INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, ALLAHABAD

# Machine Learning Fellow Capstone

Supervisor : Dr.Ranjana Vyas

# REPORT

---

# Optimising LSTM using Genetic Algorithms on TimeSeries Data.

**Group Members**
Asha Jyothi Donga-IIT2019006
Azmeera Mounika - IIT2019133
Shwethaa R - IIT2019185
Mitta Lekhana Reddy - IIT2019204
Dhanush Vasa - IIT2019208

# Contents

# Abstract

Time series datasets have been more popular in recent years. Improved computer power have made it possible to capture and analyse massive volumes of data, which has unleashed immense prospects for data analysis and the discovery of new patterns. Due to the extreme volatility and unpredictability of financial time series, investors, financial institutions, and academics have long found it difficult to accurately anticipate stock price movements. Multivariate time series forecasting, with a focus on stock price forecasting, is at the heart of this piece of work. Deep methods (LSTM) have been widely used for this time series analytic challenge of stock market forecasting. In this research, we look at how Genetic Algorithm may be used to optimise several LSTM variants for better performance. The model will be validated and trained using the NIFTY 50 dataset.

# 1. Introduction

Time series data is of the highest significance since modern companies are continually looking for ways to increase their income by modifying production or consumption to match future requirements. As a result of this, modern enterprises are constantly trying to collect more data. The past may be better understood via the application of statistical analysis, in addition to its value in forecasting the future. Time series analysis may be used to monitor not just the physical systems, software systems, and other systems that fall outside the realm of business, but it can also be utilised to assist boost the yield of an existing firm.Retail investors from all over the globe are growing more interested in the stock market, which has led to the production of a vast quantity of data each day by financial institutions. In addition, the emergence of digital trading platforms has increased the accessibility of this data.

Institutions such as huge Frequency Trading organisations and investment banks largely depend on time series analysis when it comes to making judgements that might potentially lead to huge financial losses or profits. It is difficult to create accurate forecasts of the stock market due to the fact that the stock price data is very noisy and that it is highly volatile. This causes the data to behave as if the stock market were engaging in a random walk, making it impossible to forecast the stock market. As a result, forecasting the prices of stocks on the stock market has developed into an intriguing topic of study in analytics.

Traditionally, time series analysis was performed using statistical approaches such as ARIMA and ARMA, amongst others; however, these models were only successful when applied to univariate or linear data. They are unable to accurately capture the nonlinear patterns caused by external sources such as trading price series and stock prices. Next, the performance of ML models such as SVM, tree-based models, and regression models was shown to be superior to that of statistical approaches. The results of deep learning approaches were superior. RNN is a deep learning model that works well with sequential input, but it does not have the capacity to retain long-term memory. Therefore, in order to solve this issue, LSTM, which stands for long short-term memory, was created.

LSTM models, despite the fact that they have shown high performance, are nonetheless confronted with a few problems throughout the development process. When it comes to time series analysis, the performance of LSTM models is favourable dependent on the hyperparameters, such as window size, that are used. The researchers have arrived at these hyper-parameters by a process of trial and error; nonetheless, this does not guarantee the best possible outcomes. LSTM models base their performance on the features that they

make use of, and the parameters that make up these features are very sensitive to the data that is being taken into account. In addition, in order to calculate the LSTM model's hyper-parameters, one needs a significant amount of both domain knowledge and practical expertise. The selection of features has been proven to have a considerable impact on the overall performance of LSTM models, according to research. There is not always a direct correlation between the amount of characteristics used and improved performance.

This study addresses the numerous shortcomings of LSTM models in order to develop an effective, intelligent system that enhances performance. This research employs genetic algorithms, an evolutionary technique, to optimise LSTM models for optimal performance. In a sense, genetic algorithms are natural selection-inspired search heuristics. In addition to analysing and comparing LSTM variants, the study concentrates on their efficacy and optimisation.

# 2. Problem Statement

This study examines the various variants of LSTMs (Classic, Stacked, and Bidirectional) and the optimisation of performance using various approaches to the Genetic Algorithm with multivariate time series data. With particular emphasis on equity market information.

# 3.Motivation

There are currently a variety of applications for time series analysis. From stock price prediction to weather forecasting, there are a variety of problems involving time series data that all require the use of time series data.  Forecasting in time series is a technique for predicting future values based on trends and fluctuations at previous timestamps. Additionally, it can be used to analyse the past and the present.

There are several contexts and effects that time series analysis may have. From a monetary point of view, it is feasible to amass enormous wealth via astute analysis of financial time series data. Energy generating enterprises may better anticipate future energy demand and consumption with the use of time series forecasting from an ecological standpoint. Therefore, it is essential to enhance the standard of time series analysis to get the best possible outcomes.

We want to investigate time series analysis that focuses on optimising the performance of existing models for multivariate time series analysis since there is a significant lot of unpredictability associated with time series data. This is why we want to investigate time series analysis.

# 4. Literature Survey

The discipline of astronomy is a good place to look for data that shows how time series models have been investigated and used over the course of many centuries. The utilisation of time series analysis ranges from comprehending the motions of the planets to finding the most effective answers to a variety of challenges faced by businesses. The majority of the time, time series analysis has been seen as a statistical issue falling within the umbrella of the regression model.

In the study of time series data, two of the most used statistical approaches are ARMA, which stands for auto regressive moving average, and ARIMA, which stands for auto regressive integrated moving average. When applied to univariate or linear data, these statistical approaches perform significantly more effectively. In order to address the problem of missing data, many statistical procedures have been developed, such as Hotdecking and Mean replacement. The multivariate time series model, in which a greater number of factors that influence the trend are included, has the potential to produce more accurate forecasts. However, when the time series grows more complicated and is impacted by the unpredictability of the elements that are external, the statistical regression models become less successful. [1] In the research article titled "Solar Irradiation Forecasting," written by Bhaskar P. Murkhothy and Vikas Maurya, the authors investigate how sequence to sequence deep learning models may be used to time series data of solar irradiation in order to achieve high levels of accuracy. LSTMs were shown to offer superior accuracies for long-term modelling and were able to capture the more complicated non-linear patterns. This was in contrast to the standard statistical models such as ARMA and ARIMA, which were found to be less accurate. [2] In the study that was conducted by Jui-Yu Wu and You-Ting Chien, they compared the effectiveness of LSTM to that of basic back-propagation neural networks. The findings demonstrated that the two types of neural networks were equally effective. It is clear from reading the study that the number of cells used has a significant impact on the performance of the LSTM network. This is true to a large degree. [3] Despite the fact that many different iterations of LSTM have been established throughout the course of time, time series analysis using LSTM has been less investigated, particularly in the context of stock price prediction. Samuel Olusegun Ojo and colleagues have presented a method through which stacked LSTM may be employed for the time series forecasting, and they have also investigated the behaviour of the stock market. [4] According to the findings that were presented by Sima Siami-Namini and colleagues, the performance of bidirectional LSTM in predicting time series demonstrates the suitability of bidirectional LSTM for the prediction of stock prices. They made the observation that the BiLSTM model required more time to converge than the others, but once it did, it outperformed the traditional LSTM model and the ARIMA model. The choice of hyper-parameters is also very important for the performance of bidirectional LSTM. [5] No matter what the nature of the issue may be, one cannot understate the significance of locating these hyper-parameters in order to provide accurate forecasts. This is an essential step in the process. Nguyen Nhat Anh et al. constructed an optimised LSTM for the purpose of predicting the load on an electrical grid. In addition, they demonstrated the viability of using genetic algorithms (GA) for the purpose of selecting input features. They also said that using ideal hyper parameters is required in order to achieve optimal outcomes, and that is something that they did in order to fine tune the LSTM hyper parameters using Bayesian optimisation. [6] Hyper parameters are extraneous parameters that are not a component of the model and, as a result, cannot be predicted based on the dataset. However, they may be customised by trial and error until an accuracy level that is acceptable is reached. Francesco Marino and colleagues have shown how genetic algorithms may be used to determine the ideal hyper-parameters in complicated trends for a variety of machine learning methods like as Support Vector Machine, Random Forest, and others. The performance of their classification models was significantly enhanced when they used these optimum hyper parameters. The majority of combinatorial optimisation procedures, such as integer programming, dynamic programming, branch bound

approaches, and so on, are computationally prohibitive when applied to a significant and adequate amount of variables.

