ALEXANDRIA HIGHER INSTITUTE OF ENGINEERING AND TECHNOLOGY (AIET)

Web Enabled Intelligent Information System

For Stock Management and Prediction

(E- Stock Market)

by

Ahmed Tarek Mansour

Amr Aly Elzafarani

Belal Hossny Ahmed

David Raouf Sobhy

Mohammed Ahmed Eldefrawy

Mohammed Ahmed Abo Elela

Mohammed Ahmed Dawood

Mohammed Murad Mohammed

Mohammed Hamdy El Tedawy

Marawan Gamal Aly

Mohammed Abd El Menaam Nassar

Mostafa Ahmed AbdElrazek

Mostafa Abd El-Aleem Elsheikh

Computer Engineering

Approved by: Professor Hatem Abdelkader, Project Advisor

TABLE OF CONTENTS

I. ACKNOWLEDGMENTS

We would like to thank Dr. Hatem Abdelkader, Professor of Computer Science and Engineering, for his advisement over this project.

We also would like to thank Dr. Tamer Fouad, Assistant Professor in the Department of Computer Engineering in Alexandria Higher Insatiate of Engineering & Technology (AIET) Head for his efforts with us.

II. ABSTRACT

A web-based stock prediction system is developed based on trading and machine learning algorithms performed on historical stock prices to discover rules for making future predictions and decisions.

In this project , We've built a website of Stock Market Simulation based on back-end SQL database (which is more powerful) of historical stock prices by applying Portfolio Risk Analysis, Technical Analysis with various trading strategies for making decisions such as: Moving Crossover Strategy, Bollinger Bands Indicator. Moreover, We've used two approaches of various machine learning models for stock prediction:

- First Approach: Using KNN, Linear Regression, Logistic Regression, SVM (Support Vector Machine), LSTM Neural Network, Backpropagation Neural Network.

- Second Approach: Using Sentimental Analysis on Twitter news dataset for predicting stock price indicators.

Index terms— Financial Engineering, Quantitative Analysis, Algorithm Trading, Data Science, Machine Learning, Deep Learning

نبذة مختصرة

وذلك عن (Machine Learning) تم تطوير نظام تنبؤ بأسعار اسهم البورصة المبنى على شبكة الانترنت باستخدام تعلم الآلة طريق استخدام البيانات السابقة لأسعار أسهم البورصة لإكتشاف قواعد وعمل تنبؤات مستقبلية. البيانات المدخلة يتم من خلالها .لأسعار أسهم البورصة Moving average حساب المتوسط الحركى الاسبوعى

التى تحتوى SQL في هذا المشروع، قمنا ببناء موقع على شبكة الإنترنت لمحاكاة سوق الأسهم المالية مستندة على قاعدة بيانات ، التحليل الفني مع Portfolio Risk Analysis على مجموعة بيانات لأسعار الأسهم السابقة من خلال تحليل مخاطر المحفظة ، Moving Crossover Strategy استراتيجيات التداول المتنوعة لاتخاذ لقرارات مثل: إستراتيجية نقل الحركة المتقاطعة علاوة على ذلك استخدمنا نهجين من نماذج التعلم الآلي المختلفة التى تتعلم .Bollinger Bands Indicator مؤشر البولنجر باند :نماذج لتنبؤ أسعار أسهم البورصة

- النهج الأول: استخدام KNN ، الانحدار الخطي Linear Regression، الانحدار اللوجستي Logistic Regression، SVM(دعم آلة المتجهات) ، ذاكرة المدى الطويل LSTM للشبكة العصبية ، الشبكة العصبية Backpropagation.
- النهج الثاني: استخدام تحليل الآراء على مجموعة بيانات من أخبار تويتر للتنبؤ مؤشرات أسعار الأسهم.

Chapter One

1 INTRODUCTION

1.1 GOALS

The main goals of this project were to investigate possible investment opportunities, logically weigh their respective risks and benefits, and make educated investment decisions. A detailed understanding of the risks and opportunities for high return presented in each investment opportunity were obtained through extensive research and analysis. This knowledge will aid in the future selection and execution of intelligent investments.

1.2 INTRODUCTION TO STOCK MARKET

1.2.1 What is the Stock Market?

The stock market refers to the collection of markets and exchanges where the issuing and trading of equities or stocks of publicly held companies, bonds, and other classes of securities take place. This trade is either through formal exchanges or over-the-counter (OTC) marketplaces. Also known as the equity market, the stock market is one of the most vital components of a free-market economy. It provides companies with access to capital in exchange for giving investors a slice of ownership.

The stock market consists of two main sections, the primary market, and the secondary market. The primary market is where new issues are first sold through initial public offerings (IPOs). Institutional investors typically purchase most of these shares from investment banks.

The worth of the company going public and the number of shares issued will determine the opening price of the IPO stock. All subsequent trading happens in the secondary market, where participants include both institutional and individual investors. A company uses the money raised from its IPO to grow, but once its stock starts trading, it does not receive funds from the buying and selling of its shares.

Stocks of larger companies are usually traded through exchanges. Such exchanges exist in major cities all over the world, including London and Tokyo. Exchanges are entities that bring

together buyers and sellers in an organized manner. On exchanges, stocks are listed and traded. Today, the execution of most transactions is by electronic means. Even the stocks themselves are mostly held in an electronic format, not as physical certificates.

Regarding market capitalization, the two most prominent stock exchanges in the United States are the New York Stock Exchange (NYSE), founded in 1792 and located on Wall Street, which colloquially is often the synonym for the NYSE, and the Nasdaq, founded in 1971. The Nasdaq originally featured over-the-counter (OTC) securities, but today it lists all types of stocks. Stocks may be listed on either exchange if they meet the listing criteria of the exchange. However, in general technology, firms tend to be listed on the Nasdaq.

The NYSE is still the largest and, arguably, most powerful stock exchange in the world. The Nasdaq has more companies listed, but the NYSE has a market capitalization that is larger than Tokyo, London, and the Nasdaq combined.

The Securities and Exchange Commission (SEC) is the regulatory body charged with overseeing the U.S. stock markets. The SEC is a federal agency, which is independent of the political party in power. Explicitly, the mission of the SEC states it is "to protect investors, maintain fair, orderly, and efficient markets, and facilitate capital formation." Shortly after the stock market crash of 1929, the birth of the Securities and Exchange Commission (SEC) set a goal to restore investor faith in the financial sector.

1.2.2 Stock Market Trading

When you buy a stock, you're buying a piece of the company. When a company needs to raise money, it issues shares. This is done through an initial public offering (IPO), in which the price of shares is set based how much the company is estimated to be worth, and how many shares are being issued. The company gets to keep the money raised to grow its business, while the shares (also called stocks) continue to trade on an exchange, such as the New York Stock Exchange (NYSE).

Traders and investors continue to buy and sell the stock of the company on the exchange, although the company itself no longer receives any money from this type of trading. The company only receives money from the IPO.

Over-the-counter (OTC) and listed securities are the two primary types of securities transacted on stock markets. Listed securities are those stocks traded on exchanges. These securities need to meet reporting regulations of the SEC as well as the requirements of the exchanges where they trade.

Over-the-counter securities are exchanged directly between parties, usually via a dealer network. These securities do not list on any stock market exchange but will show on the pink sheets. Pink sheet security often will not meet the requirements to list on an exchange and tend to have a low float, such as closely held companies or thinly-traded stocks. Also, companies in bankruptcy are typically listed on the pink sheets, as are penny stocks, which are loosely defined as stocks that trade below $5 a share.

OTC securities do not need to comply with SEC reporting requirements, so finding credible information on them can be difficult. This lack of data makes investing in pink sheet securities similar to investing in private companies.

There are many different players associated with the stock market, including stockbrokers, traders, stock analysts, portfolio managers and investment bankers. Each has a unique role, but many of the roles are intertwined and depend on each other to make the market run effectively.

Stockbrokers, also known as registered representatives in the U.S., are the licensed professionals who buy and sell securities on behalf of investors. The brokers act as intermediaries between the stock exchanges and the investors by buying and selling stocks on the investors' behalf.

Stock analysts perform research and rate the securities as buy, sell, or hold. This research gets disseminated to clients and interested parties who decide whether to buy or sell the stock.

Portfolio managers are professionals who invest portfolios, or collections of securities, for clients. These managers get recommendations from analysts and make the buy or sell decisions for the portfolio. Mutual fund companies, hedge funds, and pension plans use portfolio managers to make decisions and set the trading strategies for the money they hold.

Investment bankers represent companies in various capacities, such as private companies that want to go public via an IPO or companies that are involved in pending mergers and acquisitions.

1.2.3 The Stock Market Performance Indicators

If you want to know how the stock market is performing, you can consult an index of stocks for that whole market or that segment of the market. Indexes are used to measure changes in the overall stock market.

There are many different indexes, with each made up of a different pool of stocks. Some indexes may overlap in the stocks they represent. In the United States, examples of indexes include the Dow Jones Industrial Average, NASDAQ Composite Index, Russell 2000, and Standard and Poor's 500 (S&P 500), but there are many more.

The Dow Jones Industrial Average (DJIA) is perhaps the best-known of the indexes. The Dow is made up of the 30 largest companies in the U.S., and the daily Dow shows how their stocks perform on a given day. The Dow average is a price-weighted average. Price-weighted means the value has a basis on the price of the included stocks.

The S&P 500 is comprised of the 500 largest capitalization stocks traded in the U.S.

These two indexes are the most followed measurements of the U.S. stock market. As such, they are the most accepted representatives of the overall American economy. However, there are many other indexes which represent mid- and small-sized U.S. companies, such as the Russell 2000.

In the U.S., the indexes that measure the value of stocks are widely followed. As a financial barometer, the stock market has become an integral and influential part of decision-making for everyone from the average family to the wealthiest executive.

1.2.4 Why is the Stock Market Important?

The stock market allows companies to raise money by offering stock shares and corporate bonds. It lets investors participate in the financial achievements of the companies, making money through dividends. Dividends are cuts of the company's profits as they may payouts. Investors also make a profit by selling appreciated stocks. This is known as a capital gain. Of course, the downside is that investors can lose money as well if the share price falls or if the investor must sell the shares at a loss.

One of the whole points of open exchange is to provide transparency and opportunity for all investors. Furthermore, laws and governing bodies, such as the SEC, exist to "level the playing field" for

investors. However, there are undeniable advantages that institutional investors and professional money managers have over the individual investor.

Advantages of large institutional investors include the timely access to privileged information, full-time research departments, vast amounts of capital to invest, discounts on commissions, transaction fees, and even share prices based on the large dollar amount they invest, political influence, and more significant experience.

While the Internet has been somewhat of an equalizing factor, the reality is that many institutional clients get news and analysis before the public does and can act on information more quickly.

1.2.5 The history of the stocks market

Early stock and commodity markets

The first genuine stock markets didn't arrive until the 1500s. However, there were plenty of early examples of markets which were similar to stock markets.

In the 1100s, for example, France had a system where courretiers de change managed agricultural debts throughout the country on behalf of banks. This can be seen as the first major example of brokerage because the men effectively traded debts.

Later on, the merchants of Venice were credited with trading government securities as earl y as the 13th century. Soon after, bankers in the nearby Italian cities of Pisa, Verona, Genoa, and Florence also began trading government securities.

Selling stocks in coffee shops

Before investors yelled across trade floors and threw order forms into the air, they conducted business in coffee shops. Early stocks were handwritten on sheets of paper, and investors traded these stocks with other investors in coffee shops.

In other words, coffee shops were the first real stock markets due to the fact that investors would visit these markets to buy and sell stocks. Before long, somebody realized that the entire business world would be more efficient if somebody made a dedicated marketplace where businessmen could trade stocks without having to order a coffee or yell across a crowded café.

The first stock market bubble

Nobody really understood the importance of the stock market in those early days. People realized it was powerful and valuable, but nobody truly understood exactly what it would become.

That's why the early days of the stock market were like the Wild West. In London, businesses would open up overnight and issue stocks and shares of some crazy new venture. In many cases, companies were able to make thousands of pounds before a single ship had ever left harbor.

There was no regulation and few ways to distinguish legitimate companies from illegitimate companies. As a result, the bubble quickly burst. Companies stopped paying dividends to investors and the government of England banned the issuing of shares until 1825.

The first stock exchange

Despite the ban on issuing shares, the London Stock Exchange was officially formed in 1801. Since companies were not allowed to issue shares until 1825, this was an extremely limited exchange. This prevented the London Stock Exchange from preventing a true global superpower.

That's why the creation of the New York Stock Exchange (NYSE) in 1817 was such an important moment in history.

The NYSE has traded stocks since its very first day. Contrary to what some may think, the NYSE wasn't the first stock exchange in the United States. The Philadelphia Stock Exchange holds that title. However, the NYSE soon became the most powerful stock exchange in the country due to the lack of any type of domestic competition and its positioning at the center of U.S. trade and economics in New York.

The London Stock Exchange was the main stock market for Europe, while the New York Stock Exchange was the main exchange for America and the world.

Major stock market crashes throughout history

Stock market crashes are an unavoidable side effect of any market where public attitudes play a role.

Most major stock markets have experienced crashes at some point in history. Stock market crashes are by nature preceded by speculative economic bubbles. A stock market crash can occur when speculations are stretched far beyond the actual value of a stock.

There have been a number of major crashes throughout history, including Black Thursday or Terrible Thursday of 1929, which was followed by Black Monday and Black Tuesday. During this crash, the Dow Jones Industrial Average lost 50% of its value, sending America and much of the world into a deep economic depression and wiping out billions of dollars.

Other major stock market crashes include:

- Stock Market Crash of 1973-1974
- Black Monday of 1987
- Dot-com Bubble of 2000
- Stock Market Crash of 2008

All of these crashes pale in comparison to 1929 but still involved double digit percentage losses around the world. The advance of electronic trading has caused many to question the foundations of the stock market, including the theory of rational human conduct, the theory of market equilibrium, and the efficient-market hypothesis.

The stock market crash of 1987 was the first major crash of the electronic trading era and it was notable due to the fact that nobody really saw it coming. It was not predated by major news announcements or world affairs. Instead, it seemed to have just happened with no immediately apparent visible reasons.

The 1987 crash began in Hong Kong, where stock markets fell 45.5% between October 19 and October 31. By the end of October, major stock markets around the world had all experienced double digit collapses. Markets in Australia experienced a 42% drop, for example, while the United States and Canada both suffered losses of about 23%.

When do stock markets close around the world?

One of the many advantages of having stock markets around the world is the fact that there is almost always a market open in some part of the world. Most of the world's stock markets open between 9:00am and 10:00am local time and close between 4:00pm and 5:00pm local time. The NYSE, NASDAQ, TSX, and Shanghai Stock Exchange all open at 9:30.

Some stock markets also take a break for lunch. Four major Asian markets take a break for lunch that lasts for 1 hour to 1.5 hours in the middle of the day. Those markets include the Tokyo Stock Exchange, Hong Kong Stock Exchange, Shanghai Stock Exchange, and Shenzhen Stock Exchange.

What are the largest stock markets in the world today?

The list of the top 10 largest stock markets in the world today indicates the changing roles of various countries throughout history. Today, the top 10 stock markets include markets in highly-developed countries as well as markets in developing parts of Asia.

Here are the top 10 stock markets in the world today ranked by market capitalization:

1. New York Stock Exchange
2. NASDAQ
3. Tokyo Stock Exchange
4. London Stock Exchange Group
5. Euronext
6. Hong Kong Stock Exchange
7. Shanghai Stock Exchange
8. Toronto Stock Exchange
9. Frankfurt Stock Exchange
10. Australian Securities Exchange

Other rising stock markets outside of the top 10 include the Bombay Stock Exchange based in Mumbai, India, as well as the BM&F Bovespa stock exchange based in Sao Paulo, Brazil.

The future of the stock market

Stock markets aren't going away anytime soon. They remain a driving economic force in virtually every country in the world. Analysts aren't entirely sure what the future holds for the stock market, although there are some important things to consider.

First, the NYSE remains the largest and (arguably) the most powerful stock exchange in the entire world. It's so large, in fact, that its market capitalization is larger than Tokyo, London, and NASDAQ combined.

Second, we will likely see stock markets continue to merge over the coming years. Some have even suggested that we'll eventually see a single global stock market, although this appears to be unlikely.

1.2.6 Market Capitalization

Market capitalization refers to the total dollar market value of a company's outstanding shares. Using market capitalization to show the size of a company is important because company size is a basic determinant of various characteristics in which investors are interested, including risk. It is also easy to calculate. A company with 20 million shares selling at $100 a share would have a market cap of $2 billion.

Market Capitalization Ranking

Largest Market Cap Stocks in the S&P 500

Add	Symbol	Market Cap	Price
+	AAPL Apple Inc	955.82B	188.58
+	AMZN Amazon.com Inc	801.85B	1,610.15
+	MSFT Microsoft Corp	766.62B	98.36

Smallest Market Cap Stocks in the S&P 500

Add	Symbol	Market Cap	Price
+	SIG Signet Jewelers Ltd	2.86B	42.63
+	RRC Range Resources Corp	3.75B	15.20
+	NAVI Navient Corp	3.91B	14.53

Large-cap companies typically have a market capitalization of $10 billion or more, Mid-cap companies generally have a market capitalization of between $2 billion and $10 billion, while Companies that have a market capitalization of between $300 million to $2 billion are generally classified as small-cap companies.

Changes in Market Cap

Two main factors can alter company's market cap: significant changes in the price of a stock or when a company issues or repurchases shares. An investor who exercises a large number of warrants can also increase the amount of shares on the market and negatively affect shareholders in a process known as dilution.

Chapter Two

2 FINANCIAL ANALYSIS

2.1 STOCK MARKET ANALYSIS

Stock analysis is the evaluation of a particular trading instrument, an investment sector, or the market as a whole. Stock analysts attempt to determine the future activity of an instrument, sector, or market. It is also referred to market analysis, or equity analysis.

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

There are two main forms of stock analysis:

- **Fundamental analysis** looks at data, both from the company and from its macroeconomic environment, to assess potential profits from trades. It concentrates on data from sources including financial records, economic reports, company assets, and market share. (It is out of scope of our project)

- **Technical analysis** looks at charts of previous price movements in order to predict future ones. It focuses on the study of past market action to predict future price movement.

Both have the same intended outcome: to make the correct buying and selling decisions and choose the correct times to place trades. Some traders will devote all of their time to technical analysis, some to fundamental analysis, and some to a mixture of the two.

2.1.1 Stock Trends

The stock market, just like the price of individual stocks, tends to trend. Indices are prone to move in one general direction or another, until something causes that direction to change. Uptrends are characterized by prices making higher highs and higher lows. Downtrends are characterized by lower price highs and lower price lows.

Using Trendlines

The most common way to identify trends is using trendlines, which connect a series of highs or lows. Uptrends connect a series of higher lows, creating a key support level for future price movements. Downtrends connect a series of lower highs, creating a key resistance level for future price movements. In addition to support and resistance, these trendlines show the overall direction of the trend.

Distribution Days of an Uptrend

Eventually, every up-trending market reaches a day where selling activity overtakes buying. The day when the index closes down from the day before on higher volume (meaning more people are selling than buying) is called a “distribution day.

At the same time, you’re seeing distribution in the market indices, you’ll also likely see leading stocks—the stocks that have been the best price performers in that market cycle—show signs of topping (meaning the price will not go higher and is likely to go down rapidly). If you spot leading stocks exhibiting signs of weakness in their price and volume action, e.g., declining price with a high volume of sellers, you have another reason to suspect that the overall market trend is changing.

Act on Trend Emerging in The Market

After you’ve spotted signs of market weakness, you need to take action. Do not buy any stock. Consider raising some cash by selling your weakest stocks. If the market continues to weaken, you

need to continue to sell stocks and move to cash. You never know how far the downtrend will go. By taking defensive action, you're protecting yourself against the devastating losses that severe corrections can cause.

Follow-Through on a Solid Market Rally

Once the market enters a downtrend, continue to monitor it closely. At some point, it will rally and recover for a few days. Many of these rallies will be fake-outs. What you want to see is confirmation that there has been a change in trend to the upside. This comes in the form of an explosive up day signalling the tone of the market has changed.

Investors should generally consider using a systemized trading strategy but be aware of its many limitations. Trading strategies aren't a guarantee for success, but they may be effective in increasing risk-adjusted returns.

2.1.2 Technical Trading Strategy

Overview

Trading strategy is a set of objective rules defining the conditions that must be met for a trade entry and exit to occur. Trading strategies include specifications for trade entries, including trade filters and triggers, as well as rules for trade exits, money management, timeframes, order types, and other relevant information, which helps trades and investors determine if a trading idea is potentially profitable. A trading strategy, if based on quantifiably specifications, can be analysed based on historical data to project future performance. Technical trading strategies rely on technical indicators to generate trading signals.

Pros and Cons of a Trading Strategy

Trading strategies are a great way to avoid behavioural finance biases and ensure consistent results over time. For example, traders with a specific set of rules governing when to exit a trade will be less likely to succumb to the disposition effect, which causes investors to hold on to stocks that have lost value and sell those that rise in value. Trading strategies can also be stress tested under many different market conditions to ensure consistency.

The downside is that profitable trading strategies are difficult to develop and it's easy to become overly reliant on the strategy. For instance, a trader may curve fit a trading strategy to specific back-

testing data, which can generate a false sense of confidence. The strategy may have performed great based on the past data, but that's no guarantee that it will perform just as well using live market data since the conditions may be different.

2.1.3 Technical Indicators

Overview

Technical indicators are mathematical calculations based on historical trading data, such as price, volume, and open interest, rather than the fundamentals of a business, like earnings, revenue, or profit margins. By analysing historical data, technical analysts use indicators to predict future price movements.

Main Types of Technical Indicators

There are two basic types of technical indicators:

1. **Overlays** - Technical indicators that use the same scale as prices are plotted over the top of the prices on a stock chart. Examples include moving averages and Bollinger Bands®.

2. **Oscillators** - Technical indicators that oscillate between a local minimum and maximum are plotted above or below a price chart. Examples include the MACD or RSI.

Traders often use many different technical indicators when analysing a security. With thousands of different options, traders must choose the indicators that work best for them and familiarize themselves with how they work. Traders may also combine technical indicators with more subjective forms of technical analysis, such as looking at chart patterns, to come up with trade ideas. Technical indicators can also be incorporated into automated trading systems given their quantitative nature.

Examples of Technical Indicators

The following chart shows some of the most common technical indicators, including moving averages, the relative strength index (RSI), and the moving average convergence-divergence (MACD).

In this example, the 50- and 200-day moving averages are plotted over the top of the prices to show where the current price stands relative to its historical averages. The 50-day moving averages is higher than the 200-day moving average in this case, which suggests that the overall trend has been positive. The RSI above the chart shows the strength of the current trend - a neutral 49.07 in this case - and the MACD below the chart shows how the two moving averages have converged or diverged - slightly bearish in this case. (We will discuss that in detail in **Technical Analysis** section)

2.1.4 Technical Analysis

Overview

Technical analysts analyse the financial market as a whole and are primarily concerned with price and volume, as well as the demand and supply factors that move the market. Charts are a key tool for technical analysts as they show a graphical illustration of a stock's trend within a stated time period. Technical analysis based on historical price information is a typical task finance professionals and interested amateurs engage in. On Wikipedia you find the following definition:

In finance, technical analysis is a security analysis methodology for forecasting the direction of prices through the study of past market data, primarily price and volume.

In technical analysis we are dealing in probabilities, never certainties.

Stock Return

Stock Market Returns are the returns that the investors generate out of the stock market. This return could be in the form of profit through trading or in the form of dividends given by the company to its shareholders from time-to-time.

The most common form of generating stock market return is through trading in the secondary market. In the secondary market an investor could earn stock market return by buying a stock at lower price and selling at a higher price.

Hence, we can conclude by mentioning the following points: -

- Stock Market Returns are not fixed or ensured ones.
- Stock Market Returns are subjected to market risks.
- Stock Market Returns could be positive or negative.

Stock Market Returns include profit through trading in the secondary market and dividends given by the company to its shareholders from time-to-time.

Stock Growth

A growth stock is a share in a company whose earnings are expected to grow at an above-average rate relative to the market.

A growth stock usually does not pay a dividend, as the company would prefer to reinvest retained earnings in capital projects. Growth investors choose stocks based on the potential for capital gains, not dividend income, so they can be risky.

Technology companies are typically good examples of growth stocks because the opportunity for advancement is virtually limitless. However, growth stocks also carry a lot of risk because shareholders rely solely on the company's success to generate return on their investment. If the company's growth is not what was expected, shareholders may end up losing money as market confidence wanes and share prices dropped gains, not dividend income, so they can be risky.

Stock Volatility

Volatility is a statistical measure of the dispersion of returns for a given security or market index.

Volatility can either be measured by using the standard deviation or variance between returns from that same security or market index. Commonly, the higher the volatility, the riskier the security.

Investors care about volatility for at least eight reasons:

1. The wider the swings in an investment's price, the harder emotionally it is to not worry.
2. Price volatility of a trading instrument can define position sizing in a portfolio.
3. When certain cash flows from selling a security are needed at a specific future date, higher volatility means a greater chance of a shortfall.
4. Higher volatility of returns while saving for retirement results in a wider distribution of possible final portfolio values.
5. Higher volatility of return when retired gives withdrawals a larger permanent impact on the portfolio's value.
6. Price volatility presents opportunities to buy assets cheaply and sell when overpriced.
7. Portfolio volatility has a negative impact on the Compound Annual Growth Rate (CAGR) of that portfolio.
8. Volatility affects pricing of options, being a parameter of the Black-Scholes-Model.

Backtesting

Back-testing is the process of testing a trading strategy on relevant historical data to ensure its viability before the trader risks any actual capital. A trader can simulate the trading of a strategy over an appropriate period of time and analyse the results for the levels of profitability and risk.

A significant amount of the volume traded in today’s financial market is done by traders that use some sort of computer automation. This is especially true for trading strategies based on technical analysis. Backtesting is an integral part of developing an automated trading system. If the results meet the necessary criteria that are acceptable to the trader, the strategy can then be implemented with some degree of confidence that it will result in profits

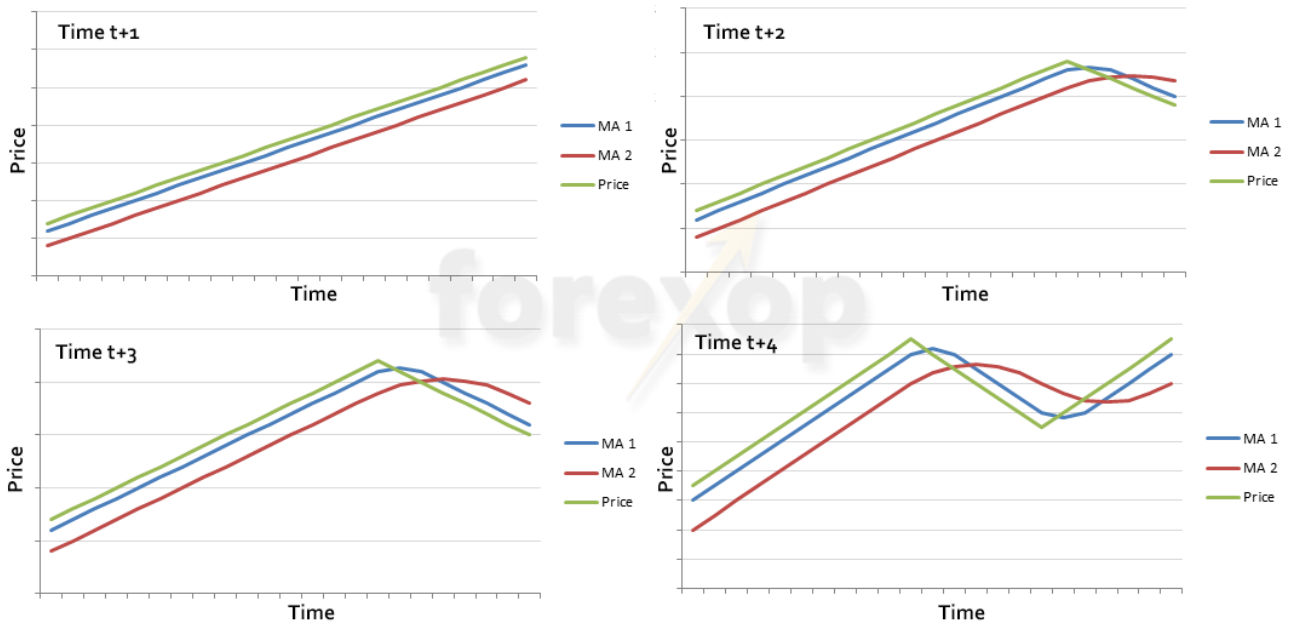
Backtesting can provide plenty of valuable statistical feedback about a given system. Some universal backtesting statistics include:

- Net profit or loss - Net percentage gained or lost
- Volatility measures - Maximum percentage upside and downside
- Averages - Percentage average gain and average loss, average bars held
- Exposure - Percentage of capital invested (or exposed to the market)
- Ratios - Wins-to-losses ratio
- Annualized return - Percentage return over a year
- Risk-adjusted return - Percentage return as a function of risk

Moving Average Crossover Strategy

The Moving Average Crossover strategy is probably the most popular trading strategy in the world, simple to implement using Simple Moving Average or Exponential Moving Average.

The Moving Average Crossover System uses two Moving Averages (MA's), a Fast MA and a Slow MA (The Fast MA has a shorter period value than the Slow MA) for example for the Fast MA lets use a 10 Period Simple Moving Average SMA(10) & for the Slow MA lets use a 40 Period Simple Moving Average SMA(40)



Crossover Events

- A **bullish crossover** occurs when the Fast MA (SMA(10)) crosses ABOVE the Slow MA (SMA(40))
- A **bearish crossover** occurs when the Fast MA (SMA(10)) crosses BELOW the Slow MA (SMA(40))

Crossover Trading Plan

- **Entry:** a common Entry point is on close of the candle when the crossover occurs or on a break of that candles High/Low
- **Stop:** Set stop loss **above** slow moving average, **below** slow moving average support line resistance line or until there is a reversal signal
- **Target:** Fixed, perhaps at 2x Risk-Reward or until there is a reversal signal meaning the Target is "Open"

The Death Cross and the Golden Cross

Perhaps the most common pairing is the 50-day verses the 200-day moving average. When the 50-day cross up through the 200-day moving average, this is said to be a golden cross. It signifies to many the possibility of a new bull market.

On the flip side, when the 50-day moving average crosses down through the 200-day moving average this is said to be a death cross. It's a sign of a market that's turning bearish.

Simple vs exponential moving averages

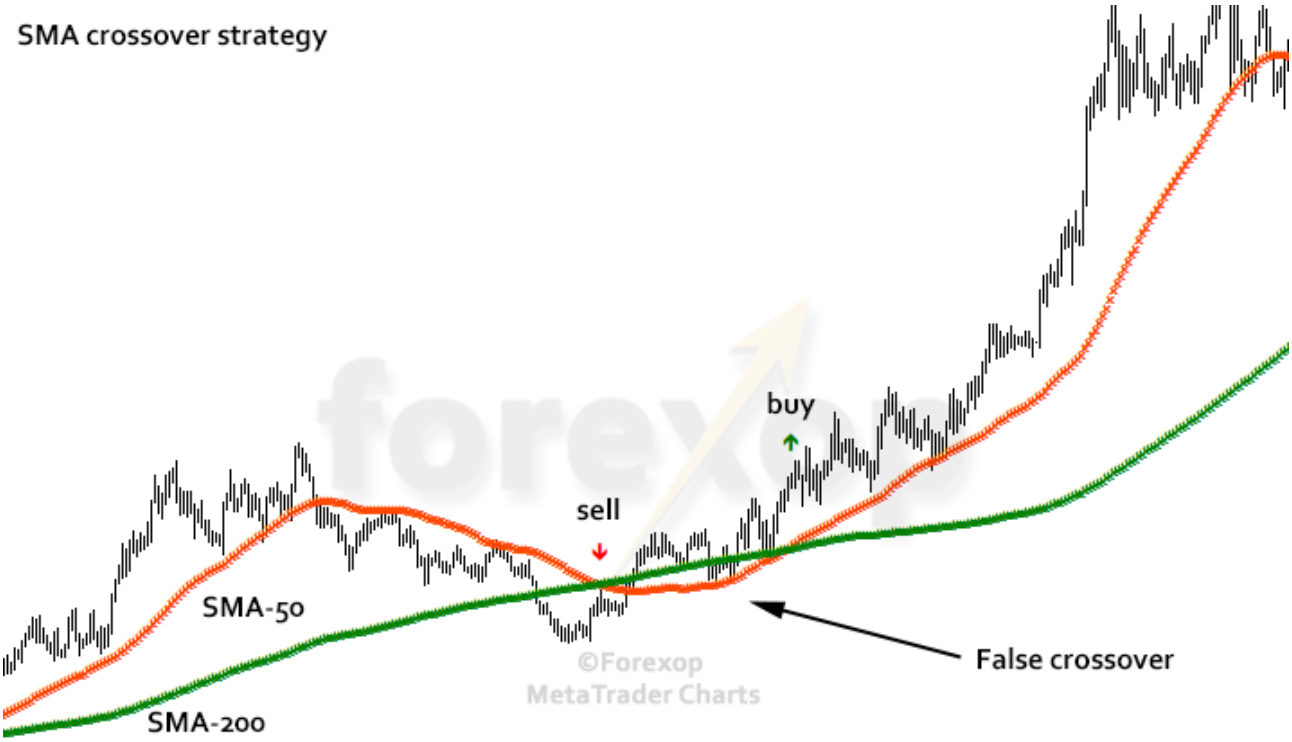
The other decision is in the formula that is used to average prices over time. Simple moving averages apply the same weighting to all prices. On the other hand, exponential moving averages apply higher weighting to more recent prices and lower weighting to price further back in time.

Exponential moving averages would on the face of it seem to be better suited because they will be more responsive to recent price changes. But at the same time this also makes them more responsive to extreme and sudden price moves. This may or may not be a good thing, depending on the market conditions.

Typically, in SMA and EMA the price taken at each interval is the mid-price. Yet this can change and sometimes the open or close price is preferred.

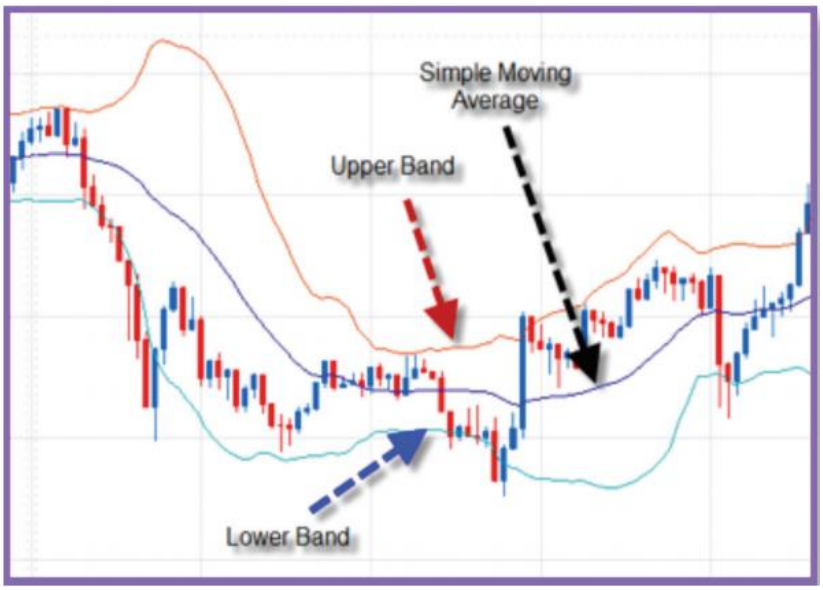
Flase Crossover

One of the complications all crossover strategies need to deal with is that of false crossover signals. False crossings or false crossovers happen when the moving average lines cross one another briefly, but then revert back again.



However, the market turns bullish again and shortly after the false sell signal, a new buy signal is generated. This would almost certainly have resulted in a loss on the short positions, though this could have been recouped if the subsequent buy signal was acted on.

Bollinger Bands®



Bollinger Bands are a technical trading tool created by John Bollinger in the early 1980s. They are a volatility indicator and use the mathematical concept of standard deviations to measure price volatility around a moving average to generate trading signals. During periods of increased fluctuation, the bands will widen to take this into account. When the fluctuation decreases, the bands are tapered for a narrower focus to the price range. The upper band is the standard deviation multiplied by a given factor above the simple moving average, and the lower band is the standard deviation multiplied by the same given factor below the simple moving average.

Interpretation of Bollinger Bands

In isolation Bollinger Bands do not give absolute buy and sell signals. Instead, they indicate whether the price is relatively high or low, allowing for more informed confirmation with other technical indicators.

General Rules of Bollinger Bands

When the price hits the upper or lower bands, if other indicators suggest that price movement shows strength or weakness, this could indicate a continuation. If other indicators do not confirm this movement, it can suggest a reversal.

When the price hits the upper or lower bands, if other indicators suggest that price movement shows strength or weakness, this could indicate a continuation. If other indicators do not confirm this movement, it can suggest a reversal.

Trading Decisions

- Breakouts

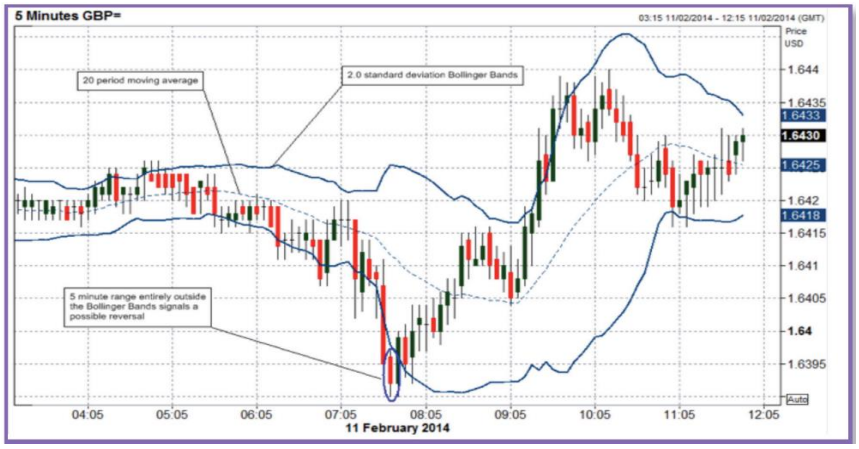
When the Bollinger Bands become very narrow this is a sign that the price is consolidating and volatility has become extremely low. However, this narrowing will often occur just before a significant move in the price. As the pressure builds, there can be a sudden burst of price action often seen, which can be either higher or lower. The trade is placed in the direction of the breakout.

- Reversals:

It is possible to use the upper and lower bands to help identify possible reversal in price. When the daily range is entirely outside the bands this suggests the increased likelihood of a reversal. This signal is strengthened by a second top or bottom being made inside the bands.

- Range Trading:

In a consolidating market it is possible to use the two bands as a basis for support and resistance. The idea would be to then buy as the price hits the bottom band and then sell again when the price hits the top band.



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.

The relative strength index is calculated using the following formula:

$$RSI = 100 - 100 / (1 + RS)$$

Where RS = Average gain of up periods during the specified time frame / Average loss of down periods during the specified time frame.

The RSI provides a relative evaluation of the strength of a security's recent price performance, thus making it a momentum indicator. RSI values range from 0 to 100. The default time frame for comparing up periods to down periods is 14, as in 14 trading days.

Traditional interpretation and usage of the RSI is that RSI values of 70 or above indicate that a security is becoming overbought or overvalued, and therefore may be primed for a trend reversal or corrective pullback in price. On the other side of RSI values, an RSI reading of 30 or below is commonly interpreted as indicating an oversold or undervalued condition that may signal a trend change or corrective price reversal to the upside.

Put Option

A put option is an option contract giving the owner the right, but not the obligation, to sell a specified amount of an underlying security at a specified price within a specified time. This is the opposite of a call option, which gives the holder the right to buy shares.

The value of a put option decreases due to time decay, because the probability of the stock falling below the specified strike price decreases. When an option loses its time value, the intrinsic value is left over, which is equivalent to the difference between the strike price less the stock price. Out-of-the-money and at-the-money put options have an intrinsic value of zero because there would be no benefit of exercising the option. Investors could sell short the stock at the current market price, rather than exercising an out-of-the-money put option at an undesirable strike price, which would produce losses.

Long Put Option and Short Put Option

Long Put Option Example: For example, assume an investor owns one put option on hypothetical stock TAZR with a strike price of $25 expiring in one month. Therefore, the investor has the right to sell 100 shares of TAZR at a price of $25 until the expiration date next month, which is usually the third Friday of the month. If shares of TAZR fall to $15 and the investor exercises the option, the investor could purchase 100 shares of TAZR for $15 in the market and sell the shares to the option's writer for $25 each. Consequently, the investor would make $1,000 (100 x ($25-$15)) on the put option. Note that the maximum amount of potential profit in this example ignores the premium paid to obtain the put option.

Short Put Option Example: Contrary to a long put option, a short put option obligates an investor to take delivery, or purchase shares, of the underlying stock. Assume an investor is bullish on hypothetical stock FAB, which is currently trading at $42.50, and does not believe it will fall below $35 over the next two weeks. The investor could collect a premium by writing one put option on FAB with a strike price of $35 for $1.50. Therefore, the investor would collect a total of $150, or $1.50 * 100. If FAB closes above $35, the investor would keep the premium collected since the options would expire out of the money and be worthless. Conversely, if FAB closes below $35, the investor must purchase 100 shares of FAB at $35, due to the contractual obligation.

