349:Machine Learning

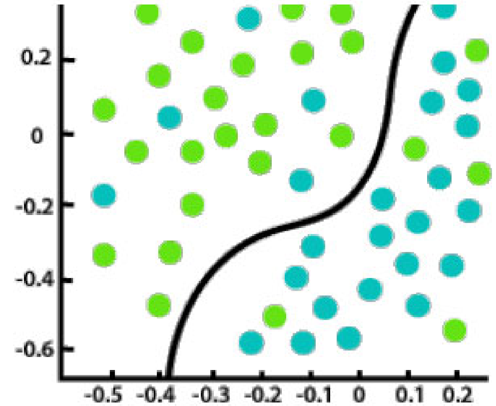
Fall 2024

Linear and Polynomial Regression

Classification vs. Regression

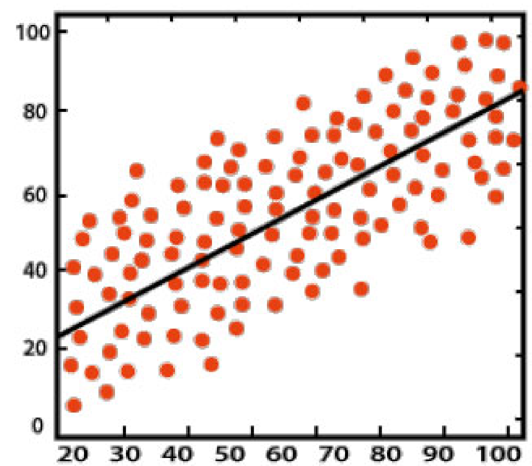
- Classification:

  Learning a function to map from a n-tuple to a ***discrete*** value from a finite set

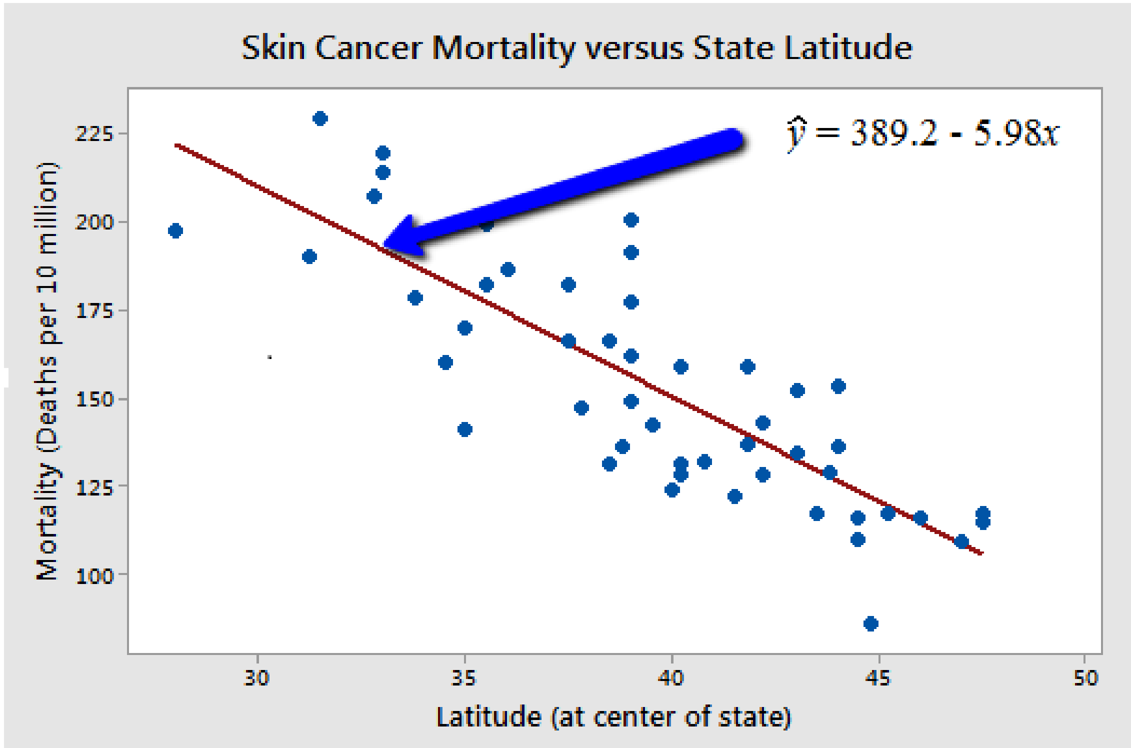


- Regression:

  Learning a function to map from a n-tuple to a ***continuous*** value



Some Examples…



- Height and weight: as height increases, you'd expect weight to increase, but not perfectly
- Driving speed and gas mileage: as driving speed increases, you'd expect gas mileage to decrease, but not perfectly.

