10-301/601: Introduction to Machine Learning Lecture 2 – ML as Function Approximation

Hoda Heidari, Henry Chai & Matt Gormley

1/22/24

Front Matter

- Announcements:
  - HW1 released 1/17, due 1/24 (Wednesday) at 11:59 PM
  - Two components: written and programming
    - Separate assignments on Gradescope
  - Unique policies specific to HW1:
    - Two opportunities to submit the written portion (see write-up for details)
    - Unlimited submissions to the autograder for the (really, just keep submitting until you get 100%)
    - **We will grant (almost) any extension request**

Taylor series

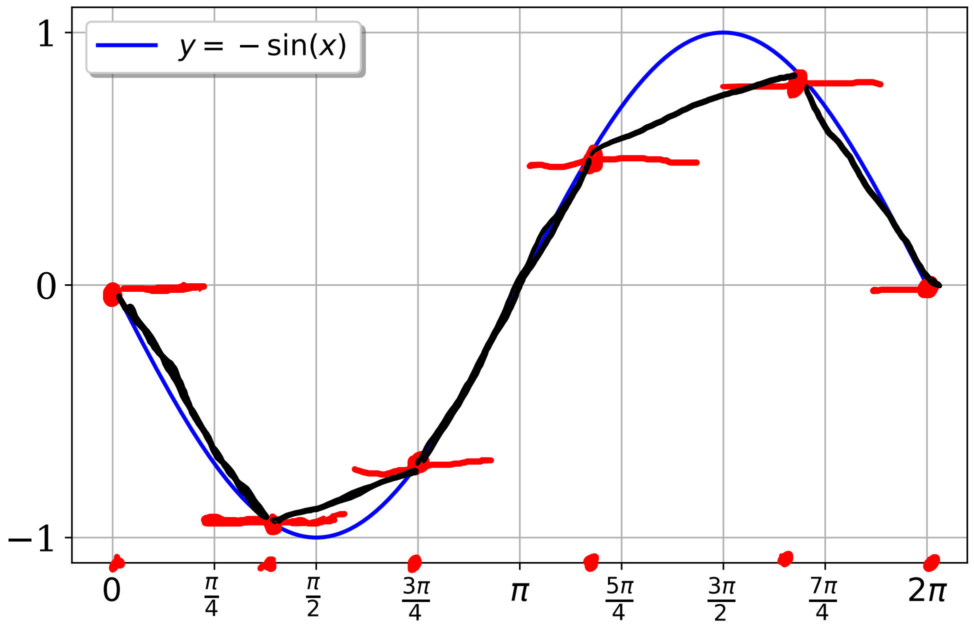
$$f(x) \approx f(0) + \frac{f'(0)}{1!}x + \frac{f''(0)}{2!}x^2$$

Function Approximation: Example

hard-code some values then round

- Challenge: implement a function that computes

  $-\sin(x)$ for $x \in [0, 2\pi]$



- You may not call any trigonometric functions
- You may call an existing implementation of $\sin(x)$ a few times (e.g., 10~~0~~) to check your work

Recall: 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

Family History	Resting Blood Pressure	Cholesterol	Heart Disease?	Predictions
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

Trying to approximate $-\sin(x)$

Notation

- Feature space, $\mathcal{X}$ $[0, 2\pi]$
- Label space, $\mathcal{Y}$ $[-1, 1]$ or $\mathbb{R}$
- (Unknown) Target function, $c^*: \mathcal{X} \rightarrow \mathcal{Y}$ $-\sin(x)$

c^* maps an element of $\mathcal{X}$ to an element of $\mathcal{Y}$

- Training dataset:

$$\mathcal{D} = \{(\boldsymbol{x}^{(1)}, c^*(\boldsymbol{x}^{(1)}) = y^{(1)}), (\boldsymbol{x}^{(2)}, y^{(2)}) \dots, (\boldsymbol{x}^{(N)}, y^{(N)})\}$$

- Example: $(\boldsymbol{x}^{(n)}, y^{(n)}) = \left(x_1^{(n)}, x_2^{(n)}, \dots, x_D^{(n)}, y^{(n)}\right)$
- Hypothesis space: $\mathcal{H}$ "all" piecewise linear functions over the domain
- Goal: find a classifier, $h \in \mathcal{H}$, that best approximates c^*

