데이터 분석 & 빅데이터

곽경일 강사

Visualization

1. Visualization

탐색적 데이터 분석 (EDA)

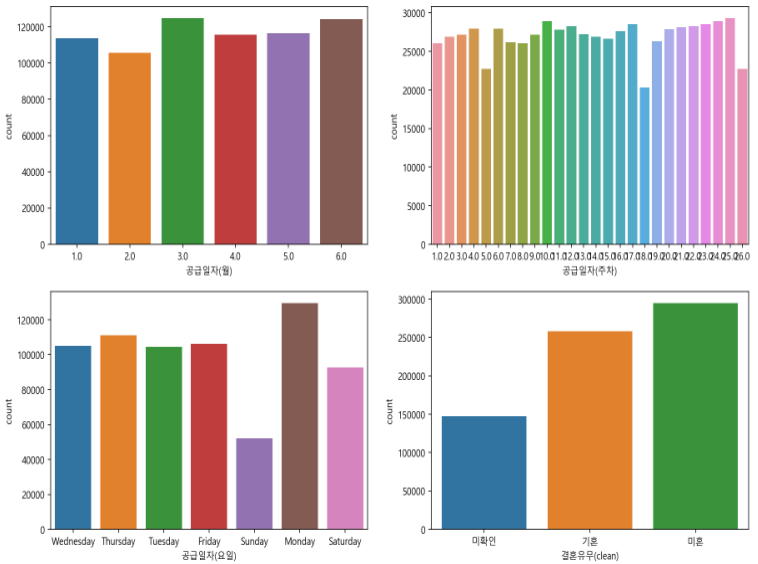
- 데이터를 시각적으로 분석하여, 데이터의 신뢰성, 경향성 확인함과 동시에, 분석의 방향성을 결정
- 데이터의 경향성을 확인하는 절차로써, **탐색적 분석 Exploratory Data Analysis (EDA)** 라고 부름
- 그래프를 이용해, 해당 데이터를 시각화하여 데이터의 분포나 트랜드, 변수간 관계 등을 개략적으로 확인
- 통계적 가설검정을 하지 않아도 쉽게 중요한 결과를 이끌어 낼 수 있음
- Python 대표 시각화 라이브러리 : Matplot / Seaborn

▪ 데이터 타입에 따른 그래프 시각화

- **연속형 데이터 : 연속형 데이터 간 상관관계, 또는 변화량, SPC 등**
  (Scatter plot(산점도), Density Plot, Smooth Plot…)

- **범주형 데이터 : 항목 또는 그룹간 비교, 비율, 순위형 자료에 대한 변화**
  (Box Plot, Bar Plot, Pie Chart, Heat Map …)

- **시계열 데이터 : 날짜나 요일 등 시간 단위 데이터에 대한 경향성, 작업 흐름에 관련된 분석**
  (Line Plot, Ribbon Plot, VEB Plot …)



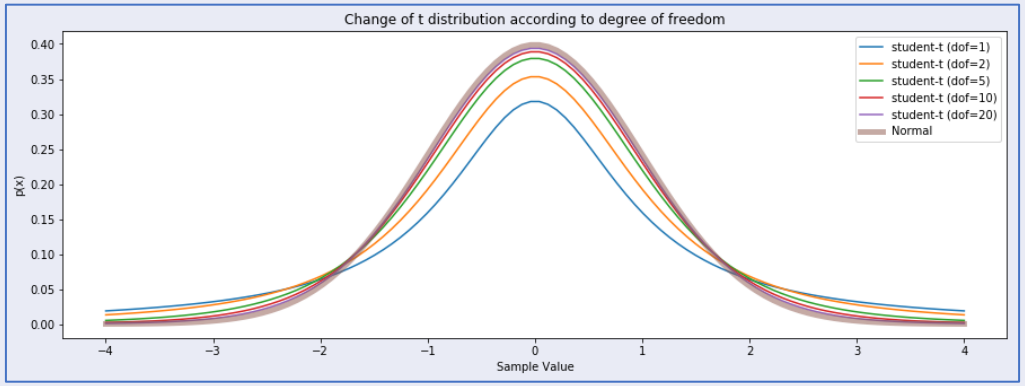
1. Visualization

탐색적 데이터 분석 (EDA)

- 데이터 타입에 따라, 분석에 사용되는 그래프가 달라짐
- 파악하고자 하는 문제나, 분석에 따라 사용되는 그래프를 적절히 선택하여 사용해야 함
- 통계분석 전 시각화를 통해, 분석이나 모델 생성의 방향성을 잡는데 유용
- **Matplot : 디테일한 시각화, 논문이나 레포트 자료에 사용. 시각화 옵션을 줄때 사용**
- **Seaborn : 빠른 시각화, 데이터 분석에 사용. 함수가 쉬우며 직관적인 결과를 확인**

▪ **그래프 분석의 종류**

1. Histogram
   - 1차원 Univariate 일변량
2. Box plot
   - 1차원 데이터 분산 파악
3. Bar plot
   - 범주형 데이터의 빈도 분포
4. Pie Chart
   - 각 범주 별 비율
5. Scatter Plot
   - 목표변수간 관계 해석



1. Visualization

▼ 시각화 옵션 설정

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt #그래프 출력시
import matplotlib as mpl # 그래프옵션

# 그래프 시각화 옵션 설정 함수
%matplotlib inline

# 그래프 한글설정
mpl.rc('font',family = "Malgun Gothic")

#그래프의 한글을 더욱 선명하게 출력
from IPython.display import set_matplotlib_formats
set_matplotlib_formats('retina')

#그래프에서 음수값이 나올때, 깨지는 현상 방지
mpl.rc('axes', unicode_minus =False)
```

Visualization

- Matplot 라이브러리를 이용해 시각화 옵션을 줄 수 있다.
- Jupyter Notebook 환경에서 한글이나, 음수 값에 대해 그래프 출력을 하려면 옵션을 설정해줘야 한다.
  (깨짐 현상 방지)
- 분석은 주로 Seaborn 라이브러리로 진행

2. Count Plot & Dist Plot

데이터 타입에 따른 시각화

- 데이터 타입에 따라 시각화의 방법과 종류가 달라짐
- 데이터 시각화를 할 때, 가장 먼저 단일 변수에 대한 시각화를 실시
- **데이터 타입이 연속형일 경우, 확률분포도와 Histogram을 확인**

  Histogram : 변수의 범위를 동일한 크기의 구간으로 나눈 다음, 각 구간마다 몇 개의 변수 값이 존재하는지 시각화

  확률분포도(probability distribution) : 확률변수가 특정 값을 가질 확률을 시각화

- **데이터 타입이 명목형일 경우, Count plot을 확인**

  Count plot : 범주형 데이터의 각 항목의 개수를 시각화

✓ 적용 (Count Plot)

1) 범주형 데이터에 대한 개수나 비율을 확인 할 수 있다 .
2) CRM 데이터, 웹 트래픽 데이터, 설문 데이터 등 방대한 영역에서 범주형 데이터를 확인할 때, 사용된다.

✓ 적용 (Dist Plot)

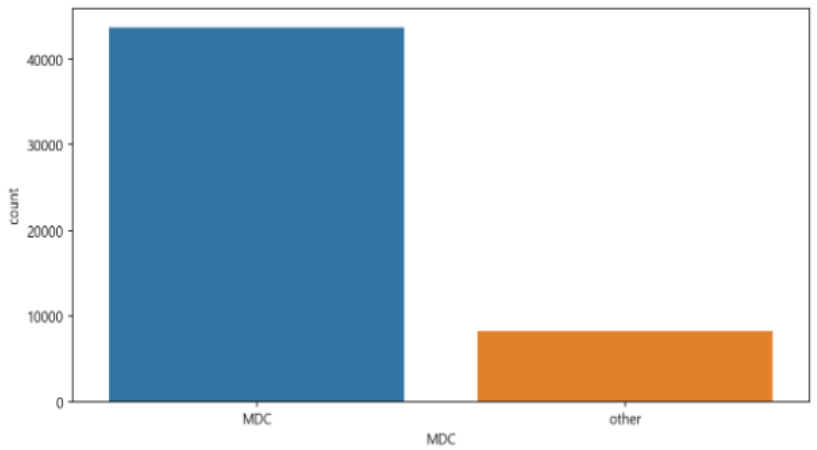
1) 도수분포를 시각적으로 확인 : 표로 되어있는 도수분포의 정보를 그래프로 표현. 밀도추정 그래프와 함께 사용된다.
2) 공업분야에서 품질관리(QC)를 위한 도구 중 하나
3) 여러 가지 패턴을 확인하여, 데이터의 신뢰성을 확인할 수 있다.

2. Count Plot & Dist Plot

▼ Count Plot

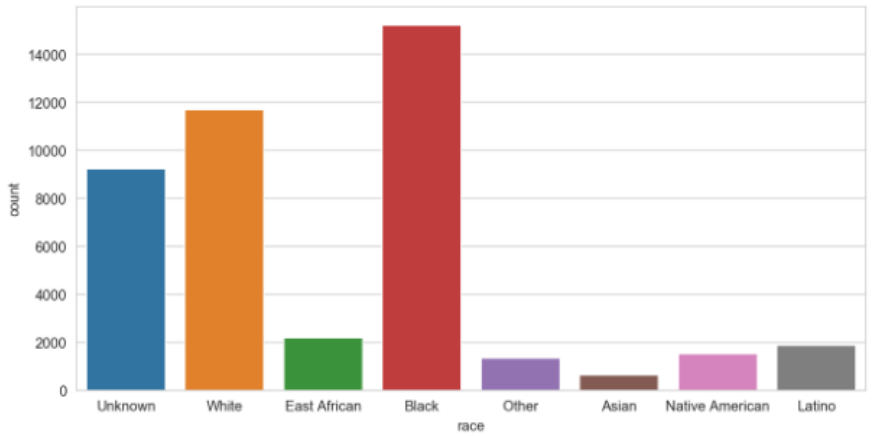
```
1 plt.figure(figsize = [10,5])
2 sns.countplot(data1['MDC'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c886810630>
```




