


TIME SERIES ANALYSIS OF STOCKS AND PORTFOLIO OPTIMIZATION



Paramasivam, Thamizhannal
INTEL SECURITY, BANGALORE

Table of Contents

1. Purpose	1
a. Objective	1
2. Start Of Art	1
a. Business Study: Hvass Laboratories	5
b. Introduction to Stock Market	
c. Behavior of stock Market	
d. Fundamental analysis based stock price prediction	
e. Technical analysis based stock price prediction	
f. Important Parameters	
g. Model performance calculation Parameters	
3. Method	6
3.1. Proposed Solution to Solve this problem	7
3.2. Why Our Solution is better	7
3.3. Algorithm Design	8
3.3.1. Stock Price Prediction	9
3.3.2. Portfolio Optimization	9
3.4. Risk and Challenges	9
3.5. Tools Used	10
3.6. Language Used	10
4. Data	11
4.1. How to Collect Input Data?	11
4.2. List of Chosen Stocks from Nifty50	11
4.3. Pre-Processing Steps	11
4.4. Type of Data	11
4.5. Assumptions and Constrains	12
4.6. How to run Project and generate Output	12
5. Result	13
a. Stock Price forecasting and Analysis	13
a. Time Series Analysis of Stock: ITC	13
b. Time Series Decompose- Stock ITC	14
c. Correlation Analysis Results-Stock ITC	15
d. Stock Price forecasting using various methods	17
e. Residual Analysis-Stock ITC	19
f. ITC Stock Forecasting Models Performance	20
g. Return computed for all stocks	20
b. Portfolio Optimization	22

a. Monte Carlo Simulation Output	22
b. Simulated Annealing Output	22
c. Portfolio Optimization methods Comparisons output.....	23
6. Analysis	
a. Analysis of Stock Price Prediction	26
i. stationary and differencing	26
ii. Correlation Analysis	27
iii. Residual Analysis	29
iv. Stock Forecasting Models Performance Analysis.....	30
b. Analysis of Portfolio Optimization	31
i. Bench mark portfolio analysis.....	31
ii. Monte Carlo simulation analysis.....	32
iii. Simulated Annealing Analysis.....	32
c. Improvements.....	33
d. Lessons Learned.....	33
e. Observation and conclusion.....	33
7. References.....	33
8. Appendices	34

1. Purpose

Stock price analysis and forecasting is classical problem in financial industries. The prediction of a stock market direction and price prediction for act as recommendation system for short-term traders. The prediction of stock growth for yearly helps investor or shareholder of the company to asses return on investment.

The accuracy of stock price prediction is the most important factor in selecting any forecasting methods ranging from Time series analysis to Neural Networks. Research efforts in improving the accuracy of forecasting models are increasing since the last decade. The appropriate stock selections those are suitable building portfolio and optimizing it to earn maximum profit with adjusted risk is a very difficult task.

a. Objective

The main objective of this work is to forecast stock price in short term for 1-2 weeks, make a portfolio using best performing stocks and allocation of optimized investment amount in each such that to earn maximum profile with minimum risk.

- **To forecast short Term Stock price using Time Series:** Perform Time series analysis of stock and predict price for one week. Pick 20 stocks from NYSE/NSE/BSE stock markets and collect the historical prices for 3-5 years of data should be good. Select stocks such that these are diverse enough either by sector or performance, etc.
- **To perform Portfolio Optimization:** Create a portfolio of 5 stocks, from stock prediction experiment, and identify how much ratio to allocate/invest in each asset in a portfolio to maximize the returns or minimize the risk.

Out of this work, we expected to learn about Time series methods for short term stock price prediction and allocation of optimal investment amount to each stocks in portfolio that expected to earn maximum profit with minimum risk.

2. State-of-the-art

a. **Business Study: Hvass Laboratories**

Method Used:

- Monte Carlo simulation of the equity growth model by Pedersen [3] which samples historical financial data for a company and simulates its future equity, earnings, dividends, etc. and then multiplies the simulated equity with samples of the historical P/Book distribution to estimate future stock prices. This model is used on several companies as well as the S&P 500 stock-market index.
- The Monte Carlo simulated stock returns are also used to construct so called Kelly optimal portfolios [4], which work as intended provided we know the true probability distributions of future asset returns.

Outcome:

- Kelly optimal portfolio has been applied in various stocks S&P 500, CocaCola, Wal-Mart, McDonald's and Gov. Bond for 10 year period.
- During this time Government bond yield is 3.5%, portfolio mean yield was 22.5% and std dev 1.2% and Kelly value=0.22 that signifies Kelly portfolio having greater value yield than the minimum-variance (aka "minimum risk") portfolio.

b. Introduction to Stock Market:

The Indian stock exchanges globally holds good reputation. The Bombay Stock Exchange (BSE) is one of the oldest exchanges across the world and the National Stock Exchange (NSE) is among the best in terms of application of advancement of technology. Investment in stock market is regarded as high risks and high return. This attracts large number of investors and economists. However, historical information exist about stock is normally incomplete, complex, uncertain and vague, making it a challenge to forecast the future economic performance.

During 80's and 90's people invest in the stock market based on some information available new paper article and Televisions News. Before the computer age, people use to do trading in stocks and commodities based on their prior experience.

As the level of investing and trading grew, people searched for tools and methods that would increase their gains while minimizing their risk [1]. Globally, trading in the stock market has gained huge popularity and it becomes the part of daily routine for many people to reap handsome profits. However, the prediction of stock price movement becomes a challenge because of the complexity of the stock market data. Though analyzing stock movement behavior is a challenging task, the robust predictive modeling can guide an investor in identifying and segmenting high performance securities, so as to take the superior investment decisions.

Statistics, technical analysis, fundamental analysis, and linear regression are all used to attempt to predict and benefit from the market's direction. None of these techniques has proven to be the consistently correct prediction tool. Also, many of these techniques are used to pre-process raw data inputs, and their results are fed into neural networks as input. The central idea to successful stock market prediction is achieving best results using minimum required input data and the least complex stock market model. Recent advances in soft computing techniques offer useful tool for analyzing the stock market's movement and the movement of individual stock prices to retrieve knowledge that may guide investors on when to buy and sell [2].

c. Behavior of stock Market:

- Stock market is a chaotic, complex, non-stationary, noisy, nonlinear and dynamic system but it does not follow random walk process [3].
- Novice investor follows a mantra called "buy low, sell high" that does not signifies reason behind business decisions. But an experienced investor states that "don't try to catch falling knives".
- Stock price prediction in long term is relatively easier than predicting on daily basis as stock price is fluctuate rapidly on causal/external events. E.g. Budget, New government policies, demonetization, GST have an impact in short term and not to worry about these facts for long term investors.

- To achieve this objective requires extensive study about company, sector and micro economics conditions needs to be understood. However, for short and medium term speculations, fundamental analysis is generally not suitable.

There are two methods widely been followed in stock price prediction (a) Fundamental analysis based stock price prediction (b) Technical analysis based stock price prediction.

d. Fundamental analysis based stock price prediction:

The fundamental analysis based stock price prediction involves the in-depth analysis about company's performance, profitability on investment. It uses revenues, earnings, future growth, return on equity, profit margins, and other data to determine a company's underlying value and potential for future growth.

As per fundamental analyst, the market price of a stock tends to move towards its "real value" or "intrinsic value" that is prices moves based on fundamental core values of the business. If this value of a stock is above the current market price, the investor can decide to purchase the stock because the stock price will bound to rise and move towards its "intrinsic or real value". If this value of a stock is below the market price, the investor may decide to sell the stock because the stock price is bound to fall and come closer to its intrinsic value. To start finding out the intrinsic value, the fundamentalist analyzer makes an examination of the current and future overall health of the economy as a whole

Assumptions:

- Stock price (current and future) depends on its intrinsic value and can anticipate return on investment.
- Investors are 90% logical, examining their investments in detail.

Advantages:

- These are systematic approach and its ability to predict changes before they show up on the charts.
- Fundamental analysis is a superior method for long-term stability and growth.

Disadvantages:

- It becomes harder to implement a software that formalize all this knowledge and interpretation of this knowledge may be subjective.
- It is hard to time the market using fundamental analysis.

Measures:

The Price-to-Book Ratio (P/B): Book value is that value of a company, which the owner is likely to gather if they decide to liquidate (sell off in dire straits) the company

Dividend Yield: The dividend yield shows how much payout you're getting for your money. It is the stock's annual dividend payout by the stock's price.

Price-to-Earnings Ratio (P/E): P/E Ratio compares the market price with the EPS. Where, EPS (Earning per share) is the company's net earnings by the numbers of outstanding shares of the stock. Higher the P/E ratio, more people are convinced to pay high for that share expecting higher growth in coming future.

Returns on Equity (ROE): It is used as a general indication of the company's efficiency, in other words, how much profit it is able to generate given the resources provided by its stockholders. Investors usually look for companies with ROE that are high and growing.

No Element Stands Alone. P/B, P/E, Dividend Yield is too narrowly focused to stand alone as a single measure of a stock. By combining these methods of valuation, you can get a better view of a stock's worth.

e. Technical analysis based stock price prediction:

Technical analysis is a method of evaluating stocks by analyzing statistics generated by market activity, past prices, and volume. It looks for peaks, bottoms, trends, patterns, and other factors affecting a stock's price movement. Future values of stock prices often depend on their past values and the past values of other correlated variables. Technical analysis looks for patterns and indicators on stock charts that will determine a stock's future performance [3].

However, it is used by approximately 90% of the major stock traders. Despite its widespread use, technical analysis is criticized because it is highly subjective. Different individuals can interpret charts in different manners.

Time series analysis, Traditional machine learning approaches ranging from Linear regression to Support Vector Machines has been attempted and been successfully used to make short term stock price predictions. Even recently, neural networks have been successfully applied in time-series problems to improve multivariate prediction ability. Neural networks have good generalization capabilities by mapping input values and output values of given patterns. Neural networks are usually robust against noisy or missing data, all of which are highly desirable properties in time series prediction problems. Various neural network models have already been developed for the stock market analysis.

Assumptions:

- Market moves in trends dictated by the constantly changing attitudes of investors in response to different forces. History repeats itself i.e. under similar kinds of inputs the stock behave in similar manner.
- Prices have tendency to go with the trend rather than against it.
- Investors are 90% psychological, reacting to changes in the market environment in predictable ways.

Advantages:

- It is used by approximately 90% of the major stock traders.
- It is also used to analyze the stock for shorter period.

Disadvantages:

- Despite its widespread use, technical analysis is criticized because it is highly subjective.
- Different individuals can interpret charts in different manners.

Important Parameters:

In technical analysis of stock market data 52 different parameters, indicators and oscillators have been defined. Even though each indicator provides some additional information about the stock, using each one of them will make the system complex and slow. Below are the list of most widely used parameters.

Moving Average (MA): This is perhaps the oldest and the most widely used technical indicator. It shows the average value of stock price over time. The shorter the time period, the more reactionary a moving average becomes. A typical short term moving average ranges from 5 to 25 days, an intermediate-term from 5 to 100, and long-term 100 to 250 days.

Exponential Moving Average (EMA): An exponential moving average gives more weight to recent prices, and is calculated by applying a percentage of today's closing price to yesterday's moving average. The longer the period of the exponential moving average, the less total weight is applied to the most recent price. The advantage to an exponential average is its ability to pick up on price changes more.

Moving Average Convergence/Divergence (MACD): It is the difference between two exponential moving averages, normally one short moving average and one long moving average.

Relative Strength Index (RSI): An oscillator, introduced by J. Welles Wilder, Jr., is based upon the difference between the average gains vs. the average loss over a given period. The RSI compares the magnitude of a stock's recent gains to the magnitude of its recent losses.

Compound Annual Growth Rate (CAGR): A useful measure of growth over multiple time periods. It can be thought of as the growth rate that gets you from the initial investment value to the ending investment value if you assume that the investment has been compounding over the time period.

f. Model performance calculation Parameters:

Accuracy of a forecasted model has been computed using MAPE metrics.

Mean Absolute Percentage Error(MAPE) :

The mean absolute percentage error (MAPE), also known as mean absolute percentage deviation (MAPD), is a measure of prediction accuracy of a forecasting method in statistics, for example in trend estimation. It usually expresses accuracy as a percentage, and is defined by the formula:

$$M = \frac{100}{n} \sum_{t=1}^n \left| \frac{A_t - F_t}{A_t} \right|,$$

Where A_t is the actual value and F_t is the forecast value.

3. Method

a. Proposed Solution to Solve this problem

We implemented a system that addressed two objectives (a) Stock price Prediction and (b) Portfolio optimization.

Our proposed method for Stock price prediction are

1. Auto Regression Integrated Moving Average
2. Holt-Winters seasonal method
3. Time Series Analysis on Neural network auto regression

Our proposed method for Portfolio Optimizations are

1. Bench Mark Approach
2. Monte Carlo simulation
3. Simulated Annealing

In our proposed method, we applied Time Series predictions using **ARIMA, Holt-Winters** and **Time series regression on Neural Networks** models and captured all the performance measuring metrics. Based on that we build portfolio of top 5 performing stocks on the basis of CARG values for past 5 years of historical price and came up with optimized investment amount allocation to all the stocks using **Monte Carlo simulation and Simulated Annealing method** and recommended a portfolio that would yield maximum return.

We will train stock price prediction methods using 5 years of historic data starting from January 2012 to December 2016 and then test our models to check which systems yields better output by forecasting 2 weeks of historical data starting from January 1st 2017 to 14th January 2017.

b. Why Our Solution is better?

Our solution is Time series stock price prediction using various methods that is well known and already proven approaches for technical analysis of stock price prediction.

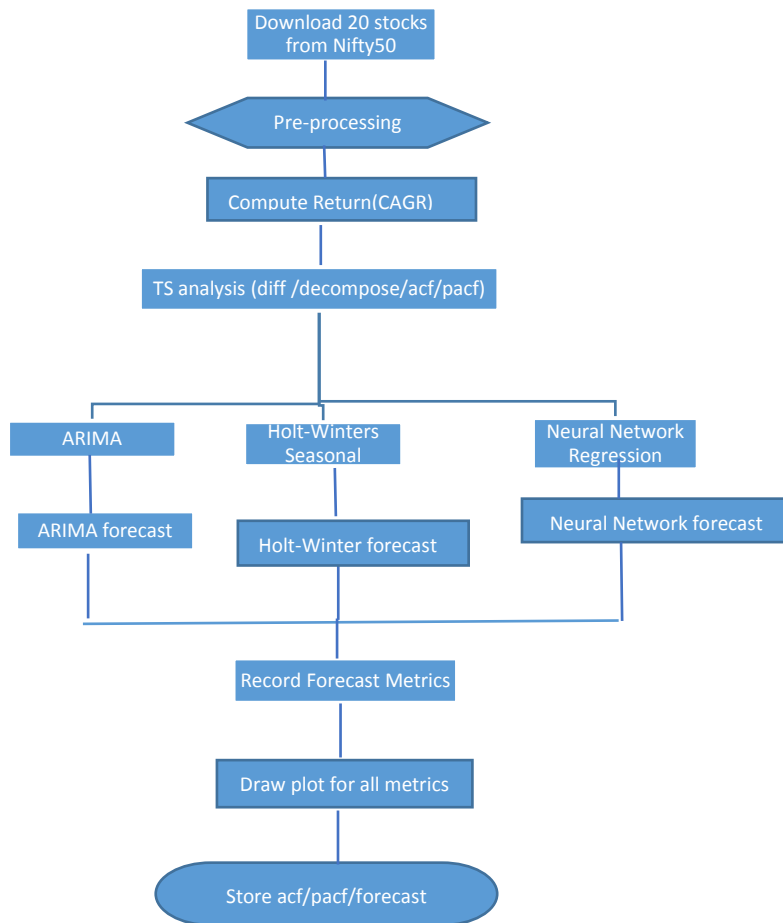
- Ease of use and metric collection
- Lower computational cost

c. **Algorithm Design**

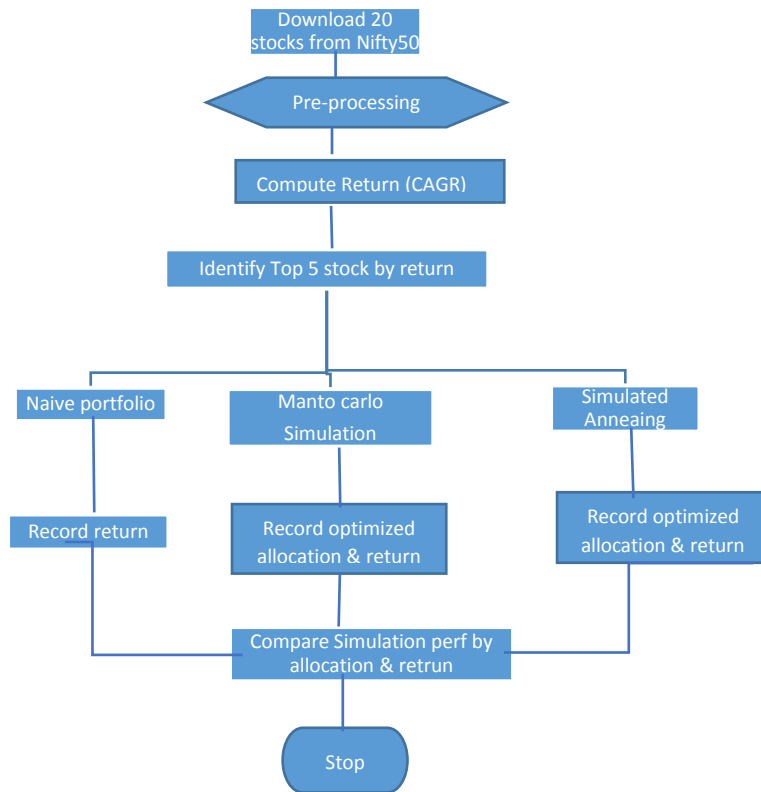
In this session, we have provided our algorithm design flow as below.

