Computer Science

CSC380: Principles of Data Science

Predictive Modeling and Classification 1

Xinchen Yu

Announcements

Midterm grade will be out by this Thursday.

Participation policy (5 points)

- Each office hour: + 1 point
- Answering question in the lecture: + 1point
- Answering question(related to course materials) on Piazza: + 1 point
- Asking question (related to course materials) on Piazza: +0.5 point

Office hours and participation in the lecture are available on Gradescope.

Piazza participation will be counted and added to the above before final exam.

Note: It is your responsibility to ensure the TA or instructor enter your participation points on gradescope during the office hour or after each lecture. Instructors will not award you these points at a later date, do not email instructors about getting points at a later date (for example, if you forget to ask the TA to enter your office hour points on gradescope).

- Probability
- Statistics

- Data Visualization

- Predictive modeling
- Linear models
- Nonlinear models
- Clustering

Introduction to Machine Learning

What is machine learning?

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

Machine Learning, Tom Mitchell, McGraw Hill, 1997.

“through experience”

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.

- A bit outdated with recent trends, but still has interesting discussion (and easy to read).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.

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.
- **Use**: sorting/searching images by category.
- **Other example**: categorize types of stars/events in the Universe (images taken from large surveying telescopes)

AI Task 2: Recommender systems

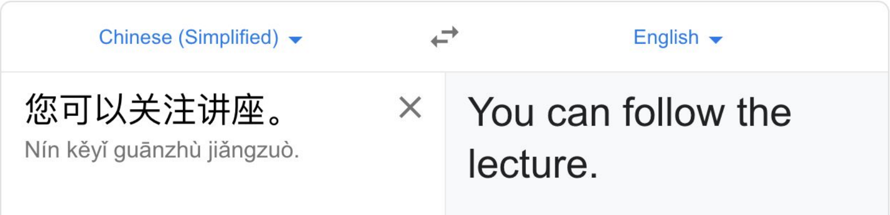
- Predict how user would rate a movie
- **Use**: For each user, pick an unwatched movie with the high predicted ratings.
- **Idea**: compute user-user similarity or movie-movie similarity, then compute a weighted average.

	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"

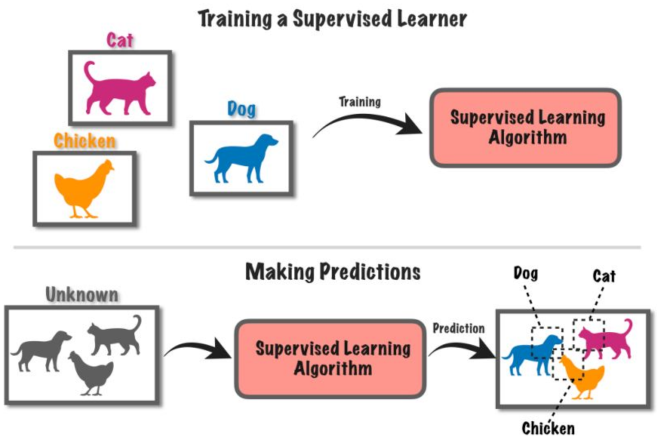
AI Task 3: Machine translation

- No need to explain how useful it is.

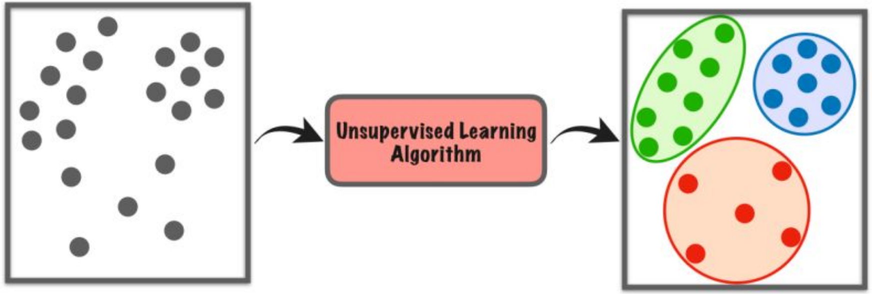


Supervised vs Unsupervised Learning

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



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

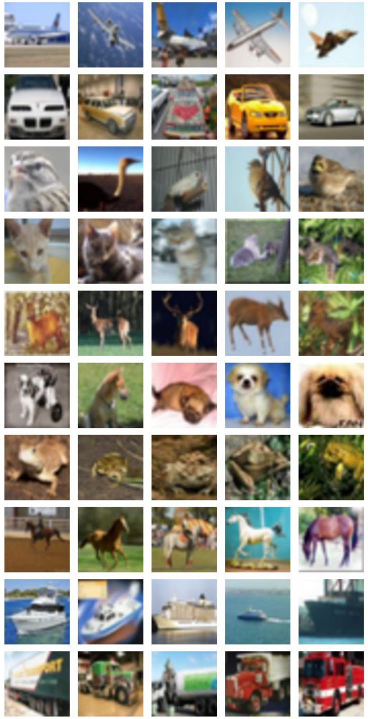


Supervised learning

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!

