Multiple Linear Regression and Time Series Analyses

Trinita John Peter
MSC in Data Analytics
Dublin, Ireland
x23266848@student.ncirl.ie

***Abstract* – This project deals with 2 important statistical analysis namely, Multiple Linear Regression and Time Series. Components taken into consideration for performing MLR Analysis are assumptions, interpreting results, presenting plots, detecting outliers, applying scaling/polynomial features for better analyses and building regression models based on those values.**

Time Series Analysis follows a different approach. Decomposition techniques are applied to observe trends and patterns, stationarity of the data is checked using rolling statistics and AD Fuller methods. Autocorrelation is identified between the time series data points. Advanced model fitting like ARIMA and simple method like Naïve modelling are used to forecast values and finally the moving average is calculated using log time series.

I. INTRODUCTION- MULTIPLE LINEAR REGRESSION

Provided with a dataset that contains 4 columns and 1000 rows where 'y' is the dependent/target variable and 'x1','x2','x3' are the independent/feature variables. Of these features, 'x3' is categorical and the rest corresponds to numerical values. The dataset is modelled using the statistical technique Multiple Linear Regression as there are more than one feature affecting the target or output variable. The objective of this analysis is to improve model efficiency by applying regression techniques that effectively reduce the errors in training, testing data set between actual and target variables as a result of accurate prediction. [1]

II. EXPLORATORY DATA ANALYSIS

Since the feature variable 'x3' is of categorical type, converting its values into numerical type is the primary process which can help the model train and perform better, ensuring all variables fall under one data type structure. This is achieved by encoding non-numerical values into numerical by the 'cat.codes' property. Secondly, checking the dataset for null values plays a huge role before proceeding with further analysis. The 'isnull()' function detects missing values which is 0 in this case. [2]

The describe() method shows mean, median, mode values in Fig MLR 1.

	y	x1	x2
count	1000.000000	1000.000000	1000.000000
mean	15454.300514	49.875794	200.126134
std	1861.868348	4.153451	3.198992
min	9183.651604	33.770110	187.626645
25%	14214.832900	46.954300	198.536816
50%	15446.202330	49.870760	200.110545
75%	16698.076227	52.518992	201.712418
max	21400.010230	61.825863	215.556281

Fig MLR 1

To visualize the relationship between all the variables, a pair plot is created from seaborn package. In order to adjust the size of plot, 'height' and 'aspect' parameters are used. [3]

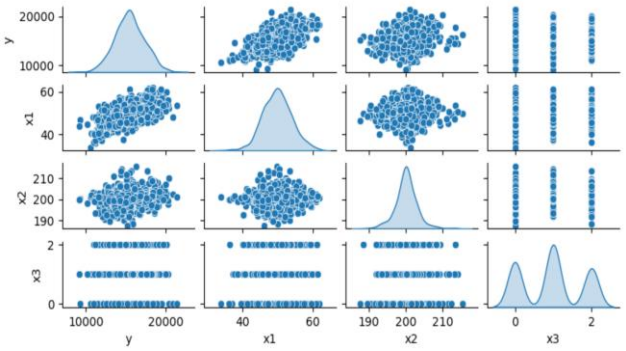

Fig MLR 2

A. *Insights from Fig MLR 2*

1. *y and x1: there appears to be positive correlation between y and x1 as values of y increase when x1 increases, suggesting that x1 maybe a useful predictor of y.*
2. *y and x2: no strong correlation between y and x2 as the plot is more scattered/randomly distributed.*
3. *y and x3: x3 is categorical, clusters in the plot do not have much clear trend or correlation with y.*
4. *x1 and x2: no clear relationship demonstrated because of random spread*
5. *x1 and x3, x2 and x3: no clear trend shown as it is all*

distinct.

III. DATA PREPARATION

- Firstly, the dataset is split into test labels and training features. X contains data of feature variables after dropping column 'y' using the 'drop()' method. Y variable contains only the column extracted from 'y'.
- For the split to be reproduced, 'random_seed' is set to '23266848'
- The 'test_train_split' function is applied on X and y. 'test_size' is set to 0.2 which will consider 80% as training data and 20% as testing data.

a. Detecting outliers

- Identifying anomalies using z-score is a crucial process especially if the values are above threshold. The value of limit is set in the code to 3, making the values falling outside as potential outliers.
- But in this case, eliminating the outliers is not being focused on since no significant number of 'TRUE' is shown in Fig MLR 3. [4]

```
Z score outliers on the data :
         y     x1     x2     x3
0    False  False  False  False
1    False  False  False  False
2    False  False  False  False
3    False  False  False  False
4    False  False  False  False
```

Fig MLR 3

- Box plots are a useful library as well to visualize the mean, median, mode and quartiles highlighting the outlier data.
- Below Fig MLR 4, shows outlier details heavily distributed for y and not much shown in x1,x2 and x3.