Our first Machine Learning Task

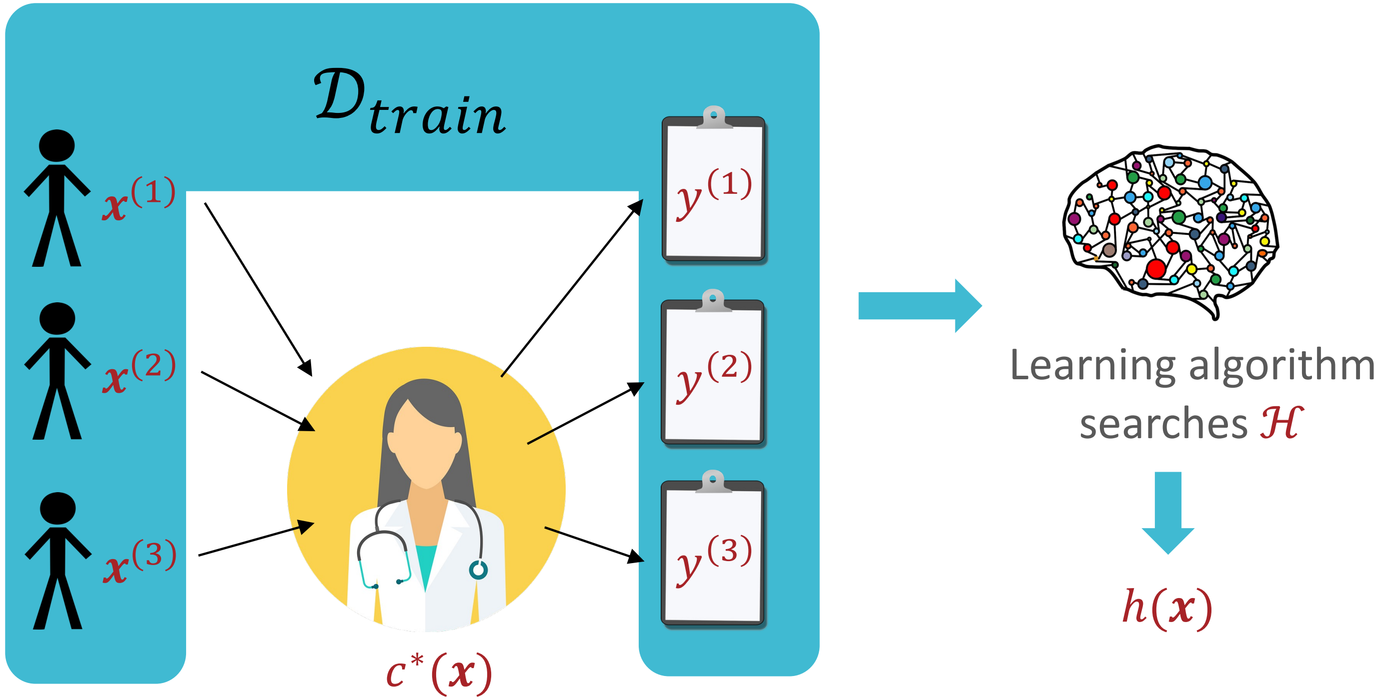


Figure courtesy of Matt Gormley

Notation: Example

	x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?	$\hat{y}$ Predictions
	Yes	Low	Normal	No	Yes
$\boldsymbol{x}^{(2)}$	No	Medium	Normal	No	Yes
	No	Low	Abnormal	Yes	Yes
	Yes	Medium	Normal	Yes	Yes
	Yes	High	Abnormal	Yes	Yes

- $N = 5$ and $D = 3$
- $\boldsymbol{x}^{(2)} = \left(x_1^{(2)} = \text{“No”}, x_2^{(2)} = \text{“Medium”}, x_3^{(2)} = \text{“Normal”}\right)$

