الجـمهورية الجزائرية الديمقراطية الشعبية
Democratic People 's Republic of Algeria

وزارة التعليم الـعــالي و البحــث العـلمــي
Ministry of Higher Education and Scientific Research

جــامعة سعد دحلب البليدة
SAAD DAHLAB University of BLIDA

كلية التكنولوجيا
Faculty of Technology

قسم الإلكترونيـك
Department of Electronics

Master's Thesis

in

electronic embedded system

presented by

Mouadh Stambouli

Application of Machine Learning and Deep Learning in Stock Markets

Proposed by : Prof.Bougherira Hamida

2017-2018

Acknowledgements

I would firstly like to express my sincerest gratitude to my supervisor professor Bougherira Hamida , who has taken much time to meet, discuss and give feedback , I would like to thank her for her help and guidance throughout this work .
I am also immensely thankful for our professors for their guidance and support .
I also wish to thank my friends that were always available and helpful .
And I am very grateful to my family for everything .

ملخص:

تهدف هذه المذكرة إلى دراسة إمكانية توقع أسعار الأسواق المالية باستخدام التعلم الآلي وخوارزميات التعلم العميق.
يتم ذلك بمقارنة العديد من خوارزميات ML و DL وأدائها.
من أجل تحليل جودة النماذج المستخدمة ، تمت مقارنة النتائج النهائية مع خوارزميات أخرى من خلال تطبيق اختبارات الإحصاء ذات الأهمية. و يتم أخيرا عرض ومناقشة تحليل لجودة نتائج الخوارزميات المختلفة.

Résumé

Cette thèse vise à étudier la possibilité de prédire les cours boursiers à l'aide d'algorithmes d'apprentissage automatique et d'apprentissage en profondeur. Il compare de nombreux algorithmes ML et DL et leurs performances.

Afin d'analyser la qualité des modèles utilisés, les résultats finaux ont été comparés avec d'autres algorithmes de ML par l'application de tests statistiques d'importance. Une analyse de la qualité des résultats des différents algorithmes est présentée et discutée.

Abstract :

This Master’s thesis aims to study the possibility to predict stock markets prices using machine learning and deep learning algorithms . It compares many ML and DL algorithms and their performance .

In order to analyze the quality of the models used , the final results were compared with other Machine Learning algorithms through the application of significance statistical tests. An analysis of the quality of the results of the different algorithms is presented and discussed.

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
Algo	Algorithm
ANN	Artificial Neural Network
CAC	French stock market index
DAX	German stock index
DL	Deep Learning
EUR	European Union Currency
Forex	Foreign Exchange Market
FTSE	Financial Times Stock Exchange 100 Index largest 100 qualifying UK companies
HANG SENG	stock market index for the Hong Kong Stock Exchange
LIBOR	London Inter-bank Offered Rate (borrow rate)
ML	Machine Learning
NIKKEI	stock market index for the Tokyo Stock Exchange
NYSE	The New York Stock Exchange
NASDAQ	the National Association of Securities Dealers Automated Quotations ,US stock exchange
OHLC	Open, High, Low, Close
OOP	object oriented programming
USD	United State of America Currency
YEN	The official currency of Japan

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Introduction

Stock price time-series are often characterized by a chaotic and non-linear behavior which makes the forecast a challenging task. The factors that produces uncertainty in this field are complex and from different nature, from economic, political and investment decisions to unclear reasons that, somehow, produce effects and make hard to predict how the prices will evolve. The stock market attracts investments due to the ability of producing high revenues. However, owing to its risky nature, there is a need for an intelligent tool that minimizes risks and, hopefully, maximizes profits [11].

Background :

The main motivation of our project was to differ from trivial econometric analysis and present a novel perspective as to how financial time series may be studied.
This project aims to apply ML algorithms to one of the most challenging tasks of the financial sector: forecasting financial time-series. Financial agents can benefit from systems based on ML to planning and monitoring their financial investments more accurately and therefore achieving higher returns. The main goal of this work is the application of machine learning (ML) and some new advances in the field, to predict the price of the stock in the future.
In our project , we tried many machine learning and deep learning algorithms to build an environment that makes trading more predictable .

Generally embedded systems can be used as data sources. The data can be any thing like your room temperature, humidity, video, images, audio, lighting system status, or financial data . So an embedded system typically collects and provides the data required to run machine learning . Machine learning can run on these data sets to come out with meaningful information which can be used to trigger actions.

This thesis is divided into four chapters , introduction to financial markets , insights to ML and DL , Python and libraries used and different algorithms of ML and DL , experiments and conclusion. The financial markets chapter 1 gives a background about forex market .. The ML chapter 2 provides insight into machine learning and deep learning and explains how the algorithms were implemented. The python chapter 3 gives an overview about python and its libraries and explains the theory used in the experiments .The Conclusion chapter 4 presents the results and explains how to understand them, and discusses their implications , and attempts to draw some conclusions from the results.

Chapter 1 :

Financial markets Background

Machine learning (ML) and related methods have produced some of the financial industry’s most consistently profitable proprietary trading strategies during the past 20 years. With markets, trade execution and financial decision making becoming more automated and competitive, practitioners increasingly recognize the need for ML.

This chapter describes basic and key concepts of financial markets .Technical parameters and indicators important to our stock market prediction with AI project , are shown and defined .

1.1 _ Overview of Financial Market

Generally, financial market is defined as a marketplace where the process of economic exchange or trade is carried out between buyers and sellers, such as Foreign Exchange (Forex), stock exchange markets and others. The process involves the transfer of funds and financial assets between two or more parties Within the financial field, financial markets are commonly referred to as markets, which are used for finance operations, expansion and economic growth .

The financial market plays an important role in promoting economic growth. The collection and analysis of financial time series data, as well as investment opportunities are provided to investors, brokers, corporations and other financial institutions. The collection of information assists them in directing funds for the most effective returns . The structure of financial market allows buyers and sellers to determine the price and value of financial claims or the desired rate of return on different types of financial assets. Furthermore, the financial market offers liquidity for investors through the ability to convert financial assets into liquid funds.

1.2 _ Investment Options

There are many options for investment . We will focus here on the Forex market. Other options are presented in Appendix A .

1.2.1_ FOREX

The following investment option is of major importance as it was the central focus of this project.

Each country has its own currency. Whenever one currency is exchanged with another it is a foreign exchange or Forex transaction. The foreign exchange market has experienced many changes since its inception. For many years the United States and its allies participated in a system under the Bretton Woods Agreement. Foreign exchange rates were tied to the amount of gold reserves belonging to the nation . However, in the summer of 1971, President Nixon took the United States off the gold standard. After this many countries followed and there was no relation between the gold reserves and the exchange rates . Floating exchange rates came into existence.

Forex investments are essentially trading in the future via options. An option gives you the right, but not an obligation, to buy or sell a specific amount or foreign exchange at a specified price within a specified period. Options are termed either call or put. A call gives the holder the right to buy the foreign currency or Forex at a specified price. The put gives the right to sell Forex at a specified price. Depending on the actual market price when you exercise the option you will gain/lose the difference between the specified price and the market price.

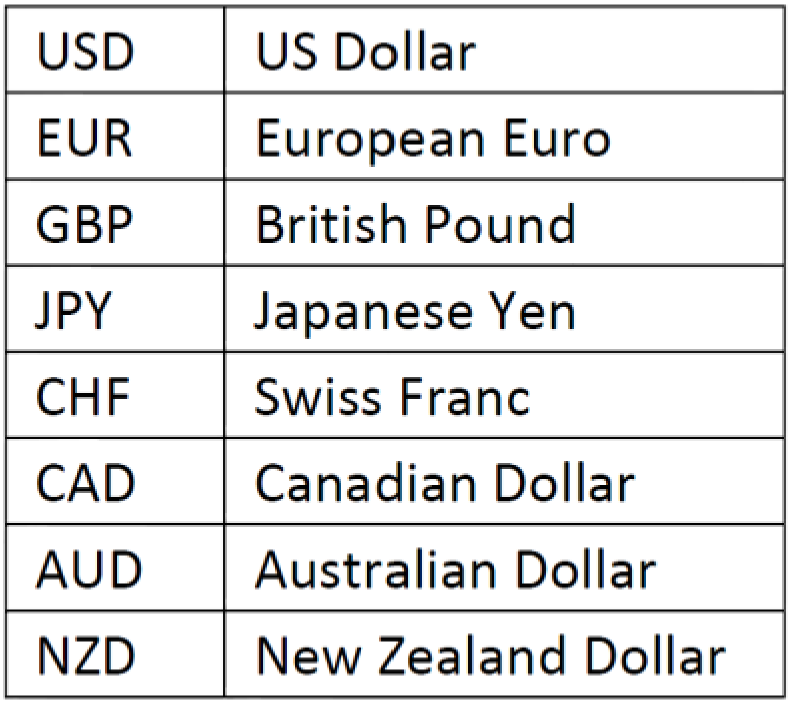
The question lies as to how people can trade in Forex Market and what underlying risks should be taken into account. Trading in the Forex market is executed through the brokerage houses that deal in foreign exchange. You can trade by options in the Forex market ; although the Forex market may go up or go down. If you expect the Forex market to go up you will invest in calls e.g. the value of 1 us dollar today is 100 Algerian Dinars . If you expect the Forex market will change to 1 us dollar equal to115 Algerian Dinars in four months then you can go invest call option; agreeing to buy 10 U.S. dollar at the rate of 105 Algerian Dinars / dollar at any time during the next six months. Whenever the actual market price is above 105 you can exercise the option. You can actually buy 10 U.S. dollar by paying only 105 DZD Per dollar i.e. by paying only 1050 DZD . But the actual value being 1100 DZD . Your gain is 50 DZD . Similarly , if you think the market is going to be 90 DZD per dollar, you can invest in a put option. In this case you will be able to sell the dollars in Forex market at the agreed price i.e. 100 DZD /dollar though the actual market price is less i.e. Only 90 DZD /dollar. The gain an investor makes will depend on the actual difference in the market price and the options price. However, there will be some fee/commission levied. This would be the cost of transaction, and result in a reduction of the gain, to some extent .

The foreign exchange market is the largest financial market in the world . Traditionally the foreign exchange market has only been available to banks, money managers and large financial institutions. Over the years, however, these institutions, including the U.S.Federal Reserve Bank, have realized large gains via currency trading . This growing market is now linked to a worldwide network of currency dealers , including banks, central banks , brokers , and customers such as importers and exporters. Today the foreign exchange market offers opportunities for profit not only to banks and institutions , but also to individual investors. More accurately it can be noted that large markets involve trading of $4 **trillion** every day. Market participants anticipate the direction of currency prices generate the bulk of currency activity .

In general , the value of currency vs. other currency (i.e. Exchange rate or foreign exchange rate) is a reflection of the condition of that country's economy with respect to the other major economies. George Soros, (in)famously, took a massive position against the British Pound in 1992 and virtually forced the U.K government out of the semi-fixed exchange rate mechanism with its EU partners. He made a fortune out of this transaction. You can lose money also. The quantum fund set up by George Soros produced remarkable annual compound returns of 30% between 1969 and 1987.
Nevertheless Forex trading presents many computational challenges that may be applicable for state of the art computer science techniques .

People invest in Forex markets because of the considerable opportunities they offer. An individual investor can realize huge profit potential by buying or selling a particular currency against the U.S. dollar or any other major currency. Investors can generate profits whether a currency is rising or falling . Buying one currency, which is expected to rise, against another currency can do this. Or you may sell one currency, which is expected to fall, against another currency. Taking a long position means buying a currency at one price and aiming to sell it later at a higher price. A short position is one in which the investor sells a currency that he hopes will fall and aims to buy it back later at a lower price. Depending on the risks that an investor is prepared to take the gains can be substantial. The style of George Soros was to take big, often interlinked speculative position using lots of leverage. It was possible to produce a 10% gain in the net worth of the fund that he was managing by means of a 1% move in the YEN .

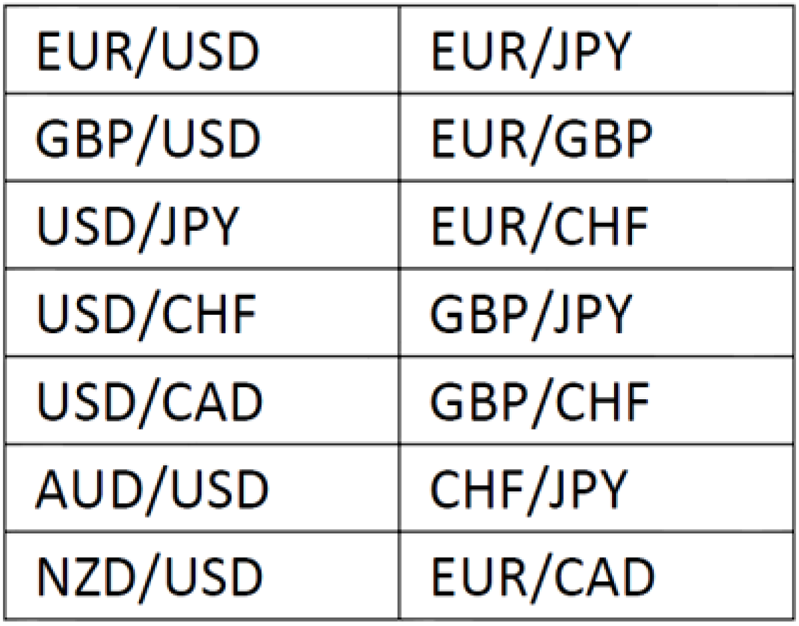
The eight most traded currencies of the FOREX Market are shown in Table 1



USD	US Dollar
EUR	European Euro
GBP	British Pound
JPY	Japanese Yen
CHF	Swiss Franc
CAD	Canadian Dollar
AUD	Australian Dollar
NZD	New Zealand Dollar

Table 1.1 Most traded currencies

Forex trading is always done in pairs, since any trade involves the simultaneous buying of a currency and selling of another currency. The trading revolves around 14 main currency pairs. These pairs are shown in Table 2



EUR/USD	EUR/JPY
GBP/USD	EUR/GBP
USD/JPY	EUR/CHF
USD/CHF	GBP/JPY
USD/CAD	GBP/CHF
AUD/USD	CHF/JPY
NZD/USD	EUR/CAD

Table 1.2 Currency Pairs

1.2.1.1 _ Stock Chart

Stock chart is a graphical plot that represents a sequence of financial time series data over a set of time stamp which shown in Figure 1. In statistics, this chart is referred to as a time series plot. It includes the time scale, price scale and the patterns of financial trend. The stock chart usually shows the movement of prices over a period of time, where each point on the chart represents the prices they trade at. Furthermore, a graphical stock chart makes it easier to spot the movement direction of the financial trends and its performance over a period of time . This category reviews on two popular types of stock chart – candlestick pattern and OHLC bar chart that are used by investors and traders in financial circle today.

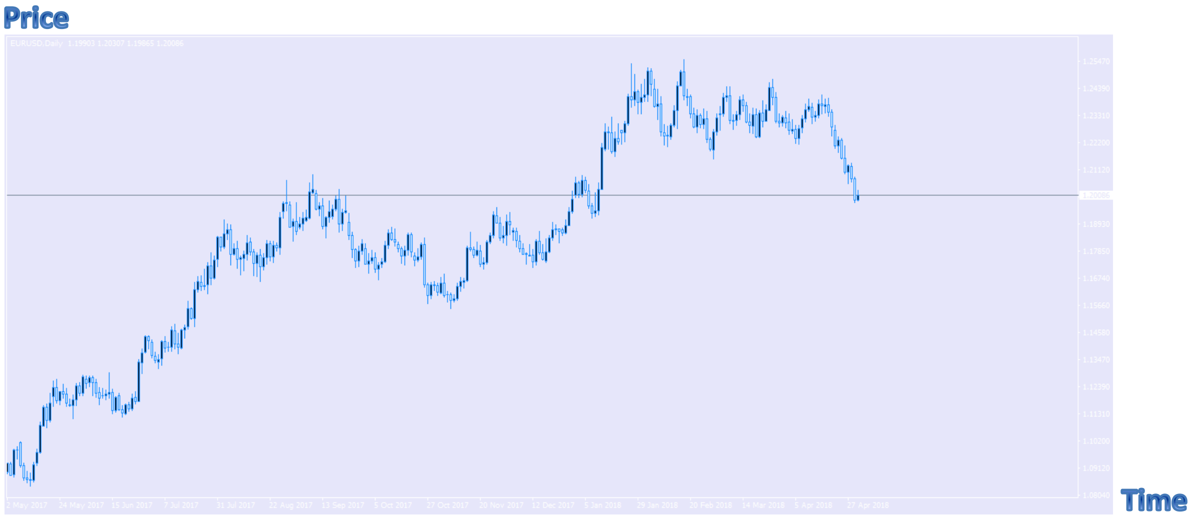


Figure 1.1 Example of Stock Chart

1.2.1.1.a- Candlestick Pattern

A Japanese rice trader Munehisa Homma [36] founded candlestick pattern in the 18th century. It is one of the most popular and oldest types of empirical prediction model which have been used for decision making in stock price, foreign exchange rates, commodity and trading .The theory of candlestick pattern assumes that the trend of financial time series could be predicted by identifying the patterns. It visualises the financial trend patterns and provides the signal of continuations and inversion about the nature of financial trends .

Candlestick pattern is composed of the thick body (black or white) and shadows. The thick part of candlestick body is called real body (RB), which represents the price range between close and open prices. The vertical lines above and bottom of the real body are called shadows. The shadow above the real body is called upper shadow (US) and the shadow under the real body is called lower shadow (LS), which represent the highest and the lowest prices of the time stamp.

Figure 2 shows the illustration of candlestick pattern where the real body of candlestick has two colours – “Black” and “White”. The “Black” real body illustrates that the open price is higher than the close price, which indicates the financial trend is decreasing, vice versa, when the close price is higher than the open price as the “White” real body representing the financial trend is increasing.

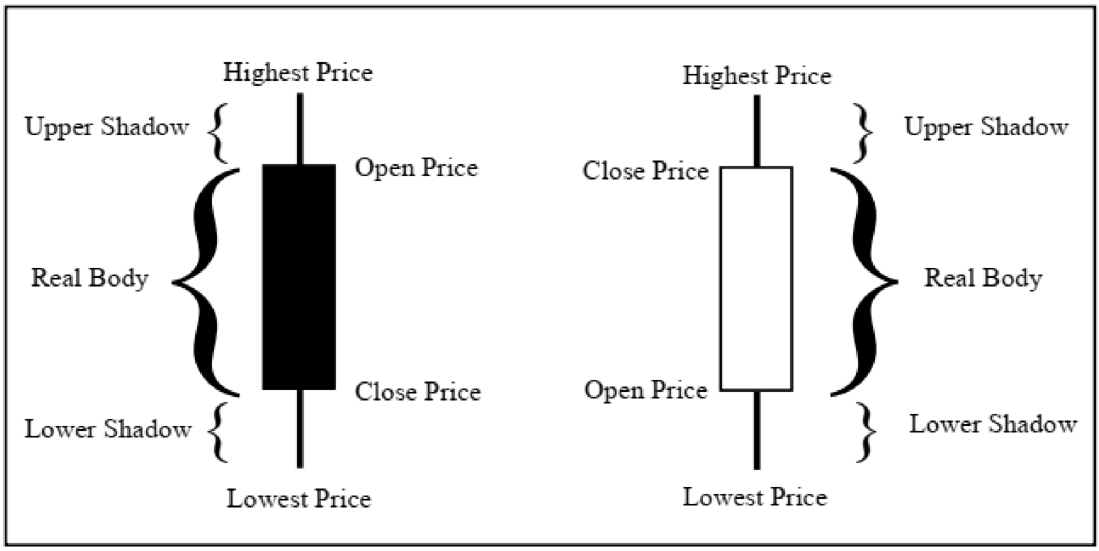


Figure 1.2 Structure of a Candlestick Pattern

1.2.1.1.b- OHLC Bar Chart

OHLC bar chart is another type of stock chart, which illustrates the direction of the financial trend. The bar chart includes four separate financial time series data price information:

- Open: Opening price of that current time stamp.
- High: The highest price of that current time stamp.
- Low: The lowest price of that current time stamp.
- Close: Closing price of that current time stamp.

OHLC chart shows the trend of the stock price in different time stamp. The horizontal line in Figure 3 represents the price range (open and close price) and the vertical line at the top and bottom represents the highest and lowest price within a time stamp, such as a day or an hour

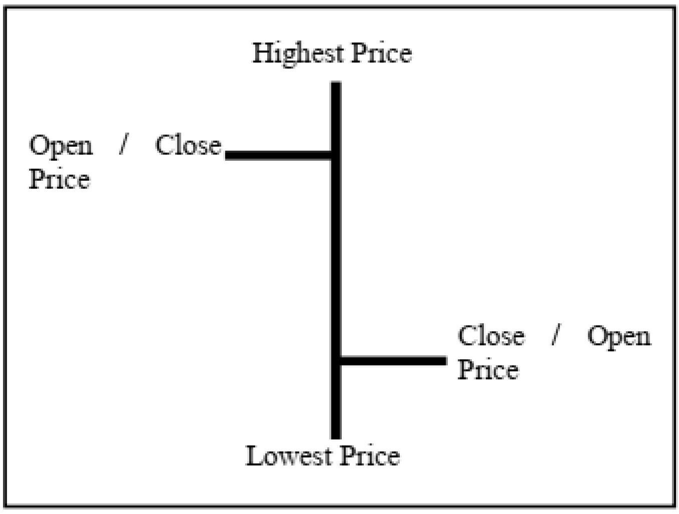


Figure 1.3 Structure of OHLC Bar Chart

1.2 .2- Technical Indicators

In the financial field, the technical indicators utilise a series of data point, which are derived by applying formulas to calculate the movement of financial time series over the specified time stamp. It uses to identify the future price levels, investigate the financial trend direction and security by looking at the historical information of stock .Technical indicators serve for three important functions – "to alert", "to confirm" and "to predict" [37] .

This category discusses certain technical indicators that include Relative Strength Index (RSI) , Momentum (MOM) ,and Moving Average Convergence-Divergence (MACD) .

1.2.2.1- Relative Strength Index (RSI)

The relative strength index (RSI) is a momentum indicator developed by noted technical analyst Welles Wilder, that compares the magnitude of recent gains and losses over a specified time period to measure speed and change of price movements of a security. It is primarily used to attempt to identify overbought or oversold conditions in the trading of an asset.

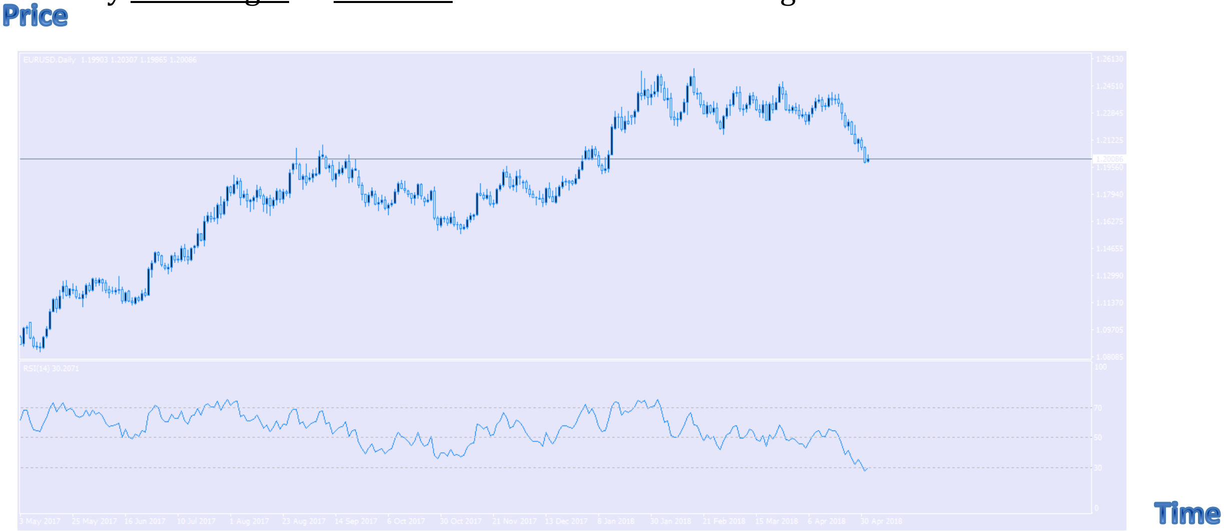


Figure 1.4 RSI Chart

1.2.2.2- Momentum (MOM)

The Momentum (MOM) indicator [39] compares the current price with the previous price from a selected number of periods ago. This indicator is similar to the "Rate of Change" indicator, but the MOM does not normalize the price, so different instruments can have different indicator values based on their point values.