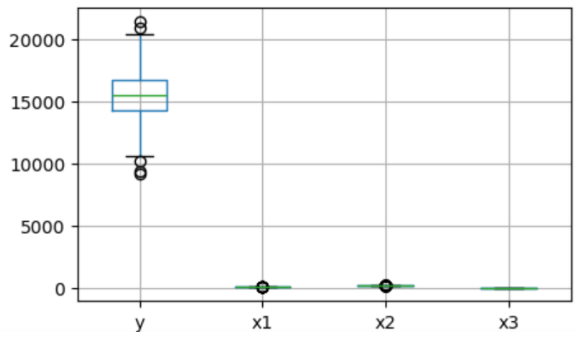

Fig MLR 4

b. Applying Linear Regression

- A simple linear regression model is used from the library to train the X training and y training dataset.
- Based on the model training, prediction is made on X testing data. The results are stored in 'y_pred' variable.

c. Error metrics on LR

- Mean squared error and r squared errors are computed for 'y_test' and 'y_pred'.
- The results are 2308816.6 and 0.29 for each metric respectively, showing that error can be reduced and only 29% of variance in the data has been explained by the model. [5]

d. Improved model by Feature Engineering

I. Standardizing the features:

This technique scales all feature variables such that the values have mean 0 and std deviation 1. The 'fit_transform()' method calculates mean and std deviation then scales the X training data accordingly. 'transform' method is then applied on the X test data to ensure consistency throughout the dataset.

II. Introducing Polynomial features:

Polynomial features help in capturing the non-linear relationships between feature and target variables. Parameters like 'degree=2' and 'include_bias=False' are included indicating that quadratic features along with original features will be created and a column of ones is not added.

The input passed to the polynomial feature object is the scaled training dataset from previous standard scaling technique.

After applying the 'fit_transform' method to X scaled training dataset, 'transform()' is used on the X scaled testing set. Thus, the input for our model is processed. [6]

IV. DATA MODELLING

3 Multiple Regression Models are built.

i. LR

a. The Linear Regression model is initialized.
b. The input passed to Linear Regression this time is polynomial x train data with y train data. Hence the model fits into the training data.
c. After the model learns from the dataset using 'fit()' method, 'predict()' method is applied on the polynomial x test data.

ii. Ridge Model – L2 regularization

a. Ridge Model is initialized.
b. The input passed to Ridge model is polynomial x train data with y train data. Hence the model fits into the training data.
c. After the model learns from the dataset using 'fit()' method, 'predict()' method is applied on the polynomial x test data.

iii. Lasso Model – L1 regularization

a. Lasso model is initialized
b. The input passed to Ridge model is polynomial

x train data with y train data. Hence the model fits into the training data. [7]

c. After the model learns from the dataset using 'fit()' method, 'predict()' method is applied on the polynomial x test data.

V. INTERPRETATION OF Fig MLR 5

Evaluation of error metrics from the above 3 models:

i. A function called 'error_metrics' is defined that encapsulates RMSE and R2 score value for all of the 3 models built above.

```
LR - rmse : 1528.73719580066, r_square 0.28842340235144404
Ridge - rmse : 1528.4896590349272, r_square 0.288653824047759
Lasso - rmse : 1528.682914379327, r_square 0.2884739338665677
```

Fig MLR 5

ii. The above results show that error metrics of LR, Ridge and Lasso models do not have much variance in RMSE and R2 scores. Hence training in dataset using any of the models can be carried out.

VI. DIAGNOSTICS

Checking assumptions for OLS model with stats model using the 'add_const()' function. [8]

- Using the statsmodel to fit OLS model using polynomial features gives a detailed comprehensive summary of regression analysis, coefficients, p-values, confidence intervals and other various metrics.
- An intercept is added to the model if all the coefficient predictors turn out to be 0.
- The OLS model is then trained with polynomial features and the intercept.

