

Linear Regression

EECS 349 Spring 2017

slides from Bryan Pardo, Mark Cartwright;
(also contains ideas and a few images from wikipedia and books by
Alpaydin, Duda/Hart/ Stork, and Bishop.)

Outline

- ▶ **Announcements**
 - ▶ Homework #2 assigned last night (due next Friday!)
- ▶ **Linear regression**

Regression Learning

There is a set of possible examples $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$

Each example is a **vector** of k **real valued attributes**

$$\mathbf{x}_i = \langle x_{i1}, \dots, x_{ik} \rangle$$

There is a target function that maps X onto some **real value** Y

$$f : X \rightarrow Y$$

The DATA is a set of tuples $\langle \text{example}, \text{response value} \rangle$

$$\{\langle \mathbf{x}_1, y_1 \rangle, \dots, \langle \mathbf{x}_n, y_n \rangle\}$$

Find a **hypothesis** h such that...

$$\forall \mathbf{x}, h(\mathbf{x}) \approx f(\mathbf{x})$$



Why *Linear* Regression?

- ▶ Easily understood/interpretable
- ▶ Well-studied
- ▶ Computationally Efficient



Linear Regression Assumption

- ▶ Response is a linear function of input, plus Gaussian Noise

Observed response $y = f(\mathbf{x}) + \varepsilon$

Where $\varepsilon \sim \mathcal{N}(0, \sigma^2)$

Hypothesis Space

- ▶ Each hypothesis characterized by a weight vector $\mathbf{w}$

$$h(\mathbf{x}) = w_0 + w_1x_1 + w_2x_2 + \dots w_kx_k$$

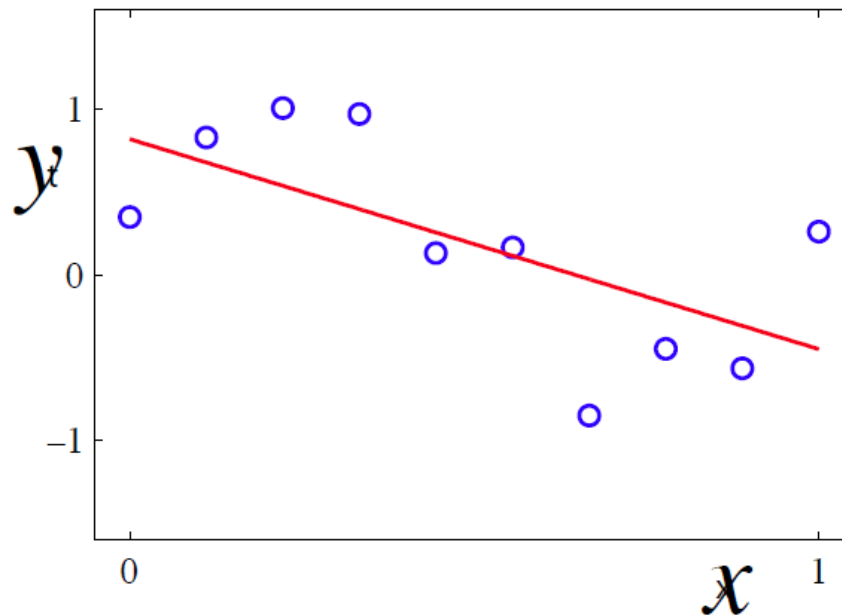
$$\mathbf{W} = \langle w_0, w_1, \dots, w_k \rangle$$

- ▶ **Goal:** Find a good $\mathbf{w}$
 - ▶ (One that minimizes some error criterion)

