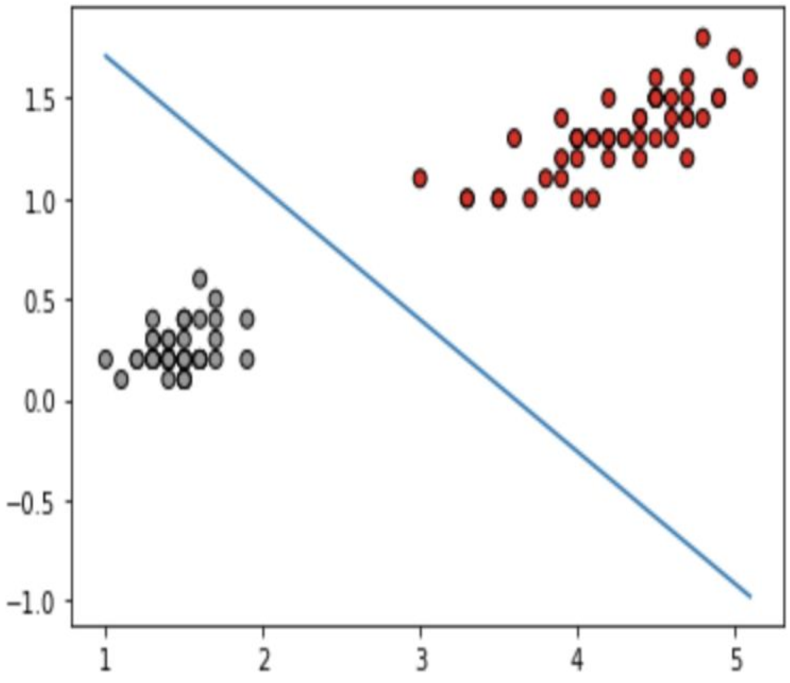
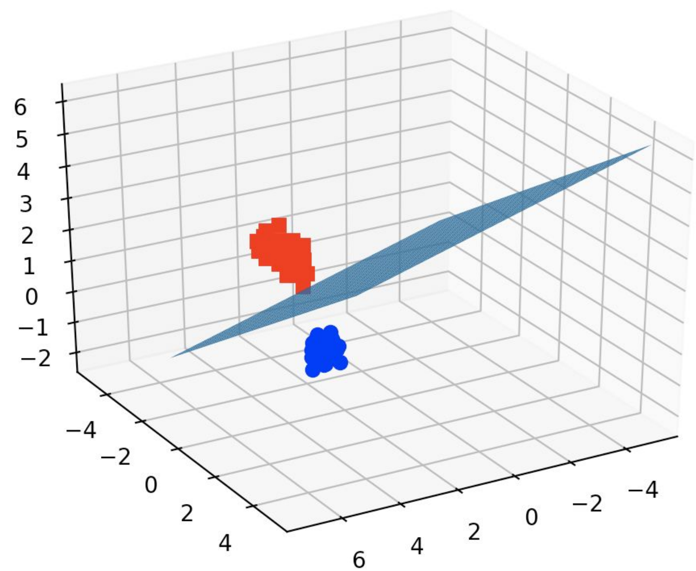
Last time

- Overfitting
- Max-margin classifiers

No laptops, screens during the class.

Visualizing decision boundaries in higher dimensions

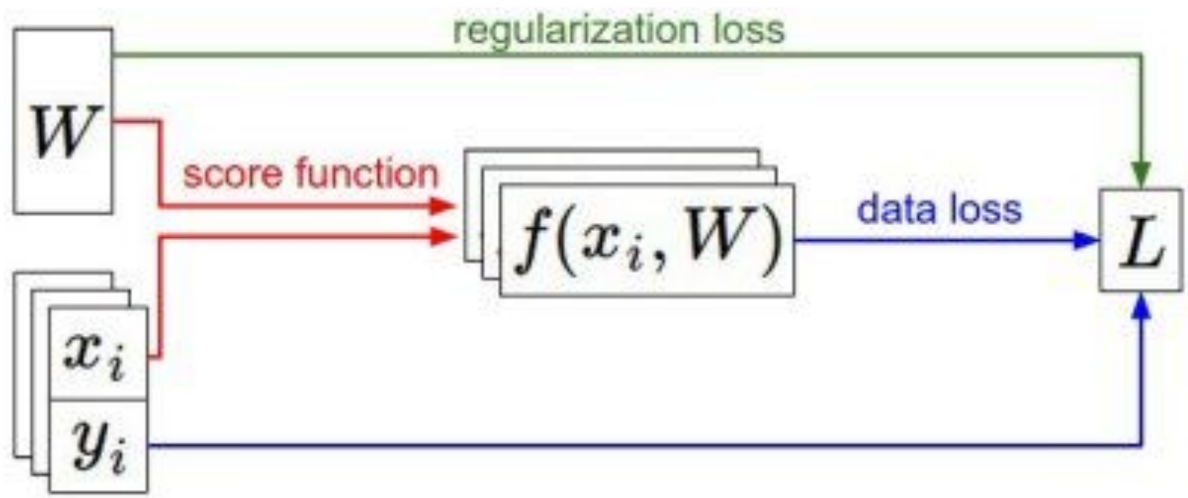




2D data points

3D data points

Recap: Regularizers



$$L(W) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)} + \underbrace{\lambda R(W)}$$

λ is a hyperparameter giving regularization strength

Data loss: Model predictions should match training data

Regularization: Prevent the model from doing *too* well on training data

Recap: Typical loss functions

$$L(W) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)}_{\text{Data loss}} + \underbrace{\lambda R(W)}_{\text{Regularization}}$$

λ is a hyperparameter giving regularization strength

Data loss: Model predictions should match training data

Regularization: Prevent the model from doing *too* well on training data

- Max margin SVM Loss (aka hinge loss).
- MSE Loss
- Count of mispredictions

Recap: Typical regularizers

$$L(W) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)} + \underbrace{\lambda R(W)}$$

λ is a hyperparameter giving regularization strength

Data loss: Model predictions should match training data

Regularization: Prevent the model from doing *too* well on training data

Simple examples

L2 regularization: $R(W) = \sum_{k,l} W_{k.l}^2$

L1 regularization: $R(W) = \sum_{k,l} |W_{k,l}|$

More complex:

Dropout

Batch normalization

Cutout, Mixup, Stochastic depth, etc…

What value should the regularizer weight take?

$$L(W) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)}_{\text{Data loss}} + \underbrace{\lambda R(W)}_{\text{Regularization}}$$

λ is a hyperparameter giving regularization strength

Data loss: Model predictions should match training data

Regularization: Prevent the model from doing *too* well on training data

- **Goal:** Minimize L(W)
- Data loss term is always positive.
  - MSE, Hinge Loss.
- R(W) : L1 norm or L2 **norms** = always positive.

