Abstract

This end-of-study project entitled "Development of a machine learning application for stock price prediction" is presented with the purpose of obtaining the National Diploma of Engineers in Industrial Electronics at the National Engineering School of Sousse was carried out within the company Intelligencia IT & Financial Solutions.

The project aims to provide the investors with a third-party investment web application to navigate the stock market. This is done using machine learning and web technologies.

As suggested by the classical analysis approach, technical and fundamental analysis were implemented in a more advanced way by integrating a deep learning model as a technical analysis engine and the Natural Language Processing model for the analysis of financial news sentiments.

The architectures and hyperparameters of the technical analysis engine are optimized and presented in the application in a more simplified way as user-friendly options.

The project serves as a basis for the democratization of machine learning technologies to the public as part of the discovery of investment opportunities. It paves the way for the extension and experimentation of new models, and for the future development of AutoML in the financial context.

Resumé

Ce projet de fin d'étude intitulé « Développement d'une application d'apprentissage machine pour la prédiction des mouvements des actions » est présenté dans le but d'obtenir le diplôme national d'ingénieurs en électronique industrielle à l'Ecole Nationale d'Ingénieurs de Sousse a été effectué au sein de l'entreprise Intelligencia IT & Financial Solutions.

Le projet vise à fournir aux différents investisseurs une application web d'investissement de tiers pour naviguer sur le marché boursier. Cela se fait par l'utilisation de l'apprentissage automatique et des technologies du web.

Comme l'approche d'analyse classique le suggérait, l'analyse technique et fondamentale ont été mises en œuvre de manière plus avancée en intégrant un model de deep learning comme moteur d'analyse technique et un model traitement de lagunage naturel (NLP) pour l'analyse des sentiments de l'actualité financière.

Les architectures et les hyperparamètres du modèle d'analyse technique est optimisé et présenté dans l'application de manière plus simplifiée sous forme d'options conviviales.

Le projet sert de base à la démocratisation des technologies d'apprentissage machine auprès du grand public dans le cadre de la découverte d'opportunités d'investissement. Il ouvre la voie à l'extension et à l'expérimentation de nouveaux modèles, et au développement futur d'AutoML dans le contexte financier.

Acknowledgements

Because a final study project is not done alone, it is with emotion that I want to thank after six month all those who near or far, supported me in the realization of this project.

First of all, I would like to thank Mr. Mongi HAMMAMI, my supervisor, for the trust and the time he has given to me, and for being able to re-motivate me in my difficult moments.

I warmly thank Mr. Ali DOUIK my academic supervisor, for the time he gave me despite his many loads, I thank him for his innovative ideas.

Dedicates

For my Dear respective Parents Ismail and Salha

You have been my constant source of encouragement and assistance.

I can never repay you for all the scarifies you made, for all your unconditional love and affection you have given me during my education, for all the prayers you have made for me, and for constantly believing in me. Without you I would not be the person I am today.

I hope I made you proud.

Table of Contents

List of figures

List of tables

Abbreviations

ADAM: ADAptive Moment estimation

AI: Artificial Intelligence

API: Application Programming Interface

EMA: Exponential Moving Average

LSTM: Long-Short Term Memory

MAE: Mean Absolute Error

MSE: Mean Squared Error

MSLE: Mean Squared Logarithmic Error

ML: Machine Learning

NLP: Natural Language Processing

NLU: Natural Language Understanding

OHLC: Open High Low Close

RNN: Recurrent Neural Network

RSI: Relative Strength Index

SMA: Simple Moving Average

UML: Unified Modeling Language

URL: Uniform Resource Locator

General introduction

Retail investors spend a lot of time finding investment opportunities. Wealthier investors may require professional financial advisory services, but for typical retail investors, the costs are prohibitive. As a result, retail investors must understand the market themselves and make decisions on their own. This makes investing in modern societies very stressful.

Unfortunately, humans are irrational by nature. Without quantitative, data-based models, decisions are influenced by cognitive biases or personal emotions, resulting in unnecessary losses.

Even if investors are cautious enough, most of them do not have the skills to handle the huge amount of data needed to make good judgments. Institutional investors rely on models supported by technology to avoid pitfalls, but retail investors do not have access and often find themselves behind the market.

"Artificial intelligence is to the financial market what fire was to cavemen". This is how one industry player described the impact of a revolutionary technology on a dormant industry.

The financial market today is mainly driven by an army of computers and quantitative statisticians who develop trading algorithms based on statistics and technical approaches. Artificial intelligence will provide investment opportunities based on the work of a neural network, building new trading patterns and trends based on previous experiences and market sentiment.

In this context, Intelligencia IT and Financial Solutions, as an innovative player in the financial industry, has proposed to design an investment tool that uses artificial intelligence to provide market estimation.

This report is a synthesis of the different stages of implementation of our final project. It includes three chapters:

- Organization of the project:
  - Chapter 1 provides the general framework for this project, in which we will present the profile of the company Intelligencia IT, some details about the requested work and the overall architecture of this project.
  - Chapter 2 focus on the state of the art of our project. We start by presenting the different notions of artificial intelligence, machine learning and deep learning, and then move on to its interest and applications in the financial field.
  - Chapter 3 presents the LSTM and NLP models and different technics used to structure the data.
  - Chapter 4 will constitute the last stage of our report and will correspond to the concretization of the work carried out by specifying at first the technological choices adopted and by exposing different screenshots of the system.
  - At the end, this report will be concluded with a general conclusion setting out a synthesis of our work and listing the prospects for improving our application.

Chapter 1: Working Environment and Objectives

- *Introduction*
- *Company profile*
- *Financial prerequisites*
- *Problem Framing*
- *Proposed solution*
- *Conclusion*

Introduction

This chapter contains the general context of this final studies project entitled "Development of a web app for stock trend prediction using deep learning".

In this part we will present the profile of the company Intelligencia IT, then we will highlight some financial terms required for the clarity of the project, and lastly, we present the proposed solution architecture.

1. Company profile

1.1 Intelligencia IT & Financial Solutions

Founded in 2009, the company was born thanks to the desire of its two founders to adopt a new approach in the development of stable and economic software packages, in order to benefit financial institutions of all sizes, in Tunisia and then throughout Africa.

“**Intelligencia IT and Financial Solutions**” is an innovative publisher that develops business software packages for the financial industry, and for all players in the asset management sector.

Its next-generation modular platform enables “**Intelligencia IT”** to adapt for the specific needs of each customer, ensuring global coverage of business processes. The speed of deployment, the ease of interfacing and the intuitiveness of the solutions offered, allow its customers a rapid grip and controlled growth prospects.

To maintain its technological lead, “**Intelligencia IT”** continues to invest heavily in the research and development of its technical frameworks, as well as in the functional evolution of its solutions [1].

1.2 Product and services

1.2.1 Integration services

“Intelligencia IT” also acts as an integrator of its own solutions. The company's executives are personally involved in the projects and pay attention to delivering a high level of service.

• From the installation of the solution to its production, Intelligencia IT teams can take over all the tasks of a project to implement a software solution.

• The software platform has been designed to facilitate the import stages and data migration.

1.2.2 Support and maintenance service

For all its customers, Intelligencia IT provides maintenance services tailored to their needs:

• Meet customer expectations

• Maintenance of all functional aspects, which need to be kept in operational conditions.

1.2.3 The products

MicroSolutions is a fully integrated range of solutions, developed from a single development framework. The core and technical platform of Intelligencia IT are the result of a thorough work in research and development to which most of the first 3 years of the company have been devoted.

The Framework is a key asset for customers, as it offers **Intelligencia IT**'s teams a very strong reactive in the face of the demands of proposed products. It delivers stable, robust, and easily customizable solutions for its customers.

- **MicroBroker** is the Front-Middle-Back Office stock exchange intermediation solution. It offers all the power, ease of use and functionality of a state-of-the-art tool, serving Stock Exchange brokers.
- **MicroHolders** is a solution dedicated to shareholder register holders wishing to maintain all the nominative positions related to one or more securities.
- **MicroMarket** offers all the possibilities of a platform for real-time trading.
- **MicroFix** is the Front Office order-passing solution in the MicroSolutions range. Entirely dedicated to the Buy-Side industry, it provides all the functions and services desired by a trading table or manager-negotiators.

2. Financial prerequisites

2.1 Introduction

Before proceeding further into the main subject, we need to examine some financial terms required for the good understanding of the project.

2.2 Stock market

A stock market, capital market or equity market is a group of buyers and sellers of stocks (also known as equities), which represent property rights over companies. Like any other market, it is dominated by the law of supply and demand.

2.3 Stocks

Stocks are capital investments that represent the legal ownership of a company. You become a shareholder in a company when you buy stocks.

The following graph displays the Amazon's historical stock price (USD) evolutions using Candlestick's chart and volume’s bars that we will discover in the next paragraph.

Figure 1 Candlestick financial chart

2.4 Charts

As they say, “A picture is worth a thousand words”, charts are the picture that forms the backbone on which the basis of any analytical endeavor rests.

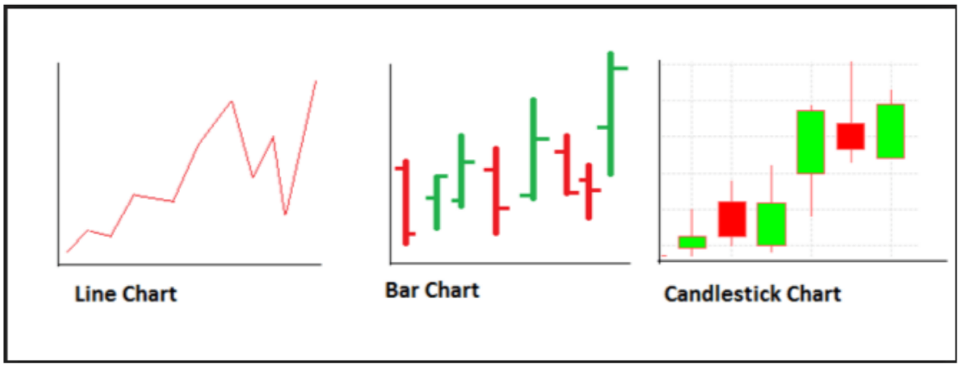


Figure 2 Types of charts in technical analysis

Candlestick charts are by far the most popular form of charting technique used by technical traders worldwide. Finding their origins in the 18th century, this charting technique was developed by a Japanese rice trader. It provides more detailed and accurate information on price movements compared to bar charts. They provide a graphic representation of supply and demand during each period of stock price.

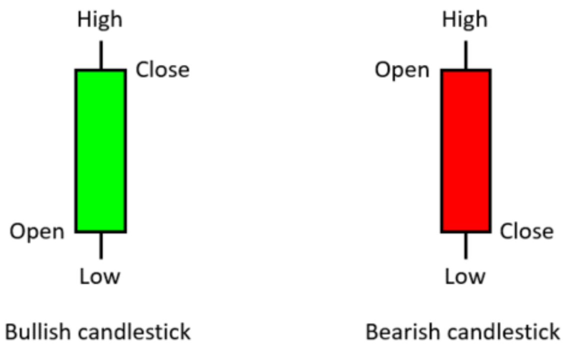


Figure 3 OHLC Candlestick

Open: the price of a stock at the beginning of a trading day on a stock market.

High: the highest price the stock reached during a time frame (month, day, hour…).

Low: the lowest price the stock reached during a time frame.

Close: the price of a stock at the ending of a time frame.

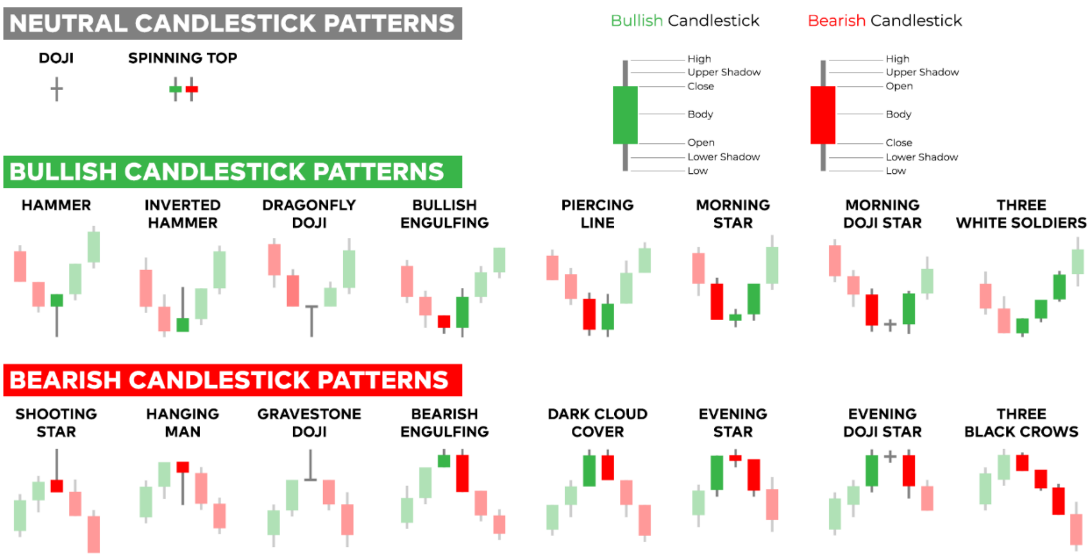


Figure 4 Candlestick patterns

2.5 Technical indicators

Indicators are additions or overlays on the graph that provide additional information through mathematical calculations of price and volume. They also tell you where the price is likely to go next.

