COM SCI M148, Lecture 3:
Linear Regression

Prof. Nathan Szymanski
nszymanski@seas.ucla.edu
Department of Materials Science

UCLA

Samueli
School of Engineering

Recap: The data science pipeline

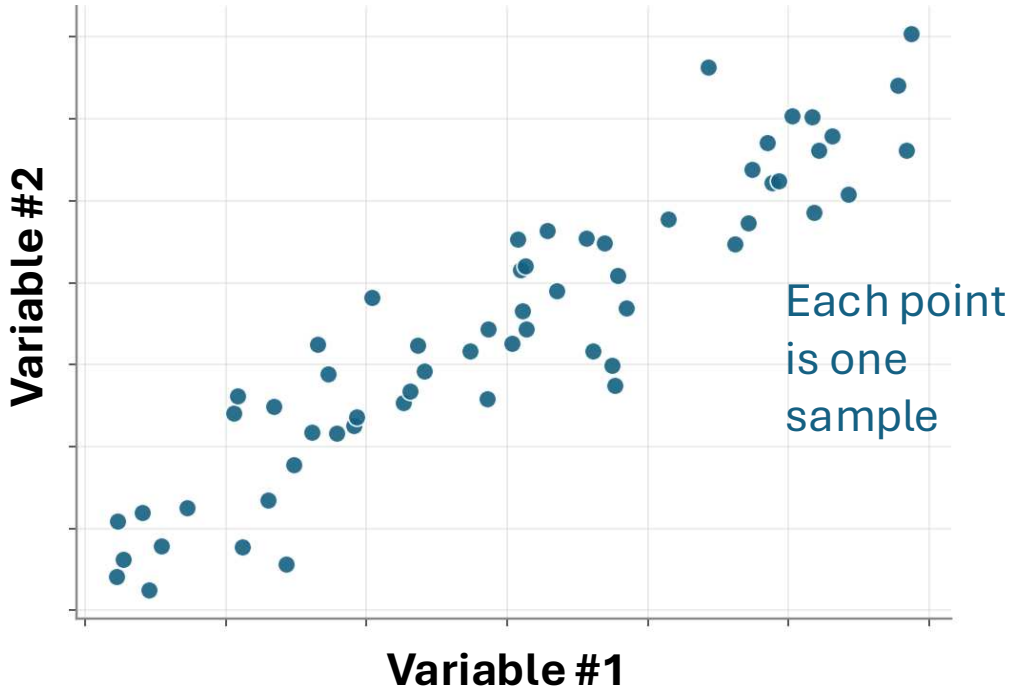
Last time we discussed how to collect and clean the data

Get the Data

Clean the Data

Model the Data

Interpret the Results



Recap: The data science pipeline

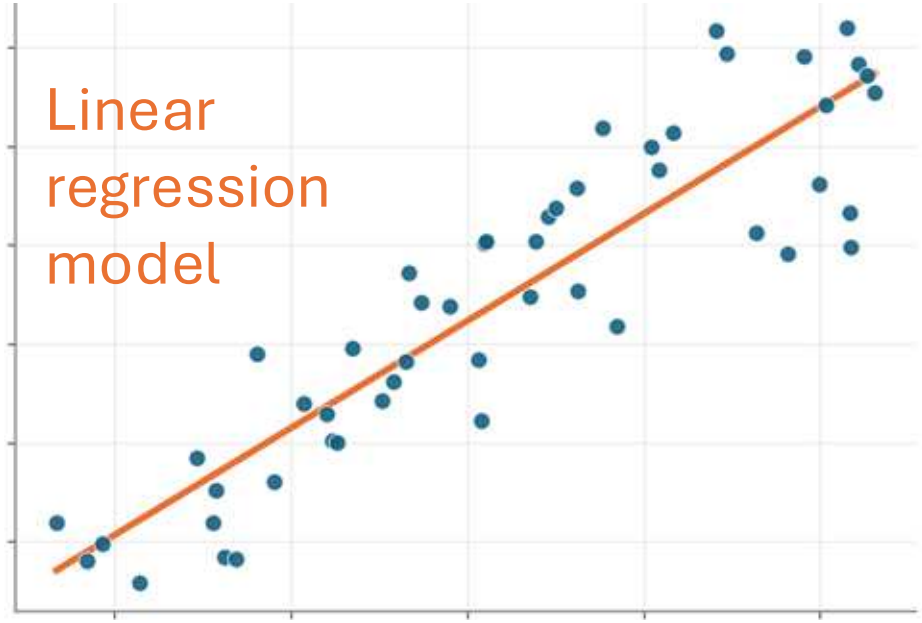
Ask an interesting question

Get the Data

Clean the Data

Model the Data

Interpret the Results



Today…We will build our first **model!**

Why build a model? Two reasons…

To make predictions



Predict **sales revenue** from **advertising spend**



Predict **travel time** from **traffic conditions**



Predict **reaction yield** from **temperature**

To understand relationships

Does **studying or sleep** improve **exam scores**?



How does **exercise** affect **longevity**?

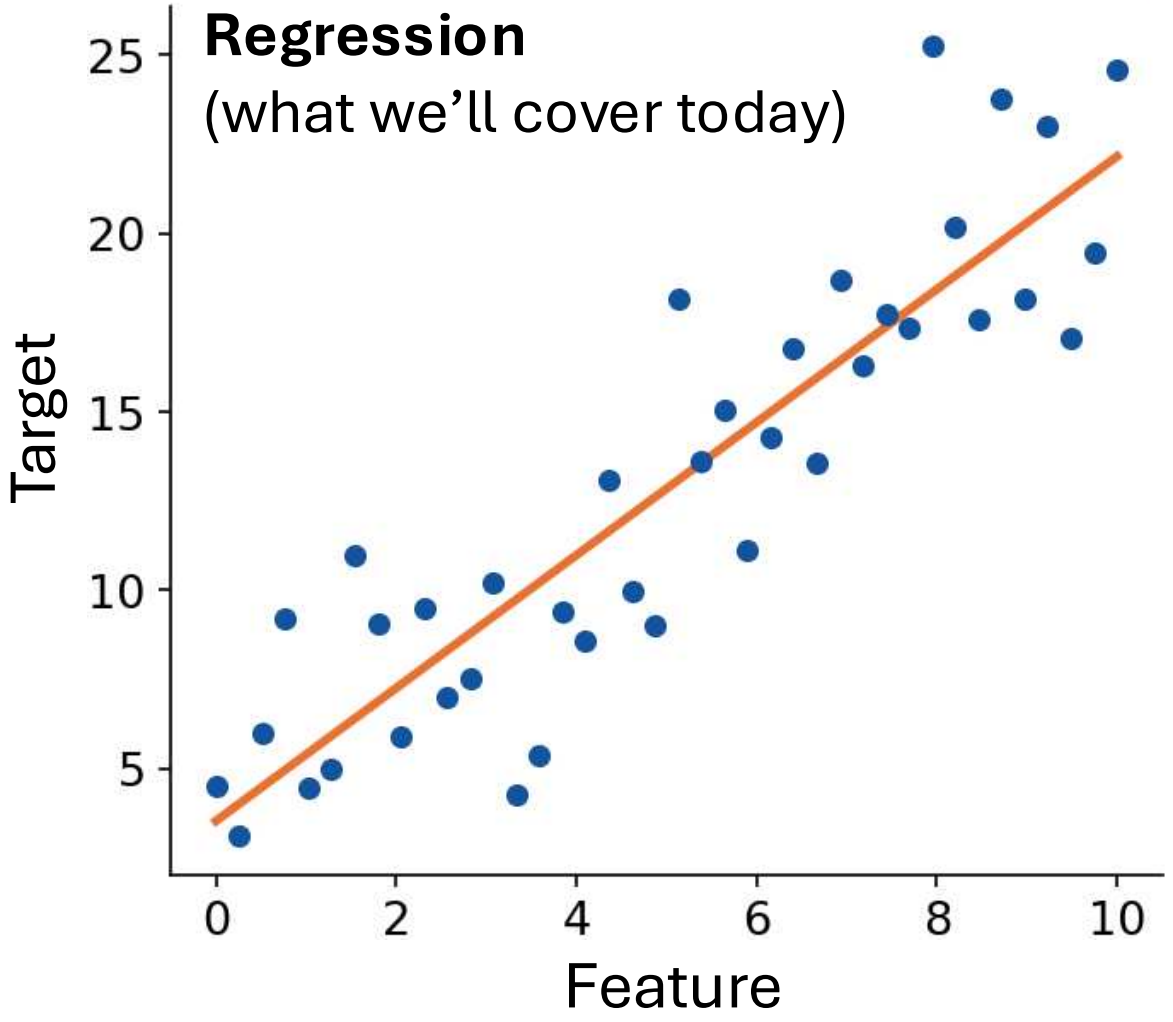


Do **CO_2 emissions** affect global **temperature**?

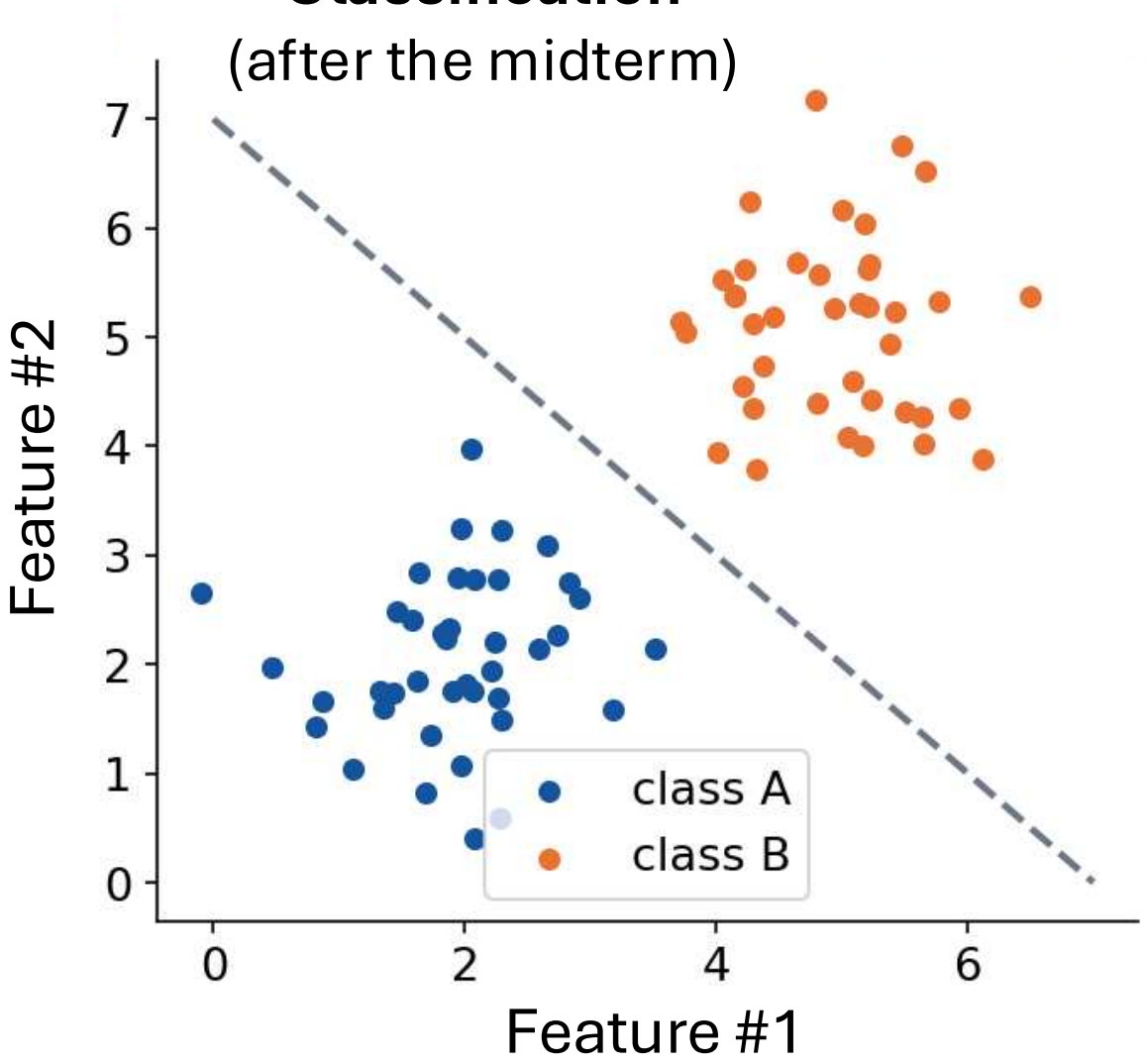


Recall: Two model types

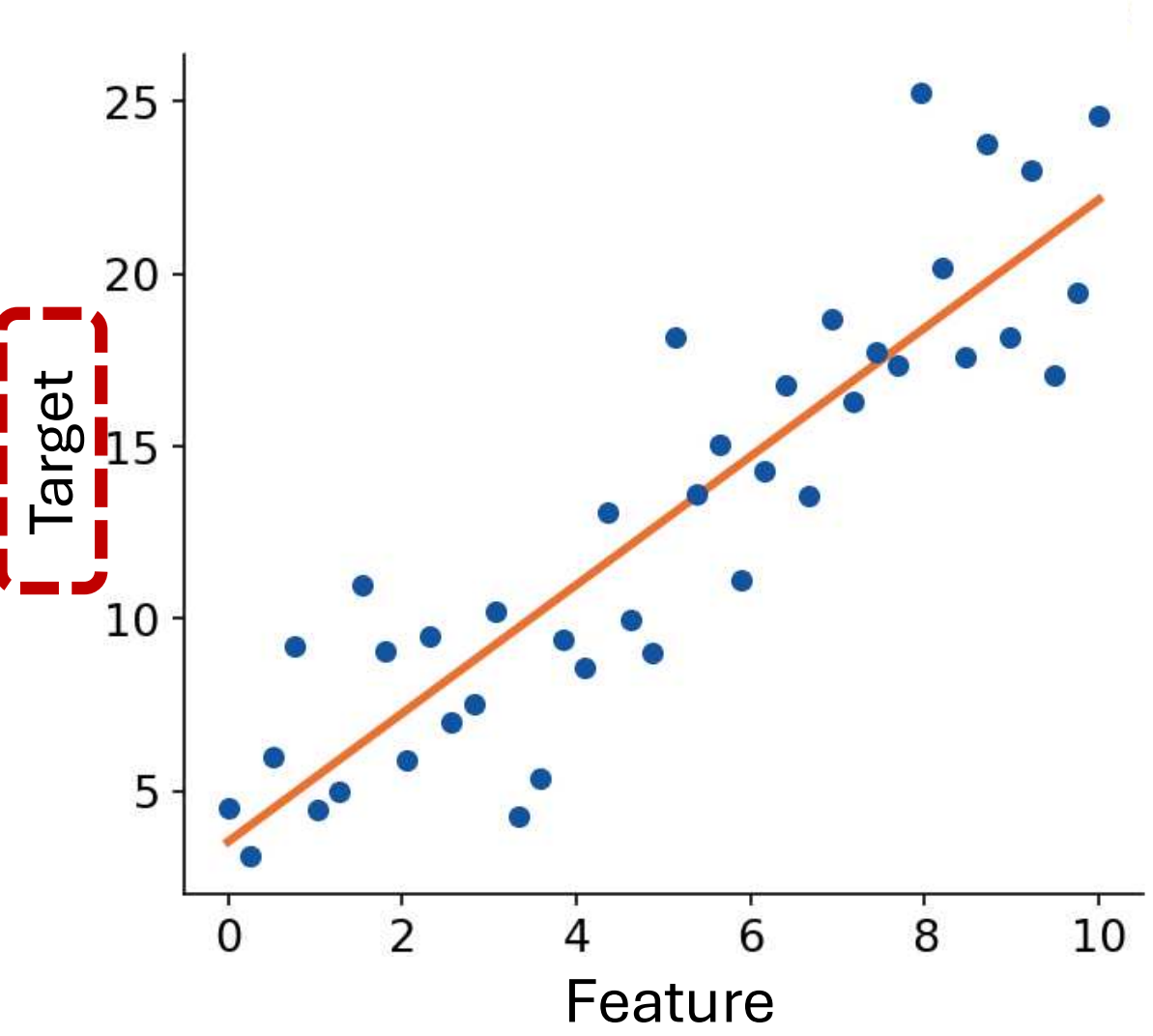
Regression
(what we'll cover today)



Classification
(after the midterm)



Regression as a *prediction* model

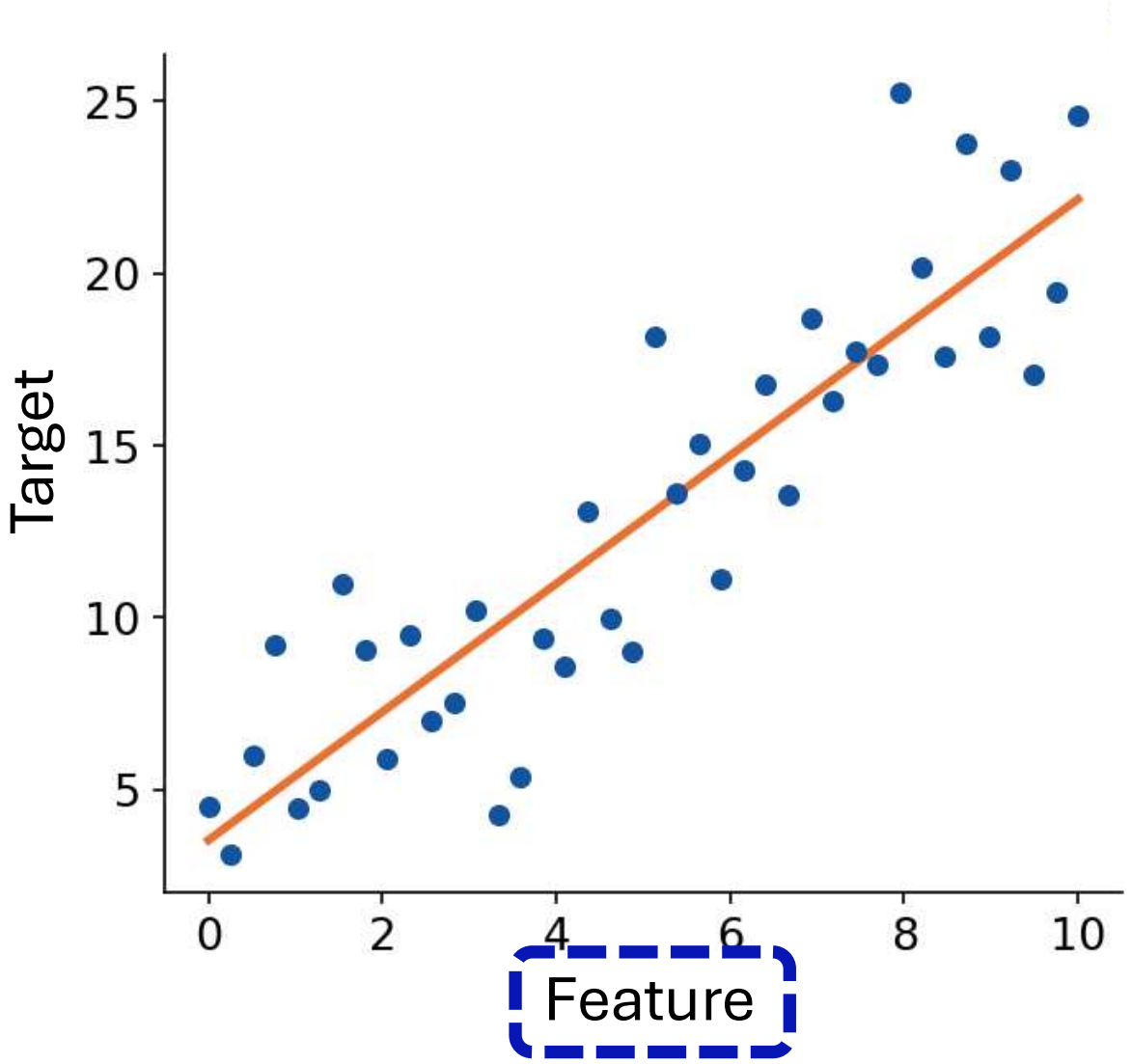


Let's define two categories of variables:

- **The response (y)** is a variable we'd like to predict or understand how it's influenced by another variable

  Analogous terms for y:
  Dependent variable,
  target, output

Regression as a *prediction* model

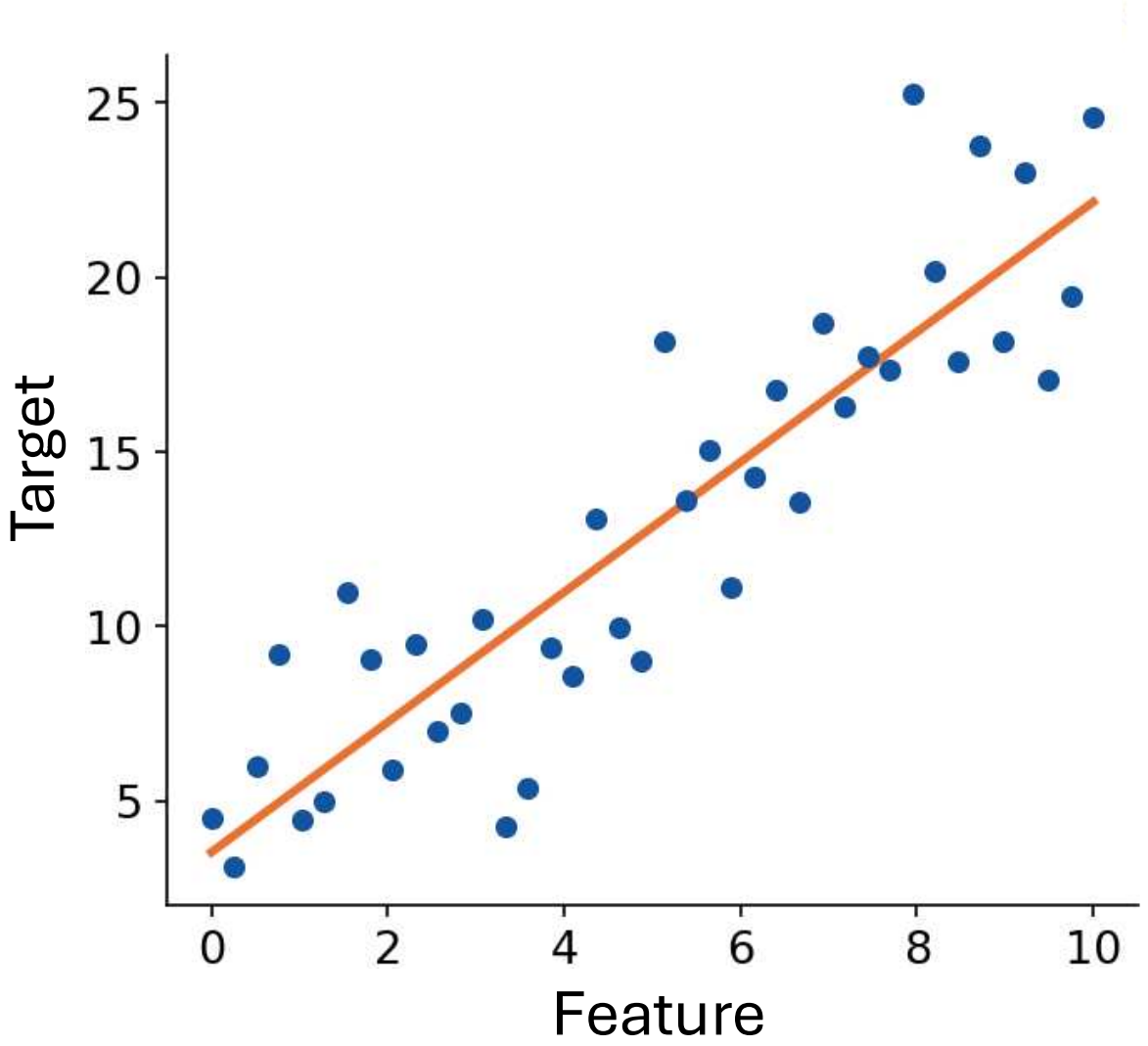


Let's define two categories of variables:

- **The predictor (x)** is a variable we use to make our prediction or affect the target