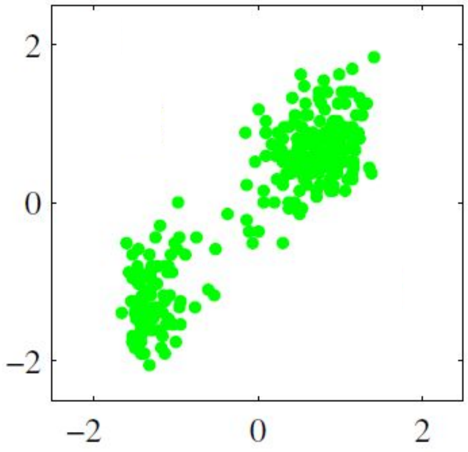
training

testing

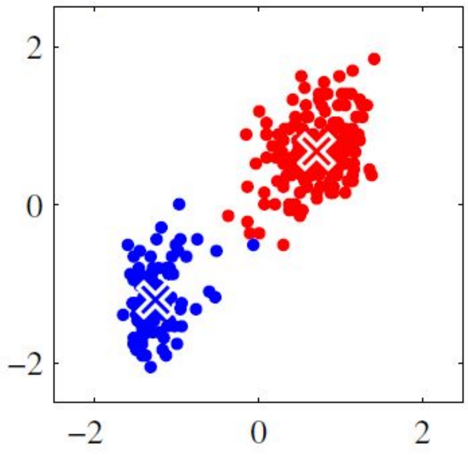
Unsupervised learning: Clustering

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..
$\Rightarrow$ happens when there is no association between features and labels in the dataset

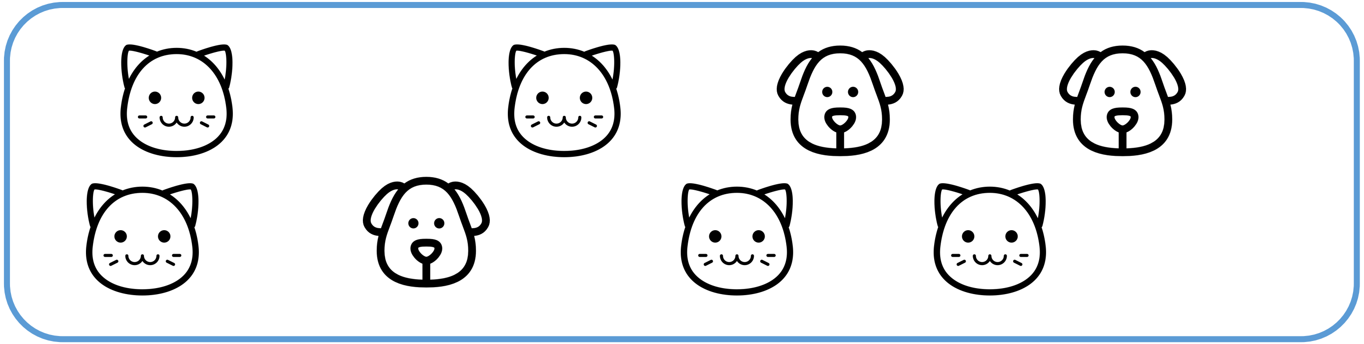
Train set accuracy

- 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:

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

Train set accuracy

If the model is majority vote classifier..

