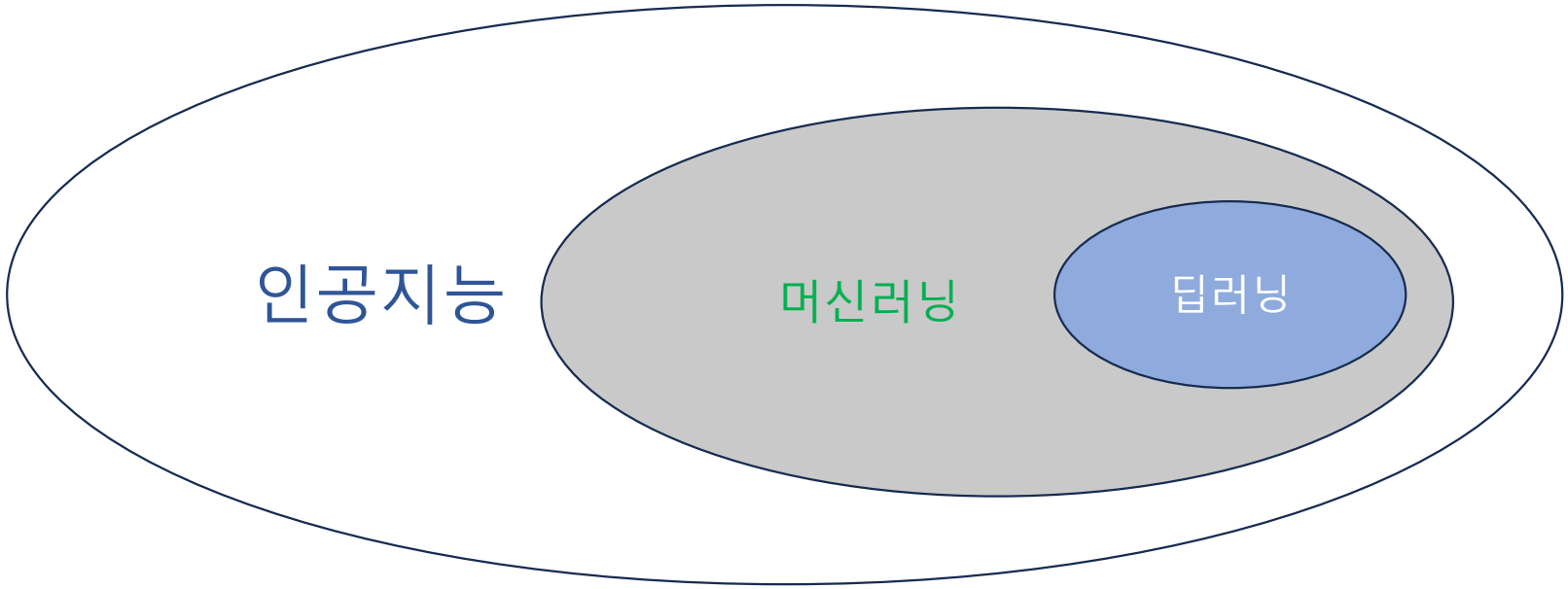
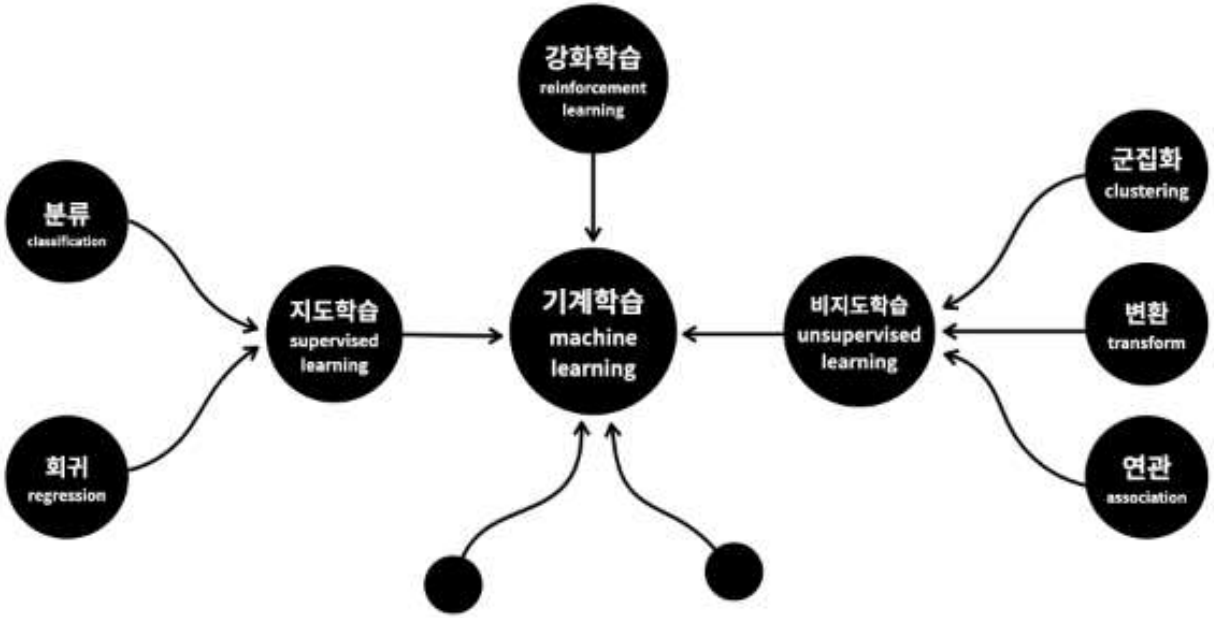
머신러닝
Machine learning

