Data, Environment and Society: Lecture 9: Intro to regression

Instructor: Duncan Callaway
GSI: Seigi Karasaki

September 20, 2018

Announcements

Today

- Review bias-variance tradeoff
- Regression
  - K-nearest neighbors
  - Linear least squares

Reading

- Today's lecture draws from DS100 Ch10, ISLR Ch 2, ISLR Ch 3.1
- For next week
  - Read Alstone *et al* for next Tuesday – in class discussion
  - Review ISLR Ch 3.1-3.2

(review) Error or residual?

$$
\begin{aligned}
y_i &= f(x_i) + \epsilon_i && \text{the “true” model, if one exists.} \\
y_i &= \hat{f}(x_i) + e_i && \text{the relationship between the data and the estimate.}
\end{aligned}
$$

(review) Error or residual?

$$y_i = f(x_i) + \epsilon_i \qquad \text{the "true" model, if one exists.}$$
$$y_i = \hat{f}(x_i) + e_i \qquad \text{the relationship between the data and the estimate.}$$

So:

$$\epsilon_i \qquad \text{variation in } y \text{ that is uncorrelated with } x.$$
$$e_i = y_i - \hat{f}(x_i) \qquad \text{the "residual" between the data and the estimate.}$$

(Review) How to evaluate how well a model performs?

Generic term: the *Cost function.*

- Cost functions can be used to describe how much of the variation in the data can be captured by the model.
- Example: The mean squared error:

$$MSE = \frac{1}{n}((y_1 - \hat{y}_1)^2 + (y_2 - \hat{y}_2)^2 + \cdots + (y_n - \hat{y}_n)^2)$$

(Review) How to evaluate how well a model performs?

Generic term: the *Cost function.*

- Cost functions can be used to describe how much of the variation in the data can be captured by the model.
- Example: The mean squared error:

$$MSE = \frac{1}{n}((y_1 - \hat{y}_1)^2 + (y_2 - \hat{y}_2)^2 + \cdots + (y_n - \hat{y}_n)^2)$$

$$= \frac{1}{n}(e_1^2 + e_2^2 + \cdots + e_n^2)$$

(Review) How to evaluate how well a model performs?

Generic term: the *Cost function.*

- Cost functions can be used to describe how much of the variation in the data can be captured by the model.
- Example: The mean squared error:

$$MSE = \frac{1}{n}((y_1 - \hat{y}_1)^2 + (y_2 - \hat{y}_2)^2 + \cdots + (y_n - \hat{y}_n)^2)$$

$$= \frac{1}{n}(e_1^2 + e_2^2 + \cdots + e_n^2)$$

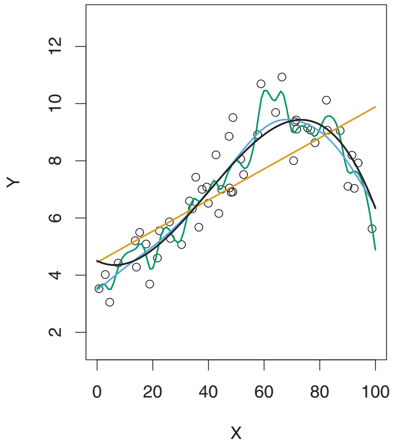
$$= \frac{1}{n}\sum_{i=1}^{n} e_i^2$$

(Review) A thought experiment from ISLR Ch 2

Suppose you have four different model forms to choose from. When you fit them to the data, you get this figure.

Which model should you choose?

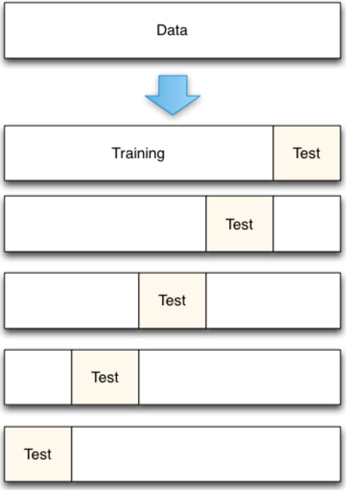
- The one that minimizes mean squared error?
- Careful! Doesn't the squiggly one minimize mean squared error?
- To do model selection we need to understand the concept of training and testing data.



