Overfitting and Regularization

Dan Sheldon

February 12, 2019

Plan

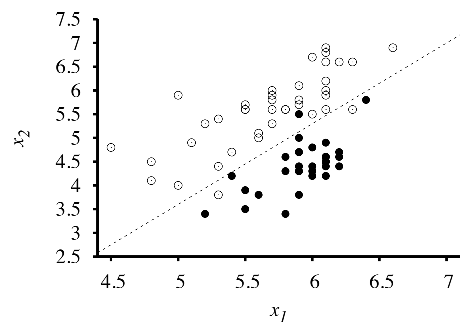
- What is Overfitting?
- How to Diagnose Overfitting
- Regularization

What is Overfitting?

Demo: polynomials

What is Overfitting?

Complex decision boundaries



What is Overfitting?

Overfitting is learning a model that fits the training data very well, but does not generalize well.

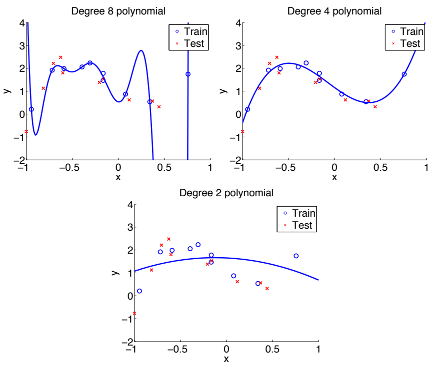
(Generalize = predict accurately for new examples.)

How to Diagnose Overfitting?

Exercise

How to Diagnose Overfitting?

Exercise
Reserve some data to test whether hypothesis generalizes well



Train Data vs. Test Data

Very simple but important methodology!!

- Start with m training examples

$$(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\mathbf{x}^{(m)}, y^{(m)})$$

- Split into *train* and *test* sets (usually random)
- Training data: use to fit model
- Test data: use to evaluate fit

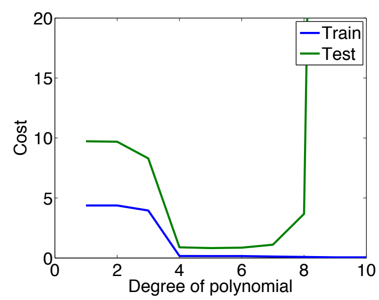
Details/illustration on board

How to Diagnose Overfitting?

Example: cost function vs. degree of polynomial

How to Diagnose Overfitting?

Example: cost function vs. degree of polynomial



How to Diagnose Overfitting?

Example: feature expansion for book data

Width x_1	Thickness x_2	Height x_3	Weight y
8	1.8	10	4.4
8	0.9	9	2.7
	...		

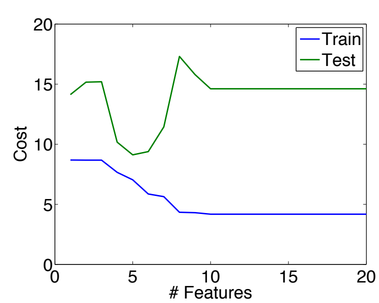
Suppose you add "quadratic" features:

$$(x_1, \ldots, x_n,) \mapsto (\underbrace{x_1, \ldots, x_n}_{\text{original features}}, \underbrace{x_1^2,\ x_1x_2,\ x_1x_3,\ \ldots,\ x_{n-1}x_n,\ x_n^2}_{\text{products of two original features}})$$

Do more features help?

How to Diagnose Overfitting?

Example: cost function vs. number of features in book data



Cost vs. Complexity

General phenomenon: training/test cost vs. model "complexity"

What Makes a Model Complex?

- Polynomial: higher degree
- Book data: more features
- Linear functions ($h_{\boldsymbol{\theta}}(\mathbf{x}) = \boldsymbol{\theta}^T\mathbf{x}$): large weights (steep hyperplanes)

Large Weights

Example

Width x_1	Thickness x_2	Height x_3	Weight y
8	1.8	10	4.4
8	0.9	9	2.7
	...		

Which is more complex?

$$y = -3.94 + 0.18x_1 + .34x_2 + 0.4x_3$$

vs.

$$y = 2842 - 957x_1 + 300x_2 + 69712x_3$$

Solution to Overfitting: Regularization

Intuition: large weights $\rightarrow$ high complexity

Modify the cost function to penalize large weights = "regularization"

For squared error, we get:

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

λ controls trade-off between model complexity and fit

Notes

Penalty / regularization term:

$$\frac{\lambda}{2}\sum_{j=1}^{n}\theta_j^2$$

- Best practice not to regularize θ_0. Why?
- Often written as $\frac{\lambda}{2}\|\boldsymbol{\theta}\|^2$ (Need to be careful to specify whether $\boldsymbol{\theta}$ include θ_0 or not!).

Discussion

Regularization is really important!!!

Why?