- i. **Stock Price Prediction**
- ii. **Portfolio Optimization**

Algorithm Design: Stock Price Prediction



Algorithm Design: Portfolio Optimization



d. Risk and Challenges:

Stock Price Prediction:

- Nobody can predict perfect stock price in future neither the experts nor amateurs traders have the least idea what is going to happen with the economy in the future.
- Economist/Traders come up with multiple complex mathematical models that helps us to alleviate few circumstances, but not all

Portfolio Optimization:

- Identify the best performing stocks to form portfolio
- Find out the optimal methods to allocate capital amount to each stock in chosen portfolio.
- Identify the best performing stocks in diversified sectors.

- Define constraints such as minimum and maximum allocation of capital amount to each stock in portfolio.
- Identify the point on which one supposed to include or exclude the stock from portfolio

e. Tools Used:

R (Programming Language) [4] –

R is a programming language and software environment for statistical computing and graphics supported by the R Foundation for Statistical Computing. The R language is widely used among statisticians and data miners for developing statistical software and data analysis. Polls, surveys of data miners, and studies of scholarly literature databases show that R's popularity has increased substantially in recent years.

Packages used :

- fpp
- ggplot2
- ggfortify
- rpart
- zoo
- dplyr

f. Language Used:

RStudio Desktop:

RStudio is a free and open source integrated development environment (IDE) for R, a programming language for statistical computing and graphics. RStudio was founded by JJ Allaire, creator of the

Programming language ColdFusion. Hadley Wickham is the Chief Scientist at RStudio. RStudio is available in two editions: RStudio Desktop, where the program is run locally as a regular desktop application; and RStudio Server, which allows accessing RStudio using a web browser while it is running on a remote Linux server. Prepackaged distributions of RStudio Desktop are available for Microsoft Windows, Mac OSX, and Linux.

RStudio is available in open source and commercial editions and runs on the desktop (Windows, Mac, and Linux) or in a browser connected to RStudio Server or RStudio Server Pro (Debian / Ubuntu, RedHat / CentOS and SUSE Linux). RStudio is written in the C++ programming language and uses the Qt framework for its graphical user interface.

4. Data

a. How to Collect Input Data?

Input data is taken from Yahoo Finance using following steps:

1. For our project, we are considering Nifty50 Companies. The list of companies in Nifty50 can be obtained from NSE India website [2].
2. Use Historical data field, time period as 1-Jan-2012 to 31-Dec-2016 and frequency as daily to get data from Yahoo Finance.
3. Identified 20 companies from Nifty50 companies manually that maps to Nifty50 index companies across various diversified sectors.
4. Repeat steps 2 and 3 for all 20 companies and save stock data as csv file in “data” folder in project work space.

b. List of Chosen Stocks from Nifty50:

BAJFINANCE, CADILAH, HDFC, IOC, LT, RELIANCE, TATAMOTORS, TITAN, BHARATFORG, HAVELLS, HINDZINC, ITC, MOTHERSUMI, SHREECEM, TORNTPHARM, BRITANNIA, HDFCBANK, INFY, PIDILITIND, TCS

c. Pre-Processing Steps:

- **Missing Values Imputation:** Weekends and NSE Holidays imputed with last working day's adjusted closing price.
- **Continuous Date Missing:** Generated a continuous date format and merged with original data and applied above missing value imputation for weekends.
- **Attributes Chosen:** Data and Adjusted Closing Price

d. Type of Data:

All stock price data is stored as CSV file with below content in data directory. Here we use Adjusted Close (Adj Close) attribute as stock price for consideration.

Date	Open	High	Low	Close	Volume	Adj Close
1/2/2012	201.8505	201.8505	197.8995	198.6495	2860800	115.6424
1/3/2012	199.8495	201.5505	199.3005	200.3505	6065500	116.6327
1/4/2012	201.4995	201.7005	198.4995	199.5495	11875000	116.1664
1/5/2012	199.05	199.9995	197.5995	199.8	11364400	116.3122
1/6/2012	199.05	203.1	198.3495	202.0995	8788200	117.6508
1/9/2012	201.6495	202.3005	199.6995	202.0005	7408200	117.5932

e. Assumptions and Constrains:

Stock Price Prediction Assumptions:

- Pick only 20 Stock from Nifty50 Index
- Consider 3-5 years of historical prices
- Choose stocks diversified sector from Nifty50 Index (Min Risk)
- Chosen 20 Stocks that Maps Nifty 50 Portfolio stocks –Finance, Pharma, IT, Auto, FMCG, Oil, Mineral etc.

Portfolio Optimization:

- Chosen only 5 stocks.
- Allocate/invest funds in each asset in a portfolio to maximize the returns or minimize the risk
- Chosen Top5 best performing stocks in diversified portfolio
- Allocation: Minimum=10% per script and Maximum=30% per script

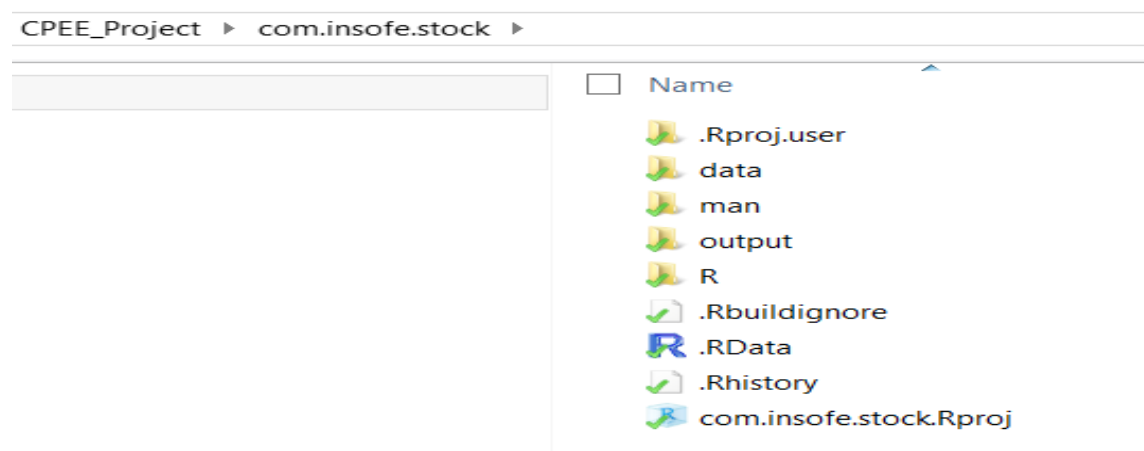
f. How to run Project and generate Output:

Perform following steps to run project and generate output:

- Open project file com.insofe.stock.Rproj as R project in R Studio.
- Navigate to R folder and Run ProjectMain_Stock_Price_Portfolio_Opt.R file. It internally run necessary source code for stock price prediction and Portfolio optimization
- Project output files would be stored in output directory.

Project Structure:

- data directory: All the 20 stocks price for 5 years has been downloaded and copied “data” folder.
- R directory: Stock price prediction and optimization source code
- output directory: All the plots, stock return and forecast results stored as .Rda file

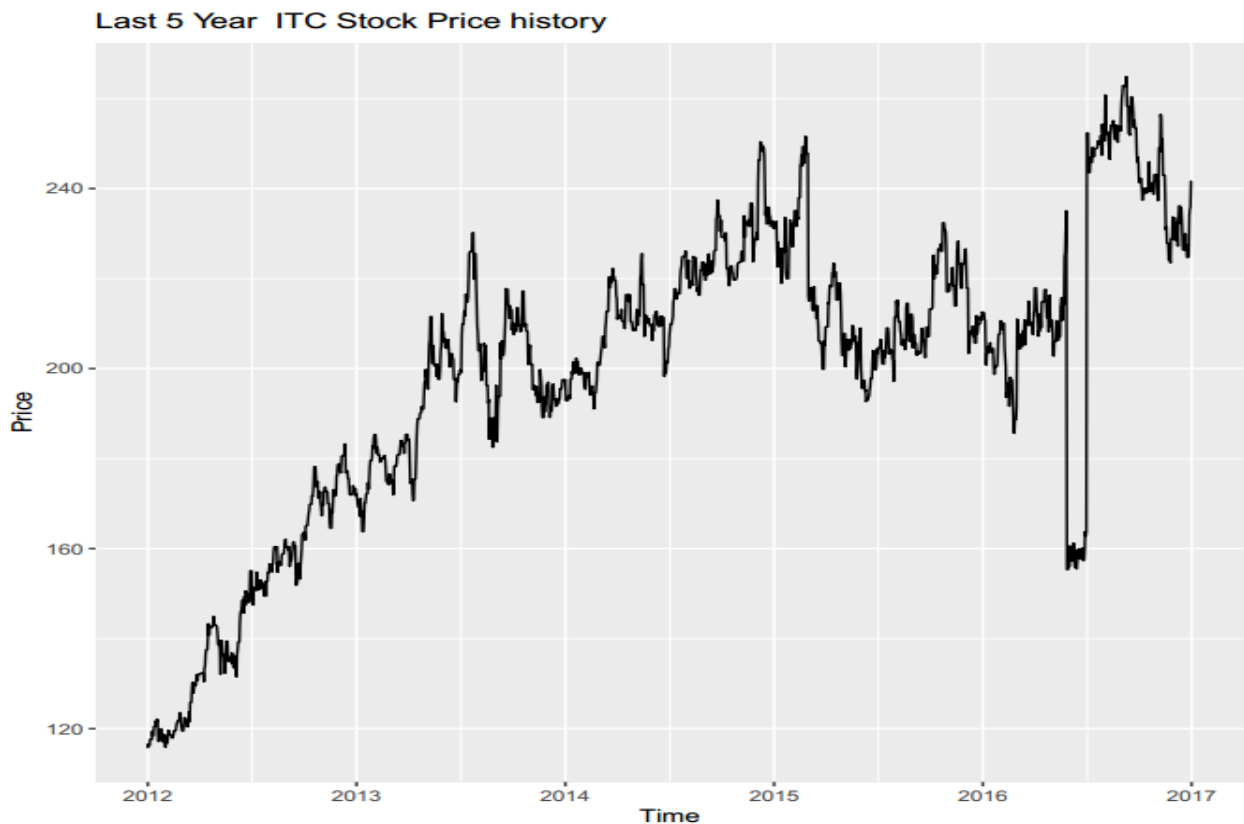


5. Results

- Time series analysis for stock price forecasting has been done using a) Arima b) Holt-Winter c) Neural Network models.
- Here we have provided detailed results about various analysis done for “ITC” stock, stock forecasting for 2 week and all stock price prediction results.
- For portfolio optimization we have applied a) Naïve Bench mark model b) Monto Carlo simulation c) Simulated annealing methods and have provided optimized allocation and computed percentage return metrics.
- All the results would be present at output director of project source code.

