SME3006 Machine Learning – 2025 Fall

Decision trees and ensemble methods

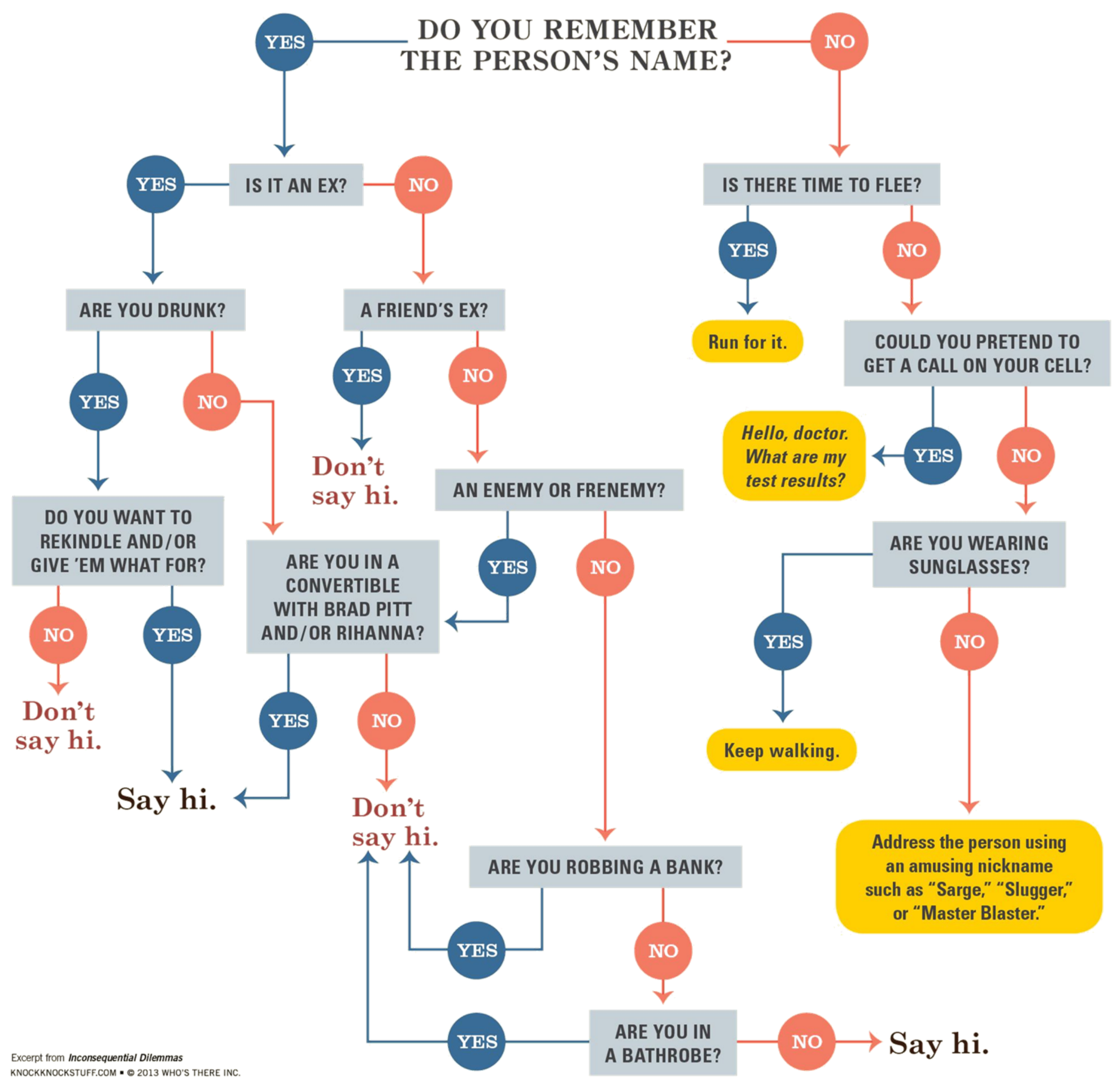
INHA UNIVERSITY

Overview

- Why decision trees?
- limitation of simple models
- from single tree to ensemble
- bagging and random forest
- boosting methods

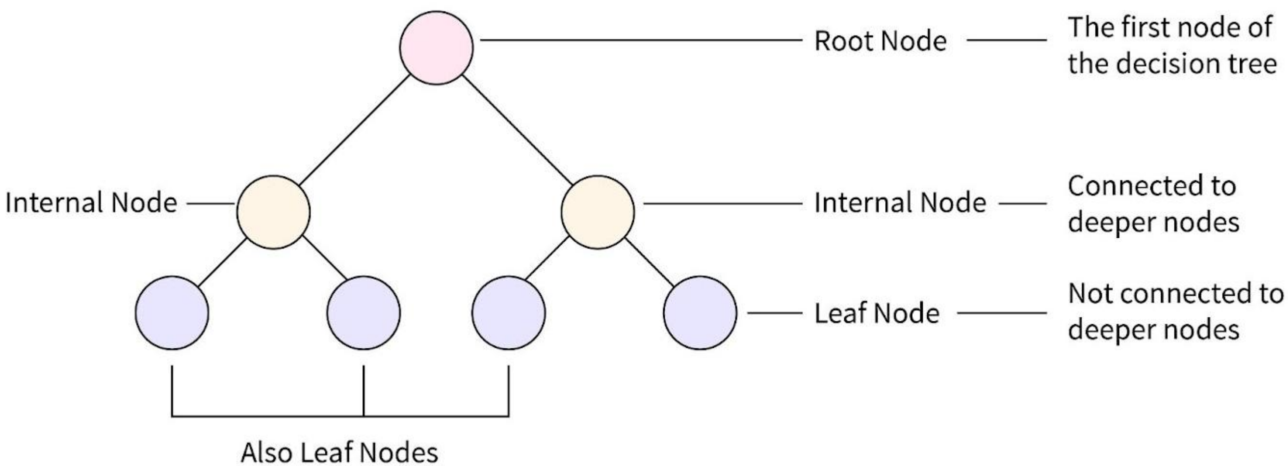
Decision tree

> Do I say hi?



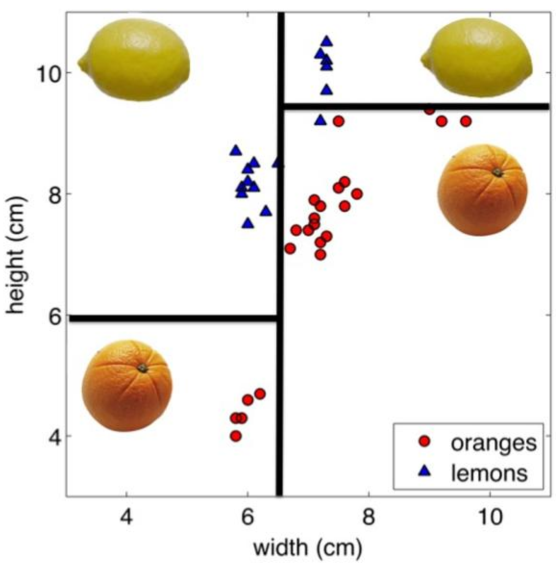
Decision tree

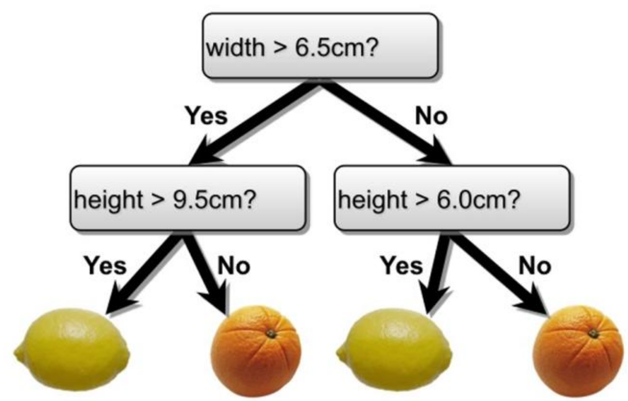
> A decision tree (DT) is a tree that
  - at each inner node has a decision rule that assigns instances uniquely child nodes of the actual node
  - at each leaf node has a class label