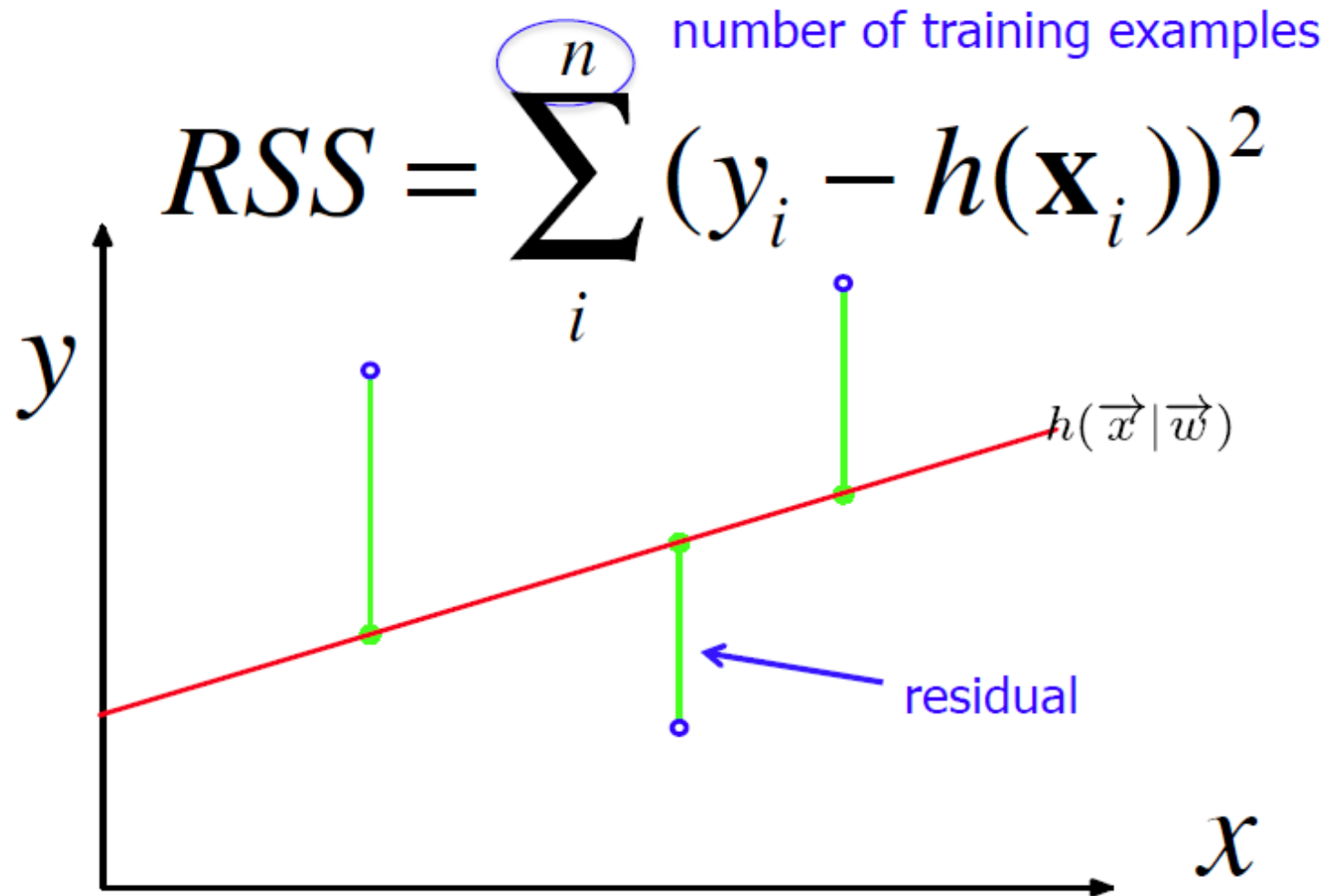
One-dimensional LR

- x has 1 attribute a (predictor variable)
- Hypothesis function is a line:

Example: $\hat{y} = h(x) = w_0 + w_1 x$

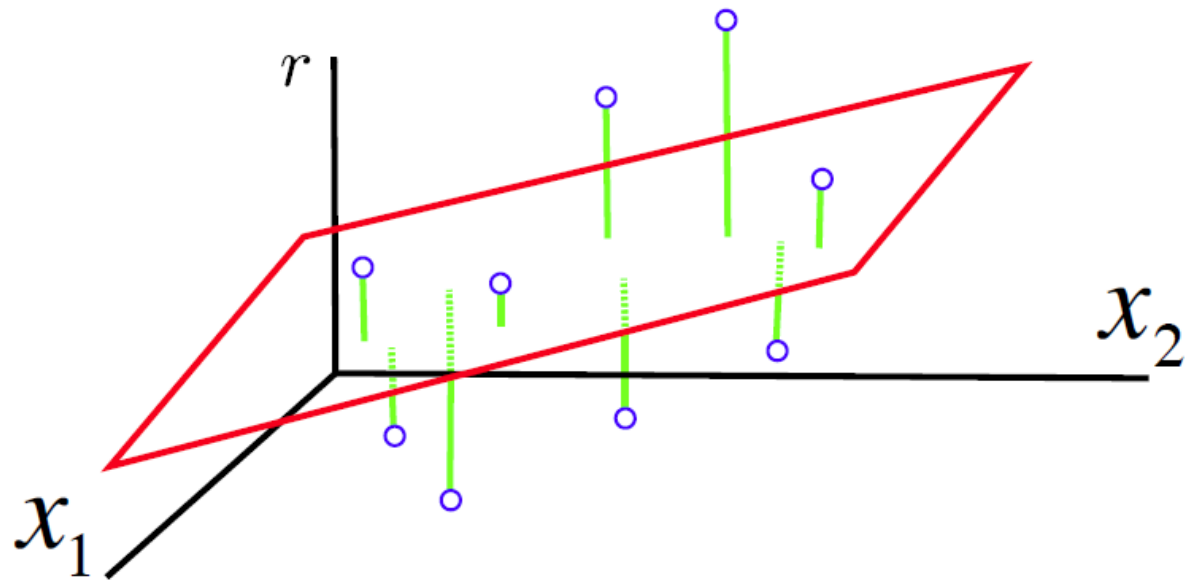


Minimize RSS (sum of squared residuals)



Multivariate Linear Regression

$$h(\mathbf{x}) = w_0 + w_1x_1 + w_2x_2 + \dots w_kx_k$$



Create a new 0 dimension with 1 and append it to the beginning of every example vector $\mathbf{x}_i$

This placeholder corresponds to the offset w_0

$$\mathbf{x}_i = \langle 1, x_{i,1}, x_{i,2}, \dots, x_{i,k} \rangle$$

Format the data as a matrix of examples $\mathbf{x}$ and a vector of response values y ...

