**Stock Price Prediction using LSTM & ANN**

1. **INTRODUCTION**

The share market is a place where the shares of a public company are traded. As discussed in the volatile nature of the stock market makes it an area which needs an abundance of analysis with the old data predicated. The previous stock trend prediction algorithms use the historic time series stock data. the typical scientific stock price forecasting procedures are focused on the statistical analysis of stock data. In the paper will develop a stock data predictor program that uses previous stock prices and data will be treated as training sets for the program to predict the stock prices of a particular share this program develops a procedure. This model considers the historical equity share price of a company price and applies Artificial Neural Networks (ANN) technique called Long Short Term Memory (LSTM). The proposed approach considers available historic data of a share and it provides prediction on a particular feature. The features of shares are Opening price, day High, day Low, previous day o price, Close price, Date of trading, Total Trade Quantity and Turnover. The proposed model uses the time series analysis in order to predict a share price for a required time span. the proposed will be considering Indian stock exchange Company named as The National Stock Exchange of India Limited (NSE).The National Stock Exchange (NSE) is the Indian stock exchange entity, the NSE was the first exchange in India to provide a modern, provides latest facility to the investors spread across the length and breadth of the country. It has thoroughly modern with all latest facilities, which provides investors with the facility to trade from anywhere in India. This has a decisive role in reforming the Indian equity market to add increased transparency, convergence and efficiency to the capital market. NSE's Common Index, The CNX NIFTY, is used prodigiously by the investor across India as well as globally. It provides accommodation for the exchange, settlement and clearing in equity and debt market and additionally in derivatives. This is one of India's most astronomically enormous mazuma, currency and index options trading exchanges worldwide. There are numerous domestic and ecumenical companies which have an interest in the exchange. Several regional companies include TATA, WIPRO, HDFC and YES BANK ltd. Among pilgrim investors, few are strategic holdings of the city party, Mauritius limited, Tiger Ecumenical five holdings. As suggested by The Long Short Term Memory (LSTM) networks are a type of ANN capable of addressing linear problems. LSTM is a deep learning technique. Long-term Memory (LSTM) Units are enforced to learn very long sequences. This is a more general version of the gated recurrent system. LSTM is more benign than other deep learning methods

**1.1 Objective of the project:**

The prediction of stock value is a complex task which needs a robust algorithm background in order to compute the longer term share prices. Stock prices are correlated within the nature of market; hence it will be difficult to predict the costs. The proposed algorithm using the market data to predict the share price using machine learning techniques Artificial Neural Networks, Long Short Term Memory, in that process weights are corrected for each data points using stochastic gradient descent. This system will provide accurate outcomes in comparison to currently available stock price predictor algorithms. The network is trained and evaluated with various sizes of input data to urge the graphical outcomes.

**2. LITERATURE SURVEY:**

**“Stock Price Prediction Using LSTM, RNN and CNN-SLIDING WINDOW MODEL”**

Stock market or equity markets have a profound impact in today's economy. A rise or fall in the share price has an important role in determining the investor's gain. The existing forecasting methods make use of both linear (AR, MA, ARIMA) and non-linear algorithms (ARCH, GARCH, Neural Networks), but they focus on predicting the stock index movement or price forecasting for a single company using the daily closing price. The proposed method is a model independent approach. Here we are not fitting the data to a specific model, rather we are identifying the latent dynamics existing in the data using deep learning architectures. In this work we use three different deep learning architectures for the price prediction of NSE listed companies and compare their performance. We are applying a sliding window approach for predicting future values on a short term basis. The performance of the models was quantified using percentage error.

**“Short Term Stock Price Prediction Using Deep Learning”**

Short - term price movements; contribute a considerable measure to the unpredictability of the securities exchanges. Accurately predicting the price fluctuations in stock market is a huge economical advantage. The aforementioned task is generally achieved by analyzing the company, this is called as fundamental analysis. Another method, which is undergoing a lot of research work recently, is to create a predictive algorithmic model using machine learning. To train machines to take trading decisions in such short - period of time, the latter method needs to be adopted. Deep Neural Networks, being the most exceptional innovation in Machine Learning, have been utilized to develop a short-term prediction model. This paper plans to forecast these short - term prices of stocks. 10 unique stocks recorded on New York Stock Exchange are considered for this review. The review essentially focuses on the prediction of these short - term prices leveraging the power of technical analysis. Technical Analysis guides the framework to understand the patterns from the historical prices fed into it, and attempts to probabilistically forecast the fleeting future prices of the stock under review. The paper discusses about two distinct sorts of Artificial Neural Networks, Feed Forward Neural Networks and Recurrent Neural Networks. The review uncovers that Feed Forwards Multilayer Perceptron perform superior to Long Short-Term Memory, at predicting the short - term prices of a stock.

**“Time series prediction of stock price using deep belief networks with Intrinsic Plasticity”**

In recent years, the stock market plays an important role, which has attracted more and more attentions. The key problem of the stock market prediction is how to design a method to improve the prediction performance. As we know, the biggest challenge is that the stock time series is essentially dynamic, nonlinear, complicated, nonparametric and chaotic. In this paper, we propose a novel method to predict the stock closing price based on the deep belief networks (DBNs) with intrinsic plasticity. In the experiments, the stock in S&P 500 is used to examine the performance. The back propagation algorithm is used for output training to make minor adjustments of structure parameters. The intrinsic plasticity (IP) is also applied into the network to make it have adaptive ability. It is believed that IP learning for adaptive adjustment of neuronal response to external inputs is beneficial for maximizing the input-output mutual information. Our results show that the application of IP learning can remarkably improve the prediction performance. Moreover, the effects of two kinds of IP rules on the performance of prediction are examined. Compared with Triesch's IP and without IP, DBN with Li's IP learning has much better prediction performance than the others. These results may have important implications on the modeling of neural network for complex time series prediction.

**“Sentiment-Aware Stock Market Prediction: A Deep Learning Method”**

Stock market prediction has attracted much attention from academia as well as business. However, it is a challenging research topic, in which many advanced computational methods have been proposed, but not yet attained a desirable and reliable performance. This study proposes a new method for stock market prediction, which adopts the Long Short-Term Memory (LSTM) neural network and incorporates investor sentiment and market factors to improve forecasting performance. By extracting investor sentiment from forum posts using Naïve Bayes, this paper makes it possible to analyze the irrational component of stock price. Our empirical study on CSI300 index proves that our prediction method provides better prediction performance. It gives a prediction accuracy of 87.86%, outperforming other benchmark models by at least 6%. Furthermore, our empirical study reveals evidence that helps to better understand investor sentiment and stock behaviors. Finally, this work shows the potential of deep learning financial time series in the presence of strong noises.

**“Predicting Stock Prices Using LSTM”**

The art of forecasting the stock prices has been a difficult task for many of the researchers and analysts. In fact, investors are highly interested in the research area of stock price prediction. For a good and successful investment, many investors are keen in knowing the future situation of the stock market. Good and effective prediction systems for stock market help traders, investors, and analyst by providing supportive information like the future direction of the stock market. In this work, we present a recurrent neural network (RNN) and Long Short-Term Memory (LSTM) approach to predict stock market indices.

**“Stock Price Prediction Using Regression And Aritificial Neural Network”,**

Prediction of stock market has been an attractive topic to the stockbrokers. In stock market the decision on when buying or selling stock is important in order to achieve profit. There are number of techniques that can be used to help investors in order to make a decision for financial gain. In this research work we have proposed a prediction algorithm that will give the relation between the dependent factor like price and independent factors like opening price, closing price, high value of stock, low value of stock and volume of stocks bought. In this research, we have explained development of stock price prediction with the use of deep learning algorithm. In this work, we are going to use different deep learning architecture for the price prediction of BSE listed company and compares their performance. Here we had used LSTM (Long short-term memory)and RNN (Recurrent neural network) algorithms. We had shown comparative study of this two deep learning algorithm. Study shows that RNN gives better performance than LSTM. Accuracy of LSTM is 87% and accuracy of RNN is 89%.

**“Deep learning for stock prediction using numerical and textual information”**

This paper proposes a novel application of deep learning models, Paragraph Vector, and Long Short-Term Memory (LSTM), to financial time series forecasting. Investors make decisions according to various factors, including consumer price index, price-earnings ratio, and miscellaneous events reported in newspapers. In order to assist their decisions in a timely manner, many automatic ways to analyze those information have been proposed in the last decade. However, many of them used either numerical or textual information, but not both for a single company. In this paper, we propose an approach that converts newspaper articles into their distributed representations via Paragraph Vector and models the temporal effects of past events on opening prices about multiple companies with LSTM. The performance of the proposed approach is demonstrated on real-world data of fifty companies listed on Tokyo Stock Exchange.

