Multivariate Time Series

Time Series Forecasting

Agenda

- Business Problem
- Approaches to Statistical Analysis in a Time Series
- General Applications of VAR Model.
- Forecasting VAR Model.
- Forecasting VARMA Model.

Business problem: A chain of stores which was been established and driving their operations across in the city of mexico, keen to predict the footfall of few specific stores of them.

- It is important for a stores to know there sales which was been influenced by multiple factors in which footfall was one of the major factor.

- This model forecast can be used to predict the footfall of the customers for an specific stores on which the management was keen to know there future footfall the stores was been labelled with an unique code ''1044","1041".

Dependent Variable

- The variable we wish to explain or predict
- Usually denoted by Y
- Dependent Variable = Response Variable = Target Variable
- Here 'Customers' is our target variable

Independent Variable

- The variables used to explain the dependent variable
- Usually denoted by X
- Independent Variable = Predictor Variable
- In our example, store,day of week, open, promotion, state holiday, school holiday are the independent variables,date

Visiting Basics

Data

Let us consider the following data.

Store	Day Of Week	Date	Customer	Open	Promotion	State Holiday	School Holiday
1044	1	5/31/2016	884	1	1	0	0
1041	1	5/31/2016	1032	1	1	0	1
1036	1	5/31/2016	1070	1	1	0	0
1047	1	5/31/2016	2043	1	1	0	0
1012	1	5/31/2016	1239	1	1	0	0
1037	1	5/31/2016	716	1	1	0	1
1027	1	5/31/2016	865	1	1	0	0
1005	1	5/31/2016	720	1	1	0	0

Approaches to Statistical Analysis in a Time Series

- Univariate Time-Series Analysis

- Multivariate Time-Series Analysis

Approaches to Statistical Analysis in a Time Series

Univariate Time Series :-

- The data which consists of single time-dependent variable.

- Example : Find the reference dataset that consists of the CO2 emissions (every month), for the previous few months. Here, CO2 is an dependent variable (dependent on Time/month).

Year-Month	CO2 ppm
1965-Jan	319.32
1965-Feb	320.36
1965-Mar	320.82
1965-Apr	322.06
1965-May	322.17

Approaches to Statistical Analysis in a Time Series

Multivariate Time Series :-

- The data which consists of more than one time-dependent feature ,Every individual features relay not on its previous/past values/data but exisiting of dependency with neighbour variables,This dependency helps the model to forecast the future values/data points.

Approaches to Statistical Analysis in a Time Series

Multivariate Time Series :-

- Example : Find the reference dataset that consists of the Room Occupancy along with temperature,humidity,light,CO2,humidity ratio of every minute for one week of data in a month. Here, there are multiple features to be considered to predict an optimal temperature . This series would come under multivariate time series category.

```
"date","Temperature","Humidity","Light","CO2","Humidity_Ratio","Occupancy"
"1","2015-02-04 17:51:00",23.18,27.272,426,721.25,0.00479298817650529,1
"2","2015-02-04 17:51:59",23.15,27.2675,429.5,714,0.00478344094931065,1
"3","2015-02-04 17:53:00",23.15,27.245,426,713.5,0.00477946352442199,1
"4","2015-02-04 17:54:00",23.15,27.2,426,708.25,0.00477150882608175,1
"5","2015-02-04 17:55:00",23.1,27.2,426,704.5,0.00475699293331518,1
"6","2015-02-04 17:55:59",23.1,27.2,419,701,0.00475699293331518,1
```

Vector Time Series Models

- Is one of the most common method used to deal with Multivariate Time-Series data.
- This is a stochastic process model utilized to seize the linear relation among the multiple variables of time-series data.
- In other words, it is a multivariate forecasting method utilized when two or more time-series variables have a strong internal relationship with each other.
- VAR is a bidirectional model, while others are unidirectional models.
- In a unidirectional model, a predictor influences the target, but not vice versa. In a bidirectional model, variables influence each other.

Why do we need VAR?

- Time-series data with autoregressive in nature (serially correlated)
- VAR model is one of the most successful and flexible models for the analysis of multivariate time series
- Especially useful for describing the dynamic behavior of economic and financial time series
- Useful for forecasting

General Applications of VAR

- In economics, VAR is used to forecast macroeconomic variables, such as GDP, money supply, and unemployment
- In finance, predict spot prices and future prices of securities; foreign exchange rates across markets
- Analysis of system response to different shocks/impacts
- Model-based forecast. In general VAR encompasses correlation information of the observed data and use this correlation information to forecast future movements or changes of the variable of interest

General Applications of VAR

- In accounting, predict different accounting variables such as sales, earnings, and accruals

- In marketing, VAR can be used to evaluate the impact of different factors on consumer behavior and forecast its future change.