One training example

$$\mathbf{X} = \begin{bmatrix} 1 & x_{1,1} & \dots & x_{1,k} \\ 1 & x_{2,1} & \dots & x_{2,k} \\ \dots & \dots & \dots & \dots \\ 1 & x_{n,1} & \dots & x_{n,k} \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix}$$

Closed-form solution

Our goal is to find the weights of a function....

$$h(\mathbf{x}) = w_0 + w_1 x_1 + w_2 x_2 + \dots w_k x_k$$

...that minimizes the sum of squared residuals:

$$RSS = \sum_i^n (y_i - h(\mathbf{x}_i))^2$$

It turns out that there is a close-form solution to this problem!

$$\mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$



$$\begin{aligned}RSS(\mathbf{w}) &= \sum_{i=1}^n (y_i - h(\mathbf{x}_i))^2 \\&= \sum_{i=1}^n (y_i - w_0 - \sum_{j=1}^k x_{ij} w_j)^2 \\&= (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w})\end{aligned}$$



$$RSS(\mathbf{w}) = (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w})$$

$$\frac{\partial RSS}{\partial \mathbf{w}} = -2\mathbf{X}^T (\mathbf{y} - \mathbf{X}\mathbf{w})$$

$$0 = -2\mathbf{X}^T (\mathbf{y} - \mathbf{X}\mathbf{w})$$

$$0 = \mathbf{X}^T (\mathbf{y} - \mathbf{X}\mathbf{w})$$

$$0 = \mathbf{X}^T \mathbf{y} - \mathbf{X}^T \mathbf{X}\mathbf{w}$$

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

$$\mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$



You're familiar with linear regression where the input has k dimensions.

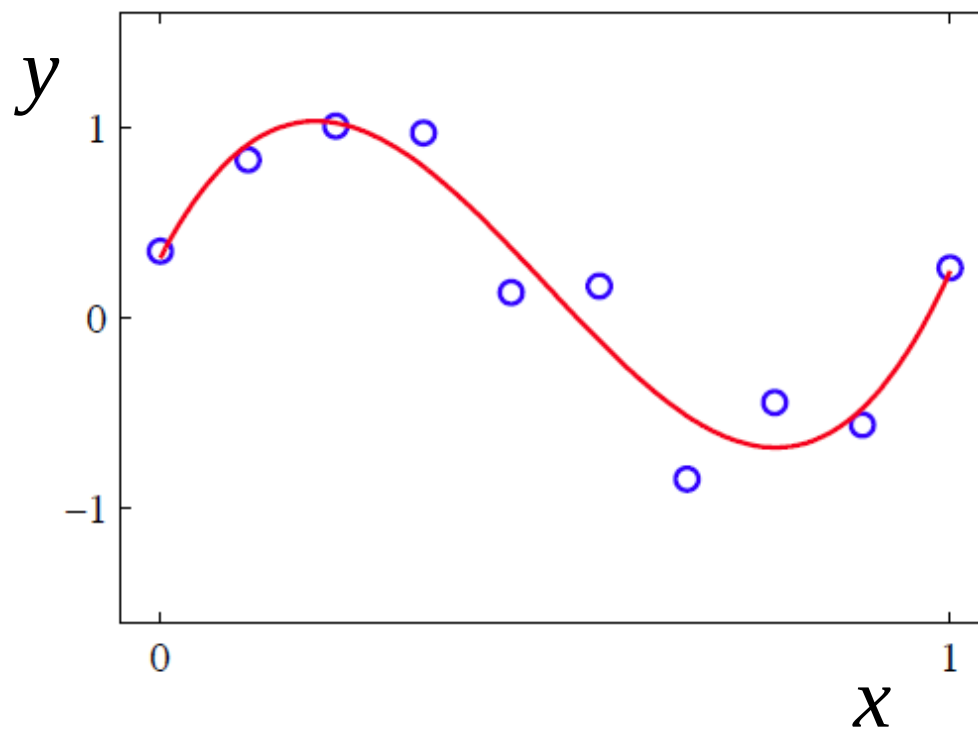
$$h(\mathbf{x}) = w_0 + w_1x_1 + w_2x_2 + \dots w_kx_k$$

We can use this same machinery to make polynomial regression from a one-dimensional input.....

$$h(x) = w_0 + w_1x + w_2x^2 + \dots w_kx^k$$



$$h(x) = w_0 + w_1 z + w_2 z^2 + w_3 z^3$$



Parameter estimation (analytically minimizing sum of squared residuals):

One training example

$$\mathbf{X} = \begin{bmatrix} 1 & z_1^1 & \dots & z_1^k \\ 1 & z_2^1 & \dots & z_2^k \\ \dots & \dots & \dots & \dots \\ 1 & z_n^1 & \dots & z_n^k \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix}$$