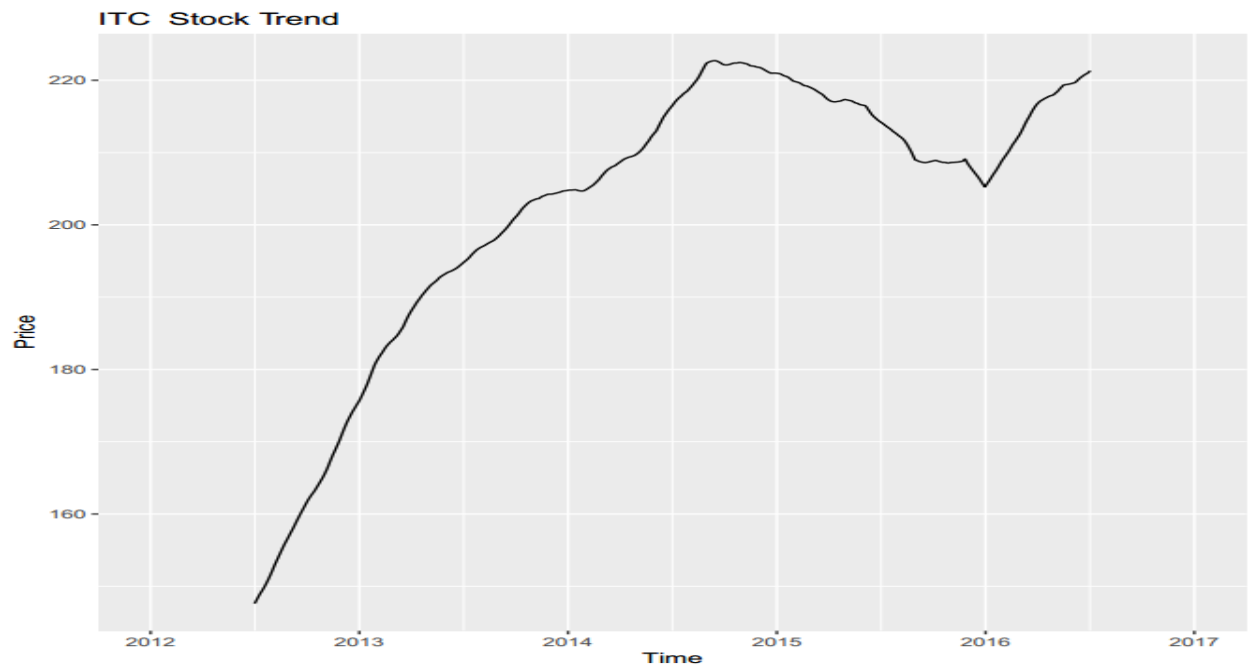
a. Stock Price forecasting and Analysis

a. Time Series Analysis of Stock: ITC

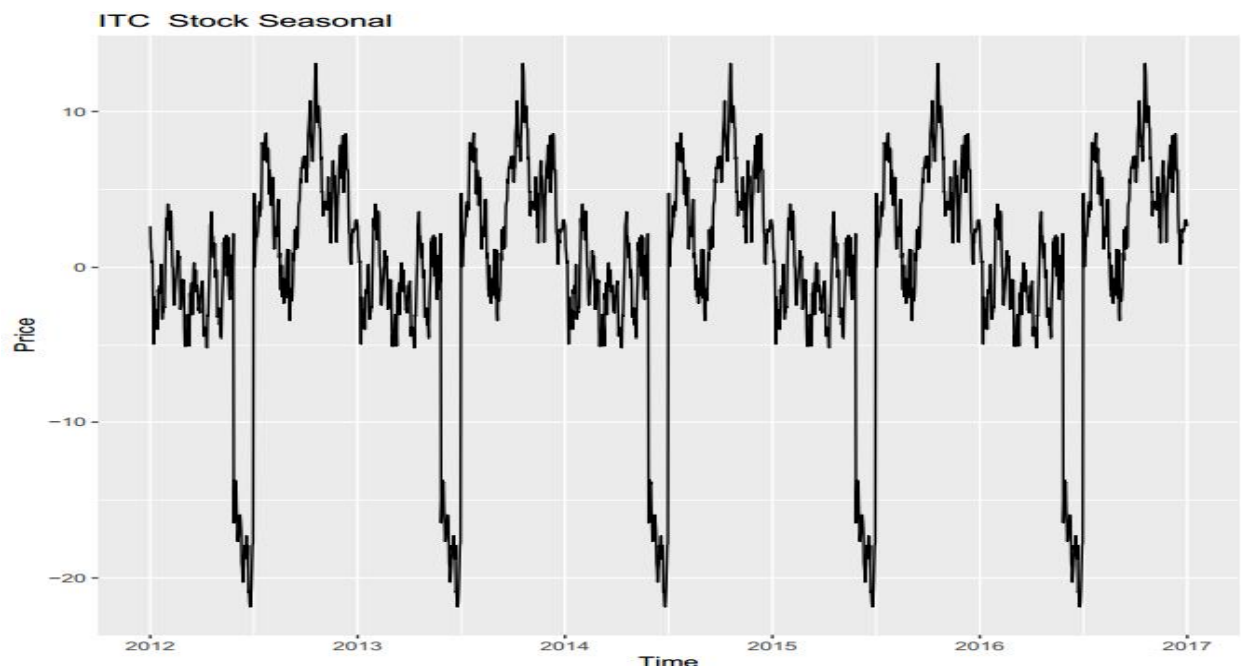


b. Time Series Decompose- Stock ITC

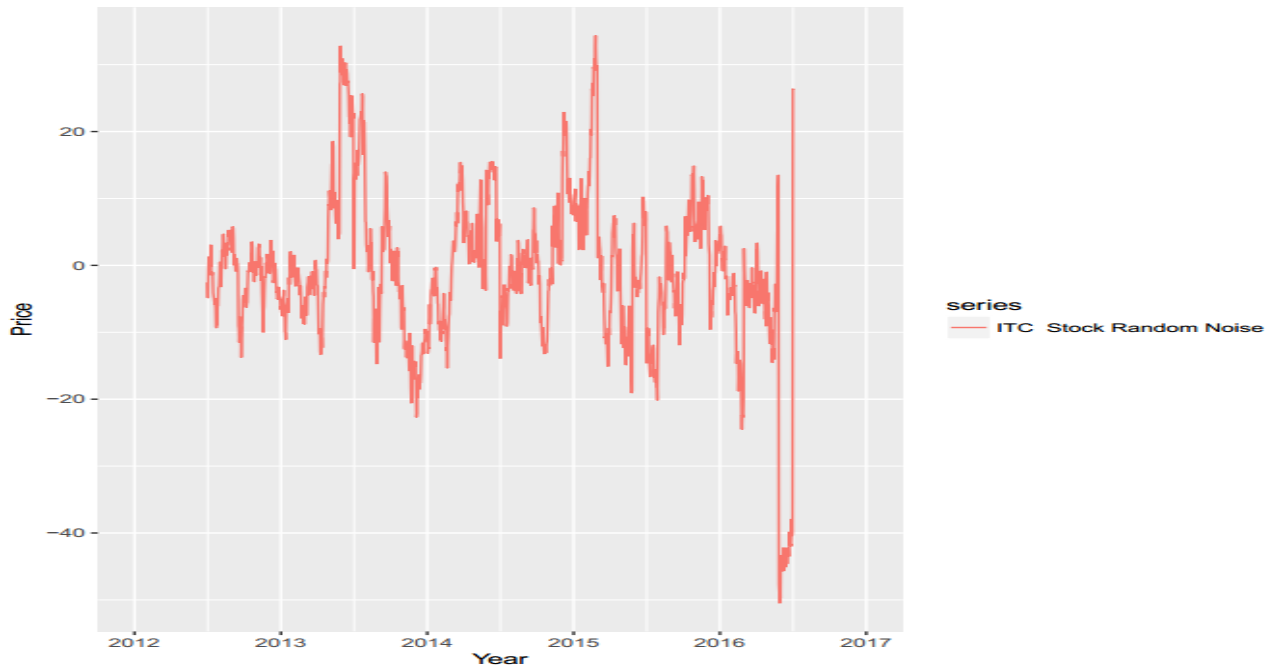
1. Trend



2. Seasonal

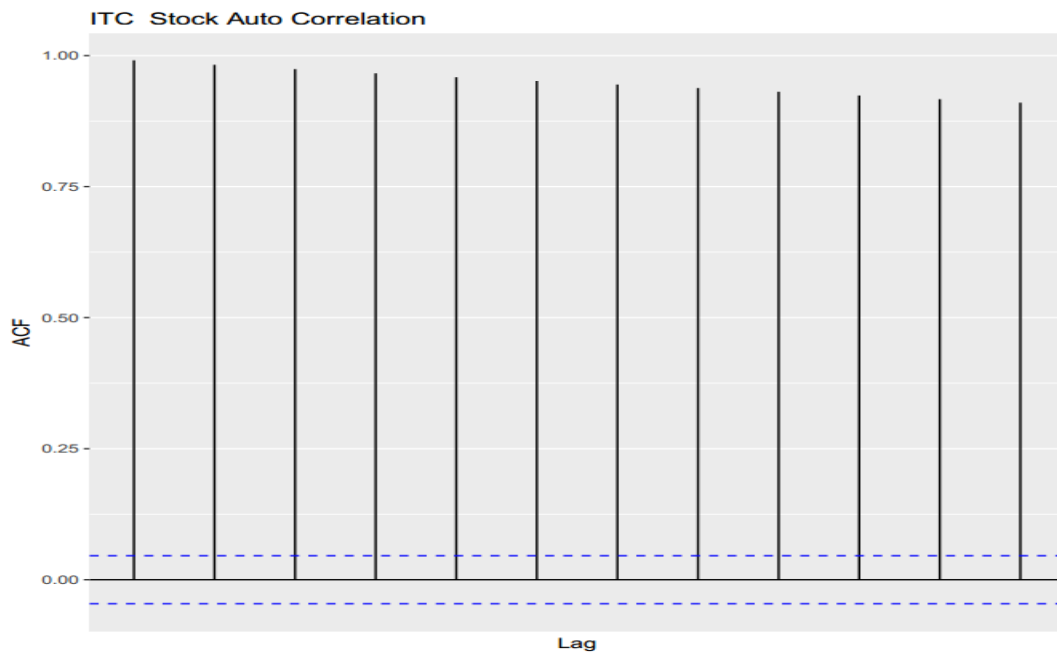


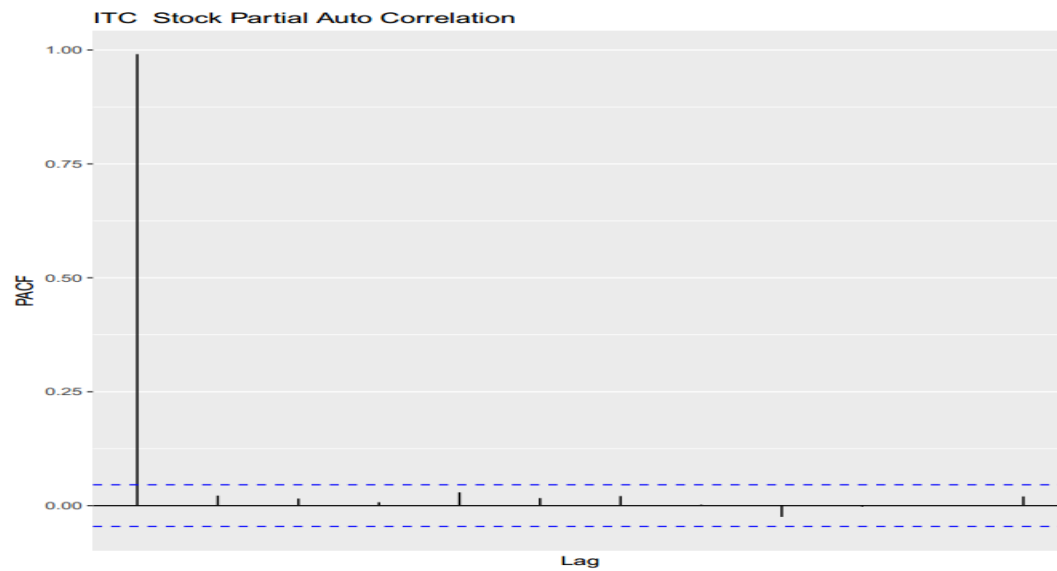
3. Random Noise:



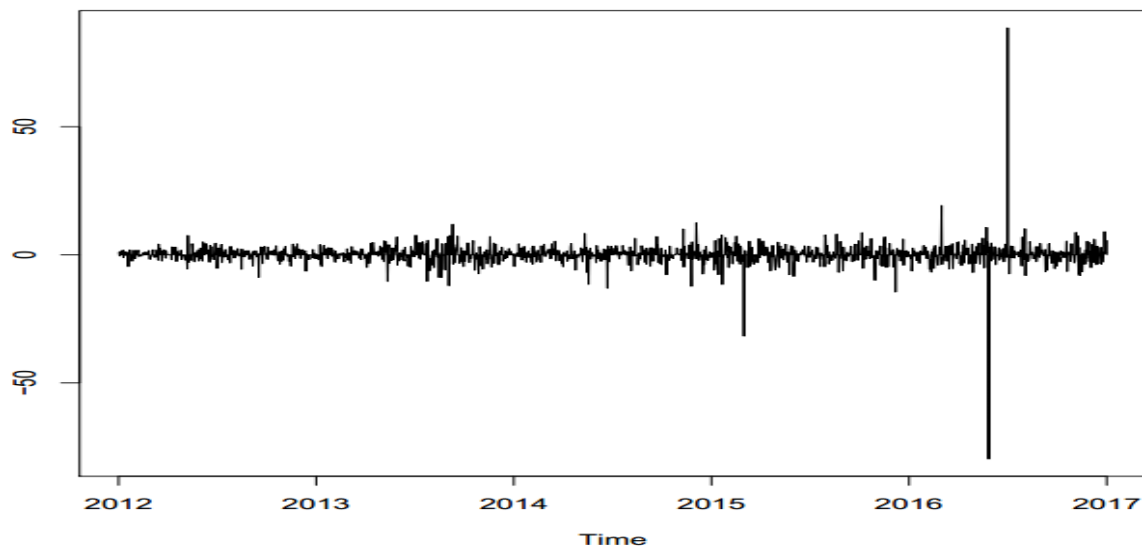
c. Correlation Analysis Results-Stock ITC

Auto Correlation Function (ACF) and Partial Auto Correlation Function (PACF) has been applied to ITC historical stock data for $h=12$ lags. ACF graph shows that significant spikes (positive auto correlations) in ACF and PACF looks fine.



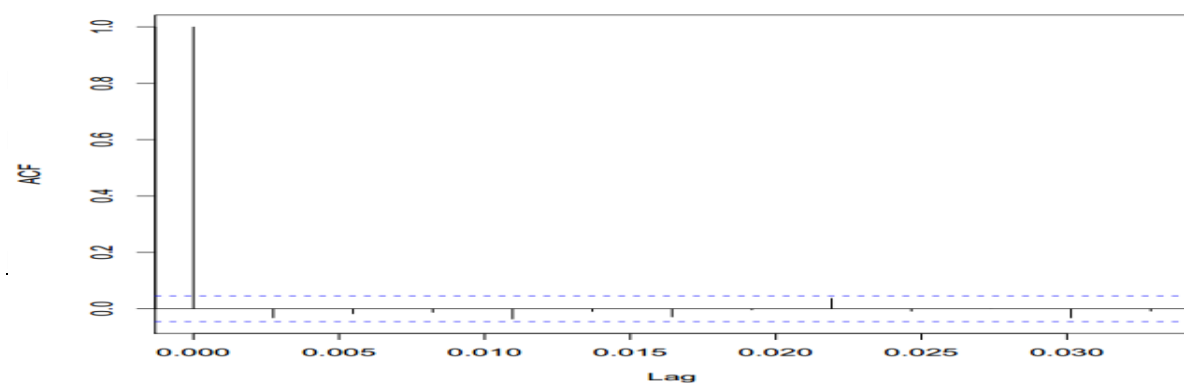


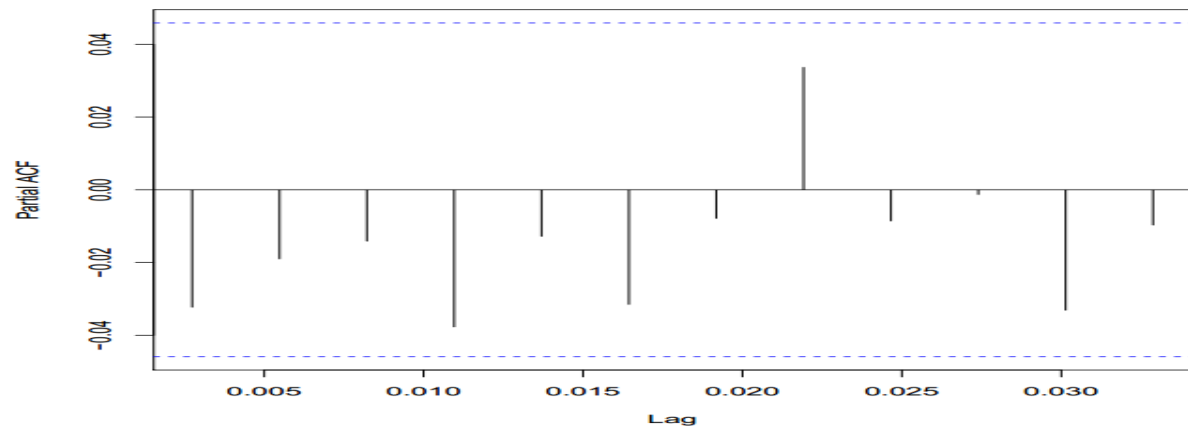
After, first time differencing: Stock price looks constant on level, as level there is no much variation on stock price.



After first differencing, ACF and PACF graphs looks fine. There is no significant spike touches or above the confident interval.

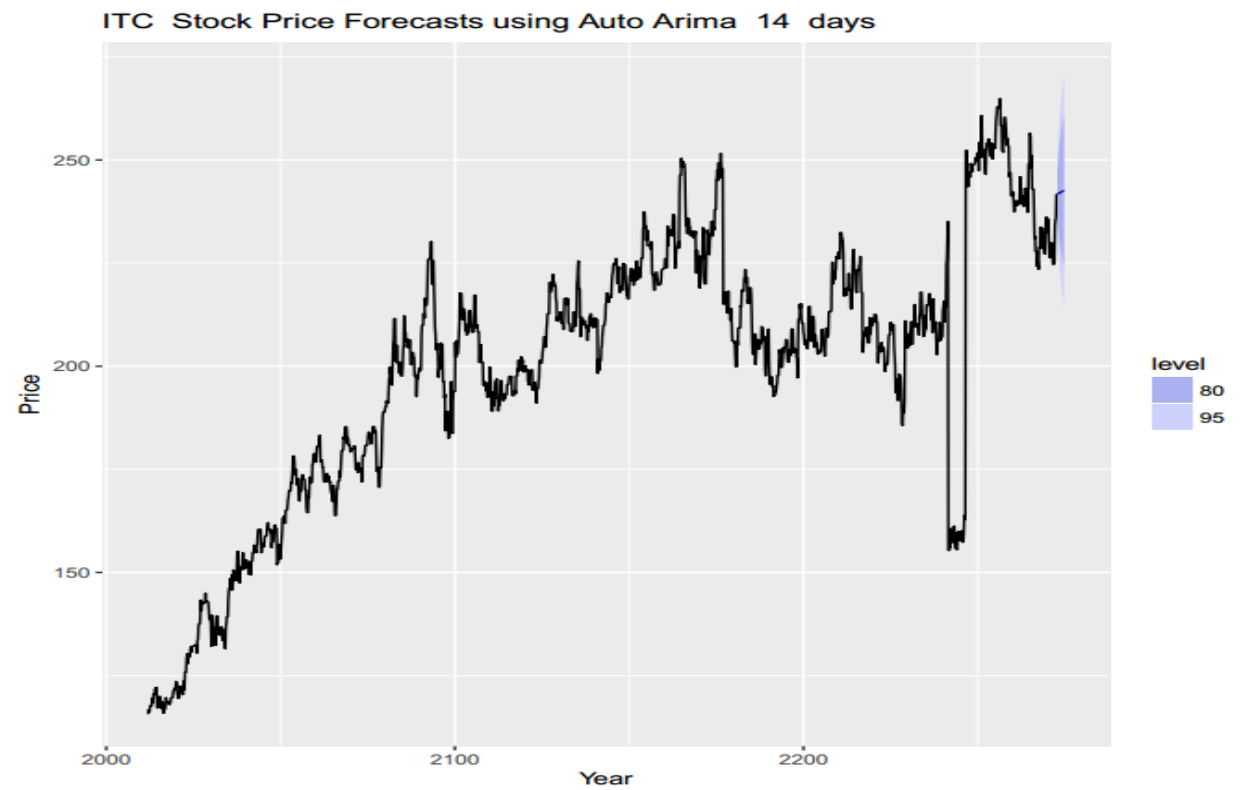
ITC Auto-Correlation graph, after first order differencing