Vector Time Series Models

- A stationary vector time series process Z_t that is purely non-deterministic can always be written as a moving average representation, and an invertible vector time series process can always be written as an autoregressive representation.

- This involve an infinite number of coefficient matrices in the representation. In practice, with a finite number of observations, we will construct a time series model with a finite number of coefficient matrices. Specifically, we will present VAR(p), VARMA(p, q).

Vector Autoregression - (VAR)

- The normal AR(p) model equation looks like this:

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots + \beta_p Y_{t-p} + \epsilon_t$$

- where α is the intercept, a constant and β1, β2 till βp are the coefficients of the lags of Y till order p.
- Order 'p' means, up to p-lags of Y is used and they are the predictors in the equation. The ε_{t} is the error, which is considered as white noise.

Vector Autoregression - (VAR)

- VAR model, each variable is modeled as a linear combination of past values of itself and the past values of other variables in the system. Since you have multiple time series that influence each other, it is modeled as a system of equations with one equation per variable (time series).
- That is, if you have 5 time series that influence each other, we will have a system of 5 equations.

Vector Autoregression - (VAR)

In General, how an equation is been exactly framed:

- Let's suppose, you have two variables (Time series) Y1 and Y2, and you need to forecast the values of these variables at time (t).
- To calculate Y1(t), VAR will use the past values of both Y1 as well as Y2. Likewise, to compute Y2(t), the past values of both Y1 and Y2 be used.

Vector Autoregression - (VAR)

- For example, the system of equations for a VAR(1) model with two time series (variables `Y1` and `Y2`) is as follows:

$$Y_{1,t} = \alpha_1 + \beta_{11,1} Y_{1,t-1} + \beta_{12,1} Y_{2,t-1} + \epsilon_{1,t}$$
$$Y_{2,t} = \alpha_2 + \beta_{21,1} Y_{1,t-1} + \beta_{22,1} Y_{2,t-1} + \epsilon_{2,t}$$

- Where, Y{1,t-1} and Y{2,t-1} are the first lag of time series Y1 and Y2 respectively.
- The above equation is referred to as a VAR(1) model, because, each equation is of order 1, that is, it contains up to one lag of each of the predictors (Y1 and Y2).

Vector Autoregression - (VAR)

- Since the Y terms in the equations are interrelated, the Y's are considered as endogenous variables, rather than as exogenous predictors.

- Likewise, the second order VAR(2) model for two variables would include up to two lags for each variable (Y1 and Y2).

$$Y_{1,t} = \alpha_1 + \beta_{11,1} Y_{1,t-1} + \beta_{12,1} Y_{2,t-1} + \beta_{11,2} Y_{1,t-2} + \beta_{12,2} Y_{2,t-2} + \epsilon_{1,t}$$
$$Y_{2,t} = \alpha_2 + \beta_{21,1} Y_{1,t-1} + \beta_{22,1} Y_{2,t-1} + \beta_{21,2} Y_{1,t-2} + \beta_{22,2} Y_{2,t-2} + \epsilon_{2,t}$$

Vector Autoregression - (VAR)

- For an example how an second order VAR(2) model with three variables (Y1, Y2 and Y3) would be framed.

$$
\begin{aligned}
Y_{1,t} &= \alpha_1 + \beta_{11,1} Y_{1,t-1} + \beta_{12,1} Y_{2,t-1} + \beta_{13,1} Y_{3,t-1} + \beta_{11,2} Y_{1,t-2} + \beta_{12,2} Y_{2,t-2} + \beta_{13,2} Y_{3,t-2} + \epsilon_{1,t} \\
Y_{2,t} &= \alpha_2 + \beta_{21,1} Y_{1,t-1} + \beta_{22,1} Y_{2,t-1} + \beta_{23,1} Y_{3,t-1} + \beta_{21,2} Y_{1,t-2} + \beta_{22,2} Y_{2,t-2} + \beta_{23,2} Y_{3,t-2} + \epsilon_{2,t} \\
Y_{3,t} &= \alpha_3 + \beta_{31,1} Y_{1,t-1} + \beta_{32,1} Y_{2,t-1} + \beta_{33,1} Y_{3,t-1} + \beta_{31,2} Y_{1,t-2} + \beta_{32,2} Y_{2,t-2} + \beta_{33,2} Y_{3,t-2} + \epsilon_{3,t}
\end{aligned}
$$

- As you increase the number of time series (variables) in the model the system of equations become larger.

