Project – AI & ML

Stock Price Prediction – Machine Learning Project in Python

Machine learning has significant applications in the stock price prediction.

In this machine learning project, we will be talking about predicting the returns on stocks.

Category: Machine Learning

Programming Language: Python

IDE: Jupyter

Prerequisites: Python, Machine Learning

Intended Audience: Education, Developers, Data Engineers, Data Scientists

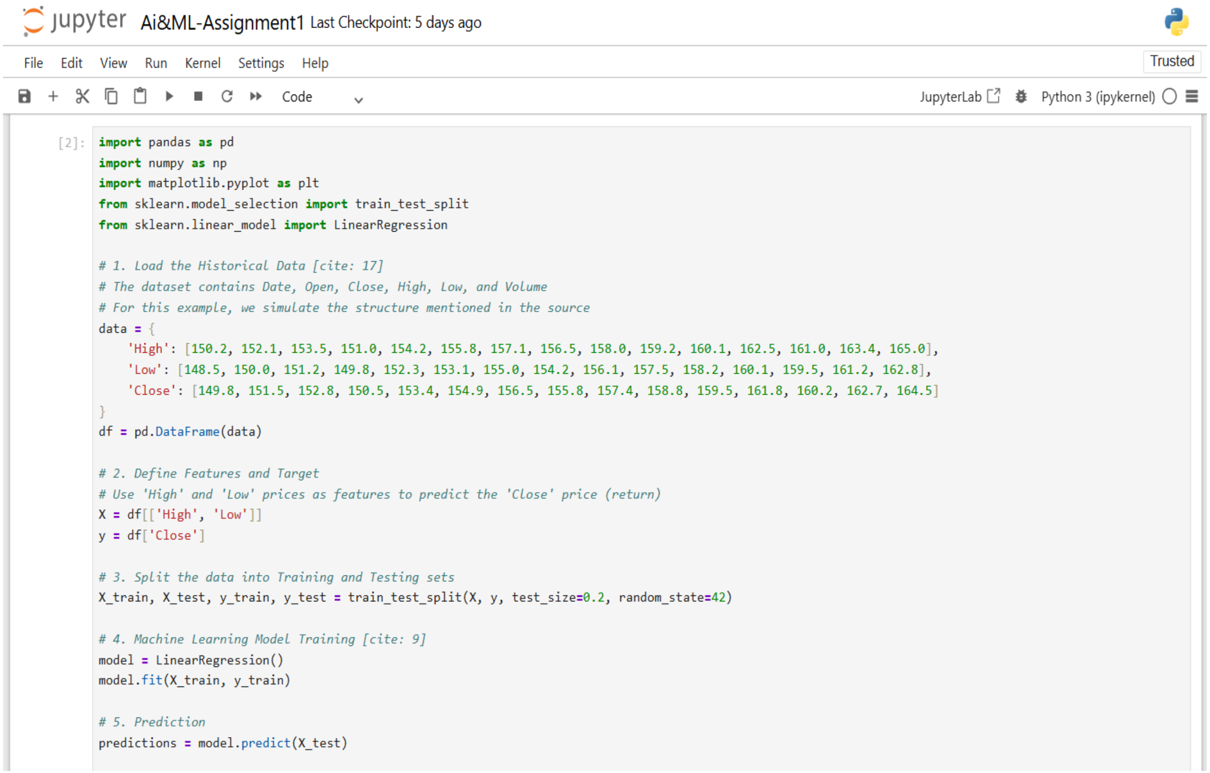
Stocks Price Data

The historical data contains records about the stock price of various stocks like Apple, Microsoft, Facebook. The dataset also contains a date-wise price of stock with open, close, high, and low prices along with volume traded on that day.

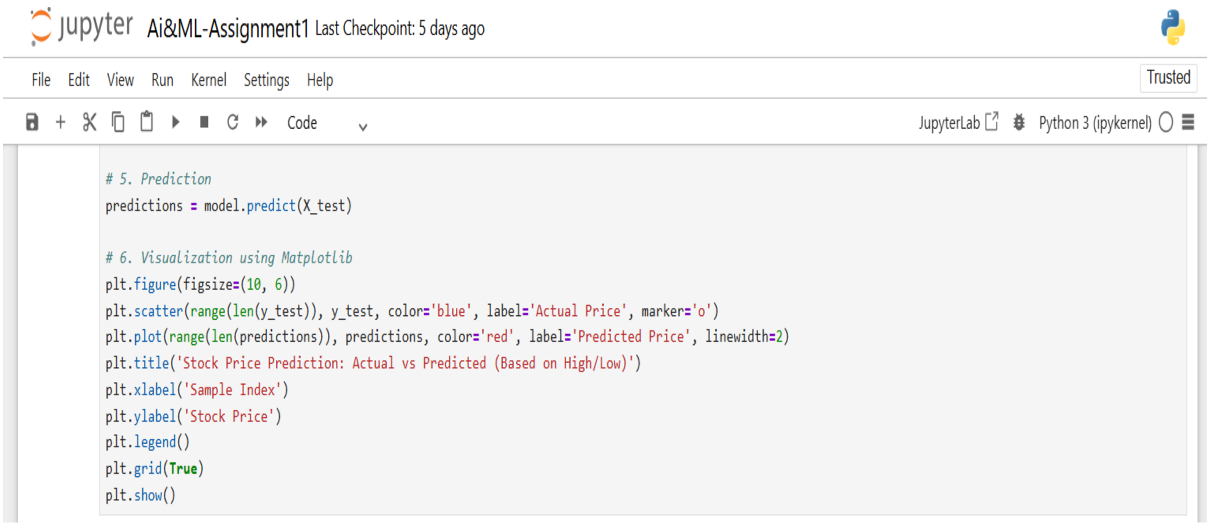
It is an excellent database for people who want to try learning techniques of data visualization, data analytics, and many different forms of data processing.

