Multiple Features;

May 13, 2021

Notation;

n = # of features

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

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

e.g. $x^{(2)} = \begin{bmatrix} 1416 \\ 3 \\ 2 \\ 40 \end{bmatrix}$ → size, → # bedrooms, → # of floors, → age of home

$x_2^{(2)} = 3$

✱Now it starts at 0

Now; $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \dots \theta_n x_n$

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

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

✱Multivariate linear regression

Gradient Descent for Multiple Variables:

Cost Function:

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

Gradient Descent:

repeat {

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

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

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

$$\vdots$$

simultaneous update

}

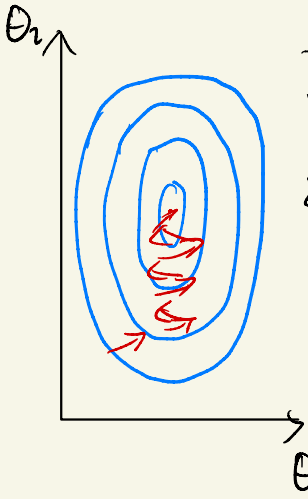
$$\therefore \boxed{\text{repeat } \{ \theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) x_j^{(i)} \}}$$

*Simultaneous update θ_j for $j = 0, 1 \dots n$

Feature Scaling:

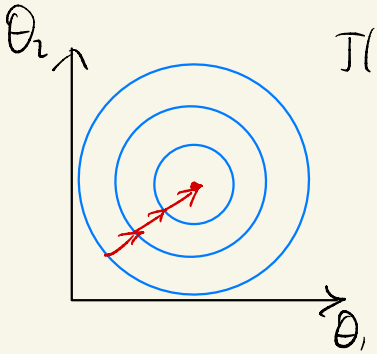
e.g. x_1 = size $(0-2000\,ft^2)$ $\Rightarrow$ $x_1 = \frac{\text{size}}{2000}$

x_2 = # bedrooms $(1-5)$ $\quad x_2 = \frac{\text{\# bedrooms}}{5}$



$J(\theta)$

- GD, takes a long time



$J(\theta)$

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

- Like ÷255 for img pixels
- Not the same as normalization!
  - ↳ changes shape of distribution

- Feature scaling allows for the accuracy to converge faster
  - ↳ usually you want features to be in $-1 \le x_i \le 1$ range
    - Some others are fine, too

Mean Normalization:

Mean normalization involves replacing x_i w/ $x_i - \mu_i$ to make features have ≈ zero mean.

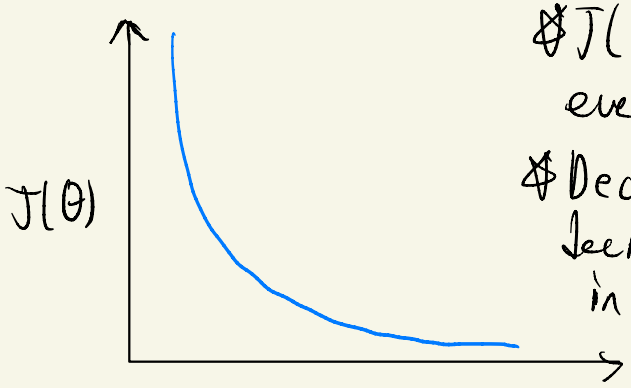
- Do not apply to $x_0 = 1$
- Can be used w/ feature scaling

e.g. $x_1 = \frac{\text{size} - 1000}{2000}$ → avg of x_1 in training set
→ range (max - min)

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

↳ $-0.5 \leq x_1 \leq 0.5$, $-0.5 \leq x_2 \leq 0.5$

Is Gradient Descent Working?

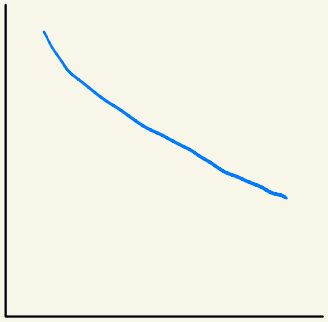


- $J(\theta)$ should decrease after every iteration
- Declare convergence if $J(\theta)$ decreases by less than 10^{-3} in one iter

↓ hyperparameter, chosen manually

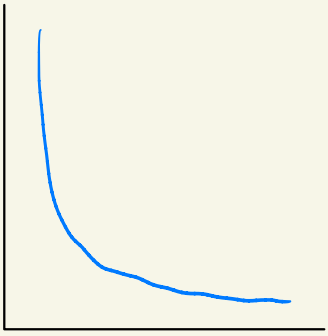
- For sufficiently small α, $J(\theta)$ should decrease on every iter

Learning Rate Cases:

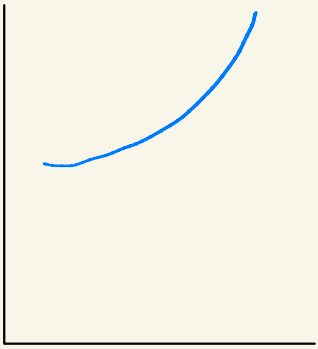


α too low

↳ slow convergence



α just right



α too high

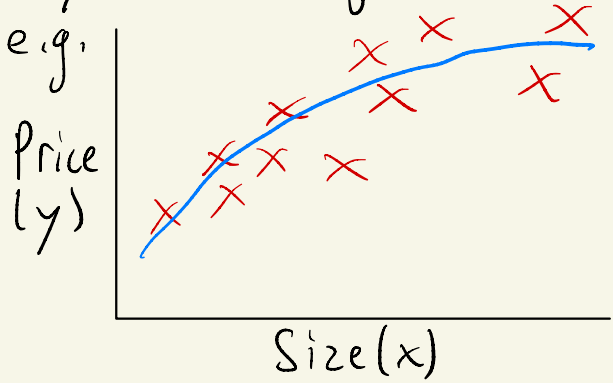
↳ doesn't converge

Try $\alpha = 0,001, 0,01 \ldots 1$ (factor of 10)

or $\alpha = 0,001, 0,003, 0,01 \ldots 1$ (factor of ~3)

Polynomial Regression:

e.g.



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

Must Feature Scale!

or else:

size $(1-1000)$

size^2 $(1-10^6)$

size^3 $(1-10^9)$

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

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

e.g. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 \sqrt{x}$

let x be size (1-1000)

Feature Scaling!

$h_\theta(x) = \theta_0 + \theta_1 \frac{x}{1000} + \theta_2 \frac{\sqrt{x}}{\sqrt{1000}}$