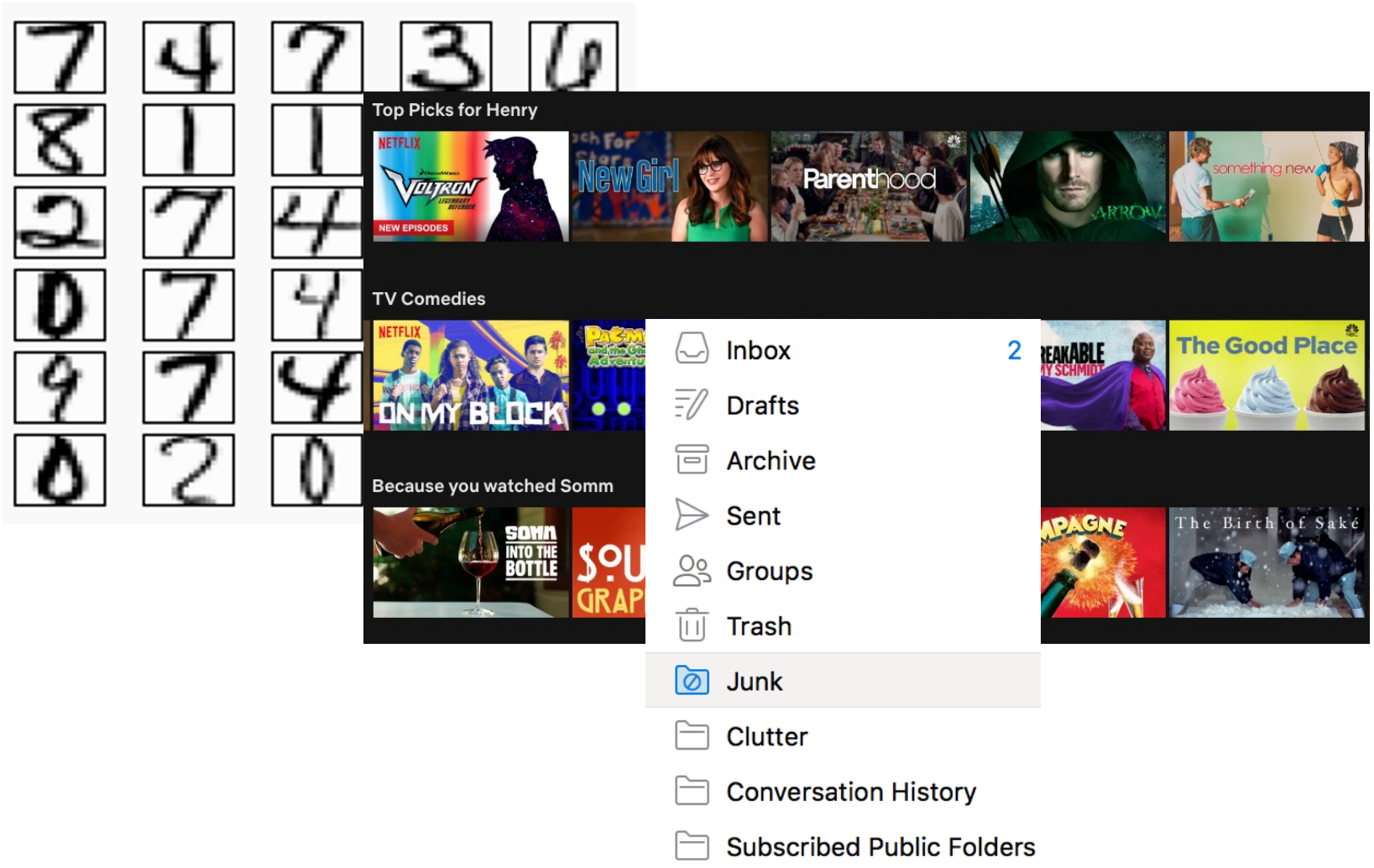
# 10-301/601: Introduction to Machine Learning Lecture 1 – Problem Formulation & Notation