Call Option

A call option is an agreement that gives an investor the right, but not the obligation, to buy a stock, bond, commodity or other instrument at a specified price within a specific time period.

It may help you to remember that a call option gives you the right to call in or buy an asset. You profit on a call when the underlying asset increases in price.

An options contract gives the holder the right to buy 100 shares of the underlying security at a specific price, known as the strike price, up until a specified date, known as the expiration date. For example, a single call option contract may give a holder the right to buy 100 shares of Apple stock at a price of $100 until Dec. 31, 2017. As the value of Apple stock goes up, the price of the options contract goes up, and vice versa. Options contract holders can hold the contract until the expiration date, at which

point they can take delivery of the 100 shares of stock or sell the options contract at any point before the expiration date at the market price of the contract at the time.

Tips on Using the RSI Indicator

Sudden large price movements can create false buy or sell signals in the RSI. It is, therefore, best used with refinements to its application or in conjunction with other, confirming technical indicators.

Some traders, in an attempt to avoid false signals from the RSI, use more extreme RSI values as buy or sell signals, such as RSI readings above 80 to indicate overbought conditions and RSI readings below 20 to indicate oversold conditions.

The RSI is often used in conjunction with trend lines, as trend line support or resistance often coincides with support or resistance levels in the RSI reading.

Watching for divergence between price and the RSI indicator is another means of refining its application. Divergence occurs when a security makes a new high or low in price but the RSI does not make a corresponding new high or low value. Bearish divergence, when price makes a new high but the RSI does not is taken as a sell signal. Bullish divergence that is interpreted as a buy signal occurs when price makes a new low, but the RSI value does not. An example of bearish divergence can unfold as follows: A security rises in price to $48 and the RSI makes a high reading of 65. After retracing slightly downward, the security subsequently makes a new high of $50, but the RSI only rises to 60. The RSI has bearishly diverged from the movement of price.

Overbought and Oversold Levels

Oversold is a condition in which the price of an underlying asset has fallen sharply to a level below where its true value resides. This condition is usually a result of market overreaction or panic selling and is generally considered short term in nature. When an asset has been oversold, the price is expected to rebound in an event referred to as a price bounce.

Overbought refers to a situation in which the demand for a certain asset or security unjustifiably pushes the price of that asset or underlying asset to levels that are not justified by fundamentals. Overbought is often a term used in technical analysis to describe a situation in which the price of a security has risen to such a degree - usually on high volume - that an oscillator has reached its upper bounds.

Divergence

Divergence appears on a bar chart when the price of an asset and an indicator, index or other related asset move in opposite directions. In technical analysis, traders make transaction decisions by identifying situations of divergence, where the price of a stock and a set of relevant indicators, such as the money flow index (MFI), are moving in opposite directions.

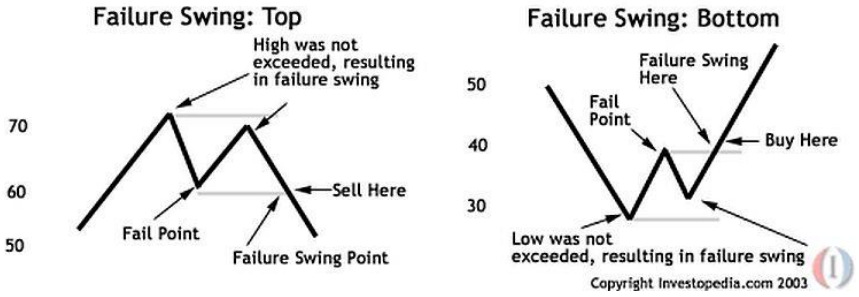
Divergence can last for a long time. Prices may continue to rise even though the RSI is showing a divergence. Therefore, divergence should not be acted on alone. If there is divergence present, it is wise to wait for the price to break in the direction of the divergence before acting. For example, if a stock is in an uptrend and the RSI is moving lower, sell once the price starts to break lower

Failure swings

Failure swings can also be used to spot price reversals. A bullish failure swing forms when RSI moves below 30, rises back above 30 and pulls back again, but holds above the 30 level. The failure swing is complete when the RSI breaks its recent high; this breakout is interpreted as a bullish signal.

A bearish failure swing forms when the RSI moves above 70, pulls back below 70 and rises again, but holds below 70. The failure swing is complete when the RSI breaks its recent low; this breakout is interpreted as a bearish signal.

A failure swing is essentially a more advanced form of the overbought/oversold trading conditions discussed above. The trader is waiting for an oversold condition in which to buy, but instead of buying immediately when the RSI moves back above 30, the trader has the option to wait and see if the RSI holds above the 30 level on the next drop. If it does so and then starts to rise again, the trader buys. The reverse would be true for selling at the 70 levels after the RSI has reached overbought conditions.



RSI Ranges

During an uptrend, RSI tends to stay between different levels than during a downtrend. This makes sense, because the RSI is measuring gains versus losses. In an uptrend, there will be more gains, keeping the RSI at higher levels. In a downtrend, the RSI will tend to stay at lower levels.

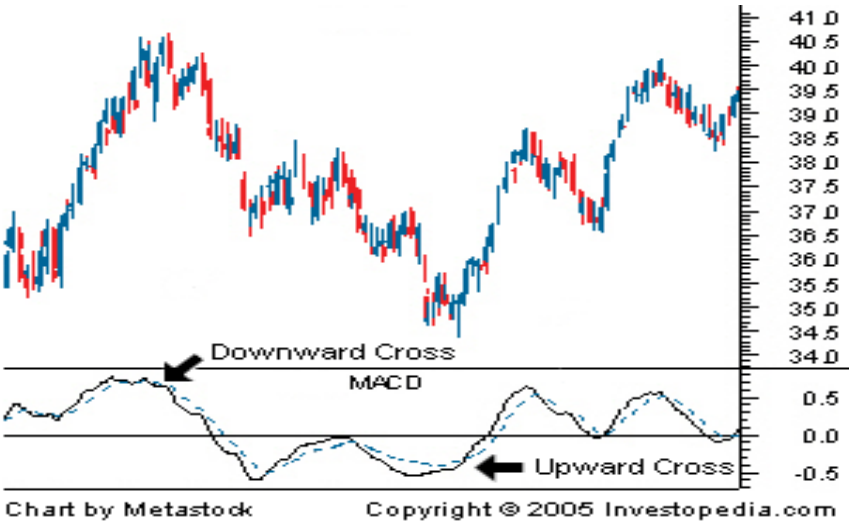
During an uptrend, the RSI tends to stay above 30 and should hit 70 often. During a downtrend, it is rare to see the RSI above 70, and the indicator frequently hits 30 or below. These guidelines can aid in determining trend strength and spotting potential reversals.

For example, if the RSI isn't able to reach 70 on a number of price swings in a row during an uptrend, and then drops below 30, the trend has weakened and could be reversing lower.

The reverse is true for a downtrend. If the downtrend is unable to reach 30 or below, and then rallies above 70, that downtrend has weakened and could be reversing to the upside.

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. 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.



MACD Trading Methods

1. **Crossovers** - As shown in the chart above, when the MACD falls below the signal line, it is a bearish signal, which indicates that it may be time to sell. Conversely, when the MACD rises above the signal line, the indicator gives a bullish signal, which suggests that the price of the asset is likely to experience upward momentum. Many traders wait for a confirmed cross above the signal line before entering into a position to avoid getting "faked out" or entering into a position too early, as shown by the first arrow.

2. **Divergence** - When the security price diverges from the MACD, it signals the end of the current trend. For example, a stock price that is rising and a MACD indicator that is falling could mean that the rally is about to end. Conversely, if a stock price is falling and the MACD is rising, it could mean that a bullish reversal could occur in the near-term. Traders often use divergence in conjunction with other technical indicators to find opportunities.

3. **Dramatic Rise** - When the MACD rises dramatically - that is, the shorter moving average pulls away from the longer-term moving average - it is a signal that the security is overbought and will soon return to normal levels. Traders will often combine this analysis with the Relative Strength Index (RSI) or other technical indicators to verify overbought or oversold conditions.

Traders also watch for a move above or below the zero line because these signals the position of the short-term average relative to the long-term average. When the MACD is above zero, the short-term average is above the long-term average, which signals upward momentum. The opposite is true when the MACD is below zero. As you can see from the chart above, the zero line often acts as an area of support and resistance for the indicator.

2.2 FINANCIAL VISUALIZATION

2.2.1 Line Chart

Line charts are the most basic type of chart because it represents only the closing prices over a set period. The line is formed by connecting the closing prices for each period over the timeframe. While this type of chart doesn't provide much insight into intraday price movements, many investors consider the closing price to be more important than the open, high, or low price within a given period. These charts also make it easier to spot trends since there's less 'noise' happening compared to other chart types.

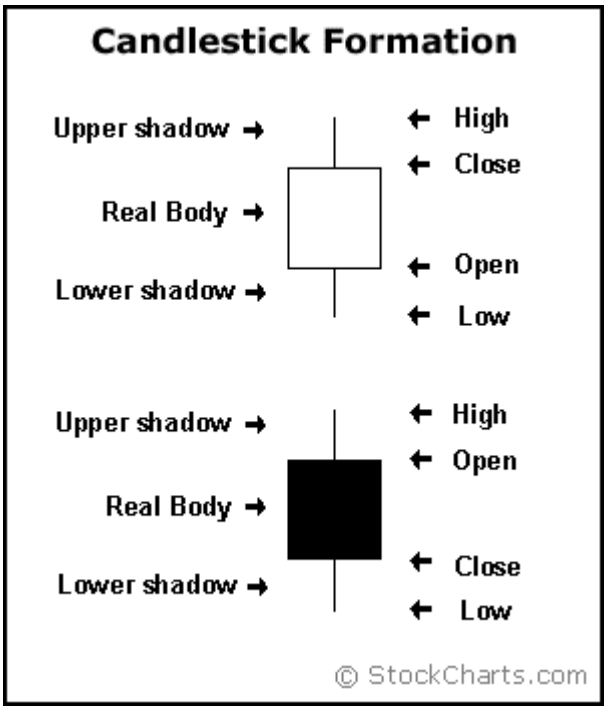
2.2.2 Bar Chart

Bar charts expand upon the line chart by adding the open, high, low, and close – or the daily price range, in other words – to the mix. The chart is made up of a series of vertical lines that represent the price range for a given period with a horizontal dash on each side that represents the open and closing prices. The opening price is the horizontal dash on the left side of the horizontal line and the closing price is located on the right side of the line. If the opening price is lower than the closing price, the line is often shaded black to represent a rising period. The opposite is true for a falling period, which is represented by a red shade.

2.2.3 Candlstick Chart

Candlestick charts originated in Japan over 300 years ago but have since become extremely popular among traders and investors. Like a bar chart, candlestick charts have a thin vertical line showing the price range for a given period that's shaded different colors based on whether the stock ended higher or lower. The difference is a wider bar or rectangle that represents the difference between the opening and closing prices.

Formation



In order to create a candlestick chart, you must have a data set that contains open, high, low and close values for each time period you want to display. The hollow or filled portion of the candlestick is called "the body" (also referred to as "the real body"). The long thin lines above and below the body represent the high/low range and are called "shadows" (also referred to as "wicks" and "tails"). The high is marked by the top of the upper shadow and the low by the bottom of the lower shadow. If the stock closes higher than its opening price, a hollow candlestick is drawn with the bottom of the body representing the opening price and the top of the body representing the closing price. If the stock closes lower than its opening price, a filled candlestick is drawn with the top of the body representing the opening price and the bottom of the body representing the closing price.

Compared to traditional bar charts, many traders consider candlestick charts more visually appealing and easier to interpret. Each candlestick provides an easy-to-decipher picture of price action. Immediately a trader can compare the relationship between the open and close as well as the high and low. The relationship between the open and close is considered vital information and forms the essence of candlesticks. Hollow candlesticks, where the close is greater than the open, indicate buying pressure. Filled candlesticks, where the close is less than the open, indicate selling pressure.

2.3 TECHNOLOGY IN FINANCE

In a sense, technology per se is nothing special to financial institutions (as compared, for instance, to industrial companies) or to the finance function (as compared to other corporate functions, like logistics). However, in recent years, spurred by innovation and also regulation, banks and other financial institutions like hedge funds have evolved more and more into technology companies

instead of being just financial intermediaries. Technology has become a major asset for almost any financial institution around the globe, having the potential to lead to competitive advantages as well as disadvantages. Some background information can shed light on the reasons for this development.

2.3.1 Technology Spending

Banks and financial institutions together form the industry that spends the most on technology on an annual basis. The following statement therefore shows not only that technology is important for the financial industry, but that the financial industry is also really important to the technology sector:

> Banks will spend 4.2% more on technology in 2014 than they did in 2013, according to IDC analysts. Overall IT spend in financial services globally will exceed $430 billion in 2014 and surpass $500 billion by 2020, the analysts say.
>
> —Crosman 2013

Large, multinational banks today generally employ thousands of developers that maintain existing systems and build new ones. Large investment banks with heavy technological requirements show technology budgets often of several billion USD per year.

2.3.2 Technology as Enabler

The technological development has also contributed to innovations and efficiency improvements in the financial sector:

> Technological innovations have contributed significantly to greater efficiency in the derivatives market. Through innovations in trading technology, trades at Eurex are today executed much faster than ten years ago despite the strong increase in trading volume and the number of quotes … These strong improvements have only been possible due to the constant, high IT investments by derivatives exchanges and clearing houses.
>
> — Deutsche Börse Group 2008

As a side effect of the increasing efficiency, competitive advantages must often be looked for in ever more complex products or transactions. This in turn inherently increases risks and makes risk management as well as oversight and regulation more and more difficult. The financial crisis of 2007 and 2008 tells the story of potential dangers resulting from such developments. In a similar vein, “algorithms and computers gone wild” also represent a potential risk to the financial markets; this materialized dramatically in the so-called flash crash of May 2010, where automated selling led to large intraday drops in certain stocks and stock indices (cf. http://en.wikipedia.org/wiki/2010_Flash_Crash).

2.3.3 Technology and Talent as Barriers to Entry

On the one hand, technology advances reduce cost over time. On the other hand, financial institutions continue to invest heavily in technology to both gain market share and defend their

current positions. To be active in certain areas in finance today often brings with it the need for large-scale investments in both technology and skilled staff.

Not only is it costly and time-consuming to build a full-fledged derivatives analytics library, but you also need to have enough experts to do so. And these experts have to have the right tools and technologies available to accomplish their tasks.

A quote about the early days of Long-Term Capital Management (LTCM), formerly one of the most respected quantitative hedge funds—which, however, went bust in the late 1990s—further supports this insight about technology and talent:

> Meriwether spent $20 million on a state-of-the-art computer system and hired a crack team of financial engineers to run the show at LTCM, which set up shop in Greenwich, Connecticut. It was risk management on an industrial level.
>
> — Patterson 2010

The same computing power that Meriwether had to buy for millions of dollars is today probably available for thousands. On the other hand, trading, pricing, and risk management have become so complex for larger financial institutions that today they need to deploy IT infrastructures with tens of thousands of computing cores.

2.3.4 Ever-Increasing Speeds, Frequencies, Data Volumes

There is one dimension of the finance industry that has been influenced most by technological advances: the speed and frequency with which financial transactions are decided and executed. The recent book by Lewis (2014) describes so-called flash trading —i.e., trading at the highest speeds possible—in vivid detail.

On the one hand, increasing data availability on ever-smaller scales makes it necessary to react in real time. On the other hand, the increasing speed and frequency of trading let the data volumes further increase. This leads to processes that reinforce each other and push the average time scale for financial transactions systematically down:

> Renaissance's Medallion fund gained an astonishing 80 percent in 2008, capitalizing on the market's extreme volatility with its lightning-fast computers. Jim Simons was the hedge fund world's top earner for the year, pocketing a cool $2.5 billion.
>
> — Patterson 2010

Thirty years' worth of daily stock price data for a single stock represents roughly 7,500 quotes. This kind of data is what most of today's finance theory is based on. For example, theories like the modern portfolio theory (MPT), the capital asset pricing model (CAPM), and value-at-risk (VaR) all have their foundations in daily stock price data.

In comparison, on a typical trading day the stock price of Apple Inc. (AAPL) is quoted around 15,000 times—two times as many quotes as seen for end-of-day quoting over a time span of 30 years. This brings with it a number of challenges:

- Data processing
- Analytics speed
- Theoretical foundations

All these challenges can in principle only be addressed by modern technology.

2.3.5 The Rise of Real-Time Analytics

Financial and data analytics has a close relationship to the insight that speeds, frequencies, and data volumes increase at a rapid pace in the industry. In fact, real-time analytics can be considered the industry's answer to this trend.

Roughly speaking, "financial and data analytics" refers to the discipline of applying software and technology in combination with (possibly advanced) algorithms and methods to gather, process, and analyze data in order to gain insights, to make decisions, or to fulfill regulatory requirements, for instance. Examples might include the estimation of sales impacts induced by a change in the pricing structure for a financial product in the retail branch of a bank. Another example might be the large-scale overnight calculation of credit value adjustments (CVA) for complex portfolios of derivatives trades of an investment bank.

There are two major challenges that financial institutions face in this context:

- Big data

  Banks and other financial institutions had to deal with massive amounts of data even before the term "big data" was coined; however, the amount of data that has to be processed during single analytics tasks has increased tremendously over time, demanding both increased computing power and ever-larger memory and storage capacities.

- Real-time economy

  In the past, decision makers could rely on structured, regular planning, decision, and (risk) management processes, whereas they today face the need to take care of these functions in real time; several tasks that have been taken care of in the past via overnight batch runs in the back office have now been moved to the front office and are executed in real time.

Again, one can observe an interplay between advances in technology and financial/ business practice. On the one hand, there is the need to constantly improve analytics approaches in terms of speed and capability by applying modern technologies. On the other hand, advances on the

technology side allow new analytics approaches that were considered impossible (or infeasible due to budget constraints) a couple of years or even months ago.

One major trend in the analytics space has been the utilization of parallel architectures on the CPU (central processing unit) side and massively parallel architectures on the GPGPU (general-purpose graphical processing units) side. Current GPGPUs often have more than 1,000 computing cores, making necessary a sometimes radical rethinking of what parallelism might mean to different algorithms. What is still an obstacle in this regard is that users generally have to learn new paradigms and techniques to harness the power of such hardware.

Chapter Four

3 SOFTWARE ENGINEERING

3.1 OVERVIEW

Development and implementation of the stock information system represents a responsible and difficult task, the realization of which require substantial financial resources, as well as human and technical resources. Limited material, staff and technical resources were the main factors that influenced many countries that were undergoing a period of transition to take over the stock exchange information systems of countries with developed market economies. However, there are examples of countries that have chosen an independent development path of their own stock information system.

The construction of this financial trading system demands an interdisciplinary approach for the projection of the whole system, which must be compatible with the surroundings in which it operates. The realization of these types of systems is usually done in phases, where you first have to deal with the functions of the stock that represent a priority and then with the remainder of the functions, which are also very important for the development of the capital market and for financial analysis to be used in prediction.

3.2 SYSTEM REQUIREMENT & ANALYSIS

The first step in the process of creating a stock predicting information system is defining the general user requirements that will be set to the information systems. Beside the standard technical and technological aspects that are given attention during the development of the stock information system, it is essential that other important problems that IT managers encounter during the realization of the system be solved too. This refers to the project orientated organization for the development of an extremely complex system that has the following elements:

- *Strategic*: operational and tactical planning for the further development of the stock trading system.
- *Defining the rules* by which the stock trading system will operate in accordance with the law regulations.

- *Controlling and managing our own time*, activities of project and relations between project teams.

During the research phase, with the aim to find the best ways to create the stock exchange information system, information is gathered and sorted regarding relevant solutions, analysis of already existing methods of development were made and experiment checks of suggested methods done by creating practical checks of the set hypotheses done by testing the trial versions and by testing under production circumstances.

The first step in development of any stock trading system is to a careful analysis of the conditions under which it will function, i.e. making the analysis of the capital market that depends on a series of different republic and state laws. This analysis should clearly show the links between the stock trading system and all other institutions that affect it in any way, and its information system. We had chosen The New York Stock Exchange (NYSE) for many reasons:

- NYSE was the first stock market to be established in the United States, tracing its roots back to 1792. The NYSE is also the largest stock exchange in the world in terms of capital invested.
- The combined capitalization of all companies listed in the NYSE as of May 2009 is $10.4 trillion.
- Over 419 billion stocks are traded each year, at an average price of $24.87 per share.
- The financial of the United States is often judged by the performance of stocks in the New York Stock Exchange.

3.3 PROJECT MANAGEMENT METHODOLOGY

3.3.1 Overview

Project management is mainly defined as being able to apply skills, knowledge and techniques in such a way as to finish all the necessary work and jobs that the project itself demands, or, in other words, that are vital to the successful completion of the project. Therefore, the analysis was made of all previously existing project management methodologies. The conclusion drawn from this analysis was that the chosen methodology must always be clearly defined and applicable to most IT projects.

3.3.2 Version Control System (VCS)

Version control systems, also known as *source code management systems,* or *revision control systems,* are a mechanism for keeping multiple versions of your files, so that when you modify a file you can still access the previous revisions.

Purpose of the Version Control System

- Allow developers to work simultaneously.
- Do not overwrite each other's changes.
- Maintain history of every version of everything.
- Easier to re-create the testing and production environments

Git is a distributed revision control and source code management system with an emphasis on speed. There's many kinds of VCS. We use *Git* in our project for many reasons:

- It's free, fast and open source.
- The chances of losing data are very rare when there are multiple copies of it.
- Secure, as It uses a common cryptographic hash function called secure hash function (SHA1) so it's is impossible to change file, date, and commit message and any other data from the Git database without knowing Git.
- Branch management with Git is very simple. It takes only a few seconds to create, delete, and merge branches.

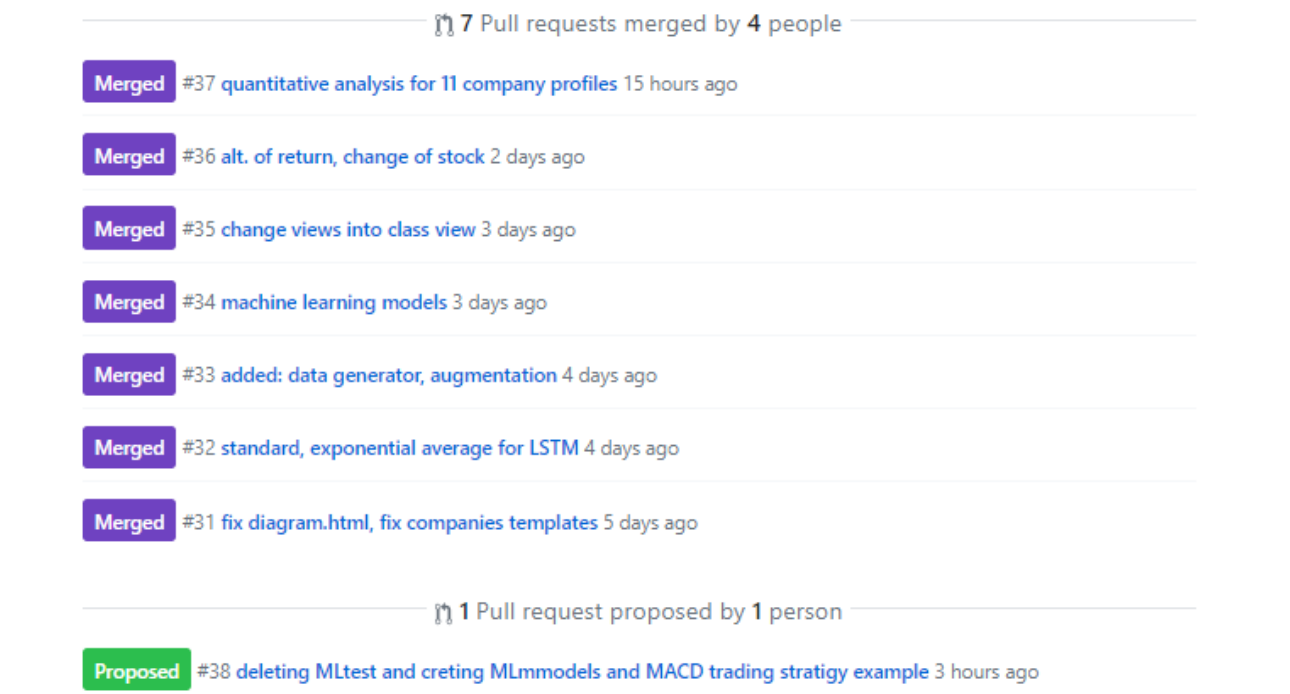


Figure: An example of an approved pull requests from contributors

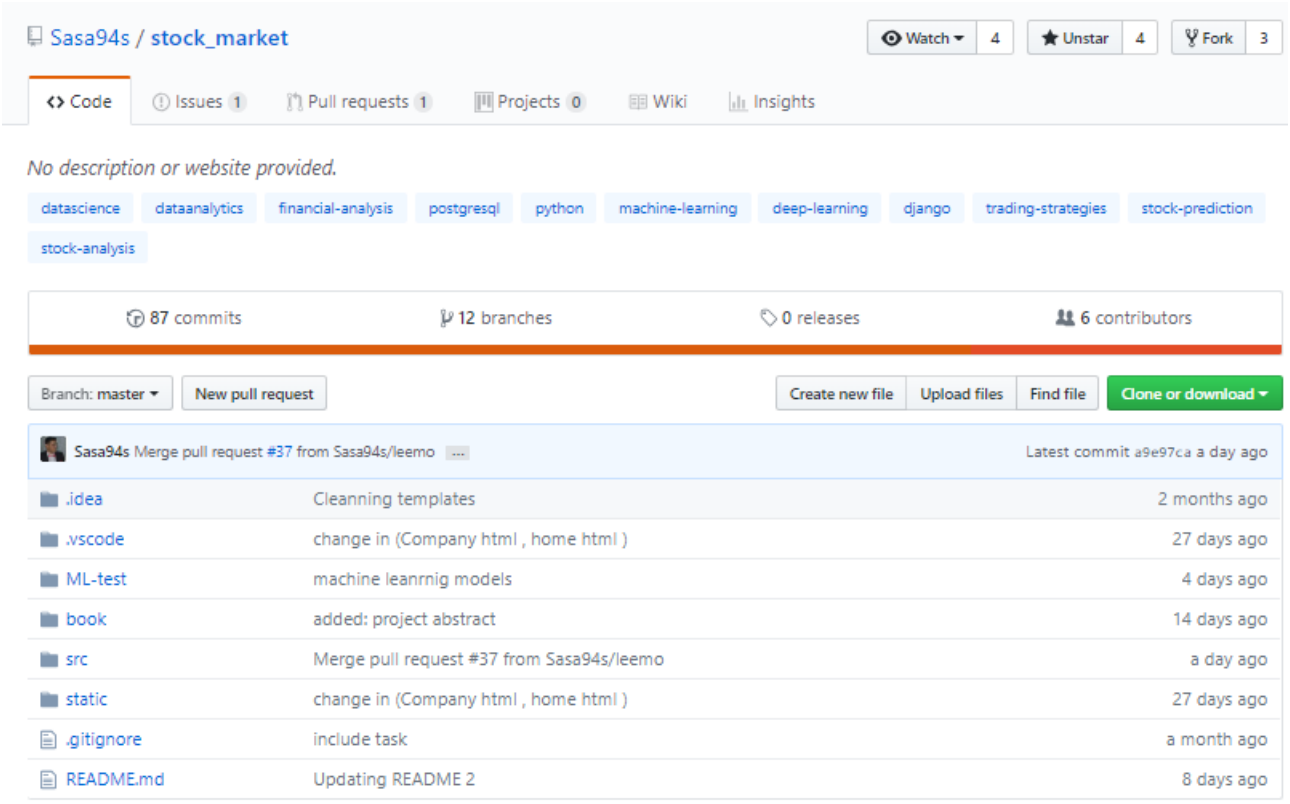


Figure : Project on github

Link Project on github: https://github.com/Sasa94s/stock_market

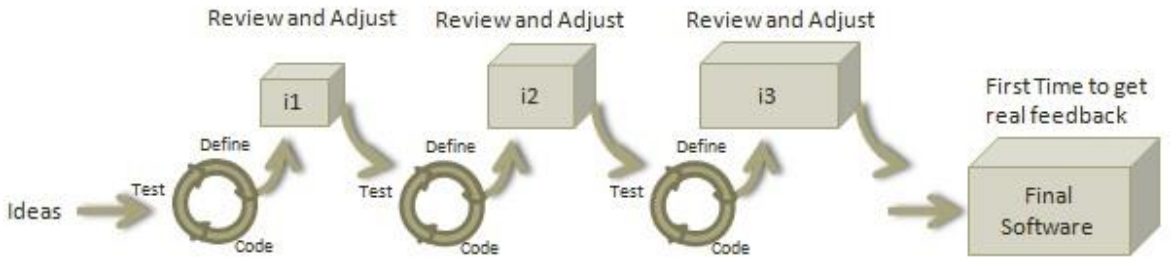
3.4 AGILE METHODOLOGY

A model is a collection of the different processes and shows the way in which the project will be managed, something similar to a schedule that shows when different activities will be carried out.

Agile is a software development methodology to build a software incrementally using short iterations of 1 to 4 weeks so that the development is aligned with the changing business needs.

3.4.1 Roles in Agile

Traditional Method



Agile Method

Scrum Master

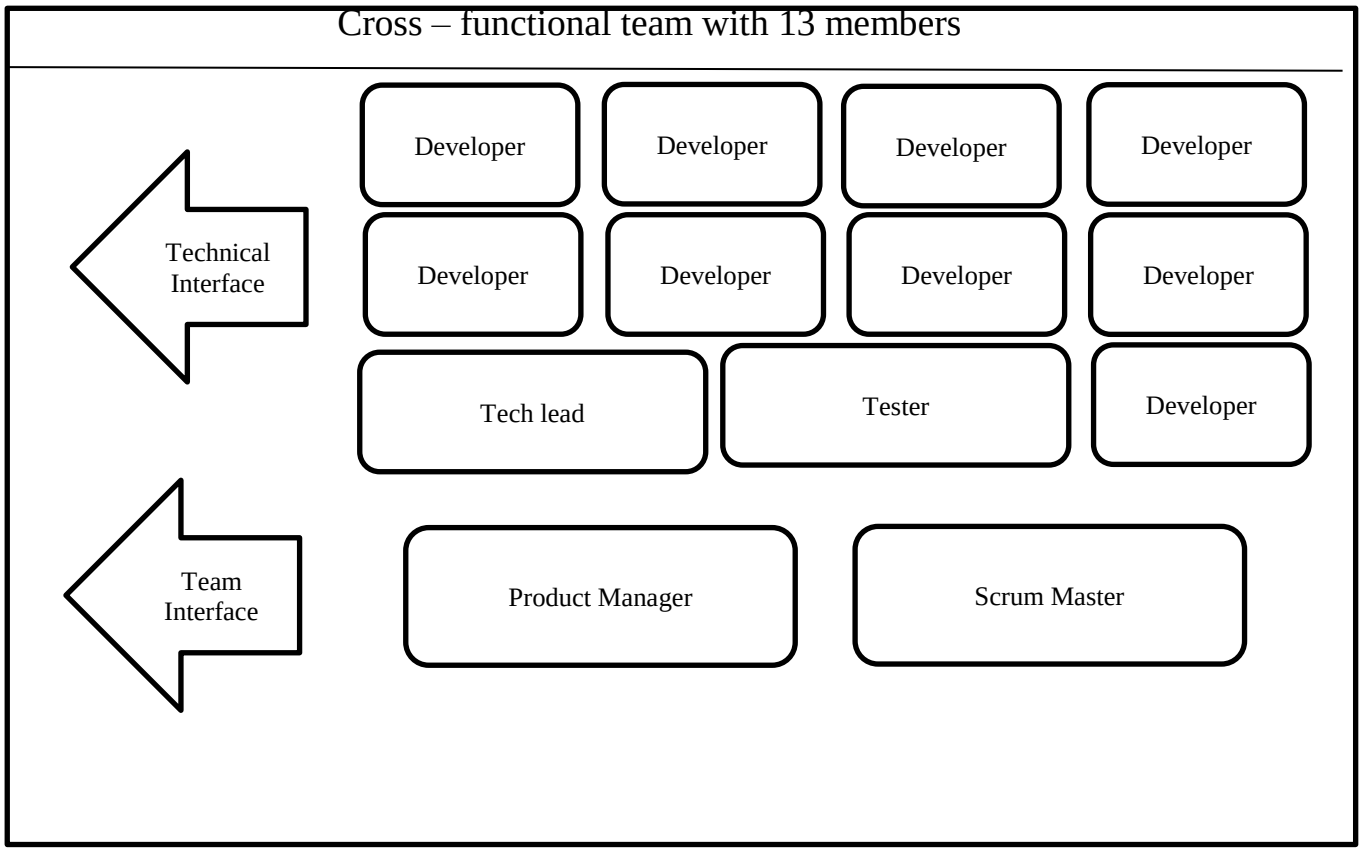
A Scrum Master is a team leader and facilitator who helps the team members to follow agile practices so that they can meet their commitments.

Product Manager

A Product manager is the one who drives the product from business perspective

Cross-functional Team

Our agile team comprises of 9 developers, 1 tester, 1 technical lead, 1 product manager and 1 scrum master.



Product Manager and Scrum Master are considered to be a part of Team Interface, whereas other members are part of Technical Interface.

3.5 UML DIAGRAMS

UML (Unified Modeling Language) is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997. It was initially started to capture the behavior of complex software and non-software system and now it has become an OMG standard.

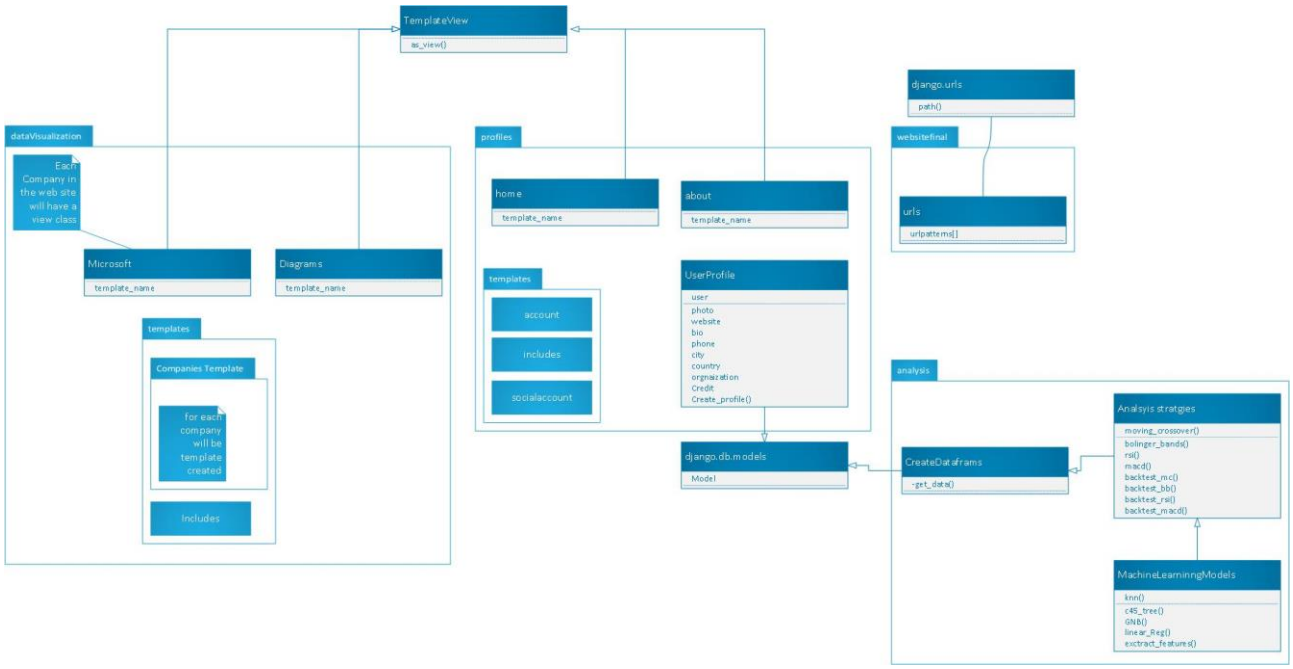
There are three important types of UML modeling.

3.5.1 Structural Modeling

Structural modeling captures the static features of a system. And we will cover Class Diagram as an example of structural modeling

Class Diagram

Class diagrams are the most popular UML diagrams used for construction of software applications. The purpose of class diagram is to model the static view of an application.

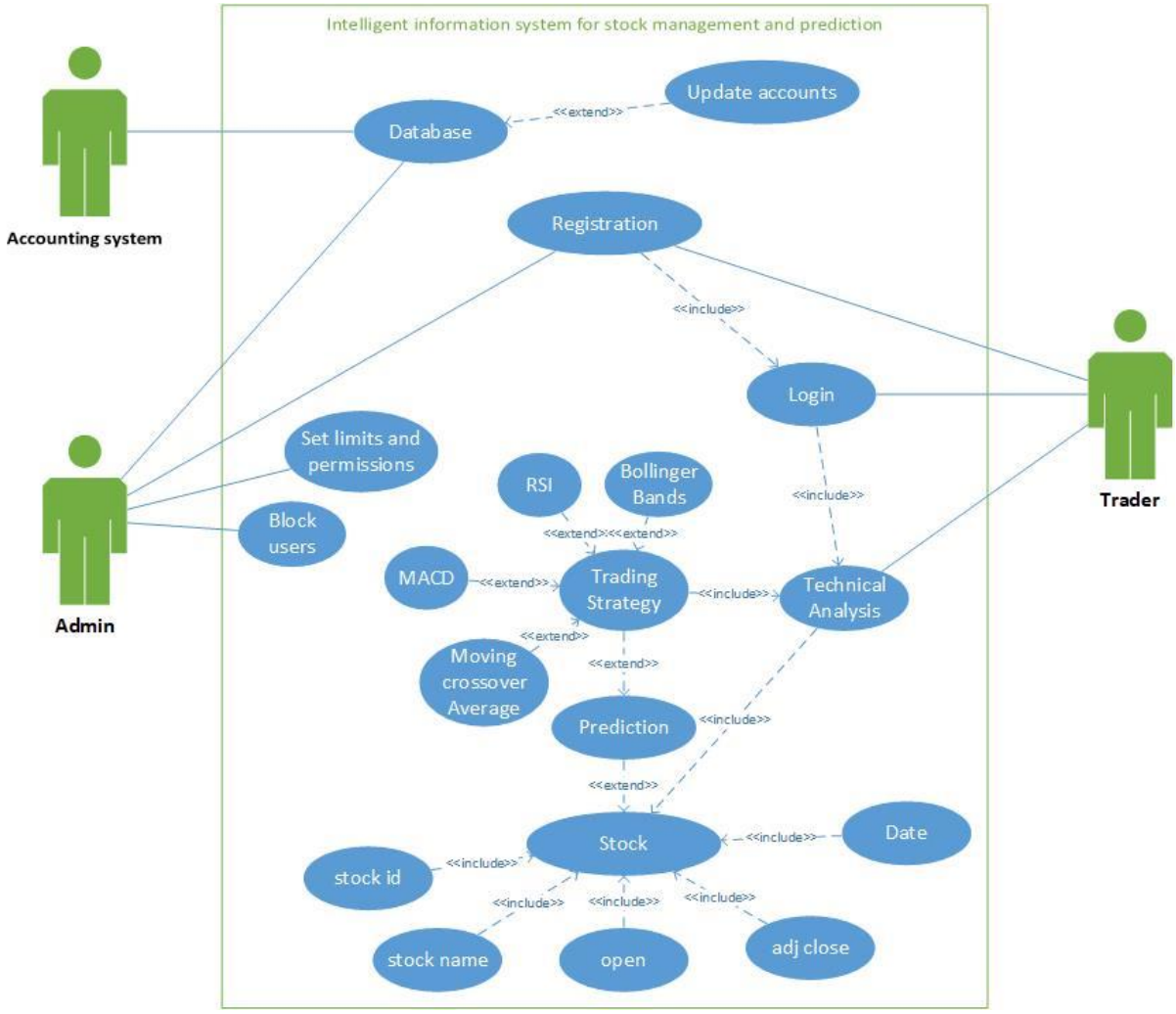


3.5.2 Behavior Modeling

Behavioral model describes the interaction in the system. It represents the interaction among the structural diagrams. Behavioral modeling shows the dynamic nature of the system. As an examples:

Use Case Diagram

To model a system, the most important aspect is to capture the dynamic behavior. Dynamic behavior means the behavior of the system when it is running/operating.



