset it is going to predict, it will select the search button to submit the request.



Stock prediction using LSTM model

Write the name of the stock you want to search

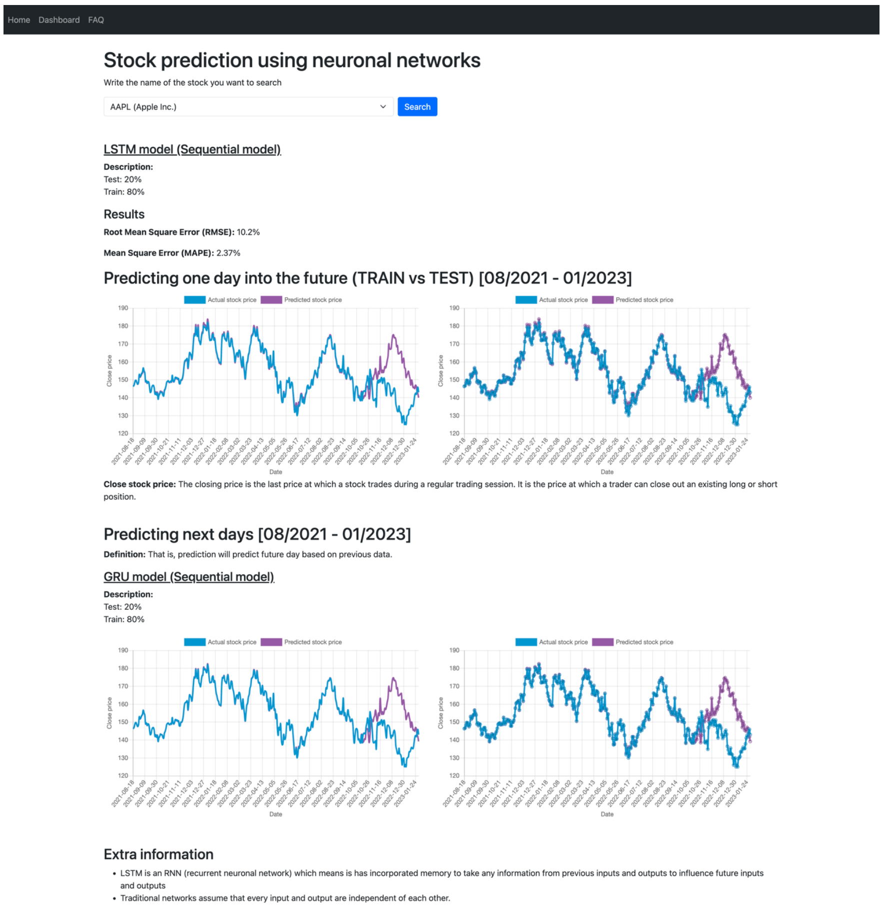
✓ AAPL (Apple Inc.)
MSFT (Microsoft Corporation)
AMZN (Amazon.com Inc.)
TSLA (Tesla Inc)
GOOGL (Alphabet Inc. Class A)
GOOG (Alphabet Inc. Class C)
BRK.B (Berkshire Hathaway Inc. Class B)
UNH (UnitedHealth Group Incorporated)
JNJ (Johnson & Johnson)
XOM (Exxon Mobil Corporation)
NVDA (NVIDIA Corporation)
META (Meta Platforms Inc. Class A)
PG (Procter & Gamble Company)
JPM (JPMorgan Chase & Co.)
V (Visa Inc. Class A)
CVX (Chevron Corporation)
HD (Home Depot Inc.)
MA (Mastercard Incorporated Class A)
PFE (Pfizer Inc.)
KO (Coca-Cola Company)

porated memory t

nt of each other.

Figure 6: List of stocks available

As the result of the request and once the back end it is called, the results will be displayed to the user.



Stock prediction using neuronal networks

Write the name of the stock you want to search

AAPL (Apple Inc.)

