**ML\_Lec 01.**

**About ML.**

1. Limitations of Explicit Programming.

Spam filter : many rules

Automatic driving : too many rules → 고려할 사항이 너무 많음.

1. Machine Learning. //빨간색: 주요 개념

“Field of study that gives computers the ability to learn without being

explicitly programmed.” Arthur Samuel(1959)

머신러닝은 일종의 소프트웨어로, 프로그램 자체가 학습하여 배우는 명령을 갖는 소프트웨어이다.

//분명하지 않은 경우 재정리

**Learning.**

1. Supervised Learning : Learning with labeled examples – Training Set

Most common problem type in ML

1. Image Labeling
2. Email Spam Filter
3. Predicting Exam Score

Types of Supervised Learning

1. 0~100까지의 점수 Regression
2. Pass/Non-pass Binary Classification
3. A, B, C, D, E grade Multi-Label Classification
4. Unsupervised Learning : un-labeled data

Google news grouping

Word clustering

**ML\_Lab 01.**

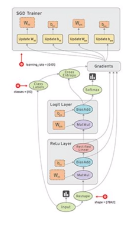
**TensorFlow.**

TensorFlow is an open source software library for numerical computation using data flow graphs.

**Data Flow Graph.**

Data Flow Graph는 노드, 엣지로 연산이 일어나 어떤 작업을 할 수 있는 것이다.

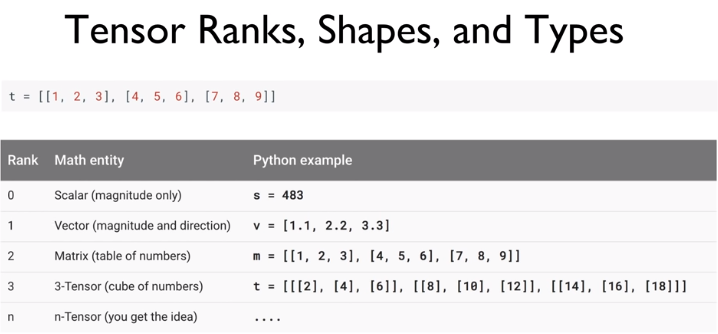
1. Nodes in the graph represent mathematical operations. //연산
2. Edges represent the multidimensional data arrays(tensors) communicated between them. //Tensor



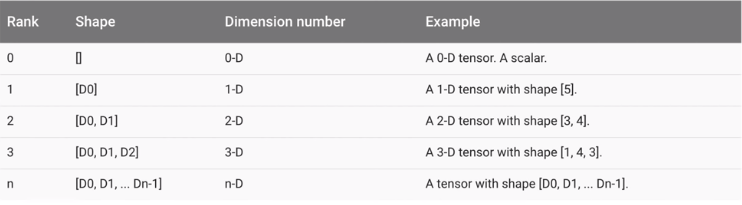
**Tensor.**

임의의 차원을 갖는 배열들을 의미한다.

