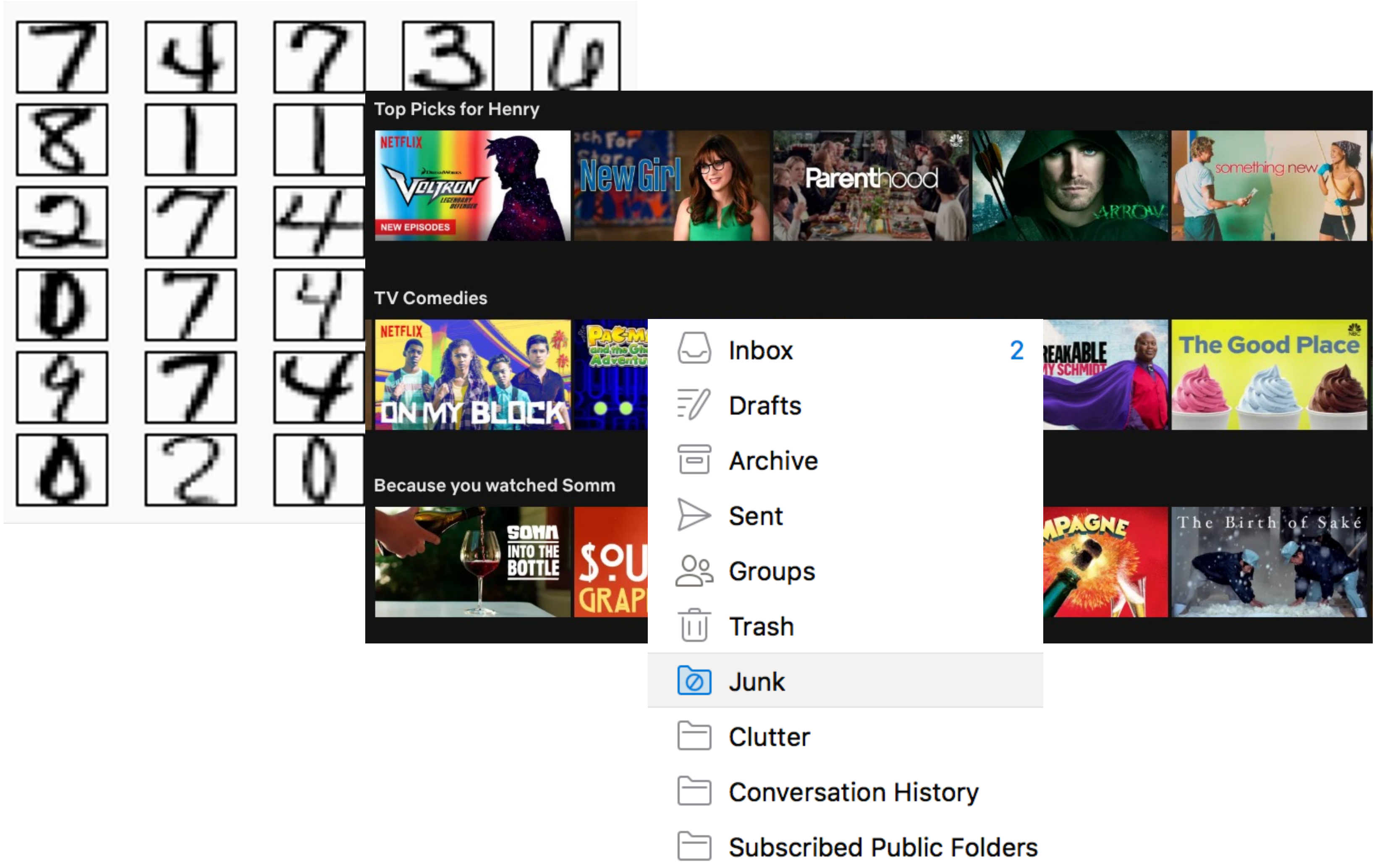
10-301/601: Introduction to Machine Learning Lecture 1 – Problem Formulation & Notation

Henry Chai & Matt Gormley

8/28/23

What is Machine Learning?

Machine Learning (A long long time ago…)



Machine Learning (A short time ago…)

Machine Learning (Now)

Machine Learning (Now)

What is ~~Machine Learning~~ 10-301/601?

