Linear Regression

Aarti Singh

Co-instructor: Barnabas Poczos

Machine Learning 10-401
Mar 24, 2016

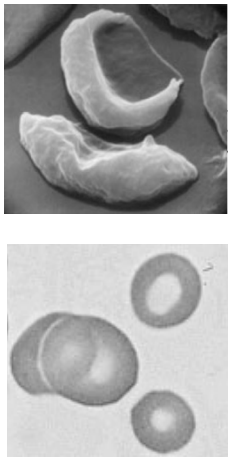
Discrete to Continuous Labels

Classification

X = Document

Sports
Science
News

Y = Topic

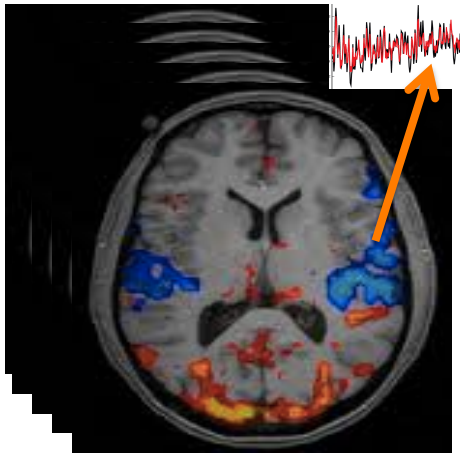


X = Cell Image

Anemic cell
Healthy cell

Y = Diagnosis

Regression

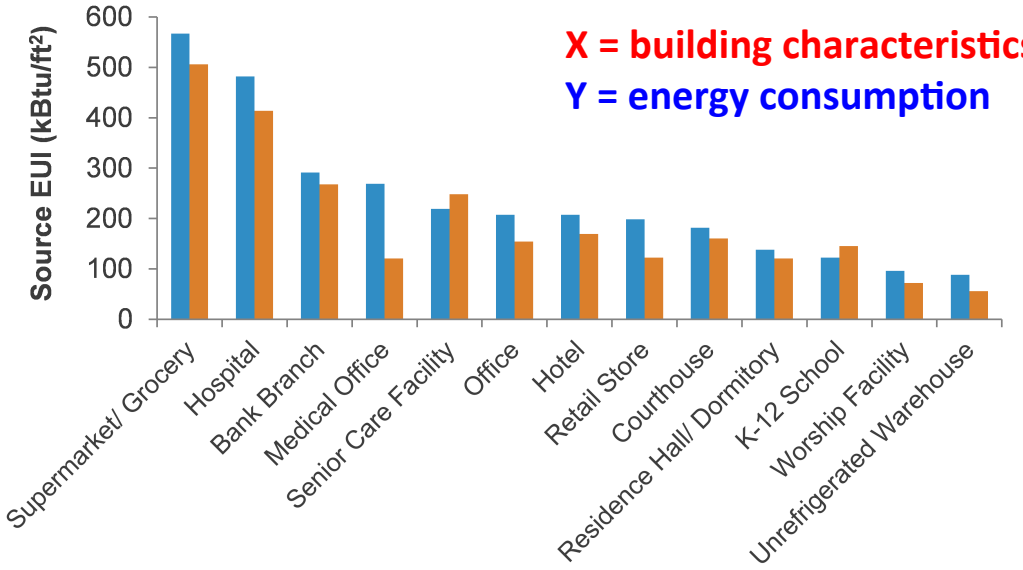


X = Brain Scan

Y = Age of a subject

Regression Tasks

Estimating Energy Usage



X = building characteristics
Y = energy consumption

energystar.gov

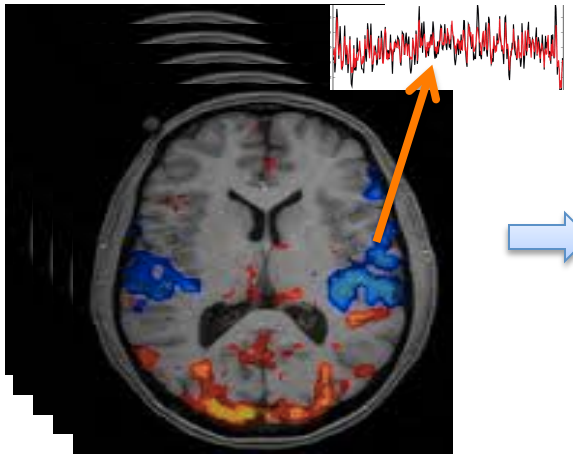
Estimating Contamination

X = new location
Y = sensor reading

Supervised Learning

Goal: Construct a **predictor** $f : X \rightarrow Y$ to minimize loss function (performance measure)

Sports
Science
News



Y = Age of a subject

Classification:

$$P(f(X) \neq Y)$$

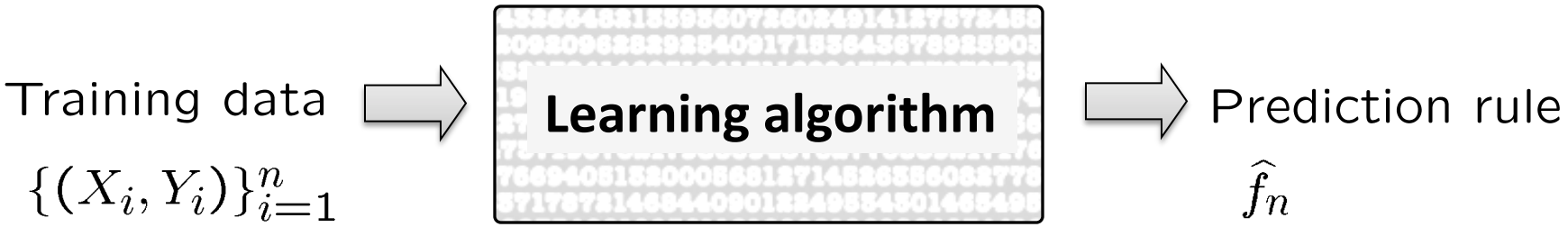
Probability of Error

Regression:

$$\mathbb{E}[(f(X) - Y)^2]$$

Mean Squared Error

Regression algorithms



Linear Regression
Regularized Linear Regression – Ridge regression, Lasso
Polynomial Regression
Kernelized Ridge Regression
Gaussian Process Regression
Kernel regression, Regression Trees, Splines, Wavelet estimators, …

Replace Expectation with Empirical Mean

Optimal predictor:

$$f^* = \arg\min_f \mathbb{E}[(f(X) - Y)^2]$$

Empirical Minimizer:

$$\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2$$

Empirical mean

Law of Large Numbers:

$$\frac{1}{n} \sum_{i=1}^{n} [\text{loss}(Y_i, f(X_i))] \xrightarrow{n \to \infty} \mathbb{E}_{XY} [\text{loss}(Y, f(X))]$$

Restrict class of predictors

Optimal predictor: $$f^* = \arg\min_f \mathbb{E}[(f(X) - Y)^2]$$

Empirical Minimizer: $$\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2$$

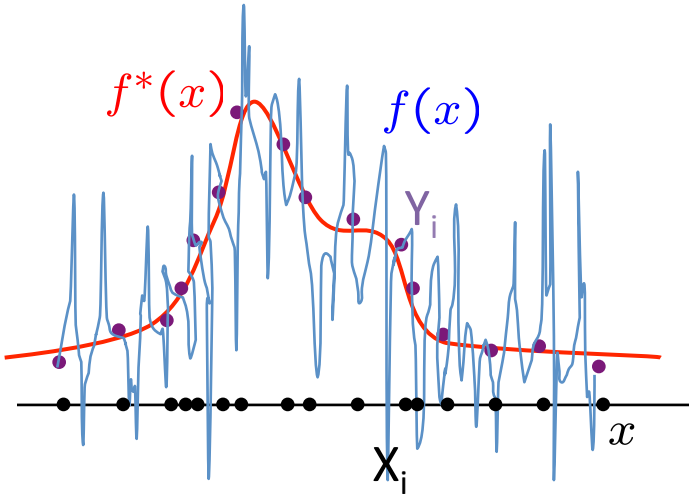
Class of predictors

Why?

Overfitting!

Empiricial loss minimized by any function of the form

$$f(x) = \begin{cases} Y_i, & x = X_i \text{ for } i = 1, \ldots, n \\ \text{any value}, & \text{otherwise} \end{cases}$$



Restrict class of predictors

Optimal predictor: $$f^* = \arg\min_f \mathbb{E}[(f(X) - Y)^2]$$

Empirical Minimizer: $$\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2$$

Class of predictors

$\mathcal{F}$ -
- Class of Linear functions
- Class of Polynomial functions
- Class of nonlinear functions

Linear Regression

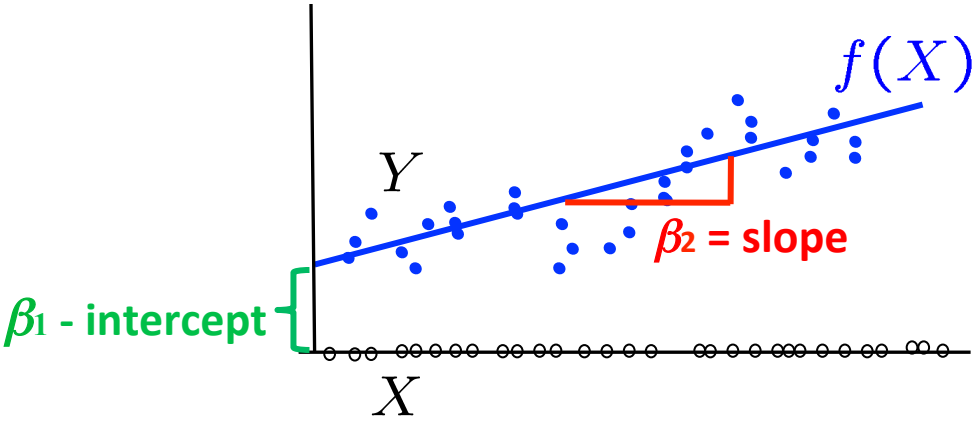
$$\hat{f}_n^L = \arg\min_{f \in \mathcal{F}_L} \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2$$

Least Squares Estimator

$\mathcal{F}_L$ - Class of Linear functions

Uni-variate case:

$$f(X) = \beta_1 + \beta_2 X$$



Multi-variate case:

$$f(X) = f(X^{(1)}, \ldots, X^{(p)}) = \beta_1 X^{(1)} + \beta_2 X^{(2)} + \cdots + \beta_p X^{(p)}$$

$X^{(1)}$ → 1

$$= X\beta \qquad \text{where} \quad X = [X^{(1)} \ldots X^{(p)}], \quad \beta = [\beta_1 \ldots \beta_p]^T$$

Least Squares Estimator

$$\widehat{f}_n^L = \arg\min_{f \in \mathcal{F}_L} \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2 \qquad f(X_i) = X_i \beta$$

$$\widehat{\beta} = \arg\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} (X_i \beta - Y_i)^2 \qquad \widehat{f}_n^L(X) = X\widehat{\beta}$$

$$= \arg\min_{\beta} \frac{1}{n} (\mathbf{A}\beta - \mathbf{Y})^T (\mathbf{A}\beta - \mathbf{Y})$$

$$\mathbf{A} = \begin{bmatrix} X_1 \\ \vdots \\ X_n \end{bmatrix} = \begin{bmatrix} X_1^{(1)} & \dots & X_1^{(p)} \\ \vdots & \ddots & \vdots \\ X_n^{(1)} & \dots & X_n^{(p)} \end{bmatrix} \qquad \mathbf{Y} = \begin{bmatrix} \mathbf{Y}_1 \\ \vdots \\ \mathbf{Y}_n \end{bmatrix}$$

Least Squares Estimator

$$\widehat{\beta} = \arg\min_{\beta} \frac{1}{n}(\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) = \arg\min_{\beta} J(\beta)$$

$$J(\beta) = (\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y})$$

$$\left.\frac{\partial J(\beta)}{\partial \beta}\right|_{\widehat{\beta}} = 0$$

Least Square solution satisfies Normal Equations

$$(\mathbf{A}^T\mathbf{A})\hat{\beta} = \mathbf{A}^T\mathbf{Y}$$

p x p p x1 p x1

If $(\mathbf{A}^T\mathbf{A})$ is invertible,

$$\hat{\beta} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y} \qquad \hat{f}_n^L(X) = X\hat{\beta}$$

Linear Regression

Aarti Singh

Co-instructor: Barnabas Poczos

Machine Learning 10-401
Mar 29, 2016

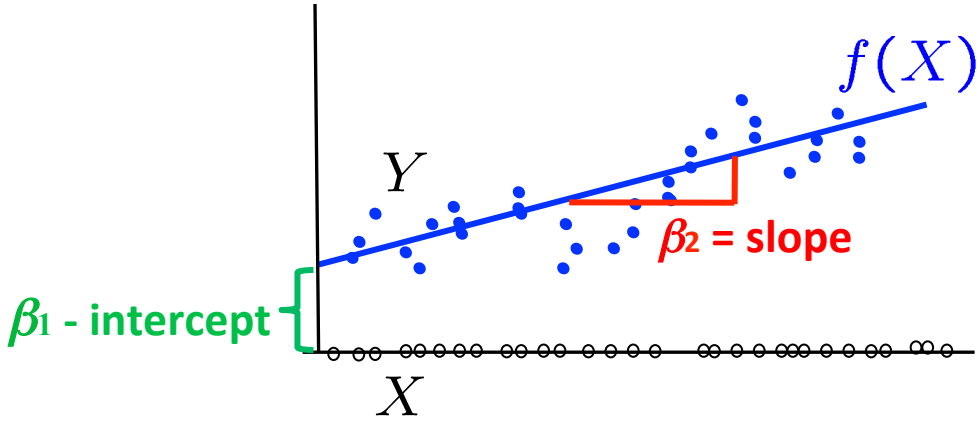
Linear Regression

$$\hat{f}_n^L = \arg\min_{f \in \mathcal{F}_L} \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2$$

Least Squares Estimator

$\mathcal{F}_L$ - Class of Linear functions

$$f(X_i) = X_i \beta$$



$$\hat{\beta} = \arg\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} (X_i \beta - Y_i)^2$$

$$= \arg\min_{\beta} \frac{1}{n} (\mathbf{A}\beta - \mathbf{Y})^T (\mathbf{A}\beta - \mathbf{Y})$$

$$\hat{f}_n^L(X) = X\hat{\beta}$$

Least Square solution satisfies Normal Equations

$$(\mathbf{A}^T\mathbf{A})\hat{\beta} = \mathbf{A}^T\mathbf{Y}$$

p x p p x1 p x1

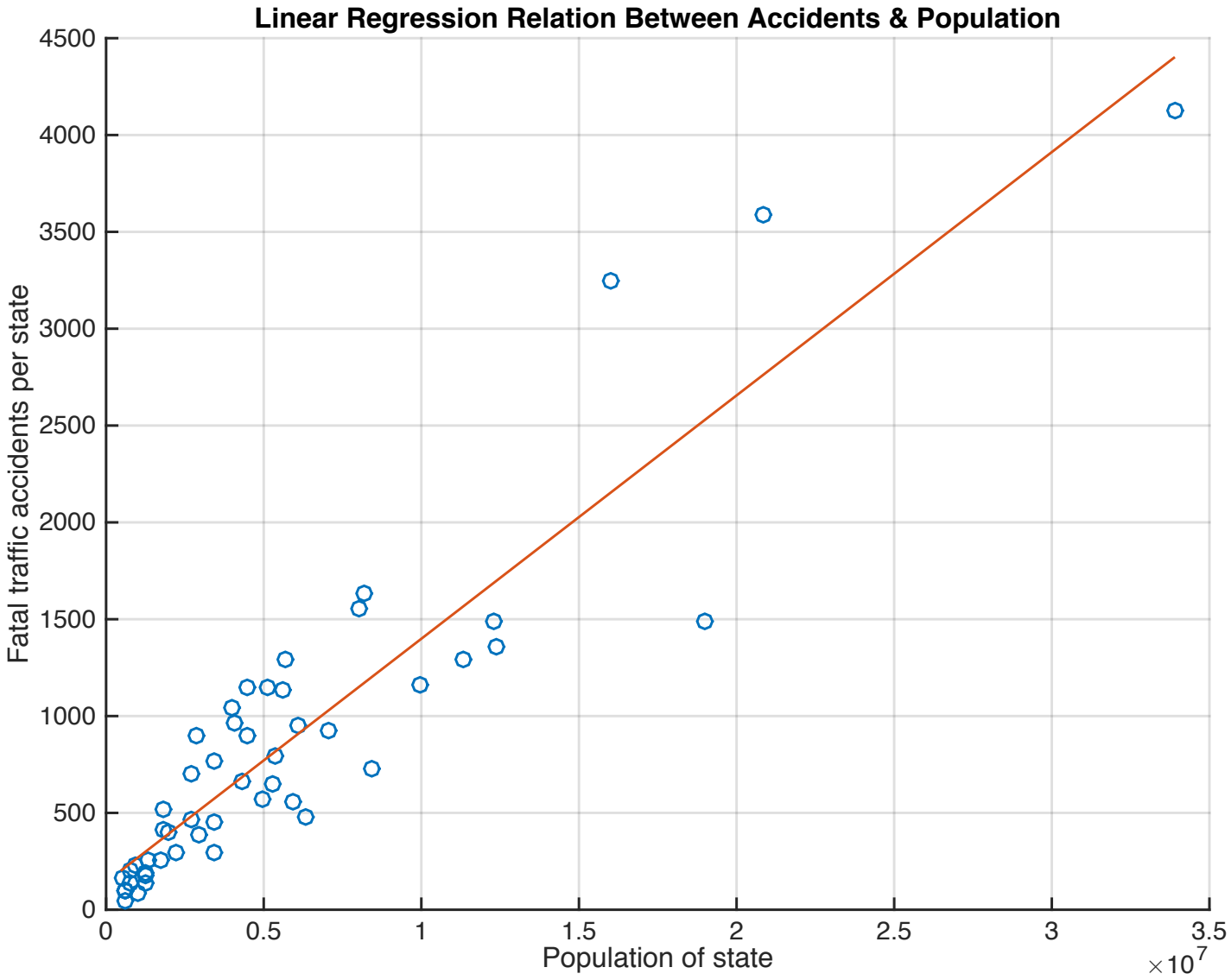
If $(\mathbf{A}^T\mathbf{A})$ is invertible,

$$\hat{\beta} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y} \qquad \hat{f}_n^L(X) = X\hat{\beta}$$

