NON-LINEAR CLASSIFIERS

Decision Trees

Bonus Assignment Available!

- 0.5 bonus point (if passing grade for the exam)
- Deadline: **4 November 23:59**
- You can use Vocareum
- Teams of 2
- Submit: Report + Code
- For full instructions, see Brightspace

Recap: Ethics & Discriminative Linear Models

ETHICS

- Ethical considerations such as bias & discrimination, are important issues to consider both in the data that is used and during the construction of a (ML) system.

LINEAR MODELS

- Linear classifiers are linear hyperplanes in feature space that partition the space in (two) classes
- Examples: least squares classifier, logistic regression, support vector classifier
- We used 3 choices in the empirical risk minimisation framework to define and study these classifiers

Learning Goals & Reading

LEARNING GOALS

After practicing with the concepts of this week you are able to

- **Explain when and why non-linear classifiers are needed**
- Explain the basic concepts of two non-linear classifiers: multi-layer perceptrons and **decision trees**
- Explain the **underlying algorithm of decision trees** and (multi-layer) perceptrons and how they are trained.
- Implement a decision tree
- Explain why and how one can **combine multiple classifiers**
- **Contrast a decision tree and a random forest**

LITERATURE

Please study the following material from Chris Bishop's "Pattern Recognition and Machine Learning":

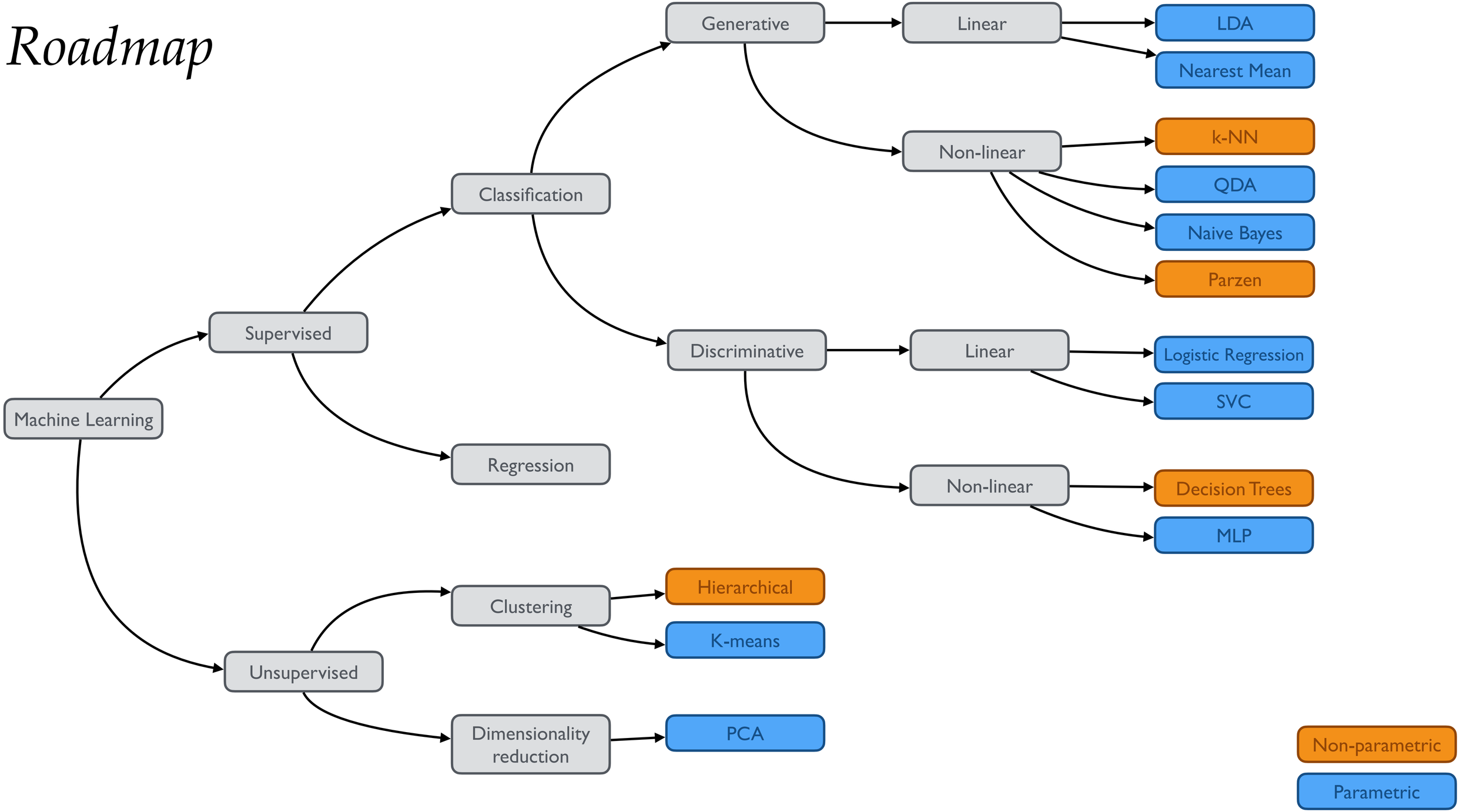
Lecture 6.1

- Section 14.2
- Section 14.4

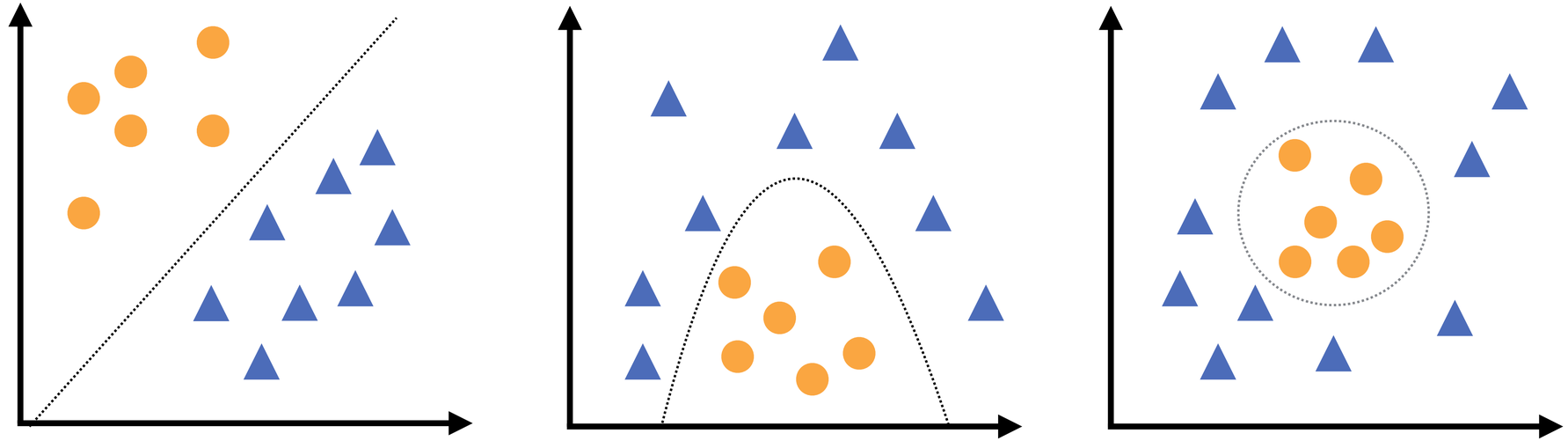
Lecture 6.2

- Section 4.1.7
- Section 5.1
- Section 5.2 up to and including 5.2.1
- Section 5.3 up to and including 5.3.1