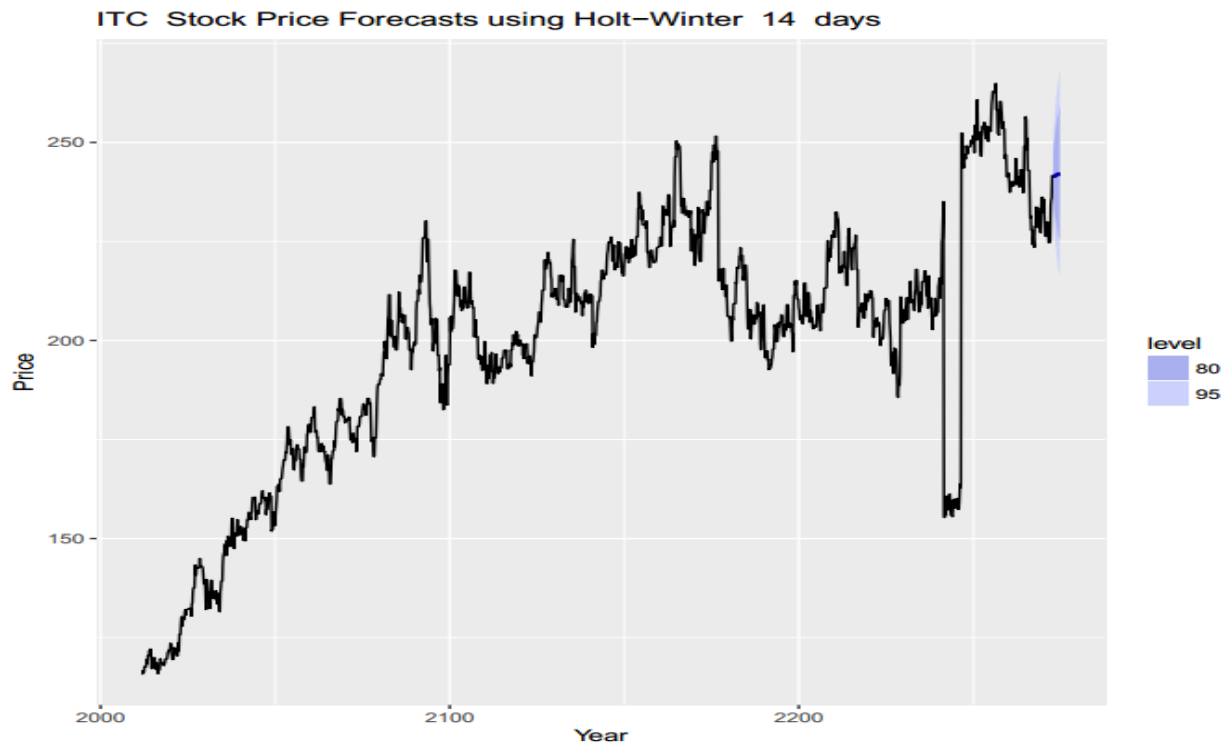


d. Stock Price forecasting using various methods

Arima Model forecasting

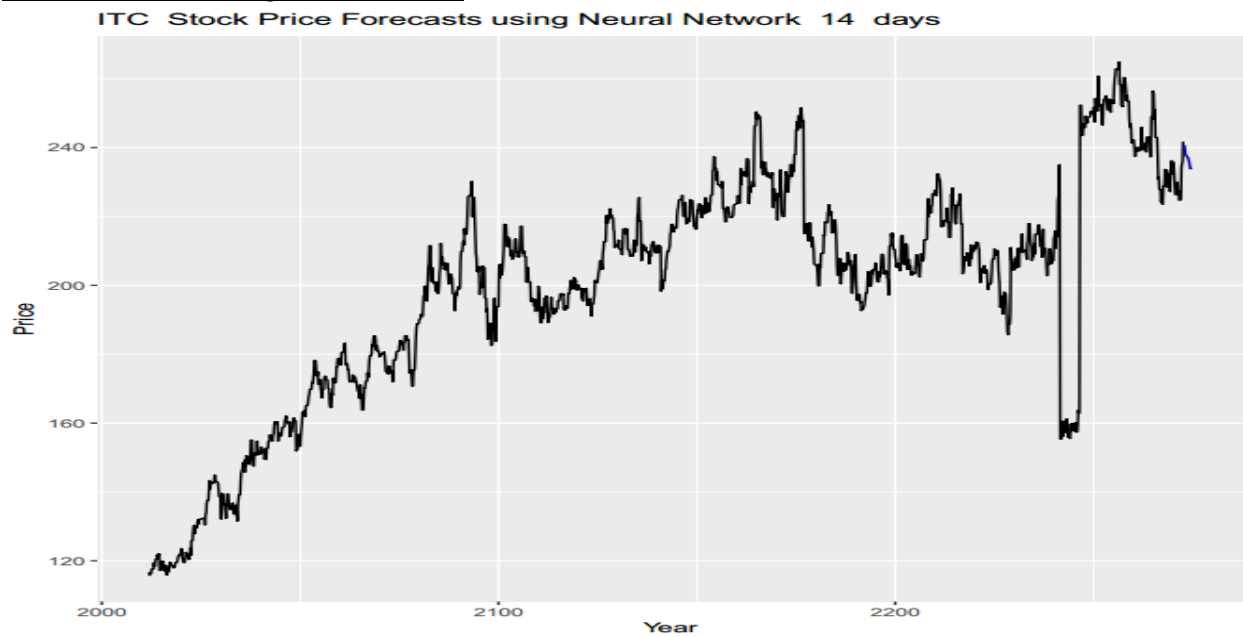


Stock Price forecasting: ITC Holt-Winters Seasonal Model



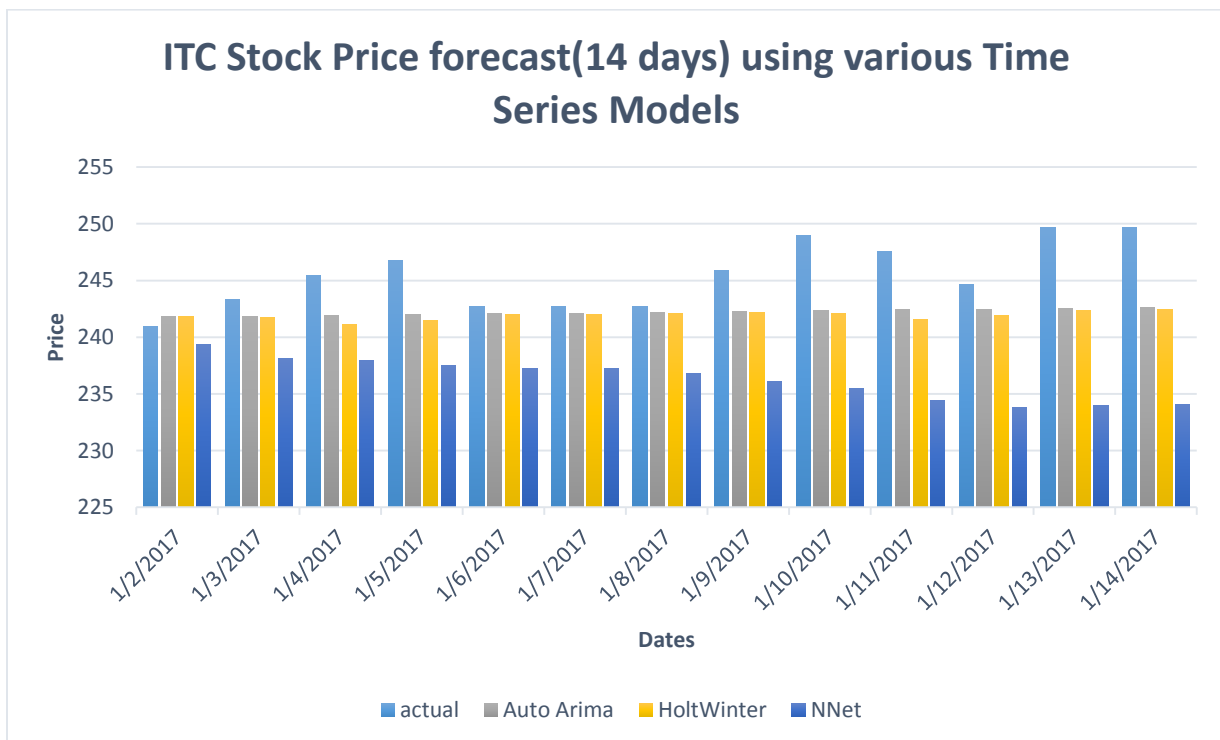
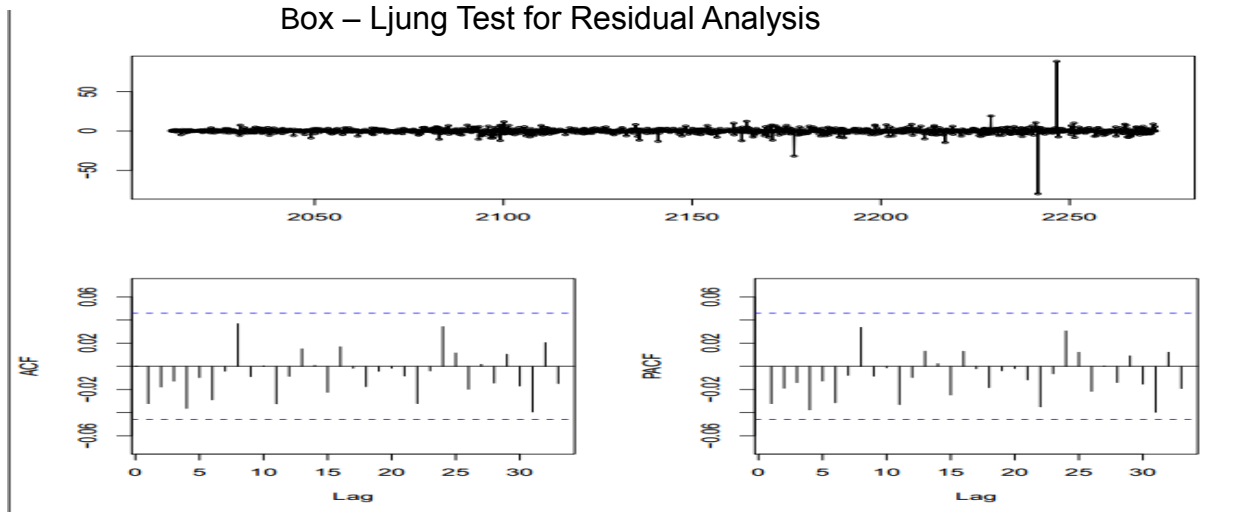
Stock Price forecasting: ITC Neural Networks Model

Stock Price forecasting all Models: ITC



e. Residual Analysis-Stock ITC

Auto Arima model outputs residuals has been analyzed using Box-Ljung test. All the spikes are now within the significance limits, and so the residuals appear to be white noise. A Ljung-Box test also shows that the residuals have no remaining autocorrelations.



f. ITC Stock Forecasting Models Performance:

The models that have the lowest AIC c values tend to give slightly better results than the other models, but there is not a large difference.

Test Results error TestMAPE for 14 days shows that Auto-Arima model forecast results is more appropriate to ITC stock price prediction and also has less AIC value. But, also there is no significant price prediction difference between Auto-Arima and Hold-Winter forecasting.

On average for all models, Auto-Arima and Holt-Winter models based price prediction looks more appropriate.

	TrainMAPE	TestMAPE	AIC	Pvalue	lag
AutoArima	0.8233056	1.366255	10094.00	0.16730602	1
HoltWinter	0.8514261	1.488546	18658.04	0.02560851	1
NeuralNet	0.8514261	1.488546	0.00	0.95846054	1

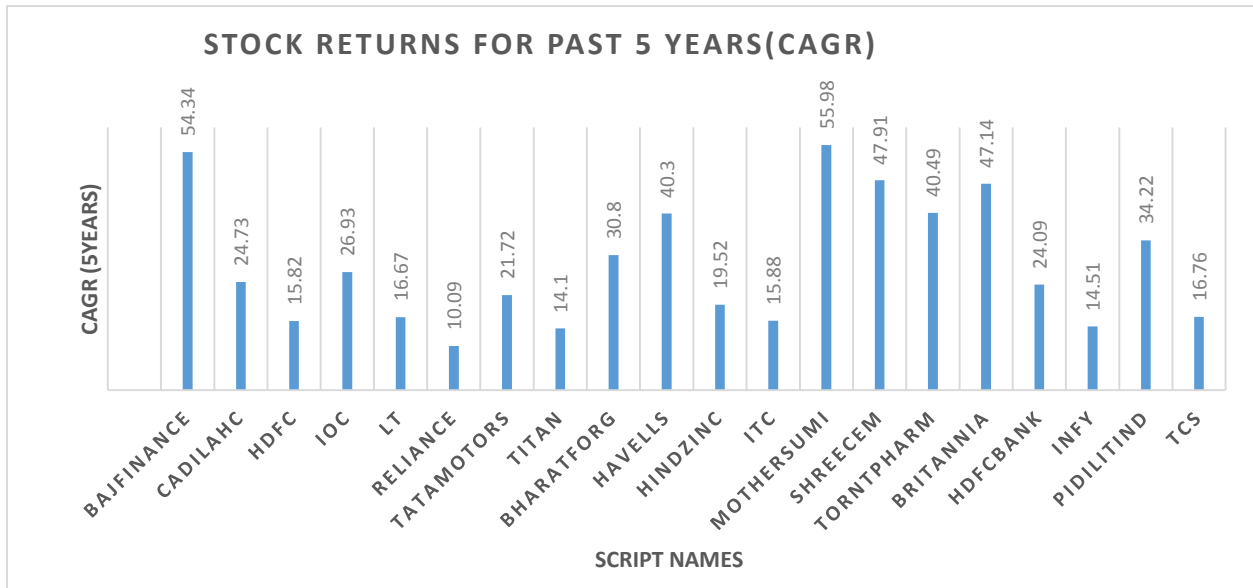
g. Return computed for all stocks:

We have downloaded 20 stocks from Nifty50 from Jan-2012 to Dec-2016, pre-processed it and computed returns using CAGR (Compound **Annual Growth Rate**) for 5.

Chosen 20 Stock Returns using CAGR from 5 years historical data:

S. No	Script Name	Buy Price	Current Price	CARG	Invest Period
1	BAJFINANCE	96.16417	842.2	54.34	5
2	CADILAH	118.12499	356.65	24.73	5
3	HDFC	605.7432	1262.6	15.82	5
4	IOC	95.2859	313.9617	26.93	5
5	LT	624.0685	1349.1	16.67	5
6	RELIANCE	669.3344	1082.4	10.09	5
7	TATAMOTORS	176.684	472	21.72	5
8	TITAN	168.9952	326.8	14.1	5
9	BHARATFORG	236.2297	904.3275	30.8	5
10	HAVELLS	62.9584	342.25	40.3	5
11	HINDZINC	104.735	255.4	19.52	5
12	ITC	115.6424	241.65	15.88	5
13	MOTHERSUMI	35.35967	326.45	55.98	5
14	SHREECEM	2081	14733.4	47.91	5
15	TORNTPHARM	238.743	1306.672	40.49	5
16	BRITANNIA	418.4815	2886.3	47.14	5
17	HDFCBANK	410	1206.2	24.09	5
18	INFY	513.2505	1010.6	14.51	5
19	PIDILITIND	135.4059	589.8	34.22	5
20	TCS	1086.846	2358.822	16.76	5

Stock Price Return using CAGR(5 years) Chart:

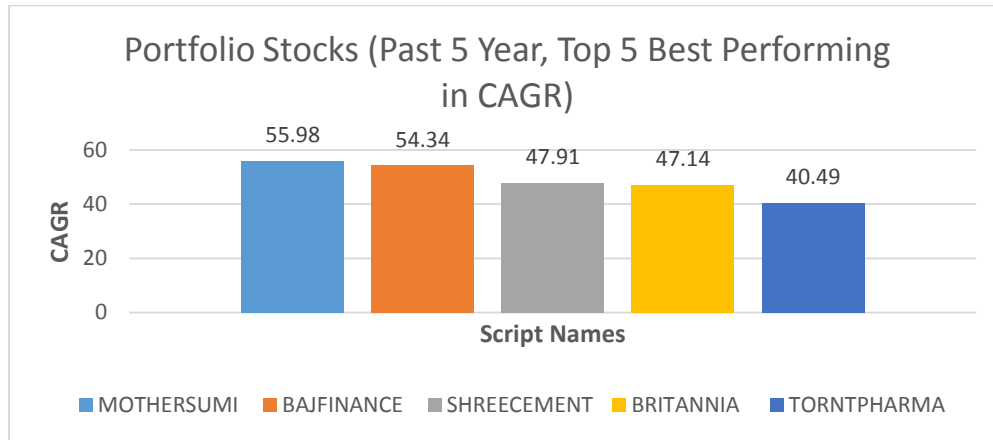


Top5 Best Performing Stocks by Return:

Script Name	CAGR(Compound Annualized Growth Rate)
BAJFINANCE	54.34
CADILAHC	24.73
HDFC	15.82
IOC	26.93
LT	16.67
RELIANCE	10.09
TATAMOTORS	21.72
TITAN	14.1
BHARATFORG	30.8
HAVELLS	40.3
HINDZINC	19.52
ITC	15.88
MOTHERSUMI	55.98
SHREECEM	47.91
TORNTPHARM	40.49
BRITANNIA	47.14
HDFCBANK	24.09
INFY	14.51
PIDILITIND	34.22
TCS	16.76

b. Portfolio optimization Output:

From historical data, Stock price prediction and Compounded Annual Growth Rate (CAGR) has been computed for all 20 stocks. From this list, best performing top 5 stocks has been chosen on CAGR basis in diversified sector and build a portfolio. Below is the chart about list of top5 best performed stocks by CAGR for 5 years.



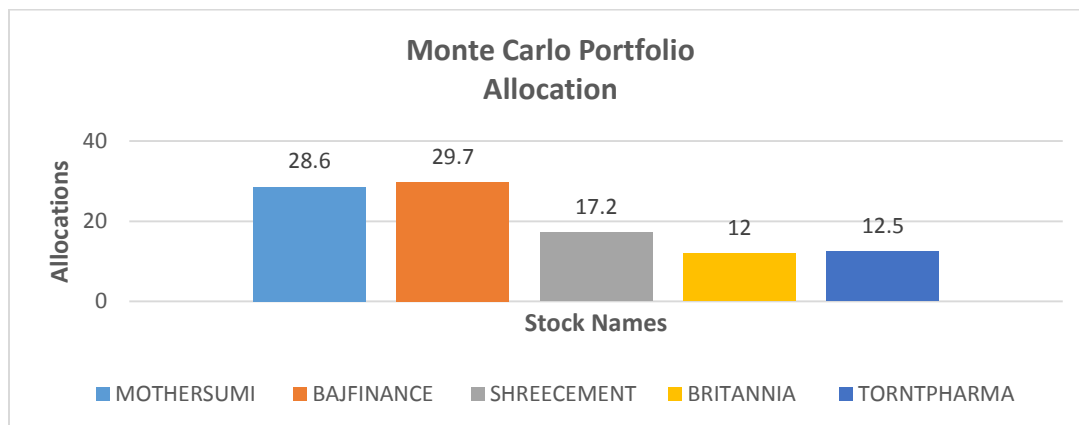
a. Monte Carlo Simulation Output:

Input:

- Number of Simulation iterations: 1,00,000
- Portfolio return period: 1 year
- Invested Amount: 1,50,000
- Initial Stock Allocation: 20% 20% 20% 20% 20%

Output:

- Optimized Stock Allocation: 28.6% 29.7% 17.2% 12% 12.5%
- Portfolio return: 76926.
- Portfolio return in percentage: 51.28



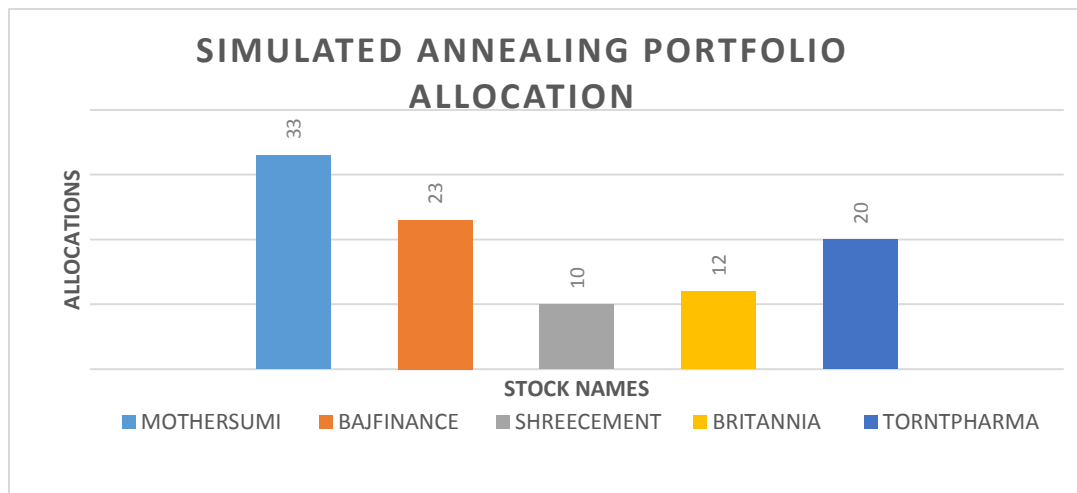
b. Simulated Annealing Output:

Input:

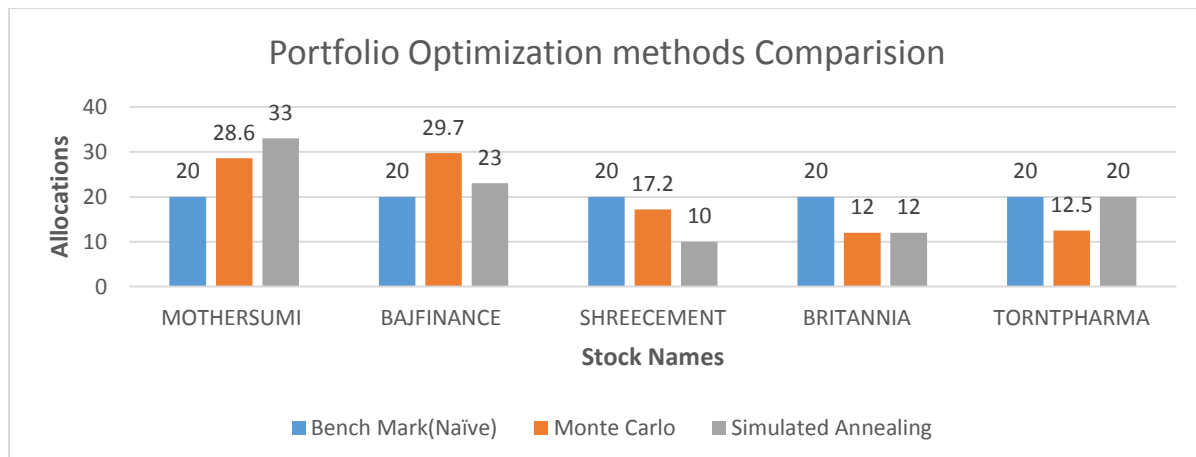
- Number of Simulation iterations: 1,00,000
- Portfolio return period: 1 year
- Invested Amount: 1,50,000
- Initial Stock Allocation: 20% 20% 20% 20% 20%

Output:

- Optimized Stock Allocation: 33% 23% 10% 12% 20%
- Portfolio return: 76946.23
- Portfolio return in percentage: 51.3



c. Portfolio Optimization methods Comparisons output:



d. Test of significance:

In statistical hypothesis testing, statistical significance (or a statistically significant result) is attained whenever the observed p-value of a test statistic is less than the significance level defined for the study. The significance level, α , is the probability of rejecting the null hypothesis, given that it is true. Here we use $\alpha=0.05$

On the results obtained, we did the Binomial upper tailed test at a threshold level of 5%. Null and Alternative Hypothesis defined as below.