Evaluation

- Loss function, $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$
  - Defines how "bad" predictions, $\hat{y} = h(\boldsymbol{x})$, are compared to the true labels, $y = c^*(\boldsymbol{x})$
  - Common choices

absolute loss : $|y - \hat{y}|$

1. Squared loss (for regression): $\ell(y, \hat{y}) = (y - \hat{y})^2$
2. Binary or 0-1 loss (for classification):

$$\ell(y, \hat{y}) = \begin{cases} 1 & \text{if } y \neq \hat{y} \\ 0 & \text{otherwise} \end{cases}$$

Evaluation

- Loss function, $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$
  - Defines how "bad" predictions, $\hat{y} = h(\boldsymbol{x})$, are compared to the true labels, $y = c^*(\boldsymbol{x})$
  - Common choices

1. Squared loss (for regression): $\ell(y, \hat{y}) = (y - \hat{y})^2$
2. Binary or 0-1 loss (for classification):

$$\ell(y, \hat{y}) = \mathbb{1}(y \neq \hat{y})$$

indicator: takes on value 1 if the argument is TRUE, 0 otherwise

- Error rate:

$$err(h, \mathcal{D}) = \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}\left(y^{(n)} \neq \hat{y}^{(n)}\right)$$

Different Kinds of Error

- Training error rate = $err(h, \mathcal{D}_{train})$

- Test error rate = $err(h, \mathcal{D}_{test})$

- True error rate = $err(h)$

  = the error rate of h on all possible examples

  - In machine learning, this is the quantity that we care about but, in most cases, it is unknowable.