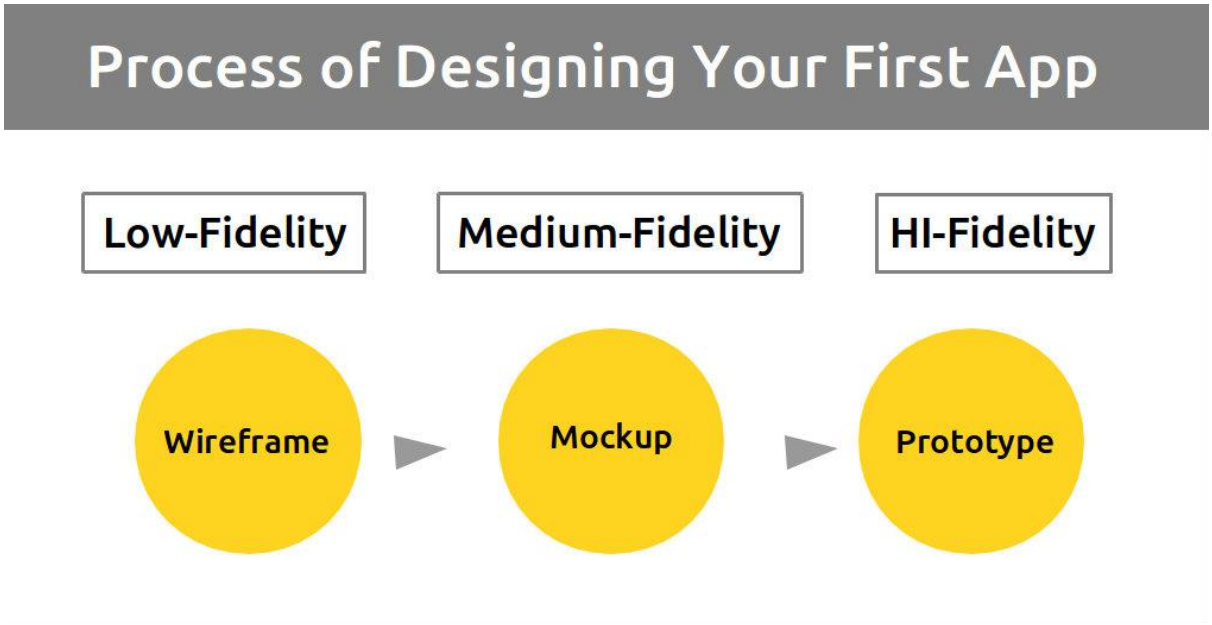
Chapter Three

4 WEB INTEGRATION

4.1 FRONT-END DEVELOPMENT

4.1.1 Mock up

What is the process of designing for App:

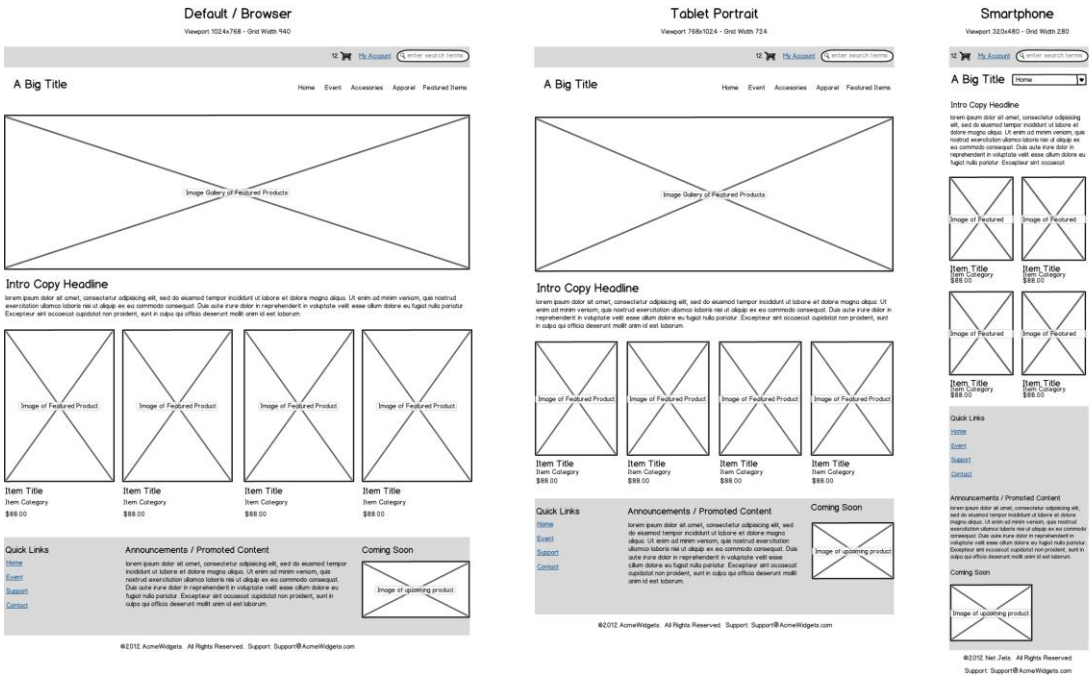


What is a Mock up for website?

Unlike a Wireframe, that is basically a simple sketch of a future site, the Mock up is almost a complete design. It displays not only the order of all the components but also the accurate colours, shapes and exact placement of the components, in relation to each other.

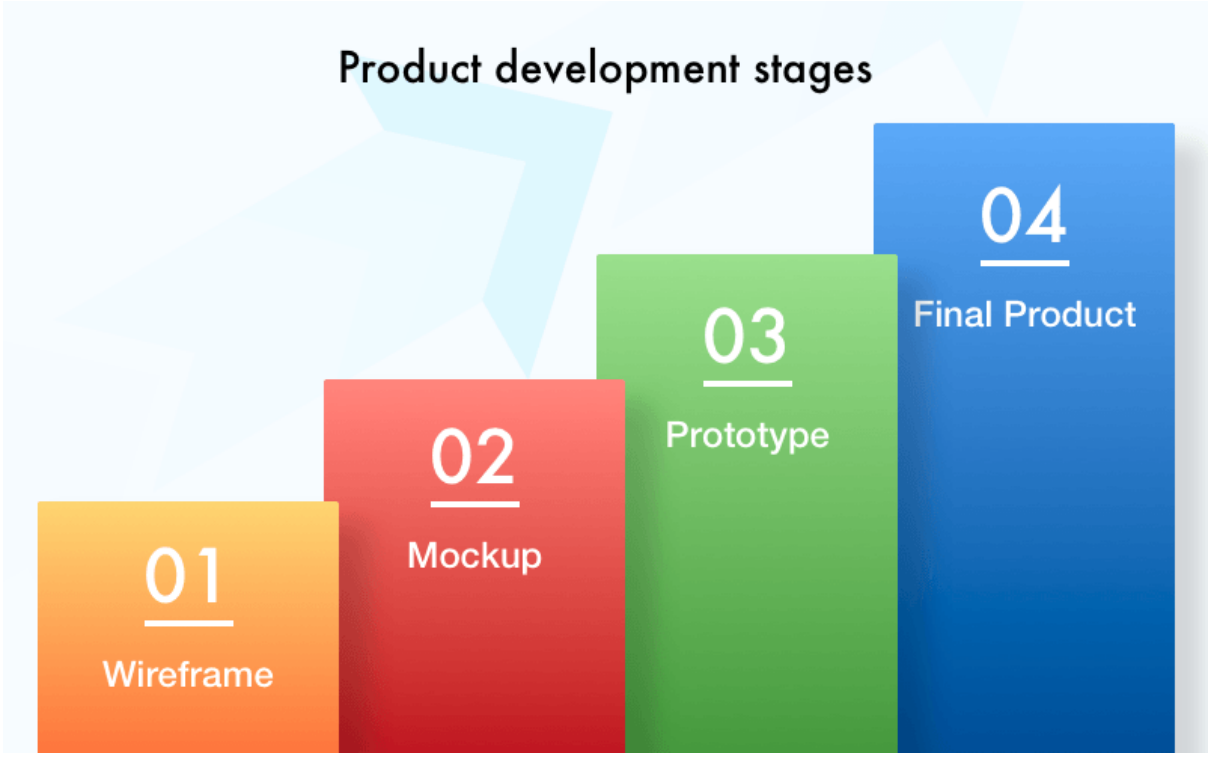
In order not to misunderstand the terms Mock up, Wireframe and a prototype, imagine that they are the steps to a final product.

What is the wireframe



The first step is a Wireframe, it reflects only the elements that you need to have on your site and their places on a page. The Wireframe can be easily made by hand.

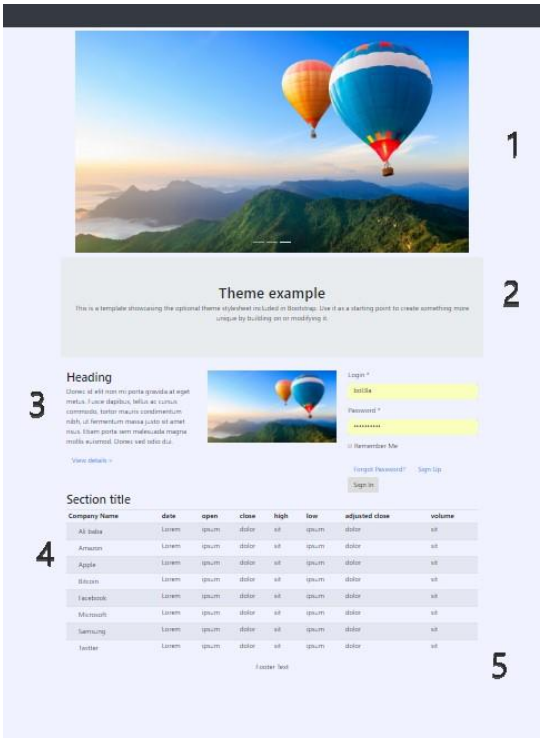
Product development stages:



The second step is a web page Mock up:

Here you have to choose the style, colour pallet, shape and design of all the components. Working on a Mock up, you can check how ergonomic the design is, as well as the overall view of a page.

Here is for example (form our website) Stock-market lay-out:



It contains more than one part for photo(Carousel) , contain of page , form , table , news and footer .

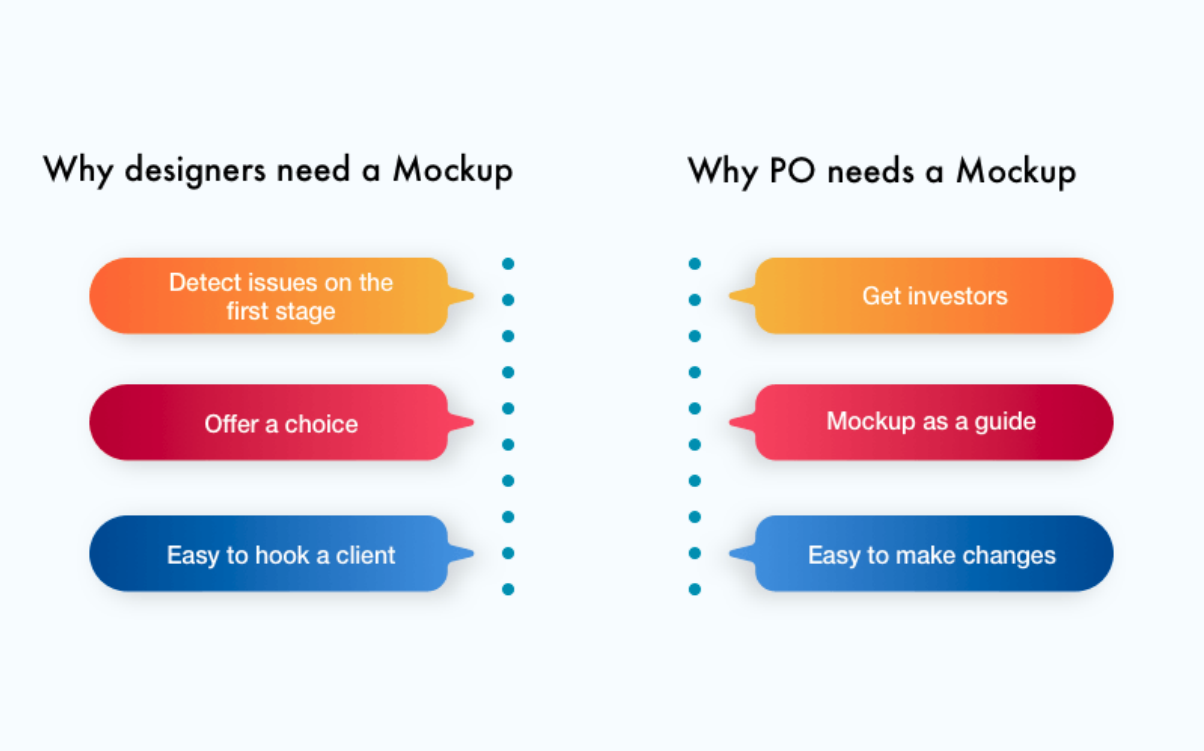
A prototype is a clickable working product, which you can test. In plain English, it’s a ready-made product but with a minimal functionality set.

Prototype is needed as proof of the concept, especially if you work with captious investors. If you want to persuade them that your idea is brilliant, show them a prototype.

Mock up for website, why do you need it?

Of course, when you develop a product you will have a wireframe. The prototype is optional; however, a Mock up web design is a must. Why?

Let’s ponder how to make a website Mock up and gain advantages that the Mock up creation brings to designers and a product owner.



How to make Mock up for website?

It’s great if a designer has lots of ideas. Although to create a website Mock up, a really fantastic one, we have to think not only about aesthetic enjoyment but also about its functionality. Users won’t appreciate pixel perfect design if it’s uncomfortable to use. The Mock up stands not only for beauty but also for the user’s comfort.

Do you remember how McDonald’s was designed?

Brothers Richard and Maurice McDonald drew a layout of their kitchen on the ground, schematically marked the order of all items and made their employees move like they were in a real kitchen. They detected the best trajectory for each worker according to the performed tasks. It was the first time when the design helped shorten food preparation time from 10 minutes to 30 seconds.

The story teaches us how important the design is for productivity. The buttons order as well as their sizes should be not only nice to watch at but comfortable to use.

4.1.2 What is Responsive Web Design?

Responsive web design (RWD) is an approach to web design which makes web pages render well on a variety of devices and window or screen sizes. Recent work also considers the viewer proximity as part of the viewing context as an extension for RWD Content, design and performance are necessary across all devices to ensure usability and satisfaction.

A site designed with RWD adapts the layout to the viewing environment by using fluid, proportion-based grids, flexible images, and CSS3 media queries, an extension of the `@media` rule.

.Responsive web design makes your web page look good on all devices.

.Responsive web design uses only HTML and CSS.

.Responsive web design is not a program or a JavaScript.

What is css grid layout?

CSS Grid Layout is the most powerful layout system available in CSS. It is an 2-dimensional system, meaning it can handle both columns and rows, unlike flexbox which is largely a 1-dimensional system. You work with Grid Layout by applying CSS rules both to a parent element (which becomes the Grid Container) and to that elements children (which become Grid Items).

why a grid based approach will improve your designs?

01. Grids Keep Your Content Organized

One of the chief uses for a grid is to keep your elements aligned and ordered, and your page design clean and neat.

02. It Will Make Your Job Quicker

Grids can greatly speed up and improve your design time, as they can act as a guide that signals where is best to place, position, and scale elements. Instead of randomly positioning elements until you find a decent looking composition.

03. Base Line Alignment

grids and type are two design elements that very happily go hand in hand. A strong grid can help make type-heavy layouts appear neater, more organized, and can help to enhance the legibility and readability of body copy.

04. Collaborating With Other Designers Will Be Much Easier

grids help in many ways to give an underlying structure to your design and provide a guide as to where certain elements should be placed. Therefore, if you have a solid grid set up in your design

and a new designer jumps in, they will (hopefully) be able to see how you've set it up and how you intend to position elements.

05. Balancing Your Design Will Get Significantly Easier

The beauty of grids is that they are consistent and even things, so when you divide your design up into a certain amount of columns, you will begin with a symmetrical layout. From here, it should be much easier to judge which side of your design is overpowering the other.

What is bootstrap?

Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web.

1. Easy to Use

It is extremely an easy and speedy procedure to begin with Bootstrap. Bootstrap is very adaptable too. You can utilize Bootstrap along with CSS.

2. Responsiveness

Every year mobile devices persist to grow hugely popular, and the requirement to have a responsive website has become compulsory and important too. As the fluid grid layout amends vigorously to the appropriate screen resolution, thus crafting a mobile-ready site is a smooth and easy task along with Bootstrap. With the use of ready-made classes of Bootstrap, you can recognize the number of spots in the grid system that you would like each column to engage in. Then only you can identify at whichever point you would like your columns to load in horizontal position, instead of vertically to exhibit accurately on mobile appliances.

3. The Speed of the Development

One of the main benefits of utilizing Bootstrap happens to be the speed of the development. While driving out a new, fresh website or application swiftly, you should certainly reflect upon utilizing Bootstrap. Instead of coding from scrape, Bootstrap lets you to use ready-made coding blocks in order to assist you in setting up. You can blend that along with CSS-Less functionality and cross-browser compatibility that can give way to saving of ample hours of coding. You can even buy ready-made Bootstrap themes and alter them to fit your requirements, for gaining the quickest potential route.

4. Customizable Bootstrap

The Bootstrap can be customized as per the designs of your project. The web developers can make a choice to select the aspects which are required which can be simply complete by utilizing Bootstrap customize page. You just have to tick off all the aspects that you do not require, such as- Common CSS: typography, code, grid system, tables, buttons, forms, print media styles; Components: input groups, button groups, pager, labels, navs, navbar, badges, pagination; JavaScript components: dropdowns, popovers, modals, tooltips, carousels; Utilities: Responsive utilities, basic utilities. Thus your custom version of Bootstrap is all set for download process.

5. Consistency

Few Twitter employees firstly expanded Bootstrap as a framework for boosting the consistency across interior tools. But later the Co-founder Mark Otto after understanding the actual potential released in August 2011 the first open-source version of Bootstrap. He even portrayed how the Bootstrap was enlarged with the use of one core concept- pairing of designers along with developers. Thus Bootstrap became popular on Twitter.

6. Support

As Bootstrap holds a big support community, you can be provided with help whenever there comes any problem. The creators always keep the Bootstrap updated. Presently Bootstrap is hosted, expanded, and preserved on the GitHub along with more than 9,000 commits, as well as more than 500 contributors.

7. Packaged JavaScript Components

Bootstrap approaches with a pack of JavaScript components for including the functionality that crafts it in simple way for operating things, such as tooltips, modal windows, alerts, etc. You can even leave out the writing scripts completely.

8. Simple Integration

Bootstrap can be simply integrated along with distinct other platforms and frameworks, on existing sites and new ones too. You can also utilize particular elements of Bootstrap along with your current CSS.

9. Grid

Bootstrap has the capability to utilize a 12-column grid that is responsive. It also upholds offset and nested elements. The grid can be maintained in a responsive mode, or you can simply modify it to a secured layout.

10. Pre-styled Components

Bootstrap approaches with pre-styled components for alerts, dropdowns, nav bars, etc. Hence, being a feature-rich, Bootstrap provides numerous advantages of using it. Hope you would have understood the above reasons so that you can easily use Bootstrap for making superb web designs for your sites!

4.2 BACK-END

4.2.1 Django

Why we use Django Framework: -

Django was developed by an online news operation with the aim to quickly create reliable web applications using the Python programming language. The framework has templates, libraries and API designed to work together for natural growth and connectivity. In practice this means that apps built in Django may be grown with minimal costs of upgrades, changes, and additions. In general, Django makes lots of things easier.

Django suits projects of any size, from small to the biggest ones. Due to its scalability, it's perfect for startups. But it also serves great for web sites with hundreds of visitors a second, such as Washington Post.

Django uses Python which was one of the most popular programming languages of 2015 and is now the most popular language for those learning to code. And the framework itself is increasingly widely used, and because it's open source, it is built and kept updated by the developers who use it. This means that you can easily find answers to your questions arising from your project created in Django/Python.

A programming framework is a toolkit of components needed to create a web or mobile application. Django is a more fully featured kit than most of other frameworks, it contains everything you need to build an app.

If something is missing in the kit, Django may be extended with numerous ‘app’ plugins. The packages let you add google maps, create complex permissions, or connect to stripe for payment processing.

Django adheres to D.R.Y. — Don’t Repeat Yourself — philosophy. That means that the framework places a premium on getting the absolute most out of very little code. As a result, it supposes less hours to get it working, less code to break, and less to change when you need re-orientation.

Django is very well documented. This allows avoiding of hours of trial and errors or effortless implementation. Every specific release of Django is supported by all necessary documentation and code examples. On top of that, the code is all publicly available on GitHub for direct investigation.

How it works

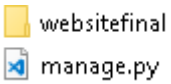
Installing Django latest version with ***pip***:

```
PS C:\Users\moham> pip install Django
Requirement already satisfied: Django in c:\users\moham\anaconda3\lib\site-packages (2.0.4)
Requirement already satisfied: pytz in c:\users\moham\anaconda3\lib\site-packages (from Django) (2018.3)
```

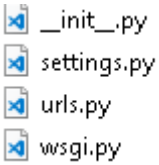
Creating Django project run the following command:

```
PS C:\Users\moham> django-admin startproject websitefinal
PS C:\Users\moham>
```

What **startproject** created:



Inside **websitefinal** directory:



Run Django local server:

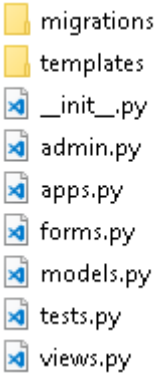
```
PS C:\Users\moham\desktop\stock-market\src> python manage.py runserver
Performing system checks...

System check identified no issues (0 silenced).
May 27, 2018 - 17:45:49
Django version 2.0.4, using settings 'websitefinal.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Create profile app:

```
PS C:\Users\moham\websitefinal> python manage.py startapp profiles
PS C:\Users\moham\websitefinal>
```

profileapp hierarchy:



Templates

contains all the HTML templates that will be rendered in *view.py*, and all the templates are provided by the Front-end developers' team.

models.py: Used to create table in the project database

```python
# Create your models here.
class UserProfile(models.Model):
    user = models.OneToOneField(User, related_name='user', on_delete=models.PROTECT)
    photo = models.ImageField(upload_to='profile_image', blank=True)
    website = models.URLField(default='', blank=True)
    bio = models.TextField(default='', blank=True)
    phone = models.CharField(max_length=20, blank=True, default='')
    city = models.CharField(max_length=100, default='', blank=True)
    country = models.CharField(max_length=100, default='', blank=True)
    organization = models.CharField(max_length=100, default='', blank=True)
    Credit = MoneyField(
        max_digits=10,
        decimal_places=2,
        validators=[
            MinMoneyValidator(0),
            MaxMoneyValidator(1500),
            MinMoneyValidator(Money(0, 'NOK')),
            MaxMoneyValidator(Money(900, 'NOK')),
            MinMoneyValidator({'EUR': 100, 'USD': 0}),
            MaxMoneyValidator({'EUR': 1000, 'USD': 500}),
        ])
```

views.py:

- Contain view classes that used to render a certain html template according to the selected URL

```python
class home(TemplateView):
    template_name = 'home.html'
```

- Contain class that will used to install the profile application in setting.py

```python
class ProfilesConfig(AppConfig):
    name = 'profiles'
```

admin.py: Used to register the table created by model.py into database

```python
class profileAdmin(admin.ModelAdmin):
    class Meta:
        model = UserProfile
admin.site.register(UserProfile, profileAdmin)
```

Migrate model into database:

```
PS C:\Users\moham\desktop\stock-market\src> python manage.py makemigrations
Migrations for 'profiles':
  profiles\migrations\0008_auto_20180528_0041.py
    - Add field Credit to userprofile
    - Add field Credit_currency to userprofile
    - Add field photo to userprofile
    - Alter field bio on userprofile
PS C:\Users\moham\desktop\stock-market\src>
```

```
PS C:\Users\moham\desktop\stock-market\src> python manage.py migrate
Operations to perform:
  Apply all migrations: account, admin, auth, contenttypes, sessions, sites, socialaccount
Running migrations:
  No migrations to apply.
PS C:\Users\moham\desktop\stock-market\src>
```

Now the profile application is ready to be install, it will be installed using setting.py

```
# Application definition

INSTALLED_APPS = [
    'profiles',
```

Every application in the website will be created with the same way.

Database

Why used PostgreSQL?

The used Database engine is ***PostgrSQL (postgres) because*** it comes with many features aimed to help developers build applications, administrators to protect data integrity and build fault-tolerant environments, and help you manage your data no matter how big or small the dataset. In addition to being free and open source, PostgreSQL is highly extensible. For example, you can define your own data types, build out custom functions, even write code from different programming languages without recompiling your database!

Data Types

Postgres provide many data types, separated into two main types:

- ***Primitives*:** Integer, Numeric, String, Boolean
- ***Structured*:** Date/Time, Array, Range, UUID

Extensibility

- **Stored procedures**
- **Procedural Languages:** PL/PGSQL, Perl, Python (and many more)

Internationalization**,** Text Search

Support for international character sets, e.g. through ICU collations

Performance

PostgreSQL is widely used in large systems where read and write speeds are crucial and data needs to validate. In addition, it supports a variety of performance optimizations.

Overall, PostgreSQL performance is utilized best in systems requiring execution of complex queries.

PostgreSQL performs well in OLTP/OLAP systems when read/write speeds are required, and extensive data analysis is needed.

PostgreSQL also works well with Business Intelligence applications but is better suited for Data Warehousing and data analysis applications that require fast read/write speeds.

Database Design

ERD

4.2.2 Django Libraries

<u>django-allauth</u>

Integrated set of Django applications addressing authentication, registration, account management as well as 3rd party (social) account authentication.

Requirements:

- Python 2.7, 3.3, 3.4, or 3.5
- Django (1.10+)
- `python-openid` or `python3-openid` (depending on your Python version)
- `requests` and `requests-oauthlib`

Supported Flows:

- Signup of both local and social accounts

- Connecting more than one social account to a local account
- Disconnecting a social account – requires setting a password if only the local account remains
- Optional instant-signup for social accounts – no questions asked
- E-mail address management (multiple e-mail addresses, setting a primary)
- Password forgotten flow
- E-mail address verification flow

Features:

- Supports multiple authentication schemes (e.g. login by user name, or by e-mail), as well as multiple strategies for account verification (ranging from none to e-mail verification).
- All access tokens are consistently stored so that you can publish wall updates etc.

Architecture & Design

Pluggable signup form for asking additional questions during signup.

Support for connecting multiple social accounts to a Django user account.

The required consumer keys and secrets for interacting with Facebook, Twitter and the likes are to be configured in the database via the Django admin using the SocialApp model.

Consumer keys, tokens make use of the Django sites framework. This is especially helpful for larger multi-domain projects, but also allows for easy switching between a development (localhost) and production setup without messing with your settings and database.

Installation:

```
pip install django-allauth
```

settings.py:

```
'allauth',
'allauth.account',
'allauth.socialaccount',
'allauth.socialaccount.providers.google',
```

urls.py:

```
path('accounts/', include('allauth.urls')),
```

Post-Installation:

```
./manage.py migrate
```

```
(.virtualenvs) E:\graduation project\stock_market\src>python manage.py migrate
Operations to perform:
  Apply all migrations: account, admin, auth, contenttypes, profiles, sessions, sites, socialaccount
```

crispy-forms

django-crispy-forms is a Django application that lets you easily build, customize and reuse forms using your favorite CSS framework, without writing template code and without having to take care of annoying details. You are currently looking at the documentation of the development release.

Requirements: django-crispy-forms supports Python 2.7/Python 3.3+ and Django 1.8/Django 1.10+

Template packs:

- Since version 1.1.0, django-crispy-forms has built-in support for different CSS frameworks, known as template packs within django-crispy-forms:
- bootstrap Bootstrap is crispy-forms's default template pack, version 2 of the popular simple and flexible HTML, CSS, and Javascript for user interfaces from Twitter.
- bootstrap3 Twitter Bootstrap version 3.
- bootstrap4 Alpha support for Twitter Bootstrap version 4, which is still in Alpha.
- uni-form Uni-form is a nice looking, well structured, highly customizable, accessible and usable forms.

Installation:

```
pip install --upgrade django-crispy-forms ( stable version )
```

settings.py:

```
'crispy_forms',
```

Django-money

A little Django app that uses (py-moneyed) to add support for Money fields in your models and forms.

Requirements:

- Django versions supported: 1.8, 1.11, 2.0
- Python versions supported: 2.7, 3.4, 3.5, 3.6
- PyPy versions supported: PyPy 2.6, PyPy3 2.4

Supported Flows:

- Support for proper Money value handling (using the standard Money design pattern).
- A currency class and definitions for all currencies in circulation.
- Formatting of most currencies with correct currency sign.

Installation:

```
pip install django-money
```

settings.py:

```
'djmoney',
```

```
55        'djmoney',
```

Model usage (models.py):

```
Credit = MoneyField(
    max_digits=10,
    decimal_places=2,
    validators=[
        MinMoneyValidator(0),
        MaxMoneyValidator(1500),
        MinMoneyValidator(Money(0, 'NOK')),
        MaxMoneyValidator(Money(900, 'NOK')),
        MinMoneyValidator({'EUR': 100, 'USD': 0}),
        MaxMoneyValidator({'EUR': 1000, 'USD': 500}),
    ])
```

psycopg2

Psycopg2 is the most popular PostgreSQL database adapter for the Python programming language. Its main features are the complete implementation of the Python DB API 2.0 specification and the thread safety (several threads can share the same connection). It was designed for heavily multi-threaded applications that create and destroy lots of cursors and make a large number of concurrent “INSERT”s or “UPDATE”s.

Psycopg2 is mostly implemented in C as a libpq wrapper, resulting in being both efficient and secure. It features client-side and server-side cursors, asynchronous communication and

notifications, “COPY TO/COPY FROM” support. Many Python types are supported out-of-the-box and adapted to matching PostgreSQL data types; adaptation can be extended and customized thanks to a flexible objects adaptation system.
Psycopg2 is both Unicode and Python 3 friendly.

Installation:

```
pip install psycopg2
```

settings.py:

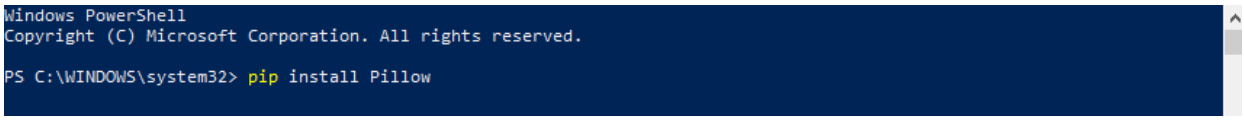
```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql_psycopg2',
        'NAME': 'stockmarket',
        'USER': 'postgres',
        'PASSWORD': '',
        'HOST': 'localhost',
        'PORT': '5432',
    }
}
```

Pillow

Pillow is the “friendly” PIL fork by Alex Clark and Contributors. PIL is the Python Imaging Library by Fredrik Lundh and Contributors.

Installation:

```
Cmd: pip install Pillow
```



models.py:

```
photo = models.ImageField(upload_to='profile_image', blank=True)
```

```
10  class UserProfile(models.Model):
11      user = models.OneToOneField(User, related_name='user', on_delete=models.PROTECT)
12      photo = models.ImageField(upload_to='profile_image', blank=True)
```

Chapter Five

5 QUANTITATIVE ANALYSIS

DISCLAIMER: THIS IS NOT FINANCIAL ADVICE! Furthermore, Our team has ZERO experience as a trader (a lot of this knowledge comes from only 3-months of research on stock trading)! This is purely introductory knowledge, not enough to make a living trading stocks. People can and do lose money trading stocks, and you do so at your own risk!

5.1 PYTHON USER SPECTRUM

Python does not only appeal to professional software developers; it is also of use for the casual developer as well as for domain experts and scientific developers.

Professional software developers find all that they need to efficiently build large applications. Almost all programming paradigms are supported; there are powerful development tools available; and any task can, in principle, be addressed with Python. These types of users typically build their own frameworks and classes, also work on the fundamental Python and scientific stack, and strive to make the most of the ecosystem.

Scientific developers or domain experts are generally heavy users of certain libraries and frameworks, have built their own applications that they enhance and optimize over time, and tailor the ecosystem to their specific needs. These groups of users also generally engage in longer interactive sessions, rapidly prototyping new code as well as exploring and visualizing their research and/or domain data sets.

The Scientific Stack

There is a certain set of libraries that is collectively labeled the scientific stack. This stack comprises, among others, the following libraries:

NumPy: provides a multidimensional array object to store homogenous or heterogeneous data; it also provides optimized functions/methods to operate on this array object.

SciPy: is a collection of sublibraries and functions implementing important standard functionality often needed in science or finance; for example, you will find functions for cubic splines interpolation as well as for numerical integration.

matplotlib: This is the most popular plotting and visualization library for Python, providing both 2D and 3D visualization capabilities.

Pandas: pandas builds on NumPy and provides richer classes for the management and analysis of time series and tabular data; it is tightly integrated with matplotlib for plotting.

Why we use Python for financial analysis?

Depending on the specific domain or problem, this stack is enlarged by additional libraries, which more often than not have in common that they build on top of one or more of these fundamental libraries.

Taking Python as a programming language alone, there are a number of other languages available that can probably keep up with its syntax and elegance. For example, Ruby is quite a popular language often compared to Python. On the language's website you find the following description:

A dynamic, open source programming language with a focus on simplicity and productivity. It has an elegant syntax that is natural to read and easy to write.

The majority of people using Python would probably also agree with the exact same statement being made about Python itself. However, what distinguishes Python for many users from equally appealing languages like Ruby is the availability of the scientific stack. This makes Python not only a good and elegant language to use, but also one that is capable of replacing domain-specific languages and tool sets like Matlab or R. In addition, it provides by default anything that you would expect, say, as a seasoned web developer or systems administrator.

5.2 PYTHON FOR FINANCE

5.2.1 Finance and Python Syntax

Most people who make their first steps with Python in a finance context may attack an algorithmic problem. This is similar to a scientist who, for example, wants to solve a differential equation, wants to evaluate an integral, or simply wants to visualize some data. In general, at this stage, there is only little thought spent on topics like a formal development process, testing, documentation, or deployment. However, this especially seems to be the stage when people fall in love with Python. A major reason for this might be that the Python syntax is generally quite close to the mathematical syntax used to describe scientific problems or financial algorithms.

Three aspects are worth highlighting:

- Syntax

  The Python syntax is indeed quite close to the mathematical syntax, e.g., when it comes to the parameter value assignments.

- Translation

  Every mathematical and/or algorithmic statement can generally be translated into a single line of Python code.

- Vectorization

  One of the strengths of NumPy is the compact, vectorized syntax, e.g., allowing for 100,000 calculations within a single line of code.

Efficiency and Productivity Through Python

At a high level, benefits from using Python can be measured in three dimensions:

- Efficiency

  How can Python help in getting results faster, in saving costs, and in saving time?

- Productivity

  How can Python help in getting more done with the same resources (people, assets, etc.)?

- Quality

  What does Python allow us to do that we could not do with alternative technologies? A discussion of these aspects can by nature not be exhaustive. However, it can highlight some arguments as a starting point.

5.2.2 Ensuring high performance

In general, it is accepted that Python has a rather concise syntax and that it is relatively efficient to code with. However, due to the very nature of Python being an interpreted language, the prejudice persists that Python generally is too slow for compute-intensive tasks in finance. Indeed, depending on the specific implementation approach, Python can be really slow. But it does not have to be slow—it can be highly performing in almost any application area. In principle, one can distinguish at least three different strategies for better performance:

- Paradigm

  In general, many different ways can lead to the same result in Python, but with rather different performance characteristics; “simply” choosing the right way (e.g., a specific library) can improve results significantly.

- Compiling

  Nowadays, there are several performance libraries available that provide compiled versions of important functions or that compile Python code statically or dynamically (at runtime or call time) to machine code, which can be orders of magnitude faster; popular ones are Cython and Numba.

- Parallelization

  Many computational tasks, in particular in finance, can strongly benefit from par- allel execution; this is nothing special to Python but something that can easily be accomplished with it.

5.2.3 Performance Computing with Python

Python per se is not a high-performance computing technology. However, Python has developed into an ideal platform to access cur- rent performance technologies. In that sense, Python has become something like a glue language for performance computing.

Later chapters illustrate all three techniques in detail. For the moment, we want to stick to a simple, but still realistic, example that touches upon all three techniques.

A quite common task in financial analytics is to evaluate complex mathematical ex- pressions on large arrays of numbers. To this end, Python itself provides everything needed:

```
In [1]: loops = 25000000
  from math import *
  a = range(1, loops)
  def f(x):
    return 3 * log(x) + cos(x) ** 2
  %timeit r = [f(x) for x in a]
Out[1]: 1 loops, best of 3: 15 s per loop
```

The Python interpreter needs 15 seconds in this case to evaluate the function f 25,000,000 times.

The same task can be implemented using NumPy, which provides optimized (i.e., pre-compiled), functions to handle such array-based operations:

```
In [2]: import numpy as np
  a = np.arange(1, loops)
  %timeit r = 3 * np.log(a) + np.cos(a) ** 2
Out[2]: 1 loops, best of 3: 1.69 s per loop
```

Using NumPy considerably reduces the execution time to 1.7 seconds.

However, there is even a library specifically dedicated to this kind of task. It is called numexpr, for “numerical expressions.” It compiles the expression to improve upon the performance of NumPy’s general functionality by, for example, avoiding in-memory copies of arrays along the way:

```
In [3]: import numexpr as ne
 ne.set_num_threads(1)
 f = '3 * log(a) + cos(a) ** 2'
 %timeit r = ne.evaluate(f)
Out[3]: 1 loops, best of 3: 1.18 s per loop
```

Using this more specialized approach further reduces execution time to 1.2 seconds. However, numexpr also has built-in capabilities to parallelize the execution of the re- spective operation. This allows us to use all available threads of a CPU:

```
In [4]: ne.set_num_threads(4)
  %timeit r = ne.evaluate(f)
Out[4]: 1 loops, best of 3: 523 ms per loop
```

This brings execution time further down to 0.5 seconds in this case, with two cores and four threads utilized. Overall, this is a performance improvement of 30 times. Note, in particular, that this kind of improvement is possible without altering the basic problem/ algorithm and without knowing anything about compiling and parallelization issues. The capabilities are accessible from a high level even by nonexperts. However, one has to be aware, of course, of which capabilities exist.

The example shows that Python provides a number of options to make more out of existing resources—i.e., to increase productivity. With the sequential approach, about 21 mn evaluations per second are accomplished, while the parallel approach allows for almost 48 mn evaluations per second—in this case simply by telling Python to use all available CPU threads instead of just one.

5.3 GETTING STARTED

Importing libraries

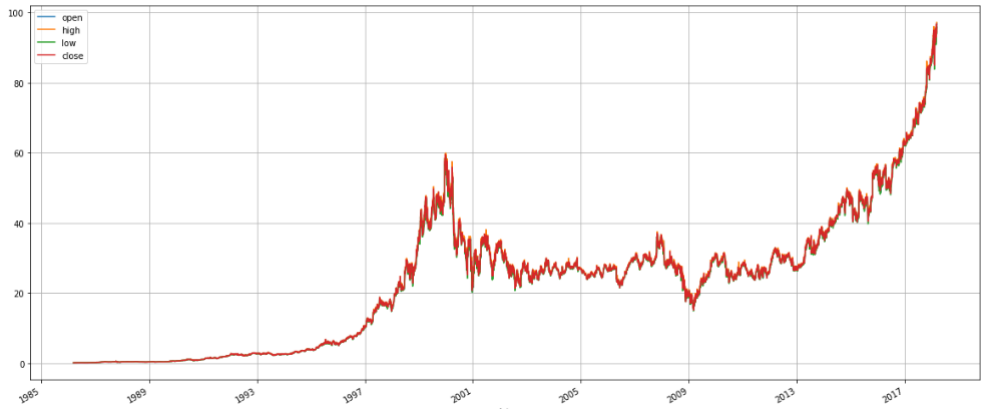
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from datetime import datetime
import quandl
from bokeh.plotting import Figure, output_file, show
from bokeh.io import show
from bokeh.models import HoverTool, ColumnDataSource
```

Getting and Visualizing Stock Data

Stock data can be obtained from Yahoo! Finance, Google Finance, or a number of other sources, and the pandas package provides easy access to Yahoo! Finance and Google Finance data, along with other sources.

```
start = "1970-01-01"
end = datetime.utcnow().strftime('%Y-%m-%d')

# Microsoft Corp.
msft = quandl.get("WIKI/MSFT", start_date=start, end_date=end)
microsoft = microsoft[["open","high","low","close"]].set_index(microsoft['date'])
microsoft.plot(grid=True)
plt.show()
```



A linechart is fine, but there are at least four variables involved for each date (open, high, low, and close), and we would like to have some visual way to see all four variables that does not require plotting four separate lines. Financial data is often plotted with a Japanese candlestick plot.