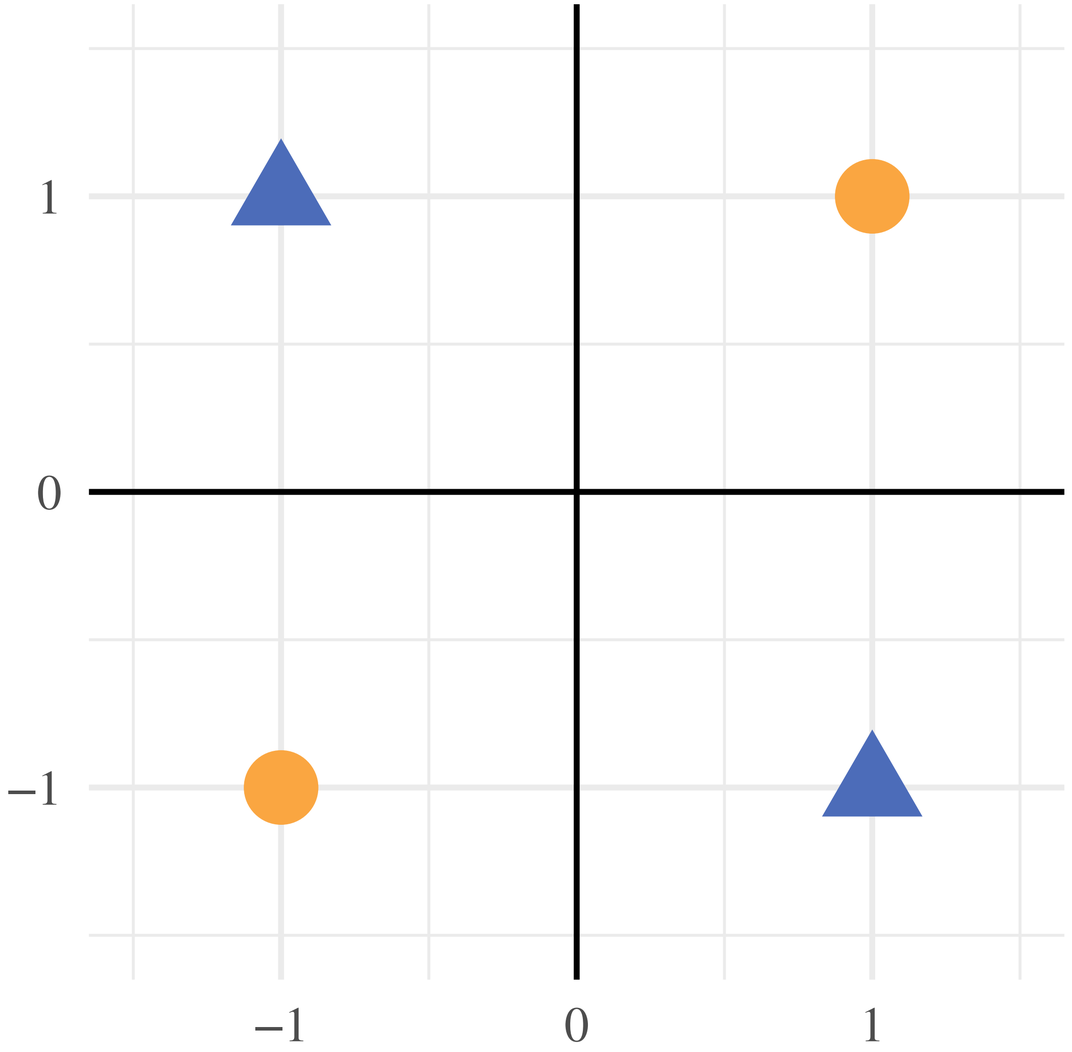
Roadmap


Machine Learning
Supervised
Classification
Generative
Linear
LDA
Nearest Mean
Non-linear
k-NN
QDA
Naive Bayes
Parzen
Discriminative
Linear
Logistic Regression
SVC
Non-linear
Decision Trees
MLP
Regression
Unsupervised
Clustering
Hierarchical
K-means
Dimensionality reduction
PCA
Non-parametric
Parametric

Why non-linear classifiers?



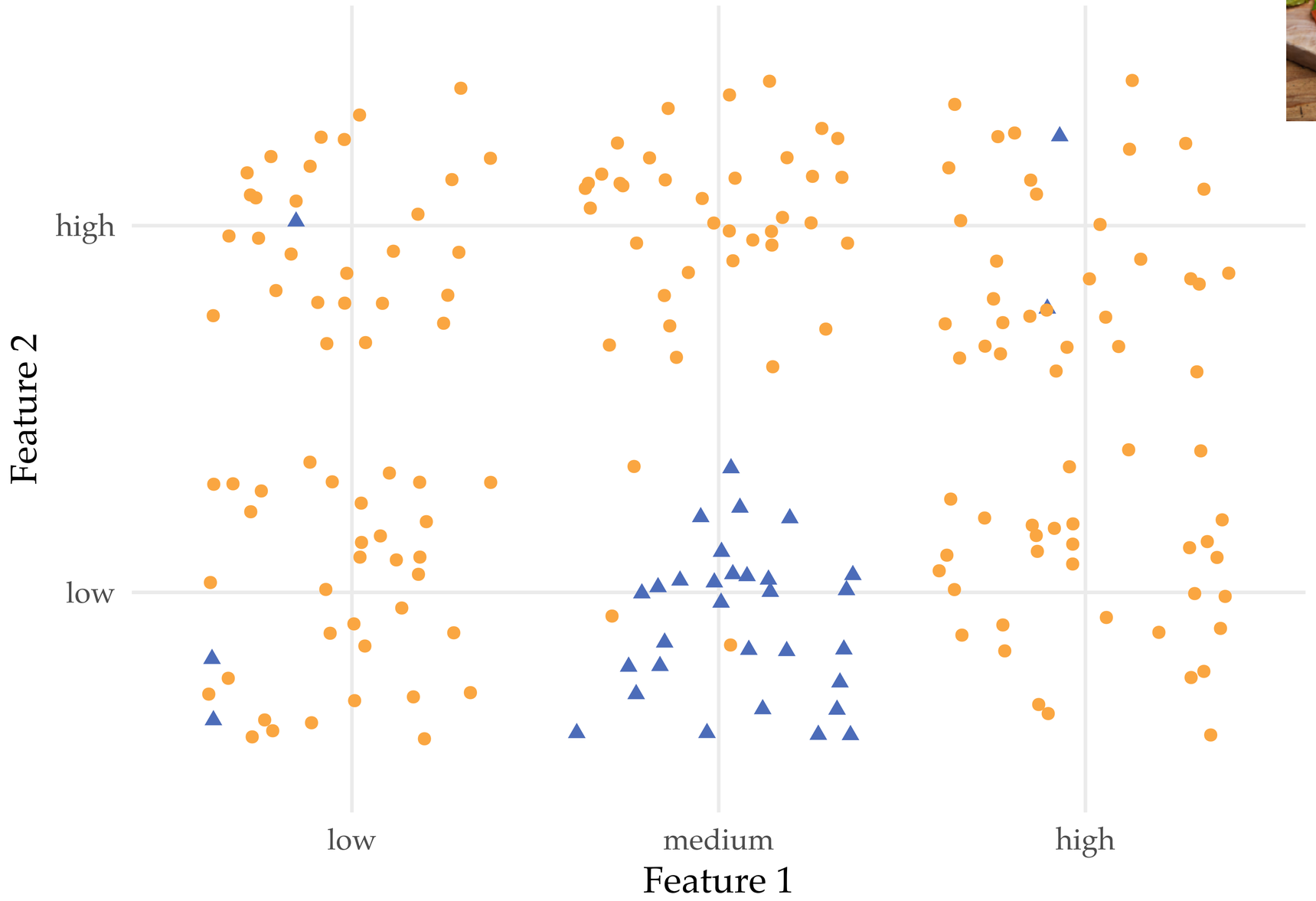
Non-Linear Classification Problem



Today

- Next time: multilayer perceptrons (parametric)
- Now: Decision Trees (non-parametric)
    1. What functions are described by decision trees?
    2. How do we make predictions using a given tree?
    3. How do we learn a tree from data?
    4. What are the advantages & disadvantages of this method?

