Lecture 14

Introduction to Modeling

EECS 398-003: Practical Data Science, Fall 2024

practicaldsc.org • github.com/practicaldsc/fa24

Announcements 📣

- Midterm Exam scores are available on Gradescope. Solutions are available at **study.practicaldsc.org/fa24-midterm**, and regrade requests are due tomorrow night.
    - If you want to meet with Suraj (or anyone else) to walk through your Midterm Exam and talk about study strategies moving forward, email us!
- Homework 6 is due **tonight**. See **#180 on Ed** for a clarification.

  If you get rate limited in Question 4, you can change your model from `llama-8b-8192` to another model listed **here**.

Agenda

- Course overview.
- Machine learning and models.
- The constant model.
- Minimizing mean squared error using calculus.
- Another loss function.

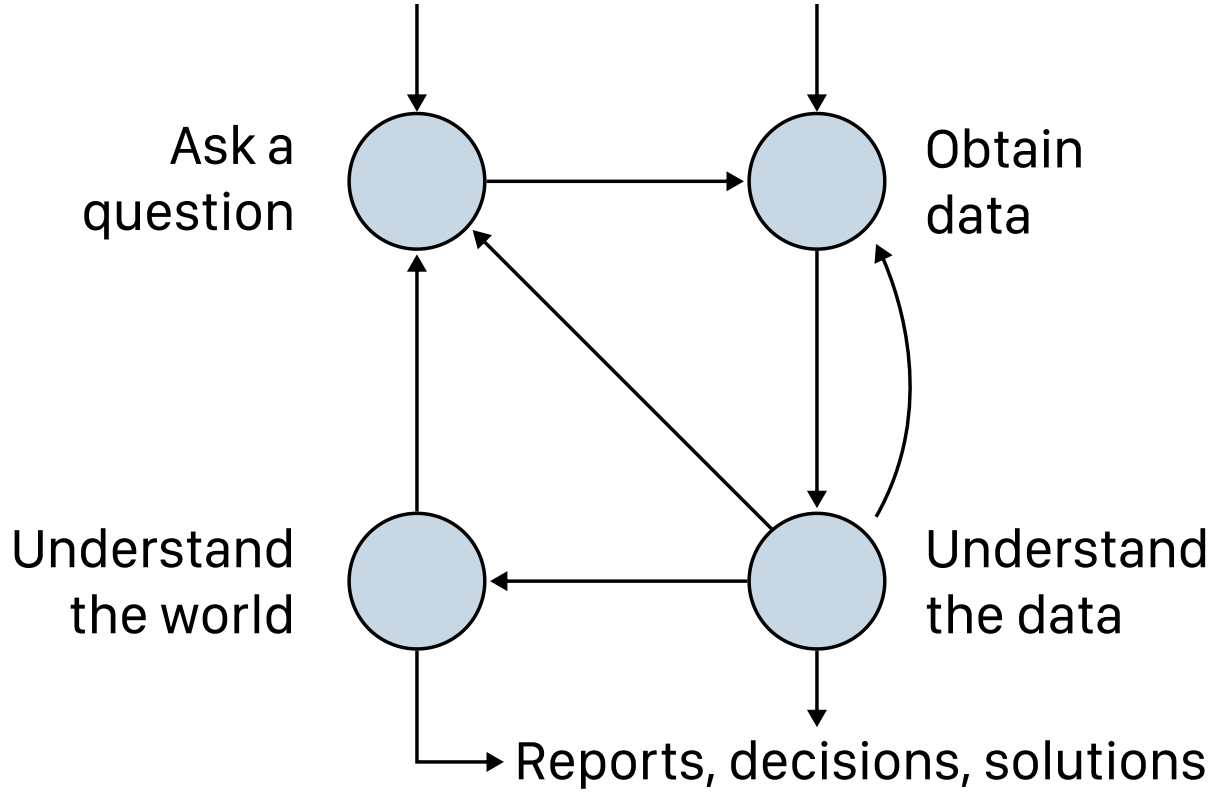
The next few lectures (and next homework!) will be primarily **math-based**.

- For these lectures, we'll post blank slides as a PDF before class, and annotated slides after class.
- If there are any code demos, we'll post those before class, too.
- **Come to discussion tomorrow for math review and practice!**

Course overview

EECS 398, Part 1

- In Lecture 1, we said the first half of this course would be about **data wrangling**.
    - Week 1: Python and Jupyter Notebooks.
    - Weeks 2-3: `numpy`, `pandas`, and Exploratory Data Analysis.
    - Weeks 4-5: Missing Data; Web Scraping and APIs.
    - Weeks 5-6: Text Processing.
    - Week 7: **Midterm Exam**.
- You're proficient in working with messy data using industry-standard tools.
- But, you've also developed an understanding of **how** processes work under the hood.
    - How many rows result from an inner join of these two DataFrames?
    - How do we quantify how important a term is to a document?
    - How do we describe a particular class of strings using mathematical notation?



The **data science lifecycle**.

EECS 398, Part 2

- The second half of this course is about **applied machine learning**.
  - Weeks 8-10: Linear Regression through Linear Algebra.
  - Weeks 11-12: Generalization, Regularization, and Cross-Validation.
  - Weeks 12-14: Gradient Descent and Logistic Regression.
  - Weeks 15-16: Unsupervised Learning, **Final Exam**.
- More generally, we'll learn how to practically build models that can make **predictions about future data** given some data from the past.
- To do so, we'll also learn the **mathematical underpinnings** of these models, so that you understand how they actually work.

  The tools may change in the future, but the ideas won't!

Machine learning and models

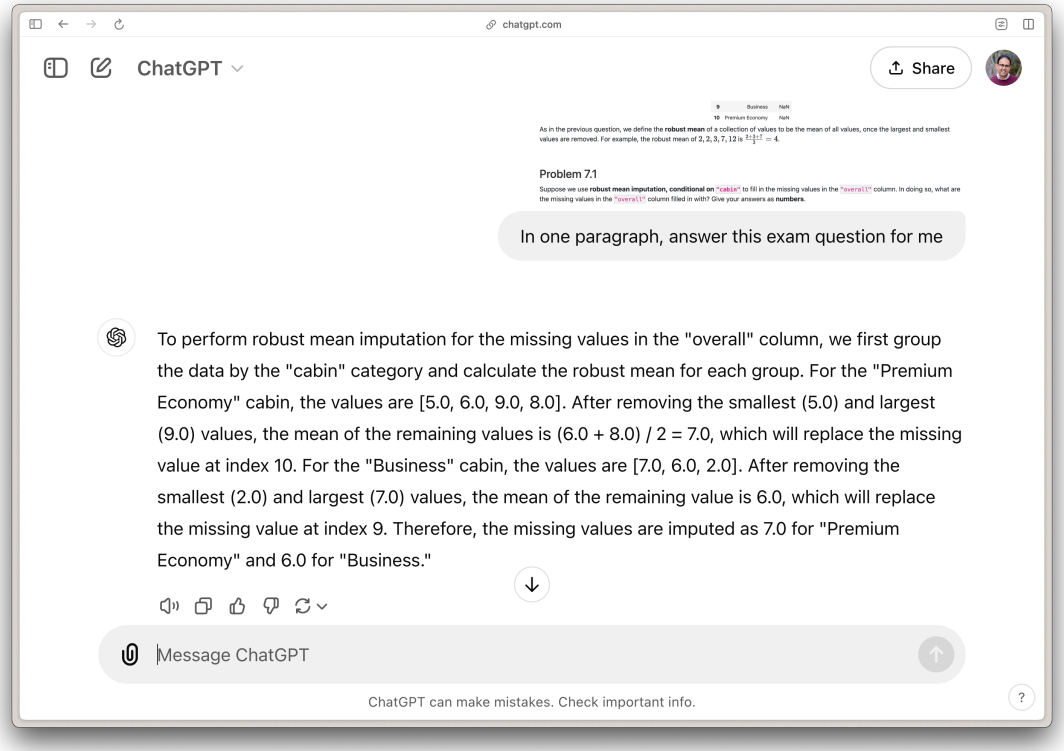
Machine learning is about **automatically** learning patterns from data.

Example: Handwritten digit classification

Humans are good at understanding handwriting,

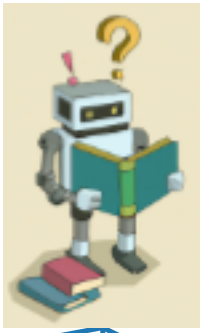
but how do we get computers to understand handwriting?

Example: ChatGPT



How did ChatGPT know how to answer Question 7 from the Fall 2024 Midterm?

