Week 1

— Linear Regression with one variable (Model Representation)

Supervised Learning

Given the "Right Answer" for each example in the data.

Regression Problem

Predict real-value output

Training set of housing prices

Size in feet^2 (x)	Price ($) in 1000's ($y$)
2104	460
1416	232
1536	315
852	178
...	...

Notation:

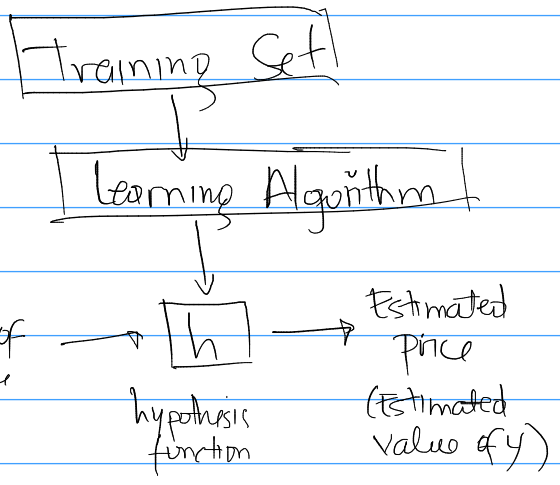
m = Number of training Examples

x's = "input" variable/features

y's = "output" variable/"target" variable

(x, y) — one training example

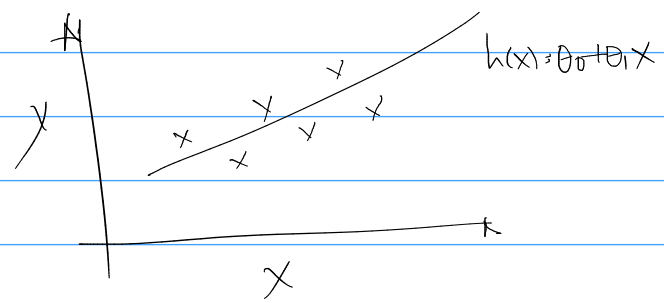
$(x^{(i)}, y^{(i)})$ — i^{th} training example



How do we represent h?

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

Shorthand: $h(x)$



Linear regression with one variable (x)

Univariate linear regression

Linear regression with one variable (Cost function)

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

θ_i's : Parameters

How do we choose θ_i's?

Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ is close to y for our training examples (x, y)

$$\underset{\theta_0, \theta_1}{\text{minimize}} \ \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

$m \rightarrow$ #training examples

$h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$

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

$$\underset{\theta_0, \theta_1}{\text{minimize}} \ J(\theta_0, \theta_1)$$

$J(\theta_0, \theta_1)$: Cost function

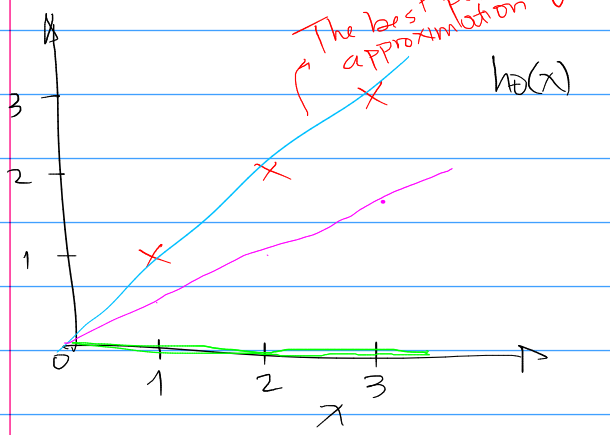
Square error function

Linear regression with one variable (Cost function Intuition I)

	Simplified
Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$	$h_\theta(x) = \theta_1 x$, $\theta_0 = 0$
Parameters: θ_0, θ_1	θ_1
Cost function: $J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$	$J(\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$, $h_\theta(x^{(i)}) = \theta_1 x^{(i)}$
Goal: minimize $J(\theta_0,\theta_1)$ over θ_0,θ_1	minimize $J(\theta_1)$ over θ_1

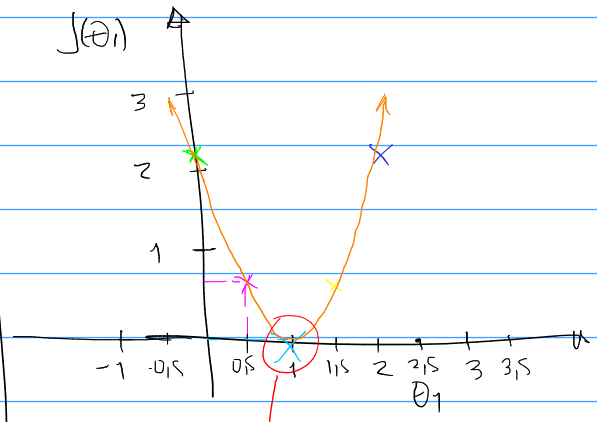
$h_\theta(x)$

(for fixed θ_1, this is a function of x)



$J(\theta_1)$

(function of the parameter θ_1)



