

# **“STOCK ANALYSIS USING PYTHON”**

Submitted in partial fulfillment of the requirements for the award of degree of

## **BACHELOR OF ENGINEERING IN COMPUTER SCIENCE & ENGINEERING**



**Submitted to: Er. Randeep Kaur  
(ECODE: E7950)**

**Submitted By: Keshav Kant Mishra  
19BCS1887**

**Project Teacher  
(Er. Randeep Kaur –E7950)**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**Chandigarh University, Gharuan**

**February, 2022**

## **CERTIFICATE**

This is to certify that the work embodied in this Project Report entitled “**Stock Analysis using Python**” being submitted by “**Keshav Kant Mishra** ” - UID “**19BCS1887**” , 6<sup>th</sup> Semester for partial fulfillment of the requirement for the degree of “ **Bachelor of Engineering in Computer Science & Engineering** ” discipline in “ **Chandigarh University** ” during the academic session Feb-May 2022 is a record of bonafide piece of work, carried out by student under my supervision and guidance in the “ **Department of Computer Science & Engineering** ”, Chandigarh University.

**APPROVED & GUIDED BY:**

**Er. Randeep Kaur**  
**(Project teacher)**

## **DECLARATION**

I, student of **Bachelor of Engineering in Computer Science & Engineering, 6<sup>th</sup> Semester** , session: **Feb – May 2021, Chandigarh University**, hereby declare that the work presented in this Project Report entitled “**Stock Analysis Using Python**” is the outcome of my own work, is bona fide and correct to the best of my knowledge and this work has been carried out taking care of Engineering Ethics. The work presented does not infringe any patented work and has not been submitted to any other university or anywhere else for the award of any degree or any professional diploma.

**Keshav Kant Mishra**  
**(19BCS1887)**

**Student details**

**APPROVED & GUIDED BY: Er. Randeep Kaur (Project teacher)**

To our parents, teachers and all the well wishers out there . . .

## **ACKNOWLEDGEMENT**

This project is a synergistic product of many minds and has been accumulated over the last few months. This has been a special project brought to fruition through the efforts of some very special people. Many people contributed enthusiastically to this project, which really came together in the last few weeks before deadline. For their continuous guidance and valuable advice I would like to take this opportunity to thank: Firstly, I would like to thank Chandigarh University who has granted me this opportunity to prepare a project which has helped me to gain knowledge beside my studies and which is also definitely going to be useful in future. I would like to place on record my deep sense of gratitude to my parents for financial wisdom and inspiration that have guided and helped me from day one. My project guide, Er. Randeep Kaur(Project teacher), who came in and batted down the hatches when things were flying about. And I would like to thank all those who have helped me contribute their valuable insights and time for this project.

## **ABSTRACT**

Stock market prediction has always caught the attention of many analysts and researchers. Popular theories suggest that stock markets are essentially a random walk and it is a fool's game to try and predict them. Predicting stock prices is a challenging problem in itself because of the number of variables which are involved. In the short term, the market behaves like a voting machine but in the longer term, it acts like a weighing machine and hence there is scope for predicting the market movements for a longer timeframe.

Application of machine learning techniques and other algorithms for stock price analysis and forecasting is an area that shows great promise. In this paper, we first provide a concise review of stock markets and taxonomy of stock market prediction methods. We then focus on some of the research achievements in stock analysis and prediction. We discuss technical, fundamental, short- and long-term approaches used for stock analysis. Finally, we present some challenges and research opportunities in this field.

Technical analysis is the study for forecasting future asset prices with past data. In this survey, we review and extend studies on not only the time-series predictive power of technical indicators on the aggregated stock market and various portfolios, but also the cross-sectional predictability with various firm characteristics. While we focus on reviewing major academic research on using traditional technical indicators, but also discuss briefly recent studies that apply machine learning approaches, such as Lasso, neural network and genetic programming, to forecast returns both in the time-series and on the cross-section.

**Keywords:** Technical Analysis, Machine Learning, Genetic Programming, Cross-sectional Returns, Predictability, stock exchanges; stock markets; analysis; prediction; statistics; machine learning; pattern recognition; sentiment analysis

# **TABLE OF CONTENT**

CONTRIBUTION

DECLARATION

ACKNOWLEDGEMENT

ABSTRACT

GLOSSARY

INTRODUCTION

What is AI & ML?

- How does AIML works

1. Rich library ecosystem
2. Flexibility
3. Simple and Consistent
4. Platform Independent

What exactly is Deep Learning?

Why is Deep Learning is Popular these Days?

Understanding Stock Analysis

Stock Price Analysis with Python

Setting up our environment

Getting Live Data

Analyzing the Market Gap

Moving Average

Percentage increase in Stock Values

The Theory

Portfolios and Other Assets

Related Studies

Machine Learning

- Supervised Learning

- Unsupervised Learning

Predicting Stock with python

- Support Vector Regression

- Linear Regression

Code for Stock prediction

- Starting with Stocker

- Additive Tools

- Change points

Challenges and Open Problems

Conclusion

BIBLIOGRAPHY

# Glossary

**Action task** – Action tasks collect, modify, and post information in systems of record, like scheduling an appointment, searching for a product, or updating critical information.

**Administration & analytics** – The Kore.ai Platform provides enterprise grade visibility and control of all enterprise bots, user groups, and security, as well as a commitment to compliance in even the most highly regulated areas like healthcare and financial services. [Learn more](#)

**Alert task** – Alert tasks deliver timely, relevant, and personalized information to customers and employees directly from enterprise systems. Bots poll the system for user requested updates in real-time.

**API** – An API, or Application Programming Interface, is a set of definitions, protocols, and tools for building application software. It helps developers by essentially providing the building blocks for a program.

**Artificial Intelligence (AI)** – AI is the development of computer systems that are able to perform tasks that normally require human-like intelligence, like decision making, speech recognition and understanding, translation between languages, and more.

**Auto-NLP** – A term we use at Kore.ai to describe our synonym-based approach to natural language processing. It allows chatbots to communicate understand intent variations right out of the gate, thus being speech enabled “automatically”

**Automatic Message Formatting** – The pre-programmed responses for tasks built into Kore.ai’s NL engine.

**Automated Speech Recognition (ASR)** – Our Platform can integrate Automated Speech Recognition Engine to enable bots to process voice-driven interactions and communicate outside of traditional text-based interfaces.

## C

**Channel** – A channel is another word for any of the various communication platforms where a bot can live such as SMS, email, mobile apps, websites, messaging apps and more.

**Chat logs** – Histories of all recorded human-to-bot interactions.

**Cisco Spark** – Cisco’s all-in-one communication platform, which is also a supported channel for the Kore.ai bots and an integration partner for chatbot development.

**Cloud (or Cloud computing)** – Internet-based computing that provides shared computer processing resources and data to computers and other devices on demand.

**Cloud Connector** – Provides an agent that runs behind your enterprise firewall that acts as a bridge to facilitate secure data exchanges between on-prem systems and Kore.ai’s cloud based infrastructure.

**Component reusability** – The ability for developers to use components they’ve already built in the Bot Builder, like APIs, synonyms, tasks, etc. and apply them to other bots.

**Context** (see also, Bot Context, Enterprise Context, Session Context, User Context) – The information that a chatbot pulls from a conversation with a user that it can leverage when performing tasks. Contextual data can vary in importance, utility, and lifespan.

**Conversational Commerce** – A term coined by Chris Messina in 2016, which is another way of describing how digital economies will be driven by text and voice based interfaces and experiences.



Conversational UI – Another way of describing text and voice-based interfaces, which don't require graphical elements for use, like Amazon's Alexa or Apple's Siri.