Taxonomy of machine learning



Labeled data

Unlabeled data

Reward

Supervised learning

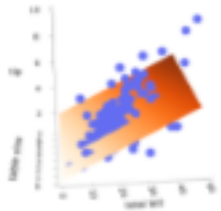
Unsupervised learning

Reinforcement learning

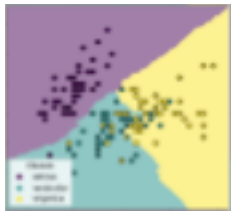
Quantitative prediction

Categorical prediction

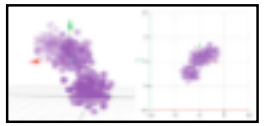
Regression



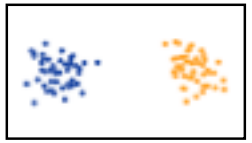
Classification

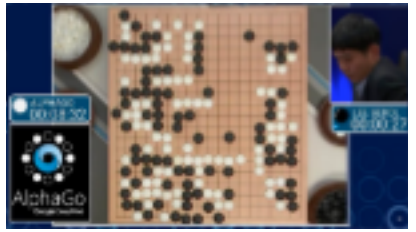


Dimensionality reduction



Clustering

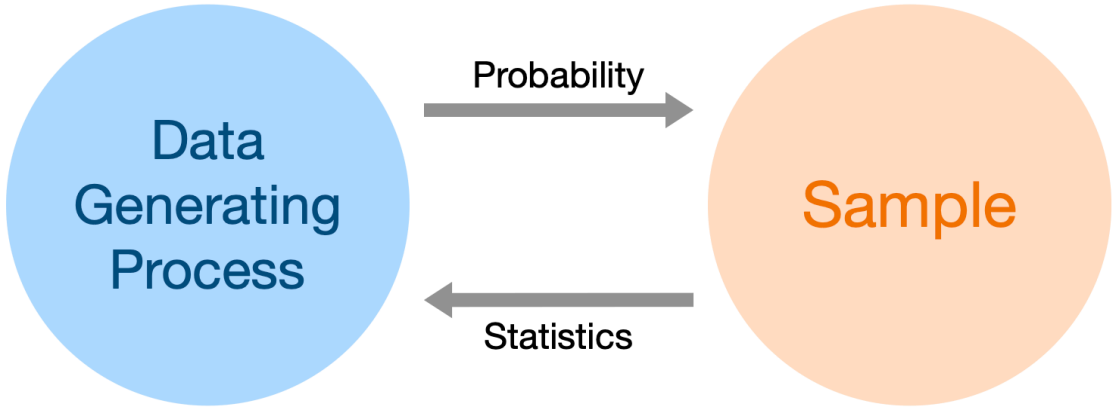




AlphaGo

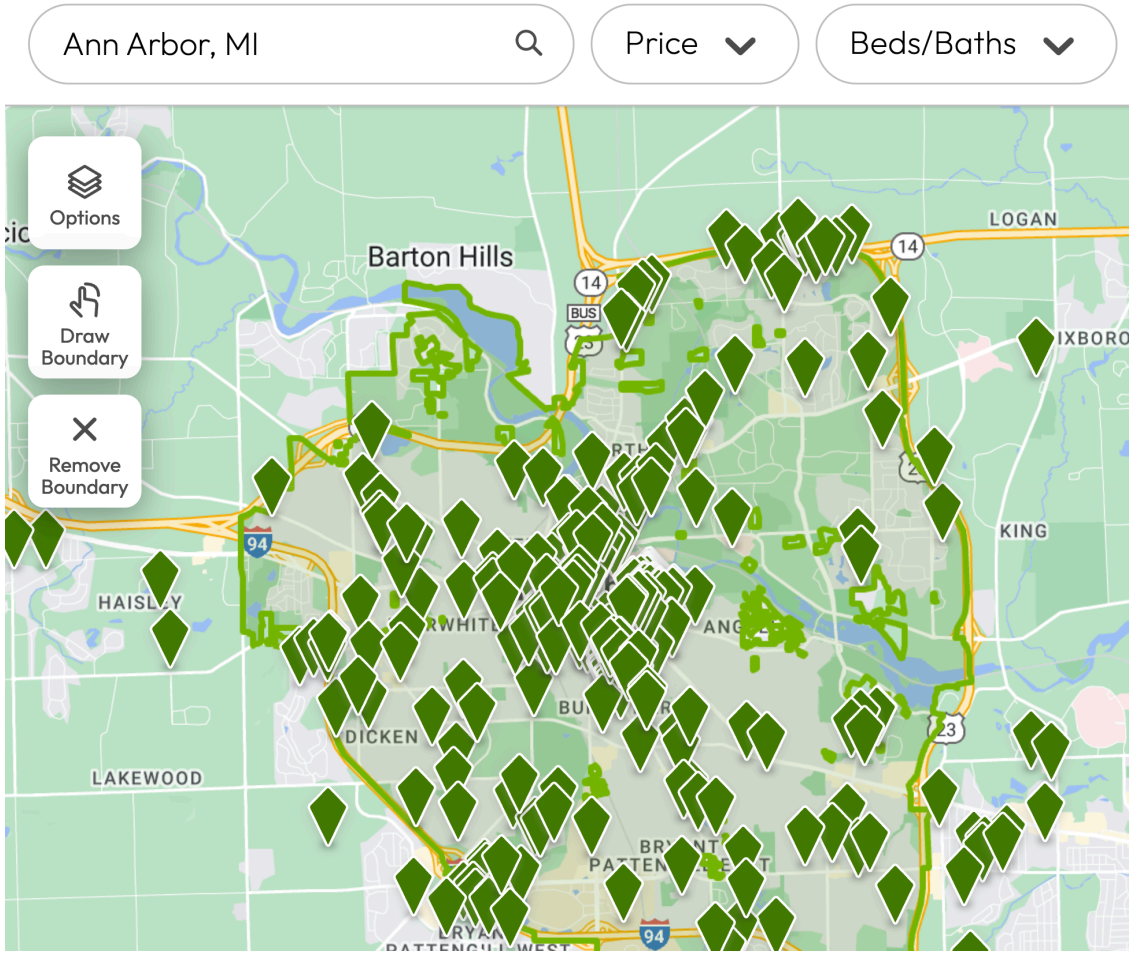
A biased coin flips heads with probability 0.8.

I flip it 100 times. What's the probability of seeing 65 heads?



I found a coin on the ground, and don't know if it's fair.

I flip it 100 times and see 65 heads. What is the most likely bias of the coin?



You might be starting to look for off-campus apartments for next year, none of which are in your price range.

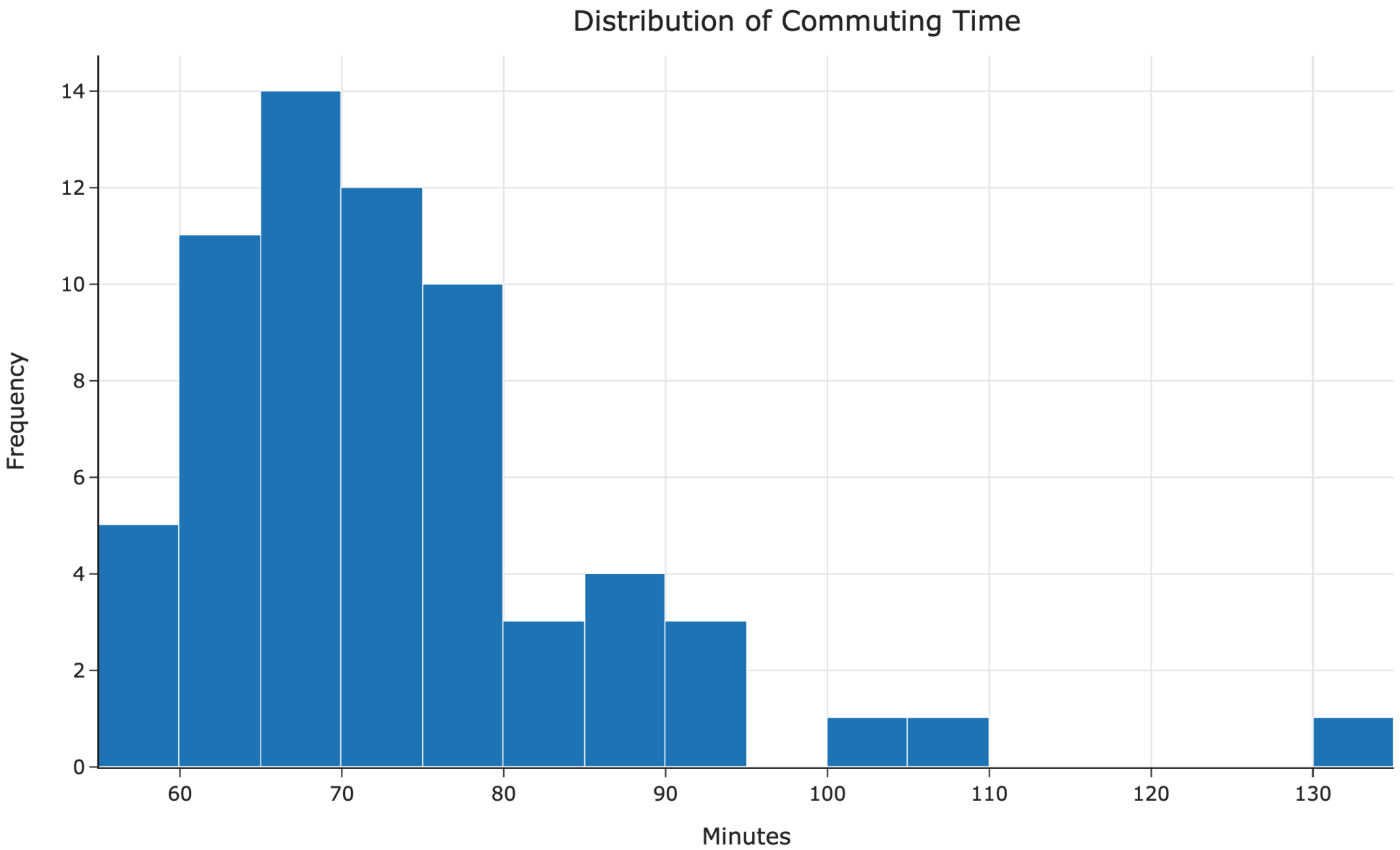
	date	day	departure_hour	minutes
0	5/22/2023	Mon	8.450000	63.0
1	9/18/2023	Mon	7.950000	75.0
2	10/17/2023	Tue	10.466667	59.0
3	11/28/2023	Tue	8.900000	89.0
4	2/15/2024	Thu	8.083333	69.0