기계가 명시적으로 코딩되지 않은 동작을 스스로 학습해 수행하게 하는 연구 분야



머신러닝의 분류


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

- 지도학습 Supervised Learning

온도 X 2 = 판매량

원인 → 결과

날짜	요일	온도	판매량
2020.1.3	금	20	40
2020.1.4	토	21	42
2020.1.5	일	22	44
2020.1.6	월	23	46
2020.1.7	화	24	48
2020.1.8	수	25	50

과거의 데이터

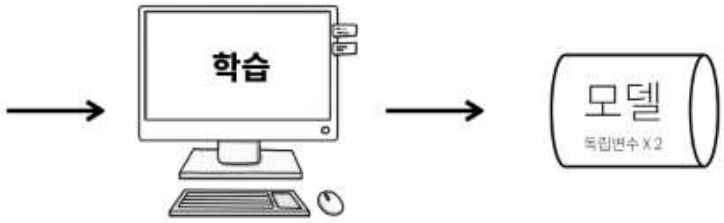
미지의 데이터

Model 생성

온도 X 2 = 판매량

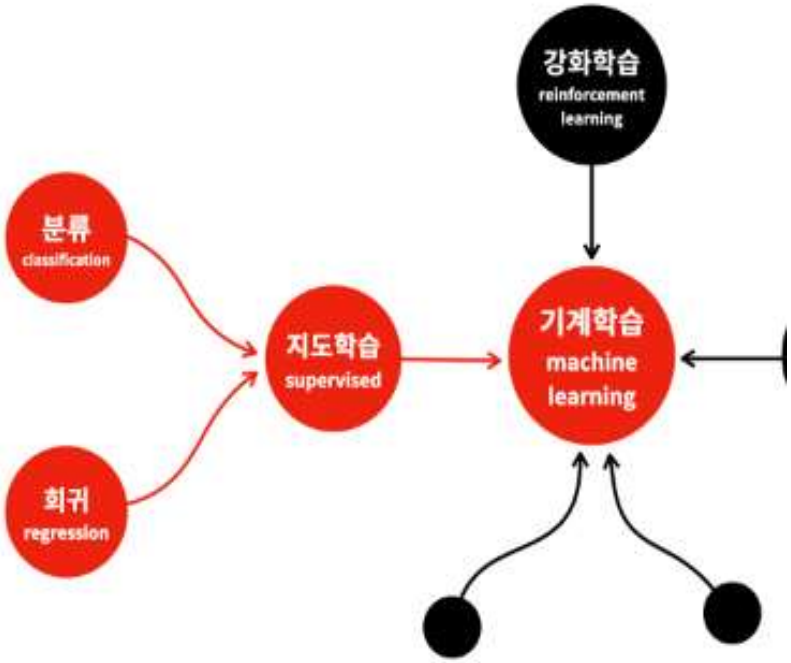
독립변수 종속변수

온도	판매량
20	40
21	42
22	44
23	46



- 지도학습 Supervised Learning

회귀 Regression=>숫자 예측



독립변수	종속변수	학습시킬 데이터를 만드는 방법
공부시간	시험점수 (10점, 20점)	사람들의 공부시간을 입력받고 점수를 확인한다.
온도	레모네이드 판매량	온도와 그날의 판매량을 기록한다.
역세권, 조망 등	집 값	집과 역까지의 거리, 수치화된 조망의 평점 등을 집 값과 함께 기록한다
온실 기체량	기온 변화량	과거에 배출된 온실 기체량과 기온의 변화량을 기록한다.
자동차 속도	충돌 시 사망 확률	충돌시 속도와 사상자를 기록한다.
나이	키	학생들의 나이에 따른 키를 기록한다.

- 지도학습 Supervised Learning

분류 Classification=>이름 or 문자

독립변수	종속변수	학습시킬 데이터를 만드는 방법
공부시간	합격 여부 (합격/불합격)	사람들의 공부시간을 입력받고, 최종 합격여부를 확인한다.
X-ray 사진과 영상 속 종양의 크기, 두께	악성 종양 여부 (양성/음성)	의학적으로 양성과 음성이 확인된 사진과 영상 데이터를 모은다.
품종, 산도, 당도, 지역, 연도	와인의 등급	소믈리에를 통해서 등급이 확인된 와인을 가지고 품종, 산도 등의 독립변수를 정하고 기록한다.
키, 몸무게, 시력, 지병	현역, 공익, 면제	키, 몸무게, 시력, 지병 등을 토대로 현역, 공익, 면제인지를 확인한다.
메일 발신인, 제목, 본문 내용 (사용된 단어, 이모티콘 등)	스팸 메일 여부	이제까지 받은 메일을 모으고, 이들을 스팸 메일과 일반 메일로 구분한다.
고기의 지방함량, 지방색, 성숙도, 육색	소고기 등급	소고기의 정보를 토대로 등급을 측정한다.

