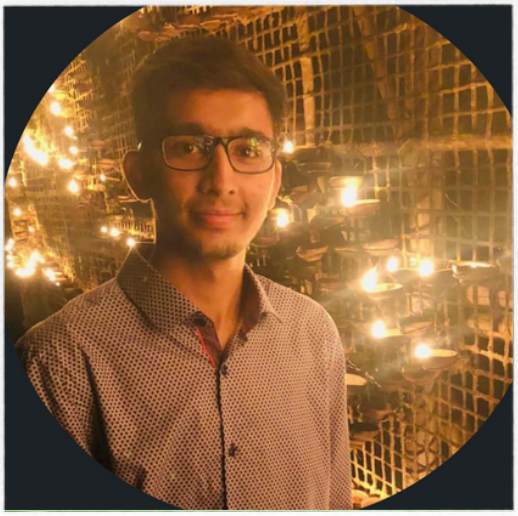
QUANT CLUB IIT KHARAGPUR

QUANT BOOK 2021-2022

COMPILATION OF INQUISITIVE QUANT IDEAS AND INSIGHTS

QC IIT KHARAGPUR 2022

QC Team



Anmol

Motiar Rehaman

Shalini

Rahul

Manthan

Sneha

Ishan

Yashvi

Aayush

Ananya

Pradipto

Shreekrishna

Archit

Abhishek

Srishty

Ayush

Raghav

Anushka

Harsh

Sourashis

Neha

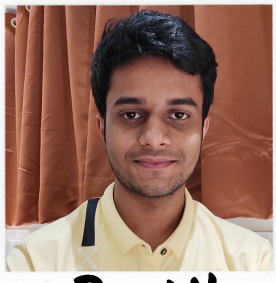
Yatindra

Shashwat

Aman

Ramashish

Rahul



Pratik

Nilesh

Index

Introduction to Algorithmic Trading

Making the right choices while trading and taking emotion away from all the decisions you make in a high-stakes live market is not the easiest thing to do, right? Even with ample knowledge of the stock market and the resources to make active trades, it's not humanly possible to be on the lookout for and capitalise on all possible profit opportunities. You do have a computer though, which is exponentially faster than you at doing preprogrammed things. If only you could teach it to take trade decisions for you, you would essentially have a money-making machine on your hands! This is where algorithmic trading comes in.

Algorithmic trading was introduced in the 1970s, this was when highly computerized trading systems emerged in the American financial markets. The New York Stock Exchange also introduced a system in 1976 which enhanced the acceptance of electronic trades by traders. Algorithmic trading was popularized by Michael Lewis, an author who drew the attention of market traders and the public to high-frequency algorithmic trading. In the status quo, around 70 percent of the trading in the US equities market is automated through algorithms. Why did algorithmic trading become so popular, and what are the risks that come along with it? That is what we seek to explore in this article.

What is Algorithmic Trading?

Algorithmic trading, black-box or algo trading are just different names used to describe the process of having a machine do the trading for you. This is achieved by writing a bunch of algorithms that carry out the trade. Algorithms are a set of predefined instructions that carry out trades instead of humans. These algorithms can be based on timing, price, or a statistical mathematical model.

A straightforward example of an algorithmic trading algorithm

A very basic example of an algorithm that can carry out trades without any predictive analysis is based on using moving averages. A trade call to buy 50 shares of a stock is sent out when its 50-day moving average goes above the 200-day moving average and these shares are sold when the 50-day moving average goes back below the 200-day moving average.

Algorithms can look at and analyse much more data than humans at a very fast rate. This makes them perfect to capitalise on arbitrage opportunities that might arise. They can look at multiple brokers for price differences or profit from price differentials between stock and future instruments.

Trading Algorithms are broadly classified into two types:-

Execution Algorithms

These algorithms are the ones used to carry out bulk orders in small chunks so as to not make a loss. For example, if you own 10 percent equity in a company, and you initiate a sell order for the same, as the order executes the market price will fall as there suddenly is more supply without a symmetric increase in the demand. Then the rest of your orders will be executed at a decreasing price and you will suffer losses. Execution algorithms aim at reducing this market impact and thus save costs. Since the impact and need of these algorithms only arise with very large orders, these are mainly used by mutual funds, investment banks, hedge funds, etc. to carry out their buying and selling.

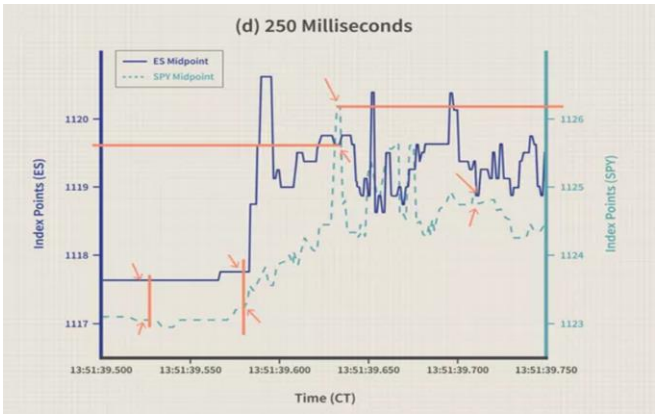
Situational Algorithms

These algorithms are designed to work in the live market and make trading decisions based on the current situation of the market. They are designed to replace humans as more intelligent decision-makers and are aimed at making profits. They can spot patterns that they are designed to look for even across market noise depending on how good the algorithm is and can make consistent profits for the entity deploying them. These algorithms are usually proprietary and coming up with a good performing situational algorithm is going to make you lots of money.

Most algorithmic trading consists of execution algorithms due to the huge impact they have on costs to large corporations. That is to say, the bulk of the trades carried out by algorithms is in the category of execution rather than situational.

Why algorithms instead of humans?

Let us take an example of E-mini S&P 500 futures (ES) and SPDR S&P 500 ETFs (SPY) at different time frequencies



As we can observe from the charts above, these two instruments follow each other very closely when seen on a larger time frame, but when we go down to the milliseconds, their prices do fluctuate a lot and a price differential does arise. These are arbitrage opportunities which means that a profit from this is guaranteed. You can simply buy the instrument at a lower price and sell the one at a higher price and eventually since they will get to the same price, the original price difference guarantees a profit for you. As we can see above, these trade opportunities arise only for milliseconds. However, their frequency is high. This is where the idea of high-frequency trading comes in. In HFT, trades are carried out frequently with a small profit every time, but this adds up to getting a substantial amount over a longer period of time.

Most of the High-Frequency trading that goes on (comparing by the number of orders) is not based upon situational algorithms, but rather by market makers who provide liquidity to the market by acting as an intermediate in market transactions. When orders come in, these HFT algorithms respond to them quickly by buying at a slightly lower price than the market and selling at a slightly higher price and this spread is what allows the market maker to make a profit. And since orders that are coming in are being responded to very quickly, the liquidity in the market increases and trades can go on smoothly even when the actual buyers and sells (non-market makers) come in to trade at different times.

In the context of this article, the major mention of HFT will be around using it for profits on a smaller scale and the market maker perspective is dealt with in specific trading strategies later on.

The Benefits of Algorithmic Trading using HFT

- Can profit significantly even from small price changes: As in HFT the trades are being made frequently, if the correct trading opportunity arises, we can even make a profit from a small price fluctuation
- Profit from more information than a human can process: Algorithmic HTF can capitalize on arbitrage opportunities between different trade markets which is virtually impossible for a human to continuously monitor
- Higher liquidity in the market: Since the market makers are employing Algorithmic HFT as well, there is always someone to buy or sell at the current price which improves the market liquidity to a great deal

Strategies for Algorithmic HFT

Trend-following Strategies

- These strategies come down to automating technical analysis.
- Instead of using indicators to predict the pricing, we can program algorithms to do so.
- We do not need to make complex predictions and instead just need to initiate trade based on simple indicators like moving averages.
- The example of using 50 and 200-day moving averages is a trend following strategy.

Capitalize on Arbitrage

- Due to certain market conditions, arbitrage opportunities exist which fast and precise algorithms can capitalise on.
- This can be as simple as buying and selling a stock that is listed on two exchanges at slightly different prices, or as complex as taking advantage of price differentials between pairs or stocks and their respective futures.
- These algorithms need to be blazing fast as these arbitrage opportunities exist for only a small period of time.
- Execution of such algorithms needs data streams at extremely low latencies.

Trading Range(Mean Reversion)

- Based on assumption that stocks maintain a mean price and fluctuate in a range around it.
- Trades can be automatically triggered when the price goes out of this range.
- This is one of the strategies used by execution algorithms to carry out large trades.

Implementation Shortfall

- This is not a trading strategy but rather a situation that arises when trading manually without an algorithm or algorithms that are slow.
- Occurs when the price changes between order placement and execution.
- For example, if you place a market order at $150 but by the time the order is executed, the price goes up to $150.5, that 50 cent loss per quantity bought is a huge loss when we are talking about trading frequently or trading based on small differences in pricing.
- Can be decreased with faster algorithms that place limit orders on current prices.

Volume Weighted Avg. Price(VWAP)

- Break up a large order and execute it in small volumes based on historical volume profiles.
- The aim is to execute the order close to VWAP.

Time-Weighted Avg. Price(TWAP)

- Executes a large order at equal intervals of time.
- The aim is to execute the order close to the TWAP.
- This and VWAP are some of the strategies used by large holders to buy or sell in the live market without affecting the price too much.

Mathematical Model Based Strategies

- Strategies based on proven mathematical models can be effectively employed using algorithms.
- One such strategy is the delta-neutral trading strategy.
- It allows trading on the basis of delta ratios which compare the change in price of an asset to the corresponding change in the underlying asset.
- An algorithm can effectively analyse the past volatility data and tell us whether employing such a strategy would be beneficial.
- Another mathematical strategy is the martingale strategy.

Deep Neural Networks

- As we know that price relationships in certain situations can be modelled mathematically.

- Deep neural networks are capable of learning complex non-linear relationships that might not even exist from the point of view of a human.
- These can then be used to forecast the prices and can then be traded accordingly.
- The deeper the neural network, the better it might be able to understand the relationship but too much training on the data can result in it picking up the noise at which point the predictions become worthless.

Sentiment Analysis

- As we know there are a lot of current securities (aka crypto) that don't have any fundamentals to go off of.
- Other than the expandability of the currency and other such factors, in the short term the dependency of such securities is correlated higher to market sentiment than traditional stocks.
- In such scenarios, sentiment analysis which is a well known NLP problem can be used to aid decision making.
- There exist live data feeds of bitcoin sentiment from the likes of Thomson Reuters which analyses market sentiment from a variety of social media platforms and news sites which can be used as part of your algorithm to trade these securities.

Disadvantages and Roadblocks

If HFT and algorithmic trading are as good as you might think them to be in this article, then why isn't everyone using them to make a sure shot profit. Let us see some of the reasons why:-

High cost of entry

HFT requires you to be close to the origin of the data feed so there is less delay as even milliseconds of delay can lead to the loss of many arbitrage opportunities to people who have better facilities. Thus, you need a good location and extremely fast and powerful computers.

High transaction costs

The number of transactions we execute during HFT is very high and might flow into hundreds per minute at times. The transaction costs on these trades are also in proportion, and this takes away quite a lot of profit margin. Moreover, this adds to the additional calculation of taking these transaction costs into account when evaluating whether a trade is going to be profitable or not.

More data translates to more profit

There are tiers of live data feeds, the higher tiers being more expensive to set up but also providing more information about the current market trade. This means you will need to pay more to have more information to retain your competitive advantage.

Simple glitches mean disaster

Since trades that are profitable are executed super fast, in case of an anomaly, a wrong trade will be executed in bulk with speed as well and the losses can skyrocket very fast. One such example is Knight Capital which due to faulty software lost 40% of the company value in a single day after sustaining losses of $440 million and was eventually bought out.

Conclusion

while being an increasingly robust approach to trading, algorithmic trading due to the competitive nature of the market is still very hard to get started and profit in if you are a small investor. But for big players in the market who have a lot of capital to invest and can afford top of the line hardware, this can be an excellent way to increase their profits and cut losses.

MODERN PORTFOLIO THEORY

What is Modern Portfolio Theory (MPT)?

Harry Markowitz developed a basic but fantastic portfolio selection theory which was published in the Journal of Finance in 1952. **Modern Portfolio Theory, also** popularly known as MPT proposes a solution to build a balanced portfolio that provides the **highest return** on a particularly given risk of the investor. **Beauty of this theory is its simplicity**. It aims for the simple use of statistical methods for predicting the optimal balance for a portfolio. It is a very popular theory all over the world. It **initiated a wave of investment and financial research**.

Main points MPT focuses on are:

- Do not keep all your eggs in one basket. (Famous quote by Warren Buffet) The point of this quote is to invest your money in different asset classes to minimize the risk and try to maximize the return. This concept is known as **Diversification of the Portfolio,** and this is the main focus of this portfolio theory.
- To find the percentage or weights of each asset class in a portfolio to maximize the return and minimize the risk. This is done by using simple statistical methods.

Modern Portfolio Theory Methodology

The mathematical relation which was given to find the optimal weights is: He used **variance** and **covariance**.

Variance = $\sigma^2 = \frac{\sum_{i=1}^{n}\left(x_i - \mu\right)^2}{n-1}$ **(proposed risk associated with an asset)**

Covariance = $\frac{var(xy)}{var(x)\,var(y)}$ **(relationship between two assets)**

Covariance is generally between -1 to 1 (completely opposite to complete parallel)

The algebra followed in this theory:

$$\mu_{p,x} = E[x^T R] = x^T E[R] = x^T \mu = \begin{pmatrix} x_A & x_n \end{pmatrix} \begin{pmatrix} \mu_A \\ \mu_n \end{pmatrix}$$

$$R = \begin{pmatrix} R_A \\ R_n \end{pmatrix}, \quad x = \begin{pmatrix} x_A \\ x_n \end{pmatrix} \qquad R_{p,x} = x^T R$$

$$\begin{pmatrix} \sigma_A^2 & \sigma_{AB} & \sigma_{AC} \\ \sigma_{BA} & \sigma_B^2 & \sigma_{BC} \\ \sigma_{CA} & \sigma_{CB} & \sigma_C^2 \end{pmatrix} = \sum \qquad \sigma_{p,x} = var(x^T R) = x^T \sum x$$

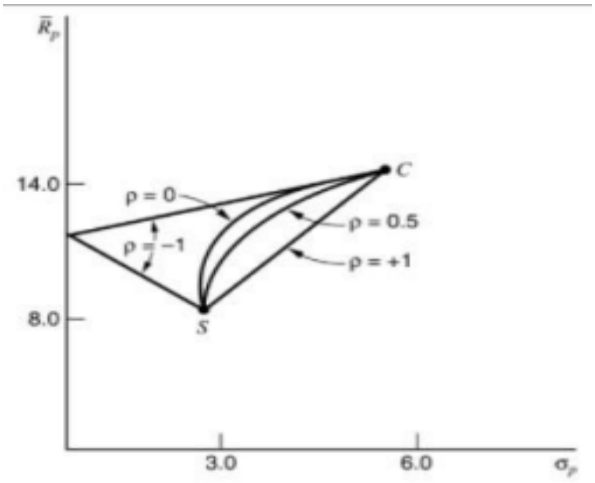
```
def portfolioreturn(weights):
  return np.dot(df.mean(),weights)
```

```
[ ] def portfoliostd(weights):
      return np.dot(np.dot(df.cov(),weights),weights)**(1/2)*np.sqrt(252)
```

The symbols have the usual meanings here.

The weight vector x is chosen randomly and for each weight Expected Return vs Risk is plotted. This plot is known as **Efficient Frontier**.

The Efficient Frontier looks like:



A parameter known as **Sharpe Ratio** was defined for this theory. It is the ratio of Returns / Risk of the portfolio. So, we need to find the weight which gives the highest Sharpe Ratio.

$$\textit{Sharpe Ratio} = \frac{R_p - R_f}{\sigma_P}$$

Where:
R_p = return of portfolio
R_f = risk-free rate
σ_P = standard deviation of the portfolio's excess return

Advantages of MPT

•MPT became incredibly famous after its release in 1952. After the introduction of MPT people started optimizing their portfolios in some rational way. Almost all the fund managers at that time used MPT to balance and rebalance the weights of their portfolio.

•People started using MPT for creating a portfolio of ETFs (which is a collection of stocks which is traded in the market as stocks. Like, you can sell and buy ETFs same as stock, you need not to hold it for years like Mutual Funds). This helped people to truly diversify their portfolio and increase their expected return.

•Each stock is related to unsystematic risk (Risks which are not under control of investors like Management Change, Faulting of Company, etc.). MPT aims to dampen this unsystematic risk by diversifying the portfolio. It proposes that one can't control the unsystematic risk but can reduce it by diversifying the portfolio.

Disadvantages of MPT

•The biggest fault of this theory was that it used variance or volatility for risk calculation. Upside Volatility is preferable to the investors, so it is not rational to account it in the risk calculation. To solve this, we should only account **downside-volatility or downside-risk which means the variance of that results only when there was a loss. This was proposed by Brian M. Rom and Kathleen Ferguson in 1991 as Post-Modern Portfolio Theory (PMPT)**.

•Each stock is related to unsystematic risk (Risks which are not under control of investors like Management Change, Faulting of Company, etc.). MPT aims to dampen this unsystematic risk by

diversifying the portfolio. It proposes that one can't control the unsystematic risk but can reduce it by diversifying the portfolio.

•People started using MPT for creating a portfolio of ETFs (which is a collection of stocks which is traded in the market as stocks. Like, you can sell and buy ETFs same as stock, you need not to hold it for years like Mutual Funds). This helped people to truly diversify their portfolio and increase their expected return.

•MPT became very famous after its release in 1952. After the introduction of MPT people started optimizing their portfolios in some rational way. Almost all the fund managers at that time used MPT to balance and rebalance the weights of their portfolio.

• It was also assumed in MPT that the market is ideal. All the investors in the market are rational; it is not affected by any factor, or the effect is gradual. Which is also a major downside of this theory because we all know that markets are never ideal. It functions majorly on sentiments. For e.g., Covid brought up a negative sentiment and the market crashed.

• Third major downside of this theory is it works on historical data only. So, it is very difficult to believe that the portfolio made on historical data will sustain the unpredictable future

Challenges on how to use it in today's scenario…

MPT can give you only weights of the stocks in the portfolio hence leaving us with a great challenge of selecting stocks.

There are certain guidelines to do this:

• The most logical hack is to select assets with negative correlation. Like you can mix govt. issued bonds with small-cap and large-cap. Bonds are pretty stable and almost considered risk-free.

• The higher your selection correlation is near to -1 the better your portfolio will be.

• Another popular hack is to apply MPT to already created 2 efficient portfolios. This is known as **TWO-FUND THEOREM.**

TWO-FUND THEOREM

This theorem states that you can hold a weighted portfolio of 2 already available efficient mutual funds. (Mutual funds means an already created pool of assets). This can create another efficient portfolio to invest in. Advantages of this is that it is easier to have such a portfolio because the mutual funds will be low in cost than

holding each individual asset. And the portfolio will be easily maintainable by the investor because he needs to see only the two fund performances.

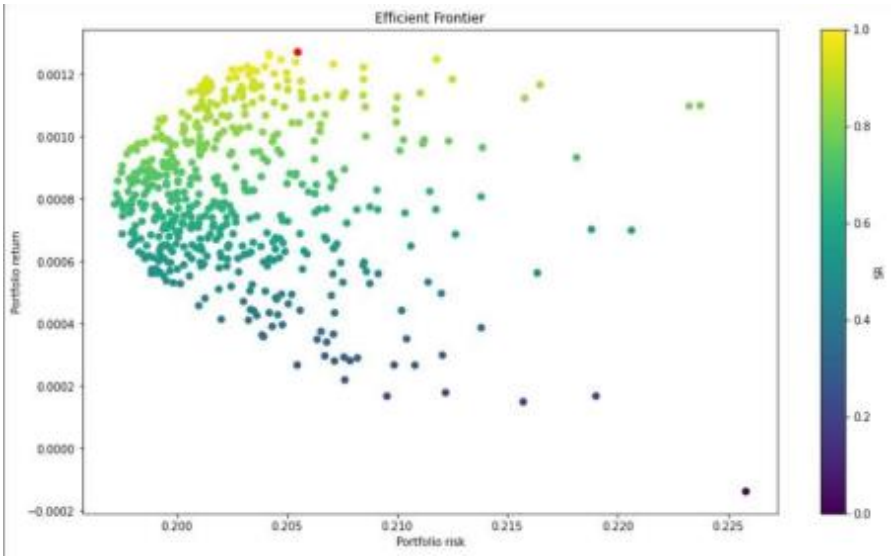
It can be shown that if you want your expected return between the 2 efficient funds then you can hold them both in certain weights (determined by MPT) to expect the return in comparatively less risk. It is similar to holding two of your favourite stocks in a portfolio.

Why MPT lost its significance

MPT lost its significance as more complicated research started in the finance industry. Many researchers started proposing better methods to evaluate risk
return considering the psychology of the investors and financial markets. This was termed as **behavioural finance** which involves the sentiment analysis of the market and other concepts related to investors and market psychology.

But MPT served as the starting point for portfolio optimization studies and encouraged the study of financial markets more critically. Some examples are CAPM**, SML, etc.**

Assets chosen for a sample run was `assets = ['AAPL','MSFT','GOOG','AMZN']` between 2021-01-31 to present. The efficient frontier derived from this run was



The red point in the efficient frontier is the point of highest Sharpe Ratio. The weights for this point is

```
array([0.22252147, 0.01210444, 0.53133647, 0.23403762])
```

We invested $10,000 on 2021-10-01 and ran the algorithm to find the returns and percentage returns.

```
[ ]  returns = []
     stds = []
     w = []
     r = []
     for i in range(500):
       weights = weightcreator(df)
       returns.append(portfolioreturn(weights))
       stds.append(portfoliostd(weights))
       w.append(weights)
       r.append(sharperatio(portfolioreturn(weights),portfoliostd(weights)))
```

```
marketreturn = portfoliovalue - startamt
marketreturn
```

```
456.9062523647517
```

```
pctreturn = (marketreturn/startamt)*100
pctreturn
```

```
4.569062523647517
```

Returns: $456.9

Percentage Return: 4.56% in almost 4 months.

FAMA AND FRENCH 3 Factor model

Introduction:

The Capital Asset pricing model (CAPM) was developed by Shape (1964), linter (1965), and Mossin (1966). This model aims to answer the question of how we can price one security taking into consideration the risk and return that this security poses. A vast amount of research has been conducted to test the validity of the CAPM in explaining the variation in the return. However, these studies provide no evidence to support this model. Motivated by the weakness and limitations of the CAPM, Eugene Fama & Kenneth French (1992), used a sample of nonfinancial firms drawn from 3 major US financial markets over the period of 1963 to 1990, Fama and french tested the ability of the market beta coefficient, size, leverage book to market equity ratio, to predict the cross-section rate of return, they found no relationship between the market beta factor and stock return. Inspired by the results Fama and French(1993) developed what become Fama and french 3-factor model.

Capital Asset Pricing Model(CAPM) and its failures:

As mentioned Fama and french is nothing but an extension of CAPM. so before we dive into it let's look into CAPM and its failures. Below mentioned are important terms that gave rise to CAPM.

Volatility is a proxy for risk: It is one of the widely accepted measures of risk. Volatility means fluctuations because the asset whose value fluctuates dramatically is perceived to have more risk because asset value at the time of selling is less predictable.

Diversification: One of the ways to reduce the risk is through diversification.it means holding multiple assets to reduce the volatility, without lowering the expected return, by spreading the same amount of money across multiple assets. The idea behind this technique is that a portfolio is constructed of different kinds of assets, on average, yield higher long-term returns and lower the risk of any individual holdings or security.

As we can now tell there can be two types of risk i.e risk that can get Diversified and risk which cannot. Now an investor cannot expect a return on a diversified risk. So one can only expect a return on undiversified risks.