Analogous terms for x:
Independent variable, feature, input, covariate

Regression as a *prediction* model



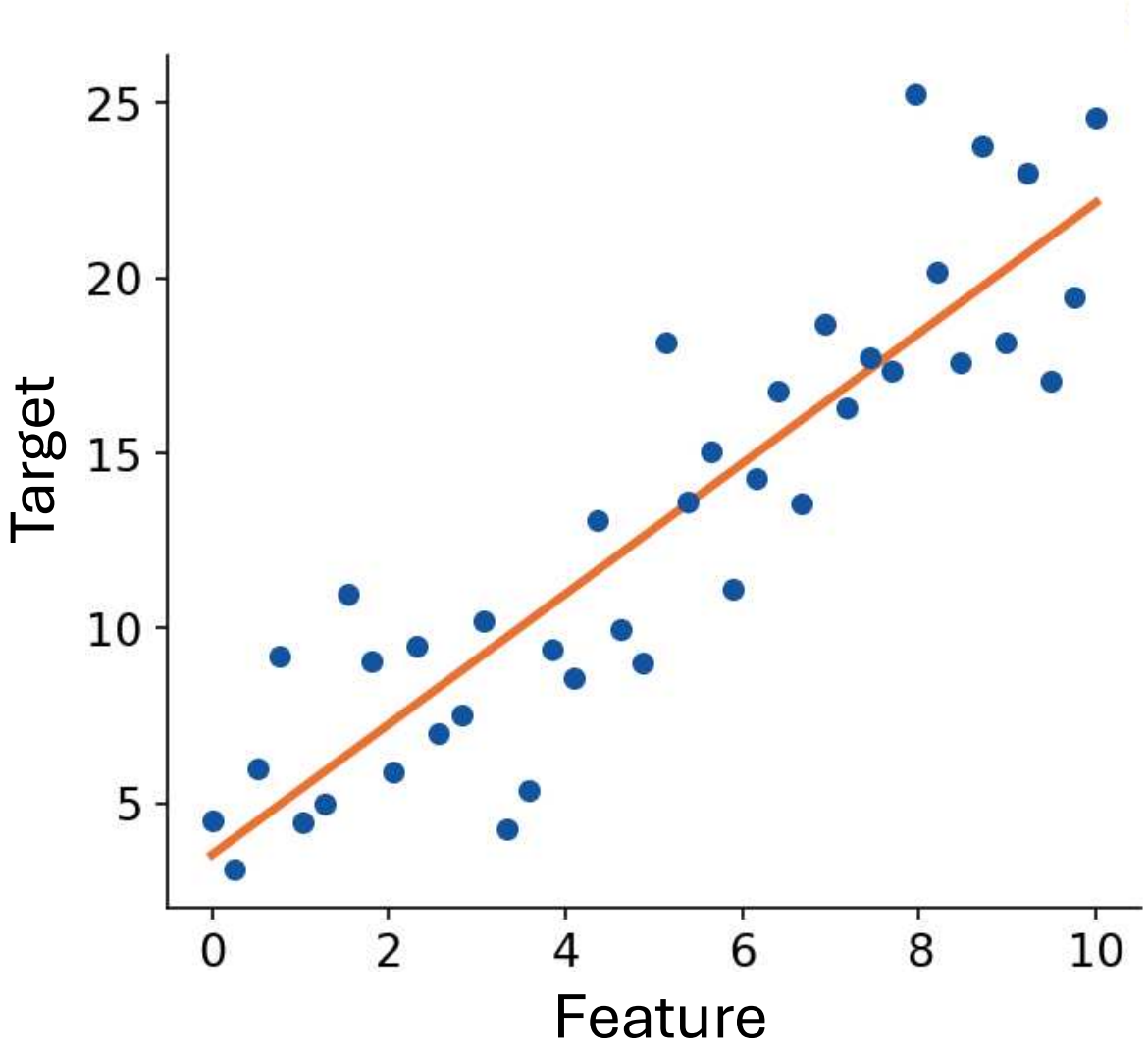
$$y = f(x) + \epsilon$$

The response is some unknown ***function*** of the predictor, with a margin of error ϵ

This is noise we cannot "beat," even with a perfect model

(measurement error, poor data collection, randomness, missing factors, *etc.*)

Regression as a *prediction* model



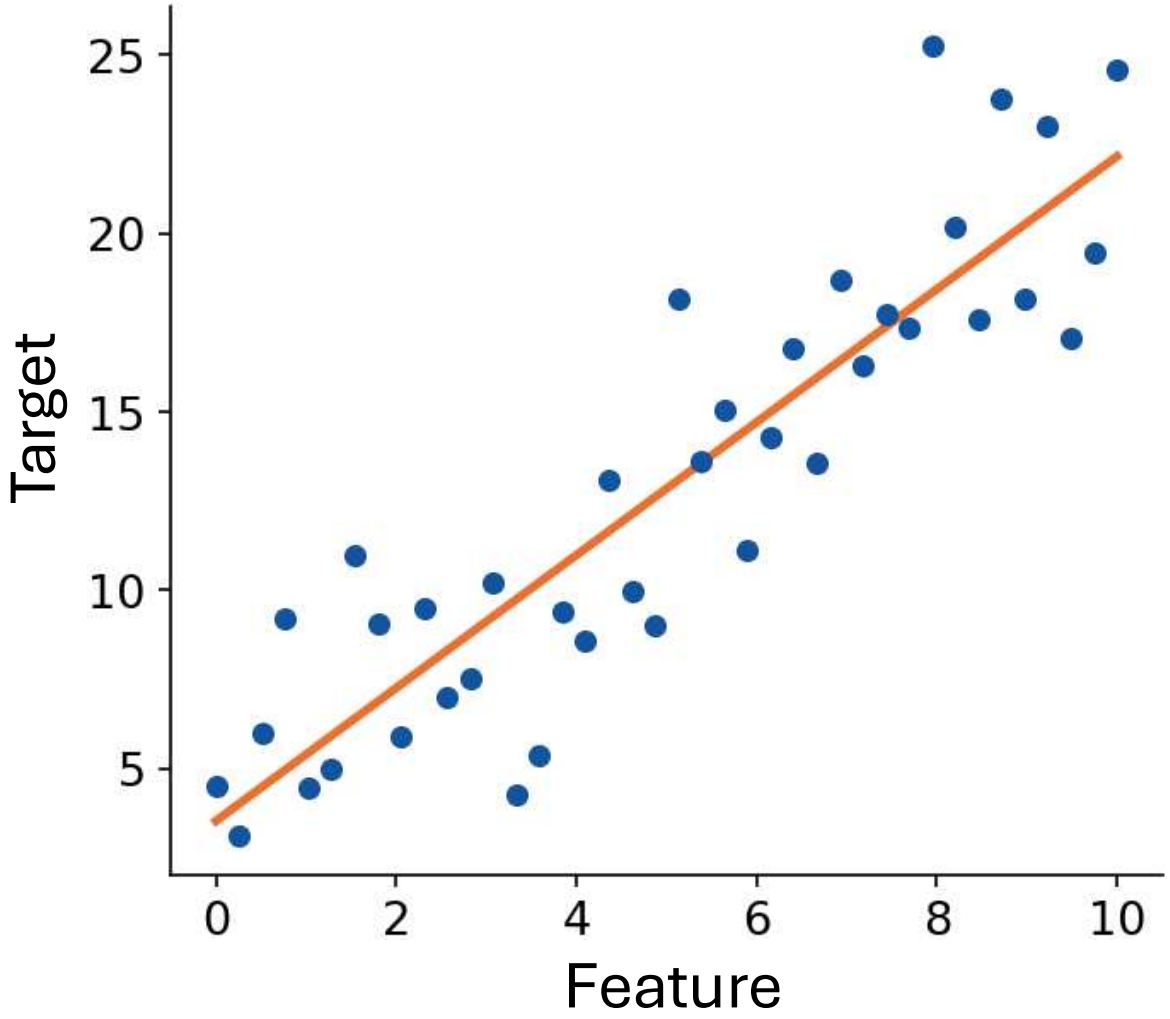
$$y = f(x) + \epsilon$$

The response is some unknown ***function*** of the predictor, with a margin of error ϵ

Our goal is to build **a model $\hat{f}$** that closely approximates f

Note: in this course, we'll often use the hat $\widehat{\square}$ to denote "estimated" or "predicted"

Regression as a *prediction* model



$$y = f(x) + \epsilon$$

Multiple variables

$$Y = f(X) + \epsilon$$

Upper case = vector

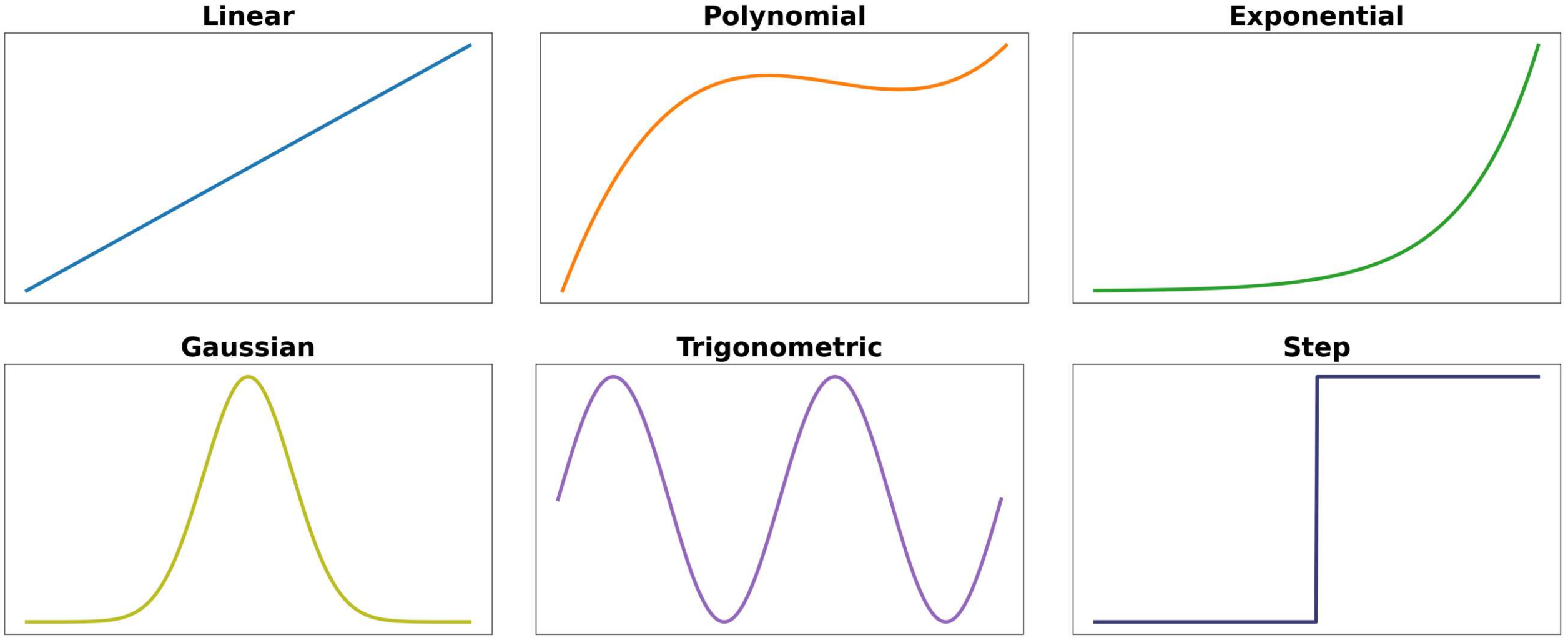
Lower case = scalar

$$Y = y_1, y_2, y_3, \dots y_n$$

$$X = x_1, x_2, x_3, \dots x_m$$

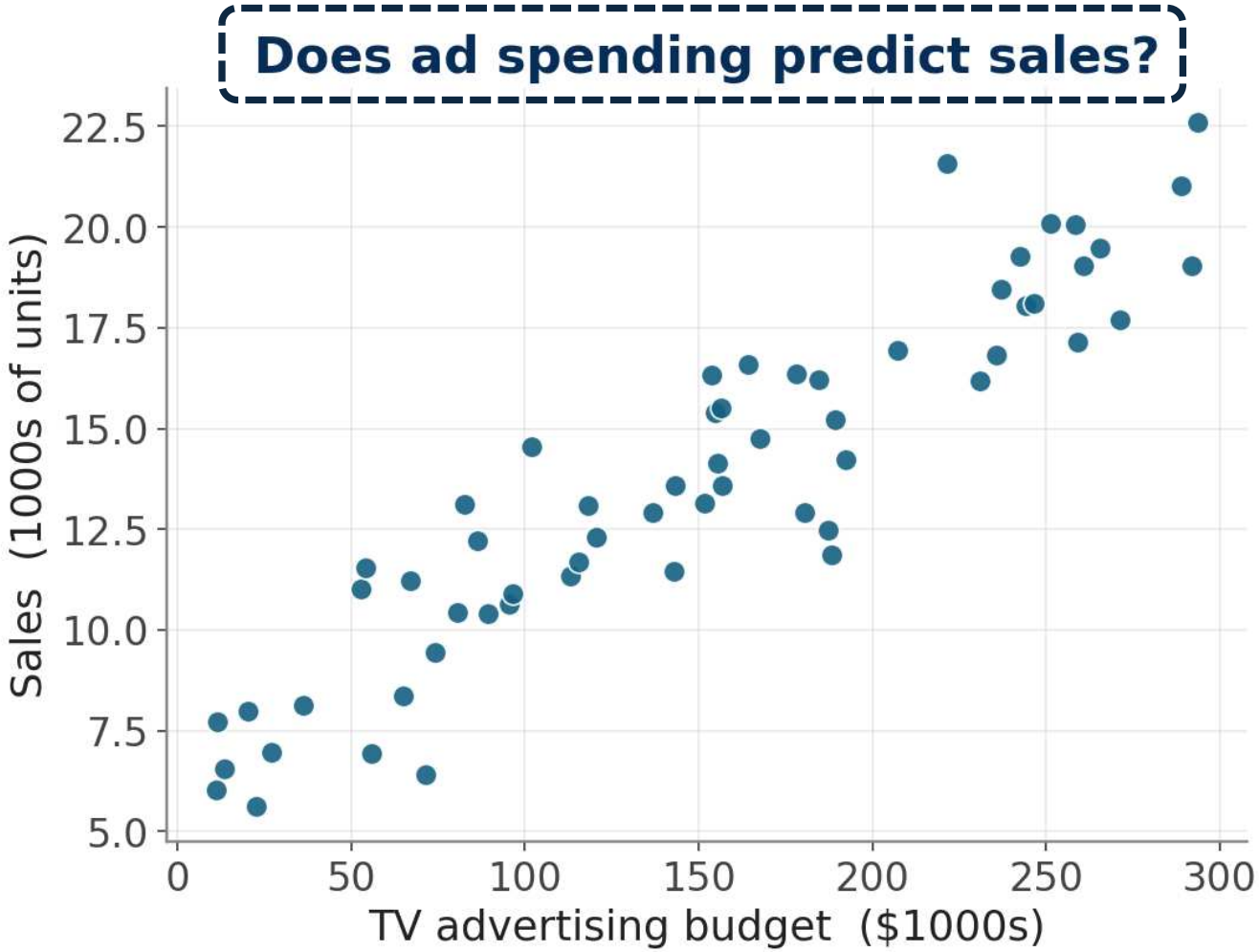
But for now, let's focus on single input/output…

Possible functions to choose for $\hat{y} = \hat{f}(x)$



Which one is best depends on our data – let's start simple…with a **linear model**

EX: data well-suited for linear modeling



Prediction: get $\hat{y}$ as close to y as possible for new inputs

"What will our sales be if we spend $150k on TV advertising?"

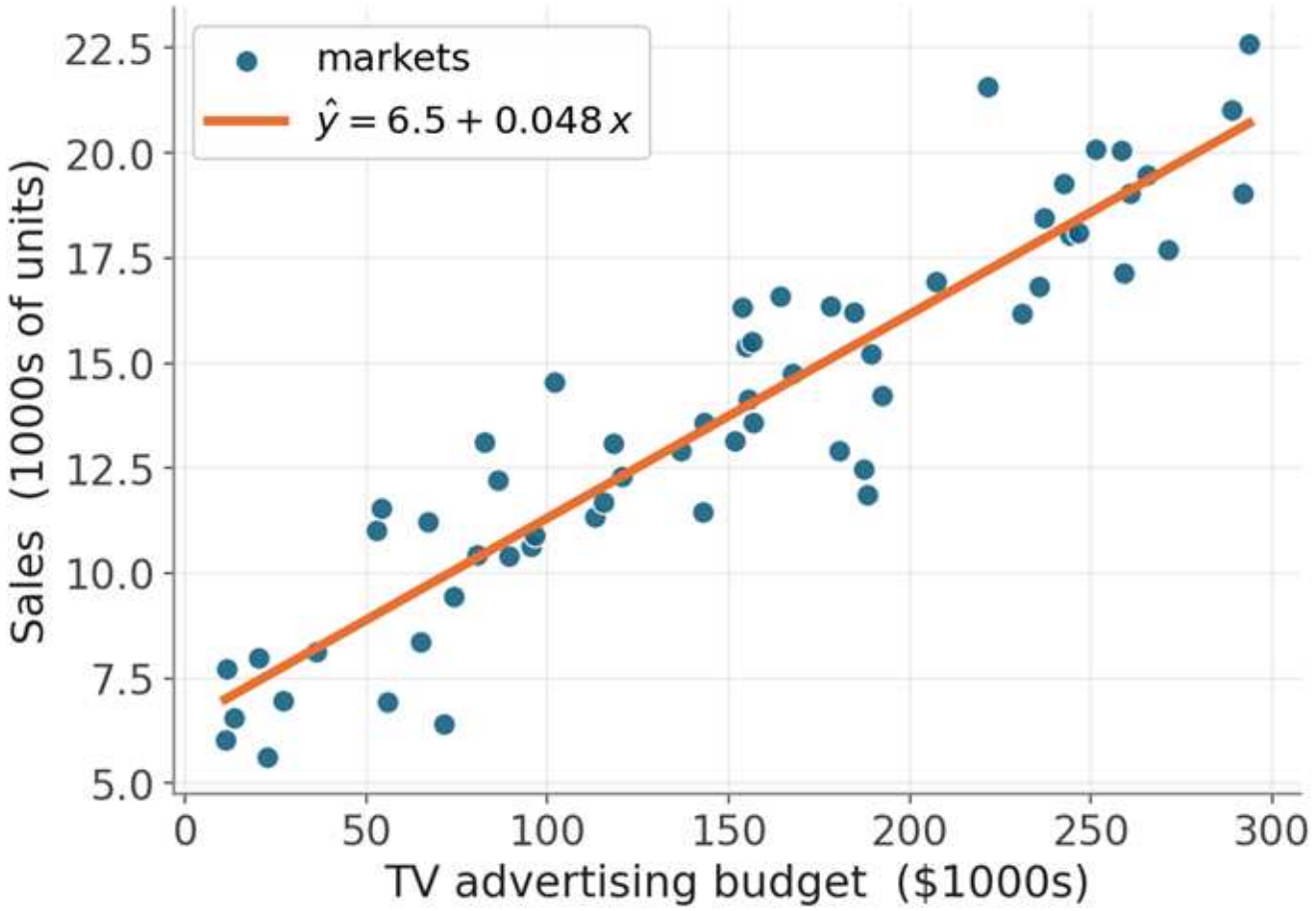
Understanding the relationship

"How much does each extra $1k of advertising spend move sales?"

Linear regression is often used because it does BOTH well

We'll use our linear model to *Predict* the future and *Explain* the present

Introducing: Linear regression…$y = mx + b$

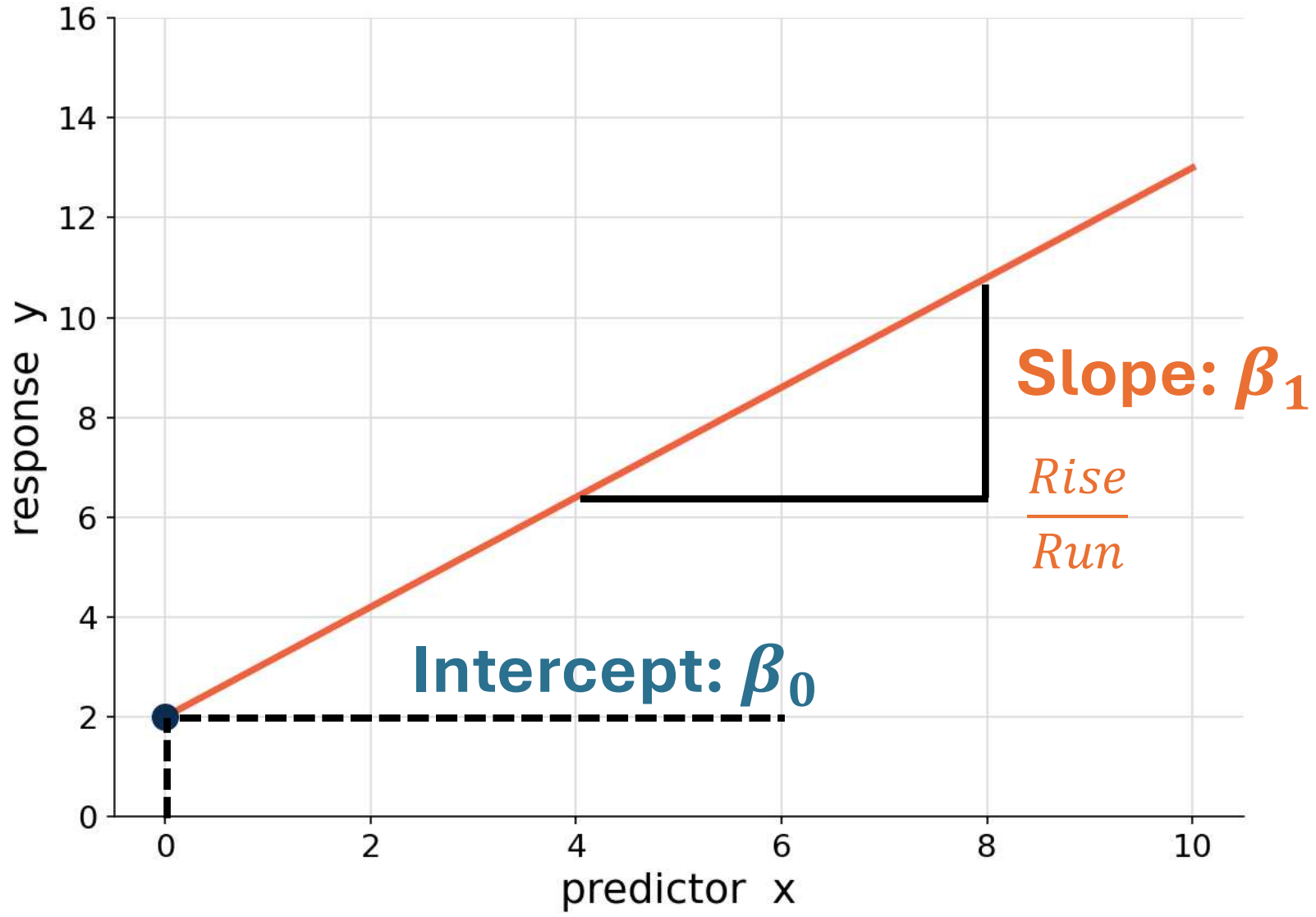


$$\hat{y} = \beta_0 + \beta_1 x$$

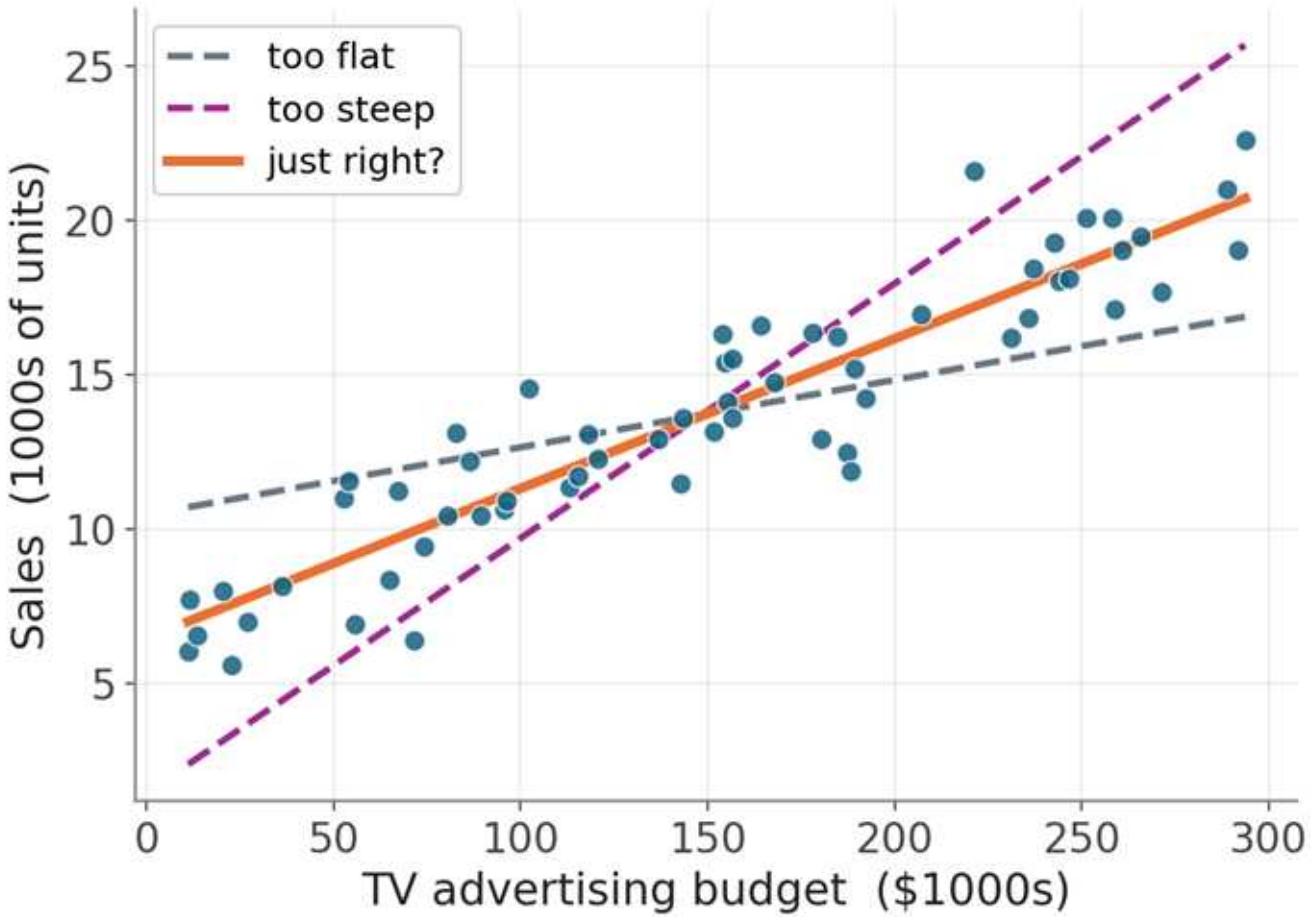
- β_0 **is the intercept:** the predicted value of y when $x = 0$
- β_1 **is the slope:** the change in y for a one-unit increase in x
- **"Fitting"** the model boils down to **choosing the best β_0 and β_1**

