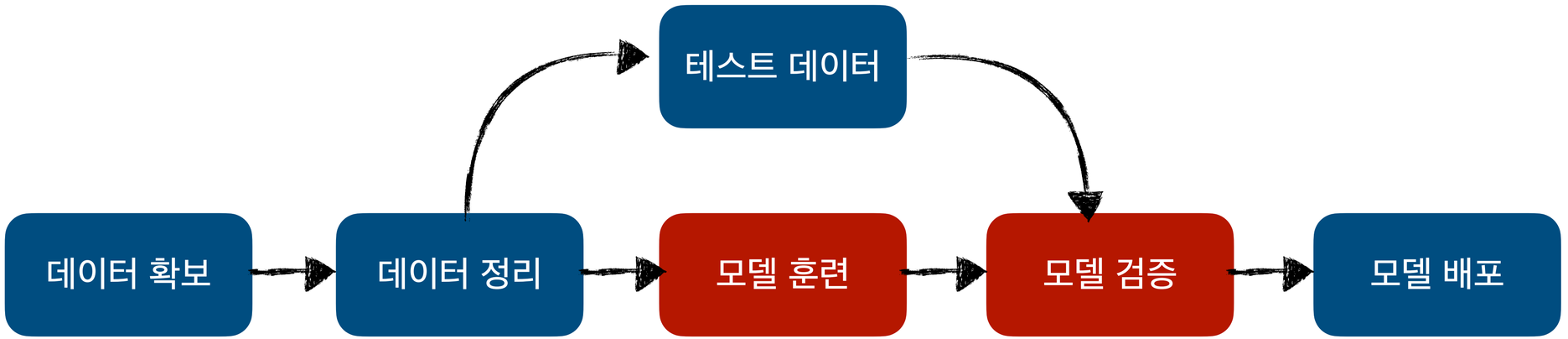
# Big Data Analytics Programming

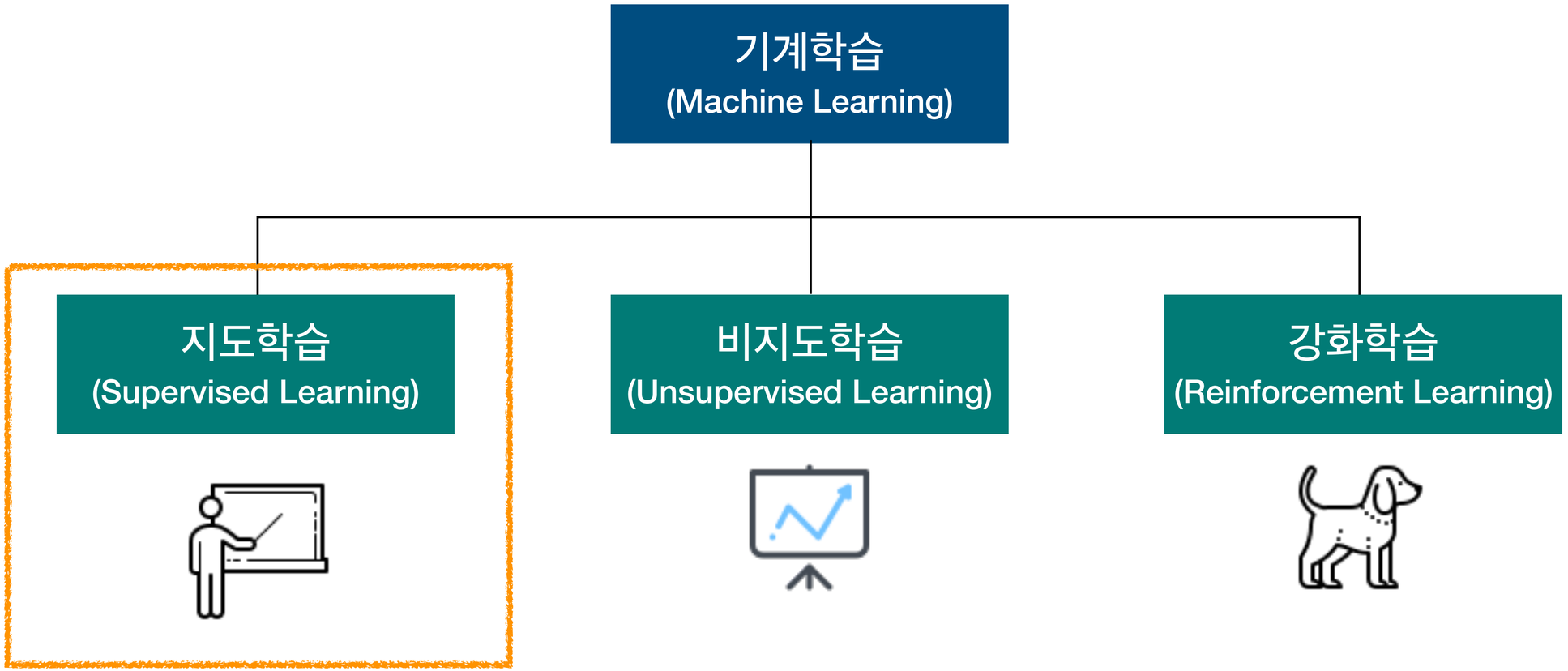
## Week-07. Supervised Learning

**Jungwon Seo, 2020-Fall**

# Machine Learning Process



# 기계학습의 종류


기계학습
(Machine Learning)
지도학습
(Supervised Learning)
비지도학습
(Unsupervised Learning)
강화학습
(Reinforcement Learning)

# 지도학습(Supervised Learning)

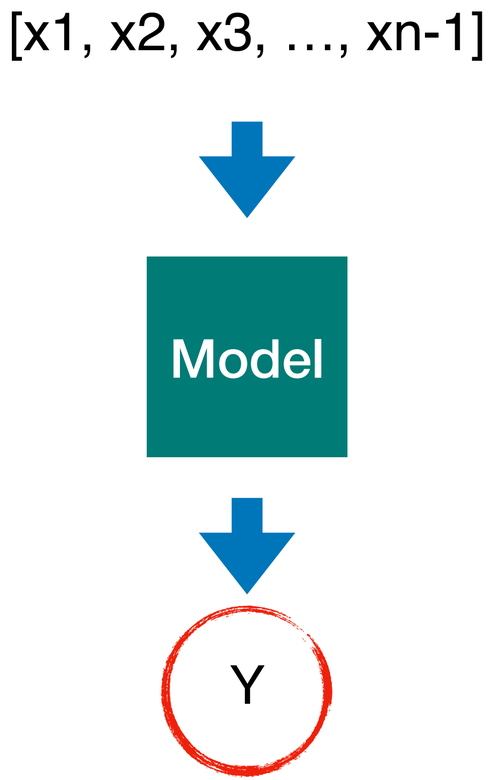
## 정답이 있는 데이터를 이용해 학습하는 방식

- 예제
  - 100,000개의 스팸메일 **여부**가 표시(Labeling)된 데이터셋
  - 10년치의 **주가**가 표시된 데이터셋
  - 강아지, 고양이와 같은 동물이 **이름이 표시된** 이미지 데이터셋
- 훈련 뒤에, 정답이 없는 데이터가 들어왔을 때, 정답을 예측하는..
- 정답의 Type에 따라서
  - 분류 문제 (Classification Task)
  - 회귀 문제 (Regression Task)
    - 回: 돌아올 회, 歸: 돌아갈 귀

# Classification

# Classification Task
## 분류문제

- Categorical 값을 예측하는 문제
  - True or False
  - A, B, C, D
- 훈련 데이터셋 구조
  - X’ = [x1, x2, x3, … , xn]
  - y = xn (클래스)
  - X = [x1, x2, x3, …, xn-1]
- 테스트 데이터셋 구조
  - X’ = [x1, x2, x3, …, xn-1]
- $x_n$ 즉, y를 예측하는 것이 목적