```
1 sns.set_style('whitegrid')
2 plt.figure(figsize = [10,5])
3 sns.countplot(data1['race'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c8867b2a58>
```



Count Plot

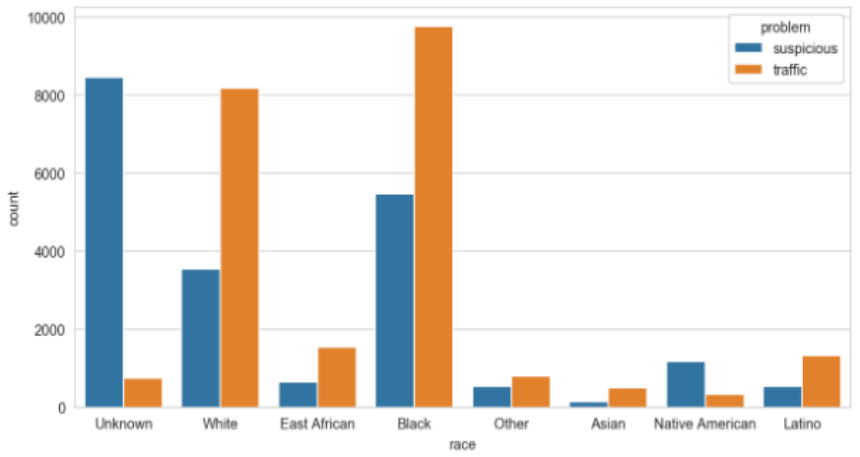
- sns.countplot(data['column']) : data라는 데이터 프레임의 명목형 변수 column에 대해 Count plot
- plt.figure(figsize= [n,m]) : 시각화 그래프의 크기를 n x m으로 설정해 준다. (단위는 inch)
- sns.set_style( ) : 특정 디자인의 그래프 바탕을 설정한다.

2. Count Plot & Dist Plot

▼ Count Plot

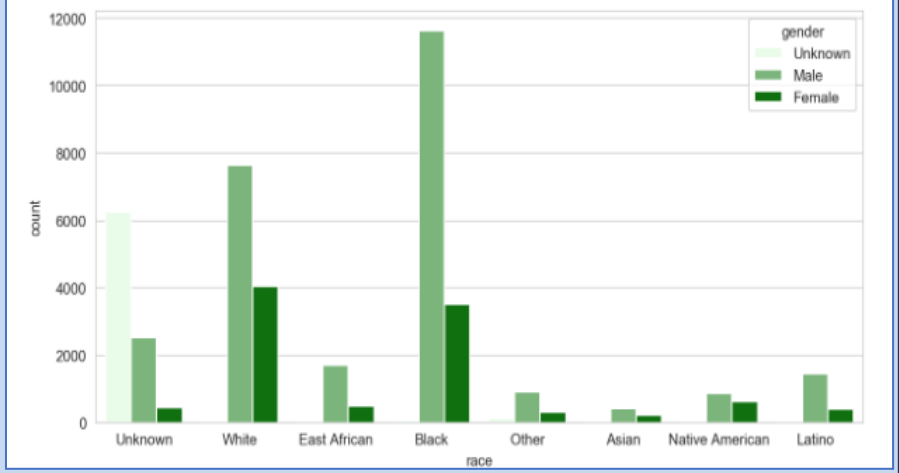
```
1 plt.figure(figsize = [10,5])
2 sns.countplot(data=data1, x='race',hue='problem')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c887cf9128>
```




```
1 plt.figure(figsize = [10,5])
2 sns.countplot(data=data1, x='race',hue='gender', color='g')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c888432d68>
```



Count Plot

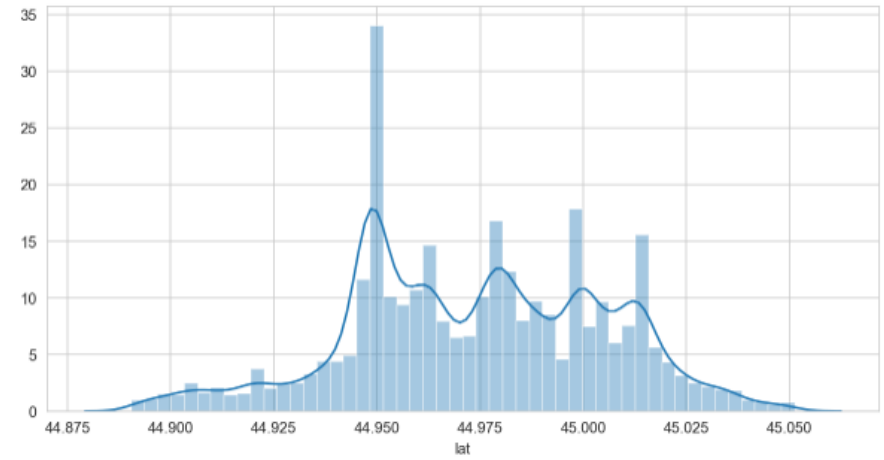
- hue : 다른 Column의 항목을 Overlay 중첩 시켜 시각화 시킨다
- color : 그래프 자체에 색상을 입혀줄 수 있다.

2. Count Plot & Dist Plot

▼ **Dist Plot**

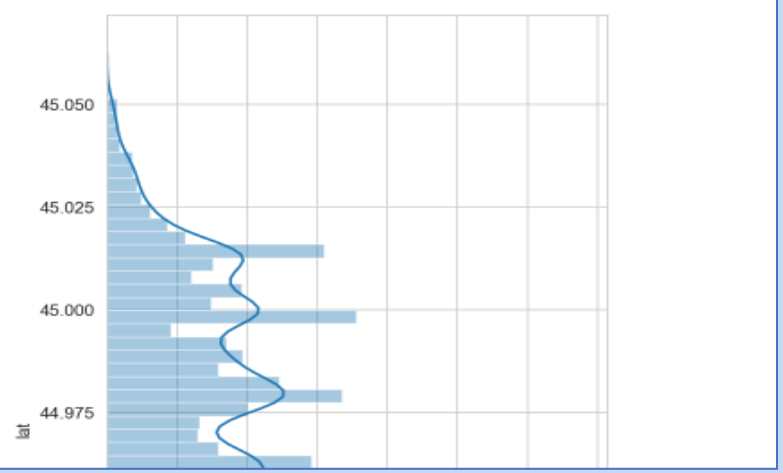
```
1  plt.figure(figsize = [10,5])
2  sns.distplot(data1['lat'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c88856ee48>
```




```
1  plt.figure(figsize = [5,10])
2  sns.distplot(data1['lat'],vertical=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c8889bc7b8>
```



Dist Plot

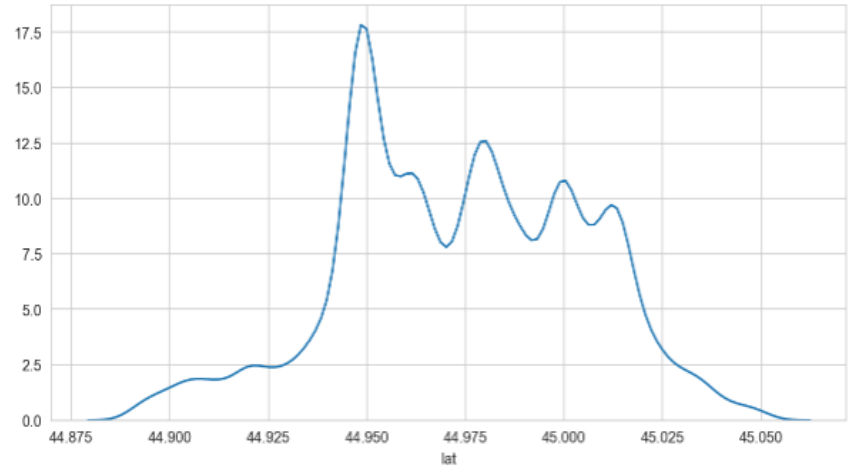
- sns.distplot(dataframe[‘column’]) : dataframe의 연속형 데이터 column에 대한 Dist Plot
- vertical 함수를 이용하면, 그래프를 Y축을 기준으로 그릴 수 있다.
- color함수도 동일하게 사용할 수 있다.

2. Count Plot & Dist Plot

▼ **Dist Plot**

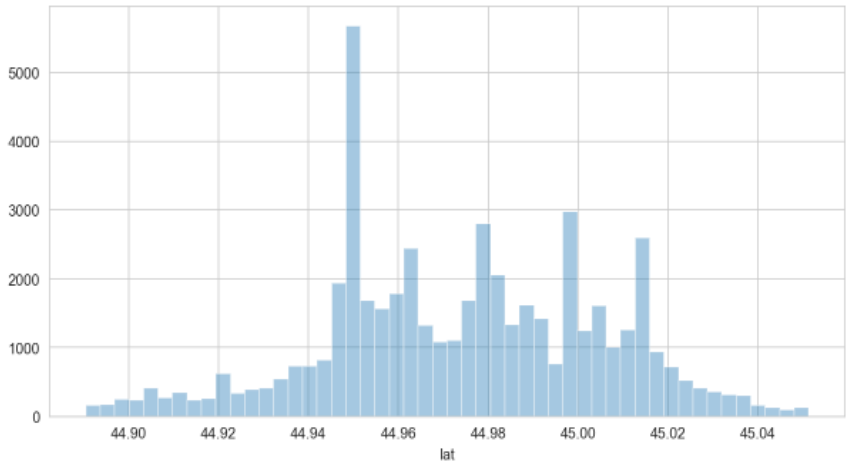
```
1 plt.figure(figsize = [10,5])
2 sns.distplot(data1['lat'], hist=False)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c88859edd8>
```




