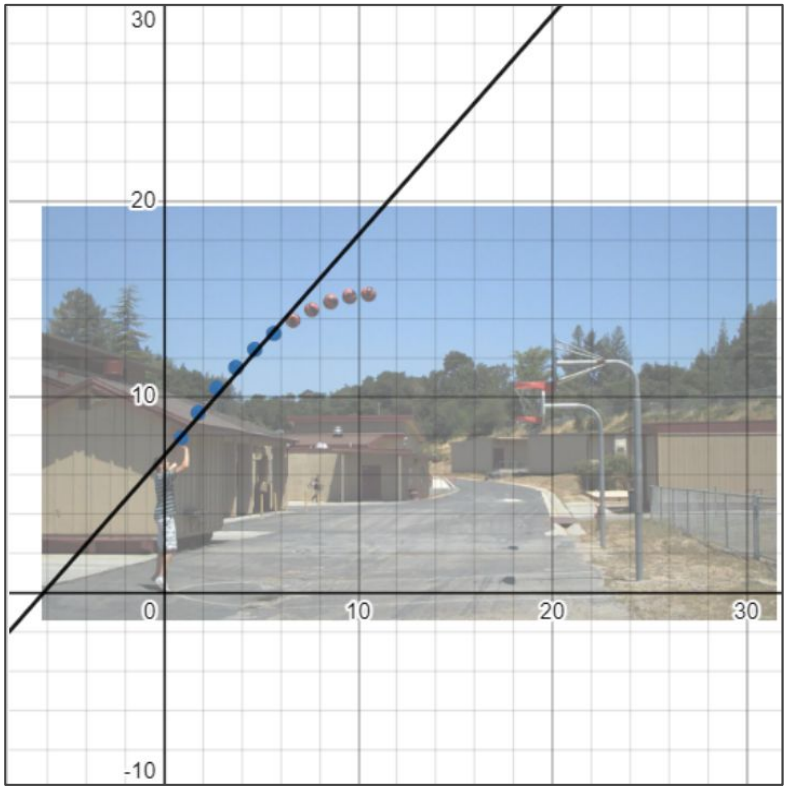
프로그래밍_인공지능3

2022학년도 2학기

선형(직선)으로 예측이 가능할까?

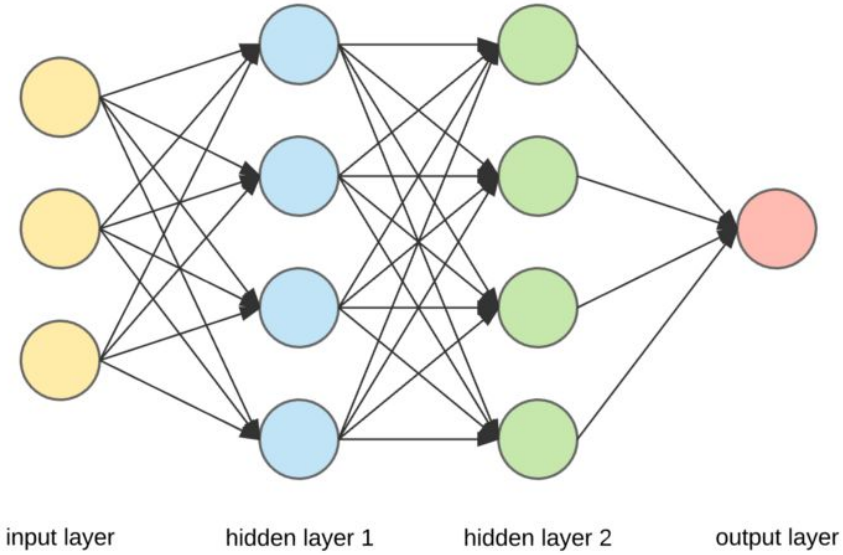



```
Epoch 496/500
1/1 [==============================] - 0s 4ms/step - loss: 0.4303
Epoch 497/500
1/1 [==============================] - 0s 4ms/step - loss: 0.4297
Epoch 498/500
1/1 [==============================] - 0s 4ms/step - loss: 0.4292
Epoch 499/500
1/1 [==============================] - 0s 3ms/step - loss: 0.4287
Epoch 500/500
1/1 [==============================] - 0s 3ms/step - loss: 0.4282
1/1 [==============================] - 0s 169ms/step
array([[20.589323]], dtype=float32)
```

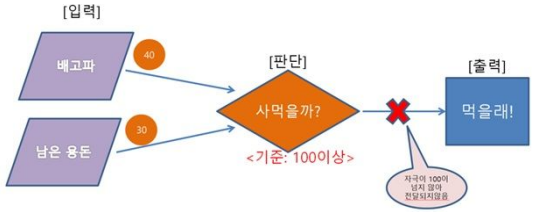
```
Epoch 45/500
1/1 [==============================] - 0s 7ms/step - loss: 844158799850058407433615398248906752.0000
Epoch 46/500
1/1 [==============================] - 0s 7ms/step - loss: 512228356514627234751378228865977548 8.0000
Epoch 47/500
1/1 [==============================] - 0s 7ms/step - loss: 310815884495259681067677407277744128 00.0000
Epoch 48/500
1/1 [==============================] - 0s 8ms/step - loss: inf
Epoch 94/500
1/1 [==============================] - 0s 7ms/step - loss: inf
Epoch 95/500
1/1 [==============================] - 0s 7ms/step - loss: nan
Epoch 96/500
1/1 [==============================] - 0s 7ms/step - loss: nan
Epoch 97/500
1/1 [==============================] - 0s 7ms/step - loss: nan
```

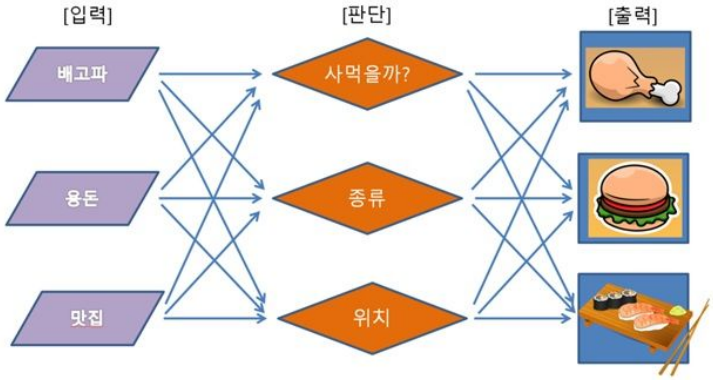
복잡한 연산을 예측하기 위해서는...

- 그에 맞는 모델이 필요함



- 자료에 맞는 가중치가 필요함





예측 모델을 구현하고자 할 때 가장 고려해야할 점?