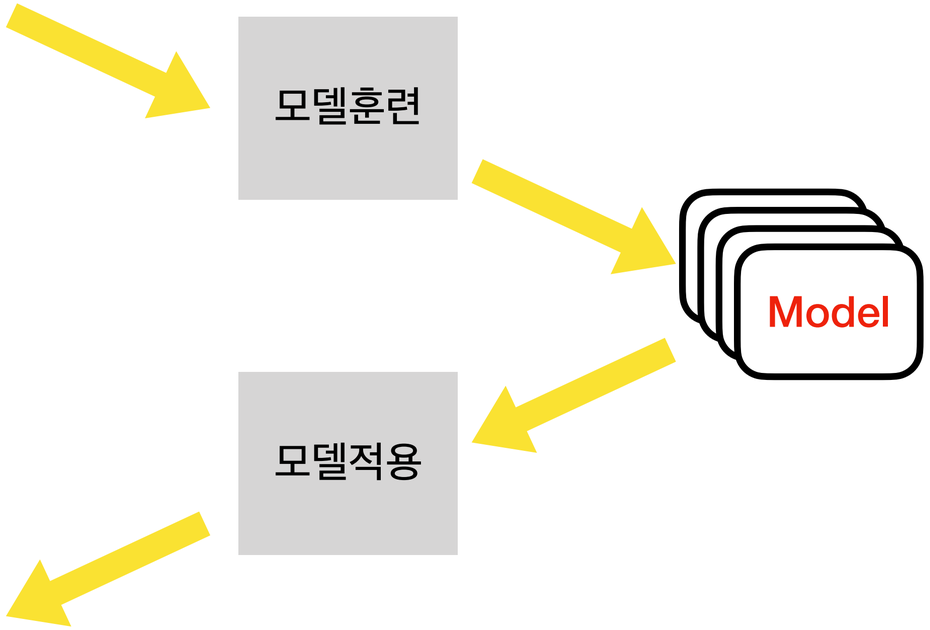


# 머신러닝 프로세스

## Pandas에 데이터를 올리고...

| ID | Attr1 | Attr2 | Attr3 | Class |
|---|---|---|---|---|
| 1 | Yes | 170 | 66 | Yes |
| 2 | Yes | 180 | 67 | Yes |
| 3 | No | 160 | 57 | No |
| 4 | No | 184 | 75 | No |
| 5 | No | 192 | 86 | Yes |
| 6 | Yes | 193 | 99 | No |
| 7 | No | 175 | 83 | Yes |
| 8 | No | 165 | 61 | Yes |
| 9 | Yes | 156 | 50 | No |

훈련 데이터




| ID | Attr1 | Attr2 | Attr3 | Class |
|---|---|---|---|---|
| 10 | No | 179 | 76 | Yes |
| 11 | No | 170 | 67 | Yes |
| 12 | Yes | 169 | 54 | No |
| 13 | Yes | 180 | 86 | No |
| 14 | Yes | 182 | 88 | No |

검증 데이터

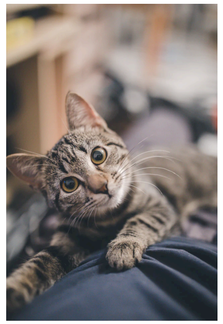
# Classifier 예

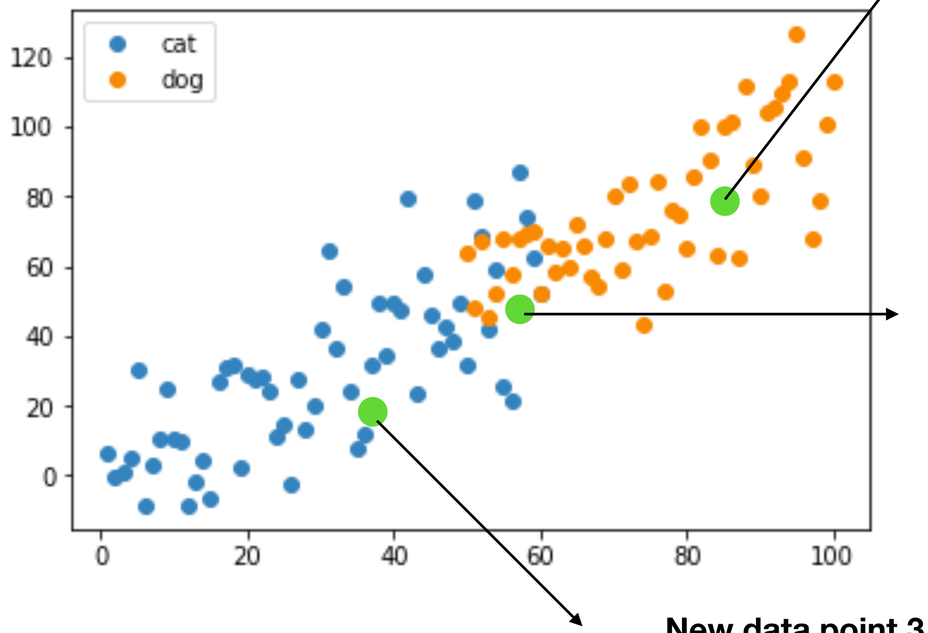
## K-Nearest Neighbor

- 간단한 Classification 알고리즘
- 이름에서 알 수 있듯이, 분류를 원하는 데이터를 기준으로 가장 가까운 K개의 주변 데이터 포인트들을 확인
- K개의 데이터 포인터들의 Class를 기준으로 새 데이터의 Class를 판별
- “친구를 보면 그 사람을 알 수 있다”