```
1 plt.figure(figsize = [10,5])
2 sns.distplot(data1['lat'],kde=False)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c8886769b0>
```



Dist Plot

- hist 함수를 이용해, Histogram을 제거할 수 있다.
- kde 함수를 이용해, 확률분포 선을 제거할 수 있다.
- hist 함수는 다른 범주형 항목들에 대해 중첩해서 사용이 가능하다.

3. Bar Plot & Box Plot

범주형 데이터 Vs 연속형 데이터

- 하나의 연속형 데이터의 IQR 값을 기준으로 데이터의 분포를 볼 때, Box Plot을 사용한다.
- **X 축 데이터에 범주형, Y 축 데이터에 연속형이** 들어갈 때, Bar Plot 또는 Box Plot을 사용한다.

  Bar Plot : 범주형 데이터를 표현하는 차트나 그래프로, 수직 또는 수평 막대그래프로 시각화

  Box Plot : 범주형 변수에 따라 분류된 연속형 변수의 분포를 시각화 하여 비교하는 방법

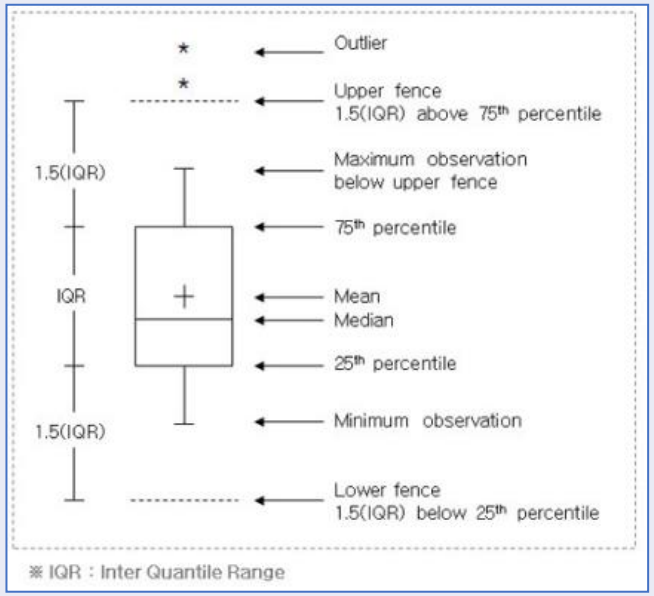
✓ 적용 (Bar Plot)

1) 범주형 자료에 대한 Counting , Category나 Class에 대한 비교를 할 때.
2) 순위형 자료에 대해 Counting 할 때에도 사용된다.
3) 여러 범주형 변수에 대한 Overlay를 쉽게 확인 할 수 있다.

✓ 적용 (Box Plot)

1) Five-number Summary : 5가지 주요 통계량이 시각적으로 표현
2) 여러 그룹 간 데이터를 비교할 때, 유용하다.
3) 데이터의 신뢰구간과 이상치를 빠르게 확인할 수 있다.

▼ Box Plot의 해석

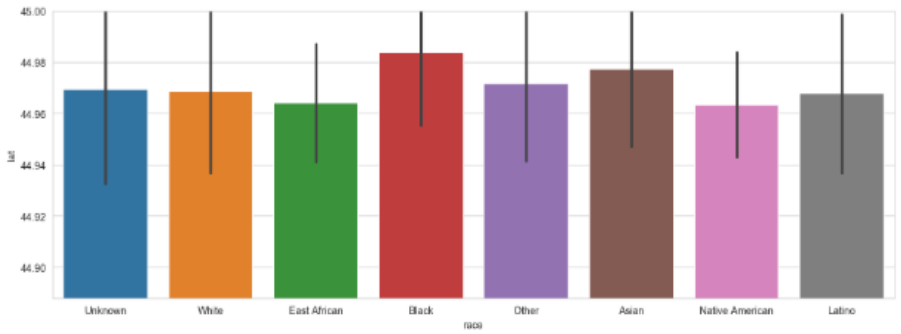


3. Bar Plot & Box Plot

▼ **Bar Plot**

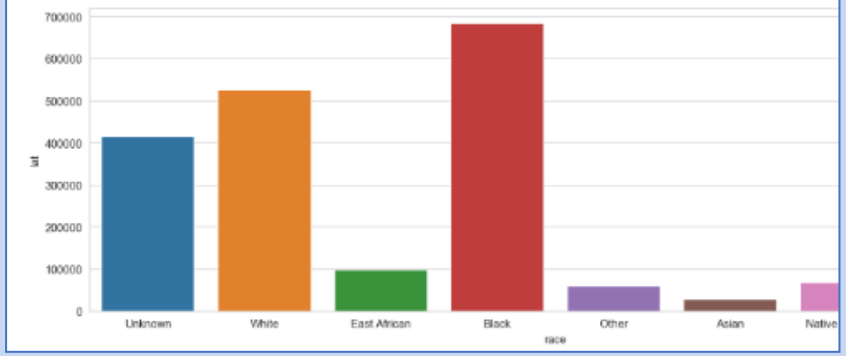
```
1 plt.figure(figsize = [15,5])
2 plt.ylim([44.888,45.0])
3 sns.barplot(data=data1, x='race',y='lat',ci='sd')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c889ff8ef0>
```




```
1 plt.figure(figsize = [15,5])
2 sns.barplot(data=data1, x='race',y='lat',estimator=sum)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c88a048f28>
```



Bar Plot

- sns.barplot(data=df1, x='col1', y='col2') :

  df1이라는 데이터프레임에 x값을 범주형 col1으로, y값을 연속형 col2로 넣어 box plot 실시

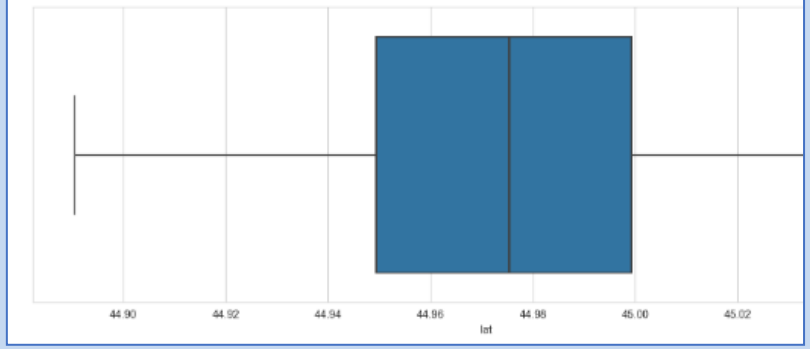
- Seaborn에서 Boxplot은 기본적으로 Box Plot위에 **신뢰구간**이 검정색 선으로 출력된다.
- ci 함수를 이용해, 신뢰구간을 표준편차로 바꿔줄 수 있다.
- estimator 함수를 이용해, 데이터 개수(counting)가 아닌 합을 구할수도 있다.

3. Bar Plot & Box Plot

▼ Box Plot

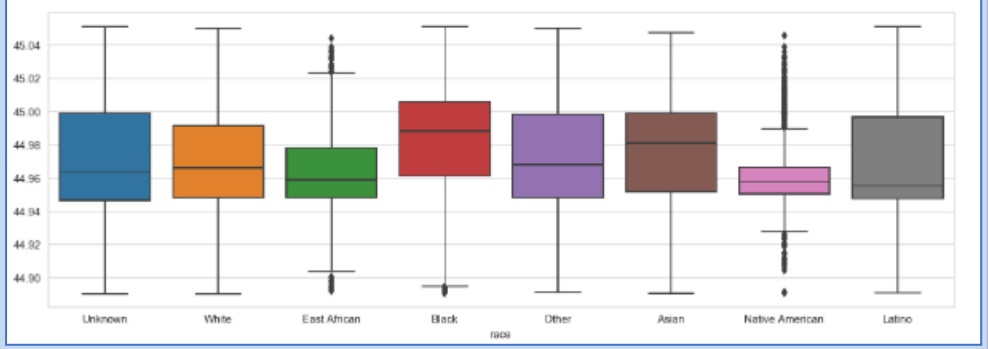
```
1 plt.figure(figsize = [15,5])
2 sns.boxplot(data=data1, x='lat')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c889f98b70>
```




```
1 plt.figure(figsize = [15,5])
2 sns.boxplot(data=data1, x='race',y='lat')
```