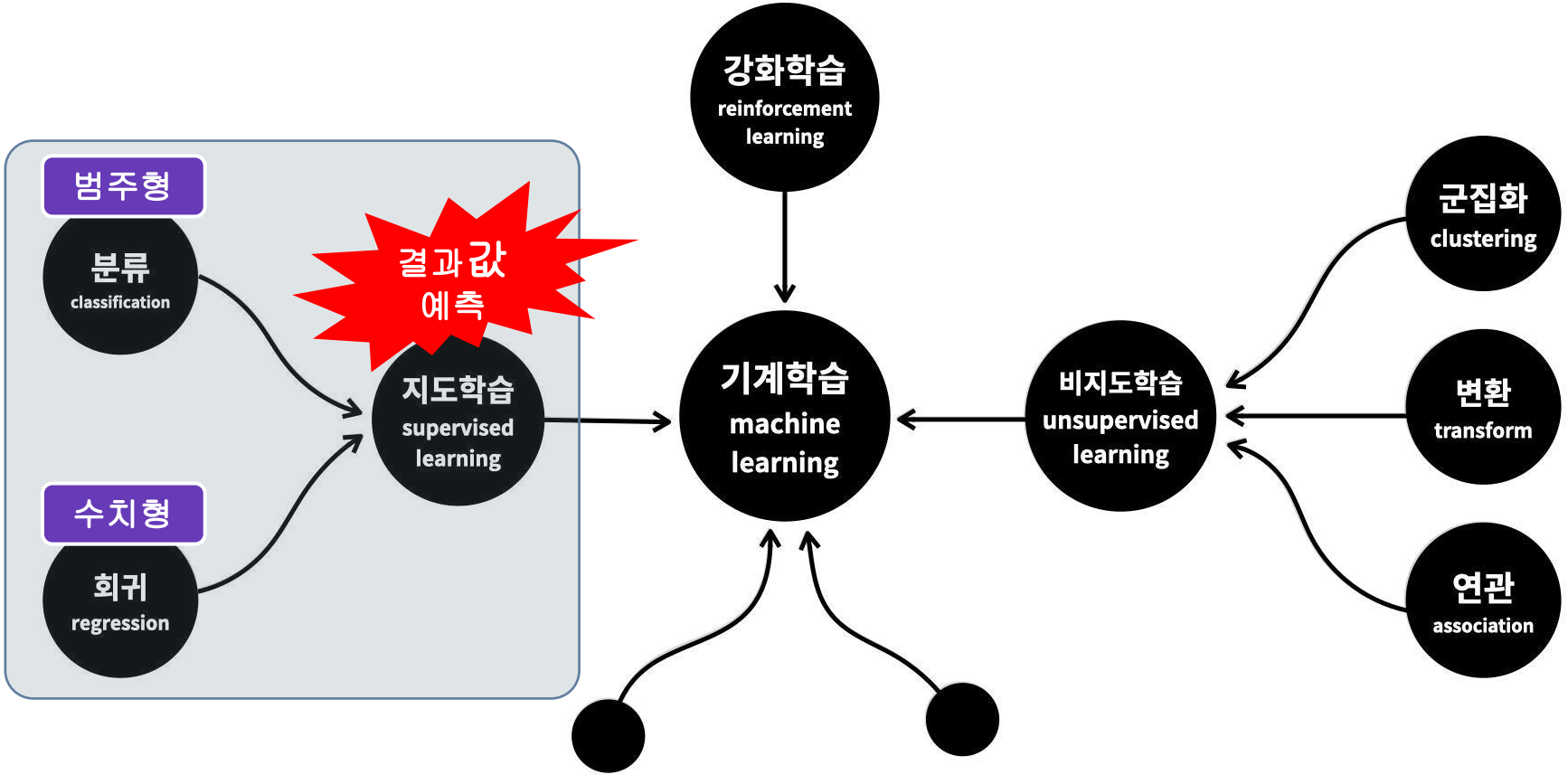
예측하고자 하는 값

수치형

인구수, 평균기온,
매출액, 신장,
체중, 혈압…

범주형

성별(남/여), 성공여부
(성공/실패), 효과
(없음/조금있음/매우있음),
혈액형(A/B/O/AB)...


강화학습
reinforcement learning
범주형
분류
classification
결과값 예측
지도학습
supervised learning
수치형
회귀
regression
기계학습
machine learning
비지도학습
unsupervised learning
군집화
clustering
변환
transform
연관
association

지도학습(회귀)

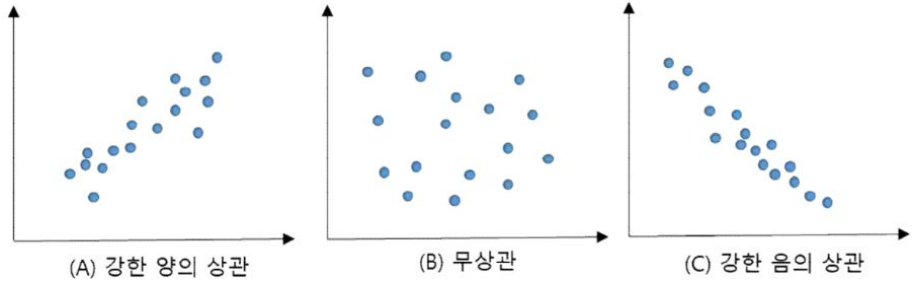
지도학습(Supervised Learning)_상관분석 1

상관분석(Correlation analysis)

- 두 연속형 변수 사이 상관관계가 존재하는지 파악하고, 상관관계의 정도를 확인하는 것
- 상관계수(관련성 파악 지표)로 선형적 상관도를 확인하여 정도를 파악
- 상관 분석 과정
  - 산점도 두 변수 상관 파악
  - 상관계수 확인
  - 의사결정

지도학습(Supervised Learning)_상관분석 2

상관분석(Correlation analysis)



상관	상관계수
양의 상관	+0.1 ~ +0.3: 약한 양의 상관관계 +0.3 ~ +0.7: 뚜렷한 양의 상관관계 +0.7 ~ +1.0: 강한 양의 상관관계
무상관	-0.1 ~ + 0.1: 상관관계가 없다
음의 상관	-0.3 ~ -0.1: 약한 음의 상관관계 -0.7 ~ -0.3: 뚜렷한 음의 상관관계 -1.0 ~ -0.7: 강한 음의 상관관계

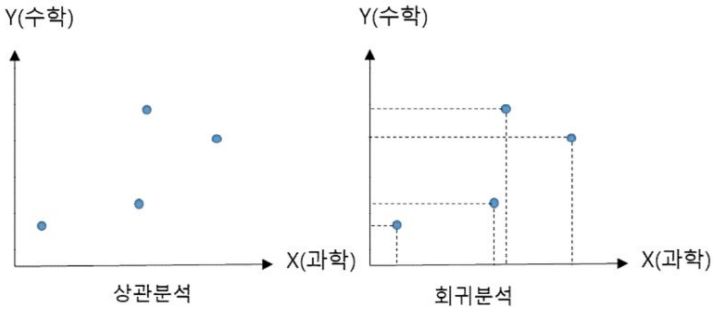
지도학습(Supervised Learning)_회귀분석 1

회귀분석(Regression analysis)

- 상관분석에서는 두 연속성 변수 X와 Y의 상관 정도만 알 수 있고, 인과관계는 알 수 없음
- 회귀분석에서는 두 연속성 변수 X와 Y를 독립변수와 종속변수라고 하는 인과관계로 설명할 수 있음
- '과학점수(X)가 좋으면 수학점수(Y)가 좋을까?' 와 같이 간단하지만 미래를 예측할 수 있는 머신러닝의 초기모델이 됨

지도학습(Supervised Learning)_회귀분석 2

회귀분석(Regression analysis)



X	Y
독립변수, 설명변수, 원인변수	종속변수, 반응변수, 결과변수 클래스, 라벨(머신러닝)
다른 변수에 영향을 주는 원인 (Dependent Variable Response)	다른 변수에 영향을 받는 결과 (Independent Variable, Predictor Fator)

지도학습(Supervised Learning)_회귀분석 3

회귀분석(Regression analysis)

선형회귀분석

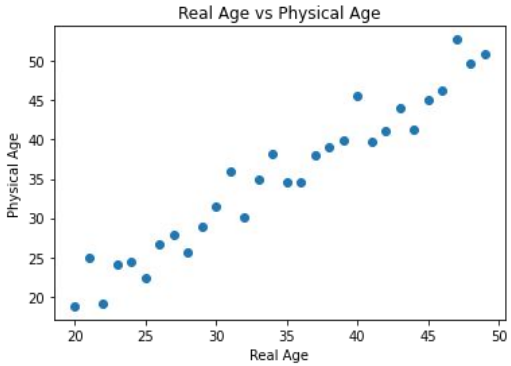
- 쌍으로 관찰된 연속형 변수들 사이의 관계에 있어서 한 변수를 원인으로 하고 다른 변수들을 결과로 하는 분석
- 독립변수와 종속변수 사이 선형식을 구하고 그 식을 이용하여 변수값들이 주어졌을 때 종속변수의 변수값을 예측하는 분석방법
- 독립변수의 개수에 따라 단순 선형과 다중 선형으로 구분함

지도학습(Supervised Learning)_회귀분석 4

회귀분석(Regression analysis)

단순선형회귀분석

- 주어진 데이터를 대표하는 하나의 직선을 찾는 것(회귀선)
- x변수와 y변수 간의 관계를 y = ax + b 와 같은 하나의 선형관계식으로 표현

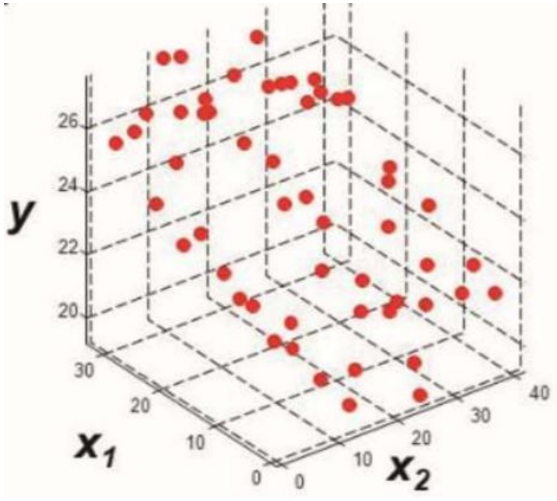


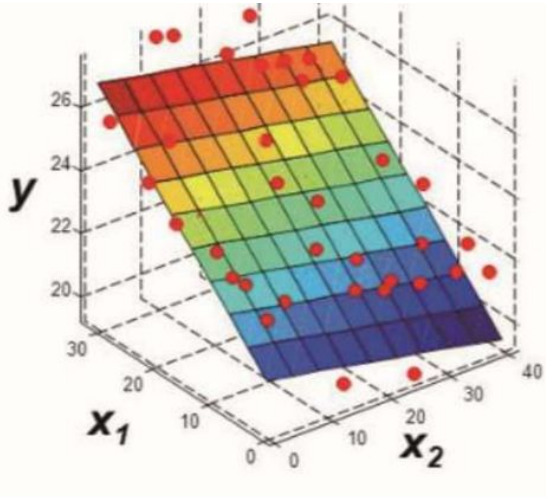
지도학습(Supervised Learning)_회귀분석 5

회귀분석(Regression analysis)