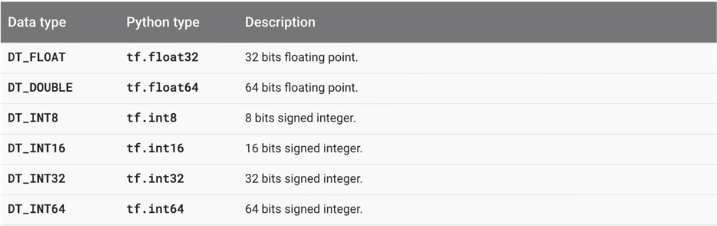
1. Ranks : n차원



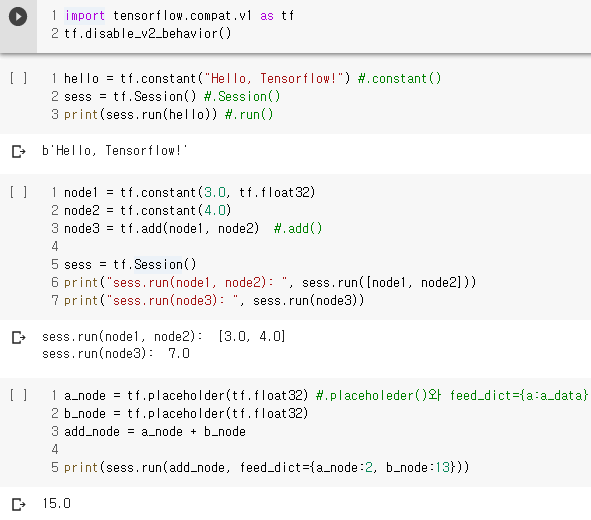
1. Shapes : [가장 바깥 요소의 개수, 가장 안쪽 요소의 개수]



1. Types : 자료형

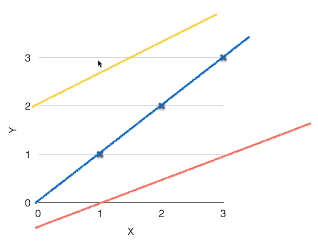


**Test.**



**ML\_Lec 02.**

**Linear Regression. //선형 회귀**



대략 직선(Linear) 형태에 맞을 것이라 가설하고 진행한다.