Decision tree

- > DT make predictions by recursively splitting on different attributes according to a tree structure
  - decision boundaries are rectangular
- > Example) classifying a lemon and an orange



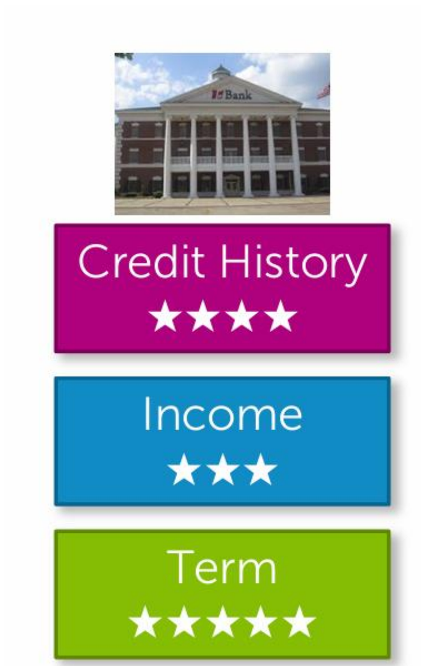


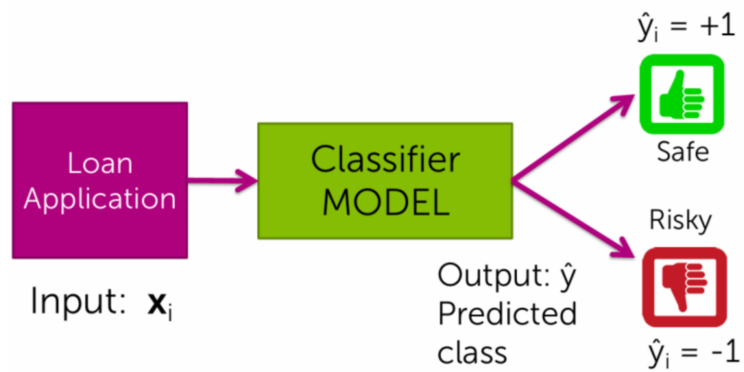
Decision tree

- How do we construct a useful decision tree?
  - learning the simplest (smallest) decision tree is an NP complete problem
  - number of possible trees increases exponentially
- Resort to a greedy heuristic:
  - start from an empty decision tree
  - split on the best attribute
  - recurse
- Which attribute is the best?

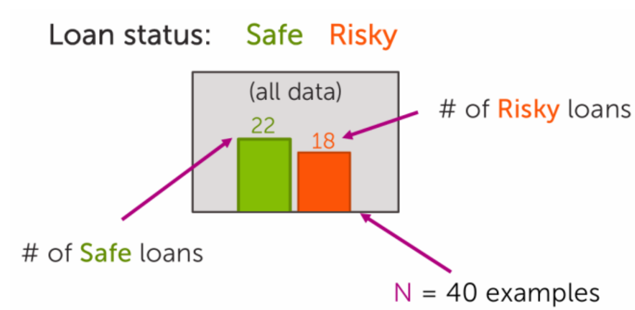
Decision tree

> Example) buying a new house





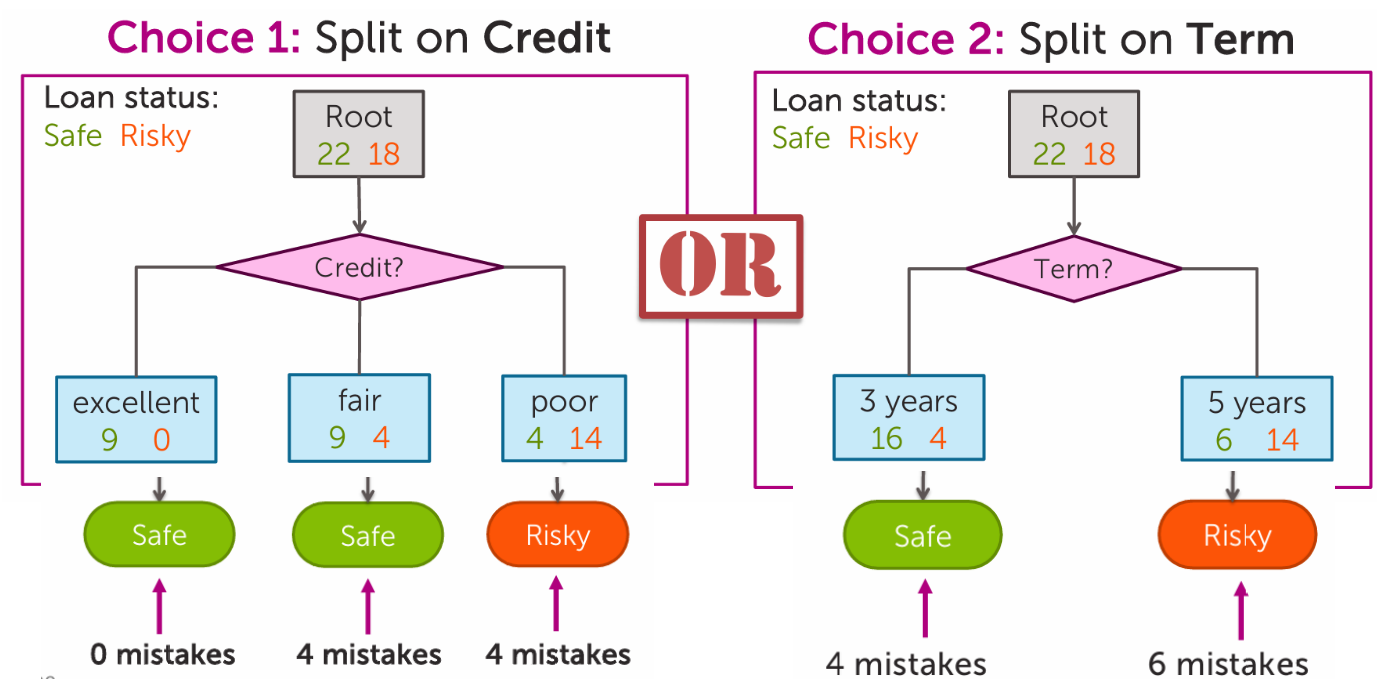
Credit	Term	Income	y
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe



Decision tree

- Quality metric:
  - classification error $Err = \frac{\#\ incorrect\ predictions}{\#\ examples} \in [0, 1]$

