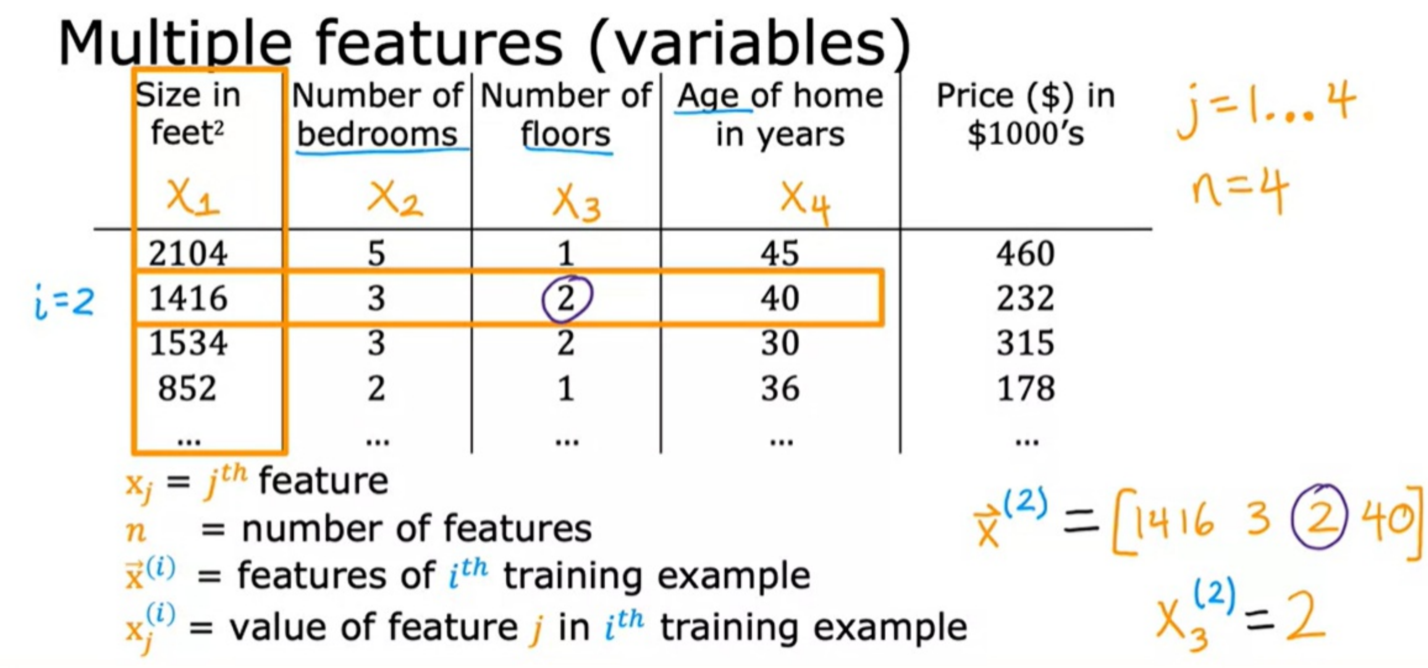
Week 2

- Multiple Linear Regression
  - Multiple Features
    - Notation
    - Vectorisation vs No Vectorisation
    - Gradient Descent with Multiple Variables
  - Feature Scaling
    - Scaling
    - Mean Normalization
    - Z-Score Normalization
    - General Rule of Thumb for Feature Scaling
  - Feature Engineering
  - Polynomial Regression

Multiple Linear Regression

Multiple Features



i: row

j: column

→: Optional, used to indicate a vector

Notation

$$f_{\vec{w},b}(\vec{x}) = w_1x_1 + w_2x_2 + \cdots + w_nx_n + b$$

$\vec{w} = [w_1 \ w_2 \ w_3 \ \ldots \ w_n]$ parameters of the model

b is a number

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

$$f_{\vec{w},b}(\vec{x}) = \vec{w} \cdot \vec{x} + b = w_1x_1 + w_2x_2 + w_3x_3 + \cdots + w_nx_n + b$$

dot product

multiple linear regression

Vectorisation vs No Vectorisation

Parameters and features

$\vec{w} = [w_1 \quad w_2 \quad w_3]$ $n=3$

b is a number

$\vec{x} = [x_1 \quad x_2 \quad x_3]$

NumPy

linear algebra: count from 1

w[0] w[1] w[2]

```
w = np.array([1.0,2.5,-3.3])
b = 4                    x[0] x[1] x[2]
x = np.array([10,20,30])
```

code: count from 0

Without vectorization $n = 100{,}000$

$$f_{\vec{w},b}(\vec{x}) = w_1x_1 + w_2x_2 + w_3x_3 + b$$

```
f = w[0] * x[0] +
    w[1] * x[1] +
    w[2] * x[2] + b
```