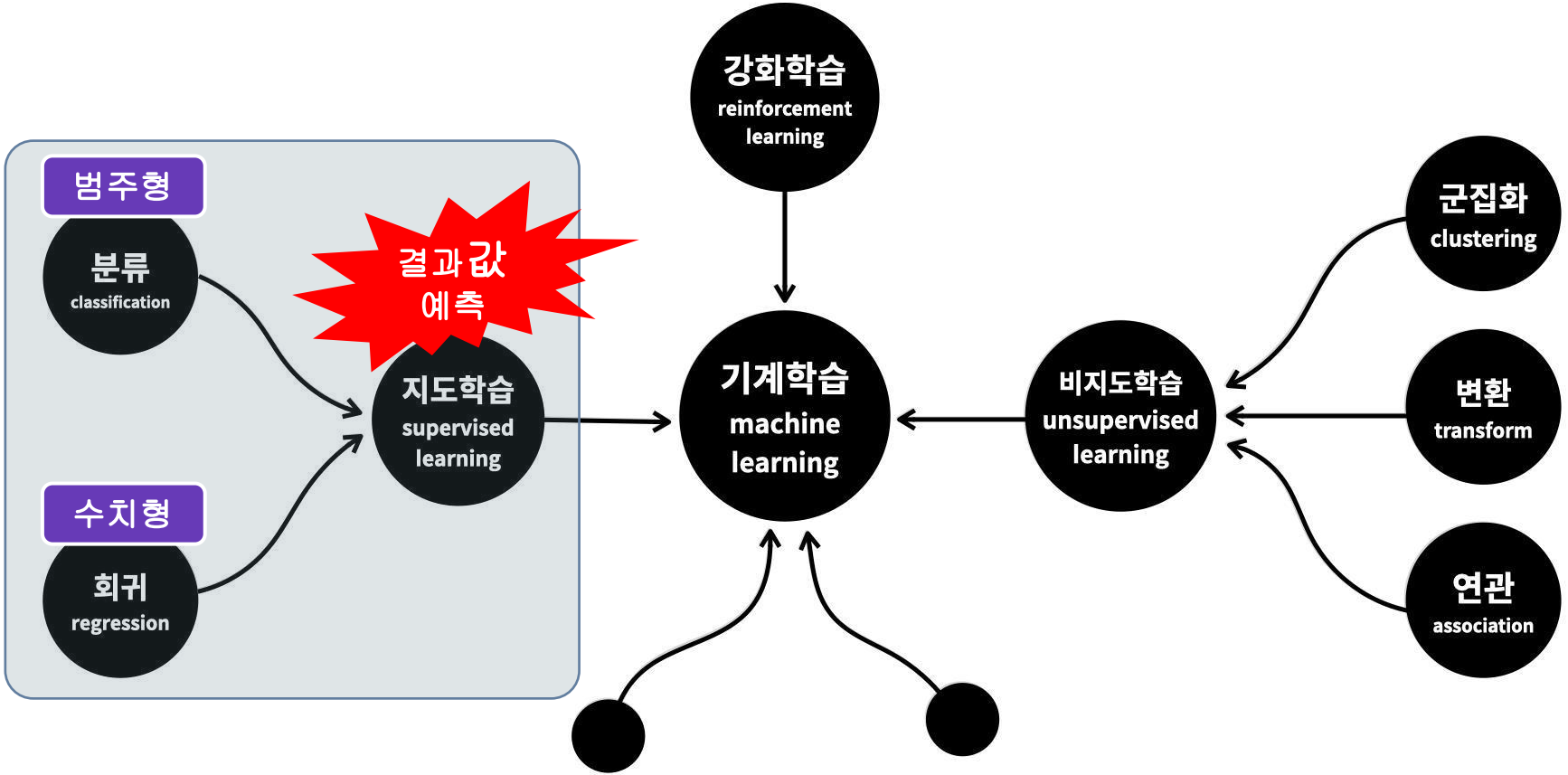
다중선형회귀분석

- 변수가 2개 이상인 경우





지도학습(분류)


강화학습
reinforcement learning
범주형
분류
classification
결과값 예측
지도학습
supervised learning
수치형
회귀
regression
기계학습
machine learning
비지도학습
unsupervised learning
군집화
clustering
변환
transform
연관
association

예측 모델을 구현하고자 할 때 가장 고려해야할 점?

예측하고자 하는 값

수치형

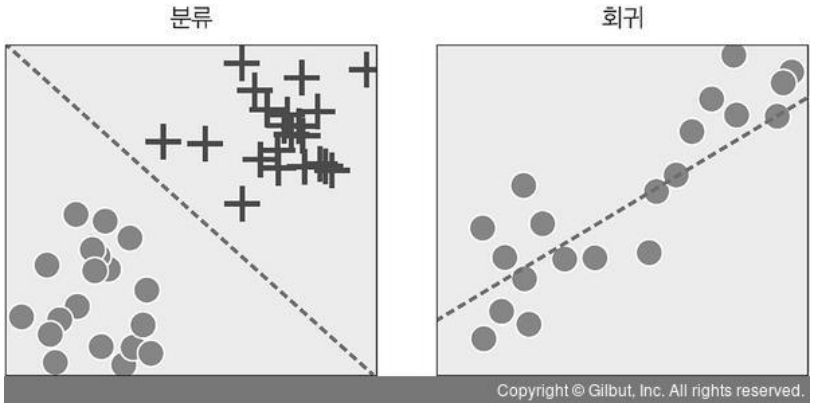
인구수, 평균기온,
매출액, 신장,
체중, 혈압…

범주형

성별(남/여), 성공여부
(성공/실패), 효과
(없음/조금있음/매우있음),
혈액형(A/B/O/AB)...

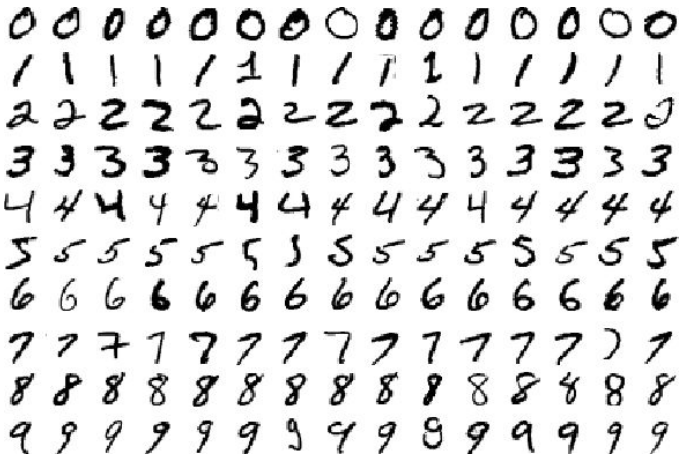
분류(Classification)의 사례

독립변수	종속변수
공부시간	합격여부
온도, 날씨, 미세먼지	야외활동 권장 여부
포도의 품종, 산도, 당도, 지역, 연도	와인의 등급
자동차 속도	과속 여부
키, 몸무게, 시력, 지병	현역, 공익, 면제
소고기의 지방함량, 지방색, 성숙도, 육색	소고기 등급



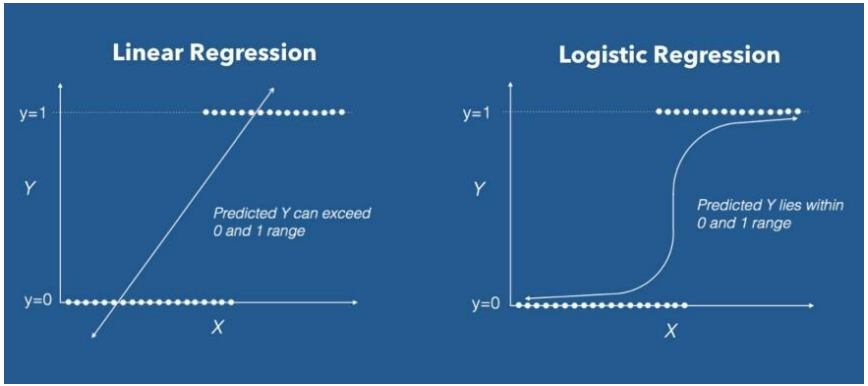
분류(Classification) 모델

- 라벨(레이블)이 달린 학습 데이터로 학습한 후 새로 입력된 데이터가 학습했던 어느 그룹에 속하는지 찾아내는 방법
- 분류 모델의 결과값은 학습한 그룹 중 하나로 출력됨
- 예) 라벨(0~9), 데이터: 아래 그림 > 입력: > 출력:



  - 학습시킨적 없는 'ㅎ'를 입력하면, 0~9 중 가장 가깝다고 생각하는 숫자 값을 출력해줄거임

지도학습_ 로지스틱 회귀



(공통점)

- 독립변수와 종속변수의 관계로 생성

(차이점)

- 선형 회귀: 연속형(0~1) 변수 예측
- 로지스틱 회귀: 범주형(0, 1) 변수 예측

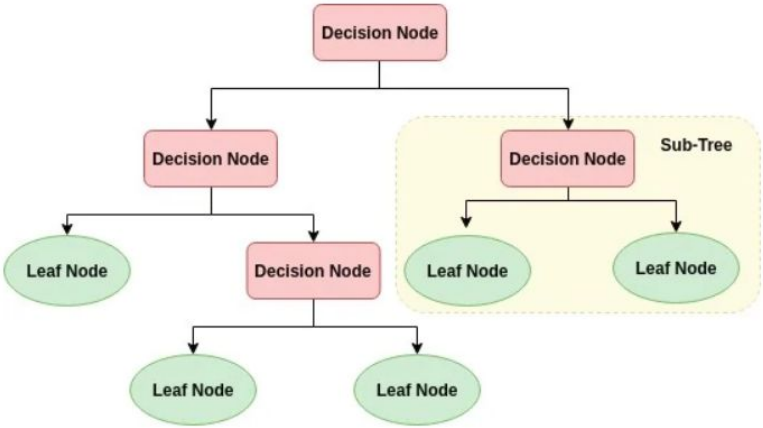
지도학습_ 나이브 베이즈

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

- P(A|B)

