Machine Learning

4771

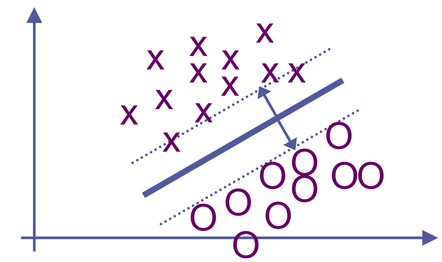
Instructor: Tony Jebara

Topic 2

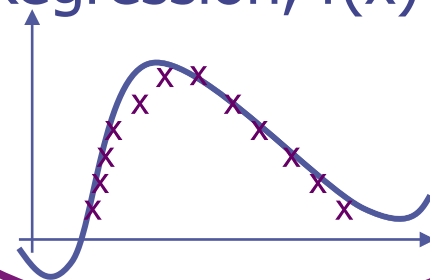
- Regression
- Empirical Risk Minimization
- Least Squares
- Higher Order Polynomials
- Under-fitting / Over-fitting
- Cross-Validation

Regression

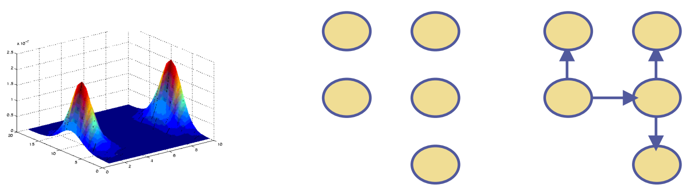
Classification



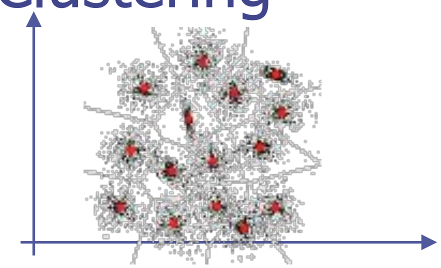
Regression, f(x)=y



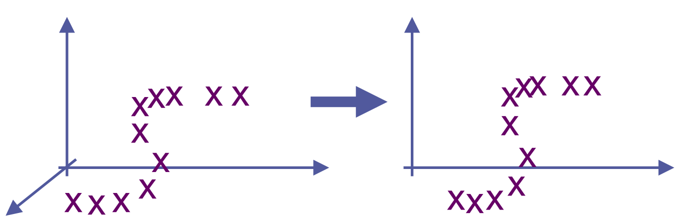
Density/Structure Estimation



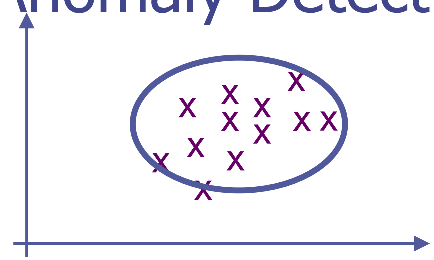
Clustering



Feature Selection



Anomaly Detection



Supervised

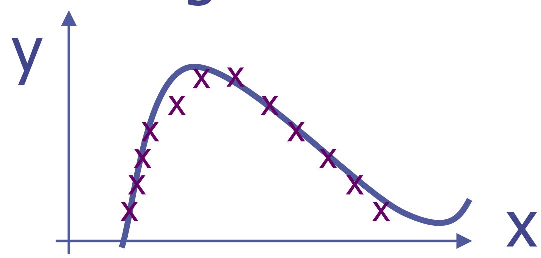
Unsupervised

Function Approximation

- Start with training dataset

$$\mathcal{X} = \left\{ (x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N) \right\} \quad x \in \mathbb{R}^D = \begin{bmatrix} x(1) \\ x(2) \\ \ldots \\ x(D) \end{bmatrix} \quad y \in \mathbb{R}^1$$

- Have N (input, output) pairs
- Find a function f(x) to predict y from x
  That fits the training data well



- Example: predict the price of house in
  dollars y using x = [#rooms; latitude; longitude; …]
- Need: a) Way to evaluate how good a fit we have
  b) Class of functions in which to search for f(x)

Empirical Risk Minimization

- Idea: minimize 'loss' on the training data set
- Empirical = use the training set to find the best fit
- Define a loss function of how good we fit a single point:

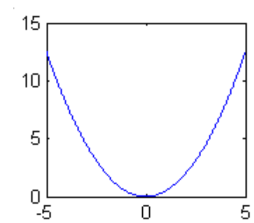
$$L\left(y, f\left(x\right)\right)$$

- Empirical Risk = the average loss over the dataset

$$R = \frac{1}{N}\sum_{i=1}^{N} L\left(y_i, f\left(x_i\right)\right)$$

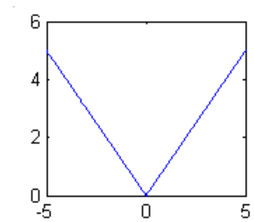
- Simplest loss: squared error from y value

$$L\left(y_i, f\left(x_i\right)\right) = \frac{1}{2}\left(y_i - f\left(x_i\right)\right)^2$$



- Other possible loss: absolute error

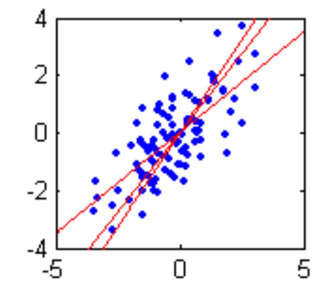
$$L\left(y_i, f\left(x_i\right)\right) = \left|y_i - f\left(x_i\right)\right|$$



Linear Function Classes

- Linear is simplest class of functions to search over:

$$f\left(x;\theta\right) = \theta^T x + \theta_0 = \sum_{d=1}^{D} \theta_d x\left(d\right) + \theta_0$$

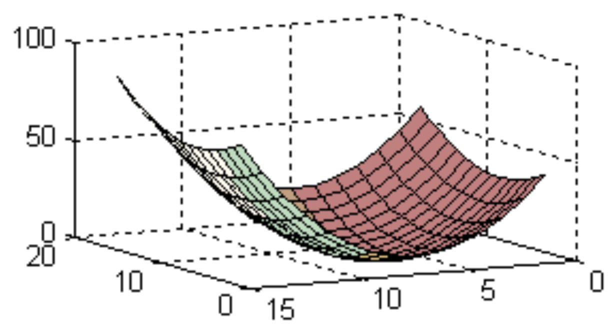


- Start with x being 1-dimensional (D=1):

$$f\left(x;\theta\right) = \theta_1 x + \theta_0$$

- Plug in the above & minimize empirical risk over θ

$$R\left(\theta\right) = \frac{1}{2N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)^2$$



- Note: minimum occurs when R(θ) gets flat (not always!)
- Note: when R(θ) is flat, gradient $\nabla_\theta R = 0$

Min by Gradient=0

- Gradient=0 means the partial derivatives are all 0

$$\nabla_\theta R = \begin{bmatrix} \frac{\partial R}{\partial \theta_0} \\ \frac{\partial R}{\partial \theta_1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- Take partials of empirical risk:

$$R(\theta) = \frac{1}{2N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)^2$$

Min by Gradient=0

- Gradient=0 means the partial derivatives are all 0

$$\nabla_\theta R = \begin{bmatrix} \frac{\partial R}{\partial \theta_0} \\ \frac{\partial R}{\partial \theta_1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- Take partials of empirical risk:

$$R(\theta) = \tfrac{1}{2N} \sum_{i=1}^{N} \left(y_i - \theta_1 x_i - \theta_0\right)^2$$

$$\frac{\partial R}{\partial \theta_0} = \tfrac{1}{N} \sum_{i=1}^{N} \left(y_i - \theta_1 x_i - \theta_0\right)\left(-1\right) = 0$$

Min by Gradient=0

- Gradient=0 means the partial derivatives are all 0

$$\nabla_\theta R = \begin{bmatrix} \frac{\partial R}{\partial \theta_0} \\ \frac{\partial R}{\partial \theta_1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- Take partials of empirical risk:

$$R(\theta) = \frac{1}{2N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)^2$$

$$\frac{\partial R}{\partial \theta_0} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)\left(-1\right) = 0$$

$$\frac{\partial R}{\partial \theta_1} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)\left(-x_i\right) = 0$$