Non-Linear Classification Problem

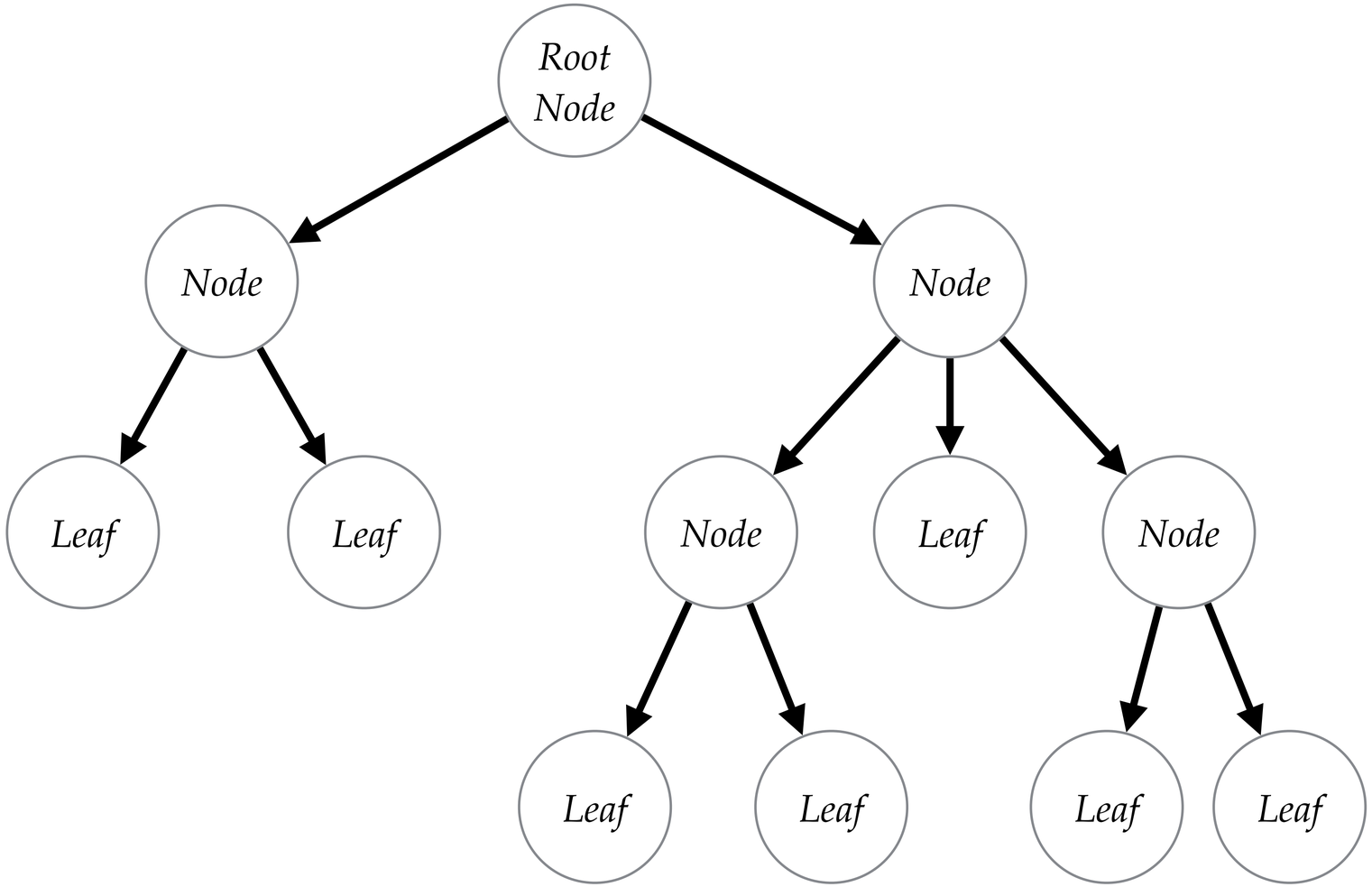


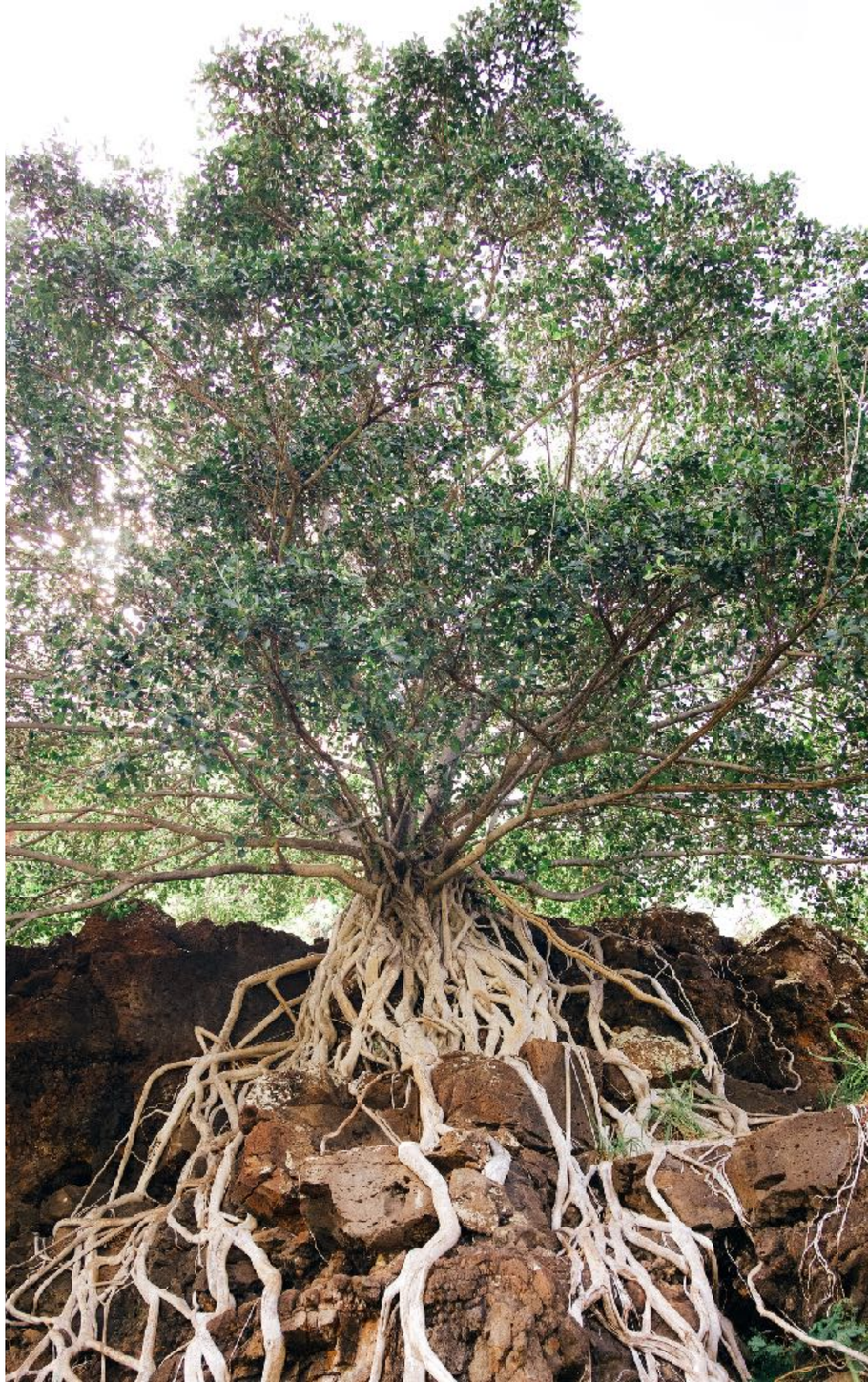
Classifier Construction using Empirical Risk Minimisation

$$\min_{h \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} L(h(\mathbf{x}_i), y_i)$$

1. Define the class of possible functions
2. Define a cost (loss) function to measure the "quality" of each hypothesis
3. Measure the average cost on the training data
4. Find the function that minimises this cost

