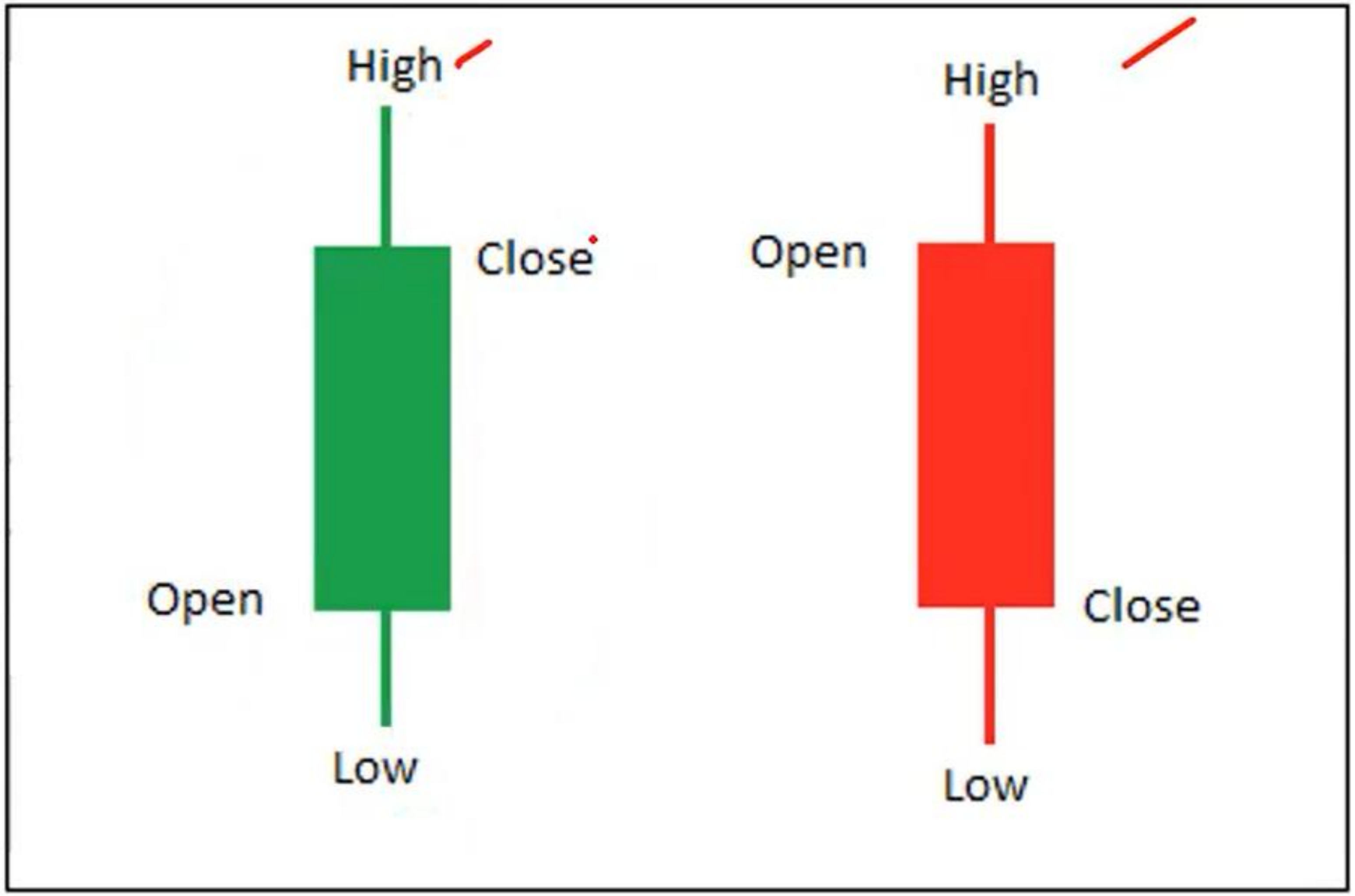
Time Series Analysis

Candle stick



TSA- Supervised Regression Problem

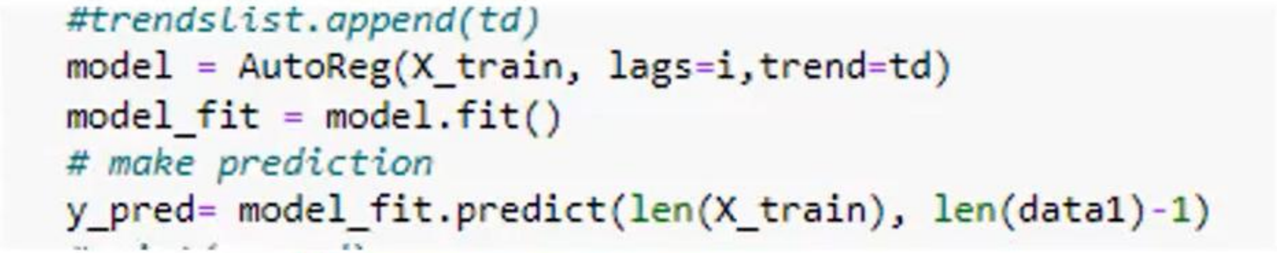
Time Series -Regression

Vs

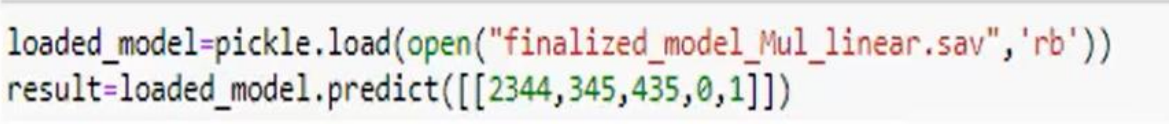
Machine Learning-Regression

Training and Test should be series

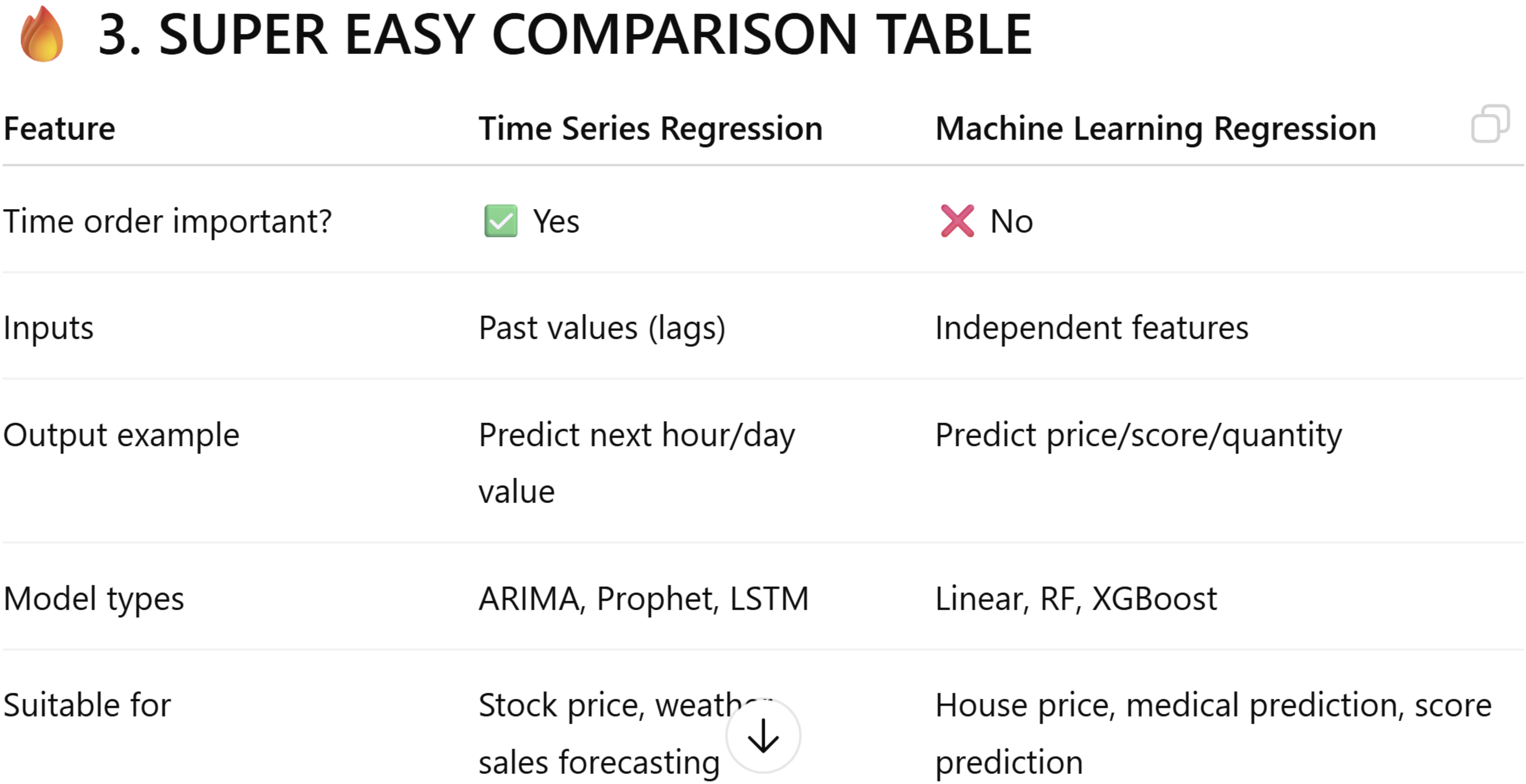
Training and Test should be random



```
#trendslist.append(td)
model = AutoReg(X_train, lags=i,trend=td)
model_fit = model.fit()
# make prediction
y_pred= model_fit.predict(len(X_train), len(data1)-1)
```



```
loaded_model=pickle.load(open("finalized_model_Mul_linear.sav",'rb'))
result=loaded_model.predict([[2344,345,435,0,1]])
```