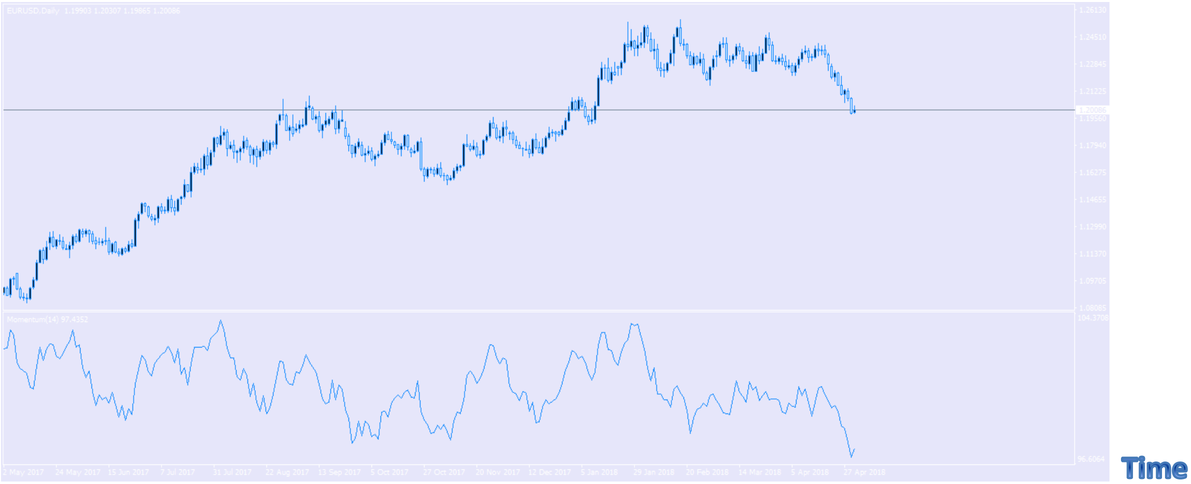


Figure 1.5 MOM Chart

1.2.2.3- Moving Average Convergence-Divergence (MACD)

Moving average convergence divergence (MACD) is a trend-following momentum indicator that shows the relationship between two moving averages of prices [38]. The MACD is calculated by subtracting the 26-day exponential moving average (EMA) from the 12-day EMA. A nine-day EMA of the MACD, called the "signal line", is then plotted on top of the MACD, functioning as a trigger for buy and sell signals.

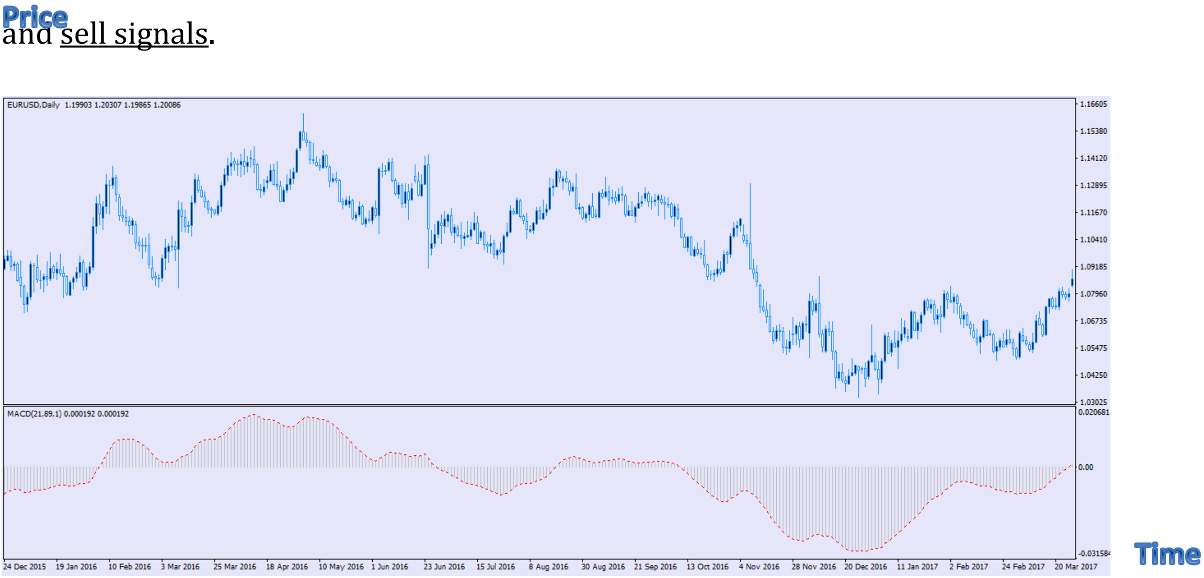


Figure 1. 6 MACD Chart

1.3 _ Summary

In this chapter we gave an insight in Investment options , especially forex market , and talked about predictability and factors that influence the markets , then we saw some composition of stock chart and some technical analysis .

And the inputs in our project are

- OpenHigh minus Low price
- Close price
- Three day moving average
- 20 day moving average
- 100 day moving average
- 200 day moving average

- Relative Strength Index RSI
- MACD

The next chapter introduces Machine Learning and Deep Learning algorithms and their application to Forex market prediction .

Chapter 2. An overview about Machine Learning

And

Deep Learning

Introduction

Because the aim of our project is the implementation of machine learning and deep learning for the prediction of stock markets, this chapter provides essential context around machine learning, and deep learning algorithms.

2-1. Artificial intelligence (AI) , machine learning, and deep learning

-First, we need to define clearly what we're talking about when we mention AI. What are artificial intelligence, machine learning, and deep learning (fig. 2.1)? How do they relate to each other?

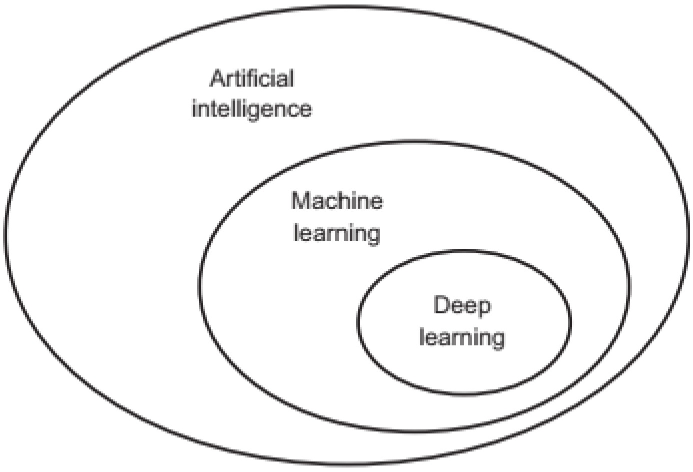


Figure 2-0 AI,ML, and DL

AI is the simulation of human intelligence processes by machines, like computer, embedded, industrial, biomedical, financial and many other systems . These processes include learning (the acquisition of information and rules for using the information), reasoning (using the rules to reach approximate or definite conclusions) and self-correction. Particular applications of AI include expert systems, speech recognition and machine vision, control, prediction and various other areas.

Thus machine learning is a subfield of AI and DL is a subfield of machine learning.

DL has been initially developed in 1998 by Lecun , but has been abandoned very quickly because at that time, processors could not provide the powerful processing required by the algorithm . By 2010, powerful processors appeared and allowed efficient processing for the DL algorithm. In fact, DL was so successful that it has boosted machine learning and artificial intelligence and brought them back to life.

2_2. Machine learning

Machine learning arises from this question: could a computer go beyond "what we know how to order it to perform" and learn on its own how to perform a specified task? . Could a computer surprise us? Rather than programmers crafting data-processing rules by hand, could a computer automatically learn these rules by looking at data?

This question opens the door to a new programming paradigm. In classical programming, the paradigm of symbolic AI, humans input rules (a program) and data to be processed according to these rules, and out come answers (see figure 8). With machine learning, humans input data as well as the answers expected from the data, and out come the rules. These rules can then be applied to new data to produce original answers.

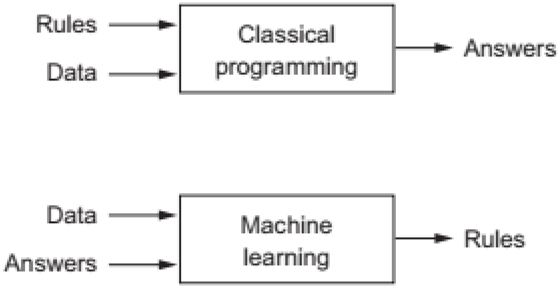


Figure 2.1 ML, a new programming paradigm

A machine-learning system is *trained* rather than explicitly programmed. It's presented with many examples relevant to a task, and it finds statistical structure

in these examples that eventually allows the system to come up with rules for automating the task.
For instance, if you wished to automate the task of tagging your vacation pictures, you could present a machine-learning system with many examples of pictures already tagged by humans, and the system would learn statistical rules for associating specific pictures to specific tags.

Although machine learning only started to flourish in the 1990s, it has quickly become the most popular and most successful subfield of AI, a trend driven by the availability of faster hardware and larger datasets. Machine learning is tightly related to mathematical statistics, but it differs from statistics in several important ways.
Unlike statistics, machine learning tends to deal with large, complex datasets (such as a dataset of millions of images, each consisting of tens of thousands of pixels) for which classical statistical analysis such as Bayesian analysis would be impractical. As a result, machine learning, and especially deep learning, exhibits comparatively little mathematical theory—maybe too little—and is engineering oriented. It's a hands-on discipline in which ideas are proven empirically more often than theoretically.

2_3. Machine Learning Algorithms

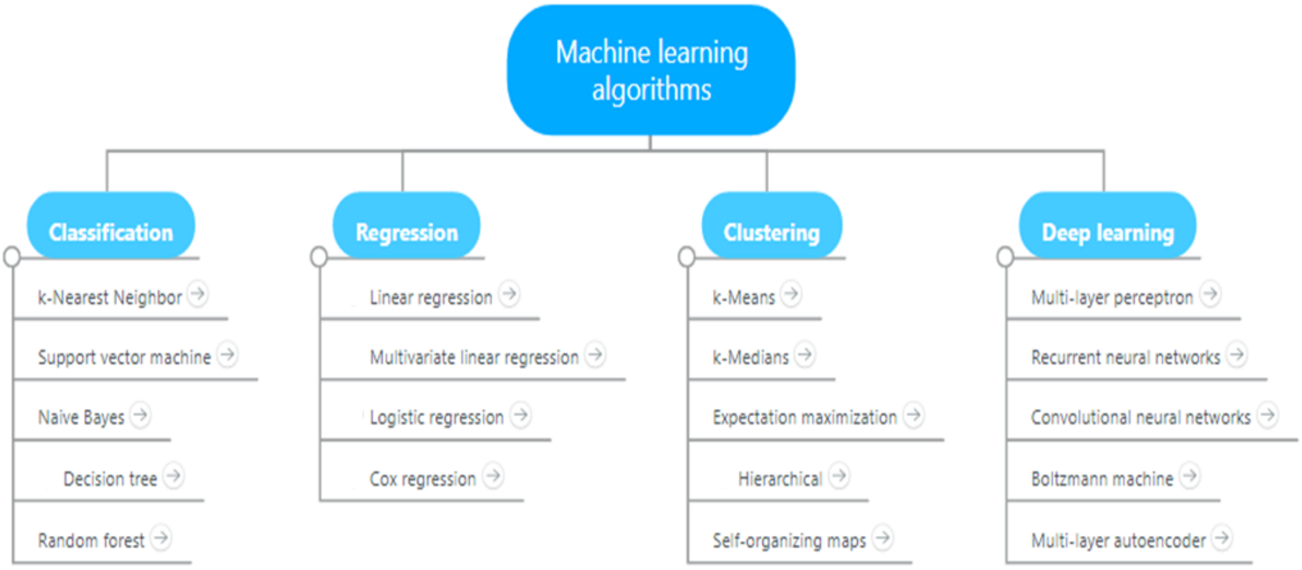


Figure 2.2 ML Algorithms

We take a tour of the most popular machine learning algorithms.

It is useful to tour the main algorithms in the field to get a feeling of what methods are available.

There are so many algorithms available that it can feel overwhelming when algorithm names are thrown around and you are expected to just know what they are and where they fit.

There are two ways to think about and categorize the algorithms you may come across in the field.

- The first is a grouping of algorithms by the **learning style**.
- The second is a grouping of algorithms by **similarity** in form or function (like grouping similar animals together).

2.3.1_ Algorithms Grouped by Learning Style

There are different ways an algorithm can model a problem based on its interaction with the experience or environment or whatever we want to call the input data.

It is popular in machine learning and artificial intelligence textbooks to first consider the learning styles that an algorithm can adopt.

There are only a few main learning styles or learning models that an algorithm can have and we'll go through them here with a few examples of algorithms and problem types that they suit.

This taxonomy or way of organizing machine learning algorithms is useful because it forces you to think about the roles of the input data and the model preparation process and select one that is the most appropriate for your problem in order to get the best result.

Let's take a look at three different learning styles in machine learning algorithms:

2.3.1.1- Supervised Learning

Input data is called training data and has a known label or result such as spam/not-spam or a stock price at a time.

A model is prepared through a training process in which it is required to make predictions and is corrected when those predictions are wrong. The training process continues until the model achieves a desired level of accuracy on the training data. Example problems are classification and regression.

Example algorithms include Logistic Regression and the Back Propagation Neural Network.

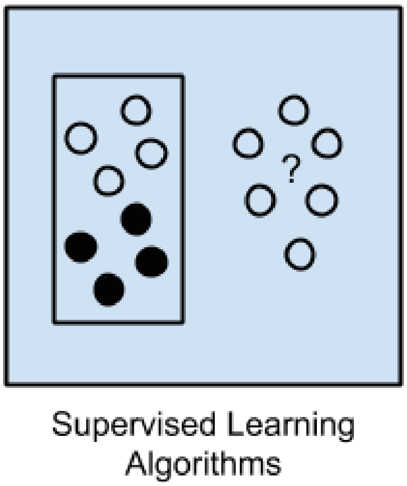


Figure 2.3 supervised learning algo

2.3.1.2. Unsupervised Learning

Input data is not labeled and does not have a known result.

A model is prepared by deducing structures present in the input data. This may be to extract general rules. It may be through a mathematical process to systematically reduce redundancy, or it may be to organize data by similarity.

Example problems are clustering, dimensionality reduction and association rule learning.

Example algorithms include: the Apriori algorithm and k-Means.

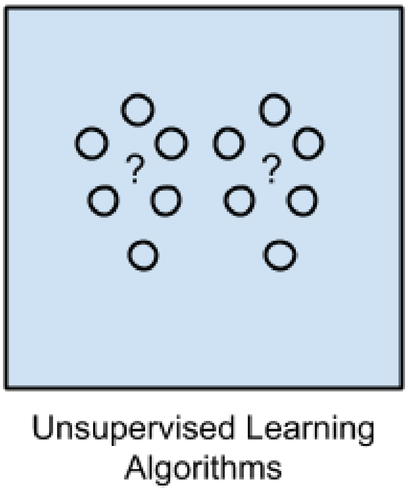


Figure 2.4 unsupervised learning algo

2.3.1.3. Semi-Supervised Learning

Input data is a mixture of labeled and unlabelled examples.

There is a desired prediction problem but the model must learn the structures to organize the data as well as make predictions.

Example problems are classification and regression.

Example algorithms are extensions to other flexible methods that make assumptions about how to model the unlabeled data.

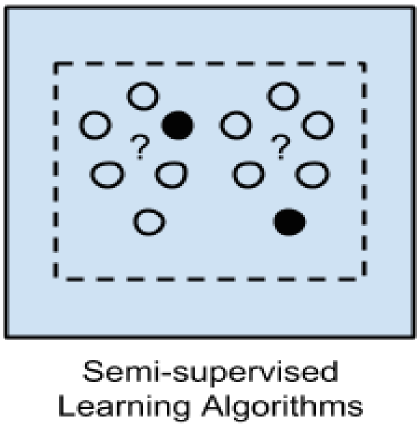


Figure 2.5 Semi supervised learning algo

*** Overview**

When crunching data to model business decisions, you are most typically using supervised and unsupervised learning methods.

A hot topic at the moment is semi-supervised learning methods in areas such as image classification where there are large datasets with very few labeled examples.

2.3.2_ Algorithms Grouped By Similarity

Algorithms are often grouped by similarity in terms of their function (how they work). For example, tree-based methods, and neural network inspired methods.

I think this is the most useful way to group algorithms and it is the approach we will use here.

This is a useful grouping method, but it is not perfect. There are still algorithms that could just as easily fit into multiple categories like Learning Vector Quantization that is both a neural network inspired method and an instance-based method. There are also categories that have the same name that describe the problem and the class of algorithm such as Regression and Clustering.

We could handle these cases by listing algorithms twice or by selecting the group that subjectively is the “best” fit. I like this latter approach of not duplicating algorithms to keep things simple.

In this section, I list many of the popular machine learning algorithms grouped the way I think is the most intuitive. The list is not exhaustive in either the groups or the algorithms, but I think it is representative and will be useful to you to get an idea of the lay of the land.

Note: There is a strong bias towards algorithms used for classification and regression, the two most prevalent supervised machine learning problems you will encounter.

2.3.2.1. *Regression Algorithms*

Regression is concerned with modeling the relationship between variables that is iteratively refined using a measure of error in the predictions made by the model.

Regression methods are a workhorse of statistics and have been co-opted into statistical machine learning. This may be confusing because we can use regression to refer to the class of problem and the class of algorithm. Really, regression is a process.

The most popular regression algorithms are:

- Ordinary Least Squares Regression (OLSR)
- Linear Regression
- Logistic Regression
- Stepwise Regression
- Multivariate Adaptive Regression Splines (MARS)
- Locally Estimated Scatterplot Smoothing (LOESS)

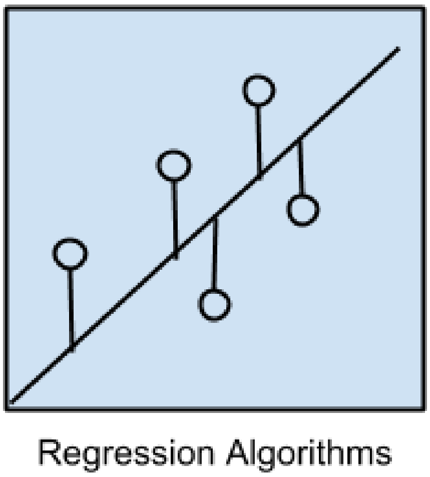


Figure 2.6 Regression algo

2.3.2.2.Instance-based Algorithms

Instance-based learning model is a decision problem with instances or examples of training data that are deemed important or required to the model.

Such methods typically build up a database of example data and compare new data to the database using a similarity measure in order to find the best match and make a prediction. For this reason, instance-based methods are also called winner-take-all methods and memory-based learning. Focus is put on the representation of the stored instances and similarity measures used between instances.

The most popular instance-based algorithms are:

- k-Nearest Neighbor (kNN)
- Learning Vector Quantization (LVQ)
- Self-Organizing Map (SOM)
- Locally Weighted Learning (LWL)

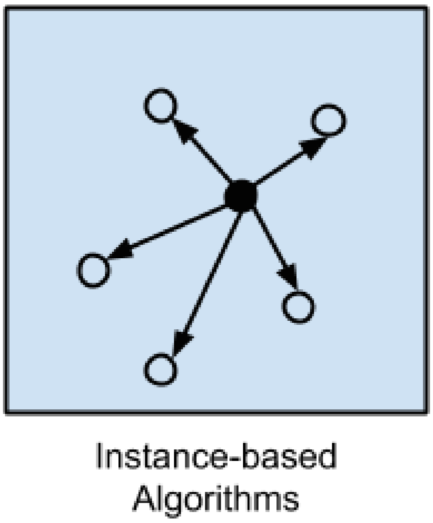


Figure 2.7 instance based algo

2.3.2.3.Regularization Algorithms

An extension made to another method (typically regression methods) that penalizes models based on their complexity, favoring simpler models that are also better at generalizing.

I have listed regularization algorithms separately here because they are popular, powerful and generally simple modifications made to other methods.

The most popular regularization algorithms are:

- Ridge Regression
- Least Absolute Shrinkage and Selection Operator (LASSO)
- Elastic Net
- Least-Angle Regression (LARS)

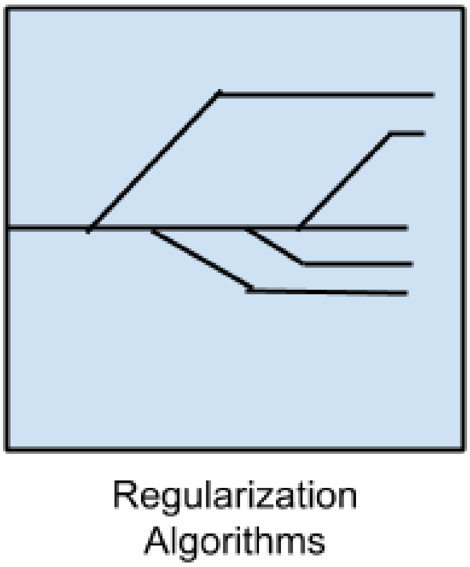


Figure 2.8 Regularization algo

2.3.2.4.Decision Tree Algorithms

Decision tree methods construct a model of decisions made based on actual values of attributes in the data.

Decisions fork in tree structures until a prediction decision is made for a given record. Decision trees are trained on data for classification and regression problems. Decision trees are often fast and accurate and a big favorite in machine learning.

The most popular decision tree algorithms are:

- Classification and Regression Tree (CART)
- Iterative Dichotomiser 3 (ID3)

- C4.5 and C5.0 (different versions of a powerful approach)
- Chi-squared Automatic Interaction Detection (CHAID)
- Decision Stump
- M5
- Conditional Decision Trees

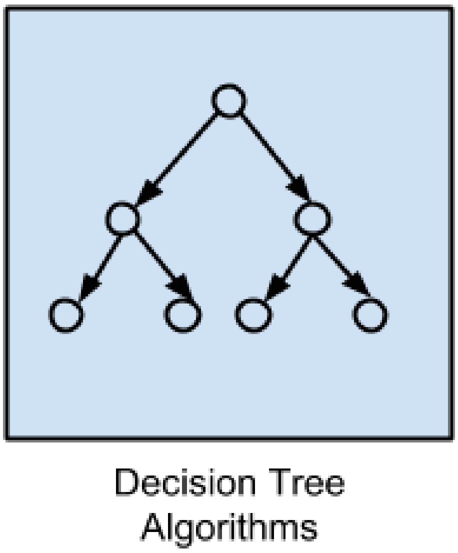


Figure 2.9 Decision tree algo

2.3.2.5.Bayesian Algorithms

Bayesian methods are those that explicitly apply Bayes' Theorem for problems such as classification and regression.

The most popular Bayesian algorithms are:

- Naive Bayes
- Gaussian Naive Bayes
- Multinomial Naive Bayes
- Averaged One-Dependence Estimators (AODE)
- Bayesian Belief Network (BBN)
- Bayesian Network (BN)

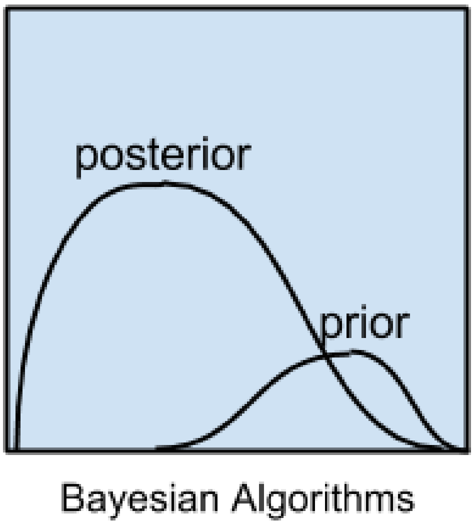


Figure 2.10 Bayesian algo

2.3.2.6.Clustering Algorithms

Clustering, like regression, describes the class of problem and the class of methods.

Clustering methods are typically organized by the modeling approaches such as centroid-based and hierarchal. All methods are concerned with using the inherent structures in the data to best organize the data into groups of maximum commonality.

The most popular clustering algorithms are:

- k-Means
- k-Medians
- Expectation Maximisation (EM)
- Hierarchical Clustering

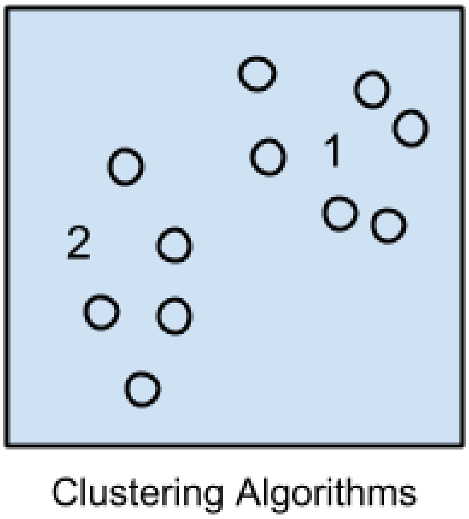


Figure 2.11 Clustering algo

2.3.2.7.Association Rule Learning Algorithms

Association rule learning methods extract rules that best explain observed relationships between variables in data.