# K-Nearest Neighbor

## 동물의 신장과 체중이 주어졌다면






**New data point 1**
**고양이? 강아지?**

**New data point 2**
**고양이? 강아지?**

**New data point 3**
**고양이? 강아지?**

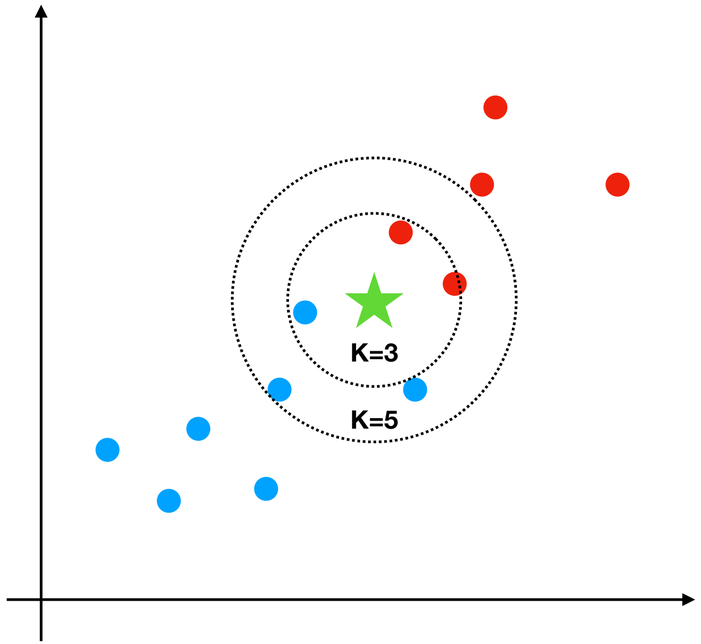
# K-Nearest Neighbor
## 알고리즘 원리

- 학습 알고리즘
  - 모든 데이터 포인트를 저장
  - data = [data1, data2,…]
  - data1 = [x1, x2, … xn-1], class
  - data2 = [x1, x2, … xn-1], class
  - data3 = [x1, x2, … xn-1], class
- 예측 알고리즘
  - 새 데이터 입력: new_data = [x1,…xn-1]
  - new_data를 기준으로 기존의 데이터들과의 거리계산
    - 데이터가 100,000개면 100,000번의 거리 계산
  - 거리 순으로 데이터들을 정렬 (오름차순)
  - 상위 K개의 데이터의 Class를 다수결로 적용하여 new_data의 Class 결정
    - If K =3
      - (data1 = “dog” data2=“dog”, data3=“cat”) => new_data = “dog”

# K-Nearest Neighbor

## 하이퍼 파라미터 튜닝

- K를 어떻게 결정하느냐에 따라 결과가 달라질 수 있음




- K가 짝수면? K가 너무 크면? K가 너무 작으면?

# Classifiers 종류

## Scikit Learn 라이브러리에서 제공하는 Classifiers

AdaBoostClassifier
BaggingClassifier
BernoulliNB
CalibratedClassifierCV
CategoricalNB
ComplementNB
**DecisionTreeClassifier**
DummyClassifier
ExtraTreeClassifier
ExtraTreesClassifier
**GaussianNB**
GaussianProcessClassifier
GradientBoostingClassifier
HistGradientBoostingClassifier
**KNeighborsClassifier**
LabelPropagation
LabelSpreading
LinearDiscriminantAnalysis
LinearSVC
**LogisticRegression**
LogisticRegressionCV
**MLPClassifier**
MultinomialNB
NearestCentroid
NuSVC
PassiveAggressiveClassifier
Perceptron
QuadraticDiscriminantAnalysis
RadiusNeighborsClassifier
**RandomForestClassifier**
RidgeClassifier
RidgeClassifierCV
SGDClassifier
**SVC**

# 모델 검증

## 잘 분류가 되었다는 것을 어떻게 검증 할 수 있을까?

- Confusion Matrix
  - [맞춤]-[예측결과]
  - True-Positive: 예측과 실제값이 모두 True 인것
  - True-Negative: 예측과 실제값이 모두 False 인것
  - False-Postive: 예측과 실제값이 틀리고 예측은 True
  - False-Negative: 예측과 실제값이 틀리고 예측은 False

