# <u>TIME SERIES FORECASTING WITH MACHINE LEARNING MODELS IN PYTHON</u>

## TIME SERIES

Time series analysis is an imperative technique used to detect changes that occur in the pattern of data over a period of time. It is technique used in diverse fields from health, manufacturing to finance.

In this research, emphasis is going to be placed on using time series data for forecasting than trend analysis and modeling.

Time series data is usually gathered at equally spaced intervals for analysis. There are many factors that could be analyzed from the data such as trends, seasonality, autocorrelation, patterns, among others.

Some of the most traditional methods used in time series analysis include exponential smoothing, Fourier analysis, etc.

It has a profound impact in binary classification, where it is used to identify outliers or anomalies which give indications of a fault in a pattern. One of its predominant uses in with binary classification is in the health sector with the use of ECG. It is also used avidly for weather forecasting.

It is significant to consider time series data by its context to make room to analyze the external factors that affect it.

However, in this study, we will scope in on time series in finance, particularly stock market forecasting.

# Time Series for Python

This study is going to be conducted with the Python programming language since it is considered one of the excellent tools for working with time series. Libraries like NumPy, pandas as well as Matplotlib make time series data easy to process and manipulate. Visualization is a key part of time series analysis and python provides great visualization tools for that matter.

## Fundamental of Time Series Analysis

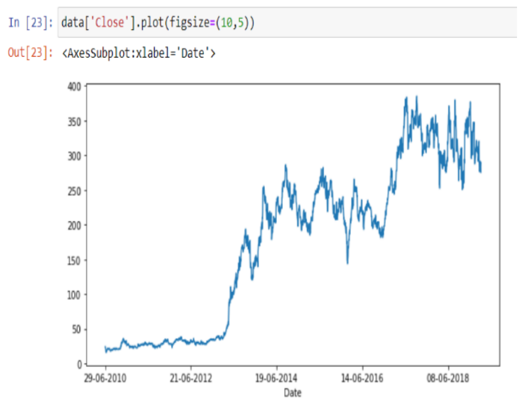
Understanding the core ideas of trend, seasonality, autocorrelation, and stationarity is essential for analyzing time series data and is addressed in this section.

Seasonality is the essence of repetitive patterns that show up in a time series data over a time interval; this interval could be daily, weekly, yearly and others. Seasonality is said to have a high proportional rate the accuracy of time series forecasting.

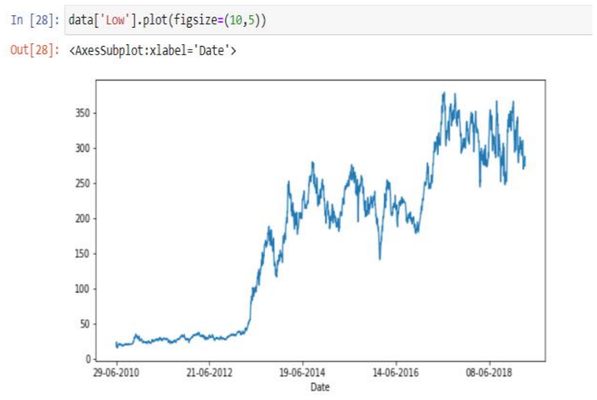
Autocorrelation is a measure of the relationship between the current value of a time series and its previous values. Autocorrelation has a high proportional rate with making prediction of a time series model easier.

A trend is an observation in the movement of time series data over an extended period of time. It could be linear or nonlinear, it could again be a downward or an upward trend.

As the images below show an upward trend in the Low and Closing price



```
In [23]: data['Close'].plot(figsize=(10,5))
Out[23]: <AxesSubplot:xlabel='Date'>
```



```
In [28]: data['Low'].plot(figsize=(10,5))
Out[28]: <AxesSubplot:xlabel='Date'>
```

Stationarity tells a story of the elements of time series data that stay relevant over time, where you notice a constancy and thus, shows no trends or patterns. Stationarity has a high proportionality with ambiguous predictions.

It is always prudent to convert a time series data to a stationary series by dropping trends and patterns before performing analysis just like we would normally do for other form of data where we use the normalization techniques.

## METRICS FOR STOCK PERFORMANCE

The main metrics used to measure the performance of a stock are Open, Low, High, Close, Adjusted Close.