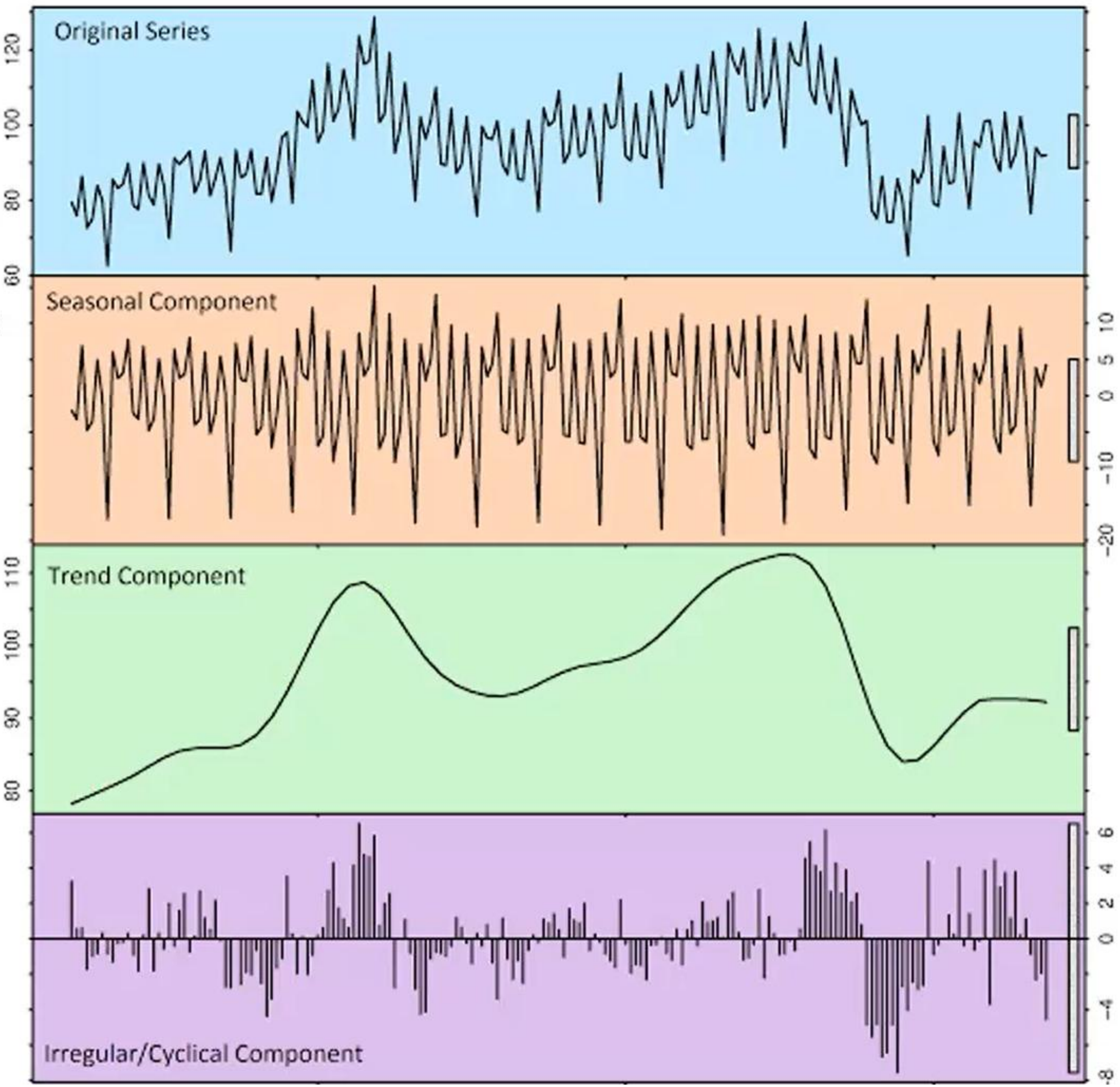


🔥 3. SUPER EASY COMPARISON TABLE

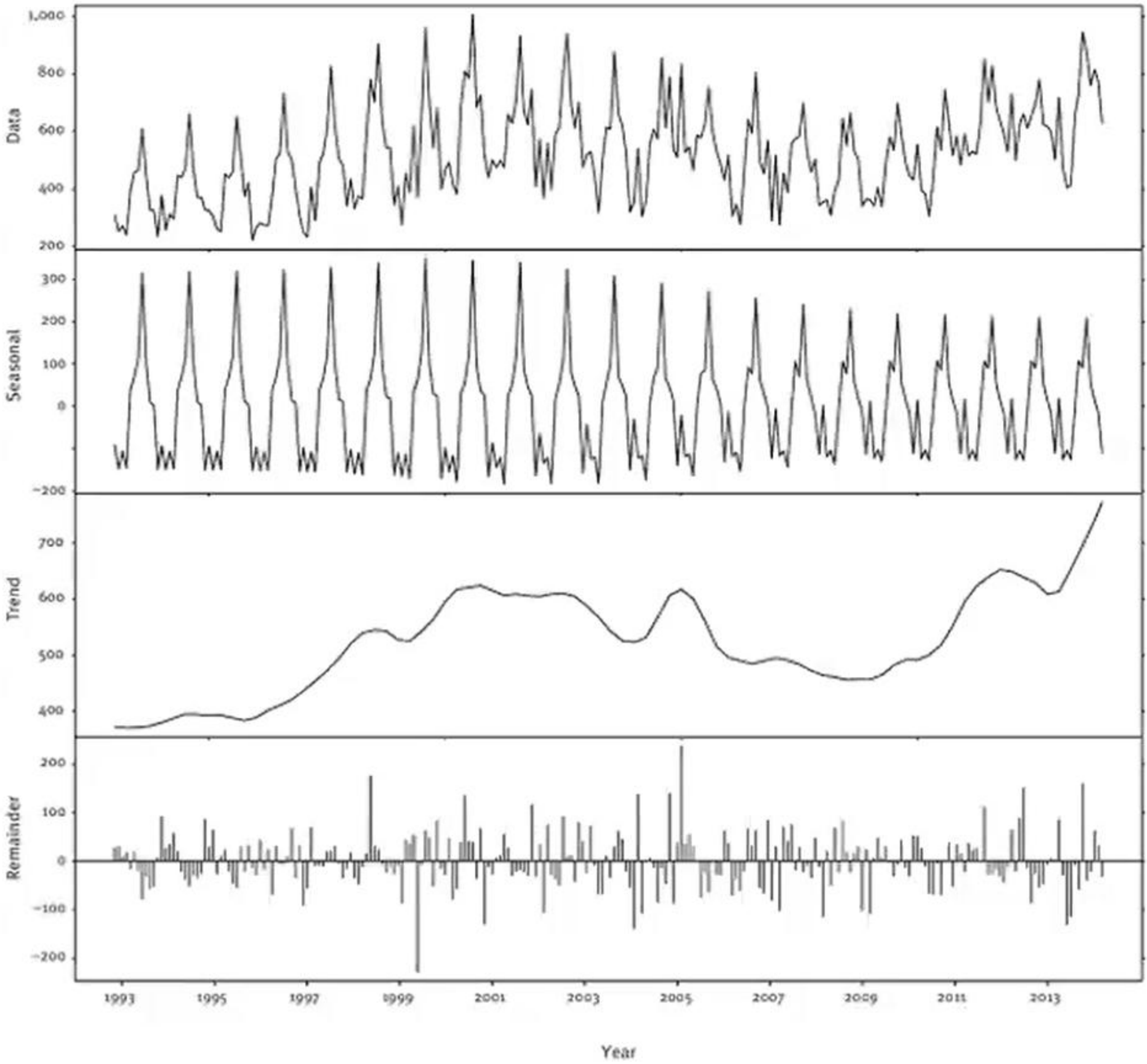
Feature	Time Series Regression	Machine Learning Regression
Time order important?	✅ Yes	❌ No
Inputs	Past values (lags)	Independent features
Output example	Predict next hour/day value	Predict price/score/quantity
Model types	ARIMA, Prophet, LSTM	Linear, RF, XGBoost
Suitable for	Stock price, weather, sales forecasting	House price, medical prediction, score prediction