OLS Regression Results

Dep. Variable:	y	R-squared:	0.427
Model:	OLS	Adj. R-squared:	0.420
Method:	Least Squares	F-statistic:	65.34
Date:	Fri, 15 Nov 2024	Prob (F-statistic):	1.98e-89
Time:	19:04:32	Log-Likelihood:	-6940.4
No. Observations:	800	AIC:	1.390e+04
Df Residuals:	790	BIC:	1.395e+04
Df Model:	9		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.543e+04	86.280	178.891	0.000	1.53e+04	1.56e+04
x1	1147.1463	50.738	22.609	0.000	1047.549	1246.744
x2	304.7813	51.777	5.886	0.000	203.145	406.418
x3	124.0726	50.950	2.435	0.015	24.060	224.085
x4	-50.8166	33.100	-1.535	0.125	-115.791	14.158
x5	139.4293	61.636	2.262	0.024	18.440	260.419
x6	55.3598	50.448	1.097	0.273	-43.668	154.387
x7	-19.7461	22.963	-0.860	0.390	-64.822	25.330
x8	80.2668	49.185	1.632	0.103	-16.283	176.816
x9	109.4501	58.248	1.879	0.061	-4.889	223.789

Fig MLR 6

KEY METRICS FROM Fig MLR 6:

1. R-squared – indicates that 42% of variance is explained by independent variables.
2. Adjusted R-squared – 0.420 indicates that some predictors may not be significantly contributing to the model.
3. F-statistic – 65.34 suggests that the model is statistically significant.
4. Prob(F-statistic) – 1.98e-89 is associated with low p value also indicating that the model is statistically significant.

Detailed Coefficient Results:

1. Coef 205 – intercept of the model
2. Std error 1255 – standard error of the intercept
3. T 0.164 – t value for the intercept
4. P 0.87 – which is not significant.

VII. RESIDUAL ANALYSIS

This analysis is carried out to evaluate the assumptions

- Residual is calculated by applying the difference between y train data and predicted OLS with polynomial constant as input.
- Scatter plot is then plotted for residual and predicted OLS polynomial feature.
- Here is the visualization between the 2 values indicating that the plot does not show a pattern rather all data points are scattered around the red line in Fig MLR 7. [9]

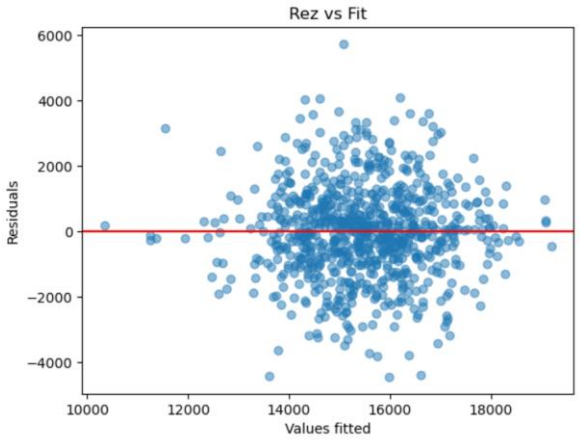

Fig MLR 7

VIII. Q-Q PLOT

The Q-Q Plot in Fig MLR 8, for residuals is generated as one of the key assumptions in OLS model is that residuals must be normally distributed. 45-degree reference line is added to compare the quantiles of sample to normal distribution.

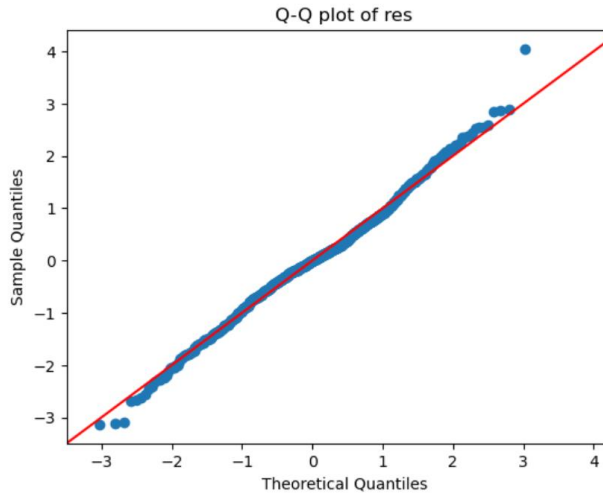

Fig MLR 8

- Most of the blue data points lie on the red line indicating that the residuals are normally distributed.
- The deviations in the tail in the lower left and upper right can be omitted. [10]

TIME SERIES

I. INTRODUCTION

Time Series Analysis can be done by plotting data points for different time ranges. Observations recorded at regular time intervals indicate the trend, seasonality or pattern the data points follow.

