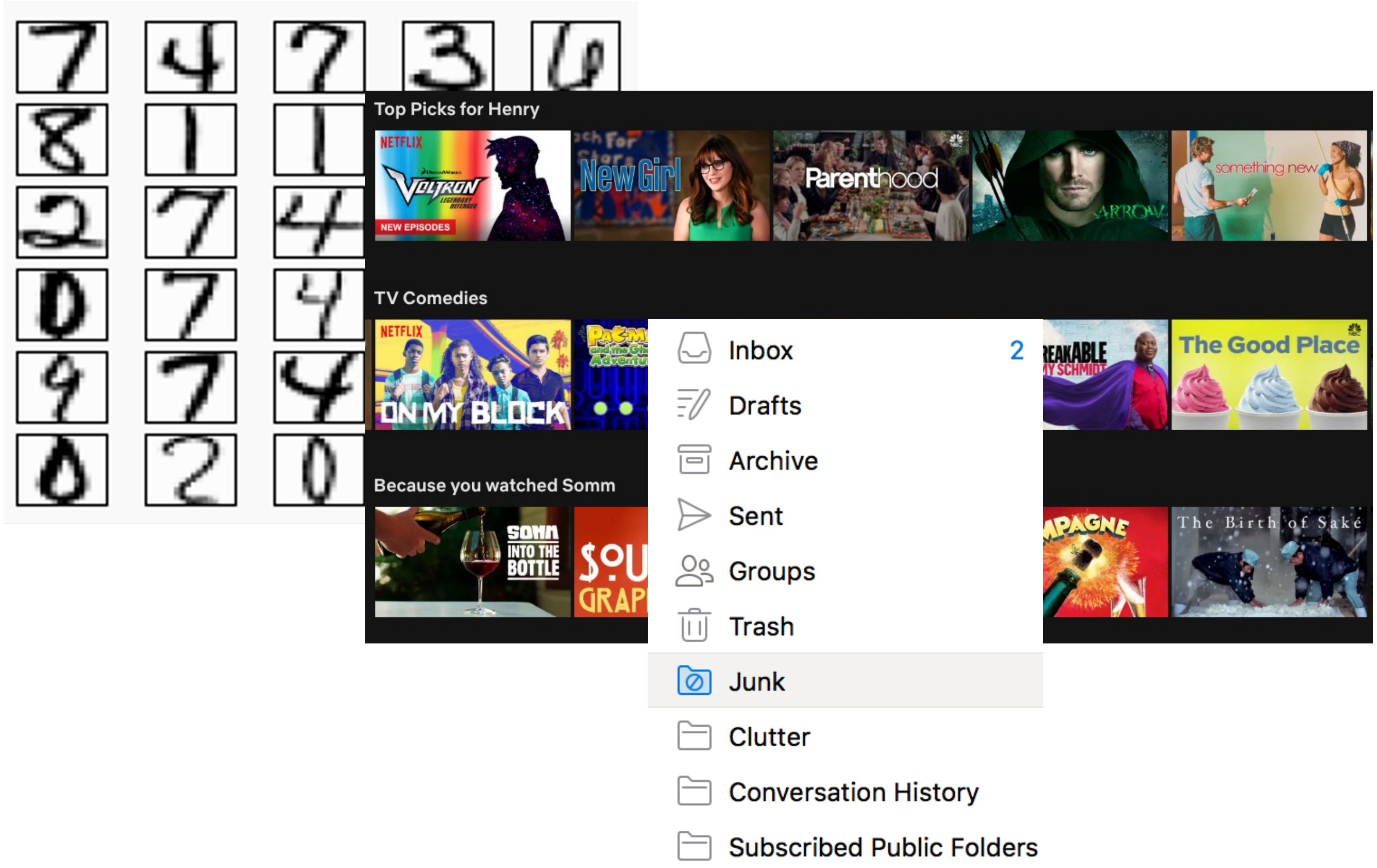
10-701: Introduction to Machine Learning Lecture 1 – Problem Formulation & Notation

Henry Chai

1/17/24

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~~ 10-701?

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

Defining a Machine Learning Task (Mitchell, 97)

- A computer program **learns** if its *performance*, *P*, at some *task*, *T*, improves with *experience*, *E*.
- Three components
  - Task, T
  - Performance metric, P
  - Experience, E

Defining a Machine Learning Task: Example

- Learning to approve loans/lines of credit

- Three components
  - Task, T

    decide whether or not to approve a line of credit
  - Performance metric, P

    % of approved loans that are paid off
  - Experience, E

    interviews with experienced loan officers

Defining a Machine Learning Task: Example

- Learning to approve loans/lines of credit
- Three components
  - Task, T

    estimating the probability that an applicant/ defaults on their loan

  - Performance metric, P

    total interest received over ~10 years

  - Experience, E

    historical records of loan applicants/ defaults

Things Machine Learning Isn't

- Neutral?

