برازش خطی چندمتغیره

❖ **برازش خطی چندگانه** (Multiple Linear Regression) برای پیشبینی یک متغیر وابسته (y) که به صورت خطی با بیش از یک متغیر مستقل ($x_1, x_2, \dots, x_n$) مرتبط است به کار می‌رود.

$$h_\theta(x) = \underbrace{\theta_0}_{x_0=1} + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n = \theta^T X$$

$x_j^{(i)}$: مقدار ویژگی j در نمونه i

$x^{(i)}$: ویژگی‌های نمونه i که می‌توانیم آن‌ها را به شکل بردار $\mathrm{X} = [1, x_1, x_2, \dots, x_n] \in \mathbb{R}^{n+1}$ تعریف کنیم. $x_0^{(i)} = 1$ است.

θ: بردار ضرایب که به شکل $\theta = [\theta_0, \theta_1, \theta_2, \dots, \theta_n] \in \mathbb{R}^{n+1}$ آن را تعریف می‌کنیم.

$J(\theta)$: MSE $\longrightarrow$

$$\theta_j := \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j} = \theta_j - \frac{\alpha}{m} \sum_{i=1}^{m} (h_\theta(x)^{(i)} - y^{(i)}) x_j^{(i)}$$

Multiple features

Size (feet²) x	Price ($1000) y
2104	460
1416	232
1534	315
852	178
…	…

Notation:

n= number of features

$x^{(i)}$ = input (features) of i^{th} training example.

$x_j^{(i)}$ = value of feature j in i^{th} training example.

x_1 Size (feet²)	x_2 Number of bedrooms	x_3 Number of floors	x_4 Age of home (years)	Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
…	…	…	…	…

$$h_\theta(x) = \theta_0 + \theta_1 x$$

Multiple features

$x^{(i)} \longrightarrow y^{(i)}$

Hypothesis

Previously: $h_\theta(x) = \theta_0 + \theta_1 x$

Multivariate linear regression: $h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$

x_1 Size (feet²)	x_2 Number of bedrooms	x_3 Number of floors	x_4 Age of home (years)	y Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
…	…	…	…	…

$h_\theta(X) = \underline{\theta}^T \underline{X}$

$\underline{\theta} = [\theta_0, \theta_1, \dots, \theta_n]^T$

$\underline{X} = [x_0, x_1, \dots, x_n]^T$, $x_0 = 1$

size $n+1$

$\underline{X}^{(i)} = \begin{bmatrix} x_0^{(i)} = 1 \\ x_1^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix}$

$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4$

Gradient descent for multiple variables

Hypothesis: $h_\theta(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$

Parameters: $\theta_0, \theta_1, \ldots, \theta_n$

Cost function:

$$J(\theta_0, \theta_1, \ldots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Gradient descent:

Repeat $\{$

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

$\}$ (simultaneously update for every $j = 0, \ldots, n$)

Gradient Descent

Previously (n=1):

كـ مميزة

$n = 1$

Repeat{

$$\theta_0 := \underbrace{\theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})}_{\frac{\partial}{\partial \theta_0} J(\theta)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$$

(simultaneously update θ_0, θ_1)

}

New algorithm $(n \geq 1)$:

عدة مميزة

$n > 1$

Repeat {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

(simultaneously update θ_j for $j = 0, \ldots, n$)

}

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)} = 1$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_1^{(i)}$$

$$\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_2^{(i)}$$

$\ldots$

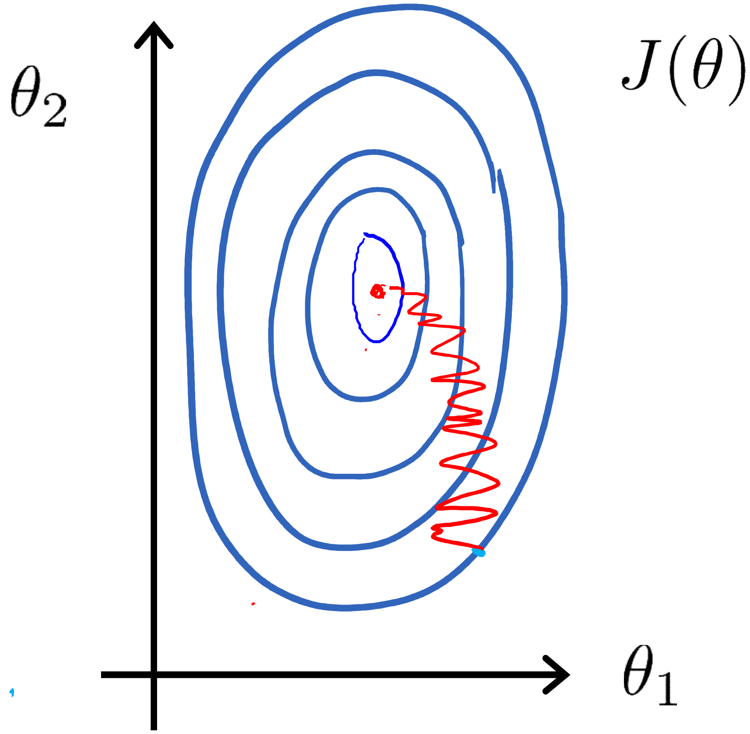
$$\theta_j = \theta - \alpha \frac{\partial J}{\partial \theta}$$

Feature Scaling

Idea: Make sure features are on a similar scale.

E.g. x_1 = size (0-2000 feet^2)

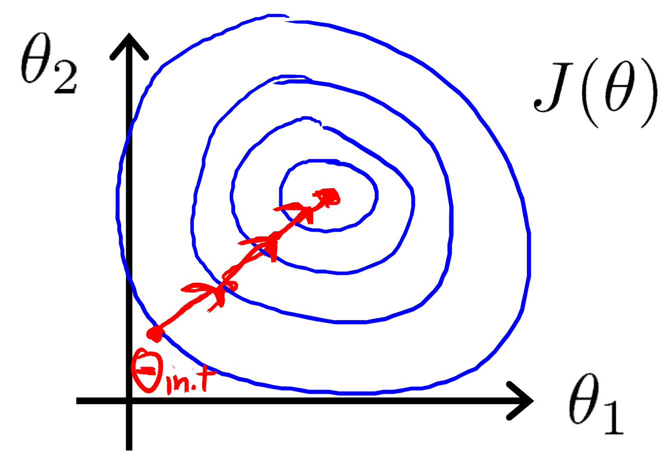
x_2 = number of bedrooms (1-5)



$$x_1 = \frac{\text{size (feet}^2\text{)}}{2000}$$

$$x_2 = \frac{\text{number of bedrooms}}{5}$$

$0 \le x_1 \le 1$ $0 \le x_2 \le 1$



Mean normalization

Replace x_i with $x_i - \mu_i$ to make features have approximately zero mean (Do not apply to $x_0 = 1$).

E.g. $x_1 = \frac{size - 1000}{2000}$

$x_2 = \frac{\#bedrooms - 2}{5}$

$$-0.5 \le x_1 \le 0.5, -0.5 \le x_2 \le 0.5$$

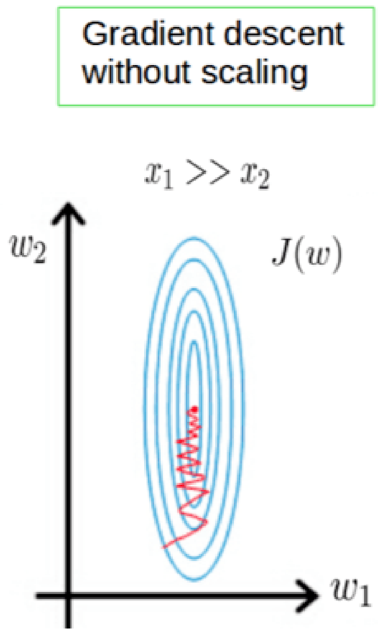
$$x_1 \leftarrow \frac{x_1 - \mu_1}{s_1} \qquad x_2 \leftarrow \frac{x_2 - \mu_2}{s_2}$$

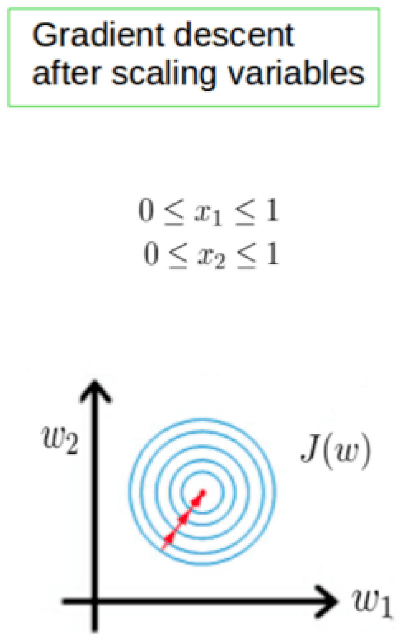
avg value of x_1 in training set

range (max - min)

(or standard deviation)

تغییر مقیاس متغیرها





min-max normalization

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Mean normalization

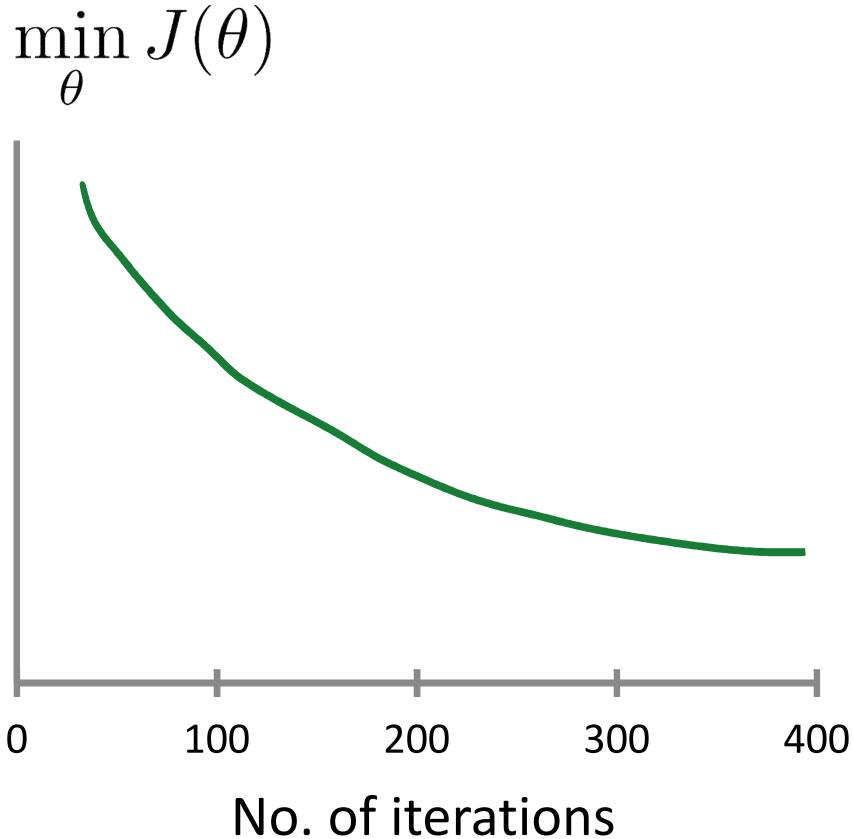
$$x' = \frac{x - mean(x)}{max(x) - min(x)}$$

Gradient descent

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

- "Debugging": How to make sure gradient descent is working correctly.
- How to choose learning rate α.

Making sure gradient descent is working correctly.



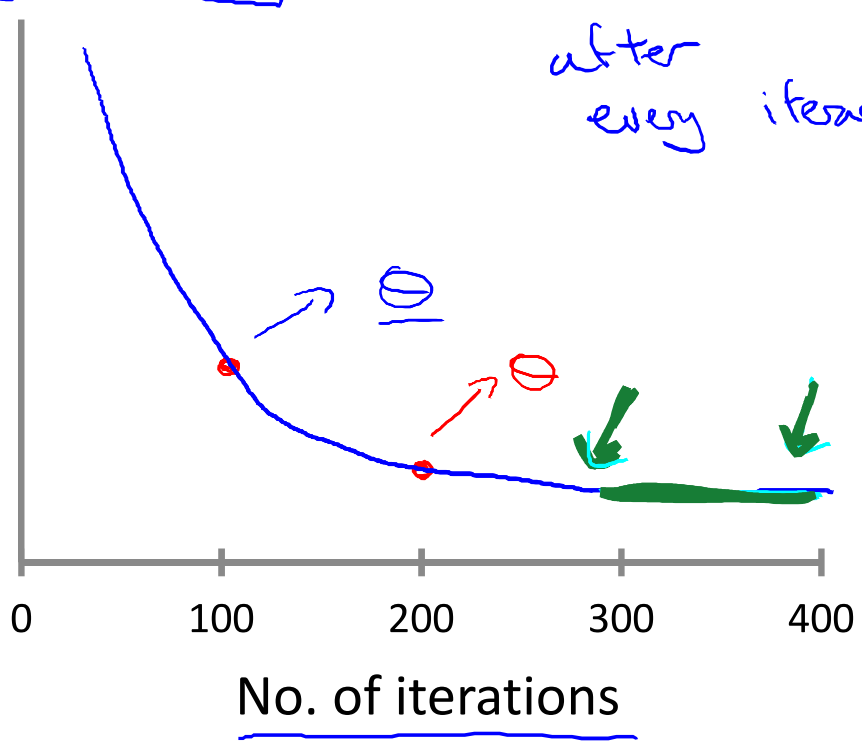
Example automatic convergence test:

Declare convergence if $J(\theta)$ decreases by less than 10^{-3} in one iteration.