Our first Machine Learning Task

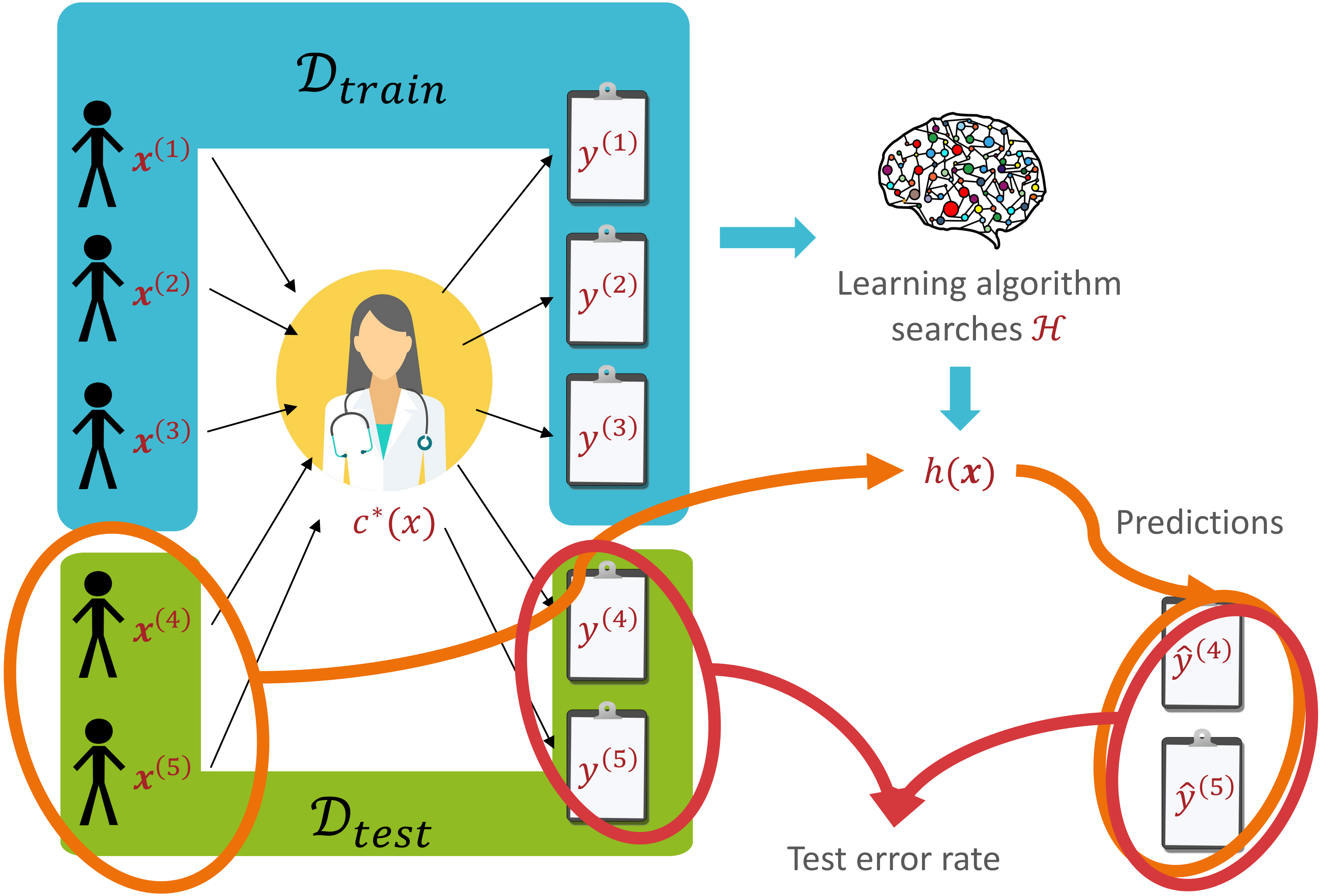
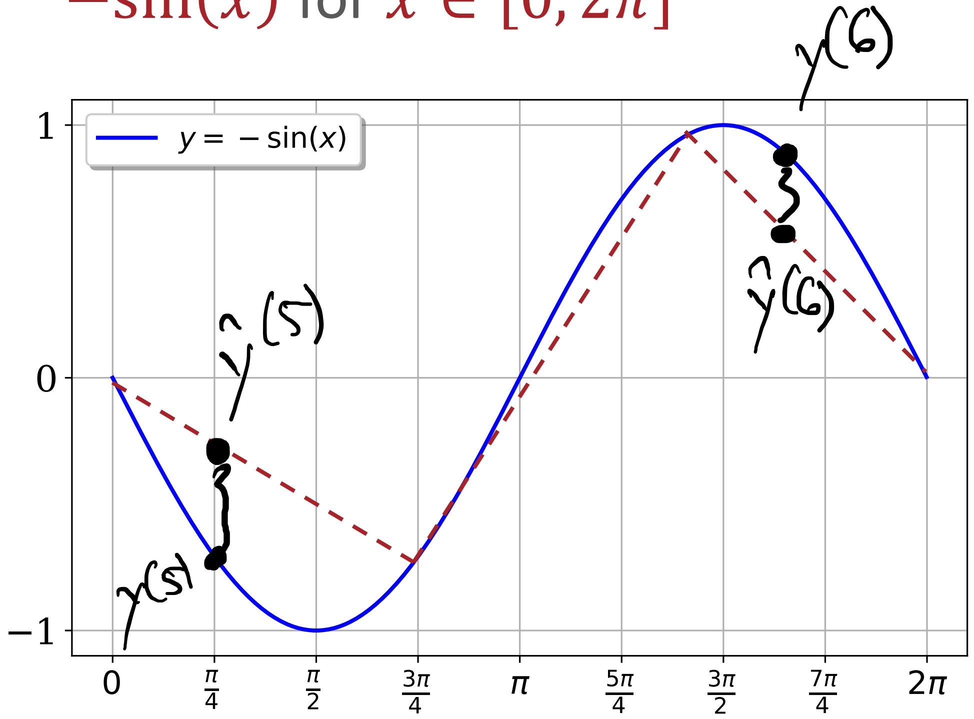


Figure courtesy of Matt Gormley

Function Approximation: Example

- Challenge: implement a function that computes

  $-\sin(x)$ for $x \in [0, 2\pi]$



$$err(h, D_{test}) = \frac{1}{2}\left(\left(y^{(5)} - \hat{y}^{(5)}\right)^2 + \left(y^{(6)} - \hat{y}^{(6)}\right)^2 \right)$$

- You may not call any trigonometric functions

- You may call an existing implementation of $\sin(x)$ a few times (e.g., 100) to check your work

Test your understanding

x_1	x_2	y
1	0	-
1	0	-
1	0	+
1	0	+
1	1	+
1	1	+
1	1	+
1	1	+

- What is the **training error** rate of the **majority vote classifier** on this dataset?