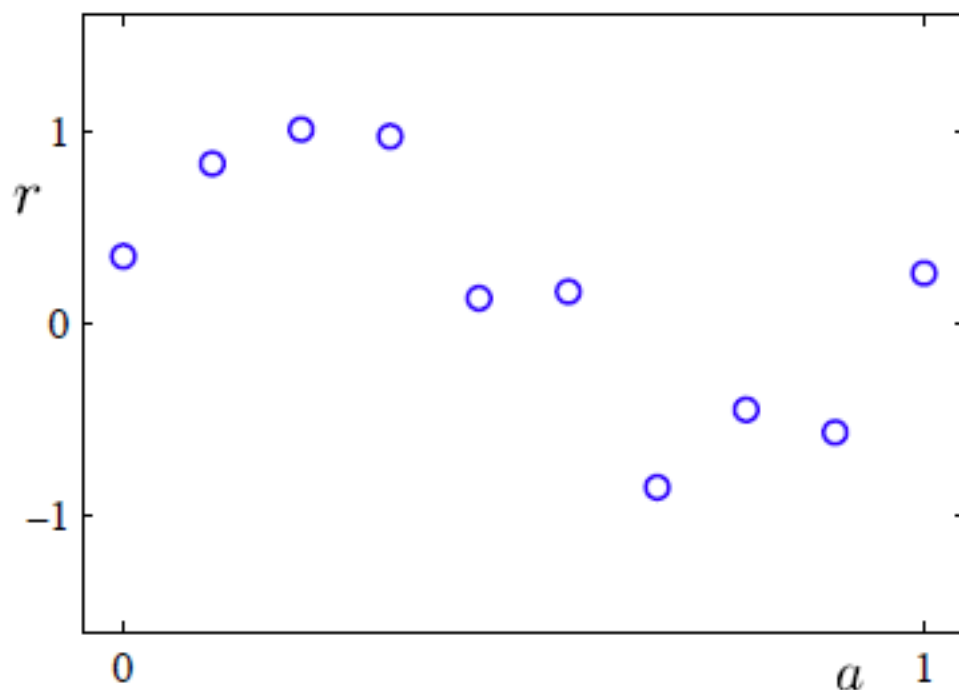
(Note, there is only 1 attribute z for each training example.
Those superscripts are powers, since we're doing polynomial regression)

$$\mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$



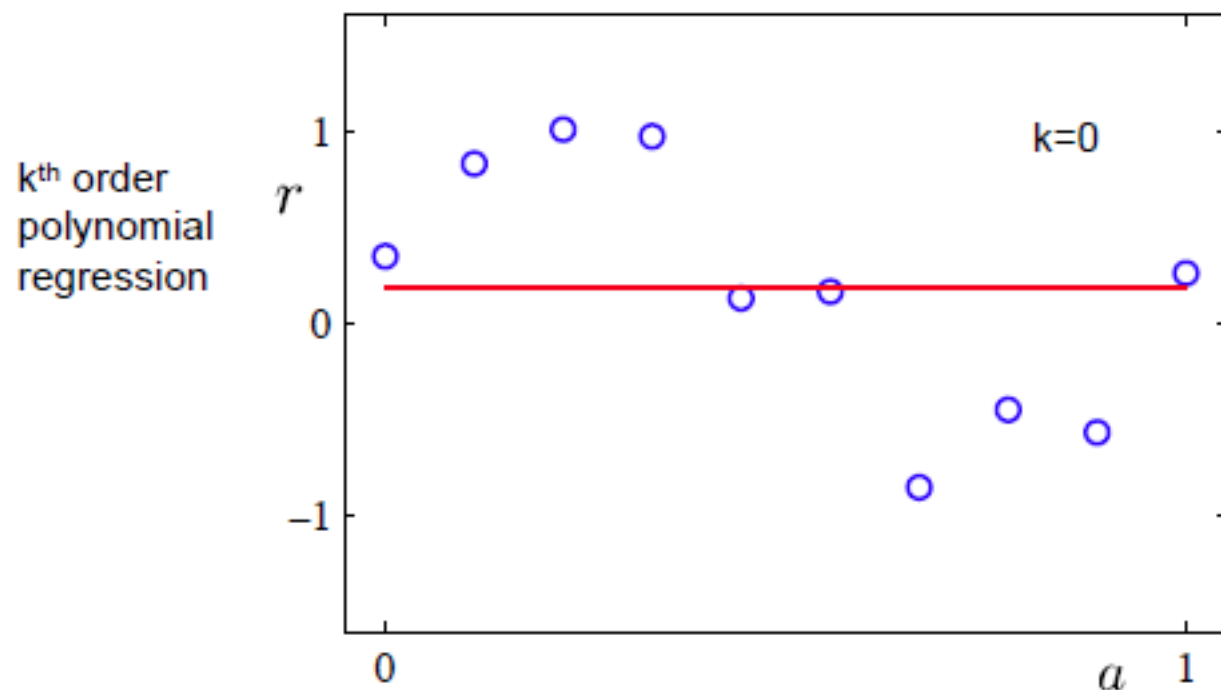
Tuning Model Complexity: Example

What is your hypothesis for $f(x)$?



