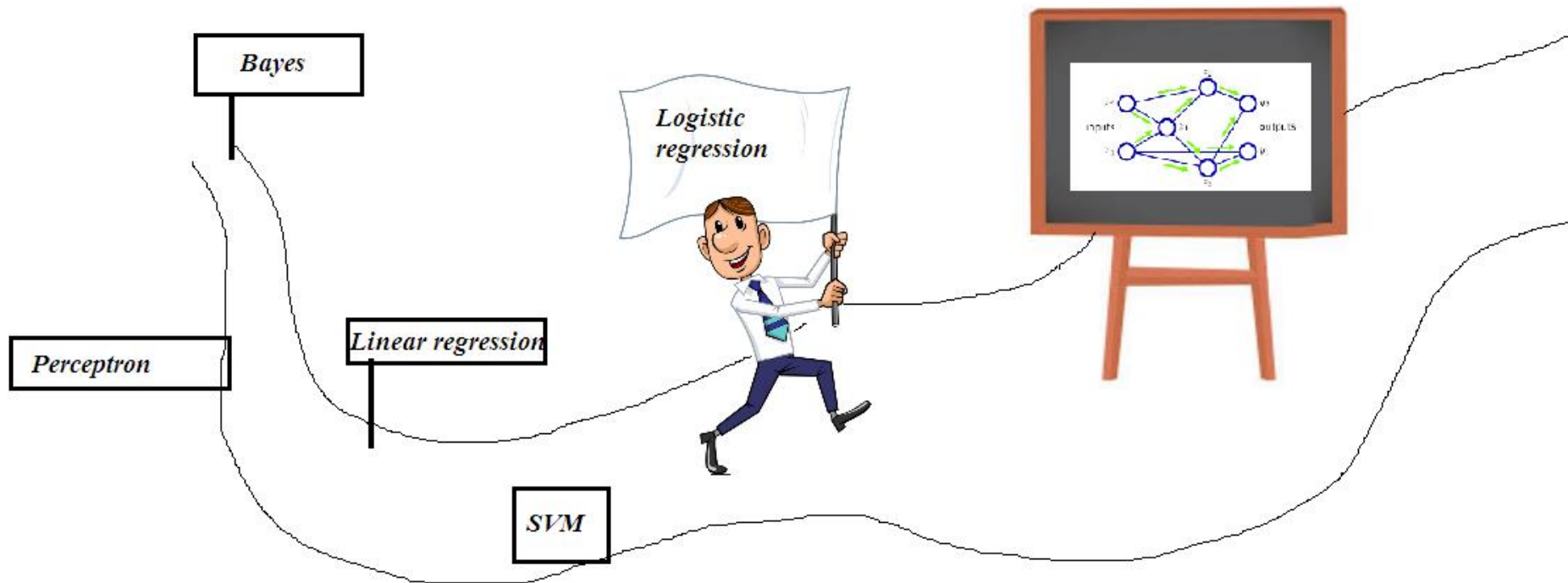


# Logistic Regression

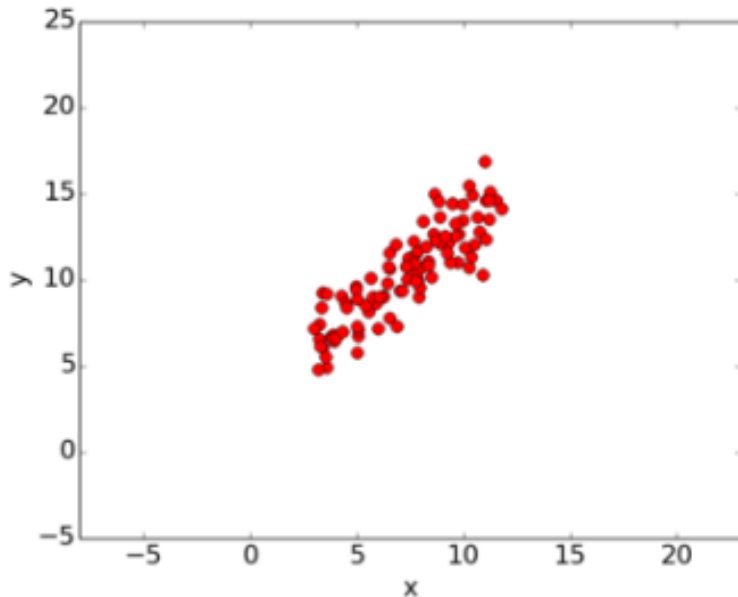
Sigmoid Activation in Perceptron



# Overview

- Linear Regression – Review
- Linear Regression – as classifier – Problem with outliers
- Logistic Regression – Sigmoid
  - Classifier ?
  - Criterion used in logistic regression
  - Solution

# Linear regression



$$y = mx + b$$

We want to fit a straight line (in 2D case).  
The sample we are given with is  $\{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$ .

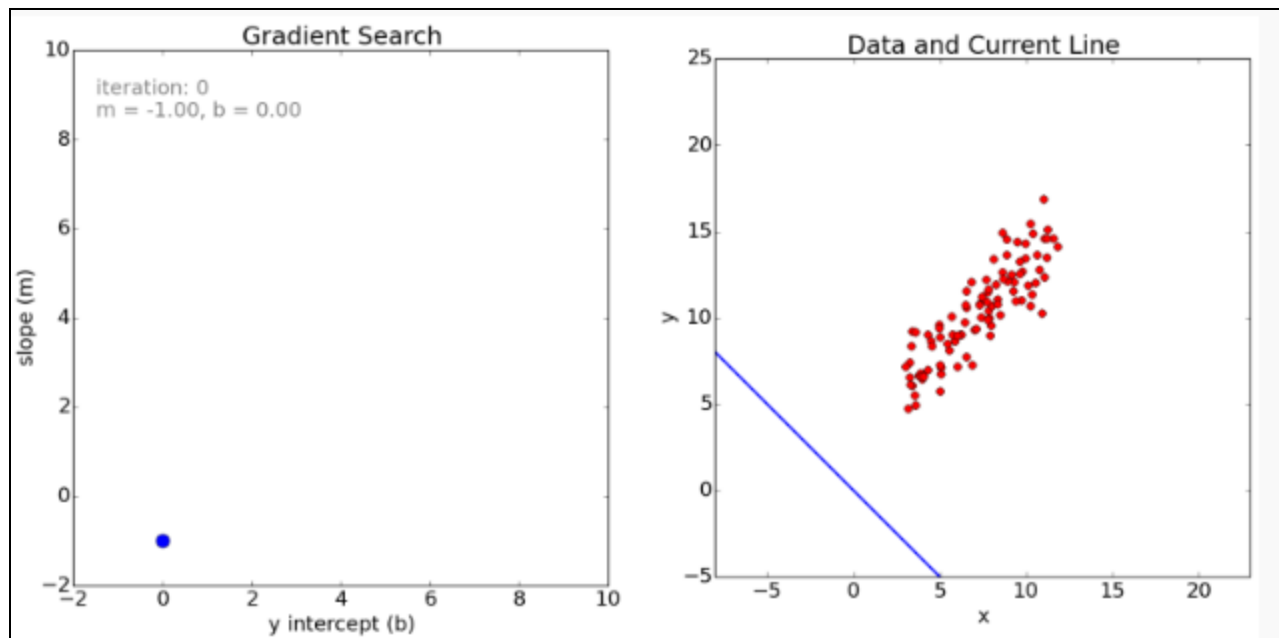
We want to find  $(m, b)$ .