Our first Machine Learning Classifier: Pseudocode

- Majority vote classifier:

classifier specific

```
def train(D_train):
    store v = mode(y^(1), y^(2), ..., y^(N))

def h(x'):
    return v
```

generic

```
def predict(D_test):
    for (x^(n), y^(n)) ∈ D_test:
        ŷ^(n) = h(x^(n))
```

Recall: Our second Machine Learning Classifier

- 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: Pseudocode

- Memorizer:

def train(D_{train}):

  store D_{train}

def $h(x')$:

  if $\exists\ (x^{(n)}, y^{(n)}) \in D_{train}$ s.t. $x^{(n)} = x'$:

    return $y^{(n)}$

  else: return $\underline{\text{mode}}(y^{(1)}, y^{(2)}, \ldots, y^{(N)})$

def predict(D_{test}):

  ⋮

Our third Machine Learning Classifier

- Alright, let's actually (try to) extract a pattern from the data

x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?
Yes	Low	Normal	No
No	Medium	Normal	No
No	Low	Abnormal	Yes
Yes	Medium	Normal	Yes
Yes	High	Abnormal	Yes

- Decision stump: based on a single feature, x_d, predict the most common label in the **training** dataset among all data points that have the same value for x_d

Our third Machine Learning Classifier

- Alright, let's actually (try to) extract a pattern from the data

x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?
Yes	Low	Normal	No
No	Medium	Normal	No
No	Low	Abnormal	Yes
Yes	Medium	Normal	Yes
Yes	High	Abnormal	Yes

- Decision stump on x_1:

$$h(\boldsymbol{x}') = h(x'_1, \dots, x'_D) = \begin{cases} ??? & \boxed{\text{if } x'_1 = \text{“Yes”}} \\ ??? & \text{otherwise} \end{cases}$$

Our third Machine Learning Classifier

- Alright, let's actually (try to) extract a pattern from the data

x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?
Yes	Low	Normal	No
No	Medium	Normal	No
No	Low	Abnormal	Yes
Yes	Medium	Normal	Yes
Yes	High	Abnormal	Yes

- Decision stump on x_1:

$$h(\boldsymbol{x}') = h(x'_1, \dots, x'_D) = \begin{cases} \text{“Yes” if } x'_1 = \text{“Yes”} \\ ??? \ \boxed{\text{otherwise}} \end{cases}$$

Our third Machine Learning Classifier

- Alright, let's actually (try to) extract a pattern from the data

x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?	$\hat{y}$ Predictions
Yes	Low	Normal	No	Yes
No	Medium	Normal	No	No
No	Low	Abnormal	Yes	No
Yes	Medium	Normal	Yes	Yes
Yes	High	Abnormal	Yes	Yes

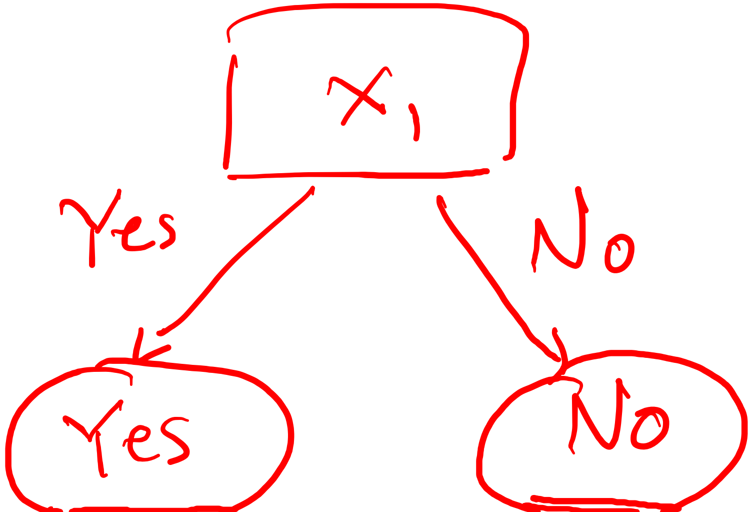
- Decision stump on x_1:

$$h(\boldsymbol{x}') = h(x'_1, \dots, x'_D) = \begin{cases} \text{“Yes” if } x'_1 = \text{“Yes”} \\ \text{“No” otherwise} \end{cases}$$