Systematic Risk: Systematic risk refers to the risk inherent to the entire market or market segment. It is also known as undiversifiable risk, volatility, or market risk that affects the entire market, not just one particular stock. And the measure of systematic risk is called Beta. Beta is measured by a measure with which return varies relative to the overall market

Beta = cov(re,rm) / var(rm)

re = return of individual stock

rm = return of overall market

cov = covariance

var = variance

The beta of the investment measures the amount of risk the investment adds to the portfolio which resembles the market

if the beta is greater than 1 then the risk is greater than market

If beta is less than 1 then the stock is less risker than market

CAPM: As the name suggests it is a model used for pricing assets.it attempts to provide a relationship between market return and a beta of an asset and its corresponding return/expected return.

CAPM made a number of simplifying assumptions about this the most important to note is the behavior of investors and the presence of a **single risk factor.**

The logic of mode:

- Let's consider an asset with no volatility, and thus, no risk thus return the risk-free rate
- Now suppose the asset moves in lockstep with the market, or beta equals one. Then we can say that return equals a return on the market
- Now suppose beta is greater than 1 then we can say that return should be greater than that of the market due to higher risk
- When we generalize the above points we get simple formula :

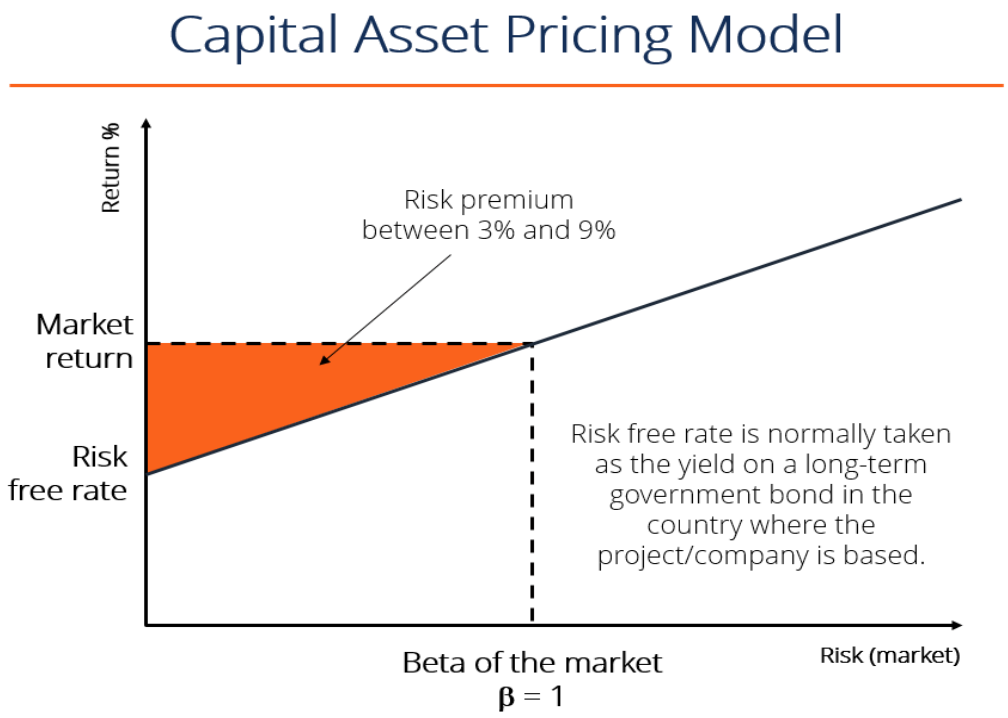
Ei = beta* (ERm - Ri) + Ri

Ei = expected return

Ri = risk free return (generally government bonds)

(ERm - Ri) = market risk premium

We can see the plot below

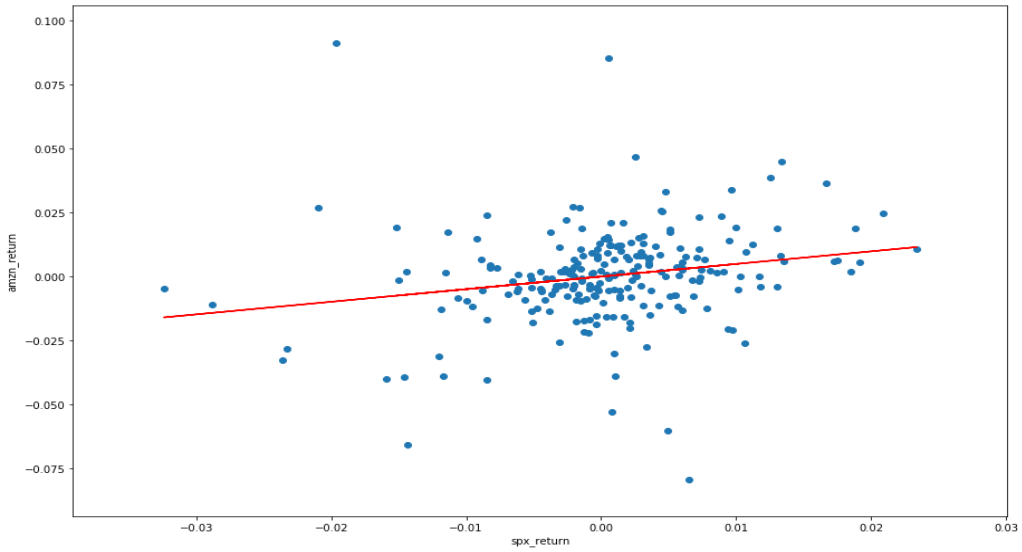


Problem with CAPM that gave rise to Fama and Fench :

Problems with CAPM were that there were lots of unreal assumptions.CAPM says that the expected return is a positive **linear function of beta.** As one of the major assumptions is that you only need one factor(beta) to estimate the expected return and other factors should not add any value to estimating the expected

return. But when we add other factors like size and book to market(B/M) the linear relationship between beta and expected returns is no longer in effect which contradicts the main assumptions of the CPAM.

The plot we get after adding other factors in the CAPM model is shown below as we can see it's almost horizontal which results in an incorrect calculation of the expected return as we calculated from the CAPM model.



Fama and French 3 Factor model:

As we can see that addition of other factors affects the expected returns so Fama and french did extensive research on samples of data from 1963 to 1990 of 3 major US financial markets and found out that factors describing “value” and “size” are two major factors outside market risk. There two new factors were added to the CAPM model.

So to represent these two new risks new factors were constructed that are

- SMB (small minus big) to address size
- HML (high minus low) to address value

So after adding this two new factors we get our new expected return model is:

E(ra) = rf + βa*(E(rm)-rf) + sa*SMB + ha*HML

Coefficients for **SMB** and **HML** are **sa** and **ha** respectively and two new factors are SML and HML and the rest is the same as in the CAPM model. Now let's understand what is SMB and HML.

SMB (small minus big): The SMB is the excess return that smaller market capitalization returns versus larger companies. If a portfolio has more small-cap companies in it then it should outperform the market over the longer run.

HML(high minus low): The HML is provided to the investors for investing in companies with high book-to-market value(means value placed on the company by accountants as a ratio relative to the value of public markets placed on the company, commonly expressed as B/M). Specifically, value stocks(high B/M) outperform growth stocks (low B/M).

To tell the truth, there is no proper explanation for the effect of HML and SMB. and there is lots of argument over the correct explanation for it but one of the easiest explanations for the SMB and HML is:

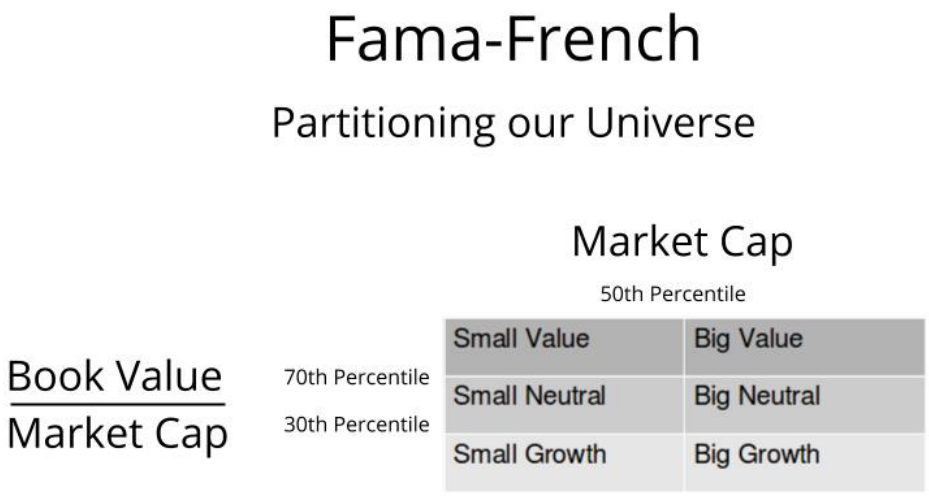
Reason for the size risk: for SMB which is a measure of size risk, small companies logically should be more sensitive to many risk factors as a result of their undiversified nature and reduced ability to absorb negative financial events. So when investors invest in these small companies taking the risk they expect an excess return

Reason for value risk: On the other hand, the HML factor suggests higher risk exposure for typical “value” stocks (high B/M) versus “growth” stocks (low B/M). This makes sense intuitively because companies need to reach a minimum size in order to execute an Initial Public Offering. If we later observe them in the bucket of high B/M, this is usually an indication that their public market value has plummeted because of hard times or doubt regarding future earnings. Since these companies have experienced some sort of difficulty, it seems plausible that they would be exposed to greater risk of bankruptcy or other financial troubles than their more highly valued counterparts.

Now the question comes of how to calculate SMB and HML so in order to calculate?.

So SMB and HML are made up of 6 portfolios formed by size and book to market ratio. So as per the method used by Fama and French in 1993.

In June of each year all stocks on the study samples are ranked based on the firm size(we take an average daily closing price till June of each year of each stock in the sample) and are assigned into two portfolios of size(small (S) and big(B)) based on split point which is 50% that means highest 50% are big and rest are small. Then stocks are divided into 3 more portfolios of value, neutral, and growth book to market ratio on the basis of the percentile system which you can see in the figure below.



Let's take an example to suppose we have to split the above-given form and we have data of stocks from each June of 2000 to 2010 then the spit will look something like this.

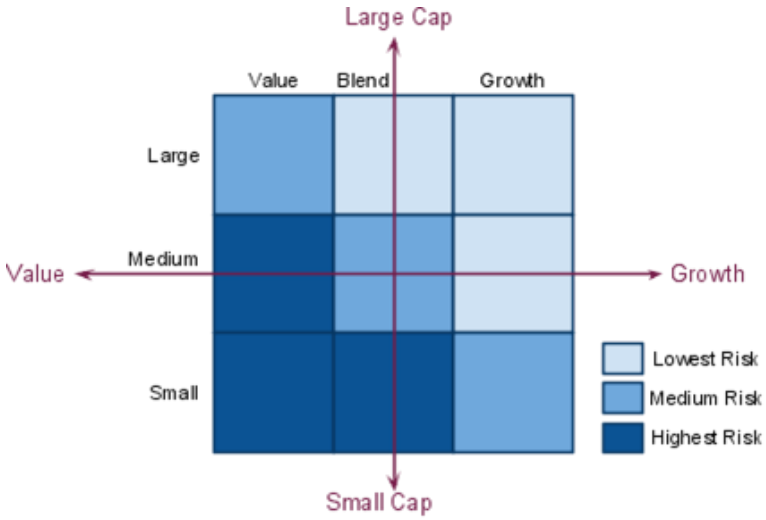
Year	SL	SM	SH	BL	BM	BH	Total
2000-2001	9	19	29	25	27	5	*114*
2001-2002	11	20	29	25	28	7	*120*
2002-2003	8	23	30	28	26	6	*121*
2003-2004	7	27	25	28	21	11	*119*
2004-2005	7	29	32	33	25	9	*135*
2005-2006	8	28	35	34	29	7	*141*
2006-2007	7	30	40	39	32	7	*155*
2007-2008	8	34	43	40	35	9	*169*
2008-2009	19	31	41	35	42	14	*182*
2009-2010	21	39	42	41	42	20	*205*
Average	***10.5***	***28***	***34.6***	***32.8***	***30.7***	***9.5***	***146.1***

So now to calculate SMB and HML we use the formula

SMB = 1/3 (Small Value + Small Neutral + Small Growth) - 1/3 (Big Value + Big Neutral + Big Growth)

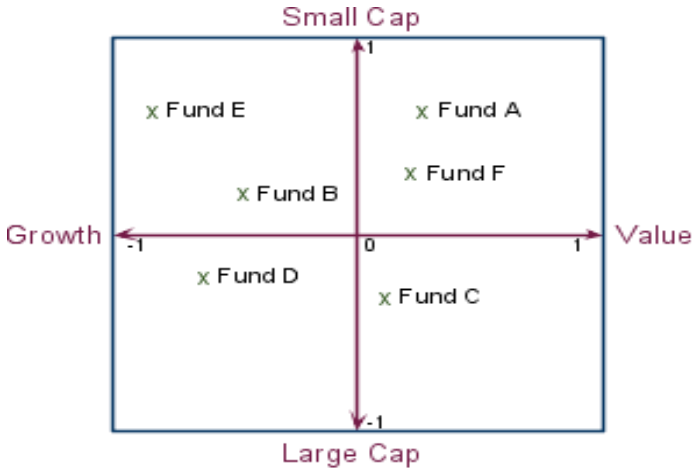
HML = 1/2 (Small Value + Big Value) - 1/2 (Small Growth + Big Growth).

Implementation:



Morningstar Investing Style Box

- Other than just asset pricing 3-factor model also used in categorizing funds is that investors can effectively choose the amount to which they are exposed to the risk factor when investing in a particular fund.



Three-Factor Model: Fund Profiles

We can say from the above distribution that there is most risk in fund A and least in fund D.

- In practice, the above characterization is executed through multivariate regression. The historical returns of a particular portfolio are regressed against the historical values of the three factors, generating estimates of the coefficients which we will see in the code part for some of the funds.

After Fama and French, there was further research done and two new models that were the extension of Fama and french were introduced. Let's look what were those models

4 Factor model:

Mark Carhart added the Fourth factor, **Momentum(UMD)**(Which is a tendency for an asset to continue on a given path, rising or falling) into the 3-factor model. The model is also known as Carhart 4 factor model. The addition of this factor boosted the explanatory power of the model to 95% vs the 90% of the 3-factor model on the same data used for the 3-factor model.

The Momentum Factor(UMD) can be calculated by subtracting the equal-weighted average of the lowest-performing firms from the equal-weighted average of the highest performing firms, that lagged one month. A stock would be considered to show momentum if its prior 12-month average of returns is positive, or greater.

So after adding the momentum factor to the 3-factor model our overall model becomes:

$$E(ra) = rf + \beta a*(E(rm)-rf) + sa*SMB + ha*HML+ma*UMD$$

Everything is the same as that of the 3-factor model except for the addition of the Momentum factor. ma is the coefficient of momentum factor.

Five-Factor model:

The Fama-French five-factor model which added two factors, profitability and investment, came about after evidence showed that the three-factor model was an inadequate model for expected returns because its three factors overlook a lot of the variation in average returns related to **Profitability(RMW)** and **Investment(CMA).**

The theoretical starting point for the Fama-French five-factor model is the dividend discount model as the model states that the value of a stock today is dependent upon future dividends. Fama and French use the dividend discount model to get two new factors from it, investment and profitability. the Five-Factor Model also explains roughly 95% of portfolio returns.

Profitability(RMW) factor can be calculated by The return of stocks with high operating profitability minus the return of stocks with low or negative operating profitability. And for Investment factor(CMA) can be calculated by The return of the stocks of companies that require little on-going capital investment to maintain and grow their businesses minus the return of the stocks of companies that require large on-going capital investment.

So after adding these two factor to out 3 factor model our overall model becomes:

$$E(ra) = rf + \beta a*(E(rm)-rf) + sa*SMB + ha*HML+ra*RMW+ca*CMA$$

Besides addition of new two factors profitability and investment rest of the model is same as of 3 factor model. ra and ca are coefficients of porfitibiliy and investments respectively.

Conclusion:

In conclusion we can say that 3 factor is nothing but extension of CAPM which has better explanatory power of portfolio as it cover additional factors by explaining how small cap and value stocks outperforms market. 3 factor model can be used for pricing assets and also mainly used for portfolio management.

Black Litterman Model

Why black Litterman:

One of the major drawbacks of a very famous market in a Portfolio initialisation model CAPM is that it does not incorporate the views of an investor i.e if he or she has a hunch about a particular asset and wants to to invest in that asset. Black Litterman model It was made to overcome this weakness by incorporating the views of an investor in the portfolio.

The Formula:

$$E[R] = \left[(\tau\Sigma)^{-1} + P'\Omega^{-1}P\right]^{-1}\left[(\tau\Sigma)^{-1}\Pi + P'\Omega^{-1}Q\right]$$

$E[R]$ is the new (posterior) Combined Return Vector (N x 1 column vector)

τ is a scalar

Σ is the covariance matrix of excess returns (N x N matrix)

P s a matrix that identifies the assets involved in the views (K x N matrix or 1 x N row vector in the special case of 1 view)

Ω is a diagonal covariance matrix of error terms from the expressed views representing the uncertainty in each view (K x K matrix)

Π is the Implied Excess Equilibrium Return Vector (N x 1 column vector)

Q is the View Vector (K x 1 column vector)

Let's not get scared by looking at the formula and let's look at each term one by one :

The first half of the formula $\left[(\tau\Sigma)^{-1} + P'\Omega^{-1}P\right]^{-1}$ is just a normalisation term so that the new combined return vector has weights with sum up to 1. Also note that τ is a very mysterious parameter and there are several research papers based on this parameter itself. Roughly speaking it gives a user defined vote of confidence in our views rather than the variance based vote of confidence as we will see later. We will be using a value of 0.05 which is a commonly used value.

In this article I will be taking an example of 7 countries to illustrate the model as given in the official paper[2].

As we all are familiar with the CAPM equation which gives us the expected rate of return of the market portfolio.

The CAPM model gives us weights according to the market capitalization of each asset.

$$w^{i}_{mkt} = \frac{MC_i}{\sum_j MC_j}$$

Here MC_i is the Market Cap of the ith asset

```
# Equilibrium Portfolio Weights
# AUL, CAN, FRA, GER, JAP, UKG, USA
w = np.array([[0.016, 0.022, 0.052, 0.055, 0.116, 0.124, 0.615]]).T
```

These are the equilibrium portfolio weights according to the market capitalization of the seven countries.

$$\Pi = \delta\Sigma w_{mkt}$$

Π is the Implied Excess Equilibrium Return Vector (N x 1 column vector)

δ is the risk aversion coefficient

Σ is the covariance matrix of excess returns (N x N matrix)

w_{mkt} is the market capitalization weight (N x 1 column vector) of the assets

```
# Correlation Matrix
# AUL, CAN, FRA, GER, JAP, UKG, USA
correlation = np.array([
    [1, 0.488, 0.478, 0.515, 0.439, 0.512, 0.491],
    [0.488, 1, 0.664, 0.655, 0.310, 0.608, 0.779],
    [0.478, 0.664, 1, 0.861, 0.355, 0.783, 0.668],
    [0.515, 0.655, 0.861, 1, 0.354, 0.777, 0.653],
    [0.439, 0.310, 0.355, 0.354, 1, 0.405, 0.306],
    [0.512, 0.608, 0.783, 0.777, 0.405, 1, 0.652],
    [0.491, 0.779, 0.668, 0.653, 0.306, 0.652, 1]])

# Standard Deviation (Volatility)
# AUL, CAN, FRA, GER, JAP, UKG, USA
std = np.array([[0.16, 0.203, 0.248, 0.271, 0.21, 0.2, 0.187]])
```

```
# Variance Covariance Matrix (which can be calculated with the correlation and volatility above)
Sigma = correlation * np.dot(std.T, std)

# delta (δ): risk aversion parameter (scalar)
delta = 2.5

# tau (τ): a scalar measures the uncertainty of the CAPM prior
tau = 0.05
```

Reverse Optimization:

Another term that is usually coined with black litterman model is Reverse Optimisation.

In Reverse Optimisation we focus on optimising the returns and then calculating the weights of each asset in the portfolio by applying some other model rather than the classical procedure in which we optimise our model so as to give the most efficient weights for each asset.

The First Half:

Now we are in a position to understand the the first part of the second term in the formula

$$(\tau\Sigma)^{-1}\Pi$$

Point to be noted here is that Σ is a covariance matrix so Σ^{-1} is a vote of confidence. Now τ is just a scalar so the complete terms represent how much confident I am in my portfolio returns if I allot the weights to each asset according to its market capitalisation.

Let's make a difference with our views :

Let us say that I have two views:

View 1 : Germany will have an excess return of 0.05% over France and United Kingdom

View 2 : Canada will have an excess return of 0.03% over USA

Now the views are represented in a view vector Q which just stores the values of excess returns that are expected by the investor

```
Q = np.array([[0.05],[0.03]])
```

Now since we have defined our view vector we need a way to say which views are related to which assets, so here comes another matrix known as Pick Matrix P.

$$P = \begin{bmatrix} p_{1,1} & \cdots & p_{1,n} \\ \vdots & \ddots & \vdots \\ p_{k,1} & \cdots & p_{k,n} \end{bmatrix}$$

Here each row is for a view therefore k is the number of views and the number of rows in the pick matrix.

```
# AUL, CAN, FRA, GER, JAP, UKG, USA
P = np.array([
    [0,0,-0.295,1,0,-0.705,0],
    [0,1,0,0,0,0,-1]])
```

In the second row we see that CAN (index 1) overtakes USA (index 6) so +1 at P[1][1] and -1 at P[1][6].

In the first row we see that GER (index 3) overtakes FRA (index 2) and UKG (index 5) combined so +1 at P[0][3] and the -1 is distributed according to the market caps of FRA and UKG thus -0.295 at P[0][2] and -0.705 at P[0][5].

Now we have our views and a way to interlink them with our assets, but our views are not absolute so we need a vote of confidence for them and thus here comes another term Ω which represents the covariance matrix in our views.

$$\Omega = \begin{bmatrix} \omega_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \omega_k \end{bmatrix}$$

Here all the non diagonal elements are zero and the diagonal elements represent the variance in the errors in each view as each view can be seen as Q + E where E is the error which is sampled from a Normal Distribution with 0 mean and ω variance.