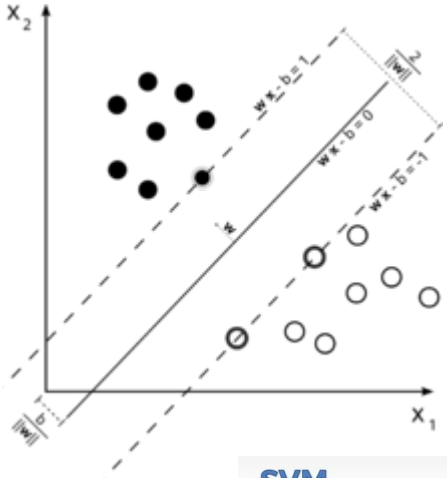
  = 사건 B가 발생했을 때, 사건 A가 발생할 확률(조건부 확률)

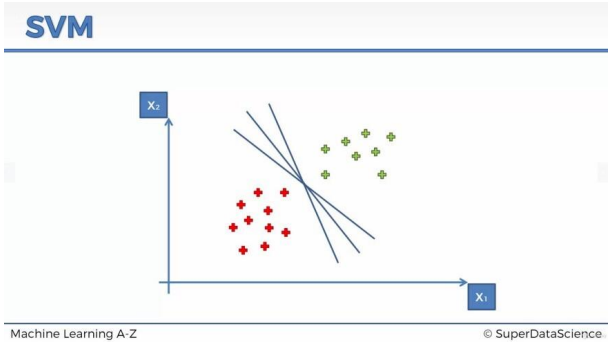
지도학습_ 결정 트리



- 트리 기반
- 규칙 노드에서 규칙에따라 분할됨
- 아래로 갈수록 많은 규칙의 영향을 받게 되는 것

지도학습_SVM(서포트 벡터 머신)



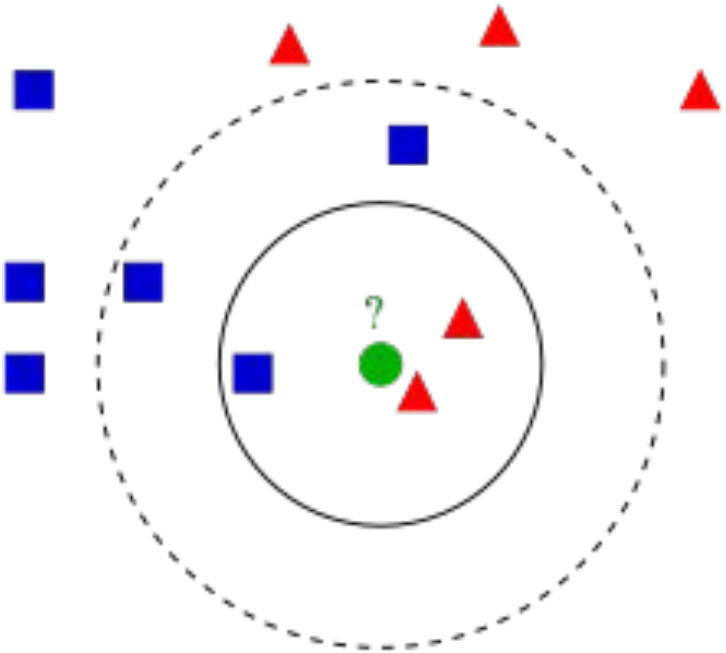


- 클래스를 분류할 수 있는 경계선 중 최적의 라인을 찾아내는 원리

(마진 값이 최대일 때, 최적의 분류 조건에 만족함)

- 마진: 경계선과 서포트 벡터 간 거리
- 서포트 벡터: 경계선과 가장 가까이 있는 데이터

지도학습_ KNN(K-Nearest Neighbor, K- 최소 근접)



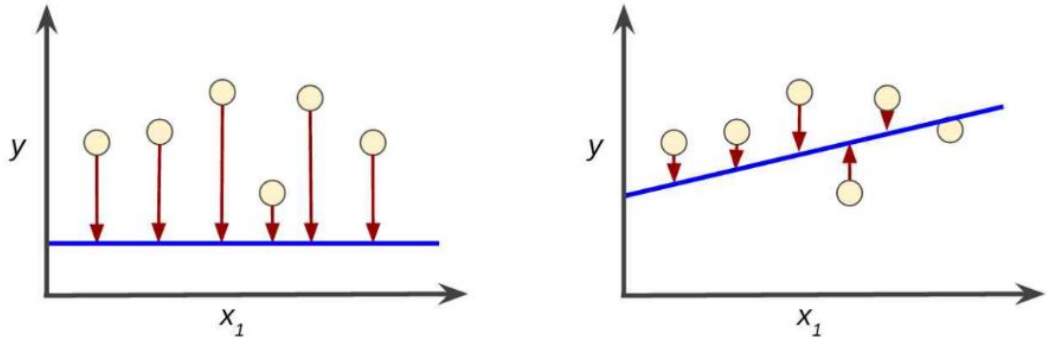
- 입력된 특징 벡터X에 가장 가까운 점 K개를 뽑아 이들의 라벨(레이블)을 사용하는 분류기
- 점들이 서로 다른 레이블들을 가질 경우, 가장 많은 라벨 선택함
- 다수결에서 결과가 특정되도록 만들기 위해 k는 홀수로 선정함

머신러닝 모델(손실함수, 최적화)

머신러닝 모델 최적화_손실함수(mse)

비용함수/손실함수

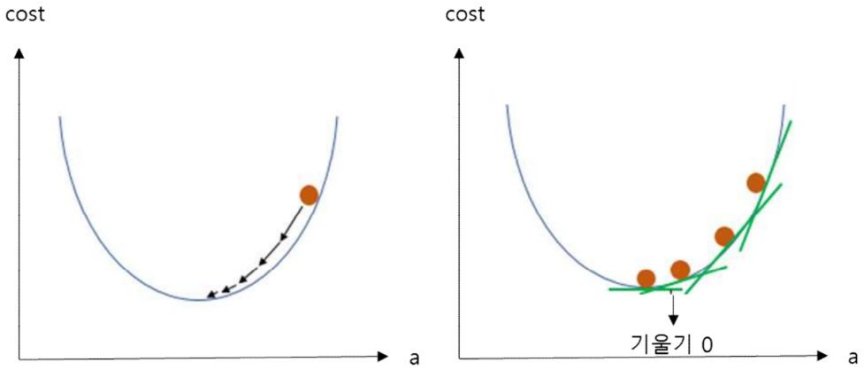
- 선형회귀식에서 **평균제곱오차(mse)**는 머신러닝 모델을 구축할 때, 값이 작을수록 원본과의 오차가 적은 것으로 추측한 값의 정확성이 높다고 할 수 있음



머신러닝 모델 최적화_경사하강법(gd)1

경사 하강법(Gradient descent)

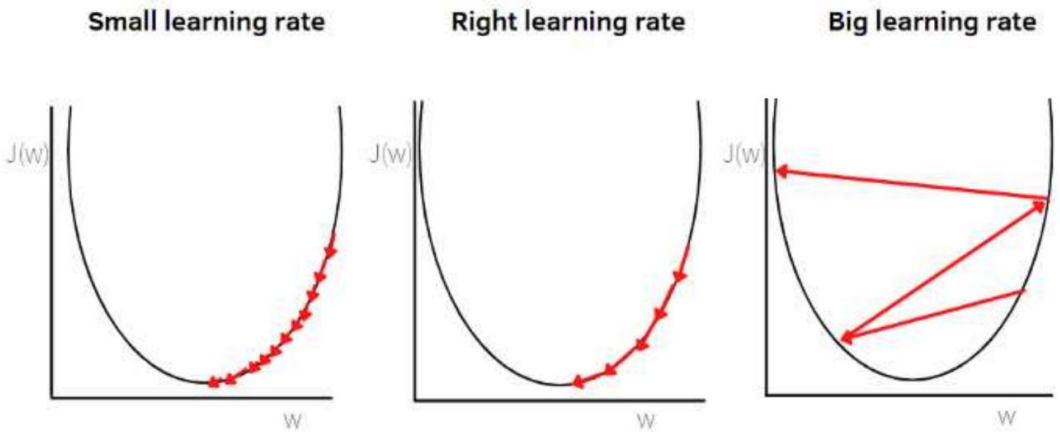
- 손실함수를 최소화하는 매개변수를 찾는 방법
- 손실함수 값이 가장 낮은 지점을 찾아가도록 손실함수의 기울기를 구해 최적값을 찾아가는 방법



머신러닝 모델 최적화_경사하강법(gd) 2

경사 하강법(Gradient descent)

