Time series analysis of financial data

Gautham Govind A
Dept. of Electrical Engineering
Indian Institute of Technology Madras
ee19b022@smail.iitm.ac.in

***Abstract*—The objective of this final exam is to explore various avenues for generating value from a given set of data from financial markets. Due to the complex dynamics of financial markets, it is often not possible to predict how a stock will behave manually. The attempt here is to make use of machine learning models to make this prediction. All the mathematical machinery built so far in the course through assignments as well as some additional concepts not discussed so far will be used for the analysis. After gaining an understanding of the given data, we build a model for forecasting the prices. The accuracy of this model can help us understand how good our understanding is as well as help us predict future prices. Data visualization, cleaning and modelling is done using Python.**

***Index Terms*—time series, python, visualization, predictive modelling, financial data**

I. Introduction

In our everyday life, we often deal with a large amount of temporal data. Temporal data is characterised by the presence of an inherent ordering, which other forms of data lack. Often, the ordering itself is of great relevance and taking this into account is indispensable for any form of analysis whatsoever. This additional requirement often makes the analysis slightly more complicated. Though there are several techniques available for analyzing such forms of data, the use of machine learning to analyze temporal data has gained traction over the last few years, and we will be focusing on these methods.

Perhaps the most well-known example of temporal data is data from financial markets. Financial markets tend to be extremely complex with a lot of dynamics. Naturally, they generate a lot of data. Over the years people have tried out various methods for extracting value from this financial data. Deriving useful insights from financial data can be beneficial for the society as a whole. Financial data is often made available in the form of stock market prices and currency exchange rates.

The attempt here is to make use of machine learning techniques for analyzing data from financial markets. In machine learning parlance, temporal data is often referred to as time series data and the analysis of such data is hence termed time series analysis. We will be using some standard time series analysis techniques for analyzing the given stock market and currency data. Though no specific goal has been given, we will be trying to find the stock which can give the best return on investment, as this is the goal of any investor. We will also try to build a model to predict the price of stocks.

Section II gives an overview of the initial qualitative exploratory analysis. Often the amount of insights that can be gained through simple visualizations is underestimated. Section III gives a short description of the mathematical formalism behind time series analysis. Section IV describes the various time series analysis techniques and machine learning models models that were tried on the given dataset and the results that were obtained. Section V gives a summary of the major conclusions drawn from the analysis.

II. Exploratory Data Analysis

The goal in this section is to do a qualitative analysis of the given stocks to determine the best choice for investment given only this data. Of course, actual investment decisions would require considering a multitude of other factors as well. However, as data scientists, our objective is to derive as much insights as we can from the given set of data.

We are given the price and volume data for six stocks: Cognizant, HCL, HDFC, ICICI, Infosys and SBI. We are also given the INR-USD exchange rates. Before moving on with the analysis, it is important to understand the features which have been provided to us:

- Opening price:This is the price of a stock at the time of opening of the market. This need not be identical to the previous day's closing price.
- High price: This is the highest selling price of a particular stock on that day.
- Low price: This is the lowest selling price of a particular stock on that day.
- Closing price: This is the price of a stock at the closing time of the market.
- Adjusted closing price: This is the closing price after adjusting for any actions which might have affected the price after the market closed.

There are two broad trading strategies: short-term trading and long-term trading. Short-term trading typically refers to strategies in which you hold only for short durations of time, typically days. Whereas in long-term trading, you are looking to hold the stocks for years. While in short-term trading it is necessary to consider all the different price information like opening price, high price etc since they are all indications of the volatility of the stock, in long-term trading you mostly deal only with any one price, typically the closing price, as you are considering the variation over a long time and the volatility within a day doesn't matter a lot. Since we are provided with data over the duration of years, we will be looking at long term performances. Hence, we will mostly be dealing with only closing prices.

A. *Trends and reversals*

We first plot the price evolution for each stock over the given period of time. Along with the closing price, we also plot the Simple Moving Average (SMA) of the price of the stock over the past 30 days. SMA is simply the mean of stock prices for the past 30 days. This helps in filtering random price fluctuations and smoothen it out in order to see the average value. These are used to identify trends and confirm reversals.

When the price of a stock is above the SMA line, we say that the stock is in an uptrend. This simply means that the stock is doing well as indicated by the rising prices. An uptrend usually validates a buying choice in the sense that if a stock that was bought is in an uptrend, the buying choice was right. Similarly, if a stock is below the SMA line, we say that the stock is in a downtrend. This usually indicates that the stock is performing poorly.

A change in trend is termed a reversal. Reversals often indicate the occurrence of strong economic activities. However, it must be noted that since SMA is based on past prices, the evolution of SMA is not an ideal metric for predicting future prices; rather it just confirms when a trend change has taken place.

We first take a look at the tech stocks. The charts for these stocks are given in Figures 1, 2 and 3.

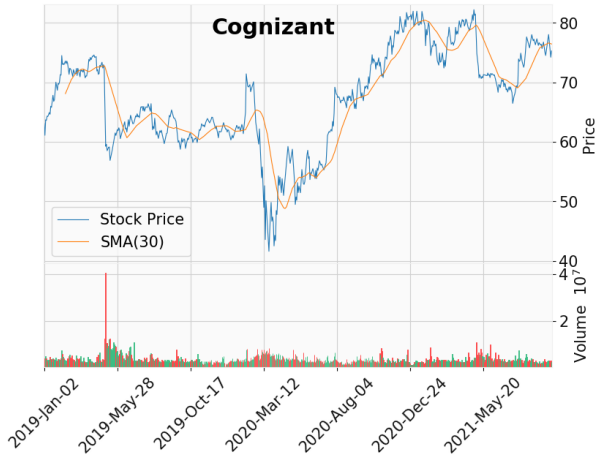

Fig. 1. Price chart for Cognizant

From a cursory look, it can be seen that the **HCL and Infosys tend to have a much more stable pattern of price evolution** when compared to Cognizant. They are also more consistently in an uptrend as opposed to Cognizant which is riddled with reversals. Also, we see two major price drops for Cognizant, one in May 2019 and one in March 2020. The one in March could be attributed to the economic slowdown, as we will see soon with the n=banking stocks. However, the drop in May seems to be something specific to Cognizant as a large fraction of shares were traded on the same day, indicating some company specific event. However, no further details can be understood without examining the economic situation at that time.