Without vectorization

$$f_{\vec{w},b}(\vec{x}) = \left(\sum_{j=1}^{n} w_jx_j\right) + b$$

$\sum_{j=1}^{n} \rightarrow j = 1 \ldots n$

1, 2, 3

range(0,n) → j = 0…n-1

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

Vectorization

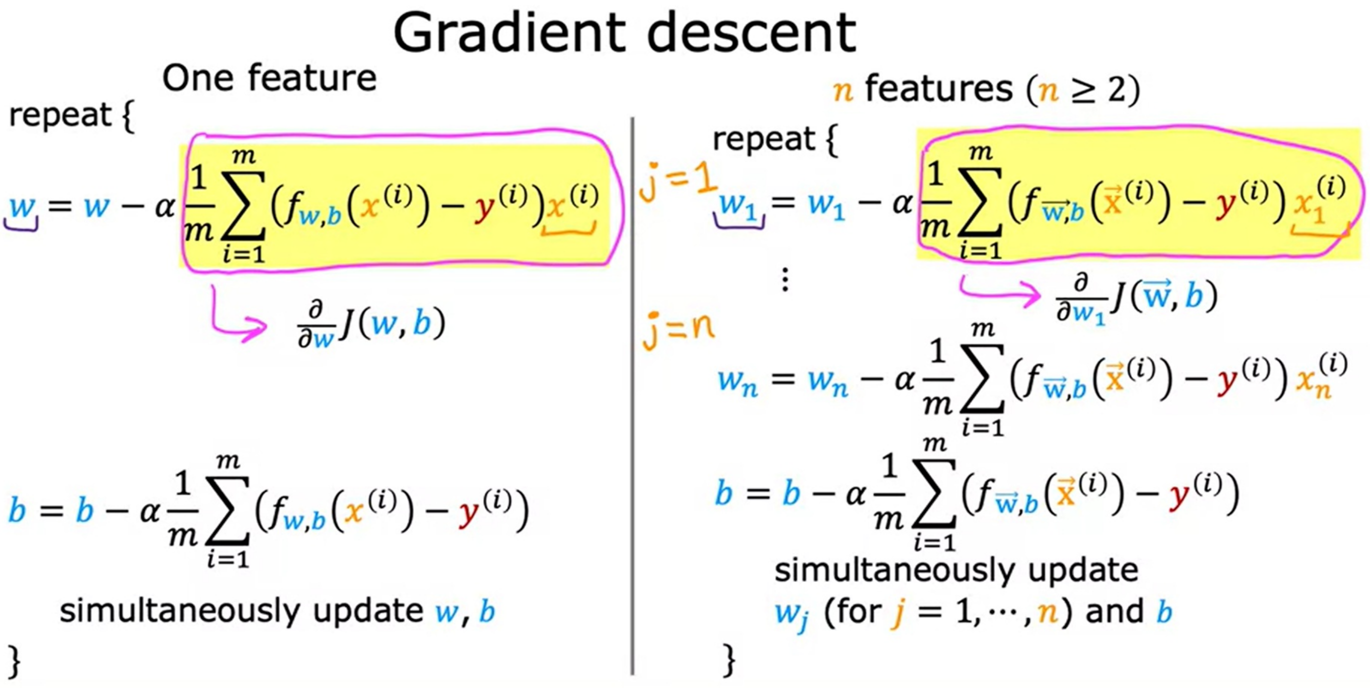
$$f_{\vec{w},b}(\vec{x}) = \vec{w} \cdot \vec{x} + b$$