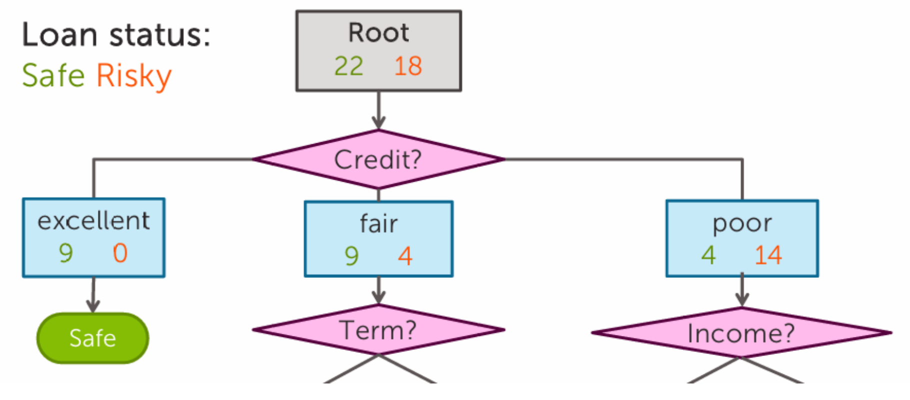
- What is the best feature to split on



Decision tree

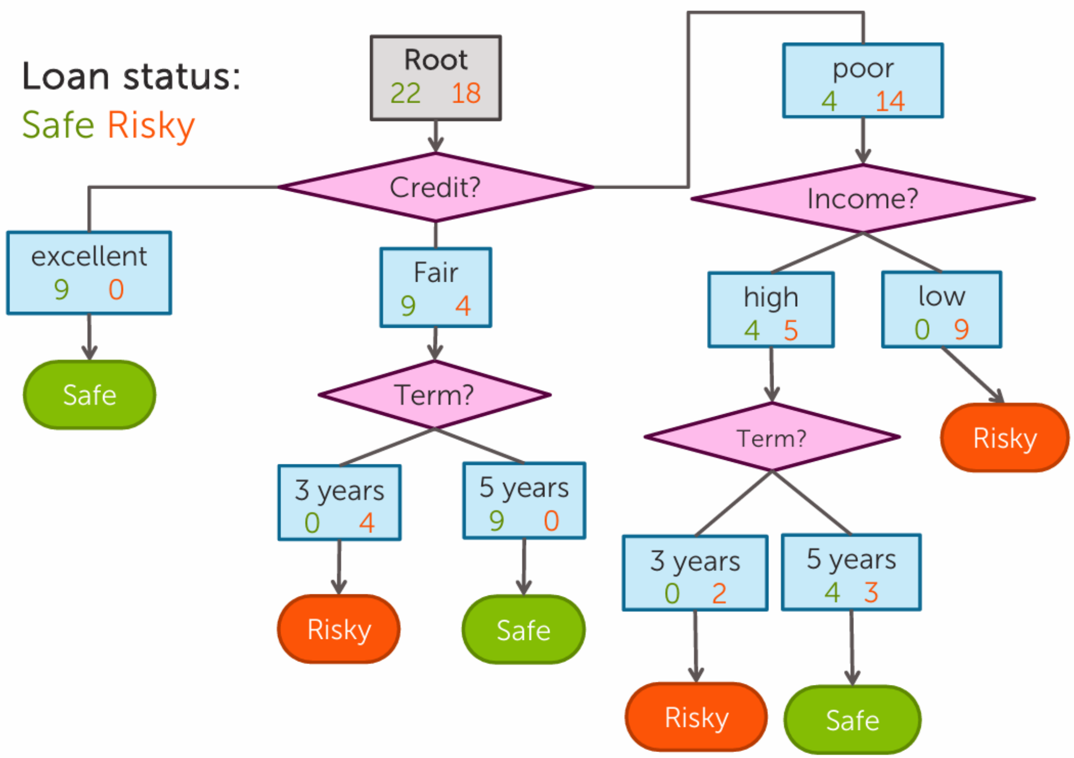
> Feature split selection algorithm
  - given a subset of data M
  - For each feature $h_i(x)$:
    - split data of M according to feature $h_i(x)$
    - compute classification error of split
  - Choose feature $h^*(x)$ with lowest classification error

> Recursion



Decision tree

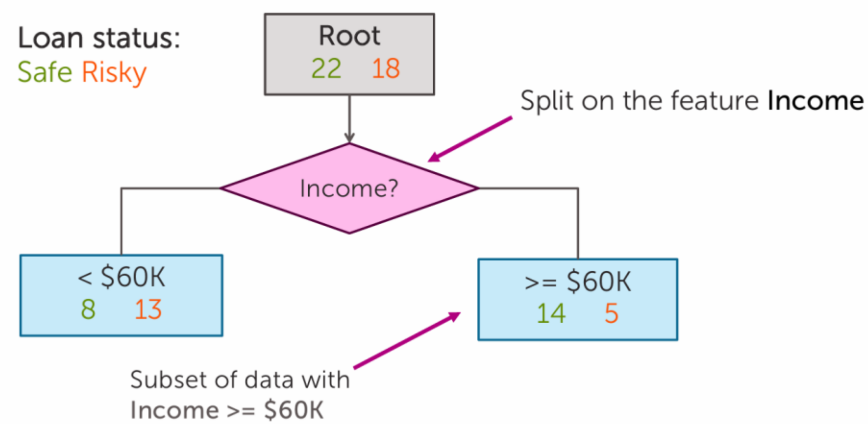
> When does the recursion stop?
  - 1) all data agrees on *output*
  - 2) already split on all features
  - 3) if no split reduces the classification error



Decision tree

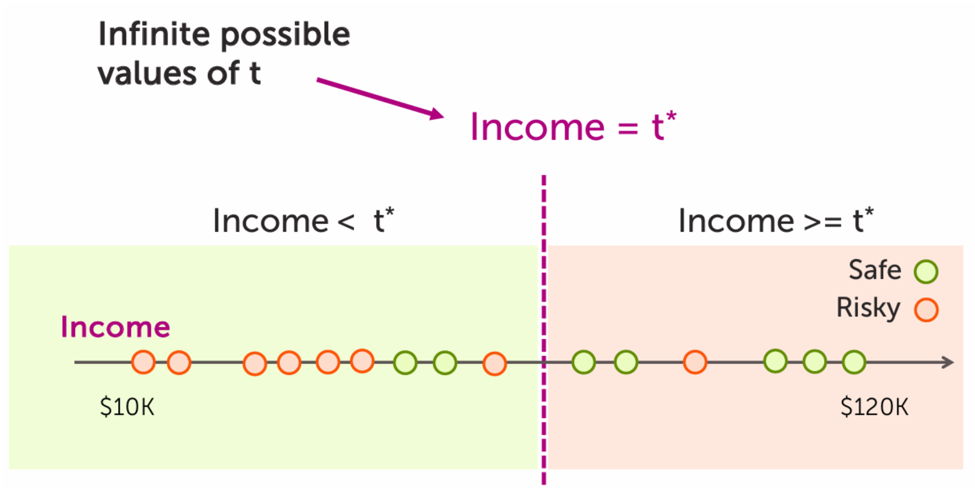
> What if we have continuous values?
  - threshold split

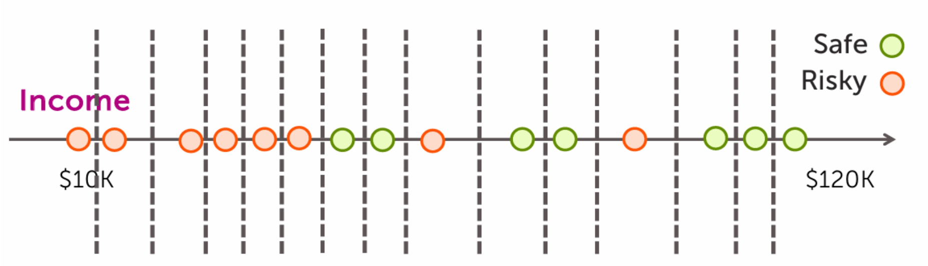
Income	Credit	Term	y
$105 K	excellent	3 yrs	Safe
$112 K	good	5 yrs	Risky
$73 K	fair	3 yrs	Safe
$69 K	excellent	5 yrs	Safe
$217 K	excellent	3 yrs	Risky
$120 K	good	5 yrs	Safe
$64 K	fair	3 yrs	Risky
$340 K	excellent	5 yrs	Safe
$60 K	good	3 yrs	Risky