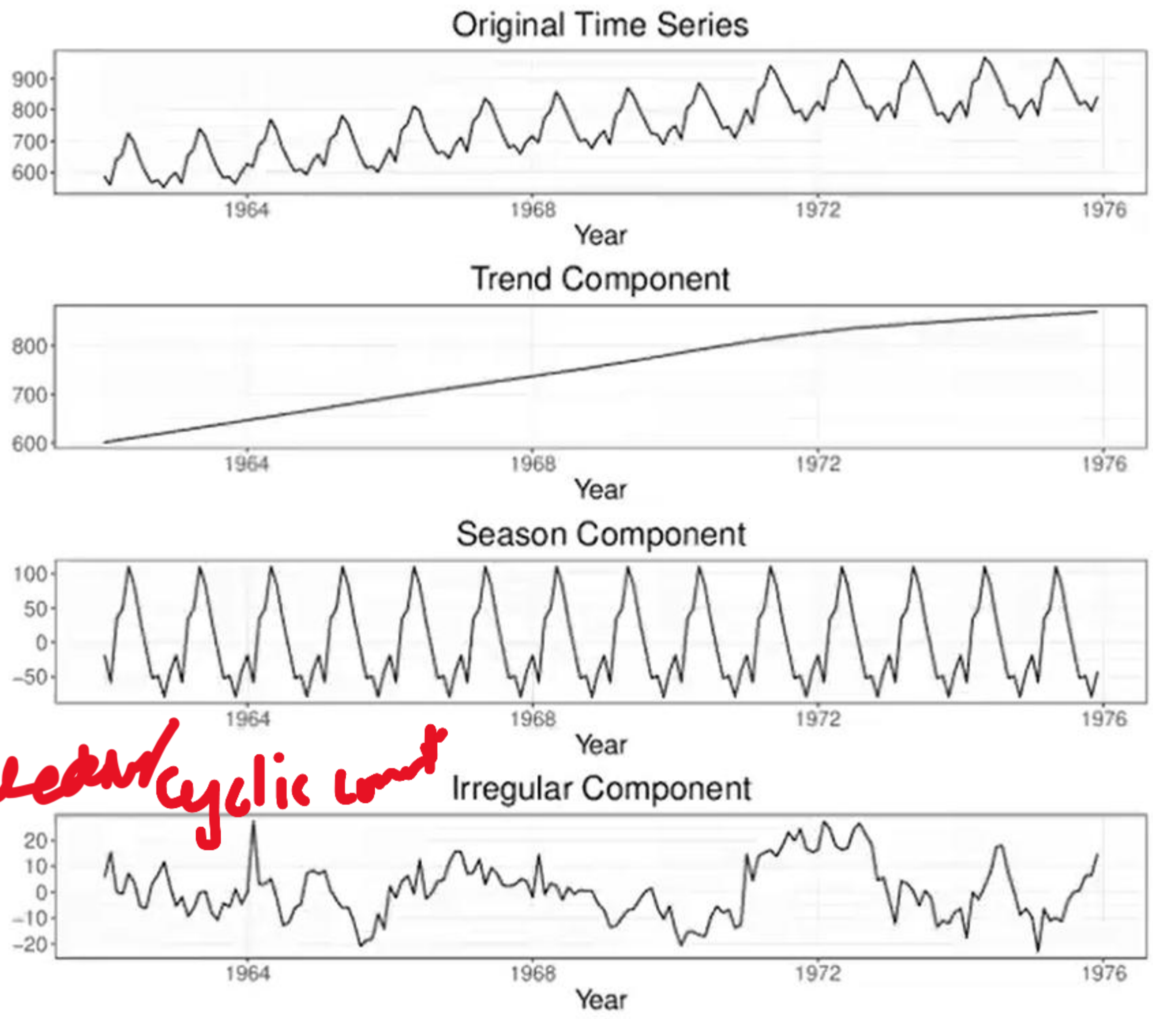
Time series graph

- Original = Seasonal + Trend + Noise



Time Series Analysis



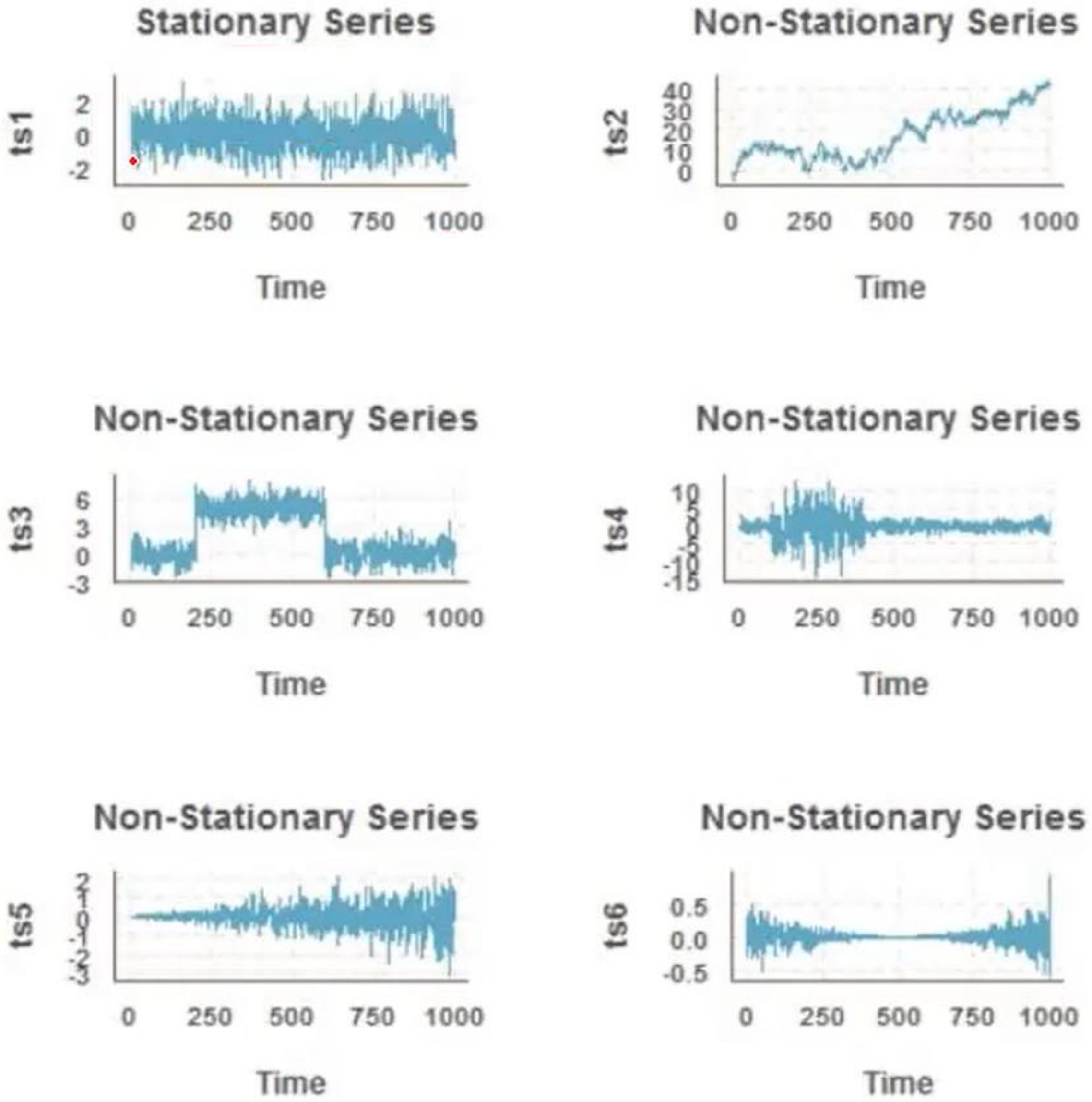


Residual/cyclic cont

07:17

Time Series Analysis

Time series Assumption Stationarity


Stationary Series

Non-Stationary Series

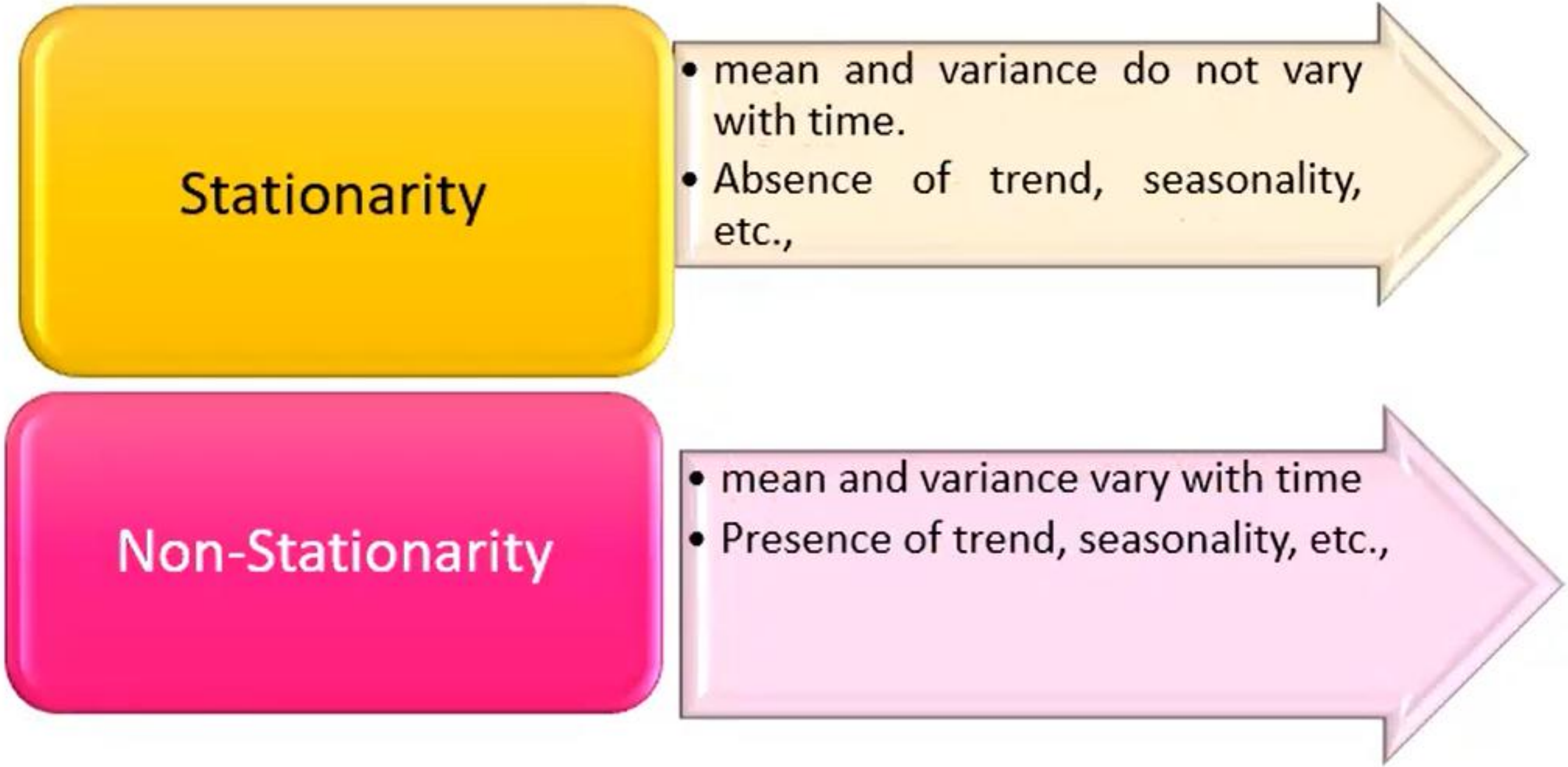
Non-Stationary Series

Non-Stationary Series

Non-Stationary Series

Non-Stationary Series

- Mean and Variance is same over the period of time

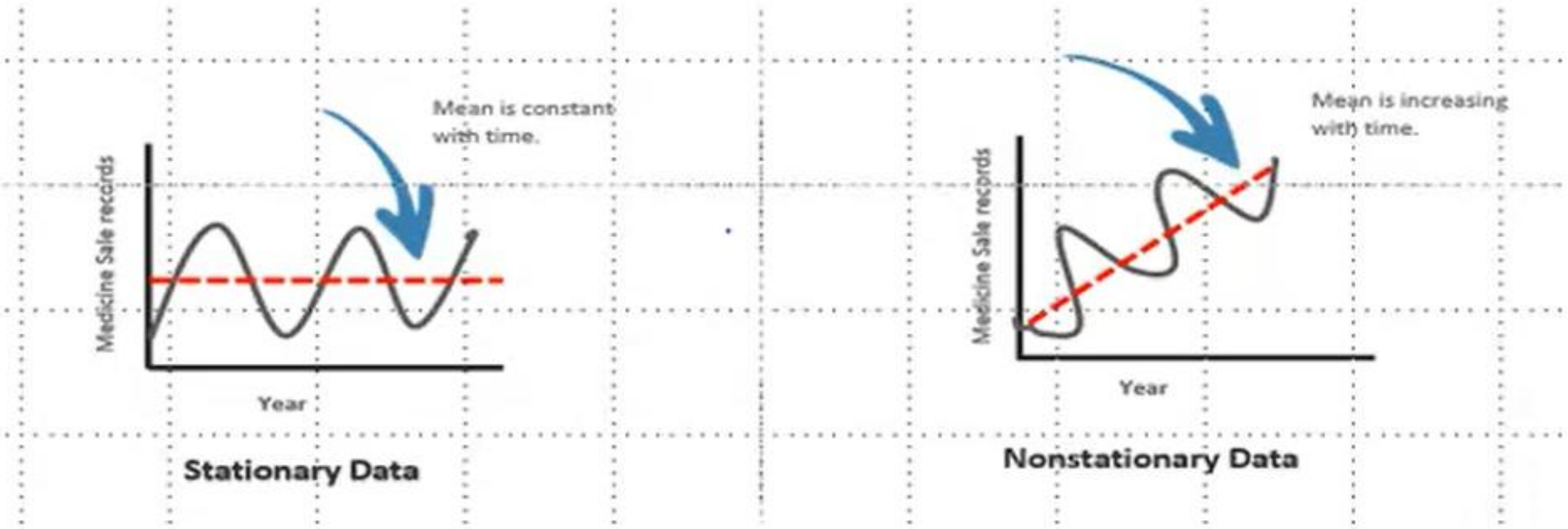


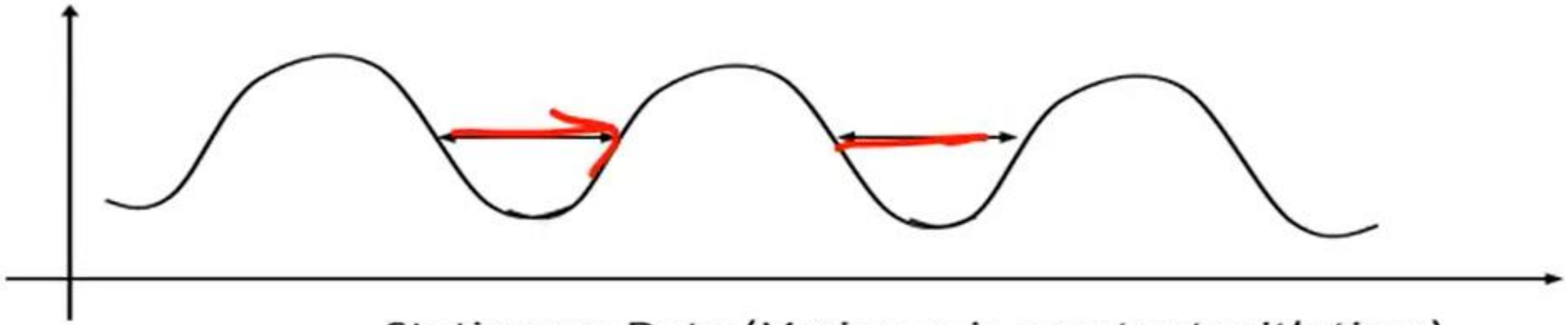
Stationarity

- mean and variance do not vary with time.
- Absence of trend, seasonality, etc.,

