Computer
Science

# CSC380: Principles of Data Science

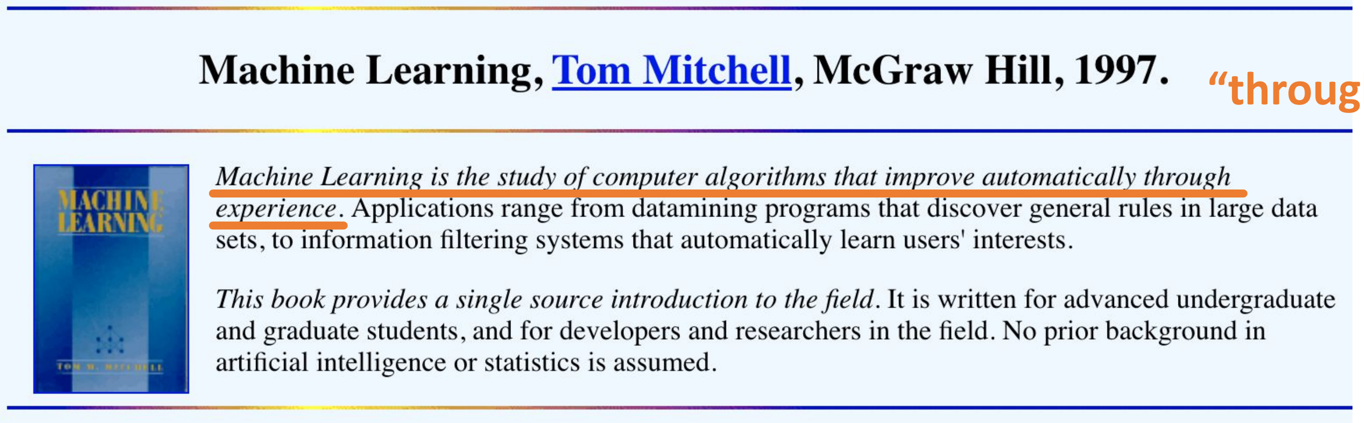
## Predictive Modeling and Classification 1

Xinchen Yu

# Introduction to Machine Learning

# What is machine learning?

- **Tom Mitchell** established Machine Learning Department at CMU (2006).



**Machine Learning, Tom Mitchell, McGraw Hill, 1997.**

*Machine Learning is the study of computer algorithms that improve automatically through experience*. Applications range from datamining programs that discover general rules in large data sets, to information filtering systems that automatically learn users' interests.

*This book provides a single source introduction to the field*. It is written for advanced undergraduate and graduate students, and for developers and researchers in the field. No prior background in artificial intelligence or statistics is assumed.

"through experience"

- A bit outdated with recent trends, but still has interesting discussion (and easy to read).
- A subfield of **Artificial Intelligence** – you want to perform nontrivial, smart tasks. The difference from the traditional AI is "**how**" you build a computer program to do it.

# Traditional AI vs Machine Learning (ML)

- **Traditional AI**: *you* encode the knowledge (e.g., logic statements/rules), and the *machine* executes it.
  - e.g., if there is feather-like texture with two eyes and a beak, classify it as a bird.
  - Advancements in automated ‘**inference**’ like “if a -> b and b-> c, then a-> c”. => ’expert system’

- **ML**: Given a set of input and output pairs (e.g., animal picture + label), and train a **function** (a set of logical statements / a neural network) that maps the input to the output accurately.
  - As the “big data” era comes, data is abundant => turns out, better than systems based on hand-coded domain knowledge!
  - “statistical” approach // data-driven approach

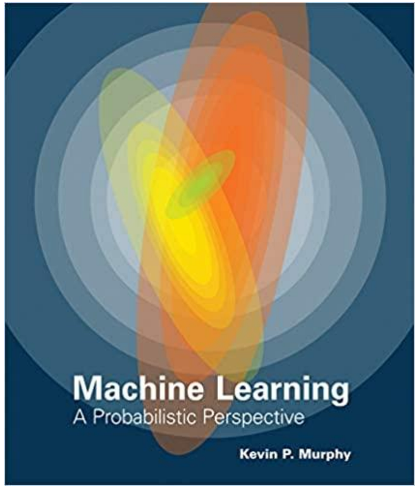
> *“Every time I fire a linguist, the performance of the speech recognizer goes up.”*
> – *1988,* Frederick Jelinek, a Czech-American researcher in information theory & speech recognition.

# Textbooks

*We will use a more recent textbook for readings*

*Takes a* ***probabilistic approach*** *to machine learning*

*Consistent with the goals of data science in this class*



Murphy, K. "Machine Learning: A Probabilistic Perspective." MIT press, 2012

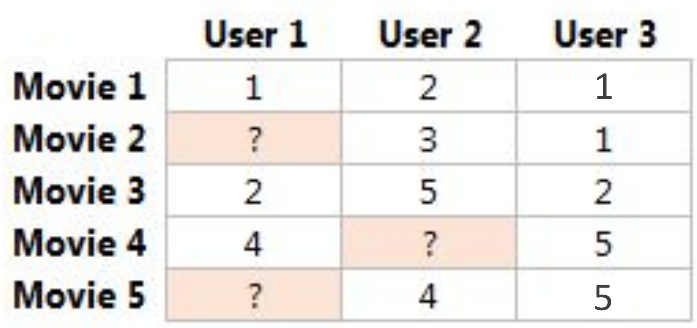
( UA Library )

# AI Task 1: Image classification

- Predefined categories: $C$ = {cat, dog, lion, …}
- Given an image, classify it as one of the categories $c \in C$ with the highest accuracy.
- **<u>Use</u>**: sorting/searching images by category.
- **<u>Other example</u>**: categorize types of stars/events in the Universe (images taken from large surveying telescopes)

# AI Task 2: Recommender systems

- Predict how user would rate a movie
- **<u>Use</u>**: For each user, pick an unwatched movie with the high predicted ratings.
- **<u>Idea</u>**: compute user-user similarity or movie-movie similarity, then compute a <u>weighted average</u>.

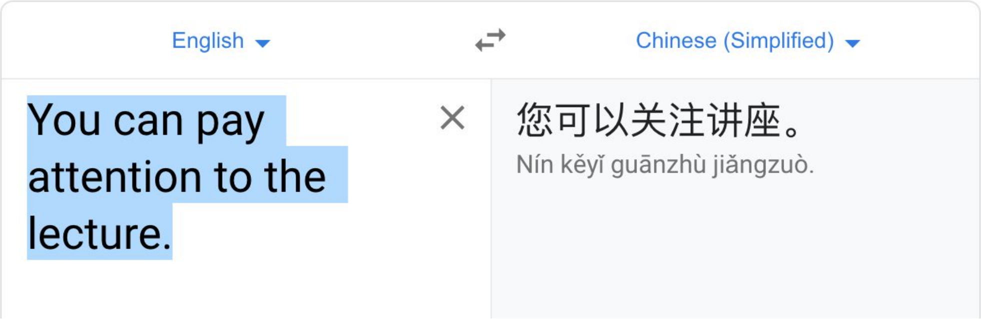


| | User 1 | User 2 | User 3 |
|---|---|---|---|
| **Movie 1** | 1 | 2 | 1 |
| **Movie 2** | ? | 3 | 1 |
| **Movie 3** | 2 | 5 | 2 |
| **Movie 4** | 4 | ? | 5 |
| **Movie 5** | ? | 4 | 5 |

“collaborative filtering”

# AI Task 3: Machine translation

- No need to explain how useful it is.

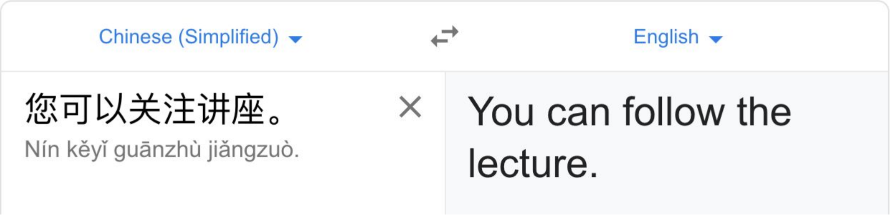


English → Chinese (Simplified)

You can pay attention to the lecture.

您可以关注讲座。

Nín kěyǐ guānzhù jiǎngzuò.



