Recap

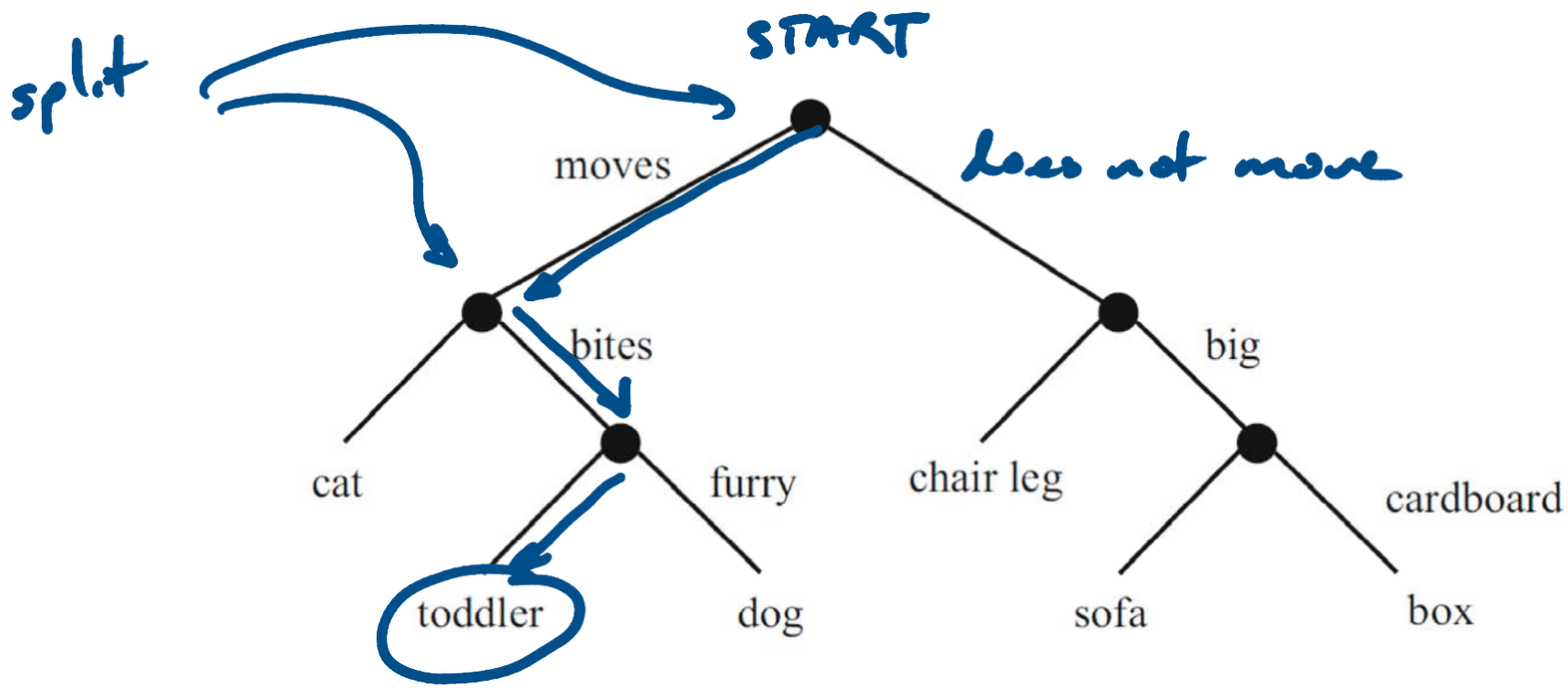
- (Ch 11) Learning to classify
  - Nearest neighbors classifier
  - Naïve Bayes classifier
  - Support vector machine (SVM) classifier

