A REPORT ON

Forecasting Stock Price Using Bayesian LSTM

ABSTRACT

Forecasting stock price and intraday direction is an important problem in the field of Quantitative Finance. This project explores the efficacy of using Bayesian Long Short-Term Memory Neural Network Model (to be precise LSTM+BNN) for stock price forecasting. We plan to test the models' performance against other ML models (like Random Forest, XGBoost, Vanilla LSTM etc.). We are collecting data of a few publicly listed Indian stocks from Yahoo Finance python library. The model will be deployed through a webapp, that will display the model's predictions for a few selected stocks along with some statistical metrics to account for the level of confidence and uncertainty with respect to the predictions.

Keywords: Machine Learning, Bayesian LSTM, Forecasting, Trading, WebApp, Python.

TABLE OF CONTENTS

TABLE OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Over the past decades, there has been a significant progress in the fundamental aspects of information technology, which has transformed the course of business. Financial markets, as one of the most fascinating sector, have a significant impact on the nation's economy. The financial market is a dynamic and complex system in which individuals actively buy and sell currencies, stocks, shares, and derivatives via digital platforms backed by stock brokers. The stock market allows investors to purchase shares of publicly traded firms through exchange or over-the-counter trading.

One of the most quintessential indicators by which the health of any economy can be estimated is by observing the state of the stock market of that economy. The financial well-being of most people of most nations is directly or indirectly affected by the fluctuations in the stock market. Modelling stock prices and direction are very important problems in Quantitative Finance.

At any moment is time there are countless factors that effect the price of a stock. All the participants in the market are most probably analyzing every single factor and all publicly available information regarding the company, in order to value the company and figure out the fair price of the stock, given all the available information at that moment in time. Every investor does his own research, has his own biases and comes to maybe different conclusions regarding what the price of a stock should be.

Nobody can say with 100% certainty what the future business growth and profitability of a company can be. So, there is a lot of uncertainty regarding the price of the stock since the future attributes of the company are uncertain. Therefore, uncertainty or stochasticity is inherent to the stock market. That's the reason we propose the fusion the LSTM models (one of the best models used for modelling sequential data) and Bayesian Neural Networks (takes into account the epistemic uncertainty of data).

Many analysts and researchers have developed tools and techniques to forecast stock price movements and assist investors in making sound decisions. Researchers can use advanced trading algorithms to forecast the market using non-traditional textual data from media.
The use of sophisticated machine learning technologies such as text data analytics and ensemble algorithms has significantly improved prediction accuracy. Meanwhile, due to dynamic, inconsistent, and chaotic data, stock market analysis and prediction remain one of the most difficult study topics.

We acquired data from yfinance python library, and after using machine learning algorithms, we statistically presented stock forecast findings.

Our aim is to develop and test the efficacy of a Bayesian LSTM (LSTM+BNN) neural network model in forecasting the stock price and direction for the following trading day. We plan to compare the results of our Bayesian LSTM model to those of other widely known and used machine learning models.

1.1. WHY DEEP LEARNING FOR STOCK PRICE FORECASTING

In finance, especially in todays' day and age of electronic stock exchanges run by computers and dominated by computers executing various investment and trading strategies. There is abundance of stock price data, available easily and publicly. Technical Analysis is a type of analysis, done by traders and investors to predict the trend, trading range and maybe even the stock price in short term. In technical analysis one important assumption made is that all available information at any moment in time is baked into the stock price.

Suppose, if a good news regarding a particular company becomes available, then the stock price may probably go up and thus it is the good news available that made the price go up. In the same manner if bad news were to come out, stock price may fall. The stock prices react to all company related information at all times. As soon as any new information becomes available it immediately gets reflected in the price.

So, all the information needed for a particular company is baked into the stock price of the company. We need a way to extract all that information baked in the stock price movements and use it to predict the future price and machine learning algorithms excel at learning patterns from data. Thus, we are using a type of deep learning architecture that models sequential data.

1.2 Project Outline

1.2.1. Project Goals.

1. To create and explore the efficacy of a Bayesian LSTM (LSTM+BNN) neural network model in effectively predicting next days stock price.
   EX: If today stock price is 100. Our model will try to predict the next days price.

2. Comparing the performance of our Bayesian LSTM model against the other ML models (like Random Forest, XGBoost, Vanilla LSTM etc).

3. We plan to deploy the model through a website, that will display the models' predictions for a few select stocks along with a few statistical metrics to account for the level of confidence and uncertainty with respect to the predictions

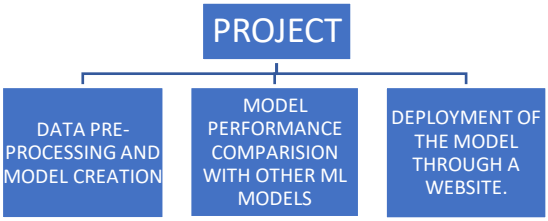

Figure 1:..

1.2.2. Creating Machine Learning Models

- Price forecasting is a Regression problem since prices are usually positive real numbers (continuous).
- We are using stock data from yahoo finance python library.
- Pre-processing includes taking care of missing values, wrong values and scaling the data.
- For most of the ML models, we will use Scikit-Learn and a few other libraries.
- For DL, we will use TensorFlow.
- Once the models are trained, we make predictions and compare their performance using different metrics (rmse, mae).

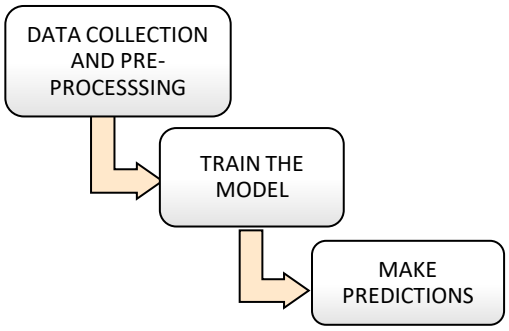

Figure 2…

1.2.3. Model Building Steps

We will be building three more models along with the proposed Bayesian LSTM model in order to compare the performance the proposed model.