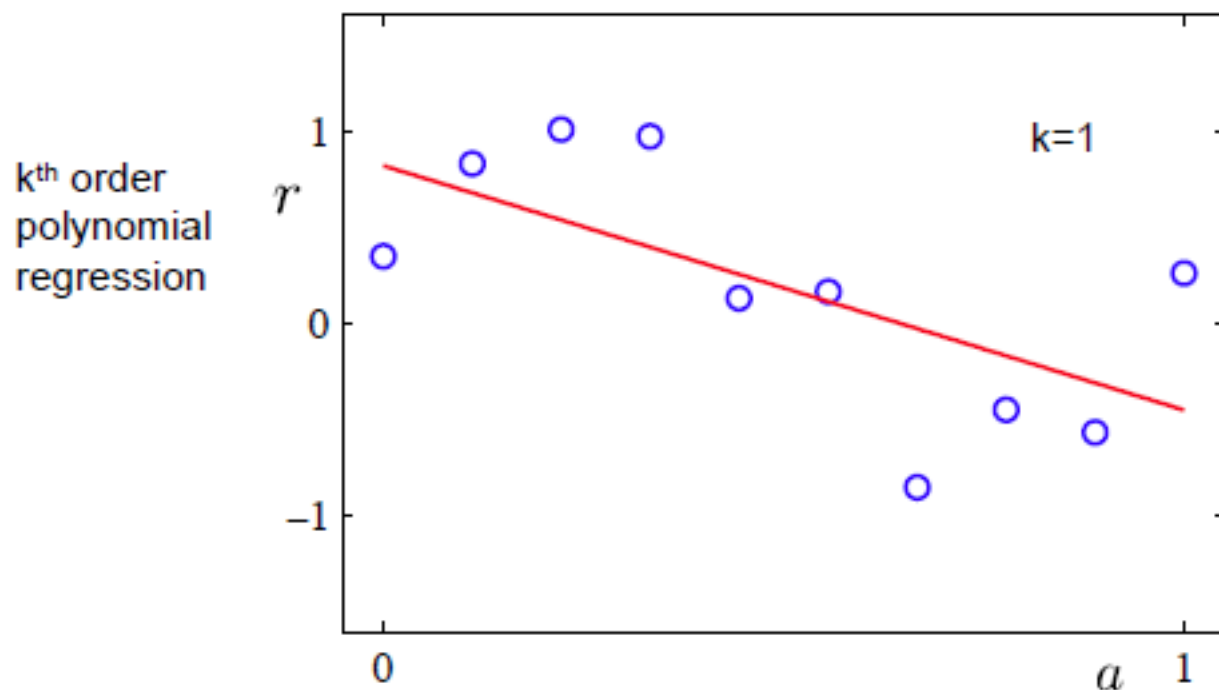
Tuning Model Complexity: Example

What is your hypothesis for $f(x)$?



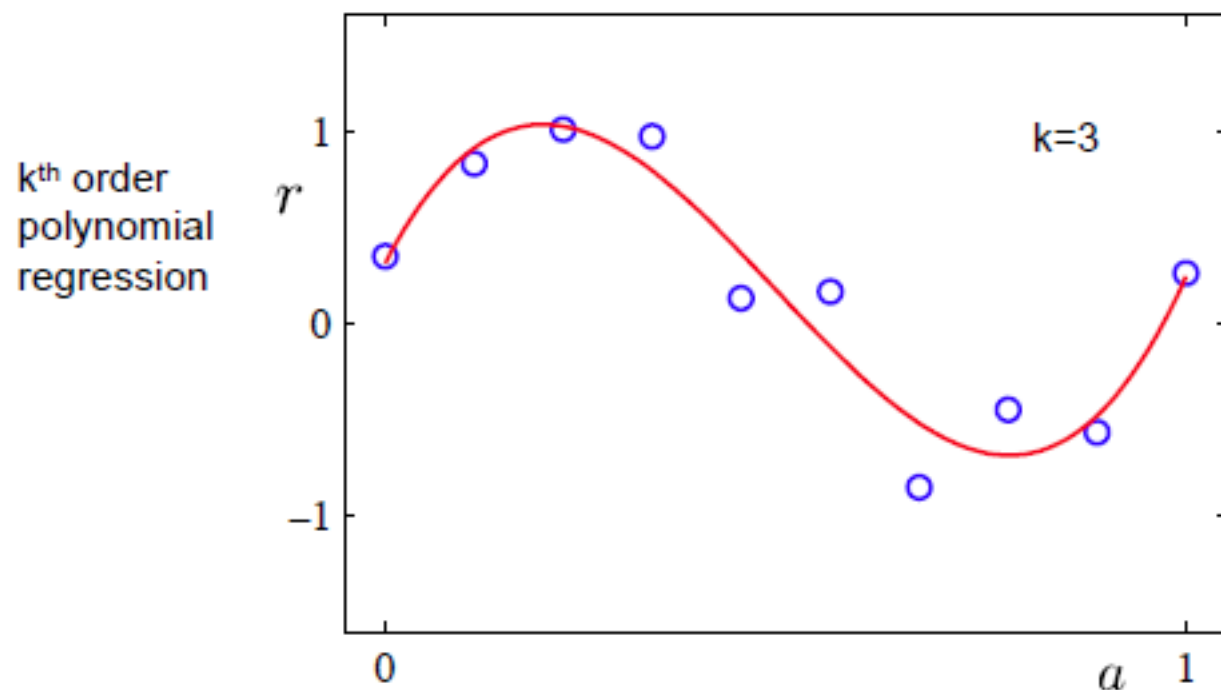
Tuning Model Complexity: Example

What is your hypothesis for $f(x)$?



