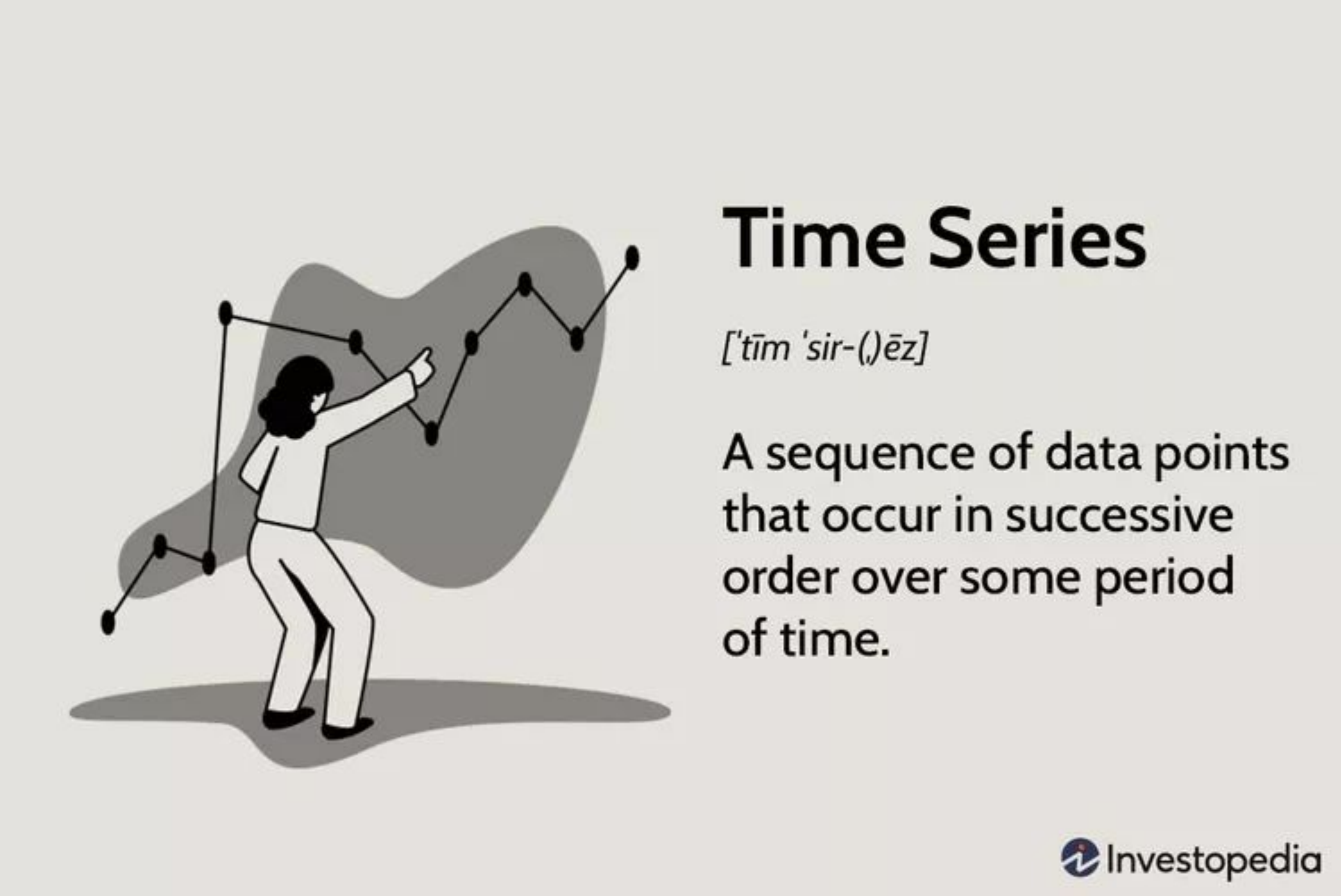
SC310005 Artificial Intelligence

Lecture 12: Time Series

teerapong.pa@chula.ac.th

Reference:

1. https://medium.com/@kasperjuunge/yfinance-10-ways-to-get-stock-data-with-python-6677f49e8282
2. https://www.accaglobal.com/gb/en/student/exam-support-resources/fundamentals-exams-study-resources/f5/technical-articles/time-series.html
3. https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-to-time-series-analysis/
4. https://towardsdatascience.com/stationarity-in-time-series-analysis-90c94f27322
5. https://wellntel.com/nonstationarity-the-importance-of-hydrologic-observations-and-data-to-water-management/
6. https://www.linkedin.com/pulse/power-time-series-analysis-stock-prediction-heri-kaniugu

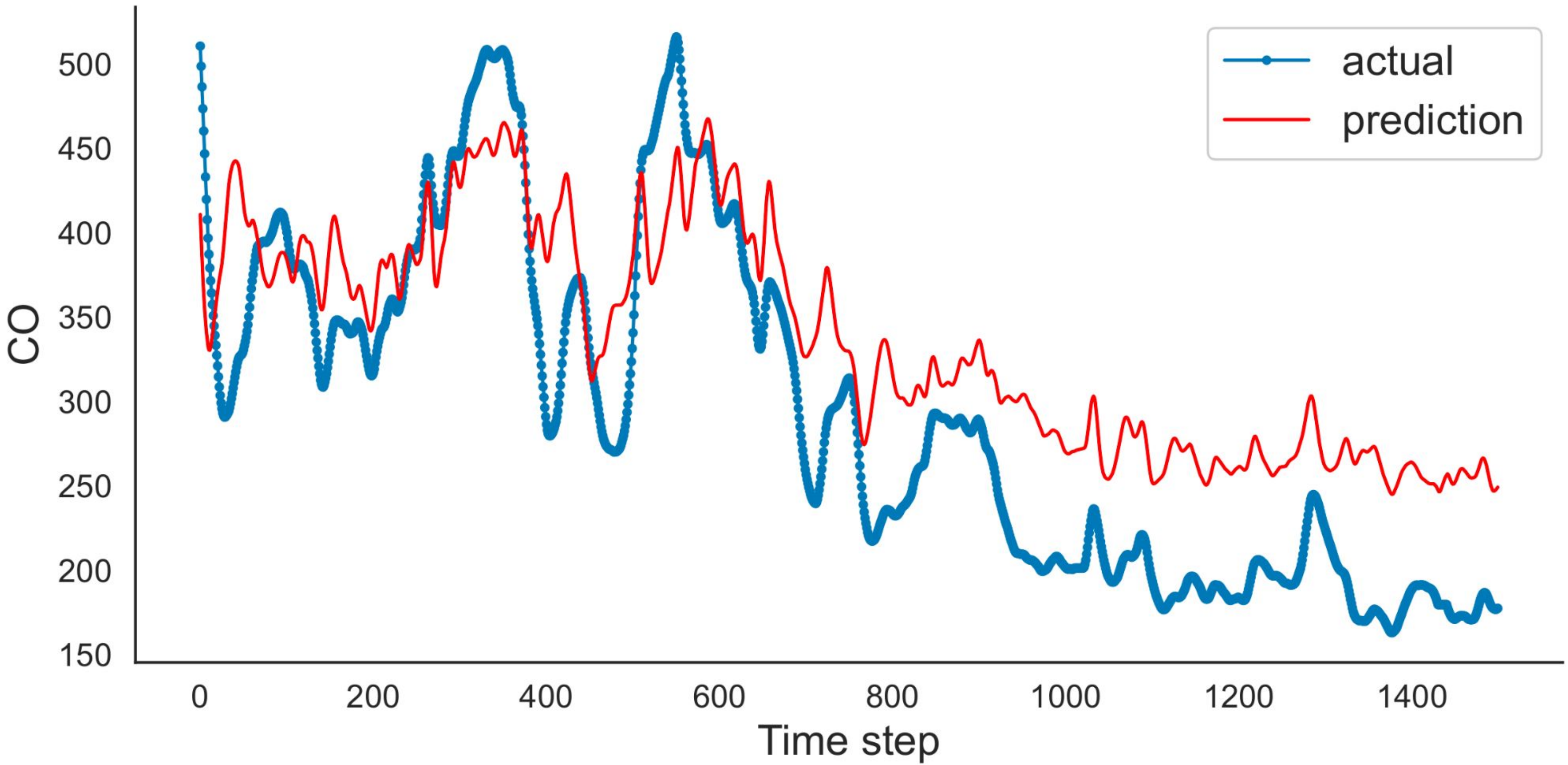

Time Series
['tīm 'sir-(,)ēz]
A sequence of data points that occur in successive order over some period of time.
Investopedia