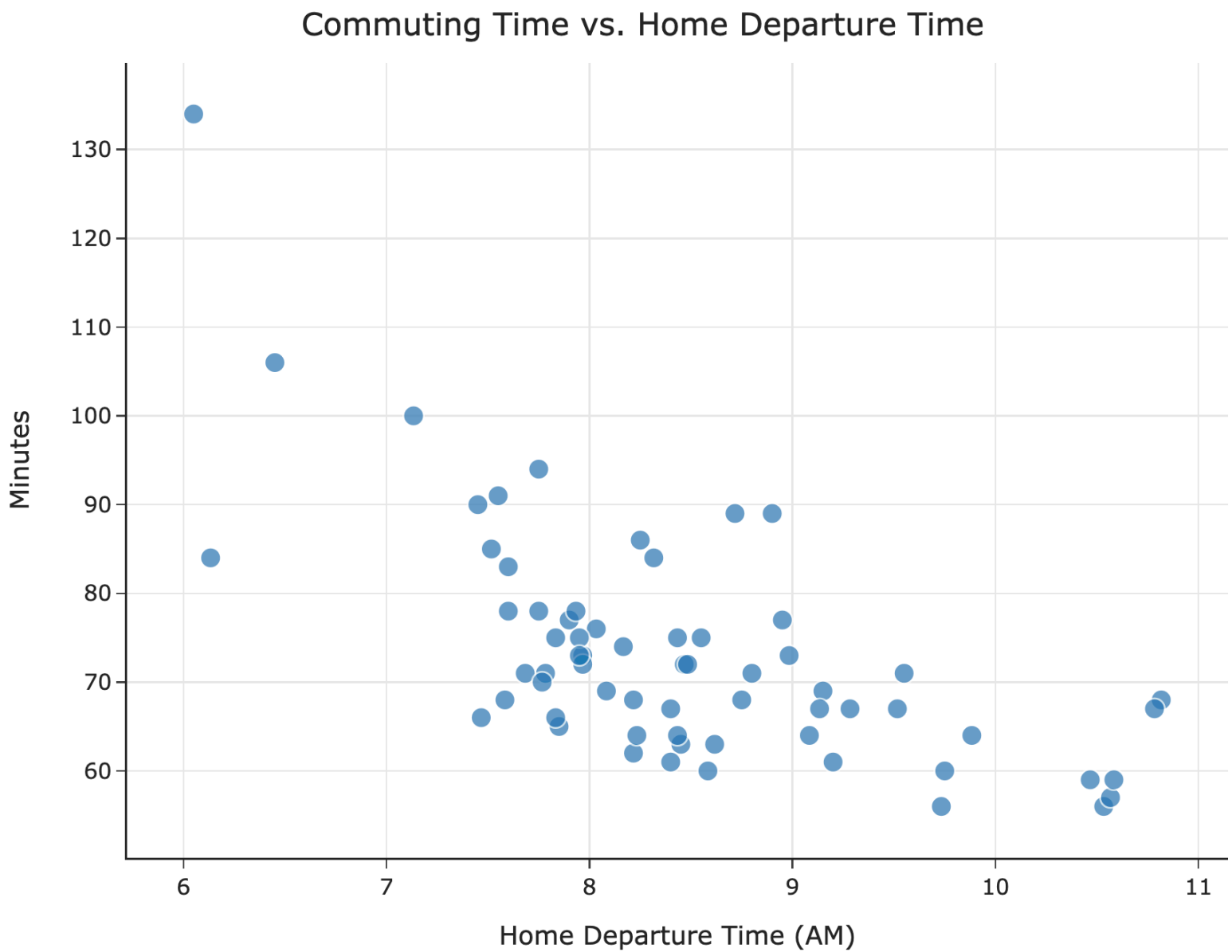
...

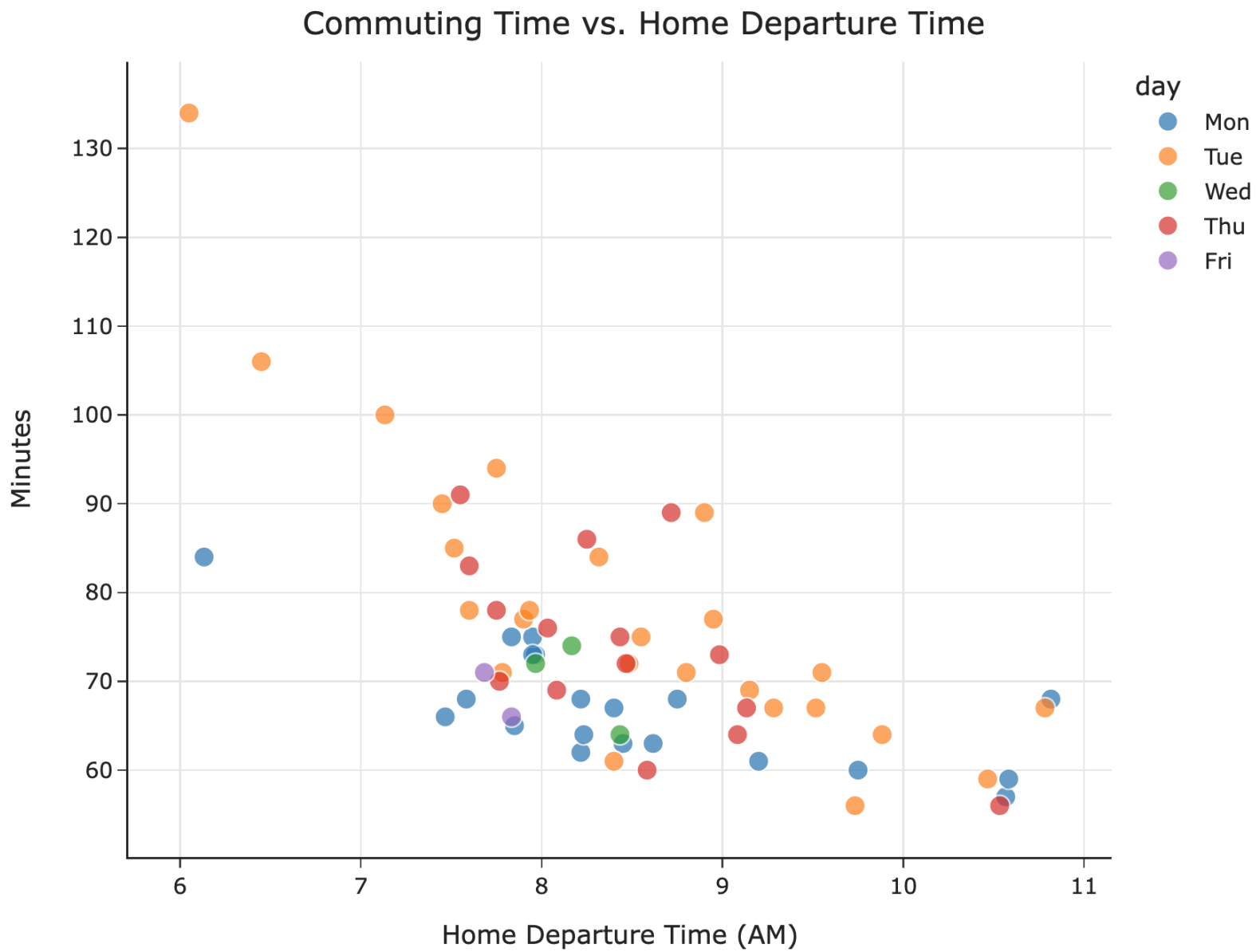
You decide to live with your parents in Detroit and commute.

You keep track of how long it takes you to get to school each day.

This is a real dataset, collected by Joseph Hearn! However, he lived in the Seattle area, not Metro Detroit.