Min by Gradient=0

- Gradient=0 means the partial derivatives are all 0

$$\nabla_\theta R = \begin{bmatrix} \frac{\partial R}{\partial \theta_0} \\ \frac{\partial R}{\partial \theta_1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- Take partials of empirical risk:

$$R(\theta) = \tfrac{1}{2N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)^2$$

$$\frac{\partial R}{\partial \theta_0} = \tfrac{1}{N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)\left(-1\right) = 0$$

$$\frac{\partial R}{\partial \theta_1} = \tfrac{1}{N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)\left(-x_i\right) = 0$$

$$\theta_0 = \tfrac{1}{N}\sum y_i - \theta_1 \tfrac{1}{N}\sum x_i$$

Min by Gradient=0

- Gradient=0 means the partial derivatives are all 0

$$\nabla_\theta R = \begin{bmatrix} \frac{\partial R}{\partial \theta_0} \\ \frac{\partial R}{\partial \theta_1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- Take partials of empirical risk:

$$R(\theta) = \frac{1}{2N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)^2$$

$$\frac{\partial R}{\partial \theta_0} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)(-1) = 0$$

$$\frac{\partial R}{\partial \theta_1} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)(-x_i) = 0$$

$$\theta_0 = \frac{1}{N}\sum y_i - \theta_1 \frac{1}{N}\sum x_i$$

$$\theta_1 \sum x_i^2 = \sum y_i x_i - \theta_0 \sum x_i$$

Min by Gradient=0

- Gradient=0 means the partial derivatives are all 0

$$\nabla_\theta R = \begin{bmatrix} \frac{\partial R}{\partial \theta_0} \\ \frac{\partial R}{\partial \theta_1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- Take partials of empirical risk:

$$R(\theta) = \tfrac{1}{2N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)^2$$

$$\frac{\partial R}{\partial \theta_0} = \tfrac{1}{N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)\left(-1\right) = 0$$

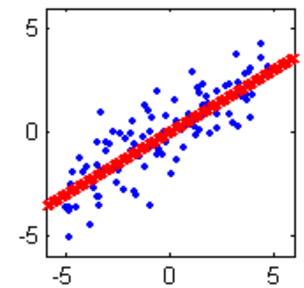
$$\frac{\partial R}{\partial \theta_1} = \tfrac{1}{N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)\left(-x_i\right) = 0$$

$$\theta_0 = \tfrac{1}{N}\sum y_i - \theta_1 \tfrac{1}{N}\sum x_i$$

$$\theta_1 \sum x_i^2 = \sum y_i x_i - \theta_0 \sum x_i$$

$$\theta_1 = \frac{\sum y_i x_i - \frac{1}{N}\sum y_i \sum x_i}{\sum x_i^2 - \frac{1}{N}\sum x_i \sum x_i}$$

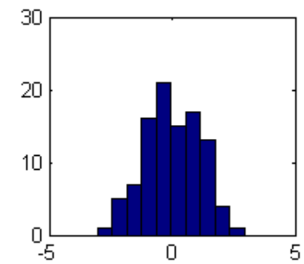
Properties of the Solution



- Setting θ^* as before gives least squared error
- Define error on each data point as:

$$e_i = y_i - \theta_1^* x_i - \theta_0^*$$

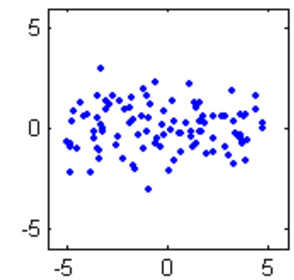
- Note property #1:



$$\frac{\partial R}{\partial \theta_0} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right) = 0$$

...average error is zero $\frac{1}{N}\sum e_i = 0$

- Note property #2:



$$\frac{\partial R}{\partial \theta_1} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)x_i = 0$$

...error not correlated with data

$$\frac{1}{N}\sum e_i x_i = \frac{1}{N} e^T x = 0$$

Multi-Dimensional Regression

- More elegant/general to do $\nabla_\theta R = 0$ with linear algebra
- Rewrite empirical risk in vector-matrix notation:

$$
\begin{aligned}
R(\theta) &= \frac{1}{2N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)^2 \\
&= \frac{1}{2N}\sum_{i=1}^{N}\left(y_i - \begin{bmatrix} 1 & x_i \end{bmatrix}\begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}\right)^2 \\
&= \frac{1}{2N}\left\| \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} - \begin{bmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}\begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} \right\|^2 \\
&= \frac{1}{2N}\left\| \mathbf{y} - \mathbf{X}\theta \right\|^2
\end{aligned}
$$