- Supervised Models
  - Decision Trees
  - KNN
  - Naïve Bayes
  - Perceptron
  - Logistic Regression
  - Linear Regression
  - Neural Networks
- Unsupervised Learning
- Ensemble Methods
- Deep Learning
- Learning Theory
- Reinforcement Learning
- Important Concepts
  - Feature Engineering
  - Regularization and Overfitting
  - Experimental Design

What is Machine Learning?

Source: https://en.wikipedia.org/wiki/Panzanella

Things Machine Learning Isn't

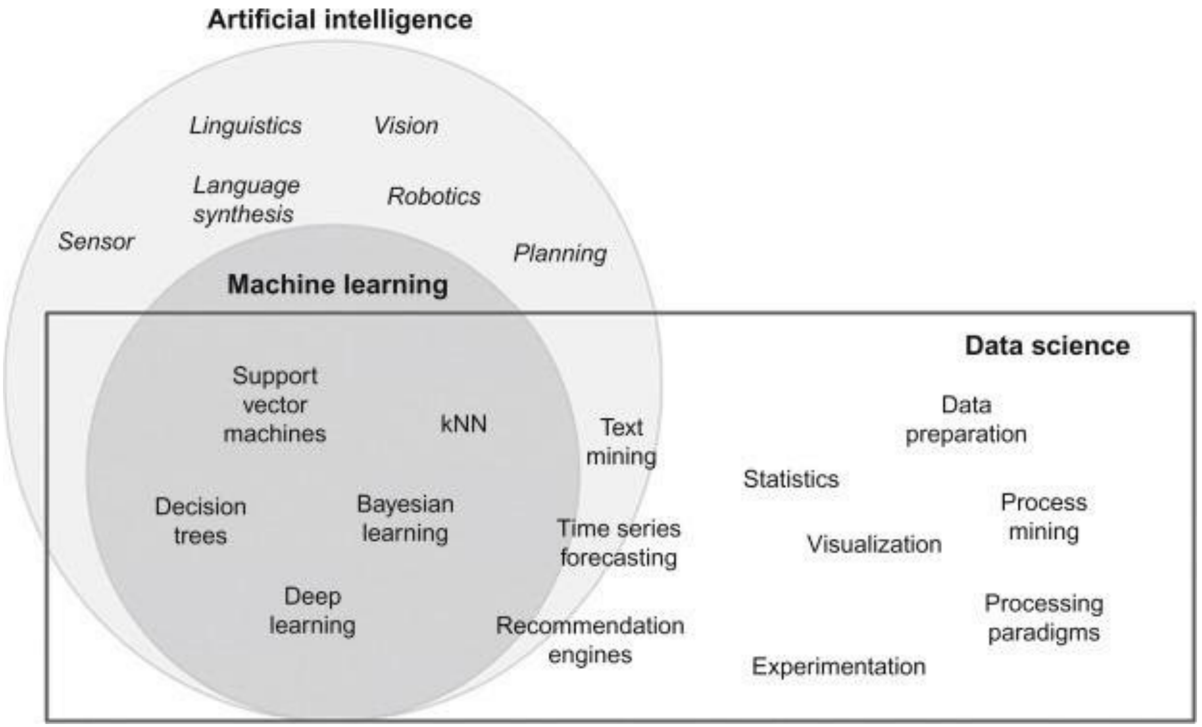
- Artificial intelligence
- Data science

Things Machine Learning Isn't

- Artificial intelligence: Creating machines that can mimic human behavior/cognition
- Data science

Things Machine Learning Isn't

- Artificial intelligence: Creating machines that can mimic human behavior/cognition
- Data science: Extracting knowledge/insights from noisy, unstructured data



Source: https://www.sciencedirect.com/topics/physics-and-astronomy/artificial-intelligence

DEFINING LEARNING PROBLEMS

Well-Posed Learning Problems

Three components *<T,P,E>***:**

1. Task, *T*
2. Performance measure, *P*
3. Experience, *E*

Definition of learning:

A computer program **learns** if its performance at task *T*, as measured by *P*, improves with experience *E*.