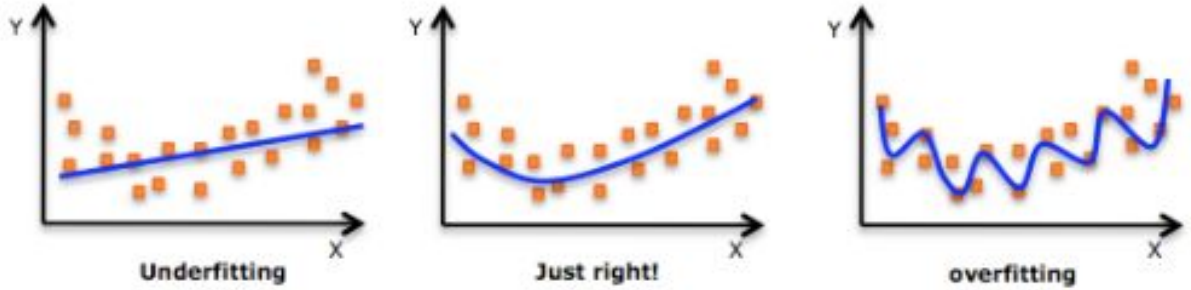
- Learning rate의 영향



머신러닝 모델 일반화 1

과대적합(Overfitting)과 과소적합(Underfitting)

- 과대적합: 모델이 훈련 데이터에 너무 잘 맞지만 일반성이 떨어지는 문제 발생
- 과소적함: 모델이 너무 단순해서 데이터의 포함된 의미를 제대로 학습하지 못하는 문제 발생



머신러닝 모델 일반화 2

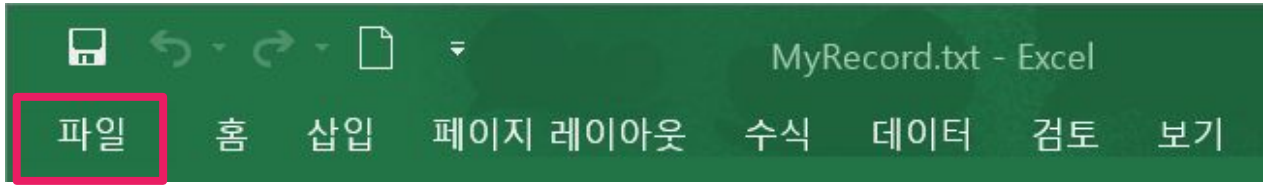
과대적합(Overfitting)과 과소적합(Underfitting)

- 좋은 머신러닝 모델(알고리즘)은?
  - 데이터의 양
  - 모델의 특징 개수
  - 일반화, 정규화, 가중치 규제, 드롭아웃

txt파일을 csv파일로 바꾸기

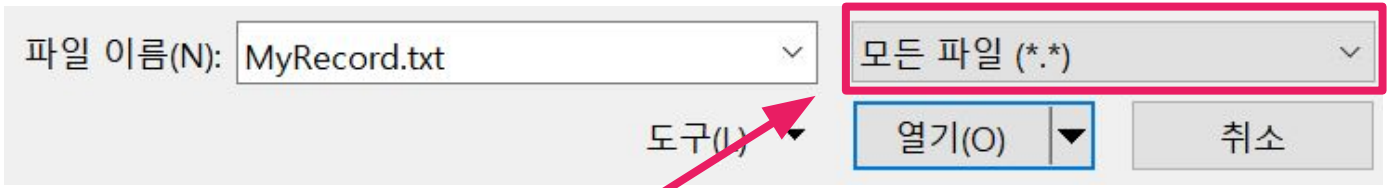
1. excel에서 txt 파일 열기

①



②

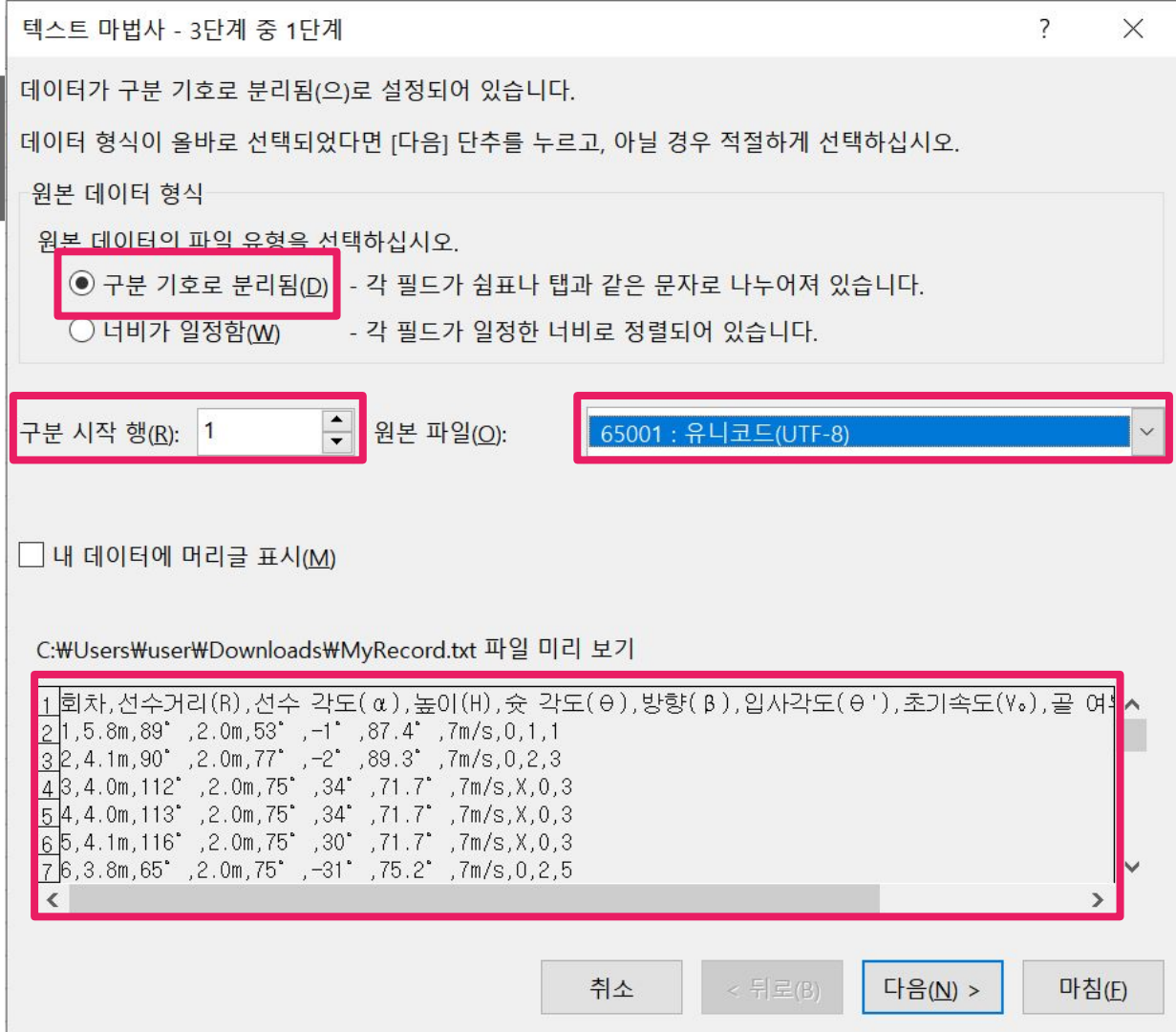
③



모든 파일로 바꿔야 함!

2-1. 구분기호설정

- 표시된 부분 확인
- 다 확인 했으면 다음!