Distribution of Commuting Time
Frequency
0
2
4
6
8
10
12
14
60
70
80
90
100
110
120
130
Minutes


Commuting Time vs. Home Departure Time
Minutes
130
120
110
100
90
80
70
60
6
7
8
9
10
11
Home Departure Time (AM)


Commuting Time vs. Home Departure Time
Minutes
130
120
110
100
90
80
70
60
6
7
8
9
10
11
Home Departure Time (AM)
day
Mon
Tue
Wed
Thu
Fri

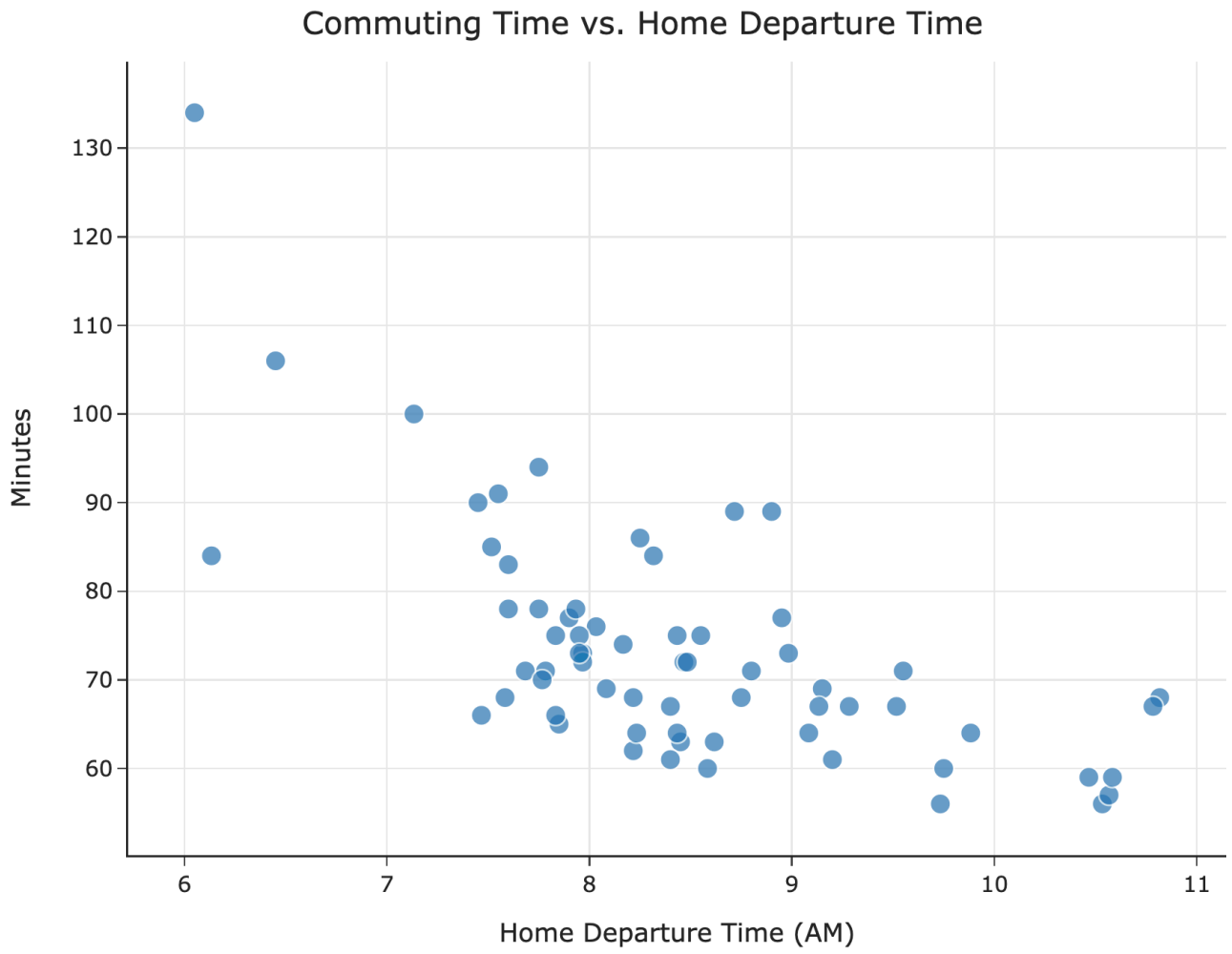
Goal: Predict your **commute time**, i.e. how long it will take to get to school.

This is a **regression** problem.

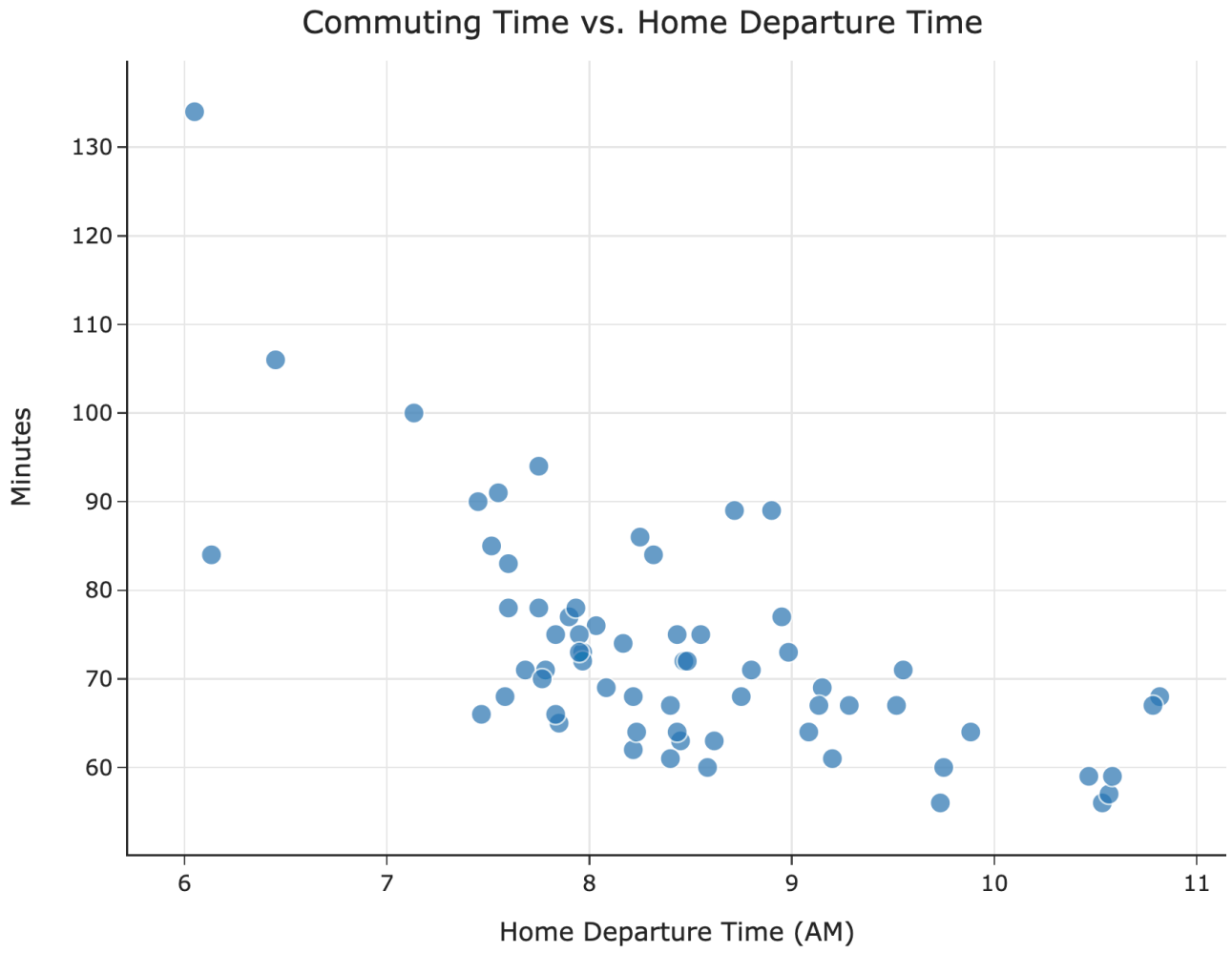
How can we do this? What will we need to assume?

A **model** is a set of assumptions about how data were generated.

Possible models



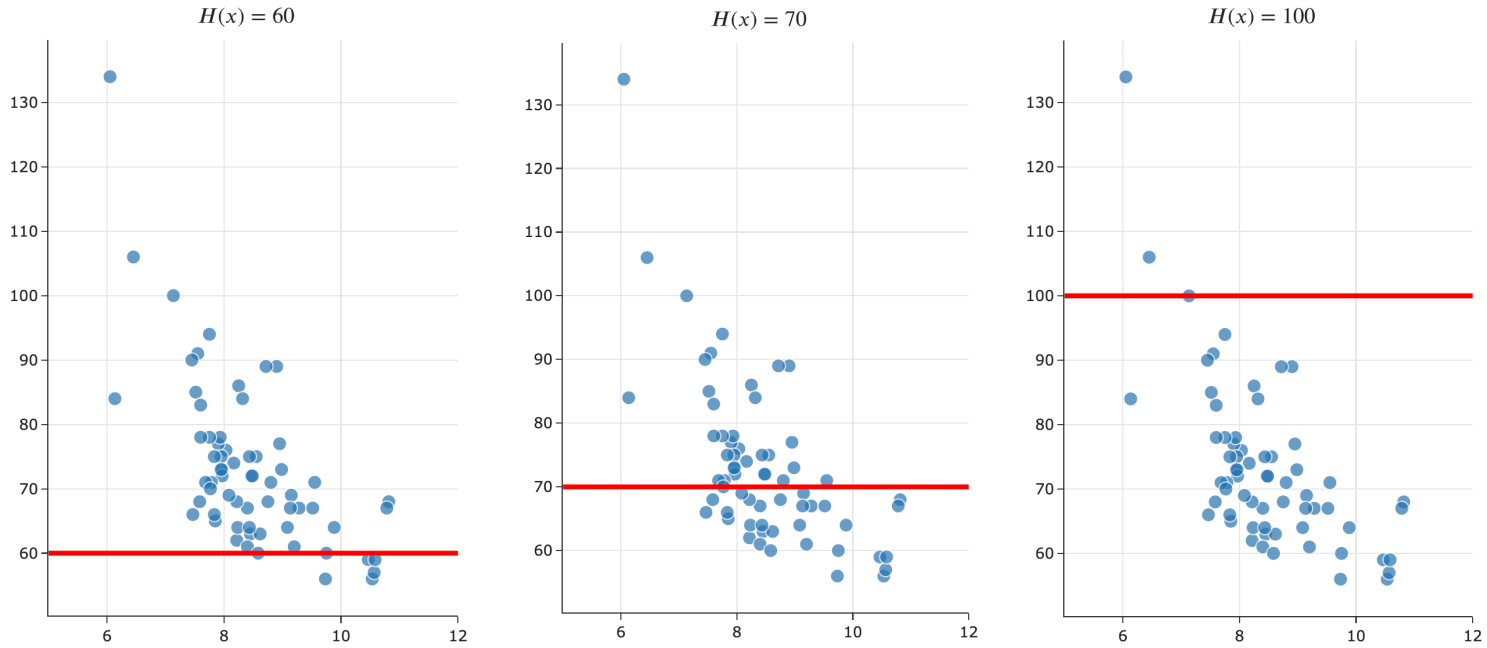
Notation



- x: "input", "independent variable", or "feature".
- y: "response", "dependent variable", or "target".
- **We use x to predict y.**
- The ith observation is denoted (x_i, y_i).