Definition from (Mitchell, 1997)

Example Learning Problems

Learning to **beat the masters at chess**

1. Task, *T*:

2. Performance measure, *P*:

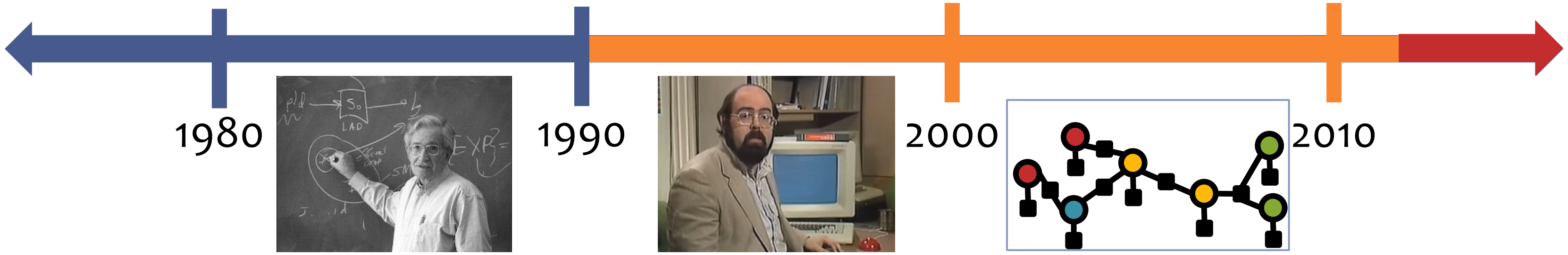
3. Experience, *E*:

Example Learning Problems

Learning to **respond to voice commands (Siri)**

1. Task, *T*:
2. Performance measure, *P*:
3. Experience, *E*:

Capturing the Knowledge of Experts



Solution #1: Expert Systems

- Over 20 years ago, we had rule-based systems:
    1. Put a bunch of linguists in a room
    2. Have them think about the structure of their native language and write down the rules they devise

Give me directions to Starbucks

```
If: "give me directions to X"
Then: directions(here, nearest(X))
```

How do I get to Starbucks?

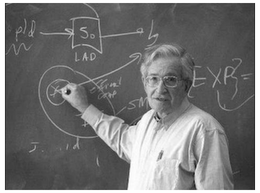
```
If: "how do i get to X"
Then: directions(here, nearest(X))
```

Where is the nearest Starbucks?

```
If: "where is the nearest X"
Then: directions(here, nearest(X))
```

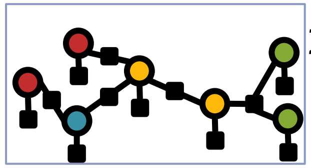
Capturing the Knowledge of Experts

1980



1990

2000



2010

Solution #1: Expert Systems

- Over 20 years ago, we had rule-based systems:
  1. Put a bunch of linguists in a room
  2. Have them think about the structure of their native language and write down the rules they devise

I need directions to Starbucks

```
If: "I need directions to X"
Then: directions(here, nearest(X))
```

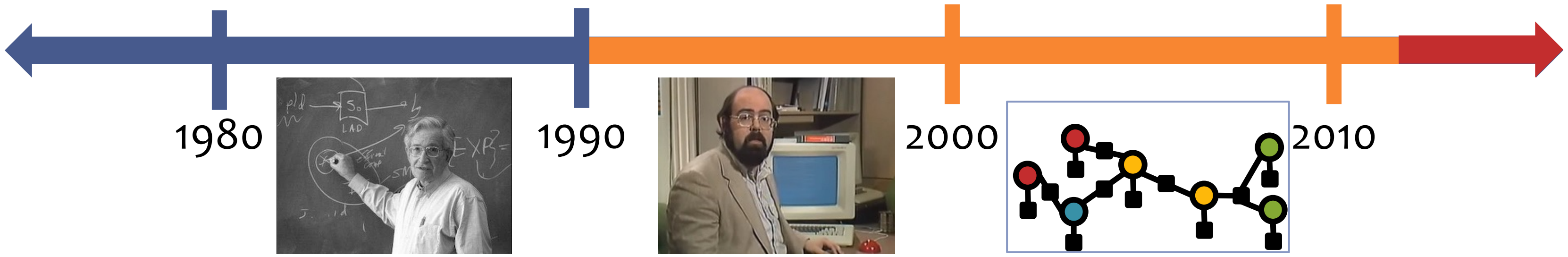
Starbucks directions

```
If: "X directions"
Then: directions(here, nearest(X))
```

Is there a Starbucks nearby?

```
If: "Is there an X nearby"
Then: directions(here, nearest(X))
```