Null HYPOTHESIS (Ho):

Ho: Current Stock price forecasting/prediction does not depends on previous day price or past historical data.

$$H_0: Y_t = Y_{t-1}$$

Y_t - current stock price

Y_{t-1} -Precious data stock price

Alternative HYPOTHESIS (H1):

H1: Current Stock price forecasting/prediction depends on previous day price or past historical data. If the Stock price prediction methods were properly chosen, than we can obtain satisfactory prediction results.

$$H_1: Y_t \text{ not equals to } Y_{t-1}$$

On the results obtained, we did the Binomial upper tailed test at a threshold level of 5%. We chose the significance level as 0.05, meaning that the outcome of the test has 5 percent chance of not being true. Significance level is the criterion used for rejecting the Null Hypothesis.

Result:

P-value (All top5 best performing stocks forecasted value) < 0.05: Rejected Null Hypothesis

We have considered Top5 best performing stocks price forecasted value from test data set of 14 days (1st January to 14th January 2017) and observed p-value for Arima and Holt Winter models. This is always <0.05. So we conclude that we have sufficient evidence to reject Null Hypothesis(H_0) and accepted alternative Hypothesis(H_1).

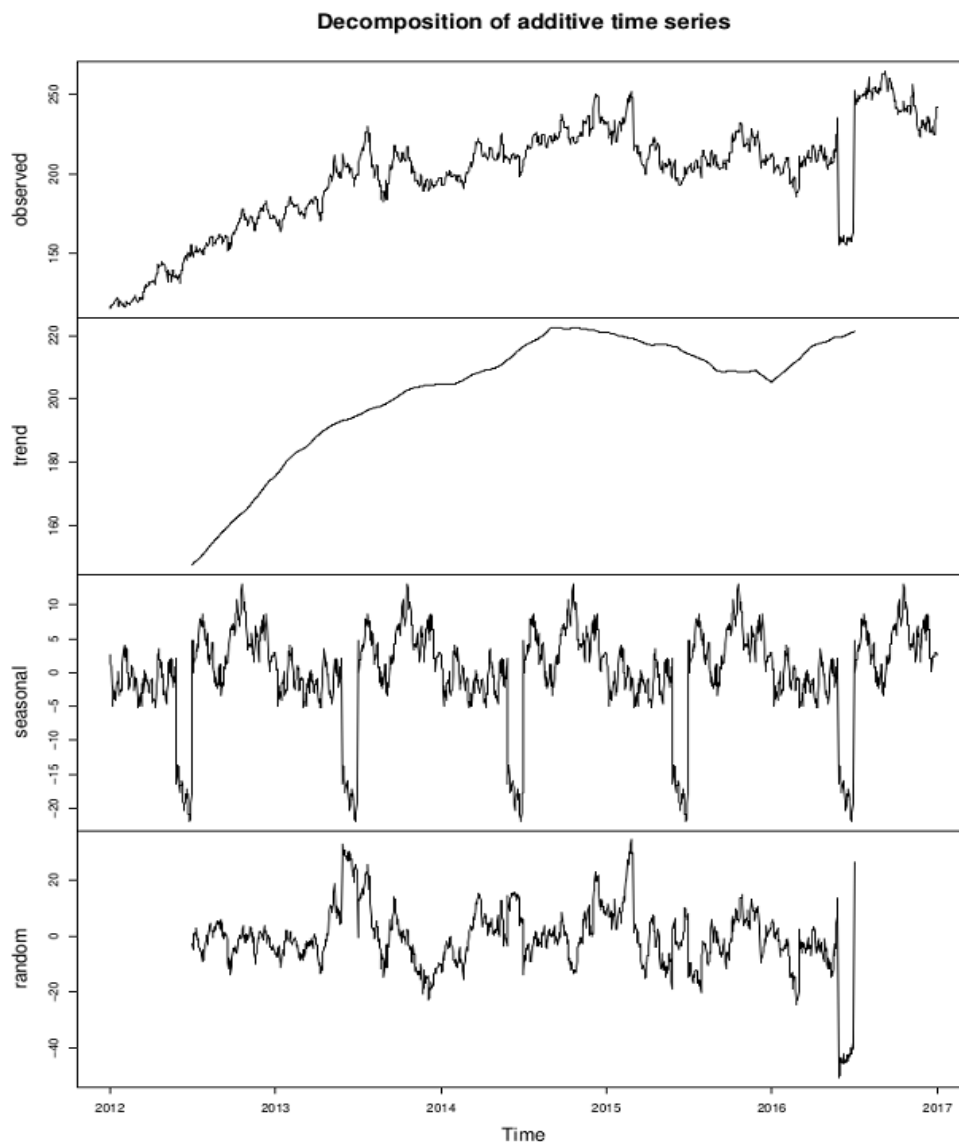
MOTHERSON SUMI					
S No	TrainMAPE	TestMAPE	AIC	P value	lag
AutoArima	1.130375	4.63151	17606.36	0.03743352	10
HoltWinter	1.375845	4.193763	26163.56	0.03673535	10
NeuralNet	1.375845	4.193763	0	0.75297251	7
BAJAJ FINANCE					
S No	TrainMAPE	TestMAPE	AIC	P value	lag
AutoArima	1.130375	4.63151	17606.36	0.03743352	10
HoltWinter	1.375845	4.193763	26163.56	0.03673535	10
NeuralNet	1.375845	4.193763	0	0.75297251	7
SHREE CEMENT					
S No	TrainMAPE	TestMAPE	AIC	P value	lag
AutoArima	0.9531933	2.507103	23297.45	0.0343341	11
HoltWinter	0.9367115	2.60806	31857.84	0.01751826	11
NeuralNet	0.9367115	2.60806	0	0.17911207	11
BRITANIA					
S No	TrainMAPE	TestMAPE	AIC	P value	lag
AutoArima	0.8997914	0.7782715	17311.08	0.01418176	3
HoltWinter	0.8930504	0.7867145	25866.8	0.01396393	3
NeuralNet	0.8930504	0.7867145	0	0.67838399	19
TORRENT PHARMA					
S No	TrainMAPE	TestMAPE	AIC	P value	lag
AutoArima	0.9200709	2.268834	15157.37	0.0426534	16
HoltWinter	0.9108983	2.190313	23708.83	0.01797945	16
NeuralNet	0.9108983	2.190313	0	0.07086692	16

6. Analysis

a. Analysis of Stock Price Prediction

i. stationary and differencing

A *stationary* time series is one whose statistical properties such as mean, variance, autocorrelation, etc. are all constant over time. Most statistical forecasting methods are based on the assumption that the time series can be rendered approximately stationary (i.e., "stationarized") through the use of mathematical transformations. A stationarized series is relatively easy to predict: you simply predict that its statistical properties will be the same in the future as they have been in the past! The predictions for the stationarized series can then be "untransformed," by reversing whatever mathematical transformations were previously used, to obtain predictions for the original series [5].

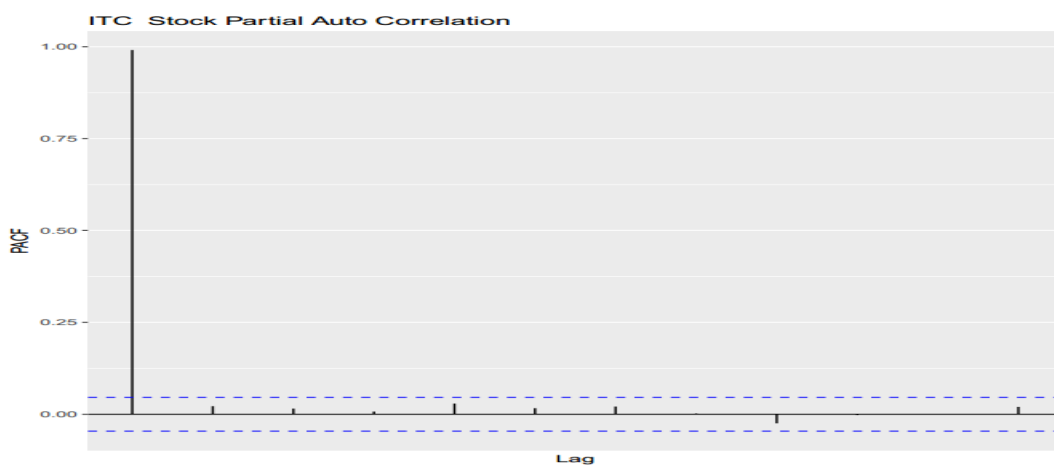
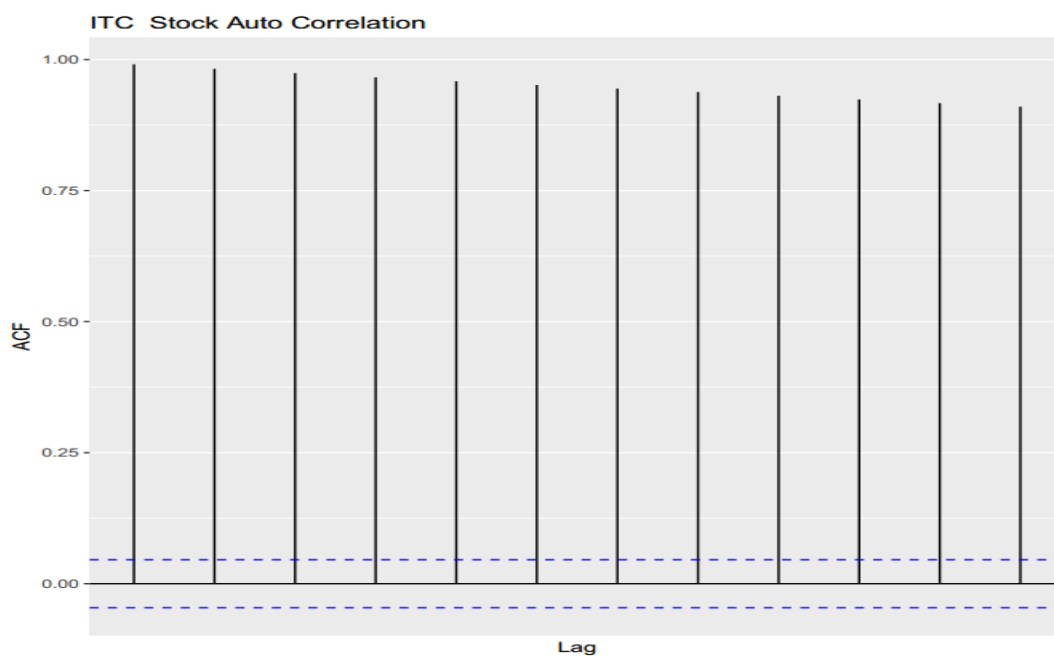


After decomposition, historical ITC Stock price shows that there is trend and seasonality. So, in order to make it stationary it is needed to be different

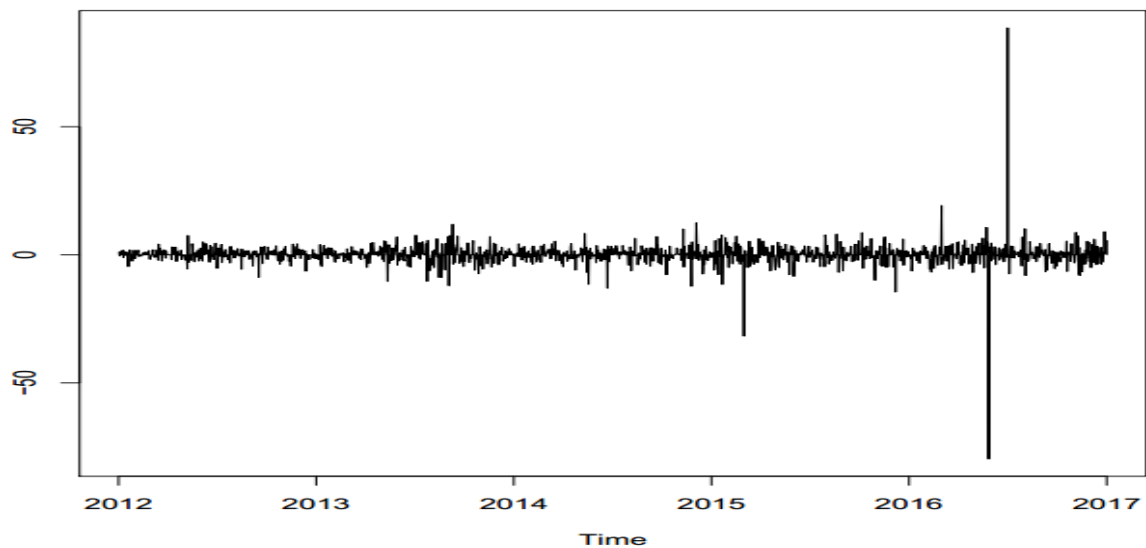
The **first difference** of a time series is the series of changes from one period to the next. If Y_t denotes the value of the time series Y at period t , then the first difference of Y at period t is equal to $Y_t - Y_{t-1}$.

ii. Correlation Analysis:

Auto Correlation Function (ACF) and Partial Auto Correlation Function (PACF) has been applied to ITC historical stock data for $h=12$ lags. ACF graph shows that many significant spikes in ACF plot. So this correlated time

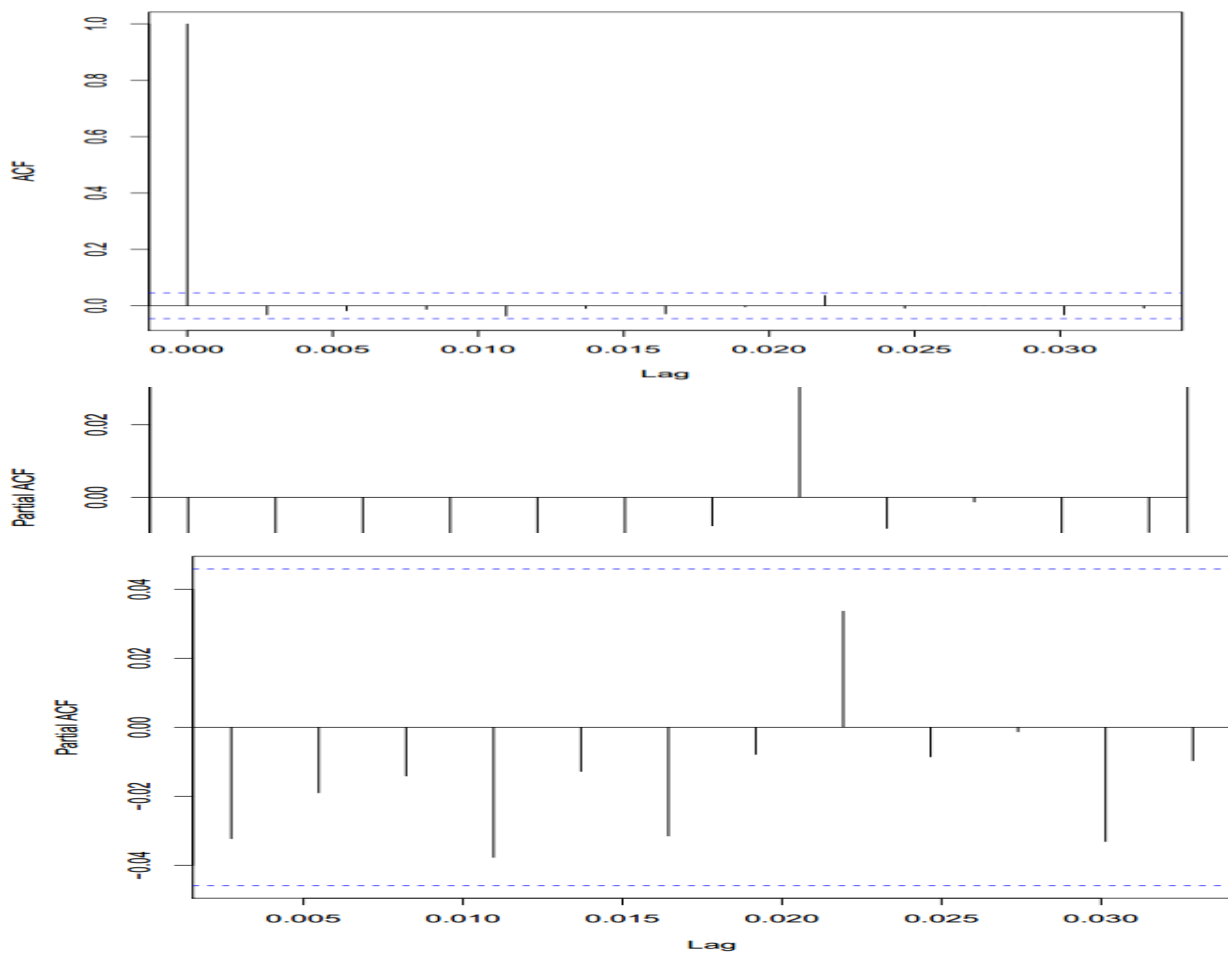


After, first time differencing: Stock price looks constant on level, as level there is no much variation on stock price.



After first differencing, ACF and PACF graphs looks fine. There is no significant spike touches or above the confident interval.

ITC Auto-Correlation graph, after first order differencing



iii. Residual Analysis:

The residuals from a regression model are calculated as the difference between the actual values and the fitted values: $e_i = y_i - \hat{y}_i$. Each residual is the unpredictable component of the associated observation.