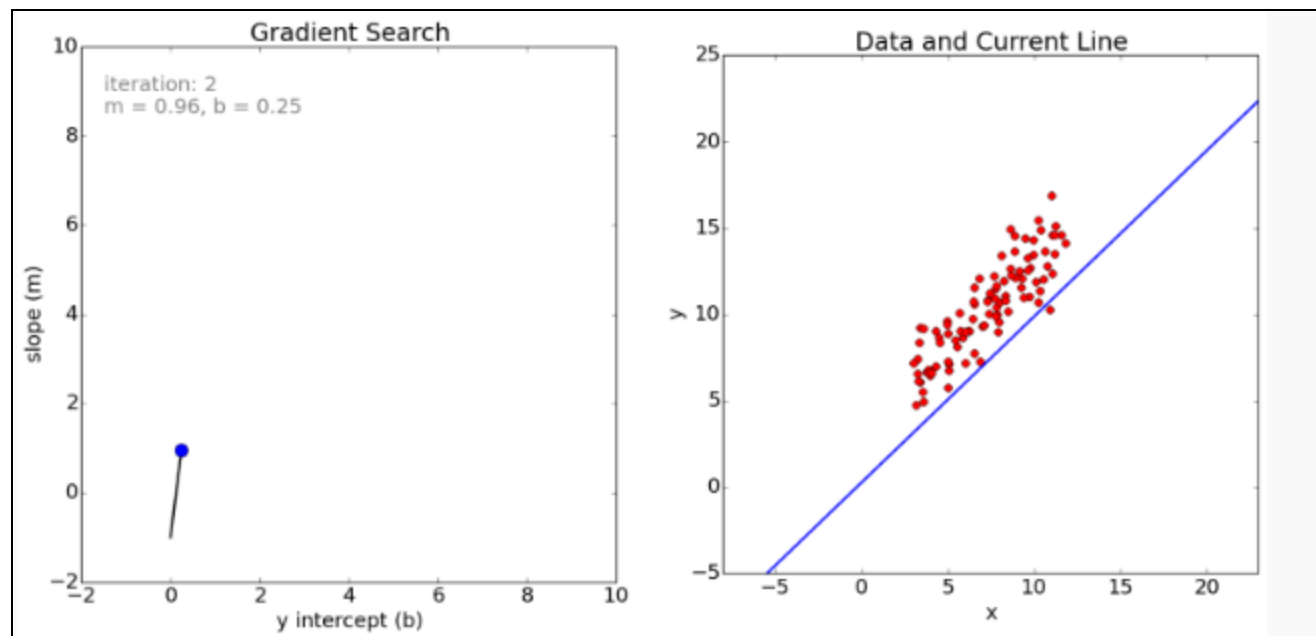
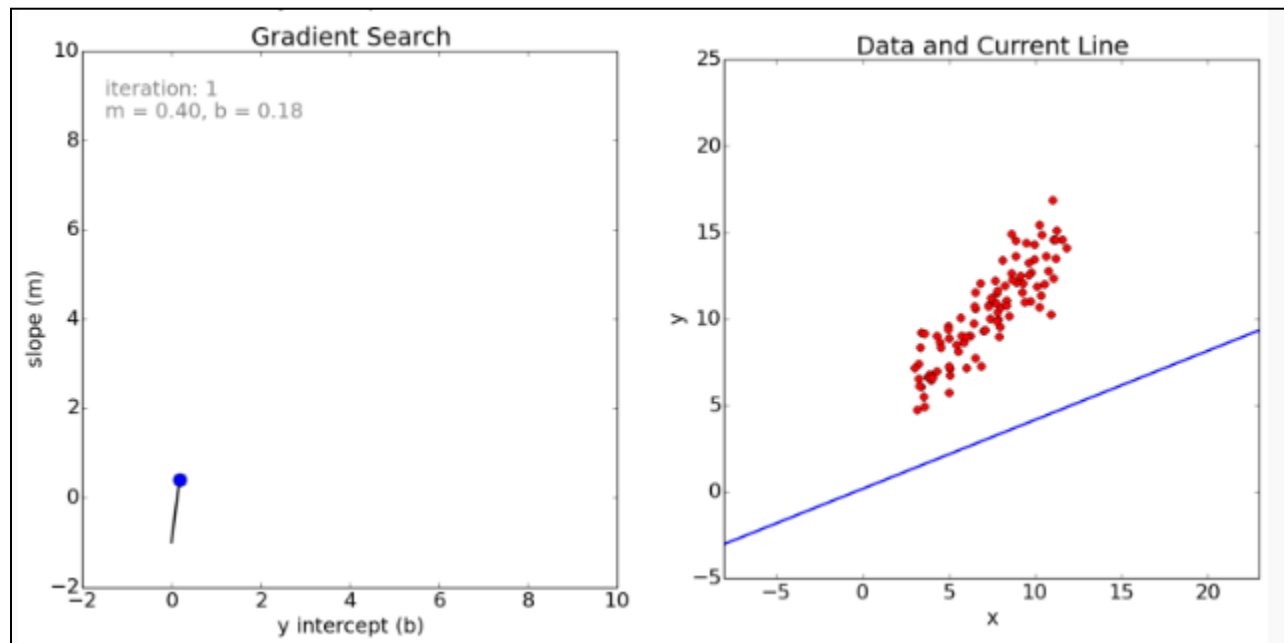
In the space  $(m, b)$  what is the function we are going to define? Minimum value of that function should be a solution for us.

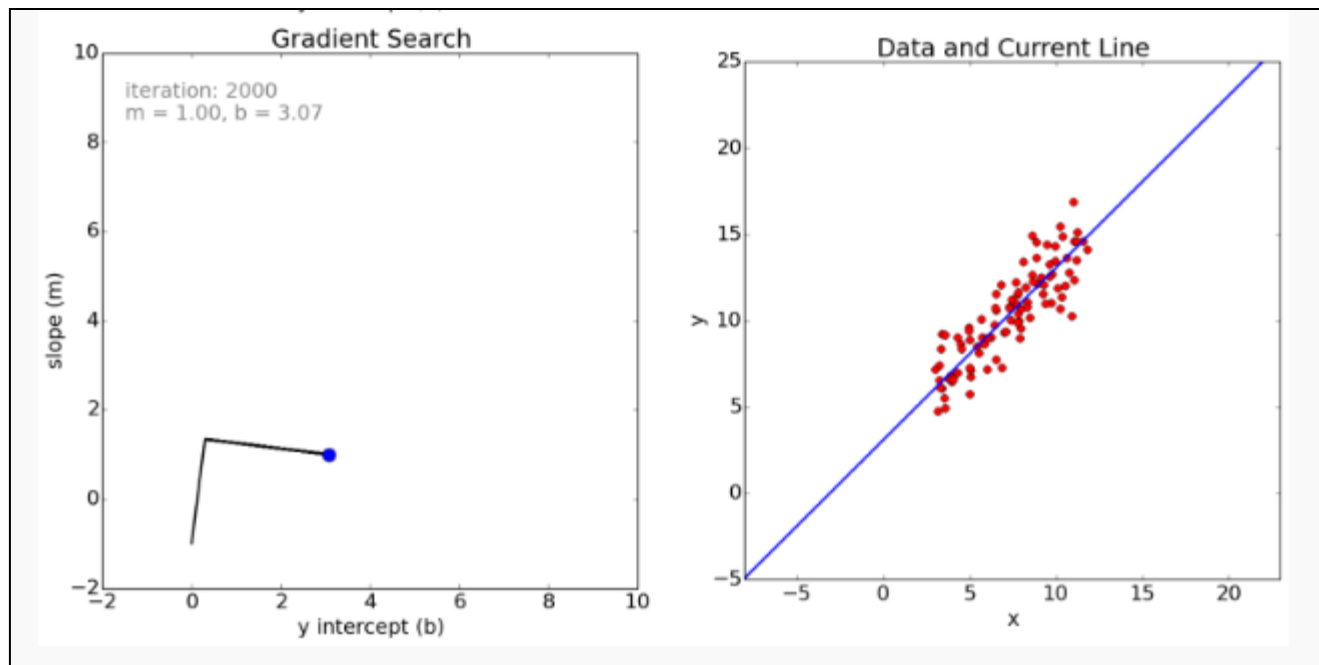
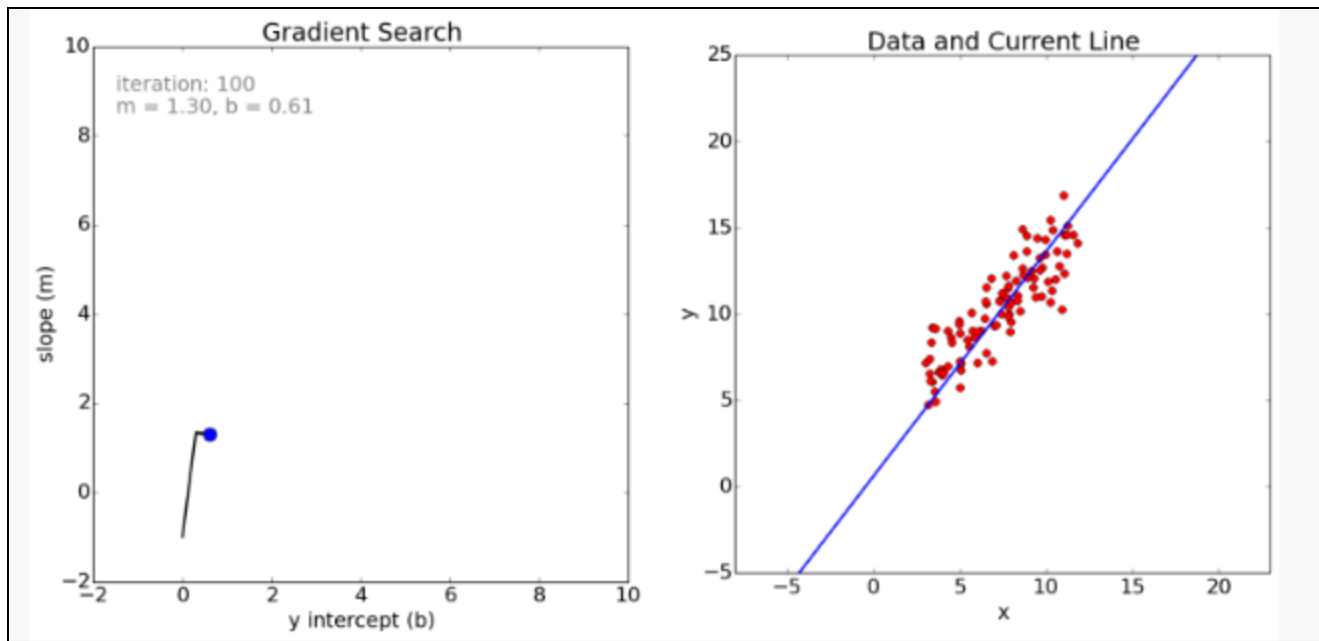
$$\text{Error}_{(m,b)} = \frac{1}{N} \sum_{i=1}^N (y_i - (mx_i + b))^2$$