5.4 COMPANY PROFILES

5.4.1 Apple Inc.

Getting Financial Data

Requesting financial dataset using Quandl API:

```
aapl = quandl.get("WIKI/AAPL", start_date=start, end_date=end)
```

Descriptive Statistics

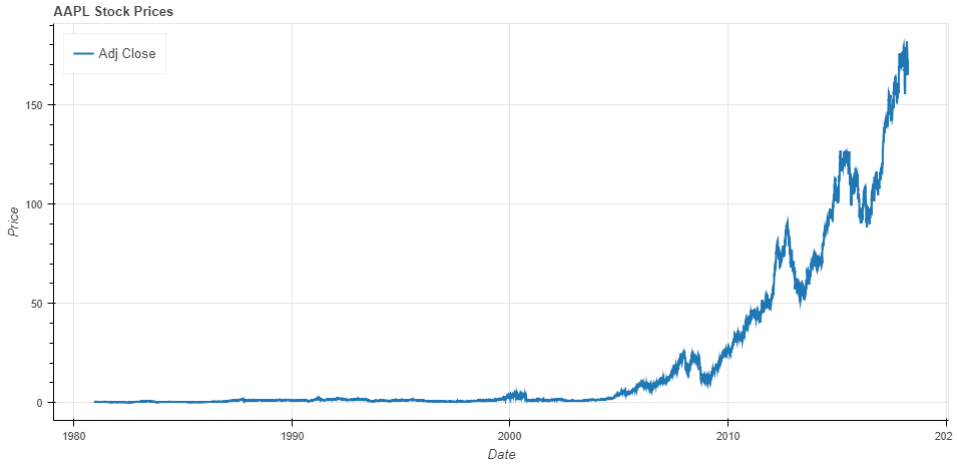
```
aapl.describe()
```

	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close
count	8076.000000	8076.000000	8076.000000	8076.000000	8.076000e+03	8076.000000	8076.000000	8076.000000	8076.000000	8076.000000	8076.000000
mean	59.832154	60.666713	59.025261	59.874821	3.125868e+07	0.001735	1.000991	18.763583	18.964604	18.564394	18.768430
std	32.526935	33.033695	32.017552	32.562926	3.204125e+07	0.038948	0.030460	17.756714	17.901221	17.607812	17.760955
min	15.200000	15.620000	14.870000	15.150000	8.000000e+03	0.000000	1.000000	0.058941	0.061253	0.058941	0.060097
25%	28.940000	29.250000	28.660000	28.940000	2.370594e+06	0.000000	1.000000	1.908662	1.934665	1.882658	1.908662
50%	54.250000	55.005000	53.500000	54.250000	2.617920e+07	0.000000	1.000000	18.581468	18.856998	18.366597	18.594782
75%	83.750000	84.955000	82.702500	83.870000	4.991018e+07	0.000000	1.000000	24.045268	24.273922	23.802299	24.060766
max	178.940000	180.380000	175.750000	179.940000	5.910522e+08	3.080000	2.000000	97.000000	97.240000	96.040000	96.770000

Data Preprocessing

Gaps in data can be filled by propagating the non-NaN values forward or backward along a Series.

```
def preprocessing(df):
    # Forward and Backward filling missing values
    df.fillna(method='ffill', inplace=True)
    df.fillna(method='bfill', inplace=True)
```



Return of Stocks

Return Equation:

$$return_t = \frac{p_t}{p_{t-1}} - 1$$

Log Return Equation:

$$change_t = log(price_t) - log(price_{t-1})$$

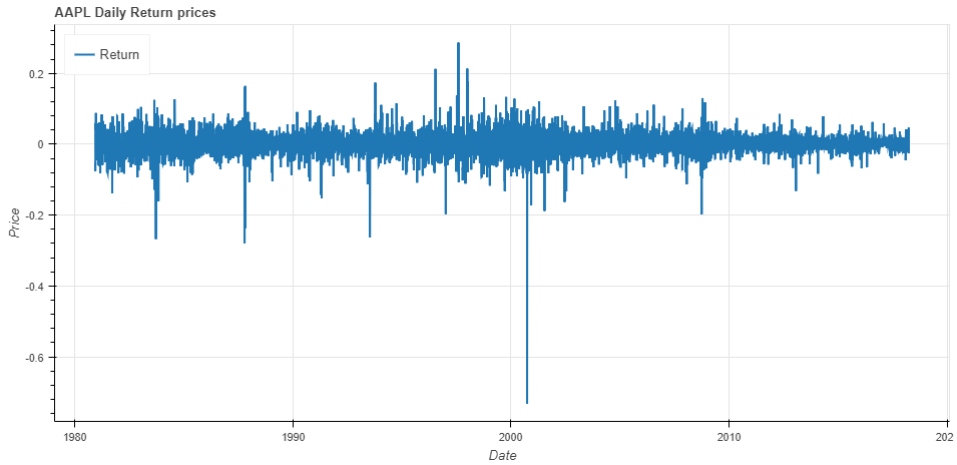
Growth Equation:

$$growth_t = \frac{(price_{t-1} - price_t)}{price_t}$$

These formulas are not the same and can lead to differing conclusions, but there is another way to model the growth of a stock: with log differences.

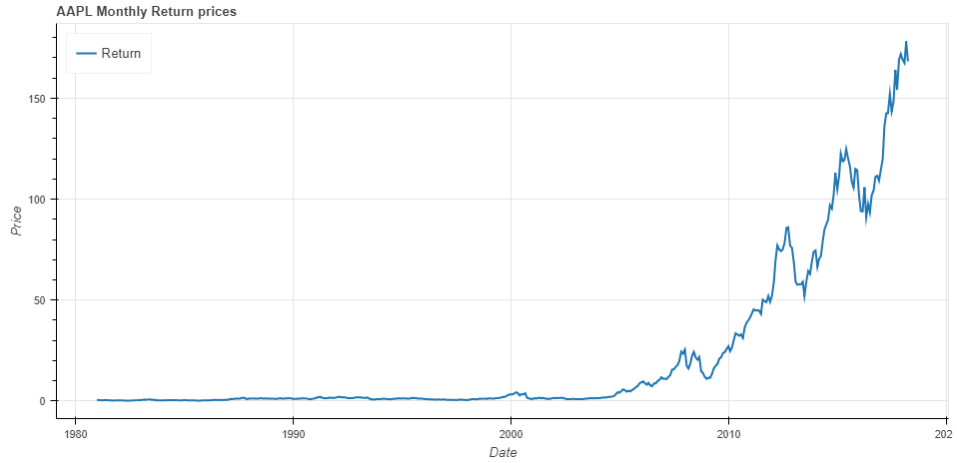
Daily Return

```
def daily_return(series):
    # Daily log returns
    daily_log_returns = np.log(series / series.shift(1))
    return daily_log_returns
```



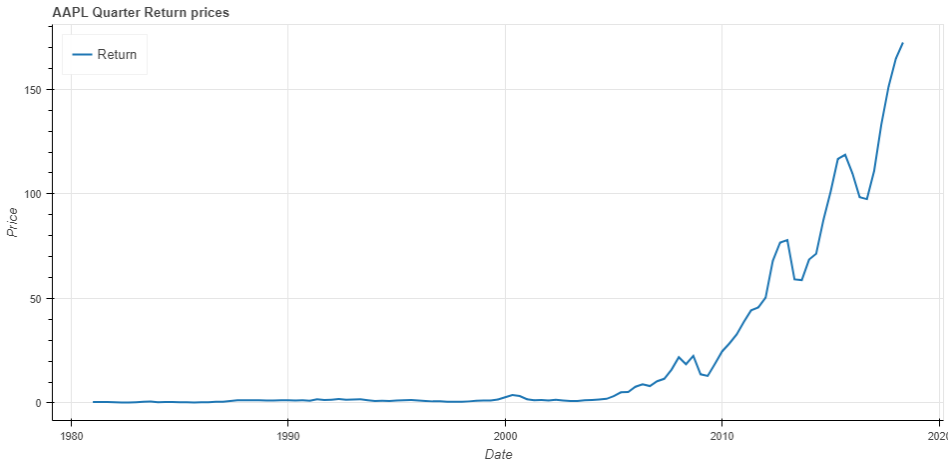
Monthly Return

```
def monthly_return(series):
    # Resample `stock` to business months, take last observation as value
    monthly = series.resample('BM').apply(lambda x: x[-1])
    # Calculate the monthly percentage change
    monthly.pct_change()
    return monthly
```



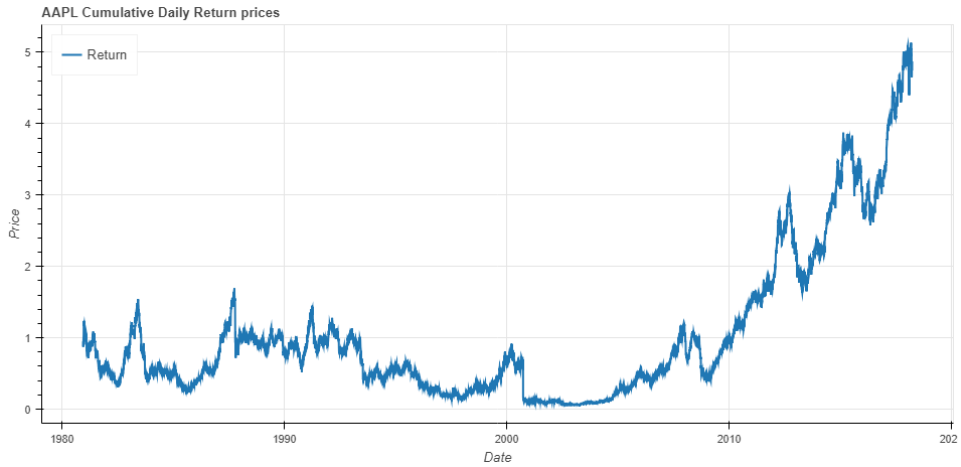
Quarter Return

```
def quarter_return(series):
    # Resample `stock` to quarters, take the mean as value per quarter
    quarter = series.resample("4M").mean()
    # Calculate the quarterly percentage change
    quarter.pct_change()
    return quarter
```



Cumulative Daily Return

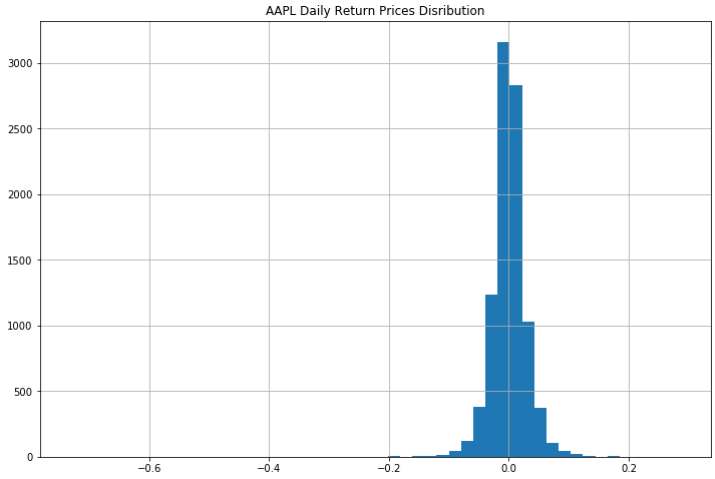
```
def cum_return(series):
    daily_pct_change = daily_return(series)
    cum_daily_return = (1 + daily_pct_change).cumprod()
    return cum_daily_return
```



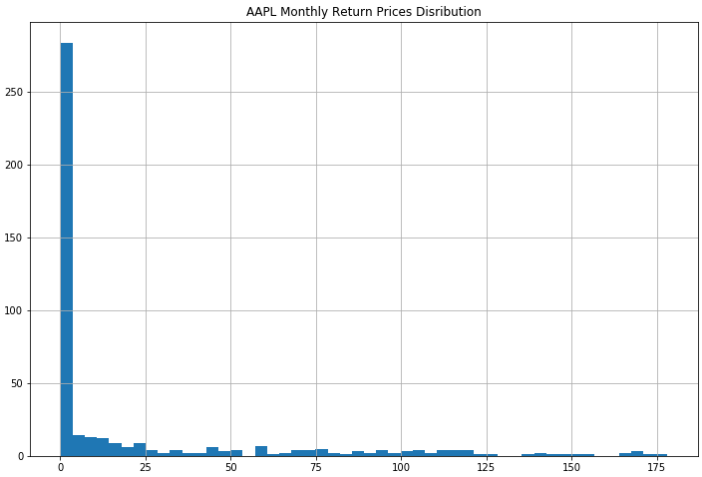
Cumulative Monthly Return

```
def monthly_cum_return(series):
    cum_daily_return = cum_return(series)
    # Resample the cumulative daily return to cumulative monthly return
    cum_monthly_return = cum_daily_return.resample("M").mean()
    return cum_monthly_return
```

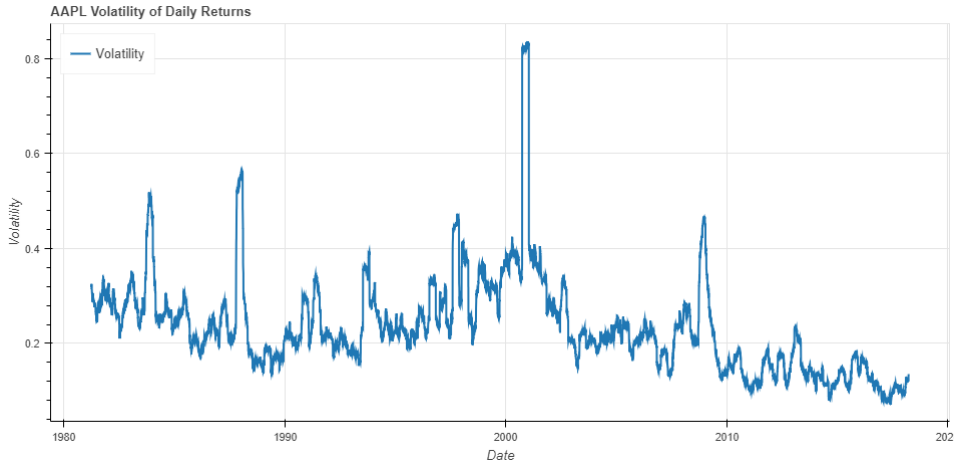
Daily Return Distribution



Monthly Return Distribution

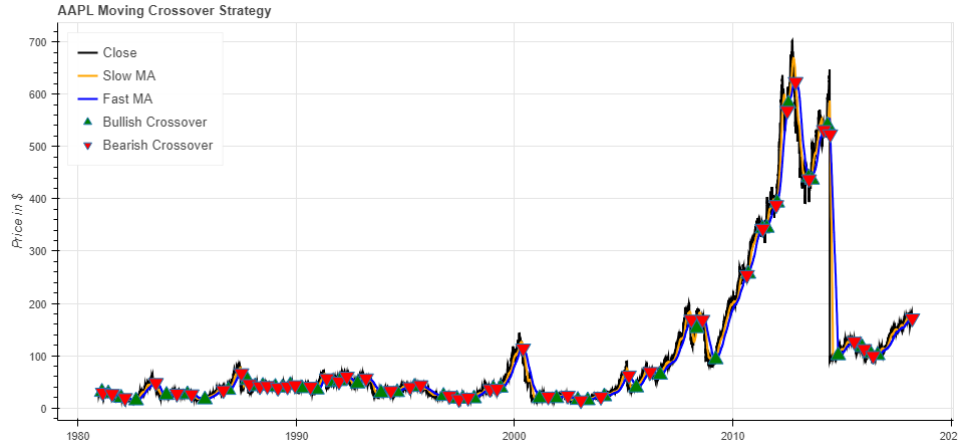


Volatility of Stock Returns

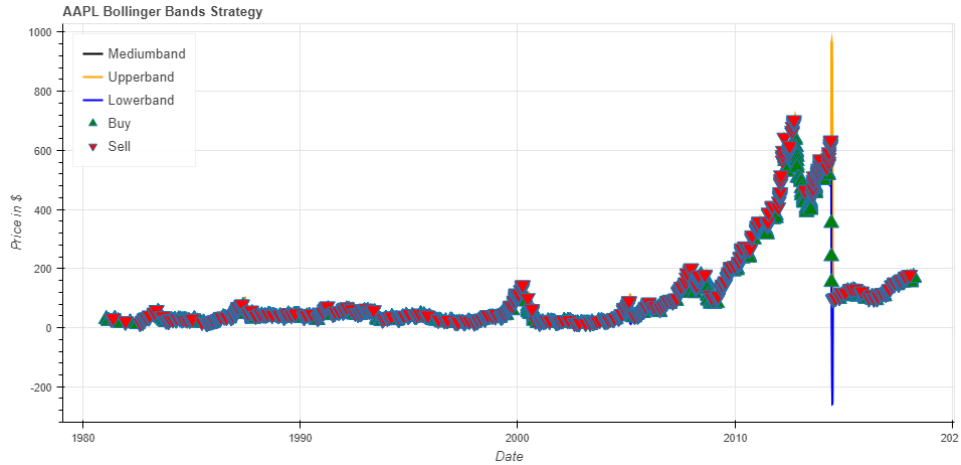


Backtesting Strategies

Moving Crossover Strategy



Bollinger Bands® Indicator



5.4.2 Microsoft Corp.

Getting Financial Data

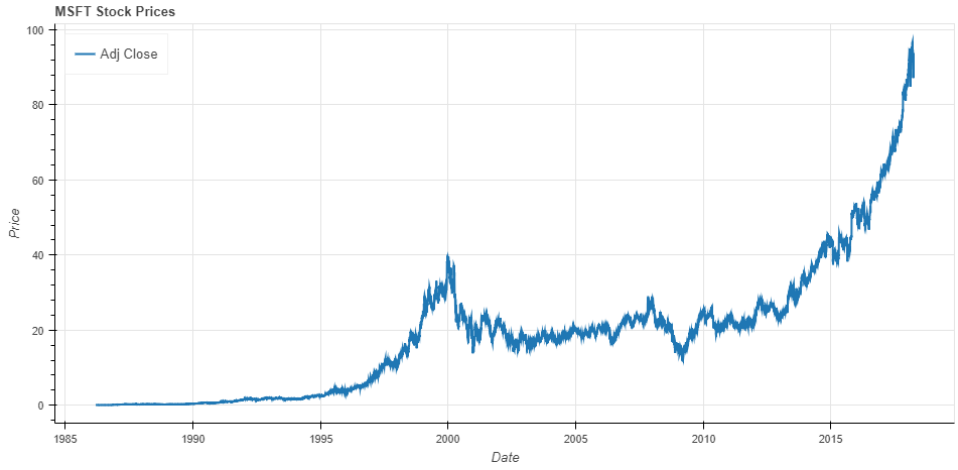
Requesting financial dataset using Quandl API:

```
msft = quandl.get("WIKI/MSFT", start_date=start, end_date=end)
```

Descriptive Statistics

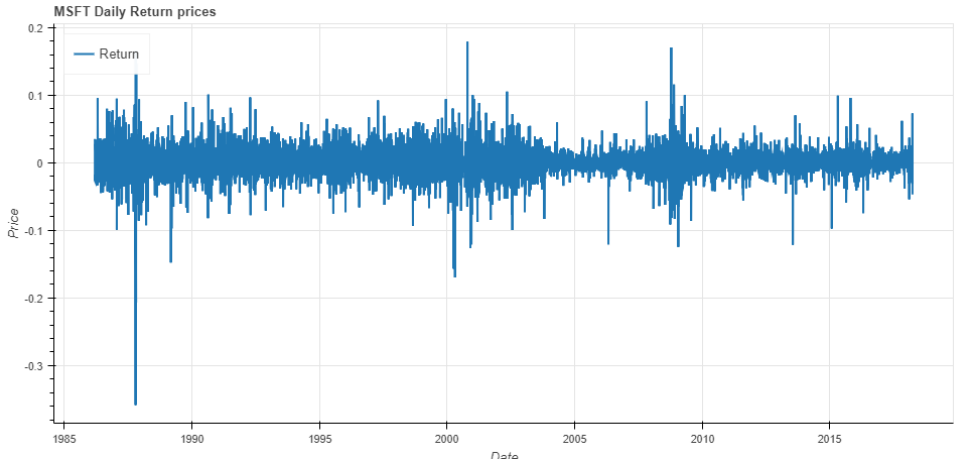
	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close
count	8076.000000	8076.000000	8076.000000	8076.000000	8.076000e+03	8076.000000	8076.000000	8076.000000	8076.000000	8076.000000	8076.000000
mean	59.832154	60.666713	59.025261	59.874821	3.125868e+07	0.001735	1.000991	18.763583	18.964604	18.564394	18.768430
std	32.526935	33.033695	32.017552	32.562926	3.204125e+07	0.038948	0.030460	17.756714	17.901221	17.607812	17.760955
min	15.200000	15.620000	14.870000	15.150000	8.000000e+03	0.000000	1.000000	0.058941	0.061253	0.058941	0.060097
25%	28.940000	29.250000	28.660000	28.940000	2.370594e+06	0.000000	1.000000	1.908662	1.934665	1.882658	1.908662
50%	54.250000	55.005000	53.500000	54.250000	2.617920e+07	0.000000	1.000000	18.581468	18.856998	18.366597	18.594782
75%	83.750000	84.955000	82.702500	83.870000	4.991018e+07	0.000000	1.000000	24.045268	24.273922	23.802299	24.060766
max	178.940000	180.380000	175.750000	179.940000	5.910522e+08	3.080000	2.000000	97.000000	97.240000	96.040000	96.770000

Data Preprocessing

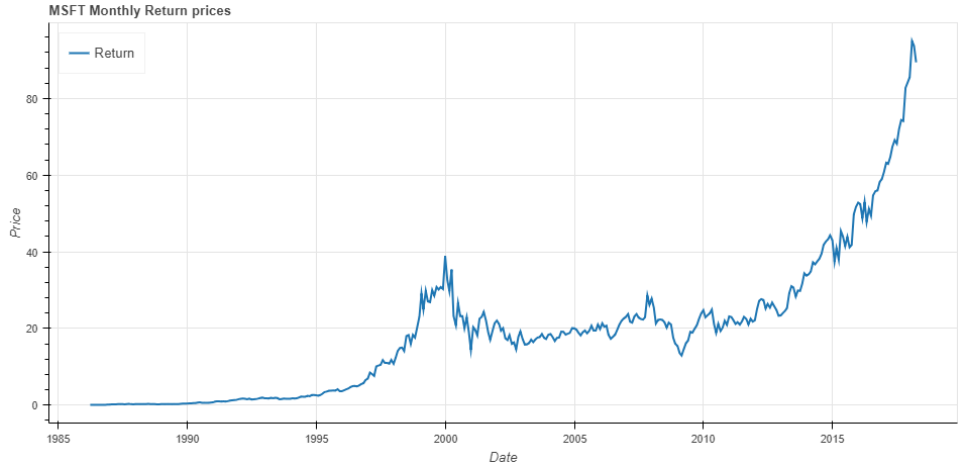


Return of Stocks

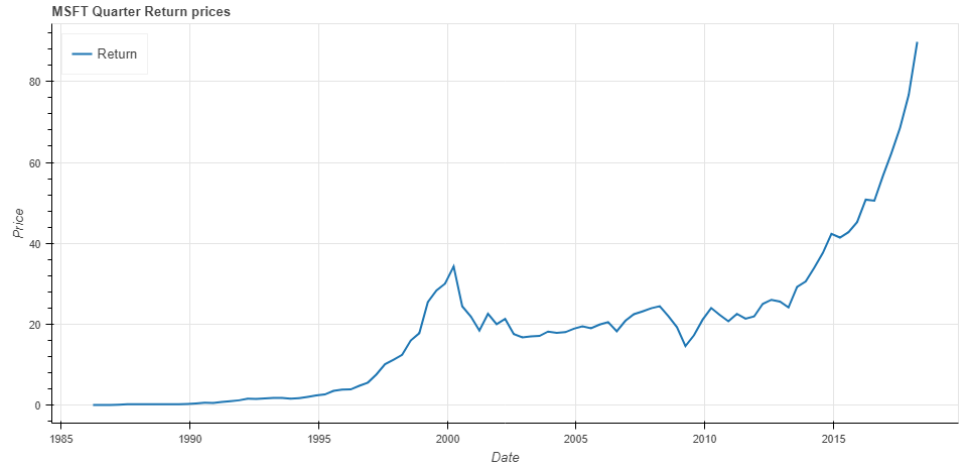
Daily Return



Monthly Return

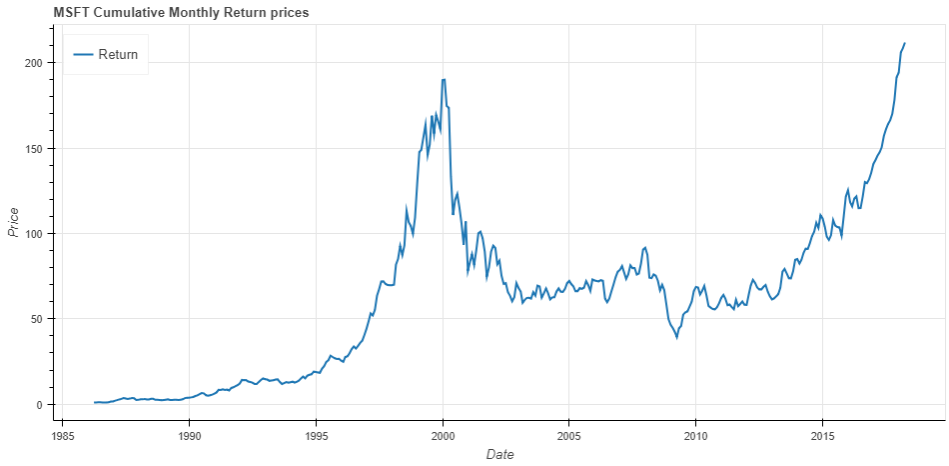


Quarter Return

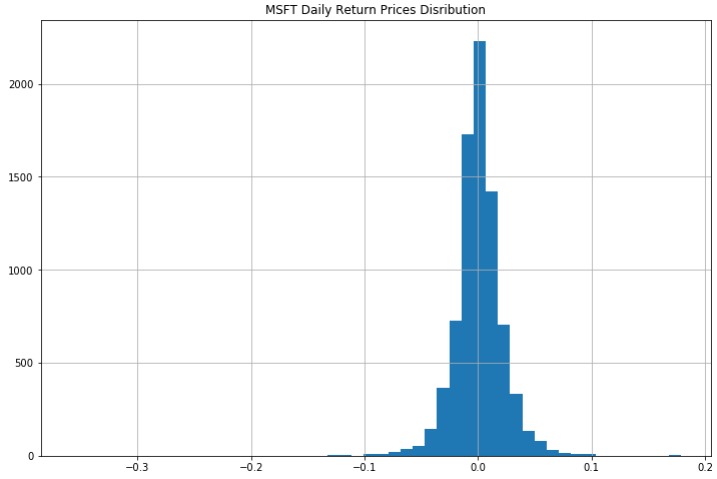


Cumulative Daily Return

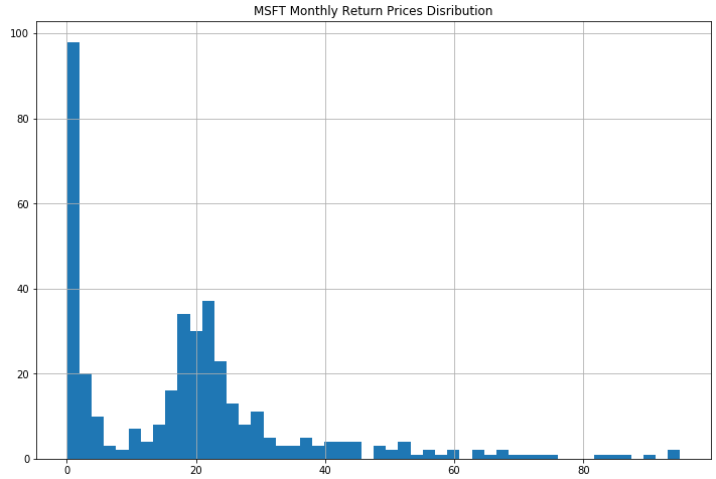
Cumulative Monthly Return



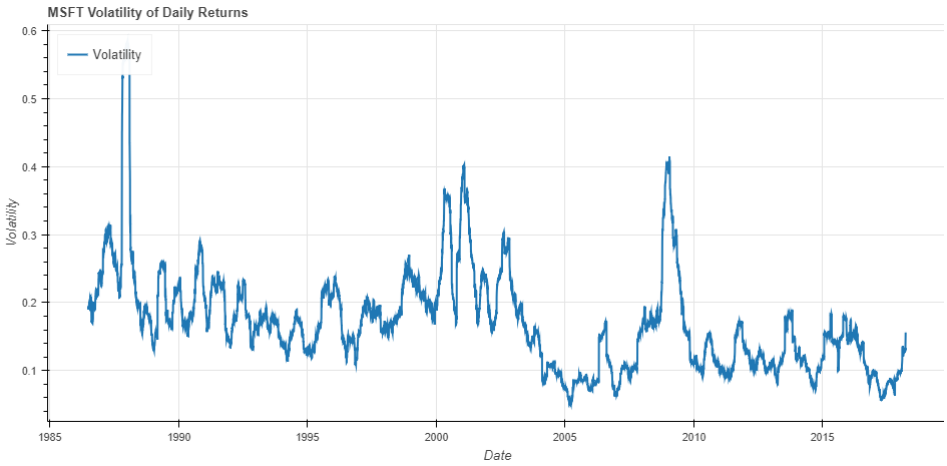
Daily Return Distribution



Monthly Return Distribution

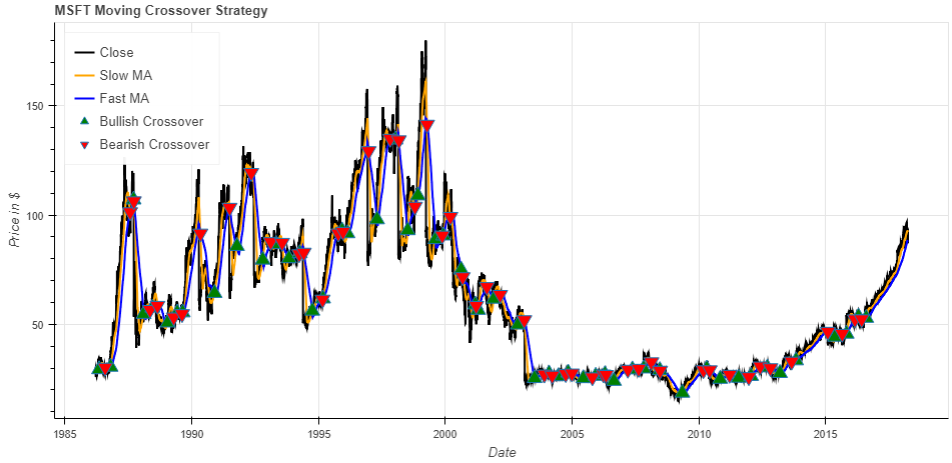


Volatility of Stock Returns

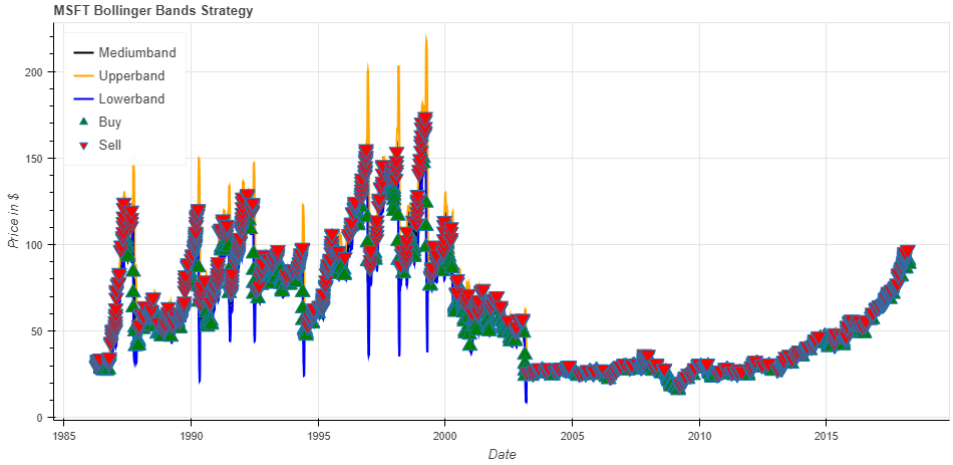


Backtesting Strategies

Moving Crossover Strategy



Bollinger Bands® Indicator



5.4.3 Google Inc.

Getting Financial Data

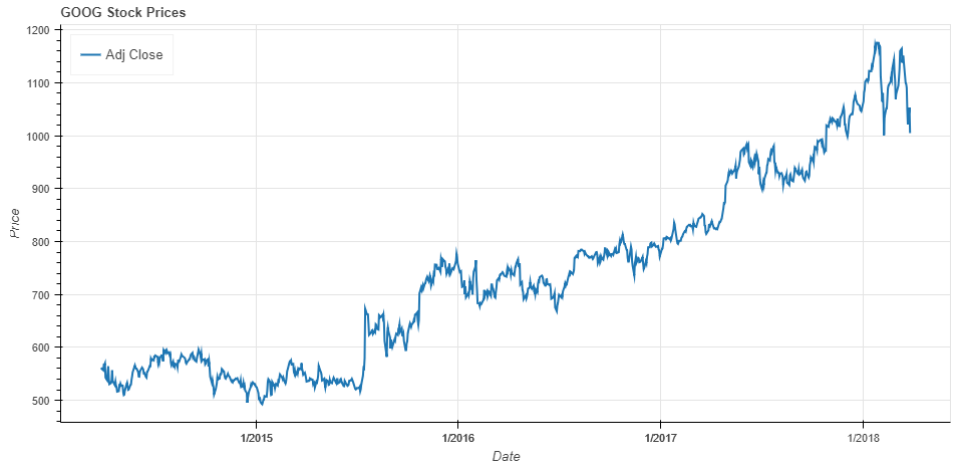
Requesting financial dataset using Quandl API:

```
goog = quandl.get("WIKI/GOOG ", start_date=start, end_date=end)
```

Descriptive Statistics

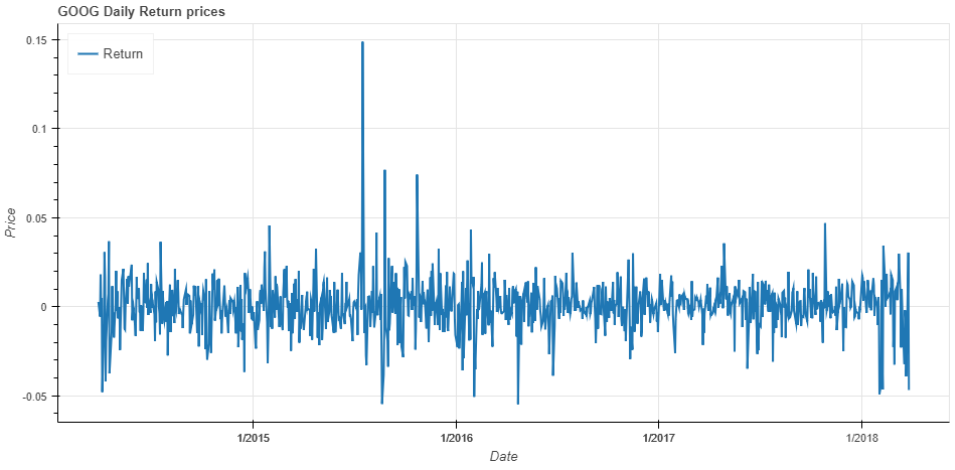
	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close	Adj. Volume
count	1007.000000	1007.000000	1007.000000	1007.000000	1.007000e+03	1007.0	1007.0	1007.000000	1007.000000	1007.000000	1007.000000	1.007000e+03
mean	737.247481	742.932792	730.986536	737.170777	1.801181e+06	0.0	1.0	737.247481	742.932792	730.986536	737.170777	1.801181e+06
std	176.154120	177.387911	175.051779	176.332774	9.367890e+05	0.0	0.0	176.154120	177.387911	175.051779	176.332774	9.367890e+05
min	494.650000	495.976000	487.560000	492.550000	7.900000e+03	0.0	1.0	494.650000	495.976000	487.560000	492.550000	7.900000e+03
25%	567.100000	572.205000	561.700000	567.422500	1.249419e+06	0.0	1.0	567.100000	572.205000	561.700000	567.422500	1.249419e+06
50%	726.810000	733.100000	720.500000	728.110000	1.576286e+06	0.0	1.0	726.810000	733.100000	720.500000	728.110000	1.576286e+06
75%	832.950000	837.070000	828.820000	832.030000	2.051788e+06	0.0	1.0	832.950000	837.070000	828.820000	832.030000	2.051788e+06
max	1177.330000	1186.890000	1171.980000	1175.840000	1.116494e+07	0.0	1.0	1177.330000	1186.890000	1171.980000	1175.840000	1.116494e+07