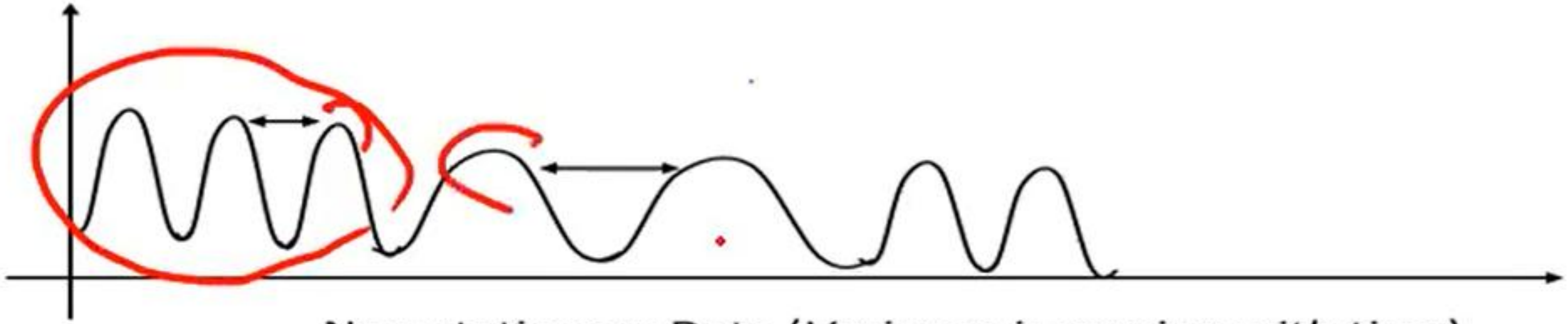
Non-Stationarity

- mean and variance vary with time
- Presence of trend, seasonality, etc.,





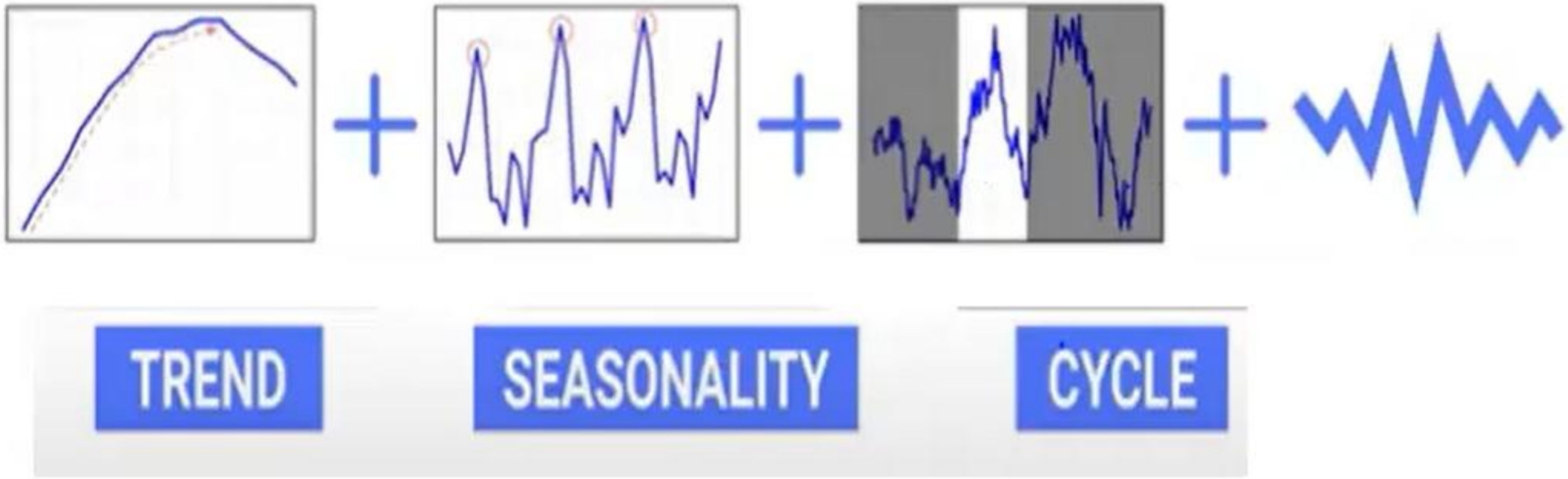
Stationary Data (*Variance is constant with time*)



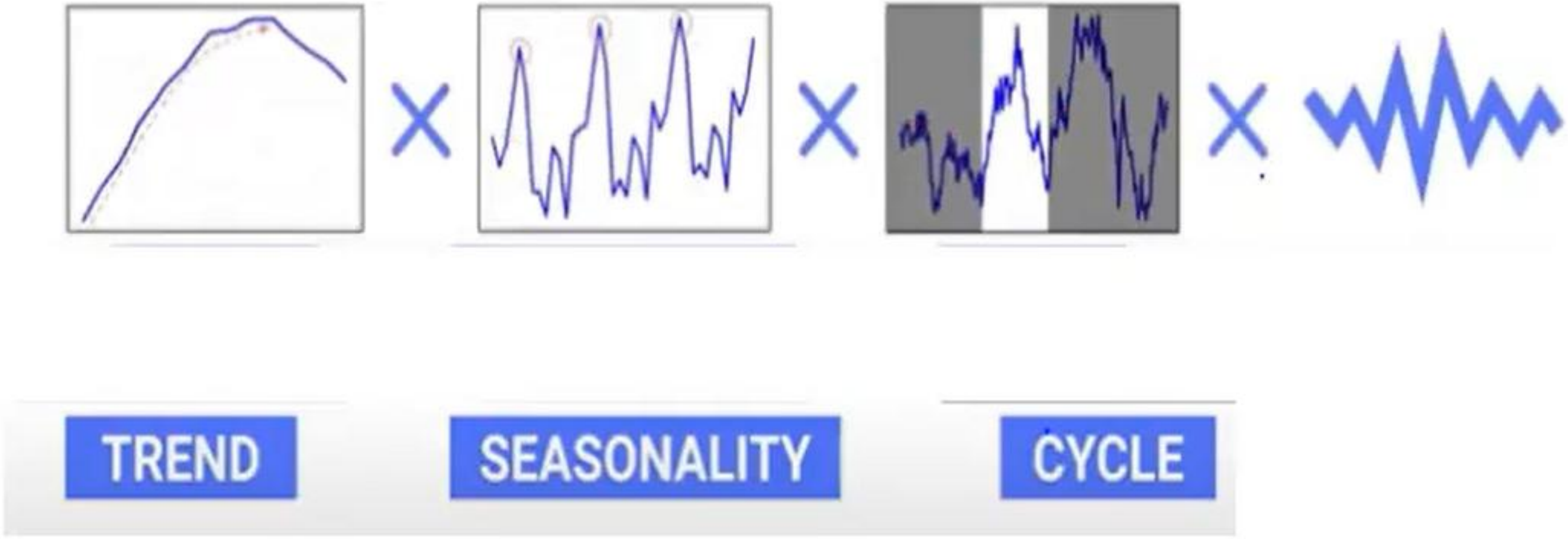
Non-stationary Data (*Variance is varying with time*)

ADDITIVE AND MULTIPLICATIVE TIME SERIES

Additive



MULTIPLICATIVE



How to Check Stationarity

Hypothesis

Null : There is n

Alter : There i

- Augmented Dickey-Fuller Test ✓

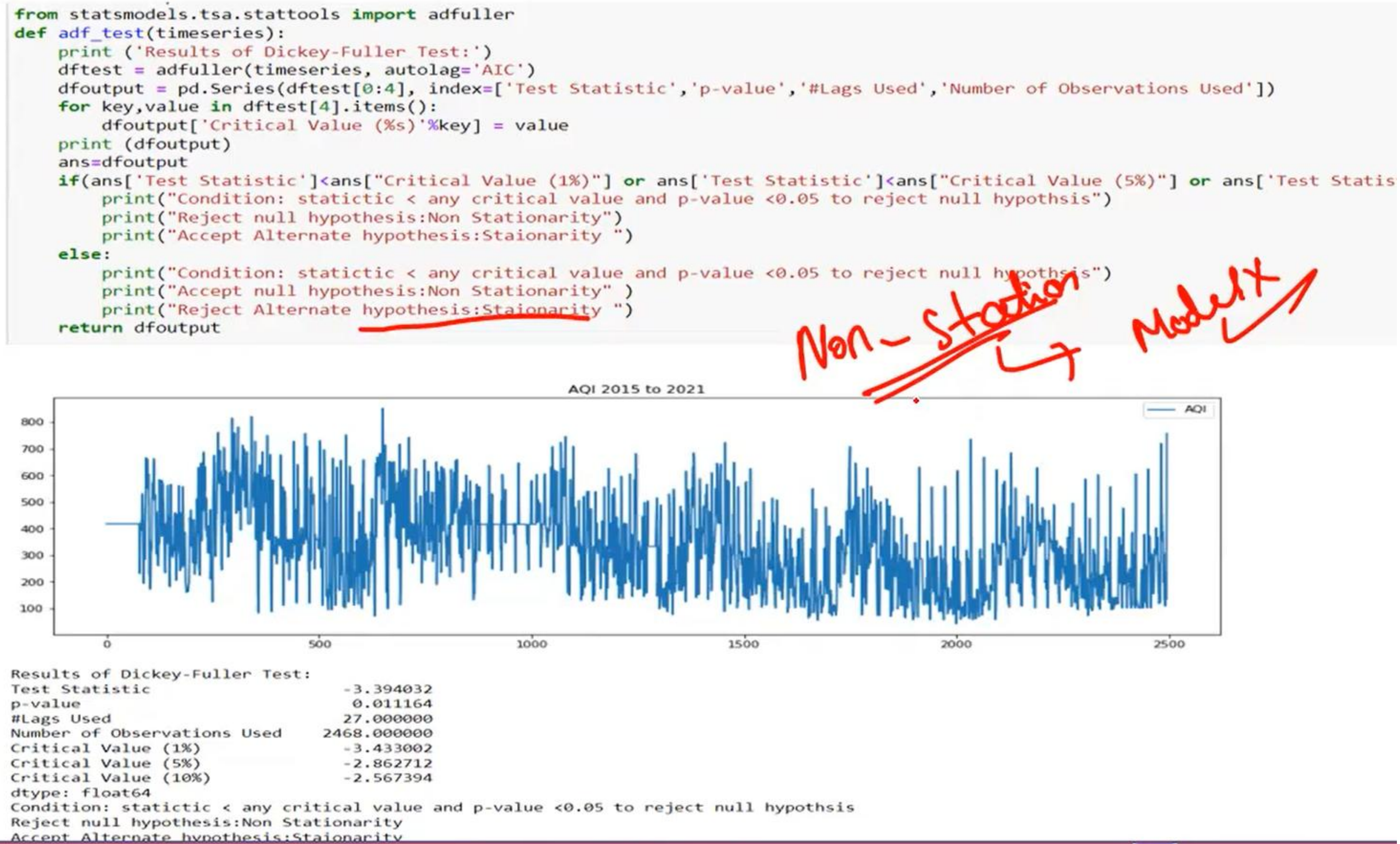
ADF test is conducted with the following assumptions.

