HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

Statistical Applications To Economics, Modelling Of Economics And Financial Data

GROUP 04

Chu Trung Anh – 20225564
Vu Duc Thang – 20225553
Dao Minh Quang – 20225552
Nguyen Sy Quan - 20225585

ONE LOVE. ONE FUTURE.

Table of content

THREE PARTS

HUST

hust.edu.vn fb.com/dhbkhn

Introduction

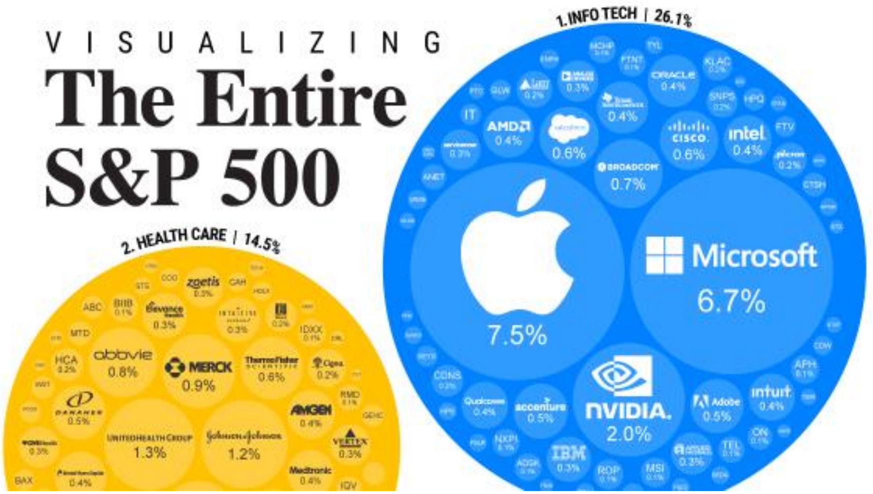
I. Introduction

1. Background

- The S&P 500 index, which includes 500 of the largest publicly traded companies in the U.S.

2. Problem Formulation

- Analyze the stock price movements of Apple Inc. (AAPL) using historical data



I. Introduction

3. Aims

- To explore and preprocess the historical stock price data of Apple Inc.
- Check the stationarity of the time series data and transform it if necessary.
- Decompose the time series to understand its underlying components.
- Build and evaluate predictive models for forecasting future stock prices based on time series analysis.
- Interpret the results and provide actionable insights for investors

HUST

hust.edu.vn fb.com/dhbkhn

Dataset

II. Dataset

1. Dataset Description

- S&P 500 stock data from Kaggle
- Historical stock data for all current S&P 500 companies
- Spans a period of 5 years, from 2013 to 2018
- Contains 7 columns without null value

II. Dataset

1. Dataset Description

	date	open	high	low	close	volume	Name
0	2013-02-08	67.7142	68.4014	66.8928	67.8542	158168416	AAPL
1	2013-02-11	68.0714	69.2771	67.6071	68.5614	129029425	AAPL
2	2013-02-12	68.5014	68.9114	66.8205	66.8428	151829363	AAPL
3	2013-02-13	66.7442	67.6628	66.1742	66.7156	118721995	AAPL
4	2013-02-14	66.3599	67.3771	66.2885	66.6556	88809154	AAPL

1. Dataset Description

```
Data columns (total 7 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   date    1259 non-null   object
 1   open    1259 non-null   float64
 2   high    1259 non-null   float64
 3   low     1259 non-null   float64
 4   close   1259 non-null   float64
 5   volume  1259 non-null   int64
 6   Name    1259 non-null   object
dtypes: float64(4), int64(1), object(2)
memory usage: 69.0+ KB
```

	open	high	low	close	volume
count	1259.000000	1259.000000	1259.000000	1259.000000	1.259000e+03
mean	109.055429	109.951118	108.141589	109.066698	5.404790e+07
std	30.549220	30.686186	30.376224	30.556812	3.346835e+07
min	55.424200	57.085700	55.014200	55.789900	1.147592e+07
25%	84.647800	85.334950	84.250650	84.830650	2.969438e+07
50%	108.970000	110.030000	108.050000	109.010000	4.566893e+07
75%	127.335000	128.100000	126.290000	127.120000	6.870872e+07
max	179.370000	180.100000	178.250000	179.260000	2.668336e+08

2. Preprocessing

a. Set the date column as index and set a fixed frequency

	close
date	
2013-02-08	67.8542
2013-02-11	68.5614
2013-02-12	66.8428
2013-02-13	66.7156
2013-02-14	66.6556

Friday

2. Preprocessing

a. Set the date column as index and set a fixed frequency

	close
date	
2013-02-08	67.8542
2013-02-11	68.5614
2013-02-12	66.8428
2013-02-13	66.7156
2013-02-14	66.6556

Monday

```
# Set the frequency of the DataFrame index
df = df.asfreq('B') # 'B' is the business day frequency
```

II. Dataset

2. Preprocessing

a. Set the date column as index and set a fixed frequency

```
Date: 2013-02-18 00:00:00, Day of Week: Monday
Date: 2013-03-29 00:00:00, Day of Week: Friday
Date: 2013-05-27 00:00:00, Day of Week: Monday
Date: 2013-07-04 00:00:00, Day of Week: Thursday
Date: 2013-09-02 00:00:00, Day of Week: Monday
Date: 2013-11-28 00:00:00, Day of Week: Thursday
Date: 2013-12-25 00:00:00, Day of Week: Wednesday
Date: 2014-01-01 00:00:00, Day of Week: Wednesday
Date: 2014-01-20 00:00:00, Day of Week: Monday
Date: 2014-02-17 00:00:00, Day of Week: Monday
Date: 2014-04-18 00:00:00, Day of Week: Friday
Date: 2014-05-26 00:00:00, Day of Week: Monday
Date: 2014-07-04 00:00:00, Day of Week: Friday
Date: 2014-09-01 00:00:00, Day of Week: Monday
Date: 2014-11-27 00:00:00, Day of Week: Thursday
Date: 2014-12-25 00:00:00, Day of Week: Thursday
Date: 2015-01-01 00:00:00, Day of Week: Thursday
Date: 2015-01-19 00:00:00, Day of Week: Monday
Date: 2015-02-16 00:00:00, Day of Week: Monday
Date: 2015-04-03 00:00:00, Day of Week: Friday
```

2. Preprocessing

b. Normalizing

```
# Scale by the first value of the series
benchmark = df['close'].iloc[0]
df['normalized'] = df['close'].div(benchmark).mul(100)
df.head()
```

date	close	normalized
2013-02-08	67.8542	100.000000
2013-02-11	68.5614	101.042235
2013-02-12	66.8428	98.509451
2013-02-13	66.7156	98.321990
2013-02-14	66.6556	98.233565

