Nghiên cứu dự báo giá chứng khoán Tesla

0.1 Giới thiệu bài toán, dữ liệu, mô hình

Bài toán dự báo giá chứng khoán là một bài toán quan trọng trong lĩnh vực tài chính và kinh doanh. Mục tiêu của bài toán này là dự báo giá chứng khoán trong tương lai dựa trên dữ liệu lịch sử.

Dữ liệu sử dụng cho bài toán này là dữ liệu giá chứng khoán của công ty Tesla (TSLA), bao gồm các thông tin như giá mở cửa, giá đóng cửa, giá cao nhất, giá thấp nhất và số lượng chứng khoán giao dịch. Dữ liệu được thu thập từ trang web *Kaggle.com* và đã được xử lý để loại bỏ các giá trị thiếu và ngoại lệ.

Trong bài báo cáo này, chúng tôi sử dụng các mô hình học máy để dự báo giá chứng khoán TSLA trong tương lai. Đặc biệt, chúng tôi sử dụng mô hình Linear Regression, Decision Tree Regressor và Random Forest Regressor để dự báo và đánh giá chúng

0.2 Giải quyết

Để giải quyết bài toán, chúng tôi tiến hành các bước sau:

1. Đọc và xử lý dữ liệu

   Tải dữ liệu giá chứng khoán của TSLA vào một pandas DataFrame và chuyển đổi cột "Date" sang dạng datetime, điều này sẽ giúp cho việc xử lý và đánh giá dữ liệu dễ dàng hơn. Tiến hành xử lý các giá trị thiếu và ngoại lệ bằng cách loại bỏ chúng.

```
# Load the data into a pandas DataFrame
df = pd.read_csv("TSLA.csv")

# Convert the Date column to a datetime data type
df["Date"] = pd.to_datetime(df["Date"])

# Handle missing values
df = df.dropna()

# Check for and handle outliers in all columns
for column in df.columns:
    if column == "Date":
        continue
    mean = df[column].mean()
    std = df[column].std()
    df = df[(df[column] > mean - 2*std) & (df[column] < mean + 2*std)]
```

2. Chuẩn hóa dữ liệu

   Chuẩn hóa tất cả các cột trong DataFrame bằng cách trừ giá trị nhỏ nhất của cột đó và chia cho khoảng giá trị giữa giá trị lớn nhất và nhỏ nhất của cột đó.

```
# Normalize the data
for column in df.columns:
    if column == "Date":
        continue
    df[column] = (df[column] - df[column].min()) / (df[column].max() - df[colum
```

3. Tách dữ liệu

   Tách dữ liệu thành tập huấn luyện và tập kiểm tra bằng cách chia dữ liệu thành 80% tập huấn luyện và 20% tập kiểm tra.

```
# Split the data into training and testing sets
train_df = df.iloc[:int(len(df)*0.8)]
test_df = df.iloc[int(len(df)*0.8):]
```

Sau khi tách dữ liệu được dùng để tính toán và hiển thị ma trận tương quan giữa các cột trong DataFrame.

Hàm corr() được sử dụng để tính toán các chỉ số tương quan giữa các cột, và hàm sns.heatmap() được sử dụng để tạo ra một heatmap để hiển thị chỉ số tương quan giữa các cột.