<u>L2 regularization</u>: $R(W) = \sum_{k,l} W_{k.l}^2$

L1 regularization: $R(W) = \sum_{k,l} |W_{k,l}|$

What value should the regularizer weight take?

Goal: Minimize L(W)

$$L(W) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)}_{0} + \underbrace{\lambda R(W)}_{10}$$

λ is a hyperparameter giving regularization strength

Data loss: Model predictions should match training data

Regularization: Prevent the model from doing *too* well on training data

- If λ = +ve, say 0.1
  - L(W) = 1 -> Find W to minimize L(W)
- If λ = -ve, say -0.1
  - L(W) = -1 -> Find W to minimize L(W)

λ: should always be positive

Today

- **Gradient descent**
- Decision tree introduction
- Information Gain
- Bagging and Random Forests

Optimization

$$w^* = \arg\min_{w} L(w)$$

Intuitive visualization of the optimization landscape

Slide credit: Justin Johnson

Intuitive visualization of the optimization landscape

Slide credit: Justin Johnson

Intuitive visualization of the optimization landscape

We use iterative algorithms to find our way to the bottom of the landscape

Slide credit: Justin Johnson

What is the simplest thing we can do?

- Randomly walk through the landscape
- Try to find the lowest point of the landscape.
- This is called random search
  - Very bad idea - will not yield optimal results.

Idea:2: Follow the slope

Gradient Descent

- Start somewhere.
- Compute slope
- Take a step towards steepest direction
- Repeat

Gradient Descent

- **Start somewhere = weight initialization**
- Compute slope
- Take a step towards steepest direction
- Repeat

Gradient Descent

- **Start somewhere = weight initialization**
  - Typically drawn from a Gaussian distribution
  - Convergence depends on where you start.

Gradient Descent

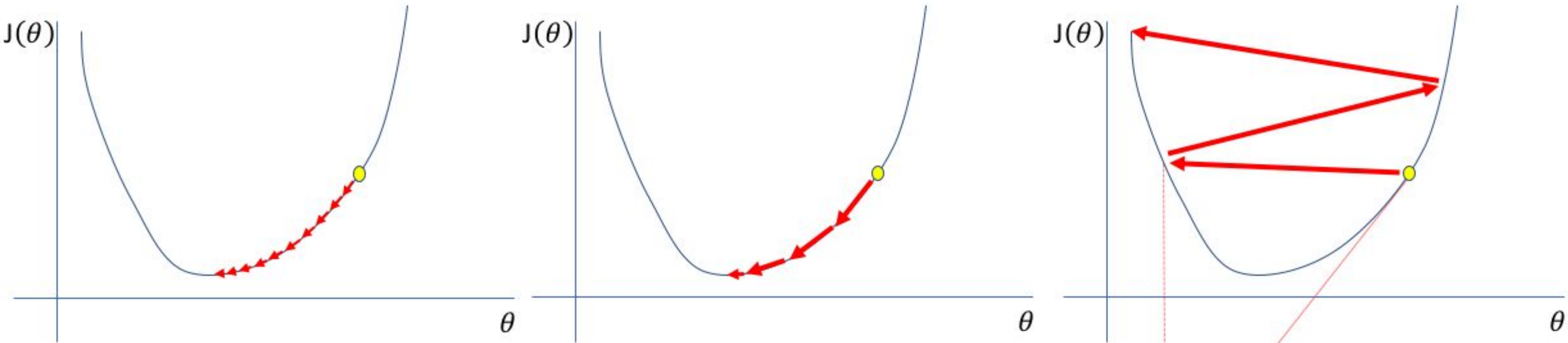
- Start somewhere == weight initialization
- **Compute slope == compute gradient of the cost function wrt parameters.**
- Take a step towards steepest direction
- Repeat

Gradient Descent

- Start somewhere == weight initialization
- Compute slope == compute gradient of the cost function wrt parameters.
- **Take a step towards steepest direction == learning rate.**
- Repeat

Choose LR wisely

Too low | **Just right** | **Too high**



A small learning rate requires many updates before reaching the minimum point

The optimal learning rate swiftly reaches the minimum point

Too large of a learning rate causes drastic updates which lead to divergent behaviors

Gradient Descent

- Start somewhere.
- Compute slope = compute gradient of the cost function wrt parameters.
- Take a step towards steepest direction
- **Repeat, for certain steps, stopping criteria.**

Meet your new best friends (or enemies)

- Weight initialization techniques.
- Learning rate (LR).
- Regularization parameters
- Number of steps.

Mini-Batch Gradient Descent

- If training data is very large:
  - Randomly sample a **batch** of data
  - Compute gradient only on that batch
  - **Underlying assumption:** The data distribution of the batch is representative of the entire training data.

Gradient descent - options

Batch gradient descent: Use **all $\boldsymbol{m}$ examples** in each iteration

Stochastic gradient descent: Use **1 example** in each iteration

Mini-batch gradient descent: Use **$\boldsymbol{b}$ examples** in each iteration

What are some potential downsides of mini-batch gradient descent? Select all that apply

What are some potential downsides of batch gradient descent? Select all that apply

Quiz question 56 answers 56 participants

A random sample may not be representative of the full training data - 40 answers ✓ 71%

