

1 데이터분석 단계(Data Analysis Cycle)

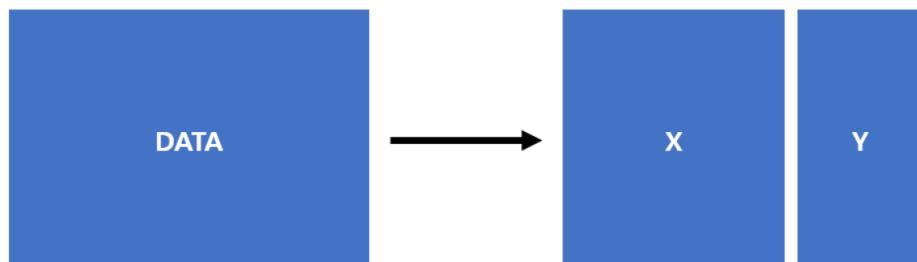
[Open in Colab](#)

✓ 데이터 전처리: (0) 쓸모 없을 뻔한 Raw를 쓸모 있는 Data로 변환

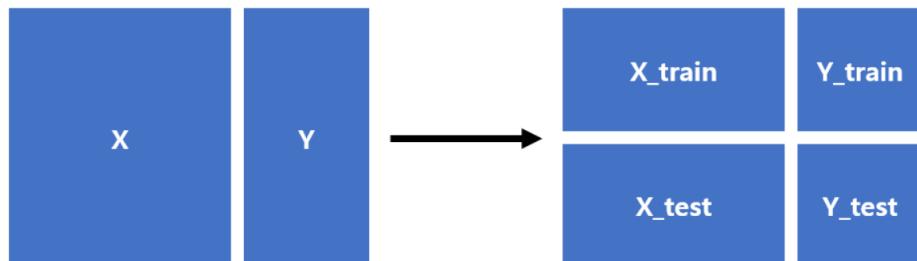
100	T50	횟수	111	TPU	...
few	Gds	Hvi	Rew	Fa	...
Fre	CT	QTP	D	합	...
'1'	1	23	22	NaN	43
76	NaN	43	32	1	8
'Hi'	NaN	NaN	NaN	NaN	87
23	98	NaN	64	46	NaN
c	90	'WW'	24	'KK'	4
t	NaN	2	NaN	NaN	6
64	NaN	90	'IU'	4	76

번호	시간	총량	기간	누적	...
1	1	23	22	21	43
76	33.3	43	32	1	8
5	33.3	52	35	21	87
23	98	52	64	46	61
90	33.3	2	24	33	4
55	2	52	35	21	6
64	33.3	90	11	4	76

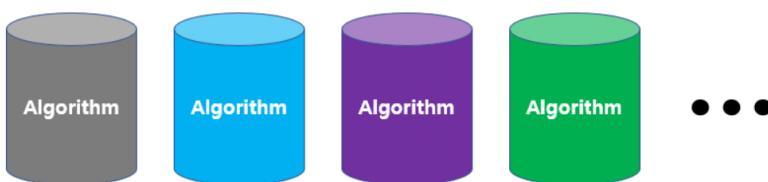
✓ 데이터 분할: (1) 목표/종속변수 Y와 설명/독립변수 X설정



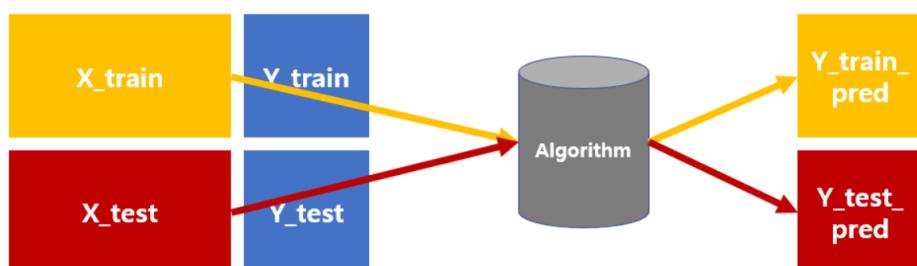
✓ 데이터 분할: (2) 학습데이터 Train과 예측 데이터 Test로 분할



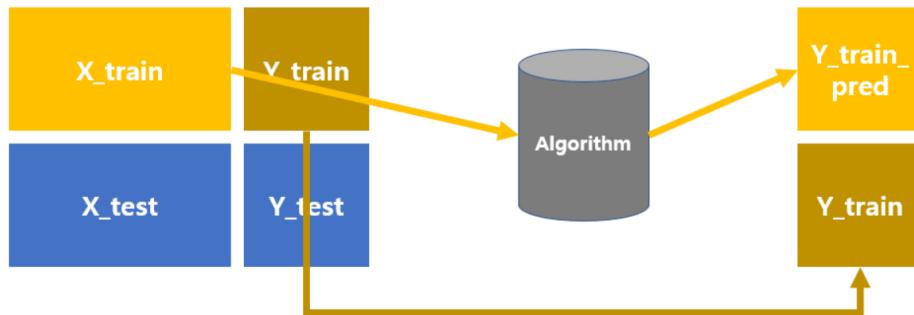
✓ 모델링: (3) 분석 목적에 맞는 알고리즘(Base & Advanced) 후보들 준비



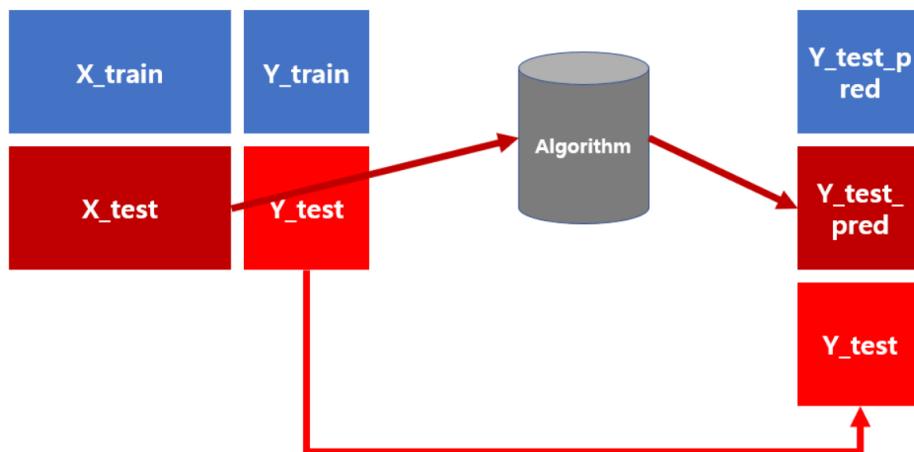
✓ 모델링 & 학습: (4) 알고리즘 평가를 위해 Train/Test의 예측값 추정



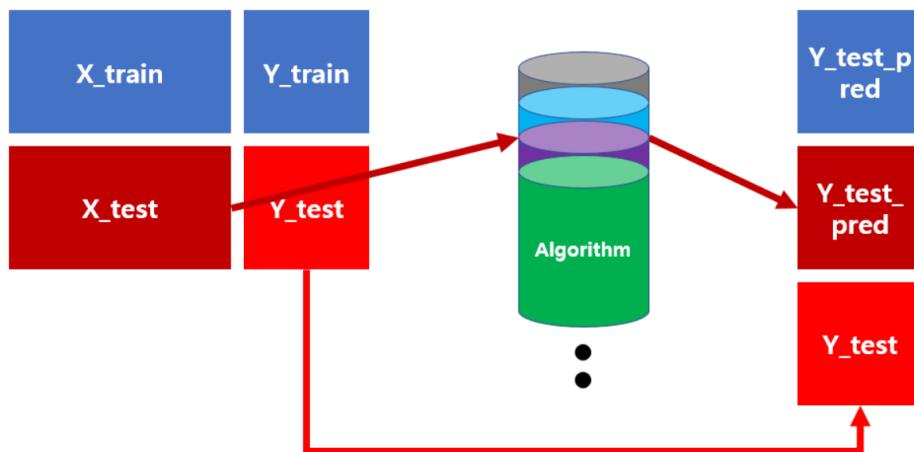
✓ 평가: (5) 학습(Train)이 잘 되었는지 알고리즘 성능검증



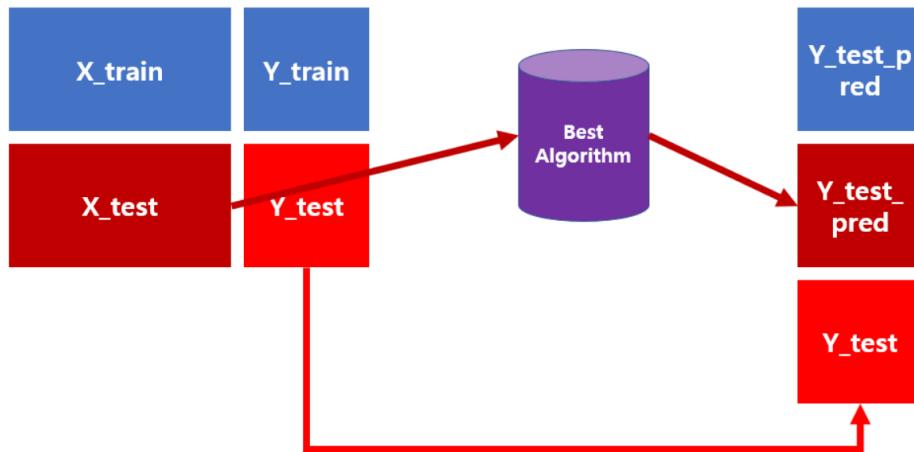
✓ 평가: (6) 예측(Test)이 잘 되었는지 알고리즘 성능검증

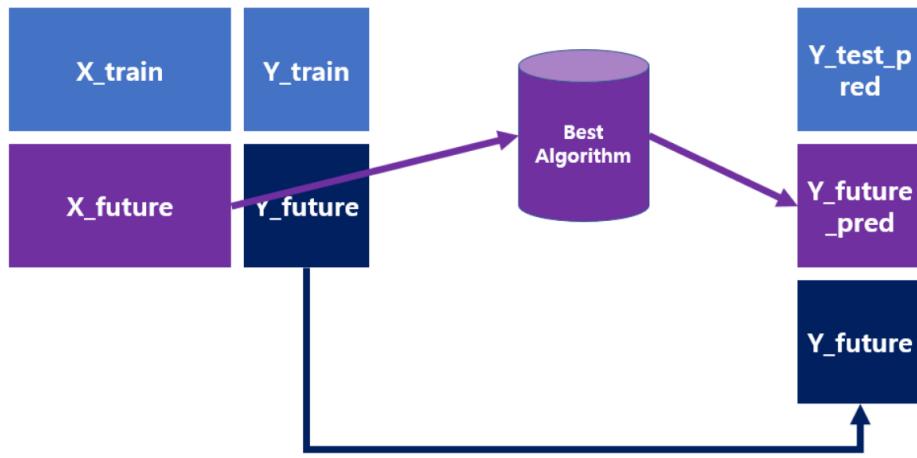


✓ 최적 알고리즘 선택: (7) 알고리즘을 변경하여 위 과정 반복 후



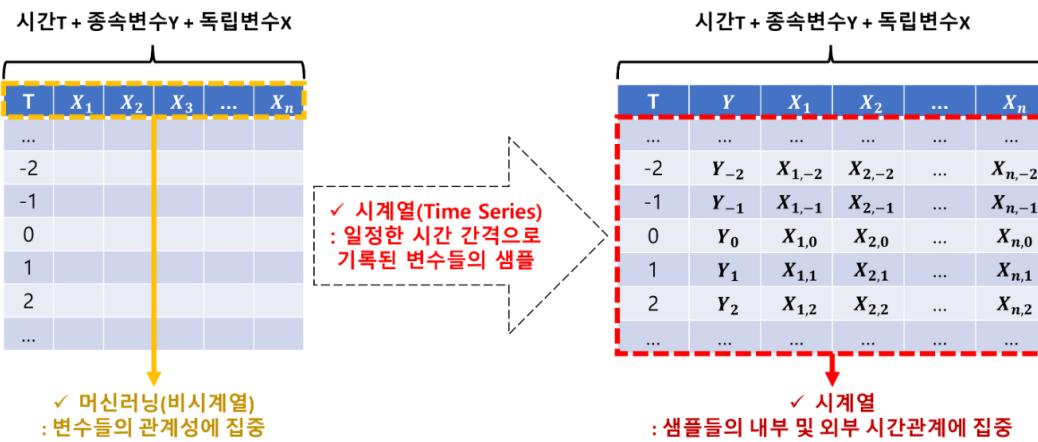
✓ 최적 알고리즘 선택: (7) 최고 성능의 알고리즘 선택





2 시계열분석과 기계학습의 차이(Comparison)

2.1 확률적 프로세스 및 시계열 데이터



- 확률적 프로세스(Stochastic Process): 시간에 따라 확률적 수치의 변화를 가지는 변수

- 변수:

$$X = \{X_1, X_2, \dots, X_n\}$$

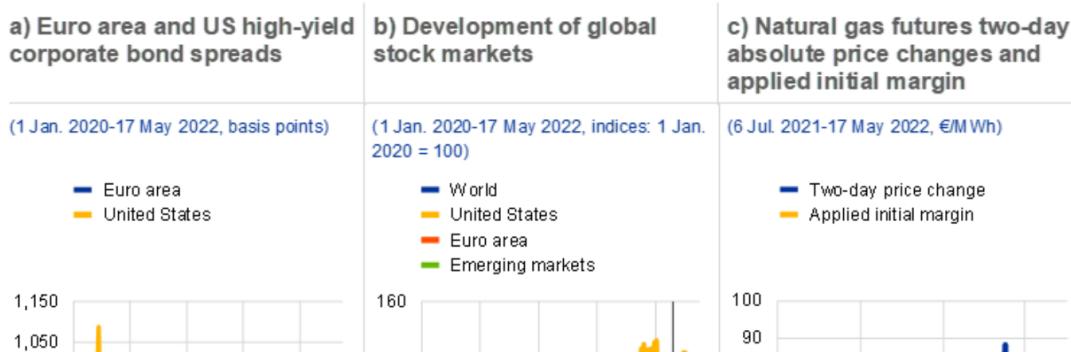
- 확률적 프로세스: 각 변수 내 확률적 수치의 변화

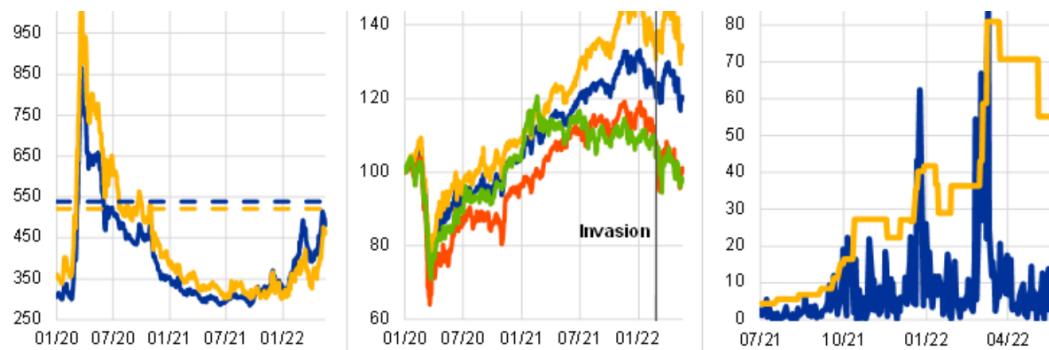
$$\begin{aligned} Y &= \{\dots, Y_{-2}, Y_{-1}, Y_0, Y_1, Y_2, \dots\} \\ X_1 &= \{\dots, X_{1,-2}, X_{1,-1}, X_{1,0}, X_{1,1}, X_{1,2}, \dots\} \\ X_2 &= \{\dots, X_{2,-2}, X_{2,-1}, X_{2,0}, X_{2,1}, X_{2,2}, \dots\} \end{aligned}$$

- 종속변수(Y_t) 또는 독립변수($X_{1,t}$)가 시간 단위 (t)를 포함
- 모델링의 출력(Output)은 변수 Y 의 특정 시간 t 에서의 예측값 (\hat{Y}_t)

- 시계열 데이터(Time Series Data): 일정한 시간 간격에 따라 순차적(Sequentially)으로 기록된 확률적 프로세스 또는 시간변화 데이터

- 시간의 흐름에 따라 불규칙적 변동을 분석하기 위해 필수적 데이터
- 과거가 미래에 어떤 영향을 주는지 분석을 통해 예측 가능
- 최근 기계학습과 딥러닝을 사용하여 복잡한 데이터 예측
- 시계열 예측과 기계학습/딥러닝 간 전처리와 알고리즘 방식 차이 때문에, 별도로 (1) 시계열 데이터 분석 단계 이해, (2) 시계열 데이터의 전처리, (3) 시계열 알고리즘 이해 필수!





a) Nominal price growth in prime commercial real estate

(Q1 2005-Q4 2021, percentages)

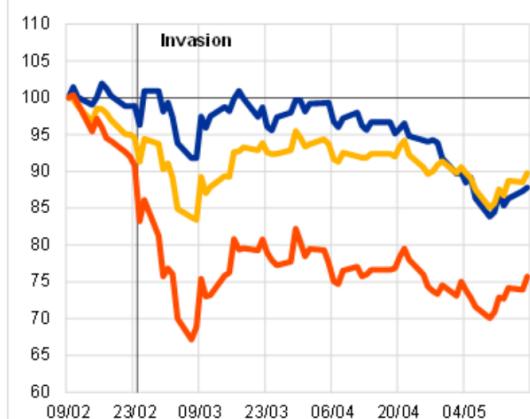
- Prime commercial real estate – retail
- Prime commercial real estate – office



b) Euro area REITs versus broader stock market

(9 Feb.-17 May 2022, indices: 9 Feb. 2022 = 100)

- Euro area REITs
- EURO STOXX
- EURO STOXX Banks

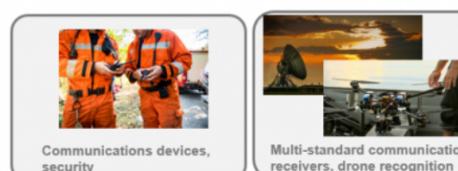


(<https://www.ecb.europa.eu/pub/financial-stability/fsr/html/ecb.fsr202205-f207f46ea0.en.html#toc6>)

2.2 시계열 데이터의 활용

- 관측된 시계열 데이터를 분석하여 미래를 예측하는 문제가 바로 시계열 예측 문제
- 흔하게 접하는 문제로 주로 경제 지표를 예측하거나 시간에 따른 정당지지율을 예측하거나, 어떤 상품의 수요를 예측하는 문제에 이르기까지 다양하게 활용되고 있으며 현실에 가까울수록 항상 고려해야 하는 문제

