Multiple Features

x_1 size (ft^2)	x_2 # of bed	x_3 # of bath	x_4 Age	y Price ($1000's)
—				—
—				—
—				—
—				—
—				—
—				—

1) One feature: $f_{w,b} = wx+b$

2) Multiple features:

$x_j = j^{th}$ feature

$\vec{x}^{(i)}$ = features of i^{th} training sample $\Rightarrow$ e.g. $\vec{x}^{(3)} = [2000 \;\; 3 \;\; 2 \;\; 30 \;\; 500]$

$x_j^{(i)}$ = value of j^{th} feature in i^{th} training sample $\Rightarrow x_4^{(3)} = 30$

$$f_{x,b}(\vec{x}) = w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4 + b$$

e.g.: $$f_{x,b}(\vec{x}) = \underset{\text{size}}{0.1\, x_1} + \underset{\text{\#bed}}{4\, x_2} + \underset{\text{\#bath}}{3\, x_3} + \underset{\text{age}}{(-2)\, x_4} + \underset{\text{base price}}{80}$$

More General:

$$f_{\vec{w},b}(\vec{x}) = w_1 x_1 + w_2 x_2 + \dots + w_n x_n + b$$

$\vec{w} = [w_1 \;\; w_2 \dots w_n]$ ← model parameters

b = number

$\vec{x} = [x_1 \;\; x_2 \dots x_n]$

more compact in matrix/vector (linear algebra) expression:

$$f_{\vec{w},b} = \vec{w} \cdot \vec{x} + b$$

dot product: $\vec{w} \cdot \vec{x} = \sum_j w_i x_i$

$$= \sum_i w_i x_i + b$$

"multiple linear regression" not "multivariate regression"

Vectorization: (code)

1) w/o vectorization / loop: $f = w[0]x[0] + w[1]x[1] + w[2]x[2] + w[3]x[3] + b$

$\Rightarrow$ cumbersome

2) Loop:

```
f = b
for j in range(0,n):
    f = f + w[j] * x[j]
```

$\Leftarrow$ n steps to compute

3) Vectorized: `f = np.dot(w,x) + b` $\Leftarrow$ 1 step! (parallelized)

In Gradient descent we have

$$\underbrace{\vec{w}}_{\text{1 op}} = \vec{w} - \underbrace{\alpha \vec{d}}_{\text{1 op}}$$

//

Gradient Descent for multiple Linear Regression:

Vector notation:

$$f_{\vec{w},b}(\vec{x}) = \vec{w} \cdot \vec{x} + b$$

$$J(\vec{w},b) = \frac{1}{2m}\sum_{i=1}^{i=m}\left[\left(\vec{w}\cdot\vec{x}^{(i)} + b\right) - y^{(i)}\right]^2$$

update rule:

$$\Rightarrow \quad w_j = w_j - \alpha \frac{\partial}{\partial w_j} J(\vec{w},b)$$

$$b = b - \alpha \frac{\partial}{\partial b} J(\vec{w},b)$$

$$\Rightarrow \quad \vec{w} = \vec{w} - \alpha \frac{1}{m}\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)\vec{x}^{(i)}$$

$$b = b - \alpha \frac{1}{m}\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)$$

simultaneous update

Alternative to gradient descent (for Linear Regression Only)

- Solve for w, b w/o iterations (symbolically) $\Rightarrow \vec{\theta} = (X^T \cdot X)^{-1} X^T \cdot y$
- Slow when large # of features
- May be used in some ML packages
- GD is recommended.

Feature Scaling

Example: $\hat{y} = w_1 x_1 + w_2 x_2 + b$

price — $\hat{y}$

size — x_1: ~200-5000 range $\Rightarrow$ big

bed — x_2: 1-5 range $\Rightarrow$ small

Example data point

(2000, 5, 500)

size, # br, price

If $w_1 = 50 \quad w_2 = 0.1 \quad b = 50$

$\hat{y} = \underbrace{50 \cdot 2000}_{\$100{,}000k} + 0.1 \cdot 5 + 50$

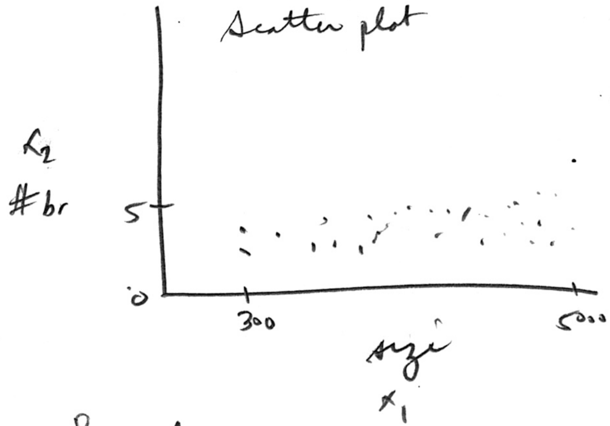
\$0.5k + \$50k = \$1,000,050.5 $\Rightarrow$ too big