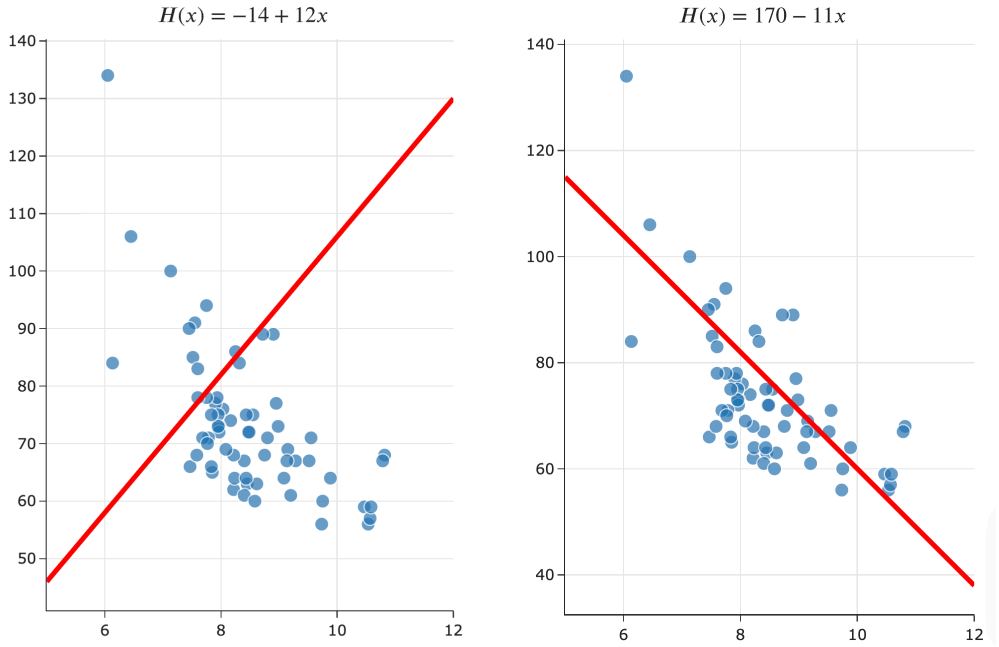
Hypothesis functions and parameters

- A hypothesis function, H, takes in an x as input and returns a predicted y.
- **Parameters** define the relationship between the input and output of a hypothesis function.
- **Example**: The constant model, $H(x) = h$, has one parameter: h.



Hypothesis functions and parameters

- A hypothesis function, H, takes in an x as input and returns a predicted y.
- **Parameters** define the relationship between the input and output of a hypothesis function.
- **Example**: The simple linear regression model, $H(x) = w_0 + w_1 x$, has two parameters: w_0 and w_1.



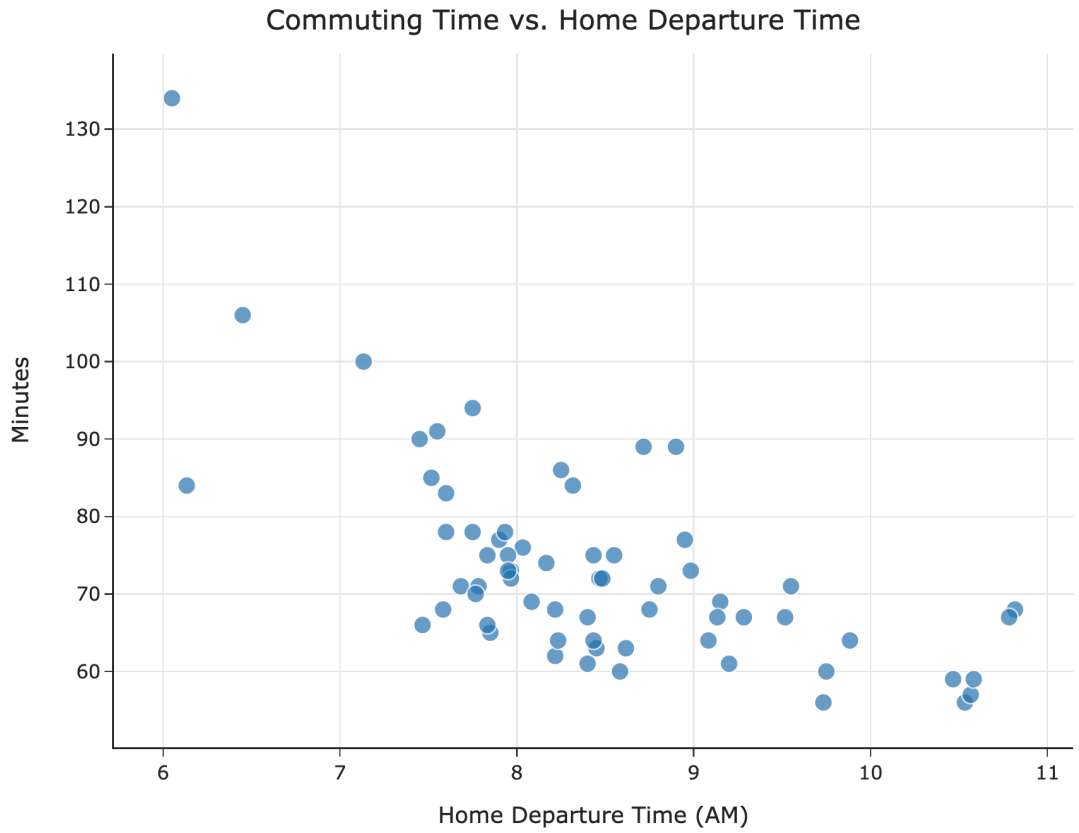
Question 🤔

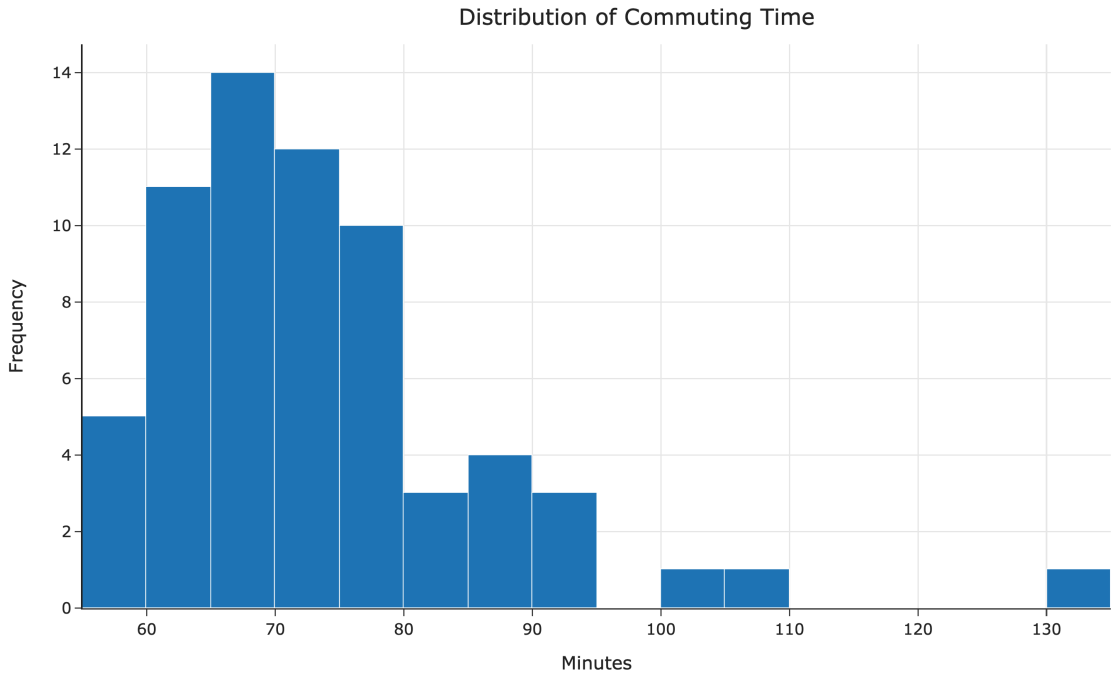
Answer at practicaldsc.org/q

What questions do you have?