**Cost function. //비용함수**

1. How fit the line to our (training) data

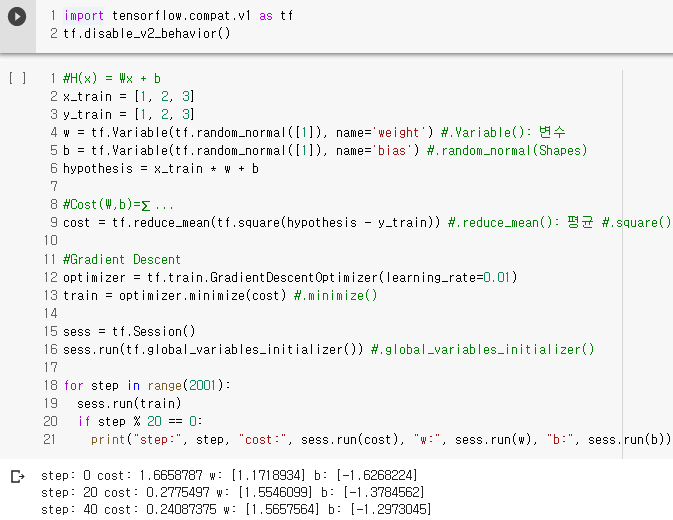
( //± 때문에 제곱

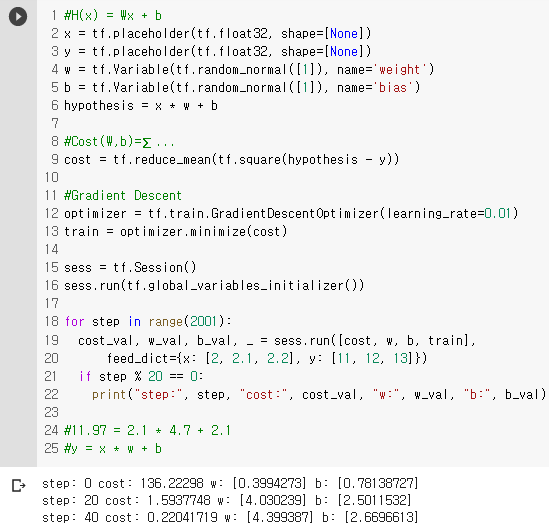
1. Goal : Minimize Cost

minimize

**ML\_Lab 02.**

**Test.**

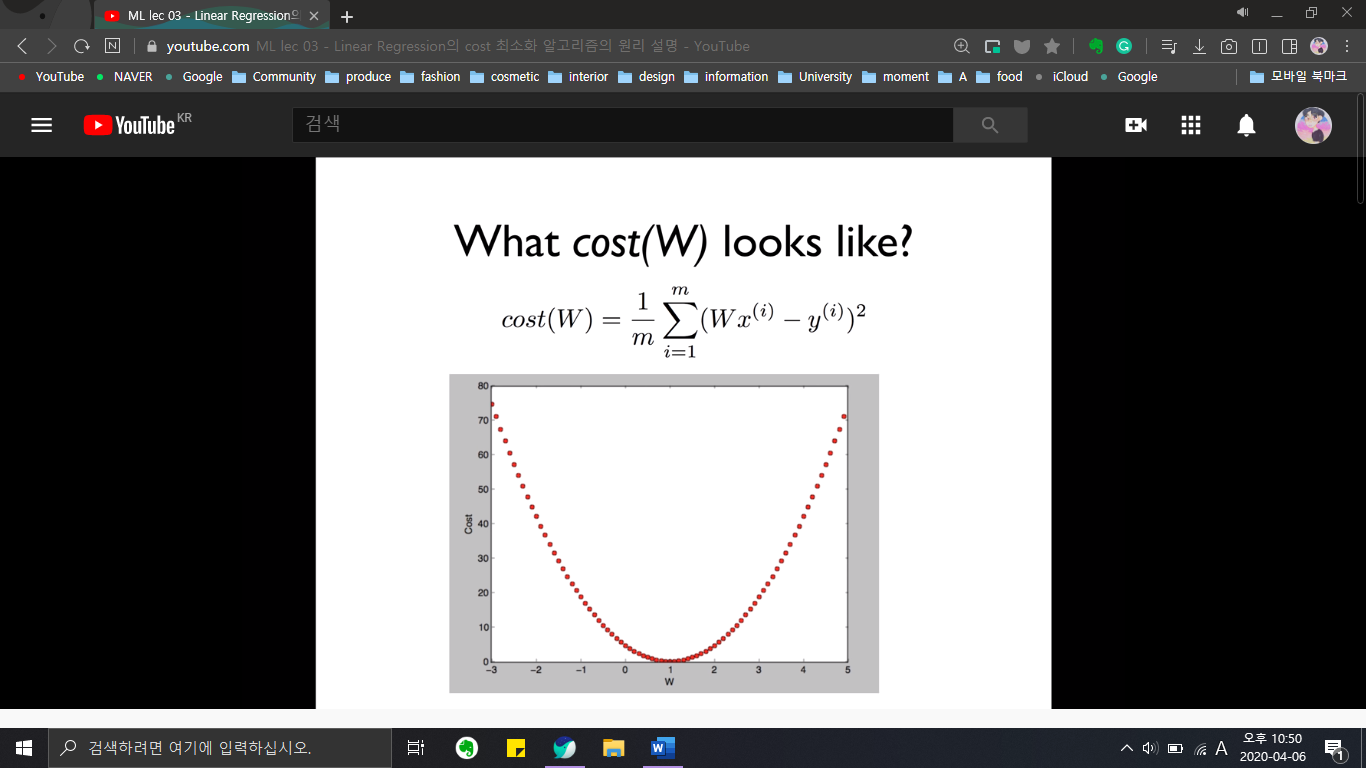




**ML\_Lec 03.**

**Simplified Hypothesis.**

,



**Minimize. //최소화**

Cost가 제일 낮은 값을 찾는 것이다.

**Gradient Descent Algorithm. //경사하강법**

1. Minimize cost function.
2. Gradient descent is used many minimization problems.
3. For a given cost function, cost(W, b), it will find W, b to minimize cost.
4. It can be applied to more general function: cost(w1, w2, …).