Henry Chai & Matt Gormley

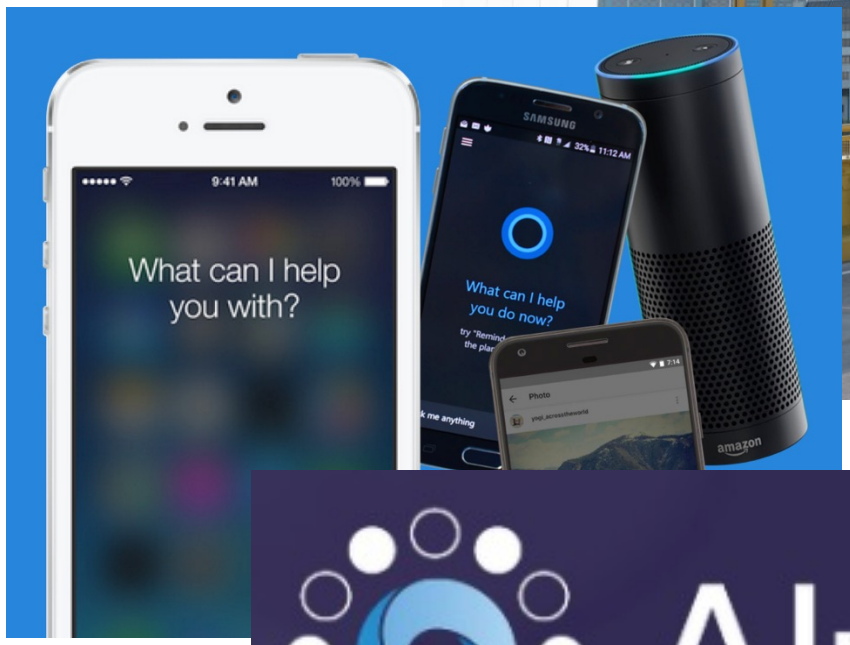
1/13/25

# What is Machine Learning?

# Machine Learning (A long long time ago…)



# Machine Learning (A short time ago…)




AlphaGo

# Machine Learning (Now)

# Machine Learning (Now)

# Machine Learning (Now – literally yesterday)


Source: https://www.bing.com/images/create?FORM=GERRLP

Source: https://chat.openai.com/

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

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

# What is Machine Learning?


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

# 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
  - Performance metric, P
  - Experience, E

# Defining a Machine Learning Task: Example

- Learning to approve loans/lines of credit
- Three components
  - Task, T
  - Performance metric, P
  - Experience, E

# Example Learning Problems

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

1. Task, *T*:

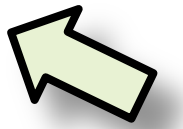
2. Performance measure, *P*:

3. Experience, *E*:

# Example Learning Problems

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

1. Task, *T*:

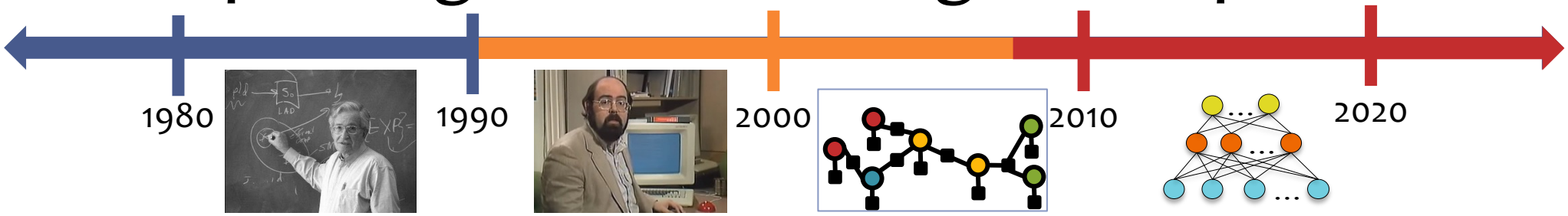


Given a transcribed sentence `x` predict the command `y`

Example:

```
x = "Give me directions to Starbucks"
y = DIRECTIONS(here, nearest(Starbucks))
```