Null Hypothesis (HO): Series is non-stationary or series has a unit root.

Alternate Hypothesis(HA): Series is stationary or series has no unit root.

If the null hypothesis is failed to be rejected, this test may provide evidence that the series is non-stationary.
Conditions to Reject Null Hypothesis(HO)

If Test statistic < Critical Value and p-value < 0.05 – Reject Null Hypothesis(HO) i.e., time series does not have a unit root, meaning it is stationary. It does not have a time-dependent structure.



```
from statsmodels.tsa.stattools import adfuller
def adf_test(timeseries):
    print ('Results of Dickey-Fuller Test:')
    dftest = adfuller(timeseries, autolag='AIC')
    dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags Used','Number of Observations Used'])
    for key,value in dftest[4].items():
        dfoutput['Critical Value (%s)'%key] = value
    print (dfoutput)
    ans=dfoutput
    if(ans['Test Statistic']<ans["Critical Value (1%)"] or ans['Test Statistic']<ans["Critical Value (5%)"] or ans['Test Statis
        print("Condition: statictic < any critical value and p-value <0.05 to reject null hypothsis")
        print("Reject null hypothesis:Non Stationarity")
        print("Accept Alternate hypothesis:Staionarity ")
    else:
        print("Condition: statictic < any critical value and p-value <0.05 to reject null hypothsis")
        print("Accept null hypothesis:Non Stationarity" )
        print("Reject Alternate hypothesis:Staionarity ")
    return dfoutput
```

Non - Station ↳ Model ✗

```
Results of Dickey-Fuller Test:
Test Statistic                   -3.394032
p-value                           0.011164
#Lags Used                       27.000000
Number of Observations Used    2468.000000
Critical Value (1%)              -3.433002
Critical Value (5%)              -2.862712
Critical Value (10%)             -2.567394
dtype: float64
Condition: statictic < any critical value and p-value <0.05 to reject null hypothsis
Reject null hypothesis:Non Stationarity
Accept Alternate hypothesis:Staionarity
```

Non stationary to stationary

- Difference
- Log

Time Series Analysis

Lag

t-1

t-1

Date	Value	Value$_{t-1}$	Value$_{t-2}$
1/1/2017	200	NA	NA
1/2/2017	220	200	NA
1/3/2017	215	220	200
1/4/2017	230	215	220
1/5/2017	235	230	215
1/6/2017	225	235	230
1/7/2017	220	225	235
1/8/2017	225	220	225
1/9/2017	240	225	220
1/10/2017	245	240	225

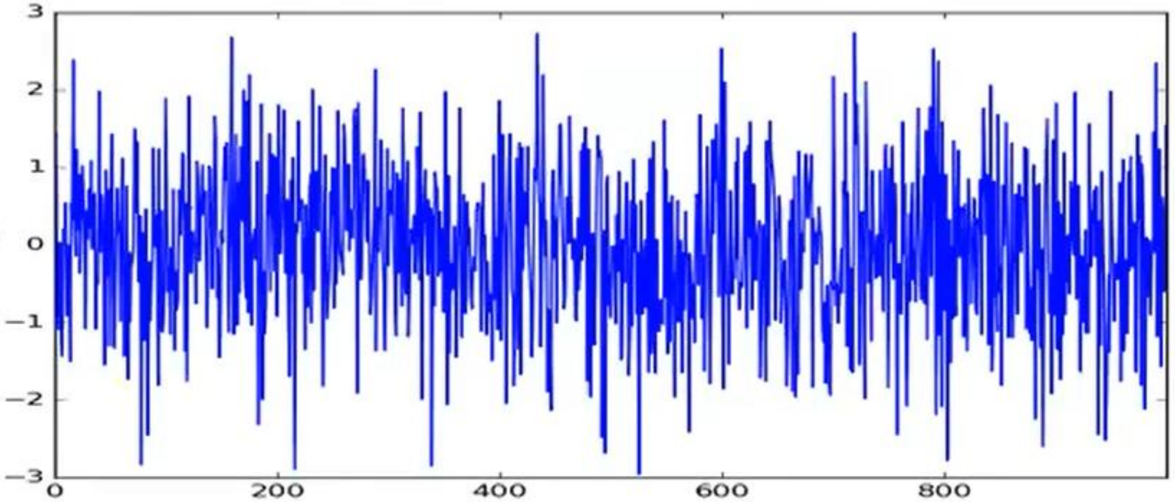
Time Series Analysis

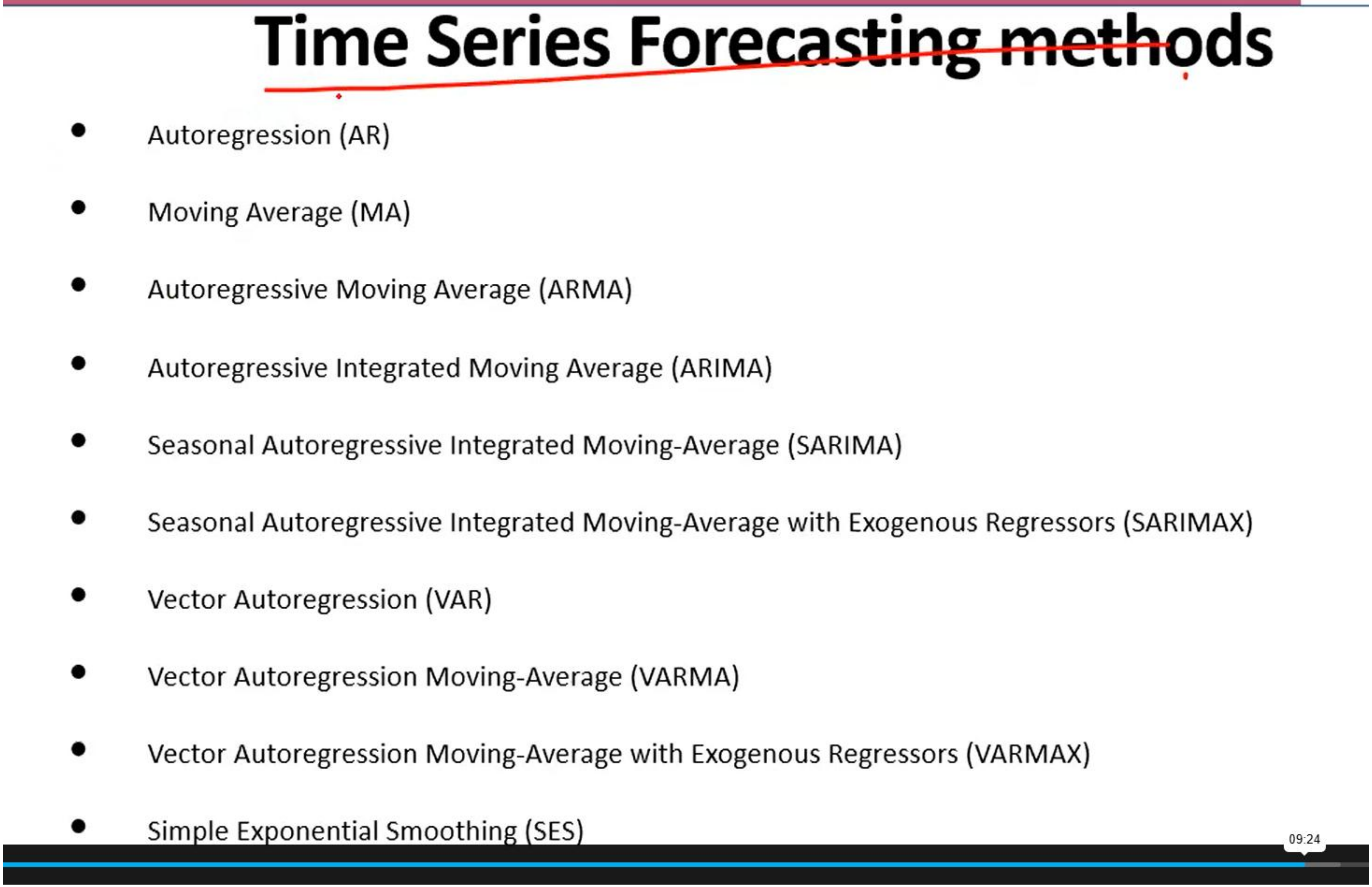
What is White Noise?

White Noise

- No pattern (does not use for forecasting)

- Completely random with





Time Series Forecasting methods

- Autoregression (AR)
- Moving Average (MA)
- Autoregressive Moving Average (ARMA)
- Autoregressive Integrated Moving Average (ARIMA)
- Seasonal Autoregressive Integrated Moving-Average (SARIMA)
- Seasonal Autoregressive Integrated Moving-Average with Exogenous Regressors (SARIMAX)
- Vector Autoregression (VAR)
- Vector Autoregression Moving-Average (VARMA)
- Vector Autoregression Moving-Average with Exogenous Regressors (VARMAX)
- Simple Exponential Smoothing (SES)

Auto Regression (AR)

Lag

The term *Auto R*egression indicates that it is a regression of the variable against itself.