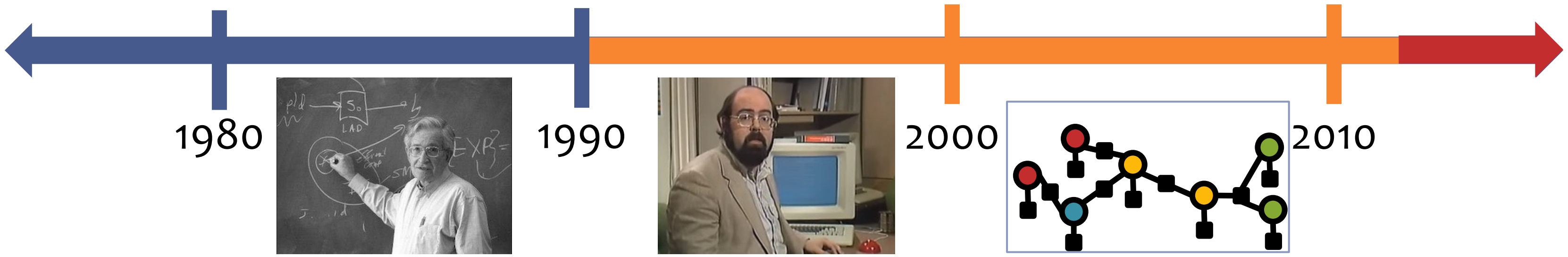
Capturing the Knowledge of Experts



Solution #2: Annotate Data and Learn

- Experts:
  - **Very good at** answering questions about specific cases
  - **Not very good at** telling **HOW** they do it
- 1990s: So why not just have them tell you what they do on **SPECIFIC CASES** and then let **MACHINE LEARNING** tell you how to come to the same decisions that they did

Capturing the Knowledge of Experts



Solution #2: Annotate Data and Learn

1. Collect raw sentences $\{x^{(1)}, \ldots, x^{(n)}\}$
2. Experts annotate their meaning $\{y^{(1)}, \ldots, y^{(n)}\}$

$x^{(1)}$: How do I get to Starbucks?

$y^{(1)}$: `directions(here, nearest(Starbucks))`

$x^{(2)}$: Show me the closest Starbucks

$y^{(2)}$: `map(nearest(Starbucks))`

$x^{(3)}$: Send a text to John that I'll be late

$y^{(3)}$: `txtmsg(John, I'll be late)`

$x^{(4)}$: Set an alarm for seven in the morning

$y^{(4)}$: `setalarm(7:00AM)`

Example Learning Problems

Learning to **respond to voice commands (Siri)**

1. Task, *T*:
   predicting action from speech
2. Performance measure, *P*:
   percent of correct actions taken in user pilot study
3. Experience, *E*:
   examples of (speech, action) pairs

Problem Formulation

- Often, the same task can be formulated in more than one way:
- Ex: Loan applications
  - creditworthiness/score (regression)
  - probability of default (density estimation)
  - loan decision (classification)

Problem Formulation:
What is the structure of our output prediction?

boolean	Binary Classification
categorical	Multiclass Classification
ordinal	Ordinal Classification
real	Regression
ordering	Ranking
multiple discrete	Structured Prediction
multiple continuous	(e.g. dynamical systems)
both discrete & cont.	(e.g. mixed graphical models)

Well-posed Learning Problems

In-Class Exercise

1. Select a **task**, T
2. Identify **performance measure**, P
3. Identify **experience**, E
4. Report ideas back to rest of class

Example Tasks

- Identify objects in an image
- Translate from one human language to another
- Recognize speech
- Assess risk (e.g. in loan application)
- Make decisions (e.g. in loan application)
- Assess potential (e.g. in admission decisions)
- Categorize a complex situation (e.g. medical diagnosis)
- Predict outcome (e.g. medical prognosis, stock prices, inflation, temperature)
- Predict events (default on loans, quitting school, war)
- Plan ahead under perfect knowledge (chess)
- Plan ahead under partial knowledge (poker, bridge)

Examples from Roni Rosenfeld

In-Class Exercise

1. Select a **task**, T
2. Identify **performance measure**, P
3. Identify **experience**, E
4. Report ideas back to rest of class

Well-posed Learning Problems

task, T	performance measure, P	experience, E

In-Class Exercise
1. Select a **task**, T
2. Identify **performance measure**, P
3. Identify **experience**, E
4. Report ideas back to rest of class