Decision tree

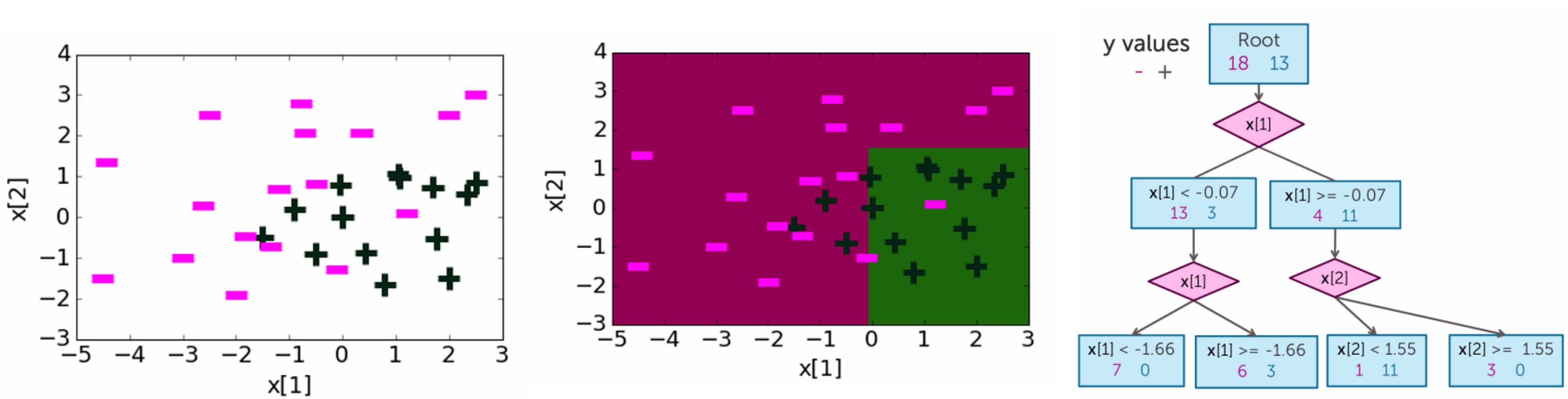
> What if we have continuous values?
  - threshold split

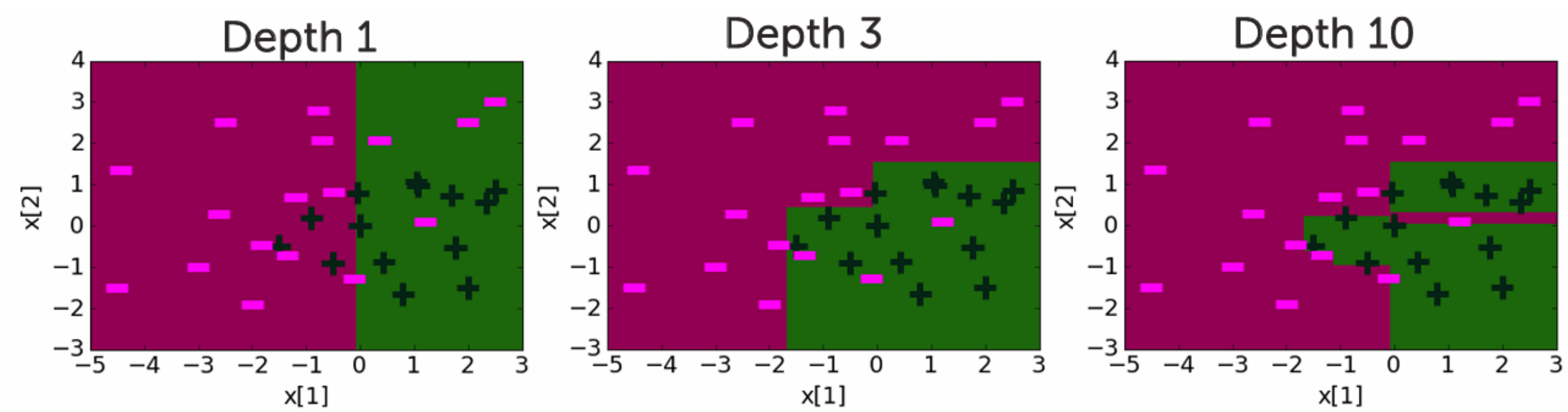




Decision tree

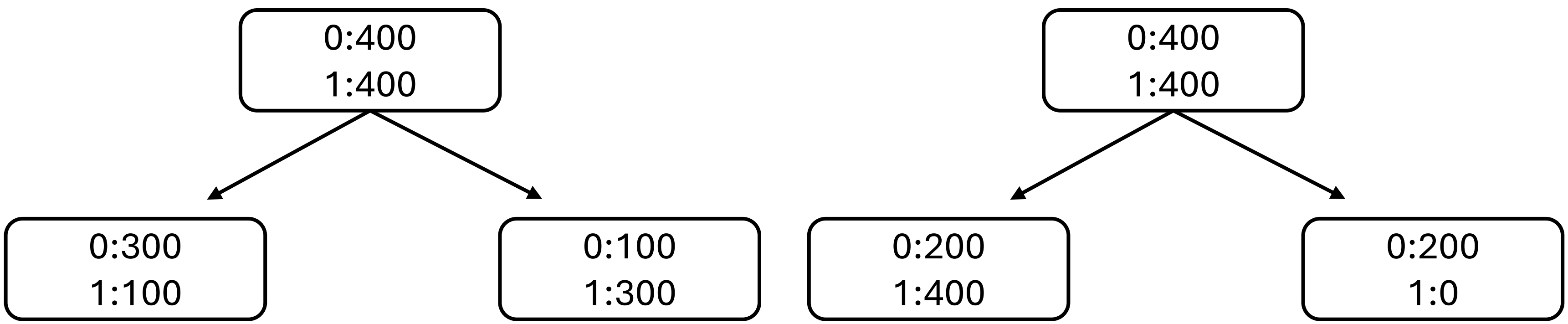
> Decision boundaries
  - for threshold splits, same feature can be used multiple times





Decision tree

- It is possible to use classification error as quality criterion, it usually is not a good idea
- Imagine a dataset with a binary target variable (0/1)
  - we have 400 cases per each(400 / 400)
  - assume the following split



  - both have 200 errors (same classification error)
  - but the right split is preferred as it contains a pure node

Information theory

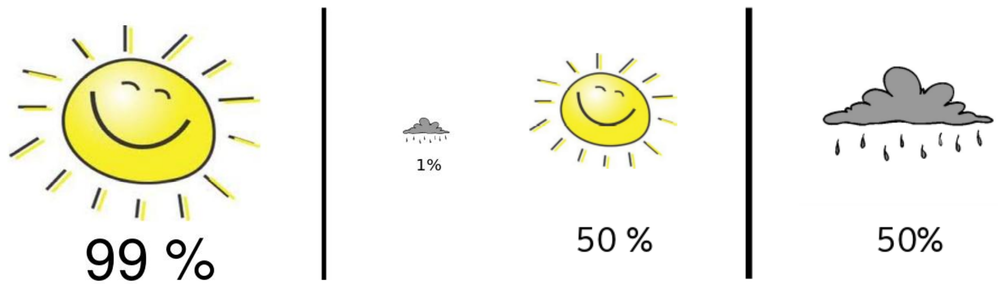
- We will use techniques from information theory
  - define probability distributions to measure uncertainty

- Which feature (attribute) is better to split on
  - deterministic (all are true or false) → good
  - uniform distribution (all classes in leaf equally probable) → bad
  - what about distributions in between?