Making sure gradient descent is working correctly.

$$\boxed{\min_{\theta} J(\theta)}$$

J(θ) should decrease after every iteration.



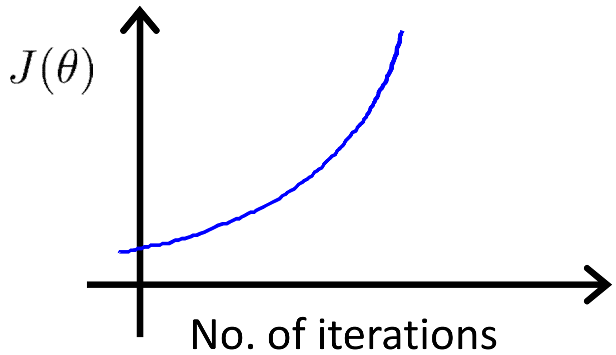
30, 3000, 3,000,000

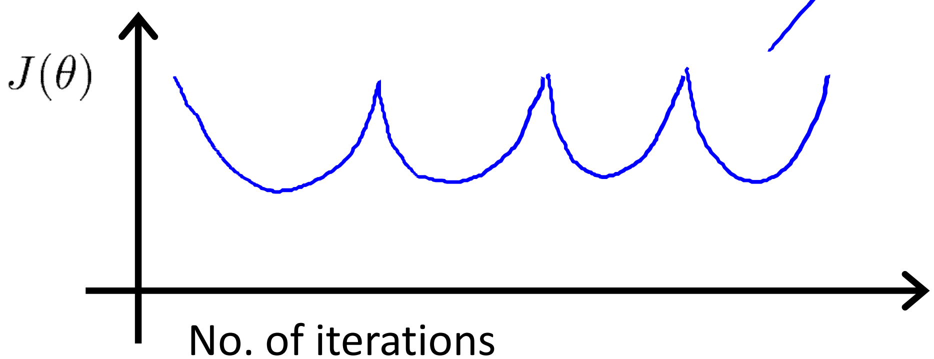
Example automatic convergence test:

Declare convergence if $J(\theta)$ decreases by less than $\boxed{10^{-3}}$ in one iteration.

ε

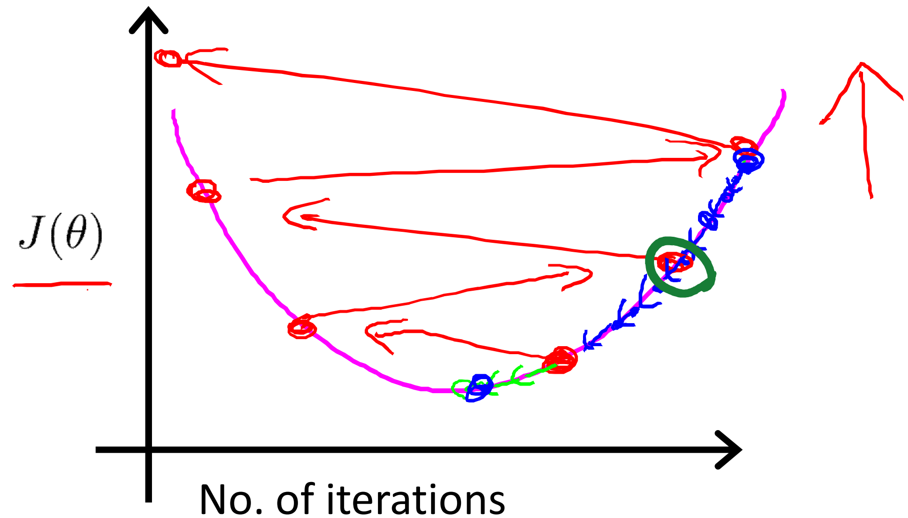
Making sure gradient descent is working correctly.





Gradient descent not working.

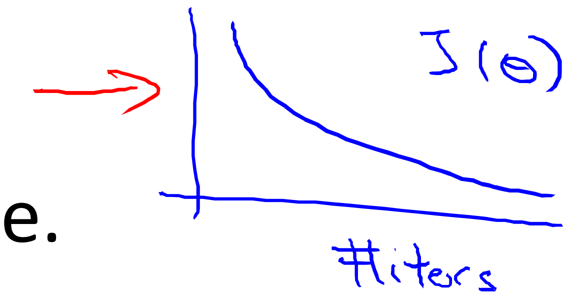
Use smaller α



- For sufficiently small α, $J(\theta)$ should decrease on every iteration.
- But if α is too small, gradient descent can be slow to converge.

Summary:

- If α is too small: slow convergence.
- If α is too large: $J(\theta)$ may not decrease on every iteration; may not converge.



(Slow converge also possible.)

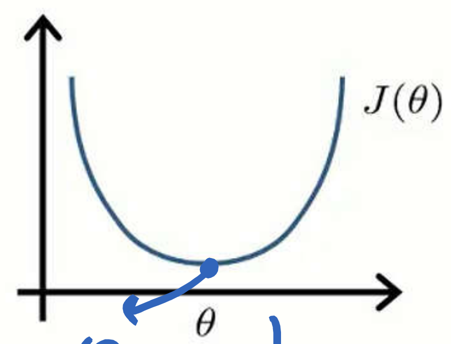
To choose α, try

$$\ldots, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, \ldots$$

3x ≈3x 3x ≈3x

روش آماری Normal Equation

$$J(\theta) = a\theta^2 + b\theta + c$$



$\theta = \frac{-b}{2a}$

حالت ساده به‌دست آوردن کمینه یک تابع:

$$\frac{\mathrm{d}J(\theta)}{\mathrm{d}\theta} = 0 \rightarrow \boxed{\theta = \frac{-b}{2a}}$$

اما در مسئله ما $\theta \in \mathbb{R}^{n+1}$ است و مشتق باید برای همه θ ها برابر صفر قرار بگیرد و حل شود.

$$\frac{\partial J(\theta)}{\partial \theta_j} = 0 \ (for\ every\ j) \rightarrow \theta_0, \dots, \theta_n$$

$j = 0 \dots n$

Normal equation

(رگرسیون خطی)

$\theta \in \mathbb{R}^{n+1} \Rightarrow J(\theta_0, \theta_1, \dots, \theta_n) \quad \frac{\partial J}{\partial \theta_j} = 0 \Rightarrow$ $j = 0 \dots n$

محاسبه $\theta_0, \dots, \theta_n$

Normal equation: Analytic solution

$$\begin{array}{cccccc} & X_0 & X_1 & X_2 & X_4\ 3 & X_5\ 4 \\ X = & \left[\begin{array}{c} 1 \\ 1 \\ 1 \\ 1 \end{array}\right. & \begin{array}{c} 2104 \\ 1416 \\ 1534 \\ 852 \end{array} & \begin{array}{c} 5 \\ 3 \\ 3 \\ 2 \end{array} & \begin{array}{c} 1 \\ 2 \\ 2 \\ 1 \end{array} & \left.\begin{array}{c} 45 \\ 40 \\ 30 \\ 36 \end{array}\right] \end{array} \qquad y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

$$\underline{\underline{X}} = \begin{bmatrix} (x^{(1)})^T \\ \vdots \\ (x^{(m)})^T \end{bmatrix}_{m \times (n+1)} \qquad \underline{y} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix}_{m \times 1}$$

$$\theta = \left(X^T X\right)^{-1} X^T y$$

$(X^T X)^{-1}$ is inverse of matrix $X^T X$.

* مقایسه کردن ویژگی‌ها لازم است

m **training examples,** n **features.**

Gradient Descent

- Need to choose α.
- Needs many iterations.
- Works well even when n is large.

$n = 10^6$

Normal Equation

- No need to choose α.
- Don't need to iterate.
- Need to compute $(X^TX)^{-1}$ → $O(n^3)$
- Slow if n is very large.

$n = 100$

$n = 1000$

What if $X^T X$ is non-invertible? (singular/ degenerate)

$$\begin{array}{ccc} x_0 & & x_1 \end{array}$$
$$\begin{bmatrix} x^{(1)T} \\ x^{(m)T} \end{bmatrix}$$

- Redundant features (linearly dependent).
  E.g. x_1 size in feet^2
  x_2 size in m^2

$\Rightarrow X^T X$ singular

- Too many features (e.g. $m \leq n$).

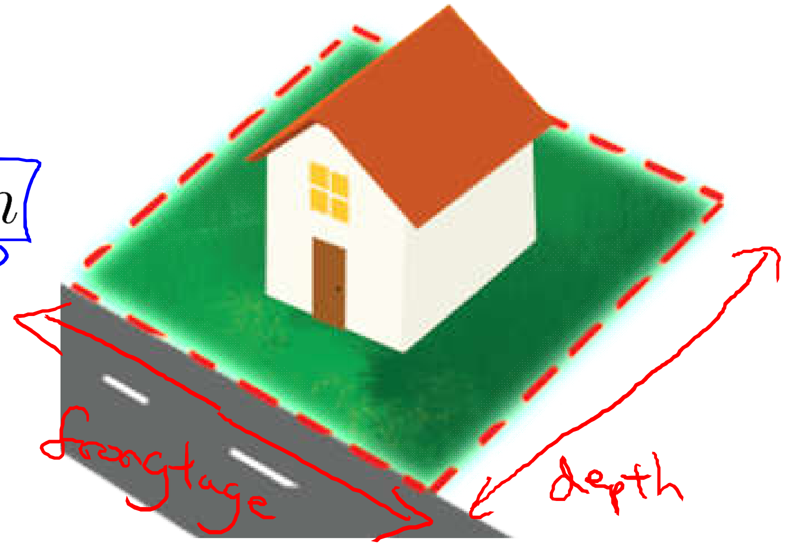
  $n = 100$ $\boxed{m = 10}$

  Θ: 101×1

  - Delete some features, or use regularization.

Housing prices prediction

$$h_\theta(x) = \theta_0 + \theta_1 \times \underbrace{\boxed{frontage}}_{x_1} + \theta_2 \times \underbrace{\boxed{depth}}_{x_2}$$



Area

$x = \underline{frontage * depth}$

$h_\theta(x) = \theta_0 + \theta_1 x$

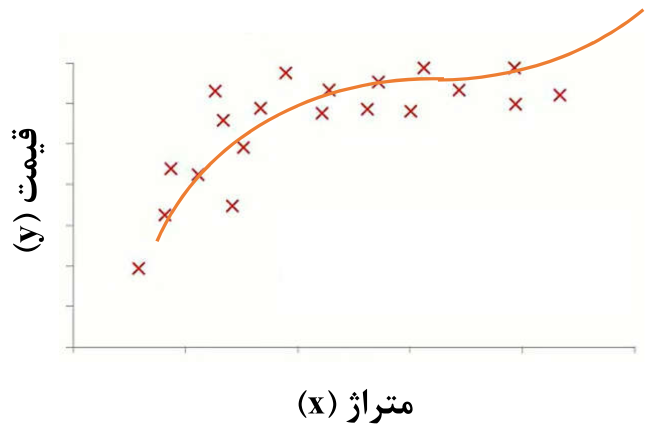
land area

برازش چندجمله

❖ **برازش چندجمله‌ای** (Polynomial Regression) برای پیشبینی متغیر وابسته‌ای که به صورت چندجمله‌ای با متغیرهای مستقل مرتبط است به کار می‌رود.

