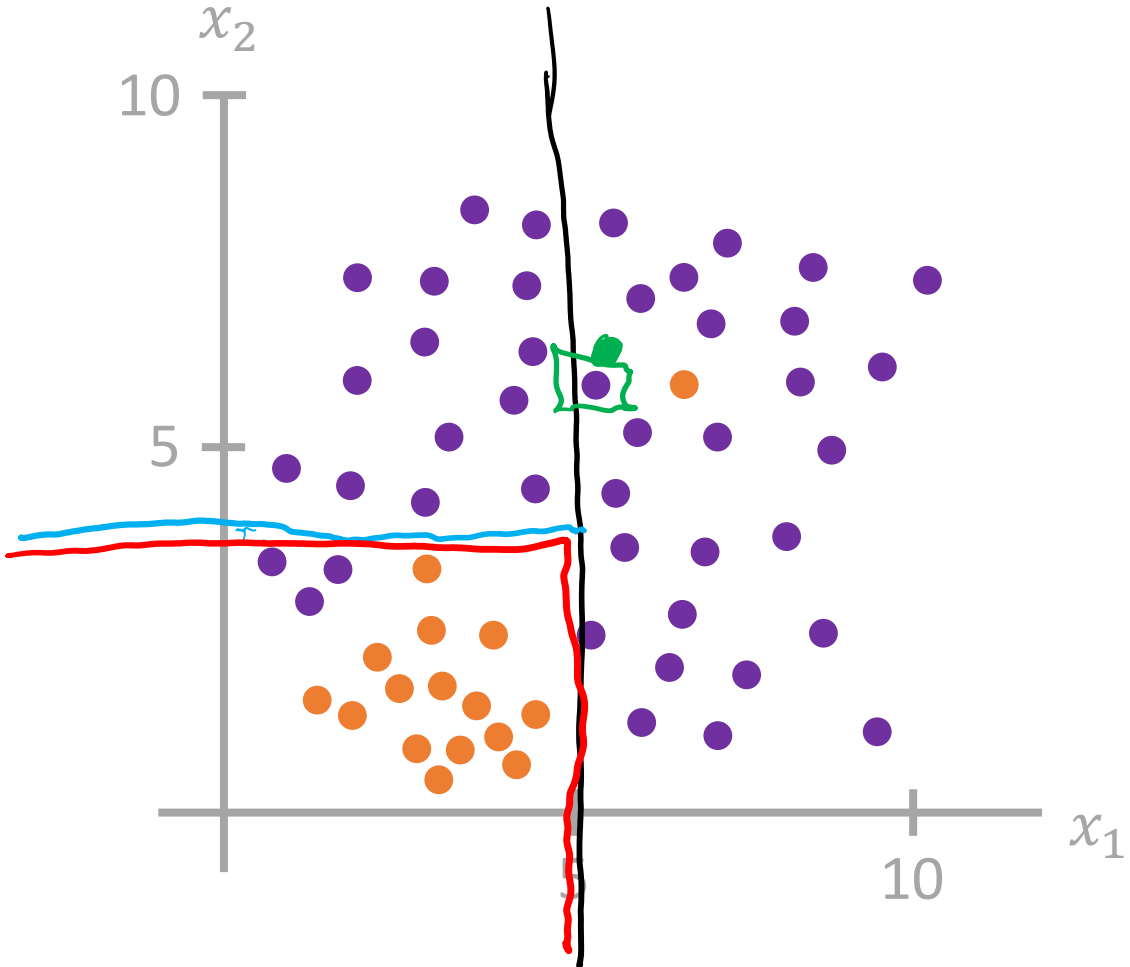
Warm-up as you walk in: Worksheet

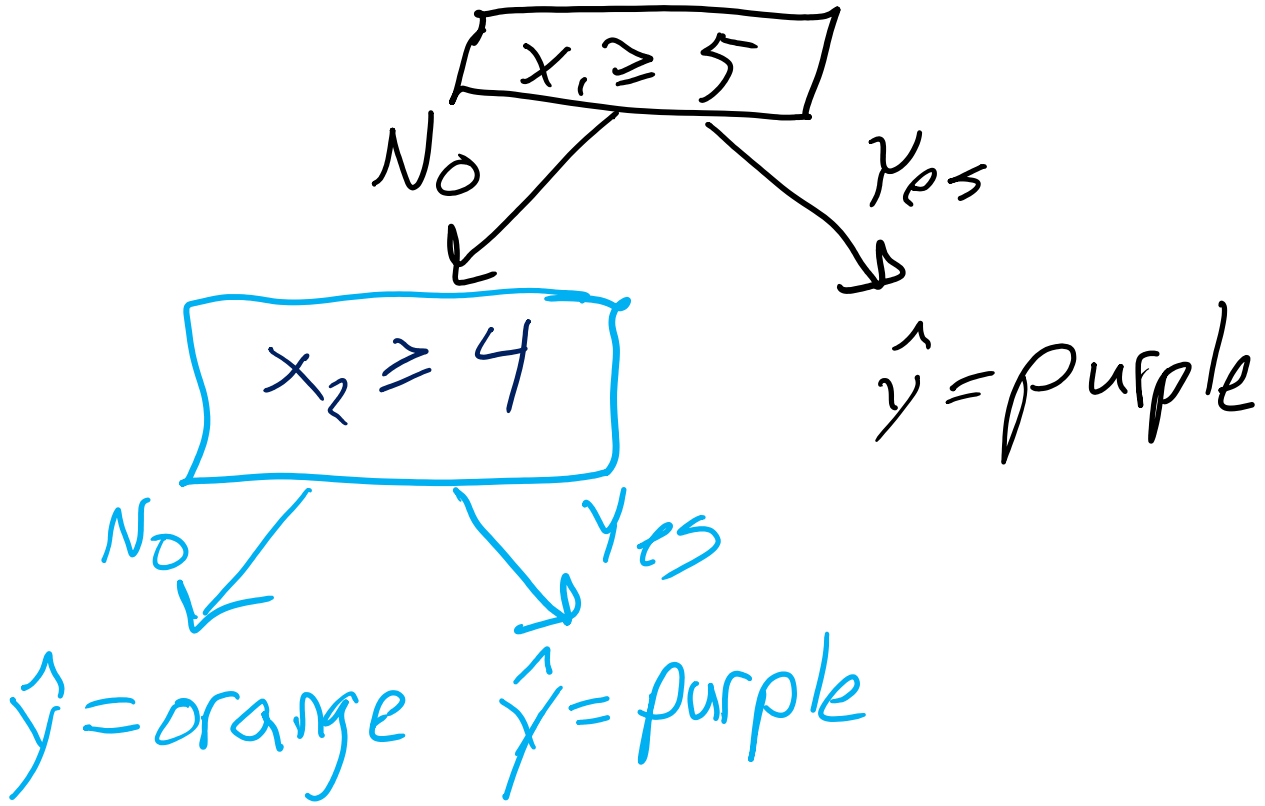
Consider input features $x \in \mathbb{R}^2$.

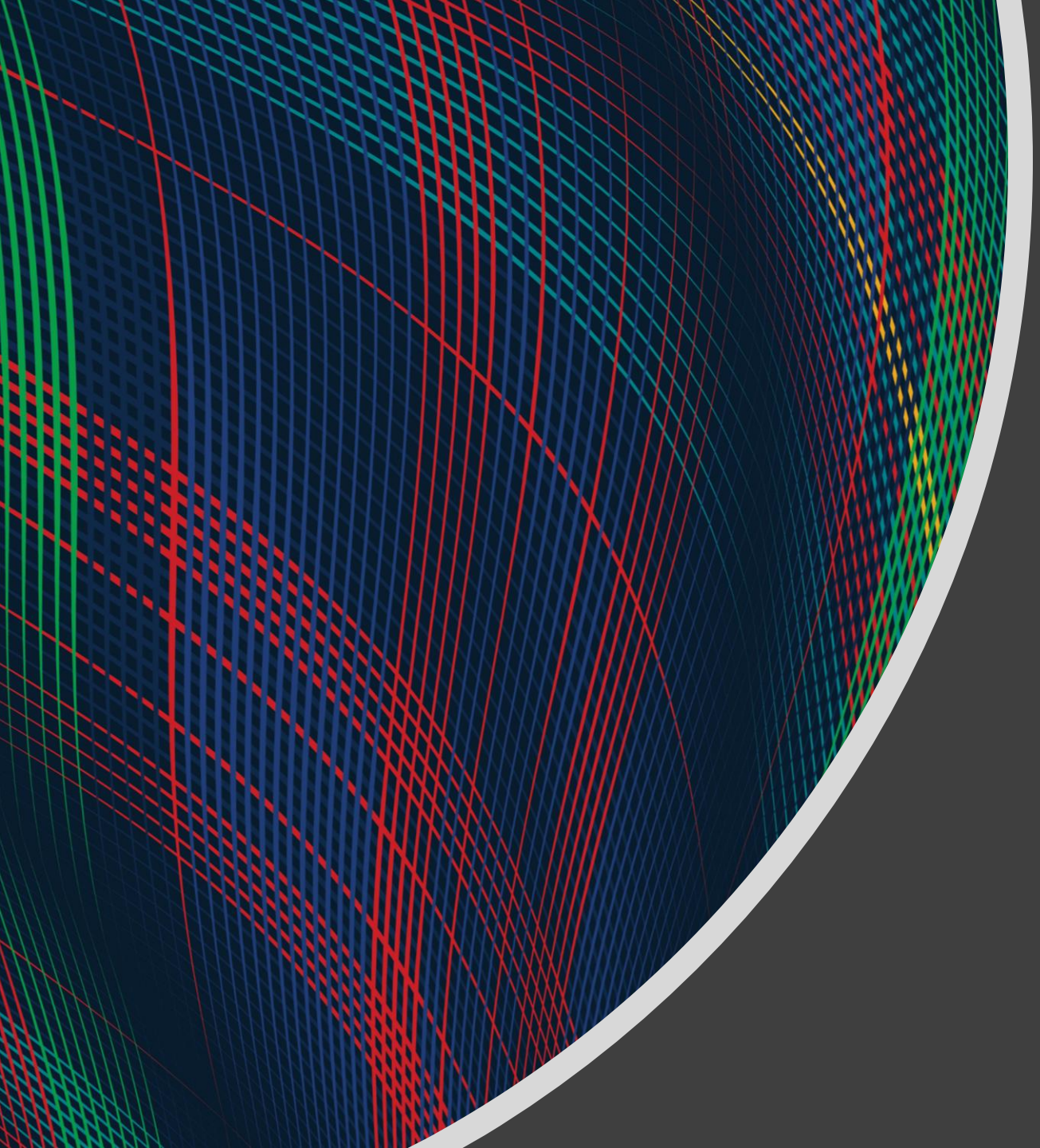
Draw a reasonable decision tree.