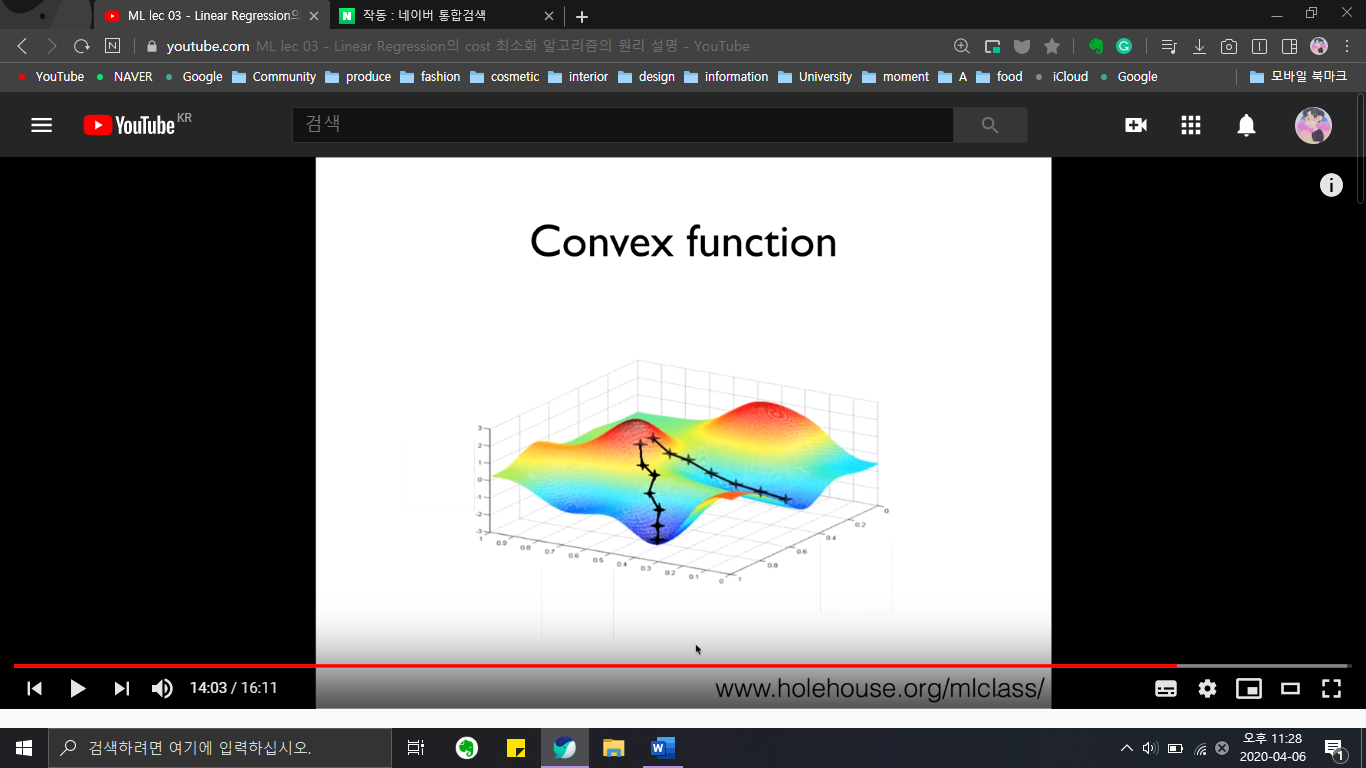
**Gradient Descent Algorithm Works.**

1. Start with initial guesses
2. Start at 0,0 (or any other value)
3. Keeping changing W and b a little bit to try and reduce cost(W, b)
4. Each time you change the parameters, you select the gradient which reduces cost(W, b) the most possible
5. Repeat
6. Do so until you converge to a local minimum
7. Has an interesting property