## D

Data retention – The continued storage of an organization's data for compliance or business matters.

Deep learning – An area of Machine Learning that is based on learning data representations as opposed to task specific algorithms.

Deployment – The process of publishing a bot to communication channel where it will be engaged by users.

Dialog task – Dialog tasks are advanced tasks that developers design with logic-driven business processes and pre-established workflows. Bots key off the primary request intent to accomplish the task at hand, then go above and beyond to execute sub-intents and additional workflows.

Dialog Builder – The Kore.ai Dialog Builder gives designers and developers the flexibility to manipulate the entire dialog process of a bot interaction and string together complex workflows in a GUI-based tool.

## E

Ecommerce – Any monetary transaction conducted on the internet.

Ediscovery – Any process in which electronic data is sought, located, secured, and searched with the intent of legal use. The Kore.ai Platform supports e-discovery.

Email – A supported channel for Kore.ai bots.

Encryption – The process of converting information or data into a code, especially to prevent unauthorized access.

End-to-end – A way of describing the Kore.ai Bots Platform which signifies that it includes all the component features to take enterprises from the very beginning of the chatbot development process, through deployment and management.

Enterprise analytics – The central dashboard within the Kore.ai Platform where administrators can get visibility into key metrics, pull detailed reports, and track bot usage (i.e. number of executed tasks, most popular channels, most active users, user enrollment, etc.)

Enterprise-grade – A way of describing all of the components and capabilities of the Kore.ai Bots Platform that are specifically designed to match the highest enterprise standards, including administration, analytics, security, compliance, and more.

Enterprise context – Information that represents company-wide rules and standards that apply to all users and bots, such as a company travel policy, or expense limits.

Entity – Entities are the fields, data, or words the developer designates are necessary for a chatbot to complete the user's request. An entity could be a date, a time, a location, a description or any number of things.

Entity extraction – This is the process by which the Kore.ai NL engine identifies words from a user's utterance to ensure all available fields match the task at hand. If the chatbot needs an entity to complete the task after initial extraction, it will prompt the user for it.

## F

Facebook Messenger – A supported channel for chatbots built on Kore.ai's Bots Platform, primarily used when companies build bots for end customers.

**FAQ** – The primary data source chatbots use to pull information to complete knowledge tasks. Coming soon to the Platform will be the ability for website and data based knowledge and document-based knowledge.

**Framework** – A framework is a skeleton that provides some basic building blocks and generic functionality for building chatbots (like ML/ NLP or a Dialog Builder), but requires additional user-written code or other third-party services (to match the functionality of an actual platform). Frameworks often are composed of piecemeal components from different vendors.

**Fundamental Meaning** – Fundamental Meaning is an approach to NLP that's all about understanding words themselves. Each user utterance is broken down word-for-word to search for intent (what the user is asking it to do) and entities (the necessary data needed to complete a task). Learn more about this approach and the Kore.ai NL engine.

## G

**Glip** – A supported channel for chatbots built on Kore.ai's Bots Platform.

**Graphic User Interface (GUI)** – A visual way of interacting with an app or system, such as buttons, images, windows, icons, menu forms, and more.

## H

**Hosting** – Enterprises have the choice of hosting the Kore.ai Bots Platform on prem or in the cloud via AWS.

## I

**Information and Communication Technologies (ICT)** – ICT refers to technologies that provide access to information through telecommunications. It is similar to Information Technology (IT), but focuses primarily on communication technologies. This includes the Internet, wireless networks, cell phones, and other communication mediums.

**Information task** – Information tasks lookup data or pull reports based on specific parameters and quickly return easy-to-consume results that are convenient for users.

**Interface** – A shared

## M

**Machine Learning** – Using algorithms, patterns, and training data, machine learning allows computers to find hidden insights without being explicitly programmed. Learn more about the way Kore.ai uses machine learning for natural language enablement.

**Managed Services Provider (MSP)** – Most often an IT provider that manages and assumes responsibility for providing a defined set of services to its clients either proactively or as the MSP determines that services are needed

**Memory** – Bots can remember actions, data, and contextual details to maintain conversation continuity and take helpful actions. The developer can designate how long the bot remembers information as either short term or long term memory.

**Message Broker** – Consumes all user inputs and system outputs, standardizes for a common messaging paradigm, and redirects to the appropriate endpoints.

**Middleware** – The Kore.ai Platform Middleware contains the Message Broker, Message Store, and built-in encryption to create a flawless conversational experience by ensuring messages are received, secured, and exchanged in real time.

## N

**Natural Language (NL)** – The method by which users can talk to systems in everyday language like text and speech, rather than programming language.

**Natural Language Processing (NLP)** – The process by which a chatbot or any other system understands and processes requests in common language, rather than programming language. NLP is typically enabled via machine learning, but Kore.ai uses a dual-pronged approach which includes intent recognition and entity extraction.

Natural Language Training – The processes in which you refine a chatbot’s ability to understand and process NL requests, and test accordingly. It can be done by adding synonyms to the chatbot’s vocabulary via the Kore.ai Bot Builder, or training with complete utterances via machine learning. You can learn more about how to NL train a bot by watching [How To Build A Chatbot In 5 Minutes](#).

Neural networks – A computer system modeled on the human brain and nervous system.

Nodes – A node is where different points of a dialog or workflow intersect. The Kore.ai Dialog Builder has multiple node types so developers can steer the human-to-bot conversation in different directions, including intent nodes, service nodes, message nodes, confirmation nodes, webhook nodes, and JavaScript nodes. Learn more about how you can use nodes to build more complex and free flowing chatbot dialogs

Omni-channel (deployment) – The process of building one chatbot that is “channel agnostic” (meaning the bot can live in any channel), and deploying it to the communication channels of your choice.

---

# ***Project Description***

---

# **INTRODUCTION**

## **“What is AI & ML?”**

AI (Artificial Intelligence)

What comes to mind when I ask you about Artificial Intelligence (AI)? Is it a case of robots taking over the world? Or something you might have seen in a science fiction film? Don't be concerned; it happens to all of us! Even when I first started learning about it, I had this thought. But as I dug deeper into AI, I realized that it is nothing like that, and yet it is so much more. If the twenty-first century is to be remembered for anything, it must be AI and the changes it has brought. Artificial Intelligence refers to the intelligence demonstrated by machines. Artificial intelligence has grown in popularity in today's world. It is the simulation of natural intelligence in machines that are programmed to learn and mimic human actions.

These machines can learn from experience and perform human-like tasks. As artificial intelligence (AI) technology advances, it will have a significant impact on our quality of life.

It is only natural that everyone today wants to connect with AI technology in some way, whether as an end user or as a developer. Humans created an intelligent entity. Capable of intelligently performing tasks without being explicitly instructed. Capable of rational and humane thought and action.

## **How does AI & ML work?**

While it's one thing to know what AI is, it's another to understand the underlying functions. Artificial intelligence operates by processing data through advanced algorithms. It combs large data sets with its algorithms, learning from the patterns or features in the data. There are many theories and subfields in AI systems including:

**Machine learning.** Machine learning uses neural networks to find hidden insights from data, without being programmed for what to look for or what to conclude. Machine learning is a common way for programs to find patterns and increase their intelligence over time.

**Deep learning.** Deep learning utilizes huge neural networks with many layers, taking advantage of its size to process huge amounts of data with complex patterns. Deep learning is an element of machine learning, just with larger datasets and more layers.

**Cognitive computing.** Cognitive computing has a goal for a human-like interaction with machines. Think robots that can see and hear, and then respond as a human would.