To execute a time series forecasting on any dataset, we can use a python library called 'Auto TS'. This library allows us to do time series forecasting without any of the pre/post processing techniques required for the data. This library can be used by loading the dataset in the library to get accurate predictions in no time. [11]

II. EXPLORATORY DATA ANALYSIS

The dataset provided contains 2 columns – 'Unnamed: 0' and 'x'. The former column represents values of time in steps and the latter column represents values associated with the time series data itself. **Note** that this column 'x' has not been converted into datetime stamp that pandas library provides as it is best practice to do so only when the values are explicitly in date format. Primary step is to check for null values and fill them with central measure tendencies like mean, median and mode based on the data type, if any. Here is the result of the above check :

`time_value  0`

The size of data can be checked by applying the shape() method on the dataset, and here is the result:

`size of dataset : (301, 1)`

For better analysis and understanding, the column names are renamed and time step is set as index as shown in Fig TS 1.

Why even analyze a time series? Because it is the preparatory step before we develop a forecast of the series.

	time_value
1	132.639999
2	132.690002
3	134.380005
4	134.410004
5	135.300003

Fig TS 1

Before delving into complex analysis, here is a simple plot titled 'Basic TS viz' with 'time_id' in the x-axis and 'time_value' in y-axis. 'matplotlib' library helps understand our dataset better yet in a simple way.

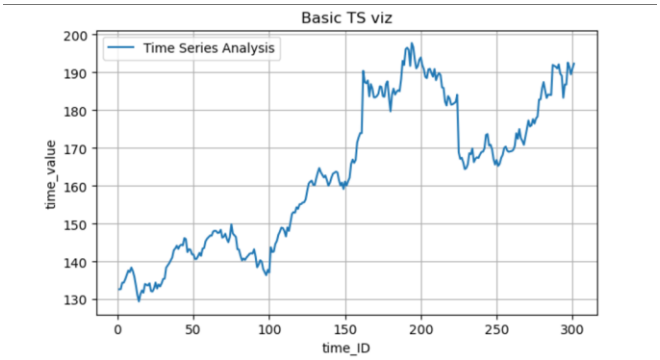

Fig TS 2

A. *Insights from Fig TS 2*:

1.The plot indicates that there exists an upward trend in which the variable 'time_value' is dependent on 'time_ID' showing increasing pattern

2.Variation is exhibited in the values over time, demonstrating that there could be seasonal factors influencing our data time points.

3.This plot shows no indication of strong periodicity, meaning no seasonal pattern displayed.

4.Wide range of data points fall between the value 130 and 190 on observation.

III. DECOMPOSITION OF TIME SERIES ANALYSIS

Important components in the Time Series Analysis include checking trend, seasonality, stationarity of data points, identifying auto-correlation, advanced model fitting and forecasting.

'Additive' model is taken into account as it is a given that there is an increasing trend captured in the plot. While Multiplicative model can be applied to exponentially increasing data points. [12]

In the below figure Fig TS 3, there are 3 decomposition components and graphs are plotted for each of them.

- Original Time Series plot -

  Quite like the plot above which has raw data points as values to capture variation in time series.

- Trend Component -

  Trends are usually captured on long patterns in data that indicate whether the series is increasing, decreasing or remaining constant over time.

- Seasonality Component -

  Seasonality is captured when checking if there are repeated short cycles within the observed data points. In other words, periodic occurrence exhibiting fluctuations.

- Residual Component -

  This represents noise or irregular component in the time series. This plot is captured only after removing the trend and seasonality components with the fluctuations.

