TIME SERIES FORECASTING OF TCS STOCK CLOSING PRICE

CIA ASSESSMENT

Abstract

Utilize historical stock price data of TCS from 2014-2024 to forecast closing price by incorporating external factors that might influence the closing price.

Report By: Atishya Ghosh (Roll 43)

TABLE OF CONTENTS

INTRODUCTION

Purpose

Time series forecasting is a crucial aspect of data-driven decision-making in various domains such as finance, economics, healthcare, and supply chain management. This project aims to develop a forecasting model using the ARIMAX (AutoRegressive Integrated Moving Average with Exogenous variables) technique and ARIMA (AutoRegressive Integrated Moving Average) technique, to predict future trends in TCS closing stock price based on historical data and some external influencing factors.

Tata Consultancy Services (TCS) is a digital transformation and technology partner of choice for industry-leading organizations worldwide. With a highly skilled workforce of over 607,000 consultants in 55 countries and 180 service delivery centres across the world, the company has been recognized as a top employer in six continents. TCS (Tata Consultancy Services) is one of the largest companies in **Nifty 50**, India's benchmark stock index. Since Nifty 50 represents the top 50 companies, **if Nifty rises or falls, TCS may follow the trend**, but not always. Independent factors such as Earnings reports, new contracts, regulatory changes, IT sector performance, foreign investors may affect TCS stock prices. For short-term analysis, global IT trends and company-specific factors may matter more but for long-term trends, TCS generally follows Nifty since it is a major index constituent. Hence **Nifty Closing Price** is considered to be an exogenous variable.

TCS is a major IT services company, and many of its competitors (e.g., Accenture, IBM, Cognizant) are listed on US markets. If US tech stocks (e.g., Nasdaq, S&P 500 IT sector) rise or fall significantly, it can influence TCS, as investors may adjust their positions in Indian IT stocks accordingly. The S&P 500 impacts global investor sentiment. If foreign institutional investors pull out money from US stocks, they might also exit Indian stocks, affecting TCS. Additionally, If US markets fall sharply, investors may reduce risk globally, affecting TCS even if nothing changes fundamentally in India. Thus **S&P 500 closing price** is also considered as an external factor influencing TCS closing price.

Volume measures how many shares were traded during a given time period. Closing price is the last traded price of a stock before the market closes. Volume often affects price trends – high volume supports strong price moves. Thus, **TCS traded Volume** is also considered as a variable affecting its stock price.

Key Objectives

1. Determine the effect of external variables Nifty Closing Price, S&P 500 Closing Price and TCS trading Volume on TCS Closing Price.
2. Identify the best forecasting model for Closing Price Prediction of TCS.
3. Identify the limitations of the model.

Below we discuss the two models ARIMA and ARIMAX and the parameters described in the models in addition to their use cases.

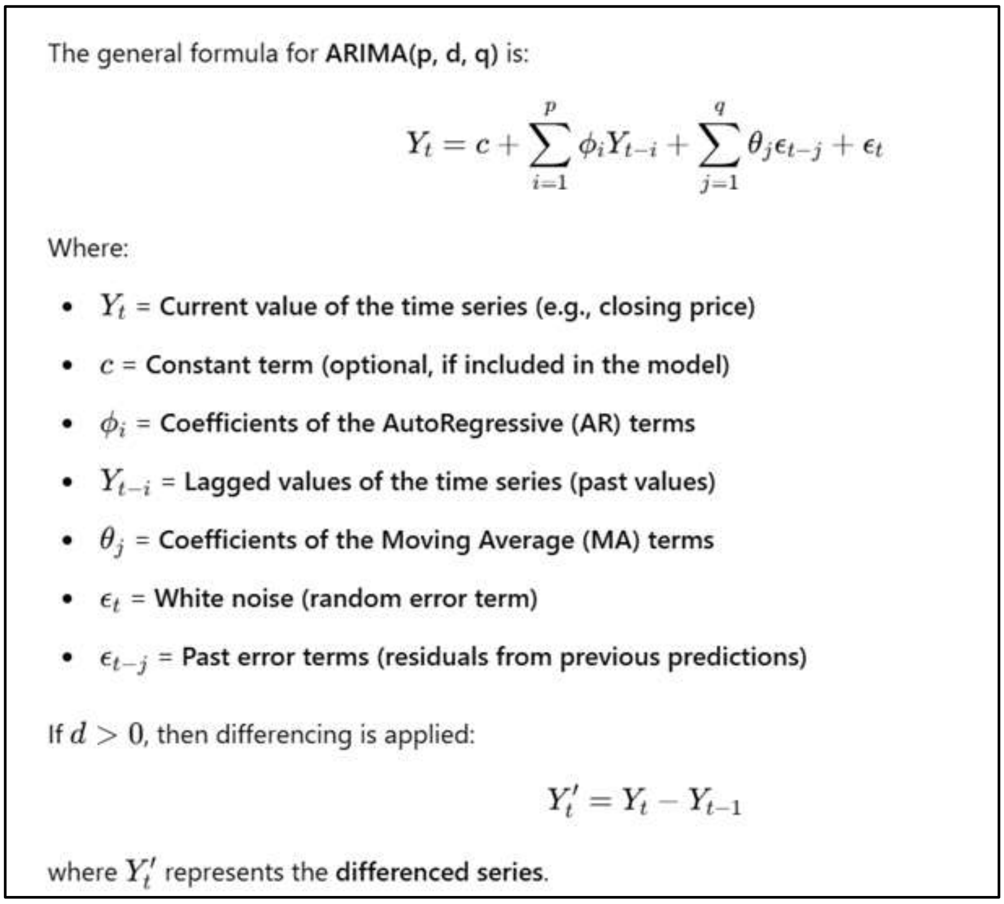
ARIMA Model

The ARIMA model is a widely used time series forecasting technique that models a variable based on its past values and past forecast errors. It is particularly useful for predicting stock prices, economic indicators, and other sequential data.

Parameters and Equation of ARIMA Model

The **ARIMA (p, d, q)** model consists of:

- **AutoRegressive (AR) term (p)** → Uses past values of the series to predict future values.
- **Integrated (I) term (d)** → Represents the number of **times the series is differenced** to make it stationary.
- **Moving Average (MA) term (q)** → Uses past forecast errors to improve predictions.



The general formula for **ARIMA(p, d, q)** is:

$$Y_t = c + \sum_{i=1}^{p} \phi_i Y_{t-i} + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} + \epsilon_t$$

Where:

- Y_t = Current value of the time series (e.g., closing price)
- c = Constant term (optional, if included in the model)
- ϕ_i = Coefficients of the AutoRegressive (AR) terms
- Y_{t-i} = Lagged values of the time series (past values)
- θ_j = Coefficients of the Moving Average (MA) terms
- ϵ_t = White noise (random error term)
- ϵ_{t-j} = Past error terms (residuals from previous predictions)

If $d > 0$, then differencing is applied:

$$Y_t' = Y_t - Y_{t-1}$$

where Y_t' represents the **differenced series**.

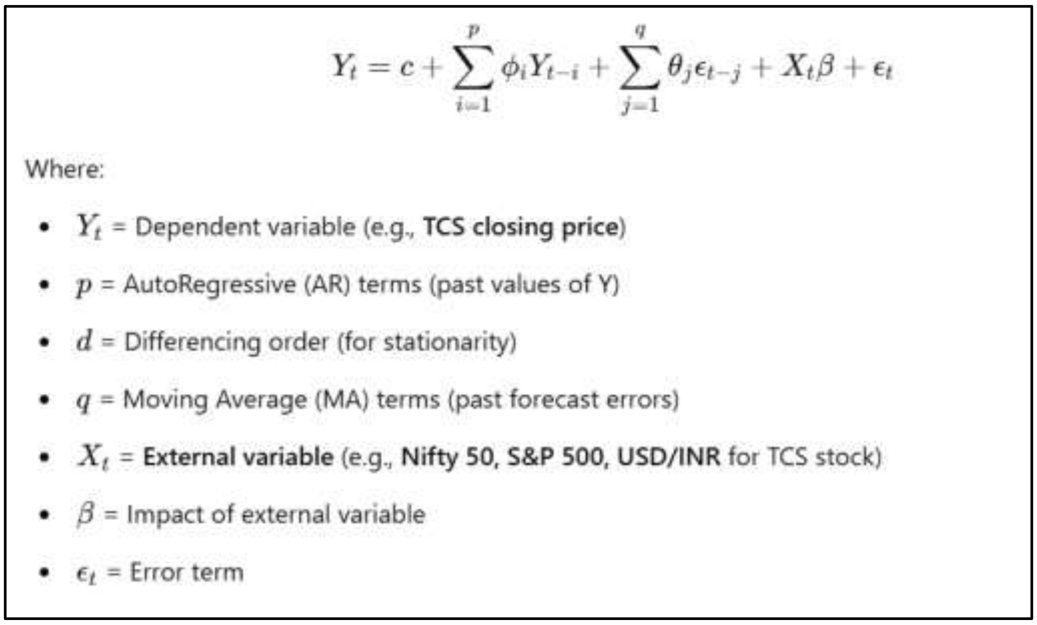
Use Case

- The data is time series.
- The data can be made stationary by differencing.
- There are no strong external influences (otherwise, use ARIMAX).

ARIMAX Model

The ARIMAX model is an extension of the ARIMA (AutoRegressive Integrated Moving Average) model, which includes exogenous (external) variables to improve forecasting accuracy.

Parameters and Equation of ARIMA Model



$$Y_t = c + \sum_{i=1}^{p} \phi_i Y_{t-i} + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} + X_t \beta + \epsilon_t$$

Where:

- Y_t = Dependent variable (e.g., **TCS closing price**)
- p = AutoRegressive (AR) terms (past values of Y)
- d = Differencing order (for stationarity)
- q = Moving Average (MA) terms (past forecast errors)
- X_t = **External variable** (e.g., **Nifty 50, S&P 500, USD/INR** for TCS stock)
- β = Impact of external variable
- ϵ_t = Error term

The ARIMAX model equation only contains two additional parameters External Variable X_t and its impact on dependent variable Beta β. To identify the p,d,q parameters, the following are needed.

p (AR term): Significant peaks in the Partial Autocorrelation Function (PACF) plot.

d (I term): Use ADF test to determine if differencing is needed.

q (MA term): Significant peaks in the Autocorrelation Function (ACF) plot.

Use Case

- Incorporates external market trends (e.g., Nifty 50, USD/INR).
- Can improve prediction accuracy compared to ARIMA alone.
- Useful for macroeconomic and financial data modelling.

Assumptions of both models

- The time series should have a constant mean and variance over time.
- The relationship between past values and future values should be linear.
- No autocorrelated errors and normally distributed residuals.
- The external variable (X) should have a real impact on the dependent variable in ARIMAX.

METHODOLOGY

1. Data Collection

Nifty 50 Closing Price, S&P 500 Closing Price, TCS Closing price, TCS traded Volume are collected using yahoo finance in RStudio.

2. Data Cleaning and Preprocessing

Missing Values are removed using R. Stationarity of variables Nifty 50 Closing Price, S&P 500 Closing Price, TCS Closing price & TCS traded Volume are checked using Augmented Dickey Fuller Test, Kwiatkowski-Phillips-Schmidt-Shin test and Phillips Perron test in R. If not stationary, then the variables are made stationary (constant mean and variance) by differencing or doing a log transform, if necessary, by using base functions in R. The variables are again checked for stationarity.

3. Exploratory Data Analysis

Visualizing time series trends using line charts in R to see trend, seasonality, stationarity, etc. Plotting autocorrelation and partial autocorrelation functions (ACF & PACF) in R to determine AR and MA terms using acf and pacf functions. Identifying correlations between the 3 mentioned exogenous variables and closing price using cor function.

4. Model Creation

Identify appropriate models using results from step 3 to determine AR, differencing and MA terms in addition to incorporating external factors. Fit the model on the training data. Formulate the model as a mathematical equation based on results by assessing the coefficients and their standard errors. Check results with auto-arima function as well to identify optimal model. Evaluate residuals for normality and independence.

5. Model Evaluation

Evaluating model performance using metrics such as: Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), AIC score, etc.

