

# Intro and problem definition



*10-701 Introduction to Machine Learning  
Geoff Gordon and Pradeep Ravikumar*

# ***Logistics***

- <https://www.cs.cmu.edu/~10701-f25/>
  - ▶ This whole website is required reading
- Highlights:
  - ▶ Tech: Piazza, Gradescope, Canvas
  - ▶ Grades: homeworks (written and coding parts), homework quizzes (after due date, check understanding), participation (small questions, 48 hrs to complete), midterm & final exams, class project
  - ▶ Prereqs: probability, coding, calculus, reasoning/proofs
  - ▶ Grace days for homeworks: no need to ask us about small illnesses, trips, etc.
    - applied greedily, limit on total lateness
    - separate from major problems (e.g., longer-lasting illness)

# ***Collaboration***

- Is encouraged and can support learning!
- But write up solutions ***on your own***
  - ▶ don't discuss full concrete solutions, just strategies and ideas
  - ▶ maybe best not to take detailed notes during discussions, but if you have notes, put them away, wait 10 minutes before writing anything
  - ▶ ensures solution isn't coming from short-term memory
- Make sure to ***protect your own work***
- And ***disclose all collaborators and sources***
- We do check, and we regularly find problems — see syllabus about academic integrity violations

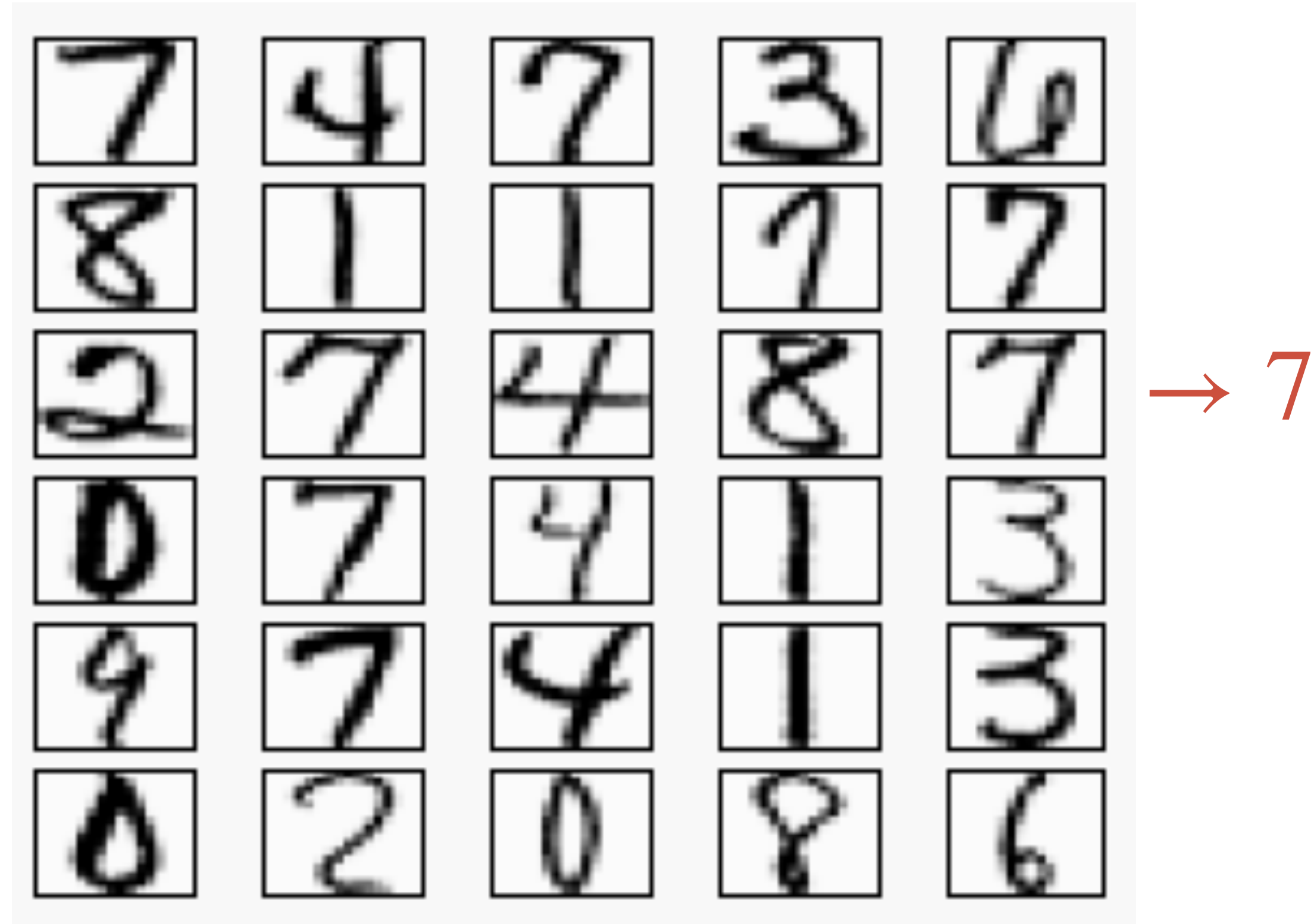
# *GenAI*

- You are encouraged to use GenAI to improve learning!
  - ▶ e.g., find literature, generate practice problems, discuss strategy — let us know if you find an interesting and helpful study practice!
- All work that you submit must be your own
- Treat GenAI as if it were another student:
  - ▶ interactions should support learning
  - ▶ do not discuss complete solutions
  - ▶ best not to take written notes during discussion
  - ▶ all notes should be closed while you are writing solutions