Multi-Dimensional Regression

- More elegant/general to do $\nabla_\theta R = 0$ with linear algebra
- Rewrite empirical risk in vector-matrix notation:

$$R(\theta) = \frac{1}{2N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)^2$$

$$= \frac{1}{2N}\sum_{i=1}^{N}\left(y_i - \begin{bmatrix} 1 & x_i \end{bmatrix}\begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}\right)^2$$

$$= \frac{1}{2N}\left\| \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} - \begin{bmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}\begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}\right\|^2$$

$$= \frac{1}{2N}\left\|\mathbf{y} - \mathbf{X}\theta\right\|^2$$

Can add more dimensions by adding columns to X matrix and rows to θ vector

Multi-Dimensional Regression

- More elegant/general to do $\nabla_\theta R = 0$ with linear algebra
- Rewrite empirical risk in vector-matrix notation:

$$R(\theta) = \frac{1}{2N}\sum_{i=1}^{N}\left(y_i - \theta_1 x_i - \theta_0\right)^2$$

$$= \frac{1}{2N}\sum_{i=1}^{N}\left(y_i - \begin{bmatrix} 1 & x_i \end{bmatrix}\begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}\right)^2$$

$$= \frac{1}{2N}\left\|\begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} - \begin{bmatrix} 1 & x_1(1) & \dots & x_1(D) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_N(1) & \dots & x_N(D) \end{bmatrix}\begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_D \end{bmatrix}\right\|^2$$

$$= \frac{1}{2N}\left\|\mathbf{y} - \mathbf{X}\theta\right\|^2$$

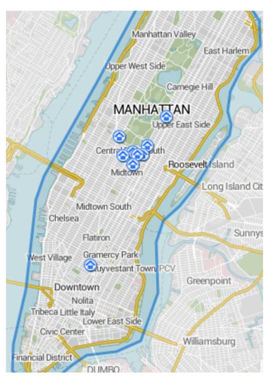
Can add more dimensions by adding columns to X matrix and rows to θ vector

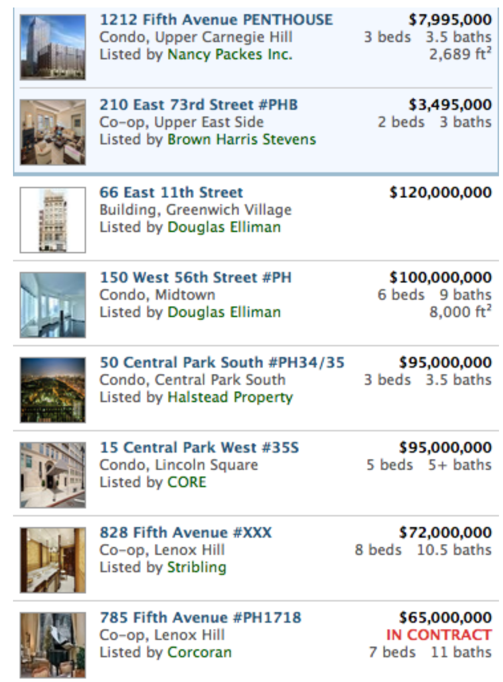
Tony Jebara, Columbia University

Multi-Dimensional Regression

- More realistic dataset: many measurements
- Have N apartments each with D measurements
- Each row of X is [#rooms; latitude; longitude,…]

$$\mathbf{X} = \begin{bmatrix} 1 & x_1(1) & \ldots & x_1(D) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_N(1) & \ldots & x_N(D) \end{bmatrix}$$





Multi-Dimensional Regression

- Solving gradient=0 $\nabla_\theta R = 0$

$$\nabla_\theta \left(\frac{1}{2N} \left\| \mathbf{y} - \mathbf{X}\theta \right\|^2 \right) = 0$$

Multi-Dimensional Regression

- Solving gradient=0

$$\nabla_\theta R = 0$$

$$\nabla_\theta \left(\frac{1}{2N} \left\| \mathbf{y} - \mathbf{X}\theta \right\|^2 \right) = 0$$

$$\frac{1}{2N} \nabla_\theta \left(\left(\mathbf{y} - \mathbf{X}\theta \right)^T \left(\mathbf{y} - \mathbf{X}\theta \right) \right) = 0$$