Fig. 2. Price chart for HCL

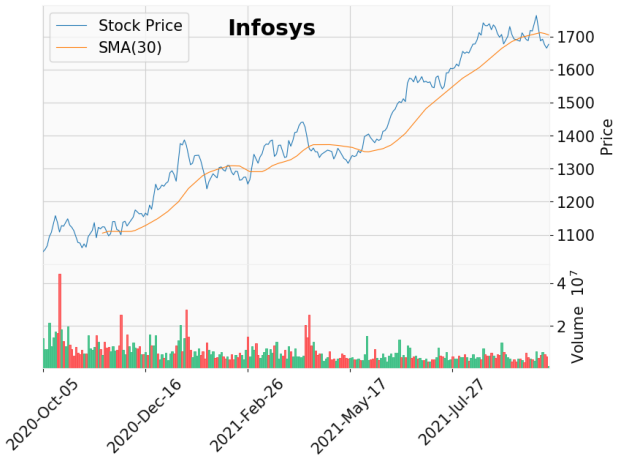

Fig. 3. Price chart for Infosys

We also look at the percentage increase in share prices over the whole of the given period. We observe that percentage increase in Cognizant share price over the period is **19.305%**, percentage increase in HCL share price over the period is **55.156%** and percentage increase in Infosys share price over the period is **59.888%**. Of the lot, Infosys and HCL seem to have done well as compared to Cognizant, which agrees with our intuition.

In any market, the time of entry (buying) and exiting (selling) are crucial. Although it practically impossible to predict the perfect entry and exit, looking at how much the ideal move would have made is often analyzed. To compute this, we look at the percentage change between the highest and lowest share prices for each stock. We obtain these to be **97.548%** for Cognizant, **68.814%** for HCL and **68.193%** for Infosys. This again shows that Cognizant is more volatile as compared to the other two stocks.

Next, we take a look at the banking stocks. The charts for these stocks are given in Figures 4, 5 and 6.

From a cursory look, it can be seen that all the three banking

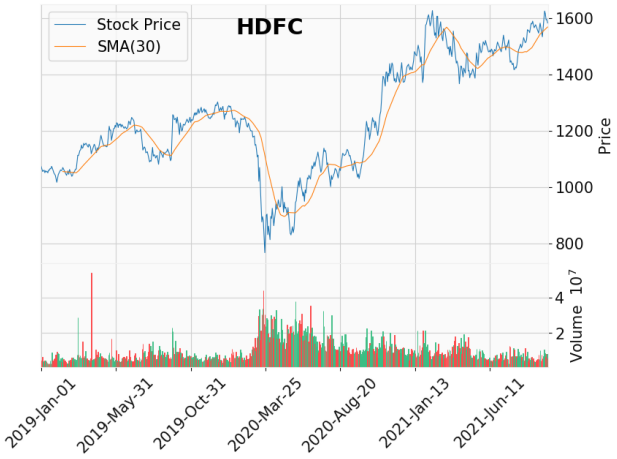

Fig. 4. Price chart for HDFC

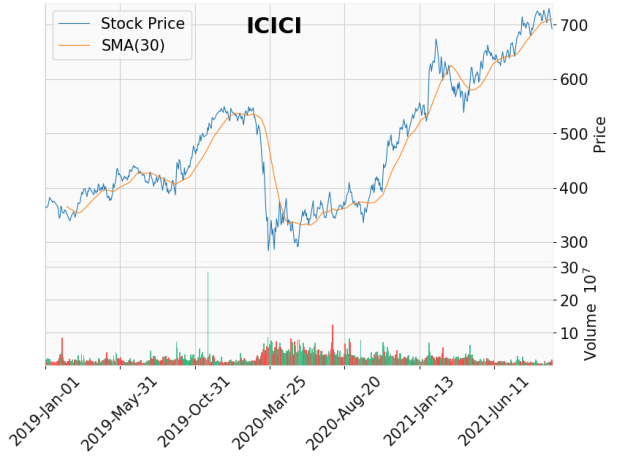

Fig. 5. Price chart for ICICI

stocks have roughly the same behaviour over the given period. All the three stocks tend to be riddled with reversals, with no trend being stable for a very long time. We also note that there is a huge dip in the prices of all the three stocks (as well as Cognizant as we saw earlier) around March 2020. This indicates that there was probably some crisis across the entire banking system. On investigation, it was discovered that this was probably due to the 'synchronised slowdown' in the World Economy. However, the stocks did recover after this slowdown and there hasn't been such a huge dip at any point after that.

We also look at the percentage increase in share prices over the whole of the given period. We observe that percentage increase in HDC share price over the period is **47.358%**, percentage increase in ICICI share price over the period is **90.336%** and percentage increase in SBI share price over the period is **50.750%**. Of the lot, the highest increase is seen for ICICI, for which the price almost doubled after the duration under consideration.