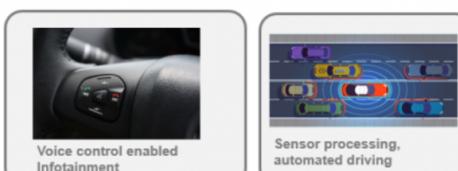
Aerospace, Defense and Communications



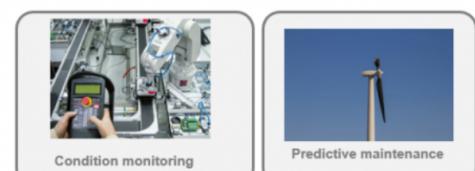
Consumer Electronics and Digital Health



Automotive



Industrial Automation



(Deep Learning for Signal Processing Applications, Mathworks)

- 예측된 결과를 바탕으로 여러 정책이나 비즈니스 전략을 결정하는 과정에 활용 되기에, 실제 비즈니스 영역에서는 시계열 예측 문제가 매우 중요

Use case mapping to data types

Number of use cases Low High

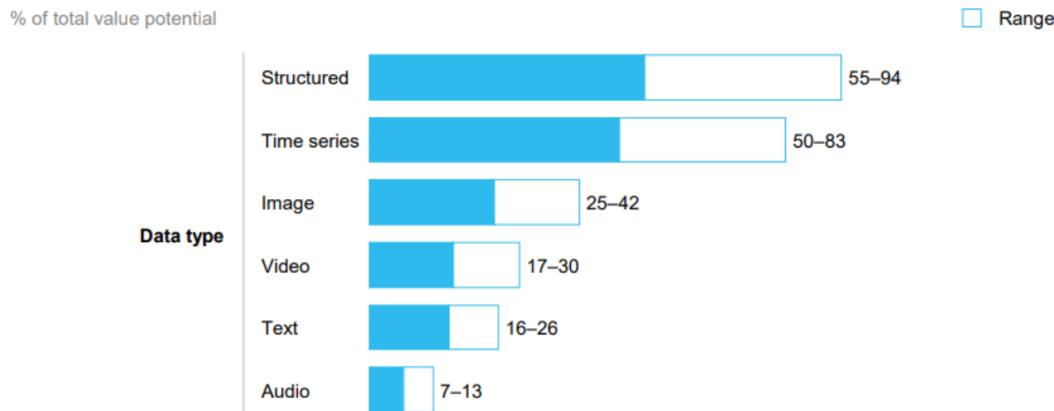
Use case domains	Structured/ semi- structured	Time series	Text	Audio	Video	Image
Analytics-driven accounting and IT						

Analytics-driven hiring and retention						
Channel management						
Churn reduction						
Customer acquisition/lead generation						
Customer service management						
Fraud and debt analytics						
Inventory and parts optimization						
Logistics network and warehouse optimization						
Marketing budget allocation						
Next product to buy/individualized offering						
Predictive maintenance						
Predictive service/intervention						
Pricing and promotion						
Procurement and spend analytics						
Product development cycle optimization						
Product feature optimization						
Risk modeling						
Sales and demand forecasting						
Smart capital expenditures						
Task automation						
Workforce productivity and efficiency						
Yield optimization						

SOURCE: McKinsey Global Institute analysis

- McKinsey Global Institute에 따르면, 시계열 데이터가 텍스트나 이미지 데이터 보다 더 큰 잠재적 가치를 가지고 있다고 보고 있음

Range of potential AI value impact by data type



SOURCE: McKinsey Global Institute analysis

2.3 데이터분석 단계의 변화

"시계열예측과 기계학습/딥러닝 간 전처리와 알고리즘 방식 차이 때문에, 별도로 (1) 시계열 데이터분석 단계 이해, (2) 시계열 데이터의 전처리, (3) 시계열 알고리즘 이해 필수!"

✓ 데이터 전처리: (0) 쓸모 없을 뻔한 Raw를 쓸모 있는 Data로 변환

100	T50	횟수	111	TPU	...						
few	Gds	Hvi	Rew	Fa	...	Fre	CT	QTP	D	합	...
'1'	1	23	22	NaN	43						
76	NaN	43	32	1	8						
'Hi'	NaN	NaN	NaN	NaN	87						
23	98	NaN	64	46	NaN						
c	90	NaN	'WW'	24	'KK'	4					
t	NaN	2	NaN	NaN	NaN	6					

→

번호	시간	총량	기간	누적	...
1	1	23	22	21	43
76	33.3	43	32	1	8
5	33.3	52	35	21	87
23	98	52	64	46	61
90	33.3	2	24	33	4
55	2	52	35	21	6

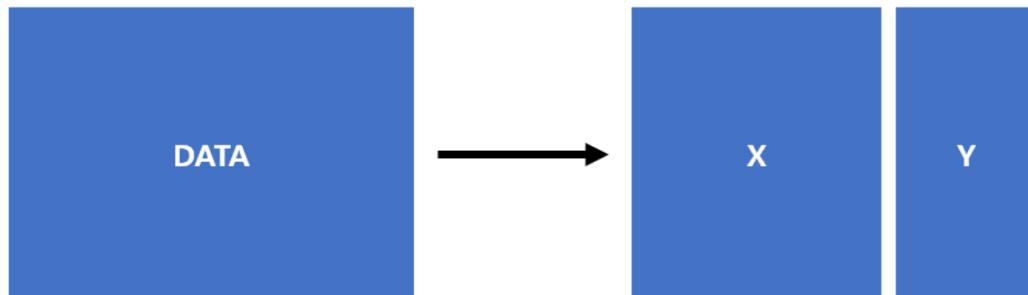
64	NaN	90	'IU'	4	76
----	-----	----	------	---	----

64	33.3	90	11	4	76
----	------	----	----	---	----

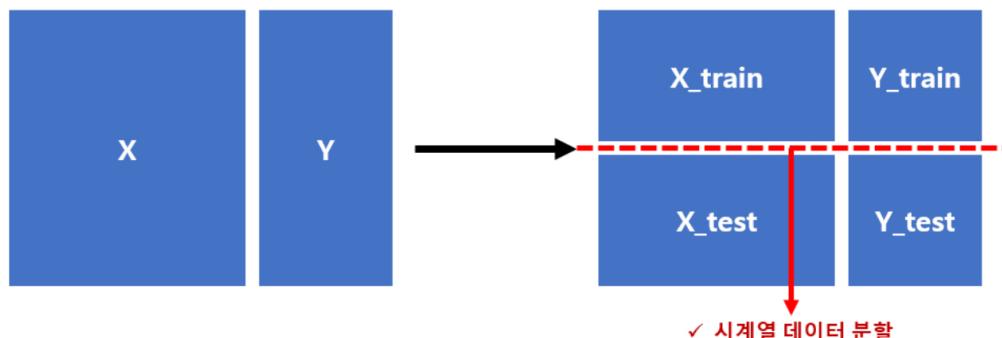
✓ 시계열 전처리

- 빈도, 추세, 계절성, 주기, 자연값, 시간정보 등 알고리즘이 이해하는 값으로 변경 포함 (여기까진 일반적 전처리)
- 시간 흐름에 따른 “수치변화”와 비어 있는 “시간간격”까지 전처리

✓ 데이터 분할: (1) 목표/종속변수 Y와 설명/독립변수 X 설정



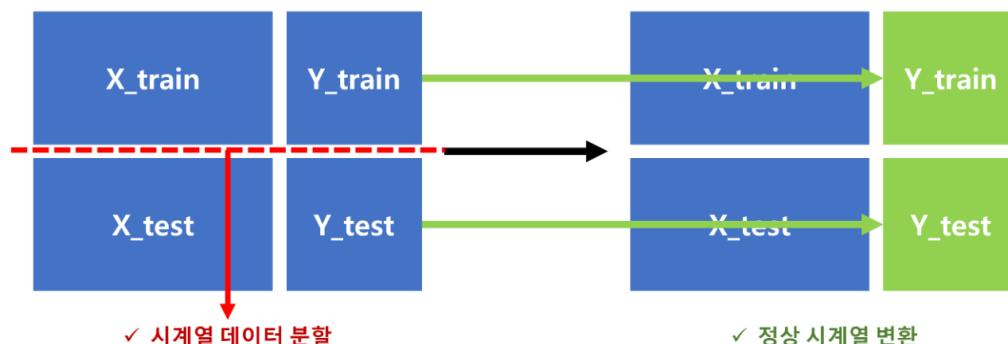
✓ 데이터 분할: (2) 학습데이터 Train과 예측 데이터 Test로 분할



✓ 시계열 데이터 분할

- (1) 절대로 “랜덤” 분할이 아닌 “시간축 유지”
- (2) 새로운 미래 시점마다 Train/Test “재분류”
- (3) 단기/중기/장기 시점에 따라 “별도 모델링” 해야 하기 때문에 Test “추가 재분류”

✓ 시계열 데이터 전처리: 머신러닝과 달리 비정상 시계열(Y)을 정상 시계열로 변환



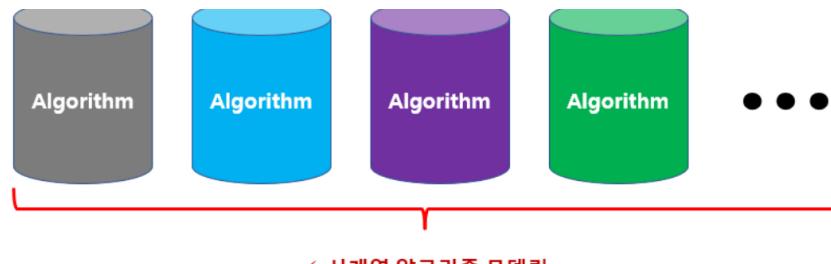
✓ 정상 시계열 변환

- (1) 절대로 “랜덤” 분할이 아닌 “시간축 유지”
- (2) 새로운 미래 시점마다 Train/Test “재분류”
- (3) 단기/중기/장기 시점에 따라 “별도 모델링” 해야 하기 때문에 Test “추가 재분류”

- (1) 필수는 아니지만 범위제한으로 예측성능↑
- (2) 모델 복잡도 낮아져서 Bias↑ + Variance↓ => 과적합↓ + 예측성능↑
- (2) 통계추론 알고리즘은 대부분 학습위해 필수

✓ 모델링: (3) 분석 목적에 맞는 알고리즘(Base & Advanced) 후보들 준비





✓ 시계열 알고리즘 모델링

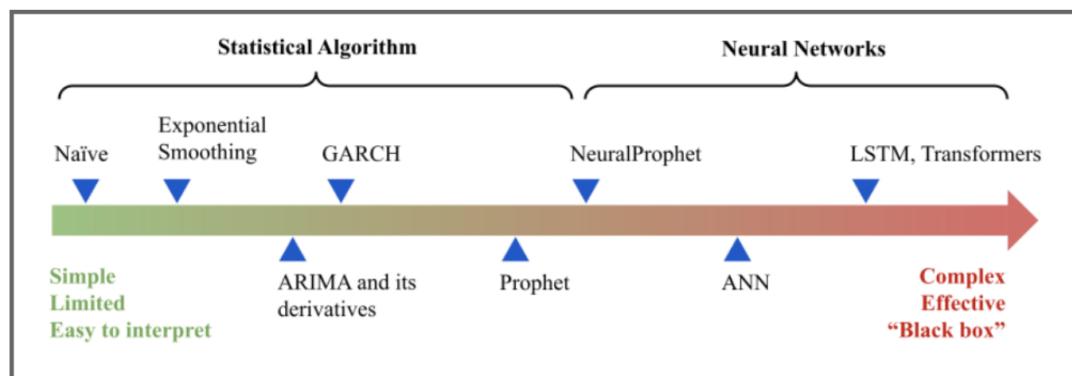
- : (1) “시간 자체를 모델링” 해야 하기 때문에
기계학습 접근과 별도의 접근 필요
- (2) 단기/중기/장기 시점에 따라 “별도 모델링” 필요

2.4 시계열 알고리즘 종류

“통계추론, 기계학습 및 딥러닝의 흐름에 시간패턴을 반영하려 진화”

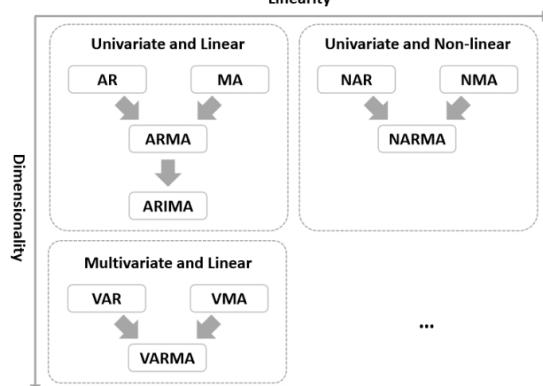
“지도학습(예측 분류), 비지도학습 문제에 모두 활용되는 필수 알고리즘”

“미래 예측을 포함한 추천 서비스와 같은 비즈니스에 활용중”



1) 통계추론(Statistical Inference) 알고리즘: 통계분포에 기반한 설명력 중심 알고리즘

Linearity



To understand the complicated methods, we first need to understand the basic concepts

(1) 단변량 선형기반: Y가 1개 & Y와 X의 관계를 선형 가정

- Linear Regression
- ARIMA(AutoRegressive Integrated Moving Average)
- ARIMAX
- SARIMAX

(2) 다변량 선형기반: Y가 2개 이상 & Y와 X의 관계를 선형 가정

- Bayesian-based Models
- Vector Autoregression(VAR)
- Vector Error Correction Model(VECM)

(3) 비선형기반: Y와 X의 관계를 비선형 가정

- Exponential Smoothing
- ETS(Error/Trend/Seasonal)
- Kalman Filter
- State Space Model
- Change Point Detection(CPD)
- Autoregressive conditional heteroskedasticity(ARCH)

- Generalized Autoregressive Conditional Heteroskedasticity(GARCH)

2) 기계학습/딥러닝 알고리즘: 컴퓨팅 기반 인공지능 알고리즘으로 정확성 높은 비선형 관계 추론

- Prophet
- Neural Prophet
- RNN(Recurrent Neural Network)
 - LSTM(Long Short-Term Memory)
 - GRU(Gated Recurrent Unit)
- Neural Networks Autoregression(NNAR)
 - Attention
 - Self-attention
- Transformer

3) Platforms: 글로벌 기업들이 독자적으로 개발한 시계열 분석 플랫폼 확대중

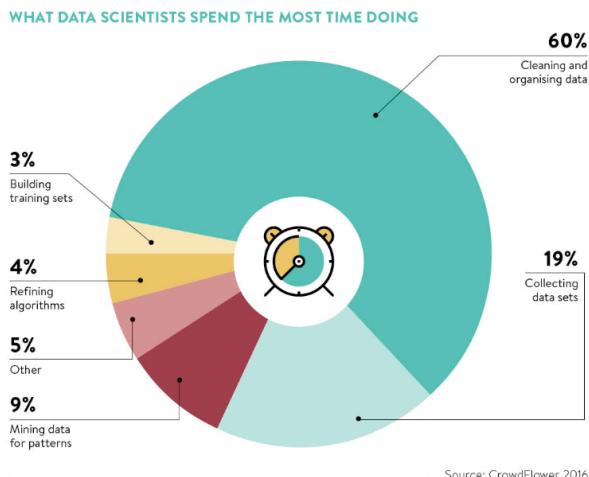
- [Amazon Forecast](#)
- [Automated ML Time-series Forecasting at Microsoft Azure](#)
- [Time Series Forecasting with Google Cloud AI Platform](#)

3 전처리 방향(Preprocessing)

"시계열예측과 기계학습/딥러닝 간 전처리와 알고리즘 방식 차이 때문에, 별도로 (1) 시계열 데이터분석 단계 이해, (2) 시계열 데이터의 전처리, (3) 시계열 알고리즘 이해 필수!"

• 목표:

- 대량으로 수집된 데이터는 그대로 활용 어려움
- 잘못 수집/처리 된 데이터는 엉뚱한 결과를 발생
- 알고리즘이 학습이 가능한 형태로 데이터를 정리



일반적인 전처리 필요항목:

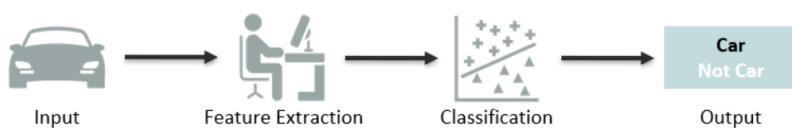
- 데이터 결합
- 결측값 처리
- 이상치 처리
- 자료형 변환
- 데이터 분리
- 데이터 변환
- 스케일 조정

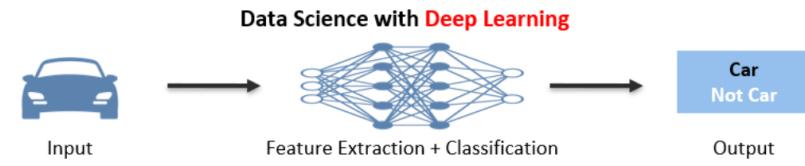
⇒ "알고리즘의 범위와 종류가 다양하여, 각 알고리즘의 입력에 맞게 변환 하는 것이 최선"

3.1 시계열 변수추출(Feature Engineering)

• 데이터 과학자들은 보통 수동/자동 변수 처리 및 변환(Feature Engineering)에 익숙하지만, 새로운 변수를 생성하는 것은 분석에서 가장 중요하고 시간이 많이 걸리는 작업 중 하나이며, 머신러닝과 딥러닝의 발전으로 점차 자동화 중

Data Science with Machine Learning





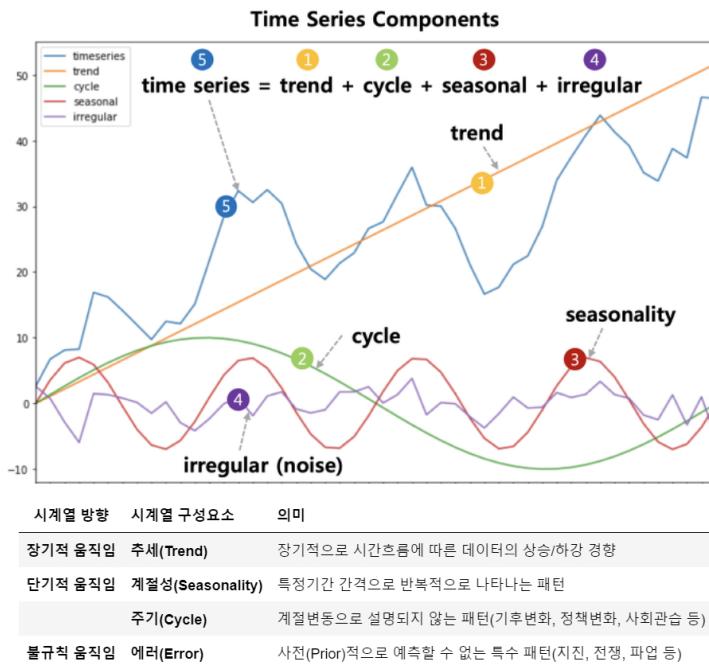
"변수 생성시 주의할 점!"