**Computer vision.** In AI, computer vision utilizes pattern recognition and deep learning to understand a picture or video. This means the machine can look around and take pictures or videos in real time, and interpret the surroundings. The overall goal of AI is to make software that can learn about an input, and explain a result with its output. Artificial intelligence gives human-like interactions, but won't be replacing humans anytime soon.

## **Why is Python So Popular for AI and Machine Learning?**

**With each passing minute, AI and machine learning are grabbing more eye balls than ever. Who'd have thought that there could exist a self-driving car or smart phones that forecast what weather it will be tomorrow! But today, all this is a reality. How Does Home Automation Work?**

Companies like Uber, Tesla, JP Morgan Chase, Apple, and other industry giants have accepted these technologies. With this, they've also befriended one programming language that is flexible, stable, with a variety of tools available: PYTHON.

Here are the top reasons that make Python so popular for AI and machine learning:

### **1. Rich library ecosystem**

A programming language library refers to a module that comes with a pre-written code that helps the user to use the same functionality to perform different actions. Python contains libraries that help in saving developer's time as they do not have to start from scratch.

List of some common libraries used for AI and machine learning:

- Pandas
- Scikit-learn
- Keras
- TensorFlow
- Caffe
- PyBrain

With the help of these libraries, AI and ML algorithms can be implemented more easily. These libraries are useful for data analysis, deep learning, machine learning, computer vision, and advanced computing. This helps in the faster development of the product as the developers can now resolve complex problems without rewriting code lines.

### **2. Flexibility**

Python is a flexible language, which means that it can be used along with other programming languages to achieve the desired result. It offers an option to the developer to choose between OOPs or scripting. Also, it does not require recompilation of the source code, making it easier to view the results. Due to its flexibility, it gives the developer a safe environment and reduces the chances of mistakes.

### 3. Simple and Consistent

This programming language offers concise, readable codes. As complex algorithms stand behind AI and ML, the simplicity of the language helps in developing reliable systems. Now the entire focus is on solving an ML problem instead of worrying about the technical details of the language.

Another reason which makes Python so popular is that it is an easy-to-learn programming language. Due to its easier understandability by humans, it is easier to make models for machine learning. Furthermore, many coders say that Python is more intuitive than other programming languages. It is suitable for a collaborative implementation as and when multiple developers are involved. Being a general-purpose language, it allows you to build prototypes faster so that you can test your product for machine learning.

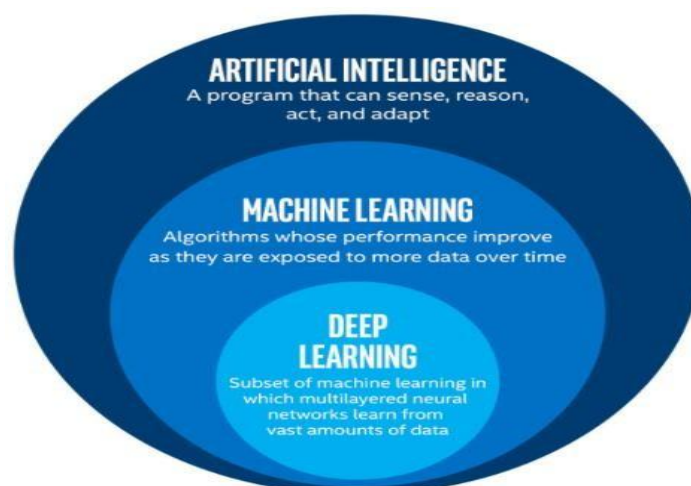
### 4. Platform Independent

Platform independence of a programming language means that it can run on a variety of platforms and software architectures. The code has to be written once and it can be compiled and run on multiple platforms.

Python is easy to learn and use and scores high on versatility. It can run on any platform, be it Windows, MacOS, Linux, Unix, and more. If one wants to run the code of different platforms, packages like PyInstaller come in handy. Let's say a coder wants to shift from one platform to another, it is far easier with Python. This saves time and money for tests on multiple platforms. As a result, the overall process becomes more convenient.

## What exactly is Deep Learning?

Deep Learning is a subset of Machine Learning, which on the other hand is a subset of Artificial Intelligence. Artificial Intelligence is a general term that refers to techniques that enable computers to mimic human behavior. Machine Learning represents a set of algorithms trained on data that make all of this possible.



## AI. vs ML. vs DL.

Deep Learning, on the other hand, is just a type of Machine Learning, inspired by the structure of a human brain. Deep learning algorithms attempt to draw similar conclusions as humans would by continually analyzing data with a given logical structure. To achieve this, deep learning uses a multi-layered structure of algorithms called neural networks.

The design of the neural network is based on the structure of the human brain. Just as we use our brains to identify patterns and classify different types of information, neural networks can be taught to perform the same tasks on data.

The individual layers of neural networks can also be thought of as a sort of filter that works from gross to subtle, increasing the likelihood of detecting and outputting a correct result.

The human brain works similarly. Whenever we receive new information, the brain tries to compare it with known objects. The same concept is also used by deep neural networks.

Neural networks enable us to perform many tasks, such as clustering, classification or regression. With neural networks, we can group or sort unlabeled data according to similarities among the samples in this data. Or in the case of classification, we can train the network on a labeled dataset in order to classify the samples in this dataset into different categories.

## Why is Deep Learning is Popular these Days?

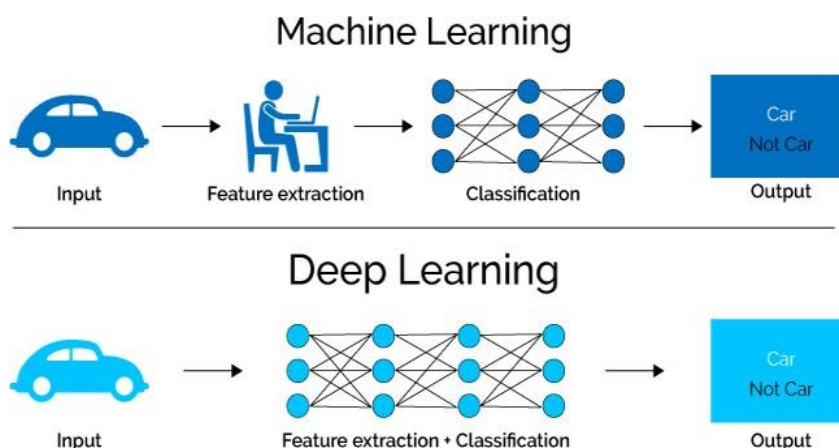
Why is deep learning and artificial neural networks so powerful and unique in today's industry? And above all, why are deep learning models more powerful than machine learning models? Let me explain it to you.

*The first advantage of deep learning over machine learning is the needlessness of the so-called feature extraction.*

Long before deep learning was used, traditional machine learning methods were mainly used. Such as Decision Trees, SVM, Naïve Bayes Classifier and Logistic Regression.

These algorithms are also called flat algorithms. Flat here means that these algorithms can not normally be applied directly to the raw data (such as .csv, images, text, etc.). We need a preprocessing step called Feature Extraction.

The result of Feature Extraction is a representation of the given raw data that can now be used by these classic machine learning algorithms to perform a task. For example, the classification of the data into several categories or classes.





## Understanding Stock Analysis

Stock analysis is a method for investors and traders to make buying and selling decisions. By studying and evaluating past and current data, investors and traders attempt to gain an edge in the markets by making informed decisions.



## Stock Price Analysis With Python

A stock is the small chunk of ownership in the company. The stock price of the company reflects the net evaluation of the company and also gives a little insight into its performance. These stocks are traded on exchanges and their prices are constantly changing due to their demand and supply in the market. If a stock is in high demand and low in supply i.e. more people want to buy it and fewer people are willing to sell it then the price for the stock will go up and similarly if the stock is in low demand and high on supply which means people more people are ready to sell it but fewer people are willing to buy it then its prices go down.