In any market, the time of entry (buying) and exiting

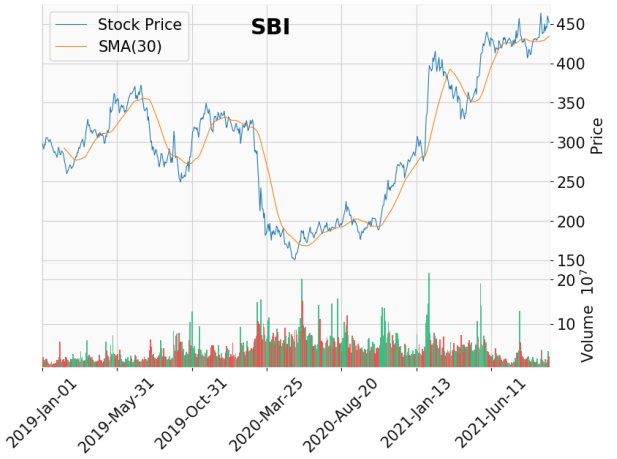

Fig. 6. Price chart for SBI

(selling) are crucial. Although it practically impossible to predict the perfect entry and exit, looking at how much the ideal move would have made is often analyzed. To compute this, we look at the percentage change between the highest and lowest share prices for each stock. We obtain these to be **111.886%** for HDFC, **157.042%** for ICICI and **207.391%** for SBI. We see that the values are all much higher than the tech stocks. This is probably due to the dip in March 2020, as someone who bought stocks then would have made a huge profit.

The charts also demonstrate the importance of having a diversified portfolio. Though the banking stocks seem to give very good returns, all of them crashed around March 2020, whereas tech stocks mostly did not. Having a diverse portfolio would help the investors stay afloat in such scenarios.

For the sake of completion, we also look at the currency exchange rates. The chart is shown in Figure 7. As we saw earlier, we see a huge decline in the value of INR in March 2020. This could again be ascribed to the economic slowdown. We also see that over the years, the value of INR has been decreasing.

B. Price and market capitalization

To get a comprehensive idea regarding the performance of all six stocks, it might be a good idea to chart the price evolution of all the stocks on the same chart. Note that the period for which information is available on the prices is somewhat different for different stocks; hence we plot for only the common time duration. The plot is shown in Figure 8.

Most stocks except Cognizant seem to have significant uptrend, at least in the first glance. HCL and Infosys seem to have a steady growth. HDFC seems to have a good uptrend but shows little stagnation in recent times. Though the are good indicators of how the companies perform, we should also consider Volume to get an estimate of amount obtained through shares. For this, we look at the market capitalization, which is the product of price and volume. The plot for market capitalization is shown in Figure 9.