For all the selected stocks, the data used is from 21-03-2017 to 02-04-2022. Which is about 5 years of daily stock data. 80% is used for model training and the rest 20% is used as a test dataset.

The models build are :

1. Random Forest.
2. XGBoost (Extreme Gradient Boosting).
3. LSTM (Long Short-Term Memory Neural Network).
4. Bayesian LSTM (LSTM+BNN)

1.2.4. Steps of Model Implementation:

1. Using Scikit-Learn and TensorFlow (For building Machine Learning and Deep Learning model).
2. Data Preprocessing and wrangling.
3. Visualization of Dataset.
4. Feature Scaling (Standardization).
5. Preparing the Datasets for training.
6. Reshaping the datasets.
7. Model Building.
8. Usage of Sequential, Dense and LSTM from TensorFlow.
9. Predicting the Output
10. Result visualization

1.2.5. The process undertaken in this project:

● Set Up Infrastructure

○ Jupyter Notebook

○ Incorporate required Libraries (Keras, Tensor flow, Pandas, Matplotlib, Sklearn, Numpy)

○ Git project organization

● Prepare Dataset

○ Integrate data of selected companies

○ Process the requested data into Pandas Dataframe

○ Develop function for normalizing data

● Develop Benchmark Model

○ Set up a basic Linear Regression model with Scikit-Learn

○ Standardize the parameters

● Develop Basic LSTM Model

○ Set up basic LSTM model with Keras using parameters from Proposed Model

● Improve LSTM Model

○ Develop, document, and compare results using additional labels for the LSMT model 5. Document and Visualize Results

● Plot Actual, Benchmark Predicted Values, and LSTM Predicted Values per time series

● Analyze and describe results for report.

....

Figure ...

2. SYSTEM CONFIGURATION

We ran entire project on an Intel I5 processor with 8 GB Ram, 3 GB Nvidia GTX 1050 Graphic Card. The processor has 4 cores which runs at 2.2 GHz. First part of the is training phase which takes 15-20 mins of time and the second part is testing part which only takes a few minutes to make predictions and calculate various error metrics.

2.1 Hardware Used:

- RAM: 8 GB
- Storage: 1 TB
- CPU: 2.2 GHz or faster
- GPU: 3GB Nvidia GTX 1050
- Architecture: 64-bit

2.2 Software requirements

• Python 3.91 in Jupyter Lab is used for data pre-processing, model training and prediction.
• Operating System: windows 10.

3. LITERATURE SURVEY

Many different types of classical and modern machine learning algorithms can be used for forecasting stock prices. The ones we have chosen to compare against BLSTM models performance are XGBoost, Random Forest and LSTM.

3.1. Long Short-Term Memory Neural Network (LSTM)

The long short-term memory network (LSTM) is a type of recurrent neural network (RNN).

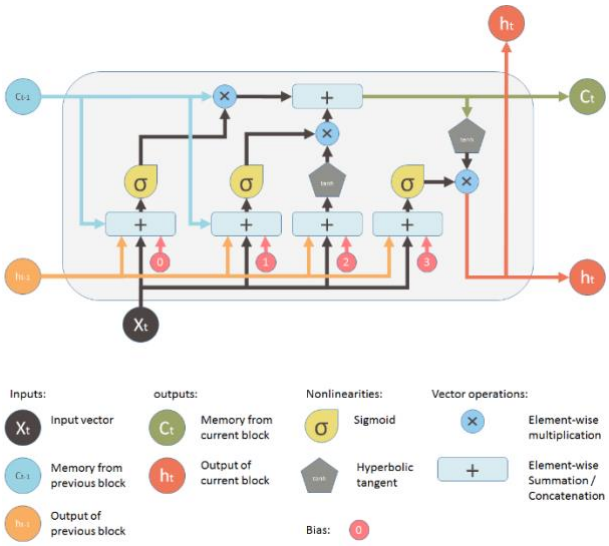

Figure 3….

LSTM is a network structure that consists of three "gate" components. An LSTM unit contains three gates: an input gate, a forgetting gate, and an output gate. Rules can be applied to information as it enters the LSTM network. Only information that adheres to the algorithm will be retained, whereas information that does not conform will be erased via the forgetting gate.

• To avoid bursting and disappearing gradients, LSTM is utilized instead of RNN. Also known as vanishing gradient problem.

• A Python library named TensorFlow is used to create and train the model in this project.

• The historical stock data table includes information such as the beginning price, the maximum price, the lowest price, the closing price, the volume, and so on.