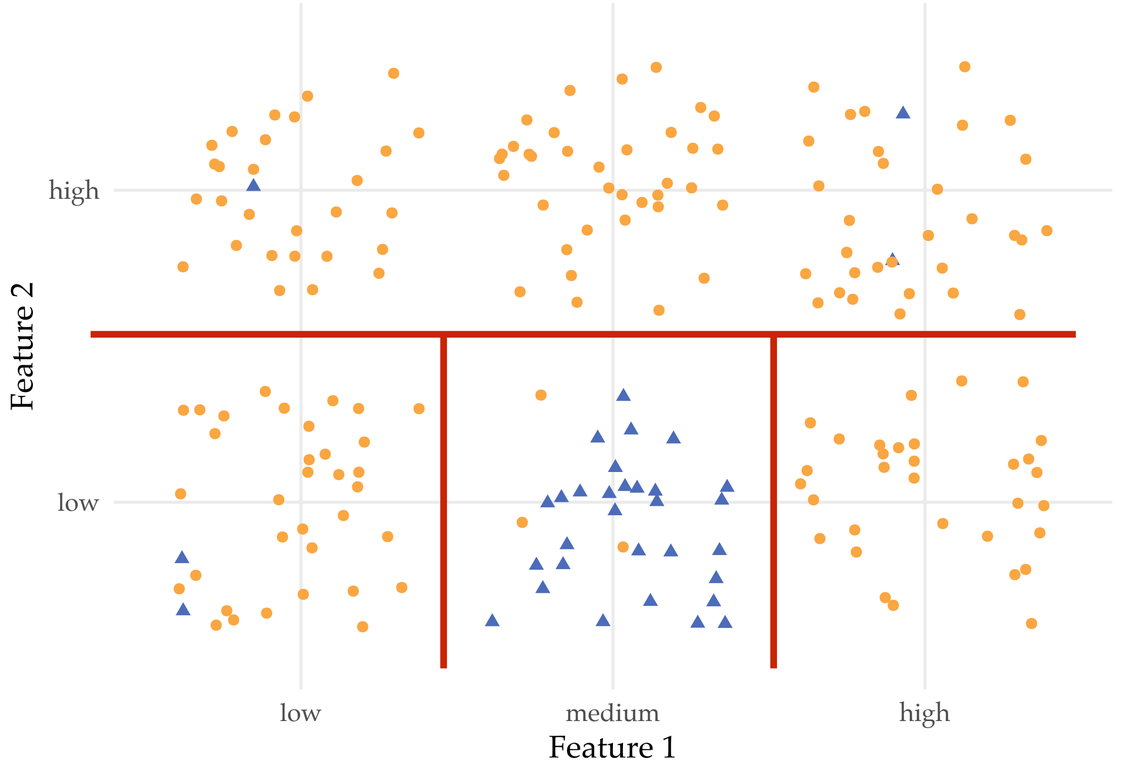
Terminology

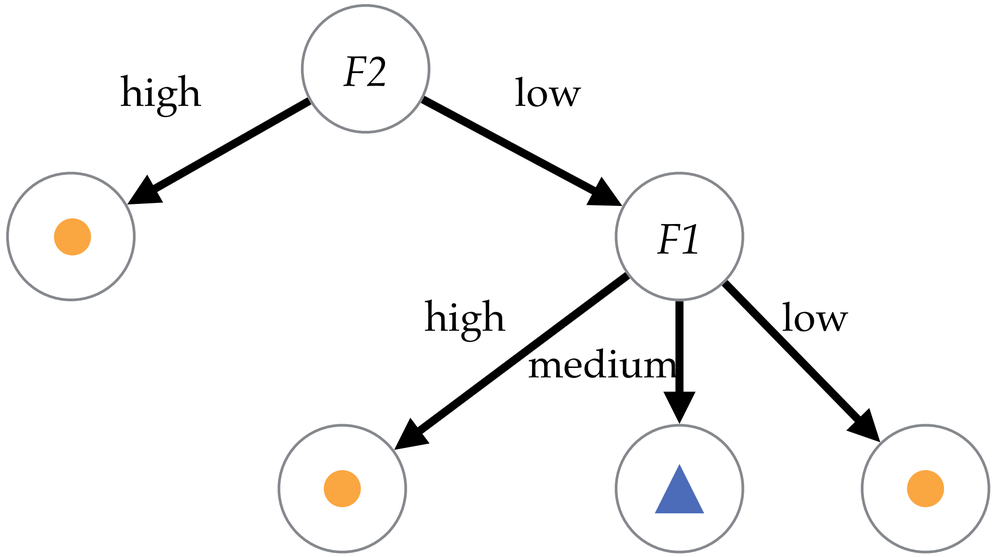




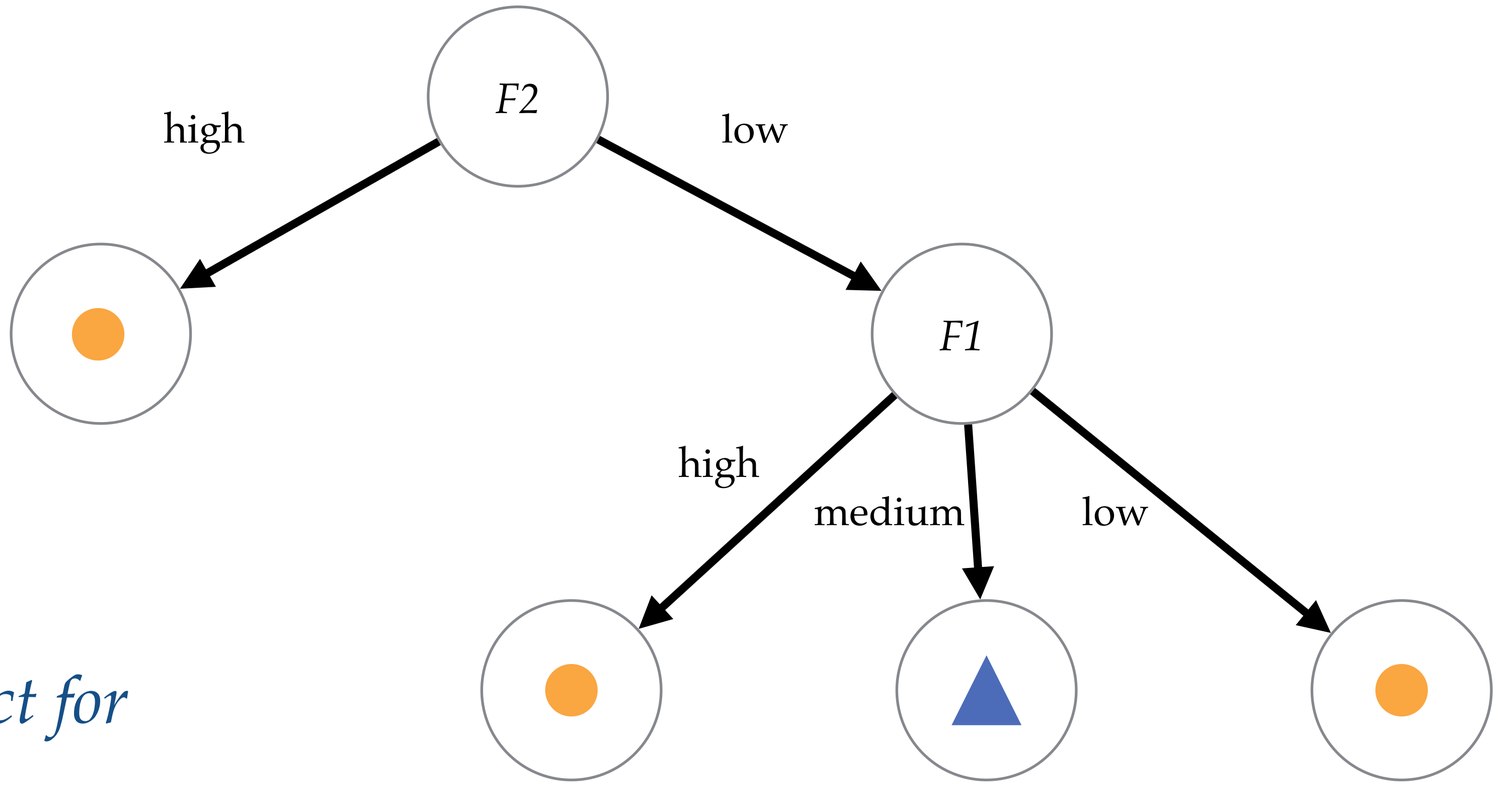
Function class

- Split the feature space using one feature at a time, recursively
- The splitting
  - Forms a tree structure
  - Partitions the space in “rectangles”
- In each rectangle/leaf, we assign a value





Decision Tree Example



What does the model predict for the following objects?