$\vec{x} \rightarrow \hat{y} = \text{purple}$

$\vec{x} \rightarrow \hat{y} = \text{orange}$





10-315 Introduction to ML

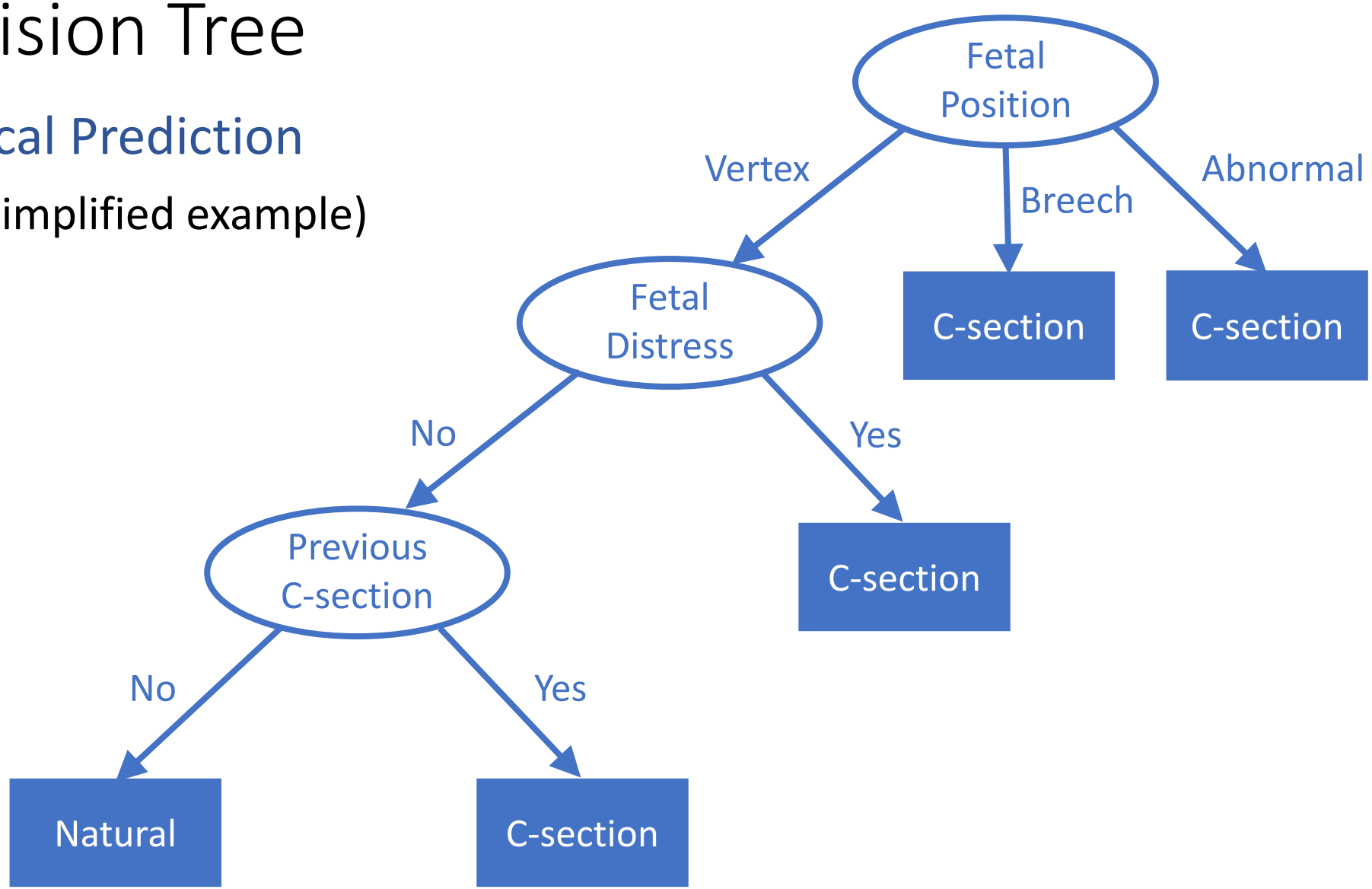
Decision Trees

Instructor: Pat Virtue

Decision Tree

Medical Prediction

(Oversimplified example)



Reminder: Machine Learning Problem Formulation

Three components <*T,P,E*>:

1. Task, *T*
2. Performance measure, *P*
3. Experience, *E*

Definition of learning:

A computer program **learns** if its performance at tasks in *T*, as measured by *P*, improves with experience *E*

Definition from (Mitchell, 1997)

Decision Trees

Why are we talking about decision trees?

Minimal prereqs: Doesn't rely on a ton of linear algebra, probability, or calc.

- So we can focus on some important ML concepts and notation, including model selection, overfitting/underfitting

Explainability

- Decision trees can be incredibly useful as they can more easily be interpreted and altered by humans than other ML algorithms

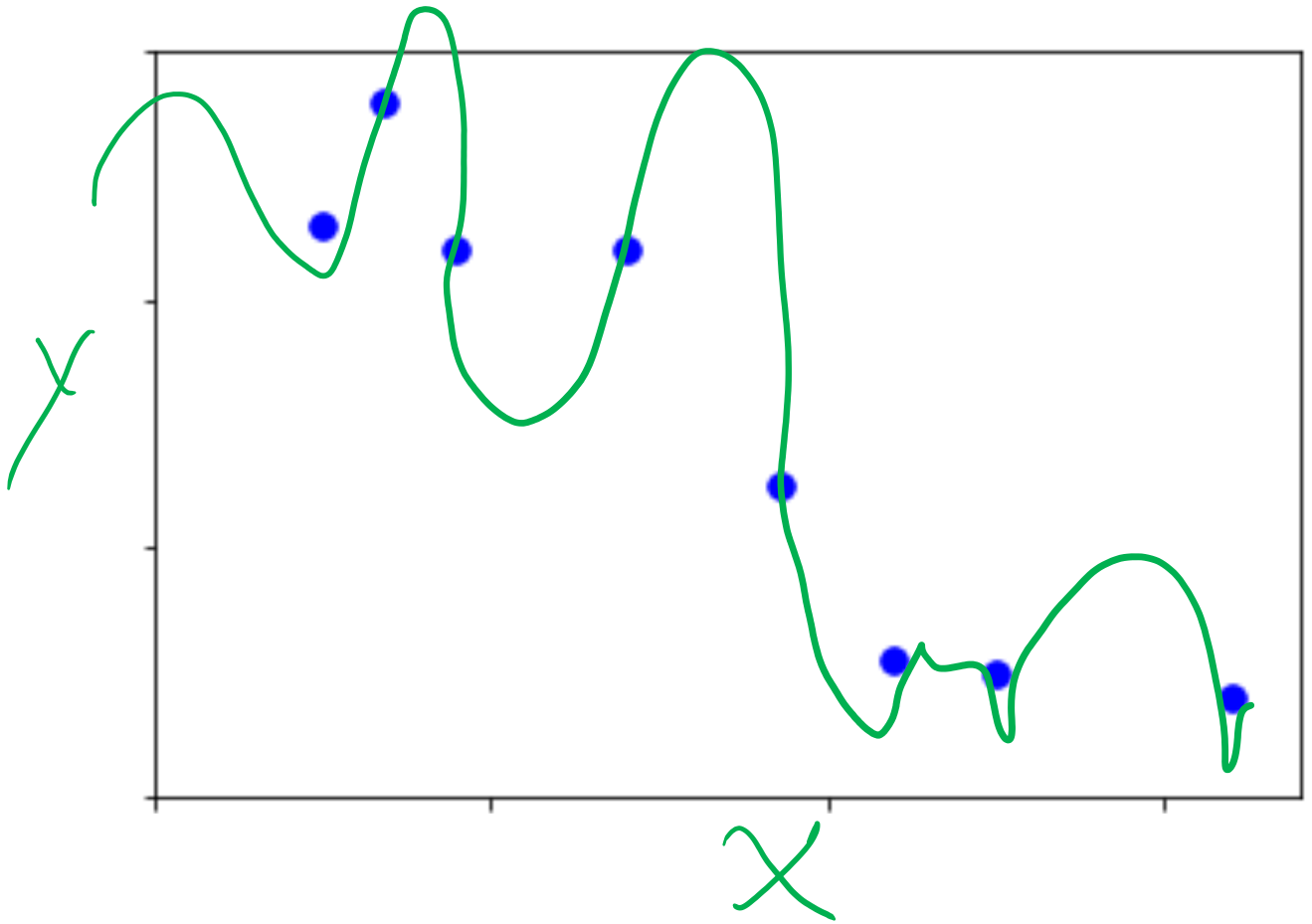
Basis of very powerful set of techniques: Random Forests

- Random forests train many simple decision trees (ML topic: ensemble learning)
- While powerful, random forests unfortunately have poor explainablility

Regression Model

Regression: learning a model to predict a numerical output (but not numbers that just represent categories, that would be classification)

Model



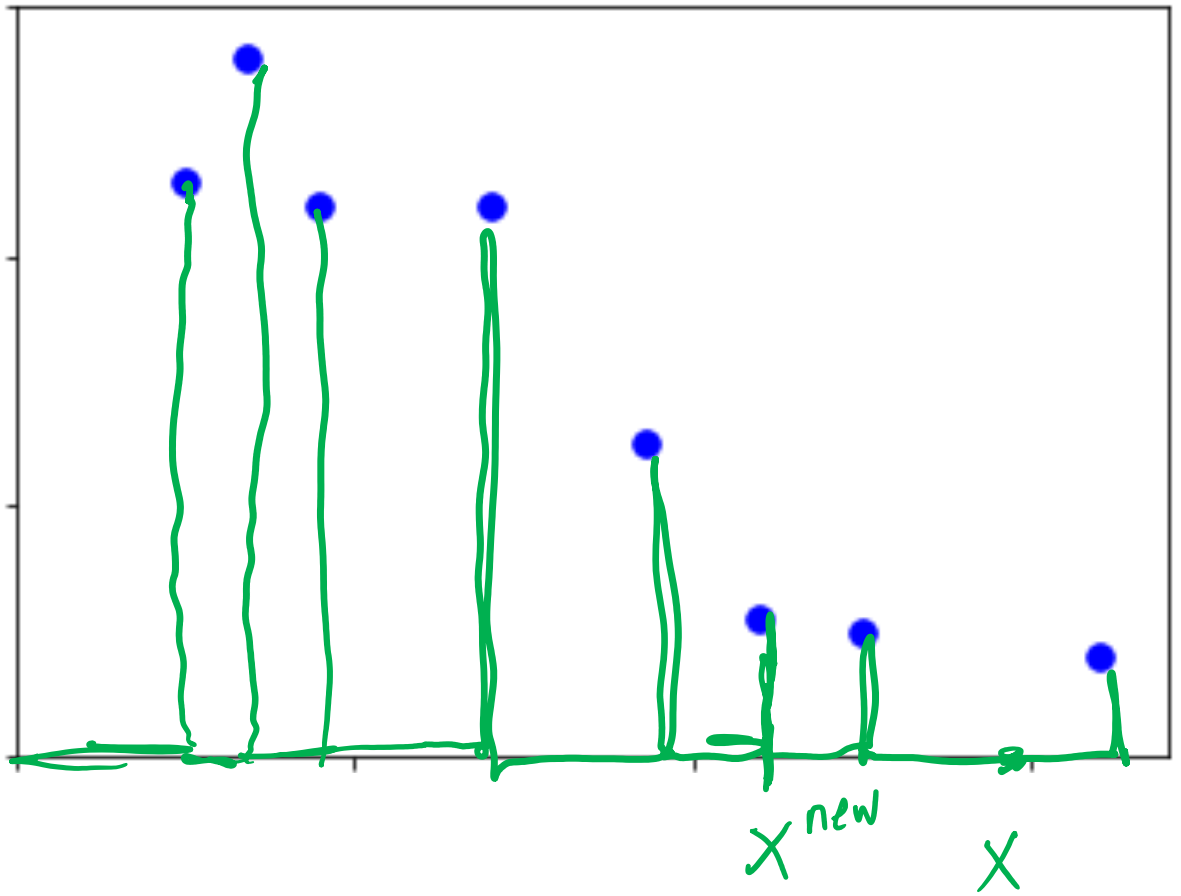
$\hat{y} = predict(x)$

Regression Model

Regression: learning a model to predict a numerical output (but not numbers that just represent categories, that would be classification)