Data Preprocessing

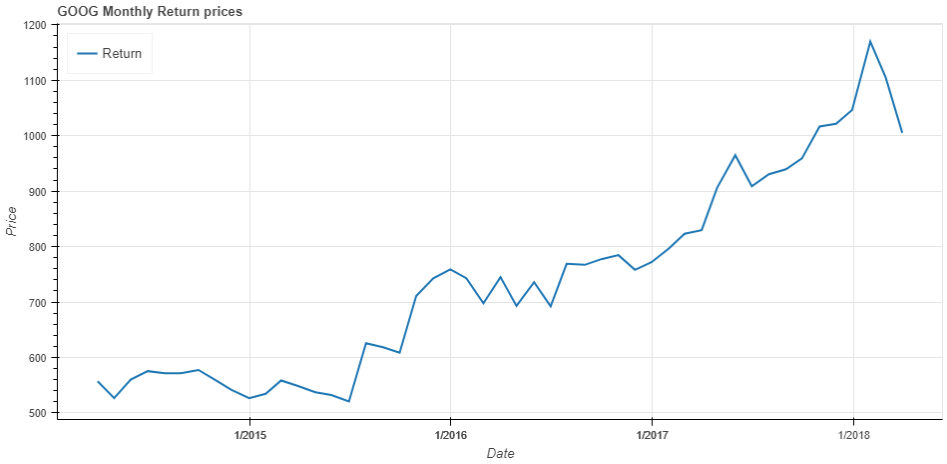


Return of Stocks

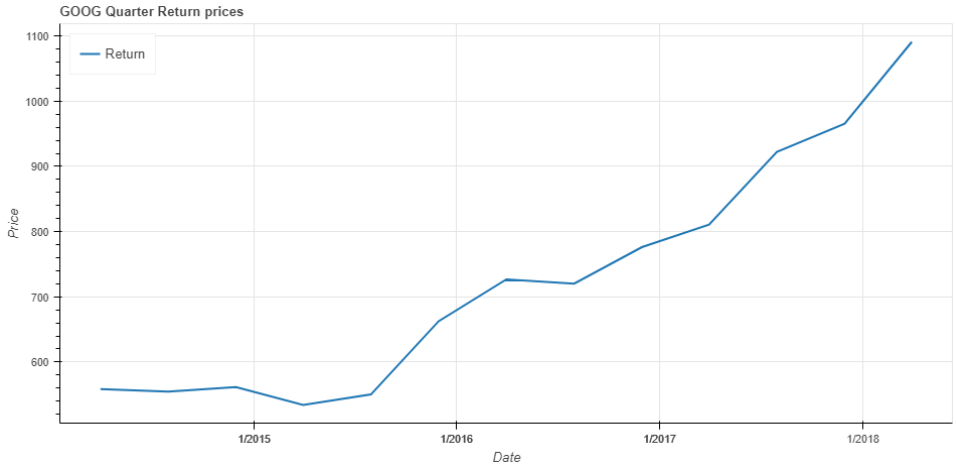
Daily Return



Monthly Return

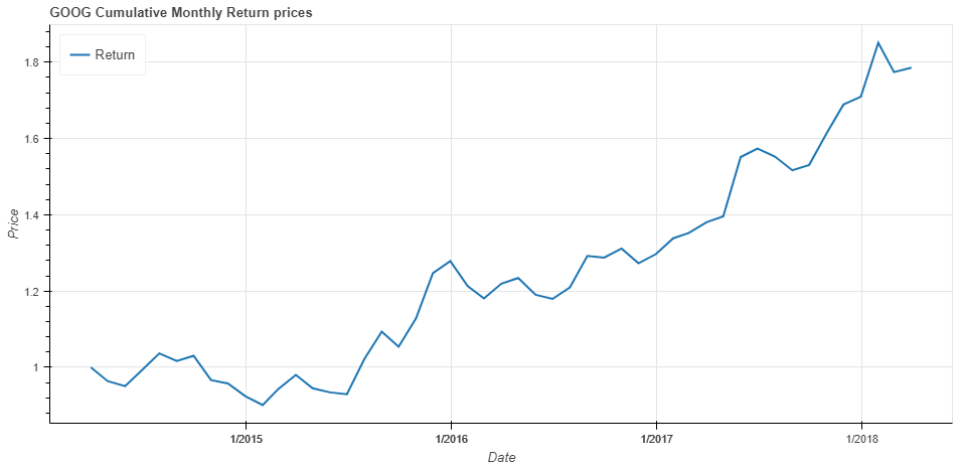


Quarter Return

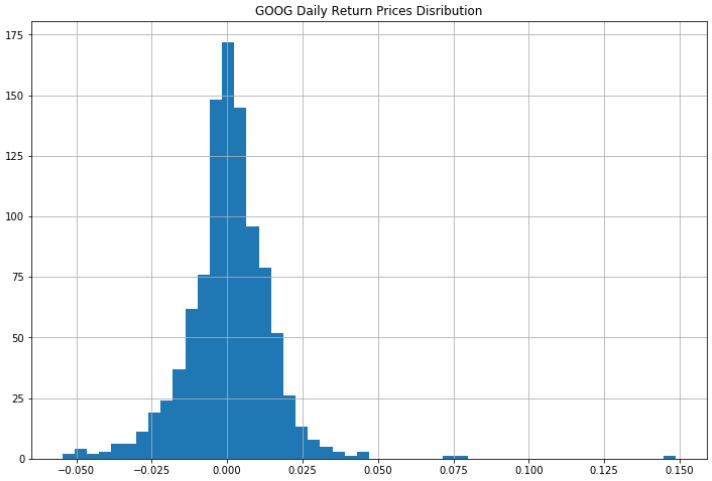


Cumulative Daily Return

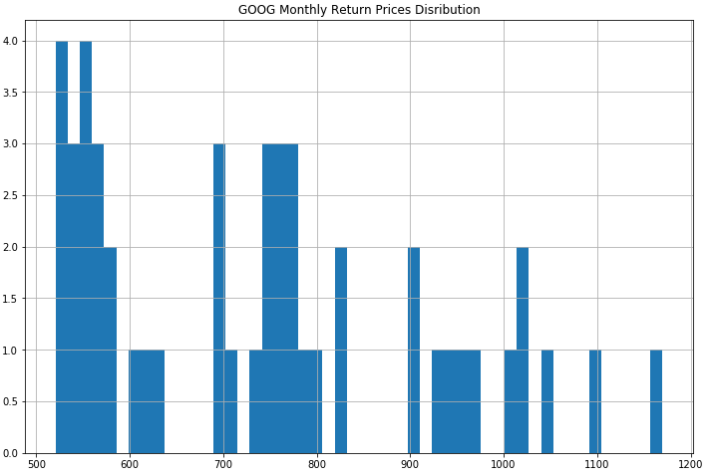
Cumulative Monthly Return



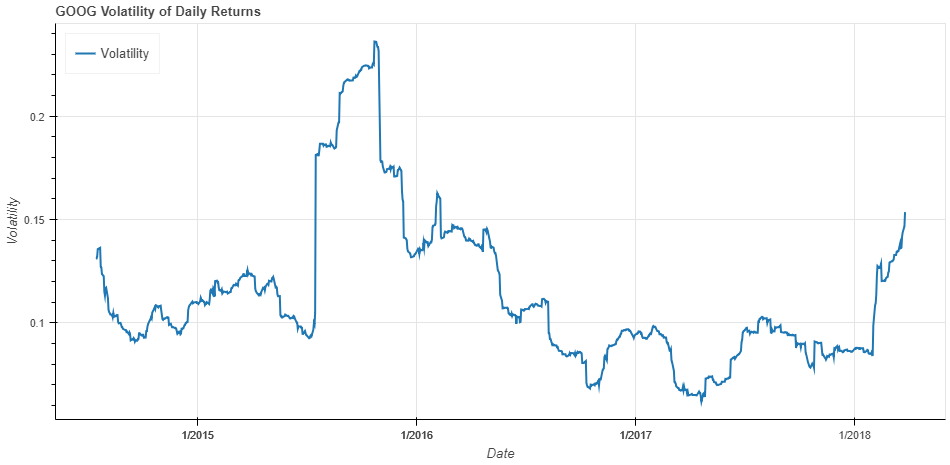
Daily Return Distribution



Monthly Return Distribution

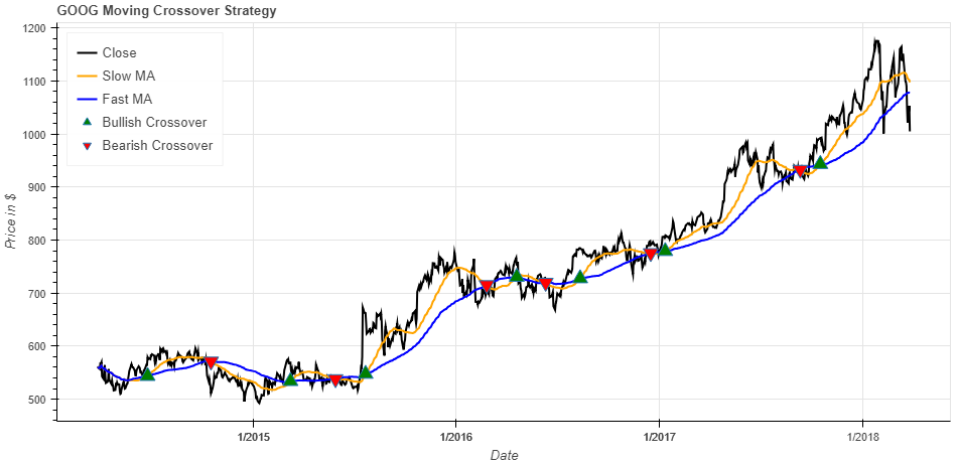


Volatility of Stock Returns

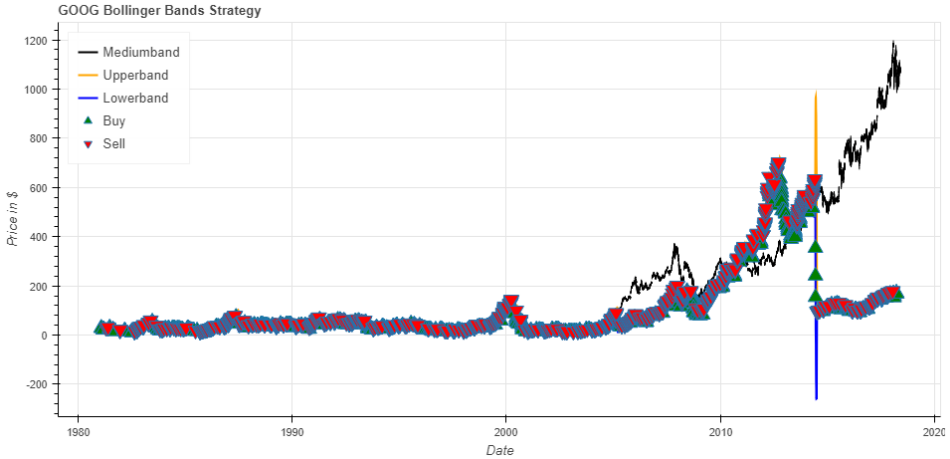


Backtesting Strategies

Moving Crossover Strategy



Bollinger Bands® Indicator



5.4.4 Facebook Inc.

Getting Financial Data

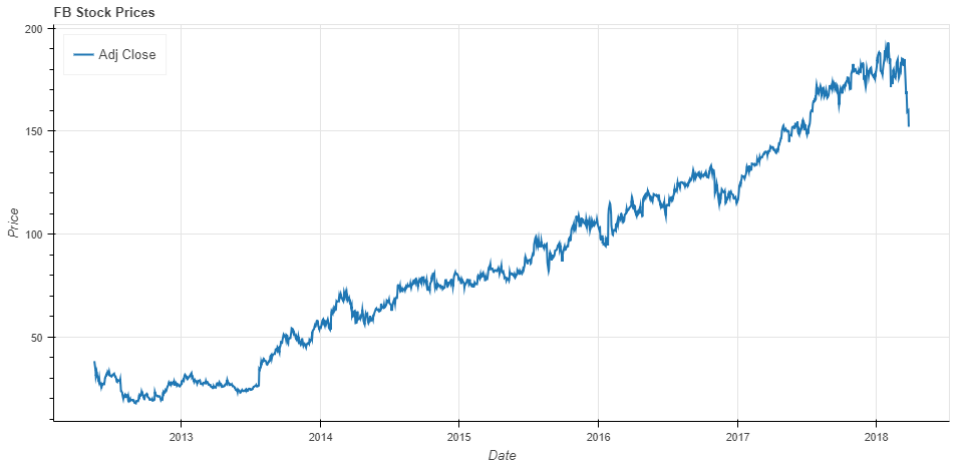
Requesting financial dataset using Quandl API:

```
fb = quandl.get("WIKI/FB", start_date=start, end_date=end)
```

Descriptive Statistics

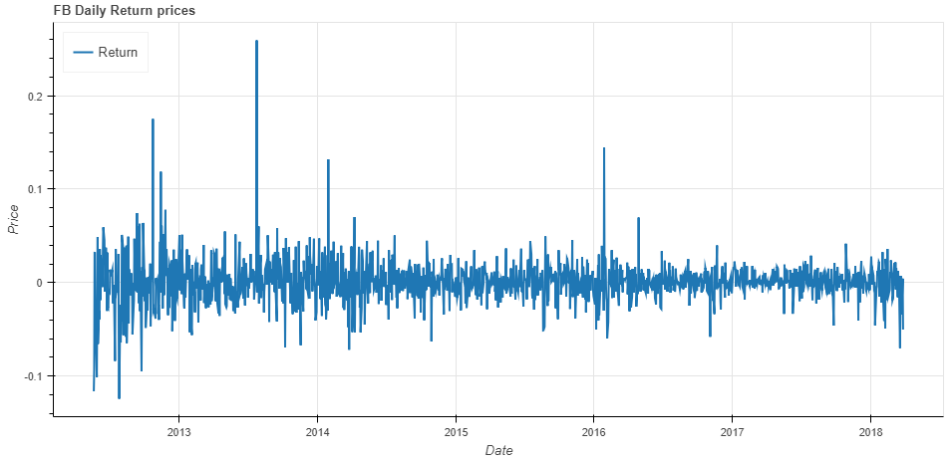
	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close
count	1472.000000	1472.000000	1472.000000	1472.000000	1.472000e+03	1472.0	1472.0	1472.000000	1472.000000	1472.000000	1472.000000
mean	89.486727	90.402976	88.494512	89.482903	3.719587e+07	0.0	1.0	89.486727	90.402976	88.494512	89.482903
std	48.507101	48.789958	48.185146	48.536888	3.243733e+07	0.0	0.0	48.507101	48.789958	48.185146	48.536888
min	18.080000	18.270000	17.550000	17.729000	5.913066e+06	0.0	1.0	18.080000	18.270000	17.550000	17.729000
25%	49.237500	50.202500	48.520000	49.167250	1.835496e+07	0.0	1.0	49.237500	50.202500	48.520000	49.167250
50%	81.025000	81.895000	80.385000	80.902500	2.718119e+07	0.0	1.0	81.025000	81.895000	80.385000	80.902500
75%	123.615000	124.715000	122.905000	123.902500	4.585621e+07	0.0	1.0	123.615000	124.715000	122.905000	123.902500
max	192.040000	195.320000	189.980000	193.090000	5.735764e+08	0.0	1.0	192.040000	195.320000	189.980000	193.090000

Data Preprocessing

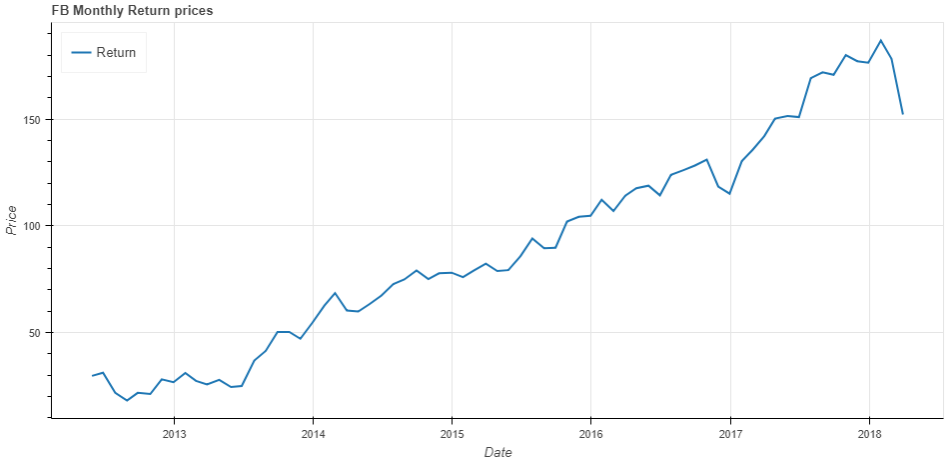


Return of Stocks

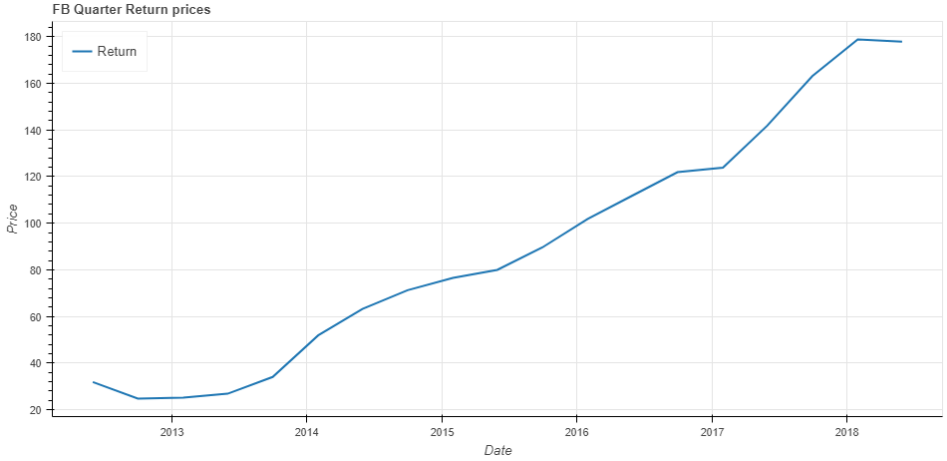
Daily Return



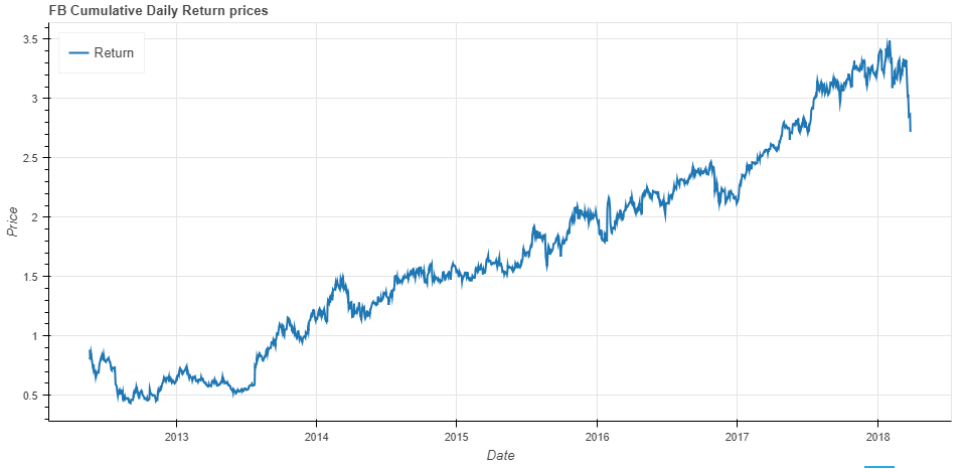
Monthly Return



Quarter Return

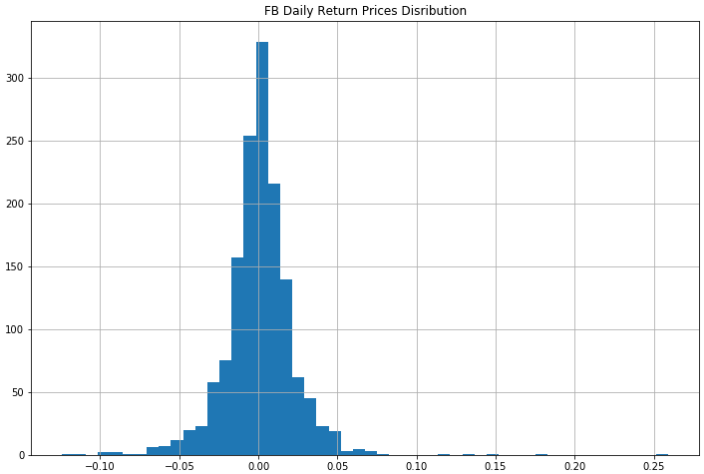


Cumulative Daily Return

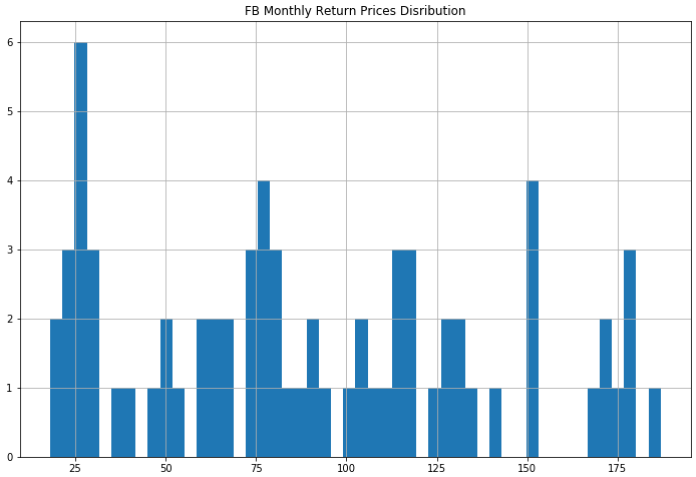


Cumulative Monthly Return

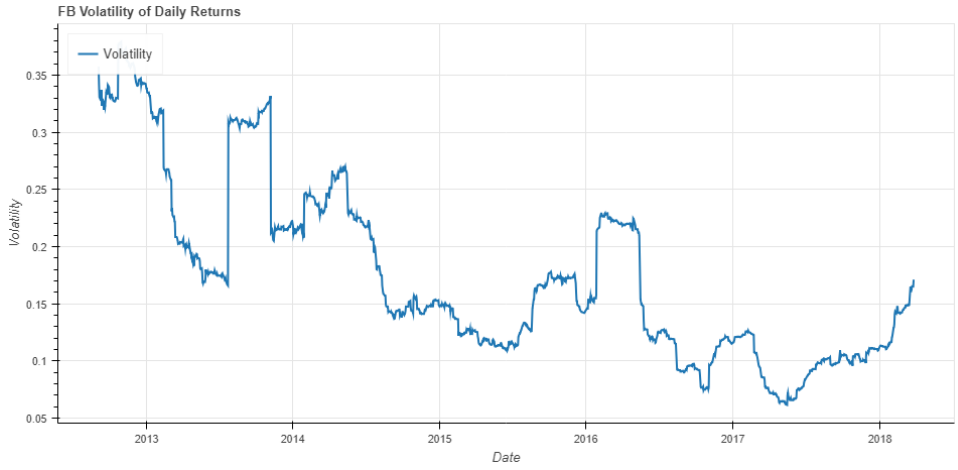
Daily Return Distribution



Monthly Return Distribution

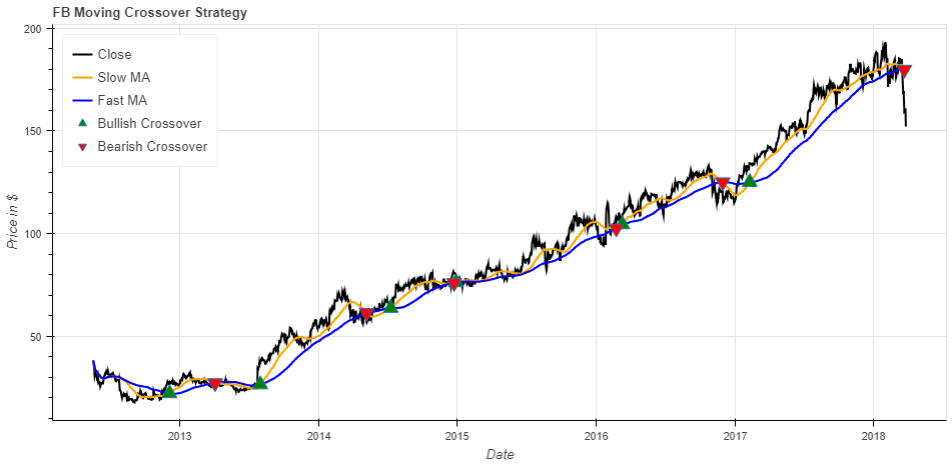


Volatility of Stock Returns

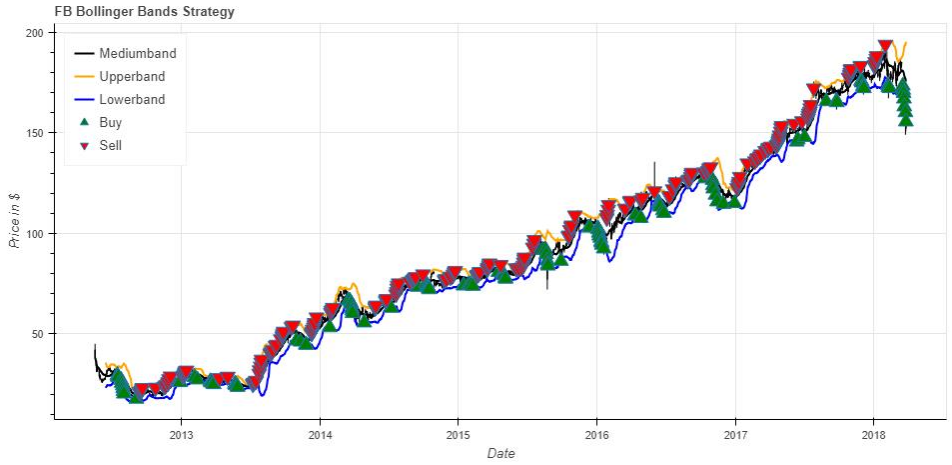


Backtesting Strategies

Moving Crossover Strategy



Bollinger Bands® Indicator



5.4.5 Twitter Inc.

Getting Financial Data

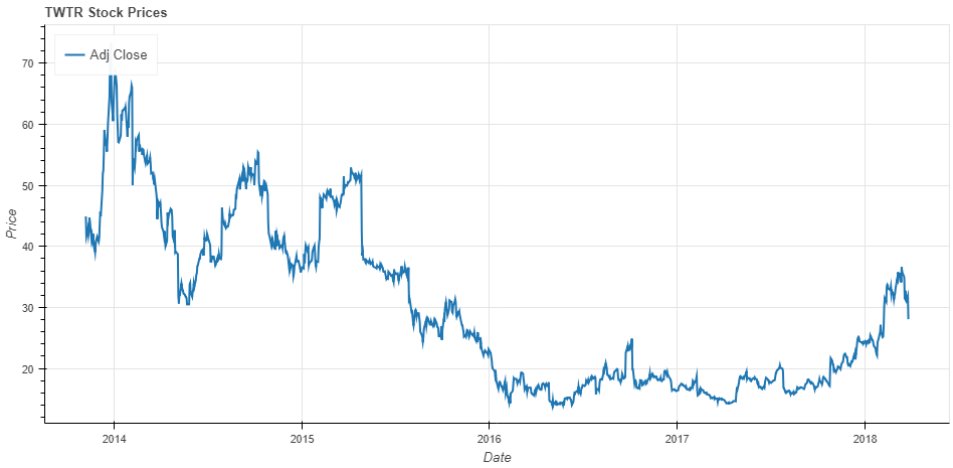
Requesting financial dataset using Quandl API:

```
Twtr = quandl.get("WIKI/TWTR", start_date=start, end_date=end)
```

Descriptive Statistics

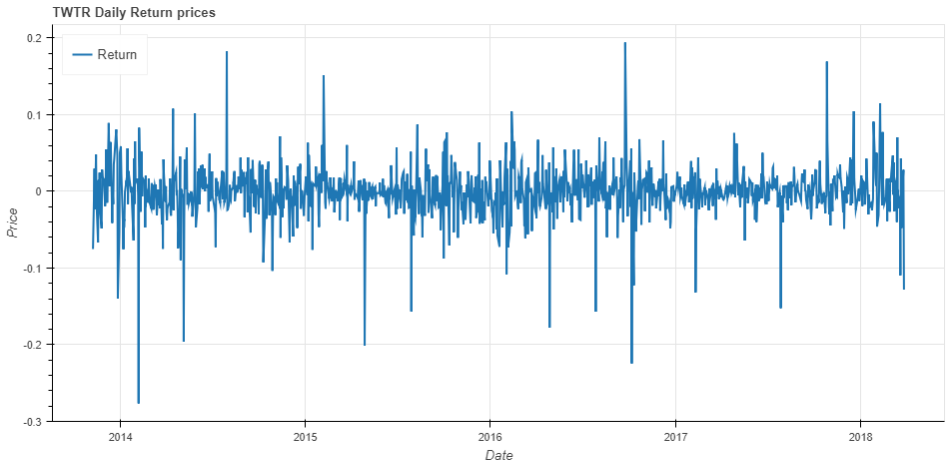
	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close	Adj. Volume
count	1102.000000	1102.000000	1102.000000	1102.000000	1.102000e+03	1102.0	1102.0	1102.000000	1102.000000	1102.000000	1102.000000	1.102000e+03
mean	29.807331	30.385377	29.213017	29.773684	2.249288e+07	0.0	1.0	29.807331	30.385377	29.213017	29.773684	2.249288e+07
std	13.619349	13.906846	13.281599	13.577790	1.682226e+07	0.0	0.0	13.619349	13.906846	13.281599	13.577790	1.682226e+07
min	13.950000	14.220000	13.725000	14.010000	4.107000e+06	0.0	1.0	13.950000	14.220000	13.725000	14.010000	4.107000e+06
25%	17.820000	18.090000	17.470000	17.782500	1.319327e+07	0.0	1.0	17.820000	18.090000	17.470000	17.782500	1.319327e+07
50%	25.485000	26.125050	24.965000	25.405000	1.785513e+07	0.0	1.0	25.485000	26.125050	24.965000	25.405000	1.785513e+07
75%	39.430000	40.167500	38.722500	39.407500	2.557620e+07	0.0	1.0	39.430000	40.167500	38.722500	39.407500	2.557620e+07
max	72.880000	74.730000	69.130000	73.310000	1.922136e+08	0.0	1.0	72.880000	74.730000	69.130000	73.310000	1.922136e+08

Data Preprocessing

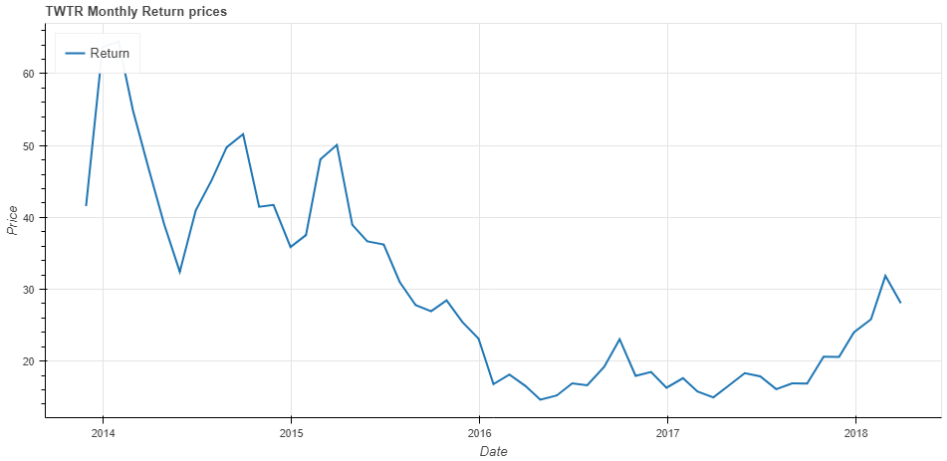


Return of Stocks

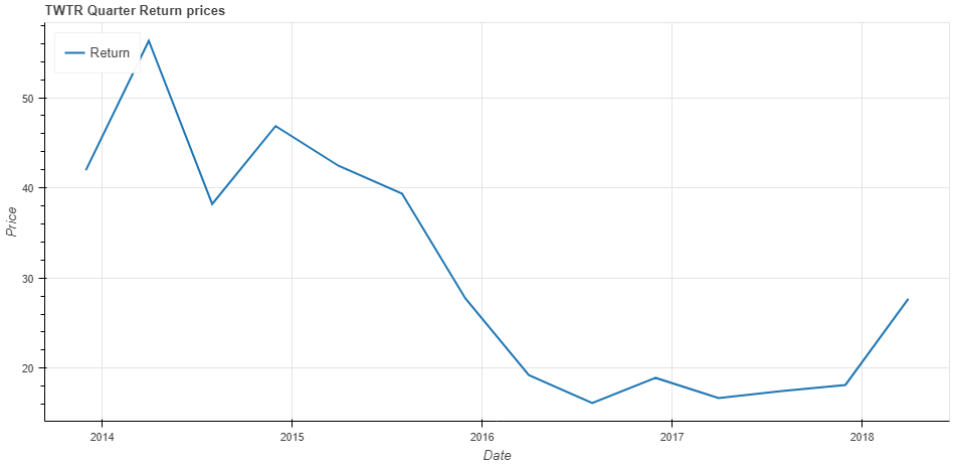
Daily Return



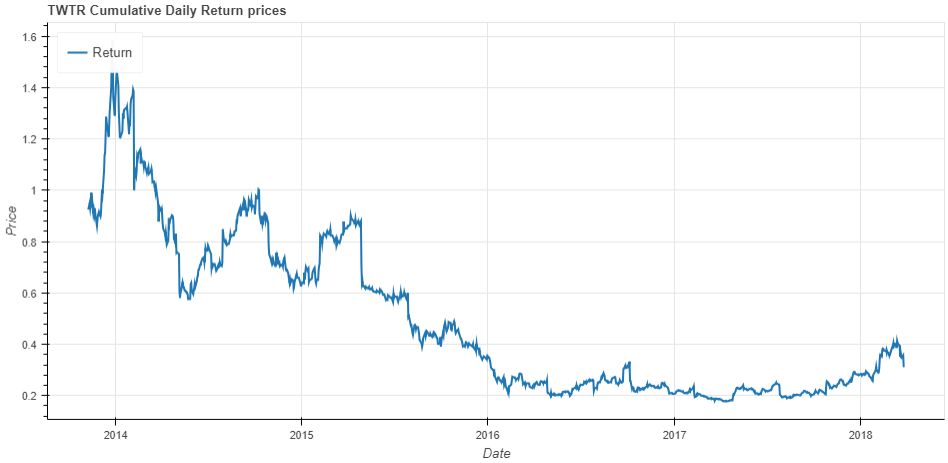
Monthly Return



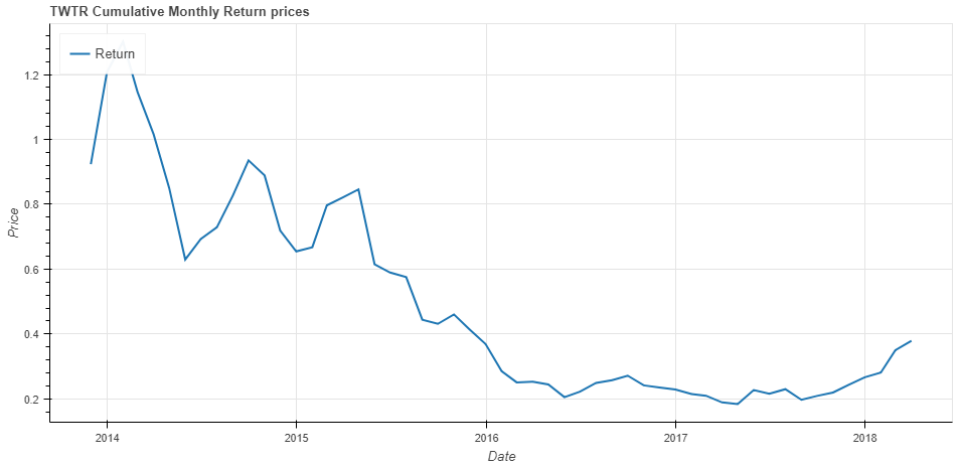
Quarter Return



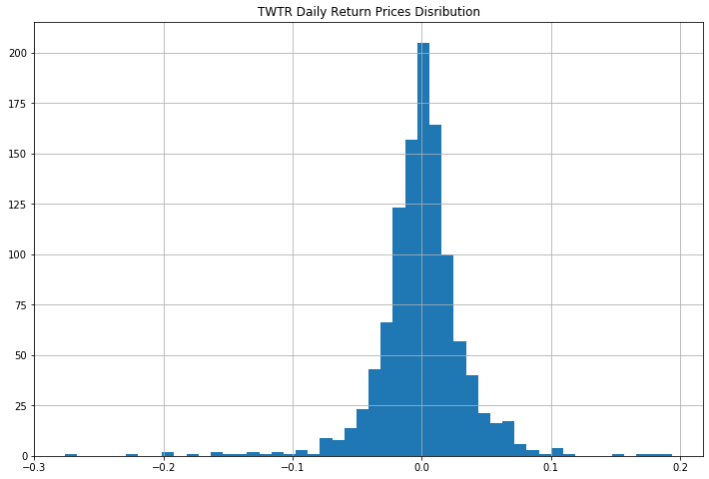
Cumulative Daily Return



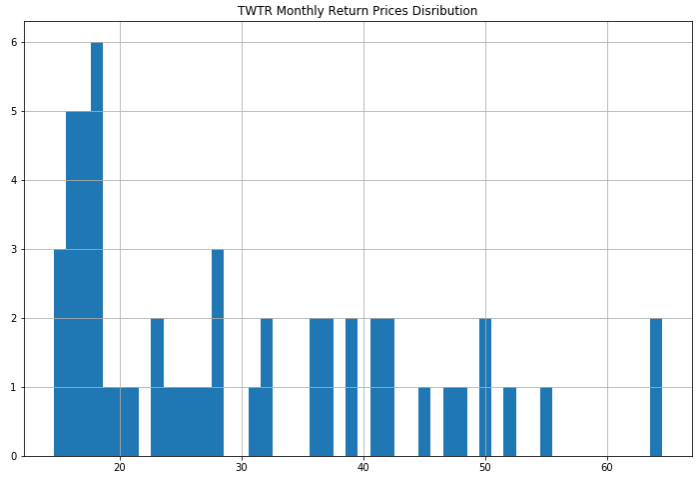
Cumulative Monthly Return



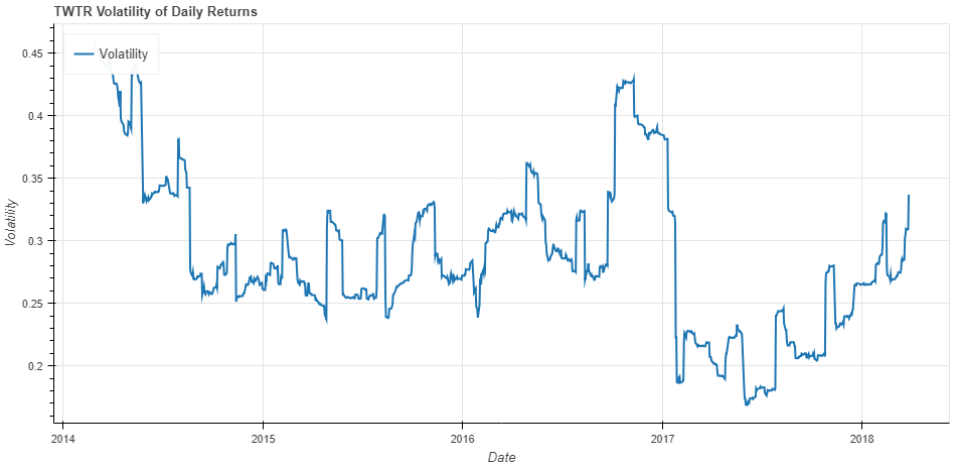
Daily Return Distribution



Monthly Return Distribution



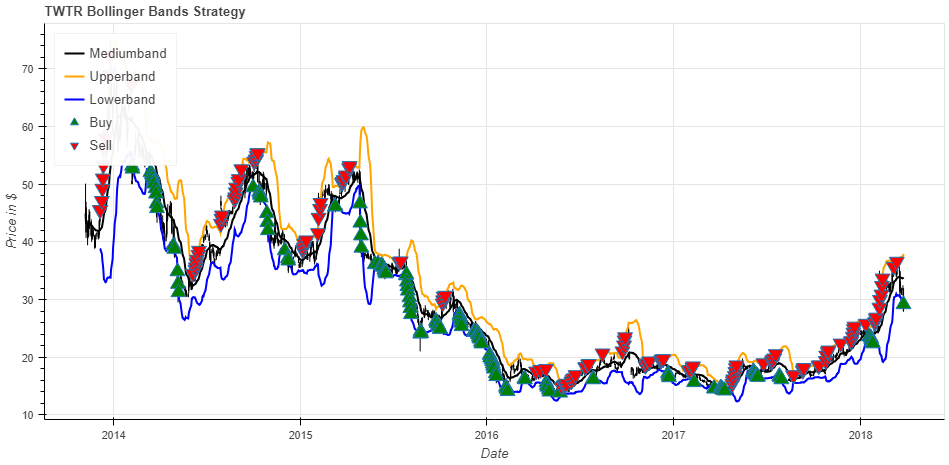
Volatility of Stock Returns



Backtesting Strategies

Moving Crossover Strategy

Bollinger Bands® Indicator



5.4.6 Yahoo! Inc.

Getting Financial Data

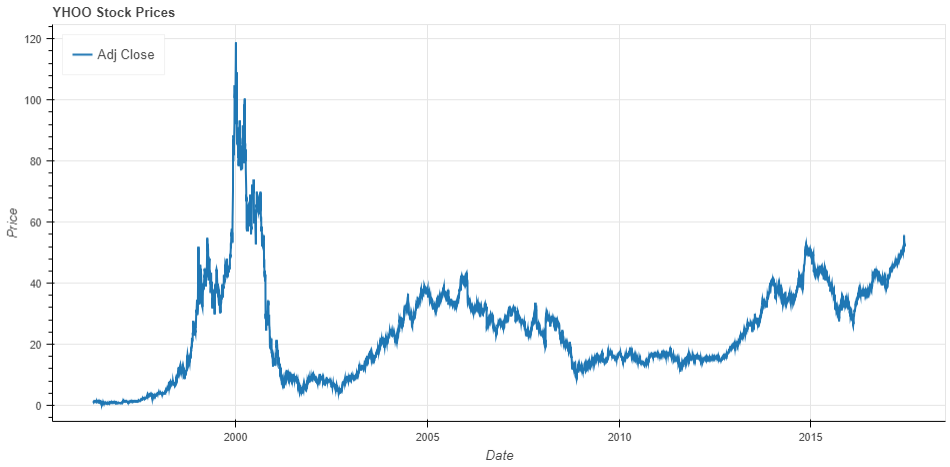
Requesting financial dataset using Quandl API:

```
yhoo = quandl.get("WIKI/YHOO", start_date=start, end_date=end)
```

Descriptive Statistics

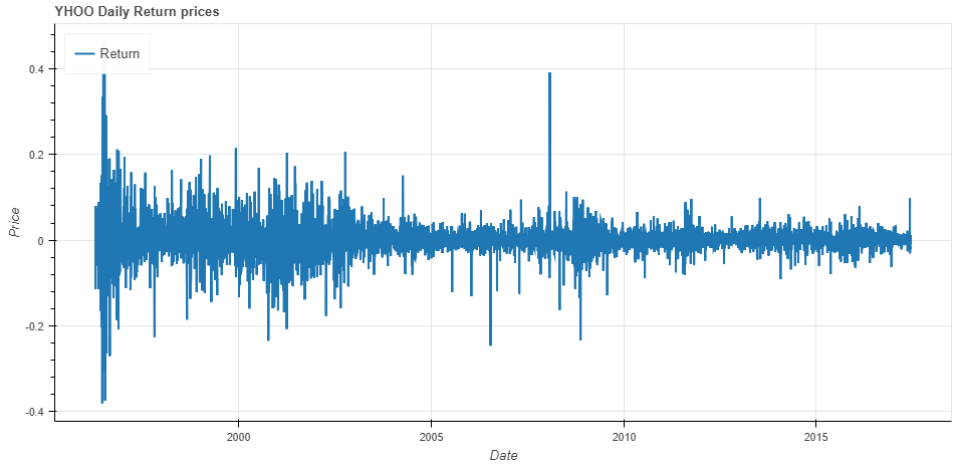
	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close
count	5332.000000	5332.000000	5332.000000	5332.000000	5.332000e+03	5332.0	5332.000000	5332.000000	5332.000000	5332.000000	5332.000000
mean	45.387832	46.551452	44.233429	45.398303	1.648195e+07	0.0	1.000844	24.685893	25.144590	24.213417	24.673363
std	54.072701	56.300208	51.985188	54.190268	1.647947e+07	0.0	0.028223	16.580440	17.002641	16.174766	16.581071
min	8.040000	9.200000	8.020000	8.110000	6.200000e+03	0.0	1.000000	0.677083	0.692708	0.656250	0.671875
25%	17.400000	17.820000	17.037500	17.487500	8.177805e+06	0.0	1.000000	13.522500	13.812750	13.180000	13.510000
50%	30.305000	30.970000	29.715000	30.250000	1.359290e+07	0.0	1.000000	22.845000	23.210000	22.460000	22.737500
75%	41.432500	42.152500	40.852525	41.432500	2.049548e+07	0.0	1.000000	34.882500	35.300000	34.350000	34.870000
max	464.500000	500.100000	442.000000	475.000000	4.382488e+08	0.0	2.000000	116.125000	125.025000	110.500000	118.750000