2. Preprocessing

c. Stationary check

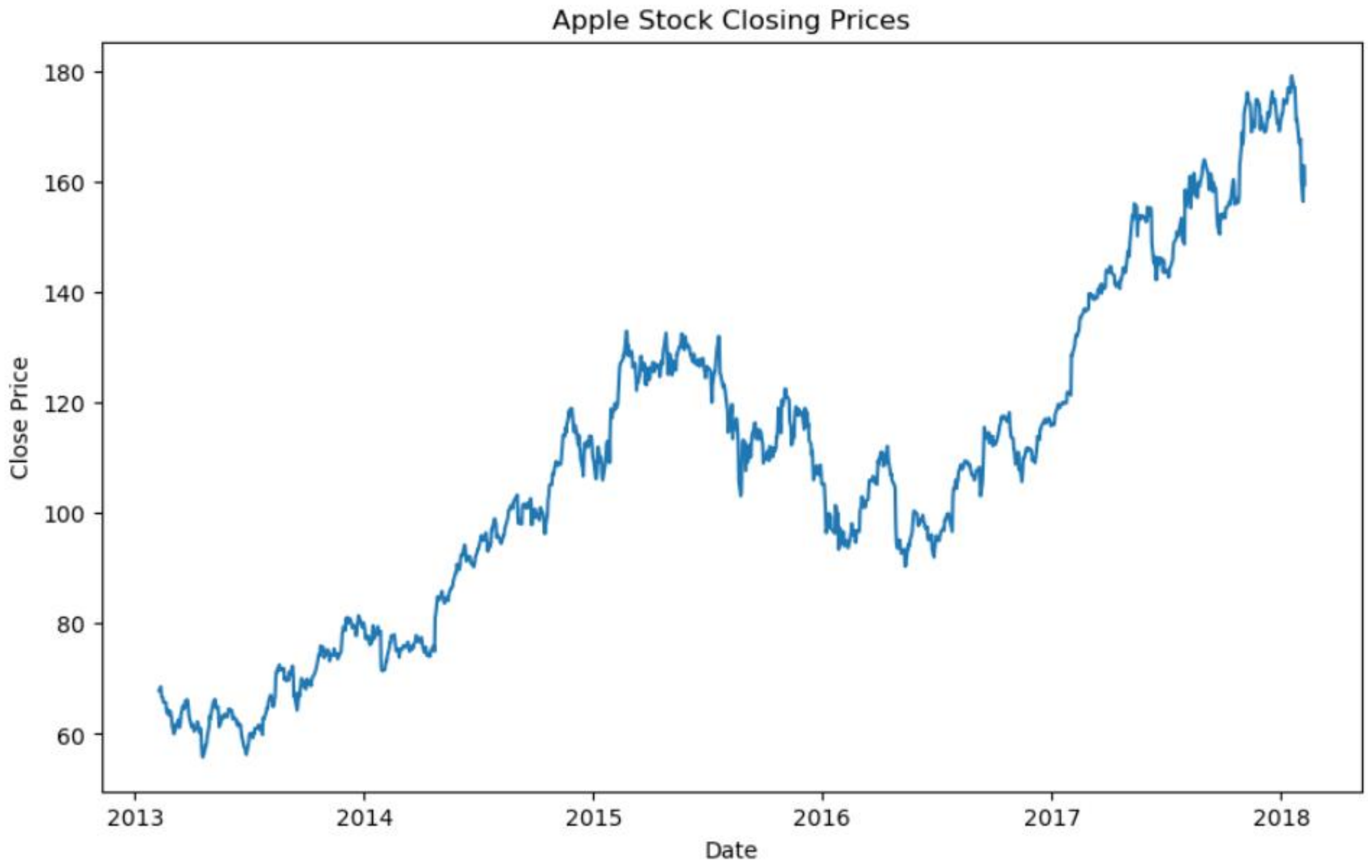
- **The mean (μ)** of the series should be constant over time
- **The variance** of the series should be constant over time
- **The autocorrelation** between values of the series at different times should depend only on the time lag between them, not on their absolute position in time.