If $\theta_1 = 1$ when $m = 3$

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

$$= \frac{1}{2m}(0^2 + 0^2 + 0^2) = 0$$

If $\theta_1 = 0.5$ when $m = 3$

$$J(0.5) = \frac{1}{2m}\left[(0.5-1)^2 + (1-2)^2 + (1.5-3)^2\right]$$

$$= \frac{1}{2\times 3}\cdot(3.5) \simeq 0.58$$

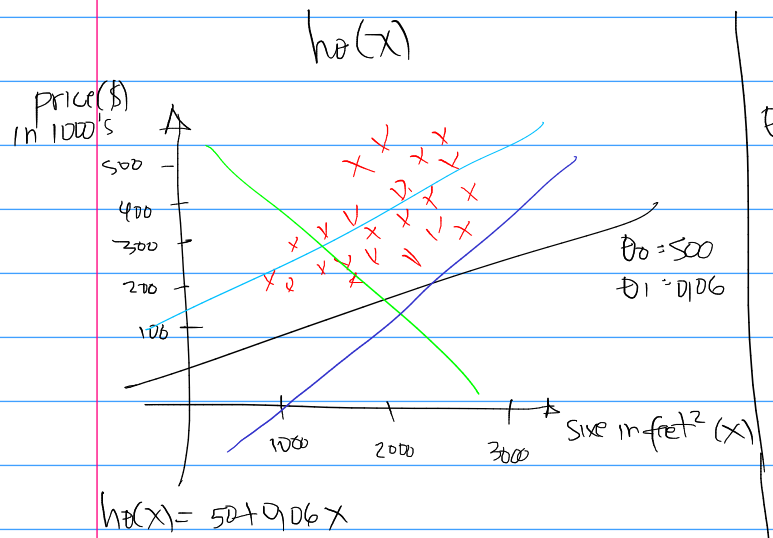
If $\theta_1 = 0$

$$J(0) = \frac{1}{2\cdot 3}\left[(0-1)^2 + (0-2)^2 + (0-3)^2\right]$$

$$= \frac{1}{2\cdot 3}\left[1+4+9\right] = \frac{1}{2\cdot 3}\cdot[14] = \frac{14}{6} = 2.25$$

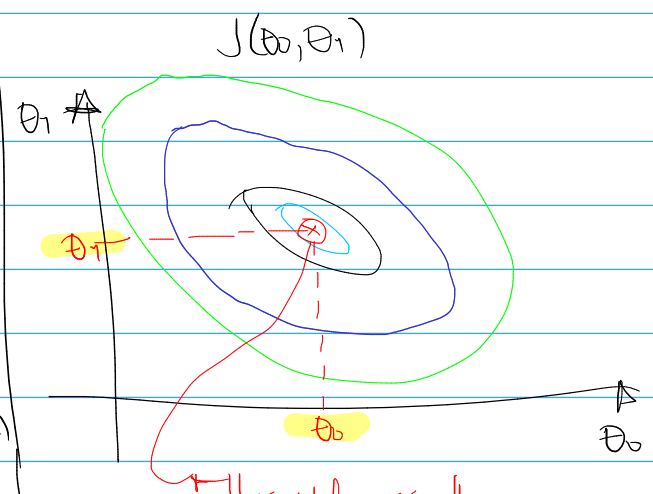
$\theta_1 = 1$ we found the minimum and therefor we find the best possible approximation

Linear regression with one variable (Cost function Intuition II)



$h_\theta(x) = 50 + 0.06x$

Each coordinate (θ_0, θ_1) correspond to an hypothesis function $h_\theta(x) = \theta_0 + \theta_1 x$



This value is the minimum and therefore. Its θ_0 and θ_1 coordinates are the one that makes a better approximation to the data.

Linear Regression with one variable [Gradient descent]

Gradient descent – The most common algorithm

Have some function $J(\theta_0, \theta_1)$

Want $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Outline

- Start with some θ_0, θ_1 (Say $\theta_0 = 0, \theta_1 = 0$)
- Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$ until we hopefully end up at a minimum.

Gradient descent Algorithm

Repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad \text{(for } j = 0 \text{ and } j = 1\text{)}$$

}

α: learning rate

$\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$: derivative term

Simultaneously update θ_0, θ_1.

Correct: Simultaneous Update

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

Incorrect

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

Exercise:

Supose $\theta_0 = 1, \theta_1 = 2$ We simultaneously update θ_0, θ_1 using the rule $\theta_j := \theta_j + \sqrt{\theta_0 \theta_1}$ (for $j = 0$ and $j = 1$)

$$temp0 := \theta_0 + \sqrt{\theta_0 \theta_1} = 1 + \sqrt{2}$$
$$temp1 := \theta_1 + \sqrt{\theta_0 \theta_1} = 2 + \sqrt{2}$$
$$\theta_0 = 1 + \sqrt{2}$$
$$\theta_1 = 2 + \sqrt{2}$$

Linear Regression with one variable (Gradient descent Intuition)