3.1.1. Summary of LSTM Model

Below, in the picture you can see the architecture of the LSTM trained for this project. We have two LSTM layers and three Dense layers with swish activation function in our model.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 10, 128)           69120
_________________________________________________________________
lstm_1 (LSTM)                (None, 100)               91600
_________________________________________________________________
dense (Dense)                (None, 25)                2525
_________________________________________________________________
dense_1 (Dense)              (None, 10)                260
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 11
=================================================================
Total params: 163,516
Trainable params: 163,516
Non-trainable params: 0
```

Figure 4

3.2. RANDOM FOREST

The random forest algorithm is composed of a large number of decorrelated decision trees. First, the method creates random subsets of the original training set, each of which forms a decision tree. The classifiers are then ranked using all of the decision trees. Finally, a voting procedure will be used to determine the final class of a single fresh sample.

Random forest classifiers construct a set of n tree decision trees, with each tree calculated over a subset of next samples drawn with replacement from the training set. At each decision split, a random subset of feature characteristics is chosen for each tree, and the decision split is only resolved over this subset. The optimum feature and decision threshold to split on at each split is the feature that minimizes the Gini index.

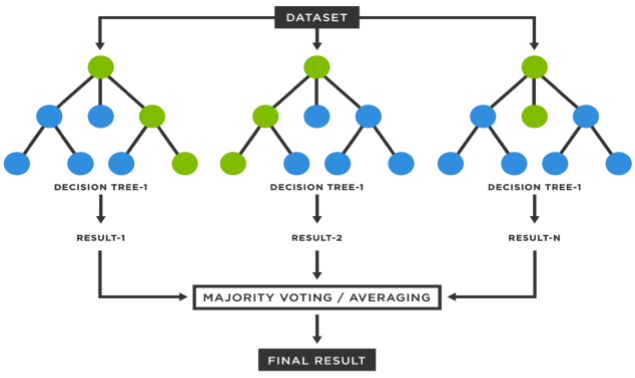

Figure 5

3.3. XGBOOST

XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. XGBoost stands for e**X**treme **G**radient **B**oosting. he XGBoost library implements the gradient boosting decision tree algorithm.

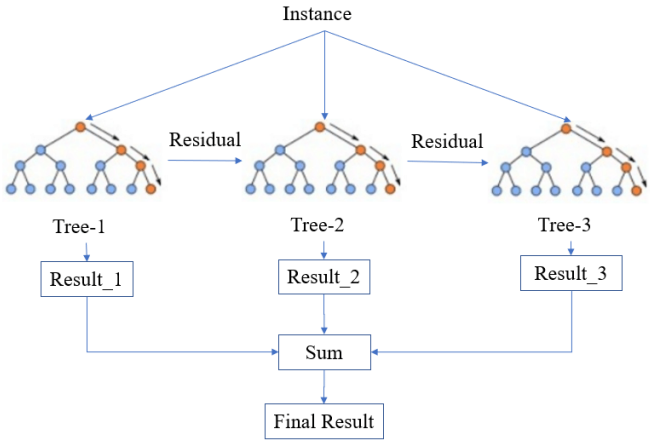

Figure 6

Boosting is an ensemble technique where new models are added to correct the errors made by existing models. Models are added sequentially until no further improvements can be made. Ensemble learning algorithms combine multiple machine learning algorithms to obtain a better model. In Gradient boosting we iteratively train an ensemble of shallow decision trees, with each iteration using the error residuals of the previous model to fit the next model. The final prediction is a weighted sum of all of the tree predictions. It is called gradient boosting because it uses a gradient descent algorithm to minimize the loss when adding new models.

This approach supports both regression and classification predictive modelling problem

4. BAYESIAN LONG SHORT-TERM MEMORY (BLSTM)

Bayesian neural network (BNN) combines neural network with Bayesian inference. Simply speaking, in BNN, we treat the weights and outputs as random variables and we are finding their marginal distributions, that best fit the data. The ultimate goal of BNN is to quantify the uncertainty introduced by the models in terms of outputs and weights so as to explain the trustworthiness of the prediction.

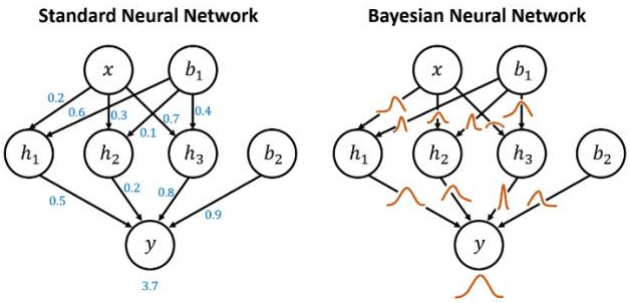

Figure 7

Creating a BNN is taking a probabilistic approach to deep learning, which allows to account for uncertainty, so that models can assign less levels of confidence to incorrect predictions. Sources of uncertainty can be found in the data, due to measurement error or noise in the labels, or the model, due to insufficient data availability for the model to learn effectively.

- Bayesian neural nets are useful for solving problems in domains where data is scarce, as a way to prevent overfitting.
- allow you to estimate uncertainty in predictions, which is a great feature for fields like finance.

The unmistakable advantage that BLSTM obviously has over all the other said classical and modern machine learning and deep learning models, is that the concept of uncertainty or the fact that the models predictions will always have associated errors due to aleatoric uncertainty (inherent non-reducible stochasticity of a phenomenon) of the process itself, whose dynamics our neural network in trying to learn, is build into architecture of the neural network itself by letting the kernel and weights be probability distributions.

4.1. SUMMARY OF BLSTM MODEL

The very first layer after input layer in the neural network is a LSTM layer, LSTM have been known to be a good model for modelling data that has sequential characteristics. After the first two stacked LSTM layers, we have bayesian dense layers with multivariate gaussian distribution as prior for the weights and biases. Here, the LSTM layers are used as feature extractors for the sequential input data, akin to how convolutional layers are used for feature extraction for image data. The final output is a univariate normal distribution, whose parameter, mean and variance are inferred from the input data by the BLSTM.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 10, 128)           69120
_________________________________________________________________
lstm_1 (LSTM)                (None, 100)               91600
_________________________________________________________________
dense_variational (DenseVari (None, 20)                2043230
_________________________________________________________________
dense_variational_1 (DenseVa (None, 10)                22365
_________________________________________________________________
dense_variational_2 (DenseVa (None, 2)                 275
_________________________________________________________________
independent_normal (Independ multiple                  0
=================================================================
Total params: 2,226,590
Trainable params: 2,226,590
Non-trainable params: 0
```

Figure 8

4.2. INFERENCE FOR BAYESIAN LSTM

The output of BLSTM is stochastic, basically for the same input the output is different every time inference is performed. So, to get proper output we are going to perform inference multiple times and then take the average of both the output parameters, mean and standard deviation. We have arbitrarily chosen 100 as the number of times we are going to perform inference for every input.