Calculating Ω is one of the most difficult tasks of the Black Litterman model so we will skip it and directly go to the result.

$$\Omega = \begin{bmatrix} \left(p_1 \Sigma p_1^{'}\right) * \tau & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \left(p_k \Sigma p_k^{'}\right) * \tau \end{bmatrix}$$

```
# AUL, CAN, FRA, GER, JAP, UKG, USA
Q = np.array([[0.05],[0.03]])
P = np.array([
    [0,0,-0.295,1,0,-0.705,0],
    [0,1,0,0,0,0,-1]])
Omega = np.array([
    [0.001065383332,0],
    [0,0.0008517381]])
```

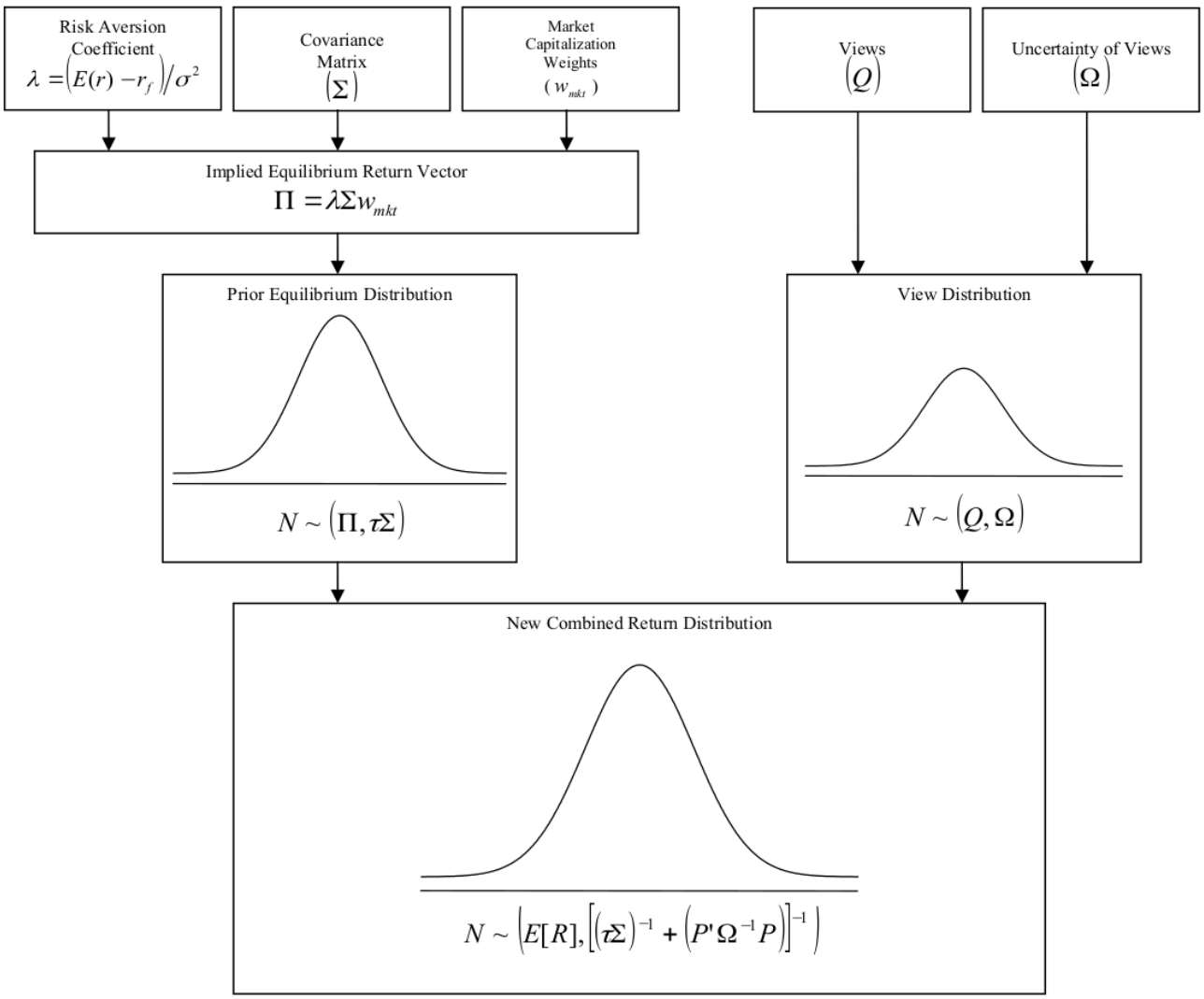
Now we are in a position to understand the second part of the second term in the formula.

$$P' \Omega^{-1} Q$$

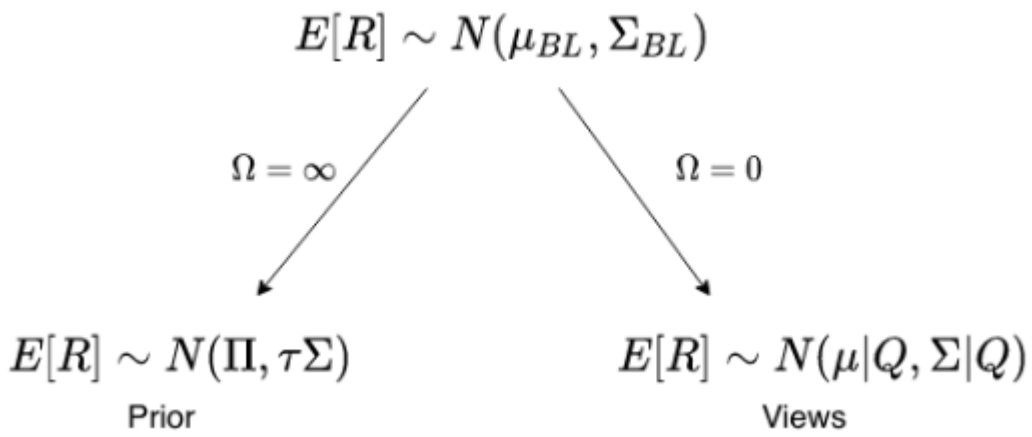
Point to be noted here is that Ω is a covariance matrix so Ω^{-1} is a vote of confidence. The complete terms represent how much confident I am in my portfolio returns if I allot the weights to each asset according to the views.

Ready to launch:

The following two flow charts are self explanatory and very crucial for the understanding of the Black Litterman Model.


Risk Aversion Coefficient
$\lambda = \left(E(r) - r_f\right) / \sigma^2$
Covariance Matrix
(Σ)
Market Capitalization Weights
(w_{mkt})
Views
(Q)
Uncertainty of Views
(Ω)
Implied Equilibrium Return Vector
$\Pi = \lambda \Sigma w_{mkt}$
Prior Equilibrium Distribution
$N \sim (\Pi, \tau\Sigma)$
View Distribution
$N \sim (Q, \Omega)$
New Combined Return Distribution
$N \sim \left(E[R], \left[(\tau\Sigma)^{-1} + \left(P'\Omega^{-1}P\right)\right]^{-1}\right)$

Source : [1]



Source : [3]

Now we have all the terms we need to calculate the returns of our model and let us plug them in and plot the graph.

$$E[R] = \left[(\tau\Sigma)^{-1} + P'\Omega^{-1}P\right]^{-1}\left[(\tau\Sigma)^{-1}\Pi + P'\Omega^{-1}Q\right]$$

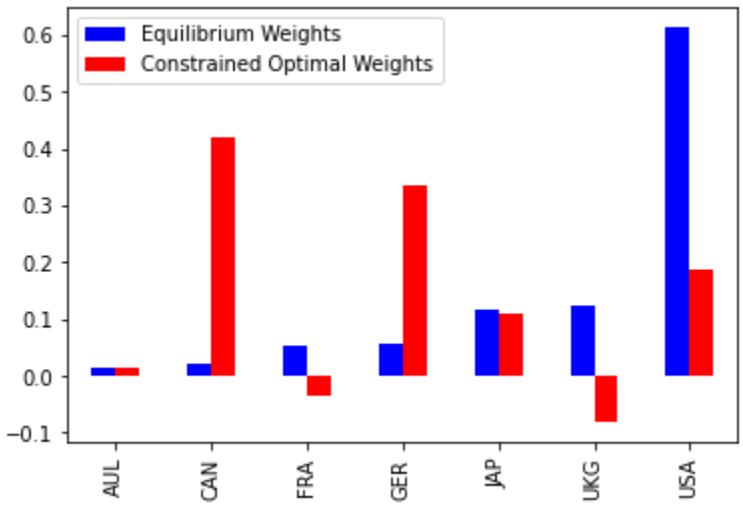
```
r_posterior = r_eq + np.dot( np.dot( tau*np.dot(Sigma,P.T), np.linalg.inv(tau*np.dot(np.dot(P,Sigma),P.T)+Omega)), (Q-np.dot(P,r_eq)))

Sigma_posterior = Sigma + tau*Sigma - tau*np.dot( np.dot( np.dot(Sigma,P.T), np.linalg.inv(tau*np.dot(np.dot(P,Sigma),P.T)+Omega)), tau*np.dot(P,Sigma))

w_posterior = np.dot(np.linalg.inv(delta*Sigma_posterior), r_posterior)

df = pd.DataFrame([w.reshape(7),w_posterior.reshape(7)],
             columns=['AUL','CAN','FRA','GER','JAP','UKG','USA'],
             index=['Equilibrium Weights','Constrained Optimal Weights'])
df.T.plot(kind='bar', color='br')
```

Results:



The blue weights are the ones which were assigned before our views and thus after saying that GER will have an excess return over FRA and UKG we can see the weights of FRA and UKG decreasing and weight of GER increasing. The same goes for CAN and USA.

‘

Pairs Trading

Introduction to Pairs Trading

At its very core Pairs Trading is a statistical arbitrage strategy which is based on statistical concepts like mean reversion principle and stationary stochastic process.

Source: Deutsche Bank, FactSet, Axioma, RavenPack - February 2016

Understanding the Terms

Before we dive into this process of understanding Pairs Trading as a strategy and implementing it, it is imperative for us to understand some of the terms that have been introduced in the paragraph explaining Pairs Trading as a concept

- Statistical Arbitrage Strategies: These are investment strategies that seek to profit from the narrowing of price gap between two or more securities
- Mean Reversion Principle: It is a financial theory that states that an asset's price tends to converge to converge to it's long term mean over a period of time
- Stochastic Process: Stochastic Process is basically a random process which is a collection of random variables that are indexed by some mathematical set
- Stationary Stochastic Process: These are stochastic processes whose statistical properties do not change with time

What exactly is Pairs Trading and what does it involve?

Pairs Trading is basically a trading strategy that involves matching a long position with a short position in two securities which have very high correlation

So now that we understand what the overview of the process in Pairs Trading is we now try to dive deeper into what are the broad processes that would be involved in it. So broadly there are two major processes that we follow, one of which is very intuitive to understand from the above passage explaining what Pairs Trading is exactly and one process which is not so intuitive but would be explained comprehensively in the further parts of this article.

Processes Involved:

- Identifying Securities which have very high positive correlation
- Using Statistical and Technical analysis to seek out potential market neutral profits

Advantages of Pairs Trading

So now that we have a basic understanding of what Pairs Trading is and what are the broad processes that we undertake while implementing Pairs Trading as a strategy we come to the question as to why we should be using Pairs Trading as a trading strategy i.e what are the advantages of Pairs Trading strategy:

- **Mitigates potential losses and risks:** In the best case scenario when the trade occurs as per expectation and both the securities converge to their original correlation after a short period of divergence, the trader is able to generate profits and potential losses are mitigated. But even in scenarios when the trade is not going ideally and one of the securities is underperforming then there is a very high likelihood that the other security will absorb the losses from the underperformance of the other security. Thus we see that Pairs Trading also provides a very robust mechanism for mitigating risks
- **Market Neutral Strategy:** Since Pairs Trading as a strategy only requires a divergence to open up between two highly correlated stocks therefore it is a market neutral strategy, that is, it is independent of the behavior of the market as a divergence could open up even when the market is in a bull run, bear run or going sideways. Simply put a pairs trader can make a profit regardless of the condition of the market
- **Inherent Hedging Mechanism:** As in a Pairs Trade the trader is selling an overvalued security and buying an undervalued security therefore he is limiting his chances of losses considerably. Therefore Pairs Trade provides an inherent hedging mechanism which is not an advantage that is found normally in most trades

Important factors Affecting Pairs Trading

There are three important factors which impact the mechanism of Pairs Trading:

- Parameter Estimation: This involves estimating the parameters such as threshold value and band gaps up to which we allow for divergence among the highly correlated securities and when these securities diverges beyond this gap we execute the trade
- Statistical Modeling: As Pairs Trading is a statistical arbitrage strategy then it is very intuitive to understand that for implementing this strategy and to execute any trade with this strategy we would need to create a statistical model which would evaluate things like the stationarity of the spread and once that is evaluated would help model a threshold line for the statistical correlation
- Market Events: This factor, although an extremely important factor affecting Pairs Trading, is a very rare occurrence as a result of which it is mostly discounted in most pairs trades that are executed on a day to day basis. But as this is an extremely factor it deserves a complete discussion in itself and hence would be taken up at the end of the article

Mechanism of Pairs Trading

As we have got the basic concept of Pairs Trading and the logic and the strategy behind it, let us now get down to as to how we are supposed to execute a pairs trade and what is the step by step procedure we need to follow to undertake a Pairs Trade ● Identifying a pair of securities

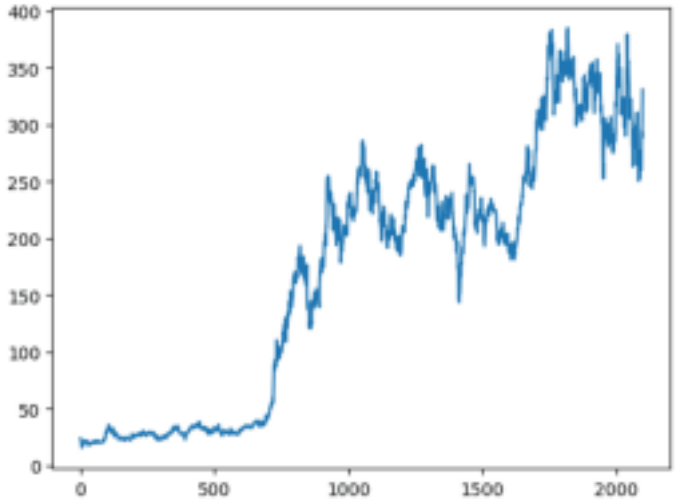
- Checking for cointegration in the pair
- Determining entry and exit points for the trade
- Determination of stop loss for protecting downside (Additional step; Not mandatory)

Breakdown of Pairs Trading

We now go in depth into each step that has been stated in the mechanism of pairs trading so that we get the nitty gritty of how each step is to be executed

- **Identifying a Stationary Pair of Securities**

  Stationarity is an extremely important prerequisite for Pairs Trading. But the dataset of stock prices is not stationary in most cases



The above graph is an example any average stock's price graph and we can very clearly see from the graph that the stock price dataset is not stationary. But to execute a Pairs Trade stationarity is extremely important, so how do we go about this problem then

But before we go into tackling the problem here are some pointers for stock selection for Pairs Trading:

- Large cap companies
- Companies from the same sector or companies under the same banner in complimentary services
- Companies having good fundamentals
- Companies having similar growth rates

So now that we know the parameters to be kept in mind while selecting stock for Pairs Trading, let us get back to tackling the issue of stationarity. To identify pairs of securities we have to create a portfolio of two price series such that the linear combination of these is stationary. If the linear combination is stationary then the individual price is said to be cointegrated with each other.

If there be two stocks A and B then we first create a spread where, Spread=log(a)-m*log(b)

Where

a=Price of stock A

b=Price of stock B

m=The number of stocks of B we have to buy per share of A (regression coefficient)

The initial step is to run a regression model to identify the value of m which would give a stationary value for spread.

For us to have a better understanding of the concept of Pairs Trading we will be doing a case study of HDFC and HDFC bank pair (The two stocks selected cover all the parameters that have been listed for stock selection as a pair)

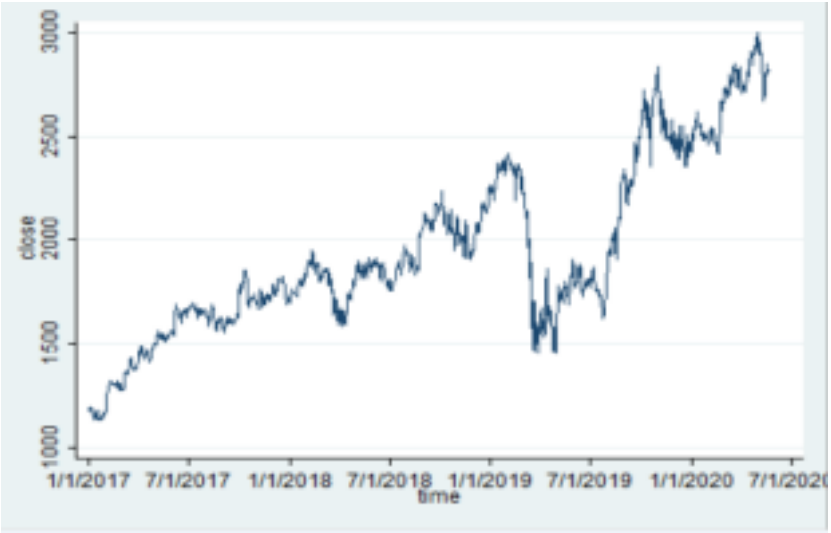
● Checking for Cointegration Among Securities

Once we have determined the value of m in the spread by running a simple linear regression model we then proceed to check the validity of our result by examining the data of spread for cointegration.

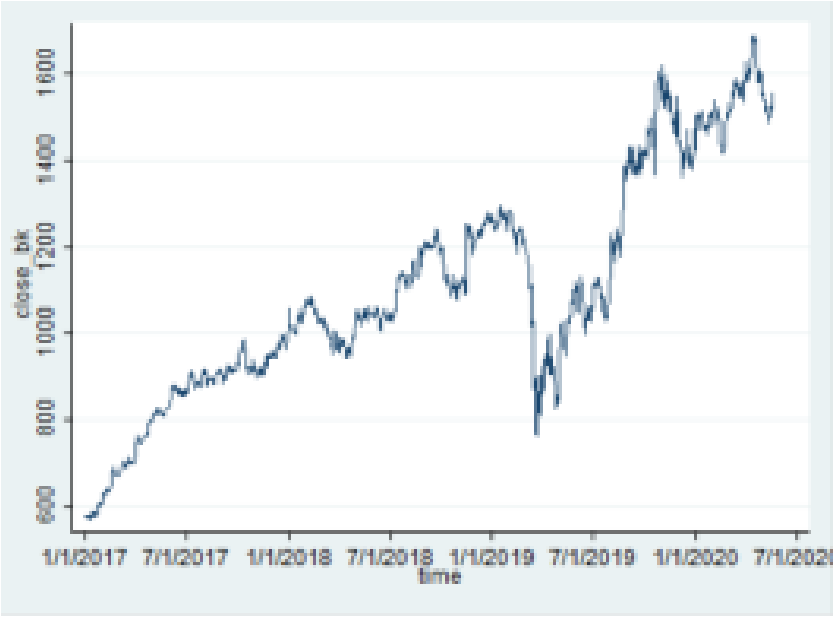
Cointegration is verified by determining the stationarity of the spread, in the scenario that the spread is stationary, both the securities are said to be cointegrated and vice versa.

The mechanism that we employ for checking stationarity of the spread equation is as follows:

- We first plot a two way line graph which is a time series plot to have a preliminary visual test for stationarity



Time series Line graph of HDFC.NSE



Time Series Line Graph of HDFC Bank.NSE

- In the scenario that the line plot is satisfactory we move on to mathematical tests for checking the stationarity of the spread
- We can either make use of the Augmented Dickey Fuller (ADF) test or the Johanssen test to check for stationarity (In this article we will be using the ADF test for checking for stationarity)

.dfuller close, lags(0)

Dickey-Fuller test for unit root Number of obs = 1235

----------Interpolated Dickey Fuller—----------

Test 1% Critical 5% Critical 10% Critical

Statistic Value Value Value

Z(t) 1.418 3.430 2.860 2.570

MacKinnon approximate p

value for Z(t) = 0.5738

The above stats are the result of the ADF test on the adjusted closing price of HDFC Limited over a period of around 4 years. closing price of HDFC limited over a period of 5 years. The highly negative stats at different critical values and a p value which is much greater than 0.05 signifies that a unit root exists and the time series data is non-stationary.

.dfuller close_bk , lags(0)

DickeyFuller test for unit root Number of obs = 1235

----------Interpolated Dickey-Fuller—----------

Test 1% Critical 5% Critical 10% Critical

Statistic Value Value Value

--

Z(t)1.495 3.430 2.860 2.570

--

MacKinnon approximate p

value for Z(t) = 0.5361

These are the ADF tests stats for HDFC Bank limited over a period of around 4 years. The p value of more than 0.5 signifies that a unit root does exist and the time series data is non stationary.

.dfuller sprd , lags(0)

Dickey-Fuller test for unit root Number of obs = 1236

----------Interpolated Dickey Fuller—---------

Test 1% Critical 5% Critical 10% Critical

Statistic Value Value Value

Z(t)26.990 3.430 2.860 2.570

MacKinnon approximate p

value for Z(t) = 0.0000

These final stats are the results of the ADF test on the spread that was calculated after running the regression model. From the result we can clearly see that the p value is 0 and therefore we can easily reject the null hypothesis and say that a unit root does not exist and the time series data of the spread is stationary.

- **Determining Entry and Exit Points for Trade**

We introduce the concept of z-score to identify entry and exit points.

z=(x-mean)/standard deviation

We calculate the z-score of the spread and we set up threshold parameters ● For a 66% confidence level we set up the threshold parameter at sigma ● For a 95% confidence level we set up the threshold parameter at 2*sigma ● For a 99% confidence level we set up the threshold parameter at 3*sigma

If the z-score crosses the upper threshold we short asset A and long asset B and if the z-score crosses the lower threshold we go long on asset A and we short asset B

Limitations of Pairs Trading

Now that we have understood the entire concept of Pairs Trading and also the mechanism and advantages that are related to it, it is also imperative for us to understand the shortcoming of Pairs Trading so that we can undertake measures to overcome these shortcomings and execute better trades. Therefore we now discuss the shortcomings of Pairs Trading:

● Challenging to identify securities with such high statistical correlation: One of the ways to overcome this issue is by identifying multiple pairs of stocks in different sectors which would help in providing trade opportunities in all kinds of situations

● Past prices are not always indicative of future trends

● Black swan events that can make random departure from equilibrium pricing of two securities, changes which are not temporary but permanent: This is a callback to one of the major factors that was mentioned which affects Pairs Trading which was *market events.* To analyse this drawback and to understand its full impact we will discuss a famous real life example.

Long Term Capital Management was an extremely successful hedge fund in the 1990's which attracted a large number of investors. They traded mostly on the basis of statistical arbitrage strategies which primarily included Pairs Trading strategy. During 1997-1998 when the Asian debt crisis the Russian debt crisis which led to a divergence in a lot of historically correlated stocks which LTCM took to be a trade signal and took highly leveraged positions in these stocks but the issue was that what LTCM had failed to identify was that these events were Black Swan events and the divergence that had opened up was a permanent departure from the historical equilibrium. Due to their failure in identifying these events as Black Swan events LTCM had to face huge losses and as a result finally went bankrupt.

This real life example clearly demonstrates that market events are extremely important factors to be considered while analysing any divergence in correlated stocks.

Link to code: Pairs Trading code

Bitcoin, Blockchain, and NFTs

History of Money

Lets's come to business and talk about money. What is money? Money is nothing but a social construct! Money, if we think about it, is just small pieces of metal or paper passed from person to person in exchange for anything in the world. Now as technology has advanced, we no longer require anything solid. Money is now a never-ending stream of ones and zeros circling the world. And we have evolved from commodity money to digital money.

Current problems in the system followed

- **Centralization:**

- Centralization refers to the control of currency supply by a single authority. It is both a good and a bad thing for governments to get involved in money. But, Manipulation of the money supply has both short- and long-term repercussions and Inflation is one of them. The concept is not new. We use digital currency every time we use internet banking to conduct a transaction

- **Fractional Reserve Banking:**
  - It refers to the practice of lending out significantly more money than a bank has in cash on hand. It's a type of digital currency generated by banks. So It's all about debt. This is an element of the money-making process. The entire system is built on the concept of trust.
- **Trust:**
  - Belief in the bank's financial stability and having faith in the debtor's ability to pay back the obligation. Almost none of the paper money we think we have in our bank account exists. So, When we give a few people authority over large sums of money, they will use it to their advantage. Any data or information that is transferred can be changed.
- **Hefty fee charges:**
  - We rely on third parties in finance, such as banks and credit card firms. They keep track of money as it flows from one account to another and charge us a hefty fee. We have faith in the accuracy of their digital ledgers of credits and debits.

What if a technical breakthrough made it possible for everybody anywhere in the globe to be their bank? To develop money that is tax- and fee-free.

What is Bitcoin?

Bitcoin is one of the most original and elegant solutions that has emerged recently. It was the first to introduce the notion of decentralized bookkeeping, which is now known as blockchain technology, and it is a tremendously powerful concept. So, Bitcoin is a lot of different things. It is an open-source computer program that is free to use. Anyone can look at the code for this software and start using it. A network is

formed by the machines that run the Bitcoin application. The network follows a set of regulations, and its main purpose is to allow users to make electronic payments. Bitcoin

is the currency used to make these transactions.

Why Bitcoin?

Bitcoin was established to allow consumers to send and receive online payments without having to rely on a trusted banking institution or any other type of central intermediary. We need some kind of utility that is equally accessible to all people.

So, it believes in the democracy of Money!

If there is no central intermediary:

- No question of a corrupt central authority
- The cost needed to sustain and trust that intermediary is removed.
- More security.

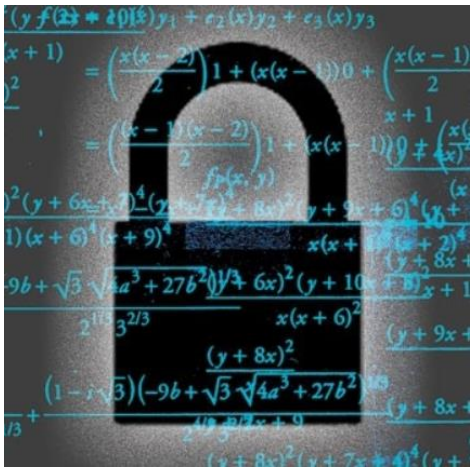
Decentralized System

Conditions for a Decentralised system:

- People can make and receive payments.
- There should be no central third party at all.
- The system should be viable enough to avoid any errors and fraud.