مثال

برازش چندجمله‌ای: پیش‌بینی قیمت خانه

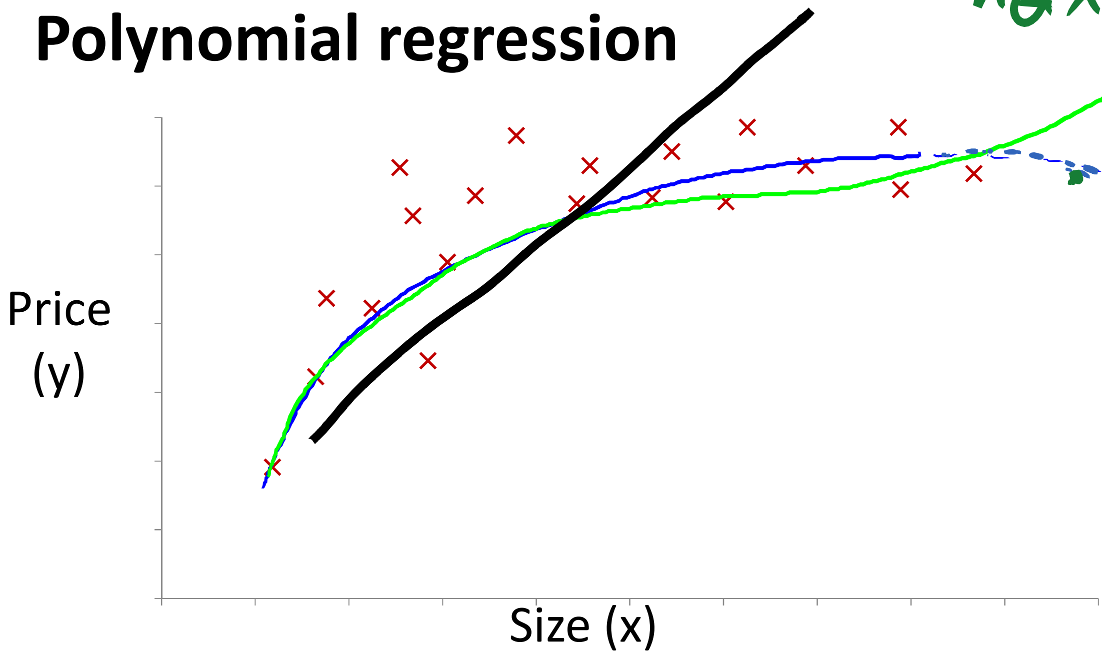


$$h_\theta(x) = \theta_0 + \theta_1(size)^1 + \theta_2(size)^2 + \theta_3(size)^3$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

Polynomial regression

$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 \qquad x_1 = x \qquad x_2 = x^2$



$$\theta_0 + \theta_1 x + \theta_2 x^2$$

$$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$$

$$
\begin{aligned}
h_\theta(x) &= \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \\
&= \theta_0 + \theta_1 (size) + \theta_2 (size)^2 + \theta_3 (size)^3
\end{aligned}
$$

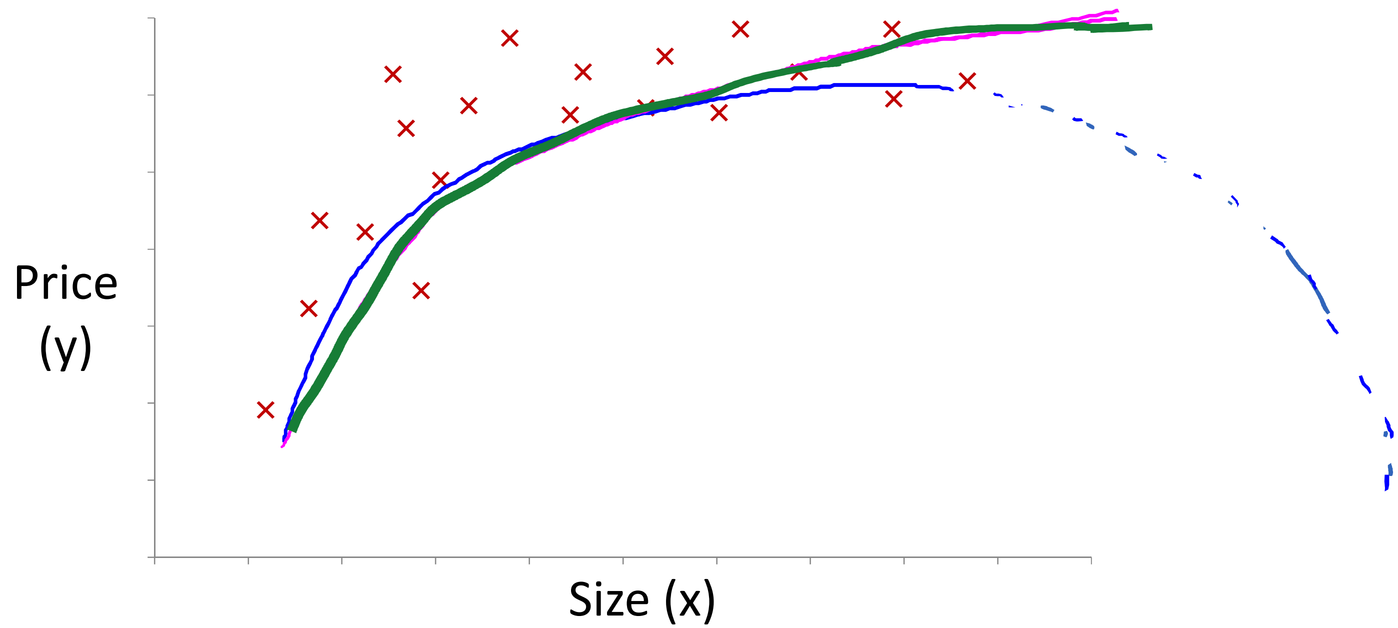
$$x_1 = (size)$$
$$x_2 = (size)^2$$
$$x_3 = (size)^3$$

Size: 1 - 1000

$size^2$: 1 - 1,000,000

$Size^3$: 1 - 10^9

Choice of features

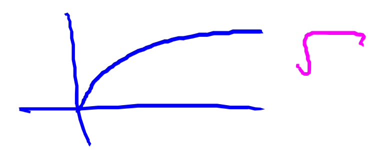


x: size

$$h_\theta(x) = \theta_0 + \theta_1(size) + \theta_2(size)^2$$

$$h_\theta(x) = \theta_0 + \theta_1(\underset{\downarrow\ x_1 = x}{size}) + \theta_2\sqrt{(size)}$$

$x_2 = \sqrt{x}$



Stochastic Gradient Descent vs Batch Gradient Descent

<u>Batch Gradient Descent</u>

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

$$\theta_{t+1} = \theta_t - \alpha J'(\theta)$$

Need to compute for all data,
if you have sample size, m = 1million
Very slow to update θ

<u>Stochastic Gradient Descent</u>

$$cost\left(\theta, \left(x^{(i)}, y^{(i)}\right)\right) = \frac{1}{2}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

1. Random Shuffle Data
2. Repeat {

   for i = 1, … , m {

$$\theta_{t+1} = \theta_t - \alpha \frac{\partial}{\partial \theta_t} cost\left(\theta, \left(x^{(i)}, y^{(i)}\right)\right)$$

   }

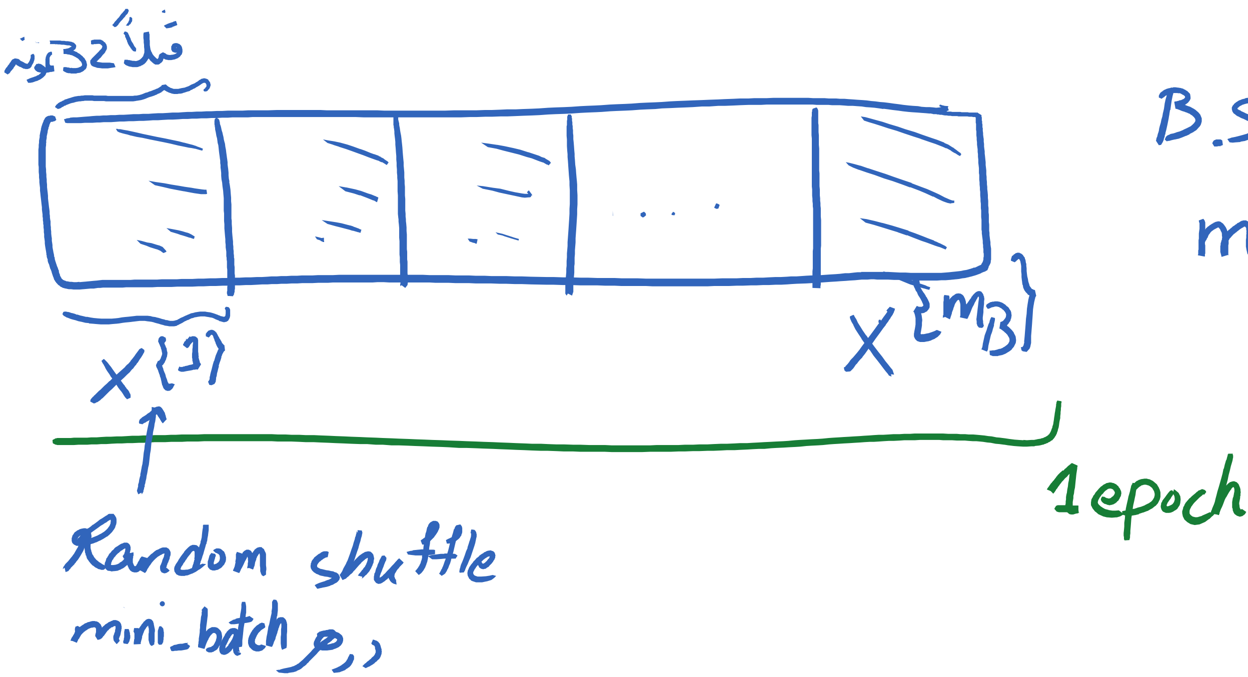
}

Update θ using only one data point.
Faster

ممکن است حول θ_{opt} نوسان کند ← α کاهنده با زمان

Mini-batch Gradient Descent

❖ اگر تعداد نمونه ها **(m)**، خیلی بزرگ باشد، به چندین **mini-batch** تقسیم می شوند و در هر مرحله یکی از **mini-batch**ها ر به هنگام رسانی پارامترها استفاده می شود.



B.S : 32, 64, 128, . . .

$$m_B = \frac{m}{B.S.}$$

Mini-batch Gradient Descent

$\theta_0 = b$

$[\theta_1 \;\dots\; \theta_n]^T = W$

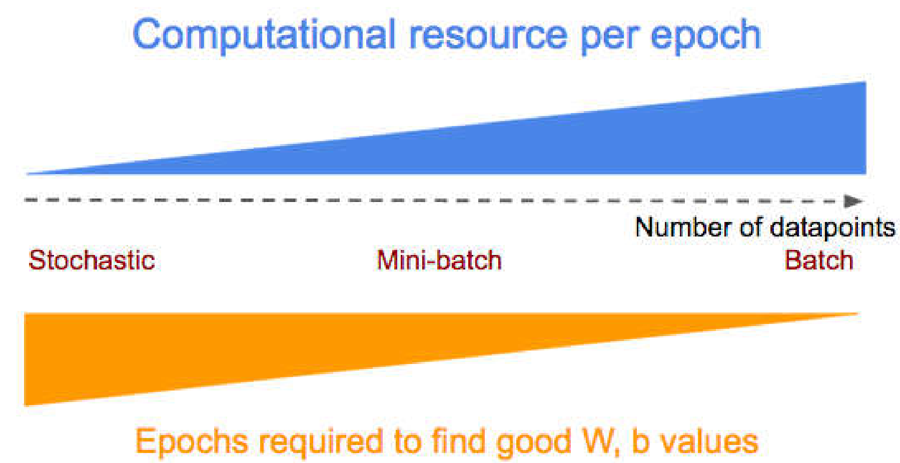
$\frac{m}{t} = m_B$

$$X = \begin{bmatrix} \vdots & \vdots & & \vdots \\ x^{(1)} & x^{(2)} & \cdots & x^{(m)} \\ \vdots & \vdots & & \vdots \end{bmatrix}_{n_x \times m}$$

$t = 1 \rightarrow Stochastic$

$t = m \rightarrow Batch$

$t = 64, 128, 256, 512 \;\rightarrow Mini - batch$



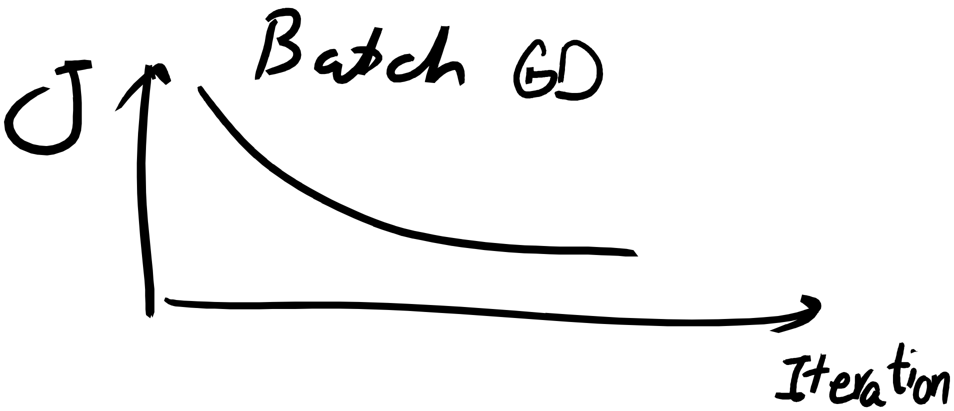
$for\ every\ t$:

Optimize based on $X^{(1:t)}$

$Compute\ cost\ J$

$Update\ \theta$

$$\Big|\; \underbrace{x^{(1)} \cdots x^{(64)}}_{X^{\{1\}}} \;\Big|\; \underbrace{x^{(65)} \cdots x^{(128)}}_{X^{\{2\}}} \;\Big| \cdots$$



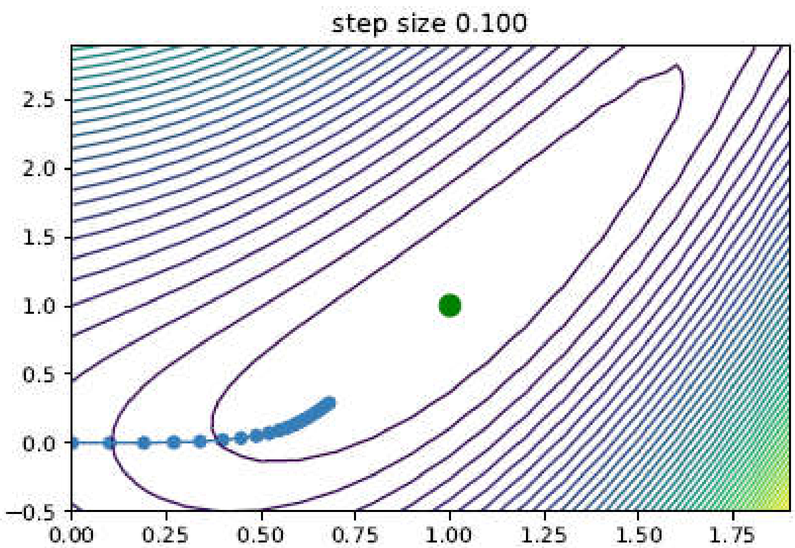
* ممکن است یادگیری $X^{\{2\}}$ از $X^{\{1\}}$ مشکل‌تر باشد $\Leftarrow$ $J^{\{2\}} > J^{\{1\}}$

