

**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**

**Todos los derechos reservados.**

Queda prohibida, salvo excepción prevista en la Ley, cualquier forma de reproducción, distribución comunicación pública y transformación de esta obra sin contar con la autorización de los titulares de la propiedad intelectual.

La infracción de los derechos mencionados puede ser constitutiva de delito contra la propiedad intelectual (arts. 270 y sgts. del Código Penal).

**DERECHOS RESERVADOS**

© 3 de Noviembre de 2017 por UNIVERSIDAD AUTÓNOMA DE MADRID Francisco Tomás y Valiente, nº 1  
Madrid, 28049  
Spain

**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

# 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, las bases de datos, 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.

# 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, databases, 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.

# Table Of Contents

<b>Resumen .....</b>	<b>3</b>
<b>Palabras clave .....</b>	<b>3</b>
<b>Abstract .....</b>	<b>IV</b>
<b>Keywords .....</b>	<b>IV</b>
<b>Table Of Contents .....</b>	<b>V</b>
<b>Figure and Table Index .....</b>	<b>VIII</b>
<b>Introduction .....</b>	<b>1</b>
1.1. Motivation.....	1
1.2. Objectives .....	1
<b>State of the art .....</b>	<b>3</b>
2.1. Machine Learning .....	3
Long Short-Term Memory neuronal network .....	3
Gated Recurrent Units (GRU).....	4
Long Short-Term Memory (LSTM) vs Gated Recurrent Units (GRU) .....	4
2.2. Python 3 .....	5
Framework Django.....	5
Python libraries .....	5
2.3. Javascript.....	5
2.4. ChartJS .....	6
2.5. Bootstrap.....	6
<b>Analysis and Design .....</b>	<b>7</b>
Backend development .....	7
3.2 Datasets .....	7
<b>3.2.1 Alphavantage API.....</b>	<b>7</b>
3.1 Deep Learning .....	8
3.3.1 Develop LSTM model .....	8
3.4 Frontend libraries .....	8
<b>Implementation.....</b>	<b>9</b>
Back-end development .....	9
Front-end development.....	11
<b>Testing and Results .....</b>	<b>13</b>
Introduction .....	13

LSTM model tests .....	13
GRU model tests .....	13
Errors found.....	13
Alternatives to improve results .....	14
<b>Conclusions and future work .....</b>	<b>15</b>
Conclusions.....	15
Future work .....	15
<b>Bibliography.....</b>	<b>16</b>
<b>Terminology .....</b>	<b>17</b>
Technical terms .....	17
Financial terms .....	17
<b>Appendices.....</b>	<b>18</b>
<b>Example of Alphavantage API response .....</b>	<b>20</b>



# Figure and Table Index

<b>Figure 1:</b> LSTM architecture.....	4
<b>Figure 2:</b> GRU architecture .....	4
<b>Figure 3:</b> Distributed system architecture.....	7
<b>Figure 4:</b> Home view .....	11
<b>Figure 5:</b> Dashboard view .....	11
<b>Figure 6:</b> List of stocks available .....	12
<b>Figure 7:</b> Chart view .....	12
<b>Figure 8:</b> Split error .....	14
<b>Figure 9:</b> Alphavantage API endpoint .....	20
<b>Figure 10:</b> Function to create LSTM neuronal network .....	20



# 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:

---

**Error! Use the Home tab to apply Título 3 to the text that you want to appear here.**

- 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.

# 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.

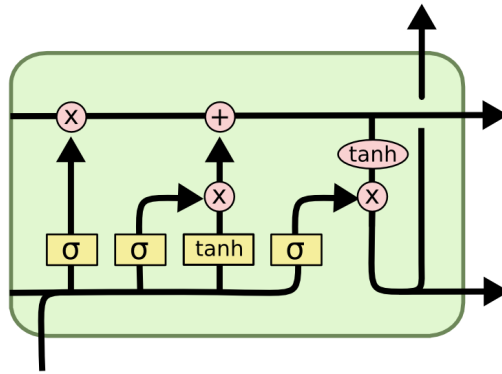


Figure 1: LSTM architecture

## Gated Recurrent Units (GRU)

A Gated Recurrent Unit (GRU) is a type of recurrent neural network architecture for sequential data processing. They have been designed to handle sequential data, such as 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.