Cryptography

Bitcoin moves the burden of trust from institutions to numbers. So, the Bitcoin protocol makes heavy use of cryptography. Cryptography is the study of secure communications techniques that allow only the sender and recipients of a message to read its contents. If we are to understand Bitcoin, we need to understand two concepts from cryptography, Hashing, and Digital Signatures.



Hashing

A hash is a mathematical function that turns an arbitrary-length input into a fixed-length encrypted output. In simpler words, a hash function takes some data and produces a unique shorter summary or fingerprint of it. It takes input, divides it into equal bite-sized packets, and then repeatedly jumbles each of those packets. Some of the Hashing algorithms are MD5, SHA-1, and SHA-256.

Some properties of any good hash algorithm are:

- **One-way:** You cannot restore the document from its given hash. That is, you cannot decrypt the hash in any way.
- **Deterministic:** Hash value for a given document always remains the same no matter how many times we run the algorithm.
- **Fast Computation:** Now this is a tricky one. The algorithm should be fast enough to churn through a lot of data but not too fast then it will be easy to break.
- **Avalanche Effect:** Any slight change in data will result in a complete change of the calculated hash which is not at all predictable.
- **Withstand collisions:** Even after a lot of variations in the hash, it is limited. But the amount of data available is infinite. We can understand it using Pigeonhole Principle. It states that, If n pigeons are placed in m containers, with n>m, then at least one container must contain more than one pigeon. So with very little probability, two different documents may have the same hash value, which is fine! But, a hacker can forge documents by artificially creating a collision and our algorithm should be designed in a way that it doesn't allow such artificial collisions.

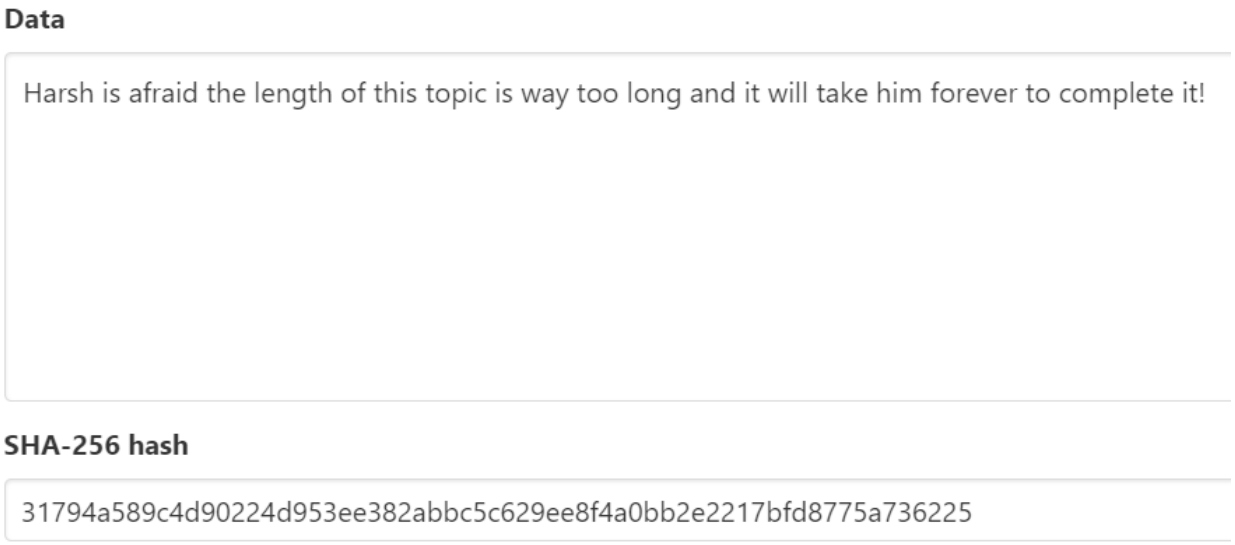
SHA-256

SHA stands for Secure Hashing Algorithm and 256 is the number of bits it takes up in memory. And, it is 64 characters long hexadecimal hash. This hashing algorithm works for any type of digital input. Complete data is represented in ones and zeroes using their corresponding ASCII Table and these operations are combined to generate the compound movements of bits.

Basic operations that happen in SHA-256 are:

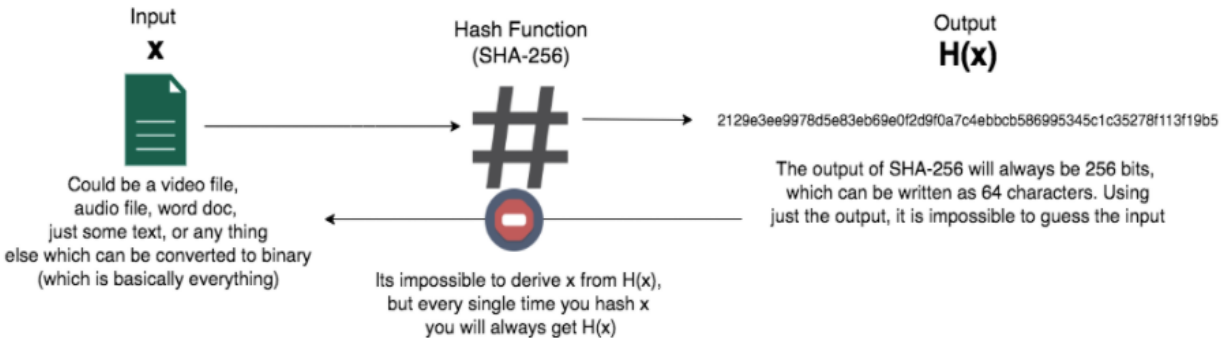
- Bit right-shift operation
- Rotational bitwise operation
- XOR
- Binary addition
-

A basic SHA-256 hash will look like this-



Use Cases of Hashing:

When you sign up for any website, they (hopefully) never store your actual password as plain text in their database. They store a hash of your password instead. Every time you enter your password on a page, the website takes the characters you entered, hashes them, and compares the resulting hash with the hash stored in the website's database. If the two hashes match, the website knows you are the real user and it logs you in. This way, if the website database ever gets hacked, the hackers will not get all of the passwords of the users, only the hashes of the passwords. But due to the increase in computation powers SHA-256 is no longer continued for preserving passwords. Another biggest use case of it is in Blockchain!



Key pairs

- **Private Key:** It's a unique number that's created as part of a cryptographic keypair. It is always accompanied by a public key. The private key can be used as a password to verify that you own a certain public key.
- **Public Key:** It is always accompanied by a private key. The public key is your primary public identity in Bitcoin. You can provide your public key to anyone, and they will be able to transfer you bitcoin using it. However, to spend bitcoin connected with your public key, you must generate a valid signature in your transaction to establish that you control the accompanying private key.

Anybody can independently derive a unique key pair by using some special algebra which is very useful.

Digital Signatures

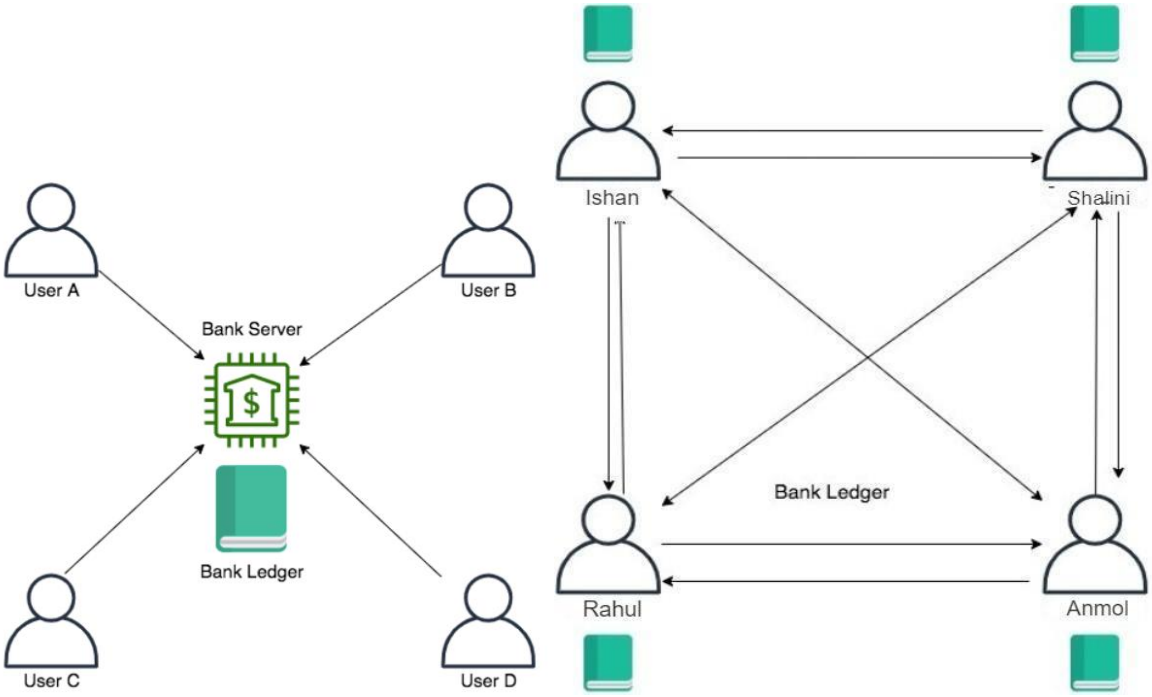
- **Signing messages:** Signing a message in cryptography is analogous to putting a stamp on an envelope.
  Inputs: Message, Private Key -----> Signing Function -------> Output: Signature
- **Verifying messages:** Verifying a signature allows you to check whether a given signature was created by the right person.
  Inputs: Message, Signature, Public Key ----> Signature Verification Function ---> Output: True/False

Uses of Digital Signature

It is useful for ensuring secure and reliable communications - and this is exactly why digital signatures are used extensively throughout the Internet. In fact, digital signatures are working in your browser at this very moment. If you look at the URL bar of your browser, you will probably see a lock icon next to it. Each website effectively has its legal public key. When your browser shows you the lock icon, it is telling you that you are communicating with a party can that prove they own the corresponding private key.

How do payments work?

Let's look at the 2 scenarios.



A B

Everyone is familiar with the first one so let's discuss the second one. Any computer running the Bitcoin program is called a node. So instead of having one central bank server, we now have four individual Bitcoin nodes run by Ishan, Shalini, Rahul, and Anmol.

Consensus

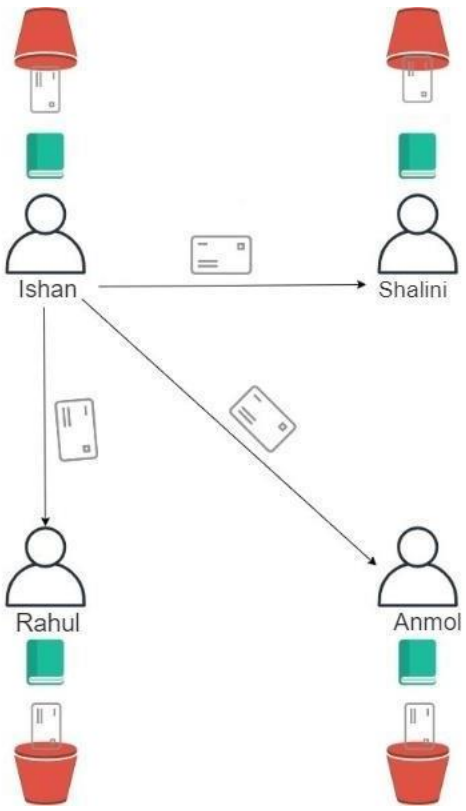
One of the problems with such a decentralized architecture is consensus i.e. how can we ensure that each ledger in this system is consistent with the others? There is no point if everybody has ledgers with different values - nobody will trust the system and it will be useless for payments.

To understand consistency first let us understand a bitcoin transaction.

Bitcoin Transaction

Simplified structure of a transaction in Bitcoin:

- **Sender:** Public key of the sender
- **Recipient:** Public key of the recipient
- **Amount:** Amount of bitcoin to be sent
- **Mining Fee** (Later)
- **Signature:** This is the result of putting the first four fields in a signing function along with the sender's private key.



These transactions get added to the ledgers once every ten minutes on average. But these transactions don't get directly added to the ledger - they are first batched into groups of transactions called 'blocks'.

1. Ishan creates a transaction and sends it out to everyone.
2. Everybody puts the transaction in their 'memory pool' bucket.

Ishan needs to make sure that this transaction is included in everybody's copy of the ledger. So Ishan 'broadcasts' his signature out to the network. So like when we use torrent, it automatically discovers other 'seeders' and gets files from them. Similarly, when

the signature is broadcasted on the network all the computers that are running the bitcoin program receive the transaction.

Before putting transactions in the memory pool of each bucket, we need to verify a few things:

➔ **Is all the necessary information included here?**

- This can be verified very easily by all the nodes.

➔ **Does the sender have the amount in his Bitcoin balance?**

- Imagine Ishan's public key is like his email address. Throughout the entire ledger, transactions are going to and from this particular email address, so everyone can just check their internal ledger to see the final balance against Ishan's email address. So that we can make sure Ishan has at least that much bitcoin in his wallet.

➔ **Is the transaction properly signed?**

- The signature allows all the nodes to check that Ishan has properly authorized the transaction using his specific password. This is how Bitcoin uses digital signatures.

If this transaction meets all of these criteria, then it is considered valid, and the nodes receiving it will put it into a bucket called a memory pool.

In reality, not every transaction is received by every node due to maybe bad internet So, in reality, each node's mempool could be different from their peers.

Bitcoin Mining

Bitcoin groups transactions into 'blocks'. Each block contains ~2000 transactions, and ledger entries are always made one block at a time. But how do nodes decide which transactions to put in the block? We can see that there is a mining fee associated with each transaction.

Mining Fee: The sender of a transaction has set aside some fee that he is willing to pay to the miner for his transaction to get through onto the ledger.

But still, some questions need to be answered.

1. **Who is a miner?**

- Any Bitcoin node that is working to try and add transactions to the ledger is a miner.

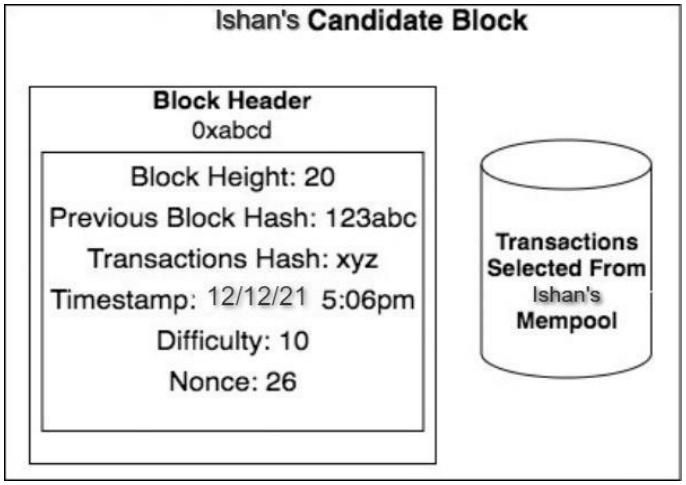
2. **Why would anyone spend so much computation power to work so hard?**

- **Mining Fee:** Since the number of transactions that can be added in a block is limited, the best thing that any miner would do is first try to process all the transactions with the highest mining fees so that he gets the maximum benefit out of it.
- **Block Reward:** Every time a new block is mined, the miner who solved the puzzle for that particular block gets 6.25 newly created bitcoin. So, mining is the only way to mint new bitcoins. In May 2020, miners stood to earn 6.25 bitcoin for every new block. Block rewards for Bitcoin miners will continue to be halved every four years until the final bitcoin is mined. Current estimates for mining of the final bitcoin put that date somewhere in February 2140. So, finally, there will be only 21 million bitcoins in circulation. By the time the supply of new bitcoin reaches 0, mining fees may be able to compensate for the loss of the block reward.

3. **Which one of the several miners gets to decide which transactions go into a block? Who gets the mining fee? Is it just equally split between all miners?**

- **Mining Puzzle:** Each miner/node creates his/her candidate block and whoever first solves the mining puzzle gets the chance to add that block into the ledger winning all the mining fee and block reward.

Too many new terms! But, don't worry we'll cover them all :)



Candidate Block

It not only includes all the transactions but also Block Header

Block Header

- **The Block Height**
  - Each block in Bitcoin has a different vertical 'height' based on when it was added to the ledger. The genesis block, mined by Satoshi, has a Block Height of 0. The next block has a Block Height of 1. The Block Height for the next one is 2, and so on. The most recent Bitcoin Block Height is 713378 at the time of writing.
- **The Previous Block Hash**(Later)
- **Transactions Hash**
  - The transaction hash is the sum of all transactions in a bitcoin block. So, when Ishan composes his candidate block, he will take the transactions he wants to include from his mempool, organize them in a specific order, and hash them all together so that they may be represented in a single short hash in the block header. This has the advantage of allowing you to compare two sets of transactions rapidly by computing their transaction hashes.

- **Timestamp**
  - This is the date and time at which the miner claims to have created the block.
- **Difficulty**(Later)
- **Nonce**(Have patience;)

Difficulty

Phrase(Input): Harsh is afraid the length of this topic is way too long and it will take him forever to complete it!

Output(Hash): 31794a589c4d90224d953ee382abbc5c629ee8f4a0bb2e2217bfd8775a736225

Output(First 10 digits in Binary): 1100010111….

In the first 10 bits of the hash of our phrase, there are 6 1s and 4 0s. So when we hash something each bit has an almost 50% chance of being a 1 or being a 0.

Now,

P(First bit as 0) = 0.5

P(First 2 bits as 0) = 0.25 …..

P(First 10 bits as 0) = 0.098%

In Bitcoin, the difficulty number tells us how many leading 0s we need in the hash of the block header for that block to successfully solve the mining puzzle. So, if difficulty is set to 10 then the chances to get a hash with the initial 10 bits as 0 in the first attempt is as low as 0.098%.

Nonce

“Number only used once”

➔ On average, a miner would need to make about 1020 different block header hashes before stumbling upon one which begins with 10 zeros. In Ishan’s candidate block, the nonce is 26. So that means that

if you include the number 26 with all the other fields in the block header and hash them, the output hash has ten leading 0s.

- ➔ The miners change the nonce when attempting a new solution to the mining puzzle because the other fields in the block header are either fixed (such as the block height) or time-consuming to compute (the transaction hash).
- ➔ The nonce exists solely for the purpose of helping miners come up with different solutions to the mining puzzle.

There will always be some nonce value that makes a miner's individual candidate block achieve the right number of 0s to solve the difficulty. This magic nonce number is what all miners are searching for!

Success Mantra for a Bitcoin Miner

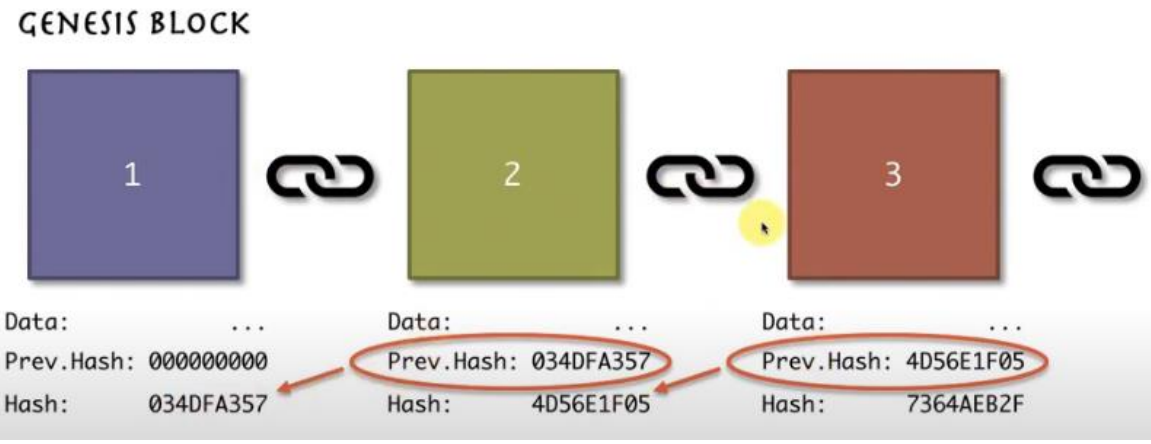
Each computer is trying to generate the one lucky hash out of quintillions of hashes that happen to begin with a large number of zeros. This is how hashes are used in Bitcoin's mining puzzle.

Luck will always be not on your side.

So your success depends on your hash power - how many hashes can you perform in a second?

- Choosing places with low energy costs.
- Designing the most efficient mining chips.
- Crowdsourcing computing power into aggregations called mining pools

Previous Block Hash



Let's say you've tampered with the 2nd block. The hash of the block is also changed as a result of this. As a result, block 3 and all subsequent blocks will be invalid since they will no longer store a valid hash of the prior block. As a result, altering a single block invalidates all subsequent blocks.

Final Verification!

Let's say Ishan has won the mining puzzle and Ishan's candidate block is sent to every miner to make sure it is valid.

- Does the block have a valid previous block hash?
- When we hash all of the transactions in the given block, does the resulting transaction hash tally up with the transaction hash given in the block header?
- Are all the other fields in the block header correct?
- When we hash the block header, do you get a hash that satisfies the difficulty criteria?

When each miner verifies, they append Ishan's candidate block to their local copy of the ledger and Ishan finally wins all the rewards :)

Performing a hash is easy to do - so checking that the hash value is correct or not is extremely quick, even though generating the right hash value for Ishan and the other miners was very unpredictable and hard!

Problems faced

★ **Why would even Shalini, Rahul, and Anmol even add Ishan's block in their local copy of the ledger if it doesn't benefit them?**

- **Rule:** If some other miner sends you a valid candidate block, then add it to your local copy of the ledger immediately for consistency.

★ **What if Ishan and Shalini both come up with a valid block at the same time?**

- **Rule:** Always believe the version of the ledger which has the highest Block Height.

Remember, miners don't know each other and they are not friends!

Proof-of-work

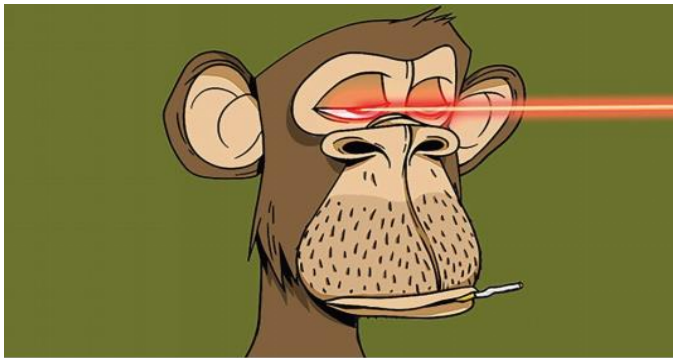
If we have a problem of having two different versions of the ledger called a fork in the ledger then Which version of the ledger is correct?

- If a version of the ledger has more blocks, that means more work has gone into generating that version of the ledger. That implies that the version of the ledger has the most public support since it took the most effort to create. This is known as the consensus mechanism in Bitcoin also called proof-of-work.

Reorganization of the ledger

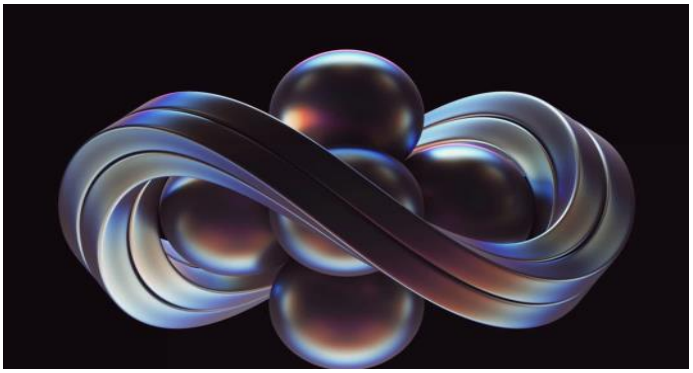
If there are conflicting copies of the ledger at a given block height, some miners had to reorganize their ledgers once a certain fork of the ledger became the longest.