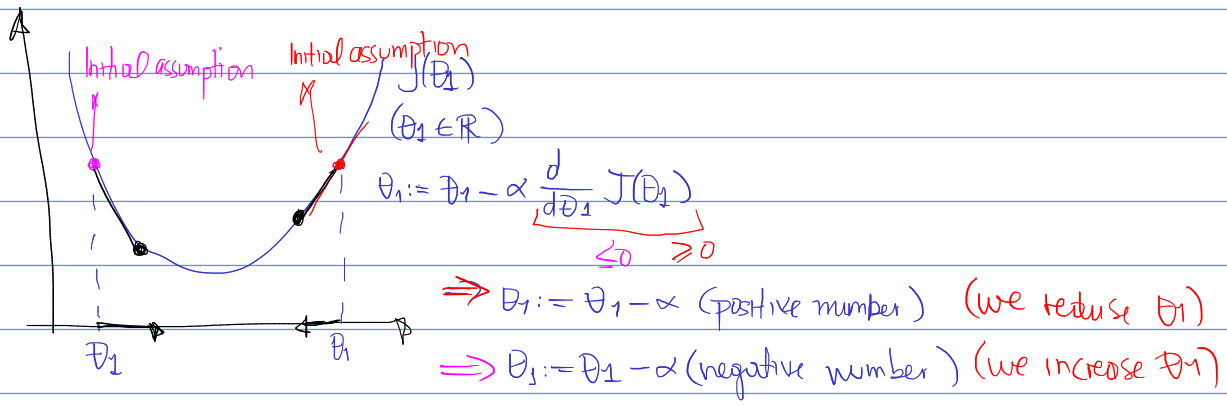
Repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad \text{(simultaneously update for } j=0 \text{ and } j=1\text{)}$$

}

We assume

$\min_{\theta_1} J(\theta_1)$ (just one parameter)



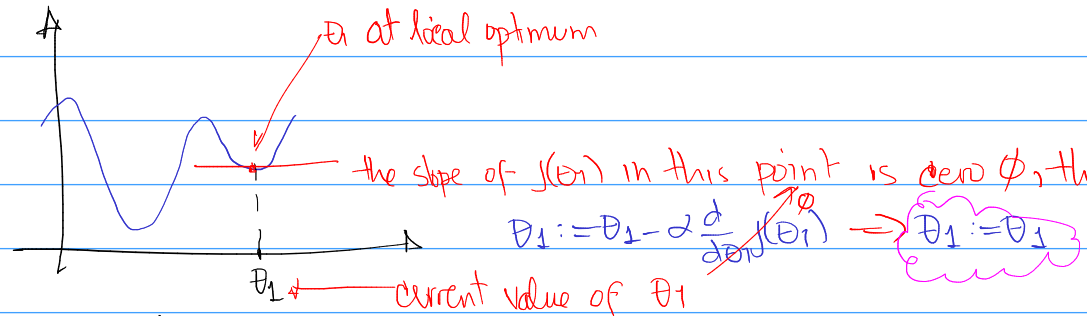
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

If α is too small, gradient descent can be slow

If α is to large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge.

Exercise

Suppose θ_1 is at a local optimum of $J(\theta_1)$, such as shown. What will one step of gradient descent $\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$ do?



Gradient descent can converge to a local minimum, even with the learning rate α fixed. As we approach a local minimum, gradient descent will automatically take smaller steps. So, no need to decrease α over time.

Linear Regression with one variable (Gradient descent for linear regression)

Gradient Descent Algorithm

Repeat until convergence {

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

(for $j=1$ and $j=0$)

}

Linear Regression Model

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

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

We are going to use gradient descent to minimize $J(\theta_0, \theta_1)$ θ_1, θ_2

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

$$= \frac{\partial}{\partial \theta_j} \cdot \frac{1}{2m} \cdot \sum_{i=1}^{m} (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

for θ_0 $j=0$: $\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$

for θ_1 $j=1$: $\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x^{(i)}$

Gradient Descent Algorithm for Linear Regression

Repeat until convergence {

$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$ ∧ $\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x^{(i)}$

}

The function $J(\theta_0, \theta_1)$ for linear regression is always convex, that's why the minimization is always a global optimum.

Gradient Descent Algorith also called "Batch" Gradient Descent

"Batch": Each step of gradient descent uses all the training examples.

Cuestionario Calificado (Linear Regression with One Variable)

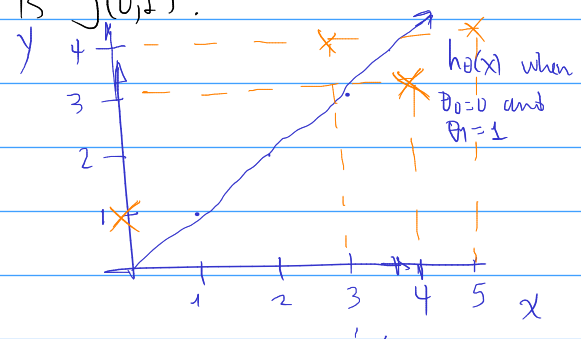
①

X: Number of "A" grades in first year	Y: The number of "A" grades in second year
5	4
3	4
0	1
4	3

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

$\boxed{m = 4}$ # of training examples ①

② $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$ What is $J(0,1)$.



If $\theta_0 = 0 \wedge \theta_1 = 1 \Rightarrow h_\theta(x) = 0 + 1x = x$

$J(\theta_0, \theta_1) = \frac{1}{2 \cdot (4)} \sum_{i=1}^{4} (h_\theta(x^{(i)}) - y^{(i)})^2$

$= \frac{1}{8} \cdot \left((1x^{(1)} - 4)^2 + (x^{(2)} - 4)^2 + (x^{(3)} - 1)^2 + (x^{(4)} - 3)^2 \right)$

$= \frac{1}{8} \cdot \left((5-4)^2 + (3-4)^2 + (0-1)^2 + (4-3)^2 \right) = \frac{1}{8} \left(1^2 + (-1)^2 + (-1)^2 + (1)^2 \right) = \frac{1}{8} \cdot 4 = \frac{4}{8} = \frac{1}{2} = 0{,}5$

③ Suppose we set $\theta_0 = -2$; $\theta_1 = 0{,}5$. What is $h_\theta(6)$?