- 미래의 실제 종속변수 예측값이 어떤 독립/종속변수의 Feature Engineering에 의해 효과가 있을지 단정할 수 없음
- 독립변수의 예측값을 Feature Engineering를 통해 생성될 수 있지만 이는 종속변수의 예측에 오류증가를 야기할 수 있음

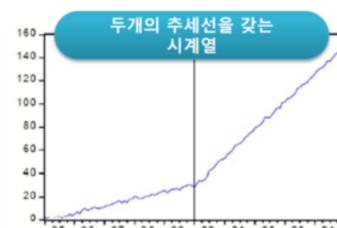
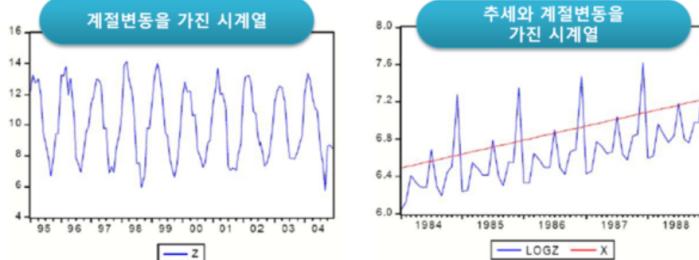
3.1.1 구성요소 및 시간빈도

1) 시계열 데이터는 여러 구성 성분 :

- 시계열 = 체계적 성분 + 불규칙 성분



2) 여러 구성성분들이 단기적 및 장기적 으로 복합적으로 융합되어 작동



3) **빈도(Frequency)**: 사람이 인지하는 데이터의 빈도를 컴퓨터는 명확히 인지하지 못하기 때문에 별도 설정 필요

- 계절성 패턴(Seasonality)이 나타나기 전까지의 데이터 갯수로 분석가가 반영

- 예시1:

Data	Seasonality	Frequency
Annual	Annual	1
Quarterly	Annual	4
Monthly	Annual	12
Weekly	Annual	52

- 예시2:

Data	Seasonality	Frequency
Daily(데이터가 Day 단위로 수집)	Weekly	7
	Annual	365

- 예시3:

Data	Seasonality	Frequency
Minutely(데이터가 Minute 단위로 수집)	Hourly	60
	Daily	24 x 60
	Weekly	24 x 60 x 7
	Annual	24 x 60 x 365

- 빈도 설정을 위한 Python 함수 옵션 :

Alias	Description
B	Business day
D	Calendar day
W	Weekly
M	Month end
Q	Quarter end
A	Year end
BA	Business year end
AS	Year start
H	Hourly frequency
T, min	Minutely frequency
S	Secondly frequency
L, ms	Millisecond frequency
U, us	Microsecond frequency
N, ns	Nanosecond frequency

- 빈도 설정 후 비어있는 시간 결측치가 발견되거나 빈도 변경으로 발생하는 결측치를 채우기 위한 Python 함수 옵션 :

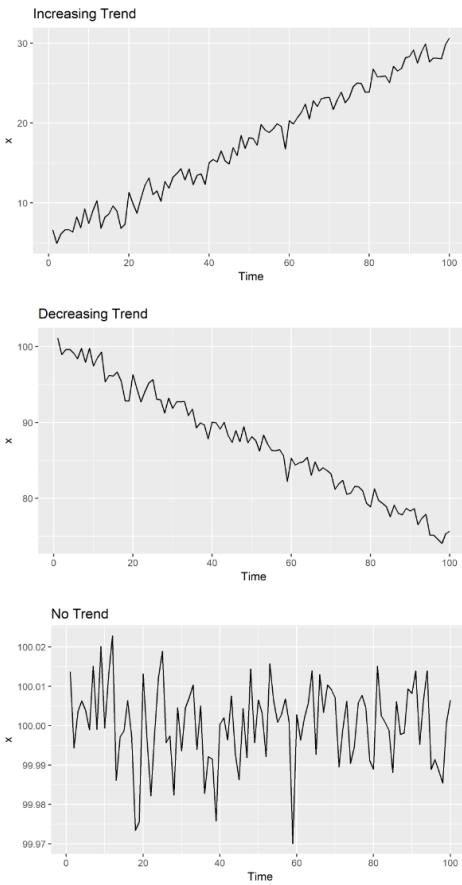
- 시계열에는 노이즈 데이터 또는 관찰되지 못한 기간이 종종 존재
- 측정하고 데이터를 기록하는 과정에서의 오류나 예측치 못한 상황으로 인해 발생
- 예를 들어 상품의 품질로 인하여 장기간 판매량이 없는 경우 데이터는 없을 수 있고 이를 적절하게 전처리하는 것 이 실제 문제 해결 성능에 매우 중요

Method	Description
bfill	Backward fill
count	Count of values
ffill	Forward fill
first	First valid data value
last	Last valid data value
max	Maximum data value
mean	Mean of values in time range
median	Median of values in time range
min	Minimum data value
nunique	Number of unique values
ohlc	Opening value, highest value, lowest value, closing value
pad	Same as forward fill
std	Standard deviation of values
sum	Sum of values
var	Variance of values

3.1.2 추세/계절성/주기/지연값 등

1) 추세(Trend, T_t): 시계열이 시간에 따라 증가, 감소 또는 일정 수준을 유지 하는 경우

(시각적 이해)



(수학적 이해)

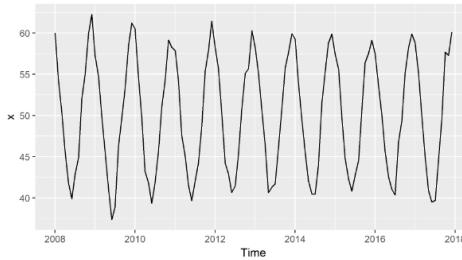
- 확률과정의 기댓값 함수를 알아내는 것
- 확률과정(Y_t)이 추정이 가능한 추세함수($f(t)$)와 정상확률과정(Y_t^s)의 합

$$Y_t = f(t) + Y_t^s$$

2) 계절성(Seasonality, S_t): 일정한 빈도로 주기적으로 반복되는 패턴(m), 특정한 시간값(달/요일)에 따라 기대값이 달라지는 것

- 예시: 주기적 패턴이 12개월마다 반복($m = 12$)

Seasonal Pattern (Frequency = 12)



- 대표적 계절성 변수 생성 2가지:

- 수치값 그대로 반영 → Label Encoding

“기존 변수”

계절
여름
봄
봄
겨울
가을
가을
여름

“변환 변수”

계절
1
0
0
3
2
2
1

- 계절성 주기 내 시점마다 별도 변수 생성 → OneHotEncoding

“기준 변수”

계절
여름
봄
봄
겨울
가을
가을
여름

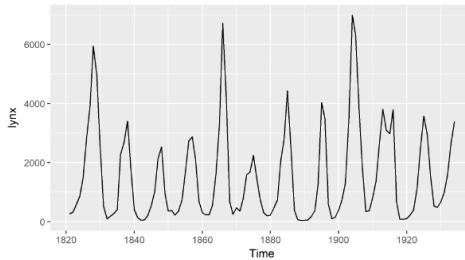
“가변수”

계절_봄	계절_여름	계절_가을	계절_겨울
0	1	0	0
1	0	0	0
1	0	0	0
0	0	0	1
0	0	1	0
0	0	1	0
0	1	0	0

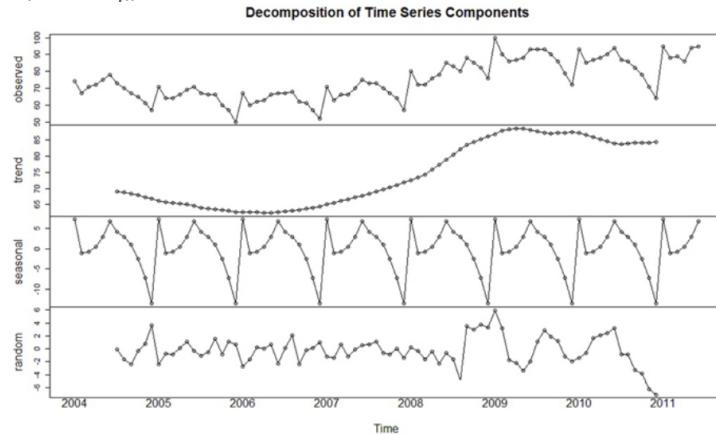
3) 주기(Cycle, C_t): 일정하지 않은 빈도로 발생하는 패턴(계절성과 다름)

- 예시: 빈도가 1인 경우에도 발생 가능($m = 1$)

Annual Canadian Lynx trappings 1821-1934



4) 시계열 분해(추세/계절성/잔차(Residual, e_t)):



(1) 가법모형(Additive Model): 추세 + 계절성 + 주기 + 오차

- 구성요소간 독립성을 가정하고 시계열이 구성요소의 합을 가정
- 주로 계절성분의 진폭/분산이 시간이 흘러도 일정한 경우 사용

(2) 승법모형(Multiplicative Model): 추세 * 계절성 * 주기 * 오차

- 구성요소간 비독립성을 가정하고 시계열이 구성요소의 곱을 가정
- 주로 계절성분의 진폭/분산이 시간에 따라 일정하지 않은 경우 사용

5) 지연값(Lagged values, $Lag_t(X_1)$): 변수의 지연된 값을 별도로 독립변수를 생성하는 것으로, ARIMA/VAR/NNAR(Neural Network Autoregression) 등의 알고리즘도 내부적으로 활용

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

6) 시간변수: 시간정보가 담고 있는 년/월/일/요일 등 자체를 별도 독립적인 변수로 생성

- 요약:

- 시계열 구성요소는 각 변수의 시간패턴을 파악하는데 중요
- FE를 통해 생성된 변수의 입력(Input) 형태나 빈도로 알고리즘 선택 시 고려해야
- 생성된 변수의 패턴이 기준 모델에서 반영하지 않은 패턴이라면 예측 성능을 높임
- 예측성능 향상 뿐 아니라 결과를 해석하고 해당 속성을 분석하여 가능한 원인 식별에 도움

3.2 데이터 분리(Data Split)

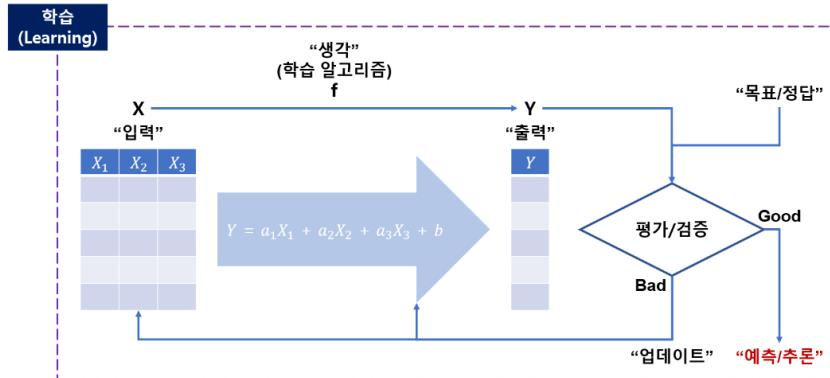
- 배경:

(1) 독립변수와 종속변수 구분

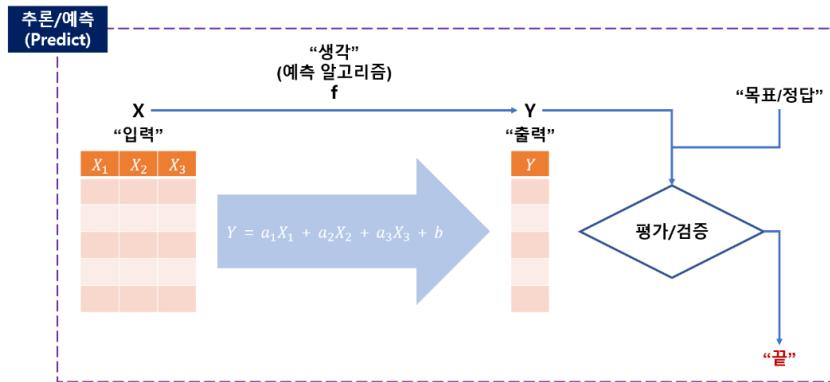
대분류	의미/예시
독립변수(Independent Variable)	다른 변수에 영향을 미치는 변수 (X)
종속변수(Dependent Variable)	다른 변수에 의해 영향을 받는 변수 (Y)

(2) 과거/현재와 미래 기간 구분: 과거/현재의 상황을 분석하고, 미래를 예측 할 수 있는 환경 구축

- Training Period: 과거/현재의 상황을 분석



- Testing Period: 미래를 예측 할 수 있는 환경



1) 간단한 방법(Holdout Validation):

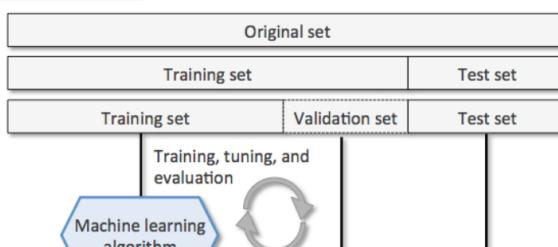
- 훈련셋(Training set): 일반적으로 전체 데이터의 70% 사용
- 테스트셋(Testing set): 일반적으로 전체 데이터의 30% 사용

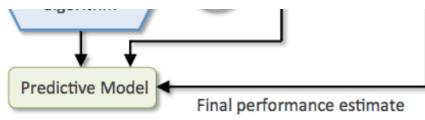
2) 일반적 방법(Simple Validation):

- 훈련셋(Training set): 일반적으로 전체 데이터의 60%를 사용
- 검증셋(Validation set):

- 개발셋이라고도 하며, 일반적으로 전체 데이터의 20%를 사용함
- 훈련된 여러 가지 모델들의 성능을 테스트하는데 사용되며 모델 선택의 기준이 됨

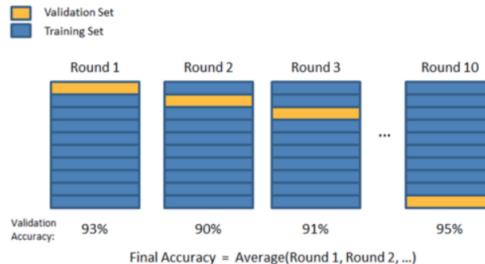
- 테스트셋(Testing set): 일반적으로 전체 데이터의 20%를 사용하며 최종 모델의 정확성을 확인하는 목적으로 사용됨





3) K교차검사(K-fold Cross Validation):

- (1) 훈련셋을 복원없이 K 개로 분리한 후, $K-1$ 는 하위훈련셋으로 나머지 1개는 검증셋으로 사용함
- (2) 검증셋과 하위훈련셋을 번갈아가면서 K 번 반복하여 각 모델별로 K 개의 성능 추정치를 계산
- (3) K 개의 성능 추정치 평균을 최종 모델 성능 기준으로 사용

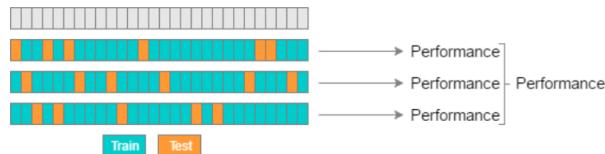


4) K-fold vs. Random-subsamples vs. Leave-one-out vs. Leave- p -out

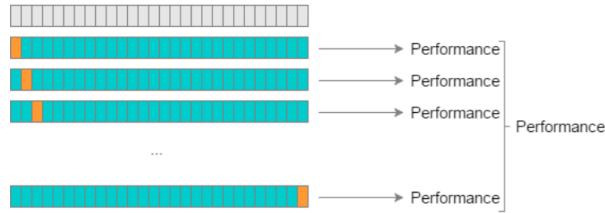
- K-fold



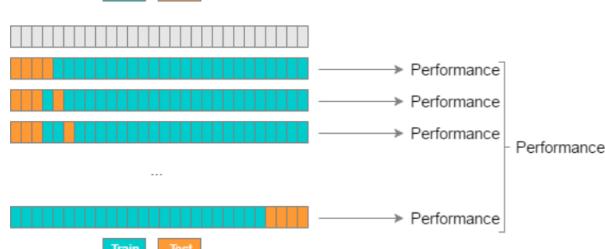
- Random-subsamples



- Leave-one-out



- Leave- p -out

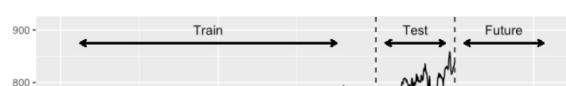


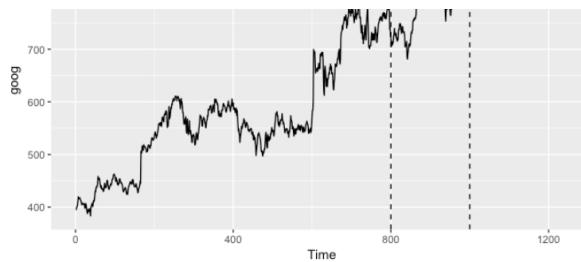
3.2.1 시계열 데이터 분리

1) 시계열 데이터인 경우 랜덤성(set.seed) 없이 시간축 유지가 핵심 !

- 훈련셋 < 검증셋 < 테스트셋

- 훈련셋(Training set): 가장 오래된 데이터
- 검증셋(Validation set): 그 다음 최근 데이터
- 테스트셋(Testing set): 가장 최신의 데이터





2) 최선의 성능을 가지는 Robust 모델링을 위해 시계열 교차검증 필수!