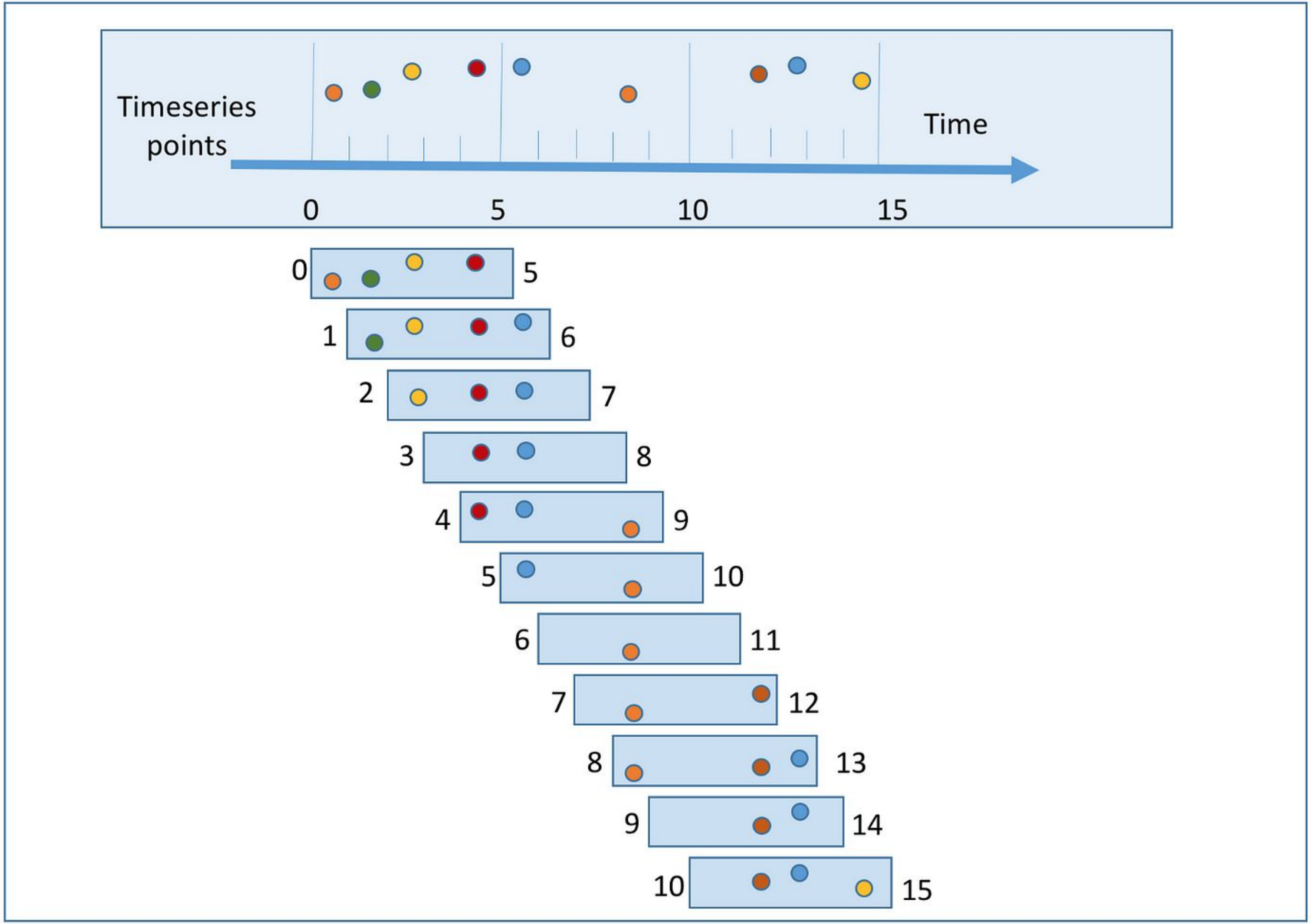
What Is Time Series Analysis?

Time Series Analysis (TSA) is a way of studying the characteristics of the response variable concerning time as the independent variable.

To estimate the target variable in predicting or forecasting, use the time variable as the reference point.

TSA represents a series of time-based orders, it would be Years, Months, Weeks, Days, Horus, Minutes, and Seconds.


actual
prediction
CO
Time step
500
450
400
350
300
250
200
150
0
200
400
600
800
1000
1200
1400



Timeseries points
Time
0
5
10
15
0
5
1
6
2
7
3
8
4
9
5
10
6
11
7
12
8
13
9
14
10
15

How to Analyze Time Series?

To perform the time series analysis, we have to follow the following steps:

- Collecting the data and cleaning it
- Preparing Visualization with respect to time vs key feature
- Observing the **stationarity** of the series
- Developing charts to understand its nature.
- Model building – AR, MA, ARMA and ARIMA
- Extracting insights from prediction

Significance of Time Series

TSA is the backbone for prediction and forecasting analysis, specific to time-based problem statements.

- Analyzing the historical dataset and its patterns
- Understanding and matching the current situation with patterns derived from the previous stage.
- Understanding the factor or factors influencing certain variable(s) in different periods.

Components of Time Series Analysis

Let’s look at the various components of Time Series Analysis:

- **Trend:** In which there is no fixed interval and any divergence within the given dataset is a continuous timeline. The trend would be Negative or Positive or Null Trend
- **Seasonality:** In which regular or fixed interval shifts within the dataset in a continuous timeline. Would be bell curve or saw tooth
- **Cyclical:** In which there is no fixed interval, uncertainty in movement and its pattern
- **Irregularity:** Unexpected situations/events/scenarios and spikes in a short time span.

What Are the Limitations of Time Series Analysis?

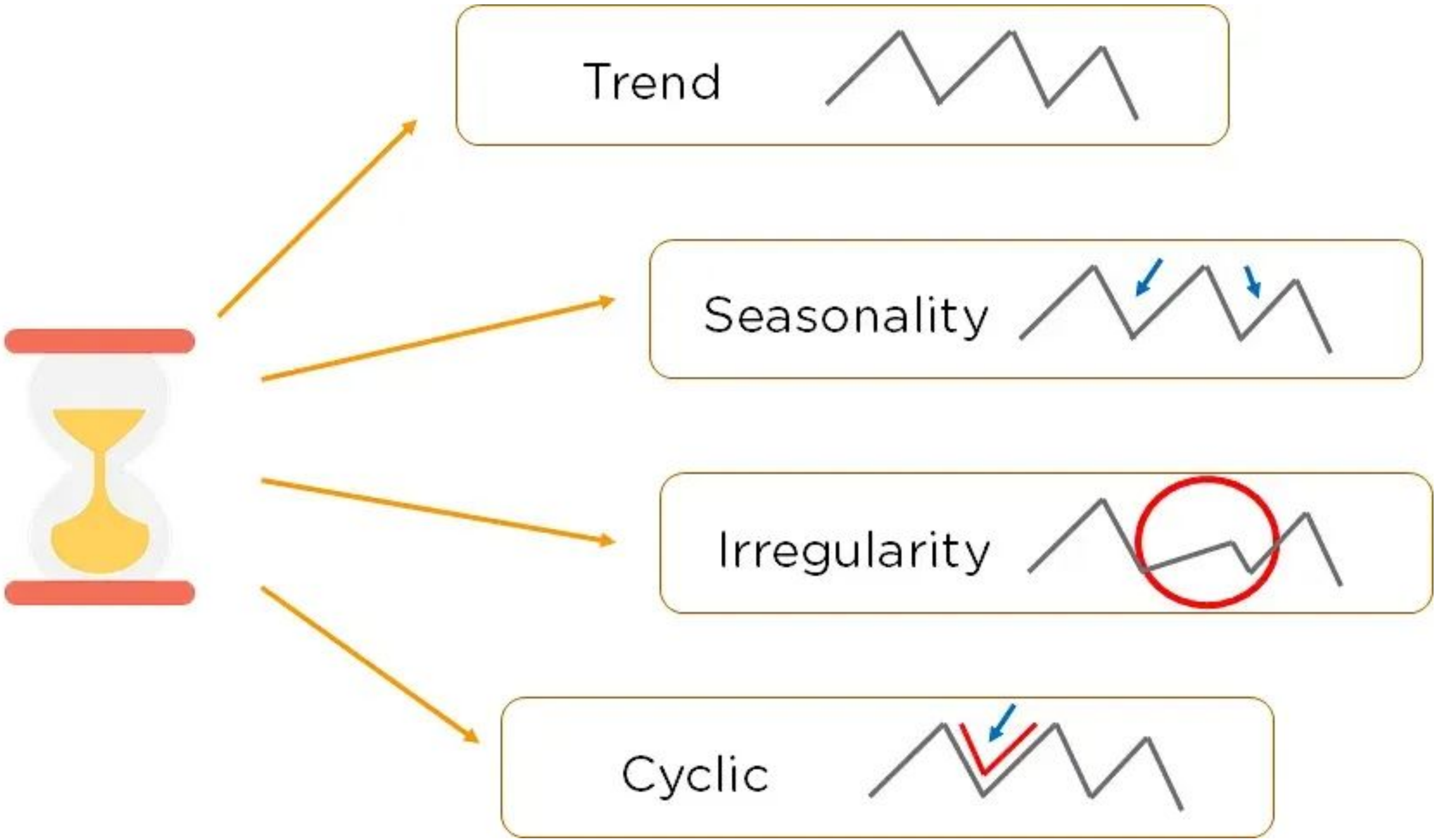
Time series has the below-mentioned limitations; we have to take care of those during our data analysis.

- Similar to other models, the missing values are not supported by TSA
- The data points must be linear in their relationship.
- Data transformations are mandatory, so they are a little expensive.
- Models mostly work on Uni-variate data.