1. Visual Inspection
2. Decomposition
3. Dickey-Fuller Test/Augmented Dickey-Fuller (ADF) Test

II. Dataset

2. Preprocessing

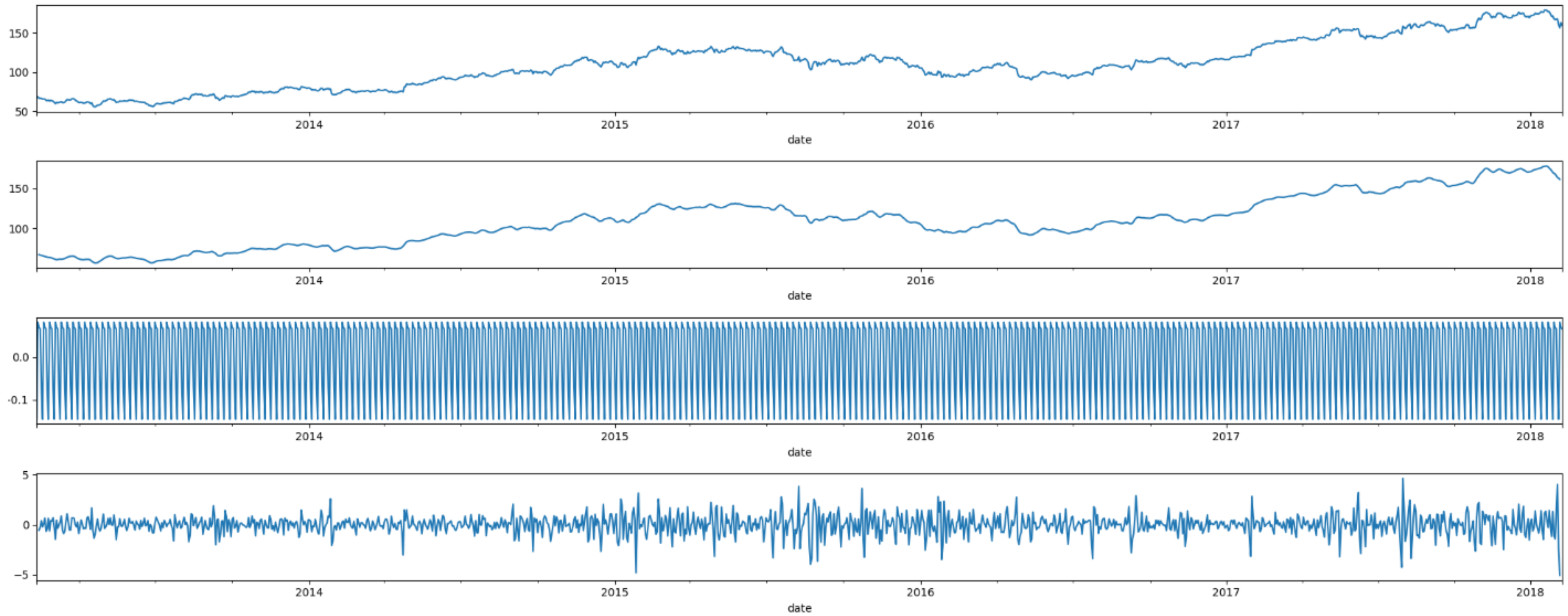
c. Stationary check



II. Dataset

2. Preprocessing

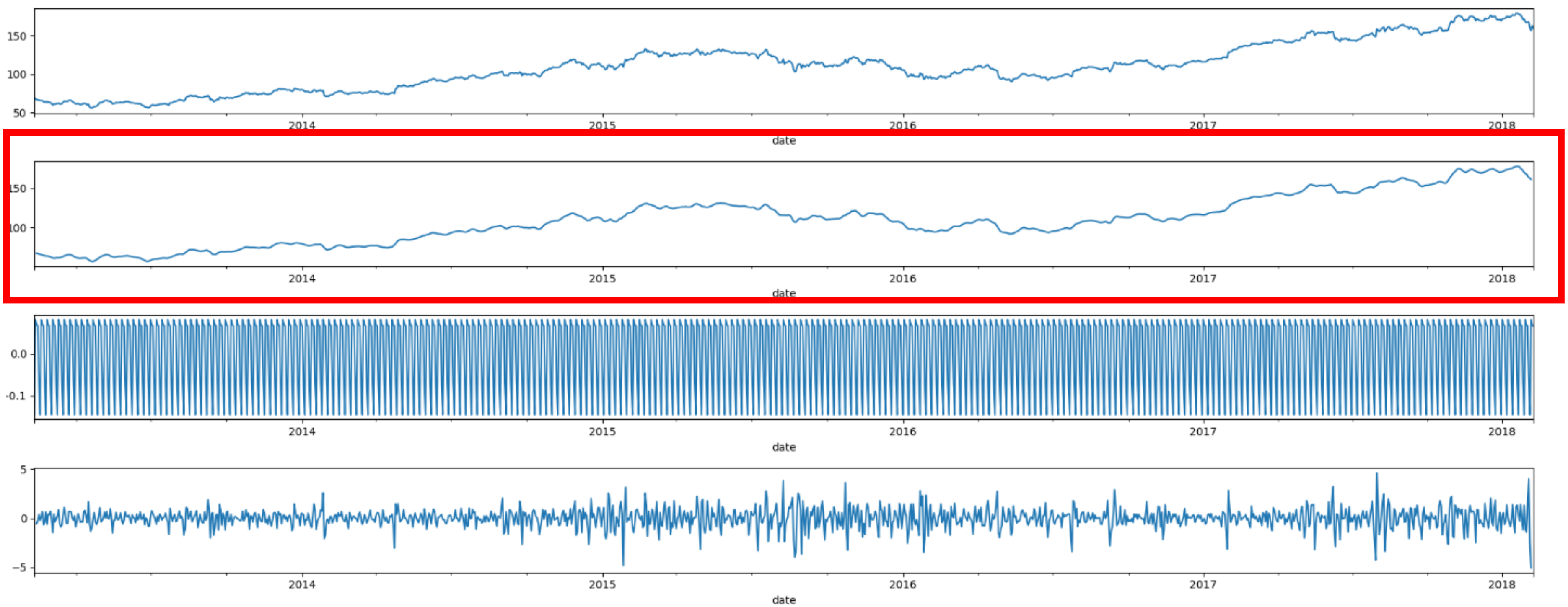
c. Stationary check



II. Dataset

2. Preprocessing

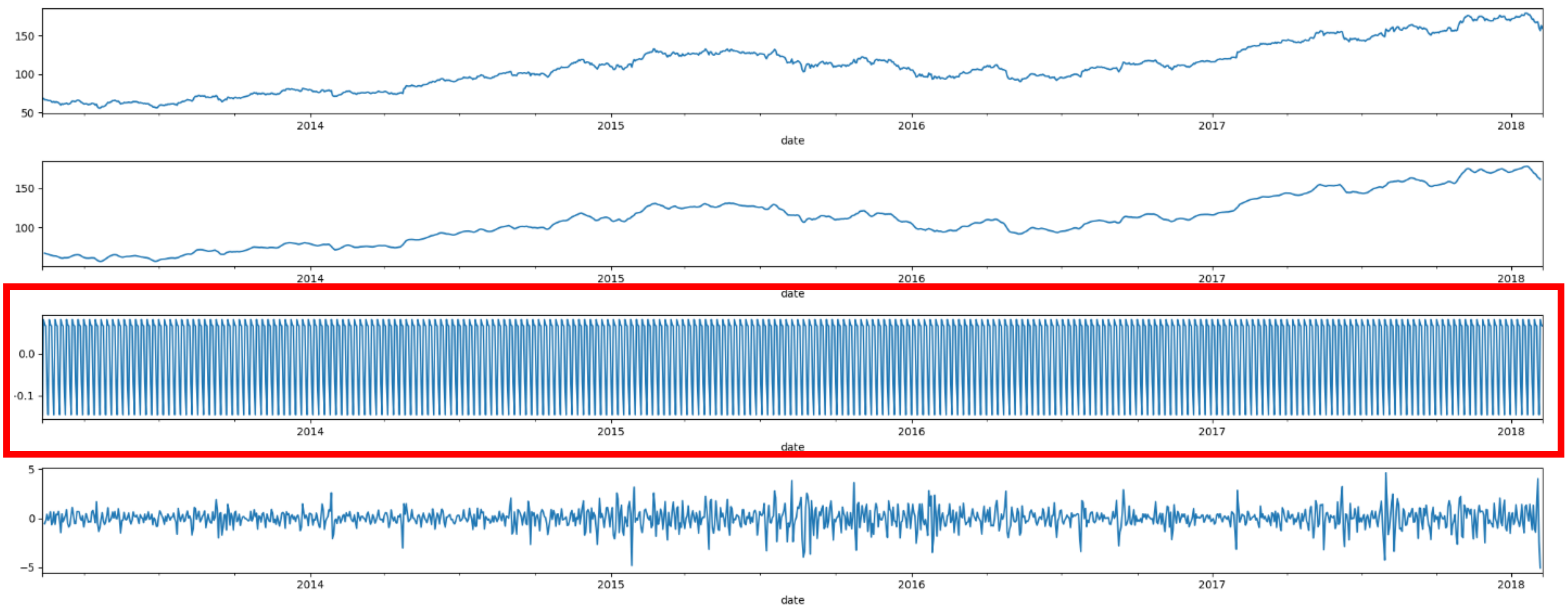
c. Stationary check



Trend

2. Preprocessing

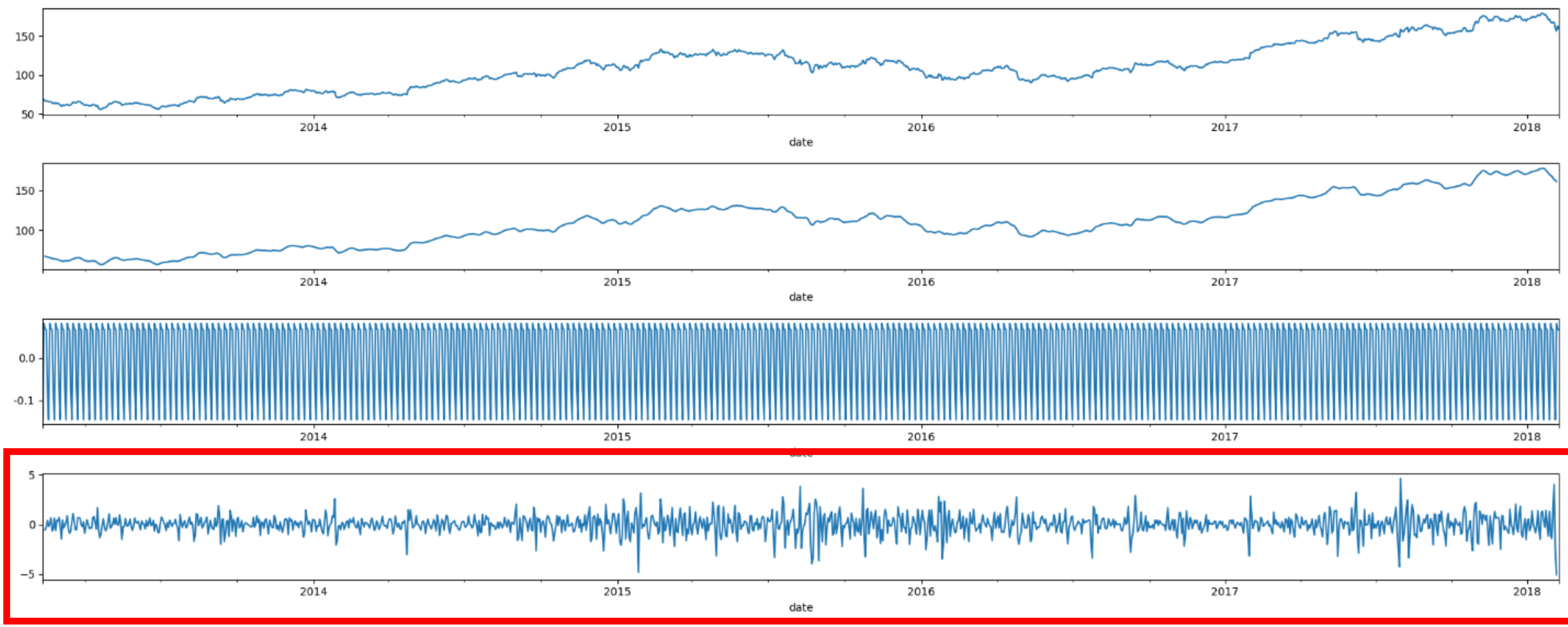
c. Stationary check



Seasonality

2. Preprocessing

c. Stationary check



Residual

2. Preprocessing

c. Stationary check

2. Preprocessing

c. Stationary check

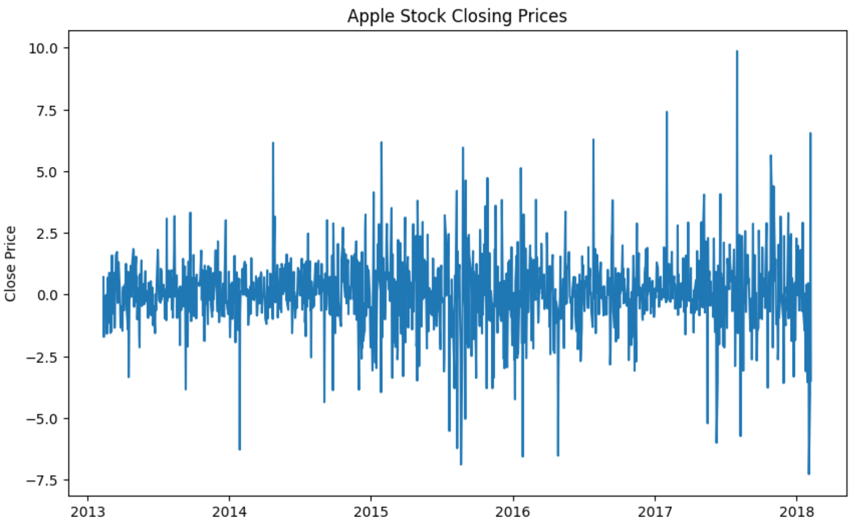