These rules can discover important and commercially useful associations in large multidimensional datasets that can be exploited by an organization.

The most popular association rule learning algorithms are:

- Apriori algorithm
- Eclat algorithm

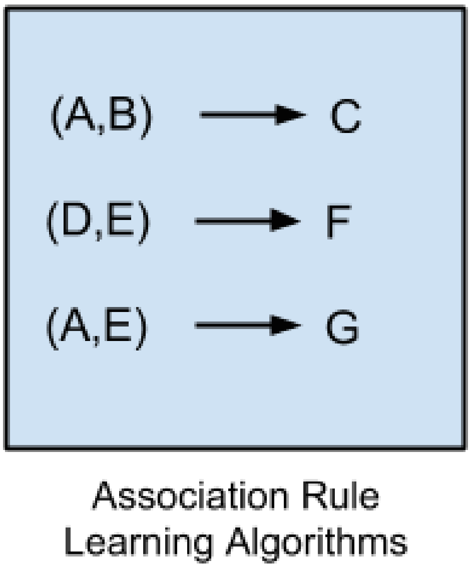


Figure 2.12 Association rule learning algo

2.3.2.8.Artificial Neural Network Algorithms

Artificial Neural Networks are models that are inspired by the structure and/or function of biological neural networks.

They are a class of pattern matching that are commonly used for regression and classification problems but are really an enormous subfield comprised of hundreds of algorithms and variations for all manner of problem types.

Note that I have separated out Deep Learning from neural networks because of the massive growth and popularity in the field. Here we are concerned with the more classical methods.

The most popular artificial neural network algorithms are:

- Perceptron
- Back-Propagation
- Hopfield Network
- Radial Basis Function Network (RBFN)

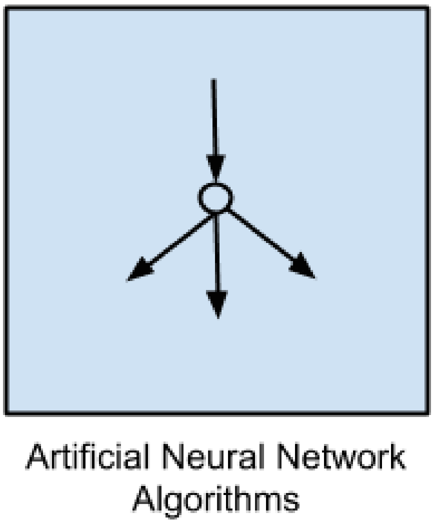


Figure 2.13 Artificial NN algo

2.3.2.9.Deep Learning Algorithms

Deep Learning methods are a modern update to Artificial Neural Networks that exploit abundant cheap computation.

They are concerned with building much larger and more complex neural networks and, as commented on above, many methods are concerned with semi-supervised learning problems where large datasets contain very little labeled data.

The most popular deep learning algorithms are:

- Deep Boltzmann Machine (DBM)
- Deep Belief Networks (DBN)
- Convolutional Neural Network (CNN)
- Stacked Auto-Encoders

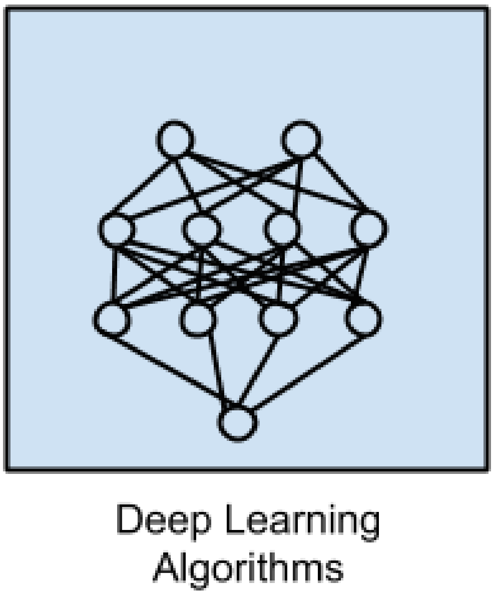


Figure 2.14 Deep learning algo

2.3.2.10.Dimensionality Reduction Algorithms

Like clustering methods, dimensionality reduction seek and exploit the inherent structure in the data, but in this case in an unsupervised manner or order to summarize or describe data using less information.

This can be useful to visualize dimensional data or to simplify data which can then be used in a supervised learning method. Many of these methods can be adapted for use in classification and regression.

- Principal Component Analysis (PCA)
- Principal Component Regression (PCR)
- Partial Least Squares Regression (PLSR)
- Sammon Mapping
- Multidimensional Scaling (MDS)
- Projection Pursuit
- Linear Discriminant Analysis (LDA)
- Mixture Discriminant Analysis (MDA)
- Quadratic Discriminant Analysis (QDA)
- Flexible Discriminant Analysis (FDA)

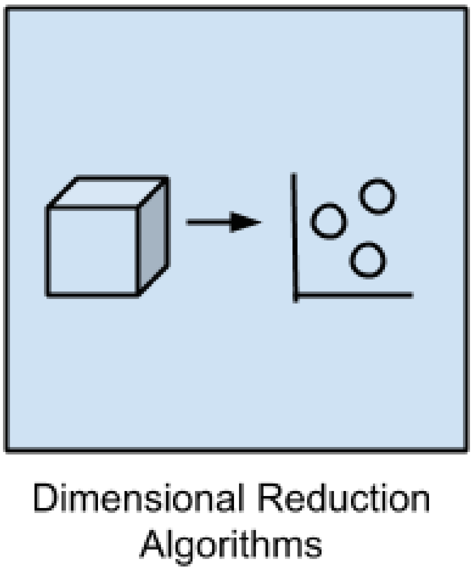


Figure 2.15 Dimensional reduction algo

2.3.2.11.Ensemble Algorithms

Ensemble methods are models composed of multiple weaker models that are independently trained and whose predictions are combined in some way to make the overall prediction.

Much effort is put into what types of weak learners to combine and the ways in which to combine them. This is a very powerful class of techniques and as such is very popular.

- Boosting
- Bootstrapped Aggregation (Bagging)
- AdaBoost
- Stacked Generalization (blending)
- Gradient Boosting Machines (GBM)
- Gradient Boosted Regression Trees (GBRT)
- Random Forest

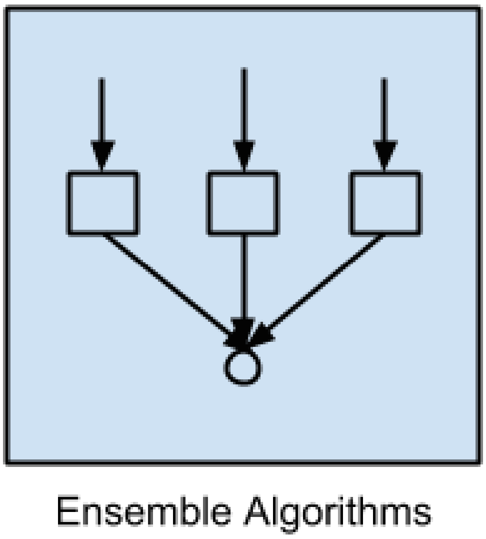


Figure 2.16 Ensemble algo

2.3.2.12. Other Algorithms

Too many algorithms cited that will not be used at all , also we did not cover all the algorithms exist .

We did not cover algorithms from specialty tasks in the process of machine learning, such as:

- Feature selection algorithms
- Algorithm accuracy evaluation
- Performance measures

We also did not cover algorithms from specialty subfields of machine learning, such as:

- Computational intelligence (evolutionary algorithms, etc.)
- Computer Vision (CV)
- Natural Language Processing (NLP)
- Recommender Systems
- Reinforcement Learning
- Graphical Models
- And more…

And the algorithms we had applied to the forex market are :

- Logistic regression
- Linear regression
- Learning Regression Forecasting and Predicting (Another style)
- Support Vector Machine (SVM)
- K-Nearest Neighbors (KNN)
- Support Vector Classifier (SVC)
- Neural Netowrk using Keras
- LSTM Recurrent Neural Networks in Python with Keras
- LSTM cell from scratch using TensorFlow
- TensorFlow

2_4. The “deep” in deep learning [5]

Deep learning is a specific subfield of machine learning: a new take on learning representations from data that puts an emphasis on learning successive *layers* of increasingly meaningful representations. The *deep* in *deep learning* isn’t a reference to any kind of deeper understanding achieved by the approach; rather, it stands for this idea of successive layers of representations. How many layers contribute to a model of the data is called the *depth* of the model. Other appropriate names for the field could have been *layered representations learning* and *hierarchical representations learning*. Modern deep learning often involves tens or even hundreds of successive layers of representations— and they’re all learned automatically from exposure to training data. Meanwhile, other approaches to machine learning tend to focus on learning only one or two layers of representations of the data; hence, they’re sometimes called *shallow learning*. In deep learning, these layered representations are (almost always) learned via models called *neural networks*, structured in literal layers connected to each other in specific topologies .

The term *neural network* is a reference to neurobiology, but although some of the central concepts in deep learning were developed in part by drawing inspiration from our understanding of the brain, deep-learning models are *not* models of the brain[5]. There's no evidence that the brain implements anything like the learning mechanisms used in modern deep-learning models. You may come across pop-science articles proclaiming that deep learning works like the brain or was modeled after the brain, but that isn't the case. It would be confusing and counterproductive for newcomers to the field to think of deep learning as being in any way related to neurobiology; you don't need that shroud of "just like our minds" mystique and mystery, and you may as well forget anything you may have read about hypothetical links between deep learning and biology. For our purposes, deep learning is a mathematical framework for learning representations from data.

Deep learning is, technically: a multistage way to learn data representations. It's a simple idea—but, as it turns out, very simple mechanisms, sufficiently scaled, can end up looking like magic.

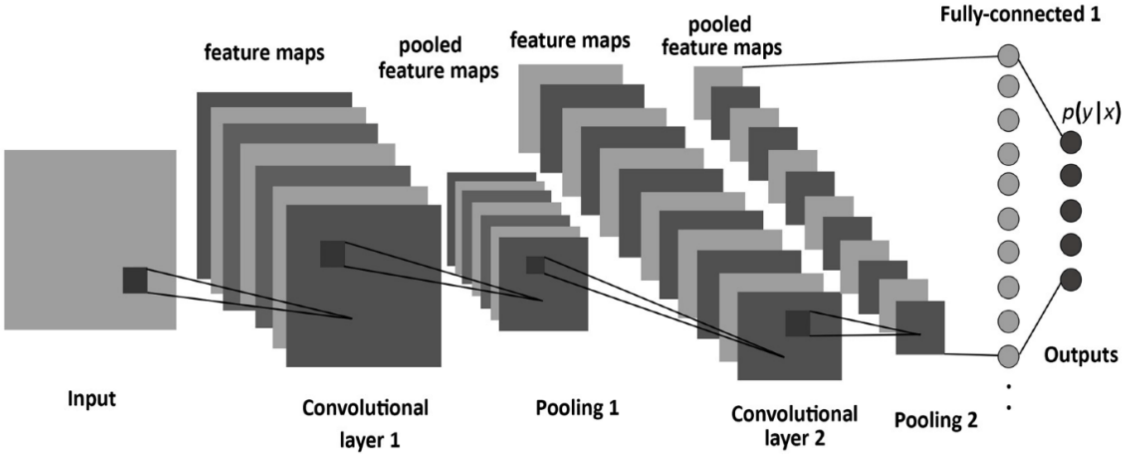


Figure 2.17.a 4 layered DNN

Deep Neural Network (DNN) is a multi-layer feed forward neural network and it uses supervised learning as shown in figure 8 . Here X_i are nodes in the input layer and Y_j represent neurons in the 1st hidden layer and it uses hyperbolic tangent function for computation. Z_k represent neurons in the 2nd layer and it again uses hyperbolic tangent function for computation. Finally the output layer has two nodes P_l which uses the softmax function for classification and linear function for regression. The hyperbolic tangent represents the activation function for the network. U_{ij} are the weights connecting the input and 1st hidden layer and b_j are the biases for 1st hidden layer. V_{jk} are the weights connecting the 1st hidden layer and the 2nd hidden layer and c_k are the biases for 2nd hidden layer.

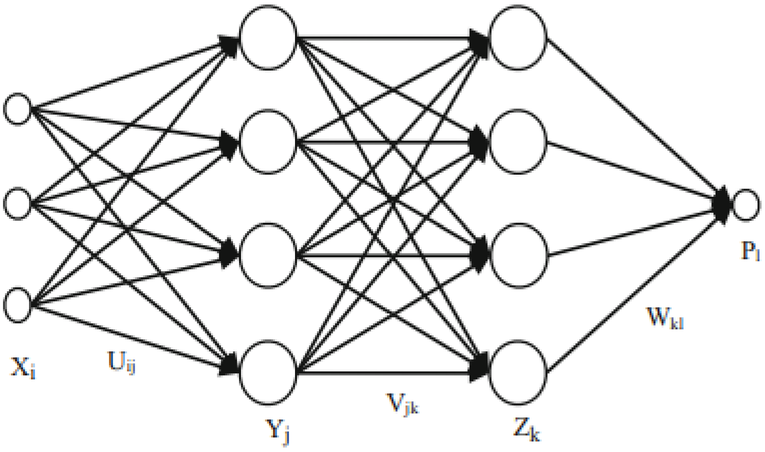


Figure 2.b Forward Propagation of a 4 layered DNN

Finally W_{kl} are the weights connecting the 2nd hidden layer and the output layer and dl are the biases for output layer.

$$Y_j = f(X_i, U_{ij}, b_j) = tanh\left\{\left(\sum_{i=1}^{3} X_i * U_{ij}\right) + b_j\right\}$$
$$= \frac{e^{\left\{\left(\sum_{i=1}^{3} X_i * U_{ij}\right)+b_j\right\}} - e^{-\left\{\left(\sum_{i=1}^{3} X_i * U_{ij}\right)+b_j\right\}}}{e^{\left\{\left(\sum_{i=1}^{3} X_i * U_{ij}\right)+b_j\right\}} + e^{-\left\{\left(\sum_{i=1}^{3} X_i * U_{ij}\right)+b_j\right\}}}$$

Equation 2.1

$$Z_k = f_1(Y_j, V_{jk}, c_k) = tanh\left\{\left(\sum_{j=1}^{4} Y_j * V_{jk}\right) + c_k\right\}$$
$$= \frac{e^{\left\{\left(\sum_{j=1}^{4} Y_j * V_{jk}\right)+c_k\right\}} - e^{-\left\{\left(\sum_{j=1}^{4} Y_j * V_{jk}\right)+c_k\right\}}}{e^{\left\{\left(\sum_{j=1}^{4} Y_j * V_{jk}\right)+c_k\right\}} + e^{-\left\{\left(\sum_{j=1}^{4} Y_j * V_{jk}\right)+c_k\right\}}}$$

Equation 2.2

$$P_l = f_2(Z_k, W_{kl}, d_l) = softmax\left\{\left(\sum_{k=1}^{4} Z_k * W_{kl}\right) + d_l\right\} = \frac{e^{\left\{\left(\sum_{k=1}^{4} Z_k * W_{kl}\right) + d_l\right\}}}{\sum_{k=1}^{4} e^{\left\{\left(\sum_{k=1}^{4} Z_k * W_{kl}\right) + d_l\right\}}}$$

Equation 2.3

Learning occurs when these weights are adapted to minimize the error on labelled training data. The loss error function which is the objective function is minimized for the model depending on whether the model terminates in a linear regression or classification. W is the collection {w_i }1:N-1, where wi denotes the weight matrix connecting layers i and i + 1 for a network of N layers. B is the collection {b_i }1:N-1, where bi denotes the column vector of biases for layer i + 1. The model given in Fig8 is a regression problem. For regression the loss function is given below:

$$Mean\ \ Squared\ \ Error = L(W,\ \ B|j) = \frac{1}{2}\sum_{j=1}^{n}\left(y_j - \hat{y}_j\right)^2$$

Equation 2.4

Here, yj is the actual output and $\hat{y}_j$ is the predicted output where j denotes number of training examples. The loss function for classification is given below:

$$Cross\ \ Entropy = L(W, B|j) = -\sum_{j=1}^{n} ln\left(\hat{y}_j\right) * y_j + ln\left(1 - \hat{y}_j\right) * \left(1 - y_j\right)$$

Equation 2.5

In order to update weights and biases of the network a supervised training algorithm Stochastic Gradient Descent (SGD) is used. Following process is iterated till the convergence criteria are reached. First, W and B are initialized and then updated according to the following equations.

$$w_{jm} = w_{jm} - \alpha * \frac{\partial L(W, B|j)}{\partial w_{jm}}$$

Equation 2.6

$$b_{jm} = b_{jm} - \alpha * \frac{\partial L(W, B|j)}{\partial b_{jm}}$$ Equation 2.7

Here, α is the learning rate and w_{jm} is the weight for m^{th} neuron connecting layer j and j + 1. Similarly, b_{jm} is the bias for mth neuron connecting layer j and j + 1 whereas

$\partial L(W, B| \frac{j)}{\partial w_{jm}}$

is computed using backward propagation. The chain rule is used to compute this function and for the last output layer the computation is shown below:

$$\frac{\partial L(W, B|j)}{\partial w_{jm}} = \frac{\partial L(W, B|j)}{\partial f_2(Z_k, w_{kl}, d_l)} * \frac{\partial f(Z_k, w_{kl}, d_l)^2}{\partial\left(\sum_{k=1}^{4} Z_k * w_{kl} + d_l\right)} * \frac{\partial\left(\sum_{k=1}^{4} Z_k * w_{kl} + d_l\right)}{\partial w_{jm}}$$

Equation 2.8

2_5 _ Summary

In this chapter we gave an introduction to Artificial intelligence and an insight in Machine learning and Deep learning , then an overview about different types of algorithms exist .

The next chapter describes implementation methods , and shows and discuss experimental results .

Chapter 3. Experimental Results

Introduction

This chapter presents the design flow and the methodology we have followed to implement the ML and DL algorithms that we have chosen in the previous chapter. It also describe the database used for learning, the python codes we developed to import and use the python libraries.

3_1. Methodology

Forecasting stock prices can be a challenging task. The process of determining which indicators and input data will be used, and gathering enough training data to training the system appropriately is not obvious. The input data may be raw data on volume, price, or daily change, but also it may include derived data such as technical indicators (moving average, trend-line indicators, etc.) . It is crucial to understand what data can be useful to capture the underlying patterns and integrate into the machine learning system. The methodology used in this work consists on applying Machine Learning systems, different algorithms of them .

The financial time series data used in this research is the foreign exchange rate of EUR against USD. USD is a benchmark in current Forex market that trades against other major currencies especially EUR..

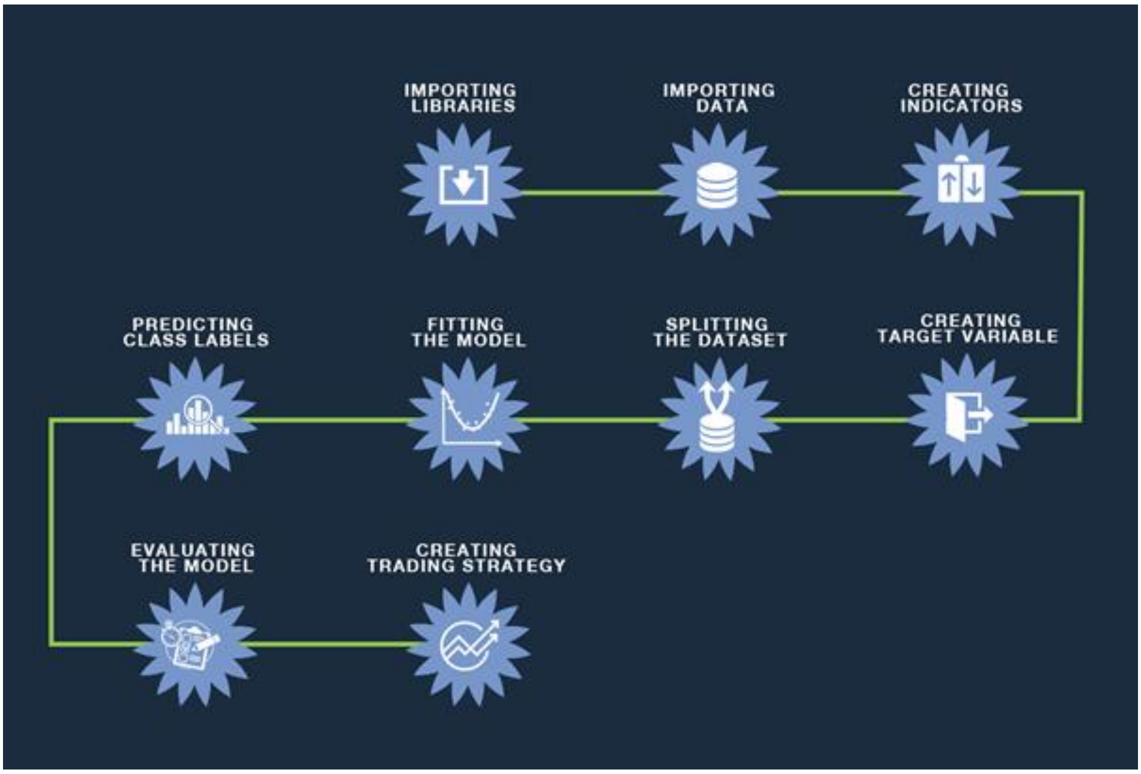


Figure 3.1 Machine Learning Process

3_2. Data Description

These financial time series data employed in the experiments consists of Day open ,high , low and close prices. The dataset encompasses the time range from 28th August 2008 until 19th March 2018. The data is collected from "Lite Forex MT4 Terminal"

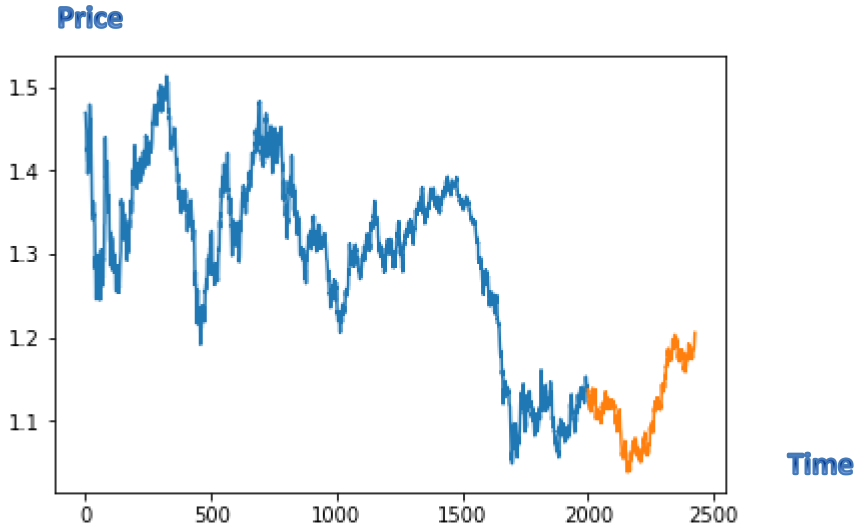


Figure 3.2 EUR/USD Daily Closing , Train Data and Test Data

3_3. Algorithms and Results

3_3.1. Neural Netowrk using Keras[1]

This shows how the NN looks like

[1] Appendix B , p86

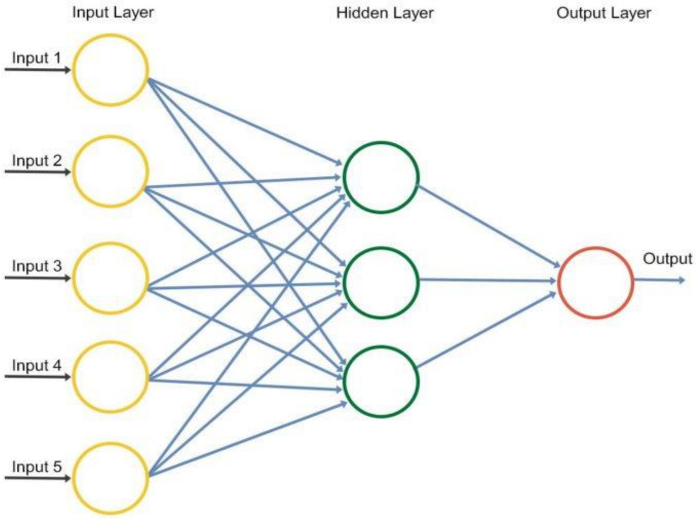
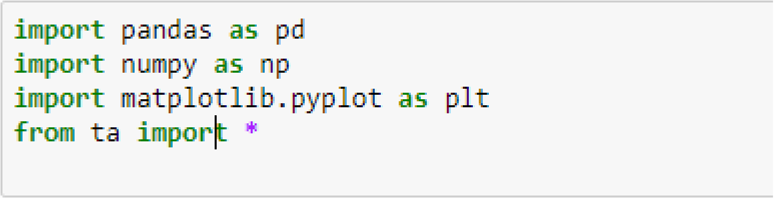


Figure 3.3 Neural Netowrk using Keras

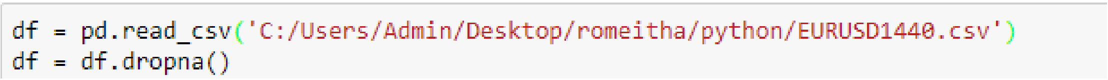
We will start by importing a few libraries, the others will be imported as and when they are used in the program at different stages. For now, we will import the libraries which will help us in importing and preparing the dataset for training and testing the model.



```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from ta import *
```

ta is a technical analysis library, which will be used to compute the RSI and MACD. These will be used as features for training our artificial neural network. We could add more features using this library.