**“Comparative Study of Stock Prediction System using Regression Techniques”**

There is a creating enthusiasm for the prediction and analysis of stock prices to augment the profits for the end client. This figuring will give an inexact estimation whether the stock price will increment or abatement sooner rather than later. To play out this prediction we will utilize information mining algorithms like linear regression, support vector regression (SVR) and ridge regression. The method will likewise give a comparative study of these three algorithms on the premise of their precision in prediction of the stocks. The prediction system alongside the information of fundamental and technical analysis can prompt an extremely precise prediction, which will bring about exceptional yields. The system can be utilized by traders, typical clients, brokers and so on.

**“Deep learning for stock prediction using numerical and textual information”**

This paper proposes a novel application of deep learning models, Paragraph Vector, and Long Short-Term Memory (LSTM), to financial time series forecasting. Investors make decisions according to various factors, including consumer price index, price-earnings ratio, and miscellaneous events reported in newspapers. In order to assist their decisions in a timely manner, many automatic ways to analyze those information have been proposed in the last decade. However, many of them used either numerical or textual information, but not both for a single company. In this paper, we propose an approach that converts newspaper articles into their distributed representations via Paragraph Vector and models the temporal effects of past events on opening prices about multiple companies with LSTM. The performance of the proposed approach is demonstrated on real-world data of fifty companies listed on Tokyo Stock Exchange.

**“Comparative Study of Stock Prediction System using Regression Techniques”**

There is a creating enthusiasm for the prediction and analysis of stock prices to augment the profits for the end client. This figuring will give an inexact estimation whether the stock price will increment or abatement sooner rather than later. To play out this prediction we will utilizeinformation mining algorithms like linear regression, support vector regression(SVR) and ridge regression. The method will likewise give a comparative study of these three algorithms on the premise of their precision in prediction of the stocks. The prediction system alongside the information of fundamental and technical analysis can prompt an extremely precise prediction, which will bring about exceptional yields. The system can be utilized by traders, typical clients, brokers and so on.

**“Stock Trend Prediction Using Regression Analysis – A Data Mining Approach”**

Organizations have been collecting data for decades, building massive data warehouses in which to store the data. Even though this data is available, very few of these organizations have been able to realize the actual value stored in it. The question these organizations are asking is how to extract meaningful data and uncover patterns and relationship from their databases. This paper presents a study of regression analysis for use in stock price prediction. Data were obtained from the daily official list of the prices of all shares traded on the stock exchange published by the Nigerian Stock Exchange using banking sector of Nigerian economy with three banks namely:-First Bank of Nigeria Plc, Zenith Bank Plc, and Skye Bank Plc to build a database. A data mining software tool was used to uncover patterns and relationships and also to extract values of variables from the database to predict the future values of other variables through the use of time series data that employed moving average method. The tools were found capable technique to describe the trends of stock market prices and predict the future stock market prices of three banks sampled.

**“A Comparative Study on Financial Stock Market Prediction Models”**

Now a day’s investors invest money in financial products. Financial operations are not local but wide-ranging to all the countries in the world. Stock Market is the market for security where organized issuance and trading of Stocks take place either through trading or over the counter in electronic or physical form. In stock markets, many guiding principle such as price limits are made to get involved between the financial operations so that a volatility of stock prices is more uncertain than one without disturbance. Investment in the financial market is one of the best ways to obtain high rewards, but it is also a great risk among many investments. The financial market is considered as a high complex and dynamic system. So the price prediction is one of the most important issues to be investigated by researchers. The objective of the proposed paper is to do study, improvement in the machine learning approaches to predict the financial products. For financial market prediction different approaches like Artificial Intelligence, Machine Learning Techniques and various data mining techniques are used.

**3. SYSTEM ANALYSIS**

**3.1 Existing System**

The share market is a place where the shares of a public company are traded. As discussed in [7] the volatile nature of the stock market makes it an area which needs an abundance of analysis with the old data predicated. The previous stock trend prediction algorithms use the historic time series stock data. the typical scientific stock price forecasting procedures are focused on the statistical analysis of stock data.

**Disadvantages of Existing System:**

* Less Accuracy

**3.2 Proposed System**

In this project we are using LSTM and ANN algorithms to predict stock prices and we trained both algorithm in different stock prices such as TATA, TESLA, Facebook, Apple etc. In both algorithm ANN is giving high accuracy and less Mean Square Error (MSE).We have experiment both algorithms on above mention different stock prices and both algorithm showing close and accurate prices of original test data. We have splitted dataset into train & test where application using 80% dataset size for training and 20% for testing

**Advantages of Proposed System:**

* High Accuracy.

**Modules Information:**

To implement this project we have used same dataset given in your requirement file and to implement this project we have designed following modules

1. Upload Stock Price Dataset: using this module we will select STOCK Name and then upload dataset and application will extract all records from selected stock name and then sort dataset in ascending date order and then plot stock price history graph
2. Preprocess Dataset: using this module we will remove missing values and then normalize stock values using MINMAX SCALER algorithm and then split dataset into train and test. 80% dataset using for training and 20% for testing
3. Run ANN Algorithm: using this module we will input 80% dataset to ANN algorithm to train a model and this model will be applied on 20% test data and then calculate difference between actual stock test price and predicted price as MSE error.
4. Run LSTM Algorithm: using this module we will input 80% dataset to LSTM algorithm to train a model and this model will be applied on 20% test data and then calculate difference between actual stock test price and predicted price as MSE error.
5. MSE Comparison Graph: using this module we will plot MSE comparison graph between both algorithms

**FUNCTIONAL REQUIREMENTS:**

**SOFTWARE REQIREMENTS:**

**System Atributes:**

1. Filename
2. dataset
3. X, Y, mse, X\_train, X\_test, y\_train, y\_test

**Data base Requirements:**

No need

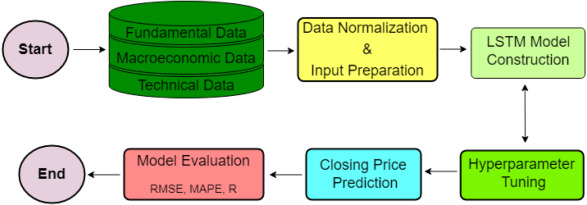
**USECASE:**

* Use cases - Use cases describe the interaction between the system and external users that leads to achieving particular goals.
* Each use case includes main elements:

1. Upload Stock Price Dataset
2. Preprocess Dataset
3. Run ANN Algorithm
4. Run LSTM Algorithm
5. MSE Comparison Graph

**User Stories:** we can see LSTM predicted prices with MSE and accuracy values and we can see LSTM MSE is little higher than ANN but its predicted also too close as LSTM graph values are also overlapping with TEST prices.

**Work down Structure:**



**Prototype:**

python 3.7.0 or 3.7.4

opencv-python==4.5.1.48

keras==2.3.1

tensorflow==1.14.0

protobuf==3.16.0

h5py==2.10.0

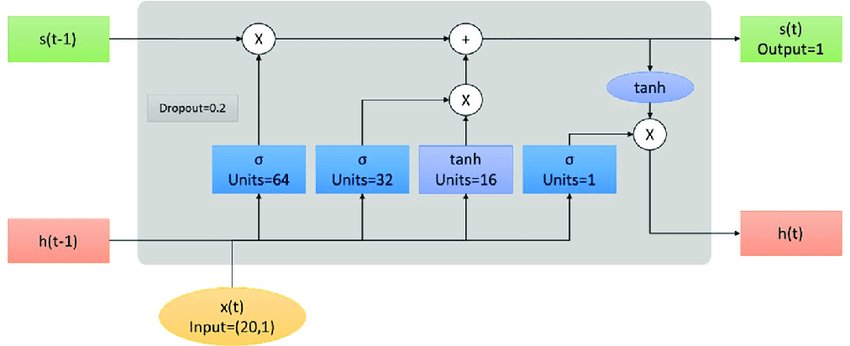
sklearn-extensions==0.0.2

scikit-learn==0.22.2.post1

Numpy

Pandas

**Models and Diagrams:**



**NON-FUNCTIONAL REQUIREMENT:**

**Usability:**  Usability is a quality attribute that assesses how easy user interfaces are to use. The word "usability" also refers to methods for improving ease-of-use during the design process.(how it was handle entire project easy)

**Security:** the quality or state of being secure: such as. a : freedom from danger : safety. b : freedom from fear or anxiety. c : freedom from the prospect of being laid off job security.

**Readability:** Readability is the ease with which a reader can understand a written text.

**Performance**: the execution of an action. : something accomplished : deed, feat. : the fulfillment of a claim, promise, or request : implementation. 3. : the action of representing a character in a play.

**Availability**: the quality or state of being available trying to improve the availability of affordable housing. 2 : an available person or thing.

**Scalability**: Scalability is the measure of a system's ability to increase or decrease in performance and cost in response to changes in application and system processing demands.