# 5. Dataset Description

NIFTY 50 data sourced from India's National Stock Exchange (NSE) will serve as the basis for our training and evaluation of the performance of our model, respectively. The data ranges from the first of January in 2000 to the thirty-first of April in 2021. This collection includes the data of each and every company that is indexed on a day-by-day basis.

It has the following features:

1. **Open:** Opening price of the stock for that day.
2. **Close:** Closing price of the stock for that day.
3. **Prev Close:** Closing price of the stock for the previous day.
4. **Low:** Lowest price of the stock reached for that day.
5. **High:** Highest price of the stock reached for that day.
6. **Volume:** Total amount of the stock traded on that day.

The function of the target variable is played by the near feature. It is the one that requires a prediction to be made. For the purpose of projecting the closing price at the timestamp "t+1," the data up to a certain timestamp "t" is employed. We utilised data from "Adani Ports" rather than any of the other fifty firms that were indexed.

# 6. Preliminaries

## 4.1: Building LSTM model

In order to forecast stock prices, we apply this model. Because it is able to learn long-term relationships, this model is the most effective one for stock prediction, particularly when applied to situations involving sequence prediction. Long-Short Term Memory Networks are a Special sort of RNN That Can Learn Long-Term Dependencies Long-Short Term Memory Networks are a sort of RNN that can learn long-term dependencies. Because the RNN suffers from issues such as vanishing gradients and the difficulty of long-term dependencies, this model is superior than the RNN. Even though RNN has the capacity to remember the information, the model is not successful when there is more of a gap. In order to get over this obstacle, we are using the LSTM model as our stock forecast. This model is reliant on long-term dependencies.

### Difference between RNN and LSTM

All RNNs share the same chain of repeating cells, and their structures are straightforward, consisting of a single layer with a tanh activation function. In addition, LSTMs have a chain structure, but the structure of the repeating model is more complicated. There is not only one layer, but rather there are four levels.

### Working of LSTM network:

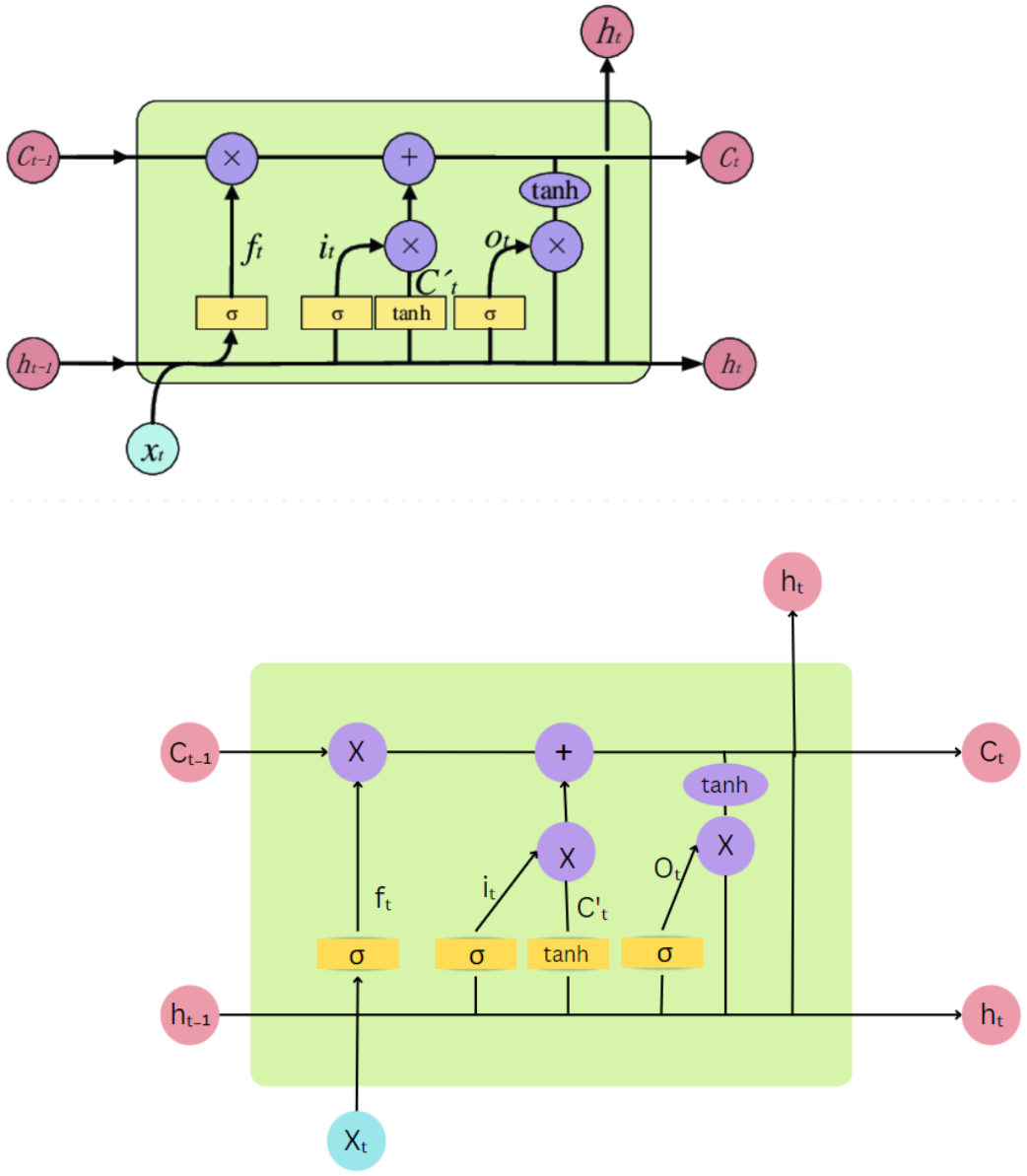



**Figure 1 : LSTM**

Lstm functions similarly to RNN. The progression from one cell state to another is shown by the top horizontal line in the figure. This is analogous to a modified conveyor belt that moves sequentially along the supply chain. LSTM uses gates to either add to or remove from the state of a cell. The sigmoid neural network layer serves as the basis for the gates' point-wise multiplication action. The output from the sigmoid layer varies between 1 and 0. A value of 1 allows the component to pass through any data. Zero indicates complete and utter closure. The three gates in Lstm are the forget gate, the input gate, and the output gate.

**Forget gate**: The forget gate filters the information from the previous time step using the input and concealed state from the most recent time step. This process occurs whenever the state of the cell changes.

