Regression

CPS 271

Ron Parr

Regression figures provided by Christopher Bishop and © 2007 Christopher Bishop

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

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

- 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:

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

Maximizing Likelihood of Data

- Assume:
  - True model is in H
  - Data have Gaussian noise
- Actually might want:

$$\arg\max_H P(H \mid X) = \frac{P(X \mid H)P(H)}{P(X)}$$

- Is maximizing P(X|H) a good surrogate? (maximizing over w)

Maximizing P(X|H)

- Assume: $t^{(i)} = y^{(i)} + \varepsilon^{(i)}$
- Where: $P(\varepsilon^{(i)}) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(\varepsilon^{(i)})^2}{2\sigma^2})$

(Gaussian distribution w/mean 0, standard deviation σ)

- Therefore:

$$P(t^{(i)} \mid x^{(i)}, w) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(t^{(i)} - w^T \boldsymbol{\varphi}(x^{(i)}))^2}{2\sigma^2})$$

Maximization Continued

- Maximizing over entire data set:

$$\prod_{i=1}^{n} P(t^{(i)} \mid x^{(i)}, \theta) = \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(t^{(i)} - w^T x^{(i)})^2}{2\sigma^2})$$

- Maximizing equivalent log formulation: (ignoring constants)

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

- Or minimizing:

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

Look familiar?

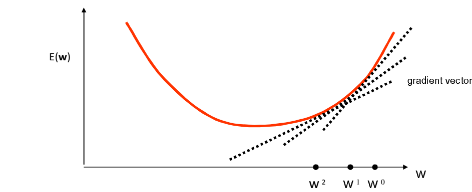
Checkpoint

- So far we have considered:
  - Minimizing squared error on training set
  - Maximizing Likelihood of training set (given model, and some assumptions)
- Different approaches w/same objective!

Solving the Optimization Problem

- Nota bene: Good to keep optimization problem and optimization technique separate in your mind
- Some optimization approaches:
  - Gradient descent
  - Direct Minimization

Minimizing E by Gradient Descent



Start with initial weight vector w_0

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

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

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

$$\nabla_w E \propto \sum_{i=1}^{n} \underbrace{(t^{(i)} - w^T \boldsymbol{\varphi}(x^{(i)}))}_{\text{scalar}} \underbrace{\boldsymbol{\varphi}(x^{(i)})^T}_{\text{row vector}}$$

Set gradient to 0 to find min:

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

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

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

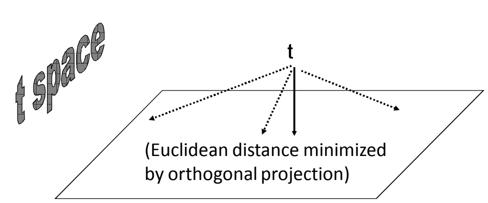
$$\mathbf{w} = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{t}$$

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

Geometric Interpretation

- $\mathbf{t}=(t^{(1)}\dots t^{(n)})$ = point in n-space
- Ranging over **w**, $\mathbf{w}^T\phi$ = H =
  - column space of features
  - subspace of R^n occupied by H
- Goal: Find "closest" point in H to **t**
- Suppose closeness = Euclidean distance

Another Geometric Interpretation



H space (linear combinations of $\phi(x)$)

Minimizing Euclidean Distance

- Minimize: $\left|\mathbf{t} - \mathbf{w}^T \Phi\right|_2$
- For n data points:

$$\sqrt{\sum_{i=1}^{n} (t^{(i)} - \mathbf{w}^T \boldsymbol{\varphi}(x^{(i)}))^2}$$

- Equivalent to minimizing:

$$\sum_{i=1}^{n} (t^{(i)} - \mathbf{w}^T \boldsymbol{\varphi}(x^{(i)}))^2$$

Look familiar?

Checkpoint

- Three different ways to pick **w** in H
  - Minimize squared error on training set
  - Maximize likelihood of training set
  - Distance minimizing projection into H
- All lead to same optimization problem!