(1) 모든 미래 시점을 한번에 예측하면 정확성 낮아짐 : 미래 시점마다 Train/Test 재분류 필요

- Sliding Window: 고정된 사이즈의 Train & Test Window를 움직이며 데이터를 재분리
- Expanding Window: 가변 사이즈 Train Window & 고정된 사이즈 Test Window를 움직이며 데이터를 재분리

Sliding Window



Expanding Window



[\(https://eng.uber.com/forecasting-introduction/\)](https://eng.uber.com/forecasting-introduction/)

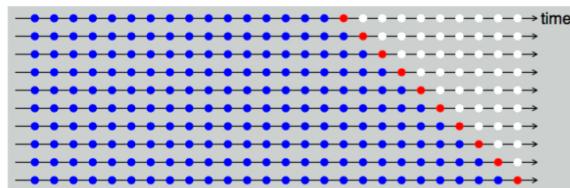
(2) 단기/중기/장기 모든 미래시점에 동일 성능을 가지는 Robust 알고리즘은 없기 때문에 시점마다 별도 모델링 필요

- 1스텝 교차검증(One-step Ahead Cross-validation):

Traditional evaluation

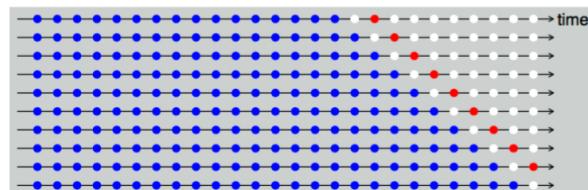


Time series cross-validation



- 2스텝 교차검증(Two-step Ahead Cross-validation):

Time series cross-validation



3.3 데이터 변환(Reality & Feature Selection)

3.3.1 시간현실 반영(Reality)

"미래의 시간패턴을 미리 반영하는건 비현실적, 이는 과적합(Overfitting) 유발"

- 데이터 전처리시, 데이터 분리 후 패턴을 추출하여 해결

✓ 데이터 전처리: (0) 끝도 없을 뻔한 Raw를 끝도 있는 Data로 변환

100	T50	횟수	111	TPU	...
few	Gds	Hvi	Rew	Fa	...
Fre	CT	QTP	D	합	...
'1'	1	23	22	NaN	43
76	NaN	43	32	1	8
'Hi'	NaN	NaN	NaN	NaN	87
23	98	NaN	64	46	NaN
c	90	NaN	'WW'	24	'KK'
t	NaN	2	NaN	NaN	6

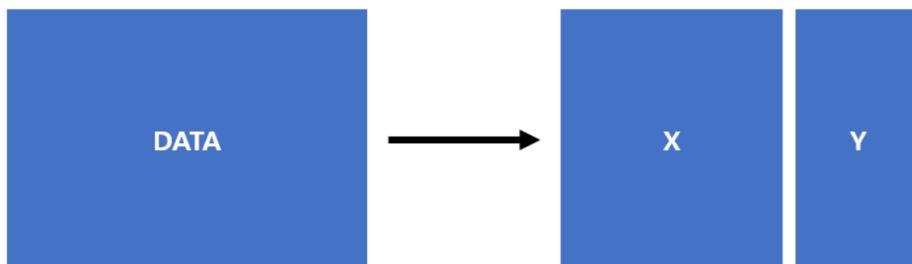
빈도	시간	총량	기간	누적	...
1	1	23	22	21	43
76	33.3	43	32	1	8
5	33.3	52	35	21	87
23	98	52	64	46	61
90	33.3	2	24	33	4
55	2	52	35	21	6
NaN	NaN	NaN	NaN	NaN	NaN

64	NaN	90	'IU'	4	76
54	55.5	90	11	4	76

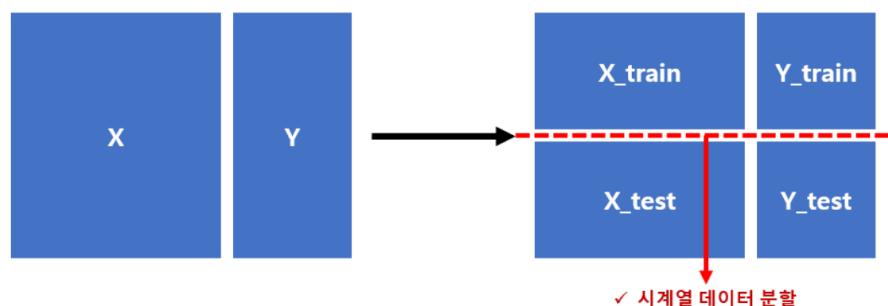
✓ 시계열 전처리

: 빈도, 추세, 계절성, 주기, 자연값, 시간정보 등
알고리즘이 이해하는 값으로 변경 포함
(여기까진 일반적 전처리)
시간 흐름에 따른 “수치변화”와
비어 있는 “시간간격”까지 전처리

✓ 데이터 분할: (1) 목표/종속변수 Y와 설명/독립변수 X 설정



✓ 데이터 분할: (2) 학습데이터 Train과 예측 데이터 Test로 분할



: (1) 절대로 “랜덤” 분할이 아닌 “시간축 유지”
(2) 새로운 미래 시점마다 Train/Test “재분류”
(3) 단기/중기/장기 시점에 따라
“별도 모델링” 해야 하기 때문에
Test “추가 재분류”

3.3.2 변수들의 독립성 향상(Condition Number)

1) 배경: 예측 성능을 향상 시키려면 독립변수들의 독립성이 높아야 함

- 이상적: Train 예측성능 ↑ + Test 예측성능 ↑
- 현실적: Train 예측성능 <<< Test 예측성능 ⇔ 낮은 조건수(Condition Number)

2) 낮은 조건수란?:

(비수학적 이해)

- 독립변수들(X)의 상호의존도(관련성) 가 분석결과(Y)에 주는 영향을 줄이고, 독립변수 각각의 순수한 영향 만을 반영

(수학적 이해)

- 독립변수(X)가 움직일 수 있는 변동성(공분산)을 줄여 분석결과(Y)의 변동을 줄여 신뢰성 향상
- 변동성(공분산)?: 알고리즘이 예측 영향력(계수/가중치) 추정 과정에서 발생하는 역행렬의 변화에 오차가 미치는 영향

$$\text{Condition Number(조건수)} = \frac{\lambda_{\max}}{\lambda_{\min}}$$

$$\lambda_{\max} = \max[\text{eigenvalue}\{\text{Cov}(X^T X)\}],$$

$$\lambda_{\min} = \min[\text{eigenvalue}\{\text{Cov}(X^T X)\}]$$

```
In [1]: # 조건수가 작을 때
# X 데이터
import numpy as np
X = np.eye(4)
X
```

executed in 195ms, finished 15:57:41 2022-06-11

```
Out[1]: array([[1., 0., 0., 0.],
   [0., 1., 0., 0.],
   [0., 0., 1., 0.],
   [0., 0., 0., 1.]])
```

```
In [2]: # Y 데이터
Y = np.ones(4)
Y
```

executed in 14ms, finished 15:57:41 2022-06-11

```

Out[2]: array([1., 1., 1., 1.])

In [3]: # 계수 추정
np.linalg.solve(X, Y)
executed in 13ms, finished 15:57:41 2022-06-11
Out[3]: array([1., 1., 1., 1.])

In [4]: # X 데이터 오차반영
X_new = X + 0.0001 * np.eye(4)
X_new
executed in 14ms, finished 15:57:41 2022-06-11
Out[4]: array([[1.0001, 0., 0., 0.],
               [0., 1.0001, 0., 0.],
               [0., 0., 1.0001, 0.],
               [0., 0., 0., 1.0001]])

In [5]: # 계수 추정
np.linalg.solve(X_new, Y)
executed in 14ms, finished 15:57:41 2022-06-11
Out[5]: array([0.99990001, 0.99990001, 0.99990001, 0.99990001])

In [6]: # 조건수 확인
np.linalg.cond(X)
executed in 29ms, finished 15:57:41 2022-06-11
Out[6]: 1.0

In [7]: # 조건수가 끌 때
# X 데이터
from scipy.linalg import hilbert
X = hilbert(4)
X
executed in 119ms, finished 15:57:41 2022-06-11
Out[7]: array([[1., 0.5, 0.33333333, 0.25],
               [0.5, 1., 0.33333333, 0.25],
               [0.33333333, 0.25, 1., 0.16666667],
               [0.25, 0.2, 0.16666667, 1.]))

In [8]: # Y 데이터
Y = np.ones(4)
Y
executed in 14ms, finished 15:57:41 2022-06-11
Out[8]: array([1., 1., 1., 1.])

In [9]: # 계수 추정
np.linalg.solve(X, Y)
executed in 12ms, finished 15:57:41 2022-06-11
Out[9]: array([-4., 60., -180., 140.])

In [10]: # X 데이터 오차반영
X_new = X + 0.0001 * np.eye(4)
X_new
executed in 14ms, finished 15:57:41 2022-06-11
Out[10]: array([[1.0001, 0.5, 0.33333333, 0.25],
               [0.5, 1.0001, 0.33333333, 0.25],
               [0.33333333, 0.25, 1.0001, 0.16666667],
               [0.25, 0.2, 0.16666667, 1.0001]])

In [11]: # 계수 추정
np.linalg.solve(X_new, Y)
executed in 14ms, finished 15:57:41 2022-06-11
Out[11]: array([-0.58897672, 21.1225671, -85.75912499, 78.45650825])

In [12]: # 조건수 확인
np.linalg.cond(X)
executed in 13ms, finished 15:57:41 2022-06-11
Out[12]: 15513.738738929038

```

3) 조건수를 낮추는 방법: 예측 안정성(Robust/Goodfit)을 높이는 방법

- (1) 변수들의 단위차이로 숫자의 스케일들이 크게 다른 경우, 스케일링(Scaling)으로 해결 가능
- (2) 독립변수들 간에 상관관계(상호의존성)가 높은 다중공선성 존재할 경우, Variance Inflation Factor(VIF)나 Principal Component Analysis(PCA)를 통한 변수선택으로 해결 가능
- (3) 독립변수들 간 상호의존성이 높은 변수들에 패널티를 부여하는 정규화(Regularization)로 해결 가능

3.3.3 다중공선성(Multicollinearity)

1) 다중공선성(Multicollinearity)이 발생 하는 경우:

- 특정 독립변수가 다른 독립변수의 조합으로 표현 될 수 있는 경우
- 독립변수들이 서로 대략似的이나 같은 관계가 갖는 경우

• 독립변수의 공분산 행렬(Covariance Matrix) 벡터공간(Vector Space)의 차원과 독립변수의 수가 달라 변수들이 모두 독립이 아닌 경우 (Full Rank가 아니다)

=> "다중공선성이 있으면 독립변수 공분산 행렬의 조건수(Condition Number) 증가"
=>"조건수(Condition Number)가 증가하면 과적합(Overfitting) 발생 가능성 증가"

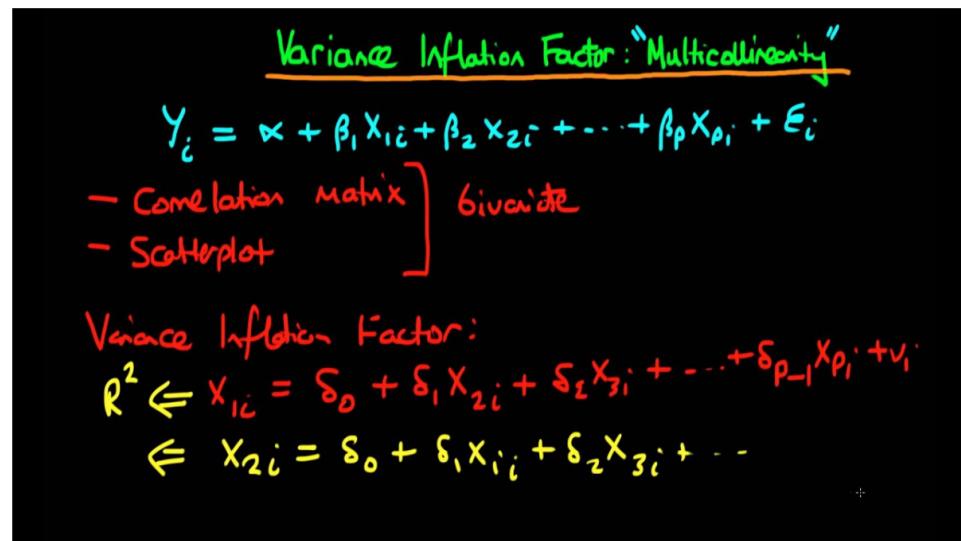
2) 다중공선성을 줄이는 방법:

(1) Variance Inflation Factor(VIF) 변수선택

"VIF는 독립변수를 다른 독립변수들로 선형회귀한 성능들을 비교하여 상호 의존성이 높은 변수를 제거하거나 의존성이 낮은 변수를 선택"

$$VIF_i = \frac{\sigma_e^2}{(n-1)Var(X_i)} \cdot \frac{1}{1-R_i^2}$$

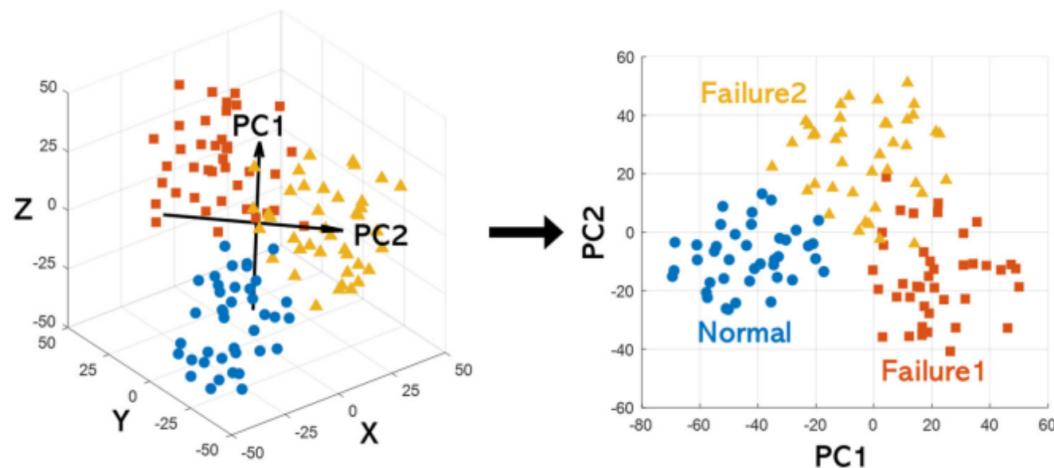
(R_i^2 : 독립변수 X_i 를 다른 독립변수들로 선형회귀한 성능)



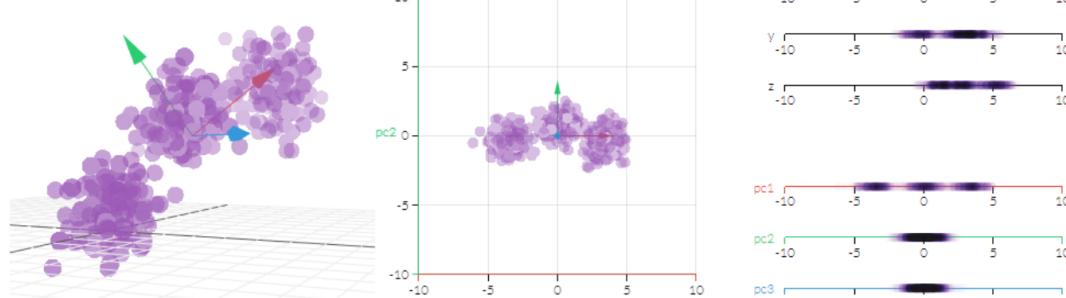
(<https://www.youtube.com/watch?v=0SBIXgPVex8>)

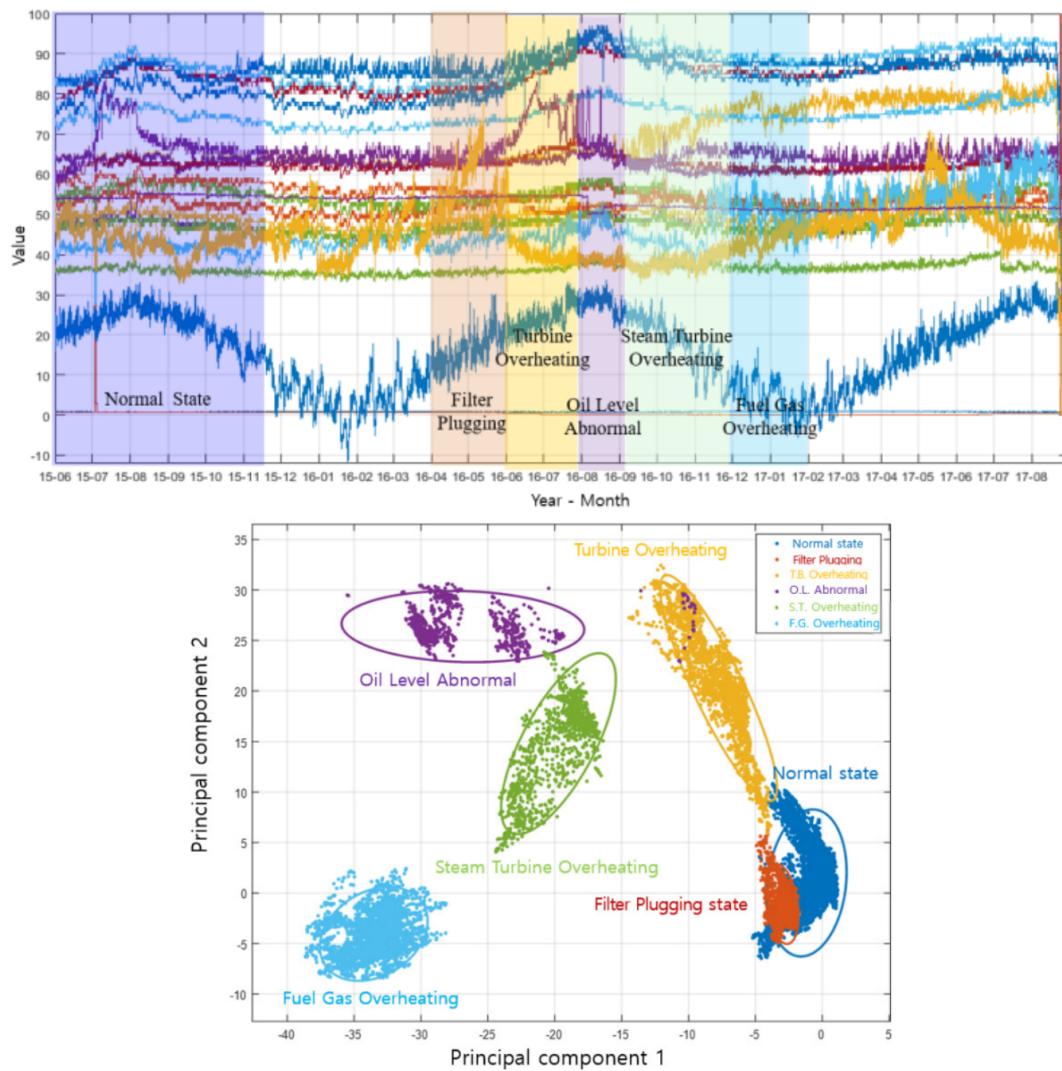
(2) Principal Component Analysis(PCA) 변수선택