Our third Machine Learning Classifier

- Alright, let's actually (try to) extract a pattern from the data

x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?	$\hat{y}$ Predictions
Yes	Low	Normal	No	Yes
No	Medium	Normal	No	No
No	Low	Abnormal	Yes	No
Yes	Medium	Normal	Yes	Yes
Yes	High	Abnormal	Yes	Yes



Decision Stumps: Pseudocode

```
def train(D_train):
    pick a feature to split on, x_d
    # split D_train according to x_d
    for v in V(x_d), all possible values for x_d:
        D_v = {(x^(n), y^(n)) ∈ D_train | x_d^(n) = v}
    # compute a majority vote for each split
    for v in V(x_d):
        store ŷ_v = mode(labels in D_v)
def h(x'):
    for v in V(x_d):
        if x'_d = v: return ŷ_v
```

Decision Stumps: Questions

1. Why stop at just one feature?
2. How can we pick which feature to split on?
3. How can we pick the order of the splits?

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 2 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
  - You may not use generative AI tools to complete any portion of the assignments
- 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/lqzftil6bgtwd/
- Gradescope, for submitting homework assignments: https://www.gradescope.com/courses/693840
- Google Forms for in-class polls (more details next lecture)
- Panopto, for lecture recordings: https://scs.hosted.panopto.com/Panopto/Pages/Sessions/List.aspx?folderID=98a22931-8b47-4fa4-89c2-b0f1014438a0

Logistics: Lecture Schedule

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

Tentative Schedule

Date	Lecture	Readings	Announcements
Classification & Regression			
Wed, 17-Jan	Lecture 1 : Course Overview [Slides] [Slides (Inked)]	• *Command Line and File I/O Tutorial*. 10601 Course Staff (2020). • *10601 Learning Objectives*. Matt Gormley (2023). • *Math Resources*. 10601 Course Staff (2023).	HW1 Out
Fri, 19-Jan	Recitation: HW1 [Handout] [Solutions]		
Mon, 22-Jan	Lecture 2 : Machine Learning as Function Approximation	• *10601 Notation Crib Sheet*. Matt Gormley (2023).	
Wed, 24-Jan	Lecture 3 : Decision Trees [Poll]	• *Visual Information Theory*. Christopher Olah (2015). blog. • *Decision Trees*. Hal Daumé III (2017). CIML, Chapter 1.	HW1 Due HW2 Out
Fri, 26-Jan	Recitation: HW2 [Handout] [Solutions]		

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)

Logistics: Exam Schedule

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

⋮

Mon, 19-Feb	Lecture 10 : Feature Engineering / Regularization [Poll]	• *Regularization for Deep Learning*. Ian Goodfellow, Yoshua Bengio, & Aaron Courville (2016). Deep Learning, Chapter 7.1 and 7.8.	
Mon, 19-Feb	Exam 1 (evening exam, details will be announced on Piazza)		HW4 Out

⋮

Wed, 27-Mar	Lecture 19 : Pre-training, Fine-tuning, In-context Learning [Poll]		
Thu, 28-Mar	Exam 2 (evening exam, details will be announced on Piazza)		HW7 Out

⋮

TBD, TBD	Exam 3 (during Final Exam Period -- exact time/date TBD by the registrar, details will be announced on Piazza)		

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 **Overleaf Templates** will be provided below.

- Homework 1: Background Material (written / programming)
  Handout Overleaf Link
- 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.