Chinese (Simplified) → English

您可以关注讲座。

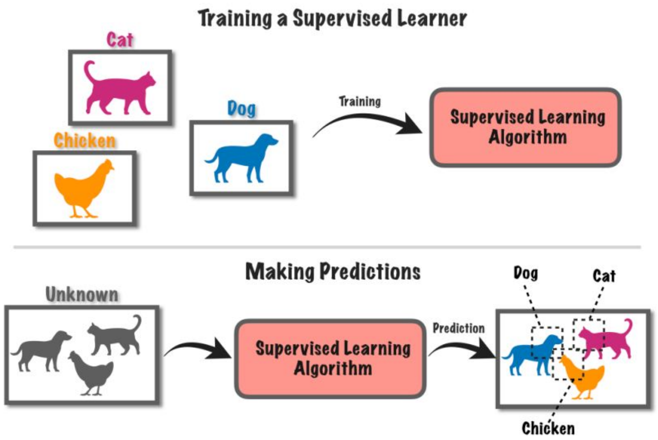
Nín kěyǐ guānzhù jiǎngzuò.

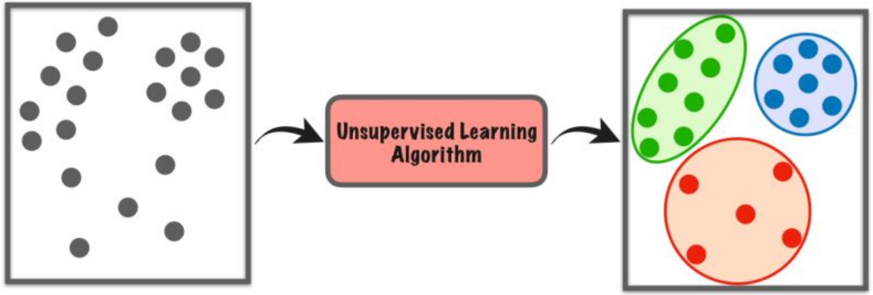
You can follow the lecture.

# Supervised vs Unsupervised Learning

- **Supervised Learning** - Training data consist of inputs and outputs
  - Classification, regression, translation, …

- **Unsupervised Learning** – Training data only contain inputs
  - Clustering, dimensionality reduction, segmentation, …

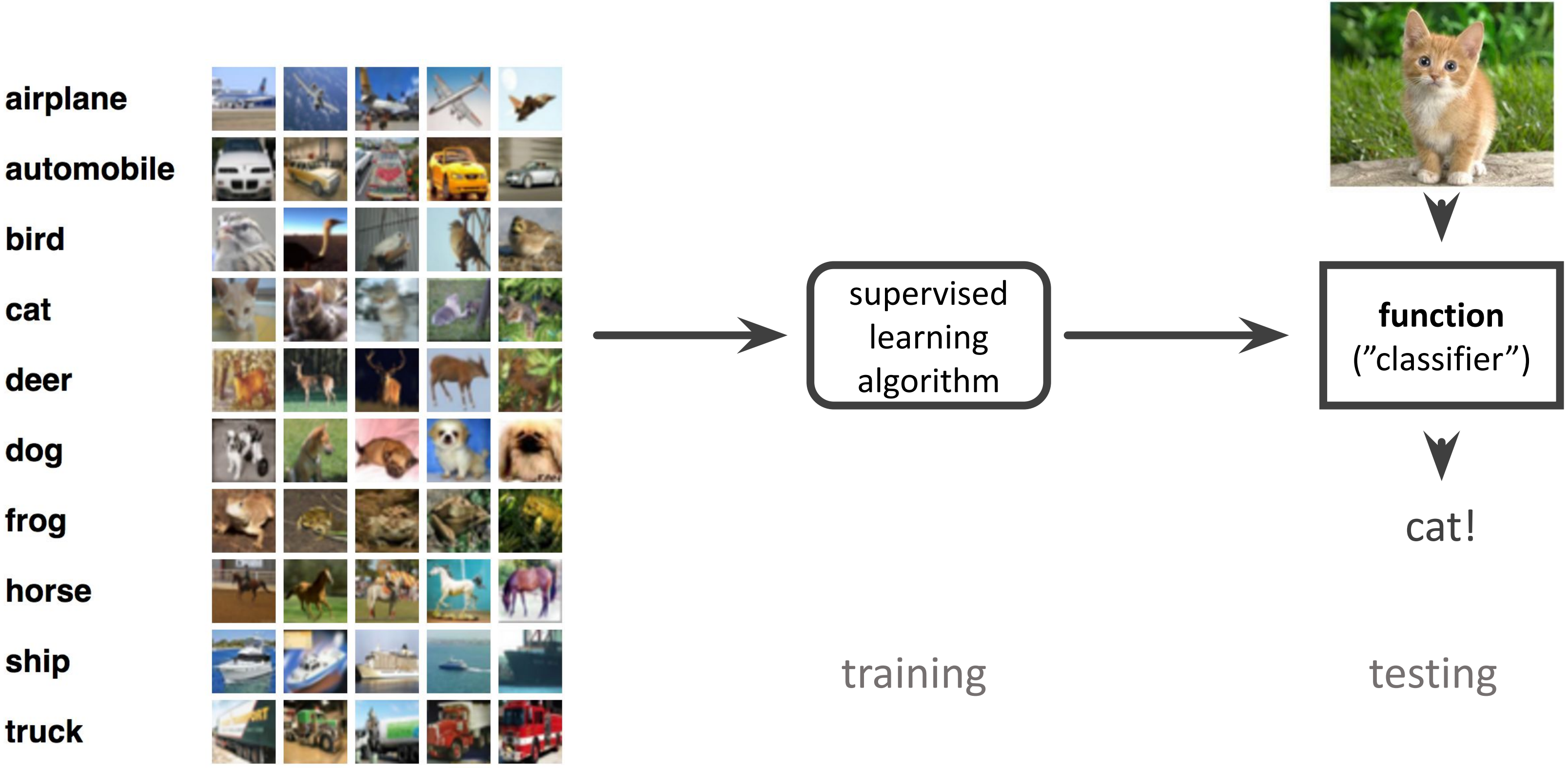

Training a Supervised Learner
Cat
Dog
Chicken
Training
Supervised Learning Algorithm
Making Predictions
Unknown
Supervised Learning Algorithm
Prediction
Dog
Cat
Chicken



# Supervised learning

example = data point
labeled = categorized

- Train data: dataset comprised of *labeled examples*: a pair of (input, label)



airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck

supervised learning algorithm

**function**
("classifier")

cat!

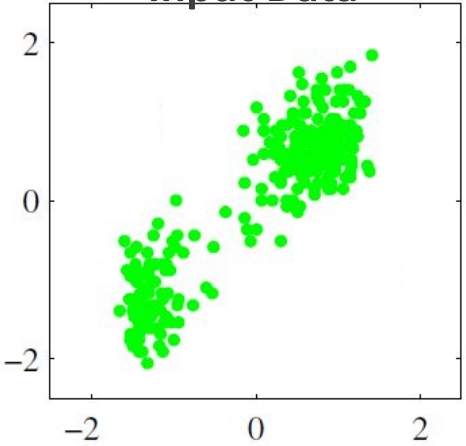
training

testing

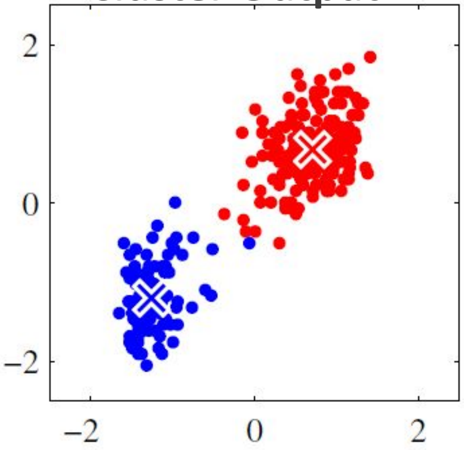
# Unsupervised learning: Clustering

Identify groups (clusters) of similar data

**Input Data**



**Cluster Output**



Useful for interpreting large datasets

Clusters are assigned arbitrary labels (e.g. 1, 2, …, K).
=> afterwards, you may look at the data and name each group.

Common clustering algorithms: K-means, Expectation Maximization (EM)

# Decision Trees

# Majority Vote Classifier

The most basic classifier you can think of.

How to train:

- Given: A (train) dataset with m data points $\{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$ with C classes.
- Compute the most common class $c^*$ in the dataset.

$$c^* = \arg \max_{c \in \{1,\ldots,C\}} \sum_{i=1}^{m} \mathbf{I}\{y^{(i)} = c\}$$

- Output a classifier $f(x) = c^*$.

Stupid enough classifier! Always try to beat this classifier.

Often, state-of-the-art ML algorithms perform barely better than the majority vote classifier..