"PCA는 다차원에서 상호 관련성이 높은 독립변수들을 관련성이 없는 서로 독립인 소차원으로 변환하는 알고리즘으로, 이를 통해 상호 의존성을 제거"



(<https://www.mdpi.com/2076-3417/11/9/3780>)





(<https://www.mdpi.com/2076-3417/11/9/3780>)

3.4 스케일 조정(Scaling)

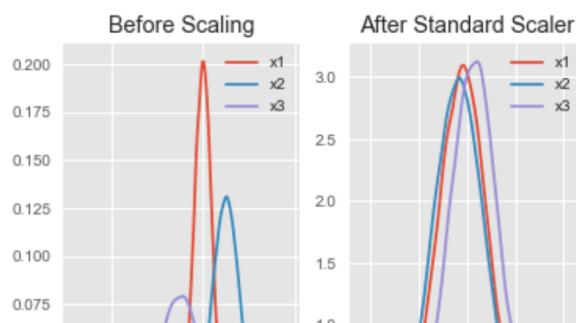
- 목적: 변수들의 크기를 일정하게 맞추어 크기 때문에 영향이 높은 현상을 회피
- 수학적: 독립 변수의 공분산 행렬 조건수(Condition Number)를 감소 시켜 최적화 안정성 및 수렴 속도 향상
- 컴퓨터적: PC 메모리를 고려하여 오버플로우(Overflow)나 언더플로우(Underflow)를 줄여줌

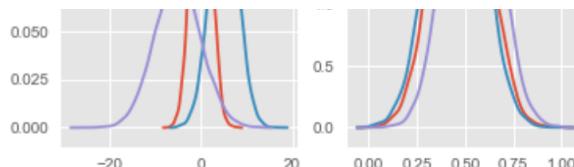
1) Standard Scaler:

$$\frac{X_{it} - E(X_i)}{SD(X_i)}$$

- 기본 스케일로 평균을 제외하고 표준편차를 나누어 변환
- 각 변수(Feature)가 정규분포를 따른다는 가정 이기에 정규분포가 아닐 시 최선이 아닐 수 있음

```
sklearn.preprocessing.StandardScaler().fit()
sklearn.preprocessing.StandardScaler().transform()
sklearn.preprocessing.StandardScaler().fit_transform()
```



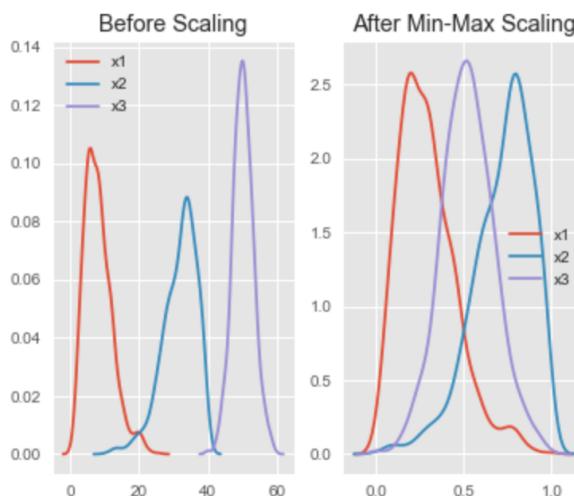


2) Min-Max Scaler:

$$\frac{X_{it} - \min(X_i)}{\max(X_i) - \min(X_i)}$$

- 가장 많이 활용되는 방식으로 최소~최대 값이 0~1 또는 -1~1 사이의 값으로 변환
- 각 변수(Feature)가 정규분포가 아니거나 표준편차가 매우 작을 때 효과적

```
sklearn.preprocessing.MinMaxScaler().fit()
sklearn.preprocessing.MinMaxScaler().transform()
sklearn.preprocessing.MinMaxScaler().fit_transform()
```

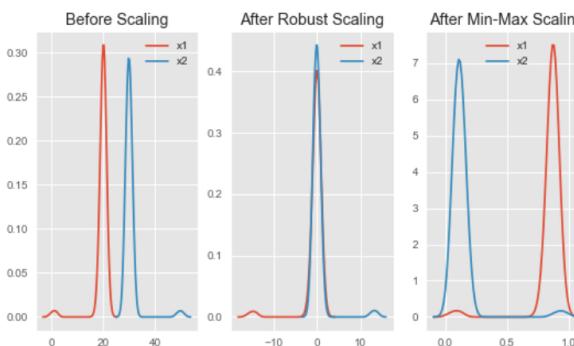


3) Robust Scaler:

$$\frac{X_{it} - Q_1(X_i)}{Q_3(X_i) - Q_1(X_i)}$$

- 최소-최대 스케일러와 유사하지만 최소-최대 대신에 IQR(Interquartile Range) 중 25%값/75%값을 사용하여 변환
- 이상치(Outlier)에 영향을 최소화하였기에 이상치가 있는 데이터에 효과적이고 적은 데이터에도 효과적인 편

```
sklearn.preprocessing.RobustScaler().fit()
sklearn.preprocessing.RobustScaler().transform()
sklearn.preprocessing.RobustScaler().fit_transform()
```



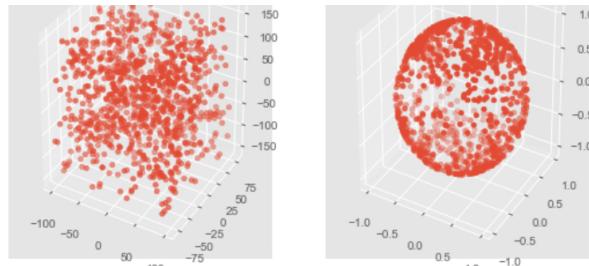
4) Normalizer:

$$\frac{X_{it}}{\sqrt{X_i^2 + X_j^2 + \dots + X_k^2}}$$

- 각 변수(Feature)를 전체 n 개 모든 변수들의 크기들로 나누어서 변환(by Cartesian Coordinates)
- 각 변수들의 값은 원점으로부터 반지름 1만큼 떨어진 범위 내로 변환

```
sklearn.preprocessing.Normalizer().fit()
sklearn.preprocessing.Normalizer().transform()
sklearn.preprocessing.Normalizer().fit_transform()
```





4 모델링 방향(Modeling)

"시계열예측과 기계학습/딥러닝 간 전처리와 알고리즘 방식 차이 때문에, 별도로 (1) 시계열 데이터분석 단계 이해, (2) 시계열 데이터의 전처리, (3) 시계열 알고리즘 이해 필수!"

4.1 시계열 알고리즘의 특징과 종류

(1) 학습된 데이터 범위 내 패턴 뿐 아니라 외부 순차적 시점들의 패턴으로도 확장 할 수 있어야 시계열 알고리즘

- 대부분의 기계학습 알고리즘은 데이터 범위 내 패턴 만을 추출
- 시계열 데이터나 FE를 통해 생성된 변수들은 미래시점의 패턴으로도 확장 시킬 수 있음

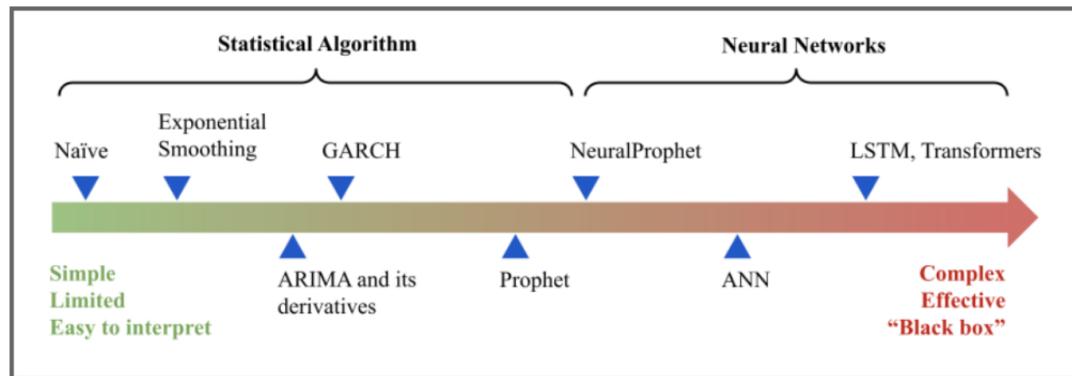
(2) 시계열 알고리즘은 특정시점(점추정)이 아닌 특정시점의 값의 범위(구간추정)를 추정 하는 알고리즘으로 설명력을 보유

- 대부분의 기계학습 알고리즘은 통계분포에 기반하지 않기 때문에 점추정 알고리즘
- 값의 범위 정확성을 담보하진 않지만, 점추정 보다 다양한 해석을 가능하게 함
- 정확성 vs. 설명력 반비례 관계 존재

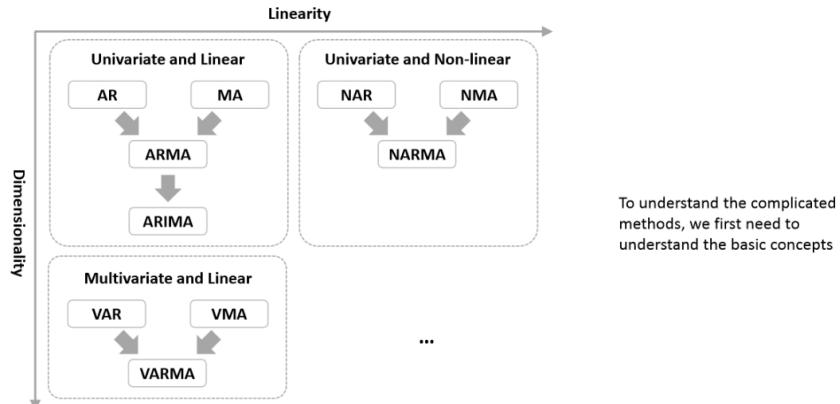
"통계추론, 기계학습 및 딥러닝의 흐름에 시간패턴을 반영하려 진화"

"지도학습(예측 분류), 비지도학습 문제에 모두 활용되는 필수 알고리즘"

"미래 예측을 포함한 추천 서비스와 같은 비즈니스에 활용중"



1) 통계추론(Statistical Inference) 알고리즘: 통계분포에 기반한 설명력 중시 알고리즘



(1) 단변량 선형기반: Y가 1개 & Y와 X의 관계를 선형 가정

- Linear Regression
- ARIMA(AutoRegressive Integrated Moving Average)
- ARIMAX
- SARIMAX

(2) 다변량 선형기반: Y가 2개 이상 & Y와 X의 관계를 선형 가정

- Bayesian-based Models
- Vector Autoregression(VAR)
- Vector Error Correction Model(VECM)

(3) 비선형기반: Y와 X의 관계를 비선형 가정

- Exponential Smoothing
- ETS(Error/Trend/Seasonal)
- Kalman Filter
- State Space Model
- Change Point Detection(CPD)
- Autoregressive conditional heteroskedasticity(ARCH)
- Generalized Autoregressive Conditional Heteroskedasticity(GARCH)

2) 기계학습/딥러닝 알고리즘: 컴퓨팅 기반 인공지능 알고리즘으로 정확성 높은 비선형 관계 추론

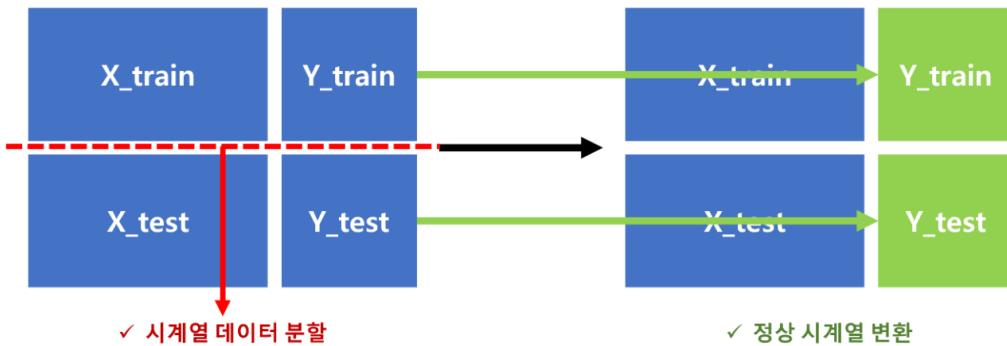
- Prophet
- Neural Prophet
- RNN(Recurrent Neural Network)
- LSTM(Long Short-Term Memory)
- GRU(Gated Recurrent Unit)
- Neural Networks Autoregression(NNAR)
- Attention
- Self-attention
- Transformer

3) Platforms: 글로벌 기업들이 독자적으로 개발한 시계열 분석 플랫폼 확대중

- Amazon Forecast
- Automated ML Time-series Forecasting at Microsoft Azure
- Time Series Forecasting with Google Cloud AI Platform

4.2 정상성(Stationarity Process)

✓ 시계열 데이터 전처리: 머신러닝과 달리 비정상 시계열(Y)을 정상 시계열로 변환



- : (1) 절대로 “랜덤” 분할이 아닌 “시간축 유지”
- (2) 새로운 미래 시점마다 Train/Test “재분류”
- (3) 단기/중기/장기 시점에 따라
“별도 모델링” 해야 하기 때문에
Test “추가 재분류”

- : (1) 필수는 아니지만 범위제한으로 예측성능↑
- (2) 모델 복잡도 낮아져서
Bias↑ + Variance↓ => 과적합↓ + 예측성능↑
- (2) 통계추론 알고리즘은 대부분 학습위해 필수

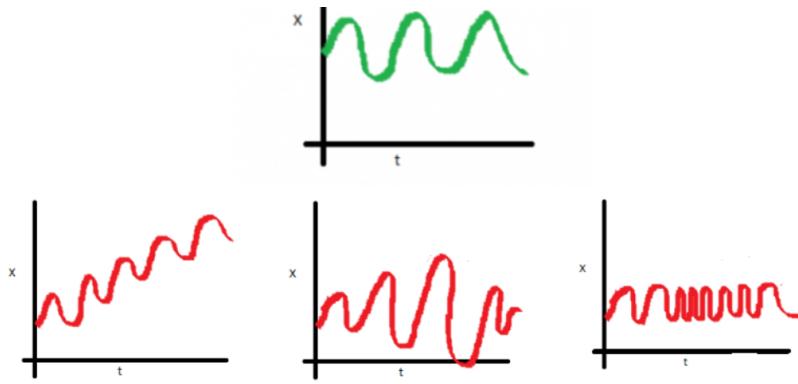
0) 배경: 시계열 데이터를 활용한 분석에서는 새로운 가정 필요

데이터 방향	현실 가정
시간 무관 일반적 데이터	모든 데이터 샘플들이 동일 분포에서 독립적으로 생성 → Independent and Identically Distributed (i.i.d.)
시간 관련 시계열 데이터	i.i.d. 가정 할 수 없고 현실문제를 단순화 한 새로운 가정 필요 → 시간이 흘러도 통계적/확률적 성질이 변하지 않는다는 정상성(Stationary)

- 새로운 가정 하에서, 정상성/비정상성 자체를 라벨로 활용하여 분류 지도학습 및 군집 비지도학습에도 활용

1) 정상성의 의미: 시간이 흘러도 통계적 특성(Statistical Properties)이 변하지 않고 일정

- 통계적 특성(Statistical Properties): 주로 평균(Mean)과 분산(Variance)/공분산(Covariance)을 의미하지만, 가장 보수적으로는 이를 포함한 모든 분포적 특성을 포함
- Homoscedasticity: 분산이 일정한(유한한, 발산하지 않는) 경우를 의미
- Heteroscedasticity: 분산이 발산하는 경우를 의미



2) 정상성의 종류:

- 변수:

$$X = \{X_1, X_2, \dots, X_n\}$$

- 확률과정:

$$\begin{aligned} Y &= \{\dots, Y_{-2}, Y_{-1}, Y_0, Y_1, Y_2, \dots\} \\ X_1 &= \{\dots, X_{1,-2}, X_{1,-1}, X_{1,0}, X_{1,1}, X_{1,2}, \dots\} \\ X_2 &= \{\dots, X_{2,-2}, X_{2,-1}, X_{2,0}, X_{2,1}, X_{2,2}, \dots\} \\ &\dots \end{aligned}$$

(1) 약정상(Weak Stationarity, Wide-sense Stationary Process):

(비수학적 이해)

1. ..., $X_{i,-1}, X_{i,0}, X_{i,-1}, \dots$ 변수의 각 시계열 샘플들은 동일 분포 며,
2. ..., $(X_{i,-3}, X_{i,-1}), (X_{i,-2}, X_{i,0}), (X_{i,-1}, X_{i,1}), \dots$ 임의간격 샘플 2개씩의 봄음도 동일 분포

(수학적 이해)

1. $E(X_{it}) = \mu$, for all time t (The first moment estimation)
2. $Var(X_{it}) = E(X_{it}^2) - E(X_{it})^2 < \infty$, for all time t (The second moment estimation)
3. $Cov(X_{is}, X_{ik}) = Cov(X_{i(s+h)}, X_{i(k+h)}) = f(h)$, for all time s, k, h (The cross moment estimation)
 \Rightarrow covariance just depends on h .