To start with, Open does not add much value to the metrics since it just tells the opening price of a specific. In other words it tells the closing price of the previous time period (daily, monthly, etc.) in question.

Low is the lowest price at which a stock was traded during a particular period of time. It stands for the bottom of the range of prices that the stock traded at in the said period. Low prices can be a good opportunity to buy stocks, all things being equal(i.e. in the case where investors have a strong belief that the stock will rise, which is the ultimate rationale most of the time).\

High, on the contrary, is the highest price at which a stock was traded at a specific period. It represents the top of the range of prices that the stock traded at. Soaring prices can imply that the stock is in demand.

Close indicates the final price at which a stock traded at a period of time. In other words, this refers to the price at which the stock landed at when the market closed for the day.

The close price is vital because it is used to calculate important metrics such as daily returns, which are used to assess a stock's performance over time.

Volume refers to the total number of shares of a stock that was traded during a given period. This metric can give investors an idea of how actively a stock is being traded and how much interest there is in it. High volume can imply that investors are actively buying and selling the stock, whereas low volume may signify that there is less concentration in the stock.

## DATA

The link to the dataset could be found below:

https://github.com/joe-t16/ML/blob/main/tesla.csv

This dataset was retrieved from the yahoo finance page, a trusted source for stock data through GitHub.

The dataset of '.csv' is then converted to a data frame using the pandas library in python.

The original data frame had a shape of (2193, 7) which explains that there are 2193 datapoints and 7 features to deal with. These datapoints contain daily stock metrics and the features are basically the names given to these metrics.

Features: Date, Open, High, Low, Close, Adjusted Close, and Volume.

The dataset had just one object datatype out of the 7, which is the column “Date” and rest as numerical (int and float) datatypes.

There were no missing values.

The target variable, ‘Close’ had values of the daily closing price of the Tesla stock.

When it comes to time series data, the aim is not to drop missing values but rather find a good technique to work around them to get great predictions because the predictions are based on a sequence of time.

## Data Preprocessing

The data seemed remarkably close to being cleaned and there was not much to do when it came to preprocessing. The general idea of time series is that when your data is in stationarity, it gives great predictions. Thus, I decided to set this time series data in a stationary format as it showed an upward trend in some of its features for great prediction results.

The preliminary step is considered trivial, but imperative for the functioning of a sequential model. The date column is then converted to a Datetime type and considered as the index.

First of all, we would have to check for stationarity using Augmented Dickey-Fuller Test or the Kwiatkowski Phillips Schmidt Shin (KPSS) test which tells if a time series data is stationery around the mean or linear trend or non-stationary as a result of unit root. I had to import the required libraries which are from “statsmodels.tsa.stattools import adfuller” and from “statsmodels.tsa.stattools import kpss”. They use the notion of the null and alternative hypothesis and therefore when we run the tests and realize test statistics value is greater than the critical value, the null hypothesis is rejected.

A stepwise approach is discussed to transform the data to stationary one is described below:

1. Convert the desired column, which in this case is the ‘Close’ using the square root function.

2. Using the shift function by one shift, we transform the previously transformed column.
3. Use the diff() function to find the difference between step 1 and step 2.

df_log=np.sqrt(df_use['AverageTemperature']) and df_diff=df_log.diff().dropna()

## METHOD

All code was written in the python language using jupyter notebook.

The idea is to explore five (6) different machine learning models on the train and test splits.

Below are the models employed:

Multilayer Perceptron (Fully Connected)

Linear Regression

Random Forest

XGBoost

Long Short Term Memory (LSTM)

Convolutional Neural Network

The training for the models were varied in many ways.

In theory, the LSTM in most cases is supposed to perform better than all other models for time series data analysis, yet for this stationary dataset, it failed to perform better than the other models.

Since LSTM is a unique Recurrent Neural Network model, it is extremely sensitive to the timely sequential data and therefore if it's tweaked in anyway, it affects its accuracy substantially. One could infer from the deficient performance of LSTM that, the transformation of the dataset to a stationary one could play a major role in its low accuracy.