Model: Memorization

```
def train(D)
    self.D = D

def predict(x_new)
    for x,y in self.D
        if x_new == x:
            return y
    return 0
```

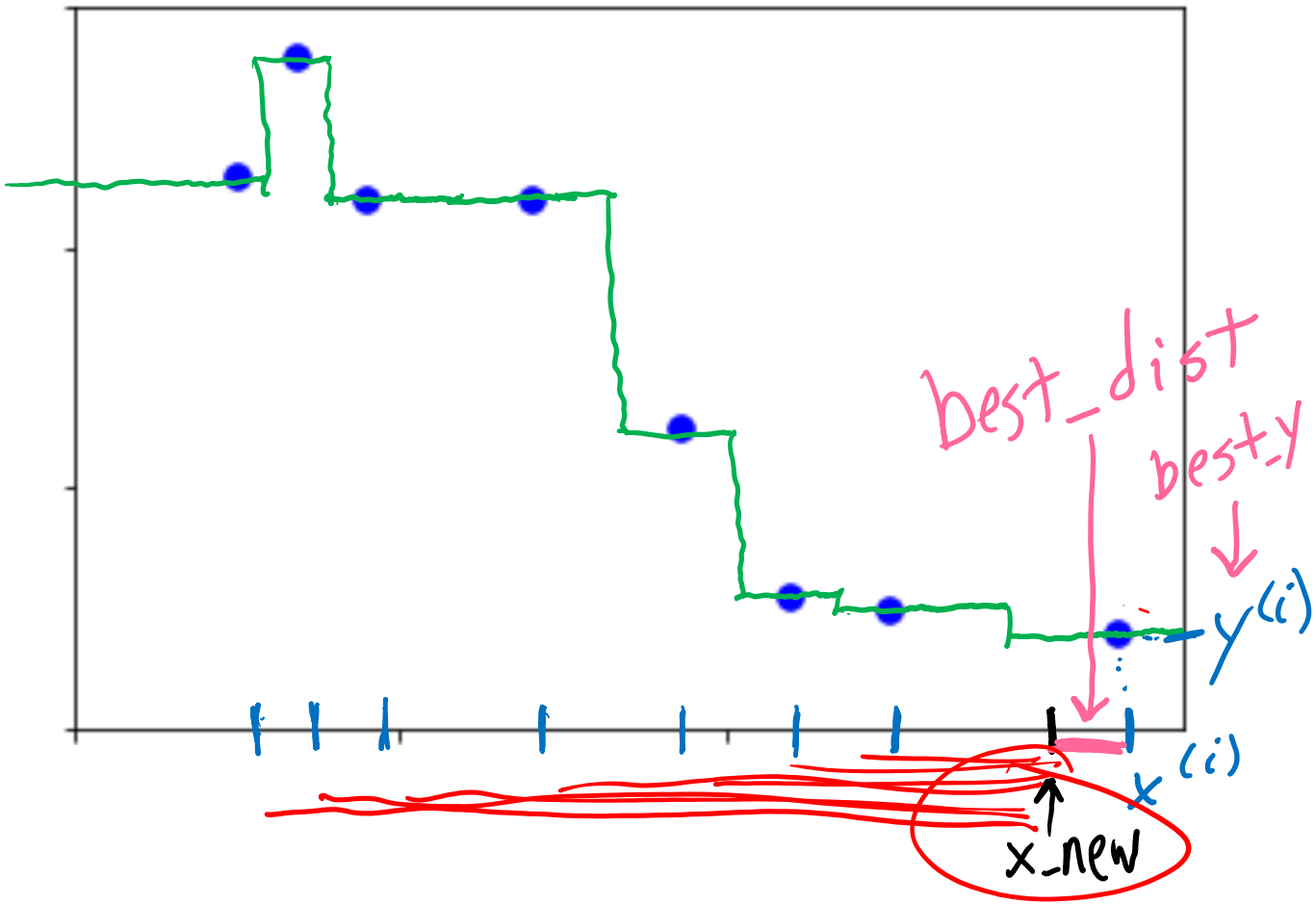


Regression Model

Regression: learning a model to predict a numerical output (but not numbers that just represent categories, that would be classification)

Model: Nearest neighbor

```
def train(D)
    self.D = D

def predict(x_new)
    for x,y in self.D
        if dist(x,x_new) < best_dist
            best_y = y
    return best_y
```



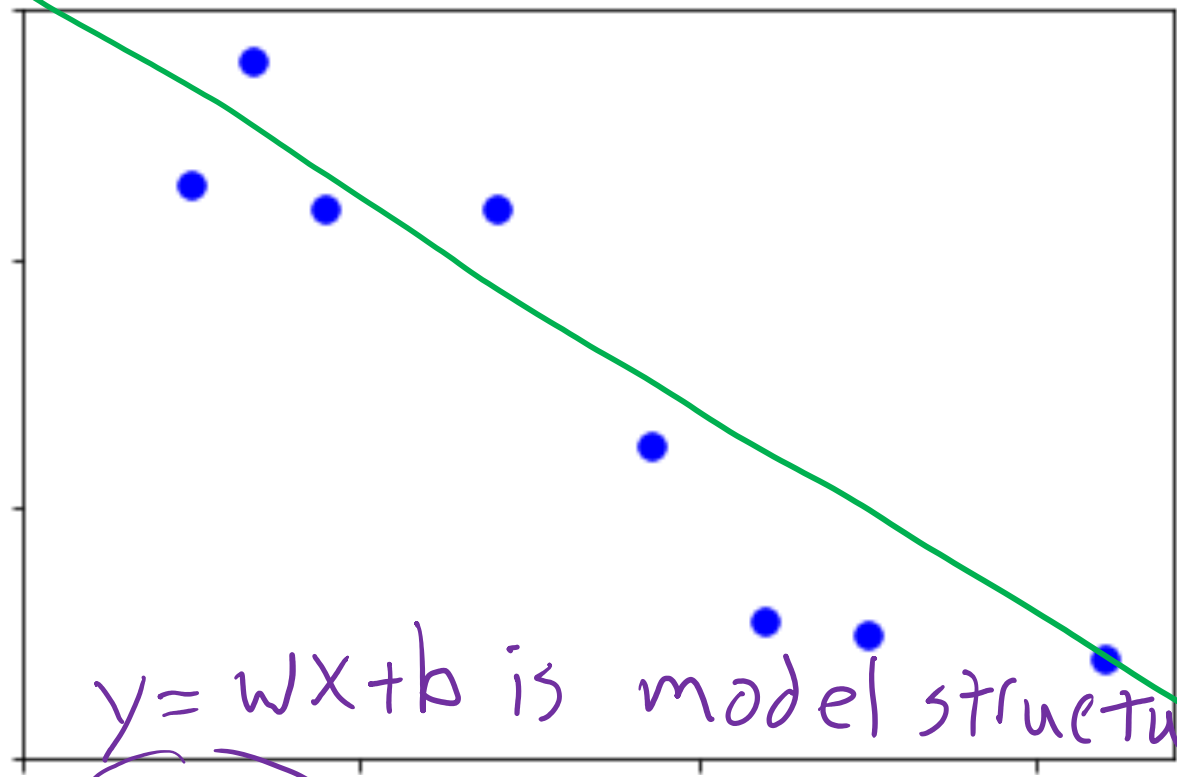
Regression Model

Regression: learning a model to predict a numerical output (but not numbers that just represent categories, that would be classification)

Model: Linear

def train(D)
Find best slope, w
and intercept, b

def predict(x_new)
return w*x_new+b



y = wx+b is model structure

θ → (w, b) are model parameters

Poll 1

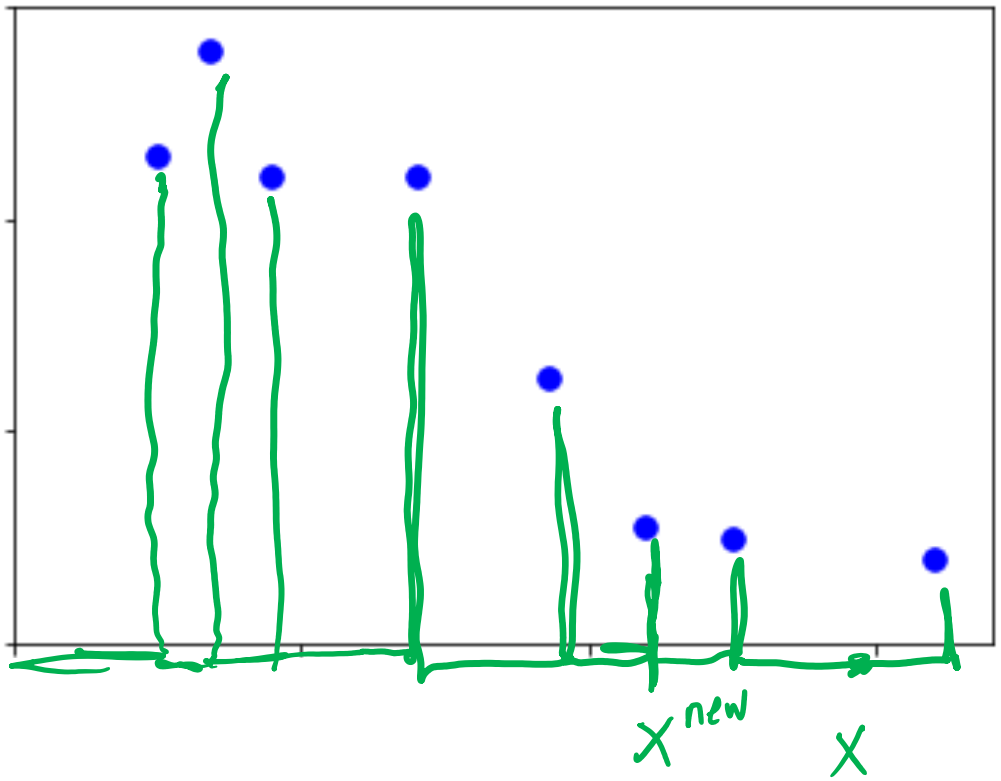
Does the memorization algorithm learn?