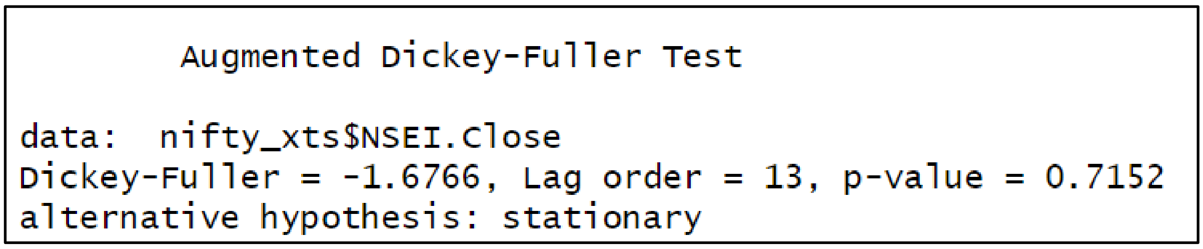
6. Interpretation

Interpret the usefulness of the models for the use case and present the best model including limitations of the model. Evaluate the effectiveness of each model and also analyse the practical implications of the forecasted results.

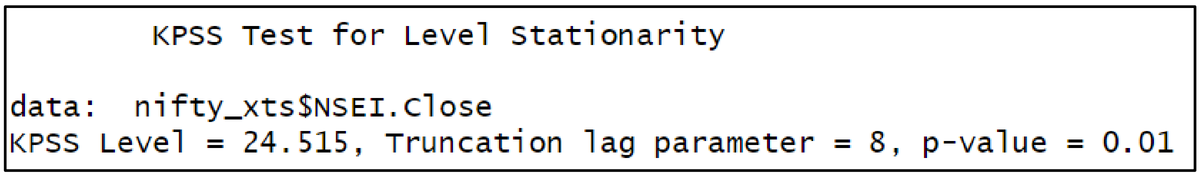
RESULTS

Check for **Stationarity** of Nifty 50 Closing Price, S&P 500 Closing Price, TCS Closing price & TCS traded Volume using Augmented Dickey Fuller test, Kwiatkowski-Phillips-Schmidt-Shin test and Phillips Perron test. Majority voting is used to determine stationarity.

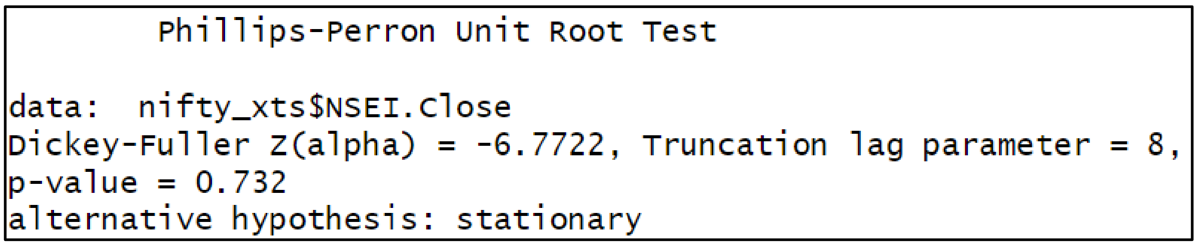
Nifty 50 Closing Price



```
        Augmented Dickey-Fuller Test

data:  nifty_xts$NSEI.Close
Dickey-Fuller = -1.6766, Lag order = 13, p-value = 0.7152
alternative hypothesis: stationary
```



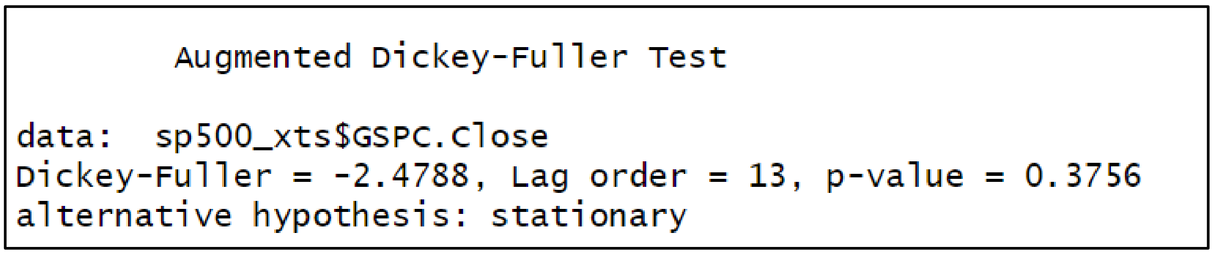
```
        KPSS Test for Level Stationarity

data:  nifty_xts$NSEI.Close
KPSS Level = 24.515, Truncation lag parameter = 8, p-value = 0.01
```



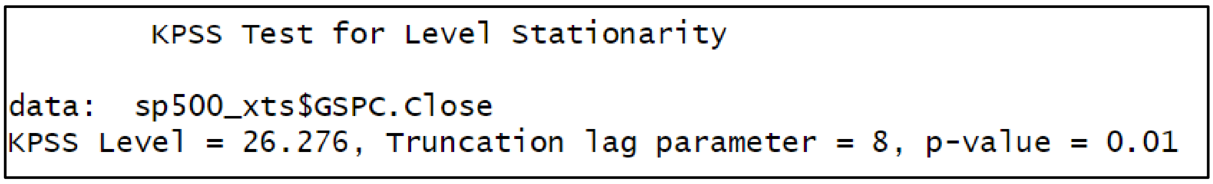
```
        Phillips-Perron Unit Root Test

data:  nifty_xts$NSEI.Close
Dickey-Fuller Z(alpha) = -6.7722, Truncation lag parameter = 8,
p-value = 0.732
alternative hypothesis: stationary
```

The Null Hypothesis for KPSS test is Data is Stationary while this is the alternate hypothesis for other two. P-value less than 0.05 allows us to reject null hypothesis. For KPSS test, we can reject null hypothesis suggesting Nifty Closing Price is not stationary. For other two tests, we accept the null hypothesis given p>0.05 which states data is not stationary.

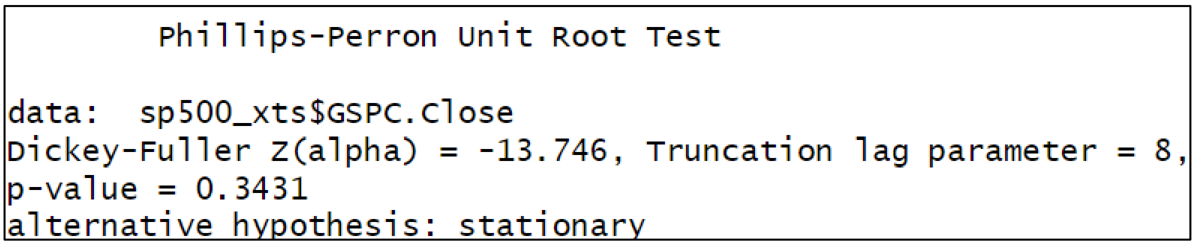
S&P 500 Closing Price



```
        Augmented Dickey-Fuller Test

data:  sp500_xts$GSPC.Close
Dickey-Fuller = -2.4788, Lag order = 13, p-value = 0.3756
alternative hypothesis: stationary
```



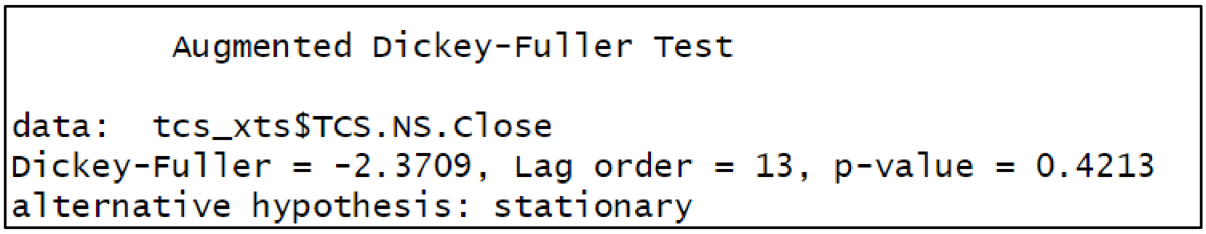
```
        KPSS Test for Level Stationarity

data:  sp500_xts$GSPC.Close
KPSS Level = 26.276, Truncation lag parameter = 8, p-value = 0.01
```



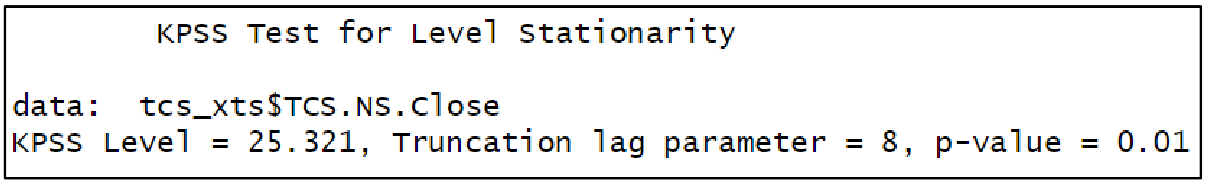
```
         Phillips-Perron Unit Root Test

data:  sp500_xts$GSPC.Close
Dickey-Fuller Z(alpha) = -13.746, Truncation lag parameter = 8,
p-value = 0.3431
alternative hypothesis: stationary
```

All three tests suggest S&P 500 Closing Price is not stationary as per p-value.

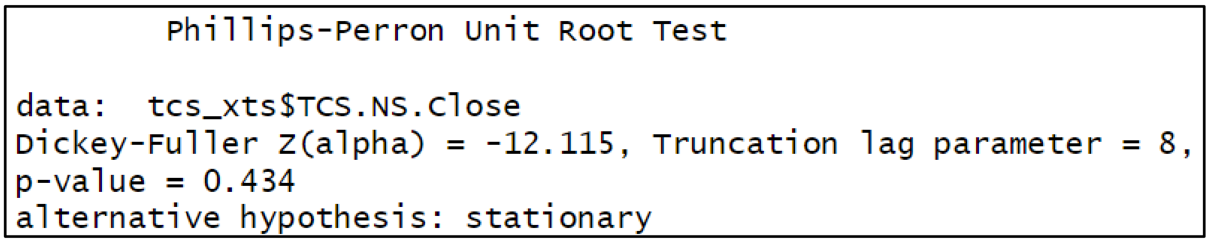
TCS Closing price



```
         Augmented Dickey-Fuller Test

data:  tcs_xts$TCS.NS.Close
Dickey-Fuller = -2.3709, Lag order = 13, p-value = 0.4213
alternative hypothesis: stationary
```



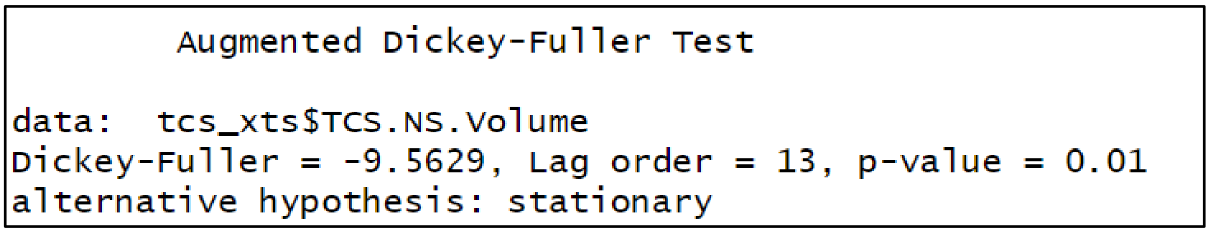
```
         KPSS Test for Level Stationarity

data:  tcs_xts$TCS.NS.Close
KPSS Level = 25.321, Truncation lag parameter = 8, p-value = 0.01
```



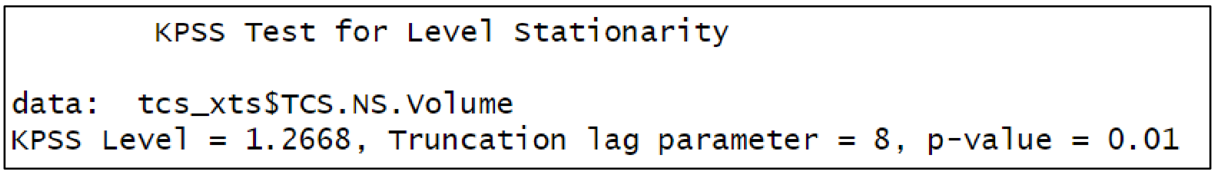
```
         Phillips-Perron Unit Root Test

data:  tcs_xts$TCS.NS.Close
Dickey-Fuller Z(alpha) = -12.115, Truncation lag parameter = 8,
p-value = 0.434
alternative hypothesis: stationary
```

All three tests suggest TCS Closing Price is not stationary as per p-value.

