Regression

CPS 170

Ron Parr

With content adapted from Lise Getoor, Tom Dietterich, Andrew Moore & Rich Maclin

Supervised Learning

- Given: Training Set
- Goal: Good performance on test set
- Assumptions:
  - Training samples are independently drawn, and identically distributed (IID)
  - Test set is from same distribution as training set

Fitting Continuous Data (Regression)

- Datum i has feature vector: $\phi=(\phi_1(x^{(i)})...\phi_k(x^{(i)}))$
- Has real valued target: $t^{(i)}$
- Concept space: linear combinations of features:

$$y(\mathbf{x}^{(i)};\mathbf{w}) = \sum_{j=1}^{k} \phi_j(\mathbf{x}^{(i)}) w_j = \boldsymbol{\phi}(\mathbf{x}^{(i)})^T \mathbf{w}$$

- Learning objective: Search to find "best" w
- (This is standard "data fitting" that most people learn in some form or another.)

Linearity of Regression

- Regression typically considered a *linear* method, but…
- Features not necessarily linear
- Features not necessarily linear
- Features not necessarily linear
- Features not necessarily linear
- and, BTW, features not necessarily linear

Regression Examples

- Predicting housing price from:
  - House size, lot size, rooms, neighborhood*, etc.
- Predicting weight from:
  - Sex, height, ethnicity, etc.
- Predicting life expectancy increase from:
  - Medication, disease state, etc.
- Predicting crop yield from:
  - Precipitation, fertilizer, temperature, etc.
- Fitting polynomials
  - Features are monomials

Features/Basis Functions

- Polynomials
- Indicators
- Gaussian densities
- Step functions or sigmoids
- Sinusoids (Fourier basis)
- Wavelets
- Anything you can imagine…

What is "best"?

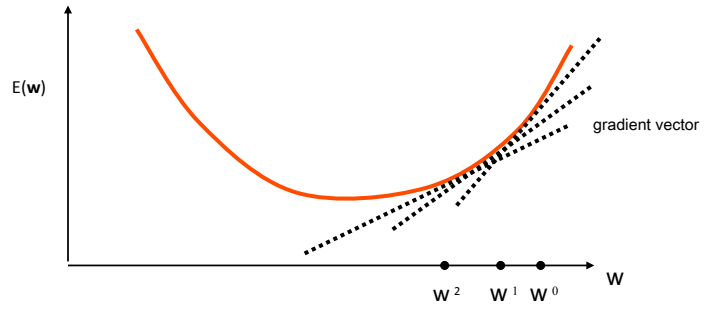
- No obvious answer to this question
- Three compatible answers:
  - Minimize squared error on training set
  - Maximize likelihood of the data (under certain assumptions)
  - Project data into "closest" approximation
- Other answers possible

Minimizing Squared Training Set Error

- Why is this good?
- How could this be bad?
- Minimize:

$$E(\mathbf{w}) = \sum_{i=1}^{N} \left(\mathbf{w}^T \phi(\mathbf{x}^{(i)}) - t^{(i)} \right)^2$$

Minimizing E by Gradient Descent



Start with initial weight vector $\mathbf{w}_0$

Compute the gradient $\nabla_{\mathbf{w}} E = \left(\frac{\partial E(\mathbf{w})}{\partial w_0}, \frac{\partial E(\mathbf{w})}{\partial w_1}, \cdots, \frac{\partial E(\mathbf{w})}{\partial w_n} \right)$

Compute $\mathbf{w} \leftarrow \mathbf{w} - \alpha \nabla E$ where α is the step size

Repeat until convergence

(Adapted from Lise Getoor's Slides)

Gradient Descent Issues

- For this particular problem:
  - Global minimum exists
  - Convergence "guaranteed" if done in "batch"
- In general
  - Local optimum only
  - Batch mode more stable
  - Incremental possible
    - Can oscillate
    - Use decreasing step size (Robbins-Monro) to stabilize