- Null Hypothesis, H0: The time series is not stationary.
- Alternative Hypothesis, H1: The time series is stationary.
- If the p-value is less than or equal to **0.05** or the absolute value of the test statistics is greater than the critical value, we reject H0 and conclude that the time series is stationary.

```
ADF Statistic: -0.660437
p-value: 0.856733
Critical Values:
	1%: -3.435
	5%: -2.864
	10%: -2.568
```

II. Dataset

2. Preprocessing

d. Transform to Stationary



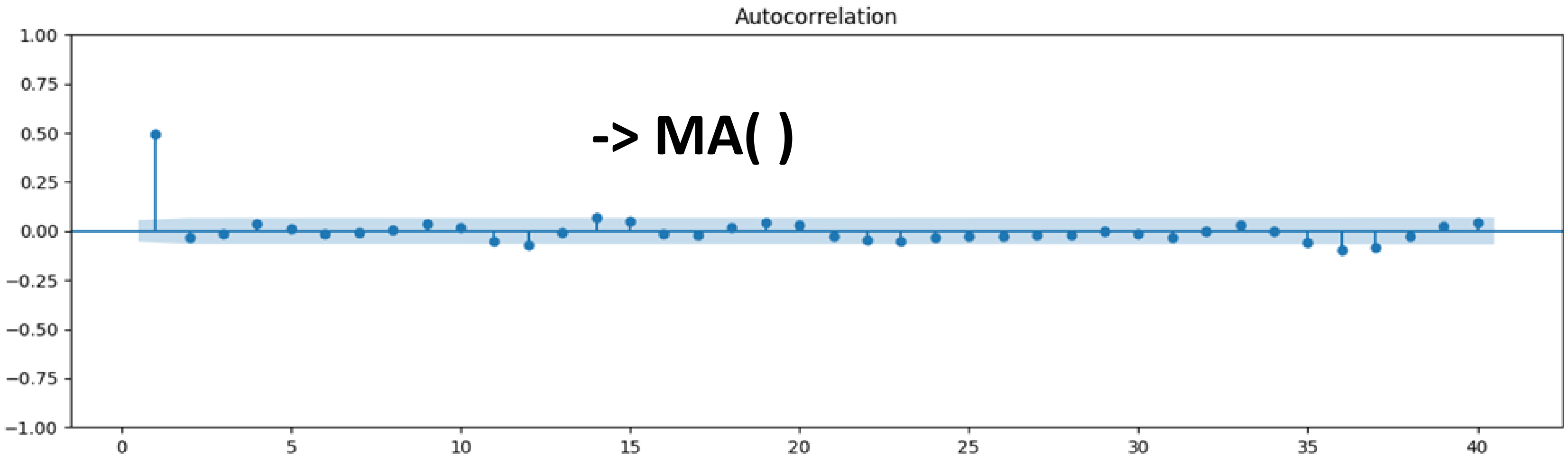

```
ADF Statistic: -7.469759
p-value: 0.000000
Critical Values:
        1%: -3.435
        5%: -2.864
        10%: -2.568
```