Matlab example – linear regression

```
load accidents
x = hwydata(:,14);                    %Population of states
y = hwydata(:,4);                     %Accidents per state
scatter(x,y)
hold on
X = [ones(length(x),1) x];

b = X\y;
yhat = X*b;
plot(x,yhat)
xlabel('Population of state')
ylabel('Fatal traffic accidents per state')
title('Linear Regression Relation Between Accidents &
Population')
```

Matlab example – linear regression



Least Square solution satisfies Normal Equations

$$(\mathbf{A}^T\mathbf{A})\hat{\beta} = \mathbf{A}^T\mathbf{Y}$$

p x p p x1 p x1

If $(\mathbf{A}^T\mathbf{A})$ is invertible,

$$\hat{\beta} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y} \qquad \hat{f}_n^L(X) = X\hat{\beta}$$

When is $(\mathbf{A}^T\mathbf{A})$ invertible ?
Recall: Full rank matrices are invertible. What is rank of $(\mathbf{A}^T\mathbf{A})$?

Rank$(\mathbf{A}^T\mathbf{A})$ = number of non-zero eigenvalues of $(\mathbf{A}^T\mathbf{A})$ = number of non-zero singular values of **A** <= min(n,p) since **A** is n x p

So, rank$(\mathbf{A}^T\mathbf{A})$, r <= min(n,p) not invertible if r < p (e.g. n < p i.e. high-dimensional setting)

Least Square solution satisfies Normal Equations

$$(\mathbf{A}^T\mathbf{A})\hat{\beta} = \mathbf{A}^T\mathbf{Y}$$

p x p p x1 p x1

If $(\mathbf{A}^T\mathbf{A})$ is invertible,

$$\hat{\beta} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y} \qquad \hat{f}_n^L(X) = X\hat{\beta}$$

When is $(\mathbf{A}^T\mathbf{A})$ invertible ?

Recall: Full rank matrices are invertible. What is rank of $(\mathbf{A}^T\mathbf{A})$?

If $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$, then normal equations $(\mathbf{S}\mathbf{V}^\top)\hat{\beta} = (\mathbf{U}^\top\mathbf{Y})$

S - r x r r x p p x 1 r x 1

r equations in p unknowns. Under-determined if r < p, hence no unique solution.

Least Square solution satisfies Normal Equations

$$(\mathbf{A}^T\mathbf{A})\hat{\beta} = \mathbf{A}^T\mathbf{Y}$$

p x p p x1 p x1

If $(\mathbf{A}^T\mathbf{A})$ is invertible,

$$\hat{\beta} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y} \qquad \hat{f}_n^L(X) = X\hat{\beta}$$

When is $(\mathbf{A}^T\mathbf{A})$ invertible ?
Recall: Full rank matrices are invertible. What is rank of $(\mathbf{A}^T\mathbf{A})$?

What if $(\mathbf{A}^T\mathbf{A})$ is not invertible ?
Constrain solution i.e. Regularization (later)

Now: What if $(\mathbf{A}^T\mathbf{A})$ is invertible but expensive (p very large)?

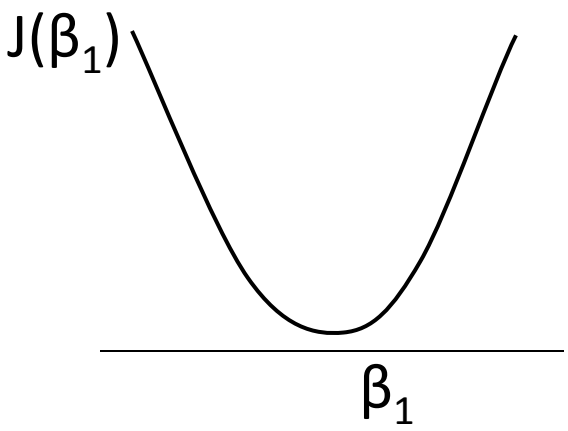
Gradient Descent

Even when $(\mathbf{A}^T\mathbf{A})$ is invertible, might be computationally expensive if **A** is huge.

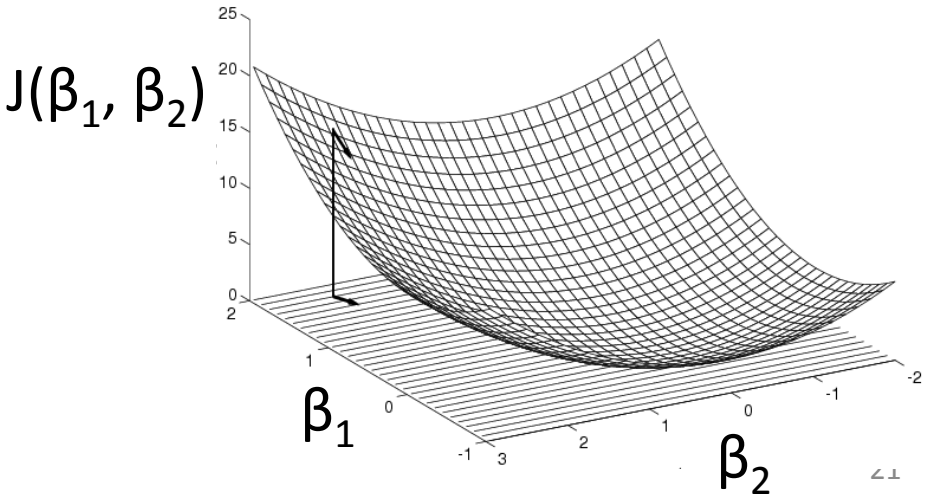
$$\widehat{\beta} = \arg\min_{\beta} \frac{1}{n}(\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) = \arg\min_{\beta} J(\beta)$$

Treat as optimization problem

Observation: J(β) is convex in β.



How to find the minimizer?



Gradient Descent

Even when $(\mathbf{A}^T\mathbf{A})$ is invertible, might be computationally expensive if **A** is huge.

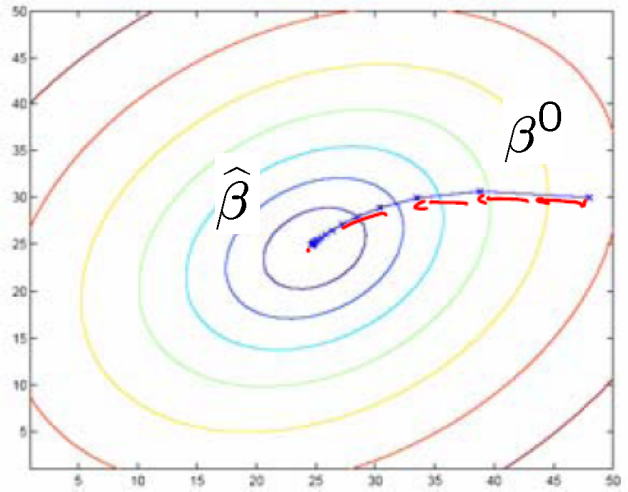
$$\hat{\beta} = \arg\min_{\beta} \frac{1}{n}(\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) = \arg\min_{\beta} J(\beta)$$

Since J(β) is convex, move along negative of gradient

Initialize: β^0

step size

Update:
$$\beta^{t+1} = \beta^t - \frac{\alpha}{2}\frac{\partial J(\beta)}{\partial \beta}\Big|_t$$
$$= \beta^t - \alpha\, \mathbf{A}^T(\underbrace{\mathbf{A}\beta^t - Y}_{0 \text{ if } \hat{\beta} = \beta^t})$$



Stop: when some criterion met e.g. fixed # iterations, or $\frac{\partial J(\beta)}{\partial \beta}\Big|_{\beta^t} < \varepsilon$.

Regularized Least Squares

What if $(\mathbf{A}^T\mathbf{A})$ is not invertible ?

r equations , p unknowns – underdetermined system of linear equations
many feasible solutions

Need to constrain solution further

e.g. bias solution to "small" values of β (small changes in input don't translate to large changes in output)

$$\hat{\beta}_{\text{MAP}} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2$$

Ridge Regression
(l2 penalty)

$$= \arg\min_{\beta} \; (\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) \; + \lambda\|\beta\|_2^2 \qquad \lambda \geq 0$$

$$\hat{\beta}_{\text{MAP}} = (\mathbf{A}^\top\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^\top\mathbf{Y}$$

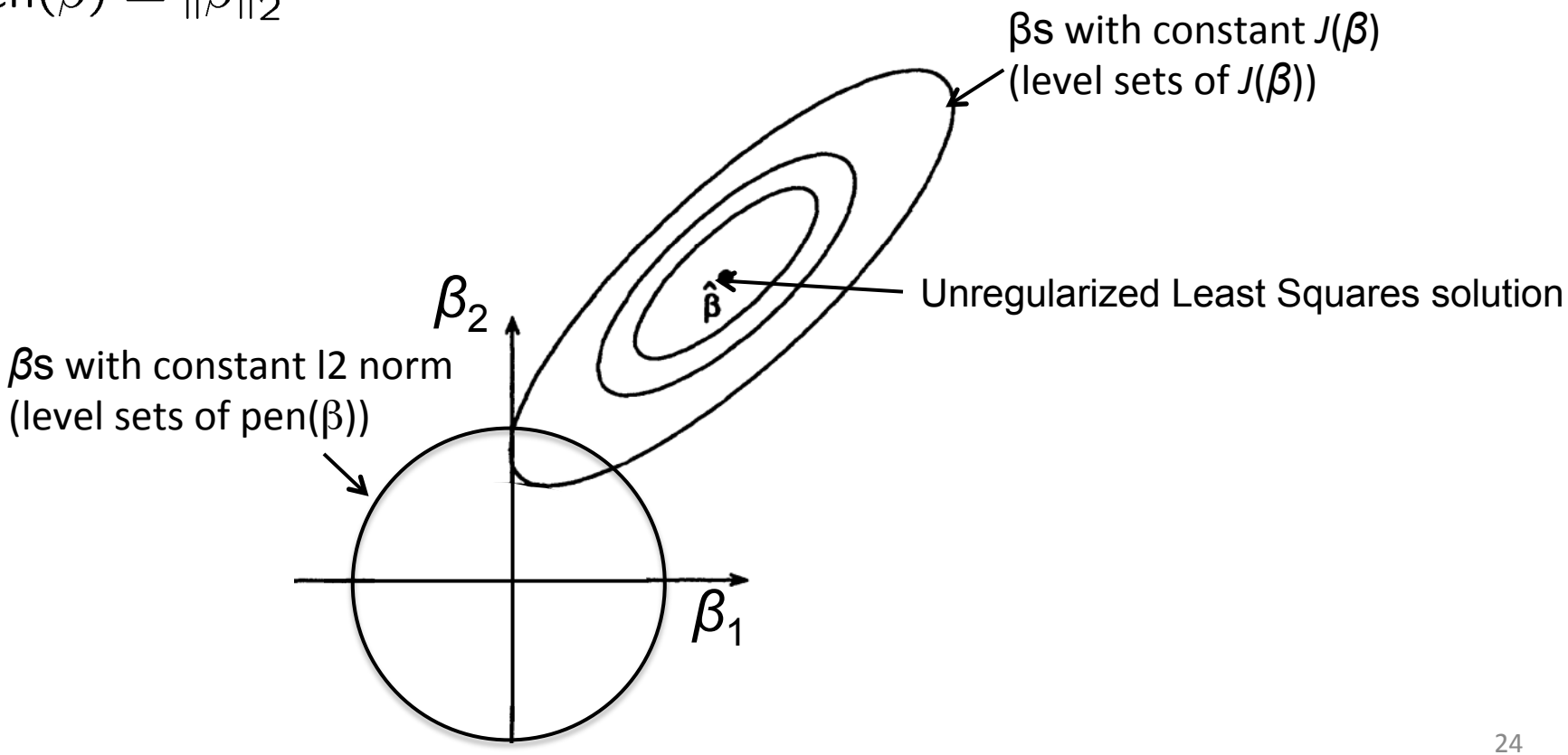
Is $(\mathbf{A}^\top\mathbf{A} + \lambda\mathbf{I})$ invertible ?

Understanding regularized Least Squares

$$\min_{\beta}(\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) + \lambda \text{pen}(\beta) = \min_{\beta} J(\beta) + \lambda \text{pen}(\beta)$$

Ridge Regression:

$$\text{pen}(\beta) = \|\beta\|_2^2$$



Regularized Least Squares

What if $(\mathbf{A}^T\mathbf{A})$ is not invertible ?

r equations , p unknowns – underdetermined system of linear equations
many feasible solutions

Need to constrain solution further

e.g. bias solution to "small" values of β (small changes in input don't translate to large changes in output)

$$\hat{\beta}_{\text{MAP}} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2$$

Ridge Regression
(l2 penalty)

$$\hat{\beta}_{\text{MAP}} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_1 \qquad \lambda \geq 0$$

Lasso
(l1 penalty)

Many β can be zero – many inputs are irrelevant to prediction in high-dimensional settings (typically intercept term not penalized)

Regularized Least Squares

What if $(\mathbf{A}^T\mathbf{A})$ is not invertible ?

r equations , p unknowns – underdetermined system of linear equations
many feasible solutions

Need to constrain solution further

e.g. bias solution to "small" values of β (small changes in input don't translate to large changes in output)

$$\hat{\beta}_{\text{MAP}} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2$$

Ridge Regression
(l2 penalty)

$$\hat{\beta}_{\text{MAP}} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_1$$

$\lambda \geq 0$

Lasso
(l1 penalty)

No closed form solution, but can optimize using sub-gradient descent (packages available)

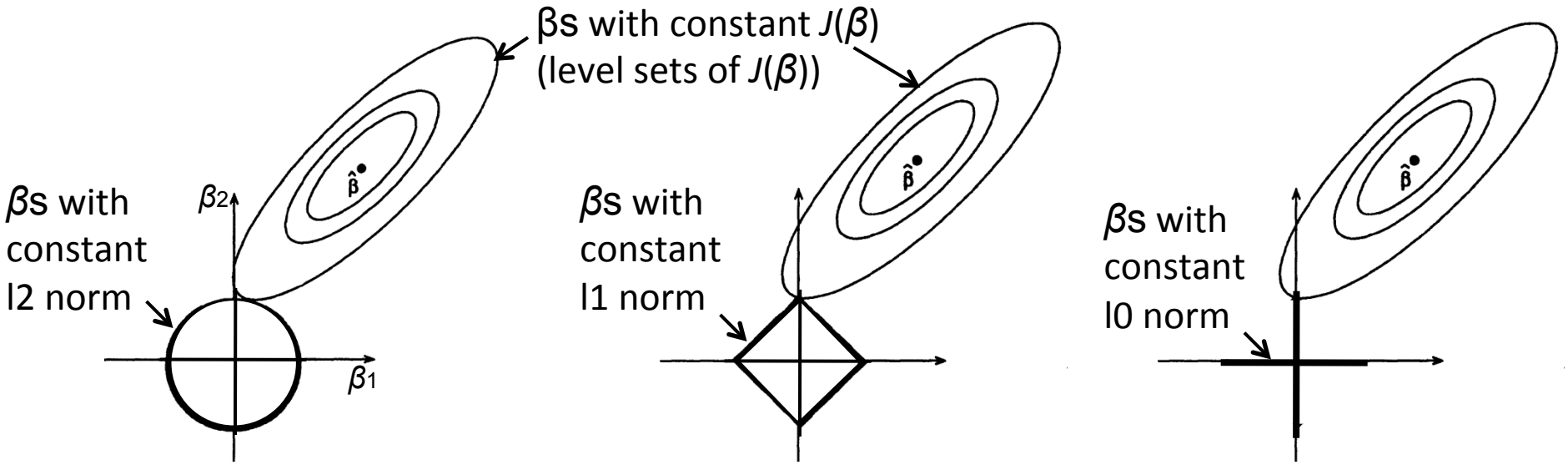
Ridge Regression vs Lasso

$$\min_{\beta}(\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) + \lambda \text{pen}(\beta) = \min_{\beta} J(\beta) + \lambda \text{pen}(\beta)$$

Ridge Regression:
$\text{pen}(\beta) = \|\beta\|_2^2$

Lasso:
$\text{pen}(\beta) = \|\beta\|_1$

Ideally l0 penalty, but optimization becomes non-convex



Lasso (l1 penalty) results in sparse solutions – vector with more zero coordinates
Good for high-dimensional problems – don't have to store all coordinates, interpretable solution!