(2) 강정상(Strong Stationarity, Strictly Stationary Process):

"확률과정의 모든 분포 모멘트(Moment)가 시간 차이에만 의존하는 것(절대시간 미의존)"

(비수학적 이해)

1. ..., $X_{i,-1}, X_{i,0}, X_{i,-1}, \dots$ 변수의 각 시계열 샘플들은 동일 분포 며,
2. ..., $(X_{i,-3}, X_{i,-1}), (X_{i,-2}, X_{i,0}), (X_{i,-1}, X_{i,1}), \dots$ 임의간격 샘플 2개씩의 봄음도 동일 분포
3. ..., $(X_{i,-5}, X_{i,-3}, X_{i,-1}), (X_{i,-2}, X_{i,0}, X_{i,2}), (X_{i,-1}, X_{i,1}, X_{i,3}), \dots$ 임의간격 샘플 3개씩의 봄음도 동일 분포
4. $(X_{i,-\infty}, \dots, X_{i,-1}, X_{i,1}, X_{i,3}, \dots, X_{i,\infty})$ 모든 시간 변화에도 동일 분포

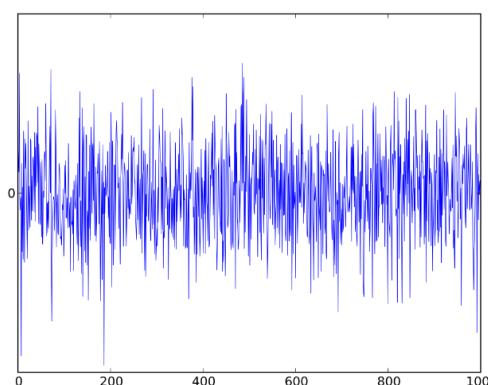
(3) 강정상과 약정상의 관계: 강정상 \Rightarrow 약정상, but 약정상 $\not\Rightarrow$ 강정상

"강정상 $\{x_{it}\}_{t=-\infty}^{t=+\infty}$ 의 특정 샘플 $\{x_{it}\}_{t=t_1}^{t=t_2}$ 의 평균과 분산까지만 일정한 약정상 일 수 있다"

"약정상 $\{x_{it}\}_{t=-\infty}^{t=+\infty}$ 이 3차 이상의 통계적 특성인 모멘트(Skewness, Kurtosis 등)에서도 일정하면 강정상이다"

3) 백색잡음(White Noise, WN):

강정상의 대표예시로 시계열분석 기본알고리즘 중 가장 중요 함



(1) 잔차들은 정규분포이고, (unbiased) 평균 0과 일정한 분산을 가져야 함:

$$\{e_t : t = \dots, -2, -1, 0, 1, 2, \dots\} \sim N(0, \sigma_e^2)$$

where $e_t \sim i.i.d$ (independent and identically distributed)

$$\begin{aligned} e_t &= Y_t - \hat{Y}_t, \\ E(e_t) &= 0, \\ Var(e_t) &= \sigma_{e_t}^2 \\ Cov(\epsilon_s, \epsilon_k) &= 0 \text{ for different times! (s \neq k)} \end{aligned}$$

(2) 잔차들이 시간의 흐름에 따라 상관성이 없어야 함: 자기상관함수(Autocorrelation Function, ACF)=0 확인

- 공분산(Covariance):

$$Cov(Y_s, Y_k) = E[(Y_s - E(Y_s))(Y_k - E(Y_k))] = \gamma_{s,k}$$

- 자기상관함수(Autocorrelation Function):

$$Corr(Y_s, Y_k) = \frac{Cov(Y_s, Y_k)}{\sqrt{Var(Y_s)Var(Y_k)}} = \frac{\gamma_{s,k}}{\sqrt{\gamma_s \gamma_k}}$$

- 편자기상관함수(Partial Autocorrelation Function): s 와 k 사이의 상관성을 제거한 자기상관함수

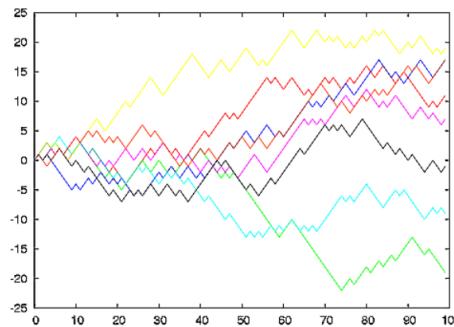
$$Corr[(Y_s - \hat{Y}_s, Y_{s-t} - \hat{Y}_{s-t})] \text{ for } 1 < t < k$$

4) 비정상 확률과정(Non-stationary Process):

- 추세가 있어서 평균인 일차 모멘트($E(y_t)$)가 0이 아니며 시간에 따라 변함
- 추세가 없지만($E(y_t) = 0$) 분산인 이차 모멘트($Var(y_t)$)가 시간에 따라 변함

- 랜덤워크(Random Walk): 비정상(Non-stationary) 대표예시로 차분시 백색잡음으로 변경

$$\begin{aligned} Y_{it} &= Y_{it-1} + \epsilon_t \\ Y_{it} - Y_{it-1} &= \epsilon_t \\ \epsilon_t &\sim N(0, \sigma_{\epsilon_t}^2) \end{aligned}$$



6) 정리:

	예시	데이터
정상성	백색잡음(White Noise)	
비정상성	랜덤워크(Random Walk)	

- 활용주요목적:

- 1) 모델링: 시계열 모형은 데이터가 Stationary라 가정 \rightarrow Stationary여야 예측 성능 높다
- 2) 잔차진단: 백색잡음 또한 Stationary \rightarrow 잔차도 Stationary여야 예측 성능 높다

- 활용단어예시:

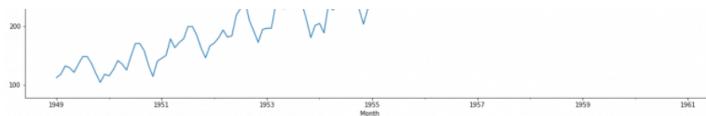
- Stationary Process: 정상성인 시계열 데이터셋(프로세스)
- Stationary Model: 정상성인 시계열데이터를 설명 하는 모델
- Trend Stationary: 트렌드를 제거하면 정상성인 시계열데이터
- Seasonal Stationary: 계절성을 제거하면 정상성인 시계열데이터
- Difference Stationary: 차분하면 정상성인 시계열데이터

4.3 정상성 테스트(Stationarity Test)

1) 기초통계 테스트(by Summary Statistics): 특정 시간(ex. 계절)에 따라 통계량이 일정한지 파악하여 정상성 확인

2) 시각화 테스트(by Visual): 추세와 계절성이 없는지 파악하여 정상성 확인

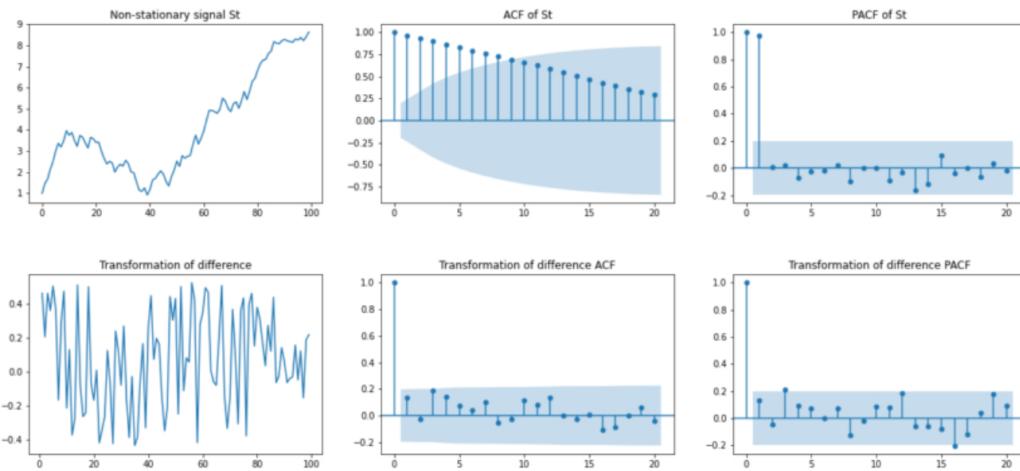




- **자기상관함수(Autocorrelation Function):** 시차변화에 따른 자기상관을 의미

$$Corr(Y_s, Y_k) = \frac{Cov(Y_s, Y_k)}{\sqrt{Var(Y_s)Var(Y_k)}} = \frac{\gamma_{s,k}}{\sqrt{\gamma_s \gamma_k}}$$

- 만약 데이터가 정상 시계열 이면, ACF 그래프가 모든 시차에서 0에 수렴하고,
- 비정상 시계열 이면, ACF 그래프가 천천히 감소하거나 큰 양의 값



(<https://www.baeldung.com/cs/acf-pacf-plots-arima-modeling>)

- 3) 통계량 테스트(by Statistics Test): 정상성 테스트 통계량으로 정상성 확인

"In statistics, a unit root test tests whether a time series variable is non-stationary and possesses a unit root."

- [Augmented Dickey-Fuller\(ADF\) test:](#) 주로 추세제거에 효과

(1) 가설확인:

종류	해석
대중주장 (귀무가설, Null Hypothesis, H_0)	데이터는 단위근(Unit Root) 있다 / 비정상 상태 / 시간의존 구조
나의주장 (대립가설, Alternative Hypothesis, H_1)	데이터는 단위근 없다 / 정상 상태 / 시간의존 구조 아니다.

(2) 유의수준 설정 및 유의확률 확인

- 유의수준: 5% (0.05) 분석가가 알아서 결정
- 유의확률(p-value): 컴퓨터가 알아서 주정

(3) 의사결정

기준	의사결정	해석
p-value \geq 유의수준(ex. 0.05)	대중주장 참	내가 수집/분석한 데이터는 단위근 있다 / 비정상 상태 / 시간의존 구조
p-value < 유의수준(ex. 0.05)	나의주장 참	내가 수집/분석한 데이터는 단위근 없다 / 정상 상태 / 시간의존 구조 아니다.

- [ADF-GLS test:](#)
 - 가설확인: ADF와 동일
- [Phillips-Perron\(PP\) test:](#)
 - 가설확인: ADF와 동일
- [Kwiatkowski Phillips Schmidt Shin\(KPSS\) test:](#) 주로 계절성 제거에 효과
 - 가설확인: ADF와 반대

4.4 정상성 이해실습

```
In [13]: # 라이브러리 호출
import pandas as pd
import numpy as np
import statsmodels.api as sm
from random import seed, random
import matplotlib.pyplot as plt
%reload_ext autoreload
%autoreload 2
from module_timeSeries import stationarity_adf_test, stationarity_kpss_test
```

```

# 랜덤워크 데이터 생성 및 통계량 Test (rho=0)
plt.figure(figsize=(10, 4))
seed(1)
rho = 0
random_walk = [-1 if random() < 0.5 else 1]
for i in range(1, 1000):
    white_noise = -1 if random() < 0.5 else 1
    value = rho * random_walk[i-1] + white_noise
    random_walk.append(value)
plt.plot(random_walk)
plt.title('Rho: {} ADF p-value: {}'.format(rho, np.ravel(stationarity_adf_test(random_walk, []))[1]))
plt.tight_layout()
plt.show()

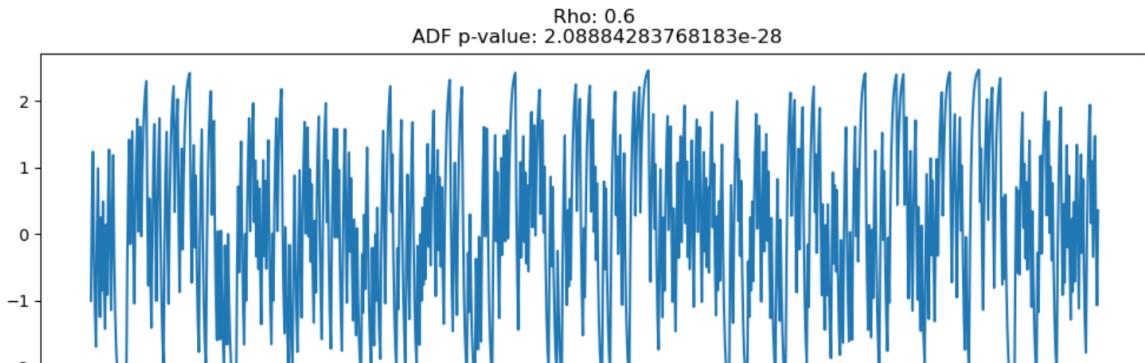
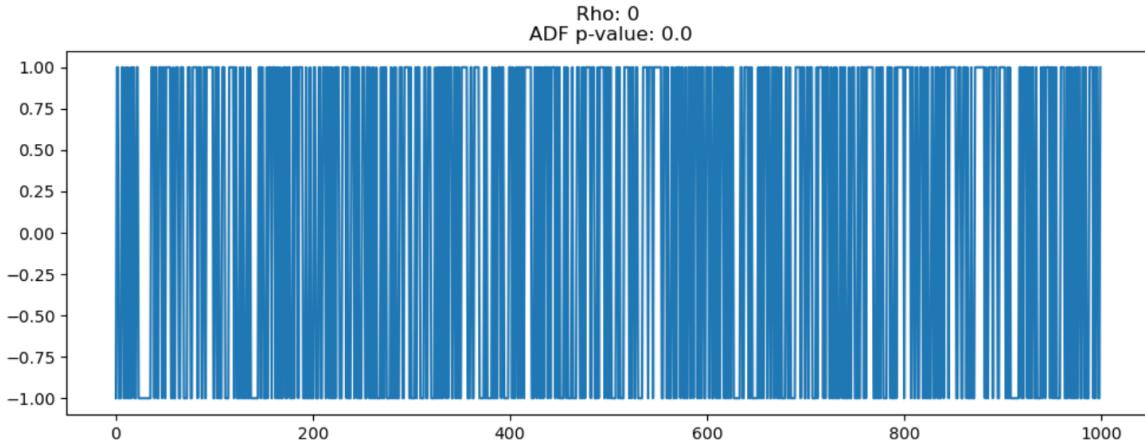
# 랜덤워크 데이터 생성 및 통계량 Test (rho=0.6)
plt.figure(figsize=(10, 4))
seed(1)
rho = 0.6
random_walk = [-1 if random() < 0.5 else 1]
for i in range(1, 1000):
    white_noise = -1 if random() < 0.5 else 1
    value = rho * random_walk[i-1] + white_noise
    random_walk.append(value)
plt.plot(random_walk)
plt.title('Rho: {} ADF p-value: {}'.format(rho, np.ravel(stationarity_adf_test(random_walk, []))[1]))
plt.tight_layout()
plt.show()

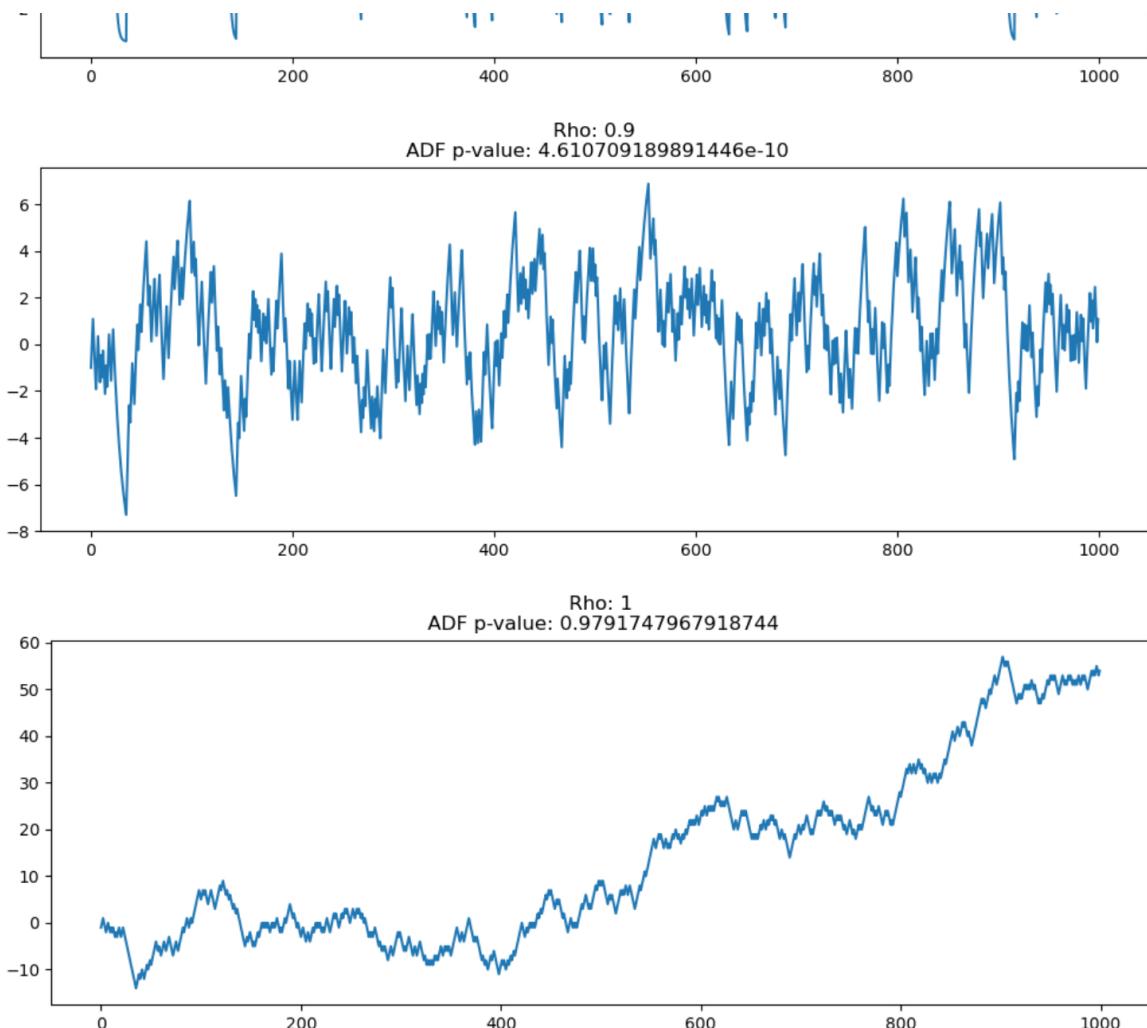
# 랜덤워크 데이터 생성 및 통계량 Test (rho=0.9)
plt.figure(figsize=(10, 4))
seed(1)
rho = 0.9
random_walk = [-1 if random() < 0.5 else 1]
for i in range(1, 1000):
    white_noise = -1 if random() < 0.5 else 1
    value = rho * random_walk[i-1] + white_noise
    random_walk.append(value)
plt.plot(random_walk)
plt.title('Rho: {} ADF p-value: {}'.format(rho, np.ravel(stationarity_adf_test(random_walk, []))[1]))
plt.tight_layout()
plt.show()

# 랜덤워크 데이터 생성 및 통계량 Test (rho=1)
plt.figure(figsize=(10, 4))
seed(1)
rho = 1
random_walk = [-1 if random() < 0.5 else 1]
for i in range(1, 1000):
    white_noise = -1 if random() < 0.5 else 1
    value = rho * random_walk[i-1] + white_noise
    random_walk.append(value)
plt.plot(random_walk)
plt.title('Rho: {} ADF p-value: {}'.format(rho, np.ravel(stationarity_adf_test(random_walk, []))[1]))
plt.tight_layout()
plt.show()