If $\theta_0 = \cancel{2}$ -2 You Idiot! $\wedge \theta_1 = 0{,}5 \Rightarrow h_\theta(x) = -2 + 0{,}5x \Rightarrow h_\theta(6) = -2 + 0{,}5(6) = -2 + 3 = 1$

④ a. and b are thruth

⑤ $J(\theta_0, \theta_1) = 0$

$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$ For $J(\theta_0, \theta_1) = 0 \Rightarrow$ Our training set can fit perfetly by a straingt line.

② for a) $y(1) = -569{,}6 + 530{,}9(1) = -38{,}7$

for b) $y(1) = -569{,}6 - 530{,}9(1) = -1100{,}5$

③ $\theta_0 = -1$; $\theta_1 = 2$

$h_\theta(x) = -1 + 2x \Rightarrow h_\theta(6) = -1 + 2(6) = -1 + 12 = 11$

④ a and d are correct

Week 2

— Linear Regression with multiple variables (Multiple features)

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

$m = 47$

Notation:

for example $x^{(2)} = \begin{bmatrix} 1416 \\ 3 \\ 2 \\ 40 \end{bmatrix} \in \mathbb{R}^4$, $n = 4$

$\Rightarrow x_3^{(2)} = 2$

Now $x^{(i)}$ is a vector of the features for each $y^{(i)}$

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

Hypothesis:

Previously: $h_\theta(x) = \theta_0 + \theta_1 x$ → We can't use this form because now we have multiple features.

$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4 + \cdots + \theta_n x_n$. → New form of hypothesis

E.g. $h_\theta(x) = 80 + 0.1 x_1 + 0.01 x_2 + 3 x_3 - 2 x_4$

For convenience of notation, define $x_0 = 1$ $(x_0^{(i)} = 1)$

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$$

$[\theta_0\ \theta_1 \cdots \theta_n] = \theta^T$, $(n+1) \times 1$ matrix

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

(1)

$\theta^T x$ $\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} = x$

Multivariate Linear Regression

Linear Regression with multiple variables (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: ~~$\theta_0, \theta_1, \ldots, \theta_n$~~ $= \theta$ $n+1$-dimensioned vector

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

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

Gradient Descent:

Repeat {

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

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

New algorithm ($n \geq 1$)

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 $j = 0, \ldots, n$)

}

$x_j^{(i)}$: just the feature that multiplies θ_j

Linear Regression with multiple variables (Gradient descent in practice 1: Feature Scaling)

Feature Scaling

Idea: Make sure features are on a similar scale

E.g. x_1 = size $(0-2000 \text{ feet}^2)$

x_2 = number of bedrooms $(1-5)$

→ We can scale the features

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

$0 \le x_1 \le 1$

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

$0 \le x_2 \le 1$

That way gradient descent converge quickly.

Get every featured into approximately a $-1 \le x_i \le 1$ range.

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

μ_1 → avg value

s_1 → range (max - min) (standard deviation)

$-0,5 \le x_1 \le 0,5$

$-0,5 \le x_2 \le 0,5$

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

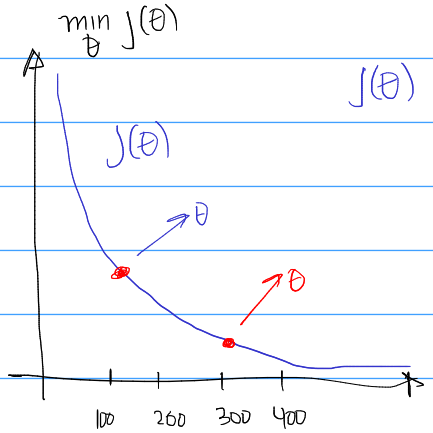
— Linear Regression with multiple variables (Gradient descent in practice II: Learning Rate)

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

— "Debugging": How to make sure gradient descent is working correctly

— How to choose learning rate α.

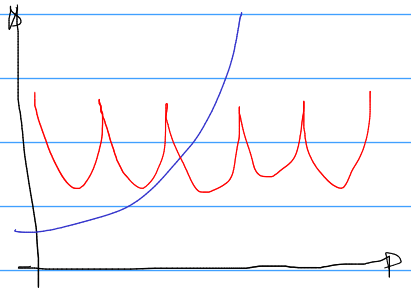
— Making Sure gradient descent is working correctly —



$J(\theta)$ should decrease after every iteration.

Example automatic convergence test:

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



Gradient descent not working.

$\Rightarrow$ Use smaller α.

— For sufficiently small α, $J(\theta)$ should decrease on every iteration.

— But if α is to small, gradient descent can be slow to converge.

To choose α, try:

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

— Linear Regression with multiple variables (Features and polynomial regression)

Housing prices prediction —

$h_\theta(x) = \theta_0 + \theta_1 \times \text{frontage} + \theta_2 \times \text{depth}$

$\text{frontage} = x_1$, $\text{depth} = x_2$

Area

$x = \text{frontage} * \text{depth}$ $\Rightarrow h_\theta(x) = \theta_0 + \theta_1 x$

x: land area.

Polinomial regression —

If we think we should model our hypothesis these ways:

Then $\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2$ v $\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$

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

$= \theta_0 + \theta_1 (\text{size}) + \theta_2 (\text{size})^2 + \theta_3 (\text{size})^3$

where $x_1 = (\text{size})$, $x_2 = (\text{size})^2$, $x_3 = (\text{size})^3$

Size: 1 – 1000

Size^2: 1 – 1000000

Size^3: 1 – 10^9

Linear Regression with multiple variables (Normal Equation)

Method to solve for θ analytically.