Solving the Minimization Directly

$$E = \sum_{i=1}^{n} (t^{(i)} - w^T \phi(x^{(i)}))^2$$

$$\nabla_w E \propto \sum_{i=1}^{n} (t^{(i)} - w^T \phi(x^{(i)})) \phi(x^{(i)})^T$$

scalar row vector

Set gradient to 0 to find min:

$$\sum_{i=1}^{n} (t^{(i)} - \mathbf{w}^T \phi(x^{(i)})) \phi(x^{(i)})^T = 0$$

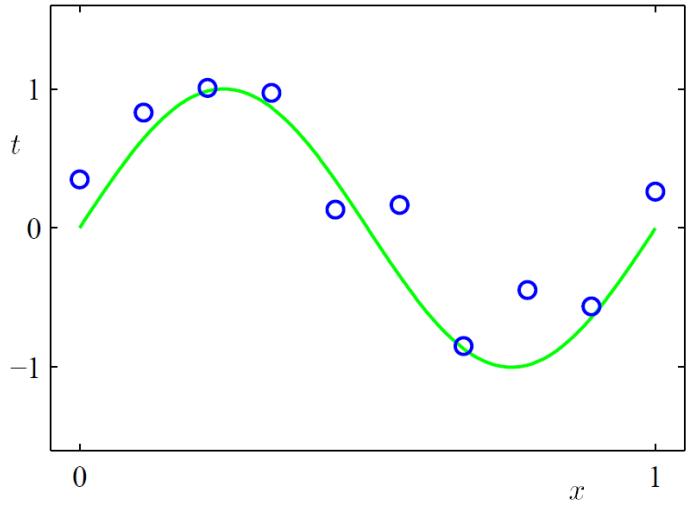
$$\sum_{i=1}^{n} \phi(x^{(i)})^T t^{(i)} - \mathbf{w}^T \sum_{i=1}^{n} \phi(x^{(i)}) \phi(x^{(i)})^T = 0$$

$$\mathbf{t}^T \Phi - \mathbf{w}^T \Phi^T \Phi = \Phi^T \mathbf{t} - \Phi^T \Phi \mathbf{w} = 0$$

$$\mathbf{w} = \left(\Phi^T \Phi\right)^{-1} \Phi^T \mathbf{t}$$

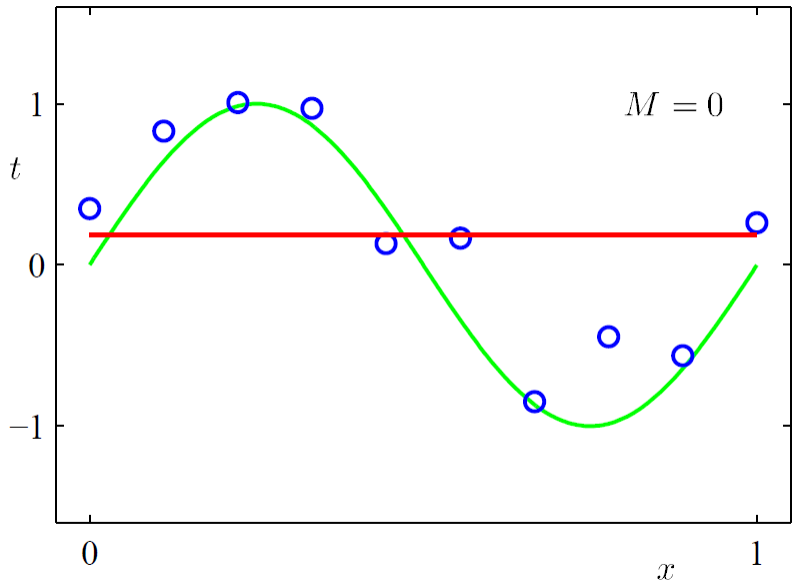
$$\Phi = \begin{bmatrix} \phi(x^{(1)}) \\ \phi(x^{(2)}) \\ \vdots \\ \phi(x^{(n)}) \end{bmatrix}$$

What is the Best Choice of Features?

