Lecture 5-1
Logistic (regression) classification

Sung Kim <hunkim+mr@gmail.com>

Acknowledgement

- Andrew Ng's ML class
  - https://class.coursera.org/ml-003/lecture
  - http://www.holehouse.org/mlclass/ (note)
- Convolutional Neural Networks for Visual Recognition
  - http://cs231n.github.io/
  - http://cs231n.stanford.edu/
- TensorFlow
  - https://www.tensorflow.org
  - https://github.com/aymericdamien/TensorFlow-Examples

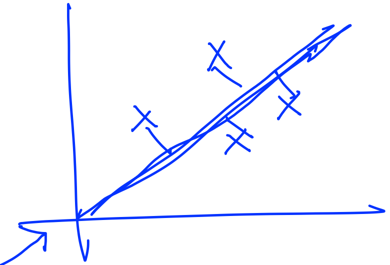
Regression (HCG)

- H
- C
- G

x1 (hours)	x2 (attendance)	y (score)
10	5	90
9	5	80
3	2	50
2	4	60
11	1	40

http://www.cse.iitk.ac.in/users/se367/10/presentation_local/Binary%20Classification.html

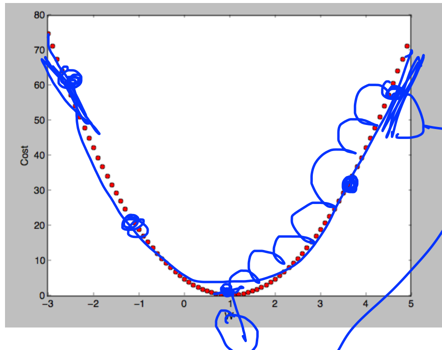
Regression



x1 (hours)	x2 (attendance)	y (score)
10	5	90
9	5	80
3	2	50
2	4	60
11	1	40

- Hypothesis: $H(X) = WX$

- Cost: $cost(W) = \frac{1}{m} \sum (WX - y)^2$



learning rate.

기울기

- Gradient decent:

$$W := W - \alpha \boxed{\frac{\partial}{\partial W} cost(W)}$$

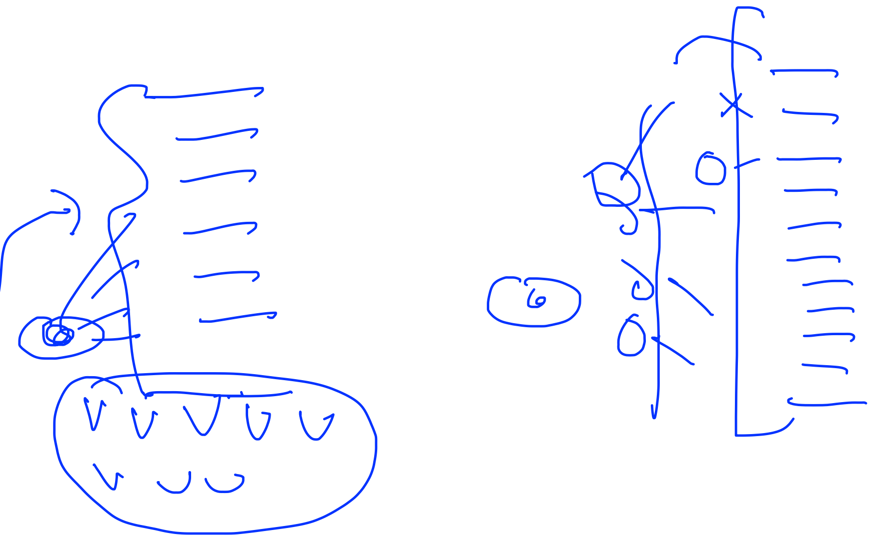
http://www.cse.iitk.ac.in/users/se367/10/presentation_local/Binary%20Classification.html

Binary

Classification

Email

- Spam Detection: Spam or Ham
- Facebook feed: show or hide
- Credit Card Fraudulent Transaction detection: legitimate/fraud

