# Deep Learning

# Session 2

# Logistic Regression

# Outline

1. Recap
2. Generative vs Discriminative Learning
3. Logistic Regression
4. Softmax and Cross-Entropy

# …in the previous chapter

Field of study that gives computers the ability to learn without being explicitly programmed.
Arthur Samuel (1959).

Instead of writing task-specific programs by hand, we build algorithms able to learn from existing cases (i.e. e-mail spam classifier algorithm).

A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.
Tom Mitchell (1998).

# …in the previous chapter

Basic elements/ingredients on Machine Learning.

- Inputs & Outputs

$$x,\ y$$

- Mapping function (model)

$$h_w(x)\quad [x \to y]$$

- Cost function

$$J(w)$$

- Learning process

# …in the previous chapter

Iterative Gradient Descent is one way to minimize the cost function to find our best parameters W

Another one is to take the derivative of the cost function with respect to W and set to zero.

The result of that is the Normal Equation:

$$W = (X^T X)^{-1} X^T y$$

The problem is that if we have a big number of samples and inputs this is far from an easy computation

# …in the previous chapter

Basic elements/ingredients on house pricing Linear Regression problem?

- Inputs & Outputs

  Input: x (the size); output y: the price

- Mapping function

  $$h_w(x) = y = wx + b$$

- Cost function

  $$J(w) = 1/2m \ \Sigma \ (h_w(x)^{(i)} - y^{(i)})^2$$

- Learning process

  Iterative Gradient Descent

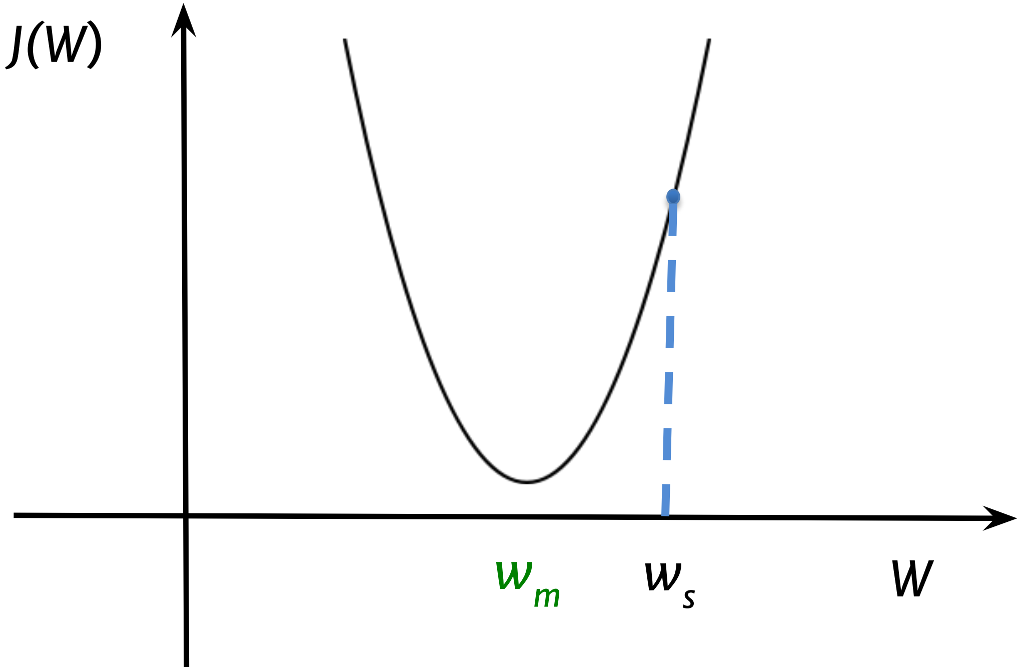
# …in the previous chapter

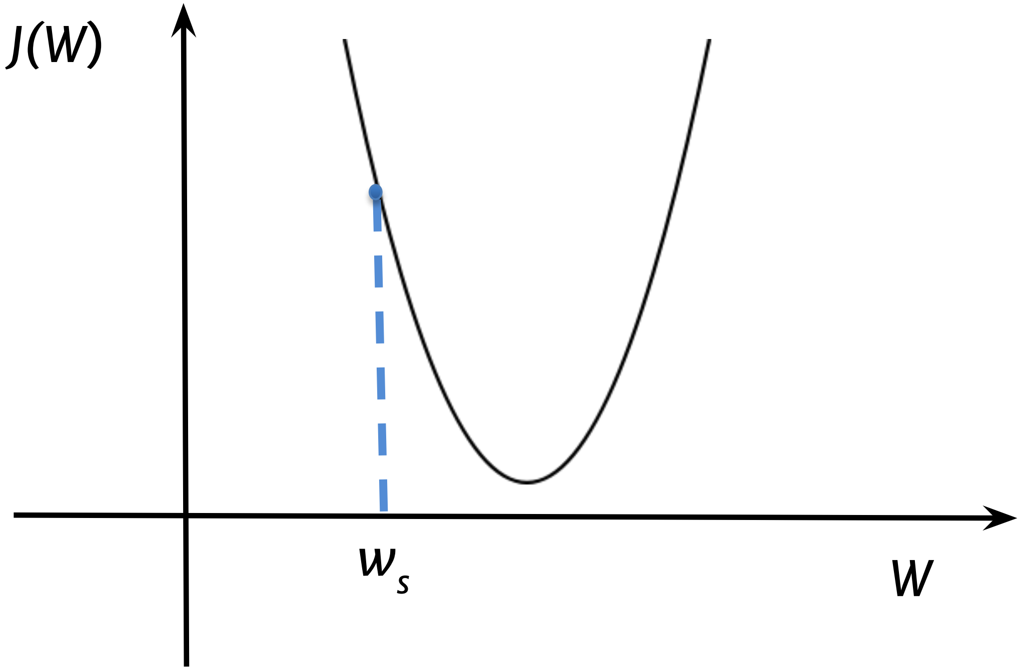
Gradient descent algorithm

```
Repeat until convergence {
```