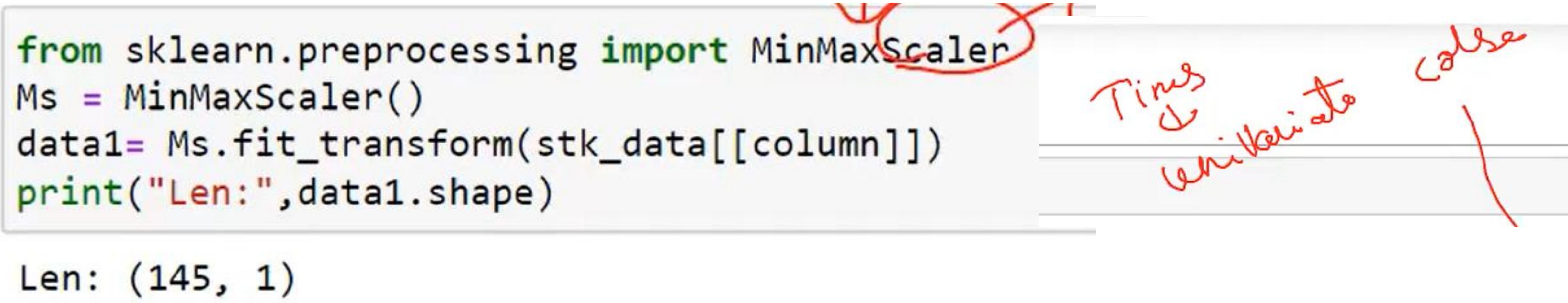
Autoregressive Model: AR(p)

- The autoregressive model uses observations from previous time steps as input to a regression equations to predict the value at the next step.
- The order, p, of the autoregressive model can be determined by looking at the **partial autocorrelation function (PACF)**. The PACF gives the partial correlation of a stationary time series with its own lagged values, regressed of the time series at all shorter lags.

lag

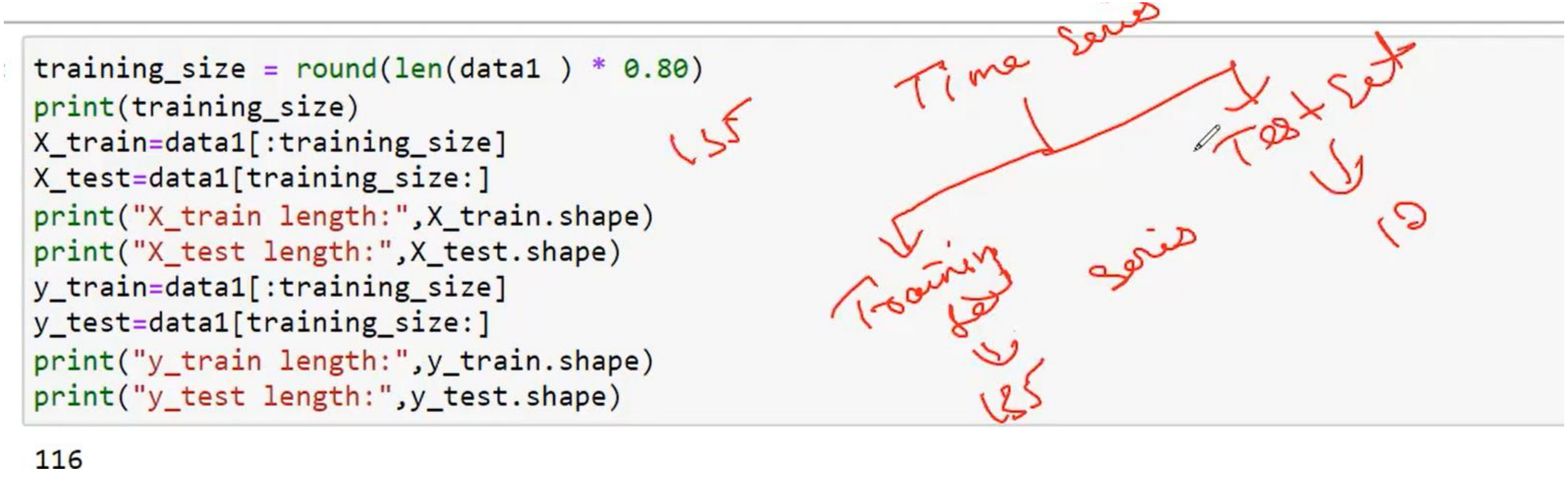
Time	y_{t-1}	y_{t-2}
	0	00
1	1	1
2	2	2
3	3	
4	4	
5	5	
6	6	
7	7	

02:13



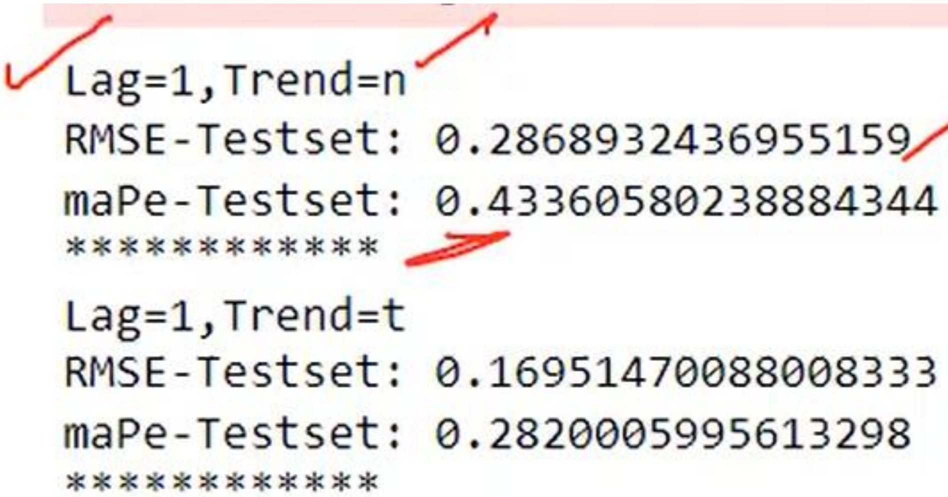
```
from sklearn.preprocessing import MinMaxScaler
Ms = MinMaxScaler()
data1= Ms.fit_transform(stk_data[[column]])
print("Len:",data1.shape)
```

```
Len: (145, 1)
```



```
training_size = round(len(data1 ) * 0.80)
print(training_size)
X_train=data1[:training_size]
X_test=data1[training_size:]
print("X_train length:",X_train.shape)
print("X_test length:",X_test.shape)
y_train=data1[:training_size]
y_test=data1[training_size:]
print("y_train length:",y_train.shape)
print("y_test length:",y_test.shape)
```

```
116
```



```
Lag=1,Trend=n
RMSE-Testset: 0.2868932436955159
maPe-Testset: 0.43360580238884344
************
Lag=1,Trend=t
RMSE-Testset: 0.16951470088008333
maPe-Testset: 0.2820005995613298
************
```