Non-Fungible Token



Introduction

Non-fungible tokens, or NFTs, are blockchain-based cryptographic assets having unique identification codes and information that separate them from one another. NFTs can be linked to digital assets that can be reproduced, such as images, movies, and audio. NFTs employ a digital ledger to offer a public certificate of authenticity or evidence of ownership, but the underlying digital data may be shared or copied freely. NFTs are distinguished from blockchain cryptocurrencies such as Bitcoin by their lack of interchangeability (fungibility). This is in contrast to fungible tokens, such as cryptocurrencies, which are identical to one another and may thus be used as a medium for economic transactions.



History:

Quantum, the first known "NFT," was constructed in May 2014 by Kevin McCoy and Anil Dash, and consisted of a video clip prepared by McCoy's wife Jennifer. During a live presentation for the Seven on Seven conferences at the New Museum in New York City, McCoy registered the video on the Namecoin network and sold it to Dash for $4. The technology was dubbed "monetized graphics" by McCoy and Dash. Following the development of many NFT initiatives that year, the word "NFT" only acquired currency with the ERC-721 standard, which was initially suggested in 2017 via the Ethereum GitHub. Curio Cards, CryptoPunks (a project to trade unique cartoon figures created by the American company Larva Labs on the Ethereum blockchain), and the Decentraland platform are examples of these.

The popularity of CryptoKitties, an online game in which participants adopt and exchange virtual kittens, raised public awareness of NFTs. The initiative quickly went viral, garnering a $12.5 million investment, with some kittens selling for more than $100,000 apiece. Decentraland, a blockchain-based virtual world that originally sold tokens in August 2017, raised $26 million in an initial coin offering in 2018, and as of September 2018, had a $20

million internal economy. The NFT market tripled in value to $250 million in 2020, owing to strong expansion. More than $200 million was spent on NFTs in the first three months of 2021.

Understanding NFT's

- NFTs alter the crypto paradigm by making each token distinct and irreplaceable, making it impossible for one non-fungible token to be identical to another.
- They are digital representations of assets that have been compared to digital passports since each token carries a unique, non-transferable identity that allows it to be distinguished from other tokens.
- Owners can also add metadata(for eg. Signature) or attributes pertaining to the asset in NFTs. NFTs are having a very unique property of extensibility which means that two NFTs can be combined to get or breed a third(new) NFT.

First NFT to be sold by a historic

Why NFTs:

NFTs can be used to represent real-world items like artwork and real estate. "Tokenizing" these real-world tangible assets allow them to be bought, sold, and traded more efficiently while reducing the probability of fraud. Non-fungible tokens are also excellent for identity management. They can be used to represent people's identities, property rights, and more. NFTs also provide the opportunity for multiple sharing of physical assets like real estate or digital art. The emergence of new markets and kinds of investing is the most intriguing opportunity for NFTs.

MARTINGALES

Introduction

Have you ever considered it possible having a gambling technique that you can establish is always a winner, a fair game, and yet might bankrupt you at the same time? The martingale theory and betting strategy essentially demonstrates this.

The martingale probability theory is a mathematical representation of a fair game, in which there is equal chance of winning or losing. It's a process in which today's events serve as the best predictor of what will happen tomorrow. Originally, a martingale was a class of betting strategies. Despite the fact that they are commonly recognized to lead to bankruptcy, they are still widely used.

When the Martingale Strategy is used, the transaction size is doubled every time a loss occurs. A common situation for the technique is to try to trade a result that has a 50% chance of occurring. We also call this strategy a zero-expectation scenario. Here we are doubling our stake each time we lose, so we say our multiplier is 2. You can use 3 or other numbers as well as your multiplier.

Let us consider example of gambling:

Given, (1) You are doubling the stake every time you lose.

(2) You will stop playing the game if you win.

(3) You are starting the game with 1$.

Calculate the profit (or loss) that the gambler will face after n^{th} bet.

Solution:

Case (1): You lose all first n bets, then the loss you will be gone through will be-

$W = -(1+2+2^2+2^3 \ldots\ldots + 2^{n-1})$

Case (2): You win at the n^{th} bet-

$$W = -(1+2+2^2+2^3 \ldots\ldots + 2^{n-2}) + 2^{n-1}$$

$$= -(2^{n-1}-1) + 2^{n-1}$$

$$= 1$$

So, you can see in this case, either we are going bankrupt (probability of which is very less) or we are winning 1$. You must have got an intuition that we don't intend to play the game with Martingale strategy from the start but we use Martingale to cover our losses after loss.

Filtration

Now that you have gotten a basic idea of a martingale, the next step is to define it mathematically. To do that, it is crucial to understand the concept of filtration.

A filtration is an increasing sequence of σ - algebras on a measurable probability space. Filtrations represent the information about an experiment that has been revealed up to a certain time t. a σ-algebra (also σ-field) on a set X is a collection Σ of subsets of X, is closed under complement,and is closed under countable unions and countable interactions.

Looking at an example for a better understanding:

Consider an experiment where a coin is tossed 3 times in 3 seconds.The sample space for our experiment will be a set of all the possible outcomes -

$$\Omega = \{HHH, HHT, HTH, HTT, THH, THT, TTH, TTT\}$$

As the sample space is finite, we will take the σ - algebra, F (set of all the events possible), as the power set of Ω - $F = 2^{\Omega}$

Now starting with time t = 0,

The coin has not been tossed yet. So, we have no extra information available to us. All we know is our sample space for the experiment. Given an outcome w, the only events from the σ - algebra that has been revealed to us (i.e., which we know have surely occurred or not) are Ω and ϕ. This is because we already know whether w will belong in Ω or not. Making a set of all the events that have been revealed,

$$F_0 = \{\phi, \Omega\}$$

We can see that F0 itself is a σ - algebra. This will be the first σ - algebra of our filtration.

At time t = 1,

The coin has been tossed once, and we have learnt some more information about the experiment. This extra information helps us reveal two new events (apart from Ω and ϕ). The event of the first toss is a head,

$$A_H = \{HHH, HHT, HTH, HTT\}$$

And the event of the first toss is a tail,

$$A_T = \{THH, THT, TTH, TTT\}$$

That is, given an outcome w, we will already know whether the first toss was an H or T, thus revealing the events A_H and A_T. The set of all the events that have been revealed will now be,

$$F_1 = \{\phi, \Omega, A_H, A_T\}$$

Notice that F1 is a σ - algebra. This will be the next σ - algebra of our filtration.

Similarly, at time t = 2,

The coin has now been tossed twice. The new information will reveal many new events. Some of them are, the event of the first two tosses is a head,

$$A_{HH} = \{HHH, HHT\}$$

the event of the first toss is a head and the second is a tail,

$$A_{HT} = \{HTH, HTT\}$$

and so on.Note that the unions and complements of the revealed events will also be revealed. The set of all the events that have been revealed will now be,

$$F_2 = \{\phi, \Omega, A_H, A_T, A_{HH}, A_{HT}, A_{TH}, A_{TT}, A^C_{HH}, A^C_{HT}, A^C_{TH}, A^C_{TT},$$

$$A_{HH} \cup A_{HT}, A_{TH} \cup A_{TT}, A_{HT} \cup A_{TH}, A_{HH} \cup A_{TT}\}$$

Notice that F2 is a σ - algebra. This will be the next σ - algebra of our filtration.

Finally, at time t = 3,

The experiment is complete. For any outcome w, we now have information about all tosses. This means all the events in the σ - algebra have been revealed. The set of all the events that have been revealed will now be,

$$F_3 = F$$

We can observe that these σ - algebras, F_0, F_1, F_2, and F_3 are such that,

$$F_0 \subset F_1 \subset F_2 \subset F_3 = F$$

This family of σ - algebras, $\{F_n\}_{0 \le n \le 3}$, can be called a filtration.

Now that we have a clear understanding of filtrations, let us give a more formal definition:

Given a probability space (Ω, F, P) and some T > 0, For $0 \le t \le T$ there exists a σ - algebra, F(t) such that

$$F(t) \subset F$$

and for some $0 \le s, t \le T$ and $s \le t$, the σ - algebras F(s) and F(t) are such that

$$F(s) \subseteq F(t).$$

Then the family of σ-algebras → {F(t)} $0 \le t \le T$ is called a filtration that is associated with the probability space (Ω, F, P).

Martingales

You may have picked up a bit on Martingales in the introduction part. However, once we've covered filtration, we can move on to the important aspect of Martingales, which includes the basic mathematics underlying it and its applications.

Martingales is a stochastic process (sequence of random variables) in which the conditional expectation of the next value in the sequence, regardless of all previous values, is equal to the current value of the sequence.

To offer a simple illustration, imagine tossing a coin in a sequence: if it comes up head, you get one dollar; if it comes up tail, you lose one dollar. Let's say you have $5 after 100 such tosses. As a result, your expected value after the 101^{st} coin is $5. Isn't it straightforward? This is the core concept of Martingales; we should grasp the basic mathematics underlying it to have delve more into it.

(1) Say, $\{F_n\}_{n\geq 0}$ is an increasing sequence of σ−algebras (filtration) in a probability space (Ω, F, P). Let X_0, X_1, X_2 …. X_n is an adapted sequence ($X_n \in F_n$) of integrable real valued random variable such that $E(X_{n+1}) < \infty$, the sequence X_0, X_1, X_2 …. X_n is said to be a Martingale relative to the filtration $\{F_n\}_{n\geq 0}$ if

$$E(X_{n+1} \mid F_n) = X_n \qquad \text{(Martingale condition)}$$

(2) We have, $E(X_{n+1} \mid F_n) = X_n$.

$$E(X_{n+k} \mid F_n) = E((X_{n+k} \mid F_{n+k-1}) \mid F_n) \qquad \text{(Using Tower's property)}$$

$$= E(X_{n+k-1} \mid F_n) \qquad \text{(Using Martingale)}$$

.

.

.

$$= E(X_{n+1} \mid F_n)$$

$$= X_n$$

(3) The expectation of next value in sequence to be equal to current value, there are several types of martingales sequences, the most basic one is sums of independent, mean zero random variables. Let Y_1, Y_2... be such a series; then with following conditions,

$$E(Y_i) = 0$$

$$F_n = \sigma(Y_1, Y_2, Y_3 \ldots . Y_n)$$

the partial sums sequence:

$$X_n = Y_1 + Y_2 + Y_3 \ldots Y_n$$

is a martingale in comparison to the natural filtering caused by the variables Y_n. The linearity and stability features, as well as the independence law for conditional expectation, make this easy to verify:

$$E(X_{n+1} \mid F_n) = E(X_n + Y_{n+1} \mid F_n)$$

$$= E(X_n \mid F_n) + E(Y_{n+1} \mid F_n) \qquad \text{(Linearity)}$$

$$= X_n + E(Y_{n+1})$$

$$= X_n.$$

Example:

We went through a 3-coin toss example in the filtration part, now let's do the same thing in Martingales. If we apply the random variables and probability terms to the three coin-toss examples, we get X_n, which is the total amount won/lost in n bets. The profit/loss on the nth bet is represented by Y_n. F_n is a collection of values that may be obtained via the use of n bets. So, if we have been given a F_n that has led us to an X_n value, we may claim that our value X will not change after one more bet since the Y_{n+1} expectation will be zero owing to fair toss.

Now, once we have done the basic mathematics of the martingales we can move towards the applications of Martingales in stock market.

Why martingale in stock market?

(1) According to the Efficient Market Hypothesis, beating the market on a risk-adjusted basis is impossible since the market should only react to fresh information. As a result, trading cannot be done in such a way that you never gain or lose with a positive probability.

(2) In the fundamental theorem of finance, the acceptable probability is known as risk-neutral probability, which is a measure of equivalent Martingale.

This is bit hard to understand if you are new to the stock world, but there is very simple meaning behind this i.e., This suggests that, regardless of how much money a firm is losing at the time, an investor should not give up, but rather keep investing in the hope that perseverance will eventually pay off. Before using the martingales in real world one should be aware of its risk. The majority of investors do not employ martingales as their primary investment strategy, but rather combine it with other momentum techniques and then use it.

Application of Martingles in Stock Market

(1) Most investors do not strictly follow the Martingale method since it is not the greatest way to invest, but every investor indirectly utilises the Martingale approach to lower the average price of the stocks he owns down if he learns that the stock price is going to rise in a short period of time.

(2) Martingale trading is a common forex trading method. The fact that currencies, unlike equities, seldom fall to zero, is one of the reasons why the martingale technique is so popular in the currency market. Although businesses may readily go bankrupt, most countries do so on their own volition. A currency's value will fluctuate from time to time. Even in extreme circumstances of depreciation, the currency's value seldom falls to zero.

Implementation of Martingles

Let us see what we have,

Initially the investor will buy one stock. The primary premise behind using Martingales to generate an idea for this stock is that if the current day's stock price is less than 1.5% lower than the previous day's, the investor should double his investment at the end of the day. Investment will rise, and investors might be able to profit from the market even if there are fewer victories.

Dataset:

Apple data containing for date and price (closing) for the time interval 6/3/2019 to 31/12/2019

Simple Martingale strategy:

Signal is a indication of action to the investor if he has buy, hold. If current day's price is 1.5% greater than previous day's close then +1, if current day's price is 1.5% less than previous day's close then -1, Otherwise 0.

```
[11] signal = []
     for i in range(len(df)):
       if i ==0:
         signal.append(0)
       elif df['price'][i] > (1.015)*df['previous_close'][i]:
         signal.append(1)
       elif df['price'][i] < (0.985)*df['previous_close'][i]:
         signal.append(-1)
       else:
         signal.append(0)
```

```
[12] df['signal'] = signal      #signal is for next day
```

Quantity is number of shares the investor own on the day.

```
[13] quantity = [1,1]
     for i in range(2, len(df)):
       if df['signal'][i-1]==-1:
         quantity.append(quantity[i-1]*2)
       else:
         quantity.append(quantity[i-1])
```

```
[14] df['quantity']=quantity          #quantity is number of stocks owned by you on that day
     df
```

invested money is total money invested from the starting date to till date.

```
[ ] invested = ['NAN',df['price'][0]]
    for i in range(2, len(df)):
      if df['quantity'][i]==df['quantity'][i-1]:
        invested.append(invested[i-1])
      else:
        invested.append(invested[i-1]+(df['previous_close'][i]*(df['quantity'][i]-df['quantity'][i-1])))
    invested
```

Returns are returns on the given date and returns_in_per are returns on the given date in percentage

```
[18] returns =[]
     returns_per = []
     for i in range(len(df)):
       if i == 0:
         returns.append('NAN')
         returns_per.append('NAN')
       else:
         returns.append((df['price'][i]-df['previous_close'][i])*df['quantity'][i])
         returns_per.append(returns[i]/(df['previous_close'][i]*df['quantity'][i]))
     df['returns'] = returns
     df['returns_in_per'] = returns_per
```

cumulative returns is net returns from the starting date to till date, cumulative returns in percentage is total returns from starting date to till date in percentage.

```
[20] cumulative_returns = ['NAN']
     cumulative_returns_per = ['NAN']
     net_returns = 0
     for i in range(1, len(df)):
       net_returns+=df['returns'][i]
       cumulative_returns.append(net_returns)
       cumulative_returns_per.append((net_returns*100)/df['invested_money'][i])


     df['cumulative_returns'] = cumulative_returns
     df['cumulative_returns_in_per'] = cumulative_returns_per
```

As you can see, the cumulative returns in percentage for the period of 6 months is 9.57 for total invested money 312360.7804$.

Code and dataset:

Dataset: File

Code: Code

Final results: Final file

HIDDEN MARKOV MODEL

A Hidden Markov Model (HMM) is a statistical model which is also used in machine learning. It can be used to describe the evolution of observable events that depend on internal factors, which are not directly observable.

A simple example of an HMM is predicting the weather (hidden variable) based on the type of clothes that someone wears (observed).

By forecasting weather precisely we can prevent and overcome many hazards that could lead to a great loss to a nation. HMM is used to predict weather using the Markov Chain property. The training of the model and probability of occurrence of an event is calculated by observing weather data for the last 21 years. The data is firstly categorized based on standard values set apart. The result obtained shows that our model is reliable and works very well in predicting the weather for the next 5 days based on today's weather pattern.

What is Markov Model?

The Markov process assumption is simply that the “future is independent of the past given the present”. In other words, assuming we know our present state, we do not need any other historical information to predict the future state.

Important terms and definitions:

1. **Transition data** — the probability of transitioning to a new state conditioned on a present state.
2. **Emission data** — the probability of transitioning to an observed state conditioned on a hidden state.

3. **Initial state information** — the initial probability of transitioning to a hidden state. This can also be looked at as the prior probability.

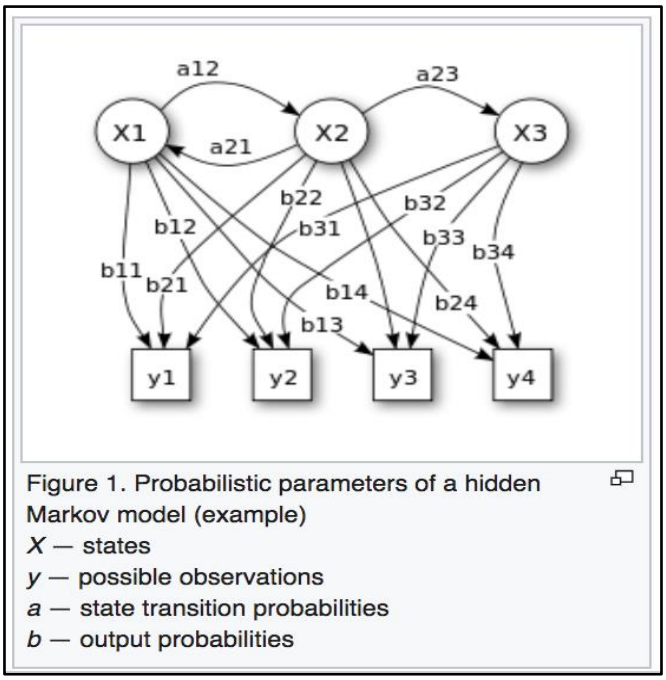


Figure 1. Probabilistic parameters of a hidden Markov model (example)
X — states
y — possible observations
a — state transition probabilities
b — output probabilities

Need for use:

1. Modeling sequences of data.
2. Used in stock prices, credit scoring, and webpage visits.
3. In Computational Genomic Annotation. It includes structural annotation for genes and other functional elements and functional annotations for assigning functions to the predicted functional elements.

Whether HMMs are supervised or unsupervised?

- Unsupervised: When we want to discover a pattern or model a distribution but don't have any labeled data.
- Supervised: When we do have labels and want to be able to accurately predict the labels.

- HMMs are used for modelling sequences:
  - x(1), x(2), x(3), x(4).....x(t),....,x(T).
  - No labels are available.

Classification (an example of HMM):

- How can we classify the voices into male and female categories using HMM?
- The key Idea is Bayes' rule.
- We model P(x | male) and P(x | female)
- P(x | male): Collect all-male data and train an HMM
- P(x | female): Collect all-female data and train another HMM.

Applications of Hidden Markov Model:

Hidden Markov models are known for their applications to thermodynamics, statistical mechanics, physics, chemistry, economics, finance, signal processing, information theory, pattern recognition - such as speech, handwriting, gesture recognition, part-of-speech tagging, musical score following, partial discharges

Applications, where the HMM can be used, have sequential data like time series data, audio, and video data, and text data or NLP data.

Conclusion over its Nature:

- Hidden Markov Models are **Unsupervised** in Nature.
- It is used for Classification using Bayes' rule and creating a separating model for each class, and choosing the class that gives us the maximum posterior probability.

Markov Property:

- When tomorrow's weather depends on today's but not on yesterday's weather
- When the next word in the sentence depends only on the previous word in the sentence but not on any other words.
- In general, our assumption is that the current state depends only upon the previous state or the next state depends only upon the current state.

Another way of saying is:

The distribution of state at a time 't' depends only on the state at a time 't-1'.

- In general: 'states'
- State at time t: s(t)
- p(s(t) | s(t-1), s(t-2),.....,s(0)) = p(s(t) | s(t-1))

We want to model the joint probability i.e. probability of a sequence of states, which becomes (using chain rule of probability):

p(s4, s3, s2, s1) = p(s4 | s3, s2, s1) * p(s3, s2, s1)

= p(s4 | s3, s2, s1) * p(s3 | s2, s1) * p(s2, s1)

= p(s4 | s3, s2, s1) * p(s3 | s2, s1) * p(s2 | s1) * p(s1)

= p(s4 | s3) * p(s3 | s2) * p(s2 | s1) * p(s1)

(assuming Markov property)

Markov Models are often used to model the probabilities of different states and the rates of transition among them.

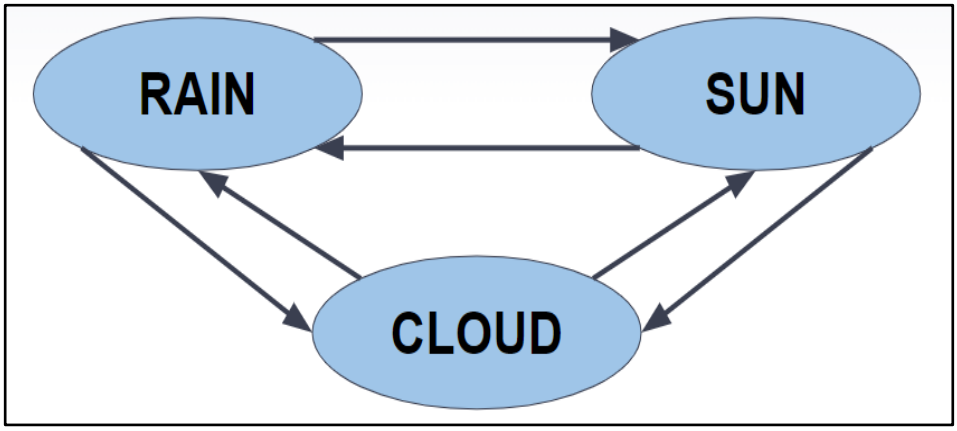
- First Order Markov: p(s(t) | s(t-1))

- Second Order Markov: p(s(t) | s(t-1), s(t-2))

MARKOV MODEL EXAMPLE:

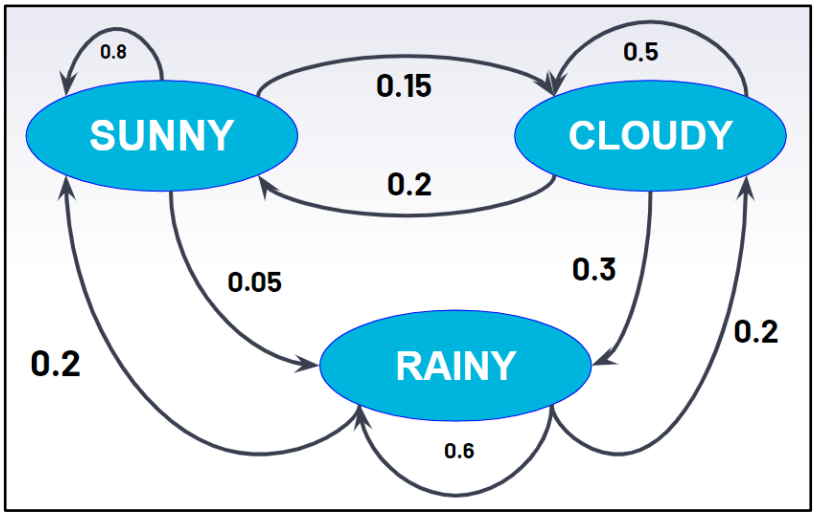
Weather Example:

3 states: SUN, RAIN, CLOUD



- Here, each node is a state and each edge is the probability of going from one state to the next state.

- This is 1st order Markov Model because weights depend only on the current state and it only affects the next state.



How many weights do we need to calculate?

Each state can go to each state, including itself.

M States —-----> M* M weights

A —-----> M X M Matrix

A(i, j) = p (s(t) = j | s(t-1) = i)

Constraints: A(i,:) must sum to 1; i = 1,....., M

Starting Position: Where do you start?

