Predicting Amazon Stock Price Using Market Data

Project Overview

This project explores the possibility of predicting Amazon's (AMZN) stock price using various financial and market indicators. By analyzing historical data and employing machine learning models, we aim to uncover the relationships between these indicators and Amazon's price movement.

Technologies Used

- **Python**: Core language for data manipulation and model development.
- **Pandas**: Facilitates data cleaning and preprocessing tasks.
- **NumPy**: Enables efficient numerical operations on data.
- **Matplotlib & Seaborn**: Create informative visualizations of the data.
- **scikit-learn**: Provides a comprehensive toolbox for machine learning model development and evaluation.
- **Jupyter Notebook**: Serves as an interactive environment for code execution, data exploration, and documentation.

Key Features

- **Data Preprocessing**: Handles missing values and standardizes features for model compatibility.
- **Exploratory Data Analysis (EDA)**: Visualizes data distributions and explores correlations between features to understand their influence on Amazon's price.
- **Model Development**: Utilizes linear regression as the initial model for predicting AMZN prices.
- **Cross-Validation**: Employs 5-fold cross-validation to assess model performance across different data subsets, enhancing generalizability.
- **Feature Importance**: Analyzes the impact of each feature on the model's prediction using linear regression coefficients.

Project Workflow

Data Exploration & Preprocessing

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler

# Load dataset
data = pd.read_csv(r"D:\Projects\HUBBLEMIND\Stock Market
Dataset.csv")

# Display the first few rows of the dataset
```

Predicting Amazon Stock Price Using Market Data

```
print(data.head())

# Check for missing values and data types
print(data.info())
print(data.isnull().sum())

# Remove non-numeric columns
data = data.select_dtypes(exclude=['object'])

# Display dataset after dropping non-numeric columns
print(data.head())

# Summary statistics
print(data.describe())
```

Data Visualization

```
# Visualizing the distribution of Amazon's price
plt.figure(figsize=(10, 6))
sns.histplot(data['Amazon_Price'], kde=True)
plt.title('Distribution of Amazon Price')
plt.xlabel('Amazon Price')
plt.ylabel('Frequency')
plt.show()

# Scatter plot to visualize the relationship between Crude Oil Price
and Amazon Price
plt.figure(figsize=(10, 6))
sns.scatterplot(x='Crude_oil_Price', y='Amazon_Price', data=data)
plt.title('Crude Oil Price vs Amazon Price')
plt.xlabel('Crude Oil Price')
plt.ylabel('Amazon Price')
plt.show()
```

Model Development

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error,
mean_squared_error

# Prepare data for modeling
X = data.drop(columns=['Amazon_Price'])
y = data['Amazon_Price']

# Handling missing values by filling with median
X.fillna(X.median(), inplace=True)
```

Predicting Amazon Stock Price Using Market Data

```
# Split 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)

# Train a linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

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

# Evaluate the model
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = mse**0.5

print(f'Mean Absolute Error: {mae}')
print(f'Mean Squared Error: {mse}')
print(f'Root Mean Squared Error: {rmse}')
```

Model Validation & Testing

```
from sklearn.model_selection import cross_val_score

# Apply 5-fold cross-validation
cv_scores = cross_val_score(model, X, y, cv=5,
scoring='neg_mean_squared_error')
mean_cv_score = -cv_scores.mean()

print(f'Mean Cross-Validation Score: {mean_cv_score**0.5}')
```

Feature Importance

```
# Get feature importance from linear regression coefficients
feature_importance = pd.Series(model.coef_,
index=X.columns).sort_values(ascending=False)

plt.figure(figsize=(10, 8))
feature_importance.plot(kind='bar')
plt.title('Feature Importance')
plt.xlabel('Features')
plt.ylabel('Coefficient')
plt.show()
```

Predicting Amazon Stock Price Using Market Data

Results

- **Model Accuracy**: The model achieved an RMSE of `7.584481133349242` on the test data, indicating a reasonable level of accuracy.
- **Cross-Validation**: Cross-validation resulted in a mean score of `29.505778768485264`, suggesting good model generalizability.
- **Feature Importance**: Analysis revealed that Nasdaq 100 Price has the strongest correlation with AMAZON's price, followed by Crude Oil Price and S&P 500 Index.

How to Run the Project

1. **Clone the repository from GitHub**:
   bash
   ```
   git clone https://github.com/your-username/your-repo-name.git
   cd your-repo-name
   ```
2. **Install required dependencies**:
   bash
   ```
   pip install -r requirements.txt
   ```
3. **Launch Jupyter Notebook**:
   bash
   ```
   jupyter notebook
   ```
4. **Open the notebook and run the cells sequentially to reproduce the results.**

Project Insights

- **Data-Driven Approach**: Demonstrates a method to predict AMZN stock price using market indicators.
- **Baseline Model**: Linear regression provides initial insights.
- **Model Generalisation**: Cross-validation confirms the model's ability to generalise beyond the training data.
- **Feature Engineering**: Significant improvements in model accuracy through preprocessing and feature analysis.

Next Steps

- **Feature Expansion**: Explore additional financial indicators to enhance prediction accuracy.
- **Advanced Models**: Implement models like Random Forest or Gradient Boosting for potentially better results.
- **Hyperparameter Tuning**: Optimise model parameters to further reduce error metrics.

Predicting Amazon Stock Price Using Market Data

Conclusion

This project lays the groundwork for predicting Amazon's stock price using market indicators. The linear regression model, validated through cross-validation, achieved a reasonable level of accuracy. Future work includes feature expansion and exploring more advanced machine learning models.

Links

- GitHub Repository Link
- Project Report