Regression Learning Task

There is a set of possible examples $X = \{\mathbf{x_1}, \ldots \mathbf{x_n}\}$

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

$$\mathbf{x}_i =< x_{i1}, \ldots, x_{ik} >$$

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

$$f : X \to Y$$

The DATA is a set of tuples <example, response value>

$$\{< \mathbf{x}_1, y_1 >, \ldots < \mathbf{x_n}, y_n >\}$$

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

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

Why Use a Linear Regression Model

- Easily understood
- Interpretable
- Well studied by statisticians $\rightarrow$ many variations and diagnostic measures
- Computationally efficient

Linear Regression Model

Assumption: The observed response (dependent) variable, r, is the true function, f(x), with additive Gaussian noise, ε, with a 0 mean.

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

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

Assumption: The expected value of the response variable ***y*** is a linear combination of the k independent attributes/features)

The Hypothesis Space

Given the assumptions on the previous slide, our hypothesis space is the set of linear functions (hyperplanes)

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

(w_0 is the offset from the origin. You always need w_0)

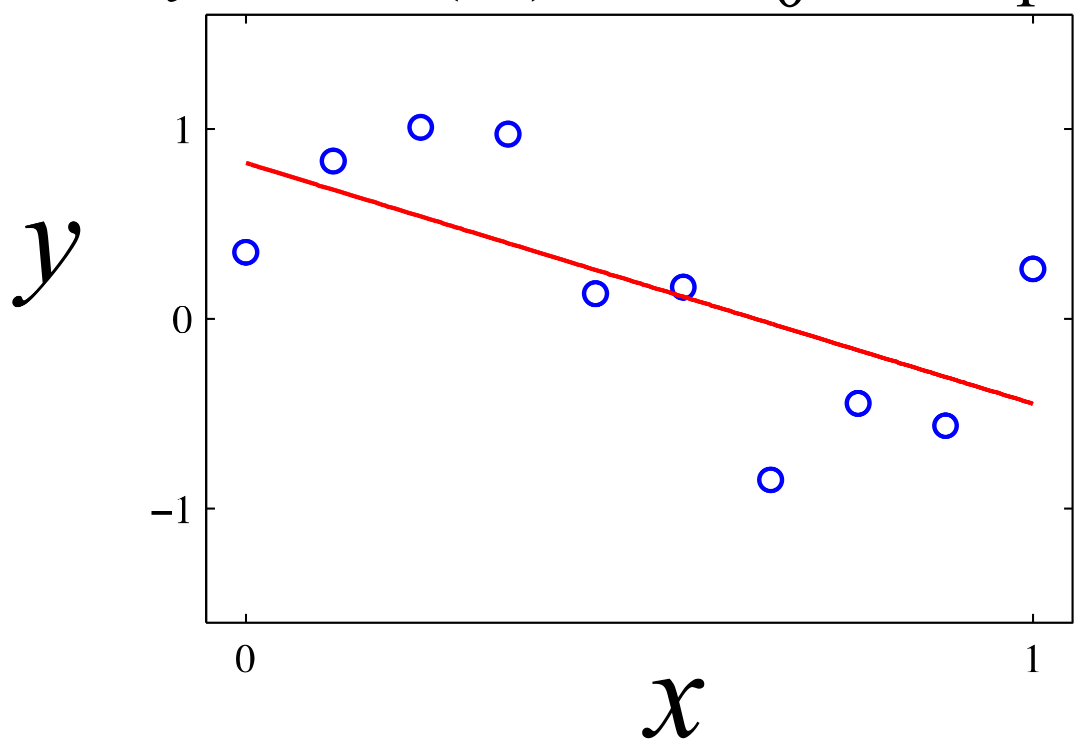
The goal is to learn a k+1 dimensional vector of weights that define a hyperplane minimizing an error criterion.

