Linear Regression for Stock Market Prediction

Amit Yadav Follow 5 min read · Jul 29, 2024

  1   

I understand that learning data science can be really challenging...

...especially when you are just starting out.

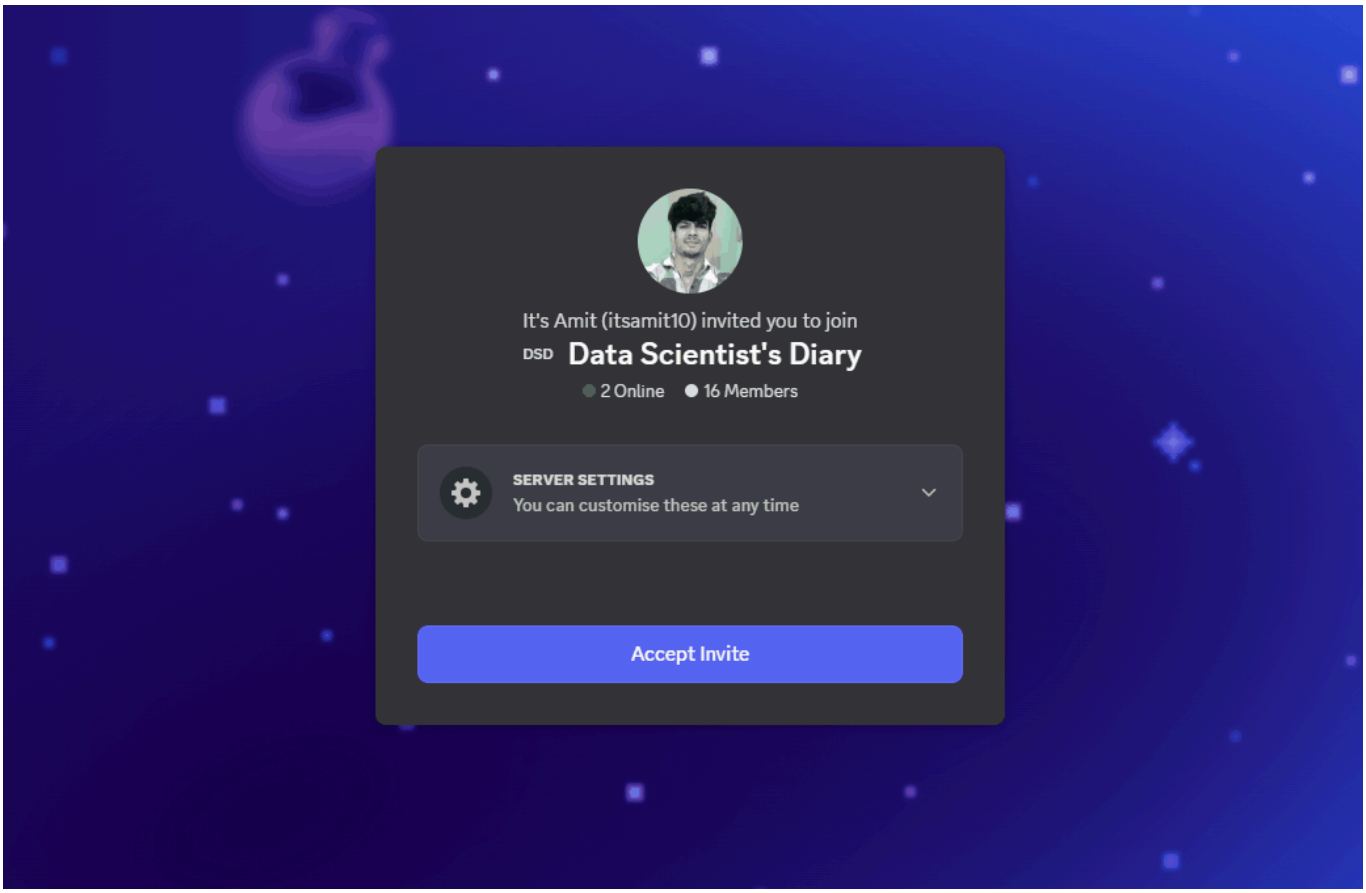
But it doesn't have to be this way.

That's why I spent weeks creating a **46-week Data Science Roadmap** with projects and study resources for getting your first data science job.

Here's what it contains:

Complete roadmap with study resources

A **Discord community** to help our data scientist buddies get access to study resources, projects, and job referrals.



Like what you're seeing?

Click here to access everything!