Tensorflow의 설치 – anaconda

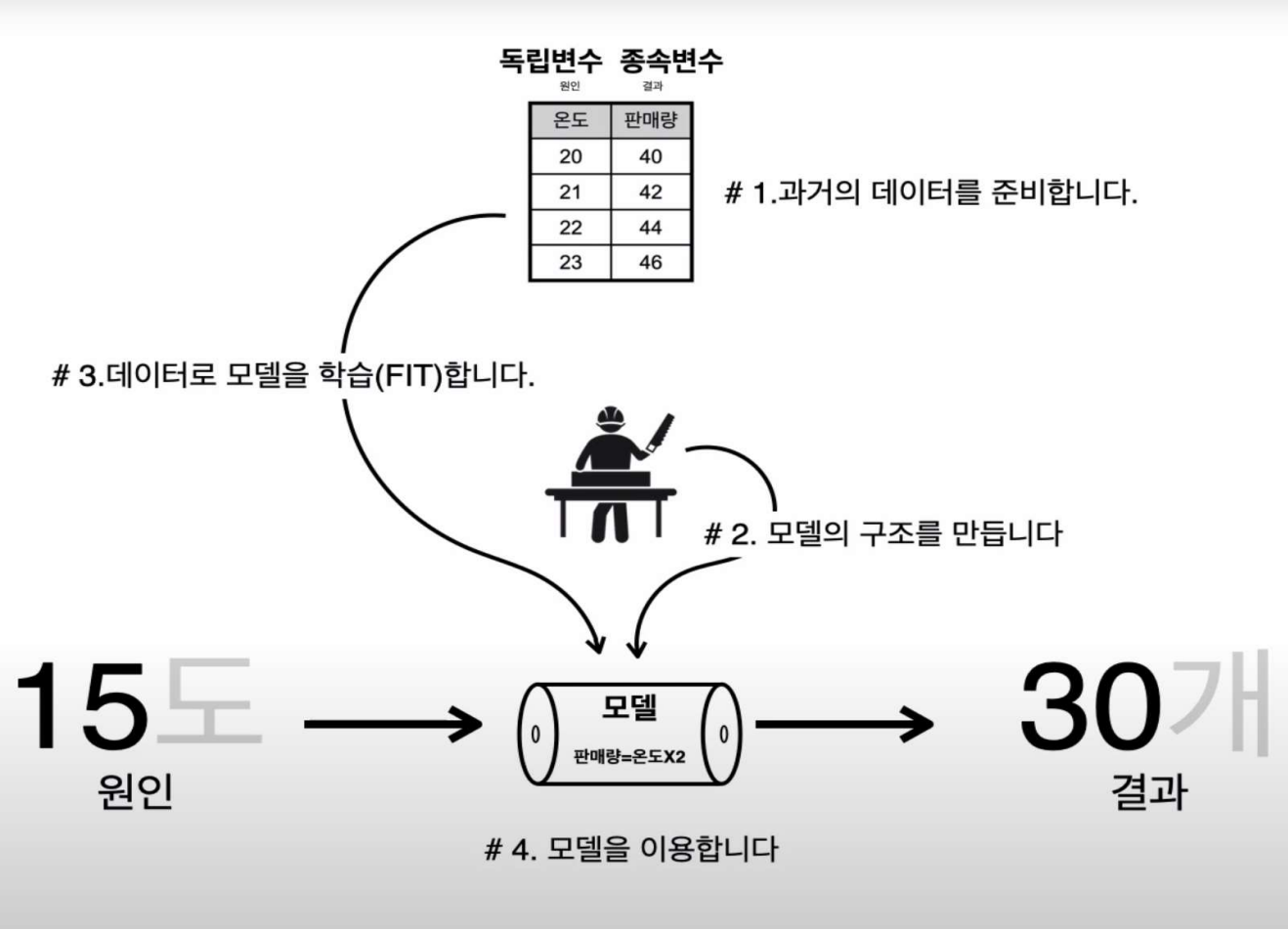
- 가상 환경 생성
  - 가상환경확인: conda env list
  - conda create –n 가상환경이름 python=3.8 anaconda
    anaconda포함 패키지(numpy, pandas, jupyter, matplotlib…)
  - (base) conda install nb_conda_kernels =>주피터 커널 맵핑 확장 프로그램

- 가상환경 활성화 / 비활성화
  - conda activate 가상환경 이름 *# 활성화*
  - (가상환경) conda deactivate *# 비활성화*

- Tensorflow 설치
  - (가상환경) python -m pip install --upgrade pip
  - (가상환경) pip install --upgrade tensorflow(==버전) or conda install tensorflow(==버전)

- Jupyter notebook설치 및 구성
  - conda install jupyter notebook
  - 홈 디렉토리 변경
    - jupyter notebook --generate-config
    - c.NotebookApp.notebook_dir = ''

머신러닝의 지도 학습(회귀) 과정



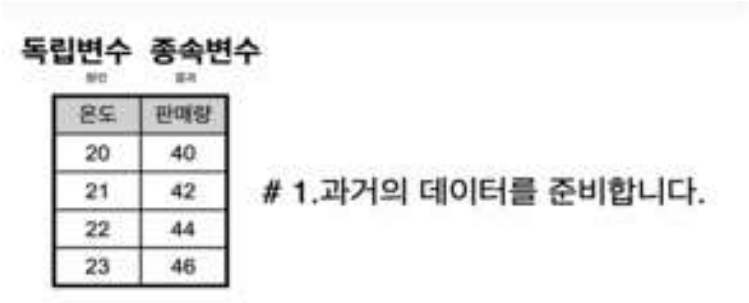
#1.과거의 데이터를 준비합니다.