⇒ happens when there is no association between features and labels in the dataset

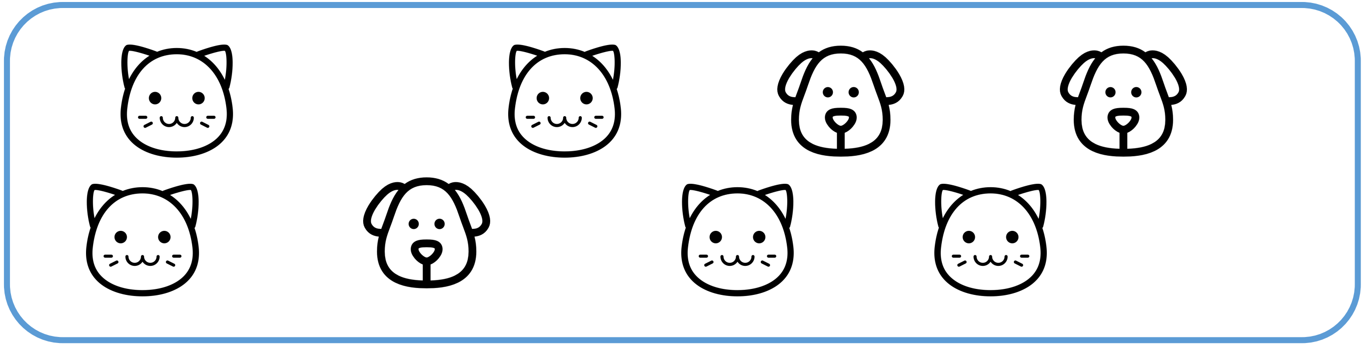
# Train set accuracy

- Suppose the ML algorithm has trained a function $f$ using the dataset $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$ where $x^{(i)}$ is input and $y^{(i)}$ is label.
- Train set accuracy:

$$\widehat{acc}(f) \coloneqq \frac{1}{m}\sum_{i=1}^{m}\mathbf{I}\{f(x^{(i)}) = y^{(i)}\}$$

# Train set accuracy

If the model is majority vote classifier..

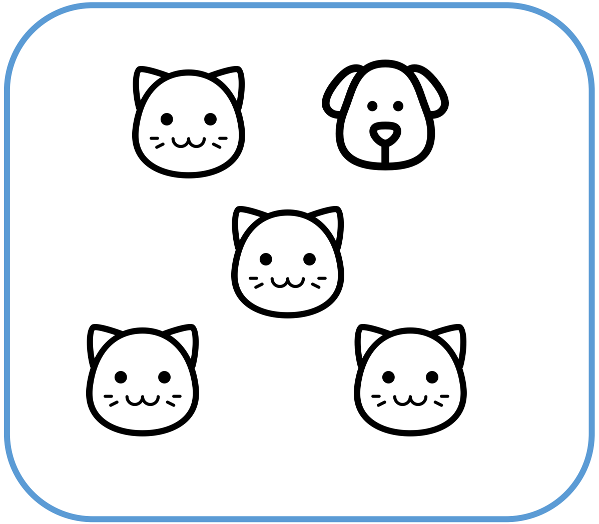


Majority vote: cat

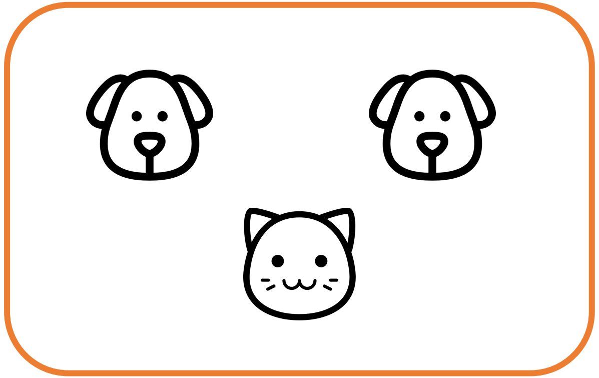
Q: what is the accuracy? $\frac{5}{8}$

# Train set accuracy

Suppose the model is a little bit smarter than majority vote classifier..



Majority vote: cat



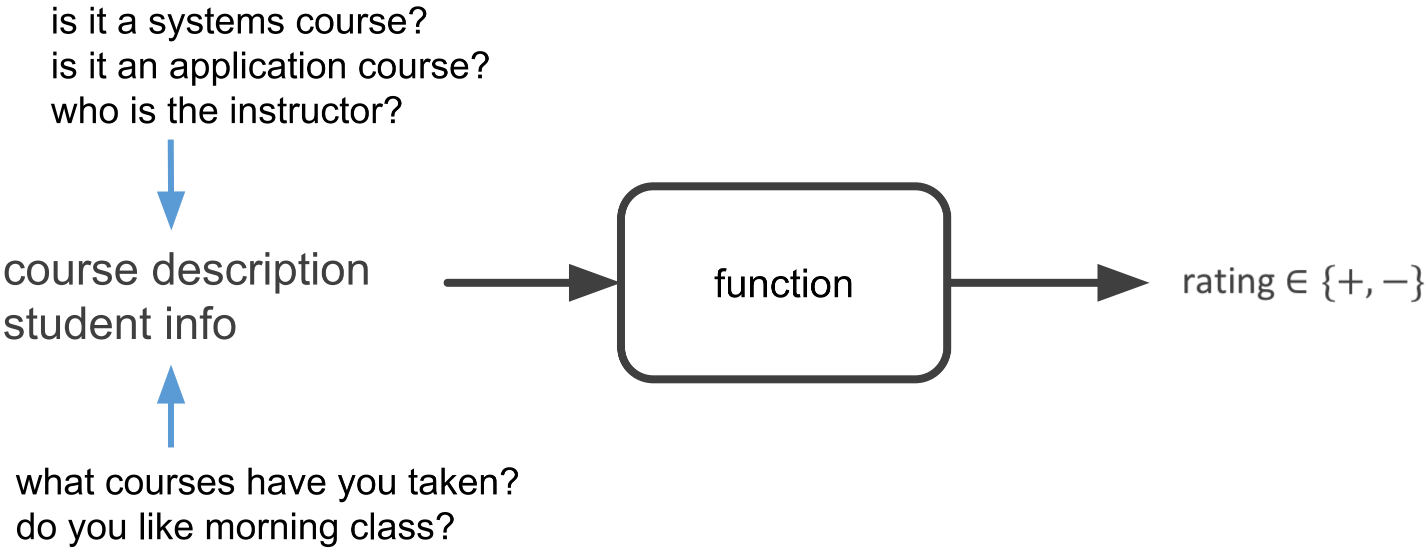
Majority vote: dog

Q: what is the accuracy?

$$\frac{6}{8} = \frac{5}{8} \cdot \frac{4}{5} + \frac{3}{8} \cdot \frac{2}{3} = \frac{3}{4}$$

# Example: course recommendation

- Data: given a student, know the preferences and a set of courses already took
- Task: predict if the student like the course or not

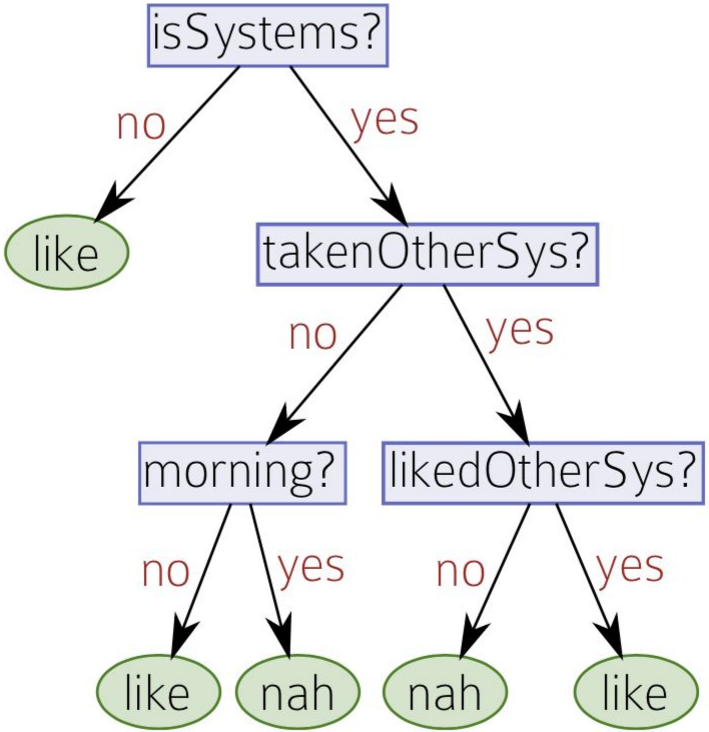


is it a systems course?
is it an application course?
who is the instructor?

course description
student info

rating $\in \{+, -\}$

what courses have you taken?
do you like morning class?