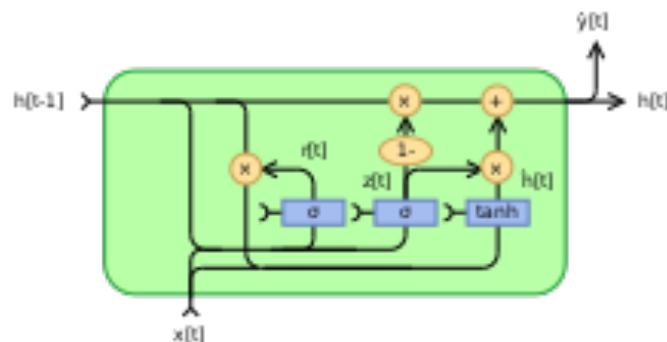


Figure 2: GRU architecture

## 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 that is widely used for web development

### 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

- **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:** is an open-source Python [4] library for machine learning. It provides a range of tools for tasks.
- **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.

# 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:

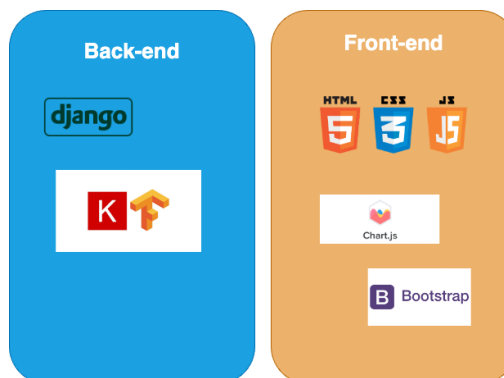


Figure 3: Distributed system architecture

## 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 [2]. 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\_ADJUSTED*, 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.

## 3.1 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.

### 3.3.1 Develop LSTM model

## 3.4 Frontend libraries

For the front e



# Implementation

---

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

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.

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:** Begin by importing the required libraries, such as pandas and numpy. Use the csv library to read the data from a CSV file. If the data is not stored locally, it will make a call to a financial API to retrieve the full database of stock. The only metadata we need for the time series prediction is the date and the closing price of the asset.

**Data normalization:** Once you have the data required it is crucial to normalize the data. This step ensures that all input features have similar scales, which helps in achieving better performance and convergence. There are a few different methods to normalize the data. In this case we have used Min-Max Scaling technique from library.

**Dataset split, train, and test arrays:** The division of the dataset into training and testing arrays is a fundamental element of machine learning and predictive modelling. This division has the purpose of ensuring an appropriate evaluation of the model's performance over unseen data, which in turn provides a thorough understanding of its generalization capacity. The split ratio used is 80:20 is adopted, with 80% of the dataset assigned to the training array, and the remaining 20% to the testing array.

**Built and train models:** Next step, is to build and train the LSTM, and GRU models using deep learning libraries like TensorFlow and Keras. We will only explain the process with the LSTM network since it's the same for the GRU network.

First, define the number of LSTM layers, hidden units, and activation functions according to your specific requirements. Train the model using the normalized data, split into training and validation sets. Adjust hyperparameters like learning rate, batch size, and number of epochs to optimize model performance. Repeat this step for the GRU (Gated Recurrent Unit) model, which is another type of recurrent neural network.

**Predictions and test:** After completing training, predictions are made on the test dataset and are compared against the actual values. To measure the accuracy and precision of the models, various 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. This includes mean absolute error (MAE), mean absolute percentage error (MAPE) and R-squared (R<sup>2</sup>) score. These metrics will provide an exhaustive insight into the model's predictive power and accuracy.

**Plot results:** In the last 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. Such visualizations can help to identify patterns and trends in the stock market data that the model was able to capture. By comparing the predictions and ground truth data, any discrepancies or patterns can be observed.

**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.

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.