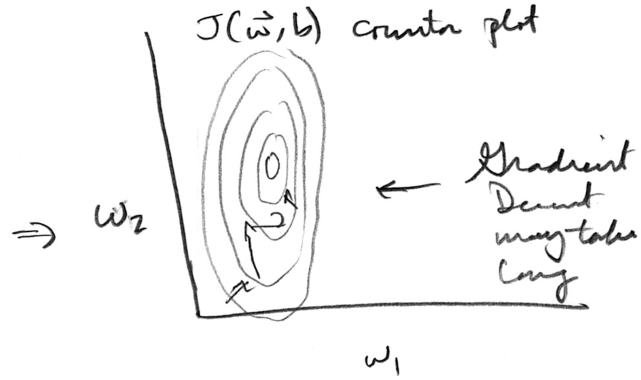
If $w = 0.1 \quad w_2 = 50 \quad b = 50$

$\hat{y} = 0.1 \cdot 2000k + 50 \cdot 5 + 50$

200k 250k \$50k = \$500k

$\Rightarrow$ small w for large x range

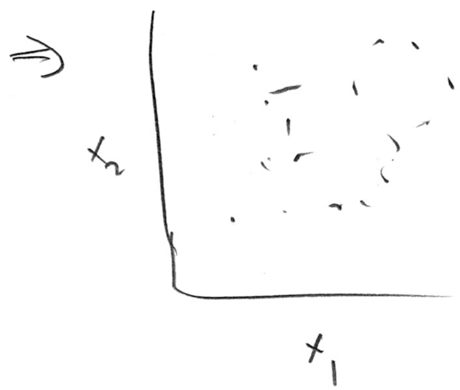


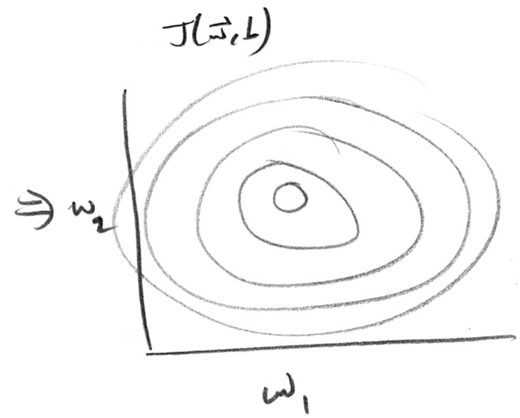


small change in w_1

⇒ Large change in price

Rescale





//

- Feature Scaling (max normalization)

$300 \le x_1 \le 5000$ $\qquad$ $1 \le x_2 \le 5$

$\Rightarrow x_{1,\text{scaled}} = \dfrac{x_1}{5000}$ $\qquad$ $x_{2,\text{scaled}} = \dfrac{x_2}{5}$

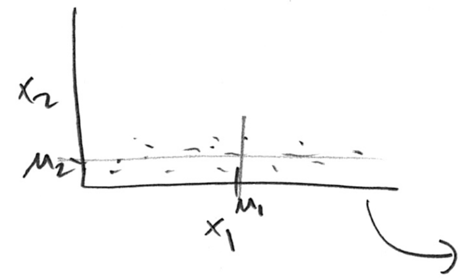
$\Rightarrow 0.15 \le x_{1,\text{scaled}} \le 1$ $\qquad$ $.2 \le x_{2,\text{scaled}} \le 1$

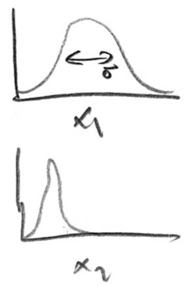
- Mean normalization $\qquad$ compute mean μ_1, μ_2 of x_1, x_2

$$x_{1,s} = \frac{x_1 - \mu_1}{\underset{\text{max}}{5000} - \underset{\text{min}}{300}} \qquad x_2 = \frac{x_2 - \mu_2}{5 - 1}$$

$\Rightarrow -0.18 \le x_1 \le 0.82$ $\qquad$ $-0.46 \le x_2 \le 0.54$

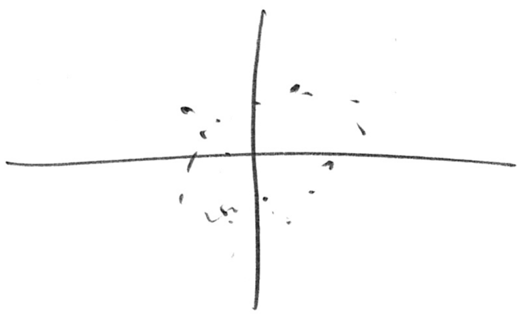
Z-Scale Normalization





Compute the standard deviation σ_1, σ_2 of x_1, x_2.