More efficient to compute - 6 answers 11%

Noisy estimates could lead to slower convergence - 34 answers ✓ 61%

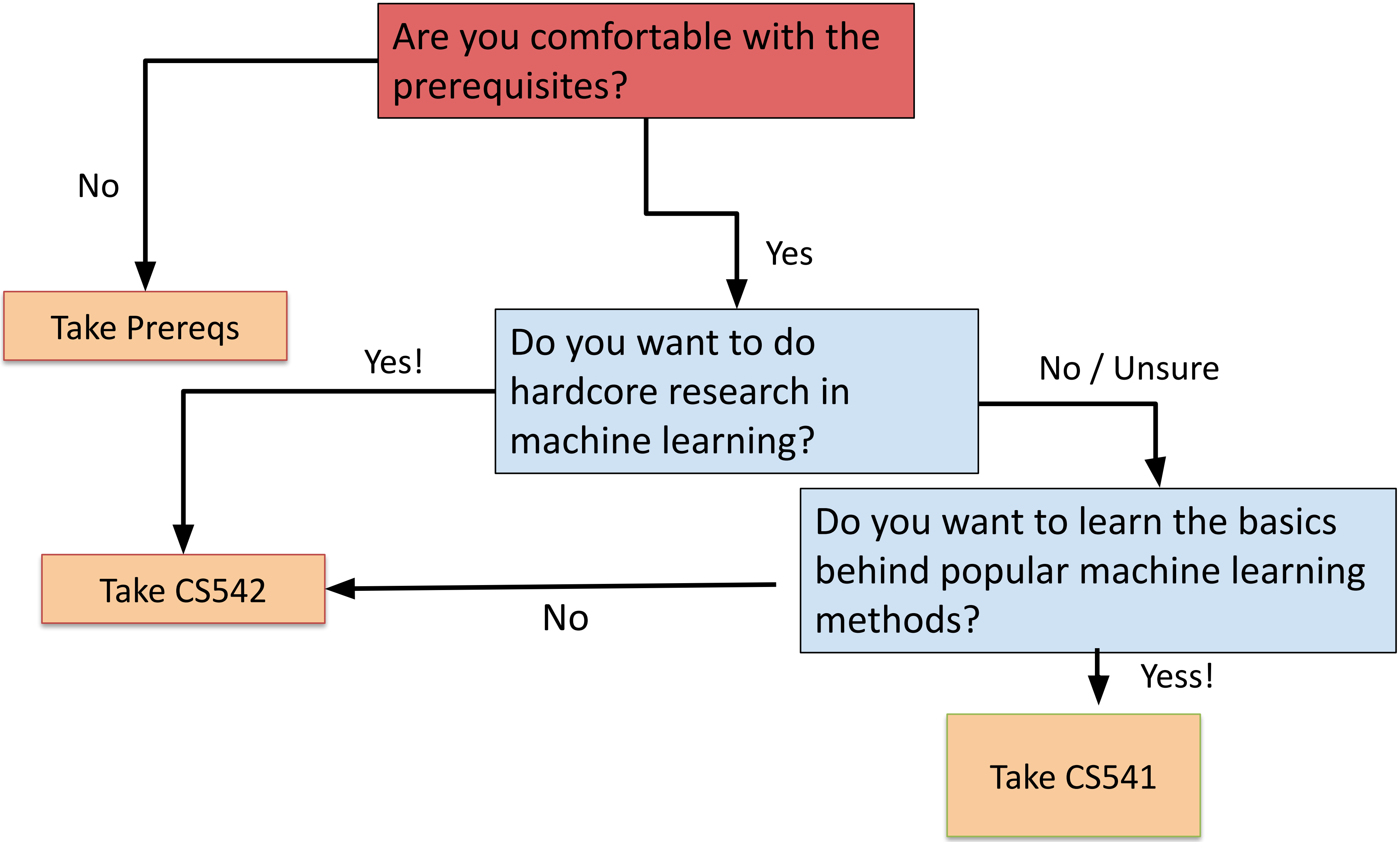
Sensitive to batch size - 50 answers ✓ 89%

slido

Today

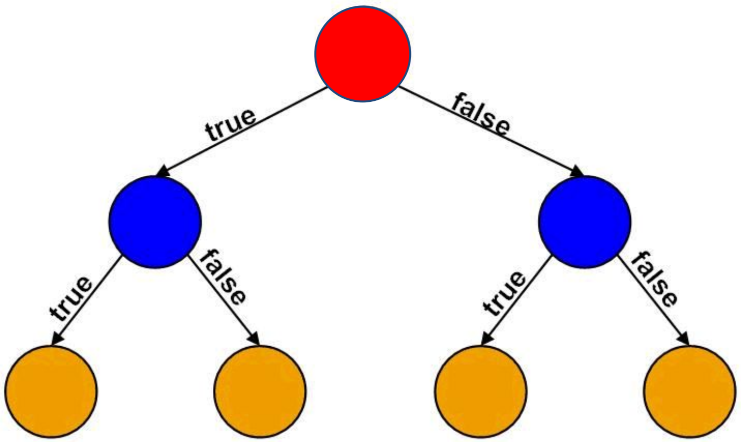
- Gradient descent
- **Decision tree introduction**
- Information Gain
- Bagging and Random Forests

From Lecture#1: Is this class for you?



A decision tree consists of

- Nodes: Tests for variables
  - Root node
  - Internal nodes
- Branches
  - Results of tests
- Leaves
  - No arrows out of them.
  - Final output.



Example

Should We Play Tennis?

Play Tennis	Outlook	Temperature	Humidity	Windy
No	Sunny	Hot	High	No
No	Sunny	Hot	High	Yes
Yes	Overcast	Hot	High	No
Yes	Rainy	Mild	High	No
Yes	Rainy	Cold	Normal	No

Example

Should We Play Tennis?

Play Tennis	Outlook	Temperature	Humidity	Windy
No	Sunny	Hot	High	No
No	Sunny	Hot	High	Yes
Yes	Overcast	Hot	High	No
Yes	Rainy	Mild	High	No
Yes	Rainy	Cold	Normal	No

- If temperature is not hot
  - Play

Example

Should We Play Tennis?

Play Tennis	Outlook	Temperature	Humidity	Windy
No	Sunny	Hot	High	No
No	Sunny	Hot	High	Yes
Yes	Overcast	Hot	High	No
Yes	Rainy	Mild	High	No
Yes	Rainy	Cold	Normal	No

- If temperature is not hot:
  - Play
- Else:
  - If outlook is overcast
    - Play
  - Otherwise
    - Don't play