$$\mathbf{w} = < w_0, w_1, \ldots w_k >$$

Simple Linear Regression

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

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

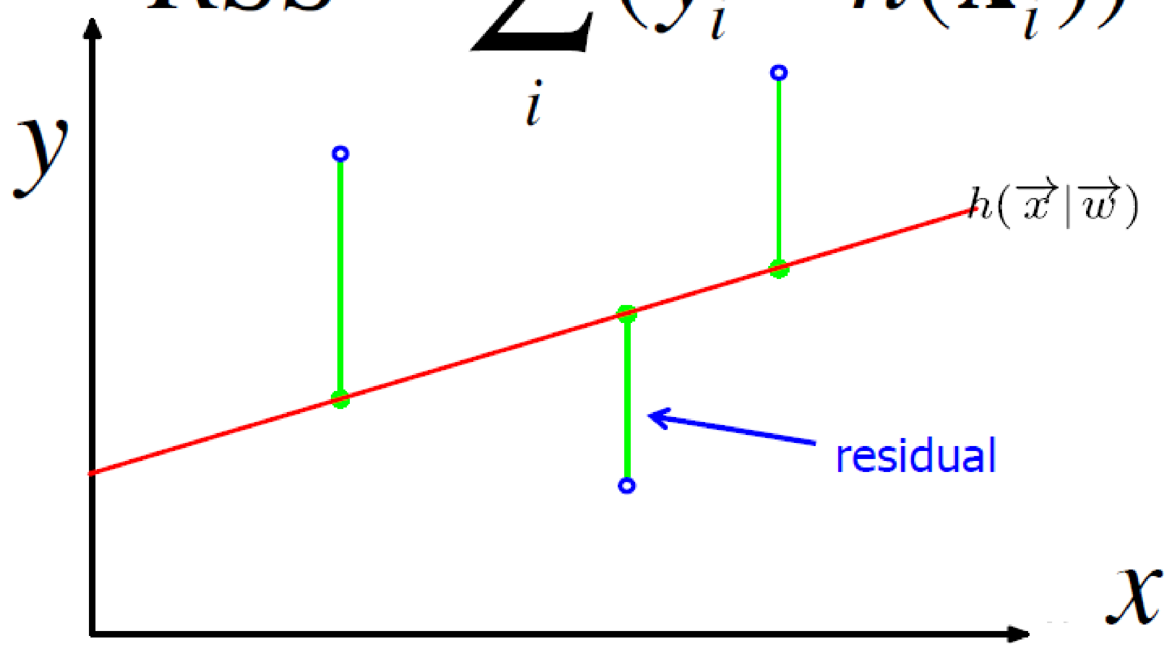


The Error Criterion

Typically estimate parameters by minimizing sum of squared residuals (RSS)…also known as the Sum of Squared Errors (SSE)