$$\frac{\partial}{\partial m} = \frac{2}{N} \sum_{i=1}^N -x_i (y_i - (mx_i + b))$$

$$\frac{\partial}{\partial b} = \frac{2}{N} \sum_{i=1}^N -(y_i - (mx_i + b))$$







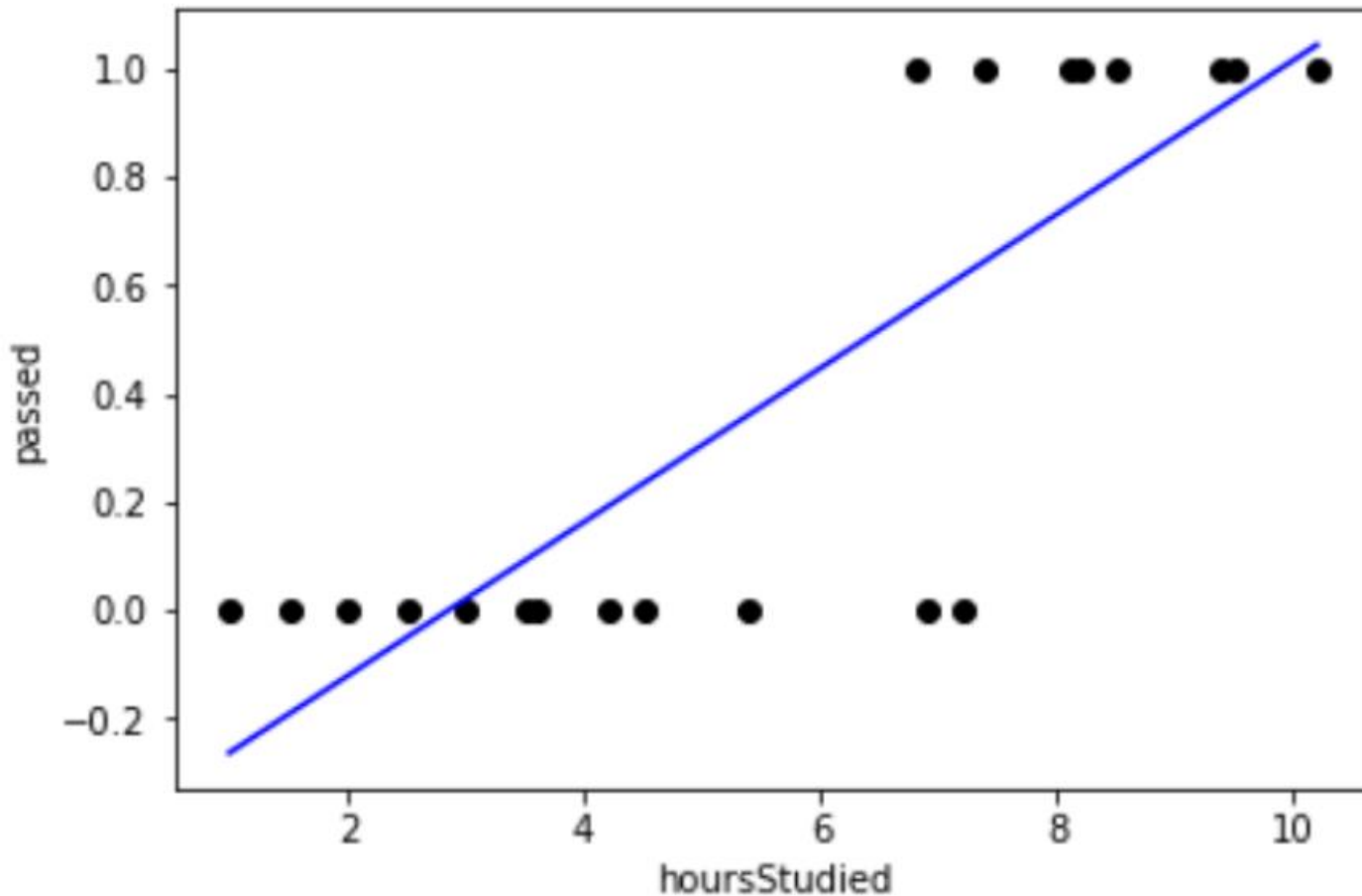
# Closed form solution

- Since the criterion that is minimized is quadratic, the linear regression problem must be having a closed form solution.
- This is nothing but applying the Newton's descent method.