$$\hat{w}_i = w_i - \alpha \frac{\partial J(w)}{w}$$

```
}
```



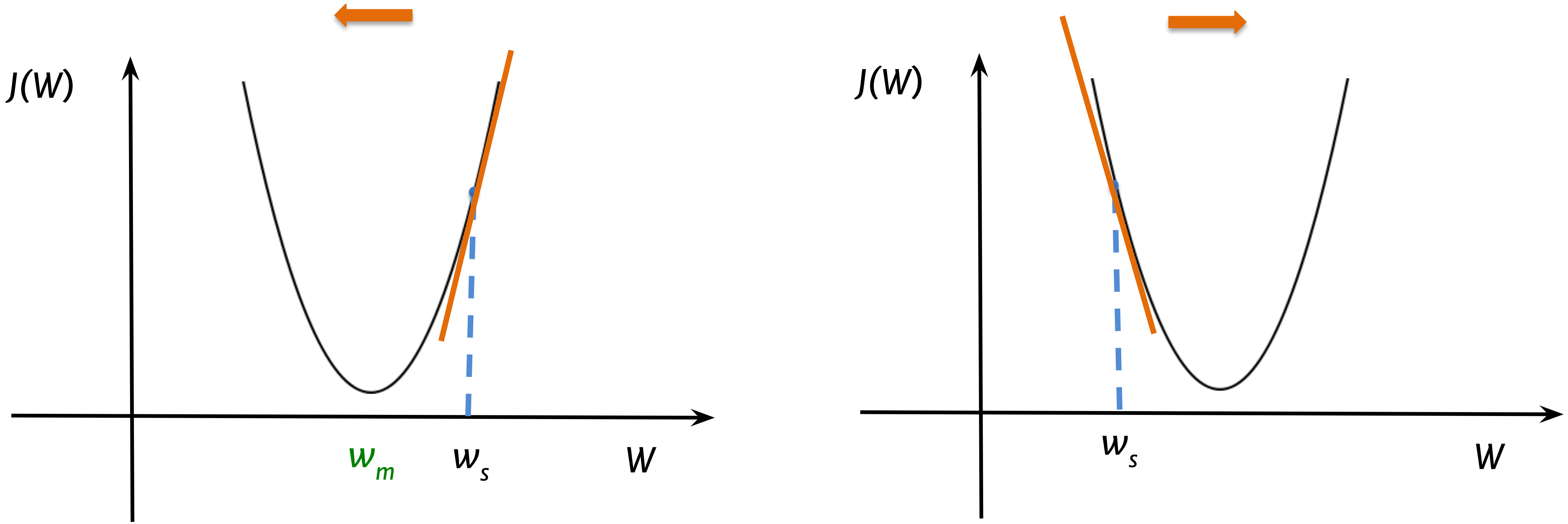
# …in the previous chapter

Gradient descent algorithm

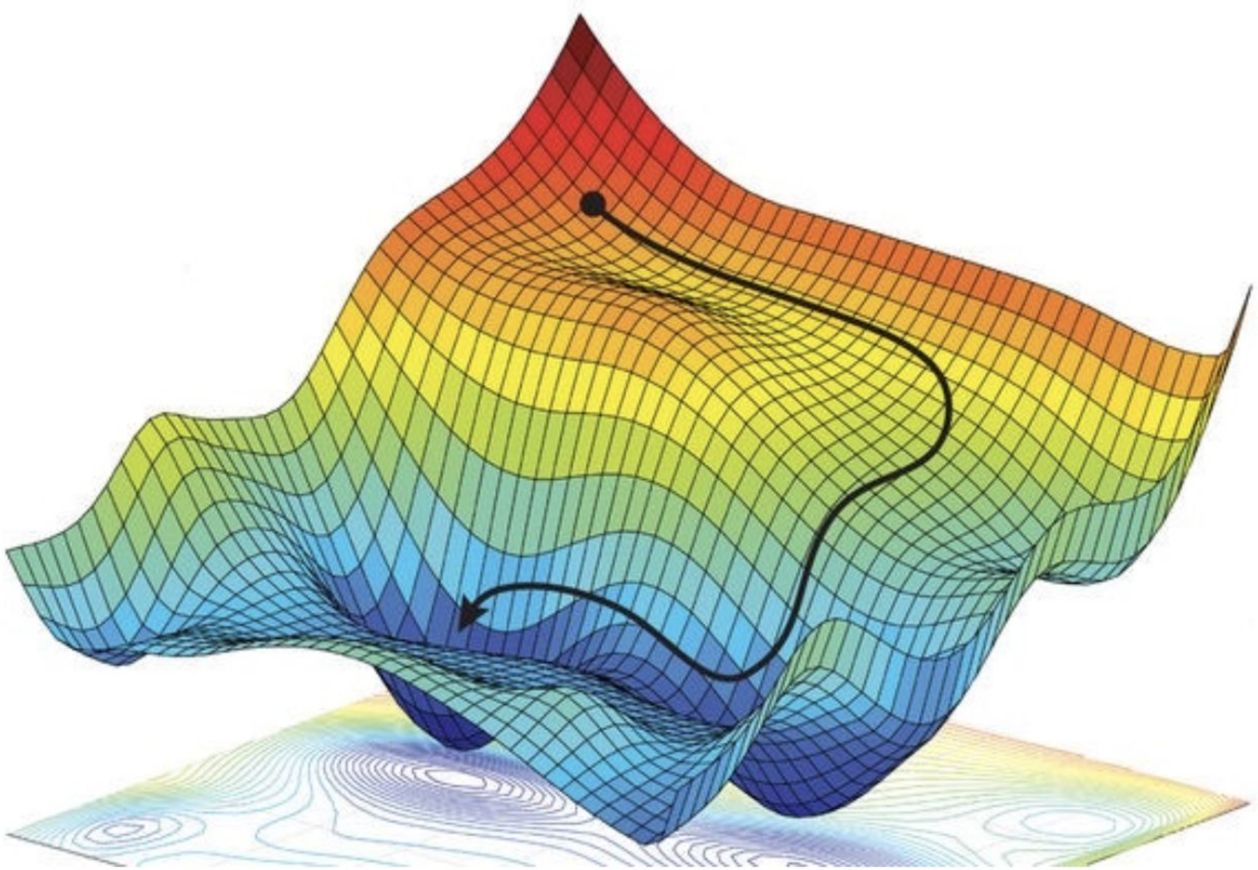
```
Repeat until convergence {
```

$$\hat{w}_i = w_i - \alpha \frac{\partial J(w)}{w}$$

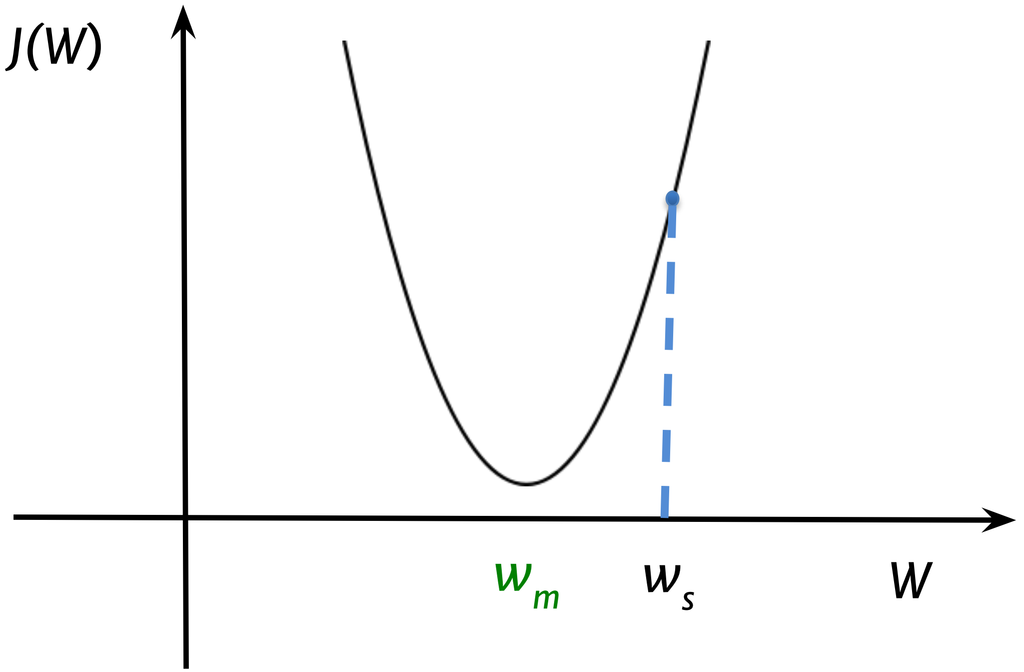