```
matplotlib.axes._subplots.AxesSubplot at 0x1c88a155e48>
```



Box Plot

- sns.boxplot(data=df1, x='col1' , y='col2') :

  df1 데이터에, 범주형 col1을 x값으로, 연속형 col2를 y값으로 boxplot 을 실행

- x에 연속형, y에 범주형 데이터를 넣으면, 그래프가 세로로 바뀌어 출력된다.
- hue 함수나 color함수가 동일하게 적용이 된다.

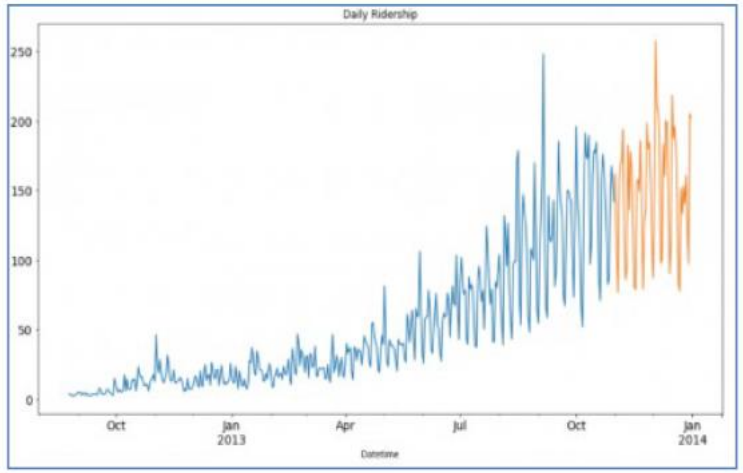
4. Point Plot & Line Plot

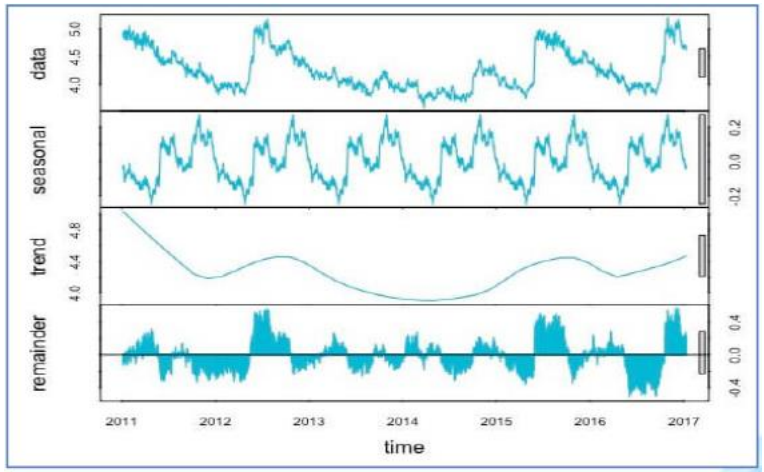
연속형 데이터 Vs 연속형 데이터 (시간)

- 시간의 흐름에 따라 여러 가지 지표를 동시에 볼 수 있다.
- Time Series 분석으로 많이 사용이 되며, 순위형 데이터에도 적용이 가능하다.

✓ 적용

1) 시즌성(Season)을 가진 데이터에 대해 경향성(Trend)를 확인
2) 설명 변수의 시간 흐름에 따른 각 지표의 관계를 규명
3) 지표 간 정보를 바탕으로 미래의 시나리오 예측
4) CRM데이터에서 시간에 따른 판매량 예측 / 공정데이터에서 시간에 따른 불량률 예측 / 의학, 사회적 데이터에서 연령에 따른 건강인자 값의 변화량 / 기상청 데이터 등 여러 분야에서 활용





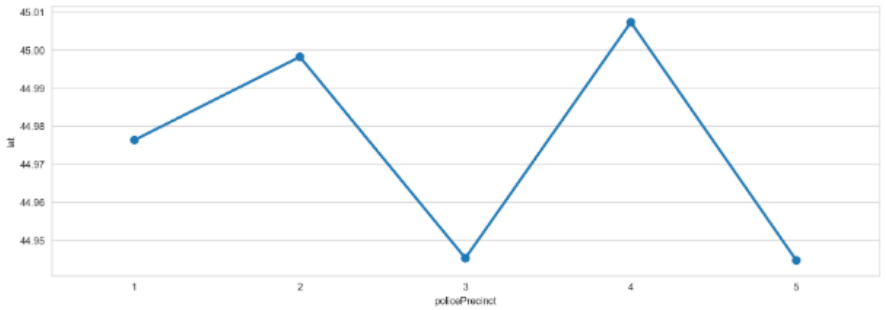
▲ 시간 데이터를 이용해, 해당 변수의 미래에 사건을 예측.

4. Point Plot & Line Plot

▼ Point Plot

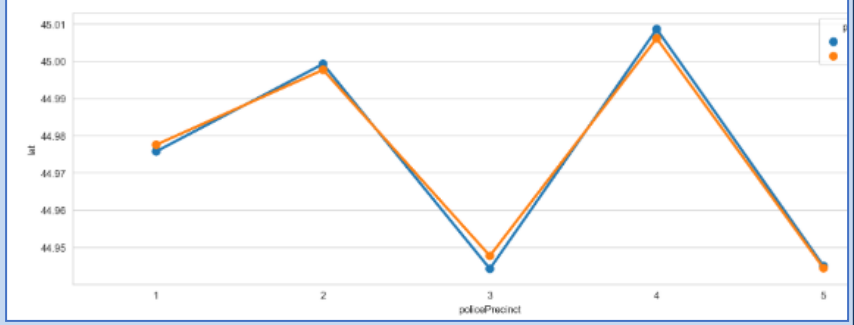
```
1 plt.figure(figsize = [15,5])
2 sns.pointplot(data=data1, x='policePrecinct',y='lat')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c888676a90>
```




```
1 plt.figure(figsize = [15,5])
2 sns.pointplot(data=data1, x='policePrecinct',y='lat',hue='problem')
```

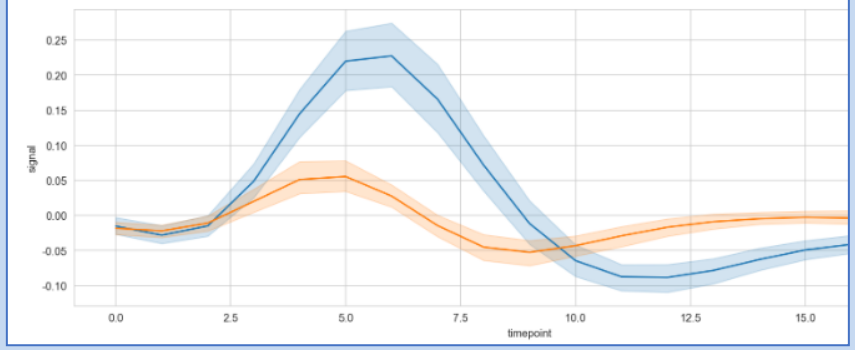
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c88a2713c8>
```



▼ Line Plot

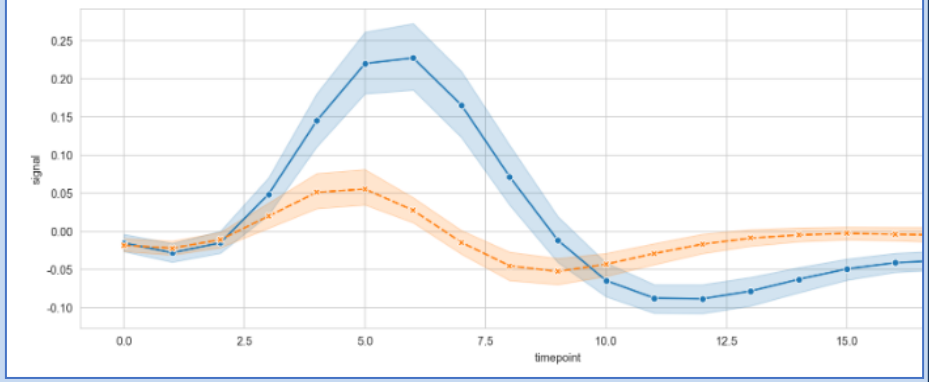
```
1 # Seaborn 라이브러리 내장 예제 데이터
2 fmri = sns.load_dataset('fmri')
3 plt.figure(figsize = [15,5])
4 sns.lineplot(data=fmri, x='timepoint',y='signal',hue='event')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c88a343c18>
```




```
1 plt.figure(figsize = [15,5])
2 sns.lineplot(data=fmri, x='timepoint',y='signal',hue='event',markers=True,style='event')
```

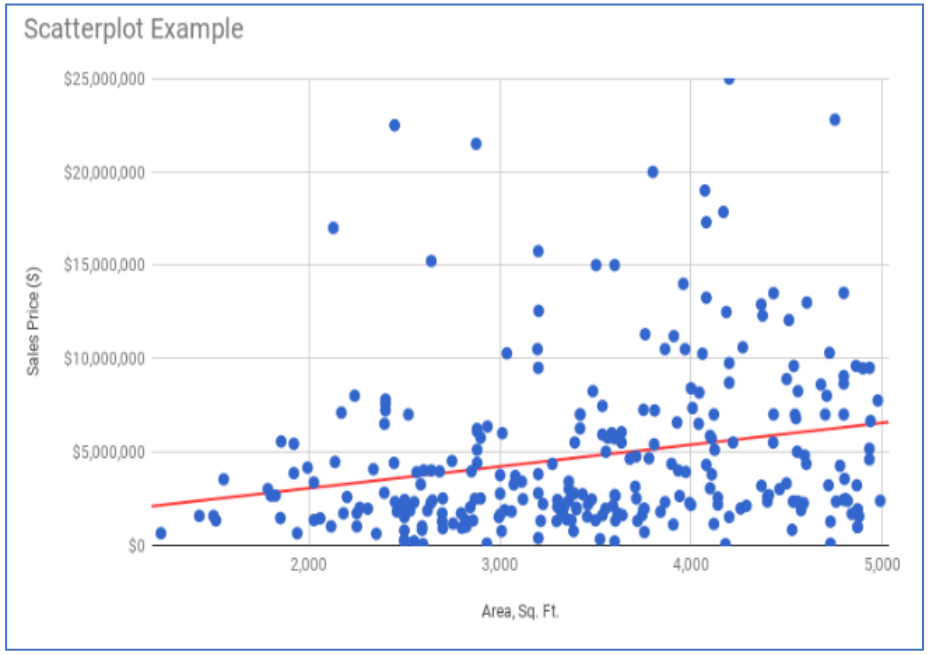
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c88a378438>
```



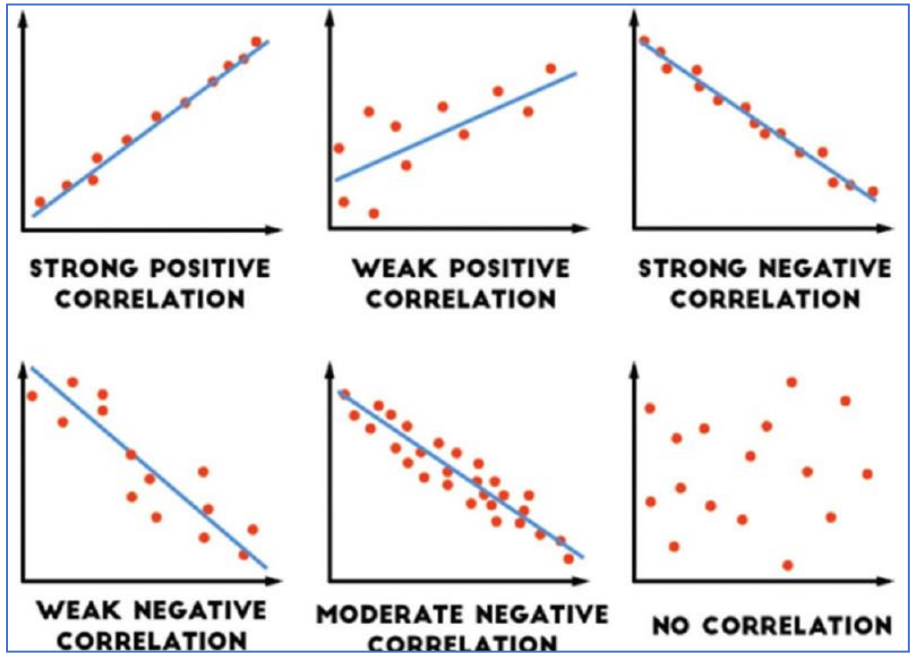
5. Scatter Plot

연속형 Vs 연속형

- 연속형 x에 대한 연속형 y 값의 지표를 시각적으로 표현
- **회귀분석이나 상관분석, 정규성 검정에** 반드시 확인하는 그래프 (잔차 그래프나 Pair Plot등 여러 분석에서 사용)
- 연속형 변수 간 관계를 대략적으로 파악할 수 있다.
- 목표변수와 설명 변수들 간 관계를 파악해 모델링에 대한 단서를 얻을 수 있다.



▲ 모델식을 생성하는 중요한 단서가 된다



▲ 상관 분석에 중요한 지표가 된다

5. Scatter Plot

▼ Scatter Plot