There are four main types of indicators:

- **Trend indicators**

Trend indicators gives an overview about where the market is moving. They are sometimes called oscillators because they tend to move between high and low values like a wave. The trend indicators we will discuss are Moving Average Convergence Divergence (MACD), Exponential Moving Average (EMA) and Simple Moving Average (SMA).

$$SMA_t = \frac{1}{N}\sum_{i=1}^{N} P_{t_i}$$

$$MACD_t = EMA_t^M - EMA_t^N$$

Here EMA_t is defined recursively:

$$EMA_t^N = \frac{2}{N}\left(P_t - EMA_{t-1}^N\right) + EMA_{t-1}^N$$

where $EMA_0 = 0$

Pt: Stock price at time t

N, M: Time frame (For MACD the most common is M=12 and N=26)

❖ **Momentum Indicators**

Momentum signals indicates the strength of the trend and can also tell you if a reversal is going to happen. They can be useful for spotting price ups and downs. Taking the RSI indicators bellow it measures the speed and change of price movements. RSI oscillates between zero and 100.

The RSI signal is defined as:

$$RSI_t = 100 - \frac{100}{1+RS_t}$$

where

$$RS_t = \frac{1}{N}\left(\sum_{i=1}^{N} Gain_{t-i}\right) / \left(\sum_{i=1}^{N} Loss_{t-i}\right)$$

and

$$\text{Gain}_t = max\,(0, P_t - P_{t-1})$$

$$\text{Loss}_t = max\,(0, P_{t-1} - P_t)$$

where Pt: Stock price at time t

N: Time frame (the most common for RSI is N=14)

❖ **Volume indicators**

Volume indicators inform about how volume changes over time, how many units are purchased and sold over time. This is useful because when the price changes, the volume gives an indication of the strength of the movement. Increasing movements over a high volume are more likely to continue than those on a low volume.

❖ **Volatility indicators**

Volatility indicators indicates how much the price changes over a time period. Volatility is a very important part of the market.

2.6 Fundamental analysis

This approach involves analyzing fundamental attributes to identify potentially promising companies. This includes features such as financial results, the company's asset position, balance sheet and inventory and growth forecasts. It is very important to understand that this type of analysis is not static. Recently released financial information, business announcements and other news can influence a company's fundamental outlook. Fundamental analysis requires expertise in a particular area and is often conducted by professional analysts. The investments they recommend are the following regularly published and updated.

2.7 Technical analysis

Unlike fundamental analysis, technical analysis does not attempt to obtain an in-depth overview of a company's activity. It assumes that the available public information does not offer a commercial advantage. Rather, it focuses on studying the historical price of a company's shares and identifying patterns in the graph by exploiting technical indicators. The intention is to recognize trends in advance and capitalize on them.

3. Problem framing

Financial market modeling and forecasting has been an attractive topic for academics and researchers from various academic fields. Financial market is an abstract concept where financial products such as stocks and precious metal transactions are between buyers and sellers.

In the current scenario of the world of financial markets, particularly the stock market, predicting the trend or stock's price using machine learning techniques and artificial neural networks is the most interesting problem to study.

Financial forecasts are an example of a signal processing problem that is difficult due to high noise, small sample size, non-stationarity, and non-linearity. Noisy features mean the in-complete information gap between the price and volume of past stock market transactions with a future price.

The stock market is sensitive to the political and macroeconomic environment. And due to the non-linearity, volatility, and complex nature of the stock market, it is quite difficult to predict it.

4. Outcomes

As an outcome, the proposed work aims to assist investors in making good investment decisions to earn profits. The suggested approach will generate a stock trend prediction based on financial news and historical data that could be employed as a trend-based analysis to decide whether to hold, buy, or sell.

5. Prediction Techniques

The table below introduces a comparative study of predictions techniques.

Table 1 Prediction techniques

Criteria	Technical Analysis	Fundamental Analysis	Traditional Time Series Analysis	Machine Learning Techniques
Data Used	Price, volume,highest, lowest prices	Growth, dividend payment, sales level, interest rates, tax rates etc.	Historical data	Set of sample data
Learning methods	Extraction of trading rules from charts	Simple trading rules extraction	Regression analysis on attributes is used	Inductive learning is used
Type of Tools	Charts are used	Trading rules	Simple Regression and Multivariate analysis used for time series.	Nearest neighbor and Neural Networks are used
Implementation	Daily basis prediction	Long –term basis prediction	Long –term basis prediction	Daily basis prediction

6. Proposed solution

The proposed solution aims to develop a Rest API that harvest a stock historical data and scrap the web for financial news to build a strong prediction and provide the user with sentiments based on latest news. The API can be consumed afterward by any web or mobile app, for demonstration purposes we will use an Angular 9 based web app to consume the API and plot the predictions on the stock chart.

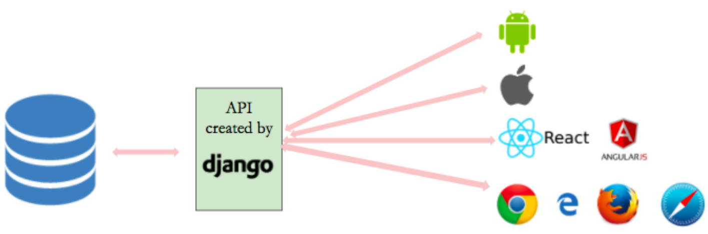


Figure 5 Rest API architecture

7. Conclusion

As we introduced in this part, the company profile, general aspects about financial terms and concepts and an overview about the proposed solution, the next chapter highlights the state of art part and the different methods and frameworks implemented during the development of the web application.

Chapter 2: Problem analysis & State of the art

- *Introduction*
- *Problem Analysis*
- *State of the art*
- *Conclusion*

Introduction

In this chapter we move forward to analyzing the problem in-depth and find the most suitable approach to implement, then we will focus on some state-of-the-art artificial intelligence techniques.

1. Problem analysis

The purpose of this project is to build an API capable of the following tasks:

1. **Collecting fundamental and technical data from the internet**
   The system should be able to crawl specific websites to extract fundamental data like news articles and analyst recommendations. Furthermore, it should be able to collect technical data in the form of historical stock prices.
2. **Tuning the model hyperparameters and train a new model**
   The system should offer ways tune the model hyperparameters via a user-friendly interface and train different models.
3. **Evaluating and visualizing stock trend predictions and the financial news sentiment analysis.**
   The application should be able to visualize the financial predictions. This allows a comparison to be made between different models and approaches.

1.1 Overall Control Flow Diagram

A control flow diagram helps us understand the detail of a process. It shows us where control starts and ends and where it may branch off in another direction, given certain situations.

- Alpha Vantage Inc. is a leading provider of free APIs for real-time and historical data on stocks, physical currencies, and digital/crypto currencies.
- Sentiment analysis is the contextual mining of text which identifies and extracts subjective information in source materials.
- After processing the gathered data (stock historical price and financial news) a stock trend prediction and financial news sentiment will be generated and displayed.

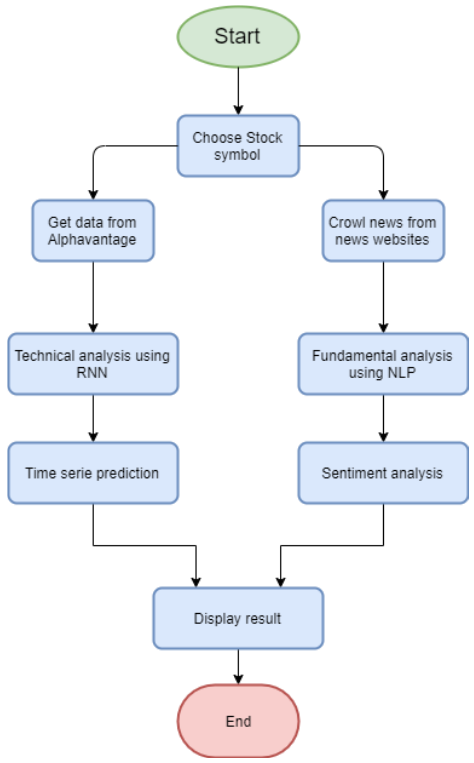


Figure 6 Control flow diagram

1.2 Proposed Architecture

A convenient way to deploy machine learning models is to use an API. Offering a machine learning (ML) solution through an API allows the person using it to focus on results, while giving ML developers total control and ease of maintenance of the model behind. This effective method of implementing ML is widely applied, even by companies with large AI divisions such as Google (for example, their Vision API) and Amazon.

Another factor that probably plays an important role in the large scale adoption of this technology is the presence of exceptional Python libraries such as Scikit-learn, Pandas and TensorFlow, which makes it easier for ML developers to deliver high quality solutions.

Django is an open source web framework, entirely written in Python. It is easy to use, stable and integrates with all Python libraries. It has become one of the top four web executives, used on the websites of companies such as Instagram and Disqus. By adding the Django REST Framework, you can have a FULLY Python-based RESTful API for your machine learning solution quickly.

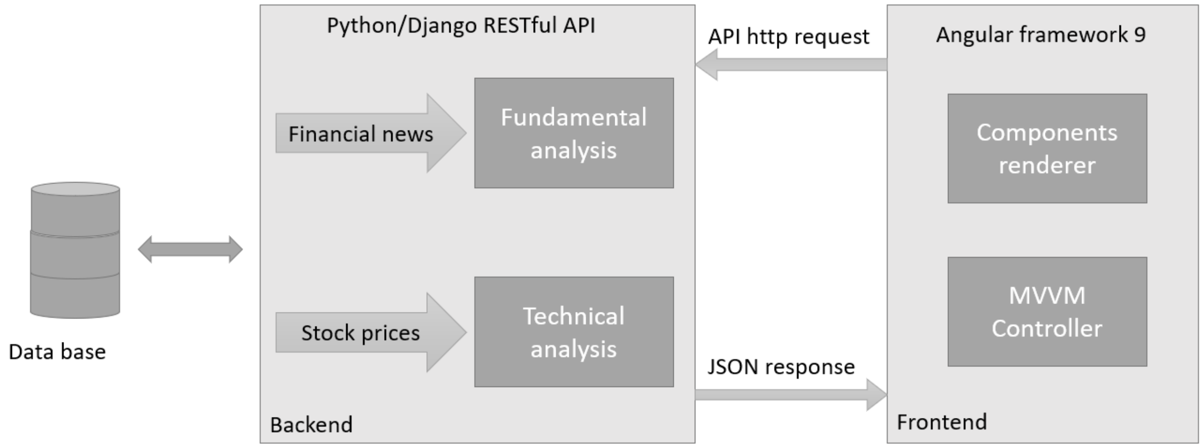


Figure 7 Application architecture

Giving the previous architecture we can divide the project into four parts:

- **Time series forecasting**
- **Sentiment analysis**
- **Back-end development**
- **Front-end development**

2. State of the art

The state of the art aims to focus on commercial and open source tools that meet our needs for modeling and developing the application.

2.1 Web frameworks and tools

2.1.1 Tools & components

- **Server**- It is the place where backend is stored and run. These are high-powered computers that provides resources which the backend needs i.e. file storage space, processing power, security and encryption, databases, and other web services.
- **Database**- It is the brain of any Website that makes it dynamic. If you are searching for anything on a website be it a profile on any social media or any product on E-commerce website it is the role of a database to take the query and fetch the required data to the user.
- **Middleware**- It is any software (Server-side) that facilitates the connection between the frontend and the backend. It acts as a medium that takes requests from the user and provides it to the backend and then facilitate the user with a response given by the backend.

- **Programming Languages and frameworks-** There is a variety of languages available in which the backend can be coded however, the language is chosen based upon the usage because of the difference in their performance, memory usage, compatibility etc.
- Most sites are built in PHP but nowadays NodeJS is gaining momentum and getting popular day by day. One may also choose Python (Django, flask) for better processing power.

2.1.2 Python

Python is easy to use, powerful, and versatile, making it a great choice for beginners and experts alike [2]. Python is the first programming language and the favorite for AI developers for multiple reasons:

- Extensive selection of libraries and frameworks

  One of the aspects that makes Python such a popular choice in general, is its abundance of libraries and frameworks that facilitate coding and save development time. Machine learning and deep learning are exceptionally well catered for.

