Machine Learning Study
2주차 Review

2016/01/16

정인태

Linear Regression for Multiple Features (다변수 선형회귀)

Single features problem

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

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

Multiple features (variables).

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

$m = 47$

Notation:

- n = number of features $n = 4$
- $x^{(i)}$ = input (features) of i^{th} training example.
- $x_j^{(i)}$ = value of feature j in i^{th} training example.

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

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

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

$$h_\theta(x) = \theta_0 \overset{=1}{x_0} + \theta_1 x_1 + \cdots + \theta_n x_n$$

$$= \boxed{\theta^T x.}$$

$$\underbrace{[\theta_0 \ \theta_1 \cdots \theta_n]}_{\theta^T} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$$

$(n+1) \times 1$ matrix

$\theta^T x$ $\qquad x$

Gradient Descent for Multiple Variables

$x_0 = 1$

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: $\cancel{\theta_0, \theta_1, \ldots, \theta_n}$ $\quad \Theta \quad$ n+1 - dimensional vector

Cost function:

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

$J(\Theta)$

Gradient descent:

Repeat $\{$

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

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

Repeat $\{$

$\frac{\partial}{\partial \Theta_j} J(\Theta)$

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

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

$\}$

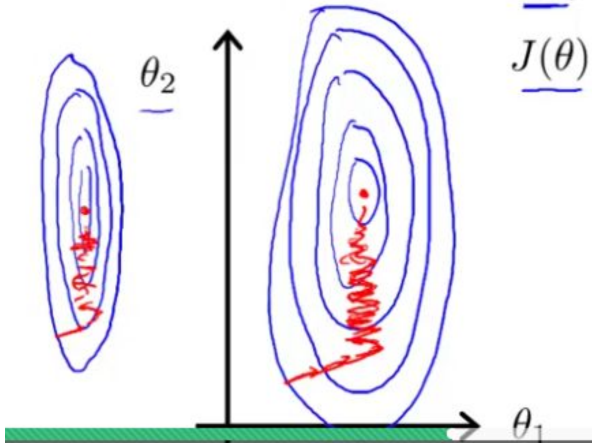
Feature Scaling

Feature Scaling

Idea: Make sure features are on a similar scale.

E.g. x_1 = size (0-2000 feet²) ←

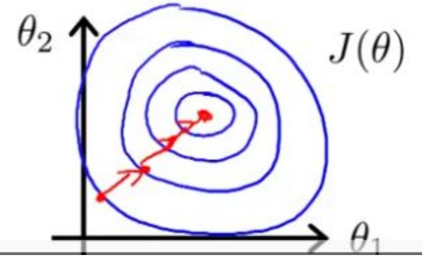
x_2 = number of bedrooms (1-5) ←



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

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

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



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

$x_0 = 1$

$0 \le x_1 \le 3$ ✓

$-2 \le x_2 \le 0.5$ ✓

$-100 \le x_3 \boxed{100}$ ✗

$-0.0001 \le x_4 \le \boxed{0.0001}$ ✗

-3 to 3 ✓

$-\frac{1}{3}$ to $\frac{1}{3}$ ✓

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. → $\boxed{x_1 = \frac{size-1000}{2000}}$ Average size = 100

$\boxed{x_2 = \frac{\#bedrooms-2}{5}}$ 1-5 bedrooms

$\boxed{-0.5 \le x_1 \le 0.5}$ $\boxed{-0.5 \le x_2 \le 0.5}$

$x_1 \leftarrow \frac{x_1 - \mu_1}{s_1}$ ← avg value of x_1 in training set

s_1: range (max - min) (or standard deviation)