**3.3. PROCESS MODEL USED WITH JUSTIFICATION**

**SDLC (Umbrella Model):**

**Umbrella Activity**

**Umbrella Activity**

**Umbrella Activity**

1. Feasibility Study
2. TEAM FORMATION
3. Project Specification PREPARATION

Business Requirement Documentation

ANALYSIS & DESIGN

CODE

UNIT TEST

DOCUMENT CONTROL

ASSESSMENT

TRAINING

INTEGRATION & SYSTEM TESTING

DELIVERY/INSTALLATION

ACCEPTANCE TEST

Requirements Gathering

SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

**Stages in SDLC:**

* Requirement Gathering
* Analysis
* Designing
* Coding
* Testing
* Maintenance

**Requirements Gathering** **stage:**

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.



These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

* Feasibility study is all about identification of problems in a project.
* No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
* Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

**Analysis Stage:**

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.



The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

**Designing Stage:**

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.

  
When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

**Development (Coding) Stage:**

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.



The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

**Integration & Test Stage:**

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.



The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

* **Installation & Acceptance Test:**

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.



The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

**Maintenance:**

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

**3.4. Software Requirement Specification**

**3.4.1. Overall Description**

A Software Requirements Specification (SRS) – a [requirements specification](http://en.wikipedia.org/wiki/Requirements_specification) for a [software system](http://en.wikipedia.org/wiki/Software_system) is a complete description of the behavior of a system to be developed. It includes a set of [use cases](http://en.wikipedia.org/wiki/Use_case) that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. [Nonfunctional requirements](http://en.wikipedia.org/wiki/Non-functional_requirements) are requirements which impose constraints on the design or implementation (such as [performance engineering](http://en.wikipedia.org/wiki/Performance_engineering) requirements, [quality](http://en.wikipedia.org/wiki/Quality_%28business%29) standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A [business analyst](http://en.wikipedia.org/wiki/Business_analyst), sometimes titled [system analyst](http://en.wikipedia.org/wiki/System_analyst), is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the [systems development lifecycle](http://en.wikipedia.org/wiki/Systems_development_life_cycle) domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

* [Business requirements](http://en.wikipedia.org/wiki/Business_requirements) describe in business terms what must be delivered or accomplished to provide value.
* Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
* Process requirements describe activities performed by the developing organization. For instance, process requirements could specify .Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:
* **ECONOMIC FEASIBILITY**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

* **Operational Feasibility**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization’s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

* **TECHNICAL FEASIBILITY**

Earlier no system existed to cater to the needs of ‘Secure Infrastructure Implementation System’. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to .the users. The database’s purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

**3.4.2. External Interface Requirements**

**User Interface**

The user interface of this system is a user friendly python Graphical User Interface.

**Hardware Interfaces**

The interaction between the user and the console is achieved through python capabilities.

**Software Interfaces**

The required software is python.

**SYSTEM REQUIREMENT:**

**HARDWARE REQUIREMENTS:**

# Processor - Intel i3(min)

* Speed - 1.1 GHz
* RAM - 4GB(min)
* Hard Disk - 500 GB
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse
* Monitor - SVGA

**SOFTWARE REQUIREMENTS:**

* Operating System - Windows10(min)
* Programming Language - Python

**4. SYSTEM DESIGN**

**CLASS DIAGRAM:**

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

* The upper part holds the name of the class
* The middle part contains the attributes of the class
* The bottom part gives the methods or operations the class can take or undertake



**USECASE DIAGRAM:**

A **use case diagram** at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as we



**SEQUENCE DIAGRAM**

A **sequence diagram** is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams**, **event scenarios**, and timing diagrams.



**COLLABORATION DIAGRAM:**

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behaviour of a system.



**COMPONENT DIAGRAM:**

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.



**DEPLOYMENT DIAGRAM:**

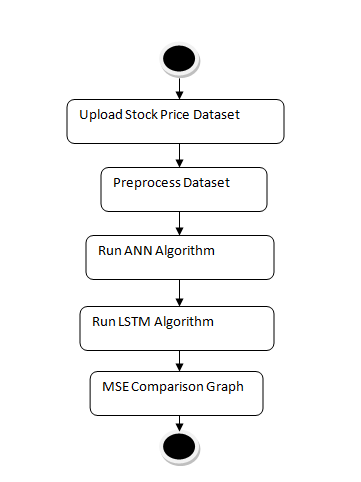
A **deployment diagram** in the Unified Modeling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

****

**ACTIVITY DIAGRAM:**

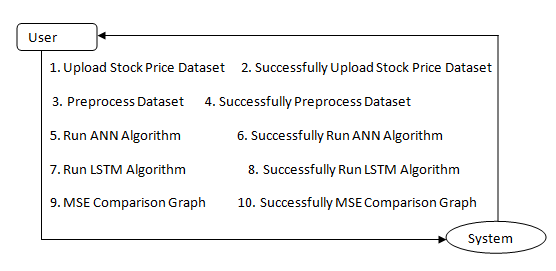
Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent



**Data flow :**

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.



**5. IMPLEMETATION**

**5.1 Python**

Python is a general-purpose language. It has wide range of applications from Web development (like: Django and Bottle), scientific and mathematical computing (Orange, SymPy, NumPy) to desktop graphical user Interfaces (Pygame, Panda3D). The syntax of the language is clean and length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax.

**History of Python:**

Python is a fairly old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

**Why Python was created?**

In late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy-to-understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to design of a new language which was later named Python.

**Why the name Python?**

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from late seventies. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

**Features of Python:**

**A simple language which is easier to learn**

Python has a very simple and elegant syntax. It's much easier to read and write Python programs compared to other languages like: C++, Java, C#. Python makes programming fun and allows you to focus on the solution rather than syntax.

If you are a newbie, it's a great choice to start your journey with Python.

**Free and open-source**

You can freely use and distribute Python, even for commercial use. Not only can you use and distribute software’s written in it, you can even make changes to the Python's source code.

Python has a large community constantly improving it in each iteration.

**Portability**

You can move Python programs from one platform to another, and run it without any changes.

It runs seamlessly on almost all platforms including Windows, Mac OS X and Linux.

**Extensible and Embeddable**

Suppose an application requires high performance. You can easily combine pieces of C/C++ or other languages with Python code.

This will give your application high performance as well as scripting capabilities which other languages may not provide out of the box.

**A high-level, interpreted language**

Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on.

Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower-level operations.

**Large standard libraries to solve common tasks**

Python has a number of standard libraries which makes life of a programmer much easier since you don't have to write all the code yourself. For example: Need to connect MySQL database on a Web server? You can use MySQLdb library using import MySQLdb .

Standard libraries in Python are well tested and used by hundreds of people. So you can be sure that it won't break your application.

**Object-oriented**

Everything in Python is an object. Object oriented programming (OOP) helps you solve a complex problem intuitively.

With OOP, you are able to divide these complex problems into smaller sets by creating objects.

**Applications of Python:**

**1. Simple Elegant Syntax**

Programming in Python is fun. It's easier to understand and write Python code. Why? The syntax feels natural. Take this source code for an example:

a = 2

b = 3

sum = a + b

print(sum)

**2. Not overly strict**

You don't need to define the type of a variable in Python. Also, it's not necessary to add semicolon at the end of the statement.

Python enforces you to follow good practices (like proper indentation). These small things can make learning much easier for beginners.

**3. Expressiveness of the language**

Python allows you to write programs having greater functionality with fewer lines of code. Here's a link to the source code of Tic-tac-toe game with a graphical interface and a smart computer opponent in less than 500 lines of code. This is just an example. You will be amazed how much you can do with Python once you learn the basics.

**4. Great Community and Support**

Python has a large supporting community. There are numerous active forums online which can be handy if you are stuck.