Intuition: If $1D$ ($\theta \in \mathbb{R}$)

$J(\theta) = a\theta^2 + b\theta + c$ $\Rightarrow$ $\frac{d}{d\theta} J(\theta) = \dots \overset{\text{set}}{=} 0$ (solve for θ) } This way we found an optimum.

Now.

$\theta \in \mathbb{R}^{n+1}$ $\quad J(\theta_0, \theta_1, \dots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$

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

Solve for $\theta_0, \theta_1, \dots, \theta_n$.

For example $m = 4$

each column is a feature x_{ij}

$$X = \begin{bmatrix} x_0^{(1)} & & & & \\ 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix} \quad m \times (n+1)$$

Each row is a feature $x^{(i)}$

$$y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

m-dimensional vector.

$\Rightarrow \theta = (X^T \cdot X)^{-1} \cdot X^T y$ → Optimum values for θ

Feature Scaling is not necessary with this method.

m training examples, n features.

Gradient Descent	Normal Equation
• Need to choose α	• No need to choose α
• Needs many iterations.	• Don't need to iterate
• Works well even when n is large	• Need to compute $(X^T X)^{-1}$ ($n \times n$) — so much computational cost
	• Slow if n is very large. $n \geq 10{,}000$

Submit() → function to submit your solution.

Quiz

① $x_2^{(4)} = 4761 \Rightarrow x_2 = \begin{bmatrix} 7921 \\ 5184 \\ 8836 \\ 4761 \end{bmatrix}$ (8836 → max, 4761 → min)

$\mu_2 = \frac{7921+5184+8836+4761}{4} = 6700.5$

$S_2 = \max - \min = 8836 - 4761 = 4075 \Rightarrow$

Normalizing $x_2^{(4)} = \frac{4761 - 6700.5}{4075} = -0.47595 \simeq -0.48$

② $m = 28 \quad n = 4$

$X = \begin{bmatrix} x_0^{(1)} & \cdots & x_4^{(1)} \\ x_0^{(2)} & & \vdots \\ \vdots & & \\ x_0^{(28)} & & x_4^{(28)} \end{bmatrix}$ 28×5 matrix ($n+1$)

$y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(28)} \end{bmatrix}$ 28-dimensional vector, 28×1 matrix

$X^T \cdot X = 5 \times 28$ matrix $\cdot$ 28×5 matrix $= 5 \times 5$ matrix

$(X^T \cdot X)^{-1} = 5 \times 5$ matrix

$(X^T \cdot X)^{-1} X^T = 5 \times 5$ matrix $\cdot$ 5×28 matrix $= 5 \times 28$ matrix

$(X^T \cdot X)^{-1} X^{-1} \cdot y = 5 \times 28$ matrix $\cdot$ 28×1 matrix $= 5 \times 1$ matrix

④ The answer is ⓐ

⑤ The answer is ⓐ

① $x^{(3)} = \begin{bmatrix} 94 \\ 8836 \end{bmatrix}$ $x_1^{(3)} = 94$

$x_1 = \begin{bmatrix} 89 \\ 72 \\ 94 \\ 69 \end{bmatrix}$ (94 → max, 69 → min)

$\mu_1 = \frac{89+72+94+69}{4} = 81$

$\mu_4 = \frac{94-69}{2} = 12.5$

$S_1 = 94 - 69 = 25 \Rightarrow$ Normalizing $x_1^{(3)} = \frac{94 - 12.5}{25} = 3.26$ X mala.

Normalizing $x_1^{(3)} = \frac{94 - 81}{25} = 0.52$

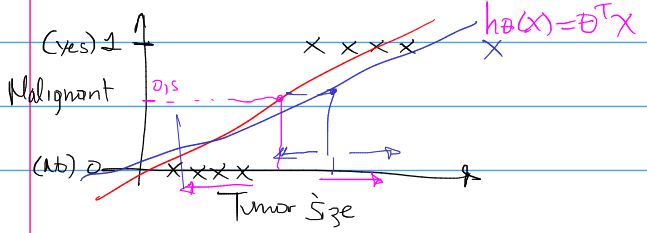
Week 3

— Logistic Regression (Classification)

Email: Spam / Not Spam?

Online transactions: Fraudulent (Yes/No)?

$y \in \{0, 1\}$ → 0: "Negative Class" (e.g., benign tumor)
1: "Positive Class" (e.g., malignant tumor)



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

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

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

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

Classification: $y = 0$ or 1

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

Logistic Regression (Hypothesis Representation)

Logistic Regression Model

Want $0 \le h_\theta(x) \le 1$

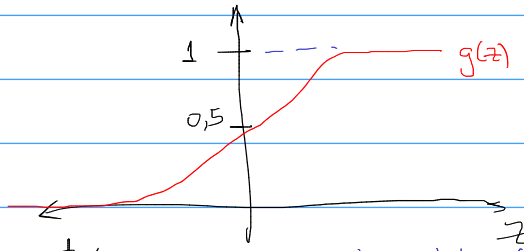
$h_\theta(x) = g(\theta^T x)$

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

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

Sigmoid function

Logistic function



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

We need to fit the parameter θ

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

$h_\theta(x) = 0.7$

Tell the patient that 70% chance of tumor being malignant

$h_\theta(x) = P(y=1|x;\theta)$ "Probability that $y=1$, given x, parameterized by θ"

