STOCK PRICE PREDICTION (Phase 1)

Madras Institute of Technology

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

INTRODUCTION:

In the realm of finance, informed decision-making is critical for successful investments. This project focuses on leveraging machine learning to forecast stock prices, providing investors with a powerful tool to optimize their investment strategies. By analyzing historical market data and employing predictive models, we aim to assist investors in making well-informed choices and ultimately enhancing their investment outcomes.

PROBLEM DEFINITION:

The challenge revolves around accurately predicting future stock prices based on historical market data. This involves handling the inherent volatility and unpredictability of financial markets. The ultimate goal is to create a predictive model that can forecast stock prices with a high level of accuracy

DATA COLLECTION:

A. Data Sources:

For this project, historical stock market data will be collected from reputable financial databases, APIs, and historical market data repositories.

B. Data Quality Assurance:

To ensure data integrity and reliability, a rigorous data quality check will be performed to address any discrepancies or anomalies.

For Example,

We gather historical stock market data for Company XYZ, including features like date, open price, close price, volume, and relevant indicators such as moving averages and relative strength index (RSI).

DATA PREPROCESSING:

A. Data Cleaning:

- Handling missing values, outliers, and anomalies in the dataset.
- Ensuring consistent formatting and eliminate data inconsistencies includes interpolation or filling with zeros.

B. Feature Transformation:

- Converting categorical features into numerical representations for machine learning compatibility.
- Normalizing or scale numerical features to a standard range.

For Example,

After collecting data from the historical stock market of XYZ Company, the pre-processed dataset will look like this:

Date	Open	High	Low	Close	Adj Close	Volume
1990-01-02	0.605903	0.616319	0.598090	0.616319	0.447268	53033600
1990-01-03	0.621528	0.626736	0.614583	0.619792	0.449788	113772800
1990-01-04	0.619792	0.638889	0.616319	0.638021	0.463017	125740800
1990-01-05	0.635417	0.638889	0.621528	0.622396	0.451678	69564800
1990-01-08	0.621528	0.631944	0.614583	0.631944	0.458607	58982400

FEATURE ENGINEERING:

A. Feature Creation:

- Compute moving averages, technical indicators, and lagged variables to enhance predictive power.
- Identify features that may correlate with stock price movements.
- For example, we calculate various moving averages (e.g., 10-day, 50-day) to capture short-term and long-term trends.
- We calculate technical indicators such as RSI, which can help in identifying overbought or oversold conditions.

B. Feature Selection:

- Choose relevant features based on their impact on prediction accuracy.
- Optimize the feature set for better model performance.

For Example,

Consider the following table:

```
|    Date    | Close Price |
|------------|-------------|
| 2023-01-01 |   100.00    |
| 2023-01-02 |   105.25    |
| 2023-01-03 |   110.75    |
| 2023-01-04 |   112.50    |
| 2023-01-05 |   107.75    |
| 2023-01-06 |   104.50    |
| 2023-01-07 |   104.00    |
| 2023-01-08 |   107.75    |
| 2023-01-09 |   109.25    |
| 2023-01-10 |   110.50    |
| ...        |     ...     |
```

We'll calculate moving average:

10-Day Moving Average (Short-term Trend):

To calculate the 10-day moving average, we'll take the average of the closing prices of the past 10 days for each data point. We start from the 10th day since we need data for the previous 9 days to calculate the first moving average value.

Date	Close Price	10-Day MA
2023-01-01	100.00	-
2023-01-02	105.25	-
2023-01-03	110.75	-
2023-01-04	112.50	-
2023-01-05	107.75	-
2023-01-06	104.50	-
2023-01-07	104.00	-
2023-01-08	107.75	-
2023-01-09	109.25	-
2023-01-10	110.50	107.80
...	...	...

MODEL SELECTION:

Step 1:

Evaluate the pros and cons of each algorithm in the context of stock price prediction.

Step 2:

The most appropriate algorithm(s) based on performance, complexity, and accuracy requirements will be selected after choosing the Company and the stock data collected.

For Example,

Date	Open	High	Low	Close	Adj Close	Volume
1990-01-02	0.605903	0.616319	0.598090	0.616319	0.447268	53033600
1990-01-03	0.621528	0.626736	0.614583	0.619792	0.449788	113772800
1990-01-04	0.619792	0.638889	0.616319	0.638021	0.463017	125740800
1990-01-05	0.635417	0.638889	0.621528	0.622396	0.451678	69564800
1990-01-08	0.621528	0.631944	0.614583	0.631944	0.458607	58982400

For this dataset, LSTM is the most appropriate model based on the accuracy requirements, hence **LSTM algorithm** can be used.