TCS traded Volume



```
         Augmented Dickey-Fuller Test

data:  tcs_xts$TCS.NS.Volume
Dickey-Fuller = -9.5629, Lag order = 13, p-value = 0.01
alternative hypothesis: stationary
```

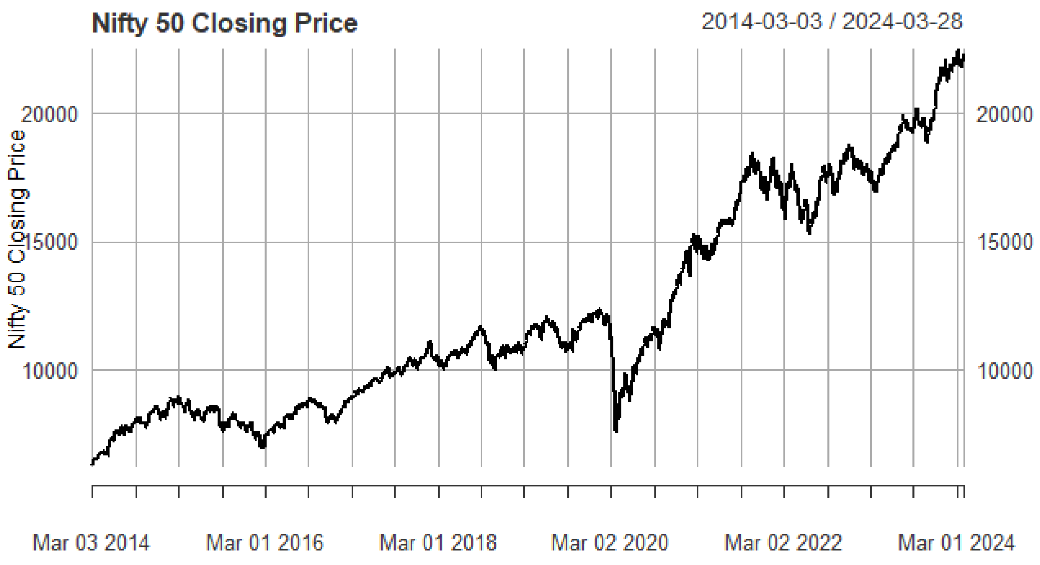


```
         KPSS Test for Level Stationarity

data:  tcs_xts$TCS.NS.Volume
KPSS Level = 1.2668, Truncation lag parameter = 8, p-value = 0.01
```

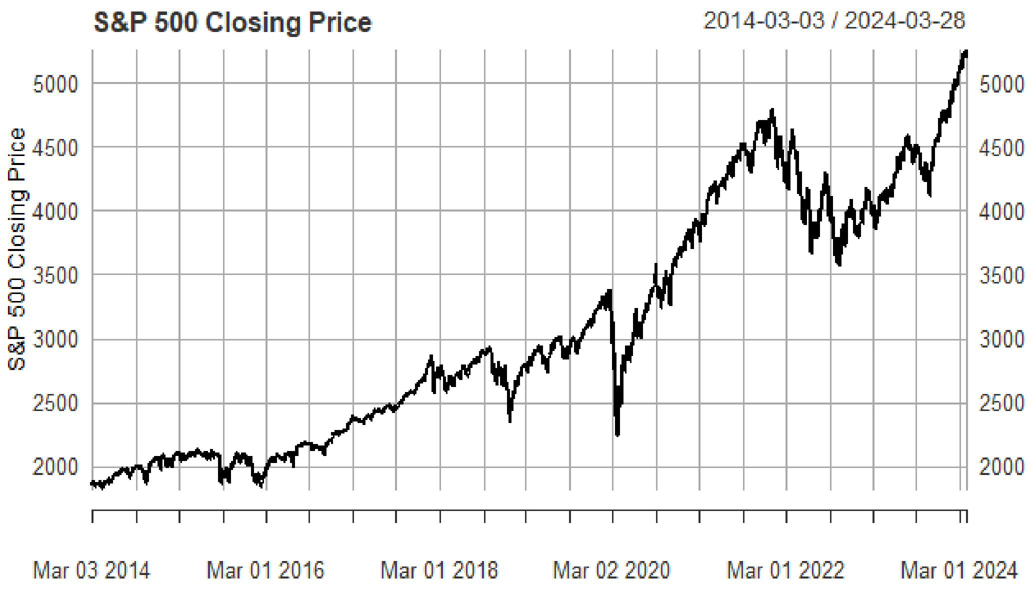
```
        Phillips-Perron Unit Root Test

data:  tcs_xts$TCS.NS.Volume
Dickey-Fuller Z(alpha) = -2436.9, Truncation lag parameter = 8,
p-value = 0.01
alternative hypothesis: stationary
```

Two out of three tests suggest TCS traded Volume is stationary as per p-value.

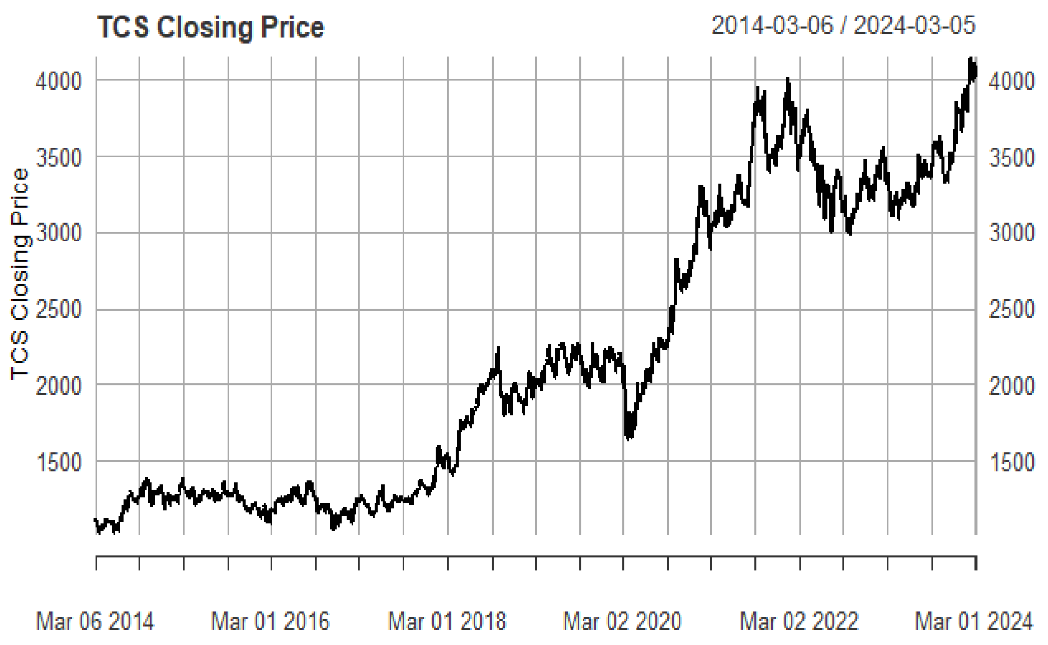
Visualizing the 4 variables on a line plot



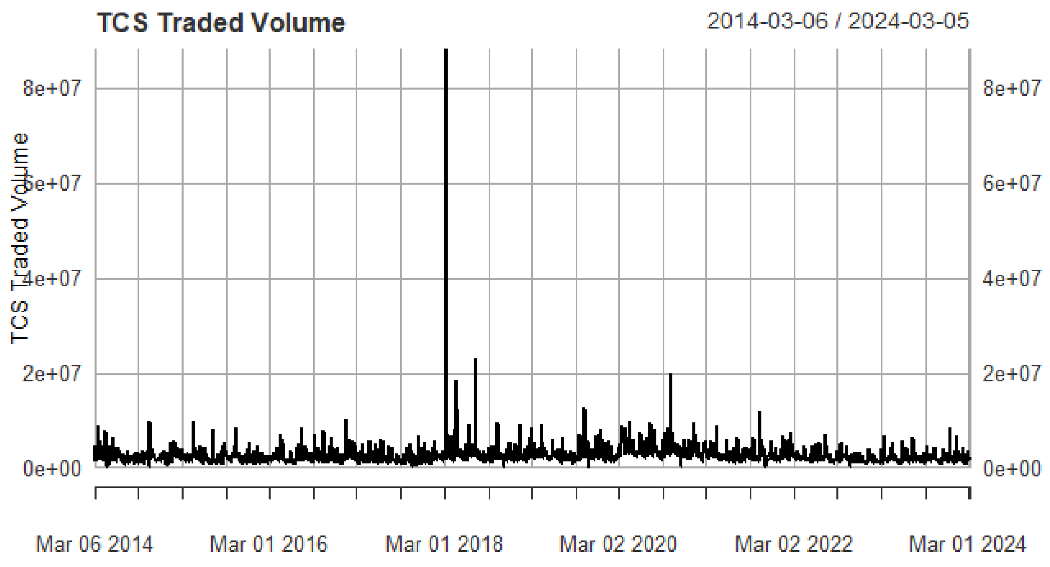
We see an increasing trend in Nifty price during the period 2014-2024 with a sharp decline in 2020 probably attributed to Covid-19. Clearly non-stationary.



We see an increasing trend in S&P 500 price during the period 2014-2024 with a sharp decline in 2020 probably attributed to Covid-19. Clearly non-stationary.



We see an increasing trend in TCS closing price during the period 2014-2024 with a sharp decline in 2020 probably attributed to Covid-19. The patterns are similar to Nifty 50 and S&P 500 closing prices in the timeframe suggesting a correlation between them. Clearly non-stationary.

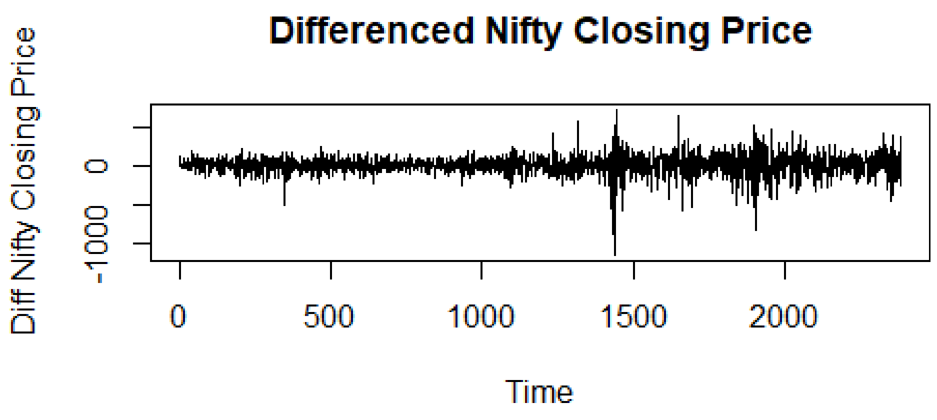


We observe TCS traded volume is stationary in the time period with the exception of a few outliers.

We have to difference the Nifty 50 Closing Price, S&P 500 Closing Price & TCS Closing price to make them stationery and check for the stationarity using the 3 tests ADF, KPSS and PP after differencing and visualize using line plots. TCS traded volume need not be transformed as it is already stationary.