The sudden increase in the demand for the stock can be due to various reasons including positive news about the company or an announcement from the company. After a period of time when the demand for the stock vanishes its prices slowly creep down as the investor loses interest in it. These stock prices going up and down is an iterative process and repeated. This volatility of stock makes investors nervous while investing in a company. So to understand the risk associated with it there must be a proper analysis of stock before buying it. In this article, we would try to explore just the tip of the iceberg for the stock market analysis as technical analysis of the stock is a vast field. This blog can prove to be the starting point for you in this industry.

The tool is not important for the analysis it can be performed in any statistical software like Python, R, or Excel but for sake of this article, we are demonstrating the analysis in Python.

## Setting up Our Environment

Before diving into any project it is important to have our toolbox ready, this is mainly setting up the environment.

We will create a folder and we will name ours 'Analysing\_Stock\_Prices', open our Anaconda prompt as an administrator and cd into the folder 'Analysing\_Stock\_Prices', and open the Jupyter Notebook.

```
Administrator: Anaconda Prompt (Anaconda3) - Jupyter Notebook
(base) C:\WINDOWS\system32>cd C:\Users\brian\Desktop\Analysing_Stock_Prices
(base) C:\Users\brian\Desktop\Analysing_Stock_Prices>Jupyter Notebook
[I 2021-12-17 09:36:49.670 LabApp] JupyterLab extension loaded from C:\ProgramData\Anaconda3\lib\site-packages\jupyterlab
[I 2021-12-17 09:36:49.671 LabApp] JupyterLab application directory is C:\ProgramData\Anaconda3\share\jupyter\lab
[I 09:36:49.731 NotebookApp] The port 8888 is already in use, trying another port.
[I 09:36:49.761 NotebookApp] The port 8889 is already in use, trying another port.
[I 09:36:49.764 NotebookApp] The port 8890 is already in use, trying another port.
[I 09:36:49.768 NotebookApp] Serving notebooks from local directory: C:\Users\brian\Desktop\Analysing_Stock_Prices
[I 09:36:49.769 NotebookApp] Jupyter Notebook 6.4.5 is running at:
[I 09:36:49.769 NotebookApp] http://localhost:8891/?token=df053943c604aa0e6c8c0f714b309817a020ec90a991f99a
[I 09:36:49.770 NotebookApp] or http://127.0.0.1:8891/?token=df053943c604aa0e6c8c0f714b309817a020ec90a991f99a
[I 09:36:49.770 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 09:36:49.983 NotebookApp]

To access the notebook, open this file in a browser:
    file:///C:/Users/brian/AppData/Roaming/jupyter/runtime/nbserver-17524-open.html
Or copy and paste one of these URLs:
    http://localhost:8891/?token=df053943c604aa0e6c8c0f714b309817a020ec90a991f99a
    or http://127.0.0.1:8891/?token=df053943c604aa0e6c8c0f714b309817a020ec90a991f99a
```

## Getting Live Data from Yahoo Finance

We will be using Pandas data reader, to get live data for us to work with and analyze.

We will start by importing the pandas data reader and the date-time module, we will use the data reader for remote data access and the datetime module for specifying the begin and end date times.

```
In [4]: bmw = web.DataReader("BMW.DE", "yahoo", start,end)
```

```
In [5]: bmw
```

```
Out[5]:
```

	High	Low	Open	Close	Volume	Adj Close
Date						
2015-01-02	89.589996	87.379997	88.599998	88.010002	1532820.0	64.752594
2015-01-05	87.230003	84.550003	87.180000	85.080002	2308143.0	62.596867
2015-01-06	87.120003	84.550003	85.800003	85.830002	1841589.0	63.148670
2015-01-07	87.050003	85.269997	86.400002	86.290001	1239393.0	63.487103
2015-01-08	89.870003	86.879997	87.519997	89.389999	1837539.0	65.767906
...	...	...	...	...	...	...
2021-12-10	90.190002	88.949997	89.529999	89.660004	1415043.0	89.660004
2021-12-13	91.879997	89.550003	89.800003	89.879997	1086537.0	89.879997
2021-12-14	90.040001	88.220001	89.980003	88.400002	1116145.0	88.400002
2021-12-15	89.410004	88.230003	88.889999	88.269997	794212.0	88.269997
2021-12-16	90.309998	89.269997	89.750000	89.639999	1364574.0	89.639999

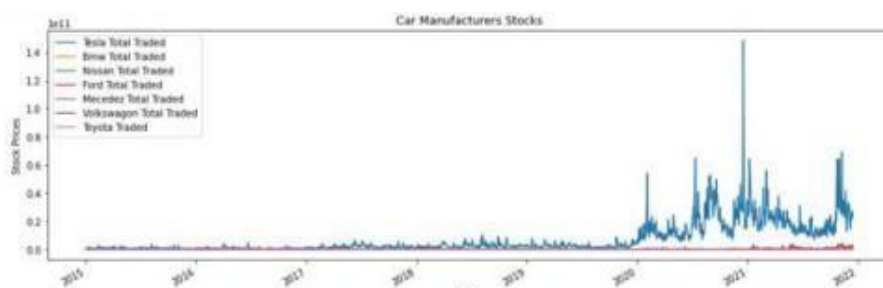
## Analyzing The Market Gap

We want to check which of the car manufacturers is more valuable than the other. We do not want to rely simply on the time series data.

Our current data can not help us analyze this, that however won't stop us from using tricks within mathematics to get that data. We will use some basic math to get the total units of the stock present.

We will use the open price and the Volume, our end goal is to get the total money traded.  
So we want to add a column for the total traded, for each car manufacturer.

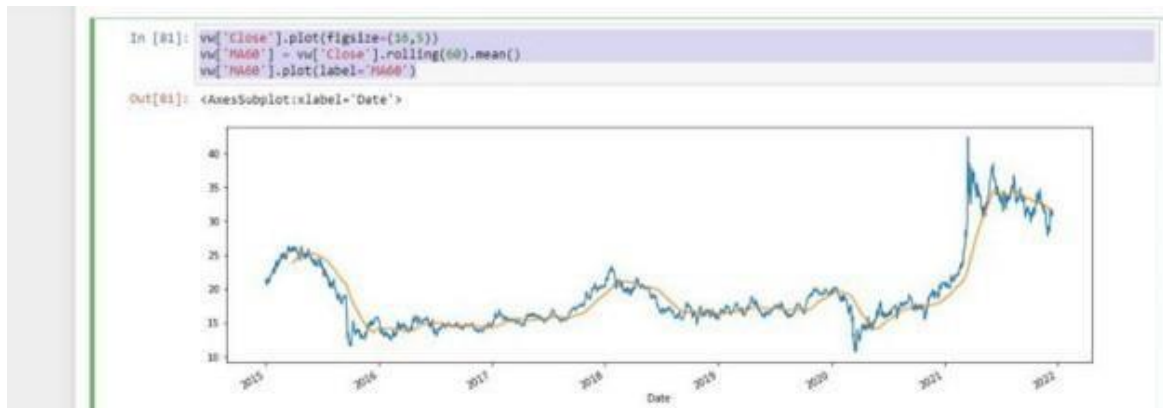
```
tesla['Total Traded'].plot(label='Tesla Total Traded', figsize=(16,5))
bmw['Total Traded'].plot(label='Bmw Total Traded')
nissan['Total Traded'].plot(label='Nissan Total Traded')
ford['Total Traded'].plot(label='Ford Total Traded')
mercedes['Total Traded'].plot(label='Mecedez Total Traded')
vw['Total Traded'].plot(label='Volkswagon Total Traded')
toyota['Total Traded'].plot(label='Toyota Traded')
plt.legend()
plt.title('Car Manufacturers Stocks')
plt.ylabel('Total Traded')
plt.show
```