(Review) Concept: Test and training data

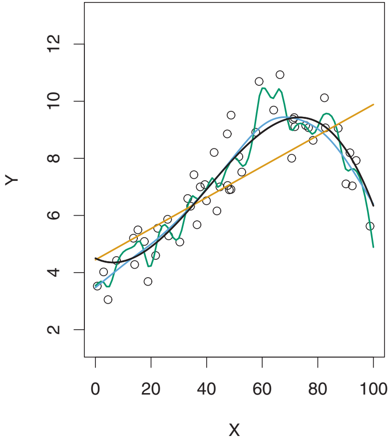
Choosing between different models can be done by partitioning your data in to “training” and “test” data.

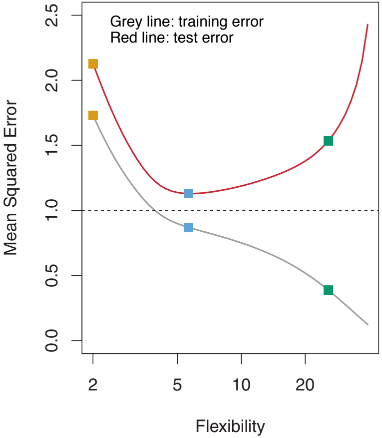
- “Training data”: The data we use to choose the parameters of an individual model.
- “Test data”: A set of data we withhold; it’s not for training. We use this data set to compare how different *models* perform relative to one another.



Source: kaggle.com

(Review) MSE for test and training data





What might a plot of MSE versus model "flexibility" look like?

Bias v. Variance

Bias:

- The propensity for a model to produce errors that are systematically high or low
- Bias can be positive in one range of the predictor and negative in another.

Variance

- The propensity for a model to make very different predictions if it is fit with two different training data sets that are sampled from the same population.

Total error can be decomposed:

$$\text{Avg } (y_0 - \hat{f}(x_0))^2 = \text{(variance in a prediction, across different training data)} + \text{(systematic bias)}^2 + \text{(variance in } y \text{ that's uncorrelated with } x\text{)}$$

Bias v. Variance

Bias:

- The propensity for a model to produce errors that are systematically high or low
- Bias can be positive in one range of the predictor and negative in another.

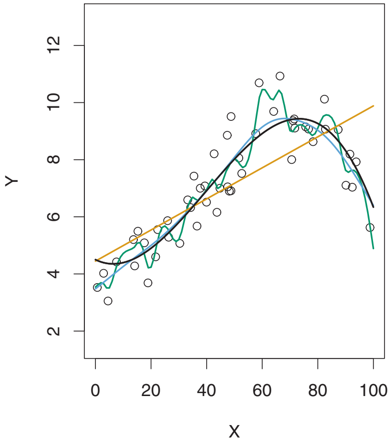
Variance

- The propensity for a model to make very different predictions if it is fit with two different training data sets that are sampled from the same population.

Total error can be decomposed:

$$\begin{aligned}\text{Avg } (y_0 - \hat{f}(x_0))^2 =& \text{(variance in a prediction, across different training data)} \\ &+ \text{(systematic bias)}^2 + \text{(variance in } y \text{ that's uncorrelated with } x\text{)} \\ =& \text{var}(\hat{f}(x_0)) + [\text{bias}(\hat{f}(x_0))]^2 + \text{var}(\epsilon_0)\end{aligned}$$

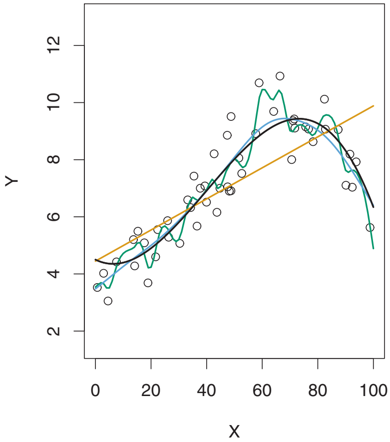
Bias v. Variance, ctd.



Which model has the greatest propensity for bias?

Which model has the greatest propensity for variance?

Bias v. Variance, ctd.