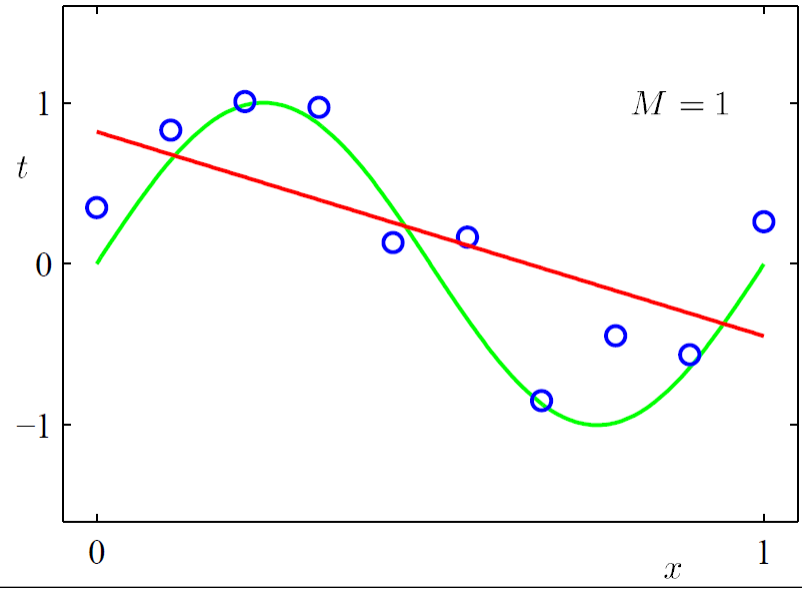


Noisy Source Data

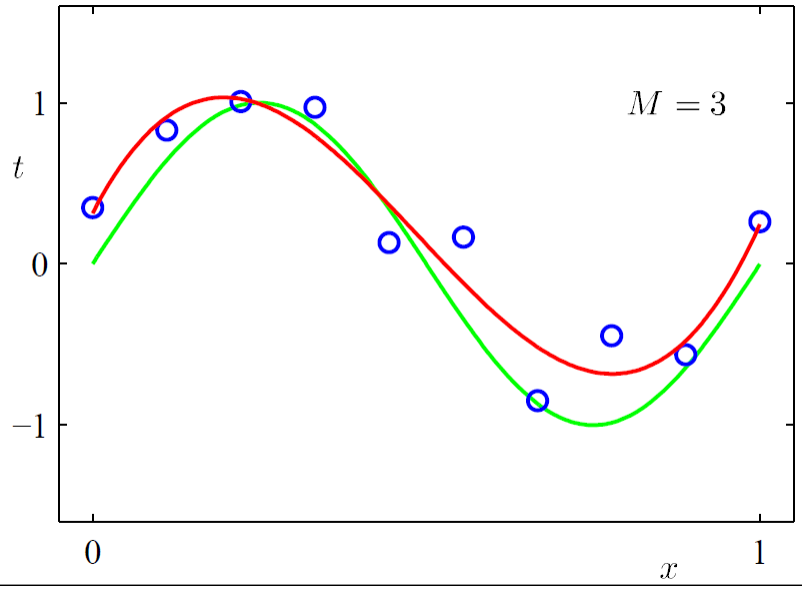
Degree 0 Fit



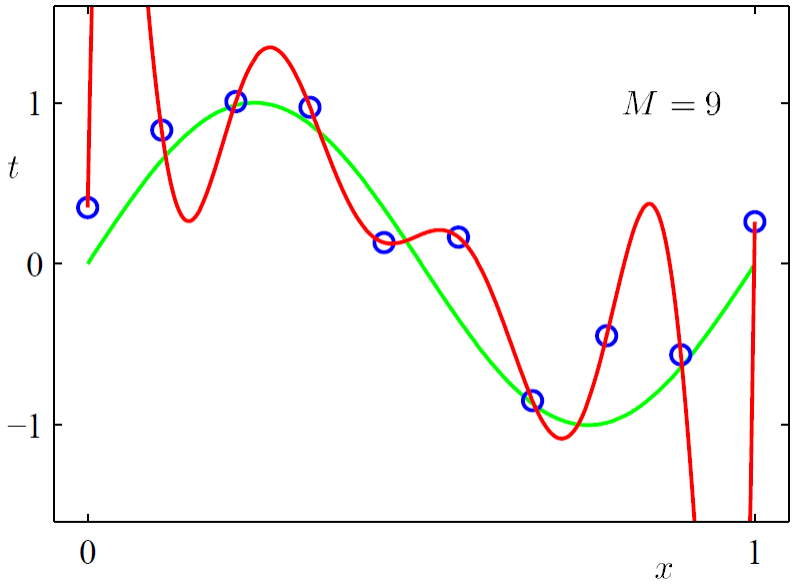
Degree 1 Fit



Degree 3 Fit

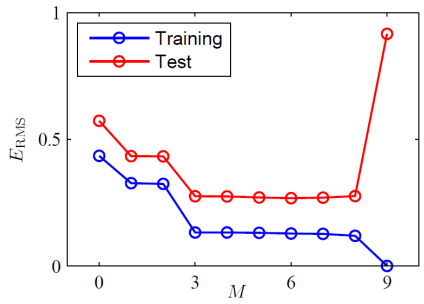


Degree 9 Fit



Observations

- Degree 3 is the best match to the source