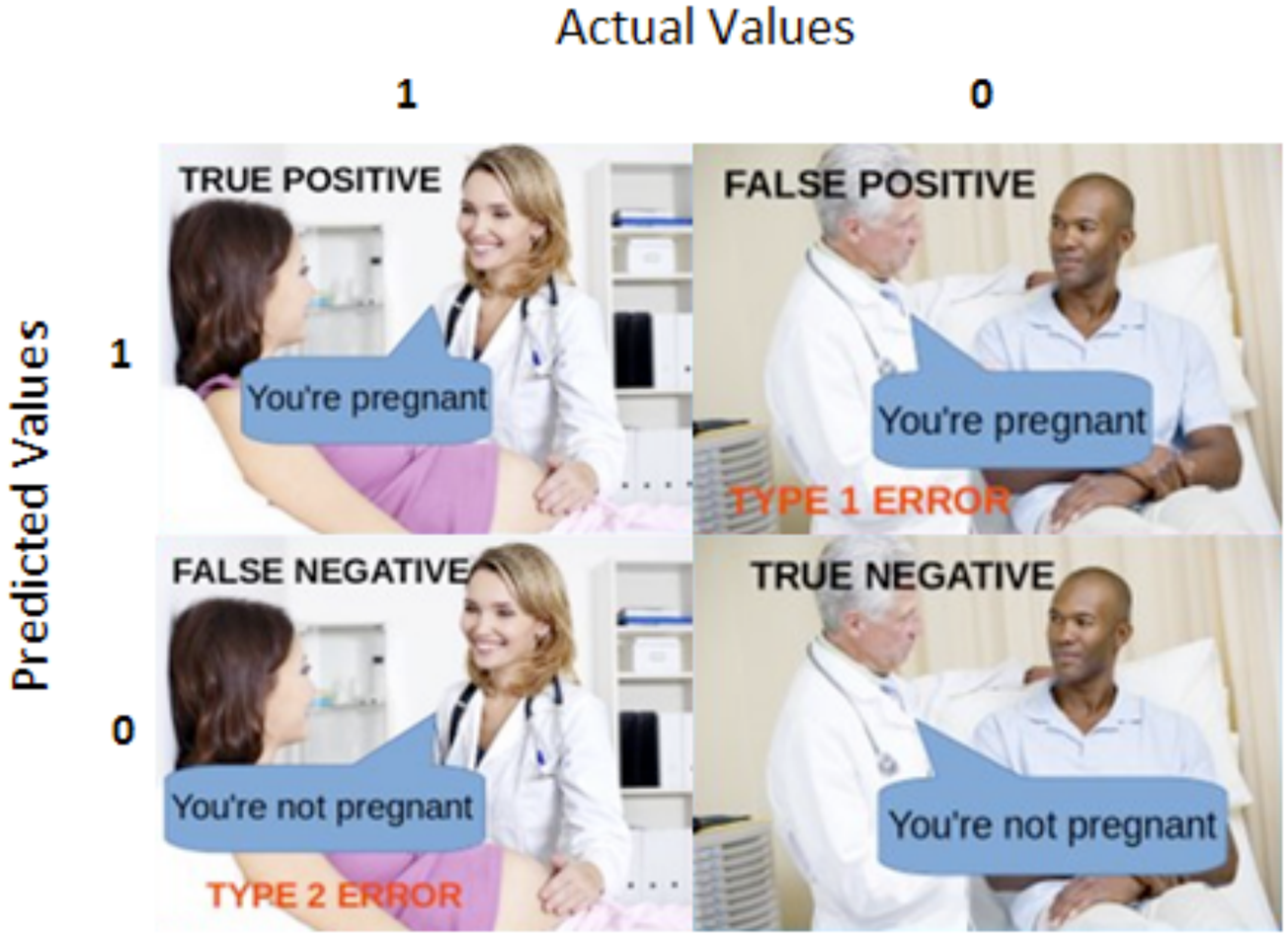
| n=190 | 예측: True | 예측: False | |
|---|---|---|---|
| 실제: True | TP 100 | FN 10 | 110 |
| 실제: False | FP 30 | TN 50 | 80 |
| | 130 | 60 | |

| n=190 | 예측: True | 예측: False | |
|---|---|---|---|
| 실제: True | TP 100 | FN 0 | 100 |
| 실제: False | FP 0 | TN 90 | 90 |
| | 100 | 90 | |

**완벽한 분류기의 경우**

# 모델 검증

## Confusion Matrix

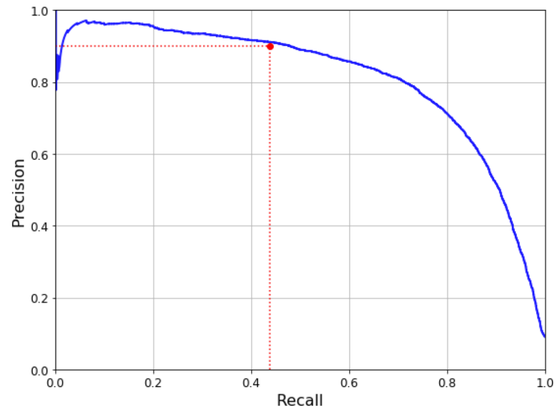


# 정밀도와 재현율

## 헷갈린다.. 헷갈려..

- 정밀도 : 모델이 True라고 예측한 것중 실제 True의 비율
  - $Precision = \frac{TP}{TP + FP}$
  - 예) 암이라고 예측한 결과중 실제 암인 경우의 비율
- 재현율 : 실제 True인 것 중 모델이 True라고 예측한 비율
  - $Recall = \frac{TP}{TP + FN}$
  - 예) 실제 암인 경우 중에 모델이 암이라고 예측한 비율
- 한가지만 높다고 좋은 모델이 아니고, 서로 상호보완적
  - 예1) FP가 0인 경우: 정밀도 100% -> 스팸 분류
    - 적어도 스팸이라고 분류한 것들은 스팸이여야 함 (스팸이 아닌 것을 스팸이라고 분류하지는 말아야..)
  - 예2) FN이 0인 경우: 재현율 100% -> 사기 결제
    - 비록 사기가 아닌 것을 사기라고 할지라도 사기는 다 인지를 해야함
- 정밀도/재현율 트레이드오프
  - 만약에 모델의 임계값을 조절 하여, 정밀도나 재현율을 변화시키면, 일반적으로 정밀도와 재현율은 반비례관계를 갖음
  - FP를 낮춘다는 것은, False로 예측하는 경우가 맞아지고, FN을 낮추면 True로 예측 하는 비율이 높아짐

| n=190 | 예측: True | 예측: False | |
|---|---|---|---|
| 실제: True | TP 100 | FN 10 | 110 |
| 실제: False | FP 30 | TN 50 | 80 |
| | 130 | 60 | |