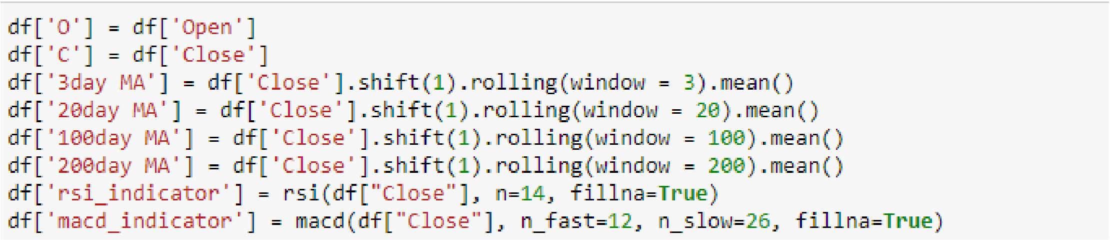
We then import our dataset, which is stored in the .csv file named ‘EURUSD1440.csv. This is done using the pandas library, and the data is stored in a data frame named df . We then drop the missing values in the dataset using the dropna() function.



```
df = pd.read_csv('C:/Users/Admin/Desktop/romeitha/python/EURUSD1440.csv')
df = df.dropna()
```

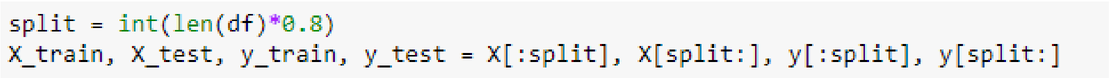
We then prepare the various input features which will be used by the artificial neural network to train itself for making the predictions. We define the following input features:

- OpenHigh minus Low price
- Close price
- Three day moving average
- 20 day moving average
- 100 day moving average
- 200 day moving average
- Relative Strength Index RSI
- MACD



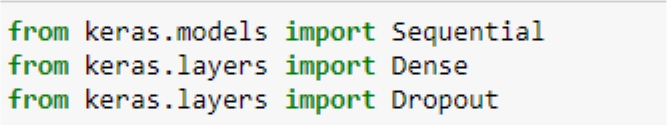
```
df['O'] = df['Open']
df['C'] = df['Close']
df['3day MA'] = df['Close'].shift(1).rolling(window = 3).mean()
df['20day MA'] = df['Close'].shift(1).rolling(window = 20).mean()
df['100day MA'] = df['Close'].shift(1).rolling(window = 100).mean()
df['200day MA'] = df['Close'].shift(1).rolling(window = 200).mean()
df['rsi_indicator'] = rsi(df["Close"], n=14, fillna=True)
df['macd_indicator'] = macd(df["Close"], n_fast=12, n_slow=26, fillna=True)
```

we will split our input and output variables to create the test and train datasets. This is done by creating a variable called split, which is defined to be the integer value of 0.8 times the length of the dataset.



```
split = int(len(df)*0.8)
X_train, X_test, y_train, y_test = X[:split], X[split:], y[:split], y[split:]
```

Now we will import the functions which will be used to build the artificial neural network. We import the Sequential method from the keras .models library. This will be used to sequentially build the layers of the neural networks. The next method that we import will be the Dense function from the keras.layers library. This method will be used to build the layers of our artificial neural network.



```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
```

We will now plot the market returns and our strategy returns to visualize how our strategy is performing against the market.

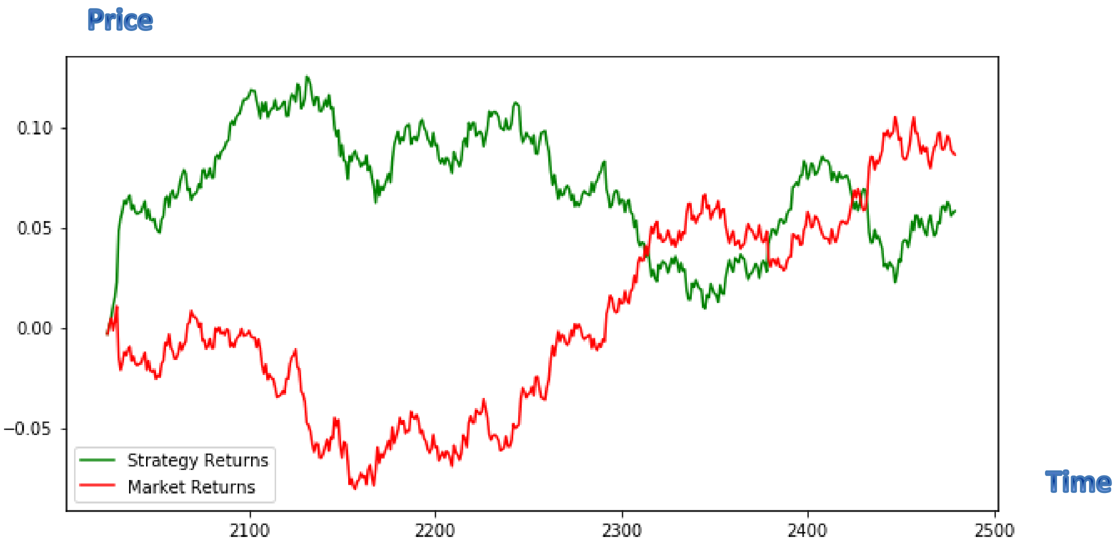


Figure 3.4 Neural Netowrk using Keras

We can get different out comes depends on parameters we use ,

But as a start , it looks promising .

3_3.2. Deep Learning , prediction with LSTM Recurrent Neural Networks in Python with Keras

A powerful type of neural network designed to handle sequence dependence is called recurrent neural networks. The Long Short-Term Memory network or LSTM network is a type of recurrent neural network used in deep learning because very large architectures can be successfully trained.

Long short-term memory is a modified RNN architecture that tackles the problem of vanishing and exploding gradients and addresses the problem of training over long sequences and retaining memory. All RNNs have feedback loops in the recurrent layer. The feedback loops help keep information in “memory” over time. But, it can be difficult to train standard RNNs to solve problems that require learning long-term temporal dependencies. Since the gradient of the loss function decays exponentially with time (a phenomenon known as the *vanishing gradient problem*), it is difficult to train typical RNNs. That is why an RNN is modified in a way that it includes a memory cell that can maintain information in memory for long periods of time. The modified RNN is better known as LSTM. In LSTM, a set of gates is used to control when information enters memory, which solves the vanishing or exploding gradient problem. The recurrent connections add state or memory to the network and allow it to learn and harness the ordered nature of observations within input sequences. The internal memory means outputs of the network are conditional on the recent context in the input sequence, not what has just been presented as input to the network.

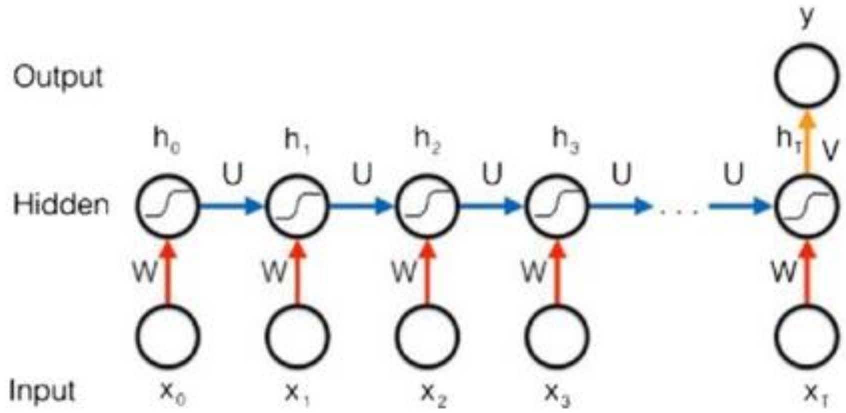
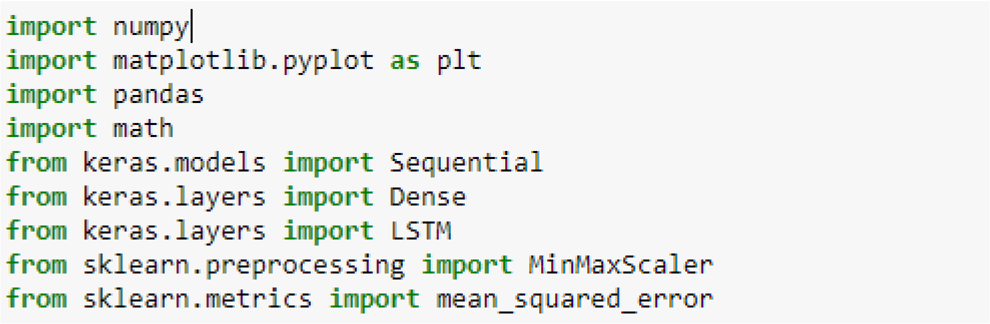


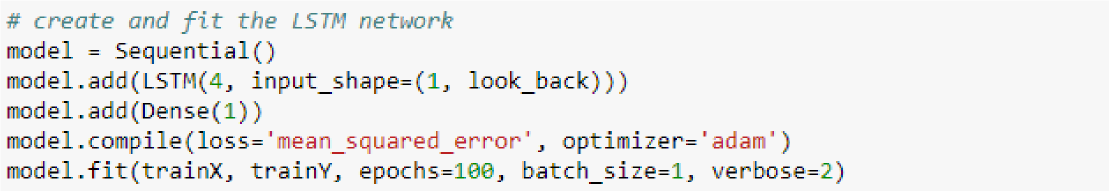
Figure 3.5 Many to one LSTM

Before we get started, let's first import all of the functions and classes we intend to use. This assumes a working SciPy environment with the Keras deep learning library installed.



```
import numpy
import matplotlib.pyplot as plt
import pandas
import math
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
```

The network has a visible layer with 1 input, a hidden layer with 4 LSTM blocks or neurons, and an output layer that makes a single value prediction. The default sigmoid activation function is used for the LSTM blocks. The network is trained for 100 epochs and a batch size of 1 is used.



```
# create and fit the LSTM network
model = Sequential()
model.add(LSTM(4, input_shape=(1, look_back)))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
model.fit(trainX, trainY, epochs=100, batch_size=1, verbose=2)
```

Finally, we can generate predictions using the model for both the train and test dataset to get a visual indication of the skill of the model.

Because of how the dataset was prepared, we must shift the predictions so that they align on the x-axis with the original dataset. Once prepared, the data is plotted, showing the original dataset in blue, the predictions for the training dataset in green, and the predictions on the unseen test dataset in orange .

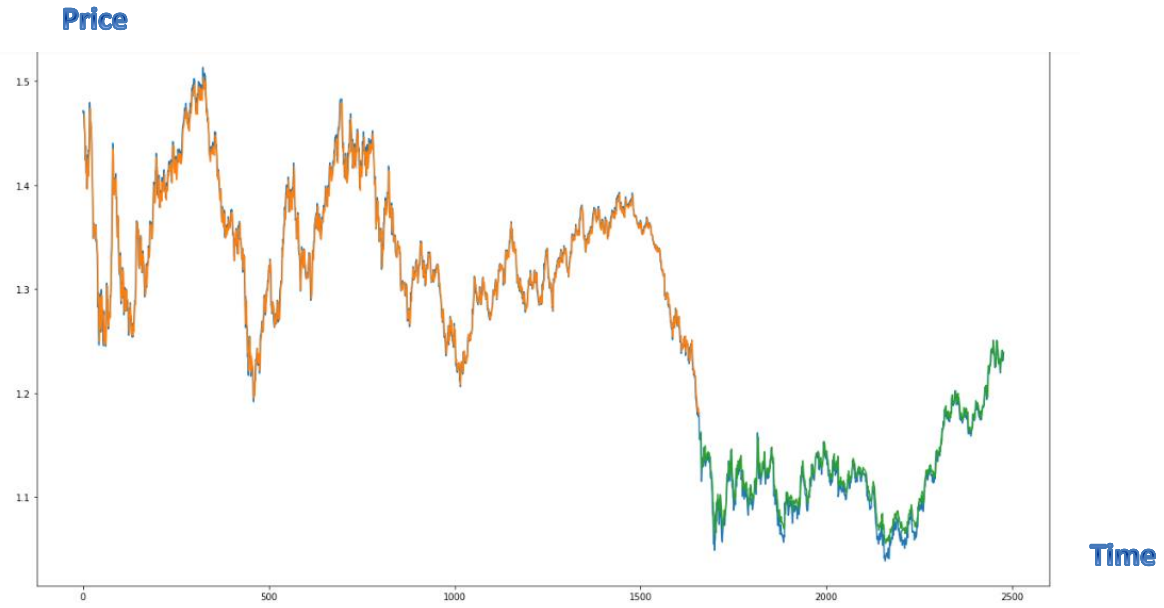
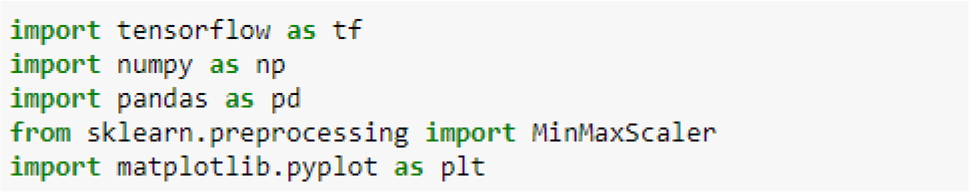


Figure 3.6 LSTM Recurrent Neural Networks in Python with Keras

We discovered how to develop LSTM recurrent neural networks for time series prediction in Python with the Keras deep learning network **.**

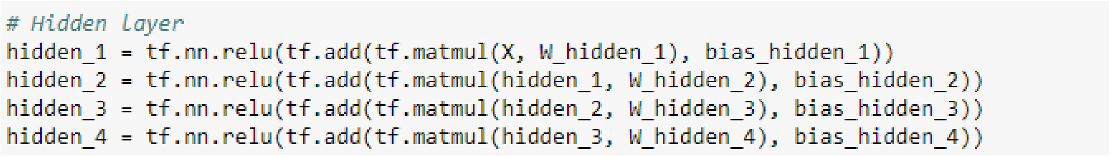
3_3.3. Deep Learning , prediction using TensorFlow[2]

First we import all of the functions and classes we will use .



```
import tensorflow as tf
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
import matplotlib.pyplot as plt
```

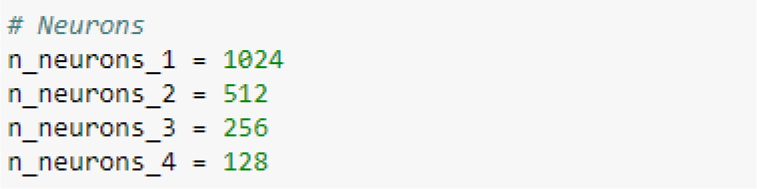
We will use four hidden layers



```
# Hidden Layer
hidden_1 = tf.nn.relu(tf.add(tf.matmul(X, W_hidden_1), bias_hidden_1))
hidden_2 = tf.nn.relu(tf.add(tf.matmul(hidden_1, W_hidden_2), bias_hidden_2))
hidden_3 = tf.nn.relu(tf.add(tf.matmul(hidden_2, W_hidden_3), bias_hidden_3))
hidden_4 = tf.nn.relu(tf.add(tf.matmul(hidden_3, W_hidden_4), bias_hidden_4))
```

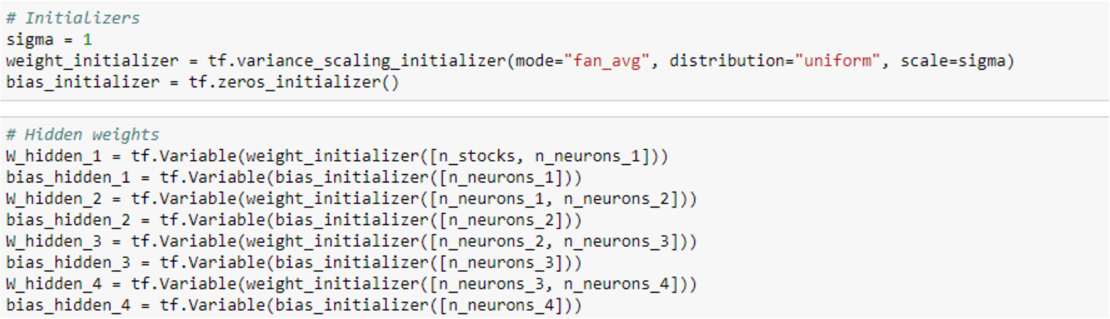
And neurons like this

[2] Appendix B p89



```
# Neurons
n_neurons_1 = 1024
n_neurons_2 = 512
n_neurons_3 = 256
n_neurons_4 = 128
```

And weights like this



```
# Initializers
sigma = 1
weight_initializer = tf.variance_scaling_initializer(mode="fan_avg", distribution="uniform", scale=sigma)
bias_initializer = tf.zeros_initializer()
```

```
# Hidden weights
W_hidden_1 = tf.Variable(weight_initializer([n_stocks, n_neurons_1]))
bias_hidden_1 = tf.Variable(bias_initializer([n_neurons_1]))
W_hidden_2 = tf.Variable(weight_initializer([n_neurons_1, n_neurons_2]))
bias_hidden_2 = tf.Variable(bias_initializer([n_neurons_2]))
W_hidden_3 = tf.Variable(weight_initializer([n_neurons_2, n_neurons_3]))
bias_hidden_3 = tf.Variable(bias_initializer([n_neurons_3]))
W_hidden_4 = tf.Variable(weight_initializer([n_neurons_3, n_neurons_4]))
bias_hidden_4 = tf.Variable(bias_initializer([n_neurons_4]))
```

We will get this graph

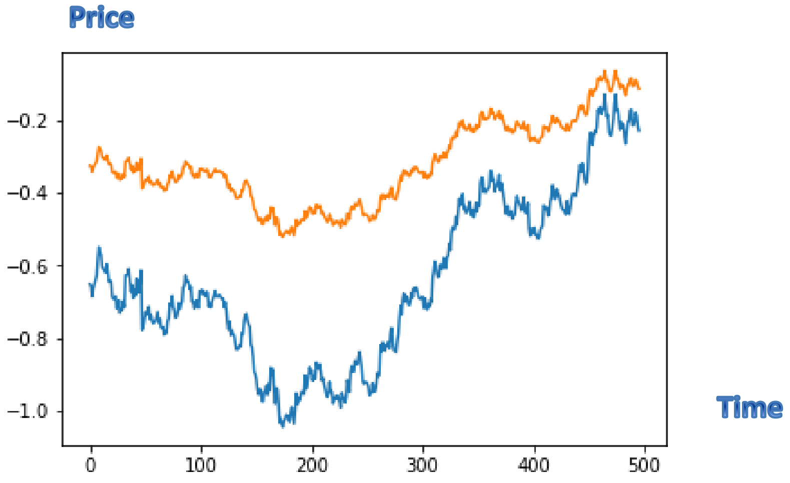
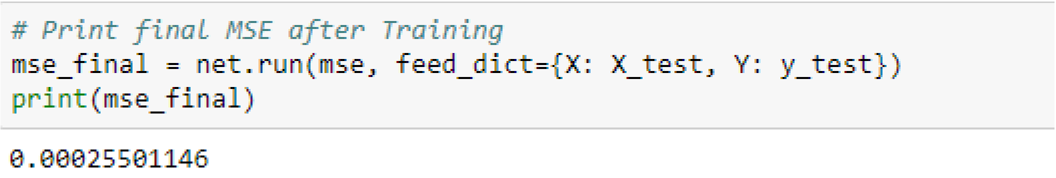


Figure 3.7 TensorFlow

the mean squared error (MSE) of an estimator is one of many ways to quantify the difference between values implied by an estimator and the true values of the quantity being estimated , it is here



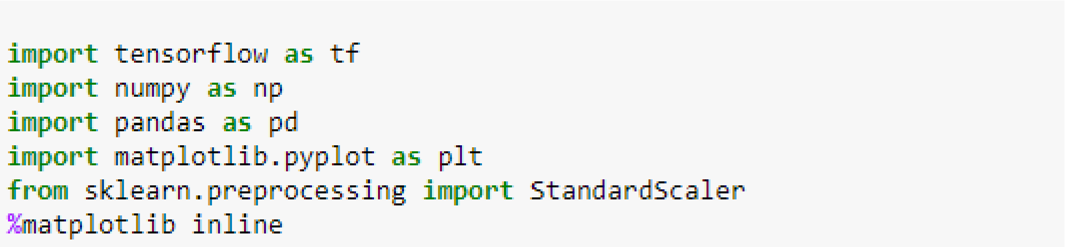
```
# Print final MSE after Training
mse_final = net.run(mse, feed_dict={X: X_test, Y: y_test})
print(mse_final)
```

```
0.00025501146
```

The best value is 0.0, higher values are worse. So it is a good result that we got .

3_3.4. Deep Learning , implementing LSTM cell from scratch using TensorFlow

Before loading the data, we imported relevant python modules .



```
import tensorflow as tf
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
%matplotlib inline
```

Here we plotted the Close prices of EUR/USD

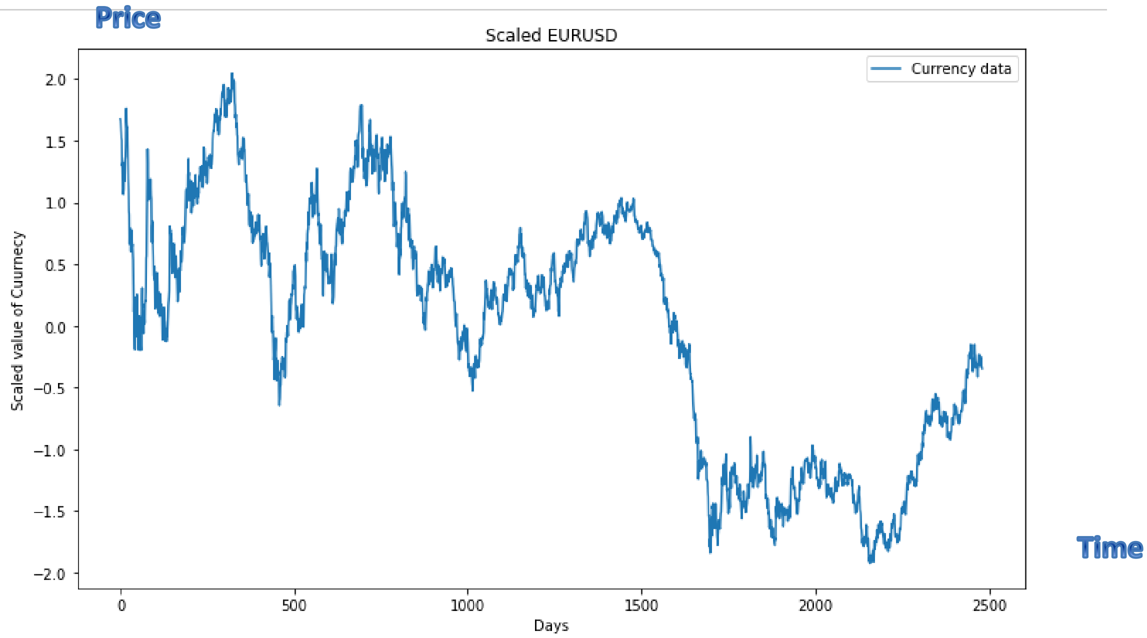


Figure 3.8 the Close prices of EUR/USD

It will take few minutes to manipulate the data , and it gives us

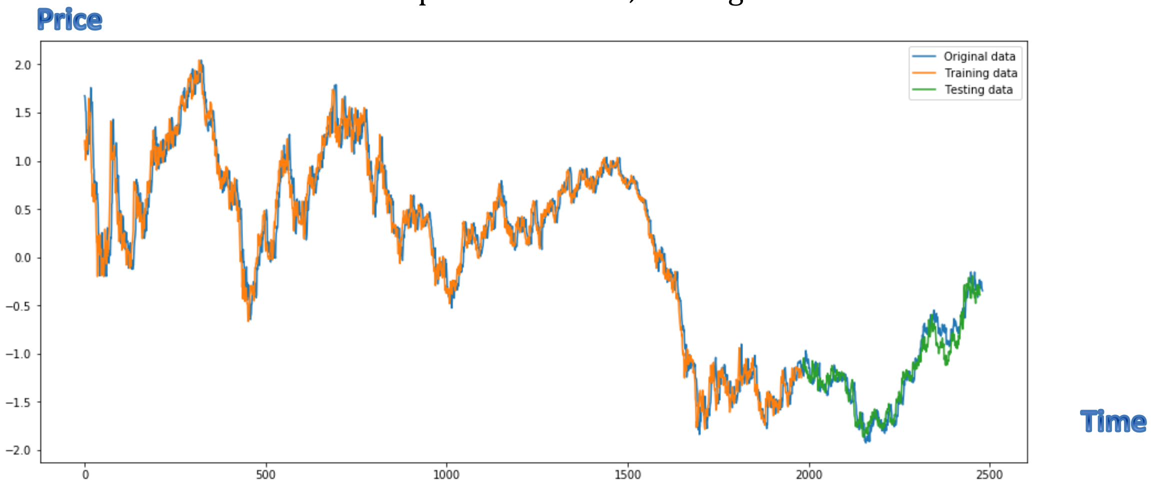


Figure 3.9 LSTM cell from scratch using TensorFlow

It looks profitable .