Vector Autoregression - (VAR)

- As per the business requirement we are segregating the two specific stores from dataframe to perform analysis.

```
df = pd.DataFrame(columns=['Store_1044','Store_1041'])
df
```

Store_1044	Store_1041

```
df['Store_1044'] = Store50_1044['Customers']
df['Store_1041'] = Store50_1041['Customers']
df
```

Date	Store_1044	Store_1041
2014-01-01	0	0
2014-01-02	473	742
2014-01-03	483	864
2014-01-04	0	0
2014-01-05	0	0

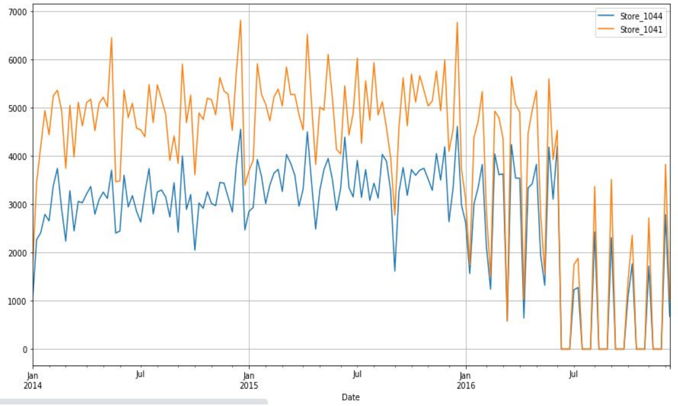
Vector Autoregression - (VAR)

- Currently, our data was split day-wise,for an better visualisation, interpreting the inference of data we have been re-sampled to week-wise how do the customers footfall weekly .

- We can observe the footfall of customers are Higher in store_1041 than store_1044.

- We can infer ,the data was stationary.

```
[ ]   3 df.resample('W').sum().plot(grid=True);
```



Vector Autoregression - (VAR)

- Let's perform Dickey-Fuller test to evaluate the stationarity of data.

- We can infer from the code output that the series Of data is stationary.

- Hence,no need of differentiation.

```
[ ]  1 adf_test(train['Store_1044'])

Results of Dickey-Fuller Test:
Test Statistic                -6.690539e+00
p-value                        6.994789e-08
#Lags Used                     2.000000e+01
Number of Observations Used    7.090000e+02
Critical Value (1%)           -3.971596e+00
Critical Value (5%)           -3.416700e+00
Critical Value (10%)          -3.130705e+00
dtype: float64
```

```
[ ]  1 adf_test(train['Store_1041'])

Results of Dickey-Fuller Test:
Test Statistic                 -6.147834
p-value                         0.000001
#Lags Used                     20.000000
Number of Observations Used   709.000000
Critical Value (1%)            -3.971596
Critical Value (5%)            -3.416700
Critical Value (10%)           -3.130705
dtype: float64
```

Best parameters are selected in accordance with the lowest Akaike Information Criteria (AIC)

- We can interpret as we are running this iteration upto 7 days since this is a day-wise data and we have seen that the customer footfall has a seasonality of 7.

```
for i in range(1,8):
    model = sm.tsa.VARMAX(train,order=(i,0),trend='c')
    model_result = model.fit()
    print('Order =',i)
    print('AIC:',model_result.aic)
```

```
Order = 1
AIC: 19230.829273497577
Order = 2
AIC: 19181.92084035652
Order = 3
AIC: 19184.99439129791
Order = 4
AIC: 19149.883695409582
Order = 5
AIC: 19143.76988617819
Order = 6
AIC: 19118.14673121335
Order = 7
AIC: 19051.728475029624
```

Model Evaluation

Model Evaluation

```
[ ]  model = sm.tsa.VARMAX(train,order=(7,0),trend='c')
     model_result = model.fit()
     model_result.summary()
```

Covariance Type: opg

Ljung-Box (L1) (Q):	0.00, 0.87	**Jarque-Bera (JB):**	65.91, 1689.39
Prob(Q):	0.99, 0.35	**Prob(JB):**	0.00, 0.00
Heteroskedasticity (H):	1.33, 1.65	**Skew:**	-0.70, -1.07
Prob(H) (two-sided):	0.03, 0.00	**Kurtosis:**	3.44, 10.14