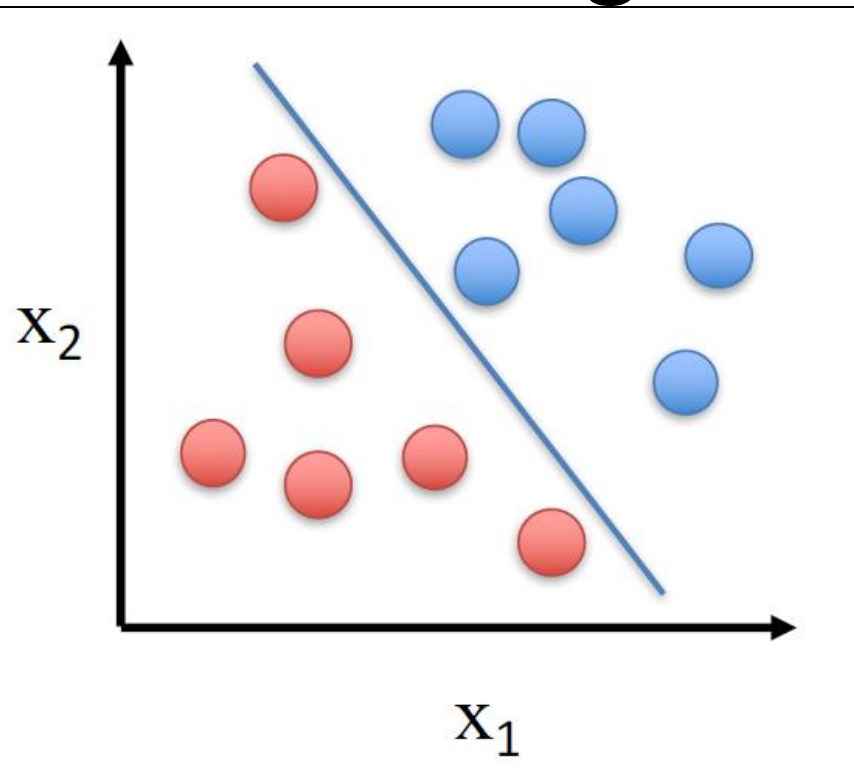


# Linear Regression for classification

Example with 1D data

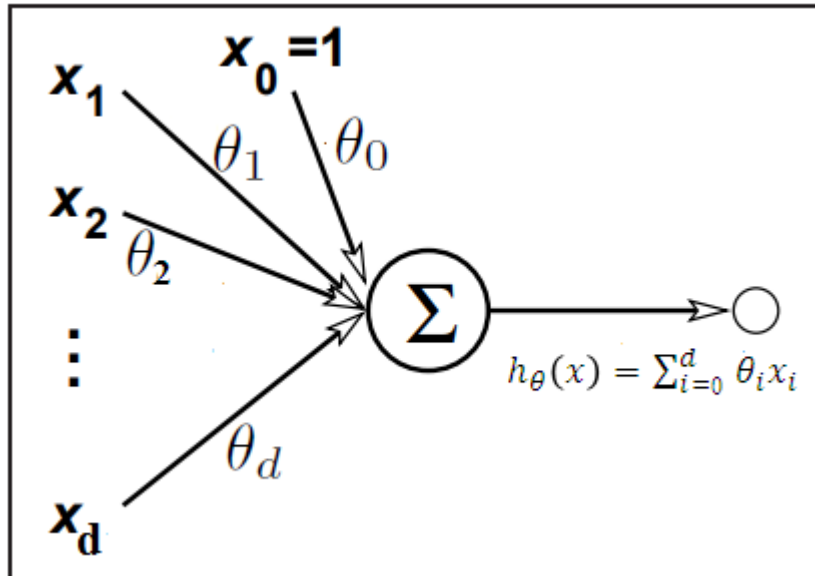


# Linear Regression for classification



Example with 2D data

# Direct attempt, in learning the linear discriminant



$$h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \cdots \theta_d x_d$$

The parameter vector  $\theta$  is learnt from the data such that the sum of squared error

$$E(\theta) = \frac{1}{2} \sum_{i=1}^n (y^{(i)} - h_{\theta}(x^{(i)}))^2$$

Where the training set

$$D = \{(x^{(i)}, y^{(i)}) | i = 1, \dots, n\}$$

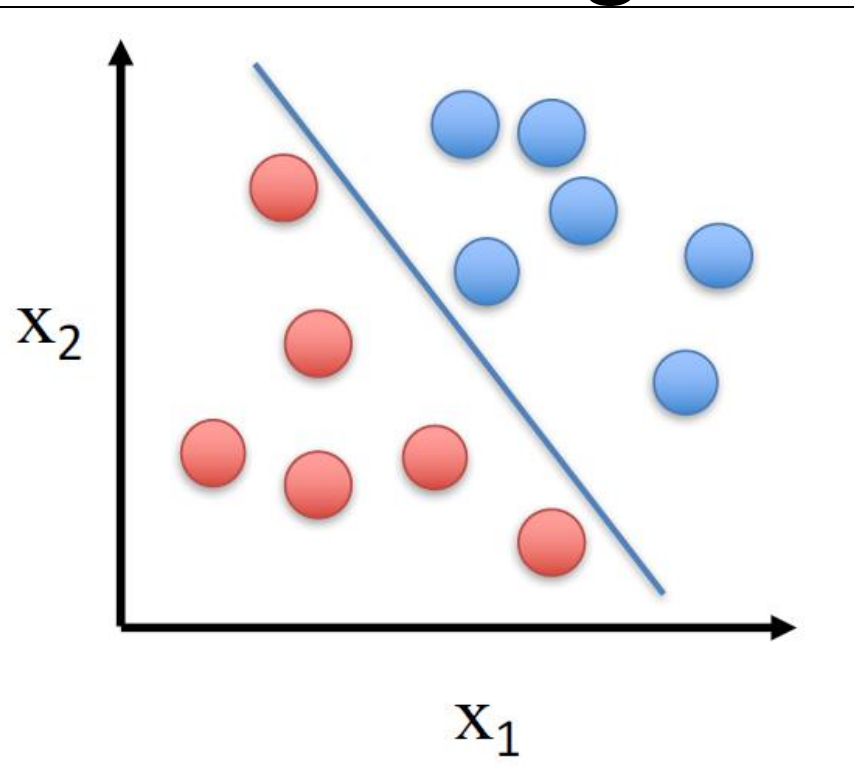
*$E(\theta)$  is convex(also quadratic)*

*$\Rightarrow$  no local minima problem (closed form solution)*

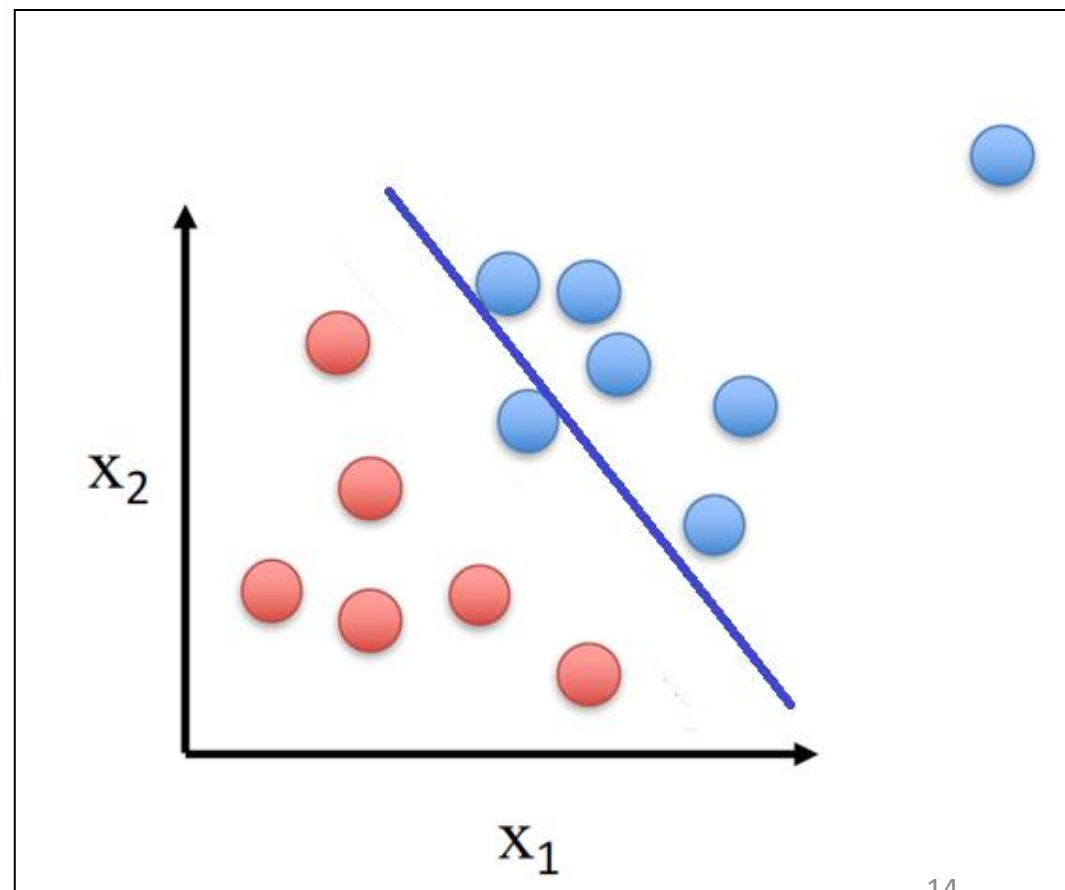
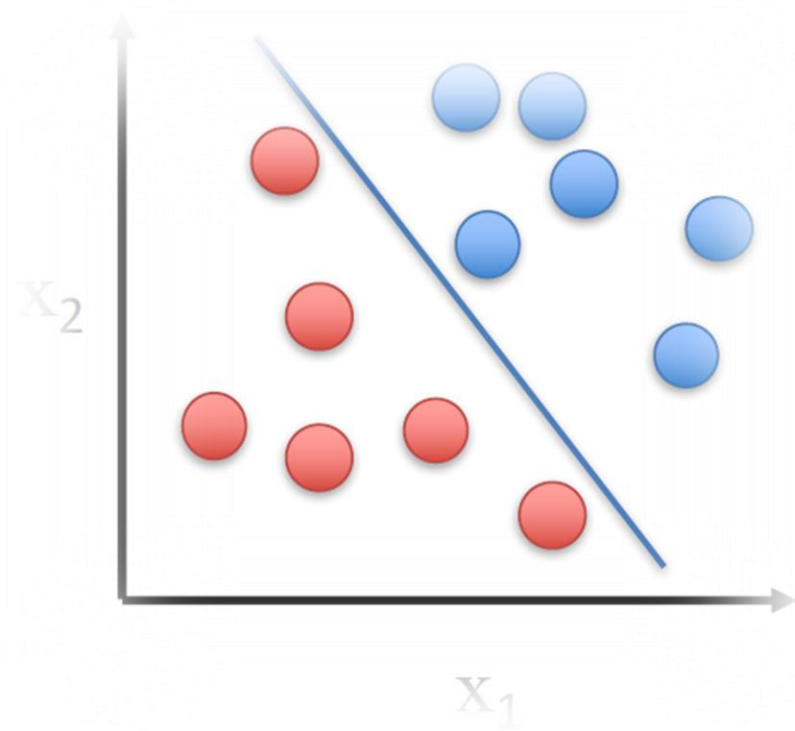
# Training Procedure

