LectureNote2: CS229 (SPRING2022, Stanford)

Supervised Learning

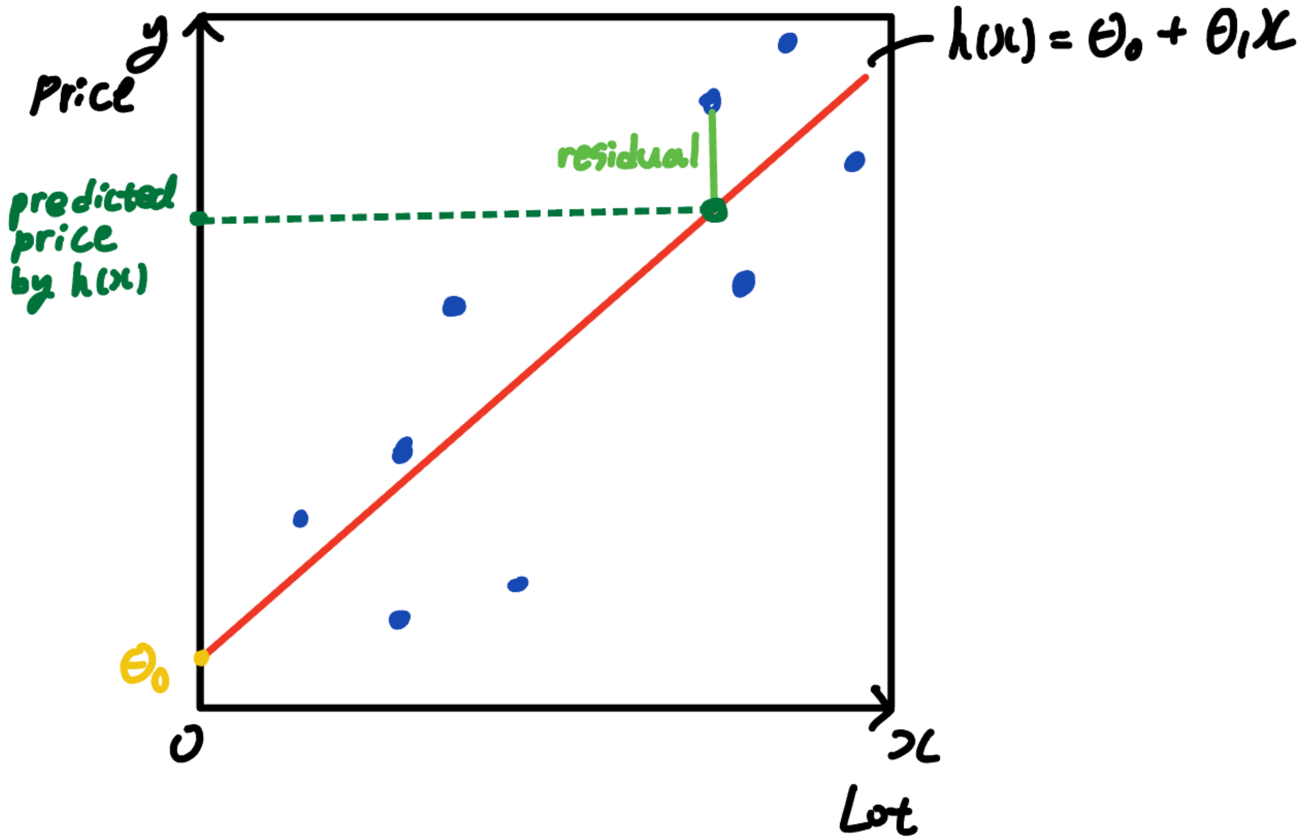
Set Up

- Prediction:
  - $h: \begin{array}{ccc} x & \rightarrow & y \\ images & & cat \\ text & & hate\ speech? \\ housedata & & price \end{array}$

- Given: training set
  - $\{\ (x^{(1)}, y^{(1)}), ..., (x^{(n)}, y^{(n)})\ \}$
  - y = supervision

- Do: find **good** $h(hypothesis) : x \rightarrow y$
  - we care about **new** x's that are not in our training set
  - if y is *discrete* $\rightarrow$ Classification
  - if y is *continuous* $\rightarrow$ Regression

- How do we represent h?
  - $h(x) = \theta_0 + \theta_1 x_1$



- Generalization
  - Training Set

	$size(x_1)$	$bedroom(x_2)$	$lot\ size(x_3)$	$\dots$	$feature(x_d)$	$price(y)$
$x^{(1)}$	2104	4	$45k$	$\dots$	$x_d^{(1)}$	400
$x^{(2)}$	2500	3	$30k$	$\dots$	$x_d^{(2)}$	900
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$x^{(n)}$	$x_1^{(n)}$	$x_2^{(n)}$	$x_3^{(n)}$	$\dots$	$x_d^{(n)}$	$y^{(n)}$

  - $h(x) = \theta_0 x_0 + \theta_1 x_1 + \dots + \theta_d x_d \;=\; \sum_{j=0}^{d} \theta_j x_j \; (x_0 = 1)$
  - $\theta_{parameters} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} \quad x^{(i)}_{features} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ \vdots \\ x_d^{(i)} \end{bmatrix}$
  - $(x^{(i)}, y^{(i)}) \leftarrow$ i-th training example

- $X_{train\ data} = \begin{bmatrix} -x^{(1)}- \\ -x^{(2)}- \\ \vdots \\ -x^{(n)}- \end{bmatrix} \in \mathbb{R}^{n(d+1)}$
  - $X_{train\ data}$ does not contain y's
  - +1 is the extra dimension x_0, whose value is set to be 1 above

- $h_\theta(x) = \sum_{j=0}^{d} \theta_j x_j \ (x_0 = 1)$, I want $h_\theta(x) \simeq y$

- $J(\theta)_{cost\ function} = \frac{1}{2} \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)})^2{}_{least\ squares}$
  - $\frac{1}{2}$ is for convenience purpose (It cancels out with 2 when we take the derivative)
  - we only care about the minimizer, which is the θ that minimizes $J(\theta)$. The "actual value" of $J(\theta)$ is not important.
  - This process is called "Optimization"

- Gradient Descent
  - $\theta^{(0)} = 0$
  - $\theta_j^{(t+1)} = \theta_j^{(t)} - \alpha \ \frac{\partial}{\partial \theta_j} J(\theta^{(t)})$ (α = learning rate, $j = 0 \cdots d$)
  - $\frac{\partial}{\partial \theta_j} J(\theta^{(t)}) = \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)}) \frac{\partial}{\partial \theta_j} h_\theta(x^{(i)})$
  - $h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \ldots + \theta_d x_d, \ \ \frac{\partial}{\partial \theta_j} h_\theta(x^{(i)}) = x_j^{(i)}$
  - $\therefore \theta_j^{(t+1)} = \theta_j^{(t)} - \alpha \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$
    - i-th data point, j-th component
  - for all j's, $\theta^{(t+1)} = \theta^{(t)} - \alpha \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$
    - α can be changed throughout the iteration

- Batch vs Stochastic Minibatch
  - Minibatch: randomly select a small subset B of the training data (B << n)
  - calculate a noisy estimation $\theta^{(t+1)} = \theta^{(t)} - \alpha_B \sum_{i \in B} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$