- The simplicity

  Python is known for its concise, readable code, and is almost unrivaled when it comes to ease of use and simplicity, particularly for new developers. This has several advantages for machine learning and deep learning.

- Abundance of support

  Python is an open-source programming language and is supported by a lot of resources and high-quality documentation. It also boasts a large and active community of developers willing to provide advice and assistance through all stages of the development process.

2.1.3 Keras

Keras [3] is a high-level neural networks library which is running on the top of TensorFlow. Using Keras in deep learning allows for easy and fast prototyping as well as running seamlessly on CPU and GPU. This framework is written in Python code which is easy to debug and allows ease for extensibility. The main advantages of Keras are described below:

User-Friendly: Keras has a simple, consistent interface optimized for common use cases which provides clear and actionable feedback for user errors.

Modular and Composable: Keras models are made by connecting configurable building blocks together, with few restrictions.

Easy to Extend: With the help of Keras, you can easily write custom building blocks for new ideas and research.

Easy to Use: Keras offers consistent & simple APIs which helps in minimizing the number of user actions required for common use cases, also it provides clear and actionable feedback upon user error.

2.1.4 Tensorflow

TensorFlow [4] is an open source platform for machine learning. It is a comprehensive and flexible ecosystem of tools, libraries and other resources that provide workflows with high-level APIs. The framework offers various levels of concepts for you to choose the one you need to build and deploy machine learning models. For instance, if you need to do some large machine learning tasks, you can use the Distribution Strategy API in order to perform distributed hardware configurations and if you need a full production machine learning pipeline, you can simply use TensorFlow Extended (TFX). Some of the salient features are described below:

Easy Model Building: TensorFlow offers multiple levels of abstraction to build and train models.

Robust ML Production Anywhere: TensorFlow lets you train and deploy your model easily, no matter what language or platform you use.

Powerful Experimentation for Research: TensorFlow gives you the flexibility and control with features like the Keras Functional API and Model Subclassing API for creation of complex topologies.

2.1.5 Django

Django [5] is a free and open source web framework written in Python programming language and used by millions of programmers each year. Its popularity is due to its user-friendliness, both for beginners and experienced programmers: Django is robust enough to be used by the world's leading websites - Instagram, Pinterest, Bitbucket, but also flexible enough to be a good choice for young start-ups and prototyping personal projects.

Django REST Framework [6]

An excellent and common way to implement an API with Django is to use Django REST framework, a Django package filled with powerful but flexible tools to build a REST API. It allows you to serialize your data (by translating your model objects into json objects), quickly write and test end views/points, add authentication, and much more.

2.1.6 Restfull API

A RESTful API is an application program interface (API) that uses HTTP queries to obtain, send, post, and delete data.

An API for a website is a code that allows two software to communicate together. The API indicates the right way for a developer to write a program requesting services from an operating system or other applications.

REST technology is generally preferred to the more robust SOAP (Simple Object Access Protocol) technology because REST uses less bandwidth, making it more suitable for efficient Internet use.

2.1.7 Json

JSON means JavaScript Object Notation is a lightweight format for data storage and transport. It is often used when data is sent from a server to a web page, it describes itself and it is easy to understand.

A JSON object:

{"name": "Bilel HADDAJI"

"age": 26,

"company": "Intelligencia IT"}

2.2 Artificial intelligence(AI)

To understand AI, we can first examine the context of the history and evolution of information technology (IT). The progress of IT is part of a 10 year generational change. We are just coming to the end of the era of mobile (universal computing device), which replaced the era of the

Internet (democratic connectivity) that followed the era of the PC (decentralized computing). Some of the current AI technologies have been in use for a very long time, such as expert systems (1970s) or recommendation engines (2000s).

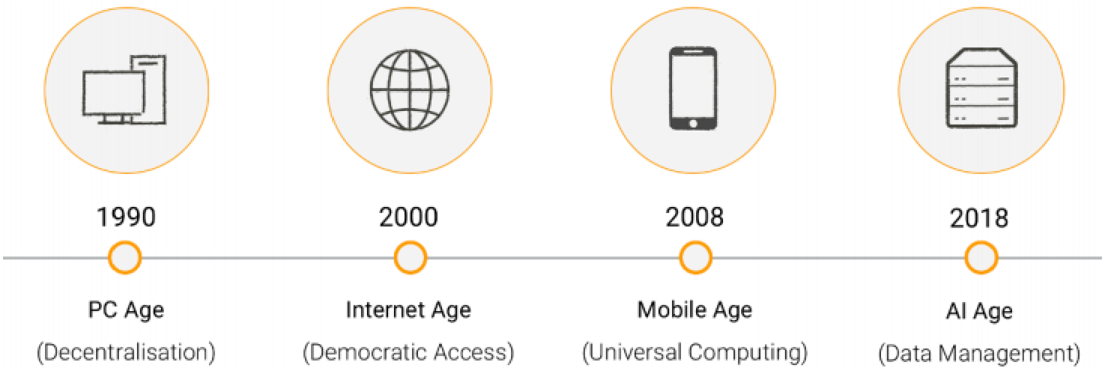


Figure 8 Evolution of Information Technology

It is preferable to view the new era of AI as a paradigm shift from "explicit programming" to "implicit programming." [7]

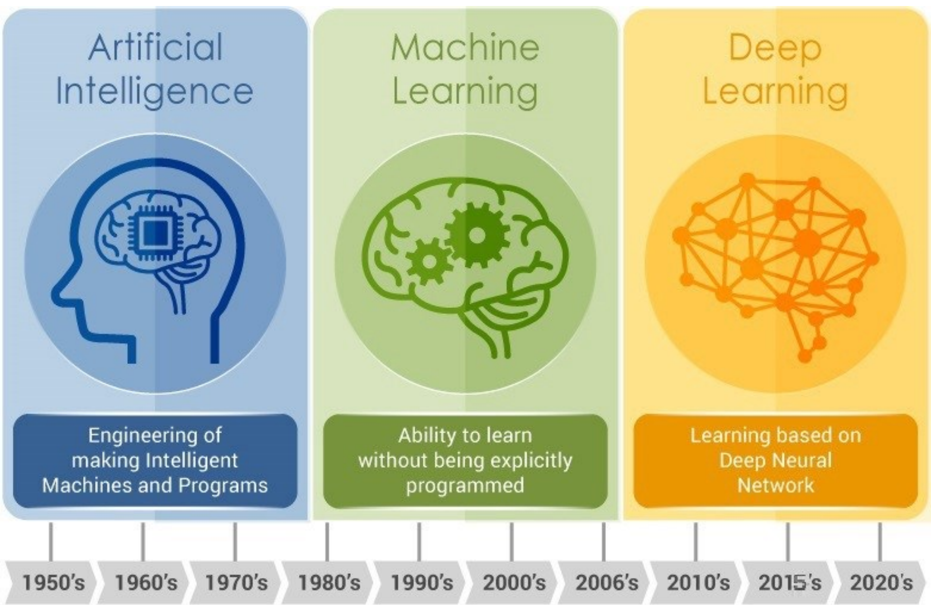


Figure 9 AI time evolution

2.2.1 Problems and methodologies

AI/ML problems must be divided into three distinct parts: the type of problem, the type of algorithm and the training method.

Understanding the type of problem that needs to be addressed is the first step in any AI/ML project. Questions focus on the amount, the availability, and the generation of additional data during the preparation of the AI/ML training.

1. **Type of the problem**: Machine Learning is concerned with what types of problems you are solving. The broad classes of problems are:

• **Regression**: Predicting a continuous variable - e.g. when you want to predict the value of a house based on the number of rooms and distance from a train station

• **Classification**: Predicting a variable with finite (or discreet) possible values - e.g. when you want, in a photo, to guess if it is human, dog, cat, house, etc. (called a class)

• **Clustering**: Grouping data - e.g. you want to count the number of people in a photo.

• **Collaborative filtering**: Filling gaps - e.g. in a recommendation for movies (or products), the ability to 'fill the gaps' so you can tell the users which movies to watch.

• **Reinforced learning (RL):** is learning by interacting with an environment. An RL agent learns from the consequences of its actions, rather than being explicitly taught, and selects its actions based on its past experiences (exploitation) and by new choices (exploration).

2. **Types of algorithms:** Once the type of problem is identified, you need to choose the best ML algorithm. For example, a neural network is only one of the types of algorithm class which at can be used. NNs are themselves sub-classified into many architectures, including:

• **Convolutional**: These neural networks apply learnt memory of historical results to processing new data.

• **Recurrent**: The output of each layer directly feeds into the input to make predictions on future data.

• **Self-organized**: An iterative network that self-organizes into clusters classifying each cluster according to different priorities.

3. **Training:** Finally, the Training is the process of self-learning. Training a model simply means learning (determining) good values for all the weights and the bias from labelled examples. In supervised learning, a machine learning algorithm builds a model by examining many examples and attempting to find a model that minimizes loss; this process is called empirical risk minimization.

• **Supervised**: Supervised learning is where we have input variables (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output. The algorithm is formed by comparing its calculated results with the predefined values. Using various methods such as regression or classification, the machine learning algorithm can predict new data based on its labeled database.

• **Unsupervised**: Unsupervised learning is where we only have input data (X) and no corresponding output variables. The algorithm is responsible for finding patterns and structures hidden in the data, but there is no "evaluation" of the results. This method is the most common for anomalies detection, grouping, density estimation.

• **Semi-supervised**: Problems where you have a large amount of input data (X) and only some of the data is labeled (Y) are called semi-supervised learning problems. Labeled data is difficult to produce and verify, which is why partially supervised learning uses a small set of labeled data, so learning from that set can then be applied to non-labeled data. This is typical for classification, prediction, and regression. In this case, it is assumed that part of the structure of the labeled set is shared with the unlabeled set, which ensures continuity, grouping and multiplicity of assumptions.

• **Reinforced learning**: This method is inspired by behavioral psychology, using cumulative reward methodology. It uses a Trial & Error concept, where the positive reward state can be achieved in the fastest possible time. For this method, an environment with an agreed set of actions and the desired outcome is defined. The algorithm (agent) is allowed to manipulate the actions and the reward is then fed back to the agent. This process is repeated until the agent has found the best configuration of actions to generate the reward.

2.2.2 Recurrent Neural Networks (RNNs)

Recurrent neural networks (RNNs) are a type of neural network in which the outputs of the previous stage serve as an entry into the current stage. In conventional neural networks, all entries and outputs are independent of each other, but in cases where it is necessary to predict the next word of a sentence, the previous words are necessary and therefore it is necessary to remember the previous words. This is how the RNN was born, which solved this problem with a hidden layer. The main and most important feature of RNN is the hidden state, which memorizes certain information about a sequence.

2.2.2.1 Characteristics

You can think of RNNs as a mechanism to hold memory, where the memory is contained within the hidden layer.