Wouldn’t it be nice to construct such a tree automatically by a computer algorithm?

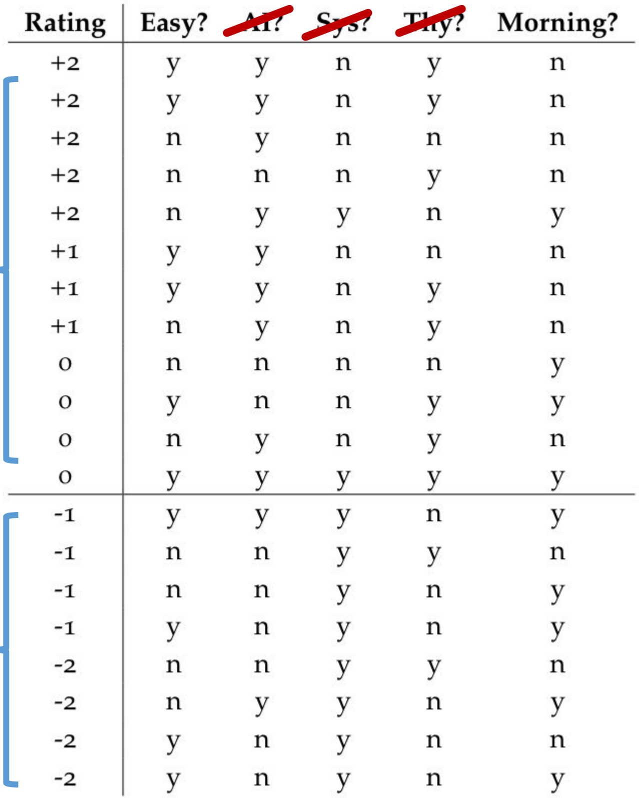
Wouldn’t it be nice if it accurately predicts?

You can, if you have data!

HasTakenPrereqs (=: Prereq)

HasTakenACourseFromTheSameLecturer (=: Lecturer)

HasLabs



| Rating | Easy? | ~~AI?~~ | ~~Sys?~~ | ~~Thy?~~ | Morning? |
|---|---|---|---|---|---|
| +2 | y | y | n | y | n |
| +2 | y | y | n | y | n |
| +2 | n | y | n | n | n |
| +2 | n | n | n | y | n |
| +2 | n | y | y | n | y |
| +1 | y | y | n | n | n |
| +1 | y | y | n | y | n |
| +1 | n | y | n | y | n |
| 0 | n | n | n | n | y |
| 0 | y | n | n | y | y |
| 0 | n | y | n | y | n |
| 0 | y | y | y | y | y |
| -1 | y | y | y | n | y |
| -1 | n | n | y | y | n |
| -1 | n | n | y | n | y |
| -1 | y | n | y | n | y |
| -2 | n | n | y | y | n |
| -2 | n | y | y | n | y |
| -2 | y | n | y | n | n |
| -2 | y | n | y | n | y |

consider it to be 'like'

consider it to be 'dislike'

For example, this table is data D;
Each row is a course you have rated;
$x^{(i)}$ is a sequence of 5 yes/no for the i-th course;
$y^{(i)}$ is the sign of rating for the i-th course.

Define the data $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$

$\in \{y, n\}^d$

$\in \{+, -\}$

Each dimension of $x^{(i)}$ is called a **feature**.
$x^{(i)}$ is called a **feature vector**.

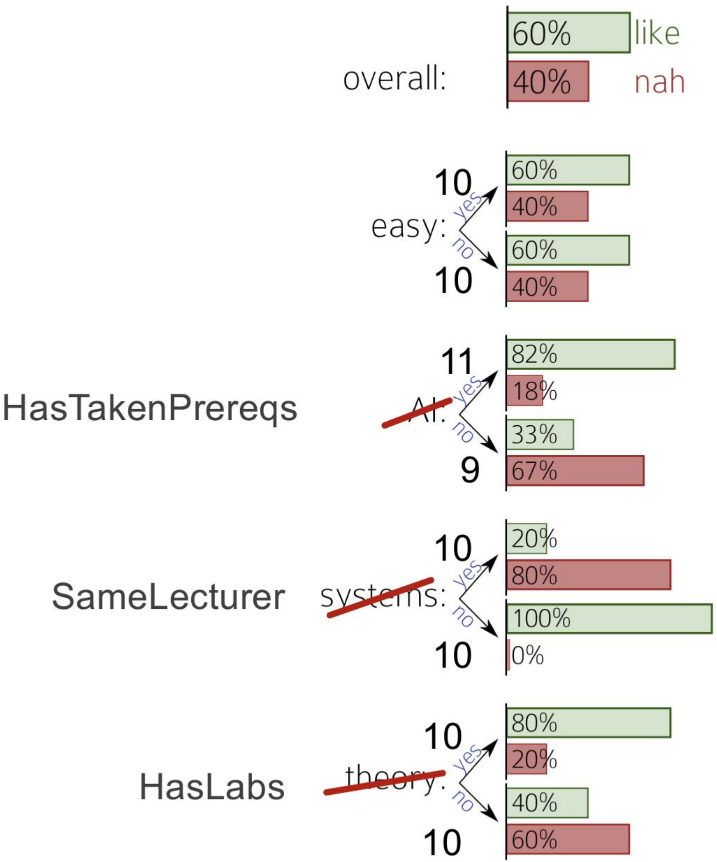
# How to Train a Tree

- Main principle: Find a tree that has a high train set accuracy

$$\widehat{acc}(f) = \frac{1}{m}\sum_{i=1}^{m} \mathbf{I}\{f\left(x^{(i)}\right) = y^{(i)}\}$$

- This is essentially the main principle governing pretty much all the machine learning algorithms!
  - "Empirical risk minimization" principle
    (empirical risk := 1 – train_accuracy)

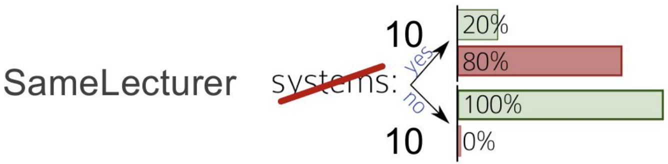
# How to construct a tree: choosing root



Prereqs ↓ Lecturer ↓ HasLabs ↓

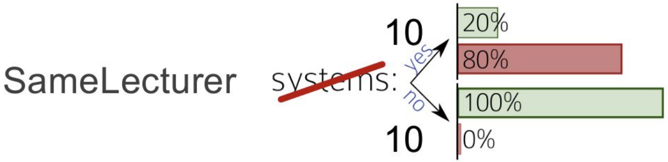
| Rating | Easy? | ~~AI?~~ | ~~Sys?~~ | ~~Thy?~~ | Morning? |
|---|---|---|---|---|---|
| +2 | y | y | n | y | n |
| +2 | y | y | n | y | n |
| +2 | n | y | n | n | n |
| +2 | n | n | n | y | n |
| +2 | n | y | y | n | y |
| +1 | y | y | n | n | n |
| +1 | y | y | n | y | n |
| +1 | n | y | n | y | n |
| 0 | n | n | n | n | y |
| 0 | y | n | n | y | y |
| 0 | n | y | n | y | n |
| 0 | y | y | y | y | y |
| -1 | y | y | y | n | y |
| -1 | n | n | y | y | n |
| -1 | n | n | y | n | y |
| -1 | y | n | y | n | y |
| -2 | n | n | y | y | n |
| -2 | n | y | y | n | y |
| -2 | y | n | y | n | n |
| -2 | y | n | y | n | y |