```
}
```



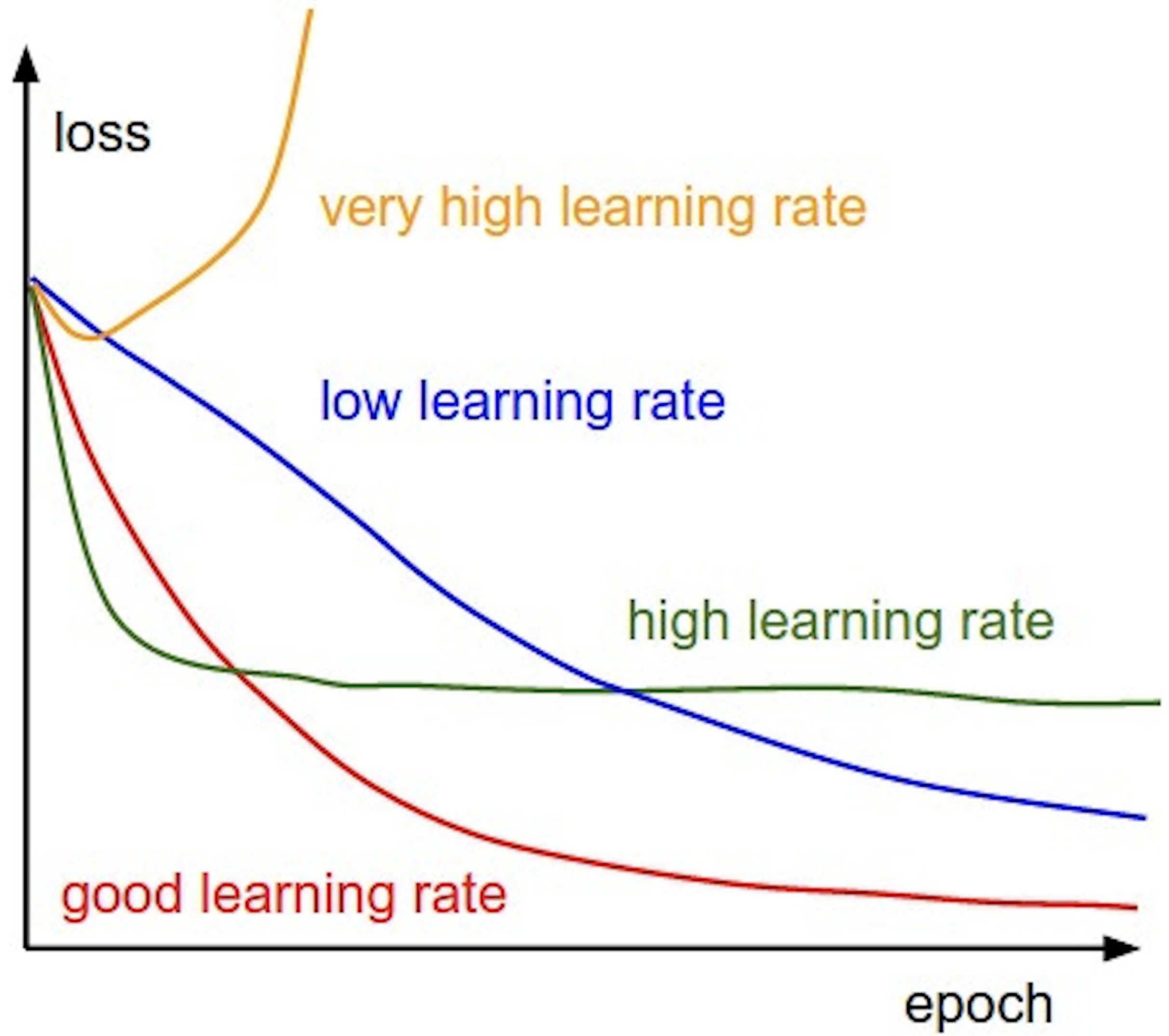
# …in the previous chapter



# The role of α: the learning rate



# The role of α: the learning rate



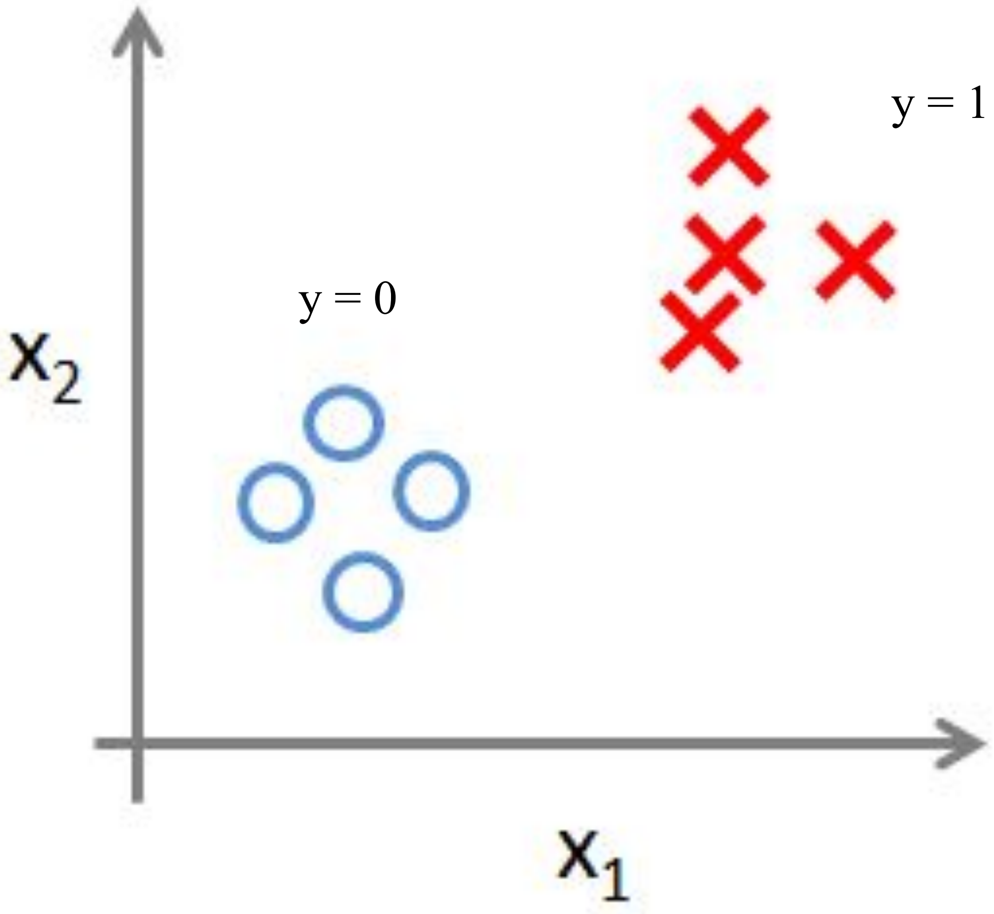
# Main Goals of this class

Understand the difference between generative and discriminative learning

Understand logistic regression.