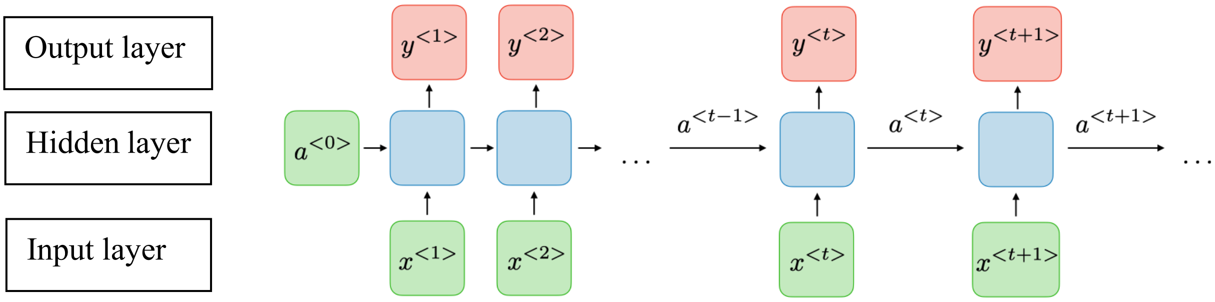


Figure 10 RNN's structure

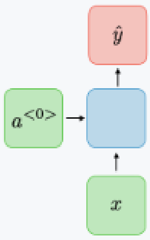
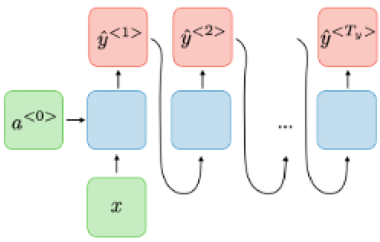
Table 2 Advantages & disadvantages of RNN

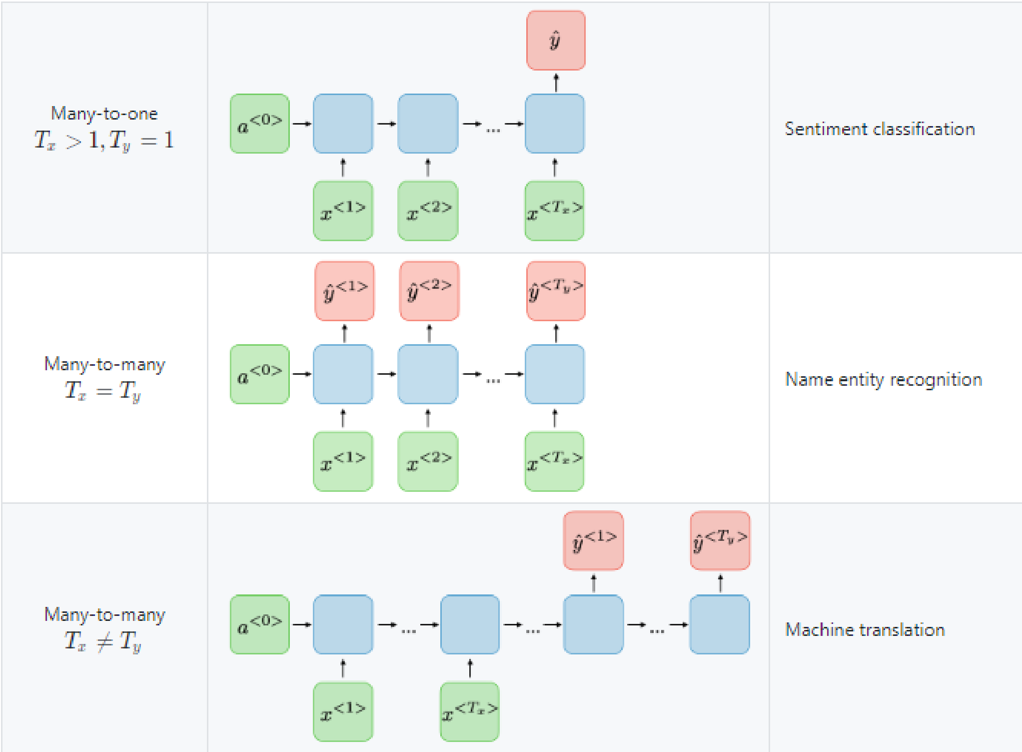
Advantages	Disadvantages
- It is possible to process entries of any length - The size of the model does not increase with the size of the entrance - The calculation considers historical information - Weights are shared over time	- The calculation is slow - Difficulty accessing information that dates back a long time

2.2.2.2 Applications

There can be a different architecture of RNN. Some of the possible ways are as follows.

Table 3 RNN architectures and examples

Type of RNN	Illustration	Example
One-to-one $T_x = T_y = 1$		Traditional neural network
One-to-many $T_x = 1, T_y > 1$		Music generation



2.2.3 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) [8] networks are a modified version of recurrent neural networks, which makes it easier to remember past data in memory. The vanishing gradient problem of RNN is resolved here. LSTM is well suited to classify, process and predict time series given time lags of unknown duration. It trains the model by using back propagation.

Basic concept:

The basic concept of LSTM is the state of the cell, and its various gates. The cellular state acts as a transport highway that transfers related information throughout the sequence series. It can be considered the "memory" of the network. The cellular state, in theory, can carry relevant information throughout the processing of the sequence. Thus, even information from the first steps of time can be found in the following steps, thus reducing the effects of short-term memory. As the state of the cell continues its journey, information is added or removed from the cell state through gates. Gates are different neural networks that decide the information allowed in the state of the cell. Gates can learn what information is relevant to keep or forget during training.

3. The structure of an LSTM

An LSTM has a similar control flow as a recurrent neural network. It processes data passing on information as it propagates forward. The differences are the operations within the LSTM's cells.

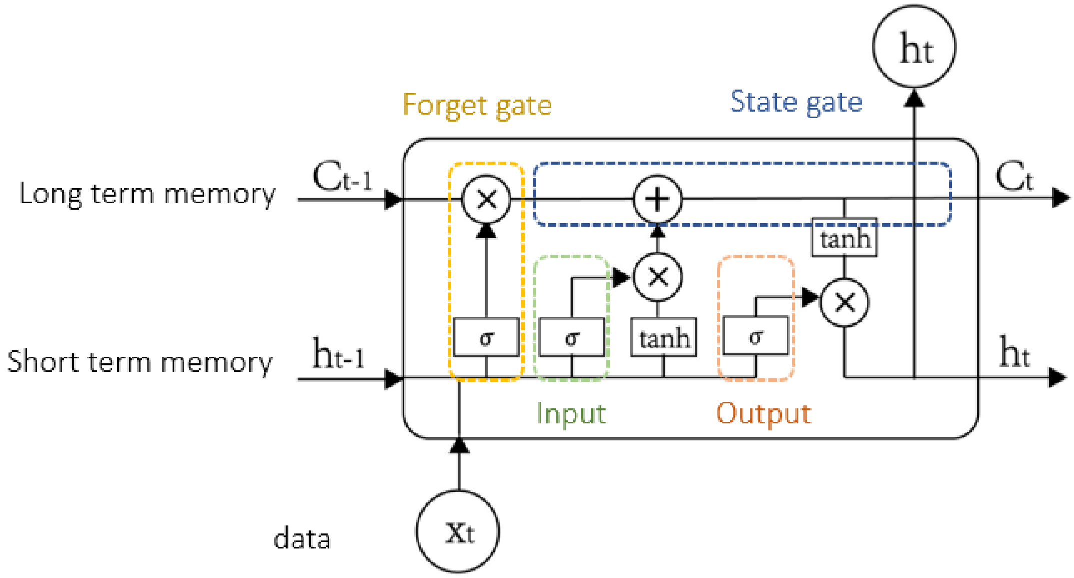


Figure 11 LSTM unit structure

An LSTM unit has four "gates": forget, State, Input and Output.

It also has three inputs: long-term memory, short-term memory, and data input.

Step 1: When the 3 inputs enter the LSTM, they either go into the **forget** gate or into the **input** gate.

Long-term information goes into the **forget** gate, where some is forgotten (the irrelevant parts).

Short-term information and new data go into the **input** gate. Where relevant information will be learned.

Step 2: information that passes through the **forget** gate (relevant information) and the information that passes through the **input** gate (learned information) will go to the **state** gate (which constitutes the new long-term memory) and to the **output** gate (which updates the short-term memory - the result of the network).

3.1.1 Natural Language Processing (NLP)

Natural language processing, usually abbreviated as NLP, is a branch of artificial intelligence that deals with the interaction between computers and humans using natural language.

The ultimate goal of the NLP is to read, decipher, understand and make sense of human languages in a way that is useful.

Most NLP techniques rely on machine learning to draw a sense of human languages.

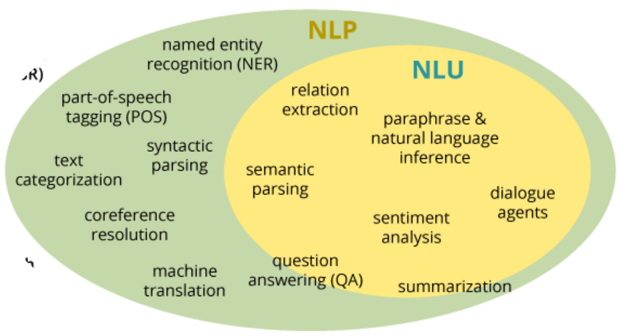


Figure 12 NLP & NLU fields

3.1.1.1 Natural Language Understanding (NLU)

Considered as a subtopic of NLP, NLU helps computers understand and interpret human language by breaking down elements of the message. While voice recognition captures the spoken language in real time, transcribes it and returns the text. The NLU goes beyond recognition to determine the user's intent. In the NLU, machine learning models improve over time as they learn to recognize syntax, context, language patterns, unique definitions, feeling and intent.

Commercial applications often rely on the NLU to understand what people are saying in the spoken and written language. This data helps virtual assistants and other applications determine a user's intent and direct them to the right task.

3.1.1.2 Sentiment analysis

The analysis of feelings, or the extraction of opinions, is a sub-domain of natural language processing (NLP) that attempts to identify and extract opinions in a given text. The objective of the analysis of feelings is to measure the attitude, feelings, evaluations, orientations, and emotions of a speaker/writer based on the computer processing of subjectivity in a text.

4. Conclusion

As stated above, Stock price prediction is a fertile field for machine learning thanks to the big amount of data available for users. After analyzing and dividing the problem into parts to ease the development approach, in the next chapter we aim to analyze each part individually.

Chapter 3: Architecture and ML Modeling

- *Introduction*
- *In-depth architecture*
- *LSTM Model*
- *NLP Model*
- *API requirement and specification*
- *Conclusion*

Introduction

Requirements and specification analysis are the first steps in application development. It is important to understand the project before proceeding to identifying, organizational and technical issues.

This chapter will focus on the specification requirements of each part to meet the objectives stated in the previous chapter,

1. Long-Short Term Memory Model

This section focuses on using a Deep LSTM Neural Network architecture to implement multidimensional time series forecasting using Keras and Tensorflow, specifically on stock market datasets to provide momentum indicators of the stock price.

1.1 Model overview

the purpose of using the LSTM model is its ability to conduct a good memory state by resolving the vanishing gradient problem encountered in the RNNs architectures. Consequently, it is more suitable for time series forecasting.

In this section, the LSTM model is operating as the Technical analysis engine for the app. And as discussed earlier, Technical indicators provide a viewpoint on the strength and direction of the price action of the stock. Thus, combining historical data with technical indicators will give the LSTM model the ability to extract patterns and relations from the data flow.

For example, here we are using all the OHLC and three indicators: SMA, RSI, and MACD. And we need to mention that the data should be sorted by date. Without taking in consideration holidays and weekend when the stock market is closed

date	open	high	low	close	volume	SMA	RSI_14D	MACD
6/11/2020	349.31	351.06	335.48	335.9	49567675	328.7183	42.63057	11.75635
6/10/2020	347.9	354.77	346.09	352.84	41662938	327.1208	68.32823	13.47461
6/9/2020	332.14	345.61	332.01	343.99	36928091	324.2917	66.81128	13.74694
6/8/2020	330.25	333.6	327.32	333.46	23913634	322.03	62.13346	14.07581
6/5/2020	323.35	331.75	323.23	331.5	34312550	320.8442	66.11537	13.91309
6/4/2020	324.39	325.62	320.78	322.32	21890091	319.3142	61.51226	13.61482

Figure 13 Data structure

1.2 Data preprocessing

1.2.1 Input shape and data flow

Even if we understand LSTMs theoretically, still many of us are confused about its input and output shapes while fitting the data to the network.

First, we need to understand the Input and its shape in LSTM Keras. The input data to LSTM looks like the following diagram.

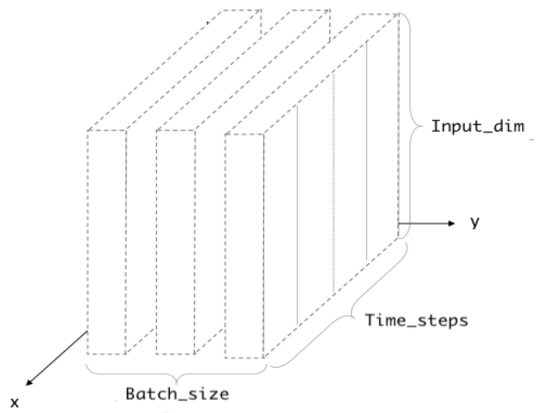


Figure 14 LSTM input shape

LSTMs consume input in format [batch_size, time_steps, Features]; a 3- dimensional array.

- **Batch Size** says how many samples of input do you want your Neural Net to see before updating the weights. As it turns out using very small batch size reduces the speed of training and on the other hand, using too big batch size (like the whole dataset) reduces the model's ability to generalize to different data and it also consumes more memory. But it takes fewer steps to find the minima for your objective function.
- **Time Steps** define how many units back in time you want your network to see.
- **Features** (input dimension) are the number of attributes used to represent each time step. Consider the data structure example above, we used Open, High, Low, Close, Volume, SMA, RSI, and MACD. Then feature size here is 8.

1.2.2 *Data normalization*

the close price is a constantly moving absolute price of the stock market. This means that if we tried to train the model on this without normalizing it, it would never converge.