Visualizing the differenced variables on a line plot and Checking for Stationarity

Differenced Nifty 50 Closing Price



```
        Augmented Dickey-Fuller Test

data:  nifty_ts
Dickey-Fuller = -12.916, Lag order = 13, p-value = 0.01
alternative hypothesis: stationary
```

```
        KPSS Test for Level Stationarity

data:  nifty_ts
KPSS Level = 0.11365, Truncation lag parameter = 8,
p-value = 0.1
```

```
        Phillips-Perron Unit Root Test

data:  nifty_ts
Dickey-Fuller Z(alpha) = -2390.6, Truncation lag
parameter = 8, p-value = 0.01
alternative hypothesis: stationary
```

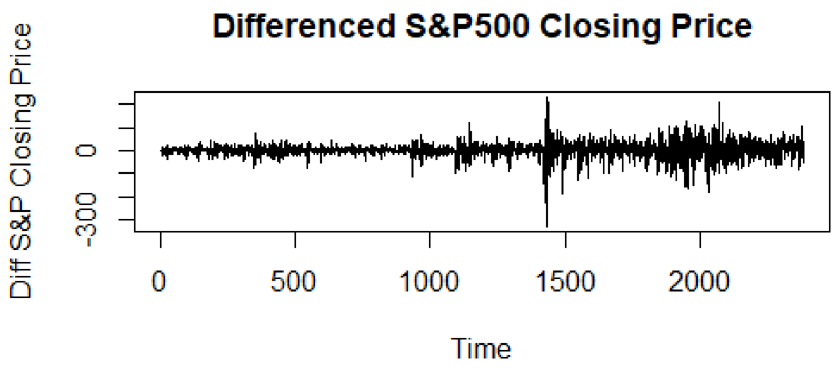
All three tests and line plot suggest that differencing is sufficient to make Nifty 50 closing price stationary.

Differenced S&P 500 Closing Price

```
        Augmented Dickey-Fuller Test

data:  sp500_ts
Dickey-Fuller = -13.58, Lag order = 13, p-value = 0.01
alternative hypothesis: stationary
```

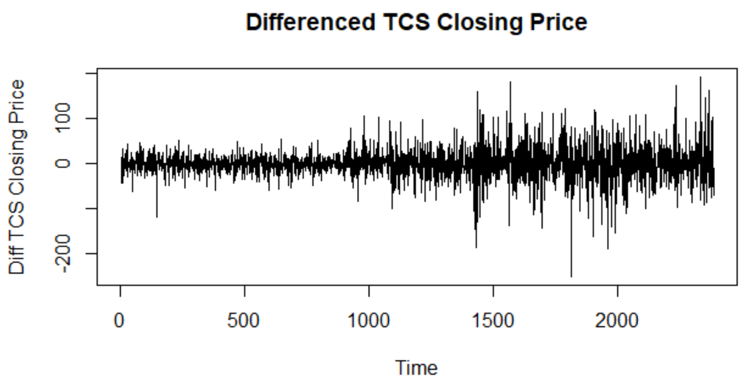
```
          KPSS Test for Level Stationarity

data:  sp500_ts
KPSS Level = 0.096471, Truncation lag parameter = 8,
p-value = 0.1
```

```
          Phillips-Perron Unit Root Test

data:  sp500_ts
Dickey-Fuller Z(alpha) = -2510.6, Truncation lag
parameter = 8, p-value = 0.01
alternative hypothesis: stationary
```

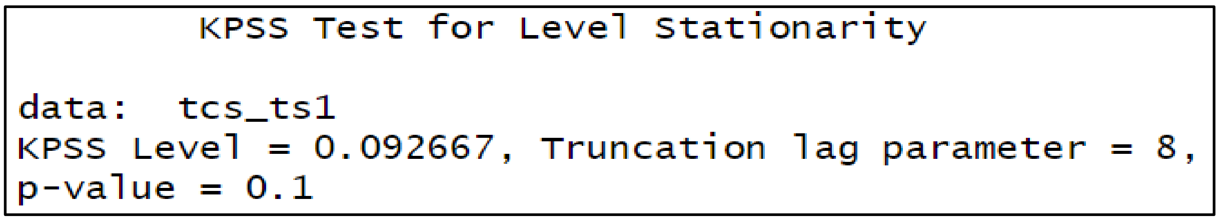


All three tests and line plot suggest that differencing is sufficient to make S&P 500 closing price stationary.

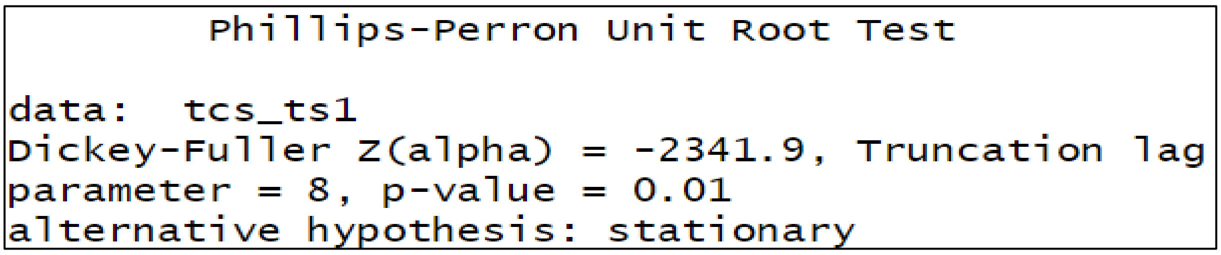
Differenced TCS Closing Price



```
          Augmented Dickey-Fuller Test

data:  tcs_ts1
Dickey-Fuller = -13.429, Lag order = 13, p-value = 0.01
alternative hypothesis: stationary
```



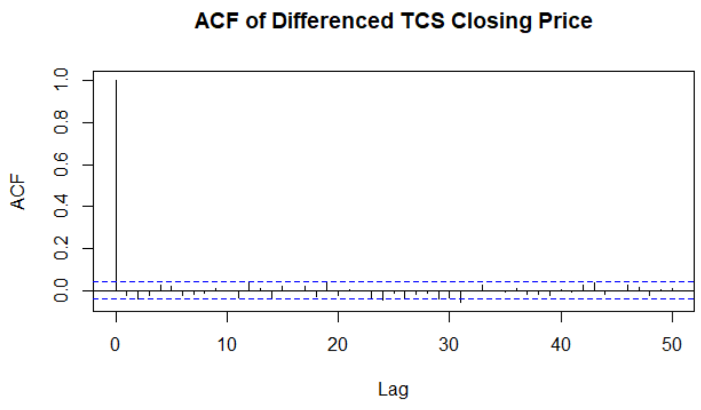
```
          KPSS Test for Level Stationarity

data:  tcs_ts1
KPSS Level = 0.092667, Truncation lag parameter = 8,
p-value = 0.1
```

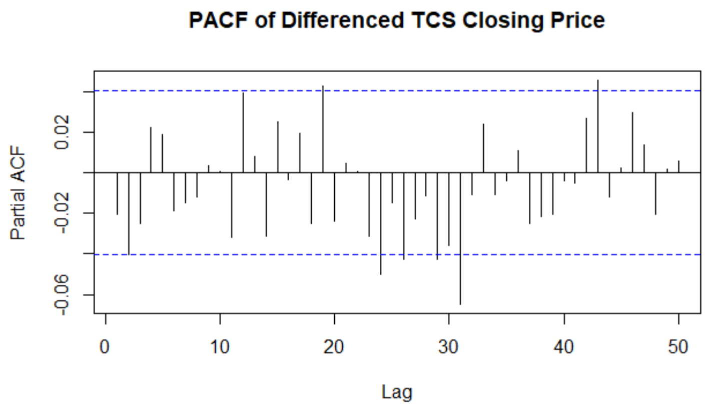


```
          Phillips-Perron Unit Root Test

data:  tcs_ts1
Dickey-Fuller Z(alpha) = -2341.9, Truncation lag
parameter = 8, p-value = 0.01
alternative hypothesis: stationary
```

All three tests and line plot suggest that differencing is sufficient to make TCS closing price stationary.

ACF & PACF Plots to determine p and q parameters of ARIMA or ARIMAX model





Since our data is daily stock prices, considering **50 lags** means looking at autocorrelations over approximately 2.5 months. As lag increases, the correlation values become **less reliable** due to noise and more lags mean more computation. Additionally, these models typically use **short-to medium-term dependencies**. Hence 50 lags are sufficient. Since ACF is decreasing quickly to zero, the series is stationary. The zeroth lag has a high positive correlation as expected in

ACF as it is correlation with itself. The rest of the lags are within the blue dashed lines, meaning they are statistically insignificant. A few lags cross the blue confidence bounds in the PACF plot, suggesting significant partial autocorrelations at those lags. 3 out of 4 significant peaks are barely just above the significant line, so we can say only one truly significant peak exists at lag 31. This pattern indicates **ARIMAX (p,1,0)** model may be suitable, where p = 4 or p =1 is the number of significant PACF lags and one differencing d term. The lack of strong early-lag spikes in PACF plot might also suggest **ARIMAX (p,1,0)** model might not be appropriate, as AR terms typically appear at early lags. Significant spikes at lags 18, 24, 31, and 45 in the pacf plot could indicate a seasonal or cyclical pattern rather than a typical AR process. Sarima model maybe more appropriate.

Correlations between external variables and TCS Closing Price

Correlation between Nifty 50 Closing Price and TCS Closing Price: 0.95

Correlation between S&P 500 Closing Price and TCS Closing Price: 0.97

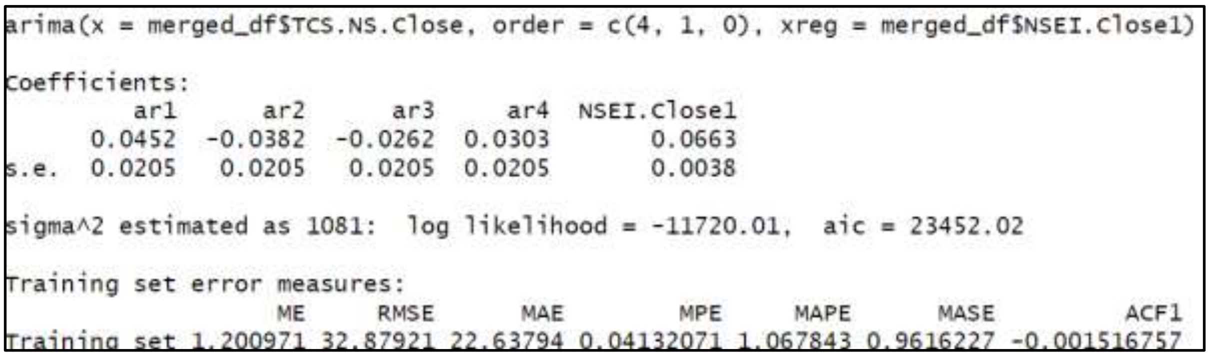
Correlation between TCS traded Volume and TCS Closing Price: -0.018

High positive correlation observed between Nifty 50 price and TCS Closing Price suggesting as Nifty 50 increases, TCS price increases. Similar interpretation for relationship between S&P 500 Closing Price and TCS Closing Price. Correlation between TCS traded Volume and TCS Closing Price is negative but approximately 0 suggesting no correlation. **Since no correlation, we cannot use TCS Trading Volume for ARIMAX model forecasting of TCS closing Price.**

Model Creation

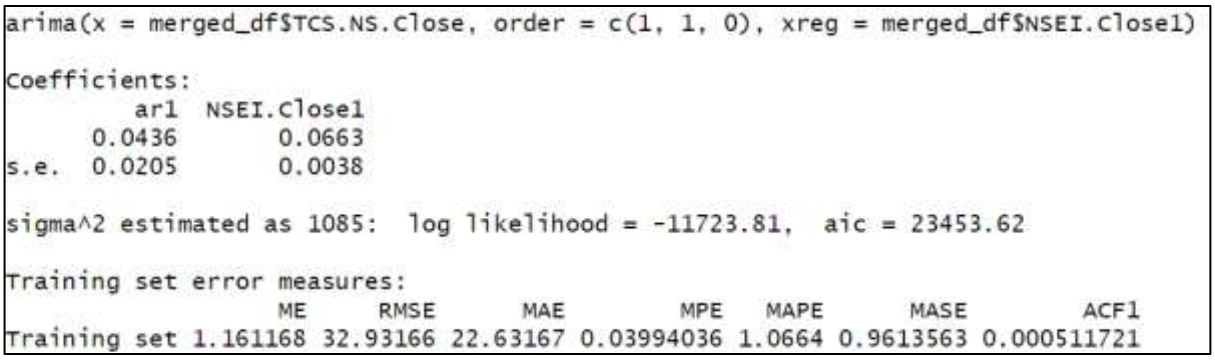
We check the model suggested by ACF and PACF plots and evaluate it with each external variable. We compare to find the best external variable for prediction using evaluation parameters MAE, RMSE< AIC etc. We then use auto arima to get the optimal model for the chosen external variable.