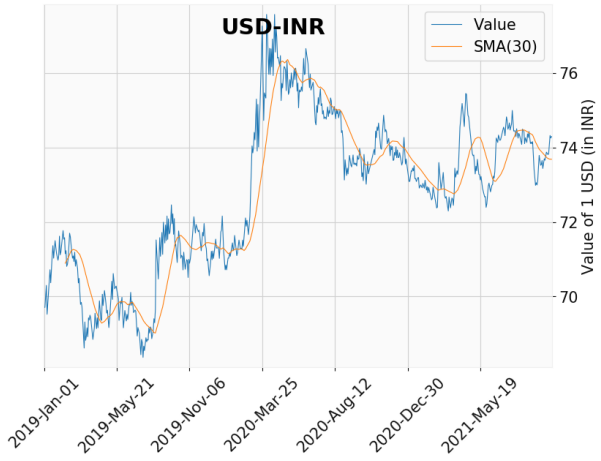

Fig. 7. USD-INR exchange rates

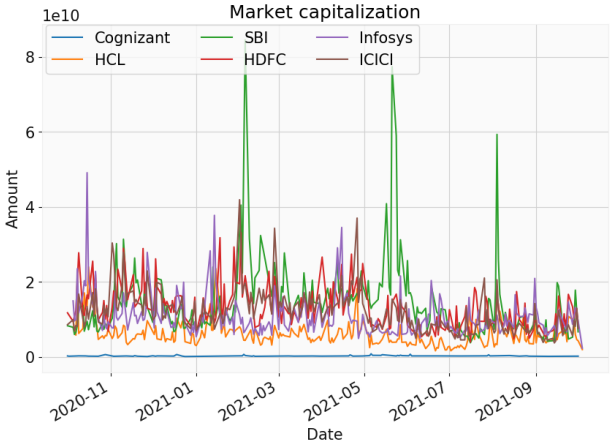

Fig. 9. Market capitalization across stocks

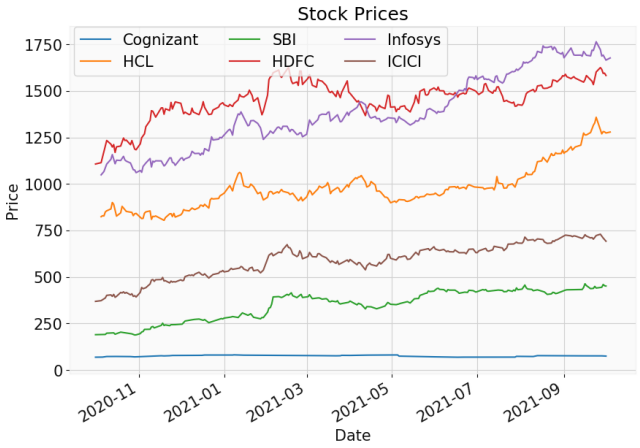

Fig. 8. Prices across stocks

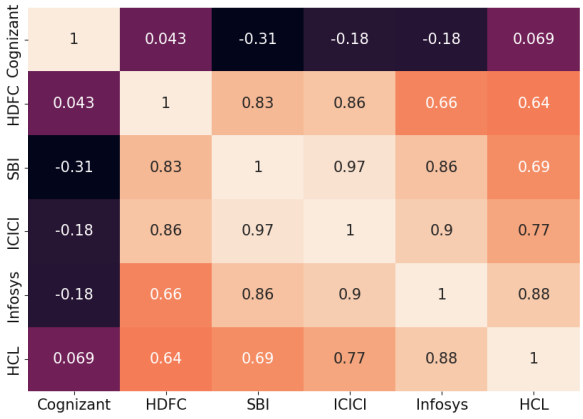

Fig. 10. Correlation across stock closing prices

Though in the previous plot the price of SBI shares weren't the highest, SBI has the highest market capitalization. It is followed closely by Infosys and the rest. One could see that SBI gains the most due to more stock volumes traded which wasn't evident from the price analysis alone. Cognizant seems to be the least market capitalization among the stocks.

We also display the correlation between the closing prices of stocks in Figure 10. Typically, we expect one stock to go up as another goes up from the same sector. This can be seen very evidently in the case of banking stocks, as the three of them have very high correlations. The trend is also followed to some extent by HCL and Infosys. Cognizant has very different behaviour, showing little to no correlation with any other stock whatsoever. This could be due to the unstable nature of the stock.

C. Rate of return

The main metric which is of importance to any investor is the Rate of Return (ROR). One would like to earn significant profits per rupee invested, taking into account the fact that for most individuals investment is limited by their financial potential. If $v[k]$ is the cost of stock on the k^{th} day, Rate of Return per rupee invested could be expressed as:

$$\text{ROR} = \frac{v[k] - v[k-1]}{v[k-1]} = \frac{v[k]}{v[k-1]} - 1$$

But since comparatively risk-free options such as Fixed Deposits (which have returns almost equal to the inflation rates) are potential options, it is useful to analyse rate of return by comparing with the inflation rate, which can be found through the changing value of INR. If $r[k]$ is the value of INR n the k^{th} day,

$$\text{ROR} = \frac{\frac{v[k]}{r[k]}}{\frac{v[k-1]}{r[k-1]}} - 1$$

The charts shown in Figures 11, 12, 13, 14, 15 and 16 display the Return of Investments per rupee invested, accounting for change in INR value. The plots seem to closely follow normal distribution centred quite close to zero. The width of the distribution is representative of the volatility of the stock.

TABLE I
RATE OF RETURN

Stock	Mean Returns (10^{-3})	Standard Deviation
Cognizant	0.476	0.021
HCL	1.836	0.017
Infosys	1.906	0.014
HDFC	0.615	0.020
ICICI	1.102	0.026
SBI	0.995	0.025

Assuming we are looking for a low risk stock, Infosys seems to be a good option. Table I mentions the mean returns per day per rupee invested and the associated risk (standard deviation) of the stock analysed in an year-long period.

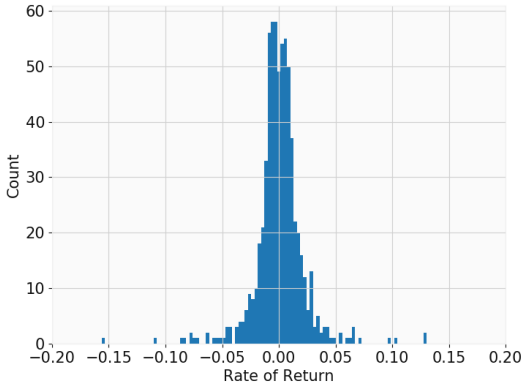

Fig. 11. Rate of returns for Cognizant

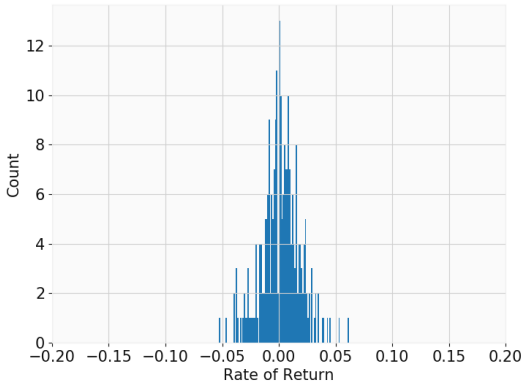

Fig. 12. Rate of returns for HCL

Infosys has the highest mean return with lowest risk. Though the standard deviation appears to be significant, we can expect it to be lower considering we are planning to hold the stock for a relatively long period of time.

Assuming that our goal is to choose a stock with maximum gains for minimal risk, we will go ahead and choose Infosys and try to forecast the prices using time series analysis. Note