A. Yes

B. No

C. I have no clue

```
def train(D)
    self.D = D

def predict(x_new)
    for x,y in self.D
        if x_new == x:
            return y
    return 0
```

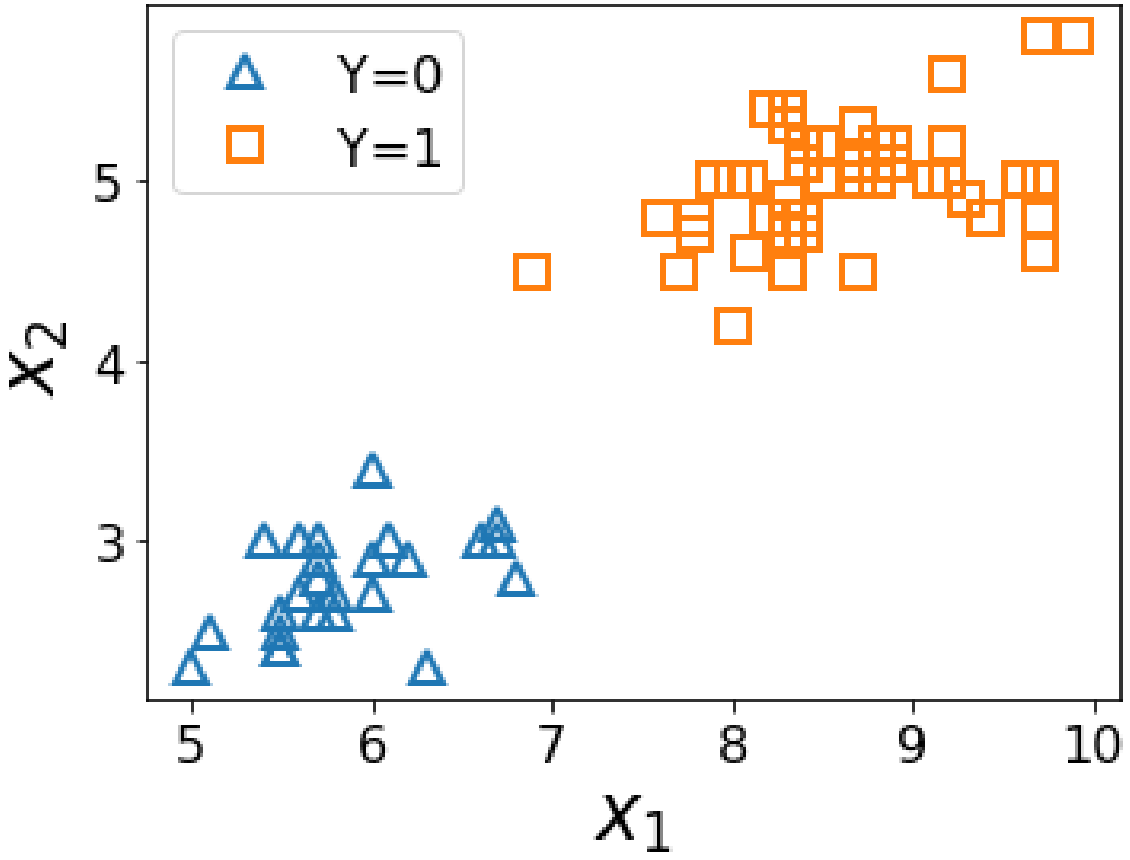


Slide credit: CMU MLD Matt Gormley

ML Task: Classification

Predict species label from first two input measurements

$h(\mathbf{x}) \rightarrow \hat{y}$



Species	Sepal Length	Sepal Width
0	4.3	3.0
0	4.9	3.6
0	5.3	3.7
1	4.9	2.4
1	5.7	2.8
1	6.3	3.3

Images and full dataset: https://en.wikipedia.org/wiki/Iris_flower_data_set

Problem Formulation

Medical Prediction

y	x_1	x_2	x_3
Outcome	**Fetal Position**	**Fetal Distress**	**Previous C-sec**
Natural	Vertex	N	N
C-section	Breech	N	N
Natural	Vertex	Y	Y
C-section	Vertex	N	Y
Natural	Abnormal	N	N

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = [x_1, x_2, x_3]^T$$

$x_1 \in \{Vertex, Breech, Abn\}$
$x_2 \in \{Y, N\}$
$x_3 \in \{Y, N\}$

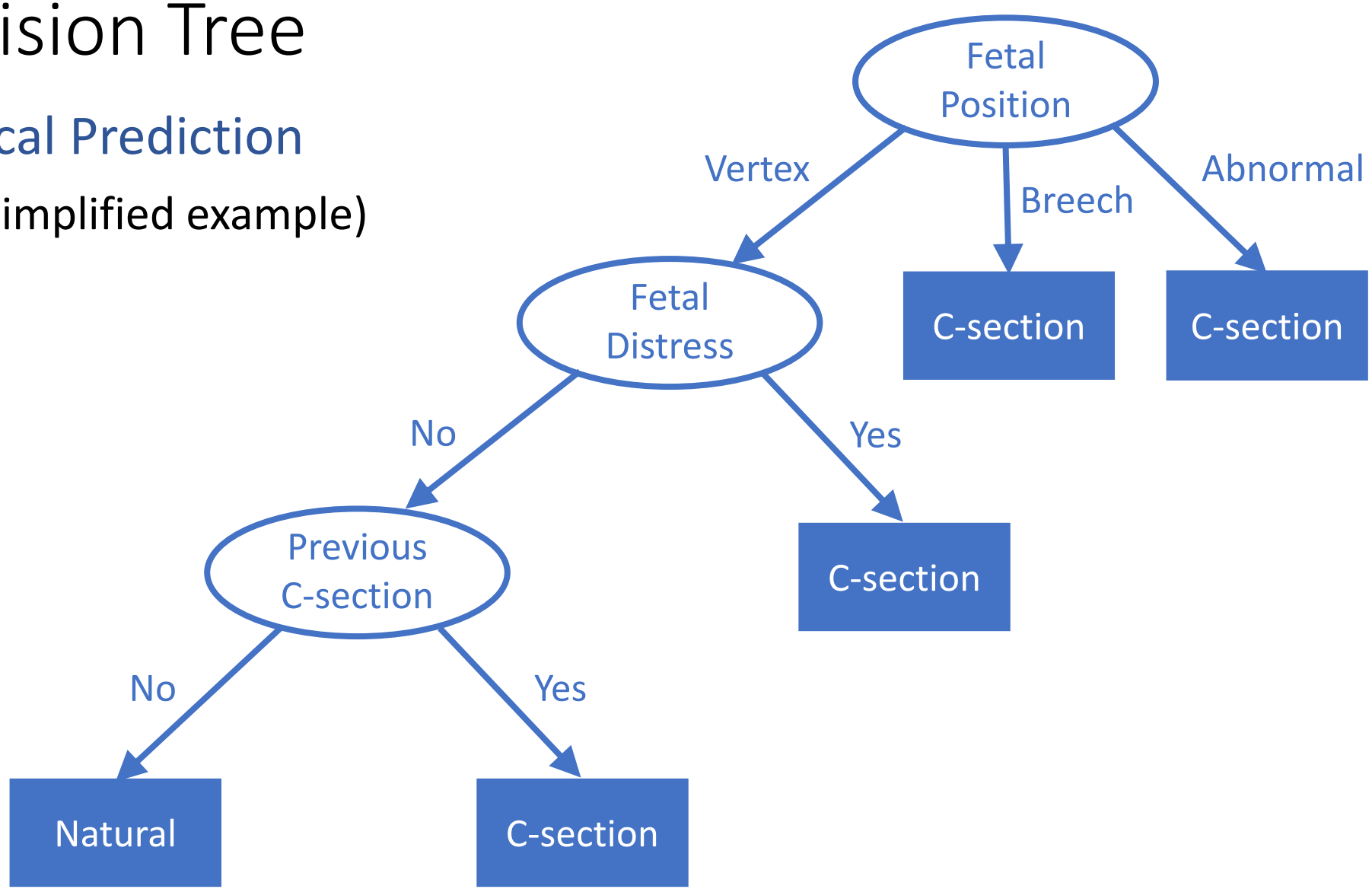
$y \in \{Csection, Natural\}$

$\hat{y} = h(\boldsymbol{x})$

Decision Tree

Medical Prediction

(Oversimplified example)



Decision Trees

A few tools

Species	Sepal Length	Sepal Width	Petal Length	Petal Width
0	4.3	3.0	1.1	0.1
0	4.9	3.6	1.4	0.1
0	5.3	3.7	1.5	0.2
1	4.9	2.4	3.3	1.0
1	5.7	2.8	4.1	1.3
1	6.3	3.3	4.7	1.6
2	5.9	3.0	5.1	1.8

N=7 $N_2 = 1$ $\frac{3}{7}$ $\frac{3}{7}$ $\frac{1}{7}$

Majority vote:

$$\hat{y} = \underset{c}{\operatorname{argmax}} \frac{N_c}{N}$$

Classification error rate:

$$ErrorRate = \frac{1}{N}\sum_i \mathbb{I}\left(y^{(i)} \neq \hat{y}^{(i)}\right)$$

What fraction did we predict incorrectly

Expected value

RV value

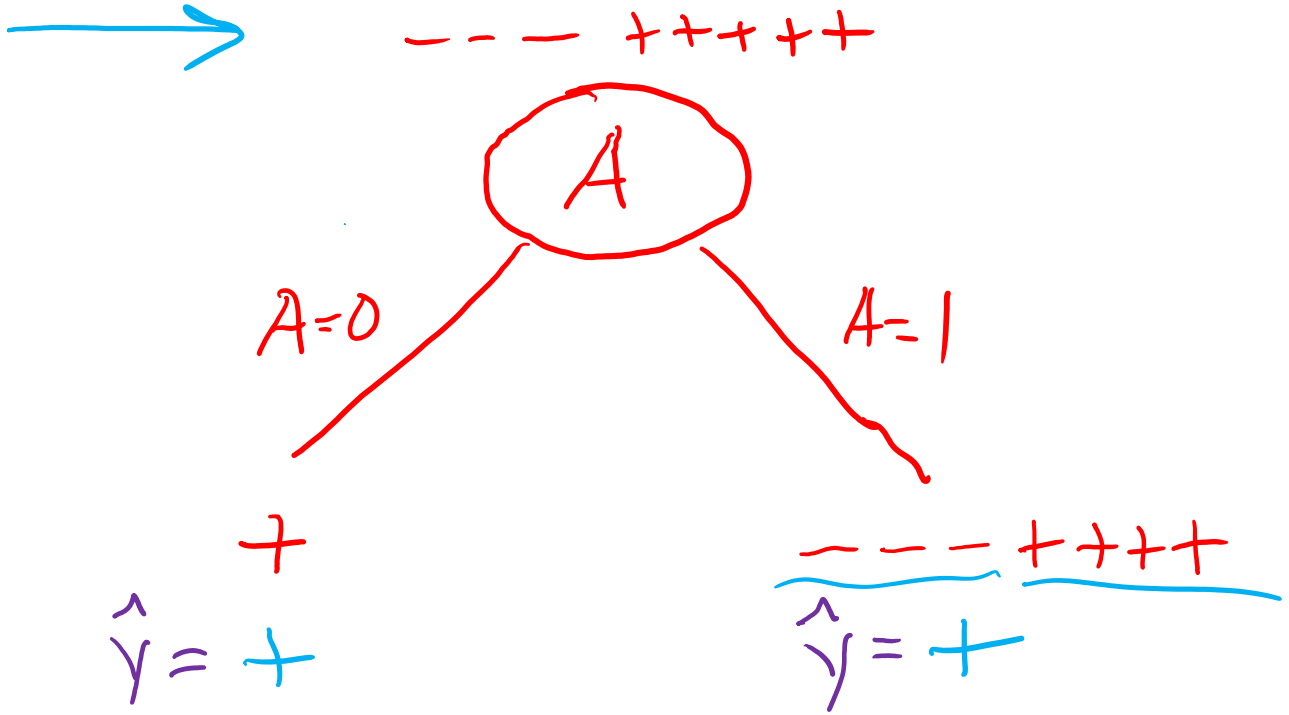
$$\mathbb{E}[f(X)] = \sum_{x \in \mathcal{X}} f(x)\, P(X = x) \quad \text{or} \quad \mathbb{E}[f(X)] = \int_{\mathcal{X}} f(x)\, p(x)\, dx$$