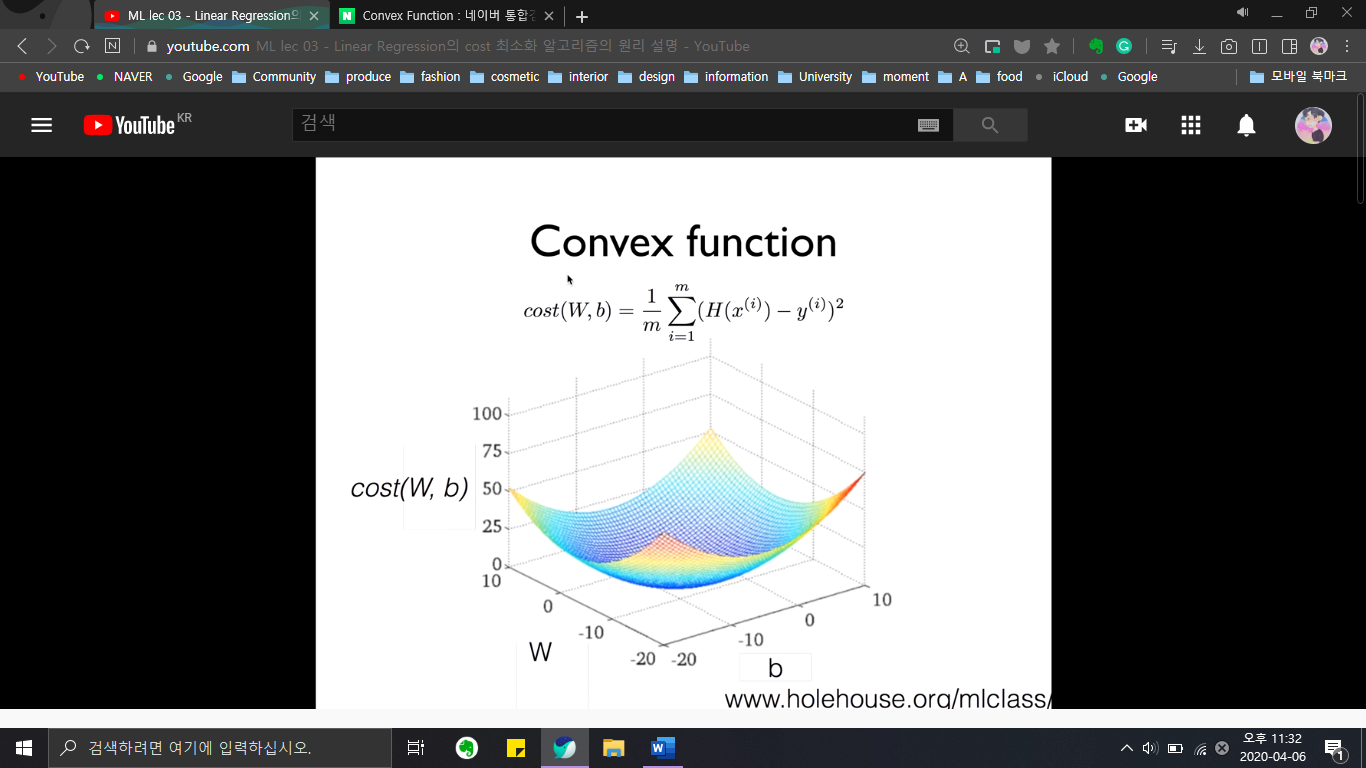
♠ Where you start can determine which minimum you end up

