CS229 Note Lecture 03

Notation

$(x^{(i)}, y^{(i)})$ -> i^{th} training example

$h_\theta(x^{(i)}) = \sum_{j=0}^{n} \theta_j x_j^{(i)} = \theta^T x$

$J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$

$\theta = (X^T X)^{-1} X^T y$

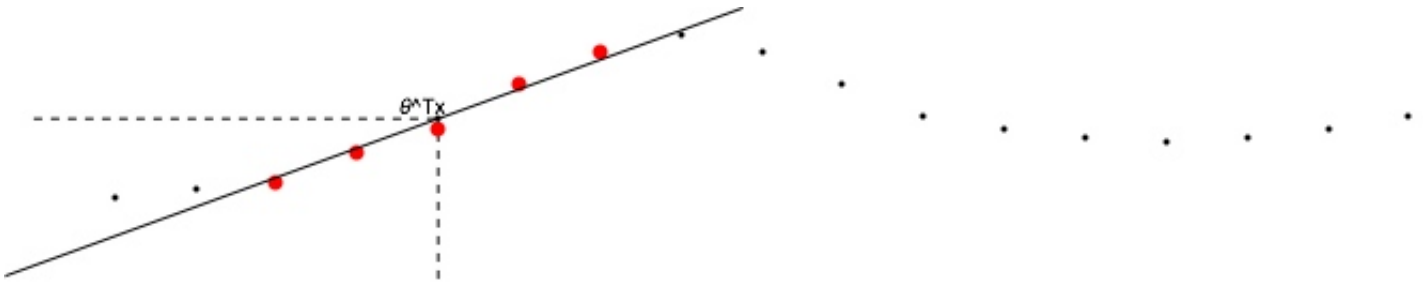
Linear Regression

Underfitting & Overfitting(talk later)

"Paramatric" Learning Algorithm: θ's find set of parameters

"Non-parametric" Learning algorithm: no. of parameters grows with m.

Locally Weighted Regression (Loess)



To evaluate h at a certain x

Linear Regression: Fit θ to minimize $\sum_i \left(y^{(i)} - \theta^T x^{(i)}\right)^2$

Return $\theta^T x$

Apply linear regression to fit a straight line just to the sub-set which in the little vicinity of x.

Take this sub-set and fit a straight line to it.

Locally Weighted Regression: Fit θ to minimize

$$\sum_i w^{(i)}\left(y^{(i)} - \theta^T x^{(i)}\right)^2 \tag{1}$$

where $w^{(i)} = e^{-\frac{\left(x^{(i)}-x\right)^2}{2}}$

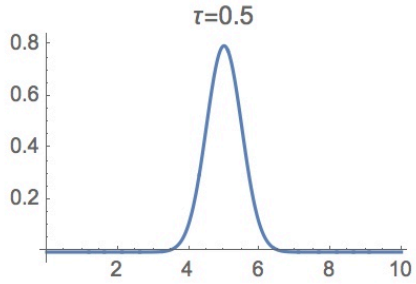
If $|x^{(i)} - x|$ small, then $w^{(i)} \approx 1$

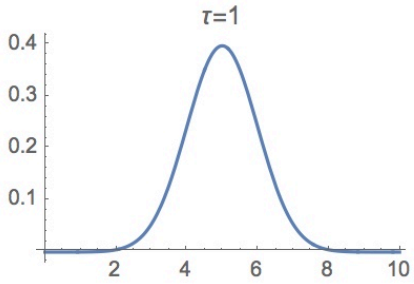
Conversely, if $|x^{(i)} - x|$ large, then $w^{(i)} \approx 0$

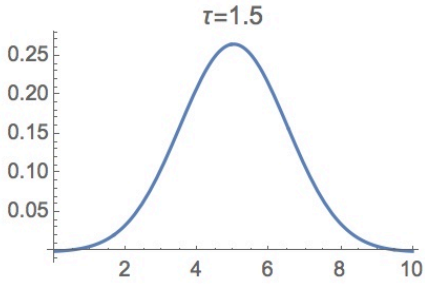
Finding a certain point x, so for the points that are far away, $w^{(i)}$ will be close to 0, and not contribute much all to equation (1).