# Capturing the Knowledge of Experts




**Solution #1: Expert Systems**

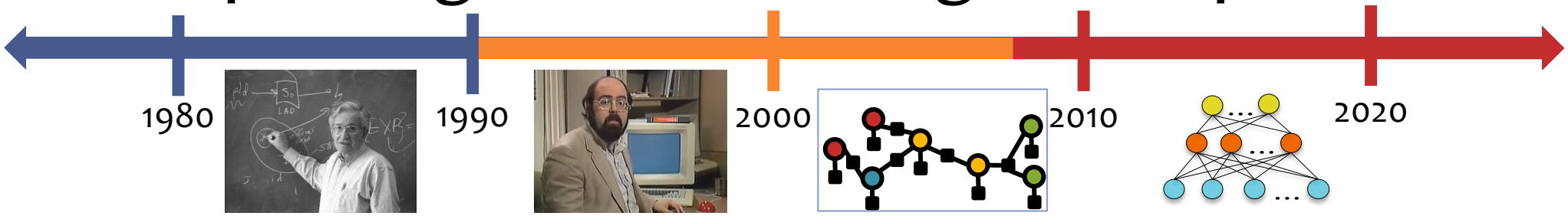
- Over 20 years ago, we had rule-based systems:
  1. Put a bunch of linguists in a room
  2. Have them think about the structure of their native language and write down the rules they devise



| **Introspection…** |
| --- |
| x = "Give me directions to Starbucks" |
| x = "Send Jill a txt asking for directions" |
| x = "Play the best hit music by TXT" |
| x = "How do I get to Pitt's Department of Music" |

| **Rules…** |
| --- |
| ~~if "directions" in x:<br>  type = DIRECTIONS()~~ |
| ~~if "txt" in x:<br>  type = TXTMSG()<br>elif "directions" in x:<br>  type = DIRECTIONS()~~ |
| ~~if "music" in x:<br>  type = MUSIC()<br>elif "txt" in x:<br>  type = TXTMSG()<br>elif "directions" in x:<br>  type = DIRECTIONS()~~ |

# Capturing the Knowledge of Experts




## Solution #1: Expert Systems

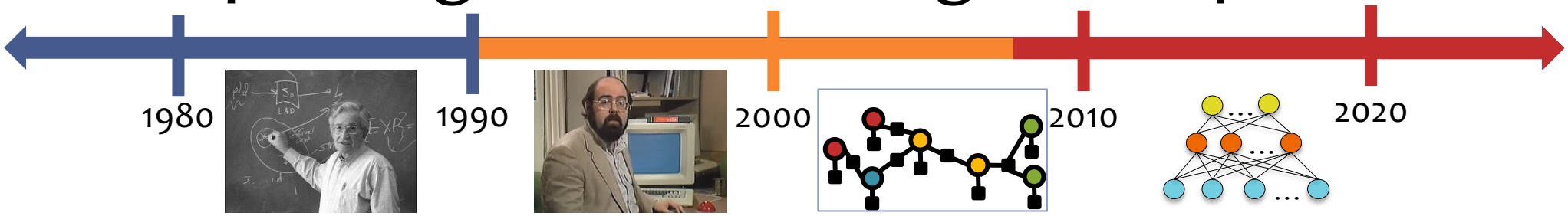
- Over 20 years ago, we had rule-based systems:
  1. Put a bunch of linguists in a room
  2. Have them think about the structure of their native language and write down the rules they devise



| **Introspection…** |
| --- |
| x = "Give me directions to Starbucks" |
| x = "How do I get to Starbucks?" |
| x = "Where is the nearest Starbucks?" |
| x = "I need directions to Starbucks" |
| x = "Is there a Starbucks nearby? |
| x = "Starbucks now!" |

| **Rules…** |
| --- |
| if x matches "give me directions to Z":<br>  cmd = DIRECTIONS(here, nearest(Z)) |
| if x matches "how do i get to Z":<br>  cmd = DIRECTIONS(here, nearest(Z)) |
| if x matches "where is the nearest Z":<br>  cmd = DIRECTIONS(here, nearest(Z)) |
| if x matches "I need directions to Z":<br>  cmd = DIRECTIONS(here, nearest(Z)) |
| if x matches "Is there a Z nearby":<br>  cmd = DIRECTIONS(here, nearest(Z)) |
| if x matches "Z now!":<br>  cmd = DIRECTIONS(here, nearest(Z)) |