```

executed in 23.5s, finished 15:58:05 2022-06-11

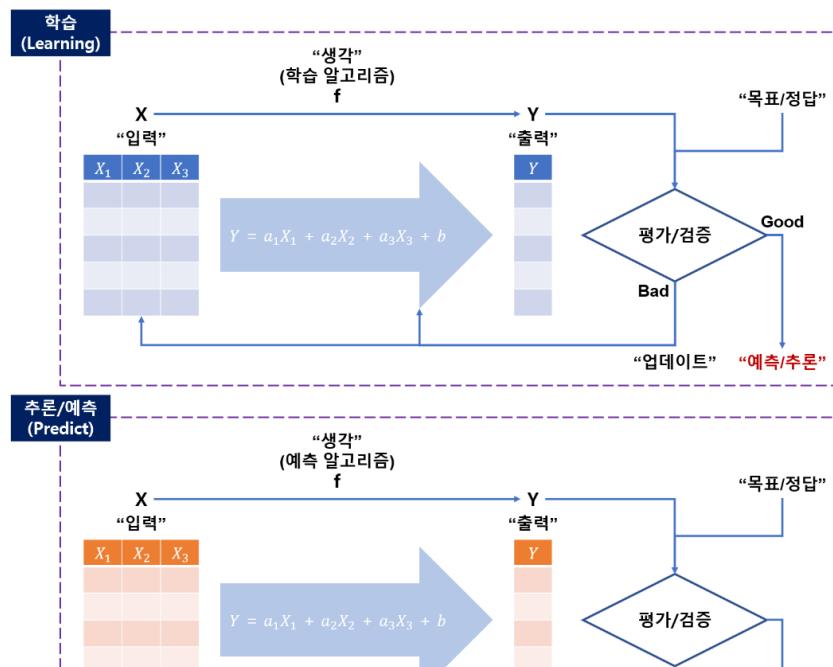




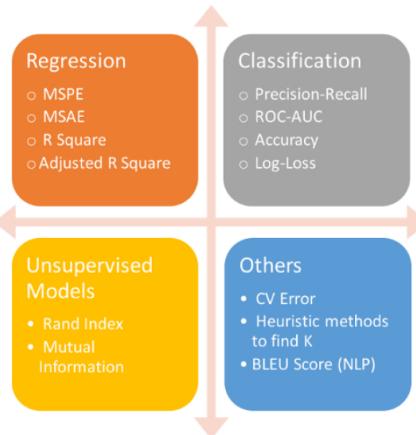
- 모델의 Y_{t-1} 계수가 1에 가까울수록 시계열 데이터는 평균에서 벗어나 랜덤워크로 변화
- 데이터의 계수가 1에 가까운지(있는지 없는지)를 체크하는 통계량 등장: Augmented Dickey-Fuller Test
- 계수가 1일 경우 수학적으로 단위근이 있다 라고 하며, 1을 제거하기 위해 차분하면 백색잡음 정상성 데이터로 변환

5 검증지표 방향(Evaluation Metrics)

5.1 대표적인 검증지표



1) 문제별 종류



- **Statistical Metrics:** Correlation

- 입력(Input): 무한대 ~ 무한대 범위의 연속형 값
- 출력(Output): 이론적으로 -1 ~ 1 범위의 연속형 값

- **Regression Metrics:** MSE, MSPE, RMSE, RMSLE, MAE, MAPE, MPE, R^2, Adjusted R^2, ... (Y의 범위가 무한대가 가능한 연속형일 때)

- 입력(Input): 무한대 ~ 무한대 범위의 연속형 값
- 출력(Output): 이론적으로 0 ~ 무한대 범위의 연속형 값

- **Classification Metrics:** Log Loss, Cross-entropy, ROC, AUC, Gini, Confusion Matrix, Accuracy, Precision, Recall, F1-score, Classification Report, KS Statistic, Concordant-Discordant Ratio, (ARI, NMI, AMI), ... (Y가 2개 또는 그 이상개수의 이산형일 때)

- 입력(Input): 무한대 ~ 무한대 범위의 연속형 값
- 출력(Output): 알고리즘 종류에 따라 출력이 달라질 수 있음

- 확률(Probability): 0 ~ 1 범위의 연속형 값 (Logistic Regression, Random Forest, Gradient Boosting, Adaboost, ...)
- 집단(Class): 0 또는 1의 이산형 값 (SVM, KNN, ...)

- **Clustering:** Dunn Index, Silhouette, ...

- **Ranking Metrics:** Gain, Lift, MRR, DCG, NDCG, ...

- **Computer Vision Metrics:** PSNR, SSIM, IoU, ...

- **NLP Metrics:** Perplexity, BLEU score, ...

- **Deep Learning Related Metrics:** Inception score, Frechet Inception distance, ...

- **Real Problem:** ???

5.2 회귀문제 검증지표

1) **R-squared(R^2):** 추정된 (선형)모형이 주어진 데이터에 잘 적합된 정도, 계량경제학에서는 모델에 의해 설명되는 데이터 분산의 정도(퍼센트), $(-\infty, 1]$

$$R^2 = \frac{ESS}{TSS} = \frac{\sum_{i=1}^n (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^n (y_i - \bar{y})^2} = 1 - \frac{RSS}{TSS} = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

```
import sklearn
sklearn.metrics.r2_score
```

2) **Mean Absolute Error(MAE):** 각 데이터의 단위에 걸 민감한 성능지표, $[0, +\infty]$

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

```
import sklearn
sklearn.metrics.mean_absolute_error
```

3) **Mean Squared Error(MSE):** 가장 많이 사용되며 큰 오차에 패널티(Penalty)를 높게 부여 하는 성능지표, $[0, +\infty)$

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

```
import sklearn
sklearn.metrics.mean_squared_error
```

4) **Mean Squared Logarithmic Error(MSLE):** MSE와 유사하나 큰값/작은값에 적은/많은 비중(Weight)을 부여하는 성능지표, Exponential 추세가 있는 데이터에 맞이 활용 $[0, +\infty)$

$$MSLE = \frac{1}{n} \sum_{i=1}^n (\log(1 + y_i) - \log(1 + \hat{y}_i))^2$$

```
import sklearn
sklearn.metrics.mean_squared_log_error
```

5) **Median Absolute Error(MedAE)**: 이상치에 둘 민감한(Robust) 성능지표, $[0, +\infty]$
 $MedAE = median(|y_1 - \hat{y}_1|, |y_2 - \hat{y}_2|, \dots, |y_n - \hat{y}_n|)$

```
import sklearn
sklearn.metrics.median_absolute_error
```

6) **Root Mean Squared Error(RMSE)**: MSE를 보정 한 성능지표, $[0, +\infty)$

$$RMSE = \sqrt{MSE}$$

7) **Mean Absolute Percentage Error(MAPE)**: MAE와 유사하나 퍼센트 형식으로 표시한 성능지표, $[0, +\infty)$

$$MAPE = \frac{100}{n} \sum_{i=1}^n \frac{|y_i - \hat{y}_i|}{y_i}$$

8) **Mean Percentage Error(MPE)**: MAPE와 유사하나 오차의 부호 (+/-)까지 고려 한 성능지표, $(-\infty, +\infty)$

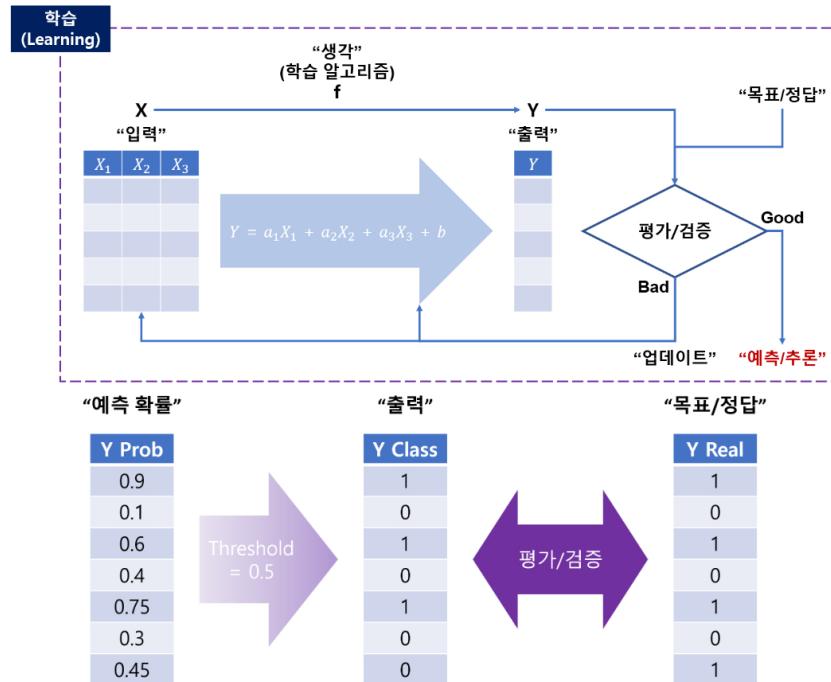
$$MPE = \frac{100}{n} \sum_{i=1}^n \frac{y_i - \hat{y}_i}{y_i}$$

- Summary:

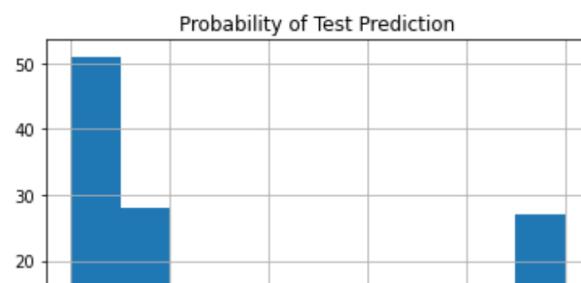
Acronym	Full Name	Residual Operation?	Robust To Outliers?
MAE	Mean Absolute Error	Absolute Value	Yes
MSE	Mean Squared Error	Square	No
MedAE	Median Absolute Error	Absolute Value	Yes
RMSE	Root Mean Squared Error	Square	No
MAPE	Mean Absolute Percentage Error	Absolute Value	Yes
MPE	Mean Percentage Error	N/A	Yes

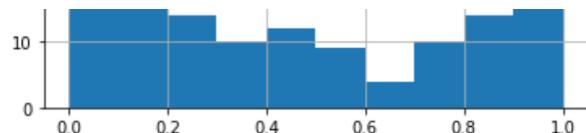
5.3 분류문제 검증지표

- Structure:



- 실제 예측결과 분포(히스토그램):





1) 오차행렬(Confusion Matrix): 정답 클래스와 알고리즘 예측 클래스의 일치 갯수 정리

- Binary Classification

	예측 0	예측 1
정답 0	정답이 0, 예측이 0인 데이터 수 (3)	정답이 0, 예측이 1인 데이터 수 (0)
정답 1	정답이 1, 예측이 0인 데이터 수 (1)	정답이 1, 예측이 1인 데이터 수 (3)

- Multi-class Classification

	예측 0	예측 1	...	예측 K
정답 0	정답 0, 예측 0인 데이터 수	정답 0, 예측 1인 데이터 수	...	정답 0, 예측 K인 데이터 수
정답 1	정답 1, 예측 0인 데이터 수	정답 1, 예측 1인 데이터 수	...	정답 1, 예측 K인 데이터 수
...
정답 K	정답 K, 예측 0인 데이터 수	정답 K, 예측 1인 데이터 수	...	정답 K, 예측 K인 데이터 수

2) 정확도(Accuracy): 전체 데이터 중 정확하게 예측한 클래스의 비율(0과 1을 모두 포함)

- 예측이 정답과 얼마나 정확한가?

	예측 0	예측 1
정답 0	True Negative (TN) (3)	False Positive (FP) (0)
정답 1	False Negative (FN) (1)	True Positive (TP) (3)

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN}$$

3) 정밀도(Precision): 클래스 1로 예측한 값들 중 실제 클래스 1의 비율

- 예측한 것중 정답의 비율은?
- 잘못예측한 클래스 1의 비중을 파악하고 줄이는데 목적
- 암환자 가 아닌데 암에 걸릴거라고 예측하여 과도한 사람들의 검진 증가 우려

	예측 1
정답 0	False Positive (FP) (0)
정답 1	True Positive (TP) (3)

$$\text{Precision} = \frac{TP}{TP + FP}$$

4) 재현율(Recall/Sensitivity/True Positive Rate): 실제 클래스 1 값들 중 예측 클래스 1의 비율

- 정답 클래스 1중 예측으로 맞춘 비율은?
- 잘못예측한 클래스 0의 비중을 파악하고 줄이는데 목적
- 암환자 인데 암이 아니라 예측하여 과도한 사망율 증가 우려

	예측 0	예측 1
정답 1	False Negative (FN) (1)	True Positive (TP) (3)

$$\text{Recall} = \text{TPR} = \frac{TP}{TP + FN}$$

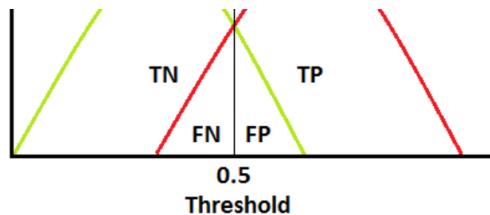
5) F1점수(F1-score): 정밀도와 재현율의 Trade Off 관계 반영위해 (가중)평균으로 모두 잘 맞추었는지 평가

- 정밀도와 재현율이 모두 중요한 문제의 경우 중요
- 정밀도와 재현율을 따로 보면 한쪽으로 편중된(Bias) 의사결정이 될 수 있는 위험
- 다양한 평균의 종류 중 조화평균을 사용하여 계산
- 정밀도와 재현율 중 한쪽에 치우치지 않았을 때 높은 값

$$\text{F1-score} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

6) ROC커브(Receiver Operator Characteristic Curve): 분류 기준값(Threshold)에 따라 재현율과 거짓율의 겸증지표의 변화를 확인하기 위한 시각화 지표





- 재현율(Recall/Sensitivity/True Positive Rate): 실제 클래스 1 값들 중 예측 클래스 1의 비율

- 정답 클래스 1중 예측으로 맞춘 비율은?
- 잘못예측한 클래스 0의 비율을 파악하고 줄이는데 목적
- 암환자 인데 암이 아니라 예측하여 과도한 사망률 증가 우려

	예측 0	예측 1
정답 1	False Negative (FN) (1)	True Positive (TP) (3)
정답 0	True Negative (TN) (3)	False Positive (FP) (0)

$$TPR = \frac{TP}{TP + FN}$$

- 거짓율(Fall-out/False Positive Rate): 실제 클래스 0 값들 중 예측 클래스 1의 비율

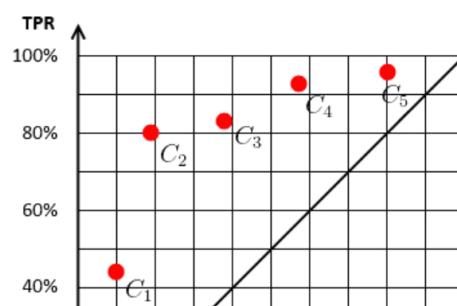
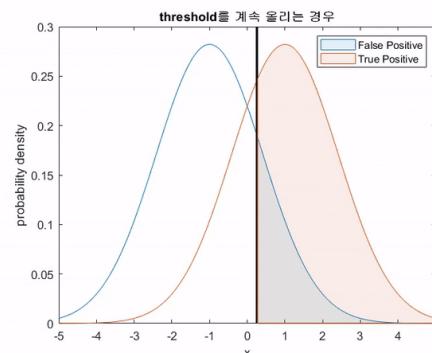
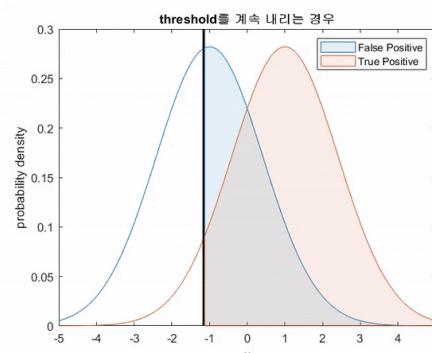
- 정답 클래스 0중 예측으로 틀린 비율은?
- 다른 Metrics와 달리 낮을 수록 좋음
- 재현율(TPR)과 거짓율(FPR)은 양의 상관관계 존재

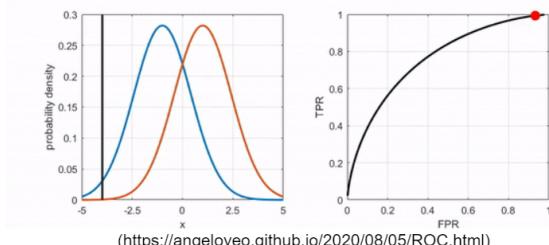
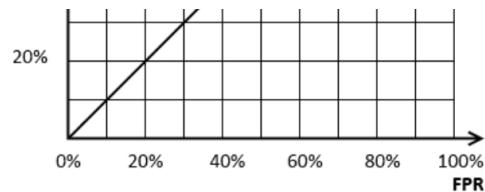
	예측 0	예측 1
정답 0	True Negative (TN) (3)	False Positive (FP) (0)
정답 1	False Negative (FN) (1)	True Positive (TP) (3)

$$FPR = \frac{FP}{FP + TN}$$

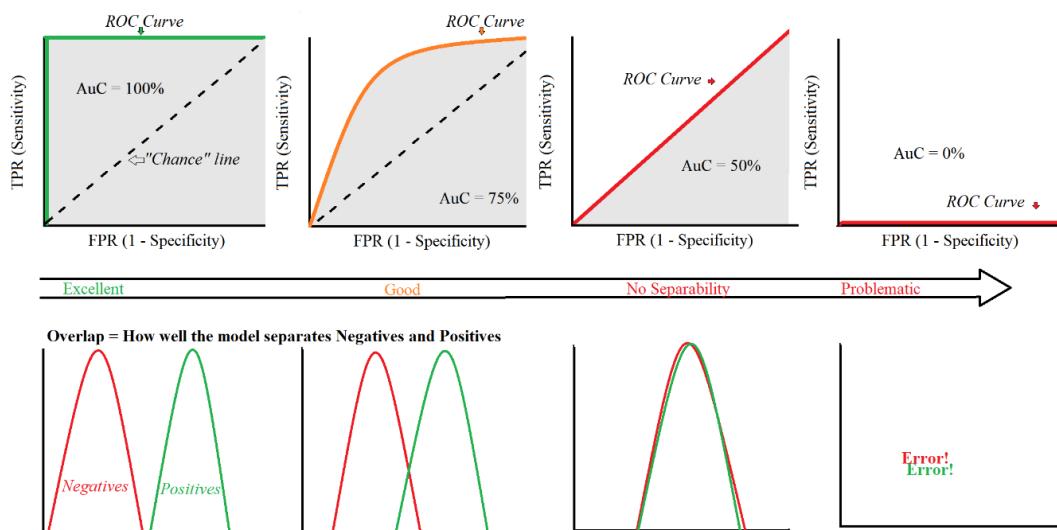
- RUC Curve: 재현율과 거짓율의 변화를 시각화

기준값 변화에 따른 변화:





예시:



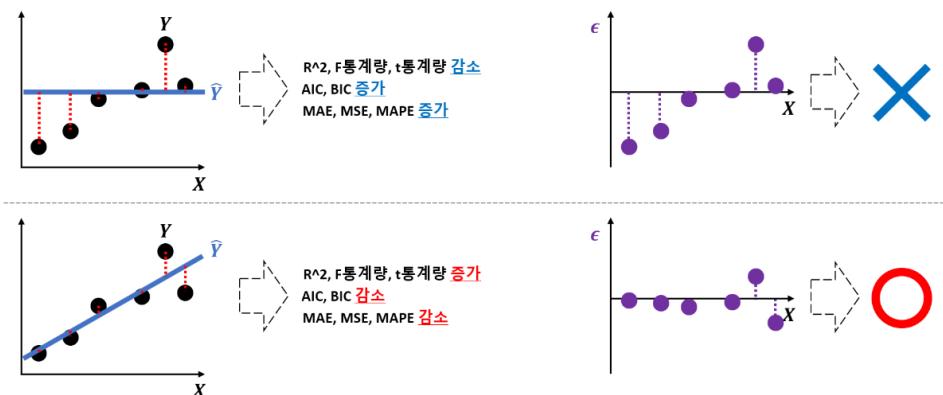
6 잔차진단 방향(Residuals Diagnostics)

0) 검증지표 vs 잔차진단: Y수치 비교가 가능한 예측문제에서 Y라벨이 있는 지도학습에 적용 가능

✓ 검증지표(Evaluation Metrics)
: 실제 Y 와 예측 \hat{Y} 이 얼마나 유사한지 측정

◀ 상호작용

✓ 잔차진단(Residual Diagnostics)
: $Y - \hat{Y} = \epsilon$ 에러/잔차에 남은 패턴이 없는지 측정



1) 잔차진단의 2가지 목적:

"예측 성능도 중요하지만(추정성능), 추정/분석 이후 데이터의 패턴이 모델링에 잘 반영되었는지 (잔차진단) 평가하는 것도 중요"

(1) 추가 할만한 데이터 전처리 또는 다른 모델링의 대안 파악

- 잔차에 남아있는 패턴을 전처리 단계에서 추가 반영 가능
- 잔차의 남은 패턴으로 다른 분석 알고리즘 고려 가능

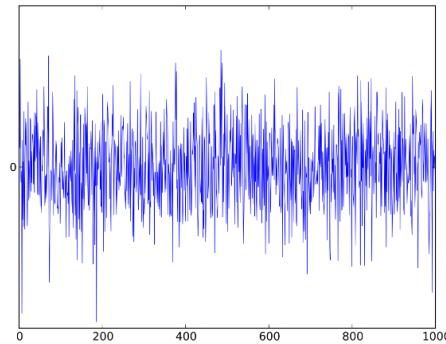
(2) 분석 시작은 여러분들이 시작했지만, 분석 종료는 잔차 진단이 알려줌

- 잔차의 남은 패턴이 없다는 것은 모델링이 데이터의 패턴을 최대한 반영 의미
- 모델링으로 더이상 할 수 있는 것들이 없으니 분석을 마무리 해도 될을 의미

2) 잔차진단의 목표: 잔차가 백색잡음 과 얼마나 유사한지 측정

- 모델링에 데이터의 패턴이 잘 반영되었다면, 추정 후의 잔차에는 아무 패턴도 없어야 함
- 잔차에 아무런 패턴도 남아있지 않은 경우 백색잡음의 형태로 분포
- 잔차 분석/진단을 통해 잔차가 백색잡음(White Noise)라면 역으로 모델링에서 데이터의 패턴을 잘 반영하여 성능이 좋음을 의미

3) 백색잡음: 잔차가 백색잡음이 아니라면 모델링으로 개선의 여지가 있음을 의미



(1) 잔차들은 정규분포이고, (unbiased) 평균 0이고 일정한 분산을 가져야 함:

$$\{\epsilon_t : t = \dots, -1, 0, 1, \dots\} \sim N(0, \sigma_{\epsilon_t}^2)$$

where $\epsilon_t \sim \text{i.i.d}(independent and identically distributed)$

$$\epsilon_t = Y_t - \hat{Y}_t$$

$$E(\epsilon_t) = 0$$

$$Var(\epsilon_t) = \sigma_{\epsilon_t}^2$$

$$Cov(\epsilon_s, \epsilon_k) = 0 \text{ for different times!}(s \neq k)$$

(2) 잔차들이 시간의 흐름에 따라 상관성이 없어야 함: 자기상관함수(Autocorrelation Function, ACF)=0 확인

- 공분산(Covariance):

$$Cov(Y_s, Y_k) = E[(Y_s - E(Y_s))(Y_k - E(Y_k))] = \gamma_{s,k}$$

- 자기상관함수(Autocorrelation Function):

$$Corr(Y_s, Y_k) = \frac{Cov(Y_s, Y_k)}{\sqrt{Var(Y_s)Var(Y_k)}} = \frac{\gamma_{s,k}}{\sqrt{\gamma_s \gamma_k}}$$

- 편자기상관함수(Partial Autocorrelation Function): s 와 k 사이의 상관성을 제거한 자기상관함수

$$Corr[(Y_s - \hat{Y}_s, Y_{s-t} - \hat{Y}_{s-t})] \text{ for } 1 < t < k$$

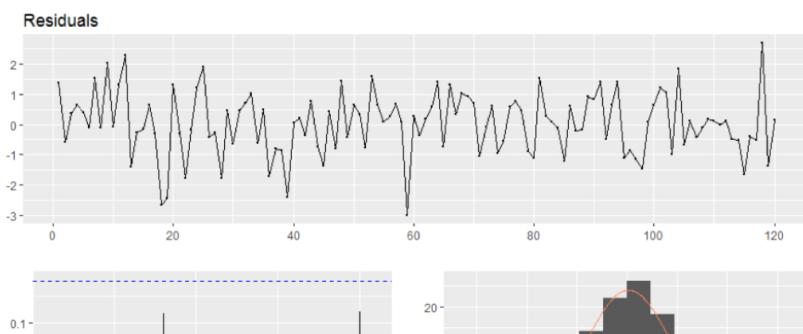
4) 자기상관 테스트:

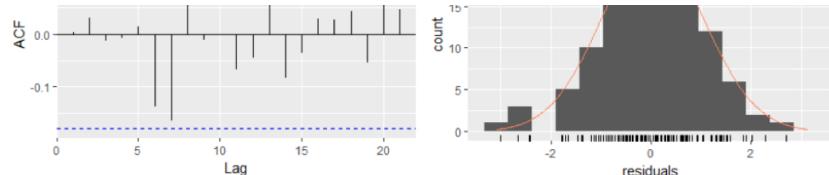
- 예시:

- Apply a portmanteau test to check the hypothesis that residuals are uncorrelated.

Model	Prob > chi2(40)	Portmanteau (Q) statistic
SARIMA	0.9221	28.0682
Naive	0.0000	162.3201
Brown Exponential Smoothing	0.0000	162.6615
Holt Exponential Smoothing	0.0000	192.8795
Holt Winters Exponential Smoothing	0.0000	182.2451
Decomposition	0.0000	126.005

- Plot the Autocorrelation function (ACF) and evaluate that at least 95% of the spikes are on the interval.

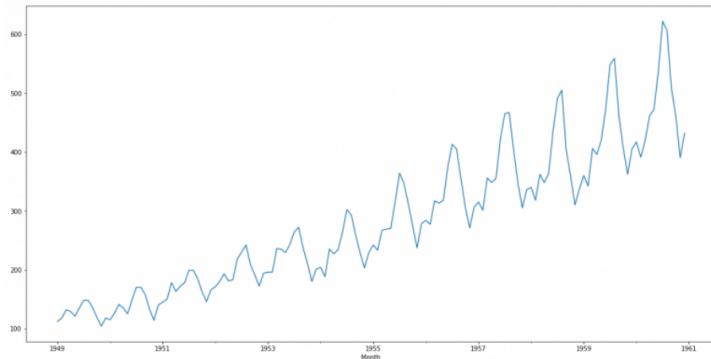




- 정규분포 테스트(Normality Test)
- 등분산성 테스트(Homoscedasticity Test)
- 자기상관 테스트(Autocorrelation Test)
- 정상성 테스트(Stationarity Test)

6.1 정상성 테스트(Stationarity Test)

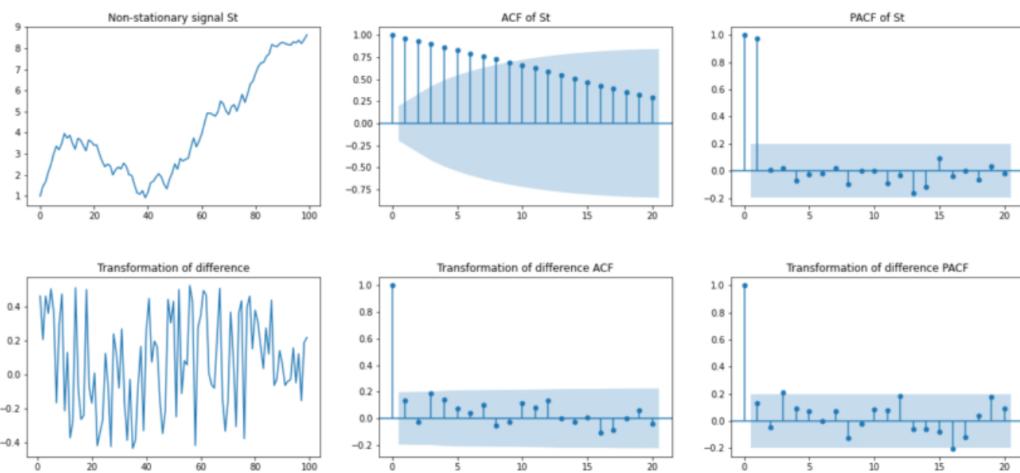
- 1) 기초통계 테스트(by Summary Statistics): 특정 시간(ex. 계절)에 따라 통계량이 일정한지 파악하여 정상성 확인
- 2) 시각화 테스트(by Visual): 추세와 계절성이 없는지 파악하여 정상성 확인



- 자기상관함수(Autocorrelation Function): 시차변화에 따른 자기상관을 의미

$$Corr(Y_s, Y_k) = \frac{Cov(Y_s, Y_k)}{\sqrt{Var(Y_s)Var(Y_k)}} = \frac{\gamma_{s,k}}{\sqrt{\gamma_s \gamma_k}}$$

- 만약 데이터가 정상시계열 이면, ACF 그래프가 모든 시차에서 0에 수렴하고,
- 비정상시계열 이면, ACF 그래프가 천천히 감소하거나 큰 양의 값



(<https://www.baeldung.com/cs/acf-pacf-plots-arma-modeling>)

- 3) 통계량 테스트(by Statistics Test): 정상성 테스트 통계량으로 정상성 확인

"In statistics, a unit root test tests whether a time series variable is non-stationary and possesses a unit root."

- Augmented Dickey-Fuller(ADF) test: 주로 추세제거에 효과

(1) 가설확인:

종류	해석
대중주장 (귀무가설, Null Hypothesis, H_0)	데이터는 단위근(Unit Root) 있다 / 비정상 상태 / 시간의존 구조
나의주장 (대립가설, Alternative Hypothesis, H_1)	데이터는 단위근 없다 / 정상 상태 / 시간의존 구조 아니다

(2) 유의수준 설정 및 유의확률 확인

- 유의수준: 5% (0.05) 분석가가 알아서 결정
- 유의확률(p-value): 컴퓨터가 알아서 주정

(3) 의사결정

기준	의사결정	해석
p-value \geq 유의수준(ex. 0.05)	대중주장 참	내가 수집/분석한 데이터는 단위근 있다 / 비정상 상태 / 시간의존 구조
p-value < 유의수준(ex. 0.05)	나의주장 참	내가 수집/분석한 데이터는 단위근 없다 / 정상 상태 / 시간의존 구조 아니다

• ADF-GLS test:

- 가설확인: ADF와 동일
- Phillips-Perron(PP) test:
 - 가설확인: ADF와 동일
- Kwiatkowski Phillips Schmidt Shin(KPSS) test: 주로 계절성 제거에 효과
 - 가설확인: ADF와 반대

• 예시:

Variable	Lag length	ADF statistic	Critical value 5%	Probability	Conclusion
None (Equations 6.1 and 7.1)					
Budget deficit as % GDP					
Level	1 (4)	2.307632	-1.960171	0.9923	Unit root
First difference	0 (4)	-6.119835	-1.960171	0.0000	No unit root
External debt as % GDP					
Level	0 (4)	-1.119552	-1.959071	0.3570	Unit root
First difference	0 (4)	-4.262988	-1.960171	0.0002	No unit root
Constant (Equations 6.2 and 7.2)					
Budget deficit as % GDP					
Level	1 (4)	1.175340	-3.029970	0.9966	Unit root
First difference	0 (4)	-7.058259	-3.029970	0.0000	No unit root
External debt as % GDP					
Level	0 (4)	-0.520473	-3.020686	0.8676	Unit root
First difference	0 (4)	-4.768799	-3.029970	0.0014	No unit root
Constant and linear time trend (Equations 6.3 and 7.3)					
Budget deficit as % GDP					
Level	0 (4)	-2.413843	-3.658446	0.3620	Unit root
First difference	0 (4)	-7.611239	-3.673616	0.0000	No unit root
External debt as % GDP					
Level	0 (4)	-0.520473	-3.020686	0.8676	Unit root
First difference	0 (4)	-4.752072	-3.029970	0.0066	No unit root

Variables	Order of Integration	ADF Test	Hypothesis
CPI	I(0)	0.99	Null hypothesis is not rejected
	I(1)	0.00	Null hypothesis is rejected
INS	I(0)	0.34	Null hypothesis is not rejected
	I(1)	0.00	Null hypothesis is rejected
TXT	I(0)	0.003	Null hypotheses is rejected
	I(1)	0.02	Null hypotheses is rejected
PNB	I(0)	0.07	Null hypothesis is not rejected
	I(1)	0.00	Null hypothesis is rejected
PETRO	I(0)	0.28	Null hypothesis is not rejected
	I(1)	0.00	Null hypothesis is rejected
OP	I(0)	0.28	Null hypothesis is not rejected
	I(1)	0.00	Null hypothesis is rejected
AUTO	I(0)	0.00	Null hypothesis is rejected

This table shows the results of stationary test that includes augmented dickey fuller (ADF) test.

	Variable	ADF Test Statistic	1% Critical Value	5% Critical Value	10% Critical Value	H0	Stationary
In Levels	Lcommer	0.46	-3.48	-2.88	-2.57	Accept	No
	Lindus	-2.19	-3.48	-2.88	-2.57	Accept	No
	Lnonen	1.31	-3.48	-2.88	-2.57	Accept	No
	Lpopen	-2.93	-3.48	-2.88	-2.57	Accept	No at 1%
	Lresiden	-2.31	-3.48	-2.88	-2.57	Accept	No
	Ltranspo	-2.19	-3.48	-2.88	-2.57	Accept	No
	Loilp	-0.20	-3.48	-2.88	-2.57	Accept	No
First Difference Form	D(Lcommer)	-5.69	-3.48	-2.88	-2.57	Reject	Yes
	D(Lindus)	-4.97	-3.48	-2.88	-2.57	Reject	Yes
	D(Lnonen)	-5.88	-3.48	-2.88	-2.57	Reject	Yes
	D(Lpopen)	-4.97	-3.48	-2.88	-2.57	Reject	Yes
	D(Lresiden)	-7.94	-3.48	-2.88	-2.57	Reject	Yes
	D(Ltranspo)	-1.46	-3.48	-2.88	-2.57	Accept	No
	D(Loilp)	-9.26	-3.48	-2.88	-2.57	Reject	Yes
Second Difference Form	DD(Ltranspo)	-9.57	-3.48	-2.88	-2.57	Reject	Yes

ADF test = augmented Dickey-Fuller test.

7. 삼고(Others)

7.1 데이터분석 정리

1) 배경: Train/Test 성능을 모두 높이면 좋겠지만 Test 향상 이 최선!

- 이상적: Train 예측성능 ↑ + Test 예측성능 ↑
- 현실적: Train 예측성능 << Test 예측성능

↔ 과적합을 줄인다 = 다중공선성을 줄인다 = 조건수가 낮아진다

2) 뭘 해야되지?

(1) 전처리 단계: 과적합을 줄인다 = 다중공선성을 줄인다 = 조건수가 낮아진다

- **VIF**: 독립변수들(X)끼리 회귀분석을 해서 독립성 높은 X 만 선택해주는데 계산량 많고 느림
- **PCA**: 독립성이 보장된 새로운 X 를 출력하는데 무슨 의미 데이터인지 알 수 없음
- **Scaling**: 독립변수들(X)의 크기를 일정하게 맞춰주면 일부 도움

(2) 모델링 단계: 현실성 높은 새로운 알고리즘 생성을 위해 비용함수를 새롭게 만든다

- **Regulaization Algorithm**: Ridge/Lasso 처럼 추정되는 계수(가중치) 크기를 작게 하기 위해 비용함수 변경
- 유사하게 비즈니스 또는 분석 목적에 맞는 신규 비용함수 생성

(3) 잔차진단 단계: 에러 분석 = 잔차 진단으로 에러를 White Noise로 만든다

- 비용함수를 새롭게 만들지 않고 할 수 있는 최선의 방법!
- 에러 분석하여 새로운 독립변수를 생성/삭제하거나 적절한 알고리즘으로 변경
- 종속변수를 정상성을 만족하도록 변경하여 예측성능을 올리고 에러를 줄임
- 정상성 데이터: 오래된 데이터는 가중치를 낮추고 최신 데이터는 가중치를 높임

7.2 분석에 도움 될 적정 시간빈도를 선택

"예측 정확성이 높은 시간영역을 선택 하는 것이 좋음"

$$\sum \hat{y}_t = \sum (y_t - e_t) = \sum y_t - \sum e_t \approx \sum y_t \Rightarrow \sum \hat{y}_t \approx \sum y_t$$

- 활용예시: 올해 비즈니스 매출 목표달성을 가능성 예측

- 가능한 한 모든 시간단위 별로 예측 모델링 실행
- 일반적으로 월별 또는 분기별 데이터를 사용하면 연간 데이터보다 나은 예측이 가능할 것
- 월/분기별 예측 후 연간으로 환산시 오류가 늘어날 것 같지만, 실제로는 예측 변동이 줄어들어 성능이 좋아지는 경우 많음
- 만약 너무 세분화된 시간영역을 사용할 시 오류가 증가될 수 있음
- 연간 비즈니스 목표를 예측하는데 일별/시간별/분별/이하 단위?의 데이터를 사용하면 도움이 될까?

7.3 시계열 데이터 관리

- 수천/수백만/수십억 데이터를 기계학습에 사용할 수 있지만, 시계열로 데이터를 정리하면 데이터 감소 발생 가능
- 모든 시간범위가 예측성능에 도움되지 않을 수 있기에 특정기간의 시간영역 분석만 필요할 수도 있음
- 고성능 시계열 Database 사용권장:

[Time Series Database\(TSDB\)](#)