**5.2 Sample Code:**

from tkinter import messagebox

from tkinter import \*

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

import matplotlib.pyplot as plt

from tkinter.filedialog import askopenfilename

import pandas as pd

import numpy as np

import os

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import MinMaxScaler

from sklearn.metrics import mean\_squared\_error

from tkinter import \*

from tkinter import ttk

from keras.models import Sequential

from keras.layers import Dense

from keras.layers import LSTM #class for LSTM regression

from keras.layers import Dropout

from keras.models import model\_from\_json

import pickle

main = Tk()

main.title("Stock Price Prediction using LSTM & ANN")

main.geometry("1300x1200")

global filename

global dataset

global X, Y, mse, X\_train, X\_test, y\_train, y\_test

sc = MinMaxScaler(feature\_range = (0, 1))

global stock\_name, stock\_list

def uploadDataset():

global filename, stock\_name, dataset, stock\_list

text.delete('1.0', END)

filename = askopenfilename(initialdir = "Dataset")

fname = os.path.basename(filename)

stock\_name = stock\_list.get()

if fname == 'NSE-Tata-Global-Beverages-Limited.csv':

dataset = pd.read\_csv(filename,usecols=['Date','Open','High','Low','Close'])

dataset["Date"] = pd.to\_datetime(dataset.Date,format="%Y-%m-%d")

dataset.index = dataset['Date']

dataset = dataset.sort\_index(ascending=True, axis=0)

dataset.fillna(0, inplace = True)

stock\_name = 'NSE-Tata-Global-Beverages-Limited'

else:

dataset = pd.read\_csv(filename,usecols=['Date','Open','High','Low','Close','Stock'])

dataset["Date"] = pd.to\_datetime(dataset.Date,format="%Y-%m-%d")

dataset.index = dataset['Date']

dataset = dataset.sort\_index(ascending=True, axis=0)

dataset.fillna(0, inplace = True)

dataset = dataset.loc[dataset['Stock'] == stock\_name]

tf1.insert(END,str(filename))

text.insert(END,"Dataset Loaded\n\n")

text.insert(END,str(dataset.head()))

tf1.insert(END,str(filename))

plt.figure(figsize=(10,6), dpi=100)

plt.plot(dataset.Date[0:20], dataset.Close[0:20], color='tab:red')

plt.gca().set(title=stock\_name+" Closing Price History", xlabel='Date', ylabel="Closing Price")

plt.show()

def preprocessDataset():

global stock\_name, dataset, sc

global X\_train, X\_test, y\_train, y\_test

text.delete('1.0', END)

dataset = dataset.values

Y = dataset[:,4:5]

X = dataset[:,1:4]

X = sc.fit\_transform(X)

Y = sc.fit\_transform(Y)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, test\_size=0.2)

text.insert(END,"Dataset Preprocessing Completed\n\n")

text.insert(END,"Dataset Normalized Values : "+str(X)+"\n\n")

text.insert(END,"Dataset Train & Test Split. 80% dataset used for training and 20% for testing\n")

text.insert(END,"80% training size : "+str(X\_train.shape[0])+"\n")

text.insert(END,"20% testing size : "+str(X\_test.shape[0])+"\n")

#function to calculate MSE values

def calculateMSE(algorithm, predict, y\_test):

mse\_value = mean\_squared\_error(y\_test,predict)

mse.append(mse\_value)

text.insert(END,algorithm+" MSE value : "+str('{:.6f}'.format(mse\_value))+"\n")

text.insert(END,algorithm+" Accuracy : "+str(1 - mse\_value)+"\n\n")

predict = sc.inverse\_transform(predict)

predict = predict.ravel()

labels = y\_test.reshape(y\_test.shape[0],1)

labels = sc.inverse\_transform(labels)

labels = labels.ravel()

labels = labels[0:100]

predict = predict[0:100]

for i in range(0,20):

text.insert(END,algorithm+" Predicted Stock Price: "+str(predict[i])+" Original Stock Price : "+str(labels[i])+"\n")

#plotting predicted and original RUL values

plt.plot(labels, color = 'red', label = 'Original Stock Price')

plt.plot(predict, color = 'green', label = 'Predicted Stock Price')

plt.title(algorithm+' Predicted Stock Price Graph')

plt.xlabel('Test Data Size')

plt.ylabel('Predicted Stock Price')

plt.legend()

plt.show()

def runANN():

text.delete('1.0', END)

global stock\_name, dataset, sc, mse

global X\_train, X\_test, y\_train, y\_test

mse = []

if os.path.exists('model/ann\_model.json'):

with open('model/ann\_model.json', "r") as json\_file:

loaded\_model\_json = json\_file.read()

ann = model\_from\_json(loaded\_model\_json)

json\_file.close()

ann.load\_weights("model/ann\_model\_weights.h5")

ann.\_make\_predict\_function()

else:

#creating neural object with 50 and 50 neurons and learning rate is 0.005

ann = Sequential()

#defining neural network with 50 neurons

ann.add(Dense(50, activation='relu', input\_shape=(X\_train.shape[1],)))

ann.add(Dense(50, activation='relu'))#50 neurons

ann.add(Dense(1))

ann.compile(optimizer="adam", loss='mean\_squared\_error')

hist = ann.fit(X\_train, y\_train, epochs = 1, batch\_size = 8, validation\_data=(X\_test, y\_test))

ann.save\_weights('model/ann\_model\_weights.h5')

model\_json = ann.to\_json()

with open("model/ann\_model.json", "w") as json\_file:

json\_file.write(model\_json)

json\_file.close()

f = open('model/ann\_history.pckl', 'wb')

pickle.dump(hist.history, f)

f.close()

#performing prediction on test data using ANN

predict = ann.predict(X\_test)

calculateMSE("ANN", predict, y\_test)

def runLSTM():

text.delete('1.0', END)

global stock\_name, dataset, sc, mse

global X\_train, X\_test, y\_train, y\_test

X\_train1 = np.reshape(X\_train, (X\_train.shape[0], X\_train.shape[1], 1))

X\_test1 = np.reshape(X\_test, (X\_test.shape[0], X\_test.shape[1], 1))

if os.path.exists('model/lstm\_model.json'):

with open('model/lstm\_model.json', "r") as json\_file:

loaded\_model\_json = json\_file.read()

lstm = model\_from\_json(loaded\_model\_json)

json\_file.close()

lstm.load\_weights("model/lstm\_model\_weights.h5")

lstm.\_make\_predict\_function()

else:

#training with LSTM algorithm and saving trained model and LSTM refrence assigned to regression variable

lstm = Sequential()

#defining 32 neurons

lstm.add(LSTM(units = 50, return\_sequences = True, input\_shape = (X\_train1.shape[1], X\_train1.shape[2])))

#0.2 as the drop out

lstm.add(Dropout(0.2))

lstm.add(LSTM(units = 50, return\_sequences = True)) #16 another layer neurons

lstm.add(Dropout(0.2))

lstm.add(LSTM(units = 50, return\_sequences = True))

lstm.add(Dropout(0.2))

lstm.add(LSTM(units = 50))

lstm.add(Dropout(0.2))

lstm.add(Dense(units = 1))

lstm.compile(optimizer = 'adam', loss = 'mean\_squared\_error')

hist = lstm.fit(X\_train1, y\_train, epochs = 1, batch\_size = 8, validation\_data=(X\_test1, y\_test))

lstm.save\_weights('model/lstm\_model\_weights.h5')

model\_json = lstm.to\_json()

with open("model/lstm\_model.json", "w") as json\_file:

json\_file.write(model\_json)

json\_file.close()

f = open('model/lstm\_history.pckl', 'wb')

pickle.dump(hist.history, f)

f.close()

#performing prediction on test data

predict = lstm.predict(X\_test1)

calculateMSE("LSTM", predict, y\_test)

def graph():

height = mse

bars = ('ANN MSE','LSTM MSE')

y\_pos = np.arange(len(bars))

plt.bar(y\_pos, height)

plt.xticks(y\_pos, bars)

plt.title("ANN & LSTM Mean Square Error (MSE) Graph Comparison")

plt.show()

def close():

main.destroy()

font = ('times', 15, 'bold')

title = Label(main, text='Stock Price Prediction using LSTM & ANN')

title.config(bg='HotPink4', fg='yellow2')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 13, 'bold')

ff = ('times', 12, 'bold')

l1 = Label(main, text='Choose Dataset:')

l1.config(font=font1)

l1.place(x=430,y=100)

tf1 = Entry(main,width=45)

tf1.config(font=font1)

tf1.place(x=580,y=100)

l2 = Label(main, text='Choose Stock:')

l2.config(font=font1)

l2.place(x=50,y=100)

names = ['AAPL', 'FB', 'MSFT', 'TSLA']

stock\_list = ttk.Combobox(main, values=names, postcommand=lambda: stock\_list.configure(values=names))

stock\_list.place(x=210,y=100)

stock\_list.current(0)

stock\_list.config(font=font1)

uploadButton = Button(main, text="Upload Stock Price Dataset", command=uploadDataset, bg='#ffb3fe')

uploadButton.place(x=1020,y=100)

uploadButton.config(font=font1)

preprocessButton = Button(main, text="Preprocess Dataset", command=preprocessDataset, bg='#ffb3fe')

preprocessButton.place(x=50,y=150)

preprocessButton.config(font=font1)

annButton = Button(main,text="Run ANN Algorithm", command=runANN, bg='#ffb3fe')

annButton.place(x=300,y=150)

annButton.config(font=font1)

lstmButton = Button(main,text="Run LSTM Algorithm", command=runLSTM, bg='#ffb3fe')

lstmButton.place(x=530,y=150)

lstmButton.config(font=font1)

graphButton = Button(main,text="MSE Comparison Graph", command=graph, bg='#ffb3fe')

graphButton.place(x=50,y=200)

graphButton.config(font=font1)

closeButton = Button(main,text="Exit", command=close, bg='#ffb3fe')

closeButton.place(x=300,y=200)

closeButton.config(font=font1)

font1 = ('times', 13, 'bold')

text=Text(main,height=20,width=130)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=10,y=300)

text.config(font=font1)

main.config(bg='plum2')

main.mainloop()

**6. TESTING:**

**Implementation and Testing:**

Implementation is one of the most important tasks in project is the phase in which one has to be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

**Implementation**

The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifies as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

## Testing

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. Actually testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

### System Testing

Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to use the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

**Module Testing**

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

**Integration Testing**

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system.