Today

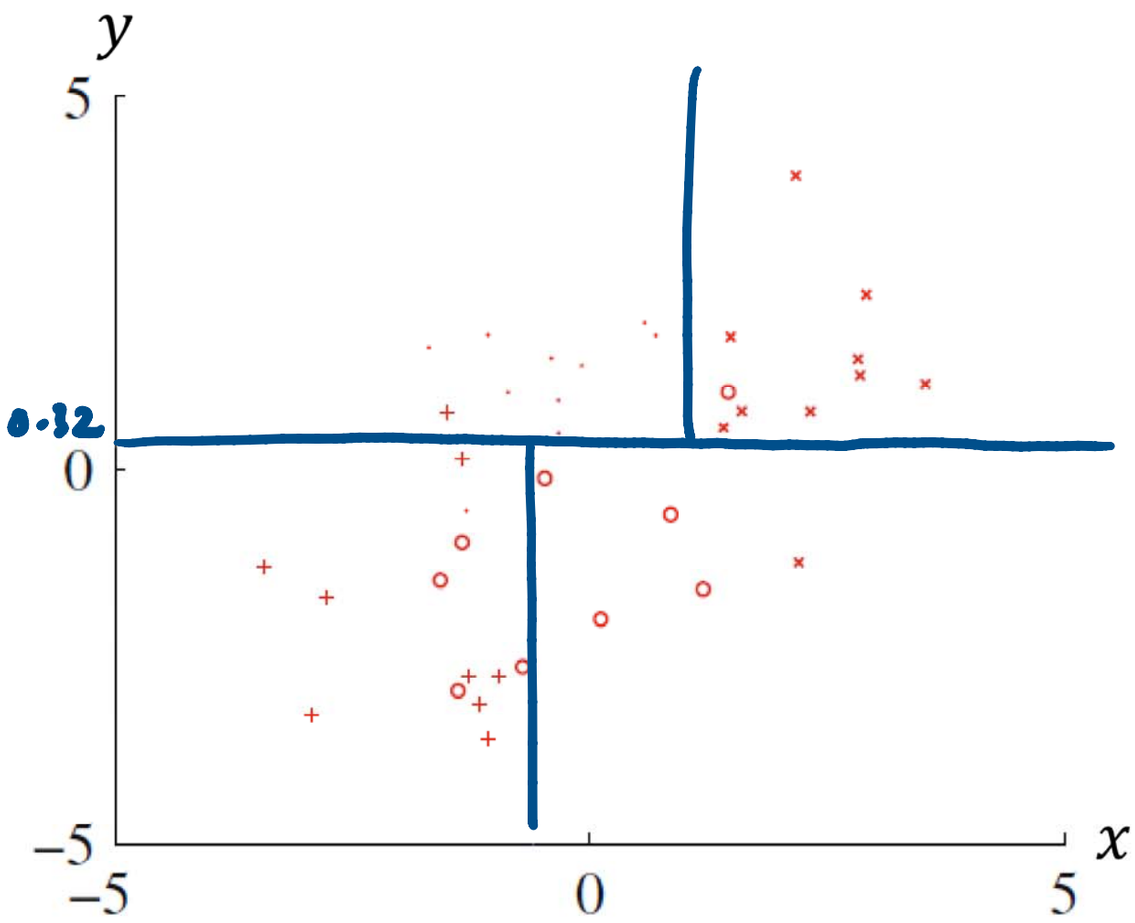
- (Ch 11) Learning to classify
  - Decision trees and random forest classifier
  - Comparing classifiers

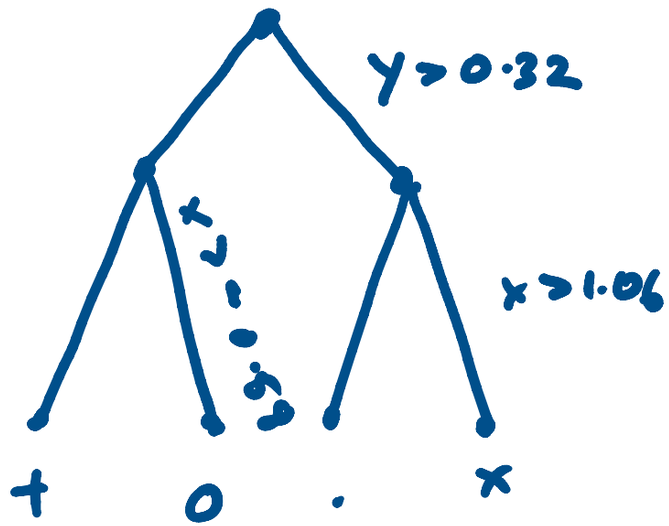
Decision tree: object classification example