# How to construct a tree



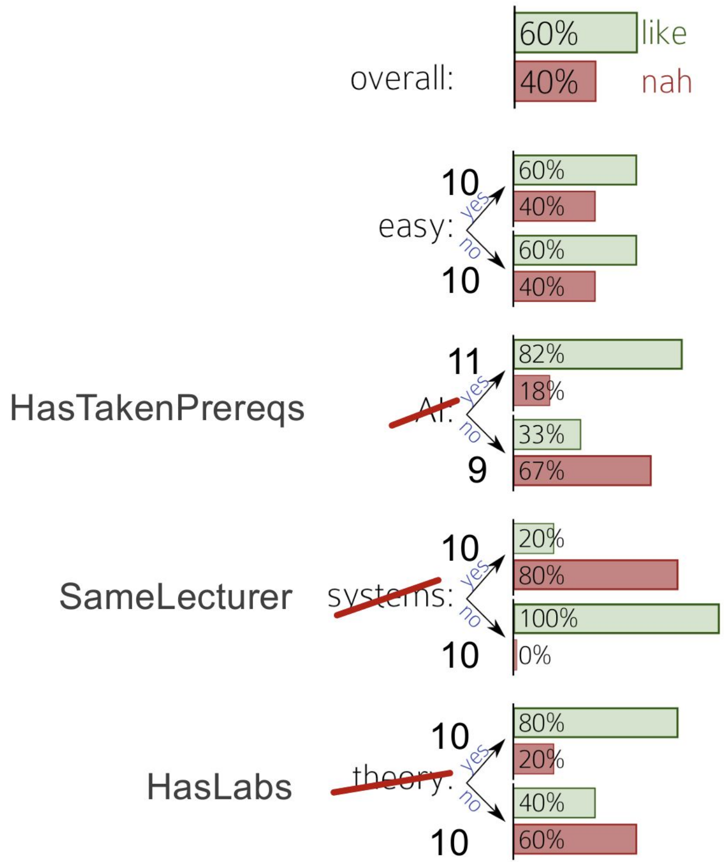
| | | Prereqs | Lecturer | HasLabs | |
|---|---|---|---|---|---|
| **Rating** | **Easy?** | ~~**AI?**~~ | ~~**Sys?**~~ | ~~**Thy?**~~ | **Morning?** |
| +2 | y | y | n | y | n |
| +2 | y | y | n | y | n |
| +2 | n | y | n | n | n |
| +2 | n | n | n | y | n |
| +2 | n | y | y | n | y |
| +1 | y | y | n | n | n |
| +1 | y | y | n | y | n |
| +1 | n | y | n | y | n |
| 0 | n | n | n | n | y |
| 0 | y | n | n | y | y |
| 0 | n | y | n | y | n |
| 0 | y | y | y | y | y |
| -1 | y | y | y | n | y |
| -1 | n | n | y | y | n |
| -1 | n | n | y | n | y |
| -1 | y | n | y | n | y |
| -2 | n | n | y | y | n |
| -2 | n | y | y | n | y |
| -2 | y | n | y | n | n |
| -2 | y | n | y | n | y |

# How to construct a tree



| | | Prereqs | Lecturer | HasLabs | |
|---|---|---|---|---|---|
| **Rating** | **Easy?** | ~~**AI?**~~ | ~~**Sys?**~~ | ~~**Thy?**~~ | **Morning?** |
| +2 | y | y | n | y | n |
| +2 | y | y | n | y | n |
| +2 | n | y | n | n | n |
| +2 | n | n | n | y | n |
| +2 | n | y | y | n | y |
| +1 | y | y | n | n | n |
| +1 | y | y | n | y | n |
| +1 | n | y | n | y | n |
| 0 | n | n | n | n | y |
| 0 | y | n | n | y | y |
| 0 | n | y | n | y | n |
| 0 | y | y | y | y | y |
| -1 | y | y | y | n | y |
| -1 | n | n | y | y | n |
| -1 | n | n | y | n | y |
| -1 | y | n | y | n | y |
| -2 | n | n | y | y | n |
| -2 | n | y | y | n | y |
| -2 | y | n | y | n | n |
| -2 | y | n | y | n | y |

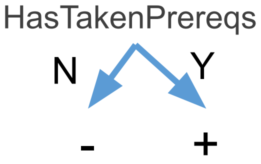
# How to construct a tree: choosing root



Baseline: majority vote classifier

Q: What is the train set accuracy? 12/20 = 0.60

Suppose we place the node HasTakenPrereqs at the root. Set the prediction at each leaf node as the majority vote.



What is the train set accuracy now?

$$\frac{9}{20}\cdot\frac{6}{9}+\frac{11}{20}\cdot\frac{9}{11}=\frac{15}{20}=0.75$$

improved!

# How to construct a tree

What is the train set accuracy now?

$$\frac{9}{20}\cdot\frac{6}{9}+\boxed{\frac{11}{20}}\cdot\frac{9}{11}=\frac{15}{20}=0.75$$

Accuracy for two groups:

- Prereqs = yes (11): 9/11
- Prereqs = no (9): 6/9

For the 11 people prereqs = y, use the majority vote label **like** (9 like, 2 dislike).

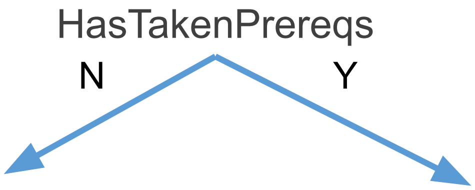
Predicted label for 11 people is **like**, 9 people are correctly predicted.

Prereqs Lecturer HasLabs

| Rating | Easy? | ~~AI?~~ | ~~Sys?~~ | ~~Thy?~~ | Morning? |
|---|---|---|---|---|---|
| +2 | y | y | n | y | n |
| +2 | y | y | n | y | n |
| +2 | n | y | n | n | n |
| +2 | n | n | n | y | n |
| +2 | n | y | y | n | y |
| +1 | y | y | n | n | n |
| +1 | y | y | n | y | n |
| +1 | n | y | n | y | n |
| 0 | n | n | n | n | y |
| 0 | y | n | n | y | y |
| 0 | n | y | n | y | n |
| 0 | y | y | y | y | y |
| -1 | y | y | y | n | y |
| -1 | n | n | y | y | n |
| -1 | n | n | y | n | y |
| -1 | y | n | y | n | y |
| -2 | n | n | y | y | n |
| -2 | n | y | y | n | y |
| -2 | y | n | y | n | n |
| -2 | y | n | y | n | y |

consider it to be 'like'

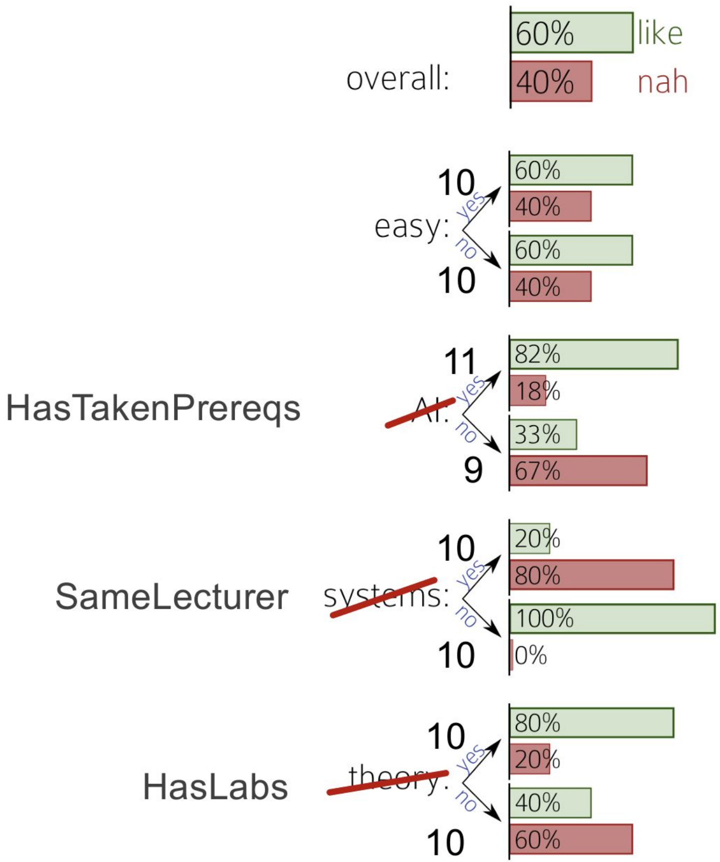
consider it to be 'dislike'



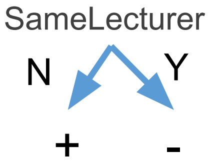
| | | |
|---|---|---|
| True label: | +, +, +, -, -, -, -, -, - | +, +, +, +, +, +, +, +, +, -, - |
| Predicted label: | -, -, -, -, -, -, -, -, - | +, +, +, +, +, +, +, +, +, +, + |

Accuracy: $$\frac{9}{20} \cdot \frac{6}{9} + \frac{11}{20} \cdot \frac{9}{11} = \frac{15}{20} = 0.75$$

# How to construct a tree



Suppose placing the node SameLecturer at the root.



What is the train set accuracy now?

$$\frac{10}{20}\cdot\frac{10}{10}+\frac{10}{20}\cdot\frac{8}{10}=\frac{18}{20}=0.9$$

even better!