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

**Input gate**: The information that should be added to the cell state is decided by this component. Determines how much of the information should be maintained by filtering the current input as well as the prior concealed state.

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Current cell is computed by adding new information and forgetting the useless information.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

**Output gate**:Using the current input, previous hidden state, and current cell state, the output gate produces the current hidden state of the cell.

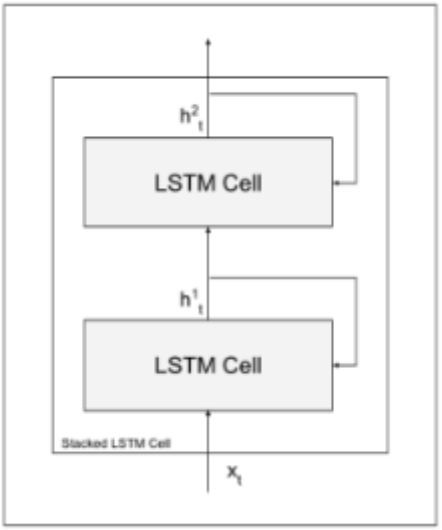
$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Comparative analysis is done for 3 types of LSTM models and dataset:

1. Classic LSTM
2. Stacked LSTM
3. Bidirectional LSTM

**Stacked LSTM:**



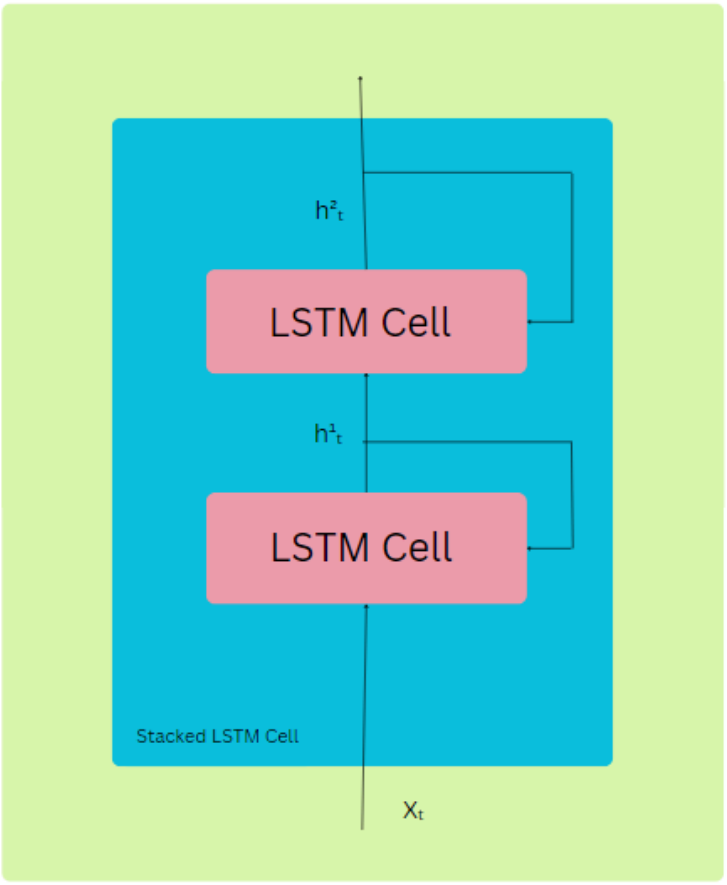



**Figure 2 : Stacked LSTM**

A stacked LSTM architecture is one that has many layers of LSTMs. The LSTM layer below receives output in the form of a sequence, not a single value, as opposed to receiving output from the LSTM layer above in the form of a single value. The model is made deeper by the use of stacked LSTM, which in turn provides improved prediction and accuracy.

**Bidirectional LSTM:**

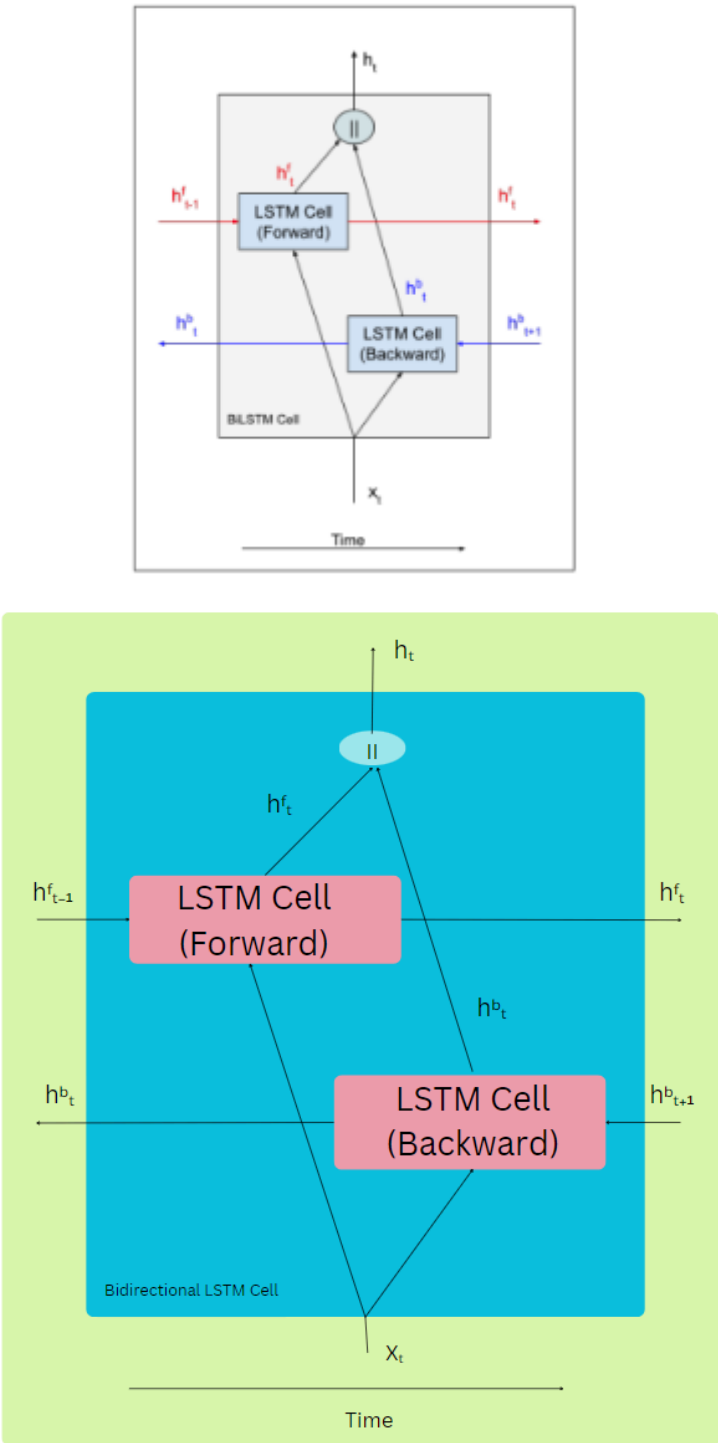



**Figure 3 : Bidirectional LSTM**

Bidirectional LSTM receives input from both sides, meaning that it considers both the input from the past and the input from the future while attempting to predict the output of the present hidden layer. For the purpose of preserving both the present and the past, input flows in both ways.