This object classification **decision tree** can classify into 6 classes



Training a decision tree: example

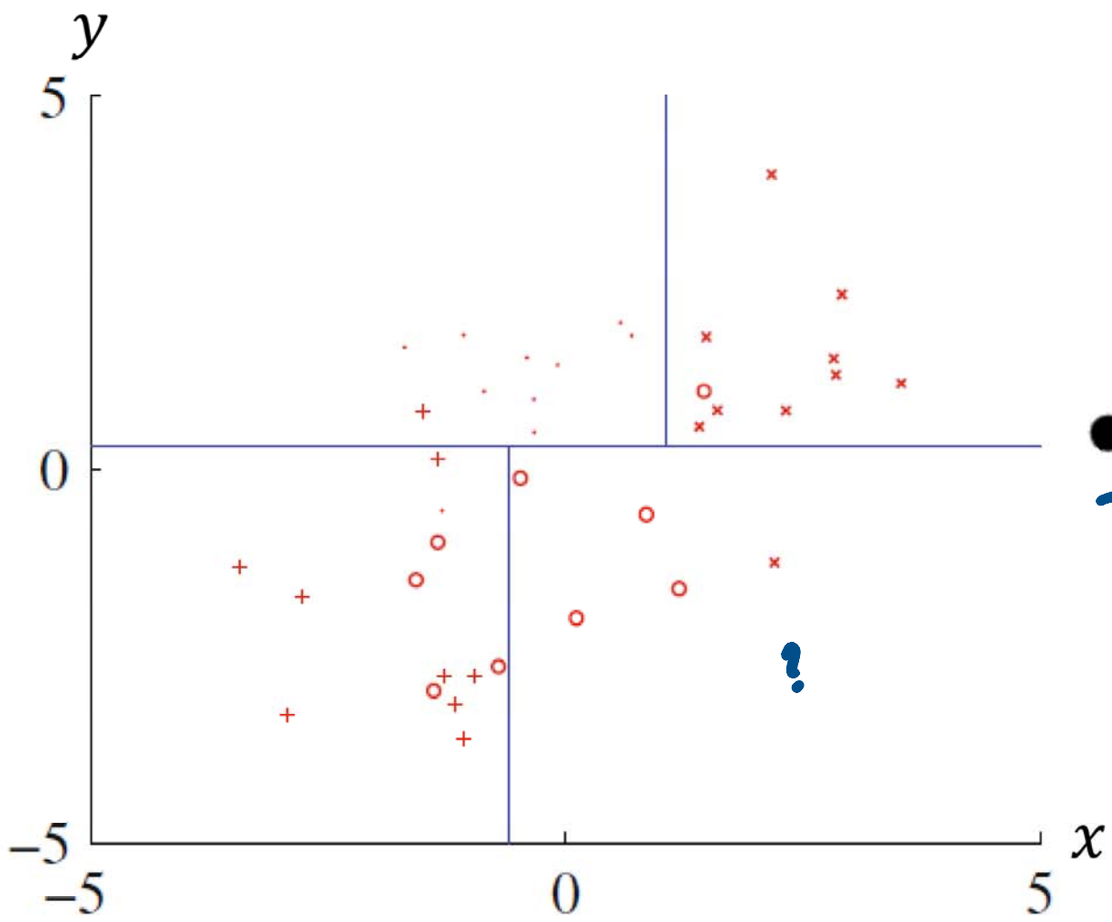




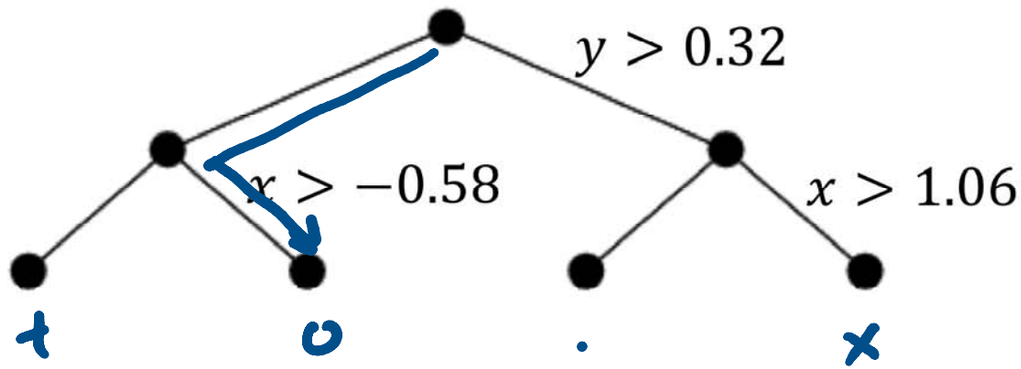
Training a decision tree

- Choose a dimension and a split
- Split the training data D into left- and right-child subsets D_l and D_r
- Repeat the two steps above recursively on each child
- Stop the recursion based on some conditions
- Label the leaves with class labels