## Moving Average

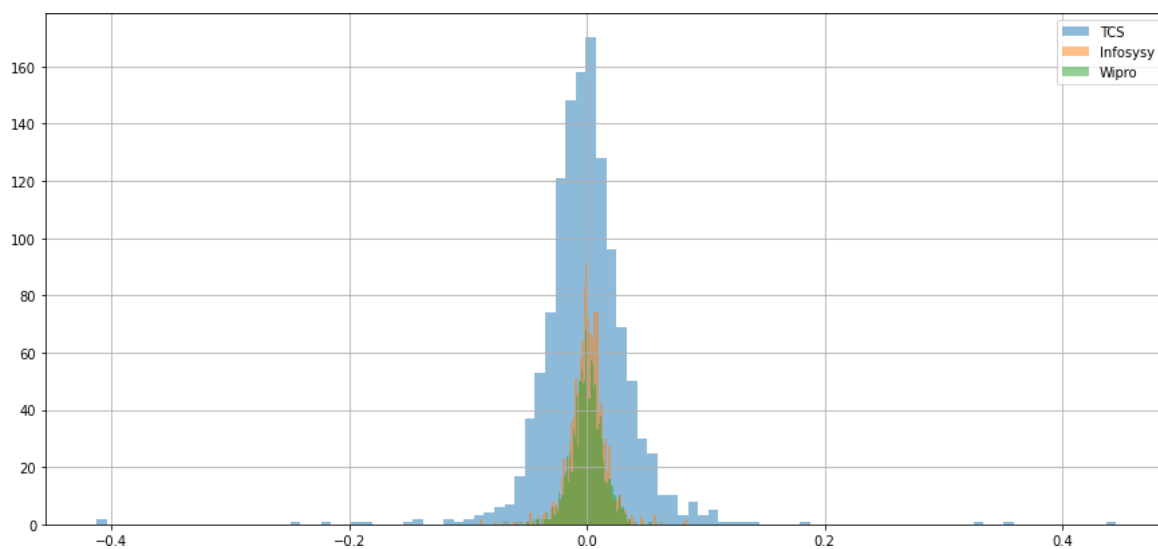
We want to read each manufacturer separately as it seems at this moment Tesla is winning, but we need need to analyze everything in order to make informed decisions. We will just work with the Close for now.

The noise has been smoothened out a bit, and we can increase our moving average to maybe 300.



## Percentage increase in stock value

A percentage increase in stock value is the change in stock comparing that to the previous day. The bigger the value either positive or negative the volatile the stock is.



It is clear from the graph that the percentage increase in stock price histogram for TCS is the widest which indicates the stock of TCS is the most volatile among the three companies compared.

## The Theory

A short survey of theoretical models that justify the use of technical analysis is provided by Han, Zhou and Zhu (2016). Here we focus on the most widely used technical indicators, the moving averages (MAs), which are the foundation of trend-following.

Zhu and Zhou (2009) seems the first to provide a theoretical basis for the MAs. In a partial equilibrium model for a small investor, the MAs are fast learning methods about the underlying true model of asset dynamics. Under uncertainty about predictability or uncertainty about the true parameters, the MA learning can add value to an asset allocation problem in reasonable sample sizes. In contrast, sophisticated econometric methods, though asymptotically optimal, underperform the simple MAs due to not enough data. As an extension, Zhou, Zhu and Qiang (2012) provide an optimal asset allocation strategy using the MAs, and show it makes a significant economic difference empirically.

Han, Zhou and Zhu (2016) propose further an equilibrium model in which there are informed traders and technical traders. In the presence of noise traders, they show that, even in equilibrium, the technical traders can survive in the long-run and the MAs have predictive power on asset prices. However, the fraction of technical traders can matter. When the fraction is small, MAs indicate trend-following, but when the fraction is large, they predict counter-trends. In contrast to Han, Zhou and Zhu (2016) where the technical traders are assumed exogenously, Detzel, Liu, Strauss, Zhou and Zhu (2021) propose a novel equilibrium model in which technical analysis can arise endogenously via rational learning. They document that ratios of prices to their moving averages forecast daily Bitcoin returns in- and out-of-sample, and similar results hold for small-cap, young-firm, and low-analyst coverage stocks as well as NASDAQ stocks during the dotcom era.

A limitation of the theoretical model of Han, Zhou and Zhu (2016) is that it justifies the use of one MA as a predictor. In practice, multiple MAs may be used at the same time. For simplicity, we consider two MA predictors. We show in what follows that the main conclusion can be extended to allowing for two MAs as predictors.

Consider now three types of investors: the informed, the short-term and long-term technical traders. Let  $w_1$  and  $w_2$  be the fraction of the short- and long-term technical traders. The informed investors observe the dividend  $D_t$ , mean growth rate of dividend  $\pi_t$ , the price as well as all history of the variables, while they do not directly observe the supply of asset, the fluctuation of which is due to noise trader behavior. Formally,

$$F_i(t) = \{D_\tau, P_\tau, \pi_\tau : \tau \leq t\}$$

is the informed investors' information set at time  $t$ . On the other hand, technical traders only observe dividend and price, and do not directly observe the mean grow rates of the dividends. The two types of technical investors use

$$A_{it} \equiv \int_{-\infty}^t \exp[-\alpha_i(t-s)] P_s ds,$$

with  $i = 1, 2$  and  $\alpha_1 > \alpha_2 > 0$ , to infer information.  $A_{1t}$  corresponds to the short-term signal and  $A_{2t}$  the long-term signal. Formally,  $F_{iu}(t) = \{1, D_t, P_t, A_{it}\}$ ,  $i = 1, 2$  is the information sets of the technical traders of type  $i$  at time  $t$ .

## **Portfolios and Other Assets**

There are many studies on the use of technical analysis in other markets besides the stock market. Taylor and Allen (1992) show that currency market seems the next largest place where technical analysis is widely used. However, as shown by Hsu and Taylor (2014), the predictive power, like that in the stock market, tends to decrease over time.

In general, Fung and Hsieh (2001) find that trend-following trading is of great importance for explaining hedge fund returns. Burghardt and Walls (2011) show that a simple mechanical trading rule based on the MAs can yield favorable returns in trading futures contracts and the return correlation with the managed futures index exceeds 70%. Olszweski and Zhou (2014) show that combining both technicals and macro/fundamentals offers a significant improvement in risk-adjusted returns.

## **Related studies**

Recently, Moskowitz, Ooi and Pedersen (2012) provide evidence on momentum across asset classes, that is, the past values have predictive power. However, Huang, Li, Wang and Zhou (2020) argue that, while asset classes may have predictability, the predictability is not simply fixed at the 12-month horizon with the past 12-month return as the sole predictor. They show that asset-by-asset time-series regressions reveal little evidence of 12-month momentum, both in- and out-of-sample. From an investment perspective, a strategy of using the 12-month momentum has similar performance with a strategy that is based on historical sample mean and does not require predictability. In other words, while Huang, Li, Wang and Zhou (2020) do not rule out the predictive power of past returns on future values for a wide range of assets, their study merely points out that the predictability can be much more complex than using just the past 12-month return.

Going beyond technical indicators, Filippou, Rapach, Taylor and Zhou (2020) show that the currency market is predictable with country characteristics, global variables, and their interactions, and the predictability yields sizable carry trade profits. For the corporate bond market, Guo, Lin, Wu, and Zhou (2020) provide the first predictability evidence across bond rating along with a survey of the literature.