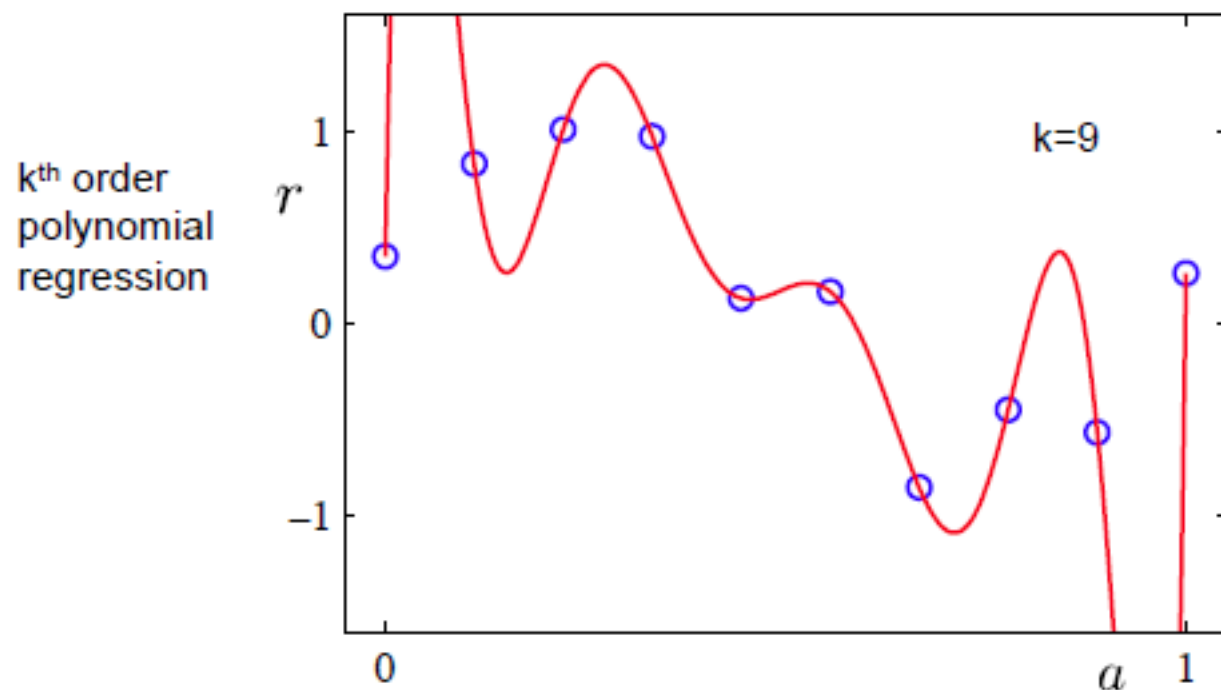
Tuning Model Complexity: Example

What is your hypothesis for $f(x)$?