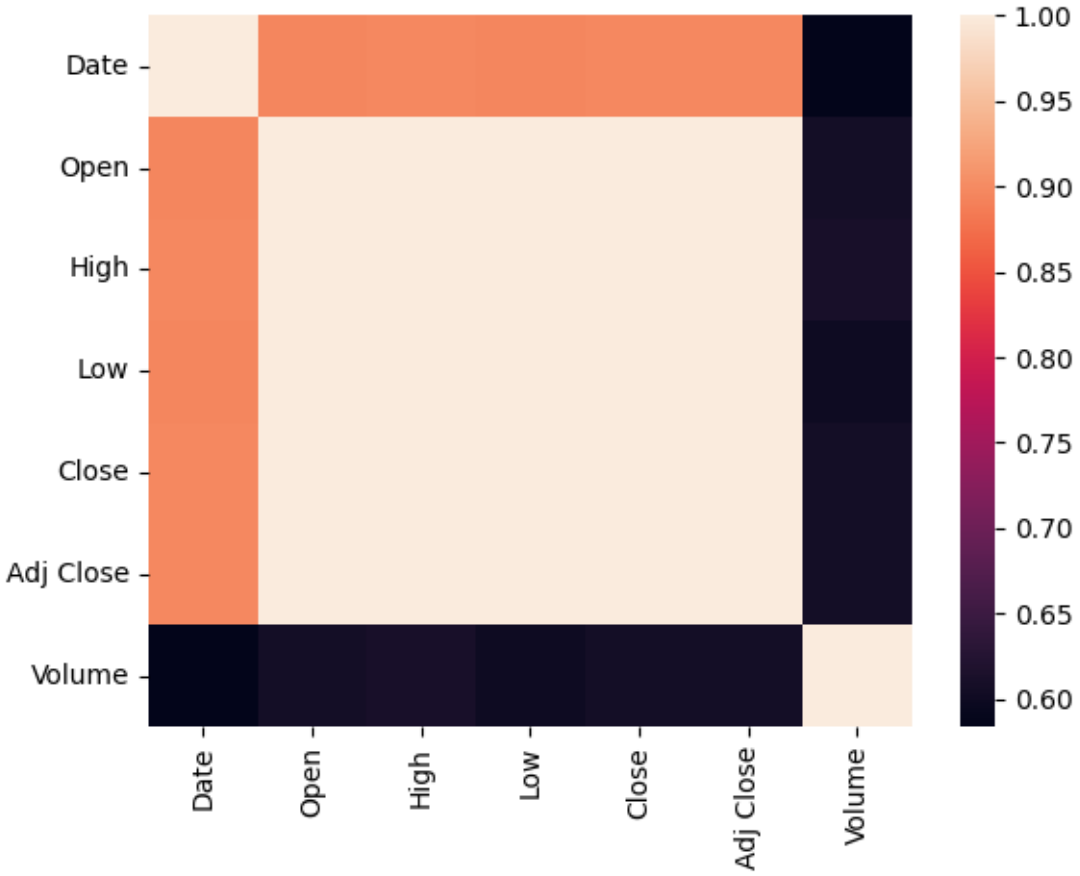
Cụ thể hơn, hàm corr() được gọi trên DataFrame đã chuẩn hóa và tách dữ liệu, để tính toán chỉ số tương quan giữa tất cả các cột trong DataFrame, chú ý tham số numeric_only=False để tính chỉ số tương quan giữa cả dữ liệu số và chuỗi.

```
# Calculate the correlations between the columns
corr_matrix = df.corr(numeric_only=False)

# Create a heatmap of the correlations
sns.heatmap(corr_matrix)

# Show the plot
plt.show()
```

Sử dụng hàm sns.heatmap để hiển thị chúng trong dạng một heatmap. Heatmap này sẽ cho ta một cái nhìn tổng quát về mối quan hệ giữa các cột trong DataFrame.

Các màu sắc từ xanh lá đến đỏ sẽ biểu thị mức độ tương quan từ thấp đến cao giữa các cột.

Chúng ta có thể dùng heatmap này để tìm ra các cột có mối quan hệ cao với nhau, hoặc cột có mối quan hệ thấp với các cột khác.



4. Đào tạo và đánh giá mô hình

   Sử dụng các mô hình Linear Regression, Decision Tree Regressor và Random Forest Regressor để huấn luyện với tập huấn luyện và đánh giá với tập kiểm tra.

   Chúng tôi sử dụng chỉ số mean squared error và root mean squared error để đánh giá chất lượng của các mô hình.

