Regularization

May 28, 2012

1 The problem of overfitting.

- **Underfitting** - High bias - doesn't fit the data very well
- **Overfitting** - High variance - high order polynomial, fits data very well, space of possible hypothesis is too variable to give a good one
  - If we have too many features, the learned hypothesis may fit the training set very well ($J(\Theta) \approx 0$), but fail to generalize to new examples (predict prices on new examples)

1.1 Addressing overfitting

Lots of features and little training data, overfitting can occur.
If we think overfitting is occuring, what can we do?

- Plotting hypothesis - to select an appropriate degree polynomial. Doesn't always work, you can have too many features.
- Reduce number of features.
  - Manually select which features to keep.
  - Model selection algorithm
- Regularization.
  - Keep all features, but reduce magnitude/values of parameters Θ_j.
  - Works well when we have a lot of features, each of which contributes a bit to predicting y.

2 Cost function

Take parameters:
$\Theta_0 + \Theta_1 x + \Theta_2 x^2$ $\rightarrow$just right
$\Theta_0 + \Theta_1 x + \Theta_2 x^2 + \Theta_3 x^3 + \Theta_4 x^4$ $\rightarrow$overfitted
Suppose we penalize and make Θ_3, Θ_4really small.
$\min_{\Theta} \frac{1}{2m} \sum_{i=1}^{m} (h_\Theta(x^{(i)}) - y^{(i)})^2 + 1000\Theta_3^2 + 1000\Theta_4^2$
The only way to make this cost function small is when $\Theta_3 \approx 0 \, and \, \Theta_4 \approx 0$ and thus essentially getting rid of the high order terms.

In general:
Small values for parameters $\Theta_0, \Theta_1, \ldots, \Theta_n$

- "Simpler" hypothesis
- Less prone to overfitting

Ex: Housing:
Features: $x_1, x_2, \ldots, x_{100}$- hard to pick in advance which are the ones that are less likely to be relevant
Parameters: $\Theta_0, \Theta_1, \ldots, \Theta_{100}$
We'll take the cost function and add a regularization term at the end to shrink all of the parameters. Not the $\Theta_0$term

$$J(\Theta) = \frac{1}{2m} \left[\sum_{i=1}^{m} (h_\Theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} \Theta_j^2 \right]$$

λ- regularization parameter, controls a tradeoff between two different goals:

- we would like to train the training set well (first term)
- we want to keep the parameters small (regularization term) and therefore keeping the hypothesis simple to avoid overfitting

What if $\lambda$is set to an extremely large value (perhaps too large for our problem, say $\lambda = 10^{10}$)?
We would end up penalizing all the parameters so they end up close to zero, ending up with a hypothesis:

$$h_\Theta(x) = \Theta_0$$

and underfit the training set. Hypothesis has too high bias or preconception.

3 Regularized linear regression

For linear regression we have two algorithms

- Gradient descent

- Normal equation

We'll generalize them for linear regression.

$$J(\Theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\Theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\Theta_j^2\right]$$

3.1 Gradient descent

$$\begin{aligned}
&\mathit{Repeat}\,\{\\
&\qquad \Theta_j := \Theta_j - \alpha\tfrac{1}{m}\sum_{i=1}^{m}(h_\Theta(x)^{(i)} - y^{(i)})x_j^{(i)}\\
&\qquad (j = 0, 1, 2, 3, \ldots, n)\\
&\}
\end{aligned}$$

With regularization for linear regression:

$$\begin{aligned}
&\mathit{Repeat}\,\{\\
&\qquad \Theta_0 := \Theta_0 - \alpha\tfrac{1}{m}\sum_{i=1}^{m}(h_\Theta(x)^{(i)} - y^{(i)})x_0^{(i)}\\
&\qquad \Theta_j := \Theta_j - \alpha\tfrac{1}{m}\sum_{i=1}^{m}(h_\Theta(x)^{(i)} - y^{(i)})x_j^{(i)} + \frac{\lambda}{m}\Theta_j\\
&\} \qquad (j = 1, 2, 3, \ldots, n)
\end{aligned}$$

The update for $\Theta_j$can be written equivalently as

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

The first term, $1 - \alpha\frac{1}{m} < 1$, has an effect of shrinking $\Theta_j$and then the second term is the same as original gradient descent update.

3.2 Normal equation

$$X = \begin{bmatrix} (x^{(1)})^\top \\ \vdots \\ (x^{(m)})^\top \end{bmatrix} \qquad y = \begin{bmatrix} y^{(1)} \\ \\ y^{(m)} \end{bmatrix} \in \mathbb{R}^m$$

$$\Theta = (X^\top X + \lambda \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 1 \end{bmatrix})^{-1} X^\top y$$

The inner matrix has dimensions: $(n+1) \times (n+1)$

3.2.1 Non-invertibility

Suppose $m \leq n$, (# examples < #features)
pinv will give an example that kindof makes sense, numerically correct but not necessarily a good hypothesis.
if $\lambda > 0$, it is possible to prove that term of

$$\Theta = (X^\top X + \lambda \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix})^{-1} X^\top y$$

in parentheses will be invertible.

4 Regularized Logistic Regression

$$J(\Theta) = -\left[\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log h_\Theta(x^{(i)}) + (1-y^{(i)})\log(1-h_\Theta(x^{(i)}))\right] + \lambda\sum_{j=1}^{n}\Theta_j^2$$

So now, even if we're fitting a very high order polynomial, we'll more likely get a decision boundary that looks reasonable.
Implementation:

$$\begin{aligned} &\textit{Repeat}\,\{ \\ &\qquad \Theta_0 := \Theta_0 - \alpha\tfrac{1}{m}\sum_{i=1}^{m}(h_\Theta(x)^{(i)} - y^{(i)})x_0^{(i)} \\ &\qquad \Theta_j := \Theta_j - \alpha\left[\tfrac{1}{m}\sum_{i=1}^{m}(h_\Theta(x)^{(i)} - y^{(i)})x_j^{(i)} + \frac{\lambda}{m}\Theta_j\right] \\ &\} \qquad\qquad (j = 1, 2, 3, \dots, n) \end{aligned}$$

The difference from linear regression is that the hypothesis is different

$$h_\Theta(x) = \frac{1}{1+e^{-\Theta^\top x}}$$

4.1 Advanced optimization

function [jVal, gradient] = costFunction(theta)
% below code now must include the regularization parameter
jval = [code to compute $J(\Theta)$];

gradient(1) = [code to compute $\frac{\partial}{\partial \Theta_0} J(\Theta)$];
gradient(2) = [code to compute $\frac{\partial}{\partial \Theta_1} J(\Theta)$];

$\vdots$

gradient(n+1) = [code to compute $\frac{\partial}{\partial \Theta_n} J(\Theta)$];

Then you pass the results to fminunc or other advanced optimization algorithm, the params you get out will correspond to regularized logistic regression.