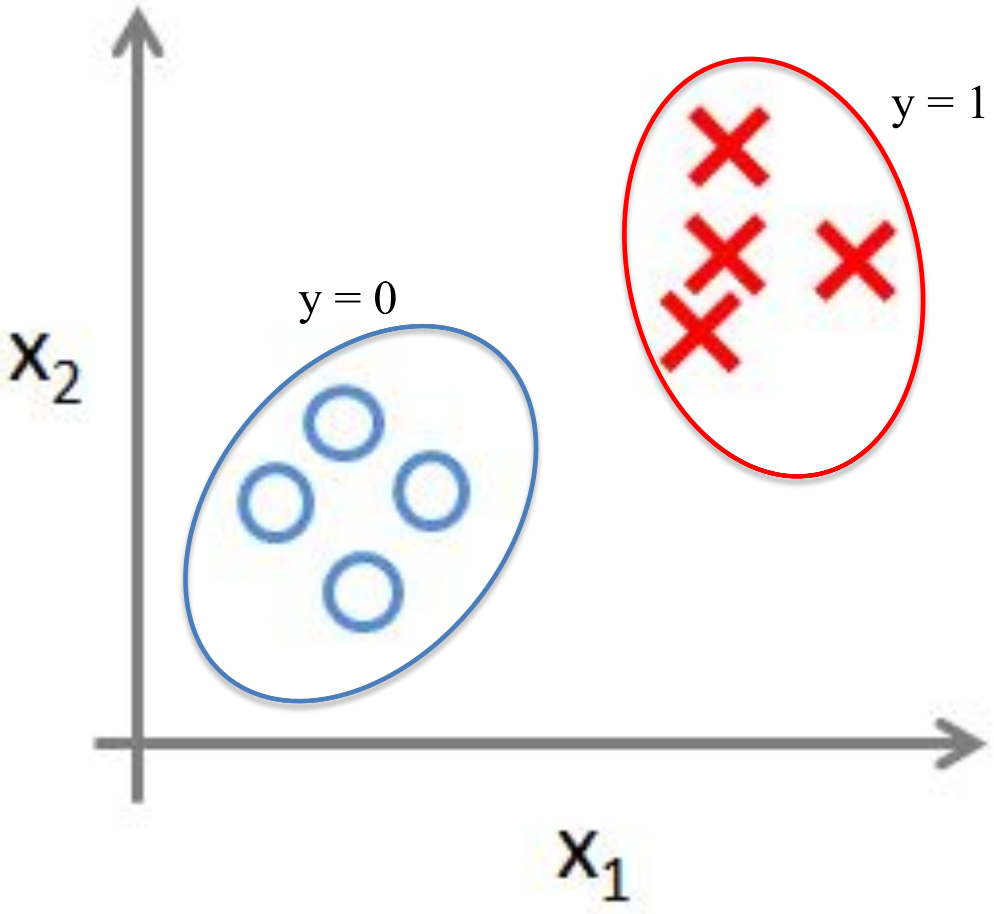
Recognize LogReg as a baby neural network

# Generative vs. Discriminative Learning



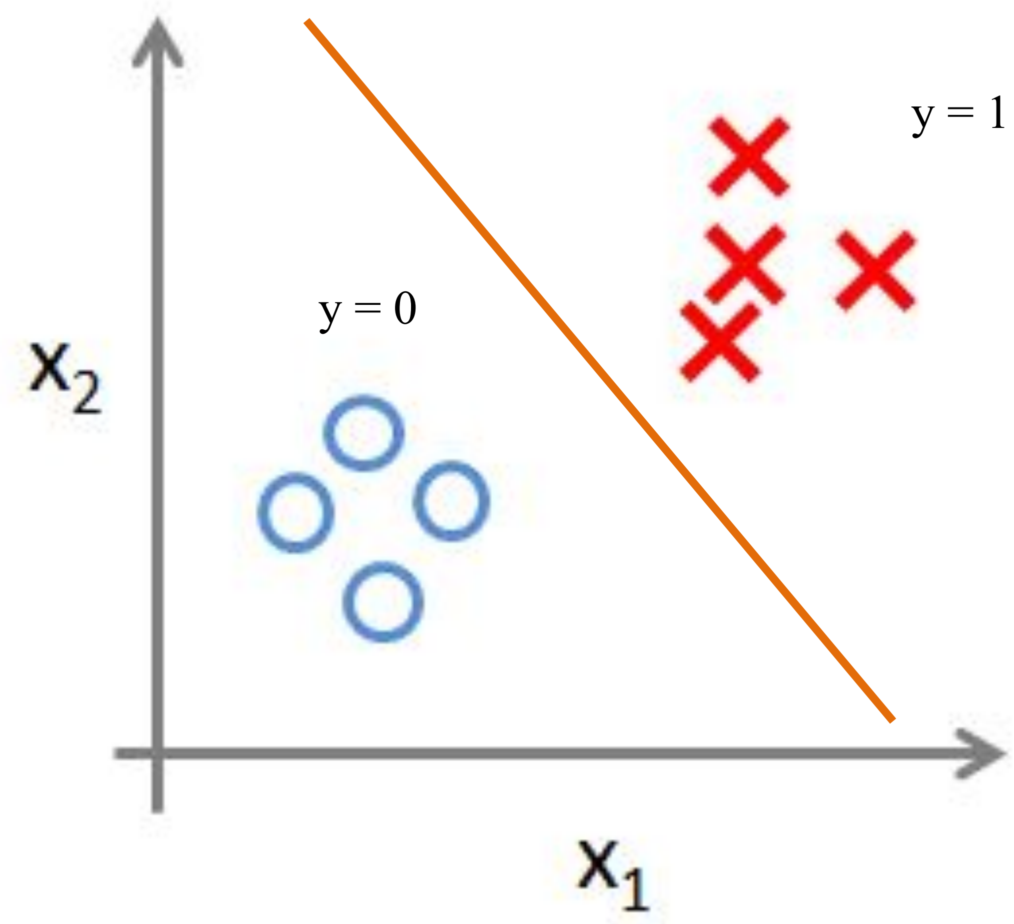
# Generative vs. Discriminative Learning

Generative Approach



# Generative vs. Discriminative Learning

Discriminative Approach



# Generative vs. Discriminative Learning

In a Generative approach, from a probabilistic perspective, we are estimating the joint probability

$$p(x, y)$$

to then compute $P(y \mid x)$ [what we really need for classification!!], through the Bayes Rule.

In a discriminative approach we directly attempt to compute what we really need for classification, the conditional probability

$$p(y \mid x)$$

# Generative vs. Discriminative Learning

Quick Remind! Bayes Rule

$$\underbrace{P(y \mid x)}_{\text{Posterior}} = \frac{P(x,y)}{P(x)} = \frac{P(x \mid y)P(y)}{P(x)} \propto \underbrace{P(x \mid y)}_{\text{likelihood}}\underbrace{P(y)}_{\text{Prior}}$$

# Generative vs. Discriminative Learning

Vladimir Vapnik https://en.wikipedia.org/wiki/Vladimir_Vapnik

**Which approach is better?**

- Vapnik said. “one should solve the classification problem directly and never solve a more general problem as an intermediate step such as modeling p(x | y)”
- If we use a generative approach, we are doing something more ambitious. With p(x, y) we might generate pair samples (x, y)!
- Use both approaches and combine them! It is not forbidden!

# Logistic Regression

… before we really start with:

- Watch out with the name! Logistic Regression is a **CLASSIFICATION** technique

# Logistic Regression

… before we really start with:

- Watch out with the name! Logistic Regression is a **CLASSIFICATION** technique
- **Supervised** learning. Thus, we have a training labeled dataset.