$x_2 \leftarrow \frac{x_2 - \mu_1}{s_2}$

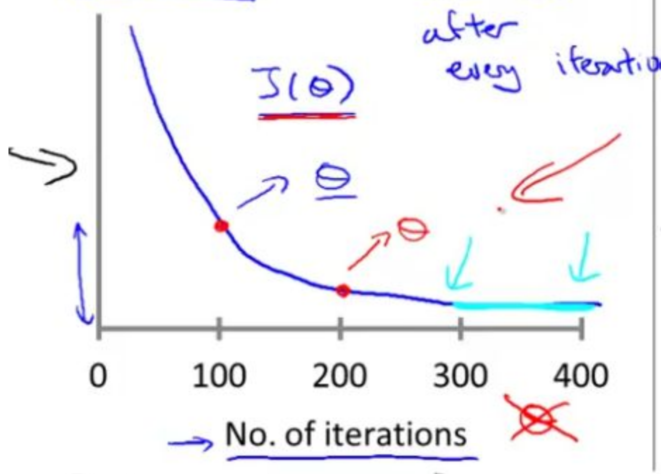
Andrew Ng

How to Choose learning rate α (학습률 선택법)

Making sure gradient descent is working correctly.

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

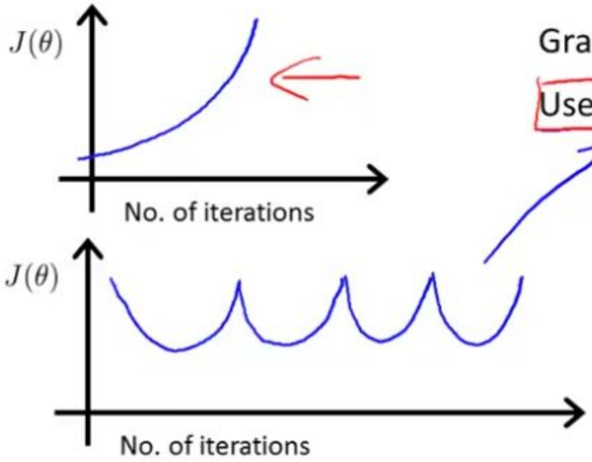
J(θ) should decrease after every iteration.



30, 3000, 3,000,000

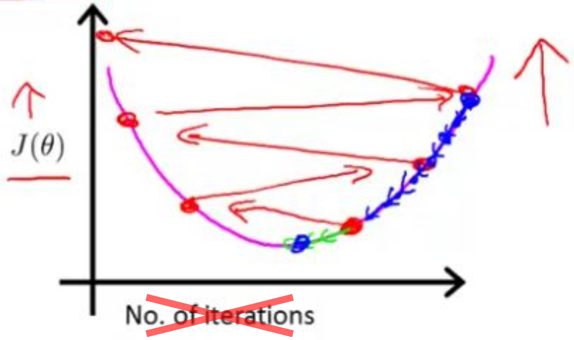
- Example automatic convergence test:
- Declare convergence if $J(\theta)$ decreases by less than $\boxed{10^{-3}}$ in one iteration.

ε



Gradient descent not working.

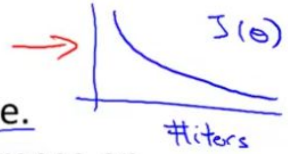
Use smaller α.



오타수정 : θ

- 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

Deriving a new feature from existing features

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

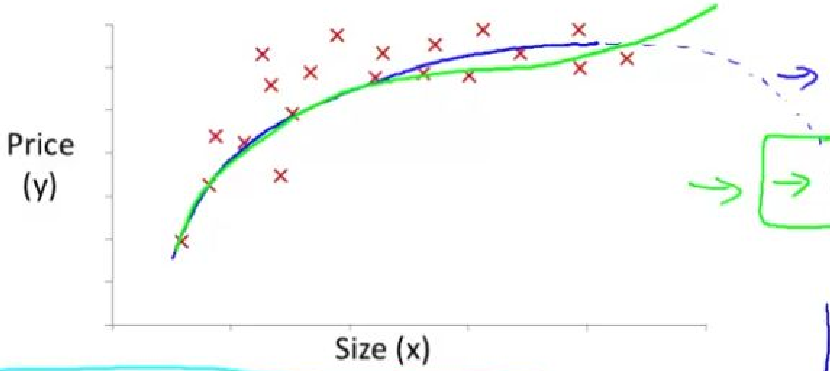
$x = \text{frontage} * \text{depth}$

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

land area

Polynomial Regression (다항식회귀)