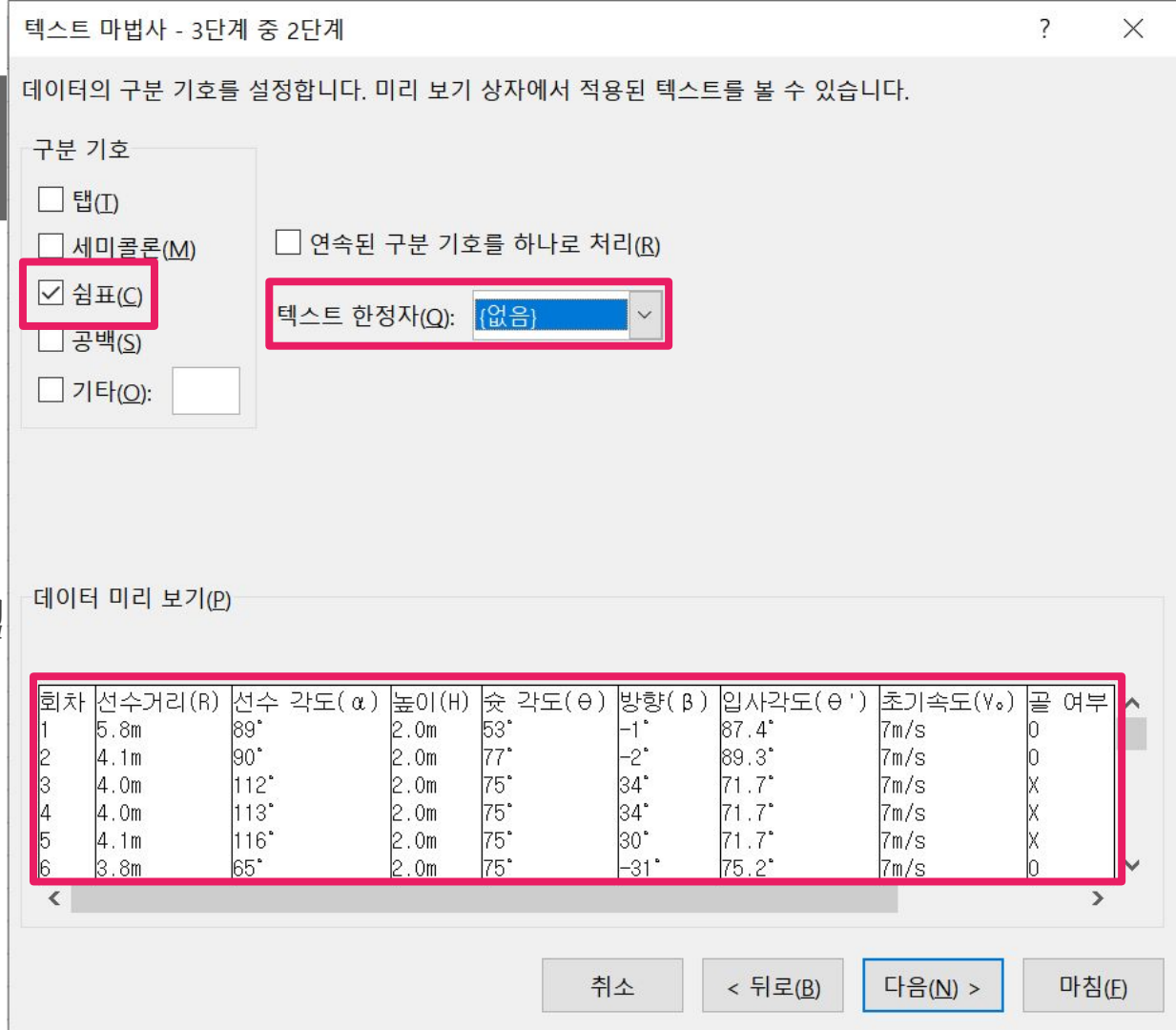


2-2. 구분기호설정

- 데이터 미리보기를 특히 유의해서 보기

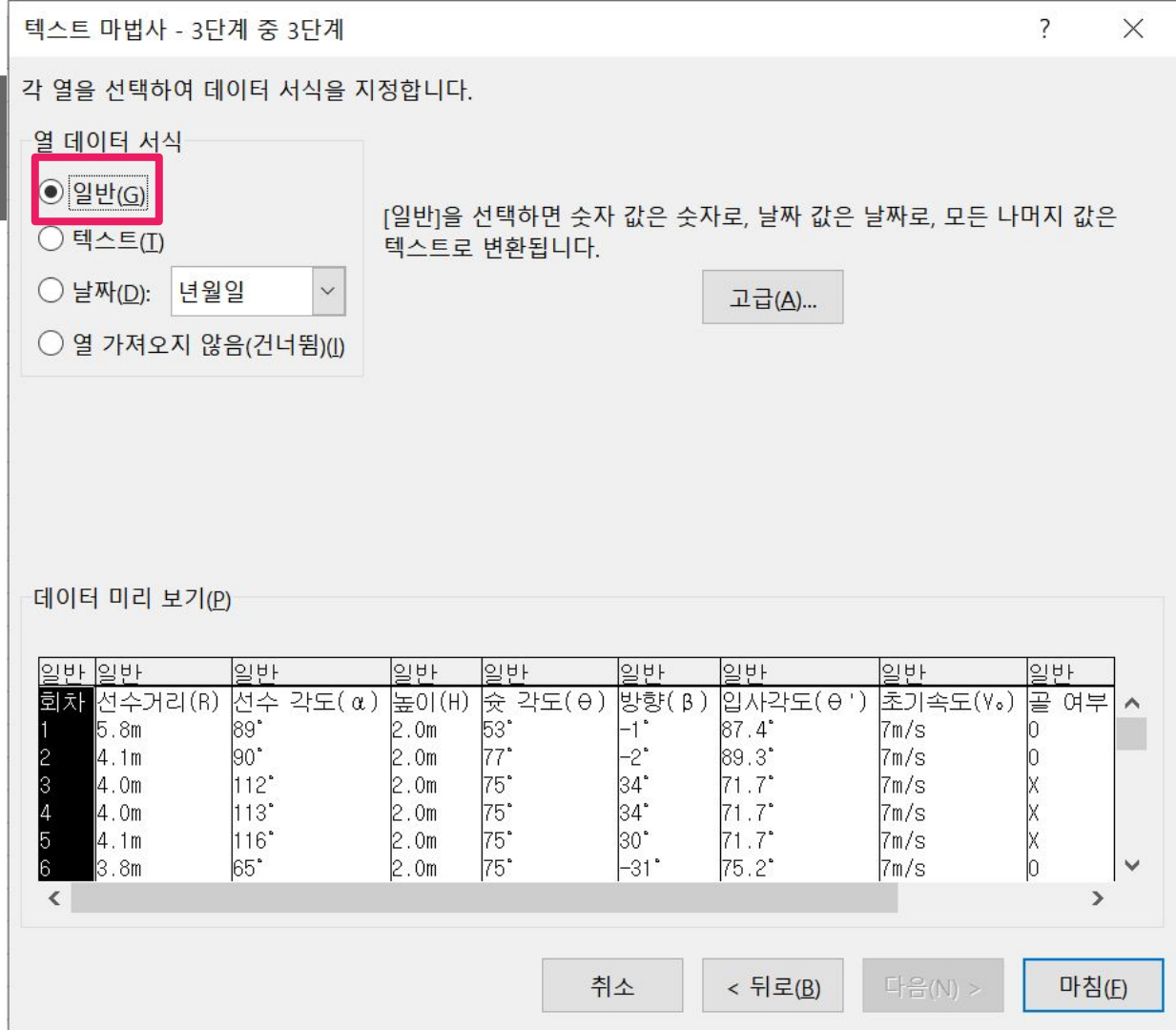
> 위의 내용을 설정하면 자동으로 세로 줄이 그어짐

- 확인 다 했으면 다음!



3. 데이터 서식

- 원래 데이터의 종류에 맞춰서 변환해줌!
- [마침] 누르기



4. 데이터 확인 및 정리하기

- 데이터 확인하기
- 회차, 선수거리, 선수각도, 높이, 슛 각도, 방향, 입사각도, 초기속도 중 독립변수로 설정했던 것 확인!
- 데이터 일반화하기:

선수거리, 선수각도, 높이, 슛 각도, 방향, 입사각도, 초기속도

> 제목에 (단위) 적힌 것 없애기

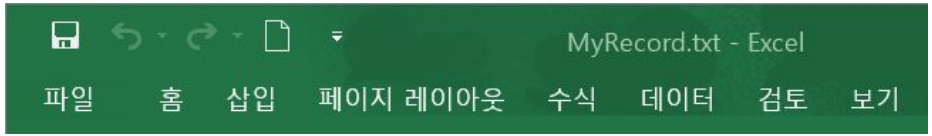
> 내용 또한 숫자로 처리되어야 하므로 단위 없애기

	A	B	C	D	E	F	G	H	I	J	K
1	회차	선수거리(f	선수 각도(	높이(H)	슛 각도(θ)	방향(β)	입사각도(ϕ	초기속도(V	골 여부	점수	누적 점수
2	1	5.8m	89°	2.0m	53°	-1°	87.4°	7m/s	O	1	1
3	2	4.1m	90°	2.0m	77°	-2°	89.3°	7m/s	O	2	3
4	3	4.0m	112°	2.0m	75°	34°	71.7°	7m/s	X	0	3
5	4	4.0m	113°	2.0m	75°	34°	71.7°	7m/s	X	0	3
6	5	4.1m	116°	2.0m	75°	30°	71.7°	7m/s	X	0	3
7	6	3.8m	65°	2.0m	75°	-31°	75.2°	7m/s	O	2	5
8	7	3.9m	115°	2.0m	75°	36°	71.7°	7m/s	X	0	5
9	8	4.0m	114°	2.0m	75°	37°	70.8°	7m/s	X	0	5
10	9	4.0m	114°	2.0m	75°	24°	73.5°	7m/s	X	0	5
11	10	4.0m	114°	2.0m	75°	24°	73.5°	7m/s	X	0	5
12	11	4.2m	109°	2.0m	75°	29°	73.5°	7m/s	X	0	5
13	12	4.1m	110°	2.0m	75°	28°	75.3°	7m/s	O	2	7
14	13	4.3m	78°	2.0m	75°	-6°	83.0°	7m/s	X	0	7
15	14	3.8m	119°	2.0m	75°	28°	0.0°	19m/s	X	0	7
16	15	4.7m	79°	2.0m	30°	-13°	-23.4°	7m/s	X	0	7
17	16	4.7m	79°	2.0m	36°	-13°	1.2°	7m/s	X	0	7
18	17	4.7m	79°	2.0m	35°	-13°	-21.6°	7m/s	X	0	7
19	18	3.7m	85°	2.0m	38°	-27°	0.0°	7m/s	X	0	7
20	19	5.7m	85°	2.0m	60°	-28°	0.0°	7m/s	X	0	7
21	20	3.8m	118°	2.0m	60°	21°	0.0°	7m/s	X	0	7