MODEL TRAINING:

A. Training Data:

- Firstly, we Split the pre-processed data into training and validation sets.
- Using historical data to train the selected model(s) for stock price prediction.

B. Hyper-parameter Tuning:

- Secondly, Optimize hyper-parameters for each selected algorithm to achieve the best predictive performance.

EVALUATION:

- To evaluate the model's performance, we use appropriate time series forecasting metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).
- We split the dataset into a training set and a testing set. The testing set contains data the model hasn't seen before, and we use it to assess how well the model predicts future stock prices.
- After making predictions, we compare them to the actual stock prices in the testing set and calculate the MAE and RMSE to measure the model's accuracy.

For Example,

Scenario:

Suppose you have developed a stock price prediction model, and you want to evaluate its performance using historical data. You've made predictions for the closing prices of a stock for a specific period, and you want to assess how well your model's predictions align with the actual prices.

Sample Data: Here's a sample of actual closing prices (ground truth) and predicted closing prices for a given period:

Date	Actual Close Price	Predicted Close Price
2023-01-01	100.00	98.50
2023-01-02	105.25	106.00
2023-01-03	110.75	112.25
2023-01-04	112.50	110.75
2023-01-05	107.75	108.00
2023-01-06	104.50	103.25
2023-01-07	104.00	105.50
2023-01-08	107.75	108.75
2023-01-09	109.25	109.00
2023-01-10	110.50	112.00

Evaluation Metrics:

Mean Absolute Error (MAE):

- MAE measures the average absolute difference between the actual and predicted values. It provides a straightforward measure of prediction accuracy.
- To calculate MAE, you take the absolute difference for each data point, sum them, and then divide by the number of data points.

Formula:

MAE = (1/n) * Σ|Actual - Predicted|

Calculation:

MAE = (|100.00 - 98.50| + |105.25 - 106.00| + ... + |110.50 - 112.00|) / 10

≈ 2.075

Root Mean Squared Error (RMSE):

- RMSE measures the square root of the average of the squared differences between the actual and predicted values. It penalizes larger errors more heavily than MAE.
- To calculate RMSE, you square the differences for each data point, calculate the mean, and then take the square root.

Formula:

RMSE = √((1/n) * Σ(Actual - Predicted)^2)

Calculation:

RMSE = √((100.00 - 98.50)^2 + (105.25 - 106.00)^2 + ... + (110.50 - 112.00)^2) / 10

≈ 2.237

Interpretation:

MAE: On average, your model's predictions have an error of approximately 2.075 units of the stock's closing price.

RMSE: The RMSE value is approximately 2.237, indicating the square root of the mean squared error. It provides a measure of the typical prediction error, with larger errors having a greater impact on the result.

Lower values of MAE and RMSE indicate better prediction performance, meaning that the model's predictions are closer to the actual values. These evaluation metrics provide a quantitative measure of how well your stock price prediction model is performing.

CODE EXAMPLE: (python)

```
import pandas as pd
import statsmodels.api as sm

# Load preprocessed data into a DataFrame
data = pd.read_csv('stock_data.csv')

# Fit an ARIMA model (p, d, q)
p = 1  # Autoregressive order
d = 1  # Differencing order
q = 1  # Moving average order
model = sm.tsa.ARIMA(data['Close Price'], order=(p, d, q))
results = model.fit()

# Make predictions
forecast = results.forecast(steps=30)  # Forecast the next 30 days, for example

print(forecast)
```

In the above code,

- Panda Library is included.
- stock_data.csv is the dataset collected from the historical record.
- **model = sm.tsa.ARIMA(data['Close Price'], order=(p, d, q))** – ARIMA model is used and this line is including ARIMA algorithm.

CONCLUSION:

In Phase 1 of this project, we successfully defined the problem and employed the principles of Design Thinking to guide our approach. We started by understanding the significance of stock price prediction and outlined the objectives of the project.

In Phase 2, we'd explore more advanced deep learning techniques like CNN-LSTM or attention mechanisms for improved accuracy in predicting stock prices.