Polynomial regression



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

$\rightarrow \boxed{\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3}$

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

$\rightarrow x_1 = (size)$

$\rightarrow x_2 = (size)^2$

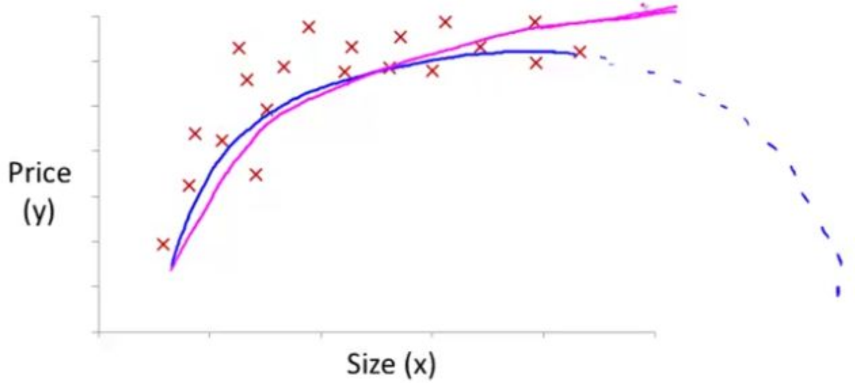
$\rightarrow x_3 = (size)^3$

Size: 1 - 1000

$Size^2$: 1 - 1000,000

$Size^3$: 1 - 10^9

Choice of features



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

$\rightarrow \boxed{h_\theta(x) = \theta_0 + \theta_1 (size) + \theta_2 \sqrt{(size)}}$

Feature scaling에 유의!

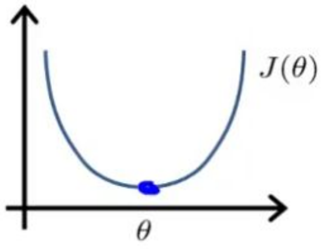
Analytic Solution : Normal Equation (1/2)

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

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

$\frac{d}{d\theta} J(\theta) = \dots \overset{set}{=} 0$

Solve for θ



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

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

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

m **examples** $(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})$; n **features.**

$$x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ x_2^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix} \in \mathbb{R}^{n+1} \qquad \underset{\text{(design matrix)}}{X} = \underset{m \times (n+1)}{\begin{bmatrix} — (x^{(1)})^T — \\ — (x^{(2)})^T — \\ \vdots \\ — (x^{(m)})^T — \end{bmatrix}}$$

E.g. If $x^{(i)} = \begin{bmatrix} 1 \\ x_1^{(i)} \end{bmatrix}$ $\quad X = \underset{m \times 2}{\cancel{\begin{bmatrix} 1 & x_1^{(1)} \\ 1 & x_2^{(1)} \\ \vdots & \vdots \\ 1 & x_m^{(1)} \end{bmatrix}}}$ $\quad y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$

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

Examples: $m = 4$.

x_0	Size (feet²) x_1	Number of bedrooms x_2	Number of floors x_3	Age of home (years) x_4	Price ($1000) y
1	2104	5	1	45	460
1	1416	3	2	40	232
1	1534	3	2	30	315
1	852	2	1	36	178

$$X = \underset{m \times (n+1)}{\begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix}} \qquad y = \underset{m\text{-dimensional vector}}{\begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}}$$

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

※ 위 예제는 m>=4 case 라서 $X^T X$ 역행렬 존재 안함
(Octave tutorial 마지막 강의 참조)

오타수정 : x 밑첨가 1로 유지, 윗첨자가 1~m 사이

Analytic Solution : Normal Equation (2/2)

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

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

Set $A = X^T X$

$\boxed{(X^T X)^{-1}} = A^{-1}$

Octave: `pinv(X'*X)*X'*y`

X' X^T

$\text{pinv}(X^T * X) * X^T * y$

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

$\min_\theta J(\theta)$

Feature Scaling

$0 \le x_1 \le 1$

$0 \le x_2 \le 1000$

$0 \le x_3 \le 10^{-5}$

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 $(X^T X)^{-1}$ $n \times n$ $O(n^3)$
	• Slow if n is very large.
$n = 10^6$	$n = 100$
	$n = 1000$
	$n = 10000$