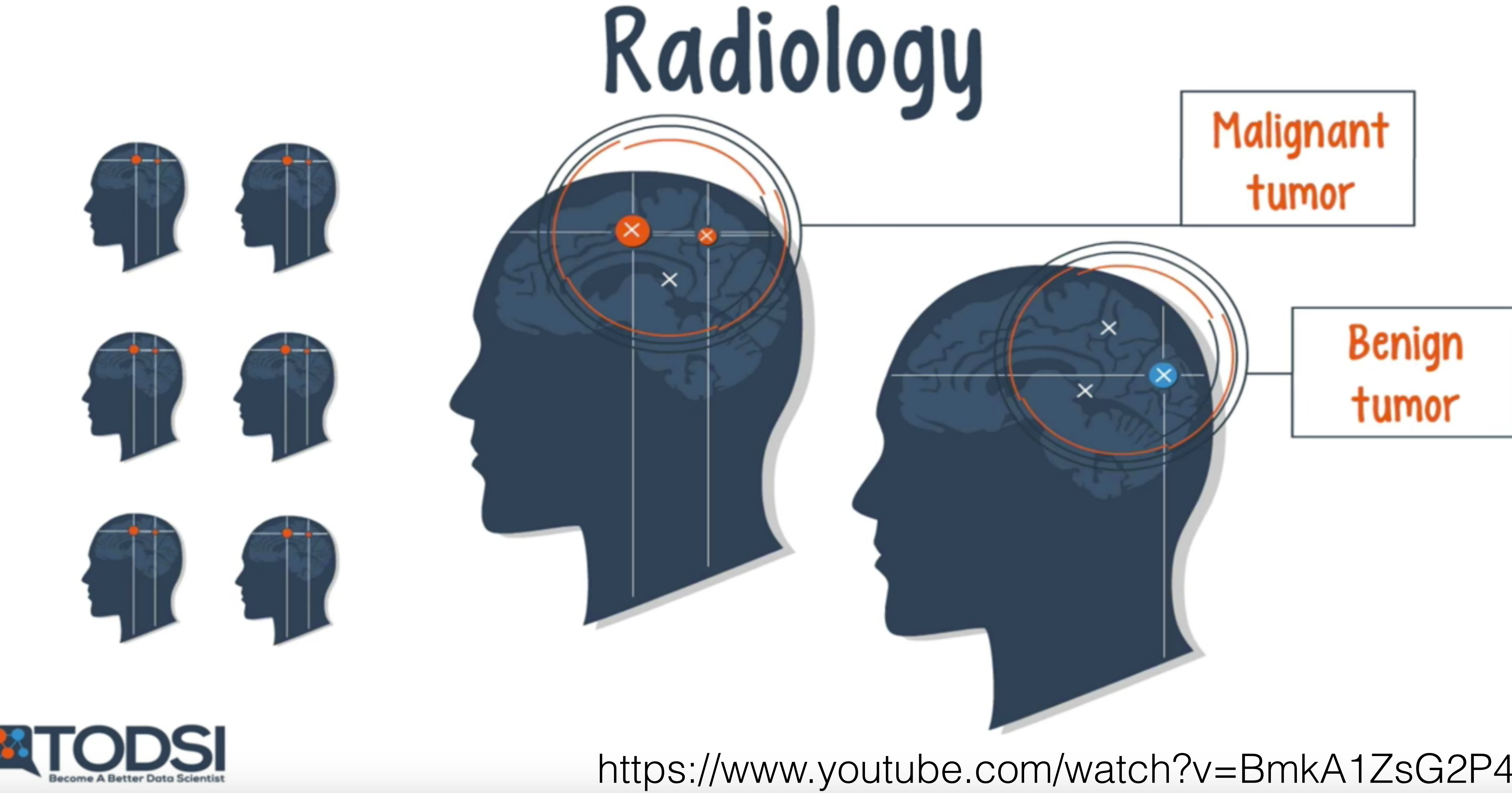


http://www.cse.iitk.ac.in/users/se367/10/presentation_local/Binary%20Classification.html