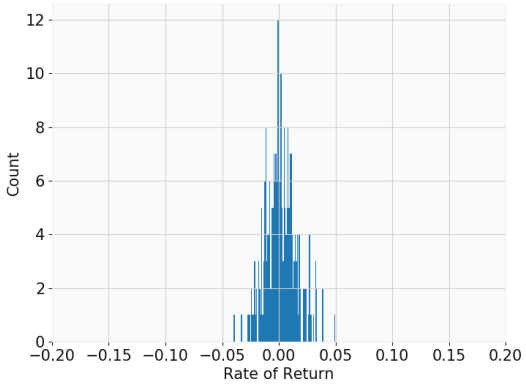

Fig. 13. Rate of returns for Infosys

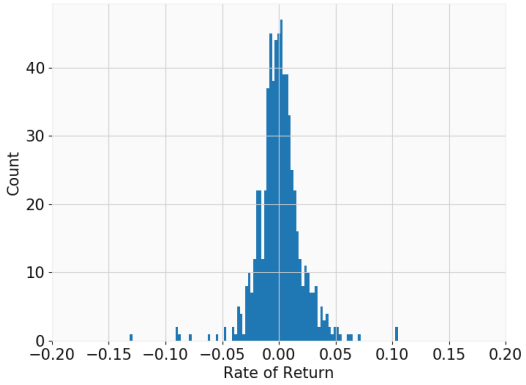

Fig. 14. Rate of returns for HDFC

that the ideal selection of stock would vary depending on variety of factors including risk tolerance of a person, how stocks in a portfolio reacts to the same market situation, corporate announcements and many more factors. We are going ahead with the Infosys stock based on the fact that it has low risk and high returns in the given period.

III. MODEL: ARIMA AND LSTM

A. Time series analysis

A time series is a sequence of data points that occur in successive order over some period of time. Time series tend to have a natural temporal ordering, and hence this can be contrasted with cross-sectional data which captures data from a point-in-time and has no natural ordering of the observations. In particular, a time series allows one to see what factors influence certain variables from period to period. Time series analysis can be useful to see how a given asset, security, or economic variable changes over time. This helps in judging the future in the context of past performance.

A set of datapoints $v[1], v[2], ..., v[k-1]$ is likely to be a time series if $v[k]$ depends on the past data. We must note that this dependence on past data is not necessarily long-term,

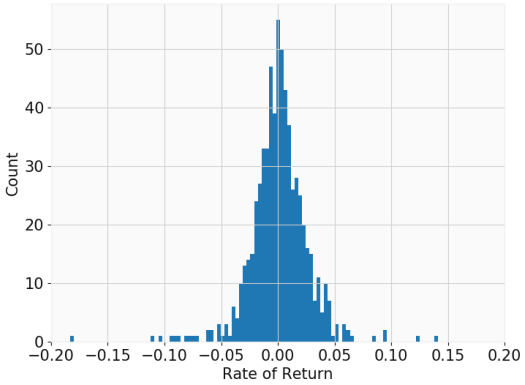

Fig. 15. Rate of returns for ICICI

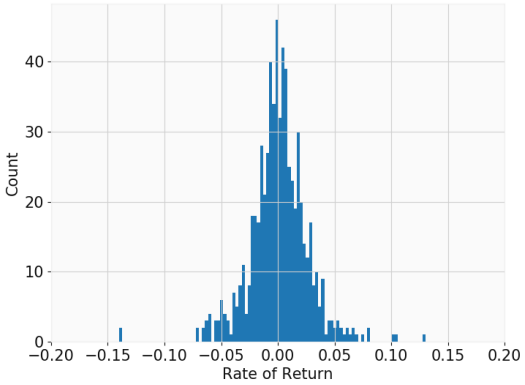

Fig. 16. Rate of returns for SBI

i.e., it is not necessary that $v[k]$ depends on values from the distant past like $v[1]$ for instance. Sometimes data could also depend on current and past residuals. The effect of past data could also be dying, in other words, could get weaker and weaker as we go more and more into the past. An example of this would be weather, where the today's weather is strongly dependent on yesterday's weather but weakly on the weather before two months.

Stationary processes: A process is said to be stationary if the statistical properties of the process are invariant with time. More formally, the process must satisfy:

$$f(v[1],..,v[k]) = f(v[T+1],..,v[T+N]) \;\; \forall N, T \in \mathbb{Z}^+$$

where f is probability density function for the process. Often it is difficult to find processes that are strictly stationary in practice, so a weaker definition of stationary processes could be taken as invariance up to second order moments. This means that the mean of the process as well as the variance should be independent of time.

At this point, it is necessary to introduce a quantity termed autocovariance. Autocovariance is nothing but the covariance matrix of the data itself, computed by considering each sample instance separately. For example, consider two instances $v[k]$ and $v[k+l]$. The autocovariance $\sigma(k, k+l)$ for this pair of samples is $var(v[k], v[k+l])$. Technically, in order to calculate auto-covariance for the entire signal, we would need the distribution of each $v[k]$. However, if we assume that the process is stationary in a weak sense, we can consider $\sigma(k, k+l)$ to be independent of sampling time k and hence the quantity can be computed irrespective of k by considering only the time lag l between the pair. It should also be noted that if the correlation is computed instead of variance, the quantity is then termed autocorrelation.

Now, any signal can be decomposed as $v[k] + e[k]$, where $v[k]$ is the predictable/signal component and $e[k]$ is the unpredictable/noise component. In an ideal model, it is common to assume that $e[k]$ is uncorrelated to other sampling instants. This would mean that we expect the ACF of the errors (Auto Correlation Function) to be 1 only at a lag of 0:

$$\rho_e[l] = \begin{cases} 1 & \text{if } l = 0 \\ 0 & \text{otherwise} \end{cases}$$

MA process: When the current prediction depends on the noise/unpredictable component of current instance and of past instances, the process is said to be an MA process(Moving Average process), We assume that the process is weak stationary. An MA process of order 1 depends only on the noise component of the previous instance, while an MA process of order M depends on M of the previous instances. Mathematically, we have:

$$v[k] = \sum_{i=1}^{M} c_i e[k-i] + e[k]$$

We have already seen that the ACF of $e[k]$, the noise component, is 1 only at lag 0 and 0 otherwise. Here, since $v[k]$ depends on past M noise components, the ACF of v[k] zeroes after M instances,i.e.,

$$\rho[l] = 0 \;\; \forall \;\; l > M$$

This is termed abrupt zeroing of ACF after M instances.