Summary of ARIMAX model (4, 1,0) to predict TCS closing price with Nifty 50 Closing price as external variable. Note here, only the Nifty 50 Price is differenced as the model will difference the TCS closing price. Manually differencing TCS price before inputting in this model will result in double differencing.



```
arima(x = merged_df$TCS.NS.Close, order = c(4, 1, 0), xreg = merged_df$NSEI.Close1)

Coefficients:
         ar1      ar2      ar3     ar4  NSEI.Close1
      0.0452  -0.0382  -0.0262  0.0303       0.0663
s.e.  0.0205   0.0205   0.0205  0.0205       0.0038

sigma^2 estimated as 1081:  log likelihood = -11720.01,  aic = 23452.02

Training set error measures:
                   ME     RMSE      MAE        MPE     MAPE      MASE          ACF1
Training set 1.200971 32.87921 22.63794 0.04132071 1.067843 0.9616227 -0.001516757
```

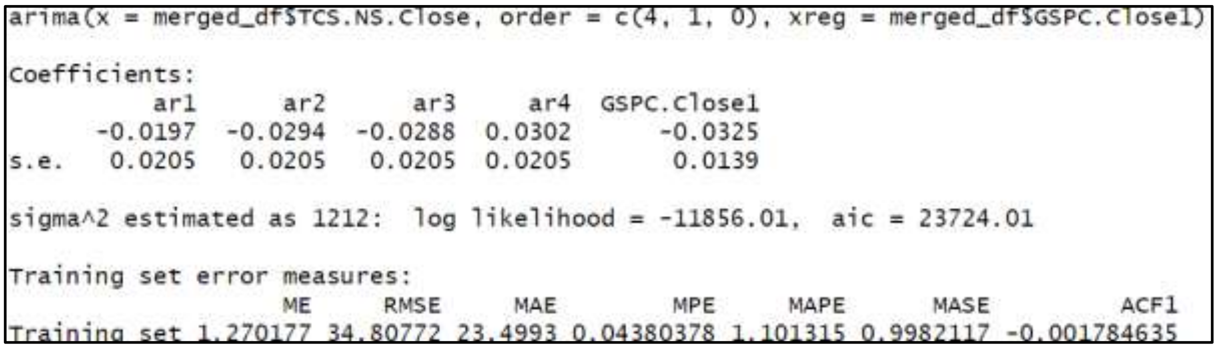
Summary of ARIMAX model (1,1,0) to predict TCS closing price with S&P 500 Closing price as external variable.



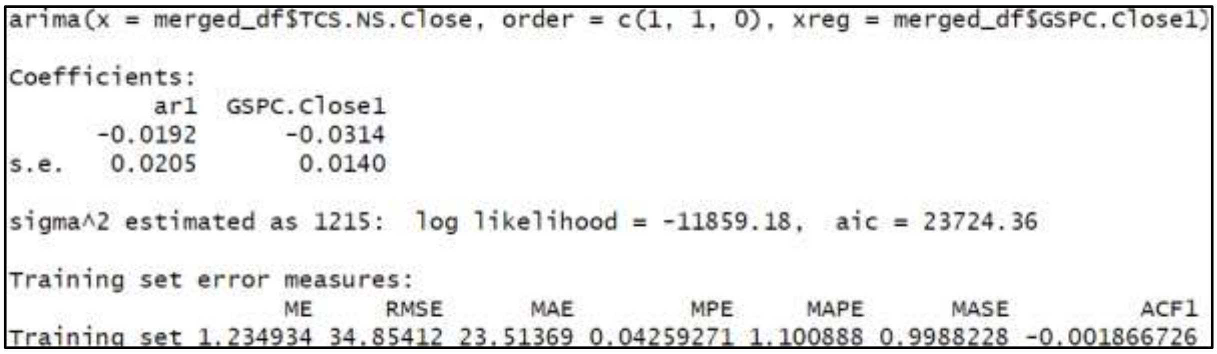
```
arima(x = merged_df$TCS.NS.Close, order = c(1, 1, 0), xreg = merged_df$NSEI.Close1)

Coefficients:
         ar1  NSEI.Close1
      0.0436       0.0663
s.e.  0.0205       0.0038

sigma^2 estimated as 1085:  log likelihood = -11723.81,  aic = 23453.62

Training set error measures:
                   ME     RMSE      MAE        MPE   MAPE      MASE        ACF1
Training set 1.161168 32.93166 22.63167 0.03994036 1.0664 0.9613563 0.000511721
```

We see that reducing the number of AR terms from 4 to 1 does not affect RMSE (difference between actual and predicted values by model) or MAPE (percent away from truth) significantly. So, we can choose to have only one term as it reduces the complexity of the model without sacrificing accuracy by much.

Summary of ARIMAX models (4,1,0) & (1,1,0) to predict TCS closing price with S&P 500 Closing price as external variable.



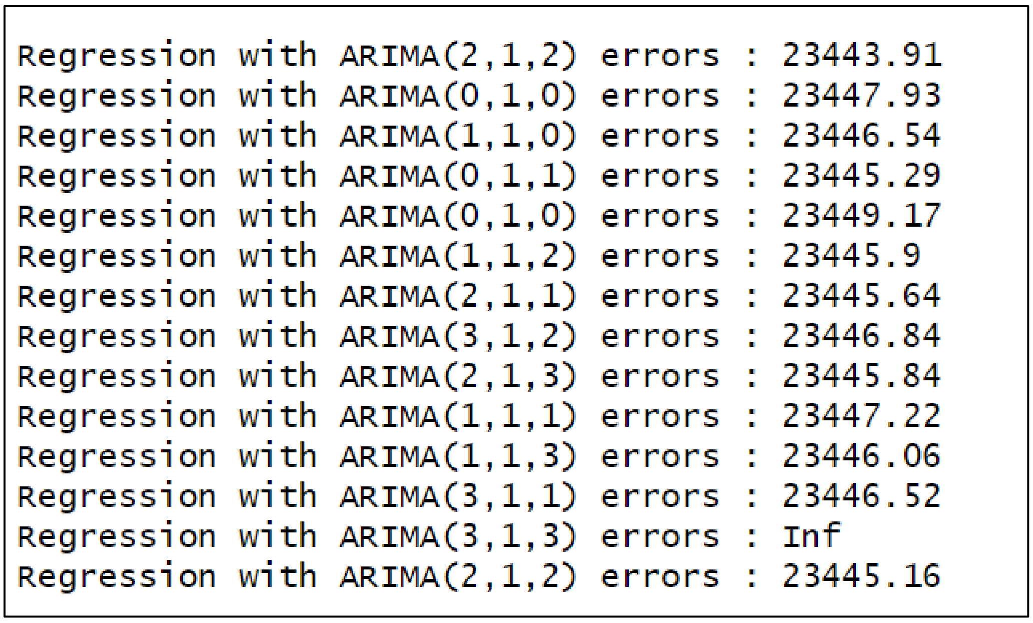
```
arima(x = merged_df$TCS.NS.Close, order = c(4, 1, 0), xreg = merged_df$GSPC.Close1)

Coefficients:
          ar1      ar2      ar3     ar4  GSPC.Close1
      -0.0197  -0.0294  -0.0288  0.0302      -0.0325
s.e.   0.0205   0.0205   0.0205  0.0205       0.0139

sigma^2 estimated as 1212:  log likelihood = -11856.01,  aic = 23724.01

Training set error measures:
                   ME     RMSE     MAE        MPE     MAPE      MASE         ACF1
Training set 1.270177 34.80772 23.4993 0.04380378 1.101315 0.9982117 -0.001784635
```



```
arima(x = merged_df$TCS.NS.Close, order = c(1, 1, 0), xreg = merged_df$GSPC.Close1)

Coefficients:
          ar1  GSPC.Close1
      -0.0192      -0.0314
s.e.   0.0205       0.0140

sigma^2 estimated as 1215:  log likelihood = -11859.18,  aic = 23724.36

Training set error measures:
                   ME     RMSE      MAE        MPE     MAPE      MASE         ACF1
Training set 1.234934 34.85412 23.51369 0.04259271 1.100888 0.9988228 -0.001866726
```

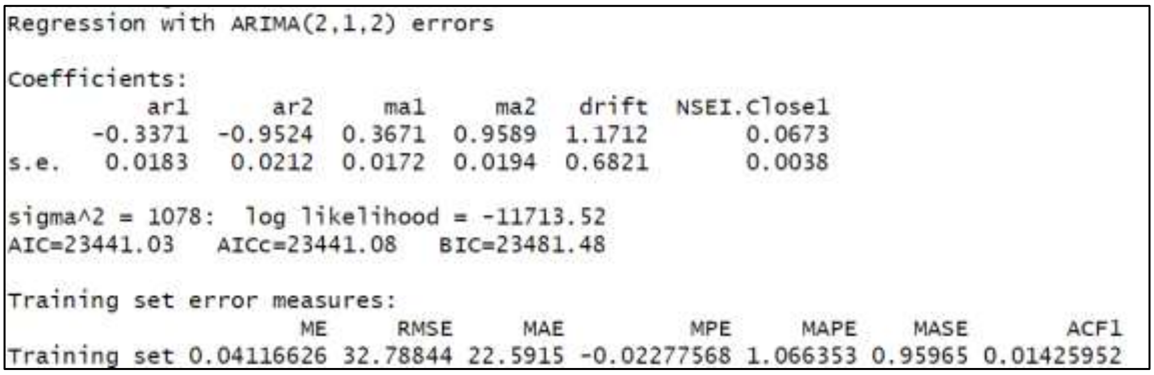
Similar effect with reducing terms using the S&P 500 variable. We see between the four models, the **best model might be ARIMAX model using Nifty 50 closing price variable as the exogenous variable with model parameters (1,1,0)**, as it has a lower RMSE and MAPE and low complexity.

We use auto arima function to identify optimal model to forecast TCS closing price using the exogenous variable Nifty 50 closing price. Auto Arima suggests (2,1,2) is the optimal model with lowest AIC. It evaluated the following models.

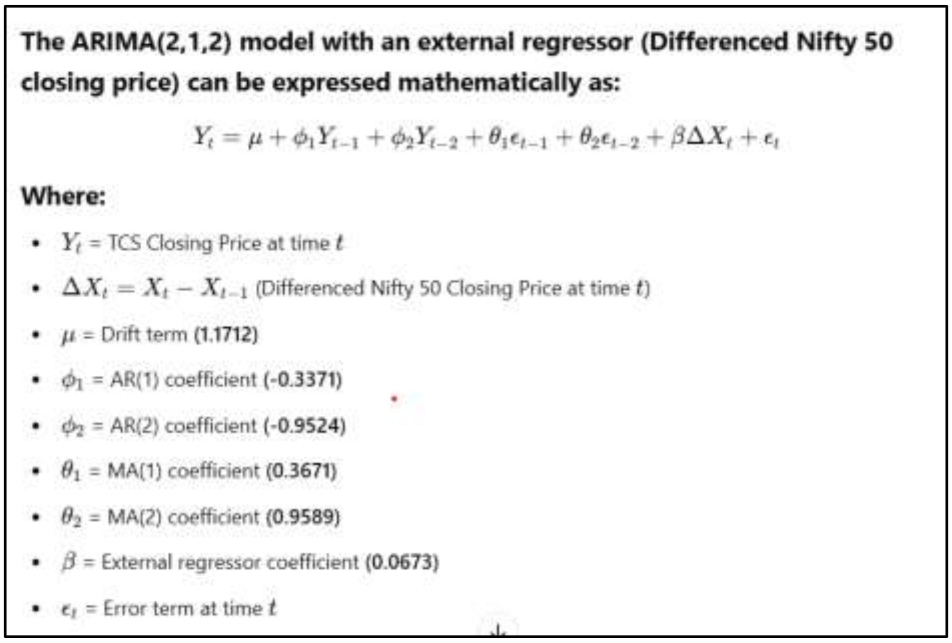


```
Regression with ARIMA(2,1,2) errors : 23443.91
Regression with ARIMA(0,1,0) errors : 23447.93
Regression with ARIMA(1,1,0) errors : 23446.54
Regression with ARIMA(0,1,1) errors : 23445.29
Regression with ARIMA(0,1,0) errors : 23449.17
Regression with ARIMA(1,1,2) errors : 23445.9
Regression with ARIMA(2,1,1) errors : 23445.64
Regression with ARIMA(3,1,2) errors : 23446.84
Regression with ARIMA(2,1,3) errors : 23445.84
Regression with ARIMA(1,1,1) errors : 23447.22
Regression with ARIMA(1,1,3) errors : 23446.06
Regression with ARIMA(3,1,1) errors : 23446.52
Regression with ARIMA(3,1,3) errors : Inf
Regression with ARIMA(2,1,2) errors : 23445.16
```

