Page No.
Date

Week 2: Regression with multiple input variables.

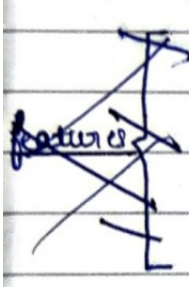
⇒ Multiple linear Regression

Video 1: Multiple features

Earlier we had seen only one input feature (size) corresponding to output $\hat{y}$ (price)

Now, suppose there are multiple features example:

features →	size in feet² x_1	no. of bedrooms x_2	no. of floors x_3	age of home x_4	price
	204	5	1	45	460
$\vec{x}^{(2)}$	1416	3	2	40	283
	1534	3	2	30	315
	852	2	1	30	178



$X_1 \dots X_4$ – features

- n = no. of features = 4

x_j = j^{th} feature

$x^{(i)}$ ⇒ features of i^{th} training example

$\vec{x}_3^{\,2}$ = no. of floors in second training example

Page No.
Date

Previously,

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

$$f_{w,b}(\vec{x}) = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b$$

example:

$$f_{(w,b)}(\vec{x}) = 0.1x_1 + 4x_2 + 10x_3 + 2x_4 + 80$$

80 ↑ base price

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

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

b = number

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

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

Video 2: Vectorisation - Part (I)

Non Vectorised Implementation:

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

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

Page No.
Date

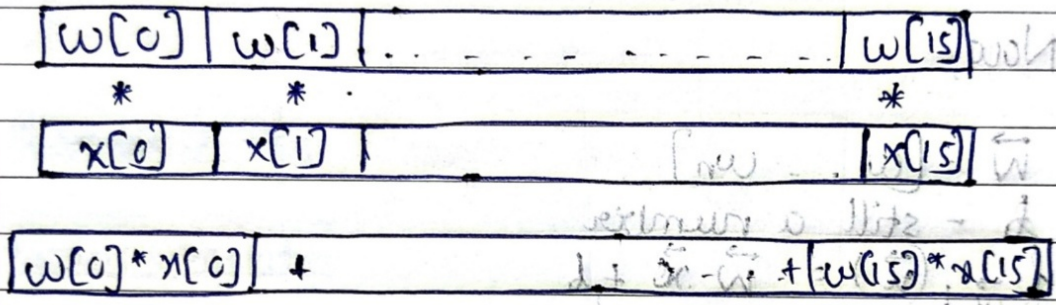
Vectorisation

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

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

Video 3: Vectorisation - Part (II)

$np.dot(w, x)$



Gradient Descent

$\vec{w} = (w_1 \; w_2 \; \ldots \; w_{16})$
$\vec{d} = (d_1 \; d_2 \; \ldots \; d_{16})$

$w = np.array([0.5, 1.3, \ldots, 3.4])$
$d = np.array([0.3, 0.2, \ldots, 0.4])$

compute: $w_j = w_j - 0.1 d_j$

0.1 ↑ learning rate

Page No.
Date

Without vectorisation

$w_1 = w_1 - 0.1 d_1$
$w_2 = w_2 - 0.1 d_2$
⋮
$w_{16} = w_{16} - 0.1 d_{16}$

With vectorisation

$\vec{w} = \vec{w} - 0.1 \vec{d}$

Video 4: Gradient Descent for multiple linear regression

Now,

$\vec{w} = [w_1 \dots w_n]$
b - still a number
$f_{\vec{w},b}(\vec{x}) = \vec{w} \cdot \vec{x} + b$

$J(\vec{w}, b)$

Gradient Descent:

repeat {

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

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

for n features

Repeat {

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

$\vdots$

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

Same for b

Alternative for gradient descent (complicated)

Normal equation:

- Only for linear regression
- Solve for w, b without iteration

Disadvantages

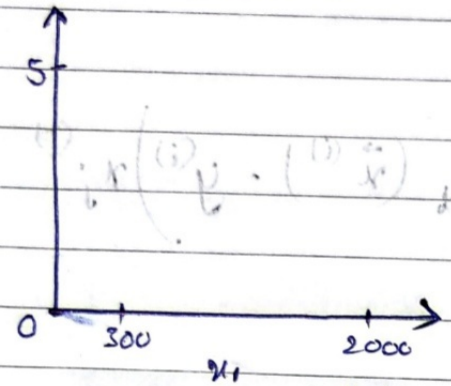
- Slow when number of features ($> 10{,}000$)
- doesn't generalise to other learning algorithms

