Stock Price Prediction - Phase 4: Development Part 2

In Phase 4, we will continue building the stock price prediction model. This phase involves feature engineering, model training, and evaluation to create an accurate and reliable model for predicting stock prices.

Dataset

We will continue to use the Microsoft Lifetime Stocks Dataset available at the following link: Microsoft Lifetime Stocks Dataset.

Phase 4 Tasks

1. **Feature Engineering**

Feature engineering is a crucial step in model development. We will create or modify features to improve the model's predictive power. Some feature engineering tasks include:

Lagged Variables: Create lag features that represent the stock price at previous time steps. This allows the model to capture temporal dependencies.

Moving Averages: Calculate and add moving average features with different time windows (e.g., 7-day, 30-day) to capture trends and smooth out noise.

Technical Indicators: Incorporate technical indicators like Relative Strength Index (RSI), Moving Average Convergence Divergence (MACD), and Bollinger Bands to provide additional insights into market behavior.

2. **Model Training**

With the newly engineered features, we will proceed to train the stock price prediction model. The steps involved in model training include:

Data Split: Divide the dataset into training and validation sets. The training set is used to train the model, and the validation set is used to monitor its performance.

Model Selection: Choose the most suitable regression model based on the dataset's characteristics. We may consider models like Linear Regression, Random Forest Regressor, or a deep learning model.

Hyperparameter Tuning: Fine-tune the hyperparameters of the selected model to optimize its performance.

Training: Train the model on the training data, utilizing the features generated during the feature engineering phase.

3. **Evaluation**

Model evaluation is essential to assess its performance and ensure it meets the project's objectives. We will use appropriate regression metrics to evaluate the model, such as:

Mean Absolute Error (MAE): This measures the average absolute difference between predicted and actual stock prices.

Mean Squared Error (MSE): This measures the average squared difference between predicted and actual stock prices.

R-squared (R2): This metric quantifies the proportion of variance in stock prices explained by the model. A higher R2 indicates a better fit.

By evaluating the model's performance using these metrics, we can determine its accuracy and suitability for predicting stock prices.

Code:

```
# Import libraries

import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

import matplotlib.pyplot as plt


# Load the dataset (Replace 'your_dataset.csv' with the actual dataset file)

data = pd.read_csv('your_dataset.csv')


# Feature Engineering

# Example: Creating lag features for the past 7 days

data['lag_1'] = data['stock_price'].shift(1)

data['lag_2'] = data['stock_price'].shift(2)

data['lag_3'] = data['stock_price'].shift(3)

data['lag_4'] = data['stock_price'].shift(4)

data['lag_5'] = data['stock_price'].shift(5)

data['lag_6'] = data['stock_price'].shift(6)

data['lag_7'] = data['stock_price'].shift(7)


# Split the data into features (X) and target (y)
```

```python
X = data.drop(['stock_price'], axis=1)

y = data['stock_price']


# Split the data into training and validation sets

X_train, X_valid, y_train, y_valid = train_test_split(X, y, test_size=0.2, random_state=0)


# Model Selection (Random Forest Regressor)

model = RandomForestRegressor(n_estimators=100, random_state=0)


# Model Training

model.fit(X_train, y_train)


# Model Predictions

predictions = model.predict(X_valid)


# Evaluation

mae = mean_absolute_error(y_valid, predictions)

mse = mean_squared_error(y_valid, predictions)

r2 = r2_score(y_valid, predictions)


# Print evaluation metrics

print("Mean Absolute Error:", mae)

print("Mean Squared Error:", mse)

print("R-squared:", r2)


# Visualize actual vs. predicted prices

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

plt.plot(y_valid.values, label='Actual Prices', color='b')

plt.plot(predictions, label='Predicted Prices', color='r')
```

```
plt.legend()

plt.title('Stock Price Prediction')

plt.xlabel('Time')

plt.ylabel('Price')

plt.show()
```

Conclusion

In Phase 4, we will continue to build the stock price prediction model by performing feature engineering to enhance the dataset, training the model with the engineered features, and evaluating its performance. This phase is critical in ensuring the model's accuracy and reliability, which is essential for making informed investment decisions.