```
1 data = pd.read_csv('cell_img_data.csv')
2 data.head()
```

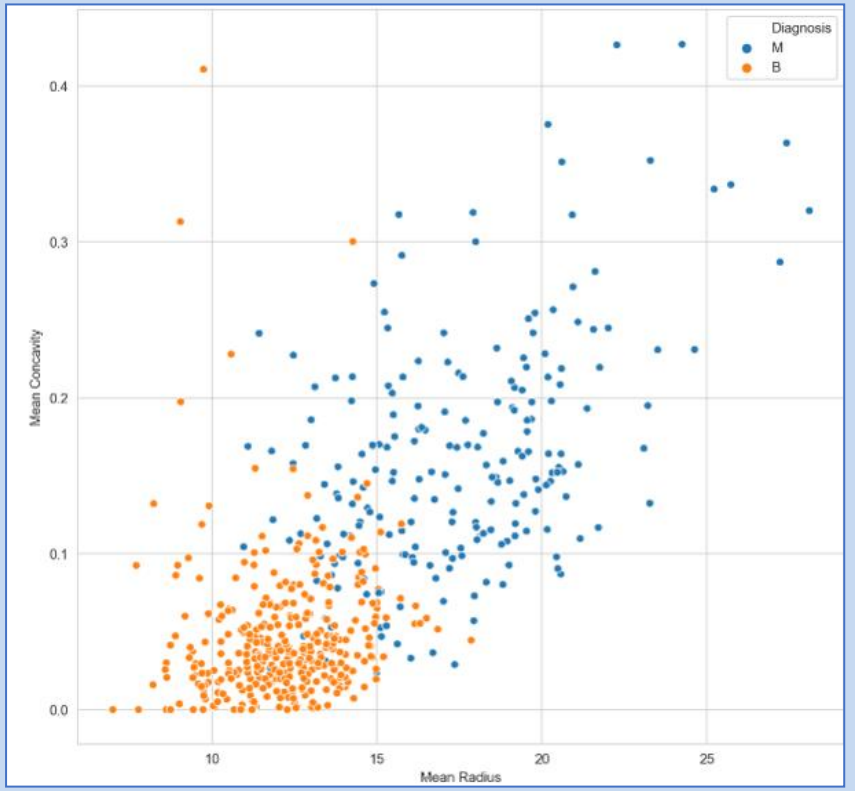
	Image ID	Diagnosis	Mean Radius	Mean Perimeter	Mean Area	Mean Texture	Mean Smoothness
0	842302	M	17.99	122.80	1001.0	10.38	0.12
1	842517	M	20.57	132.90	1326.0	17.77	0.08
2	84300903	M	19.69	130.00	1203.0	21.25	0.11
3	84348301	M	11.42	77.58	386.1	20.38	0.14
4	84358402	M	20.29	135.10	1297.0	14.34	0.10

5 rows × 32 columns

새로운 예제 데이터를 불러와 분석

```
1 plt.figure(figsize = [10,10])
2 sns.scatterplot(data=data, x='Mean Radius',y='Mean Concavity')
```

