

MindX School

07/07/2024 - FINAL PROJECT

# GOLD PRICE PREDICTION

Chau Nguyen

Mentor Tang Vi Thai





# Nội dung trình bày

- 1 Giới thiệu về dữ liệu
- 2 Xử lý và làm sạch dữ liệu
- 3 Phân tích dữ liệu
- 4 Lựa chọn và huấn luyện mô hình
- 5 Kết quả và kết luận

# 1. Giới thiệu về dữ liệu

## Dataset

- Bộ dataset được thu thập từ trang Investing.com, trong đó có 13 chỉ số quan trọng có ảnh hưởng đến dự báo giá vàng.
- Thời gian dữ liệu:
  - Từ 1/1/2012 đến 03/07/2024

## Chi tiết

- Gold Future Contract
- Silver Future
- Platinum Future
- Brent Oil
- Eldorado Gold Corporation
- USD/VND Exchange Rate
- United States Oil Fund
- VNIndex (Vietnam)
- DAX Index (European)
- Hang Seng Index (Hongkong)
- Nikken 225 (Japan)
- Dow Jones (US)
- S&P500 (US(



# Tổng quan dữ liệu

3150 dòng và 41 cột

	Date	Close	Open	High	Low	Volume	Change
0	3/7/2024	2,372.45	2,339.45	2,372.85	2,335.95	NaN	1.67%
1	2/7/2024	2,333.40	2,341.60	2,346.10	2,327.40	153.71K	-0.24%
2	1/7/2024	2,338.90	2,338.90	2,348.80	2,328.20	142.60K	-0.03%
3	28/06/2024	2,339.60	2,338.60	2,350.60	2,329.70	135.38K	0.13%
4	27/06/2024	2,336.60	2,309.40	2,342.00	2,306.80	140.23K	1.55%
...	...	...	...	...	...	...	...
3145	9/1/2012	1,608.10	1,617.70	1,624.60	1,605.70	134.39K	-0.54%
3146	6/1/2012	1,616.80	1,622.40	1,632.30	1,609.00	155.21K	-0.20%
3147	5/1/2012	1,620.10	1,614.40	1,626.80	1,597.70	176.47K	0.46%
3148	4/1/2012	1,612.70	1,604.90	1,619.80	1,593.80	154.22K	0.76%
3149	3/1/2012	1,600.50	1,571.00	1,608.70	1,566.80	112.94K	2.15%

Silver_Close	Silver_Volume	Silver_Change	VNI_Close	VNI_Volume	VNI_Change
30.84	75.32K	3.99%	1,276.85	589.51K	0.56%
29.66	52.27K	0.15%	1,269.79	528.47K	1.21%
29.61	45.99K	0.18%	1,254.56	487.19K	0.74%
29.56	51.91K	1.04%	1,245.32	840.43K	-1.09%
29.26	63.25K	0.57%	1,259.09	596.99K	-0.17%
...	...	...	...	...	...
28.78	30.25K	0.35%	339.32	171.19K	0.77%
28.68	38.61K	-2.09%	336.73	29.08K	-1.23%
29.30	39.30K	0.68%	340.94	20.63K	-2.26%
29.10	36.12K	-1.61%	348.84	25.45K	-0.33%
29.57	35.83K	5.94%	350	20.64K	-0.44%

## 2. Xử lý và làm sạch dữ liệu

### Thông tin dữ liệu

0	Date	3150	non-null	object
1	Close	3150	non-null	object
2	Open	3150	non-null	object
3	High	3150	non-null	object
4	Low	3150	non-null	object
5	Volume	3147	non-null	object
6	Change	3150	non-null	object
7	Silver_Close	3150	non-null	float64
8	Silver_Volume	3094	non-null	object
9	Silver_Change	3150	non-null	object
10	VNI_Close	3021	non-null	object
11	VNI_Volume	3021	non-null	object
12	VNI_Change	3021	non-null	object
13	EGO_Closing	3069	non-null	float64
14	EGO_Volume	3069	non-null	object
15	EGO_Change	3069	non-null	object
16	USD_VND_Rate	3107	non-null	object
17	USD_VND_Change	3107	non-null	object
18	USO_Close	3070	non-null	float64
19	USO_Volume	3069	non-null	object
...				
39	DJ_Volume	3069	non-null	object
40	DJ_Change	3069	non-null	object

### Xử lý null

Silver_Volume	56	Platin_Volume	2447
Silver_Change	0	Platin_Change	0
VNI_Close	129	Hseng_Close	147
VNI_Volume	129	Hseng_Volume	152
VNI_Change	129	Hseng_Change	147
EGO_Closing	81	Nikkei_Close	0
EGO_Volume	81	Nikkei_Volume	194
EGO_Change	81	Nikkei_Change	0
USD_VND_Rate	43	Dax_Close	57
USD_VND_Change	43	Dax_Volume	58
USO_Close	80	Dax_Change	57
USO_Volume	81	SP_Close	81
USO_Change	80	SP_Change	81

- Xóa cột Platin\_volume do dữ liệu bị null nhiều
- Forward fill hay backward fill các tệp dữ liệu null còn lại

### Kết quả


```
df.isnull().values.any()
✓ 0.0s
False
```



# 2. Xử lý và làm sạch dữ liệu

## Bộ dữ liệu cuối cùng

0	Date	3149	non-null	datetime64[ns]
1	Close	3149	non-null	float64
2	Open	3149	non-null	float64
3	High	3149	non-null	float64
4	Low	3149	non-null	float64
5	Volume	3149	non-null	float64
6	Change	3149	non-null	float64
7	Silver_Close	3149	non-null	float64
8	Silver_Volume	3149	non-null	float64
9	Silver_Change	3149	non-null	float64
10	VNI_Close	3149	non-null	float64
11	VNI_Volume	3149	non-null	float64
12	VNI_Change	3149	non-null	float64
13	EGO_Closing	3149	non-null	float64
14	EGO_Volume	3149	non-null	float64
15	EGO_Change	3149	non-null	float64
16	USD_VND_Rate	3149	non-null	float64
17	USD_VND_Change	3149	non-null	float64
18	USO_Close	3149	non-null	float64
19	USO_Volume	3149	non-null	float64
...				
38	DJ_Volume	3149	non-null	float64
39	DJ_Change	3149	non-null	float64