Matlab example

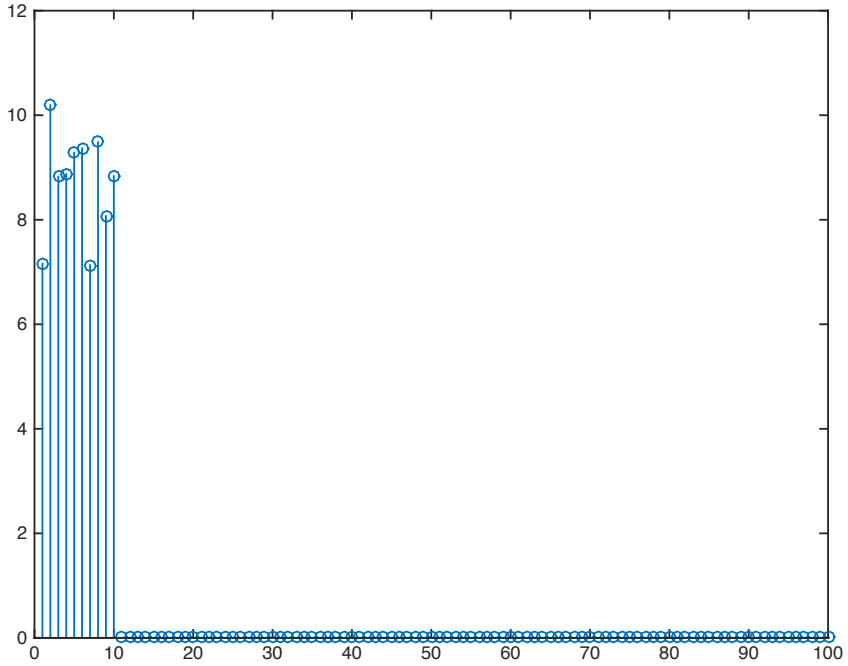
```
clear all
close all

n = 80;     % datapoints
p = 100;   % features
k = 10;      % non-zero features

rng(20);
X = randn(n,p);
weights = zeros(p,1);
weights(1:k) = randn(k,1)+10;
noise = randn(n,1) * 0.5;
Y = X*weights +  noise;

Xtest = randn(n,p);
noise = randn(n,1) * 0.5;
Ytest = Xtest*weights + noise;
```

```
lassoWeights = lasso(X,Y,'Lambda',1,
'Alpha', 1.0);
Ylasso = Xtest*lassoWeights;
norm(Ytest-Ylasso)

ridgeWeights = lasso(X,Y,'Lambda',1,
'Alpha', 0.0001);
Yridge = Xtest*ridgeWeights;
norm(Ytest-Yridge)

stem(lassoWeights)
pause
stem(ridgeWeights)
```

Matlab example

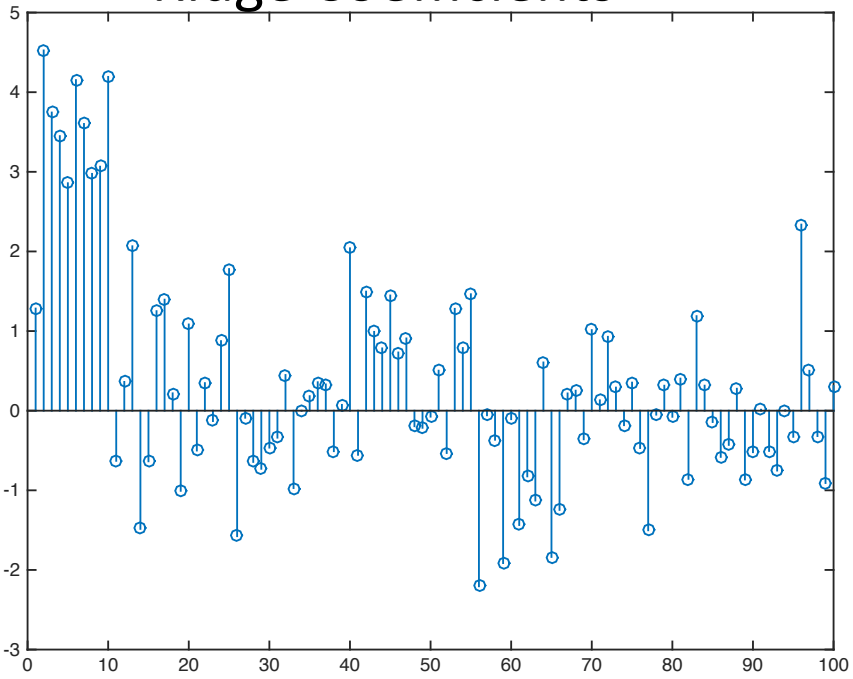
Test MSE = 33.7997

Lasso Coefficients



Test MSE = 185.9948

Ridge Coefficients



Regularized Least Squares Regression

Aarti Singh

Co-instructor: Barnabas Poczos

Machine Learning 10-401
Mar 31, 2016

Regularized Least Squares

What if $(\mathbf{A}^T\mathbf{A})$ is not invertible ?

r equations , p unknowns – underdetermined system of linear equations
many feasible solutions

Need to constrain solution further

e.g. bias solution to "small" values of β (small changes in input don't translate to large changes in output)

$$\widehat{\beta}_{\text{MAP}} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2$$

Ridge Regression
(l2 penalty)

$$\widehat{\beta}_{\text{MAP}} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_1$$

$\lambda \geq 0$

Lasso
(l1 penalty)

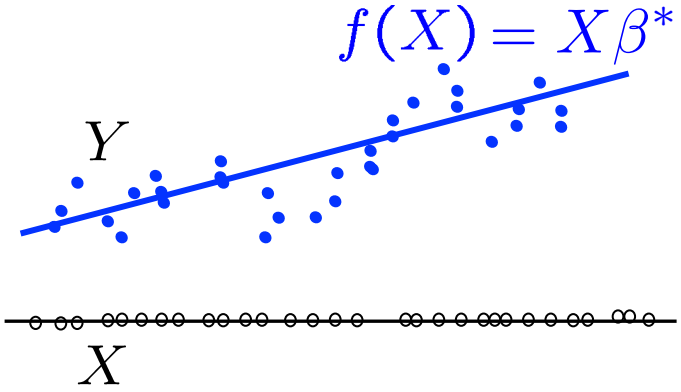
Many β can be zero – many inputs are irrelevant to prediction in high-dimensional settings (typically intercept term not penalized)

Regularized Least Squares – connection to MLE and MAP

Least Squares and M(C)LE

Intuition: Signal plus (zero-mean) Noise model

$$Y = f^*(X) + \epsilon = X\beta^* + \epsilon$$



$$\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I}) \quad Y \sim \mathcal{N}(X\beta^*, \sigma^2 \mathbf{I})$$

$$\widehat{\beta}_{\mathsf{MLE}} = \arg\max_{\beta} \underbrace{\log p(\{Y_i\}_{i=1}^n | \beta, \sigma^2, \{X_i\}_{i=1}^n)}_{\text{Conditional log likelihood}}$$

$$= \arg\min_{\beta} \sum_{i=1}^{n} (X_i\beta - Y_i)^2 = \widehat{\beta}$$

Least Square Estimate is same as Maximum Conditional Likelihood Estimate under a Gaussian model !

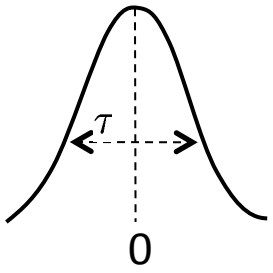
Regularized Least Squares and M(C)AP

What if $(\mathbf{A}^T\mathbf{A})$ is not invertible ?

$$\widehat{\beta}_{\text{MAP}} = \arg\max_{\beta} \underbrace{\log p(\{Y_i\}_{i=1}^n | \beta, \sigma^2, \{X_i\}_{i=}^n}_{\text{Conditional log likelihood}} + \underbrace{\log p(\beta)}_{\text{log prior}}$$

I) Gaussian Prior

$$\beta \sim \mathcal{N}(0, \tau^2\mathbf{I}) \qquad p(\beta) \propto e^{-\beta^T\beta/2\tau^2}$$



$$\widehat{\beta}_{\text{MAP}} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2$$

$\lambda \downarrow$ constant(σ^2, τ^2)

Ridge Regression

$$\widehat{\beta}_{\text{MAP}} = (\boldsymbol{A}^\top\boldsymbol{A} + \lambda\boldsymbol{I})^{-1}\boldsymbol{A}^\top\boldsymbol{Y}$$

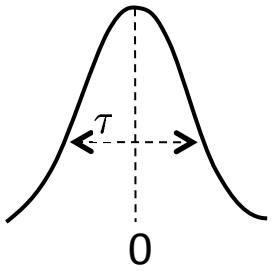
Regularized Least Squares and M(C)AP

What if $(\mathbf{A}^T\mathbf{A})$ is not invertible ?

$$\widehat{\beta}_{\text{MAP}} = \arg\max_{\beta} \underbrace{\log p(\{Y_i\}_{i=1}^n | \beta, \sigma^2, \{X_i\}_{i=}^n}_{\text{Conditional log likelihood}} + \underbrace{\log p(\beta)}_{\text{log prior}}$$

I) Gaussian Prior

$$\beta \sim \mathcal{N}(0, \tau^2\mathbf{I}) \qquad p(\beta) \propto e^{-\beta^T\beta/2\tau^2}$$



$$\widehat{\beta}_{\text{MAP}} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2$$

$\lambda \rightarrow$ constant(σ^2, τ^2)

Ridge Regression

Prior belief that β is Gaussian with zero-mean biases solution to "small" β

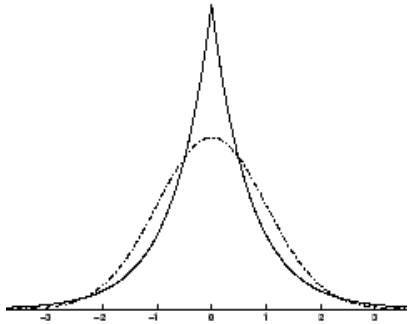
Regularized Least Squares and M(C)AP

What if $(\mathbf{A}^T\mathbf{A})$ is not invertible ?

$$\widehat{\beta}_{\text{MAP}} = \arg\max_{\beta} \underbrace{\log p(\{Y_i\}_{i=1}^n | \beta, \sigma^2, \{X_i\}_{i=}^n}_{\text{Conditional log likelihood}} + \underbrace{\log p(\beta)}_{\text{log prior}}$$

II) Laplace Prior

$$\beta_i \overset{iid}{\sim} \text{Laplace}(0, t) \qquad p(\beta_i) \propto e^{-|\beta_i|/t}$$



$$\widehat{\beta}_{\text{MAP}} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda \|\beta\|_1$$

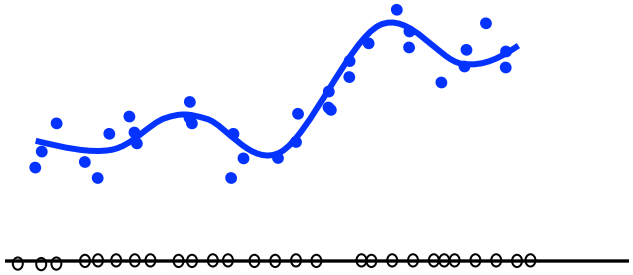
$\lambda \|\beta\|_1 \downarrow \text{constant}(\sigma^2, t)$

Lasso

Prior belief that β is Laplace with zero-mean biases solution to "sparse" β

Beyond Linear Regression

Polynomial regression
Regression with nonlinear features



Kernelized Ridge Regression

Local Kernel Regression

Polynomial Regression

degree m

Univariate (1-dim) case: $f(X) = \beta_0 + \beta_1 X + \beta_2 X^2 + \cdots + \beta_m X^m = \mathbf{X}\beta$

where $\mathbf{X} = [1 \;\; X \;\; X^2 \ldots X^m]$, $\beta = [\beta_1 \ldots \beta_m]^T$

$$\widehat{\beta} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y} \quad \text{or} \quad (\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{Y} \qquad \widehat{f}_n(X) = \mathbf{X}\widehat{\beta}$$

where $\mathbf{A} = \begin{bmatrix} 1 & X_1 & X_1^2 & \ldots & X_1^m \\ \vdots & & & \ddots & \vdots \\ 1 & X_n & X_n^2 & \ldots & X_n^m \end{bmatrix}$

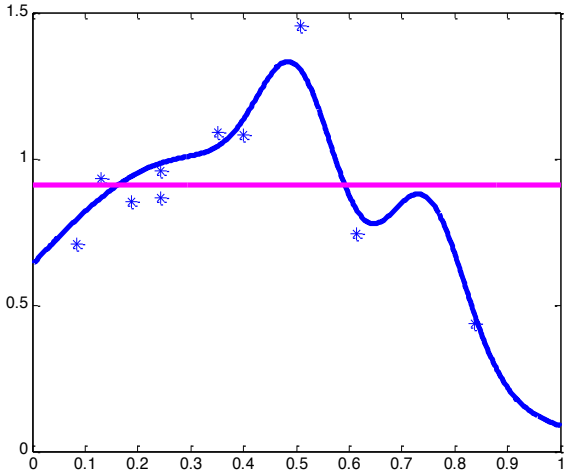
Multivariate (p-dim) case:

$$f(X) = \beta_0 + \beta_1 X^{(1)} + \beta_2 X^{(2)} + \cdots + \beta_p X^{(p)} + \sum_{i=1}^{p}\sum_{j=1}^{p} \beta_{ij} X^{(i)} X^{(j)} + \sum_{i=1}^{p}\sum_{j=1}^{p}\sum_{k=1}^{p} X^{(i)} X^{(j)} X^{(k)} + \ldots \text{terms up to degree m}$$