3_3.5. machine learning Logistic regression

Logistic regression falls under the category of supervised learning; it measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic/sigmoid function. In spite of the name ‘*logistic regression*’, this is not used for regression problem where the task is to predict the real-valued output. It is a classification problem which is used to predict a binary outcome (1/0, -1/1, True/False) given a set of independent variables.

Logistic regression is a bit similar to the linear regression or we can say it as a generalized linear model. In linear regression, we predict a real-valued output ‘*y*’ based on a weighted sum of input variables.

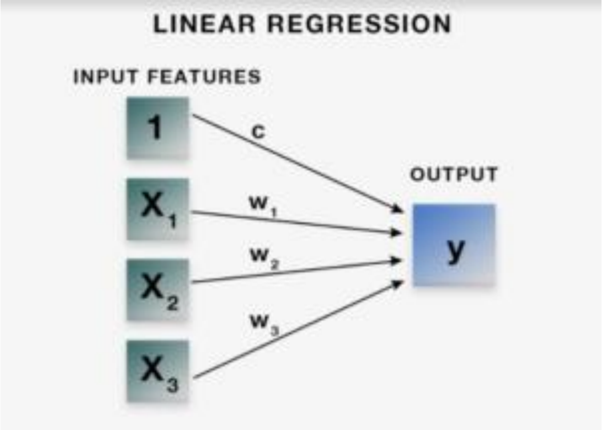


Figure 3.10 Logistic regression

$$y = c + x_1{*}w_1 + x_2{*}w_2 + x_3{*}w_3 + \ldots\ldots.. + x_n{*}w_n$$

Equation 3.1

The aim of linear regression is to estimate values for the model coefficients c, w_1, w_2, w_3 ….w_n and fit the training data with minimal squared error and predict the output y.

Logistic regression does the same thing, but with one addition. The logistic regression model computes a weighted sum of the input variables similar to the linear regression, but it runs the result through a special non-linear function, the logistic function or sigmoid function to produce the output y. Here, the output is binary or in the form of 0/1 or -1/1.

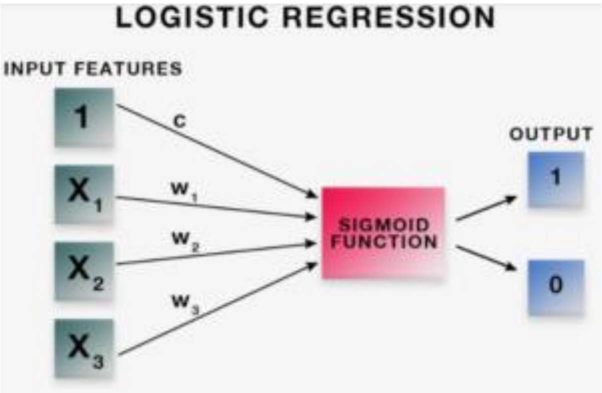


Figure 3.11 Logistic regression

$$y = \text{logistic}\ (c + x_1{*}w_1 + x_2{*}w_2 + x_3{*}w_3 + \ldots\ldots.. + x_n{*}w_n)$$

$$y = 1\ /\ 1 + e\ [-\ (c + x_1{*}w_1 + x_2{*}w_2 + x_3{*}w_3 + \ldots\ldots.. + x_n{*}w_n)]$$

Equation 3.2

The sigmoid/logistic function is given by the following equation.

$y = 1 / 1+ e^{-x}$ Equation 3.3

As you can see in the graph, it is an S-shaped curve that gets closer to 1 as the value of input variable increases above 0 and gets closer to 0 as the input variable decreases below 0. The output of the sigmoid function is 0.5 when the input variable is 0.

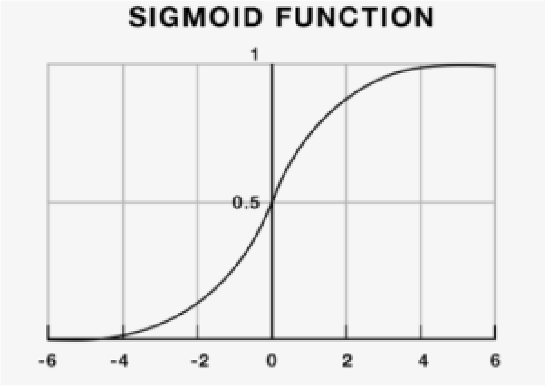


Figure 3.12 sigmoid funcion

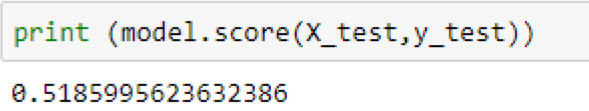
Thus, if the output is more than 0.5, we can classify the outcome as 1 (or positive) and if it is less than 0.5, we can classify it as 0 (or negative).

Now, let us consider the task of predicting the stock price movement. If tomorrow's closing price is higher than today's closing price, then we will buy the stock (1), else we will sell it (-1). If the output is 0.7, then we can say that there is a 70% chance that tomorrow's closing price is higher than today's closing price and classify it as 1.

Now, we have a basic intuition behind the logistic regression and the sigmoid function. We will learn how to implement logistic regression in Python and predict the stock price movement using the above condition.

We will start by importing the necessary libraries, then we import the data , and define the inputs same before .

We will calculate the model accuracy on the test dataset using '*score*' function.



```
print (model.score(X_test,y_test))
```

```
0.5185995623632386
```

We will get this Graph :

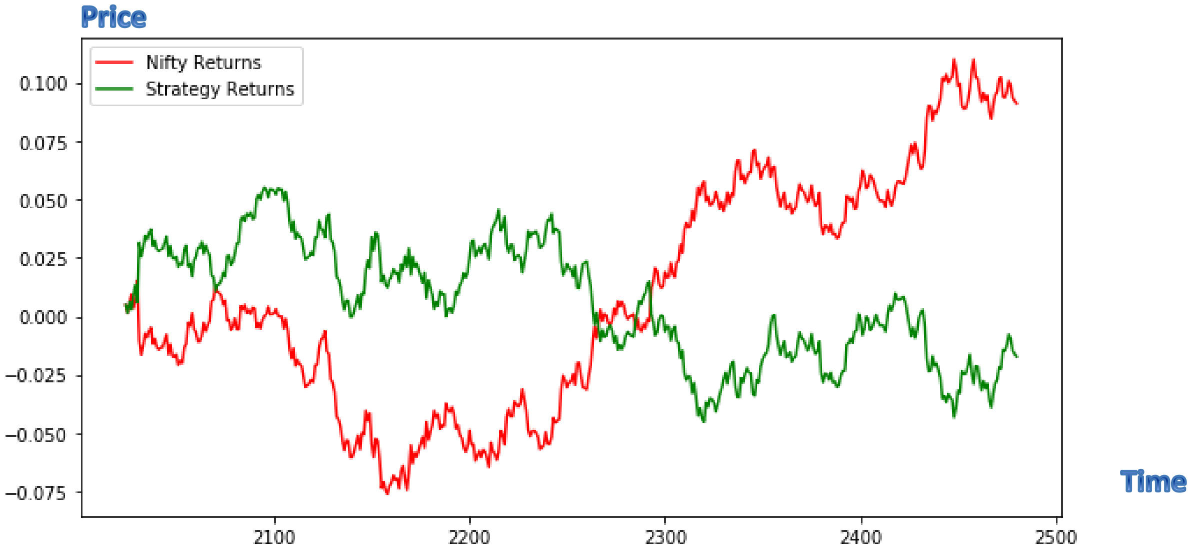


Figure 3.13 Logistic regression

It can be observed that the Logistic Regression model predict the classes with an accuracy of approximately 52% and generates good returns.

3.3.6 Machine Learning Linear regression

We will create a machine learning linear regression model that takes information from the past EUR/USD prices and returns a prediction of the EUR/USD price the next day.

The linear model has been present for a long time now and remains one of the most important tools in the statistics field. A linear regression model can be represented by the following mathematical expression:

$$\boldsymbol{X} = (X_1, X_2, \dots, X_p)$$

$$\boldsymbol{\beta} = (\beta_1, \beta_2, \dots, \beta_p)$$

$$\hat{Y} = \hat{\beta}_0 + \sum_{j=1}^{p} X_j \hat{\beta}_j$$

Equation 3.4

where X_i represents the model input variables and Y is the model output variable. The β_i , i=1,2,...,p are the model parameters which need to be estimated. The term β_0 is the intercept, also known as the bias in machine learning. Often it is convenient to include the constant variable 1 in X, include β0 in the vector of coefficients β, and then write the linear model in vector form as an inner product:

$$\hat{Y} = X^T \hat{\beta}$$

Equation 3.5

For the estimation of the coefficients of the model β the most common approach is to estimate using the least squares method. In this approach β is estimate in order to minimize the residual sum of squares:

$$RSS(\beta) = \sum_{j=1}^{N} (y_i - x_i^T \beta)^2$$

Equation 3.6

writing the formula in matrix notation we have:

$$RSS(\beta) = (y - X\beta)^T (y - X\beta)$$

Equation 3.7

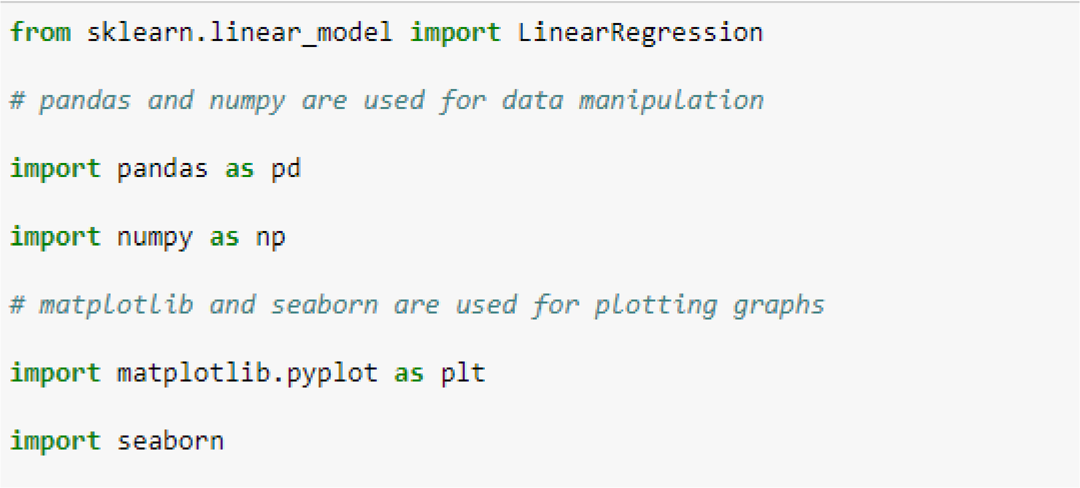
where X is an N * p matrix with each row an input vector, and y is an N-vector of the outputs in the training set. Differentiating in order of β and equal to zero we get the equations:

$$X^T(y - X\beta) = 0$$

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

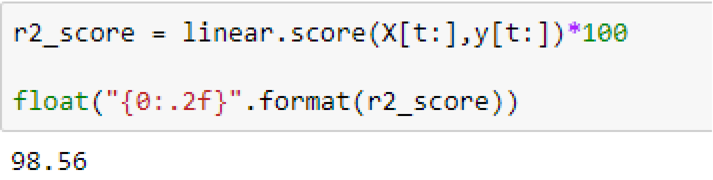
Equation 3.8

First things first: import all the necessary libraries which are required to implement this strategy.



```
from sklearn.linear_model import LinearRegression
# pandas and numpy are used for data manipulation
import pandas as pd
import numpy as np
# matplotlib and seaborn are used for plotting graphs
import matplotlib.pyplot as plt
import seaborn
```

We will calculate the model accuracy on the test dataset using ‘*score*’ function.



```
r2_score = linear.score(X[t:],y[t:])*100
float("{0:.2f}".format(r2_score))
```

```
98.56
```

We will get this Graph :

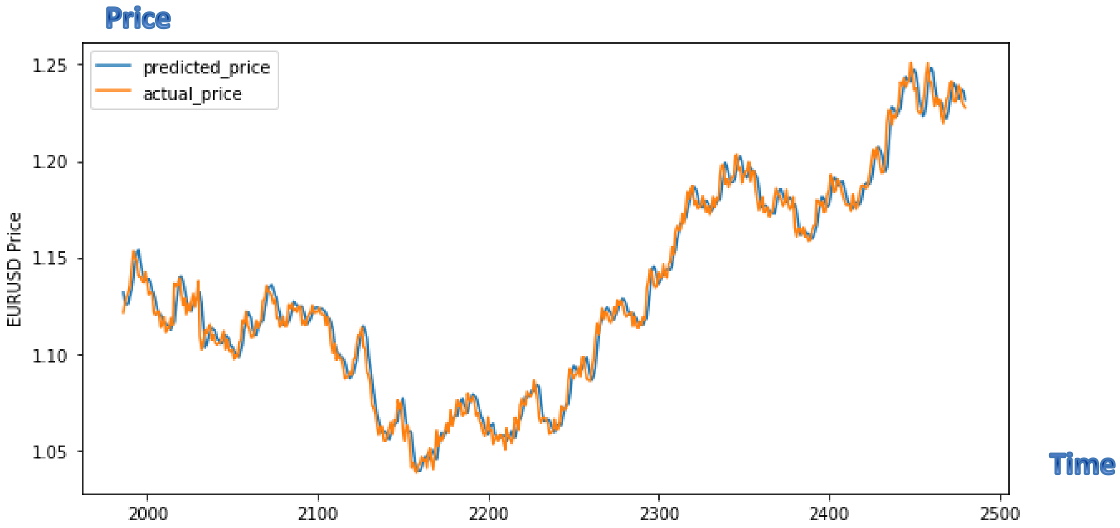


Figure 3.14 Learning Linear regression

As it can be seen, the R-squared of the model is 98 %. R-squared is always between 0 and 100%. A score close to 100% indicates that the model explains the EUR/USD prices well.

3_3.7 Machine Learning Support Vector Machine <u>(SVM)</u>

We will Create an unsupervised ML (machine learning) algorithm to predict the regimes. Plot these regimes to visualize them.

We will Train a Support Vector Classifier algorithm with the regime as one of the features, use this Support Vector Classifier algorithm to predict the current day's trend at the Opening of the market.

Then , visualize the performance of this strategy on the test data.

Suppport vector machine (SVM) have been implemented in many types of problems such classification, recognition and regression. It was firstly on classification problems, principle to develop binary classifications. The goal of support vector machine is to build a hyperplane as the decision surface such the margin of separation between labels is maximized.

For SVM regression, the inputs X are first mapped into a m-dimensional feature space using some nonlinear relation, and then a linear model is constructed in this feature space. Using mathematical notation, the linear model is given by :

$$f(X,\phi) = \sum_{j=1}^{m} (\phi_j * g_j(X)) + b$$

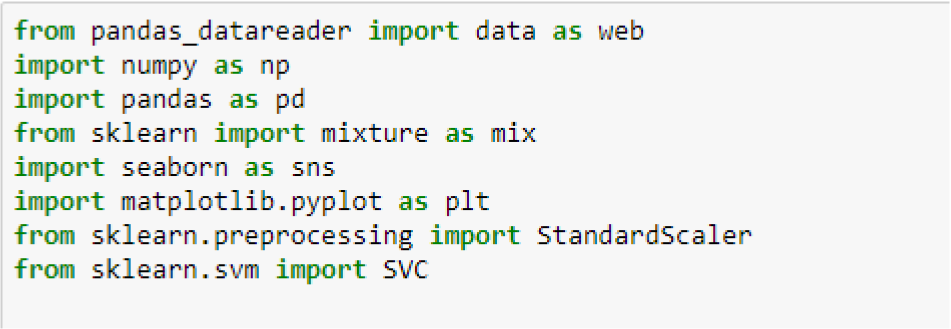
Equation 3.9

where g_j (X),j=1,...m is the function representing the nonlinear transformations and b is the 'bias' term. In order to estimate the quality of the produced outputs is used a loss function proposed by Vapnik.

$$L_\varepsilon(y, f(X,\phi)) = \begin{cases} 0, & \text{if } |y - f(X,\phi)| <= \varepsilon \\ |y - f(X,\phi)| - \varepsilon, & \text{othetwise} \end{cases}$$

Equation 3.10

First thing , we import all the necessary libraries which are required to implement this strategy.



```
from pandas_datareader import data as web
import numpy as np
import pandas as pd
from sklearn import mixture as mix
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
```

We calculated the cumulative strategy returns and the cumulative market returns and saved them in df. Then, we calculated the sharpe ratio to measure the performance. To get a clear understanding of this metric we plotted the performance to measure it.

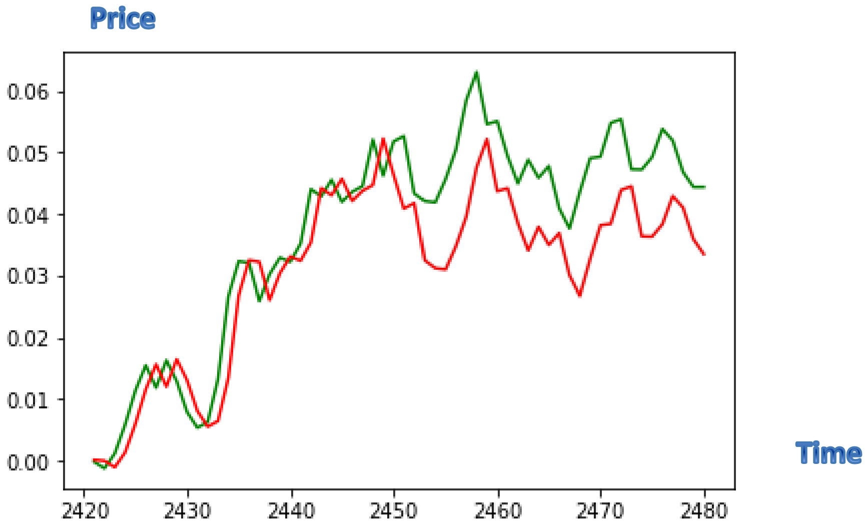


Figure 3.15 Support Vector Machine

This strategy works on some stocks but doesn't work on others, which is the case with most quant strategies. There are a few reasons why the algorithm did work consistently and we will list some of them here.

1. No autocorrelation of returns
2. No Support Vector hyper parameter optimization

3. No error propagation
4. No feature selection

We have not checked for autocorrelation of the returns, which would have increased the predictability of the algorithm.

3_3.8. Machine Learning K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is one of the simplest algorithms used in Machine Learning. KNN algorithms use a data and classify new data points based on a similarity measures (e.g. distance function). Classification is done by a majority vote to its neighbors. The data is assigned to the class which has the most nearest neighbors. As you increase the number of nearest neighbors, the value of k, accuracy might increase.

The ***k*-nearest neighbors algorithm** (***k*-NN**) is a non-parametric method used for classification and regression. In both cases, the input consists of the *k* closest training examples in the feature space. The output depends on whether *k*-NN is used for classification or regression:

- In *k-NN classification*, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small). If $k = 1$, then the object is simply assigned to the class of that single nearest neighbor.

- In *k-NN regression*, the output is the property value for the object. This value is the average of the values of its *k* nearest neighbors.

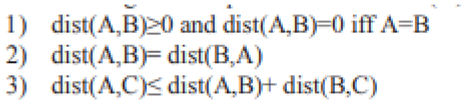
k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The *k*-NN algorithm is among the simplest of all machine learning algorithms.

Both for classification and regression, a useful technique can be to assign weight to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of $1/d$, where d is the distance to the neighbor.

The neighbors are taken from a set of objects for which the class (for *k*-NN classification) or the object property value (for *k*-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.

There are various ways of measuring the similarity between two instances with n attribute values. Every measure has the following three requirements.

Let dist (A, B) be the distance between two points A,B then

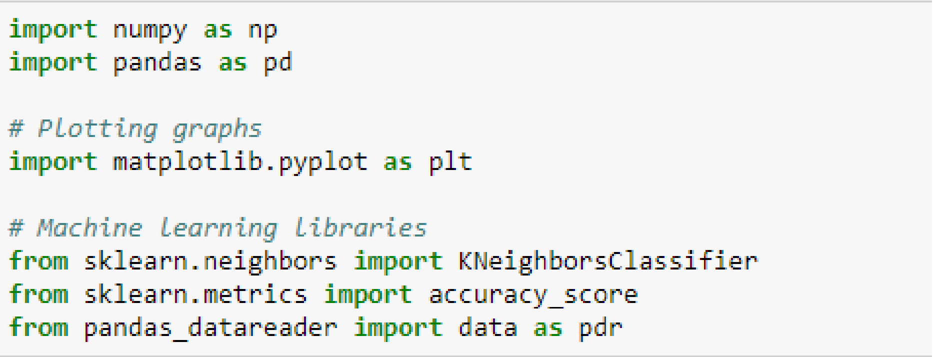


1) $\text{dist}(A,B) \geq 0$ and $\text{dist}(A,B)=0$ iff $A=B$
2) $\text{dist}(A,B)= \text{dist}(B,A)$
3) $\text{dist}(A,C) \leq \text{dist}(A,B)+ \text{dist}(B,C)$

Equation 3.11

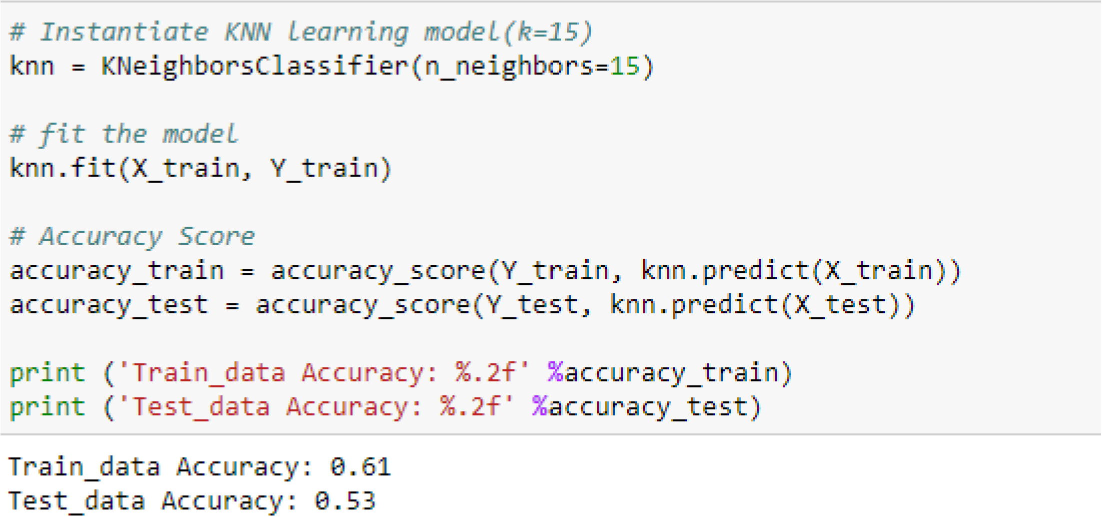
Property 3 is called as “Triangle in equality”. It states that the shortest distance between any two points is a straight line. Most common distance measures used is Euclidean distance .For continuous variables Z score standardization and min max normalization are used .