In order to train these models, not only do the best hyper-parameters that were developed by the genetic algorithm but also all of the characteristics that were found in the dataset are used.

$$RMSE = \sqrt{\frac{\sum_{t=1}^{N}(Y_t - F_t)^2}{N}}$$

RMSE can be used to measure the performance of each model. RSME can be calculated as

Among all combinations  of hyper-parameters, the combination which has less RMSE value is optimal.

## 4.2: Optimization of Hyperparameters using genetic algorithm

The genetic algorithm (GA), a kind of search strategy that takes its cues from the natural world, explores a solution space in search of the best possible path. This stochastic evolutionary algorithm drew its motivation from the idea of natural selection as it evolved through time. Since one of the primary goals of GA is to simulate natural processes, it starts with a population of people and then changes them over the course of time. During the search phase, the people in the population stand in for prospective solutions that might be applied to the optimisation strategy that has been described.

- **Genotype**: It is the representation of the individual/potential solution with which the genetic algorithm works. It's sometimes also called a chromosome/solution.
- **Phenotype**: It is the representation of the individual/potential solution that has the actual meaning for the problem for the optimal solution.

During the process of determining a person's physical fitness, the genetic representation of that person is converted into a phenotypic representation of that person.
When looking for the most efficient solution in the search space, the GA strikes a compromise between two competing objectives: first, it investigates the whole of the search universe (exploration), and second, it searches a subset of the search space for the optimal solution (exploration).
When looking for the most efficient solution in the search space, the GA strikes a compromise between two competing objectives: first, it investigates the whole of the search universe (exploration), and second, it searches a subset of the search space for the optimal solution (exploration).
The genetic algorithm (GA) develops the population pool until it converges on the optimal solution by using a selection function for the exploitation purpose and genetic operators (crossover and mutation) for the exploration purpose. This allows the GA to accomplish the aforementioned objectives.

These operations are described:

- **Selection:** The selection function determines which people must be retained for future reproduction and chooses the individuals based on their fitness evaluation. The selection processes being analyzed are: top selection, roulette wheel selection, tournament selection.
    - **Top Selection/Rank select**: This selection process consists of obtaining all the scores of the individual genotypes and selecting the top scores as the fittest parents.
    - **Roulette Wheel:** When choosing a roulette wheel, the circumference of the wheel is divided according to the genotype evaluation, a fixed point is selected on the circumference of the wheel, and the wheel is rotated. The wheel's area that makes contact with the fixed point is selected as the parent.
    - **Tournament Selection:** In the K-Way tournament, we randomly choose K people from the population, and we choose the best of these to become parents. The following parent is chosen using the same procedure as before. Due to its ability to function even when fitness values are negative, tournament selection is a very common literary device.

    -
- **Genetic Operators:** From the current generation's chosen individuals, new individuals for the following generation are created using genetic operators. Two categories of genetic operators exist:
    - **Crossover:** It combines the genetic information of more than one parent individual to produce new offspring. It produces offspring by recombination of genotypes of selected individuals. Different Crossover techniques that can be used are : One-Point crossover, 2-Point/K-Point crossover, Uniform crossover.
        - **One-Point crossover:** In this one-point crossover, the parents' tails are switched to produce new offspring at a randomly selected crossing site.
        - **2-point/K-point crossover:** Points are chosen at random from the parent chromosomes in 2-point crossover. The parent organisms switch the bits between the two crossover locations. similarly to K-Points
        - **Uniform Crossover:** Each bit in a uniform crossover is equally likely to come from either parent. There are occasions when different mixing ratios are employed, resulting in offspring that receive a greater genetic contribution from one parent than the other. Instead of segmenting the chromosome, we treat each gene separately in a uniform crossover. For each chromosome, we effectively flip a coin to determine whether or not it will be present in the progeny. We can also tilt the odds in favour of one parent so that more genetic material from that parent is present in the child.

    - **Mutation:** Mutation is a minor, arbitrary change made to a chromosome to produce a novel outcome. It is usually utilised with a low probability and is used to preserve and introduce variation in the genetic population. Different mutation techniques based on the genotype representation are: bit inversion, swap mutation, scramble mutation.
        - **Bit inversion:** In bit inversion we select one or more random bits and flip them. This is used for binary encoded GAs.
        - **Swap mutation:** In swap mutation we select two positions on the chromosome at random, and interchange the values.
        - **Scramble Mutation:** In scramble mutation of the entire chromosome, a subset of genes is chosen and their values are scrambled or shuffled randomly.
- **Fitness / Cost Evaluation:** The fitness function is a function that takes a potential solution to the problem as input and outputs how suitable or effective the answer is in relation to the problem under discussion. The calculation of fitness value must be quick enough because it is performed repeatedly in a GA. A GA may be negatively impacted and become excessively slow due to a slow computation of the fitness value.

    - **RMSE (Root Mean Square Error):** is the square root of the mean of the square of all of the error. The use of RMSE is very common, and it is considered an excellent general purpose error metric for numerical predictions.

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

- **Validation by k fold:** It is commonly used in applied machine learning to compare and select a model for a given predictive modeling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods. Dataset is spliced into k equal parts. K-1 parts are used for training, and the other 1 part is used for testing.
  The general procedure is as follows:

    1. Shuffle the dataset randomly.
    2. Split the dataset into k groups
    3. For each unique group:
        1. Take the group as a hold out or test data set
        2. Take the remaining groups as a training data set
        3. Fit a model on the training set and evaluate it on the test set
        4. Retain the evaluation score and discard the model
    4. Summarise the skill of the model using the sample of model evaluation scores.

The k value must be chosen carefully for your data sample. A poorly chosen value for k may result in a misrepresentative idea of the skill of the model, such as a score with a high variance (that may change a lot based on the data used to fit the model), or a high bias (such as an overestimate of the skill of the model).

# 7. Proposed Methodology

In this paper, we have implemented 3 different LSTM models: Classical, Stacked and Bidirectional LSTM. All of these algorithms were run without using any optimizations and later Genetic Algorithm was used to optimize the selection of hyper-parameters.
The hyper-parameters of this model are:

- Window size - The total number of days we consider to predict the close price of stock for the next day.
- Number of LSTM units - Describes the number of cells present in a layer.

## 7.1 Without Genetic Algorithm

Keras was used to create the LSTM models. To avoid over-fitting to the training set, the models were trained for 100 epochs on the training data with a dropout of 0.2. There was an 80/20 split in the dataset. Using Mean Squared Error (MSE) as the loss function, ADAM Optimiser ensured quicker convergence. For model training with all features, the hyper-parameters were left at their default settings. The number of LSTM units utilized was also set to 30, and the window size was assumed to be 30.

## 7.2 With Genetic Algorithm

There are 2 phases in our proposed methodology. In the first phase we look to optimize the hyper parameters using genetic algorithms, and in the second phase LSTM models are built.

### The First phase: Optimization using genetic algorithm

In LSTM layers, it is necessary to optimize the hyper-parameters, as well as the size of the time frame and the number of units. In order to satisfy both sets of criteria, it is necessary to make feature choices that are acceptable. Because both the size of the time frame and the number of units are integers, the information may be encoded using the binary representation. The one bit is sufficient for the parts of the feature to which it applies. If the bit's value is 1, the feature will be used; if the bit's value is 0, the feature will not be used since it is not necessary. Assuming that the number of days in the time window is less than 64 and the number of units is less than 128, there are five features, and because each feature needs one bit, the time window must be less than 64. Therefore, a total of 6 plus 7 plus 5 bits is needed. As a result, an 18-bit binary string was used as the genotype.