Multi-Dimensional Regression

- Solving gradient=0

$$\nabla_\theta R = 0$$

$$\nabla_\theta \left(\frac{1}{2N} \left\| \mathbf{y} - \mathbf{X}\theta \right\|^2 \right) = 0$$

$$\frac{1}{2N} \nabla_\theta \left[\left(\mathbf{y} - \mathbf{X}\theta \right)^T \left(\mathbf{y} - \mathbf{X}\theta \right) \right] = 0$$

$$\frac{1}{2N} \nabla_\theta \left(\mathbf{y}^T \mathbf{y} - 2\mathbf{y}^T \mathbf{X}\theta + \theta^T \mathbf{X}^T \mathbf{X}\theta \right) = 0$$

Multi-Dimensional Regression

- Solving gradient=0

$$\nabla_\theta R = 0$$

$$\nabla_\theta \left(\frac{1}{2N} \left\| \mathbf{y} - \mathbf{X}\theta \right\|^2 \right) = 0$$

$$\frac{1}{2N} \nabla_\theta \left[\left(\mathbf{y} - \mathbf{X}\theta \right)^T \left(\mathbf{y} - \mathbf{X}\theta \right) \right] = 0$$

$$\frac{1}{2N} \nabla_\theta \left(\mathbf{y}^T \mathbf{y} - 2\mathbf{y}^T \mathbf{X}\theta + \theta^T \mathbf{X}^T \mathbf{X}\theta \right) = 0$$

$$\frac{1}{2N} \left(-2\mathbf{X}^T \mathbf{y} + 2\mathbf{X}^T \mathbf{X}\theta \right) = 0$$

Multi-Dimensional Regression

- Solving gradient=0

$$\nabla_\theta R = 0$$

$$\nabla_\theta \left(\frac{1}{2N} \left\| \mathbf{y} - \mathbf{X}\theta \right\|^2 \right) = 0$$

$$\frac{1}{2N} \nabla_\theta \left(\left(\mathbf{y} - \mathbf{X}\theta \right)^T \left(\mathbf{y} - \mathbf{X}\theta \right) \right) = 0$$

$$\frac{1}{2N} \nabla_\theta \left(\mathbf{y}^T \mathbf{y} - 2\mathbf{y}^T \mathbf{X}\theta + \theta^T \mathbf{X}^T \mathbf{X}\theta \right) = 0$$

$$\frac{1}{2N} \left(-2\mathbf{X}^T \mathbf{y} + 2\mathbf{X}^T \mathbf{X}\theta \right) = 0$$

$$\mathbf{X}^T \mathbf{X}\theta = \mathbf{X}^T \mathbf{y}$$

Multi-Dimensional Regression

- Solving gradient=0

$$\nabla_\theta R = 0$$

$$\nabla_\theta \left(\frac{1}{2N} \left\| \mathbf{y} - \mathbf{X}\theta \right\|^2 \right) = 0$$

$$\frac{1}{2N} \nabla_\theta \left(\left(\mathbf{y} - \mathbf{X}\theta \right)^T \left(\mathbf{y} - \mathbf{X}\theta \right) \right) = 0$$

$$\frac{1}{2N} \nabla_\theta \left(\mathbf{y}^T \mathbf{y} - 2\mathbf{y}^T \mathbf{X}\theta + \theta^T \mathbf{X}^T \mathbf{X}\theta \right) = 0$$

$$\frac{1}{2N} \left(-2\mathbf{X}^T \mathbf{y} + 2\mathbf{X}^T \mathbf{X}\theta \right) = 0$$

$$\mathbf{X}^T \mathbf{X}\theta = \mathbf{X}^T \mathbf{y}$$

$$\theta^* = \left(\mathbf{X}^T \mathbf{X} \right)^{-1} \mathbf{X}^T \mathbf{y}$$

- In Matlab: “t=pinv(X)*y” or “t=X\y” or “t=inv(X’*X)*X’*y”

Multi-Dimensional Regression

- Solving gradient=0

$$\mathbf{X}^T \mathbf{X} \theta = \mathbf{X}^T \mathbf{y}$$

$$\theta^* = \left(\mathbf{X}^T \mathbf{X} \right)^{-1} \mathbf{X}^T \mathbf{y}$$

- In Matlab: "t=pinv(X)*y" or "t=X\y" or "t=inv(X'*X)*X'*y"
- If the matrix X is skinny, the solution is probably unique
- If X is fat (more dimensions than points) we get multiple solutions for theta which give zero error.