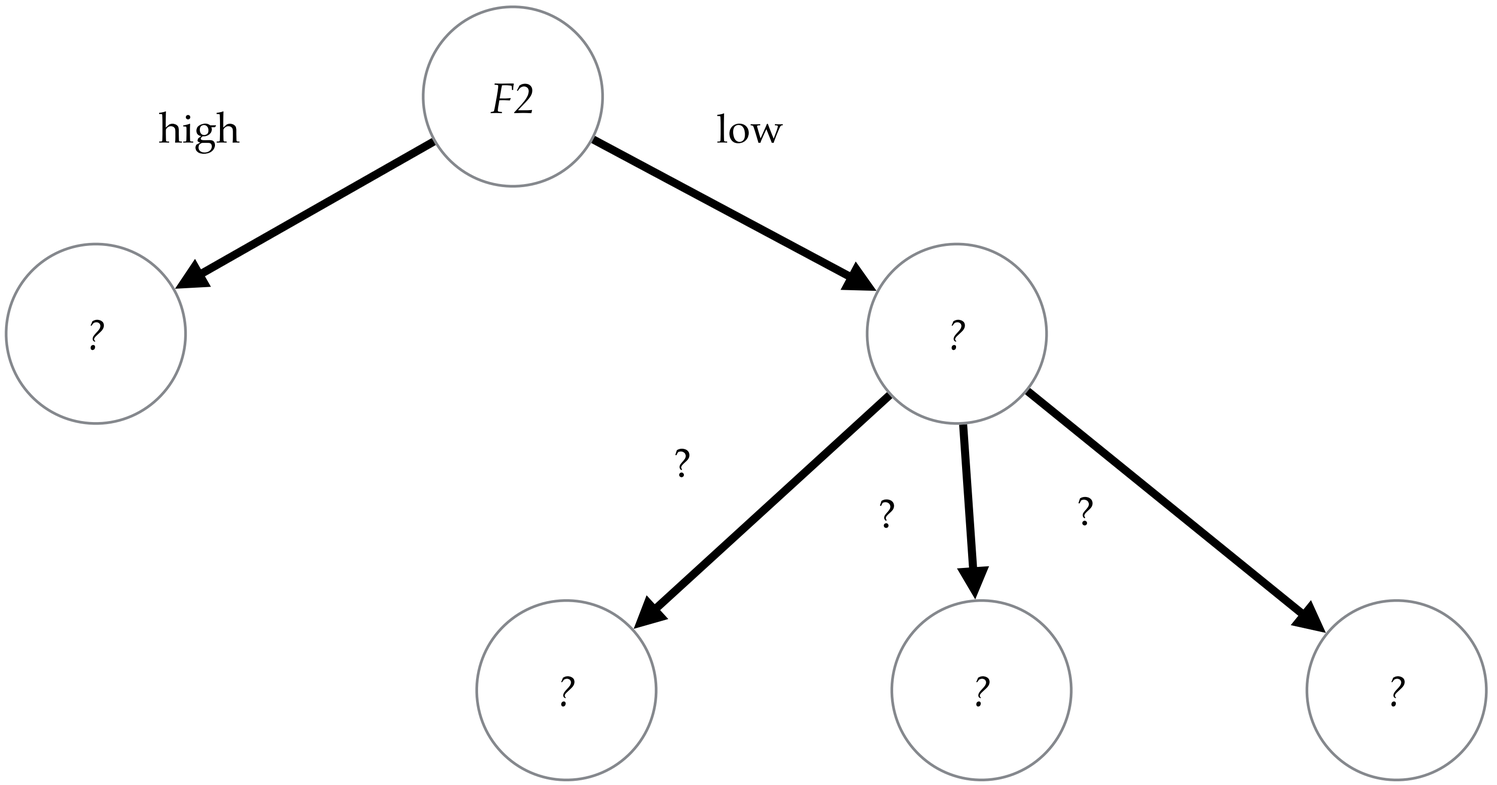
Feature 1	Feature 2	Feature 3	Prediction
low	high	medium	?
medium	low	high	?

Classifier Construction using Empirical Risk Minimisation

$$\min_{h \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} L(h(\mathbf{x}_i), y_i)$$

1. Define the class of possible functions
2. Define a cost (loss) function to measure the "quality" of each hypothesis
3. Measure the average cost on the training data
4. Find the function that minimises this cost

Tree Learning: 2 parts



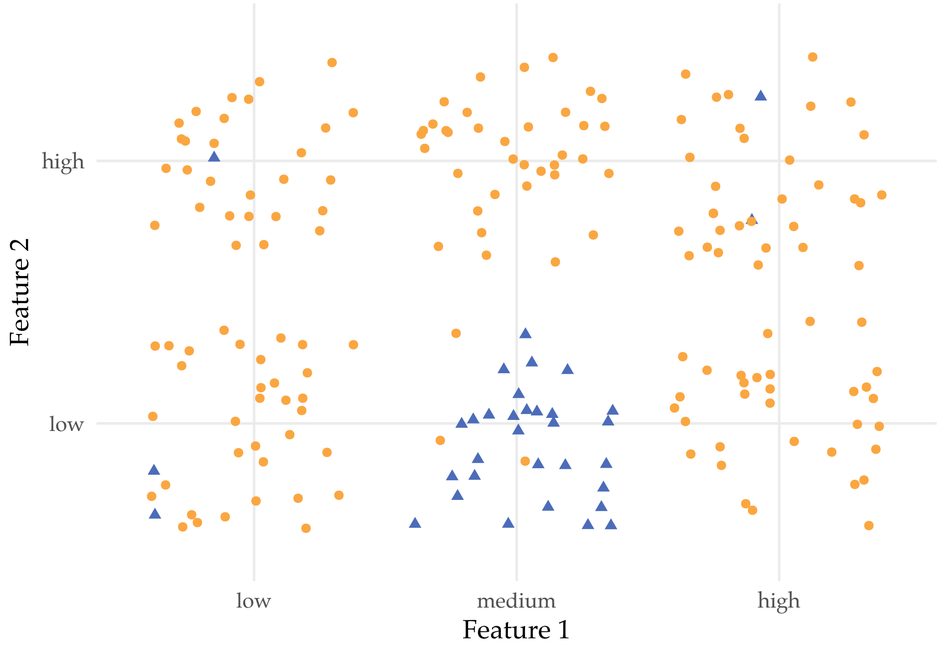
1. Partitioning the feature space/learning the structure of the tree
2. Assigning values to each part (**let's start here**)

Learning: Value Assignment

Given 0-1 loss and the given partitioning, what is the optimal value c_l to assign to each leaf l?

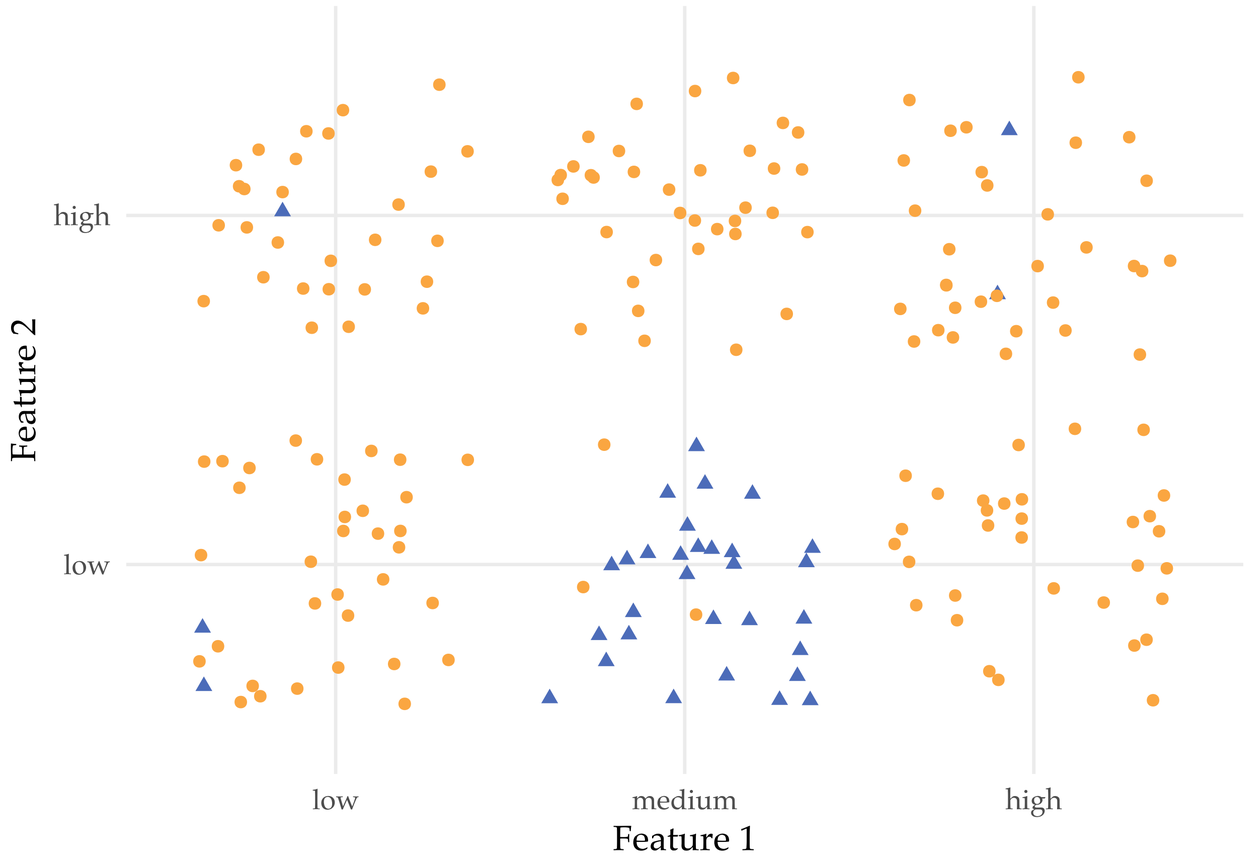
$$h(\mathbf{x}) = \sum_{l \in \text{Leaves}} c_l \mathbb{I}(\mathbf{x} \in l)$$