number of training examples

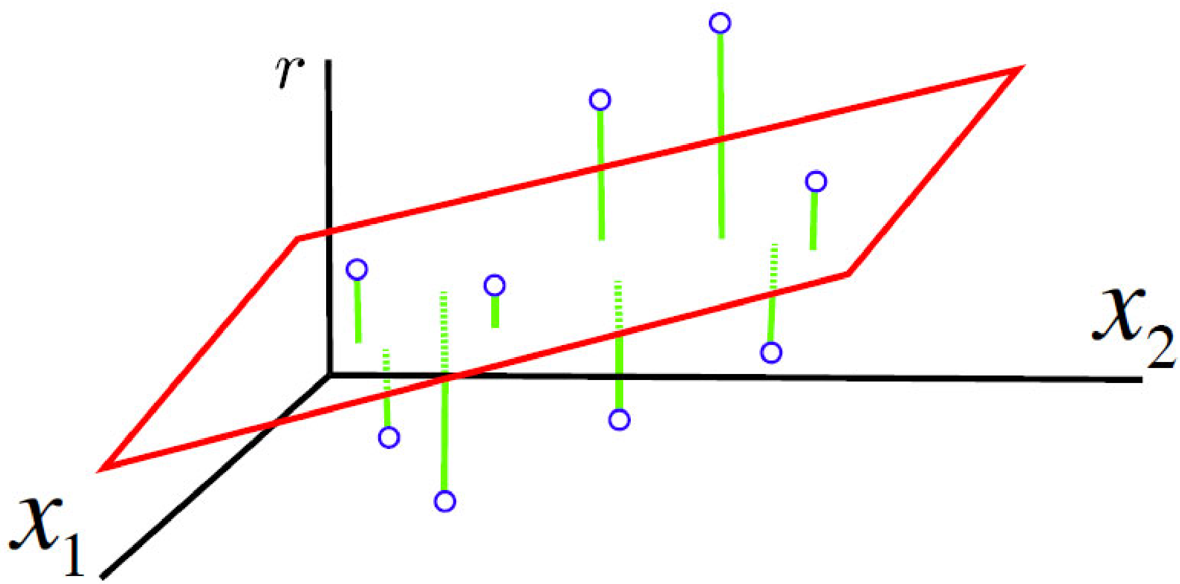
$$RSS = \sum_{i}^{n} (y_i - h(\mathbf{x}_i))^2$$



Multiple (Multivariate*) Linear Regression

- Many attributes $x_1, \ldots x_k$
- h(**x**) function is a hyperplane

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



Some Math…

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

$$\boldsymbol{w} = < w_0, \ldots, w_k > \in \mathbb{R}^{Kx1}$$

$$\boldsymbol{x} = < 1, x_1, \ldots, x_k >^T \in \mathbb{R}^{Kx1}$$

$$\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{x}^{\boldsymbol{T}} \boldsymbol{w}$$

Formatting the Data

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

$$\mathbf{x}_i = < 1, x_{i,1}, x_{i,2} \ldots, x_{i,k} >$$

Format the data as a matrix of examples **x** and a vector of response values *y*...

One training example

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

There Is a Closed-Form Solution!

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

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

…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!

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

Just plug your training data into the above formula and the best hyperplane comes out!

There Is a Closed-Form Solution!

$$X \rightarrow X^T$$

(n x k) (k x n)

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

(? x ?)	=	(k x n)	(n x k)	(k x n)	(n x 1)
(? x ?)	=	(k x k)		(k x 1)	
(? x ?)	=	(k x 1)			