0, 1 encoding

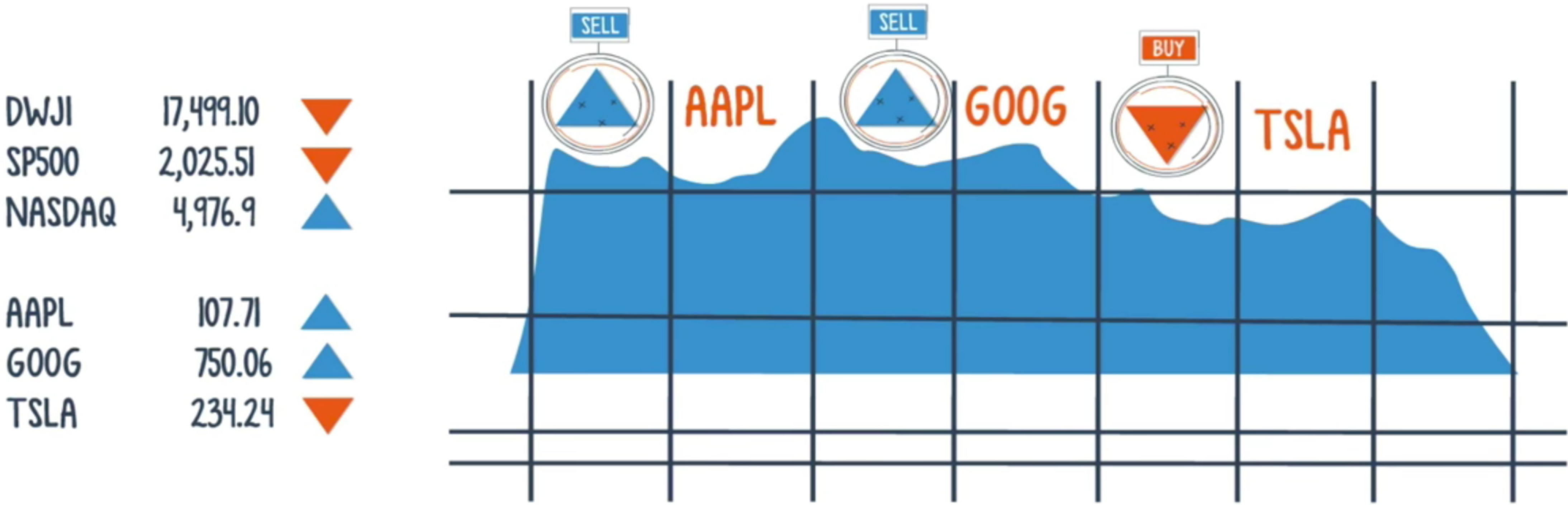
- Spam Detection: Spam (1) or Ham (0)
- Facebook feed: show(1) or hide(0)
- Credit Card Fraudulent Transaction detection: legitimate(0) or fraud (1)

http://www.cse.iitk.ac.in/users/se367/10/presentation_local/Binary%20Classification.html



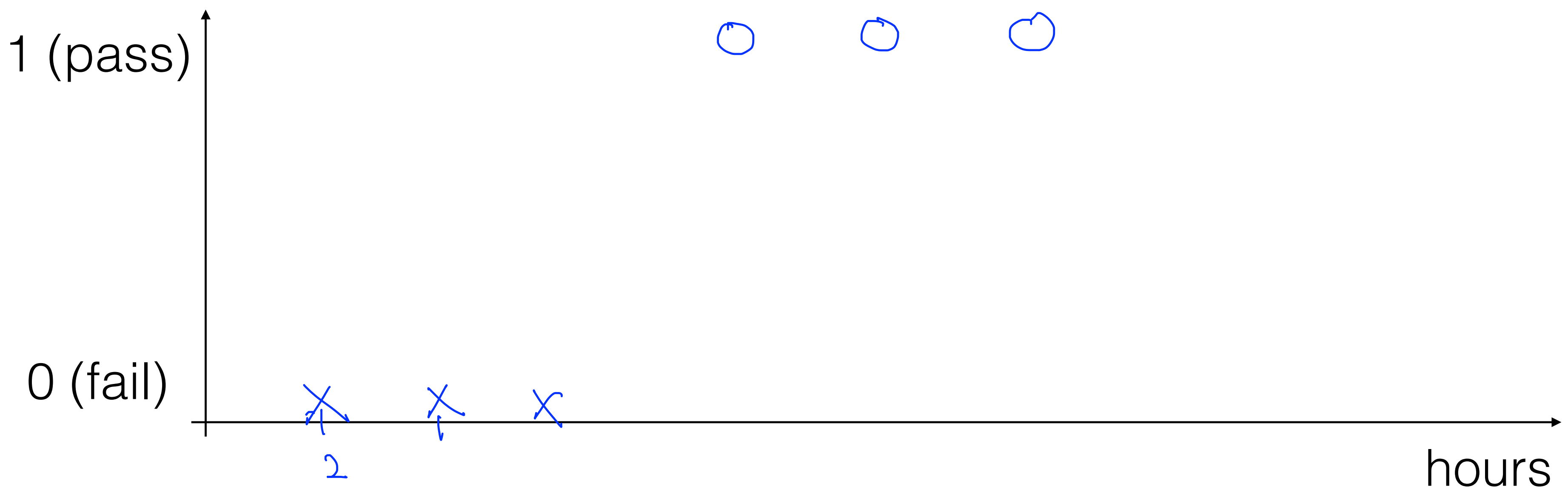
https://www.youtube.com/watch?v=BmkA1ZsG2P4