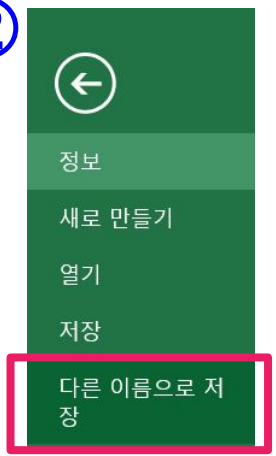
5. 저장하기 1

- [파일]-[다른 이름으로 저장]-[찾아보기]-저장할 곳 선정

①



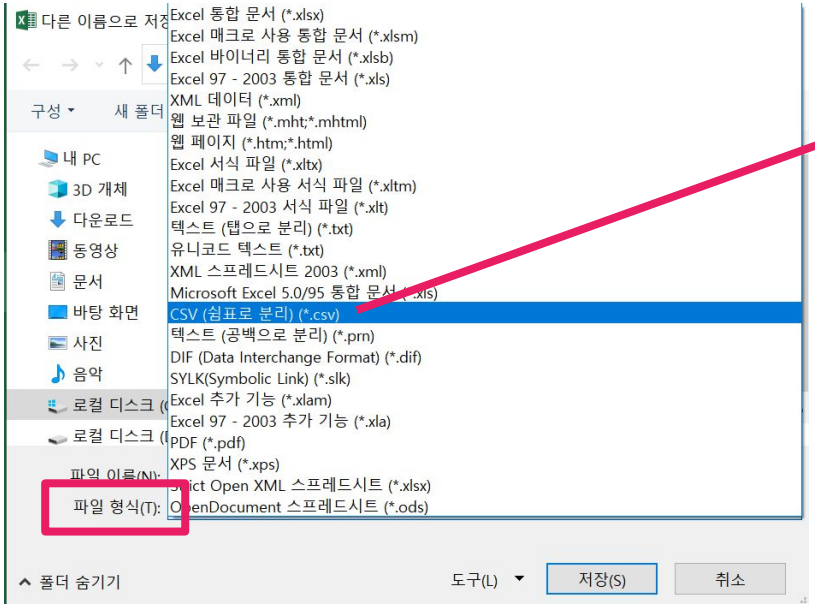
②

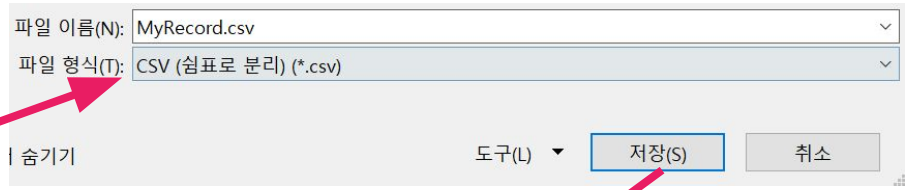


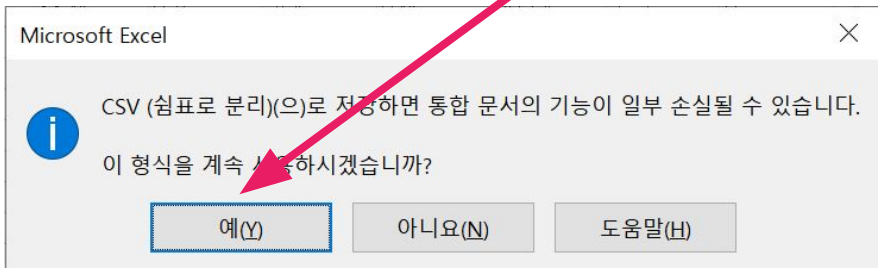
③

5. 저장하기 2

- [파일형식]-[CSV(쉼표로 분리)(*.csv)]-[저장]
- 이 형식을 계속 사용하시겠습니까? > [예(Y)]



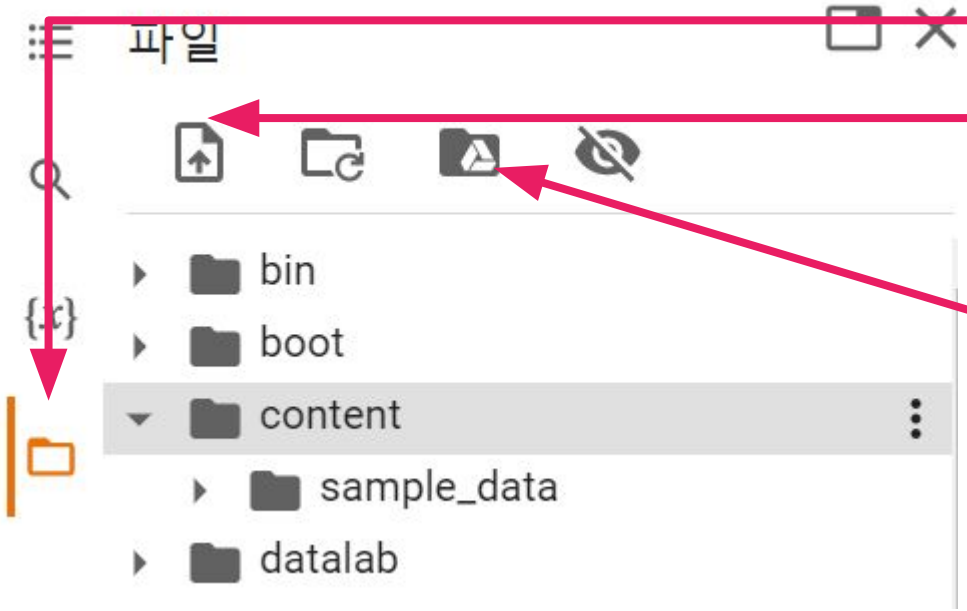




Colab에 데이터 가지고 오기

[임시 데이터용] 데이터 업로드 1

- 해당 방법은 데이터를 임시로 사용하는 것으로 다시 켜면 없어져있습니다!



1. 파일 리스트 열기

2. 파일 업로드

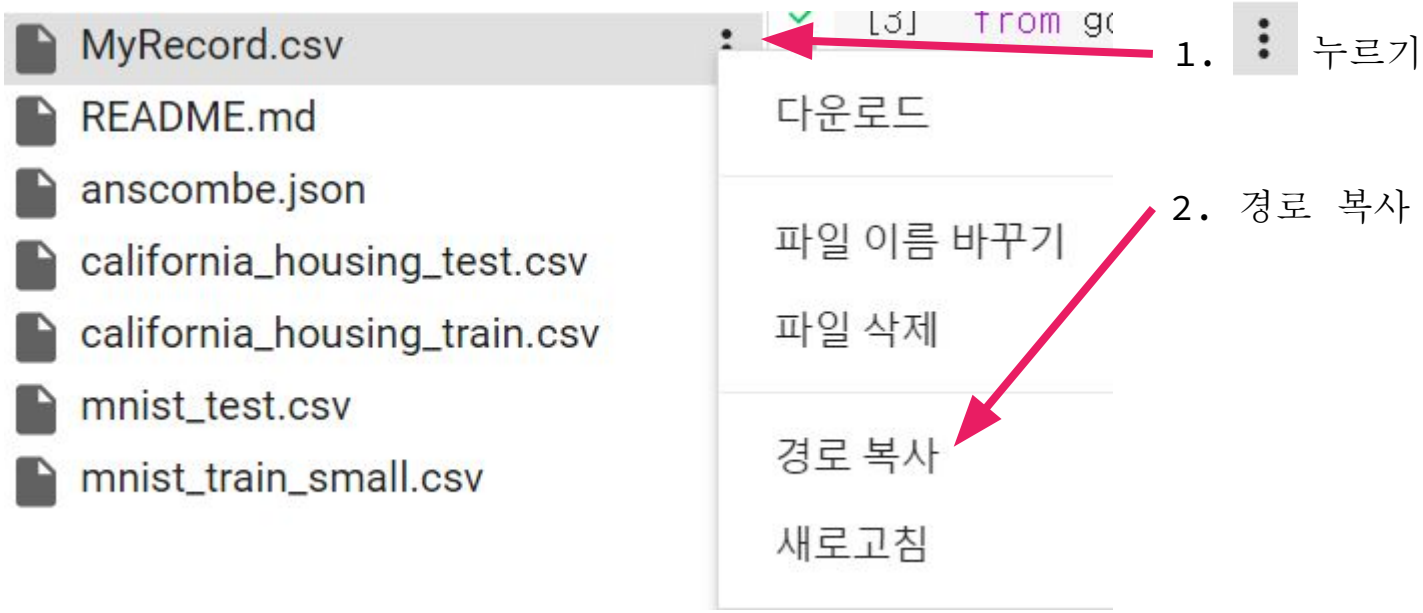
3. 구글 마운트(연결)하기

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

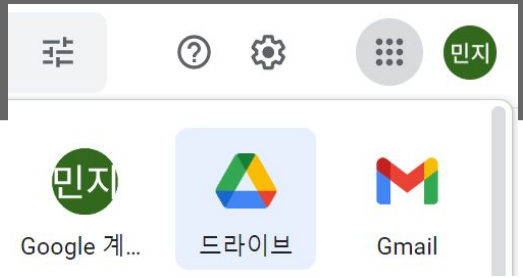
[임시 데이터용] 데이터 업로드 2

- 해당 방법은 데이터를 임시로 사용하는 것으로 다시 켜면 없어져있습니다!