- 
- Chuyển đổi cột Date sang dạng datetime
  - Sắp xếp thứ tự ngày tháng năm theo thứ tự tăng dần

- Chuyển đổi dữ liệu chứa K,M,B về dạng số
- Chuyển tỷ lệ % sang dạng số
- Chuyển các cột dữ liệu số thành dạng "float64"

**Data đã sạch, không còn dữ liệu bị thiếu và null**

# 3. PHÂN TÍCH DỮ LIỆU

## Mô tả dữ liệu

	Date	Close	Open	High	Low	Volume	Change
count	3150	3,150.00	3,150.00	3,150.00	3,150.00	3,149.00	3,150.00
mean	2018-04-03 04:15:05.142857216	1,531.30	1,531.49	1,542.08	1,520.31	207,545.75	0.02
min	2012-01-03 00:00:00	1,049.60	1,051.50	1,062.70	1,045.40	390.00	-9.34
25%	2015-02-18 06:00:00	1,264.80	1,265.20	1,272.80	1,256.83	145,490.00	-0.45
50%	2018-04-04 12:00:00	1,437.20	1,448.00	1,459.70	1,423.85	188,890.00	0.02
75%	2021-05-18 18:00:00	1,791.15	1,790.97	1,802.68	1,780.07	247,720.00	0.53
max	2024-07-03 00:00:00	2,449.50	2,442.30	2,464.50	2,425.60	816,530.00	5.95
std	NaN	310.74	310.68	313.22	308.07	90,350.38	0.98

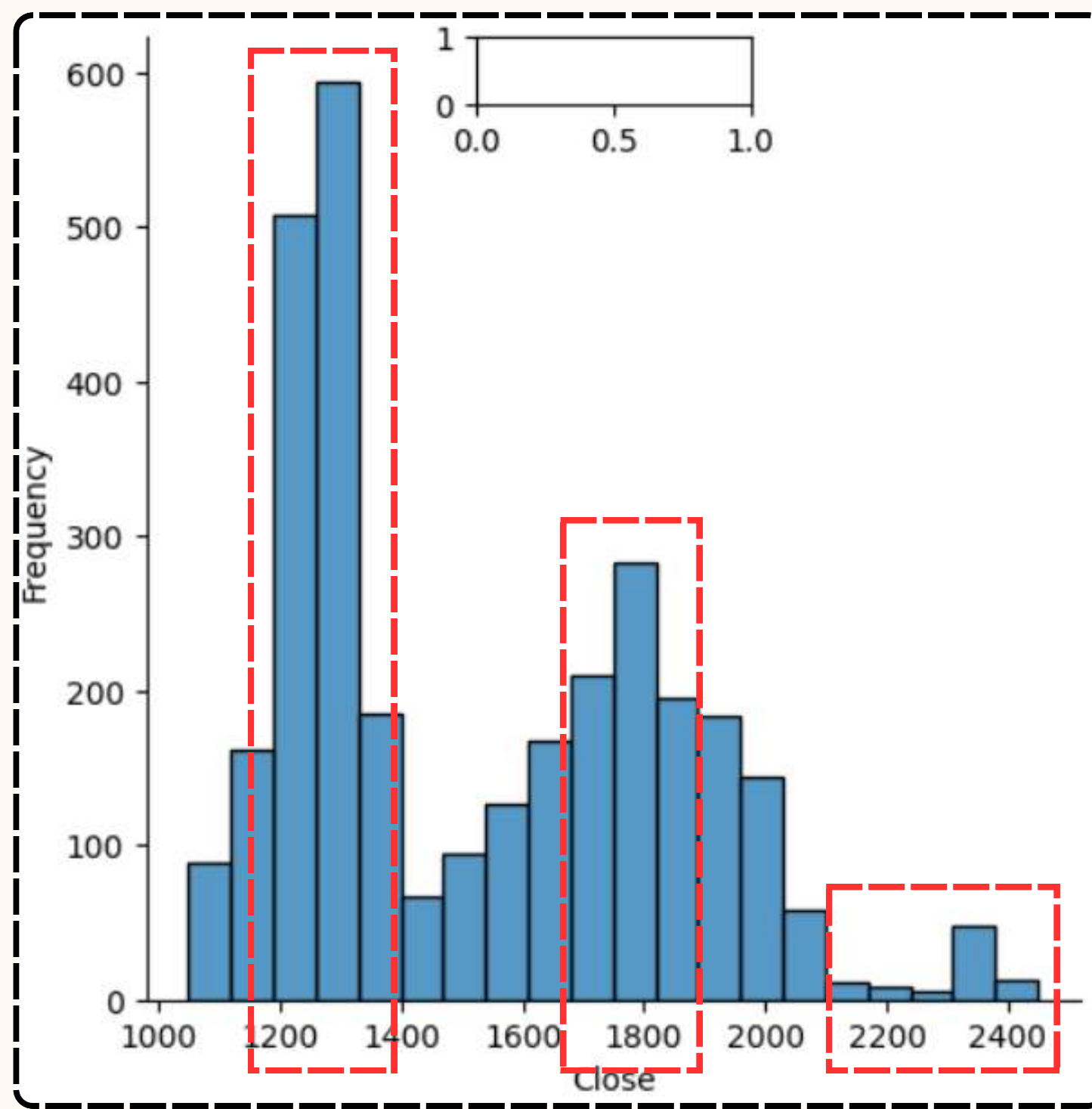
## GIẢI ĐOẠN 2012-2024

Giá vàng:	USD
• Cao nhất (25/05/2024)	2449.5
• Thấp nhất (17/12/2015)	1049.6
• Đa số	1791.15
Khối lượng giao dịch:	Contract
• Cao nhất (09/11/2015)	816,530
• Thấp nhất (18/06/2024)	390
• Trung bình	207.545
Tỷ lệ tăng/giảm trong ngày	%
• Tăng mạnh nhất	5.95
• Giảm mạnh nhất	9.34