—--> we need to model p(s(0))

We usually store it in the symbol pi

1 X M row vector

EXAMPLE:

Q) What is the probability of the sequence? (sun, sun, rain, cloud)

p(sun, sun, rain, cloud) = p(cloud | rain) * p(rain | sun) * p(sun | sun) * p(sun)

= 0.2 * 0.05 * 0.8 * p(sun)

In general:

p(s(0),.....,s(T)) = π(s(0))

How is a Markov Model trained?

This can be done by using **maximum likelihood.**

For example: Suppose we have training data of 3 sentences.

1) I like dogs
2) I like cats
3) I love kangaroos

6 states: 0 = I, 1 = like, 2 = love, 3 = dogs, 4= cats, 5 = kangaroos

If we use maximum likelihood, then our initial state distribution is just one hundred percent probability.

Π = [1, 0, 0, 0, 0, 0]

p(like | I) = ⅔

p(love | I) = ⅓

p(dogs | like) = p(cats | like) = ½

p(kangaroos | love) = 1

Markov Chain:

A Markov chain is a mathematical system that experiences transitions from one state to another according to certain probabilistic rules. The defining characteristic of a Markov chain is that no matter how the process arrived at its present state, the possible future states are fixed.

Markov Chain: Example

Example: What is the probability of a sunny day 5 days from now?

P(sun(1)) = P(sun(1),sun(0)) + P(sun(1),rain(0)) + P(sun(1),cloud(0))

= P(sun(1)|sun(0)) 𝚷 (sun)

+

P(sun(1)|rain(0)) 𝚷 (rain)

+

P(sun(1)|cloud(0)) 𝚷 (cloud)

In General:

P(s(1)) = 𝚷 A

P(s(2)) = 𝚷 A A = 𝚷 A^2

P(s(t)) = 𝚷 A^t

Stationary Distribution

What if P(S) = P(S) A?

I.e. We end up with the same state distribution.

The state distribution never changes-> Stationery

=> This is just the eigenvalue problem where the eigenvalue is 1.

Issues associated with it:

- Eigenvalues are not unique -> make it sum to 1 so it's a proper distribution.
- P(s) is a row vector, typically a solver will solve Av = $\boldsymbol{\lambda}$v
  (so transpose it first)

Limiting Distribution:

It is the state distribution that you settle into after a very long time.

What is the final state distribution?

$\Pi_\infty = \Pi A^\infty$

(Called a limiting distribution or equilibrium distribution)

But $A^\infty A = A^\infty$

So, $\Pi_\infty = \Pi_\infty A$

This means Π is also a stationary distribution.

Conclusion:

- If I take an Equilibrium distribution multiplied by A and get the same distribution, then it's a stationary distribution

- So, All the Equilibrium distributions are stationary distributions, but vice-versa is not true.

FROM MARKOV MODEL TO HIDDEN MARKOV MODEL

- Unsupervised ML (Cluster Analysis) and Unsupervised DL (hidden/latent variables)
- K-Means Clustering, Gaussian Mixture models, principal components analysis.
- Observations are stochastic
- Hidden causes are a stochastic tool

HMM, Real-life Example:

Suppose you are at a carnival, magician has 2 coins hidden behind his back.

He will choose one coin to flip at random, you can only see the result of the coin flip (H/T)

H/T - space of observed values

Hidden State - Which coin it is.

(This is a stochastic/random process)

The intuition behind HMMs:

HMMs are probabilistic models. They allow us to compute the joint probability of a set of hidden states given a set of observed states. The hidden states are also referred to as latent states. Once we know the joint

probability of a sequence of hidden states, we determine the best possible sequence i.e. the sequence with the highest probability, and choose that sequence as the best sequence of hidden states.

HMM:

It has 3 parts: $\boldsymbol{\Pi}$, A, B

$\mathbb{\Pi}_i$: the probability of starting at state i

A(i, j) = probability of going to state j from state i.

B(j, k) = probability of observing symbol k in state j.

Example 1: Magician really likes coin 1, $\mathbb{\Pi}_1$ = 0.9

Example 2: Magician is fidgety, A(2,1) = 0.9

A(1, 2) = 0.9

Here, A is a **state transition matrix**

In HMM, the state themselves are hidden.

HMM for Stock Price Prediction:

The hidden Markov model has been widely used in the financial mathematics area to predict economic regimes or predict stock prices.

Choosing a number of hidden states for the HMM is a critical task. We use four

common criteria: the AIC, the BIC, the HQC, and the CAIC to evaluate the performances of HMM with different numbers of states. These criteria are suitable for HMM because, in the model training algorithm, the Baum–Welch Algorithm, the EM method was used to maximize the log-likelihood of the model. We limit the number of states from two to six to keep the model simple and feasible for stock prediction.

AIC = -2 ln(L) + 2k

BIC = -2 ln(L) + k ln(M)

HQC = -2 ln(L) + k ln(ln(M))

CAIC = -2 ln(L) + k(ln(M) + 1)

where L is the likelihood function for the model, M is the number of observation points, and k is the number of estimated parameters in the model.We assume that the distribution corresponding with each hidden state is a Gaussian distribution, therefore, the number of parameters, k, is formulated as k = N2 + 2N - 1, where N is numbers of states used in the HMM.

Independence Assumptions:

- More than just Markov's assumption
  The next state depends only on the previous state

- Observed 'k' depends only on state j.
  Does not depend on the state at any other time

  Does not depend on any other observation

Q) How to choose the number of hidden states:

- It is a hyperparameter
- It can be done by Cross-Validation

N training samples + N parameters —--> can achieve the perfect score

But this doesn't say anything about how well the model will perform on the unseen dataset.

How to avoid overfit and underfit?

- Leave some data out of training (validation set)
- Compute the Cost of the Validation set

Choose the number of hidden states that give the highest validation accuracy

The Kalman Filter

What is Kalman Filter?

Before we actually start discussing the essence of Kalman Filter, let us first present its formal definition:

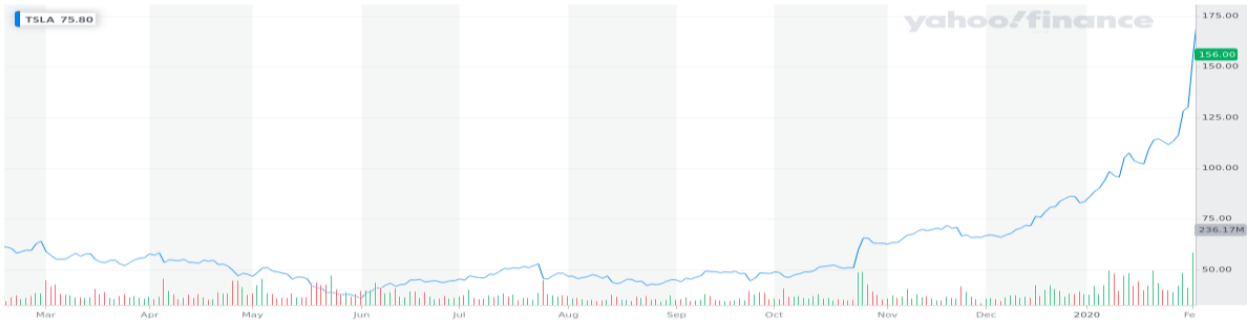
*"Kalman Filter is an **iterative** mathematical process that uses a set of equations and consecutive data inputs to quickly estimate the **true value** of the **measurement**."*

There are several important keywords in this definition. Let's talk about them.

- **Iterative**: It is important to remember that Kalman filter is an iterative process in its core. The output that we get in a certain iteration is used as input in the next. You may encounter vectorized implementation of Kalman Filter out there in the wild which, in a way, bypasses iteration but the core concepts remain the same.
- **True value and measurement**: When talking about Kalman Filters, we consider the measured and true value of an event as two very different quantities. We shall understand this in greater detail in the following example.
-

I did not see that coming

Let's say we are interested in the movement of a certain stock. We analyze its past prices and its trends.



From the graph, it is pretty evident that the stock is on a bullish trend. Suppose we are optimistic about the market and predict that the stock will continue its upward ascent. This seems to be a reasonable assumption to make. However, something happens which we did not anticipate.

Some external factor influences the market and the trend immediately reverses.

Under these circumstances, we find that a person with optimistic views on the market will have to suffer heavy losses. Moreover, the model which the person employs for predicting the future of the series may not give correct predictions until the downward trend is firmly established. During this period, the trader will suffer great losses. Our target is to minimize such losses due to unforeseeable circumstances.

Note that we are dealing with several quantities related to the price series. Firstly there is the **predicted value**. This is the value that we expected to attain before actually measuring the quantity. This is based on some prediction model that we have employed. This model can never be 100% accurate and therefore there is bound to be some error in our predicted value. Then there is the **measured value** which is the actual value that the series takes. In some other contexts where our measurement is based on experimental observations, there may be some errors in the measurement as well. In our current example, we can see that the measured value at any instance can be a possible outlier or may represent a short-term trend. Thus it may give wrong inferences about the long-term behaviour of the series. We can therefore say that there can be possible errors in measured value as well. Lastly we have our **estimated value**. Since we have errors in our predicted as well as measured value, we want to estimate the true underlying behaviour based on these values and their errors. This is our estimated value.

Self-correcting models

We essentially want a model that self-corrects itself when unforeseeable events occur. There are various models which accomplish this. The simplest among them is the **Simple Moving Average** (SMA). We simply take the average over a certain time period and expect the price to lie close to this average. This way, when the stock price falls down, the average also falls down and our predicted value also goes down. A similar thing happens when there is a sudden jump in stock prices. In this way SMA self-corrects the predicted value based on the stock prices.

So why can't we use SMA everywhere? Well, there are certain shortcomings of using SMA. The most prominent among them being slow reaction time. If there is a sudden reversal in trend, it takes a while for SMA to react to this change (until the trend reversal has been established). This is because SMA gives equal weightage to all data. As a result, older data dominates the newer data (which suggest a trend reversal) for a while.

So how do we overcome this? Well, one solution is to use **Exponential Moving Average** (EMA). In EMA, we give greater weight to the latest data and lesser weight to older data. This allows it to react quickly to sharp changes. However, there are still problems with using an EMA (or any moving average in general). Note that moving averages work over a certain time period. This time period is often arbitrary and is chosen by the trader as a hyperparameter. Thus by using a moving average, we are essentially introducing another hyperparameter to our strategy. This hyperparameter involves additional effort on the trader's end for tuning and has associated risks of overfitting.

What if we had a self-correcting model which reacts quickly to sharp changes and does not involve an additional hyperparameter?

You probably know where this is heading.

Let's talk Math

As mentioned earlier, Kalman filter is an iterative process. In each iteration there are three major steps.

1. Calculating the Kalman Gain

The Kalman Gain is the heart of the Kalman Filter. It decides how much weight must be given to the estimated and measured values in our current estimate. Mathematically, it can be computed using the following relation

$$KG = \frac{E_{Est}}{E_{Est} + E_{Mea}}$$

where,

KG = Kalman Gain

E_{Est} = Error in Estimation

E_{Mea} = Error in Measurement

2. Calculating current estimate

Once we have our Kalman Gain, we can calculate the current estimate. This can be calculated using the following formula:

$$Est_t = Pred_t + KG \cdot \left(Mea_t - Pred_t \right)$$

where,

Est_t = Estimated value at time *t*

$Pred_t$ = Predicted value at time *t*

Mea_t = Measured Value at time *t*

KG = Kalman Gain

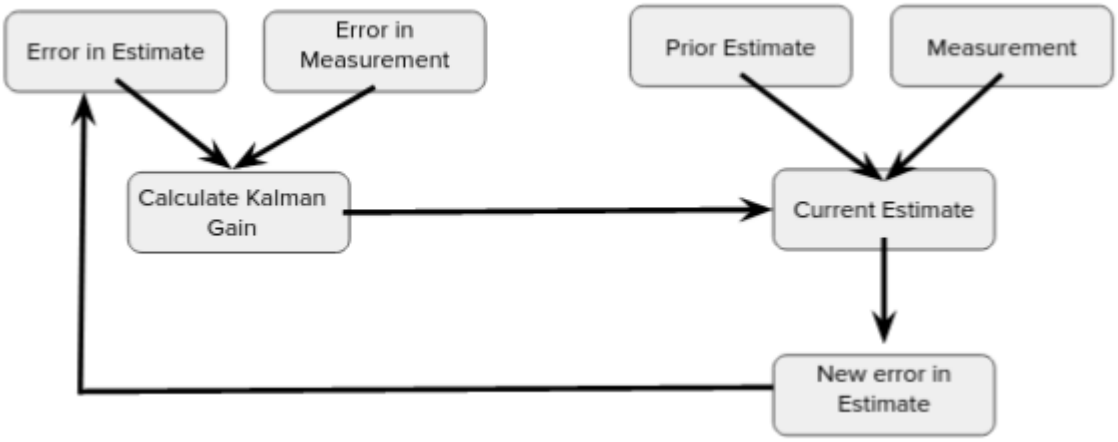
From these two equations we can get an intuitive idea about how the Kalman Filter works. Let's say in a certain reading, the error in prediction is very high. Compared to this, the error in measurement is very low. Under these circumstances, we can clearly see that the Kalman Gain approaches 1. When this happens, the current estimation gets very close to the Measured value. In other words, when the error in prediction is very high, the Kalman Filter gives very little weight to the predicted value and very large weight to the measured value. Similarly, when the error is Prediction is very less as compared to the error in measurement, the Kalman Gain approaches 0. Consequently, the estimation gets very close to the predicted value.

3. Calculating error in Estimation

Finally we can calculate the error in our current estimate using the following formula:

$$E_{Est_t} = (1 - KG) \cdot E_{Est_{t-1}}$$

All these operations have been summed up graphically in this flowchart.



Assumptions

Kalman Filter will always work with

1. Gaussian distributions
2. Linear Functions

Applications in finance

We can now say that the Kalman filter helps us find dynamic (self-correcting) parameters and can be used as a replacement for moving averages. We can use it in various stock market indicators which use moving averages (eg. Bollinger Bands). We can also use it to calculate quantities which may not remain static over time for example hedge ratio in Pairs Trading.

Code Sample

```
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(10)
```

```
def kalman_filter(z_meas, x_est, E_est, E_mea):

  # Kalman Gain.
  K = E_est / (E_est + E_mea)

  # Estimation.
  x_est = x_est + K * (z_meas - x_est)

  # Error.
  E_est = E_est - K * E_est

  return x_est, E_est
```

```
# Input parameters.
time_end = 10
dt = 0.2
t = 4
R = 4
# Initialization for estimation.
x_0 = -10
E_0 = 6
time = np.arange(0, time_end, dt)
n_samples = len(time)
meas_save = np.zeros(n_samples)
```

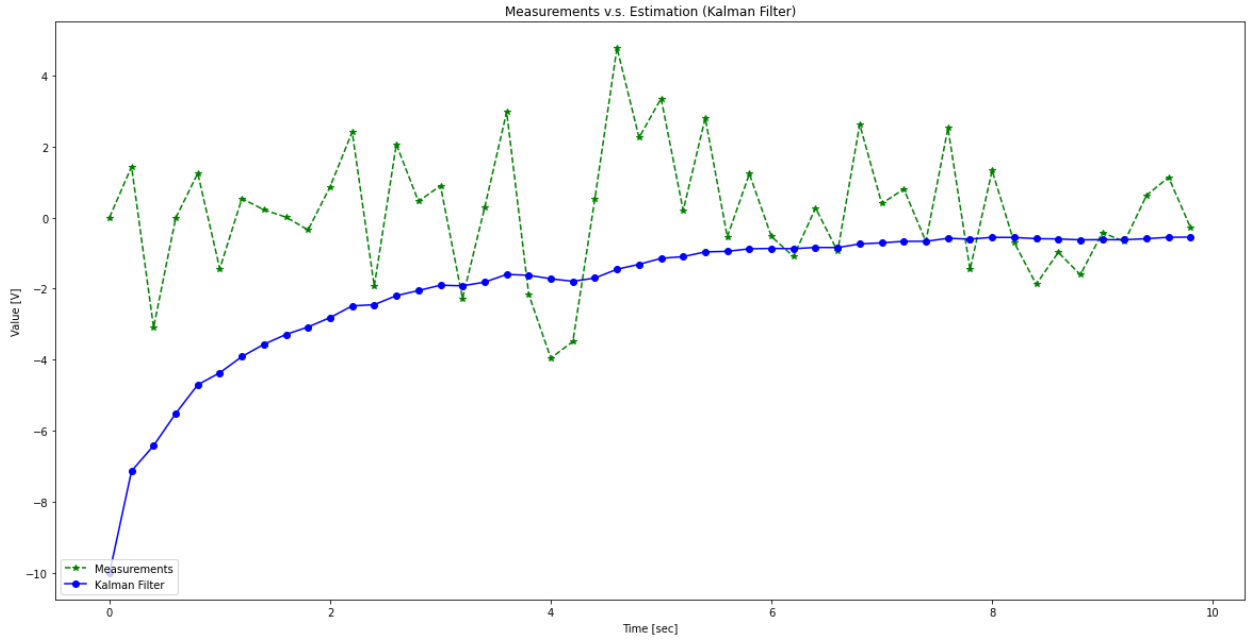
```
esti_save = np.zeros(n_samples)
err = []
for i in range(10):
  err.append(n_samples)
```

```
x_esti, E = None, None

for i in range(n_samples):
  z_meas = np.random.normal(0, 2)
  err.append(z_meas)
  E_mea = np.std(err)
  if i == 0:
    x_esti, E = x_0, E_0
  else:
    meas_save[i] = z_meas
    x_esti, E = kalman_filter(z_meas, x_esti, E, E_mea)

  esti_save[i] = x_esti
```

```
plt.rcParams["figure.figsize"] = (20,10)
plt.plot(time, meas_save, 'g*--', label='Measurements')
plt.plot(time, esti_save, 'bo-', label='Kalman Filter')
plt.legend(loc='lower left')
plt.title('Measurements v.s. Estimation (Kalman Filter)')
plt.xlabel('Time [sec]')
plt.ylabel('Value [V]')
```



REINFORCEMENT LEARNING IN FINANCE

What is Reinforcement Learning?

Reinforcement learning is an area of machine learning concerned with how intelligent **agents**

ought to take **actions** in an **environment** to maximize cumulative **reward**.

The above definition brings a handful of terms to the table at once. Let us understand them and how they form the bigger picture in detail.

The two main entities involved are the *environment* and *agent*. The *agent* performs an *action* 'a' on the *environment* which results in a change in its *state.* The *environment* returns this *state* to the agent in addition to another entity called the *reward.* This *reward* acts like a feedback loop that tells the *agent* whether the *action* it took is simply put, a good one.

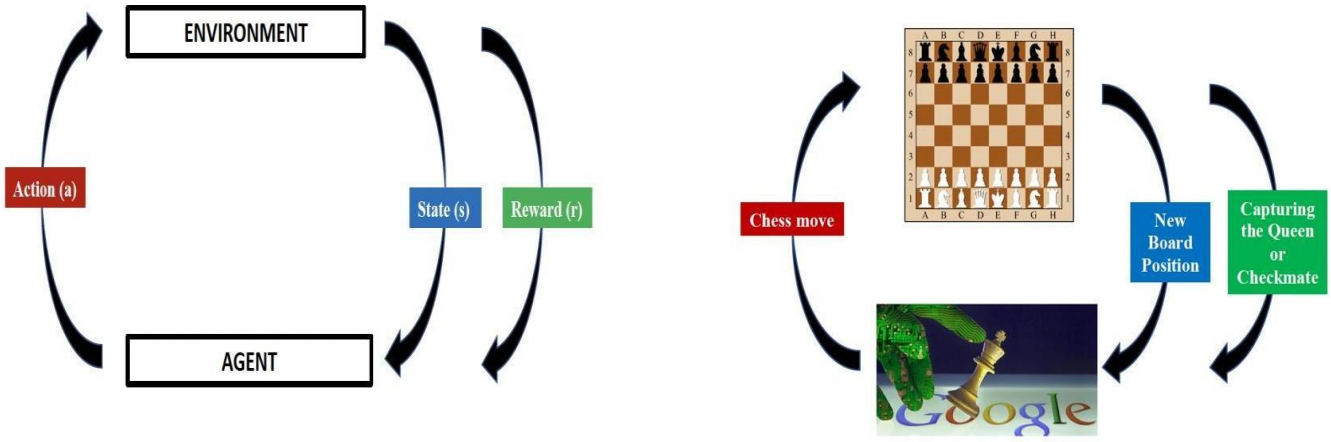
Chessbots as an example

To understand the underlying principle of reinforcement learning let us go through one of the applications of reinforcement learning in the real world: a chess bot.

You must have heard about different powerful Chessbots out there like Leela, Stockfish, and AlphaGo. They have famously even defeated the best human players in the world. So how do they do it? Let's understand it roughly.

These bots are fueled via reinforcement learning. The environment here is the Chessboard, this is where all the action takes place. The agent i.e., the chess bot can perform an action on the environment through a chess move. This action then returns its new state, the new board position to the agent. Along with the new state, it returns a reward that gives the agent an idea of how good its action was or how favourable the current

state of the board is for the chess bot. This reward can be just a real number that scores the position, or a flag for checkmate or queen capture. After the opponent plays its move, the agent observes the new state and plans its next action.



Markov's Decision Process

All the mathematics behind reinforcement learning is based on Markov decision processes. The underlying assumption behind such a process is that the successor state depends only and only on the current state.

We define a few important sets as follows:

- Set of all possible states: S
- Set of all possible actions: *P*
- Set of all possible rewards: $\mathbb{R}$
- Policy $\pi \in \Pi$

POLICY: Policy π is a function that yields an action 'a' given the current state 's' as input:

$$\pi(s) = a$$

A policy can also be thought of as a stochastic function, which emits the probability of taking an action 'a' given an action 's'.

$$\pi(s, a) = \mathbb{P}(action = a \mid state = s)$$

VALUE FUNCTION: This function helps to calculate the 'value' of a state 's'. This is synonymous with quantifying how much the current state is favourable to the user. It is calculated as follows:

$$V_\pi(s) = \mathbb{E}\left(\sum_t \gamma^t r_t | s_0 = s\right)$$

In the above expression, γ is called the discount rate, a number between 0 and 1 whereas r_t is the reward at a time instant 't'. Thus, this expression is the expected value of the cumulative value of future rewards given the current state is s_0. Due to the discount rate being raised to the power of 't', the value function tends to favour immediate rewards, and this behaviour can be controlled externally by varying the discount reward.

Q – LEARNING

The value function simply gives the expected reward for a current state. Now, we introduce an entity called the Q-value which takes into account a state and an action and gives us the value of the maximum reward that can be achieved if the action 'a' is performed at a state 's'.

The last equation we reached, is the famous Bellman's equation which is used extensively in reinforcement

$$Q(s_t, a_t) = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \gamma^3 R_{t+3} + \gamma^4 R_{t+4} + \cdots$$

$$\Rightarrow Q(s_t, a_t) = R_t + \gamma(R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots)$$

$$\boxed{\Rightarrow Q(s_t, a_t) = R_t + \gamma \max_a Q(s_{t+1}, a)}$$

learning systems

DIRECT IMPLICATIONS IN FINANCE.

Now, after toying around with the foundations for a while we will finally dive into the meat of this report. Let us try to understand roughly how Q-Learning might be used in a simple trading bot. This is a very rough example just to demonstrate the overall functioning of Bellman's function.

Assume, we have purchased an equity share for Rs. 100. And we have information on the stock prices for the next five days: 120, 130, 110, 125, 150. Here is a table representing the profit we can make if we hold/sell the share on any particular day.

Action/State	120	130	110	125	150
Hold	0	0	0	0	0
Sell	20	30	10	25	50

Now, if this information is fed into a trading bot, on Day 1, it sees that it earns more on selling the stock than on holding the stock, so without further ado, the stock is sold on the first day. Hence, it is deprived of the maximum profit on Day 5 i.e., Rs. 50. In some way, holding the stock on the first day should have more value than selling it. Let's see if Q-learning can solve this issue.

Through Bellman's equation, we will attempt to update the 'value' of holding the stock on the days so that the trading bot can complete the transaction in the most profitable manner.