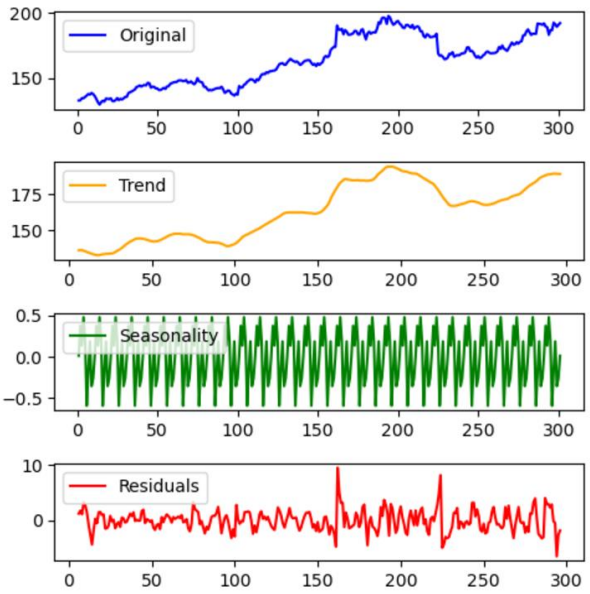

Fig TS 3

-> In summary, the decomposition method helps recognize the underlying patterns in time series for easy interpretation of data.

-> Analyzing trends and seasonality separately helps achieve accurate predictions and forecasts.

-> Anomaly Detection helps identify outliers or deviations in the data.

IV. STATIONARITY OF DATA

While checking stationarity of the data, mean should be constant. In this case, there is an increase of data points, hence indicating mean also increases. There are 2 tests that help in checking the stationarity of the data: rolling statistics and Augmented Dickey Fuller method.

i. We define 2 variables, `roll_mean` and `roll_std` with window period on yearly basis to calculate the mean and standard deviation.

ii. Plots against the rolling statistics value – mean, standard deviation and original time series data points is shown in Fig TS 4. [13]

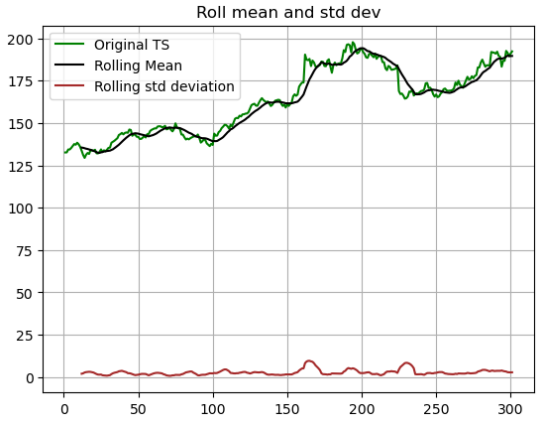

Fig TS 4

A. Components of graph:

- Original Time Series – green line, represents raw values over range of 0-300.
- Rolling Mean – black line, is a smoothed version of the data set containing time series values that is computed by considering the average of specific window period. Indicates increase in data points.
- Rolling standard deviation – red line, measures the variability within data points. Random spike shows unstable or inconsistency in time series.

V. ADF TEST

```
Results of DF test
Test Statistic                  -1.005504
p-value                          0.751251
#lags used                       0.000000
No. of obs                     300.000000
Critical value (1%)             -3.452337
Critical value (5%)             -2.871223
Critical value (10%)            -2.571929
dtype: float64
```

Fig TS 5

Results of Dickey-Fuller Test is shown in Fig TS 5:

- Augmented Dickey-Fuller Test comes handy to check for presence of unit root in a time series stationarity.
- P-value of 0.7 is comparatively higher than the thresholds for 1%, 5% and 10%.
- Since the test statistic which is -1.005 is higher than all critical values listed and p-value is greater than 0.05, we reject that the null hypothesis theorem.
- Implies that the data is non-stationary Time Series plot.

To make the time series data stationary, we apply the differencing method.

- ADF test is then carried out on the differenced data. The results are as follows:

`ADF Statistic after differencing: 18.291707630192576
p-value after differencing: 2.296587604957597e-30`.

In summary, the null hypothesis can be rejected with high confidence. [13]

VI. PLOTTING ACF AND PACF:

- The autocorrelation technique suggests that if the if the values fall within the blue shaded area as shown in Fig TS 6, they are not statistically significant. [13]

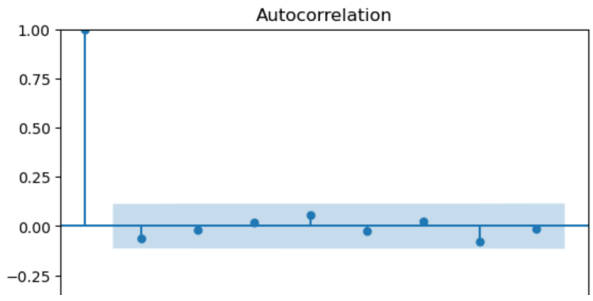

Fig TS 6

VII. ARIMA MODEL AND FORECASTING

SARIMAX Results

Dep. Variable:	time_value	**No. Observations:**	301
Model:	ARIMA(1, 1, 1)	**Log Likelihood**	-665.177
Date:	Fri, 22 Nov 2024	**AIC**	1336.355
Time:	11:50:13	**BIC**	1347.466
Sample:	0	**HQIC**	1340.802
	- 301		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	0.0704	1.488	0.047	0.962	-2.846	2.986
ma.L1	-0.1228	1.476	-0.083	0.934	-3.015	2.769
sigma2	4.9361	0.142	34.663	0.000	4.657	5.215