A genetic algorithm will be used to evolve the initial population of 20 individuals. For every iteration in the genetic algorithm, lambda operations of offspring are produced. From the parent population, the individuals are selected for the mutation to produce the offspring.

After generation of offspring the mutation of the individuals is selected from the pool of parent population and offspring population.

**Algorithm 1: genetic algorithm**

**Require**: number_Of_Generations, gene_Length, population_size, selection_Stratagy, μ,λ, cx_pb, mut_pb

```
Initialise Initial_Population(population_size, gene_length)
fitness _evaluation(inital_Population)

For g:1 to number_Of_Generations do
    parent_population = g-1th population
    Offspring_population-> list()
    For o : 1 to λ do
        offspring=generat_offspring(parent_population, cx_pb, mute_pb)
        offspring_population.append(offspring)
    fitness _evaluation(offspring_population)
    gth_population=select(selection_strategy(g-1th_population, offspring_ population))
Return best individual of last generation
```

## The second phase: Building LSTM models and Calculating fitness

At first, the model will be trained using the hyper parameter values that are used the most often, and we will utilize all of the characteristics from the dataset. The dataset will be divided for training and testing in the proportion of 80:20, and the process will be executed on a predetermined number of epochs. The Mean Square Error will be used as the loss function, and the Adam optimizer will be utilized for the purpose of optimization.

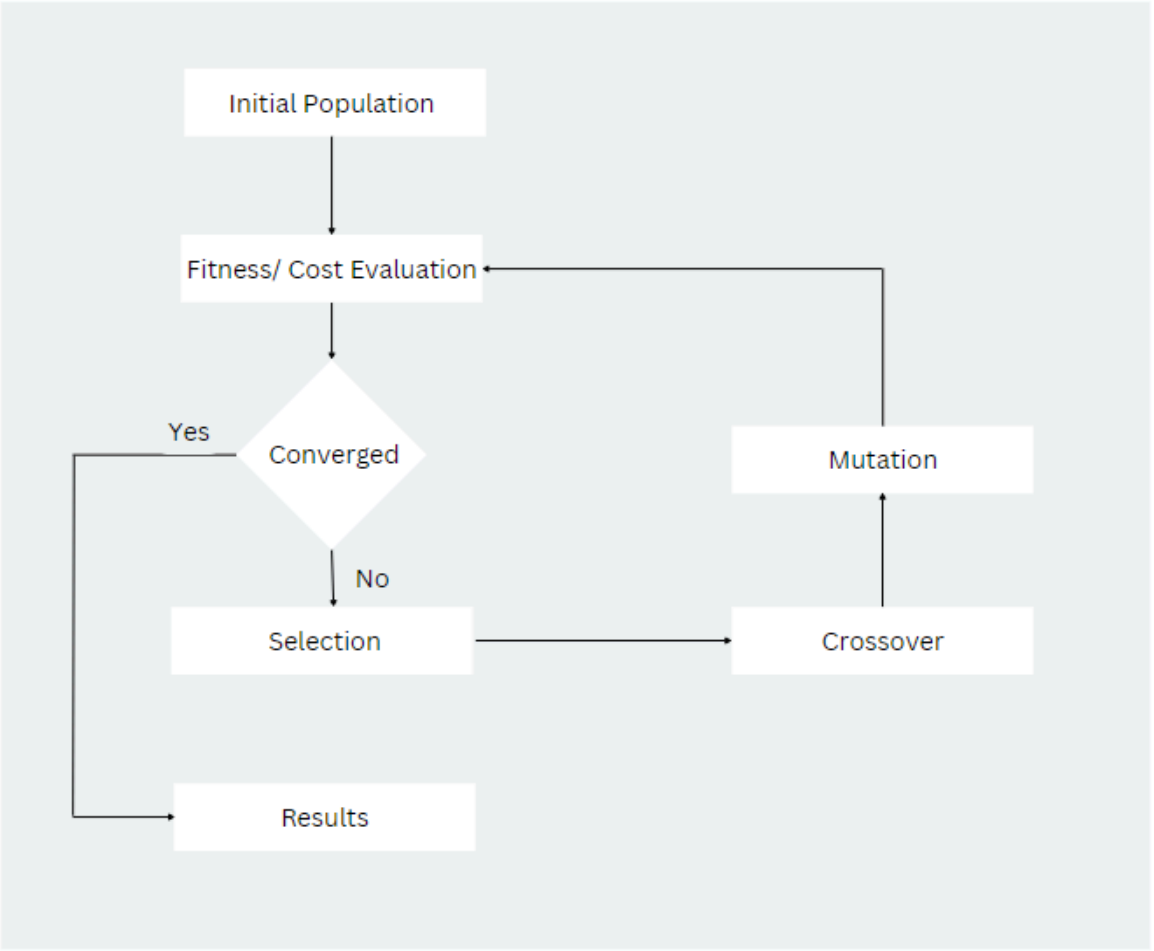
To calculate the fitness of the individuals, the RMSE was used. The 80 percent of training data was used for the LSTM variant and 20 percent used for the validation. The RMSE produced 20 percent of the data used as fitness value. The algorithm is shown below

**Algorithm 2 : Fitness evaluation algorithm**

```
Require : Genotype of the individual, training_data, feature_code_dictionary
  Do Genotype to Phenotype conversion:
        Window_size = INT( BitArray(genotype[0:6]) )
        num_of_units = INT( BitArray(genotype[6:13]) )
        Features_code = INT( BitArray(genotype[13:18]) )
  If window_size == 0 or num_units == 0 or feature_code == 0:
        Return INT_MAX
  Selected_features = list()
  For item in feature_code_dictionary :
        if feature_code && item.key():
                selected _features.append( item.value() )
```

**Train** lstm_model(window_size, num_of)units, selected_features, 80% of training data)
RMSE = LSTM_model.predict( 20% training data)
**Return** RMSE

**Diagram Flow of Genetic Algorithm:**




Genetic Algorithm

**Figure 4 : Genetic Algorithm**

1. Encoding the solution (generation of genotypes)
2. Initialising the initial population
3. Fitness evaluation of individuals
4. Selection of individuals for reproduction
5. Creation of new generation using crossover and mutation
6. Repeating iterations till convergence;