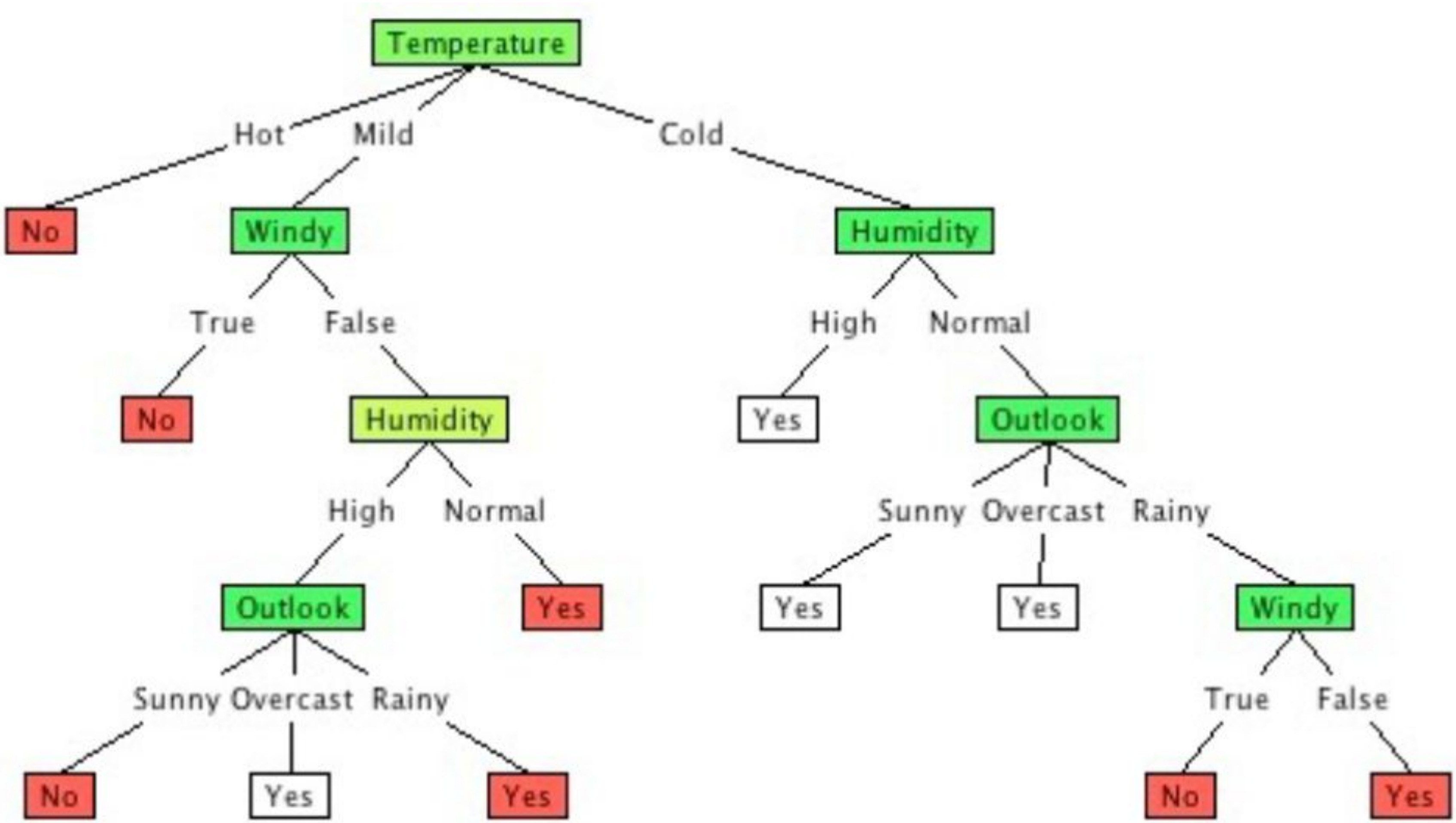
Example

Should We Play Tennis?

Play Tennis	Outlook	Temperature	Humidity	Windy
No	Sunny	Hot	High	No
No	Sunny	Hot	High	Yes
Yes	Overcast	Hot	High	No
Yes	Rainy	Mild	High	No
Yes	Rainy	Cold	Normal	No

- If temperature is not hot:
  - Play
- Else:
  - If outlook is overcast
    - Play
  - Otherwise
    - Don't play