# Logistic Regression

… before we really start with:

- Watch out with the name! Logistic Regression is a **CLASSIFICATION** technique
- **Supervised** learning. Thus, we have a training labeled dataset.
- Follow a **discriminative** approach. Thus, it attempts to model the conditional probability p(y | x)

# Logistic Regression

… before we really start with:

- Watch out with the name! Logistic Regression is a **CLASSIFICATION** technique
- **Supervised** learning. Thus, we have a training labeled dataset.
- Follow a discriminative approach. Thus, it attempts to model the conditional probability $p(y \mid x)$
- As a machine learning technique, we have a: mapping function, a cost function and a learning algorithm.

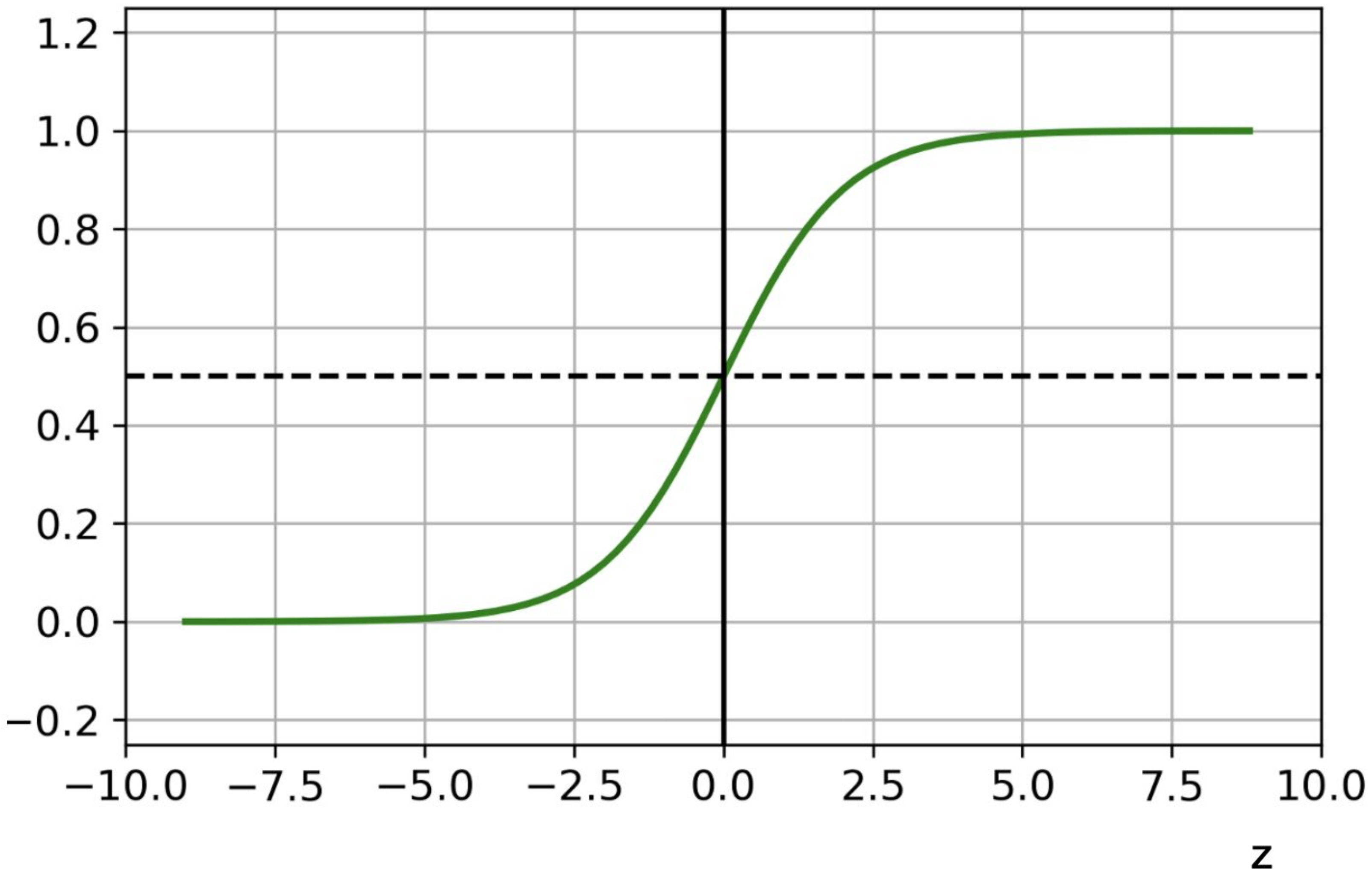
# Logistic Regression

- Inputs & outputs
  - Email containing (counting on different words)  spam or not spam (binary)
  - Speech signals  language spoken (multiclass)
  - Faces pictures  is it a pretty kitty or not? (binary)
- Hypothesis or mapping function