$y = 0$ or 1

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

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

Logistic Regression (Decision boundary)

$h_\theta(x) = g(\theta^T x) = P(y=1 \mid x; \theta)$

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

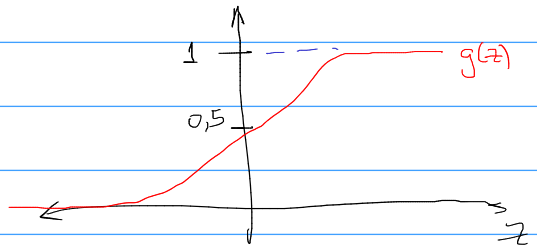
$\theta^T x \geq 0$

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

predict '$y=0$' is $h_\theta(x) < 0.5$ ↳ $g(z) < 0.5$

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

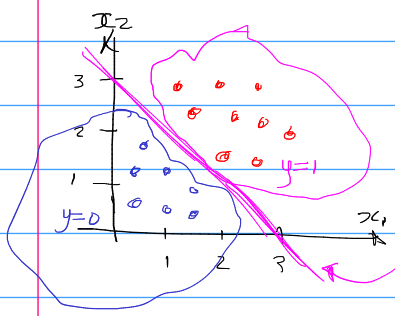
$\theta^T x < 0$



$g(z) \geq 0.5$ when $z \geq 0$

$h_\theta(x) = g(\theta^T x) \geq 0.5$ whenever $\theta^T x \geq 0$

Decision Boundary



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

$\theta_0 = -3$, $\theta_1 = 1$, $\theta_2 = 1$

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

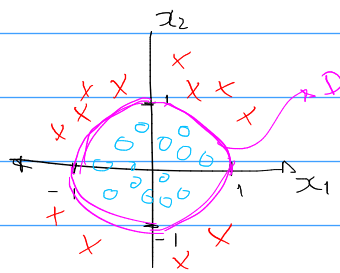
$\theta^T x$

↳ $x_1 + x_2 \geq 3$

x_1, x_2 → $h_\theta(x) = 0.5$ → $x_1 + x_2 = 3$

$x_1 + x_2 < 3$ ↳ $y = 0$

Non-linear decision boundaries



$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$

$\theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$

Predict '$y=1$' if $-1 + x_1^2 + x_2^2 \geq 0$

$x_1^2 + x_2^2 = 1$ $\quad$ $x_1^2 + x_2^2 \geq 1$

Logistic regression (Cost function)

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

$\text{cost}(h_\theta(x^{(i)}), y)$

$\text{cost}(h_\theta(x^{(i)}) - y^{(i)}) = \frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$ ⇝ Non convex function for $\frac{1}{1+e^{-\theta x}}$

Logistic Regression cost function

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

$h_\theta(x) \to 0$, $\text{cost} \to \infty$ ↑ $h_\theta(x) = 1$, $\text{cost} \to 0$

$h_\theta(x) \to 0$, $\text{cost} \to 0$ ↑ $h_\theta(x) \to 1$, $\text{cost} \to \infty$

— Logistic Regression (Simplified cost function and gradient descent)

$\text{cost}(h_\theta(x), y) = -y \log(h_\theta(x)) - (1-y)\log(1-h_\theta(x))$ Better form to express cost function

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

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

To fit parameters θ

$\min_\theta J(\theta)$

To make a prediction given new x:

output $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$ $P(y=1|x;\theta)$

Gradient Descent

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

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

Repeat {

(1) $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$ (simultaneously update all θ_j)

}

Replacing (2) in (1)

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

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

Trying to obtain the $\frac{\partial}{\partial \theta_j} J(\theta) \Rightarrow \frac{\partial}{\partial \theta_j} y^{(i)} \log(h_\theta(x^{(i)})) = \frac{y^{(i)}}{h_\theta(x^{(i)})} \cdot \frac{\partial}{\partial \theta_j}\left(\frac{1}{1+e^{-\theta^T x}}\right)$

$\Rightarrow \frac{y^{(i)}}{h_\theta(x^{(i)})} \cdot \left(-1(1+e^{-\theta^T x})^{-2}\right) \cdot \frac{\partial}{\partial \theta_j}(1+e^{-\theta^T x}) = -\frac{y^{(i)}}{h_\theta(x^{(i)})} \cdot \frac{1}{(1+e^{-\theta^T x})^2} \cdot e^{-\theta^T x}(-x_j)$

$\frac{1}{(1+e^{-\theta^T x})^2} = h_\theta(x^{(i)})^2$

$\frac{\partial}{\partial \theta_j}(1-y^{(i)}) \log(1-h_\theta(x^{(i)})) = \frac{(1-y^{(i)})}{1-h_\theta(x^{(i)})} \cdot \frac{\partial}{\partial \theta_j}(-h_\theta(x^{(i)})) = \frac{(1-y^{(i)})}{(1-h_\theta(x^{(i)}))} \cdot \frac{\partial}{\partial \theta_j}\left(-(1+e^{-\theta^T x})^{-1}\right)$

$= \frac{(1-y^{(i)})}{1-h_\theta(x^{(i)})} \cdot (1) \cdot (1+e^{-\theta^T x})^{-2} \cdot \frac{\partial}{\partial \theta_j}(1+e^{-\theta^T x}) = \frac{(1-y^{(i)})}{1-h_\theta(x^{(i)})} \cdot \frac{1}{(1+e^{-\theta^T x})^2} \cdot e^{-\theta^T x} \cdot (-x_j)$