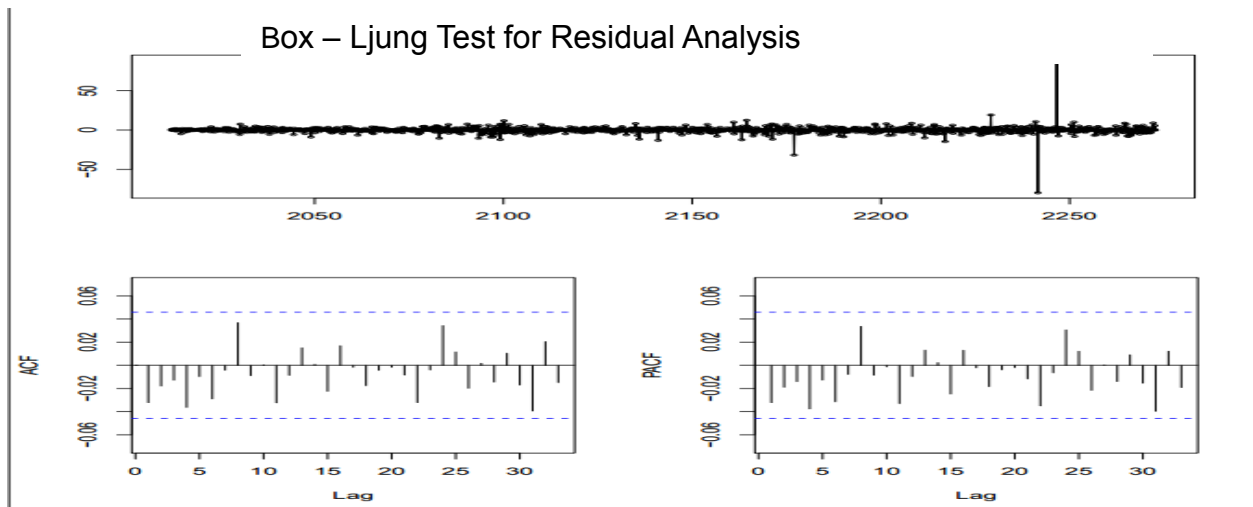
Autocorrelation in residuals:

When the data are a time series, you should look at an ACF plot of the residuals. This will reveal if there is any autocorrelation in the residuals (suggesting that there is information that has not been accounted for in the model).

Arima Model Residual Analysis:

Auto Arima model outputs residuals has been analyzed using Box-Ljung test. All the spikes are now within the significance limits, and so the residuals appear to be white noise. A Ljung-Box test also shows that the residuals have no remaining autocorrelations.

The following figure shows a time plot and ACF of the residuals from the model fitted to historical stock price data better.



iv. Stock Forecasting Models Performance Analysis:

ITC Price Forecasting Models:

	date	Arima	Auto Arima	Holt Winter	Neural Network
2	1/2/2017	241.65	241.7881	241.7857	239.3881
3	1/3/2017	241.65	241.8571	241.6957	238.134
4	1/4/2017	241.65	241.9262	241.1637	237.9088
5	1/5/2017	241.65	241.9952	241.4897	237.5413
6	1/6/2017	241.65	242.0643	241.961	237.2184
7	1/7/2017	241.65	242.1333	242.0133	237.2451
8	1/8/2017	241.65	242.2024	242.1154	236.7733
9	1/9/2017	241.65	242.2714	242.1818	236.0898
10	1/10/2017	241.65	242.3405	242.0918	235.4838
11	1/11/2017	241.65	242.4095	241.5598	234.3964
12	1/12/2017	241.65	242.4785	241.8858	233.8253
13	1/13/2017	241.65	242.5476	242.3571	233.9644
14	1/14/2017	241.65	242.6166	242.4094	234.0727

Analysis of performance of various models for ITC Stock Price Prediction:

	TrainMAPE	TestMAPE	AIC	Pvalue	lag
AutoArima	0.8233056	1.366255	10094.00	0.16730602	1
HoltWinter	0.8514261	1.488546	18658.04	0.02560851	1
NeuralNet	0.8514261	1.488546	0.00	0.95846054	1

Final Results:

- The models that have the lowest AIC c values tend to give slightly better results than the other models, but there is not a large difference.
- If MAPE is measure of model performance, then it is clear that Arima performs better for short time price prediction. But on average, there is no much significant difference between Arima and Holt-Winter models.
- From ITC stock price prediction, it is clear that , Overall Forecasting Models Performance report states for short Term forecasting (1-2weeks) **Arima or Holt-Winter model** performed well, than neural network based model

b. Analysis of Portfolio Optimization:

i. Bench mark portfolio analysis:

Input and constrains:

- Invest equal allocation to all stocks. 20% each stocks
- Chosen Stocks: MOTHERSUMI, BAJFINANCE, SHREECEMENT, BRITANNIA, TORNTPHARMA
- Investment amount: 1,50,000
- Investment period: 1 year
- Minimum Investment : 10% (per stock)
- Maximum Investment : 30% (per stock)

Output Analysis:

- Portfolio return: 73758
- Portfolio return in percentage: 49.1

Result:

Benchmark portfolio analysis models state that 49% return on investment if equal amount of 20% has been invested in chosen 5 stocks.

ii. Monte Carlo simulation analysis:

Input and constrain:

- Number of Simulation iterations: 1,00,000
- Portfolio return period: 1 year
- Invested Amount: 1,50,000
- Initial Stock Allocation: 20% 20% 20% 20% 20%
- Minimum Investment : 10% (per stock)
- Maximum Investment : 30% (per stock)

Output:

- Optimized Stock Allocation: 28.6% 29.7% 17.2% 12% 12.5%
- Portfolio return: 76926.
- Portfolio return in percentage: 51.28

Result:

- This simulation has been performed for various iterations such as 1000, 10000 and 100000 and provided 1, 00,000 as stopping criteria.
- After 10000 iterations there is very less change in portfolio return in percentage that can be discarded (≤ 0.2). So simulation has been stopped at 1, 00,000 iterations.
- We conclude that optimized Stock Allocation: 28.6% 29.7% 17.2% 12% 12.5% is global optimal solution and this yield return of 51.28% for 1 year.

iii. Simulated Annealing Analysis

Input and constrain:

- Number of Simulation iterations: 1,00,000
- Portfolio return period: 1 year
- Invested Amount: 1,50,000
- Initial Stock Allocation: 20% 20% 20% 20% 20%
- Minimum Investment : 10% (per stock)
- Maximum Investment : 30% (per stock)

Output:

- Optimized Stock Allocation: 33% 23% 10% 12% 20%
- Portfolio return: 76946.23
- Portfolio return in percentage: 51.3

Result:

- This simulation has been performed for various iterations such as 1000, 10000 and 100000 and stopping criteria is provided as 1, 00,000 iterations.
- At the end of Simulations, we conclude that Optimized Stock Allocation: 33% 23% 10% 12% 20% as global optimal solution(practical) and this allocation yield return as 51.3% for 1 year.

c. Improvements:

- Stock Prediction:
 - Experiment SVM with technical indicators RSI, MCAD, EMA as independent variables and compare metrics with existing models Arima and Holt-Winter.
- Portfolio Optimization :
 - Consider Genetic Algorithm and Quadratic Programming in near term and compare those stock allocation and return with existing simulations.

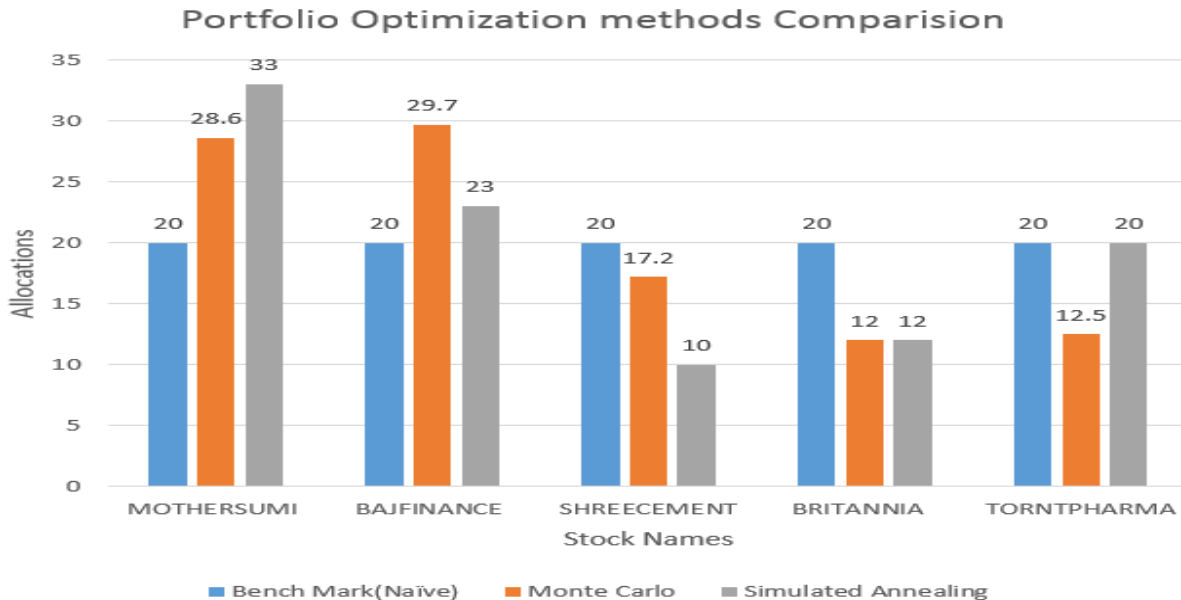
d. Lessons Learned:

- Simulations techniques are powerful techniques for portfolio optimization.
- Time series models ARIMA and Holt-Winter are good for short term forecasting.
- R Studio is best IDE for Development project in R language.
- Explored quandmod(not used), forecast, fpp, ggplot2 are nice tools for Time series forecasting and plotting graphs.

e. Observation and conclusion:

Stock Price forecasting:

- Analyzed stock price forecasting for 20 stocks in Nifty50.
- Overall Forecasting Models Performance report states for short Term forecasting (1-2weeks) Arima or Holt-Winter model perform well.



Portfolio Optimization:

- Monte Carlo simulation with 1, 00,000 iterations and simulated annealing with same iterations provided 51.28% and 51.3% returns for 1 year for invested amount 1, 50, 000.
- Simulations solutions signifies that chosen stocks and allocation are optimal, worthwhile to stick to any one of these approach for long Term investment.

7. References

- [1] . Masih, A. M. M. & Masih, R. (1997). A comparative analysis of the propagation of stock market fluctuations in alternative models of dynamic causal linkages. Applied Financial Economics, 7 (1), 59-74.
- [2]. Nifty50 companies list https://www.nseindia.com/products/content/equities/indices/nifty_50.htm
- [3] . H. Markowitz, "Portfolio Selection," Journal of Finance, vol. 7, no. 1, pp. 77-91, 1952
- [4]. Test of significance, <http://www.stat.yale.edu/Courses/1997-98/101/sigtest.htm>
- [5]. ARIMA components, <https://people.duke.edu/~rnau/411diff.htm>

8. Appendices

ProjectMain_Stock_Price_Portfolio_Opt.R

Project: STOCK PRICE PREDICTION AND PORTFOLIO OPTIMIZATION

Name: Thamizhannal Paramasivam

Batch No: 22

Note: Load this project as R project in RStudio and Run it.

STOCK PRICE PREDICTION

rm(list=ls(all=TRUE))

Input Data: Downloaded 20 stocks from Nifty50

Train Data: Duration: start_date: 2012-01-01 end_date: 2016-12-31

Test Data: Duration: start_date: 2017-01-02 end_date: 2017-01-14

Output: Please refer output folder for

1. <script_name>.pdf - ITC.pdf - contains all the plots about ITC

2. <script_name__forecast_metrics.Rda> - ITC_forecast_metrics.Rda all the forecast metrics

3. <script_name_ITC_models_metrics.Rda> - ITC_models_metrics.Rda contains model metrics

4. all_script_return.Rda - contains all stocks return for 5 years.

source("R/StockMainProg.R")

PORTFOLIO OPTIMIZATION

METHOD-1: BENCH MARK portfolio return

> naive_portfolio_ret()

[1] "Naive Portfolio Return computation"

[1] "Top stocks return data:"

stocks returns

MOTHERSUMI MOTHERSUMI 55.98

BAJFINANCE BAJFINANCE 54.34

SHREECEM SHREECEM 47.91

```

# BRITANNIA BRITANNIA 47.14
# TORNTPHARM TORNTPHARM 40.49

# [1] "Stock Allocation:"
# [1] 0.2 0.2 0.2 0.2 0.2
# [1] "portfolio return: 73758"
# portfolio return in percentage: 49.1


# METHOD-2: MonteCarlo Simulation
# assumptions:
# start_date = "2012-01-02"
# end_date = "2016-12-31"
# min_investment = 0.10 # 10% per eacg stock
# Max inviestment is not more than 30% on each stock in portfolio
# max_investment = 0.30


rm(list=ls(all=TRUE))

# Run Portfolio Optimization using MonteCarlo Simulation
source("R/OptimizationMonteCarlo.R")

# OUTPUT:
# Portfolio return for 1 year
# Invested Amount: 15000
# Optimized Stock Allocation: 0.286 0.2970 0.170 0.120 0.125
# portfolio return: 76926.2
# portfolio return in percentage: 51.28
# Simulation Iteration: 1e+05
#"Top stocks return data:"
# stocks returns
# MOTHERSUMI MOTHERSUMI 55.98

```

```
# BAJFINANCE BAJFINANCE 54.34
```

```
# SHREECEM SHREECEM 47.91
```

```
# BRITANNIA BRITANNIA 47.14
```

```
# TORNTPHARM TORNTPHARM 40.49
```

```
## METHOD-3: SIMULATED ANNEALING OPTIMIZATION SOLUTIONS
```

```
# ASSUMPTIONS:
```

```
# Portfolio return for 1 year
```

```
# start_date = "2012-01-02"
```

```
# end_date = "2016-12-31"
```

```
# min_investment = 10% per each stock
```

```
# Max investment is not more than 30% on each stock in portfolio
```

```
# max_investment = 30% per each stock
```

```
# max_iterations = 100000
```

```
# Investment amount = 150000
```

```
# Chosen Top5 stocks returns
```

```
# MOTHERSUMI MOTHERSUMI 55.98
```

```
# BAJFINANCE BAJFINANCE 54.34
```

```
# SHREECEM SHREECEM 47.91
```

```
# BRITANNIA BRITANNIA 47.14
```

```
# TORNTPHARM TORNTPHARM 40.49
```

```
rm(list=ls(all=TRUE))
```

```
# Run Portfolio Optimization using SA
```

```
source("R/OptimizationSimulatedAnnealing.R")
```

```
# OUTPUT:
```

```
# Invested Amount: 15000
```

```
# Optimized Stock Allocation: 0.33 0.23 0.10 0.12 0.20
```

```
# portfolio return: 76443.23
```

```
# portfolio return in percentage: 51.3
```

```
# Total returns = 76946.61
```

```
#####
```

```
StockMainProg.R
```

```
# Project: STOCK PRICE PREDICTION AND PORTFOLIO OPTIMIZATION
```

```
# Name: Thamizhannal Paramasivam
```

```
# Batch No: 22
```

```
#
```

```
# Note: Load this project as R project in RStudio and Run it.
```

```
##### STOCK PRICE PREDICTION - MAIN PROGRAM
```

```
#####
```

```
rm(list=ls(all=TRUE))
```

```
# Loading Library
```

```
library(fpp)
```

```
library(ggplot2)
```

```
library(ggfortify) # Extended ggplot2
```

```
# Default working dir is package folder, com.insofe.stock
```

```
#setwd(getwd())
```

```
print(getwd())
```

```
# Include source from R files
```

```
source("R/preprocess.R")
```

```
source("R/timeseries.R")
```

```
source("R/Utils.R")
```

```
source("R/const.R")
```

```

# create

data_path = paste(getwd(),"data",sep="/")
out_path = paste(getwd(),"output",sep="/")


# Script return calculator

script_return = data.frame(matrix(nrow=20,ncol=5))

colnames(script_return) = c("script", "buy_price", "curr_price",
                             "CARG", "invest_period" )


all_script_list = c("BAJFINANCE.csv", "CADILAHK.csv", "HDFC.csv", "IOC.csv", "LT.csv",
                    "RELIANCE.csv", "TATAMOTORS.csv", "TITAN.csv", "BHARATFORG.csv",
                    "HAVELLS.csv", "HINDZINC.csv", "ITC.csv", "MOTHERSUMI.csv",
                    "SHREECEM.csv", "TORNTPHARM.csv", "BRITANNIA.csv", "HDFCBANK.csv", "INFY.csv",
                    "PIDILITIND.csv", "TCS.csv")

#all_script_list = c("ITC.csv")

# Number of days to forecast a stock price

h=14

# Investment period in years, default = 5 years starting from 2012 to 2016.

invest_period = 5


# Train data - Historical Stock price Stating date

#start_date = "2012-01-02"

start_date = "2012-01-02"

# Historical Stock price End date

end_date = "2016-12-31"


# Test data -

test_start_date = "2017-01-01"