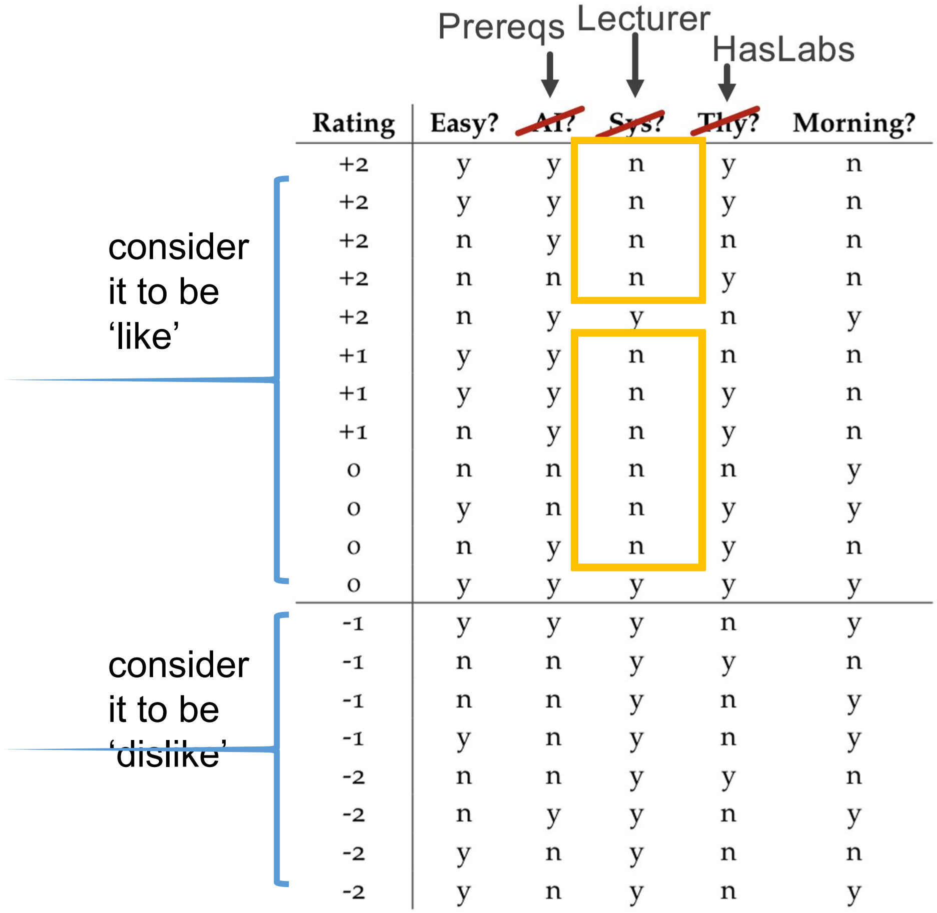
What would you do to build a depth-1 tree?

try out each feature and choose the one that leads to the largest accuracy!

# How to construct a tree

What is the train set accuracy now?

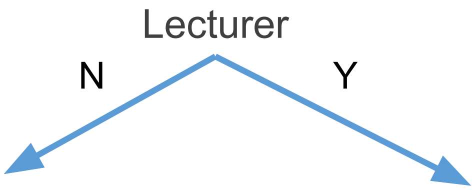
$$\boxed{\frac{10}{20}} \cdot \frac{10}{10} + \frac{10}{20} \cdot \frac{8}{10} = \frac{18}{20} = 0.9$$


Prereqs
Lecturer
HasLabs

consider it to be 'like'

consider it to be 'dislike'

| Rating | Easy? | ~~AI?~~ | ~~Sys?~~ | ~~Thy?~~ | Morning? |
|---|---|---|---|---|---|
| +2 | y | y | n | y | n |
| +2 | y | y | n | y | n |
| +2 | n | y | n | n | n |
| +2 | n | n | n | y | n |
| +2 | n | y | y | n | y |
| +1 | y | y | n | n | n |
| +1 | y | y | n | y | n |
| +1 | n | y | n | y | n |
| 0 | n | n | n | n | y |
| 0 | y | n | n | y | y |
| 0 | n | y | n | y | n |
| 0 | y | y | y | y | y |
| -1 | y | y | y | n | y |
| -1 | n | n | y | y | n |
| -1 | n | n | y | n | y |
| -1 | y | n | y | n | y |
| -2 | n | n | y | y | n |
| -2 | n | y | y | n | y |
| -2 | y | n | y | n | n |
| -2 | y | n | y | n | y |

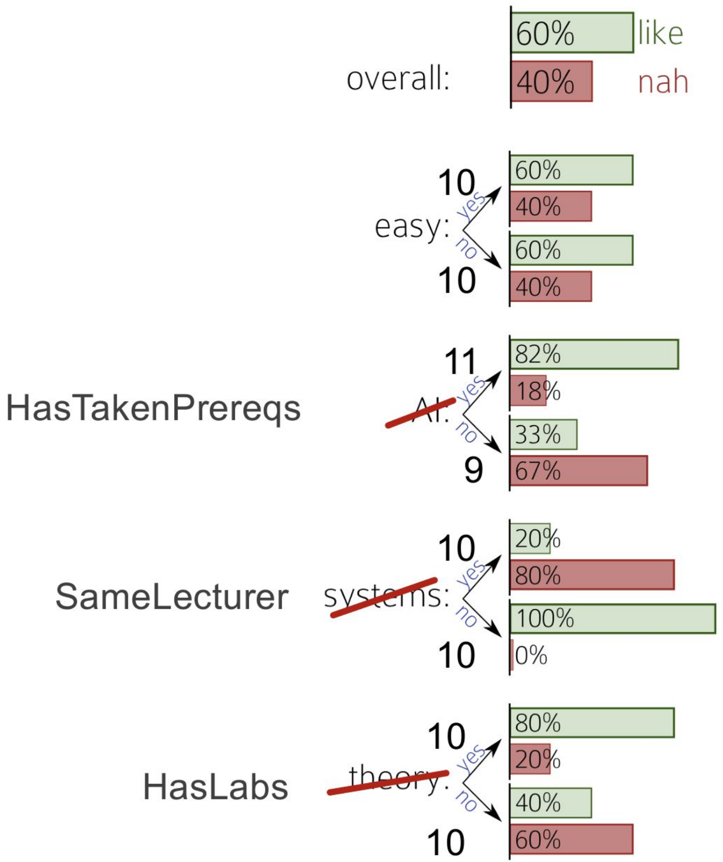


True label: +, +, +, +, +, +, +, +, +, + +, +, -, -, -, -, -, -, -, -

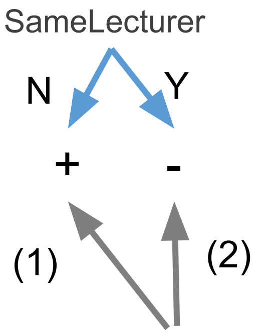
Predicted label: +, +, +, +, +, +, +, +, +, + -, -, -, -, -, -, -, -, -, -

Accuracy: $$\frac{10}{20} \cdot \frac{10}{10} + \frac{10}{20} \cdot \frac{8}{10} = \frac{18}{20} = 0.9$$

# How to construct a tree



What about depth 2?

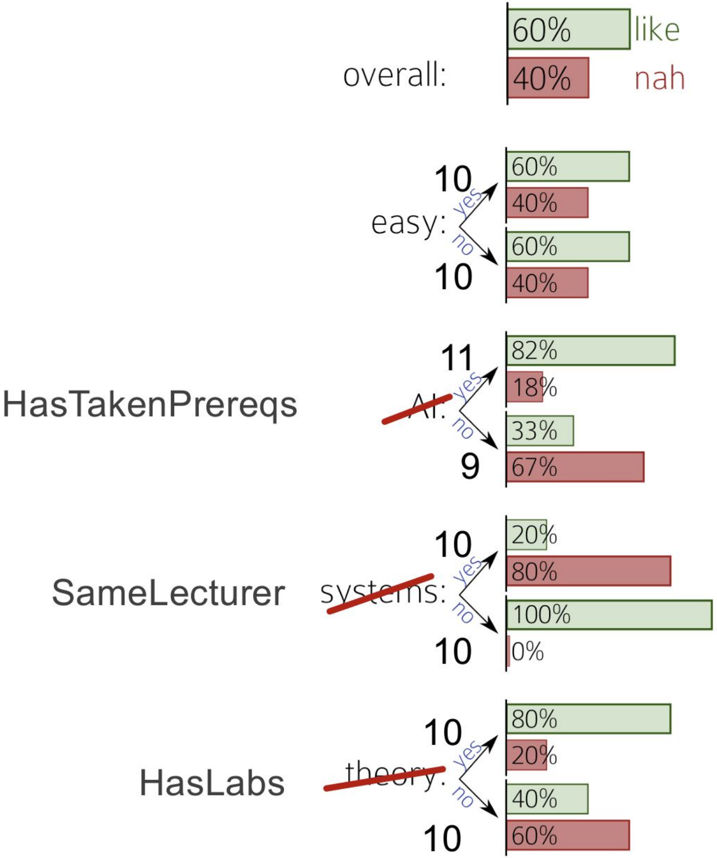


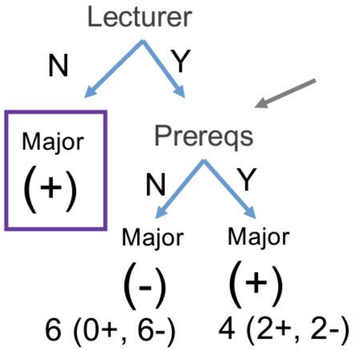
Which nodes to put at each leaf node?