## RESULTS AND EVALUATION

Since time series is easily visualized on the cartesian plane, it was prudent for me to evaluate the models I used with the **mean absolute error**, a distance evaluation method which would give a better insight into the models than other metrics like accuracy or confusion matrix in this context being time series.

| | |
|---|---|
| Multilayer Perceptron (MLP) | 5.313058140899098 |
| Linear Regression | 3.315582509790901 |
| Random forest | 3.6337087317889907 |
| XGBoost | 4.091311438130511 |
| Long Short Term Memory (LSTM) | 7.457008734681982 |
| Convolutional Neural Network (CNN) | 7.644577730217379 |
| Long Short Term Memory (Adjusted features) | 3.0924792655098488 |
| Long Short Term Memory (Stationary Data) | 2.9702010540320124 |

As seen in the results above, the Long Short Term Memory (with Adjusted features) gave the best results for prediction.

Firstly, I tried doing the prediction of the closing price by using the previous closing prices(N) to predict the future closing price. For instance, having a period of  15 days, we take the closing prices of each of the 15 day period as the features to predict the 16th day's closing price. Thus, in this instance, N is 15. This is done by using the "Shift function" from the pandas library.

The Shift function is used to an index by the desired number of periods with an optimal time, which in this case, the index is the Date feature. It takes a required parameter of the period in an integer. This integer could be negative or positive. When it is negative, it shifts the periods desired upwards and vice versa when it's positive. For the context of this work, I had to do an upward shift.

For the **Adjusted features,** I used three main columns of the dataset, which are the Open, High and Low features in predicting the Closing price.

Open, High, and low are considered three prominent features theoretically for predicting the closing price of a ticker(stock).
Thus, using that on the LSTM gave the best results with the same parameters used for the other models.

The traditional models, which are Linear Regression, Random Forest and XGBoost gave satisfactory results because these have an advantage of being simple and interpretable and there by making it easy to understand the relationship between the features for good predictions.

However, to understand the concealed patterns in time series data it is significant not to base results on the traditional models since they don't give good interpretations of some nonlinear nature of time series data.

It could be realized that the neural network models which are MLP, LSTM, CNN had the worst results of all the models used and this I realized was due to a few factors discussed below.

To start with, the primary factor was overfitting. The neural network models were prone to overfitting because the models memorized the training data, which caused the poor results when used on the new data which was meant for testing.

Again, as discussed earlier, the complexity of time series data and its sequential manner causes the models to perform poorly sometimes. In this context, the neural network models were not able to capture all the interactions of the features well enough as it was in a sequential manner to make good predictions.

Moreover, one might be tempted to infer the poor results of the said models to limited data but that is not certain as per the context of this paper.

Lastly, the non stationarity of the data definitely played a role in the poor results. When the statistical properties of the data change over time, it makes it difficult for the neural networks to model the data accurately and this results to poor validation results.

## CHALLENGES AND FUTURE WORK

The use of Datetime type as an index feature and manipulation for sequential models is an important skill to develop as this has a long run effect in making good predictions.

Another challenge is dealing with non-stationary data as this makes it difficult for the neural networks to model the data accurately.

Lastly, parameter tuning for the neural network models is a tough task to achieve in sequential data.

However, more work should be done for implementing faster approaches in making time series data stationary and evaluation research has to go into parameter tuning of neural networks.

# CONCLUSION

In as much as traditional machine learning models do significantly good at predicting time series models, they still cannot interpret hidden patterns in time series data.

Neural network models, particularly (Long Short Term Memory) are the best for predicting  time series data since they are able to deal with the complexity of its sequential characteristics.

To end with, the essence of stationarity for good performance of machine learning models cannot be emphasized more since it helps model the data accurately considering the statistical properties  will remain constant in that regard.

## REFERENCES

Parray, I. R., Khurana, S. S., Kumar, M., & Altalbe, A. A. (2020). Time series data analysis of stock price movement using machine learning techniques. Soft Computing, 24, 16509-16517.

Löning, M., Bagnall, A., Ganesh, S., Kazakov, V., Lines, J., & Király, F. J. (2019). sktime: A unified interface for machine learning with time series. arXiv preprint arXiv:1909.07872.

Auffarth, B. (2021). Machine Learning for Time-Series with Python: Forecast, Predict, and Detect Anomalies with State-Of-the-art Machine Learning Methods. Packt Publishing, Limited.

https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.shift.html

https://www.statsmodels.org/dev/generated/statsmodels.tsa.seasonal.seasonal_decompose.html