### 3. PHÂN TÍCH DỮ LIỆU

**Giá vàng dao động nhiều nhất  
và ít nhất ở vùng giá nào?**





### 3. PHÂN TÍCH DỮ LIỆU



**Biểu đồ chuyển động của giá vàng qua các năm**

### 3. PHÂN TÍCH DỮ LIỆU

#### Kiểm tra tương quan giữa các biến với giá vàng

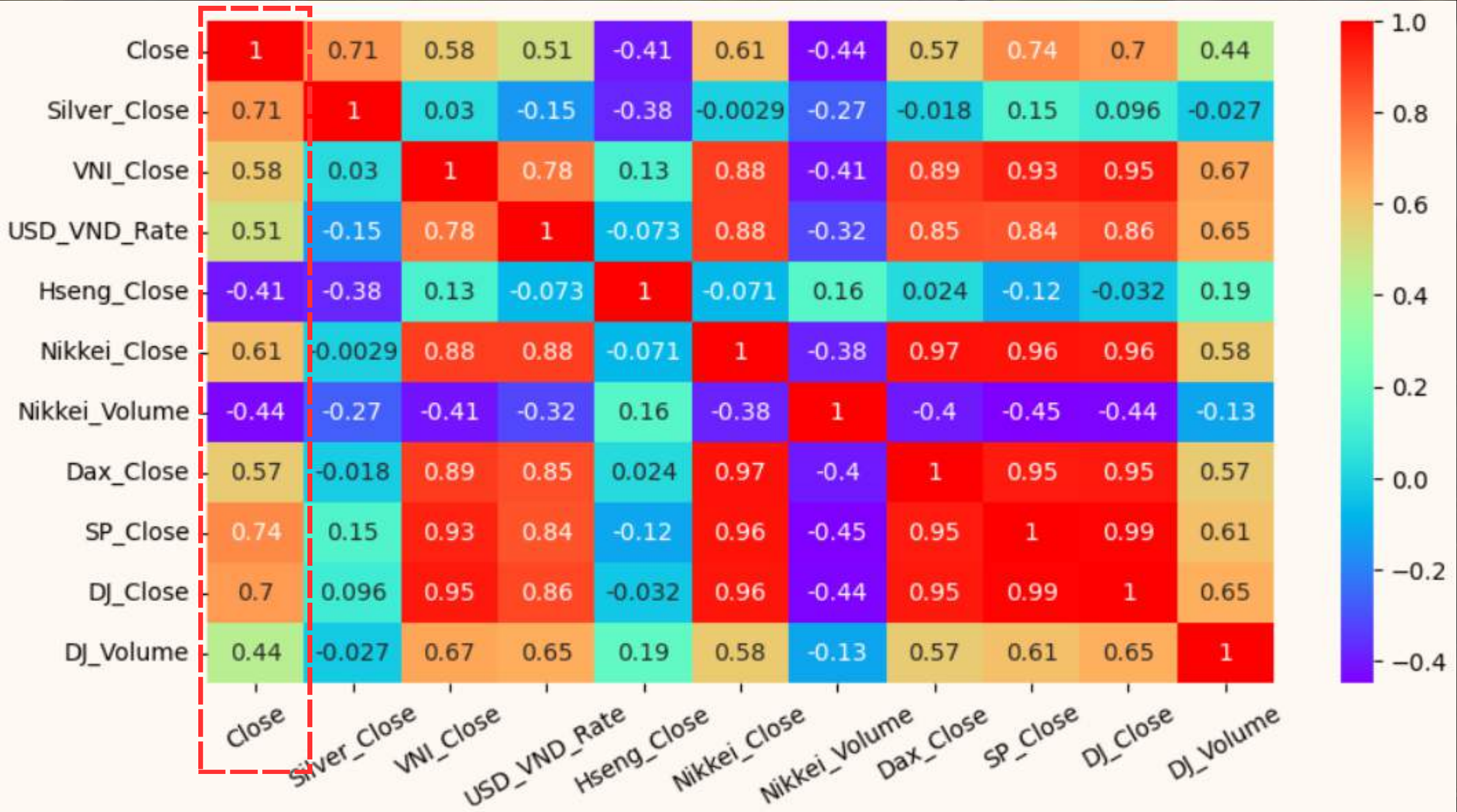
- \* Các chỉ số SP, Silver, Nikkei, VNI, Dax, USD\_VND, Hengseng có tỷ lệ tương quan với giá vàng khá nhiều (>35%)
- \* Các chỉ số USO, Brient\_Oil,... có độ tương quan thấp  
=> Xóa bớt các cột có độ tương quan nhỏ hơn 35%

Bảng hệ số tương quan của các biến

Close	1.00
Low	1.00
High	1.00
Open	1.00
SP_Close	0.74
Silver_Close	0.71
DJ_Close	0.70
Date	0.65
Nikkei_Close	0.61
VNI_Close	0.58
Dax_Close	0.57
USD_VND_Rate	0.51
DJ_Volume	0.44
Nikkei_Volume	0.44
Hseng_Close	0.41
Hseng_Volume	0.32
USO_Close	0.30
Brient_Oil_Close	0.25
EGO_Volume	0.21
USO_Volume	0.18
Dax_Volume	0.15
Silver_Volume	0.07
Brient_Oil_Volume	0.05



# 3. PHÂN TÍCH DỮ LIỆU



# 4. Lựa chọn và huấn luyện mô hình

## MÔ HÌNH LỰA CHỌN

- Linear Regression
- DecisionTree Regression, XGBRegression, RandomForest Regression
- ARIMA (Autoregressive Integrated Moving Average)
- LSTM (Long Short Term Memory networks)

4



# DecisionTree Regression, XGBRegression, RandomForest Regression

## QUY TRÌNH HUẤN LUYỆN

- Standard Scaler (sklearn.preprocessing)
- Train, Test Split (sklearn.model\_selection)
- Sử dụng GridSearchCV để train các mô hình (sklearn.model\_selection)
- Huấn luyện mô hình
- Độ tin cậy và độ lệch chuẩn

```
def find_best_model(X1, y1):
    algos = {
        'decision_tree': {
            'model': DecisionTreeRegressor(),
            'param': {
                'criterion': ['mse', 'friedman_mse'],
                'splitter': ['best', 'random']}},
        'random_forest': {
            'model': RandomForestRegressor(),
            'param': {
                'n_estimators': [10, 50, 100, 130],
                'criterion': ['squared_error', 'absolute_error', 'friedman_mse', 'poisson'],
                'max_depth': range(2, 4, 1),
                'max_features': ['auto', 'log2']}},
        'xgb_regressor': {
            'model': XGBRegressor(),
            'param': {
                'learning_rate': [0.5, 0.1, 0.01, 0.001],
                'max_depth': [2, 3],
                'n_estimators': [10, 50, 100, 200]}}}

    scores = []
    cv = ShuffleSplit(n_splits=5, test_size=0.2, random_state=20)
    for algo_name, config in algos.items():
        gs = GridSearchCV(config['model'], config['param'], cv=cv, return_train_score=False)
        gs.fit(X1, y1)
        scores.append({
            'model': algo_name,
            'best_score': gs.best_score_,
            'best_params': gs.best_params_
        })
    return pd.DataFrame(scores, columns=['model', 'best_score', 'best_params'])
```