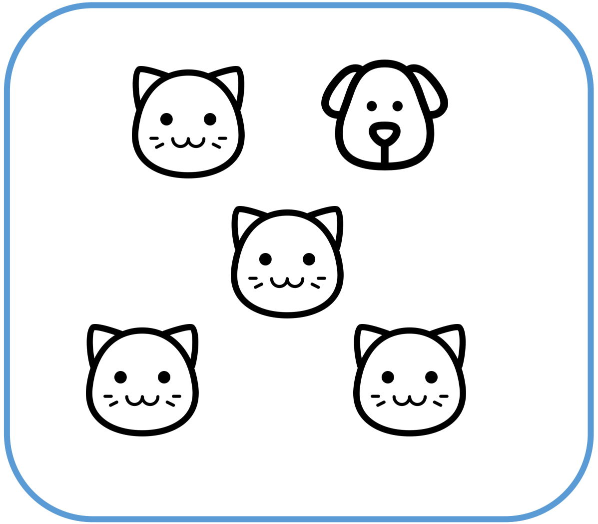


Majority vote: cat

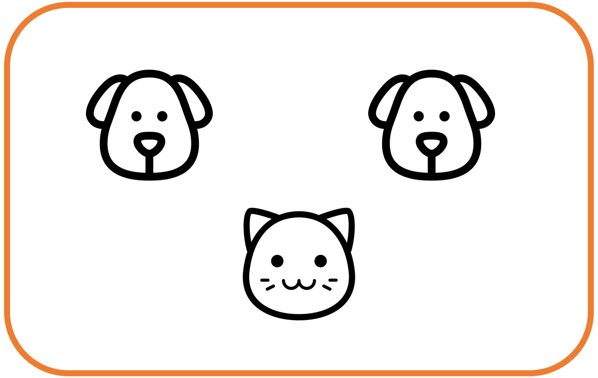
Q: what is the accuracy? $\frac{5}{8}$

Train set accuracy

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



Majority vote: cat



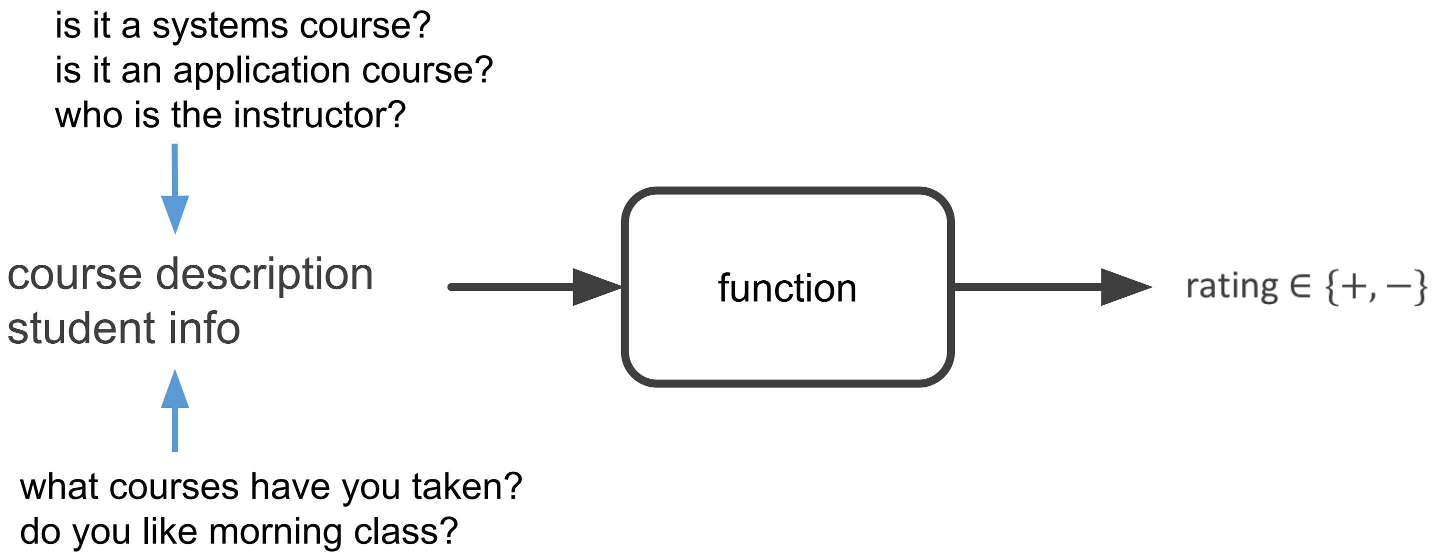
Majority vote: dog

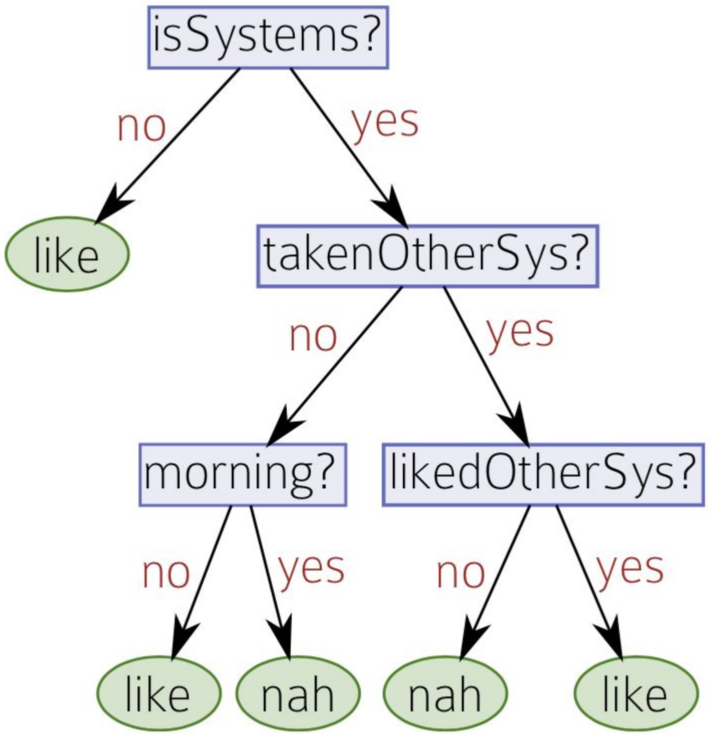
Q: what is the accuracy?

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

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





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

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