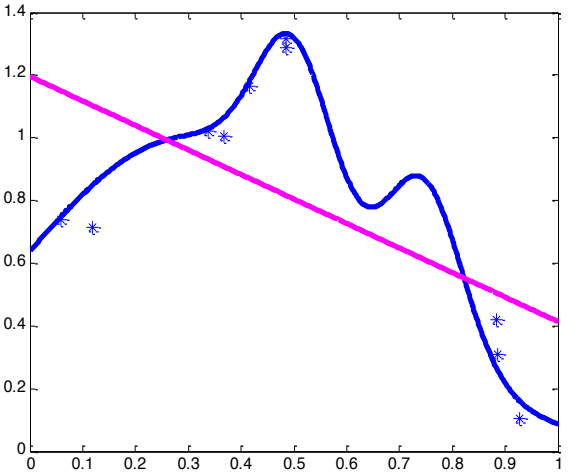
Polynomial Regression

Polynomial of order k, equivalently of degree up to k-1

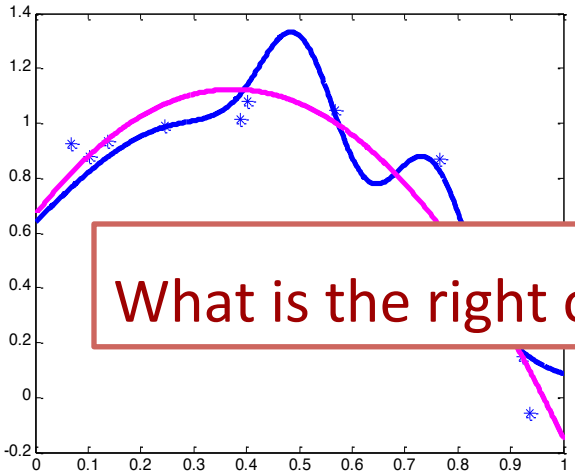
k=1



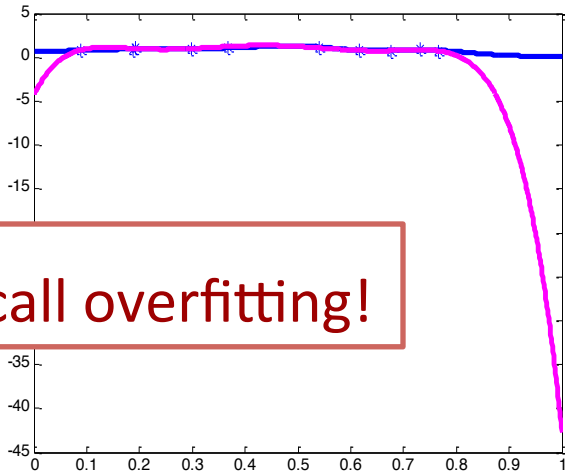
k=2



k=3



k=7

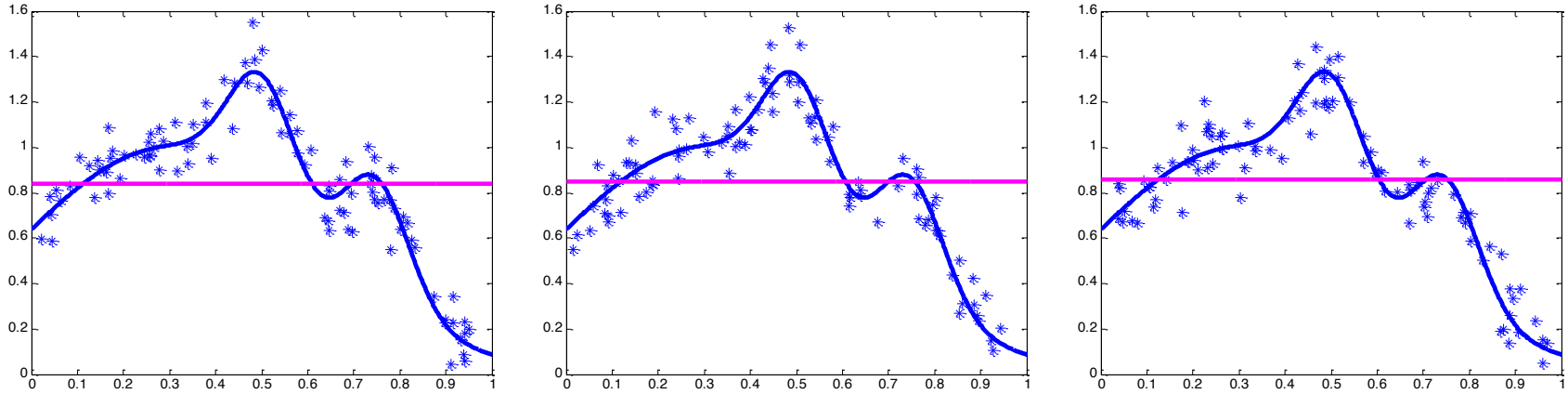


What is the right order? Recall overfitting!

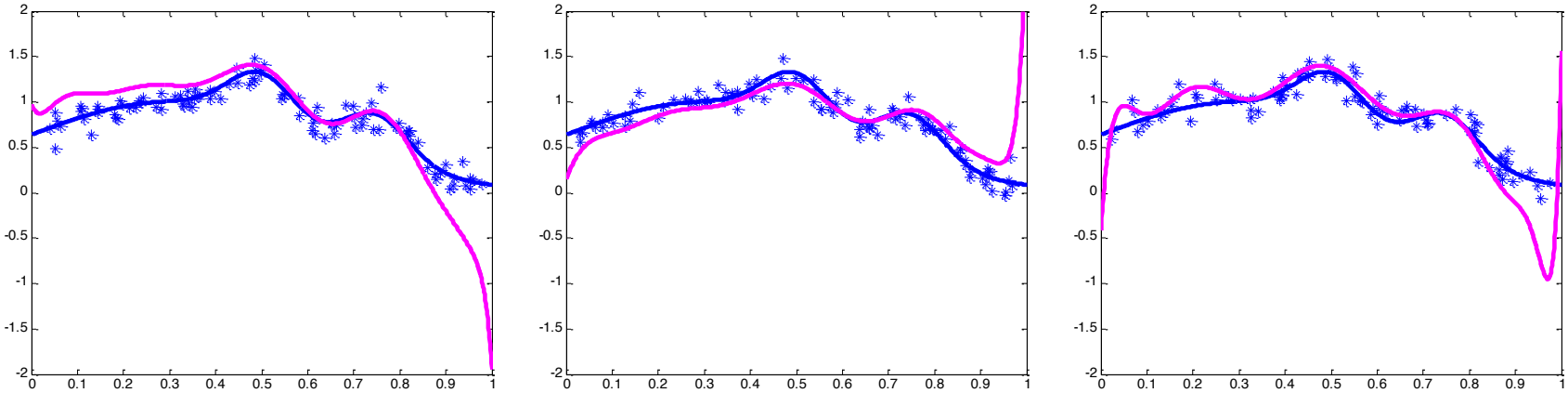
Bias – Variance Tradeoff

3 Independent training datasets

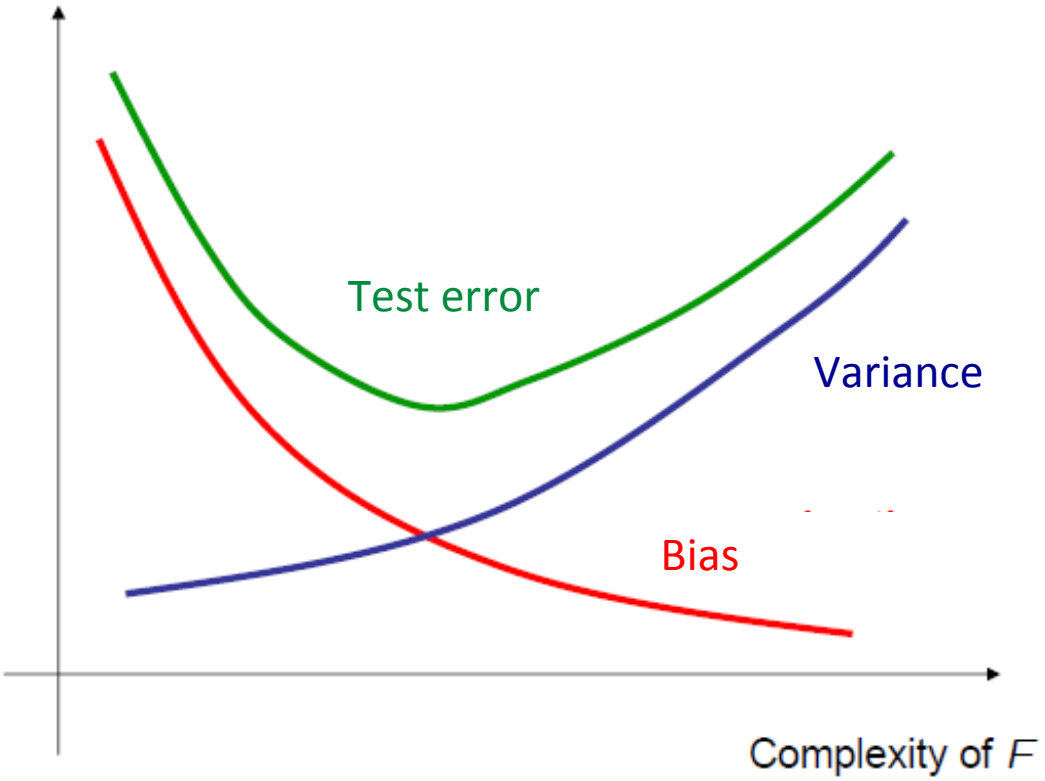
Large bias, Small variance – poor approximation but robust/stable



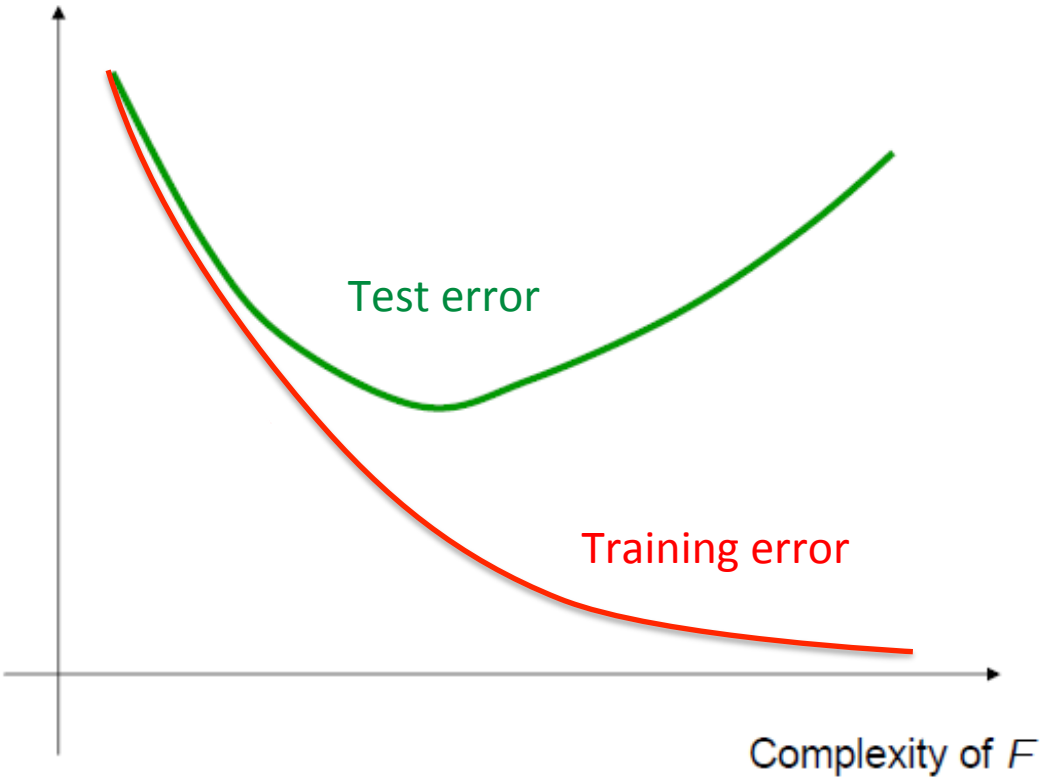
Small bias, Large variance – good approximation but unstable



Effect of Model Complexity



Effect of Model Complexity



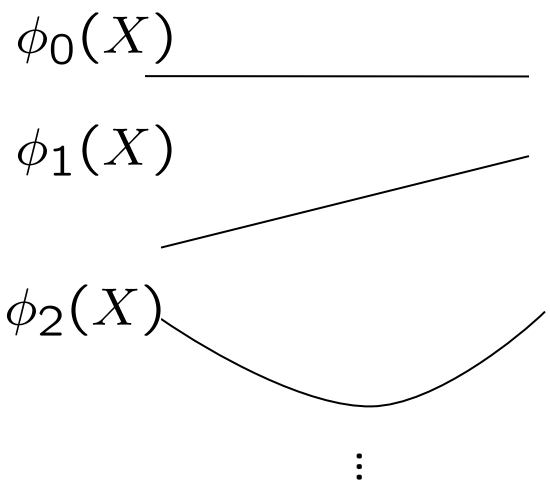
Regression with basis functions

$$f(X) = \sum_{j=0}^{m} \beta_j \phi_j(X)$$

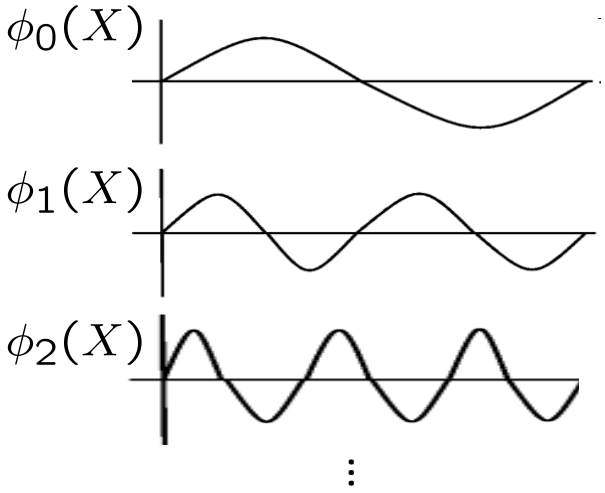
Basis coefficients

Basis functions (Linear combinations yield meaningful spaces)

Polynomial Basis

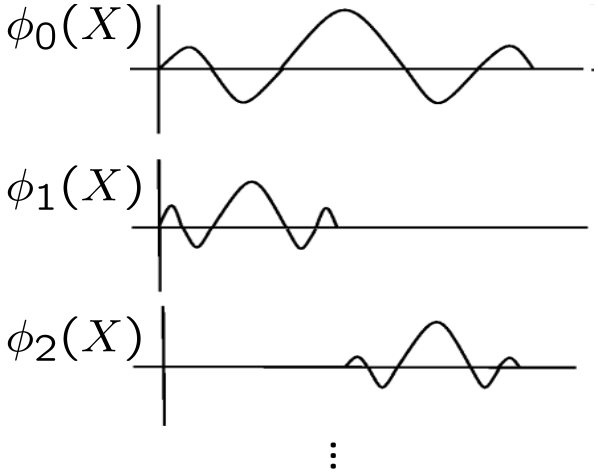


Fourier Basis



Good representation for periodic functions

Wavelet Basis



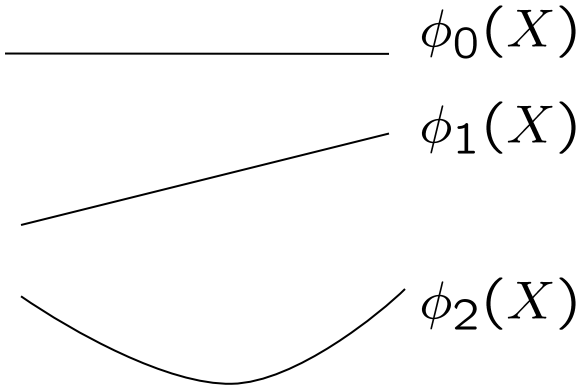
Good representation for local functions

Regression with nonlinear features

$$f(X) = \sum_{j=0}^{m} \beta_j X^j = \sum_{j=0}^{m} \beta_j \phi_j(X)$$

Weight of each feature

Nonlinear features



In general, use any nonlinear features

e.g. e^X, log X, 1/X, sin(X), …

$$\widehat{\beta} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y}$$

or

$$(\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{Y}$$

$$\mathbf{A} = \begin{bmatrix} \phi_0(X_1) & \phi_1(X_1) & \dots & \phi_m(X_1) \\ \vdots & & \ddots & \vdots \\ \phi_0(X_n) & \phi_1(X_n) & \dots & \phi_m(X_n) \end{bmatrix}$$

$$\widehat{f}_n(X) = \mathbf{X}\widehat{\beta}$$

$$\mathbf{X} = [\phi_0(X) \;\; \phi_1(X) \;\; \dots \;\; \phi_m(X)]$$

Can we use kernels?