# 정확도와 조화평균

## 당연히 정확도가 최선인줄 알았건만..!

- 정확도 : 재현율/정밀도와 달리 False를 False라고 답한 경우(TN)도 고려
  - $Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$
  - Imbalance 데이터에 유의미한 수치를 보이기 어려움
- 조화평균: 정밀도와 재현율을 동시에 고려하는 지표
  - $F_1 = \frac{2}{\frac{1}{Prec} + \frac{1}{Recall}} = 2 * \frac{Prec * Recall}{Prec + Recall} = \frac{TP}{TP + \frac{FN + FP}{2}}$
  - 현실적으로 레이블이 균형잡힌 데이터의 확보가 어렵기때문에 F1스코어를 평가지표로 사용함

| n=190 | 예측: True | 예측: False | |
|---|---|---|---|
| 실제: True | TP 100 | FN 10 | 110 |
| 실제: False | FP 30 | TN 50 | 80 |
| | 130 | 60 | |

# 연습문제

## 각각의 Accuracy, Precision, Recall, F1-score는?

- $Precision = \frac{TP}{TP + FP}$
- $Recall = \frac{TP}{TP + FN}$
- $Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$
- $F_1 = \frac{TP}{TP + \frac{FN + FP}{2}}$

n=20

| | 예측: True | 예측: False | |
|---|---|---|---|
| 실제: True | TP 5 | FN 10 | 15 |
| 실제: False | FP 0 | TN 5 | 5 |
| | 5 | 15 | |

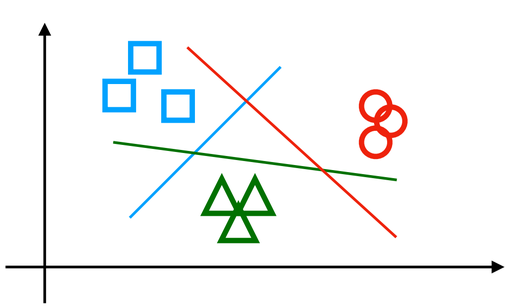
n=100

| | 예측: True | 예측: False | |
|---|---|---|---|
| 실제: True | TP 0 | FN 5 | 5 |
| 실제: False | FP 5 | TN 90 | 95 |
| | 5 | 95 | |

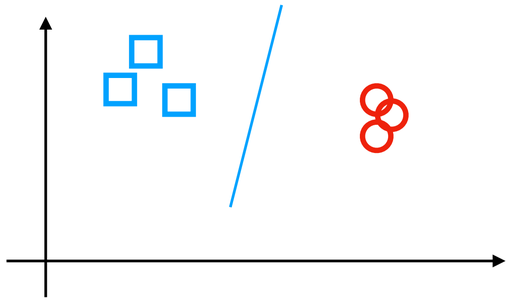
# 다중 분류

## 로지스틱 회귀나 서포트 벡터머신 분류기는 이진분류만 가능

- 다음과 같은 방법으로 다중 분류기로 변환
  - One-versus-all: 1과 나머지, 2와 나머지 …
  - One-versus-one: 0과 1구별, 0과 2구별… 9와 10 구별
    - 클래스 N개일때 분류기는 개가 필요
  - scikit-learn 라이브러리에서는 위의 두가지 방법중 알아서 선택후 분류