레모네이드 = pd.read_csv('lemonade.csv')

독립변수, 종속변수

독립 = 레모네이드[['온도']]

종속 = 레모네이드[['판매량']]



독립변수 종속변수

온도	판매량
20	40
21	42
22	44
23	46

1.과거의 데이터를 준비합니다.

#2.모델의 구조를 만듭니다.

X = tf.keras.layers.Input(shape=[1])

Y = tf.keras.layers.Dense(1)(X)

model = tf.keras.models.Model(X, Y)

model.compile(loss='mse')

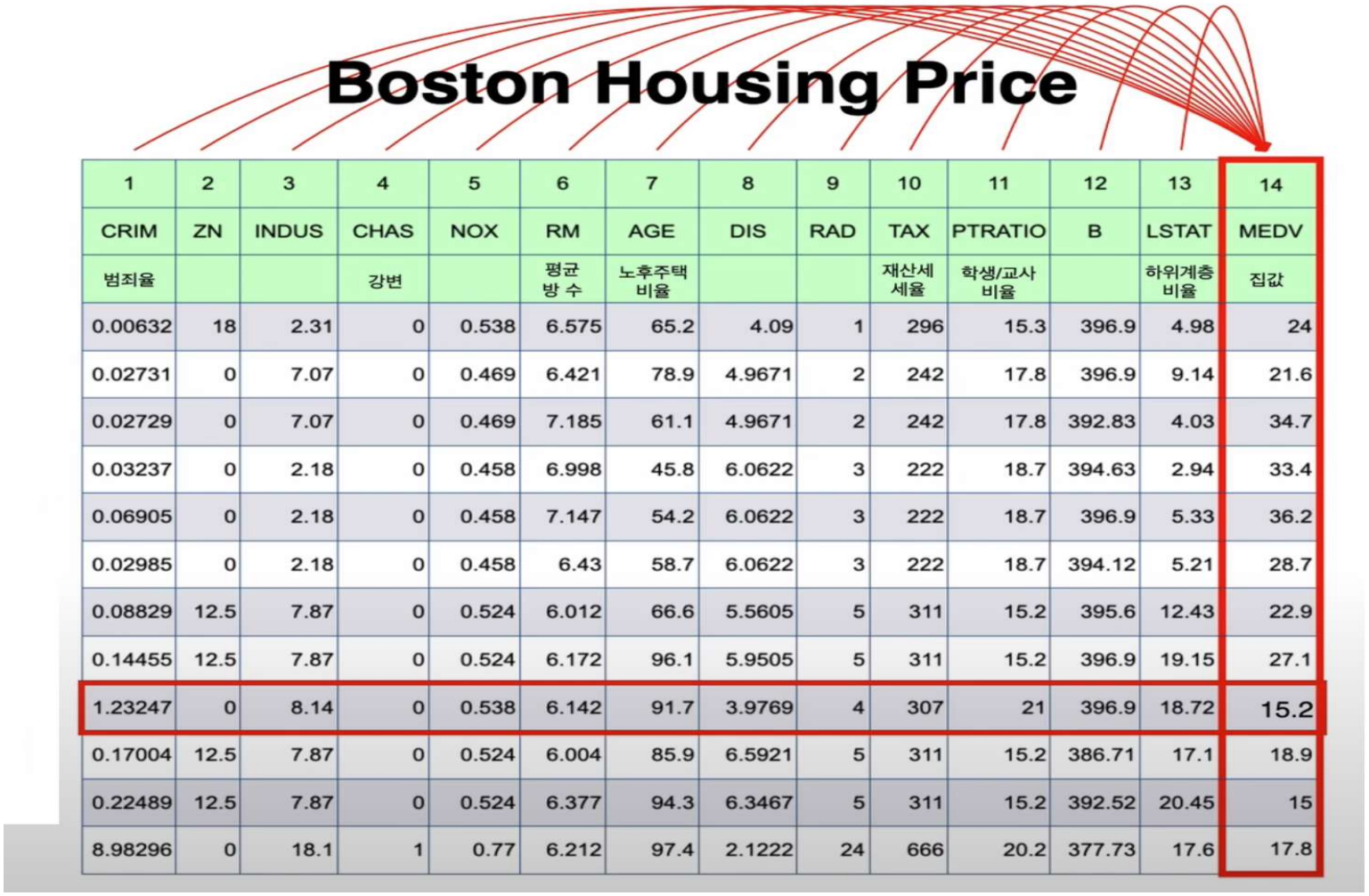
#3.데이터로 모델을 학습(FIT)합니다.

model.fit(독립, 종속, epochs=1000, verbose=0)

#4.모델을 이용합니다.

print(model.predict([[15]]))