$P(Y = y \mid X = x)$

Decision Stumps

Split data based on a single attribute

Majority vote at leaves



Dataset:

Output Y, Attributes A, B, C

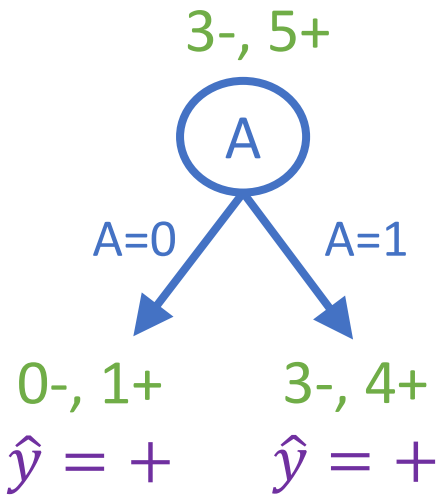
Y	A	B	C
-	1	0	0
-	1	0	1
-	1	0	0
+	0	0	1
+	1	1	0
+	1	1	1
+	1	1	0
+	1	1	1

Slide credit: CMU MLD Matt Gormley

Decision Stumps

Split data based on a single attribute

Majority vote at leaves



Error: (0 + 3) / 8

= 3/8

Dataset:

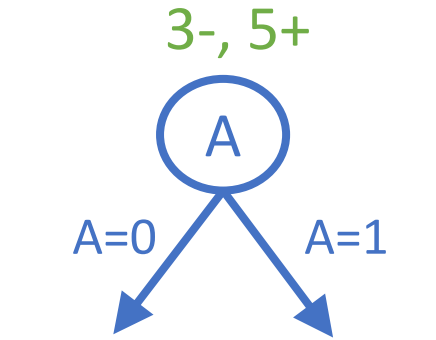
Output Y, Attributes A, B, C

Y	A	B	C
-	1	0	0
-	1	0	1
-	1	0	0
+	0	0	1
+	1	1	0
+	1	1	1
+	1	1	0
+	1	1	1

Slide credit: CMU MLD Matt Gormley

Poll 2

Splitting on which attribute {A, B, C} creates a decision stump with the lowest training error?



$\hat{y} = +$ $\hat{y} = +$

Error: (0 + 3) / 8

= 3/8

Dataset:

Output Y, Attributes A, B, C

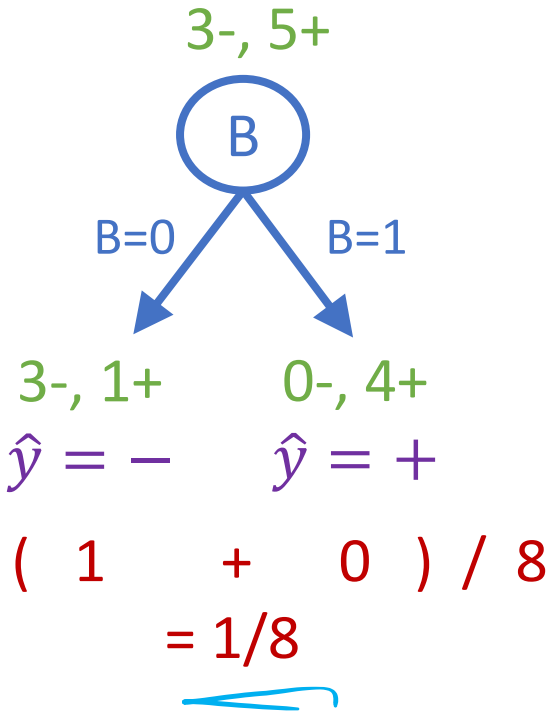
Y	A	B	C
-	1	0	0
-	1	0	1
-	1	0	0
+	0	0	1
+	1	1	0
+	1	1	1
+	1	1	0
+	1	1	1

Slide credit: CMU MLD Matt Gormley

Poll 2

Splitting on which attribute {A, B, C} creates a decision stump with the lowest training error?

Answer: B



Dataset:

Output Y, Attributes A, B, C

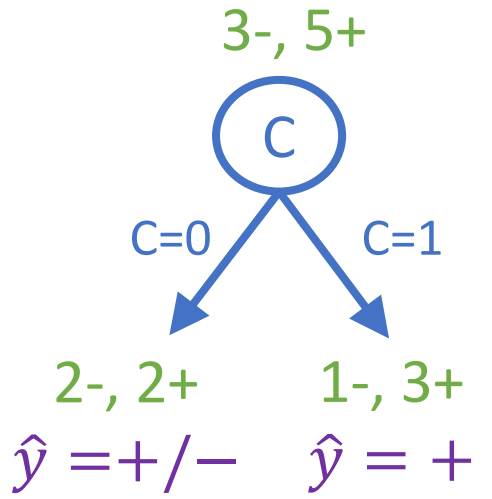
Y	A	B	C
-	1	0	0
-	1	0	1
-	1	0	0
+	0	0	1
+	1	1	0
+	1	1	1
+	1	1	0
+	1	1	1

Slide credit: CMU MLD Matt Gormley

Poll 2

Splitting on which attribute {A, B, C} creates a decision stump with the lowest training error?

Answer: B



Error: (2 + 1) / 8
= 3/8

Dataset:

Output Y, Attributes A, B, C

Y	A	B	C
-	1	0	0
-	1	0	1
-	1	0	0
+	0	0	1
+	1	1	0
+	1	1	1
+	1	1	0
+	1	1	1

Slide credit: CMU MLD Matt Gormley

Building a Decision Tree

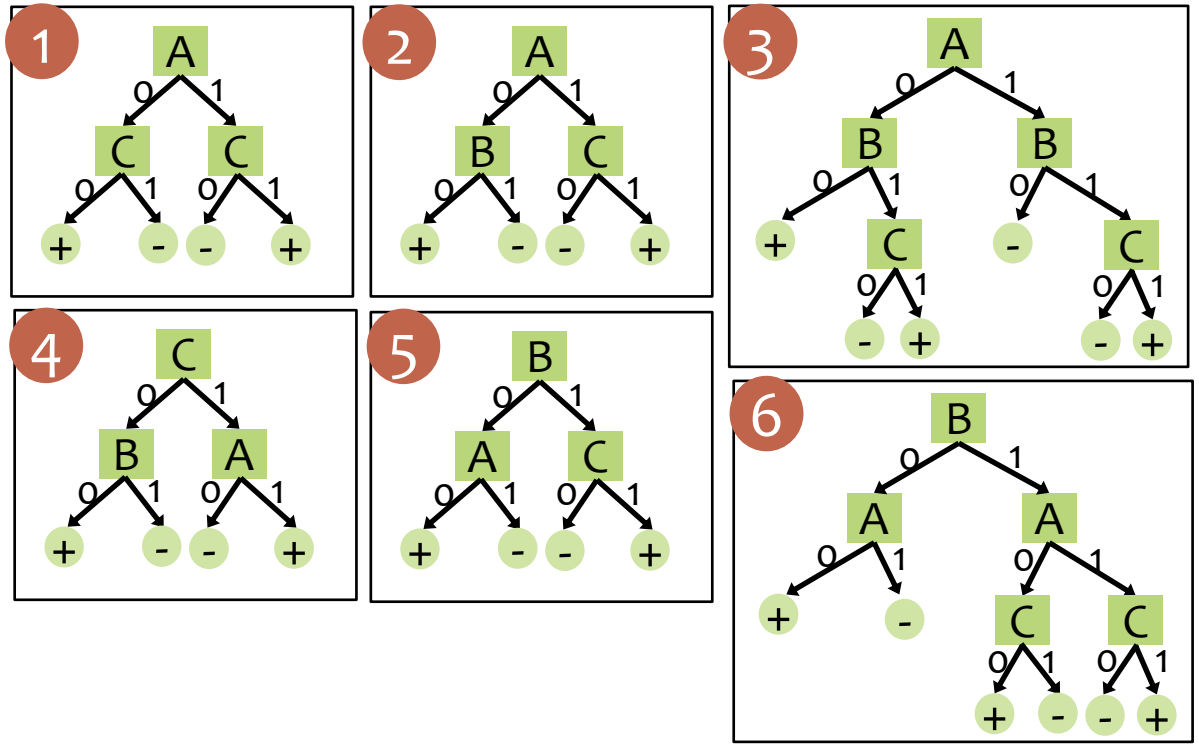
Binary, Doesn't reuse attributes

```
Function BuildTree(D, Attributes)
    # D: dataset at current node
    # Attributes : current set of attributes

    # TODO Base Case


    else
        # Internal node
        X ⇐ bestAttribute(D, Attributes)
        LeftNode = BuildTree(D(X=1), Attributes \ {X})
        RightNode = BuildTree(D(X=0), Attributes \ {X})
    end
end
```

Poll 3

Which of the following trees would be learned by the decision tree learning algorithm using "error rate" as the splitting criterion?

(Assume ties are broken alphabetically.)



Dataset:

Output Y, Attributes A, B, C

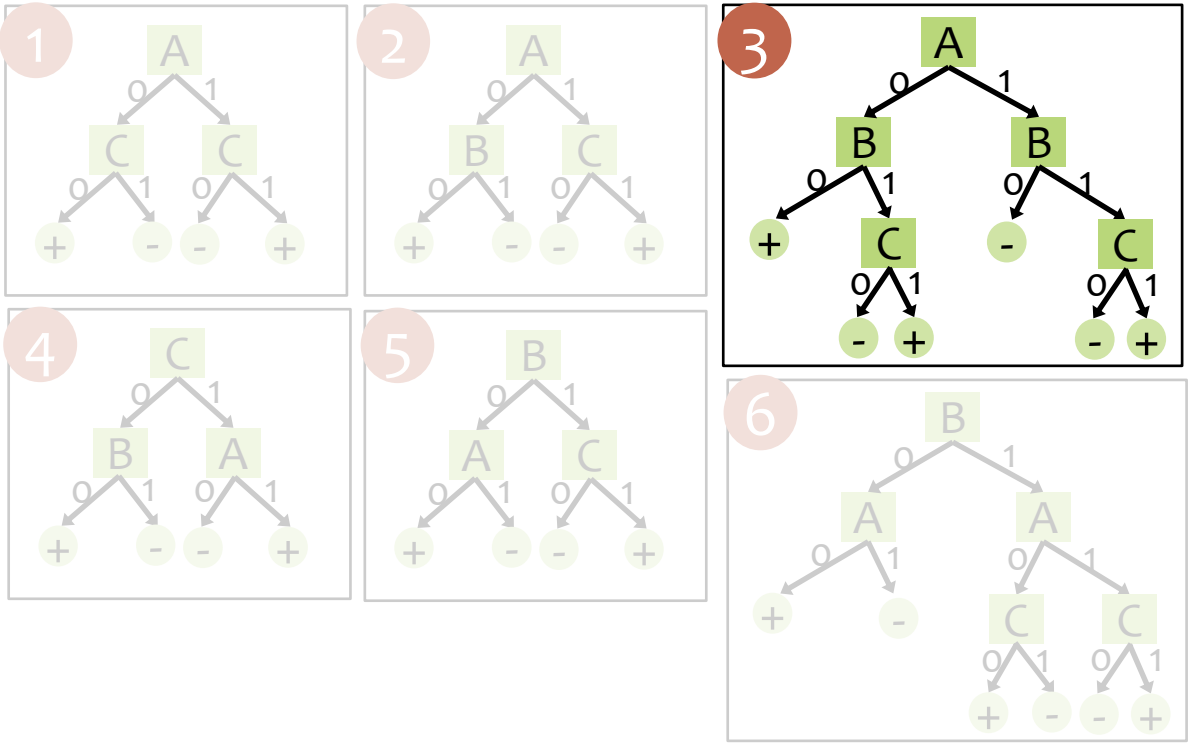
Y	A	B	C
+	0	0	0
+	0	0	1
-	0	1	0
+	0	1	1
-	1	0	0
-	1	0	1
-	1	1	0
+	1	1	1

Slide credit: CMU MLD Matt Gormley

Poll 3

Which of the following trees would be learned by the the decision tree learning algorithm using “error rate” as the splitting criterion?