Octave Tutorial (1/4)

변수대입 : =
논리연산 equal : ==
논리연산 not equal : ~=
And : &&   Or : ||

Prompt 바꾸기 : PS('>> ')
원주율상수 : pi
pi를 3.14로 표기 예) disp(sprint('0.2f', a))

소숫점아래 표현수 길거나 짧게 :
format long, format short

행렬변수 입력 예: A=[1 2; 3 4; 5 6]
행벡터 예: x=[1 2 3] or x=[1,2,3]
열벡터 예: y=[1; 2; 3]
균등한 수로 행벡터 만들기 예 :
v=1:6 , v=1:0.1:2

1로 구성된 3x3 행렬 : ones(3)
1로 구성된 3x2 행렬 : ones(3, 2)
0로 구성된 행렬은 zeros(...)

2로 구성된 3x3 행렬 : 2*ones(3)
0~1 uniform random 수로 구성된 행렬 : rand(...)
표준정규분포 N(0,1) random 수로 구성된 행렬 : randn(...)
정규분포 $N(m,\sigma^2)$: m + σ*randn(...)

※ 위의 행렬 생성 함수의 argument 를 생략시 default 1

히스토그램 : hist(a)
히스토그램 간격 50 : hist(a,50)

4x4 단위행렬 : eye(4)

명령어 도움말 : help eye (등등..)

3x2 행렬 A 크기 얻기 : size(A)   (=> [3 2] return 함)
3x2 행렬 A 행크기 얻기 : size(A,1)
3x2 행렬 A 열크기 얻기 : size(A,2)
열,행벡터 크기 얻기 : length(v)
행x렬 중 큰 크기 얻기 : length(A)   (3x2 행렬이면 3 return)

Octave Tutorial (2/4)

현재 작업 path : pwd
폴더 변경 : cd 'C:\Users\and\Desktop
파일 불러오기 : load 파일명 or load('파일명')
메모리에 있는 변수 list 보기 : who
변수들 정보 자세히 보기 : whos

벡터 일부만 잘라오기 : v=priceY(1:10)
(첫 10원소 반환)
변수 저장 (binary) : save 파일명 변수명
변수 저장 (text) : save 파일명 변수명 -ascii

변수 모두 지우기 : clear

행렬 A의 (m,n) 원소반환 : A(m,n)
행렬 A의 2행벡터 반환 : A(2,:)
행렬 A의 2열벡터 반환 : A(:,2)
행렬 A의 1, 3행 반환 : A([1 3], :)
행렬 A의 2열에 입력 : A(:,2) = [10; 11; 12]

행렬 A 열 확장 : A = [A, [열벡터]]
행렬 A 행 확장 : A = [A; [행벡터]]

행렬의 모든 원소를 열벡터로 표현 : A(:)
행렬 A B 옆에 덧 붙이기 : C = [A B]
행렬 A B 밑에 덧 붙이기 : C = [A; B]

행렬 곱 : A*B
원소마다 곱하기 : A .*B
모든 원소 제곱하기 : A .^2
모든 원소 역수취하기 : 1 ./ A
모든원소에 함수 적용 : log(A) , exp(A) , abs(A) 등...
스칼라 곱 : -A , 3*A 등..
벡터 v 모든원소 1씩 더하기 : v+1

Transpose 취하기 (전치행렬) : A'

벡터 원소 중 최댓값 : max(v)
벡터 원소 최댓값, 좌표 반환 : [val, ind] = max(v)
행렬 A의 열 별 최댓값 : max(A)
벡터 a 의 원소별 논리 연산 : a >3
(3보다 큰 원소 위치에 1, 아니면 0 반환)
벡터 a 에서 3보다 큰 원소 좌표 반환 : find(a >3)

3x3 마방진 행렬 반들기 : magic(3)

Octave Tutorial (3/4)

3x3 마방진 행렬 반들기 : A=magic(3)
(모든 행, 열, 각선합 동일)
행렬 A에서 3보다 큰 좌표(행,렬)찾기 : [r,c]=find(A>3)

벡터 모든 원소 합 : sum(v)
벡터 모든 원소 곱 : prod(v)
(행렬에 적용시 열마다 따로 함)