In summary, technical analysis appears useful not only in the stock market, but also valuable across asset classes. However, as it is in the stock market, the predictability is small and tends to decline over time.

## **Machine Learning**

Many machine learning techniques have been explored for stock price direction prediction (Ballings et al. 2015). ANN and Support Vector Regression (SVR) are two widely used machine learning algorithms for predicting stock price and stock market index values (Patel et al. 2015). A literature survey of supervised and unsupervised machine learning methods applied in stock market analysis will be presented next.

## Supervised Learning

Supervised learning techniques like Support Vector Machine (SVM) and Decision Trees can learn to predict stock market prices and trends based on historical data and provide meaningful analysis of historical price. Bernal et al. (2012) implemented a subclass of Recurrent Neural Networks (RNN) known as Echo State Networks (ESN) to predict S&P 500 stock prices using price, moving averages, and volume as features. The technique outperforms the Kalman Filter technique with a meagre test error of 0.0027. In order to generalize and validate their result, Bernal et al. (2012) examined the algorithm on 50 other stocks and reported that their results performed well against state of the art techniques.

Ballings et al. (2015) benchmark ensemble methods consisting of Random Forest, AdaBoost, and Kernel Factory against single classifier models such as Neural Networks, Logistic Regression, Support Vector Machines, and K-Nearest Neighbor using data from 5767 publicly listed European companies. The authors used five times two-fold cross-validation and Area Under the Curve (AUC) as a performance measure for predict long term stock price direction and reported Random Forest as the top algorithm.

Milosevic (2016) proposed an approach for long term prediction of stock market prices through a classification task where a stock is 'good' if the stock price increases by 10% in a year otherwise it is a 'bad'. Furthermore, Milosevic (2016) performed a manual feature selection, selected 11 relevant fundamental ratios, and applied several machine learning algorithms to stock prediction. It follows that Random Forest achieved the best F-Score of 0.751 against techniques such as SVM and Naïve Bayes.

Another technique that has grappled the attention of data scientists is the eXtreme Gradient Boosting (XGBoost). Dey et al. (2016) predicted the direction of stocks based on XGBoost algorithm using the technical indicators as the features. The results show that XGBoost beats the other techniques in performance achieving an accuracy of 87–99% for long term prediction of Apple and Yahoo stocks.

Long Short-Term Memory (LSTM) network have shown a lot of promises for time series prediction; Di Persio and Honchar et al. (2017) applied three different Recurrent Neural Network models namely a basic RNN, the LSTM, and the Gated Recurrent Unit (GRU) on Google stock price to evaluate which variant of RNN performs better. It was evident from the results that the LSTM outperformed other variants with a 72% accuracy on a five-day horizon and the authors also explained and displayed the hidden dynamics of RNN. Roondiwala et al. (2017) implemented an LSTM network to predict Nifty prices with features like OHLC. Their results show that the LSTM achieves an RMSE of 0.00859 for the test data in terms of daily percentage changes.

Yang et al. (2017) proposed an ensemble of multi-layer feedforward networks for Chinese stock prediction. Three component networks were trained using training algorithms like backpropagation and Adam. The ensemble was formed using the bagging approach (Efron and Tibshirani 1994). The results obtained demonstrate that the Chinese markets are partially predictable and achieve a satisfactory accuracy, precision, and recall.

Zhang et al. (2018) propose a stock price trend prediction system that can predict both stock price movement and its interval of growth (or decline) rate within predefined prediction durations. They trained a random forest model from historical data from the Shenzhen Growth Enterprise Market (China) to classify multiple clips of stocks into four main classes (up, down, flat, and unknown) according to the shapes of their close prices. Their evaluation shows that the proposed system is robust to the market volatility and outperforms some existing predictions methods in terms of accuracy and return per trade.

Hossain et al. (2018) propose a deep learning-based hybrid model that consists of two well-known DNN architectures: LSTM and GRU. The authors trained a prediction model using S&P 500 time series dataset spanning about 66 years (1950 to 2016). The approach involves passing the input data to the LSTM network to generate a first level prediction and then passing the output of LSTM layer to the GRU layer to get the final prediction. The proposed network achieved a Mean Squared Error (MSE) of 0.00098 in prediction with outperforming previous neural network approaches.



Recently, Lv et al. (2019) synthetically evaluated various ML algorithms and observed the daily trading performance of stocks under transaction cost and no transaction cost. They utilized 424 S&P 500 index component stocks (SPICS) and 185 CSI 300 Index Component Stocks (CSICS) between 2010 and 2017 and compared traditional machine learning algorithms with advanced deep neural network (DNN) models. The traditional machine learning algorithms are SVM, Random Forest, Logistic Regression, naïve Bayes, Classification and Regression Tree (CART), and eXtreme Gradient Boosting while the DNN architectures include Multilayer Perceptron (MLP), Deep Belief Network (DBN), Stacked Autoencoders (SAE), RNN, LSTM, and GRU. Their results show that traditional machine learning algorithms have a better performance in most of the directional evaluation indicators without considering the transaction cost, however, DNN models show better performance considering transaction cost.

## Unsupervised Learning

Unsupervised learning helps to identify correlations in an uncorrelated dataset like stock markets. Powell et al. (2008) drew a comparison between the supervised technique SVM and unsupervised technique K-means. They perform Principal Component Analysis (PCA) to reduce the dimensions or features. Both models are tested on S&P 500 data and the results show that both techniques have similar performance, SVM achieves 89.1% and K-means achieves 85.6% respectively. They also state how different distance measures for clustering affect the prediction accuracy and the best performance is shown by the Canberra distance metric.

Babu et al. (2012) proposed a clustering method called the HAK by combining the Hierarchical Agglomerative Clustering (HAC) and the reverse K-means clustering to predict the short-term impact on stocks after the release of financial reports. The study compared three different clustering techniques namely the HAC, K-means, and the reverse K-means. It also compared the proposed HRK against the three techniques and the SVM. Firstly, HAK takes financial reports and stock quotes as input and uses text analysis to convert each financial report into a feature vector. It then divides the feature vectors into clusters using HAC. Secondly, for each cluster, the K-means clustering method was applied to partition each cluster into sub-clusters and for each sub-cluster the centroids were computed. Finally, the centroids were used as the representative feature vectors to predict stock price movements. The experimental results show that the proposed technique outperforms SVM in terms of accuracy.

Wu et al. (2014) proposed a model based on the AprioriAll algorithm (association rule learning) and K-means. They converted stock data into charts using a sliding window and then the charts were clustered using K-means to extract chart patterns. Frequent patterns were extracted using AprioriAll to predict trends that are often associated (bought or sold together). The results show that their model outperforms other related work (Wang and Chan 2007; Chen 2011) in terms of average returns and also mutual funds.

Peachavanish (2016) proposes a clustering method to identify a group of stocks with the best trend and momentum characteristics at a given time, and therefore are most likely to outperform the market during a short time period. The author conducted an experiment on five-year historical price data of stocks listed on the Stock Exchange of Thailand (SET) and reported that the proposed method can outperform the market in the long run. Table 1 presents a summary of the existing literature on supervised ML approaches.



# Predicting Stock with Python

In this report of python for stock market, we will discuss two ways to predict stock with Python- Support Vector Regression (SVR) and Linear Regression.