  - ▶ this includes turning off large-scale code completion: you will have trouble with homework quizzes if you yourself don't learn what the libraries do (and later you will have trouble with understanding and debugging code)

# *What is ML?*

- Code that gets better at some *task* by taking advantage of some kind of *experience*
  - ▶ Task is formalized with a performance *metric*
  - ▶ Experience is formalized with *data*

*For example*



- Task: read handwritten ZIP code on mail
- Experience: mail that has been correctly routed
- Becomes:
  - ▶ data: images of individual digits w/ corresponding labels
  - ▶ metric: per-digit accuracy (%)

## *Another example*

- Learning whether to approve a loan or line of credit
  - ▶ Task:
  - ▶ Experience:
  - ▶ Data:
  - ▶ Metric:

# ***No single answer***

- Often the same/similar ML problem can be formulated several ways
  - ▶ previous slide's metric: accuracy of approval decision (classification)
  - ▶ could also predict a numerical credit score (regression)
  - ▶ or distribution of total amount repaid (conditional density estimation)
  - ▶ or each loan payment over time (sequence learning)
  - ▶ ...

*No single answer*

- Often the same/similar ML problem can be formulated several ways

	<b>Output type</b>	<b>Name</b>
▶ prev (clas	boolean	Binary classification
▶ coul	categorical	Multiclass classification
▶ or di	real	Regression
estir	ordering	Ranking
▶ or ea	multiple related	Structured prediction (e.g., graphical models)
▶ ...	multiple over time	Sequence learning
	policy / control	Reinforcement learning
	...	...

sion  
ssion)  
density  
ng)

***Self-driving  
car***



***What is our  
experience?  
How do we  
turn it into  
data?***





***What is our task? How do we measure performance?***





*this one is a freebie: no penalty if you didn't bring a device*

## ***Participation question***

- Define your own learning problem (any one you find interesting) in terms of  $T, E, D, P$