- Degree 9 is the best match to the samples
- Performance on test data:



Bias and Variance

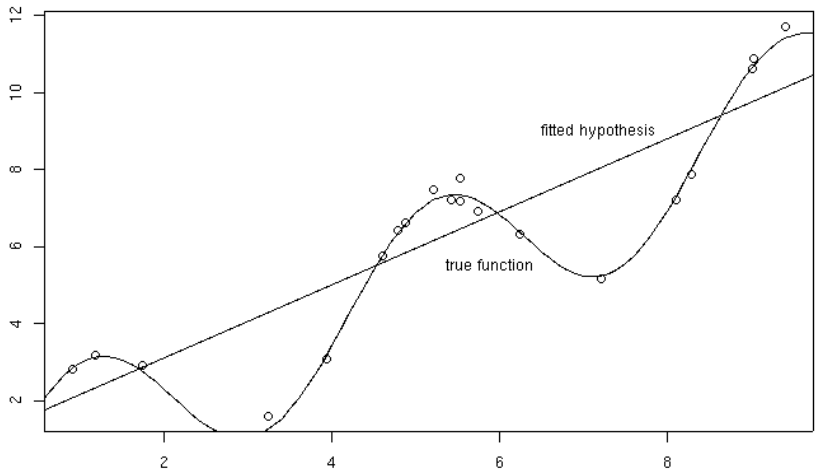
- Bias: How much of our error comes from our choice of hypothesis space?

- Variance: How much of our error comes from noise in the training data?

