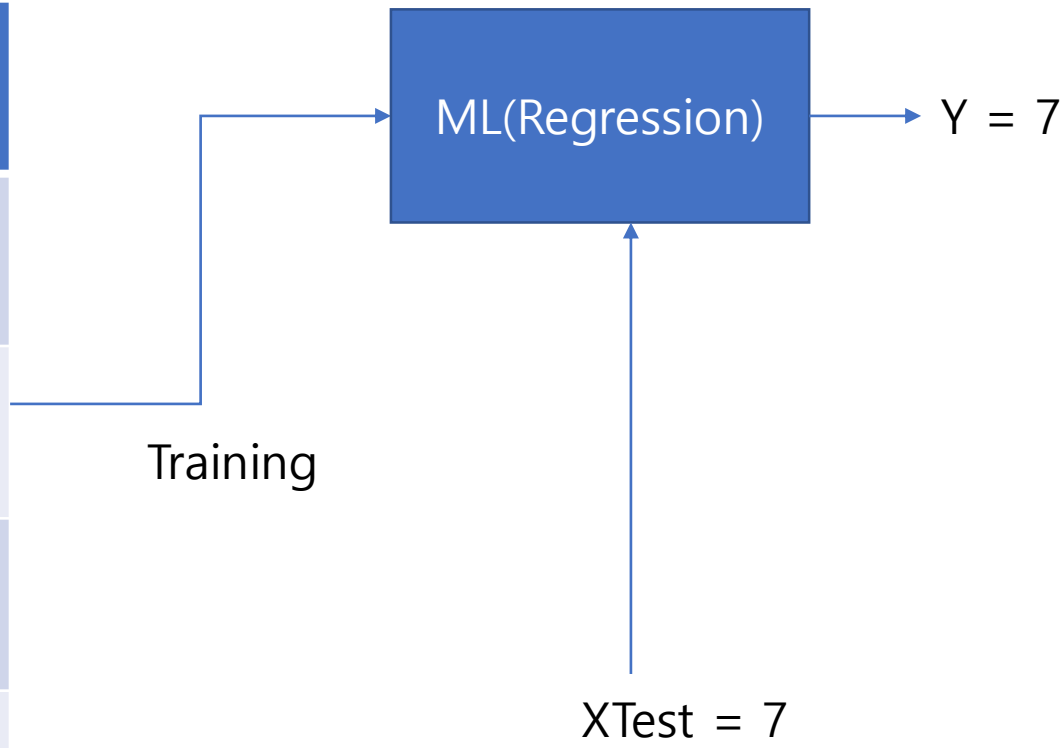
Linear Regression 개념

Predicting exam score: regression

X(hours)	Y(score)
10	90
9	80
3	50
2	30

Training data



Regression(data)

X	Y
1	1
2	2
3	3

(Linear) Hypothesis
Linear 한 모델이 우리가 가진 Data와 맞을 것이다.
Linear 한 선을 찾는 것이 학습!

1. 가설 세우기
H(x) = Wx + b
2. 어떤 선이 Data와 잘 맞을까?
실제 data와 선(가설) 사이의 거리 계산
=Cost(Loss) function
보통 차이의 제곱(음수 제외) (H(x) – y)^2 사용

$H(x) = Wx + b$.