Example Decision Tree

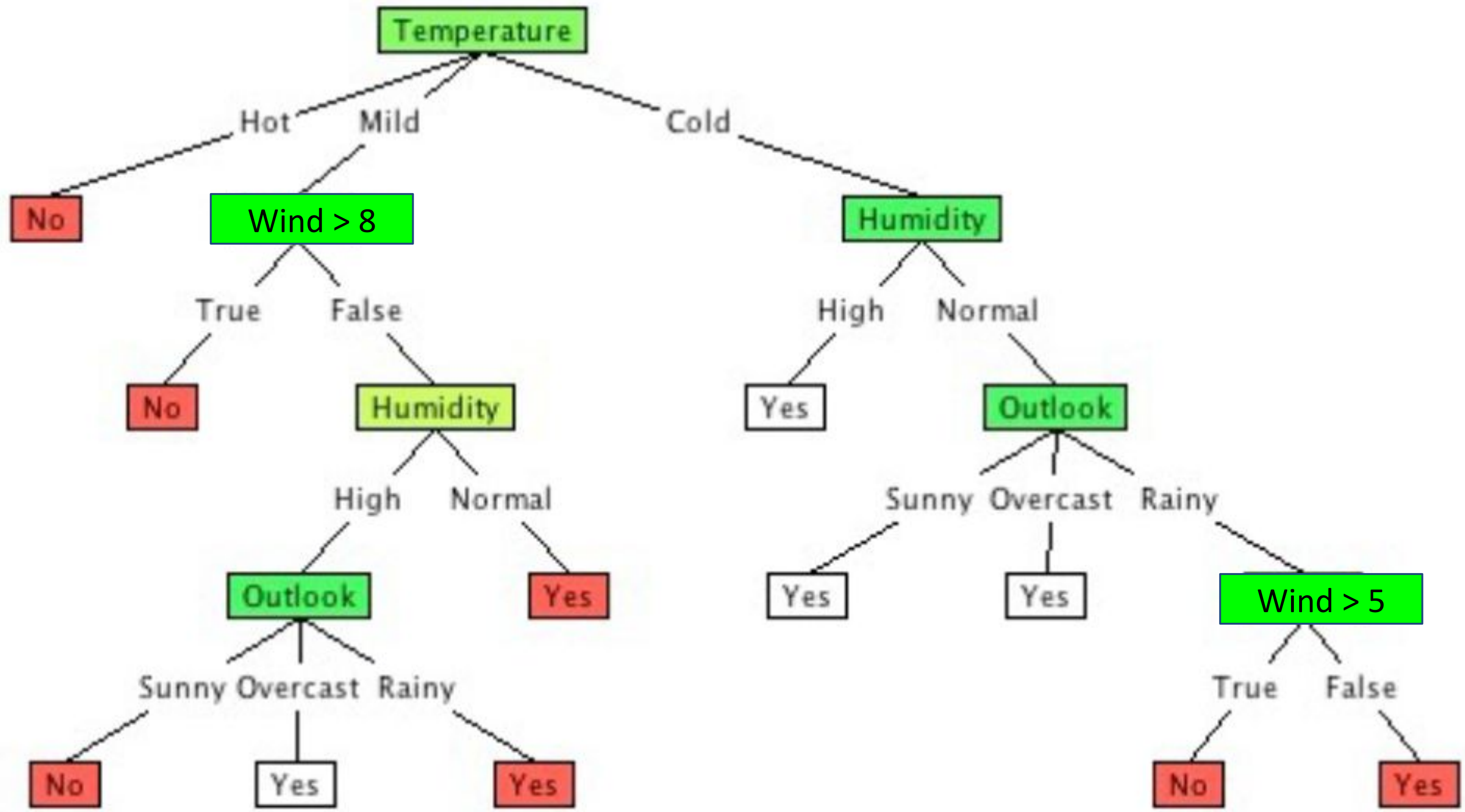


Advantages of Decision Trees

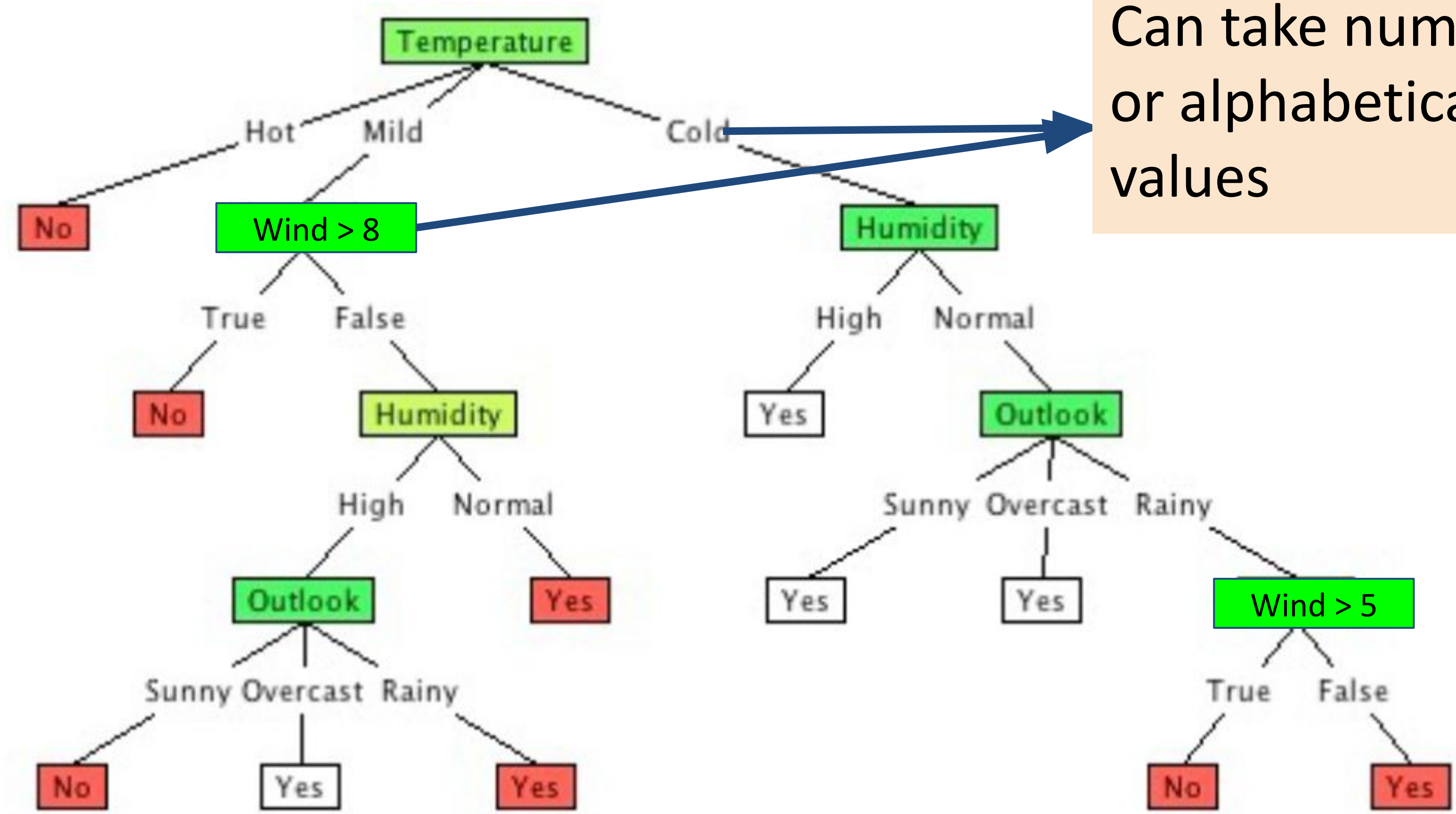
Should We Play Tennis?

Play Tennis	Outlook	Temperature	Humidity	Windy (mph)
No	Sunny	Hot	High	2
No	Sunny	Hot	High	15
Yes	Overcast	Hot	High	3
Yes	Rainy	Mild	High	5
Yes	Rainy	Cold	Normal	7