```



```
# Historical Stock price End date
```

```
test_end_date = "2017-01-14"
```

```
# Script index
```

```
idx = 0
```

```
for(script in all_script_list) {
```

```
    idx = idx + 1
```

```
    script_name_pdf = sub(pattern = ".csv", replacement = ".pdf", x=script)
```

```
    script_name = sub(pattern = ".csv", replacement = "", x=script)
```

```
    script_path = paste(data_path,script,sep="/")
```

```
    script_out_path = paste(out_path,script_name_pdf,sep="/")
```

```
    row_idx=1
```

```
    # For Storing Model metrics
```

```
    all_models_metrics = data.frame(matrix(c(0),nrow=3,ncol=5 ))
```

```
    colnames(all_models_metrics) = c("TrainMAPE", "TestMAPE", "AIC", "Pvalue", "lag")
```

```
    rownames(all_models_metrics) = c("AutoArima", "HoltWinter", "NeuralNet")
```

```
    # For storing forecast metrics
```

```
    res_forecast = data.frame(matrix(c(0),nrow=h,ncol=6))
```

```
    colnames(res_forecast) = c("date", "actual", "arima", "auto_arima", "HoltWinter", "NNet")
```

```
    # TODO: File Existence check
```

```
    # 1. Load Script from CSV File and Preprocess Stock Data
```

```
    # a) Extract stock data and adjustedClosing price from CSV
```

```
    # b) Merge this extracted stock date and AdjClose price with continous date. Make Holiday price as NA.
```

```
    # c) Impute weekends and Nifty Holidays stock price as last working day price
```

```

stock_data = proc_stock_data(script_path)
stock_ts_freq7 = stock_data_ts(stock_data, freq=7,script_name )

# Compute script return for 5 years and update it script_return data frame
ret = comp_stock_return(script, stock_data, start_date, end_date, invest_period)
script_return$script[idx] = ret$name
script_return$buy_price[idx] = ret$buy
script_return$curr_price[idx] = ret$curr
script_return$invest_period[idx] = ret$period
script_return$CARG[idx] = round(ret$CARG,2)
sprintf("script: %s : Retrun: %s",ret$name,round(ret$CARG,2))

```

#. 2. Get TS data format

```

stock_ts = stock_data_ts(stock_data, 365,script_name)
ts_decom = stock_ts_decom(stock_ts, TRUE,script_name)

```

```

plot_acf_pacf(stock_ts,lag=12,script_name)

```

ndiffs() -Arima-order diff

To determine the appropriate number of first differences required for a non-seasonal TS.

```

ndiff_valae= ndiffs(stock_ts)

```

Arima - seasonal diff component.

determining whether seasonal differencing is required is nsdiffs()

```

nsdiff = nsdiffs(stock_ts)

```

```

stock_ts_diff1= stock_ts_diff(stock_ts,diff=ndiff_valae, plot=TRUE,script_name)

```

```

plot_acf_pacf(stock_ts_diff1,lag=12,script_name)

```

```

##### Prepare Test Data #####

stock_test_data = proc_stock_test_data(script_path,
                                       from =as.Date(test_start_date),
                                       to=as.Date(test_end_date))

res_forecast$actual = stock_test_data$adjclose
res_forecast$date = stock_test_data$date


print(paste("Test Data set: start date:", start_date," End Date:", test_end_date))

##### RUN MODELS
#####

##### 1. Run manual arima model
#####

arima_model = run_manual_arima(stock_ts,
                               order = c(0,ndiff_valae,0),
                               seasonal = c(0,nsdiff,0) )

arima_model

forecast_arima = model_forecast(object = arima_model,h=h)


forecast_arima

res_forecast$arima = forecast_arima$mean


# plot residuals - TODO

display_residual(arima_model)

# Run Box-Test and Identify lag for small p-value

res_box_test = run_box_test(arima_model)

sprintf("Box-Test result: small p-value:%f found at lag=%d",res_box_test$p,res_box_test$lag)

acc = accuracy(forecast_arima, stock_test_data$adjclose)

```

```
#update_res_metrics(all_models_metrics, aic=arima_model$aic, acc, res_box_test, row_idx)
```

```
##### 2. Run Auto Arima Model #####
```

```
auto_arima_model = run_auto_arima(stock_ts_freq7)
```

```
forecast_auto_arima = model_forecast(object = auto_arima_model,h=h)
```

```
res_forecast$auto_arima = forecast_auto_arima$mean
```

```
# plot residuals - TODO
```

```
display_residual(forecast_auto_arima)
```

```
# Run Box-Test and Identify lag for small p-value
```

```
res_box_test = run_box_test(forecast_auto_arima)
```

```
sprintf("Box-Test Result: small p-value:%f found at lag=%d",res_box_test$p,res_box_test$lag)
```

```
acc = accuracy(forecast_auto_arima, stock_test_data$adjclose)
```

```
#update_res_metrics(all_models_metrics, aic=auto_arima_model$aic, acc, res_box_test, row_idx)
```

```
all_models_metrics$TrainMAPE[row_idx] = acc[1,5]
```

```
all_models_metrics$TestMAPE[row_idx] = acc[2,5]
```

```
all_models_metrics$AIC[row_idx] = auto_arima_model$aic
```

```
all_models_metrics$pvalue[row_idx] = res_box_test$p
```

```
all_models_metrics$lag[row_idx] = res_box_test$lag
```

```
##### 3. Run Holt-Winter Model #####
```

```
hw_md1 = hw(stock_ts_freq7, seasonal = "additive")
```

```
summary(hw_md1)
```

```
forecast_hw = model_forecast(object = hw_md1,h=h)
```

```
forecast_hw
```

```
res_forecast$HoltWinter = forecast_hw$mean
```

```
# plot residuals - TODO
```

```

display_residual(hw_mdl)

# Run Box-Test and Identify lag for small p-value

res_box_test = run_box_test(hw_mdl)

sprintf("Box-Test Result: small p-value:%f found at lag=%d",res_box_test$p,res_box_test$lag)

acc = accuracy(forecast_hw, stock_test_data$adjclose)


row_idx = row_idx + 1

#update_res_metrics(all_models_metrics,aic=hw_mdl$model$aic , acc, res_box_test, row_idx)

all_models_metrics$TrainMAPE[row_idx] = acc[1,5]
all_models_metrics$TestMAPE[row_idx] = acc[2,5]
all_models_metrics$AIC[row_idx] = hw_mdl$model$aic
all_models_metrics$Pvalue[row_idx] = res_box_test$p
all_models_metrics$lag[row_idx] = res_box_test$lag

```

4. Neural Network Time Series analysis Model

```

nnet_model <- nnetar(stock_ts_freq7)

summary(nnet_model)


nnet_forecast<-as.vector(forecast(nnet_model,h = h))

nnet_forecast

display_residual(nnet_model)

# Run Box-Test and Identify lag for small p-value

res_box_test = run_box_test(nnet_model)

sprintf("Box-Test Result: small p-value:%f found at lag=%d",res_box_test$p,res_box_test$lag)

acc = accuracy(forecast_hw, stock_test_data$adjclose)

res_forecast$NNet = nnet_forecast$mean


row_idx = row_idx + 1

```

```

#update_res_metrics(all_models_metrics, 0, acc, res_box_test, row_idx)

all_models_metrics$TrainMAPE[row_idx] = acc[1,5]

all_models_metrics$TestMAPE[row_idx] = acc[2,5]

all_models_metrics$AIC[row_idx] = 0

all_models_metrics$Pvalue[row_idx] = res_box_test$p

all_models_metrics$lag[row_idx] = res_box_test$lag

```

```

##### SAVE ALL PLOT IN <<script_name.pdf on output directory
#####

```

```

draw_all_plots(stock_data, stock_ts, ts_decom, stock_ts_diff1,
               arima_model,forecast_arima,
               auto_arima_model,forecast_auto_arima,
               hw_mdl, forecast_hw,
               nnet_model,nnet_forecast,
               script_name, h,script_out_path)

```

```

##### PERSIST MODEL and FORECAST METRICS
#####

```

```

print(paste("Model Metrics", script_name))

print(all_models_metrics)

models_name = paste0(script_name,"_models_metrics",".Rda")

script_out_model_metrics_path = paste(out_path,models_name,sep="/")

save(all_models_metrics, file=script_out_model_metrics_path)


forecast_metrics = paste0(script_name,"_forecast_metrics",".Rda")

script_out_forecast_metrics_path = paste(out_path,sep="/",forecast_metrics)

print(paste("Forecast Metrics: ",script_name))

print(res_forecast)

save(res_forecast, file=script_out_forecast_metrics_path)

```

```

sprintf("#####  
#####")
}

print("ALL 20 STOCKS RETURNS METRICS")

script_out_all_script_return_metrics_path = paste(out_path,sep="/","all_script_return.Rda")

save(script_return, file = script_out_all_script_return_metrics_path)

print(script_return)

#####

```

Preprocess.R

```

# Include library

library(zoo) # imputation - last observed carry forward.

library(dplyr)

# Loading CSV File

load_csv = function(script_path) {

  # 1. Read the data into R

  raw_stock_data = read.csv(script_path, header = T)

  # 2. Look at the summary of all the variables and convert the following variables as factors

  summary(raw_stock_data)

  #str(raw_stock_data)

  raw_stock_data

}

# Extract data and adj close price

extract_stock_adjclose_sorted = function(raw_stock_data) {

  stock_dates = raw_stock_data$Date

```

```

stock_prices = raw_stock_data$Adj.Close
stocks_data = data.frame(as.Date(stock_dates),stock_prices)
colnames(stocks_data) <- c("date","adjclose")
tmp_stocks_data <- stocks_data[order(stocks_data$date),]
tmp_stocks_data
}

```

Imputation Function

1. Generate continous stock price including weekends

2. Impute Missing stock price value for week ends with Friday adjclose price

```

impute_stock_data = function(stocks_data, from = as.Date("2012/01/02"),to=as.Date("2016/12/31"))
{

```

Generate Continues data from 2012-2016, including market holiday for smooth TS processing

```

continous_dates= (seq(from=from, to=to, by="day"))

```

```

continous_adjclose = seq(from=0,to=0, by = 1)

```

```

continous_stocks_data = data.frame(continous_dates,continous_adjclose)

```

```

colnames(continous_stocks_data) = c("date","adjclose")

```

```

#str(continous_stocks_data)

```

```

merged_stock_data = merge(continous_stocks_data,stocks_data, by="date",all.x = TRUE )

```

```

stock_data = data.frame(merged_stock_data$date, merged_stock_data$adjclose.y )

```

Impute stock price NA on weekends with last observed values such as Friday

```

imputed_stocks_data = na.locf(stock_data)

```

```

colnames(imputed_stocks_data) = c("date","adjclose")

```

```

stock_data = data.frame(as.Date(imputed_stocks_data$date),

```

```

                        as.numeric(imputed_stocks_data$adjclose))

```

```

colnames(stock_data) = c("date","adjclose")

```



```

#View(stock_data)

stock_data
}

proc_stock_data = function (path ) {
  csv_file =load_csv(path)
  stock_data = extract_stock_adjclose_sorted(csv_file)
  imputed_stock_data = impute_stock_data(stock_data)
  imputed_stock_data
}

proc_stock_test_data = function (path,from =as.Date("2017/01/02"),to=as.Date("2017/01/31") ) {
  csv_file =load_csv(path)
  stocks_data = extract_stock_adjclose_sorted(csv_file)
  imputed_stock_data = impute_stock_data(stocks_data,from =from,to=to)
  imputed_stock_data
}

#####

Timeseries.R

# loading libraries

library(ggplot2)

library(ggfortify) # Extended ggplot2

# print all the plots in pdf file on script name

draw_all_plots = function(stock_data, stock_ts, ts_decom,
  stock_ts_diff1, arima_model, forecast_arima,
  auto_arima_model,forecast_auto_arima,
  hw_mdI, forecast_hw,
  nnet_model,nnet_forecast,
  script_name,h,script_out_path) {

```

```
title = paste("Last 5 Year ",script_name,"Stock Price history")
plot_ts = autoplot(object=ts(stock_data$adjclose, frequency = 365,
                             start=c(2012,1)),lab="Year", ylab="Price", main=title)
```

```
title = paste(script_name," Stock Trend")
decom_trend = autoplot(object=ts_decom$trend ,lab="Year", ylab="Price", main=title)
```

```
title = paste(script_name," Stock Seasonal")
decom_sea = autoplot(object=ts_decom$seasonal ,lab="Year", ylab="Price", main=title)
```

```
title = paste(script_name," Stock Random Noise")
decom_rnd = autoplot(ts_decom$random,xlab="Year", ylab="Price",title)
```

```
# To apply acf and pacf
title = paste(script_name," Stock Auto Correlation ")
plot_acf=autoplot.acf(acf(stock_ts, lag.max=12), main=title)
title = paste(script_name," Stock Partial Auto Correlation ")
plot_pacf=autoplot.acf( pacf(stock_ts, lag.max=12), main=title)
```

```
title = paste("After differencing, ", script_name," Stock Price")
plot.ts(stock_ts_diff1)
acf(stock_ts_diff1, lag.max=12)
pacf(stock_ts_diff1, lag.max=12)
```

```
title = paste(script_name," Stock Price Forecasts using Arima ", h ," days")
plot_forecast=autoplot.forecast(forecast_arima, xlab="Year", ylab="Price", main=title)
title = paste(script_name," Stock Price Forecasts using Auto Arima ", h ," days")
plot_auto_arima_forecast=autoplot.forecast(forecast_auto_arima, xlab="Year", ylab="Price",
main=title)
```

```

title = paste(script_name," Stock Price Forecasts using Holt-Winter ", h ," days")
plot_hw_forecast=autoplot.forecast(forecast_hw, xlab="Year", ylab="Price", main=title)

title = paste(script_name," Stock Price Forecasts using Neural Network ", h ," days")
plot_nnet_forecast=autoplot.forecast(nnet_forecast, xlab="Year", ylab="Price", main=title)


pdf(script_out_path)

print(plot_ts)
print(decom_trend)
print(decom_sea)
print(decom_rnd)
print(plot_acf)
print(plot_pacf)


plot.ts(stock_ts_diff1)
acf(stock_ts_diff1, lag.max=12)
pacf(stock_ts_diff1, lag.max=12)


tsdisplay(residuals(arma_model))
print(plot_forecast)


print(plot_auto_arma_forecast)
tsdisplay(residuals(auto_arma_model))
print(plot_hw_forecast)
tsdisplay(residuals(hw_md))
print(plot_nnet_forecast)
tsdisplay(residuals(nnet_model))

dev.off()
}

```

```

# converting numerical values into time series format.

stock_data_ts = function(stock_data, freq=365, script_name) {

  stock_ts = ts(stock_data$adjclose, frequency = freq, start=c(2012,1))

  #Plotting

  #plot.ts(stock_ts)

  title = paste("Last 5 Year ",script_name,"Stock Price")

  #auto_plot(stock_ts,xlab="Year", ylab="Price",title)

  #autoplot(object=ts(stock_data$adjclose, frequency = 365,
  #                    start=c(2012,1)),lab="Year", ylab="Price", main=title)

  stock_ts
}

```

```

# auto_plot = function(ts,xlab=xlab, ylab=ylab,title) {

#   autoplot( object = ts,
#   # object=ts(ts, frequency = 365, start=c(2012,1)),
#   #   xlab=xlab, ylab=ylab , main=title)
#   print(title)
# }

```

```

# Decompose Time Series and plot trend, seasonal and random noise components

stock_ts_decom = function(stock_ts,ts_plot=FALSE, script_name) {

  ts_decom <- decompose(stock_ts)

  # if (ts_plot) {

  #   print("Plotting...")

  #

  #

  #   par(mfrow=c(4,1))

  #   #plot.ts(stock_ts)

  #   title = paste("Last 5 Year ",script_name,"Stock Price")

```

```

# autoplot(object=stock_ts ,lab="Year", ylab="Price", main=title)
#
#
#
# #plot.ts(stocks_ts_decom$seasonal)
# #plot.ts(stocks_ts_decom$random)
# }
ts_decom
}

```

```

plot_acf_pacf = function (stock_ts, lag=12, script_name) {
  # To apply acf and pacf
  par(mfrow=c(1,3))
  plot.ts(stock_ts)
  acf(stock_ts, lag.max=lag)
  pacf(stock_ts, lag.max=lag)
}

```

```

stock_ts_diff = function(stock_ts, diff=1,plot=FALSE, script_name) {
  ts_diff <- diff(stock_ts, differences=diff)
  # normal diff,acf,pacf
  if (plot) {
    par(mfrow=c(3,1))
    plot.ts(stock_ts)
    plot.ts(ts_diff)
  }
  ts_diff
}

```

```

run_manual_arima = function (stock_ts, order=c(0,0,0), seasonal=c(0,0,0)) {

```

```

    arima_model <- Arima(stock_ts,order=order, seasonal=seasonal)
    arima_model
  }

```

```

run_auto_arima = function(stock_ts) {
  auto_arima_model <- auto.arima(stock_ts,ic='aic')
  auto_arima_model
}

```

```

model_forecast = function(object,h = 365 , plot=TRUE) {
  ts_forecast = forecast(object = object, h=h)
  if (plot)
    plot(ts_forecast)
  ts_forecast
}

```

```

display_residual = function(model) {
  tsdisplay(residuals(model))
}

```

```

box_test = function(model, lag=16, fitdf=4, type="Ljung") {
  box_test = Box.test(residuals(model), lag=lag, fitdf=fitdf, type=type)
  box_test
}

```

```

run_box_test = function(model) {
  p_val = 100
  lag = 1
  pvalues = array()
  for(j in 1:min(20)) {

```

```

pvalues[j] <- Box.test(residuals(model), type="Ljung-Box", lag=j)$p.value
val = as.numeric(pvalues[j])
if ( (val < p_val) && (val > 0)) {
  p_val = val
  lag = j
}
}
return (list("p"= p_val,"lag"=lag))
}

```

```

eval_results = function(arima_model,forecast_arima,stock_test_data,row_idx=1) {
  # plot residuals - TODO
  display_residual(arima_model)

  # Run Box-Test and Identify lag for small p-value
  res_box_test = run_box_test(arima_model)
  sprintf("Box-Test Result: small p-value:%f found at lag=%d",res_box_test$p,res_box_test$lag)

  # Run box-Ljung test
  #run_box_test(arima_model, lag = lag)
  acc = accuracy(forecast_arima, stock_test_data$adjclose)
  print(acc)
  aic = arima_model$aic
  res=arima_model$residuals
  p_value = res_box_test$p
  sprintf("AIC=%f ",aic)
  print(aic)
  #row_idx = row_idx+1

```

```

result$TrainMAPE[row_idx] = acc[1,5]
result$TestMAPE[row_idx] = acc[2,5]
result$AIC[row_idx] = aic
result$Pvalue[row_idx] = res_box_test$p
result$lag[row_idx] = res_box_test$lag
result
}

#####