one-versus-all



one-versus-one

# 다중 분류에 대한 에러분석

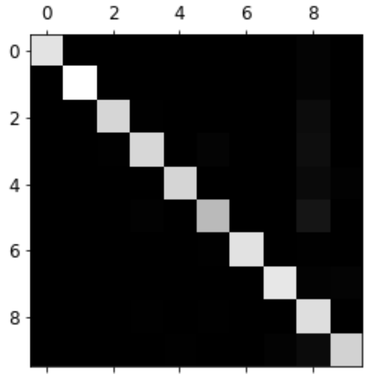
## Confusion Matrix

- 멀티 클래스 데이터에 대한 confusion matrix 출력시 주대각선 외의 값들은 모두 오류 분류

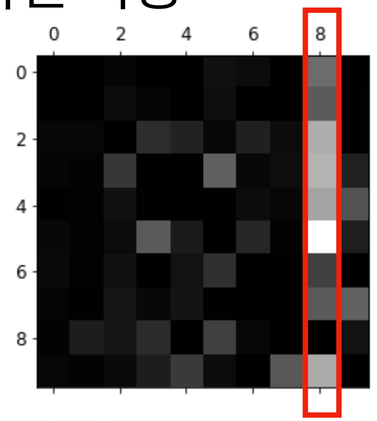
예측값 →

| 실제값 ↓ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 5577 | 0 | 22 | 5 | 8 | 43 | 36 | 6 | 225 | 1 |
| 1 | 0 | 6400 | 37 | 24 | 4 | 44 | 4 | 7 | 212 | 10 |
| 2 | 27 | 27 | 5220 | 92 | 73 | 27 | 67 | 36 | 378 | 11 |
| 3 | 22 | 17 | 117 | 5227 | 2 | 203 | 27 | 40 | 403 | 73 |
| 4 | 12 | 14 | 41 | 9 | 5182 | 12 | 34 | 27 | 347 | 164 |
| 5 | 27 | 15 | 30 | 168 | 53 | 4444 | 75 | 14 | 535 | 60 |
| 6 | 30 | 15 | 42 | 3 | 44 | 97 | 5552 | 3 | 131 | 1 |
| 7 | 21 | 10 | 51 | 30 | 49 | 12 | 3 | 5684 | 195 | 210 |
| 8 | 17 | 63 | 48 | 86 | 3 | 126 | 25 | 10 | 5429 | 44 |
| 9 | 25 | 18 | 30 | 64 | 118 | 36 | 1 | 179 | 371 | 5107 |

- 히트맵으로 표현시, 어떤 클래스가 오류가 많은지 쉽게 확인 가능




Heatmap




대각행렬을 0으로 변경 뒤 heatmap

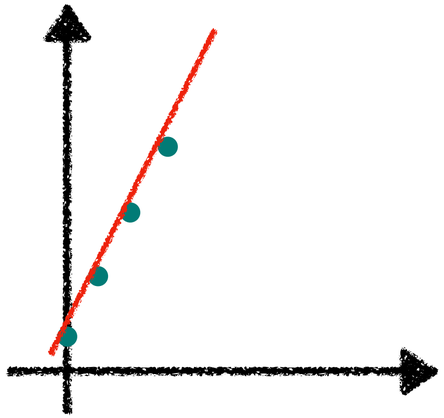
# Regression

# 기본원리

## 3과 1을 데이터 기반으로 결정

$$y = 3x+1$$

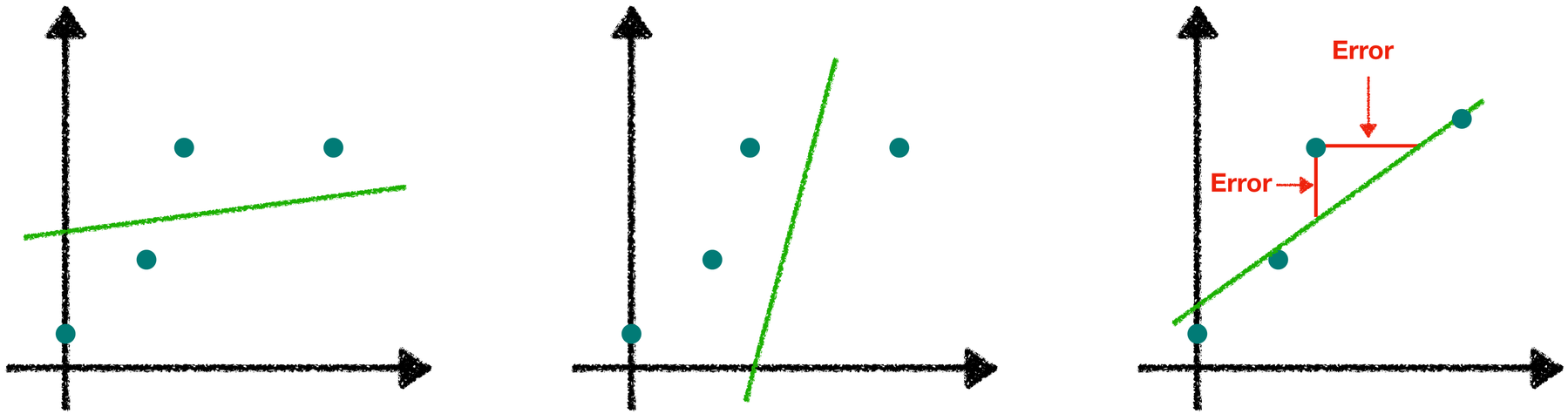
| x | y |
|---|---|
| 1 | 4 |
| 2 | 7 |
| 3 | 10 |
| 4 | ? |