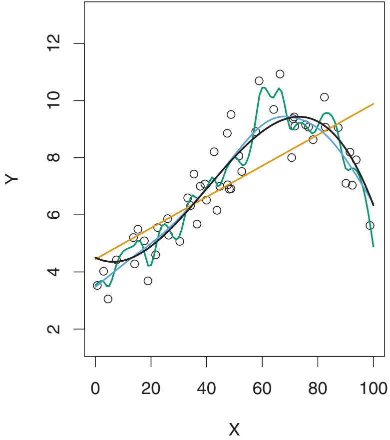


Which model has the greatest propensity for bias?

- The linear one. In ranges of x, it systematically under- or over-estimates.

Which model has the greatest propensity for variance?

Bias v. Variance, ctd.



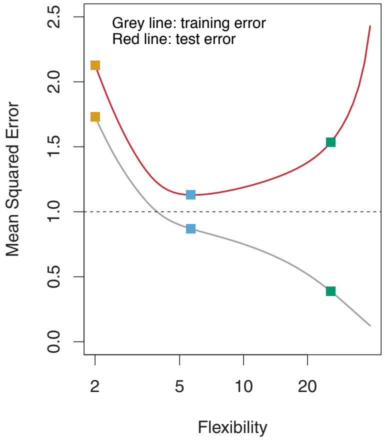
Which model has the greatest propensity for bias?

- The linear one. In ranges of x, it systematically under- or over-estimates.

Which model has the greatest propensity for variance?

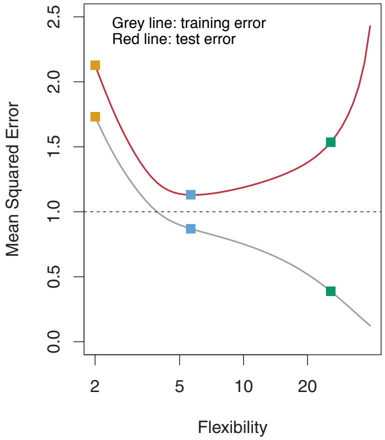
- The squiggly one. If we drew another sample of data, we'd probably get very different squiggles.

Decomposing bias-variance

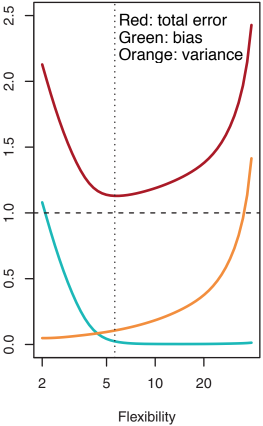


Take a moment to think about how bias and variance add up to make the red curve on the left. Try to draw bias and variance separately.

Decomposing bias-variance



Take a moment to think about how bias and variance add up to make the red curve on the left. Try to draw bias and variance separately.



Parametric vs. non-parametric models

The model examples we discussed Tuesday were **parametric**, meaning they relate inputs to outputs with a mathematical function defined by parameters.

But **non-parametric** models are also possible.

- These don't use functions with coefficients
- Instead the data *become* the model

It's easiest to see this by example using the K-nearest neighbors algorithm.

K-nearest neighbors (KNN)

We'll work with just a one-dimensional independent variable. For example,

- y_i could be NOx emissions from a power plant,
- x_i could be its coal use;
- different i would correspond to different power plants in different years.

Definitions:

- First, define proximity between two points as $|x_i - x_j|$
- Next, define $\mathcal{N}_i$ as the set of K points closest to x_i

K-nearest neighbors

The basic idea behind using KNN for regression (i.e. predicting a continuous variable or set of variables) is simple:

$$\hat{y}_j = \frac{1}{K} \sum_{i \in \mathcal{N}_j} y_i$$

In other words, the prediction equals the average of the K nearest points.

If you're working with KNN, what is your most important decision?

If you're working with KNN, what is your most important decision?

What is K?

Check of intuition: Would increasing K reduce or increase bias?

If you're working with KNN, what is your most important decision?

What is K?

Check of intuition: Would increasing K reduce or increase bias? **Increase!**

- Using a lower K would cause the estimates to more closely follow the underlying data.
- In the extreme, $K = 1$ would make the model equal the underlying data.
- At the other extreme, $K = n$ would make the model equal the sample mean.