Example: 20 points

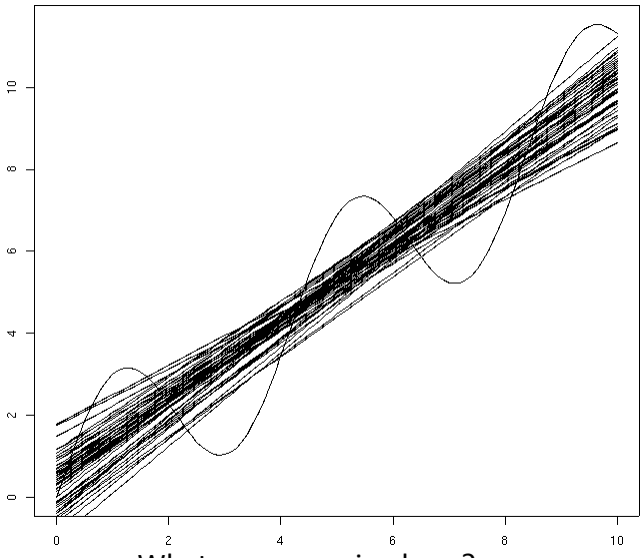
$y = x + 2 \sin(1.5x) + N(0,0.2)$

Noise



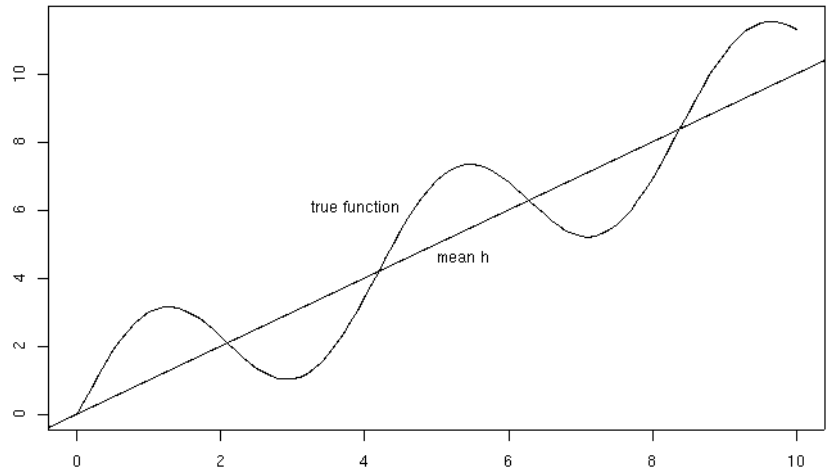
Hypothesis space = linear in x

50 fits (20 examples each)

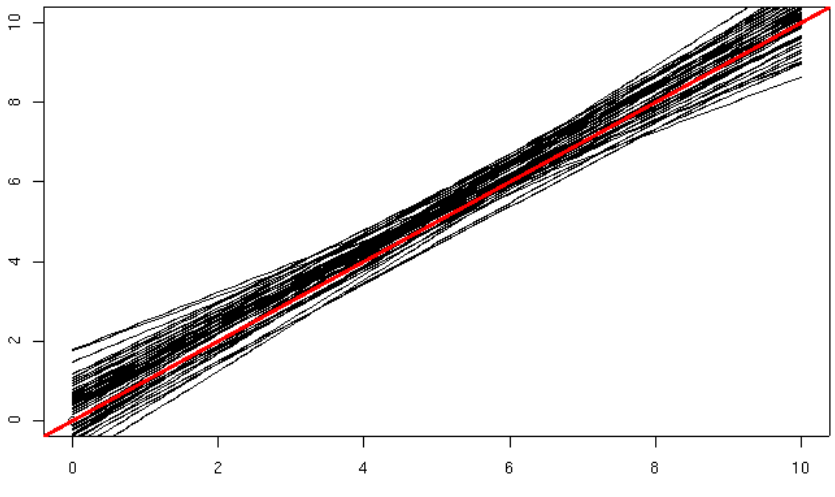


What are we seeing here?

Bias

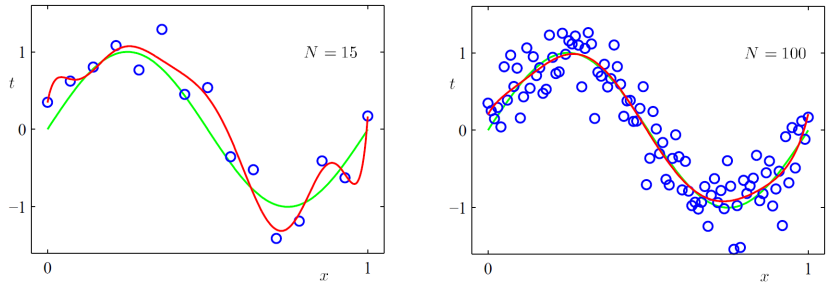


Variance



Trade off Between Bias and Variance

- Is the problem a bad choice of polynomial?
- Is the problem that we don't have enough data?
- Answer: Yes
- For small datasets:
  - Lower bias -> Higher Variance
  - Higher bias -> Lower Variance



Bias and Variance: Lessons Learned

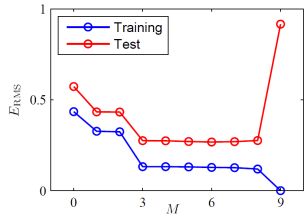
- When data are scarce relative to the "capacity" of our hypothesis space
  - Variance can be a problem
  - Restricting hypothesis space can reduce variance at cost of increased bias
- When data are plentiful
  - Variance is less of a concern
  - May afford to use richer hypothesis space

Methods for Choosing Features

- Cross validation
- Regularization

Cross Validation

- Suppose we have many possible hypothesis spaces, e.g., different degree polynomials
- Recall our empirical performance results:



- Why not use the data to find min of the red curve?

