



ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC KINH TẾ - LUẬT

ĐỀ TÀI CUỐI KỲ

# DỰ BÁO GIÁ CARDANO BẰNG MÔ HÌNH ARIMA

Môn: Gói phần mềm ứng dụng trong tài chính 1

Giảng viên hướng dẫn:  
ThS. Ngô Phú Thanh

Sinh viên thực hiện:

Nguyễn Thị Huệ Minh

MSSV: K194141733

Lớp: K19414C

Tp, Hồ Chí Minh, ngày 25 tháng 1 năm 2022

## MỤC LỤC

1. Tổng quan đề tài.....	2
1.1. Lý do chọn đề tài .....	2
1.2. Sơ lược về Cardano.....	2
1.3. Sơ lược về mô hình ARIMA: .....	2
2. Dữ liệu.....	2
2.1. Quy trình lấy dữ liệu.....	2
2.2. Quy trình xử lý dữ liệu .....	3
2.2.1. Xóa các cột không cần thiết.....	3
2.2.2. Phân tích dữ liệu thô.....	3
2.2.3. Kiểm tra tính mùa vụ của dữ liệu.....	4
2.2.4. Kiểm tra tính dừng của dữ liệu .....	5
2.2.5. Khắc phục dữ liệu không dừng.....	6
3. Dự báo giá Carnado bằng mô hình ARIMA .....	7
3.1. Lựa chọn tham số.....	7
3.2. Dự đoán với mô hình ARIMA.....	8
3.2.1. Đào tạo mô hình.....	8
3.2.2. Kiểm định mô hình .....	10
3.2.3. Dự báo giá trong tương lai.....	12
4. Ý nghĩa và kết luận .....	13
4.1. Ý Nghĩa .....	13
4.2. Kết luận .....	13
5. Tài liệu tham khảo .....	13

## 1. Tổng quan đề tài

### 1.1. Lý do chọn đề tài

Trong những năm gần đây, tiền điện tử trở nên khá phổ biến và dần thu hút nhiều người đầu tư sinh lời. Cho nên có nhiều nghiên cứu về dự báo giá của tiền điện tử ra đời. Nhưng hiện nay, đa phần các nghiên cứu này đều chỉ tập trung vào Bitcoin - tiền điện tử có giá trị cao nhất. Rất hiếm những bài nghiên cứu dự báo loại tiền điện tử khác, đặc biệt là Cardano. Bài nghiên cứu này thực hiện dự báo giá của Cardano với mô hình ARIMA bằng ngôn ngữ lập trình Python.

### 1.2. Sơ lược về Cardano

Cardano là một blockchain công khai được sáng lập vào năm 2015 và ra mắt vào năm 2017 bởi Johns Hokinson - người đồng sáng lập Ethereum. Đây là một blockchain có mã nguồn mở hoàn toàn nhằm mục đích cung cấp một cơ sở hạ tầng toàn diện, công bằng và linh hoạt cho các ứng dụng tài chính và xã hội trên quy mô toàn cầu. Mã token của Cardano là ADA.

Cardano được giám sát và phát triển bởi 3 tổ chức: Quỹ Cardano ở Thụy Sĩ, IOHK - công ty về công nghệ và kỹ thuật phần mềm và Emurgo - đối tác công nghệ toàn cầu.

Cardano là một blockchain thế hệ thứ ba sau Bitcoin và Ethereum. Dự án này ra đời với những đặc điểm vượt trội so với những dự án trước đi trước như: khả năng mở rộng, khả năng tương tác, tính bền vững.

Cardano vừa phát hành phiên bản nâng cấp Alonzo vào tháng 9/2021. Với phiên bản nâng cấp này, người dùng có thể sử dụng các hợp đồng thông minh hoặc các ứng dụng phi tập trung DApp trên nền tảng. Bản nâng cấp này cũng đánh dấu việc hoàn thành kỷ nguyên Goguen và bắt đầu bước vào kỷ nguyên mới trong lộ trình phát triển là Basho.

### 1.3. Sơ lược về mô hình ARIMA:

ARIMA (Auto regressive Intergated Moving Average) được biết như mô hình dự đoán cho các dữ liệu chuỗi thời gian trong tài chính. Thành phần của mô hình ARIMA bao gồm:

- AR: Autoregression - Mô hình tự hồi quy: mô hình sử dụng các mối quan hệ phụ thuộc giữa một quan sát và số lượng các quan sát bị trễ. Độ trễ của mô hình phụ thuộc vào tham số p trong ARIMA.
- I: Interagrated - Quá trình tích hợp: Việc sử dụng sự khác nhau giữa các quan sát thô để tạo một chuỗi thời gian có tính dừng. Bậc của sai phân thể hiện bằng tham số d trong ARIMA
- MA (Moving Average): Mô hình trung bình trượt: mô hình sử dụng sự phụ thuộc giữa các quan sát và sai số. Bậc của mô hình là nhân tố q trong ARIMA

Như vậy mô hình ARIMA được ký hiệu là ARIMA(p,d,q)

ARIMA có 2 loại mô hình là ARIMA theo mùa và không theo mùa. Đề tài này chỉ sử dụng mô hình ARIMA không theo mùa để dự đoán giá của Cardano, yếu tố mùa vụ sẽ bị loại bỏ.