Data Preprocessing

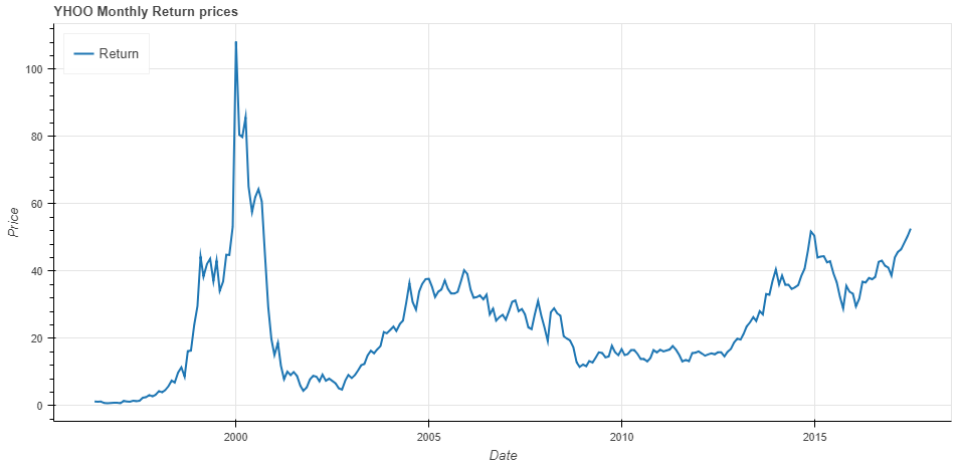


Return of Stocks

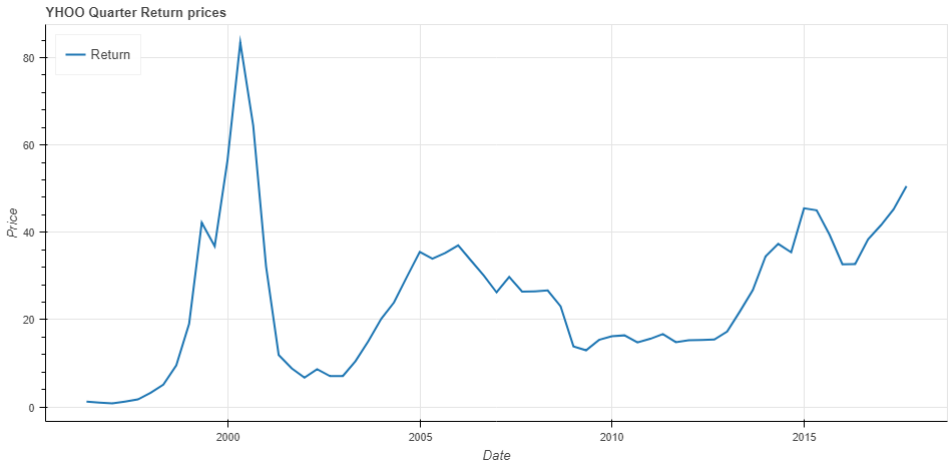
Daily Return



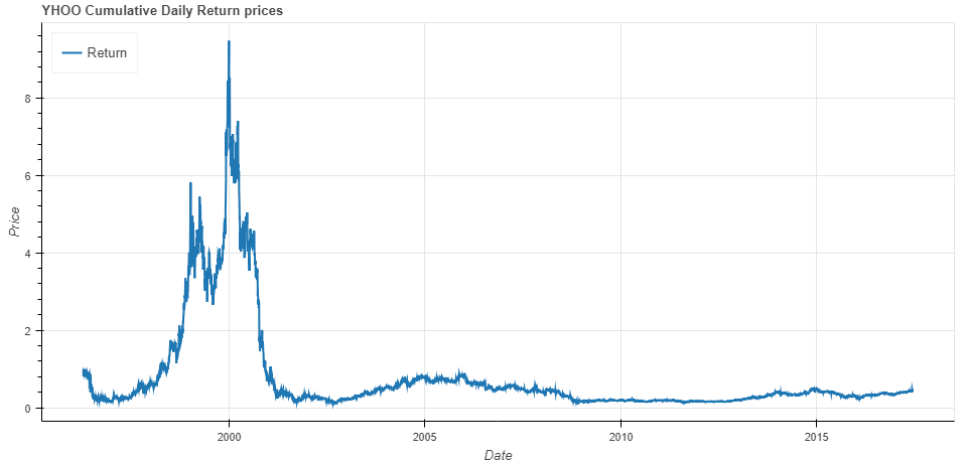
Monthly Return



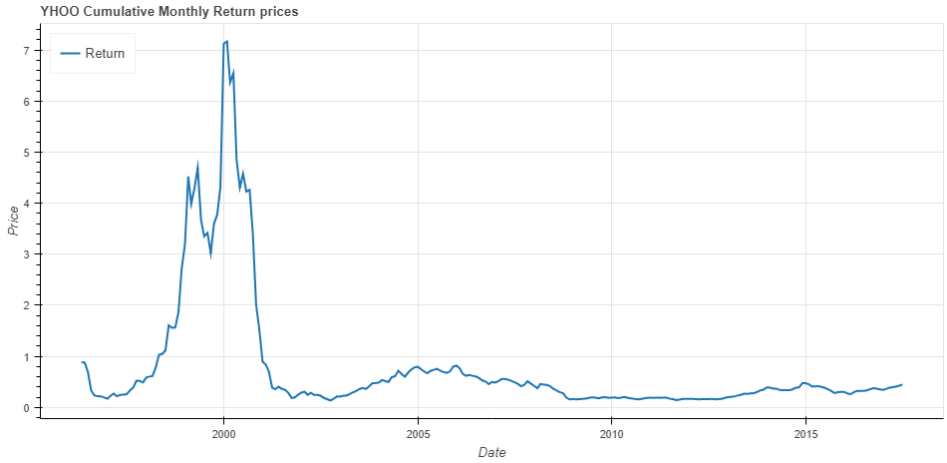
Quarter Return



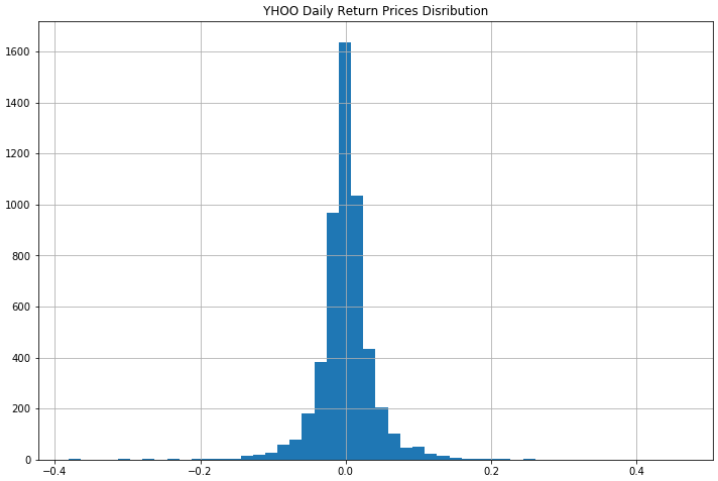
Cumulative Daily Return



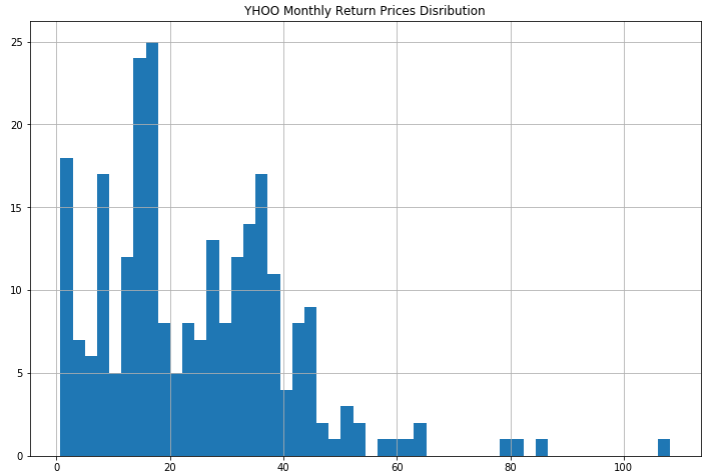
Cumulative Monthly Return



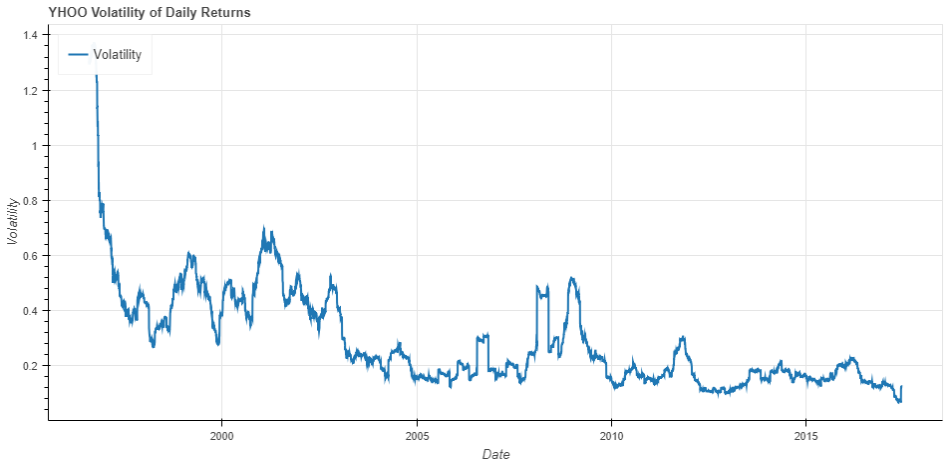
Daily Return Distribution



Monthly Return Distribution

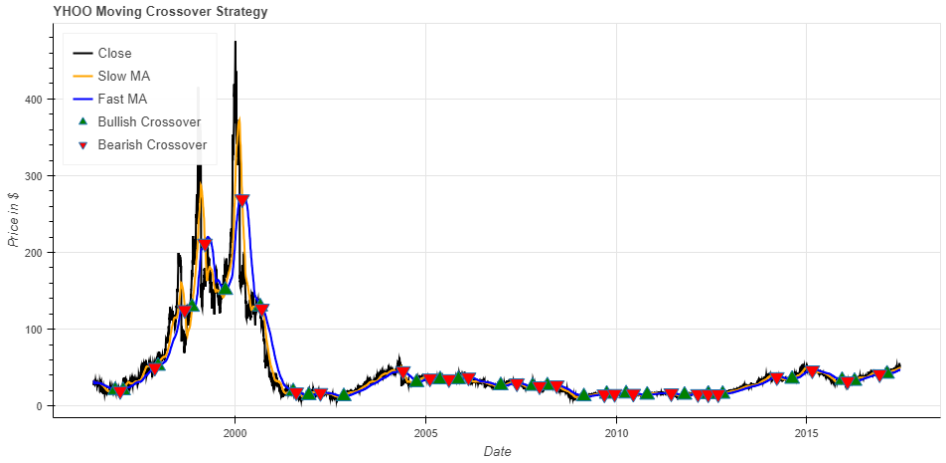


Volatility of Stock Returns

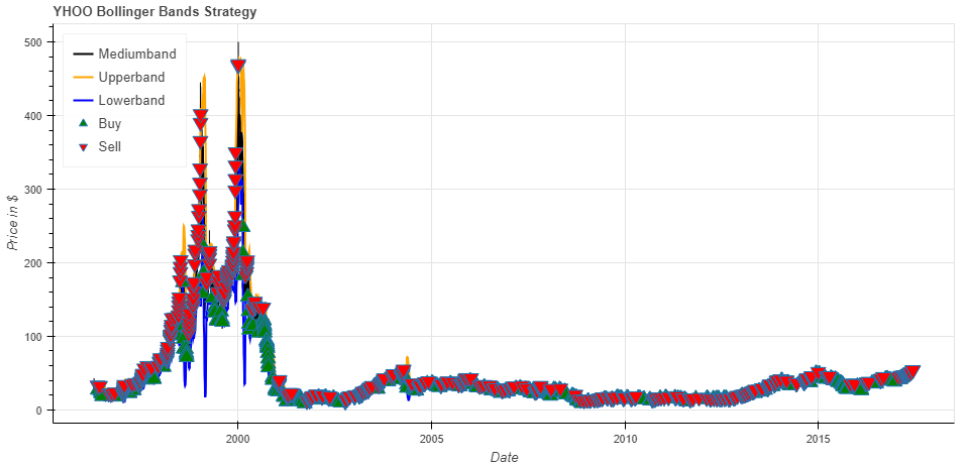


Backtesting Strategies

Moving Crossover Strategy



Bollinger Bands® Indicator



5.4.7 Electronic Arts Inc.

Getting Financial Data

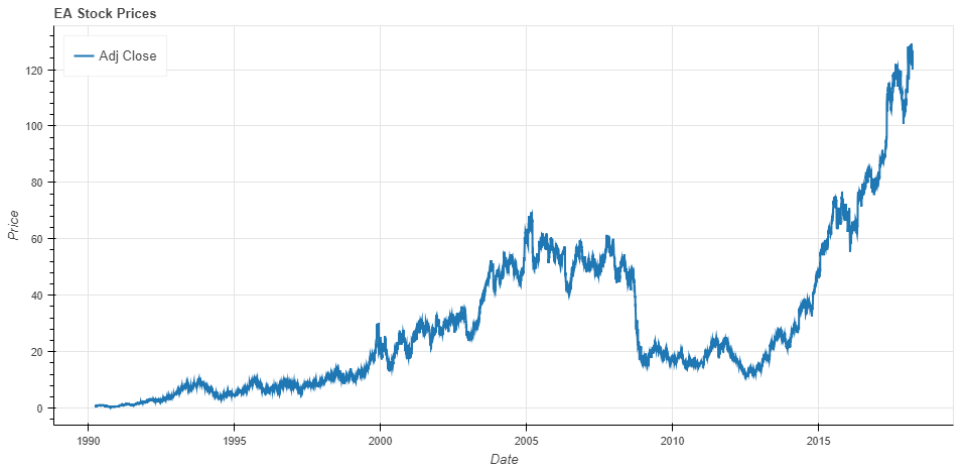
Requesting financial dataset using Quandl API:

```
ea = quandl.get("WIKI/EA", start_date=start, end_date=end)
```

Descriptive Statistics

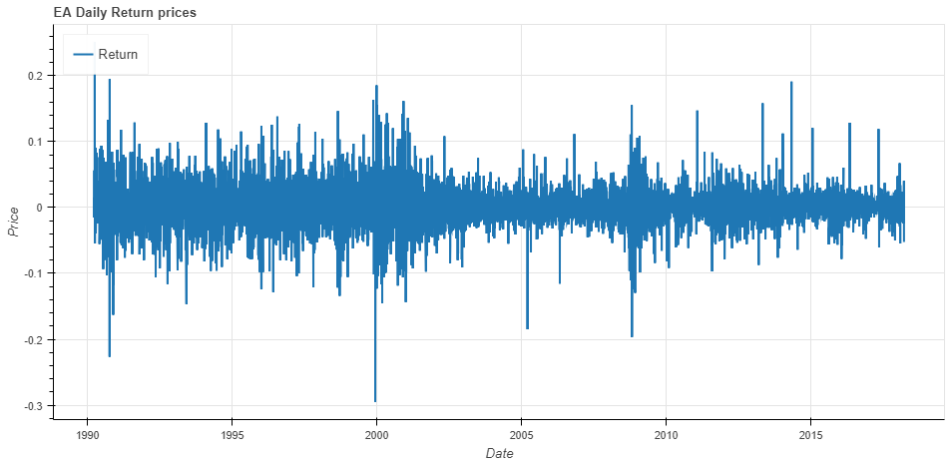
	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close
count	7056.000000	7056.000000	7056.000000	7056.000000	7.056000e+03	7056.0	7056.000000	7056.000000	7056.000000	7056.000000	7056.000000
mean	43.433853	44.244873	42.610001	43.446960	3.197403e+06	0.0	1.000567	29.700857	30.165897	29.231701	29.708054
std	24.316065	24.660085	23.959497	24.321542	3.145565e+06	0.0	0.023804	26.493737	26.762888	26.212169	26.496791
min	6.620000	7.620000	6.120000	6.380000	3.200000e+03	0.0	1.000000	0.413750	0.476250	0.382500	0.398750
25%	22.850000	23.342500	22.440000	22.877500	9.879500e+05	0.0	1.000000	8.750000	9.000000	8.562500	8.761250
50%	41.060000	42.000000	39.995000	41.010000	2.570900e+06	0.0	1.000000	21.000000	21.533750	20.441250	20.938750
75%	56.560000	57.672500	55.492500	56.592500	4.339700e+06	0.0	1.000000	48.080000	48.800000	47.410250	48.121250
max	130.350000	131.130000	128.190000	129.120000	4.739810e+07	0.0	2.000000	130.350000	131.130000	128.190000	129.120000

Data Preprocessing

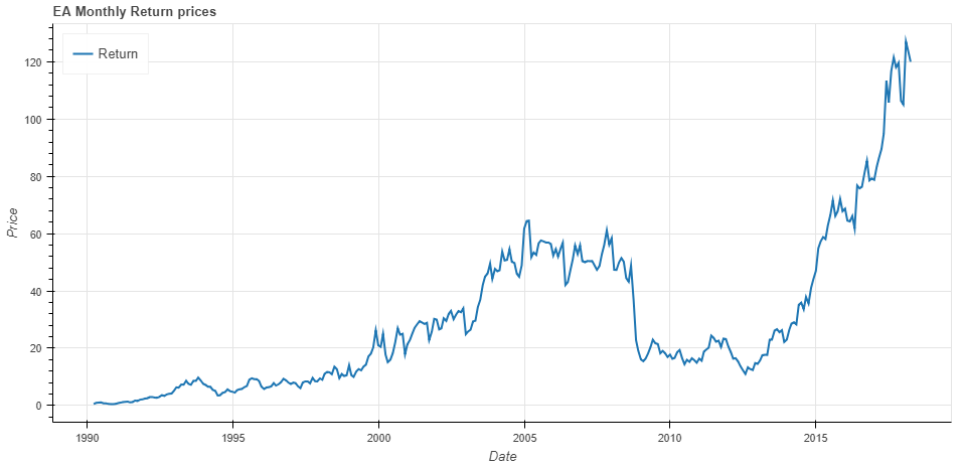


Return of Stocks

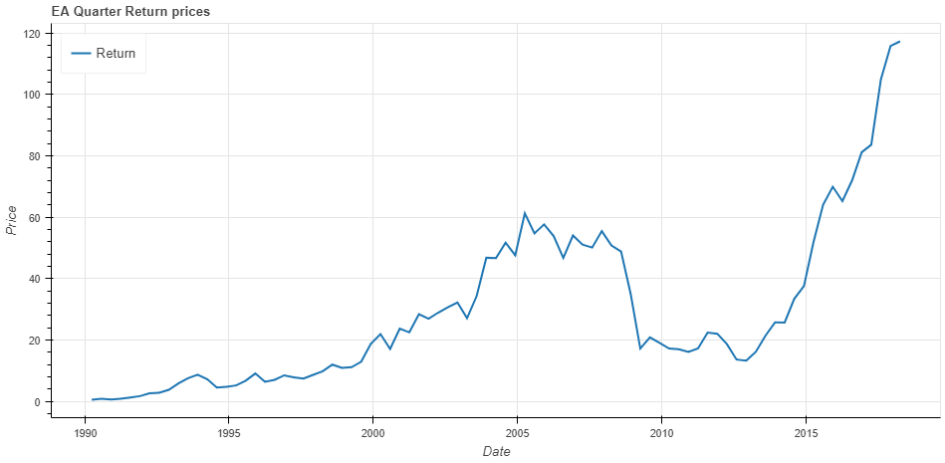
Daily Return



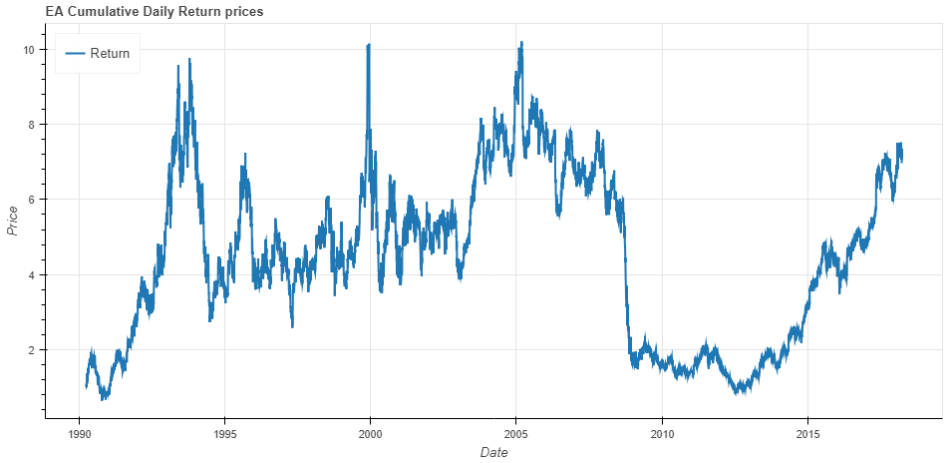
Monthly Return



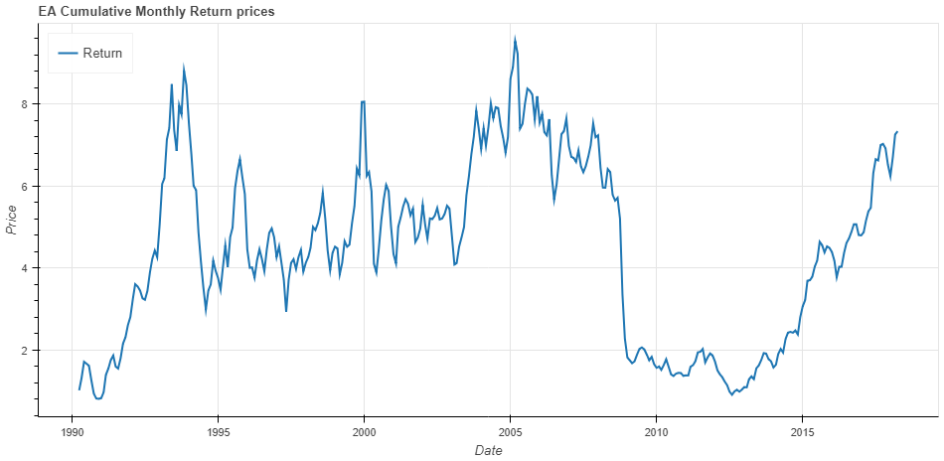
Quarter Return



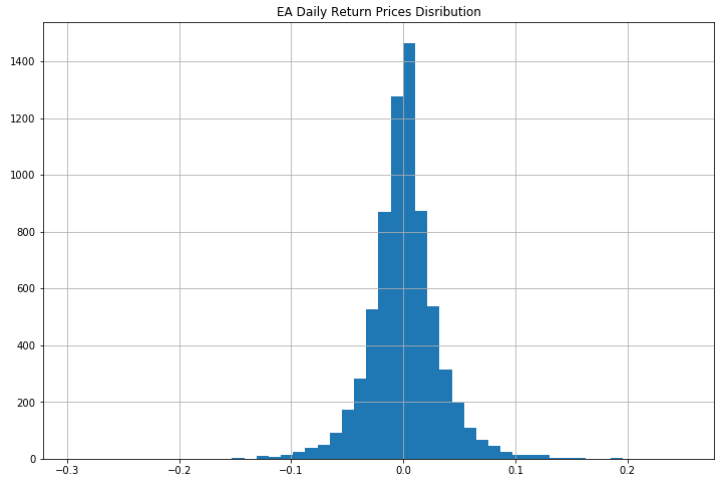
Cumulative Daily Return



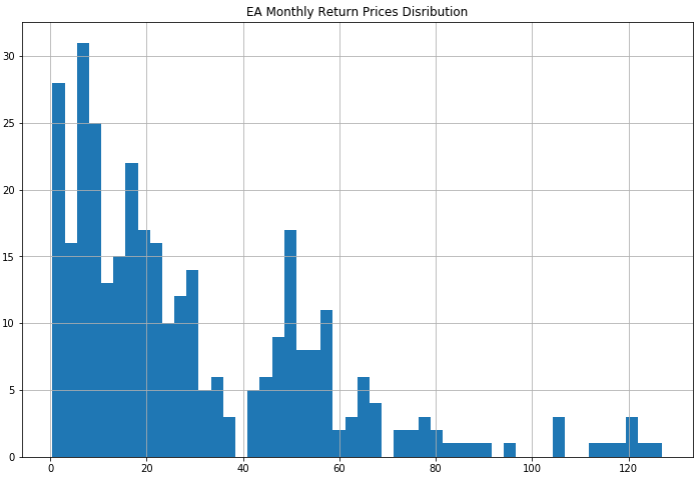
Cumulative Monthly Return



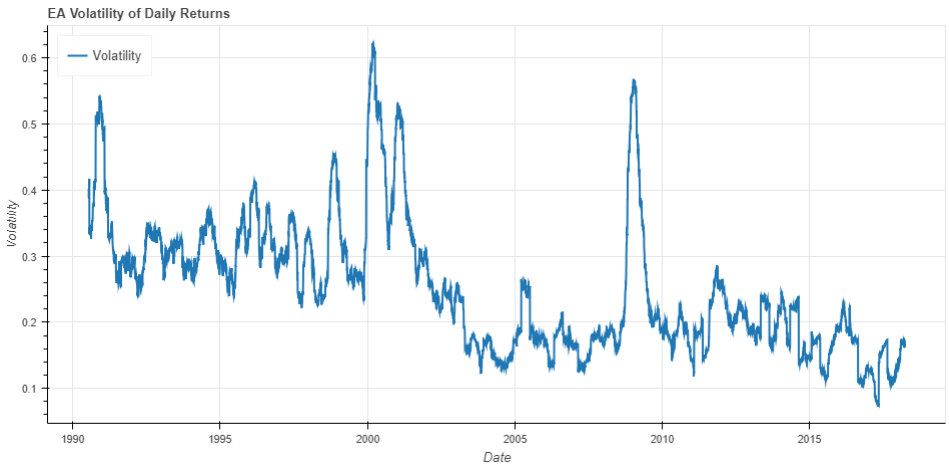
Daily Return Distribution



Monthly Return Distribution

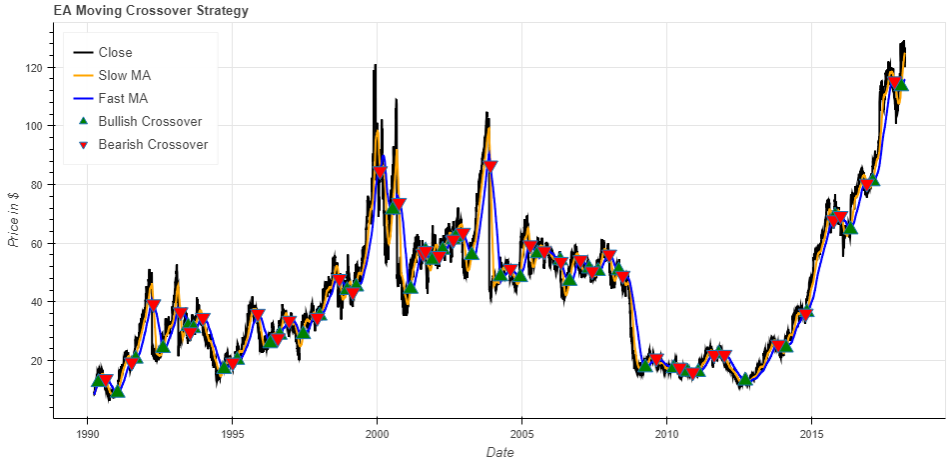


Volatility of Stock Returns

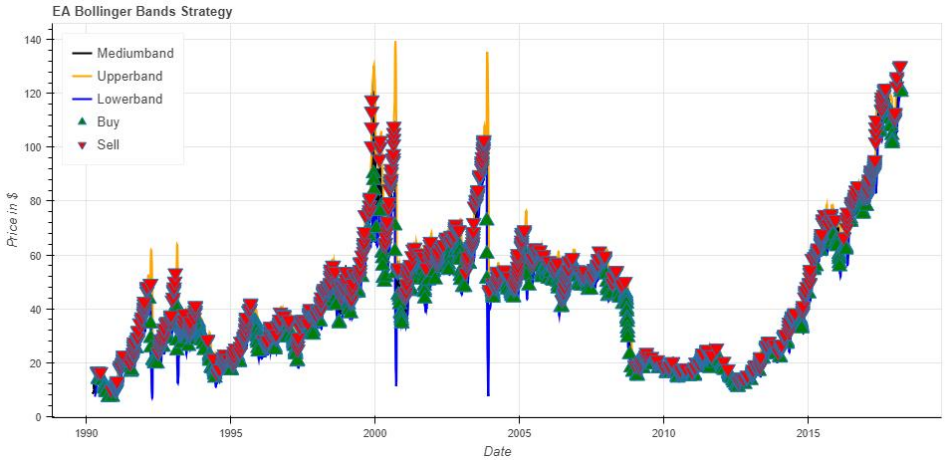


Backtesting Strategies

Moving Crossover Strategy



Bollinger Bands® Indicator



5.4.8 Raytheon Company

Getting Financial Data

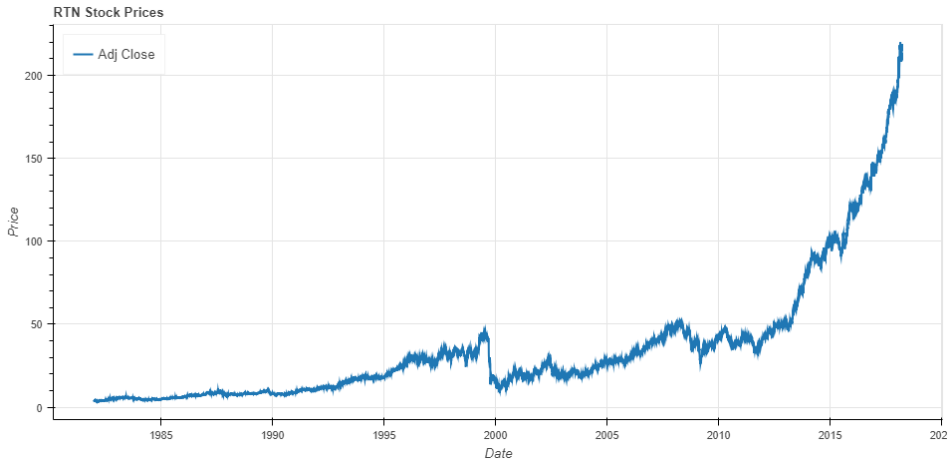
Requesting financial dataset using Quandl API:

```
rtn = quandl.get("WIKI/RTN ", start_date=start, end_date=end)
```

Descriptive Statistics

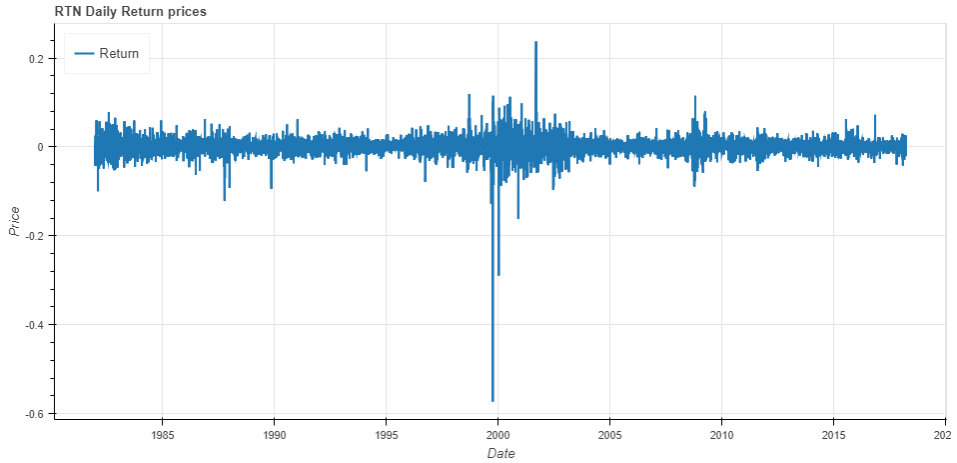
	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close
count	9137.000000	9137.000000	9137.000000	9137.000000	9.137000e+03	9137.000000	9137.000000	9137.000000	9137.000000	9137.000000	9137.000000
mean	61.666727	62.226335	61.118108	61.702997	1.226835e+06	0.005123	1.000219	35.311253	35.627331	35.002601	35.328632
std	31.223055	31.411586	31.073840	31.243109	1.273624e+06	0.049085	0.014794	38.359981	38.646793	38.106408	38.379354
min	17.620000	17.870000	17.500000	17.750000	6.000000e+03	0.000000	1.000000	3.266802	3.323615	3.209988	3.308844
25%	43.910000	44.380000	43.380000	43.940000	2.107000e+05	0.000000	1.000000	9.678775	9.758423	9.585844	9.678775
50%	54.050000	54.520000	53.520000	54.050000	8.735000e+05	0.000000	1.000000	24.210468	24.494546	23.997202	24.280308
75%	68.380000	68.880000	67.880000	68.380000	1.860600e+06	0.000000	1.000000	40.927658	41.262481	40.545025	40.911134
max	220.000000	222.820000	219.190000	219.860000	2.209300e+07	1.031250	2.000000	220.000000	222.820000	219.190000	219.860000

Data Preprocessing



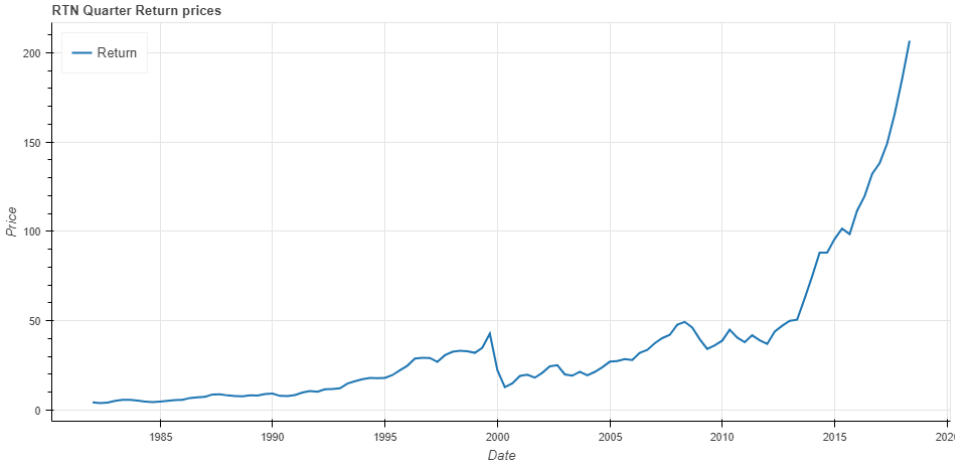
Return of Stocks

Daily Return



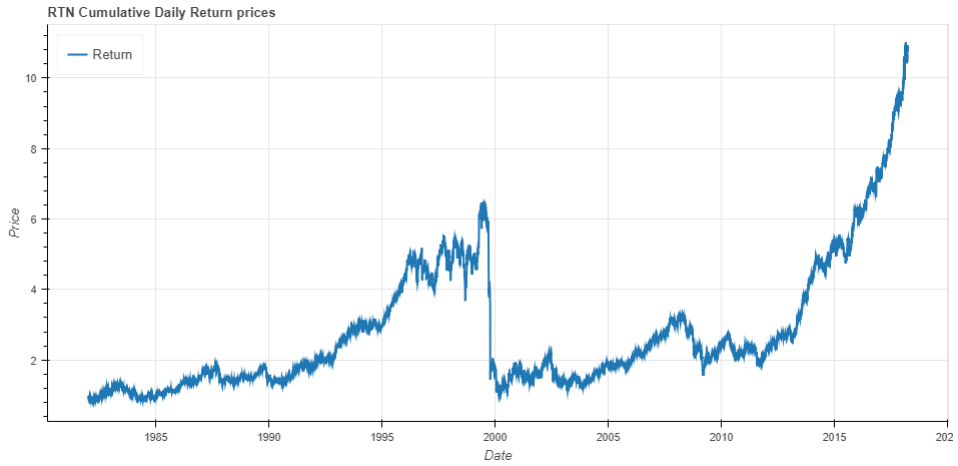
Monthly Return

Quarter Return

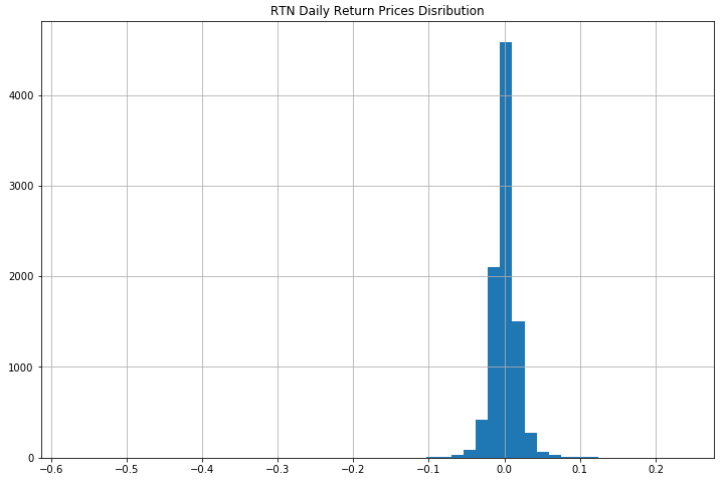


Cumulative Daily Return

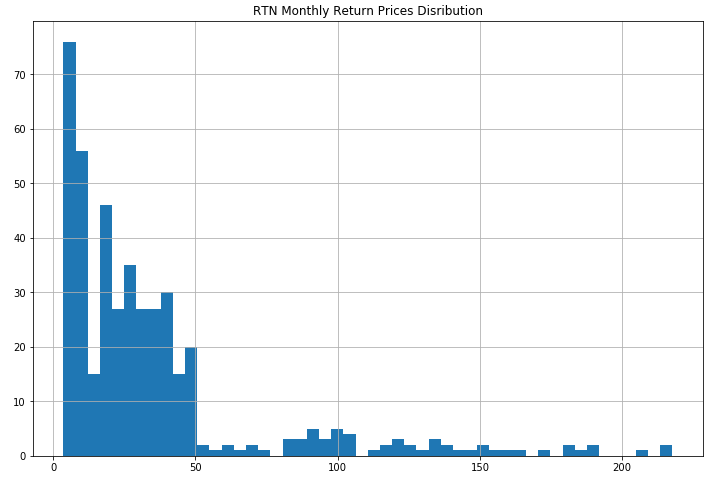
Cumulative Monthly Return



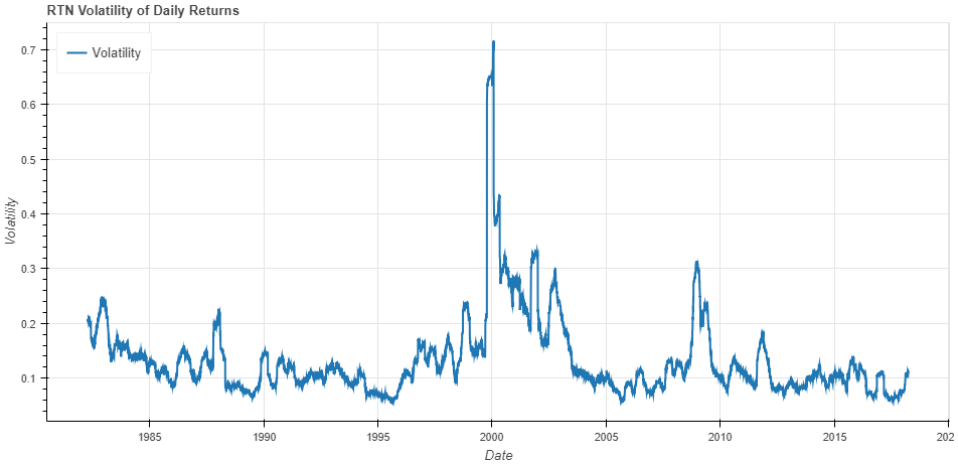
Daily Return Distribution



Monthly Return Distribution

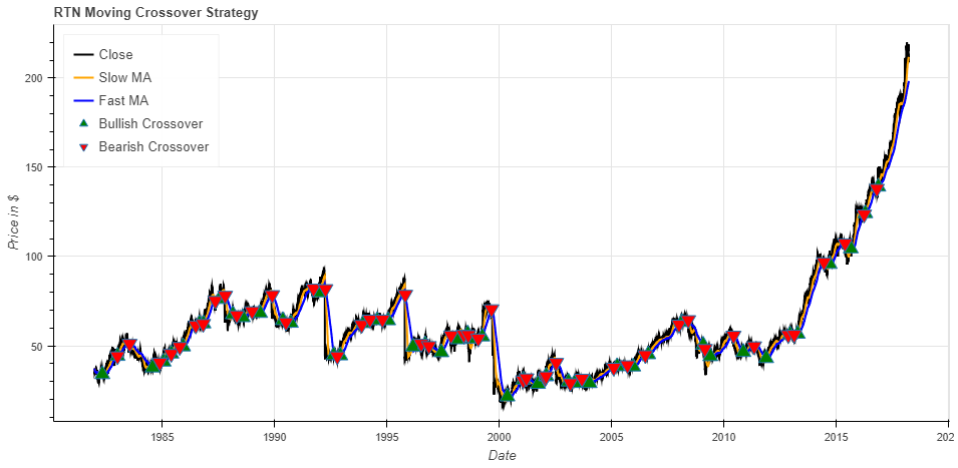


Volatility of Stock Returns

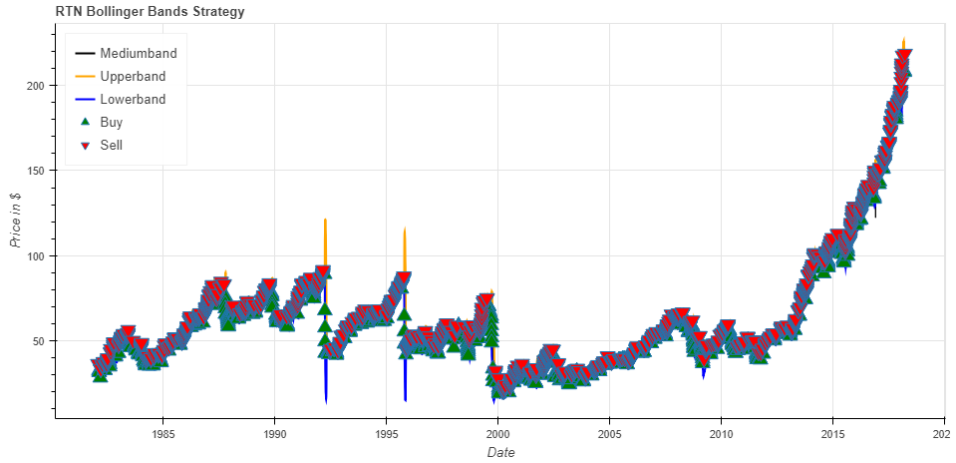


Backtesting Strategies

Moving Crossover Strategy



Bollinger Bands® Indicator



5.4.9 NVIDIA Corp.

Getting Financial Data

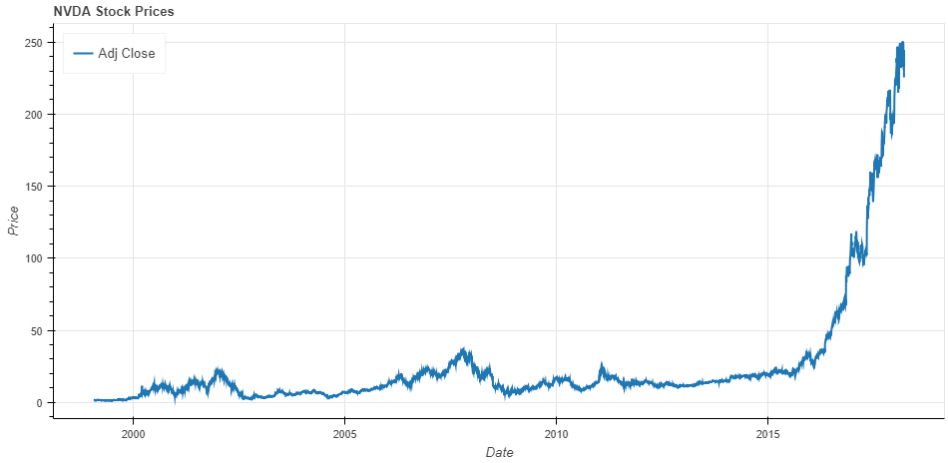
Requesting financial dataset using Quandl API:

```
nvda = quandl.get("WIKI/NVDA", start_date=start, end_date=end)
```