We observe ARIMA (2,1,2) has the lowest AIC score. The summary & mathematical formulation of this model is below.



```
Regression with ARIMA(2,1,2) errors

Coefficients:
          ar1      ar2     ma1     ma2   drift  NSEI.Close1
      -0.3371  -0.9524  0.3671  0.9589  1.1712       0.0673
s.e.   0.0183   0.0212  0.0172  0.0194  0.6821       0.0038

sigma^2 = 1078:  log likelihood = -11713.52
AIC=23441.03   AICc=23441.08   BIC=23481.48

Training set error measures:
                     ME     RMSE     MAE         MPE     MAPE    MASE       ACF1
Training set 0.04116626 32.78844 22.5915 -0.02277568 1.066353 0.95965 0.01425952
```

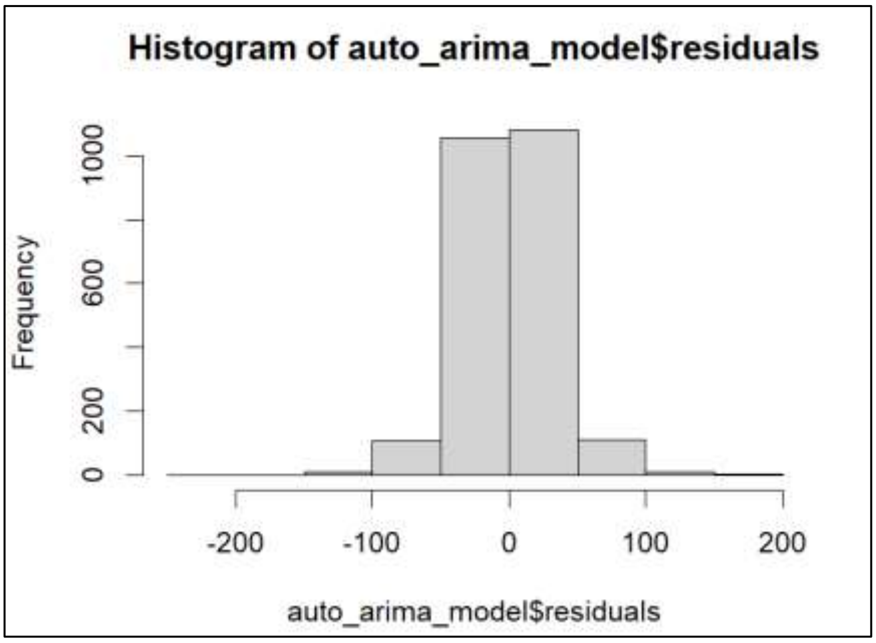


The ARIMA(2,1,2) model with an external regressor (Differenced Nifty 50 closing price) can be expressed mathematically as:

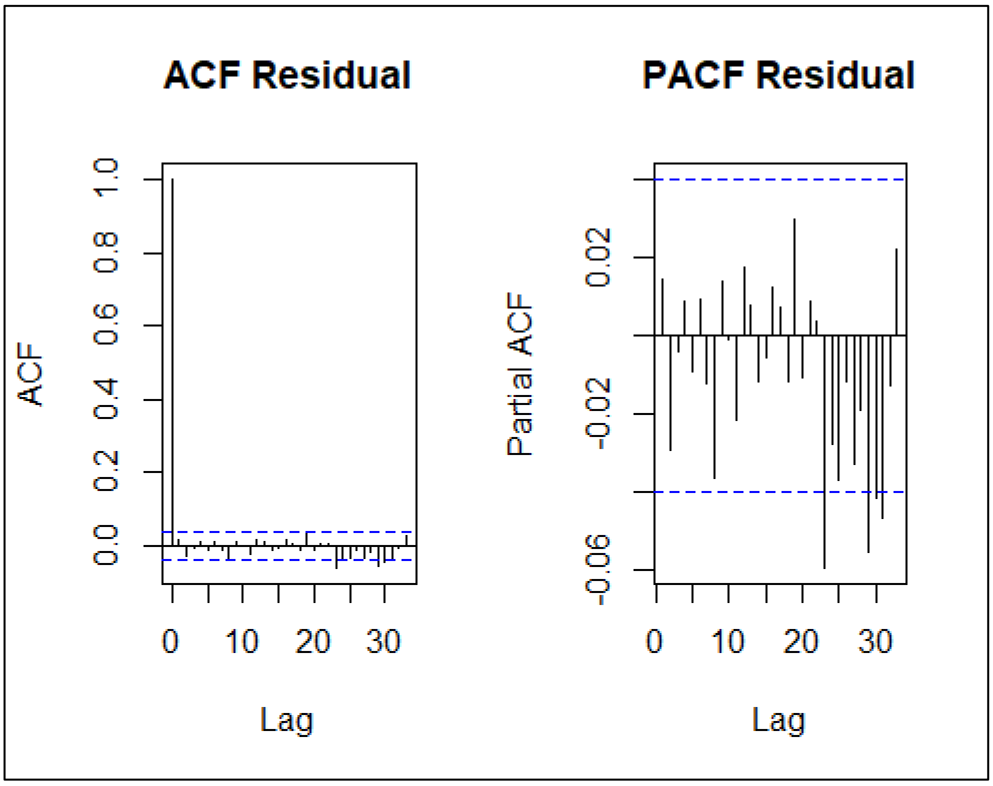
$$Y_t = \mu + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \beta \Delta X_t + \epsilon_t$$

Where:

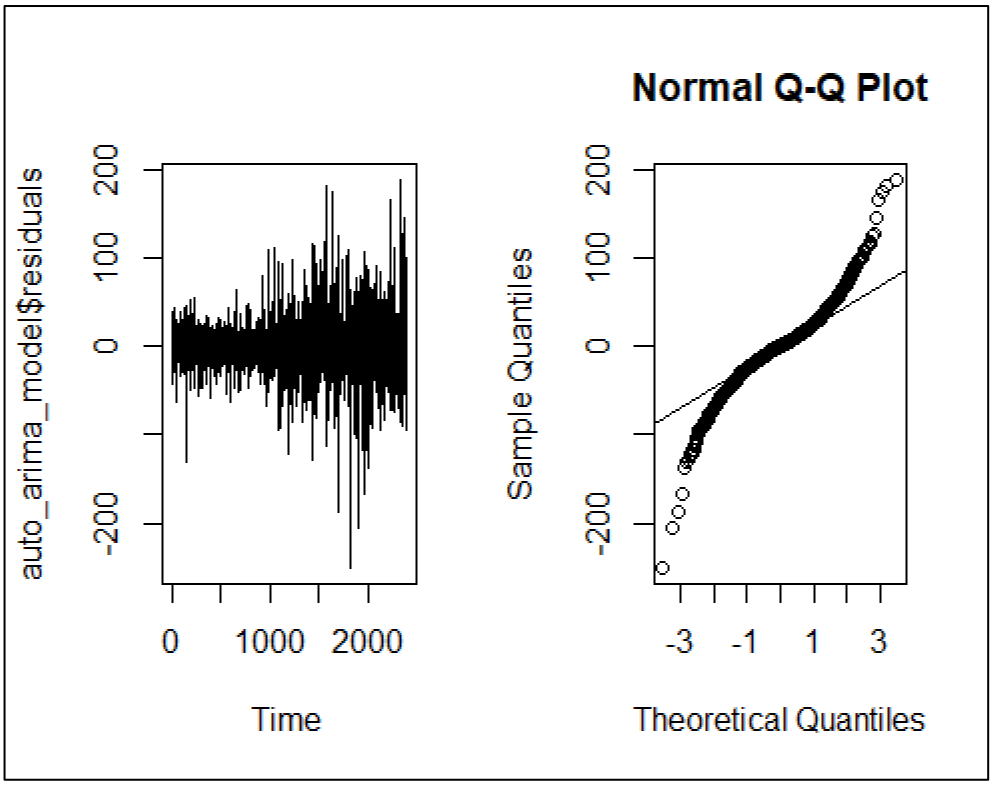
- Y_t = TCS Closing Price at time t
- $\Delta X_t = X_t - X_{t-1}$ (Differenced Nifty 50 Closing Price at time t)
- μ = Drift term **(1.1712)**
- ϕ_1 = AR(1) coefficient **(-0.3371)**
- ϕ_2 = AR(2) coefficient **(-0.9524)**
- θ_1 = MA(1) coefficient **(0.3671)**
- θ_2 = MA(2) coefficient **(0.9589)**
- β = External regressor coefficient **(0.0673)**
- ϵ_t = Error term at time t



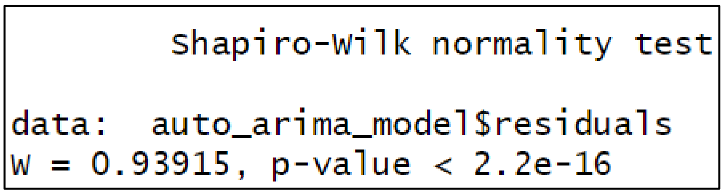
Histogram of residuals suggests residuals are normally distributed with mean zero. But we will check with normality tests on the next page.



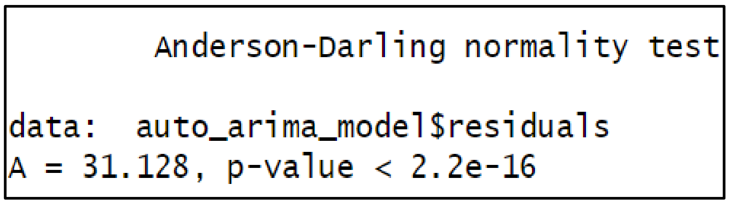
The residuals are within the blue bands for ACF except zeroth lag which is expected. Remaining within blue bars suggest there is no autocorrelation between residuals. A couple bars are outside the blue bands in PACF but since majority are within, we can say most lags do not show significant correlation with past residuals. It means the model has captured most of the time-dependent structure in the data. A few points outside these bands are not necessarily a problem. It suggests Model may need additional AR/MA terms or seasonal components or both.



The residuals deviate significantly from the straight diagonal line, especially at the extremes. This means residuals do not follow a normal distribution—they have fat tails (extreme values occur more often than expected). Despite differencing TCS closing price once resulting in stationary data as per our tests, these plots suggest potential outliers or a need for transformation (e.g., log transformation). The residuals appear to have small variance initially but increase over time. This suggests heteroscedasticity, meaning the model does not have constant variance, suggesting missing volatility modelling.



```
        Shapiro-Wilk normality test

data:  auto_arima_model$residuals
W = 0.93915, p-value < 2.2e-16
```

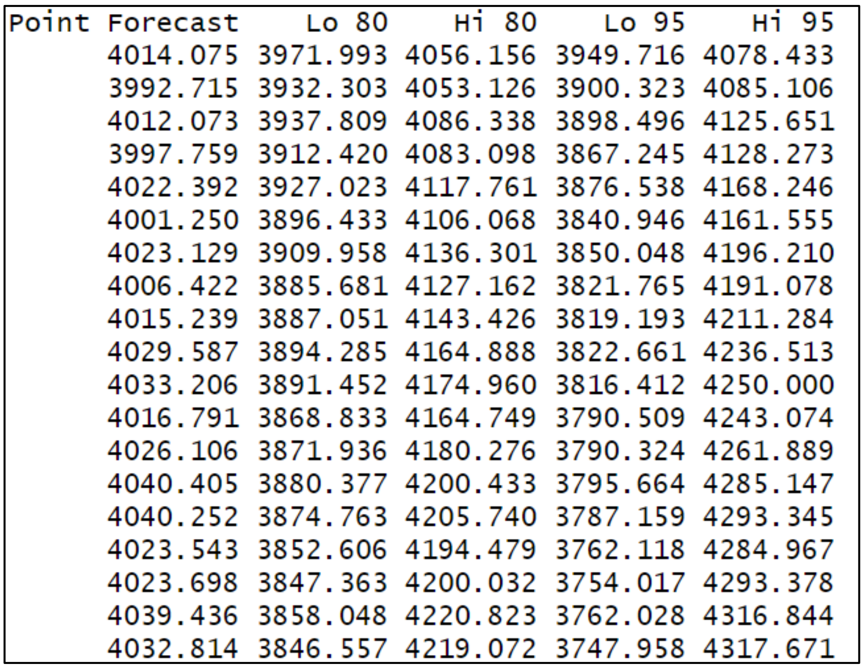


```
        Anderson-Darling normality test

data:  auto_arima_model$residuals
A = 31.128, p-value < 2.2e-16
```

The Shapiro Wilk normality test and Anderson Darling normality test both with Null Hypothesis Data follows normal distribution are conducted. P-value < 0.05 rejects null hypothesis for both confirming residuals do not follow normal distribution. This suggests the need for removal of extreme outliers or include additional AR/MA terms and/or seasonality terms.