Finance

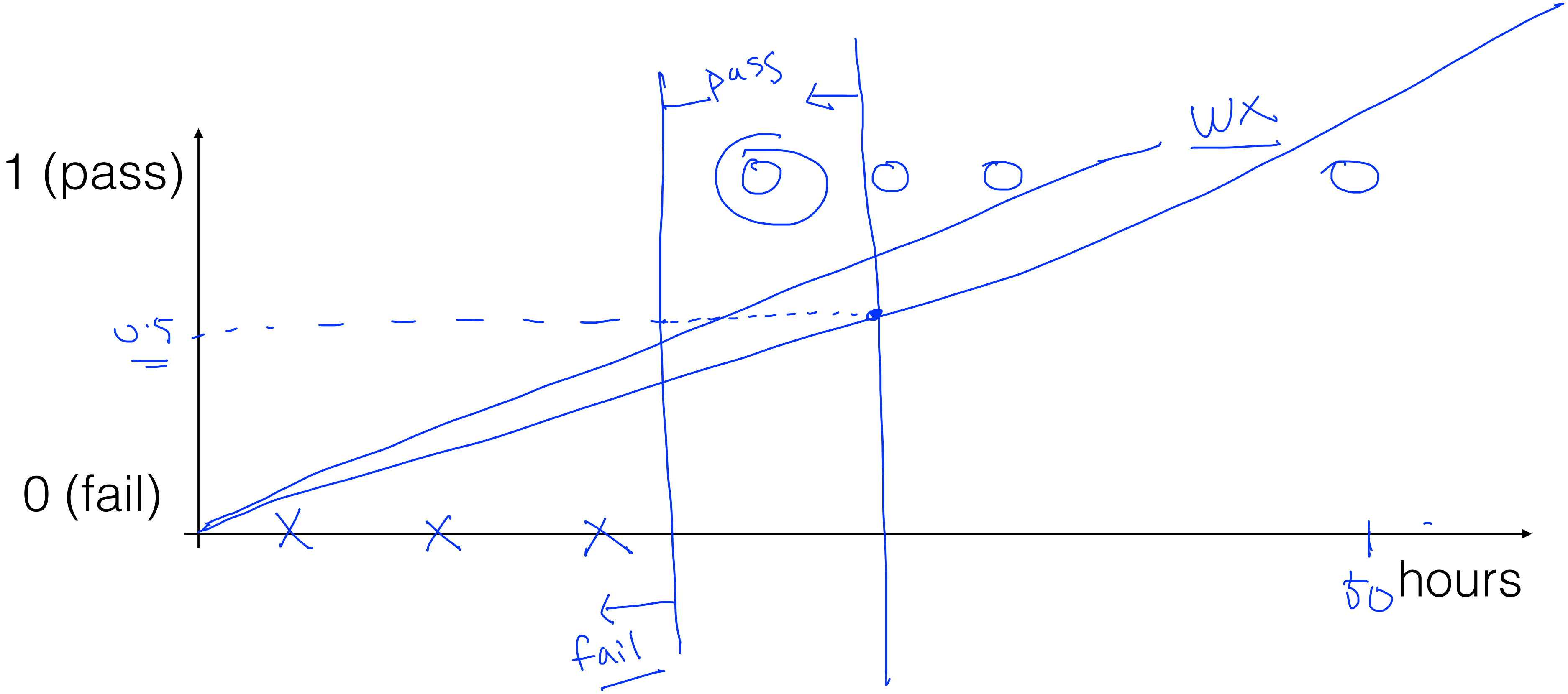


https://www.youtube.com/watch?v=BmkA1ZsG2P4

Pass(1)/Fail(0) based on study hours



Linear Regression?