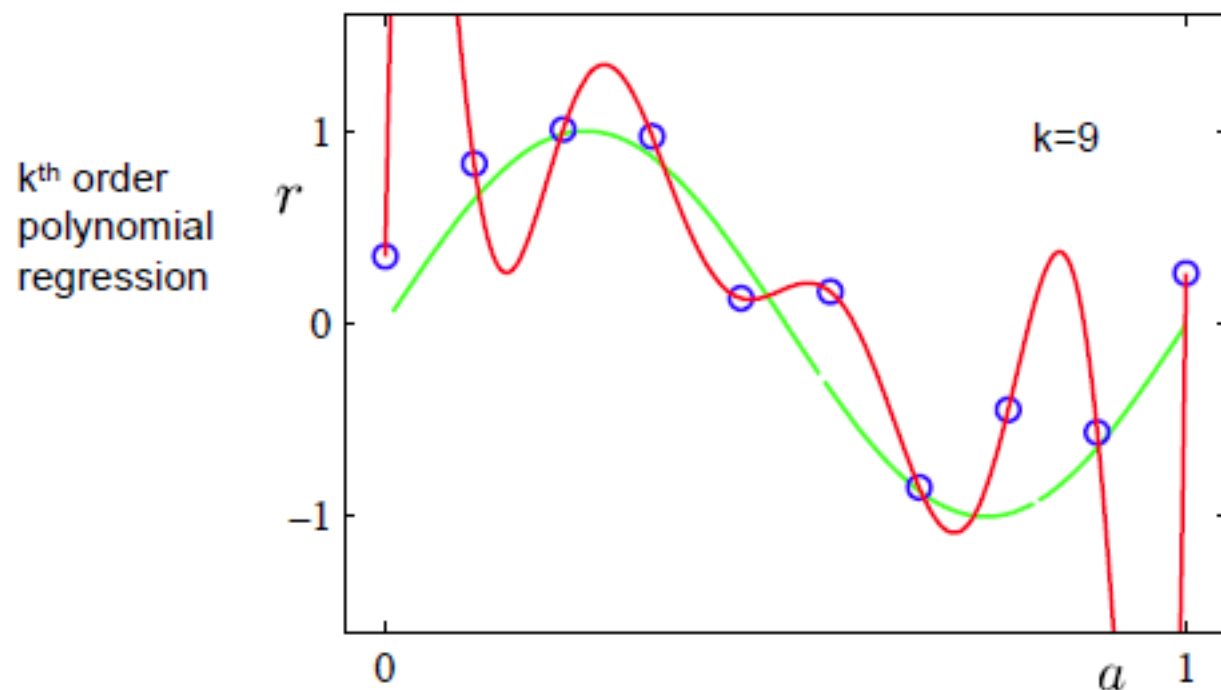
Tuning Model Complexity: Example

What is your hypothesis for $f(x)$?



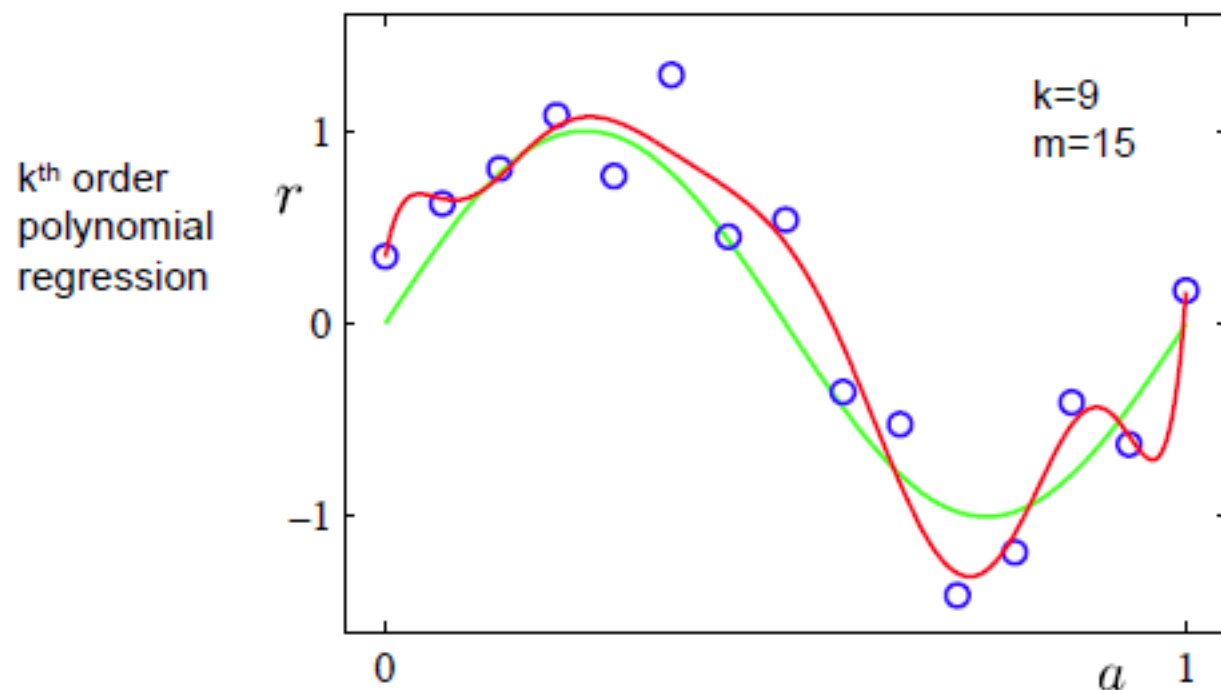
Tuning Model Complexity: Example

What is your hypothesis for $f(x)$?