**Acceptance Testing**

When that user fined no major problems with its accuracy, the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case Id** | **Test Case Name** | **Test Case Desc.** | **Test Steps** | | | | **Test Case Status** | **Test Priority** |
| **Step** | **Expected** | | **Actual** |
| 01 | Upload Stock Price Dataset | Verify Your Stock Price Dataset  Uploaded or not | If Upload Stock Price Dataset may not upload | we cannot do any further operations | we can do further operations | | High | High |
| 02 | Preprocess Dataset | Verify Preprocess Dataset or not | If Dataset may not Preprocess | we cannot do any further operations | we can do further operations | | High | High |
| 03 | Run ANN Algorithm | Verify Run ANN Algorithm or not | If Run ANN Algorithm not be | we cannot do any further operations | we can do further operations | | High | High |
| 04 | Run LSTM Algorithm | Verify Run LSTM Algorithm or not | If LSTM Algorithm not Run | We cannot run  operation | We can Run the Operation | | High | High |
| 05 | MSE Comparison Graph | Verify MSE Comparison Graph or not | If MSE Comparison Graph may not plot | We cannot run  operation | We can Run the Operation | | High | High |

**7. SCREENSHOTS:**

Stock Price Prediction using LSTM & ANN

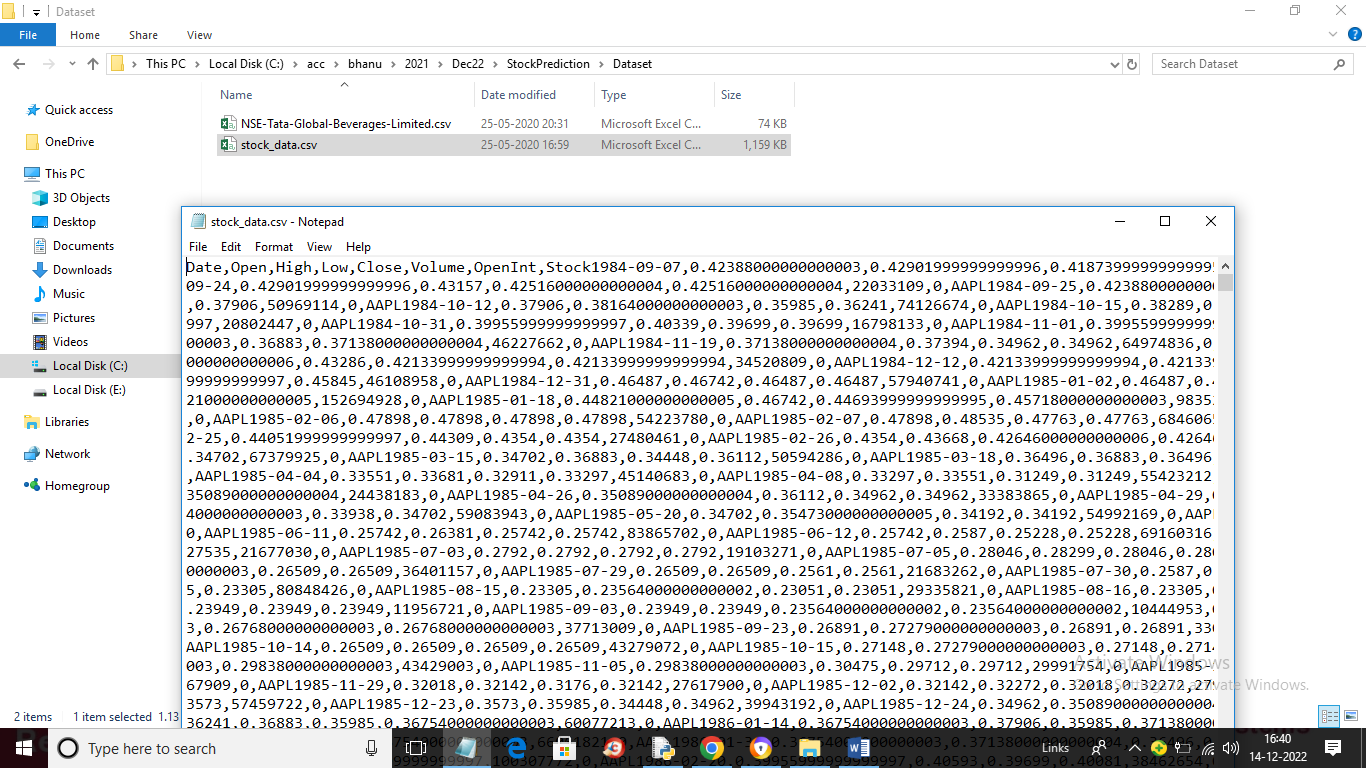
In this project we are using LSTM and ANN algorithms to predict stock prices and we trained both algorithm in different stock prices such as TATA, TESLA, Facebook, Apple etc. In both algorithm ANN is giving high accuracy and less Mean Square Error (MSE).

We have experiment both algorithms on above mention different stock prices and both algorithm showing close and accurate prices of original test data. We have splitted dataset into train & test where application using 80% dataset size for training and 20% for testing

To implement this project we have used same dataset given in your requirement file and to implement this project we have designed following modules

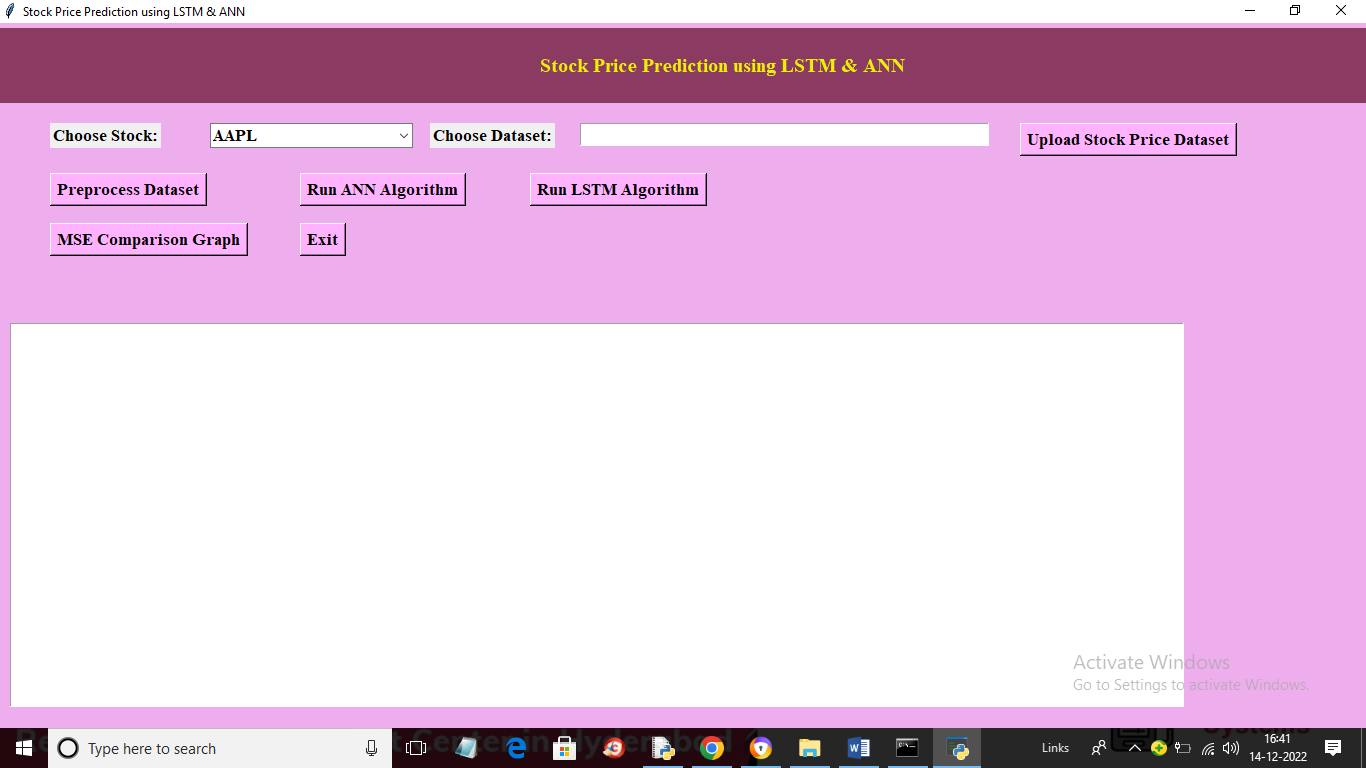
1. Upload Stock Price Dataset: using this module we will select STOCK Name and then upload dataset and application will extract all records from selected stock name and then sort dataset in ascending date order and then plot stock price history graph
2. Preprocess Dataset: using this module we will remove missing values and then normalize stock values using MINMAX SCALER algorithm and then split dataset into train and test. 80% dataset using for training and 20% for testing
3. Run ANN Algorithm: using this module we will input 80% dataset to ANN algorithm to train a model and this model will be applied on 20% test data and then calculate difference between actual stock test price and predicted price as MSE error.
4. Run LSTM Algorithm: using this module we will input 80% dataset to LSTM algorithm to train a model and this model will be applied on 20% test data and then calculate difference between actual stock test price and predicted price as MSE error.
5. MSE Comparison Graph: using this module we will plot MSE comparison graph between both algorithms

