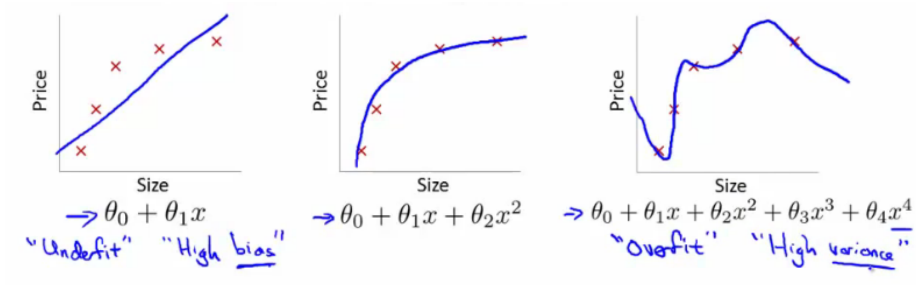
Week 7

July 17, 2021

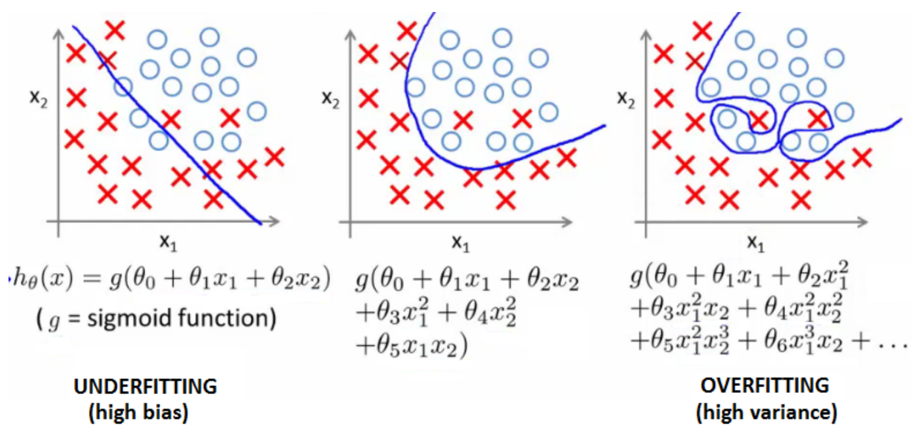
Overfitting with linear regression

- We will use our house pricing example yet another time.
- If we fit a linear function to the data, we have underfitting - also known as high bias.
- If we fit a quadratic function, it will fit better than a straight line, but it will not pass through all of the points.
- If we fit 4th order polynomial, the curve fit's through all five examples. This is overfitting - also known as high variance.



Overfitting with linear regression

- The same thing may happen with logistic regression.
- Sigmoidal function is an underfit.
- A high order polynomial, on the other hand, results in overfitting (high variance hypothesis).

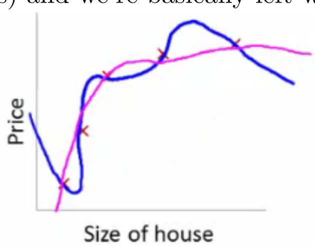


Cost function optimization for regularization

Penalize and make some of the θ parameters really small.

$$min \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)} + y^{(i)})^2 + 1000\theta_3^2 + 1000\theta_4^2$$

So we simply end up with θ_3 and θ_4 being near to zero (because of the huge constants) and we're basically left with a quadratic function.



$$\underline{\theta_0 + \theta_1 x + \theta_2 x^2} + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$$

Regularization
Small parameter values correlate to a simpler hypothesis (you effectively get rid of some of the terms). Overfitting is less likely with a simpler hypothesis.

$$J(\theta) = \frac{1}{2m} [\sum_{i=1}^{m} (h_\theta(x^{(i)} + y^{(i)})^2 + \lambda \sum_{j=1}^{m} \theta_j^2]$$

λ is the regularization parameter that controls a trade off between our two goals:

- Want to fit the training set well.
- Want to keep parameters small.

Later in the course, we'll look at various automated methods for selecting λ.

Regularized linear regression

Algorithm 1 Regularized linear regression

while not converged **do**
$\theta_j := \theta_j - \alpha[\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)} + y^{(i)}) x_j^{(i)} + \frac{\lambda}{m} \theta_j]$
$(for\ j = 1, 2, 3, ..., n)$

Regularization with the normal equation

$$\Theta = \left(X^T X + \lambda \underbrace{\begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & & & \ddots & \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}}_{(n+1)\times(n+1)} \right)^{-1} X^T y$$

e.g. if n = 2 $\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Regularized logistic regression

Logistic regression cost function is as follows:

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} log h_\theta(x^{(i)}) + (1 - y^{(i)}) log(1 - h_\theta(x^{(i)}))] + \frac{\lambda}{2m}\sum_{j=1}^{m}\theta_j^2$$

Algorithm 2 Regularized logistic regression

while not converged **do**

$\quad \theta_j := \theta_j - \alpha[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)} + y^{(i)})x_j^{(i)} + \frac{\lambda}{m}\theta_j]$

$\quad (for\ j = 1, 2, 3, ..., n)$

It appears to be the same as linear regression, except that the hypothesis is different.