Classifying with a decision tree: example



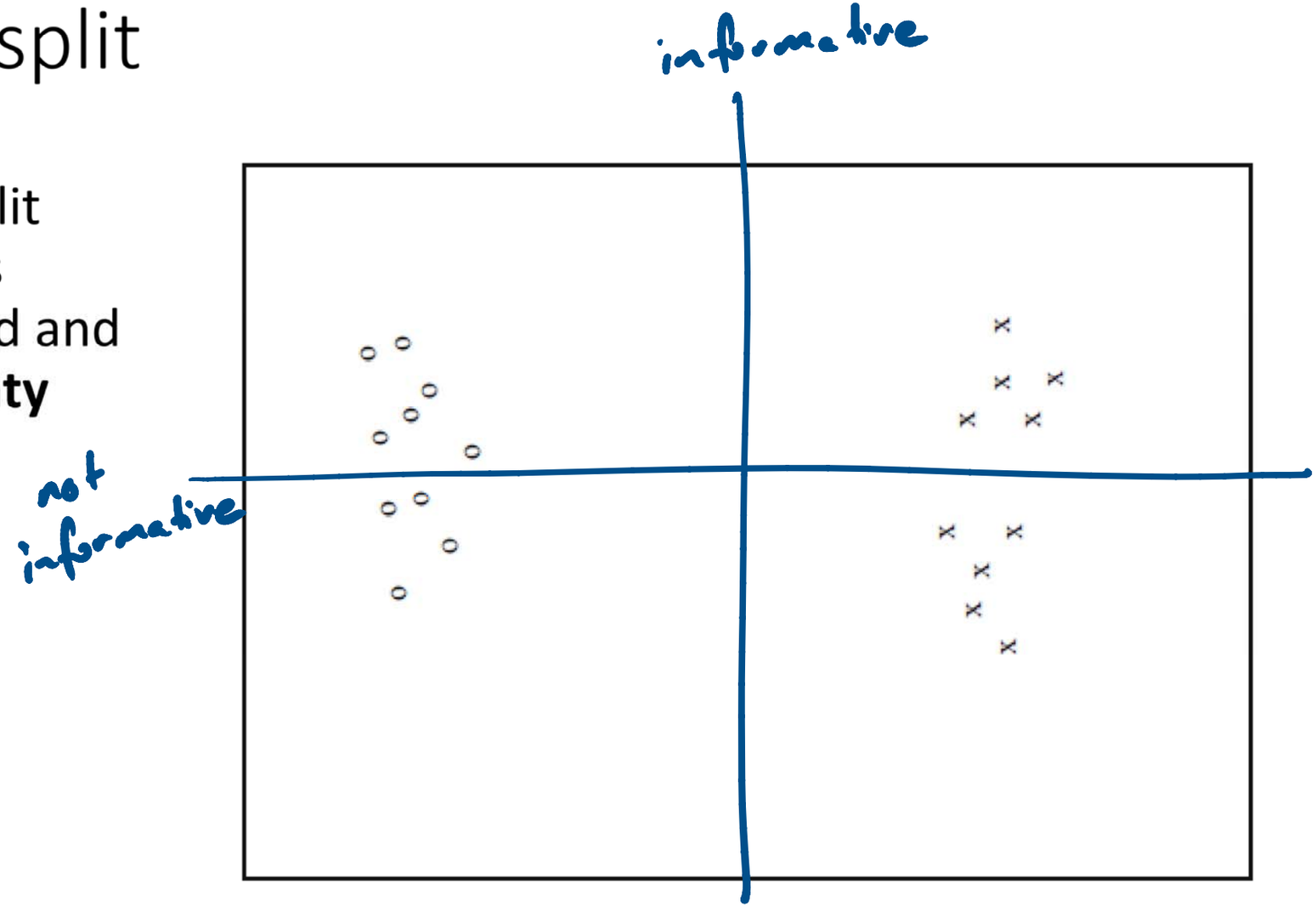
Say ? is at (2, -2)