Two parameters define the entire line! This makes it (relatively) easy

Brief anatomy of a line



Finding *the best* line to model our data



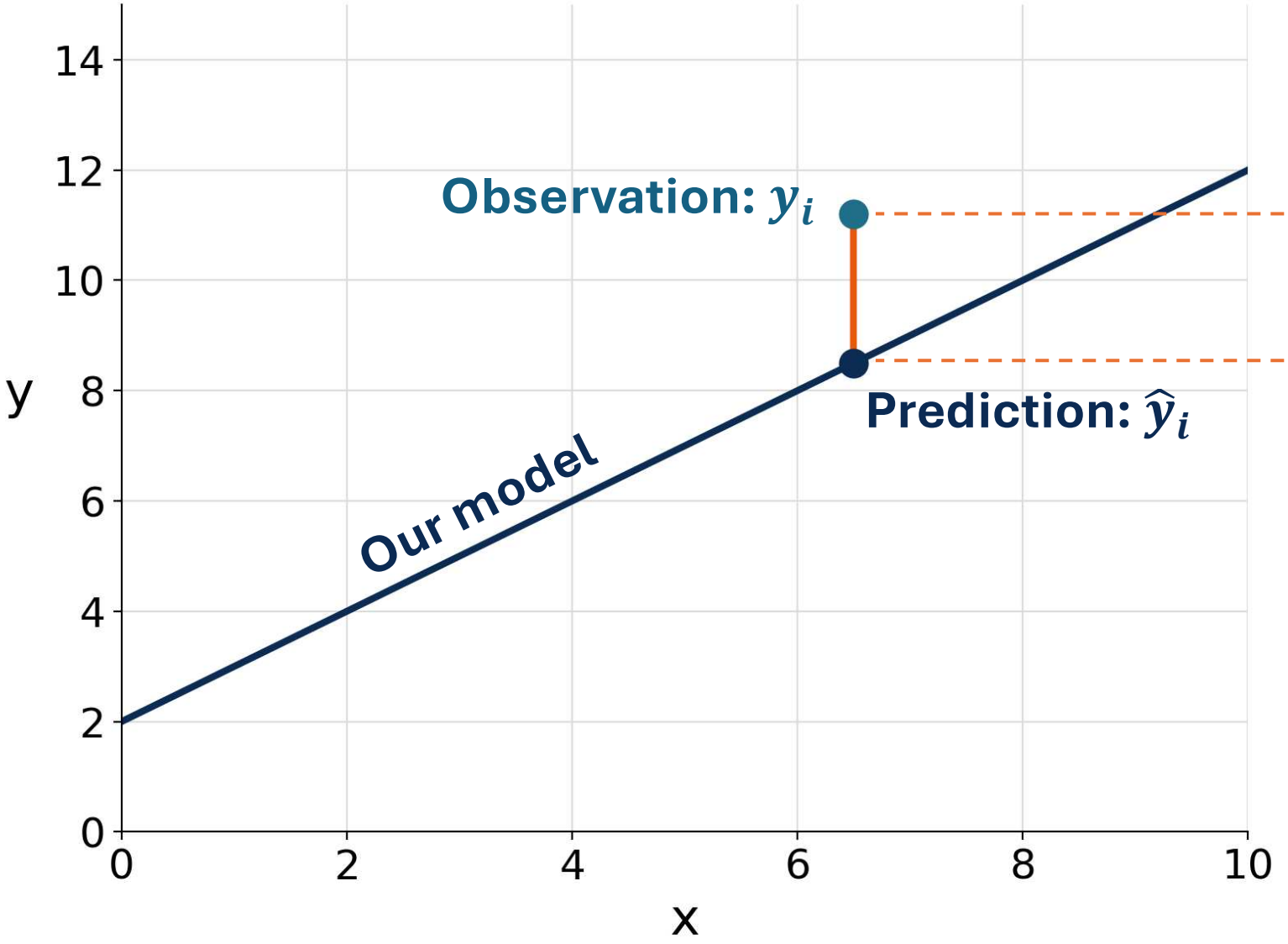
$$\hat{y} = \beta_0 + \beta_1 x$$

Which line is "the best?"
We need a criterion

- **"Fitting"** the model boils down to **choosing the best β_0 and β_1**

There are *infinitely* many lines (defined by β_0 and β_1) we could choose from!

The *residual* measures deviation from expectation



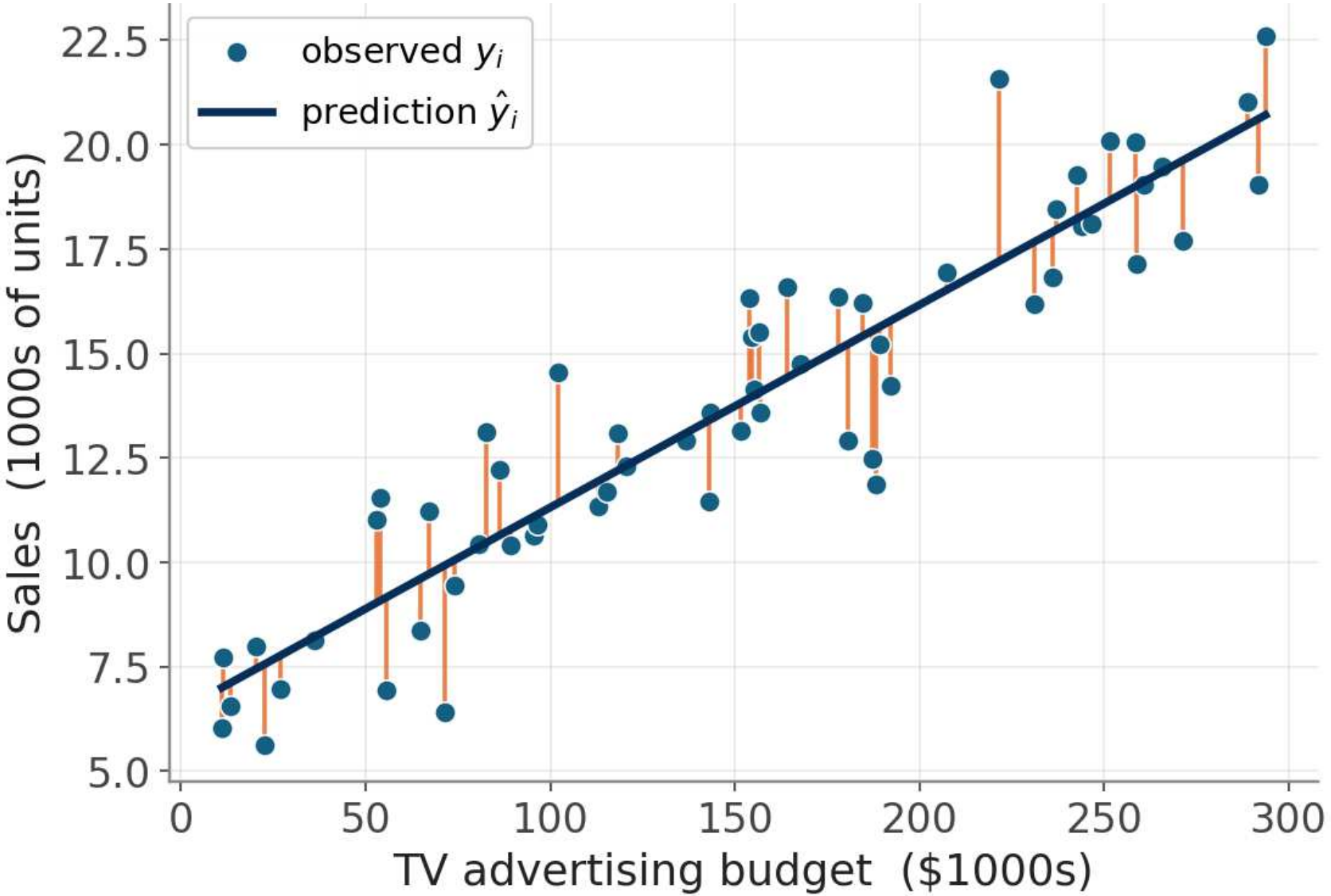
Residual:

$$e_i = y_i - \hat{y}_i$$

Observed – Predicted

A good model **minimizes e_i** for as many observations as possible

Residuals over an entire dataset



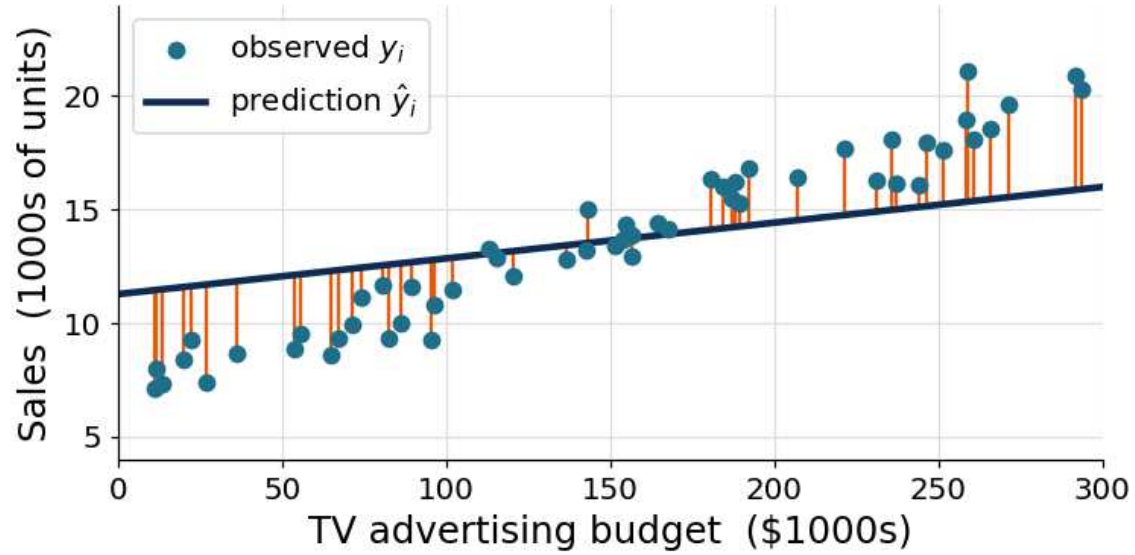
If we want these residuals to be small, on average, we should define a **loss** to be minimized:

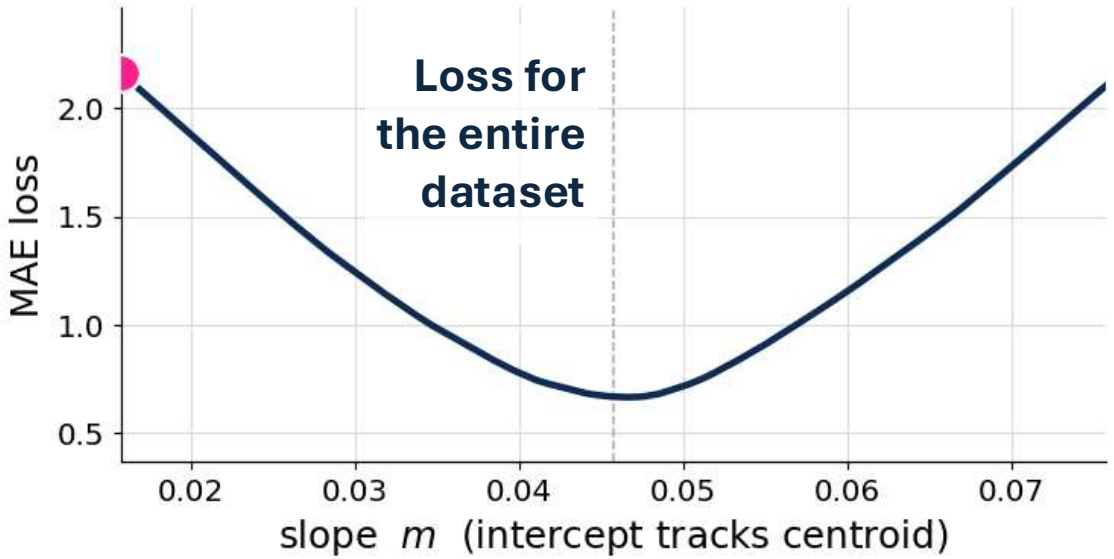
$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n} e_i$$

Any problem with this?

Positive and negative values of e_i cancel ☹

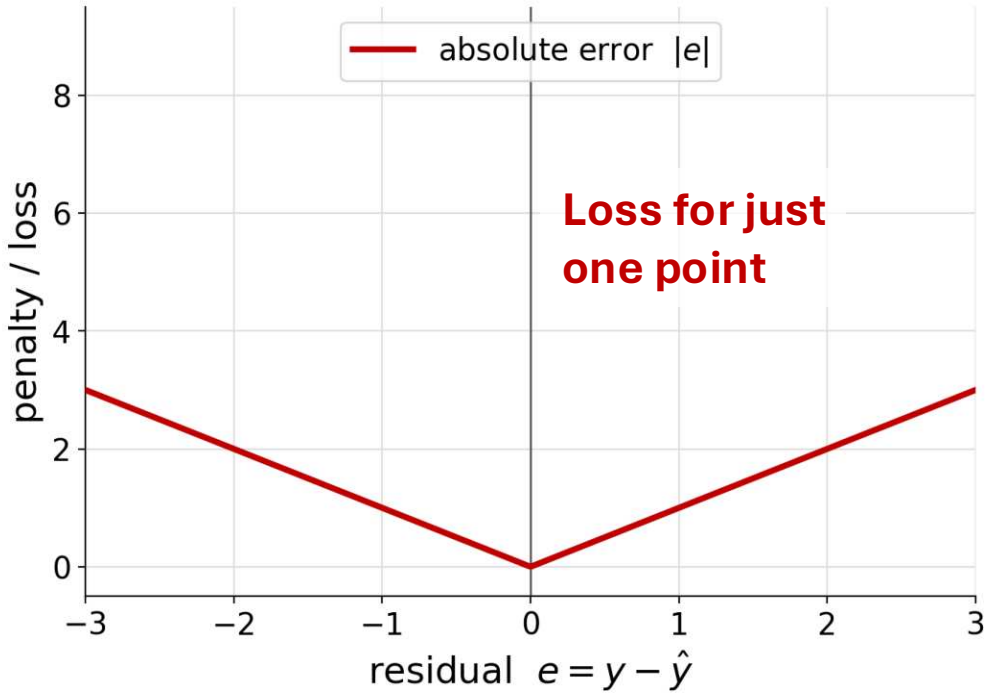
Solution: Mean Absolute Error (MAE)



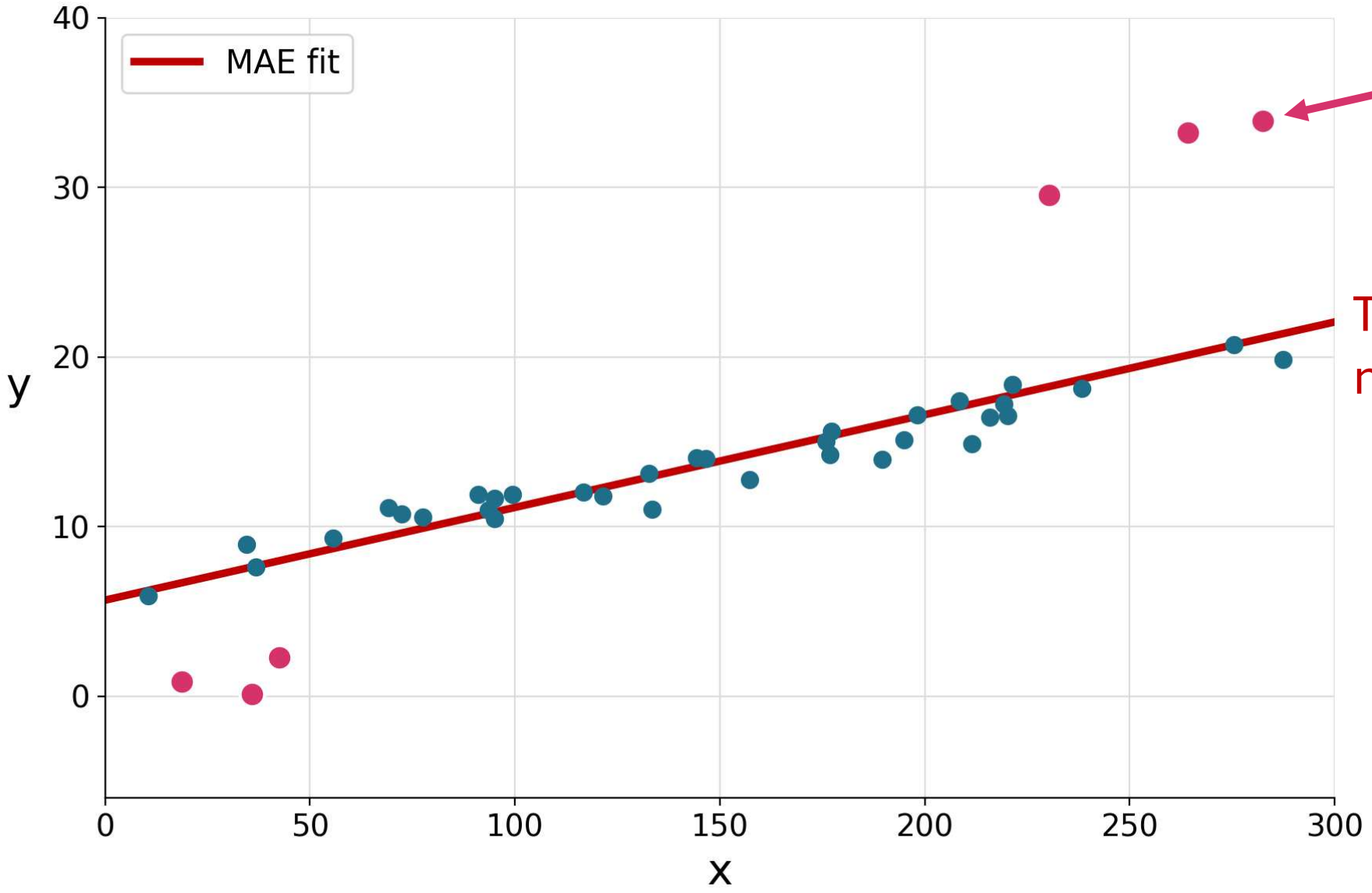


Mean *absolute* error:

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}|e_i|$$



MAE is (somewhat) insensitive to outliers

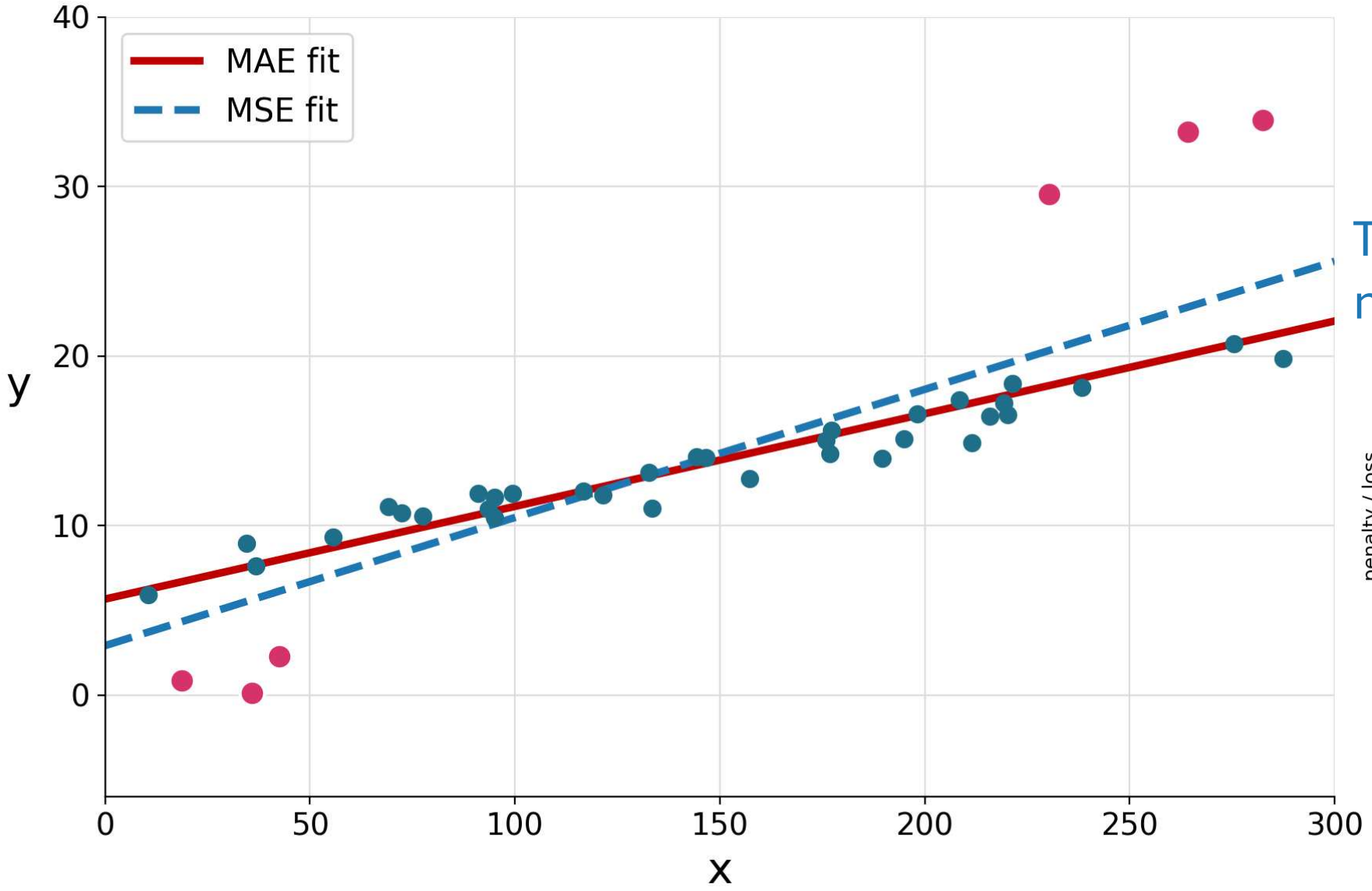


Outliers

This is the line that minimizes the MAE

Because there are so few outliers, they have a relatively small effect on the loss…This may be be good or bad!

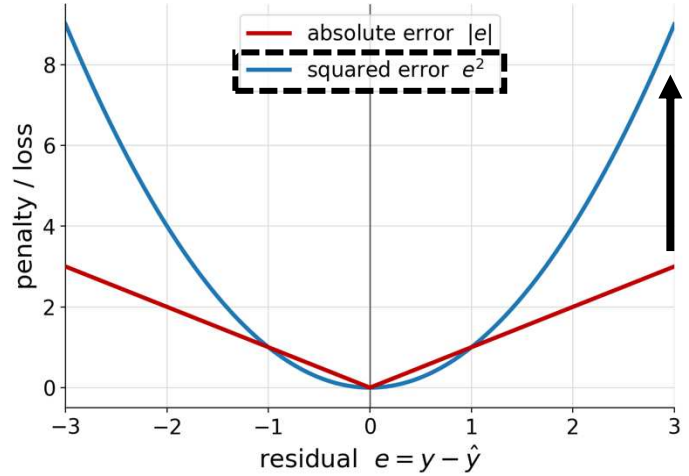
Another solution: Mean Squared Error (MSE)



Squared!

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

This is the line that minimizes the MSE



MSE is much more sensitive to outliers!