The constant model

The constant model





A concrete example

- Let's suppose we have just a smaller dataset of just five historical commute times in minutes.

$$
\begin{aligned}
y_1 &= 72 \\
y_2 &= 90 \\
y_3 &= 61 \\
y_4 &= 85 \\
y_5 &= 92
\end{aligned}
$$

- Given this data, can you come up with a prediction for your future commute time? How?

Some common approaches

- The **mean**:

$$\frac{1}{5}(72 + 90 + 61 + 85 + 92) = \boxed{80}$$

- The **median**:

$$61 \quad 72 \quad \boxed{85} \quad 90 \quad 92$$

- Both of these are familiar **summary statistics**.

  Summary statistics summarize a collection of numbers with a single number, i.e. they result from an **aggregation**.

- But which one is better? Is there a "best" prediction we can make?

The cost of making predictions

- A **loss function** quantifies how bad a prediction is for a single data point.
    - If our prediction is **close** to the actual value, we should have **low** loss.
    - If our prediction is **far** from the actual value, we should have **high** loss.
- A good starting point is error, which is the difference between **actual** and **predicted** values.

$$e_i = y_i - H(x_i)$$

- Suppose my commute **actually** takes 80 minutes.
    - If I predict 75 minutes:
    - If I predict 72 minutes:
    - If I predict 100 minutes:

Squared loss

- One loss function is squared loss, L_{sq}, which computes $(\text{actual} - \text{predicted})^2$.

$$L_{\text{sq}}(y_i, H(x_i)) = (y_i - H(x_i))^2$$

- Note that for the constant model, $H(x_i) = h$, so we can simplify this to:

$$L_{\text{sq}}(y_i, h) = (y_i - h)^2$$

- Squared loss is not the only loss function that exists!

  Soon, we'll learn about absolute loss. Different loss functions have different pros and cons.

A concrete example, revisited

- Consider again our smaller dataset of just five historical commute times in minutes.

  $y_1 = 72$
  $y_2 = 90$
  $y_3 = 61$
  $y_4 = 85$
  $y_5 = 92$

- Suppose we predict the median, $h = 85$. What is the squared loss of 85 for each data point?

Averaging squared losses

- We'd like a single number that describes the quality of our predictions across our entire dataset. One way to compute this is as the **average of the squared losses**.
- For the median, $h = 85$:

$$\frac{1}{5}\left((72 - 85)^2 + (90 - 85)^2 + (61 - 85)^2 + (85 - 85)^2 + (92 - 85)^2\right) = \boxed{163.8}$$

- For the mean, $h = 80$:

$$\frac{1}{5}\left((72 - 80)^2 + (90 - 80)^2 + (61 - 80)^2 + (85 - 80)^2 + (92 - 80)^2\right) = \boxed{138.8}$$

- Which prediction is better? Could there be an even better prediction?

Mean squared error

- Another term for **average squared loss** is **mean squared error (MSE)**.
- The mean squared error on our smaller dataset for any prediction h is of the form:

$$R_{\text{sq}}(h) = \frac{1}{5}\left((72 - h)^2 + (90 - h)^2 + (61 - h)^2 + (85 - h)^2 + (92 - h)^2\right)$$

R stands for "risk", as in "empirical risk." We'll see this term again soon.

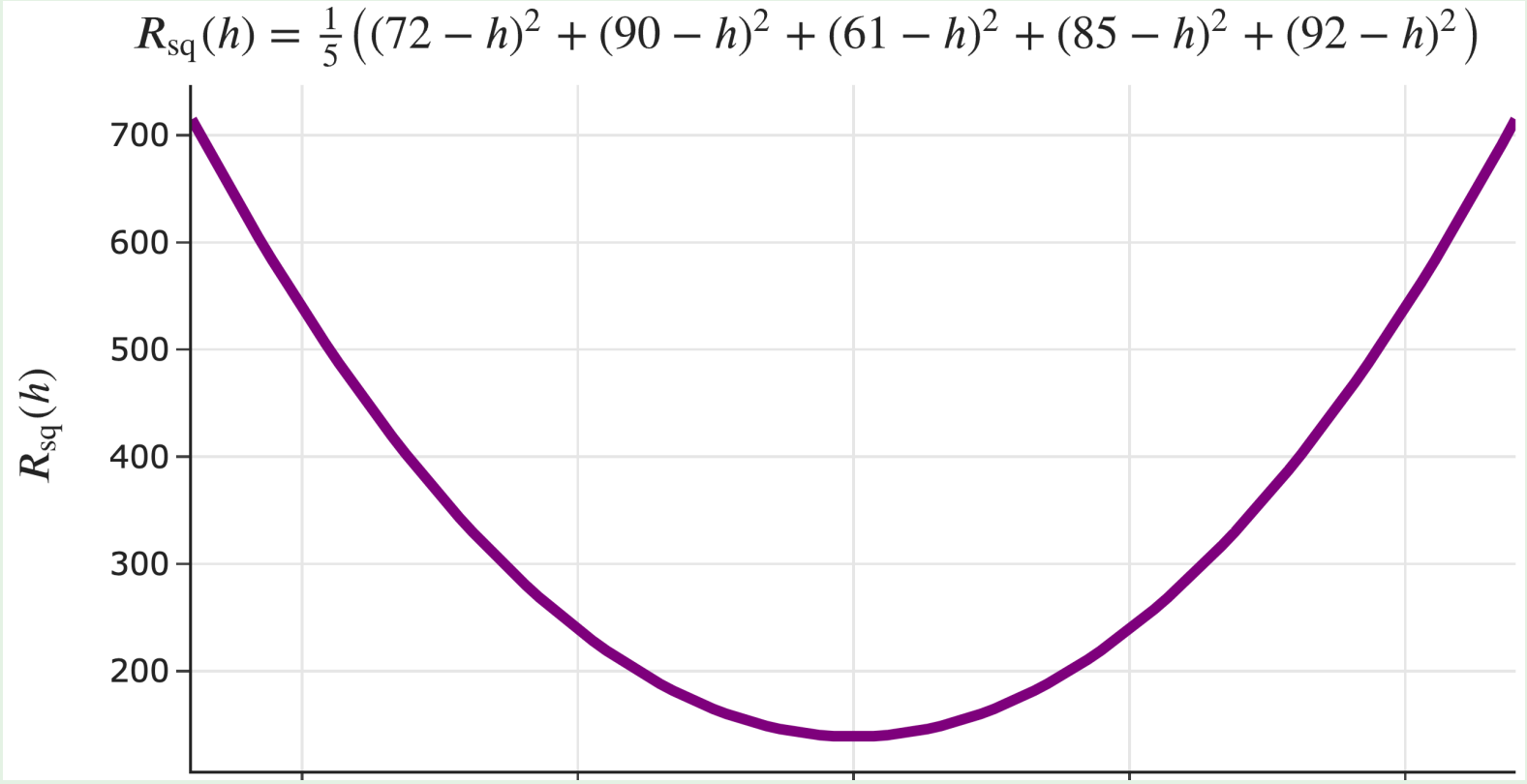
- For example, if we predict $h = 100$, then:

$$\begin{aligned} R_{\text{sq}}(100) &= \frac{1}{5}\left((72 - 100)^2 + (90 - 100)^2 + (61 - 100)^2 + (85 - 100)^2 + (92 - 100)^2\right) \\ &= \boxed{538.8} \end{aligned}$$

- **We can pick any h as a prediction, but the smaller $R_{\text{sq}}(h)$ is, the better h is!**

Activity

Answer at practicaldsc.org/q (use the free response box!)



Which h corresponds to the vertex of $R_{\text{sq}}(h)$?

Mean squared error, in general

- Suppose we collect n commute times, $y_1, y_2, \ldots, y_n$.
- The mean squared error of the prediction h is:

- Or, using **summation notation**:

The best prediction

$$R_{\text{sq}}(h) = \frac{1}{n}\sum_{i=1}^{n}(y_i - h)^2$$

- We want the **best** constant prediction, among all constant predictions h.
- The smaller $R_{\text{sq}}(h)$ is, the better h is.
- **Goal**: Find the h that minimizes $R_{\text{sq}}(h)$.

  The resulting h will be called h^*.
- **How do we find h^*?**

Minimizing mean squared error using calculus

Minimizing using calculus

- We'd like to minimize:

$$R_{\text{sq}}(h) = \frac{1}{n}\sum_{i=1}^{n}(y_i - h)^2$$

- In order to minimize $R_{\text{sq}}(h)$, we:
  1. take its derivative with respect to h,
  2. set it equal to 0,
  3. solve for the resulting h^*, and
  4. perform a second derivative test to ensure we found a minimum.

Step 0: The derivative of $(y_i - h)^2$

- Remember from calculus that:
  - if $c(x) = a(x) + b(x)$, then
  - $\frac{d}{dx}c(x) = \frac{d}{dx}a(x) + \frac{d}{dx}b(x)$.
- This is relevant because $R_{\text{sq}}(h) = \frac{1}{n}\sum_{i=1}^{n}(y_i - h)^2$ involves the sum of n individual terms, each of which involve h.
- So, to take the derivative of $R_{\text{sq}}(h)$, we'll first need to find the derivative of $(y_i - h)^2$.

$\frac{d}{dh}(y_i - h)^2 =$

Question 🤔

Answer at practicaldsc.org/q

$$R_{\text{sq}}(h) = \frac{1}{n}\sum_{i=1}^{n}(y_i - h)^2$$

Which of the following is $\frac{d}{dh}R_{\text{sq}}(h)$?