We will start by importing the necessary libraries.



```
import numpy as np
import pandas as pd

# Plotting graphs
import matplotlib.pyplot as plt

# Machine learning libraries
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
from pandas_datareader import data as pdr
```

After splitting the dataset into training and test dataset, we will instantiate k-nearest classifier. Here we are using ‘k =15’, you may vary the value of k and notice the change in result. Next, we fit the train data by using ‘fit’ function. Then, we will calculate the train and test accuracy by using ‘accuaracy_score’ function.



```
# Instantiate KNN learning model(k=15)
knn = KNeighborsClassifier(n_neighbors=15)

# fit the model
knn.fit(X_train, Y_train)

# Accuracy Score
accuracy_train = accuracy_score(Y_train, knn.predict(X_train))
accuracy_test = accuracy_score(Y_test, knn.predict(X_test))

print ('Train_data Accuracy: %.2f' %accuracy_train)
print ('Test_data Accuracy: %.2f' %accuracy_test)
```

```
Train_data Accuracy: 0.61
Test_data Accuracy: 0.53
```

Our trading strategy is simply to buy or sell. We will predict the signal to buy or sell using ‘predict’ function. Then, we will calculate the cumulative EUR/USD returns for test dataset. Next, we will calculate the cumulative strategy return based on the signal predicted by the model in the test dataset. Next, we will plot the cumulative EUR/USD returns and cumulative strategy returns and visualize the performance.

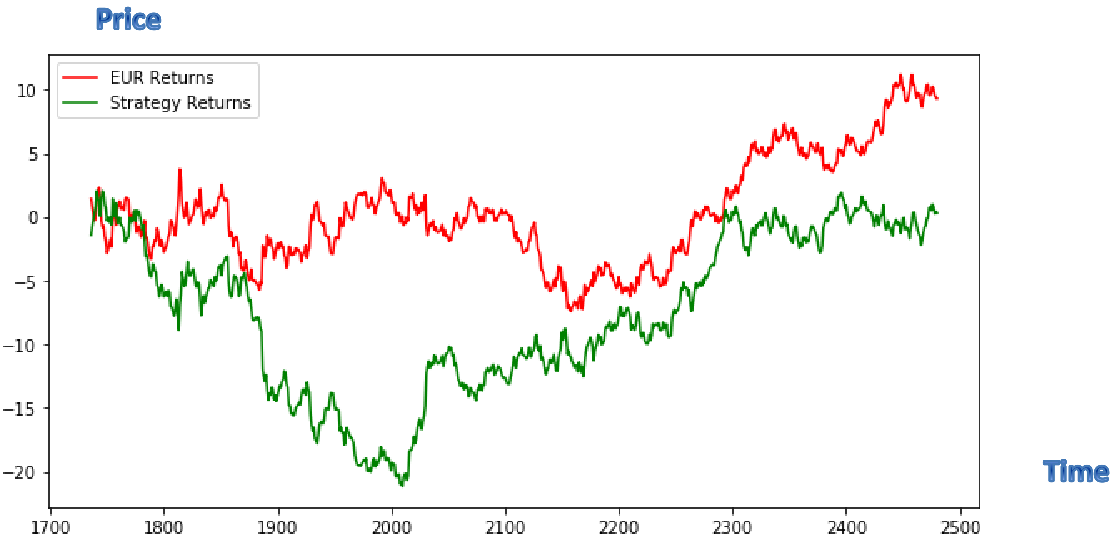
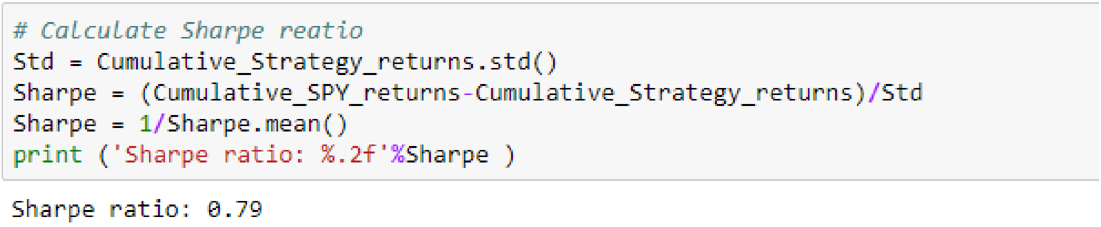


Figure 3.16 K-Nearest Neighbors

The Sharpe ratio is the return earned in excess of the market return per unit of volatility. First, we will calculate the standard deviation of the cumulative returns, and use it further to calculate the Sharpe ratio .



```
# Calculate Sharpe reatio
Std = Cumulative_Strategy_returns.std()
Sharpe = (Cumulative_SPY_returns-Cumulative_Strategy_returns)/Std
Sharpe = 1/Sharpe.mean()
print ('Sharpe ratio: %.2f'%Sharpe )
```

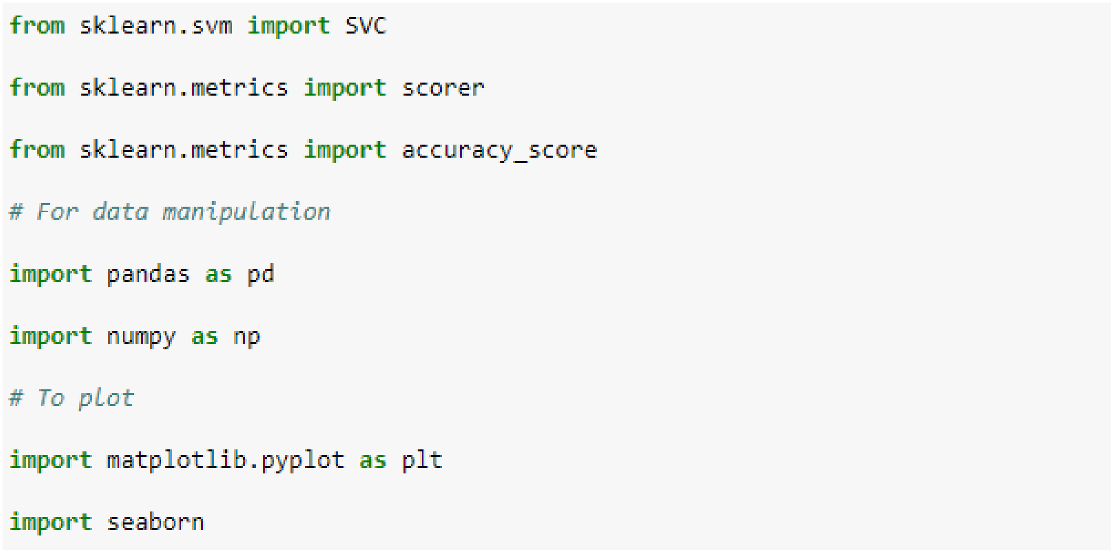
```
Sharpe ratio: 0.79
```

It is a good ratio , we can improve it more .

3_3.9. Machine Learning Support Vector Classifier (SVC)

SVCs are supervised learning classification models. A set of training data is provided to the machine learning classification algorithm, each belonging to one of the categories. For instance, the categories can be to either buy or sell a stock. The classification algorithm builds a model based on the training data and then, classifies the test data into one of the categories.

we will import the necessary libraries that will be needed to create the strategy.

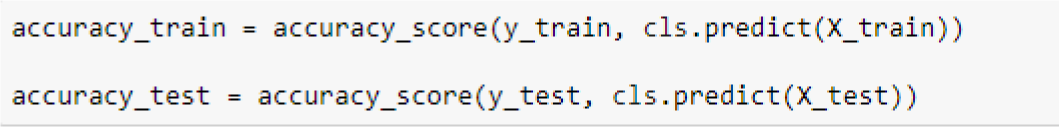


```
from sklearn.svm import SVC

from sklearn.metrics import scorer

from sklearn.metrics import accuracy_score

# For data manipulation

import pandas as pd

import numpy as np

# To plot

import matplotlib.pyplot as plt

import seaborn
```

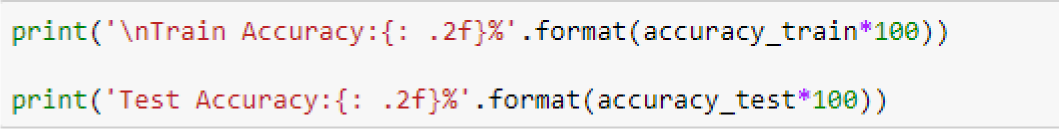
After the regular steps , we will compute the accuracy of the classification model on the train and test dataset, by comparing the actual values of the trading signal with the predicted values of the trading signal. The function accuracy_score() will be used to calculate the accuracy.

Syntax: accuracy_score(target_actual_value,target_predicted_value)

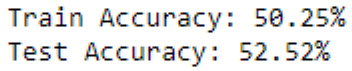
1. target_actual_value: correct signal values
2. target_predicted_value: predicted signal values



```
accuracy_train = accuracy_score(y_train, cls.predict(X_train))

accuracy_test = accuracy_score(y_test, cls.predict(X_test))
```



```
print('\nTrain Accuracy:{: .2f}%'.format(accuracy_train*100))

print('Test Accuracy:{: .2f}%'.format(accuracy_test*100))
```



```
Train Accuracy: 50.25%
Test Accuracy: 52.52%
```

An accuracy of 50%+ in test data suggests that the classification model is effective.

We will predict the signal (buy or sell) for the test data set, using the cls.predict() function. Then, we will compute the strategy returns based on the signal predicted by the model in the test dataset. We save it in the column 'Strategy_Return' and then, plot the cumulative strategy returns.

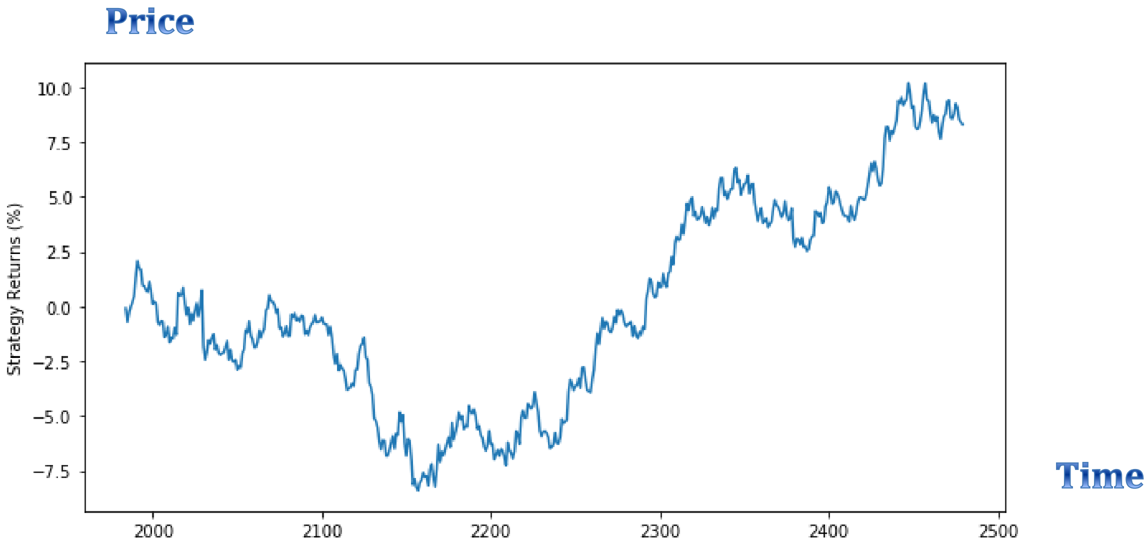
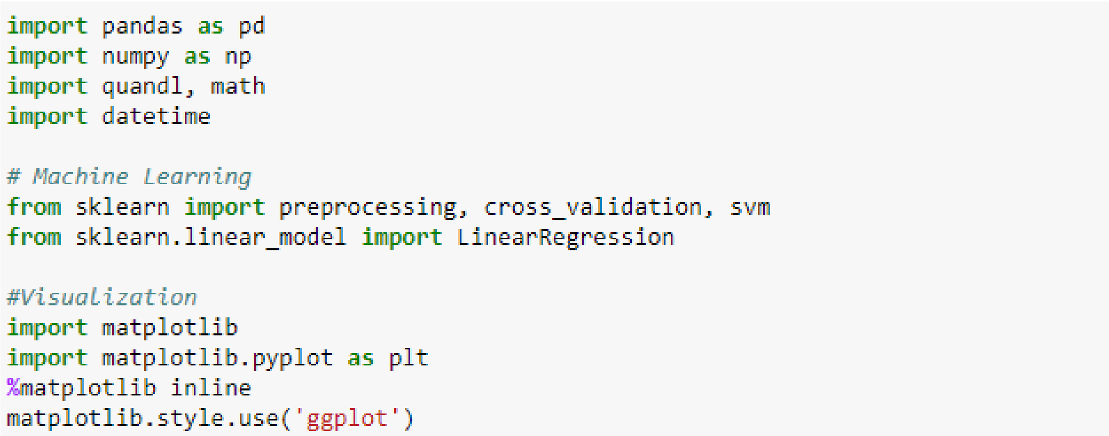


Figure 3.17 Support Vector Classifier

As seen from the graph, the machine learning classification strategy generates a return of around 10% in the test data set.

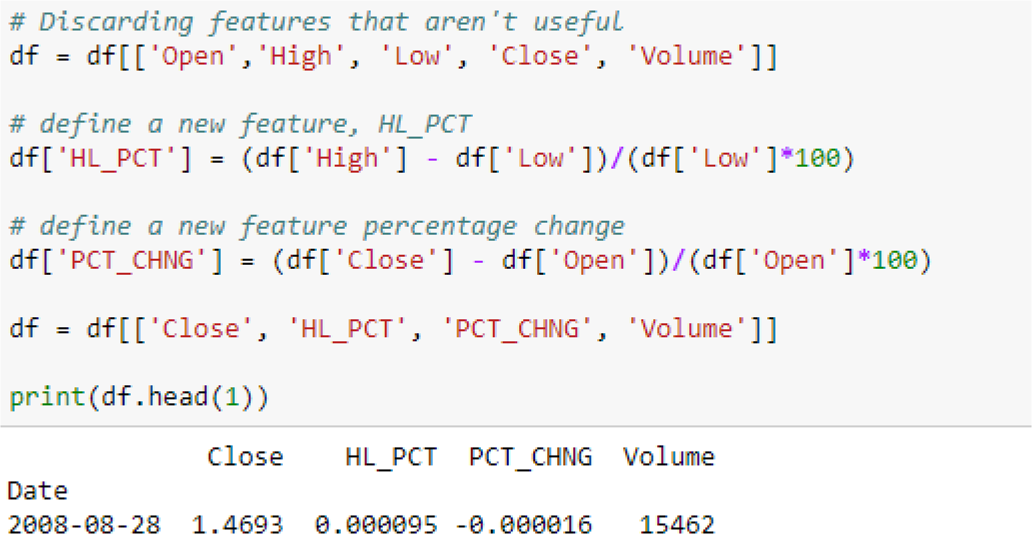
3_3.10. Machine Learning Regression Forecasting and Predicting (Another style)

Before loading the data, we imported relevant python modules .



```
import pandas as pd
import numpy as np
import quandl, math
import datetime

# Machine Learning
from sklearn import preprocessing, cross_validation, svm
from sklearn.linear_model import LinearRegression

#Visualization
import matplotlib
import matplotlib.pyplot as plt
%matplotlib inline
matplotlib.style.use('ggplot')
```

We refined features further based on general understanding of financial data (this step is optional if dealing with unfamiliar concepts). For instance, instead of dealing with High and Low separately, we created volatility percentages as my new features as shown below:

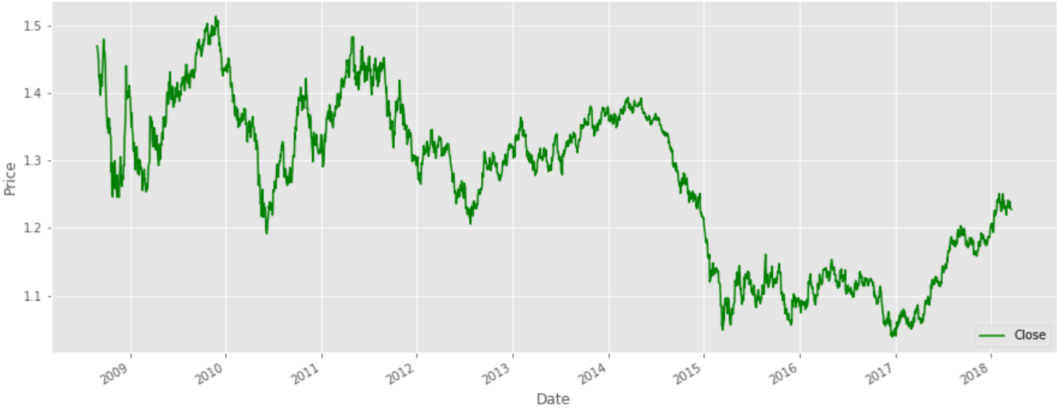


```
# Discarding features that aren't useful
df = df[['Open','High', 'Low', 'Close', 'Volume']]

# define a new feature, HL_PCT
df['HL_PCT'] = (df['High'] - df['Low'])/(df['Low']*100)

# define a new feature percentage change
df['PCT_CHNG'] = (df['Close'] - df['Open'])/(df['Open']*100)

df = df[['Close', 'HL_PCT', 'PCT_CHNG', 'Volume']]

print(df.head(1))
```

```
             Close    HL_PCT  PCT_CHNG  Volume
Date
2008-08-28  1.4693  0.000095 -0.000016   15462
```

We plotted our features as a function of dates, which are saved in the index of our data frame. Since the shares prices are almost linearly rising with time, linear regression should give us a reasonably good prediction!

Price



Price

Figure 3.18 EUR/USD prices

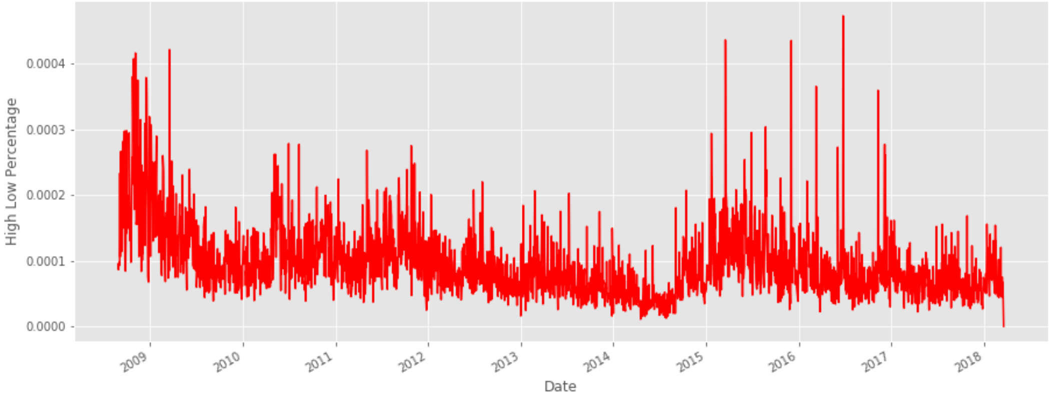


Figure 3.19 High Low %

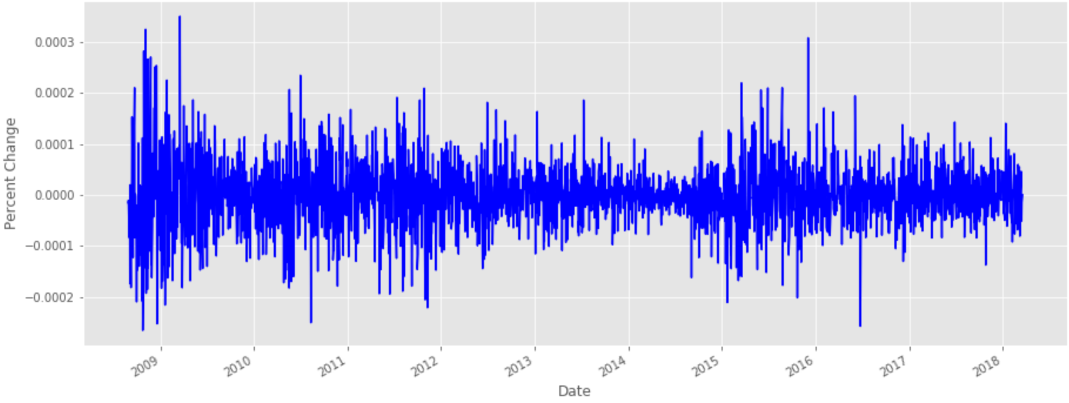


Figure 3.20 percent change

We chose the forecast of *close*, *forecast_out* after next 100 days for our label (the entity that we want to predict). This is completely flexible, the smaller the value of *forecast_out*, more accurate would be the model. An important thing to note here is that once we have shifted our data according to number of days in forecast (say *n*) to create our column *'label'*, we will end up with Nan's in last *n* rows of column *'label'*.

Finally, we try out Linear Regression on our data set by dividing it into *train* and *test* data.

We then plot the predicted prices as a function of dates. The piece of code below just adds dates for the predicted days.

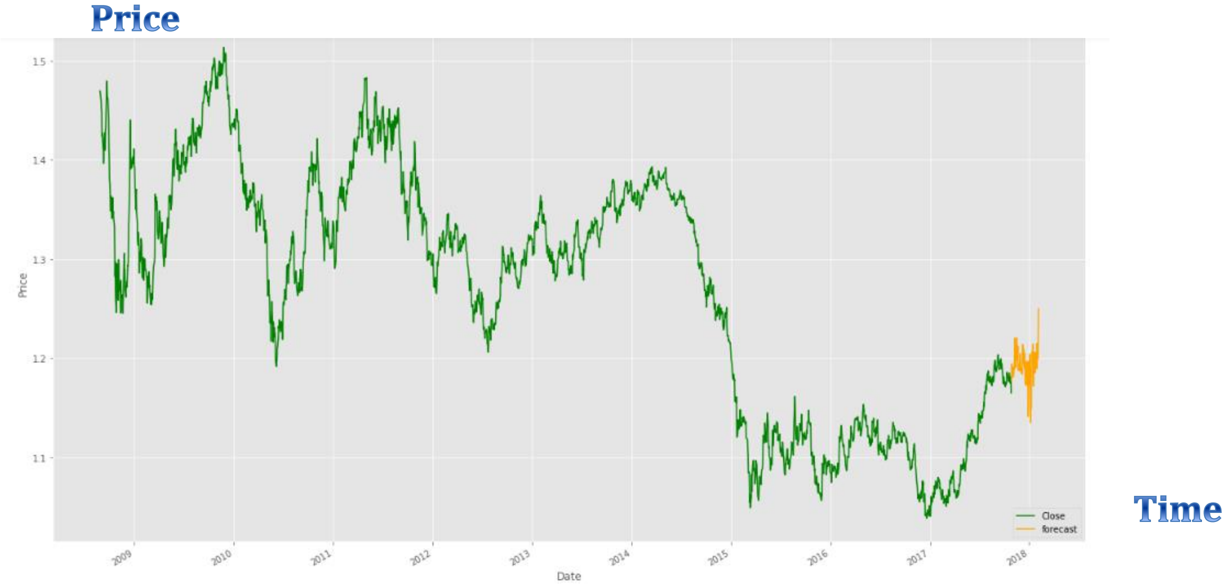


Figure 3.21 Learning Regression

There it is! The prediction of stock prices for the next 30 days, by using linear regression using another trick.

3_4 _ Summary

We tried different algorithms of machine learning and deep learning ant we found it promising for stock markets . For trading purposes , we need more trials to choose the appropriate one .

Conclusion

* Research Questions

This study aimed to determine whether or not it is possible to make a profitable stock trading scheme using machine learning and deep learning , to compare algorithms and their performance, investigate whether a machine learning algorithm improves when given predictions from other algorithms as features and discuss whether it is possible at all to predict the stock market. The main research question was :

1- Is it possible to make a profit on the Stock Exchange using machine learning ?

   As we can recall ,predicting stocks is hard . There are numbers of well cited papers insisting that the stock market is governed by random walk . Some say it is impossible to make any meaningful prediction of the stocks, and that the best strategy of investing is buying stocks at random . A few researchers state that although price changes may not be strictly random, the interdependence is so slight that it is impossible to make a profit . Anyway it is a question that is challenging to answer, and drawing a definite conclusion seems almost impossible. There are simply too many unknown factors that may influence the stock market.