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

$\in \{y, n\}^d$ $\quad \in \{+, -\}$

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

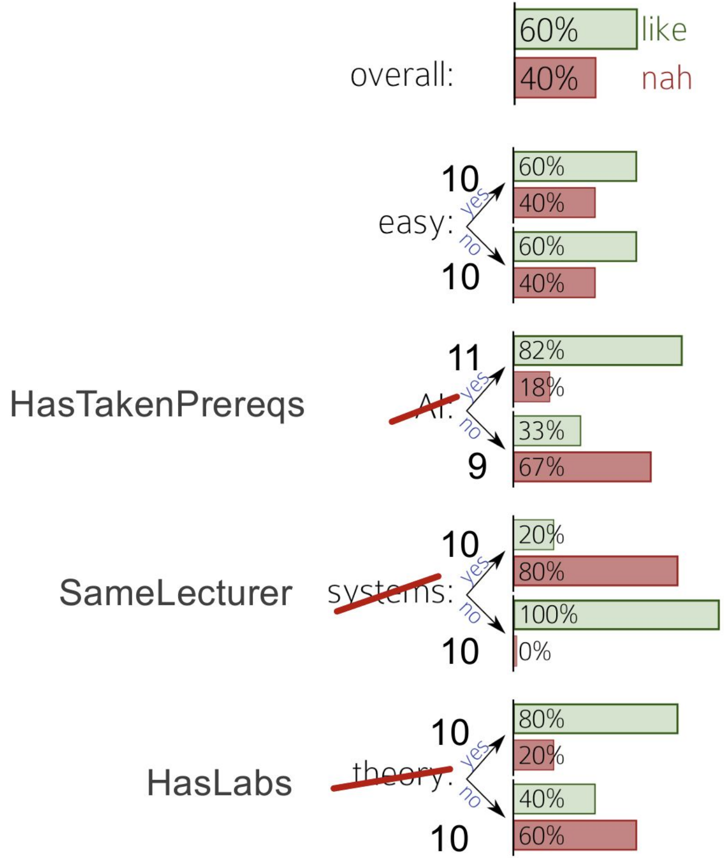
How to Train a Tree

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

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

- 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

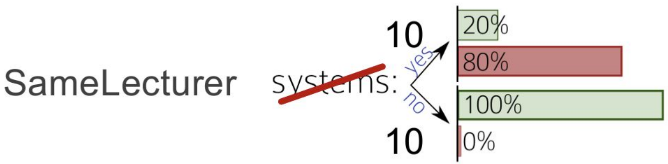


Prereqs Lecturer HasLabs

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

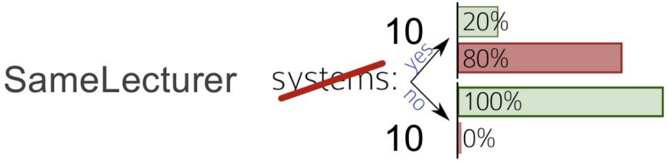
When SameLeturer = no:



Rating	Easy?	Prereqs ~~AI?~~	Lecturer ~~Sys?~~	HasLabs ~~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

When SameLeturer = yes:



Rating	Easy?	~~AI?~~ Prereqs	~~Sys?~~ Lecturer	~~Thy?~~ HasLabs	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: majority vote

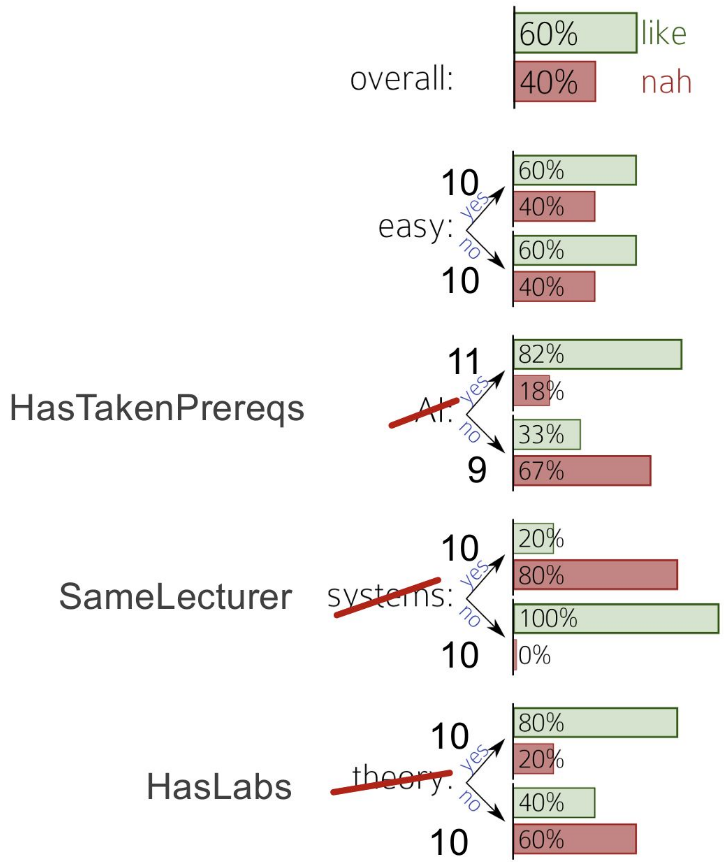
Baseline: majority vote classifier

Q: What is the train set accuracy?

12/20 = 0.60

Rating
+2
+2
+2
+2
+2
+1
+1
+1
0
0
0
0
-1
-1
-1
-1
-2
-2
-2
-2

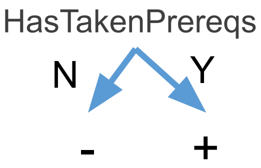
How to construct a tree: choosing root



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).

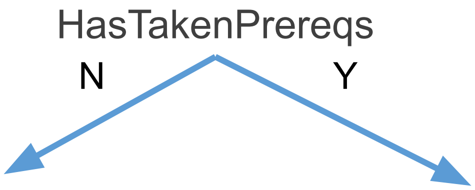
Predicted label for 11 people is **like**, 9 people are correctly predicted.

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'

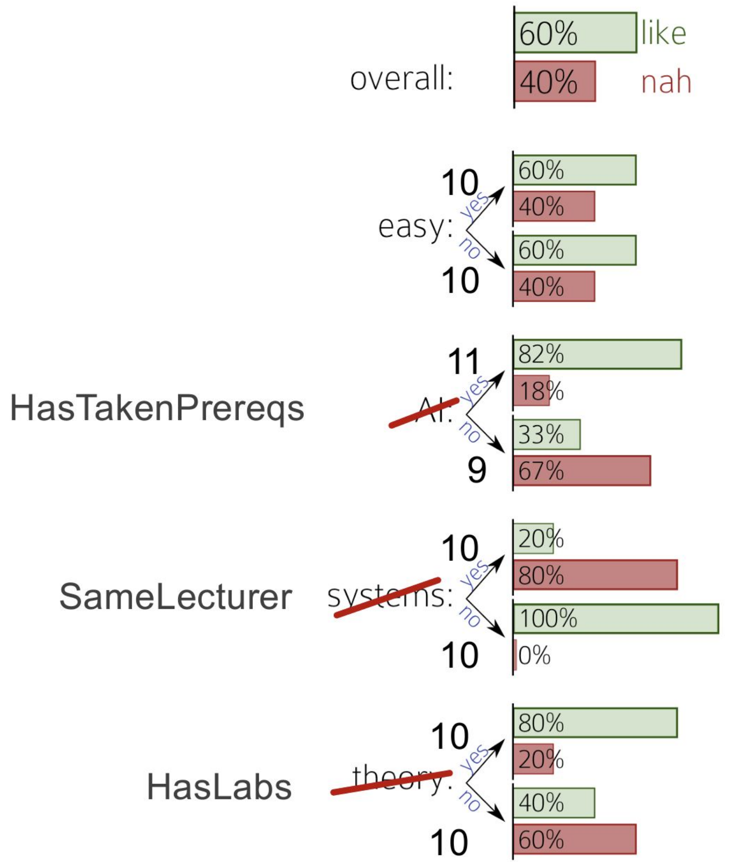
consider it to be 'dislike'