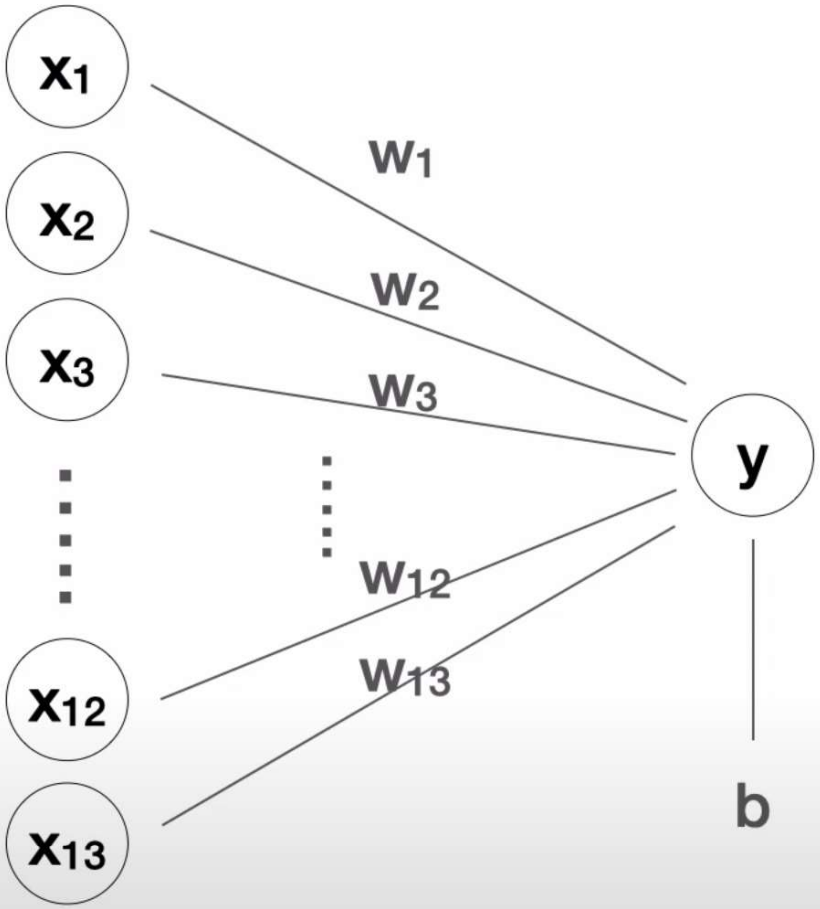
보스턴 집값예측



1	2	3	4	5	6	7	8	9	10	11	12	13	14
CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
범죄율			강변		평균 방 수	노후주택 비율			재산세 세율	학생/교사 비율		하위계층 비율	집값
0.00632	18	2.31	0	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98	24
0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	21.6
0.02729	0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
0.06905	0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.9	5.33	36.2
0.02985	0	2.18	0	0.458	6.43	58.7	6.0622	3	222	18.7	394.12	5.21	28.7
0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.6	12.43	22.9
0.14455	12.5	7.87	0	0.524	6.172	96.1	5.9505	5	311	15.2	396.9	19.15	27.1
1.23247	0	8.14	0	0.538	6.142	91.7	3.9769	4	307	21	396.9	18.72	15.2
0.17004	12.5	7.87	0	0.524	6.004	85.9	6.5921	5	311	15.2	386.71	17.1	18.9
0.22489	12.5	7.87	0	0.524	6.377	94.3	6.3467	5	311	15.2	392.52	20.45	15
8.98296	0	18.1	1	0.77	6.212	97.4	2.1222	24	666	20.2	377.73	17.6	17.8

```
X = tf.keras.layers.Input(shape=[13])
Y = tf.keras.layers.Dense(1)(X)
model = tf.keras.models.Model(X, Y)
model.compile(loss='mse')
```

1	2	3	4	5	6	7	8	9	10	11	12	13	14
CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0.00632	18	2.31	0	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98	24
0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	21.6
0.02729	0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4