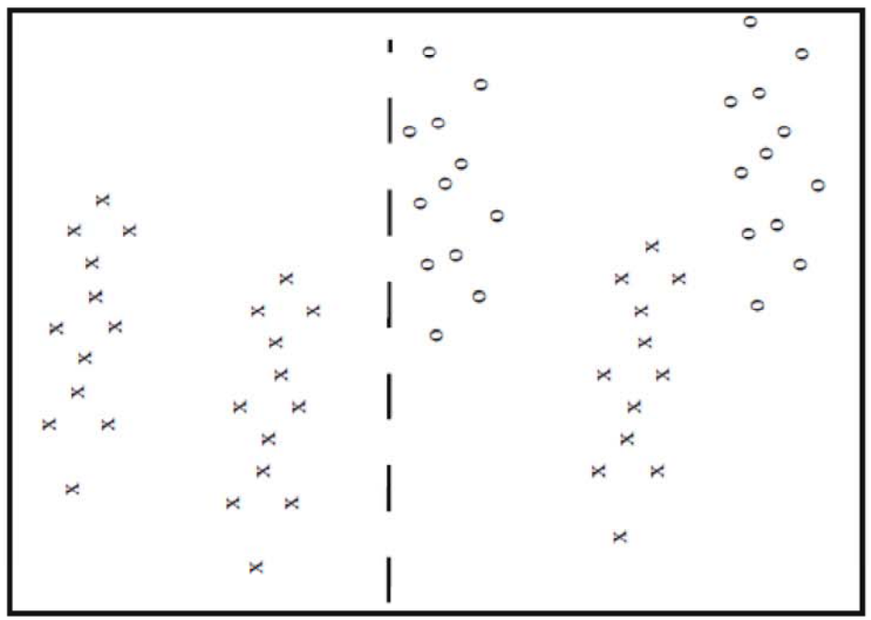
So ? is classified as o

Choosing a split

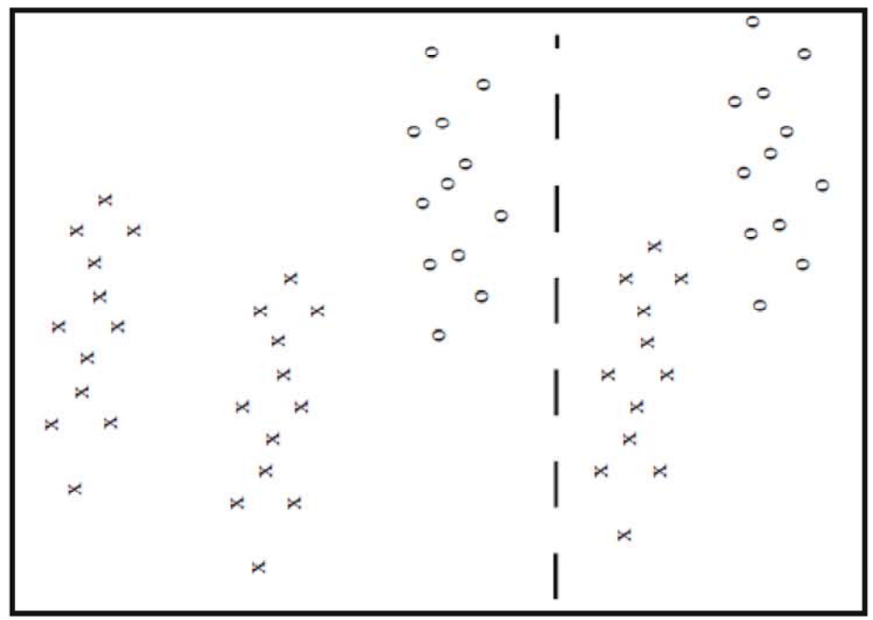
An **informative** split makes the subsets more concentrated and **reduces uncertainty** about class labels



Which split is more informative?