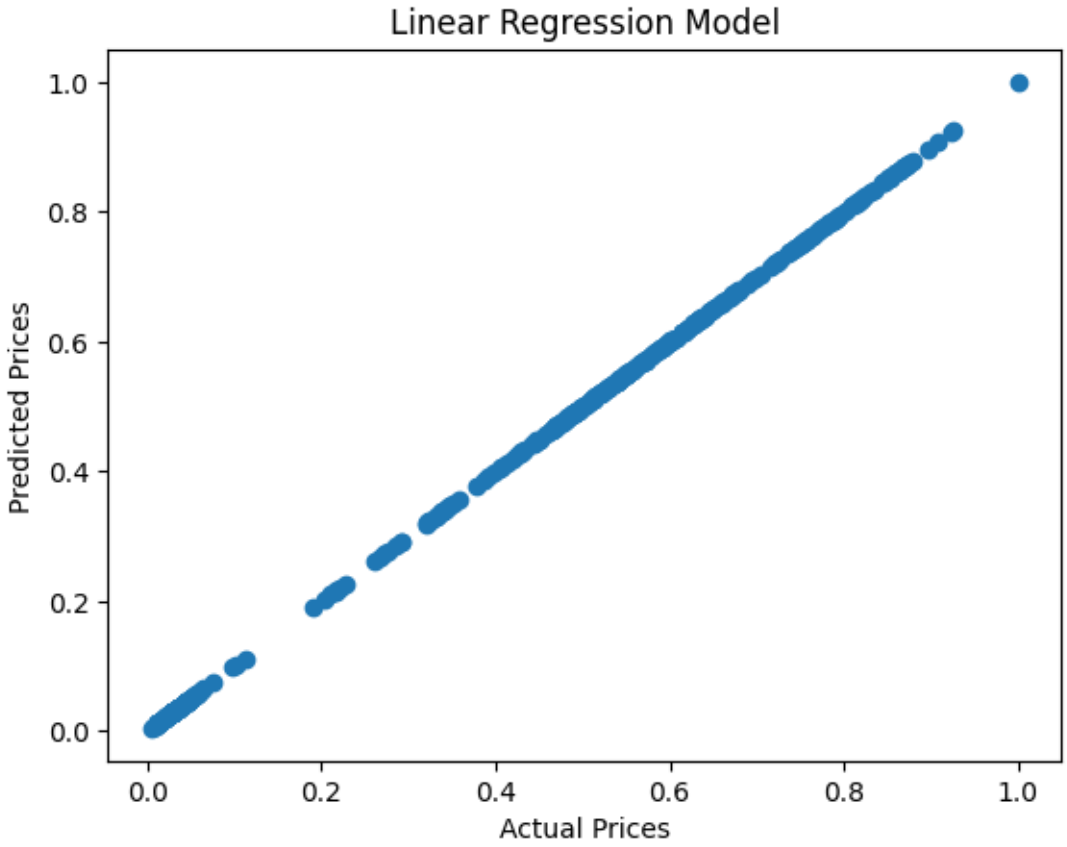
```
# Split the data into features and targets
X = df.drop(columns=["Date"])
y = df["Close"]

# 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)

# Choose a machine learning model
model = LinearRegression()

# Train the model
model.fit(X_train, y_train)

# Evaluate the model
print("LinearRegression: ", model.score(X_test, y_test))
```

```
# Decision Tree Model
dt_model = DecisionTreeRegressor()
dt_model.fit(X_train, y_train)
dt_predictions = dt_model.predict(X_test)
dt_mse = mean_squared_error(y_test, dt_predictions)
dt_rmse = np.sqrt(dt_mse)
print("Decision Tree RMSE: ", dt_rmse)

# Random Forest Model
rf_model = RandomForestRegressor()
rf_model.fit(X_train, y_train)
rf_predictions = rf_model.predict(X_test)
rf_mse = mean_squared_error(y_test, rf_predictions)
rf_rmse = np.sqrt(rf_mse)
print("Random Forest RMSE: ", rf_rmse)
```

Đây là kết quả sau khi tiến hành đánh giá các mô hình:

```
LinearRegression:  1.0
Decision Tree RMSE:  0.0008072430883783102
Random Forest RMSE:  0.0007884836129544221
```

Vậy, ta chọn mô hình LinearRegression để giải quyết bài toán của chúng ta.



0.3 Ứng dụng trong thực tế

Kết quả của bài báo cáo này có thể được sử dụng để dự báo giá chứng khoán trong tương lai và đưa ra các quyết định đầu tư hợp lý.

```python
# Collect new data that contains future dates and any other relevant features
future_data = pd.read_csv("future_data.csv")

# If applicable, preprocess the new data in the same way that you preprocessed
future_data["Date"] = pd.to_datetime(future_data["Date"])
future_data = future_data.dropna()
for column in future_data.columns:
    if column == "Date":
        continue
    future_data[column] = (future_data[column] - future_data[column].min())
    / (future_data[column].max() - future_data[column].min())

# Use the predict method on the trained model to make predictions on the new d
future_predictions = model.predict(future_data.drop(columns=["Date"]))

# Use the resulting predictions in a meaningful way
print(future_predictions)
```

Tuy nhiên, hiện tại chưa tìm được dữ liệu Stock Tesla 2020-present, nên đoạn code trên chỉ là demo sau khi chúng ta có được dữ liệu tương ứng. Các mô hình học máy có thể được sử dụng để dự báo giá chứng khoán của các công ty khác và cả trong các lĩnh vực khác nhau. Ngoài ra, kết quả của bài báo cáo này cũng có thể được sử dụng để cải thiện và tinh chỉnh các mô hình học máy cho các bài toán tương tự.