RSS in Vector/Matrix Notation

$$RSS(w) = \sum_{i=1}^{n} \left(y_i - h(x_i)\right)^2$$

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

$$= \sum_{i=1}^{n} \left(y_i - x^T w\right)^T \left(y_i - x^T w\right)$$

$$= (y - Xw)^T (y - Xw)$$

$\mathbb{R}^{1xn}$ $\mathbb{R}^{nx1}$

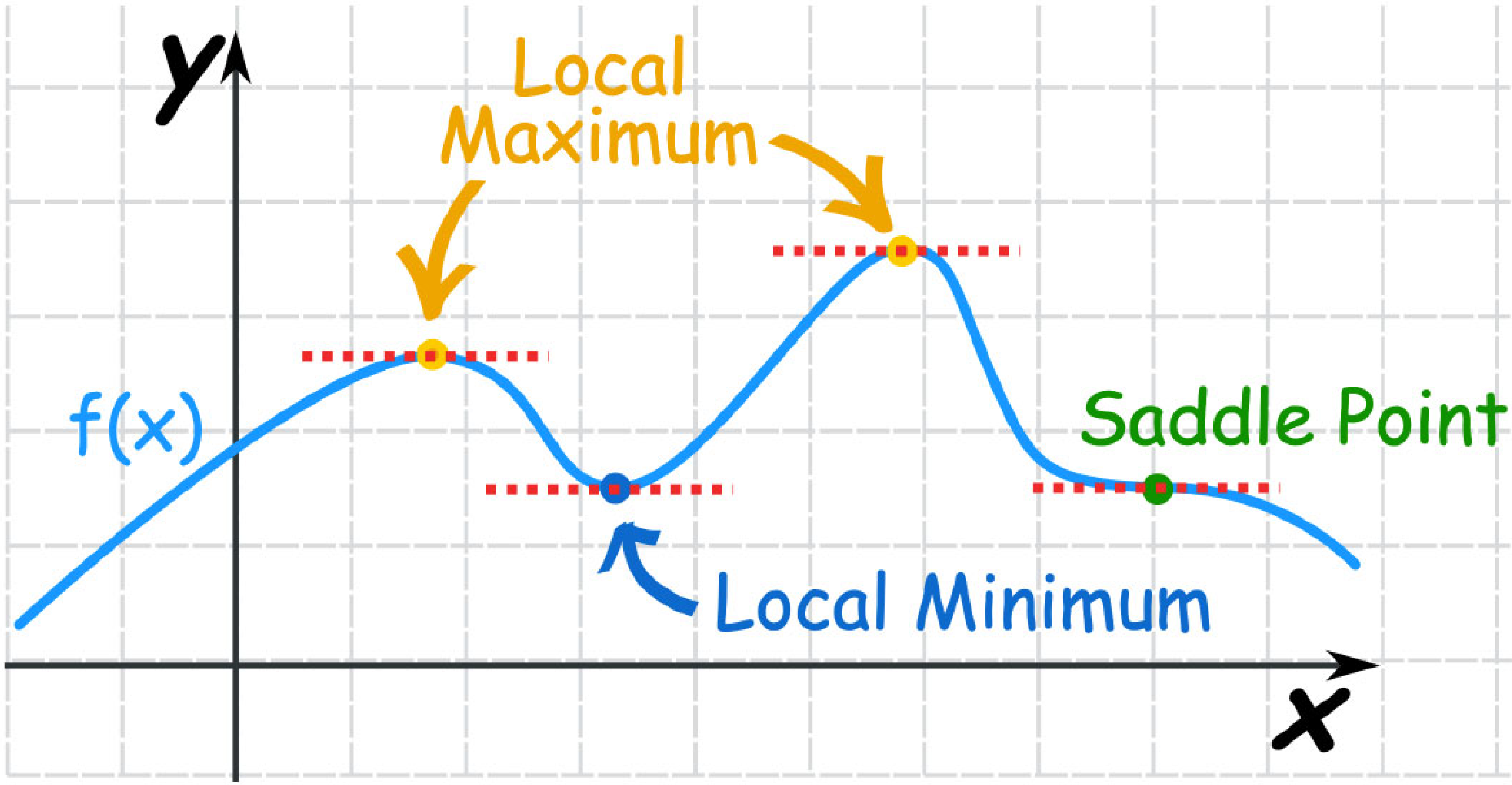
Some Math for Understanding Notation

$$\boldsymbol{x}^2 = \boldsymbol{x}^T \boldsymbol{x}$$

$$\boldsymbol{x} = [1, 0, 2]^T$$

$$\boldsymbol{x}^2 = \boldsymbol{x}^T \boldsymbol{x} = [1, 0, 2] * \begin{pmatrix} 1 \\ 0 \\ 2 \end{pmatrix} = 1^2 + 0^2 + 2^2 = 5$$

Gradient Descent Methodologies



Deriving the Formula for w

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

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

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

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

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

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

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

Is *X* Invertible

- We said there was a closed form solution:

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

- This presupposes matrix $(\mathbf{X}^T\mathbf{X})$ is invertible (non singular) and we can therefore find $(\mathbf{X}^T\mathbf{X})^{-1}$

- If two columns of X are exactly linearly related and thus not independent, then $(\mathbf{X}^T\mathbf{X})$ is NOT invertible

- What then?