$1 < m_B < m$

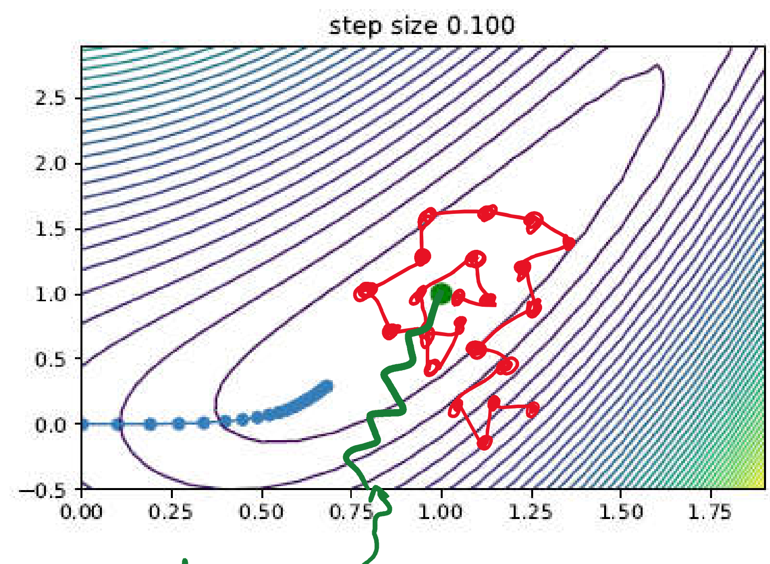
$m_B = m \rightarrow$ stochastic G.D.

$m_B = 1 \rightarrow$ Batch G.D.

Batch Gradient Descent



Stochastic Gradient Descent



$1 < m_B < m$

Fastest learning

Batch Gradient Descent

Too long per iteration especially when *m* is very large

Stochastic Gradient Descent

Lose speedup from vectorization (Inefficient implementation)

Newton Method

ریشه یابی : ریشه $f(x)$

$x_{init.}$

$$x_{t+1} = x_t - \frac{f(x_t)}{f'(x_t)}$$

$J(\theta)$ $\quad J'(\theta) = 0$

θ_{init} $\quad \theta_{t+1} = \theta_t - \frac{J'(\theta_t)}{J''(\theta_t)}$

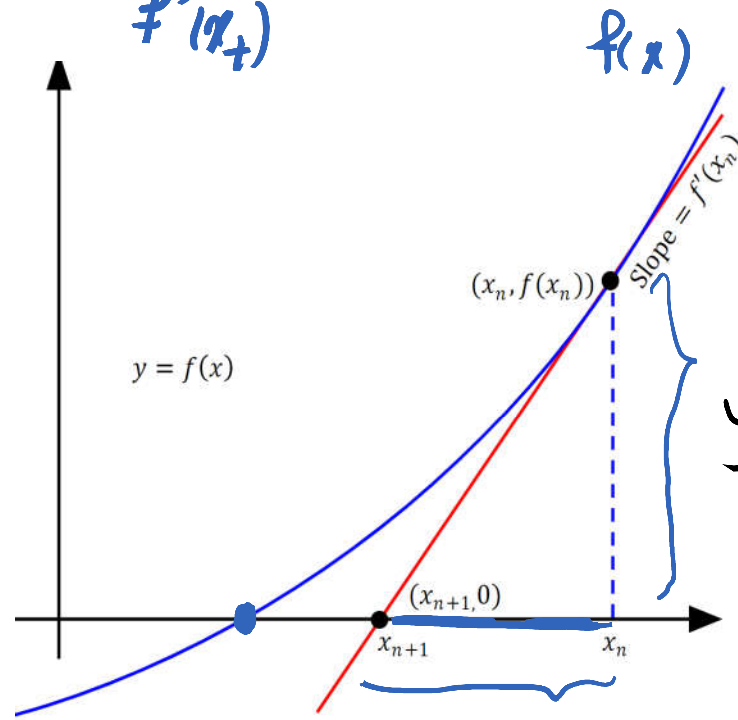
نیوتن تک متغیره

$$\underline{\theta} = [\theta_0 \; \cdots \; \theta_n]^T_{n+1 \times 1}$$

$$J(\underline{\theta}) \quad \nabla_\theta J(\underline{\theta}_{opt}) = \left[\frac{\partial J}{\partial \theta_0}, \cdots, \frac{\partial J}{\partial \theta_n}\right]^T$$

$\underline{\theta}_{init}$ $\quad \underline{\theta} := \underline{\theta} - \underline{H}^{-1} \nabla_\theta J(\theta)$

نیوتن رافسون



$$H_{(n+1)(n+1)} : H_{ij} = \frac{\partial J}{\partial \theta_i \partial \theta_j}$$

$$\begin{bmatrix} \frac{\partial^2 J}{(\partial \theta_0)^2} & \cdots & \\ & \ddots & \\ & & \frac{\partial^2 J}{(\partial \theta_n)^2} \end{bmatrix}$$

- Faster convergence

Newton Method

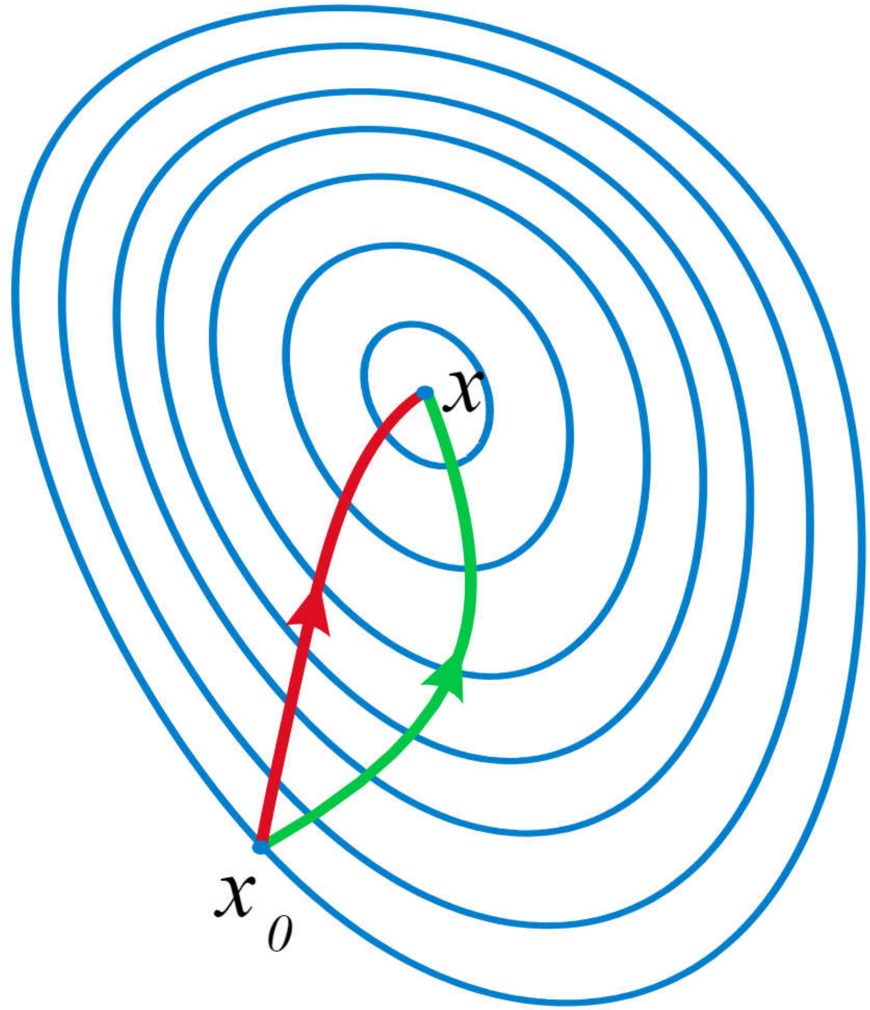
سرعت همگرایی نیوتن - رافسون می‌تواند بیشتر از D (دو) باشد.

ولی $H^{-1}_{(n+1)(n+1)}$ برای n بزرگ هزینه محاسباتی زیادی دارد

وقتی روش نیوتن رافسون برای حل مسئله log. Reg به کار می‌رود ←

Fisher Scoring

Newton Method vs Gradient Descent



Logistic Regression

Introduction

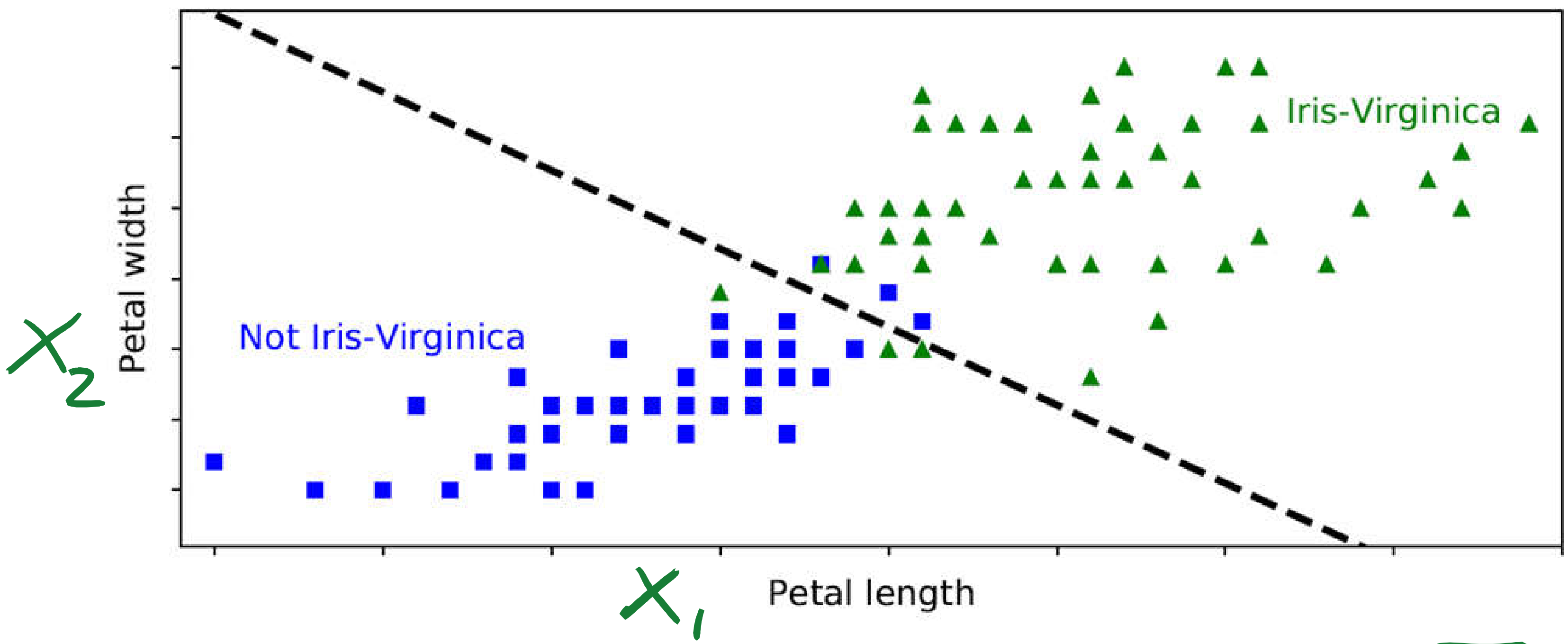
- Logistic regression is a widely used discriminative classification model $p(y \mid \mathbf{x}; \boldsymbol{\theta})$.
- where $\mathbf{x} \in \mathbb{R}^D$ is a fixed-dimensional input vector, $y \in \{1, \dots, C\}$ is the class label.
- If $C = 2$, this is known as binary logistic regression.
- if $C > 2$, it is known as multinomial logistic regression, or alternatively, multiclass logistic regression.