Descriptive Statistics

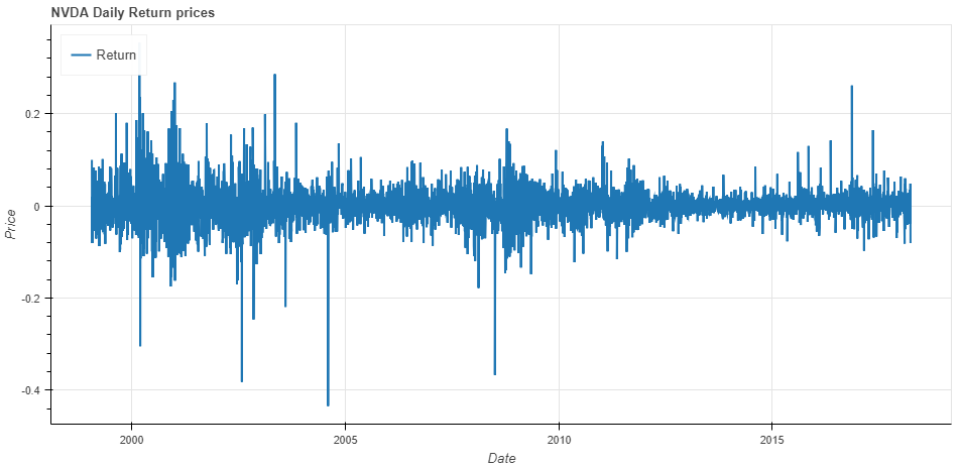
	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close
count	4825.000000	4825.000000	4825.000000	4825.000000	4.825000e+03	4825.000000	4825.000000	4825.000000	4825.000000	4825.000000	4825.000000
mean	36.968612	37.800557	36.112342	36.975141	1.073168e+07	0.000420	1.000725	24.616083	25.027754	24.166688	24.613528
std	41.166129	41.957559	40.329233	41.183512	8.721507e+06	0.006679	0.025946	40.785047	41.326064	40.118625	40.763092
min	6.000000	6.380000	5.750000	5.900000	4.100000e+04	0.000000	1.000000	1.295193	1.319164	1.237200	1.266583
25%	14.830000	15.030000	14.560000	14.810000	5.051700e+06	0.000000	1.000000	8.091285	8.332539	7.850031	8.091285
50%	21.560000	22.000000	21.070000	21.500000	8.821296e+06	0.000000	1.000000	13.185454	13.454545	12.888527	13.166896
75%	37.020000	37.670000	36.280000	37.000000	1.422530e+07	0.000000	1.000000	19.572497	19.841588	19.245117	19.544135
max	251.690000	254.500000	248.480000	250.480000	9.197937e+07	0.140000	2.000000	251.690000	254.500000	248.480000	250.480000

Data Preprocessing

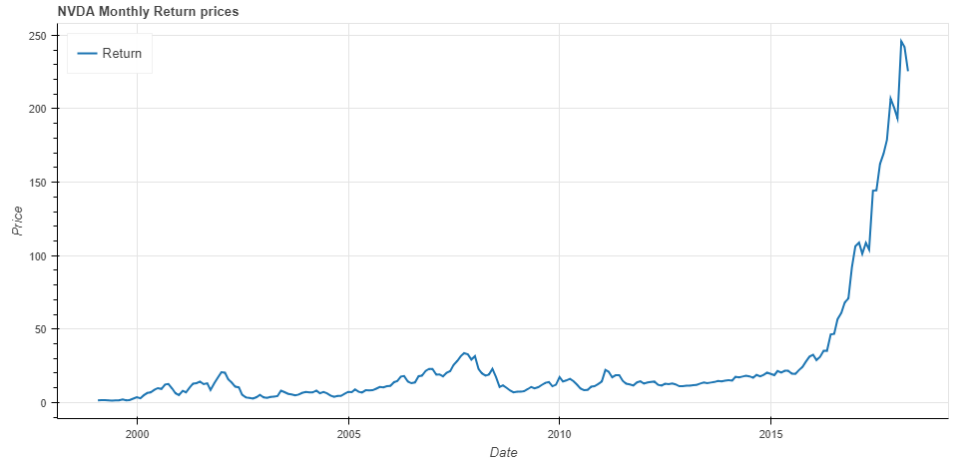


Return of Stocks

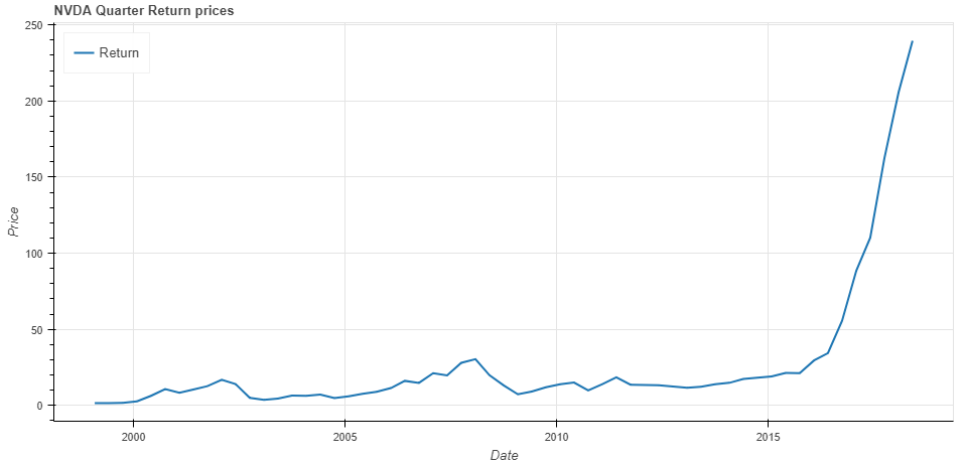
Daily Return



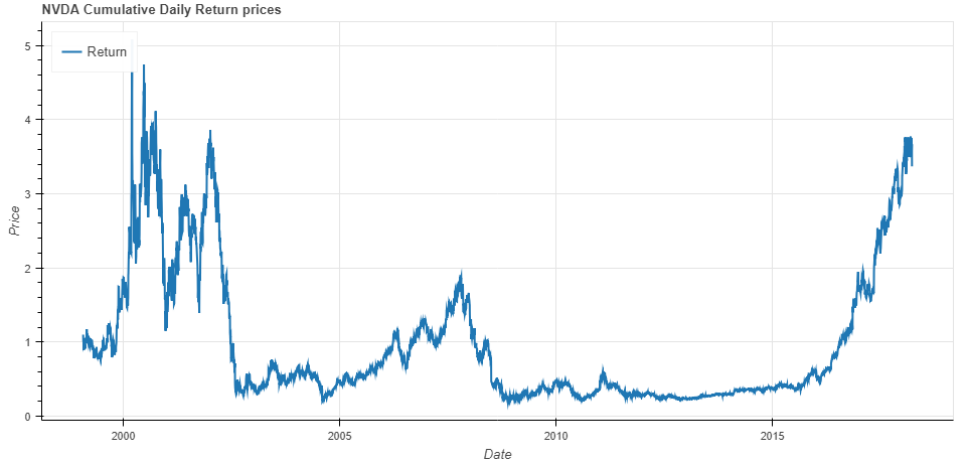
Monthly Return



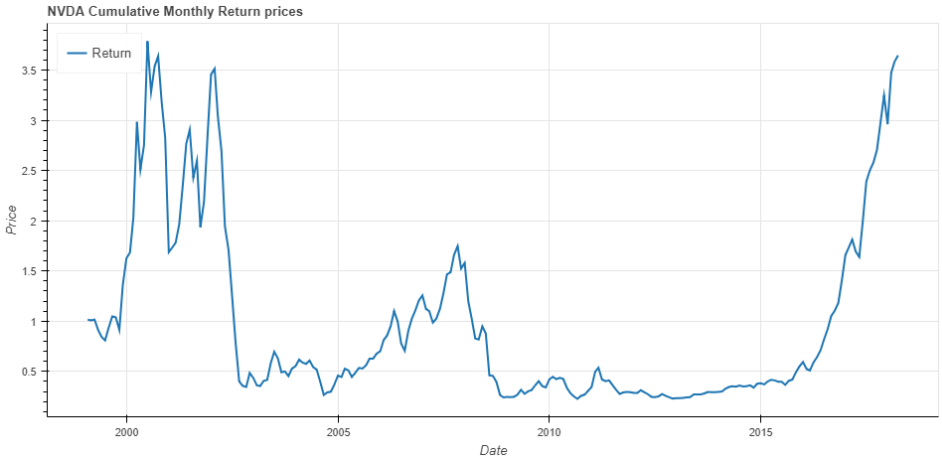
Quarter Return



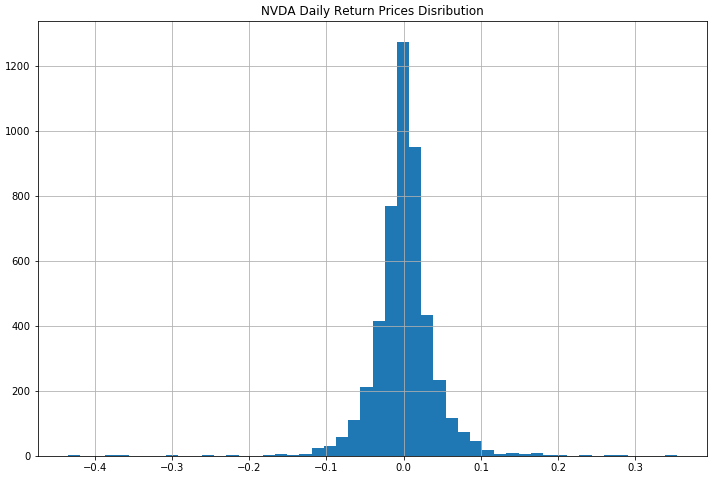
Cumulative Daily Return



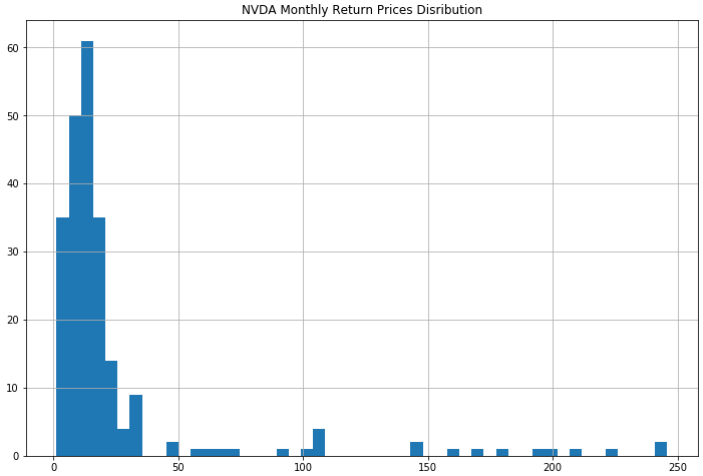
Cumulative Monthly Return



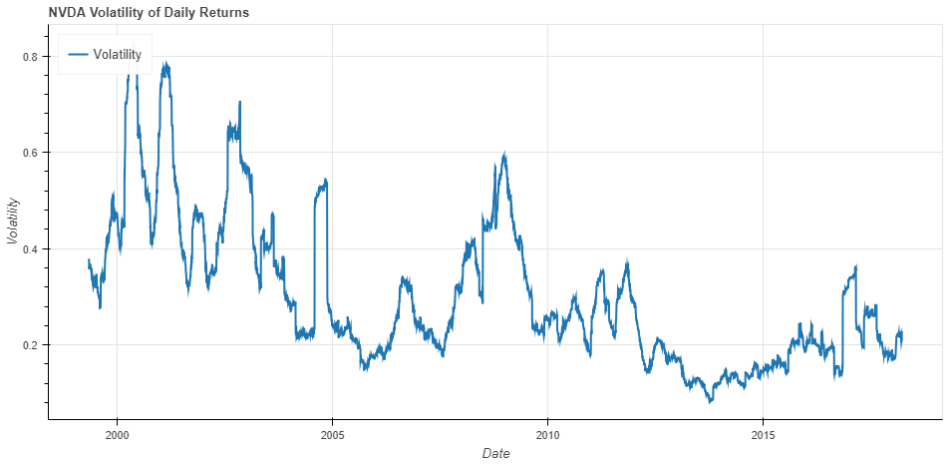
Daily Return Distribution



Monthly Return Distribution

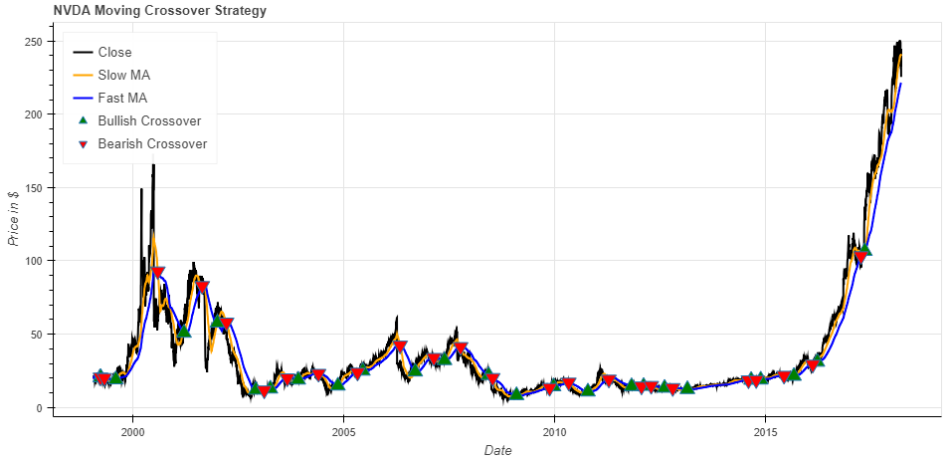


Volatility of Stock Returns

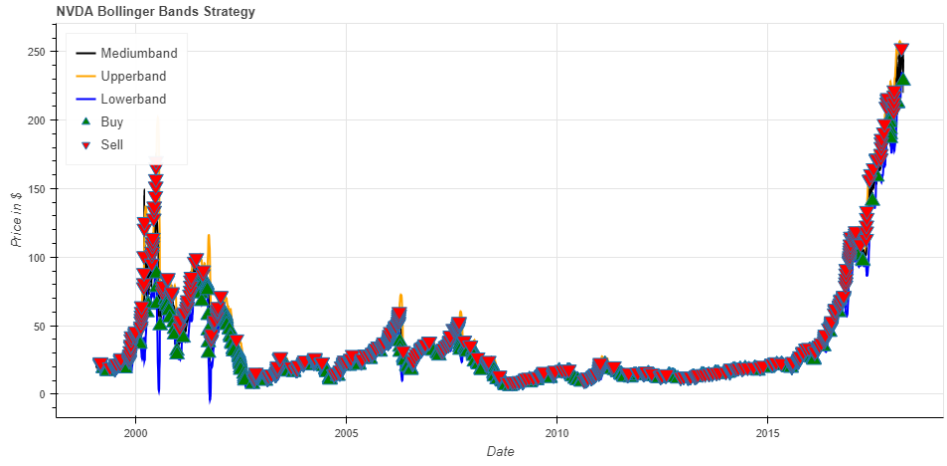


Backtesting Strategies

Moving Crossover Strategy



Bollinger Bands® Indicator



5.4.10 Intel Corp.

Getting Financial Data

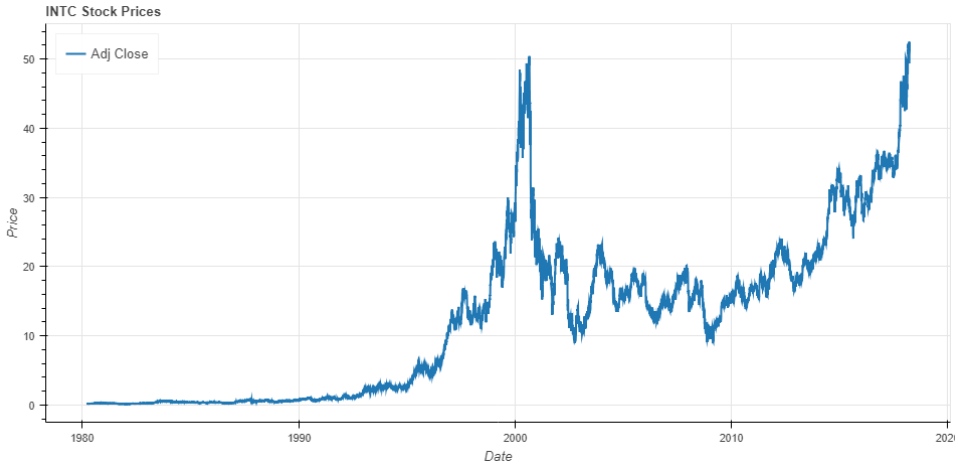
Requesting financial dataset using Quandl API:

```
intc = quandl.get("WIKI/INTC", start_date=start, end_date=end)
```

Descriptive Statistics

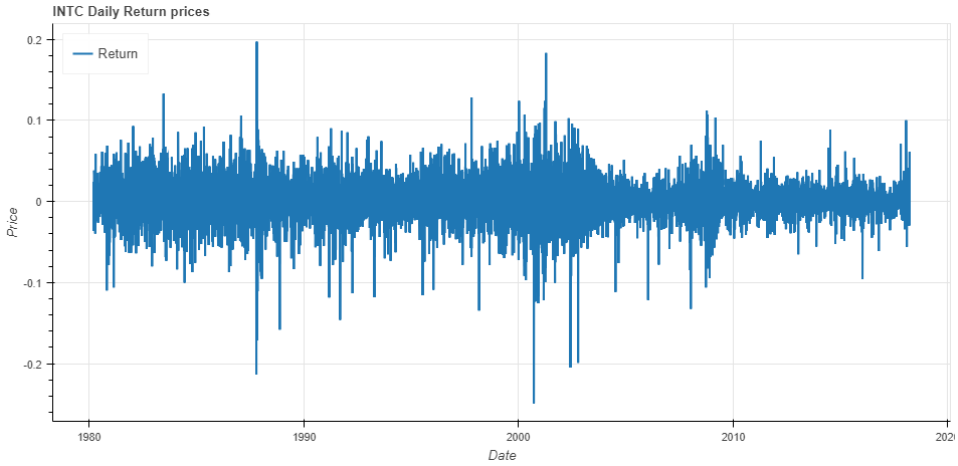
	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close
count	9589.000000	9589.000000	9589.000000	9589.000000	9.589000e+03	9589.000000	9589.000000	9589.000000	9589.000000	9589.000000	9589.000000
mean	42.435340	43.080088	41.801577	42.440523	2.678291e+07	0.001185	1.000782	12.265063	12.431286	12.098802	12.264278
std	27.822021	28.294908	27.372790	27.846844	2.983959e+07	0.014412	0.027487	11.634821	11.787411	11.487861	11.637280
min	12.150000	12.630000	12.050000	12.080000	0.000000e+00	0.000000	1.000000	0.144408	0.144408	0.142688	0.142688
25%	24.130000	24.490000	23.840000	24.130000	1.482400e+06	0.000000	1.000000	0.691093	0.701408	0.675621	0.691093
50%	32.000000	32.470000	31.500000	32.000000	1.693602e+07	0.000000	1.000000	12.584957	12.801037	12.376322	12.593451
75%	52.250000	53.000000	51.250000	52.190000	4.729670e+07	0.000000	1.000000	19.131677	19.375647	18.799004	19.088773
max	168.400000	169.800000	165.500000	169.400000	3.093476e+08	0.273000	2.000000	53.010000	53.780000	51.390000	52.480000

Data Preprocessing

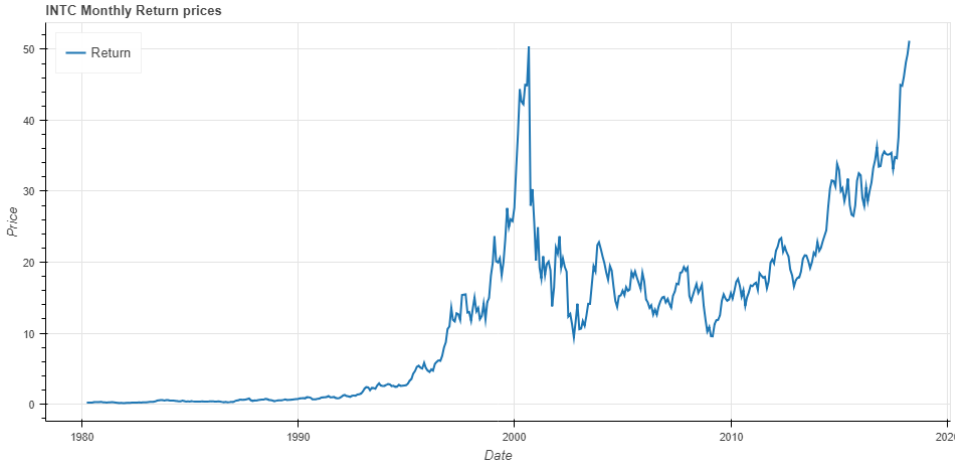


Return of Stocks

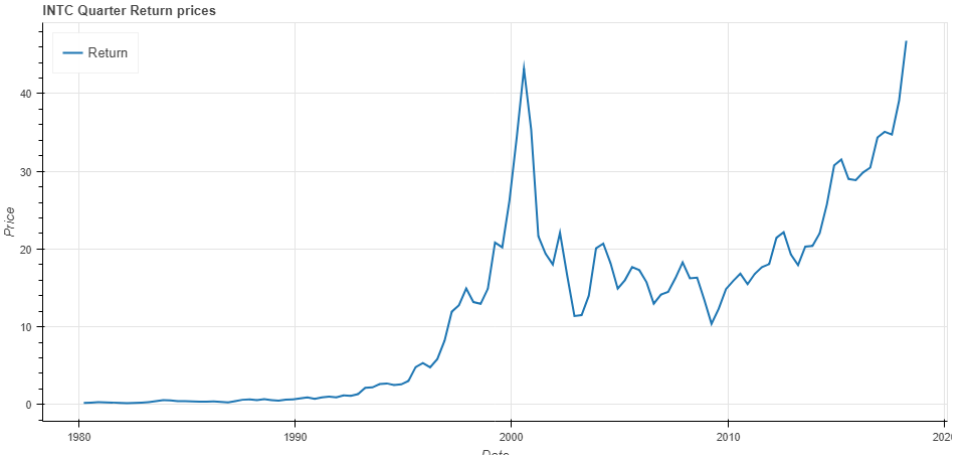
Daily Return



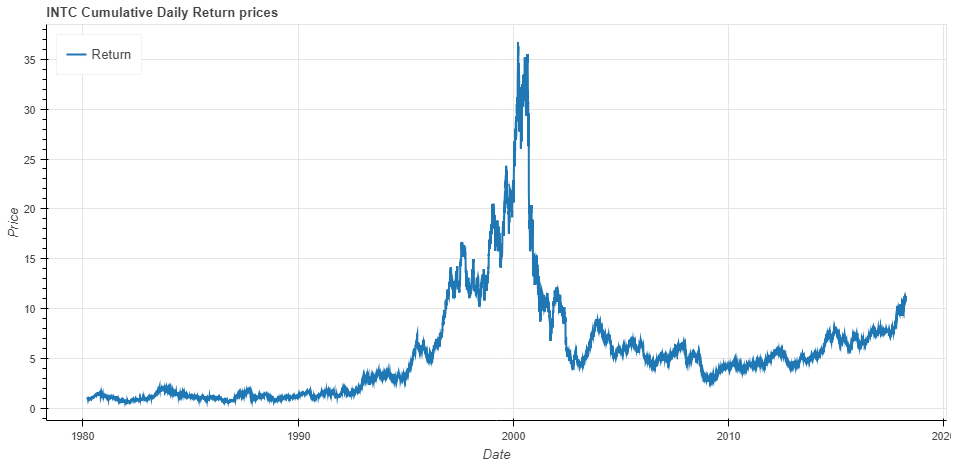
Monthly Return



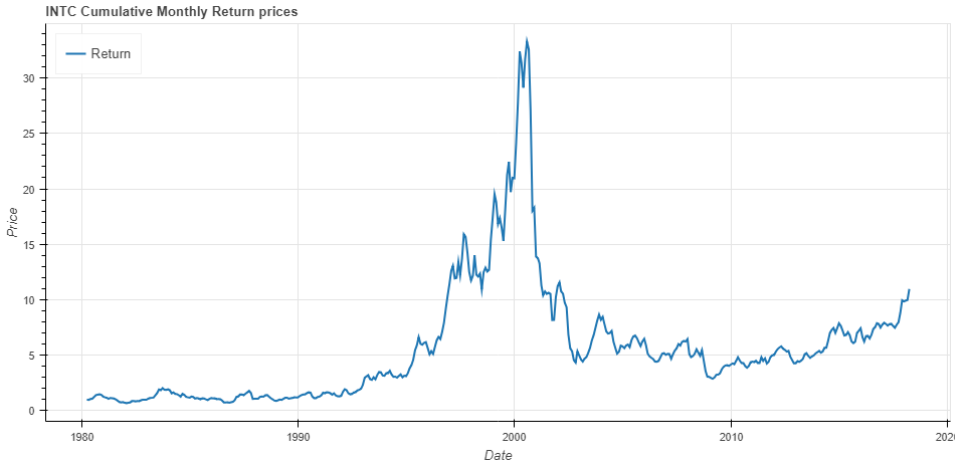
Quarter Return



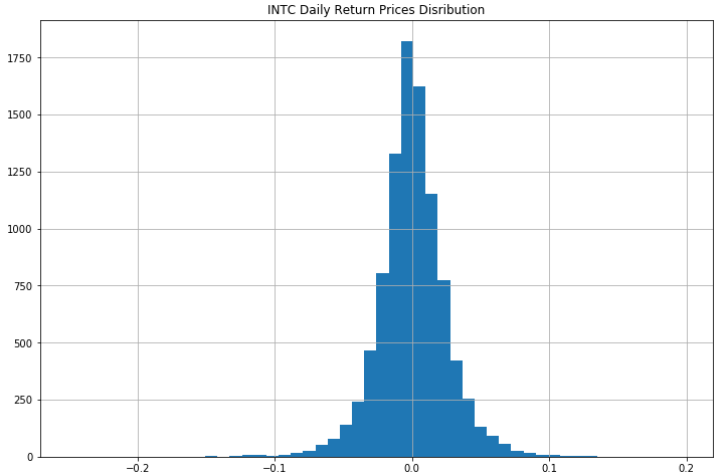
Cumulative Daily Return



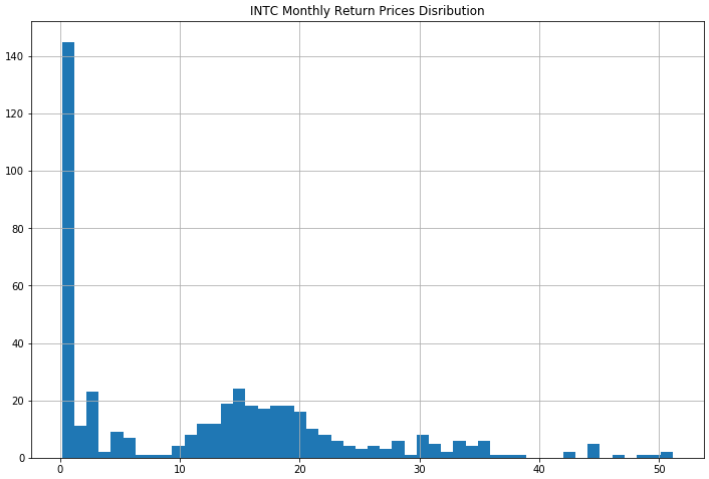
Cumulative Monthly Return



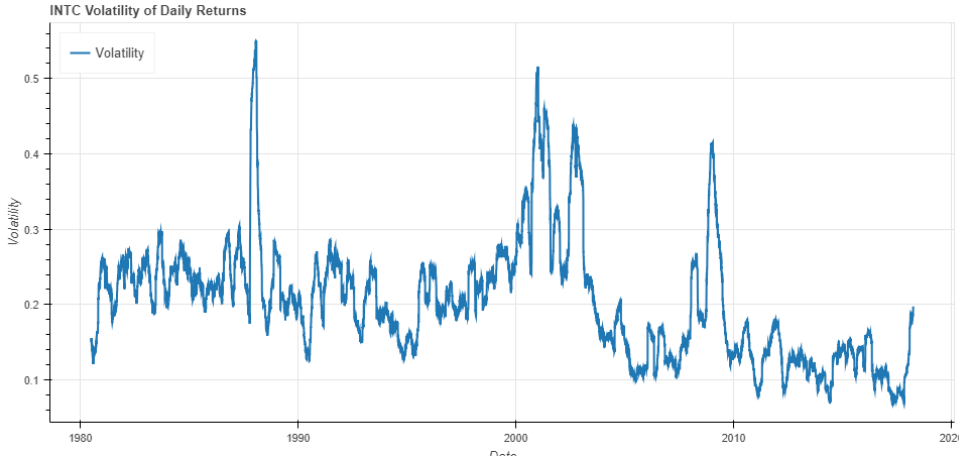
Daily Return Distribution



Monthly Return Distribution

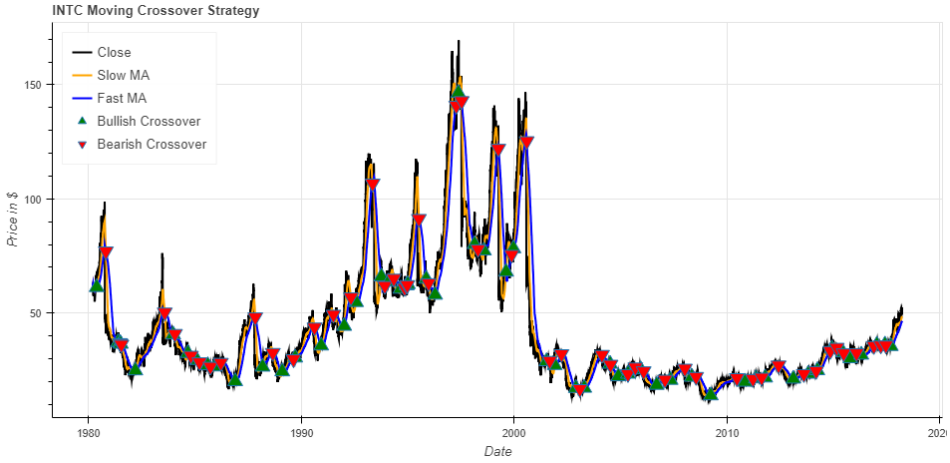


Volatility of Stock Returns

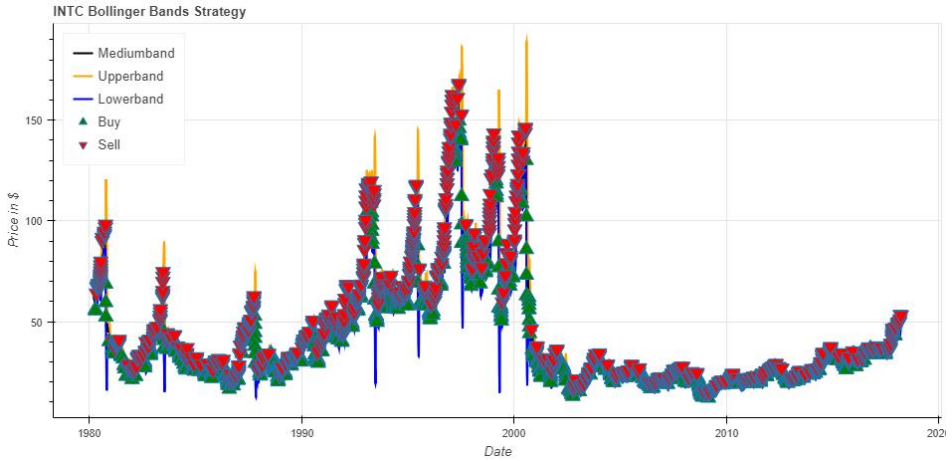


Backtesting Strategies

Moving Crossover Strategy



Bollinger Bands® Indicator



5.4.11 Advanced Micro Devices Inc.

Getting Financial Data

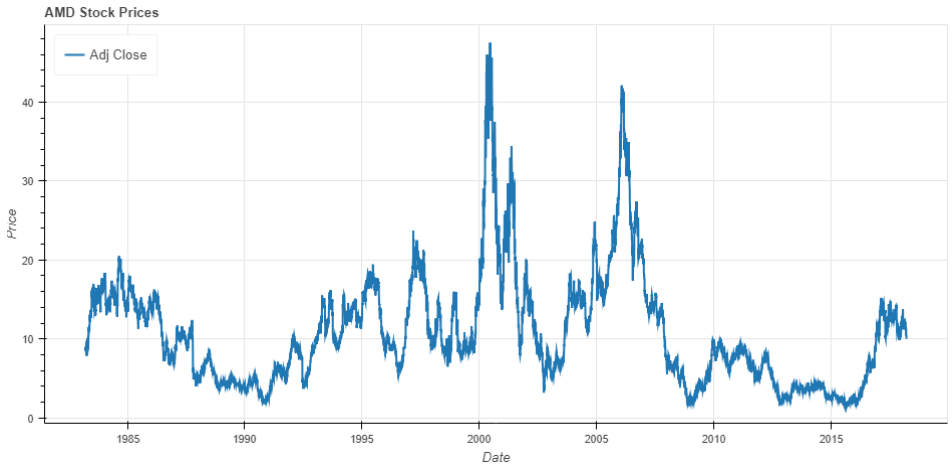
Requesting financial dataset using Quandl API:

```
amd = quandl.get("WIKI/AMD", start_date=start, end_date=end)
```

Descriptive Statistics

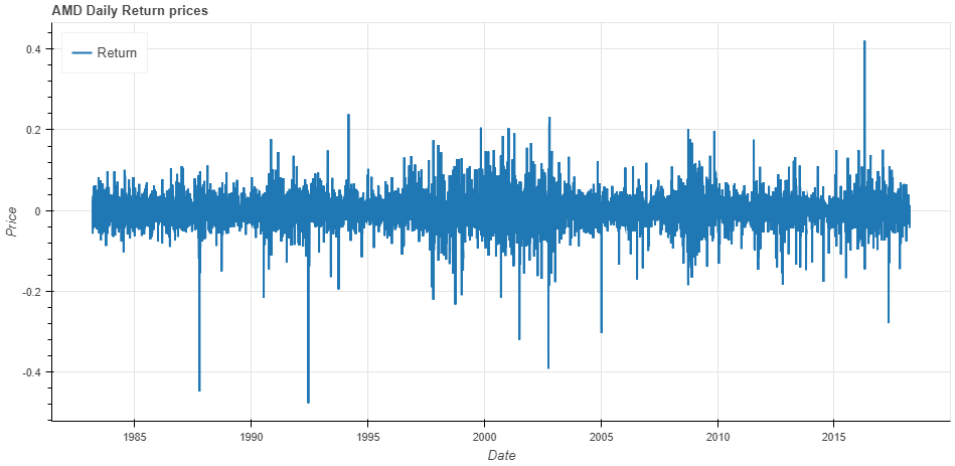
	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close
count	8829.000000	8829.000000	8829.000000	8829.000000	8.829000e+03	8829.000000	8829.000000	8829.000000	8829.000000	8829.000000	8829.000000
mean	16.732966	17.091615	16.341195	16.717803	1.150221e+07	0.000001	1.000227	10.847113	11.080770	10.590938	10.834230
std	12.678512	12.993987	12.347168	12.689244	1.801184e+07	0.000106	0.015050	7.152346	7.329946	6.964920	7.151443
min	1.620000	1.690000	1.610000	1.620000	8.000000e+02	0.000000	1.000000	1.620000	1.690000	1.610000	1.620000
25%	7.410000	7.580000	7.250000	7.400000	7.083000e+05	0.000000	1.000000	5.498494	5.623459	5.340000	5.498494
50%	14.160000	14.440000	13.880000	14.130000	4.503500e+06	0.000000	1.000000	9.290000	9.470000	9.040000	9.250000
75%	23.600000	24.000000	22.750000	23.490000	1.582410e+07	0.000000	1.000000	14.496028	14.750000	14.186113	14.496028
max	94.000000	97.000000	91.000000	95.000000	2.659830e+08	0.010000	2.000000	47.000000	48.500000	45.500000	47.500000