AR process: When the present data depends purely on the past instances, and the process is weak stationary, then the process is said to be an AR process (Auto Regressive process). An AR process of order 1 depends only on its immediate past, whereas an AR process of order p depends on the previous p instances, i.e., we have:

$$v[k] = \sum_{j=1}^{P} d_j e[k-j] + e[k]$$

where $e[k]$ is the noise/unpredictable component. Unlike MA process, the ACF for an AR process does not zero abruptly but rather decays exponentially. We have seen that ACF is nothing but correlation with the same variable at different sampling instants. In order to establish a similar measure for AR processes, we take the conditional correlation so that link between $v[k]$ and $v[k-l]$ is broken. This is called Partial Auto

Correlation Function (PACF). For an AR process, PACF goes to zero after p instances and the ACF decays exponentially.

The important aspect of ACF and PACF graphs is that they help up in understanding the type of process and the mathematical model behind it even though it may not be known to us beforehand. By analysing the nature of ACF and PACF plots, it is possible to come up with a linear time series model.

ARMA process: In practice, a time series process could be a combination of both MA and AR processes, i.e., current predictions could depend on both past data and past residuals. Such a process is called an ARMA process. This can be represented mathematically as:

$$v[k] = \sum_{i=1}^{M} c_i e[k-i] + \sum_{j=1}^{P} d_j e[k-j] + e[k]$$

Here, both ACF and PACF plots decays exponentially and it is difficult to judge the order of the process without further analysis.

It is often the case that the assumption of a process being weak stationary does not hold condition. For many processes, the mean and variance could change with time. An example of such a process would be $v[k] = v[k-1] + e[k] + c$, where $e[k]$ is noise. This process tends to accumulate with time and grow upwards. Hence, it cannot be classified as an ARMA process. However, the difference $v[k] - v[k-1]$ is clearly an MA/AR process of order 0. Here, on taking the difference once, the process reduces to a stationary process and hence the process is said to be integrating of order 1. If D differences are needed to reduce a process to a stationary process, the process is said to be integrating of order D.

ARIMA process: Let us represent one-time differencing $v[k] - v[k-1]$ as $\nabla v[k]$. Then if

$$\nabla^D v[k] = w[k]$$

where ∇^D denotes D-time differencing and $w[k]$ is an ARMA process of order (P, M), then we say that the $v[k]$ is in an ARIMA of order (P,D,M). In other words, if repeated D differences reduce a sample to an ARMA process of order (P,M), the sample itself is said to be in ARIMA of (P,D,M).

So far we have discussed time series analysis quite extensively, however we still haven't answered how all this analysis connects to the machine learning techniques we covered as part of this course. The idea is that once a time series model is determined, the problem reduces to a simple linear regression problem of estimating the coefficients. For example, let's say a given process is an AR process of order 2, i.e., $v[k] = d_1 v[k-1] + d_2 v[k-2] + e[k]$. Once this is known (by making use of visualizations of ACF and PACF plots), the problem reduces to finding the coefficients d_1 and d_2 which could be easily found out by setting the feature matrix X appropriately and using techniques of linear regression.

B. LSTM

LSTM, which is an acronym for Long Short Term Memory, is one of the most recent algorithms which can be used for time series analysis. LSTMs essentially belong to the class of neural networks which rose to prominence over the past decade. Over the past few years, neural networks have become ubiquitous, finding applications in a huge number of domains. Depending on the domain of application, several structures of neural networks have evolved over the years, including fully-connected networks, Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).

Of these, RNNs are of relevance to us as they evolved to deal with sequential data. Since any time series is essentially sequential data, RNNs can naturally be extended to time series data. However, traditional RNNs suffer from a problem known as the 'vanishing gradients' problem. What this means is that as the network grows in size, which is typically required by long-term time series data, the gradients start diminishing significantly which brings down model performance significantly.

LSTM networks were designed specifically to overcome this long-term dependency problem faced by RNNs. LSTMs have feedback connections which make them different from traditional neural nets. This property enables LSTMs to process entire sequences of data without treating each point in the sequence independently, but rather, retaining useful information about previous data in the sequence to help with the processing of new data points. As a result, LSTMs are particularly good at processing sequences of data such as text, speech and general time-series.

Model structure: At a basic level, the output of an LSTM at a particular point in time is dependant on three things:

1) The current long-term memory of the network — known as the cell state.
2) The output at the previous point in time — known as the previous hidden state.
3) The input data at the current time step.

LSTMs use a series of gates which control how the information in a sequence of data comes into, is stored in and leaves the network. There are three gates in a typical LSTM; forget gate, input gate and output gate. These gates can be thought of as filters and are each their own neural network.

Working: The key steps involved in the working of the LSTM gate are described below:

The first step in the process is the forget gate. The previous hidden state and the new input data are fed into a neural network. This network generates a vector where each element is in the interval $[0, 1]$. This network (within the forget gate) is trained so that it outputs a value close to 0 when a component of the input is deemed irrelevant and closer to 1 when relevant. These outputted values are then sent up and point-wise multiplied with the previous cell state. By doing this, the forget gate essentially decides which pieces of the long-term memory should now be forgotten given the previous hidden state and the new data point in the sequence.

The next step involves the new memory network and the input gate. The goal of this step is to determine what new information should be added to the networks long-term memory (cell state), given the previous hidden state and new input data. The new memory network is a tanh activated neural network

which has learned how to combine the previous hidden state and new input data to generate a 'new memory update vector'. This vector essentially contains information from the new input data given the context from the previous hidden state. The input gate is a sigmoid activated network which acts as a filter, identifying which components of the 'new memory vector' are worth retaining. This network will output a vector of values in $[0, 1]$. The outputs of new memory network and input gate are point-wise multiplied. The resulting combined vector is then added to the cell state, resulting in the long-term memory of the network being updated.