Linear regression

- We know Y is 0 or 1

  $H(x) = Wx + b$

- Hypothesis can give values large than 1 or less than 0

$x \; [1, 2, 5, 10, 11]$

$W = 0.5, \; b = 0$

$0 < \sim < 1$

$x = 100 \rightarrow \boxed{\times 0.5} \rightarrow 50 \gg 1$

http://www.holehouse.org/mlclass/06_Logistic_Regression.html

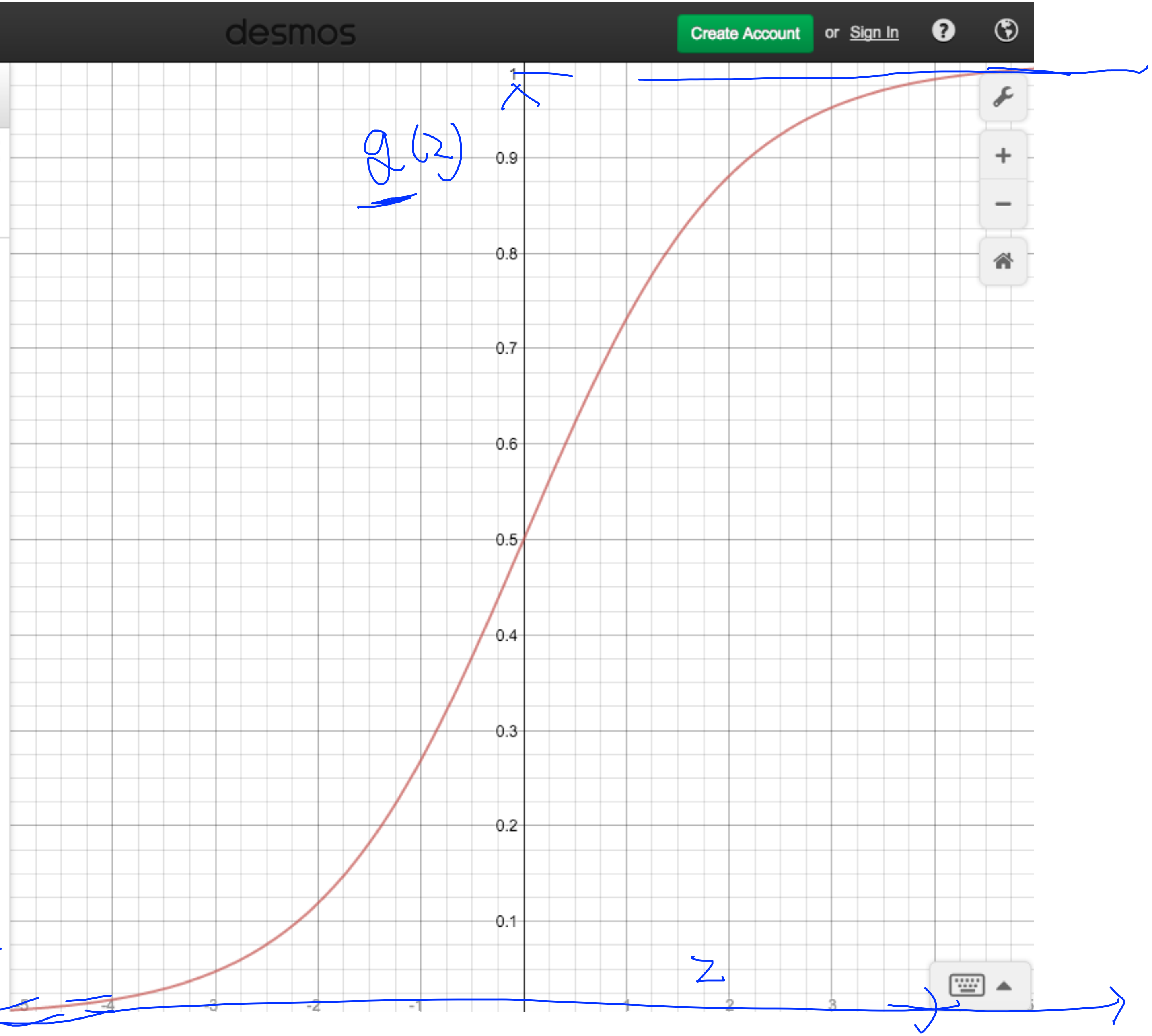
Logistic Hypothesis

$$z \quad \cancel{H(x)} = \underline{Wx + b}$$

$g(z) \rightarrow 0 \sim 1$

Untitled Graph

$$g(z) = \frac{1}{\left(1 + e^{-z}\right)}$$



$z = WX$