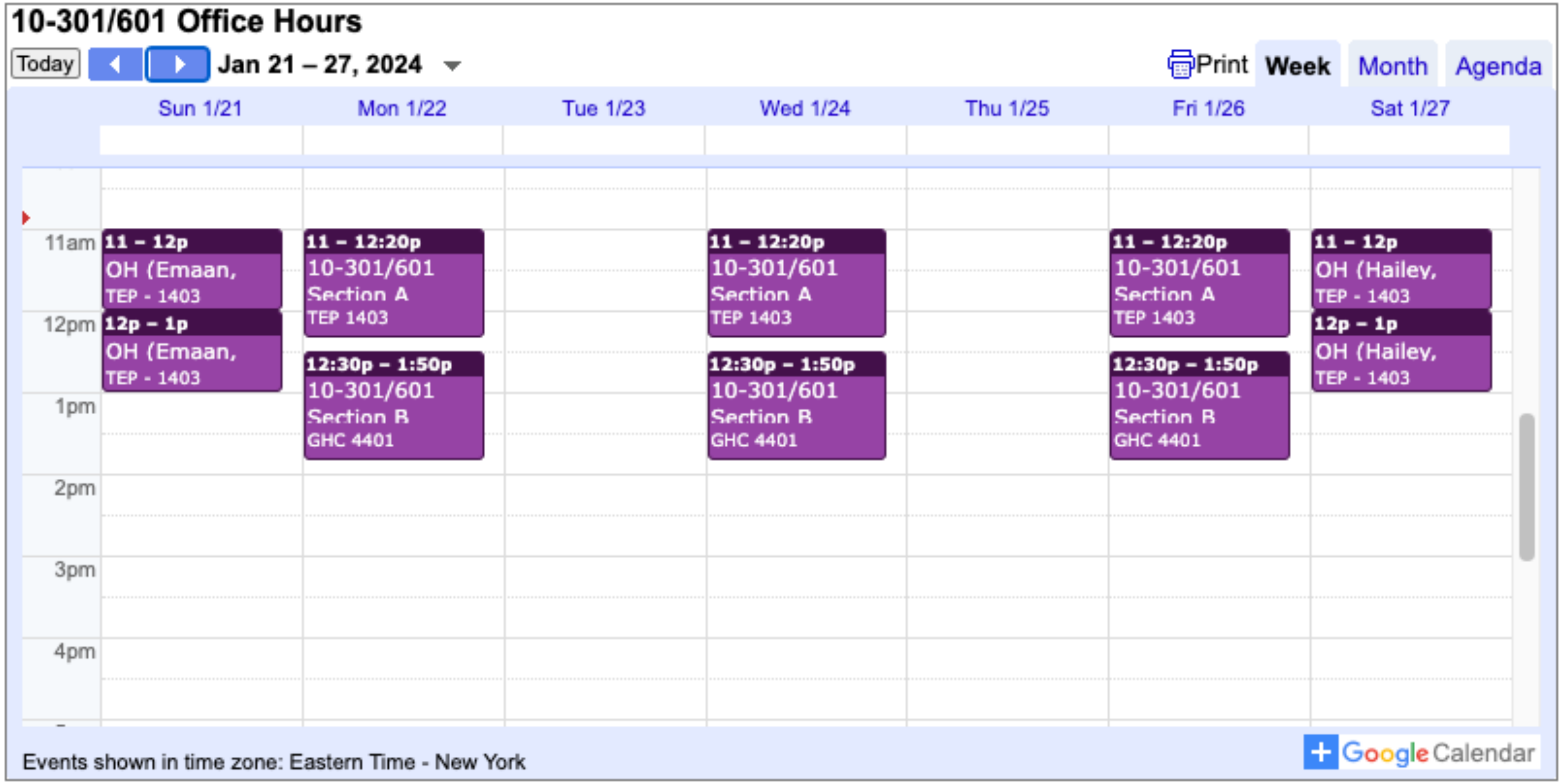
Exams

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 8-16
- Exam 3 (in-person): Lectures 17-27