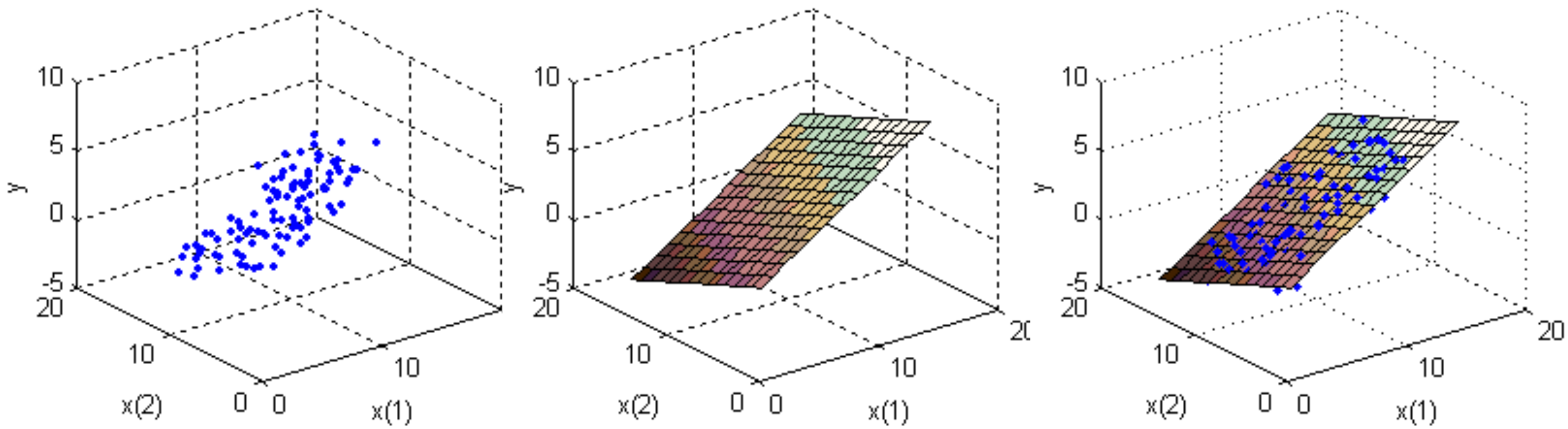
- The pseudoinverse (pinv(X)) returns the theta with zero error and which has the smallest norm.

$$\min_\theta \left\| \theta \right\|^2 \ \mathit{such\ that}\ \mathbf{X}\theta = \mathbf{y}$$

2D Linear Regression

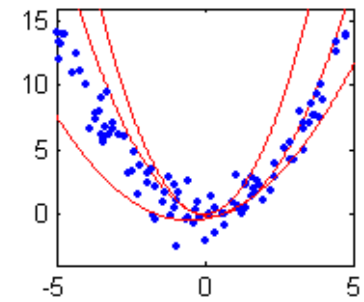
•Once best θ* is found, we can plug it into the function:

$$f\left(x;\theta^*\right) = \theta_2^* x(2) + \theta_1^* x(1) + \theta_0^*$$



•What would a fat X look like?