***Your well-posed ML problems***

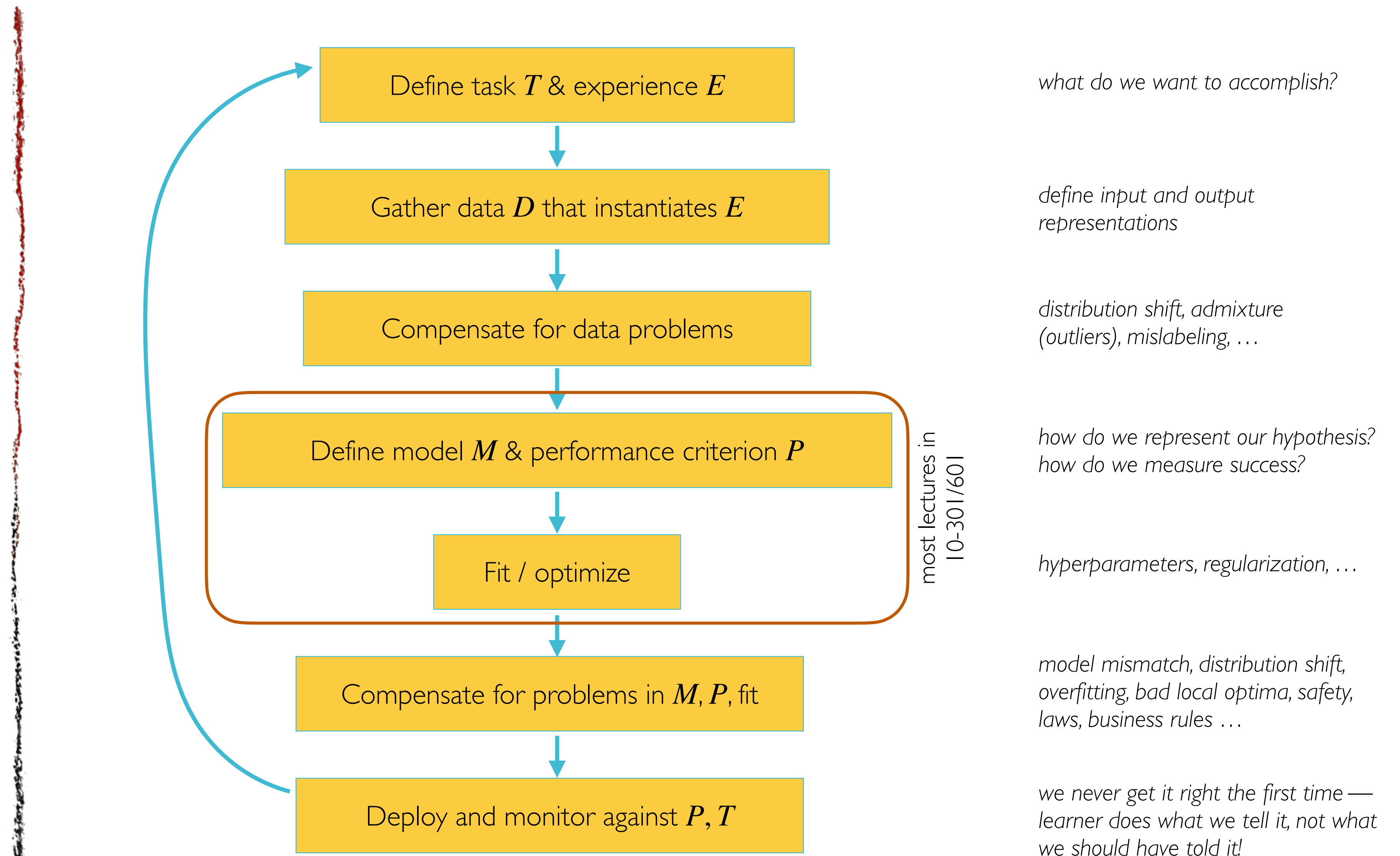
Task

Experience

Performance metric

Data

# ML pipeline



# ***A brief history of machine learning***

- Early Foundations (1940s–1960s)
  - ▶ 1957: Rosenblatt develops the perceptron, an early neural network for classification
- Symbolic AI & the First AI Winter (1970s–1980s)
  - ▶ Limitations of perceptrons (Minsky & Papert, 1969) and lack of computing power lead to skepticism and reduced funding
- Statistical & Algorithmic Advances (1980s–1990s)
  - ▶ 1986: Rumelhart, Hinton & Williams popularize backpropagation, enabling multi-layer neural networks to learn
  - ▶ 1980s–90s: Emergence of support vector machines (SVMs), decision trees, boosting (AdaBoost), Bayesian methods, Reinforcement learning