To combat this we will take each n-sized window of training/testing data and normalize each one to reflect percentage changes from the start of that window (so the data at point i=0 will always be 0). We will use the following equations to normalize and subsequently de-normalize at the end of the prediction process to get a real-world number out of the prediction:

n = normalized list [window] of price changes

p = raw list [window] of daily return prices

Normalization: $n_i = \left(\frac{P_i}{P_0}\right) - 1$

De-Normalization: $P_i = P_0(n_i + 1)$

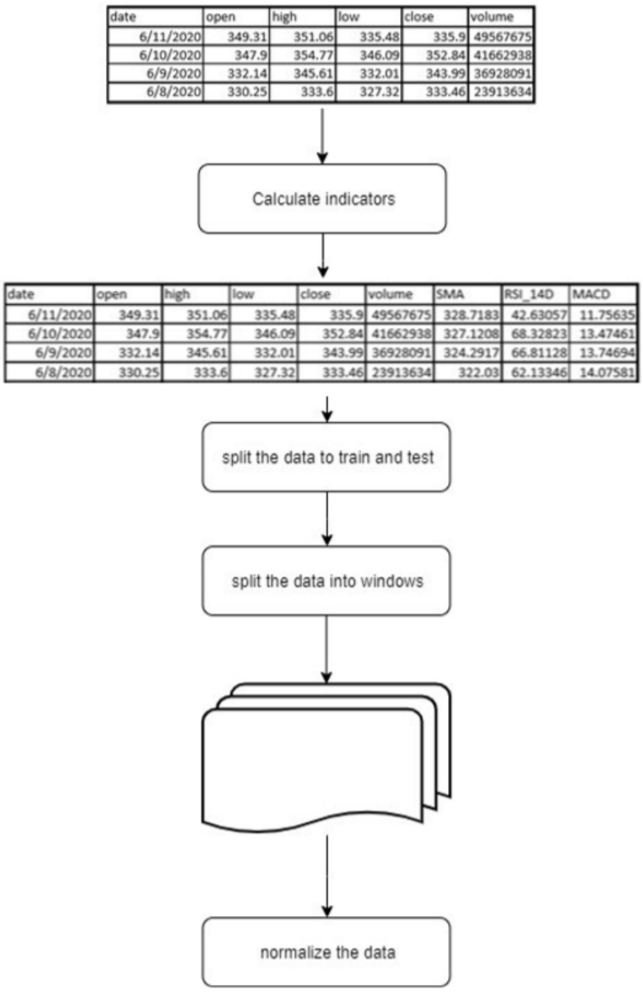


Figure 15 Data preprocessing logic

1.3 Model architecture

Keras layers are the primary building block of Keras models. Each layer receives input information, do some computation, and finally output the transformed information. The output of one layer will flow into the next layer as its input.

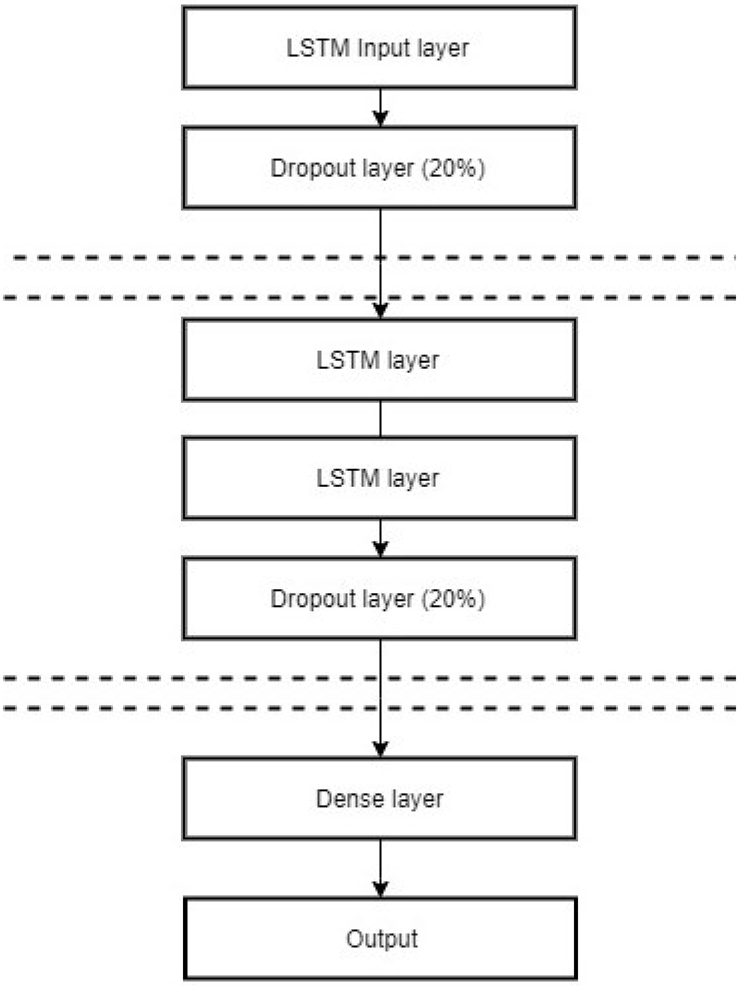


Figure 16 LSTM model architecture

- **The input layer**

The LSTM input layer was previously encountered when we explained the input shape.

- **The Dropout layer**

Dropout is a technique that prevents overfitting and provides a way of approximately combining exponentially many different neural network architectures efficiently. The term “dropout” refers to dropping out units (hidden and visible) in a neural network. By dropping a unit out, we mean temporarily removing it from the network, along with all its incoming and outgoing connections, as shown in Figure 17.

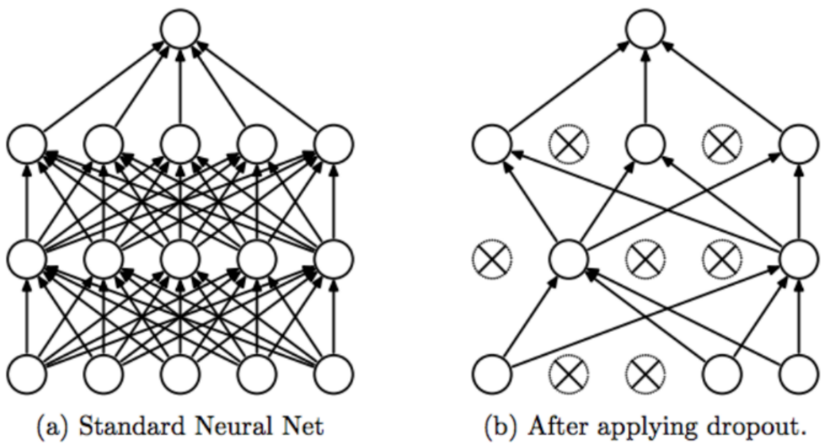


Figure 17 The dropout layer influence

- **The Dense layer**

A dense layer is just a regular layer of neurons in a neural network. Each neuron receives input from all the neurons in the previous layer, thus densely connected. The layer has a weight matrix W, a bias vector b, and the activations of previous layer.

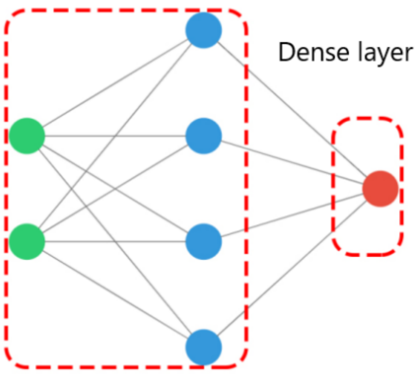


Figure 18 The dense layer

1.4 Experiments and Results

We have tried to predict the next day stock price, each point is predicted based on the previous true data. and we got the result in Figure.

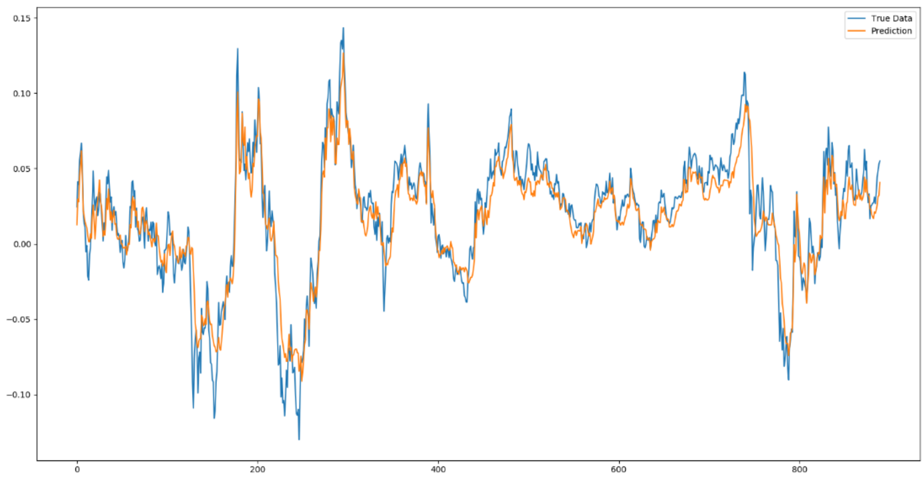


Figure 19 Point-by-point prediction result

As much as it seems promissing but it is rather deceptive, the model is lagging which makes the prediction wrong, if we zoom closer the patern is similar but if we take as example the point 6 on the X axis the predicted price is the same as the current day. This result is far from being acceptable.

Figure 20 Zoomed point-by-point prediction result

Moving on to the full sequence prediction it seems like this proves to be the least useful prediction for this type of time series (at least trained on this model with these hyperparameters). We can see a slight bump on the start of the prediction where the model followed a momentum of some sorts, however very quickly we can see the model decided that the most optimal pattern was to converge onto some equilibrium of the time series.

Figure 21 Full sequence prediction result

Finally, we have composed a third type of prediction for this model, a multi-sequence prediction. This is a blend of the full sequence prediction and point-by-point prediction, in the sense that it still initializes the testing window with test data, predicts the next point over that and makes a new window with the next point. However, once it reaches a point where the input window is made up fully of past predictions it stops, shifts forward one full window length, resets the window with the true test data, and starts the process again. This gives multiple trend-line like predictions over the test data to be able to analyze how well the model can pick up future momentum trends.

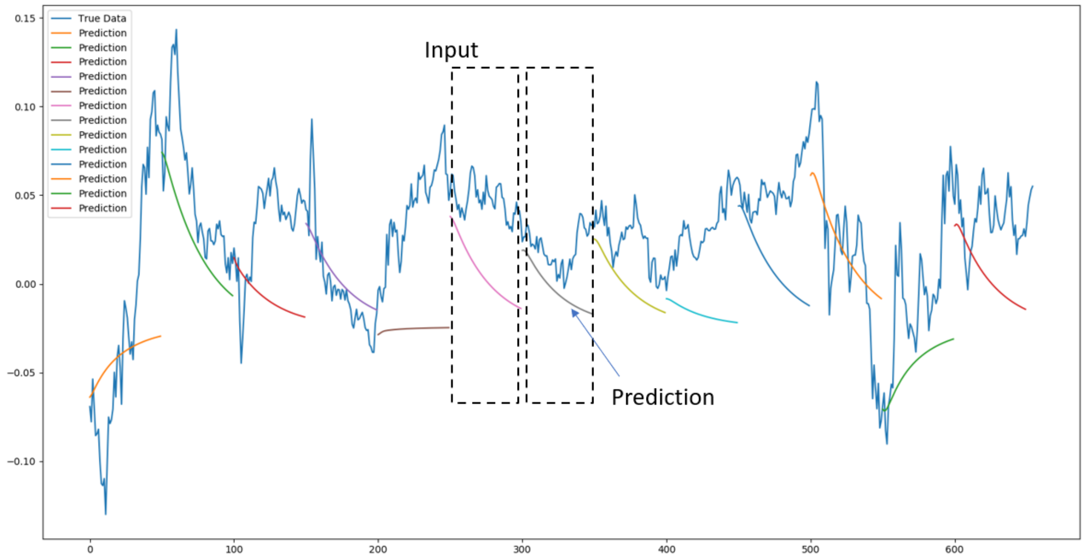


Figure 22 Multi-sequence prediction result

2. Natural language processing model

The idea from implementing an NLP model in the application is to help users with fundamental analysis by crawling news from financial websites using a web scraper and make a sentiment analysis for each one trying to assign a score: 1 good, 0 neutral, -1 bad.

2.1 VADER sentiment analysis

VADER (**V**alence **A**ware **D**ictionary and s**E**ntiment **R**easoner) [9] is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. it uses a combination of A sentiment lexicon with a list of lexical features (e.g., words) which are generally labelled according to their semantic orientation as either positive or negative.

VADER has been found to be quite successful when dealing with social media texts, NY Times editorials, movie reviews, and product reviews. This is because VADER not only tells about the Positivity and Negativity score but also tells us about how positive or negative a sentiment is.