**Diagram Flow of Fitness/Cost Evaluation:**

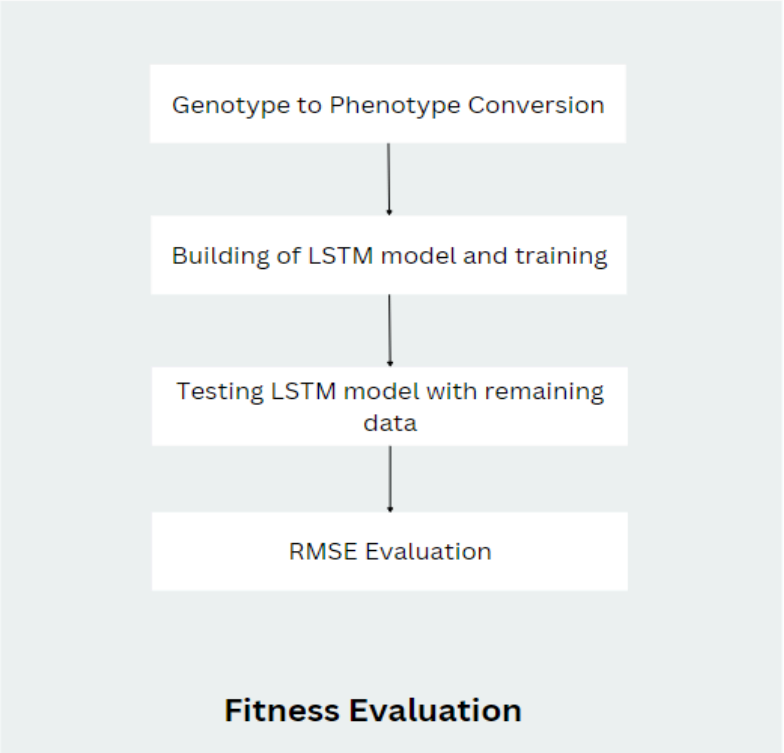
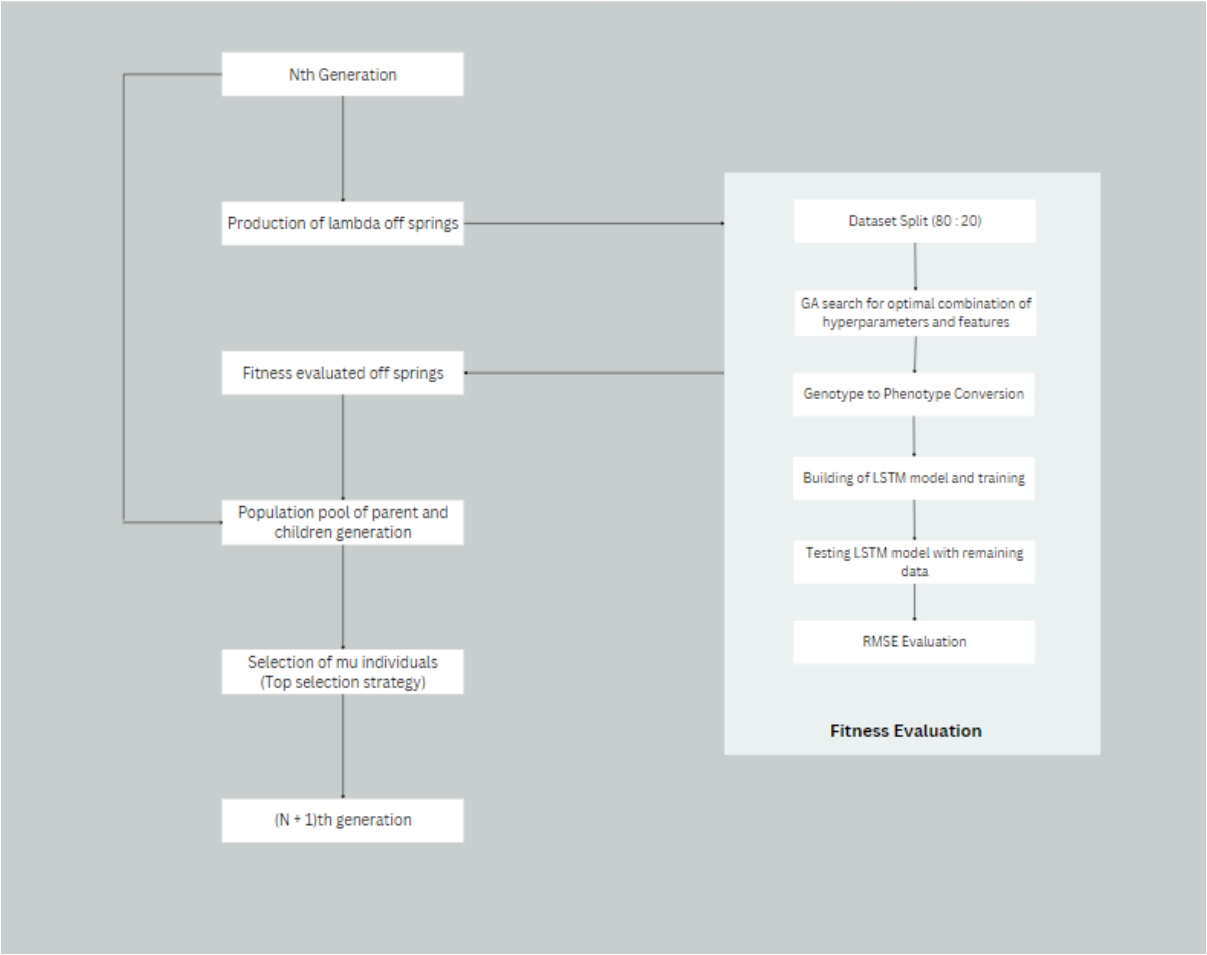



**FIGURE 5 : Fitness Evaluation**




# 8.Results

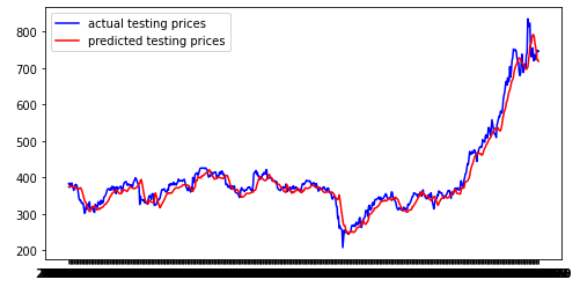
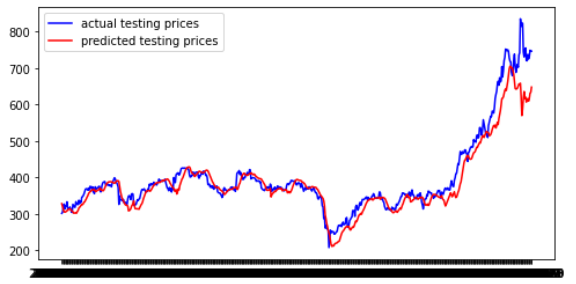
## 8.1 Without K fold:

### Classical

| Adani Ports without K Fold Classical | | |
|---|---|---|
| | Test Root Mean Squared Error: | Time taken |
| Without GA | 28.63686711 | 15.32555151 |
| Base case parameters | | |
| Window size | Number of units | |
| 30 | 30 | |
| | Test Root Mean Squared Error | Time Taken |
| With GA | 17.98899191 | 1405.623412 |

| GA Parameter | | |
|---|---|---|
| Best Window size | Best Number of units | Best Feature Code |
| 8 | 11 | 22 |

For classic LSTM ,the window size 30 and number of units 30 are taken as base case parameters. Without the genetic algorithm we obtained 28.63 as root mean square error. For classic LSTM, the best individual produced by Genetic Algorithm gives a time window size of 8 days with the number of units being 11 and the best feature code is 22. The RMSE value observed is 17.988. So, the genetic algorithm improved the RMSE. It can be seen in the figure given below.

| With GA | Without GA |
|---|---|
|  |  |

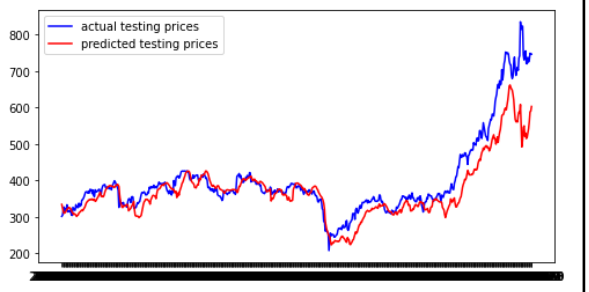


## Bidirectional

| Adani Ports without K Fold Bidirectional | | |
|---|---|---|
| | Test Root Mean Squared Error: | Time taken |
| Without GA | 54.22322573 | 8.779038906 |
| Base case parameters | | |
| Window size | Number of units | |
| 30 | 30 | |
| | Test Root Mean Squared Error | Time Taken |
| With GA | 22.03828247 | 2151.121552 |
| GA Parameter | | |
| Best Window size | Best Number of units | Best Feature Code |
| 61 | 42 | 28 |

The window size of 30 and the number of units that are used for bidirectional LSTM are both considered to be basic case characteristics. In the absence of the genetic algorithm, we were able to calculate the root mean square error as 54.22.

