MULTIPLE FEATURES

→ So far in the previous lectures of linear regression that we discussed, we had linear regression in only 1 variable — the size of the house — with which we wanted to predict the price of the house

→ Now imagine if we had not only the size of the house but more information like this -

→ This would give us a lot more intuition

Size (feet²) 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
...	...	...	...	...

→ $n = 4$

→ $x^{(2)} = \begin{bmatrix} 1416 \\ 3 \\ 2 \\ 40 \\ 232 \end{bmatrix}$ 4 Dimensional Vector

→ $x_3^{(2)} = 2$

→ From now on, we will denote each such variable / feature as (x_n)
↳ So we will have x_1, x_2, x_3, x_4 and so on for features
→ The value being predicted still remains the same — y

→ To formalize this approach, now that we have 4 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

• How does this affect the hypothesis Function ?

→ Previously, hypothesis was : $h_\theta(x) = \theta_0 + \theta_1 x$: but now that we multiple features we are not going to use this simple representation anymore.
→ Instead we are going to use something like this :

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

→ So, for example the house pricing problem :

$$h_\theta(x) = 80 + \underbrace{0.1x_1}_{\text{size}} + \underbrace{0.01x_2}_{\#\text{bedrooms}} + \underbrace{3x_3}_{\#\text{floors}} - \underbrace{2x_4}_{\text{age}}$$

→ For convenience of notation, let $x_0 = 1$. We will add this feature with value 1 to simplify the equation :

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

$$\Rightarrow \quad x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$$

These are both $(n+1)$ dimensional vectors now

And,

$$h_\theta(x) = \theta_0 \overset{=1}{x_0} + \theta_1 x_1 + \theta_2 x_2 + \ldots\ldots + \theta_n x_n$$

$= \theta^T x$

MULTIVARIATE LINEAR REGRESSION

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

GRADIENT DESCENT FOR MULTIVARIATE LINEAR REGRESSION

→ Hypothesis — $h_\theta(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_3 + \dots + \theta_n x_n$

→ Parameters — $\theta_0, \theta_1, \theta_2 \dots \theta_n$ now replaced by $(n+1)$ dimensional vector $\boxed{\theta}$

→ Cost Function — $J(\theta_0, \theta_1, \theta_2 \dots \theta_n)$ now replaced by $J(\theta)$

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

→ Gradient Descent — Repeat {

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

}

(simultaneously update for every $j = 0, 1 \dots n$)

- So, after putting the values from these new definitions :

Gradient Descent —

$$\boxed{\begin{aligned} &\text{Repeat } \{ \\ &\quad \theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) x_j^{(i)} \\ &\} \\ &(\text{simultaneously update } \theta_j \text{ for } j = 0, 1, 2 \dots n) \end{aligned}}$$

→ New algorithm derivation

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

$$\Rightarrow \frac{1}{2m} \sum_{i=1}^{m} (\theta_0 x_0^{(i)} + \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)} + \dots + \theta_n x_n^{(i)} - y^{(i)})^2$$

↳ $\frac{\partial}{\partial \theta_0} J(\theta) = \frac{1}{\cancel{2}m} \cdot \cancel{2} \cdot \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_0^{(i)}$

↳ $\frac{\partial}{\partial \theta_1} J(\theta) = \frac{1}{\cancel{2}m} \cdot \cancel{2} \cdot \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_1^{(i)}$

↳ We can clearly see a pattern here :

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

→ This is what we have used in the new gradient descent algorithm

FEATURE SCALING

→ If we have a dataset with multiple features and we scale them to a similar scale, then gradient descent will converge more quickly