2.1.1 Advantages

VADER has a lot of advantages over traditional methods of Sentiment Analysis, including:

- It works exceedingly well on social media type text, yet readily generalizes to multiple domains.
- It does not require any training data but is constructed from a generalizable, valence-based, human-curated gold standard sentiment lexicon.
- It is fast enough to be used online with streaming data.

2.1.2 How it works

VADER makes use of certain rules to incorporate the impact of each sub-text on the perceived intensity of sentiment in sentence-level text. These rules are called Heuristics. We present most significant of them.

- **Punctuation**, namely the exclamation point (!), increases the magnitude of the intensity without modifying the semantic orientation. For example: “The weather is hot!!!” is more intense than “The weather is hot.”
- **Capitalization**, specifically using ALL-CAPS to emphasize a sentiment-relevant word in the presence of other non-capitalized words, increases the magnitude of the sentiment

intensity without affecting the semantic orientation. For example: “The weather is HOT.” conveys more intensity than “The weather is hot.”

- **Degree modifiers** (also called intensifiers, booster words, or degree adverbs) impact sentiment intensity by either increasing or decreasing the intensity. For example: “The weather is extremely hot.” is more intense than “The weather is hot.”, whereas “The weather is slightly hot.” reduces the intensity.
- **Polarity shift due to Conjunctions**, the contrastive conjunction “but” signals a shift in sentiment polarity, with the sentiment of the text following the conjunction being dominant. For example: “The weather is hot, but it is bearable.” has mixed sentiment, with the latter half dictating the overall rating.
- **Catching Polarity Negation**, by examining the contiguous sequence of 3 items preceding a sentiment-laden lexical feature, we catch nearly 90% of cases where negation flips the polarity of the text. For example, a negated sentence would be “The weather isn't really that hot.”.

2.2 Model design

In this part we designed the overall architecture of the sentiment analysis engine providing it with data from a web scraper and tuning its dictionary with financial lexicon.

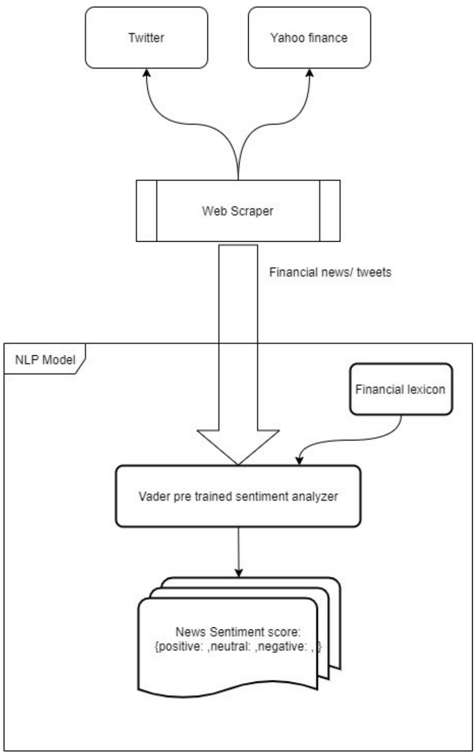


Figure 23 Financial analysis engine design

2.2.1 VADER and financial news

VADER is focused on social media and short texts, unlike Financial News which are almost the opposite. To resolve this, we opted to update the model's lexicon with words and sentiments from the Loughran-McDonald Financial Sentiment Word Lists[10], Which is an English sentiment lexicon created for use with financial documents.

A lexicon is a vocabulary, a list of words, a dictionary used in a specific field.

Here we can see the difference between the pretrained model before and after adding the financial lexicon Fig [25]. The neutrality of the model toward some financial terms was dropped from 0.565 to 0.149, the positive prediction also raised from 0.177 to 0.538.


```
sentiment_analyzer_scores("Facebook rejects call to share revenue with Australian media")
Facebook rejects call to share revenue with Australian media {'neg': 0.258, 'neu': 0.565, 'pos': 0.177, 'compound': -0.25}
```

```
sentiment_analyzer_scores("Facebook rejects call to share revenue with Australian media")
Facebook rejects call to share revenue with Australian media {'neg': 0.313, 'neu': 0.149, 'pos': 0.538, 'compound': 0.0042}
```

Figure 24 Before and after adding the financial lexicon.

2.2.2 Web scraper

To provide VADER with the required financial news and tweets we considered using a web scraper to collect news from multiple data sources and exporting it in a text format exploitable by the NLP model.

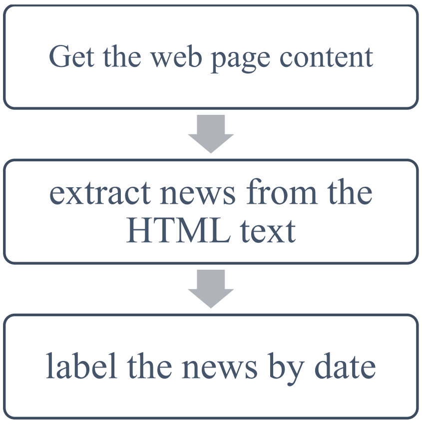


Figure 25 Web scraper logic diagram

❖ Used python's libraries

To accomplish the job stated above we requested the help of some python's open source libraries:

- The **request** library to make network requests

  To scrape data from a website, we need to extract the content of the webpage. Once the request is made to a website, the entire content of the webpage is available, and we can then evaluate the web content to extract data out from it. The content is made available in the form of plain text.

- The **html5lib** library for parsing HTML

Once the content is available, we need to specify the library that represents the parsing logic for the text available. We will use the html5lib library to parse the text content to HTML DOM-based representation.

- The **beautifulsoup4** library for navigating the HTML tree structure

beautifulsoup4 takes the raw text content and parsing library as the input parameters. In our example, we have exposed html5lib as a parsing library. It can then be used to navigate and search for elements from the parsed HTML nodes. It can pull data out from the HTML nodes and extract/search required nodes from HTML structure.

3. Conclusion

In this chapter, we have studied the different deep learning models implemented in the web application, First, we presented the overall and detailed architecture of the LSTM model and described different model's components and layers. Then, we studied the VADER sentiment analysis pretrained model as well as its implementation with the web scraper.

The next chapter will concretize this study by presenting the part relating to the realization of our application.

Chapter 4: Software designing and tests

- *Introduction*
- *UML designing*
- *Software environment & Tools*
- *Graphics interfaces*
- *Conclusion*

Introduction

This part is dedicated to the design of software capable of visualizing the data generated by the RestAPI, we will start with the UML design for this application to clarify the process using different diagrams, then we will go to the development using the visual studio environment, and finish by presenting some screenshots of our application.

1. UML Conception

One of the key steps is to identify the overall approach to be followed in order to clarify the required development and achieve our objectives. For this purpose, we use a modelling language such as UML to have a clearer idea of the software's specifications.

The Unified Modeling Language (UML) is considered as the standard for object-oriented modeling. It allows us to analyze our specific needs.

1.1 Actors

After we move to the UML diagrams, we must define the actors of this software:

- User: is the main actor

1.2 Use case diagram

Describes the functionality provided by a system in terms of actors, their goals represented as use cases, and any dependencies among those use cases.

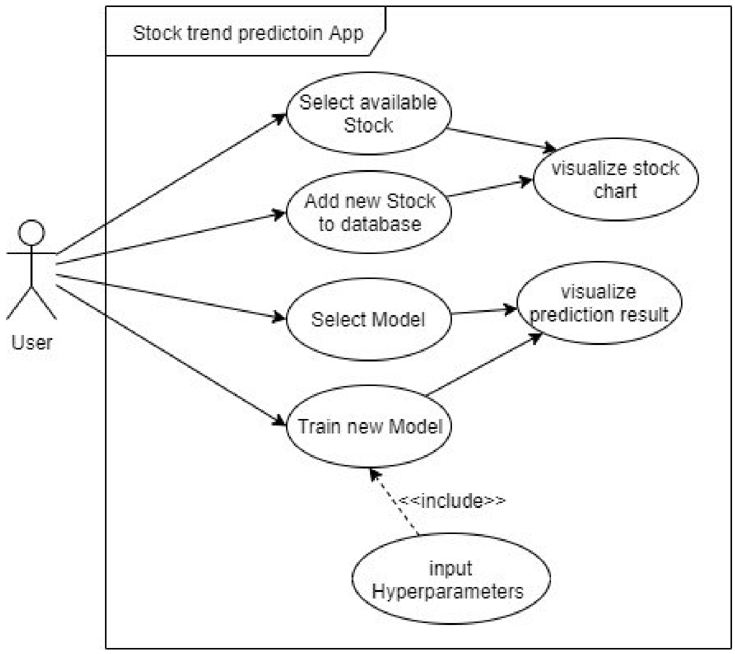


Figure 26 Use case diagram

1.3 Activity diagram

Describes the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

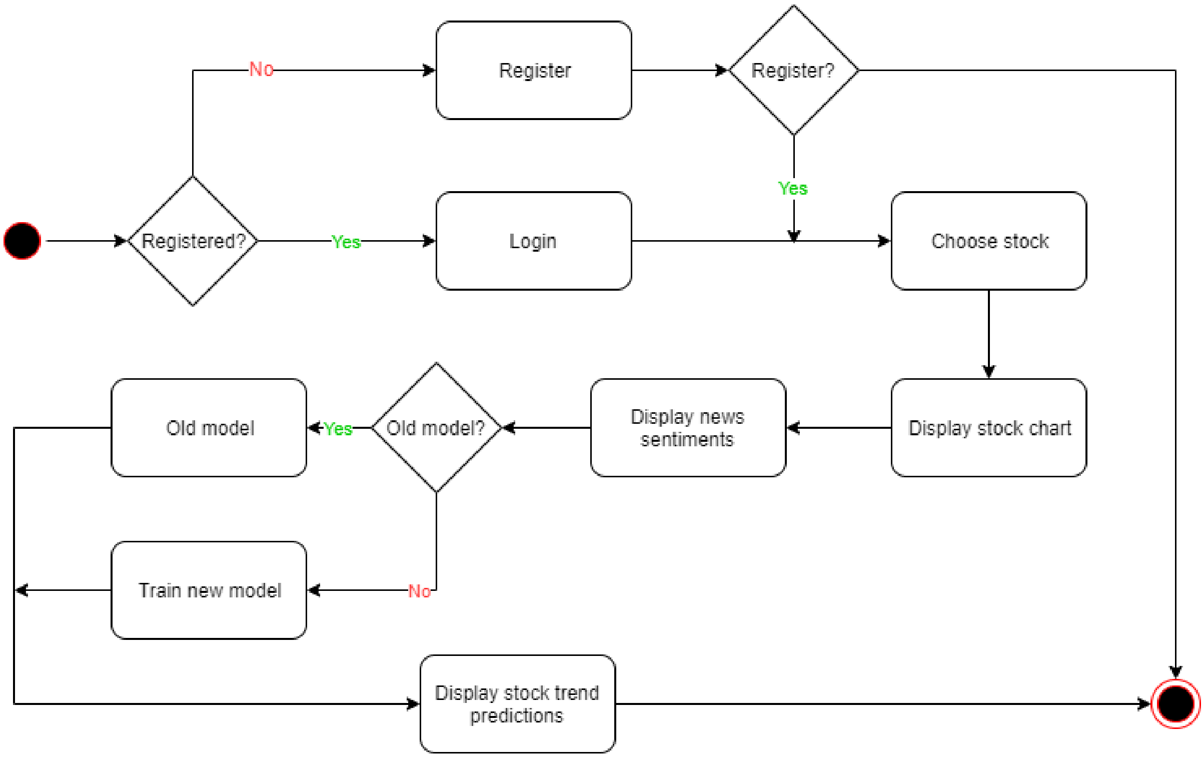


Figure 27 Activity diagram

1.4 Sequence diagram

The sequence diagram shows how objects communicate with each other in terms of a sequence of messages. Also indicates the lifespans of objects relative to those messages.