Results for equation Store_1044

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	465.1398	50.587	9.195	0.000	365.991	564.288
L1.Store_1044	-0.4575	0.091	-5.022	0.000	-0.636	-0.279
L1.Store_1041	0.3380	0.061	5.522	0.000	0.218	0.458
L2.Store_1044	0.5199	0.085	6.121	0.000	0.353	0.686
L2.Store_1041	-0.4059	0.056	-7.195	0.000	-0.516	-0.295
L3.Store_1044	-0.1108	0.085	-1.299	0.194	-0.278	0.056
L3.Store_1041	0.1010	0.058	1.731	0.083	-0.013	0.215
L4.Store_1044	0.2254	0.103	2.180	0.029	0.023	0.428
L4.Store_1041	-0.1696	0.070	-2.423	0.015	-0.307	-0.032
L5.Store_1044	-0.1766	0.096	-1.846	0.065	-0.364	0.011
L5.Store_1041	0.0620	0.064	0.973	0.331	-0.063	0.187
L6.Store_1044	-0.1004	0.086	-1.173	0.241	-0.268	0.067
L6.Store_1041	0.0008	0.057	0.015	0.988	-0.111	0.112
L7.Store_1044	0.2598	0.112	2.313	0.021	0.040	0.480
L7.Store_1041	-0.0333	0.075	-0.445	0.656	-0.180	0.113

[ ]

Results for equation Store_1041

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	758.3079	79.917	9.489	0.000	601.673	914.942
L1.Store_1044	-0.8403	0.138	-6.103	0.000	-1.110	-0.570
L1.Store_1041	0.5908	0.090	6.540	0.000	0.414	0.768
L2.Store_1044	0.6228	0.136	4.595	0.000	0.357	0.889
L2.Store_1041	-0.5777	0.090	-6.396	0.000	-0.755	-0.401
L3.Store_1044	-0.2684	0.139	-1.926	0.054	-0.541	0.005
L3.Store_1041	0.2115	0.094	2.244	0.025	0.027	0.396
L4.Store_1044	0.4055	0.156	2.599	0.009	0.100	0.711
L4.Store_1041	-0.3743	0.106	-3.521	0.000	-0.583	-0.166
L5.Store_1044	-0.3818	0.136	-2.814	0.005	-0.648	-0.116
L5.Store_1041	0.1913	0.091	2.097	0.036	0.013	0.370
L6.Store_1044	-0.1092	0.137	-0.797	0.426	-0.378	0.160
L6.Store_1041	0.0242	0.091	0.266	0.790	-0.154	0.202
L7.Store_1044	0.0666	0.161	0.414	0.679	-0.249	0.382
L7.Store_1041	0.1838	0.107	1.713	0.087	-0.026	0.394

Error covariance matrix

	coef	std err	z	P>|z|	[0.025	0.975]
sqrt.var.Store_1044	211.2900	6.567	32.172	0.000	198.418	224.162
sqrt.cov.Store_1044.Store_1041	295.8125	13.848	21.361	0.000	268.671	322.954
sqrt.var.Store_1041	124.6403	3.020	41.269	0.000	118.721	130.560

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).

Model Evaluation

- Hence, our series of data is stationary we are able to secure the optimal RMSE.

- We can still tune the model for much better evaluation metric scores.

```
## Calculating the RMSE for Store 1044

from sklearn.metrics import mean_squared_error
rmse = mean_squared_error(test['Store_1044'],pred['Store_1044'],squared=False)
print('Store_1044:',rmse)
```

```
Store_1044: 270.29210953106235
```

```
## Calculating the RMSE for Store 1041

rmse = mean_squared_error(test['Store_1041'],pred['Store_1041'],squared=False)
print('Store_1041:',rmse)
```

```
Store_1041: 373.43176967674265
```

Summary

- Vector Autoregression (VAR) model is an extension of univariate autoregression model to multivariate time series data.
- VAR model is a multi-equation system where all the variables are treated as endogenous (dependent)
- There is one equation for each variable as dependent variable. In its reduced form, the right-hand side of each equation includes lagged values of all dependent variables in the system, no contemporaneous variables.
- The method is suitable for multivariate time series without trend and seasonal components.

VARMA

Business problem: The wall street journal want to publish an article on Stock predictions after the presidential elections held in US for the year 2016, how it impacts the stock indices.

- S&P Index is stock market index that measures the stock performance of 30 large companies listed on stock exchanges in the United States. Although it is one of the most commonly followed equity indices.

- This model forecast can be used to predict the stock price indexes by considering the past one year data of 2016 and predict one month of stock price index for the year 2017.