LSTM model (Sequential model)

Description:
Test: 20%
Train: 80%

Results

Root Mean Square Error (RMSE): 10.2%

Mean Square Error (MAPE): 2.37%

Predicting one day into the future (TRAIN vs TEST) [08/2021 - 01/2023]

Close stock price: The closing price is the last price at which a stock trades during a regular trading session. It is the price at which a trader can close out an existing long or short position.

Predicting next days [08/2021 - 01/2023]

Definition: That is, prediction will predict future day based on previous data.

GRU model (Sequential model)

Description:
Test: 20%
Train: 80%

Extra information

- LSTM is an RNN (recurrent neuronal network) which means is has incorporated memory to take any information from previous inputs and outputs to influence future inputs and outputs
- Traditional networks assume that every input and output are independent of each other.

Figure 7: Chart view

Frequently asked question's view
Since the are many technical and financial terms, it is very important to show users s

5 Testing and Results

Introduction

Since one of the objectives of this work is to evaluate the prediction of the models developed, we will go into the detail on all the tests carried out to evaluate the efficacy of our web application tool.

As we have mention, testing and results has been a very important role in the evolution of the project. As the project consist in developing a web application, the testing will involve in the improvement of the technology used in the backend. Without testing different parameters and machine learning techniques, the results itself would not be as accurate as the obtained.

The first approach to test is the use of the python library Matplot, used to illustrate the performance of the different models evaluated and to demonstrate the efficacy of the stock prediction process with graphical plots.

One of the problems found on the development of the project is the creation of the neuronal network models, it takes an enormous consideration of time, and we cannot let the user wait that long to generate a model. Therefore, we found a solution, the models are pre-generated, the best solution will be to

Another problem found is the stock market has thousands of different assets, as we have a limit storage space for the models, only the biggest 20 stocks in the US market will be added. You can search the list at the Annex.

LSTM model tests

All the models are created with the same neuronal network structure.

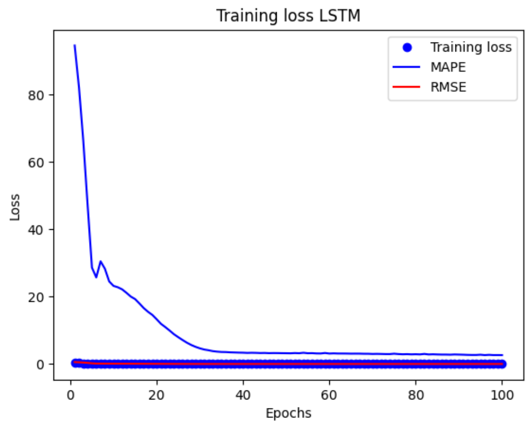


Figure 8: Training loss LSTM

GRU model tests

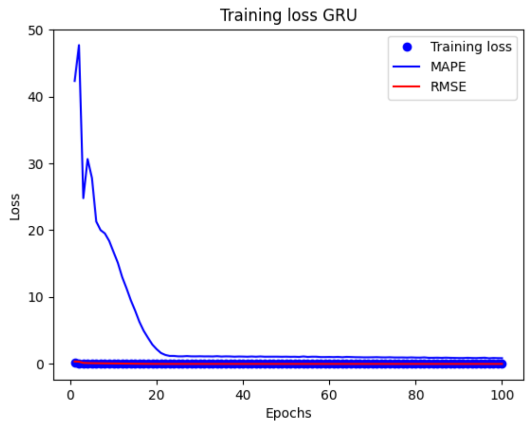


Figure 9: Training loss GRU

Metrics

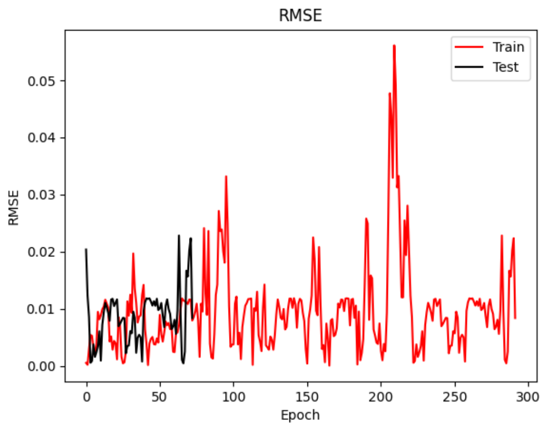


Figure 10: Metric RMSE (Root Mean Square Error)