Well-posed Learning Problems

task, T	performance measure, P	experience, E

Our first Machine Learning Task

- Learning to diagnose heart disease

  as a **(supervised) binary classification task**

features | 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

Our first Machine Learning Task

- Learning to diagnose heart disease

  as a **(<u>supervised</u>) binary classification task**

features

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

Our first Machine Learning Task

- Learning to diagnose heart disease

  as a **(supervised)** **binary classification** **task**

features | 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

Our first Machine Learning Task

- Learning to diagnose heart disease

  as a **(supervised)** **<u>classification</u> task**

features

labels

data points

Family History	Resting Blood Pressure	Cholesterol	Risk
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

Our first Machine Learning Task

- Learning to diagnose heart disease

  as a **(supervised)** **regression task**

Family History	Resting Blood Pressure	Cholesterol	Medical Costs
Yes	Low	Normal	$0
No	Medium	Normal	$20
No	Low	Abnormal	$30
Yes	Medium	Normal	$100
Yes	High	Abnormal	$5000

features

targets

data points

Our first Machine Learning Classifier

- A **classifier** is a function that takes feature values as input and outputs a label
- Majority vote classifier: always predict the most common label in the dataset

features | 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

Is this a "good" Classifier?

- A **classifier** is a function that takes feature values as input and outputs a label
- Majority vote classifier: always predict the most common label in the dataset

features 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

Training vs. Testing

- A **classifier** is a function that takes feature values as input and outputs a label
- Majority vote classifier: always predict the most common label in the **training** dataset (Yes)

training dataset

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

Training vs. Testing

- A **classifier** is a function that takes feature values as input and outputs a label
- Majority vote classifier: always predict the most common label in the **training** dataset (Yes)
- A **test** dataset is used to evaluate a classifier's **predictions**

test dataset

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

- The **error rate** is the proportion of data points where the prediction is wrong

Training vs. Testing

- A **classifier** is a function that takes feature values as input and outputs a label
- Majority vote classifier: always predict the most common label in the **training** dataset (Yes)
- A **test** dataset is used to evaluate a classifier's **predictions**

test dataset

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

- The **test error rate** is the proportion of data points in the test dataset where the prediction is wrong ($1/3$)

A Typical (Supervised) Machine Learning Routine

- Step 1 – training
  - Input: a labelled training dataset
  - Output: a classifier
- Step 2 – testing
  - Inputs: a classifier, a test dataset
  - Output: predictions for each test data point
- Step 3 – evaluation
  - Inputs: predictions from step 2, test dataset labels
  - Output: some measure of how good the predictions are; usually (but not always) error rate

Our first Machine Learning Classifier

- A **classifier** is a function that takes feature values as input and outputs a label
- Majority vote classifier: always predict the most common label in the training dataset

labels

data points

Heart Disease?
No
No
Yes
Yes
Yes

- This classifier completely ignores the features…

Our first Machine Learning Classifier

- A **classifier** is a function that takes feature values as input and outputs a label
- Majority vote classifier: always predict the most common label in the **training** dataset

labels

data points

Heart Disease?	Predictions
No	Yes
No	Yes
Yes	Yes
Yes	Yes
Yes	Yes

- The training error rate is $2/5$

Our second Machine Learning Classifier

- A **classifier** is a function that takes feature values as input and outputs a label
- Memorizer: if a set of features exists in the **training** dataset, predict its corresponding label; otherwise, predict the majority vote

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

Our second Machine Learning Classifier

- A **classifier** is a function that takes feature values as input and outputs a label
- Memorizer: if a set of features exists in the **training** dataset, predict its corresponding label; otherwise, predict the majority vote

Family History	Resting Blood Pressure	Cholesterol	Heart Disease?	Predictions
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

- The training error rate is 0!

Is the memorizer learning?

- A **classifier** is a function that takes feature values as input and outputs a label
- Memorizer: if a set of features exists in the **training** dataset, predict its corresponding label; otherwise, predict the majority vote

Family History	Resting Blood Pressure	Cholesterol	Heart Disease?	Predictions
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

- The training error rate is 0!