$\frac{1}{(1+e^{-\theta^T x})^2} = h_\theta(x^{(i)})^2$

$\Rightarrow m\left[\frac{-y^{(i)}}{h_\theta(x^{(i)})} + \frac{(1-y^{(i)})}{1-h_\theta(x^{(i)})}\right] = \left[\frac{-y^{(i)}[1-h_\theta(x^{(i)})] + h_\theta(x^{(i)})[1-y^{(i)}]}{h_\theta(x^{(i)})[1-h_\theta(x^{(i)})]}\right] m$

$m = h_\theta(x^{(i)})^2 \cdot e^{-\theta^T x} \cdot (-x_j)$

$= \frac{-y^{(i)}[1-h_\theta(x^{(i)})] + h_\theta(x^{(i)})[1-y^{(i)}]}{1-h_\theta(x^{(i)})} \cdot h_\theta(x^{(i)}) \cdot e^{-\theta^T x} \cdot (-x_j)$

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

$\left[-y^{(i)} + y^{(i)} h_\theta(x^{(i)}) + h_\theta(x^{(i)}) - y^{(i)} h_\theta(x^{(i)})\right](-x_j) = -(h_\theta(x^{(i)}) - y^{(i)})\, x_j^{(i)}$

$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m}\sum_{i=1}^{m} [h_\theta(x^{(i)}) - y^{(i)}]\, x_j^{(i)}$ (2) replace (2) in (1)

- Logistic Regression (Advanced Optimization)
  - Optimization Algorithms
    - Gradient Descent
    - Conjugate gradient
    - BFGS
    - L-BFGS

Advantage
- No need to manually pick α
- Often faster than gradient descent

Disadvantage
- More complex

Examples

$$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$$

$\min_\theta J(\theta)$

$\theta_1 = 5$

$\theta_2 = 5$

$$J(\theta) = (\theta_1 - 5)^2 + (\theta_2 - 5)^2$$

Octave Example.

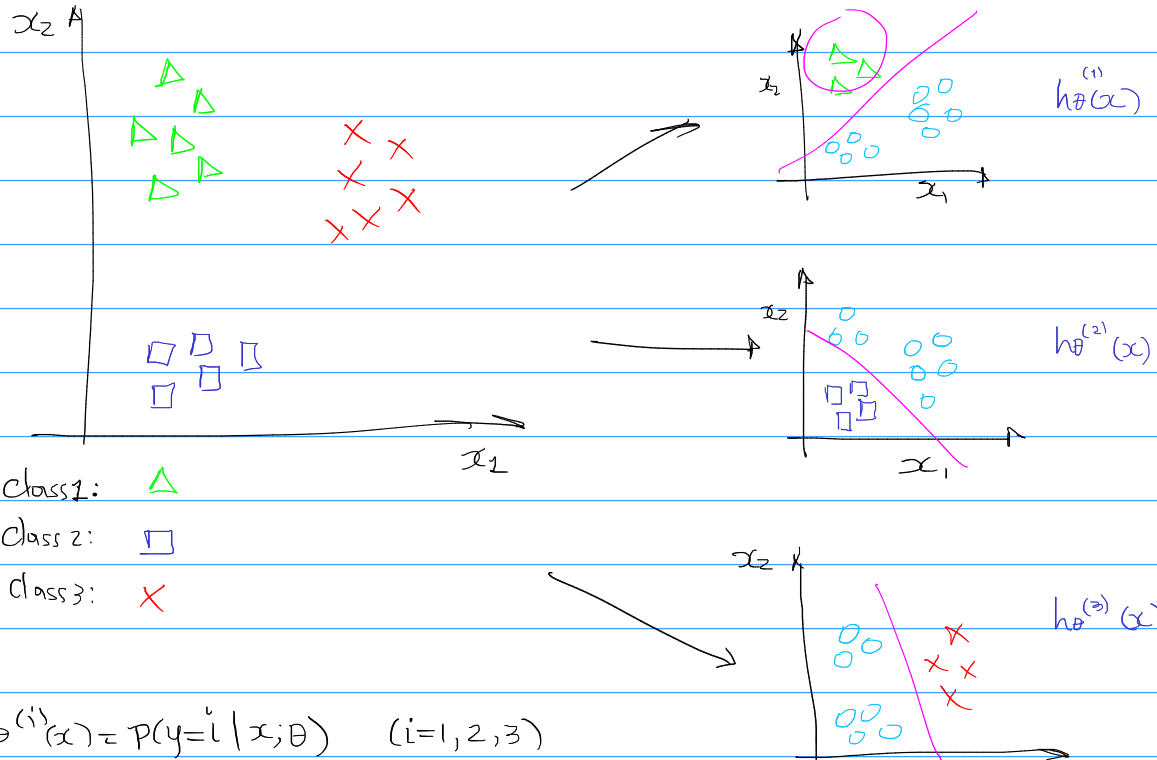
$$\frac{\partial}{\partial \theta_1} J(\theta) = 2(\theta_1 - 5)$$

$$\frac{\partial}{\partial \theta_2} J(\theta) = 2(\theta_2 - 5)$$

- Logistic Regression [Multi-class classification: one-vs-all]

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

$y=1$ $y=2$ $y=3$ $y=4$



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

Train a logistic regression classifier $h_\theta^{(i)}(x)$ for each class i to predict the probability that $y=i$