# DecisionTree Regression, XGBRegression, RandomForest Regression

## ( KẾT QUẢ )

	model	best_score	best_params
0	decision_tree	0.99	{'criterion': 'friedman_mse', 'splitter': 'best'}
1	random_forest	0.92	{'criterion': 'friedman_mse', 'max_depth': 3, 'max_features': 'log2', 'n_estimators': 130}
2	xgb_regressor	0.99	{'learning_rate': 0.5, 'max_depth': 3, 'n_estimators': 200}

**R2\_score của 3 mô hình đều lớn hơn 90%**  
**=> Mô hình đang bị overfitting**





# Linear Regression

## ( KẾT QUẢ )

```
mse = mean_squared_error(y1_test, y1_pred)
print(f'Mean Squared Error: {mse}')
r2_score = r2_score(y1_test, y1_pred)
print(f'R2_score: {r2_score}')
```

✓ 0.0s

Mean Squared Error: 0.0476939686270145

R2\_score: 0.9541700618439228

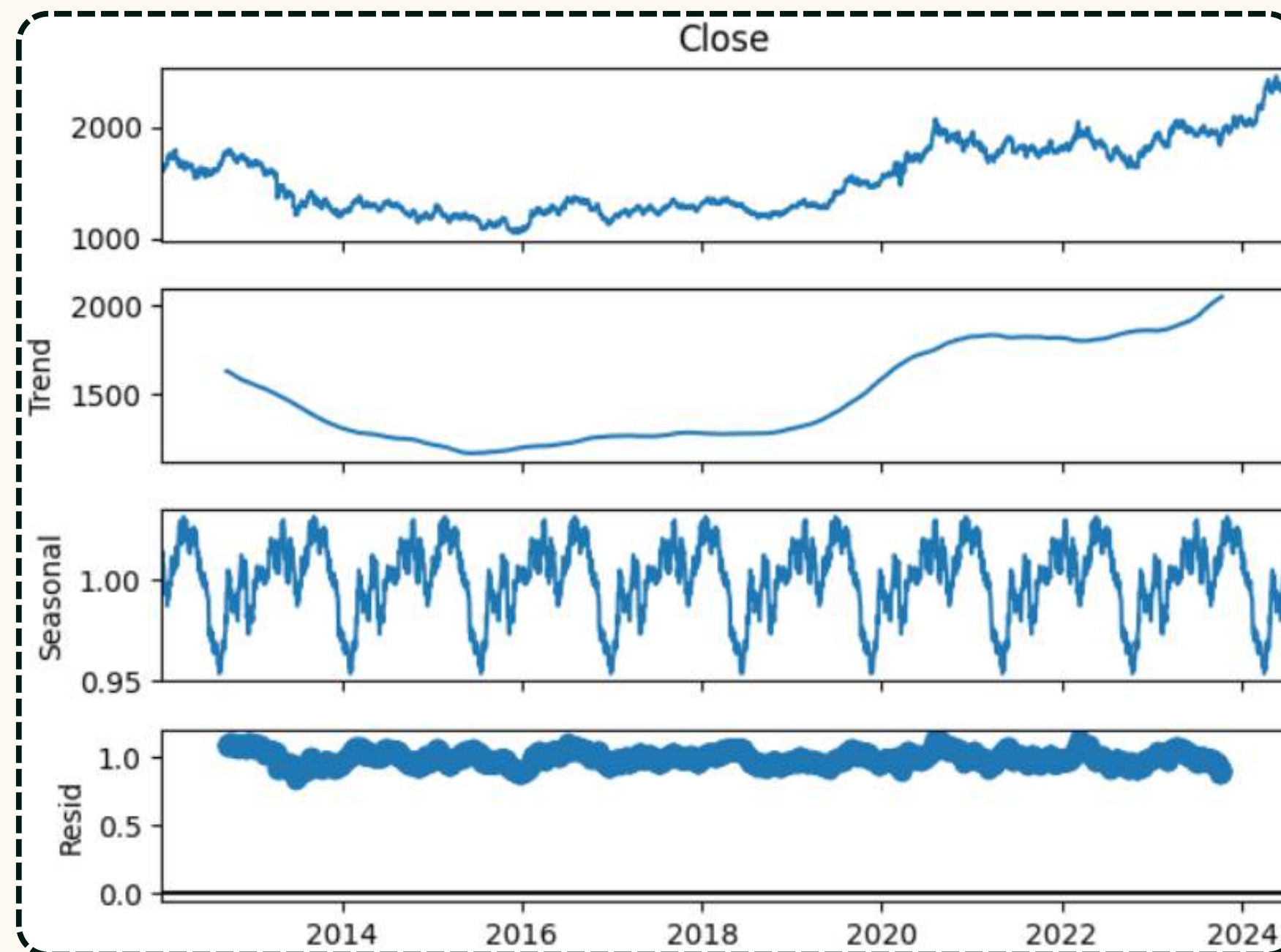
**R2\_score của mô hình Linear Regreesion > 90%**

**=> Mô hình đang bị overfitting**



# MÔ HÌNH ARIMA

## Các thành phần của chuỗi thời gian



- **Trend (Xu hướng):**

1. Trong năm 2013 đến 2016 có sự sụt giảm nhẹ. bắt đầu từ năm 2016 đến 2019 xu hướng tăng nhẹ qua thời gian.
2. Các năm 2019-2022 có sự tăng mạnh.