Things Machine Learning Isn't

- Neutral


Big Data: A Report on Algorithmic Systems, Opportunity, and Civil Rights

Executive Office of the President

May 2016

Source: https://obamawhitehouse.archives.gov/sites/default/files/microsites/ostp/2016_0504_data_discrimination.pdf

Things Machine Learning Isn't

- Neutral

OPPORTUNITIES AND CHALLENGES IN BIG DATA

The Assumption: Big Data is Objective

It is often assumed that big data techniques are unbiased because of the scale of the data and because the techniques are implemented through algorithmic systems. However, it is a mistake to assume they are objective simply because they are data-driven.[13]

The challenges of promoting fairness and overcoming the discriminatory effects of data can be grouped into the following two categories:

1) Challenges relating to ***data used as inputs*** to an algorithm; and
2) Challenges related to ***the inner workings of the algorithm itself***.

Source: https://obamawhitehouse.archives.gov/sites/default/files/microsites/ostp/2016_0504_data_discrimination.pdf

Defining a Machine Learning Task: Example

- Learning to predict flight delays

- Three components
  - Task, T
    determine whether or not a flight will be delayed
  - Performance metric, P
    # of correct predictions by airline
  - Experience, E
    historical data including weather patterns

Defining a Machine Learning Task: Example

- Learning to model the brain
- Three components
  - Task, T
    associate electric signals/EEG to human behavior
  - Performance metric, P
    prediction accuracy or % of correctly predicted behaviors
  - Experience, E
    collection of patients
    or
    one longitudinal data source i.e. a single person

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

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

data points

Our first Machine Learning Task

- Learning to diagnose heart disease

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

features / targets

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

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

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

features

labels

data points

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!

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
- The memorizer (typically) does not **generalize** well, i.e., it does not perform well on unseen data points
- In some sense, good generalization, i.e., the ability to make accurate predictions given a small training dataset, is the whole point of machine learning!

Notation

- Feature space, $\mathcal{X}$
- Label space, $\mathcal{Y}$
- (Unknown) Target function, $c^*: \mathcal{X} \rightarrow \mathcal{Y}$
- Training dataset:

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

- Data point:

$$\left(\boldsymbol{x}^{(n)}, y^{(n)}\right) = \left(x_1^{(n)}, x_2^{(n)}, \dots, x_D^{(n)}, y^{(n)}\right)$$

- Classifier, $h : \mathcal{X} \rightarrow \mathcal{Y}$
- Goal: find a classifier, h, that best approximates c^*

Notation

- 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):

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

indicator function

- Error rate:

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

Notation: Example

x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?	$\hat{y}$ 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

$\boldsymbol{x}^{(2)}$

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

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: example

- 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:

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

Our third Machine Learning Classifier: example

- 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: example

- 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:

$$h(\boldsymbol{x}') = h(x_1', \dots, x_D') = \begin{cases} \text{"Yes" if } x_1' = \text{"Yes"} \\ \text{"No" otherwise} \end{cases}$$

Our third Machine Learning Classifier: example

- 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: Questions

1. How can we pick which feature to split on?

2. Why stop at just one feature?

Key Takeaways

- Components of a machine learning problem
- Algorithmic bias
- Components of a labelled dataset for supervised learning
- Training vs. test datasets
- Majority vote classifier
- Decision stumps

Logistics: Course Website

https://www.cs.cmu.edu/~hchai2/courses/10701/

Logistics: Course Syllabus

https://www.cs.cmu.edu/~hchai2/courses/10701/#Syllabus

- This whole section is **required** reading

Logistics: Grading

https://www.cs.cmu.edu/~hchai2/courses/10701/#Syllabus

- 25% midterm & 25% final
- 25% homework assignments
  - First 4 assignments = 5% each
  - HW5 and HW6 are = 2.5% each
- 20% project
  - You must work on the project in groups of 2 or 3
- 5% participation
  - 5% (full credit) for 80% or greater poll participation
  - 3% for 65%-80% poll participation.
  - 1% for 50%-65% poll participation.
  - “Correctness” will not affect your participation grade
  - 50% credit for responses after lecture within 48 hours

Logistics: Late Policy

https://www.cs.cmu.edu/~hchai2/courses/10701/#Syllabus

- 6 grace days for use across all homework assignments
- Only 2 grace days may be used per homework
- Late submissions w/o grace days:
  - 1 day late = 50% multiplicative penalty
  - 2 days late = 25% multiplicative penalty
- No submissions accepted more than 2 days late
- Grace days cannot be applied to project deliverables