Ridge regression (dual)

$$\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2 \qquad \widehat{\beta} = (\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{Y}$$

Similarity with SVMs

Primal problem:

$$\min_{\beta, z_i} \sum_{i=1}^{n} z_i^2 + \lambda\|\beta\|_2^2$$
$$\text{s.t. } z_i = Y_i - X_i\beta$$

SVM Primal problem:

$$\min_{w, \xi_i} C\sum_{i=1}^{n} \xi_i + \frac{1}{2}\|w\|_2^2$$
$$\text{s.t. } \xi_i = \max(1 - Y_i X_i\, w, 0)$$

Lagrangian:

$$\sum_{i=1}^{n} z_i^2 + \lambda\|\beta\|^2 + \sum_{i=1}^{n} \alpha_i(z_i - Y_i + X_i\beta)$$

α_i – Lagrange parameter, one per training point

Ridge regression (dual)

$$\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2 \qquad \widehat{\beta} = (\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{Y}$$

Dual problem:

$$\max_{\alpha} \min_{\beta, z_i} \sum_{i=1}^{n} z_i^2 + \lambda\|\beta\|^2 + \sum_{i=1}^{n} \alpha_i(z_i - Y_i + X_i\beta)$$

$\alpha = \{\alpha_i\}$ for i = 1,…, n

Taking derivatives of Lagrangian wrt β and z_i we get:

$$\beta = -\frac{1}{2\lambda}\mathbf{A}^\top\alpha \qquad z_i = -\frac{\alpha_i}{2}$$

Dual problem: $$\max_{\alpha} \; -\frac{\alpha^\top\alpha}{4} - \frac{1}{4\lambda}\alpha^\top\mathbf{A}\mathbf{A}^\top\alpha - \alpha^\top\mathbf{Y}$$

n-dimensional optimization problem

Ridge regression (dual)

$$\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2$$

$$\widehat{\beta} = (\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{Y}$$
$$= \mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \lambda\mathbf{I})^{-1}\mathbf{Y}$$

Dual problem:

$$\max_{\alpha} \; -\frac{\alpha^\top\alpha}{4} - \frac{1}{4\lambda}\alpha^\top\mathbf{A}\mathbf{A}^\top\alpha - \alpha^\top\mathbf{Y} \qquad \Rightarrow \widehat{\alpha} = -\left(\frac{\mathbf{A}\mathbf{A}^\top}{\lambda} + \mathbf{I}\right)^{-1} 2\,\mathbf{Y}$$

can get back $\hat{\beta} = -\frac{1}{2\lambda}\mathbf{A}^\top\hat{\alpha} = \mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top + \lambda\mathbf{I})^{-1}\mathbf{Y}$

Weighted average of training points

Weight of each training point

Kernelized ridge regression

$$\widehat{\beta} = (\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{Y} \qquad \widehat{f}_n(X) = \mathbf{X}\widehat{\beta}$$

Using dual, can re-write solution as:

$$\widehat{\beta} = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \lambda\mathbf{I})^{-1}\mathbf{Y}$$

How does this help?

- Only need to invert n x n matrix (instead of p x p or m x m)
- More importantly, kernel trick!

$$\widehat{f}_n(X) = \mathbf{K}_X(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{Y} \quad \text{where} \quad \begin{aligned} \mathbf{K}_X(i) &= \boldsymbol{\phi}(X) \cdot \boldsymbol{\phi}(X_i) \\ \mathbf{K}(i,j) &= \boldsymbol{\phi}(X_i) \cdot \boldsymbol{\phi}(X_j) \end{aligned}$$

Work with kernels, never need to write out the high-dim vectors

Kernelized ridge regression

$$\widehat{f}_n(X) = \mathbf{K}_X(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{Y}$$ where $$\mathbf{K}_X(i) = \boldsymbol{\phi}(X) \cdot \boldsymbol{\phi}(X_i)$$ $$\mathbf{K}(i,j) = \boldsymbol{\phi}(X_i) \cdot \boldsymbol{\phi}(X_j)$$

Work with kernels, never need to write out the high-dim vectors

Examples of kernels:

Polynomials of degree exactly d $K(\mathbf{u}, \mathbf{v}) = (\mathbf{u} \cdot \mathbf{v})^d$

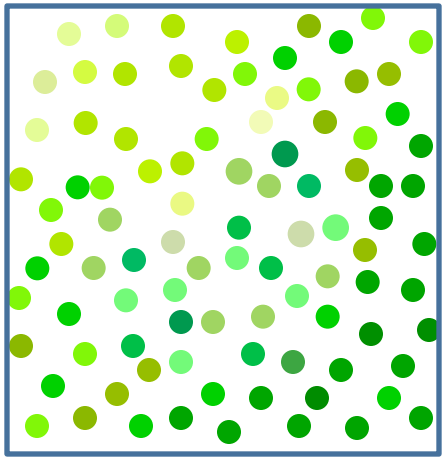
Polynomials of degree up to d $K(\mathbf{u}, \mathbf{v}) = (\mathbf{u} \cdot \mathbf{v} + 1)^d$

Gaussian/Radial kernels $K(\mathbf{u}, \mathbf{v}) = \exp\left(-\frac{||\mathbf{u} - \mathbf{v}||^2}{2\sigma^2}\right)$

Ridge Regression with (implicit) nonlinear features $\boldsymbol{\phi}(X)$! $f(X) = \boldsymbol{\phi}(X)\beta$

Local Kernel Regression

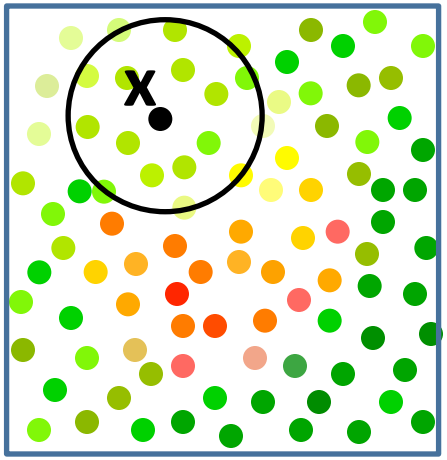
- What is the temperature in the room?



$$\widehat{T} = \frac{1}{n} \sum_{i=1}^{n} Y_i$$

Average

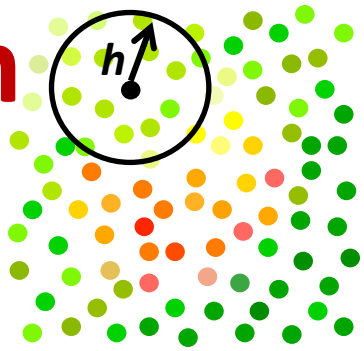
at location x?



$$\widehat{T}(x) = \frac{\sum_{i=1}^{n} Y_i 1_{||X_i - x|| \leq h}}{\sum_{i=1}^{n} 1_{||X_i - x|| \leq h}}$$

"Local" Average

Local Kernel Regression

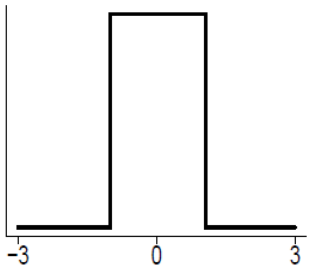


- Nonparametric estimator akin to kNN
- Nadaraya-Watson Kernel Estimator

$$\widehat{f}_n(X) = \sum_{i=1}^{n} w_i Y_i \qquad \text{Where} \qquad w_i(X) = \frac{K\left(\frac{X - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X - X_i}{h}\right)}$$

- Weight each training point based on distance to test point
- Boxcar kernel yields local average

$$\text{boxcar kernel}: \quad K(x) = \frac{1}{2} I(x),$$



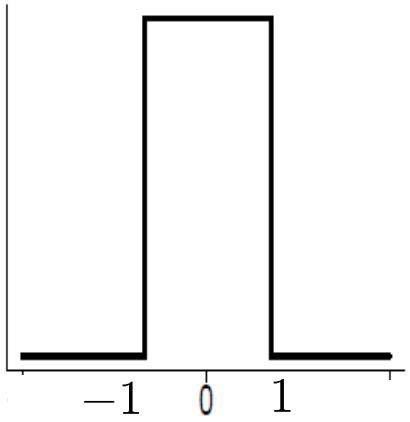
Kernels

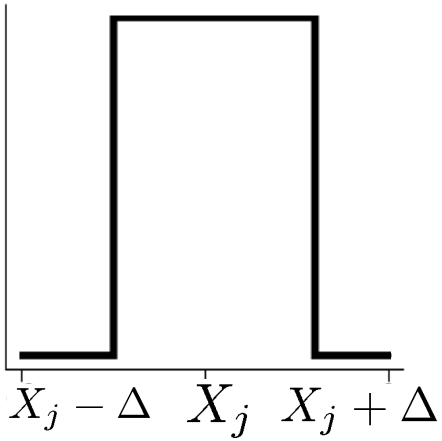
$$K(x) \geq 0,$$
$$\int K(x)dx = 1$$

$$K\left(\frac{X_j - x}{\Delta}\right)$$

boxcar kernel :

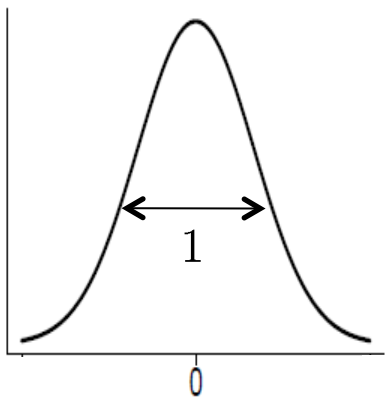
$$K(x) = \frac{1}{2}I(x),$$

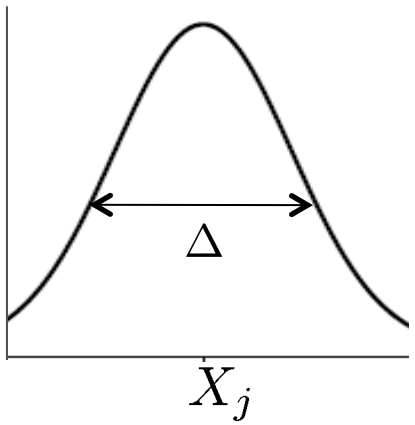




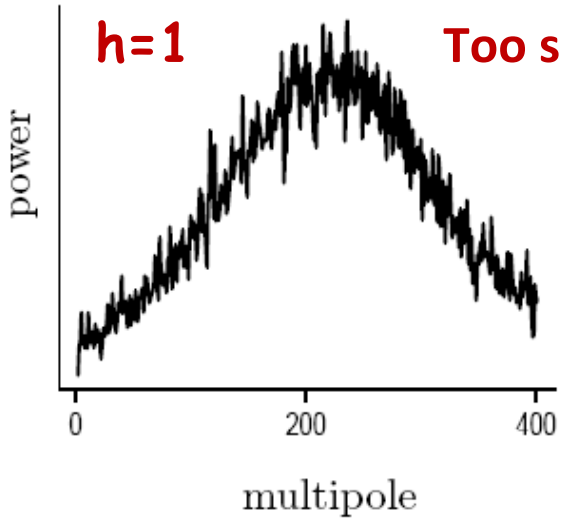
Gaussian kernel :

$$K(x) = \frac{1}{\sqrt{2\pi}}e^{-x^2/2}$$





Choice of kernel bandwidth h



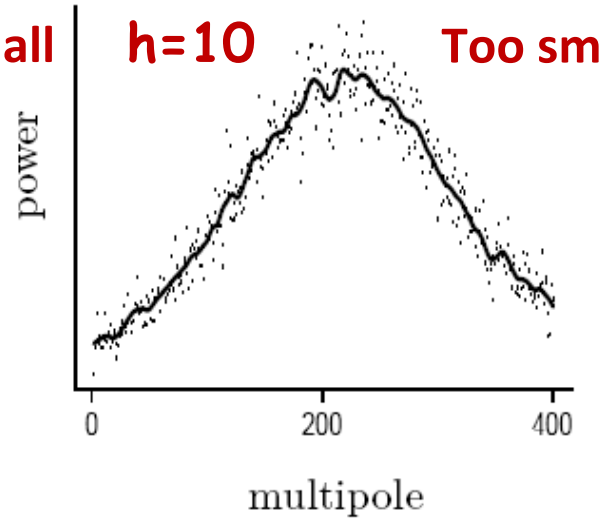
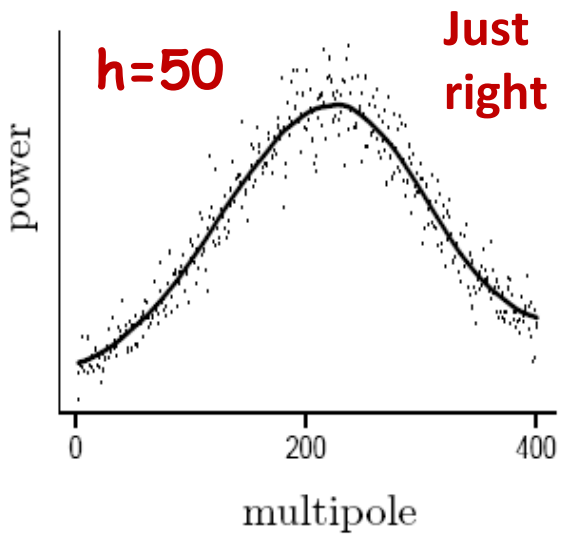
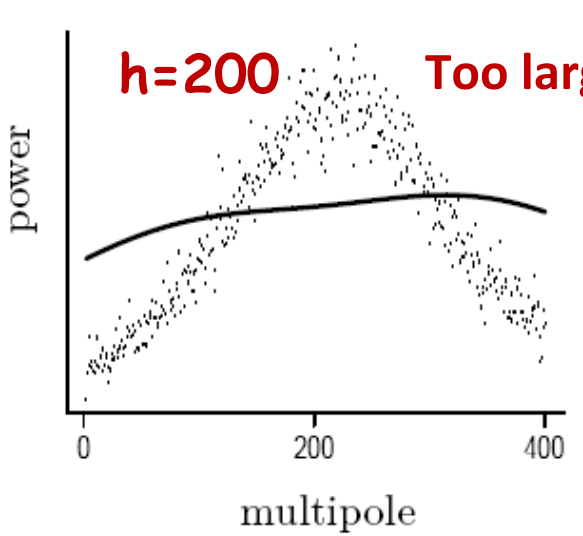


Image Source: Larry's book – All of Nonparametric Statistics





Choice of kernel is not that important

Kernel Regression as Weighted Least Squares

$$\min_f \sum_{i=1}^{n} w_i (f(X_i) - Y_i)^2 \qquad w_i(X) = \frac{K\left(\frac{X - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X - X_i}{h}\right)}$$

Weighted Least Squares

Kernel regression corresponds to locally constant estimator obtained from (locally) weighted least squares

i.e. set $f(X_i) = \beta$ (a constant)

Kernel Regression as Weighted Least Squares

set $f(X_i) = \beta$ (a constant)

$$\min_{\beta} \sum_{i=1}^{n} w_i(\beta - Y_i)^2$$

β: constant

$$w_i(X) = \frac{K\left(\frac{X - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X - X_i}{h}\right)}$$

$$\frac{\partial J(\beta)}{\partial \beta} = 2\sum_{i=1}^{n} w_i(\beta - Y_i) = 0$$

Notice that $\sum_{i=1}^{n} w_i = 1$

$$\Rightarrow \hat{f}_n(X) = \hat{\beta} = \sum_{i=1}^{n} w_i Y_i$$

Local Linear/Polynomial Regression

$$\min_f \sum_{i=1}^{n} w_i(f(X_i) - Y_i)^2 \qquad w_i(X) = \frac{K\left(\frac{X-X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X-X_i}{h}\right)}$$

Weighted Least Squares

Local Polynomial regression corresponds to locally polynomial estimator obtained from (locally) weighted least squares

$$f(X_i) = \beta_0 + \beta_1(X_i - X) + \frac{\beta_2}{2!}(X_i - X)^2 + \cdots + \frac{\beta_p}{p!}(X_i - X)^p$$

i.e. set

(local polynomial of degree p around X)

What you should know

Linear Regression

- Least Squares Estimator
- Normal Equations
- Gradient Descent
- Probabilistic Interpretation (connection to MCLE)

Regularized Linear Regression (connection to MCAP)

- Ridge Regression, Lasso

Beyond Linear

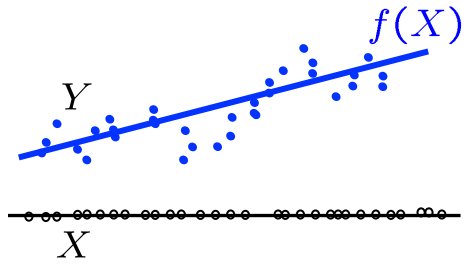
- Polynomial regression, Regression with Basis functions and Non-linear features, Bias-variance tradeoff, Kernelized ridge regression, Local Kernel Regression and Weighted Least Squares