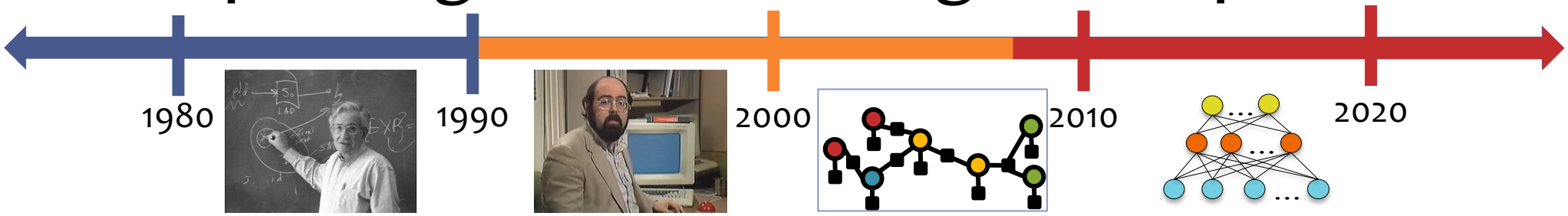
# Capturing the Knowledge of Experts




**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? | $x^{(3)}$: Send a text to John that I'll be late |
| $y^{(1)}$: DIRECTIONS(here, nearest(Starbucks)) | $y^{(3)}$: TXTNSG(John, I'll be late) |
| $x^{(2)}$: Show me the closest Starbucks | $x^{(4)}$: Set an alarm for seven in the morning |
| $y^{(2)}$: MAP(nearest(Starbucks)) | $y^{(4)}$: 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.

Example: 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

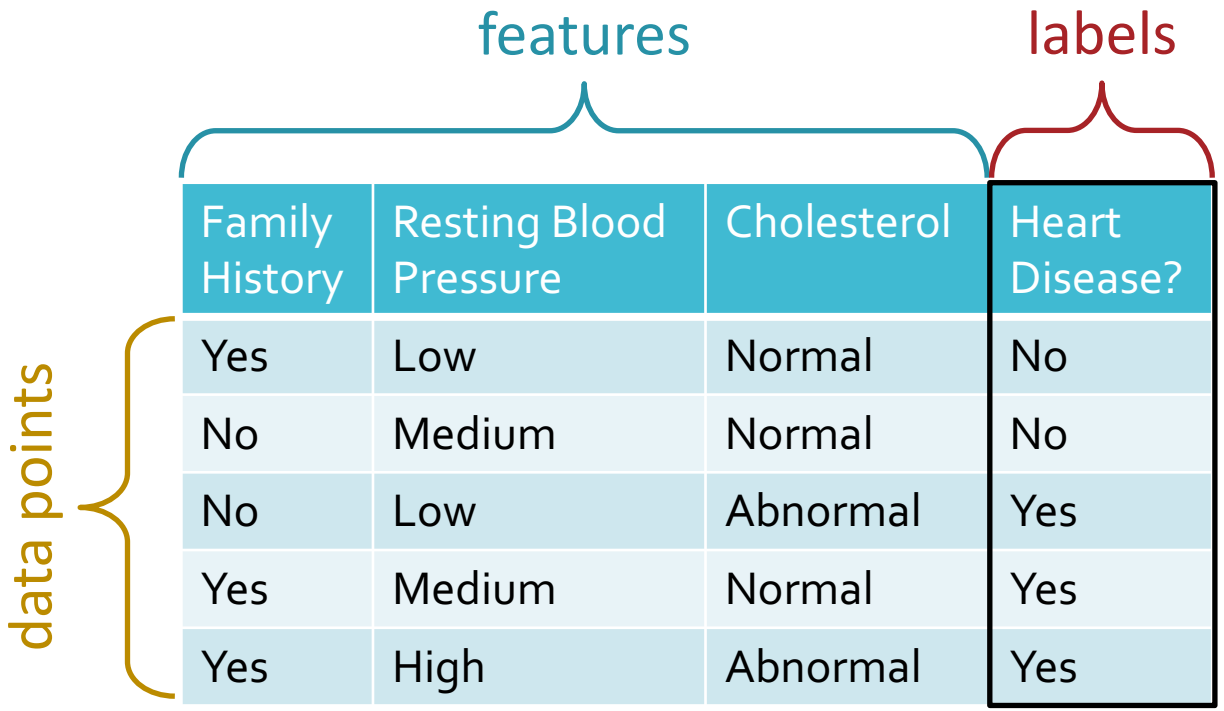
data points

| 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**




| 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)** **classification 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**