Note: Here, tqdm is nothing but a handy python module that lets create process bars with minimal to see how much the progress has been made. You just have to wrap the tqdm around the iterator in which ever loop you use.

```python
from tqdm import tqdm

def compute_predictions(model, iterations=100):
    predicted = []
    for _ in tqdm(range(iterations)):
        predicted.append(model.predict(X_val))
    predicted = np.concatenate(predicted, axis=1).astype('float16')

    prediction_mean = np.mean(predicted[:, ::2], axis=1)
    prediction_std = np.mean(predicted[:, 1::2], axis=1)

    return prediction_mean, prediction_std
```

Figure 9

4.3 DATASETS

The data used for training and testing the models in this project, has been procured from a python library for stock data, called yfinance. We are using roughly last five years of daily stock data i.e, from 21-03-2017 to 02-04-2022. We specifically use the down load function available in yfinance library which takes the stock symbol and start and end date of the range for which you want the stock data.

GET STOCK DATA

```python
def get_stock_data(symbol: str, start: str = '2017-03-21', end: str = '2022-04-19') -> pd.DataFrame:
    return yfinance.download(symbol, start=start, end=end).drop('Close', axis=1)
```

Figure 10

4.3.1. Features Used

The features or attributes used by the model to predict stock prices are

1. Open
2. High
3. Low
4. Close
5. Volume
6. Return made last day
7. Target

Date	Open	High	Low	Adj Close	Volume	returns_lag1	target
2017-03-23	637.900024	648.000000	628.150024	634.849976	4690121	-0.004754	-0.005794
2017-03-24	635.450012	640.450012	612.599976	616.900024	4865013	-0.005794	-0.028274
2017-03-27	615.000000	623.000000	602.349976	614.400024	3535098	-0.028274	-0.004053
2017-03-28	617.200012	633.400024	615.099976	628.799988	2981122	-0.004053	0.023437
2017-03-29	632.900024	637.000000	622.400024	628.250000	2407007	0.023437	-0.000875

Figure 11

1. **Open:** The price at which the stock gets traded

for the first time, daily when market opens.

2. **High:** The highest price at which stock gets traded during the day.

3. **Low:** The lowest price at which stock gets traded during the day.

4. **Adj Close:** The last price at which the stock gets traded during the day(Adjusted for corporate actions).

5. **Volume:** The total number of stocks that were traded during the day.

6. **returns_lag1:** The percentage change between the day before yesterday's close price and yesterday's close price.

7. **Target:** Percentage change from today's close price to tomorrow

5. RESULTS

We have used two metrics for measuring the performance of our models

1. Mean Absolute Error (MAE).
2. Root Mean Square Error (RMSE).

5.1. MEAN ABSOLUTE ERROR

Statistical metrics are regression error measures that I utilized to calculate risks. Model assessment is crucial, and it must be done in order to decrease risks and improve model performance. Mean Absolute Error is model evaluation metric used with regression models. The mean absolute error of a model with respect to a test set is the mean of the absolute values of the individual prediction errors on over all instances in the test set. Each prediction error is the difference between the true value and the predicted value for the instance.

$$\text{MAE} = \underbrace{\frac{1}{n}}_{\text{test set}} \sum_{i=1}^{n} \underbrace{|y_i}_{\text{predicted vaue}} - \underbrace{\hat{y}_i|}_{\text{actual value}}$$

5.2. ROOT MEAN SQUARE ERROR

Root Mean Square Error (RMSE) is the standard deviation of the residuals. Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit. Root mean square error is commonly used in regression analysis to verify experimental results.

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

RANDOM FOREST

Table 1….

Company	Mean Absolute Error	Root Mean Square Error
RELIANCE	0.64316578	0.802307886
DRREDDY	0.889898892	0.791508692
DMART	0.878363656	0.937226365
TCS	0.668591044	0.81665996
HINDUNILVR	0.82491006	0.95843139

XG BOOST

Company	Mean Absolute Error	Root Mean Square Error
RELIANCE	0.578790354	0.772887016
DRREDDY	0.596687908	0.842765383
DMART	0.611012931	0.875152381
TCS	0.588608086	0.805233605
HINDUNILVR	0.63478113	0.865925098

Table 2…

LSTM

Company	Mean Absolute Error	Root Mean Square Error
RELIANCE	0.032816578	0.043307886
DRREDDY	0.024998892	0.035508692
DMART	0.0345618	0.047063656
TCS	0.014991044	0.02035996
HINDUNILVR	0.021491006	0.028431399

Table 3…

B-LSTM

Company	Mean Absolute Error	Root Mean Square Error
RELIANCE	0.015351936	0.019722749
DRREDDY	0.014902782	0.019386198
DMART	0.020524899	0.025631355
TCS	0.012512391	0.016087439
HINDUNILVR	0.013078887	0.016761115

Table 4…

5.3 PREDICTIONS MADE BY THE BLSTM MODEL

Below you see the visual representation of our model predictions and the true value of the stock price for the various selected stocks, whose names and the industry/sector they operate in specified in parenthesis.

1. DMART (RETAIL SECTOR)
2. DRREDDY (PHARMACAUTICAL)
3. HINDUNILVR (FMCG)
4. RELIANCE (CONGLOMERATE – OIL & GAS REFINING, RETAIL, TELECOMMUNICATIONS, PETROCHEMICALS)
5. TCS (IT SECTOR)