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

NB: Vectorisation uses parallelism

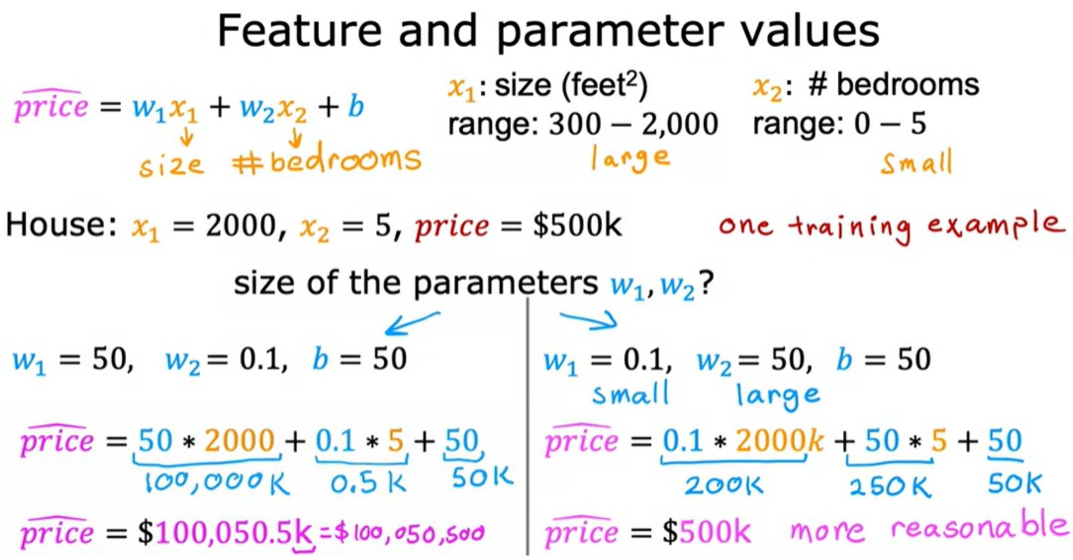
Gradient Descent with Multiple Variables

- One vs multiple
- For multiple, you need to update every w



Feature Scaling

- How to choose your w's?
- big x_i, small w_i, vice versa.