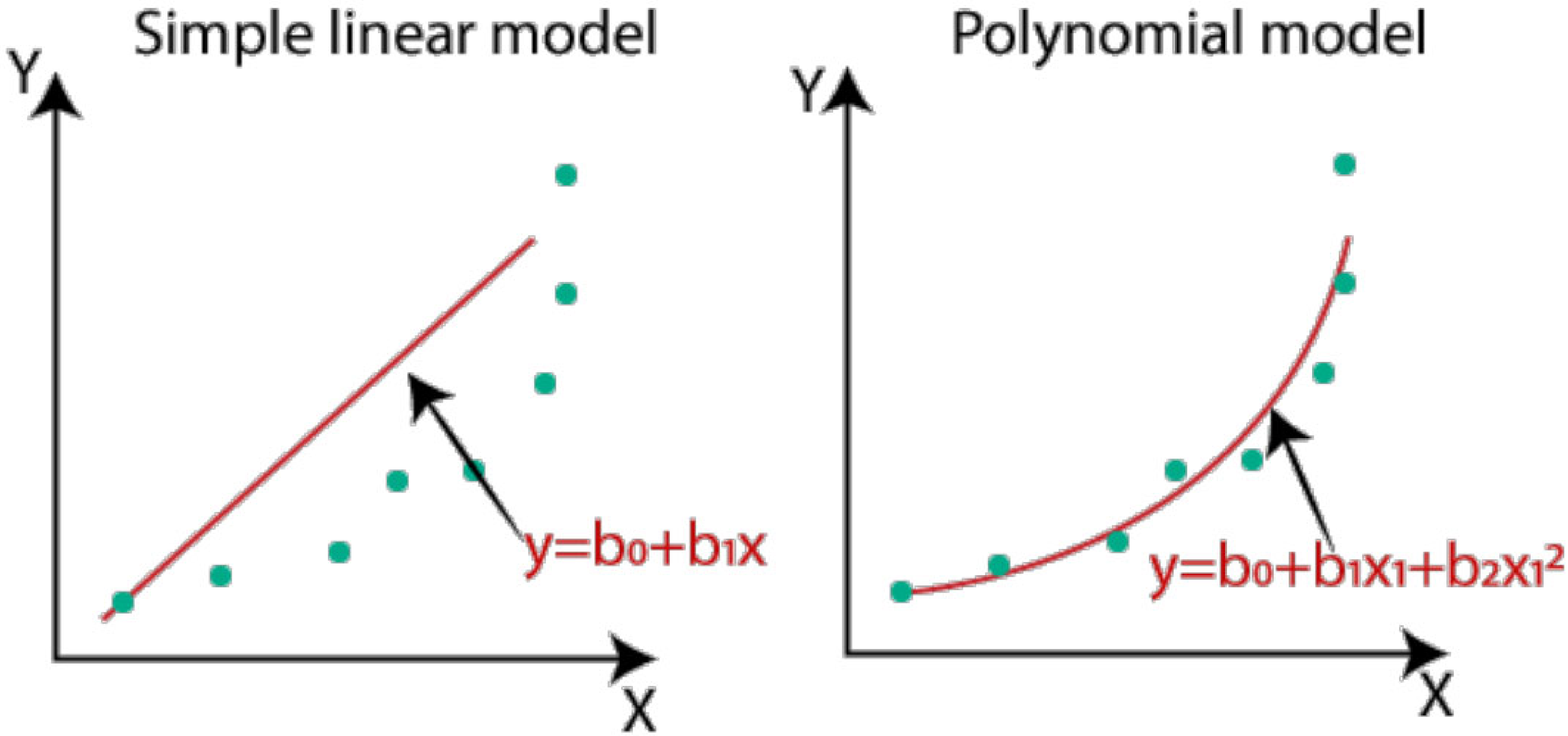
Dealing with a Singular X

- We need to make every column of X independent.

- The easy way: add a small amount random noise (with an expected value of 0) to X.
  - This is useful when you can't get rid of redundant columns for some reason
  - For example, your input data file is a 1000 examples of a constant value . You still want the code to return something, so you add a touch of noise and it will run and return something.

- The (often) better way: do dimensionality reduction to get rid of those redundant columns.