Logistics: Collaboration Policy

https://www.cs.cmu.edu/~hchai2/courses/10701/#Syllabus

- 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

https://www.cs.cmu.edu/~hchai2/courses/10701/#Syllabus

- Piazza, for course discussion: https://piazza.com/class/lr0i0sfzjdn2im
- Gradescope, for submitting homework assignments: https://www.gradescope.com/courses/695056
- Polleverywhere, for in-class participation: https://pollev.com/10701polls
- Canvas, for hosting the gradebook and lecture recordings: https://canvas.cmu.edu/courses/39031

Logistics: Lecture Schedule

https://www.cs.cmu.edu/~hchai2/courses/10701/#Schedule

Schedule

Lectures are the primary mode of content delivery in this course. Attending lectures is highly recommended; there will be regular in-class activities and polls which will constitute a small portion of your final grade. Engaging in these real-time activities can greatly improve your understanding of the material. Lectures will be recorded and made available to you after the fact. However, the primary purpose of these recordings is to allow you to refer back to the content; watching recordings in lieu of attending lectures is not encouraged.

Date	Topic	Slides	Readings/Resources
Wed, 1/17	Introduction: Notation & Problem Formulation		
Mon, 1/22	Decision Trees		
Wed, 1/24	KNNs & Model Selection		
Mon, 1/29	Linear Regression		
Wed, 1/31	MLE/MAP		
Mon, 2/5	Naïve Bayes		
Wed, 2/7	Logistic Regression & Regularization		
Mon, 2/12	Neural Networks		

Logistics: Exam Schedule

https://www.cs.cmu.edu/~hchai2/courses/10701/#Schedule

Schedule

Lectures are the primary mode of content delivery in this course. Attending lectures is highly recommended; there will be regular in-class activities and polls which will constitute a small portion of your final grade. Engaging in these real-time activities can greatly improve your understanding of the material. Lectures will be recorded and made available to you after the fact. However, the primary purpose of these recordings is to allow you to refer back to the content; watching recordings in lieu of attending lectures is not encouraged.

Date	Topic	Slides	Readings/Resources
⋮			
Mon, 3/18	Reinforcement Learning: Value & Policy Iteration		
Tue, 3/19	Midterm (Evening Exam: Details TBD)		
Wed, 3/20	Reinforcement Learning: Q-Learning & Deep RL		
⋮			
Wed, 4/24	Algorithmic Bias		
TBD, TBD	Final to be Scheduled by the Registrar		

Logistics: Recitations

https://www.cs.cmu.edu/~hchai2/courses/10701/#Recitations

Recitations

Attendance at recitations is not required, but strongly encouraged. Recitations will be interactive and focus on problem solving; we strongly encourage you to actively participate. A problem sheet will usually be released prior to the recitation. If you are unable to attend one or you missed an important detail, feel free to stop by office hours to ask the TAs about the content that was covered. Of course, we also encourage you to exchange notes with your peers.

Date	Topic	Handout
Fri, 1/19	No Recitation	
Fri, 1/26	Recitation 1: Decision Trees & kNNs	
Fri, 2/2	Recitation 2: Linear Regression & MLE/MAP	
Fri, 2/9	Recitation 3: Naïve Bayes & Logistic Regression	

Logistics: Homework Assignments

https://www.cs.cmu.edu/~hchai2/courses/10701/#Homework

Assignments

Our homework assignments are an opportunity for you all to reason about and build/experiment with some of the models that we introduce in class. All programming questions must be completed in Python and you must use LaTeX to typeset your responses to the written questions. You will submit both your code and your written responses using Gradescope; note that each assignment will have separate submissions for the code and the written portion.

Release Date	Topic	Files	Due Date
Wed, 1/24	HW1: Decision Trees & kNNs		Fri, 2/2 at 11:59 PM
Wed, 2/7	HW2: Linear Regression, Naïve Bayes & Logistic Regression		Fri, 2/16 at 11:59 PM
Fri, 2/16	HW3: Neural Networks		Fri, 2/26 at 11:59 PM
Wed, 2/28	HW4: Deep Learning in PyTorch		Fri, 3/15 at 11:59 PM
Fri, 3/22	HW5: Unsupervised Learning & Reinforcement Learning		Mon, 4/1 at 11:59 PM
Wed, 4/10	HW6: Learning Theory & Ensemble Methods		Fri, 4/19 at 11:59 PM

https://www.cs.cmu.edu/~hchai2/courses/10701/#Calendar

Logistics: Office Hours