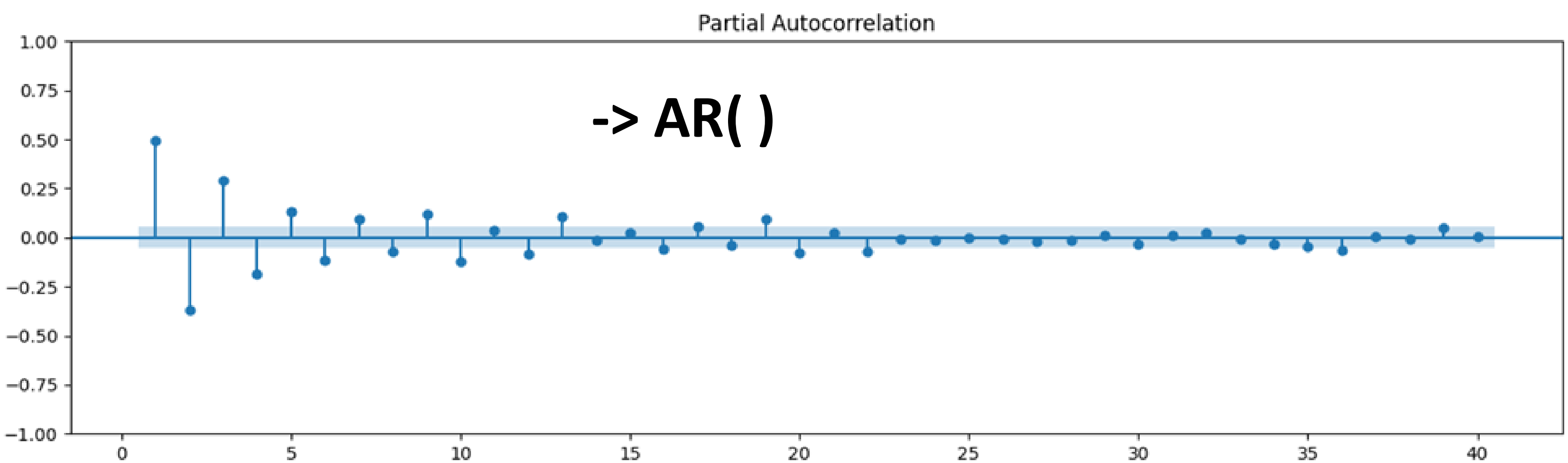
HUST

Models

hust.edu.vn fb.com/dhbkhn

ACF, PACF


Autocorrelation
-> MA( )
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40


Partial Autocorrelation
-> AR( )
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40

AR Model

```
Result for AR(1)
                               SARIMAX Results
==============================================================================
Dep. Variable:                 differ2   No. Observations:                 1040
Model:                  ARIMA(1, 0, 0)   Log Likelihood               -2492.438
Date:                 Wed, 19 Jun 2024   AIC                           4990.877
Time:                         09:19:45   BIC                           5005.718
Sample:                     02-13-2013   HQIC                          4996.507
                          - 02-07-2017
Covariance Type:                   opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.1792      0.168      1.069      0.285      -0.149       0.508
ar.L1          0.5051      0.019     26.260      0.000       0.467       0.543
sigma2         7.0639      0.207     34.047      0.000       6.657       7.471
===================================================================================
Ljung-Box (L1) (Q):                  34.23   Jarque-Bera (JB):               266.22
Prob(Q):                              0.00   Prob(JB):                         0.00
Heteroskedasticity (H):               2.21   Skew:                             0.03
Prob(H) (two-sided):                  0.00   Kurtosis:                         5.48
===================================================================================
```

AR Model

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.1792	0.168	1.069	0.285	-0.149	0.508
ar.L1	0.5051	0.019	26.260	0.000	0.467	0.543
sigma2	7.0639	0.207	34.047	0.000	6.657	7.471

AR Model

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.1792	0.168	1.069	0.285	-0.149	0.508
ar.L1	0.5051	0.019	26.260	0.000	0.467	0.543
sigma2	7.0639	0.207	34.047	0.000	6.657	7.471

$$y_t = c + \theta_1 y_{t-1} + \epsilon_t$$

AR Model

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.1792	0.168	1.069	0.285	-0.149	0.508
ar.L1	0.5051	0.019	26.260	0.000	0.467	0.543
sigma2	7.0639	0.207	34.047	0.000	6.657	7.471

AR Model

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.1792	0.168	1.069	0.285	-0.149	0.508
ar.L1	0.5051	0.019	26.260	0.000	0.467	0.543
sigma2	7.0639	0.207	34.047	0.000	6.657	7.471

- The p-value tests the null hypothesis that the coefficient is equal to zero (no effect).
- A low p-value (< 0.05) indicates that we can reject the null hypothesis. In other words, the coefficient is significant and can be added to the model.

AR Model

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.1792	0.168	1.069	0.285	-0.149	0.508
ar.L1	0.5051	0.019	26.260	0.000	0.467	0.543
sigma2	7.0639	0.207	34.047	0.000	6.657	7.471

- Non – significant p-value for the highest lag coefficients
- Non-significant p-value for the LLR test

AR Model

AR(1)

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.1792	0.168	1.069	0.285	-0.149	0.508
ar.L1	0.5051	0.019	26.260	0.000	0.467	0.543
sigma2	7.0639	0.207	34.047	0.000	6.657	7.471

AR(11)

ar.L9	0.3031	0.046	6.547	0.000
ar.L10	-0.2153	0.038	-5.709	0.000
ar.L11	0.0687	0.029	2.329	0.020

AR(12)

ar.L11	1.508821e-04
ar.L12	3.374042e-03

AR(13)

ar.L12	9.065850e-07
ar.L13	5.288544e-05

AR(14)

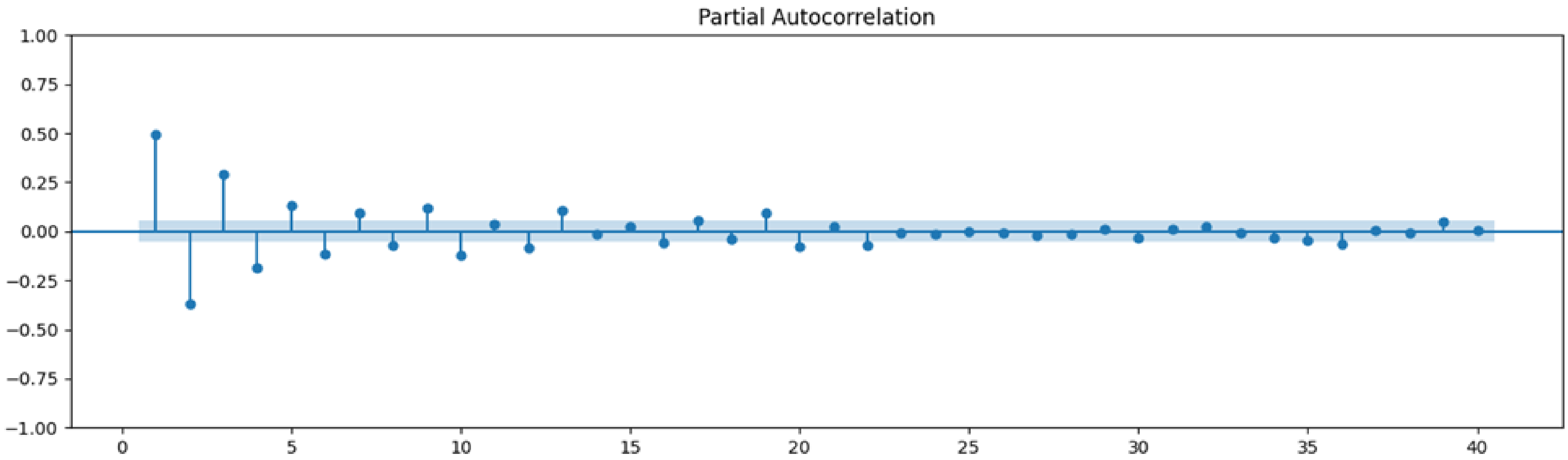
ar.L13	1.569782e-04
ar.L14	2.325376e-01

```
p = LLR_test(result_ar_13, result_ar_14,1)
print('p-value:', p)
```