Page No.
Date

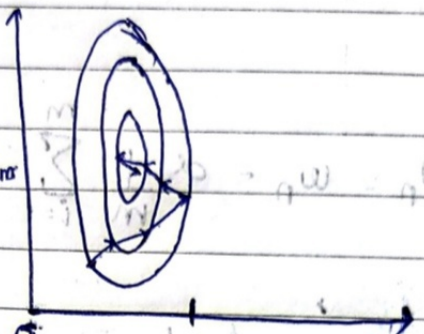
⇒ Gradient Descent in Practice

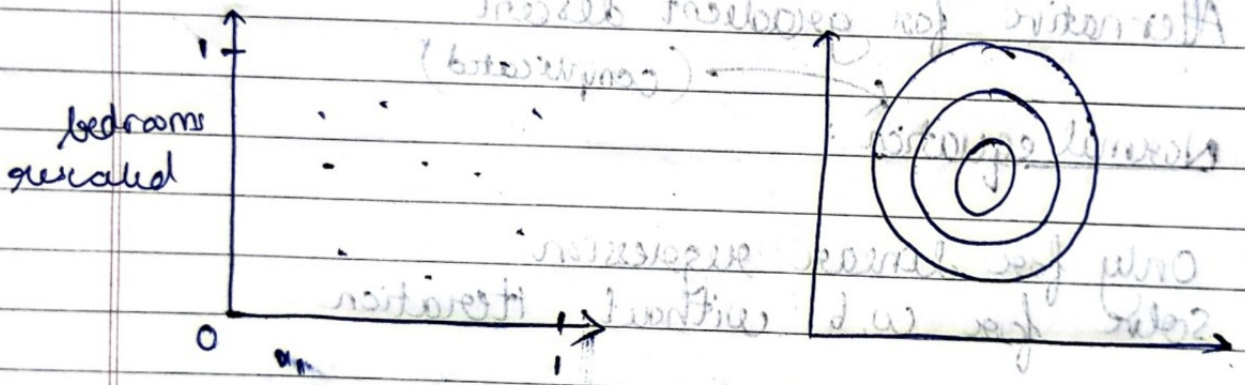
Video 1: Feature Scaling Part (I)

Features



Parameters





Video 2: Feature Scaling - Part (II)

$300 \leq x_1 \leq 2000$

$0 \leq x_2 \leq 5$

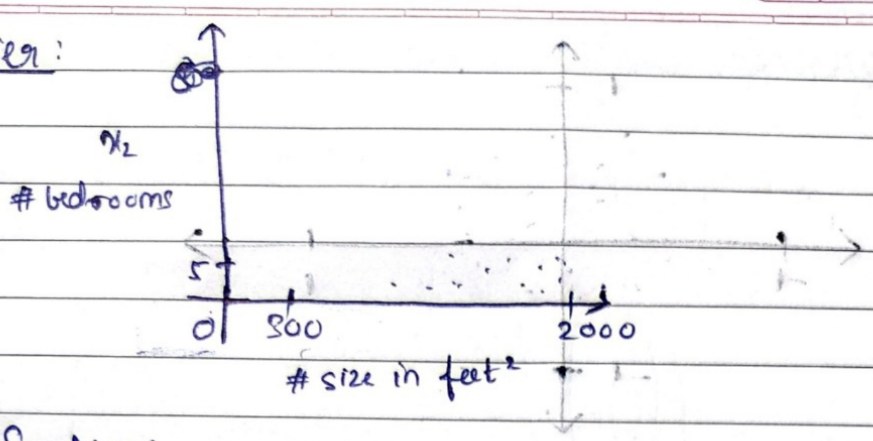
$x_{1\,\text{scaled}} = \frac{x_1}{2000}$

$x_{2\,\text{scaled}} = \frac{x_2}{5}$

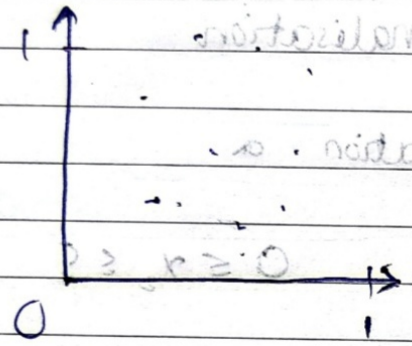
$0.15 \leq x_{1,\,\text{scaled}} \leq 1$