So the effect of using this weighting is that locally weighted linear regression fits a set of parameters θ, pay much more attention to fitting the points close by accurately, whereas ignoring the contribution from faraway points.

For normally, let $w^{(i)} = e^{-\frac{\left(x^{(i)}-x\right)^2}{2\tau^2}}$. The parameter τ is called the **bandwidth parameter**, it controls how fast the weights fall of with distance.







Probabilistic Interpretation

Why least square?

Assume $y^{(i)} = \theta^T x^{(i)} + \epsilon^{(i)}$, $\epsilon^{(i)}$ is called **error**(**random noise**), and $\epsilon^{(i)} \sim \mathcal{N}\left(0, \sigma^2\right)$.

$$P(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi}\tau} e^{-\frac{(\epsilon^{(i)})^2}{2\tau^2}}$$

From the **Central Limit Theorem**, we can derive y from the gaussion distribution

$$y^{(i)} \mid x^{(i)}; \theta \sim \mathcal{N}\left(\theta, \sigma^2\right)$$

so we get (use ; to denote θ is not a random variable)

$$P\left(y^{(i)} \mid x^{(i)}; \theta\right) = \frac{1}{\sqrt{2\pi}\tau} e^{-\frac{\left(y^{(i)}-\theta^T x^{(i)}\right)^2}{2\tau^2}}$$

Assume $\epsilon^{(i)}$'s are **IID(Independently and Identically Distributed)**, so the **Likelihood** of θ $L(\theta)$ will be

$$\begin{aligned} L(\theta) &= P\left(\vec{y} \mid X; \theta\right) \\ &= \prod_{i=1}^{m} P\left(y^{(i)} \mid x^{(i)}; \theta\right) \\ &= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\left(y^{(i)}-\theta^T x^{(i)}\right)^2}{2\sigma^2}} \end{aligned}$$

Maximum Likelihood: Choose θ to maximize $L(\theta)$

For mathematical convenience, define

$$\begin{aligned} l(\theta) &= log\ L(\theta) \\ &= log \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\left(y^{(i)}-\theta^T x^{(i)}\right)^2}{2\sigma^2}} \\ &= \sum_{i=1}^{m} log \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\left(y^{(i)}-\theta^T x^{(i)}\right)^2}{2\sigma^2}} \\ &= m\ log \frac{1}{\sqrt{2\pi}\sigma} + \sum_{i=1}^{m} log\ e^{-\frac{\left(y^{(i)}-\theta^T x^{(i)}\right)^2}{2\sigma^2}} \\ &= m\ log \frac{1}{\sqrt{2\pi}\sigma} + \sum_{i=1}^{m} -\frac{\left(y^{(i)} - \theta^T x^{(i)}\right)^2}{2\sigma^2} \end{aligned}$$

So maximize $l(\theta)$ is the same as minimize

$$\sum_{i=1}^{m} \frac{\left(y^{(i)} - \theta^T x^{(i)}\right)^2}{2\sigma^2} = J(\theta)$$