Visiting Basics

Vector Autoregression Moving Average - (VARMA)

- The Vector Autoregression Moving-Average (VARMA) method models the next step in each time series using an ARMA model. It is the generalization of ARMA to multiple parallel time series, e.g. multivariate time series.
- The notation for the model involves specifying the order for the AR(p) and MA(q) models as parameters to a VARMA function, e.g. VARMA(p, q). A VARMA model can also be used to develop VAR or VMA models.
- The method is suitable for multivariate time series without trend and seasonal components.

Vector Autoregression Moving Average - (VARMA)

- The VARMA model is another extension of the ARMA model for a multivariate time-series model that contains a vector autoregressive (VAR) component, as well as the vector moving average (VMA).
- VARMA is an inductive version of ARMA for multiple parallel time series. The ARMA notation for the model comprises the individually ordered AR(p), and the MA(q) models are the parameters to a VARMA model.
- For instance, VARMA(p, q) is an example. A VARMA model will be able to develop VAR or VMA models.

Vector Autoregression Moving Average - (VARMA)

Here are the VAR (1) and VMA(1) models with two time-series datasets (Y1 and Y2):

$$\hat{Y}_{1,t} = \mu_1 + \phi_{11} Y_{1,t-1} + d_{1,0} u_{1,t} + d_{11} u_{1,t-1}$$

$$\hat{Y}_{2,t} = \mu_2 + \phi_{21} Y_{2,t-1} + d_{2,0} u_{1,t} + d_{22} u_{2,t-1}$$

where y1,t-1, y2,t-1 are the first lag of time series Y1 and Y2.

The VARMA model follows the same rules for the design model similar to the univariate time series. It is utilizing the corresponding evaluation matrices such as AIC, BIC, FPE, and HQIC.

Vector Autoregression Moving Average - (VARMA)

- As per the business requirement we are segregating and considering the respective records to build a model and forecast accordingly.

```
df = df[(df['Date'] > '2016-01-14') & (df['Date'] <= '2017-01-30')]
```

```
df.head(10)
```

	Date	Open	High	Low	Close	Adj Close	Volume
1	2016-01-15	16354.330078	16354.330078	15842.110352	15988.080078	15988.080078	239210000
2	2016-01-19	16009.450195	16171.959961	15900.250000	16016.019531	16016.019531	144360000
3	2016-01-20	15989.450195	15989.450195	15450.559570	15766.740234	15766.740234	191870000
4	2016-01-21	15768.870117	16038.589844	15704.660156	15882.679688	15882.679688	145140000
5	2016-01-22	15921.099609	16136.790039	15921.099609	16093.509766	16093.509766	145850000

Vector Autoregression Moving Average - (VARMA)

- Checking for stationarity of data
- We can infer from the given output that the attributes : 'Open','High' are Non- stationary.

```
[ ] for name, column in df[['Open', 'High', 'Low', 'Close']].iteritems():
        Augmented_Dickey_Fuller_Test_func(df[name],name)
        print('\n')
```

```
Results of Dickey-Fuller Test for column: Open
Test Statistic                  -0.776223
p-value                          0.826007
No Lags Used                     0.000000
Number of Observations Used    261.000000
Critical Value (1%)             -3.455656
Critical Value (5%)             -2.872678
Critical Value (10%)            -2.572705
dtype: float64
Conclusion:====>
Fail to reject the null hypothesis
Data is non-stationary


Results of Dickey-Fuller Test for column: High
Test Statistic                  -1.240162
p-value                          0.656085
No Lags Used                     2.000000
Number of Observations Used    259.000000
Critical Value (1%)             -3.455853
Critical Value (5%)             -2.872765
Critical Value (10%)            -2.572752
dtype: float64
Conclusion:====>
Fail to reject the null hypothesis
Data is non-stationary
```

Vector Autoregression Moving Average - (VARMA)

- Checking for stationarity of data

- We can infer from the given output that the attributes : 'Low','Close' are Non- stationary.

```
Results of Dickey-Fuller Test for column: Low
Test Statistic                  -0.981046
p-value                          0.760114
No Lags Used                    13.000000
Number of Observations Used    248.000000
Critical Value (1%)             -3.456996
Critical Value (5%)             -2.873266
Critical Value (10%)            -2.573019
dtype: float64
Conclusion:====>
Fail to reject the null hypothesis
Data is non-stationary

Results of Dickey-Fuller Test for column: Close
Test Statistic                  -1.265244
p-value                          0.644919
No Lags Used                     0.000000
Number of Observations Used    261.000000
Critical Value (1%)             -3.455656
Critical Value (5%)             -2.872678
Critical Value (10%)            -2.572705
dtype: float64
Conclusion:====>
Fail to reject the null hypothesis
Data is non-stationary
```