- **Seasonal (Mùa vụ):**

Xu hướng mùa vụ biến động đều đặn qua các năm

- **Resid (Nhiều trắng)**  
(Hay gọi là yếu tố bất thường)

Hoạt động tốt



# MÔ HÌNH ARIMA

## LOGARITHM

Giảm Biến Động và Biến Đổi Không Tuyến Tính  
Đảm Bảo Tính Bình Ổn (Stationarity) và ảnh hưởng yếu tố ngoại lai

```
df3['Close'] = np.log(df3['Close'])
```

## TRAIN, TEST

Train: 80% giá trị đầu tiên của dataset  
Test: 20% giá trị còn lại

```
train = df3['Close'][: int(len(df3)*0.8)]  
test = df3['Close'][int(len(df3)*0.8):]
```

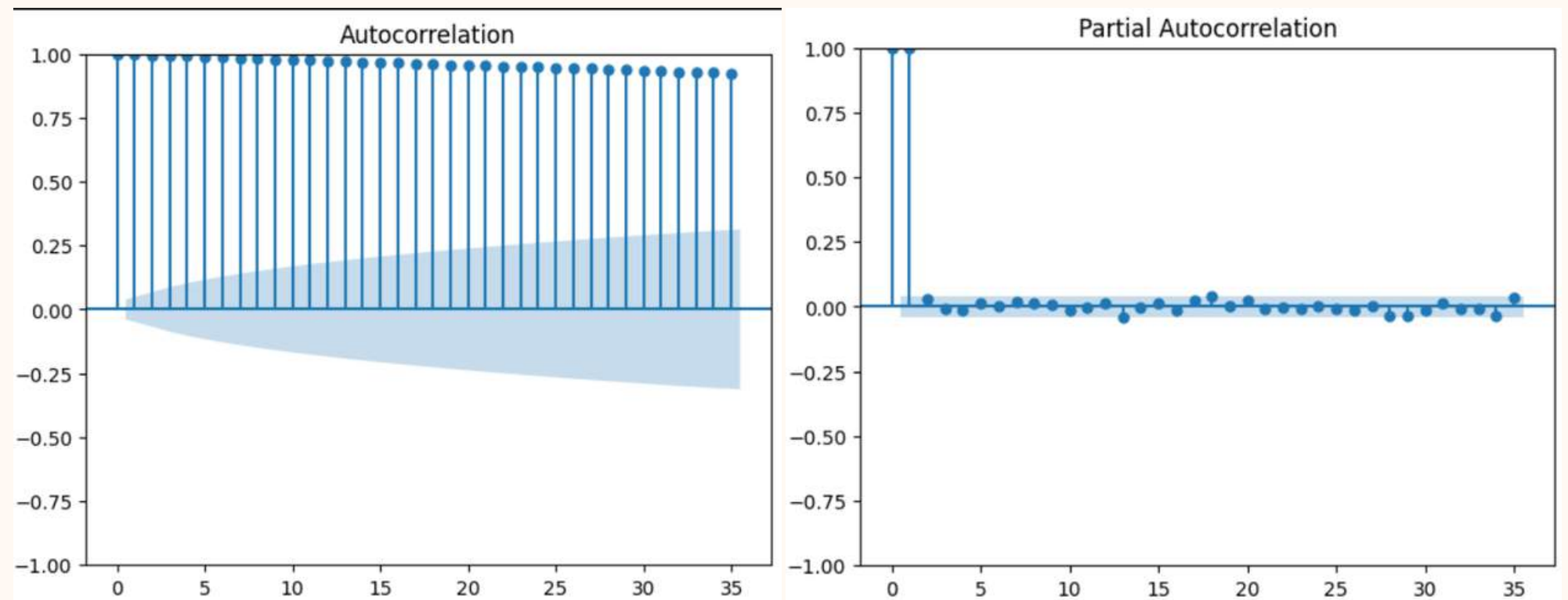
## STATIONARY

Sử dụng kiểm định Dickey và Fuller mở rộng  
(ADF) -> Hàm adfuller

**Không dừng**

# MÔ HÌNH ARIMA

## ACF Plot and PACF Plot



## Check p\_value

```
from statsmodels.tsa.stattools import adfuller
adf_test = adfuller(train)

print(f'p-value: {adf_test[1]}')

if adf_test[1] < 0.05:
    print("Reject the null hypothesis. The time series does not have a unit root (stationary).")
else:
    print("Fail to reject the null hypothesis. The time series has a unit root (non-stationary).")
```

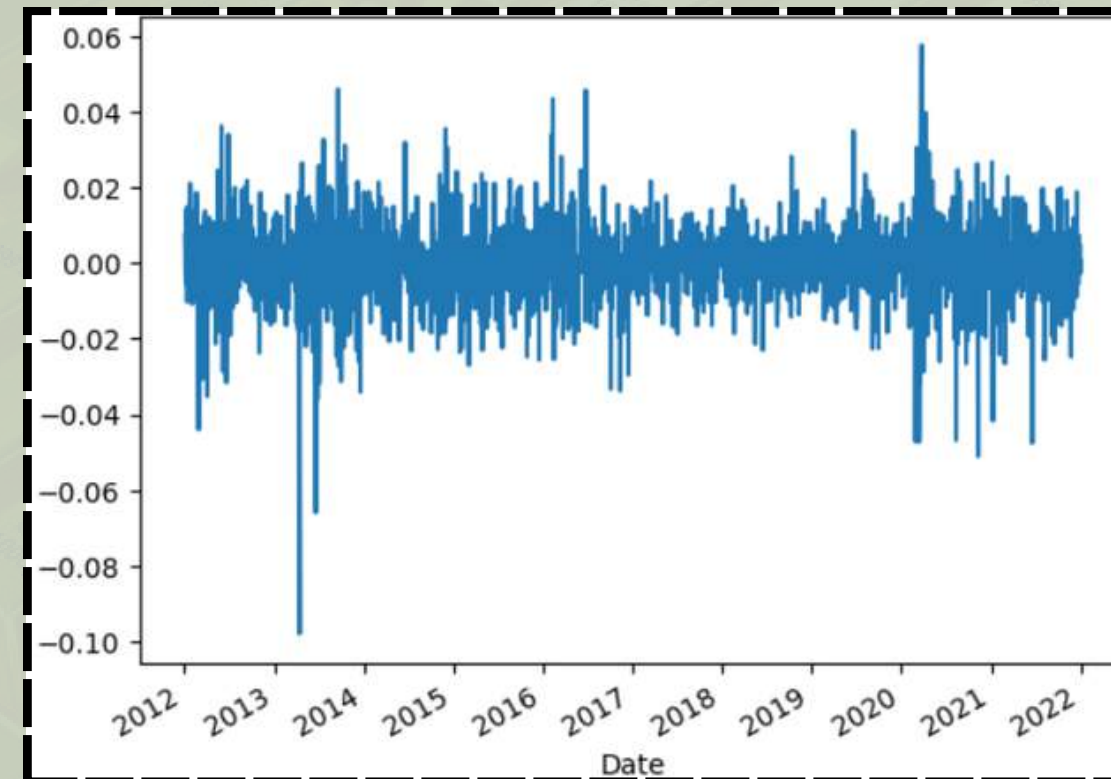
✓ 0.1s

p-value: 0.6824564686927029  
Fail to reject the null hypothesis. The time series has a unit root (non-stationary).