# 기본 원리

## 컴퓨터 + 숫자 = 빠르게 모든 케이스를 계산

- Error를 최소화 할 수 있는 방정식(Equation) 찾기



# Linear Regression
## 대표적인 회귀분석 기법

- 단순 선형회귀
  - $y = wx + b$
- 다중 선형회귀 (n개의 feature)
  - $y = w_1x_1 + w_2x_2 + \cdots + w_nx_n + b$
- 가설 (Hypothesis)
  - 변수간의 관계를 유추하기 위해 수학적으로 나타낸 식
  - $H(x) = wx + b$

# Loss Function

## 예측을 하긴 했는데....

- 실제값과 예측값과의 차이를 측정 할 수 있는 함수
- Regression Loss Functions
  - Mean Squared Error Loss => $\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$
  - Mean Squared Logarithmic Error Loss
  - Mean Absolute Error Loss => $\mathrm{MAE} = \frac{\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right|}{n} = \frac{\sum_{i=1}^{n}\left|e_i\right|}{n}$

# 모델검증

## Regression이 제대로 됐는지를 어떻게 측정 할 수 있을까?

예측 값들 [1,0,1,1,1]
정답 값들 [1,1,0,1,1]

→ $\frac{\text{맞춘 값 수}}{\text{전체 값 수}}$ → $\frac{3}{5}$ → 60%

높을 수록 좋다

<분류문제 - Accuracy>

---

예측 값들 [100, 99, 39, 22]
정답 값들 [100, 89, 49, 30]

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 .$$

$$MSE = \frac{1}{4}((100-100)^2 + (99-89)^2 + (39-49)^2 + (22-30)^2)$$

= 66

0에 가까울 수록 좋다

<회귀문제 - Mean Squared Error>

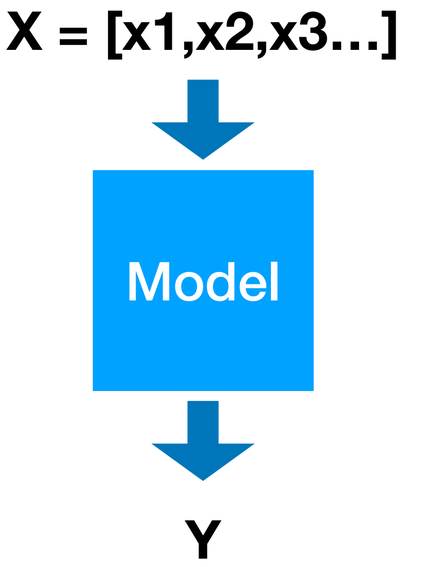
# Regressor 종류

## Scikit Learn 라이브러리에서 제공하는 Regressors

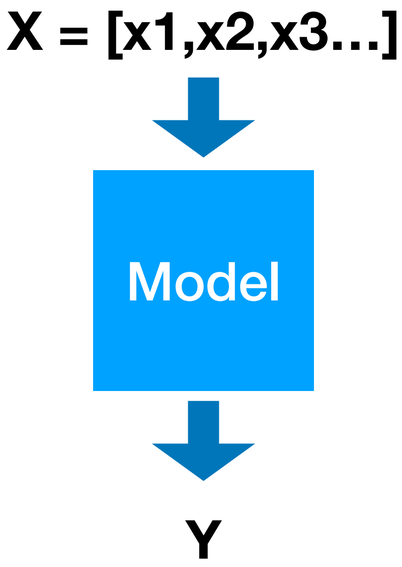
```
ARDRegression
AdaBoostRegressor
BaggingRegressor
BayesianRidge
CCA
DecisionTreeRegressor
DummyRegressor
ElasticNet
ElasticNetCV
ExtraTreeRegressor
ExtraTreesRegressor
GaussianProcessRegressor
GradientBoostingRegressor
HistGradientBoostingRegressor
HuberRegressor
IsotonicRegression
KNeighborsRegressor
KernelRidge
Lars
LarsCV
Lasso
LassoCV
LassoLars
LassoLarsCV
LassoLarsIC
LinearRegression
LinearSVR
MLPRegressor
MultiTaskElasticNet
MultiTaskElasticNetCV
MultiTaskLasso
MultiTaskLassoCV
NuSVR
OrthogonalMatchingPursuit
OrthogonalMatchingPursuitCV
PLSCanonical
PLSRegression
PassiveAggressiveRegressor
RANSACRegressor
RadiusNeighborsRegressor
RandomForestRegressor
Ridge
RidgeCV
SGDRegressor
SVR
TheilSenRegressor
TransformedTargetRegressor
```

# Supervised Learning 정리

## Classification vs Regression




이때 Y는 Categorical
e.g., (1,0), (True,False), (A, B, C, D)




이때 Y는 Numerical
e.g., 133, 0.11, 103030

# E.O.D