The final step involves the output gate for computing the new hidden state. We first apply the tanh function to the current cell state to obtain the squished cell state, which now lies in $[-1, 1]$. The previous hidden state and current input data are passed through the sigmoid activated neural network to obtain the filter vector. Finally, this filter vector is applied to the squished cell state by point-wise multiplication to get the new hidden state.

LSTMs generally give rise to models which are more sophisticated and accurate as compared to ARIMA models. However, one key requirement for LSTM models is that there should be an abundance of training data available. This would mean that such an analysis is possible only for long-term trading and not for short-term trading.

IV. Modelling

In this section, we apply ARIMA and LSTM models to the problem at hand for making forecasts. As discssed in section 2, we will be focusing on the Infosys stock as this was found to have highest returns with minimal risk.

We take a look at the plot for stock price of Infosys again. The plot is shown in Figure 17. On visualizing the stock, it is evident that the prices are not stationary and possess some integrating effect. We examine this through the ACF(Auto correlation) and PACF(Partial Auto correlation) plots.

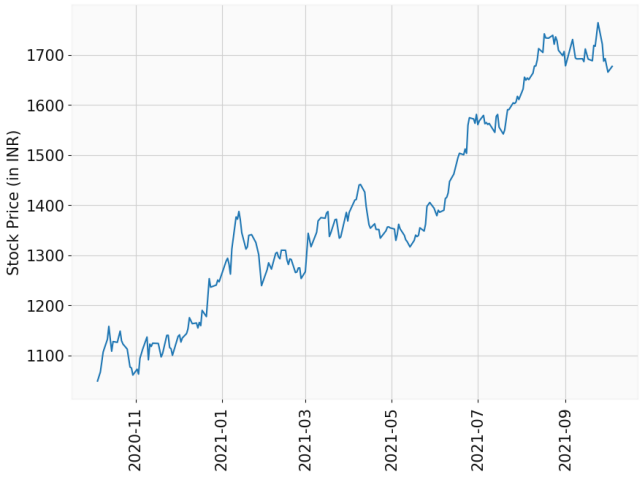

Fig. 17. Stock price chart for Infosys

These plots are shown in Figures 18 and 19. ACF plot shows long memory trend and PACF plot suggests a strong relationship between $v[k]$ and $v[k-1]$. This means that the signal could be integrating of order 1. To confirm this, ADF test was used.

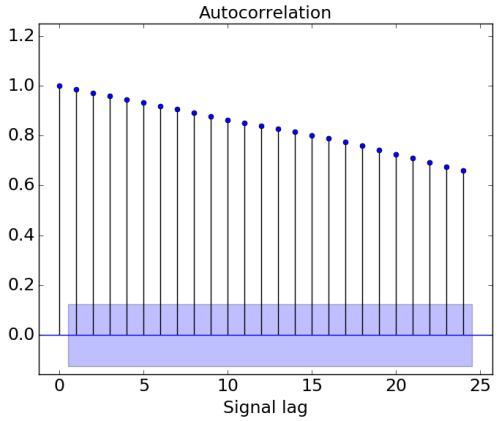

Fig. 18. Autocorrelation for prices

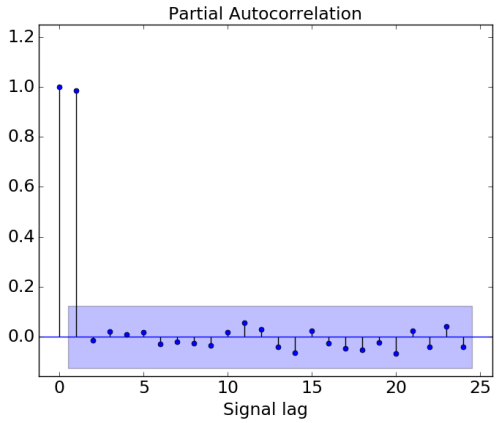

Fig. 19. Partial Autocorrelation for prices

On differencing, we obtain the plots as shown in Figures 20 and 21. The ACF and PACF clearly shows zero MA and AR processes. Hence:

$$v[k] = v[k-1] + e[k] + c$$

is the modelling choice for our Infosys stock. The residuals of the differenced series, plotted in Figure 22, shows white noise characteristics and thus further strengthens our model assumption.

The time series model we derived just now was used for fitting the stock price data till Oct-2020. The prices for the remaining time was forecasted using the fitted model and the prediction was compared with the true stock prices. The plot showing the predictions is given in Figure 23. The grey shaded region around the prediction represents the confidence interval for the predictions, obtained through statistical analysis.