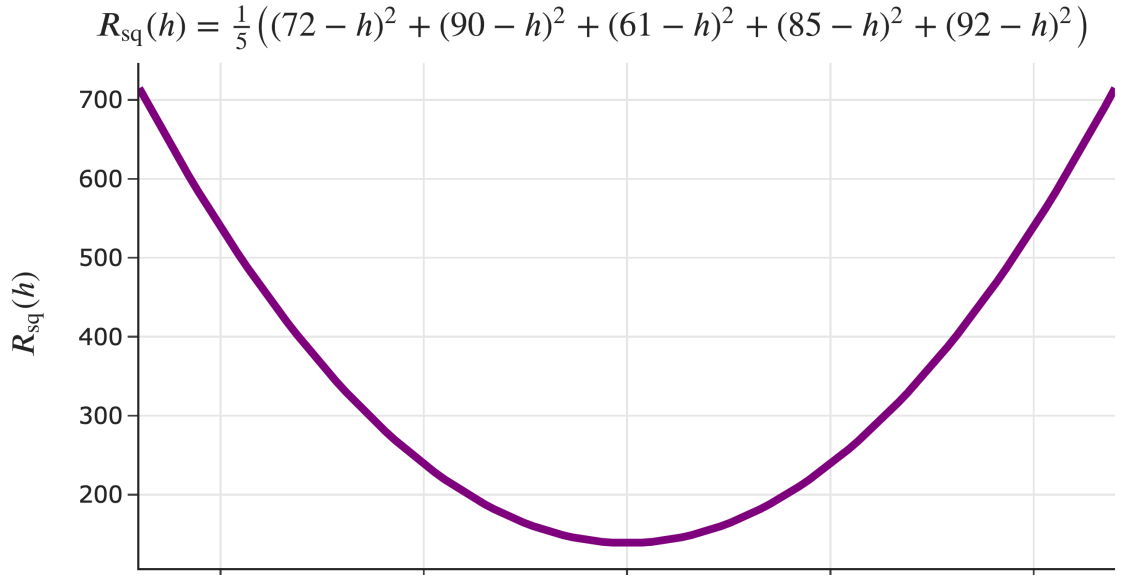
- A. 0
- B. $\sum_{i=1}^{n} y_i$
- C. $\frac{1}{n}\sum_{i=1}^{n}(y_i - h)$
- D. $\frac{2}{n}\sum_{i=1}^{n}(y_i - h)$
- E. $-\frac{2}{n}\sum_{i=1}^{n}(y_i - h)$

Step 1: The derivative of $R_{\text{sq}}(h)$

$$\frac{d}{dh} R_{\text{sq}}(h) = \frac{d}{dh} \left(\frac{1}{n} \sum_{i=1}^{n} (y_i - h)^2 \right)$$

Steps 2 and 3: Set to 0 and solve for the minimizer, h^*

Step 4: Second derivative test



We already saw that $R_{\text{sq}}(h)$ is **convex**, i.e. that it opens upwards, so the h^* we found must be a minimum, not a maximum.

The mean minimizes mean squared error!

- The problem we set out to solve was, find the h^* that minimizes:

$$R_{\text{sq}}(h) = \frac{1}{n} \sum_{i=1}^{n} (y_i - h)^2$$

- The answer is:

$$h^* = \text{Mean}(y_1, y_2, \ldots, y_n)$$

- The **best constant prediction**, in terms of mean squared error, is always the **mean**.
- We call h^* our **optimal model parameter**, for when we use:
  - the constant model, $H(x) = h$, and
  - the squared loss function, $L_{\text{sq}}(y_i, h) = (y_i - h)^2$.

Aside: Notation

- Another way of writing:

$$h^* \text{ is the value of } h \text{ that minimizes } \frac{1}{n}\sum_{i=1}^{n}(y_i - h)^2$$

  is:

$$h^* = \underset{h}{\text{argmin}} \left(\frac{1}{n}\sum_{i=1}^{n}(y_i - h)^2 \right)$$

- h^* is the solution to an **optimization problem**.

The modeling recipe

- We've implicitly introduced a three-step process for finding optimal model parameters (like h^*) that we can use for making predictions:

    1. Choose a model.

    2. Choose a loss function.

    3. Minimize average loss to find optimal model parameters.

- Most modern machine learning methods today, including neural networks, follow this recipe, and we'll see it repeatedly this semester!

Question 🤔

Answer at practicaldsc.org/q

What questions do you have?

Another loss function

Another loss function

- We started by computing the **error** for each of our predictions, but ran into the issue that some errors were positive and some were negative.

$$e_i = y_i - H(x_i)$$

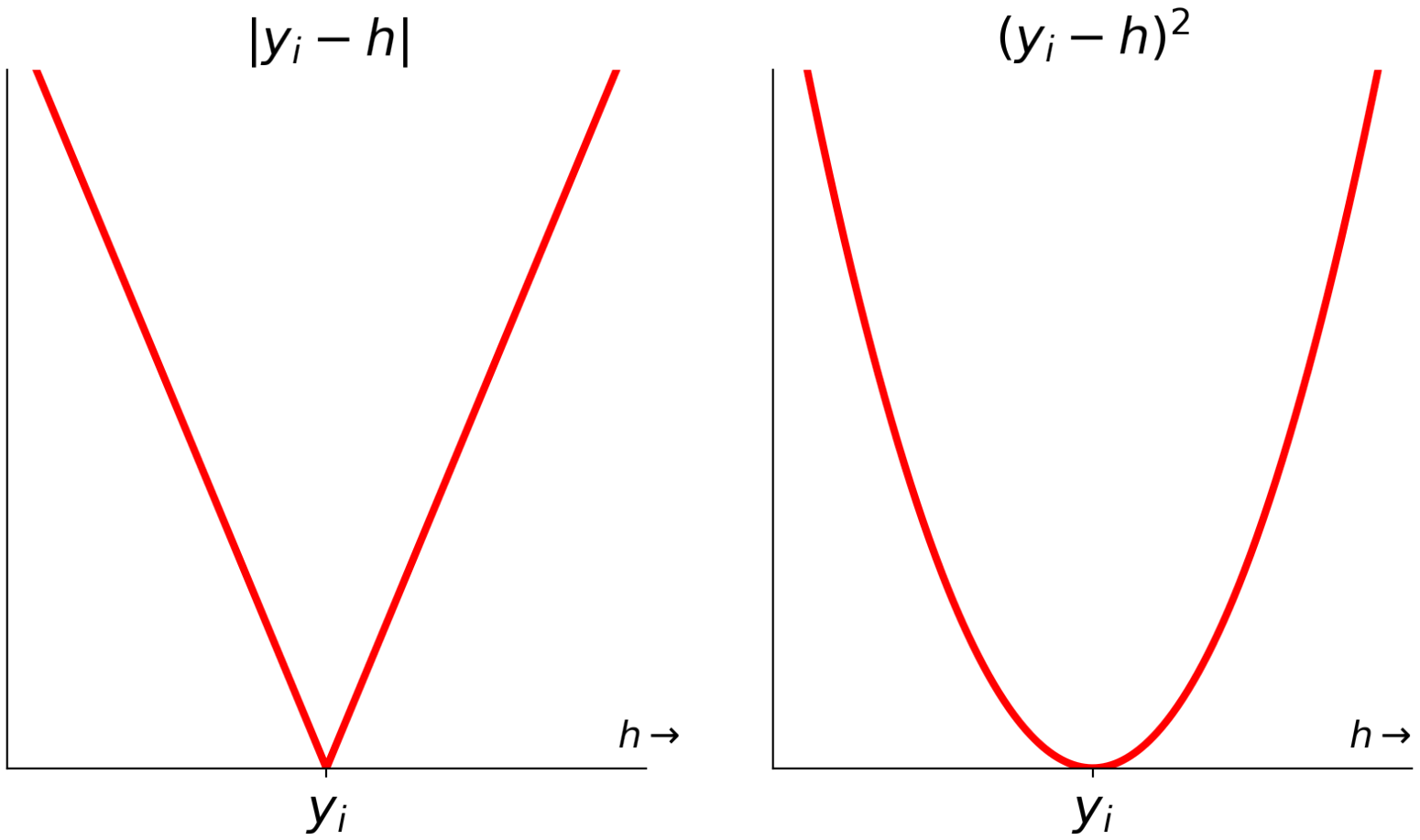
- The solution was to **square** the errors, so that all are non-negative. The resulting loss function is called **squared loss**.

$$L_{\text{sq}}(y_i, H(x_i)) = (y_i - H(x_i))^2$$

- Another loss function, which also measures how far $H(x_i)$ is from y_i, is **absolute loss**.

$$L_{\text{abs}}(y_i, H(x_i)) = |y_i - H(x_i)|$$

Absolute loss vs. squared loss



Mean absolute error

- Suppose we collect n commute times, $y_1, y_2, \ldots, y_n$.
- The **average absolute loss**, or **mean absolute error (MAE)**, of the prediction h is:

$$R_{\text{abs}}(h) = \frac{1}{n} \sum_{i=1}^{n} |y_i - h|$$

- We'd like to find the best constant prediction, h^*, by finding the h that minimizes **mean absolute error**.
- Any guesses?

The median minimizes mean absolute error!