**Formal Definition. //미분**

**Convex Function. //볼록함수**

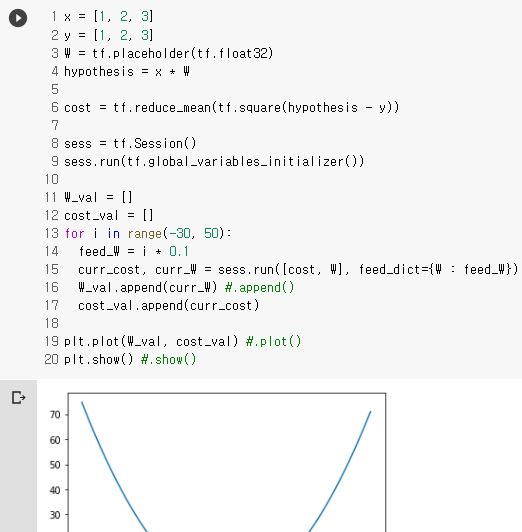


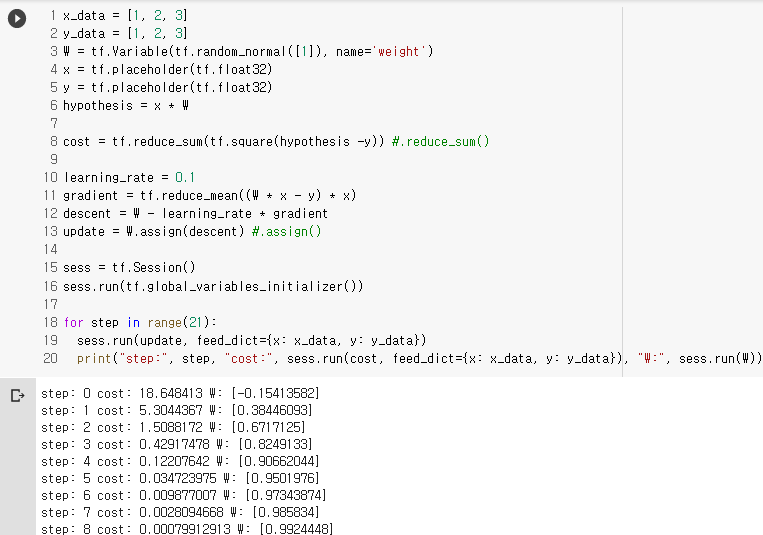
Cost W b



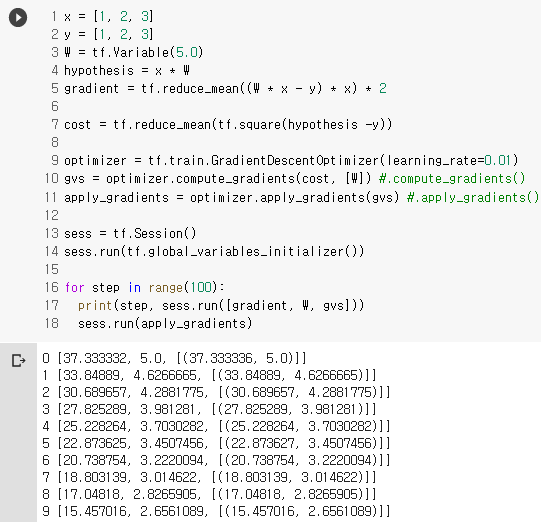
**ML\_Lab 03.**

**Test.**









**Memo**

.constant() //상수

.Session() //tensor에 데이터를 넣어 흐르게 함.

.run() //실행

.add() //더하기

.placeholeder(), feed\_dict={a:a\_data} //변수, 값을 나중에 할당.

.Variable() //변수, 자동으로 업데이트.

.random\_normal(Shapes) //랜덤 값 반환.

.reduce\_mean() //평균

.square() //제곱

.GradientDescentOptimizer() //미니 배치 확률적 경사하강법(SGD) 구현.

.minimize() //최소화

.global\_variables\_initializer() //.Variable()를 초기화.

.append() //append

.plot() //plot

.show() //show

.reduce\_sum() //총합

.assign() //.Variable()의 값 변경.

.compute\_gradients() //compute\_gradients

.apply\_gradients() //apply\_gradients

//Lab으로 연결 https://colab.research.google.com/drive/1gaTpEufmhoK2CsEsNyfDDtyynQ\_HRpSu

//14폰트, 12폰트, 10폰트

//1. 1) a. \*♠

//0.71 1.34