$$h_w(x) = \sigma(w^t x + b)$$

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

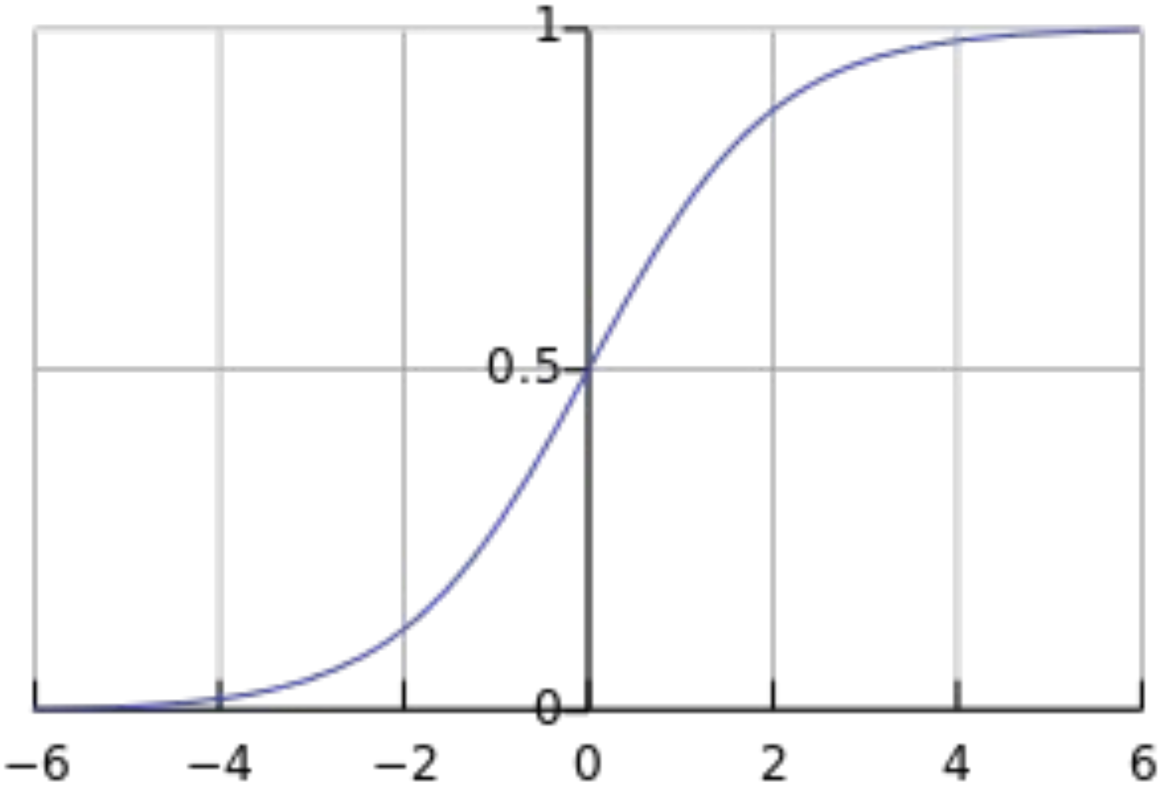
# Logistic Regression




$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

# Logistic Regression

$$\sigma(z) = \frac{1}{1+e^{-z}}$$



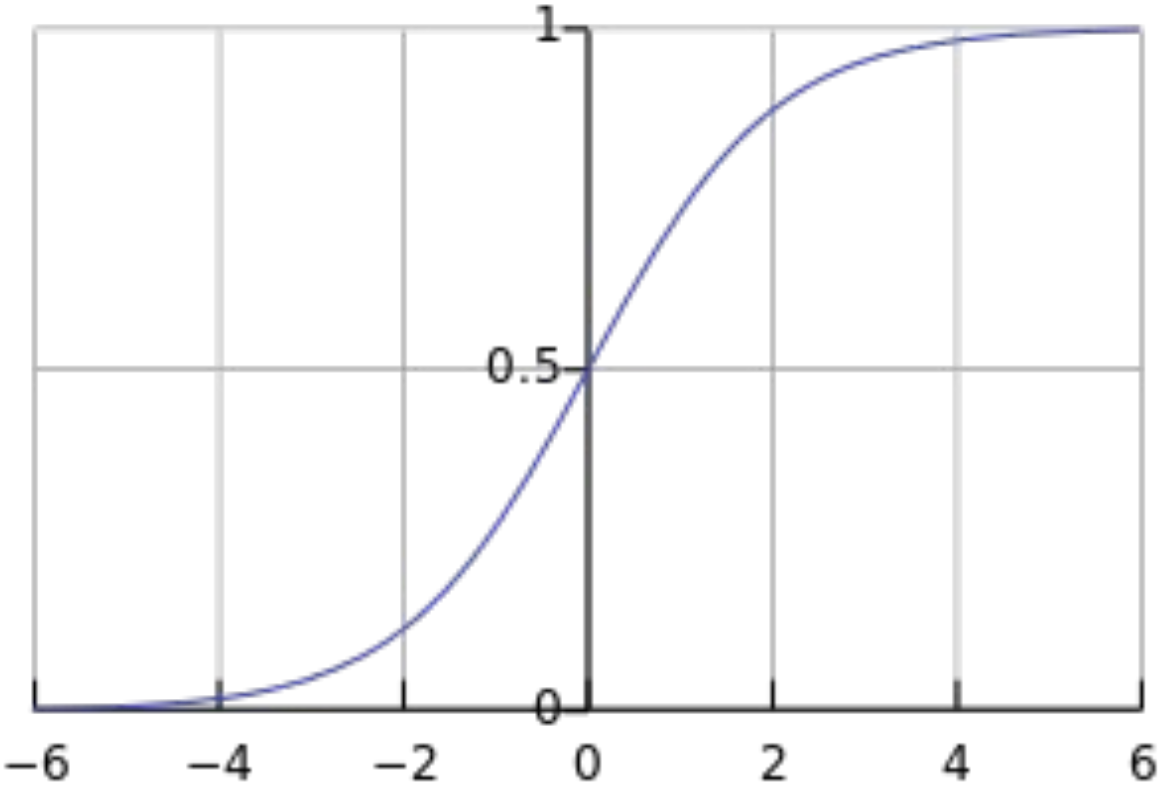

- This meaning …

$$0 \leq h_w(x) = \sigma(w^t x + b) \leq 1$$

- Output ranges between 0 and 1, what can be interpreted as a probability

# Logistic Regression

$$\sigma(z) = \frac{1}{1+e^{-z}}$$




- This meaning …

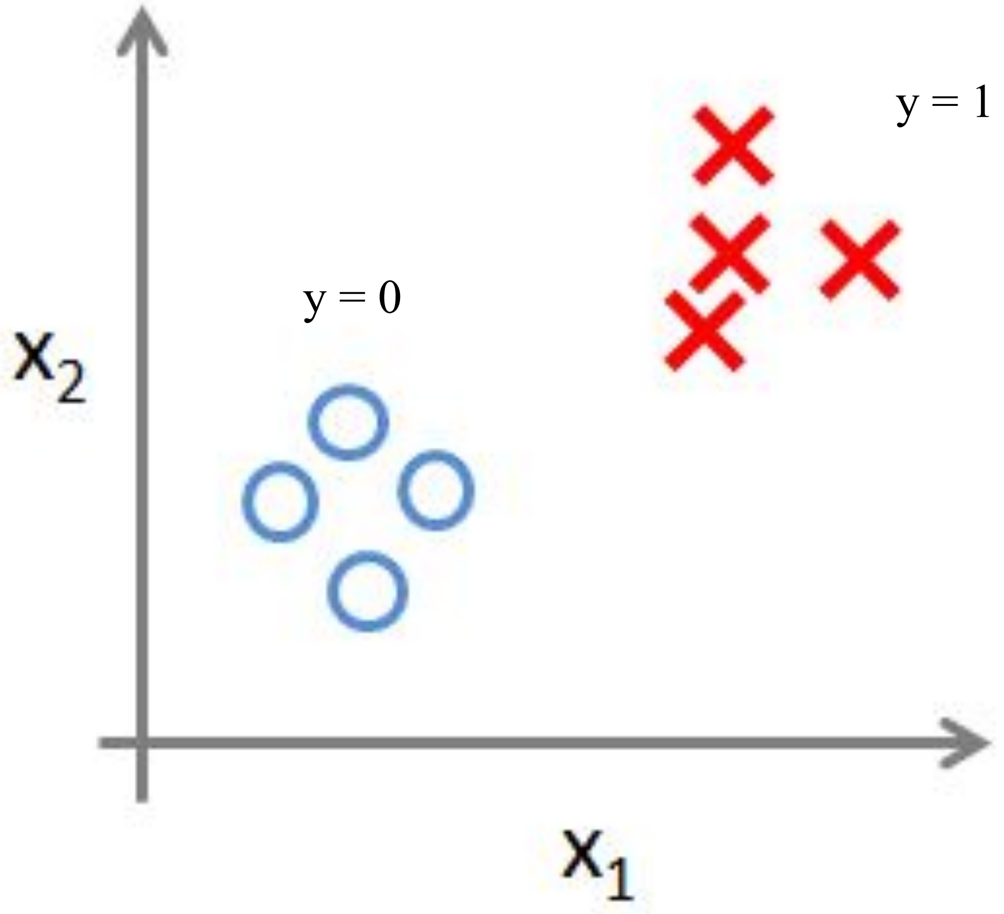
$$0 \leq h_w(x) = \sigma(w^t x + b) \leq 1$$