- It turns out that the constant prediction h^* that minimizes mean absolute error,

$$R_{\text{abs}}(h) = \frac{1}{n} \sum_{i=1}^{n} |y_i - h|$$

  is:

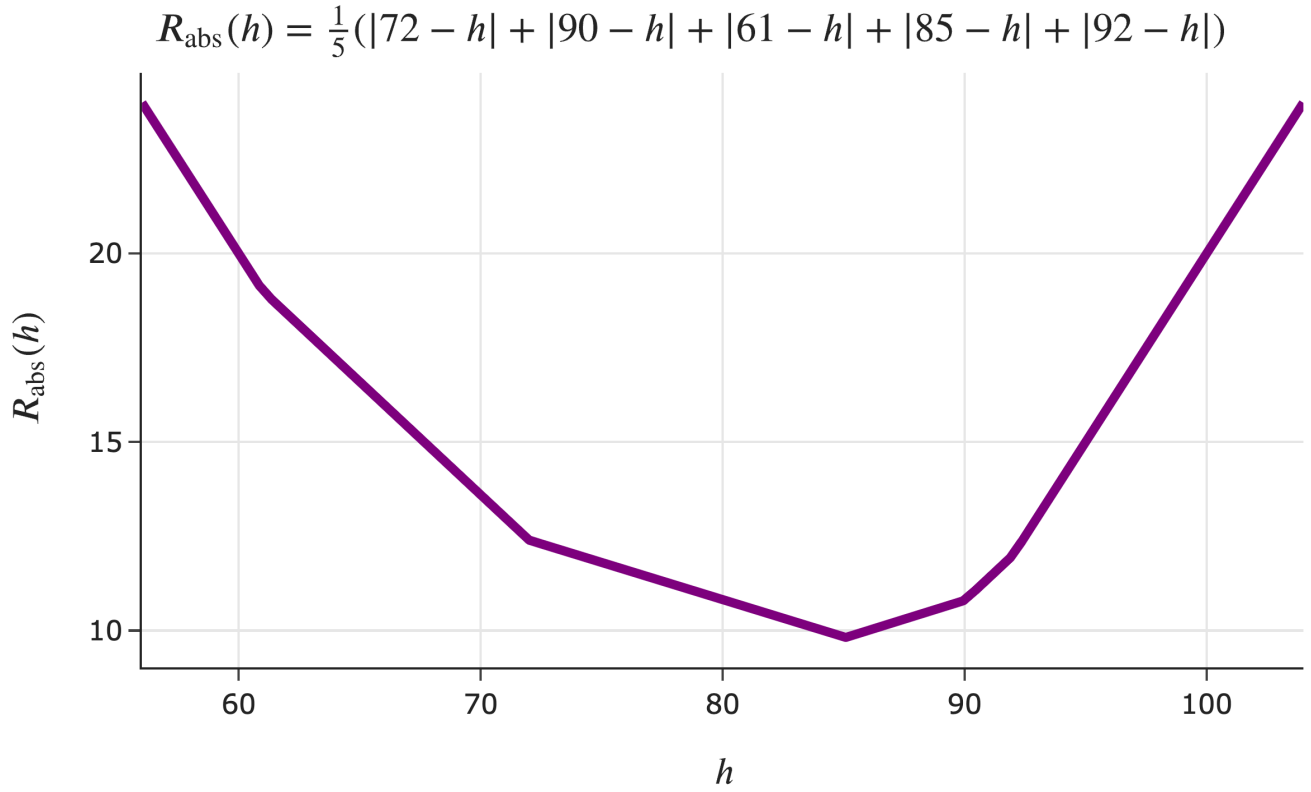
$$h^* = \text{Median}(y_1, y_2, \ldots, y_n)$$

- We won't prove this in lecture, but **this extra video** walks through it.

  Watch it!

- To make a bit more sense of this result, let's graph $R_{\text{abs}}(h)$.

Visualizing mean absolute error

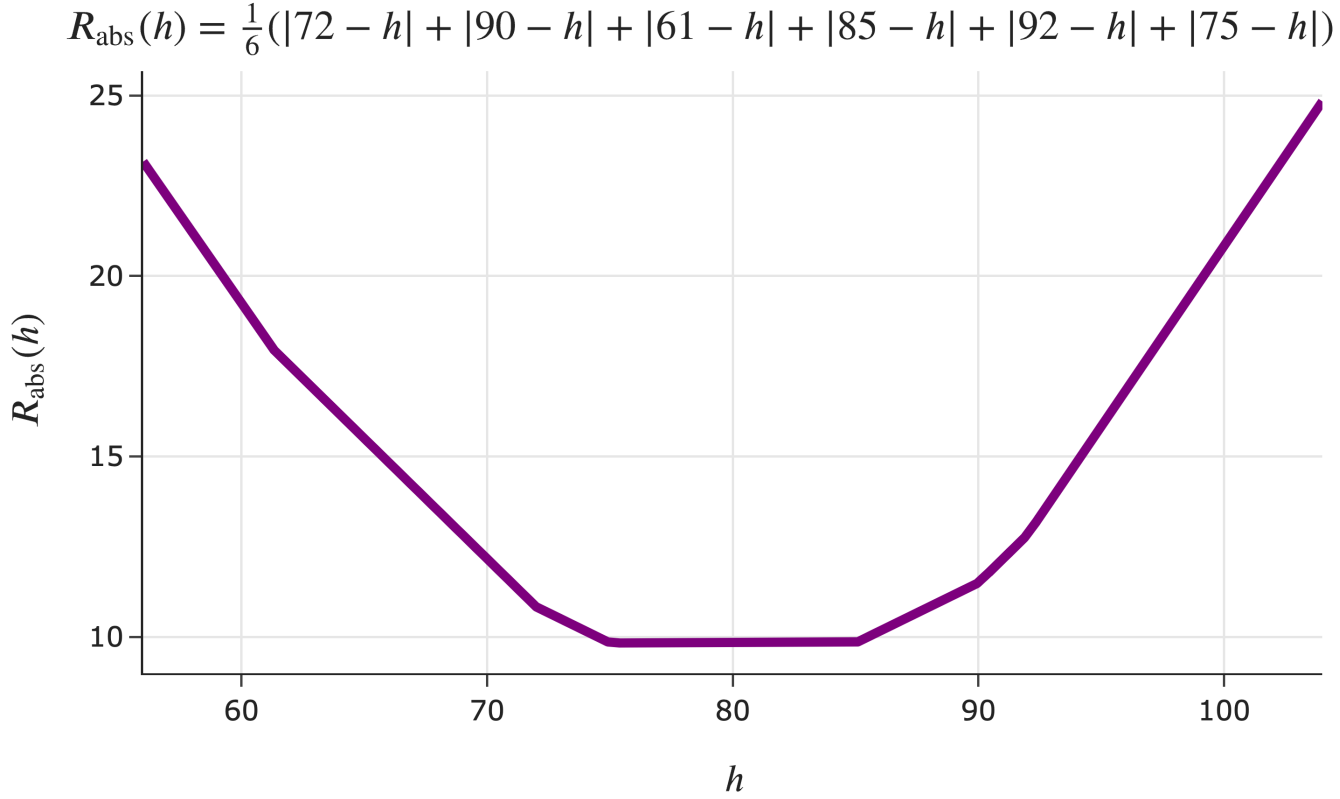


- Consider, again, our example dataset of five commute times.

$$72, 90, 61, 85, 92$$

- Where are the "bends" in the graph of $R_{\text{abs}}(h)$ – that is, where does its slope change?

Visualizing mean absolute error, with an even number of points



- What if we add a sixth data point?

  $72, 90, 61, 85, 92, 75$

- Is there a unique h^*?

The median minimizes mean absolute error!

- The new problem we set out to solve was, find the h^* that minimizes:

$$R_{\text{abs}}(h) = \frac{1}{n} \sum_{i=1}^{n} |y_i - h|$$

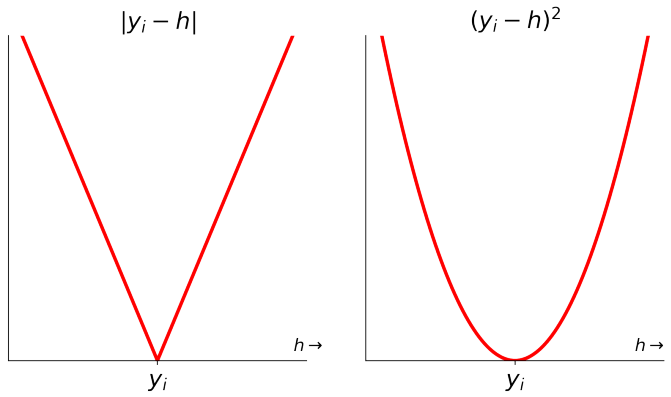
- The answer is:

$$h^* = \text{Median}(y_1, y_2, \ldots, y_n)$$

- The **best constant prediction**, in terms of mean absolute error, is always the **median**.
  - When n is odd, this answer is unique.
  - When n is even, any number between the middle two data points (when sorted) also minimizes mean absolute error.
  - When n is even, define the median to be the mean of the middle two data points.

Choosing a loss function

- For the constant model $H(x) = h$, the **mean** minimizes mean **squared** error.
- For the constant model $H(x) = h$, the **median** minimizes mean **absolute** error.
- In practice, squared loss is the more common choice, as it's easily **differentiable**.



- But how does our choice of loss function impact the resulting optimal prediction?
- We'll discuss this more next class.

Question 🤔

Answer at practicaldsc.org/q

What questions do you have?