Ljung-Box (L1) (Q):	0.02	**Jarque-Bera (JB):**	3467.62
Prob(Q):	0.88	**Prob(JB):**	0.00
Heteroskedasticity (H):	3.02	**Skew:**	0.43
Prob(H) (two-sided):	0.00	**Kurtosis:**	19.63

Fig TS 7

A. INTERPRETATION of Fig TS 8:

Significance and coefficients - ar.L1 and ma.L2 possess high values indicating not statistically significant. Hence showing that average terms do not contribute significantly to variance.

Sigma2 – variance of residuals is highly significant.

Normality and Variability – high skew and kurtosis indicate residuals are not normally distributed. [14]

B. FORECASTING:

Forecast Data indicates the values predicted by the ARIMA model stating that those values align with original data shown in Fig TS 8. [14]

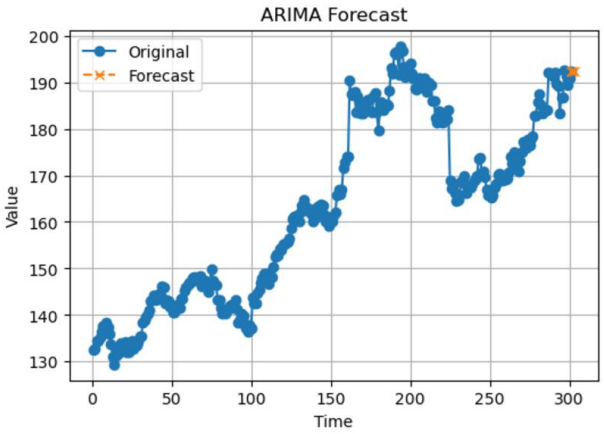

Fig TS 8

C. MOVING AVERAGE WITH LOG TIME SERIES

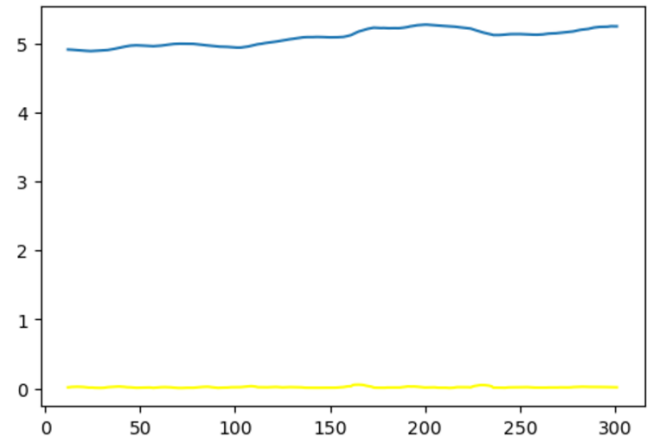

Fig TS 9

COMBINED ANALYSIS OF Fig TS 9:

Stability and consistency are exhibited on the above plot indicating that the data is stable and consistent.

VIII. NAÏVE MODEL

A. MODELLING TECHNIQUE

- This modelling technique is applied when the future value of a time series is similar or relative to the most recent observed value.
- It is a simpler model approach that does not require parameter tuning.
- This approach is only based on the last observed value and does not consider the trends or seasonality pattern.
- Below figure Fig TS 10, shows the block of code that creates a column in the dataset of values where the shift of 1 is applied on the old time series column. [15]

```
[17]: # applying naive modelling by shifting the values
      ts_file_naive['pred_values'] = ts_file_naive["time_x"].shift(1)
      ts_file_naive['pred_values']
```

```
[17]: 2            NaN
      3     132.690002
      4     134.380005
      5     134.410004
      6     135.300003
               ...
      297   186.800003
      298   192.610001
      299   191.449997
      300   189.479996
      301   190.990005
      Name: pred_values, Length: 300, dtype: float64
```