   On one hand, the results yielded show that there are machine learning algorithms that are profitable in the test period, and can be optimized to a much greater extent, which likely will increase potential profit yielded. This shows that there is promise in predicting stocks with machine learning. There are, however, some underlying problems with predicting the stock market and knowing whether the test period is representative for any future time period.

2- Does the performance of machine learning algorithms predicting stocks vary?

   As noted when the question first was posed, it may seem obvious that different machine learning algorithms perform differently, however, if the stock market and individual stocks follow a random walk as many claim, it may not be the case. And as we observed before , there were some algorithms from some schemes that outperformed the others. But behind the top performers there were relatively small differences between the profit estimates. And if we scrutinize our results even further, we can, as we recall from the previous section, see that there are often not significant differences between many of the presented results. One may therefore easily think that much of the profit may be a result of chance and coming from the limitations of the profit estimate. This is an important finding, as it may be used as an argument that it is possible to predict the stock market. This is because one can argue that a market that can be predicted could not be entirely governed by random walk in the test period.

3- Will Machine Learning Algorithm will perform better when other machine learning algorithms predictions are included as a feature?

The experiments did not provide any statistical significant evidence that the performance of the algorithms improved when getting the extra features of other algorithms predictions, nor did it show that any ensemble learning scheme outperformed the other algorithms. The initial idea of the ensemble learners being better at handling problems with overfitting have not been proven true; this may be because the cross-validation making overfitting less of an issue. It seems that we may draw the conclusion that these ensemble schemes will not improve their performance by getting the extra features, however, more extensive test should be performed before making a final conclusion.

4- Is Binary Prediction suitable for a stock market problem?

The results presented give reason to believe, or at least imply, that binary prediction is suitable for making a profit predicting the stock market, as the results show a possible profit. One can also say that it is suitable because every profit making stock scheme boils down to the question; should I own this stock or not. The need for more extensive research appears obvious when attempting to answer all of the research question. There is simply not enough data and tests to conclude with absolute certainty one way or the other.

* Future work

This work can be used as the basis for several future research directions, some of which are listed below

- Feature selection

One of the more apparent problems with stock market prediction is attempting to find an optimal set of features. There is almost an infinite number of possible factors that may influence the price of a stock, but by including too many features one can run into other problems such as the curse of dimensionality and finding correlations without causation . Finding an optimal set of features may be a way of improving the performance of the machine learning algorithms. And one could use deeper domain knowledge to select more fitting features, since it is, as noted, highly unlikely that the features selected in this thesis were optimal.

- Parameter Optimization

Every one of the algorithms tested have a wide set of parameters that can be optimized. Perhaps some of the simple ensemble learners would have performed much better simply by changing the threshold. Since machine learning algorithms so often get a big improvement in performance by optimizing parameters, this would be an obvious and smart move for further research, since an increase in performance may imply that one can make a profit in the stock market using machine learning .

- Other Problems

For the sub problem of testing and comparing the different machine learning algorithms against each other, it is not enough to use a single data set to determine whether there is a difference. The algorithms should also be tested on completely different data-sets as the performance might vary a great deal. This would also open up for applying different performance measures that may better show the difference between the algorithms.

- Time Frames and Markets

It is known that many of the automated trading schemes operating today use quite different time frames and resolutions for their predictions. High frequency trading is getting increasingly popular and might also be suited for machine learning. Also long term predictions may be profitable and may certainly decrease risk, and should therefore also be tested, as it may give a more concise answer to whether it is possible to yield profits over extended time periods on the stock market. Other stock markets can also provide valuable insight into the research questions. stock markets may very well be more applicable for machine learning predictions and should therefore be tested. Applying the machine learning algorithms on different stock markets over other time frames is needed in order to decide with certainty whether it is possible to predict the stock market using machine learning.

- Trading Strategy Improvement

The most important improvement would be to have an improved, sophisticated trading strategy which would take into account all the previous history and decide when to buy or sell not only based on the prediction but in another learning algorithm.

To summarise , this work provides both an overview of the principal ML techniques used to predict stock market prices and an empirical study about the application of ML to predict stock markets .

References

[1] Navin Kumar Manaswi : ' Deep Learning with Applications Using Python', Apress , 2018 .

[2] LESLIE TIONG CHING OW : ' FINANCIAL TIME SERIES PREDICTION USING MACHINE LEARNING ALGORITHMS ', MSc in Computer Science, Sunway University, 2013.

[3] Eleftherios Soulas, Dennis Shasha : ' Online Machine Learning Algorithms For Currency Exchange Prediction ', Technical Report TR-2013-953 , NYU New York, 2013 .

[4] Taiwo Oladipupo Ayodele: ' Types of Machine Learning Algorithms ', University of Portsmouth United Kingdom .

[5] FRANÇOIS CHOLLET: ' Deep Learning with Python', Manning Publications, 2018 .

[6] Yuxi Liu : ' Python Machine Learning by example ', Packt .

[7] Hal Daume : ' A course in machine learning ',Published by TODO, 2015 .

[8] PETER HARRINGTON: ' Machine Learning in Action', Manning Publications, 2012 .

[9] NEGAR FAZELI: 'Machine Learning to Uncover Correlations Between Software Code Changes and Test Results-Master's thesis', University of Gothenburg Sweden, 2017 .

[10] EMIL ANDERSSON, RICKARD ENGLUND : 'Machine Learning for Technical Information Quality Assessment-Master of Science Thesis', University of Gothenburg Sweden, 2016 .

[11] André Dinis Oliveira : 'Forecasting Stock Markets Using Machine Learning-Master Thesis', Universidade Nova de Lisboa Portuguesa , 2016 .

[12] Magnus Olden : 'Predicting Stocks with Machine Learning -Master of Science Thesis', University of Oslo , 2016 .

[13] Ankita Garg: 'Forecasting exchange rates using machine learning models with time-varying volatility-Master of Science Thesis', Linköping University .

[14] Jonathan Millin : 'In Search of Structure: Unsupervised Learning in Foreign Exchange-Master of Science Thesis', University of Edinburgh, 2010 .

[15] LESLIE TIONG CHING OW : 'FINANCIAL TIME SERIES PREDICTION USING MACHINE LEARNING ALGORITHMS-Master of Science Thesis', Sunway University, 2013 .

[16] Yuxing Yan: 'Python for Finance', Packt , 2017 .

[17] Yves Hilpisch : 'Python for Finance', Oreilly , 2015 .

[18] Ankita Thakur: 'Python Real-World Data Science', Packt , 2016 .

[19] Sebastian Raschka: 'Python Machine Learning', Packt , 2015 .

[20] Sebastian Raschka: 'Python Machine Learning', Packt , 2015 .

[21] David M Cutler, James M Poterba and Lawrence H Summers. 'What moves stock prices?' In: The Journal of Portfolio Management 15.3 , pp. 4–12 ,(1989).

[22] Richard A Ajayi and Mbodja Mougou. 'On the dynamic relation between stock prices and exchange rates'. In: Journal of Financial Research 19.2 , pp. 193–207 ,(1996).

[23] Nicholas Apergis and Stephen M Miller. 'Do structural oilmarket shocks affect stock prices?' In: Energy Economics 31.4 , pp. 569–575 ,(2009).

[24] Yasushi Hamao, Ronald W Masulis and Victor Ng. 'Correlations in price changes and volatility across international stock markets'. In: Review of Financial studies 3.2, pp. 281–307 , (1990).

[25] A Ronald Gallant, Peter Eric Rossi and George Tauchen. 'Stock prices and volume'. In: Review of Financial studies 5.2 , pp. 199–242 ,(1992).

[26] Werner FM Bondt and Richard Thaler. 'Does the stock market overreact?' In: The Journal of finance 40.3 , pp. 793–805 ,(1985).

[27] Jerold B Warner, Ross L Watts and Karen H Wruck. 'Stock prices and top management changes'. In: Journal of financial Economics 20 , pp. 461–492 ,(1988).

[28] Grant McQueen and V Vance Roley. 'Stock prices, news, and business conditions'. In: Review of financial studies 6.3 , pp. 683–707 ,(1993).

[29] Markus Konrad Brunnermeier. Asset pricing under asymmetric information: Bubbles, crashes, technical analysis, and herding. Oxford University Press, 2001.

[30] Frank Cross. 'The behavior of stock prices on Fridays and Mondays'. In: Financial analysts journal 29.6, pp. 67–69, (1973).

[31] David Hirshleifer, Tyler Shumway et al. 'Good day sunshine: Stock returns and the weather'. In: Journal of finance 58.3 (2003).

[32] Eugene F Fama. 'The behavior of stock-market prices'. In: Journal of business , pp. 34–105 , (1965).

[33] Andrew W Lo and A Craig MacKinlay. 'Stock market prices do not follow random walks: Evidence from a simple specification test'. In: Review of financial studies 1.1 , pp. 41–66 ,(1988).

[34] Eugene F Fama. 'The behavior of stock-market prices'. In: Journal of business , pp. 34–105, (1965).

[35] Rajiv Sant and Mir A Zaman. 'Market reaction to Business Week 'Inside Wall Street'column: a self-fulfilling prophecy'. In: Journal of Banking & Finance 20.4 , pp. 617–643 ,(1996).

[36] Nison, S. 'Constructing the Candlesticks' . In *Japanese Candlestick*. New York: New York Institute of Finance , (pp. 21 – 26) , (1991).

[37] Bar Charts (OHLC). (2001). *Incrediblecharts*. Retrieved September 20, 2012, from
http://www.incrediblecharts.com/technical/bar_charts.php

[38] Using Technical Indicators. (2009). *Learn Stock Options Trading.com*. Retrieved September 12, 2012, from http://www.learn-stock-options-trading.com/technicalindicators.html

[39] Introduction to Technical Indicators and Oscillators. (2012). *StockCharts.com*. Retrieved October 30, 2012, from http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators: Introduction_to_tech#momentum_oscillators

[40] Jason Brownlee :
https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/

Appendix A

Other investment options beside forex :

_ Stocks

A stock of a corporation is an equity stake, or more simply: a stock is an ownership share in a corporation. A stock market is an aggregation or gathering of buyers and sellers of stocks and other financial instruments, a place where financial instruments are traded.

The stock can be bought in the primary or secondary market. When the company issues the stock for the first time , also called public issue , anyone can subscribe to this issue.
There may be some restrictions regarding the number , the mode of payment and place where the applications can be submitted. The company issues a prospectus , a detailed document giving complete information regarding the company including the time frame
for the project, utilization of the funds, future prospects, and risks perceived by the management and so on. The prospective applicants are advised to study this document carefully. The public issue is kept open for a few days, enabling the interested persons to apply. After all the applications have been received the shares are issued within
the stipulated time. The company may also invite applications in case of substantial expansion, although such public issues are few and far between.

The other market is the secondary market. Huge transactions take place in this market every working day. Here the existing shareholders sell their shares to buyers. To purchase or sell shares in this market a person has to register himself with a broker or a broker house, authorized to operate in this market; the shares are quoted on a stock exchange and this information is widely available. The intimation to purchase or sell (quantity and price) has to be confirmed to the broker. Some brokerage commission has to be paid. After the necessary formalities, which may take at most a few days, the transaction is completed. The shares are kept in a depository and the details are given to the account holder periodically . The advantage of the secondary market is that the past performance of the company is available for study.

While investing in stocks it is necessary to remember that liquidity is low. Only funds not likely to be needed urgently should be invested. It is absolutely essential to study the background and the past performance of the company. The performance should be compared with the performance of the competitors. To minimize the risks, it is advisable to have diversified stocks. One must devote time to study the trends and the market movement of stocks. Stock markets these days follow a global trend. Investors monitor not only NYSE & NASDAQ but also FTSE, NIKKEI, HANG SENG as well as DAX and CAC .

A Stock Exchange is a stock market where brokers and traders buy, sell or exchange publicly listed financial instruments. Commonly stock exchanges provide a way for brokers and traders to exchange financial instruments. Traditionally stock exchanges were physical places, often referred to as the floor, where stock-brokers and -traders exchanged stocks for other stocks or money. These days nearly all stock trades take place through electronic communication. Most stock exchanges work as an institution that allows for trading certain stocks and other financial instruments through a near instant electronic trading system. Most stock exchanges use a continuous auction principle. This principle includes an instant execution of stock orders as they are received by the market. By operating with the continuous principle and rapid electronic orders, modern day stock exchanges are driven solely by supply and demand. There are now hundreds of stock exchanges throughout the world. Some stock exchanges, like Oslo Stock Exchange, include all of the listed stocks in a country or region, while others like NASDAQ and NYSE are more specialized in certain types of corporations and industries. As stated, most of the stock exchanges in the world have automated the trading process, but some stock exchanges like NYSE and some other smaller stock exchanges, still have a floor where stocks can be traded.

_ Bonds

A bond is defined as a long-term promissory note with stipulated interest supported by a consideration or under seal secured by a mortgage . Bonds hold the promise of stipulated interest on a long-term basis. There is a guarantee for the performance. Those issued by the Governments are also termed securities. The issuing Government or Federal Government , in the case of issue by State Government or Local Authority, guarantees the payment . Companies issue debentures. These may be secured by a charge on specific assets of the company . To ensure proper compliance of the regulations and proper upkeep and maintenance of the assets, a trust is formed or trusties are appointed. Even debt instruments issued by companies are covered under the broad term BOND for the purpose of investments. It is compulsory for such companies to get a rating from the recognized Rating Agencies. This helps in estimating the repaying capacity of the company. Triple A (AAA)is the highest rating. The interest on a bond can be fixed for the entire period, or it can be floating .

The floating rate will be linked to either the bank rate or some other independently determined rate such as LIBOR. In general, the safety of the investment and the regular income from the interest are assured . The market price of the bonds does not fluctuate widely only on the market. This ensures liquidity .

A bond-holder is a secured creditor. He has legal rights to sue the company in case of default. Bonds maintain a balance of safety, yield and liquidity. The returns in investments from bonds over a period of time are likely to yield lower returns than the stock market. In several financial applications, such as stock trading, past data hold much information, which , if studied can produce meaningful results. The whole story lies in the ways one utilizes the research for techniques and algorithms and so exploit those data, using them to derive more appropriate future decisions and/or explaining the past .

_ Mutual Funds

An individual investor who wishes to invest in stock but has limited money. On the other hand , the different stocks being traded in the stock market are quite large. When an opportunity arises to purchase some stock , he may not have the liquidity necessary capital. He may not be able to study the trends in stock market. He may not be able to analyse the movement of prices in the stock market . It may be difficult for him to visualize the future prospects of different categories of industries. He may not be able to analyse the performance of individual companies and the changes in their management. In short very few persons can have the time, knowledge and skills to take the best advantage of opportunities that arise in the stock market. Mutual funds are basically investment companies, which continuously sell and buy stock. Anyone can participate in its activities by investing in the mutual fund. The investment company, usually a trust, manages the total capital available to a mutual fund. All the stock owned, by this company, valued at the market price, is the net asset value or NAV. This amount divided by the total number of units issue, will be the NAV per unit. The Mutual Fund Company continuously sells the units and repurchases its units on a daily basis by announcing NAV daily. The Mutual Fund Company will buy the units from the investor at his option at any time at the NAV. For managing the fund, the company will charge some commission called load. This can be charged either at the time of selling or at the time of repurchase. It can be seen that by investing in a mutual fund one can get the benefit from the broader market and the expertise of the professional management. The fund manager of AMC observes the stock market all the time, trying to get the best yield for the investors. Mutual funds state specific investment objectives in their prospectus. The main type or objectives are growth, balanced income, and industry specific funds.

Growth funds possess diversified portfolios of common stocks in the hope a portfolio of stocks, and bonds . This achieves both capital gains and dividend along with interest income. Income funds concentrate heavily on high interest and high dividend yielding securities. Industry specific funds invest in portfolios of selected industries. This appeals to investors who are extremely optimistic about the prospects of these few industries. One should be willing to assume the risks associated with such a concentration of investment. As happened in information technology bad performance results in huge losses. Sometimes the same company may have a family of mutual funds. The investors may be allowed to shift from a fund with one objective to a fund with a different objective for a fee.

_ Forces that move stock prices

Since this is a thesis about predicting stock prices, an important part of it will be attempting to understand and utilize the relationship between stock prices and various factors. One can read countless theses, papers and books on forces that move the stock prices, and still get none the wiser, or at least not fully understand or know half of what goes into the pricing of stocks. Macroeconomics , psychological effects , politics, news, country borders, the corporation's current financial are just some of the factors that affect the price of a stock.

Due to the limitations of this thesis, it would not be feasible to account for all of them, but efforts will be made to a least include the most important short term factors. Movements in stock prices can be looked at both short term and long term. The terms short term and long term are not academically defined.

David M. Cutler tries in his paper "What moves stock prices" to determine what factors that go into the stock price and estimates the fraction of the variance in stock returns that can be attributed to different kinds of news. His paper is about short term changes. First the paper examines what effect macroeconomic news have on the stock prices. The conclusion is that macroeconomic news cannot explain more than one third of the variance [3]. In the same paper he also explores political events and other news, and conclude that every type of news they have looked into effects the stock price. Economists like to talk about ideal scenarios; in an ideal world the stock price is fully explicable by a corporation's future cash flow and discount. Unfortunately, the world is far from ideal and numerous research papers such as Cutler's have shown that other factors go into the pricing of stock [4].

Exchange rates and currencies are two of the more obvious factors for transnational companies. The relationship between stock prices and exchange rates are shown rigorously in [5]. Raw material prices, such as oil prices or aluminum prices, have also been shown to effect the pricing of stocks [6]. Other stock markets also have to be taken into consideration [7].The Volume of stocks being traded [8]. News on the company can make massive impacts on the stock price [9], as can changes in a corporation's management [10]. Macro financial news, such as news about changes in interest rates and changes in inflation can move stock prices with an amplified force [11] . Even speculations on Internet forums may change the volume of traded stock and its prices.

That something seemingly peripheral and insignificant as an Internet forum post can move the stock price of a billion dollar corporation leads us into the perhaps most significant short term factor for stocks, the psychological effects. Stock market trends often begin with bubbles and end in crashes. Some researchers regard this as an example of herd behavior, as investors are driven by greed in bubbles and fear in crashes. Traders join the herd of other traders in rushes to get

[3] David M Cutler, James M Poterba and Lawrence H Summers. 'What moves stock prices?' In: The Journal of Portfolio Management 15.3 (1989), pp. 4–12.

[4] David M Cutler, James M Poterba and Lawrence H Summers. 'What moves stock prices?' In: The Journal of Portfolio Management 15.3 (1989), pp. 4–12.

[5] Richard A Ajayi and Mbodja Mougou. 'On the dynamic relation between stock prices and exchange rates'. In: Journal of Financial Research 19.2 (1996), pp. 193–207.

[6] Nicholas Apergis and Stephen M Miller. 'Do structural oilmarket shocks affect stock prices?' In: Energy Economics 31.4 (2009), pp. 569–575.

[7] Yasushi Hamao, Ronald W Masulis and Victor Ng. 'Correlations in price changes and volatility across international stock markets'. In: Review of Financial studies 3.2 (1990), pp. 281–307.

[8] A Ronald Gallant, Peter Eric Rossi and George Tauchen. 'Stock prices and volume'. In: Review of Financial studies 5.2 (1992), pp. 199–242.

[9] Werner FM Bondt and Richard Thaler. 'Does the stock market overreact?' In: The Journal of finance 40.3 (1985), pp. 793–805.

[10] Jerold B Warner, Ross L Watts and Karen H Wruck. 'Stock prices and top management changes'. In: Journal of financial Economics 20 (1988), pp. 461–492.

[11] Grant McQueen and V Vance Roley. 'Stock prices, news, and business conditions'. In: Review of financial studies 6.3 (1993), pp. 683–707.

in and out of the market [12]. Greed, fear and herd mentality are just three examples of psychological effects that play a part in governing the stock market. Other, less intuitive, factors also play a role. As an example stock have been shown to move differently on Mondays and Fridays [13]. And just like a nice sunny day puts a smile on your face, it has also been known to make traders more optimistic [14] .

Which forces that move the stock market is no simple question. It has been shown that there are many, many different forces that can change the pricing of stocks. This sub chapter was an attempt to give an overview of some of the most important factors. Now we need to narrow them down to a selection of parameters that may be used within the limitation of a thesis.

_ Predictability

Are the movements of stock prices predictable? . Some researchers suggest that stock prices move by the theory of random walk, that is that the future path of the price of a stock is not any more predictable than random numbers [15]. However, Stock prices do not follow random walks [16] is the title of a heavily cited paper. The authors of the paper claim that considerable empirical evidence exists that show that stock returns are to some extent predictable. This means that we can make the basic assumption that past behaviour of a stock's price is rich in information, and may show signs of future behaviour. Some claim to have shown that history is repeated in patterns, and that some of the patterns tend to recur in the future. And since there are patters, it is possible through analysis and modelling to develop an understanding of such patterns. These patterns can further be used to predict the future behaviour of stock prices [17].

[12] Markus Konrad Brunnermeier. Asset pricing under asymmetric information: Bubbles, crashes, technical analysis, and herding. Oxford University Press, 2001.

[13] Frank Cross. 'The behavior of stock prices on Fridays and Mondays'. In: Financial analysts journal 29.6 (1973), pp. 67–69.

[14] David Hirshleifer, Tyler Shumway et al. 'Good day sunshine: Stock returns and the weather'. In: Journal of finance 58.3 (2003).

[15] Eugene F Fama. 'The behavior of stock-market prices'. In: Journal of business (1965), pp. 34–105.

[16] Andrew W Lo and A Craig MacKinlay. 'Stock market prices do not follow random walks: Evidence from a simple specification test'. In: Review of financial studies 1.1 (1988), pp. 41–66.

[17] Eugene F Fama. 'The behavior of stock-market prices'. In: Journal of business (1965), pp. 34–105.

Academically, economists cannot seem to agree with each other on whether or not stock prices move by random walk or not. Both supporters of random walk theory and supporters of predictable movements claim to have shown empirically that their theory is correct. And since researchers cannot seem to agree on the predictability of the movements of stock prices, one can investigate the more practical side of this question. There are certainly numerous of anecdotal stories of people succeeding in predicting the stock market; an example is Nicolas Darvas, a Hungarian dancer who in 1960 published the book How I made $2,000,000 in the stock market ,where he claimed to have recognized patterns in the movements of stocks that eventually lead him to great wealth. Other examples are the thousands of Technical Traders that have made a big impact on the stock market for at least 40 years, and the emerging market of automated trading schemes often known as stock robots. With only anecdotal evidence, one should be careful making generalizations (leave that to the machine learners), and for every successful stock prediction, there might be an opposite story of loss and bankruptcy.

So is the stock market predictable? It depends on which researchers you believe use the most correct methodology for their research, and even then the best answer to the question is, perhaps. What we can conclude is that there are certainly a lot of people that believe that the stock market is predictable, which coincidentally might be what makes the stock market predictable. Traders' belief in themselves and experts might create a selffulfilling prophecy, like when the magazine Business Week recommends a stock, that stock gives abnormally high returns [18].

Even if the entirety of the stock market is not predictable, one can still create profitable trading schemes focusing only on parts of the stock market or certain time periods. For example has a trading scheme focusing on only buying "Business Week's" recommended stock in periods been shown to outperform the overall market . Purely buying and selling stocks based on current oil prices, may perhaps yield great returns. And as previously stated, if currency changes have impact on

[18] Rajiv Sant and Mir A Zaman. 'Market reaction to Business Week 'Inside Wall Street'column: a self-fulfilling prophecy'. In: Journal of Banking & Finance 20.4 (1996), pp. 617–643.

certain stocks, it might be possible to outperform the market by acting quicker on macroeconomical news than most competitors.

Appendix B

Python And Libraries used

_ Generalities

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

_ History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

_ Python Features

Python's features include

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** – Python's source code is fairly easy-to-maintain.
- **A broad standard library** – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** – Python provides interfaces to all major commercial databases.
- **GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below –

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

_ Some of Python Libraries for Data Science and Machine Learning

_ Core Libraries

1. NumPy

When starting to deal with the scientific task in Python, one inevitably comes for help to Python's SciPy Stack, which is a collection of software specifically designed for scientific computing in Python (do not confuse with SciPy library, which is part of this stack, and the community around this stack). This way we want to start with a look at it. However, the stack is pretty vast, there is more than a dozen of libraries in it, and we want to put a focal point on the core packages (particularly the most essential ones).

The most fundamental package, around which the scientific computation stack is built, is NumPy (stands for Numerical Python). It provides an abundance of useful features for operations on n-arrays and matrices in Python. The library provides vectorization of mathematical operations on the NumPy array type, which ameliorates performance and accordingly speeds up the execution.