On a new input x, to make a prediction, pick the class i that maximizes.

$$\max_i h_\theta^{(i)}(x)$$

— Quiz —

⑤ $\theta_0 + \theta_1 x_1 + \theta_2 x_2$ when $\theta_0 = -6$; $\theta_1 = 1$; $\theta_2 = 0$

$\theta_0 = 6$; $\theta_1 = -1$; $\theta_2 = 0$

$-6 + x_1 = 0$

$x_1 = 6$

$6 - x_1 = 0$ → $x_1 = 6$ $\quad$ $6 - x_1 > 0$ → $6 > x_1$

① $P(y=0|x;\theta) = 0,4$ dado $h_\theta(x) = 0,4$ $\quad$ $P(y=1|x;\theta) = 0.4$

$P(y=1|x;\theta) = 0,6$. $\quad$ $P(y=0|x;\theta) = 0,6$

② • adding polynomial features could increase how well we can fit the training data • At optimal value of θ, we will have $J(\theta) \geq 0$

③ for $\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$

↳ $\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(\frac{1}{1+e^{-\theta^T x^{(i)}}} - y^{(i)} \right) x_j^{(i)}$ $\quad$ $\theta := \theta - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$

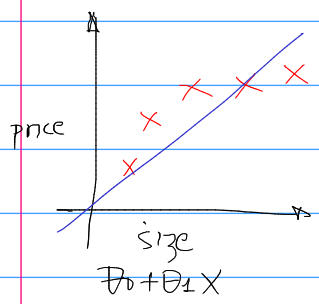
$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$ $\quad$ $\theta := \theta - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(\frac{1}{1+e^{-\theta^T x^{(i)}}} - y^{(i)} \right) x^{(i)}$

④ ↘ sigmoid is never greater than one

↘ The cost $J(\theta)$ for logistic is always greater than or equal to zero

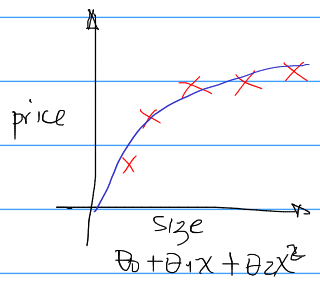
↘ The one vs all allows to use logistic regression for problems in which each $y^{(i)}$ comes from a fixed, discrete set of values.

— Regularization (The problem of overfitting)



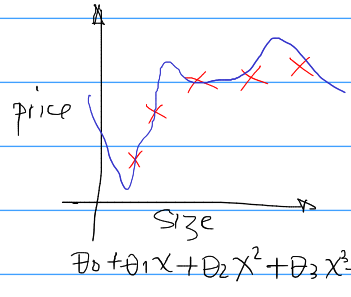
$\theta_0 + \theta_1 x$

'underfit' 'high bias'



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

'just right'



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

'overfit' 'high variance'

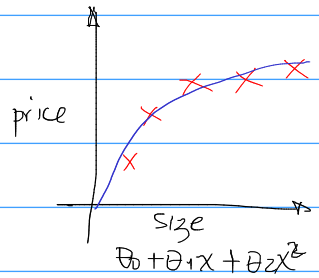
Overfitting: If we have too many features, the learned hypothesis may fit the training set very well ($J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 \simeq 0$), but fail to generalize to new examples (predict prices on new examples)

Addressing overfitting

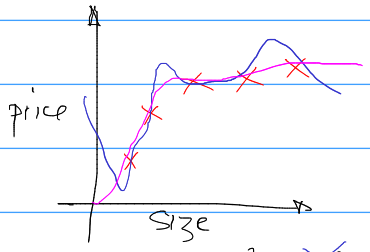
Options:

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)



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



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

Suppose we penalize and make θ_3, θ_4 really small.

$$\min_\theta \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000\,\theta_3^2 + 1000\,\theta_4^2 \implies \theta_3 \approx 0 \wedge \theta_4 \approx 0$$

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

- "Simpler" hypothesis
- Less prone to overfitting

Example.

Housing

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

Regularized cost function

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

λ → regularization parameters

If λ is extremely large value $\implies$ "underfit" prediction

Regularization (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:

Repeat{

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

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

$$(j = 1, 2, 3, \dots, n)$$

}

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

$1 - \alpha\frac{\lambda}{m} < 1$

Normal Equation

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

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

$$\Rightarrow \theta = (X^TX + \lambda\begin{bmatrix} 0 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix})^{-1}X^Ty$$

— Regularization (Regularized logistic Regression)

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

Gradient descent

repeat {

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

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

→ $\frac{\partial}{\partial \theta_j} J(\theta)$ and $h_\theta(x) = \frac{1}{1+e^{-\theta x}}$

Quiz Regularization

① - adding a new feature to the model always results in equal or better performace on the training set ✓

- Introducing regularization to the model always results in equal or better performance on examples not in the training set. *① ✗

② $\theta = \begin{bmatrix} 1.37 \\ 0.51 \end{bmatrix}$ 80°1.

③ - Consider a classification problem. Adding regularization may cause your classifier to incorrectly classify some training examples (which it had correctly classified when not using regularization, i.e. when $\lambda = 0$)

④, ⑤ overfit and underfit graphics.

① - Adding many new features to the model make it more likely to overfit the training set. ✓

- *① ✗

② $\theta = \begin{bmatrix} 2.75 \\ 1.32 \end{bmatrix}$

③ using too large values of λ can cause your hypothesis to underfit the data.

④⑤ graphics