(Assume ties are broken alphabetically.)



Dataset:

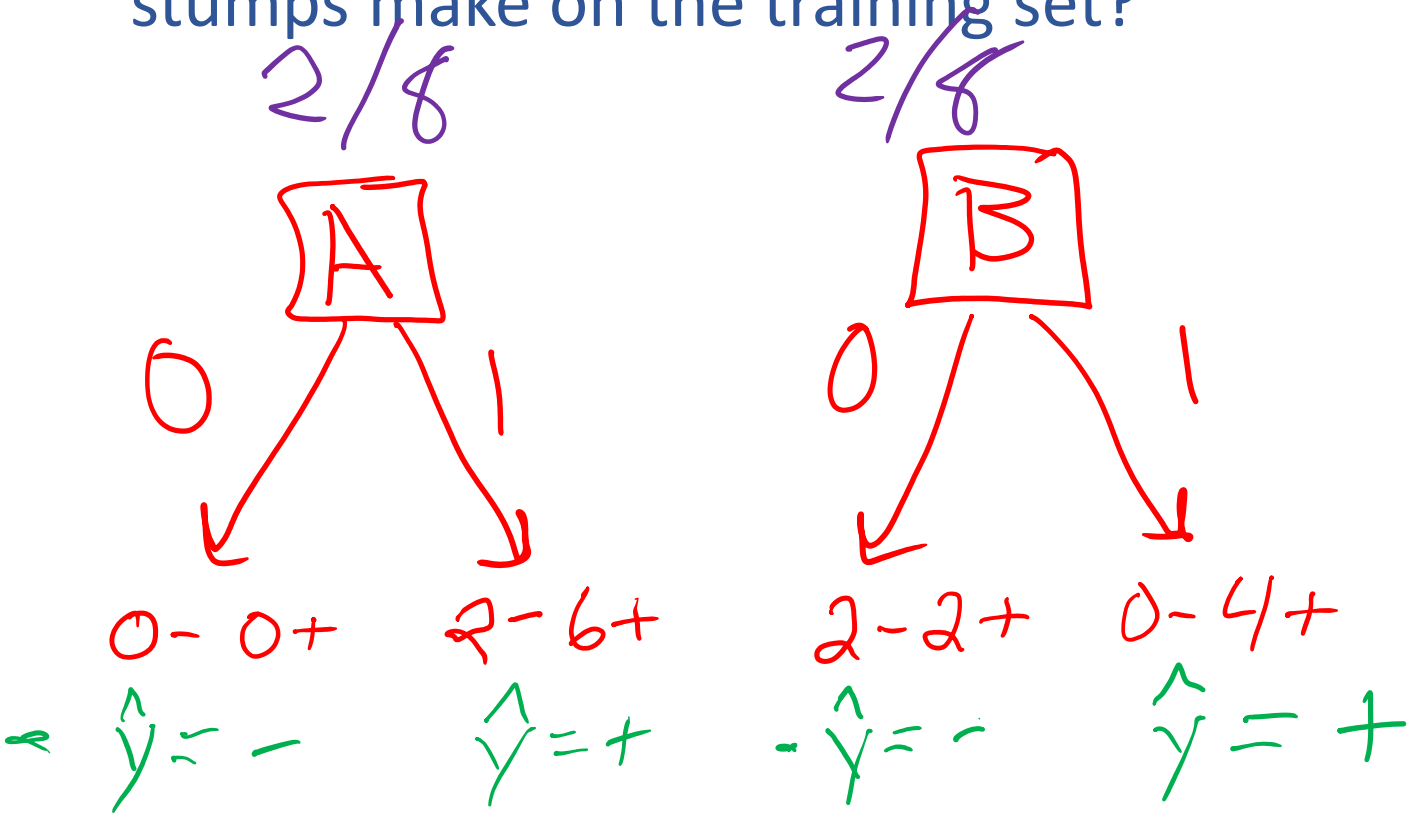
Output Y, Attributes A, B, C

Y	A	B	C
+	0	0	0
+	0	0	1
-	0	1	0
+	0	1	1
-	1	0	0
-	1	0	1
-	1	1	0
+	1	1	1

Slide credit: CMU MLD Matt Gormley

Poll 4

How many errors do each of the two decision stumps make on the training set?



Dataset:

Output Y, Attributes A and B

Y	A	B
-	1	0
-	1	0
+	1	0
+	1	0
+	1	1
+	1	1
+	1	1
+	1	1

Slide credit: CMU MLD Matt Gormley

Building a Decision Tree

```
Function BuildTree(D, Attributes)
    # D: dataset at current node
    # Attributes : current set of attributes

    # TODO Base Case


    else
        # Internal node
        X ⇐ bestAttribute(D, Attributes)
        LeftNode = BuildTree(D(X=1), Attributes \ {X})
        RightNode = BuildTree(D(X=0), Attributes \ {X})
    end
end
```

Options

X error rate

best based on splitting criteria

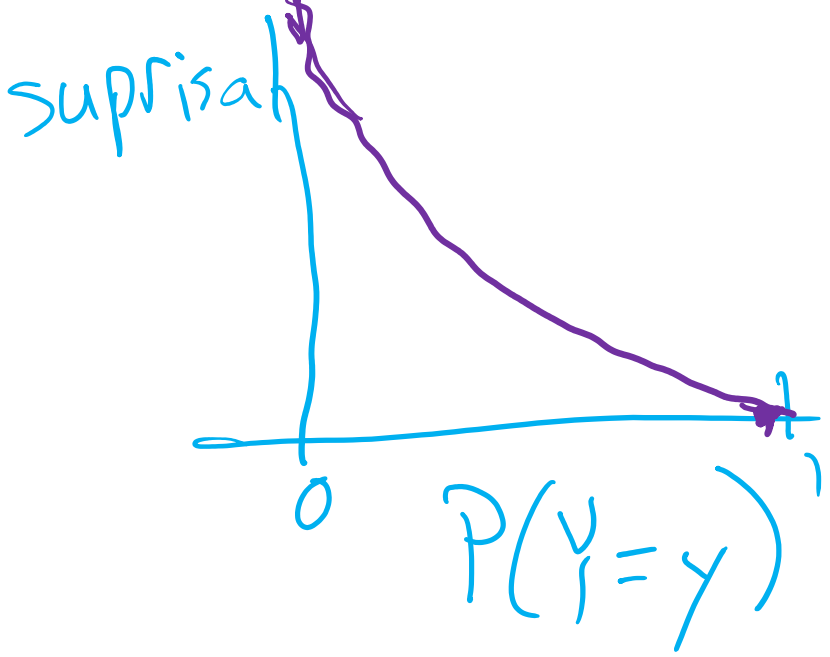
- gini impurity

→ Mutual information (info. gain)

Entropy

Surprisal

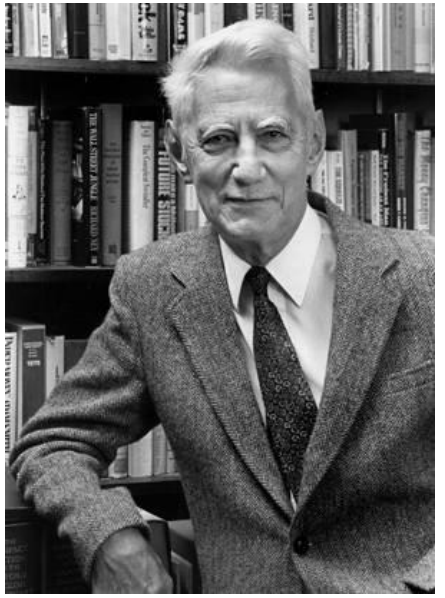
$$\log_2 \frac{1}{P(Y=y)}$$



Entropy

$$\mathbb{E}_Y[\text{surprise}] = \mathbb{E}_Y \log_2 \frac{1}{P(Y=y)}$$

$$= p(Y=1) \log_2 \frac{1}{P(Y=1)} + p(Y=0) \log_2 \frac{1}{P(Y=0)}$$



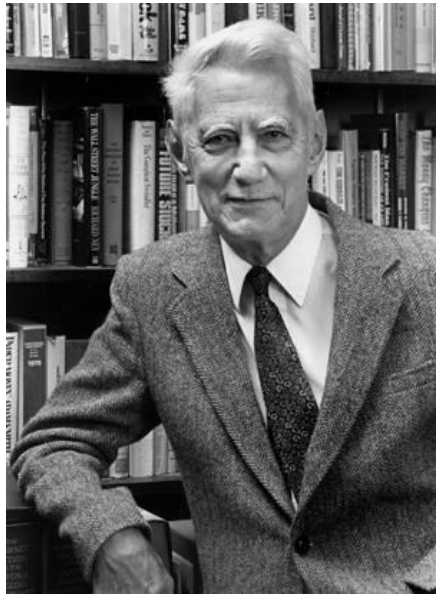
Claude Shannon (1916 – 2001), most of the work was done in Bell labs