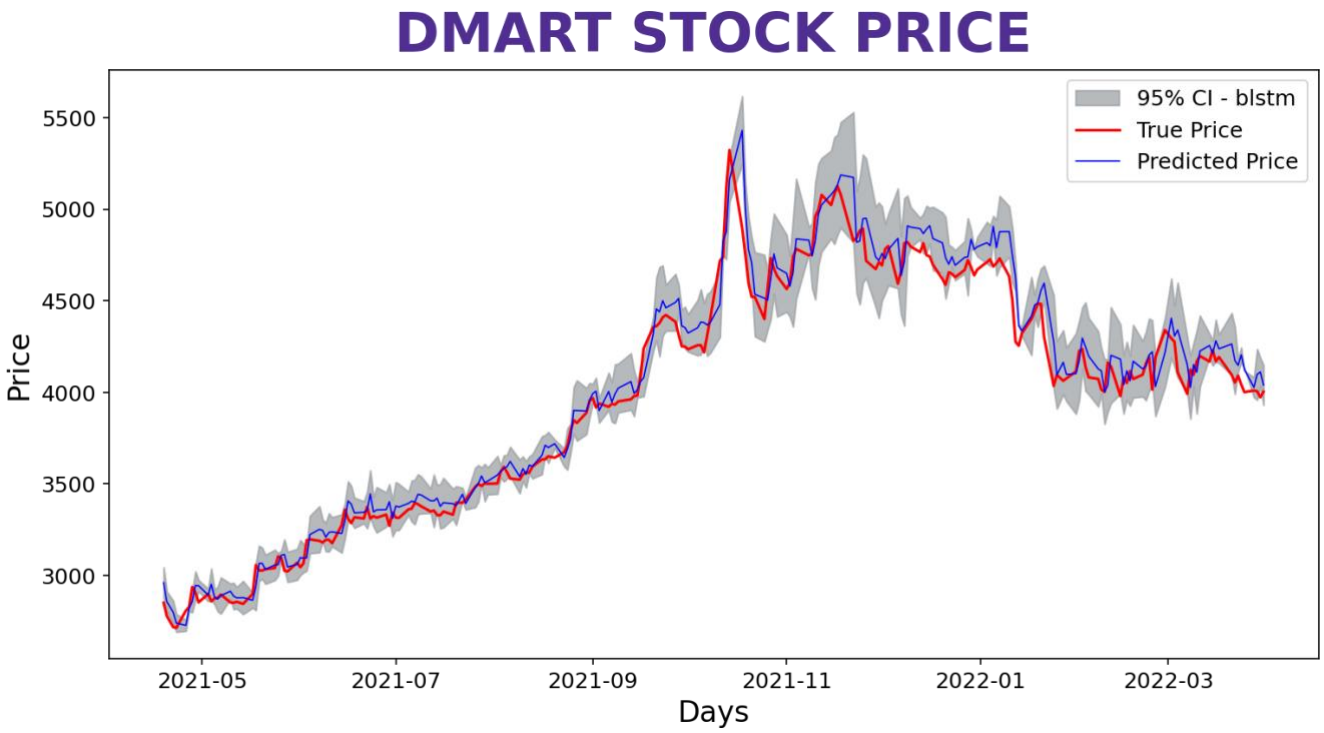

Figure 12

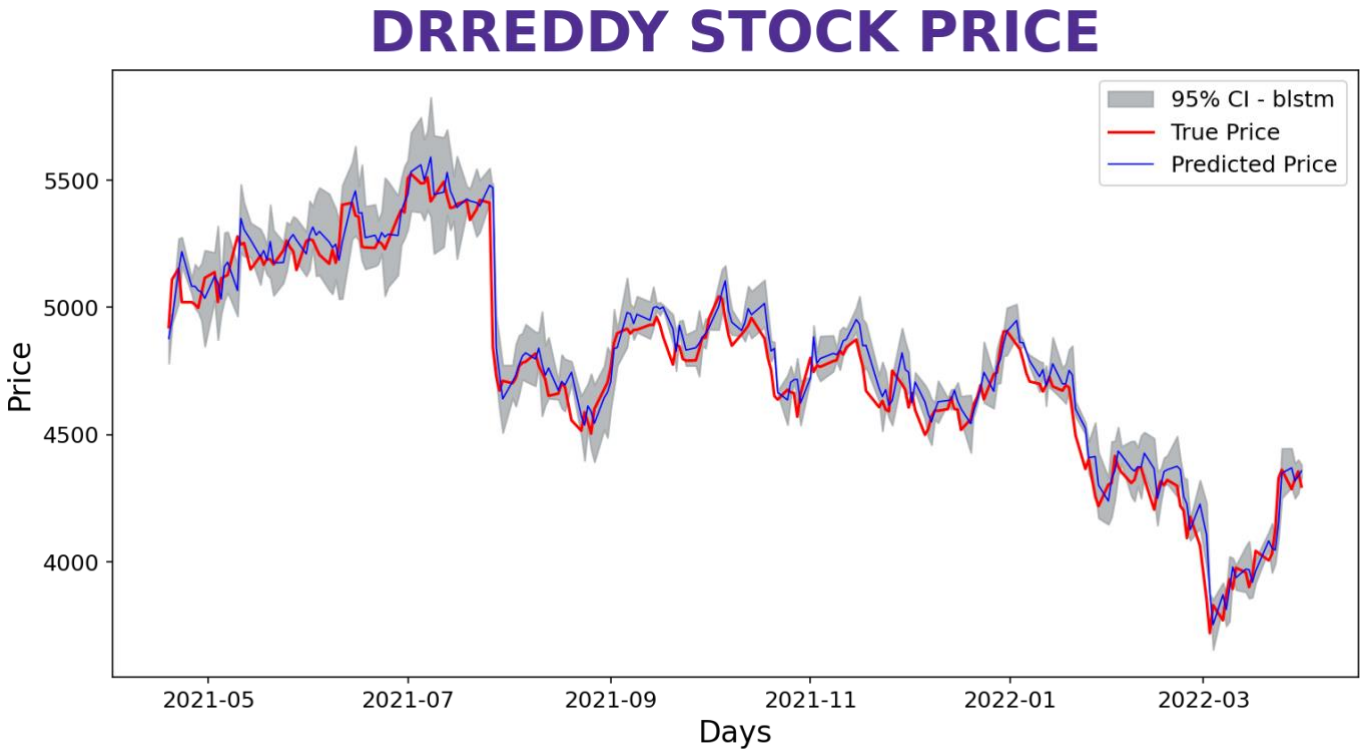

Figure 13

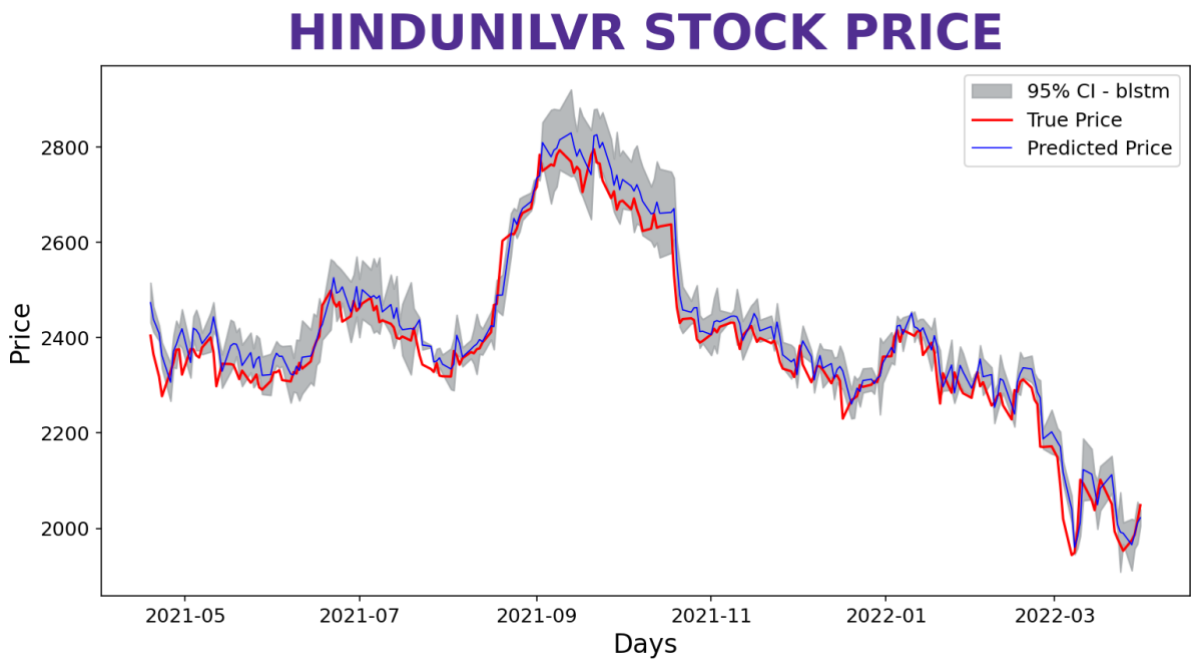

Figure 14

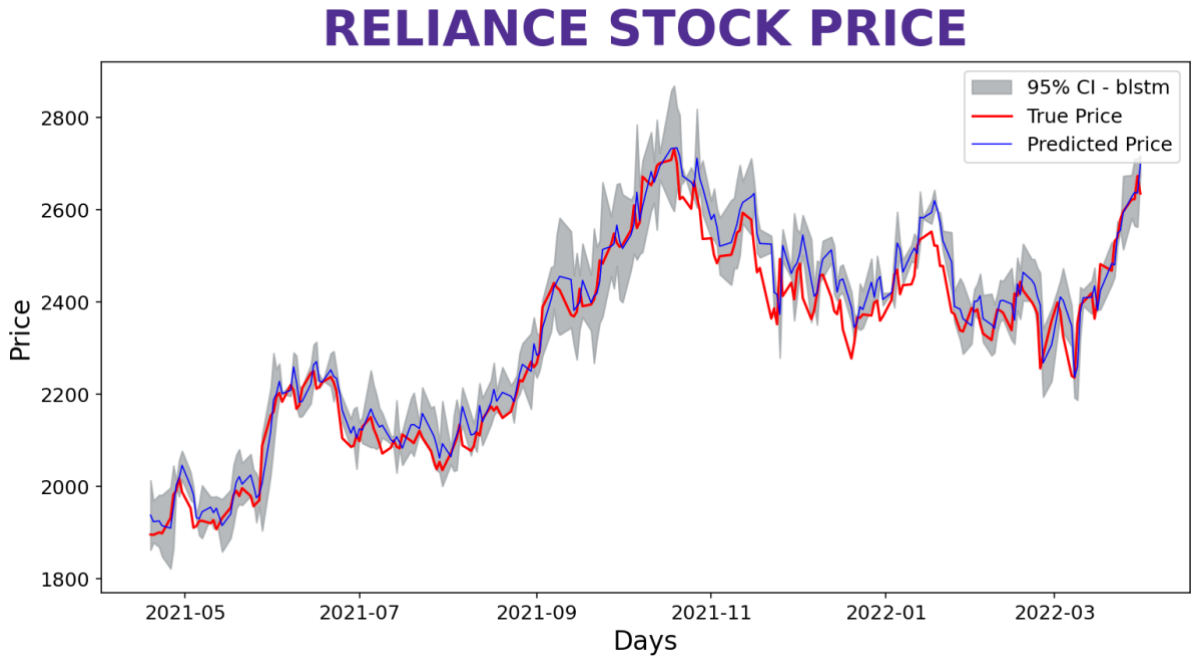

Figure 15

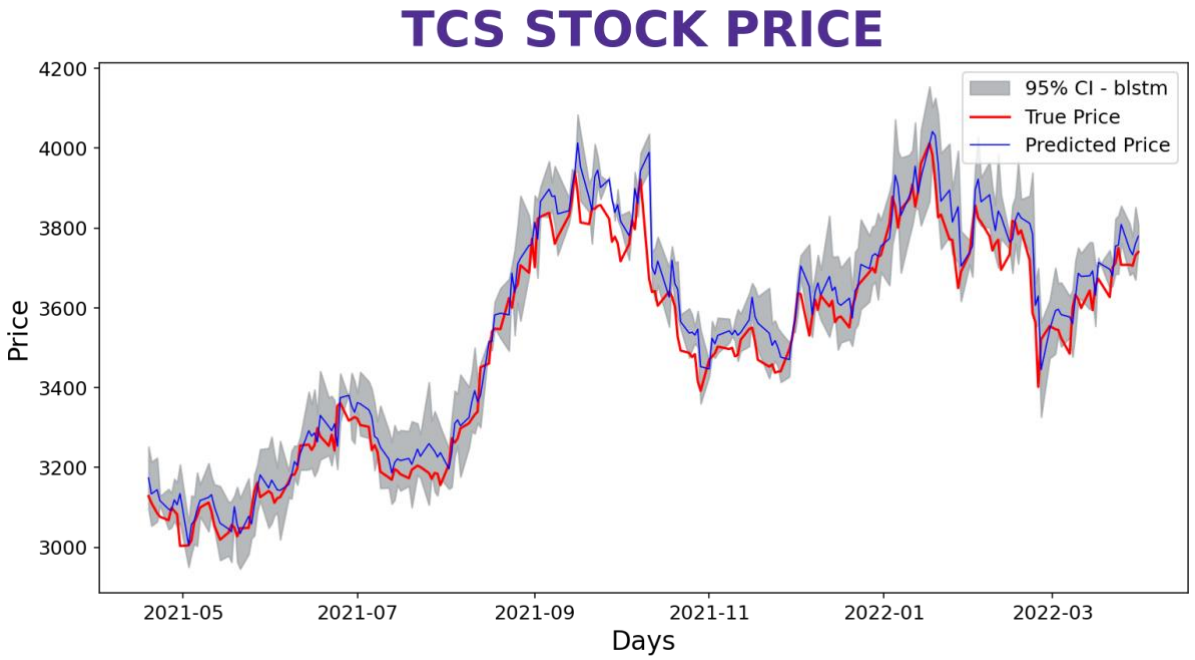

Figure 16

6. DEPLOYING THE MODEL USING STREAMLIT:

Data visualization is a critical stage in data analysis. It is a method of communicating your study and data (set) conclusions using interactive plots and charts. There