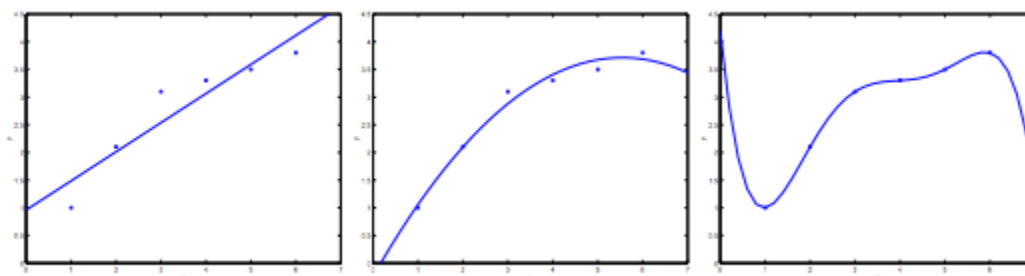# The Problem of Overfitting

Consider the problem of predicting y from $x \in R$. The leftmost figure below shows the result of fitting a $y = \theta\_0 + \theta\_1 x$ $\theta_0 + \theta_1 x$ to a dataset. We see that the data doesn't really lie on straight line, and so the fit is not very good.

Instead, if we had added an extra feature x^2 x_2, and fit $y = \theta\_0 + \theta\_1 x + \theta\_2 x^2$ $y = \theta_0 + \theta_1 x + \theta_2 x_2$, then we obtain a slightly better fit to the data (See middle figure). Naively, it might seem that the more features we add, the better. However, there is also a danger in adding too many features: The rightmost figure is the result of fitting a 5th $5th$ order polynomial $y = \sum\_{j=0}^5 \theta\_j x^j$ $y = \sum_{j=0}^5 \theta_j x_j$. We see that even though the fitted curve passes through the data perfectly, we would not expect this to be a very good predictor of, say, housing prices (y) for different living areas (x). Without formally defining what these terms mean, we'll say the figure on the left shows an instance of **underfitting**—in which the data clearly shows structure not captured by the model—and the figure on the right is an example of **overfitting**.

Underfitting, or high bias, is when the form of our hypothesis function h maps poorly to the trend of the data. It is usually caused by a function that is too simple or uses too few features. At the other extreme, overfitting, or high variance, is caused by a hypothesis function that fits the available data but does not generalize well to predict new data. It is usually caused by a complicated function that creates a lot of unnecessary curves and angles unrelated to the data.

This terminology is applied to both linear and logistic regression. There are two main options to address the issue of overfitting:

1) Reduce the number of features:

  - Manually select which features to keep.
  - Use a model selection algorithm (studied later in the course).

2) Regularization

  - Keep all the features, but reduce the magnitude of parameters θ_j θ_j.

- Regularization works well when we have a lot of slightly useful features.

Cost Function

Note: [5:18 - There is a typo. It should be $\sum_{j=1}^{n} \theta_j ^2 \sum_{j=1}^{n}\theta_{j2}$ instead of $\sum_{i=1}^{n} \theta_j ^2 \sum_{i=1}^{n}\theta_{j2}$]

If we have overfitting from our hypothesis function, we can reduce the weight that some of the terms in our function carry by increasing their cost.

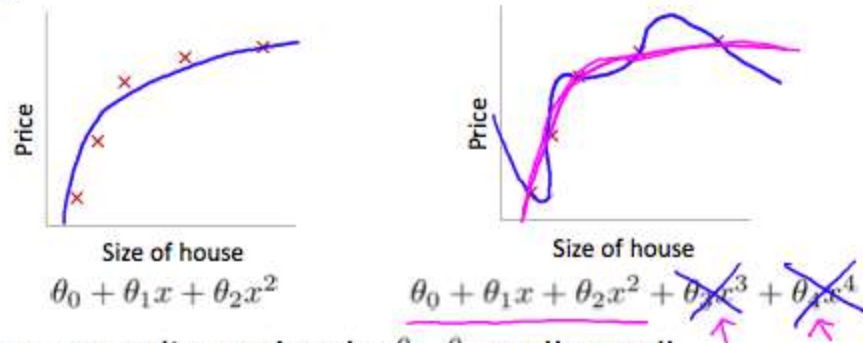Say we wanted to make the following function more quadratic:

$\theta_0 + \theta_1x + \theta_2x^2 + \theta_3x^3 + \theta_4x^4 \theta_0+\theta_1x+\theta_2x_2+\theta_3 x_3+\theta_4x_4$

We'll want to eliminate the influence of $\theta_3x^3 \theta_3x_3$ and $\theta_4x^4 \theta_4x_4$. Without actually getting rid of these features or changing the form of our hypothesis, we can instead modify our **cost function**:

$min_\theta\ \dfrac{1}{2m}\sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000\cdot\theta_3^2 + 1000\cdot\theta_4^2 min_\theta\ 2m1\sum_{i=1m}(h_\theta (x_{(i)})-y_{(i)})2+1000\cdot\theta_{32}+1000\cdot\theta_{42}$

We've added two extra terms at the end to inflate the cost of $\theta_3 \theta_3$ and $\theta_4 \theta_4$. Now, in order for the cost function to get close to zero, we will have to reduce the values of $\theta_3 \theta_3$ and $\theta_4 \theta_4$ to near zero. This will in turn greatly reduce the values of $\theta_3x^3 \theta_3x_3$ and $\theta_4x^4 \theta_4x_4$ in our hypothesis function. As a result, we see that the new hypothesis (depicted by the pink curve) looks like a quadratic function but fits the data better due to the extra small terms $\theta_3x^3 \theta_3x_3$ and $\theta_4x^4 \theta_4x_4$.