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

Geometric Solution

- Geometric Approach (Strang)
- Let $\mathbf{\Phi}$ be the design matrix (see board)
- Require orthogonality:

$$\forall z : (\Phi z)^T (\Phi \mathbf{w} - \mathbf{t}) = 0$$

Any vector in H — Line from **t** to solution

$$\forall z : z^T [\Phi^T \Phi \mathbf{w} - \Phi^T \mathbf{t}] = 0$$

Direct Solution Continued

- When is this true: $\forall z : z^T [\Phi^T \Phi \mathbf{w} - \Phi^T \mathbf{t}] = 0$
- When:

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

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

← Same solution as direct minimization of error

When does the inverse exist?

Adding Regularization

- We previously considered adding a penalty to error function do discourage overfitting

$$E = 0.5\lambda \mathbf{w}^T \mathbf{w} + \sum_{i=1}^{M} (y(x^{(i)}; \mathbf{w}) - t_i)^2$$

- Equivalent to a Gaussian, mean 0 prior on w
- Direction solution (exercise):

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

What if $t^{(i)}$ is a vector?

- Nothing changes!
- Scalar prediction:

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

- Vector prediction (exercise):

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

Weight matrix ↗ — ↖ Target matrix

What about other criteria?

- How about minimizing worse case loss?

$$\min_{\mathbf{w}} \max_i \left(\mathbf{w}^T \mathbf{x}^{(i)} - t^{(i)}\right)$$

- Solve by linear program…

Minimizing Max Error

- Constraints: $\forall i:$
  $\varepsilon > \mathbf{w}^T \varphi(\mathbf{x}^{(i)}) - t^{(i)}$
  $\varepsilon > t^{(i)} - \mathbf{w}^T \varphi(\mathbf{x}^{(i)})$
- Objective: Minimize ε
- Don't use for noisy data!

Understanding Loss

- Suppose we have a squared error loss function: L (gets too confusing to use E)
- Define h(**x**)=E[t|**x**]

$$E[L] = \underbrace{\int \{y(\mathbf{x}) - h(\mathbf{x})\}^2 p(\mathbf{x}) d\mathbf{x}}_{\text{Mismatch between hypothesis and target – we can influence this}} + \underbrace{\int \{h(\mathbf{x}) - t\} p(\mathbf{x}, t) d\mathbf{x} dt}_{\text{Noise in distribution of targets (nothing we can do)}}$$

Bias and Variance

$$E[L] = \int \{y(\mathbf{x}) - h(\mathbf{x})\}^2 p(\mathbf{x}) d\mathbf{x} + \int \{h(\mathbf{x}) - t\}^2 p(\mathbf{x}, t) d\mathbf{x} dt$$

- Since y(x) is fit to data, consider expectation over data sets for the part we control

$$E_D\left[\{y(\mathbf{x}; D) - h(\mathbf{x})\}^2\right]$$
$$= \underbrace{\{E_D[y(\mathbf{x}; D) - h(\mathbf{x})]\}^2}_{\text{bias}^2} + \underbrace{E_D\left[\{y(\mathbf{x}; D) - E_D[y(\mathbf{x}; D)]\}^2\right]}_{\text{variance}}$$

Understanding Bias

$$\{E_D[y(\mathbf{x}; D) - h(\mathbf{x})]\}^2$$

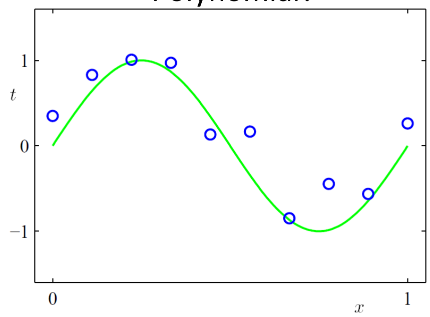
- Measures how well our approximation architecture can fit the data
- Weak approximators (e.g. low degree polynomials) will have high bias
- Strong approximators (e.g. high degree polynomials, will have lower bias)