$$\min_{c_l} \sum_{i=1}^{N} \mathbb{I}(h(\mathbf{x}_i) \neq y_i)$$



Answer: the majority label of all the objects assigned to the leaf is the optimal choice

Example: Root Node



Consider the following “tree”. What should the assignment in the node be?

?

Learning: Splitting

- If all objects in a node belong to one class: we are done!
- If not: find a node-variable combination that increases the **quality** of the tree the most if we split the node further
- How do we measure the **quality**?
  - Reduction in error before/after the split?
  - Not sensitive enough?

Measure of improvement

Number of objects in the node with value V

Set of objects in the node

$$I(S) - \sum_{V \in Values(F)} \frac{|S_V|}{|S|} I(S_V)$$

Feature that we consider

If $I(.)$ is the misclassification error in a set, this measures how much the 0-1 loss will decrease if we split the node into multiple nodes using feature F

Other ways to measure impurity of a node $I(.)$?

Splitting Criteria: Choosing I(S)

$$1 - \max_c p_c$$

Misclassification

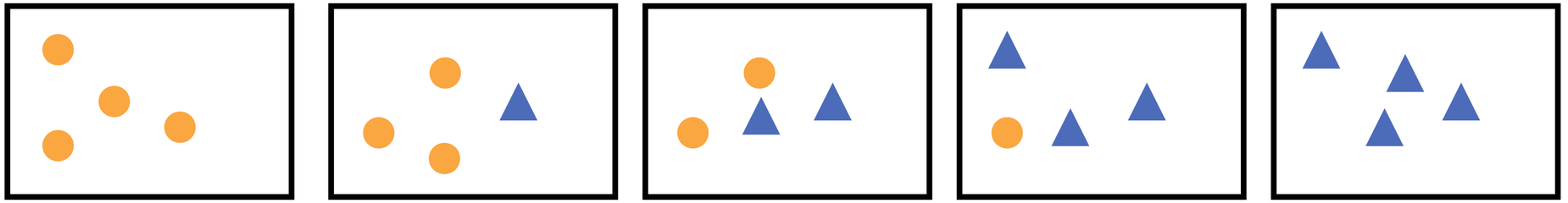
$$-\sum_c p_c \log(p_c)$$

Entropy

$$\sum_c p_c(1 - p_c)$$

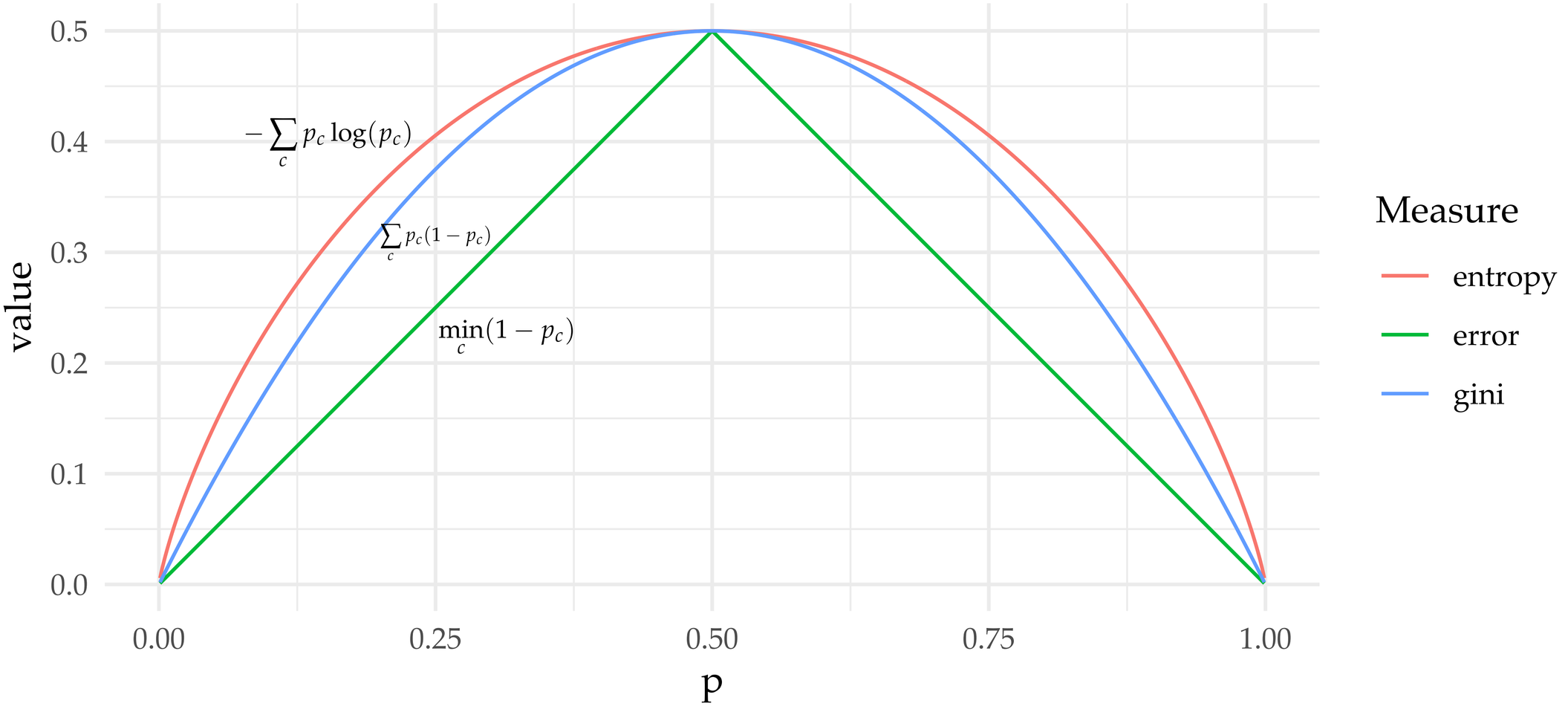
Gini Index

Splitting Criteria: Information Measures



$p_{\bullet}$	1.0	0.75	0.5	0.25	0.0
$1 - \max_c p_c$	0.0	0.25	0.5	0.25	0.0
$-\sum_c p_c \log(p_c)$	0.0 (!)	0.56	0.69	0.56	0.0 (!)
$\sum_c p_c(1 - p_c)$	0.0	0.375	0.5	0.375	0.0

Comparing Splitting Criteria



Note: Entropy scaled to have the same maximum

Example: Information Gain Calculation

Overal:

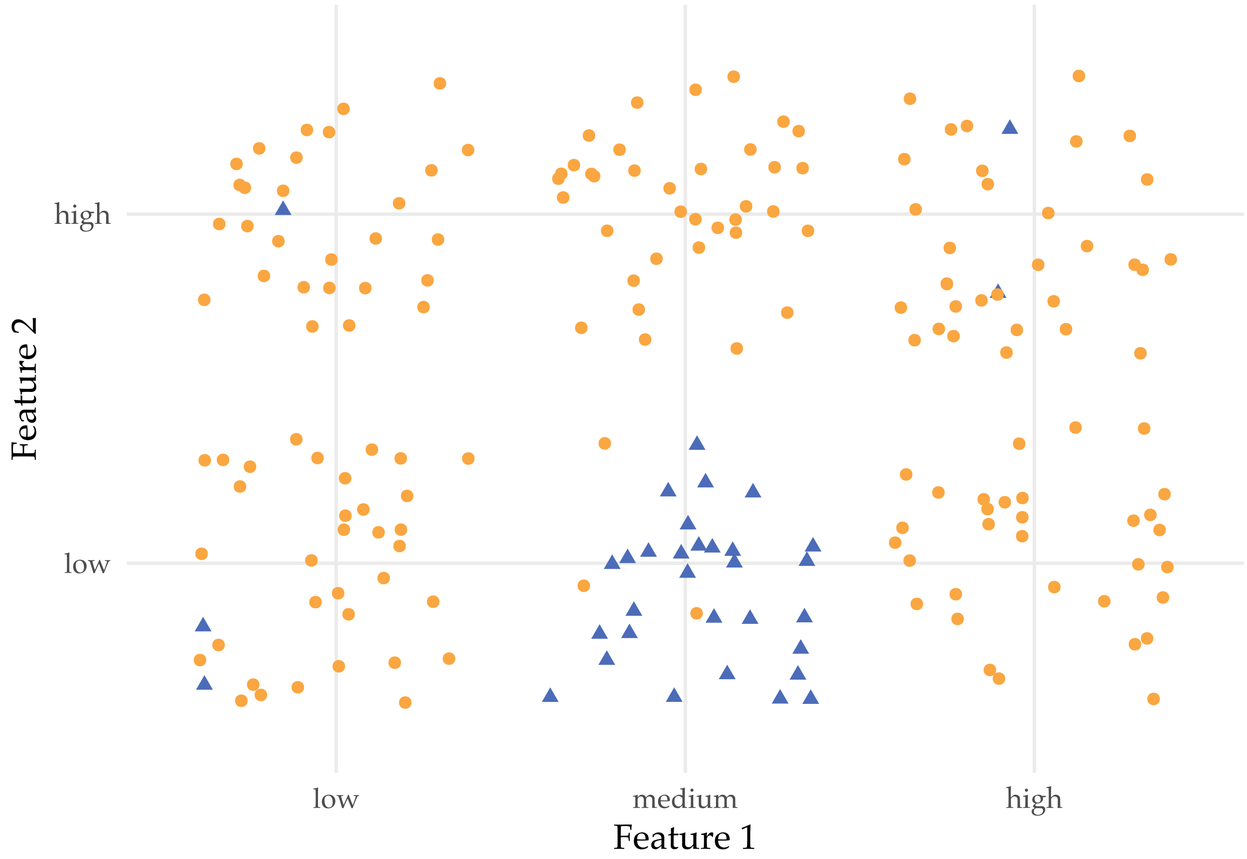
35/200

Feature 1:

Low (3/65), Medium (30/69), High (2/66)

Feature 2:

High (3/99), Low (32/101)



Should we split using feature 1 or feature 2?

Information Gain

$$I(S) - \sum_{V \in Values(F)} \frac{|S_V|}{|S|} I(S_V)$$

with

$$I(S) = -\sum_{c} p_c \log(p_c)$$

Example: Information Gain Calculation

$$I(S) = -\sum_c p_c \log(p_c) \qquad I(S) - \sum_{V \in Values(F)} \frac{|S_V|}{|S|} I(S_V)$$

Node Entropy

Overal:

35/200

$$-(\frac{35}{200}\log(\frac{35}{200}) + \frac{165}{200}\log(\frac{165}{200})) = 0.464$$

Feature 1:

$$-(\frac{3}{65}\log(\frac{3}{65}) + \frac{62}{65}\log(\frac{62}{65})) = 0.187$$

Low (3/65)

$$-(\frac{30}{69}\log(\frac{30}{69}) + \frac{39}{69}\log(\frac{39}{69})) = 0.685$$

Medium (30/69)

$$-(\frac{2}{66}\log(\frac{2}{66}) + \frac{64}{66}\log(\frac{64}{66})) = 0.136$$

High (2/66)

Feature 2:

$$-(\frac{3}{99}\log(\frac{3}{99}) + \frac{96}{99}\log(\frac{96}{99})) = 0.136$$

High (3/99)

$$-(\frac{32}{101}\log(\frac{32}{101}) + \frac{69}{101}\log(\frac{69}{101})) = 0.624$$

Low (32/101)

Information Gain

$$0.464 - (\frac{65}{200}0.187 + \frac{69}{200}0.685 + \frac{66}{200}0.136) = 0.122$$

$$0.464 - (\frac{99}{200}0.136 + \frac{101}{200}0.624) = 0.081$$

Splitting Criteria: Gain ratio

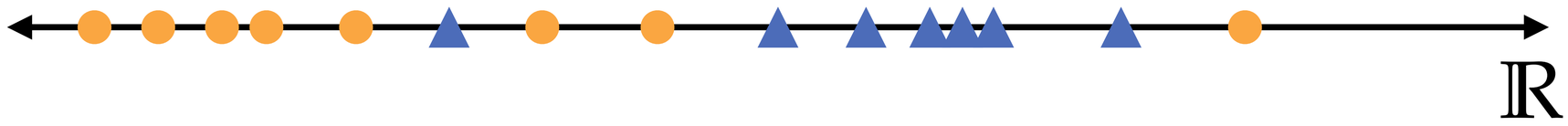
- The information gain is biased towards features with many discrete values
- We may overfit on such features (for instance, someone's social security number)
- One idea by Quinlan is to correct for this using the entropy of the feature, the intrinsic value (IV)
- Actually some (many) implementations only use binary splits.

$$IG(S) = I(S) - \sum_{V \in Values(F)} \frac{|S_V|}{|S|} I(S_V)$$

$$IV(S) = - \sum_{V \in Values(F)} \frac{|S_V|}{|S|} \log \frac{|S_V|}{|S|}$$

$$GainRatio(S) = \frac{IG(S)}{IV(S)}$$

Splitting: Continuous Variables



Extra challenge: we need to find a threshold for the split.

Consider all thresholds and pick the optimal one

How many thresholds do we need to consider?

Learning: Stopping

- If we fit pure trees, we quickly start overfitting (why?)
- Criteria:
  - minimum node size
  - maximum tree depth
  - minimum information gain
- Alternative/complementary strategy: grow and prune