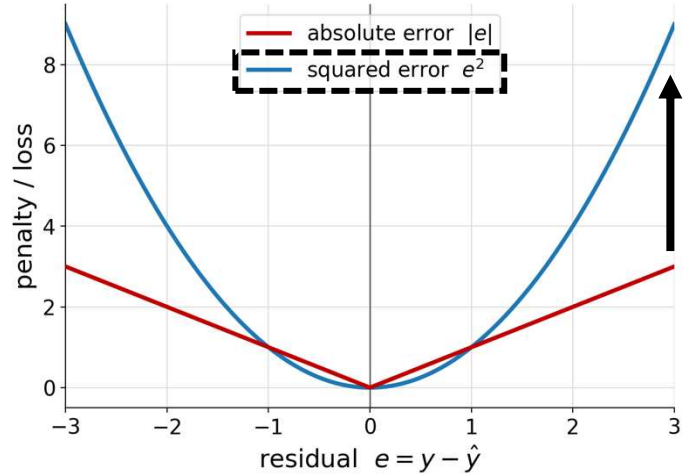
Root Mean Squared Error (RMSE)

- MSE is more commonly used than MAE
- One issue with MSE is it gives weird "squared" units that are hard to interpret (*e.g.*, what does $\$1^2$ mean?)
- Accordingly, people will often report the **root mean squared error (RMSE)** which has standard units – same as target

$$RSME = \sqrt{MSE}$$

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

This is the line that minimizes the MSE



MSE is much more sensitive to outliers!

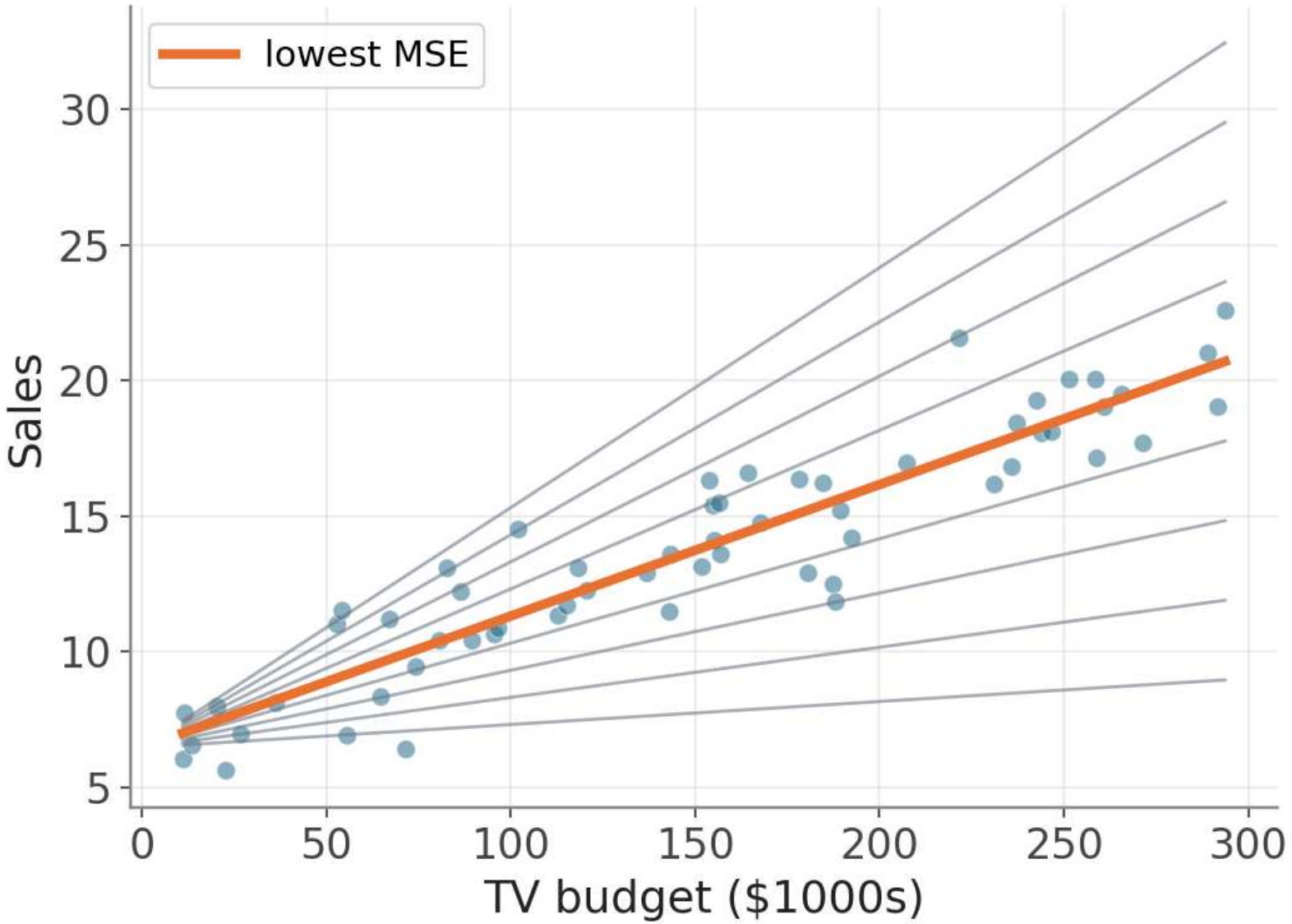
So…How do we find the model that minimizes MSE?

Fitting (or “Training”) the Model

*Finding β_0 and β_1 that **minimize the MAE or MSE loss***

$$\hat{y} = \beta_0 + \beta_1 x$$

Approach #1: Brute force search



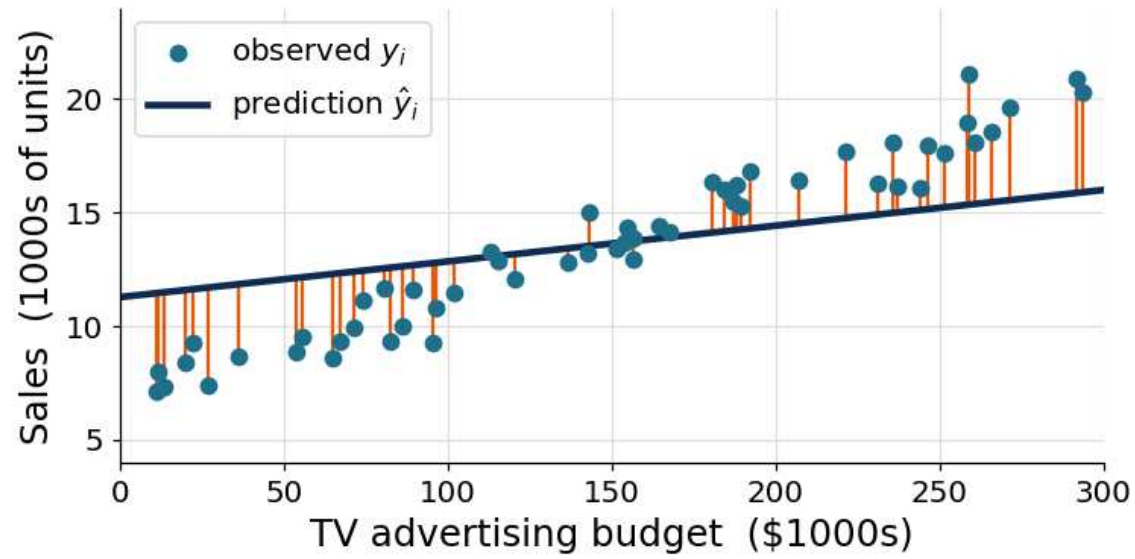
The simplest (but worst) approach is **brute force:**

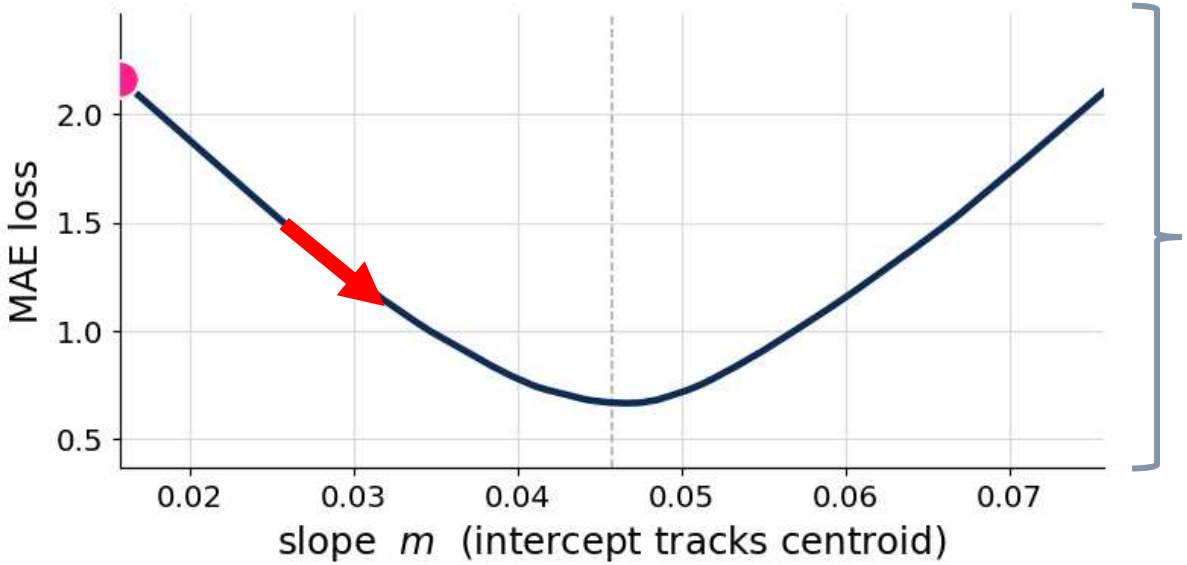
Try many different lines (different β_0, β_1) and see which one has the lowest MSE

This is hopeless at scale! Many features = many lines we could try…

Beware the curse of dimensionality 

Approach #2: Gradient descent (GD)





GD in one dimension (1D):

Easily computed for a line!

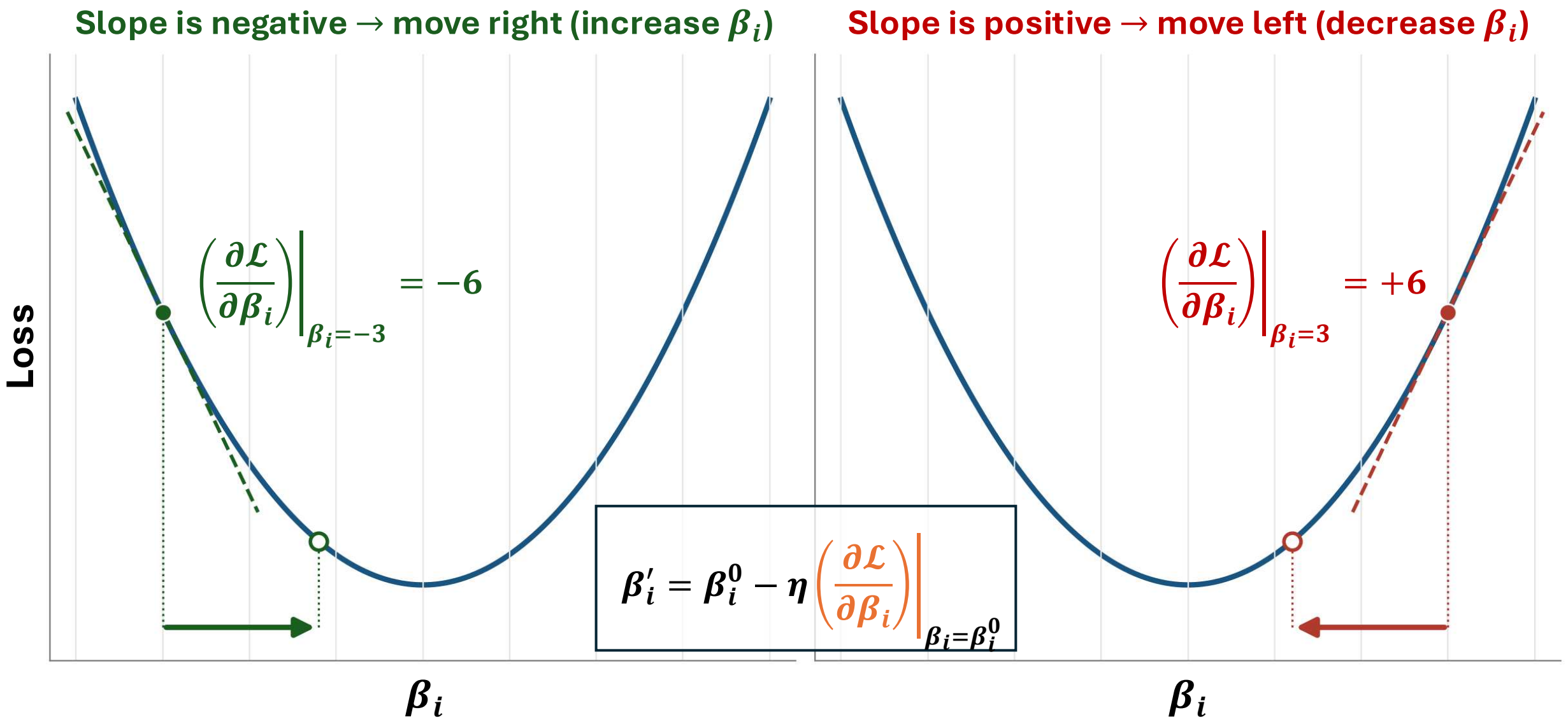
- Give a starting point ($\boldsymbol{\beta_i^0}$)
- Compute the derivative: $\boldsymbol{\partial \mathcal{L} / \partial \beta_i}$
- Update your parameters using:

$$\boldsymbol{\beta_i' = \beta_i^0 - \eta \left(\frac{\partial \mathcal{L}}{\partial \beta_i} \right) \Big|_{\beta_i = \beta_i^0}}$$

Learning rate

Notice this loss landscape is **convex**. We can simply find its minimum by **following the gradient downward** to find the minimum loss.

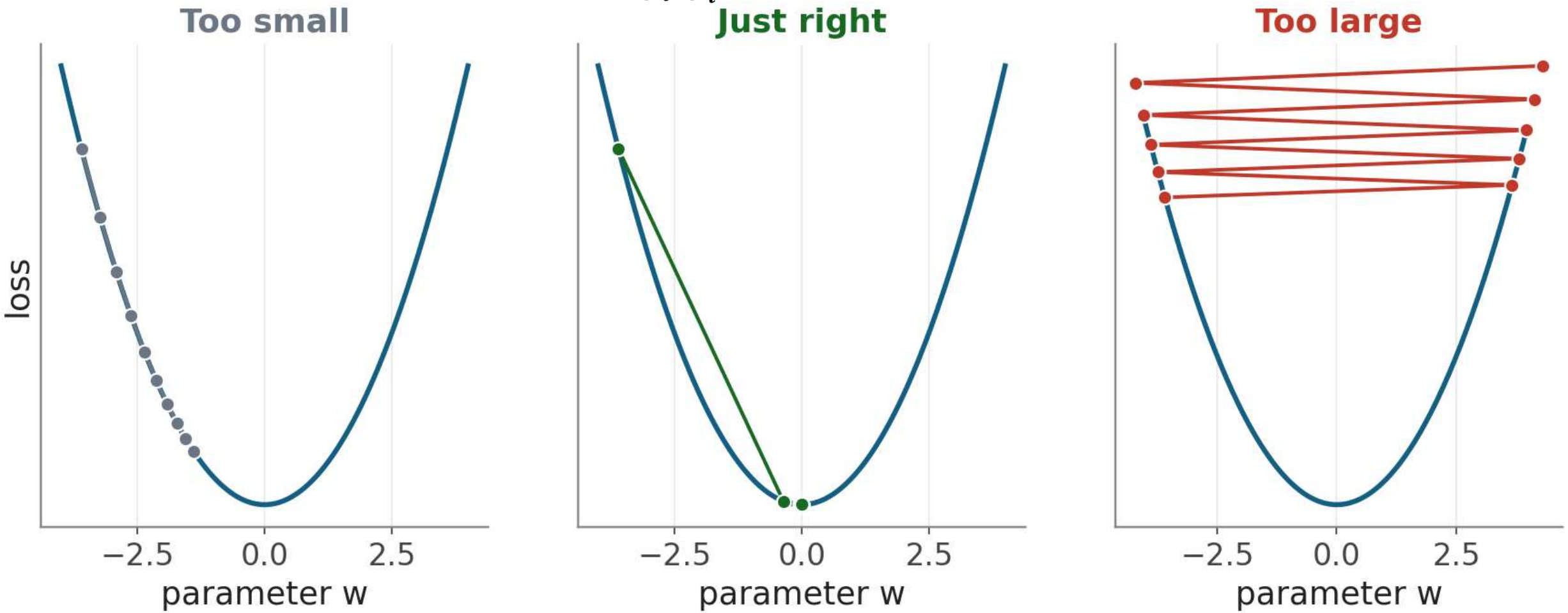
Illustrating gradient descent



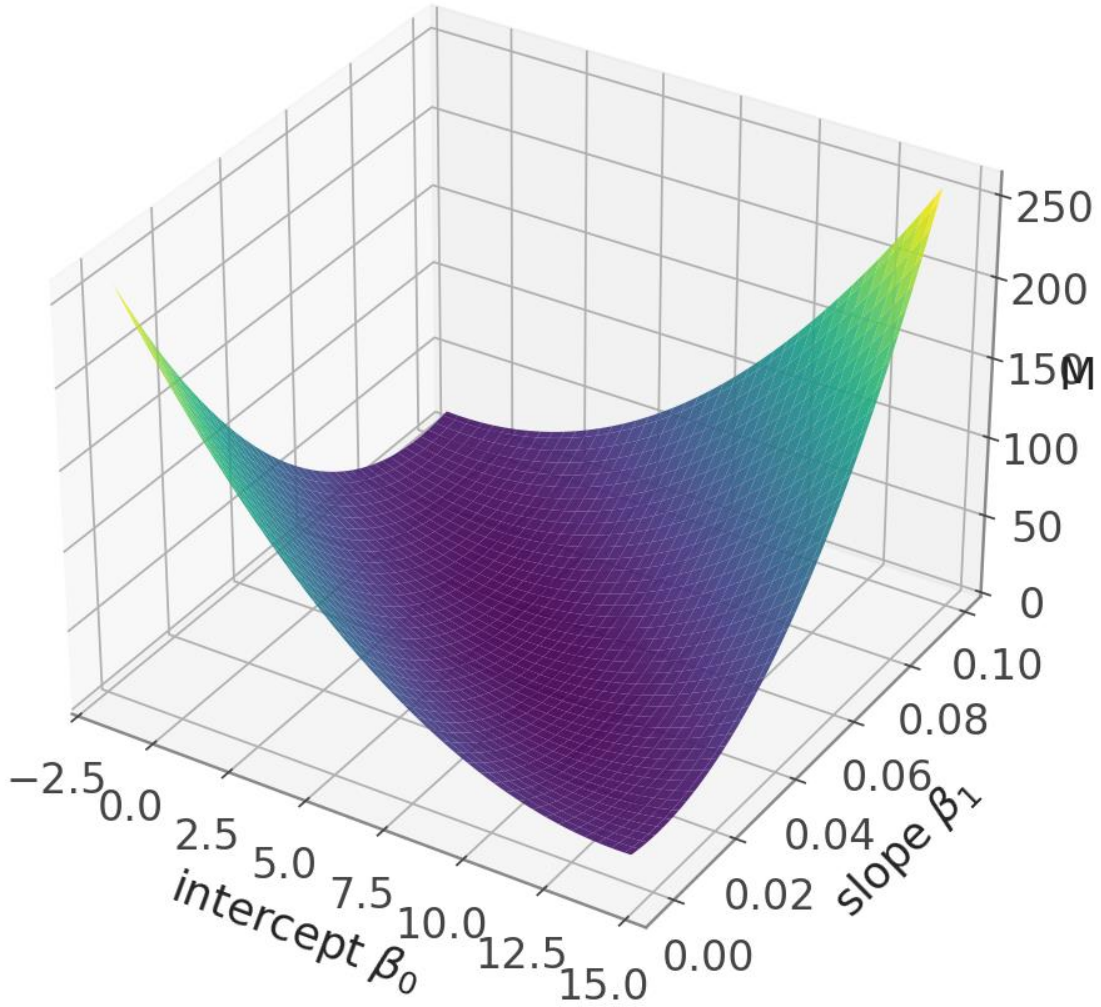
Learning rate controls the step size

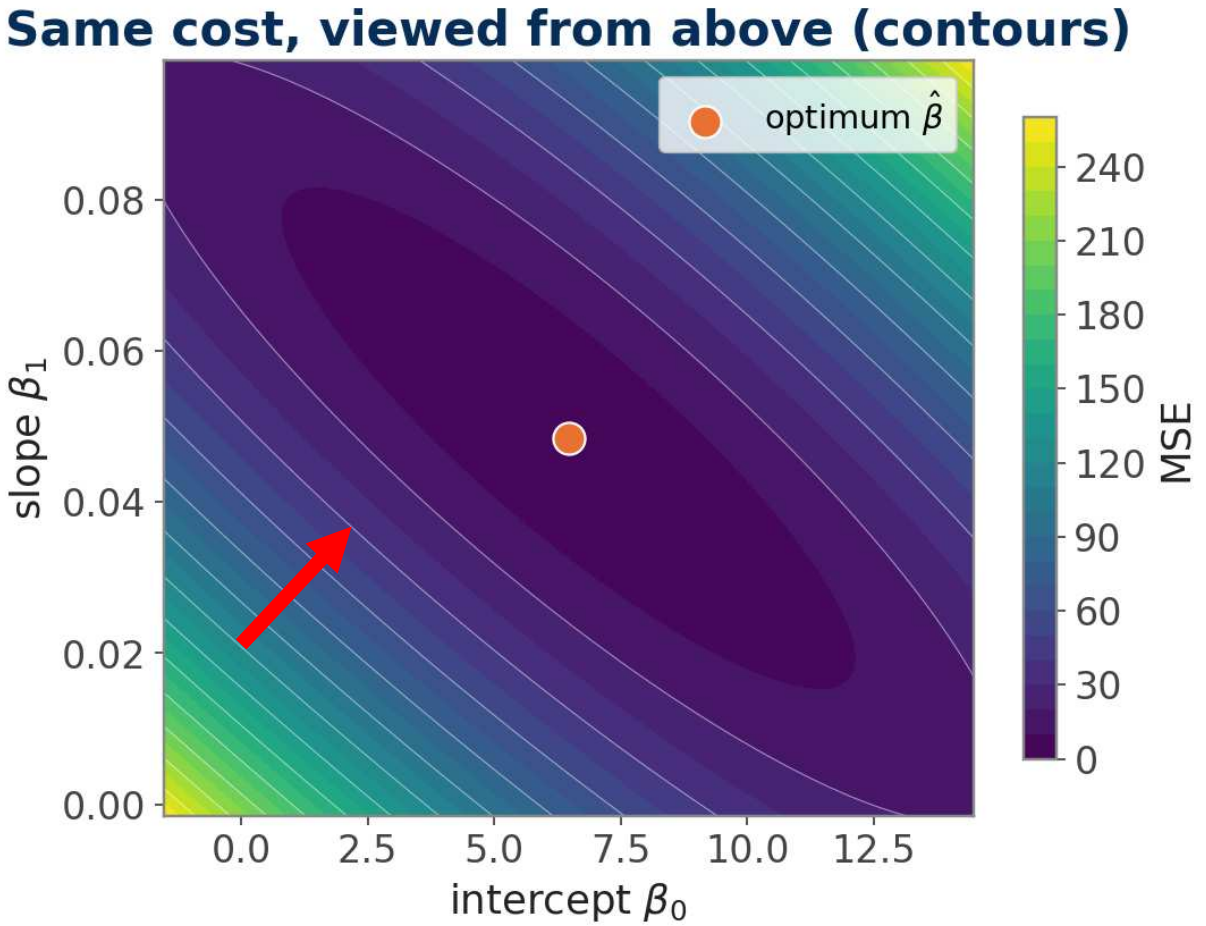
$$\beta_i' = \beta_i^0 - \eta \left(\frac{\partial \mathcal{L}}{\partial \beta_i} \right) \Bigg|_{\beta_i = \beta_i^0}$$

η is the learning rate



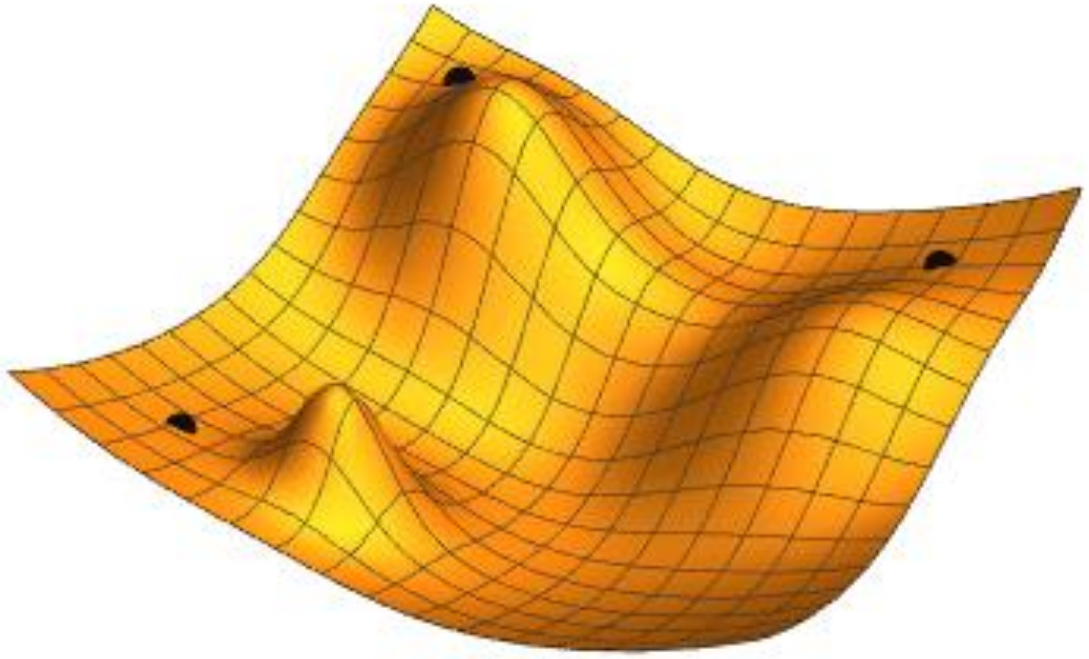
We have two parameters: β_0 and β_1





Fortunately, it is still convex in 2D, and we can therefore safely use gradient descent!