# ***A brief history of machine learning***

- Second AI Winter, Rise of Data (1990s–2000s)
  - ▶ Disillusionment with difficulty of training deep networks and problems with scaling RL, eventually overcome by...
  - ▶ Explosion of digital data + faster computing power
  - ▶ Kernel methods, ensemble methods come into their own
- Deep Learning Revolution (2010s)
  - ▶ 2012: AlexNet (Krizhevsky, Sutskever, Hinton) wins ImageNet competition w/ GPU + deep CNNs, igniting deep learning boom
  - ▶ Reinforcement learning breakthroughs (e.g., AlphaGo in 2016)
- Foundation Models & Generative AI (2020s–present)
  - ▶ Rise of transformers (Vaswani et al., 2017) revolutionizes NLP (BERT, GPT)
  - ▶ Emergence of foundation models trained on massive datasets for general-purpose use
  - ▶ Policy, ethics, and responsible AI practices gain prominence due to societal impacts

# *Classical and modern models*

- Huge amount of change over time in the types of problems and types of models that people focus on
- But, some ideas and techniques span the whole history
  - ▶ how to set up problems, useful mathematical ideas, relationship to optimization, even just overall ML mindset
- One of our goals this course: expose these common ideas so that you can develop new models & techniques and be part of whatever the next revolution is
  - ▶ simple example: linear models vs. last layer of deep net

# *Techniques*

- Some of the common techniques:
  - ▶ linear algebra
  - ▶ differential calculus
  - ▶ optimization (e.g., SGD)
  - ▶ hypothesis classes, complexity, regularization
  - ▶ tensor manipulation
  - ▶ function spaces
  - ▶ ...

***Driving  
(some of  
the)  
differences:  
how much  
data,  
compute***

- Data:
  - ▶ deep nets can get very high performance with big data, can lose badly to classical methods like feature engineering + logistic regression if we have small data
  - ▶ kernel methods are intermediate (may need more data than simple classical models, but don't seem to be SOTA on huge data)
  - ▶ small data still happens
    - e.g., if expensive to collect
    - e.g., long tail

***Driving  
(some of  
the)  
differences:  
how much  
data,  
compute***

- Compute:
  - ▶ kernel methods can be prohibitively expensive on big data (there are approximations that can trade off accuracy)
  - ▶ deep nets are driving a huge boom in construction of data centers (and corresponding electricity use)
  - ▶ small compute still happens (e.g., phones, sensors)
- Extreme compute limitation: models that humans can run in their heads (e.g., MMSE for evaluating cognitive impairment)
  - ▶ for interpretability, trust, unplugged use

## ***Related areas***

- Data science: Extracting knowledge/insights from noisy, unstructured data
- Statistics: Understanding how to draw verifiable conclusions from observable evidence
- Artificial intelligence: Creating machines that can mimic human behavior/cognition
- Computational neuroscience: Understanding how learning (and many other things) are implemented in the brain
- ...

# Our first ML task

- Learning to diagnose heart disease ( $T$ )
  - ▶ as a **(supervised) binary classification task** ( $E$ )

	features			labels
	Family History	Resting Blood Pressure	Cholesterol	Heart Disease?
data points	Yes	Low	Normal	No
	No	Medium	Normal	No
	No	Low	Abnormal	Yes
	Yes	Medium	Normal	Yes
	Yes	High	Abnormal	Yes

←  $D$

define: instances, attributes, ground truth

# Our first ML task

- Learning to diagnose heart disease ( $T$ )
  - ▶ as a **(supervised) binary classification task** ( $E$ )

our goal: a *classifier*  $h$  : features  $\rightarrow$  labels

data points

Family History	Resting Blood Pressure	Cholesterol	Heart Disease?
Yes	Low	Normal	No
No	Medium	Normal	No
No	Low	Abnormal	Yes
Yes	Medium	Normal	Yes
Yes	High	Abnormal	Yes