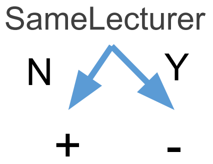
	N	Y
True label:	+, +, +, -, -, -, -, -, -	+, +, +, +, +, +, +, +, +, -, -
Predicted label:	-, -, -, -, -, -, -, -, -	+, +, +, +, +, +, +, +, +, +, +

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?

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

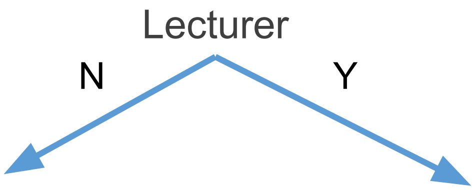
consider it to be 'like'

consider it to be 'dislike'

Prereqs → AI?, Lecturer → Sys?, HasLabs → Thy?

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

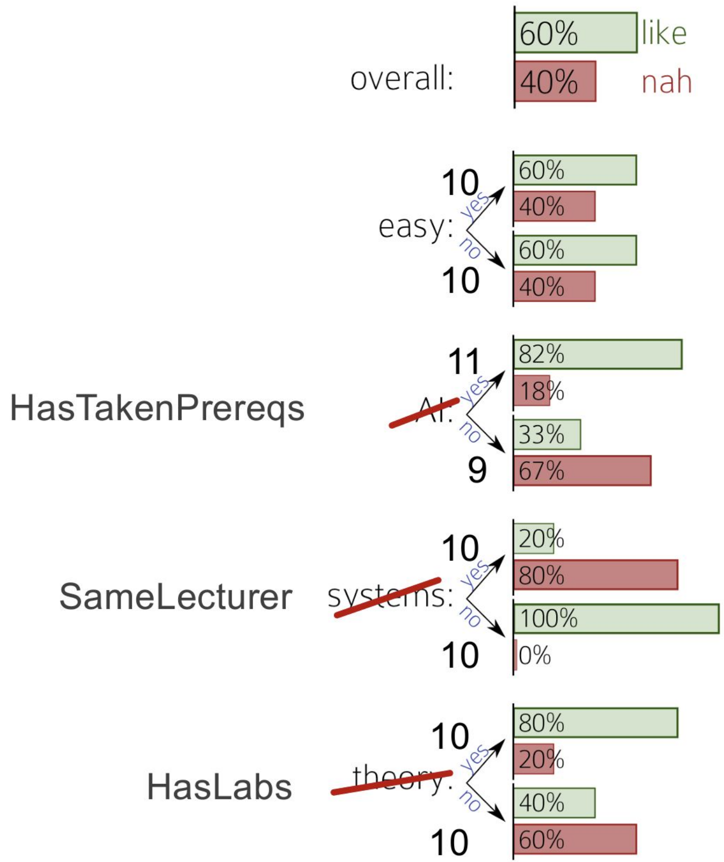


True label: +, +, +, +, +, +, +, +, +, + | +, +, -, -, -, -, -, -, -, -

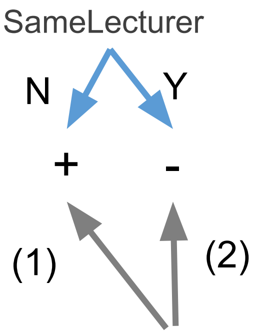
Predicted label: +, +, +, +, +, +, +, +, +, + | -, -, -, -, -, -, -, -, -, -

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?