Polynomial Function Classes



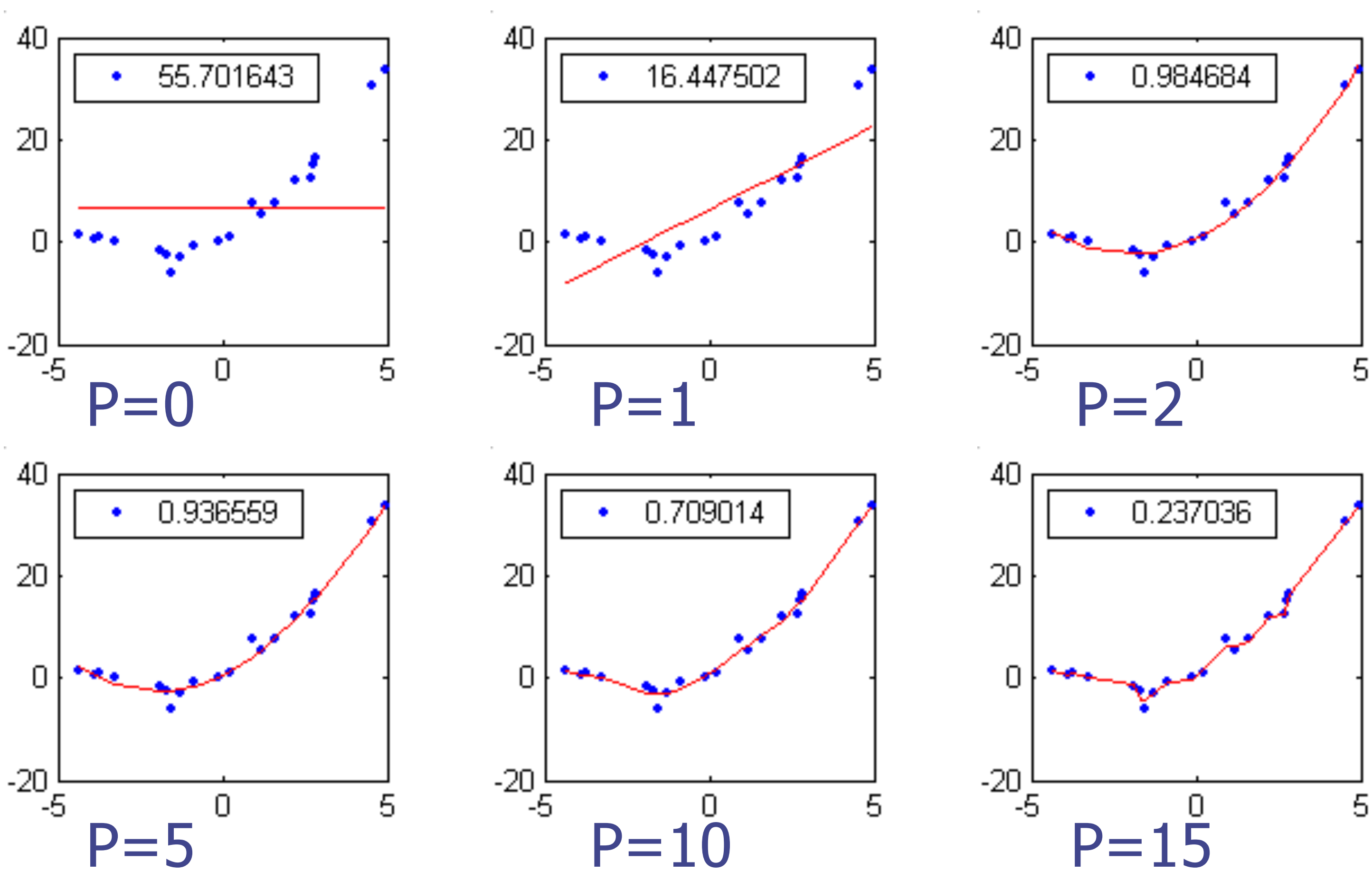
- Back to 1-dim x (D=1) BUT Nonlinear
- Polynomial: $f\left(x;\theta\right) = \sum_{p=1}^{P} \theta_p x^p + \theta_0$
- Writing Risk:

$$R\left(\theta\right) = \frac{1}{2N}\left\| \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} - \begin{bmatrix} 1 & x_1^1 & \ldots & x_1^P \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_N^1 & \ldots & x_N^P \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_P \end{bmatrix} \right\|^2$$

- Order-P polynomial regression fitting for 1D variable is same as P-dimensional linear regression!
- Construct a multidim x-vector from x scalar $\mathbf{x}_i = \begin{bmatrix} x_i^0 & x_i^1 & x_i^2 & x_i^3 \end{bmatrix}^T$
- More generally any $\mathbf{x}_i = \begin{bmatrix} \phi_0\left(x_i\right) & \phi_1\left(x_i\right) & \phi_2\left(x_i\right) & \phi_3\left(x_i\right) \end{bmatrix}^T$

