Notes – Week 2

Multiple features

house example continued:

Size (feet^2)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
x_1	x_2	x_3	x_4	y_1
2104	3	2	40	
1416	3	2	30	

(*) → output

↳ features.

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.

index into training set.

$$x^{(2)} = \begin{bmatrix} 1416 \\ 3 \\ 2 \\ 40 \end{bmatrix} \in \mathbb{R}^4 = 4\text{th dimensional vector}$$

$\mathbb{R}^n$

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

---

hypothesis in linear regression.

now with example (*)

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

$$= \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

$(x_0^{(i)} = 1)$ $x_0 = 1$

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

notes week 2 (multiple features cont'd)

$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \dots + \theta_n x_n$

($x_0 = 1$)

$= \theta^T x$

$\theta^T = [\theta_0\ \theta_1 \dots \theta_n]$

θ^T $(n+1) \times 1$ matrix

or row vector...

multivariate linear regression

↳ (many variables) for which we try to predict the value Y.

Notes - Week 2

Gradient descent for Multiple Variables

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

Parameters: $\underline{\theta}$ $n+1$ dimensional vector

↳ $\theta_0, \theta_1, \ldots \theta_n$

Cost function:

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

∴ Gradient descent:

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

new algorithm for when features $(n) > 1$

Repeat

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

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

$x_0^{(i)} = 1$!!!

Notes - Week 2

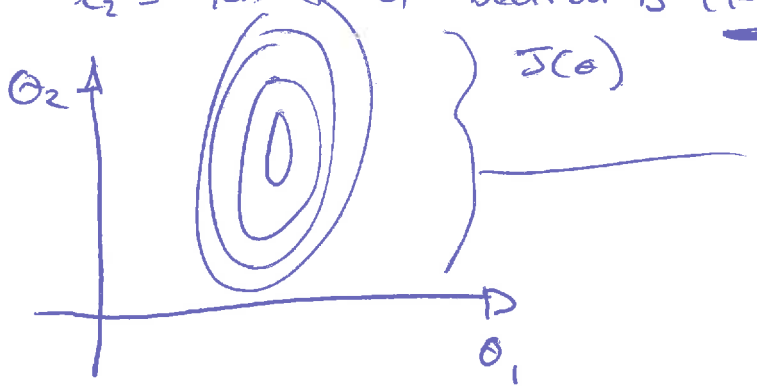
Gradient Descent in Practise I

Feature Scaling

Idea: Make sure features are on a similar scale

e.g. x_1 = size (0-2000 $feet^2$) Δ

x_2 = number of bedrooms (1-5) □



skewed due to the difference in scale between Δ e □

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

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

} then we scale J(θ) so if looks like so

Feature scaling

Lo get every feature into appoximately a $-1 \leq x_i \leq 1$ range.

Mean normalization

Replace x_i with $x_i - \mu_i$ to make features have appoximately zero mean (DO NOT Apply to $x_0 = 1$)

e.g. $x_1 = \frac{\text{size} - 1000}{2000}$; $x_2 = \frac{\#\text{bedrooms} - 2}{5}$

$-0.5 \leq x_1 \leq 0.5$; $-0.5 \leq x_2 \leq 0.5$

$$x_1 = \frac{x_1 - \mu}{s_1}$$

μ — average value of x_1 in training set

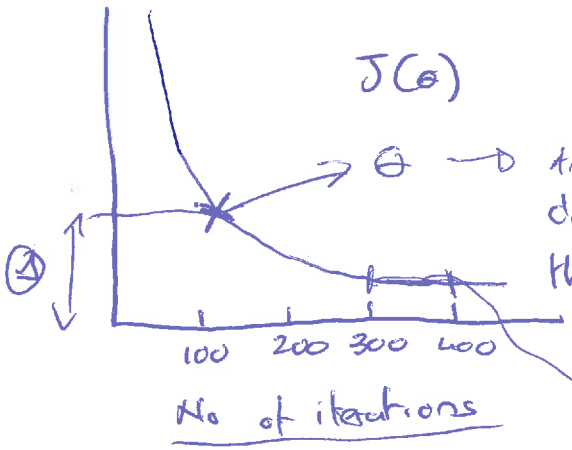
s_1 — (max − min) range of x_1 (or standard deviation.)

Notes - Week 2

Gradient descent in Practice II
- learning rate

"Debugging" gradient descent

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



→ after 100 evaluations of gradient descent → the y axis (A) is the magnitude

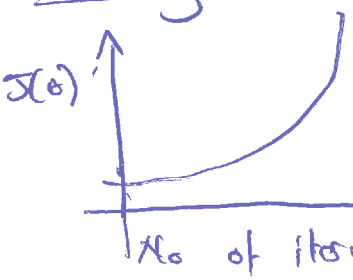
almost converged.

