**PROJECT 6:** STOCK PRICE PREDICTION

**PROJECT TITLE:** STOCK PRICE PREDICTION

**PHASE 3:** DEVELOPMENT PART 1

***Given Dataset:***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S.No** | **Date** | **Open** | **High** | **Low** | **Close** | **Volume** |
| 1 | 01-03-2012 | 325.25 | 332.83 | 324.97 | 663.59 | 73,80,500 |
| 2 | 01-04-2012 | 331.27 | 333.87 | 329.08 | 666.45 | 57,49,400 |
| 3 | 01-05-2012 | 329.83 | 330.75 | 326.89 | 657.21 | 65,90,300 |
| 4 | 01-06-2012 | 328.34 | 328.77 | 323.68 | 648.24 | 54,05,900 |
| 5 | 01-09-2012 | 322.04 | 322.29 | 309.46 | 620.76 | 1,16,88,800 |
| … | … | … | … | … | … | … |
| 1254 | 12-23-2016 | 790.9 | 792.74 | 787.28 | 789.91 | 6,23,400 |
| 1255 | 12-27-2016 | 790.68 | 797.86 | 787.66 | 791.55 | 7,89,100 |
| 1256 | 12-28-2016 | 793.7 | 794.23 | 783.2 | 785.05 | 11,53,800 |
| 1257 | 12-29-2016 | 783.33 | 785.93 | 778.92 | 782.79 | 7,44,300 |
| 1258 | 12-30-2016 | 782.75 | 782.78 | 770.41 | 771.82 | 17,70,000 |

(1258 rows x 6 columns)

**1. Import Libraries**

*Program:*

#Import libraries

import os

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

%matplotlib inline

**2. Importing the Training set**

*Program:*

dataset\_train = pd.read\_csv (“Google\_Stock\_Price\_Train.csv”)

training\_set = dataset\_train.ilot [:, 1:2].values

**3. Feature Scaling**

*Program:*

from sklearn.preprocessing import MinMaxScaler

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

training\_set\_scaled = sc.fit\_transform (training\_set)

**4. Creating a data structure with 60 time steps and 1 output**

*Program:*

x\_train = []

y\_train = []

for i in range (60, 1258):

x\_train.append (training\_set\_scaled [i-60: i, 0])

y\_train.append (training\_set\_scaled [I, 0])

x\_train, y\_train = np.array (x\_train), np.array (y\_train)

**5. Reshaping**

*Program:*

x\_train = np.reshape (x\_train, (x\_train.shape [0],

x\_train.shape [1], 1))

**Importance of Loading and processing dataset:**

Loading and pre-processing the dataset is an important first step in building any machine learning model. However, it is especially important for Stock price prediction models, as Stock price datasets are often complex and noisy.

By loading and pre-processing the dataset, we can ensure that the machine learning algorithm is able to learn the data effectively and accurately.

**Challenges involved in loading and pre-processing a Stock price dataset:**

Loading and Pre-processing a Stock price prediction dataset can be a complex task due to various challenges associated with financial data.

***1. Data Quality and Cleaning:***

* *Inaccurate or missing data:* Stock price datasets often contain missing or inaccurate values, which can affect the quality of predictions.
* *Outliers:* Extreme values or outliers can distort the dataset and need to be handled appropriately.

***2. High-Frequency Data:***

* Stock data can come in various frequencies (daily, hourly, minute-level), each with its own challenges. Higher-frequency data can be noisy and may require down sampling to a lower frequency for modelling.

***3. Time-Series Decomposition:***

* Stock price data can have trend, seasonality, and noise components that need to be decomposed before analysis.

***4. Feature Engineering:***

* Creating meaningful features for stock price prediction, such as moving averages, technical indicators, and sentiment analysis, requires domain expertise.

***5. Scaling and Normalization:***

* Different stocks can have significantly different price ranges, so proper scaling and normalization are necessary to ensure that the model can generalize across stocks.

**How to overcome the Challenges involved in loading and pre-processing a Stock price dataset:**

Loading and pre-processing stock price prediction dataset can be a complex and challenging task, as financial data can be messy and require careful handling.

***1. Data Collection:***

* *Identify reliable data sources:* Choose reputable sources for stock price data, such as financial websites, APIs, or data providers. Quality data is essential for accurate predictions.
* *Data format:* Ensure that the data is available in a structured format like CSV, JSON. Or through an API for easy access.

***2. Data Cleaning:***

* *Handle missing values:* Stock price data often has missing values, which need to be imputed or handled carefully. Common methods include forward-fill, backward-fill, or interpolation.
* *Outlier detection:* Identify and deal with outliers that can distort your analysis. You can use statistical methods or visualization to detect outliers.