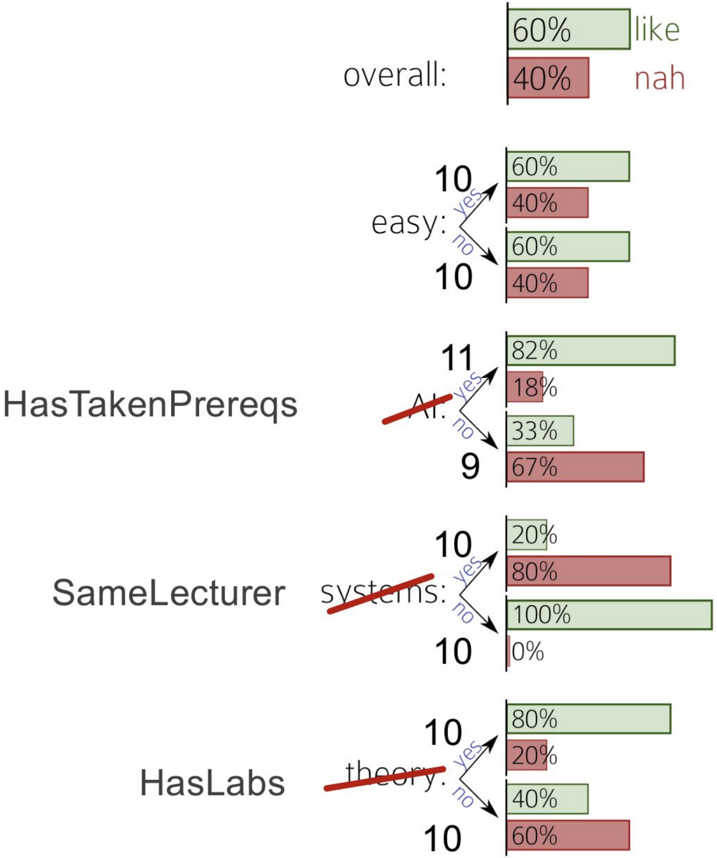


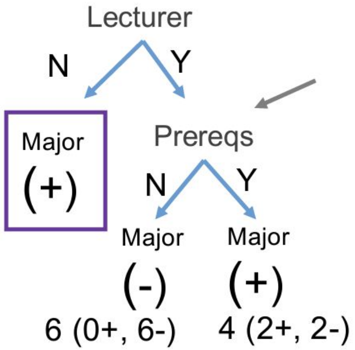
Which nodes to put at each leaf node?

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

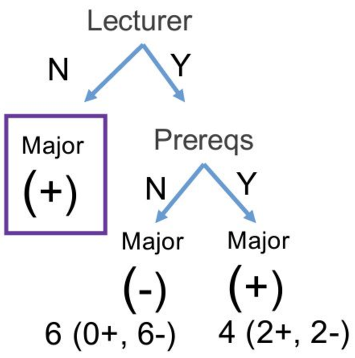
$$\frac{6}{10}\cdot\frac{6}{6}+\frac{4}{10}\cdot\frac{2}{4}=\frac{8}{10}$$

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

Then, repeat the same for SameLecturer=N branch!

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

How to construct a tree



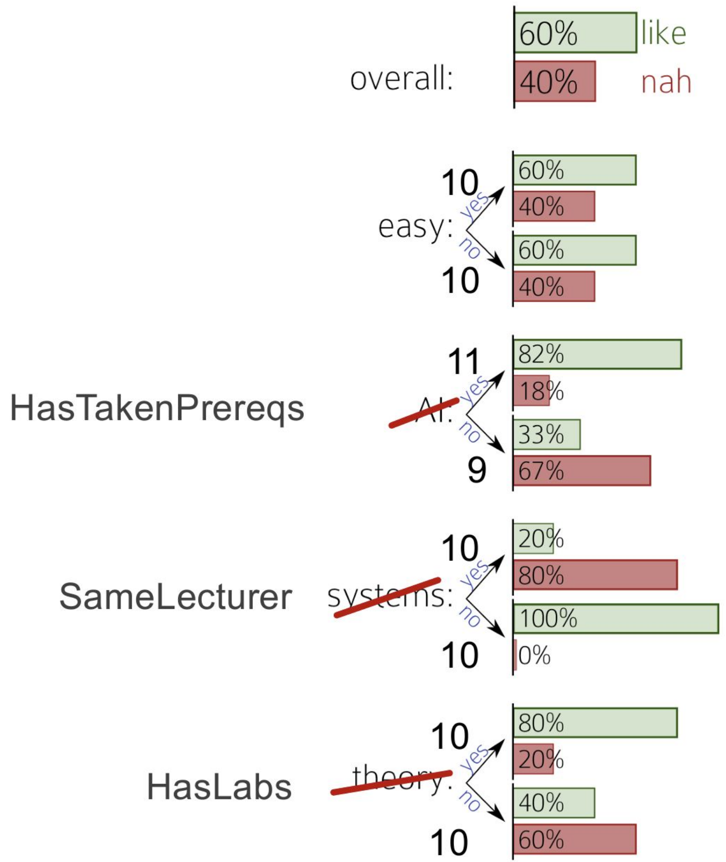
Q: What is the train set accuracy, conditioning on SameLecturer=Y?

