Task_28

Linear Regression with multiple variables

"Multiple features"

x_1	x_2	x_3	x_4	y
1416	3	2	40	232

$x_j \to j^{th}$ feature

$n \to$ number of feature

$\vec{x}^{(i)} \to$ feature of i^{th} in exam

$x_j^{(i)} \to$ value of featn j in i^{th} inex

vector $\to \vec{x}^{(2)} = [1416 \ 3 \ 2 \ 40]$

$x_3^{(2)} = 2$

Model $\to$ for the Pervieviously $\to f_{w,b}(x) = wx + b \to$ for one variable

for 4 variables $\to f_{w,b}(x) = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b$

$f_{w,b}(x) = w_1x_1 + w_2x_2 + \dots + w_nx_n + b \to$ for n feature

$\vec{w} = [w_1 \ w_2 \ w_3 \ \dots \ w_n] \to$ Parameters of the Model

b is a number

$\to$ Multiple linear regression

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

Dot Product

$\vec{w}, b, \vec{x} \to$ by Code

$w = \boxed{np.array}[- \ - \ -]$

$b = -$

$\vec{x} = \boxed{np.array}[- \ - \ -]$

$\to$ linear algebra: Count from 1

$\to$ code: " = 0

for n variables: without vectorization

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

" " " with "

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

Code $\to f = \boxed{np.dot(w,x)} + b$

Code $\to$

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

an alternative to gradient decent

Normal equation

- only for linear regression
- Solve for w, b without iterations

Disadvantages

- Doesn't generalize to other learning algorithms
- Slow when nu. of fea. > 2000

$$x_{scaled} = \frac{x}{\max \text{ of } x}$$

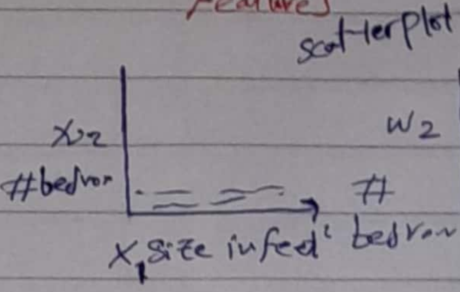
what you need to know

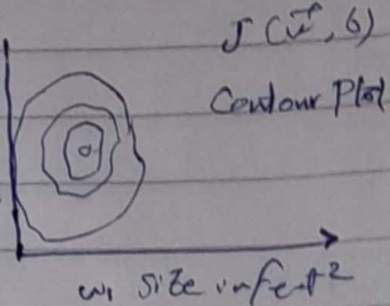
- Normal equation Method may be used in machine learning libraries that implement linear regression
- Graident descent is the recommended method for finding Param w, b

Features   Parameters

Feature size and Parameters size

	Size of Feature x_j	Size of Parameters w_j
size in feet^2	⟷	⟷
#bedrooms	⟷	⟷





→ $x_{scaled} = \frac{x}{\max \text{ of } x}$ → Mean normalization → $x_1 = \frac{x_1 - \mu_1}{\max - \min}$

→ Zscore normalization   , σ: stander deviation → $x_1 = \frac{x_1 - \mu_1}{\sigma_1}$

Feature Scaling

→ aim for about $-1 \le x_j \le 1$ for each feature x_j

$-1 : 1$
$-3 : 3$
$-0.3 : 0.3$ } acceptable range

$-100 : 100$ → rescale, too large

$-0.001 : 0.001$ → resc.., too small

Automatic Convergence Test

let $\varepsilon = 0.001$

If $J(\vec{w}, b)$ decreases by $\le \varepsilon$ in one iteration declare Convergence

found Parameters $\vec{w}, b$ to get close to global minimum.

Feature engineering

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

(x_1: Frontage, x_2: width)

→ $x_3 = x_1 x_2$ = area → new feature

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

→ using intuition to design new features, by transforming or Combining orginal features

Polynomial regression

$$f_{\vec{w},b}(\vec{x}) = w_1 \underset{\text{size}}{x} + w_2 \underset{\text{size}^2 \text{ "area"}}{x^2} + w_3 \underset{\text{size}^3 \text{ "volume"}}{x^3}$$