Time Ser

player.vimeo.com – To exit full screen, press Esc

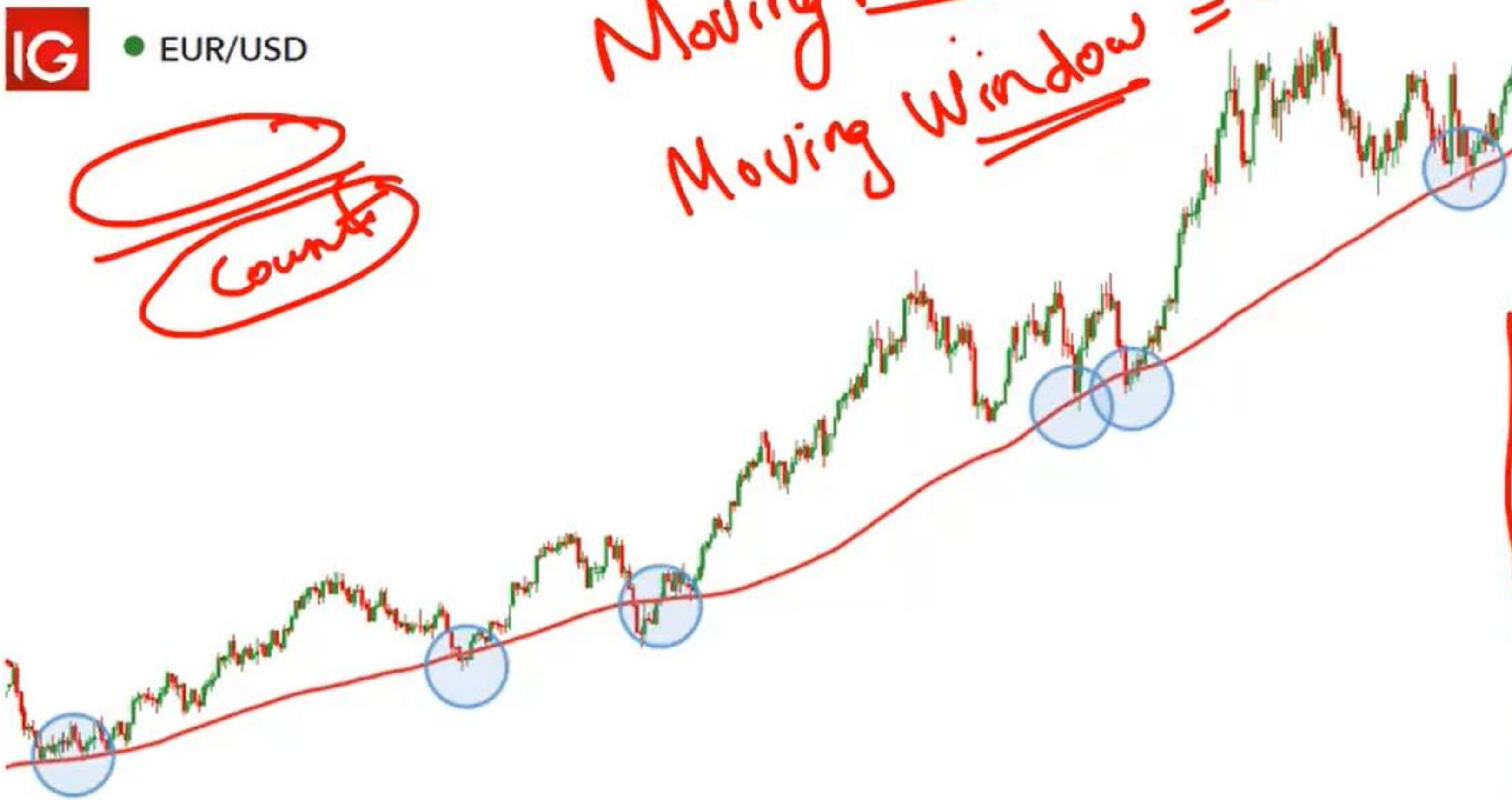
Moving Average(MA) →

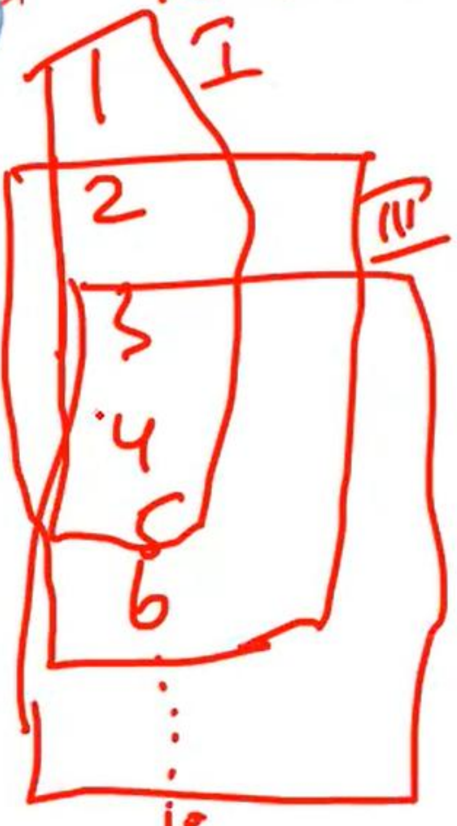
Moving Average

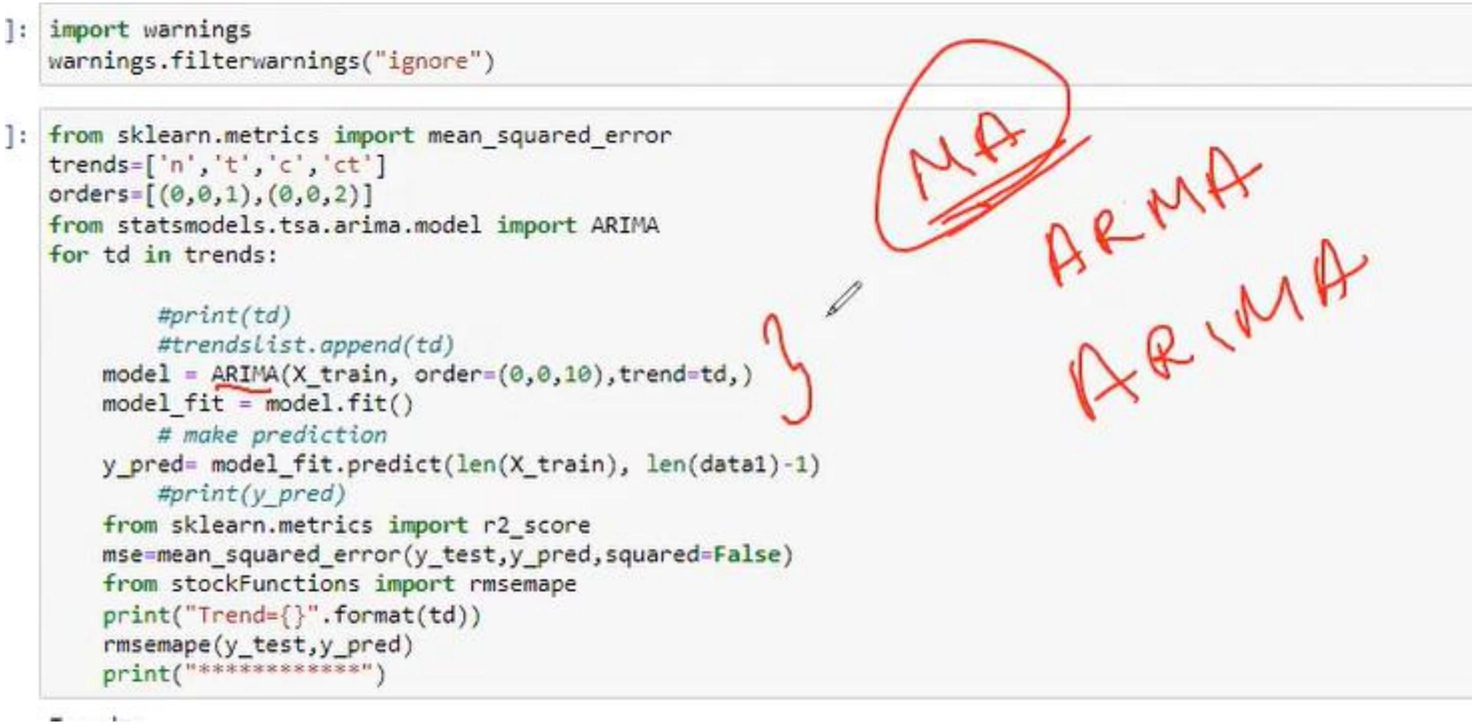
Moving Window = 5

100 Entries

10 → Aver







```
]: import warnings
   warnings.filterwarnings("ignore")
```

```
]: from sklearn.metrics import mean_squared_error
   trends=['n','t','c','ct']
   orders=[(0,0,1),(0,0,2)]
   from statsmodels.tsa.arima.model import ARIMA
   for td in trends:

           #print(td)
           #trendslist.append(td)
       model = ARIMA(X_train, order=(0,0,10),trend=td,)
       model_fit = model.fit()
           # make prediction
       y_pred= model_fit.predict(len(X_train), len(data1)-1)
           #print(y_pred)
       from sklearn.metrics import r2_score
       mse=mean_squared_error(y_test,y_pred,squared=False)
       from stockFunctions import rmsemape
       print("Trend={}".format(td))
       rmsemape(y_test,y_pred)
       print("************")
```

MA

ARMA

ARIMA

Autoregressive Integrated Moving Average (ARIMA)

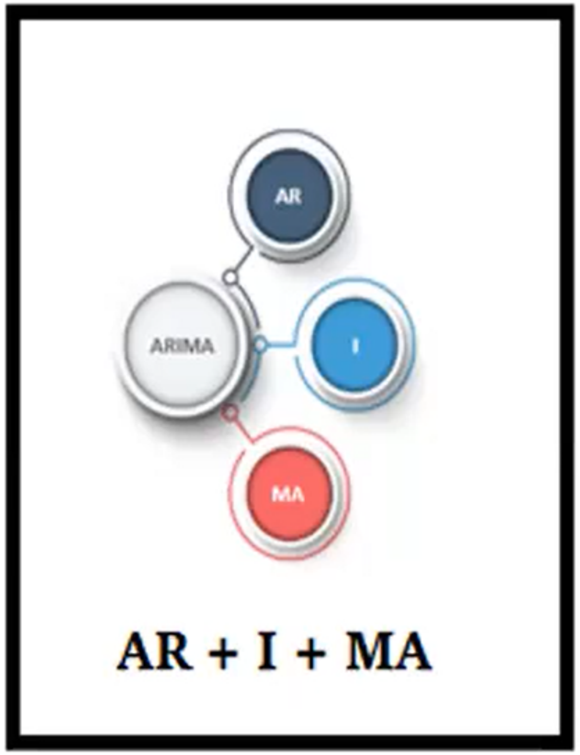
Log

difference

MA

Non stationarity

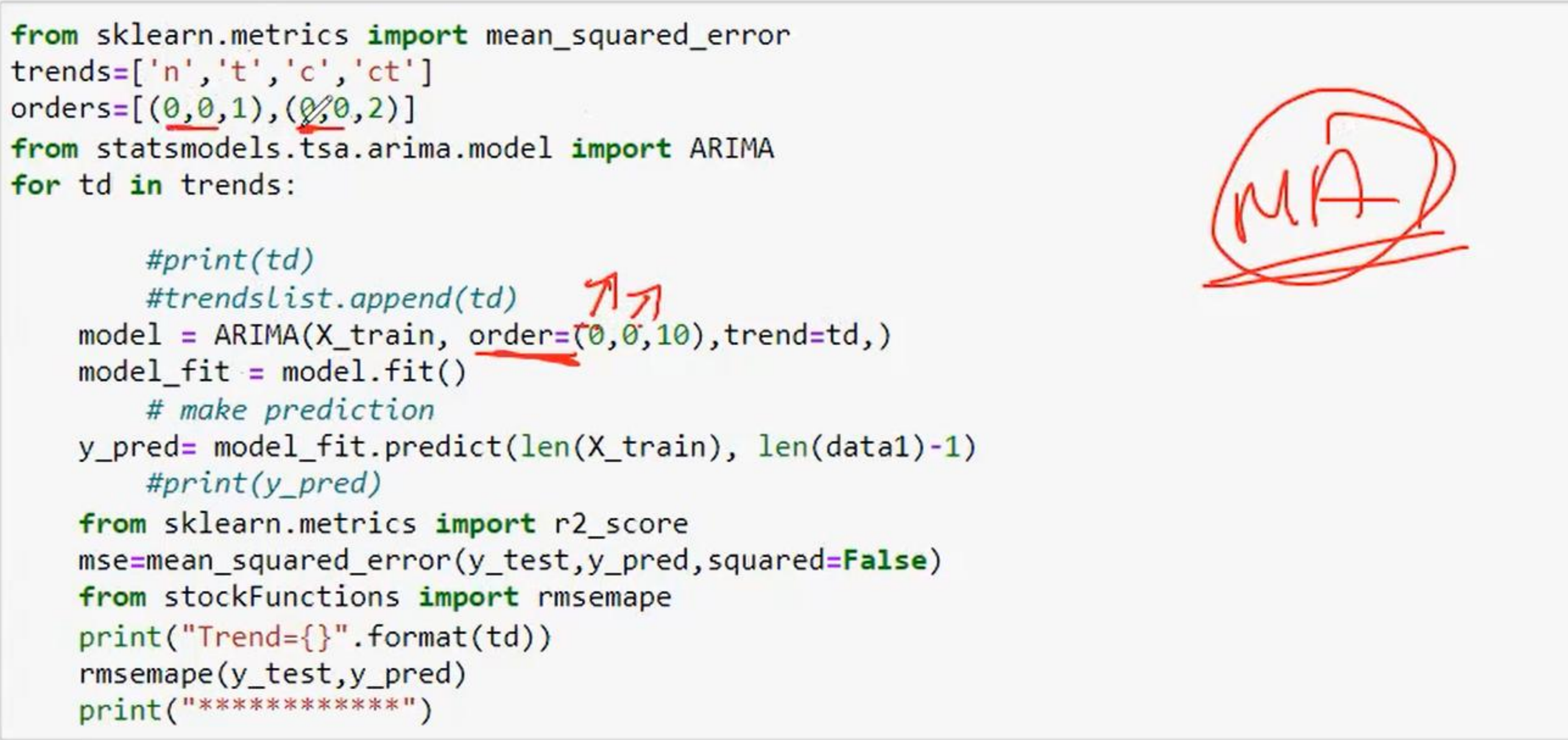
AR+I+MA= ARIMA



AR(Autoregression) - Uses the past values to predict the future.

I(Integrated) - The use of differencing of raw observations in order to make the time series stationary.