Learning Goals

- You should be able to

1. Formulate a well-posed learning problem for a real-world task by identifying the task, performance measure, and training experience
2. Describe common learning paradigms in terms of the type of data available, when it's available, the form of prediction, and the structure of the output prediction
3. Explain the difference between memorization and generalization [CIML]
4. Identify examples of the ethical responsibilities of an ML expert

Logistics: Course Website

http://www.cs.cmu.edu/~mgormley/courses/10601/

(or mlcourse.org)

Logistics: Course Syllabus

http://www.cs.cmu.edu/~mgormley/courses/10601/syllabus.html

- This whole section is **required** reading

Logistics: Grading

http://www.cs.cmu.edu/~mgormley/courses/10601/syllabus.html

- 50% homeworks
- 15% exam 1
- 15% exam 2
- 15% exam 3
- 5% participation

Logistics: Late Policy

http://www.cs.cmu.edu/~mgormley/courses/10601/syllabus.html

- You have 6 grace days for homework assignments
- Only 3 grace days may be used per homework
  - Only <u>2</u> grace days may be used on homeworks leading up to an exam (HW3, HW6, HW9)
- Late submissions w/o grace days will be penalized as:
  - 1 day late = 75% multiplicative penalty
  - 2 days late = 50% multiplicative penalty
  - 3 days late = 25% multiplicative penalty
- No submissions will be accepted more than 3 days late

Logistics: Collaboration Policy

http://www.cs.cmu.edu/~mgormley/courses/10601/syllabus.html

- Collaboration on homework assignments is encouraged but must be documented
- **You must always write your own code/answers**
  - You may not re-use code/previous versions of the homework, whether your own or otherwise
- Good approach to collaborating on programming assignments:
  1. Collectively sketch pseudocode on an impermanent surface, then
  2. Disperse, erase all notes and start from scratch

Logistics: Technologies

http://www.cs.cmu.edu/~mgormley/courses/10601/syllabus.html

- Piazza, for course discussion: https://piazza.com/class/llay25hzand2fa/
- Gradescope, for submitting homework assignments: https://www.gradescope.com/courses/571249
- Google Forms for in-class polls (more details next week)
- Panopto, for lecture recordings: https://scs.hosted.panopto.com/Panopto/Pages/Sessions/List.aspx?folderID=d5bf275d-ff88-4bf6-a865-b065010f55c2

Logistics: Lecture Schedule

http://www.cs.cmu.edu/~mgormley/courses/10601/schedule.html

Date	Lecture	Readings	Announcements
	Classification & Regression		
Mon, 28-Aug	Lecture 1 : Course Overview	• *Command Line and File I/O Tutorial*. 10601 Course Staff (2020). • *10601 Learning Objectives*. Matt Gormley (2023). • *Math Resources*. 10601 Course Staff (2023).	
Wed, 30-Aug	Lecture 2 : Machine Learning as Function Approximation	• *10601 Notation Crib Sheet*. Matt Gormley (2023).	
Fri, 1-Sep	Background Test (in-class, required)		HW1 Out
Sat, 2-Sep	Recitation: HW1 (video recording only)		
Mon, 4-Sep	Labor Day		
Wed, 6-Sep	Lecture 3 : Decision Trees	• *Visual Information Theory*. Christopher Olah (2015). blog. • *Decision Trees*. Hal Daumé III (2017). CIML, Chapter 1.	HW1 Due HW2 Out
Fri, 8-Sep	Recitation: HW2		

Logistics: Lectures

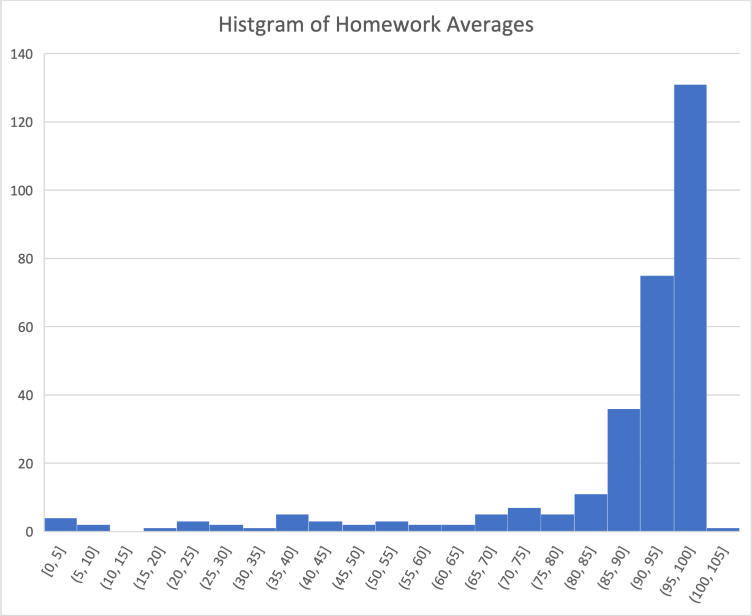
- During lecture, you should ask lots of questions!
  - Interrupting (by raising a hand) to ask your question is strongly encouraged
  - Asking questions over Zoom or later via Piazza is also great
- When we ask you all a question, we really do want you to answer!
  - Even if you don’t answer, think it through as if we had called on you
- Interaction improves learning, in-class, at office hours and amongst yourselves (to a point of course)