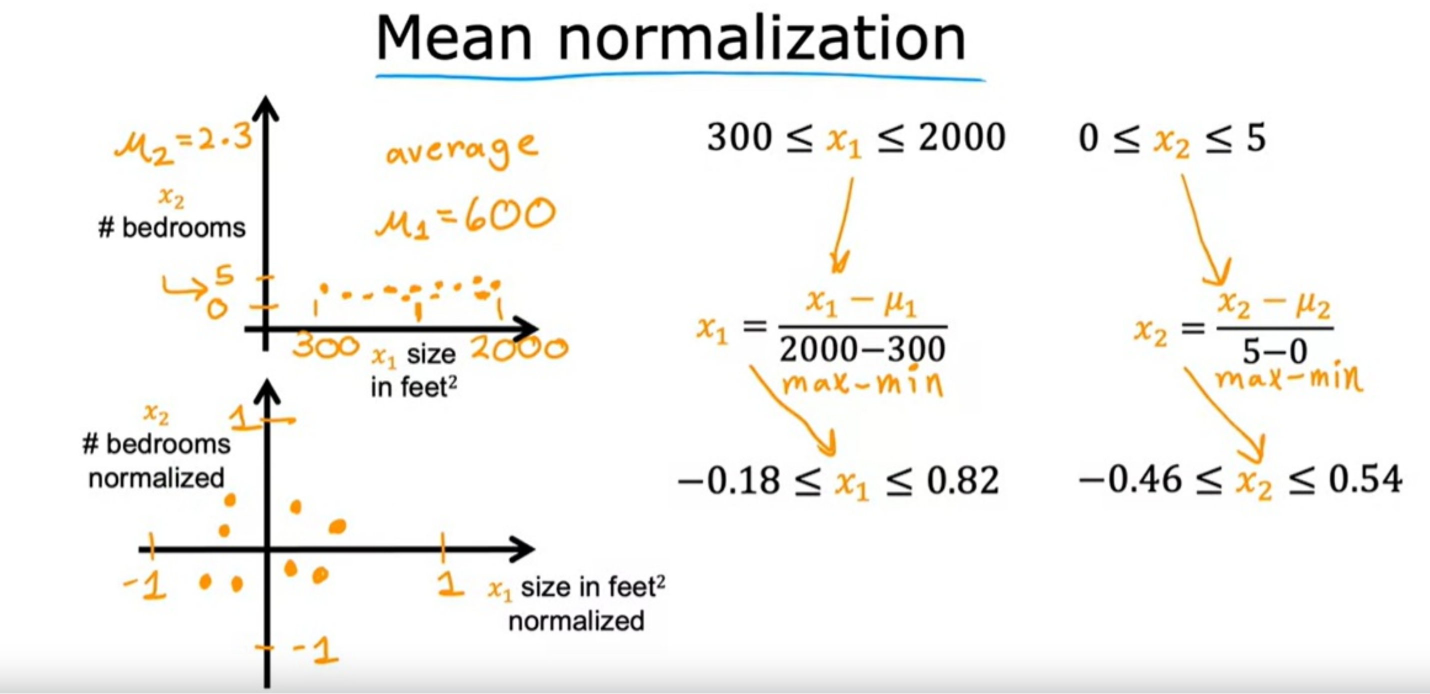


Scaling

- Divide by the max range

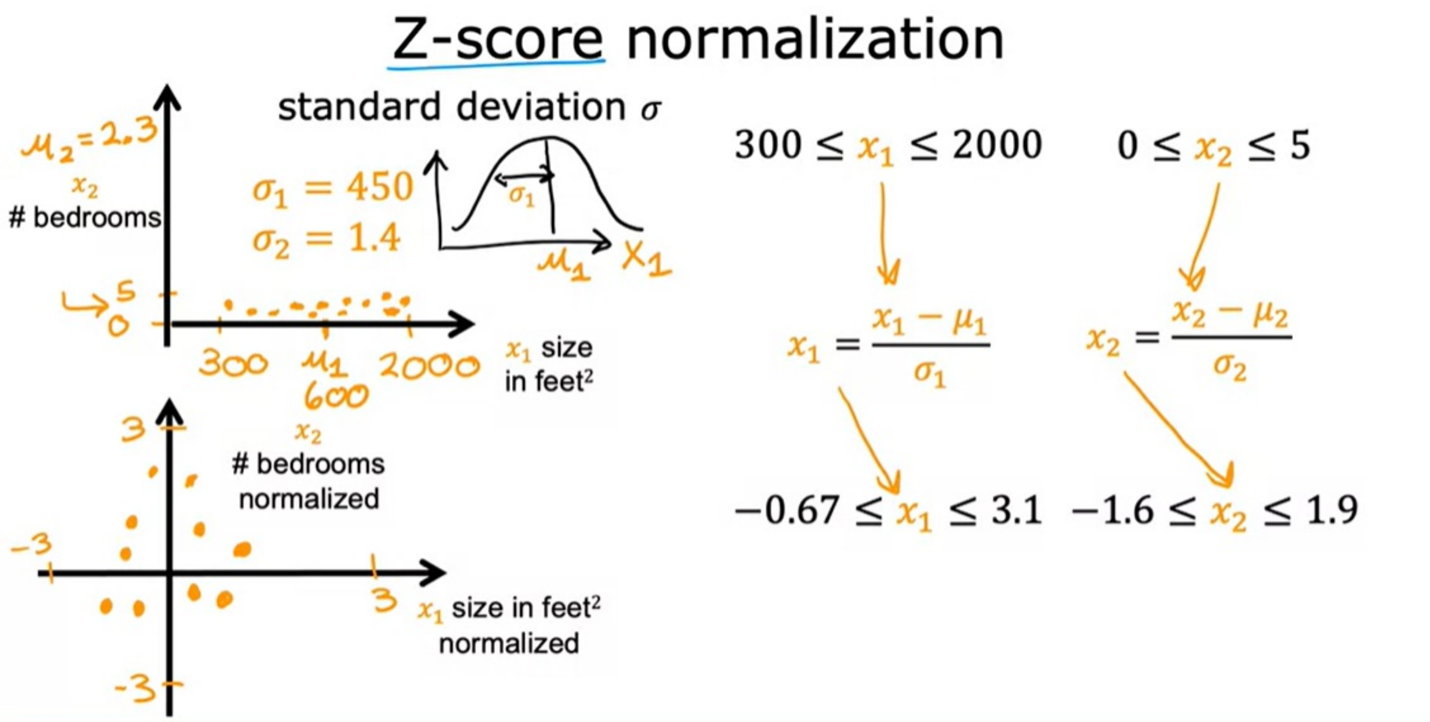
Mean Normalization

- Minus the avg, divide by max - min



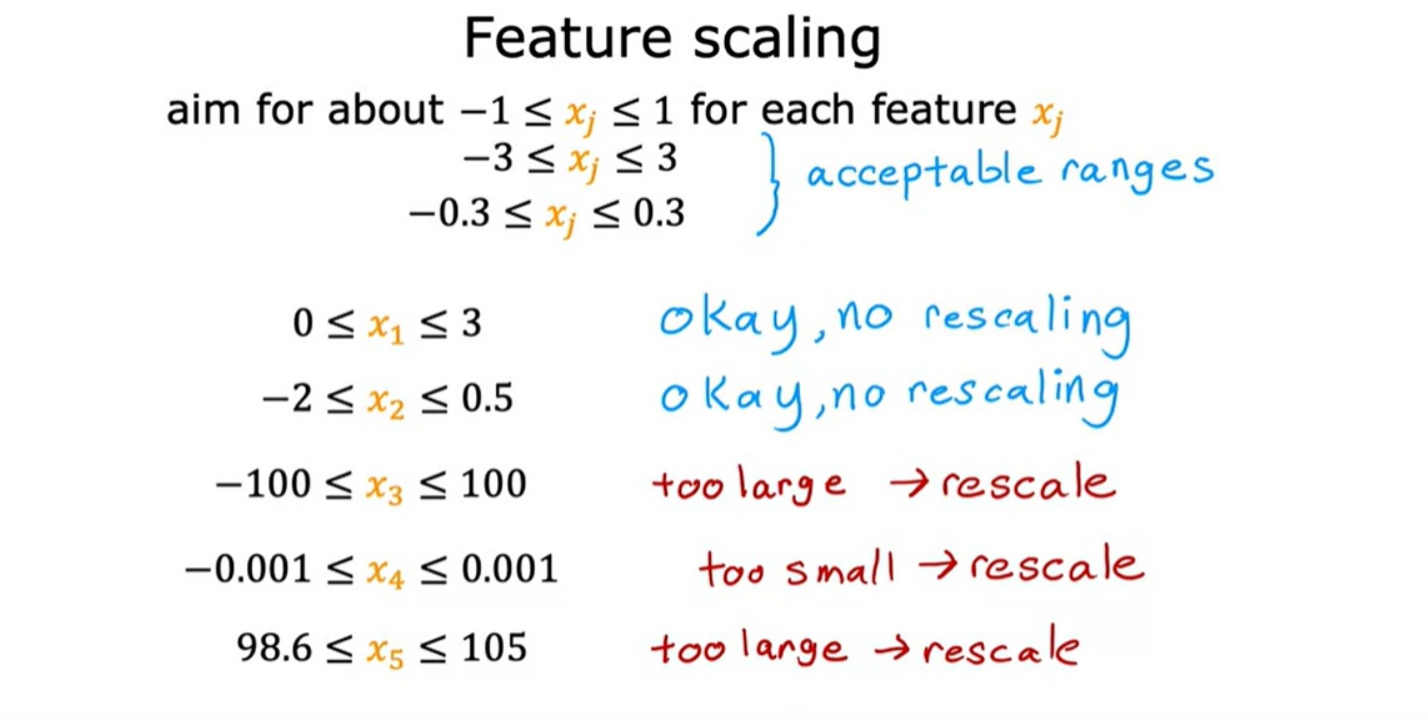
Z-Score Normalization

- Minus mean divide by standard deviation

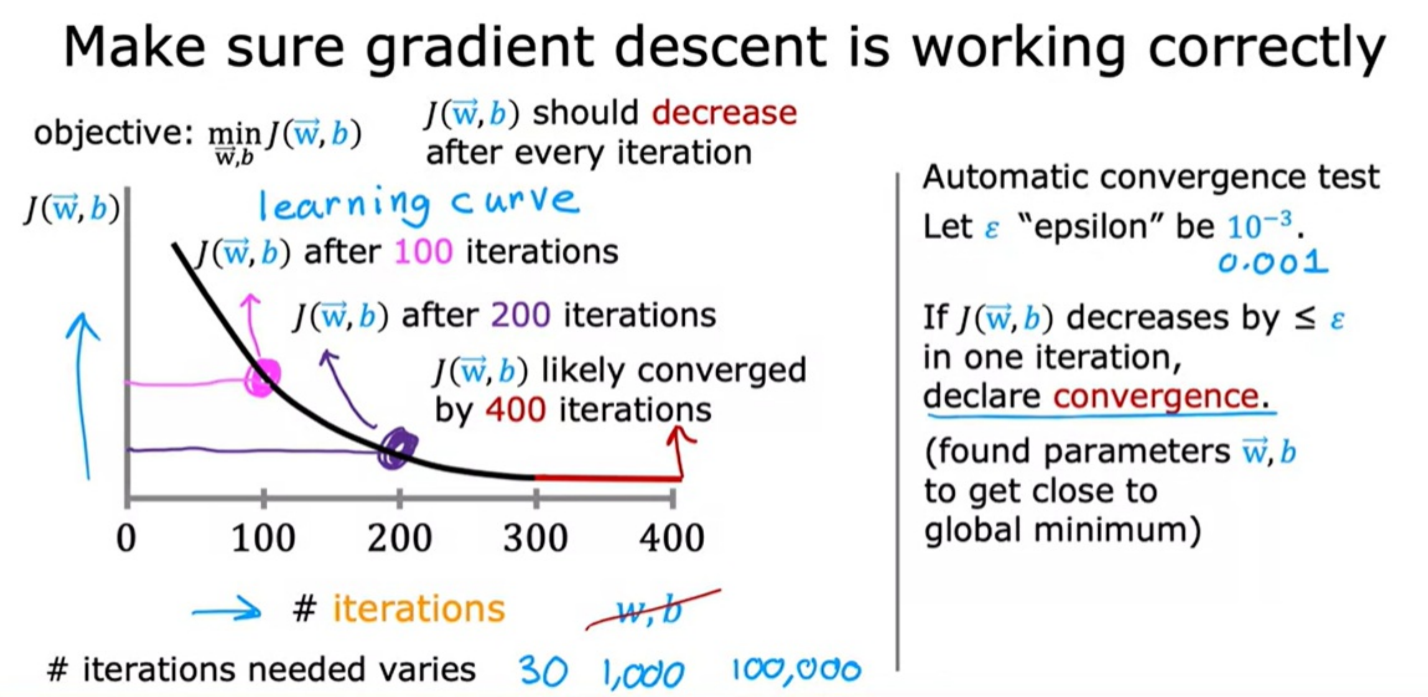


General Rule of Thumb for Feature Scaling

- Feature scaling helps to improve gradient descent speeds!