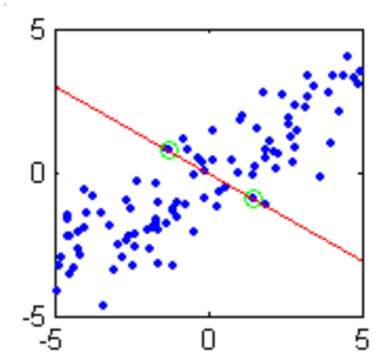
Underfitting/Overfitting

- Try varying P. Higher P fits a more complex function class
- Observe R(θ*) drops with bigger P



Evaluating The Regression

- Unfair to use empirical to find best order P
- High P (vs. N) can overfit, even linear case!
- min R(θ*) not on training but on future data
- Want model to *Generalize* to future data



True loss: $$R_{true}(\theta) = \int P(x,y)\tfrac{1}{2}\left(y - \theta^T x\right)^2 dx\, dy$$

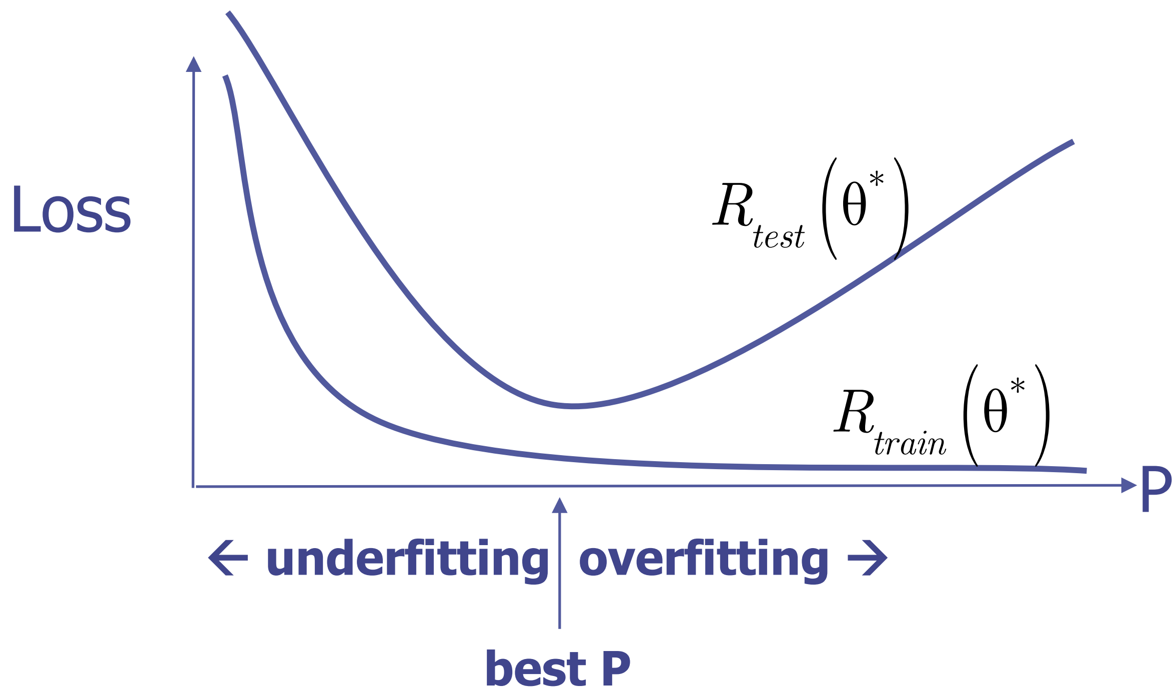
- One approach: split data into training / testing portion

$$\left\{(x_1, y_1), \ldots, (x_N, y_N)\right\} \qquad \left\{(x_{N+1}, y_{N+1}), \ldots, (x_{N+M}, y_{N+M})\right\}$$

- Estimate θ* with training loss: $R_{train}(\theta) = \frac{1}{2N}\sum_{i=1}^{N}\left(y_i - \theta^T x_i\right)^2$
- Evaluate P with testing loss: $R_{test}(\theta) = \frac{1}{2M}\sum_{i=N+1}^{N+M}\left(y_i - \theta^T x_i\right)^2$

Crossvalidation

- Try fitting with different polynomial order P
- Select P which gives lowest $R_{test}(\theta^*)$



- Think of P as a measure of the complexity of the model
- Higher order polynomials are more flexible and complex