Information theory

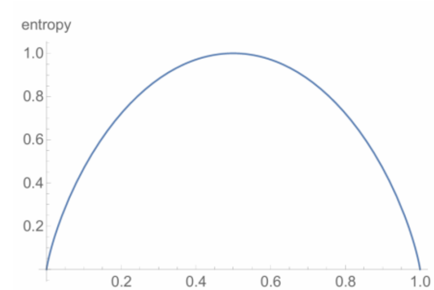
> Entropy

- weather forecasts



- which of the two forecasts would give you the most information?
  - first one: you (certainly) know tomorrow’s weather
  - second one: you have no idea (the highest uncertainty)

- entropy measures the lack of information or uncertainty
  - maximal if the random distribution is uniform
  - minimal if the random distribution is deterministic



Information theory

> Entropy measures the lack of information or uncertainty

$$H(X) = -\sum_{x \in X} p(x) \log_2 p(x)$$

- Why? (Shannon uniqueness theorem, math proof)
  - continuity: small changes in probabilities should cause small changes in H
  - maximal for uniform distribution: when all outcomes are equally likely, uncertainty should be largest
  - additivity for independent events: the total uncertainty of two independent processes should be the sum of their individual uncertainties
    $H(X,Y) = H(X) + H(Y) \quad if\ X\ and\ Y\ are\ independent$

- high entropy → outcome is very unpredictable → you need information
- zero entropy → outcome is fully predictable → no need information

Information theory

> Entropy of a joint distribution

$$H(X,Y) = -\sum_{x,y} p(x,y) \log_2 p(x,y)$$

- we have two random variables and measures the total uncertainty

- weather forecasts
  - predicting Seoul and New York's weather (assume they are independent)
    → need information for both Seoul and New York, $H(x,y) = H(x) + H(y)$

  - predicting Seoul and Incheon's weather (they are somewhat dependent)
    → need information for Seoul and rest for Incheon, $H(x,y) = H(x) + H(y|x)$

Information theory

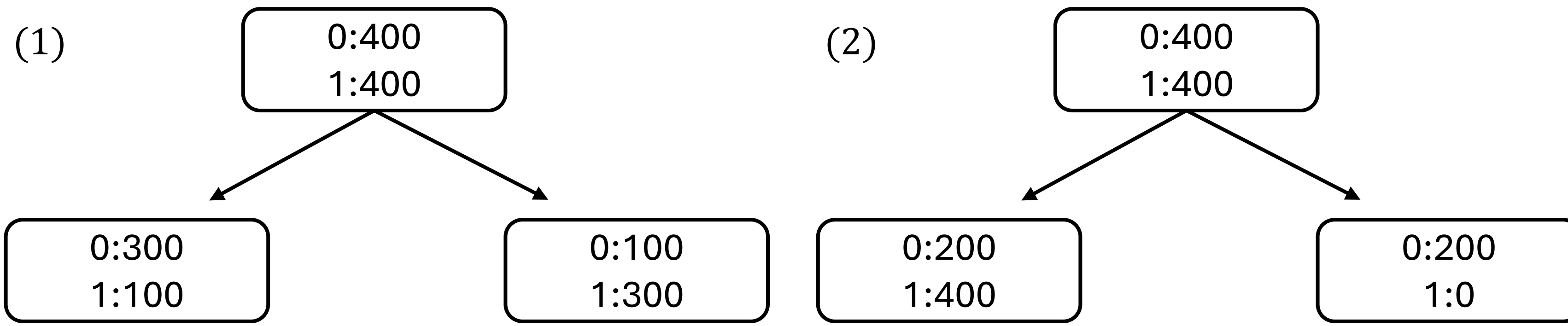
- Useful properties:
  - H is always non-negative
  - chain rule: $H(X,Y) = H(X|Y) + H(Y) = H(Y|X) + H(X)$
  - If X and Y independent, X doesn't tell us anything about Y: $H(Y|X) = H(Y)$
  - but y tells us everything about y: $H(Y|Y) = 0$
  - by knowing x, we can only decrease about y: $H(Y|X) \leq H(Y)$

- Information gain
  - how much information do we get by discovering x
  - $IG(Y|X) = H(Y) - H(Y|X)$
  - this is called the information gain or the mutual information
    - x is completely uninformative about y: $IG(Y|X) = 0$
    - x is completely informative about y: $IG(Y|X) = H(Y)$

Information theory

$$H(X) = -\sum_{x \in X} p(x) \log_2 p(x)$$

> Information gain measures the informativeness of a variable
  - what is the information gain of the splits?



- Root entropy: $H(Y) = -\frac{400}{800} \log_2 \left(\frac{400}{800}\right) \times 2 = 1$
- Leaf entropy (1): $H\left(Y|X_{(1)}\right) = \frac{1}{2} \times \left(-\frac{300}{400} \log_2 \frac{300}{400} - \frac{100}{400} \log_2 \frac{100}{400}\right) \times 2$
- Leaf entropy (2): $H\left(Y|X_{(2)}\right) = \frac{3}{4} \times \left(-\frac{400}{600} \log_2 \frac{400}{600} - \frac{200}{600} \log_2 \frac{200}{600}\right) - \frac{1}{4} \times (\frac{200}{200} \log_2 \frac{200}{200})$
- Information gain: $IG_{(1)} = H(y) - H\left(y\middle|x_{(1)}\right) = 0.1887$
  $IG_{(2)} = H(y) - H\left(y\middle|x_{(2)}\right) = 0.3113$

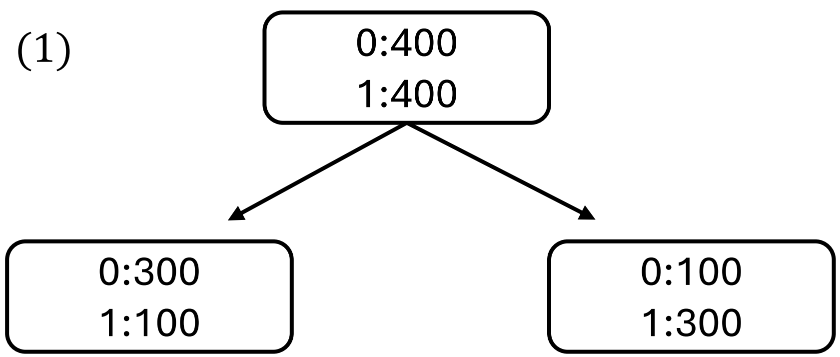
Information theory

> Quadratic entropy (Gini impurity)

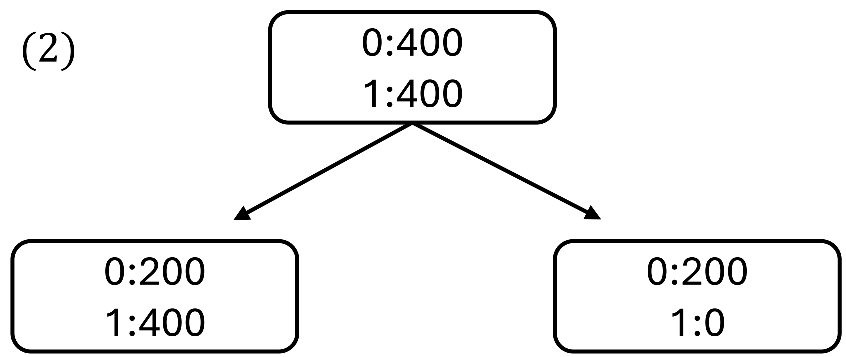
$$H(X) = \sum_{x \in X} p(x)\big(1 - p(x)\big) = 1 - \sum_{x \in X} p(x)^2$$