Now, let's get back to the blog:

Predicting stock market prices is a critical task in finance and investment. Accurate predictions can lead to substantial profits, while poor predictions can result in significant losses. One of the simplest yet effective methods to

start with is Linear Regression. In this guide, you'll learn how to use Linear Regression to predict stock prices. We'll walk through the process step-by-step, from setting up your environment to training your model and making predictions.

Understanding Linear Regression

Linear Regression is a statistical method used to model the relationship between a dependent variable (target) and one or more independent variables (features). The goal is to find the line that best fits the data points. The equation for a simple linear regression model is:

$$y = \beta 0 + \beta 1\, x + \epsilon$$

Where:

- y is the dependent variable.
- x is the independent variable.
- β0 is the intercept.
- β1 is the slope of the line.
- ϵ is the error term.

How it Works in Predictive Modeling

Linear regression fits a line to the data points in such a way that the sum of the squared differences (errors) between the actual data points and the predicted data points is minimized. This line, known as the regression line, can then be used to make predictions.

Why It's Suitable for Stock Market Prediction

Linear Regression is suitable for stock market prediction because it can capture linear trends and patterns in historical data. While it may not account for all the complexities of the stock market, it provides a solid foundation for building more advanced models.

Setting Up the Environment

To get started, you need to set up your Python environment with the necessary libraries. You can install these libraries using pip:

```
pip install numpy pandas matplotlib scikit-learn yfinance
```

Then, import the required libraries in your Python script:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import yfinance as yf
```

Data Collection

Sources for Stock Market Data

There are various sources where you can download historical stock market data. Some popular ones include Yahoo Finance, Alpha Vantage, and Quandl. In this guide, we'll use Yahoo Finance for its simplicity and ease of use.

Downloading Data

Let's download historical data for Apple Inc. (AAPL) using the Yahoo Finance API:

```
ticker = 'AAPL'
data = yf.download(ticker, start='2020-01-01', end='2023-01-01')
print(data.head())
```

This code downloads the stock data from January 1, 2020, to January 1, 2023, and prints the first few rows of the dataset.

Data Preprocessing

Cleaning the Data

The raw data may contain missing values that need to be handled before analysis. We can drop rows with missing values:

```
data.dropna(inplace=True)
```

Feature Selection

Selecting relevant features is crucial for building an effective model. For this example, we'll use the following features: Open, High, Low, Volume, and some moving averages.

Feature Engineering

Creating new features from the existing ones can help improve the model's performance. Let's create 20-day and 50-day moving averages:

```
data['SMA_20'] = data['Close'].rolling(window=20).mean()
data['SMA_50'] = data['Close'].rolling(window=50).mean()
data.dropna(inplace=True)
```

Splitting the Data

To evaluate the model's performance, we need to split the data into training and testing sets:

```
X = data[['Open', 'High', 'Low', 'Volume', 'SMA_20', 'SMA_50']]
y = data['Close']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
```

Training the Linear Regression Model

Now, let's train the Linear Regression model on the training data:

```
model = LinearRegression()
model.fit(X_train, y_train)
```

Model Evaluation

After training the model, we need to evaluate its performance using the testing data:

```
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')
```

The Mean Squared Error (MSE) and R-squared (R^2) metrics help us understand how well the model has performed. A lower MSE and a higher R^2 indicate a better fit.

Making Predictions

We can use the trained model to make predictions on new data. Assume `future_data` contains the features for future dates:

```
future_data = pd.DataFrame({
    'Open': [values],
    'High': [values],
    'Low': [values],
    'Volume': [values],
    'SMA_20': [values],
    'SMA_50': [values]
})
future_predictions = model.predict(future_data)
print(future_predictions)
```

Replace `[values]` with the actual values for the features.

Visualizing the Results

Visualizing the predicted versus actual stock prices helps us understand the model's performance better:

```
plt.figure(figsize=(10,5))
plt.plot(y_test.index, y_test, label='Actual')
plt.plot(y_test.index, y_pred, label='Predicted')
plt.xlabel('Date')
plt.ylabel('Stock Price')
plt.title('Actual vs Predicted Stock Prices')
plt.legend()
plt.show()
```

This plot shows the actual stock prices and the prices predicted by the model, providing a visual comparison.

Conclusion

In this guide, you learned how to use Linear Regression to predict stock market prices. We covered everything from setting up the environment to collecting and preprocessing data, training the model, making predictions, and visualizing the results. While Linear Regression is a good starting point, it's essential to explore more complex models and additional features to improve prediction accuracy.

Linear Regression | Stock Market Prediction

Written by Amit Yadav

1.3K followers · 141 following

Get Data Science Roadmap for Your First Data Science Job :
https://amit404.gumroad.com/l/ds-diary

Follow