$$\frac{(H(x_1)-y(1))^2 + (H(x_2)-y(2))^2 + (H(x_3)-y(3))^2}{3}$$

예측 값 ↓ 정답 labeled 값 ↓

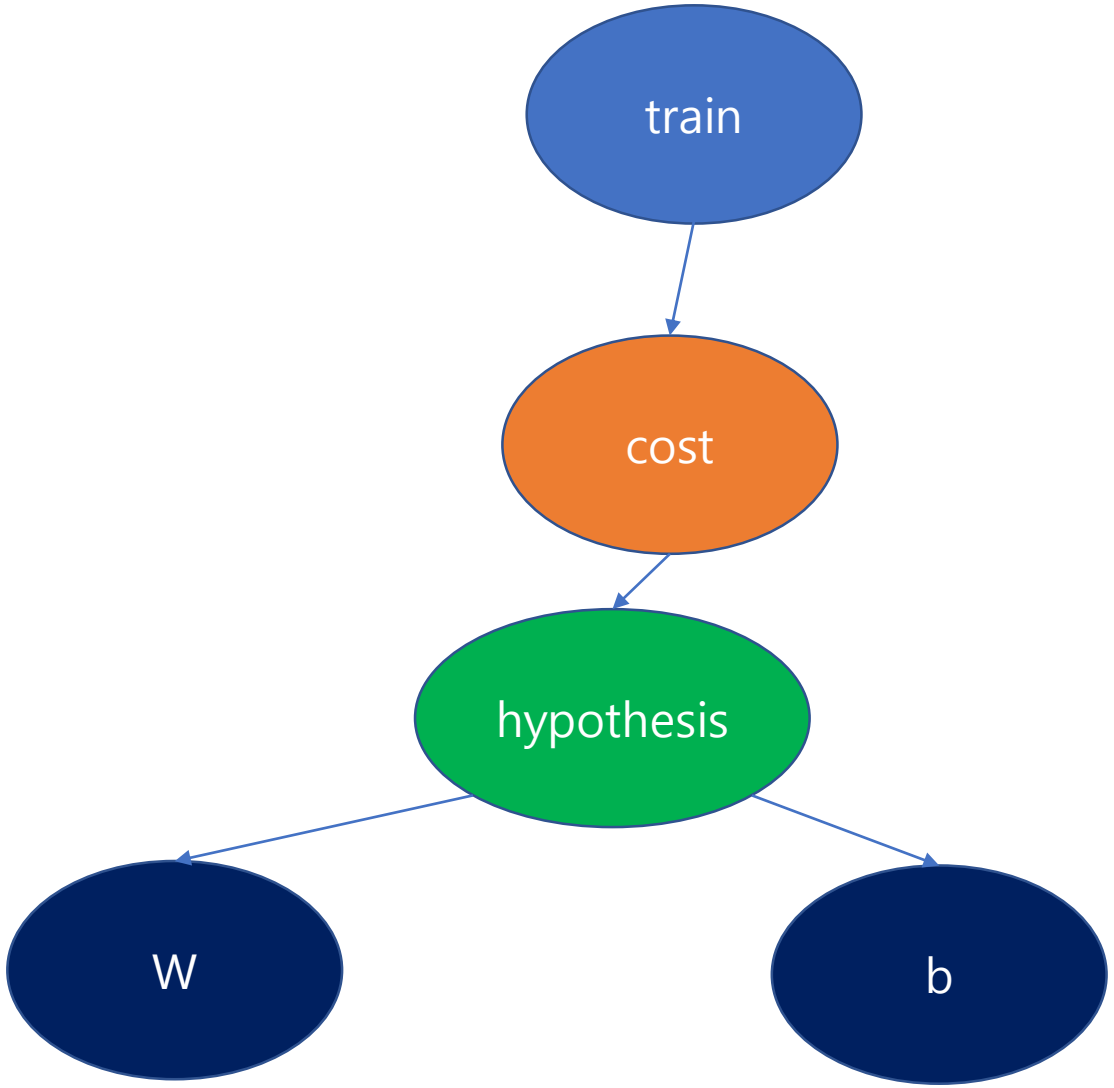
$$\Rightarrow \text{일반화} \quad Cost = \frac{1}{m}\sum_{i=1}^{m}(H(x^{(i)})-y^{(i)})^2, \quad H(x)=Wx+b$$

$$Cost(W,b) = \frac{1}{m}\sum_{i=1}^{m}(H(x^{(i)})-y^{(i)})^2$$

Goal: minimize Cost(W, b)

"W와 b를 최소화"

Tensorflow로 간단한 linear regression 구현




```python
# x_train = [1,2,3]
# y_train = [1,2,3]

# placeholder을 사용하면 값을 직접 넣겨줄 수 있다.
X = tf.placeholder(tf.float32, shape=[None])
Y = tf.placeholder(tf.float32, shape=[None])

# Part1 - Build graph using TF operations
# [rank]
W = tf.Variable(tf.random_normal([1]) , name = 'weight')
b = tf.Variable(tf.random_normal([1]), name = 'bias')

# H(x) = Wx+b
hypothesis = x_train * W + b

# cost(loss) function
cost = tf.reduce_mean(tf.square(hypothesis - y_train))

# minimize
optimizer = tf.train.GradientDescentOptimizer(learning_rate = 0.01)
train = optimizer.minimize(cost)
```

```python
# Part2,3 - Run/update graph and get results
sess = tf.Session()
sess.run(tf.global_variables_initializer())

# for step in range(2001) :
#     sess.run(train)
#     if step % 20 == 0:
#         print(step, sess.run(cost), sess.run(W), sess.run(b))
# "W = 1, b = 0일 때 이상적"

for step in range(2001):
    cost_val, W_val, b_val, _ = sess.run([cost, W, b, train],
                                         feed_dict = {X: [1,2,3,4,5],
                                                      Y:[2.1,3.1,4.1,5.1,6.1]})
    if step % 20 == 0:
        print(step, cost_val, W_val, b_val)

# 검증
print(sess.run(hypothesis, feed_dict = {X: [5]}))
print(sess.run(hypothesis, feed_dict = {X: [2.5]}))
print(sess.run(hypothesis, feed_dict = {X: [1.5,3.5]}))
```

```
[6.1008625]
[3.5993102]
[2.598689  4.5999317]
```