- measure probability of misclassification
- only multiplications and additions (no logarithm)

(1)



(2)



Gini-Impurity

$$= \frac{1}{2}((\frac{3}{4})^2 + (\frac{1}{4})^2) + \frac{1}{2}((\frac{3}{4})^2 + (\frac{1}{4})^2)$$
$$= 0.625$$

Gini-Impurity

$$= \frac{3}{4}((\frac{1}{3})^2 + (\frac{2}{3})^2) + \frac{1}{4}(1^2 + 0^2)$$
$$\approx 0.667$$

Decision tree

- Concept: split data to reduce uncertainty about the target variable
  - what makes a good split? → good criteria
  - which split strategy to use (multiway vs. binary)
  - how to stop splitting? (pruning needed to avoid overfitting)
  - how to deal with incomplete / missing data?

- Algorithms
  - ID3: multiway split, Information gain
  - C4.5: Gain ratio (fixed IG bias), continuous features, pruning, missing value handling
  - CART: binary splits only, Gini impurity for classification, variance reduction for regression, cost-complexity pruning, surrogate splits

Ensemble methods

> Tree-based models
  - interpretable
  - can capture non-linear relationships
  - don't require scaling of the data
  - but single decision trees are likely to overfit

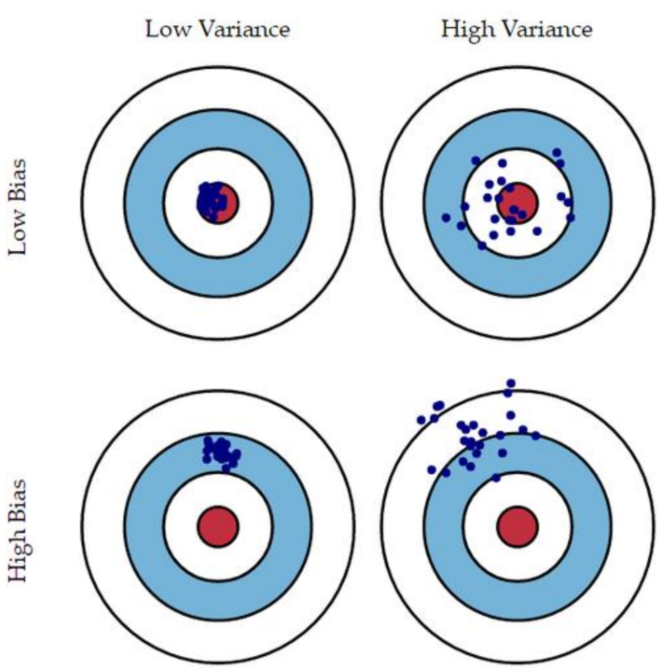
> Ensembles
  - key idea: combine multiple machine learning models to create more powerful methods
  - can be applied to almost any learning algorithms
  - particularly well suited to decision trees

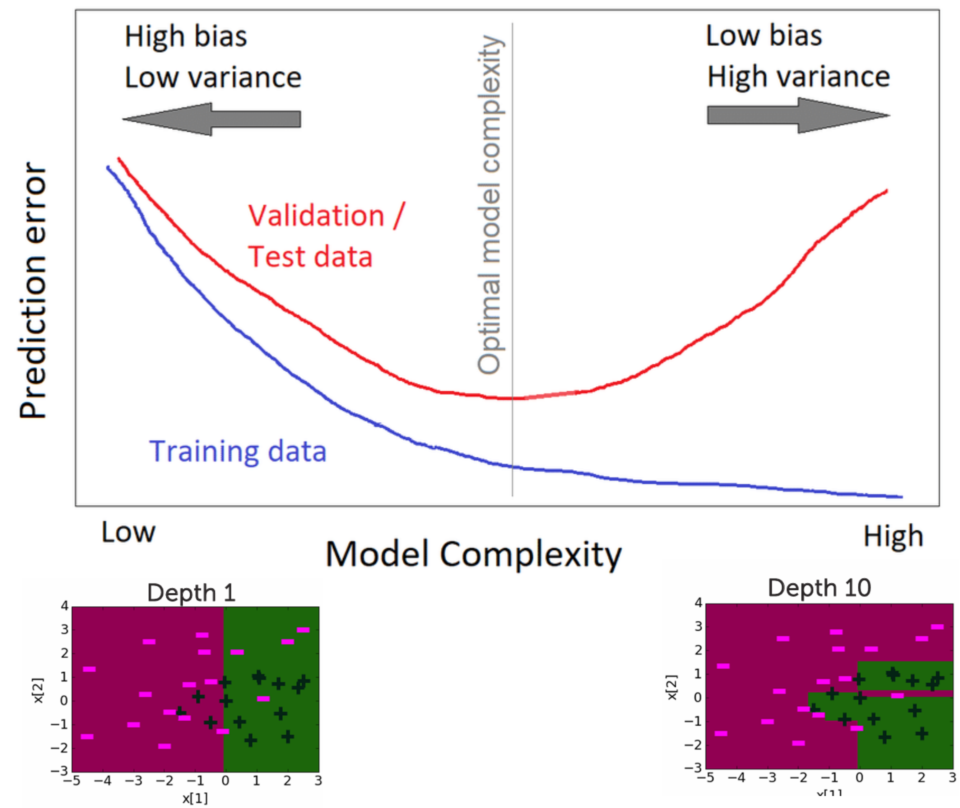
Ensemble methods

- An ensemble of predictors is a set of predictors whose individual decisions are combined in some way to predict new examples
  - E.g., (weighted) majority vote
- For this to be nontrivial, the learned hypotheses must differ somehow
  - different algorithm
  - different choice of hyperparameters
  - trained on different data
  - trained with different weight of the training examples
- Bagging: train classifiers independently on random subsets of data
- Boosting: train classifiers sequentially, focusing on examples that the previous ones got wrong

Ensemble methods

> Bias and variance
  - one way to reduce model variance is averaging





Bagging

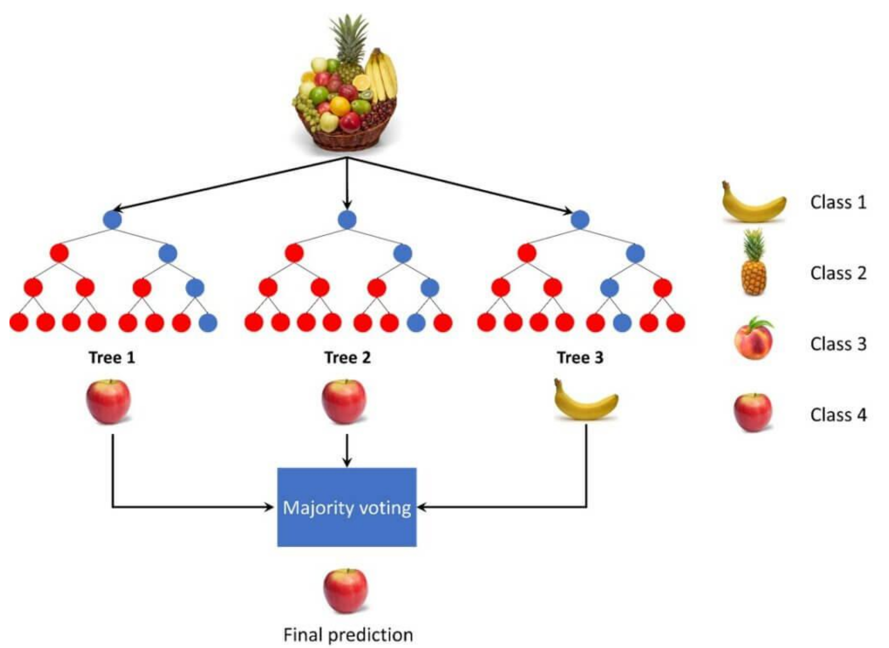
- Suppose we could somehow sample m independent training sets
  - we could then compute the prediction y_i based on each one
  - bias: unchanged, $\mathbb{E}[y] = \mathbb{E}\left[\frac{1}{m}\sum_i y_i\right] = \mathbb{E}[y_i]$
  - variance: reduced, $Var[y] = Var\left[\frac{1}{m}\sum_i y_i\right] = \frac{1}{m^2}\sum_i Var[y_i] = \frac{1}{m}Var[y_i]$