Using for binary classification

Decide y = 1 if $h_w(x) \geq 0.5$

Decide y = 0 if $h_w(x) < 0.5$

# Logistic Regression

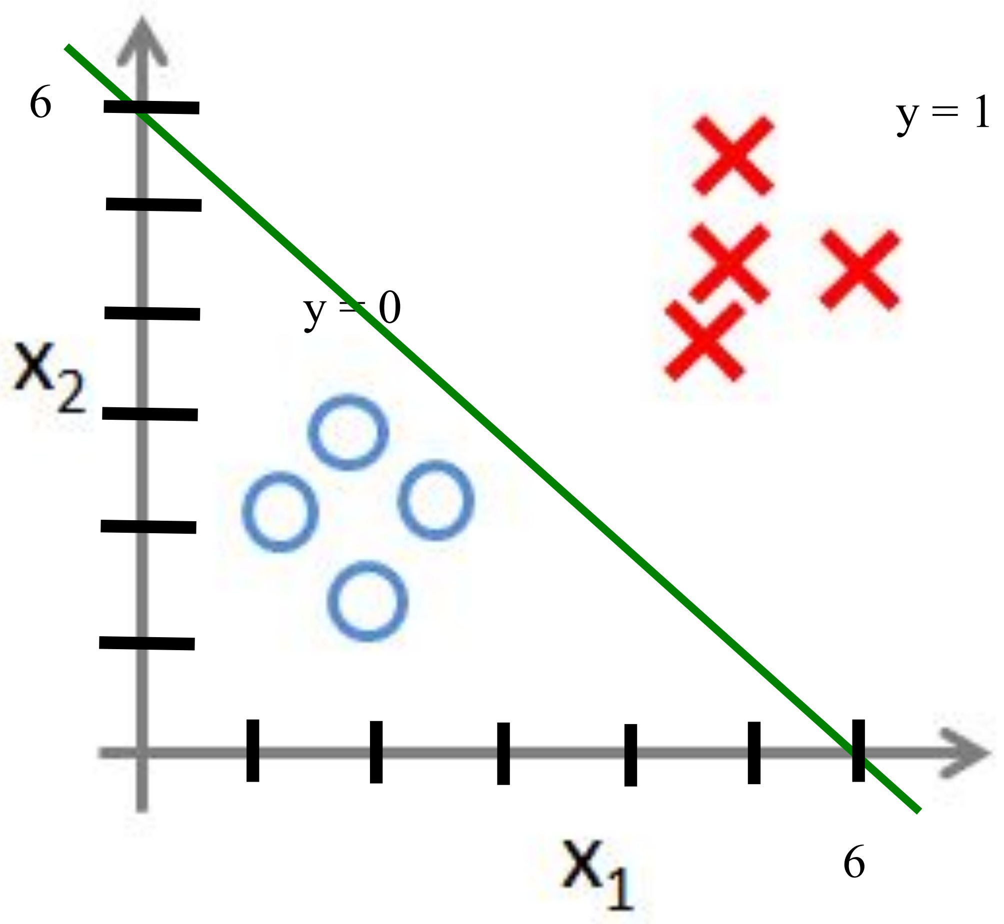



$$0 \leq h_w(x) = \sigma(w^t x + b) \leq 1$$

$y = 1 \text{ if } (w^T x + b) \geq 0$

$y = 0 \text{ if } (w^T x + b) < 0$

# Logistic Regression




$$0 \le h_w(x) = \sigma(w^t x + b) \le 1$$

$$y = 1 \text{ if } (w^T x + b) \ge 0$$

$$y = 0 \text{ if } (w^T x + b) < 0$$

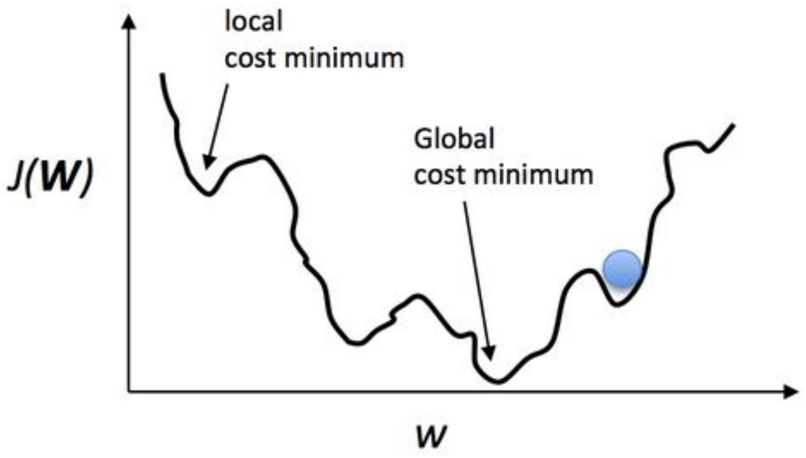
$$-6 + x_1 + x_2 \ge 0 => y = 1$$
$$-6 + x_1 + x_2 < 0 => y = 0$$

# Logistic Regression

- Cost function. Trying with linear regression one!

$$J(W) = \frac{1}{2N}\sum_{i=1}^{N}(h_w(x_i) - y_i) = \frac{1}{2N}\sum_{i=1}^{N}\sigma(w^t x_i + b) - y_i)^2$$

- Uhmm, that does not seem convex … no the best thing for gradient descent.



# Logistic Regression

- Better (log loss function)

$$J(W) = \frac{1}{N}\sum_{i=1}^{N} c(h_w(x_i), y_i)$$

$$c(h_w(x_i), y_i) \begin{cases} -\log h_w(x)\, if\, y = 1 \\ -\log(1 - h_w(x))\, if\, y = 0 \end{cases}$$

# Logistic Regression

- Exercise: Does the log loss function behave as a cost function?

$$J(W) = \frac{1}{N}\sum_{i=1}^{N} c(h_w(x_i), y_i)$$

$$c(h_w(x_i), y_i) \begin{cases} -\log h_w(x) \, if \; y = 1 \\ -\log(1 - h_w(x)) \, if \; y = 0 \end{cases}$$

# Logistic Regression

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)}))]$$

**Gradient descent algorithm**

repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

}

---

Correct: Simultaneous update

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$
$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$
$$\theta_0 := \text{temp0}$$
$$\theta_1 := \text{temp1}$$

- Now that we have a cost function… We can train the model!
- Of course, good old gradient descent algorithm again…

# Logistic Regression

- Can we extend this to work with more classes?

# Logistic Regression

- Can we extend this to work with more classes?
  - Multiclass Logistic regression, what do we need?

# Logistic Regression

- Can we extend this to work with more classes?
  - Multiclass Logistic regression, what do we need?
    - » Inputs and outputs
    - » Mapping function
    - » Cost function
    - » Training algorithm