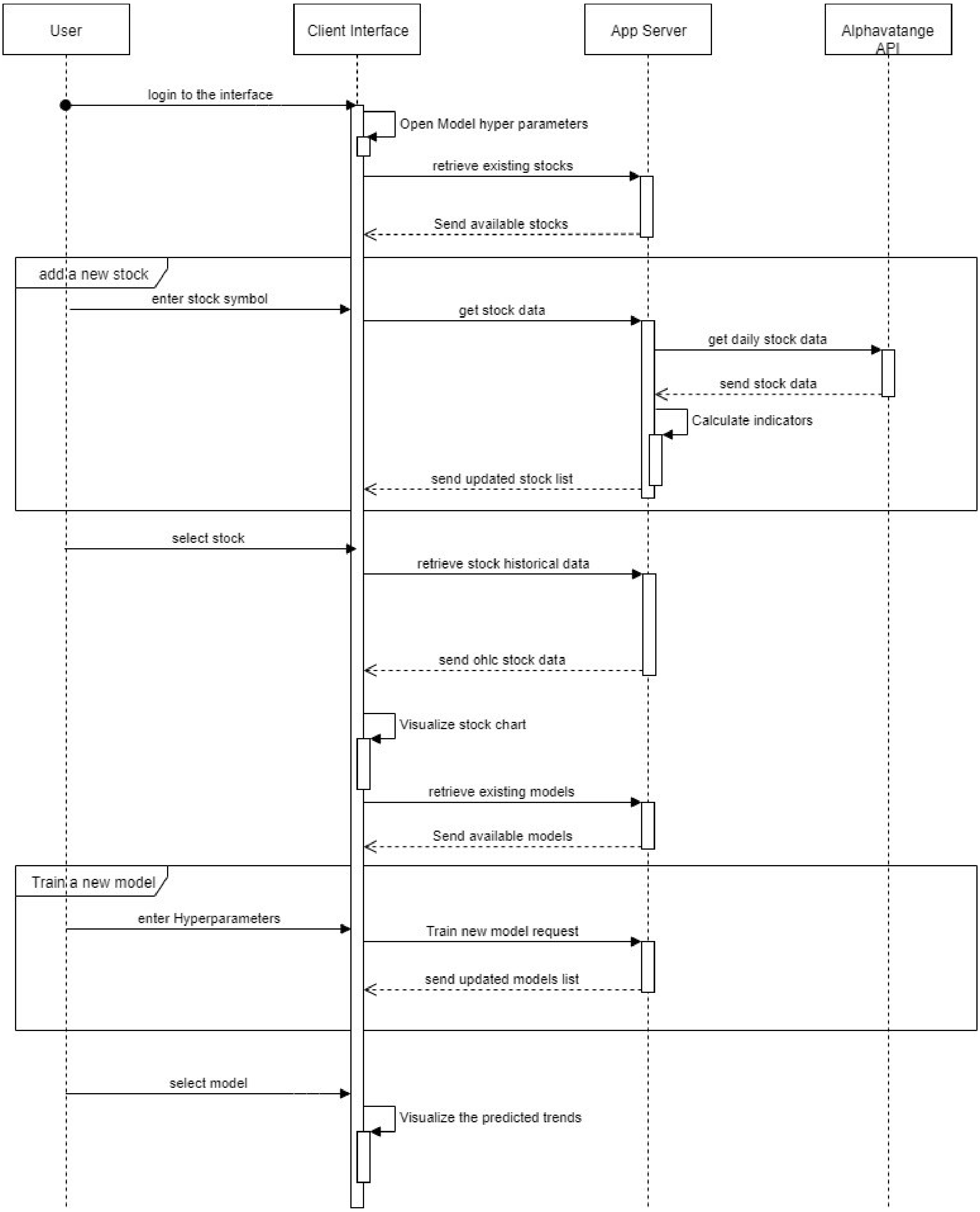


Figure 28 Sequence diagram

2. Tools & Environment

2.1 Visual studio

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is a complete set of development tools for building a variety of applications like desktop applications. The Visual Studio or VS includes an integrated debugger that works both as a source-level debugger and a machine-level debugger

Also, this software supports different programming languages and allows the code editor and debugger to support nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++, VB.NET, C#, and Python. It also supports XML, HTML, JavaScript and CSS among other languages available via language services installed separately.

2.2 Docker

Docker is a software platform for building applications based on containers, small and lightweight execution environments that make shared use of the operating system kernel but otherwise run in isolation from one another. While containers as a concept have been around for some time, Docker, an open source project launched in 2013, helped popularize the technology, and has helped drive the trend towards containerization and microservices in software development that has come to be known as cloud-native development.

2.3 Postman

Postman is a tool created by Google, used by web developers to test custom http requests. It is the only REST client that allows direct connections with sockets. It offers a total control on the connections of the request/response and the headers of the requests.

2.4 Angular

Angular is an open-source front-end web application framework based on Google's TypeScript. It is designed specifically for the creation of dynamic web applications. With this framework, you can develop front-end applications without having to use other plug-ins or other frameworks. Angular is used to develop state-of-the-art client applications, particularly single-page applications. It has a series of features and tools that simplify the development of the applications themselves while ensuring excellent performance results.

- **Pros of Angular**

- **Component-based architecture** that provides a higher quality of code

The component-based architecture is one of the things that makes the difference between AngularJS and its successor. Angular components can be thought of as small pieces of user interface, like a section of the application. While each component is encapsulated with its functionality, there is a strict hierarchy of components in Angular.

Reusability. Components of similar nature are well encapsulated, in other words, self-sufficient. Developers can reuse them across different parts of an application. This is particularly useful in enterprise-scope applications where different systems converge but may have many similar elements like search boxes, date pickers, sorting lists, etc.

Readability. Encapsulation also ensures that new developers, who have been recently onboarded to a project, can read code better and eventually reach their plateau of productivity faster.

Unit-test friendly. The independent nature of components simplifies unit tests, quality assurance procedures aimed at verifying the performance of the smallest parts of the application, units.

Maintainability. Components that are easily decoupled from each other can be easily replaced with better implementations. Basically, your engineering team will be more efficient in maintaining and updating the code within the iterative development workflow.

- **Model-View-ViewModel (MVVM)** is a structural design pattern that separates objects into three distinct groups:

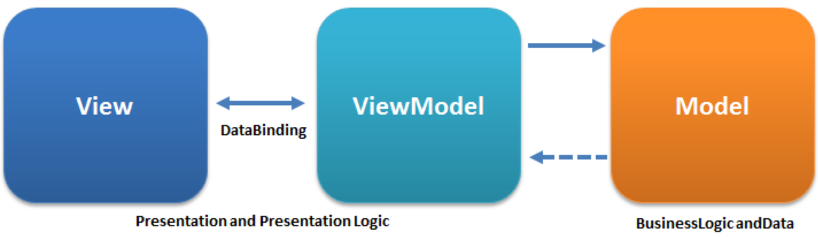


Figure 29 MVVM structure

- **Hierarchical dependency injection.** Angular uses improved hierarchical dependency injection. The technique decouples actual components from their dependencies by running them parallel to each other. Angular builds a separate tree of dependency injectors that can be altered without reconfiguring the components. So, classes do not have dependencies in themselves but consume them from the external source.

2.5 ngx-echarts

ngx-echarts is an Angular directive for ECharts.

ECharts is an open-sourced, web-based, cross-platform framework that supports the rapid construction of interactive visualization.

2.6 Typescript

TypeScript is a strongly typed, object oriented, compiled language. It was designed by Anders Hejlsberg at Microsoft. TypeScript is both a language and a set of tools. TypeScript is a typed superset of JavaScript compiled to JavaScript. In other words, TypeScript is JavaScript plus some additional features.

2.7 HTML

Hyper Text Markup Language (HTML) is a markup language for creating a webpage. Webpages are usually viewed in a web browser. They can include writing, links, pictures, and even sound and video. HTML is used to mark and describe each of these kinds of content so the web browser can display them correctly.

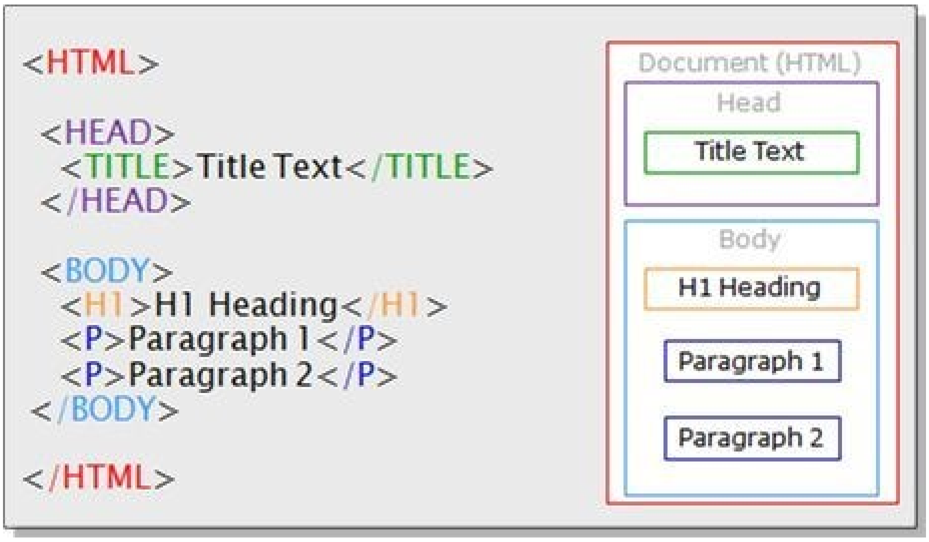


Figure 30 HTML tags

2.8 Cascading Style Sheets (CSS)

CSS stands for Cascading Style Sheets with an emphasis placed on “Style.” While HTML is used to structure a web document (defining things like headlines and paragraphs, and allowing you to embed images, video, and other media), CSS comes through and specifies your document’s style—page layouts, colors, and fonts are all determined with CSS. Think of HTML as the foundation (every house has one), and CSS as the aesthetic choices.

3. Graphics interfaces and backends’ results

In this part, we will detail the execution of the various functionalities through a sequence of screenshots.

3.1 Frontend interface

Here we are illustrating the user interface that will allow us to demonstrate the Backend rest API.

3.1.1 Options Component

The options drop down window fig [31] allow users to tune the Hyperparameters and train a new model, once the user change the parameters and start training the ‘Backend model Hyperparameters’ changes as requested by the user and the newly trained model is saved as a ‘h5’ file under a unique name ‘stock-date-time’.

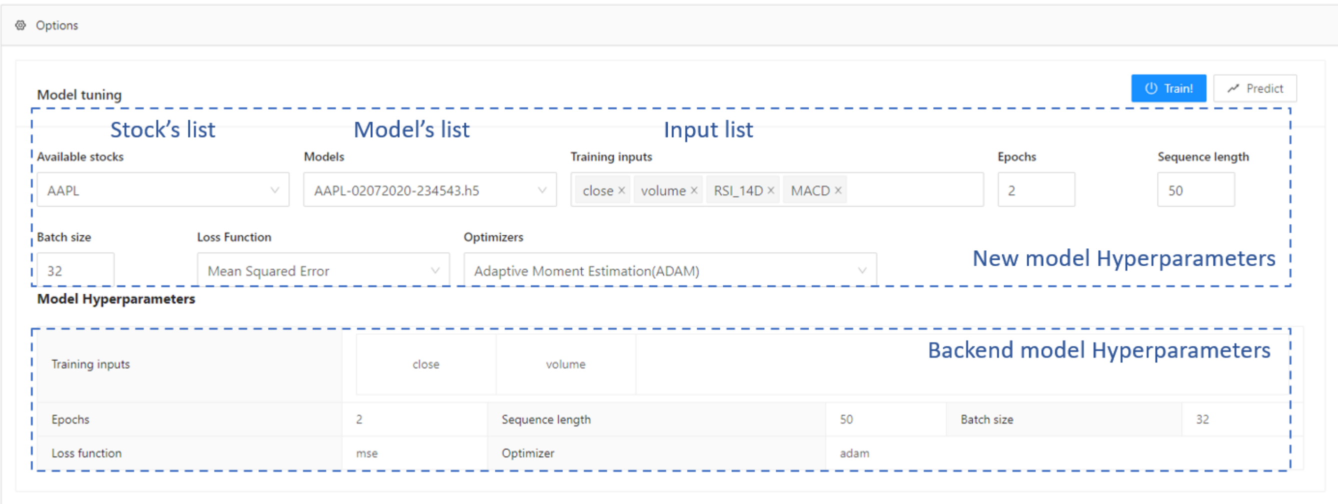


Figure 31 Model options

The parameters and inputs used in the Model tuning fig [31] are stated below:

Available stocks: a dropdown list used to select a given stock and if it is not available it can be added using the company stock's symbol for example: "AAPL" for apple or "GOOGL" for google.

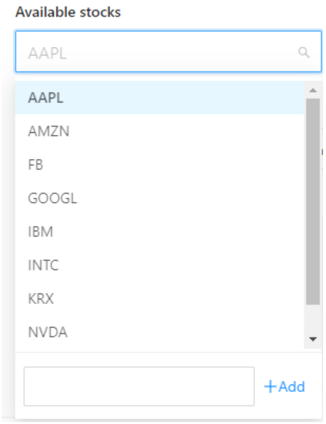


Figure 32 stock's dropdown list

Models: to select a previously trained model.

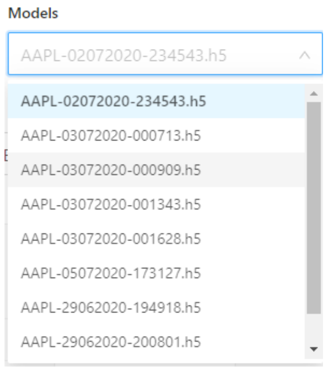


Figure 33 models dropdown list

Training inputs: to select training inputs (indicators and stock data previously discussed in details in 2.5) for the LSTM model.