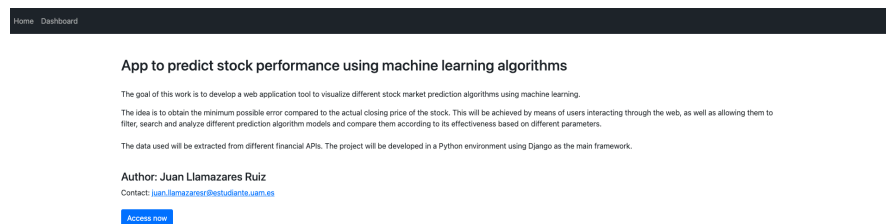


Figure 4: Home view

The dashboard will be the main view of the project, it has a

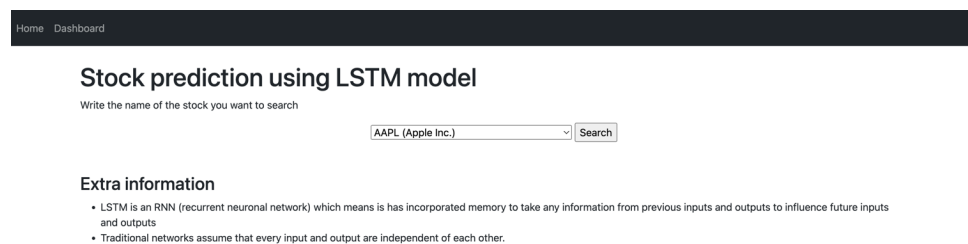


Figure 5: Dashboard view

Once the user interacts with the search input, it will have access to the list of all the stock models generated.

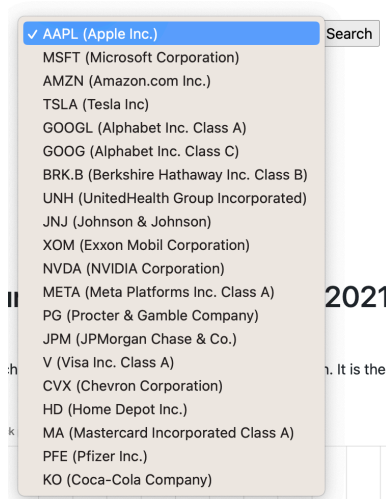


Figure 6: List of stocks available

#### LSTM model (Sequential model)

**Description:**

Test: 20%

Train: 80%

**Root Mean Square Error (RMSE):** 10.2%

**Mean Square Error (MAPE):** 2.69%

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

**Definition:**

**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.



Figure 7: Chart view

# Testing and Results

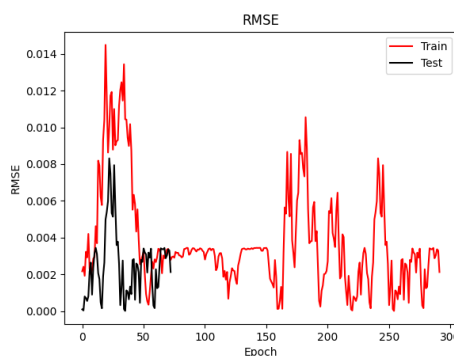
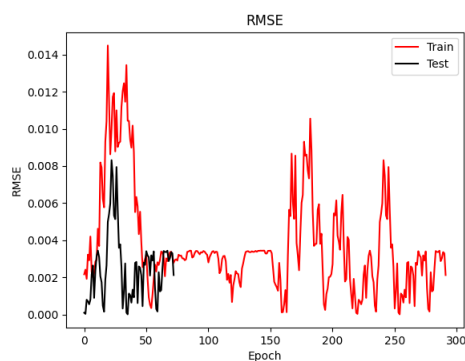
## Introduction

In this section, we will go into detail on all the tests carried out including the development of results obtained.

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 consist 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 Plots from Pyplot, used to illustrate the performance of the different models evaluated and to demonstrate the efficacy of the stock prediction process.

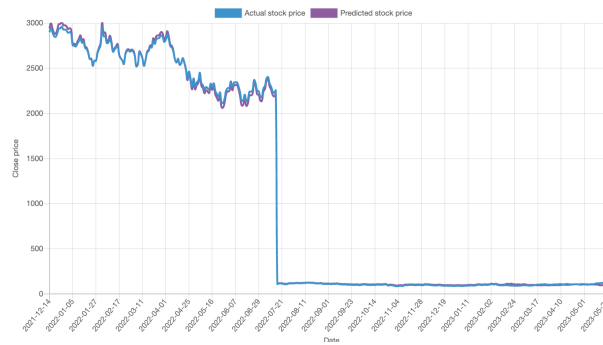
## LSTM model tests



## GRU model tests

## Errors found

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.



**Figure 8:** 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

# Conclusions and future work

---

## Conclusions

To conclude with the project, we have successfully created a web application to allow users to interact and experience with a tool to predict some financial assets, in this case stocks with recurrent neuronal networks.

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





# Example of Alphavantage API

## response

---

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

Figure 9: Alphavantage API endpoint

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

Figure 10: Function to create LSTM neuronal network

**Error! Use the Home tab to apply Título 3 to the text that you want to appear here.**

---