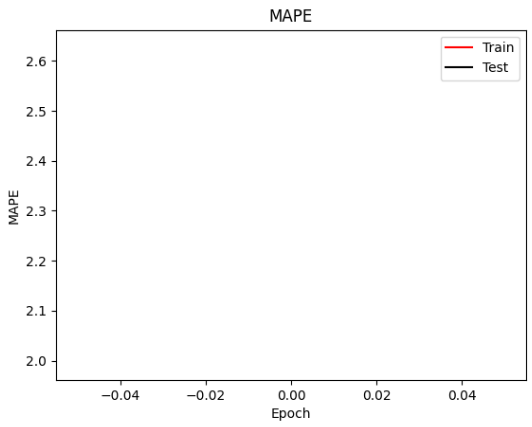


Figure 11: Metric MAPE (Mean Absolute Percentage Error)

Improvements

We will now comment very few errors found on the project and how can they be solved.

First, the stock split error, it is caused because of the use of a free tier API. [4]

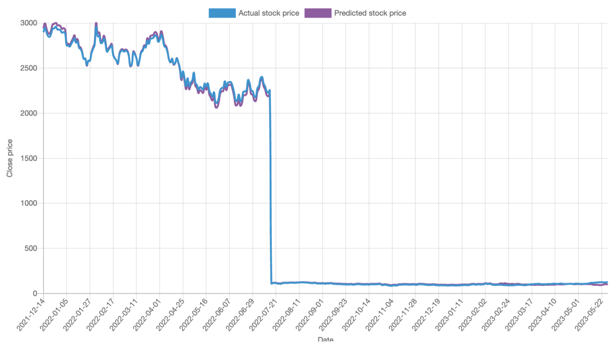


Figure 12: Split error

Alternatives to improve results

Improve financial API (free tier API does not consider splits and other data)

Improve the size of the testing

Use other techniques like

6

Conclusions and future work

Conclusions

To conclude with the project, we have successfully created a web application to allow users and other individuals interact and experience with a tool to predict some financial assets, in this case stocks with recurrent neuronal networks. It will also teach investors another approach of understanding the future market trends using machine learning technology.

Talking about the neuronal networks used on the project, we can conclude that stock prediction based only on the historical price's movement will not be the only factor that affects the future price movement. However, using this recurrent neuronal network like LSTM or GRU will provide in a way the stock trend for the future days.

The conclusion of this technical bachelor thesis is that stock prediction using neural networks should not solely be relied upon. While neural networks can be used to analyse the past data and trends to generate predictions, these predictions should be considered in combination with an analysis of other market trends and financial data. Using a combination of both methodologies will increase the accuracy of the predictions being made. Instead of relying solely on the output generated from a web application tool, a comprehensive analysis of the markets and stocks should be conducted. This will help to provide a deeper insight into the stocks and the trends of the market which should then be used to generate the most accurate predictions.

Future work

There are different alternatives to improve this project, we will go into the details of the most effective ways to keep the development of the web application.

Bibliography

[1] “Alphavantage API,” [Online]. Available: https://www.alphavantage.co/.

[2] “Alphavantage API,” [Online]. Available: https://www.alphavantage.co/.

[3] “Python,” [Online]. Available: https://www.python.org/. [Accessed 2023].

Terminology

Technical terms

- **API:** Application Programming Interface is a set of functionalities and protocols for facilitating communication between two or more computer systems.

Financial terms

- **Stock split:** is when a stock's existing shares are split into multiple smaller shares to reduce their market price. It is typically done when a company's stock has become too expensive for potential investors to buy.