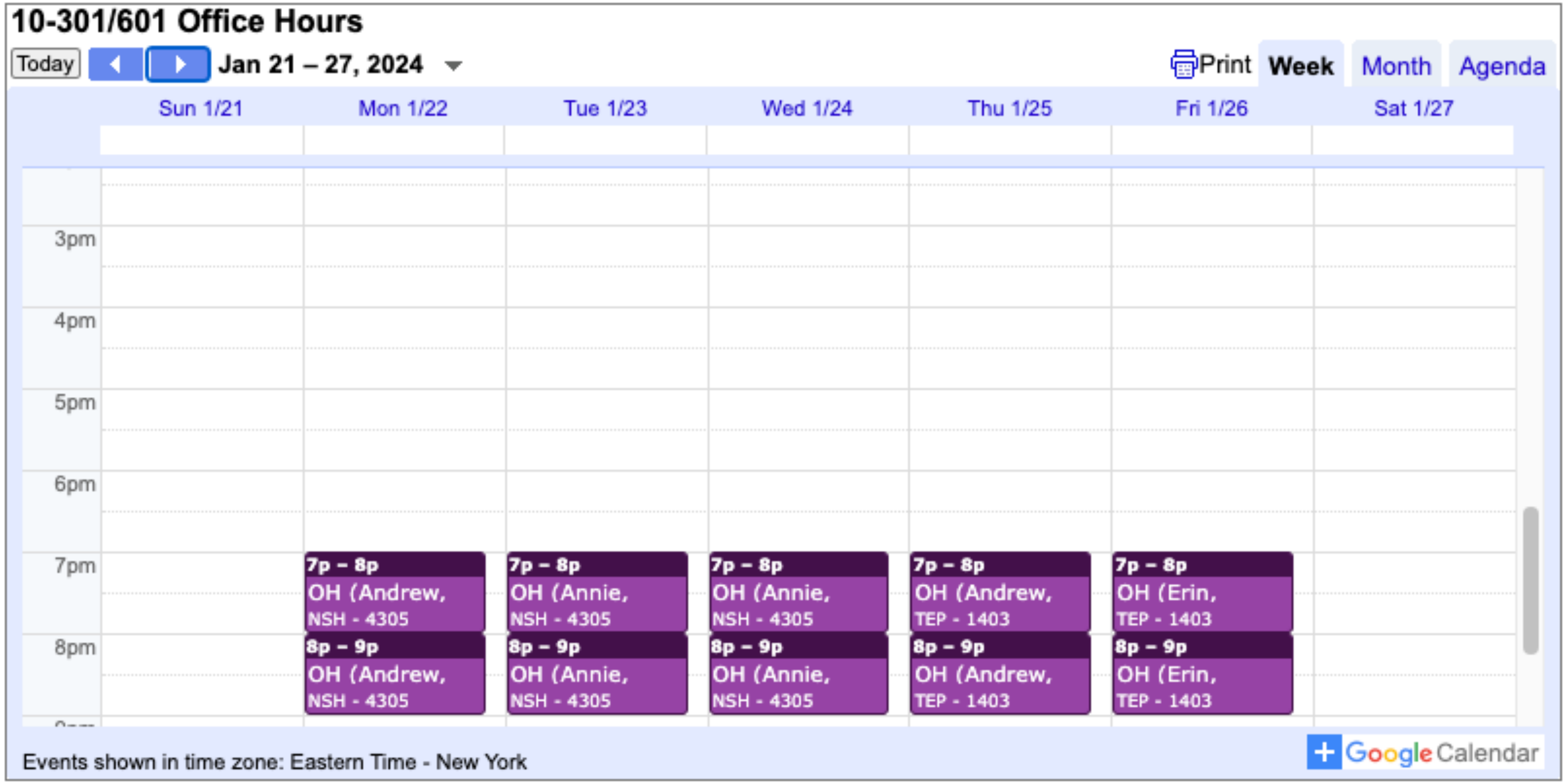
Logistics: Office Hours

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



Logistics: Office Hours

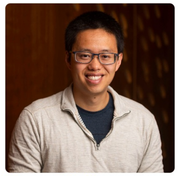
http://www.cs.cmu.edu/~mgormley/courses/10601/officehours.html



Logistics: Course Staff

Instructors

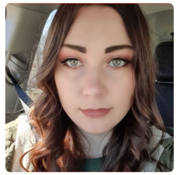

Hoda Heidari


Henry Chai


Matt Gormley

Education Associate


Nichelle Phillips


Brynn Edmunds

HW2 / HW6


Erin Gao

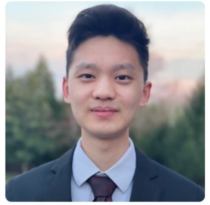

Annie Wu


Hailey Xia


Aadit Deshpande

HW3 / HW7


Bhargav Hadya


Sebastian Lu


Varsha Reddy Redla

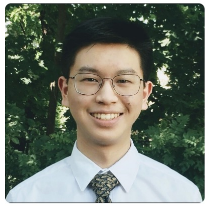

Rohan Chawla

HW4 / HW8


Monica Geng


Emaan Ahmed


Shivi Jindal


Max Tang

HW5 / HW9


Andrew Wang


Emily Xie


Kushagra Agarwal


Zoe Xu