Vector Autoregression Moving Average - (VARMA)

- Splitting the data into train and test.
- We would be considering the train data that consists of all the data except the last 30 days, and the test data which consists of only the last 30 days to evaluate on future forecasting.

```
X = df[['Open', 'High', 'Low', 'Close' ]]
train, test = X[0:-30], X[-30:]
```

Vector Autoregression Moving Average - (VARMA)

- We are performing differencing on data by using pandas to stationarize.

```
[ ]    train_diff = train.diff()
       train_diff.dropna(inplace = True)
```

Vector Autoregression Moving Average - (VARMA)

- Checking for stationarity of data after the First differencing.

- We can infer from the given output that the attributes : 'Open','High' are Stationarised now.

```
[ ]  for name, column in train_diff[['Open', 'High', 'Low', 'Close' ]].iteritems():
         Augmented_Dickey_Fuller_Test_func(train_diff[name],name)
         print('\n')
```

```
Results of Dickey-Fuller Test for column: Open
Test Statistic                 -1.579687e+01
p-value                         1.085613e-28
No Lags Used                    0.000000e+00
Number of Observations Used     2.300000e+02
Critical Value (1%)            -3.459106e+00
Critical Value (5%)            -2.874190e+00
Critical Value (10%)           -2.573512e+00
dtype: float64
Conclusion:====>
Reject the null hypothesis
Data is stationary


Results of Dickey-Fuller Test for column: High
Test Statistic                 -1.172782e+01
p-value                         1.364178e-21
No Lags Used                    1.000000e+00
Number of Observations Used     2.290000e+02
Critical Value (1%)            -3.459233e+00
Critical Value (5%)            -2.874245e+00
Critical Value (10%)           -2.573541e+00
dtype: float64
Conclusion:====>
Reject the null hypothesis
Data is stationary
```

Vector Autoregression Moving Average - (VARMA)

- Cointegration is used to check for the Existence of a long-run relationship between two or more variables.
- However, the correlation does not necessarily mean “long run.”
- We can infer from output that there is the presence of a long-run relationship between features,hence because the Significance is ‘True’.

```
from statsmodels.tsa.vector_ar.vecm import coint_johansen

def cointegration_test(df):
    res = coint_johansen(df,-1,5)
    d = {'0.90':0, '0.95':1, '0.99':2}
    traces = res.lr1
    cvts = res.cvt[:, d[str(1-0.05)]]
    def adjust(val, length= 6):
        return str(val).ljust(length)
    print('Column Name   >  Test Stat > C(95%)    =>   Signif  \n', '--'*20)
    for col, trace, cvt in zip(df.columns, traces, cvts):
        print(adjust(col), '> ', adjust(round(trace,2), 9),
              ">", adjust(cvt, 8), ' =>  ' , trace > cvt)

```

```
cointegration_test(train_diff[['Open', 'High', 'Low', 'Close']])
```

```
Column Name   >  Test Stat > C(95%)    =>   Signif
 ----------------------------------------
Open    >  311.57    > 40.1749   =>   True
High    >  201.62    > 24.2761   =>   True
Low     >  102.52    > 12.3212   =>   True
Close   >  32.21     > 4.1296    =>   True
```

Vector Autoregression Moving Average - (VARMA)

- We are performed the analysis to obtain an optimal ‘p’,’q’ which possess an minimal RMSE .
- We can infer from the output that order(‘p’,’q’)(0,2) has an optimal/least RMSE.

```
df_results.sort_values(by = ['RMSE Open','RMSE High','RMSE Low','RMSE Close'] )
```

	p	q	RMSE Open	RMSE High	RMSE Low	RMSE Close
0	0.0	2.0	263.292055	190.769946	286.703322	319.011671
1	0.0	2.0	263.292055	190.769946	286.703322	319.011671
2	0.0	1.0	314.116822	209.738485	336.623834	345.694531

Model Evaluation

Model Evaluation

- We have fitted the time series data with VARMA model.
- We can infer from the input code that we have considered the order('p','q')(0,2) hence,because its providing an optimal/lease. RMSE

```
[ ]
# from above example we can see that p=0 and q=2 gives least RMSE
model = VARMAX(train_diff[[ 'Open', 'High', 'Low', 'Close' ]], order=(0,2)).fit( disp=False)
result = model.forecast(steps = 30)
```

Model Evaluation

- Evaluate the results for every Individual attribute .