Prior for kernelized ridge regression

- Recall ridge regression was obtained as MAP under a Gaussian prior for a linear model with Gaussian noise

$$Y = f(X) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

$$f(X) = X\beta \quad \beta \sim \mathcal{N}(0, \tau^2 \mathbf{I})$$



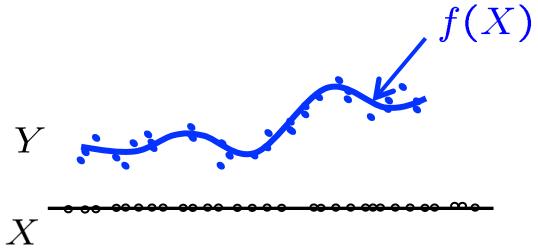
- For kernelized ridge regression, what's the prior?

$$Y = f(X) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

$$f(X) = \phi(X)\beta \quad \beta \sim \mathcal{N}(0, \tau^2 \mathbf{I})$$

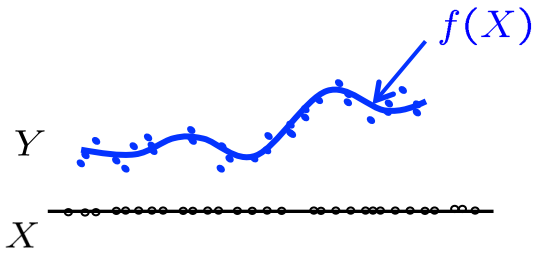
$$\Rightarrow f(X) \sim \mathcal{N}(0, \tau^2 \phi(X) \cdot \phi(X))$$

$$\sim \mathcal{N}(0, \tau^2 K_{X,X})$$



Prior for kernelized ridge regression

- For kernelized ridge regression, what's the prior?



$$Y = f(X) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

$$f(X) = \phi(X)\beta \quad \beta \sim \mathcal{N}(0, \tau^2 \mathbf{I})$$

$$\Rightarrow f(X) \sim \mathcal{N}(0, \tau^2 \phi(X) \cdot \phi(X)) \sim \mathcal{N}(0, \tau^2 K_{X,X})$$

$$\begin{bmatrix} f(X_i) \\ f(X_j) \end{bmatrix} = \begin{bmatrix} \phi(X_i) \\ \phi(X_j) \end{bmatrix} \beta$$

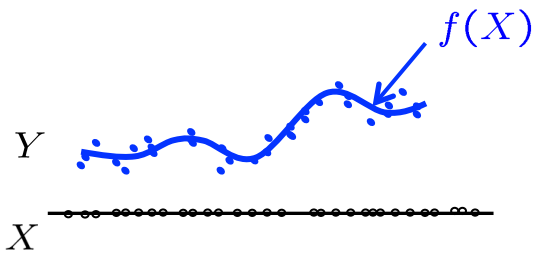
$$\begin{bmatrix} f(X_i) \\ f(X_j) \end{bmatrix} \sim \mathcal{N}\left(0, \tau^2 \begin{bmatrix} \phi(X_i) \cdot \phi(X_i) & \phi(X_i) \cdot \phi(X_j) \\ \phi(X_j) \cdot \phi(X_i) & \phi(X_j) \cdot \phi(X_j) \end{bmatrix}\right)$$

$$\sim \mathcal{N}\left(0, \tau^2 \begin{bmatrix} K(i,i) & K(i,j) \\ K(j,i) & K(j,j)) \end{bmatrix}\right)$$

Prior for kernelized ridge regression

- For kernelized ridge regression, what's the prior?

$$Y = f(X) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$



$$\begin{bmatrix} f(X_i) \\ f(X_j) \end{bmatrix} \sim \mathcal{N}\left(0, \tau^2 \begin{bmatrix} K(i,i) & K(i,j) \\ K(j,i) & K(j,j)) \end{bmatrix}\right)$$

Then $\tau^2 K(i,j) = \Sigma(i,j)$ i.e. we can interpret Kernel function as the covariance function under a Gaussian process prior.

$$f \sim \mathcal{N}(0, \mathbf{\Sigma})$$

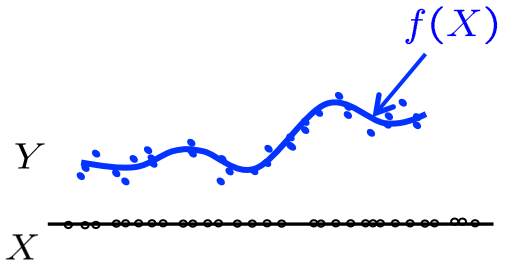
$$\text{i.e. } \mathbb{E}[f(X_i) f(X_j)] = \Sigma(i,j)$$

Gaussian process regression

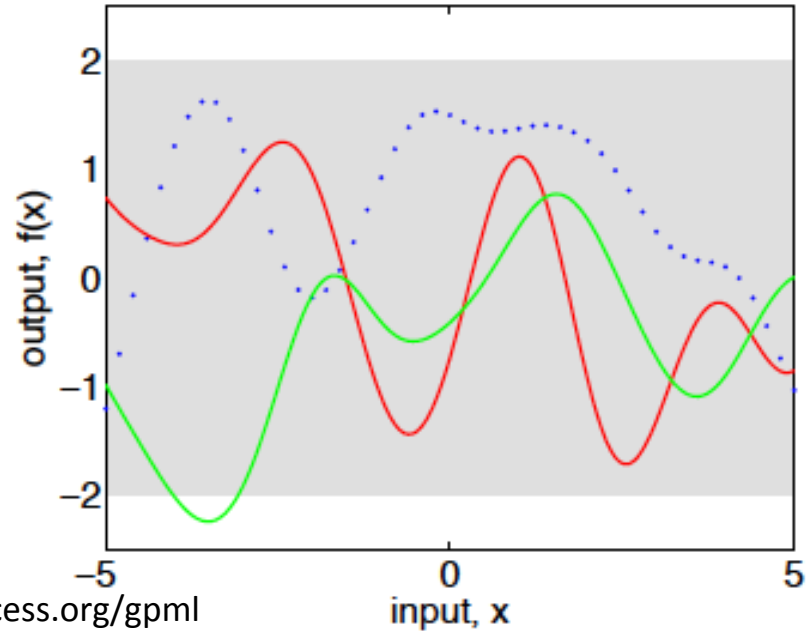
$$Y = f(X) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

$$f \sim \mathcal{N}(0, \boldsymbol{\Sigma})$$ Gaussian process prior

$$\text{i.e. } \mathbb{E}[f(X_i) f(X_j)] = \Sigma(i, j)$$



3 functions drawn at random from GP **prior**

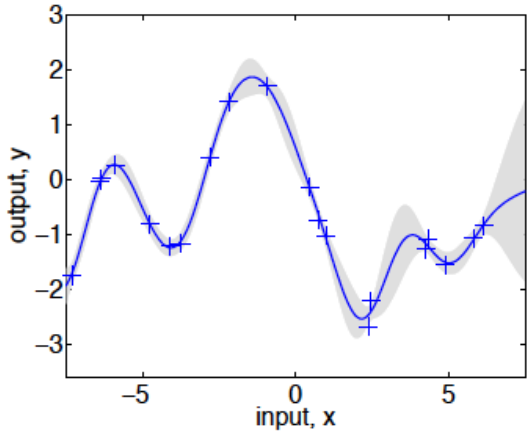


$$\Sigma(i, j) = \exp\left(-\frac{1}{2}\|X_i - X_j\|^2\right)$$

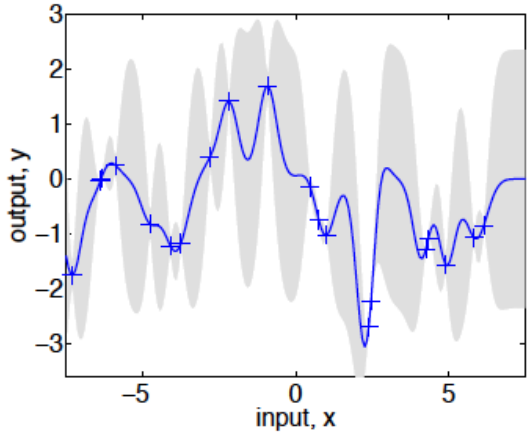
Shaded region represents pointwise mean +/- 2 standard deviation

guassianprocess.org/gpml

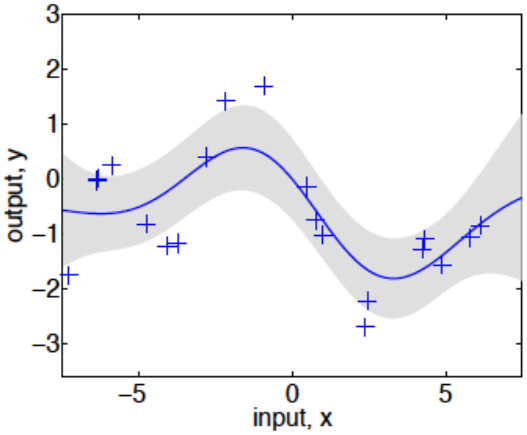
Gaussian process regression



(a), $\ell = 1$



(b), $\ell = 0.3$



(c), $\ell = 3$

Figure 2.5: (a) Data is generated from a GP with hyperparameters $(\ell, \sigma_f, \sigma_n) = (1, 1, 0.1)$, as shown by the + symbols. Using Gaussian process prediction with these hyperparameters we obtain a 95% confidence region for the underlying function f (shown in grey). Panels (b) and (c) again show the 95% confidence region, but this time for hyperparameter values $(0.3, 1.08, 0.00005)$ and $(3.0, 1.16, 0.89)$ respectively.

Controls "smoothness" of functions

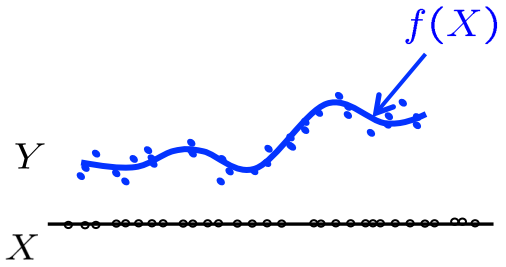
$$k_y(x_p, x_q) = \sigma_f^2 \exp\left(-\frac{1}{2\ell^2}(x_p - x_q)^2\right) + \sigma_n^2 \delta_{pq}$$

Gaussian process regression

$$Y = f(X) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

$$f \sim \mathcal{N}(0, \mathbf{\Sigma}) \qquad \text{Gaussian process prior}$$

$$\text{i.e. } \mathbb{E}[f(X_i) f(X_j)] = \Sigma(i,j)$$



Lets derive the MAP estimate under this prior:

$$\begin{bmatrix} \mathbf{Y} \\ f(X) \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} \mathbf{\Sigma}_{nn} + \sigma^2 \mathbf{I} & \mathbf{\Sigma}_{nX} \\ \mathbf{\Sigma}_{Xn} & \mathbf{\Sigma}_{XX} \end{bmatrix}\right)$$

Posterior distribution

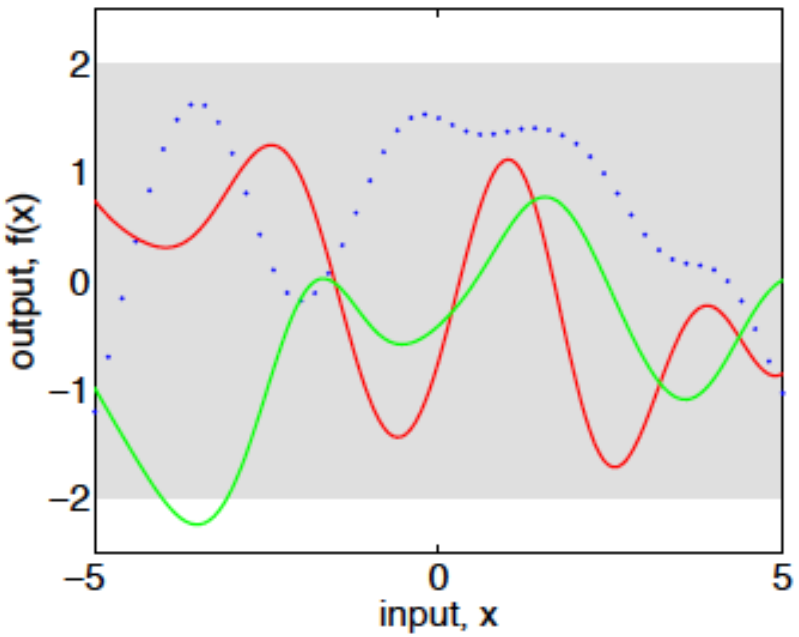
$$\Rightarrow f(X)|\mathbf{Y} \sim \mathcal{N}(\mathbf{\Sigma}_{Xn}(\mathbf{\Sigma}_{nn} + \sigma^2 \mathbf{I})^{-1}\mathbf{Y}, \mathbf{\Sigma}_{XX} - \mathbf{\Sigma}_{Xn}(\mathbf{\Sigma}_{nn} + \sigma^2 \mathbf{I})^{-1}\mathbf{\Sigma}_{nX})$$

MAP estimate $\widehat{f}(X) = \mathbf{\Sigma}_{Xn}(\mathbf{\Sigma}_{nn} + \sigma^2 \mathbf{I})^{-1}\mathbf{Y} \equiv \mathbf{K}_X(\mathbf{K} + \lambda \mathbf{I})^{-1}\mathbf{Y}$

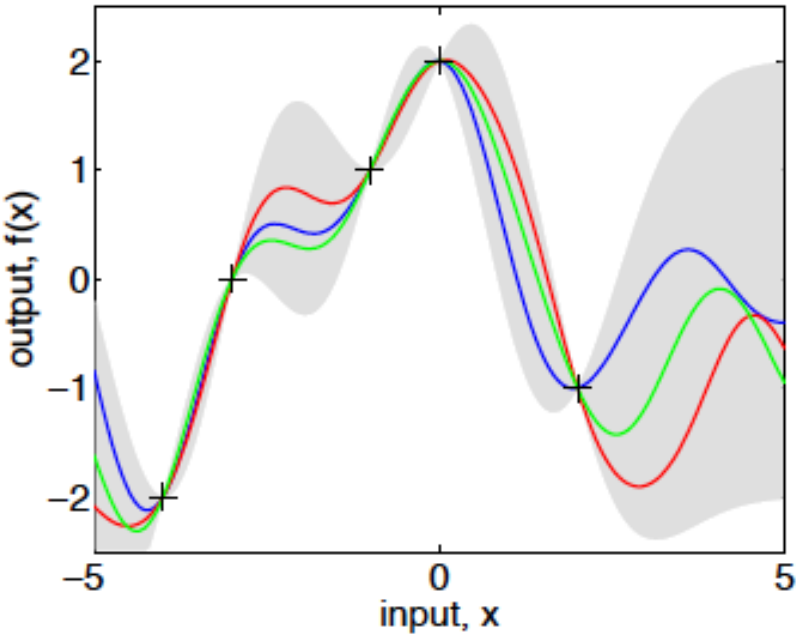
Kernel Ridge Regression!