## 2. Dữ liệu

### 2.1. Quy trình lấy dữ liệu

Dữ liệu được tải từ trang Yahoo! Finance với mã là "ADA-USD" (giá đồng ADA tính theo USD) được thiết lập lấy từ từ 01/01/2017 đến ngày ngày 21/01/2022.

```
# Lấy dữ liệu ADA từ yahoo finance
df = yf.download('ADA-USD',
                  start='2017-01-01',
```

```
end='2022-01-21',  
progress=False)
```

Dữ liệu tải về được lưu với tên df và có thời gian từ ngày 09/01/2017 đến ngày 20/01/2022. Bộ dữ liệu gồm 6 cột ('Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume') và 1534 hàng tương đương với 1534 ngày lịch sử. Như thông tin bên dưới, không có cột nào bị thiếu dữ liệu nên không thực hiện xử lý giá trị bị thiếu.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
DatetimeIndex: 1534 entries, 2017-11-09 to 2022-01-20  
Data columns (total 6 columns):  
 #   Column      Non-Null Count  Dtype     
 ---  --          --          --         
 0   Open        1534 non-null    float64  
 1   High        1534 non-null    float64  
 2   Low         1534 non-null    float64  
 3   Close        1534 non-null    float64  
 4   Adj Close    1534 non-null    float64  
 5   Volume       1534 non-null    int64  
 dtypes: float64(5), int64(1)  
 memory usage: 83.9 KB
```

Date	Open	High	Low	Close	Adj Close	Volume
2017-11-09	0,02516	0,03506	0,025006	0,032053	0,032053	18716200
2017-11-10	0,032219	0,033348	0,026451	0,027119	0,027119	6766780
2017-11-11	0,026891	0,029659	0,025684	0,027437	0,027437	5532220
2017-11-12	0,02748	0,027952	0,022591	0,023977	0,023977	7280250
2017-11-13	0,024364	0,0263	0,023495	0,025808	0,025808	4419440

## 2.2. Quy trình xử lý dữ liệu

### 2.2.1. Xóa các cột không cần thiết

Vì chỉ sử dụng giá đóng cửa điều chỉnh đến dự đoán với mô hình arima, nên thực hiện xóa tất cả các cột ngoại trừ cột 'Adj Close'.

```
df.drop(df.columns.difference(['Adj Close']), 1, inplace=True)
```

Dữ liệu sau khi xóa các cột không cần dùng đến:

```
Adj Close  
Date  
2017-11-09  0.032053  
2017-11-10  0.027119  
2017-11-11  0.027437  
2017-11-12  0.023977  
2017-11-13  0.025808
```

### 2.2.2. Phân tích dữ liệu thô

Tính trung bình luân phiên và độ lệch chuẩn luân phiên của giá đóng cửa điều chỉnh so với 30 ngày trước đó.

```

rolmean = df.rolling(window= 30,center=False).mean()
rolstd = df.rolling(window=30 ,center=False).std()

```

Vẽ biểu đồ so sánh giữa giá đóng cửa điều chỉnh với giá trị trung bình và độ lệch chuẩn vừa tính

```

orig = plt.plot(df_ada, color='blue', label='Adj Close')
mean = plt.plot(rolmean, color='red', label='Rolling Mean')
std = plt.plot(rolstd, color='black', label = 'Rolling Standard Deviation')
plt.legend(loc='best')
plt.title('Raw Data: Rolling Mean & Standard Deviation')
plt.ylabel("Cardano's Daily Price")
plt.show(block=False)
plt.show()

```

Bên dưới là biểu đồ so sánh giữa giá đóng cửa điều chỉnh với trung bình và độ lệch chuẩn tính bằng 30 ngày trước đó.



Theo như biểu đồ trên, giá của Cardano khá ổn định trong những năm đầu. Đến đầu năm 2021 thì bắt đầu tăng dần và lập đỉnh vào khoảng sau tháng 7/2021 trùng với thời điểm Cardano phát hành phiên bản mới vào tháng 9/2021. Sau khi lập đỉnh, giá bắt đầu giảm dần.

### 2.2.3. Kiểm tra tính mùa vụ của dữ liệu

Việc phân rã sẽ giúp hình dung rõ hơn về dữ liệu, dễ dàng cho quá trình phân tích và dự báo. Dữ liệu được phân tích thành các thành phần như: giá trị trung bình, khuynh hướng, tính mùa vụ, và nhiễu.

Để phân rã dữ liệu, sử dụng gói seasonal\_decompose của statsmodels.

Tiến hành gọi gói seasonal\_decompose