- In practice, we don't have access to the data generating distribution
  - solution: bootstrap aggregation or bagging

  - from a single dataset $\mathcal{D}$, generating m new datasets, each by sampling n training examples from $\mathcal{D}$
  - average the predictions of models trained on each of these datasets

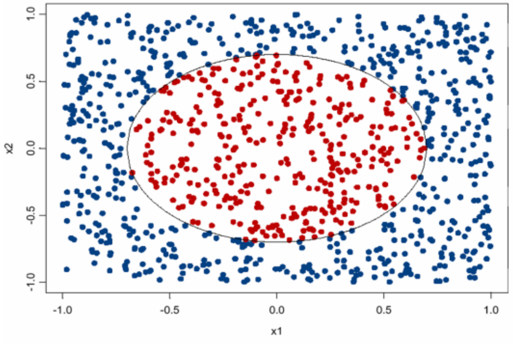
Bagging

- Random forests = bagged decision trees, with one extra trick to decorrelate the predictions
  - when choosing each node of the decision tree, choose a random set of d input features, and only consider splits on those features
  - random forests work well – one of the most widely used algorithms
  - final prediction: average (regression) or majority vote (classification)

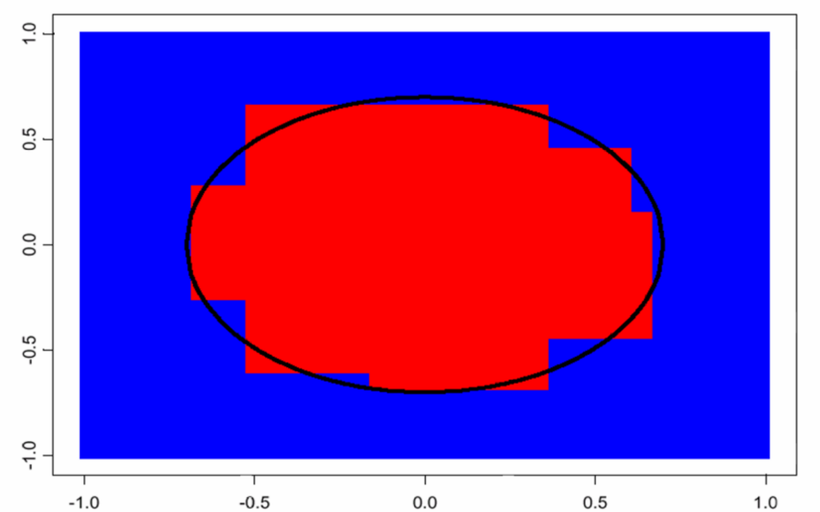


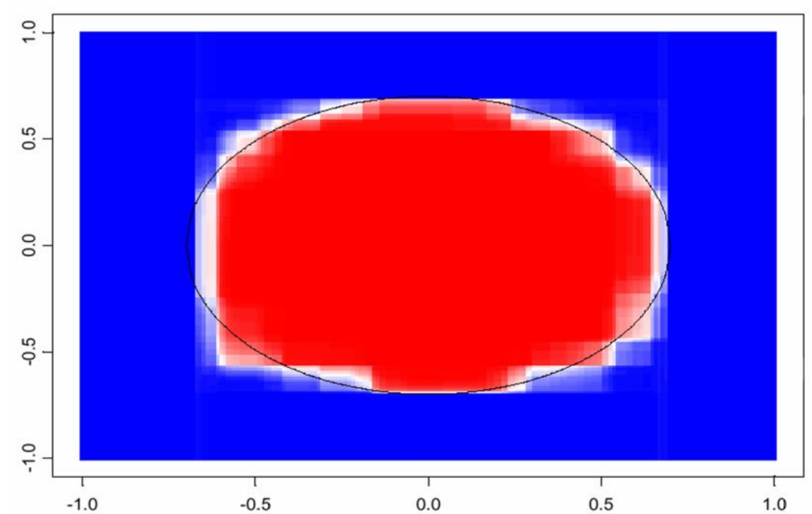
Bagging

> Example: classifying points as being inside a circle



CART decision boundary vs. 100 bagged trees





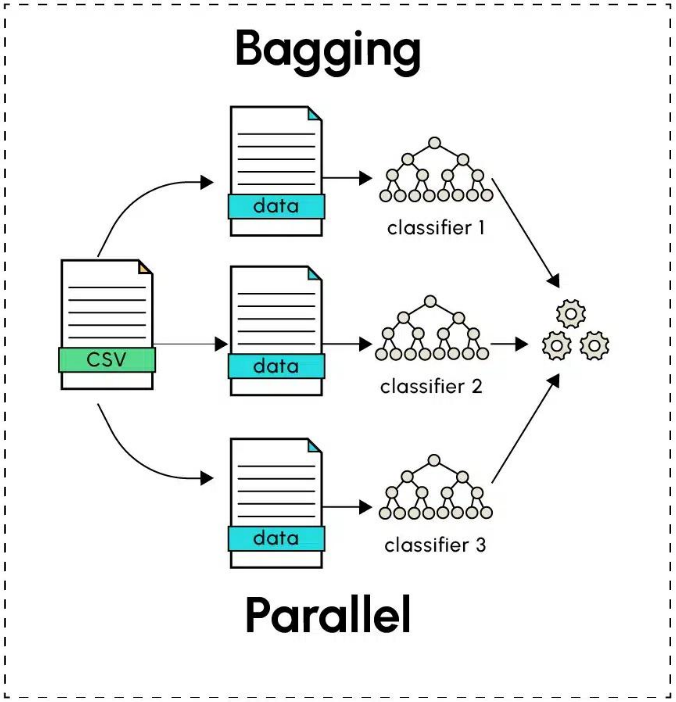
Bagging

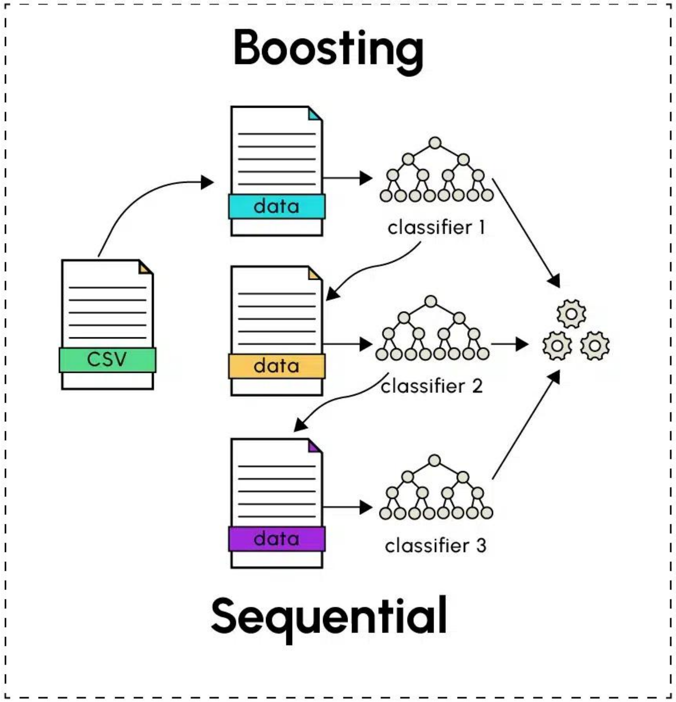
> Random forest reduces overfitting by averaging predictions
  - each tree overfits on some part of the data, but we can reduce overfitting by averaging the results (can be shown mathematically)
  - work well without heavy tuning of hyperparameters
  - even if a single model is great, a small ensemble usually helps