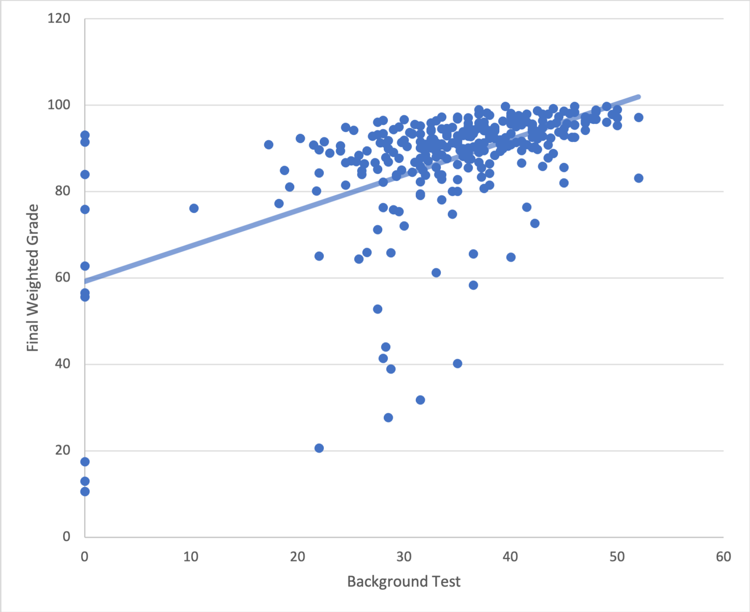
FAQ: Am I prepared to take this course?

- Background Test:
  - Friday, September 1st during recitation (same time and place as lecture)
  - Covers prerequisite material (probability, statistics, linear algebra, geometry, calculus and computer science)

FAQ: A test in the first week of class??? Really???

- Background Test:
  - $\alpha =$ % of points on the Background Test
  - $\beta =$ % of points on the written portion of HW1
  - Grade $= \alpha + (1 - \alpha)\beta$





Logistics: Exam Schedule

http://www.cs.cmu.edu/~mgormley/courses/10601/schedule.html

⋮

Thu, 28-Sep	Exam 1 (evening exam, details will be announced on Piazza)		
Fri, 29-Sep	Recitation: HW4		HW4 Out

⋮

Thu, 9-Nov	Exam 2 (evening exam, details will be announced on Piazza)		
Fri, 10-Nov	Recitation: HW7		HW7 Out

⋮

TBD, TBD	Exam 3 (during Final Exam Period -- exact time/date TBD by the registrar)		

Logistics: Assignments

http://www.cs.cmu.edu/~mgormley/courses/10601/coursework.html

Assignments

There will be 9 homework assignments during the semester in addition to the exams. The assignments will consist of both theoretical and programming problems. Homework assignments will be released via a Piazza announcement explaining where to find the handout, starter code, LaTeX template, etc.

The links to the **Homework Handouts** and **Homework Exit Polls** will be provided below.

- Homework 1: Background Material (written / programming)
- Homework 2: Decision Trees (written / programming)
- Homework 3: KNN, Perceptron, and Linear Regression (written)
- Homework 4: Logistic Regression (written / programming)
- Homework 5: Neural Networks (written / programming)
- Homework 6: Generative Models (written)
- Homework 7: Transformers in PyTorch (written / programming)
- Homework 8: Reinforcement Learning (written / programming)
- Homework 9: Learning Paradigms (written)

Tentative release dates and due dates are listed on the Schedule page.

Exams

There will be three exams. The links to the **Practice Problems** and **Exam Exit Polls** will be provided below.

- Exam 1 (in-person): Lectures 1-7
- Exam 2 (in-person): Lectures 8-17
- Exam 3 (in-person): Lectures 18-27

Logistics: Office Hours

http://www.cs.cmu.edu/~mgormley/courses/10601/officehours.html