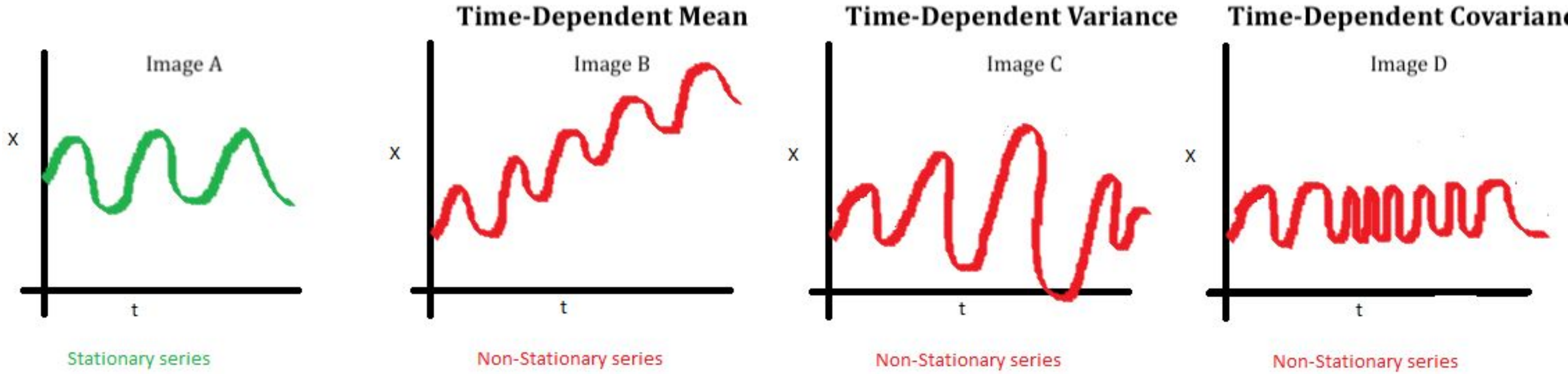
Data Types of Time Series

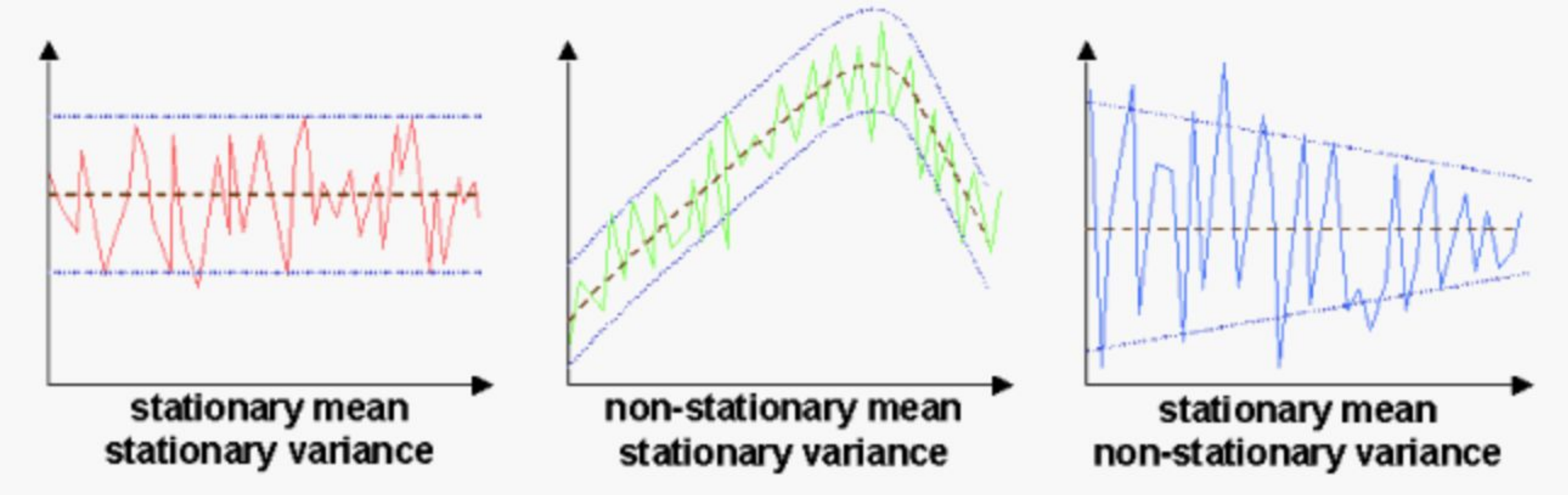
Let's discuss the time series' data types and their influence. While discussing TS data types, there are two major types – stationary and non-stationary.

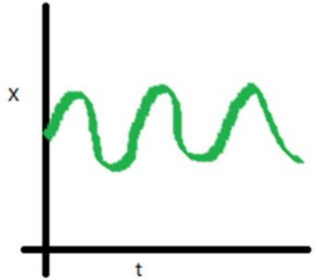
- **Stationary:** A dataset should follow the below thumb rules without having Trend, Seasonality, Cyclical, and Irregularity components of the time series.
  - The mean value of them should be completely constant in the data during the analysis.
  - The variance should be constant with respect to the time-frame
  - Covariance measures the relationship between two variables.
- **Non- Stationary:** If either the mean-variance or covariance is changing with respect to time, the dataset is called non-stationary.


Trend
Seasonality
Irregularity
Cyclic

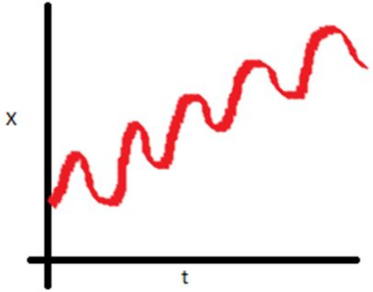
The Principles of Stationarity



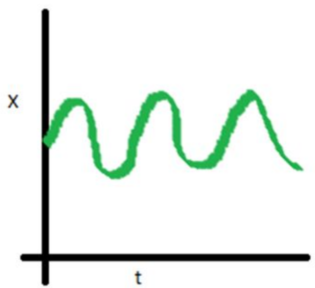

stationary mean
stationary variance
non-stationary mean
stationary variance
stationary mean
non-stationary variance