Fig TS 10

- The next block of code in Fig TS 10, removes the first value in the row as it results in NaN and checks for the sum of null values present in the new column,

```
[14]: # dropping null values:
      ts_file_naive.dropna(inplace=True)
      ts_file_naive.isnull().sum()
```

```
[14]: time_x         0
      pred_values    0
      dtype: int64
```

Fig TS 11

B. INTERPRETATION OF ERROR METRICS:

- Applying the mean squared error metric, on the predicted value from Naïve and actual values tells us about the error difference between the 2 values and well the model captures the variability in the dataset.

```
]: # calculating the error metrics
   from sklearn.metrics import mean_squared_error, r2_score
   naive_rmse = np.sqrt(mean_squared_error(ts_file_naive["time_x"], ts_file_naive["pred_values"]))
   print(f"naive rmse : {naive_rmse}")
```

```
naive rmse : 2.2248717978903314
```

```
]: # r2 score
   naive_r2 = r2_score(ts_file_naive["time_x"], ts_file_naive["pred_values"])
   print(f"naive r2 : {naive_r2}")
```

```
naive r2 : 0.9872511698764501
```

Fig TS 12

- The results shown in Fig TS 12 are RMSE : 2.2239 and R2 score : 0.988.
- The lower the RMSE value, the better - meaning since the time series values range between 200 and 500 the obtained RMSE value is relatively lower, implying less error between actual values and predicted ones and that the model is good in predicting future values.
- Higher R2 score indicates that the model is able to capture large variability in the data which is about 98%.

C. VISUALISATION:

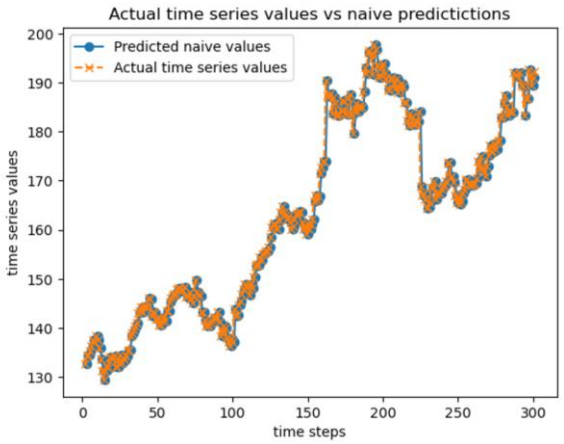

Fig TS 13

Fig TS 13 indicates close alignments between the 2 values, suggesting that naïve model's predictions are accurate in comparison with actual values.

IX. CONCLUSION:

In conclusion, the Naïve modelling approach produces best results and is evident from the error metrics that the prediction has been done well.

References

[1] [Online]. Available: Multiple Linear Regression | A Quick Guide (Examples).

[2] [Online]. Available: Multiple Linear Regression. A complete study—Model Interpretation… | by Sangeet Aggarwal | Towards Data Science.

[3] [Online]. Available: https://seaborn.pydata.org/generated/seaborn.pairplot.html.

[4] [Online]. Available: https://medium.com/@stevernewman/eliminating-outliers-in-python-with-z-scores-dd72ca5d4ead.

[5] [Online]. Available: Multiple Linear Regression in Machine Learning.

[6] [Online]. Available: https://serokell.io/blog/feature-engineering-for-machine-learning.

[7] [Online]. Available: https://www.statology.org/when-to-use-ridge-lasso-regression/.

[8] [Online]. Available: https://bookdown.org/ripberjt/qrmbook/ols-assumptions-and-simple-regression-diagnostics.html.

[9] [Online]. Available: https://www.statology.org/residuals/.

[10] [Online]. Available: https://www.geeksforgeeks.org/quantile-quantile-plots/.

[11] [Online]. Available: Time Series Analysis in Python - A Comprehensive Guide with Examples - ML+.

[12] [Online]. Available: https://machinelearningmastery.com/decompose-time-series-data-trend-seasonality/.

[13] [Online]. Available: https://towardsdatascience.com/detecting-stationarity-in-time-series-data-d29e0a21e638.

[14] [Online]. Available: Time-Series-Forecasting/Time Series Shampoo Sales Forecast.ipynb at master · vijendra-code/Time-Series-Forecasting.

[15] [Online]. Available: https://statisticseasily.com/glossario/what-is-naive-approach-understanding-its-basics/.