# Logistic Regression

- Multiclass Logistic regression mapping function: **Softmax function**

$$P(y=j\mid x)=\frac{e^{w_j^t x+b}}{\sum_{i=k}^{K} e^{w_j^t x+b}}$$

# Logistic Regression

- Multiclass Logistic regression mapping function: **Softmax function**

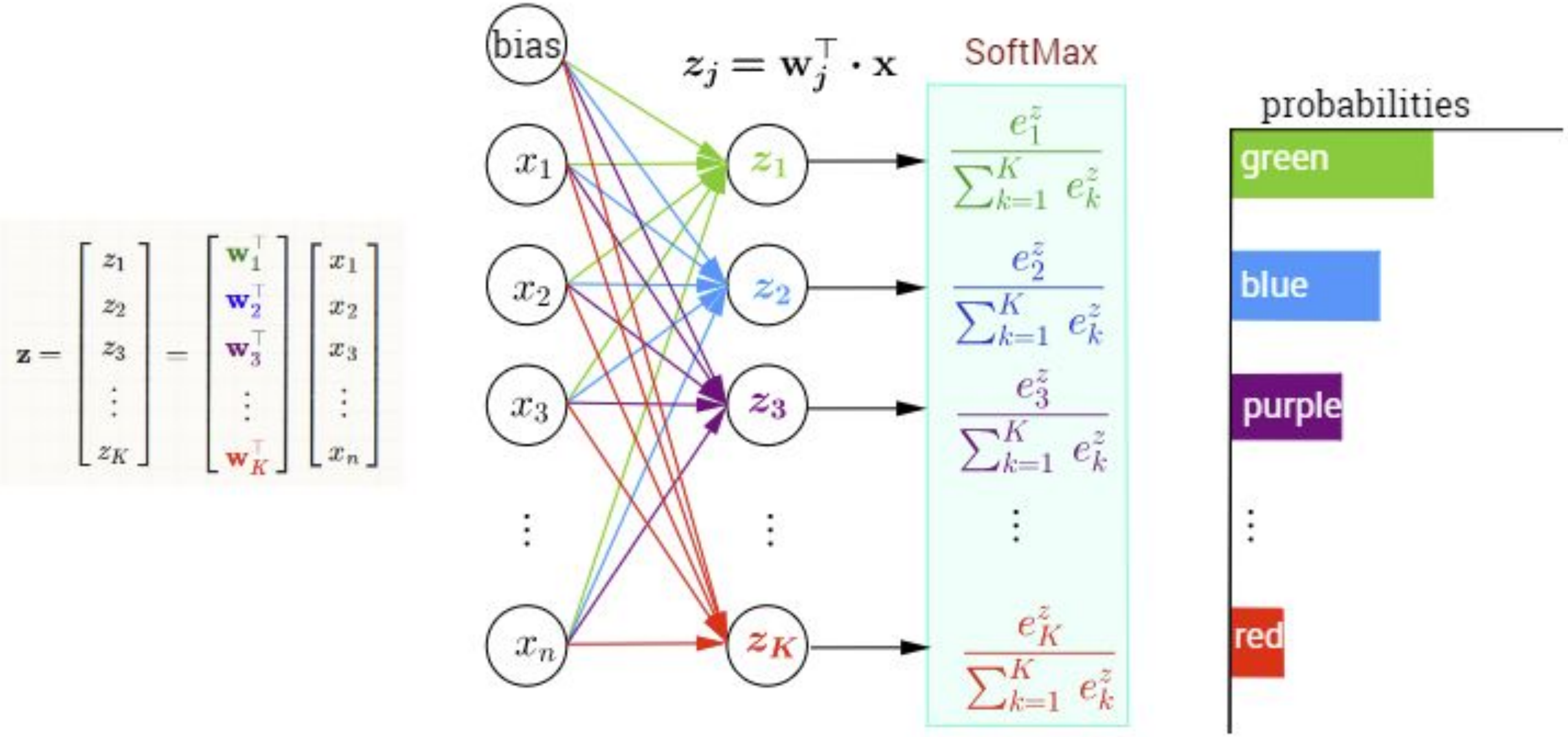
$$P(y=j\mid x)=\frac{e^{w_j^t x+b}}{\sum_{i=k}^{K} e^{w_j^t x+b}}$$

The outputs of the new mapping function can be directly interpreted as probabilities for classes as:

$$\sum_{i=1}^{C} P(y=j\mid x)=1$$

# Logistic Regression

- Multiclass Logistic regression mapping function: **Softmax function**

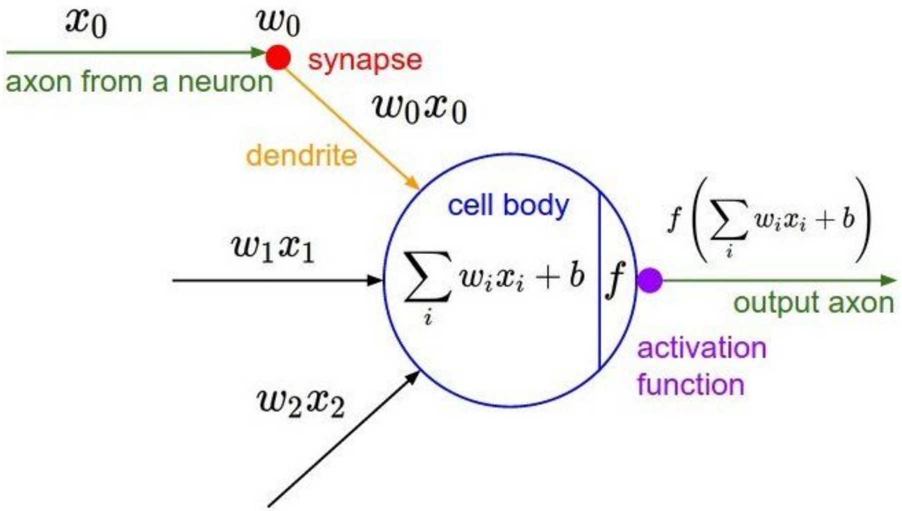


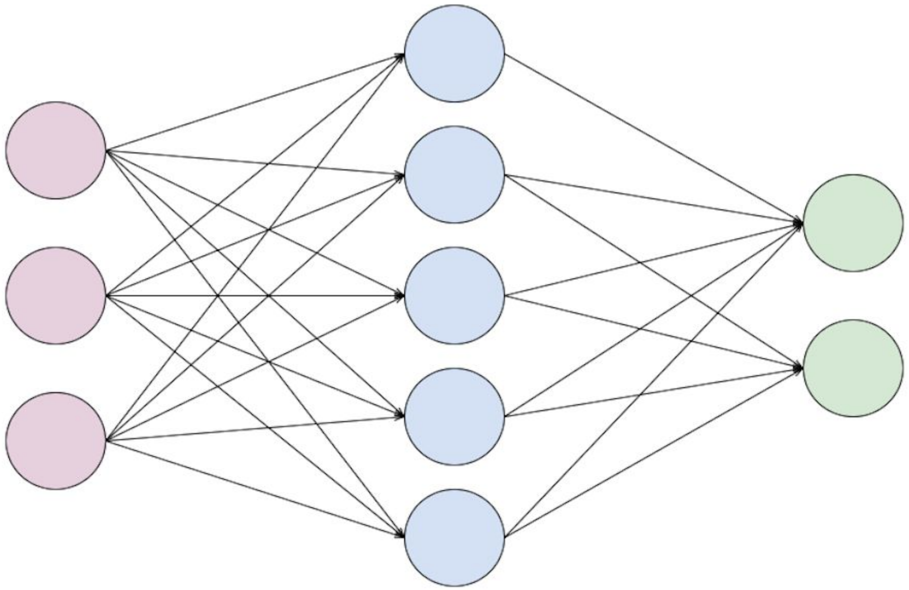
# Logistic Regression

- Multiclass Logistic regression cost function: **Cross-Entropy loss func.**

$$-\sum_{i=1}^{N} y \log(\hat{y}) = -\sum_{i=1}^{N} y \log(h_w(x_i)) = -\sum_{i=1}^{N} y \log(\sigma(w^t x_i + b))$$

# Logistic Regression as a baby Neural Net



Binary Logistic regression as a ‘degenerated’ neural net without hidden layers and one output node

# Summary

- Logistic Regression is a discriminative, supervised classification technique. It uses the sigmoid over the common linear regression hypothesis to output a [0,1] probability.
- The softmax function besides cross entropy cost function allows us to perform multiclass logistic regression in an elegant manner.
- Logistic Regression is the base of DNNs. It can be seen as a degenerated DNN with one neuron in the output layer and no hidden layers