```

Utils.R

Utility function

```

comp_stock_return = function(script, stock_data, start_date, end_date, invest_period) {
  buy_price = stock_data[stock_data$date==as.Date(start_date),]$adjclose
  curr_price = stock_data[stock_data$date==as.Date(end_date),]$adjclose
  CARG = ((curr_price/buy_price)^(1/invest_period)-1)*100
  script_name = sub(pattern = ".csv", replacement = "", x=script)
  return (list("name"=
script_name,"buy"=buy_price,"curr"=curr_price,"CARG"=CARG,"period"=invest_period))
}

```

```

update_script_return = function (ret , script_return, idx) {
  script_return$script[idx] = ret$name
  script_return$buy_price[idx] = ret$buy
  script_return$curr_price[idx] = ret$curr
  script_return$invest_period[idx] = ret$period
  script_return$CARG[idx] = ret$CARG
  script_return
}

```



```

update_res_metrics = function(all_models_metrics, aic, acc, res_box_test, row_idx) {
  all_models_metrics$TrainMAPE[row_idx] = acc[1,5]
  all_models_metrics$TestMAPE[row_idx] = acc[2,5]
  all_models_metrics$AIC[row_idx] = aic
  all_models_metrics$Pvalue[row_idx] = res_box_test$p
  all_models_metrics$lag[row_idx] = res_box_test$lag
  all_models_metrics
}

#####

OptimizationMonteCarlo.R

#

# MONTE CHARLO SIMULATION - PORTFOLIO OPTIMIZATION

#

rm(list=ls(all=TRUE))

# Include source from R files
source("R/preprocess.R")
source("R/Utils.R")

print(paste("Current Working directory:",getwd()))

data_path = paste(getwd(),"data",sep="/")

all_script_list = c("BAJFINANCE.csv", "CADILAH.C.csv", "HDFC.csv", "IOC.csv", "LT.csv",
  "RELIANCE.csv", "TATAMOTORS.csv", "TITAN.csv", "BHARATFORG.csv",
  "HAVELLS.csv", "HINDZINC.csv", "ITC.csv", "MOTHERSUMI.csv",
  "SHREECEM.csv", "TORNTPHARM.csv", "BRITANNIA.csv", "HDFCBANK.csv", "INFY.csv",
  "PIDILITIND.csv", "TCS.csv")

start_date = "2012-01-02"

# Historical Stock price End date

```

```

end_date = "2016-12-31"

# Investment period in years, default = 5 years starting from 2012 to 2016.

invest_period = 5

comp_all_stocks_return = function(all_script_list) {
  stocks_ret = list()
  for(script in all_script_list) {
    script_path = paste(data_path,script,sep="/")
    script_name = sub(".csv","",script)
    stock_data = proc_stock_data(script_path)
    ret = comp_stock_return(script, stock_data, start_date, end_date, invest_period)
    stocks_ret[script_name]<-ret$CARG
  }
  return(stocks_ret)
}

stocks_return = comp_all_stocks_return(all_script_list)

# Pick Top best performin stocks by CARG
port_stocks_data <- unlist(stocks_return[order(unlist(stocks_return), decreasing=TRUE)][1:5])
stocks_data <- data.frame(stocks = names(port_stocks_data),returns = round(port_stocks_data,2))


amount = 150000

stopping_criteria = 10000

weight_limit = 1

1/5


# Min inviestment is 10% on each stocks in portfolio

min_investment = 0.10

# Max inviestment is not more than 30% on each stock in portfolio

max_investment = 0.30

```

```
# Max return initialized with 15% return and later this gets computed by algorithm
```

```
max_portfolio_return = (amount*0.15)
```

```
# Equal allocation for all stocks for naive model
```

```
allocation = c(0.20, 0.20, 0.20, 0.20, 0.20)
```

```
# optimized allocation computed using algorithm and initialized with naive model
```

```
optimized_allocation = allocation
```

```
naive_portfolio_ret = function() {
```

```
  # Calculate portfolio actual return amount for equal contribution on each stock.
```

```
  portfolio_return = (stocks_data$returns %*%  
    (allocation * amount))/100
```

```
  # Calculate portfolio return in % for equal contribution on each stock.
```

```
  portfolio_return_in_percentage = (portfolio_return/amount)*100
```

```
  print(paste("Naive Portfolio Return computation"))
```

```
  print("Top stocks return data:")
```

```
  print(stocks_data)
```

```
  print(paste("Stock Allocation:"))
```

```
  print(allocation)
```

```
  print(paste("portfolio return:", portfolio_return))
```

```
  print(paste("portfolio return in percentage:", portfolio_return_in_percentage))
```

```
}
```

```
optimized_portfolio_allocation_fn<-function(simulation){
```

```
  set.seed(1234)
```

```

for(i in 1:simulation){
  #initialize stock allocation
  stockalloc = numeric()
  nums = runif(5, min = 0.1, max = 0.28)
  tot = round(sum(nums),2)
  if(tot>1) {
    d = (tot-1)/5
    stockalloc = nums -d
  } else if (tot < 1){
    d = (1-tot)/5
    stockalloc = nums + d
  }
  else {
    stockalloc=nums
  }
  portfolio_return = (stocks_data$returns %*%
    (stockalloc * amount))/100
  #returns<-stocks_data$returns %*% (stockalloc * stockInves)
  if(portfolio_return > max_portfolio_return ){
    max_portfolio_return <- portfolio_return[,1]
    optimized_allocation <-stockalloc
  }
}
print(paste("#####"))
cat("optimized Allocation ", optimized_allocation,
  "with returns = ", max_portfolio_return, "\n")

print(paste("Monte Carlo Portfolio Optimization"))

```

```

print(paste("Simulation Iteration:", simulation))
print("Top stocks return data:")
print(stocks_data)
print(paste("Optimized Stock Allocation:"))
print(optimized_allocation)
print("Invested Period:")
print(invest_period)
print(paste("portfolio return:", round(max_portfolio_return,2)))
portfolio_return_in_percentage = round(((max_portfolio_return/amount)*100),2)
print(paste("portfolio return in percentage:",portfolio_return_in_percentage ))
print(paste("#####"))
return(list(max_portfolio_return,optimized_allocation))
}

```

```

naive_portfolio_ret()

```

```

simulations = c(1000,10000,100000)

```

```

# instead for loop use sapply

```

```

start = Sys.time()

```

```

time = sapply(simulations, optimized_portfolio_allocation_fn)

```

```

end = Sys.time() - start

```

```

end

```

```

#####
#####

```

```

# OUTPUT

```

```

##### BENCH MARK MODEL RETURN #####

```

```

# > naive_portfolio_ret()

```

```

# [1] "Naive Portfolio Return computation"

```

```

# [1] "Top stocks return data:"

# stocks returns

# MOTHERSUMI MOTHERSUMI  55.98

# BAJFINANCE BAJFINANCE  54.34

# SHREECEM  SHREECEM  47.91

# BRITANNIA  BRITANNIA  47.14

# TORNTPHARM TORNTPHARM  40.49

# [1] "Stock Allocation:"

# [1] 0.2 0.2 0.2 0.2 0.2

# [1] "portfolio return: 73758"

# [1] "portfolio return in percentage: 49.172"

#

# ##### MONTE CARLO SIMULATION #####

# [1] "#####"

# optimized Allocation 0.2806153 0.2733009 0.1350734 0.1666344 0.1489449 with returns = 76375.95

# [1] "Monte Carlo Portfolio Optimization"

# [1] "Simulation Iteration: 1000"

# [1] "Top stocks return data:"

# stocks returns

# MOTHERSUMI MOTHERSUMI  55.98

# BAJFINANCE BAJFINANCE  54.34

# SHREECEM  SHREECEM  47.91

# BRITANNIA  BRITANNIA  47.14

# TORNTPHARM TORNTPHARM  40.49

# [1] "Optimized Stock Allocation:"

# [1] 0.2806153 0.2733009 0.1350734 0.1666344 0.1489449

# [1] "Invested Period:"

# [1] 5

# [1] "portfolio return: 76375.95"

```

```
# [1] "portfolio return in percentage: 50.92"
# [1] "#####"
# [1] "#####"
# optimized Allocation 0.2773343 0.2740724 0.2081766 0.139142 0.1041057 with returns = 76749.6
# [1] "Monte Carlo Portfolio Optimization"
# [1] "Simulation Iteration: 10000"
# [1] "Top stocks return data:"
# stocks returns
# MOTHERSUMI MOTHERSUMI 55.98
# BAJFINANCE BAJFINANCE 54.34
# SHREECEM SHREECEM 47.91
# BRITANNIA BRITANNIA 47.14
# TORNTPHARM TORNTPHARM 40.49
# [1] "Optimized Stock Allocation:"
# [1] 0.2773343 0.2740724 0.2081766 0.1391420 0.1041057
# [1] "Invested Period:"
# [1] 5
# [1] "portfolio return: 76749.6"
# [1] "portfolio return in percentage: 51.17"
# [1] "#####"
# [1] "#####"
# optimized Allocation 0.2860507 0.2976422 0.1722426 0.1222493 0.1255165 with returns = 76926.2
# [1] "Monte Carlo Portfolio Optimization"
# [1] "Simulation Iteration: 1e+05"
# [1] "Top stocks return data:"
# stocks returns
# MOTHERSUMI MOTHERSUMI 55.98
# BAJFINANCE BAJFINANCE 54.34
# SHREECEM SHREECEM 47.91
```

```

# BRITANNIA BRITANNIA 47.14

# TORNTPHARM TORNTPHARM 40.49

# [1] "Optimized Stock Allocation:"

# [1] 0.2860507 0.2976422 0.1722426 0.1222493 0.1255165

# [1] "Invested Period:"

# [1] 5

# [1] "portfolio return: 76926.2"

# [1] "portfolio return in percentage: 51.28"

#####

OptimizationSimulatedAnnealing.R

rm(list=ls(all=TRUE))

# Include source from R files

source("R/preprocess.R")

source("R/Utils.R")

#source("R/const.R")

print(paste("Current Working directory:",getwd()))

# Location of stock data folder

data_path = paste(getwd(),"data",sep="/")

all_script_list = c("BAJFINANCE.csv", "CADILAH.C.csv", "HDFC.csv", "IOC.csv", "LT.csv",

                    "RELIANCE.csv", "TATAMOTORS.csv", "TITAN.csv", "BHARATFORG.csv",

                    "HAVELLS.csv", "HINDZINC.csv", "ITC.csv", "MOTHERSUMI.csv",

                    "SHREECEM.csv", "TORNTPHARM.csv", "BRITANNIA.csv", "HDFCBANK.csv", "INFY.csv",

                    "PIDILITIND.csv", "TCS.csv")

start_date = "2012-01-02"

```



```
# Historical Stock price End date
```

```
end_date = "2016-12-31"
```

```
# Investment period in years, default = 5 years starting from 2012 to 2016.
```

```
invest_period = 5
```

```
comp_all_stocks_return = function(all_script_list) {
```

```
  stocks_ret = list()
```

```
  for(script in all_script_list) {
```

```
    script_path = paste(data_path,script,sep="/")
```

```
    script_name = sub(".csv","",script)
```

```
    stock_data = proc_stock_data(script_path)
```

```
    ret = comp_stock_return(script, stock_data, start_date, end_date, invest_period)
```

```
    stocks_ret[script_name]<-ret$CARG
```

```
  }
```

```
  return(stocks_ret)
```

```
}
```

```
stocks_return = comp_all_stocks_return(all_script_list)
```

```
# Pick Top best performin stocks by CARG
```

```
port_stocks_data <- unlist(stocks_return[order(unlist(stocks_return), decreasing=TRUE)][1:5])
```

```
stocks_data <- data.frame(stocks = names(port_stocks_data),returns = round(port_stocks_data,2))
```

```
min_investment = 0.10
```

```
amount = 150000
```

```
stopping_criteria = 8000
```

```
weight_limit = 1
```

```

# Min inviestment is 10% on each stocks in portfolio

min_investment = 0.10

# Max inviestment is not more than 30% on each stock in portfolio

max_investment = 0.30


# Max return initialized with 15% return and later this gets computed by algorithm
max_portfolio_return = (amount*0.15)


init_soln_fn <- function(){
  # Let us initiate solution with almost equal weights to make it total 1
  initial_soln = c(0.20, 0.20, 0.20, 0.20, 0.20)
  return(initial_soln)
}


# Purterbation function: randomly select a point and do operation
purterb_fn = function(solution){
  idx = sample(1:length(solution), 2)
  purt_solution = solution
  x = solution[idx[1]]
  y = solution[idx[2]]

  # Taking the diff of 1st stock weight and 0.10 and take half of it
  # Subtract that value and the same value to the secnd stock weight
  # (As total should be 1)

```

```

diff = ((solution[idx[1]] - min_investment)/2)
put_solution[idx[1]] = solution[idx[1]] - diff
put_solution[idx[2]] = solution[idx[2]] + diff

return(put_solution)
}

# Evaluation function.
evaluate_fn = function(stocks_data, weight_limit, solution, amount) {
  #single_script_invest = (solution * rep(amount, length(solution)))

  #total_return = stocks_data$returns %*% single_script_invest

  # Calculate portfolio actual return amount for equal contribution on each stock.
  portfolio_return = (stocks_data$returns %*%
    (solution * amount))/100
  # Calculate portfolio return in % for equal contribution on each stock.
  #portfolio_return_in_percentage = (portfolio_return/amount)*100

  if (round(sum(solution), 1) != weight_limit)
    return(0)
  else
    return(portfolio_return)
}

run_simulated_annealing_algo = function(max_iterations, amount){
  cat("Max iterations =", max_iterations, "\n")
  # Generate a random solution
  initial_soln = init_soln_fn()
  initial_investment_amount = evaluate_fn(stocks_data, weight_limit, initial_soln, amount)

```

```
print(paste("Initial Solution:"))
```

```
print(initial_soln)
```

```
base_soln = initial_soln
```

```
base_val = initial_investment_amount
```

```
counter = 0
```

```
# solution vs available
```

```
cat(paste("Initial investment amount is : ", base_val, "\n"))
```

```
global_val = base_val
```

```
global_solu = base_soln
```

```
for (i in 1:max_iterations) {
```

```
  # Purterbation
```

```
  next_soln = purterb_fn(base_soln)
```

```
  next_val = evaluate_fn(stocks_data, weight_limit, next_soln, amount)
```

```
  if(any(next_soln > min_investment) == FALSE){
```

```
    return(0)
```

```
  }else{
```

```
    counter = counter + 1
```

```
    if(next_val > base_val){
```

```
      base_solu = next_soln
```

```
      base_val = next_val
```

```
    }
```

```
  }else{
```

```
    # Accept with acceptance probability
```

```
    acceptance_prob = runif(1, 0, 1)
```

```
    if(acceptance_prob > 0.5){
```

```
      base_soln = next_soln
```

```

        base_val = next_val
    }
}
}
if(global_val <= base_val){
    global_val = base_val
    global_solu = base_solu
}
i = counter
# solution
cat("Returns in ", i, "iteration is : ", base_val,"\n")
}
cat("\n","Returns in ", i, "iteration is : ", global_val,"global_val:",global_solu,"\n")

print(paste("Simulated Anealing - Portfolio Optimization"))
print(paste("Simulation Iteration:", max_iterations))
print("Top stocks return data:")
print(stocks_data)
print(paste("Optimized Stock Allocation:"))
print(round(global_solu,2))
print("Invested Period:")
print(invest_period)
print(paste("portfolio return:", round(global_val,2)))
portfolio_return_in_percentage = round(((global_val/amount)*100),2)
print(paste("portfolio return in percentage:",portfolio_return_in_percentage ))

return(list(global_solu, global_val))
}

```

```

execute_main_fn = function(stocks_data, max_iterations, amount){
  set.seed(1234)
  solution_list = run_simulated_annealing_algo(max_iterations, amount)
  final_solution = as.numeric(solution_list[[1]])
  final_solution_value = solution_list[[2]]
  stocks_data$finalSolution = final_solution

  cat("Total returns = ", final_solution_value, "\n")
  return(stocks_data)
}

```

```

result = execute_main_fn(stocks_data, max_iterations = 100000, amount = 150000)
print("SIMULATED ANNEALING OPTIMIZATION SOLUTIONS")
print(result)

```

```

#####
#####

```

```
# OUTPUT
```

```
# Returns in 10000 iteration is : 76443.23 global_val: 0.3295739 0.1944862 0.1980905 0.1018766
0.1759728
```

```
# [1] "Simulated Anealing - Portfolio Optimization"
```

```
# [1] "Simulation Iteration: 10000"
```

```
# [1] "Top stocks return data:"
```

```
# stocks returns
```

```
# MOTHERSUMI MOTHERSUMI 55.98
```

```
# BAJFINANCE BAJFINANCE 54.34
```

```
# SHREECEM SHREECEM 47.91
```

```
# BRITANNIA BRITANNIA 47.14
```

TORNTPHARM TORNTPHARM 40.49

[1] "Optimized Stock Allocation:"

[1] 0.33 0.19 0.20 0.10 0.18

[1] "Invested Period:"

[1] 5

[1] "portfolio return: 76443.23"

[1] "portfolio return in percentage: 50.96"

Total returns = 76443.23

#####

Returns in 1e+05 iteration is : 76946.61 global_val: 0.3328701 0.2335364 0.1016767 0.1219498
0.209967

[1] "Simulated Anealing - Portfolio Optimization"

[1] "Simulation Iteration: 1e+05"

[1] "Top stocks return data:"

stocks returns

MOTHERSUMI MOTHERSUMI 55.98

BAJFINANCE BAJFINANCE 54.34

SHREECEM SHREECEM 47.91

BRITANNIA BRITANNIA 47.14

TORNTPHARM TORNTPHARM 40.49

[1] "Optimized Stock Allocation:"

[1] 0.33 0.23 0.10 0.12 0.21

[1] "Invested Period:"

[1] 5

[1] "portfolio return: 76946.61"

[1] "portfolio return in percentage: 51.3"

Total returns = 76946.61

#

```
#  
# [1] "SIMULATED ANNEALING OPTIMIZATION SOLUTIONS"  
# > print(result)  
# stocks returns finalSolution  
# MOTHERSUMI MOTHERSUMI 55.98 0.3328701  
# BAJFINANCE BAJFINANCE 54.34 0.2335364  
# SHREECEM SHREECEM 47.91 0.1016767  
# BRITANNIA BRITANNIA 47.14 0.1219498  
# TORNTPHARM TORNTPHARM 40.49 0.2099670  
#  
#####
```