Focus on (2). Try placing HasTakenPrereqs

All the data on (1) has same label “like”, no need to do further splitting.

# How to construct a tree





Q: How many training data points fall here? 10

Q: How many training data points arrive at these two leaves? How many for each label?

Q: what prediction should we use for each leaf?

Q: What is the train set accuracy, conditioning on SameLecturer=Y?

‘local’ train set accuracy

$$\frac{6}{10}\cdot\frac{6}{6}+\frac{4}{10}\cdot\frac{2}{4}=\frac{8}{10}$$

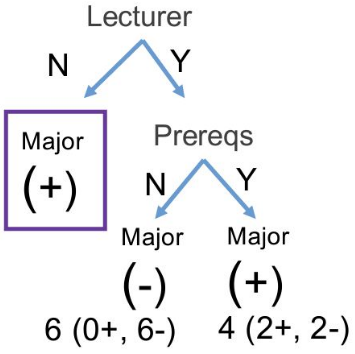
ry all the other nodes and pick the one with the largest acc.!

hen, repeat the same for SameLecturer=N branch!

> but this has 1 local train set acc. So leave it be!

love onto expanding nodes at depth 2!

# How to construct a tree

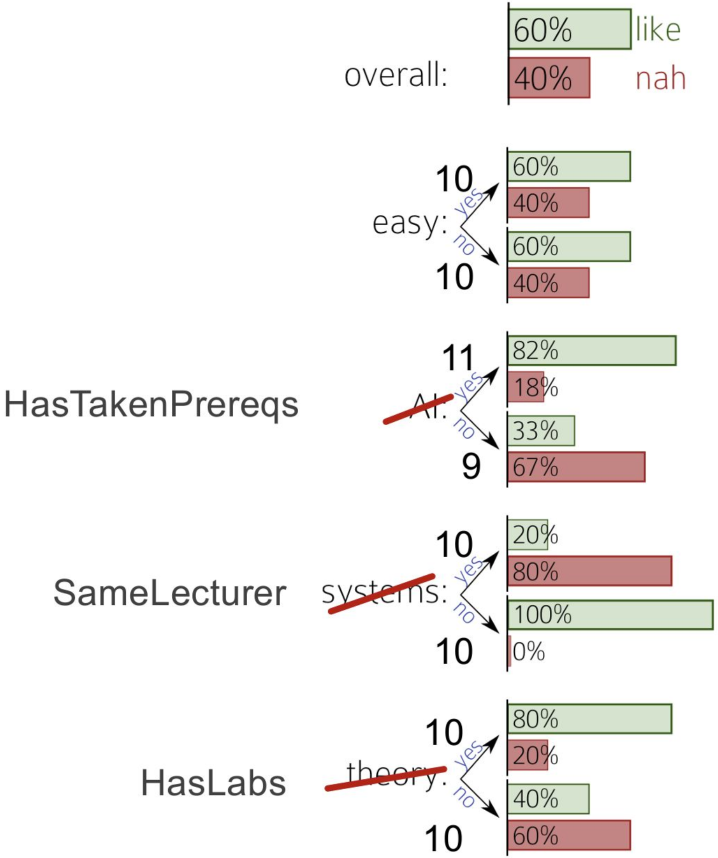


Q: What is the train set accuracy, conditioning on SameLecturer=Y?

$$\boxed{\frac{6}{10}}\cdot\frac{6}{6}+\boxed{\frac{4}{10}}\frac{2}{4}=\frac{8}{10}$$

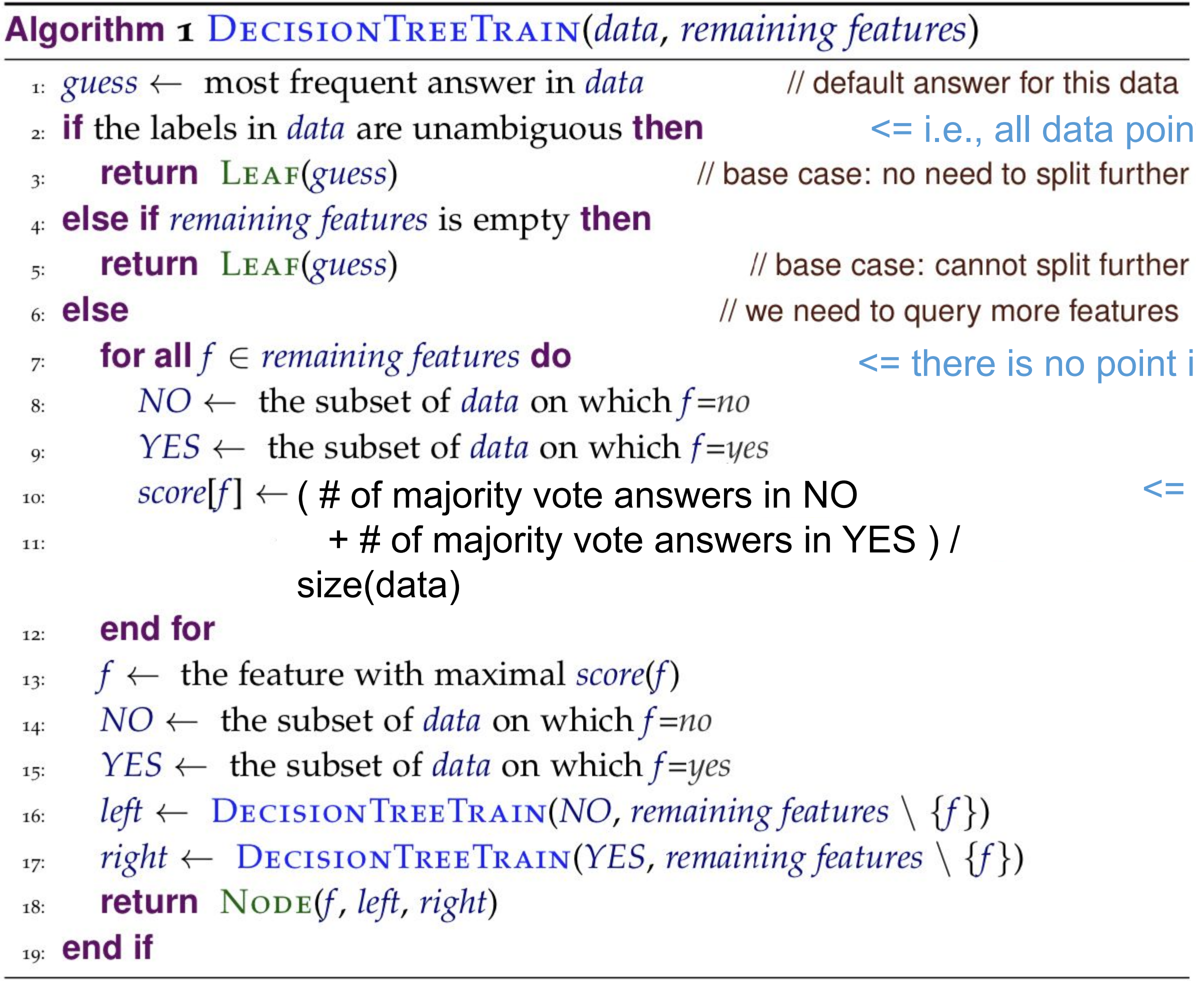
| | | Prereqs | Lecturer | HasLabs | |
|---|---|---|---|---|---|
| **Rating** | **Easy?** | ~~**AI?**~~ | ~~**Sys?**~~ | ~~**Thy?**~~ | **Morning?** |
| +2 | y | y | n | y | n |
| +2 | y | y | n | y | n |
| +2 | n | y | n | n | n |
| +2 | n | n | n | y | n |
| +2 | n | y | y | n | y |
| +1 | y | y | n | n | n |
| +1 | y | y | n | y | n |
| +1 | n | y | n | y | n |
| 0 | n | n | n | n | y |
| 0 | y | n | n | y | y |
| 0 | n | y | n | y | n |
| 0 | y | y | y | y | y |
| -1 | y | y | y | n | y |
| -1 | n | n | y | y | n |
| -1 | n | n | y | n | y |
| -1 | y | n | y | n | y |
| -2 | n | n | y | y | n |
| -2 | n | y | y | n | y |
| -2 | y | n | y | n | n |
| -2 | y | n | y | n | y |

# How to construct a tree



Overall idea:

1. Set the root node as a leaf node.
2. Grab a leaf node for which its ‘local’ train accuracy is not 1.
3. Find a feature that maximizes the ’local’ train accuracy and replace the leaf node with a node with that feature; add leaf nodes and set their predictions by majority vote.
4. Repeat 2-3.