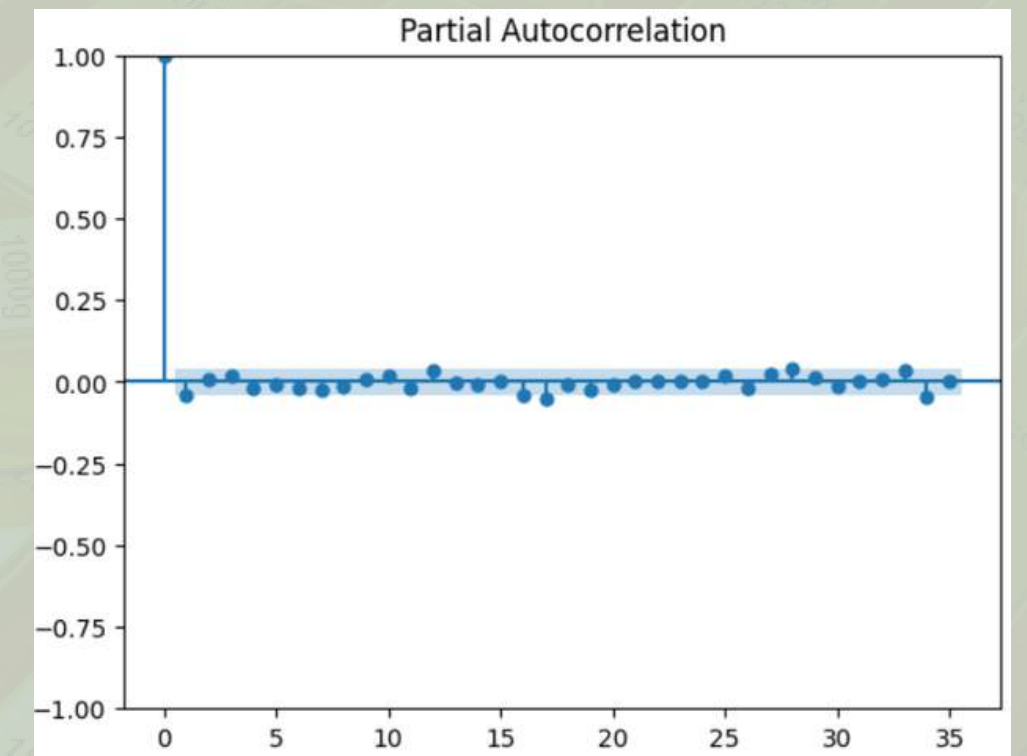
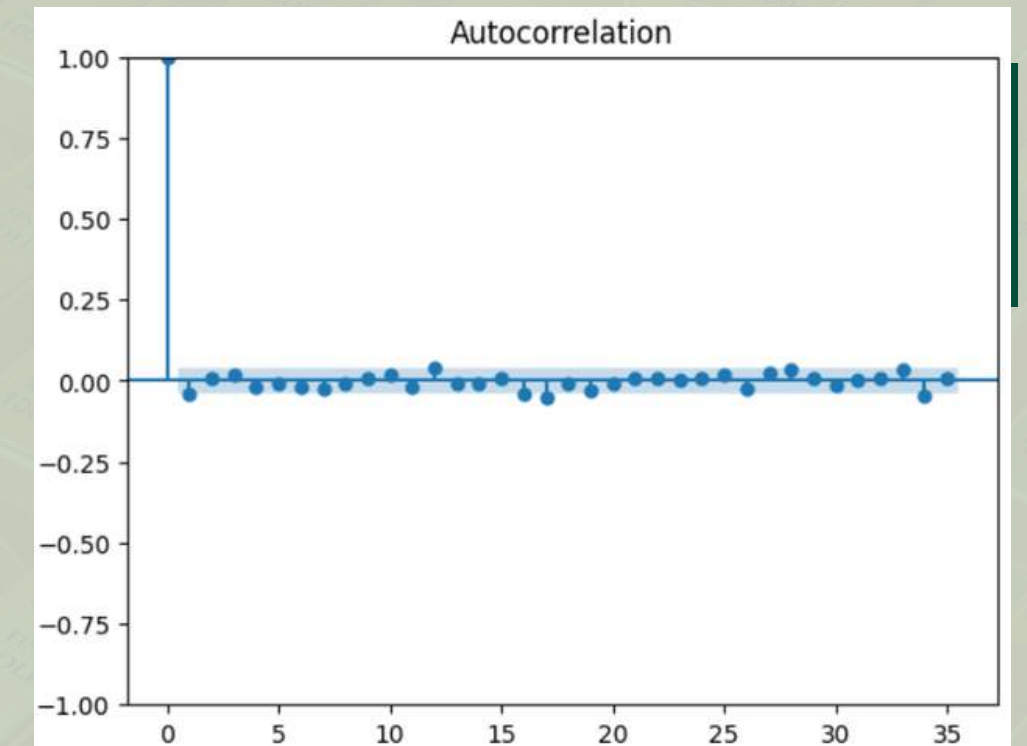
# MÔ HÌNH ARIMA

## TRANSFORM TO STATIONARY: DIFFERENCING

Biểu đồ phần dư trên trục đầu tiên



ACF Plot and PCAF Plot



Check p\_value

```
from statsmodels.tsa.stattools import adfuller
adf_test_diff = adfuller(df3_train_diff)
p_value = adf_test_diff[1]
print(f'p-value: {p_value}')
```

if adf\_test\_diff[1] < 0.05:  
 print("Reject the null hypothesis. The differenced time series is stationary.")  
else:  
 print("Fail to reject the null hypothesis. The differenced time series is non-stationary.")

✓ 0.1s

p-value: 0.0  
Reject the null hypothesis. The differenced time series is stationary.