We see that the model predicts the uptrend accurately and that the true values lie within the confidence interval. However, though the model predicts the error accurately, the model

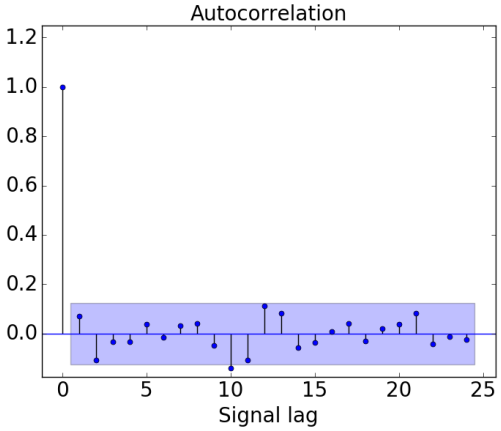

Fig. 20. Autocorrelation for difference of prices

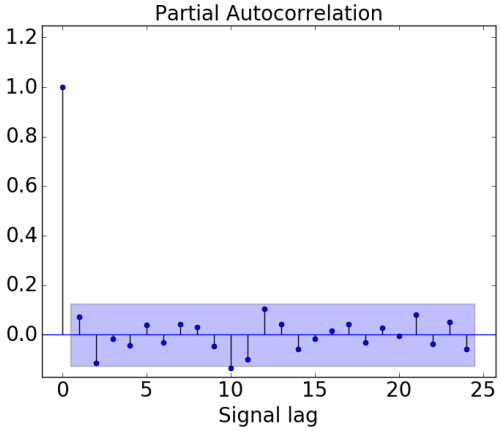

Fig. 21. Partial Autocorrelation for difference of prices

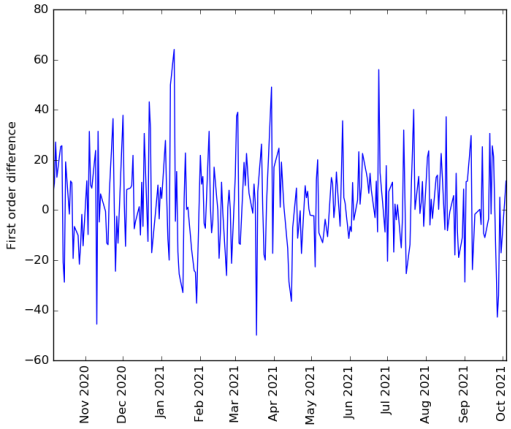

Fig. 22. First order differencing

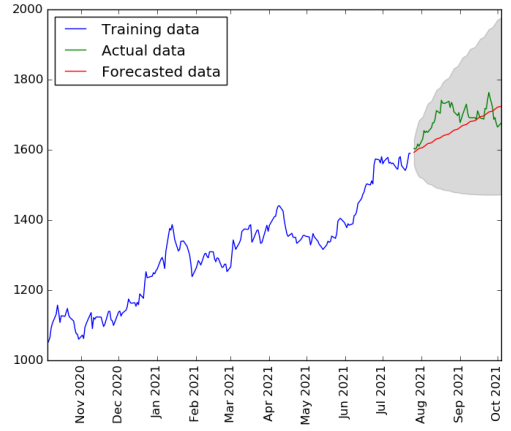

Fig. 23. Forecasting of Infosys prices using ARIMA model

seems to be too simple to accurately model the complexity associated with making the predictions.

In this context, we make use of the more sophisticated LSTM model. This model will yield much better results compared to traditional time series analysis and could capture non-linearity in a better fashion. The output from the model is shown in Figure 24. The neural network clearly outperforms the ARIMA model. However, it must be noted that training the LSTM model generally requires a larger amount of training data. Furthermore, confidence intervals cannot be generated in general for LSTM-based networks.

V. Conclusions

Based on our extensive analysis so far regarding the financial dataset, we arrive at the following set of conclusions:

- A lot of information regarding the dataset can be obtained through simple exploratory analysis. It might be worthwhile to spend time understanding the data rather than blindly applying algorithms on it.
- Exploratory analysis of the given set of stocks revealed that, of the lot, Infosys was the one with rather high gain despite having low risk. Of course, one may choose other stocks depending on their goals and risk-appetite, but from a minimal risk perspective, Infosys is a good choice.
- Real life problems such as stock market analysis could be cleverly formulated as time series analysis problems by making use of traditional time series analysis concepts.
- Classical time series analysis methods like ARIMA as well as more modern approaches like LSTMs could be used for modelling time series data. The theoretical background for these models were examined thoroughly.
- Both the ARIMA model as well as the LSTM model were able to forecast the prices for Infosys stock. ARIMA, despite being a rather simple model, predicted the uptrend accurately whereas LSTM was able to predict the price much more accurately thanks to its sophisticated nature.

VI. Avenues for further research

Only two models. namely ARIMA and LSTM were explored in this analysis. Several other tools exist in literature for the analysis of time series data. Trying out more models might help achieve better results. Furthermore, rather than considering only historical price data, effect of influence of

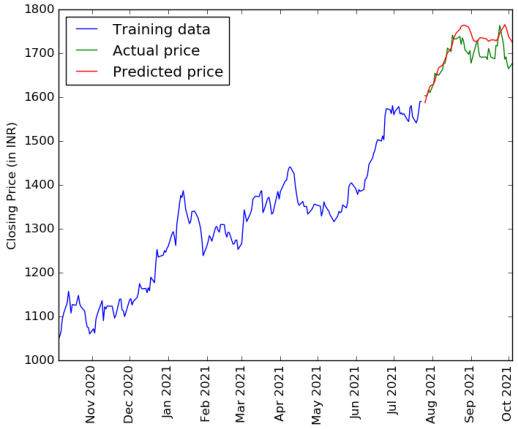

Fig. 24. Forecasting of Infosys prices using LSTM model

other factors in forecasting the prices of stocks could also be studied to give a more comprehensive picture of the scenario.

REFERENCES

[1] A. Geron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Inc., 2nd ed., 2019.

[2] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[3] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[4] Wes McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference* (Stéfan van der Walt and Jarrod Millman, eds.), pp. 56 – 61, 2010.

[5] "Time series." https://en.wikipedia.org/wiki/Time_series.

[6] "Long short term memory." https://en.wikipedia.org/wiki/Long_short-term_memory.