```
from statsmodels.tsa.seasonal import seasonal_decompose
```

Chuyển dữ liệu sang dạng tuần để phân rã dữ liệu rõ ràng hơn

```
# Chuyển dữ liệu sang dạng tuần
df_weekly= df.resample('w').last()
```

```
# Phân rã dữ liệu
#Vẽ biểu đồ phân rã dữ liệu
```

Biểu đồ phân rã dữ liệu



Theo như biểu đồ trend, trong vài năm đầu giá có xu hướng ổn định, thay đổi thành xu hướng tăng dần từ tháng 7/2020. Với biểu đồ 'Seasonal', dữ liệu giá có tính thời vụ lặp lại theo chu kỳ từng năm.

#### 2.2.4. Kiểm tra tính dừng của dữ liệu

Theo Gujarati(2003) một chuỗi thời gian là dừng khi giá trị trung bình, phương sai, hiệp phương sai không đổi tại bất cứ thời điểm nào. Chuỗi không dừng sẽ thay đổi các giá trị này theo thời gian. Theo Ramanathan (2002) hầu hết các chuỗi thời gian trong kinh tế là không dừng vì chúng thường có xu hướng tuyến tính. Một mô hình tốt trong phân tích dữ liệu chuỗi thời gian là khi phân tích trên dữ liệu dừng. Vì vậy cần kiểm định tính dừng của dữ liệu. Để thực hiện điều này có thể sử dụng phương pháp kiểm định Dickey và Fuller mở rộng (ADF) bằng gói adfuller của statsmodels.

Kiểm định giả thuyết:

$H_0: \beta = 0 \rightarrow$  Dữ liệu là chuỗi không dừng

$H_1: \beta < 0 \rightarrow$  Dữ liệu là chuỗi dừng

```
from statsmodels.tsa.stattools import adfuller

def ad_test(dataset):
    dftest = adfuller(dataset, autolag ='AIC')
    print('1. ADF: ', dftest[0])
    print('2. P-value: ', dftest[1])
    print('3. Số các độ lệch : ', dftest[2])
    print('4. Số lượng quan sát được sử dụng để tính toán hồi quy ADF và các giá trị tới hạn: ', dftest[3])
    print('5. Giá trị tới hạn (Critical values: ')
    for key,val in dftest[4].items():
        print('\t',key,':',val)

ad_test(df_ada['Adj Close'])
```

```

1. ADF: -1.4935203840534317
2. P-value: 0.5367142423563943
3. Số các độ lệch : 22
4. Số lượng quan sát được sử dụng để tính toán hồi quy ADF và các giá trị tới hạn: 1511
5. Giá trị tới hạn:
   1% : -3.434685171403452
   5% : -2.863454705264858
   10% : -2.5677893640196907
Dữ liệu không có tính dừng

```

Theo như kết quả cho thấy,  $p\text{-value} = 0.53 > \alpha = 10\%$  hay  $|ADF| = 1.58 <$  trị tuyệt đối giá trị tới hạn ở các mức ý nghĩa 1%, 5%, 10%. Vậy nên chấp nhận giả thuyết  $H_0$ , dữ liệu là chuỗi không dừng. Vậy nên cần phải thực hiện chuyển đổi dữ liệu khắc phục chuỗi không dừng.

### 2.2.5. Khắc phục dữ liệu không dừng

Một cách phổ để khắc phục chuỗi không dừng là lấy sai phân bậc 1 của dữ liệu trong chuỗi. Sử dụng `.diff()` để tính toán. Sau khi lấy sai phân bậc 1, ngày đầu tiên sẽ không có dữ liệu, tiến hành xóa giá trị không có dữ liệu.

```
df_stationary_diff = df_ada.diff().dropna()
```

Bên dưới là dữ liệu sau khi lấy sai phân bậc 1 và xóa giá trị rỗng.

Date	Adj Close
2017-11-10	-0.004934
2017-11-11	0.000318
2017-11-12	-0.003460
2017-11-13	0.001831
2017-11-14	0.000422

Kiểm tra tính dừng của dữ liệu sau khi lấy sai phân bằng kiểm định ADF.