- We can find the various evaluation Metric's with score's from the Given output for the attributes 'Open','High'.

```python
for i in ['Open', 'High', 'Low', 'Close' ]:
    print(f'Evaluation metric for {i}')
    timeseries_evaluation_metrics_func(test[str(i)] , res[str(i)+'_1st_inv_diff'])
```

```
Evaluation metric for Open
Evaluation metric results:-
MSE is : 69322.70648306681
MAE is : 224.24223243977565
RMSE is : 263.292055487944
MAPE is : 1.1277702479939442
R2 is : -10.34006693369997

Evaluation metric for High
Evaluation metric results:-
MSE is : 36393.17247139601
MAE is : 152.9002013818139
RMSE is : 190.7699464574963
MAPE is : 0.7673581832223293
R2 is : -5.539038158041874
```

Model Evaluation

- Evaluate the results for every Individual attribute .

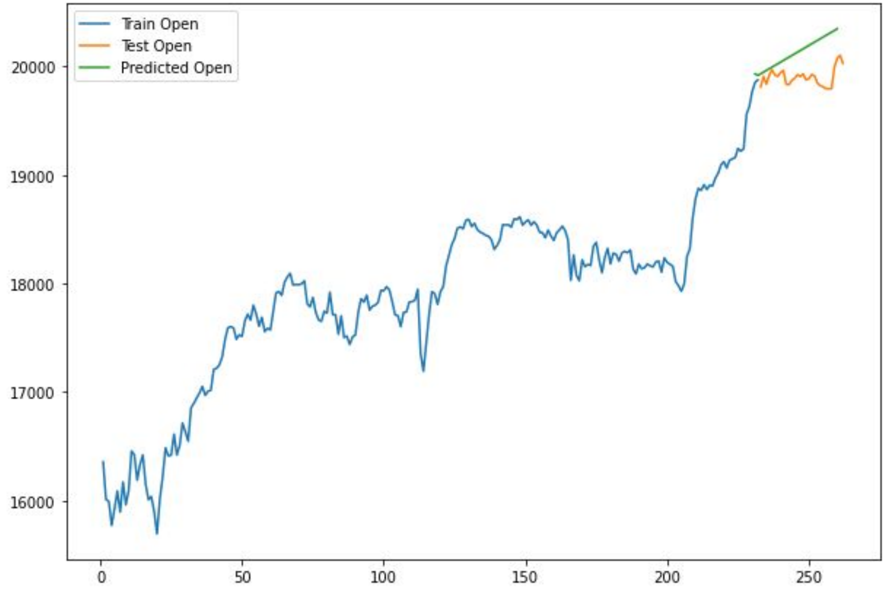
- We can find the various evaluation Metric's with score's from the Given output for the attributes 'Low', 'Close'.

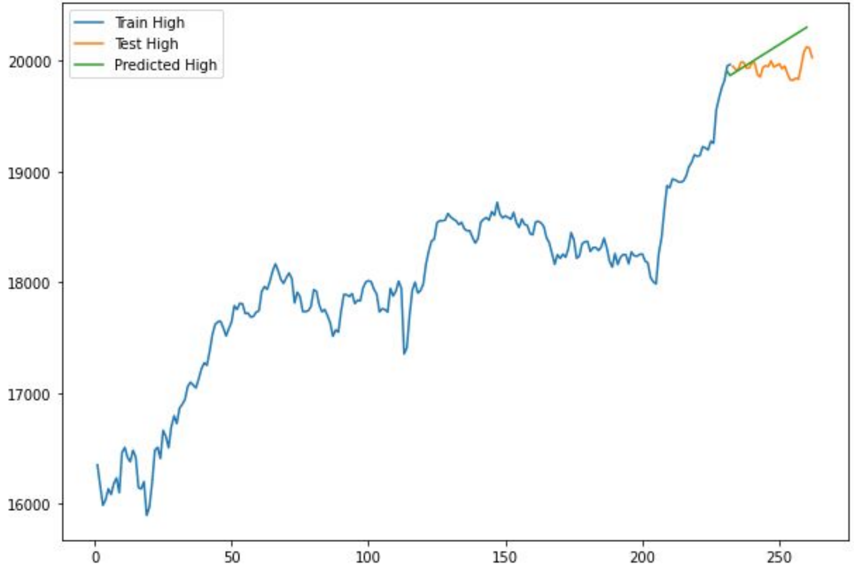
```
Evaluation metric for Low
Evaluation metric results:-
MSE is : 82198.79470416254
MAE is : 238.7760429063243
RMSE is : 286.70332175292725
MAPE is : 1.205080775441637
R2 is : -8.625550649982372

Evaluation metric for Close
Evaluation metric results:-
MSE is : 101768.44638431296
MAE is : 282.75981653930563
RMSE is : 319.011671235259
MAPE is : 1.4221000273357538
R2 is : -12.43348780600549
```