To implement above modules we have used below 2 datasets

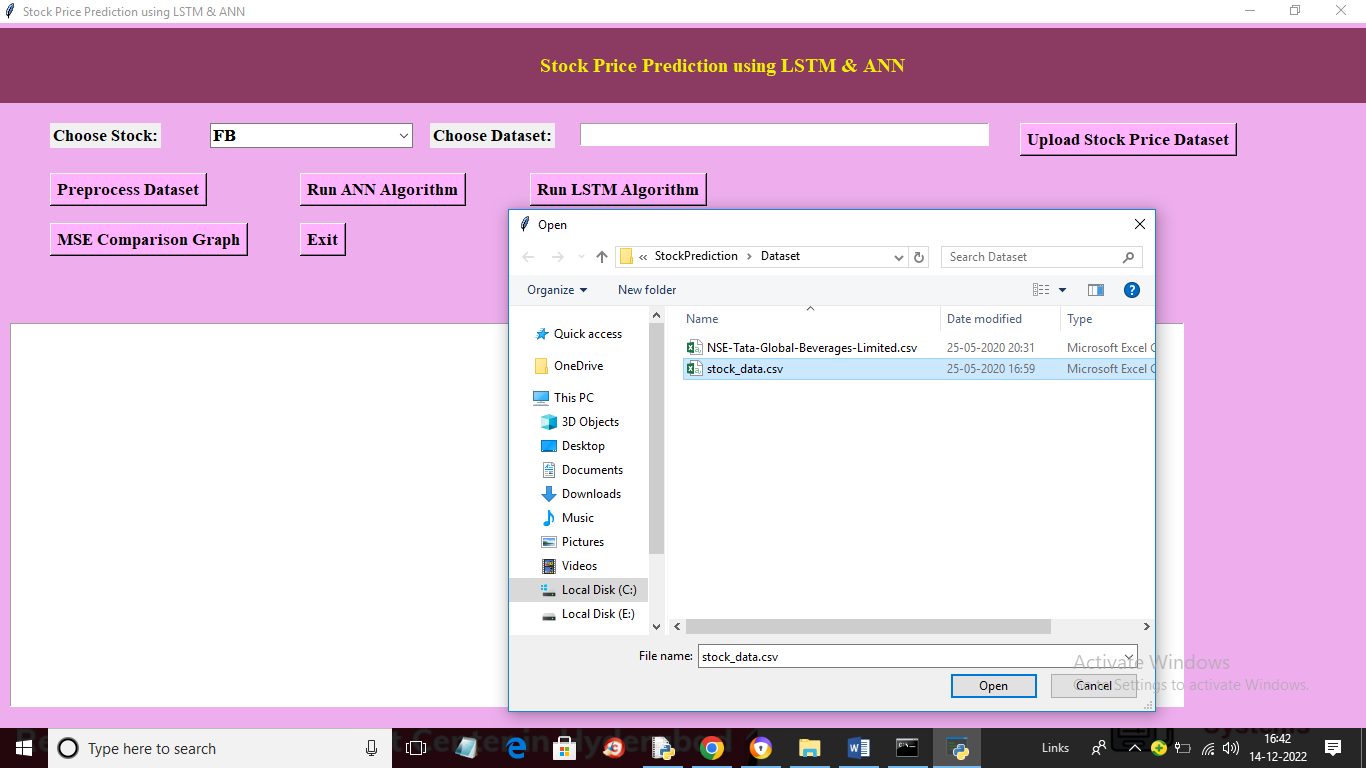


SCREEN SHOTS

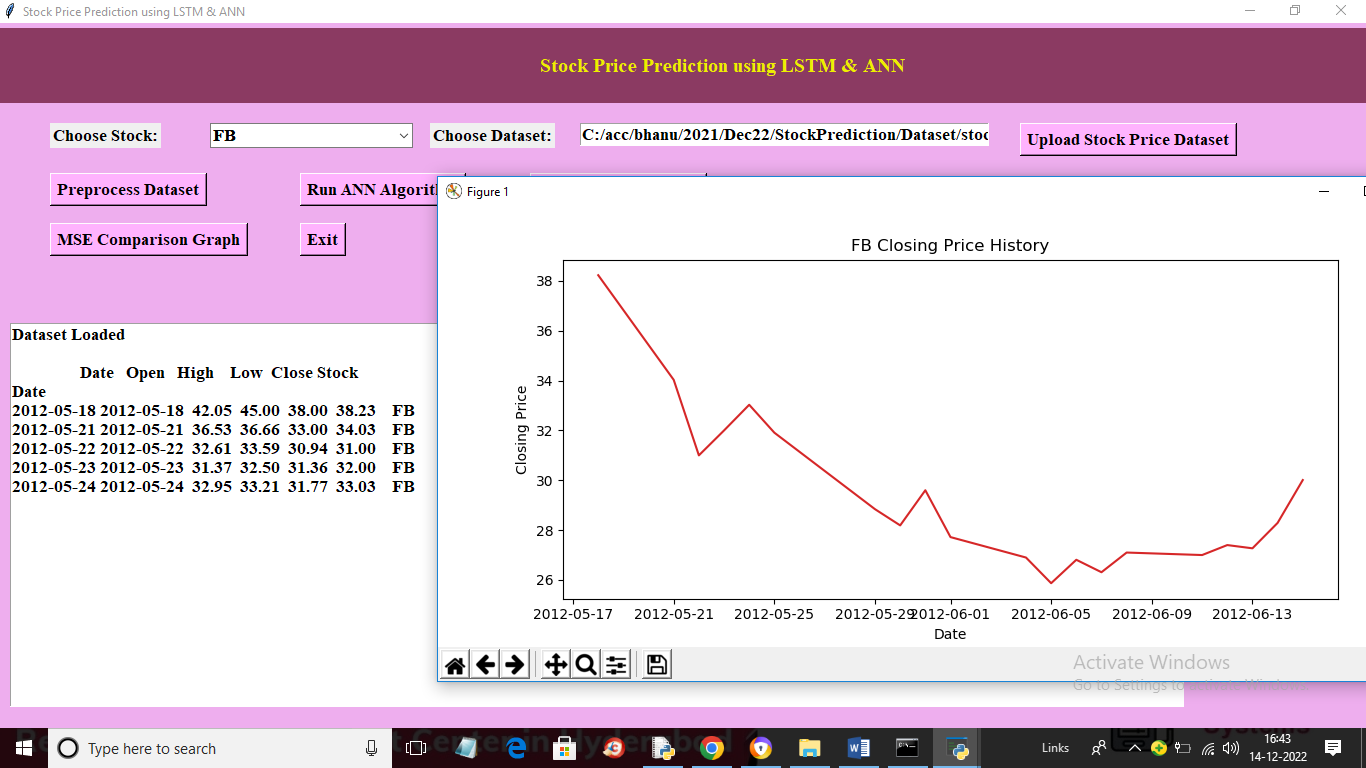
To run project double click on ‘run.bat’ file to get below screen