DISCUSSION

Below is the **forecast** for the next 30 days (6 March 2024 – 6 April 2024) based on 10-year historical data from March 1 2014 to March 5, 2024 using ARIMAX (2,1,2) with external variable Nifty 50.



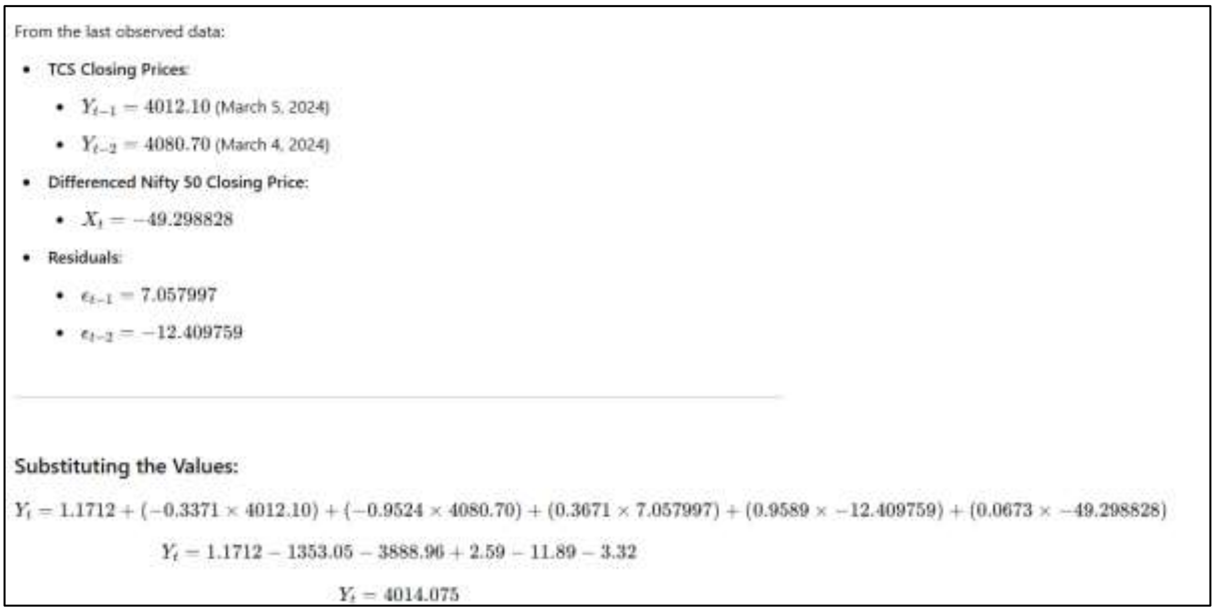
```
Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
      4014.075 3971.993 4056.156 3949.716 4078.433
      3992.715 3932.303 4053.126 3900.323 4085.106
      4012.073 3937.809 4086.338 3898.496 4125.651
      3997.759 3912.420 4083.098 3867.245 4128.273
      4022.392 3927.023 4117.761 3876.538 4168.246
      4001.250 3896.433 4106.068 3840.946 4161.555
      4023.129 3909.958 4136.301 3850.048 4196.210
      4006.422 3885.681 4127.162 3821.765 4191.078
      4015.239 3887.051 4143.426 3819.193 4211.284
      4029.587 3894.285 4164.888 3822.661 4236.513
      4033.206 3891.452 4174.960 3816.412 4250.000
      4016.791 3868.833 4164.749 3790.509 4243.074
      4026.106 3871.936 4180.276 3790.324 4261.889
      4040.405 3880.377 4200.433 3795.664 4285.147
      4040.252 3874.763 4205.740 3787.159 4293.345
      4023.543 3852.606 4194.479 3762.118 4284.967
      4023.698 3847.363 4200.032 3754.017 4293.378
      4039.436 3858.048 4220.823 3762.028 4316.844
      4032.814 3846.557 4219.072 3747.958 4317.671
```

19 forecasts for the 19 Nifty 50 closing prices available on the 19 trading days.

The equation for the optimal model is

$Y_t = 1.1712 - 0.3371Y_{t-1} - 0.9524Y_{t-2} + 0.3671\epsilon_{t-1} + 0.9589\epsilon_{t-2} + 0.0673(X_t - X_{t-1}) + \epsilon_t$

Thus, the first forecast is calculated as follows.



From the last observed data:

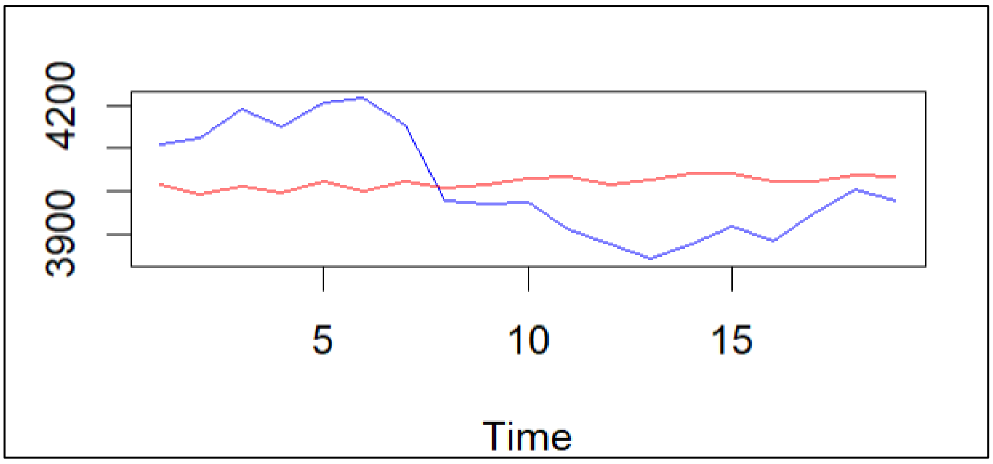
- TCS Closing Prices:
  - $Y_{t-1} = 4012.10$ (March 5, 2024)
  - $Y_{t-2} = 4080.70$ (March 4, 2024)
- Differenced Nifty 50 Closing Price:
  - $X_t = -49.298828$
- Residuals:
  - $\epsilon_{t-1} = 7.057997$
  - $\epsilon_{t-2} = -12.409759$

Substituting the Values:

$$Y_t = 1.1712 + (-0.3371 \times 4012.10) + (-0.9524 \times 4080.70) + (0.3671 \times 7.057997) + (0.9589 \times -12.409759) + (0.0673 \times -49.298828)$$

$$Y_t = 1.1712 - 1353.05 - 3888.96 + 2.59 - 11.89 - 3.32$$

$$Y_t = 4014.075$$



For the 19 time points, blue line shows actual TCS closing price values and red shows the predicted values by the chosen model. It initially under forecasts and then over forecasts.

The model can reduce its complexity by dropping the drift term. The drift term can be excluded as its value is lower than twice the standard error (2*SE).

Model Evaluation using MAPE, MAE, RMSE, and AIC

The ARIMAX (2,1,2) model with Nifty 50 closing price as an external variable was selected based on key evaluation metrics:

- **MAPE: 1.06 %** – MAPE measures the average percentage error between the predicted and actual values. It is useful for understanding how far predictions deviate from actual values in percentage terms. The model under-predicts or over forecasts by 1.06 % on average. A **1% error** suggests that the model is making highly accurate predictions especially for highly volatile data like stock data.

  - Formula:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right| \times 100$$

  - Where:
    - Y_i = Actual value
    - $\hat{Y}_i$ = Predicted value
    - n = Number of observations

- **RMSE (Standard Error): 32.79** – We can be 68% confident that the actual value falls within ±32.79 of the predicted value and 95% confident within ±65.58.

  - Formula:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}$$

- **MAE – 22.59** Measures the average magnitude of prediction errors. Low values indicate better models. It is quite low for this use case and thus our model is pretty good.

Formula:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i|$$

- **AIC (Akaike Information Criterion)** – Used to compare models, where a lower AIC indicates a better fit. The ARIMAX (2,1,2) model was optimal as it had the lowest AIC among tested models.

- Formula:

$$AIC = 2k - 2\ln(L)$$

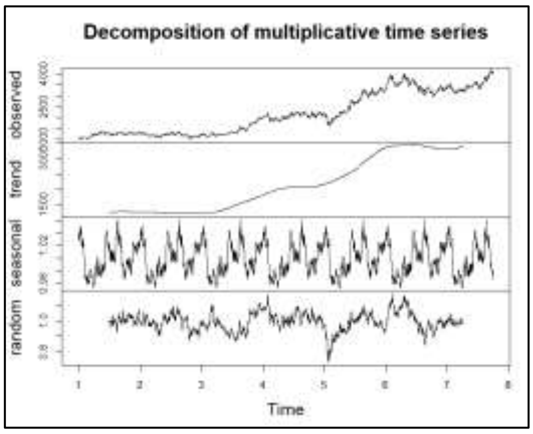
- Where:
  - k = Number of model parameters
  - L = Maximum likelihood estimate of the model

A more complex model generally has lower bias but higher variance, whereas a simpler model has higher bias but lower variance. RMSE measures variance, and MAPE represents bias. A model must balance both for optimal forecasting. We could pick a potentially better model which is simpler and more general by compromising on MAPE and lowering the RMSE value.

Impact of Non-Normal Residuals and Remedies

The Shapiro-Wilk and Anderson-Darling tests rejected normality. The Q-Q plot showed fat tails, suggesting extreme values occur more frequently than a normal distribution predicts. To address this:

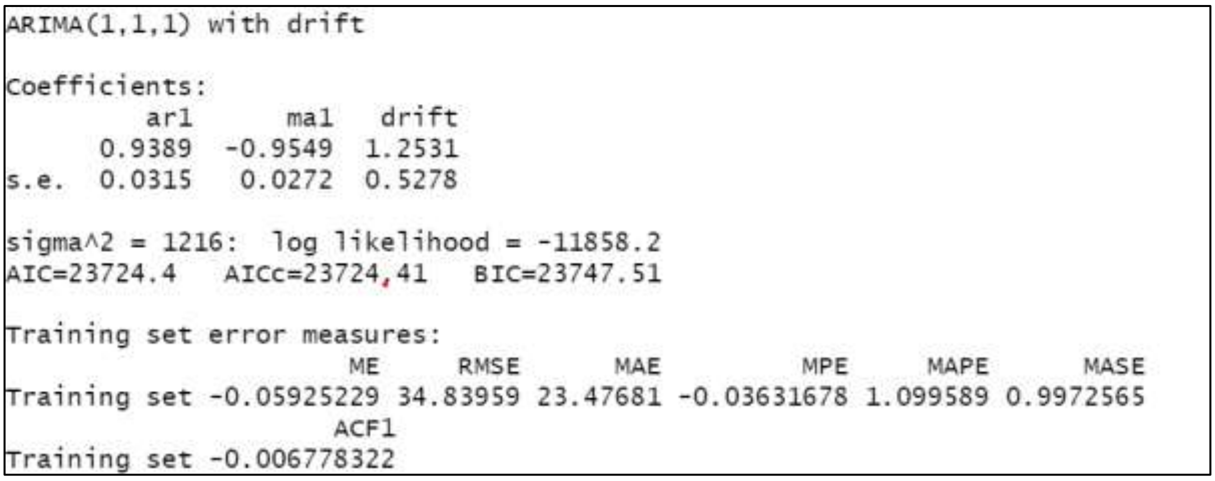
- Outliers such as the 2020 COVID-19 market crash should be accounted for.
- Transformations like log transformation can stabilize variance.
- Adding seasonal components or higher-order AR/MA terms may improve residual normality. Checked for seasonality by decomposing the TCS Closing Price.