Linear regression

Regression: A method to estimate the expected value of an output variable (y), *conditional* on one or more input values (x)

- KNN regression does this by averaging nearby values.
- Linear regression does this by fitting a linear function to the data.
- Broadly speaking, *regression* can be used for prediction.
- *Linear* regression specifically can also be used for inference.
- Many of the methods we'll work with later in the semester will be rooted in linear regression.

The basic model

- x_i: one dimensional independent variable
- y_i: one dimensional dependent variable

$$y_i = \hat{\beta}_0 + \hat{\beta}_1 x_i + e_i$$
$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$$

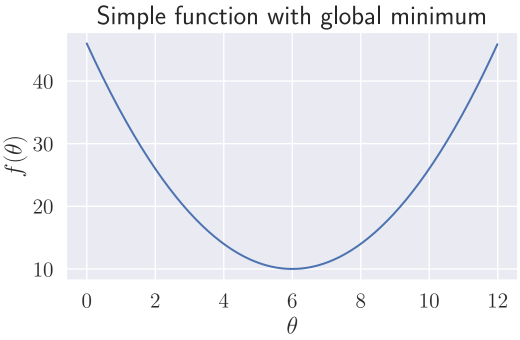
- We use the $\hat{\cdot}$ symbol to denote an estimate, or prediction

(extremely important) Side note: Optimality.

Define the “argument” that minimizes a function f with respect to θ as:

$$\theta^* = \arg\min_{\theta} f(\theta)$$

In the plot below, what’s θ^*?

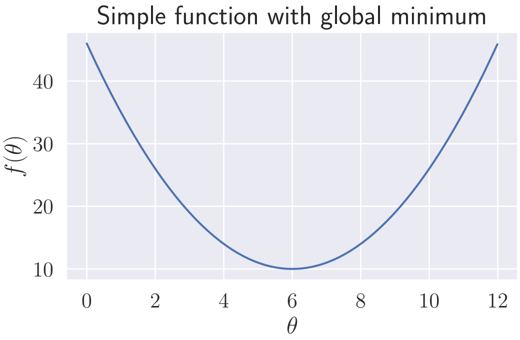


(extremely important) Side note: Optimality.

Define the "argument" that minimizes a function f with respect to θ as:

$$\theta^* = \arg\min_\theta f(\theta)$$

In the plot below, what's θ^*?



$$\theta^* = \arg\min_\theta f(\theta) = 6$$

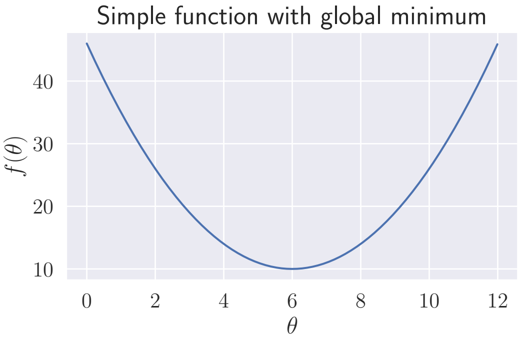
$$f(\theta^*) = 10$$

(extremely important) Side note: Optimality.

Define the "argument" that minimizes a function f with respect to θ as:

$$\theta^* = \arg\min_{\theta} f(\theta)$$

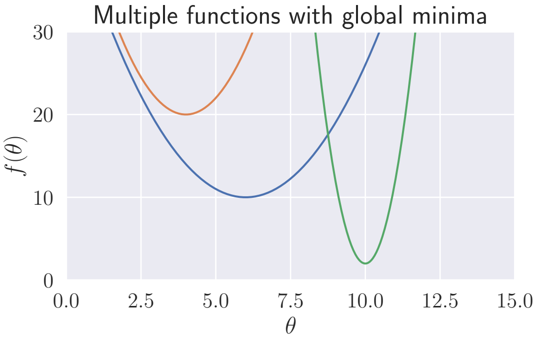
In the plot below, what's θ^*?