$0 \leq x_{2,\,\text{scaled}} \leq 1$

Earlier:



After Scaling:



Mean Normalisation

Step (1) find mean (average (μ)

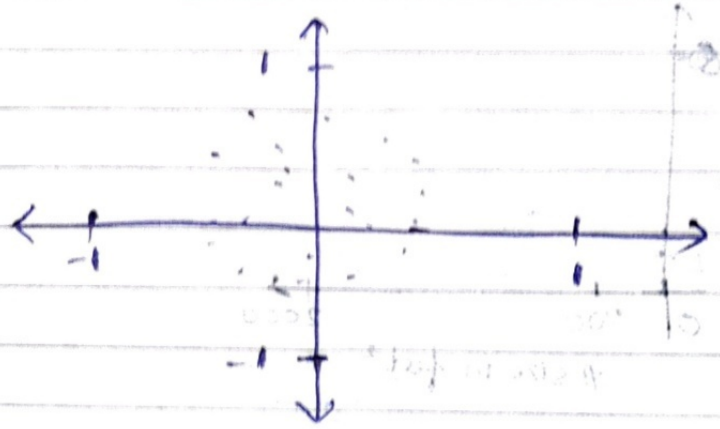
Step (2) divide each term of the feature by (max - min) after subtracting avg.

eg.

$300 \le x_1 \le 2000$ $\qquad$ $0 \le x_2 \le 5$

$x_1 = \frac{x_1 - \mu_1}{2000 - 300}$ $\qquad$ $x_2 = \frac{x_2 - \mu_2}{5 - 0}$

$-0.18 \le x_1 \le 0.82$ $\qquad$ $-0.46 \le x_2 \le 0.5$



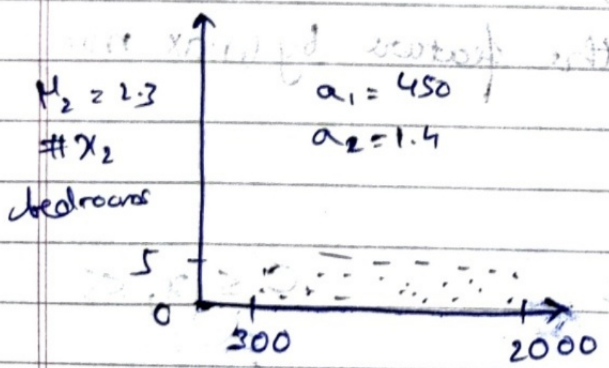
Z-Score Normalisation

Calculate standard deviation of

$300 \leq x_1 \leq 2000$ $\quad$ $0 \leq x_2 \leq 5$

$$x_1 = \frac{x_1 - \mu_1}{\sigma_1} \qquad x_2 = \frac{x_2 - \mu_2}{\sigma_2}$$

$-0.67 \leq x_1 \leq 3.1$ $\quad$ $-1.6 \leq x_2 \leq 1.9$



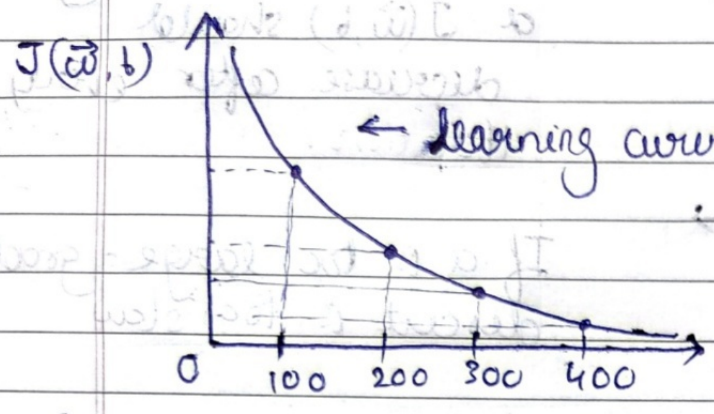
acceptable ranges: $-1 \leq x_j \leq 1$

(examples) $-3 \leq x_j \leq 3$

$-0.3 \leq x_j \leq 0.3$

Page No.
Date

Video 3: Checking Gradient Descent for Convergence.