$H(x) = \underline{g(z)}$

logistic function,
sigmoid function.

sigmoid:
Curved in two directions,
like the letter "S",
or the Greek ς (sigma).

Logistic Hypothesis

$$H(X) = \frac{1}{1 + e^{-(W^T X)}}$$

$\underline{\underline{WX}}$

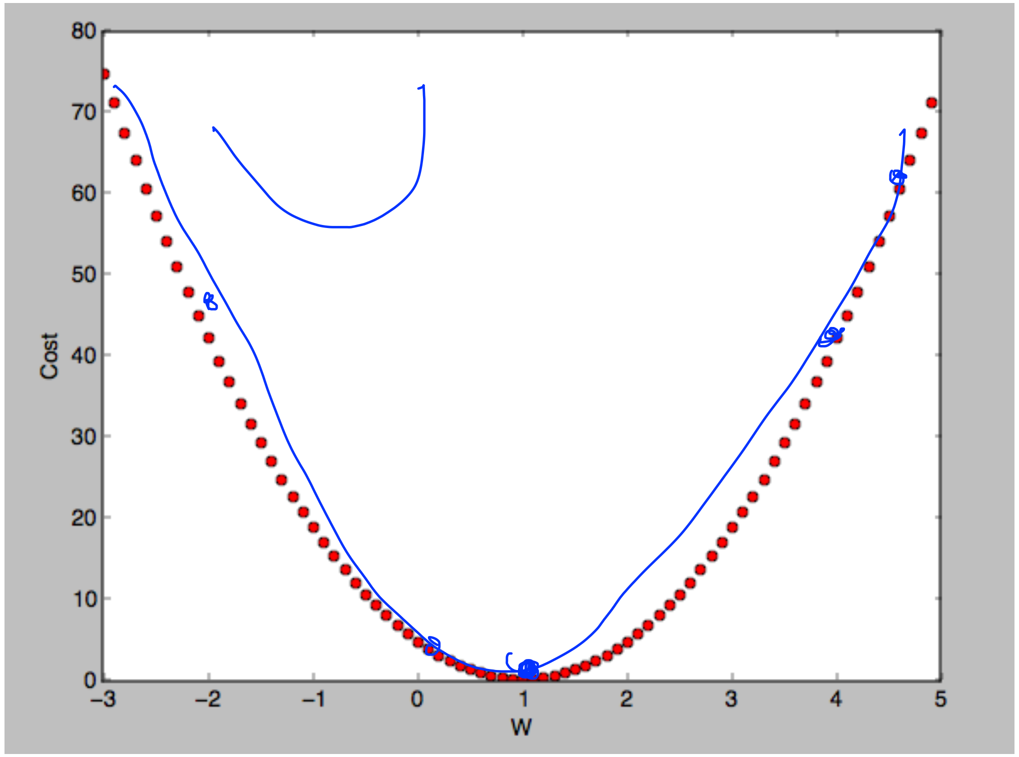
Lecture 5-2

Logistic (regression) classification: cost function & gradient decent

Sung Kim <hunkim+mr@gmail.com>

Cost

$$cost(W,b) = \frac{1}{m}\sum_{i=1}^{m}(H(x^{(i)}) - y^{(i)})^2 \quad \text{when} \quad \underline{H(x) = Wx + b}$$