Gaussian process regression

3 functions drawn at random from GP **prior**



3 functions drawn at random from **posterior** based on five noiseless observations



Shaded region represents pointwise mean +/- 2 standard deviation

http://www.gaussianprocess.org/gpml/

What you should know

Nonlinear regression

- Polynomial regression
- Regression with Non-linear features
- Kernelized ridge regression
- GP Regression

- Bias-variance decomposition

Ridge regression (dual)

$$\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2$$

$$\widehat{\beta} = (\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{Y}$$

$$= \mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \lambda\mathbf{I})^{-1}\mathbf{Y}$$

Similarity with SVMs

Primal problem:

$$\min_{\beta, z_i} \sum_{i=1}^{n} z_i^2 + \lambda\|\beta\|_2^2$$

$$\text{s.t. } z_i = Y_i - X_i\beta$$

SVM Primal problem:

$$\min_{w, \xi_i} C\sum_{i=1}^{n} \xi_i + \frac{1}{2}\|w\|_2^2$$

$$\text{s.t. } \xi_i = \max(1 - Y_i X_i\, w, 0)$$

Dual problem:

$$\max_{\alpha} -\frac{\alpha^\top\alpha}{2} - \frac{1}{2\lambda}\alpha^\top\mathbf{A}\mathbf{A}^\top\alpha - \alpha^\top\mathbf{Y}$$

α_i – one per training point (n-dimensional optimization problem)

Ridge regression (dual)

$$\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2$$

$$\widehat{\beta} = (\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{Y}$$

$$= \mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \lambda\mathbf{I})^{-1}\mathbf{Y}$$

Alternate derivation

Dual problem:

$$\max_{\alpha} \; -\frac{\alpha^\top\alpha}{2} - \frac{1}{2\lambda}\alpha^\top\mathbf{A}\mathbf{A}^\top\alpha - \alpha^\top\mathbf{Y} \qquad \Rightarrow \widehat{\alpha} = -\left(\frac{\mathbf{A}\mathbf{A}^\top}{\lambda} + \mathbf{I}\right)^{-1}\mathbf{Y}$$

α_i – one per training point

can get back $\widehat{\beta} = -\frac{1}{\lambda}\sum_{i=1}^{n}\alpha_i X_i^\top = -\frac{1}{\lambda}\mathbf{A}^\top\alpha = \mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top + \lambda\mathbf{I})^{-1}\mathbf{Y}$

Weight of each training point

Ridge regression (dual)

$$\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2$$

$$\widehat{\beta} = (\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{Y}$$

$$= \mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \lambda\mathbf{I})^{-1}\mathbf{Y}$$

Dual interpretation

$$\widehat{\beta} = -\frac{1}{\lambda}\sum_{i=1}^{n} \alpha_i X_i^\top = -\frac{1}{\lambda}\mathbf{A}^\top\alpha = \mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top + \lambda\mathbf{I})^{-1}\mathbf{Y}$$

Weight of each training point

Prediction $\widehat{f}_n(X) = \mathbf{X}\widehat{\beta} = -\frac{1}{\lambda}\sum_{i=1}^{n} \alpha_i \mathbf{X}\cdot X_i$

which can be kernelized

$$\widehat{f}_n(X) = \mathbf{K}_X(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{Y} = -\frac{1}{\lambda}\sum_{i=1}^{n} \alpha_i K(\mathbf{X}, X_i)$$

Prediction is a weighted sum of "bumps" placed at each training data.

Online ques

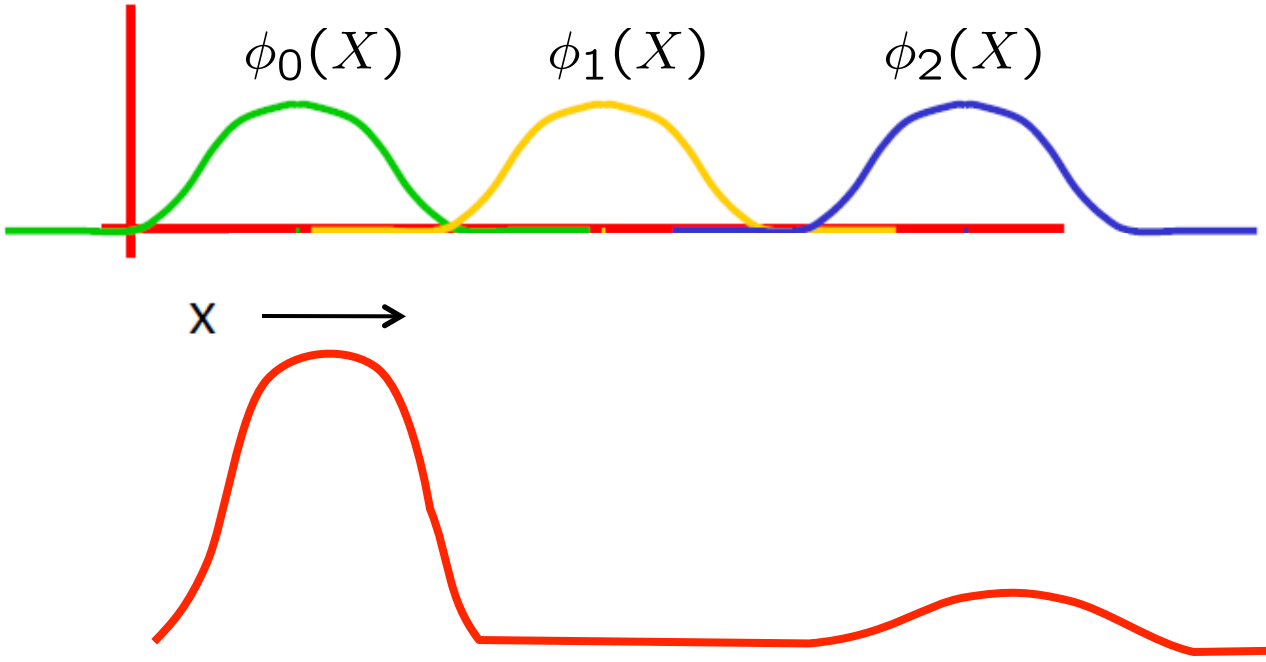
- If you train a linear regression estimator with only half the data, its variance is larger.
- If you train a linear regression estimator with only half the data, its bias is smaller.
- As you increase the bandwidth of a Gaussian kernel, the bias of the corresponding kernelized ridge regression decreases.
- A least square regression estimator essentially forms a weighted sum of the training labels.
- Ridge regression estimator essentially forms a weighted sum of the training labels.
- As the regularization parameter λ is increased, the estimated regression coefficients become sparser.
- If we have a dataset of brain scans of 20 patients, and we are trying to predict age of a patient using their recorded brain activity in 600 regions of interest in the brain as features, then we should prefer to use ridge regression over least squares linear regression.
- Polynomial regression estimator involves a nonlinear transformation of the training labels.
- Polynomial regression estimator typically involves a nonlinear transformation of the training features.
- As the number of test points increases, the average mean square error on test points goes down.
- As the number of training points increases, the average mean square error on test points goes down.
- As the number of test points increases, the average mean square error on training points goes down.
- If we set the covariance function of a Gaussian process prior to be diagonal i.e. 1 if $X_i = X_j$ and 0 otherwise, then we can't hope to learn the regression function.
- A polynomial of degree up to 2 has a lower bias than a polynomial of degree up to 3 for fitting labels that are generated according to a quadratic signal function plus noise.
- Suppose we are solving ridge regression using gradient descent. Then increasing the regularization parameter \(\lambda\) in ridge regression causes a larger change in the regression coefficients.

- Gradient descent update for kernel ridge regression?
- The average Constant fit?

Local Regression

$$f(X) = \sum_{j=0}^{m} \beta_j \phi_j(X)$$

Basis coefficients ← Nonlinear features/basis functions

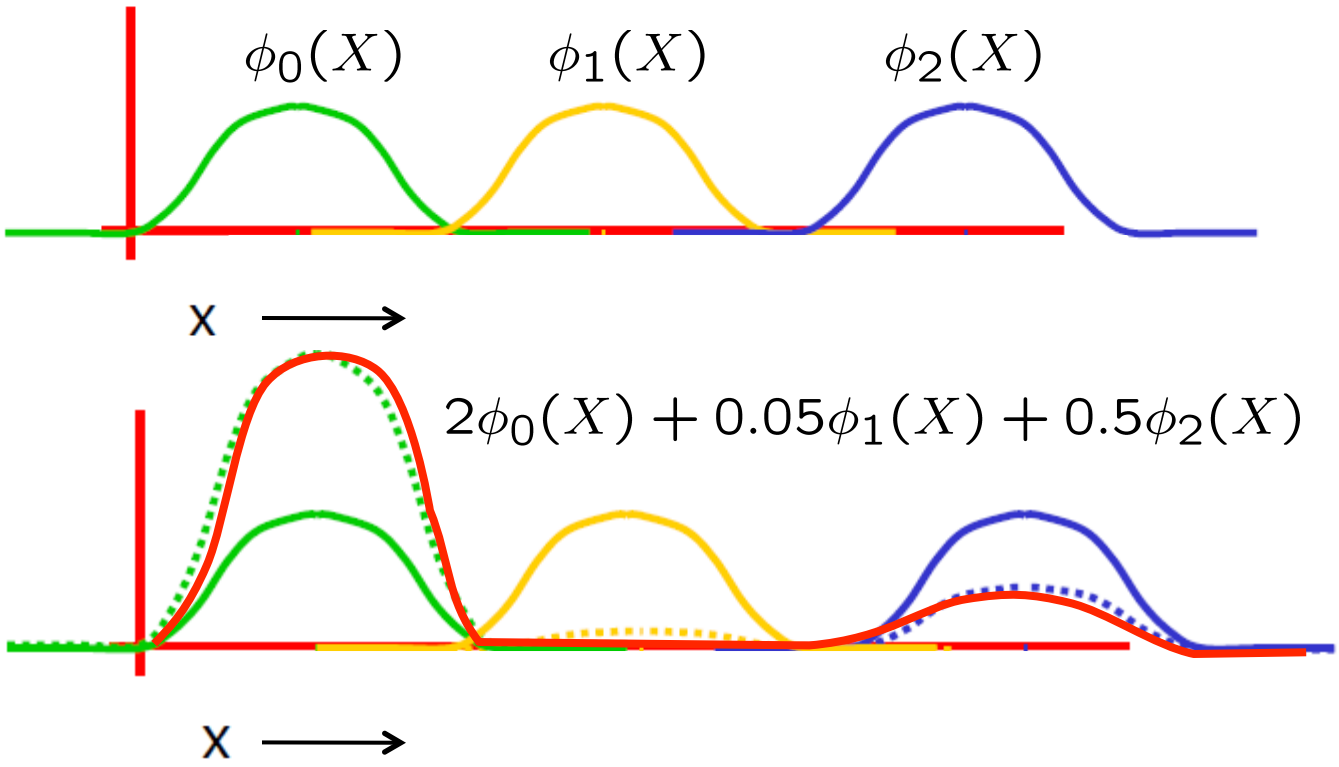


Globally supported basis functions (polynomial, fourier) will not yield a good representation

Local Regression

$$f(X) = \sum_{j=0}^{m} \beta_j \phi_j(X)$$

Basis coefficients ← Nonlinear features/basis functions



Globally supported basis functions (polynomial, fourier) will not yield a good representation

What you should know

Linear Regression

- Least Squares Estimator
- Normal Equations
- Gradient Descent
- Geometric and Probabilistic Interpretation (connection to MLE)

Regularized Linear Regression (connection to MAP)

- Ridge Regression, Lasso

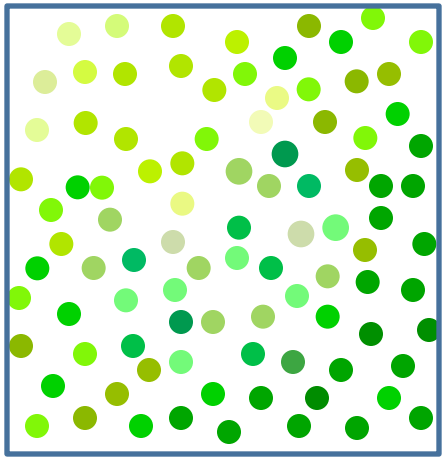
Polynomial Regression, Basis (Fourier, Wavelet) Estimators

Next time

- Kernel Regression (Localized)
- Regression Trees

Temperature sensing

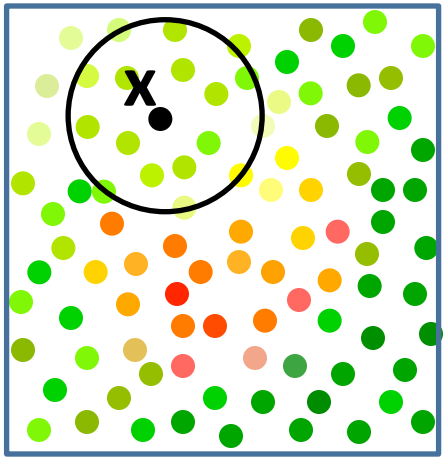
- What is the temperature in the room?



$$\widehat{T} = \frac{1}{n} \sum_{i=1}^{n} Y_i$$

Average

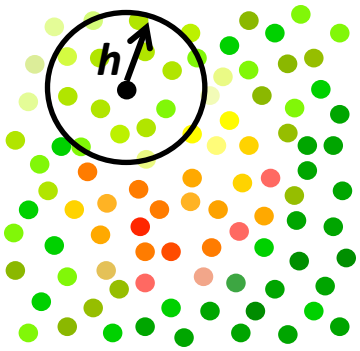
at location x?



$$\widehat{T}(x) = \frac{\sum_{i=1}^{n} Y_i 1_{||X_i - x|| \leq h}}{\sum_{i=1}^{n} 1_{||X_i - x|| \leq h}}$$

"Local" Average

Kernel Regression

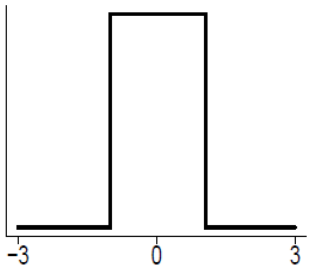


- Aka Local Regression
- Nadaraya-Watson Kernel Estimator

$$\widehat{f}_n(X) = \sum_{i=1}^{n} w_i Y_i \quad \text{Where} \quad w_i(X) = \frac{K\left(\frac{X - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X - X_i}{h}\right)}$$

- Weight each training point based on distance to test point
- Boxcar kernel yields local average

$$\text{boxcar kernel}: \quad K(x) = \frac{1}{2} I(x),$$

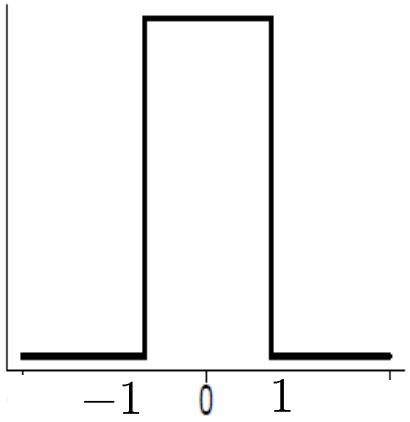


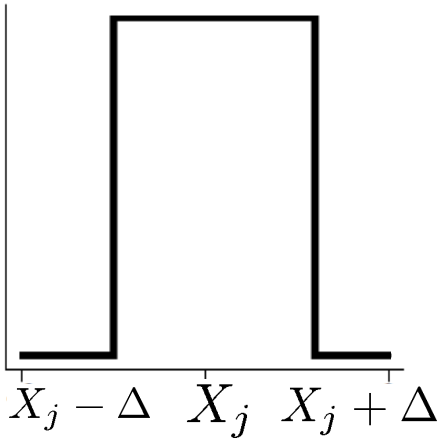
Kernels

$$K\left(\frac{X_j - x}{\Delta}\right)$$

boxcar kernel :

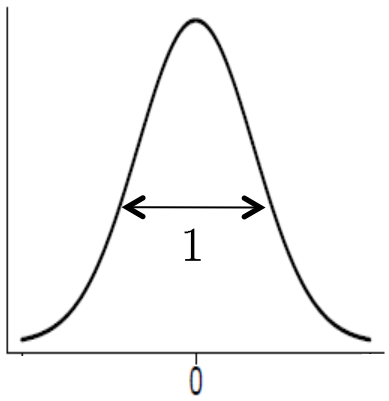
$$K(x) = \frac{1}{2} I(x),$$

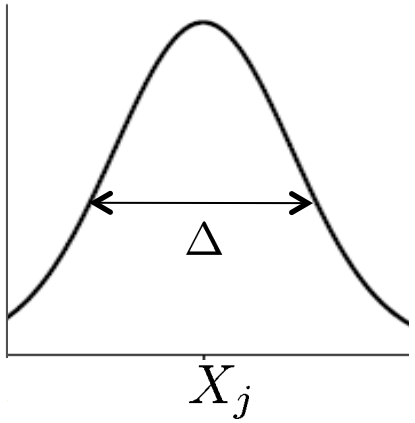




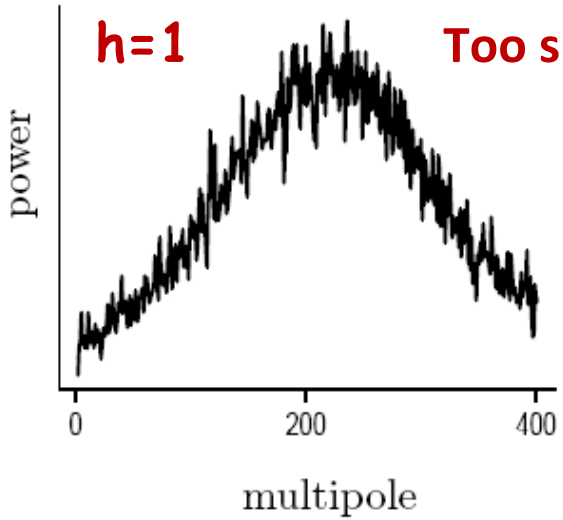
Gaussian kernel :