> Limitations
  - require more memory
  - hard to interpret
  - does not reduce bias
  - there is still correlation between classifiers

Boosting

> Weak learners are trained sequentially, each one focusing more on the mistakes of the previous ones
  - boosting is strong at reducing bias (use simple, weak learners)
  - can a set of weak learners to be combined to create a stronger learner?

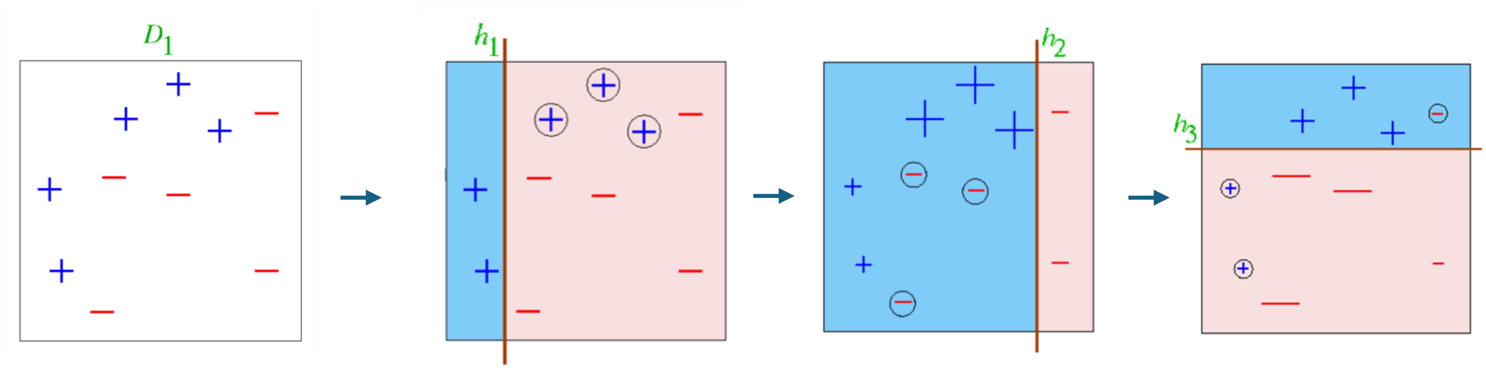




Boosting

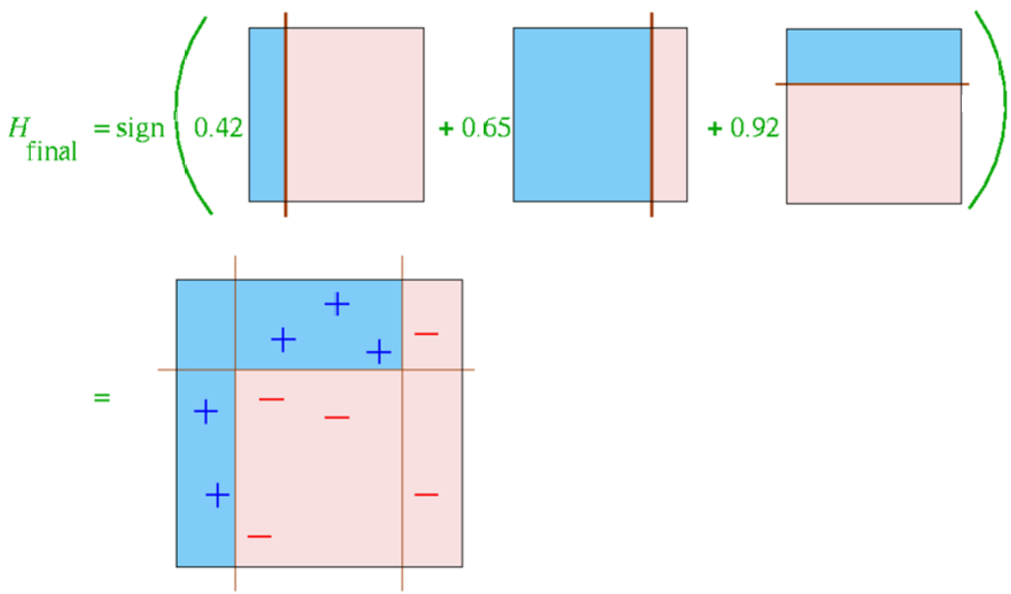
- Classifier tries harder on examples with higher cost
  - misclassification error $\sum_i \mathbb{I}\left[h\left(x^i\right) \neq t^i\right]$
  - weighted error $\sum_i w^i \; \mathbb{I}\left[h\left(x^i\right) \neq t^i\right]$

  - Iterate
    - at each iteration we re-weight the training samples by assigning larger weights to samples that were classified incorrectly
    - we train a new weak classifier and add to the ensemble



Boosting

- Final classifier
  - weighted sum of each weak learners
  - training error converges to zero
    - if each weak learner must have error rate under 0.5 on the weighted data



Boosting

- It is quite resilient to overfitting, though it can overfit

- Types
  - AdaBoost (1996): assigns weights to data points and trains new models on updated weights
  - Gradient boosting (1999): fits new model to the residual errors of previous models
  - Stochastic gradient boosting (2002): uses random subsets of training data and features for each new model
  - XGBoost (2016): optimized gradient boosting with regularization and parallelization
  - LightBGM (2017): gradient boosting with leaf-wise growth and histogram-based splits

Ensemble methods

- Ensembles combine models to improve performance
- Bagging
  - reduce variance (large ensemble can't cause overfitting)
  - bias is not changed (much)
  - parallel
  - need to minimize correlation between ensemble elements
- Boosting
  - reduce bias
  - increase variance (large ensemble can cause overfitting)
  - sequential
  - high dependency between ensemble elements

Reference

- Decision trees
  - https://cs229.stanford.edu/notes2022fall/decision-trees.pdf
  - https://people.csail.mit.edu/dsontag/courses/ml16/slides/lecture11.pdf
  - https://www.ismll.uni-hildesheim.de/lehre/ml-09w/script/ml-04-decisiontrees-2up.pdf
  - https://www.cs.toronto.edu/~mren/teach/csc411_19s/lec/lec03.pdf
- Ensemble methods
  - https://ubc-cs.github.io/cpsc330-2024W2/lectures/notes/12_ensembles.html
  - https://cs229.stanford.edu/notes2022fall/boosting.pdf
  - https://www.cs.toronto.edu/~mren/teach/csc411_19s/lec/lec04.pdf
  - https://www.cs.toronto.edu/~mren/teach/csc411_19s/lec/lec05_matt.pdf