```
1 plt.figure(figsize = [10,10])
2 sns.scatterplot(data=data, x='Mean Radius',y='Mean Concavity',hue='Diagnosis')
```



Scatter Plot

- sns.scatterplot(data=df1, x='col1' , y='col2') :

df1 데이터에, x에 연속형 col1, y에 연속형 col2를 넣어 Plot

- hue 함수를 이용하여, Overlay를 할 수 있다 .
- 이후 Matplot Library를 이용해, 회귀식을 생성할 수 있다.

6. Machine Learning Introduction

1. 데이터 마이닝

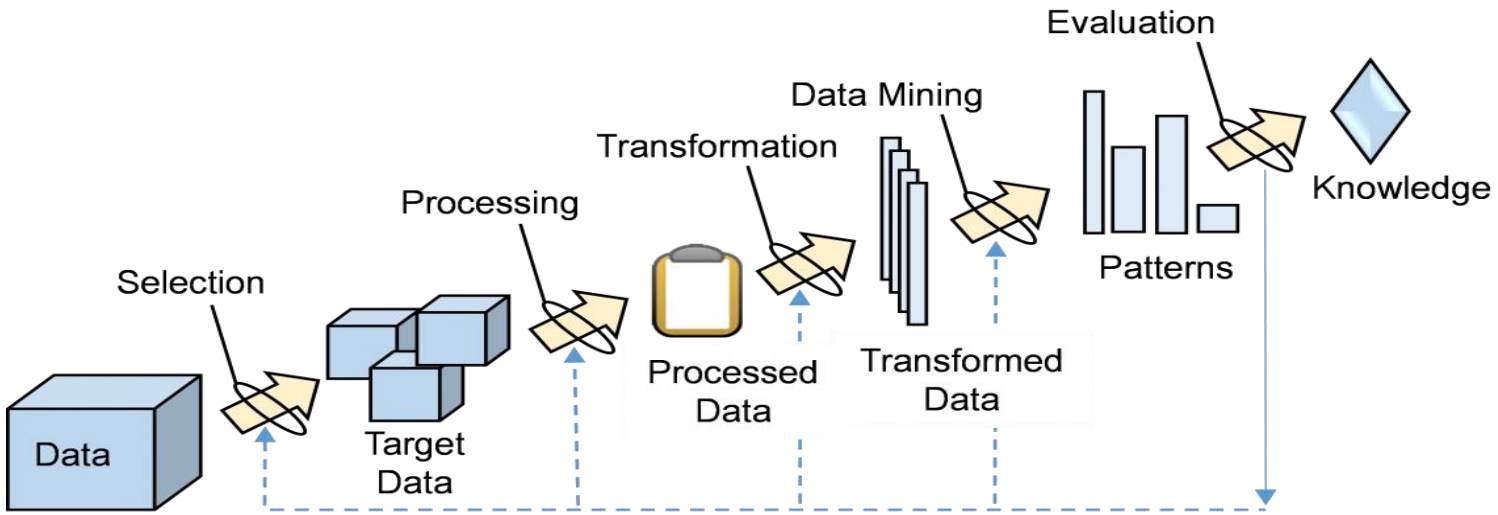
데이터 마이닝

- 기존의 전통적 통계분석에서, 컴퓨터 기술과 수학 알고리즘의 발달로 분석 매커니즘이 변화
- 모집단에서 표본을 추출하는 것이 아닌, 모집단을 그대로 분석에 사용
- 데이터 정제와 분할의 중요성이 높아짐
- 데이터의 양과 크기가 늘어남에도 기존에 사용된 회귀분석의 정확도가 유지됨

=> 기존의 분석기법을 개선하는 방향으로 알고리즘이 발전

=> 여러 형태의 데이터를 처리할 수 있는 알고리즘 개발

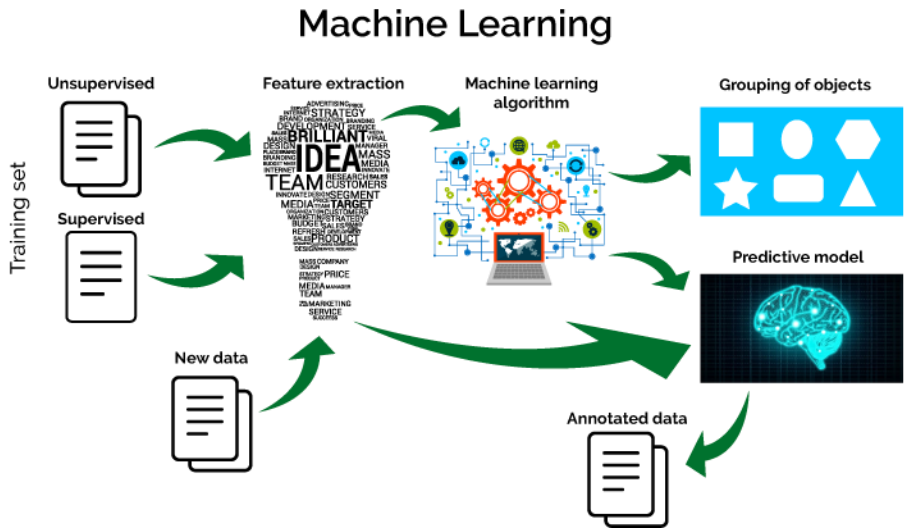
=> 해당 알고리즘이 데이터에 맞게 최적화 되는 방향으로 학습 (기계학습)



2. 기계학습의 이해

기계학습 (Machine Learning)

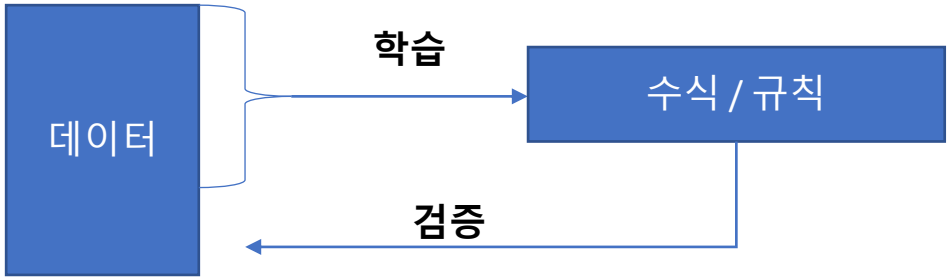
- 학습 : 좋은 결과를 얻기 위해, 어떤 지식이나 기술을 배우고 익힘
- 기계 학습 : 기계가 더 좋은 결과를 얻기 위해 특정 상황이나 데이터를 배우고 익힘
- 속성 :
  - 데이터 분석을 통한 새로운 지식 발견
  - 컴퓨터가 자체적으로 가장 최적의 분석모델을 발견할 수 있도록 알고리즘 학습
  - 새로운 데이터가 들어올 때 마다, 상황에 맞게 개선
  - 지도학습 / 비지도학습 / 강화학습의 3가지 학습이 존재



2. 기계학습의 이해

Machine Learning

- 기계 학습 : 컴퓨터가 데이터로부터 새로운 규칙/수식을 도출해내는 작업



1. **학습 능력** : 기존의 데이터를 잘 학습하여, 적절한 규칙을 도출해내는 능력
2. **일반화 능력 :** 새로운 데이터가 들어왔을 때, 정확한 값을 예측하여 대응하는 능력

- **기계 학습의 핵심 3요소 :**
- **데이터 (교과서)** : 데이터를 적절한 수식/규칙을 찾을 수 있도록 깔끔하게 다듬어야 함 **(특성공학)**
- **알고리즘 (선생님)** : 데이터의 종류에 따라 적절한 알고리즘을 선택
- 성능 (학생의 능력) : 컴퓨터의 하드웨어 능력

2. 기계학습의 이해

기계학습 (Machine Learning)

- **지도학습과 비지도학습은 레이블의 유무**
  (레이블 : 기계가 학습할 때, 참고할 수 있는 정답지)

- **지도학습 (Supervised Learning) :**
  목표변수(레이블)가 존재하여, 설명변수(X)와 목표변수(Y)의 관계를 파악해 학습
  => 회귀분석 / 분류분석

- **비지도학습 (Unsupervised Learning) :**
  목표변수(레이블)이 없이, 설명변수(X) 사이의 관계를 이용해 학습
  실제 기준에 의한 분류가 아닌 확률적, 조건적 분류
  => 군집분석 / 연관분석 / 차원축소

- **강화학습 (Reinforcement Learning) :**
  현재의 상태에서 다음의 최적의 행동을 찾아 학습 (State – Action)
  행동을 취할 때 마다 보상이 주어져, 보상을 최대화 하는 방향으로 학습
  => Deep Q Learning / Genetic Algorithm / 몬테카를로 알고리즘

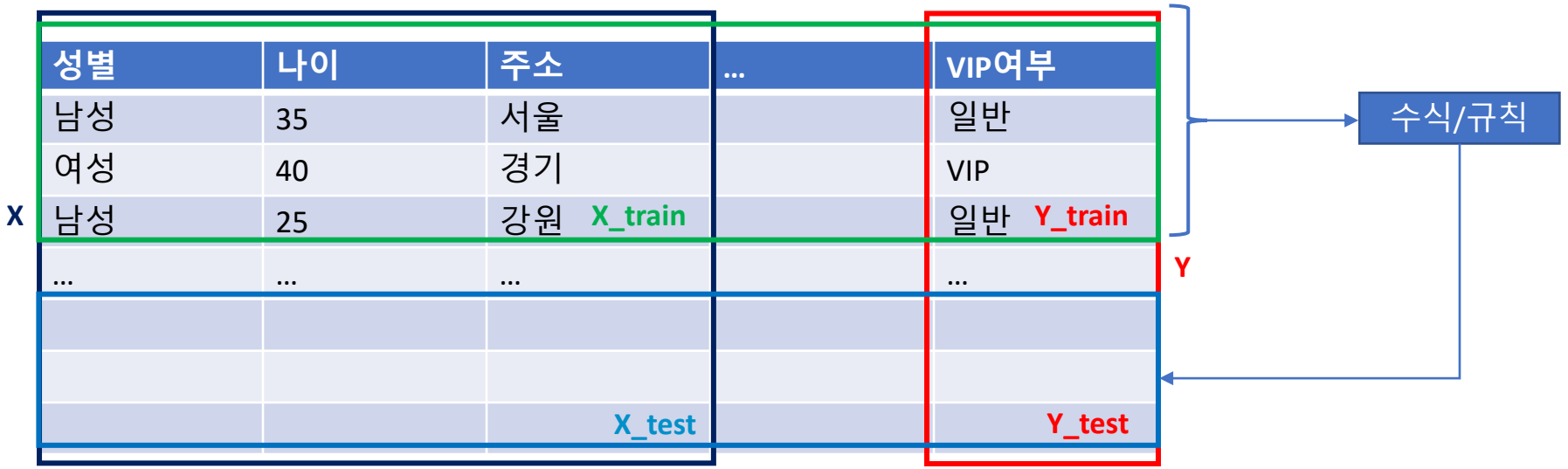
2. 기계학습의 이해

방법	분석구분	알고리즘
지도학습 (Supervised Learning)	분류분석 Classificaion	의사결정나무모델 (Decision Tree) 서포터벡터머신 (SVM) 로지스틱회귀분석 (Logistic Regression) 랜덤 포레스트 (Random Forest) 그레디언트 부스팅 (Gradient Boosting) KNN (K-Nearest Neighbors)
	회귀분석 Regression	선형회귀 (Linear Regression) Lasso Regression Riged Regression 시계열 (ARIMA Model)
	인공신경망 Artificial Neural Network	합성곱신경망 (CNN) 순환신경망 (RNN) DQN
비지도학습 (Unsupervised Learning)	군집화 / 연관분석 Clustering	K-Means 계층적 군집분석 비계층적 군집분석
	차원축소 Dimensionality	PCA ICA SVD NNMF
강화학습 (Reinforcement Learning)	몬테카를로 알고리즘 유전자 알고리즘 TD Learning	