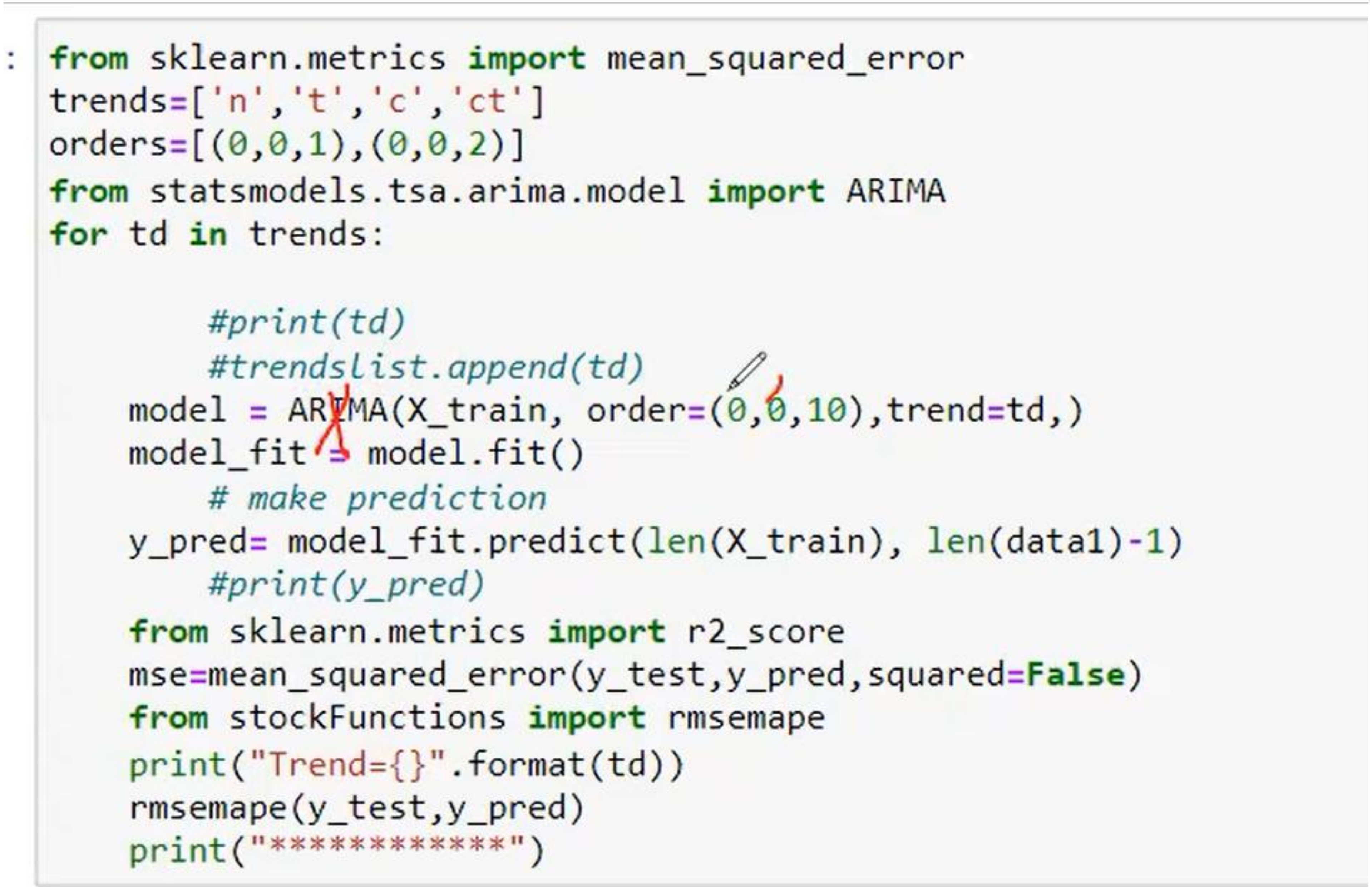
MA(Moving Average) - Uses the past error terms in the given series to predict the future



```
from sklearn.metrics import mean_squared_error
trends=['n','t','c','ct']
orders=[(0,0,1),(0,0,2)]
from statsmodels.tsa.arima.model import ARIMA
for td in trends:

        #print(td)
        #trendslist.append(td)
    model = ARIMA(X_train, order=(0,0,10),trend=td,)
    model_fit = model.fit()
        # make prediction
    y_pred= model_fit.predict(len(X_train), len(data1)-1)
        #print(y_pred)
    from sklearn.metrics import r2_score
    mse=mean_squared_error(y_test,y_pred,squared=False)
    from stockFunctions import rmsemape
    print("Trend={}".format(td))
    rmsemape(y_test,y_pred)
    print("************")
```

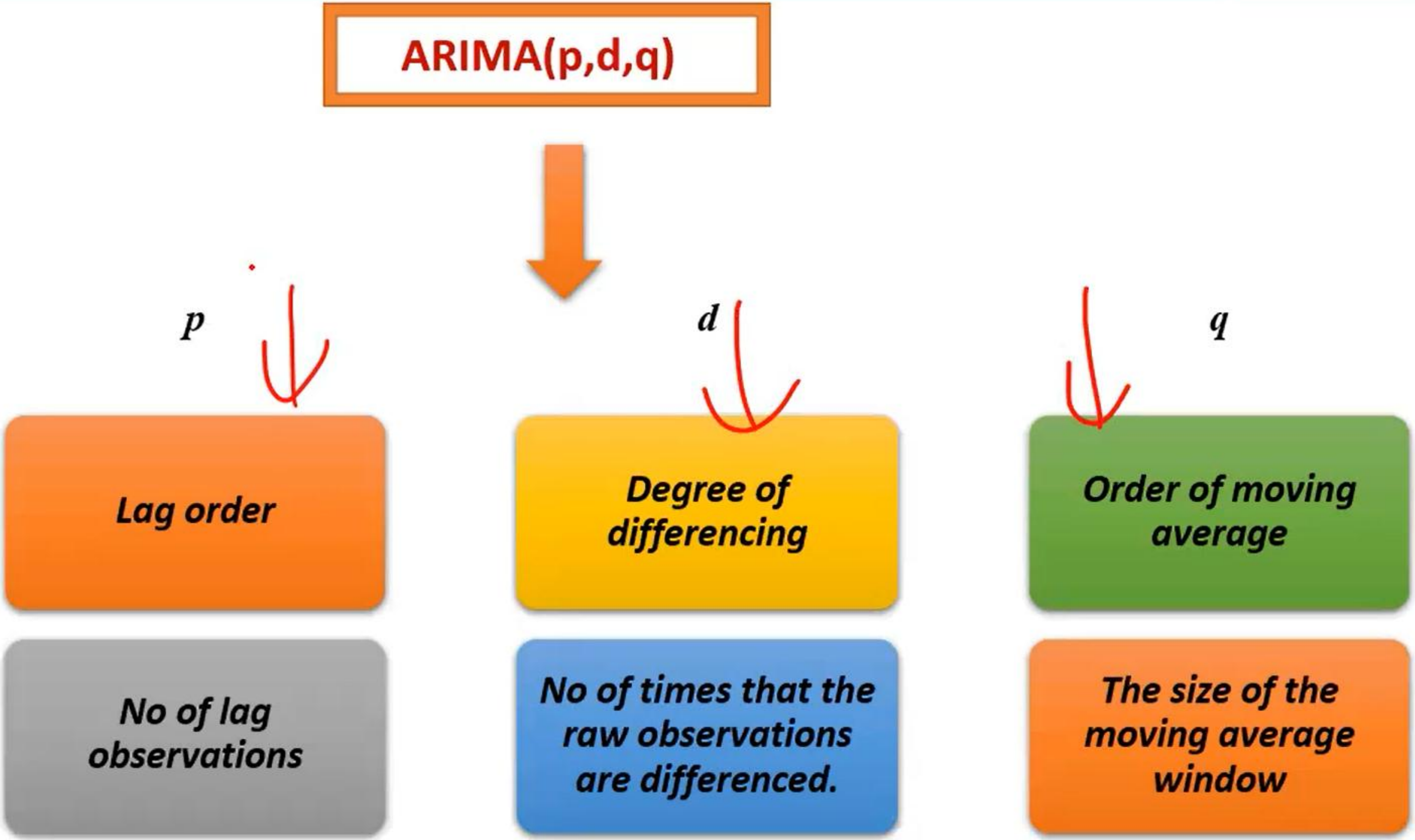
MA



```
from sklearn.metrics import mean_squared_error
trends=['n','t','c','ct']
orders=[(0,0,1),(0,0,2)]
from statsmodels.tsa.arima.model import ARIMA
for td in trends:

        #print(td)
        #trendslist.append(td)
    model = ARIMA(X_train, order=(0,0,10),trend=td,)
    model_fit = model.fit()
        # make prediction
    y_pred= model_fit.predict(len(X_train), len(data1)-1)
        #print(y_pred)
    from sklearn.metrics import r2_score
    mse=mean_squared_error(y_test,y_pred,squared=False)
    from stockFunctions import rmsemape
    print("Trend={}".format(td))
    rmsemape(y_test,y_pred)
    print("************")
```

Time Series Analysis



ARIMA(p,d,q)

p

Lag order

No of lag observations

d

Degree of differencing

No of times that the raw observations are differenced.

q

Order of moving average

The size of the moving average window

Seasonal Autoregressive Integrated Moving-Average (SARIMA)

Seasonal ARIMA, is an extension of ARIMA that explicitly supports univariate time series data with a seasonal component.

SARIMA(p,d,q)(P,D,Q)m

P: Seasonal autoregressive order.

D: Seasonal difference order.

Q: Seasonal moving average order.

m: The number of time steps for a single seasonal period.

Seasonal Autoregressive Integrated Moving-Average with Exogenous Regressors (SARIMAX)

SARIMAX is an updated version of the ARIMA model. It includes seasonal effects and eXogenous factors with the autoregressive and moving average component in the model.

Eg: In farming, production forecast is based on factors like area of land, quality of seed etc. but in this rainfall is an exogenous variable whose value will be determined by other factors like humidity, air speed etc.

X – Exogenous variable(external factor)

Multi Variate

Vector Auto-Regression (VAR)

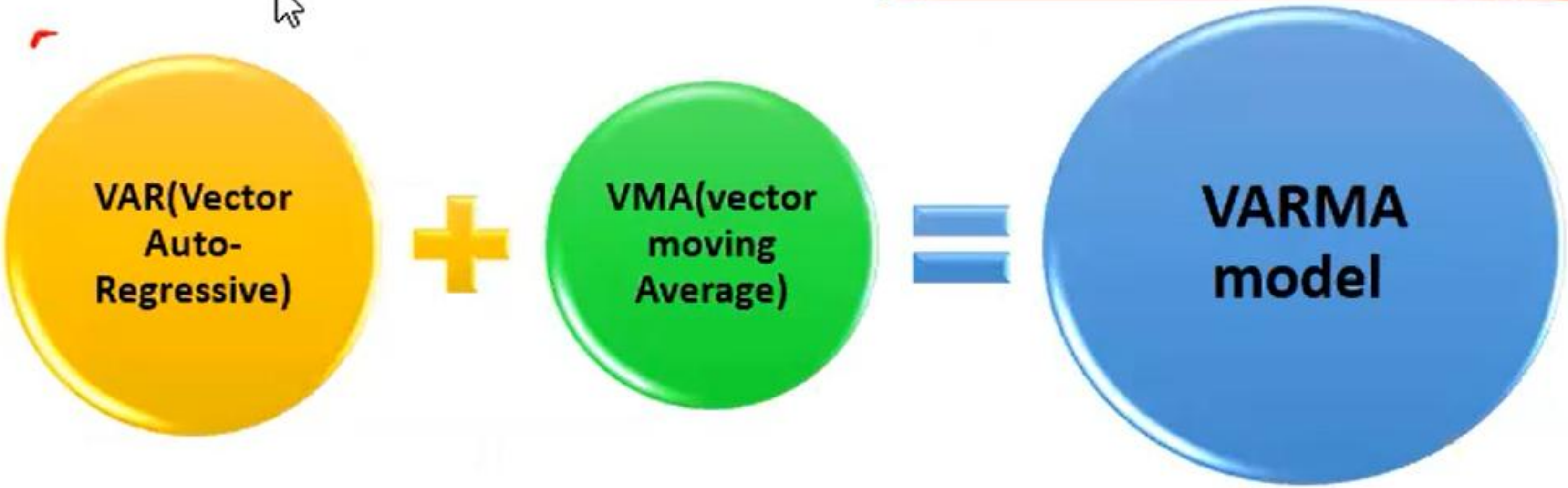
□ Vector Autoregression (**VAR**) is a **multivariate** forecasting algorithm that is used when two or more time series influence each other.

□ In the VAR model, each variable is modeled as a **linear combination of past values of itself and the past values of other variables in the system**.

VAR different from other Autoregressive models like AR, ARMA or ARIMA?

The primary difference is those models are **uni-directional**. Whereas, Vector Auto Regression (VAR) is **bi-directional**. That is, the variables influence each other.

Vector Auto Regression Moving-Average (VARMA)



VAR(Vector Auto-Regressive)

VMA(vector moving Average)

VARMA model

VARMA model where both lag order and order of moving average (p and q) is used in the modelling and also we can convert the VARMA model in VAR by setting the moving average as zero and in VMA by setting the lag order as zero in the modelling.

NOTE: It is used for multivariate time series forecasting.