STOCK PRICE PREDICTION (Phase 5)

Madras Institute of Technology

2021506088 – Santhi Priya D N
2021506096 - Shiek Sajnathul Faizana
2021506103 – Sowmiya J
2021506323 - Revanth P

INTRODUCTION:

The phase 5 submission aims to provide a collective data on all the phases submitted beforehand. In this section, provide a brief introduction to the project, its objectives, and the significance of stock price prediction in financial decision-making, the code used for pre-processing the dataset (phase-3) and the code used for feature engineering, model training and evaluation (phase-4) are also included in this document.

PROBLEM STATEMENT:

The problem at hand is to develop a robust and accurate stock price prediction model to assist investors in making well-informed decisions and optimizing their investment strategies. This project encompasses several key phases, including data collection, pre-processing, model selection, training, and evaluation.

DESIGN THINKING AND PHASES OF DEVELOPMENT:

Phase 1:

This phases focuses on understanding the project, that is to study about what is the actual aim of stock price prediction project what the methods and algorithms available for predicting the stock, what is a dataset etc., and the process of collecting the dataset is also covered in this phase.

Phase 2:

In Phase 2, the focus was on leveraging advanced deep learning techniques to enhance the accuracy of our predictions. Specifically, CNN-LSTM and attention mechanisms to address the complexities of stock price prediction.

Phase 3:

This phase is where we actually used the code to pre-process the given dataset. Python IDLE was installed in our own laptops and we studied about pre-processing the dataset and pre-processed the given dataset.

Phase 4:

Phase 4 is an advanced phase 3, in sense we coded to derive the feature engineering, model training and evaluation of the given dataset which are the final steps for perfect stock price prediction process.

DESCRIPTION ABOUT THE DATASET:

The dataset used for pre-processing and model training were provided the evaluator from the kaggle.com . MSFT.csv contains all the life time stocks data from 3/13/1986 to 12/10/2019 this dataset contains 7 columns including dates, opening, high, low, closing, adj_close, volume, code up your first kernel: LSTMs and Deep Reinforcement Learning agents works well for this dataset.

Dataset Link: https://www.kaggle.com/datasets/prasoonkottarathil/microsoft-lifetime-stocks-dataset

DATA PRE-PROCESSING STEPS:

Pre-processing the dataset includes the following steps:

1. Understanding the dataset.

2. Removing duplicates.

3. Data transformation (Standardization and Normalization)

4. Handling Outliers

5. Data splitting

The above processes were implemented and the python code is as follows:

(Note: For the below python code to be implemented the pandas library, sklearn library has to be installed)

SOURCE CODE:

```
#importing libraries
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, LabelEncoder
from sklearn.model_selection import train_test_split

#reading the file
df = pd.read_csv(r"C:\Users\DELL\Downloads\MSFT.csv")
print(df)

#understanding the dataset
print(df.head())
print(df.describe())
print(df.isnull().sum())

#removing duplicates
df = df.drop_duplicates()
print(df)
#-- to check if any duplicates rows are removed
print(df.isnull().sum())

#DATA TRANSFORMATION - z-score standardization
```

```
scaler = MinMaxScaler()
df['Normalized_Open']=scaler.fit_transform(df[['Open']])
label_encoder = LabelEncoder()
df['Encoded_Date'] = label_encoder.fit_transform(df['Date'])
print(df)

#feature engineering - a new column(feature) High minus Low is obtained by subtracting low
from high
df['High_Minus_Low'] = df['High'] - df['Low']
print(df)

#handling outliers
thresholds = {'Date': ("1986-03-19","2020-01-02")}
for col, (lower,upper) in thresholds.items():
   df=df[(df[col] >= lower ) & (df[col] <= upper)]
print(df)

#data splitting
#-- Splitting the dataset
X = df.drop('Close', axis=1)
y = df['Close']
#--Split into training, validation, and testing sets
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.3, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5,
random_state=42)
#--Display the split datasets
print("Training set:")
print(X_train)
print("Validation set:")
print(X_val)
print("Testing set:")
print(X_test)
```

MODEL TRAINING PROCESS:

Model training is the phase where we feed our prepared data into a chosen predictive model. The model learns to identify patterns and relationships in the training data, allowing it to make predictions on unseen data. In our case, we continue to explore and implement models, particularly focusing on LSTM, to capture the temporal dependencies in stock price movements.

SOURCE CODE:

```
import pandas as pd
df = pd.read_csv('Modified_msft.csv')
print(df)
```

```
#feature engineering
#--adding lag values
df['Volume_lag1']=df['Volume'].shift(1)
df['Volume_lag2']=df['Volume'].shift(2)

#-- 7 day moving average
df['7-day_MA'] = df['Volume'].rolling(window=7).mean()
print(df)

# Import necessary libraries
import numpy as np
import pandas as pd
import tensorflow as tf
from tensorflow import keras
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error, r2_score

# Load your time series data or sequence data
# Replace 'data' with your time series data
# Example: data = load_your_data()

# Data preprocessing
scaler = MinMaxScaler()
scaled_data = scaler.fit_transform(df[['Volume']])

# Define the sequence length and split the data into sequences
sequence_length = 10  # Adjust this based on your problem
X, y = [], []
for i in range(len(scaled_data) - sequence_length):
    X.append(scaled_data[i:i+sequence_length])
    y.append(scaled_data[i+sequence_length])

X, y = np.array(X), np.array(y)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Reshape the input data to match the expected input shape
num_features = X_train.shape[2]  # Extract the number of features from the reshaped data

# Define and compile the LSTM model
model = keras.Sequential()
model.add(keras.layers.LSTM(units=50, activation='relu', input_shape=(sequence_length,
num_features)))
model.add(keras.layers.Dense(1))  # Adjust the output layer for your problem
model.compile(optimizer='adam', loss='mean_squared_error')

# Train the model
model.fit(X_train, y_train, epochs=50, batch_size=32)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Inverse transform predictions to the original scale if needed
y_pred = scaler.inverse_transform(y_pred)
y_test = scaler.inverse_transform(y_test)

# Evaluate the model using appropriate metrics
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Print or visualize the model's performance metrics
print(f"Mean Squared Error: {mse}")
print(f"R-squared (R²): {r2}")
```

KEY FINDINGS AND INSIGHTS:

Feature Engineering: Effective feature engineering is critical for improving prediction accuracy. Data scientists create relevant features, such as technical indicators (e.g., moving averages, Relative Strength Index), trading volume, and sentiment analysis scores, to capture meaningful patterns in stock price data.

Time Series Analysis: Time series analysis is a fundamental technique for understanding and modeling stock price data. Data scientists use methods like autoregressive integrated moving average (ARIMA) and GARCH models to model and forecast stock price volatility.

Machine Learning Models: Data scientists leverage a wide range of machine learning models, including linear regression, decision trees, random forests, and support vector machines, for stock price prediction. Ensemble methods are also used to combine the strengths of multiple models.

Deep Learning: Deep learning techniques, particularly recurrent neural networks (RNNs) and Long Short-Term Memory (LSTM) networks, are employed for modeling sequential data, making them well-suited for time series prediction.

CONCLUSION:

As we conclude Phase 5, we recognize that our project is a dynamic and ongoing effort. Our documentation serves as a valuable resource for understanding the project's evolution, from inception to its current state. With each phase, we have inched closer to our ultimate goal of delivering a reliable stock price prediction model for investors.