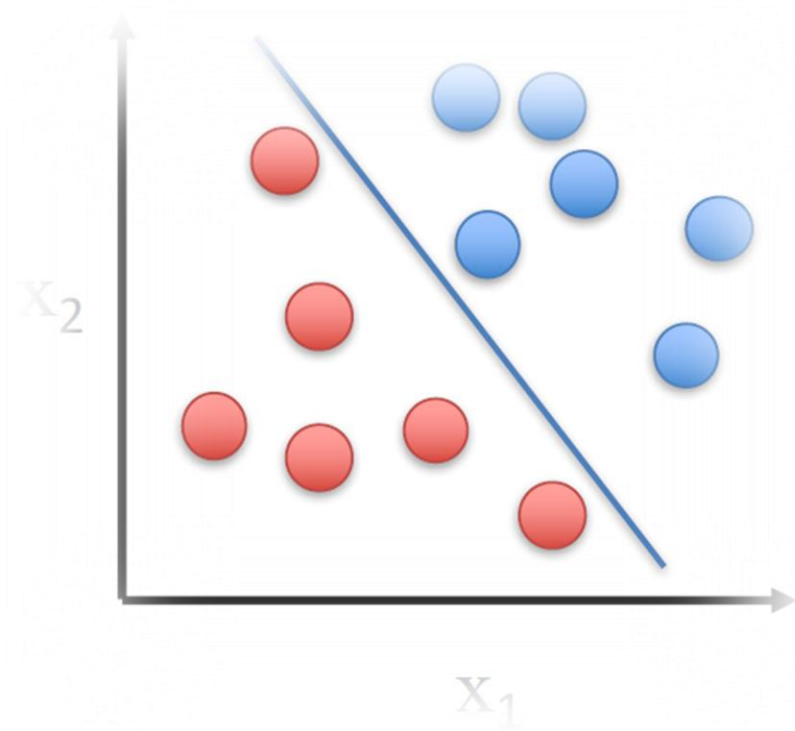
- $\nabla_{\theta}(E) = \sum_{i=1}^n (y^{(i)} - h_{\theta}(x^{(i)}))(-x^{(i)})$
- *Batch Method:*
- $\theta_{new} = \theta + \eta \sum_{i=1}^n (y^{(i)} - h_{\theta}(x^{(i)})) x^{(i)}$
- Single Sample (Stochastic update)
- $\theta_{new} = \theta + \eta (y^{(i)} - h_{\theta}(x^{(i)})) x^{(i)}$
- $h_{\theta}(x^{(i)}) = \theta^T x^{(i)}$

# Linear Regression for classification



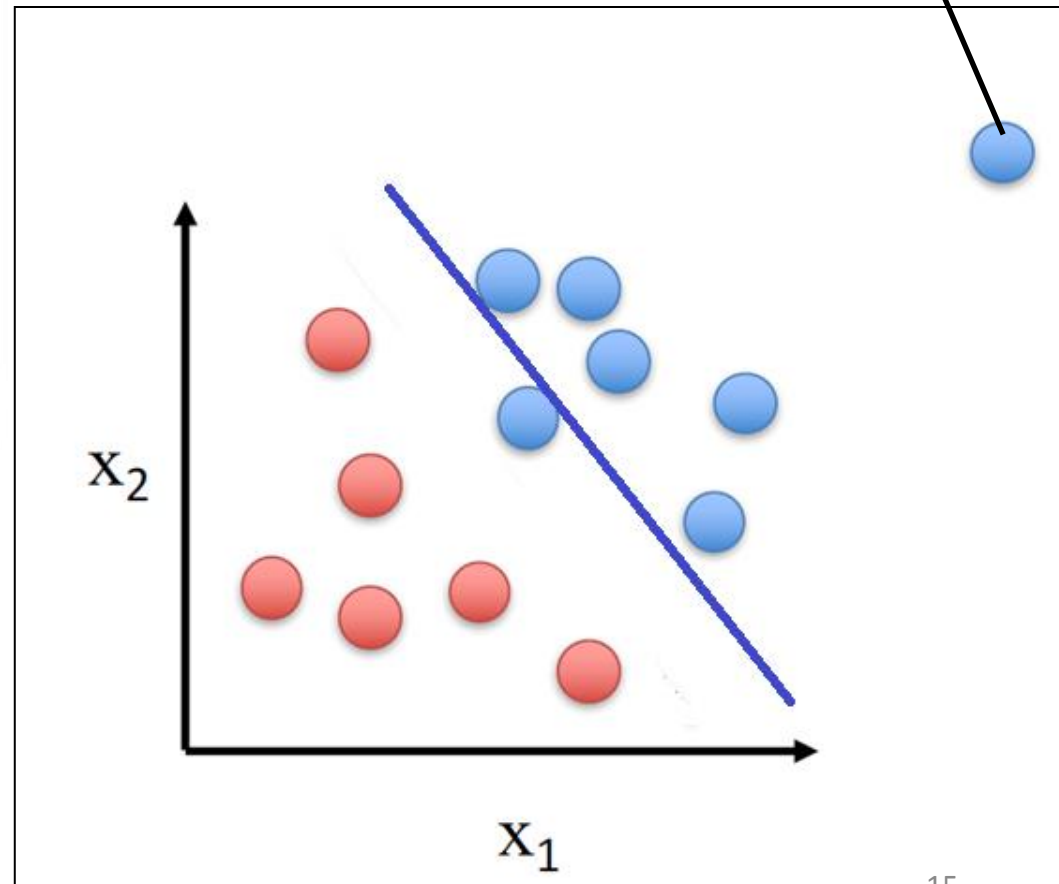
- But, outliers can be a big problem.





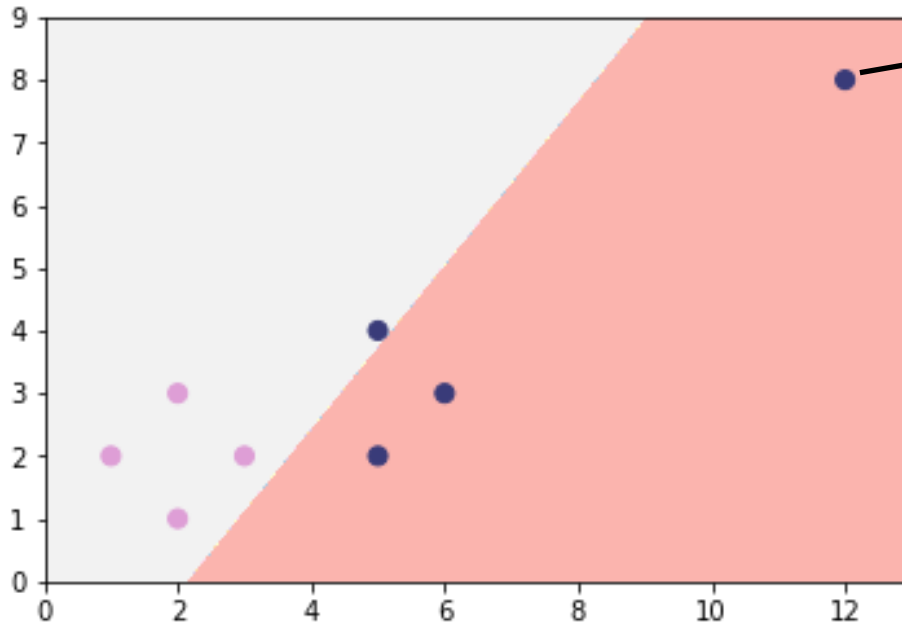
This is well within the blue class. This is a good training example.

**Why this is causing us problem??**



This is well within the blue class. This is a good training example.