2. 기계학습의 이해

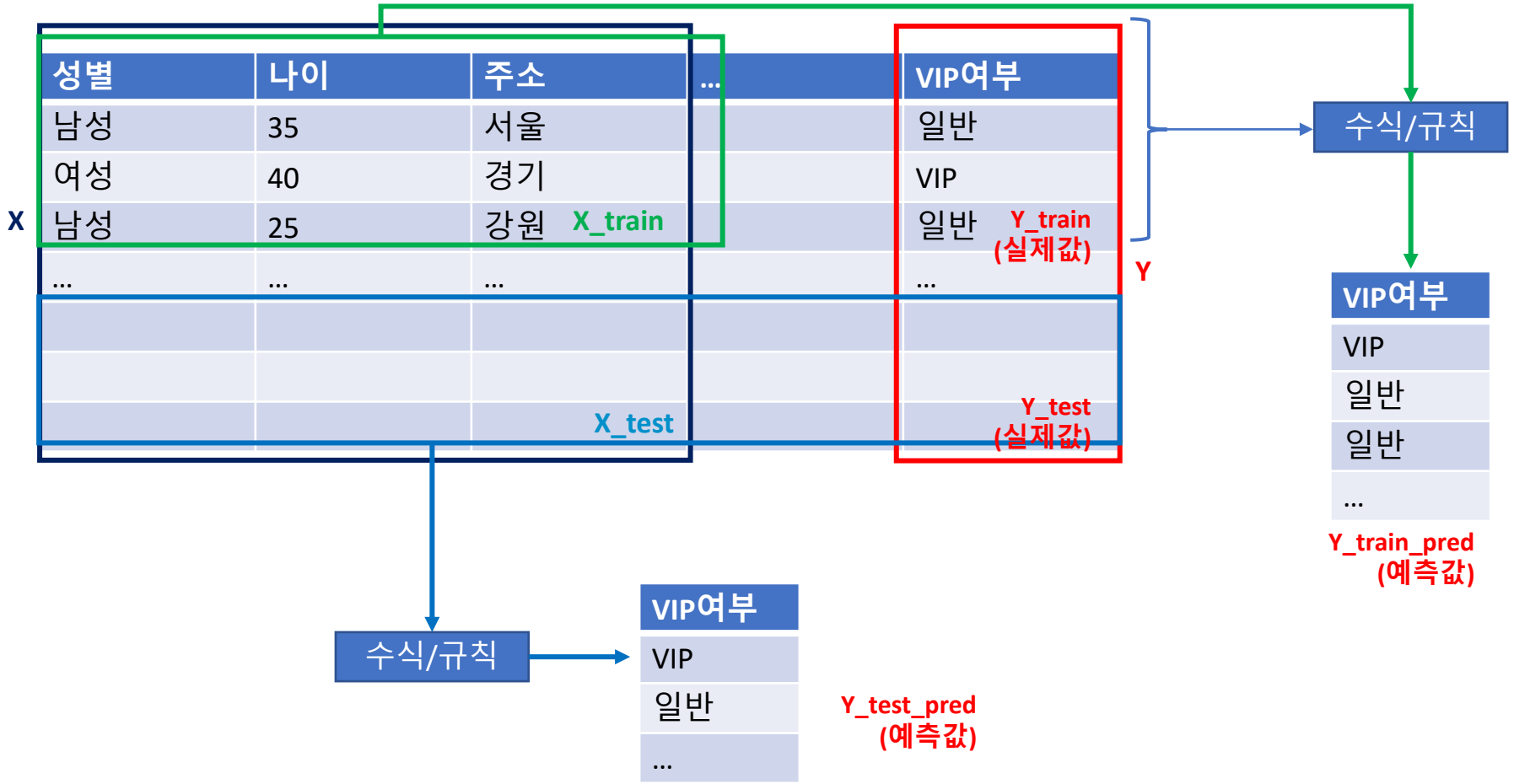
Machine Learning의 절차

1. 데이터의 결측치/이상치 제거,처리 (시각화, 가설검정 ...)
2. X(설명변수), Y(목표변수)를 선언
3. 학습데이터와 검증데이터를 분할
4. 학습데이터를 가져와, 알고리즘을 이용해 학습 실시
5. 검증데이터를 이용하여, 평가작업 실시
6. 새로운 데이터 적용



2. 기계학습의 이해

Machine Learning 모델 평가 방법



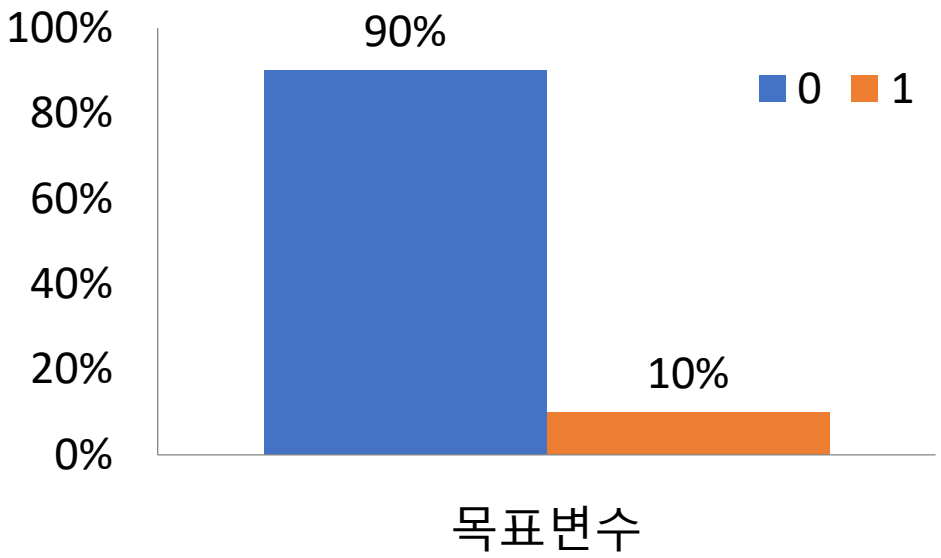
-실제값과 모델(수식/규칙)에 의해 계산된 예측값의 차이가 있다 => 학습이 잘 이뤄지지 않음

-실제값과 모델(수식/규칙)에 의해 계산된 예측값의 차이가 없다 => 학습이 잘 이뤄짐

1. 모델 평가 - 분류

모델 평가

- 훈련 데이터 셋으로부터 생성된 모델이 적절한 모델인지 판단하는 작업
- 보통 Accuracy(정확도)를 기준으로 판단하지만,
  실제 데이터를 다룰 때 정확도만 가지고 모델이 적절한지 판단이 어려움
- 특히 데이터 내 목표변수의 비중이 맞지 않을 때, 정확도가 높으나 모델이 좋지 않은 경우가 발생
- 정확도(Accuracy)이외의 Confusion Matrix나, Precision, Recall 값들을 확인해 모델을 평가



◀ 왼쪽과 같이 목표변수에서 찾고자 하는 값의 비율이 너무 적으면, 모든 데이터를 0으로 예측할 가능성이 높다.

그럴 경우 정확도는 매우 높으나, 실제 1 데이터를 정확히 예측할 확률은 낮아지게 된다.

2. 오차행렬 - 분류

Confusion Matrix

- 모델을 이용해 실제 데이터를 예측한 값과 실제 데이터 값을 비교하여 만든 행렬
- 오차행렬(Confusion Matrix)값을 이용해, 정확도(Accuracy), 정밀도(Precision), 재현률(Recall)을 계산
- 이진 분류 모델에서, 찾아야 하는 적은 수의 결과값에 Positive (1), 나머지 대다수 값에 Negative (0) 부여
- 불균형한 이진 데이터에서는 Negative 쪽으로 예측 정확도가 높게 나타나는 경향이 있다.

		Predicted class: P	Predicted class: N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

▼ 오차행렬을 이용해 계산된 모델 성능 지표

$$precision = \frac{TP}{TP + FP}$$

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

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

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

3. 정밀도 & 재현율 - 분류

Precision & Recall

정밀도 (Precision)

- 예측을 Positive (1)로 한 대상 중에 예측값과 실측값이 Positive(1)로 일치한 데이터의 비율
- Positive 예측 성능을 더욱 정밀하게 측정하기 위한 평가 지표 (양측 예측도)
- 문제가 없는 데이터를 문제가 있다고 잘못 판단할 때 발생하는 이슈를 나타내는 지표
- False Positive를 낮추는데 초점을 둠

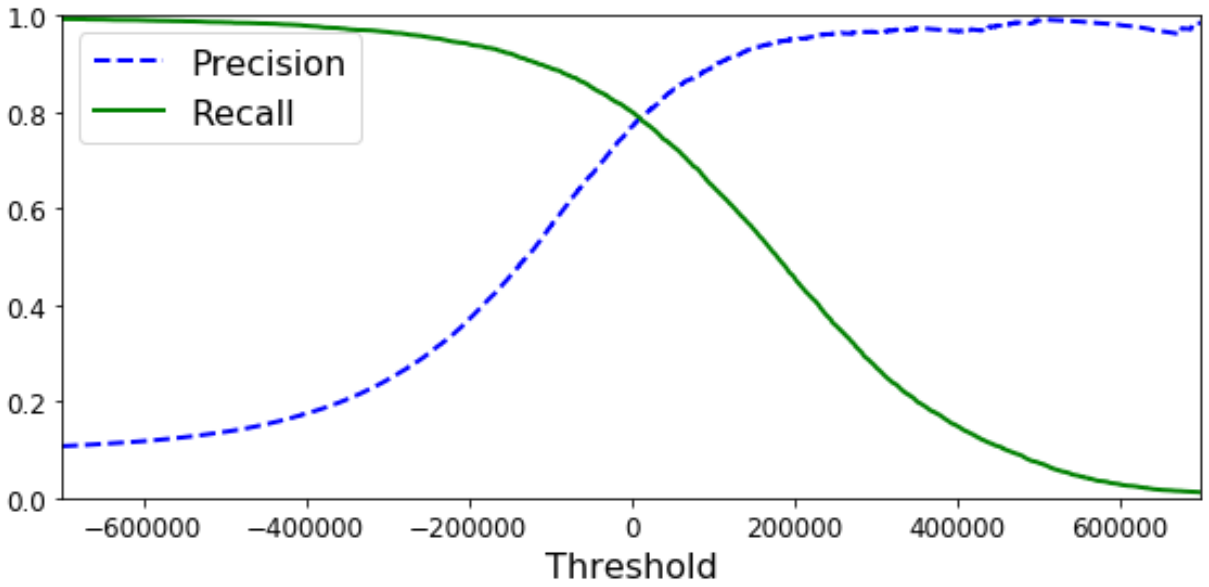
재현율 (Recall)

- 실제값이 Positive (1)인 대상 중에 예측값과 실측값이 Positive(1)로 일치한 데이터의 비율
- 민감도 (Sensitivity) 또는 TPR(True Positive Rate)라고 불림
- 실제 문제가 있는 데이터를 문제가 없다고 잘못 판단할 때 발생하는 이슈를 나타내는 지표
- False Negative를 낮추는데 초점을 둠

3. 정밀도 & 재현율 - 분류

Precision & Recall Trade - Off

- Sklearn의 predict_proba( ) 함수를 이용해, Model에 데이터를 넣었을 때, Positive나 Negative가 될 확률을 구할 수 있다.
- 보통 0.5 (50%)를 기준으로 Positive와 Negative가 결정되지만, Threshold를 낮추면, Positive가 나올 확률이 낮아도 Positive로 판단하게 된다.
- 정밀도나 재현율을 특별히 강조해야 할 경우, Threshold를 조정하여, 정밀도 또는 재현율을 높일 수 있다
- 그러나 한쪽을 강제로 높이면, 다른 한쪽이 낮아지기 때문에, (Trade-Off) 적절한 Threshold를 찾는 것이 중요하다.



4. F1 Score - 분류

F1 Score

- F1 Score : 정밀도와 재현률을 하나의 지표로 결합하여, 모델을 평가
- F1 Score값이 높을 수록 정밀도와 재현률이 높은 모델
- Threshold를 조절하여, 최적의 F1 Score를 찾을 수 있음
- 그러나 찾은 Threshold값이 목표변수에 논리적으로 알맞은 값이 되어야 함