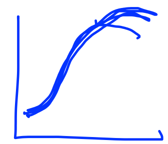
Cost function

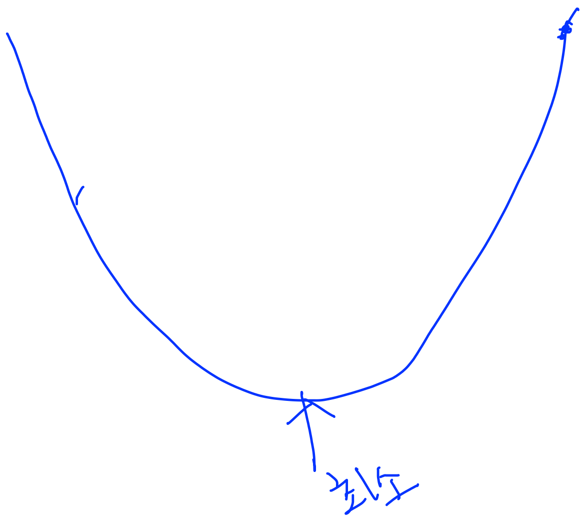
$$cost(W,b) = \frac{1}{m}\sum_{i=1}^{m}(H(x^{(i)}) - y^{(i)})^2$$

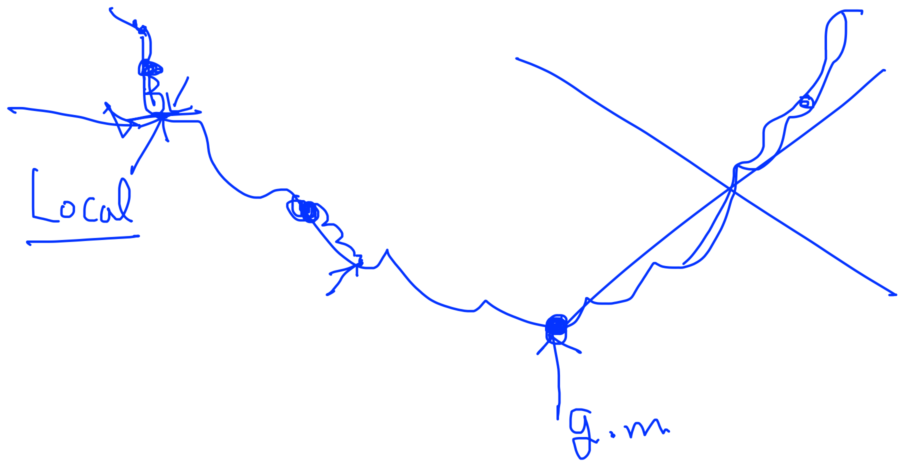
0 < ~ < 1

$$H(x) = Wx + b$$

$$H(X) = \frac{1}{1+e^{-W^T X}}$$







New cost function for logistic

$$cost(W) = \frac{1}{m} \sum \quad c(H(x), y)$$

$$c(H(x), y) = \begin{cases} -log(H(x)) & : y = 1 \\ -log(1 - H(x)) & : y = 0 \end{cases}$$

understanding cost function

$$C(H(x), y) = \begin{cases} -log(H(x)) & : y = 1 \\ -log(1 - H(x)) & : y = 0 \end{cases}$$