2. SciPy

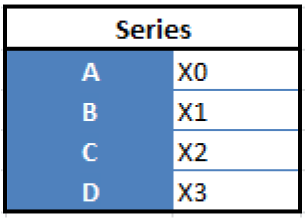
SciPy is a library of software for engineering and science. Again you need to understand the difference between SciPy Stack and SciPy Library. SciPy contains modules for linear algebra, optimization, integration, and statistics. The main functionality of SciPy library is built upon NumPy, and its arrays thus make substantial use of NumPy. It provides efficient numerical routines as numerical integration, optimization, and many others via its specific submodules. The functions in all submodules of SciPy are well documented—another coin in its pot.

3. Pandas

Pandas is a Python package designed to do work with “labeled” and “relational” data simple and intuitive. Pandas is a perfect tool for data wrangling. It designed for quick and easy data manipulation, aggregation, and visualization.

There are two main data structures in the library:

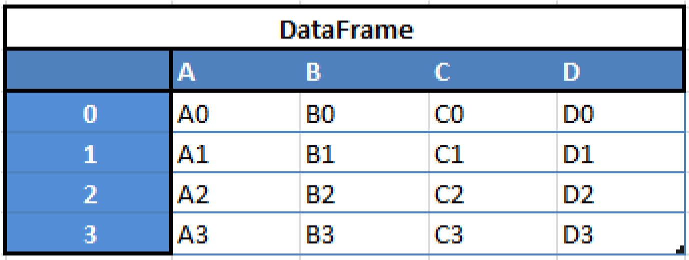
“Series”—one-dimensional



Series	
A	X0
B	X1
C	X2
D	X3

Tableau 3 pandas1

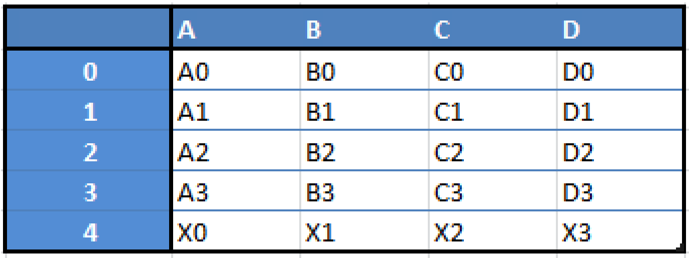
“Data Frames”, two-dimensional



DataFrame				
	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1
2	A2	B2	C2	D2
3	A3	B3	C3	D3

Tableau 4 pandas2

For example, when you want to receive a new Dataframe from these two types of structures, as a result you will receive such DF by appending a single row to a DataFrame by passing a Series:



	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1
2	A2	B2	C2	D2
3	A3	B3	C3	D3
4	X0	X1	X2	X3

Tableau 5 pandas3

Here is just a small list of things that you can do with Pandas:

- Easily delete and add columns from DataFrame
- Convert data structures to DataFrame objects
- Handle missing data, represents as NaNs
- Powerful grouping by functionality

_ Visualization

4. Matplotlib

Another SciPy Stack core package and another Python Library that is tailored for the generation of simple and powerful visualizations with ease is Matplotlib. It is a top-notch piece of software which is making Python (with some help of NumPy, SciPy, and Pandas) a cognizant competitor to such scientific tools as MatLab or Mathematica.

However, the library is pretty low-level, meaning that you will need to write more code to reach the advanced levels of visualizations and you will generally put more effort, than if using more high-level tools, but the overall effort is worth a shot.

With a bit of effort you can make just about any visualizations:

- Line plots
- Scatter plots
- Bar charts and Histograms
- Pie charts
- Stem plots
- Contour plots
- Quiver plots

- Spectrograms

There are also facilities for creating labels, grids, legends, and many other formatting entities with Matplotlib. Basically, everything is customizable.

The library is supported by different platforms and makes use of different GUI kits for the depiction of resulting visualizations. Varying IDEs (like IPython) support functionality of Matplotlib.

There are also some additional libraries that can make visualization even easier.

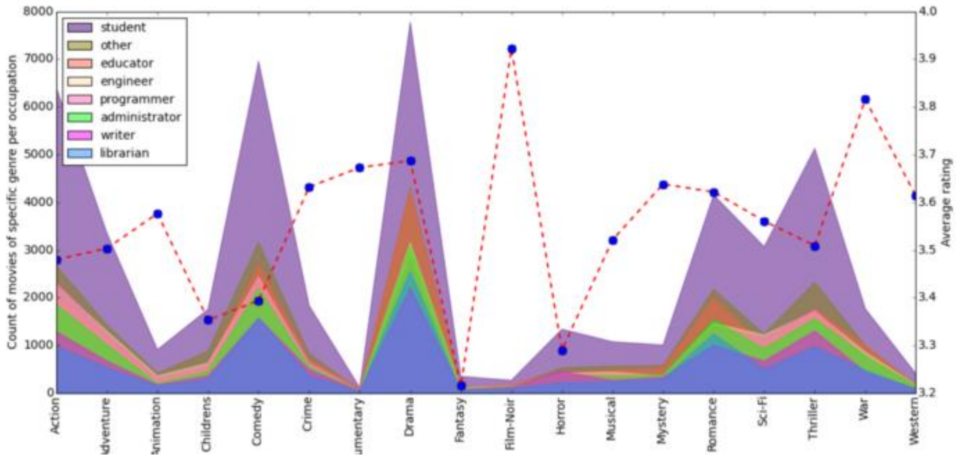


Figure B.1 Matplotlib

5. Seaborn

Seaborn is mostly focused on the visualization of statistical models; such visualizations include heat maps, those that summarize the data but still depict the overall distributions. Seaborn is based on Matplotlib and highly dependent on that.

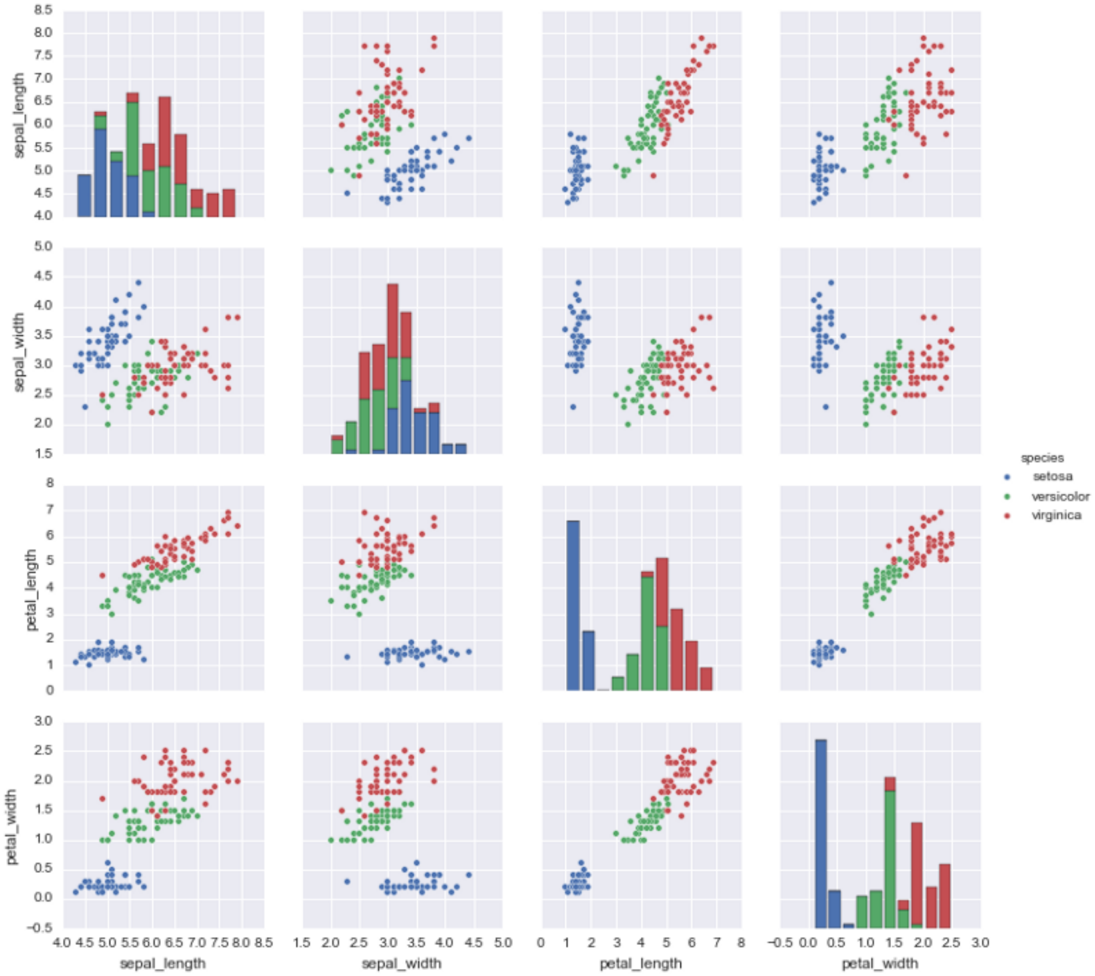


Figure B.2 Seaborn

6. Bokeh

Another great visualization library is Bokeh, which is aimed at interactive visualizations. In contrast to the previous library, this one is independent of Matplotlib. The main focus of Bokeh, as we already mentioned, is interactivity and it makes its presentation via modern browsers in the style of Data-Driven Documents

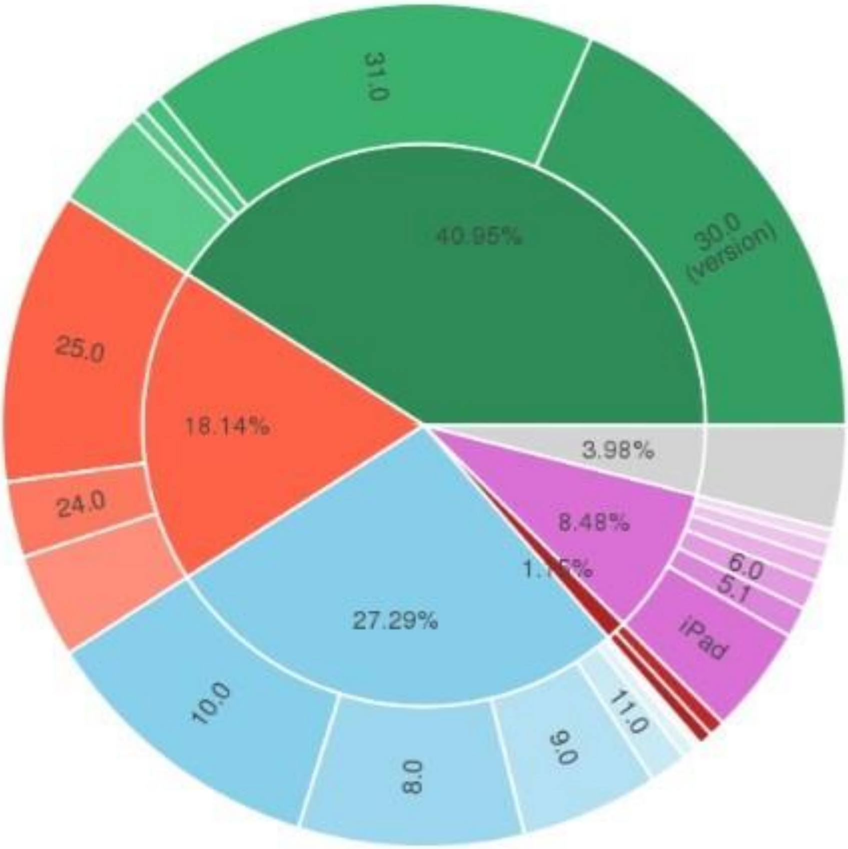


Figure B.3 Bokeh

7. Plotly

Finally, a word about Plotly. It is rather a web-based toolbox for building visualizations, exposing APIs to some programming languages (Python among them). There is a number of robust, out-of-box graphics on the plot.ly website. In order to use Plotly, you will need to set up your API key. The graphics will be processed server side and will be posted on the internet, but there is a way to avoid it.

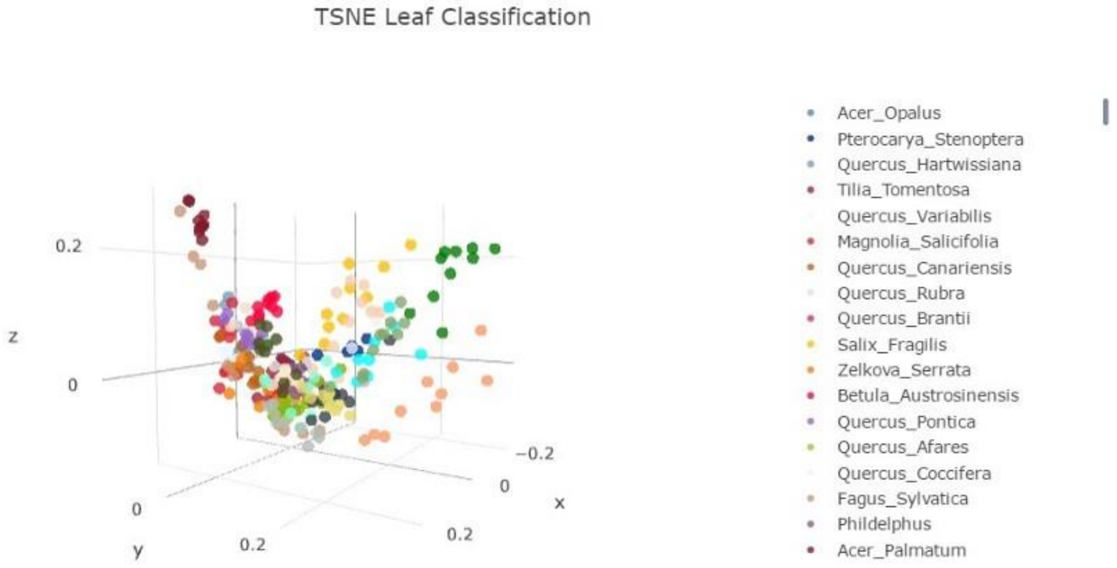


Figure B.4 Plotly

_ Machine Learning

8. SciKit-Learn

Scikits are additional packages of SciPy Stack designed for specific functionalities like image processing and machine learning facilitation. In the regard of the latter, one of the most prominent of these packages is scikit-learn. The package is built on the top of SciPy and makes heavy use of its math operations.

The scikit-learn exposes a concise and consistent interface to the common machine learning algorithms, making it simple to bring ML into production systems. The library combines quality code and good documentation, ease of use and high performance and is de-facto industry standard for machine learning with Python.

_ Deep Learning—Keras / TensorFlow / Theano

In the regard of Deep Learning, one of the most prominent and convenient libraries for Python in this field is Keras, which can function either on top of TensorFlow or Theano. Let's reveal some details about all of them.

9.Theano

Theano is a Python package that defines multi-dimensional arrays similar to NumPy, along with math operations and expressions. The library is compiled, making it run efficiently on all architectures. Originally developed by the Machine Learning group of Université de Montréal, it is primarily used for the needs of Machine Learning.

The important thing to note is that Theano tightly integrates with NumPy on low-level of its operations. The library also optimizes the use of GPU and CPU, making the performance of data-intensive computation even faster .Efficiency and stability tweaks allow for much more precise results with even very small values, for example, computation of log(1+x) will give cognizant results for even smallest values of x.

10. TensorFlow

Coming from developers at Google, it is an open-source library of data flow graphs computations, which are sharpened for Machine Learning. It was designed to meet the high-demand requirements of Google environment for training Neural Networks and is a successor of DistBelief, a Machine Learning system, based on Neural Networks. However, TensorFlow isn’t strictly for scientific use in border’s of Google—it is general enough to use it in a variety of real-world application.

The key feature of TensorFlow is their multi-layered nodes system that enables quick training of artificial neural networks on large datasets. This powers Google’s voice recognition and object identification from pictures.

11. Keras

It is an open-source library for building Neural Networks at a high-level of the interface, and it is written in Python. It is minimalistic and straightforward with high-level of extensibility. It uses Theano or TensorFlow as its backends, but Microsoft makes its efforts now to integrate CNTK (Microsoft's Cognitive Toolkit) as a new back-end.

The minimalistic approach in design aimed at fast and easy experimentation through the building of compact systems.

Keras is really eased to get started with and keep going with quick prototyping. It is written in pure Python and high-level in its nature. It is highly modular and extendable. Notwithstanding its ease, simplicity, and high-level orientation, Keras is still deep and powerful enough for serious modeling.

The general idea of Keras is based on layers, and everything else is built around them. Data is prepared in tensors, the first layer is responsible for input of tensors, the last layer is responsible for output, and the model is built in between.

_ Natural Language Processing

12. NLTK

The name of this suite of libraries stands for Natural Language Toolkit and, as the name implies, it used for common tasks of symbolic and statistical Natural Language Processing. NLTK was intended to facilitate teaching and research of NLP and the related fields (Linguistics, Cognitive Science Artificial Intelligence, etc.) and it is being used with a focus on this today.

The functionality of NLTK allows a lot of operations such as text tagging, classification, and tokenizing, name entities identification, building corpus tree that reveals inter and intra-sentence dependencies, stemming, semantic reasoning. All of the building blocks allow for building complex research systems for different tasks, for example, sentiment analytics, automatic summarization.

13. Gensim

It is an open-source library for Python that implements tools for work with vector space modeling and topic modeling. The library designed to be efficient with large texts, not only in-memory processing is possible. The efficiency is achieved by the using of NumPy data structures and SciPy operations extensively. It is both efficient and easy to use.

Gensim is intended for use with raw and unstructured digital texts. Gensim implements algorithms such as hierarchical Dirichlet processes (HDP), latent semantic analysis (LSA) and latent Dirichlet allocation (LDA), as well as tf-idf, random projections, word2vec and document2vec facilitate examination of texts for recurring patterns of words in the set of documents (often referred as a corpus). All of the algorithms are unsupervised—no need for any arguments, the only input is corpus.

_ Data Mining . Statistics

14. Scrapy

Scrapy is a library for making crawling programs, also known as spider bots, for retrieval of the structured data, such as contact info or URLs, from the web.

It is open-source and written in Python. It was originally designed strictly for scraping, as its name indicate, but it has evolved in the full-fledged framework with the ability to gather data from APIs and act as general-purpose crawlers.

The library follows famous Don't Repeat Yourself in the interface design—it prompts its users to write the general, universal code that is going to be reusable, thus making building and scaling large crawlers.

The architecture of Scrapy is built around Spider class, which encapsulates the set of instruction that is followed by the crawler.

15. Statsmodels

As you have probably guessed from the name, statsmodels is a library for Python that enables its users to conduct data exploration via the use of various methods of estimation of statistical models and performing statistical assertions and analysis.

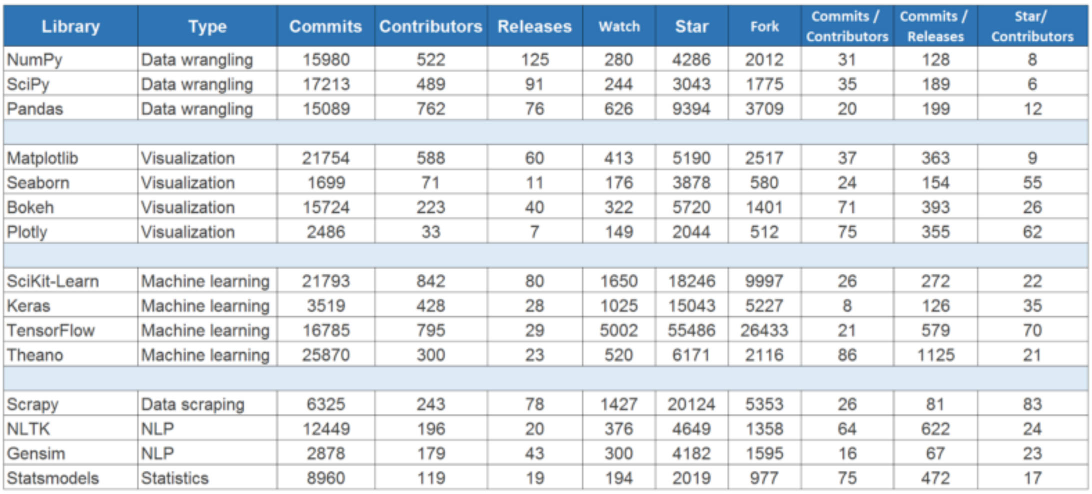
Among many useful features are descriptive and result statistics via the use of linear regression models, generalized linear models, discrete choice models, robust linear models, time series analysis models, various estimators.

The library also provides extensive plotting functions that are designed specifically for the use in statistical analysis and tweaked for good performance with big data sets of statistical data.

_ Conclusions

These are the libraries that are considered to be the top of the list by many data scientists and engineers and worth looking at them as well as at least familiarizing yourself with them.

And here are the detailed stats of Github activities for each of those libraries:



Library	Type	Commits	Contributors	Releases	Watch	Star	Fork	Commits / Contributors	Commits / Releases	Star/ Contributors
NumPy	Data wrangling	15980	522	125	280	4286	2012	31	128	8
SciPy	Data wrangling	17213	489	91	244	3043	1775	35	189	6
Pandas	Data wrangling	15089	762	76	626	9394	3709	20	199	12
Matplotlib	Visualization	21754	588	60	413	5190	2517	37	363	9
Seaborn	Visualization	1699	71	11	176	3878	580	24	154	55
Bokeh	Visualization	15724	223	40	322	5720	1401	71	393	26
Plotly	Visualization	2486	33	7	149	2044	512	75	355	62
SciKit-Learn	Machine learning	21793	842	80	1650	18246	9997	26	272	22
Keras	Machine learning	3519	428	28	1025	15043	5227	8	126	35
TensorFlow	Machine learning	16785	795	29	5002	55486	26433	21	579	70
Theano	Machine learning	25870	300	23	520	6171	2116	86	1125	21
Scrapy	Data scraping	6325	243	78	1427	20124	5353	26	81	83
NLTK	NLP	12449	196	20	376	4649	1358	64	622	24
Gensim	NLP	2878	179	43	300	4182	1595	16	67	23
Statsmodels	Statistics	8960	119	19	194	2019	977	75	472	17

ActiveWizards.com
28.04.2017

Tableau 6 Python libraries

Of course, this is not the fully exhaustive list and there are many other libraries and frameworks that are also worthy and deserve proper attention for particular tasks.

A great example is different packages of SciKit that focus on specific domains, like SciKit-Image for working with images.

_ Why choose Python for AI Projects?

Python provides a huge list of benefits to all. The usage of Python is such that it cannot be limited to only one activity. Its growing popularity has allowed it to enter into some of the most popular and complex processes like Artificial Intelligence (AI), Machine Learning (ML), natural language processing, data science etc. The question is why Python is gaining such momentum in AI? And the answer lies below:

* Less Code:

AI involves algorithms - a LOT of them. Python provides ease of testing - one of the best among competitors. Python helps in easy writing and execution of codes. Python can implement the same logic with as much as 1/5th code as compared to other OOPs languages. Thanks to its interpreted approach which enables check as you code methodology.

* Prebuilt Libraries

Python has a lot of libraries for every need of your AI project. Few names include **Numpy** for scientific computation, **Scipy** for advanced computing and **Pybrain** for machine learning. **AIMA** - Python implementation of algorithms from Russell and Norvig's 'Artificial Intelligence: A Modern Approach' is one of the best library available for Artificial Intelligence till today. Such a dedicated library saves developer’s time spent on coding base level items.

* Support

Python is a completely open source with a great community. There is a host of resources available which can get any developer up to speed in no time. Not to forget, there is a huge community of active coders willing to help programmers in every stage of developing cycle.

* Platform Agnostic

Python provides the flexibility to provide an API from an existing language which indeed provides extreme flexibility. It is also platform independent. With just a few changes in codes, you can get your app up and running in a new OS. This saves developers time in testing on different platforms and migrating code.

* Flexibility

Flexibility is one of the core advantages of Python. With the option to choose between OOPs approach and scripting, Python is suitable for every purpose. It works as a perfect backend and it also suitable for linking different data structures together. The option to check a majority of code in the IDE itself is also a big plus for developers who are struggling between different algorithms.

* Popularity

Python is winning the heart of millennials. Its ease of learning is attracting millennials to learn this language. Though AI Projects need a highly experienced programmer yet Python can smoothen the learning curve. It is practically more easy to look for Python developers than to hunt for LISP or Prolog programmers, particularly in some nations. Its extended libraries and active community with an ever developing and improving code have led it to be one of the hottest languages today.

_ Summary

In this chapter we gave an overview of Python and the most important libraries of it that used in data analysis and machine learning .