```
ad_test(df_stationary_diff)
```

```

1. ADF: -7.185179983033441
2. P-value: 2.587977257748026e-10
3. Số các độ lệch : 21
4. Số lượng quan sát được sử dụng để tính toán hồi quy ADF và các giá trị tới hạn: 1511
5. Giá trị tới hạn:
   1% : -3.434685171403452
   5% : -2.863454705264858
   10% : -2.5677893640196907
Dữ liệu có tính dừng

```

Theo như kết quả có  $p\text{-value} < \alpha$  và  $|ADF| = 7.18 >$  trị tuyệt đối của các giá trị tới hạn ở các mức ý nghĩa 1%, 5%, 10%, cho nên dữ liệu sau khi lấy sai phân có tính dừng.

Thực hiện vẽ biểu đồ của dữ liệu sau khi lấy sai phân so sánh với trung bình luân phiên và độ lệch chuẩn luân phiên tính bằng 30 ngày trước đó.

```

Vẽ biểu đồ của dữ liệu sau khi lấy sai phân
#Thống kê luân phiên với 30 kỳ
rolmean = df_stationary_diff.rolling(window=30,center=False).mean()
rolstd = df_stationary_diff.rolling(window=30,center=False).std()

# Vẽ biểu đồ
orig = plt.plot(df_stationary_diff, color='blue', label='Diff of adj close')

```

```

mean = plt.plot(rolmean, color='red', label='Rolling Mean')
std = plt.plot(rolstd, color='black', label = 'Rolling Standard Deviation')
plt.legend(loc='best')
plt.title('Diff of price: Rolling mean and standard deviation')
plt.ylabel("Diff of Cardano's Daily price ")
plt.show(block=False)
plt.show()

```

Biểu đồ của dữ liệu sau khi lấy sai phân bậc 1



### 3. Dự báo giá Carnado bằng mô hình ARIMA

#### 3.1. Lựa chọn tham số

AIC (Akaike Information Criteria): Tiêu chí thông tin Akaike là một công cụ ước tính lỗi dự báo và đánh giá chất lượng các mô hình thống kê trên một tập dữ liệu. AIC càng nhỏ thì mô hình càng phù hợp.

Để tìm ra bộ tham số (p,d,q) phù hợp với mô hình nhanh chóng, sử dụng gói auto\_arima của pmdarima. Gói này sẽ lựa chọn bộ tham số có chỉ số AIC thấp nhất.

Để sử dụng auto\_arima, cần cài đặt pmdarima

```
pip install pmdarima
```

Tìm bộ tham số (p, d, q) với auto\_arima loại đi tính mùa vụ của dữ liệu

```
stepwise_fit = auto_arima(df, trace = True, suppress_warnings=True, seasonal =False)
```

```

Performing stepwise search to minimize aic
ARIMA(2,1,2)(0,0,0)[0] intercept : AIC=-4739.236, Time=2.72 sec
ARIMA(0,1,0)(0,0,0)[0] intercept : AIC=-4730.578, Time=0.24 sec
ARIMA(1,1,0)(0,0,0)[0] intercept : AIC=-4737.455, Time=0.44 sec
ARIMA(0,1,1)(0,0,0)[0] intercept : AIC=-4737.203, Time=0.41 sec
ARIMA(0,1,0)(0,0,0)[0] : AIC=-4732.210, Time=0.13 sec
ARIMA(1,1,2)(0,0,0)[0] intercept : AIC=-4741.247, Time=1.89 sec
ARIMA(0,1,2)(0,0,0)[0] intercept : AIC=-4735.384, Time=0.76 sec
ARIMA(1,1,1)(0,0,0)[0] intercept : AIC=-4743.202, Time=1.02 sec
ARIMA(2,1,1)(0,0,0)[0] intercept : AIC=-4733.795, Time=1.13 sec
ARIMA(2,1,0)(0,0,0)[0] intercept : AIC=-4735.955, Time=0.58 sec
ARIMA(1,1,1)(0,0,0)[0] : AIC=-4744.795, Time=0.48 sec
ARIMA(0,1,1)(0,0,0)[0] : AIC=-4738.765, Time=0.10 sec
ARIMA(1,1,0)(0,0,0)[0] : AIC=-4739.023, Time=0.46 sec
ARIMA(2,1,1)(0,0,0)[0] : AIC=-4742.835, Time=0.81 sec
ARIMA(1,1,2)(0,0,0)[0] : AIC=-4742.846, Time=0.54 sec
ARIMA(0,1,2)(0,0,0)[0] : AIC=-4736.960, Time=0.24 sec
ARIMA(2,1,0)(0,0,0)[0] : AIC=-4737.541, Time=0.18 sec
ARIMA(2,1,2)(0,0,0)[0] : AIC=-4740.819, Time=0.81 sec

Best model: ARIMA(1,1,1)(0,0,0)[0]
Total fit time: 13.272 seconds

```

Kết quả mô hình tốt nhất là ARIMA(1,1,1)

