**CHAPTER 3**

**METHODOLOGY**

Introduction

In this chapter we are going to discuss how we implemented all the methods for achieving the desired result, the methods which we’re going to discuss are data collection, data pre-processing, model making, tuning the model, predicting the results and forecasting for a particular range of time.

Methodology

Detailed Methodology

The stock market prediction is a tuff job to do and to accurately guess the result one needs to analyze the market thoroughly i.e., he/she should observe the past performance of the stock , the company’s numbers i.e., volume traded, profit making ability of the company etc., the market news about the stock or now-a-days twitter trends, whether it is positive or negative, positive news always help the stock to perform well in the market while negative news results in chaos among the investors resulting in poor performance of the stock then comes insider news; many a times important news/deals are kept private from the public for the benefit of the company but if it gets leaked then it effects the performance of the stock. Mainly these are the factors upon which stock market can be predicted.

But for this project we’ve used only the past data of the stock as our factor, we took last 5 years of stock’s performance data. **The data was collected** from yahoo finance’s website (<https://in.finance.yahoo.com/>) it is a trusted site and the data is easily available and can be directly downloaded in .csv format.

After downloading the data we **cleaned the data** i.e., the data contained some null values and if raw data was fed in the model then it could either throw an error or the model wouldn’t had trained efficiently. So, the rows containing null values were deleted and data was cleaned.

After cleaning the data, the thing we did was **pre-processed the data** i.e., we took only the closing prices column and then splitted it into two subgroups training data and testing data. Training data contained the data which we were going to use for training our model and the testing data contained the data upon which we were going to test our model. We took 67% of the data as our training data and rest for testing.

This data was still not ready to be served in the model as our model expected time series data, so we converted the linear data into time series data.

After all the data pre-processing now we started to **make our model**, we made the model using tensorflow library present in python and used **LSTM** layers for making the model.

After creating the model we **trained** it on our training model for a particular number of epochs and after that checked the model’s accuracy for ensuring it efficiently trained. If the model’s performance was not up to the mark then we had done some hyperparameter tuning where we tuned some parameters such as number of epochs, number of LSTM layers, number of nodes present in the layers etc.

After all this our model was **ready to be predicted**, so we fed it with our testing data and predicted whether it yielded satisfying results or not. After the prediction work we then moved on to the forecasting part where the model future forecast performance for the chosen stock.

// for circuit diagram give the detail of the model summary

Tools Used

For Data cleaning and Data preprocessing:

* numpy (1.19.2) – NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
* pandas (1.1.3) - Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.
* scikit\_learn (0.24.2) - Scikit-learn is a free software machine learning library for the Python programming language. In this project it is used for data scaling.

For making the model:

* tensorflow (2.5.0) - TensorFlow is a free and open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks
* keras (2.5.0) - Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library.

For Data visualization:

* plotly (5.1.0) - Plotly provides online graphing, analytics, and statistics tools for individuals and collaboration, as well as scientific graphing libraries for Python, R, MATLAB, Perl, Julia, Arduino, and REST.

For the web app:

* streamlit (0.83.0) - Streamlit is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science