Computing Parameters Analytically

Normal Equation

Assume we have a dataset as follows:

Size (ft^2)	Beds	Floors	Age	Price ($\$10^3$)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178

Here, $m = 4$

We'll construct matrices X and Y as follows:

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix}_{[m*(n+1)]} \qquad y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}_{[m*1]}$$

Add an extra column x_0 with all 1's to the matrix X.

Now, we'll compute $\theta = (X^T \cdot X)^{-1} \cdot X^T \cdot y$

For using this in Octave,

```
pinv(X'*X)*X'*y
```

Note: Feature scaling is not necessary in case we're using Normal Equation

Gradient Descent	Normal Equation
Need to choose the learning rate α	No need to choose the learning rate α
Needs many iteration	Doesn't need any iteration
Requires less computation	Requires huge computation of O(n^3) since calculation of $(X^T \cdot X)^{-1}$ is needed
Works well even when n is large (n is the number of features)	Becomes slow if n is large

> **When to use Normal Equation?**

> We can comfortably use normal equation **when the value of ‘n’ (features) is less than 10,000**. More than that, mordern computers will take a lot of time for computation.

What if $X^T X$ is non-invertible?

In rare cases, $X^T X$ might turn out to be non-invertible (singular/degenerate). Our Octave code which we’re using will take care of it (since we’re using **pinv** instead of **inv**).

Causes for this:

- Redundant features (linearly dependent)

  For example, we might have two features like:

  - x_1 = size in $feet^2$
  - x_2 = size in m^2
- Too many features ($m \leq n$)

  m = Number of training examples

  n = Number of features

  Delete some features in this case, or use regularization