Advantages of Decision Trees



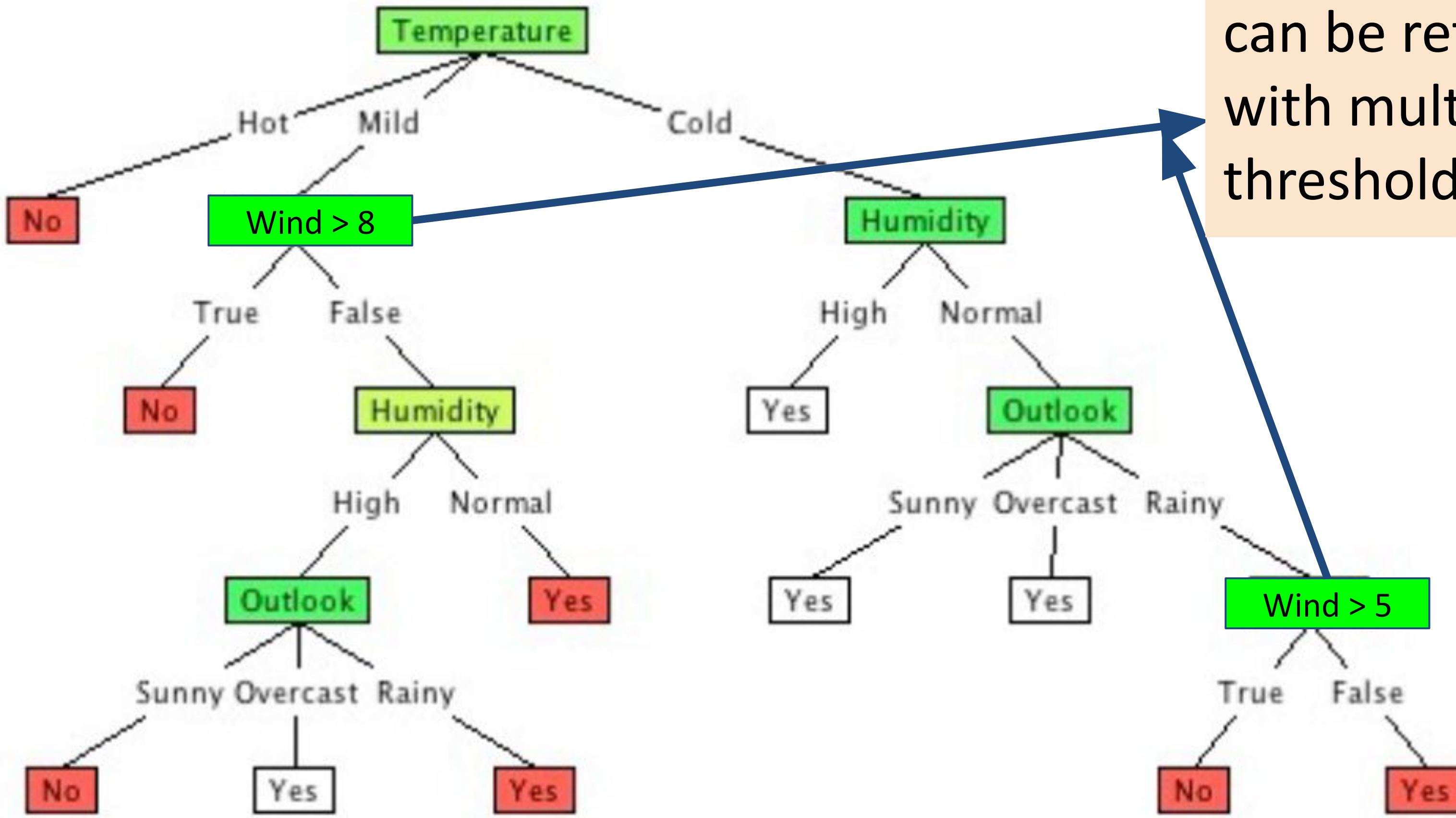
Advantages of Decision Trees



Can take numerical or alphabetical values

This gives more flexibility while curating dataset

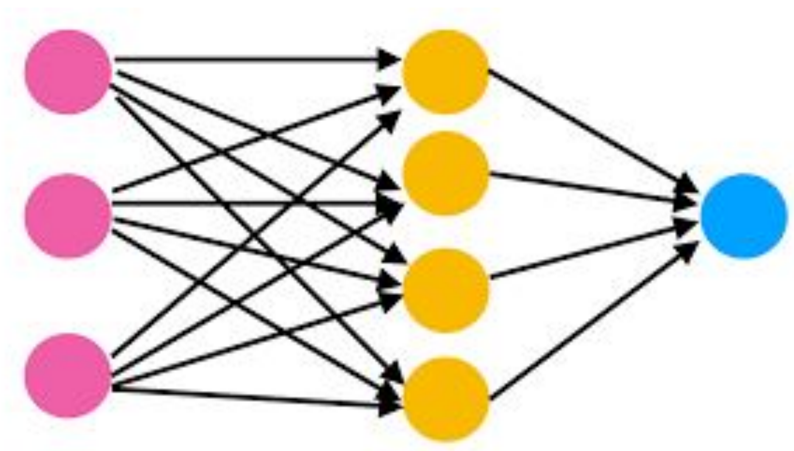
Advantages of Decision Trees



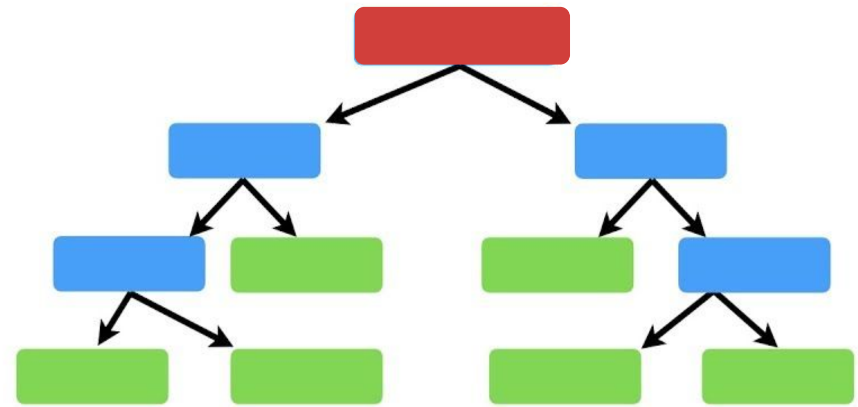
Same category can be referred with multiple thresholds

Advantages of Decision Trees

Ease of interpretability



vs



- Decision trees clearly show the path the model took to reach the final outcome.
- Not possible with neural networks and even with non-linear SVMs

Decision trees can be used for

Decision trees can be used for

Quiz question 66 answers 66 participants

only linearly separable data distributions - 7 answers

11%

only data which has binary outcomes - 11 answers

17%

only non-linearly separable data distributions - 1 answer

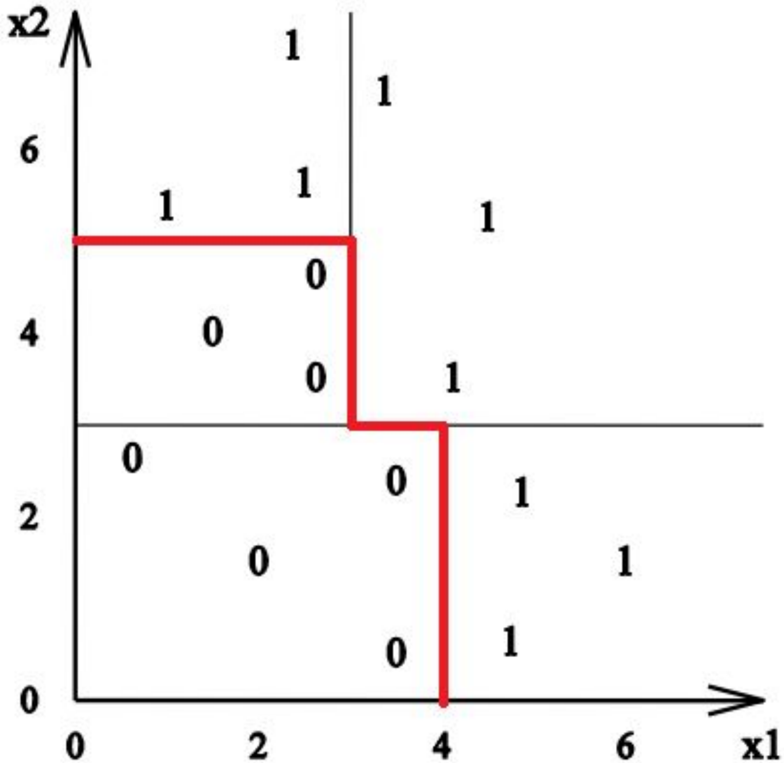
2%

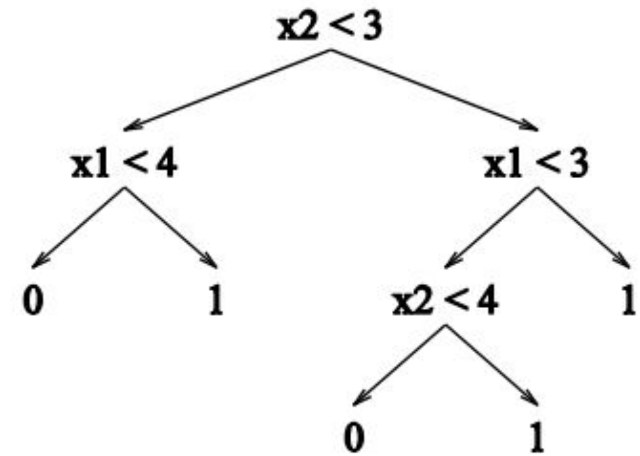
Any kind of data distributions - 47 answers

71%

slido

Decision Trees can tackle non-linear data





Applications of Decision Trees

- Widely adopted in applications where interpretability is very important.
- Eg:
  - Healthcare diagnostics
  - financial risk assessment

What makes a good tree?