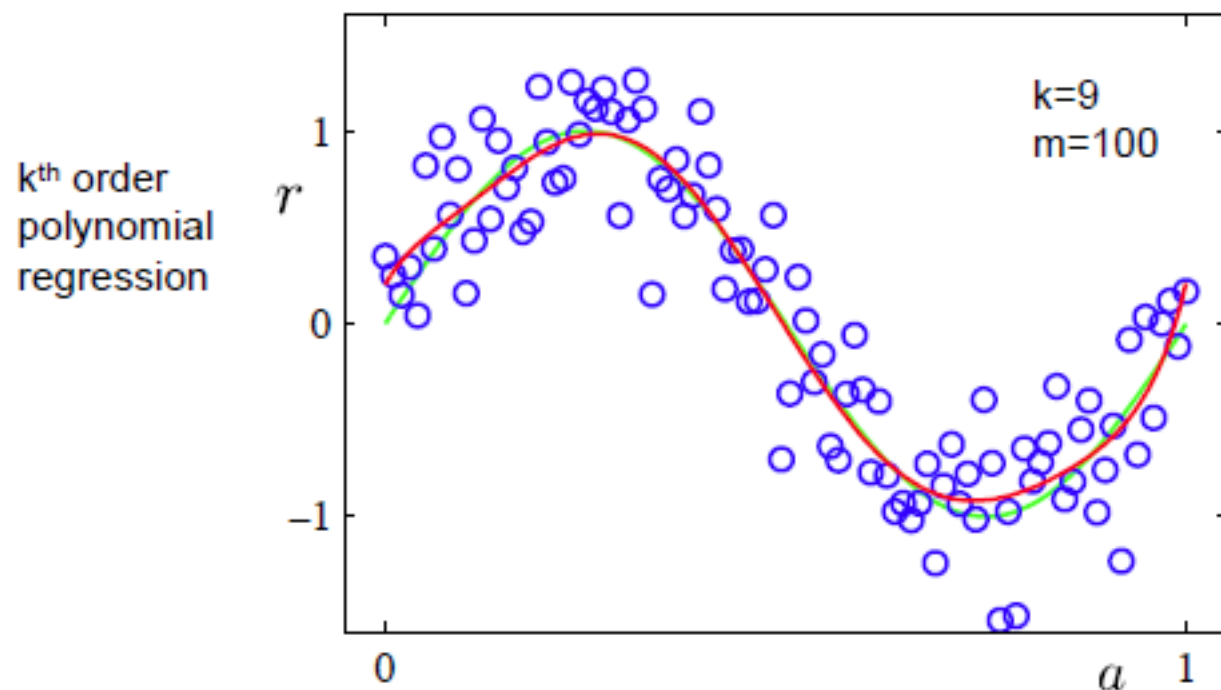
Tuning Model Complexity: Example

What happens if we fit to more data?



Tuning Model Complexity: Example

What happens if we fit to more data?



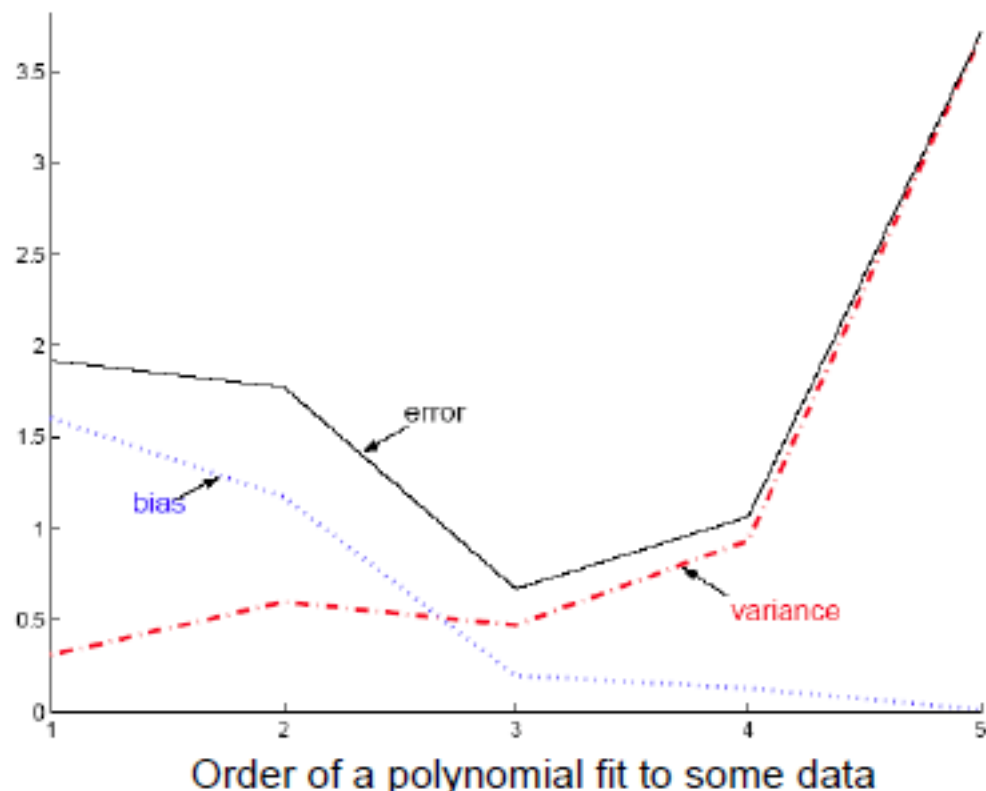
Bias and Variance of an Estimator

- Let X be a sample from a population specified by a true parameter θ
- Let $d=d(X)$ be an estimator for θ

$$\mathbb{E}[(d - \theta)^2] = \mathbb{E}[(d - \mathbb{E}[d])^2] + (\mathbb{E}[d] - \theta)^2$$

mean square error *variance* *bias²*

Bias and Variance



As we increase complexity, bias decreases (a better fit to data) and variance increases (fit varies more with data)

Reading

- ▶ Chapter 3 from Elements of Statistical Learning
 - ▶ <http://statweb.stanford.edu/~tibs/ElemStatLearn/>