```
stepwise_fit.summary()
```

SARIMAX Results						
Dep. Variable:	y	No. Observations:	1534			
Model:	SARIMAX(1, 1, 1)	Log Likelihood	2375.398			
Date:	Tue, 25 Jan 2022	AIC	-4744.795			
Time:	03:04:44	BIC	-4728.790			
Sample:	0	HQIC	-4738.839			
	- 1534					
Covariance Type:	opg					
coef	std err	z	P> z	[0.025	0.975]	
ar.L1	-0.7278	0.050	-14.597	0.000	-0.826	-0.630
ma.L1	0.6563	0.054	12.150	0.000	0.550	0.762
sigma2	0.0026	3.03e-05	86.992	0.000	0.003	0.003
Ljung-Box (L1) (Q):	0.01	Jarque-Bera (JB):	24923.27			
Prob(Q):	0.91	Prob(JB):	0.00			
Heteroskedasticity (H):	8.30	Skew:	0.46			
Prob(H) (two-sided):	0.00	Kurtosis:	22.73			
Warnings:	[1] Covariance matrix calculated using the outer product of gradients (complex-step).					

### 3.2. Dự đoán với mô hình ARIMA

#### 3.2.1. Đào tạo mô hình

Để dễ dàng kiểm định mô hình dự báo, chia dữ liệu thành 2 phần gồm tập dữ liệu đào tạo và kiểm định, 90% dữ liệu sẽ là tập dữ liệu đào tạo, 10% còn lại là dữ liệu kiểm định.

```
n = int(0.9*df.shape[0])
train = df_ada.iloc[:n]
test = df_ada.iloc[n:]
```

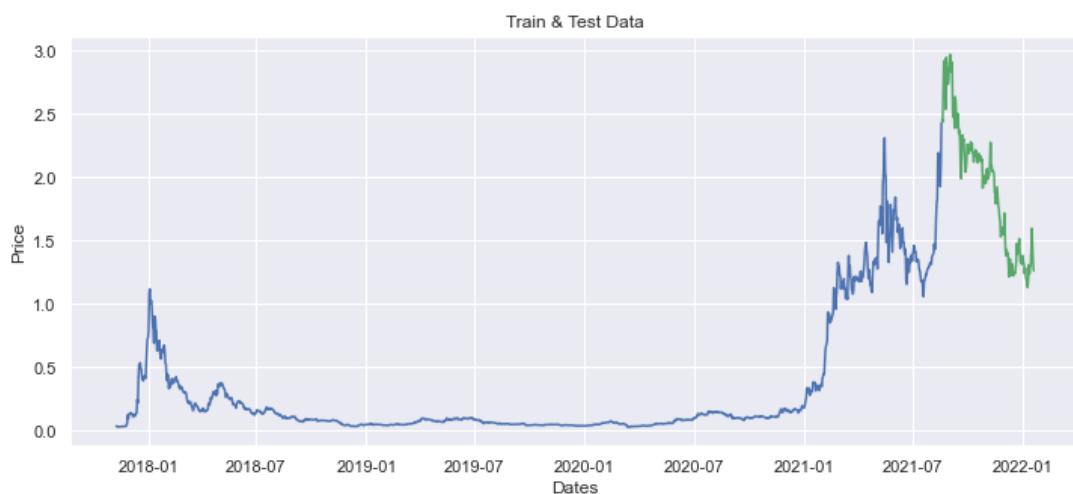
Kiểm tra kích thước của tập dữ liệu đào tạo và thử nghiệm vừa chia

```
print('\n', 'Kích thước của tập dữ liệu đào tạo là: ', train.shape,
      '\n', 'Kích thước của tập dữ liệu thử nghiệm là: ', test.shape)
```

```
Kích thước của tập dữ liệu đào tạo là: (1379, 1)
Kích thước của tập dữ liệu thử nghiệm là: (154, 1)
```

Như vậy thì 1379 ngày đầu tiên là dữ liệu đào tạo, và 154 ngày tiếp theo sẽ là dữ liệu kiểm định.

Bên dưới là biểu đồ trực quan dữ liệu sau khi chia thành dữ liệu đào tạo và kiểm định



Gọi gói ARIMA của statsmodel để thực hiện chạy mô hình arima.

```
from statsmodels.tsa.arima_model import ARIMA
```

Đào tạo mô hình ARIMA với tập dữ liệu “train”

```
model = ARIMA(train, order=(1,1,1))
model = model.fit()
```

Kết quả mô hình ARIMA với dữ liệu đào tạo