Classification

Email: Spam / Not Spam?
Online Transactions: Fraudulent (Yes / No)?
Tumor: Malignant / Benign ?

$$y \in \{0, 1\}$$

0: "Negative Class" (e.g., benign tumor)
1: "Positive Class" (e.g., malignant tumor)



Threshold classifier output $h_\theta(x)$ at 0.5:

$h_\theta(x) = \theta^T x$

$-\infty \to \infty$

If $h_\theta(x) \geq 0.5$, predict "y = 1"

If $h_\theta(x) < 0.5$, predict "y = 0"

Classification: y = 0 or 1

$h_\theta(X) = \theta^T X$

$h_\theta(x)$ can be > 1 or < 0

Logistic Regression: $\boxed{0 \leq h_\theta(x) \leq 1}$

$h_\theta(X) = P(y=1 \mid \underline{X})$

طبقه‌بندی باینری

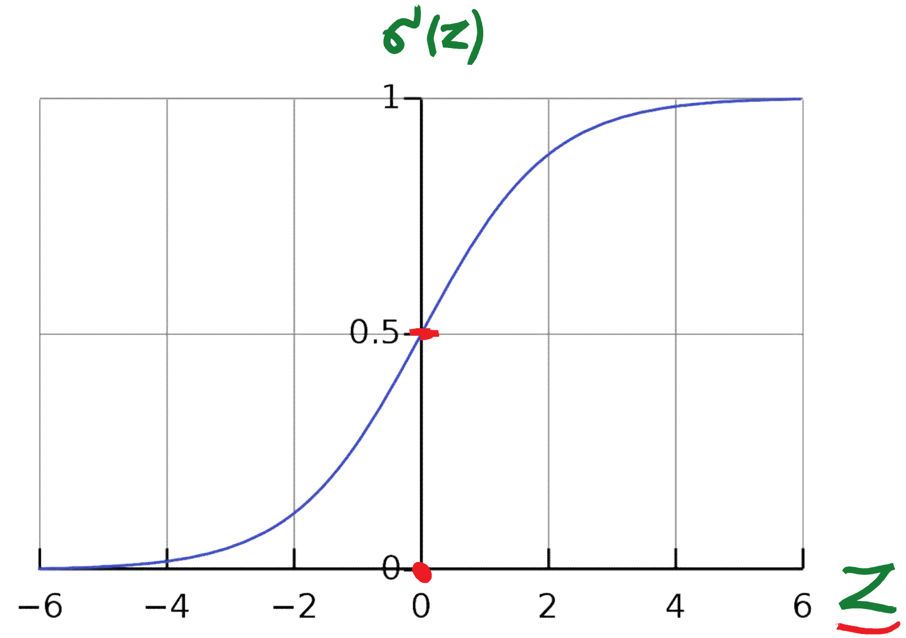
Logistic Regression Model

We want $0 \leq h_\theta(x) \leq 1$

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

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

$$\boxed{H_\theta(X) = \frac{1}{1+e^{-\theta^T X}}}$$



Interpretation of Hypothesis Output

$\{x^{(i)}, y^{(i)}\}_{i=1}^{m}$ $\quad y^{(i)} \in \{0, 1\}$

1 → بدخیم

$h_\theta(x)$ = estimated probability that y = 1 conditioned on input x

Example: If $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumorSize} \end{bmatrix}$

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

$x^{(i)} \longrightarrow y^{(i)}$

$h_\theta(x_{test}) = 0.7 \longrightarrow$ بدخیم

$h_\theta(x) = P(y=1 \mid x)$

$1 - h_\theta(x) = P(y=0 \mid x)$

$$P(y=0|x;\theta) + P(y=1|x;\theta) = 1$$
$$P(y=0|x;\theta) = 1 - P(y=1|x;\theta)$$

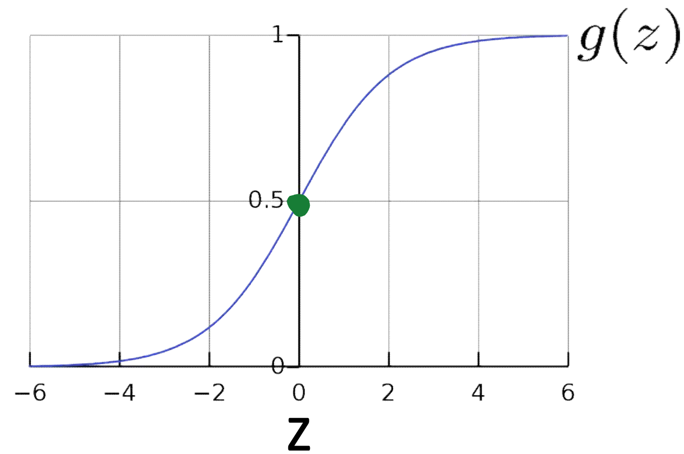
$h_\theta(x_{test}) = 0.3 \longrightarrow$ خوش خیم

$x_{test} \longrightarrow \dfrac{1}{1+e^{-\theta^T x_{test}}}$ $>$ th بدخیم, < 0.5 خوش خیم

Logistic regression

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

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



Suppose predict "$y = 1$" if $h_\theta(x) \geq 0.5$

$\theta^T x = 0 \Rightarrow g(\theta^T x) = \frac{1}{2}$

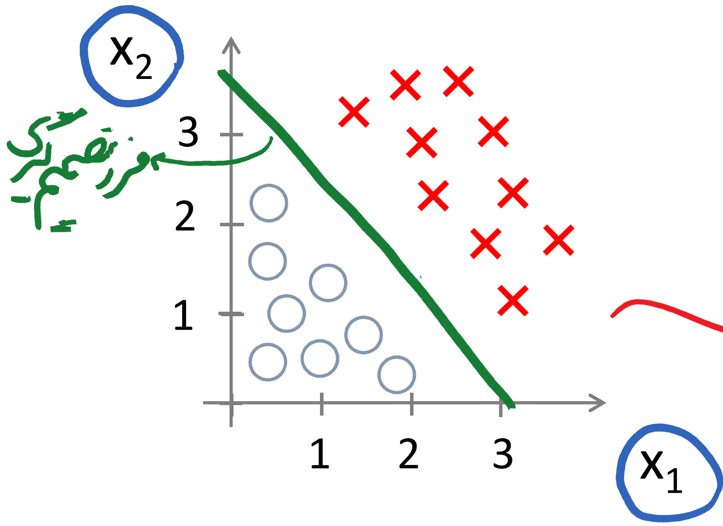
predict "$y = 0$" if $h_\theta(x) < 0.5$

$th = 0.5$: مرز تصمیم گیری

$\theta^T x \geq 0 \rightarrow y = 1$

$\theta^T x < 0 \rightarrow y = 0$

Decision Boundary



$$\theta^T X = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2$$

$x_0 = 1$

$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

$(x^{(i)}, y^{(i)})$ $\Rightarrow$ $\theta_0 = -3$ $\quad \theta_1 = \theta_2 = 1$

$i = 1 \ldots m$

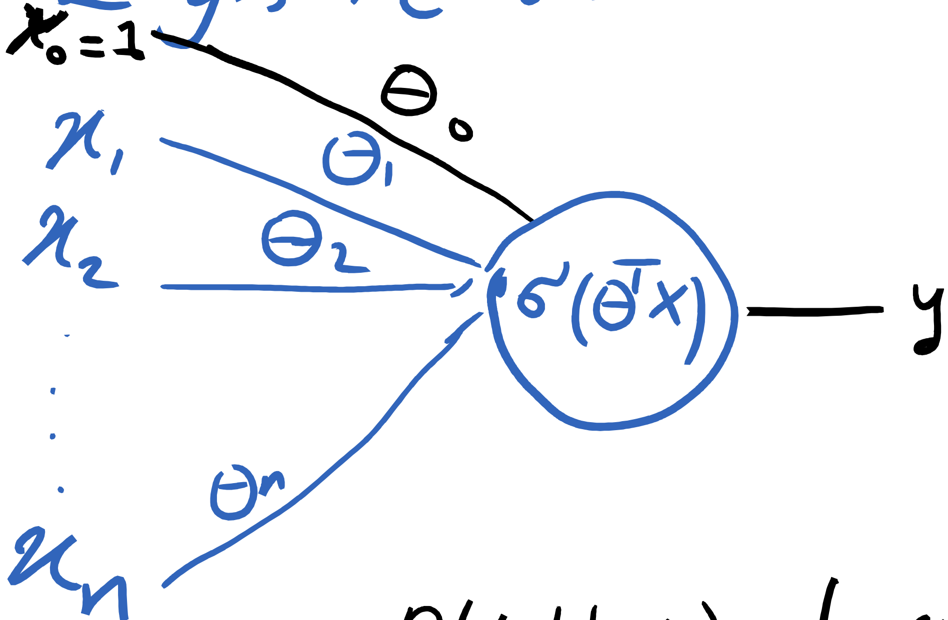
Predict "$y = 1$" if $-3 + x_1 + x_2 \geq 0$ $\longrightarrow$ $y = 1$

مرز تصمیم گیری

$\theta^T X \geq 0$ $\quad < 0$ $\longrightarrow$ $y = 0$

$th = 0.5$

Logistic Unit:



$$0 \leq h_\theta(x) \leq 1$$

$$P_\theta(y=1 \mid x) = h_\theta(x) = \frac{1}{1+e^{-\theta^T x}} \gtrless 0.5$$

$$\theta^T x \gtrless 0$$

$\sigma(\theta^T x)$ $\theta_0, \theta_1, \theta_2$

$h_\theta(X) = \sigma(P_\theta(\underline{x}))$

$P_\theta(X) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2$

Non-linear decision boundaries

Transform input features in suitable way

$$\phi(x_1, x_2) = [1, x_1^2, x_2^2]$$ $x_1 x_2, x_1, x_2$

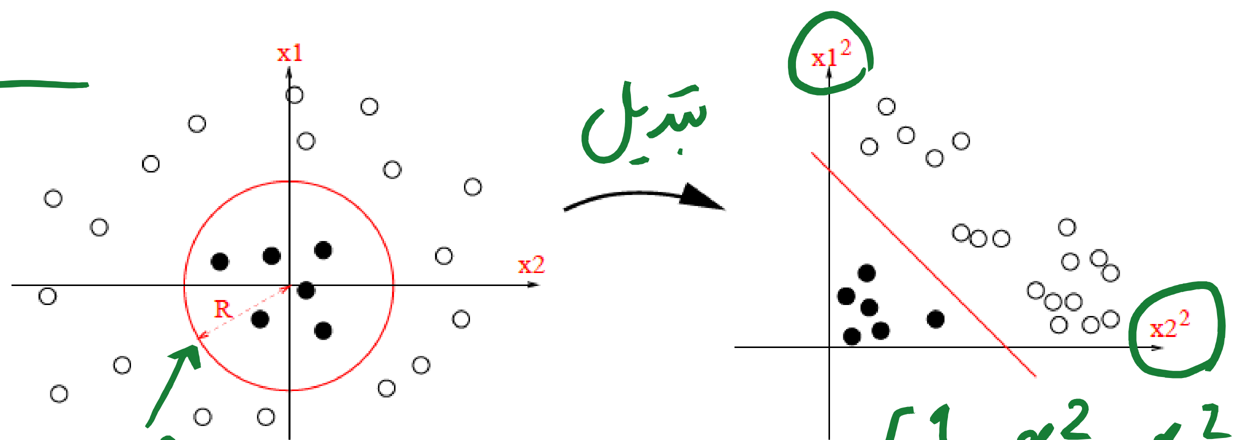
$$\boldsymbol{w} = [-R^2, 1, 1]. \text{ Then } \boldsymbol{w}^\mathsf{T}\phi(\boldsymbol{x}) = x_1^2 + x_2^2 - R^2$$

$> 0 \rightarrow y = 1$

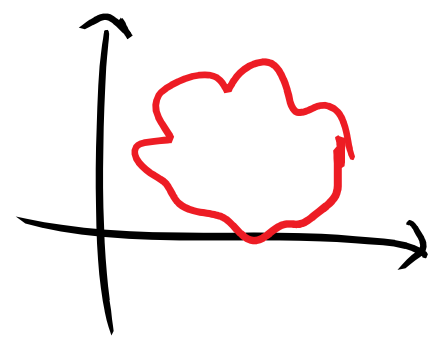
$< 0 \rightarrow y = 0$

Decision boundary (where f(x) = 0) defines a circle with radius R

با مرز تصمیم گیری خطی تفکیک پذیر نیست.



$[1, x_1, x_2]$ $[1, x_1^2, x_2^2]$



Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$

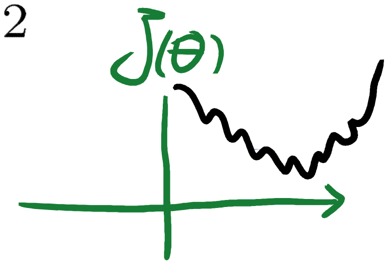
m examples $\quad x \in \begin{bmatrix} x_0 \\ x_1 \\ \cdots \\ x_n \end{bmatrix} \quad x_0 = 1, y \in \{0, 1\}$

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

How to choose parameters θ ?