- Threshold값에 따른 Accuracy, Precision, Recall, F1 Score를 계산하는 함수를 제작하여, 모델을 평가 및 개선할 수 있음

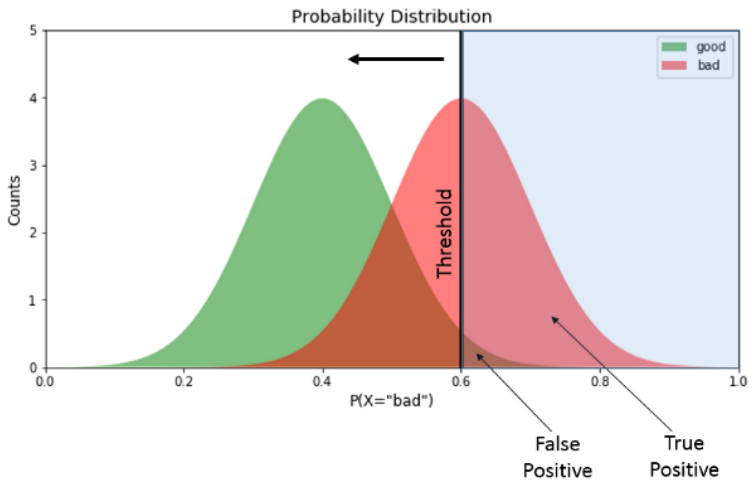
$$\text{F1-score} \triangleq 2\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

5. ROC & AUC - 분류

ROC & AUC

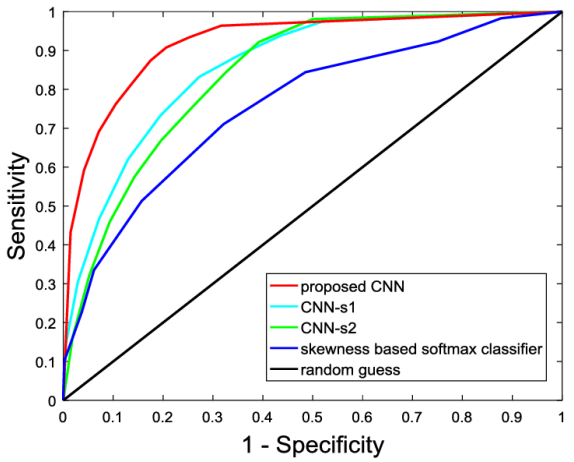
ROC (Receiver Operation Characteristic Curve)

- FPR(False Positive Rate)가 변화함에 따라 TPR(True Positive Rate, 재현률 Recall)의 변화를 나타낸 곡선
- '수신자판단곡선' 이라고도 불림, 일반적으로 의학분야 또는 머신러닝 이진분류 모델에서 모델의 성능을 평가하는 용도로 사용
- 민감도 (Sensitivity, TPR) : 실제값 Positive가 정확히 예측되어야 하는 수준을 나타냄
  (이상이 있는 것을 이상이 있다고 판단하는 지표)
- 특이성 (Specificity) : 실제값 Negative가 정확히 예측되어야 하는 수준을 나타냄
  (이상이 없는 것을 이상이 없다고 판단하는 지표)



$$TPR = \frac{True\ Positive}{Total\ Positive}$$

$$FPR = \frac{False\ Positive}{Total\ Negative}$$

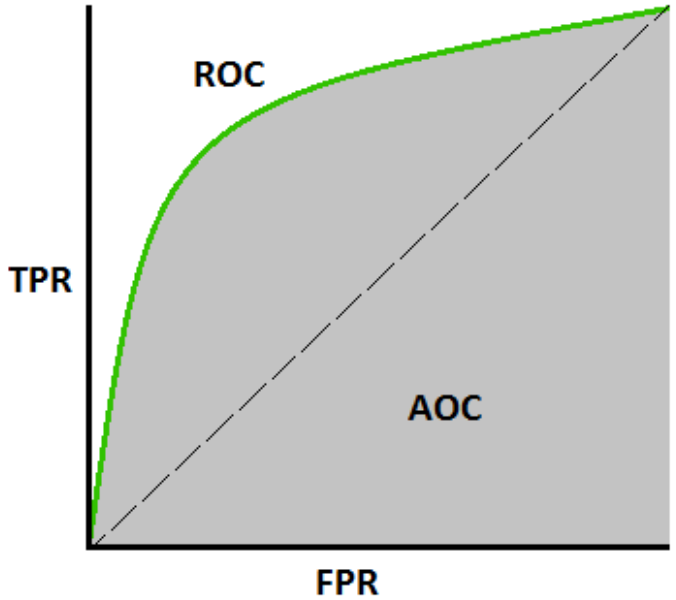


5. ROC & AUC - 분류

ROC & AUC

AUC (Area Under Curve)

- ROC 곡선 밑의 면적을 계산한 값
- 1에 가까울 수록 좋은 지표 (좋은 모델)
- AUC 수치가 커지려면, FRP가 작은 상태에서 큰 TPR을 얻을 수 있어야 함
- 가운데, Random 수준은 0.5 의 AUC 값을 가짐. (0.5 부터 1 사이 값)



6. 모델 평가 - 회귀

회귀 모델에서 평가

$$R^2 = 1 - \frac{RSS}{TSS}$$ (결정계수, 0~1 사이 값) / r2_sqaure

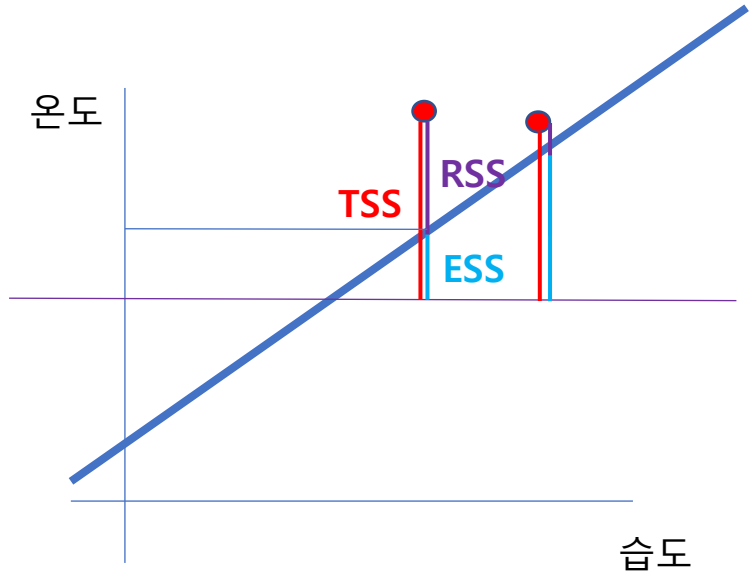
SST (Total Sum of Square, 총변동) : ∑(실제값 – 평균) ² = ESS + RSS

SSE (Residual(Error) Sum of Square, 오차변동) : ∑(실제값 – 예측값) ²

SSR (Regression Sum of Square, 회귀변동) : ∑(예측값 - 평균) ²

MSE (Mean Square Error) : $\sum \frac{(\text{실제값} - \text{예측값})^2}{\text{데이터 수}}$

MAE (Mean Absolute Error) : $\sum \frac{|\text{실제값} - \text{예측값}|}{\text{데이터 수}}$

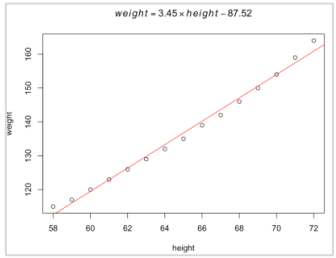


6. 모델 평가 - 회귀

▪ Residuals(잔차)

: 예측 모델의 예측값과 실제 값의 차이(error)

- 일반적으로 잔차 Plot(X축-설명변수, Y축-잔차)으로 모델의 적합성을 판단



▪ 잔차의 종류

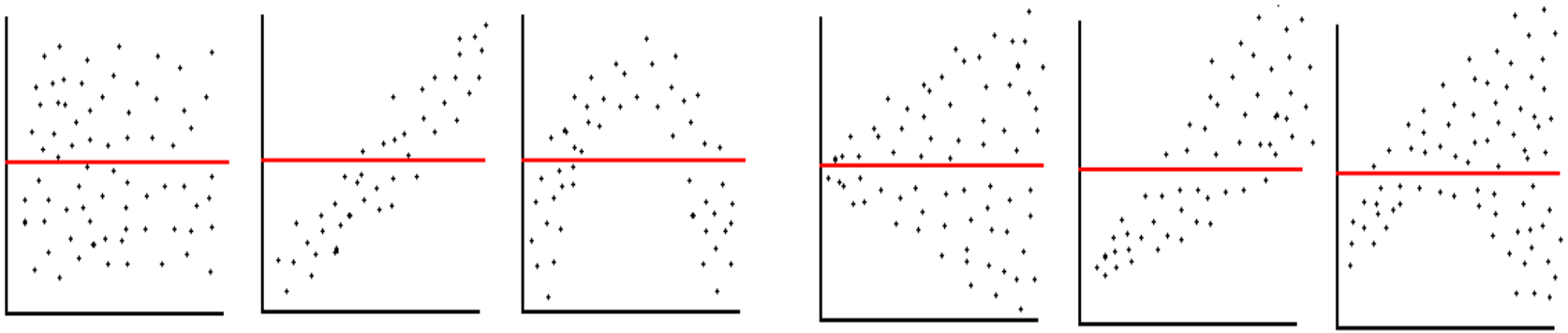
최소제곱잔차 (Ordinary Residual)

$$e_i = y_i - \hat{y}_i,\ i = 1, 2, \ldots, n$$

표준화 잔차 (Standardized Residual)

$$z_i = \frac{e_i}{\sigma\sqrt{1 - p_{ii}}}$$

분산이 동일하지 않은 경우i번째 잔차 e_i를
표준편차로 나눔 (평균=0, 표준편처=1)



비편향 등분산 | 편향(추세) 등분산 | 편향(2차 곡선) 등분산 | 비편향 이분산(확대) | 편향(추세) 이분산(확대) | 편향(2차 곡선) 이분산(확대)

6. 모델 평가 - 회귀

- **MSE(Mean Squared Error, 평균제곱오차)**

  : 회귀선과 모델 예측값 사이의 오차를 제곱한 값의 평균

  - 최근에는 분류 모델에서 목표변수의 범주를 숫자로 변환(예)이탈→1,유지→0) 후 모델 생성 및 평가에 사용

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

- **RMSE(Root MSE)**

  : OLS(최소 자승법) 추정에서 사용되는 일반적인 표준 오차. 단, scale(단위)에 영향을 받음

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

- **MAPE(Mean Absolute Percentage Error)**

  : MSE, RMSE의 단점(scale 영향)을 보완. 절대적 크기, 비율로 판단

$$\text{MAPE} = \frac{\sum_{t=1}^{n}\left|(A_t - F_t)/A_t\right|}{n}$$