$\frac{1}{1+e^{-z}}$

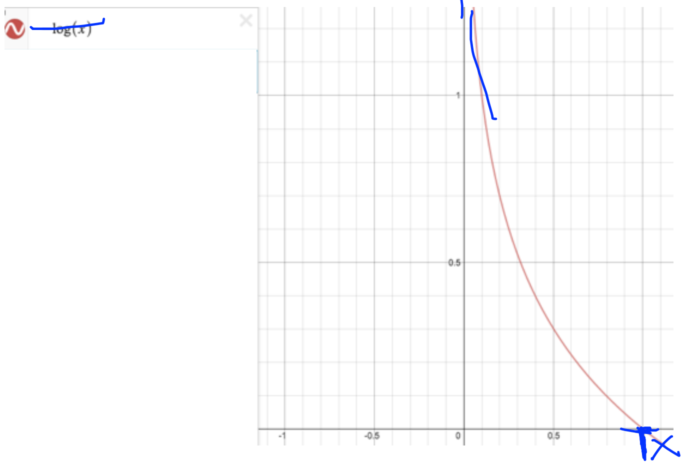
log

~~Cost~~ $\boxed{y=1}$

$H(x) = 1 \rightarrow cost(1) = 0$

$H(x) = 0 \rightarrow cost = \infty \uparrow$

cost = $g(z) = -\log(z)$

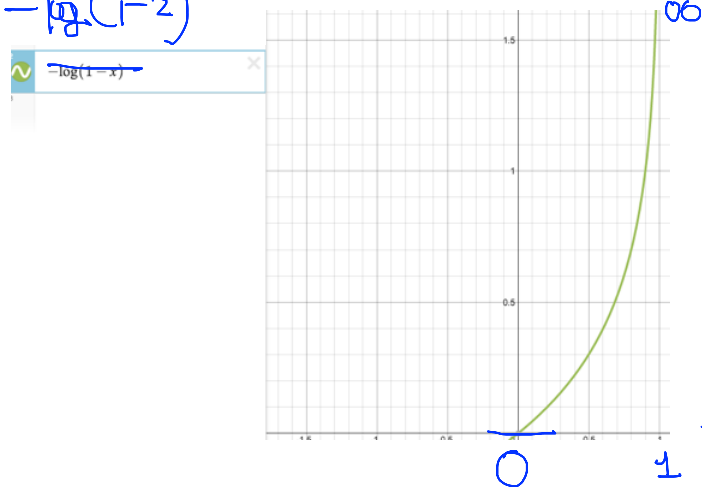


$y=0$

$H(x) = 0$, $cost = 0$

$H(x) = 1$, $cost = \infty \uparrow$

$-\log(1-z)$



Cost function

$$cost(W) = \frac{1}{m} \sum c(H(x), y)$$

$$c(H(x), y) = \begin{cases} -log(H(x)) & : y = 1 \\ \underline{-log(1 - H(x))} & : y = 0 \end{cases}$$

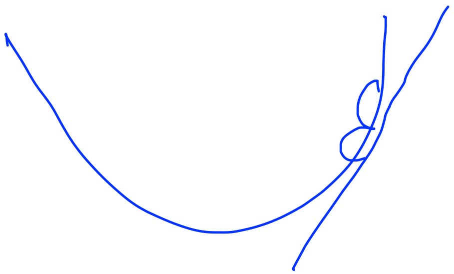
$$\boxed{c(H(x), y) = -ylog(H(x)) - (1 - y)log(1 - H(x))}$$

$y=1$, $c = -log(H(x))$

$y=0$, $c = -1 * log(1-H(x))$

Minimize cost - Gradient decent algorithm

$$cost(W) = -\frac{1}{m}\sum y\log(H(x)) + (1-y)\log(1-H(x))$$



$$W := W - \alpha \boxed{\frac{\partial}{\partial W} cost(W)}$$

Gradient decent algorithm

$$cost(W) = -\frac{1}{m}\sum ylog(H(x)) + (1-y)log(1-H(x))$$

$$W := W - \alpha\frac{\partial}{\partial W}cost(W)$$

```
# cost function
cost = tf.reduce_mean(-tf.reduce_sum(Y*tf.log(hypothesis) + (1-Y)*tf.log(1-hypothesis)))

# Minimize
a = tf.Variable(0.1) # Learning rate, alpha
optimizer = tf.train.GradientDescentOptimizer(a)
train = optimizer.minimize(cost)
```

Next

Multinomial classification (Softmax)