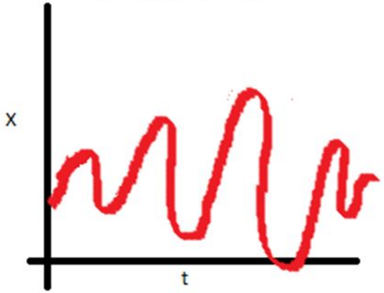
Stationary series



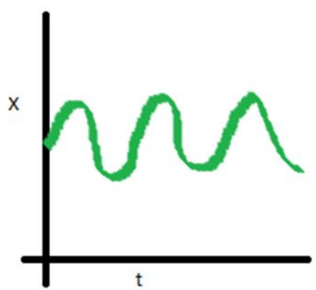
Non-Stationary series



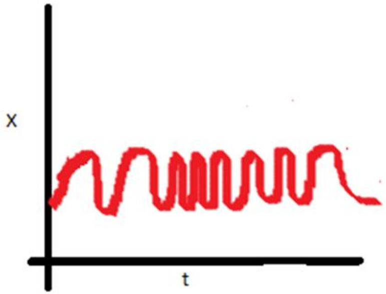
Stationary series



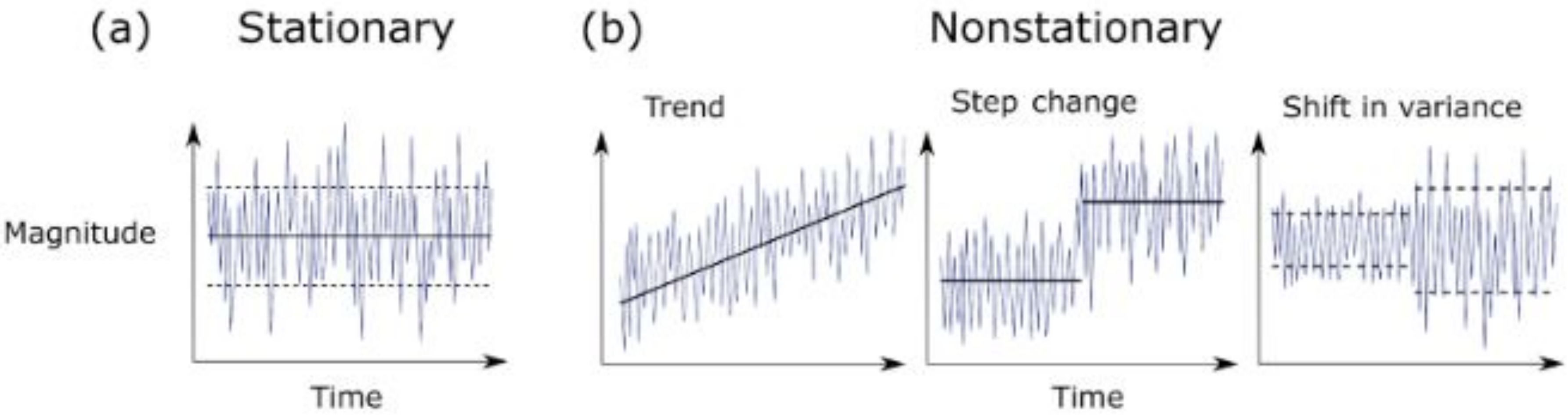
Non-Stationary series

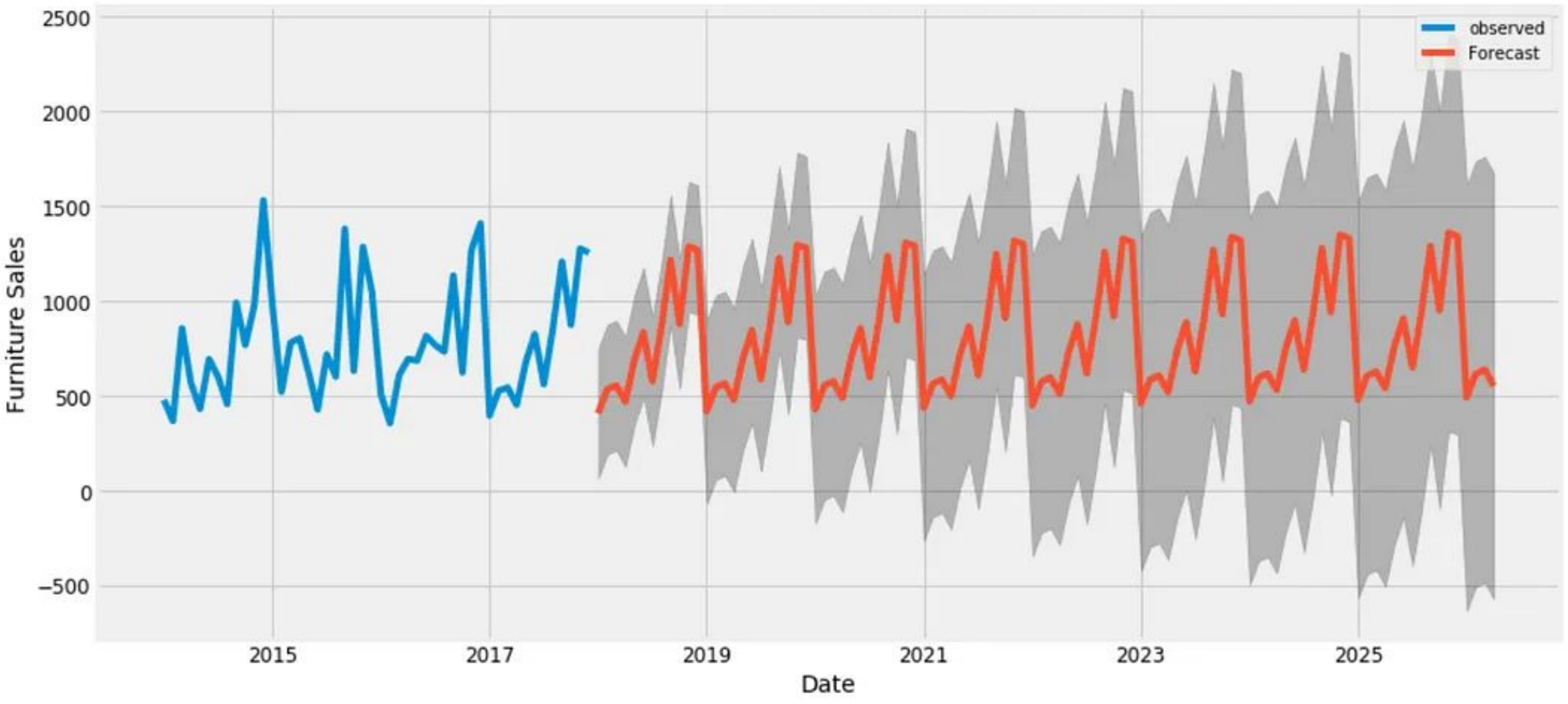


Stationary series

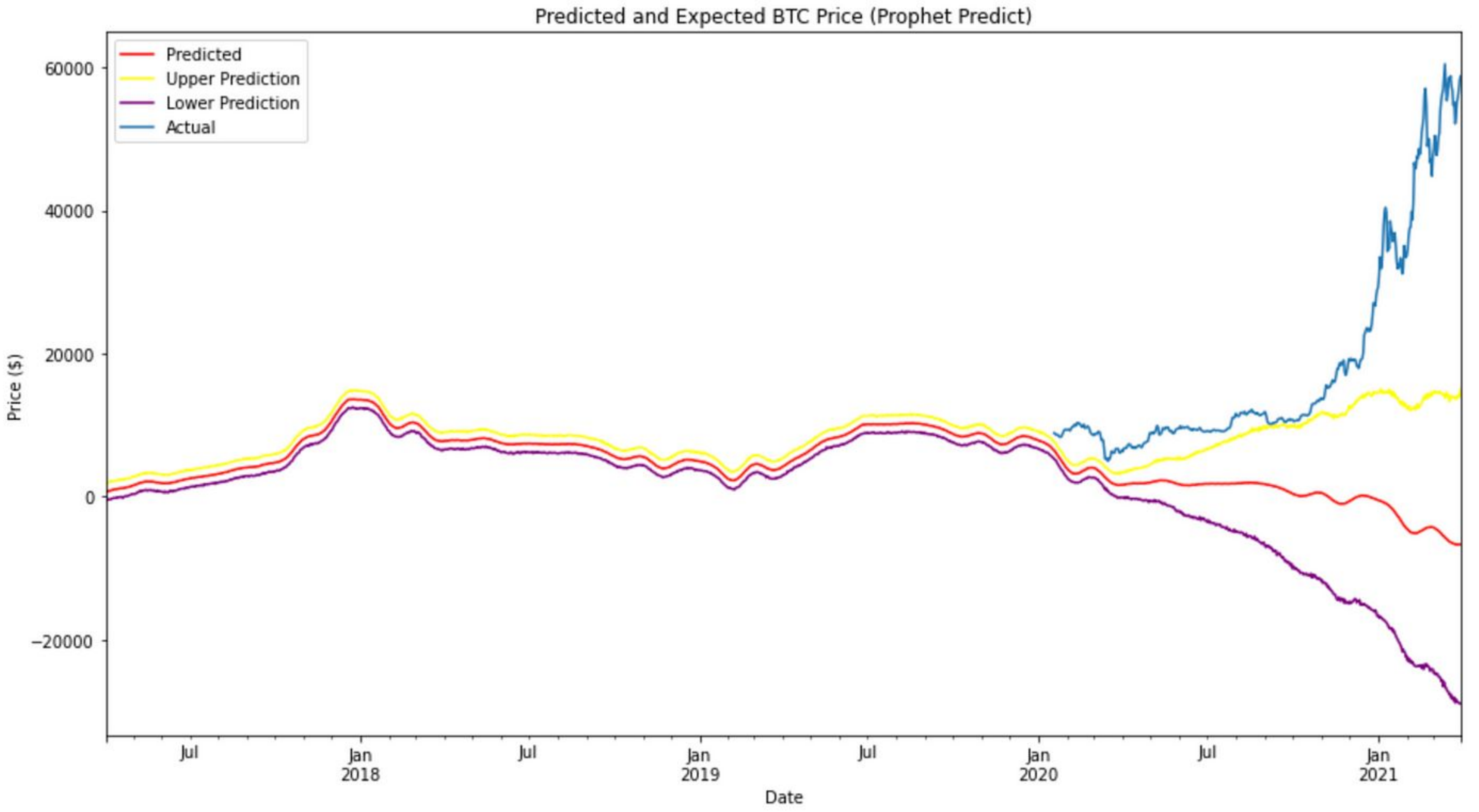


Non-Stationary series


(a)
Stationary
(b)
Nonstationary
Magnitude
Time
Trend
Step change
Shift in variance
Time

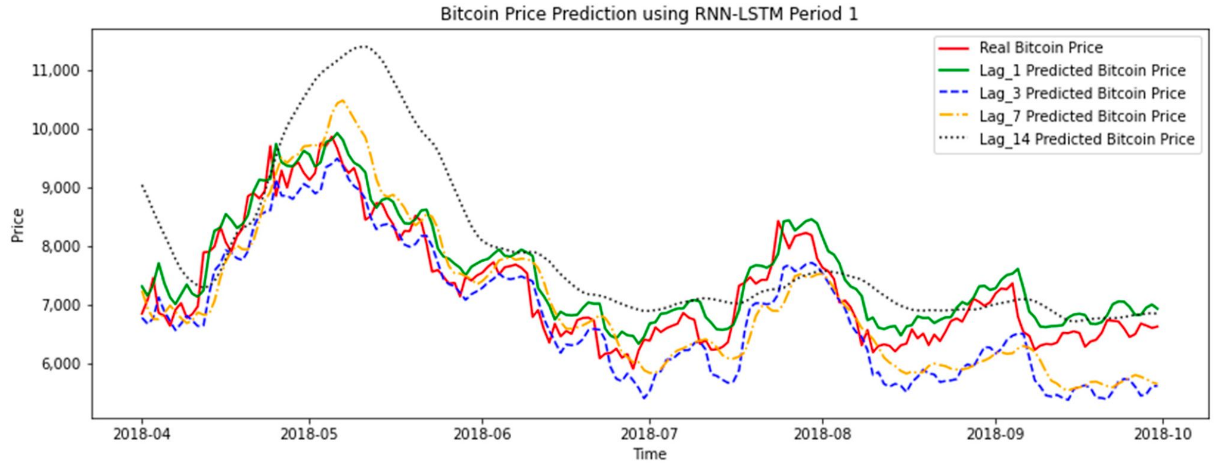


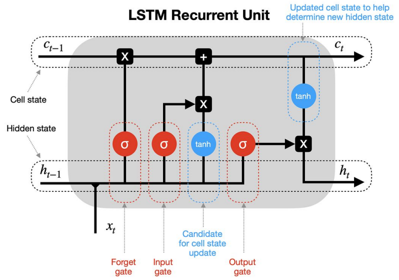
observed
Forecast
2500
2000
1500
1000
500
0
−500
Furniture Sales
2015
2017
2019
2021
2023
2025
Date

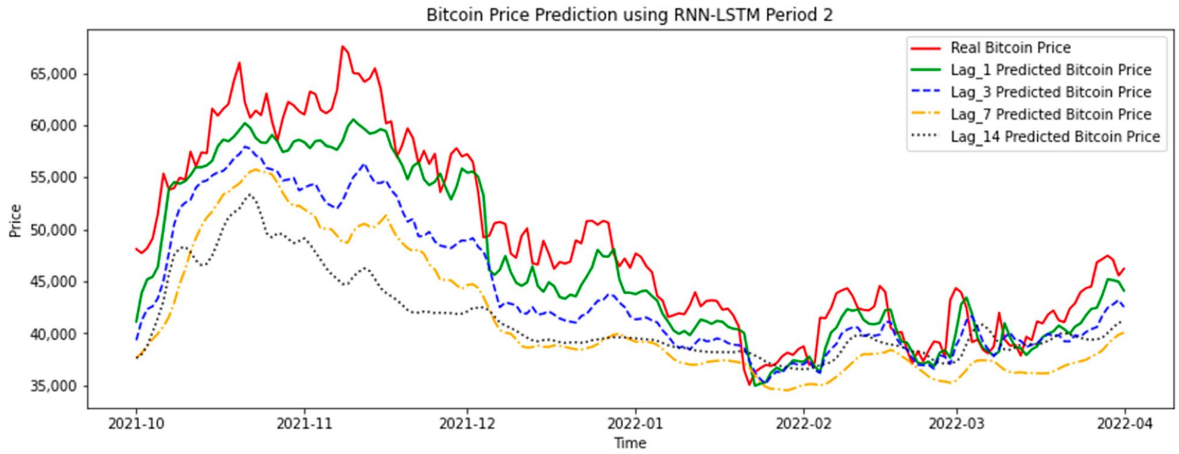

Predicted and Expected BTC Price (Prophet Predict)
Predicted
Upper Prediction
Lower Prediction
Actual
60000
40000
20000
0
−20000
Price ($)
Jul
Jan
2018
Jul
Jan
2019
Jul
Jan
2020
Jul
Jan
2021
Date

https://www.mdpi.com/1911-8074/16/1/51

LONG SHORT-TERM MEMORY NEURAL NETWORKS



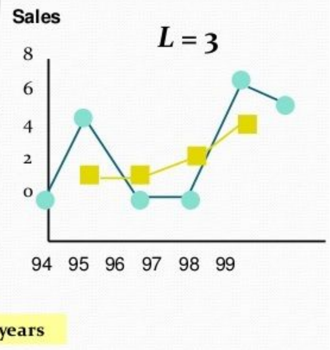




Moving Average

The moving average (MA) is a simple technical analysis tool that smooths out price data by creating a constantly updated average price. The average is taken over a specific period of time, like 10 days, 20 minutes, 30 weeks, or any time period the trader chooses.