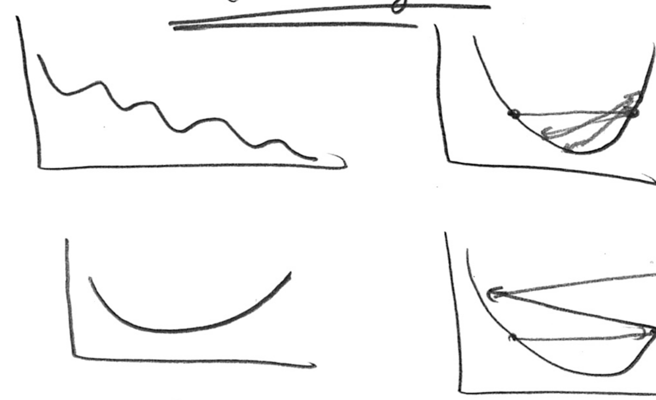
$\Downarrow$



$$\Rightarrow x_{1,s} = \frac{x_1 - \mu_1}{\sigma_1}$$

$$x_{2,s} = \frac{x_2 - \mu_2}{\sigma_2}$$

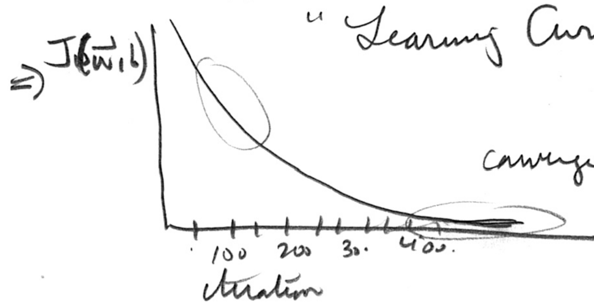
α too large



Gradient Descent Convergence:

$$w_{j+1} = w_j - \alpha \frac{\partial}{\partial w_j} J(\vec{w}, b)$$

$$b_{j+1} = b - \alpha \frac{\partial}{\partial b} J(\vec{w}, b)$$



$\Rightarrow$ "Learning Curve"

converged $\Rightarrow$ by eye!

$\Rightarrow$ try different values of α 10^{-6} – 10^{-1}

Automatic (not perfect)

Compute $\epsilon = w_j - w_{j-1}$

Converge when ϵ small e.g. 0.001

("declare victory done")

Feature Engineering

Example x_1 length, x_2 width of property

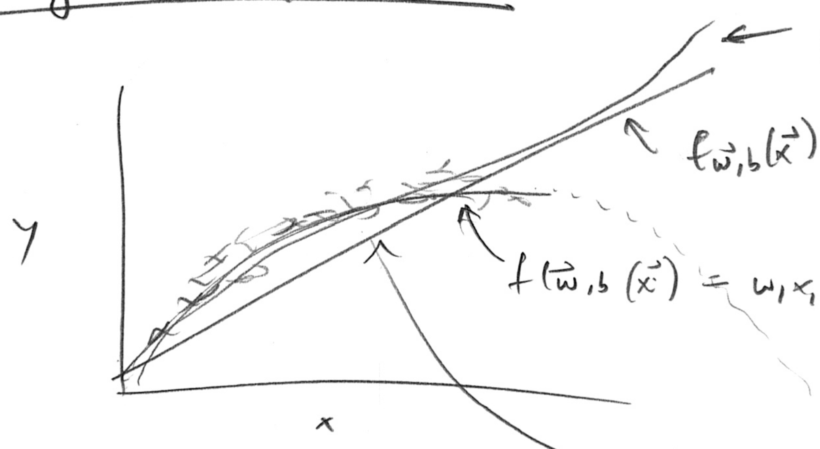
$$\Rightarrow \quad f_{\vec{w},b}(\vec{x}) = w_1 x_1 + w_2 x_2 + b$$

But price more likely go as area $= x_1 \cdot x_2 \leftarrow$ new feature "x_3"

$$\Rightarrow \quad f_{\vec{w},b}(\vec{x}) = w_1 x_1 + w_2 x_2 + \underset{\uparrow\, x_1 x_2}{w_3 x_3} + b$$

Feature Engineering: use intuition to design new features (transform / combine other features)

Polynomial Regression



$\leftarrow f_{w,b}(\vec{x}) = w_1 x + w_2 x^2 + w_3 x^3 + b$

$f_{\vec{w},b}(\vec{x}) = w_1 x_1 + b$

$f(\vec{w},b\ (\vec{x}) = w_1 x_1 + w_2 x_1^2 + b$

$f_{\vec{w},b}(x) = w_1 x + w_2 \sqrt{x} + b$