**Why this is causing us problem??**





# Note

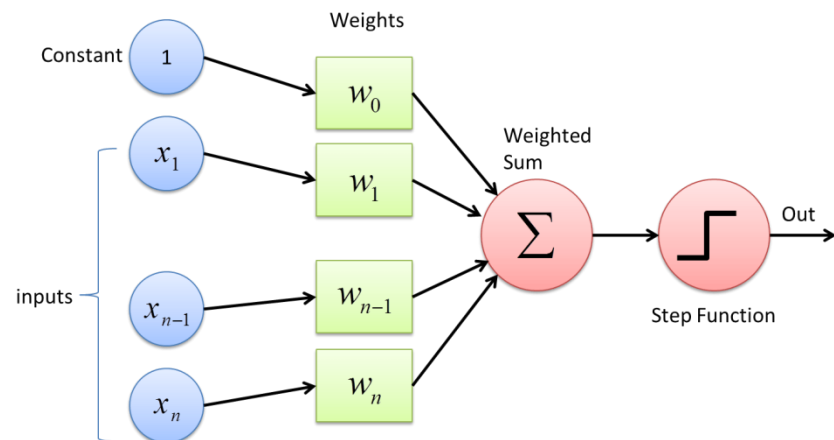
- Perceptron (Rosenblatt) does not have this problem (i.e., problem with seeming outliers).

# Note

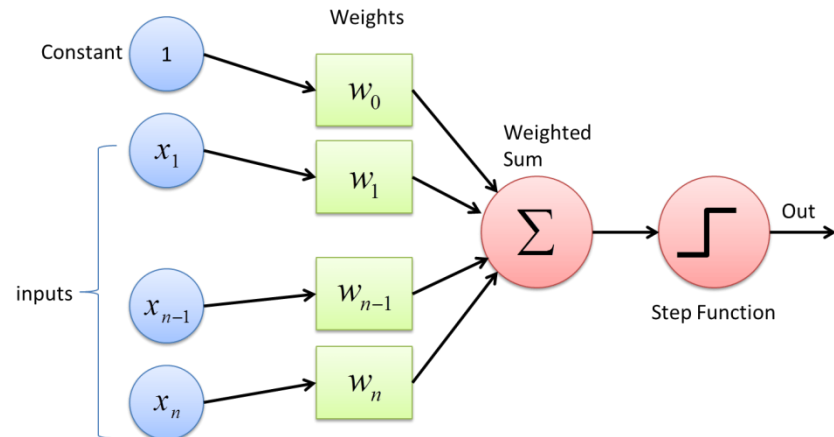
- Perceptron (Rosenblatt) does not have this problem (i.e., problem with seeming outliers).
- But works only for linearly separable data.

# Note

- Perceptron (Rosenblatt) does not have this problem (i.e., problem with seeming outliers).
- But works only for linearly separable data.
- Actually, it is

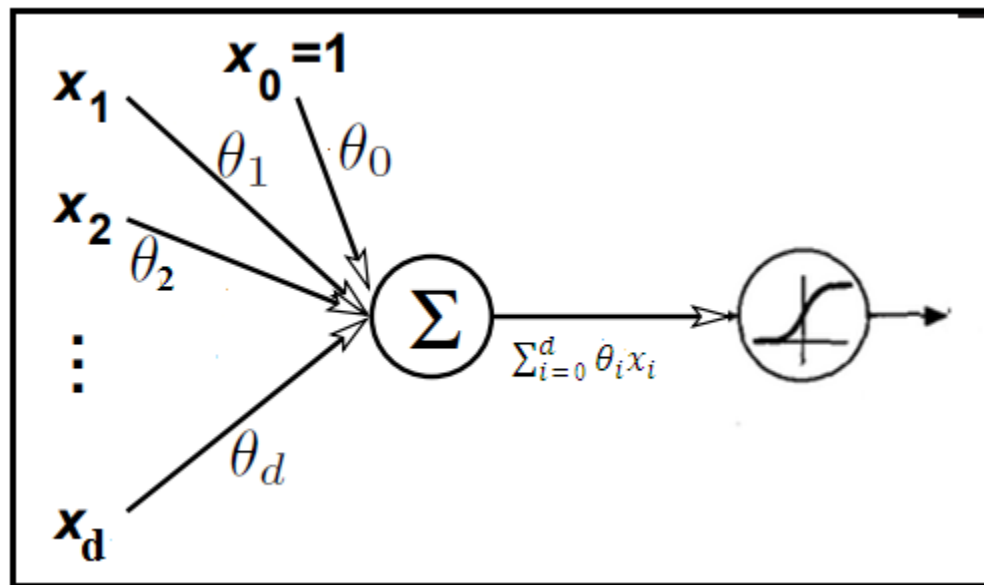


- Step (or Sign) function is not differentiable.
- So we can't employ gradient descent to get the minimum error (in this formulation) solution.