Year	Response	Moving Avg.
1994	2	—
1995	5	3
1996	2	3
1997	2	3.67
1998	7	5
1999	6	—



Month	Sales ($000)	Three-month moving total ($000)	Three-month moving average ($000)	Seasonal variation ($000)
January	125			
February	145	456=(125+145+186)	(456 ÷ 3) = 152	
March	186	462=(145+186+131)	(462 ÷ 3) = 154	
April	131	468=(186+131+151)	(468 ÷ 3) = 156	
May	151	474	158	
June	192	480	160	
July	137	486	162	
August	157	492	164	
September	198	498	166	
October	143	504	168	
November	163	510	170	
December	204			

Exponential Smoothing

Exponential smoothing is a broadly accurate forecasting method for short-term forecasts. The technique assigns larger weights to more recent observations while assigning exponentially decreasing weights as the observations get increasingly distant. This method produces slightly unreliable long-term forecasts.

$$F = \alpha A + (1-\alpha)B$$

Key:
F: forecast, A: actual sales data, B: forecast sales from previous periods, α: the smoothing constant that is set as a value between 0.1 and 1.0.

https://www.youtube.com/watch?v=PIvvWrvFDvY

$\alpha = 0.4$

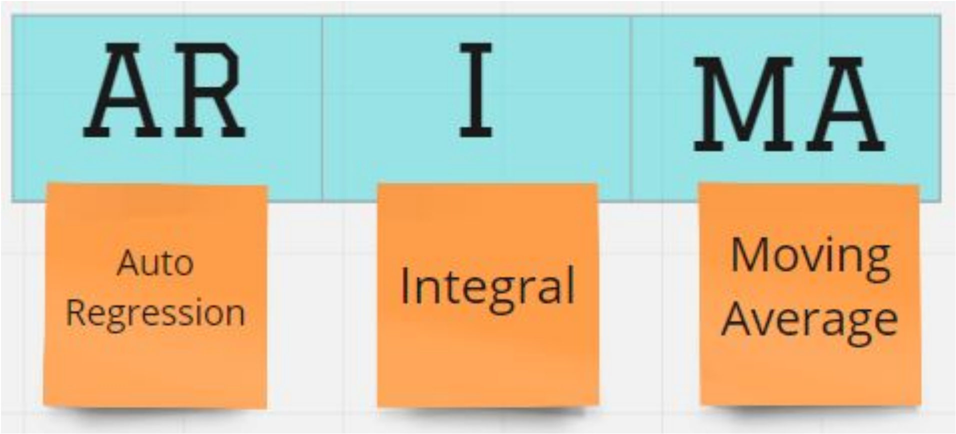
$$F_{t+1} = \alpha A_t + (1-\alpha)F_t$$

$(1-\alpha) = 1-0.4 \rightarrow 0.6$

Year	Actual Demand (A_t)	Forecast Demand (F_t)	
1	310	310	
2	365	310	$\rightarrow 0.4(310) + 0.6(310)$
3	395	332	$\rightarrow 0.4(365) + 0.6(310)$
4	415	357	$\rightarrow 0.4(395) + 0.6(332)$
5	450	380	$\rightarrow 0.4(415) + 0.6(357)$
6	465	408	$\rightarrow 0.4(450) + 0.6(380)$
7		431	$\rightarrow 0.4(465) + 0.6(408)$

ARIMA

An autoregressive integrated moving average (ARIMA) model is a statistical analysis model that leverages time series data to forecast future trends.



https://twitter.com/daansan_ml/status/1734479020195254377

ARIMA models family

When to use each component?

S — **Seasonal**
recurring pattern or variation in the data at fixed intervals

AR — **Auto-Regressive**
current value of the time series influenced by its past values

I — **Integrated**
to make the time series stationary when there is a trend or seasonality

MA — **Moving Average**
current value of the time series influenced by its past residuals

X — **eXogenous variables**
there are external factors that impact the time series

@daansan_ml

yahoo!
finance

Disclaimer

The topic of "stock prediction" discussed here is intended for **academic purposes only, focusing on time series analysis.**

It is not intended to provide commercial advice or encourage students to engage in trading.

The techniques presented are for **educational use** and should not be used for making real-world investment decisions.

Trading and investing involve risks, and individuals should seek professional financial advice before making any investment decisions based on the concepts discussed here.

Kao (Feb 16, 2024)

```python
1 # Fetch Apple stock data from Yahoo Finance
2 apple_data = yf.download('AAPL', start='2010-01-01', end='2022-01-01', progress=False)
3 apple_data.head()
```

Date	Open	High	Low	Close	Adj Close	Volume
2010-01-04	7.622500	7.660714	7.585000	7.643214	6.470740	493729600
2010-01-05	7.664286	7.699643	7.616071	7.656429	6.481928	601904800
2010-01-06	7.656429	7.686786	7.526786	7.534643	6.378825	552160000
2010-01-07	7.562500	7.571429	7.466071	7.520714	6.367033	477131200
2010-01-08	7.510714	7.571429	7.466429	7.570714	6.409363	447610800

Next steps: View recommended plots

Understanding the data -

In finance stocks, **Open, High, Low, Close, Adjusted Close, and Volume** are commonly used terms to describe different aspects of a stock’s performance on a particular day or over a certain period of time.

Here’s what each term means:

Open: The opening price of a stock is the price at which it starts trading for the day. This is usually based on the closing price from the previous day, but can also be affected by after-hours trading or news events that occur before the market opens.

Understanding the data -

In finance stocks, Open, High, Low, Close, Adjusted Close, and Volume are commonly used terms to describe different aspects of a stock's performance on a particular day or over a certain period of time.

Here's what each term means:

High: The high price of a stock is the highest price that it reached during the trading day. This can be an important indicator of the stock's performance, as it shows how high investors were willing to bid for the stock.

Understanding the data -

In finance stocks, Open, High, Low, Close, Adjusted Close, and Volume are commonly used terms to describe different aspects of a stock's performance on a particular day or over a certain period of time.

Here's what each term means:

Low: The low price of a stock is the lowest price that it reached during the trading day. This can be an important indicator of the stock's performance, as it shows how low investors were willing to bid for the stock.

Understanding the data -

In finance stocks, Open, High, Low, Close, Adjusted Close, and Volume are commonly used terms to describe different aspects of a stock’s performance on a particular day or over a certain period of time.

Here’s what each term means:

Close: The closing price of a stock is the price at which it ends trading for the day. This is usually based on the last trade of the day, but can also be affected by after-hours trading or news events that occur after the market closes.

Understanding the data -

In finance stocks, Open, High, Low, Close, Adjusted Close, and Volume are commonly used terms to describe different aspects of a stock's performance on a particular day or over a certain period of time.

Here's what each term means:

Adjusted Close: The adjusted close price of a stock takes into account any corporate actions that might affect the stock's value, such as stock splits or dividend payments. This is important because it allows investors to track the actual performance of the stock, rather than just the price movements.

Understanding the data -

In finance stocks, Open, High, Low, Close, Adjusted Close, and Volume are commonly used terms to describe different aspects of a stock's performance on a particular day or over a certain period of time.

Here's what each term means:

Volume: The volume of a stock is the number of shares that were traded during the trading day. This can be an important indicator of the stock's liquidity, as it shows how many shares were available for investors to buy or sell. High volume can also indicate increased investor interest in the stock.