Learning with Regularization

Let's see how to solve two learning problems with regularized cost functions:

- Linear regression
- Logistic regression

Linear Regression: Normal Equations with Regularization

Find $\boldsymbol{\theta}$ to minimize regularized $J(\boldsymbol{\theta})$

$$\boldsymbol{\theta} = (X^T X + \lambda \hat{I})^{-1} X^T y$$

Linear Regression: Normal Equations with Regularization

Find $\boldsymbol{\theta}$ to minimize regularized $J(\boldsymbol{\theta})$

$$\boldsymbol{\theta} = (X^T X + \lambda \hat{I})^{-1} X^T y$$

$$\hat{I} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix}$$

(Identity matrix with top left entry replaced by 0)

Normal Equations Derivation: Vectorized Cost Function

$$\begin{aligned} J(\boldsymbol{\theta}) &= \frac{1}{2}\sum_{i=1}^{m}(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2}\sum_{j=1}^{n}\theta_j^2 \\ &= \frac{1}{2}(X\boldsymbol{\theta} - \mathbf{y})^T(X\boldsymbol{\theta} - \mathbf{y}) + \frac{\lambda}{2}\hat{\boldsymbol{\theta}}^T\hat{\boldsymbol{\theta}}. \end{aligned}$$

Normal Equations Derivation: Vectorized Cost Function

$$\begin{aligned} J(\boldsymbol{\theta}) &= \frac{1}{2}\sum_{i=1}^{m}(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2}\sum_{j=1}^{n}\theta_j^2 \\ &= \frac{1}{2}(X\boldsymbol{\theta} - \mathbf{y})^T(X\boldsymbol{\theta} - \mathbf{y}) + \frac{\lambda}{2}\hat{\boldsymbol{\theta}}^T\hat{\boldsymbol{\theta}}. \end{aligned}$$

$$\hat{\boldsymbol{\theta}} = \begin{bmatrix} 0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} = \hat{I}\boldsymbol{\theta}$$

Normal Equations Derivation

$$J(\boldsymbol{\theta}) = \frac{1}{2}(X\boldsymbol{\theta} - \mathbf{y})^T(X\boldsymbol{\theta} - \mathbf{y}) + \frac{\lambda}{2}\hat{\boldsymbol{\theta}}^T\hat{\boldsymbol{\theta}}$$

Set derivative to zero and solve (review on your own)

$$\begin{aligned} 0 = \frac{d}{d\boldsymbol{\theta}}J(\boldsymbol{\theta}) &= (X\boldsymbol{\theta} - \mathbf{y})^T X + \lambda\hat{\boldsymbol{\theta}}^T \\ 0 &= X^T(X\boldsymbol{\theta} - \mathbf{y}) + \lambda\hat{I}\boldsymbol{\theta} \\ X^T X\boldsymbol{\theta} + \lambda\hat{I}\boldsymbol{\theta} &= X^T\mathbf{y} \\ (X^T X + \lambda\hat{I})\boldsymbol{\theta} &= X^T\mathbf{y} \\ \boldsymbol{\theta} &= (X^T X + \lambda\hat{I})^{-1}X^T\mathbf{y} \end{aligned}$$

Linear Regression: Regularized Gradient Descent

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

Repeat until convergence

Linear Regression: Regularized Gradient Descent

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

Repeat until convergence

$$\theta_0 \leftarrow \theta_0 - \alpha\sum_{i=1}^{m}(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)})$$

$$\theta_j \leftarrow \theta_j - \alpha\Big(\lambda\theta_j + \sum_{i=1}^{m}(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)})x_j^{(i)}\Big), \quad j = 1, \ldots, n$$

Linear Regression: Regularized Gradient Descent

Update rule for θ_j after simplification:

$$\theta_j \leftarrow \underbrace{\theta_j(1-\alpha\lambda)}_{\text{shrink}} - \underbrace{\alpha\sum_{i=1}^{m}(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)})x_j^{(i)}}_{\text{old gradient}}, \quad j = 1, \ldots, n$$

Interpretation: first "shrink" weights, then take gradient step for *unregularized* cost function

Logistic Regression: Regularized Gradient Descent

$$J(\boldsymbol{\theta}) = \frac{\lambda}{2}\sum_{j=1}^{n}\theta_j^2 + \sum_{i=1}^{m}\Big(-y^{(i)}\log h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - (1-y^{(i)})\log\big(1-h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})\big)\Big)$$

Algorithm:

Repeat until convergence

$$\theta_0 \leftarrow \theta_0 - \alpha\sum_{i=1}^{m}(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)})$$

$$\theta_j \leftarrow \theta_j(1-\alpha\lambda) - \alpha\sum_{i=1}^{m}(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)})x_j^{(i)}, \quad j = 1, \ldots, n$$

(Again: same as linear regression, but different $h_{\boldsymbol{\theta}}(\mathbf{x})$)

What You Need To Know

- Concept of overfitting
- Diagnosis: train/test sets
- Regularized cost function (penalize weights)
- Regularized gradient descent ("weight shrinking")
- See it work: polynomial regularization demo