The third panel labelled **"seasonal"** shows a repeating pattern at regular intervals, which indicates **seasonality** in the data. Thus, SARIMAX or SARIMA model might be more appropriate.

ARIMA MODEL

Nifty 50 and S&P 500 represent a basket of companies across various sectors, whereas TCS is a single IT company with its own fundamentals. Broader indices react to macro events (interest rates, inflation, global crises), whereas individual stocks also react to micro events (company earnings, sector performance). Correlation does not imply causation—TCS stock may move independently due to company-specific factors like earnings reports, client deals, and management decisions. A global market crash might impact indices, but TCS might still perform well due to strong IT demand or favourable USD-INR movement. Nifty 50 consists of companies from multiple sectors (IT, banks, FMCG, pharma, etc.), and their performance averages out in index movements. S&P 500 has even broader exposure to different economies, making it less relevant to a single Indian stock. In such cases, ARIMA model maybe more favourable.



```
ARIMA(1,1,1) with drift

Coefficients:
         ar1      ma1   drift
      0.9389  -0.9549  1.2531
s.e.  0.0315   0.0272  0.5278

sigma^2 = 1216:  log likelihood = -11858.2
AIC=23724.4   AICc=23724.41   BIC=23747.51

Training set error measures:
                      ME     RMSE      MAE         MPE     MAPE      MASE
Training set -0.05925229 34.83959 23.47681 -0.03631678 1.099589 0.9972565
                      ACF1
Training set -0.006778322
```

ARIMA (1,1,1) is optimal using auto.arima for forecasting TCS Stock prices and we observe the RMSE, MAE and MAPE are approximately equal even without the external variables.

Effect of Outliers (e.g., COVID-19)

The sharp market drop in 2020, influenced by COVID-19, introduced volatility affecting the stationarity of the time series. This suggests incorporating models that account for varying volatility.

Limitations

The model assumes that Nifty 50 significantly influences TCS stock price, but TCS-specific factors (earnings, contracts, global IT demand) play a crucial role. Future research should explore alternative exogenous variables, such as USD-INR exchange rates, IT sector indices, and global economic indicators, to improve forecasting accuracy. **Data from 10 years ago** may not reflect **current market conditions**, making long-term models less relevant. A shorter time frame ensures that forecasts are based on **recent patterns** rather than outdated trends. Further research should use a **shorter time frame** (e.g., 6 months to 2 years) rather than **10 years** for forecasting.

CONCLUSION

The ARIMAX (2,1,2) model effectively captures the relationship between TCS closing price and Nifty 50 closing price. However, non-normal residuals and market outliers indicate areas for further refinement, such as incorporating volatility models and handling extreme events. The ARIMAX model effectively integrates external variables to enhance time series forecasting, demonstrating its superiority over standard ARIMA in scenarios where exogenous factors play a crucial role. By incorporating these additional predictors, the model can improve accuracy and provide a more nuanced understanding of the underlying data patterns. However, the effectiveness of ARIMAX depends on careful selection and preprocessing of exogenous variables, ensuring they contribute meaningful insights rather than introducing noise. Overall, the model is a powerful tool for forecasting, particularly in fields like finance, economics, and demand prediction, where external influences significantly impact outcomes.

KEY FINDINGS

ARIMAX or ARIMA: By incorporating exogenous variables, ARIMAX can improve forecasting accuracy beyond traditional ARIMA models. But choosing ARIMAX over ARIMA should depend on careful selection of external variable that does not create noise or add complexity without improving accuracy.

Importance of Exogenous Variables: The model's performance depends on selecting relevant external factors that influence the time series data.

Data Preprocessing Matters: Proper handling of missing values, outliers, scaling, and stationarity checks are crucial for effective modelling.

Model Complexity: While ARIMAX is powerful, it requires careful tuning of parameters and validation to avoid overfitting.

Application Across Domains: The ARIMAX model is particularly useful in finance, economics, and business forecasting, where external factors impact trends.

PRACTICAL IMPLICATIONS

1. Improved Forecasting for Investment Decisions

- The ARIMAX model helps investors predict TCS stock price movements more accurately by integrating Nifty 50 as an external variable.
- This can assist portfolio managers, traders, and analysts in making data-driven investment decisions.
- However, since broader indices do not always drive individual stock prices, investors should use this model alongside fundamental analysis.

2. Risk Management for Institutional Investors

- By understanding how Nifty 50 influences TCS stock prices, institutional investors can create hedging strategies to reduce risk.
- If a strong dependency is found, investors can use index-based derivatives (e.g., Nifty futures and options) to hedge TCS stock positions.

3. Strategic Decision-Making for Business including TCS

- TCS and other IT firms can use such models to forecast stock price trends and gauge market sentiment.
- This can help in strategic decisions like timing share buybacks, issuing new stock, or setting executive compensation linked to stock performance.

4. Enhancing Algorithmic Trading Strategies

- Quantitative traders and hedge funds can integrate ARIMAX-based forecasts into automated trading algorithms to exploit short-term price movements.
- If the model captures significant patterns, it can be combined with technical indicators for improved trading signals.

5. Macroeconomic Analysis & Policy Insights

- If the model establishes a strong relationship between TCS stock price and Nifty 50, it can indicate broader market trends affecting the IT sector.
- Policymakers and market regulators can use such insights to assess sector-specific impacts of macroeconomic policies (e.g., interest rate changes, global tech regulations).

REFERENCES

Time Series Forecasting with ARIMA, SARIMA, and SARIMAX. Towards Data Science. Retrieved from https://towardsdatascience.com/time-series-forecasting-with-arima-sarima-and-sarimax-ee61099e78f6/

National Stock Exchange of India. (Year). *TCS Stock Quote*. Retrieved from https://www.nseindia.com/get-quotes/equity?symbol=TCS

Tata Consultancy Services. (Year). *Who We Are*. Retrieved from https://www.tcs.com/who-we-are

ANNEXURES

R Code for the above analysis

```
###Installing packages

install.packages("quantmod")  # For downloading stock data

install.packages("tidyquant") # Alternative package for finance data

install.packages("dplyr")     # For data manipulation

install.packages("xts")     # For handling missing trading data automatically

install.packages("lubridate") # For handling dates

###loading packages

library(quantmod)

library(tidyquant)

library(dplyr)

library(xts)

library(forecast)

library(tseries)

library(lubridate)

library(stats)

library (nortest)

# Define stock symbol and date range

stock_symbol <- "^NSEI"  # Replace with any stock symbol

start_date <- "2014-03-01"

end_date <- "2024-03-06"

# Get data using quantmod

stock_data <- getSymbols(stock_symbol, src = "yahoo", from = start_date, to = end_date,
auto.assign = F)

stock_data = na.omit(stock_data)

# Convert to a data frame

nifty_data <- data.frame(Date = index(stock_data), coredata(stock_data))

# View first few rows
```

```
head(nifty_data)

#write to csv

write.csv(nifty_data, file = "D:/R_practice/nifty_data_test.csv", row.names = FALSE)

# Load stock data

nifty_data <- read.csv("D:/R_practice/nifty_data_test.csv")

# Convert Date column to Date format

nifty_data$Date <- as.Date(nifty_data$Date, format="%Y-%m-%d")

# Convert to xts time series object (handles missing dates automatically)

nifty_xts<- xts(nifty_data[, -1], order.by = nifty_data$Date)

# Check structure

head(nifty_xts)
```

##Visualization and Stationarity test

```
plot(nifty_xts$NSEI.Close, main = "Nifty 50 Closing Price", ylab = "Nifty 50 Closing Price",
xlab = "Time")

adf.test(nifty_xts$NSEI.Close)

kpss.test(nifty_xts$NSEI.Close)

pp.test(nifty_xts$NSEI.Close)
```

Do the same to get S&P 500 closing price using stock symbol ^GSPC. The tcs price is given by professor and needs to be loaded using read CSV.

```
tcs_data <- read.csv("D:/R_practice/TCS_stock_data.csv")
```

###Differencing and subsequent stationarity test, other tests can be implemented such as KPSS and PP using above mentioned codes

```
sp500_xts$GSPC.Close1=diff(sp500_xts$GSPC.Close)

nifty_xts$NSEI.Close1=diff(nifty_xts$NSEI.Close)

adf.test(sp500_xts$GSPC.Close1)
```

##Merging data

```
merged_df <- merge.xts(nifty_xts$NSEI.Close1, sp500_xts$GSPC.Close1,
tcs_xts$TCS.NS.Close, tcs_xts$TCS.NS.Volume, by = "Date", all = FALSE)

merged_df = merged_df[,-5]

merged_df = na.omit(merged_df)
```

# Create time series objects and plot differenced variables

nifty_ts <- ts(merged_df$NSEI.Close1)

sp500_ts <- ts(merged_df$GSPC.Close1)

plot(nifty_ts, main = "Differenced Nifty Closing Price", ylab = "Diff Nifty Closing Price")

plot(sp500_ts, main = "Differenced S&P500 Closing Price", ylab = "Diff S&P Closing Price")

## Code to check for Correlations

cor(merged_df2$TCS.NS.Volume,merged_df2$TCS.NS.Close), where merged_df2 contains both the variables for which correlation is being checked at the same time points.

##Model using auto arima

arima_model = auto.arima(merged_df$TCS.NS.Close)

summary(arima_model)

auto_arima_model <- auto.arima(merged_df$TCS.NS.Close, xreg = merged_df$NSEI.Close1, trace = T, ic = 'aic')

summary(auto_arima_model)

##ACF & PACF of TCS closing Price differenced

acf(merged_df1$TCS.NS.Close1, main = "ACF of Differenced TCS Closing Price", lag.max = 50)

pacf(merged_df1$TCS.NS.Close1, main = "PACF of Differenced TCS Closing Price", lag.max = 50)

##Model by specifying order

arima_model = arima(merged_df$TCS.NS.Close, order = c(4,1,0), xreg = merged_df$NSEI.Close1)

summary(arima_model)

###ACF and PACF of Residual

acf(auto_arima_model$residuals, main='ACF Residual')

pacf(auto_arima_model$residuals, main='PACF Residual')

##Residual Normality check

hist(auto_arima_model$residuals)

plot(auto_arima_model$residuals)

qqnorm(auto_arima_model$residuals)

qqline(auto_arima_model$residuals)

shapiro.test(auto_arima_model$residuals)

ad.test(auto_arima_model$residuals)

###Forecast

forecast_values <- forecast(auto_arima_model, xreg = nifty_xts$NSEI.Close1, h = 30)

print(forecast_values)

nifty_xts contains differenced Nifty 50 closing price value obtained from yahoo finance from 6th March 3024 to 6th April 2024 accounting for 19 trading days.

##Seasonality decomposition

tcs_ts <- ts(na.omit(tcs_xts$TCS.NS.Close), frequency = 365)

td <- decompose(tcs_ts, type = "multiplicative")

plot(td)

tcs_xts object contains all the relevant stock data of tcs as time series.

###Comparison between Forecast and Actual

t1= c (4108.6, 4122.35, 4192.25, 4149,4207.6, 4219.25, 4152.5, 3977.3, 3970.9,3972.95

,3910.9,3877.5,3840.9,3876.3,3916.75,3883.8,3947.3,4003.3,3979.25)

t2 <- c (4014.075, 3992.715, 4012.073, 3997.759, 4022.392, 4001.250, 4023.129,4006.422,4015.239,4029.587,4033.206,4016.791,4026.106,4040.405,4040.252,4023.543,4023.698, 4039.436,4032.814)

acw=cbind(t1,t2)

ts.plot(acw, col= c(blue,red))

t1 contains actual TCS closing price values and t2 forecasted.