The best possible person generated by the Genetic Algorithm has a temporal window size of 61 days, a total of 42 units, and a feature code of 28. This is the finest possible combination. The value of the RMSE that was found is 22.038. Therefore, the RMSE is better now thanks to the genetic algorithm. The figure that follows demonstrates this point clearly.

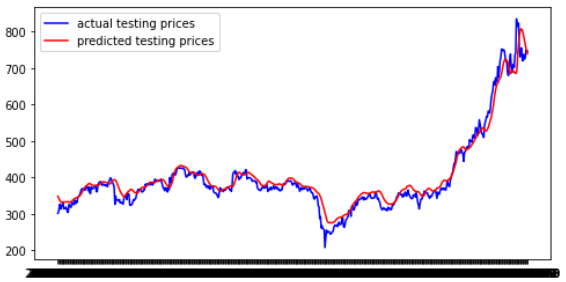
| With GA | Without GA |
| --- | --- |
|  |  |



## Stacked

| Adani Ports without K Fold Stacked | | |
| --- | --- | --- |
| | Test Root Mean Squared Error: | Time taken |
| Without GA | 22.61751359 | 14.73648834 |
| Base case parameters | | |
| Window size | Number of units | |
| 30 | 30 | |
| | Test Root Mean Squared Error | Time Taken |
| With GA | 17.76085635 | 1880.907345 |
| GA Parameter | | |
| Best Window size | Best Number of units | Best Feature Code |
| 1 | 87 | 14 |

The window size of 30 and the number of units that are used in stacked LSTM are both considered to be basic case characteristics. In the absence of the genetic algorithm, we were able to calculate the root mean square error as 22.617.
The greatest individual that the Genetic Algorithm has created for traditional LSTM has a time window size of 1 day, the number of units is 87, and the best feature code is 14. The value of RMSE that was found is 17.761. Therefore, the RMSE is better now thanks to the genetic algorithm. The figure that follows demonstrates this point clearly.

| With GA | Without GA |
|---|---|
|  |  |



## 8.2 With K fold :

### Classical

| Adani Ports With K-Fold Classical | | |
|---|---|---|
| | Test Root Mean Squared Error: | Time taken (Seconds) |
| Without GA | 54.62915901 | 42.896818 |
| Base Case Parameter | | |
| Window Size: | Number of Units: | |
| 30 | 30 | |
| | Test Root Mean Squared Error: | Time taken (Seconds) |
| With GA | 17.71359183 | 8618.100413 |
| GA Parameter | | |
| Best Window Size: | Best Number of Units: | Best Features Code: |
| 63 | 126 | 28 |

For classic LSTM, the window size 30 and number of units 30 are taken as base case parameters with the extra validation of K Fold. Without the genetic algorithm we obtained 54.63 as root mean square error.
For classic LSTM, the best individual produced by Genetic Algorithm and K Flod validation gives a time window size of 63 days with the number of units being 126 and the best feature code is 28. The RMSE value observed is 17.71. So, the genetic algorithm improved the RMSE. It can be seen in the figure given below.

| With GA | Without GA |
|---|---|
|  |  |

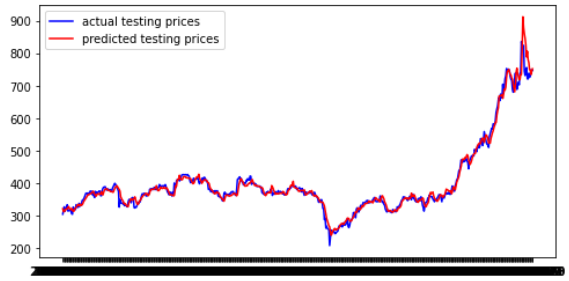


## Bidirectional

| Adani Ports With K-Fold Bidirectional | | |
|---|---|---|
| | Test Root Mean Squared Error: | Time taken (Seconds) |
| Without GA | 29.81426064 | 69.89442729 |
| Base Case Parameter | | |
| Window Size: | Number of Units: | |
| 30 | 30 | |
| | Test Root Mean Squared Error: | Time taken (Seconds) |
| With GA | 21.71415274 | 10384.73271 |
| GA Parameter | | |
| Best Window Size: | Best Number of Units: | Best Features Code: |
| 18 | 123 | 13 |

For bidirectional LSTM, the window size 30 and number of units 30 are taken as base case parameters with K Fold Validation. Without the genetic algorithm we obtained 29.82 as root mean square error.

The best individual is produced by Genetic Algorithm and K Flod validation gives a time window size of 18 days with the number of units being 123 and the best feature code is 13. The RMSE value observed is 21.71. So, the genetic algorithm improved the RMSE. It can be seen in the figure given below.

| With GA | Without GA |
|---|---|
|  |  |



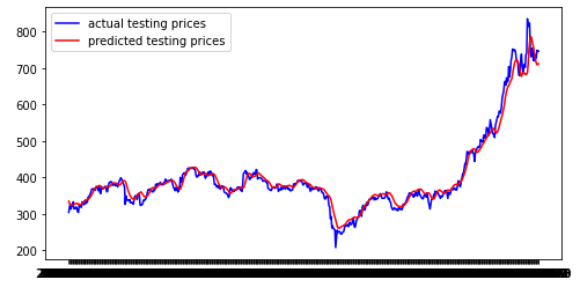
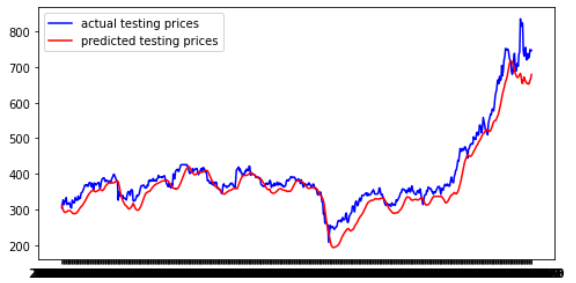


## Stacked

| Adani Ports With K-Fold Stacked | | |
|---|---|---|
| | Test Root Mean Squared Error: | Time taken (Seconds) |
| Without GA | 43.6924137 | 60.498612 |
| Base Case Parameter | | |
| Window Size: | Number of Units: | |
| 30 | 30 | |
| | Test Root Mean Squared Error: | Time taken (Seconds) |
| With GA | 21.29268533 | 11844.56959 |
| GA Parameter | | |
| Best Window Size: | Best Number of Units: | Best Features Code: |
| 11 | 51 | 15 |

For stacked LSTM, the window size 30 and number of units 30 are taken as base case parameters and K Fold validation. Without the genetic algorithm we obtained 43.69 as root mean square error.
For classic LSTM, the best individual is produced by Genetic Algorithm and K Flod validation gives a time window size of 11 days with the number of units being 51 and the best feature

code is 15. The RMSE value observed is 21.29. So, the genetic algorithm improved the RMSE. It can be seen in the figure given below.

| With GA | Without GA |
| --- | --- |
|  |  |



# 9. Conclusion

The genetic algorithm was successfully used in order to effectively optimize the LSTM and its variations, which ultimately resulted in superior performance. The performance of the models was improved as a result of the genetic algorithm. The stacked and conventional LSTM models fared the best in the absence of k-fold validation, with a root mean square of 17.76 after optimization. The performance of the traditional model was greater than that of the bidirectional model, but it was not quite as good as the stacked model. The traditional LSTM models had the greatest results in the k-fold validation, with a root mean square score of 17.71 after being optimized. Stacking models performed far better than bidirectional models.