are several libraries available for data visualization, such as matplotlib, seaborn, and others, that allow us to visualize a wide range of charts and plots, but these libraries do not provide any functionality for deploying them in the form of a web page or web app.

You can use Streamlit to build applications for your machine learning project using simple Python scripts. It also supports hot-reloading, which allows your app to update in real time as you change and save content. With the Streamlit API, you can build an app in only a few lines of code.

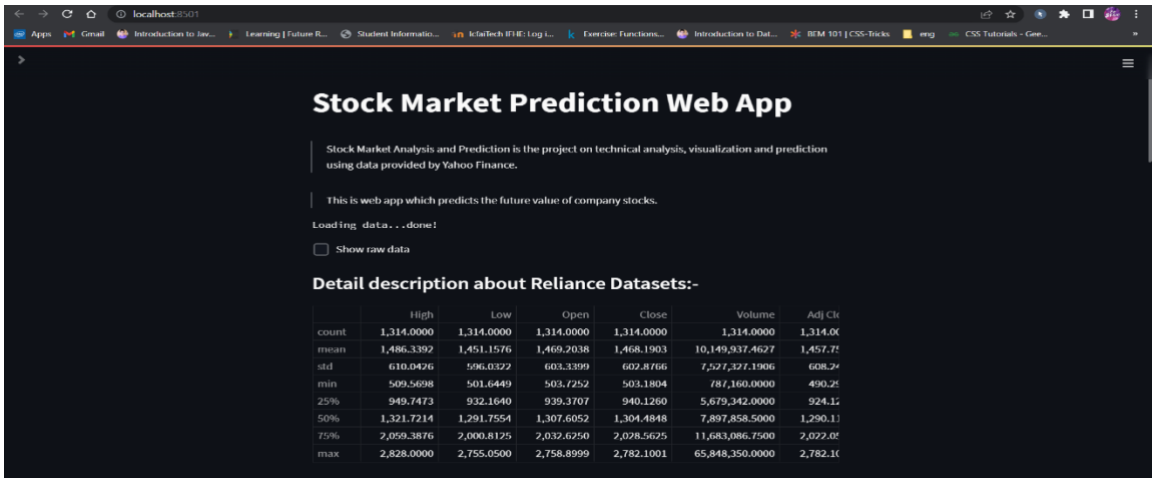

Figure 17 : Stock Prediction Web App

Code Snippet:

```
# showing the description of data
st.subheader('Detail description about Reliance Datasets:-')
descrb=data.describe()
st.write(descrb)
```

```
# Loading data from yahoo finance.
start=dt.date(2017,1,1)
end=dt.date.today()
data=pdr.get_data_yahoo("RELIANCE.NS", start, end)
```