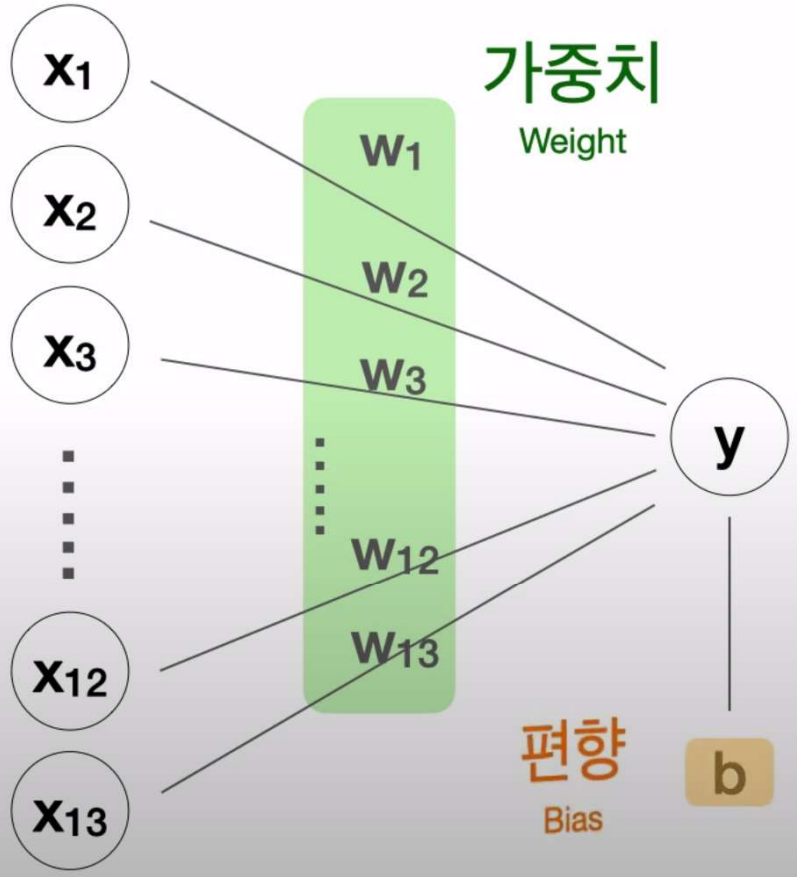


$$y = w_1x_1 + w_2x_2 + \cdots\cdots + w_{13}x_{13} + b$$

$$y = w_1x_1 + w_2x_2 + \cdots\cdots + w_{13}x_{13} + b$$

퍼셉트론

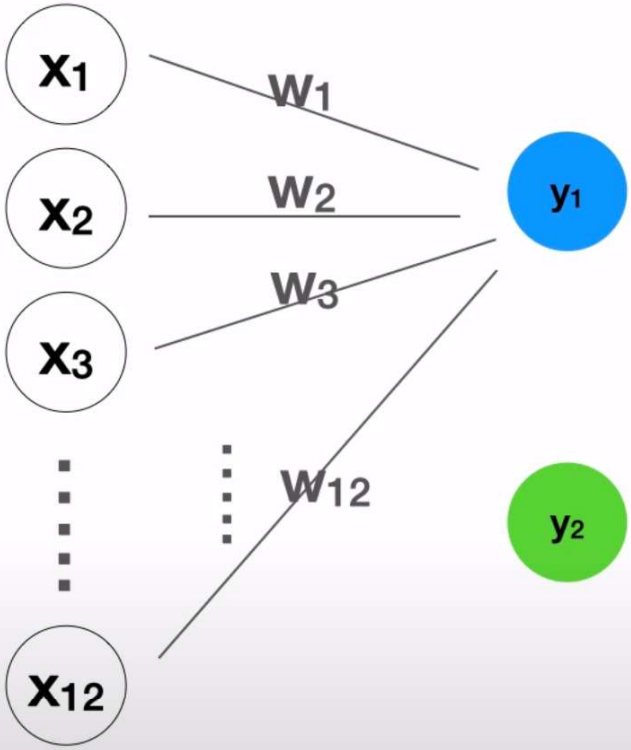
Perceptron



```
X = tf.keras.layers.Input(shape=[12])
Y = tf.keras.layers.Dense(2)(X)
model = tf.keras.models.Model(X, Y)
model.compile(loss='mse')
```

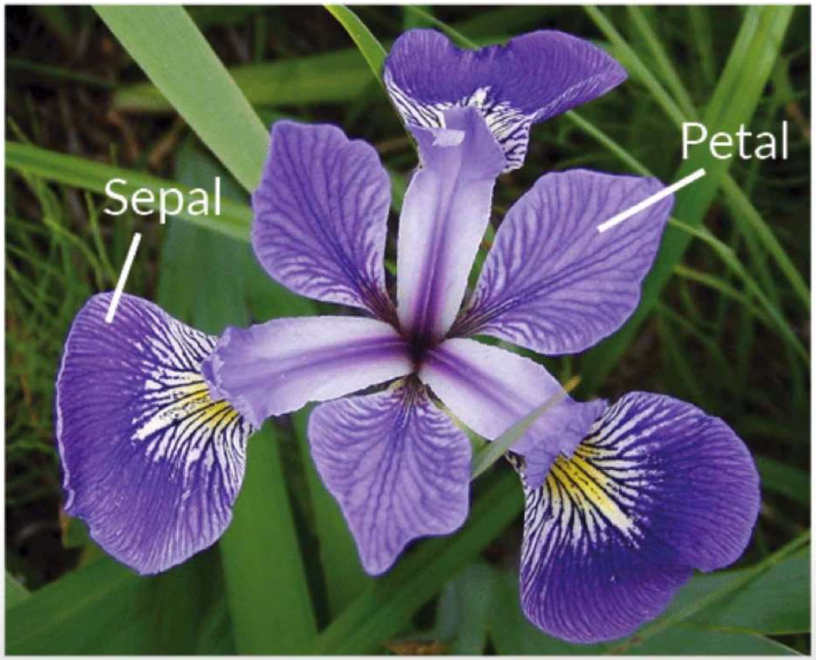
1	2	3	4	5	6	7	8	9	10	11	12	13	14
CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0.00632	18	2.31	0	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98	24
0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	21.6
0.02729	0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4

$$y_1 = w_1x_1 + w_2x_2 + \cdots\cdots + w_{12}x_{12} + b$$



$$y_2 = w_1x_1 + w_2x_2 + \cdots\cdots + w_{12}x_{12} + b$$

아이리스 품종 분류(Classification)



Iris Versicolor

Iris Setosa

Iris Virginica

https://www.datacamp.com/community/tutorials/machine-learning-in-r

#1.과거의 데이터를 준비합니다.

```
아이리스 = pd.read_csv(파일경로)
아이리스 = pd.get_dummies(아이리스)

# 독립변수, 종속변수
독립= 아이리스[['꽃잎길이', '꽃잎폭', '꽃받침길이', '꽃받침폭']]
종속= 아이리스['품종']
```

#2.모델의 구조를 만듭니다.

```
X = tf.keras.layers.Input(shape=[4])
Y = tf.keras.layers.Dense(3,activation='softmax')(X)
model = tf.keras.models.Model(X, Y)
model.compile(loss='categorical_crossentropy)
```

#3.데이터로 모델을 학습(FIT)합니다.