Model Evaluation: Mean Squared Error (MSE)

MSE measures the average squared difference between the predicted and actual values, giving higher weight to large errors. MAE, on the other hand, measures the average absolute difference between the predicted and actual values, giving equal weight to all errors.

$$\text{MSE} = \overset{\text{Mean}}{\boxed{\frac{1}{n}\sum_{i=1}^{n}}} \Big(\overset{\text{Error}}{\boxed{Y_i - \hat{Y}_i}}\Big)^{\overset{\text{Squared}}{\boxed{2}}}$$

(Optional) Financial Markets

Following are some of the major stock markets and their market capitalizations:

STOCK EXCHANGE	SYMBOL	CITY	MARKET CAP 2019 ($B)
New York Stock Exchange	NYSE	New York City	22,923
Nasdaq	NASDAQ	New York City	10,857
Japan Exchange Group	JPX	Tokyo	5,679
Shanghai Stock Exchange	SSE	Shanghai	4,026
Hong Kong Stock Exchange	SEHK	Hong Kong	3,936
London Stock Exchange	LSE	London	3,767
Shenzhen Stock Exchange	SZSE	Shenzhen	2,504
TMX Group	TSX	Toronto	2,095
Bombay Stock Exchange	BSE	Mumbai	2,056
Australian Securities Exchange	ASX	Sydney	1,328
Ho Chi Minh Stock Exchange	HSX	Ho Chi Minh	128
Pakistan Stock Exchange	PSX	Karachi	54

(Optional) Market Indexes

MARKET INDEX	SYMBOL	Description	
Dow Jones Industrial Average	DJIA	or Dow, is an index that tracks the 30 largest, publicly owned companies trading on the New York Stock Exchange**	NYSE) and the NASDAQ.
Nasdaq Composite	IXIC	or Nasdaq, is an index of more than 3,000 stocks listed on the Nasdaq exchange.	
Standard & Poor's 500 Index	GSPC	or S&P 500, is an index of the 500 largest U.S. publicly traded companies. The index is widely regarded as the best gauge of large-cap U.S. equities. S&P500 is the **market benchmark**.	
CBOE Volatility Index10	VIX	is a real-time market index that represents the market's expectation of 30-day forward-looking volatility and is derived from the price inputs of the S&P500 index options. It is also known by "Fear Gauge" or "Fear Index."	
Financial Times Stock Exchange 100 Share Index	FTSE	or *Footsie*, is the dominant index, containing 100 of the top blue chips on the London Stock Exchange.	
Nikkei Index	N225	is composed of Japan's top 225 blue-chip companies traded on the Tokyo Stock Exchange.	
Hang Seng Index	HSI	is an index of the largest companies that trade on the Hong Kong Exchange.	
Sensex	BSESB	also known as the S&P BSE Sensex index, is the benchmark index comprising of 30 of the largest and most actively-traded stocks on the Bombay Stock Exchange.	
Karachi Stock Exchange	KSE-100	consists of 100 companies representing about 90 percent of market capitalization of the Pakistan Stock Exchange.	

(Optional) Financial Instruments

SECURITY	Description
Stocks	an ownership position in a publicly-traded corporation
Bonds	a creditor relationship with a governmental body or a corporation
Derivative (Options)	or rights to ownership – also called underlying financial instrument

(Optional) Investment Instruments

INSTRUMENT	Description
Mutual Funds	A mutual fund is a company that pools money from many investors and invests in securities such as stocks, bonds, and short-term debt (bonds). The combined holdings of the mutual fund are known as its *portfolio*.
Exchange-Traded Funds (ETF)	are in many ways similar to mutual funds, however, they are listed on exchanges and ETF shares trade throughout the day just like ordinary stock.
Index and Sector Funds	is similar to an ETF, except the index fund portfolio consists of securities listed on a particular market index (such as Nasdaq or S&P500).
Hedge Funds	have very aggressive portfolios and are very high-risk. Hedge-funds are tailored to high-end investor.
Real Estate Investment Trusts (REITs)	allow individuals to invest in large-scale, income-producing real estate.
Certificate of Deposit (CD)	is a savings account that holds a fixed amount of money for a fixed period of time, and are considered to be one of the safest savings option.

Week 12: Time Series: Stock Price Prediction

[1] 0s
```
# !pip install yfinance
# !pip install statsmodels
# !pip install pmdarima
```

[2] 0s
```
import warnings
warnings.filterwarnings("ignore")
```

[3] 1s
```
import yfinance as yf
import pandas as pd
from sklearn.model_selection import TimeSeriesSplit
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt

from statsmodels.tsa.arima.model import ARIMA
from statsmodels.tsa.holtwinters import ExponentialSmoothing
```

Moving Average

```
# Fetch Apple stock data from Yahoo Finance
apple_data = yf.download('AAPL', start='2010-01-01', end='2022-01-01', progress=False)
apple_data.head()
```

Date	Open	High	Low	Close	Adj Close	Volume
2010-01-04	7.622500	7.660714	7.585000	7.643214	6.470740	493729600
2010-01-05	7.664286	7.699643	7.616071	7.656429	6.481928	601904800
2010-01-06	7.656429	7.686786	7.526786	7.534643	6.378825	552160000
2010-01-07	7.562500	7.571429	7.466071	7.520714	6.367033	477131200
2010-01-08	7.510714	7.571429	7.466429	7.570714	6.409363	447610800

Next steps: View recommended plots

[5]

```python
# Select only the 'Close' column as our target variable
apple_close = apple_data['Close']
```

[6]

```python
# Split data into train and test sets using date-based split
train_end_date = '2021-01-01'
train_data = apple_close[:train_end_date]
test_data = apple_close[train_end_date:]
```

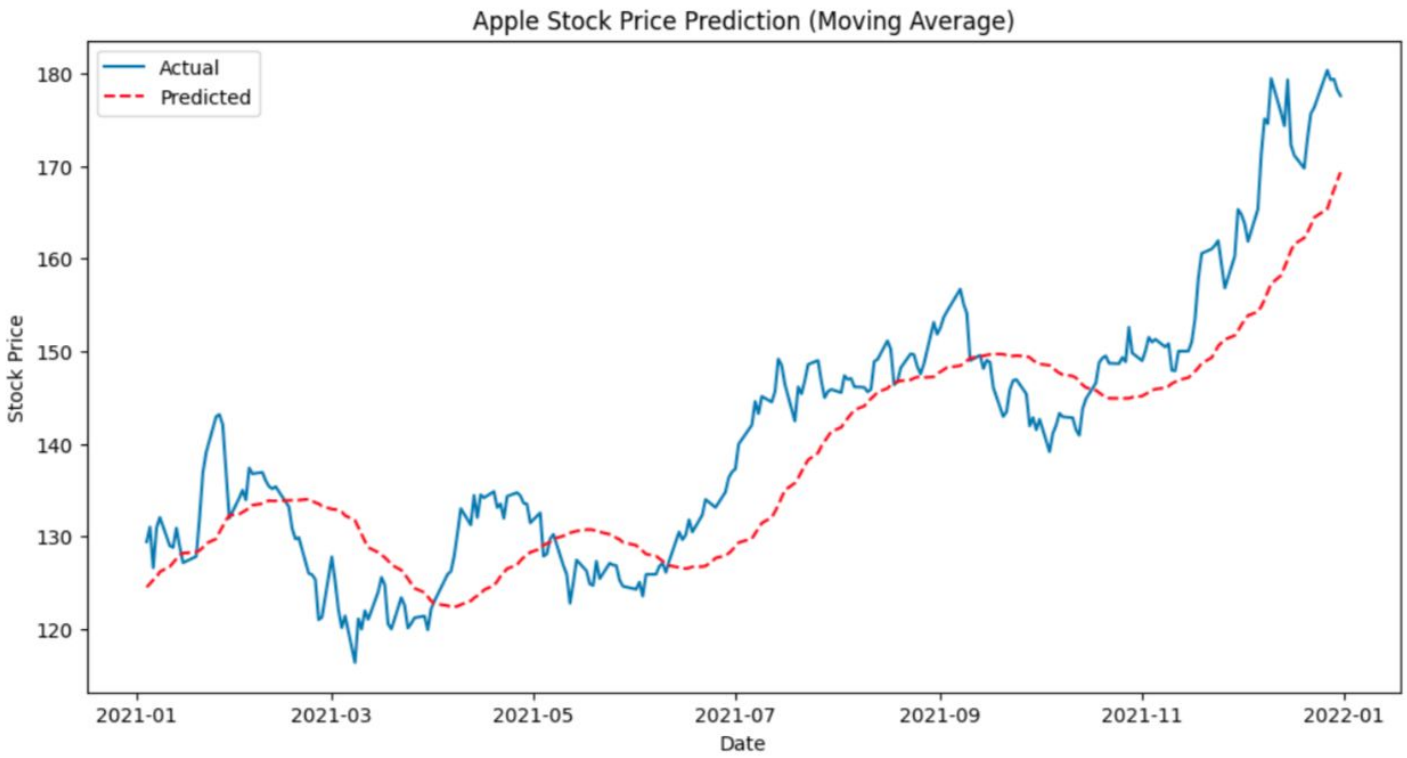
[7]

```python
# Define a simple Moving Average model
def moving_average_model(train_data, test_data, window_size):
    history = train_data.tolist()
    predictions = []
    for t in range(len(test_data)):
        # Calculate the rolling mean
        yhat = sum(history[-window_size:]) / window_size
        predictions.append(yhat)
        history.append(test_data[t])
    return predictions

# Evaluate the model
window_size = 30
predictions = moving_average_model(train_data, test_data, window_size)
mse = mean_squared_error(test_data, predictions)
print('Mean Squared Error (Moving Average):', mse)
```

```
Mean Squared Error (Moving Average): 54.59246045975243
```