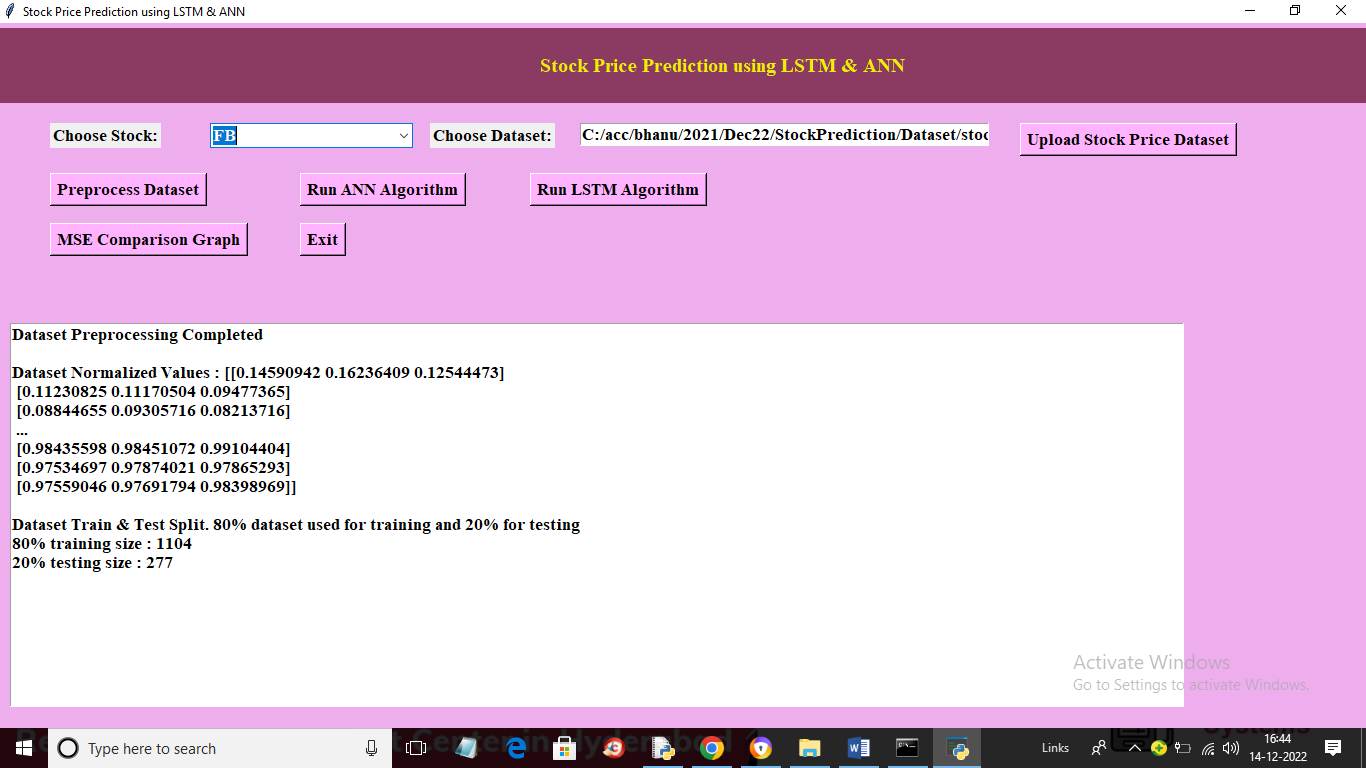
In above screen choose stock name from drop down box and then click on ‘Upload Stock Price Dataset’ button to upload dataset and get below output



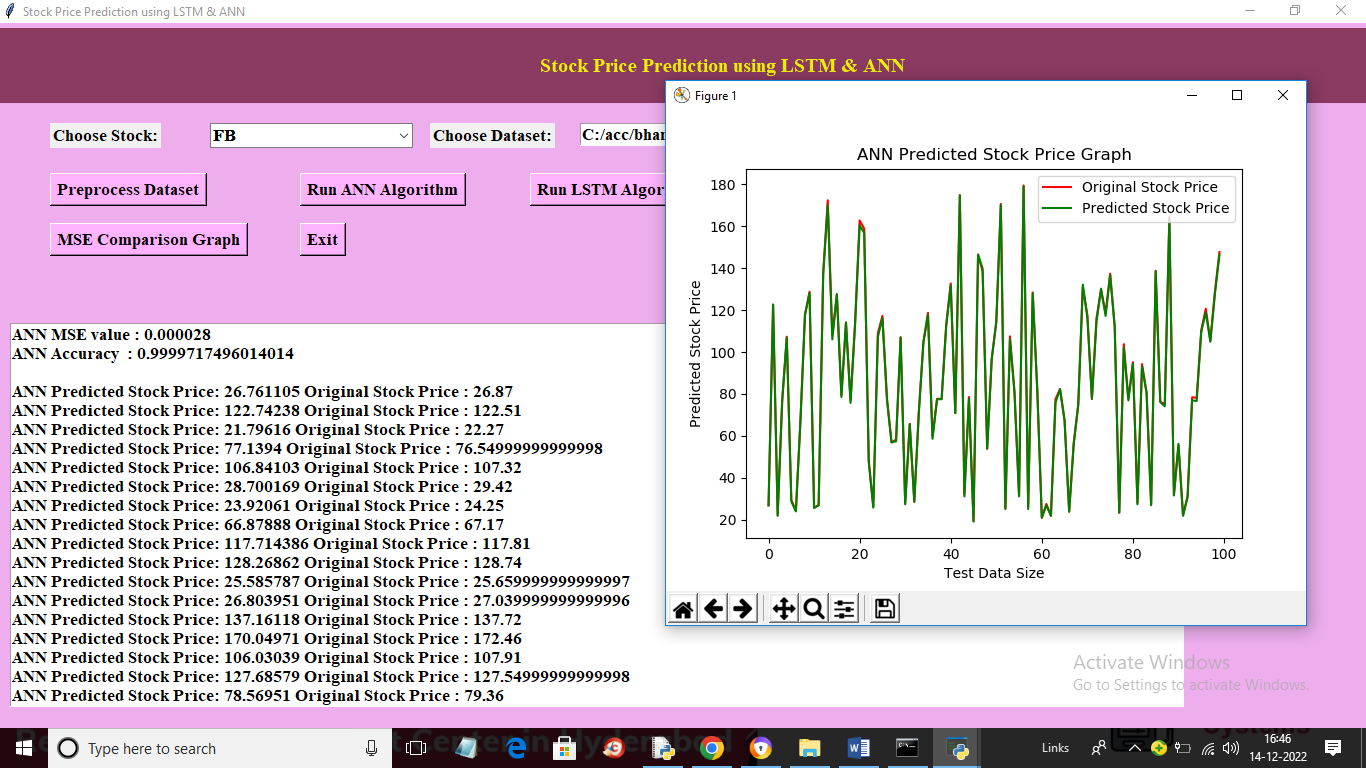
In above screen I selected stock name as ‘FB’ and then uploading ‘stock\_data.csv’ file and then click on ‘Open’ button to load dataset and get below output



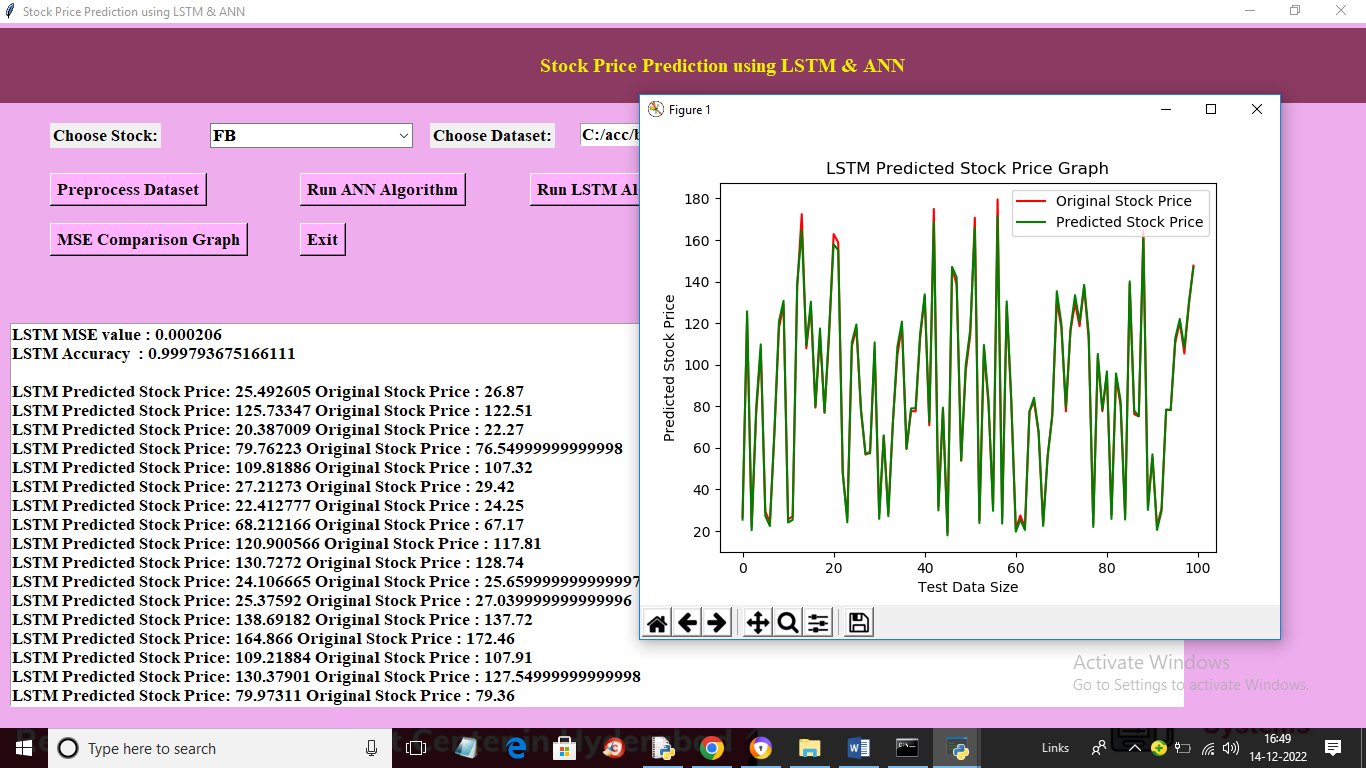
In above screen dataset loaded and in graph we can see stock history prices on different dates where x-axis contains DATE and y-axis represents stock price on that date. Now close above graph and then click on “Preprocess Dataset’ button to normalize dataset values and then split into train and test