```
model.summary()
```

```

ARIMA Model Results
=====
Dep. Variable: D.Adj Close   No. Observations: 1379
Model: ARIMA(1, 1, 1)   Log Likelihood 2380.309
Method: css-mle   S.D. of innovations 0.043
Date: Tue, 25 Jan 2022 AIC -4752.618
Time: 01:57:17   BIC -4731.701
Sample: 11-10-2017 HQIC -4744.793
- 08-19-2021
=====
            coef    std err      z    P>|z|    [0.025    0.975]
-----
const    0.0017    0.001    1.579    0.114    -0.000    0.004
ar.L1.D.Adj Close -0.7328    0.062   -11.740    0.000    -0.855    -0.610
ma.L1.D.Adj Close  0.6376    0.069     9.279    0.000     0.503    0.772
Roots
-----
          Real      Imaginary      Modulus      Frequency
-----
AR.1     -1.3646    +0.0000j    1.3646    0.5000
MA.1     -1.5684    +0.0000j    1.5684    0.5000
-----
"""

```

### 3.2.2. Kiểm định mô hình

Thực hiện dự đoán trên tập kiểm định 'test' bằng .predict() với ngày bắt đầu và kết thúc giống với ngày trong tập kiểm định.

```

start = len(train)
end= len(train) + len(test) -1
pred = model.predict(start=start,end=end, typ='levels')

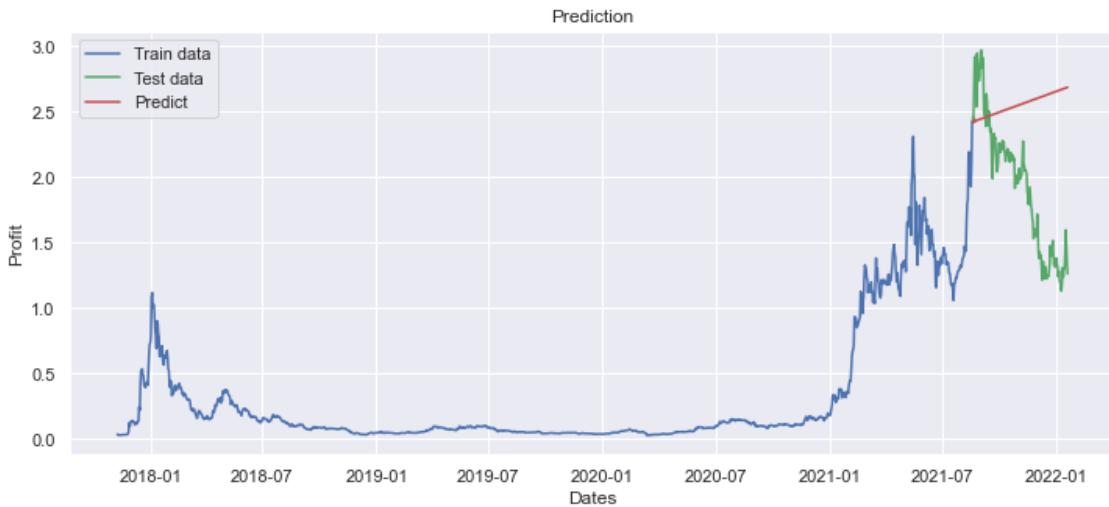
```

Kết quả được dự đoán

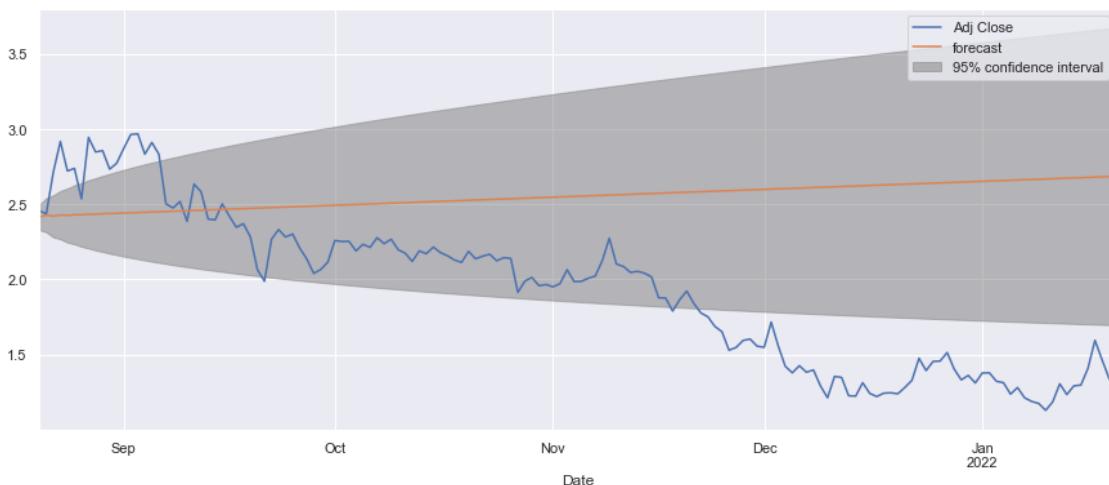
2021-08-21	2.427146
2021-08-22	2.419921
2021-08-23	2.428215
2021-08-24	2.425137
	...
2022-01-16	2.678148
2022-01-17	2.679879
2022-01-18	2.681610
2022-01-19	2.683341
2022-01-20	2.685072

Freq: D, Length: 154, dtype: float64

Biểu đồ giá trị dự đoán với tập dữ liệu kiểm định và dữ liệu đào tạo