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$





Choice of kernel bandwidth h



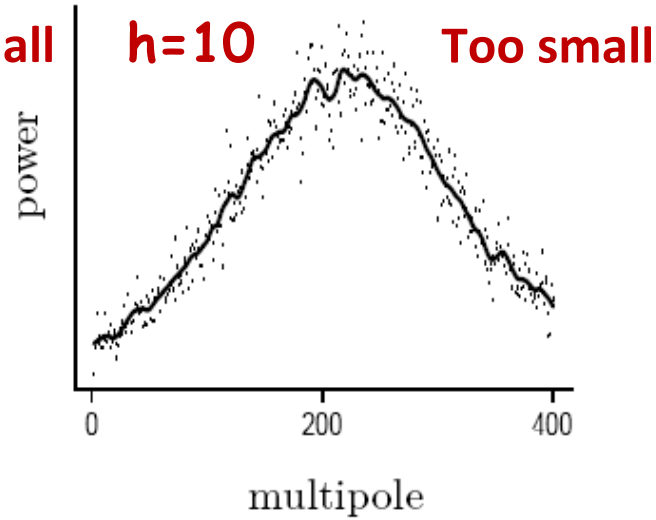
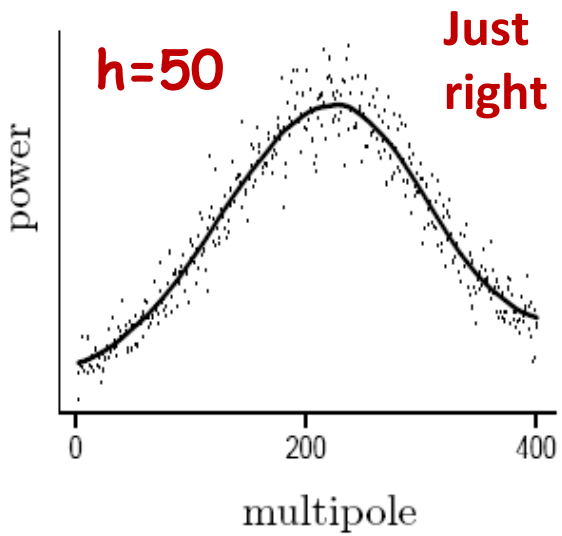
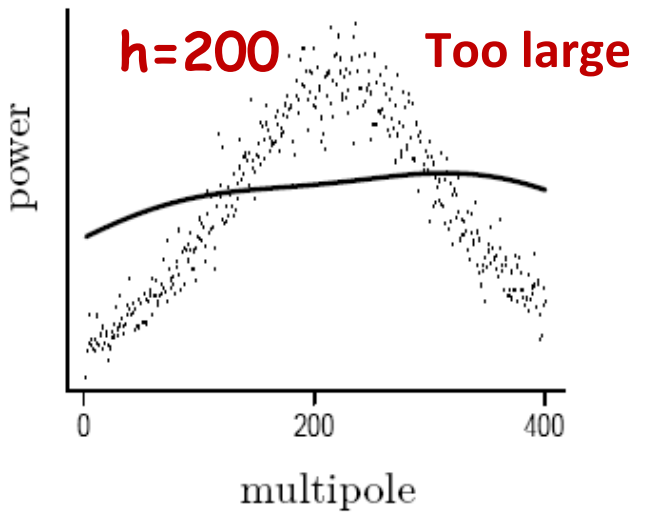


Image Source: Larry's book – All of Nonparametric Statistics





Choice of kernel is not that important

Kernel Regression as Weighted Least Squares

$$\min_f \sum_{i=1}^{n} w_i (f(X_i) - Y_i)^2 \qquad w_i(X) = \frac{K\left(\frac{X - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X - X_i}{h}\right)}$$

Weighted Least Squares

Kernel regression corresponds to locally constant estimator obtained from (locally) weighted least squares

i.e. set $f(X_i) = \beta$ (a constant)

Kernel Regression as Weighted Least Squares

set $f(X_i) = \beta$ (a constant)

$$\min_{\beta} \sum_{i=1}^{n} w_i(\beta - Y_i)^2$$

β: constant

$$w_i(X) = \frac{K\left(\frac{X - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X - X_i}{h}\right)}$$

$$\frac{\partial J(\beta)}{\partial \beta} = 2\sum_{i=1}^{n} w_i(\beta - Y_i) = 0$$

Notice that $\sum_{i=1}^{n} w_i = 1$

$$\Rightarrow \hat{f}_n(X) = \hat{\beta} = \sum_{i=1}^{n} w_i Y_i$$

Local Linear/Polynomial Regression

$$\min_f \sum_{i=1}^{n} w_i(f(X_i) - Y_i)^2 \qquad w_i(X) = \frac{K\left(\frac{X-X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X-X_i}{h}\right)}$$

Weighted Least Squares

Local Polynomial regression corresponds to locally polynomial estimator obtained from (locally) weighted least squares

$$f(X_i) = \beta_0 + \beta_1(X_i - X) + \frac{\beta_2}{2!}(X_i - X)^2 + \cdots + \frac{\beta_p}{p!}(X_i - X)^p$$

i.e. set

(local polynomial of degree p around X)

More in HW, 10-702 (statistical machine learning)

Kernel Regression (Local)

$$\min_f \frac{1}{n}\sum_{i=1}^{n} w_i(f(X_i) - Y_i)^2 \qquad \frac{1}{n}\sum_{i=1}^{n} w_i = 1$$

Weighted Least Squares

Weigh each training point based on distance to test point

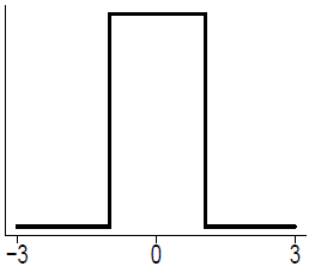
$$w_i(X) = \frac{K\left(\frac{X - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X - X_i}{h}\right)}$$

K – Kernel

h – Bandwidth of kernel

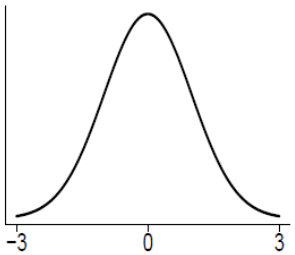
boxcar kernel :

$$K(x) = \frac{1}{2} I(x),$$



Gaussian kernel :

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$



Nadaraya-Watson Kernel Regression

$$\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} w_i (\beta - Y_i)^2$$

constant

$$w_i(X) = \frac{K\left(\frac{X - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X - X_i}{h}\right)}$$

$$\frac{\partial J(\beta)}{\partial \beta} = 2 \sum_{i=1}^{n} w_i (\beta - Y_i) = 0$$

$$\Rightarrow \hat{f}_n(X) = \hat{\beta} = \sum_{i=1}^{n} w_i Y_i$$

Nadaraya-Watson Kernel Regression

$$\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} w_i (\beta - Y_i)^2$$

constant

$$w_i(X) = \frac{K\left(\frac{X - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X - X_i}{h}\right)}$$

boxcar kernel :

$$K\left(\frac{X - X_i}{h}\right) = 1_{|X - X_i| \le h}$$

$$\frac{\partial J(\beta)}{\partial \beta} = 2 \sum_{i=1}^{n} w_i (\beta - Y_i) = 0$$

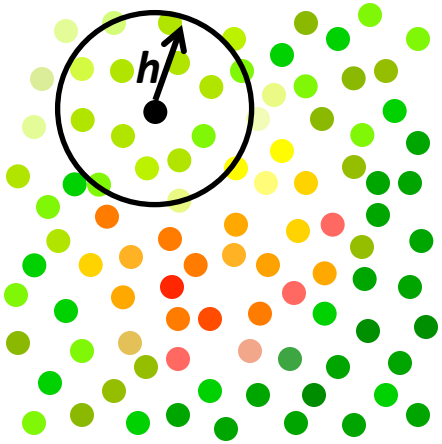
$$\Rightarrow \hat{f}_n(X) = \hat{\beta} = \sum_{i=1}^{n} w_i Y_i$$

with box-car kernel

$$= \frac{1}{n_X^h} \sum_{i=1}^{n} Y_i \, 1_{|X - X_i| \le h}$$

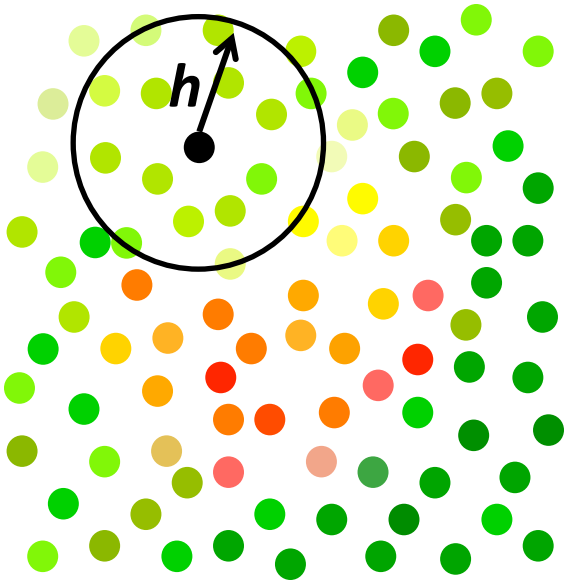
#pts in h ball around X

Sum of Ys in h ball around X



Recall: NN classifier
Average <-> majority vote

Choice of Bandwidth



Should depend on n, # training data
(determines variance)

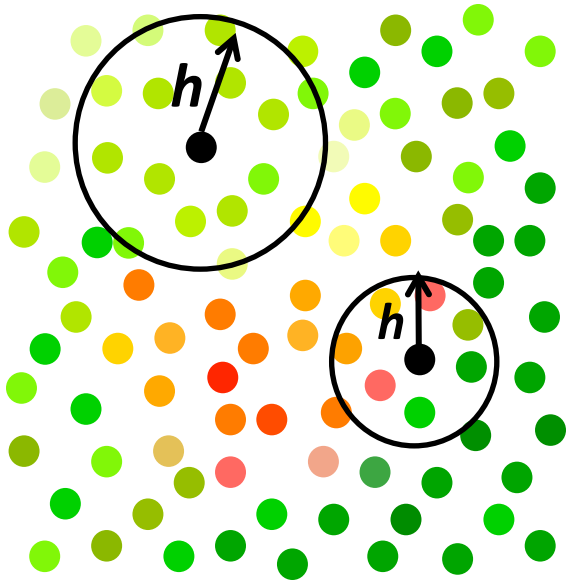
Should depend on smoothness of function
(determines bias)

Large Bandwidth – average more data points, reduce noise (Lower variance)

Small Bandwidth – less smoothing, more accurate fit (Lower bias)

Bias - Variance tradeoff : More to come in later lectures

Spatially adaptive regression



If function smoothness varies spatially, we want to allow bandwidth h to depend on X

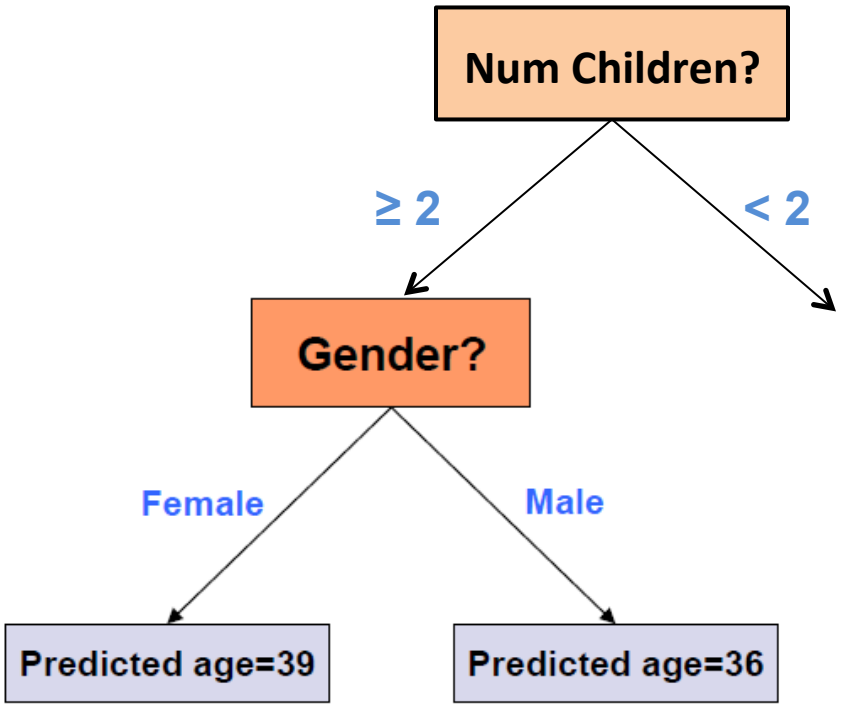
Local polynomials, splines, wavelets, regression trees …

Regression trees

$X^{(1)}$ .... $X^{(p)}$ Y

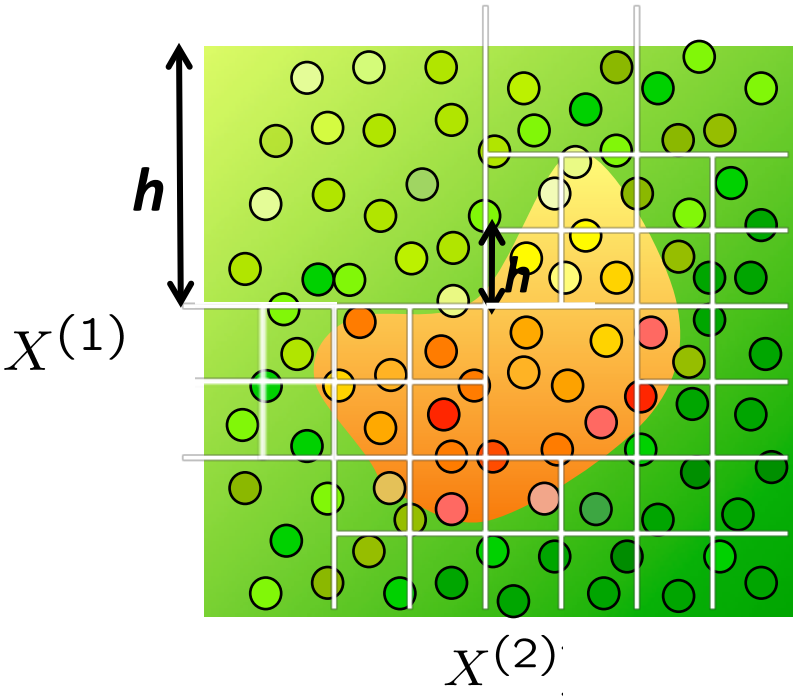
Gender	Rich?	Num. Children	# travel per yr.	Age
F	No	2	5	38
M	No	0	2	25
M	Yes	1	0	72
:	:	:	:	:

Binary Decision Tree

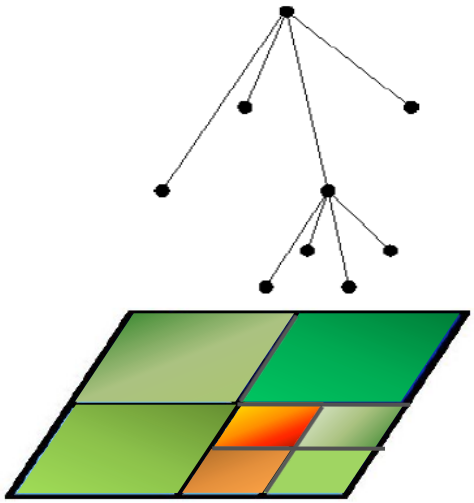


Average (fit a constant) on the leaves

Regression trees



Quad Decision Tree



f - Polynomial fit on each leaf

$$\widehat{f}_n^T = \arg\min_{f \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2$$

If $\sum_{\text{small cells}, L} \sum_{i \in L} (\hat{f}(X_i) - Y_i)^2 < \sum_{i \in \text{merged cell}} (\hat{f}(X_i) - Y_i)^2$, then split

Else stop

Compare residual error with and without split

Summary

Discriminative vs Generative Classifiers
- Naïve Bayes vs Logistic Regression

Regression
- Linear Regression
  - Least Squares Estimator
  - Normal Equations
  - Gradient Descent
  - Geometric Interpretation
  - Probabilistic Interpretation (connection to MLE)
- Regularized Linear Regression (connection to MAP)
  - Ridge Regression, Lasso
- Polynomial Regression, Basis (Fourier, Wavelet) Estimators
- Kernel Regression (Localized)
- Regression Trees