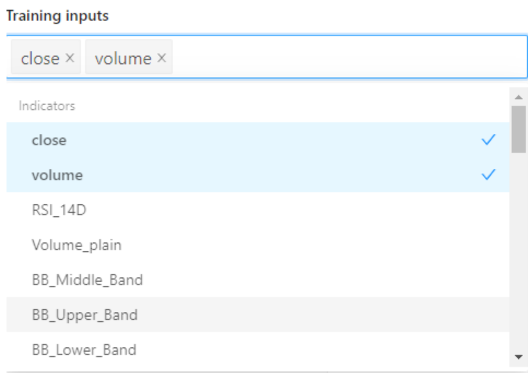


Figure 34 training data selection list

Epochs: the number of times the training data is used to update the neural network's weights.

Sequence length: to select a future time window to be predicted for example 50 days ahead.

Batch size: defines the number of samples to work through before updating the internal model parameters.

Loss function: to select the loss function which is a measure of how good the prediction is.

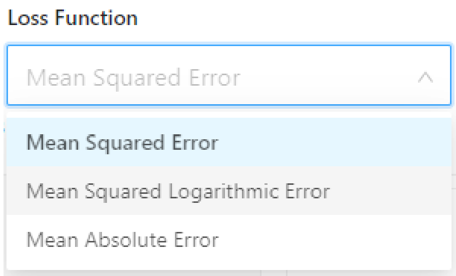


Figure 35 loss functions list

- Mean Absolute Error (MAE) is the simplest regression error metric. it calculates the error between true and predicted value for every data point, taking only the absolute value of each so that negative and positive values do not cancel out. Then it takes the average of all these errors.

$$MAE = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

With: y: The predicted value

x: The real value

n: data length

- Mean Squared Error (MSE): The mean squared error tells you how close a regression line is to a set of points. It does this by taking the distances from the points to the regression line (these distances are the "errors") and squaring them. The squaring is necessary to remove any negative signs. It also gives more weight to larger differences.

$$MSE = \frac{\sum_{i=1}^{n} (y_i - x_i)^2}{n}$$

- Mean Squared Logarithmic Error (MSLE): will treat small differences between small true and predicted values approximately the same as big differences between large true and predicted values.

$$MSLE = \frac{\sum_{i=1}^{n} (\log(y_i + 1) - \log (x_i + 1))^2}{n}$$

Table 4 MSE vs MSLE

True value	Predicted value	MSE loss	MSLE loss
30	20	100	0.02861
30000	20000	100 000 000	0.03100
	Comment	big difference	small difference

Optimizers: optimizers are algorithms or methods used to change the attributes of the neural network such as weights and learning rate in order to reduce the losses.

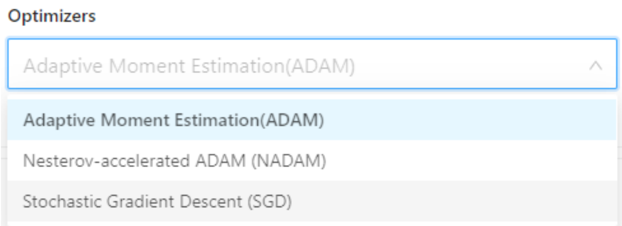


Figure 36 optimizers list

- ADAptive Moment estimation or ADAM is an optimization algorithm that can be used instead of the classical stochastic gradient descent procedure to update network weights iteratively based in training data. Adam is a popular algorithm in the field of deep learning because it achieves good results fast [11].
- Nestrov-accelerated ADAM (NADAM) as the name states combines Nesterov accelerated gradient and ADAM, it is employed for noisy gradients or for gradients with high curvatures [12].
- Stochastic Gradient descent SGD is an iterative algorithm, that starts from a random point on a function and travels down its slope in steps until it reaches the lowest point of that function.

3.1.2 Chart Component

The chart component is responsible for visualizing the data gathered from the backend API.

The trend prediction is back tested on the historical data to analyze the model performance over old data.

Each news gathered from the financial news websites is analyzed and a score is assigned describing its sentiment analysis positive negative and the overall sentiment.

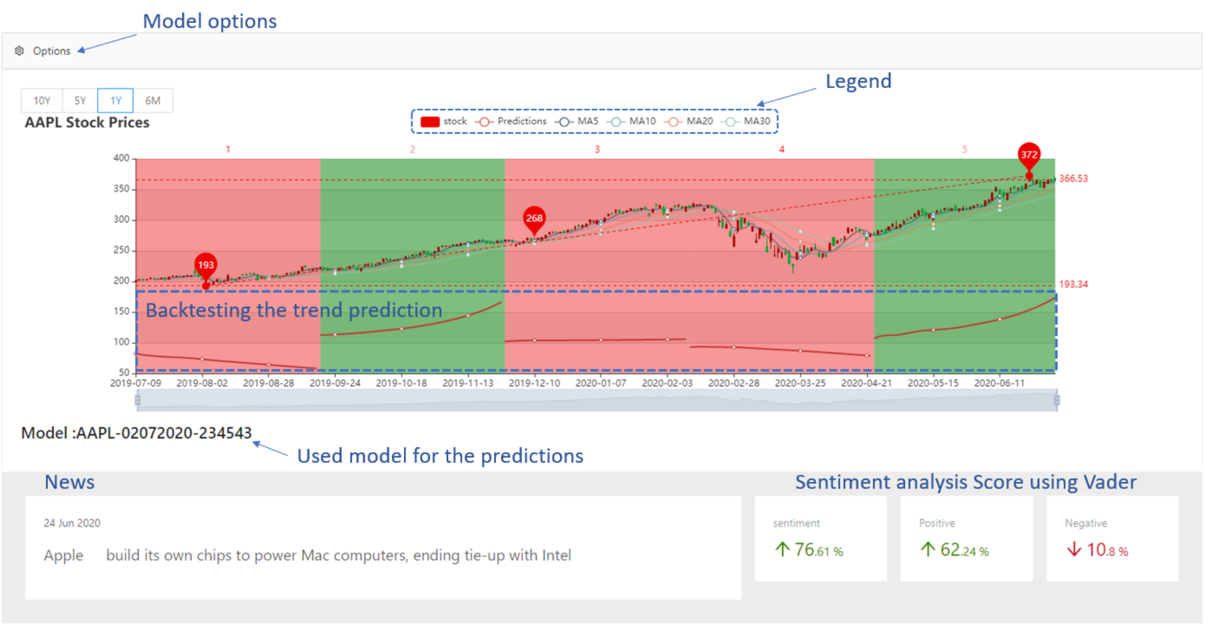


Figure 37 Interface overview

3.2 Backend results

The backend is the main engine of the app it is responsible for all the data preprocessing, database management, sentiment analysis, model tuning and training, these operations are accessible as a web API services via http requests.

3.2.1 Web services

A web service or a restfull web API in this case is a set of routines and functionalities designed to communicate with other application to execute certain tasks.

A web request is sent via e Uniform Source locator address (URL) which is composed of **base URL** and **endpoint:** base URL/endpoint

Base URL: rest API have a base URL to which the endpoint paths are appended, for example here we are using the localhost as a base URL: “http://127.0.0.1:8000/”

Endpoint: is a URL that enables the API to gain access to resources on a server, the following are the main services provided by the web API.

1. Get available stocks [endpoint: “/stocks”]: gather the available stocks
2. Get the models [endpoint: “models/*stock symbol*”]: get the list of the previously trained models corresponding to the selected stock symbol.

3. Get historical data [endpoint: "*stock symbol/data length*"]: get the list of historical data of a selected stock symbol and data length, if the length is not provided the server return all the available historical data.
4. Update or create if not available a stock [endpoint: "Update/*stock symbol*"]: update or create a new stock using the stock symbol as an input and calculate its indicators.
5. Get indicators data [endpoint: "indicators/*stock symbol/data length*"]: get the indicators data of a selected stock symbol and data length, if the length is not provided the server return all the available indicators data.
6. Get stock price prediction [endpoint: "prediction/*stock symbol/model name*"]: using the stock symbol and the trained model name to get the stock predictions.
7. Train a new model [endpoint: "train/*stock symbol*"]: train a new model on the selected stock's data and save it in the database.
8. Get and change the model hyperparameters [endpoint: "model/params"]: tune the model hyperparameters.

3.2.2 Model Hyperparameters

After choosing the appropriate model hyperparameters they are saved under a JSON object. the JSON object will be used by the backend to build the model according to the parameters stated below fig [38].

the same hyperparameters cannot be used for all the stocks because each one got its patterns, so the user should tune the options, train the model over different inputs and backtest the model with historical data to analyze its performance.

```
{
    "data": {
        "columns": [
            "close",
            "volume"
        ],
        "sequence_length": 50,
        "train_test_split": 0.85,
        "normalise": true
    },
    "training": {
        "epochs": 2,
        "batch_size": 32
    },
    "model": {
        "loss": "mse",
        "optimizer": "adam",
        "layers": [
            {
                "type": "lstm",
                "neurons": 100,
                "input_timesteps": 49,
                "input_dim": 2,
                "return_seq": true
            },
            {
                "type": "dropout",
                "rate": 0.2
            },
            {
                "type": "lstm",
                "neurons": 100,
                "return_seq": true
            },
            {
                "type": "lstm",
                "neurons": 100,
                "return_seq": false
            },
            {
                "type": "dropout",
                "rate": 0.2
            },
            {
                "type": "dense",
                "neurons": 1,
                "activation": "linear"
            }
        ]
    }
}
```


Figure 38 model Hyperparameters json object

4. Conclusion

This chapter was devoted to the presentation of the application developed throughout the internship period. We started with the Application modeling, then the presentation of the working environment, followed by the technological choice which describes all the technologies used. Finally, we presented the screenshots of the application and the API services.

General Conclusion and Future Work

The project lays the foundation for democratizing machine learning technologies for investors, connecting predictions made by machine learning models to investors through a web application. It helps investors navigate through the stock markets with additional analysis and help them make more informed decisions.

After studying different financial techniques to predict the stock trend, the project was divided into 4 parts, time series prediction based on historical data, sentiment analysis for financial news, Rest API for data processing, and the user dashboard that will ease the interaction with the models.

First, each part was treated separately starting with the time series forecasting which is the most important part of the application, it is capable of extracting patterns between technical indicators and the trend of the stock price.

The second part was dealing with financial news which is a large amount of data scattered all over the internet, a web scraper was developed to gather this data from financial news websites and this data is fed to a natural language understanding model that will predict the overall sentiment of each news.

The third part focuses on the implementation of the previously discussed models in a Rest API allowing us to build multiple models based on a simple JSON object.

Finally, we have created a web app giving the users the ability to interact with the models intuitively without profound knowledge about machine learning.

The results proved that the application provides a significant asset in trend prediction. When compared to the baseline, the prediction shows a useful trend tendency with the real stock trend. Through the application interface, the user can easily compare the predictions and performances from different machine learning models, then choose the one that fits their preference. The models used in the application can be configured for a better structure and hyperparameters through a user-friendly interface.

Certainly, our work remains open to enhancements. the LSTM and the NLP models are working separately. An implementation of a reinforced learning algorithm gives the possibility to enhance the result by creating trading strategies to execute buy, hold and sell calls.

Bibliography

[1] “Inteligencia IT”: http://www.intelligencia-it.com ,Consulted on February 2020.

[2] “Python”: https://www.python.org , Consulted on February 2020.

[3] “Keras”: https://keras.io/guides/working_with_rnns/ , Consulted on March 2020.

[4] “TensorFlow”: https://www.tensorflow.org/learn, Consulted on March 2020.

[5] “Django”: https://www.djangoproject.com/start/overview/, Consulted on April 2020.

[6] “Django rest framework”: https://www.django-rest-framework.org, Consulted on April 2020.

[7] “AI evolution” : https://www.edureka.co/blog/what-is-deep-learning . Consulted on April 2020.

[8] “LSTM”: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ , Consulted on April 2020 .

[9] “Vader”: https://www.researchgate.net/publication/275828927_VADER_A_Parsimonious _Rule-based_Model_for_Sentiment_Analysis_of_Social_Media_Text , Consulted on May 2020.

[10] “Loughran-McDonald Financial Sentiment Word Lists”: https://sraf.nd.edu/textual-analysis/resources ,Consulted on March 2020.

[11] “Adam: A Method for Stochastic Optimization”: https://arxiv.org/abs/1412.6980 , Consulted on June 2020.

[12] “Incorporating Nesterov Momentum into Adam”: http://cs229.stanford.edu/proj2015/054_report.pdf , Consulted on June 2020.