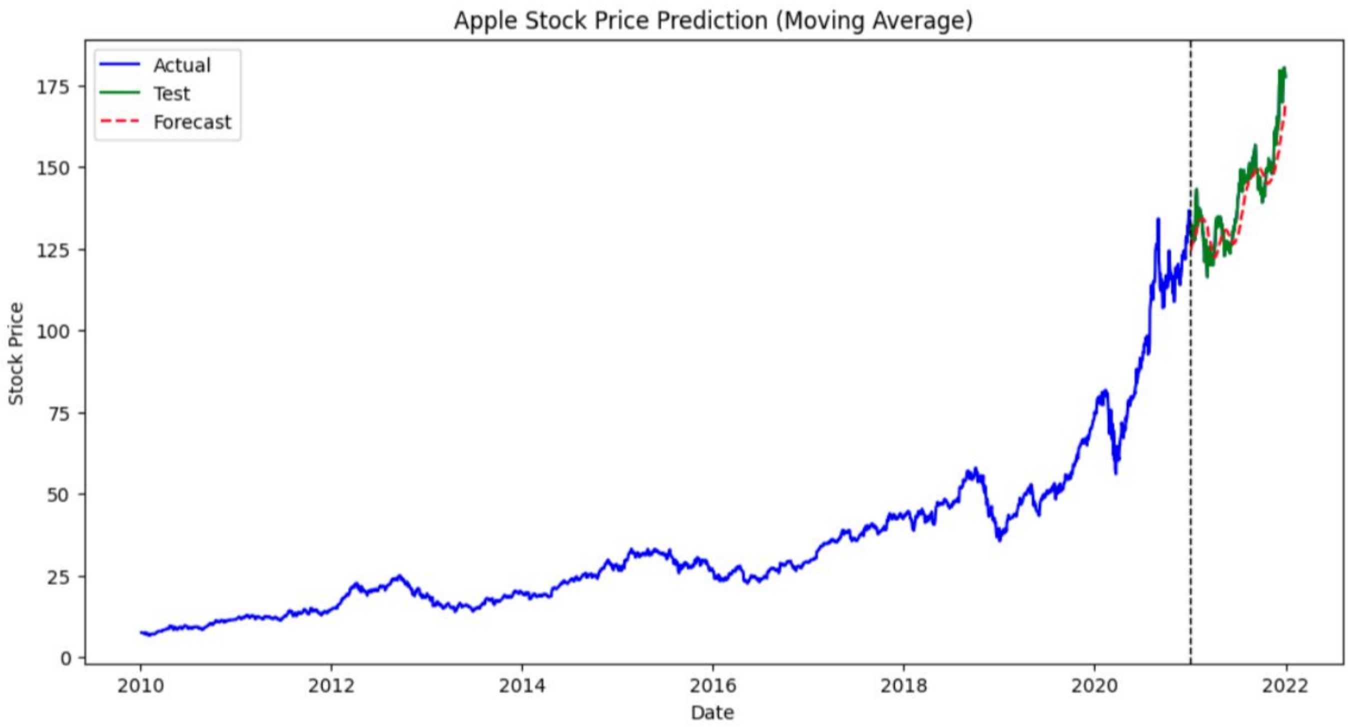
[8]

```python
# Plot the actual vs. predicted values
plt.figure(figsize=(12, 6))
plt.plot(test_data.index, test_data.values, label='Actual')
plt.plot(test_data.index, predictions, color='red', linestyle='--', label='Predicted')
plt.title('Apple Stock Price Prediction (Moving Average)')
plt.xlabel('Date')
plt.ylabel('Stock Price')
plt.legend()
plt.show()
```



[9]

```python
# Plot the entire dataset with training, testing, and forecasted values
plt.figure(figsize=(12, 6))
plt.plot(apple_close.index, apple_close.values, label='Actual', color='blue')
plt.plot(test_data.index, test_data.values, label='Test', color='green')
plt.plot(test_data.index, predictions, color='red', linestyle='--', label='Forecast')
plt.axvline(x=pd.Timestamp(train_end_date), color='black', linestyle='--', linewidth=1)  # Changed to use pd.Timestamp(train_end_date)
plt.title('Apple Stock Price Prediction (Moving Average)')
plt.xlabel('Date')
plt.ylabel('Stock Price')
plt.legend()
plt.show()
```



Exponential Smoothing

[33]

```python
# Fetch Apple stock data from Yahoo Finance
apple_data = yf.download('AAPL', start='2010-01-01', end='2022-01-01', progress=False)

# Select only the 'Close' column as our target variable
apple_close = apple_data['Close']

# Handle missing values by forward filling
apple_close.fillna(method='ffill', inplace=True)

# Split data into train and test sets using date-based split
train_end_date = '2021-01-01'
train_data = apple_close[:train_end_date]
test_data = apple_close[train_end_date:]
```

[34]

```python
# Exponential smoothing model
def exponential_smoothing_model(train_data, test_data, smoothing_level=0.2):
    model = ExponentialSmoothing(train_data, trend='add', seasonal=None)
    model_fit = model.fit(smoothing_level=smoothing_level)
    predictions = model_fit.forecast(steps=len(test_data))
    return predictions
```

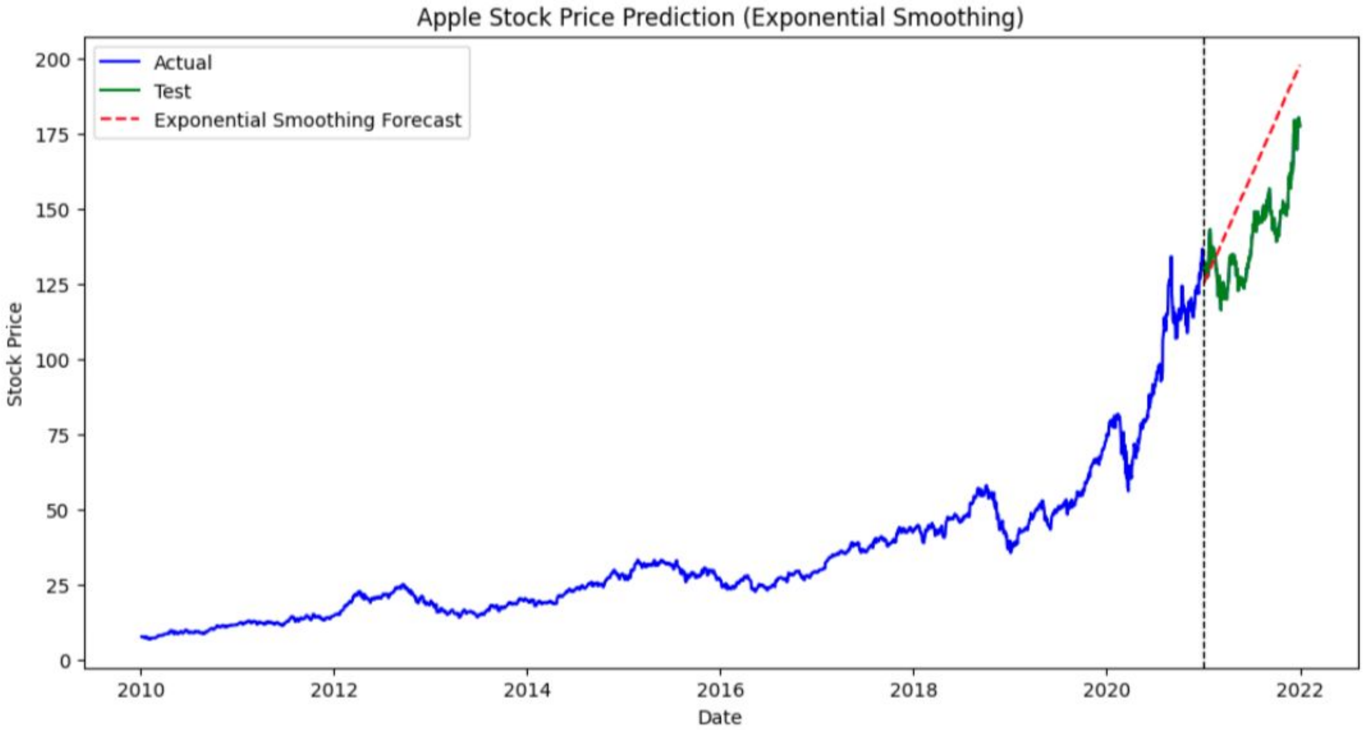
[35]

```python
# Evaluate the model
smoothing_level = 0.01
predictions = exponential_smoothing_model(train_data, test_data, smoothing_level)
mse = mean_squared_error(test_data, predictions)
print('Mean Squared Error (Exponential Smoothing):', mse)
```

```
Mean Squared Error (Exponential Smoothing): 571.444269293314
```

[36]

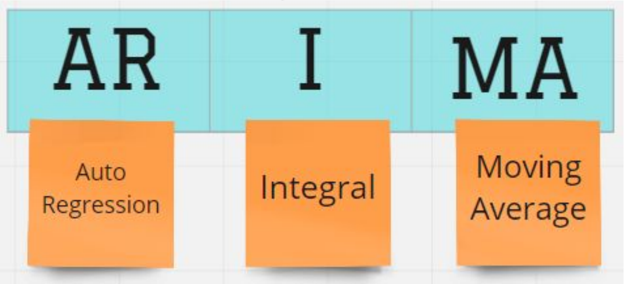
```
# Plot the entire dataset with training, testing, and forecasted values
plt.figure(figsize=(12, 6))
plt.plot(apple_close.index, apple_close.values, label='Actual', color='blue')
plt.plot(test_data.index, test_data.values, label='Test', color='green')
plt.plot(test_data.index, predictions, color='red', linestyle='--', label='Exponential Smoothing Forecast')
plt.axvline(x=pd.Timestamp(train_end_date), color='black', linestyle='--', linewidth=1)  # Changed to use pd.Timestamp(train_end_date)
plt.title('Apple Stock Price Prediction (Exponential Smoothing)')
plt.xlabel('Date')
plt.ylabel('Stock Price')
plt.legend()
plt.show()
```



ARIMA

[37]

```
# Fetch Apple stock data from Yahoo Finance
apple_data = yf.download('AAPL', start='2010-01-01', end='2022-01-01', progress=False)

# Select only the 'Close' column as our target variable
apple_close = apple_data['Close']

# Split data into train and test sets using date-based split
train_end_date = '2021-01-01'
train_data = apple_close[:train_end_date]
test_data = apple_close[train_end_date:]
```