정수로 내림 또는 올림 : floor(a) , ceil(a)

두 3x3 랜덤 행렬을 원소별로 더 큰 원소 취한 행렬 :
max(rand(3), rand(3))

행렬 A 열 별 최댓값 으로 구성된 행벡터 : max(A,[],1)
행렬 A 행 별 최댓값 으로 구성된 열벡터 : max(A,[],2)

행렬 A 원소중 최댓값 : max(max(A)) or max(A(:))

행렬 열 별 합으로 행벡터 구성 : sum(A,1)
행렬 행 별 합으로 열벡터 구성 : sum(A,2)
행렬 위아래 뒤집기 : flipud(A)

pseudo 역행렬 : pinv(A) (역행렬 존재 안해도 뭔가 나옴)
그냥 역행렬 : inv(A)

<삼각함수 그래프 그리기>
t = [0:0.01:0.98]
y1 = sin(2*pi*4*t)
y2 = cos(2*pi*4*t)
plot(t, y1, ‘r’) (빨간색으로 그리기)
plot(t, y2) (그래프 갱신됨, 중첩시키려면 중간에 hold on)

xlabel(‘time’)
ylabel(‘value’)
legend(‘sin’, ‘cos’)
title(‘my plot’)
print –dpng ‘myPlot.png’   (png 파일로 저장)

close

Octave Tutorial (4/4)

figure 선택 : figure(1); plot(t, y1)
figure(2); plot(t, y2)
plot 나누기 : subplot(1,2,1); plot(t, y1)
(1x2 로 나누고 1번째 grid선택)
subplot(1,2,3); plot(t, y2)
x, y axis 범위 바꾸기 : axis([0.5 1 -1 1])
figure 지우기 (닫지말고) : clf

<Matrix map 예제>

```
A = magic(5)
imagesc(A)
imagesc(A) , colorbar , colormap gray;
```

(, 로 명령어 여러개 한줄에 처리 가능)

<For, while, if 문 예제들>

```
for i=1:10
    v(i) = 2^i;
end;
v

Indices = [1:10]

for i=Indices,
    disp(i);
end;

i=1;
while i<=5
    v(i)=100;
    i=i+1;
end;

i=1;
while true
    v(i) = 999;
    i = i+1;
    if i==6,
        break
    end;
end;

v(1)=2;
if v(1)==1,
    disp('The value is one')
elseif v(1)==2,
    disp('The value is two')
else
    disp('The value is not one or two.')
end;
```

<함수만들기 예 : file로 저장>

```
function y = squareThisNumber(x)
y=x^2;

squareThisNumber(5)
```

path추가 : addpath('...')

return 값 2개도 가능..

```
function [y1,y2] = SquareCubeNumber(x)
y =x^2;
y2=x^3
```

Vectorization

Vectorization example.

$$h_\theta(x) = \sum_{j=0}^{n} \theta_j x_j$$

$$= \theta^T x$$

theta(1)

theta(2)

theta(3)

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix} \qquad x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix}$$

Unvectorized implementation

```
prediction = 0.0;
for j = 1:n+1,
  prediction = prediction +
               theta(j) * x(j)
end;
```

Vectorized implementation

```
prediction = theta' * x;
```

Vectorization example.

$$h_\theta(x) = \sum_{j=0}^{n} \theta_j x_j$$

$$= \theta^T x$$

Unvectorized implementation

```
double prediction = 0.0;
for (int j = 0; j <= n; j++)
  prediction += theta[j] * x[j];
```

Vectorized implementation

```
double prediction
  = theta.transpose() * x;
```

Normal Equation Noninvertibility (역행렬 존재 X)

What if $\boxed{X^TX}$ is non-invertible?

- Redundant features (linearly dependent).
  E.g. x_1 =size in feet^2
  ~~x_2 = size in m^2~~
  $x_1 = (3.28)^2 x_2$

  1m = 3.28 feet

  $m = 10$
  $n = 100$
  $\theta \in \mathbb{R}^{101}$

- Too many features (e.g. $m \leq n$).
  - Delete some features, or use regularization.

  later