Implementing Cross Validation

- Many possible approaches to cross validation
- Typical approach divides data into k equally sized chunks:
  - Do k instances of learning
  - For each instance hold out 1/k of the data
  - Train on (k-1)/k fraction of the data
  - Test on held out data
  - Average results
- Can also sample subsets of data with replacement
- Cross validation can be used to search range of hypothesis classes to find where **overfitting** starts

Problems with Cross Validation

- Cross validation is a sound method, but requires a lot of data and/or is slow
- Must trade off two factors:
  - Want enough data within each run
  - Want to average over enough trials
- With scarce data:
  - Choose k close to n
  - Almost as many learning problems as data points
- With abundant data (then why are you doing cross validation?)
  - Choose k = a small constant, e.g., 10
  - Not too painful if you have a lot of parallel computing resources and a lot of data, e.g., if you are Google

Regularization

- Cross validation may also be impractical if range of hypothesis classes is not easily enumerated a searched iteratively
- Regularization aims to avoid overfitting, while
  - Avoiding speed penalty of cross validation
  - Not assuming an ordering on hypothesis spaces

Regularization

- Idea: Penalize overly complicated answers
- Ordinary regression minimizes:

$$\sum_{i=1}^{M} (y(x^{(i)}; \mathbf{w}) - t_i)^2$$

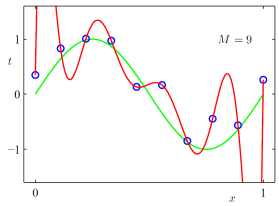
- L_2 Regularized regression minimizes:

$$\lambda \|\mathbf{w}\|_2 + \sum_{i=1}^{M} (y(x^{(i)}; \mathbf{w}) - t^{(i)})^2$$

- Note: May exclude constants form the norm

L_2 Regularization: Why?

$$\lambda\|\mathbf{w}\|_2 + \sum_{i=1}^{M}(y(x^{(i)};\mathbf{w}) - t^{(i)})^2$$



- For polynomials, extreme curves typically require extreme values
- In general, encourages use of features only when they lead to a substantial increase in performance
- Problem: How to choose λ (cross validation?)

The L_2 Regularized Solution

- Minimize:

$$\lambda\|\mathbf{w}\|_2 + \sum_{i=1}^{M}(y(x^{(i)};\mathbf{w}) - t^{(i)})^2$$

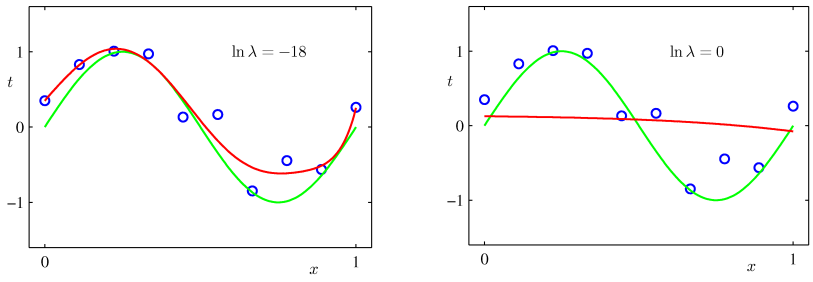
- Set gradient to 0, solve for w for features Φ:

$$\mathbf{w} = (\Phi^T\Phi + \lambda I)^{-1}\Phi^T\mathbf{t}$$

- Compare with unregularized solution

$$\mathbf{w} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{t}$$

Regularization Example



High regularization produces "flat" solutions because weights must approach 0. Lower values allow for more curviness in the value function.

Concluding Comments

- Regression is the most basic machine learning algorithm for continuous targets
- Multiple views are all equivalent:
  - Minimize squared loss
  - Maximize likelihood
  - Orthogonal projection
- Big question: Choosing features
- Step towards understanding this: *Bias/variance trade off*
- Cross validation, regularization automate (to some extent) balancing bias and variance