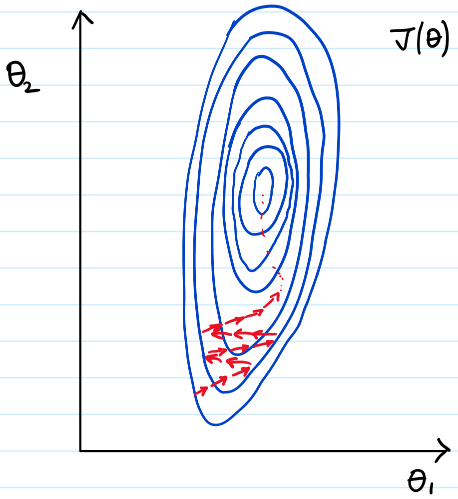
★ EXAMPLE

Suppose we are dealing with 2 features while predicting the price of a house

$$x_1 = \text{Size of the house} \quad (0 - 2000 \text{ feet}^2)$$

$$x_2 = \text{Number of bedrooms} \quad (1 - 5)$$

If we are to plot the contour of the cost function for this problem, we will get something like this



→ When the contour is like this, gradient descent will take a lot of time before it reaches the global minima

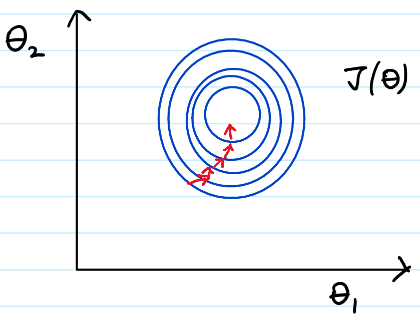
→ All the incremental steps will only do very little and hence a LOT of such steps will be required

→ In these settings, a useful thing to do is to scale the features. So now, the features will be:

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

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

Now, the contours will become much more like circles something like this :



Now gradient descent will find a much more direct path to the global minima rather than taking a huge number of iterative steps

→ More generally, the aim of feature scaling is to get every feature into approximately a range of $\underline{\underline{-1 \le x_i \le 1}}$

↳ The numbers -1 and 1 are not too important, it is this range which should be small and comparable with other features

$$0 \le x_1 \le 3 \quad ✓$$
$$-2 \le x_2 \le 0.5 \quad ✓$$

$$0 \le x_1 \le 3 \quad ✓$$
$$-2 \le x_2 \le 0.5 \quad ✓$$
$$-100 \le x_3 \le 100 \quad ✗ \quad (\text{too large})$$
$$-0.0001 \le x_4 \le 0.0001 \quad ✗ \quad (\text{too small})$$

- MEAN NORMALIZATION

→ Replace x_i with $(x_i - \mu_i)$ to make features have approximately ZERO mean
→ Do not apply on x_0

☆ EXAMPLE

$$x_1 = \frac{\text{size} - 1000}{2000} \quad (\text{Mean size} = 1000 \text{ feet}^2)$$

$$x_2 = \frac{\#\text{bedrooms} - 2}{5} \quad (\text{Mean \# bedrooms} = 2)$$

$$\Rightarrow \quad -0.5 \le x_1 \le 0.5$$
$$-0.5 \le x_2 \le 0.5$$

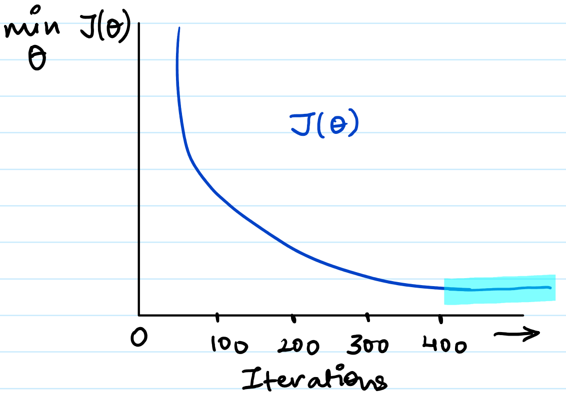
→ As a general rule, if $\mu \rightarrow$ mean and $s \rightarrow$ range (max − min)

$$x_1 \leftarrow \frac{x_1 - \mu_1}{s_1}$$

LEARNING RATE – α

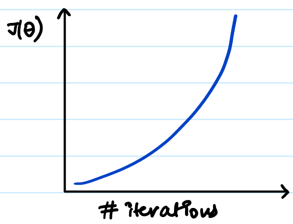
Q How to make sure gradient descent is working?