# MÔ HÌNH ARIMA

## Fit the ARIMA model

SARIMAX Results						
=====						
Dep. Variable:	Close	No. Observations:	2519			
Model:	ARIMA(1, 1, 0)	Log Likelihood	8040.815			
Date:	Sun, 07 Jul 2024	AIC	-16077.631			
Time:	07:46:36	BIC	-16065.968			
Sample:	0	HQIC	-16073.398			
	- 2519					
Covariance Type:	opg					
=====						
	coef	std err	z	P> z	[0.025	0.975]
-----						
ar.L1	-0.0413	0.015	-2.744	0.006	-0.071	-0.012
sigma2	9.858e-05	1.39e-06	70.961	0.000	9.59e-05	0.000
=====						
Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	6771.98			
Prob(Q):	0.99	Prob(JB):	0.00			
Heteroskedasticity (H):	0.82	Skew:	-0.58			
Prob(H) (two-sided):	0.00	Kurtosis:	10.95			
=====						

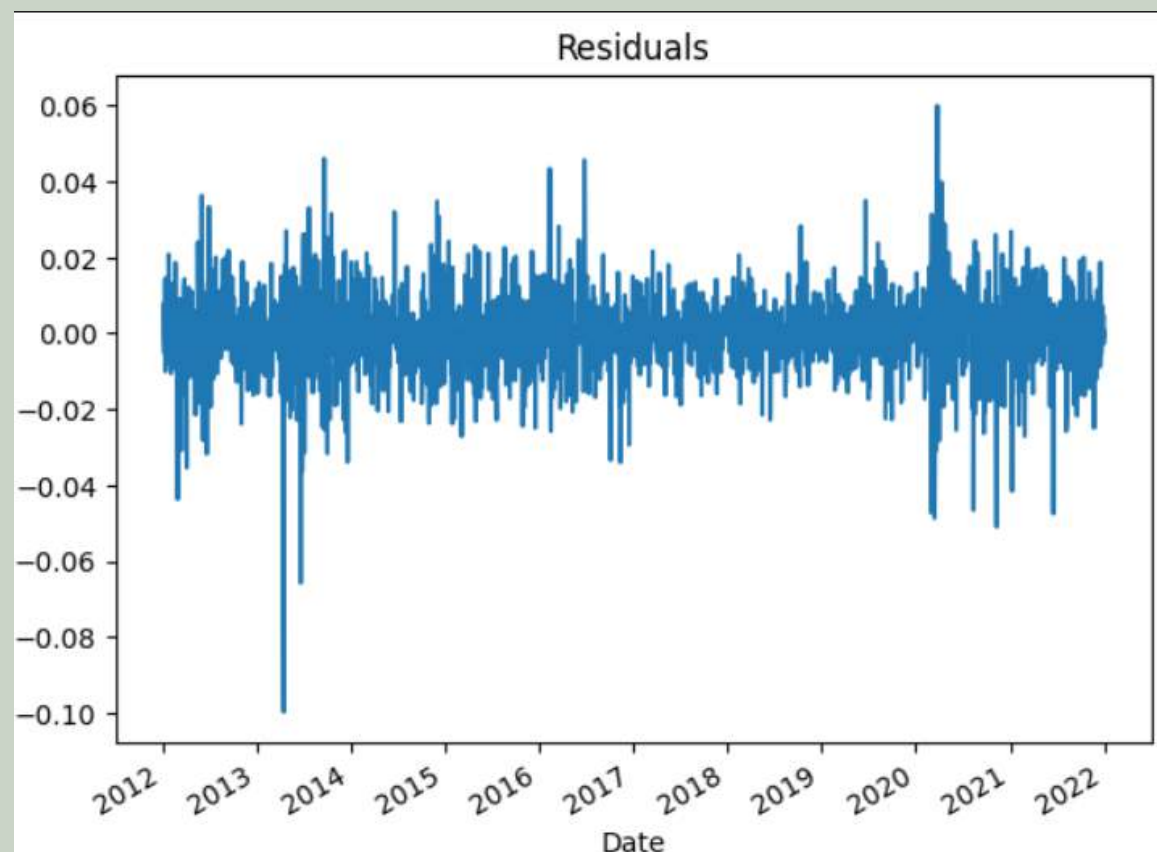
### TRAIN MODEL

Biến phụ thuộc	Close
Số lượng quan sát	2519
Model	ARIMA (1,1,0)