Data Preprocessing

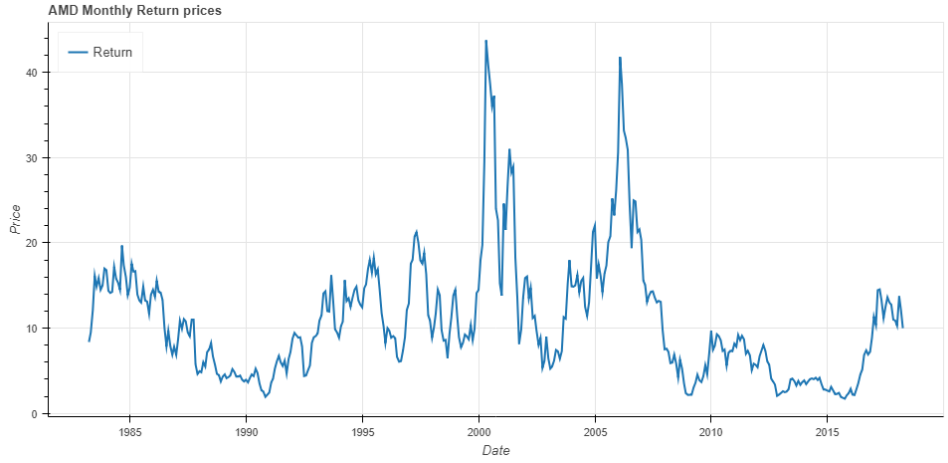


Return of Stocks

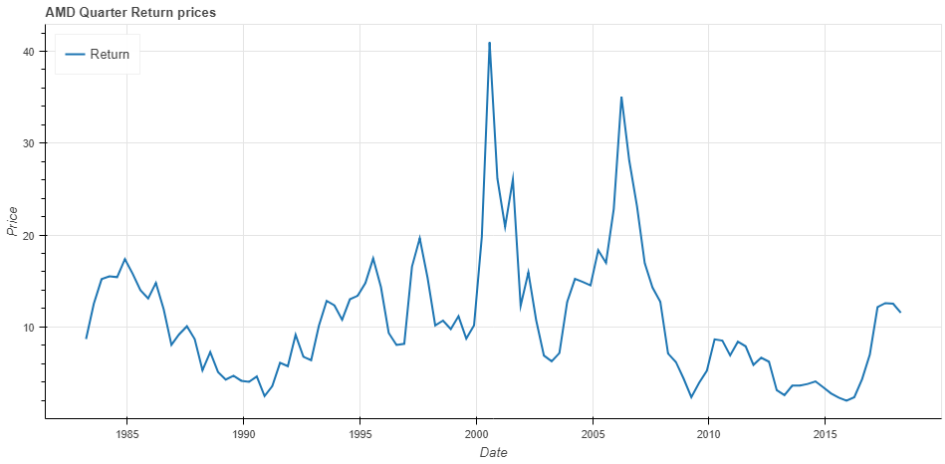
Daily Return



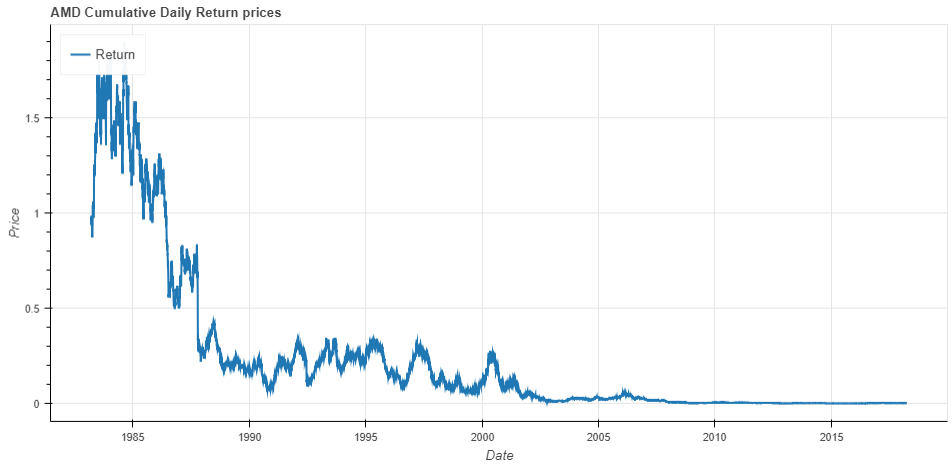
Monthly Return



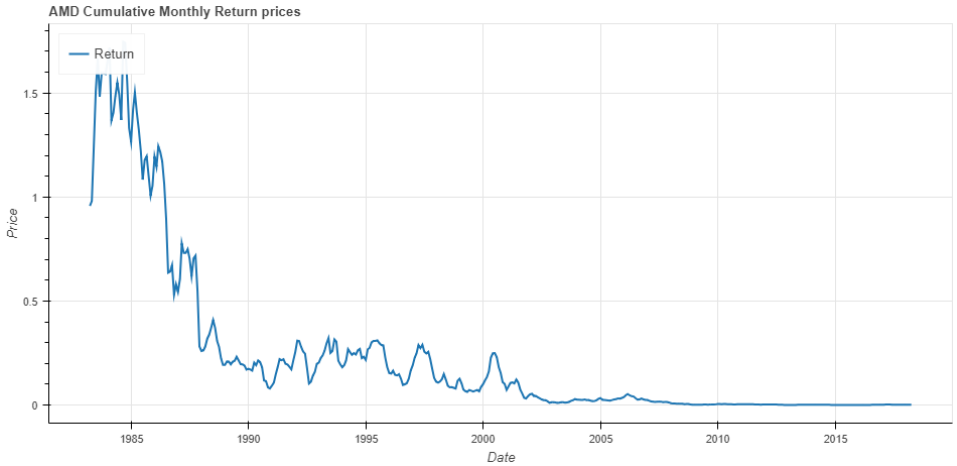
Quarter Return



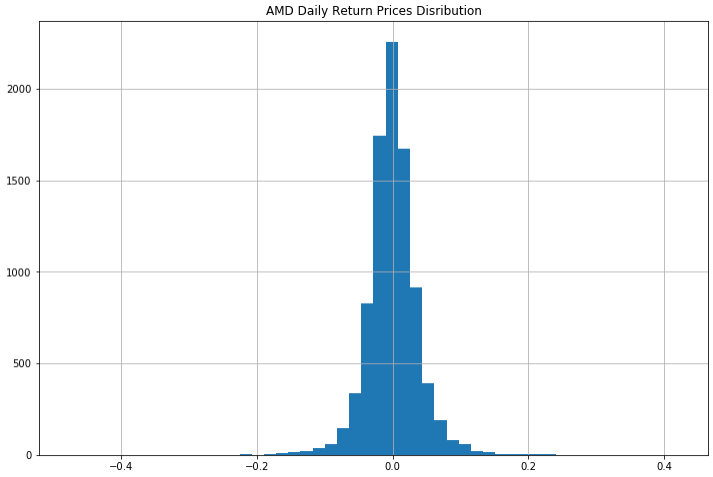
Cumulative Daily Return



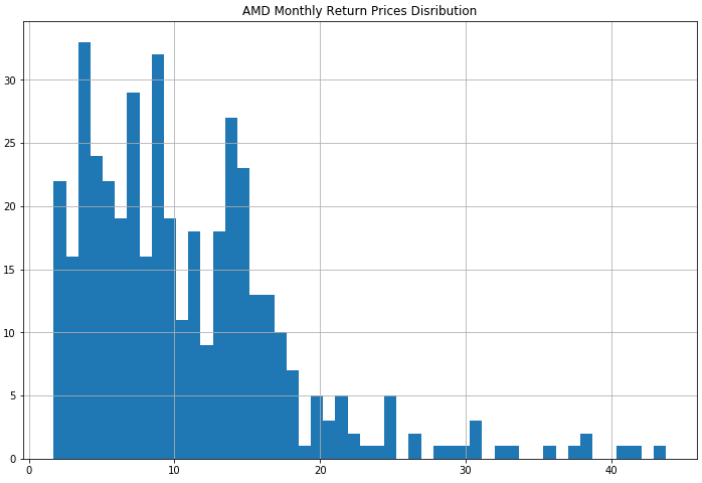
Cumulative Monthly Return



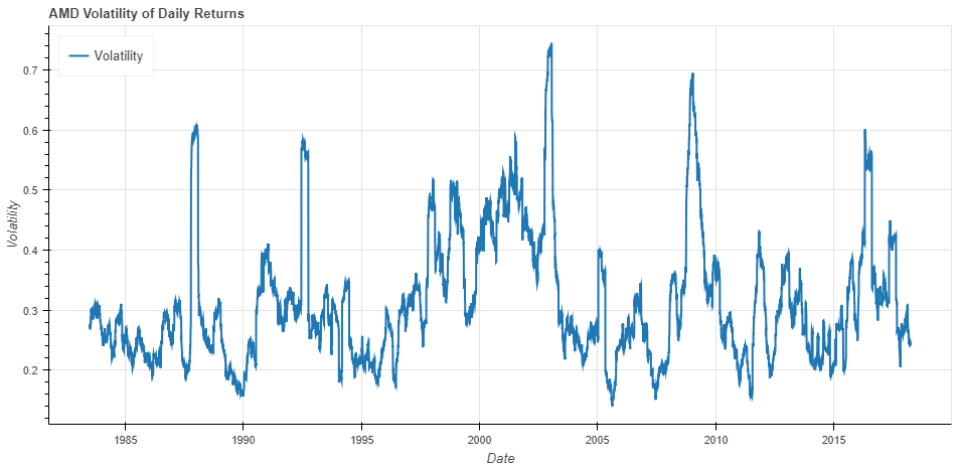
Daily Return Distribution



Monthly Return Distribution

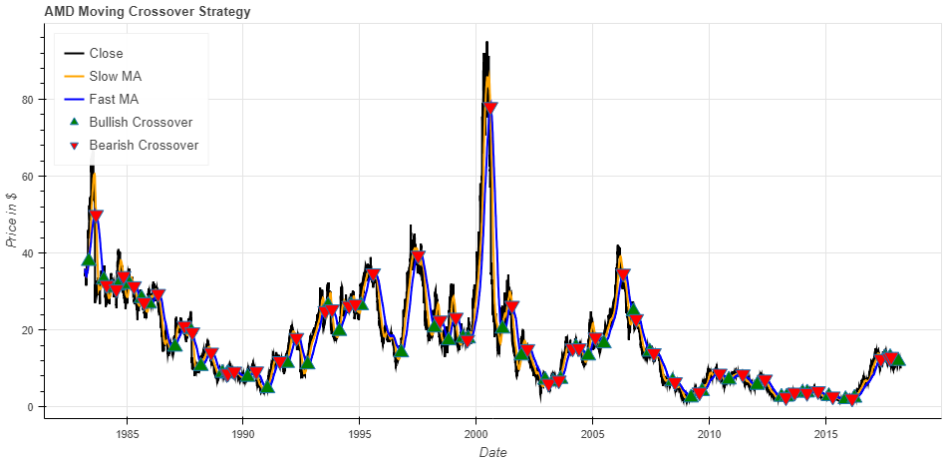


Volatility of Stock Returns

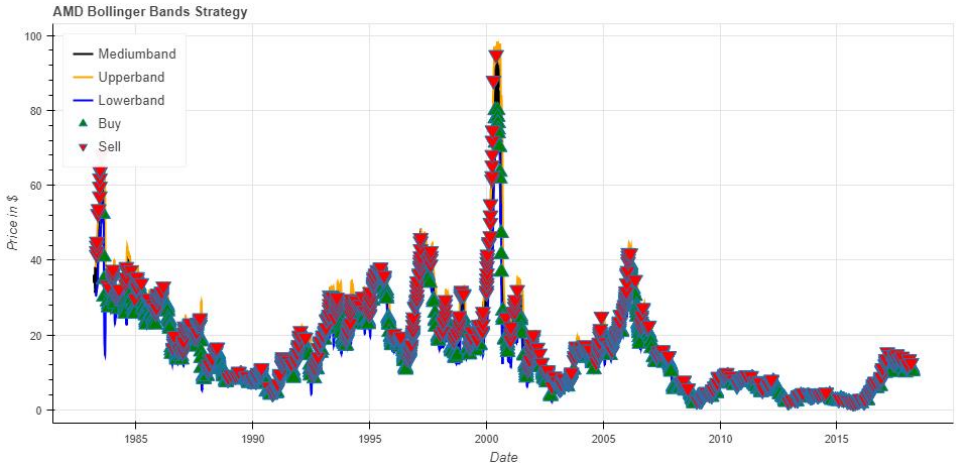


Backtesting Strategies

Moving Crossover Strategy



Bollinger Bands® Indicator



5.5 COMPARING BETWEEN DIFFERENT STOCKS

We will classify them in some way into two groups:

First Group:

- MSFT
- GOOG
- FB
- TWTR
- YHOO

- EA

Second Group:

- AAPL
- RTN
- INTC
- AMD
- NVDA

Defined functions for combining stocks and plotting histogram distribution:

```
def combine_stocks(data, tickers):
    # Concatenating dataframes for each ticker
    return pd.concat(data, keys=tickers, names=['Ticker', 'Date'])

def transform_stocks(data, tickers):
    df = combine_stocks(data,tickers)
    # Isolate the `Adj Close` values and transform the DataFrame
    return df[['Adj. Close']].reset_index().pivot('Date', 'Ticker', 'Adj. Close')

def hist_returns(all_data):
    # Calculate the daily percentage change for `daily_close_px`
    daily_pct_change = daily_return(all_data)
    # Plot the distributions
    daily_pct_change.hist(bins=50, sharex=True, figsize=(12,8))
    # Show the resulting plot
    plt.show()
```

Defined function for scatter matrix of return prices distribution:

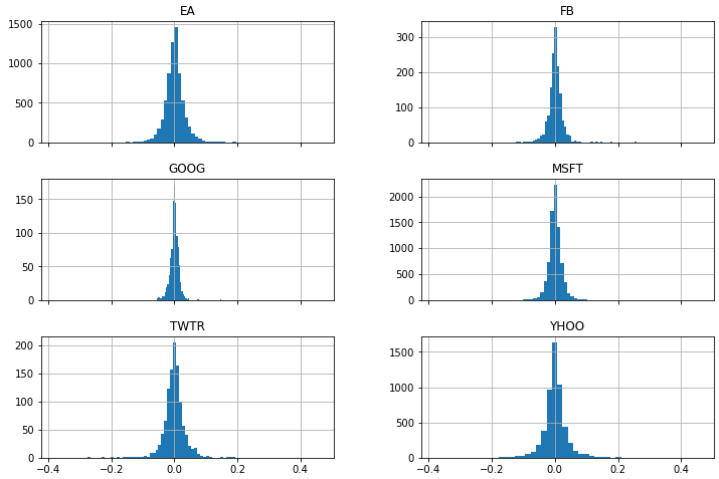
```
def scatter_returns(data):
    # Plot a scatter matrix with the `daily_pct_change` data
    pd.plotting.scatter_matrix(data, diagonal='kde', alpha=0.1,figsize=(12,12))
    # Show the plot
    plt.show()
```

5.5.1 Return Distribution

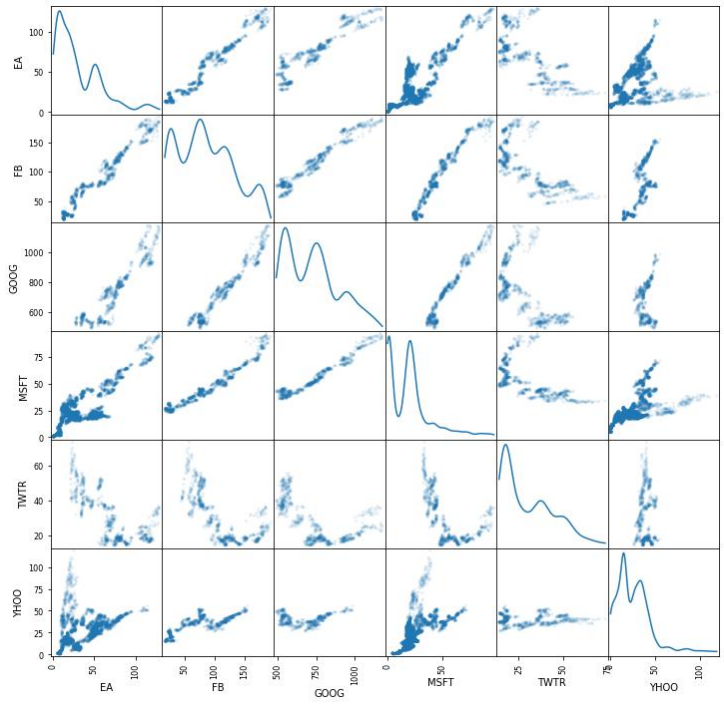
```
first_group = transform_stocks([msft,goog,fb,twtr,yhoo,ea],
['MSFT','GOOG','FB','TWTR','YHOO','EA'])
hist_returns(first_group)
```

First Group

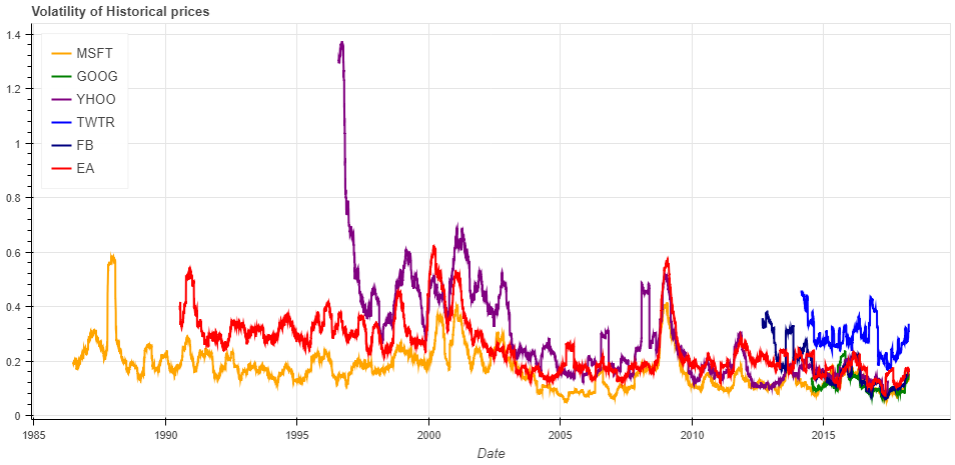
Histogram Distribution



Scatter Matrix

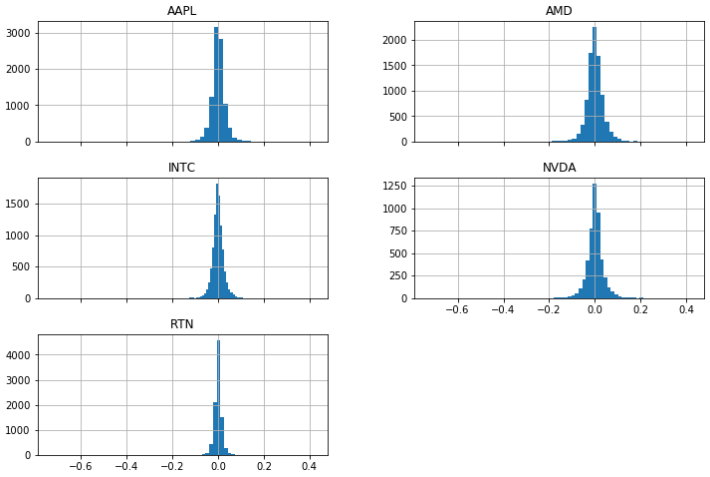


Volatility of Historical prices

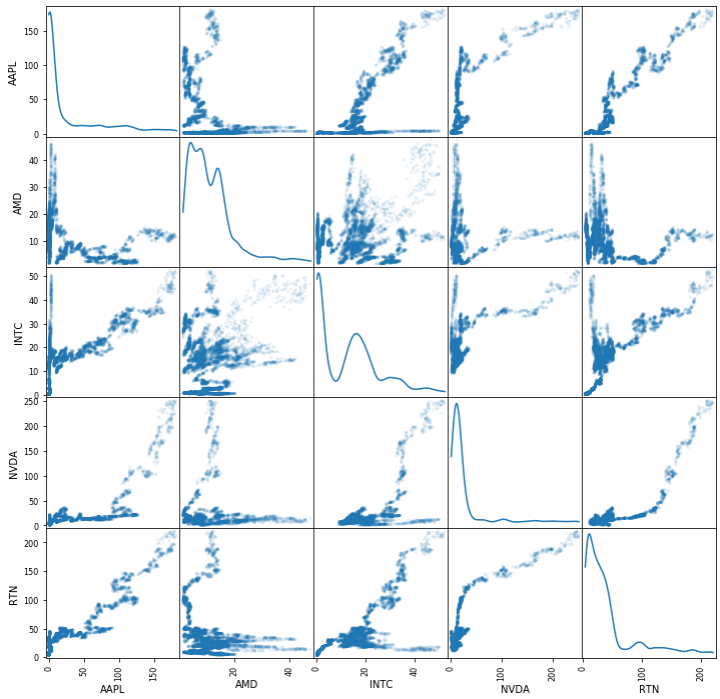


Second Group

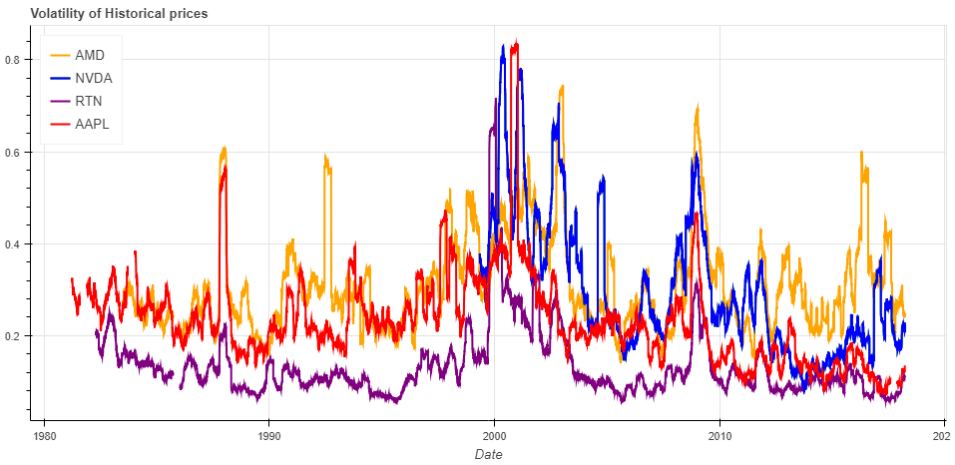
Histogram Distribution



Scatter Matrix

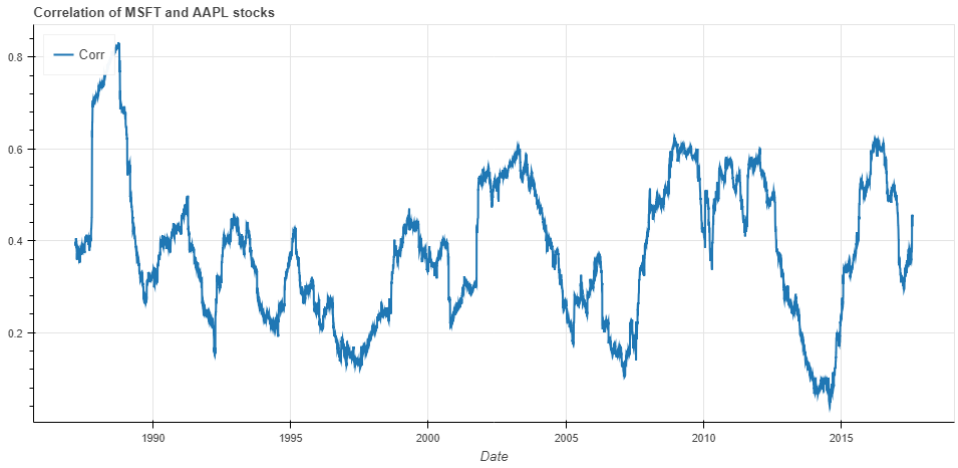


Volatility of Historical prices



5.5.2 Correlation Calculation

The rolling correlation of returns as a way to crosscheck your results:



5.6 STOCK MARKET PREDICTION

5.6.1 Introduction Machine Learning

In general, a machine learning problem considers a set of n samples of data and then tries to predict properties of unknown data. If each sample is more than a single number and, for instance, a multi-dimensional entry, it is said to have several attributes or **features**.

We can separate learning problems in a few large categories:

Supervised learning in which the data comes with additional attributes that we want to predict (Label). This problem can be either:

Classification: samples belong to two or more classes and we want to learn from already labeled data how to predict the class of unlabeled data. An example of classification problem would be the handwritten digit recognition example, in which the aim is to assign each input vector to one of a finite number of discrete categories. Another way to think of classification is as a discrete (as opposed to continuous) form of supervised learning where one has a limited number of categories and for each of the n samples provided, one is to try to label them with the correct category or class.

Regression: if the desired output consists of one or more continuous variables, then the task is called *regression*. An example of a regression problem would be the prediction of the length of a salmon as a function of its age and weight.

Unsupervised learning: in which the training data consists of a set of input vectors x without any corresponding target values. The goal in such problems may be to discover groups of similar

examples within the data, where it is called Clustering, or to determine the distribution of data within the input space.

Predicting the direction of stock prices is a widely studied subject in many fields including trading, finance, statistics, and computer science. Investors in the stock market can maximize their profit by buying or selling their investment if they can determine when to enter and exit a position.

There are many machine learning techniques to predict the stock market have been developed using the various existing computational techniques, such as LR (Linear Regression), RNN (Recurrent Neural Networks), KNN, and Naive Bayes.

K- Nearest Neighbor Classification

The K-Nearest Neighbor Algorithm is the simplest of all machine learning algorithms. It is based on the principle that the samples that are similar, generally lies in close vicinity. K-Nearest Neighbor is instance-based learning method. Instance based classifiers are also called lazy learners as they store all the training samples and do not build a classifier until a new, unlabeled sample needs to be classified. Lazy-learning algorithms require less computation time during the training phase than eager-learning algorithms (such as decision trees, neural networks, and Bayes networks) but more computation time during the classification process.

Nearest-neighbor classifiers are based on learning by resemblance, i.e. by comparing a given test sample with the available training samples which are like it. For a data sample X to be classified, its The K-NN classifier works as follows:

1. Initialize value of K.
2. Calculate distance between input sample and training samples.
3. Sort the distances.
4. Take top K- nearest neighbors.
5. Apply simple majority.
6. Predict class label with more neighbors for input sample.

Advantages:

- Easy to understand and implement.
- Training is very fast.
- It is robust to noisy training data.
- It performs well on applications in which a sample can have many class labels

Disadvantages:

- Lazy learners incur expensive computational costs when the number of potential neighbors which to compare a given unlabeled sample is large.
- It is sensitive to the local structure of the data [1]. Memory limitation. As it is supervised lazy learner, it runs slowly.

Naive Bayes Classification

Naive Bayes Classifier is the simple Statistical Bayesian Classifier. It is called Naive as it assumes that all variables contribute towards classification and are mutually correlated. This assumption is called class conditional independence. It is also called Idiot's Bayes, Simple Bayes, and Independence Bayes.

They can predict class membership probabilities, such as the probability that a given data item belongs to a class label. A Naive Bayes classifier considers that the presence (or absence) of a feature(attribute) of a class is unrelated to the presence (or absence) of any other feature when the class variable is given.

Advantages:

- It requires short computational time for training.
- It improves the classification performance by removing the irrelevant features.
- It has good performance.

Disadvantages:

- The Naive Bayes classifier requires a very large number of records to obtain good results.
- Less accurate as compared to other classifiers on some datasets.

Decision Tree Induction

Decision tree learning uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value. Decision tree algorithm is a data mining induction technique that recursively partitions a dataset of records using depth-first greedy approach or breadth-first approach until all the data items belong to a particular class.

A decision tree structure is made of root, internal and leaf nodes. It is a flow chart like tree structure, where every internal node denotes a test condition on an attribute, each branch represents result of the test condition, and each leaf node (or terminal node) is assigned with a class label. The topmost node is the root node. Decision tree is constructed in a divide and conquer approach. Each path in decision tree forms a decision rule. Generally, it utilizes greedy approach from top to bottom.

Decision tree classification technique is performed in two phases:

- Tree building: it is performed in top-down approach. During this phase, the tree is recursively partitioned till all the data items belong to the same class label. It is very computationally intensive as the training dataset is traversed repeatedly.
- Tree pruning: it is done in a bottom-up manner. It is used to improve the prediction and classification accuracy of the algorithm by minimizing over-fitting problem of tree.

Advantages:

- Decision Trees are very simple and fast.
- It produces the accurate result.
- Representation is easy to understand i.e. comprehensible.
- It supports incremental learning.
- It takes the less memory.

- It can also deal with noisy data.
- It uses different measures such as Entropy, Gini index, Information gain etc.to find best split attribute.

Disadvantages:

- It has long training time.
- Decision trees can have significantly more complex representation for some concepts due to replication problem.
- It has a problem of over fitting

.

RNN (Recurrent Neural Network)

The idea behind RNNs is to make use of sequential information. In a traditional neural network, we assume that all inputs (and outputs) are independent of each other. But for many tasks that's a very bad idea. If you want to predict the next word in a sentence you better know which words came before it. RNNs are called *recurrent* because they perform the same task for every element of a sequence, with the output being depended on the previous computations. Another way to think about RNNs is that they have a "memory" which captures information about what has been calculated so far. In theory RNNs can make use of information in arbitrarily long sequences, but in practice they are limited to looking back only a few steps.

RNN have shown great success in many NLP tasks. The most commonly used type of RNN are LSTM, which are much better at capturing long-term dependencies than vanilla RNN are

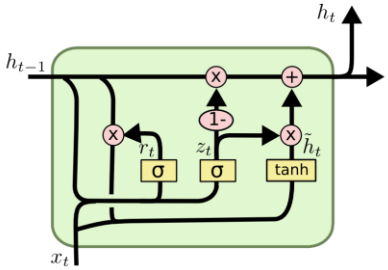
Long Short-Term Memory (LSTM)

Units (or blocks) are a building unit for layers of a recurrent neural network (RNN). Consider trying to predict the last word in the text "I grew up in France… I speak fluent *French*." Recent information suggests that the next word is probably the name of a language, but if we want to narrow down which language, we need the context of France, from further back. It's entirely possible for the gap between the relevant information and the point where it is needed to become very large. A RNN composed of LSTM units is often called an LSTM network. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell is responsible for "remembering" values over arbitrary

time intervals; hence the word "memory" in LSTM. Each of the three gates can be thought of as a "conventional" artificial neuron, as in a multi-layer (or feedforward) neural network: that is, they compute an activation (using an activation function) of a weighted sum. Intuitively, they can be thought as regulators of the flow of values that goes through the connections of the LSTM; hence the denotation "gate". There are connections between these gates and the cell.

The expression long short-term refers to the fact that LSTM is a model for the short-term-memory which can last for a long period of time.

An LSTM is well-suited to classify, process and predict time series given time lags of unknown size and duration between important events. LSTMs were developed to deal with the exploding and vanishing gradient problem when training traditional RNNs.



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Architecture:

There are several architectures of LSTM units. A common architecture is composed of a memory cell, an input gate, an output gate and a forget gate.

An LSTM (memory) cell stores a value (or state), for either long or short time periods. This is achieved by using an identity (or no) activation function for the memory cell. In this way, when an LSTM network (that is an RNN composed of LSTM units) is trained with backpropagation through time, the gradient does not tend to vanish.

The LSTM gates compute an activation, often using the logistic function. Intuitively, the input gate controls the extent to which a new value flows into the cell, the forget gate controls the extent to which a value remains in the cell and the output gate controls the extent to which the value in the cell is used to compute the output activation of the LSTM unit.

There are connections into and out of these gates. A few connections are recurrent. The weights of these connections, which need to be learned during training, of an LSTM unit are used to direct the

operation of the gates. Each of the gates has its own parameters, that is weights and biases, from possibly other units outside the LSTM unit.

Applications: Semantic parsing, Grammar learning, Time series anomaly detection

5.6.2 Implementation

We will use scikitLearn a machine learning library in python which has simple and efficient tools for data mining and data analysis, accessible to everybody, reusable in various contexts, Built on NumPy, SciPy, matplotlib, open source, Commercially usable – BSD license

KNN model

First step we must import the necessary libraries:

```
import numpy as np
import pandas as pd
import psycopg2
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import datetime
from sklearn.neighbors import KNeighborsClassifier
from sklearn.cross_validation import train_test_split
from sklearn.grid_search import GridSearchCV, RandomizedSearchCV
from sklearn import preprocessing
from sklearn import utils
from sklearn.metrics import scorer
from sklearn.metrics import accuracy_score
```

Second step connect to the database and read data

```
conn_string = "host='localhost' dbname='stockmarket' user='postgres' password='5265104d'"
conn = psycopg2.connect(conn_string)
```

```
df = pd.read_sql_query(
    '''select i.date ,i.open ,i.high ,i.low ,i.close ,i.adj_close ,i.volume ,c.name
    from dataset_information i
    join dataset_company c
    on i.company_id=c.id
    where company_id = {}'''.format(3)  , conn, index_col='date')
```

The output data frame will be like this

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

```
                open      high       low     close  adj_close      volume  \
date
1986-03-12  0.444196  0.448661  0.441964  0.441964   0.020108  21420000.0
1986-03-13  0.441964  0.446429  0.435268  0.441964   0.020108  28991200.0
1986-03-14  0.441964  0.468750  0.441964  0.466518   0.021225  96213600.0
1986-03-17  0.464286  0.464286  0.453125  0.464286   0.021123  29680000.0
1986-03-18  0.464286  0.486607  0.462054  0.479911   0.021834  62339200.0

            name
date
1986-03-12  AAPL
1986-03-13  AAPL
1986-03-14  AAPL
1986-03-17  AAPL
1986-03-18  AAPL
```

Fourth step calculating Bollinger band

```
df["20d"] = np.round(df["close"].rolling(window = 20, center = False).mean(), 2)
# 2. Compute rolling standard deviation
apple_rstd = np.round(df['close'].rolling(window = 20, center = False).std(), 2)

# 3. Compute upper and lower bands
df['upperband'] = df['20d'] + 2 * apple_rstd
df['lowerband'] = df['20d'] - 2 * apple_rstd
```

Fifth step preparing label column of the Bollinger band

```
df['high-upperband'] = df['high']-df['upperband']
df['low-lowerband'] =df['low']-df['lowerband']
df['decision'] = np.where(df['high-upperband'] > 0.0 , 'Sell',np.where( df['low-lowerband'] < 0.0,'Buy','Buy & Sell'))
print(df)
```

The output data frame will be like

```
print(df.tail())
```

```
                 open        high         low       close   adj_close  \
date
2018-03-07  174.940002  175.850006  174.270004  175.029999  175.029999
2018-03-08  175.479996  177.119995  175.070007  176.940002  176.940002
2018-03-09  177.960007  180.000000  177.389999  179.979996  179.979996
2018-03-12  180.289993  182.389999  180.210007  181.720001  181.720001
2018-03-13  182.589996  183.500000  179.240005  179.970001  179.970001

               volume  name     20d  upperband  lowerband  high-upperband  \
date
2018-03-07  31703500.0  AAPL  170.85     185.69     156.01       -9.839994
2018-03-08  23774100.0  AAPL  171.72     185.78     157.66       -8.660005
2018-03-09  32185200.0  AAPL  172.97     185.13     160.81       -5.130000
2018-03-12  32207100.0  AAPL  174.23     184.19     164.27       -1.800001
2018-03-13  31244000.0  AAPL  175.09     183.77     166.41       -0.270000

            low-lowerband    decision
date
2018-03-07      18.260004  Buy & Sell
2018-03-08      17.410007  Buy & Sell
2018-03-09      16.579999  Buy & Sell
2018-03-12      15.940007  Buy & Sell
2018-03-13      12.830005  Buy & Sell
```

Sixth step preparing KNN model

```
feature_cols = ['close','20d','upperband','lowerband']
label_cols = ['decision']
x = df[feature_cols]
y = df[label_cols]
```

The used method to extract features is

```
def extractBolingerFeatures(df):
    feature_cols = ['adj_close','20d','upperband','lowerband']
    label_cols = ['decision']
    x = df[feature_cols]
    y = df[label_cols]
    npY = np.array(y)
    return (x,y,npY)
```

```
metric_options =['minkowski','euclidean','manhattan','cityblock']
knn = KNeighborsClassifier(n_neighbors=3,metric='euclidean')
```

Seventh step split the data set into test and train parts to test our model accuracy (80% train, 20% test)

```
X_train, X_test, y_train, y_test = train_test_split(x,y, test_size=0.2, random_state=1)
```

Eights step train machine with the training data and predict the test data

```
knn.fit(X_train,y_train)
y_predict = knn.predict(X_test)
```

Last step calculate accuracy

```
y_predict = knn.predict(X_test)
print(float(accuracy_score(y_test, y_predict)) *100)
```

```
86.27329192546583
```

The used methods are

```
def ApplyKnnModel(fetures,label,predictList):
    knn = KNeighborsClassifier(n_neighbors=3,metric='euclidean')
    knn.fit(fetures,label)
    y_predict = str(knn.predict(predictList))
    return y_predict
```

Decision Tree C4.5

First step imports the Decision Tree module

```
from sklearn.tree import DecisionTreeClassifier
```

Second step preparing Decision Tree model

```
dtc = DecisionTreeClassifier(criterion='entropy')
```

Third step train machine with the training data and predict the test data

```
dtc.fit(X_train,y_train)
y_predict = dtc.predict(X_test)
print(float(accuracy_score(y_test, y_predict)) *100)
```

```
83.29192546583852
```

At last the used method is

```
def ApplyDecisionTreeModel(fetures,label,predictList):
    dtc = DecisionTreeClassifier(criterion='entropy')
    dtc.fit(fetures,label)
    y_predict = str(dtc.predict(predictList))
    return y_predict
```

LSTM (Long Short-Term Memory) Neural Network

Running the LSTM

Here you will train and predict stock price movements for several epochs and see whether the predictions get better or worse over time. You follow the following procedure.

Define a test set of starting points (test_points_seq) on the time series to evaluate the model on

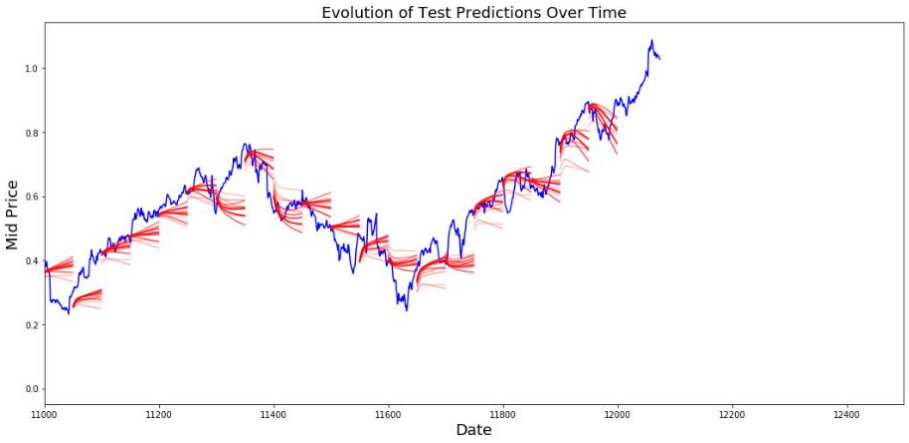
- For each epoch
- For full sequence length of training data
- Unroll a set of num_unrollings batches
- Train the neural network with the unrolled batches
- Calculate the average training loss

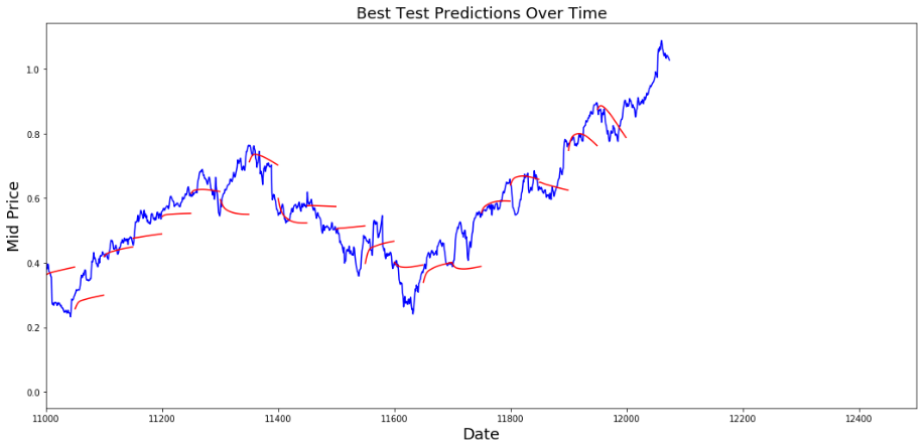
- For each starting point in the test set
    - Update the LSTM state by iterating through the previous `num_unrollings` data points found before the test point
    - Make predictions for `n_predict_once` steps continuously, using the previous prediction as the current input
    - Calculate the MSE loss between the `n_predict_once` points predicted and the true stock prices at those time stamps

```
Initialized
Average loss at step 1: 1.703350
    Test MSE: 0.00318
    Finished Predictions
  ...
  ...
  ...
Average loss at step 30: 0.033753
    Test MSE: 0.00243
    Finished Predictions
```

Visualizing the Predictions

You can see how the MSE loss is going down with the amount of training. This is good sign that the model is learning something useful. To quantify your findings, you can compare the network's MSE loss to the MSE loss you obtained when doing the standard averaging (0.004). You can see that the LSTM is doing better than the standard averaging. And you know that standard averaging (though not perfect) followed the true stock prices movements reasonably.





Though not perfect, LSTMs seem to be able to predict stock price behaviour correctly most of the time. Note that you are making predictions roughly in the range of 0 and 1.0 (that is, not the true stock prices). This is okay, because you're predicting the stock price movement, not the prices themselves.

Gaussian Naive Bayes

implements the Gaussian Naive Bayes algorithm for classification. The likelihood of the features is assumed to be Gaussian:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

The parameters σ_y and μ_y are estimated using maximum likelihood

First step import Gaussian Naive Bayes module

```
from sklearn.naive bayes import GaussianNB
```

Second step preparing gaussian naive bayes model extract

```
gnb = GaussianNB()
```

Third step train machine with the training data and predict the test data

```
gnb.fit(X_train,y_train)
y_predict = gnb.predict(X_test)
print(float(accuracy_score(y_test, y_predict)) *100)
```

```
78.50931677018633
```

The used method is

```
def ApplyGuassianNbModel(fetures,label,predictList):
    gnb = GaussianNB()
    gnb.fit(fetures,label)
    y_predict = str(gnb.predict(predictList))
    return y_predict
```

The following table will show the accuracy of each model

All models are tested Using (80% train data, 20% test data)

Model	**Parameters**	**Accuracy**
KNN	K = 3, distance='Euclidean'	86%
Decision Tree c4.5	Criterion= 'entropy'	83%
Gaussian Naive Bayes		78.5%
Model	**Average Loss**	**MSE**
LSTM	0.033753	0.00243

6 REFERENCES

[1] Yves Hilpisch, Python for Finance – Analyze Big Financial Data, OReilly-Media 2014’

[2] Python for Finance: Analyze Big Financial Data

[3] Investopedia : https://www.investopedia.com

[4] Agarwal (July 14, 2017). “Introduction to the Stock Market”. Intelligent Economist. Retrieved December 18, 2017.

[5] Brooks, John (1968). “The Fluctuation: The Little Crash in '62”, in “Business Adventures: Twelve Classic Tales from the World of Wall Street”. (New York: Weybright & Talley, 1968)

[6] BeBusinessed (Apr 8, 2015) “History of The Stock Market”