Intuition

$\theta_0 + \theta_1 x + \theta_2 x^2$ $\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

Suppose we penalize and make θ_3, θ_4 really small.

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

$\theta_3 \approx 0$ $\theta_4 \approx 0$

We could also regularize all of our theta parameters in a single summation as:

> min_\theta\ \dfrac{1}{2m}\ \sum_{i=1}^^m (h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\ \sum
> $2m1 \sum_{i=1}m(h\theta(x(i))-y(i))2+\lambda \sum_{j=1}n\theta_j2$

The λ, or lambda, is the **regularization parameter**. It determines how much the costs of our theta parameters are inflated.

Using the above cost function with the extra summation, we can smooth the output of our hypothesis function to reduce overfitting. If lambda is chosen to be too large, it may smooth out the function too much and cause underfitting. Hence, what would happen if $\lambda = 0$ $\lambda=0$ or is too small ?

Regularized Linear Regression

Note: [8:43 - It is said that X is non-invertible if m $\leq$ n. The correct statement should be that X is non-invertible if m < n, and may be non-invertible if m = n.

We can apply regularization to both linear regression and logistic regression. We will approach linear regression first.

Gradient Descent

We will modify our gradient descent function to separate out θ_0 from the rest of the parameters because we do not want to penalize θ_0.

$$\text{Repeat \{ } \quad \theta_0 := \theta_0 - \alpha \, \frac{1}{m} \sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_0^{(i)} \quad \theta_j := \theta_j - \alpha \left[\left(\frac{1}{m} \sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \right) + \frac{\lambda}{m}\theta_j \right] \}$$

The term $\frac{\lambda}{m}\theta_j$ performs our regularization. With some manipulation our update rule can also be represented as:

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

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

The first term in the above equation, $1 - \alpha\frac{\lambda}{m}$ will always be less than 1. Intuitively you can see it as reducing the value of θ_j by some amount on every update. Notice that the second term is now exactly the same as it was before.

Normal Equation

Now let's approach regularization using the alternate method of the non-iterative normal equation.

To add in regularization, the equation is the same as our original, except that we add another term inside the parentheses:

$$\theta = (X^TX + \lambda \cdot L)^{-1}X^Ty \quad \text{where} \quad L = \begin{bmatrix} 0 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}$$

L is a matrix with 0 at the top left and 1's down the diagonal, with 0's everywhere else. It should have dimension $(n+1)\times(n+1)$. Intuitively, this is the identity matrix (though we are not including x_0), multiplied with a single real number λ.

Recall that if m < n, then X^TX is non-invertible. However, when we add the term $\lambda \cdot L$, then $X^TX + \lambda \cdot L$ becomes invertible.

Regularized Logistic Regression

We can regularize logistic regression in a similar way that we regularize linear regression. As a result, we can avoid overfitting. The following image shows how the regularized function, displayed by the pink line, is less likely to overfit than the non-regularized function represented by the blue line:

Regularized logistic regression.
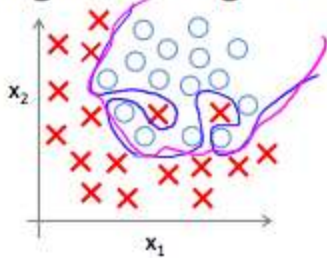
$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2$$
$$+ \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2$$
$$+ \theta_5 x_1^2 x_2^3 + \dots)$$

Cost function:

$$J(\theta) = -\left[\frac{1}{m} \sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right]$$

$$+ \frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2 \qquad \boxed{\theta_1, \theta_2, \dots, \theta_n}$$

Cost Function

Recall that our cost function for logistic regression was:

J(\theta) = - \frac{1}{m} \sum_{i=1}^m \large[y^{(i)}\ \log (h_\theta (x^{(i)}))
+ (1 - y^{(i)})\ \log (1 - h_\theta(x^{(i)})) \large]$J(\theta)=-m1\sum_{i=1m}[y(i) \log(h\theta
(x(i)))+(1-y(i)) \log(1-h\theta(x(i)))]$

We can regularize this equation by adding a term to the end:

J(\theta) = - \frac{1}{m} \sum_{i=1}^m \large[y^{(i)}\ \log (h_\theta (x^{(i)})) + (1 - y^{
h_\theta(x^{(i)}))\large] + \frac{\lambda}{2m}\sum_{j=1}^n \theta_j^2$J(\theta)=-m1\sum_{i=1m}[y(i)
(x(i)))+(1-y(i)) \log(1-h\theta(x(i)))]+2m\lambda\sum_{j=1n}\theta{j2}$

The second sum, \sum_{j=1}^n \theta_j^2$\sum_{j=1n}\theta{j2}$ **means to explicitly exclude** the
bias term, \theta_0θ_0. I.e. the θ vector is indexed from 0 to n (holding n+1 values,
\theta_0θ_0 through \theta_nθ_n), and this sum explicitly skips \theta_0θ_0, by running from
1 to n, skipping 0. Thus, when computing the equation, we should continuously update the
two following equations:

Gradient descent

Repeat {

$\rightarrow$ $\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$

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

$(j = \textcolor{red}{✗}, 1, 2, 3, \ldots, n)$

$\theta_1 \ldots \theta_n$

}

$\frac{\partial}{\partial \theta_j} J(\theta)$

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