$\leftarrow D$

e.g. want  $h(\text{No, Medium, Normal}) = \text{No}$

# Our first ML task

- Learning to diagnose heart disease ( $T$ )
  - ▶ as a (**supervised**) binary classification task ( $E$ )

our goal: a *classifier*  $h$  : features  $\rightarrow$  labels

data points

Family History	Resting Blood Pressure	Cholesterol	Heart Disease?
Yes	Low	Normal	No
No	Medium	Normal	No
No	Low	Abnormal	Yes
Yes	Medium	Normal	Yes
Yes	High	Abnormal	Yes

$\leftarrow D$

# Our first ML task

- Learning to diagnose heart disease ( $T$ )
  - ▶ as a **(supervised) binary classification task** ( $E$ )

our goal: a *classifier*  $h$  : features  $\rightarrow$  labels

data points

Family History	Resting Blood Pressure	Cholesterol	Heart Disease?
Yes	Low	Normal	No
No	Medium	Normal	No
No	Low	Abnormal	Yes
Yes	Medium	Normal	Yes
Yes	High	Abnormal	Yes

$\leftarrow D$

# Our first ML task

- Learning to diagnose heart disease ( $T$ )
  - ▶ as a **(supervised)** classification task ( $E$ )

features labels

	Family History	Resting Blood Pressure	Cholesterol	Heart Disease?
data points	Yes	Low	Normal	Low Risk
	No	Medium	Normal	Low Risk
	No	Low	Abnormal	Medium Risk
	Yes	Medium	Normal	High Risk
	Yes	High	Abnormal	High Risk

←  $D$

# Our first ML task

- Learning to diagnose heart disease ( $T$ )
  - ▶ as a (supervised) regression task ( $E$ )

	features			labels
	Family History	Resting Blood Pressure	Cholesterol	Medical cost from heart disease?
data points	Yes	Low	Normal	\$0k
	No	Medium	Normal	\$0k
	No	Low	Abnormal	\$10k
	Yes	Medium	Normal	\$17k
	Yes	High	Abnormal	\$23k