- Smaller, well-balanced tree is better.

What makes a good tree?

- Smaller, well-balanced tree is better.
  - Avoids overfitting.
- How do we build small trees that accurately capture data?

Today

- Gradient descent
- Decision tree introduction
- **Quantifying Entropy**
- Bagging and Random Forests

Entropy

- Entropy in the context of decision tree:
  - A measure of node impurity.
  - Leafs = least entropy.

- Goal: as we traverse down the tree, goal is to reduce the entropy.

How to choose which variable to split on?

Entropy: $$E(S)=\sum_{i=1}^{c}-p_i \log_2 p_i$$

Dataset with 2 classes, yes or no

$$E(S) = -(P_{yes}\log_2 P_{yes} + P_{no}\log_2 P_{no})$$

Of the 10 data points, 6 marked as yes, 4 as no

$$\mathrm{E(S)} = -\ (6/10 * \log_2 * 6/10 + 4/10 * \log_2 * 4/10) \quad \approx 0.971$$

Of the 10 data points, 10 marked as yes, 0 as no

$$\mathrm{E(S)} = -\ (1 \log_2 1) = 0$$

Less entropy

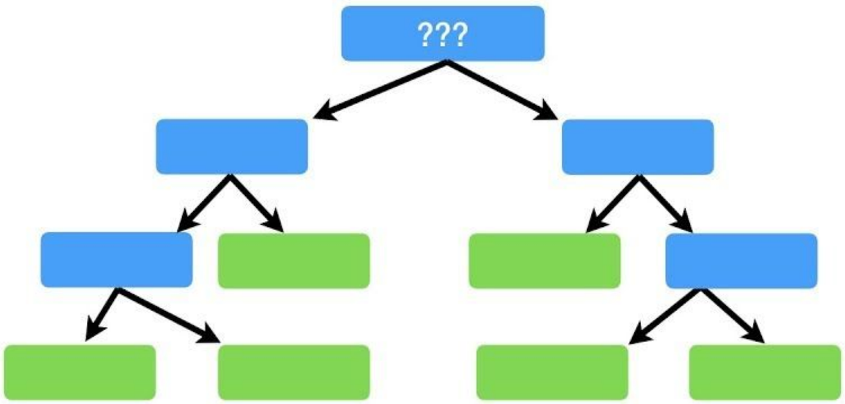
Of the 10 data points, 5 marked as yes, 5 as no

$$\mathrm{E(S)} = -2(0.5 \log_2 0.5) = 1$$

High entropy

How to choose which variable to split on?

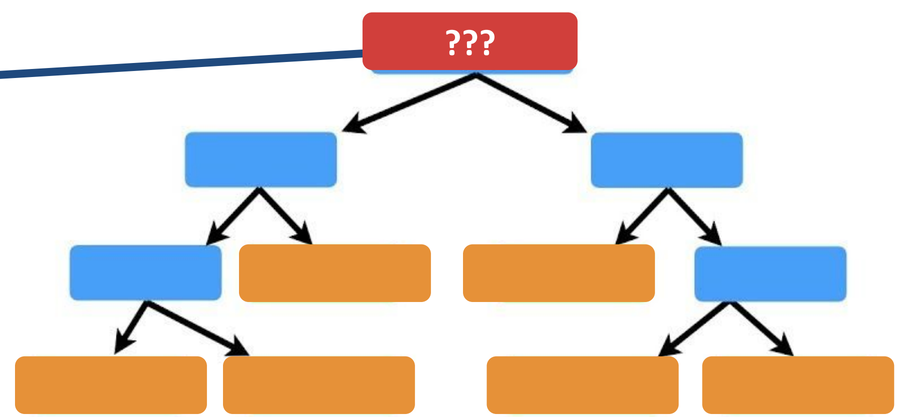
Play Tennis	Outlook	Temperature	Humidity	Windy
No	Sunny	Hot	High	No
No	Sunny	Hot	High	Yes
Yes	Overcast	Hot	High	No
Yes	Rainy	Mild	High	No
Yes	Rainy	Cold	Normal	No



How to choose which variable to split on?

Play Tennis	Outlook	Temperature	Humidity	Windy
No	Sunny	Hot	High	No
No	Sunny	Hot	High	Yes
Yes	Overcast	Hot	High	No
Yes	Rainy	Mild	High	No
Yes	Rainy	Cold	Normal	No

Temp == hot



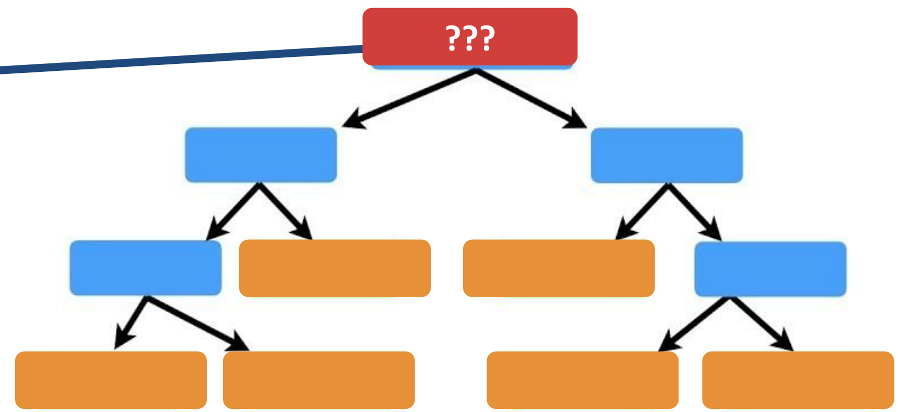
How to choose which variable to split on?

Play Tennis	Outlook	Temperature	Humidity	Windy
No	Sunny	Hot	High	No
No	Sunny	Hot	High	Yes
Yes	Overcast	Hot	High	No
Yes	Rainy	Mild	High	No
Yes	Rainy	Cold	Normal	No

Temp == hot

Yes → Play Tennis

No → Play Tennis



How to choose which variable to split on?