- You can also plot a graph, and ensure function J does not increase at any iterations
- Helps to ensure model is done correctly
- Always try a range of values for **α**



Feature Engineering

- Defining a new feature by combining existing features which can be a better fit for the model.

Feature engineering

$$f_{\vec{w},b}(\vec{x}) = w_1 \underbrace{x_1}_{\text{frontage}} + w_2 \underbrace{x_2}_{\text{depth}} + b$$

$$area = frontage \times depth$$

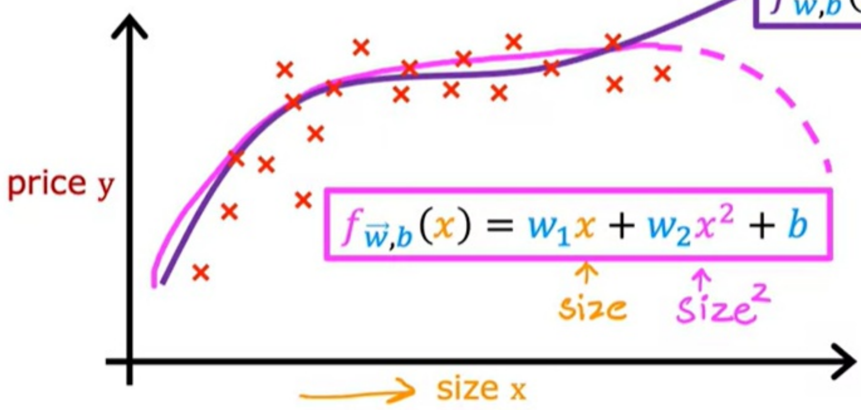
$$\underbrace{x_3 = x_1 x_2}_{\text{new feature}}$$

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

Feature engineering:
Using intuition to design new features, by transforming or combining original features.

Polynomial Regression

Polynomial regression



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

size	size²	size³
$1-10^3$	$1-10^6$	$1-10^9$

feature scaling