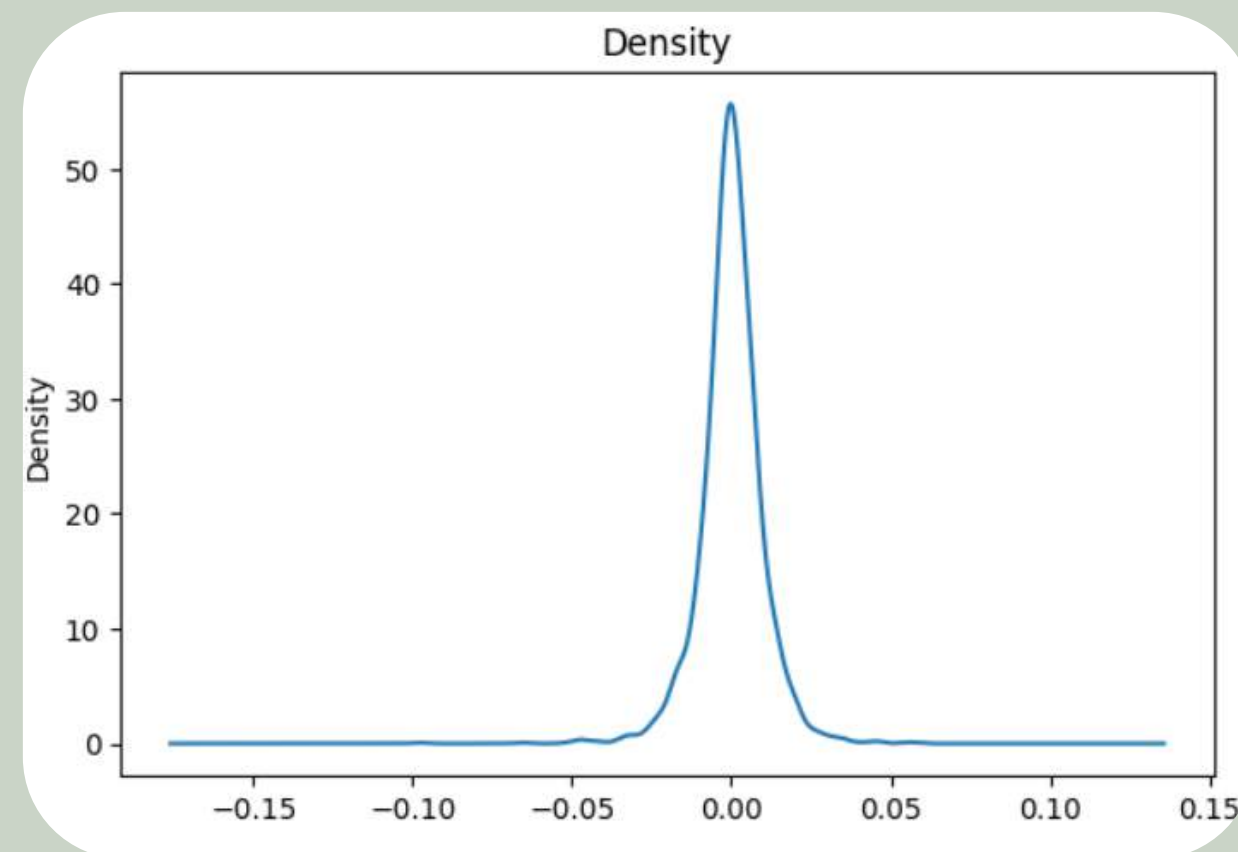
# MÔ HÌNH ARIMA

## Residual

### Mẫu hình của phần dư



### Phân phối của phần dư



### Ghi chú:

- Phần dư không có mẫu hình rõ ràng, điều đó cho thấy mô hình có thể phù hợp với dữ liệu.
- Phần dư phân phối chuẩn đều đặn xung quanh giá trị trung bình là 0.

# MÔ HÌNH ARIMA

## Dự báo cho mô hình



Dự báo không theo kịp các biến động thực tế của dữ liệu kiểm tra.

Hiệu suất của mô hình kiểm tra rõ ràng là không tốt, vì mô hình không dự báo được các biến động và xu hướng trong dữ liệu thực tế.

**=> Mô hình tăng dần theo diễn biến kinh tế ngoài thị trường.**



# MÔ HÌNH ARIMA

```
mape = mean_absolute_percentage_error(np.exp(test), np.exp(forecast))
mae = mean_absolute_error(np.exp(test), np.exp(forecast))
rsme = np.sqrt(mean_squared_error(np.exp(test), np.exp(forecast)))
print(f'Mean_absolute_percentage_error là {mape}')
print(f'Mean_absolute_error là {mae}')
print(f'Mean_squared_error là {rsme}')
```

✓ 0.0s

Mean\_absolute\_percentage\_error là 0.08109741118576926

Mean\_absolute\_error là 167.98063615981

Mean\_squared\_error là 225.36127072226412

- MAPE là 0.081, tương đương với 8.1%. Điều này cho thấy rằng, trung bình, mô hình dự báo với sai số khoảng 8.1% so với giá trị thực tế
- MAE là 167.98, dự báo của mô hình sai lệch khoảng 167.98 USD so với giá trị thực tế
- RMSE là 225.36, đại diện cho độ lệch chuẩn của lỗi dự báo.

# Mô hình LSTM

Train & test  
(80:20)

```
# Tạo vòng lặp các giá trị
X2_train, y2_train = [], []
for i in range(50, len(train)):
    X2_train.append(data[i-50:i, 0])
    y2_train.append(data[i, 0])

print(len(X2_train)) #2096 - 50 dòng
print(len(y2_train))

# Xếp dữ liệu thành 1 mảng:
X2_train = np.array(X2_train)
y2_train = np.array(y2_train)

# Xếp lại dữ liệu thành mảng 1 chiều:
X2_train = np.reshape(X2_train, (X2_train.shape[0], X2_train.shape[1], 1))
y2_train = np.reshape(y2_train, (y2_train.shape[0], 1))
```

Xây dựng  
mô hình

```
# Xây dựng mô hình:
model = Sequential()
model.add(LSTM(units=120, input_shape = (X2_train.shape[1], 1), return_sequences=True))
model.add(LSTM(units=64))
model.add(Dropout(0.5))
model.add(Dense(1))
model.compile(loss='mean_absolute_error', optimizer = 'adam')
```

# Mô hình LSTM

## Huấn luyện mô hình

### Tập train

```
# Huấn luyện mô hình:
# Define the checkpoint
save_model = 'save_model.keras'
best_model = ModelCheckpoint(save_model, monitor='loss', verbose=2, save_best_only=True, mode='auto')

# Fit the model
model.fit(X2_train, y2_train, epochs=100, batch_size=50, verbose=2, callbacks=[best_model])

# dữ liệu train
y2_train = ss.inverse_transform(y2_train) #=> Quy lại giá trị y_train giá ban đầu , convert to the original value
final_model = load_model('save_model.keras')
y_train_pred = final_model.predict(X2_train)
y_train_pred = ss.inverse_transform(y_train_pred)
len(y_train_pred)
```

### Tập test

```
# Xử lý dữ liệu test
X_test = []
for i in range(50, test.shape[0]):
    X_test.append(ss_test[i-50:i,0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1)) ## Hiện tại giá trị đã được scaler

# Dữ liệu test
y_test = df2_Close[int(len(df1)*0.8):].values
y_test_pred = final_model.predict(X_test)
y_test_pred = ss.inverse_transform(y_test_pred) ## Quy trả về giá trị thực
```



# Mô hình LSTM

- Mô hình LSMT dự đoán khá tốt so với mô hình ARIMA
- MAPE là 0.7%, tỷ lệ chênh lệch khá nhỏ giữa giá dự đoán và giá thực tế.
- R2\_square cao do mô hình học tốt trong 50 ngày trước và dự báo tốt cho ngày tiếp theo.



```
Mean_absolute_percentage_error là 0.0071409052105087835
Mean_absolute_error là 13.981750604538693
Mean_squared_error là 18.436004203991832
R2_square là 0.989491184432478
```



MindX School

# Thank for listening

**Chau Nguyen**

Final Project

DA54 \_ Khóa 3 “Data Analyst