✓ 0.0s

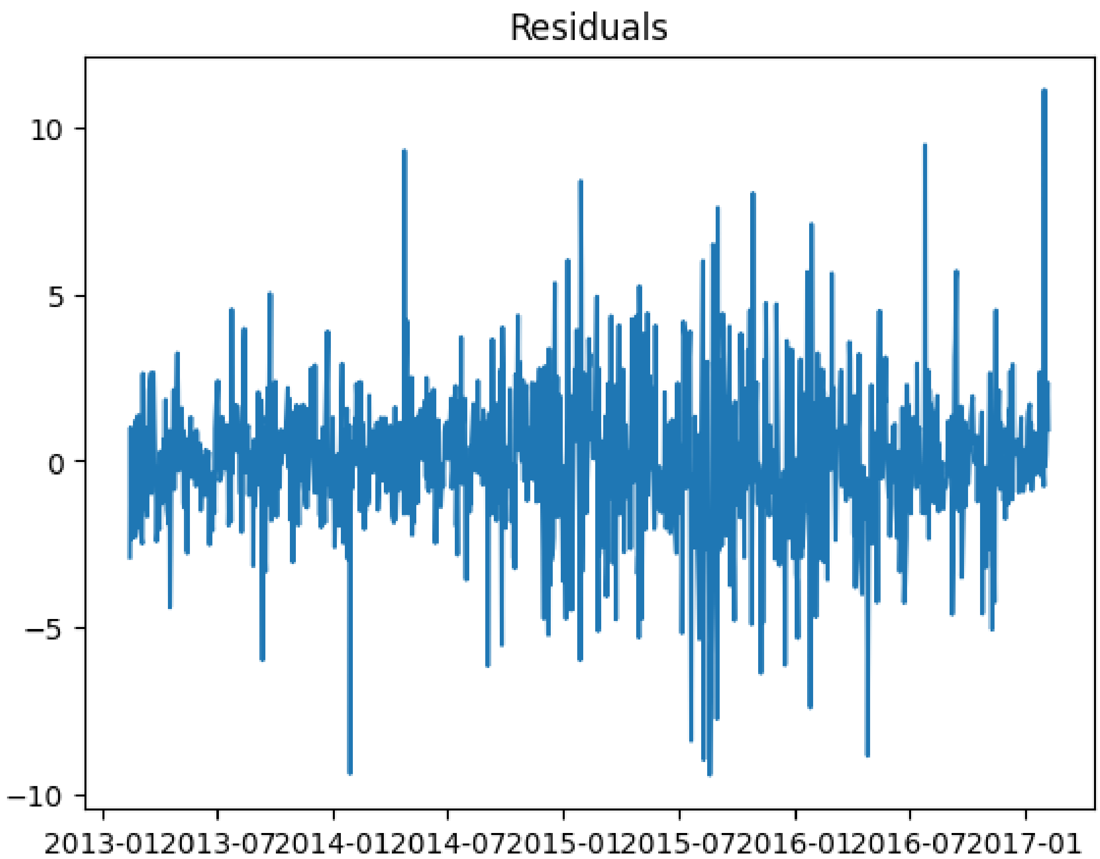
```
p-value: 0.436
```

AR Model



Partial Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40

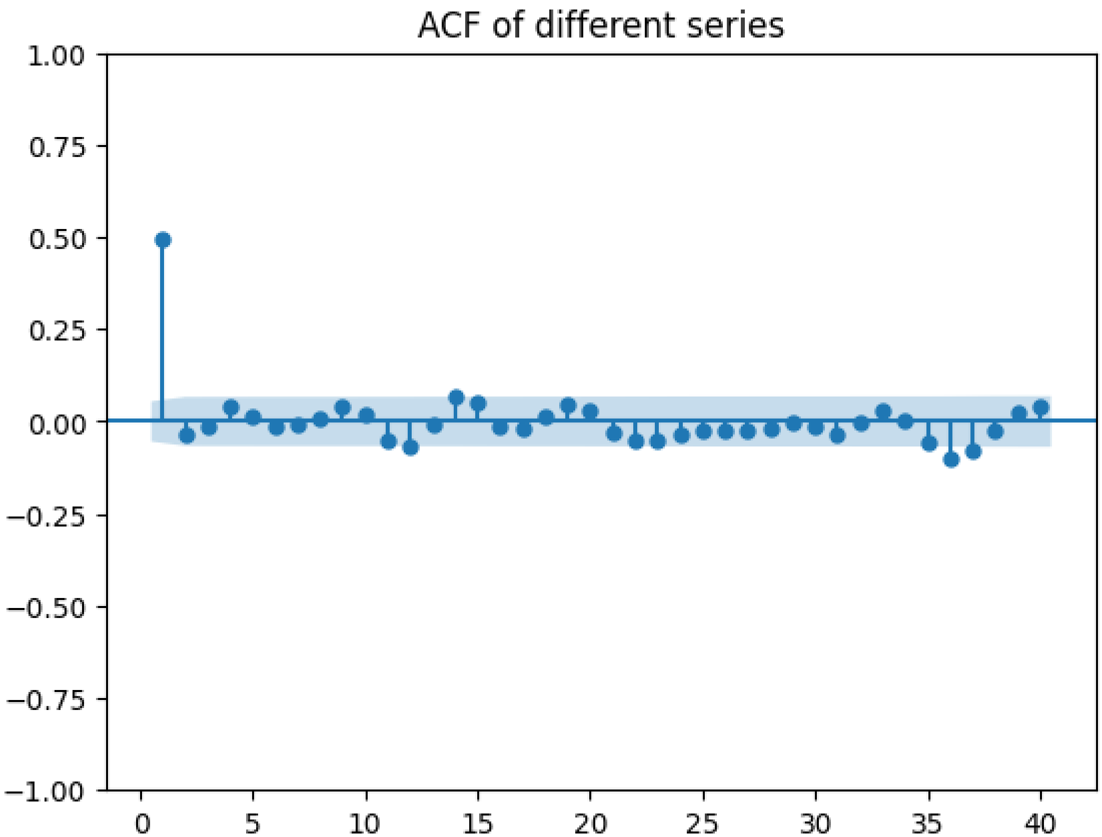
AR Model



	lb_stat	lb_pvalue
10	8.640308	0.566545
20	26.283395	0.156655
30	32.761847	0.332970