## Support Vector Regression (SVR)

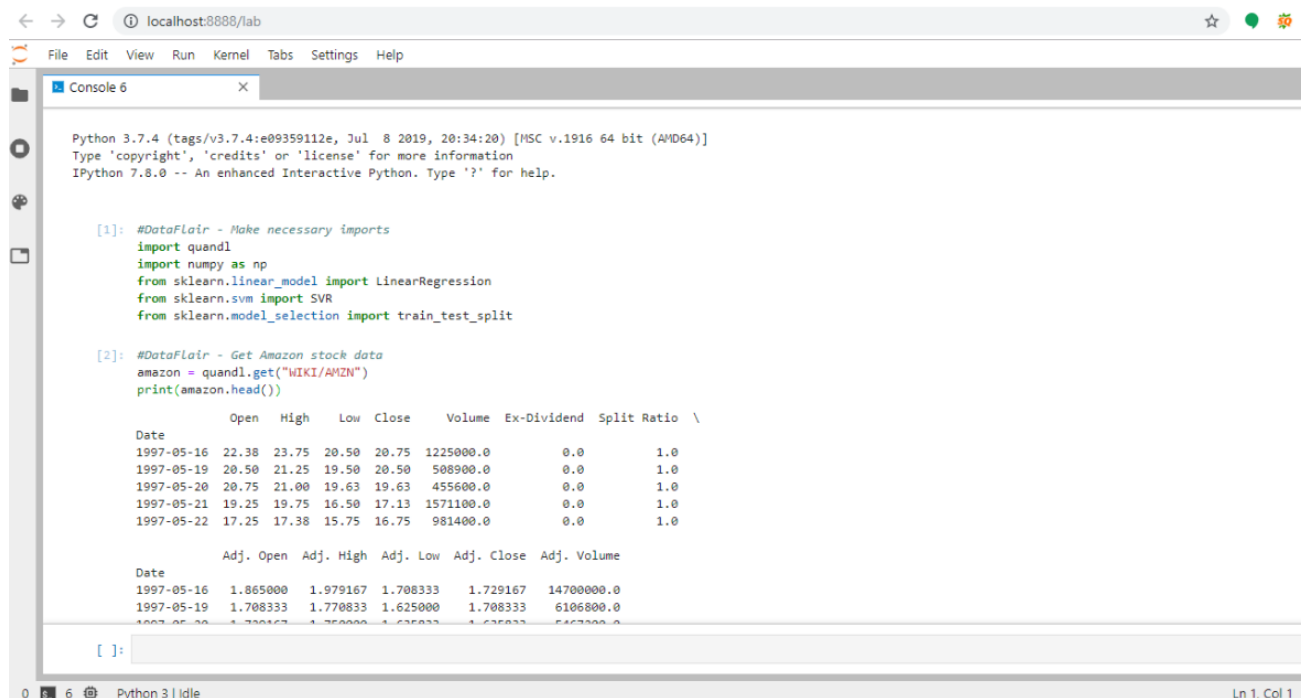
Support Vector Regression (SVR) is a kind of Support Vector Machine (SVM). It is a supervised learning algorithm which analyzes data for regression analysis. This was invented in 1996 by Christopher Burges et al. The cost function for building a model with SVR ignores training data close to the prediction model, so the model produced depends on only a subset of the training data.

SVMs are effective in high-dimensional spaces, with clear margin of separation and where the number of samples is less than the number of dimensions. However, they don't perform so well with large or noisy datasets.

## Linear Regression

Linear Regression linearly models the relationship between a dependent variable and one or more independent variables. This is simple to implement and is used for predicting numeric values. But this is prone to overfitting and can't be used where there's a non-linear relationship between dependent and independent variables.

## Code for Stock Prices Prediction



```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 20:34:20) [MSC v.1916 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.8.0 -- An enhanced Interactive Python. Type '?' for help.

[1]: #DataFlair - Make necessary imports
import quandl
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn.model_selection import train_test_split

[2]: #DataFlair - Get Amazon stock data
amazon = quandl.get("WIKI/AMZN")
print(amazon.head())
```

Date	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio \
1997-05-16	22.38	23.75	20.50	20.75	1225000.0	0.0	1.0
1997-05-19	20.50	21.25	19.50	20.50	508900.0	0.0	1.0
1997-05-20	20.75	21.00	19.63	19.63	455600.0	0.0	1.0
1997-05-21	19.25	19.75	16.50	17.13	1571100.0	0.0	1.0
1997-05-22	17.25	17.38	15.75	16.75	981400.0	0.0	1.0

Date	Adj. Open	Adj. High	Adj. Low	Adj. Close	Adj. Volume
1997-05-16	1.865000	1.979167	1.708333	1.729167	14700000.0
1997-05-19	1.708333	1.770833	1.625000	1.708333	6106800.0
1997-05-20	1.708333	1.750000	1.625000	1.625000	5467200.0

```
[ ]:
```

1. We will use the `quandl` package for the stock data for Amazon. Quandl indexes millions of numerical datasets across the world and extracts its most recent version for you. It cleans the dataset and lets you take it in whatever format you want.

```
1. #DataFlair - Make necessary imports
2. import quandl
3. import numpy as np
4. from sklearn.linear_model import LinearRegression
5. from sklearn.svm import SVR
6. from sklearn.model_selection import train_test_split
```

2. Get the Amazon stock data from `quandl`. Print the top 5 rows.

```
1. #DataFlair - Get Amazon stock data
2. amazon = quandl.get("WIKI/AMZN")
3. print(amazon.head())
```

3. Now get only the data for the Adjusted Close column. Print the first 5 rows for this.

```
1. #DataFlair - Get only the data for the Adjusted Close column
2. amazon = amazon[['Adj. Close']]
3. print(amazon.head())
```

4. Set the forecast length to 30 days. Create a new column 'Predicted' - this should have the data of the Adj. Close column shifted up by 30 rows. The last 5 rows will have NaN values for this column.

```
1. #DataFlair - Predict for 30 days; Predicted has the data of Adj. Close shifted up by 30 rows
2. forecast_len=30
3. amazon['Predicted'] = amazon[['Adj. Close']].shift(-forecast_len)
4. print(amazon.tail())
```

5. Now, drop the predicted column and create a NumPy array from it, call it 'x'. This is the independent dataset. Remove the last 30 rows and print x.

```
1. #DataFlair - Drop the Predicted column, turn it into a NumPy array to create dataset
2. x=np.array(amazon.drop(['Predicted'],1))
3. #DataFlair - Remove last 30 rows
4. x=x[:-forecast_len]
5. print(x)
```

6. Create a dependent dataset y and remove the last 30 rows. Print it then.

```
1. #DataFlair - Create dependent dataset for predicted values, remove the last 30 rows
2. y=np.array(amazon['Predicted'])
3. y=y[:-forecast_len]
4. print(y)
```

7. Split the datasets into training and testing sets. Keep 80% for training.

```
1. #DataFlair - Split datasets into training and test sets (80% and 20%)
2. x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

8. Create an SVR model now and train it.

```
1. #DataFlair - Create SVR model and train it
2. svr_rbf=SVR(kernel='rbf',C=1e3,gamma=0.1)
3. svr_rbf.fit(x_train,y_train)
```

9. Get the score of this model and print it in percentage.

```
1. #DataFlair - Get score
2. svr_rbf_confidence=svr_rbf.score(x_test,y_test)
3. print(f"SVR Confidence: {round(svr_rbf_confidence*100,2)}%")
```

10. Now, create a model for Linear Regression and train it.

```
1. #DataFlair - Create Linear Regression model and train it
2. lr=LinearRegression()
3. lr.fit(x_train,y_train)
```

11. Get the score for this model and print it in percentage.