Entropy

- Quantifies the amount of uncertainty associated with a specific probability distribution
- The higher the entropy, the less confident we are in the outcome
- Definition

$$H(X) = \sum_x p(X = x) \log_2 \frac{1}{p(X = x)}$$

$$H(X) = -\sum_x p(X = x) \log_2 p(X = x)$$



Claude Shannon (1916 – 2001), most of the work was done in Bell labs

Conditional Entropy

Entropy Definition

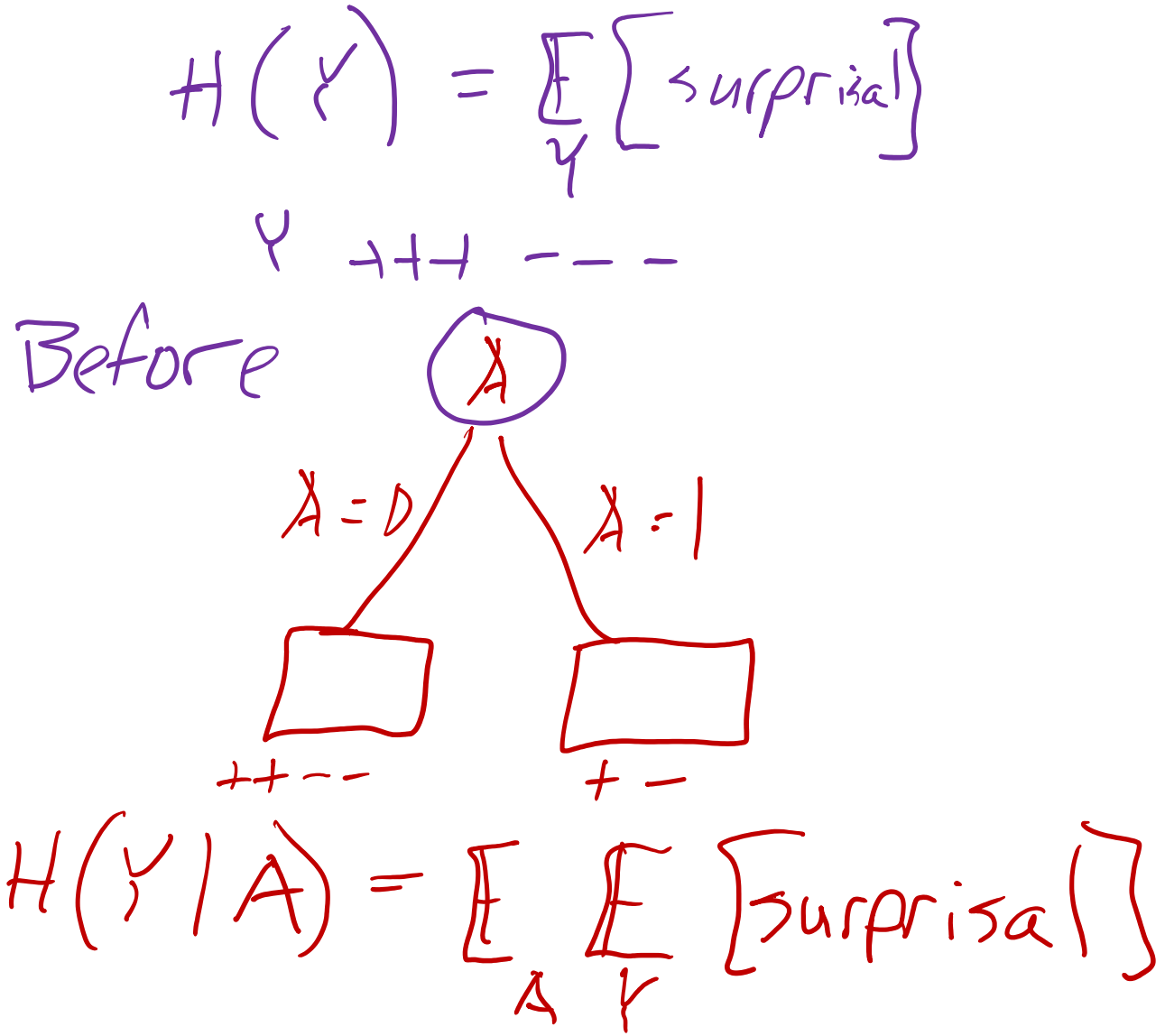
$$H(Y) = \sum_y p(Y = y) \log_2 \frac{1}{p(Y=y)}$$

$$H(Y) = -\sum_y p(Y = y) \log_2 p(Y = y)$$

Conditional Entropy

Entropy after splitting on a particular feature

- Must consider expected value over both branches!



Conditional Entropy

Entropy Definition

$$H(Y) = \sum_y p(Y = y) \log_2 \frac{1}{p(Y=y)}$$

$$H(Y) = -\sum_y p(Y = y) \log_2 p(Y = y)$$

Conditional Entropy

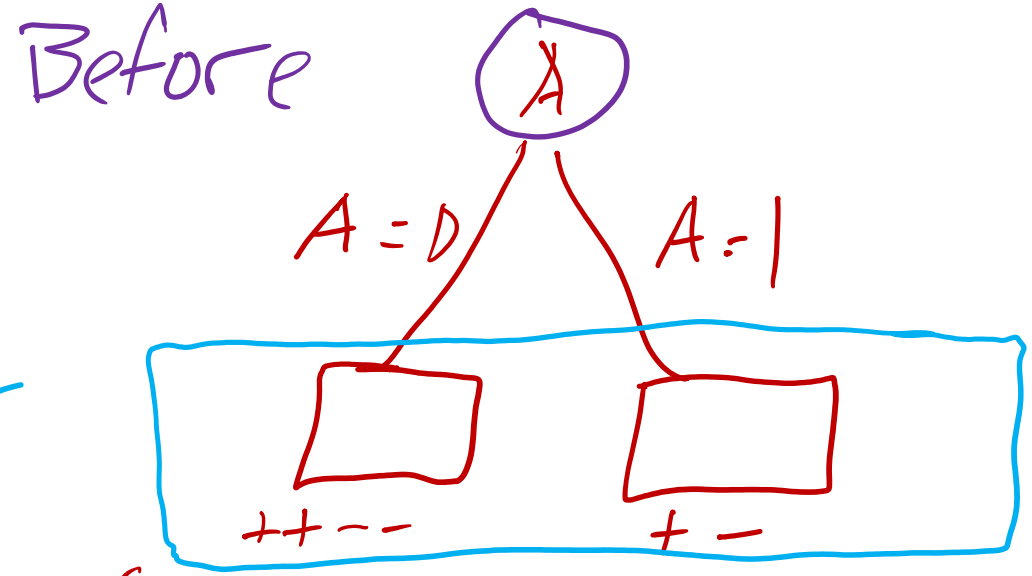
Entropy after splitting on a particular feature

- Must consider expected value over both branches!

Mutual Information: $I(Y; X) = H(Y) - H(Y|X)$

$H(Y) = \mathbb{E}_Y[\text{surprisal}]$

Y +++ ---



$H(Y|A) = \mathbb{E}_A \mathbb{E}_Y[\text{surprisal}]$

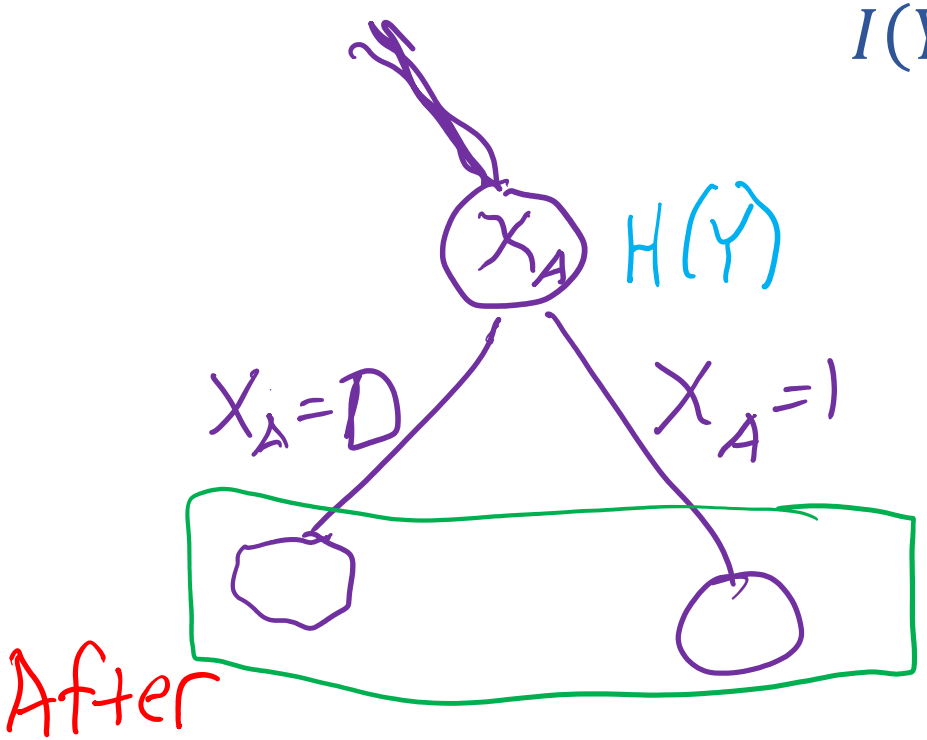
Before → After

Mutual Information Notation

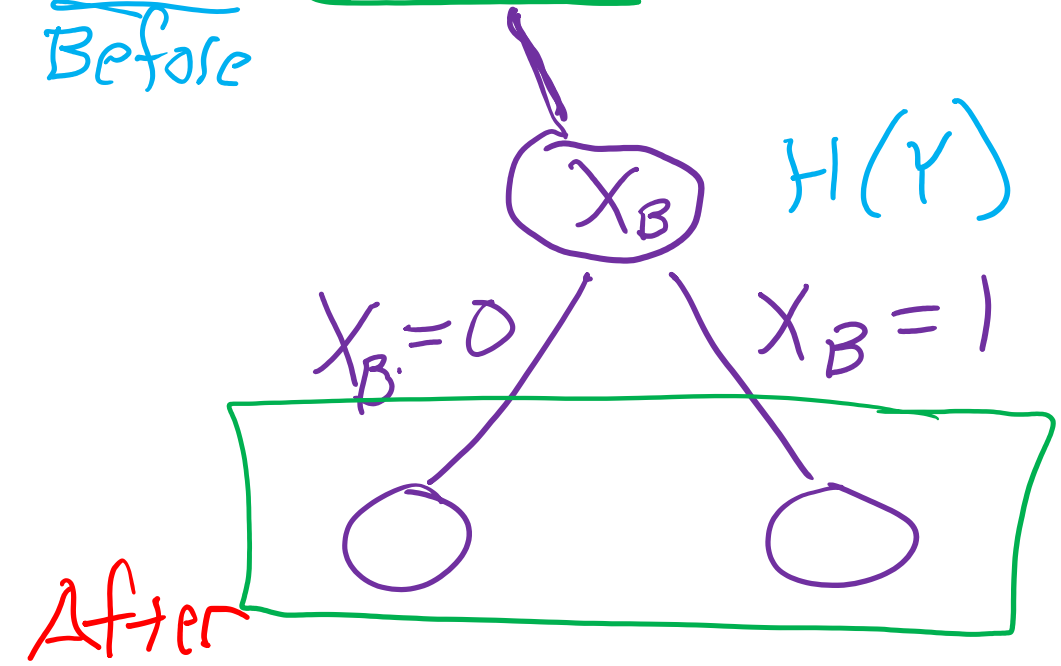
We use mutual information in the context of before and after a split, regardless of where that split is in the tree.

$$I(Y;X) = H(Y) - H(Y \mid X)$$

Before



$$I(Y;X_A) = H(Y) - H(Y|X_A)$$



$$I(Y;X_B) = H(Y) - H(Y|X_B)$$

Mutual Information

Let X be a random variable with $X \in \mathcal{X}$.
Let Y be a random variable with $Y \in \mathcal{Y}$.

Entropy: $H(Y) = -\sum_{y \in \mathcal{Y}} P(Y = y) \log_2 P(Y = y)$

Specific Conditional Entropy: $H(Y \mid X = x) = -\sum_{y \in \mathcal{Y}} P(Y = y \mid X = x) \log_2 P(Y = y \mid X = x)$

Conditional Entropy: $H(Y \mid X) = \sum_{x \in \mathcal{X}} P(X = x) H(Y \mid X = x)$

Mutual Information: $I(Y; X) = H(Y) - H(Y|X)$

Slide credit: CMU MLD Matt Gormley

Mutual Information

Let X be a random variable with $X \in \mathcal{X}$.
Let Y be a random variable with $Y \in \mathcal{Y}$.