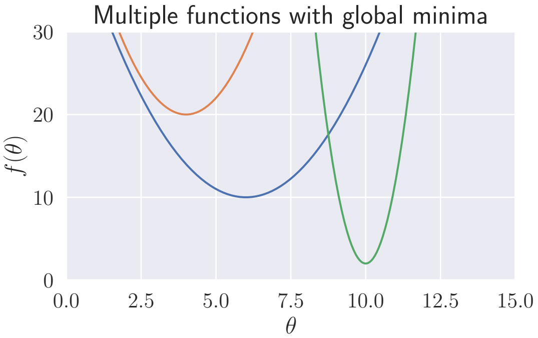
$$\theta^* = \arg\min_{\theta} f(\theta) = 6$$

$$f(\theta^*) = 10$$

What do the minima share in common?

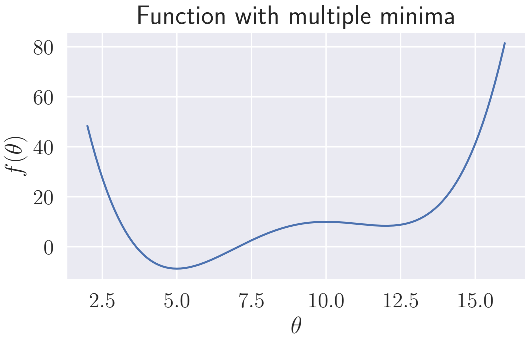


What do the minima share in common?

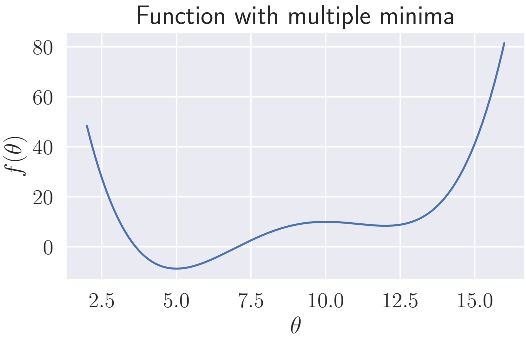


$$\left.\frac{\partial f(\theta)}{\partial \theta}\right|_{\theta^*} = 0$$

What's the challenge here?



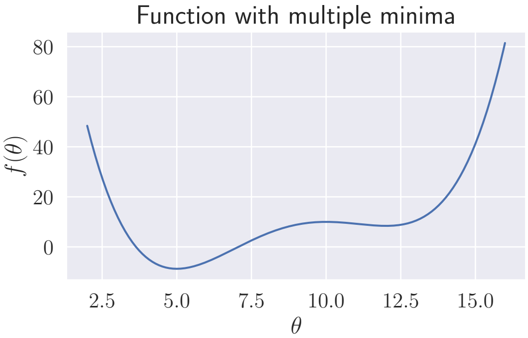
What's the challenge here?



$\frac{\partial f(\theta)}{\partial \theta} = 0$ at more than one point.

The function is said to be "non-convex"

What's the challenge here?



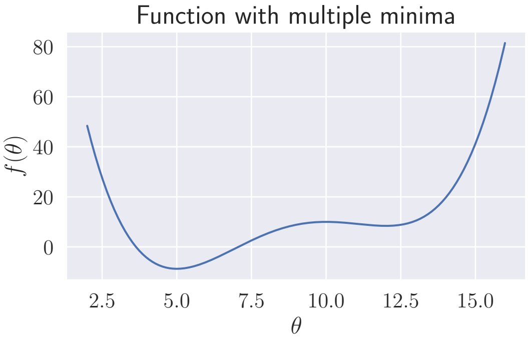
$\frac{\partial f(\theta)}{\partial \theta} = 0$ at more than one point.

The function is said to be "non-convex"

Which should we choose?

- We could enumerate all the solutions and choose the best.

What's the challenge here?



$\frac{\partial f(\theta)}{\partial \theta} = 0$ at more than one point.

The function is said to be "non-convex"

Which should we choose?

- We could enumerate all the solutions and choose the best.
- But that can get really tedious with complicated functions.

Estimation can be framed as an optimization problem

In many forms of estimation, we set up the problem as follows:

$$\{\hat{\beta}_0, \hat{\beta}_1\} = \arg\min_{\beta_0, \beta_1} J(\beta_0, \beta_1)$$

...where βs are the parameters we wish to identify.

In this course, we'll be looking at a broad variety of ways to define the *cost function*, J.