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

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

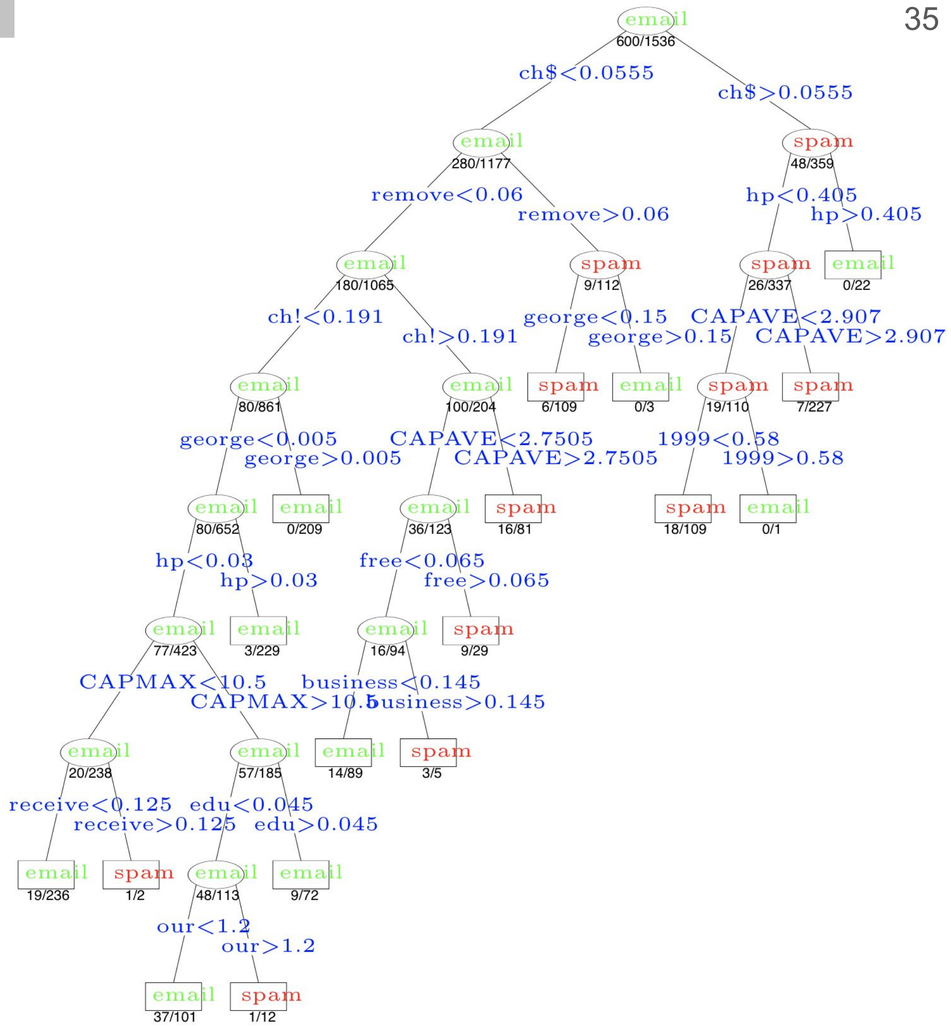


Overall idea:

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. If local accuracy is 1, no need to split the leaf node.
5. Repeat 2-3.

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



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                <= i.e., all data points have the same label
    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 ∈ remaining features do                     <= there is no point in adding a feature that
                                                             appeared in its parent!
        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     <= answer = label
                      + # 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 \ {f})
    right ← DecisionTreeTrain(YES, remaining features \ {f})
    return Node(f, left, right)
end if
```

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
```