Convexity is important!



If the loss landscape was *not* convex:
Three runs starting from different points might settle in **different local minima**…
But there is only one *global* minimum.

Luckily **for linear regression**, we do *not* have this problem. Both the MAE and MSE losses are **always convex!**

If you'd like to practice at home, prove convexity by showing the Hessian is positive semidefinite for MSE:

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

$$\det\begin{pmatrix} \frac{\partial^2 \mathcal{L}}{\partial \beta_0^2} & \frac{\partial^2 \mathcal{L}}{\partial \beta_0 \partial \beta_1} \\ \frac{\partial^2 \mathcal{L}}{\partial \beta_0 \partial \beta_1} & \frac{\partial^2 \mathcal{L}}{\partial \beta_1^2} \end{pmatrix} > 0 \qquad \frac{\partial^2 \mathcal{L}}{\partial \beta_0^2} > 0$$

Approach #3: Solving the problem *analytically*

The minimum occurs at the point where the **gradient is zero:**

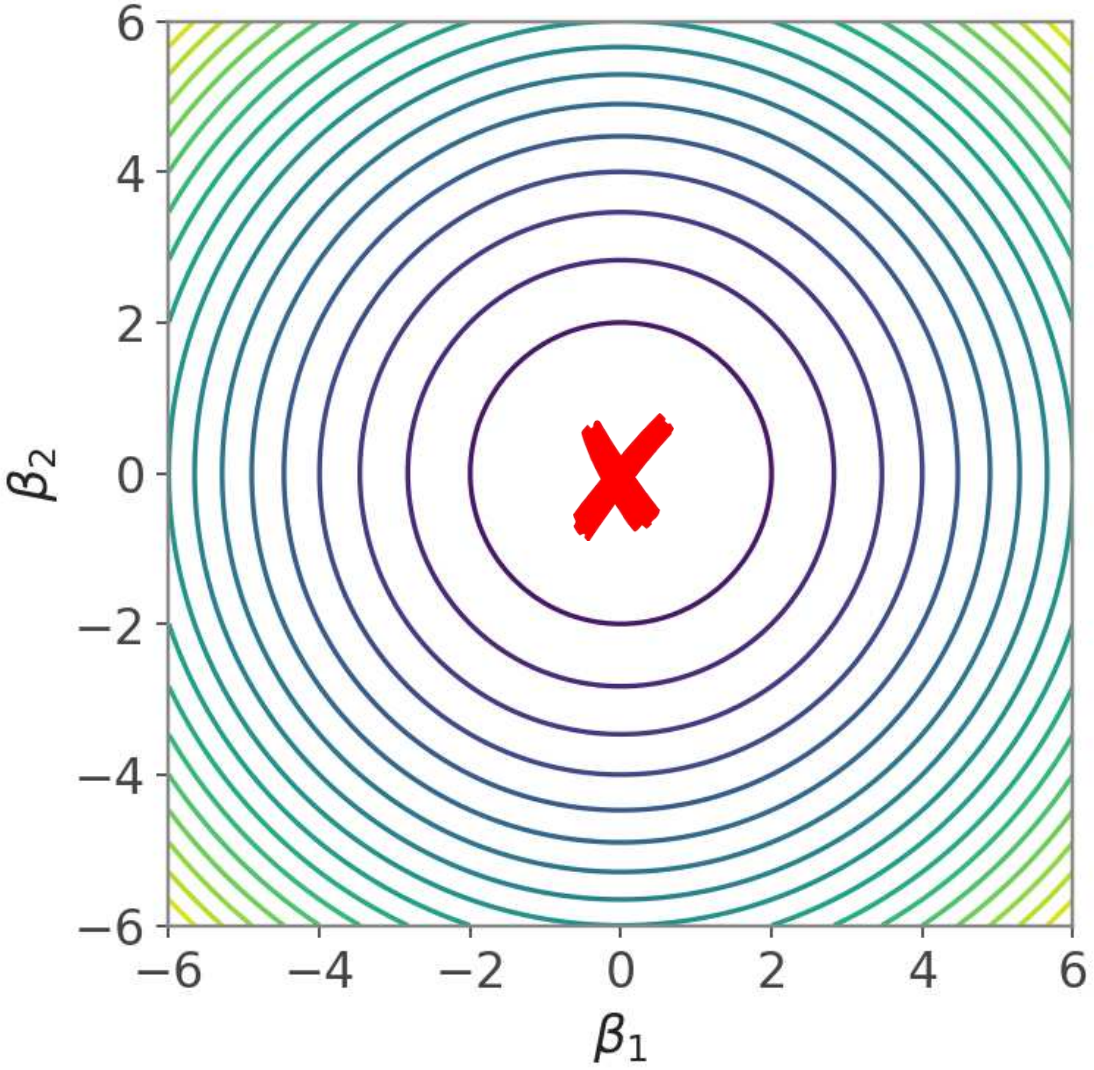
$$\frac{\partial \mathcal{L}}{\partial \beta_0} = 0, \qquad \frac{\partial \mathcal{L}}{\partial \beta_1} = 0$$

$$\mathcal{L}_{\text{MSE}} = \frac{1}{n}\sum_{i=1}^{n}(y_i - (\beta_0 + \beta_1 x_i))^2$$

observed predicted

For linear regression, we can just solve this analytically!

Note: this is *not* the case for most complex models (you'll see later, neural networks! GD will return…)



Approach #3: Solving the problem analytically

Remember your old friend, chain rule

$$\mathcal{L}_{\text{MSE}} = \frac{1}{n}\sum_{i=1}^{n}(y_i - (\beta_0 + \beta_1 x_i))^2$$

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

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

$$\sum_{i=1}^{n} y_i - \sum_{i=1}^{n} \beta_0 - \sum_{i=1}^{n} \beta_1 x_i = 0$$

Separate the summations

$$n\bar{y} - n\beta_0 - n\beta_1 \bar{x} = 0$$

These are just the means, since: $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$

And β_0 is just a constant that we add n times

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

Sample means

Then rearrange to solve for β_0

Approach #3: Solving the problem analytically

$$\mathcal{L}_{\text{MSE}} = \frac{1}{n}\sum_{i=1}^{n}(y_i - (\beta_0 + \beta_1 x_i))^2$$

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

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

Plug and chug so we can solve for β_1

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

Approach #3: Solving the problem analytically

$$\mathcal{L}_{\text{MSE}} = \frac{1}{n}\sum_{i=1}^{n}(y_i - (\beta_0 + \beta_1 x_i))^2$$

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

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

$$\sum_{i=1}^{n} x_i(y_i - (\bar{y} - \beta_1 \bar{x}) - \beta_1 x_i) = 0$$

$$\sum_{i=1}^{n} x_i(y_i - \bar{y}) + x_i \beta_1 (\bar{x} - x_i) = 0$$

$$\sum_{i=1}^{n} x_i(y_i - \bar{y}) = \beta_1 \sum_{i=1}^{n} x_i\,(x_i - \bar{x})$$

$$\beta_1 = \frac{\sum_{i=1}^{n} x_i(y_i - \bar{y})}{\sum_{i=1}^{n} x_i\,(x_i - \bar{x})}$$

Typically, this is rewritten as:

$$\beta_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

← **Covariance** of x, y

← **Variance** of x

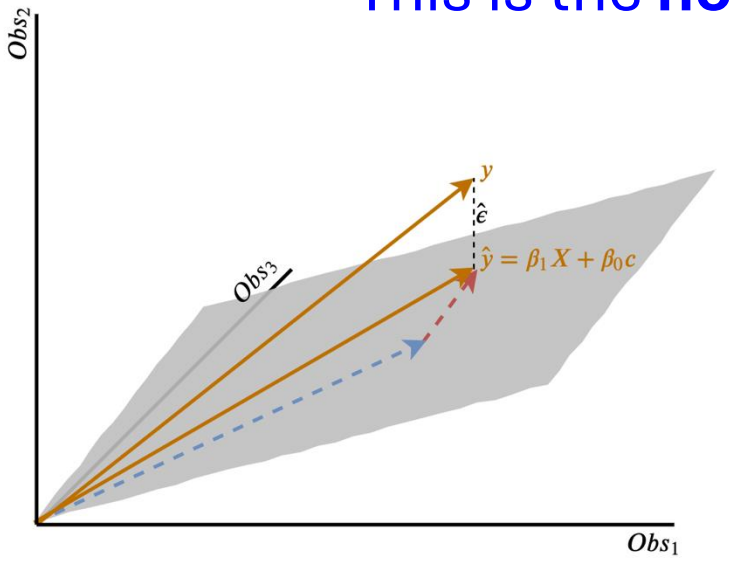
Approach #3: Solving the problem analytically

So…Given our observed data (x_i, y_i) we can "easily" compute the best-fit line using:

$$\beta_0 = \bar{y} - \beta_1 \bar{x} \qquad \beta_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

On HW you'll show these are equivalent

It is sometimes more convenient to write these equations in **matrix form:**



This is the **normal equation** for ordinary **least-squares regression**

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}$$

Because it minimizes the **square of the residuals**

$$\boldsymbol{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \dots & \dots \\ 1 & x_n \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix}$$

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta}$$

We'll call $\boldsymbol{X}$ **the design matrix** and $\boldsymbol{y}$ **the response vector**

Approach #3: Solving the problem analytically

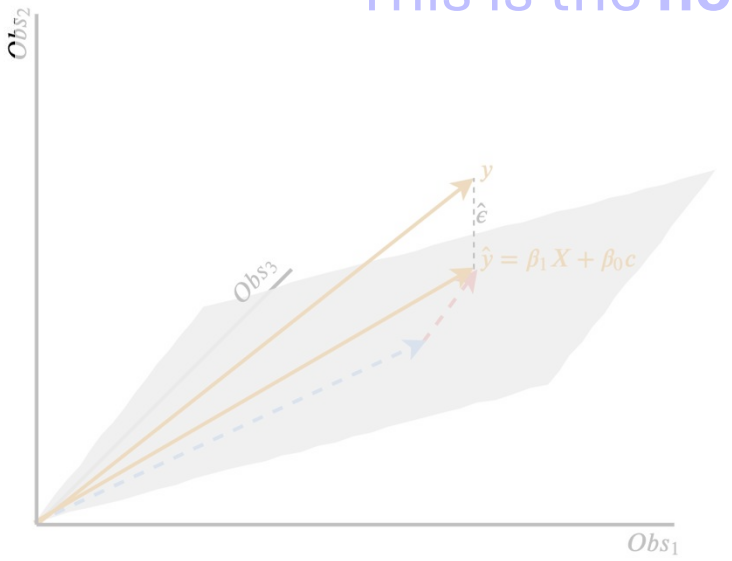
This solution is exact! No iteration needed. Least-squares is usually the way to go. That said, there are sometimes reasons we'd use gradient descent instead:

Model does not have a nice closed form

Can't calculate derivatives

We are dealing with a huge number of variables

Matrix inversion is expensive



This is the **normal equation** for ordinary **least-squares regression**

Because it minimizes the **square of the residuals**

$$\hat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}$$

$$\boldsymbol{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \dots & \dots \\ 1 & x_n \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix}$$

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta}$$

Questions?

How to evaluate the trained model

Recall: Data splits for training and testing



Model learns the task

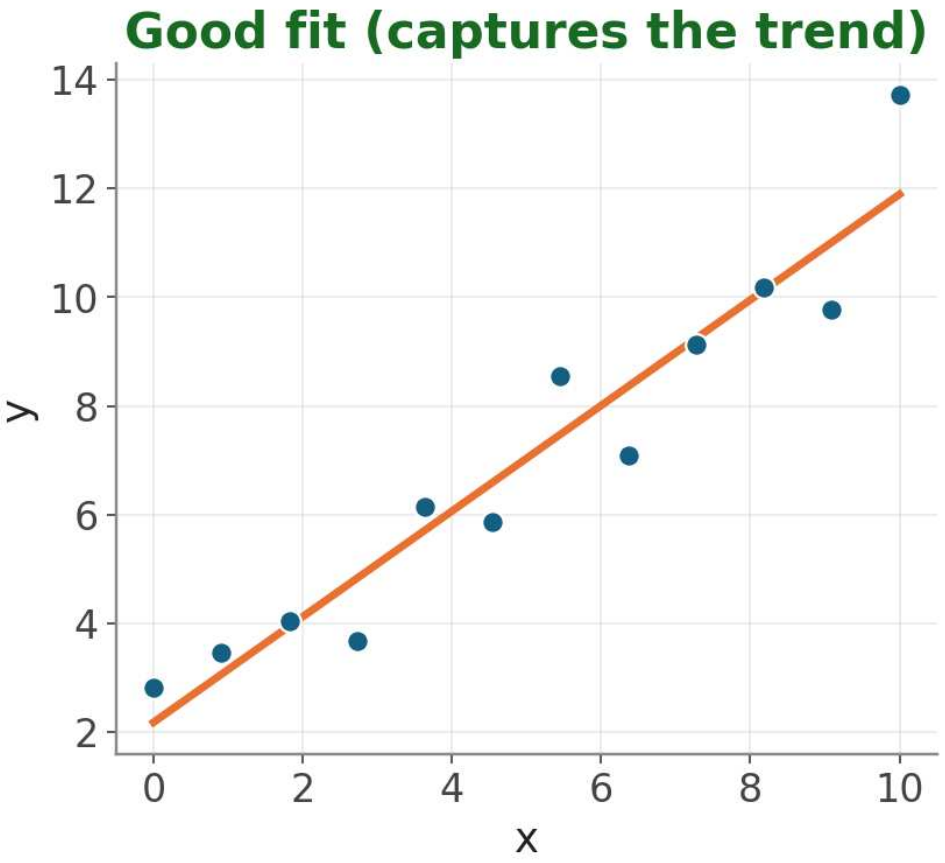
How good is the model on "new" data?

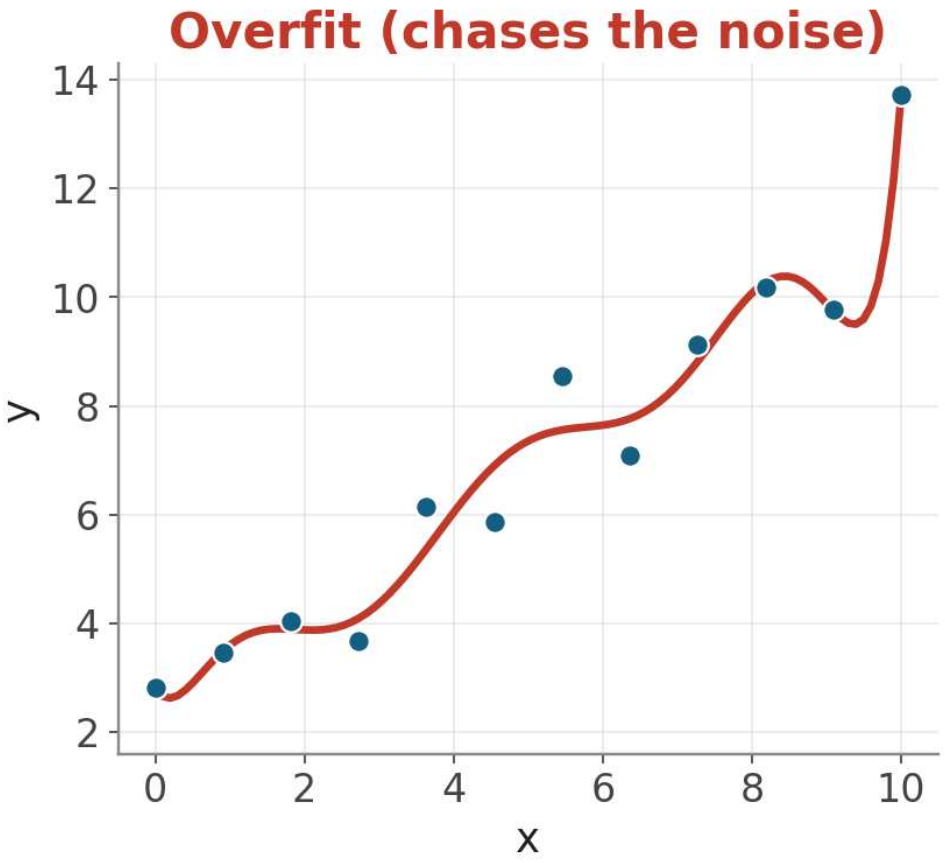
Performance metrics will be reported on the test dataset

Remember we must avoid data leakage between training/test

We'll discuss this in more detail later!

A good fit *vs.* an overfit model



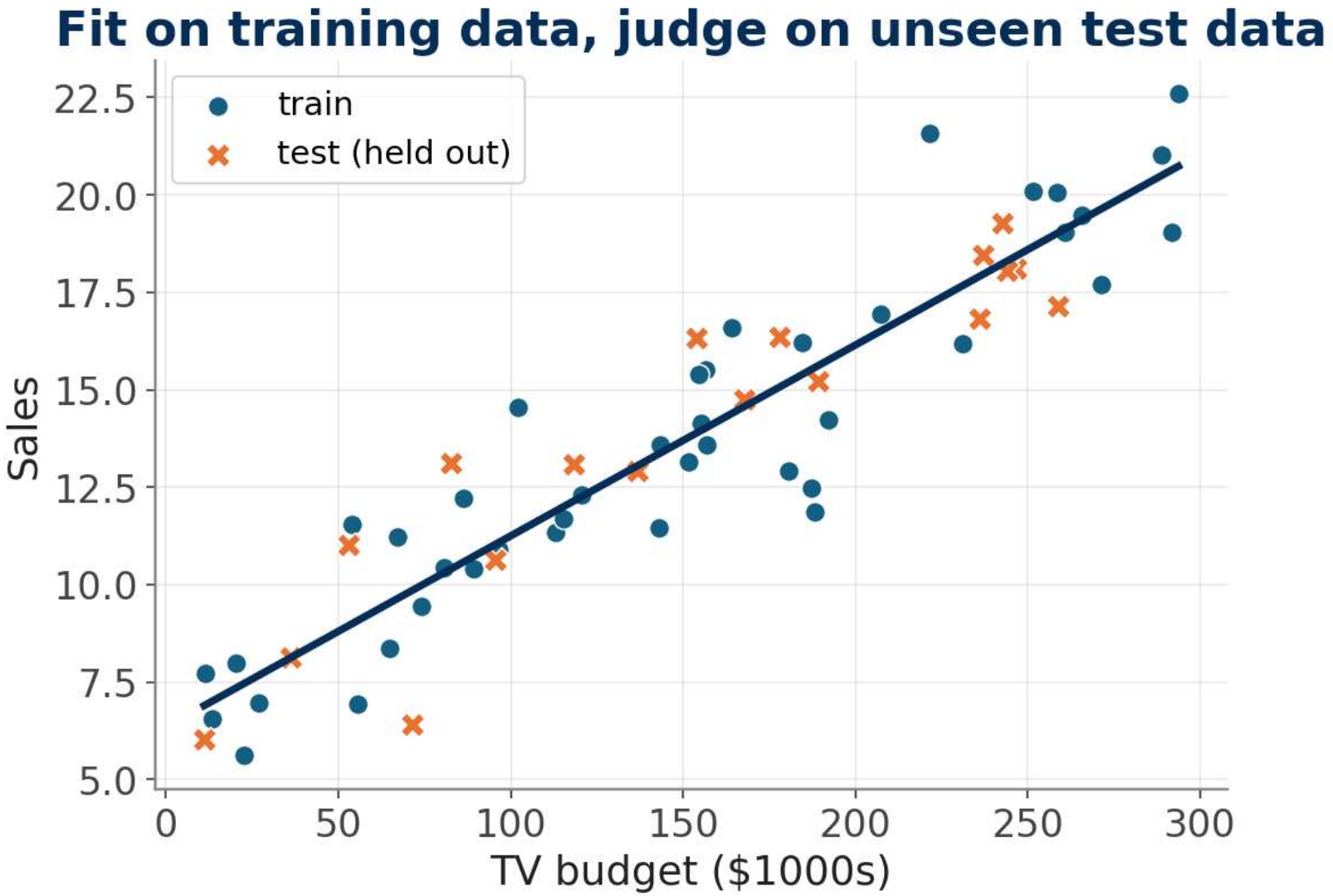


According to our MSE loss, the right one looks better...

How to avoid overfitting?

Note: with linear models, it is very hard to overfit – so this is less of a concern. But later on, we'll cover more complex models that can very easily overfit!

Solution: Split the data for training & testing



- We compute β_i using x_i and y_i from the training set only.
- Then we compute MSE of that best-fit line on new (unseen) data

This way: if we overfit to the training data, we'll see high MSE on the test data → a red flag!

Solution: Split the data for training & testing



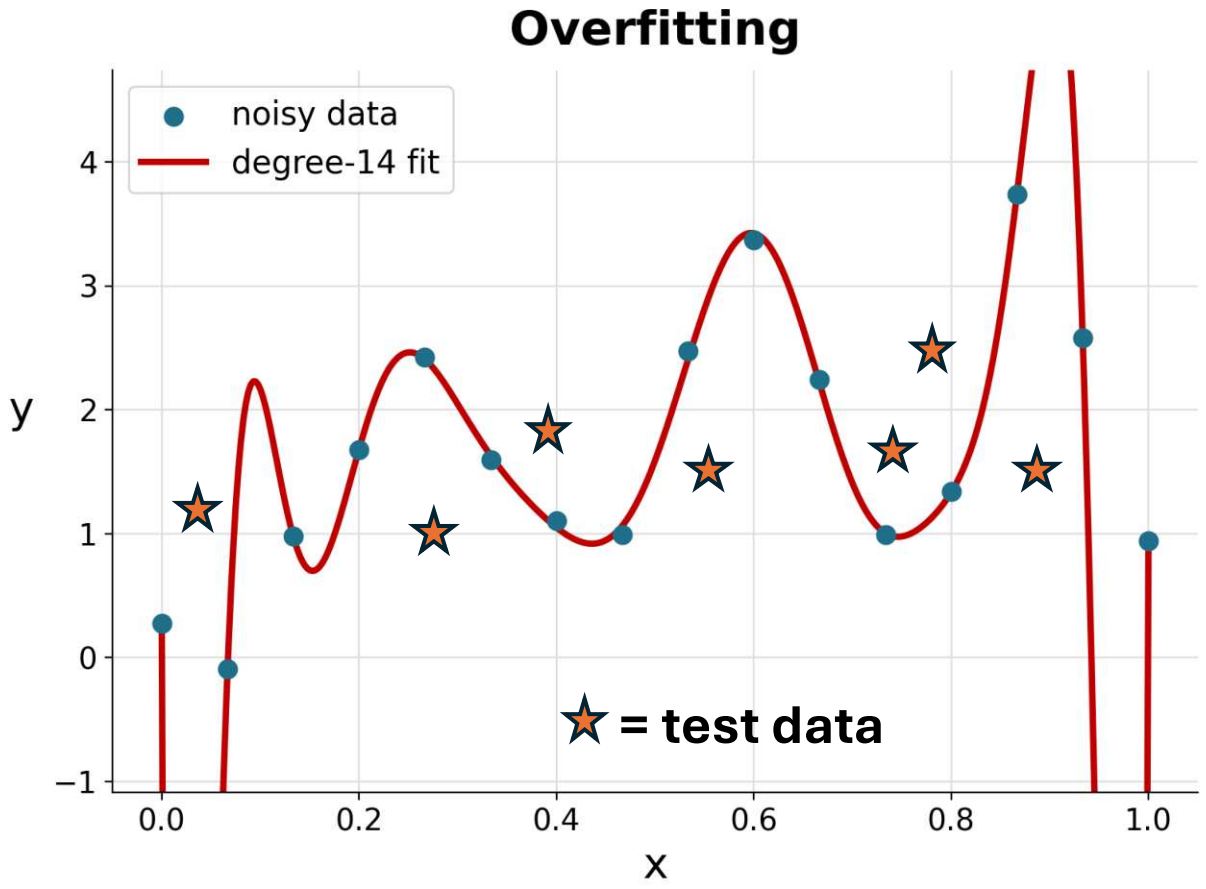
We typically use “most” of the data for training, so our model is well-informed

Leave just enough test data to get trustworthy metrics

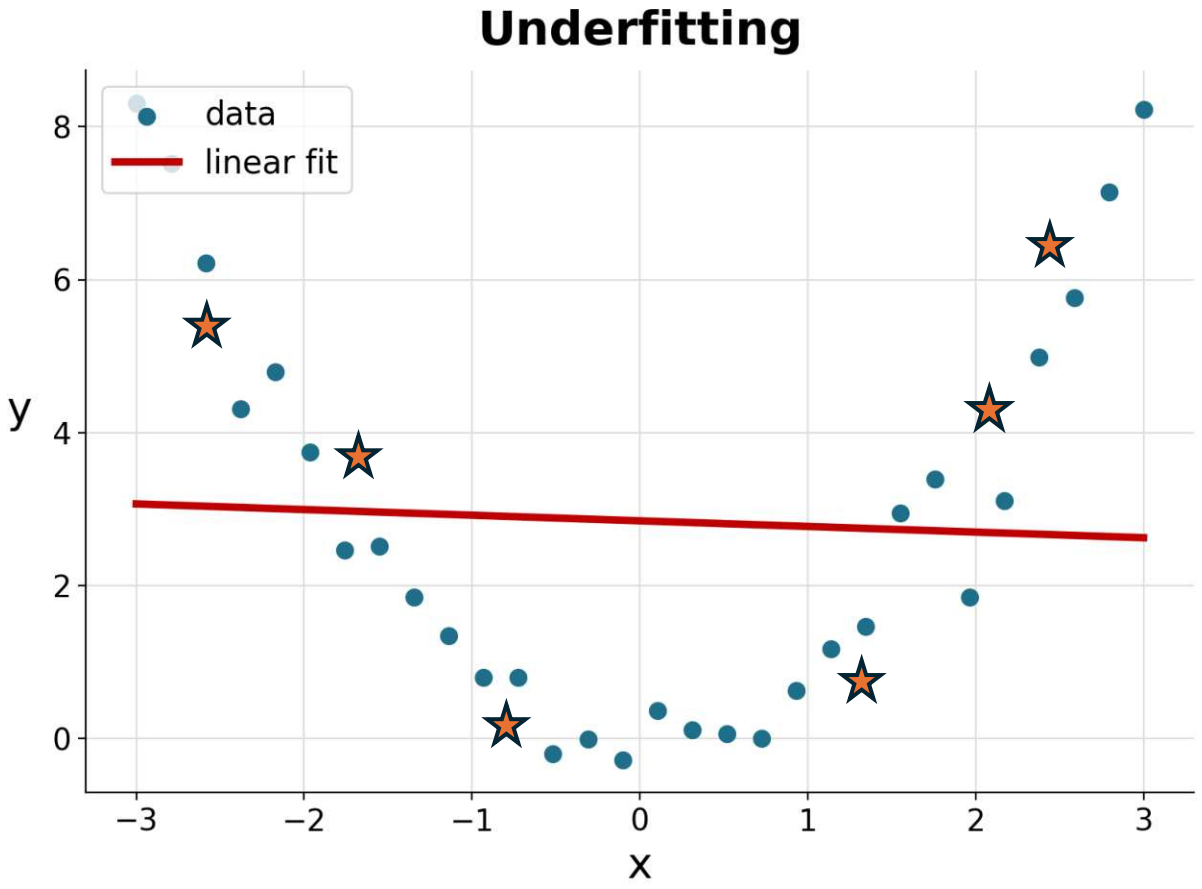
Report performance metrics on both datasets.
This will help you diagnose and improve your model.