Polynomial Regression



Formulating a Polynomial Regression

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

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

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

$$h(x) = w_0 + w_1 x + w_2 x^2 + \ldots w_k x^k$$

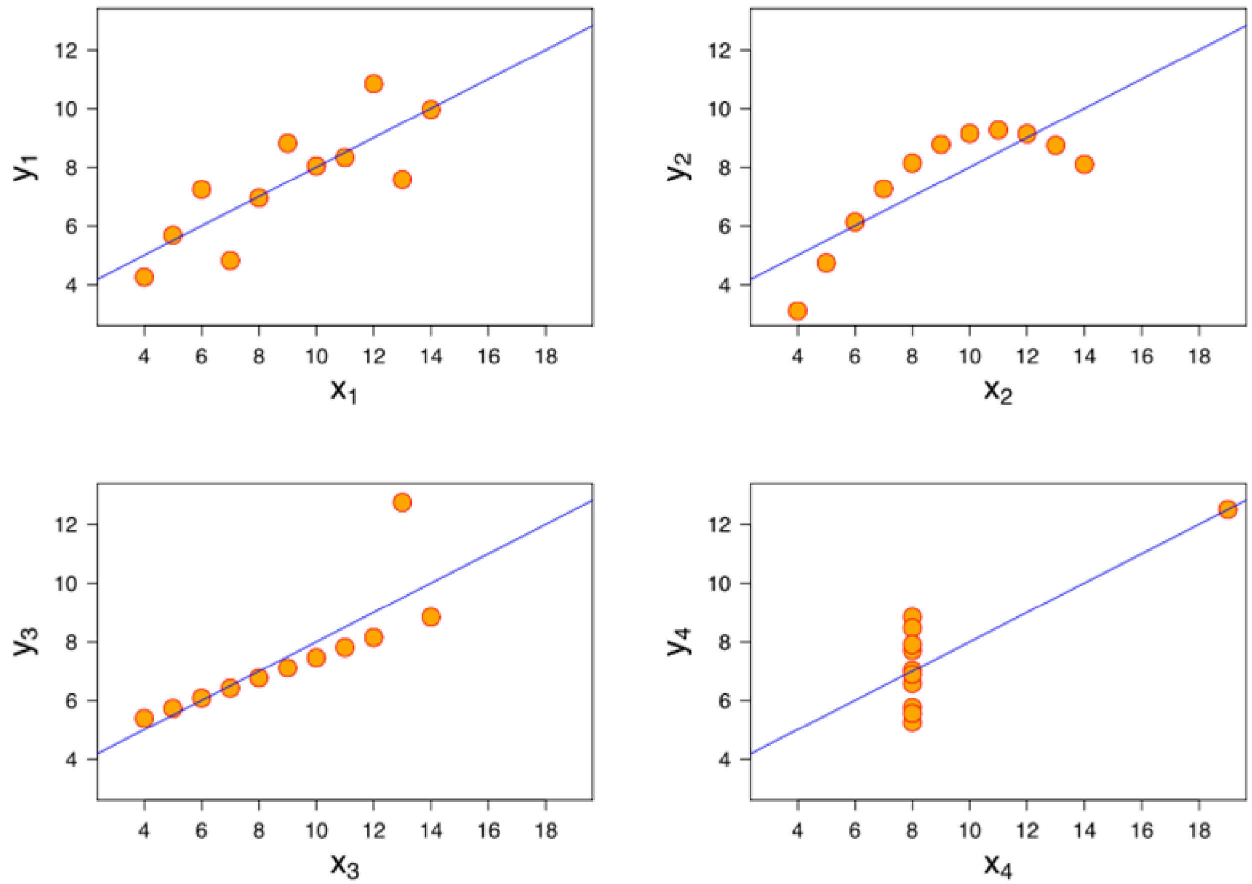
Coefficient of Determination

- the **coefficient of determination**, or $\boldsymbol{R^2}$ indicates how well data points fit a line or curve. We'd like R^2 to be close to 1

$$R^2 = 1 - E_{RSE}$$

$$E_{RSS} = \frac{\sum_{i}^{n} (y_i - h(\mathbf{x}_i))^2}{\sum_{i}^{n} (y_i - \bar{y})^2} \qquad \text{where } \bar{y} \text{ is the sample mean}$$

Don’t Rely On Metrics Only -- Visualize!



For all 4 sets: same mean and variance for x, same mean and variance (almost) for y, and same regression line and correlation between x and y (and therefore same R-squared).

Summary of Regression Models

- Easily understood
- Interpretable
- Well studied by statisticians
- Computationally efficient
- Can handle non-linear situations if formulated properly
- Bias/variance tradeoff (occurs in all machine learning)
- Visualize!