```
1. #DataFlair - Get score for Linear Regression
2. lr_confidence=lr.score(x_test,y_test)
3. print(f"Linear Regression Confidence: {round(lr_confidence*100,2)}%")
```

## Python for Stock market

Let's look at the analytical capabilities of Stocker in parts.

### Starting with Stocker

The first thing that should be done is importing the Stocker class into the current python session after installing the required libraries. You can use it to create an object. The constructed object will contain all the properties of the Stocker class. As the stocker is built on quandl WIKI database hence it allows access to 3000 and more US stocks.

Python classes are comprised of – attributes and methods. Amongst all the attributes of the class, one of it is stock data for a specific company.

The benefits of using the Python class include – the functions and the data it acts on are associated with the same object. The entire history of the stock can be plotted by using the method of the Stocker object. The ‘plot\_stock’ function has a number of arguments that are optional and by default, it plots the adjusted closing price for the entire date range that can also be customized according to our needs (range, stats to be plotted, type of plot). Using ‘plot(stock)’ we can investigate any number of quantities in the data present in any data range and also suggest real-world correlations.

## **Additive tools**

These are very powerful for analyzing and predicting time series. We know that the long term trend of any established multinational company seems to be increasing in nature but there is a possibility of identifying yearly or daily basis patterns. Such help of time series with daily observations can be provided by Prophet, developed by Facebook. Stocker can do all the work that be done by Prophet behind the scenes using simple method call to create and inspect the model.

These types of models remove disturbance present in data and smoothen it. Prophet models also look into fluctuations of data in real-life processes and make predictions for the future. Though there is concern related to past data but future data analysis is what companies strive for. This method call returns two objects (data and model) which are then assigned to variables that are later on used to plot time series components.

## **Changepoints**

It occurs when the time-series go from increasing to decreasing or vice-versa. These patterns are also very important as one needs to know when the stock rate is at its peak or there are significant economic benefits. Identifying these points and their cause of change helps in predicting the future. The stocker object can automatically predict the 10 largest changepoints which tend to line up near the peaks and valleys of the stock price graph (generally). On the other hand, the prophet can only find changepoints in the first 80% data only. Google search tools allow us to see the popularity of any search word over time in Google searches. Stocker can automatically retrieve this data for any specific term.

## **Predictions**

They are designed for forecasting, or predicting future prices. This is a tiresome exercise and hence needs plenty of learning to get into the actual process. The capabilities are publically available, even creating the tool itself.

## Challenges and Open Problems

Stock market analysis and prediction continue to be an interesting and challenging problem. As more data are becoming available, we face new challenges in acquiring and processing the data to extract knowledge and analyse the effect on stock prices. These challenges include issues of live testing, algorithmic trading, self-defeating, long-term predictions, and sentiment analysis on company filings.

Regarding live testing, most of the literature on stock analysis and prediction claim that the proposed techniques can be used in real time to make profits in the stock market. It is a big claim to make because an algorithm may work fine on backtesting in controlled environments, but the main challenge is live testing, because a lot of factors like price variations, and uneventful news and noise exist. One such example is the Knight Capital Tragedy<sup>1</sup> where the company suffered a loss of 440 million. Hence, a viable research direction would be to understand how some of the popular stock analysis techniques work in live or simulated environments.

Algorithmic trading systems have changed the way stock markets function. Most of the trading volumes in equity futures are generated by algorithms and not by humans. While algorithmic trading gives benefits like reduced cost, reduced latency, and no dependence on sentiments, it also brings up challenges for retail investors who do not have the necessary technology to build such systems. Today, it is common to see events where panic selling is triggered due to these systems and hence the markets overreact. As a result, it becomes more difficult to evaluate market behaviour. With new algorithms continuing to flood the markets every day, comparison of the efficacy and accuracy of these algorithms pose yet another challenge.

An interesting aspect of this research area on stock market prediction is its self-defeating nature. In simple words, if an algorithm can use a novel approach to generate high profits, then sharing it in any way to the market participants will render the novel approach useless. Thus, state of the art algorithms which are trading out there in the markets is proprietary and confidential. The research or methodology behind such algorithms is generally never published.

Researchers, analysts, and traders mostly focus on short term prediction of stock prices compared to longer term, i.e., weekly or monthly predictions based on historical data. Some good approaches to long term price prediction already exist such as the ARIMA. Stock markets are generally more predictable in the longer term. Several newer ANN approaches such as the LSTM and RNN are now being explored and compared against existing approaches in predicting long term dependencies in the data and the stock prices, which are equally valuable to the investors and data scientists.

Recently, due to the rising influence of social media on many aspects of our lives, a lot of attention is being given to sentiment analysis based on Twitter or news data. Social media data can be unreliable and difficult to process, and fake news is being posted on the web by multiple sources. A good alternative to these or additional resource would be the quarterly or annual reports filed by the companies (e.g., 10-Q and 10-K) for stock prediction to apply sentiment analysis. These documents, if decoded correctly, give a major insight into a company's status, which can help to understand the future trend of the stock.

## **Conclusions**

Financial markets provide a unique platform for trading and investing, where trades can be executed from any device that can connect to the Internet. With the advent of stock markets, people have the opportunity to have multiple avenues to make their investment grow. Not only that, but it also gave rise to different types of funds like mutual funds, hedge funds and index funds for people and institutions to invest money according to their risk appetite. Governments of most countries invest a part of their healthcare, employment, or retirement funds into stock markets to achieve better returns for everyone. Online trading services have already revolutionised the way people buy and sell stocks. The financial markets have evolved rapidly into a strong and interconnected global marketplace. These advancements bring forth new opportunities and the data science techniques offer many advantages, but they also pose a whole set of new challenges. In this paper, we propose a taxonomy of computational approaches to stock market analysis and prediction, present a detailed literature study of the state-of-the-art algorithms and methods that are commonly applied to stock market prediction, and discuss some of the continuing challenges in this area that require more attention and provide opportunities for future development and research. Unlike traditional systems, stock market today are built using a combination of different technologies, such as machine learning, expert systems, and big data which communicate with one another to facilitate more informed decisions. At the same time, global user connectivity on the internet has rendered the stock market susceptible to customer sentiments, less stable due to developing news, and prone to malicious attacks. This is where further research can play an important role in paving the way how stock markets will be analysed and made more robust in the future. A promising research direction is to explore various algorithms to evaluate whether they are powerful enough to predict for the longer term, because markets act like weighing machines in the long run having less noise and more predictability. Hybrid approaches that combine statistical and machine learning techniques will probably prove to be more useful for stock prediction.

## BIBLIOGRAPHY

<https://projecttunnel.com/search.php?category=&s=stock+analysis+>  
[https://github.com/pydata/pandas-datareader/tree/main/pandas\\_datareader](https://github.com/pydata/pandas-datareader/tree/main/pandas_datareader)  
<https://www.analyticsvidhya.com/blog/2021/07/stock-prices-analysis-with-python/>  
<https://www.investopedia.com/terms/s/stock-analysis.asp>  
<https://levelup.gitconnected.com/stock-market-analysis-using-python-pandas-ec278f76e217>  
<https://data-flair.training/blogs/python-for-stock-market/>  
[https://www.researchgate.net/publication/333409685\\_Stock\\_Market\\_Analysis\\_A\\_Review\\_and\\_Taxonomy\\_of\\_Prediction\\_Techniques](https://www.researchgate.net/publication/333409685_Stock_Market_Analysis_A_Review_and_Taxonomy_of_Prediction_Techniques)  
<https://deliverypdf.ssrn.com/delivery.php?ID=517114114008124126094094083090088121042008063067056033068109126025116082111006080102027022096016013008018069092008066116087098028042091029065098114007070001104076029061043020019118116087070090122077092112008114087024006090095077119002118106021115068&EXT=pdf&INDEX=TRUE>  
<https://www.mdpi.com/2227-7072/7/2/26/htm>