Find the prediction of stocks based on the columns of high and low prices and visualize it using matplotlib library.

Code:


jupyter Ai&ML-Assignment1 Last Checkpoint: 5 days ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[2]:

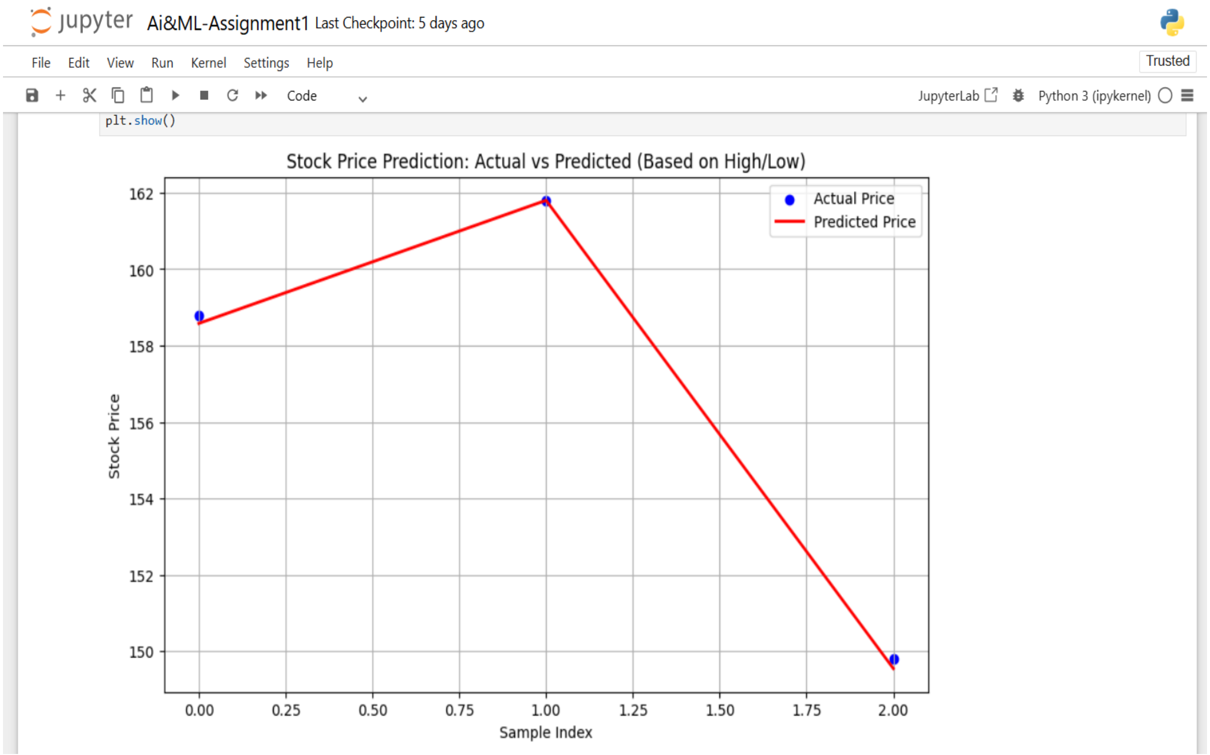
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

# 1. Load the Historical Data [cite: 17]
# The dataset contains Date, Open, Close, High, Low, and Volume
# For this example, we simulate the structure mentioned in the source
data = {
    'High': [150.2, 152.1, 153.5, 151.0, 154.2, 155.8, 157.1, 156.5, 158.0, 159.2, 160.1, 162.5, 161.0, 163.4, 165.0],
    'Low': [148.5, 150.0, 151.2, 149.8, 152.3, 153.1, 155.0, 154.2, 156.1, 157.5, 158.2, 160.1, 159.5, 161.2, 162.8],
    'Close': [149.8, 151.5, 152.8, 150.5, 153.4, 154.9, 156.5, 155.8, 157.4, 158.8, 159.5, 161.8, 160.2, 162.7, 164.5]
}
df = pd.DataFrame(data)

# 2. Define Features and Target
# Use 'High' and 'Low' prices as features to predict the 'Close' price (return)
X = df[['High', 'Low']]
y = df['Close']

# 3. 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)

# 4. Machine Learning Model Training [cite: 9]
model = LinearRegression()
model.fit(X_train, y_train)

# 5. Prediction
predictions = model.predict(X_test)
```



jupyter
Ai&ML-Assignment1 Last Checkpoint: 5 days ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

```
# 5. Prediction
predictions = model.predict(X_test)

# 6. Visualization using Matplotlib
plt.figure(figsize=(10, 6))
plt.scatter(range(len(y_test)), y_test, color='blue', label='Actual Price', marker='o')
plt.plot(range(len(predictions)), predictions, color='red', label='Predicted Price', linewidth=2)
plt.title('Stock Price Prediction: Actual vs Predicted (Based on High/Low)')
plt.xlabel('Sample Index')
plt.ylabel('Stock Price')
plt.legend()
plt.grid(True)
plt.show()
```

Output:



jupyter
Ai&ML-Assignment1 Last Checkpoint: 5 days ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

```
plt.show()
```