more informative



less informative

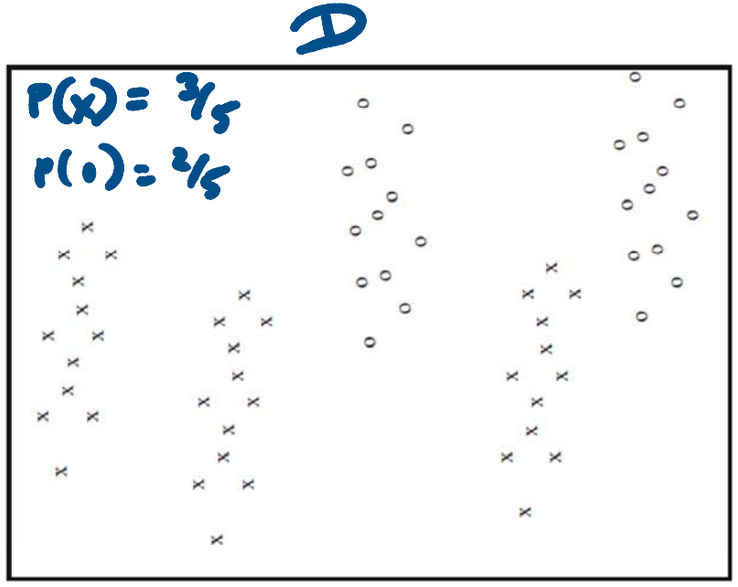
Quantifying uncertainty using entropy

- We can measure uncertainty as the number of bits of information needed to distinguish between classes in a dataset
  - We need $\log_2 2 = 1$ bit to tell apart 2 equally likely classes
  - We need $\log_2 4 = 2$ bits to tell apart 4 equally likely classes

- **Entropy** is the measure of uncertainty for a general distribution
  - If class i contains a fraction $P(i)$ of the data, we need $\log_2 \frac{1}{P(i)}$ bits for that class
  - The entropy $H(D)$ of a dataset D is the weighted mean across all c classes

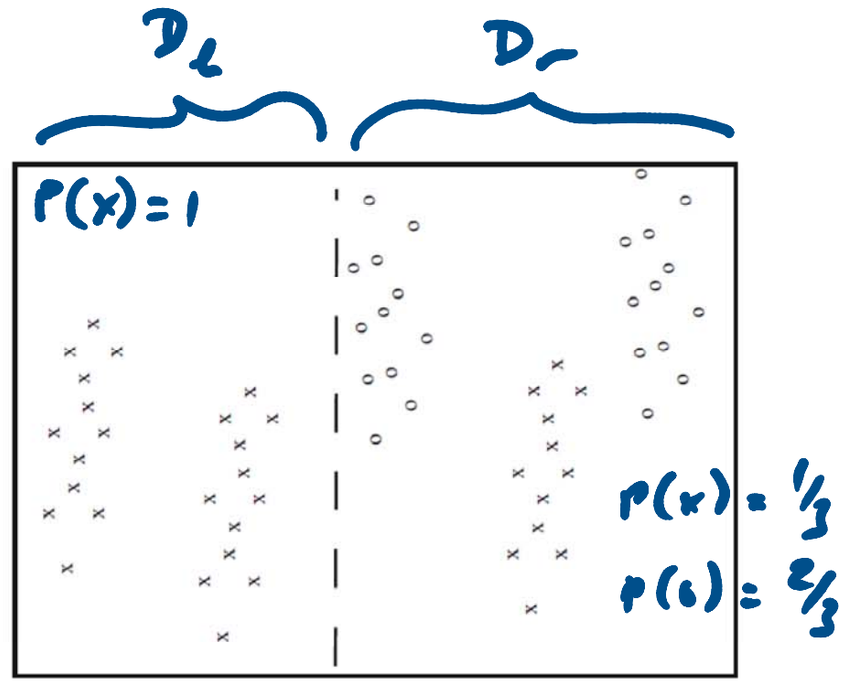
$$H(D) = \sum_{i=1}^{c} P(i) \log_2 \frac{1}{P(i)} \quad \text{bits}$$