←  $D$

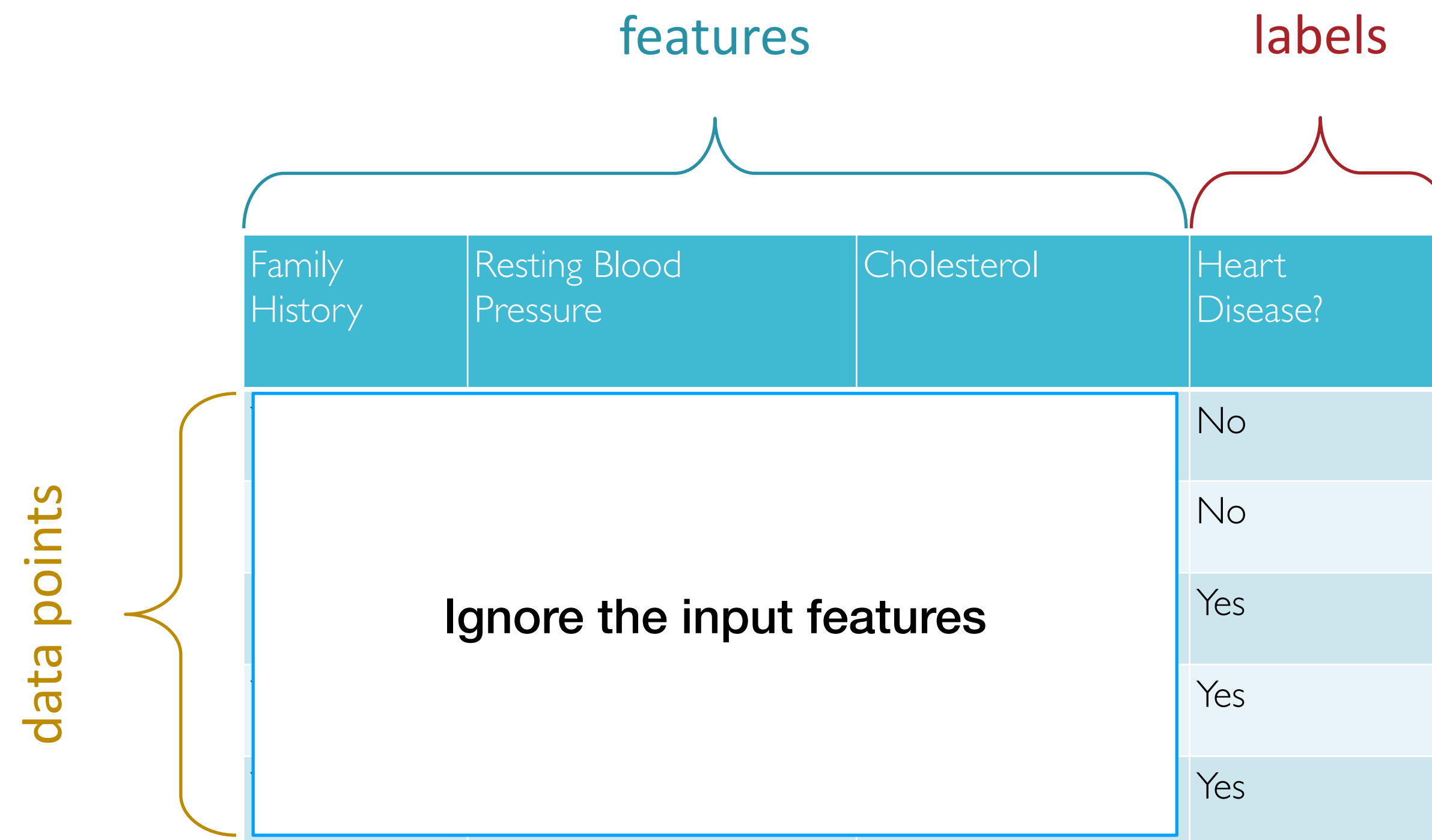
# Our first ML method

- Majority vote classifier
  - ▶ for a **(supervised) classification task**

	features			labels
	Family History	Resting Blood Pressure	Cholesterol	Heart Disease?
data points	Yes	Low	Normal	No
	No	Medium	Normal	No
	No	Low	Abnormal	Yes
	Yes	Medium	Normal	Yes
	Yes	High	Abnormal	Yes