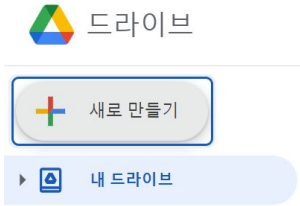


데이터 업로드1

1. 구글드라이브에 들어가서



2. 새로 만들기



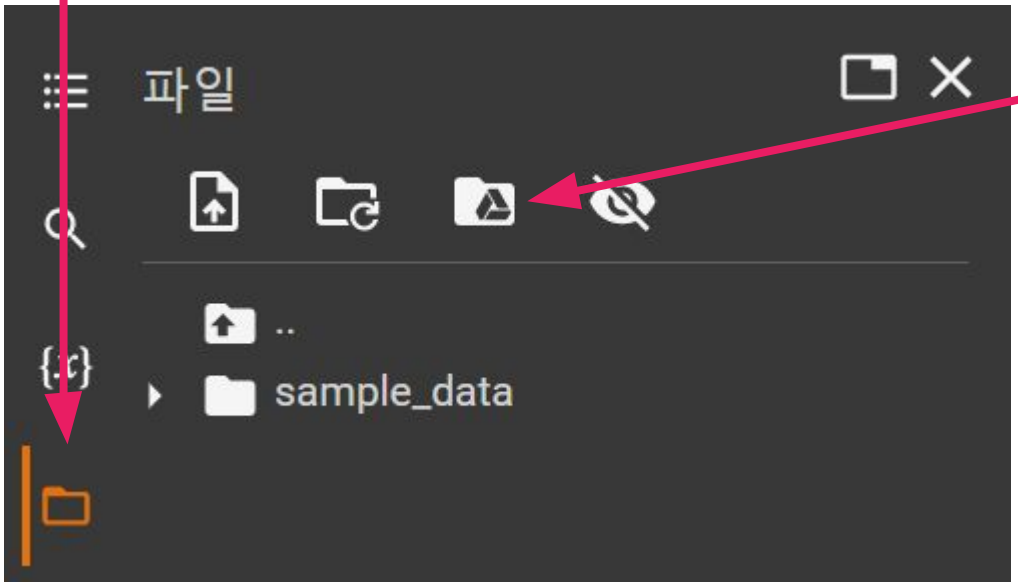
▶ 새 폴더 ▶ 폴더명 data

3. data 폴더에 들어가서 내 데이터 업로드

파일을 여기 끌어다 놓거나

'새로 만들기' 버튼을 사용하세요.

데이터 업로드 2



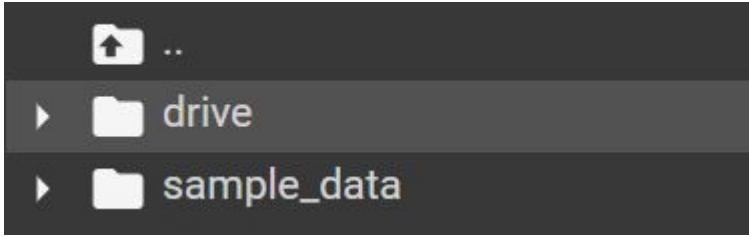
4. 파일 리스트 열기

5. 구글 마운트(연결)하기

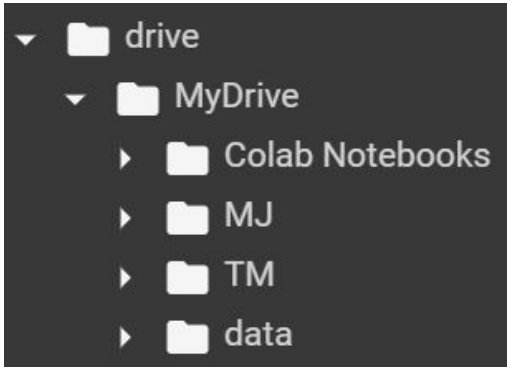
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

6. 연결되면, drive 생김



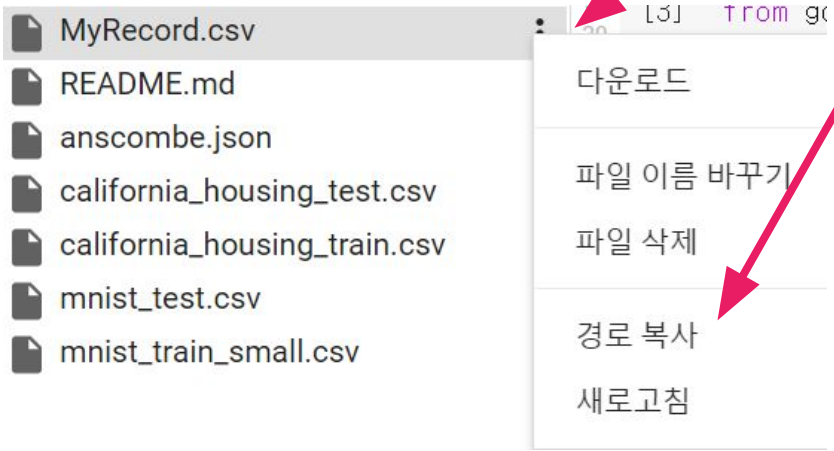
데이터 업로드 3



1. [drive] - [MyDrive] - [data]

2.  누르기

3. 경로 복사



데이터 다뤄보기

1. 라이브러리 사용

```
import tensorflow as tf
import pandas as pd
```

2. 구글 드라이브와 연결

```
from google.colab import drive
drive.mount('/content/drive')
```

3. 데이터 준비하기

```
파일경로 = '/content/drive/MyDrive/data/MyRecord1.csv'
골성공여부 = pd.read_csv(파일경로)
골성공여부
```

실행해보고 안되면, 아래와 같이 바꿔서 해보기

```
pd.read_csv(파일경로, encoding = 'cp949')
```

4. 원핫인코딩

```
인코딩 = pd.get_dummies(골성공여부)
인코딩.head()
```

골 여부_0	골 여부_X
1	0
1	0
0	1
0	1
0	1

5. 열이름 출력

```
print(인코딩.columns)
```

6. 독립변수, 종속변수

```
독립 = 인코딩[['독립변수1', '독립변수2', '독립변수3']]
종속 = 인코딩[['골 여부_O', '골 여부_X']]
print(독립.shape, 종속.shape)
```

인공지능 모델 설계하기

1. 라이브러리 가지고 오기

```
import tensorflow as tf
import numpy as np
import pandas as pd #자료를 표의 형태(csv, 엑셀파일)로 가져올 수도 있어서 추가함
from tensorflow import keras
from tensorflow import layers #층을 여러개 쌓으려고 가지고 옴
from pandas import read_csv #csv 파일 처리를 위함(API나 다운로드 받는 자료들이 대부분 csv 파일로 되어있을 것)
```

2. 독립변수와 종속변수 반영_ 방법1

- 지난 시간에 했던 것처럼 직접 입력하기

```
xs = np.array([-1.0, 0.0, 1.0, 2.0, 3.0, 4.0], dtype = float)
ys = np.array([-3.0, -1.0, 1.0, 3.0, 5.0, 7.0], dtype = float)
```

2. 독립변수와 종속변수 반영_ 방법2

- csv 파일을 드라이브에 업로드하고, 파일에 있는 내용을 가져와 사용하기

1. 구글 드라이브 연결하기

```
from google.colab import drive
drive.mount('/content/drive')
```

2. csv 파일 읽어오기

```
자료 = pd.read_csv('/content/drive/ #파일경로')

열_이름_독립변수 = 자료[[#파일 안을 확인해서 독립변수에 해당하는 열이름 적어주기]]

열_이름_종속변수 = 자료[[#파일 안을 확인해서 종속변수에 해당하는 열이름 적어주기]]

      예)  독립변수 = 자료[‘나이’, ‘성별’]

           종속변수 = 자료[‘키’]
```

3. 확인하기

```
print(열_이름_독립변수.shape, 열_이름_종속변수.shape)
```

2. 독립변수와 종속변수 반영_ 방법3

- github 파일 이용하기

```
파일경로 =
'https://raw.githubusercontent.com/blackdew/tensorflow1/master/csv/boston.csv'
보스턴 = pd.read_csv(파일경로)
```

3. 모델 생성하기_ 방법1

```
model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(10)
])
```

3. 모델 생성하기_ 방법2

```
model = tf.keras.Sequential([
# 64개의 유닛을 가진 완전 연결 층을 모델에 추가합니다:
layers.Dense(64, activation='relu', input_shape=(32,)),
# 또 하나를 추가합니다:
layers.Dense(64, activation='relu'),
# 10개의 출력 유닛을 가진 소프트맥스 층을 추가합니다:
layers.Dense(10, activation='softmax')])
```

3. 모델 생성하기_ 방법3

```
X = tf.keras.layers.Input(shape=[13])
H = tf.keras.layers.Dense(8, activation='swish')(X)
H = tf.keras.layers.Dense(8, activation='swish')(H)
H = tf.keras.layers.Dense(8, activation='swish')(H)
Y = tf.keras.layers.Dense(1)(H)
model = tf.keras.models.Model(X, Y)
```

4. 모델 컴파일러 정해주기

```
1.  model.compile(optimizer='sgd', loss='mse')
2.  model.compile(loss='categorical_crossentropy',
    metrics='accuracy')
3.  model.compile(loss='mse')
4.  model.compile(optimizer='adam',
                  loss='mse',
                  metrics=['accuracy'])
5.  model.compile(optimizer=tf.keras.optimizers.Adam(),
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_l
    ogits=True), metrics=['accuracy'])
```