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

an example of an automatic convergence test:

→ "declare convergence if $J(\theta)$ decreases by less than 10^{-3} in one iteration" ✓ good idea.

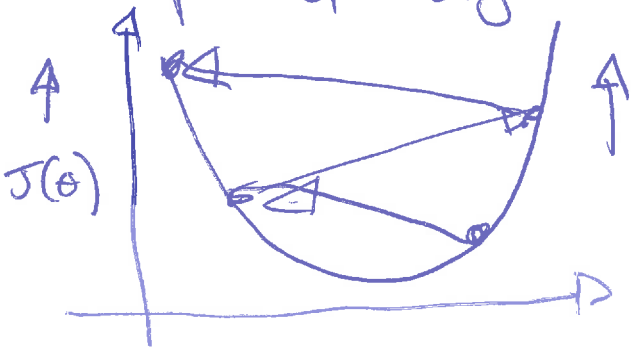
Making sure gradient descent is working correctly



Gradient descent not working

↳ Use smaller α

example of big α.



To choose α try:

0.001, 0.01, 0.1, 1

3x → 0.003

3x → 0.03

0.3

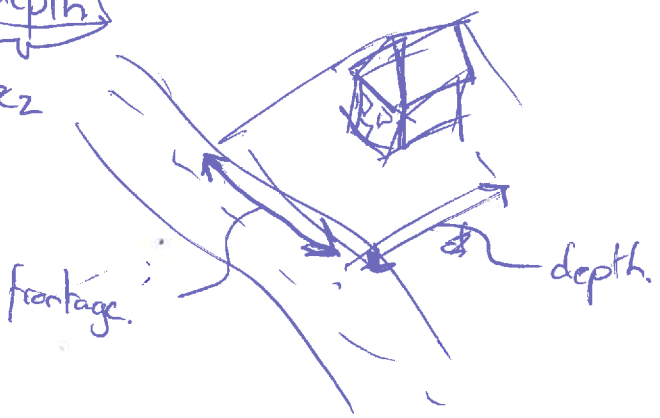
3.

Week 2 – Notes

Features and polynomial regression

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

x_1 x_2



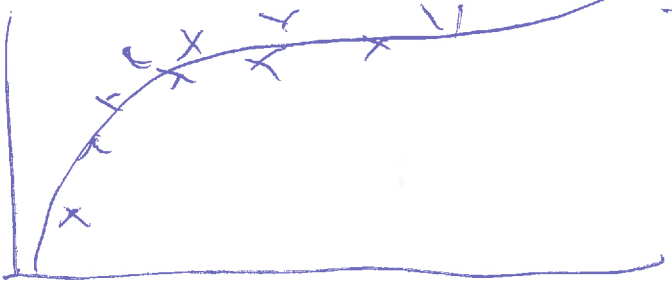
Area

x = frontage × depth

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

new feature. ← Land area.

Polynomial regression



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

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

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

$x_1 = (\text{size})$

$x_2 = (\text{size})^2$

$x_3 = (\text{size})^3$.

house

size: 1 – 1000

size²: 1 – 1000,000

size³: 1 – 10^3.

week 2 notes

Normal equation

examples: $m = 4$

x_0	Size x_1	Number of bedrooms x_2	number of floors x_3	Age of home 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

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & & & & \\ 1 & & & & \end{bmatrix} \qquad y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

$m \times (n+1)$ matrix

m-dimensional vector.

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

↑ value of θ that minimizes cost function

$$x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ x_2 \\ \vdots \\ x_n^{(i)} \end{bmatrix} \in \mathbb{R}^{n+1}$$

e.g. If $x^{(i)} = \begin{bmatrix} 1 \\ x_1^{(i)} \end{bmatrix}$

constructed matrix $X = \begin{bmatrix} 1 & x_1^{(i)} \\ & \\ & \end{bmatrix}$

$$y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

week 2 notes

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

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

octave pinv(X'*X)*X'*y

Set $A = X^T X$

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

---

m training examples, n features

Gradient descent

- need to choose α
- needs many iterations
- Works well even when n is large.

Normal equation

- no need to choose α
- don't need to iterate.
- Need to compute $(X^T X)^{-1}$ $n \times n$ $O(n^3)$ ← cost to invert matrix.
- slow if n is very large.

$n = 100$ NE

$n = 1000$ NE

$n = 10000$ HE

$n > 10000$ NE

Week 2

notes

Normal equation non-invertibility

What if X^TX is non-invertible?

Redundant features (linearly dependant)

e.g. x_1 = size in feet^2

x_2 = size in m^2

$\Big\}$ $\therefore$ X^TX becomes non invertible.

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

do this

1st and this

2nd