→ One way to find out if gradient descent is working properly is to plot the cost function against the number of iterations of gradient descent
→ If gradient descent is running properly, the value of the cost function should decrease with each iteration

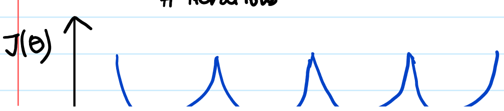


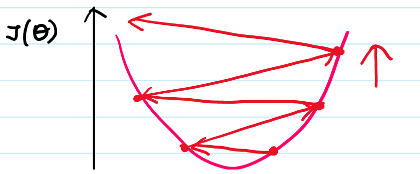
→ Also, if we look after 400 iterations the value of $J(\theta)$ more or less remains the same
→ This suggests that after 400 iterations the cost function converged
→

→ The plot cost vs. # iterations also tells us when the algorithm is not working

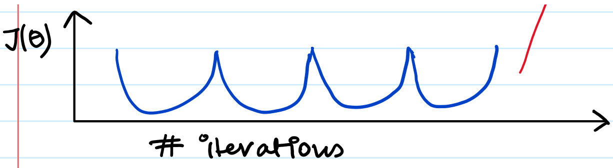


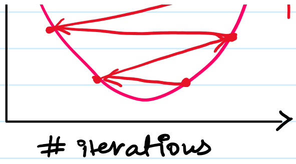
→ This suggests that we might be using larger than required value of the learning rate
→ We need to use a smaller learning rate





→ If the learning rate is high, the algorithm overshoots and gradient descent fails to converge





is high, the algorithm overshoots and gradient descent fails to converge

→ It might even diverge

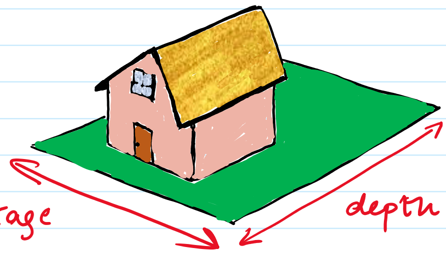
→ For a sufficiently small α, $J(\theta)$ should decrease on every iteration

→ But if α is too small, gradient descent can be slow to converge

FEATURE GENERATION

→ Suppose we have 2 features for a house :

- frontage
- depth



→ When applying linear regression we can have :

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

→ But we don't necessarily have to use the features x_1 and x_2 that we are given

→ What we can do is to create new features from the given ones if we feel they are more appropriate

☆ EXAMPLE

→ If we have frontage and depth, we can come up with a new feature :

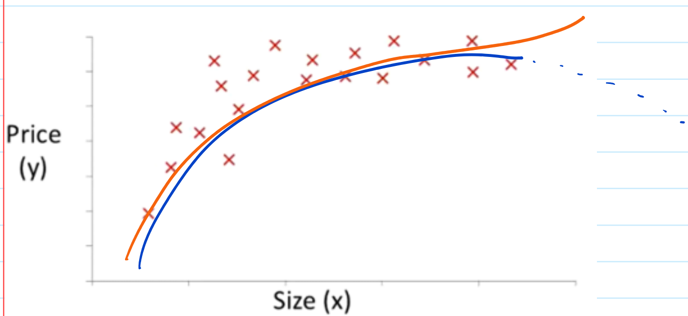
$$\text{Area} = \text{frontage} \times \text{depth}$$

$$\Rightarrow \quad h_\theta(x) = \theta_0 + \theta_1 \times \text{Area}$$

POLYNOMIAL REGRESSION

→ Instead of a linear model, which will not be the most appropriate choice all the time, we can have a polynomial model

→ This will enable us to generate a line which better fits the data



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

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