automatic convergence test

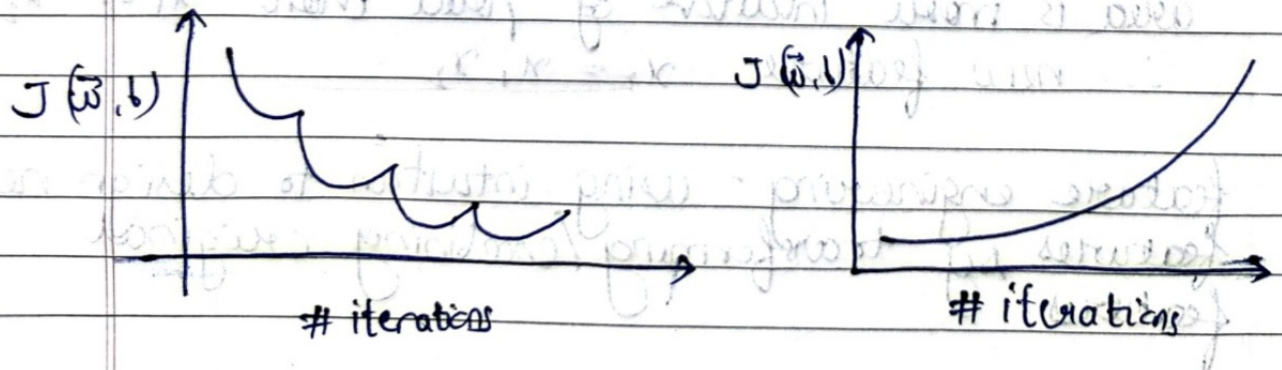
$\varepsilon = 10^{-3}$

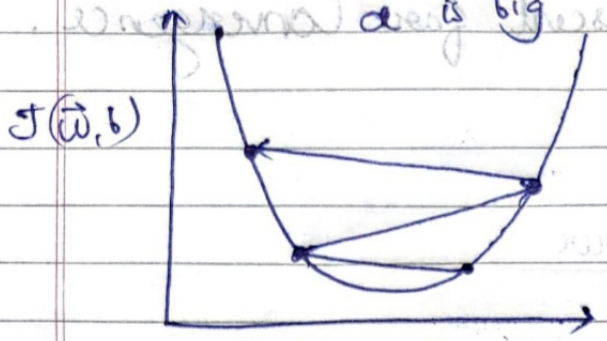
If $J(\vec{w}, b)$ decreases by $\leq \varepsilon$ in one of iteration, declare convergence.

found parameters $(\vec{w}, b)$ to get close to global minimum

Video 4: Choose the learning rate

Identify problem with learning rate





with a small enough α J(w,b) should decrease after every iteration.

If α is too large - gradient descent is too slow

Values of α to try:

0.001 0.01 0.1 1 ...

Video 5: Feature Engineering

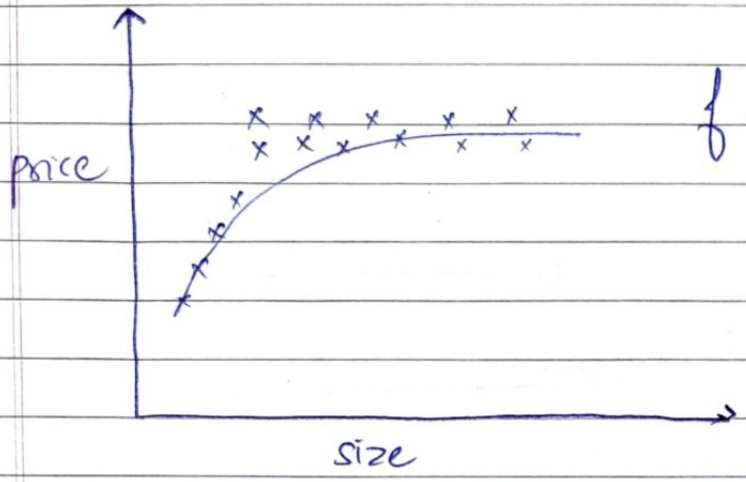
area is more intuitive of price than x_1 or x_2

∴ new feature: $x_3 = x_1 x_2$

feature engineering - using intuition to design new features by transforming/combining original features

Page No.
Date

Video 6: Polynomial Regression



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