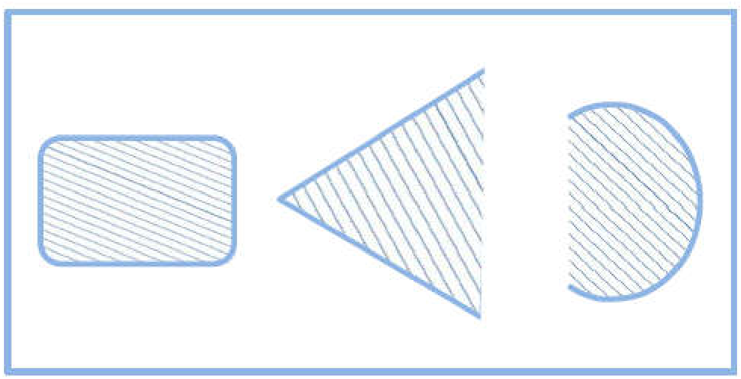
Cost function

Linear regression: $J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \frac{1}{2}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$

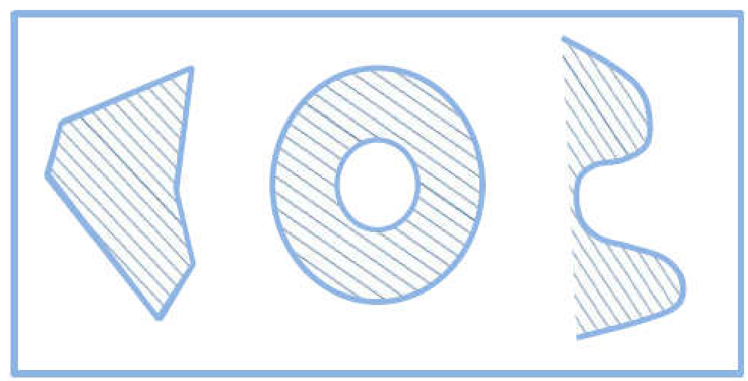


$\sigma(\theta^T x^{(i)})$

$\mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)}) = \frac{1}{2}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$ → غير محدب



Convex



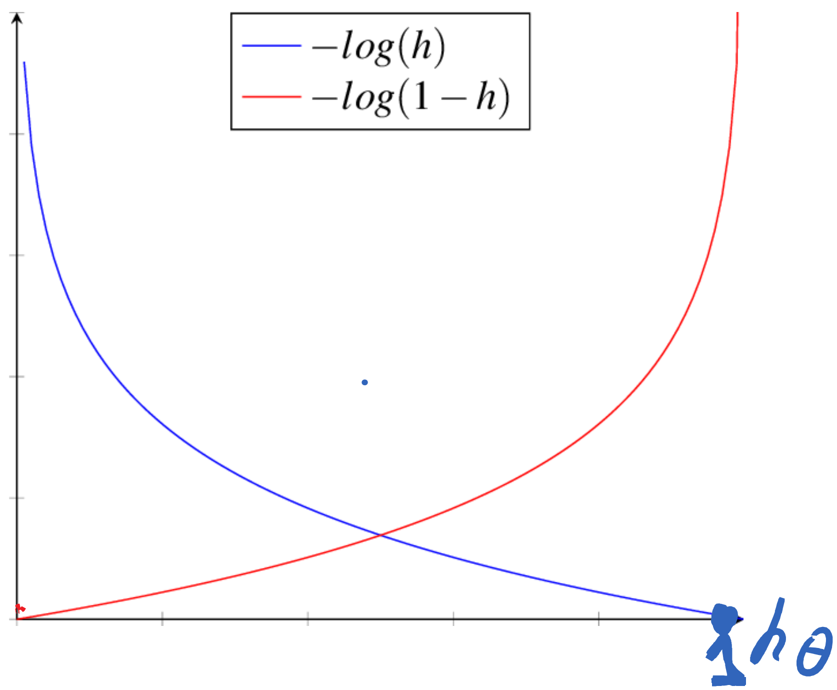
Not Convex

$h_\theta(x) = P(y=1 \mid x)$

Logistic regression cost function

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

$\boxed{\text{convex}}$



Logistic regression cost function

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

Note: $y = 0$ or 1 always

$$\boxed{\mathrm{Cost}(h_\theta(x), y) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x))}$$

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

Logistic regression cost function

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$= -\frac{1}{m} [\sum_{i=1}^{m} y^{(i)} \log \underbrace{h_\theta(x^{(i)})}_{p(y=1|x)} + (1 - y^{(i)}) \log (\underbrace{1 - h_\theta(x^{(i)})}_{p(y=0|x)})]$$

Gradient Descent

$\underline{\theta}$

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

Want $\min_\theta J(\theta)$:

Repeat $\{$

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

$\}$

(simultaneously update all θ_j)

$j = 0 \ldots n$

$$\frac{\partial}{\partial \theta_j} \sigma(\theta^T x)?$$

$$\frac{\partial}{\partial z} \sigma(z) = \sigma(z)[1 - z]$$

True fact about sigmoid functions

$$\frac{\partial}{\partial \theta_j} \sigma(\theta^T x) = \frac{\partial}{\partial z} \sigma(z) \cdot \frac{\partial z}{\partial \theta_j}$$

Chain rule!

$$\frac{\partial}{\partial \theta_j} \sigma(\theta^T x) = \sigma(\theta^T x)[1 - \sigma(\theta^T x)] x_j$$

Plug and chug

Gradient Descent

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

Want $\min_\theta J(\theta)$: $\underline{\theta}_{init}$

Repeat $\{$

$$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (\underbrace{h_\theta(x^{(i)})}_{= \sigma(\theta^T x)} - y^{(i)}) x_j^{(i)}$$

(simultaneously update all θ_j)

$j = 0 \dots n$

$\}$

$$\underline{\theta} = \begin{bmatrix} \theta_0 \\ \vdots \\ \theta_n \end{bmatrix}$$

Algorithm looks identical to linear regression!

logistic regression (Probabilistic view)

$$\underline{w}^T \underline{x} + b \equiv \theta^T x$$

$$\underbrace{\theta_0}_{b} + \underbrace{\theta_1 x_1 + \dots + \theta_n x_n}_{\underline{w}^T \underline{x}}$$

Binary logistic regression often follows the following model

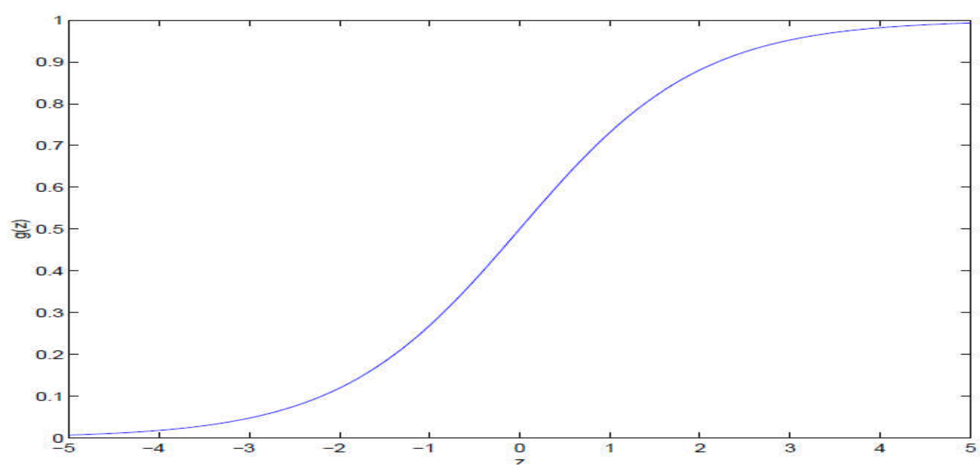
$$p(y|x;\theta) = \mathrm{Ber}(y|\sigma(\underline{w}^{\top}\underline{x} + b))$$

Bernoulli sigmoid weight bias

Bernoulli: $p = \sigma(\theta^T x)$

$$f(k;p) = \begin{cases} p & \text{if } k = 1, \\ q = 1 - p & \text{if } k = 0. \end{cases}$$

$$p(y = 1|x;\theta) = \sigma(a) = \frac{1}{1 + e^{-a}}, \text{ , where } a = \log \frac{p}{1-p}$$



logistic regression (Probabilistic view)

Finding the Maximum Likelihood (ML) solution is equivalent to minimizing the cross entropy cost function

$P(y \mid X; \theta)$ $P_\theta(y \mid X)$

$\max_\theta \log P_\theta(y \mid X) \equiv \min_\theta J(\theta)$

log-likelihood

Cross entropy

$P_\theta(y=1 \mid X) = h_\theta(X) = \sigma(\theta^T X)$

$P_\theta(y=0 \mid X) = 1 - h_\theta(X) = 1 - \sigma(\theta^T X)$

log-likelihood: برای نمونه i ام $y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)}))$

$$P_\theta(y^{(i)} | x^{(i)}) = (h_\theta(x^{(i)}))^{y^{(i)}} + (1-h_\theta(x^{(i)}))^{(1-y^{(i)})}$$

$$\log P_\theta(y^{(i)} | x^{(i)}) = \boxed{y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)}))}$$

$i = 1 \dots m$

$$P(\underline{y} | \underline{X}) = \prod_{i=1}^{m} P_\theta(y^{(i)} | x^{(i)})$$

$$\max \log P(\underline{y} | \underline{X}) = \boxed{\max \sum_{i=1}^{m} \log P_\theta(y^{(i)} | x^{(i)})}$$

Multiclass classification

Email foldering/tagging: Work, Friends, Family, Hobby

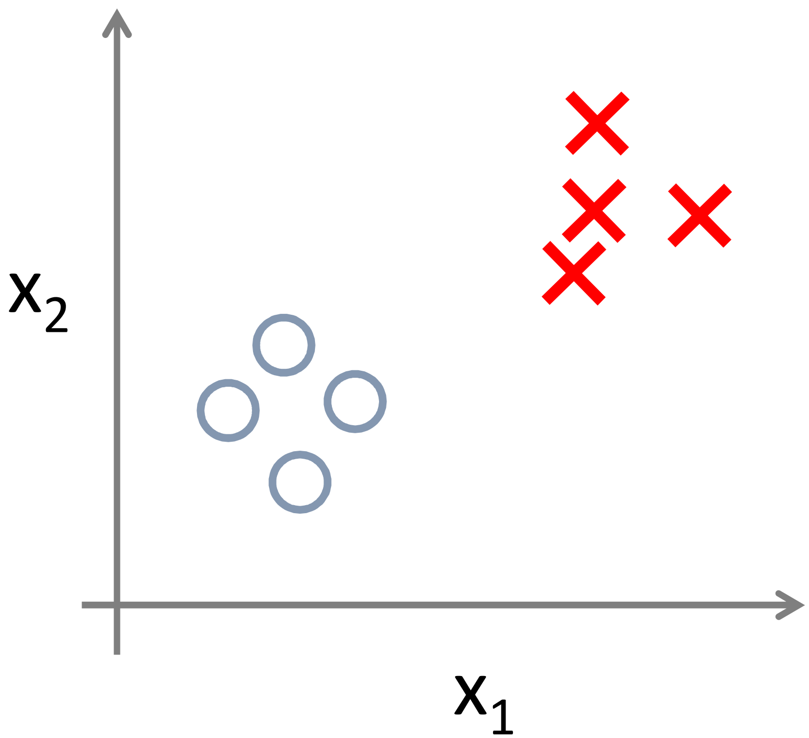
Medical diagrams: Not ill, Cold, Flu

Weather: Sunny, Cloudy, Rain, Snow

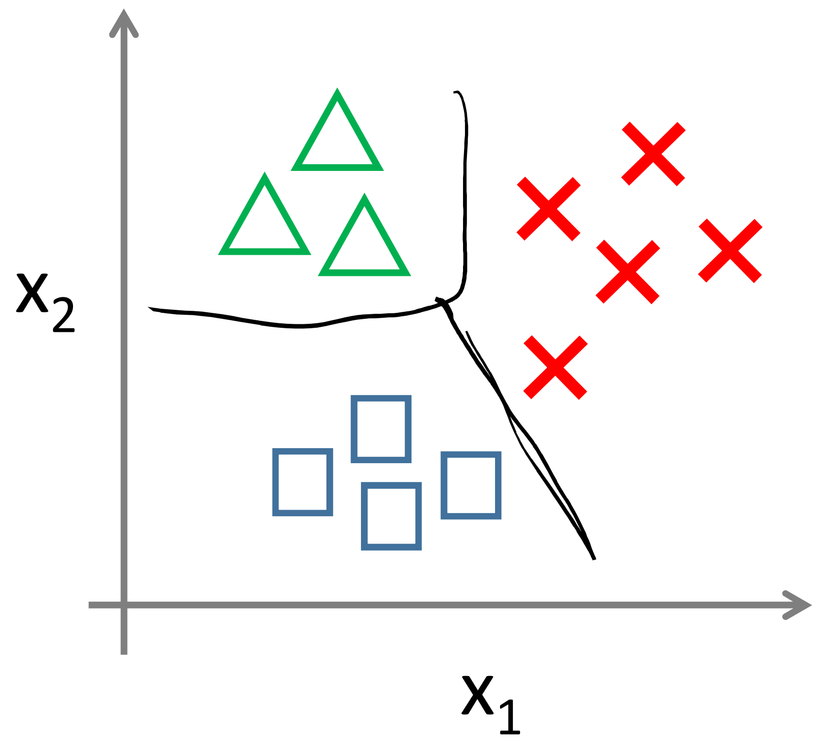
$y \in \{0, 1, 2, 3\}$

$x_{test} \rightsquigarrow$ 2 کلاس

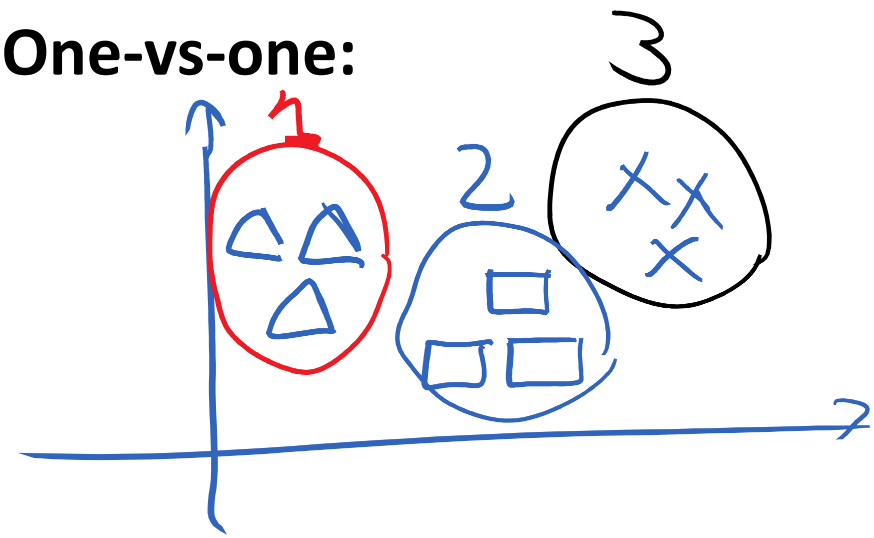
Binary classification:



Multi-class classification:



One-vs-one:



1 - 2

2 - 3

1 - 3

$k = 3$

$\binom{k}{2}$ تا مدل یاد بگیرد

$x_{test} \longrightarrow \binom{k}{2}$ خروجی

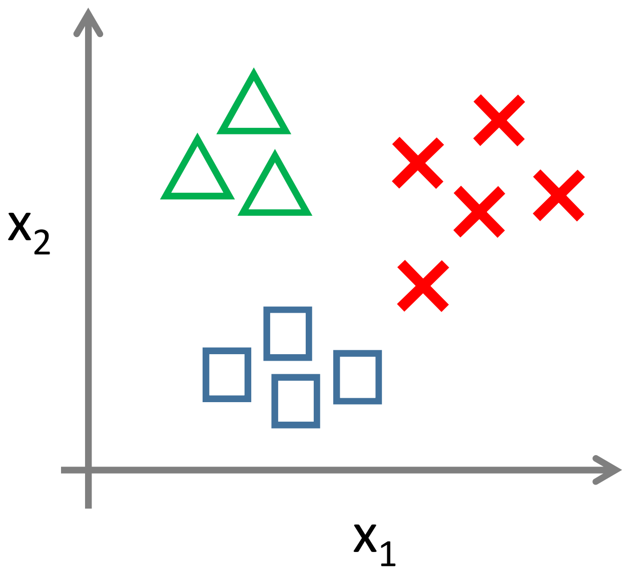
$\rightarrow 1$

$\rightarrow 1$

$\rightarrow 3$

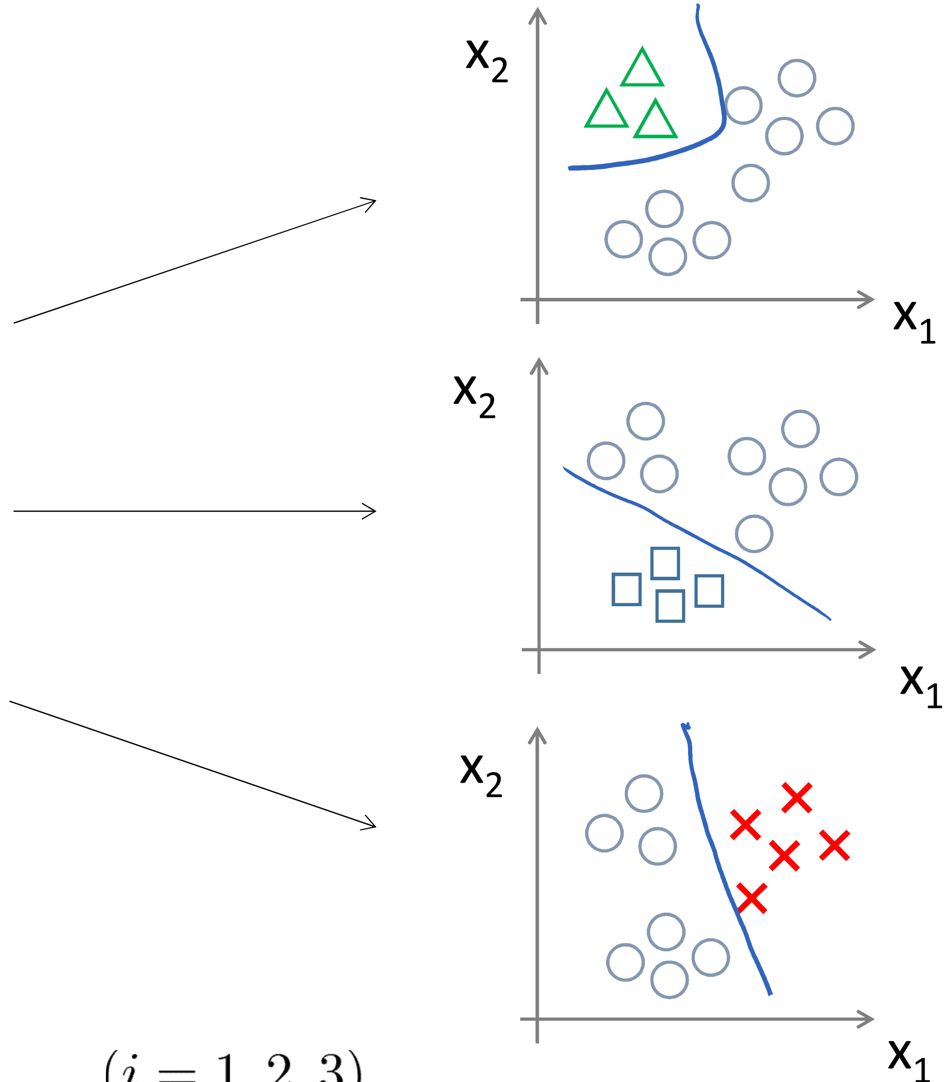
$\Longrightarrow y_{test} = 1$

One-vs-all:



Class 1:
Class 2:
Class 3:

$$h_\theta^{(i)}(x) = P(y = i|x;\theta) \qquad (i = 1, 2, 3)$$



softmax regression:

تعمیم یافته logistic regression برای k > 2 کلاس

$(x^{(i)}, y^{(i)})$ $i = 1..m$ $y^{(i)} \in \{1, 2, .., k\}$

$\boxed{\text{MNIST } K = 10}$

$p(y = k \mid x) \Longrightarrow h_\theta(X) = \begin{bmatrix} p(y=1 \mid x; \theta) \\ p(y=2 \mid x; \theta) \\ p(y=k \mid x; \theta) \end{bmatrix} =$

$$h_\theta^{(j)}(X) = \frac{e^{\theta^{(j)T} X}}{\sum_{i=1}^{k} e^{\theta^{(i)T} X}}$$

$$\sum_{j=1}^{k} h_\theta^{(j)}(X)$$

$\underline{\theta} = [\theta^{(1)}, \ldots, \theta^{(k)}]_{(n+1) \times k}$: پارامتر θ : $\boxed{\underline{\theta}^{(k)} \text{ و } \underline{\theta}^{(1)}_{(n+1) \times 1}}$

Cross entropy cost function:

1(یک عبارت درست) $= 1$ $\qquad$ 1(عبارت نادرست) $= 0$

$$J(\theta) = -\left[\sum_{i=1}^{m} \sum_{k=1}^{K} 1\{y^{(i)}=k\} \log \frac{\exp(\theta^{(k)T} \underline{x}^{(i)})}{\sum_{j=1}^{k} \exp(\theta^{(j)T} \underline{x}^{(i)})} \right]$$

تعمیم یافته تابع هزینه log Reg.

$K=4$

$$X_{test} \longrightarrow h_\theta(\underline{X_{test}}) = \begin{bmatrix} 0.8 \\ 0.1 \\ 0/05 \\ 0/05 \end{bmatrix} \longrightarrow y_{test} = 1$$

$$\underline{P(y \mid x, \theta)} \sim \text{Categorical}(P_1, P_2, \dots, P_k) \qquad P_k = 1 - \sum_{i=1}^{k-1} P_i$$

Regularization

Regularization

A fundamental problem is that the algorithm tries to pick parameters that minimize loss on the training set, but this may not result in a model that has low loss on future data. This is called **overfitting**.

example

suppose we want to predict the probability of heads when tossing a coin. We toss it $N = 3$ times and observe 3 heads. The MLE is $\boxed{\theta_{\text{mle}} = N_1/(N_0 + N1) = 3/(3+0) = 1}$. However, if we use $\text{Ber}\left(y \mid \theta_{\text{mle}}\right)$ as our model, we will predict that all future coin tosses will also be heads, which seems rather unlikely.

Regularization

- The core of the problem is that the model has enough parameters to perfectly fit the observed training data, so it can **perfectly match** the empirical distribution.
- However, in most cases the empirical distribution is not the same as the true distribution, so putting all the probability mass on the observed set of N examples will not leave over any probability for novel data in the future. That is, **the model may not generalize**.

Solution

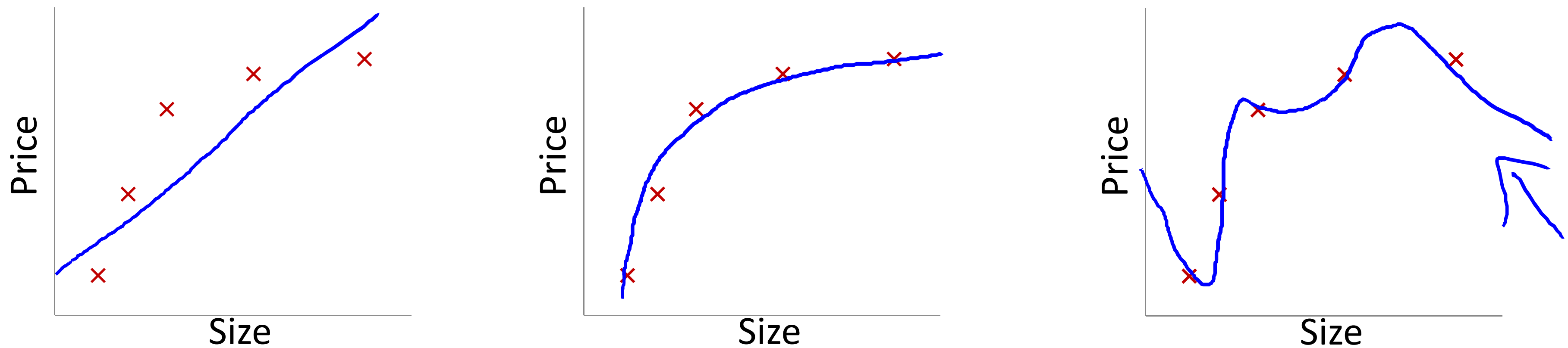
The main solution to overfitting is to use regularization, which means to add a penalty term to the Cost function. Thus we optimize an objective of the form

$$\mathcal{L}(\boldsymbol{\theta};\lambda) = \left[\frac{1}{M}\sum_{i=1} \ell(\mathbf{y}_i,\boldsymbol{\theta};\mathbf{x}_i)\right] + \lambda C(\boldsymbol{\theta})$$

$\lambda \geq 0$ is a tuning parameter and control the relative impact of these two terms on the regression coefficient estimates.

When $\lambda = 0$, the penalty term has no effect
However, as $\lambda \to \infty$, the impact of the shrinkage penalty grows, and the ridge regression coefficient estimates will approach zero.