Model Evaluation

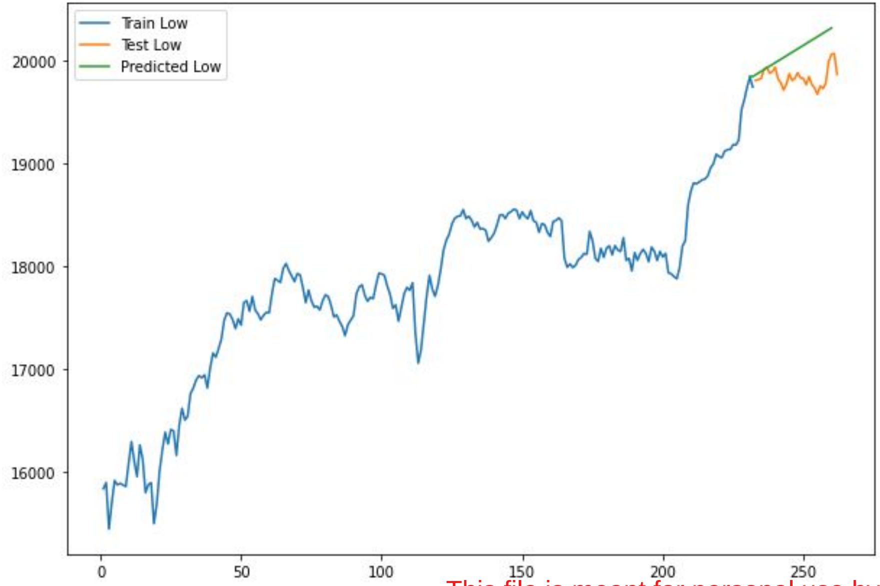
- We can infer the results for the attributes ‘Open’,’High’ from the below visualisation plot.

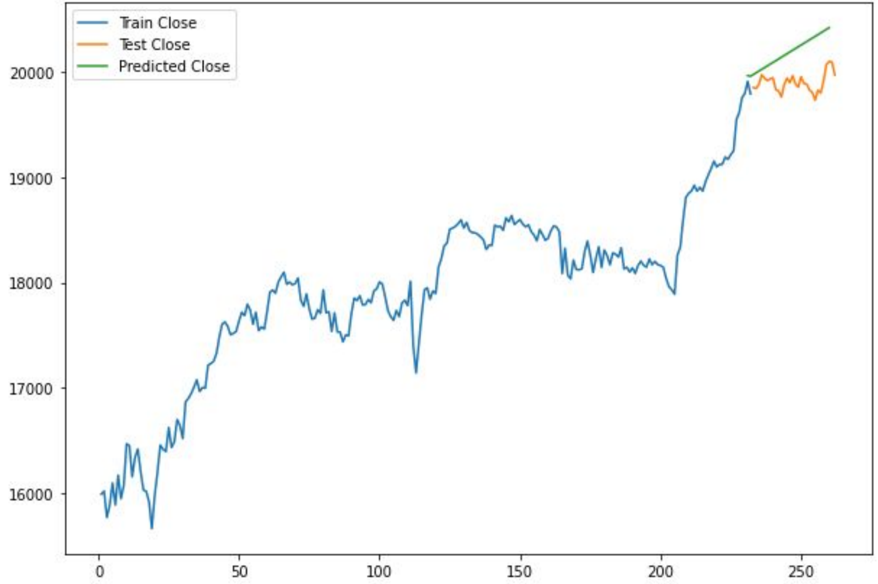




Model Evaluation

- We can infer the results for the attributes ‘Low’,’close’ from the below visualisation plot.





Reference

- Vector Autoregressive Models for Multivariate Time Series. http://faculty.washington.edu/ezivot/econ584/notes/varModels.pdf
- Dwyer, Gerald P., Jr. Why Are Vector Autoregressions Useful in Finance? http://jerrydwyer.com/pdf/lectvar.pdf
- Vector Autoregressions: Forecasting and Reality.http://www.frbatlanta.org/filelegacydocs/robtallman.pdf