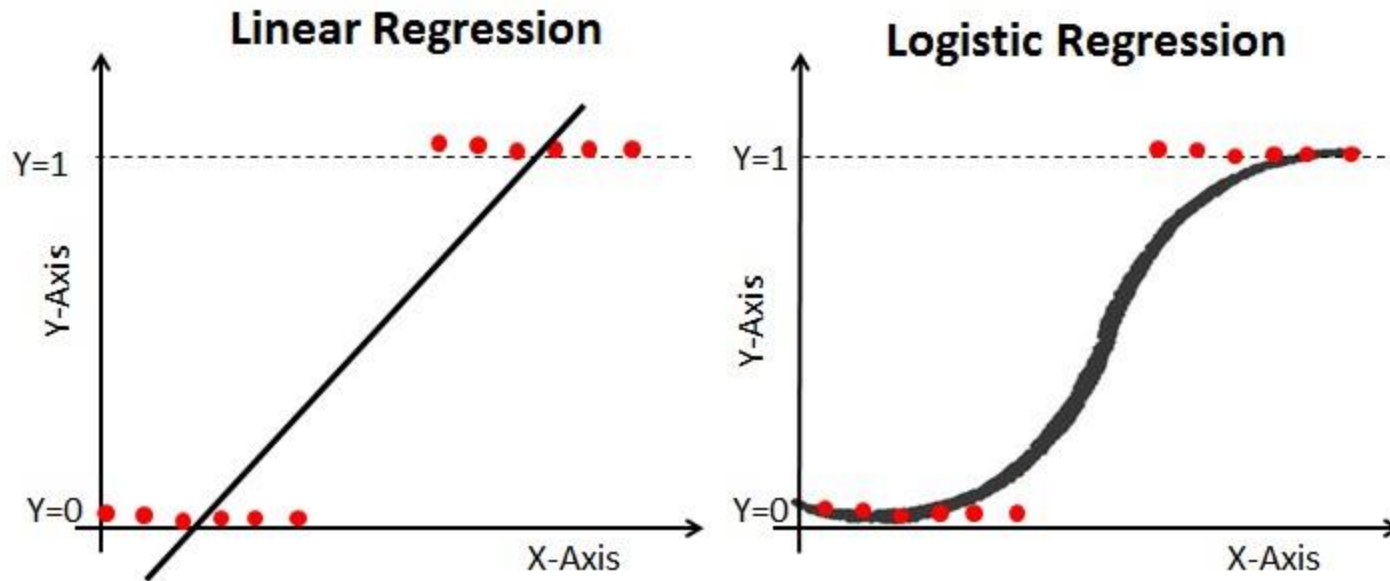




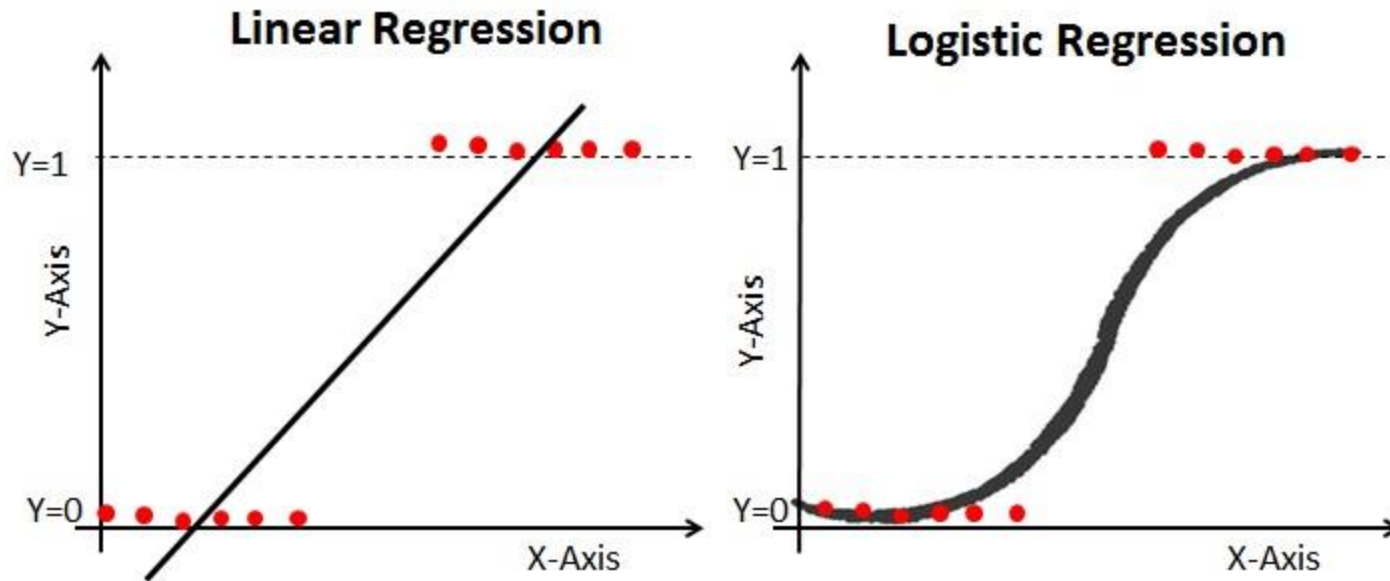
# Logistic Regression



# 1D : what it does



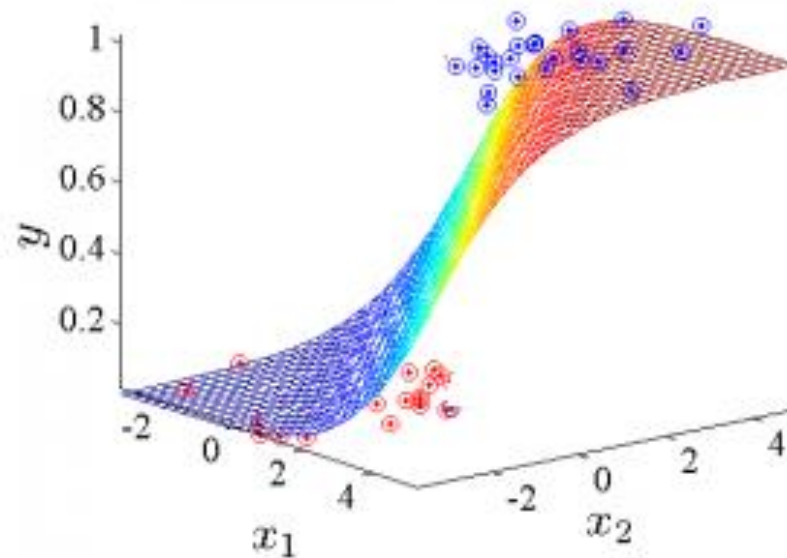
# 1D : what it does



See, the seeming outliers are indeed very good fits.



# 2D : what it does



# Notation used

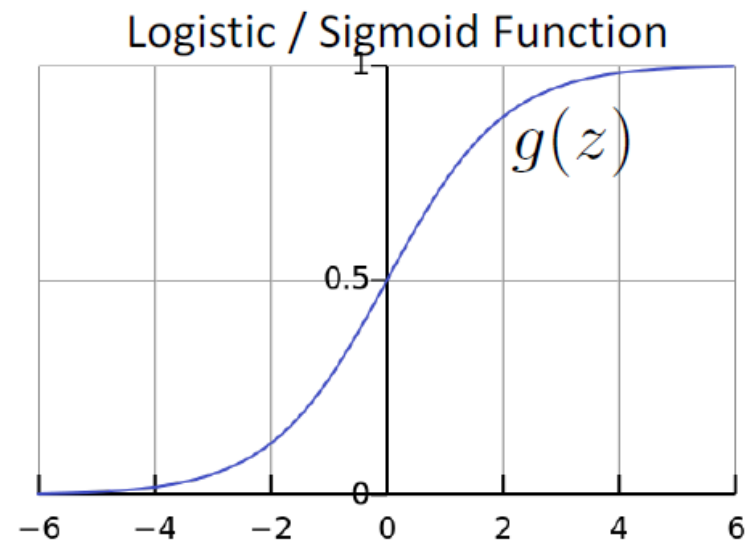
- $\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix}, x = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}$

- Logistic regression model:

$$h_{\theta}(x) = g(\theta^T x)$$

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

$$h_{\theta}(x) = \frac{1}{1 + e^{-\theta^T x}}$$



# Notation used

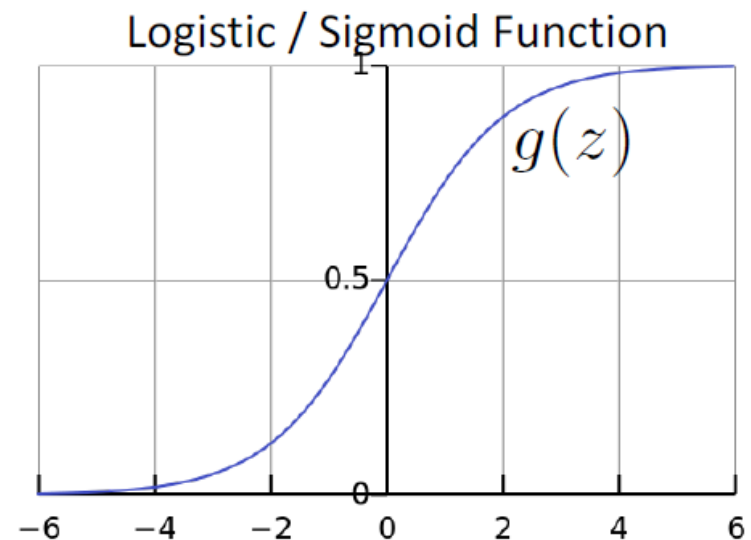
- Given  $\left\{ \left( \mathbf{x}^{(1)}, y^{(1)} \right), \left( \mathbf{x}^{(2)}, y^{(2)} \right), \dots, \left( \mathbf{x}^{(n)}, y^{(n)} \right) \right\}$   
where  $\mathbf{x}^{(i)} \in \mathbb{R}^d$ ,  $y^{(i)} \in \{0, 1\}$

- Logistic regression model:

$$h_{\theta}(\mathbf{x}) = g(\boldsymbol{\theta}^{\top} \mathbf{x})$$

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

$$h_{\theta}(\mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^{\top} \mathbf{x}}}$$



# Differentiation of the sigmoid

$$\begin{aligned} g'(z) &= \frac{d}{dz} \frac{1}{1 + e^{-z}} \\ &= \frac{1}{(1 + e^{-z})^2} (e^{-z}) \\ &= \frac{1}{(1 + e^{-z})} \cdot \left( 1 - \frac{1}{(1 + e^{-z})} \right) \\ &= g(z)(1 - g(z)). \end{aligned}$$

# Logistic Regression Objective Function

- Can't just use squared loss as in linear regression:

$$J(\boldsymbol{\theta}) = \frac{1}{2n} \sum_{i=1}^n \left( h_{\boldsymbol{\theta}} \left( \boldsymbol{x}^{(i)} \right) - y^{(i)} \right)^2$$

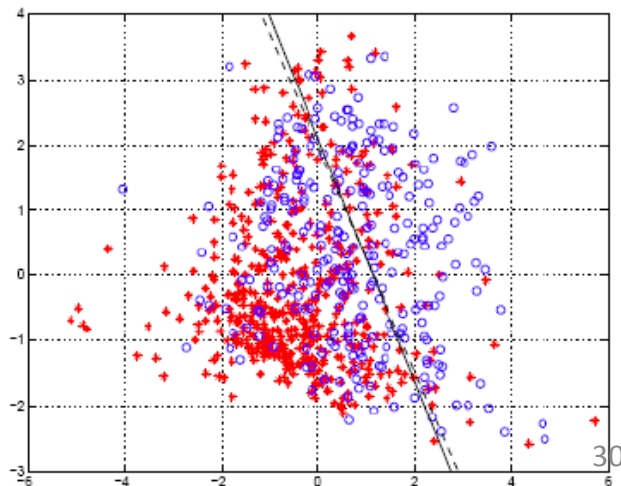
- Using the logistic regression model

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^T \boldsymbol{x}}}$$


results in a non-convex optimization

# Classification Based on Probability

- Instead of just predicting the class, give the probability of the instance being that class
  - i.e., learn  $p(y \mid \mathbf{x})$
- Comparison to perceptron:
  - Perceptron doesn't produce probability estimate
- Recall that:
$$0 \leq p(\text{event}) \leq 1$$
$$p(\text{event}) + p(\neg \text{event}) = 1$$