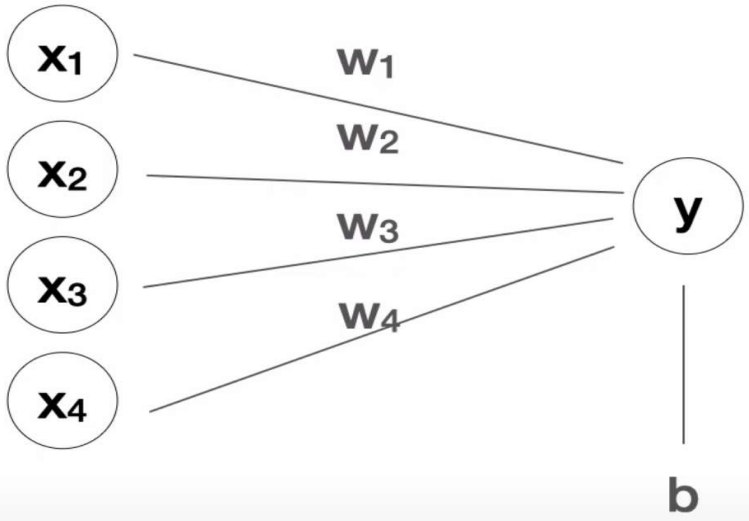
```
model.fit(독립, 종속, epochs=1000, verbose=0)
```

#4.모델을 이용합니다.

```
print(model.predict(독립[:5]))
```

종속변수=범주형 데이터

꽃잎길이	꽃잎폭	꽃받침길이	꽃받침폭	품종
5.1	3.5	1.4	0.2	setosa
4.9	3	1.4	0.2	versicolor
4.7	3.2	1.3	0.2	virginica
4.6	3.1	1.5	0.2	setosa



$$y = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b$$

원핫인코딩

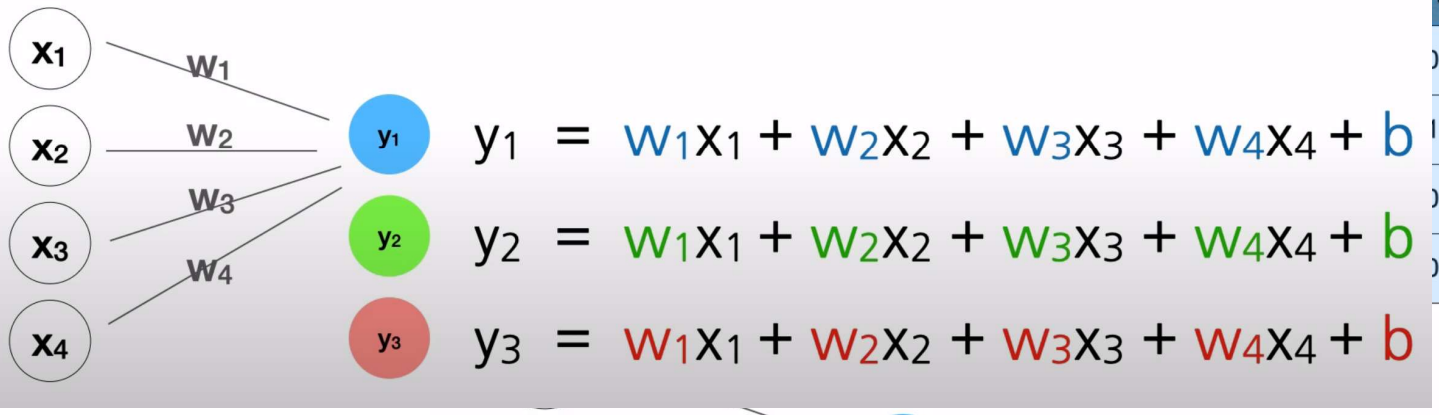
꽃잎길이	꽃잎폭	꽃받침길이	꽃받침폭	품종
5.1	3.5	1.4	0.2	setosa
4.9	3	1.4	0.2	versicolor
4.7	3.2	1.3	0.2	virginica
4.6	3.1	1.5	0.2	setosa

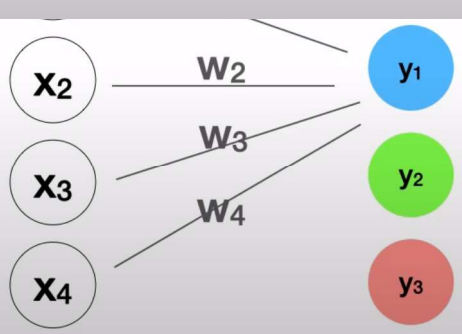
```
# 학습할 데이터를 준비합니다.
아이리스 = pd.read_csv('iris.csv')

# 원핫인코딩
아이리스 = pd.get_dummies(아이리스)
```

꽃잎길이	꽃잎폭	꽃받침길이	꽃받침폭	품종.setosa	품종.virginica	품종.versicolor
5.1	3.5	1.4	0.2	1	0	0
4.9	3	1.4	0.2	0	1	0
4.7	3.2	1.3	0.2	0	0	1
4.6	3.1	1.5	0.2	1	0	0

꽃잎길이	꽃잎폭	꽃받침길이	꽃받침폭	품종.setosa	품종.virginica	품종.versicolor
5.1	3.5	1.4	0.2	1	0	0
4.9	3	1.4	0.2	0	1	0
4.7	3.2	1.3	0.2	0	0	1
4.6	3.1	1.5	0.2	1	0	0





2. 모델의 구조를 만듭니다

```
X = tf.keras.layers.Input(shape=[4])
Y = tf.keras.layers.Dense(3, activation='softmax')(X)
model = tf.keras.models.Model(X, Y)
model.compile(loss='categorical_crossentropy')
```