Assume discount rate γ = 0.9. Beginning from Day 4, the Q value for holding the stock on that day can be calculated as follows:

Original reward + 0.9 × Maximum reward of the successor state = 0 + 0.9×50 = 45 Going backward with a similar calculation we can evaluate the value for holding the stock for all the other days.

$$\Rightarrow Q(s_t, a_t) = R_t + \gamma \max_a Q(s_{t+1}, a)$$

One share purchased at 100

Action/State	120	130	110	125	150
Hold	32.805	36.45	40.5	45	0
Sell	20	30	10	25	50

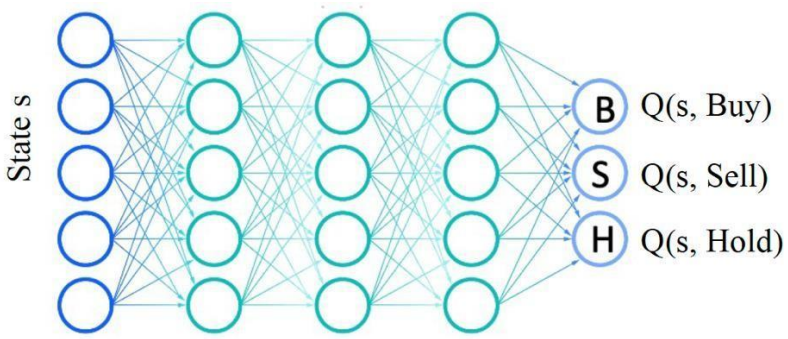
Now, we can clearly see holding the stock on Day 1 is more valuable than selling it, and likewise for the following days so that it can enjoy

DEEP Q-LEARNING & EXPERIENCE REPLAY

In the above example, one must have noticed one fundamental issue. It is not possible to know the future prices of stock accurately, nor can one evaluate Q-values for every state-action pair because the state-space is simply too large to deal with in the case of stock prices.

This is where deep learning comes into play.

We use a deep neural network to estimate the Q-values for a given state-action pair. The input to this layer is a state that is not restricted only to the stock price, it can have other components too which might include indicators such as RSI, moving averages, market sentiment scores, portfolio value, remaining funds, etc.



Now we will discuss in detail how does experience replay work towards optimizing this model. We have at our disposal historical data of the prices of a particular stock.

- First, the weights of the deep neural network are initialized randomly.
- The first state s is passed into the network, and the Q values for the three actions (buy, sell & hold) are calculated. The action a with the highest Q-value is chosen, and a new state s' is achieved. The reward (~profit) is also calculated corresponding to this action.
- This whole transaction can be identified through four entities (s, a, s', r). This tuple is known as an 'experience' and is stored in a memory called the replay memory.

The replay memory has a memory limit beyond which it begins deleting old experiences.

- A random experience is picked up from the replay memory say $e = (s, a, s', r)$.

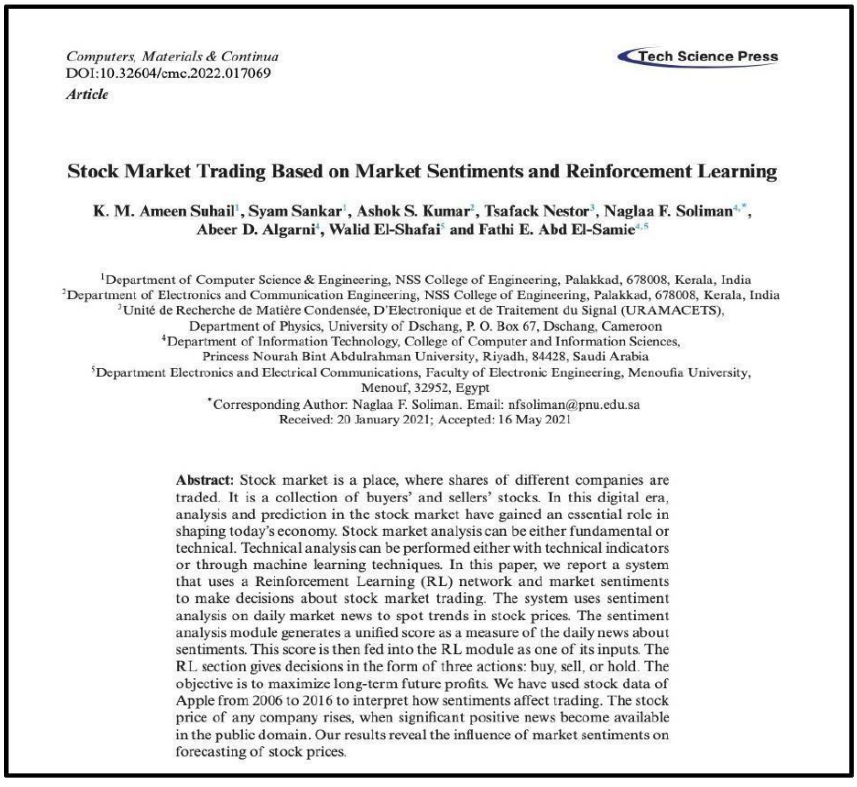
- The state s is then again passed through the neural network, and now the Q value is obtained corresponding to the action a.
- The expected Q value for this state-action pair (s, a) is however different. The expected Q-value is calculated by using Bellman's equation.
-

$$Q^*(s,a) = r + \max_a Q(s',a)$$

To calculate the second part of RHS, max $[Q(s', a)$ needs to be calculated. This is found by passing the state s' again through the network and retrieving the maximum Q value for all possible actions a.

- Finally, the loss for the neural network is calculated as LOSS = (Q* - Q)
- This loss is then used to update the weights of the neural network.

maximum profit on Day 5. With a sufficient number of iterations and data points, we eventually can create a model which can almost accurately estimate Q values for any state-action pair.



Computers, Materials & Continua
DOI:10.32604/cmc.2022.017069
Article

Tech Science Press

Stock Market Trading Based on Market Sentiments and Reinforcement Learning

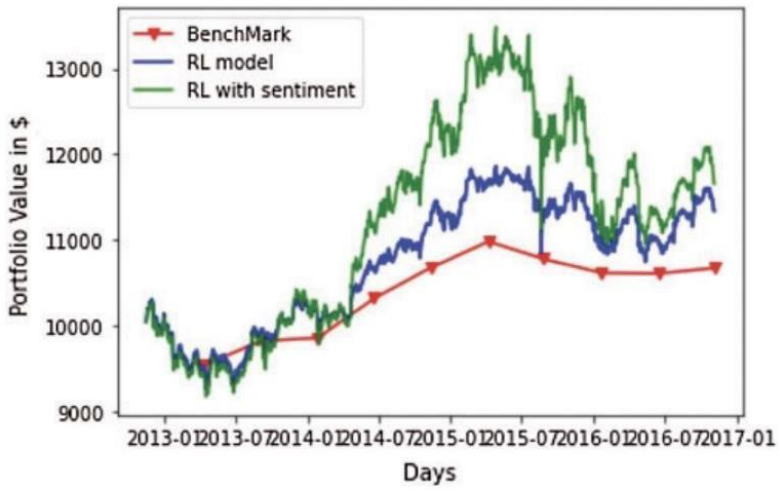
K. M. Ameen Suhail[1], Syam Sankar[1], Ashok S. Kumar[2], Tsafack Nestor[3], Naglaa F. Soliman[4,*], Abeer D. Algarni[4], Walid El-Shafai[5] and Fathi E. Abd El-Samie[4,5]

[1]Department of Computer Science & Engineering, NSS College of Engineering, Palakkad, 678008, Kerala, India
[2]Department of Electronics and Communication Engineering, NSS College of Engineering, Palakkad, 678008, Kerala, India
[3]Unité de Recherche de Matière Condensée, D'Electronique et de Traitement du Signal (URAMACETS), Department of Physics, University of Dschang, P. O. Box 67, Dschang, Cameroon
[4]Department of Information Technology, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, Riyadh, 84428, Saudi Arabia
[5]Department Electronics and Electrical Communications, Faculty of Electronic Engineering, Menoufia University, Menouf, 32952, Egypt
*Corresponding Author: Naglaa F. Soliman. Email: nfsoliman@pnu.edu.sa
Received: 20 January 2021; Accepted: 16 May 2021

Abstract: Stock market is a place, where shares of different companies are traded. It is a collection of buyers' and sellers' stocks. In this digital era, analysis and prediction in the stock market have gained an essential role in shaping today's economy. Stock market analysis can be either fundamental or technical. Technical analysis can be performed either with technical indicators or through machine learning techniques. In this paper, we report a system that uses a Reinforcement Learning (RL) network and market sentiments to make decisions about stock market trading. The system uses sentiment analysis on daily market news to spot trends in stock prices. The sentiment analysis module generates a unified score as a measure of the daily news about sentiments. This score is then fed into the RL module as one of its inputs. The RL section gives decisions in the form of three actions: buy, sell, or hold. The objective is to maximize long-term future profits. We have used stock data of Apple from 2006 to 2016 to interpret how sentiments affect trading. The stock price of any company rises, when significant positive news become available in the public domain. Our results reveal the influence of market sentiments on forecasting of stock prices.

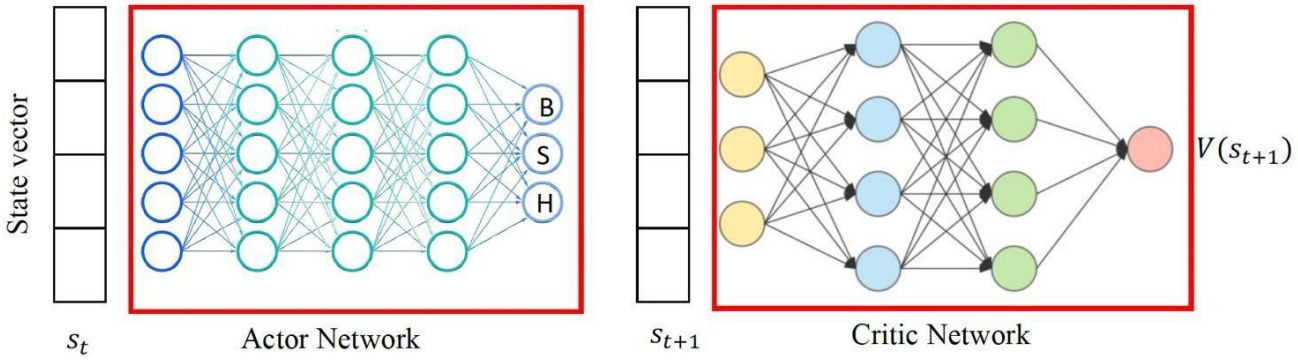
The above work by Suhail et. al. implements Deep Q-learning and experience replay to create a trading bot that performs really well on real-world data. The input state vector in this implementation contains 5 quantities: Open, 5-day SMA, Stocks held, Cash held, and sentiment score. The market sentiment score is calculated by processing relevant market news through a robust NLTK library called VADER (Valence Aware Dictionary and Sentiment Reasoner).

The evaluation results are represented in the following graph:



ACTOR-CRITIC NETWORKS

This is a slightly different kind of reinforcement learning model, which estimated the Value function for the successor state. It implements two networks: the actor and the critic network. The actor-network decides which action to take, whereas the critic network on the other hand estimates the value function for the successor state thereby 'criticizing' or keeping the actor-network in check.



The 'advantage' value is calculated as follows:

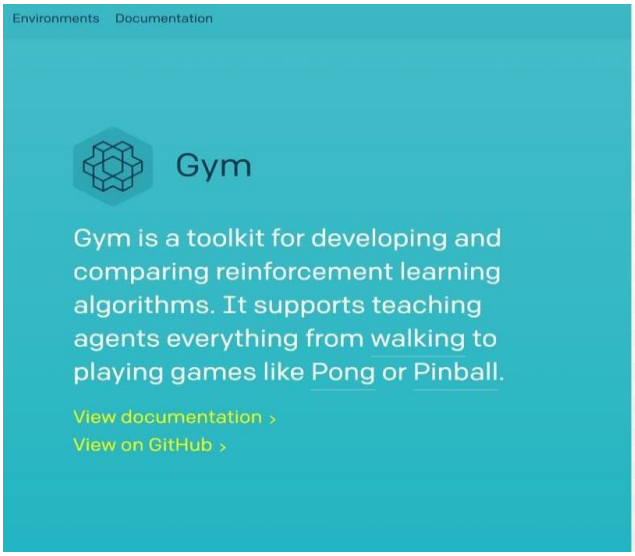
$$\boldsymbol{\delta = R_t + \gamma V(s_{t+1}) - V(s_t)}$$

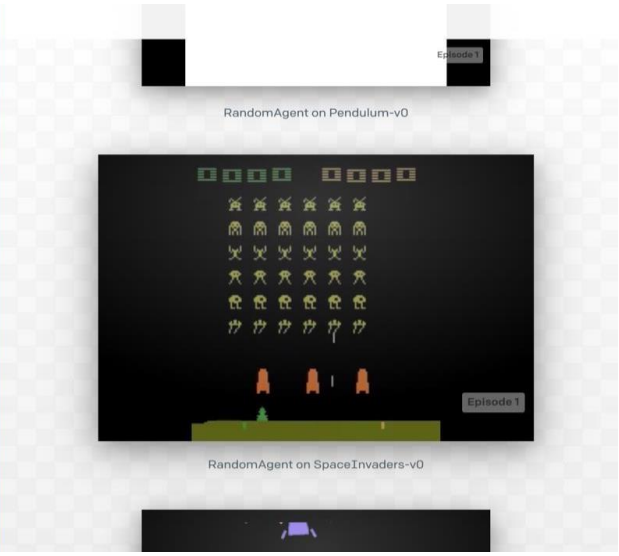
The above expression can be expressed in terms of Q-value as $Q(s_t, a_t) - V(s_{t+1})$

The losses for the above two networks are calculated as follows:

$$\text{Actor Loss} = \delta^2$$
$$\text{Critic Loss} = \delta \cdot \ln \pi(a_t|s_t)$$

Such actor-critic networks can be easily implemented through a dataset using the famous open-source library Gym. This implements many other useful models based on reinforcement learning.





IMPLEMENTATION OF A2C NETWORK ON MARKET DATA

Using the gym open-source library, the A2C network can be implemented on real-world stock data, in this case, the $AAPL stock is used for the period of 1 year: 2019-2020.

Setting up indicators (SMA, RSI, OBV):

Source library Gym. This imp elements many other useful models based on reinforcement learning.

```
[ ]  from gym_anytrading.envs import StocksEnv
     from finta import TA
```

```
[ ]  df1 = yf.download('AAPL', start = '2019-01-01', end = '2020-01-01')
     df1['SMA'] = TA.SMA(df, 12)
     df1['RSI'] = TA.RSI(df)
     df1['OBV'] = TA.OBV(df)
     df1.fillna(0, inplace=True)
```

```
[**********************100%**********************]  1 of 1 completed
```

Setting up the trading environment and adding the indicators to it:

```
[ ]  def add_signals(env):
         start = env.frame_bound[0] - env.window_size
         end = env.frame_bound[1]
         prices = env.df.loc[:, 'Low'].to_numpy()[start:end]
         signal_features = env.df.loc[:, ['Low', 'Volume','SMA', 'RSI', 'OBV']].to_numpy()[start:end]
         return prices, signal_features
```

```
[ ]  class MyCustomEnv(StocksEnv):
         _process_data = add_signals

     env2 = MyCustomEnv(df=df1, window_size=12, frame_bound=(12,50))
```

Training phase:

```
[ ]  env_maker = lambda: env2
     env = DummyVecEnv([env_maker])
```

```
[ ]  model = A2C('MlpLstmPolicy', env, verbose=1)
     model.learn(total_timesteps=1000000)
```

Final results:

As we can see, this trained bot earned a net profit of about 13.97% over the period of one year. This performance can be further improved through Deep Q-Learning, using better indicators and integrating market sentiment scores with the model.

Link to notebook:
https://colab.research.google.com/drive/1j1h3q_oxNax_f-2OLdH8H9JAn0hyEjD-?usp=sharing

CONCLUSION

Reinforcement Learning is just an artificial way of imitating how living organisms learn to do various tasks: by rewarding good actions and penalizing bad ones. This helps a system learn and exploit patterns that otherwise would not be comprehendible by the human eye. It turns out to be very efficient in situations where training data is unavailable, where it learns through experience. It has been extensively used in robots for performing physical tasks with unprecedented accuracy, and in video games where RL has even defeated the best players in the world.

However, reinforcement learning has its own shortcomings. It assumes the world to be purely Markovian which is incorrect. It is data-hungry, meaning it needs a lot of time and experience to construct a usable model. But taking into consideration these disadvantages and pursuing appropriate measures to combat them, RL can open a plethora of opportunities in the domain of artificial intelligence.

The field of finance has made great use of this emerging concept in portfolio optimization, optimized trade execution, market-making, etc. These new RL solutions have proved to be better than many pre-existing solutions to the above-mentioned problem areas. We can expect reinforcement learning to take the world of artificial intelligence by storm and perform tasks like or even better than the most skilled human beings.

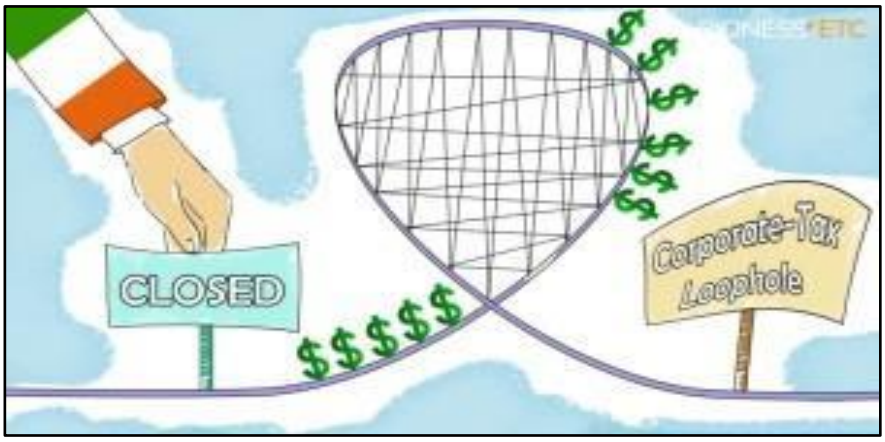
Double Irish Dutch Sandwich

Introduction

Before understanding the importance of this concept, let us have a look at some supporting figures-

1. By 2017, tech companies that were users of the Double Irish Dutch Sandwich (DIDS) scheme, had saved up to 1 Trillion dollars of tax-free European profits in offshore islands such as the Bermuda Islands, the Caribbean islands, etc.
2. In the period 2004-2014, Apple had collected over 111 Billion dollars of European profits.
3. In 2003, Apple had paid a 1% tax on all European profits. By 2014, Apple paid only 0.005% tax on its European profits.

American corporate firms need to pay a high tax rate of 35% but tax evasion schemes like the DIDS help them achieve a <1% effective tax rate (ETR). DIDS in essence is a Base Erosion and Profit Shifting (BEPS) scheme.



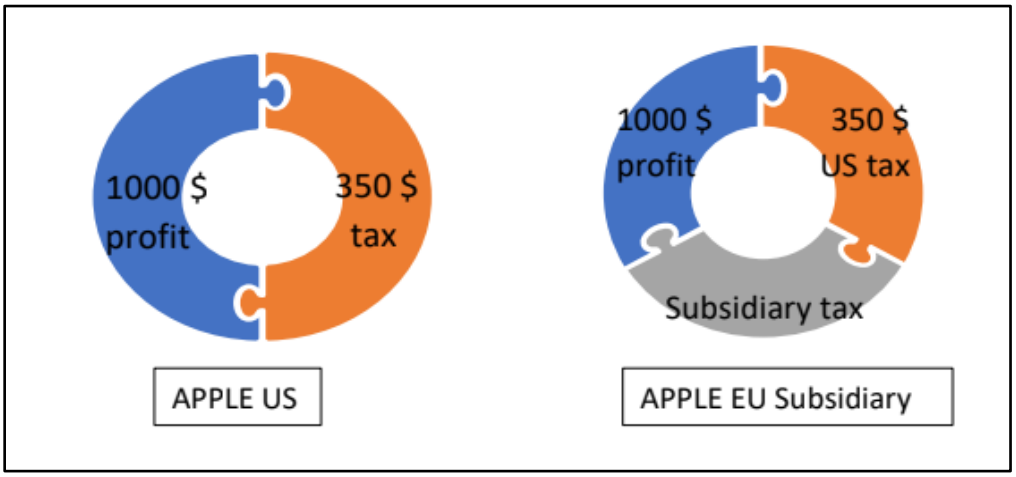
BEPS schemes refer to a set of tax-saving tool based on profit reallocation developed by tech firms exploiting incoherence in international tax rules. Now a natural question arises, why BEPS? This is because of the worldwide tax system in the US.

The US Tax System

There are two types of tax systems, the territorial tax system, and the worldwide tax system. In the territorial tax system, a firm should pay taxes on whatever income it earns in the territory of the nation. However, in the worldwide tax system, a firm will be taxed on its income earned globally. The first system, promotes the free flow of cash allowing companies to expand their reach and perform free trade. However, the

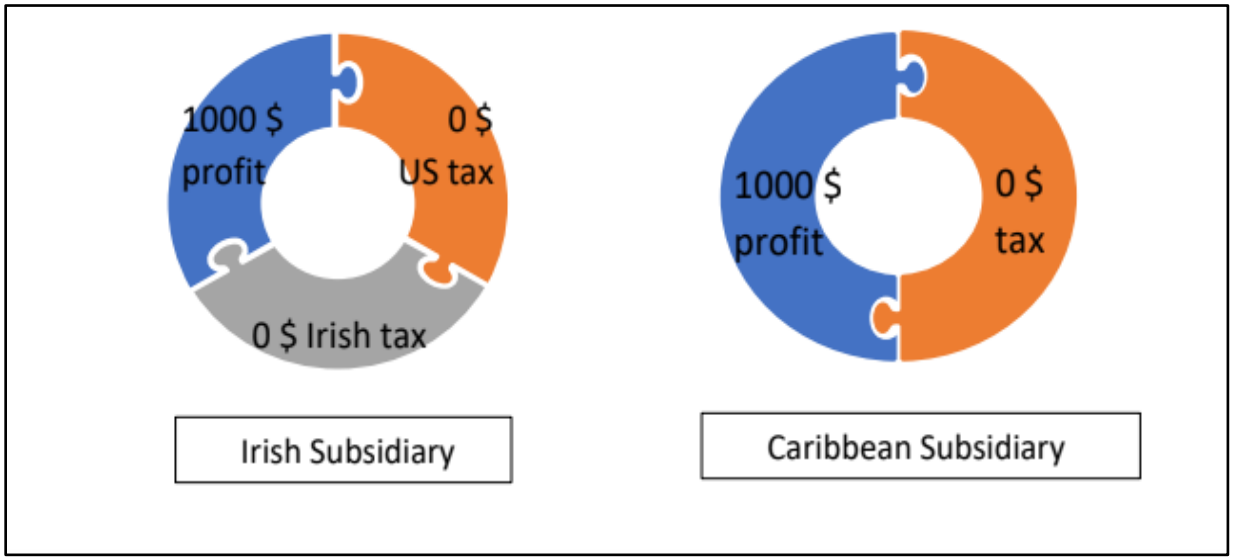
worldwide tax system makes free trade difficult by doubly taxing the country's foreign profits. All the present-day G7 nations use the territorial tax system. The US used the worldwide taxation system till 2017 and hence it forced MNCs based in the US to use BEPS tools such as the DIDS.

To understand the concept of DIDS, let us take the example of Apple US. Suppose Apple sells an iPhone for 1100$ in the US and achieves a 1000$ profit. Now as per corporate tax rule, it would be supposed to pay 350$ on the profit earned. Now, suppose Apple wants to do business in the European countries. For this, it sets up a European subsidiary. The European subsidiary may be Irish as Ireland has a lot of friendly tax policies that help companies use BEPS tools. Let's suppose that it sells the same iPhone for 1100$ in Europe as well. Now, as per the old worldwide tax system followed in the US, Apple will have to pay 350$ of tax to the US government. In addition to this, it will also have to pay taxes in the nation where its European subsidiary is located. This will decrease their profits as they are getting doubly taxed.