Biểu đồ dự đoán giá bằng tập kiểm định với độ tin cậy 95%



Nhìn 2 hình trên có thể thấy được, giá trị dự đoán chênh lệch khá cao so với giá trị kiểm định thực tế. Từ tháng 9 đến gần cuối tháng 11 năm 2021, giá thực tế vẫn nằm trong khoảng tin cậy của dự báo, nhưng sau đó đã vượt ra ngoài khoảng tin cậy. Việc giá vượt ra ngoài khoảng tin cậy một phần do ảnh hưởng bởi sự lao dốc của thị trường tiền điện tử. Nhà đầu tư bắt đầu bán tháo các đồng tiền này sau khi nghe tin giám đốc của các công ty tiền điện tử hàng đầu phải tham gia điều trần tại Hạ viện Mỹ vào tháng 12 năm 2021.

Một mô hình phù hợp với dữ liệu đào tạo chưa chắc là mô hình dự báo tốt. Cho nên cần kiểm tra chất lượng mô hình bằng các sai số dự báo đo lường giữa giá trị thực tế và giá trị dự báo. Các sai số này càng nhỏ thì mô hình được đánh giá tốt. Các sai số dự báo bao gồm:

- MSE (Mean Square Error): Trung bình bình phương sai số.
- RMSE (Root Mean Square Error): Căn bậc 2 của trung bình bình phương sai số.
- MAE (Mean Absolute Error): Sai số trung bình tuyệt đối.
- MAPE (Mean Absolute Percentage Error): Sai số trung bình tuyệt đối tính theo phần trăm.

Để tính toán các sai số dự báo sử dụng các gói `mean_squared_error`, `mean_absolute_error`, `mean_absolute_percentage_error` của `skleans.metrics` và tính toán như bên dưới. Sau đó i ra màn

hình các giá trị của MSE, RMSE, MAE, MAPE lấy 4 chữ số thập phân.