features

targets

data points

| 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 |

# 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

features | labels

data points

| 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 |

- The **training error rate** is $2/5$

# 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 |

- This classifier completely ignores the features…

# 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!

# 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 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/m5gpeq5rfa3sg
- Gradescope, for submitting homework assignments: https://www.gradescope.com/courses/937548
- 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=%22ae47ec9c-b29d-496a-9c3a-b25d0123781a%22

# Logistics: Lecture Schedule

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

| Date | Lecture | Readings | Announcements |
|---|---|---|---|
| | Classification & Regression | | |
| Mon, 13-Jan | Lecture 1 : Course Overview | • *Command Line and File I/O Tutorial*. 10601 Course Staff (2020).<br>• *10601 Learning Objectives*. Matt Gormley (2023).<br>• *Math Resources*. 10601 Course Staff (2023). | HW1 Out |
| Wed, 15-Jan | Lecture 2 : Machine Learning as Function Approximation | • *10601 Notation Crib Sheet*. Matt Gormley (2023). | |
| Fri, 17-Jan | Recitation: HW1 | | |
| Mon, 20-Jan | (MLK Day - No Class) | | |
| Wed, 22-Jan | Lecture 3 : Decision Trees | • *Visual Information Theory*. Christopher Olah (2015). blog.<br>• *Decision Trees*. Hal Daumé III (2017). CIML, Chapter 1. | HW1 Due<br>HW2 Out |
| Fri, 24-Jan | 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)

# Wait, was there something about a HW in that lecture schedule?

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

| Date | Lecture | Readings | Announcements |
|---|---|---|---|
| Classification & Regression | | | |
| Mon, 13-Jan | Lecture 1 : Course Overview | • *Command Line and File I/O Tutorial*. 10601 Course Staff (2020).<br>• *10601 Learning Objectives*. Matt Gormley (2023).<br>• *Math Resources*. 10601 Course Staff (2023). | HW1 Out |
| Wed, 15-Jan | Lecture 2 : Machine Learning as Function Approximation | • *10601 Notation Crib Sheet*. Matt Gormley (2023). | |
| Fri, 17-Jan | Recitation: HW1 | | |
| Mon, 20-Jan | (MLK Day - No Class) | | |
| Wed, 22-Jan | Lecture 3 : Decision Trees | • *Visual Information Theory*. Christopher Olah (2015). blog.<br>• *Decision Trees*. Hal Daumé III (2017). CIML, Chapter 1. | HW1 Due<br>HW2 Out |
| Fri, 24-Jan | Recitation: HW2 | | |

# FAQ: Am I prepared to take this course?

- Answer: We don't know!
- But we have designed a way for you to assess your background knowledge for yourselves!
- HW1 released 1/13 (today!), due 1/22 at 11:59 PM
- Most HWs consist of two parts:
  - a written component
  - a programming component
- **Unique policies for HW1 only:**
  - Any written submission that receives a grade of 90% or higher will receive full credit
  - Any written submission that receives less than 90% can be resubmitted once for a (potentially) higher grade
  - You will have unlimited submissions to the autograder

# 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)
- 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: PAC Learning and Ethics (written)
- Homework 7: RNNs (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: Exam Schedule

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

| | | | |
|---|---|---|---|
| Mon, 17-Feb | Lecture 11 : Neural Networks | • *Deep Feedforward Networks*. Ian Goodfellow and Yoshua Bengio and Aaron Courville (2016). Deep Learning, Chapter 6.1-6.4. | |
| Mon, 17-Feb | Exam 1 (evening exam, details will be announced on Piazza) | | HW4 Out |
| Wed, 26-Mar | Lecture 20 : Reinforcement Learning: MDPs | • *Reinforcement Learning: A Survey*. Kaelbling, et al (1996). | |
| Wed, 26-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: Office Hours

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