Diagnostics based on training and test error

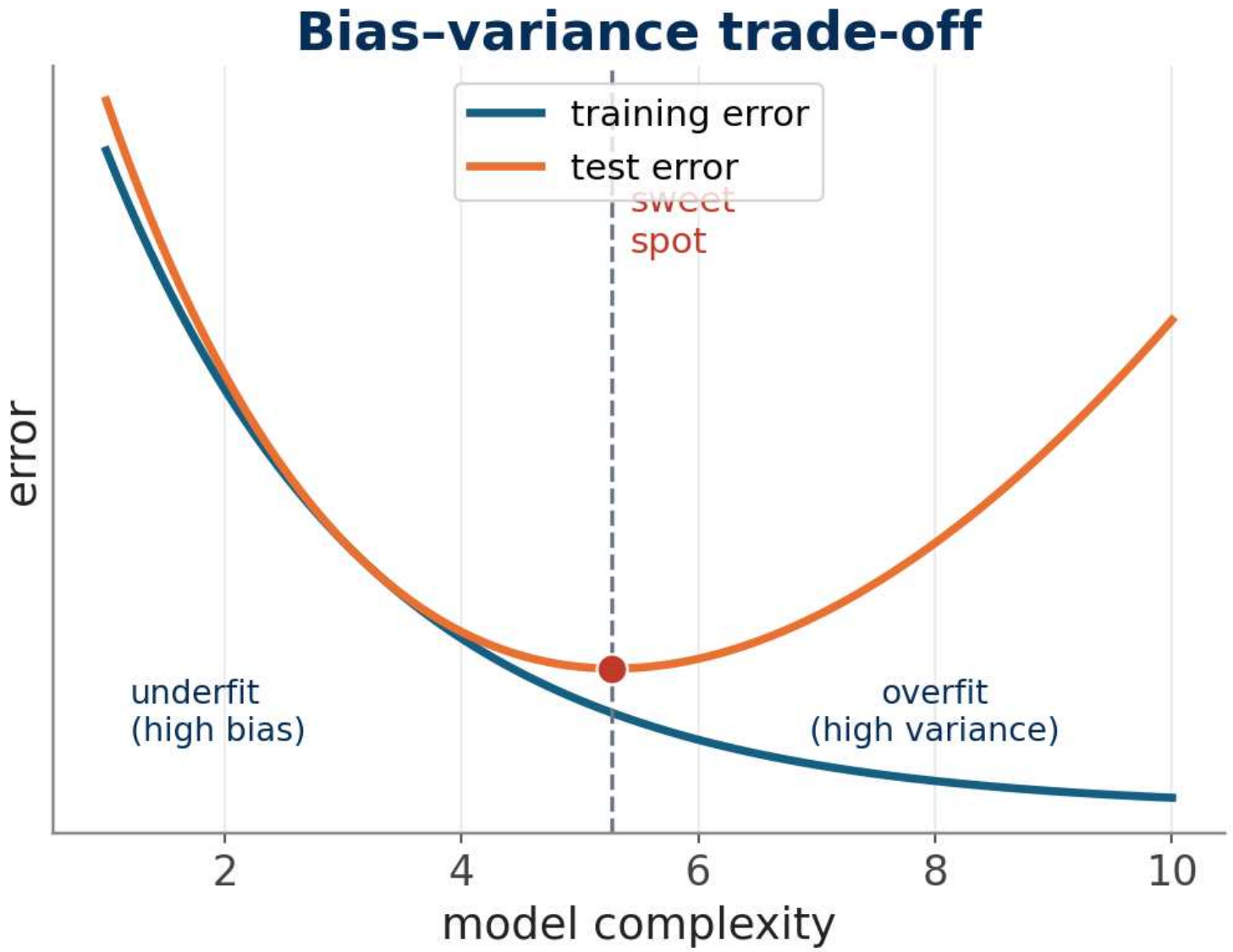
Low training error
High test error



High training error
High test error



Diagnostics based on training and test error



Bias = systematic error due to some incorrect assumption ("I believe the data is linear")

Variance = how much the model changes when trained on different subsets of the data

Model complexity:
Too simple → high bias;
Too complex → high variance

There is a "sweet spot" where test error is minimal

Diagnostics based on training and test error

To make predictions

To understand relationships

Raw error can be hard to judge since it varies so much across datasets...

Sales in dollars RMSE = $5.0

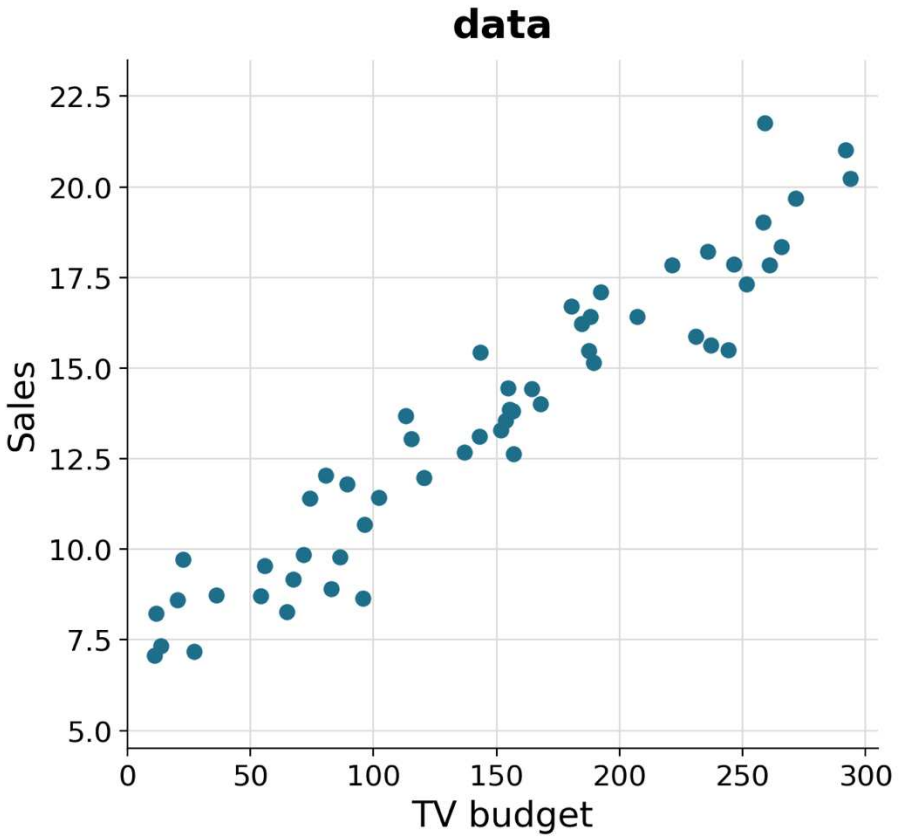
Sales in cents RMSE = 500

Perhaps we can use **another metric that is more universal**

Is MSE (or RMSE) alone enough to understand our model's performance?

There is a "sweet spot" where **test error** is minimal

The coefficient of determination: R^2

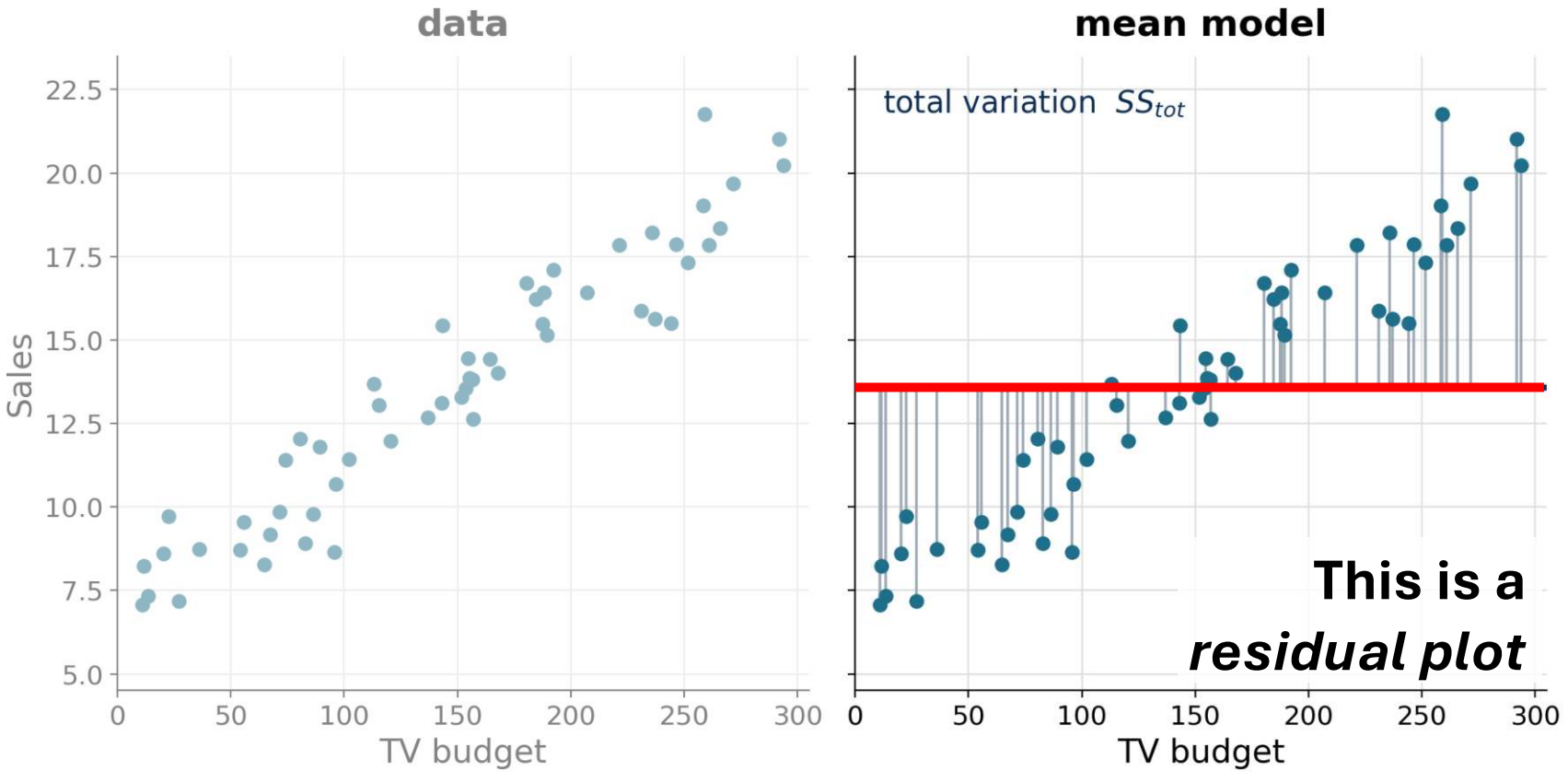


Question: Is there strong ***linear correlation*** between sales and TV budget?

In other words…When TV budget increases, do sales increase proportionally?

Let's answer this by computing the coefficient of determination, or "R-squared" ($\boldsymbol{R^2}$)

The coefficient of determination: R^2



Step 1:

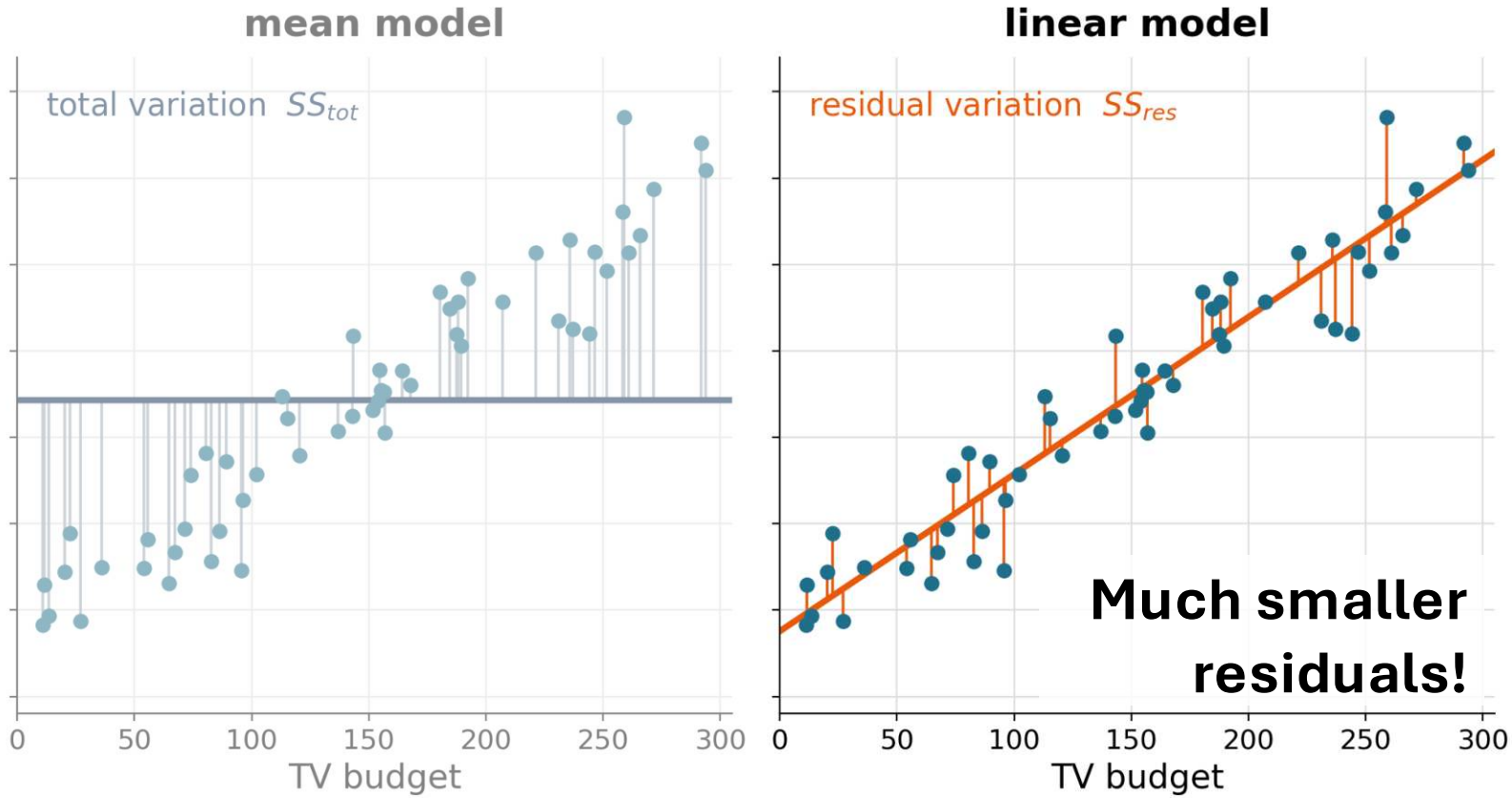
Compute the **total variance (SS_{tot})** of the data – the error of always predicting **the mean**

$$SS_{\text{tot}} = \text{sum of square residuals} = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

The coefficient of determination: R^2

Step 2:

Compute the **residual variance (SS_{res})** after building a linear model of the data



$$SS_{\text{res}} = \sum_{i=1}^{n} (y_i - \hat{y})^2$$

The coefficient of determination: R^2

Step 3:

Compute R^2
(a unitless quantity)

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$$

The fraction of variance explained by a linear model *vs.* always guessing the mean

Suppose you have a linear model that **fits the data very well**

- SS_{res} should be small in comparison to SS_{tot}
- $\frac{SS_{\text{res}}}{SS_{\text{tot}}} \to 0$
- $R^2 \to 1$

Suppose you have a linear model that **fits the data poorly**

- If the model is no better than the mean: $SS_{\text{res}} \sim SS_{\text{tot}}$
- $\frac{SS_{\text{res}}}{SS_{\text{tot}}} \to 1$
- $R^2 \to 0$

The coefficient of determination: R^2

Step 3:

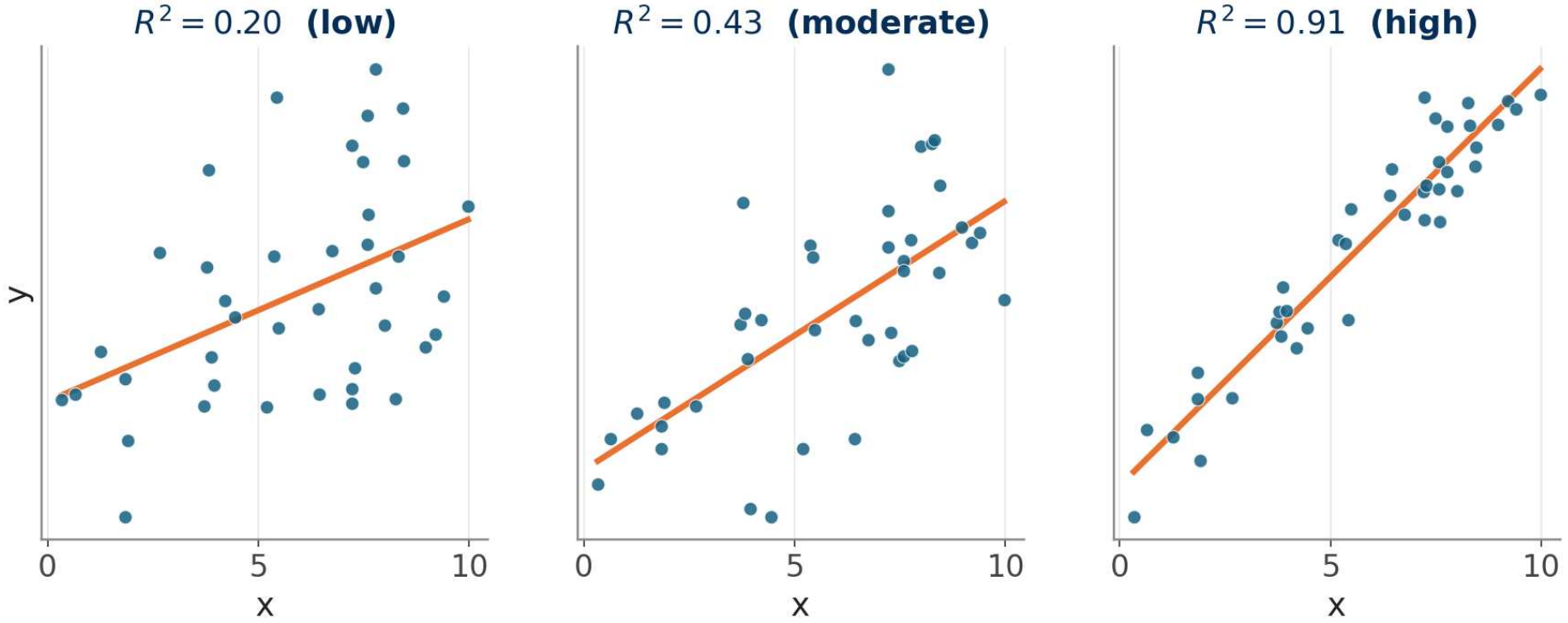
Compute R^2
(a unitless quantity)

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$$

The fraction of variance explained by a linear model *vs.* always guessing the mean

R^2 = 1 → perfect;
R^2 = 0 → no better than the mean

R^2 = 0.85 → model explains 85% of the variance in *y*



Some caveats on using metrics like R^2

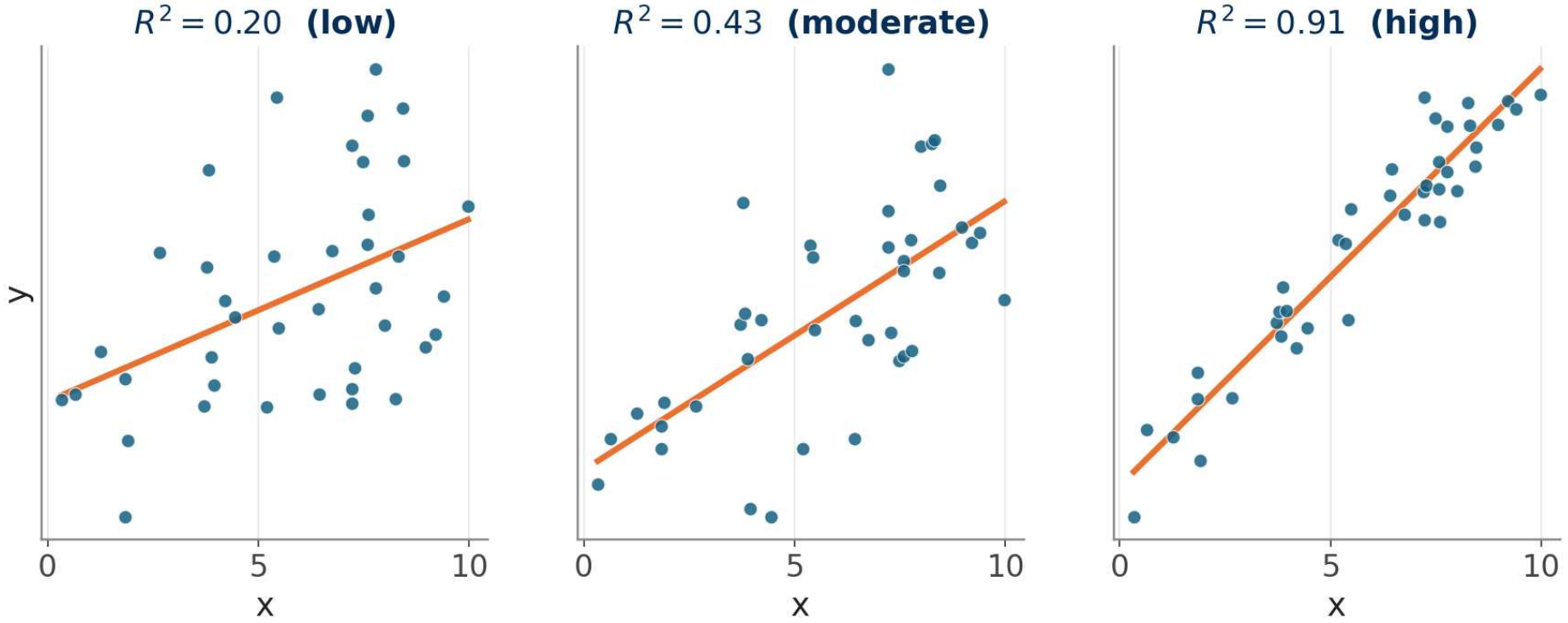
High R^2 does NOT mean the model has the right complexity

High R^2 does NOT prove causation (does x *cause* y?)