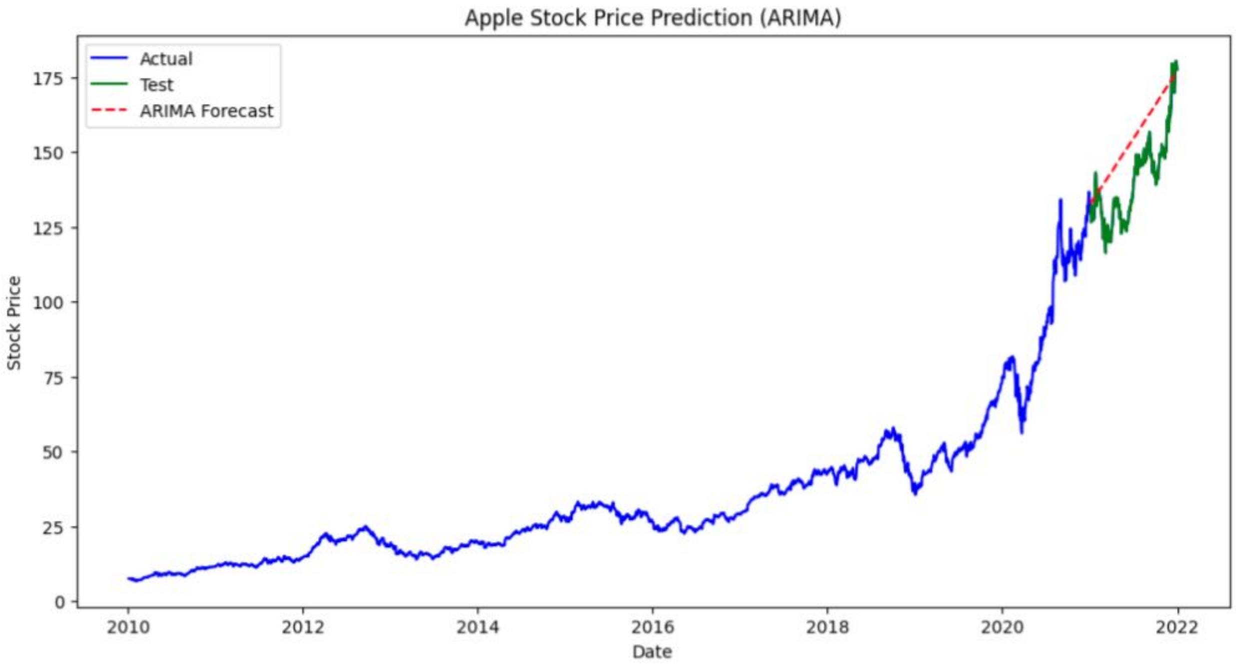
[38]

```
# ARIMA model
def fit_arima(train_data, order):
    model = ARIMA(train_data, order=order)
    fitted_model = model.fit()
    return fitted_model

# Evaluate the model
def evaluate_model(model, test_data):
    predictions = model.forecast(steps=len(test_data))
    mse = mean_squared_error(test_data, predictions)
    return mse, predictions

# ARIMA
order = (1, 2, 2)
arima_model = fit_arima(train_data, order)
arima_mse, arima_predictions = evaluate_model(arima_model, test_data)

print('Mean Squared Error (ARIMA):', arima_mse)
```

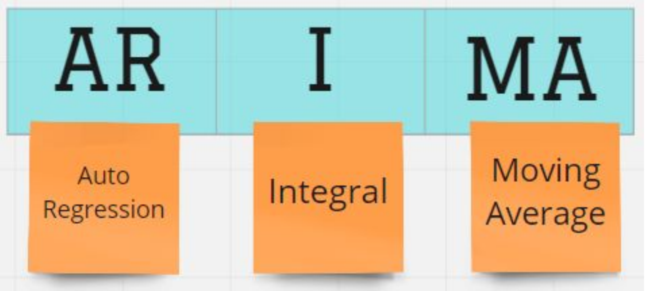
```
Mean Squared Error (ARIMA): 256.92254300338965
```

[39]
```python
# Forecast
forecast_steps = len(test_data)
forecast = arima_model.forecast(steps=forecast_steps)
```

[40]
```python
# Plot the entire dataset
plt.figure(figsize=(12, 6))
plt.plot(apple_close.index, apple_close.values, label='Actual', color='blue')
plt.plot(test_data.index, test_data.values, label='Test', color='green')
plt.plot(test_data.index, arima_predictions, color='red', linestyle='--', label='ARIMA Forecast')
plt.title('Apple Stock Price Prediction (ARIMA)')
plt.xlabel('Date')
plt.ylabel('Stock Price')
plt.legend()
plt.show()
```



Search for best ARIMA order



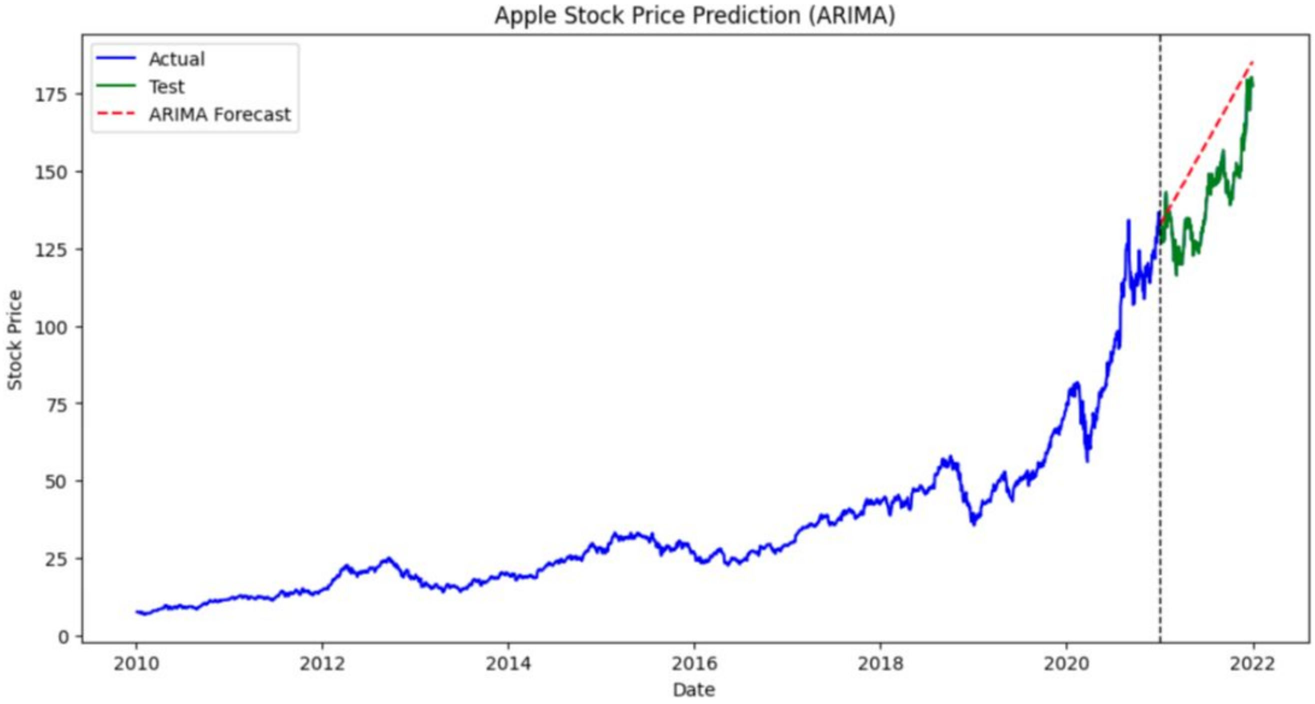
[41]

```python
# Grid search for best ARIMA order

p_values = range(0, 3)  # AR parameter
d_values = range(0, 3)  # I(d) parameter
q_values = range(0, 3)  # MA parameter

best_mse = float('inf')
best_order = None

for p in p_values:
    for d in d_values:
        for q in q_values:
            order = (p, d, q)
            try:
                arima_model = ARIMA(train_data, order=order)
                arima_model_fit = arima_model.fit()
                arima_predictions = arima_model_fit.forecast(steps=len(test_data))
                arima_mse = mean_squared_error(test_data, arima_predictions)
                if arima_mse < best_mse:
                    best_mse = arima_mse
                    best_order = order
            except:
                continue

print('Best ARIMA Order:', best_order)
print('Best ARIMA MSE:', best_mse)
```

```
Best ARIMA Order: (1, 2, 2)
Best ARIMA MSE: 256.92254300338965
```

[42]

```
# Plot the entire dataset with training, testing, and forecasted values
plt.figure(figsize=(12, 6))
plt.plot(apple_close.index, apple_close.values, label='Actual', color='blue')
plt.plot(test_data.index, test_data.values, label='Test', color='green')
plt.plot(test_data.index, arima_predictions, color='red', linestyle='--', label='ARIMA Forecast')
plt.axvline(x=pd.Timestamp(train_end_date), color='black', linestyle='--', linewidth=1)  # Changed to use pd.Timestamp(train_end_date)
plt.title('Apple Stock Price Prediction (ARIMA)')
plt.xlabel('Date')
plt.ylabel('Stock Price')
plt.legend()
plt.show()
```



Homework

[47]
```
import yfinance as yf

# Thai stock ticker symbols
thai_tickers = 'PTT.BK' # ['AOT.BK', 'SCB.BK', 'PTT.BK', 'CPALL.BK', 'BDMS.BK']

# Download stock data
thai_stock_data = yf.download(thai_tickers, start='2010-01-01', end='2022-01-01')
```

```
[*********************100%***********************]  1 of 1 completed
```

[48]
```
thai_stock_data.head()
```

Date	Open	High	Low	Close	Adj Close	Volume
2010-01-04	24.600000	24.700001	24.100000	24.5	13.929267	40044000
2010-01-05	24.799999	25.000000	24.299999	24.4	13.872417	69048000
2010-01-06	24.500000	24.600000	24.299999	24.5	13.929267	29298000
2010-01-07	24.700001	24.799999	24.400000	24.4	13.872417	48300000
2010-01-08	24.400000	24.799999	24.400000	24.6	13.986121	41024000

Next steps: View recommended plots

Week 12: Assignment

Time Series Analysis and Forecasting: Stock Price Prediction

Understanding time series analysis and forecasting is crucial for financial analysis and investment decision-making.

SET
The Stock Exchange of Thailand