In light of the contrast between the two distinct models of assessment, one with and one without K Fold. It was noted that we were not able to significantly lower the RMSE error values, and when it came to the amount of time required to carry out the procedure, it took an average of five times longer to do so using K Fold validation. Which, in turn, was not the optimal situation given that the K Fold was supposed to converge in a less number of generations. Based on the findings of this study, we are able to draw the following conclusion about genetic algorithms: it is dependent on the amount of ML used as well as the amount of time required to produce the complexity. As a result, K Fold we divided it in five folds, which resulted in the execution taking five times as long. If we had been able to parallelize the process, we would have significantly improved our results in terms of the time complexity, as all folds would have been completed concurrently, resulting in a reduction in complexity.

# 10. Future Scope

With the dilemma of K Fold taking substantial time. What we can do is by parallelizing the K Fold operations with Genetic Algorithm to make use of GPUs and execute them with more efficiency.

# 11. References


[1] B. P. Mukhoty, V. Maurya and S. K. Shukla, "Sequence to sequence deep learning models for solar irradiation forecasting," 2019 IEEE Milan PowerTech, 2019, pp. 1-6, doi: 10.1109/PTC.2019.8810645.

[2] Jui-Yu Wu, You-Ting, “Applying a Long Short-Term Memory Approach to a Chaotic Time Series Problem – A Case Study”. The 35th Annual Conference of the Japanese Society for Artificial Intelligence, 2021.

[3] S. O. Ojo, P. A. Owolawi, M. Mphahlele and J. A. Adisa, "Stock Market Behaviour Prediction using Stacked LSTM Networks," 2019 International Multidisciplinary Information Technology and Engineering Conference (IMITEC), 2019, pp. 1-5, doi: 10.1109/IMITEC45504.2019.9015840.

[4] S. Siami-Namini, N. Tavakoli and A. S. Namin, "The Performance of LSTM and BiLSTM in Forecasting Time Series," 2019 IEEE International Conference on Big Data (Big Data), 2019, pp. 3285-3292, doi: 10.1109/BigData47090.2019.9005997.

[5] Anh, N.N., Anh, N.H.Q., Tung, N.X., Anh, N.T.N. (2021). Feature Selection Using Genetic Algorithm and Bayesian Hyper-parameter Optimization for LSTM in Short-Term Load Forecasting. In: Tran, DT., Jeon, G., Nguyen, T.D.L., Lu, J., Xuan, TD. (eds) Intelligent Systems and Networks . ICISN 2021. Lecture Notes in Networks and Systems, vol 243. Springer, Singapore. https://doi.org/10.1007/978-981-16-2094-2_9

[6] Chiara Di Francescomarino, Marlon Dumas, Marco Federici, Chiara Ghidini, Fabrizio Maria Maggi, Williams Rizzi, Luca Simonetto, Genetic algorithms for hyperparameter optimization in predictive business process monitoring, Information Systems, Volume 74, Part 1, 2018, Pages 67-83, ISSN 0306-4379, https://doi.org/10.1016/j.is.2018.01.003 . (https://www.sciencedirect.com/science/article/pii/S0306437916305695

[7] Zhang J, Qu S, Zhang Z, Cheng S. Improved genetic algorithm optimized LSTM model and its application in short-term traffic flow prediction. PeerJ Comput Sci. 2022 Jul 19;8:e1048. doi: 10.7717/peerj-cs.1048. PMID: 36091988; PMCID: PMC9454874.

[8] Gorgolis, Nikolaos & Hatzilygeroudis, Ioannis & Istenes, Zoltan & Grad-Gyenge, László. (2019). Hyperparameter Optimization of LSTM Network Models through Genetic Algorithm. 1-4. 10.1109/IISA.2019.8900675.

[9] Chung, Hyejung & Shin, Kyung-shik. (2018). Genetic Algorithm-Optimized Long Short-Term Memory Network for Stock Market Prediction. Sustainability. 10. 3765. 10.3390/su10103765.

[10] Anh, N.N., Anh, N.H.Q., Tung, N.X., Anh, N.T.N.: Feature selection using genetic algorithm and bayesian hyper-parameter optimization for 1stm in short-term load forecasting. In: Tran, D.-T., Jeon, G., Nguyen, T.D.L., Lu, J., Xuan, T.-D. (eds.) Intelligent Systems and Networks, pp. 69-79. Springer, Singapore (2021)

[11] F. H. F. Leung, H. K. Lam, S. H. Ling and P. K. S. Tam, "Tuning of the structure and parameters of a neural network using an improved genetic algorithm," in *IEEE Transactions on Neural Networks*, vol. 14, no. 1, pp. 79-88, Jan. 2021, doi: 10.1109/TNN.2002.804317.

[12] N. Gorgolis, I. Hatzilygeroudis, Z. Istenes and L. –. G. Gyenne, "Hyperparameter Optimization of LSTM Network Models through Genetic Algorithm," 2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA), 2019, pp. 1-4, doi: 10.1109/IISA.2019.8900675.

[13] Mehmet Ozcalici, Mete Bumin, Optimizing filter rule parameters with genetic algorithm and stock selection with artificial neural networks for an improved trading: The case of Borsa Istanbul, Expert Systems with Applications, Volume 208, 2022, 118120, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2022.118120.

[14] Chung, H.; Shin, K.-s. Genetic Algorithm-Optimized Long Short-Term Memory Network for Stock Market Prediction. *Sustainability* **2018**, *10*, 3765. https://doi.org/10.3390/su10103765

[15] Zeynep Idil Erzurum Cicek, Zehra Kamisli Ozturk, Optimizing the artificial neural network parameters using a biased random key genetic algorithm for time series forecasting, Applied Soft Computing, Volume 102, 2021, 107091, ISSN 1568-4946, https://doi.org/10.1016/j.asoc.2021.107091.

[16] Kumar, R., Kumar, P. & Kumar, Y. Integrating big data driven sentiments polarity and ABC-optimized LSTM for time series forecasting. *Multimed Tools Appl* 81, 34595–34614 (2022). https://doi.org/10.1007/s11042-021-11029-1

[17] Shahvaroughi Farahani, M., Razavi Hajiagha, S.H. Forecasting stock price using integrated artificial neural network and metaheuristic algorithms compared to time series models. *Soft Comput* 25, 8483–8513 (2021). https://doi.org/10.1007/s00500-021-05775-5