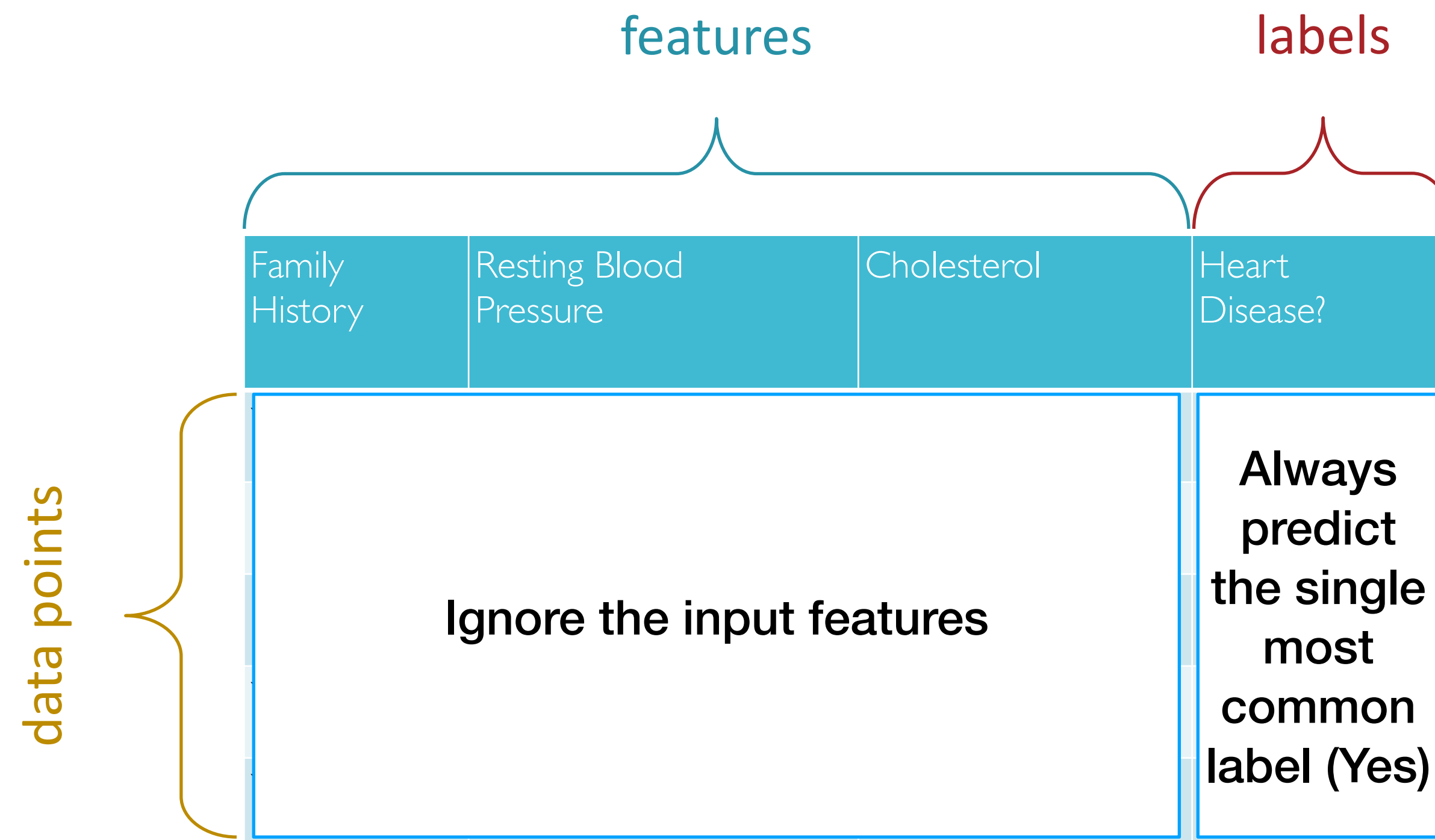
# Our first ML method

- Majority vote classifier
  - ▶ for a **(supervised) classification task**



# Our first ML method

- Majority vote classifier
  - ▶ for a **(supervised) classification task**



# Our first ML method

- Majority vote classifier
  - ▶ for a **(supervised) classification task**

The diagram shows a table with 5 columns and 5 rows. The columns are labeled 'features', 'labels', and 'predictions'. The first three columns are grouped under 'features', the fourth under 'labels', and the fifth under 'predictions'. The first four rows are grouped under 'data points'. The table content is as follows:

	Family History	Resting Blood Pressure	Cholesterol	Heart Disease?	Model output
	Yes	Low	Normal	No	Yes
	No	Medium	Normal	No	Yes
	No	Low	Abnormal	Yes	Yes
	Yes	Medium	Normal	Yes	Yes
	Yes	High	Abnormal	Yes	Yes

error rate =

(P)

***Is this  
good?***

- Is 40% error high or low?
- Do we care about this error rate?

# Training set vs. testing set

	features			labels
	Family History	Resting Blood Pressure	Cholesterol	Heart Disease?
training data points	Yes	Low	Normal	No
	No	Medium	Normal	No
	No	Low	Abnormal	Yes
	Yes	Medium	Normal	Yes
	Yes	High	Abnormal	Yes
testing data points	No	Low	Normal	No
	No	Medium	Normal	No
	Yes	Medium	Abnormal	Yes

training error rate =

testing error rate =

$(\hat{P})$

$(P)$

# Training set vs. testing set

Majority vote classifier: Always predict the single most common label *in the training set* (Yes)