- Takes a probabilistic approach to learning discriminative functions (i.e., a classifier)
- $h_{\theta}(\mathbf{x})$  should give  $p(y = 1 \mid \mathbf{x}; \theta)$ 
  - Want  $0 \leq h_{\theta}(\mathbf{x}) \leq 1$



Can't just use linear regression with a threshold

# Interpretation of Hypothesis Output

$$h_{\theta}(\mathbf{x}) = \text{estimated } p(y = 1 \mid \mathbf{x}; \theta)$$

Example: Cancer diagnosis from tumor size

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumorSize} \end{bmatrix}$$

$$h_{\theta}(\mathbf{x}) = 0.7$$

→ Tell patient that 70% chance of tumor being malignant

Note that:  $p(y = 0 \mid \mathbf{x}; \theta) + p(y = 1 \mid \mathbf{x}; \theta) = 1$

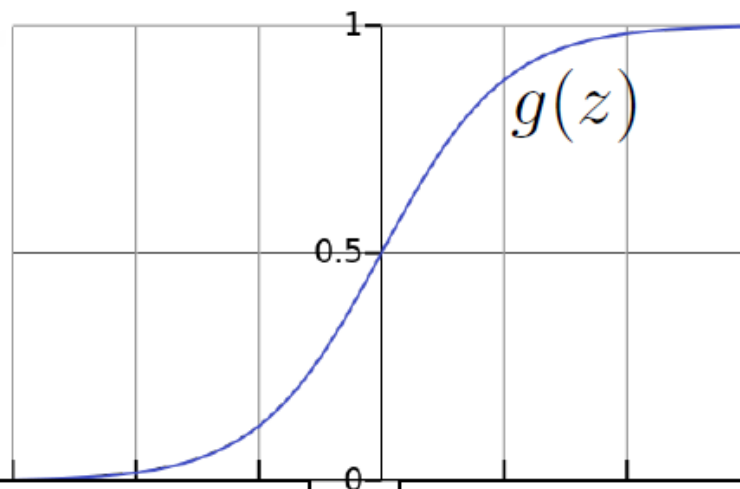
Therefore,  $p(y = 0 \mid \mathbf{x}; \theta) = 1 - p(y = 1 \mid \mathbf{x}; \theta)$



# Logistic Regression

$$h_{\theta}(x) = g(\theta^T x)$$

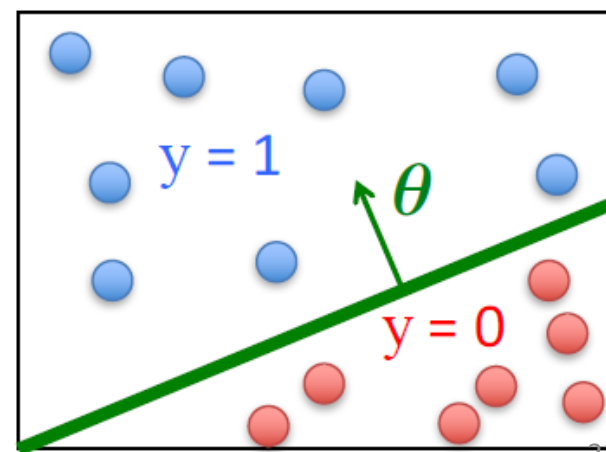
$$g(z) = \frac{1}{1 + e^{-z}}$$



$\theta^T x$  should be large negative values for negative instances

$\theta^T x$  should be large positive values for positive instances

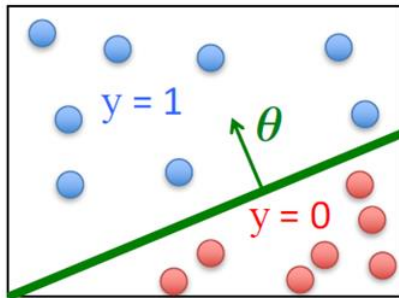
- Assume a threshold and...
  - Predict  $y = 1$  if  $h_{\theta}(x) \geq 0.5$
  - Predict  $y = 0$  if  $h_{\theta}(x) < 0.5$



# What we want ...??

- We have
$$P(y = 1 \mid x; \theta) = h_{\theta}(x)$$
$$P(y = 0 \mid x; \theta) = 1 - h_{\theta}(x)$$

- Given



- Find  $\theta$  so that the data agrees to the maximum extent.

- We can use maximum likelihood parameter estimation (MLE)

- Likelihood of data is given by:  $l(\boldsymbol{\theta}) = \prod_{i=1}^n p(y^{(i)} \mid \boldsymbol{x}^{(i)}; \boldsymbol{\theta})$
- So, looking for the  $\boldsymbol{\theta}$  that maximizes the likelihood
- Can take the log without changing the solution:

$$\begin{aligned}\boldsymbol{\theta}_{\text{MLE}} &= \arg \max_{\boldsymbol{\theta}} \log \prod_{i=1}^n p(y^{(i)} \mid \boldsymbol{x}^{(i)}; \boldsymbol{\theta}) \\ &= \arg \max_{\boldsymbol{\theta}} \sum_{i=1}^n \log p(y^{(i)} \mid \boldsymbol{x}^{(i)}; \boldsymbol{\theta})\end{aligned}$$

$$\begin{aligned}P(y = 1 \mid x; \theta) &= h_{\theta}(x) \\P(y = 0 \mid x; \theta) &= 1 - h_{\theta}(x)\end{aligned}$$

Note that this can be written more compactly as

$$p(y \mid x; \theta) = (h_{\theta}(x))^y (1 - h_{\theta}(x))^{1-y}$$

# Likelihood

$$\begin{aligned} L(\theta) &= \prod_{i=1}^n p(y^{(i)} \mid x^{(i)}; \theta) \\ &= \prod_{i=1}^n (h_{\theta}(x^{(i)}))^{y^{(i)}} (1 - h_{\theta}(x^{(i)}))^{1-y^{(i)}} \end{aligned}$$

- Log-likelihood

$$\begin{aligned} \ell(\theta) &= \log L(\theta) \\ &= \sum_{i=1}^n y^{(i)} \log h(x^{(i)}) + (1 - y^{(i)}) \log(1 - h(x^{(i)})) \end{aligned}$$

# Find $\theta$ that maximizes $l(\theta)$

- $l(\theta)$  is taken to be the criterion, but, note this has to be maximized.
- We can employ gradient ascent method.
- We can show that, the negative of the log-likelihood, i.e.,  $-l(\theta)$  which should be minimized is convex (hence no local minima problem)
- For proof of this refer the link  
<http://mathgotchas.blogspot.com/2011/10/why-is-error-function-minimized-in.html>

- $\nabla_{\theta} l(\theta) = (y - h_{\theta}(x)) x$ 
  - Note, here  $x$  is a vector.
  - This is for a single training example.

- This we obtained from,

$$\begin{aligned}
 \frac{\partial}{\partial \theta_j} \ell(\theta) &= \left( y \frac{1}{g(\theta^T x)} - (1 - y) \frac{1}{1 - g(\theta^T x)} \right) \frac{\partial}{\partial \theta_j} g(\theta^T x) \\
 &= \left( y \frac{1}{g(\theta^T x)} - (1 - y) \frac{1}{1 - g(\theta^T x)} \right) g(\theta^T x)(1 - g(\theta^T x)) \frac{\partial}{\partial \theta_j} \theta^T x \\
 &= (y(1 - g(\theta^T x)) - (1 - y)g(\theta^T x)) x_j \\
 &= (y - h_{\theta}(x)) x_j
 \end{aligned}$$



# Stochastic or single sample update rule

- $$\begin{aligned}\theta_{new} &= \theta + \eta \nabla_{\theta} l(\theta) \\ &= \theta + \eta (y - h_{\theta}(x)) x\end{aligned}$$

## Batch Method

$$\theta_{new} = \theta + \eta \sum_{i=1}^n (y^{(i)} - h_{\theta}(x^{(i)})) x^{(i)}$$

# Stochastic or single sample update rule

- $$\begin{aligned}\theta_{new} &= \theta + \eta \nabla_{\theta} l(\theta) \\ &= \theta + \eta (y - h_{\theta}(x)) x\end{aligned}$$

## Batch Method

$$\theta_{new} = \theta + \eta \sum_{i=1}^n (y^{(i)} - h_{\theta}(x^{(i)})) x^{(i)}$$

*This looks IDENTICAL to linear regression!!!*

- However, the form of the model is very different:*

$$h_{\theta}(x) = \frac{1}{1 + e^{-\theta^T x}}$$

- The End

- Supplementary Material

- $h_{\theta}(x) = \sum_{i=1}^d \theta_i x_i$