Understanding Variance

$$E_D\left[\{y(\mathbf{x}; D) - E_D[y(\mathbf{x}; D)]\}^2\right]$$

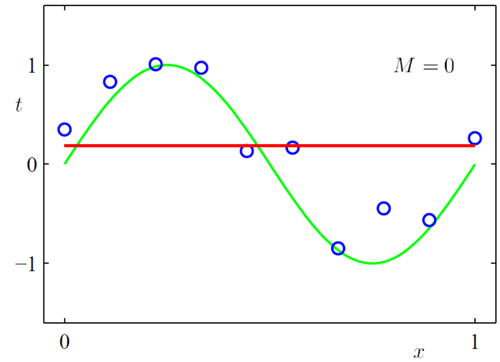
- No *direct* dependence on target values
- For a fixed size D:
  - Strong approximators will tend to have more variance
  - Weak approximators will tend to have less variance
- Variance will typically disappear as size of D goes to infinity

What is the Best Choice of Polynomial?

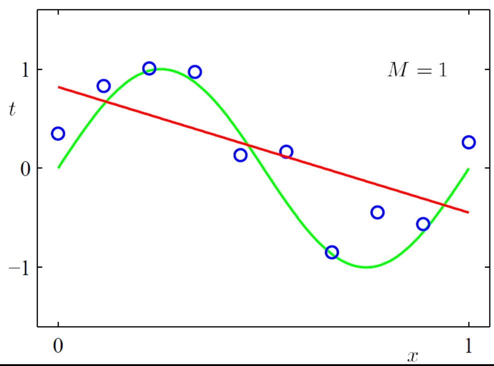


Noisy Source Data

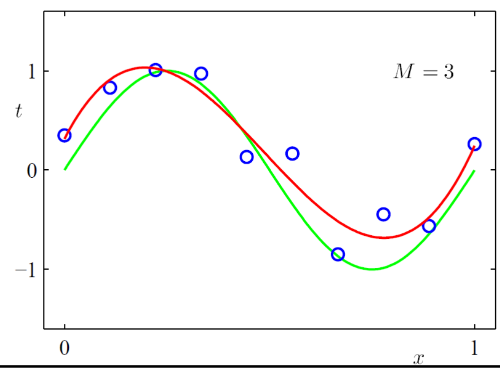
Degree 0 Fit



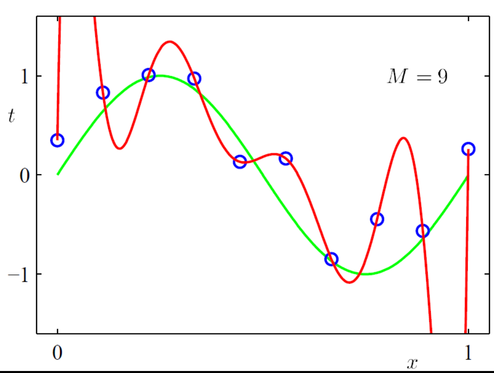
Degree 1 Fit



Degree 3 Fit

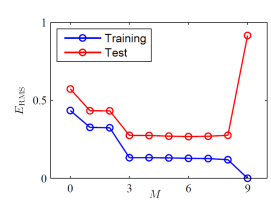


Degree 9 Fit



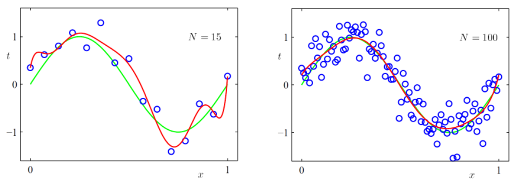
Observations

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



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
- Lower bias -> Higher Variance
- Higher bias -> Lower Variance



Concluding Comments

- Regression is the most basic machine learning algorithm
- Multiple views are all equivalent:
  - Minimize squared loss
  - Maximize likelihood
  - Orthogonal projection
  - Regularization with norm of weights, Bayesian prior
- Bias and variance trade off