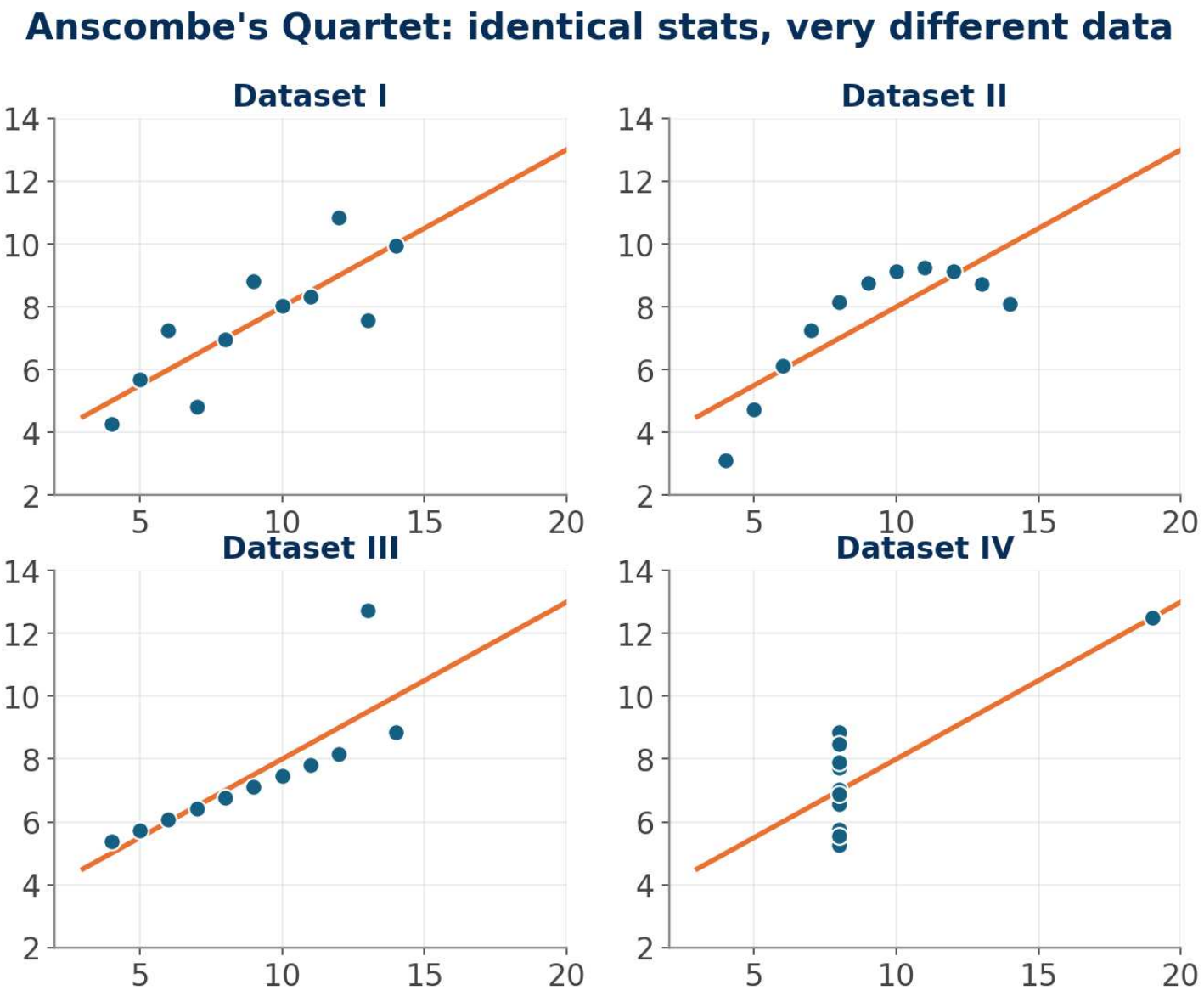
Always pair R^2 with visualization of the data!

R^2 = 1 → perfect;
R^2 = 0 → no better than the mean

R^2 = 0.85 → model explains 85% of the variance in *y*



Some caveats on using metrics like R^2



These all have the exact same mean, variance, best-fit line, and R^2

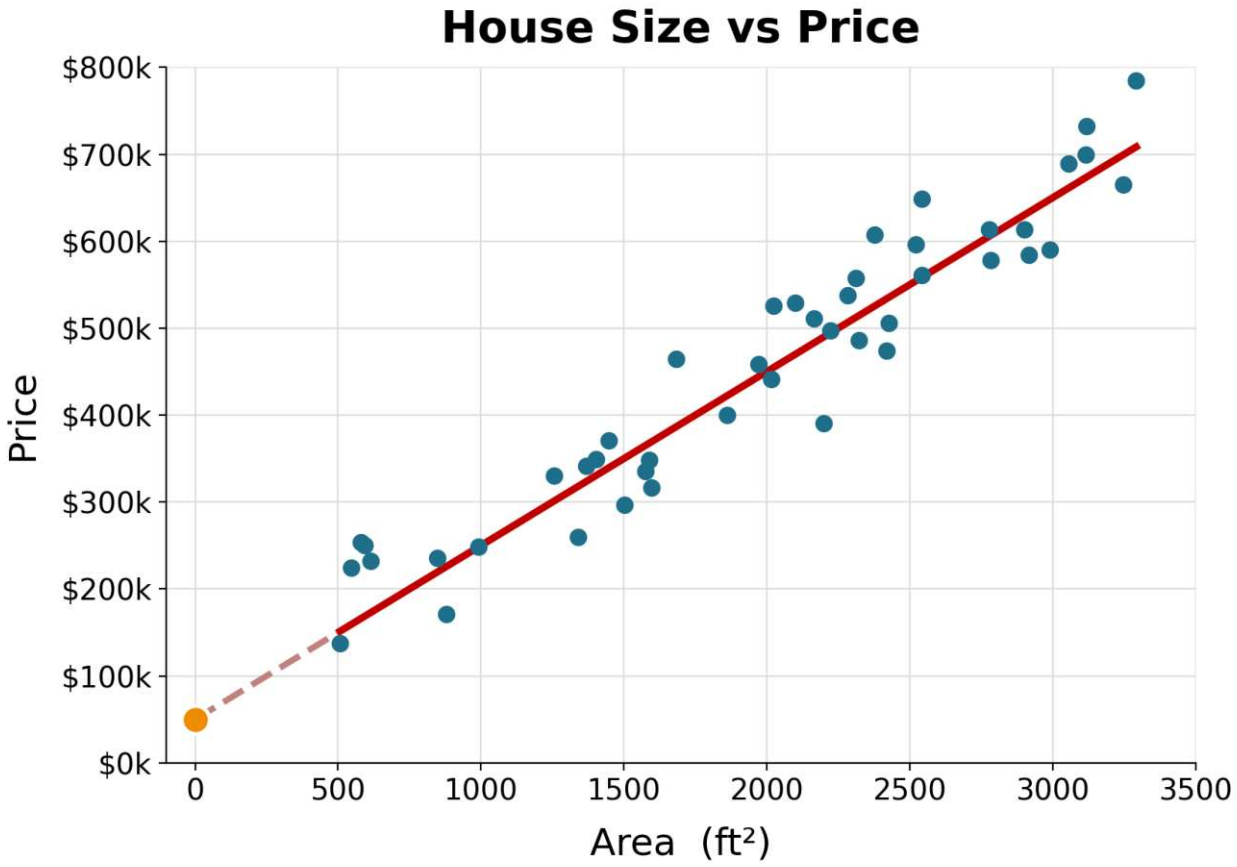
Always plot your data!

Questions?

Interpreting Linear Models

The meaning of each coefficient: β_0 and β_1

$$\hat{y} = \beta_0 + \beta_1 x$$



Question: How does the size of a house (in ft) affect its sale price?

From our linear fit:

- **The intercept $(\beta_0) \approx$ \$50k**
  - In this case, the intercept is not physically meaningful
- **The slope $(\beta_1) \approx$ \$200/ft²**
  - This means we pay \$200 more for every added sq. ft.

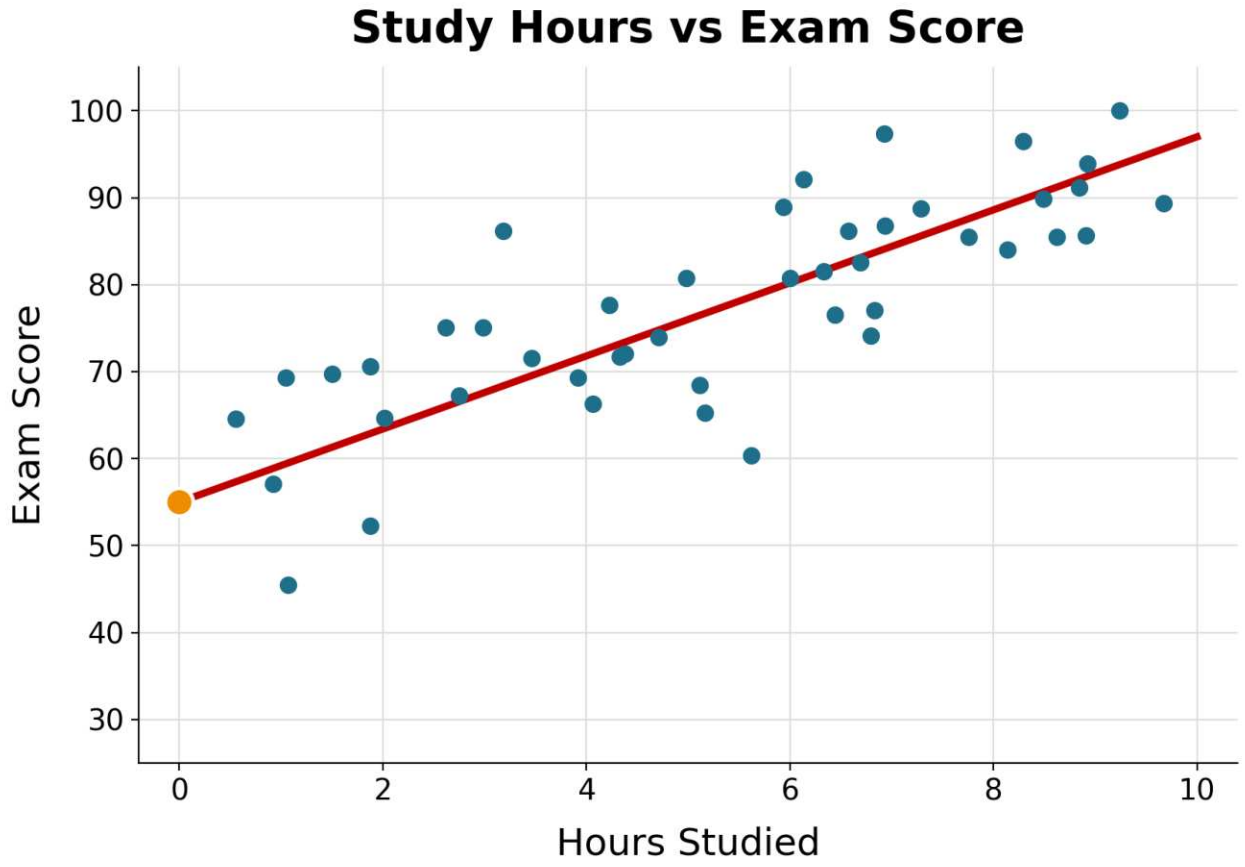
The meaning of each coefficient: β_0, β_1

$$\hat{y} = \beta_0 + \beta_1 x$$

Question: Does studying more lead to better exam scores?

From our linear fit:

- **The intercept $(\beta_0) \approx$ 55%**
  - Here, the intercept *is* meaningful (score without studying)
- **The slope $(\beta_1) \approx$ 4%/hr**
  - Your score increases by 4% every hour that you study



Measuring confidence

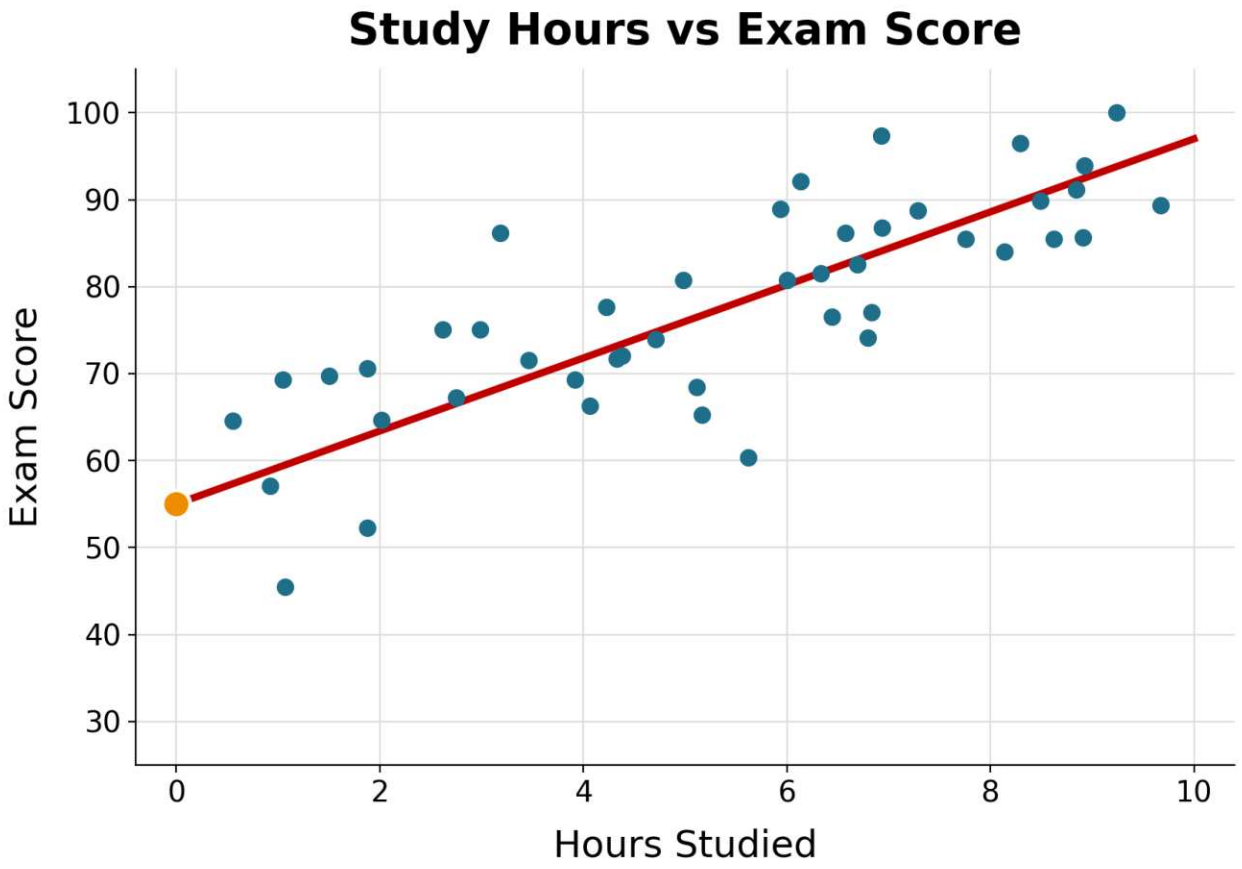
Our linear fit seems to indicate that YES, studying does lead to better exam scores.

But how *confident* are we in this answer?

Question: Does studying more lead to better exam scores?

From our linear fit:

- **The intercept $(\beta_0) \approx 55\%$**
  - Here, the intercept *is* meaningful (score without studying)
- **The slope $(\beta_1) \approx 4\%/hr$**
  - Your score increases by 4% every hour that you study



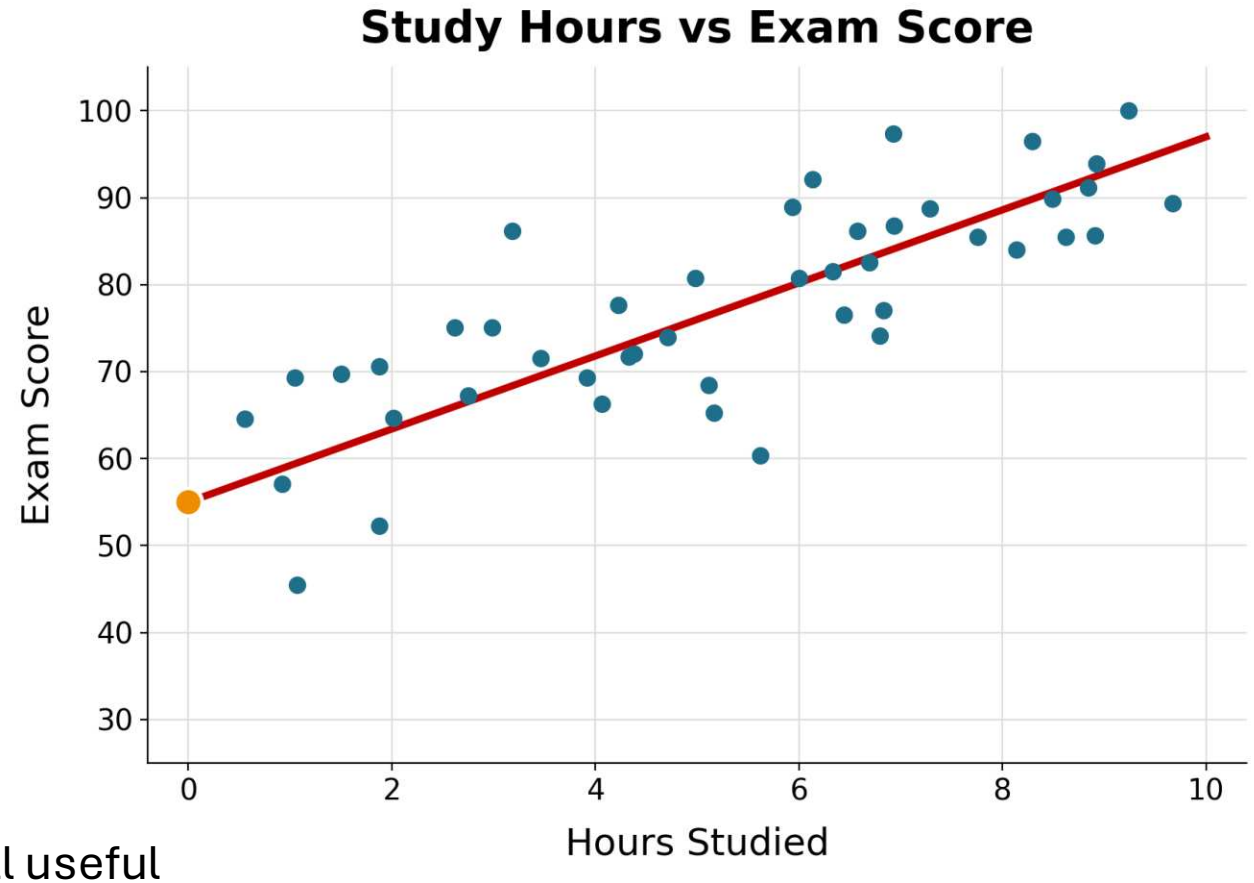
Measuring confidence: R^2

Our linear fit seems to indicate that YES, studying does lead to better exam scores.

But how *confident* are we in this answer?

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}} \approx 0.63$$

→ Our linear model explains 63% of the variation in predicted exam scores. The remaining variation is likely caused by unaccounted for factors (sleep, intelligence, lecture attendance, *etc.*)



Note: R^2 is not strictly a measure of confidence – but still useful

Measuring confidence: Standard Error

Our linear fit seems to indicate that YES, studying does lead to better exam scores.

But how *confident* are we in this answer?

We can measure the **standard error** in each coefficient – *e.g.*, the slope:

$$SE(\beta_1) = \sqrt{\frac{SS_{\text{res}}/(n-2)}{\sum_i (x_i - \bar{x})^2}}$$

Because we estimate **2 coefficients**

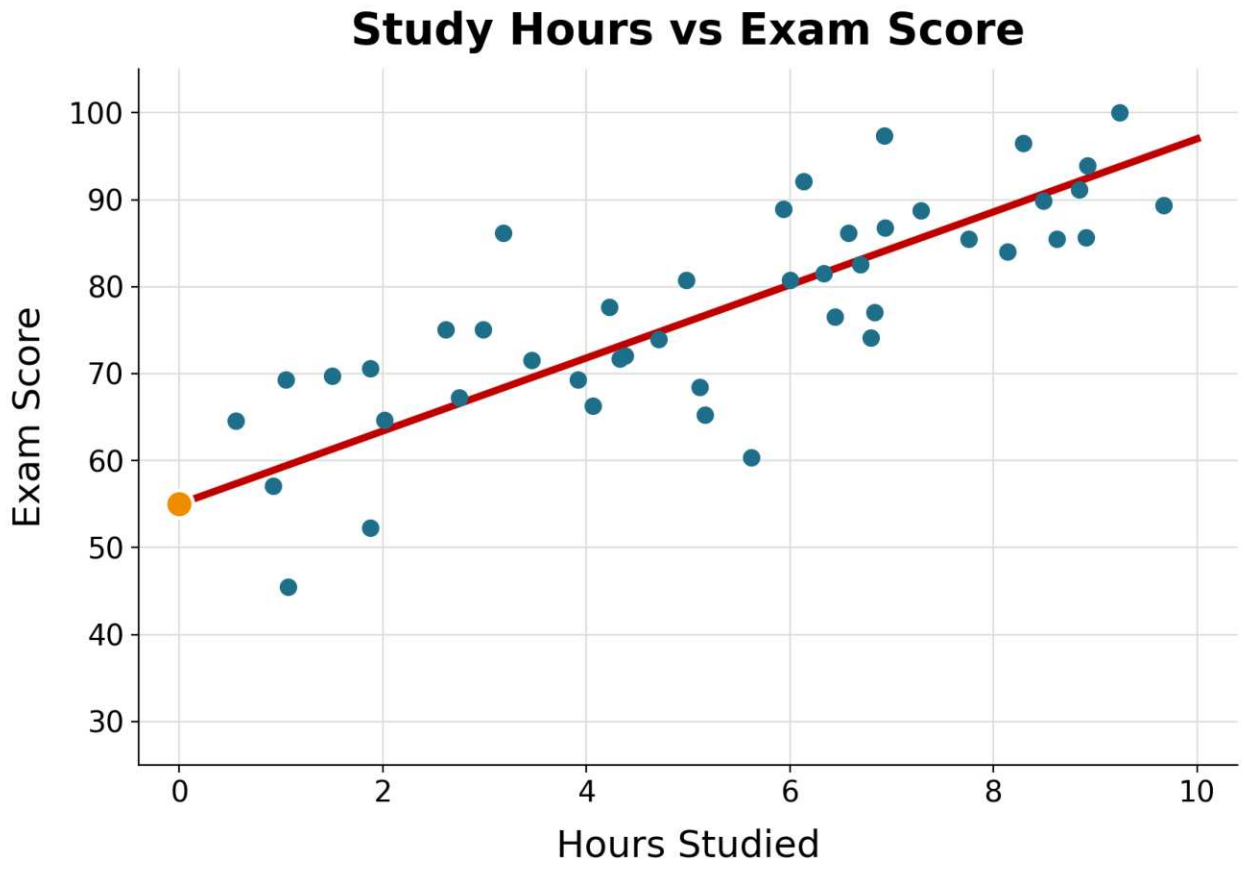
Spread in x

Smaller residuals → smaller SE

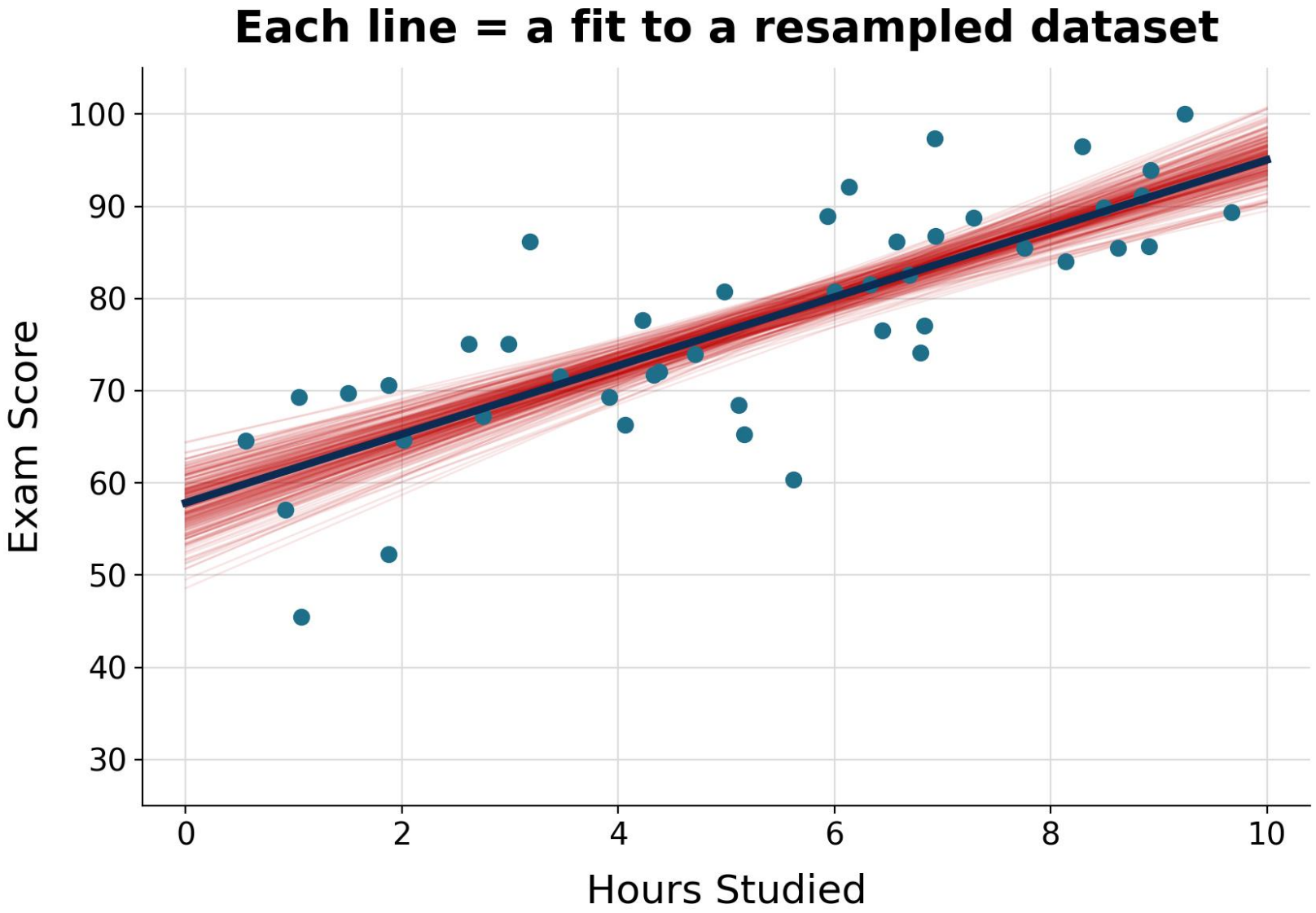
More spread in x → smaller SE

Smaller SE = less uncertainty

n is the total number of samples

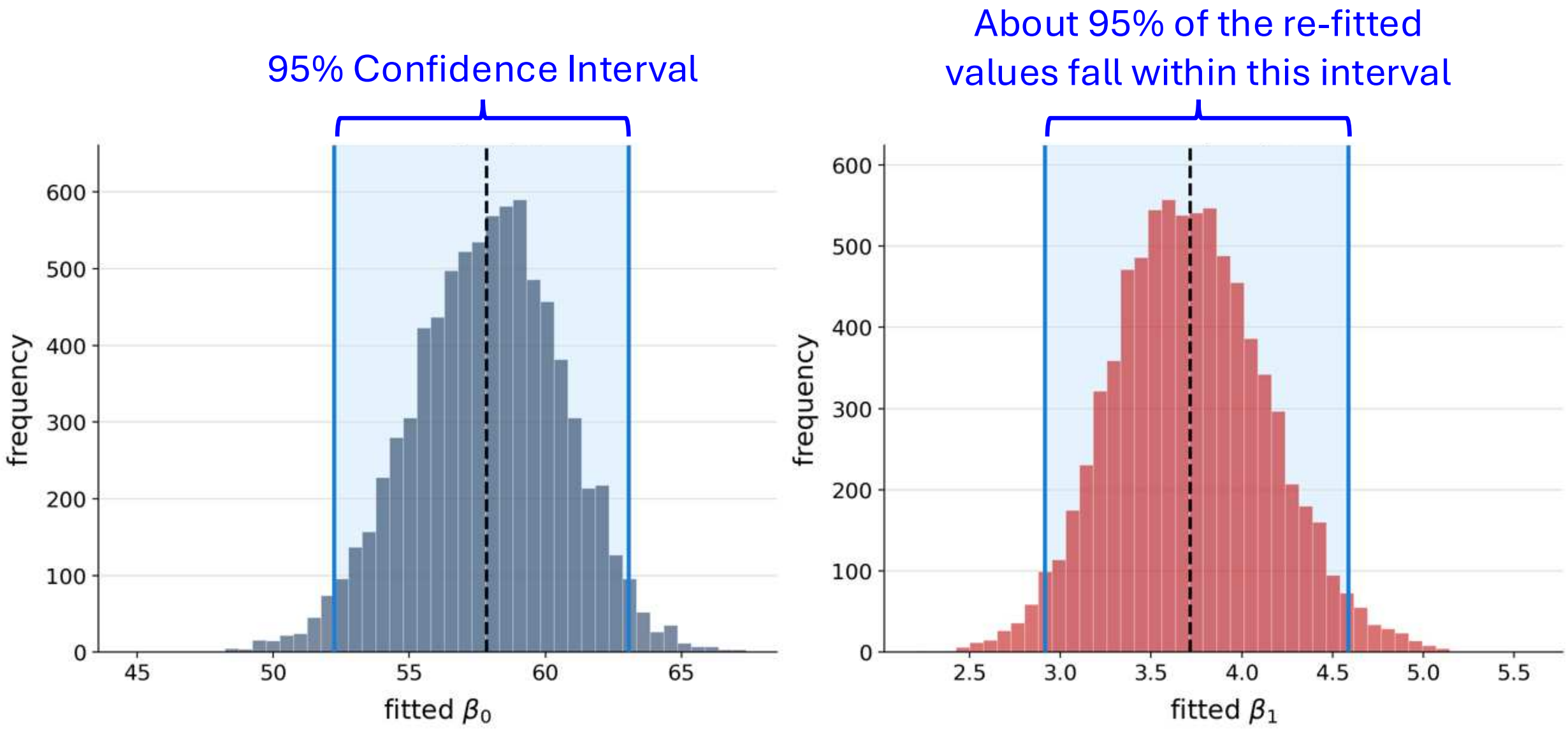


Measuring confidence: Standard Error

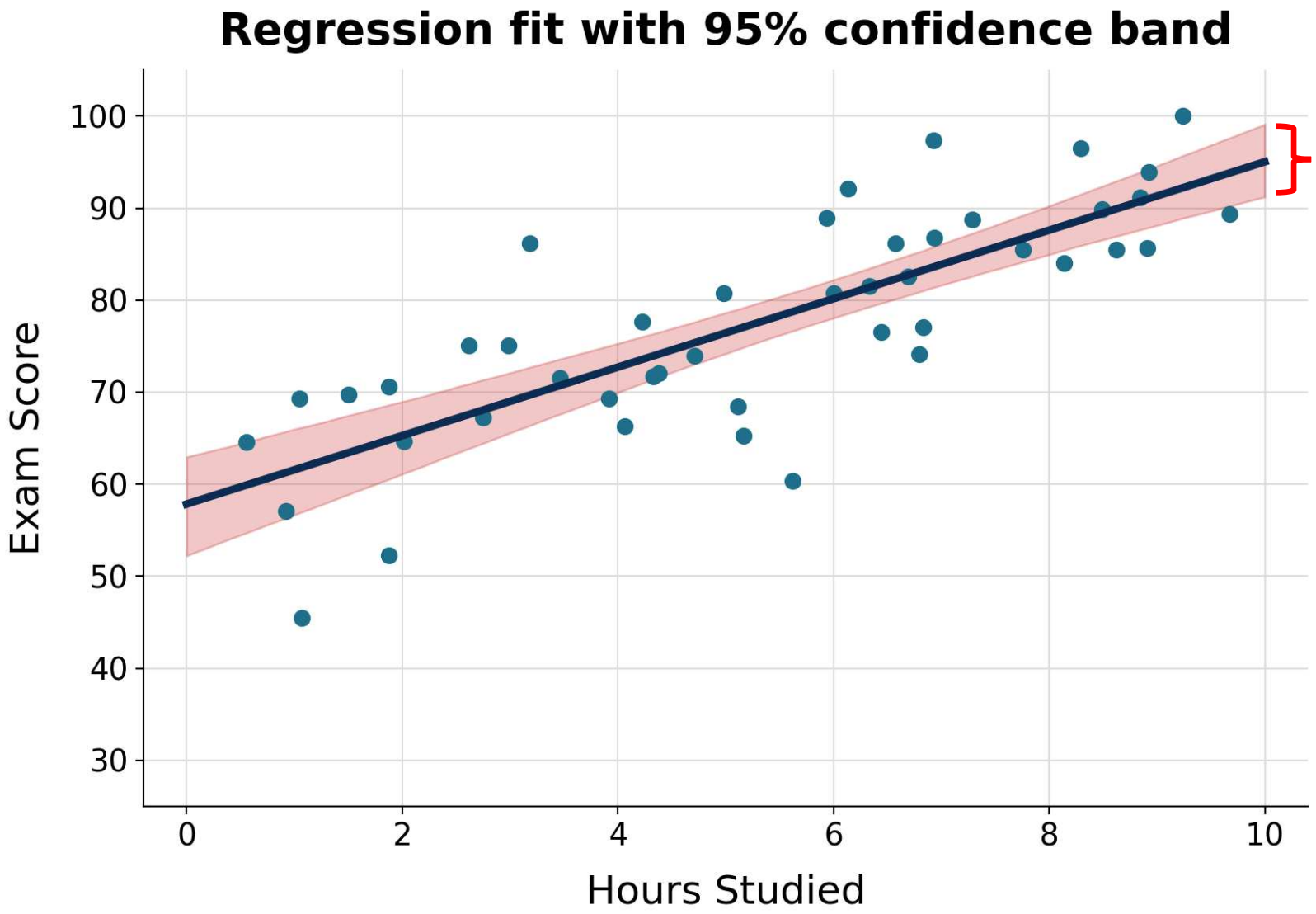


Think about the standard error as measuring how much our estimated intercept and slope would vary if we **repeatedly drew different random samples** from the population and refit the model.

Measuring confidence: Confidence intervals



Measuring confidence: Confidence intervals

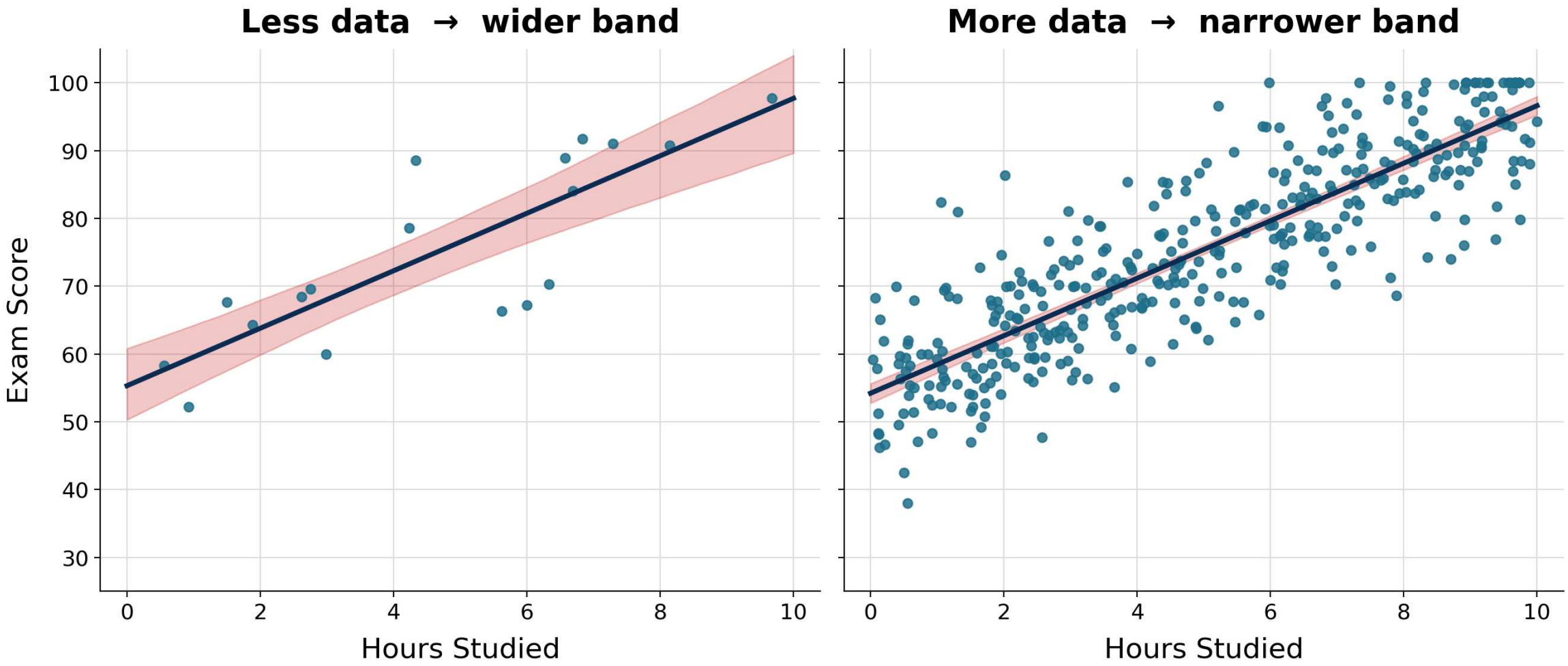


95% confidence interval

We can draw this using a shaded region around our linear model

Roughly:
95% CI = $\hat{\beta}_i \pm 2(SE)$
for each coefficient

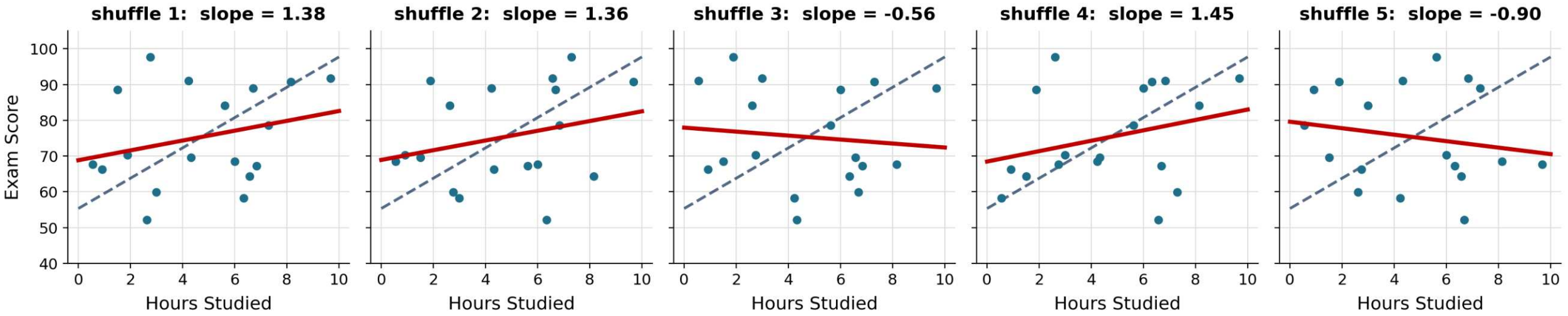
Measuring confidence: Confidence Intervals

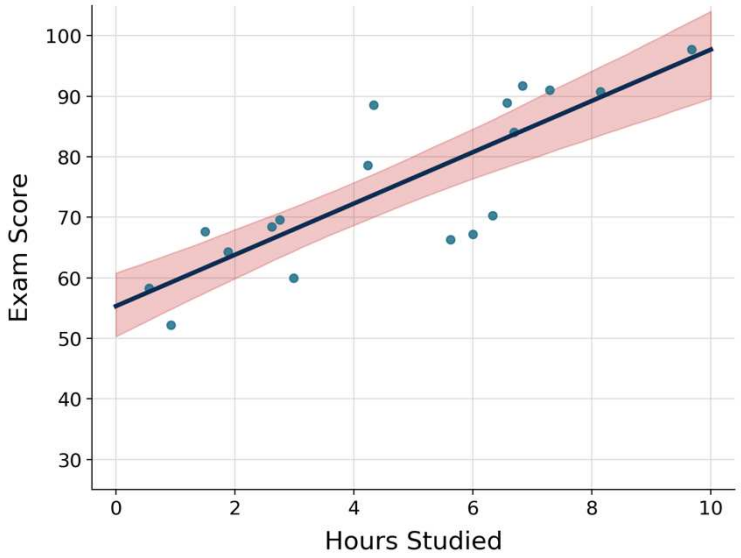


Bigger (*and* more diverse) datasets pin down the coefficients

Measuring confidence: p-values

*We'll come back to this later





Could our result be explained by chance?

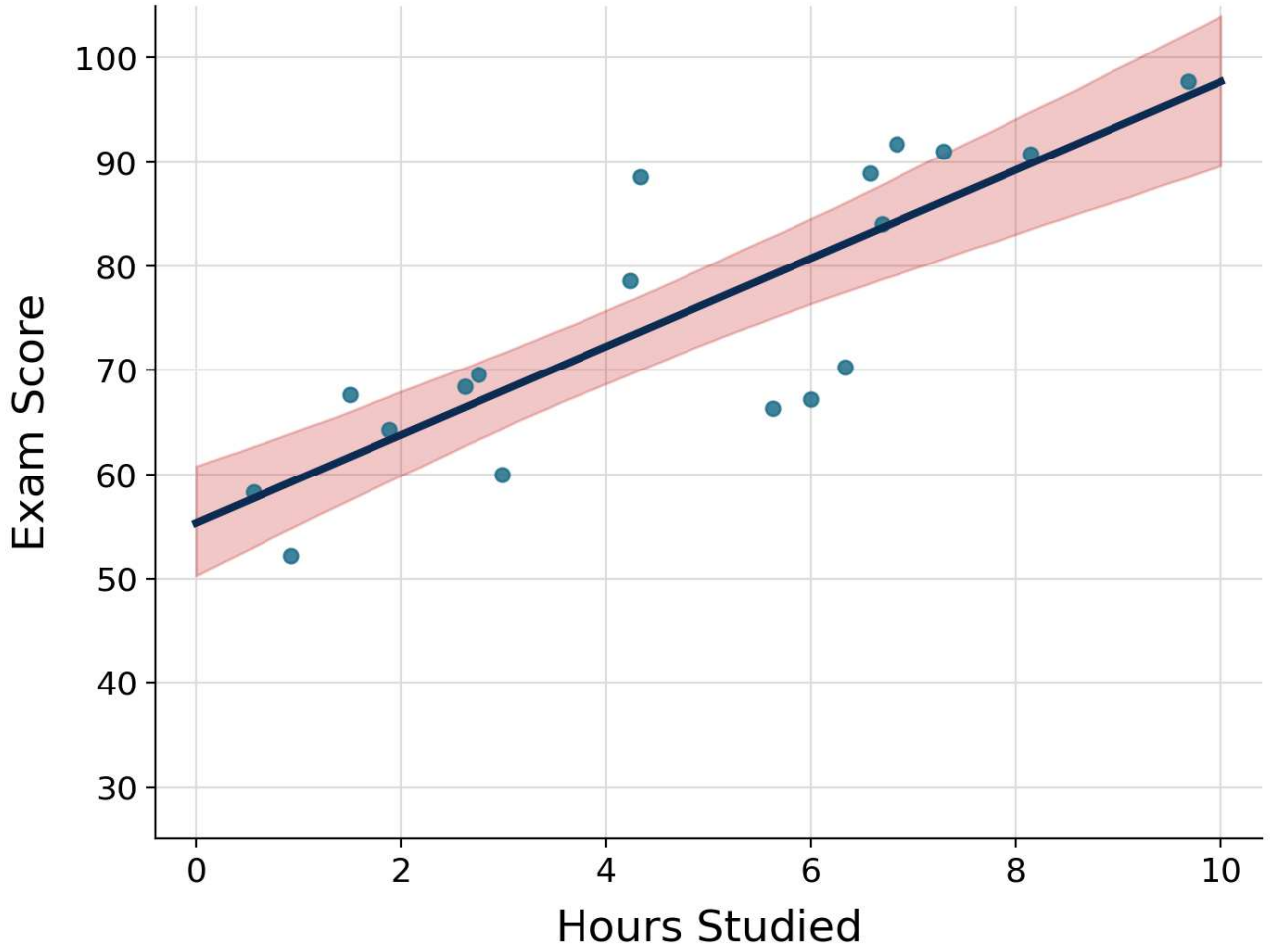
- Shuffle the exam scores against study hours
- Refit the line each time to show what pure chance produces
- Shuffled slopes range from −0.9 to 1.5, while the real slope (> 4) is much steeper!
- **Conclusion:** chance almost never gives a slope this big, so the relationship is likely real (**small *p*-value**)

***p*-value = fraction of shuffled slopes as extreme as ours**

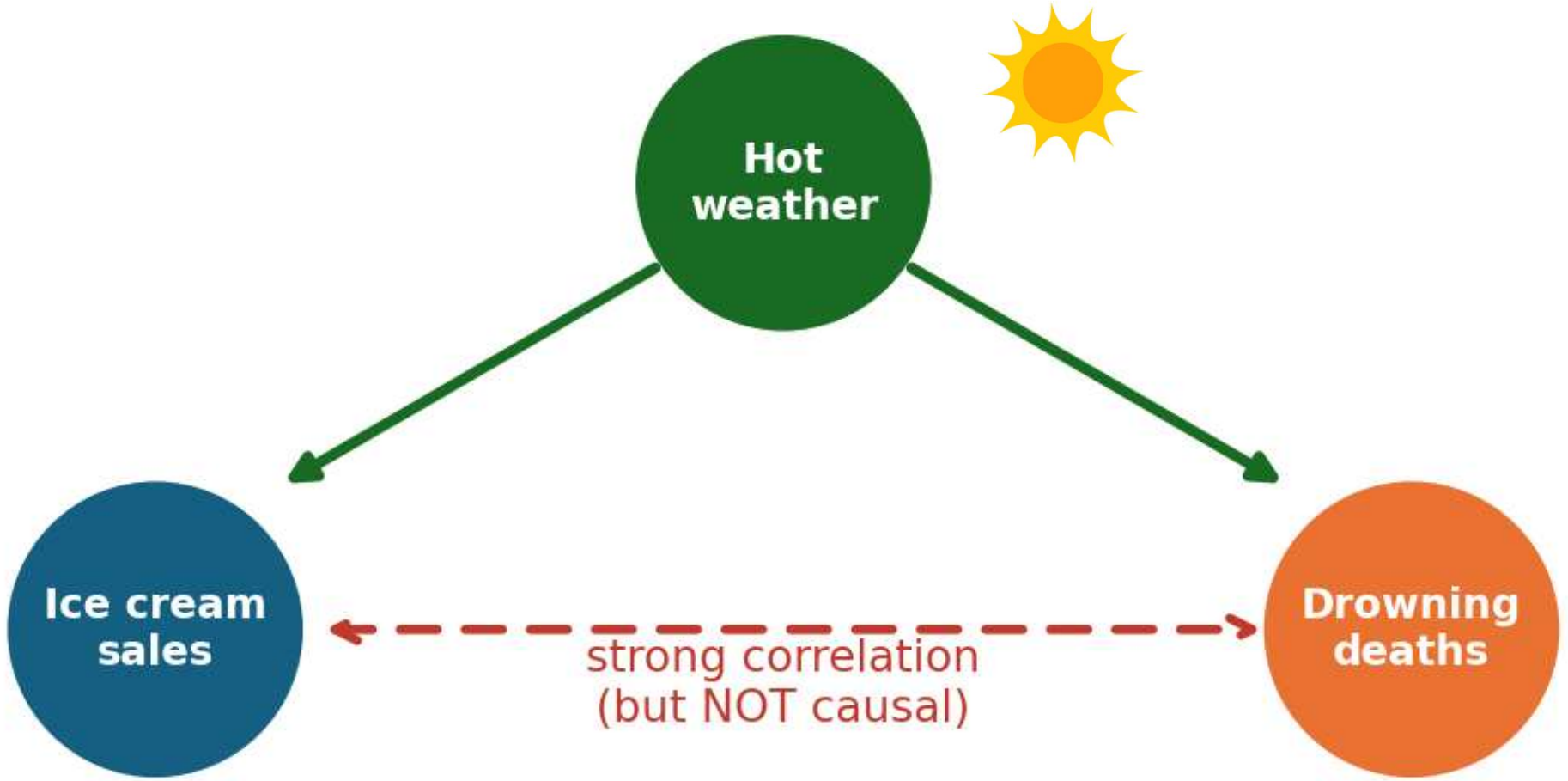
Measuring confidence: *Not* Causation

- Reporting R^2, standard error, and p-values provides a good way to measure correlations and uncertainties
  - $p < 0.05$ is an often-used threshold on statistical significance

- **But remember…None of these actually prove causation!**



Confounding variables







Questions?

What to know for HW and midterm

- How to **calculate $\boldsymbol{\beta_0}$ and $\boldsymbol{\beta_1}$**
- How to ***interpret* $\boldsymbol{\beta_0}$ and $\boldsymbol{\beta_1}$** (important!)
- The difference between **MAE and MSE loss**
- How to ***evaluate* a linear regression model**
  - Training *vs.* testing
  - Metrics: R^2, RMSE, standard error, *p*-values