Appendices

A

```python
        url = 'https://www.alphavantage.co/query'
        params = {"function": "TIME_SERIES_DAILY_ADJUSTED", "symbol": symbol, "apikey": API_KEY, "outputsize": "full"}
        try:
            r = requests.get(url, params=params)
        except requests.exceptions.HTTPError as err:
            raise SystemExit(err)
```

Figure 13: Alphavantage API endpoint

```python
from keras.models import Sequential
from keras.layers import Dense, GRU
from keras.layers import LSTM
```

Figure 14: Functions used to create model from python library Keras

```python
def create_lstm_model(train, metrics):
    lstm_model = Sequential()
    # First layer
    lstm_model.add(LSTM(units=50, return_sequences=True, input_shape=(train.shape[1], 1)))
    # Second layer
    lstm_model.add(LSTM(units=50))
    lstm_model.add(Dense(1))
    # Compiling the RNN (Recurrent Neuronal Network)
    lstm_model.compile(optimizer="adam", loss='mse', metrics=metrics)
    # Fitting the RNN to the Training set
    history = lstm_model.fit(train, train, epochs=100, batch_size=32, verbose=2)
    return lstm_model, history
```

Figure 15: Function to create LSTM neuronal network

```python
def create_gru_model(train, metrics):
    gru_model = Sequential()
    # First layer
    gru_model.add(GRU(units=50, return_sequences=True, input_shape=(train.shape[1], 1)))
    # Second layer
    gru_model.add(GRU(units=50))
    gru_model.add(Dense(1))
    # Compiling the RNN (Recurrent Neuronal Network)
    gru_model.compile(optimizer="adam", loss='mse', metrics=metrics)
    # Fitting the RNN to the Training set
    history = gru_model.fit(train, train, epochs=100, batch_size=32, verbose=2)
    return gru_model, history
```

Figure 16: Function to create GRU neuronal network

```python
    try:
        if new_model:
            lstm_model, history_lstm = create_lstm_model(train, metrics)
            lstm_model.save("models/lstm_model_" + symbol + ".h5")
            history_dict_lstm = history_lstm.history
            json.dump(history_dict_lstm, open("models/lstm_model_" + symbol + "_history.json", "w"))

            gru_model, history_gru = create_gru_model(train, metrics)
            gru_model.save("models/gru_model_" + symbol + ".h5")
            history_dict_gru = history_gru.history
            json.dump(history_dict_gru, open("models/gru_model_" + symbol + "_history.json", "w"))
        else:
            try:
                lstm_model = keras.models.load_model("models/lstm_model_" + symbol + ".h5")
                history_dict_lstm = json.load(open("models/lstm_model_" + symbol + "_history.json", "r"))
            except OSError:
                lstm_model, history = create_lstm_model(train, metrics)
                lstm_model.save("models/lstm_model_" + symbol + ".h5")
                history_dict_lstm = history.history
                json.dump(history_dict_lstm, open("models/lstm_model_" + symbol + "_history.json", "w"))

    except:
        print("[ERROR] Creating/Reading model")
```

Figure 17: Method to load / save neuronal network

List of chosen stocks from the stock market

1. AAPL (Apple Inc.)
2. MSFT (Microsoft Corporation)
3. AMZN (Amazon.com Inc.)
4. TSLA (Tesla Inc)
5. GOOGL (Alphabet Inc. Class A)
6. GOOG (Alphabet Inc. Class C)
7. BRK.B (Berkshire Hathaway Inc. Class B)
8. UNH (UnitedHealth Group Incorporated)
9. JNJ (Johnson & Johnson)
10. XOM (Exxon Mobil Corporation)
11. NVDA (NVIDIA Corporation)
12. META (Meta Platforms Inc. Class A)
13. PG (Procter & Gamble Company)
14. JPM (JPMorgan Chase & Co.)
15. V (Visa Inc. Class A)
16. CVX (Chevron Corporation)
17. HD (Home Depot Inc.)
18. MA (Mastercard Incorporated Class A)
19. PFE (Pfizer Inc.)
20. KO (Coca-Cola Company)