Figure 18 : Web App Code

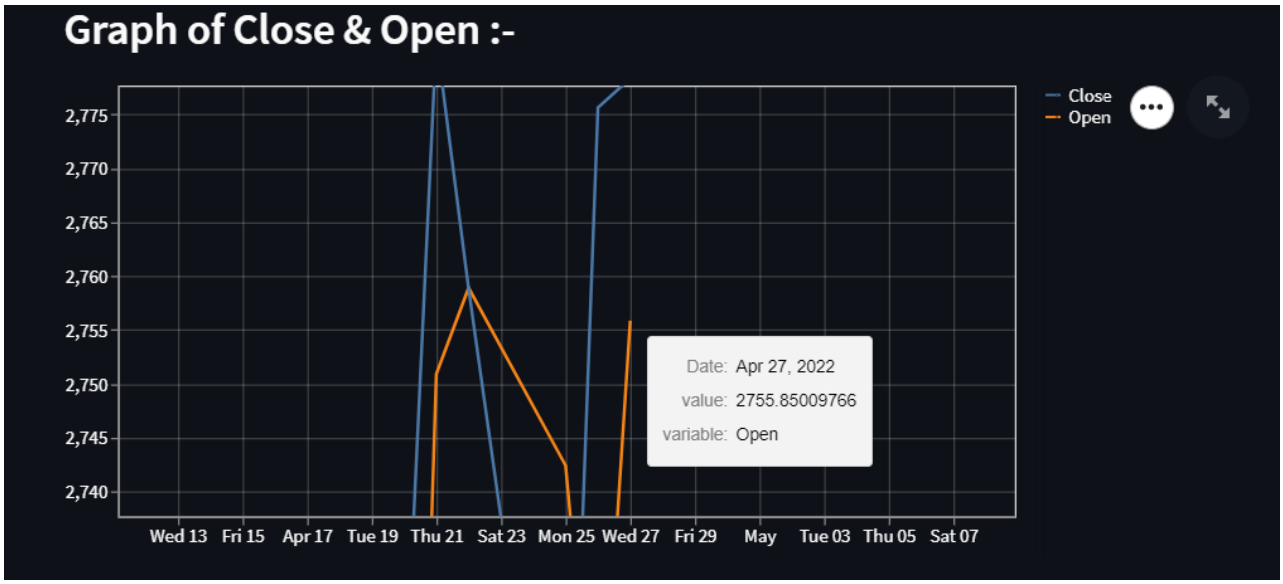

Figure 19 : Graph of Open & Close

Figure 20 : Graph of High

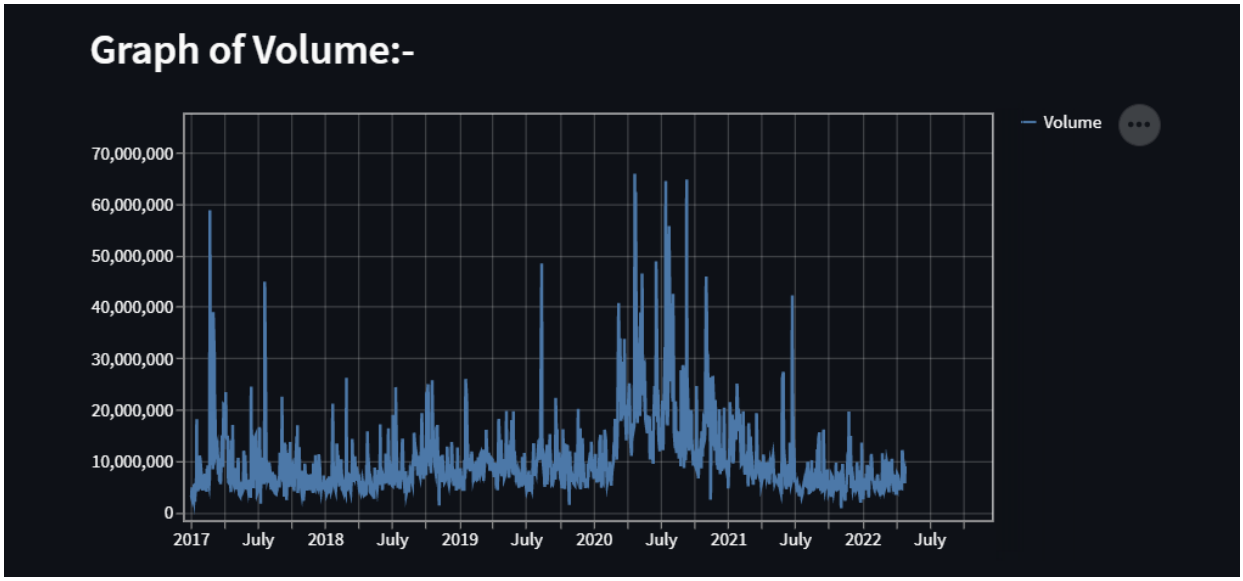

Figure 21 : Graph of Volume

Code Snippet:

```
# dislaying graph of open and close column
st.subheader('Graph of Close & Open :-')
st.line_chart(data[["Open","Close"]])

# displaying plot of High column in datasets
st.subheader('Graph of High:-')
st.line_chart(data['High'])

# displaying plot of Adj Close column in datasets
st.subheader('Graph of Adjacent Close:-')
st.line_chart(data['Adj Close'])

# displaying plot of volume column in datasets
st.subheader('Graph of Volume:-')
st.line_chart(data['Volume'])
```

Figure 22 : Code for graphs

7. CONCLUSION & FUTURE SCOPE

7.1 CONCLUSION

We have found that when compared to other modern (XGBOOST) and classic (RANDOM FOREST) machine learning and deep learning models and architectures (vanilla LSTM), the Bayesian LSTM performs the best with a RMSE and MAE far less than the said other machine learning and deep learning models. The unmistakable advantage that BLSTM obviously has over all the other said classical and modern machine learning and deep learning models, is that the concept of uncertainty or the fact that the models predictions will always have associated errors due to aleatoric uncertainty (inherent non-reducible stochasticity of a phenomenon) of the process itself, whose dynamics our neural network in trying to learn, is build into architecture of the neural network itself by letting the kernel and weights be probability distributions.

7.2 FUTURE SCOPE

8. References

[1] https://www.analyticsvidhya.com/blog/2020/12/deploying-machine-learning-models-using-streamlit-an-introductory-guide-to-model-deployment/

[2]https://github.com/bshubham5359/Stock-Market-Analysis-and-Prediction-WebApps/blob/master/AnalysisStockPrice.py

[3] https://arxiv.org/abs/2004.10178

[4] https://arxiv.org/abs/2009.10819

[5] https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0253217

[6]https://machinelearningmastery.com/how-to-develop-lstm-models-for-time-series-forecasting/