**Algorithm 1** DecisionTreeTrain(*data*, *remaining features*)

```
guess ← most frequent answer in data                    // default answer for this data
if the labels in data are unambiguous then
    return Leaf(guess)                                  // base case: no need to split further
else if remaining features is empty then
    return Leaf(guess)                                  // base case: cannot split further
else                                                    // we need to query more features
    for all $f \in$ remaining features do
        NO ← the subset of data on which $f$=no
        YES ← the subset of data on which $f$=yes
        score[$f$] ← ( # of majority vote answers in NO
                          + # of majority vote answers in YES ) /
                       size(data)
    end for
    $f$ ← the feature with maximal score($f$)
    NO ← the subset of data on which $f$=no
    YES ← the subset of data on which $f$=yes
    left ← DecisionTreeTrain(NO, remaining features $\setminus \{f\}$)
    right ← DecisionTreeTrain(YES, remaining features $\setminus \{f\}$)
    return Node($f$, left, right)
end if
```

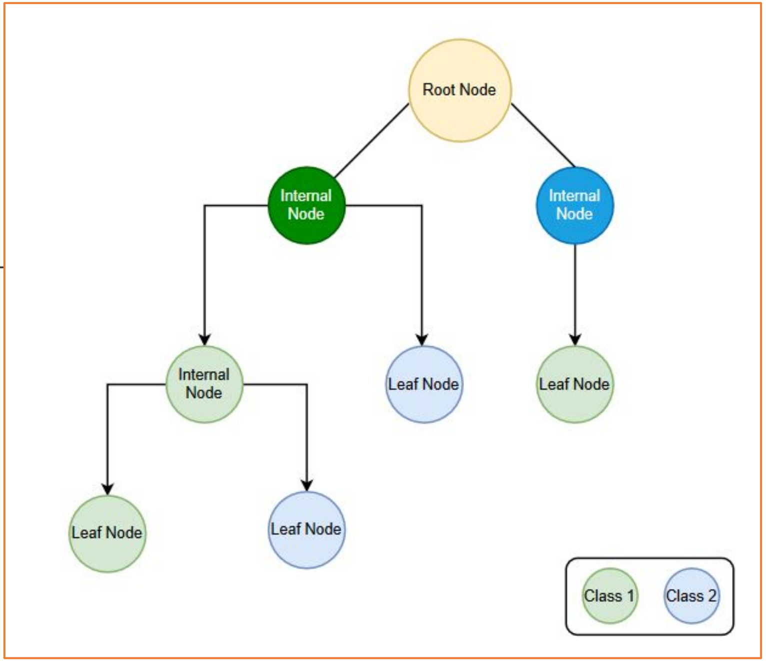
<= i.e., all data points have the same label

<= there is no point in adding a feature that appeared in its parent!

<= answer = label

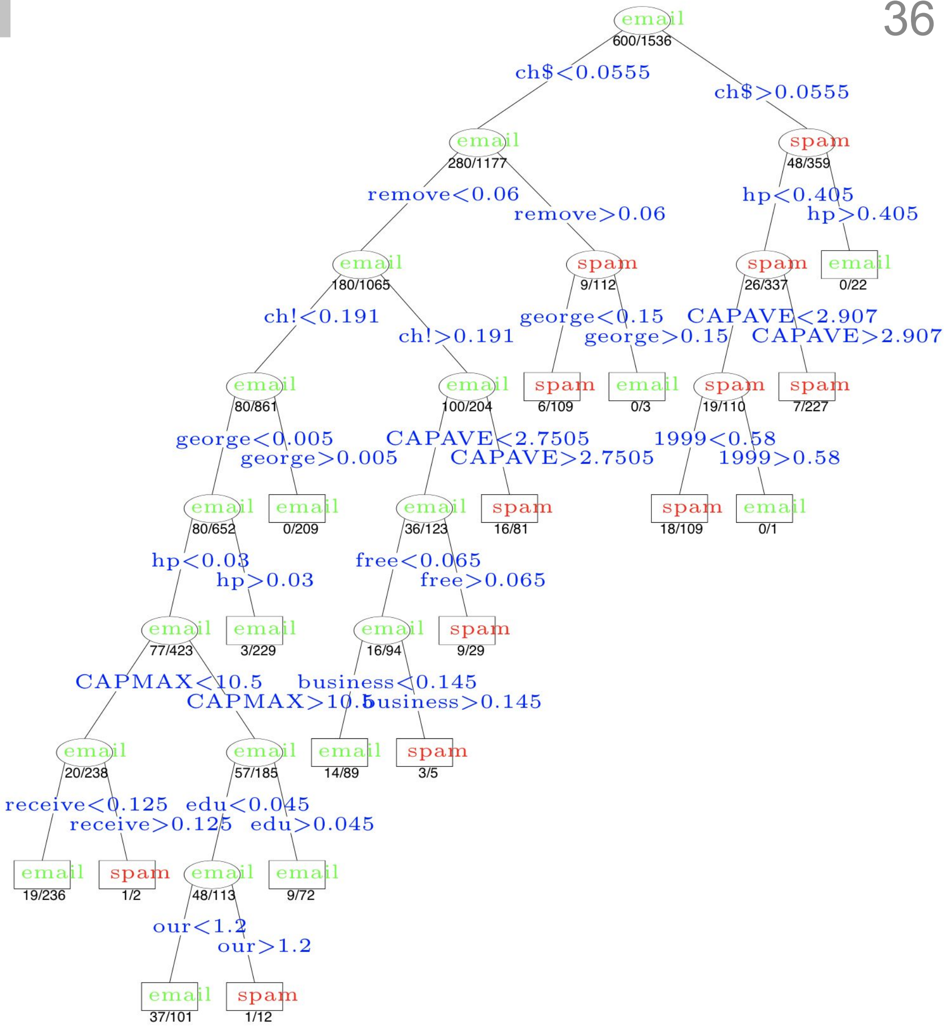
**Algorithm 2** DecisionTreeTest(*tree*, *test point*)

```
if tree is of the form Leaf(guess) then
    return guess
else if tree is of the form Node(f, left, right) then
    if $f = no$ in test point then
        return DecisionTreeTest(left, test point)
    else
        return DecisionTreeTest(right, test point)
    end if
end if
```



# Example: spam filtering I

- Spam dataset
- 4601 email messages, about 39% are spam
- Classify message by spam and not-spam
- 57 features
  - 48 are of the form “percentage of email words that is (WORD)”
  - 6 are of the form “percentage of email characters is (CHAR)”
  - 3 other features (e.g., “longest sequence of all-caps”)
- Final tree after pruning has 17 leaves, 9.3% test error rate


email
600/1536
ch$<0.0555
ch$>0.0555
email
280/1177
spam
48/359
remove<0.06
remove>0.06
hp<0.405
hp>0.405
email
180/1065
spam
9/112
spam
26/337
email
0/22
ch!<0.191
ch!>0.191
george<0.15
george>0.15
CAPAVE<2.907
CAPAVE>2.907
email
80/861
email
100/204
spam
6/109
email
0/3
spam
19/110
spam
7/227
george<0.005
george>0.005
CAPAVE<2.7505
CAPAVE>2.7505
1999<0.58
1999>0.58
email
80/652
email
0/209
email
36/123
spam
16/81
spam
18/109
email
0/1
hp<0.03
hp>0.03
free<0.065
free>0.065
email
77/423
email
3/229
email
16/94
spam
9/29
CAPMAX<10.5
CAPMAX>10.5
business<0.145
business>0.145
email
20/238
email
57/185
email
14/89
spam
3/5
receive<0.125
receive>0.125
edu<0.045
edu>0.045
email
19/236
spam
1/2
email
48/113
email
9/72
our<1.2
our>1.2
email
37/101
spam
1/12