Multiple Features (variables) — multiple linear regression

	size	Number of Bedrooms	Number of floors	Age of home in years	Price ($) in $1000
	x_1	x_2	x_3	x_4	
$i=1$	2000	5	1	45	460
$i=2$	1000	3	2	20	500
$i=3$	800	2	2	30	350
$i=4$	600	1	1	10	700
$i=n$	...	...	...	...	...

$j = 1 \dots 4$

$n = 4$

x_j = j^{th} feature

n = number of features

$\vec{x}^{(i)}$ = features of i^{th} training example

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

vector

$\vec{x}^{(2)} = [1000, 3, 2, 20]$

$\vec{x}_3^{(2)} = 2$

Model

Previously: $f_{w,b}(x) = wx + b$

$$f_{w,b}(x) = w_1 x_1 + w_2 x_2 + w_3 x_3 + b$$

Example

$$f_{w,b}(x) = 0.1 x_1 + 4 x_2 + 10 x_3 + -2 x_4 + 100$$

x_1: size; x_2: # Bedrooms; x_3: # floors; x_4: years; 100: base price

The definition of the model with n features

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

$\vec{w} = [w_1 \; w_2 \; w_3 \ldots w_n]$

b = is a number

parameters of the model

$\vec{x} = [x_1 \; x_2 \; x_3 \ldots x_n]$

Then

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

$\cdot$ ↑ Dot product

Vectorization in python:

```
f = np.dot(w,x) + b
```

Gradient descent for multiple linear regression

One feature

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

$$\frac{\partial}{\partial w} J(w,b)$$

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

Simultaneously update w, b

n features ($n > 2$)

j = 1

$$w_1 = w_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}) x_1^{(i)}$$

$$\frac{\partial}{\partial w_1} J(\vec{w},b)$$

j = n

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

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

Simultaneously update w_j (for $j = 1 \ldots n$) and b