Entropy: $H(Y) = -\sum_{y \in \mathcal{Y}} P(Y = y) \log_2 P(Y = y)$

Specific Conditional Entropy: $H(Y \mid X = x) = -\sum_{y \in \mathcal{Y}} P(Y = y \mid X = x) \log_2 P(Y = y \mid X = x)$

Conditional Entropy: $H(Y \mid X) = \sum_{x \in \mathcal{X}} P(X = x) H(Y \mid X = x)$

Mutual Information: $I(Y; X) = H(Y) - H(Y|X)$

- For a decision tree, we can use **mutual information** of the output class *Y* and some attribute *X* on which to split **as a splitting criterion**
- Given a dataset *D* of training examples, we can estimate the required probabilities as…

$$P(Y = y) = N_{Y=y}/N$$
$$P(X = x) = N_{X=x}/N$$
$$P(Y = y|X = x) = N_{Y=y,X=x}/N_{X=x}$$

where $N_{Y=y}$ is the number of examples for which $Y = y$ and so on.

Slide credit: CMU MLD Matt Gormley

Mutual Information

Let X be a random variable with $X \in \mathcal{X}$.
Let Y be a random variable with $Y \in \mathcal{Y}$.

Entropy: $H(Y) = -\sum_{y \in \mathcal{Y}} P(Y = y) \log_2 P(Y = y)$

Specific Conditional Entropy: $H(Y \mid X = x) = -\sum_{y \in \mathcal{Y}} P(Y = y \mid X = x) \log_2 P(Y = y \mid X = x)$

Conditional Entropy: $H(Y \mid X) = \sum_{x \in \mathcal{X}} P(X = x) H(Y \mid X = x)$

Mutual Information: $I(Y; X) = H(Y) - H(Y|X)$

- **Entropy** measures the **expected # of bits** to code one random draw from X**.**
- For a decision tree, we want to **reduce the entropy of the random variable we are trying to predict**!

Conditional entropy is the expected value of specific conditional entropy
$E_{P(X=x)}[H(Y \mid X = x)]$

Informally, we say that **mutual information** is a measure of the following:
If we know X, how much does this reduce our uncertainty about Y?

Slide credit: CMU MLD Matt Gormley

Splitting with Mutual Information

Which attribute {A, B} would **mutual information** select for the next split?

1) A
2) B
3) A or B (tie)
4) I don't know

Dataset:
Output Y, Attributes A and B

Y	A	B
-	1	0
-	1	0
+	1	0
+	1	0
+	1	1
+	1	1
+	1	1
+	1	1

Decision Tree Learning Example

Entropy: $H(Y) = -\sum_{y \in \mathcal{Y}} P(Y = y) \log_2 P(Y = y)$

Specific Conditional Entropy: $H(Y \mid X = x) = -\sum_{y \in \mathcal{Y}} P(Y = y \mid X = x) \log_2 P(Y = y \mid X = x)$

Conditional Entropy: $H(Y \mid X) = \sum_{x \in \mathcal{X}} P(X = x) H(Y \mid X = x)$

Mutual Information: $I(Y; X) = H(Y) - H(Y|X)$

Y	A	B
-	1	0
-	1	0
+	1	0
+	1	0
+	1	1
+	1	1
+	1	1
+	1	1

Slide credit: CMU MLD Matt Gormley

Decision Tree Learning Example

Entropy: $H(Y) = -\sum_{y \in \mathcal{Y}} P(Y = y) \log_2 P(Y = y)$

Specific Conditional Entropy: $H(Y \mid X = x) = -\sum_{y \in \mathcal{Y}} P(Y = y \mid X = x) \log_2 P(Y = y \mid X = x)$

Conditional Entropy: $H(Y \mid X) = \sum_{x \in \mathcal{X}} P(X = x) H(Y \mid X = x)$

Mutual Information: $I(Y; X) = H(Y) - H(Y|X)$

Y	A	B
-	1	0
-	1	0
+	1	0
+	1	0
+	1	1
+	1	1
+	1	1
+	1	1

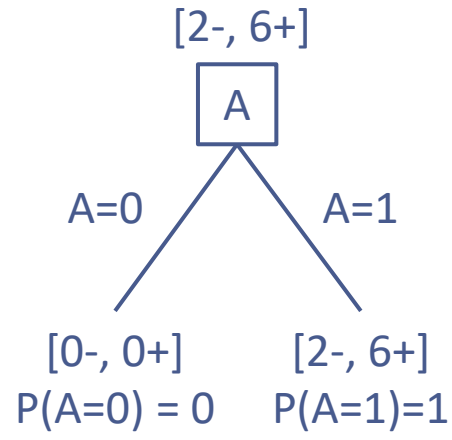
$$H(Y) = -\left[\frac{2}{8}\log_2\frac{2}{8} + \frac{6}{8}\log_2\frac{6}{8}\right]$$

$$H(Y \mid A = 0) = undefined$$

$$H(Y \mid A = 1) = -\left[\frac{2}{8}\log_2\frac{2}{8} + \frac{6}{8}\log_2\frac{6}{8}\right] = H(Y)$$

$$H(Y \mid A) = P(A = 0)H(Y \mid A = 0) + P(A = 1)H(Y \mid A = 1)$$
$$= 0 + H(Y \mid A = 1)$$
$$= H(Y)$$

$$I(Y; A) = H(Y) - H(Y \mid A) = 0$$



Slide credit: CMU MLD Matt Gormley

Decision Tree Learning Example

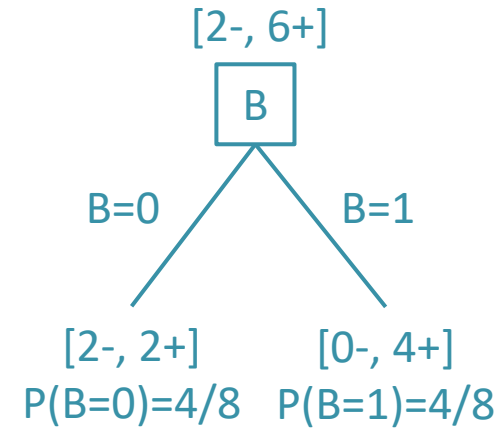
Entropy: $H(Y) = -\sum_{y \in \mathcal{Y}} P(Y = y) \log_2 P(Y = y)$

Specific Conditional Entropy: $H(Y \mid X = x) = -\sum_{y \in \mathcal{Y}} P(Y = y \mid X = x) \log_2 P(Y = y \mid X = x)$

Conditional Entropy: $H(Y \mid X) = \sum_{x \in \mathcal{X}} P(X = x) H(Y \mid X = x)$

Mutual Information: $I(Y; X) = H(Y) - H(Y|X)$

Y	A	B
-	1	0
-	1	0
+	1	0
+	1	0
+	1	1
+	1	1
+	1	1
+	1	1



Slide credit: CMU MLD Matt Gormley

Decision Tree Learning Example

Entropy: $H(Y) = -\sum_{y \in \mathcal{Y}} P(Y = y) \log_2 P(Y = y)$

Specific Conditional Entropy: $H(Y \mid X = x) = -\sum_{y \in \mathcal{Y}} P(Y = y \mid X = x) \log_2 P(Y = y \mid X = x)$

Conditional Entropy: $H(Y \mid X) = \sum_{x \in \mathcal{X}} P(X = x) H(Y \mid X = x)$

Mutual Information: $I(Y; X) = H(Y) - H(Y|X)$

Y	A	B
-	1	0
-	1	0
+	1	0
+	1	0
+	1	1
+	1	1
+	1	1
+	1	1

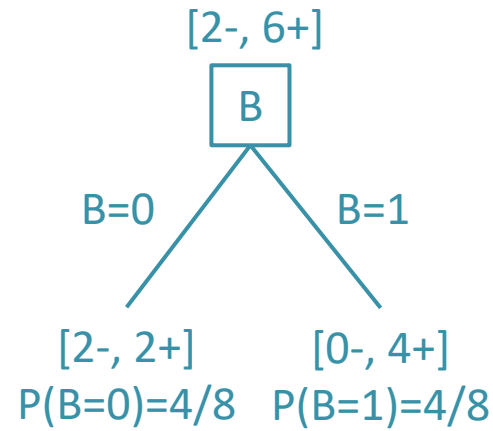
$$H(Y) = -\left[\frac{2}{8}\log_2\frac{2}{8} + \frac{6}{8}\log_2\frac{6}{8}\right]$$

$$H(Y \mid B = 0) = -\left[\frac{2}{4}\log_2\frac{2}{4} + \frac{2}{4}\log_2\frac{2}{4}\right]$$
$$H(Y \mid B = 1) = -[0 \log_2 0 + 1 \log_2 1] = 0$$

$$H(Y \mid B) = P(B = 0)H(Y \mid B = 0) + P(B = 1)H(Y \mid B = 1)$$
$$= \frac{4}{8}H(Y \mid B = 0) + \frac{4}{8} \cdot 0$$

$$I(Y; B) = H(Y) - H(Y \mid B) > 0$$

$I(Y; B)$ ends up being greater than $I(Y; A) = 0$, so we split on B



Slide credit: CMU MLD Matt Gormley

Building a Decision Tree

How do we choose the best feature?

A **splitting criterion** is a function that measures how good or useful splitting on a particular feature is *for a specified dataset*

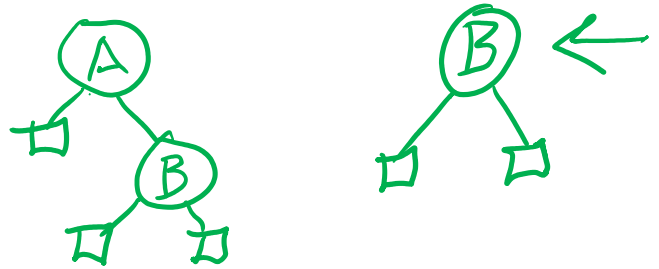
Insight: use the feature that optimizes the splitting criterion current decision

Potential splitting criteria:

- Training error rate (minimize)
- Gini impurity (minimize) → CART algorithm
- Mutual information (maximize) → ID3 algorithm

Slide credit: CMU MLD Henry Chai

Why bother with splitting criteria at all?



Occam's razor: try to find the "simplest" (e.g., smallest decision tree) classifier that explains the training dataset

The **inductive bias** of a machine learning algorithm is the principal by which it generalizes to unseen examples

What is the inductive bias of the ID3 algorithm i.e., decision tree learning with mutual information maximization as the splitting criterion?

- Try to find the smallest tree that achieves zero training error (or as small as poss.) with high M.I. features at the top

Slide credit: CMU MLD Henry Chai

Are decision trees algorithms optimal?

Well, what do we mean by optimal?

$h^* \in H$

Considering all possible decision trees (i.e., trees splitting on one feature per node),

will the ID3 algorithm (each split maximizes mutual information; stopping when mutual information is zero)…

produce the smallest decision tree that has lowest classification training error?

No, they aren't optimal

Decision trees are greedy algorithms, i.e., they make the best local decision without considering longer term possibilities.

- Better trees are possible, but it takes too long to search all combinations

Decision Trees: Pros & Cons

Pros

- Interpretable
- Efficient (computational cost and storage)
- Can be used for classification and regression tasks
- Compatible with categorical and real-valued features

Cons

- Greedy: each split only considers the immediate impact
  - Not guaranteed to find the smallest (fewest number of splits) tree that achieves a training error rate of 0.
- Liable to overfit!

Slide credit: CMU MLD Henry Chai