분류 예측

30%

99%

50%

Sigmoid

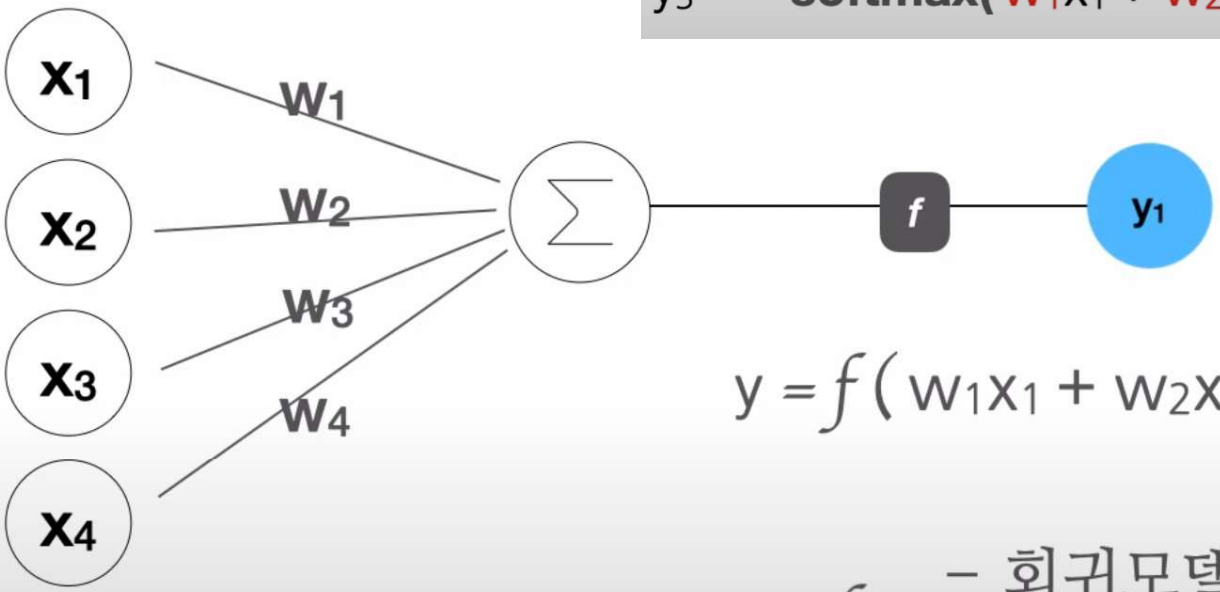
Softmax

0% ~ 100%

$$y_1 = \text{softmax}(w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b)$$

$$y_2 = \text{softmax}(w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b)$$

$$y_3 = \text{softmax}(w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b)$$

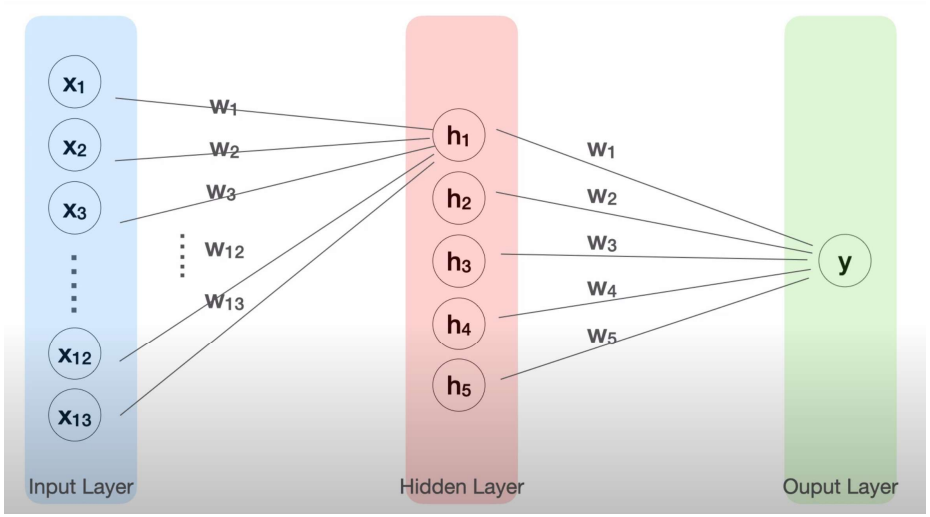


$$y = f(w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b)$$

f

- 회귀모델: Identity (y=x)
- 분류모델: Softmax

인공신경망(딥러닝)의 히든 레이어(Hidden Layer)




```
# 2. 모델의 구조를 만듭니다
X = tf.keras.layers.Input(shape=[13])
H = tf.keras.layers.Dense(5, activation='swish')(X)
Y = tf.keras.layers.Dense(1)(H)
model = tf.keras.models.Model(X, Y)
model.compile(loss='mse')
```

데이터 정리

- 변수(컬럼)타입확인:데이터.dtype
- 변수를 범주형으로 변경:
  - 데이터['칼럼명'].astype('category')
- 변수를 수치형으로 변경:
  - 데이터['칼럼명'].astype('int')
  - 데이터['칼럼명'].astype('float')
- NA 값의 처리:
  - NA갯수 체크:데이터.isna().sum()
  - NA값 채우기:데이터['칼럼명'].fillna(특정숫자)