The time series is likely white noise

MA Model



ACF of different series
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40

III. Models

MA Model

MA(1)

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.1783	0.134	1.331	0.183	-0.084	0.441
ma.L1	0.9978	0.006	164.488	0.000	0.986	1.010
sigma2	4.6127	0.119	38.735	0.000	4.379	4.846

MA(2)

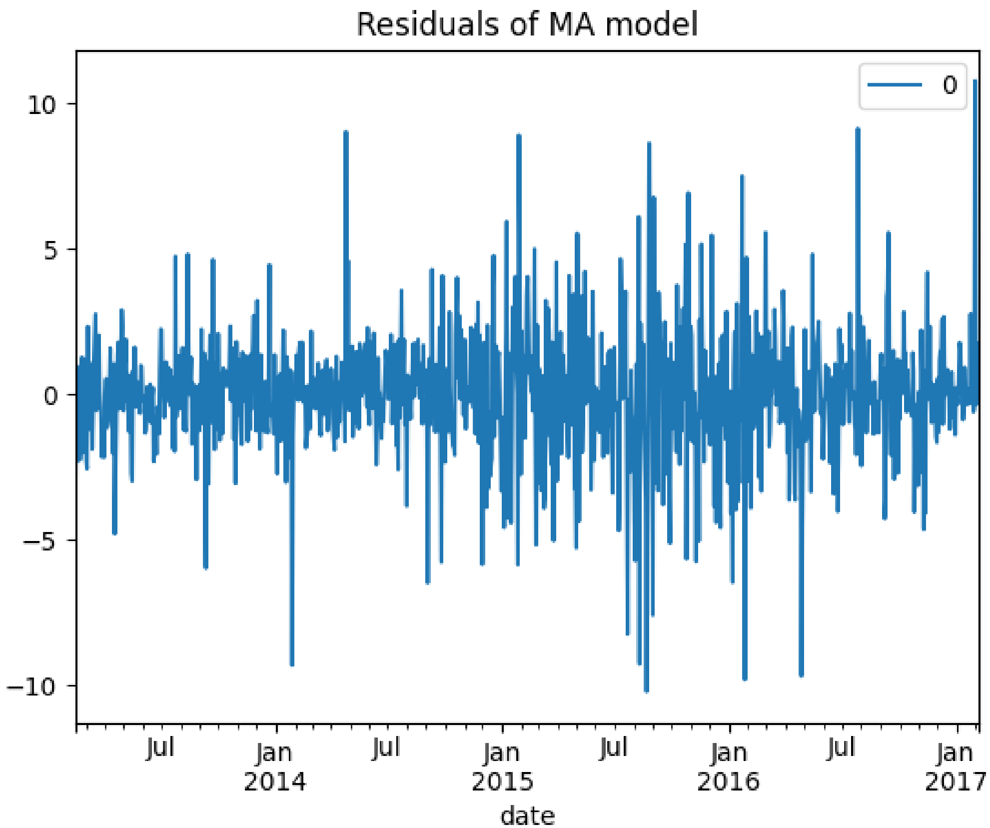
	coef	std err	z	P>|z|	[0.025	0.975]
const	0.1782	0.139	1.284	0.199	-0.094	0.450
ma.L1	1.0292	0.024	42.220	0.000	0.981	1.077
ma.L2	0.0316	0.024	1.296	0.195	-0.016	0.079
sigma2	4.6085	0.119	38.766	0.000	4.376	4.842

```
p = LLR_test(result_ma_1, result_ma_2,1)
print('p-value:', p)
```

✓ 0.0s

```
p-value: 0.481
```

MA Model




```
     lb_stat  lb_pvalue
10   5.280209   0.871691
20  16.975366   0.654573
30  22.775576   0.824372
The time series is likely white noise
```

III. Models

ARIMA (AutoRegressive Integrated Moving Average)

In Auto ARIMA, the model itself will generate the optimal p, d, and q values which would be suitable for the data set to provide better forecasting.

It works similarly like hyper tuning techniques to find the optimal value of p, d, and q with different combinations and the final values would be determined with the lower AIC, BIC parameters taking into consideration

```
model = pm.auto_arima(df, test = 'adf',
                      start_p = 1, start_q = 1,
                      max_p = 3, max_q = 3,
                      d = None, seasonal = True,
                      start_P = 0, m = 3,
                      trace = True, error_action = 'ignore',
                      suppress_warnings = True, stepwise = True,
                      D = 1, information_criterion = 'aic')
```

III. Models

ARIMA

Find the best fit for ARIMA model

```
Arima model for APPLE
Performing stepwise search to minimize aic
 ARIMA(1,0,1)(0,1,1)[3] intercept   : AIC=inf, Time=1.19 sec
 ARIMA(0,0,0)(0,1,0)[3] intercept   : AIC=4777.616, Time=0.02 sec
 ARIMA(1,0,0)(1,1,0)[3] intercept   : AIC=3959.833, Time=0.21 sec
 ARIMA(0,0,1)(0,1,1)[3] intercept   : AIC=4301.457, Time=0.27 sec
 ARIMA(0,0,0)(0,1,0)[3]             : AIC=4780.761, Time=0.02 sec
 ARIMA(1,0,0)(0,1,0)[3] intercept   : AIC=4180.988, Time=0.06 sec
 ARIMA(1,0,0)(2,1,0)[3] intercept   : AIC=3863.582, Time=0.52 sec
 ARIMA(1,0,0)(2,1,1)[3] intercept   : AIC=inf, Time=1.25 sec
 ARIMA(1,0,0)(1,1,1)[3] intercept   : AIC=inf, Time=0.65 sec
 ARIMA(0,0,0)(2,1,0)[3] intercept   : AIC=4780.361, Time=0.19 sec
 ARIMA(2,0,0)(2,1,0)[3] intercept   : AIC=3853.350, Time=0.42 sec
 ARIMA(2,0,0)(1,1,0)[3] intercept   : AIC=3943.424, Time=0.22 sec
 ARIMA(2,0,0)(2,1,1)[3] intercept   : AIC=inf, Time=2.01 sec
 ARIMA(2,0,0)(1,1,1)[3] intercept   : AIC=inf, Time=1.02 sec
 ARIMA(2,0,1)(2,1,0)[3] intercept   : AIC=3849.485, Time=1.23 sec
 ARIMA(2,0,1)(1,1,0)[3] intercept   : AIC=3934.957, Time=0.85 sec
 ARIMA(2,0,1)(2,1,1)[3] intercept   : AIC=inf, Time=3.28 sec
 ARIMA(2,0,1)(1,1,1)[3] intercept   : AIC=inf, Time=1.38 sec
 ARIMA(1,0,1)(2,1,0)[3] intercept   : AIC=3855.484, Time=0.63 sec
 ARIMA(2,0,2)(2,1,0)[3] intercept   : AIC=inf, Time=2.50 sec
 ARIMA(1,0,2)(2,1,0)[3] intercept   : AIC=inf, Time=1.37 sec
 ARIMA(2,0,1)(2,1,0)[3]             : AIC=3849.379, Time=0.37 sec
 ARIMA(2,0,1)(1,1,0)[3]             : AIC=3934.919, Time=0.26 sec
 ARIMA(2,0,1)(2,1,1)[3]             : AIC=inf, Time=3.03 sec
 ARIMA(2,0,1)(1,1,1)[3]             : AIC=inf, Time=1.22 sec
 ARIMA(1,0,1)(2,1,0)[3]             : AIC=3854.837, Time=0.24 sec
 ARIMA(2,0,0)(2,1,0)[3]             : AIC=3852.784, Time=0.24 sec
 ARIMA(2,0,2)(2,1,0)[3]             : AIC=inf, Time=2.45 sec
 ARIMA(1,0,0)(2,1,0)[3]             : AIC=3862.742, Time=0.22 sec
 ARIMA(1,0,2)(2,1,0)[3]             : AIC=inf, Time=1.74 sec

Best model:  ARIMA(2,0,1)(2,1,0)[3]
Total fit time: 29.063 seconds
```