Play Tennis	Outlook	Temperature	Humidity	Windy
No	Sunny	Hot	High	No
No	Sunny	Hot	High	Yes
Yes	Overcast	Hot	High	No
Yes	Rainy	Mild	High	No
Yes	Rainy	Cold	Normal	No

Temp == hot

Yes

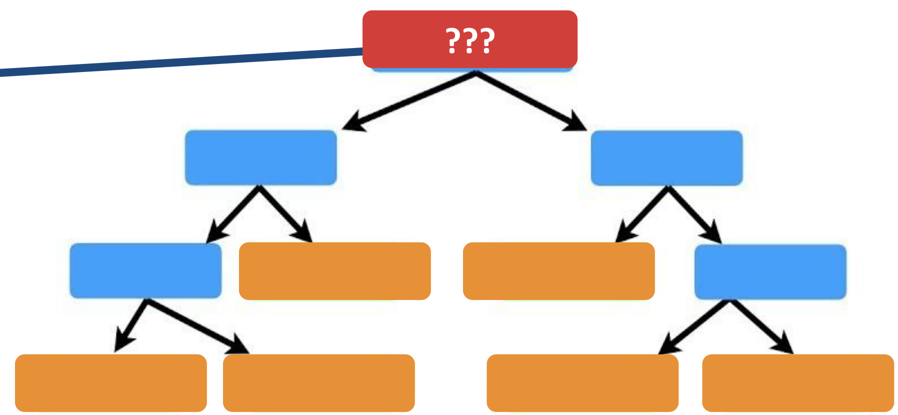
No

Play Tennis

Play Tennis

Yes = 1
No = 2

Yes = 2
No = 0



How to choose which variable to split on?

Play Tennis	Outlook	Temperature	Humidity	Windy
No	Sunny	Hot	High	No
No	Sunny	Hot	High	Yes
Yes	Overcast	Hot	High	No
Yes	Rainy	Mild	High	No
Yes	Rainy	Cold	Normal	No

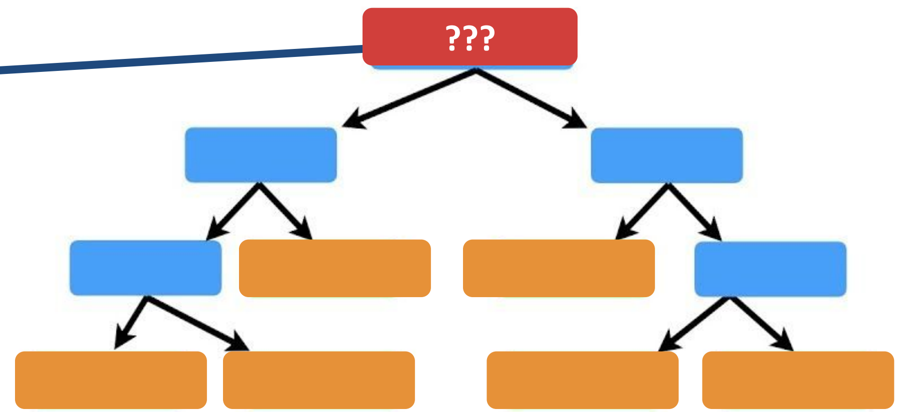
Temp == hot

Yes

No

Play Tennis

Play Tennis



Yes = 1
No = 2

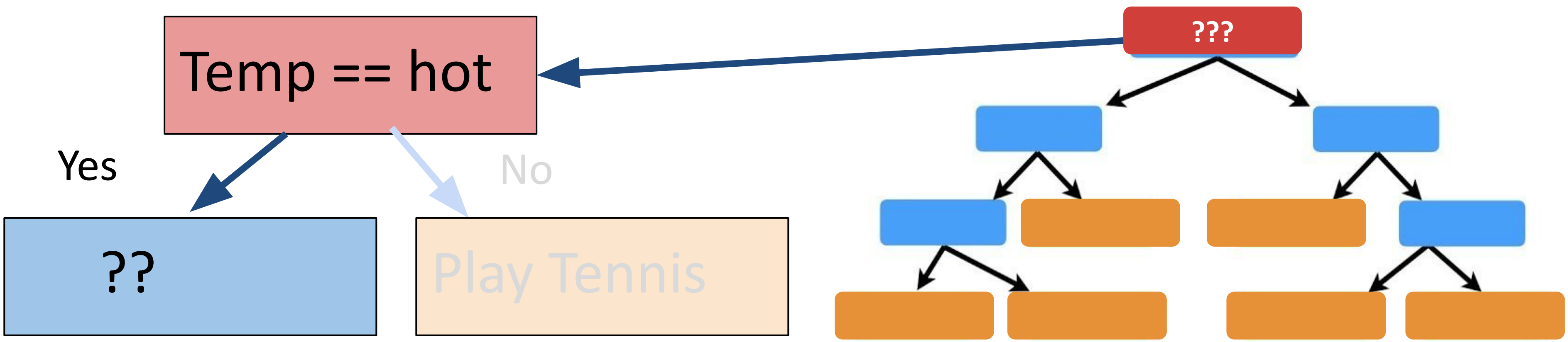
❌ “impure”

Yes = 2
No = 0

✅ “pure”

How to choose which variable to split on?

Play Tennis	Outlook	Temperature	Humidity	Windy
No	Sunny	Hot	High	No
No	Sunny	Hot	High	Yes
Yes	Overcast	Hot	High	No
Yes	Rainy	Mild	High	No
Yes	Rainy	Cold	Normal	No



Yes = 1
No = 2

Yes = 2
No = 0

How to quantify node purity using entropy?

How to choose which variable to split on?

Gini Impurity = 1 - (the probability of "Yes")^2 - (the probability of "No")^2

How to choose which variable to split on?

Play Tennis	Outlook	Temperature	Humidity	Windy
No	Sunny	Hot	High	No
No	Sunny	Hot	High	Yes
Yes	Overcast	Hot	High	No
Yes	Rainy	Mild	High	No
Yes	Rainy	Cold	Normal	No

Gini Impurity = 1 - (the probability of "Yes")^2 - (the probability of "No")^2

Temp == hot

Yes

No

Play Tennis

3/5

Yes = 1

No = 2

$1 - (2/3)^2 - (1/3)^2$

$3/5 * 0.44$
$= 0.27$

Keep splitting!

Play Tennis

Yes = 2

No = 0

2/5

$1 - (2/2)^2 - 0/2)^2$

$2/5 * 0$
$= 0$

Leaf!

Summary: Algorithm for Decision Trees

1. Start with an empty decision tree (undivided feature space).
2. Choose the ‘optimal’ variable on which to split and choose the ‘optimal’ threshold value for splitting.
    1. Often happens by going through all variables.
3. Recurse on each new node until stopping condition is met
4. For classification, we label each region in the model with the label of the class to which the plurality of the points within the region belong.

When do we stop training decision trees aka splitting the data into further nodes? (Select all that apply)

When do we stop training decision trees aka splitting the data into further nodes? (Select all that apply)

Quiz question · 68 answers · 68 participants

Keep splitting till all the data in a given node has the same label. - 39 answers ✓

57%

Keep splitting till the information gain = 0 - 59 answers ✓

87%

Keep splitting till the overall decision tree reaches a specific depth. - 41 answers ✓

60%

Keep splitting till a given node has two few examples - 25 answers ✓

37%

slido