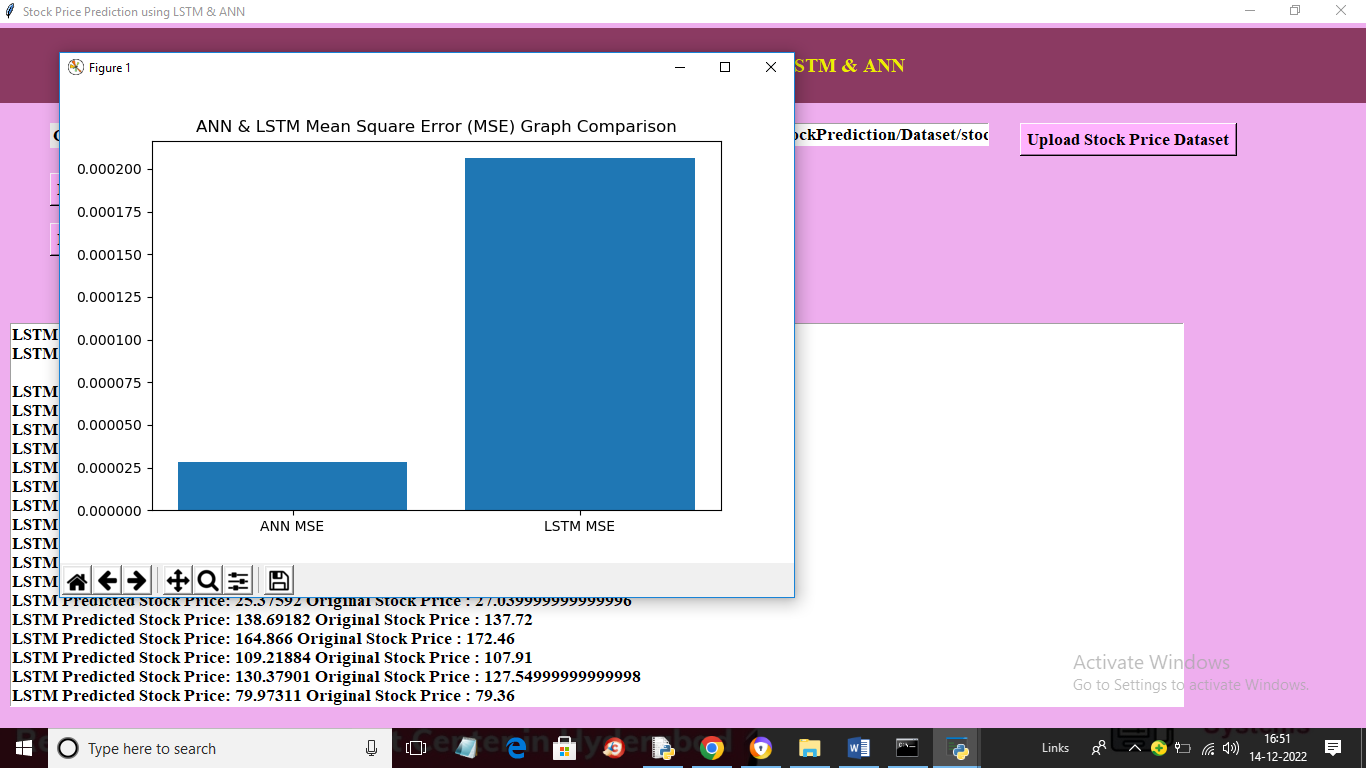
In above screen we can see normalized dataset values and then we can see 80% training records size and 20% testing records size and now click on ‘Run ANN Algorithm’ button to train ANN and get below predicted prices



In above screen in first 2 lines we can see ANN MSE value (the lower the MSE the better is the algorithm) and we can see accuracy also and then in next lines we can see ANN predicted prices and original 20% test data prices and we can see both prices are too close and in graph x-axis represents TEST data DAYS and y-axis represents STOCK PRICES and green line represents PREDICTED prices and red line represents ORIGINAL STOCK prices and we can see both lines are fully overlapping so predicted and test prices are too close and accurate. Now close above graph and then click on ‘Run LSTM Algorithm’ button to get below output



In above screen we can see LSTM predicted prices with MSE and accuracy values and we can see LSTM MSE is little higher than ANN but its predicted also too close as LSTM graph values are also overlapping with TEST prices. Now click on “MSE Comparison Graph’ button to get below graph



In above graph x-axis represents algorithm names and y-axis represents MSE values and in both algorithms ANN got less MSE compare to LSTM so we can say ANN is better than LSTM. Similarly you can upload other dataset and select stock name and predict prices

**8. CONCLUSION:**

The study of the share is carried out in this paper and it can be carried out for several shares in the future. Prediction could be more reliable if the model trains a greater number of data sets using higher computing capacities, an increased number of layers, and LSTM modules. In future enhancement the inclusion of sentiment analysis from social media to understand what the market thinks about the price variation for a particular share and it can be implement this by adding twitter and Facebook API to our program as Facebook is a leading social media which has lots of market trend information posted by users.

**9. REFERENCES:**

1. Sreelekshmy Selvin, Vinayakumar R, Gopalakrishnan E.A, Vijay Krishna Menon, Soman K.P, “Stock Price Prediction Using LSTM, RNN And CNN-SLIDING WINDOW MODEL”, 2017

2. Kaustubh Khare, Omkar Darekar, Prafull Gupta, Dr. V. Z. Attar, “Short Term Stock Price Prediction Using Deep Learning”, 2017.

3. Xiumin Li, Lin Yang, Fangzheng Xue, Hongjun Zhou, “Time series prediction of stock price using deep belief networks with Intrinsic Plasticity”, 2017.

4. Jiahong Li, Hui Bu, Junjie Wu, “Sentiment-Aware Stock Market Prediction: A Deep Learning Method”. 5. Murtaza Roondiwala, Harshal Patel, Shraddha Varma, “Predicting Stock Prices Using LSTM”, April 2017.

7. Ryo Akita, Akira Yoshihara, Takashi Matsubara, Kuniaki Uehara, “Deep learning for stock prediction using numerical and textual information”, 2016.

8. Bhagyashree Nigade at el., “Comparative Study of Stock Prediction System using Regression Techniques”, March - April 2017.

9. S Abdul Salam Suleiman Polanyi, Adele, Kayode S., Jimoh, R. G, “Stock Trend Prediction Using Regression Analysis – A Data Mining Approach”, July 2011.

10. Mr. Amit B. Suthar, Ms. Hiral R. Patel, Dr. Satyen M. Parikh, “A Comparative Study on Financial Stock Market Prediction Models”, 2012.

11. S.Prasanna, Dr.D.Ezhilmaran, “An analysis on Stock Market Prediction using Data Mining Techniques”, 2013.

12. Ruchi Desai, Prof.Snehal Gandhi, “Stock Market Prediction Using Data Mining”, 2014. 13. Bini B.S, Tessy Mathew, “Clustering And Regression Techniques For Stock Prediction”, 2015. 14. G. S. Navale, Nishant Dudhwala, Kunal Jadhav, “Prediction of Stock Market using Data Mining and Artificial Intelligence”, 2016.

15. Bhagyashree Nigade, Aishwarya Pawar at el., “Stock Trend Prediction Using Regression Analysis – A Data Mining Approach”, February 2017.

16. Shalini Lotlikar at el, “Stock Prediction Using Clustering And Regression Techniques”, May 2017

17. Mr. Pramod Mali, Hemangi Karchalkar at el., “Open Price Prediction of Stock Market using Regression Analysis”, May 2017.