ARIMA (AutoRegressive Integrated Moving Average)

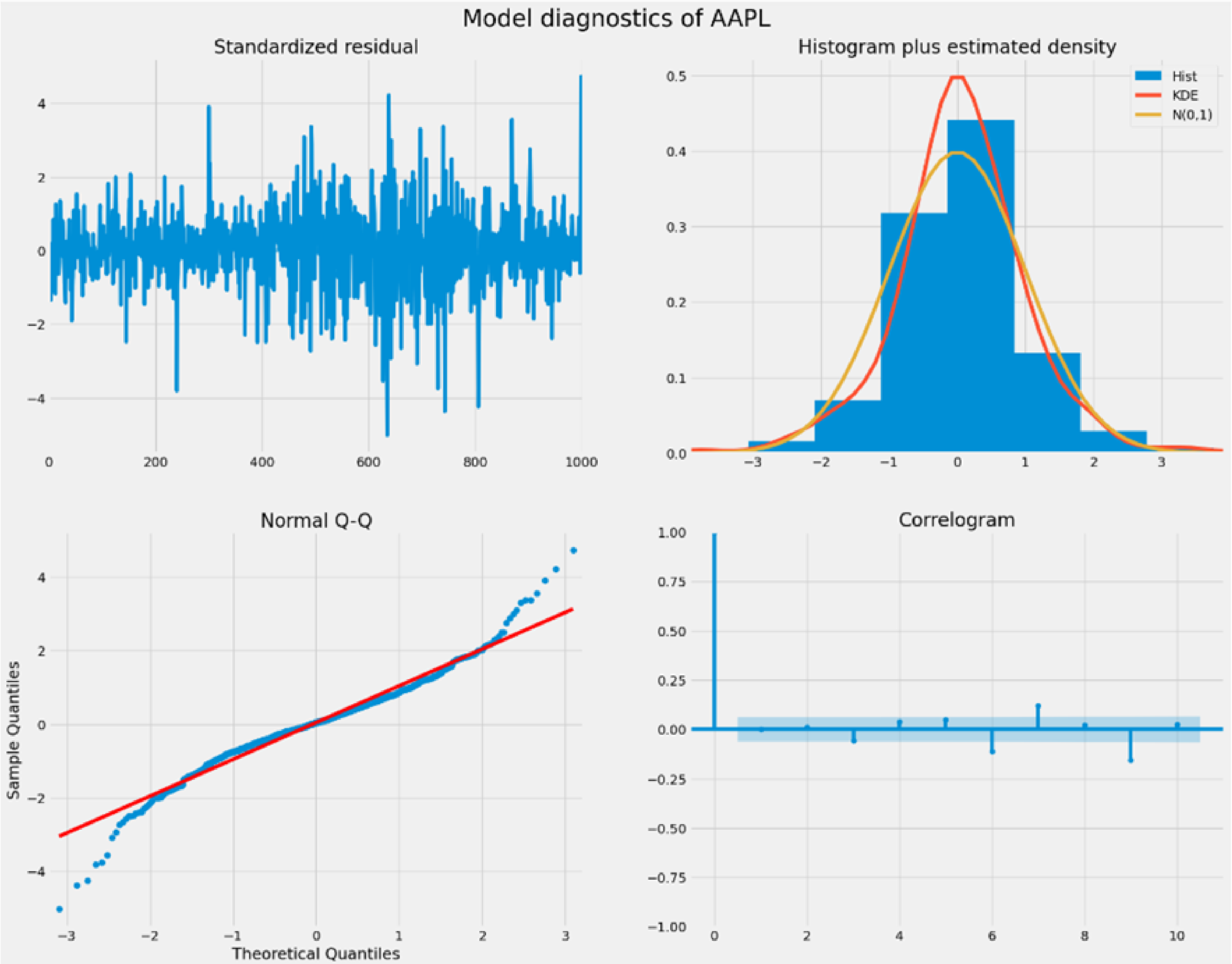
Summary of model

```
                                    SARIMAX Results
==========================================================================================
Dep. Variable:                                   y   No. Observations:                 1007
Model:             SARIMAX(2, 0, 1)x(2, 1, [], 3)   Log Likelihood               -1918.689
Date:                             Tue, 18 Jun 2024   AIC                           3849.379
Time:                                     16:54:13   BIC                           3878.849
Sample:                                          0   HQIC                          3860.577
                                            - 1007
Covariance Type:                               opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          1.5449      0.133     11.630      0.000       1.285       1.805
ar.L2         -0.6180      0.111     -5.560      0.000      -0.836      -0.400
ma.L1         -0.6089      0.139     -4.394      0.000      -0.880      -0.337
ar.S.L3       -0.6548      0.029    -22.594      0.000      -0.712      -0.598
ar.S.L6       -0.3174      0.026    -12.160      0.000      -0.369      -0.266
sigma2         2.6703      0.079     33.806      0.000       2.515       2.825
===================================================================================
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):               360.62
Prob(Q):                              1.00   Prob(JB):                         0.00
Heteroskedasticity (H):               2.04   Skew:                            -0.15
Prob(H) (two-sided):                  0.00   Kurtosis:                         5.92
===================================================================================
```

III. Models

ARIMA

Model diagnostics interpretation



PROPHET

Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.

A forecast is made by calling the predict() function and passing a DataFrame that contains one column named ‘ds‘ and rows with date-times for all the intervals to be predicted.

PROPHET

PROPHET

The result of the predict() function in prophet model is a DataFrame that contains many columns. Perhaps the most important columns are the forecast date time (‘ds‘), the forecasted value (‘yhat‘), and the lower and upper bounds on the predicted value (‘yhat_lower‘ and ‘yhat_upper‘) that provide uncertainty of the forecast.

Few predictions

	ds	yhat	yhat_lower	yhat_upper
1619	2019-02-03	224.556478	196.755399	252.914918
1620	2019-02-04	221.250834	192.218048	249.701917
1621	2019-02-05	221.769962	192.332048	249.999502
1622	2019-02-06	222.222453	194.043353	249.532111
1623	2019-02-07	222.724806	194.346758	250.581311

III. Models

PROPHET

Results

HUST

hust.edu.vn fb.com/dhbkhn

THANK YOU !