Linear regression as optimization

In “least squares” linear regression, the starting point for estimation is

$$\{\hat{\beta}_0, \hat{\beta}_1\} = \arg\min_{\beta_0, \beta_1} \sum_{i=1}^{n} (e_i)^2$$

Linear regression as optimization

In “least squares” linear regression, the starting point for estimation is

$$\begin{aligned}\{\hat{\beta}_0, \hat{\beta}_1\} &= \arg\min_{\beta_0,\beta_1} \sum_{i=1}^{n} (e_i)^2 \\ &= \arg\min_{\beta_0,\beta_1} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2\end{aligned}$$

Linear regression as optimization

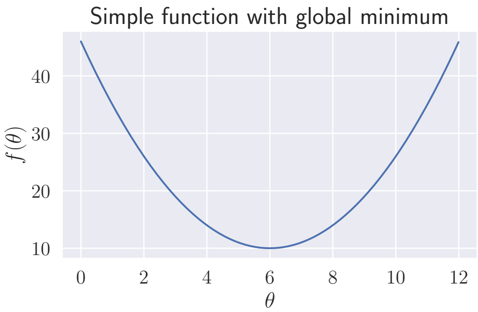
In “least squares” linear regression, the starting point for estimation is

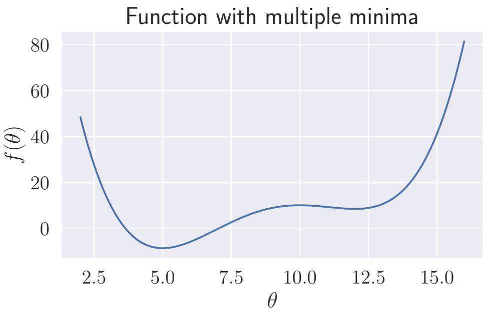
$$
\begin{aligned}
\{\hat{\beta}_0, \hat{\beta}_1\} &= \underset{\beta_0,\beta_1}{\arg\min} \sum_{i=1}^{n} (e_i)^2 \\
&= \underset{\beta_0,\beta_1}{\arg\min} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \\
&= \underset{\beta_0,\beta_1}{\arg\min} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2
\end{aligned}
$$

Why choose a quadratic (squared) objective function?

Why choose a quadratic (squared) objective function?

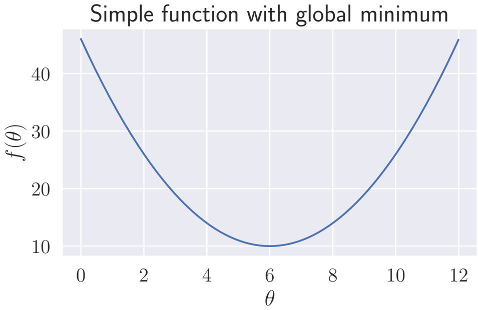
Hint:

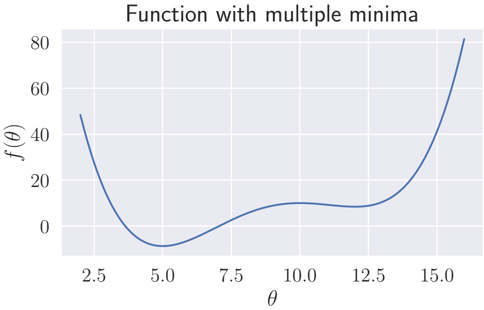




Why choose a quadratic (squared) objective function?

Hint:





With least squares, the cost function

- Has one global minimum
- Is differentiable – we can write an equation for $\frac{\partial f(\theta)}{\partial \theta} = 0$

Solving the estimation problem

$$\{\hat{\beta}_0, \hat{\beta}_1\} = \arg\min_{\beta_0,\beta_1} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2$$

So the optimal parameters must satisfy:

$$\frac{\partial \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2}{\partial \beta_0} = 0$$

$$\frac{\partial \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2}{\partial \beta_1} = 0$$

The solution:

$$\frac{\partial \sum_{i=1}^{n}(y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i))^2}{\partial \hat{\beta}_0} = 0 \qquad \Rightarrow \hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

$$\frac{\partial \sum_{i=1}^{n}(y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i))^2}{\partial \hat{\beta}_0} = 0 \qquad \Rightarrow \hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Before moving on, a little linear algebra:

Here are two vectors:

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$$
$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

Then the “dot” product of the two vectors is

$$\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2$$

Next, a little more linear algebra:

We can also multiply *matrices* and vectors. Matrices are like column vectors stacked side by side

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

Then matrix multiplication gives us

$$\mathbf{Ab} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} a_{11}b_1 + a_{12}b_2 \\ a_{21}b_1 + a_{22}b_2 \end{bmatrix}$$

Next, a little more linear algebra:

We can also multiply *matrices* and vectors. Matrices are like column vectors stacked side by side

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

Then matrix multiplication gives us

$$\mathbf{Ab} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} a_{11}b_1 + a_{12}b_2 \\ a_{21}b_1 + a_{22}b_2 \end{bmatrix}$$

Each element of the resulting matrix (or vector) is the dot product of a row of the first term (**A**) and a column of the second (**b**)

Therefore: the horizontal “dimension” of the first must be the same as the vertical “dimension” of the second.

Let's define matrices for our data:

Suppose we have n observations, (x_i, y_i). We'll arrange them all into a matrix form:

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

Note: when we start working with more than one independent variable, X will have a new column for each new variable.

And then a lot more linear algebra:

Let's define the 'transpose':

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \quad \Rightarrow \quad X^T = \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \end{bmatrix}$$

And then a lot more linear algebra:

Let's define the 'transpose':

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \quad \Rightarrow \quad X^T = \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \end{bmatrix}$$

Now a challenge question: what's the product of these two matrices:

$$X^T X = \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \end{bmatrix} \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}$$

Product of a matrix and its transpose

$$X^T X = \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \end{bmatrix} \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}$$

Product of a matrix and its transpose

$$X^T X = \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \end{bmatrix} \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}$$

$$= \begin{bmatrix} \text{1st row dot 1st col} & \text{1st row dot 2nd col} \\ \text{2nd row dot 1st col} & \text{2nd row dot 2nd col} \end{bmatrix}$$

Product of a matrix and its transpose

$$X^T X = \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \end{bmatrix} \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}$$

$$= \begin{bmatrix} \text{1st row dot 1st col} & \text{1st row dot 2nd col} \\ \text{2nd row dot 1st col} & \text{2nd row dot 2nd col} \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{i=1}^n 1 \cdot 1 & \sum_{i=1}^n 1 \cdot x_i \\ \sum_{i=1}^n 1 \cdot x_i & \sum_{i=1}^n x_i \cdot x_i \end{bmatrix}$$

Product of a matrix and its transpose

$$X^T X = \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \end{bmatrix} \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}$$

$$= \begin{bmatrix} \text{1st row dot 1st col} & \text{1st row dot 2nd col} \\ \text{2nd row dot 1st col} & \text{2nd row dot 2nd col} \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{i=1}^n 1 \cdot 1 & \sum_{i=1}^n 1 \cdot x_i \\ \sum_{i=1}^n 1 \cdot x_i & \sum_{i=1}^n x_i \cdot x_i \end{bmatrix} = \begin{bmatrix} n & n\bar{x} \\ n\bar{x} & \sum_{i=1}^n x_i^2 \end{bmatrix}$$

Doing linear algebra in numpy:

See the in-class workbook!

Finally, the “normal equations”

We showed a way to compute β coefficients individually a few slides ago.

Finally, the "normal equations"

We showed a way to compute β coefficients individually a few slides ago.

However that can get tedious if you're doing *multiple* linear regression – i.e. if you have more than one independent variable.

The so-called "normal equations" give a nice, compact form to get the parameters.

$$\Theta = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} = (X^T X)^{-1} X^T Y$$

$$= \left(\begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \end{bmatrix} \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

A note for computing and linear algebra geeks

The normal equations are an efficient way to solve the least squares linear regression problem *when the number of independent variables is relatively small.*

But! Inverting a matrix (the $(\cdot)^{-1}$ part) is a heavy computational lift – especially as the size of the matrix gets big.

Later in the semester we'll talk about an alternative approach, called "gradient descent",

- It searches for the optimal point on the cost function in a more manual way.
- But it's actually faster than getting the solution using the normal equations.