```
from sklearn.metrics import mean_squared_error, mean_absolute_error, mean_absolute_percentage_error
from math import sqrt

mse = mean_squared_error(test,pred)
rmse = sqrt(mse)
mae = mean_absolute_error(test, pred)
mape = mean_absolute_percentage_error(test, pred)

print('Mean Square Error: ' '%.4f' % mse )
print('Root Mean Square Error: ' '%.4f' % rmse )
print('Mean Absolute Error: ' '%.4f' % mae )
print('Mean Absolute Percentage Error: ' '%.4f' % mape )
```

```
Mean Square Error (MSE): 0.7231
Root Mean Square Error (RMSE): 0.8504
Mean Absolute Error (MAE): 0.7097
Mean Absolute Percentage Error (MAPE): 0.4643
```

### 3.2.3. Dự báo giá trong tương lai

Xây mô hình với trên toàn bộ tập dữ liệu và dự báo giá của 30 ngày trong tương lai.

```
#%% Dự đoán giá Cardano trong 30 ngày tiếp theo trong tương lai
model2 = ARIMA(df, order=(1,1,1)).fit()
predict = model2.predict(start = len(df),end=len(df)+30,typ='levels').rename('ARIMA prediction')

print(predict)
```

2022-01-24	1.263655
2022-01-25	1.266068
2022-01-26	1.265698
2022-01-27	1.267353
2022-01-28	1.267535
2022-01-29	1.268788
2022-01-30	1.269262
2022-01-31	1.270303
2022-02-01	1.270931
2022-02-02	1.271860
2022-02-03	1.272570
2022-02-04	1.273439
2022-02-05	1.274192
2022-02-06	1.275030
2022-02-07	1.275806
2022-02-08	1.276627
2022-02-09	1.277415
2022-02-10	1.278228
2022-02-11	1.279022
2022-02-12	1.279830
2022-02-13	1.280628
2022-02-14	1.281433
2022-02-15	1.282233
2022-02-16	1.283037
2022-02-17	1.283838
2022-02-18	1.284641
2022-02-19	1.285442
2022-02-20	1.286245

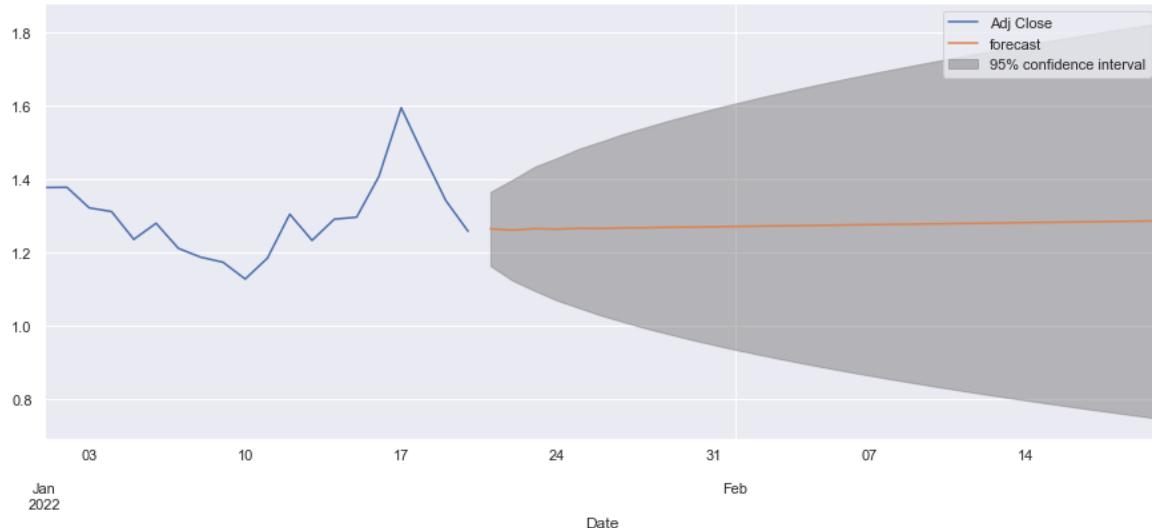
Freq: D, Name: ARIMA prediction, dtype: float64

Vẽ biểu đồ dự đoán giá vào 30 trong tương lai với khoảng tin cậy 95%. Biểu đồ này sẽ dự đoán từ ngày cuối cùng trong tập dữ liệu đến 30 ngày tiếp theo.

```

Biểu đồ dự đoán giá trong tương lai
fig, ax = plt.subplots()
ax = df.loc['2022'].plot(ax=ax)
model.plot_predict(start=len(df), end=len(df)+30, dynamic=True, ax=ax, plot_insample=False)

```



## 4. Ý nghĩa và kết luận

### 4.1. Ý Nghĩa

Với tập kiểm định, các sai số dự báo có giá trị như sau

- RMSE: Biên độ giao động của giá trị dự báo xung quanh giá trị thực tế là 0,8504.
  - MAE: Sai số trung bình tuyệt đối giữa giá trị dự báo và giá trị thực tế là 0,7097.
  - MAPE: Sai số trung bình tuyệt đối giữa giá trị dự báo và giá trị thực tế tính theo phần trăm là 46%.
- Như vậy có thể thấy mô hình dự báo có thể tin cậy được.

Với biểu đồ dự đoán giá của 30 ngày trong tương lai, mô hình dự báo giá của Cardano có khả năng tăng lên, nhưng tăng cực kỳ ít. Ngoài ra trong khoảng tin cậy 95%, giá cũng có thể tăng hoặc giảm; trong 30 ngày giá có thể tăng lên mức cao nhất khoảng 1,8 usd và giảm xuống mức thấp nhất khoảng 0,7 usd.

### 4.2. Kết luận

Kết quả dự báo với tập kiểm định cho thấy giá trị dự báo chênh lệch khá lớn so với giá trị thực nhưng vẫn nằm trong khoảng tin cậy. Điều này cho thấy có thể tin cậy được như độ tin cậy không cao. Điều này là do mô hình chỉ hạn chế trong một biến là lịch sử của giá và cỡ mẫu nhỏ (chưa tới 5 năm), điều này khiến độ chính xác của mô hình không cao.

Để đề tài trở nên hoàn thiện hơn, cần thêm vào các biến khác ảnh hưởng đến giá của Cardano như ảnh hưởng của thông tin và tâm lý nhà đầu tư, ngoài ra cần tăng thêm cỡ mẫu của dữ liệu đào tạo. Những điều này sẽ giúp mô hình dự báo chính xác hơn

## 5. Tài liệu tham khảo

- Nguyễn Anh Phong và cộng sự (2020), Ứng dụng Python trong tài chính, NXB Đại học Quốc gia Tp. Hồ Chí Minh

- Azari, A. (2019). Bitcoin price prediction: An ARIMA approach. *arXiv preprint arXiv:1904.05315*.ISO 690
- <http://nghiencuudinhluong.com/tinh-dung-cua-du-lieu-chuoi-thoi-gian-vacac-kiem-dinh-tinh-dung/>
- <https://phamdinhhanh.github.io/2019/12/12/ARIMAmode.html#31-thu-thap-dữ-liệu>
- <https://coinmarketcap.com/currencies/cardano/>
- <https://cardano.org/>