Entropy: examples



$$H(D) = \frac{3}{5}\log_2\frac{5}{3} + \frac{2}{5}\log_2\frac{5}{2}$$

$$= 0.971 \text{ bits}$$



$$H(D_L) = 1\log_2 1 = 0 \text{ bits}$$

$$H(D_R) = \frac{1}{3}\log_2 3 + \frac{2}{3}\log_2 3/2$$

$$= 0.918 \text{ bits}$$

Information gain of a split

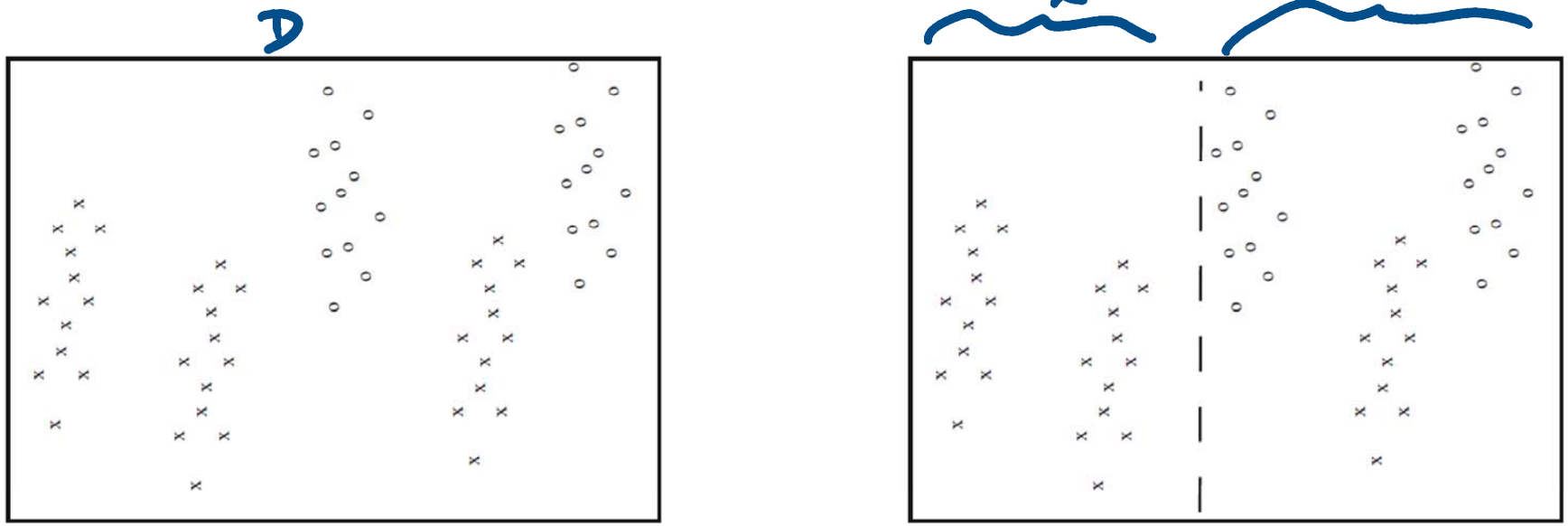
The information gain of a split is the amount it reduces entropy on average

$$I = H(D) - \left(\frac{N_{D_l}}{N_D} H(D_l) + \frac{N_{D_r}}{N_D} H(D_r) \right)$$

where

- N_D is the number of items in dataset D
- N_{D_l} is the number of items in left-child dataset D_l
- N_{D_r} is the number of items in right-child dataset D_r

Information gain: example



$$I = H(D) - \left(\frac{24}{60} H(D_L) + \frac{36}{60} H(D_R) \right) = 0.420 \text{ bits}$$

How to choose a dimension and split

- If there are d dimensions, choose approximately $\sqrt{d}$ of them as candidates at random
- For each candidate, find the split that maximizes information gain
- Choose the best overall dimension and split
- Note that splitting can be generalized to categorical features for which there is no natural ordering of the data

When to stop growing the decision tree

- Growing the tree too deep can lead to overfitting to the training data
- Stop recursion on a data subset if any of the following occurs
  - All items in the data subset are in the same class
  - The data subset becomes smaller than a predetermined size
  - A predetermined maximum tree depth has been reached

How to label the leaves of the decision tree

- A leaf will usually have a data subset containing many class labels
- Choose the class that has the most items in the subset
- Alternatively, label the leaf with the number of items it contains in each class for a probabilistic “soft” classification

From decision trees to random forests (RF)

- Decision trees have some drawbacks
  - May not perform well on training data because of simplistic random training
  - May not perform well on test data because of overfitting

- A random forest is a randomly generated **ensemble** of decision trees that avoids both of the above problems by merging the classifications of the individual trees

Training, evaluation and classification

- Build the random forest by training each decision tree on a new bootstrap replicate of the training data (called a bag)
- Evaluate the random forest by evaluating each decision tree on its out-of-bag items
- Classify by merging the classifications of individual decision trees
  - By simple vote
  - Or by adding soft classifications together and then taking a vote

Considerations in choosing a classifier

- When solving a classification problem, it's usually a good idea to try several techniques

- Here are some criteria with corresponding strong classifiers
  - Accuracy (SVM, random forests)
  - Training speed (nearest neighbors, naïve Bayes)
  - Classification speed (naïve Bayes, SVM)
  - Performance with small training dataset (naïve Bayes)
  - Interpretability (nearest neighbors, naïve Bayes, SVM)