***3. Feature Engineering:***

* *Create relevant features:* Stock price prediction often benefits from the creation of additional features like moving averages, technical indicators (e.g., MCD, RSI), and volume-based metrics.
* *Time-based features:* Incorporate time-related information, such as day of the week, month, or year, to capture seasonality and temporal patterns.

***4. Normalization and Scaling:***

* *Normalize data:* Scaling the data (e.g., Min-Max scaling or Standardization) helps models converge faster and perform better. Normalize both input features and target variables.

***5. Handling Time Series Data:***

* Time series data requires special handling. Ensure that your data is sorted by date and consider using techniques like sliding windows to create input sequences for your models.

**Google\_Stock\_Price\_Test.csv**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S.No** | **Date** | **Open** | **High** | **Low** | **Close** | **Volume** |
| 1 | 01-03-2017 | 778.81 | 789.63 | 775.8 | 786.14 | 16,57,300 |
| 2 | 01-04-2017 | 788.36 | 791.34 | 783.16 | 786.9 | 10,73,000 |
| 3 | 01-05-2017 | 786.08 | 794.48 | 785.02 | 794.02 | 13,35,200 |
| 4 | 01-06-2017 | 795.26 | 807.9 | 792.2 | 806.15 | 16,40,200 |
| 5 | 01-09-2017 | 806.4 | 809.97 | 802.83 | 806.65 | 12,72,400 |
| 6 | 01-10-2017 | 807.86 | 809.13 | 803.51 | 804.79 | 11,76,800 |
| 7 | 01-11-2017 | 805 | 808.15 | 801.37 | 807.91 | 10,65,900 |
| 8 | 01-12-2017 | 807.14 | 807.39 | 799.17 | 806.36 | 13,53,100 |
| 9 | 01-13-2017 | 807.48 | 811.22 | 806.69 | 807.88 | 10,99,200 |
| 10 | 01-17-2017 | 807.08 | 807.14 | 800.37 | 804.61 | 13,62,100 |
| 11 | 01-18-2017 | 805.81 | 806.21 | 800.99 | 806.07 | 12,94,400 |
| 12 | 01-19-2017 | 805.12 | 809.48 | 801.8 | 802.17 | 9,19,300 |
| 13 | 01-20-2017 | 806.91 | 806.91 | 801.69 | 805.02 | 16,70,000 |
| 14 | 01-23-2017 | 807.25 | 820.87 | 803.74 | 819.31 | 19,63,600 |
| 15 | 01-24-2017 | 822.3 | 825.9 | 817.82 | 823.87 | 14,74,000 |
| 16 | 01-25-2017 | 829.62 | 835.77 | 825.06 | 835.67 | 14,94,500 |
| 17 | 01-26-2017 | 837.81 | 838 | 827.01 | 832.15 | 29,73,900 |
| 18 | 01-27-2017 | 834.71 | 841.95 | 820.44 | 823.31 | 29,65,800 |
| 19 | 01-30-2017 | 814.66 | 815.84 | 799.8 | 802.32 | 32,46,600 |
| 20 | 01-31-2017 | 796.86 | 801.25 | 790.52 | 796.79 | 21,60,600 |

**Loading the dataset:**

***Output:***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S.No** | **Date** | **Open** | **High** | **Low** | **Close** | **Volume** |
| 1 | 01-03-2012 | 325.25 | 332.83 | 324.97 | 663.59 | 73,80,500 |
| 2 | 01-04-2012 | 331.27 | 333.87 | 329.08 | 666.45 | 57,49,400 |
| 3 | 01-05-2012 | 329.83 | 330.75 | 326.89 | 657.21 | 65,90,300 |
| 4 | 01-06-2012 | 328.34 | 328.77 | 323.68 | 648.24 | 54,05,900 |
| 5 | 01-09-2012 | 322.04 | 322.29 | 309.46 | 620.76 | 1,16,88,800 |
| … | … | … | … | … | … | … |
| 1254 | 12-23-2016 | 790.9 | 792.74 | 787.28 | 789.91 | 6,23,400 |
| 1255 | 12-27-2016 | 790.68 | 797.86 | 787.66 | 791.55 | 7,89,100 |
| 1256 | 12-28-2016 | 793.7 | 794.23 | 783.2 | 785.05 | 11,53,800 |
| 1257 | 12-29-2016 | 783.33 | 785.93 | 778.92 | 782.79 | 7,44,300 |
| 1258 | 12-30-2016 | 782.75 | 782.78 | 770.41 | 771.82 | 17,70,000 |