So, to avoid paying so much tax, we can open a Caribbean islands subsidiary that will control the European subsidiary. This can be done by transferring assets such as IP from the European subsidiary to the Caribbean island subsidiary. Since the major asset of tech firms are intangible in nature such as IP, it is easier for them to transfer assets digitally.

As the Irish subsidiary uses the IP of the Caribbean islands' subsidiary, we can redirect all the profits made by the Irish subsidiary to the Caribbean islands subsidiary citing royalty payments. In this way, the Irish subsidiary has 0$ of effective profit and hence it won't be subject to any taxes. Meanwhile, as the Caribbean islands have a ~0% corporate tax rate, all of the European profits will now lie untaxed there.

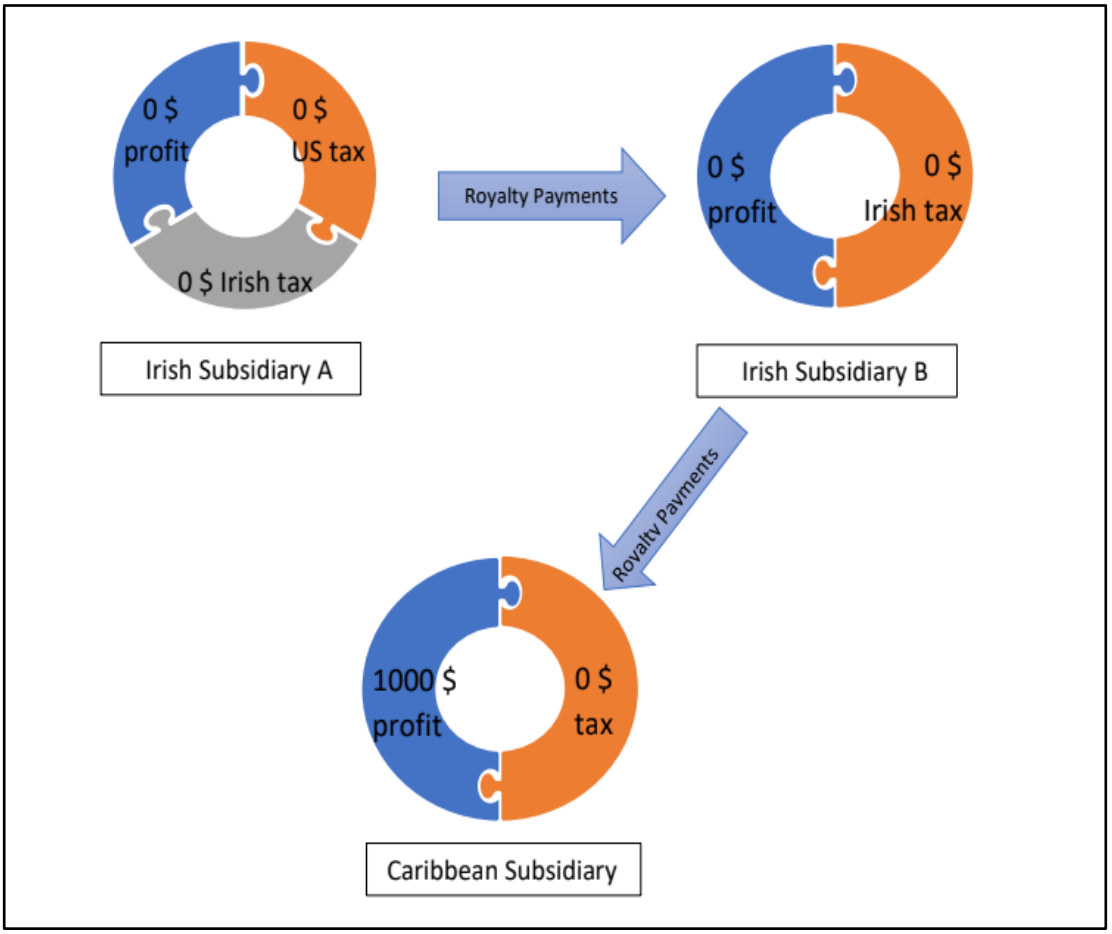


Controlling Foreign Corporations

However, the above system has one problem – CFC. CFC stands for controlling foreign corporations. In the above scenario, if the US authorities find out, they will apply the CFC rule and consider the whole system to be a single unit eligible to pay American tax as it is shielding tax using a controlling firm in the offshore islands.

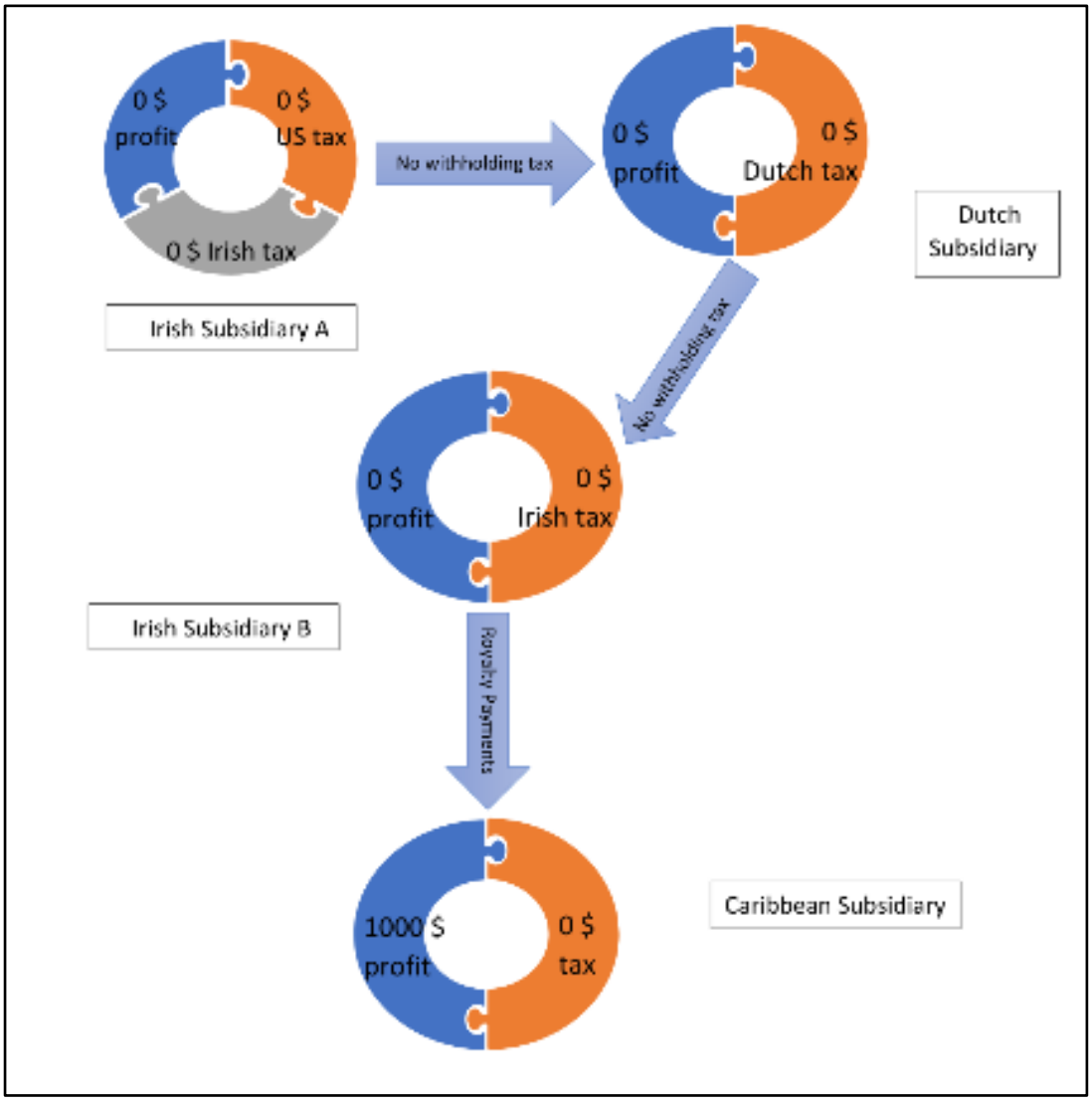
A workaround for this rule would be to introduce another Irish company with control in Ireland itself to

prevent the CFC rule from kicking in. So, in the new scenario, we have Apple's EU subsidiary (A) with control in Ireland itself, a second Irish subsidiary (B) which has control in Caribbean islands, and a Caribbean island subsidiary. The firm-A distributes goods and earns profits across all European states. A now makes full payment to firm B since B is the controlling corporation of A. Since B has its controlling corporation in the Caribbean islands, it transfers all of its profits there. This is known as a Double Irish arrangement as it involves two Irish subsidiaries A and B for shifting profits and saving tax.



To improvise the Effective Tax Rate (ETR) in the above scheme, some companies have introduced a Dutch twist. Since Ireland imposes withholding tax for monetary transfers between two Irish firms, company A will have to pay tax to the Irish government while transferring money to company B. But as Ireland has tax agreements with few countries such as the Netherlands, it does not impose any withholding tax for

monetary transfers between the firms of both nations. In the above example, we can introduce a Dutch company (C) to achieve an even lower ETR. This arrangement is known as the Double Irish Dutch Sandwich.



Discovery & Shutdown

The DIDS scheme was officially discovered by the US in 2014 and it gave a 5-year time period to all tech

firms to discontinue the use of this policy. The European Union also pressurized the Irish government to reform its tax policies so that MNCs cannot avoid paying taxes on their profits by means of profit shifting. The US government also decided to switch to the territorial tax system in 2017 and brought down corporate tax rates from 35% to 15.5% in 2018 under the Trump tax reforms in order to promote free trade and decrease the use of BEPS tools such as the DIDS. Following these tax reforms, Apple had announced the repatriation of 252$ Billion of money in offshore havens. However, the new tax reforms and international tax policy changes did not achieve the expected success and the use of BEPS tools continued to increase over time.

Newer Alternatives

After the shutdown of the Double Irish Dutch Sandwich scheme, two BEPS strategies became prominent, namely the CAIA and the Single Malt scheme.

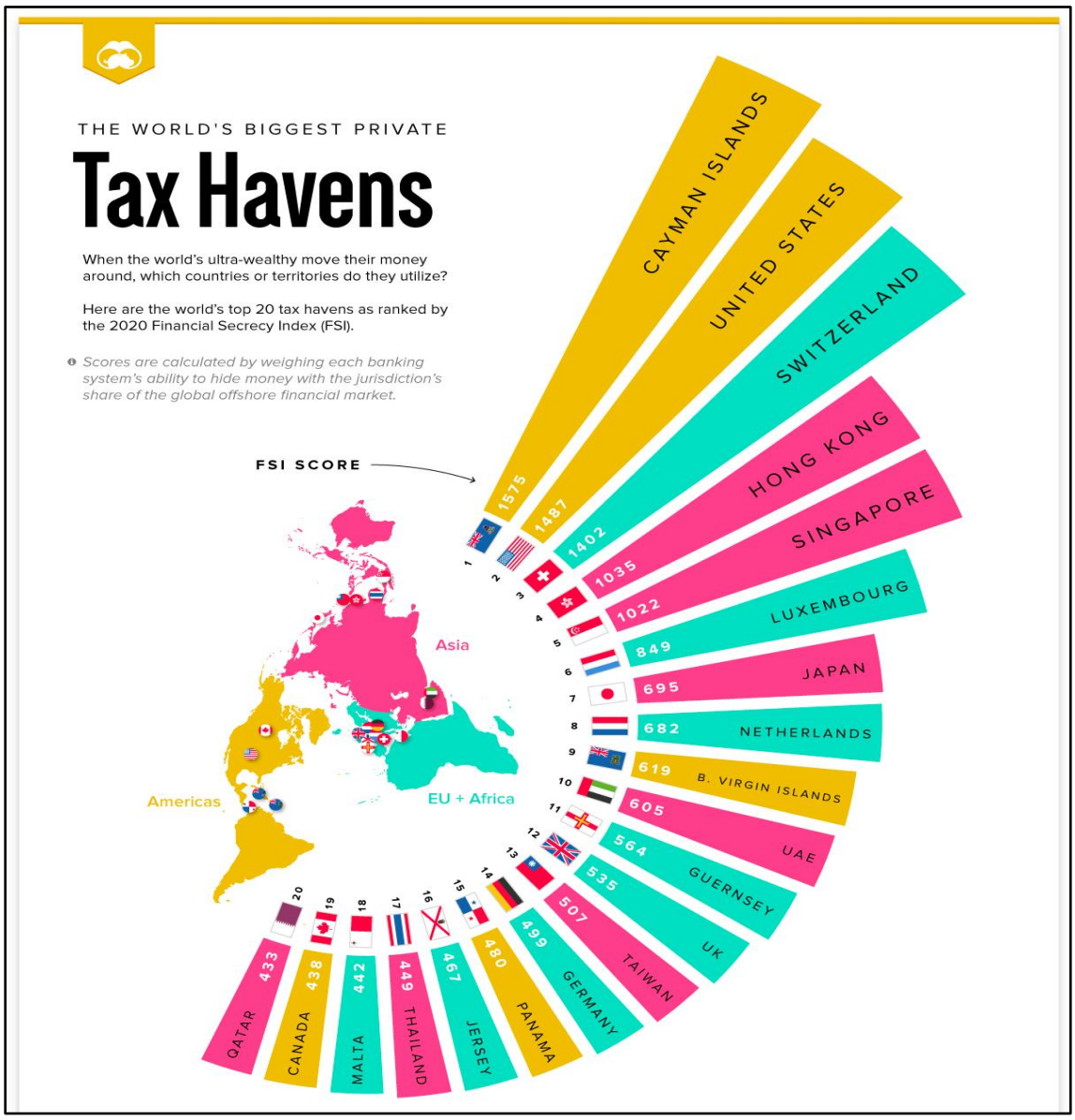
Single Malt Scheme

The Single Malt scheme is quite similar to the Double Irish Dutch Sandwich scheme with a Maltese/UAE firm replacing the Dutch Sandwich. The use of this scheme increased over time and by 2018, it made Ireland the largest global tax haven. Finally, in 2018, a major amendment was made to the Ireland Malta tax treaty to discourage its use.

Capital Allowances for Intangible Assets (CAIA)

The Capital Allowances for Intangible Assets (CAIA) or Green jersey is another popular BEPS tool. It is more powerful than the DIDS or Single Malt scheme as it is based on providing capital allowances for intangible assets like IP which companies use to book completely tax-free profits. In Q1 of 2015, Apple booked $600 Billion of tax-free profit in Ireland in the following manner-

- Apple's Irish subsidiary bought $300 Billion of IP from Apple's Jersey subsidiary
- The Irish subsidiary made use of the CAIA rule to write off this amount against future Irish profits, i.e. the next $300 Billion of profits that Apple's Irish subsidiary makes would be subject to $0 tax.
- Further, Apple made use of the Irish interest relief laws to double its tax shield to $600 Billion with an ETR of exactly 0%.



Conclusion

In this article, we primarily focused on the development of the DIDS scheme, its need, workarounds, etc. We also shed light on modifications and recent alternatives to the Standard Double Irish scheme such as the Single Malt and CAIA.

Dragon King Theory

A rescue to financial crisis…….

Dragon King…….

Seems quite interesting right and so it is Come let's explore the powers of this dragon. Dragon King (DK) is a double metaphor for an event that is both *extremely large* in size or impact (a "king") and born of *unique origins* (a "dragon") relative to its peers (other events from the same system).

These events are generated by mechanisms such as phase transitions, bifurcations, positive feedback, etc which usually take place in complex non-linear systems. In simple words, these are the extreme special

events. They are the outliers. However, being generated by specific mechanisms makes them ***predictable*** and at best, ***controllable.***

Why are we talking about this theory?

So basically this concept came up after the global financial crisis of 2008. When the economy was shattered and the economist pondered upon the event and analyzed that this could have been controlled and the economy could have been in a much better

condition., It's believed that with the help of Dragon King Theory economists will be able to spot similar financial bubbles before they burst in the future.

Now you might have understood the intensity of the topic……

Some might ask then what is the difference between a black swan and a dragon King??

Dragon King
(Sornette 2009)

vs.

Black Swan
(Taleb 2001, 2007)

For the ones unaware of the tern black swan, I would like to ask, how frequently, have you seen a black swan? The answer would be rare and very hard to find. Just as a black swan which is hard to find unpredictable and shattered your belief that all swans should be white so is the black swan theory. A black swan is an unpredictable event that is beyond what is normally expected of a situation and has potentially severe consequences. Black swan events are characterized by their extreme rarity, severe impact, and the widespread insistence they were obvious in hindsight. This is unlike the Dragon King theory which s exactly the opposite, that most extreme events are knowable and predictable. So we can be empowered and take responsibility and make predictions about them.

So what is the main hypothesis behind the dragon king theory?

The two main hypotheses behind the dragon king theory are :

Hypothesis I: Financial bubbles can be identified in real-time

Hypothesis II: The end of the financial bubble can be assessed using probabilistic forecasts.

How to predict dragon king for finance?

Few steps are needed to be taken to identify when will a dragon king appear.

- Find the dragon king outlier drawdowns
- Set up a new different mechanism
- Follow the excesses,i.e. " bubbles".Bubbles are collective endogenous excesses
- fueled by positive feedbacks. Most crises are "endogenous"(can be predicted through a model).
- Whenever there is a super-exponential growth coupled with positive feedbacks then there comes a warning signal.

Let's learn about a new term called the Power Law

The power law is the exponential relationship between two quantities where one quality varies as a power

of the other quantity. A relative change in one quantity results in a proportional relative change in the other quantity. For instance, considering the area of a square in terms of the length of its side, if the length is doubled, the area is multiplied by a factor of four.

But why are we concerned about the power-law?

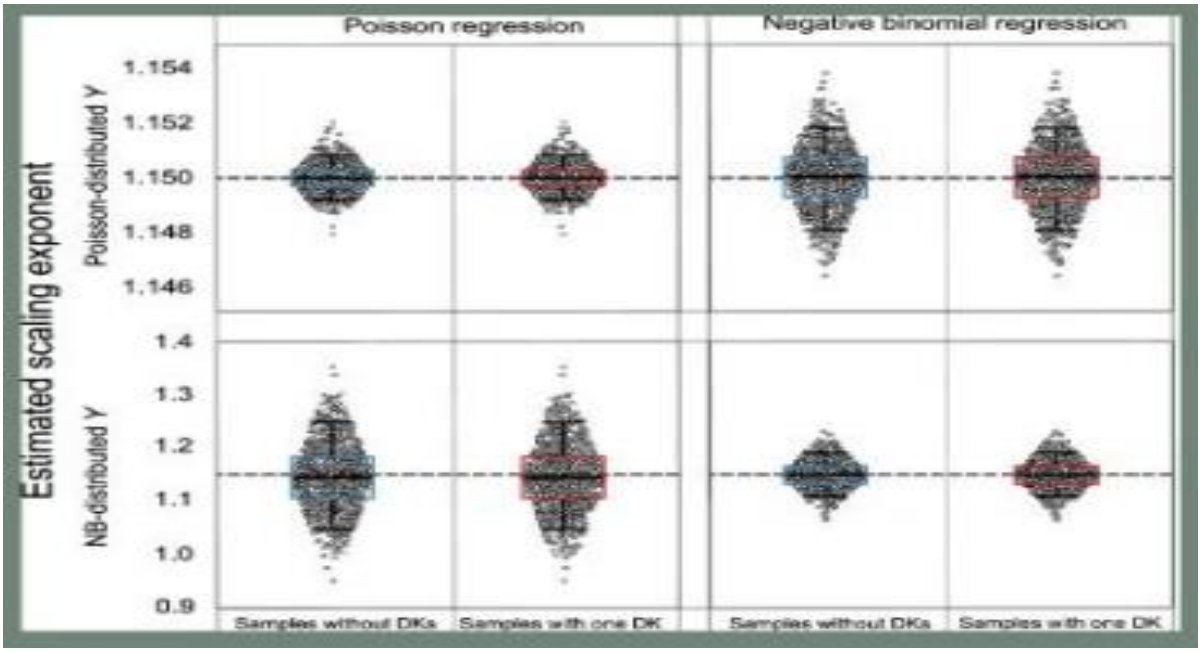
It is well known that many phenomena in both the natural and social sciences have power-law statistics. However, In a variety of studies, it has been found that even though a power-law models the tail of the empirical distribution well, the largest events are significantly outlying (i.e., much larger than what would be expected under the model). Such events are interpreted as dragon kings as they indicate a departure from the generic process underlying the power law. Examples of this include the largest radiation release events occurring in nuclear power plant accidents, the largest city (agglomeration) within the sample of cities in a country, the largest crashes in financial markets, and intraday wholesale electricity prices.

How does the dragon king event impact the scaling law parameter??

Considering Poisson and Negative Binomial distribution we shall try to predict scaling parameters (alpha and beta).

$$Y = \alpha X^{\beta}$$

When Y follows a Poisson distribution, the Poisson regression must be applied to estimate the scaling parameters. Similarly, when Y has an NB distribution, the negative binomial regression must be applied. However, it can be difficult to ascertain the distribution of Y when real data is considered and this can lead to the wrong choice of a regression model. Hence, to show how this can affect the estimated values for the scaling parameters, we also apply the Poisson regression to samples where Y is NB-distributed and the negative binomial regression to samples where Y is Poisson-distributed.



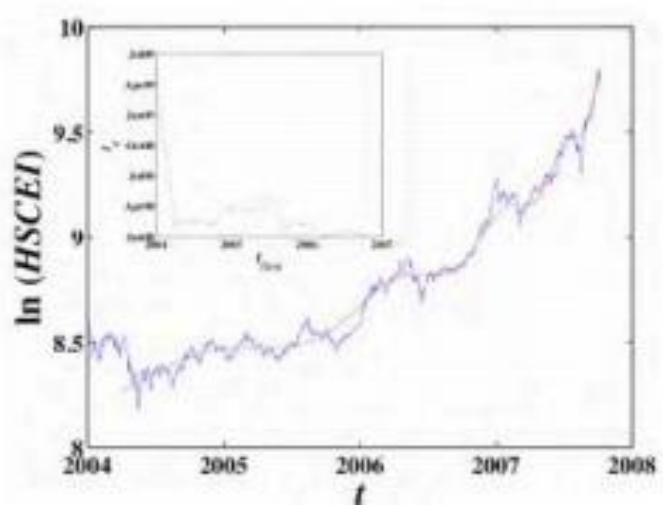
We could observe that the same results are obtained with and without dragon king(DKs)

We conclude that as long as X and Y satisfy the same scaling law for all the points in each sample, the presence of dragon-kings does not have an impact on the estimated scaling exponent, regardless of the type of regression used for the estimation.

Prediction of the end of financial bubbles

The main concepts that are needed to understand stock markets are imitation, herding, self-organized cooperativity, and positive feedbacks, leading to the development of endogenous instabilities. According to this theory, local effects such as interest raises, new tax laws, new regulations, and so on, invoked as the cause of the burst of a given bubble leading to a crash, are only one of the triggering factors but not the fundamental cause of the bubble collapse. We propose that the true origin of a bubble and its collapse lies

in the unsustainable pace of stock market price growth based on self-reinforcing over-optimistic anticipation. As a speculative bubble develops, it becomes more and more unstable and very susceptible to any disturbance. In a given financial bubble, it is the expectation of future earnings rather than the present economic reality that motivates the average investor. History provides many examples of bubbles driven by unrealistic expectations of future earnings followed by crashes. The development of a given financial bubble releases precursory “fingerprints” observable in the stock market prices.



These fingerprints have been modeled by “log-periodic power laws” (LPPL), which are mathematical patterns associated with the mathematical generalization of the notion of fractals to complex imaginary dimensions. We refer to the book of Sornette for a detailed description and the review of many empirical tests and of several forward predictions. Analysis with the LPPL model of the Hang Seng China Enterprises Index (HSCEI) led to (i) a diagnostic of an on-going bubble and (ii) the prediction of the end of the bubble in early 2008

As an investor, you should look out for warning signs of instability such as exponential growth and positive feedback. While these factors can make a stock seem highly lucrative, Dragon King Theory can be used to predict that over time growth will become unsustainable and a crash will likely occur. Beware of the bubble burst.

QUANT CLUB IIT KHARAGPUR

THANK YOU