Example: Linear regression (housing prices)



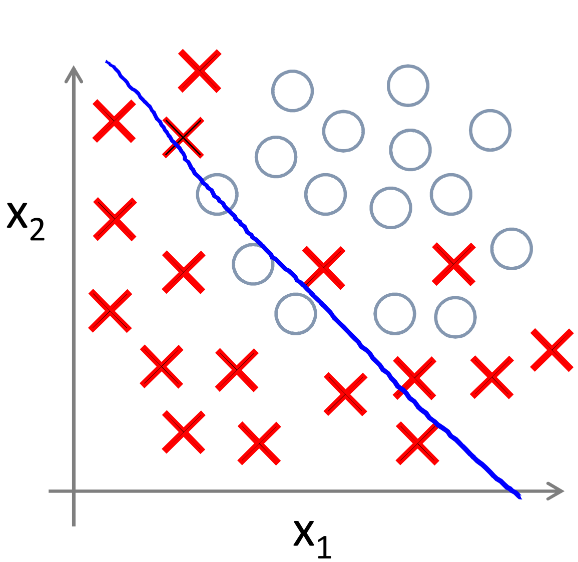
→ $\theta_0 + \theta_1 x$ → $\theta_0 + \theta_1 x + \theta_2 x^2$ → $\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

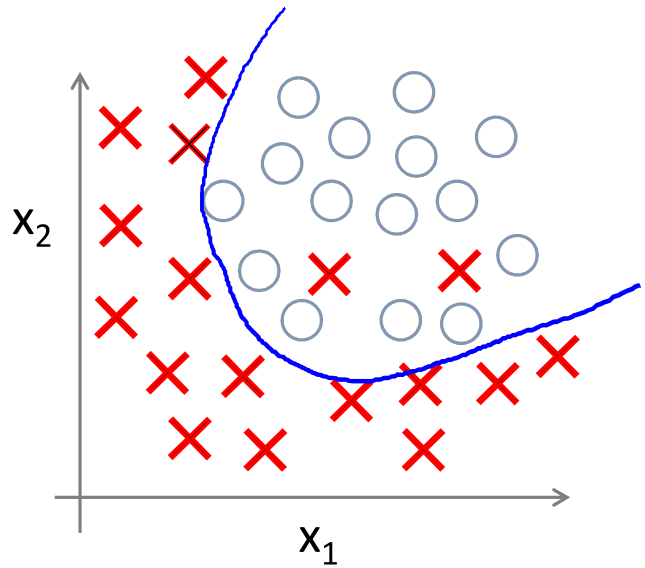
"Underfit" "High bias" $\lambda \uparrow$ "Just right" λ "Overfit" "High variance" $\lambda = 0$

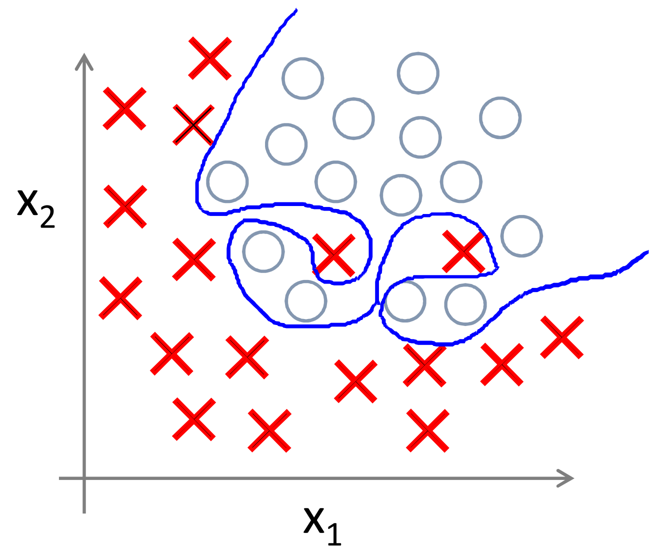
Overfitting: If we have too many features, the learned hypothesis may fit the training set very well ($J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 \approx 0$), but fail to generalize to new examples (predict prices on new examples).

Example: Logistic regression

(طبقه‌بندی)







$h_\theta(x) = g(\underline{\theta_0 + \theta_1 x_1 + \theta_2 x_2})$

(g = sigmoid function)

"Underfit"

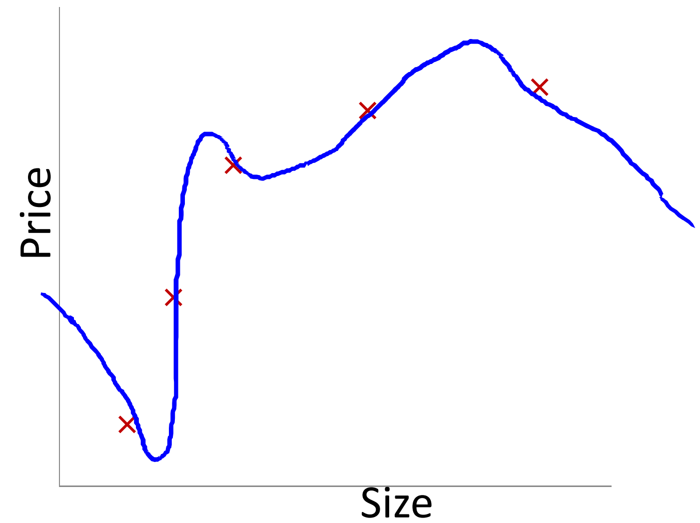
$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 \underline{x_1^2 x_2} + \theta_4 \underline{x_1^2 x_2^2} + \theta_5 \underline{x_1^2 x_2^3} + \theta_6 \underline{x_1^3 x_2} + \ldots)$

"Overfit"

Addressing overfitting:

$$x_1 = \text{ size of house}$$
$$x_2 = \text{ no. of bedrooms}$$
$$x_3 = \text{ no. of floors}$$
$$x_4 = \text{ age of house}$$
$$x_5 = \text{ average income in neighborhood}$$
$$x_6 = \text{ kitchen size}$$
$$\vdots$$
$$x_{100}$$



Addressing overfitting:

Options:

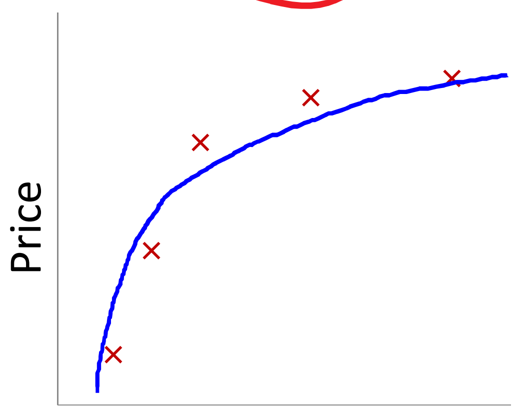
1. Reduce number of features
   - Manually select which features to keep.
   - Model selection algorithm (later in course).
2. Regularization
   - Keep all the features, but reduce magnitude/values of parameters θ_j.
   - Works well when we have a lot of features, each of which contributes a bit to predicting y.

Regularization

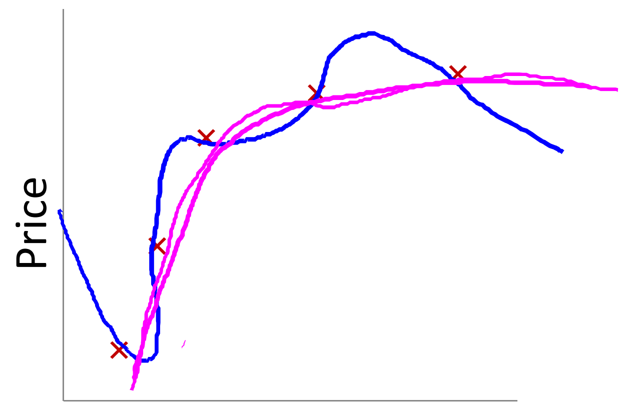
Cost function

Intuition

Bias - Variance tradeoff



$\theta_0 + \theta_1 x + \theta_2 x^2$



$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

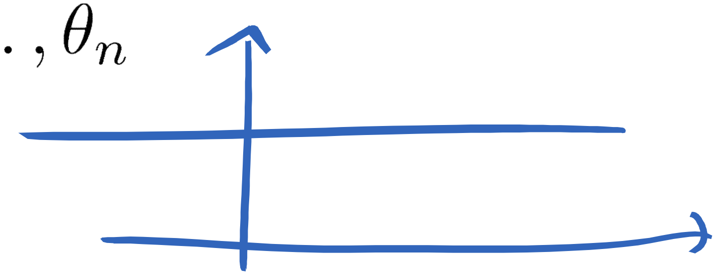
Suppose we penalize and make θ_3, θ_4 really small.

$$\min_\theta \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000\,\theta_3^2 + 1000\,\theta_4^2$$

$\theta_3 \approx 0 \qquad \theta_4 \approx 0$

Regularization.

$y = \theta_0 \leftarrow \theta_1 = \theta_2 = \ldots \theta_{100} = 0$



Small values for parameters $\theta_0, \theta_1, \ldots, \theta_n$

- "Simpler" hypothesis
- Less prone to overfitting

Housing:

- Features: $x_1, x_2, \ldots, x_{100}$
- Parameters: $\theta_0, \theta_1, \theta_2, \ldots, \theta_{100}$

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 + \lambda \sum_{j=1}^{n} \theta_j^2\right]$$

$\cancel{\theta_0}$ $\theta_1, \theta_2, \theta_3, \ldots, \theta_{100}$

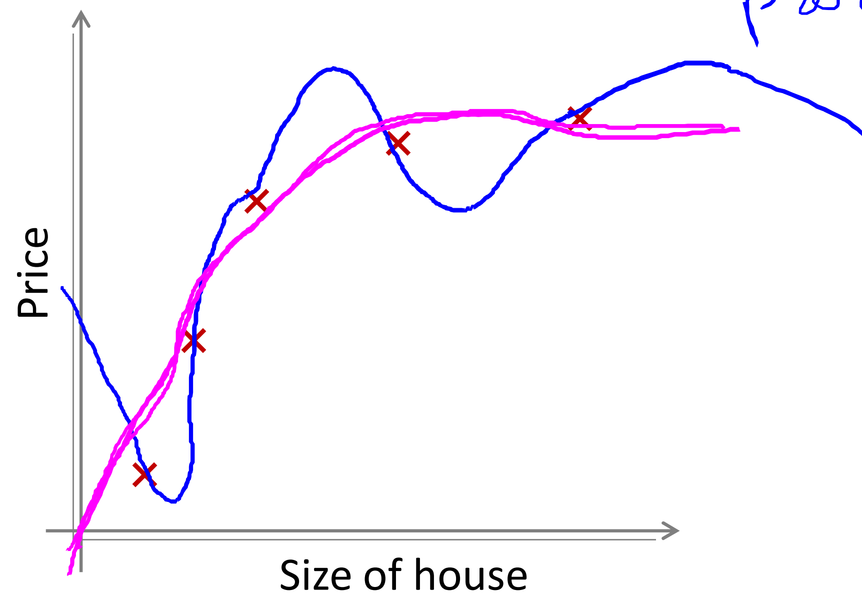
$\cancel{\theta_0}$

Regularization.

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n}\theta_j^2\right]$$

regularization parameter

$$\min_\theta J(\theta)$$



$L1\_norm: \lambda \cdot \sum_{j=1}^{n} |\theta_j|$

In regularized linear regression, we choose θ to minimize

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right]$$

L_2-norm regularization

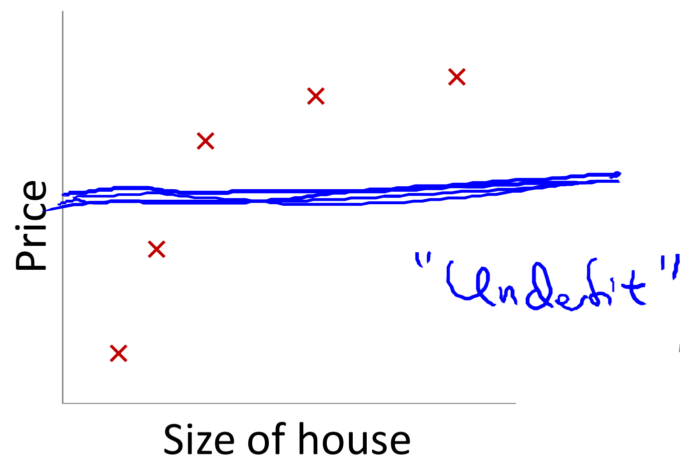
What if λ is set to an extremely large value (perhaps for too large for our problem, say $\lambda = 10^{10}$)?

- Algorithm works fine; setting λ to be very large can't hurt it
- Algortihm fails to eliminate overfitting.
- Algorithm results in underfitting. (Fails to fit even training data well).
- Gradient descent will fail to converge.

In regularized linear regression, we choose θ to minimize

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n}\theta_j^2\right]$$

What if λ is set to an extremely large value (perhaps for too large for our problem, say $\lambda = 10^{10}$)?



$\theta_1, \theta_2, \theta_3, \theta_4$

$\theta_1 \approx 0, \theta_2 \approx 0$

$\theta_3 \approx 0, \theta_4 \approx 0$

$\boxed{h_\theta(x) = \theta_0}$

$h_\theta(x)$

$$\theta_0 + \cancel{\theta_1 x} + \cancel{\theta_2 x^2} + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$$

Regularization

Regularized linear regression

Regularized linear regression

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} \theta_j^2\right]$$

$$\min_\theta J(\theta)$$

Gradient descent

l_2 - norm

Repeat {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$\frac{\partial}{\partial \theta_0} J(\theta)$

$$\theta_j := \theta_j - \alpha \left[\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} - \frac{\lambda}{m} \theta_j \right]$$

$(j = \cancel{0}, 1, 2, 3, \ldots, n)$

}

$$\theta_j := \theta_j (1 - \alpha \frac{\lambda}{m}) - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

$1 - \alpha \frac{\lambda}{m} < 1$ $\qquad$ 0.99 $\qquad$ $\theta_j \times 0.99$

Normal equation(Regularized) (L_2-norm)

$$X = \begin{bmatrix} (x^{(1)})^T \\ \vdots \\ (x^{(m)})^T \end{bmatrix} \qquad y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix} \quad \mathbb{R}^m$$

$m \times (n+1)$

$$\rightarrow \min_\theta J(\theta)$$

$$\frac{\partial}{\partial \theta_j} J(\theta) \stackrel{\text{set}}{=} 0$$

$$\theta = \Big(\boxed{X^T X} + \lambda \underbrace{\begin{bmatrix} 0 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}}_{(n+1)\times(n+1)}\Big)^{-1} X^T y$$

E.g. $n=2$

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Non-invertibility

Suppose $m \leq n$,

(#examples) (#features)

$$\theta = \underbrace{(X^T X)^{-1}}_{\text{non-invertible / singular}} X^T y$$

If $\lambda > 0$,

$$\theta = \underbrace{\left(X^T X + \lambda \begin{bmatrix} 0 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix} \right)}_{\text{invertible}}^{-1} X^T y$$

$m << n$

ویژگیهای وابسته خطی