	features			labels
	Family History	Resting Blood Pressure	Cholesterol	Heart Disease?
training data points	Yes	Low	Normal	No
	No	Medium	Normal	No
	No	Low	Abnormal	Yes
	Yes	Medium	Normal	Yes
	Yes	High	Abnormal	Yes
testing data points	No	Low	Normal	No
	No	Medium	Normal	No
	Yes	Medium	Abnormal	Yes

training error rate =

testing error rate =

$(\hat{P})$

$(P)$

# Training set vs. testing set

Majority vote classifier: Always predict the single most common label *in the training set* (Yes)

	features			labels	predictions
	Family History	Resting Blood Pressure	Cholesterol	Heart Disease?	Model output
training data points	Yes	Low	Normal	No	Yes
	No	Medium	Normal	No	Yes
	No	Low	Abnormal	Yes	Yes
	Yes	Medium	Normal	Yes	Yes
	Yes	High	Abnormal	Yes	Yes
testing data points	No	Low	Normal	No	Yes
	No	Medium	Normal	No	Yes
	Yes	Medium	Abnormal	Yes	Yes

training error rate =

$(\hat{P})$

testing error rate =

$(P)$

Memorizer: if we've seen *exact same* inputs before, predict corresponding output, else majority

# Our second ML method

	features			labels
	Family History	Resting Blood Pressure	Cholesterol	Heart Disease?
training data points	Yes	Low	Normal	No
	No	Medium	Normal	No
	No	Low	Abnormal	Yes
	Yes	Medium	Normal	Yes
	Yes	High	Abnormal	Yes
testing data points	No	Low	Normal	No
	No	Medium	Normal	No
	Yes	Medium	Abnormal	Yes

training error rate =

testing error rate =

Memorizer: if we've seen *exact same* inputs before, predict corresponding output, else majority

# Our second ML method

	features			labels	predictions
	Family History	Resting Blood Pressure	Cholesterol	Heart Disease?	Model output
training data points	Yes	Low	Normal	No	No
	No	Medium	Normal	No	No
	No	Low	Abnormal	Yes	Yes
	Yes	Medium	Normal	Yes	Yes
	Yes	High	Abnormal	Yes	Yes
testing data points	No	Low	Normal	No	
	No	Medium	Normal	No	
	Yes	Medium	Abnormal	Yes	

training error rate =

testing error rate =

Memorizer: if we've seen *exact same* inputs before, predict corresponding output, else majority

# Our second ML method

	features			labels	predictions
	Family History	Resting Blood Pressure	Cholesterol	Heart Disease?	Model output
training data points	Yes	Low	Normal	No	No
	No	Medium	Normal	No	No
	No	Low	Abnormal	Yes	Yes
	Yes	Medium	Normal	Yes	Yes
	Yes	High	Abnormal	Yes	Yes
testing data points	No	Low	Normal	No	Yes
	No	Medium	Normal	No	No
	Yes	Medium	Abnormal	Yes	Yes

training error rate =

testing error rate =

# Notation

- Data point =  $\langle x^{(t)}, y^{(t)} \rangle$
- Feature vector and feature space  $x^{(t)} \in \mathcal{X}$
- Label and label space  $y^{(t)} \in \mathcal{Y}$
- Hypothesis (classifier) and hypothesis space  $h \in \mathcal{H}$
- **Unknown** target or true classifier  $h^*$ 
  - ▶ best possible test error, might not even be in  $\mathcal{H}$
- Apparent best classifier  $\hat{h} \in \mathcal{H}$  given training set (sometimes called ERM, empirical risk minimizer)
  - ▶ minimizes training error over  $\mathcal{H}$
- ML method: a way of picking a hypothesis given data
  - ▶ e.g., majority vote = empirical risk minimization over constant classifiers

# *Learning goals*

- You should be able to
  - ▶ Formulate a well-posed learning problem for a real- world task by identifying the task, performance measure, and training experience
  - ▶ Describe the supervised learning paradigm in terms of the type of data needed, the form of prediction, and the structure of the output prediction
  - ▶ Explain the difference between memorization and generalization