Logistic Regression

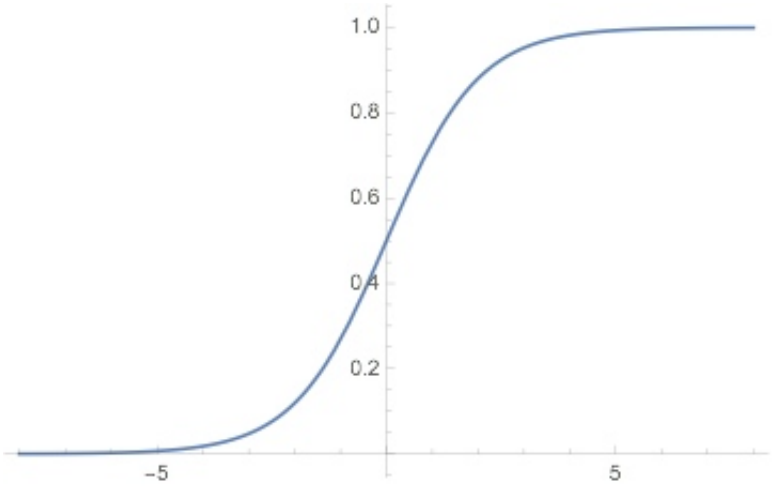
Classification: $y \in \{0, 1\}$

Changing hypothesis as

$$h(x) \in \{0, 1\}$$

Define **Sigmoid Function** (**Logistic Function**)

$$g(z) = \frac{1}{1 + e^{-z}}$$



Choose

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$$

Endow the outputs and hypothesis with probabilistic interpretation

$$P\left(y = 1 \mid x; \theta\right) = h_\theta\left(x\right)$$
$$P\left(y = 0 \mid x; \theta\right) = 1 - h_\theta\left(x\right)$$

Simply,

$$P\left(y \mid x; \theta\right) = h_\theta(x)^y (h_\theta(x))^{1-y}$$

So the likelihood of the parameters is

$$\begin{aligned} L(\theta) &= P\left(\vec{y} \mid x; \theta\right) \\ &= \prod_i P\left(y^{(i)} \mid x^{(i)}; \theta\right) \\ &= \prod_i h_\theta(x^{(i)})^{y^{(i)}} (1 h_\theta(x^{(i)}))^{1-y^{(i)}} \end{aligned}$$

Finding the parameters θ that maximizes the likelihood $L(\theta)$

$$
\begin{aligned}
l(\theta) &= log\ L(\theta) \\
&= \sum_i y^{(i)} log\ h_\theta(x^{(i)}) + (1 - y^{(i)}) log\ (1 - h_\theta(x^{(i)})
\end{aligned}
$$

Apply gradient **ascent** algorithm

$$
\theta := \theta + \alpha \nabla_\theta l(\theta)
$$

Compute the partial derivative of l with respect θ

$$
\frac{\partial}{\partial \theta_j} l(\theta) = \sum_{i=1}^{m} \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)}
$$

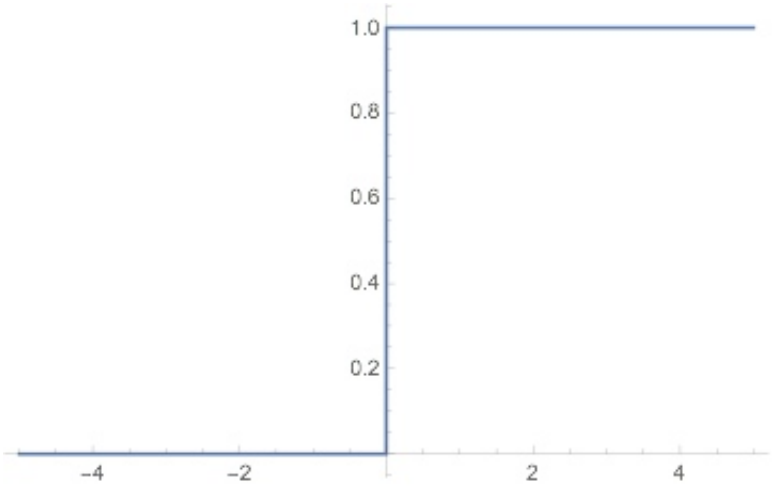
Therefore

$$
\theta_j := \theta_j + \sum_{i=1}^{m} \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)}
$$

Digression Perceptron

To Force $g(z)$ equal 1 or 0, the **perceptron algorithm** defines $g(z)$ to be

$$
g(z) = \begin{cases} 1 & if\ z \geq 0 \\ 0 & otherwise \end{cases}
$$



$$
h_\theta(x) = g(\theta^T x)
$$

And the learning rule is given by this

$$
\theta_j := \theta + \alpha \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)}
$$