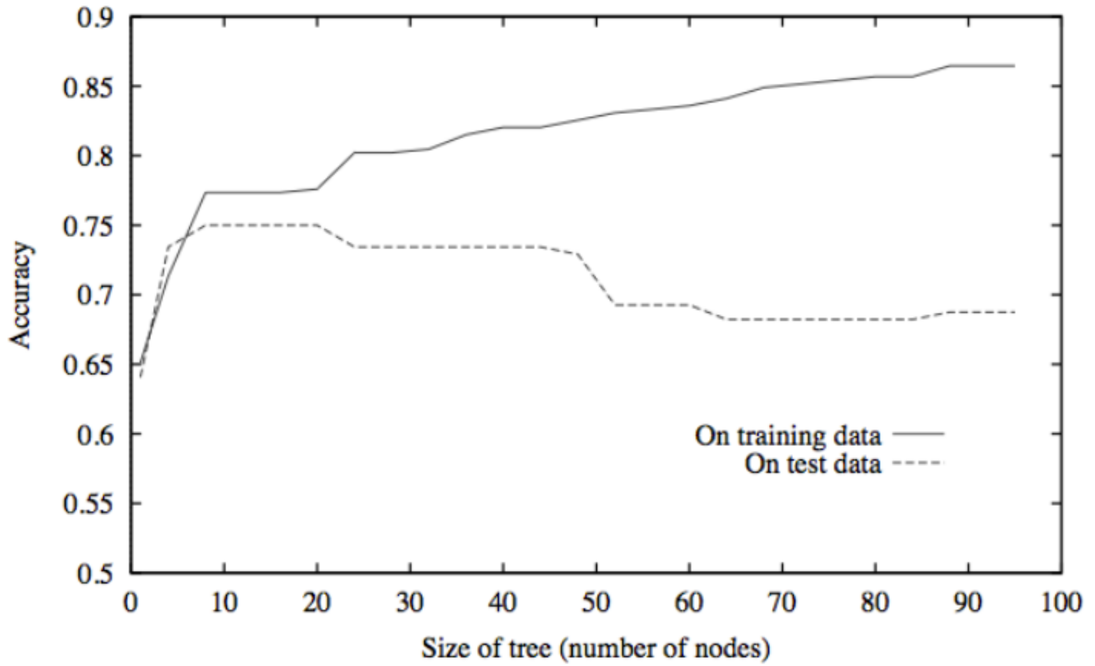


Figure credit: Tom Mitchell, 1997

Learning: Pruning

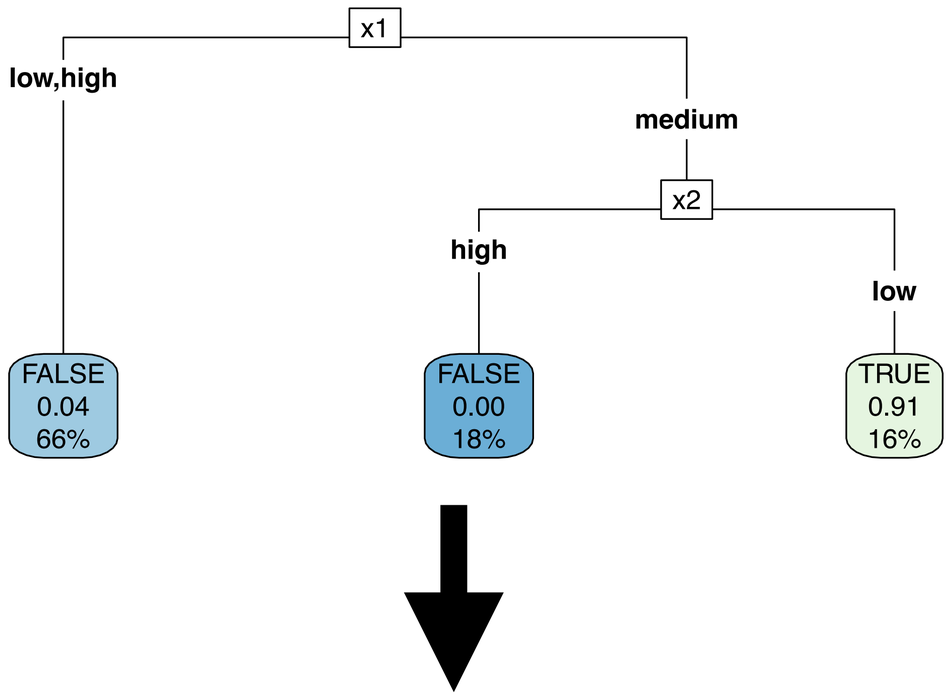
$$C_\alpha(Tree) = \sum_{l \in \text{Leaves}} |S_l| I(S_l) + \alpha |\text{Leaves}|$$

Leaves in the tree

For each alpha, optimise the tree by removing subtrees

Select optimal alpha using a validation set

From Trees to Rule Lists



We can convert the decisions in the tree to a set of rules, which may be more interpretable, or easier to communicate. We can also try to further simplify these rules

```
0.00 when x1 is      medium & x2 is high

0.04 when x1 is low or high

0.91 when x1 is      medium & x2 is  low
```

Generalizations

- Multi-class
  - Take the sum over all classes in the entropy term
- Regression
  - Squared loss as the loss function
  - Leaf value: average outcome
  - Split criterion: minimize the variance
  - Covered in Probability & Statistics

Summary: Decision Trees

- Non-linear classifier
- Choices in learning:
  - Splitting criterion (information gain, gain ratio, entropy / gini index, etc.)
  - Stopping (minimum gain, node size, max depth)
  - Pruning (remove parts based on performance on validation set)

Advantages & Disadvantages

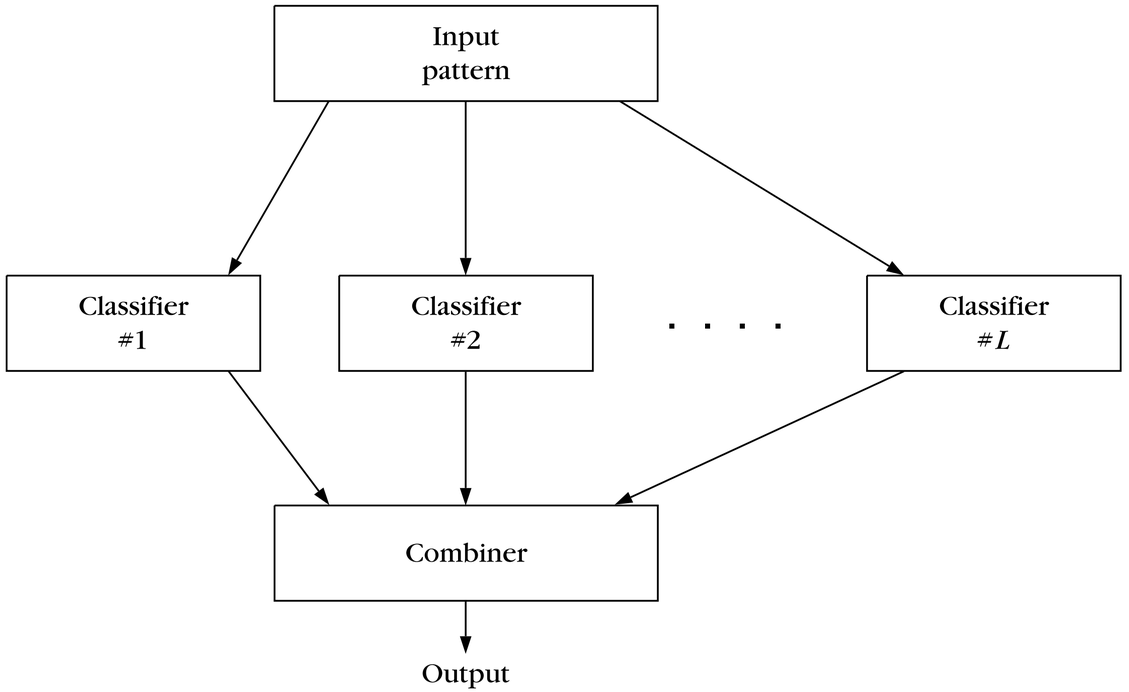
- ✅ “Interpretable”
- ✅ Automatic feature selection
- ✅ Easy to incorporate discrete features and missing values
- ✅ Fast

- ❌ Unstable
- ❌ Cannot model linear relationships efficiently
- ❌ “Greedy”

CLASSIFIER COMBINING

Classifier Combining

- Multiple classifiers that each make a prediction: perhaps they make different kinds of mistakes
- How do we combine these predictions?
- Does it help to combine them?



Condorcet's Jury Theorem

Condorcet's Jury Theorem

Question

When does it make sense to combine decisions?

Theorem

If $p > 0.5$, adding voters to the jury increases probability of correct majority vote (assuming voting is independent)

Nicolas de Condorcet (1743-1794)

Kuncheva, L. I. (2004). Combining Pattern Classifers. Methods and Algorithms. Wiley, Chichester.

Combining Strategies

- Fixed Rules
  - Hard rules e.g. majority voting
  - Soft rules: e.g. mean, product

$$\max_y \sum_{j=1}^{C} \frac{1}{C} h_j(y|x) \qquad \max_y \prod_{j=1}^{C} h_j(y|x) \qquad \min_{p(y|\mathbf{x})} \sum_j D(h_j(y|\mathbf{x}), p(y|\mathbf{x}))$$

- Learned rules
  - Learn a classifier to output a decision based on the outputs of a set of base classifiers

What to combine

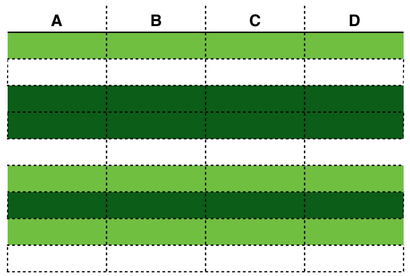
- Where do the multiple classifiers come from?
  - Different classifiers on the same dataset?
  - The same classifier on different datasets?
- Diversity may be helpful
  - Let's consider an example where we construct "diverse" classifiers using different versions of the same dataset

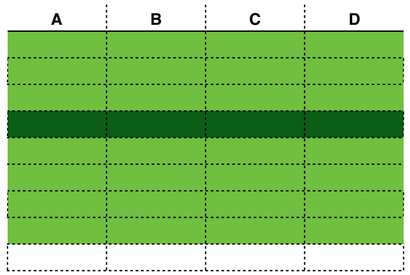
Random Forests

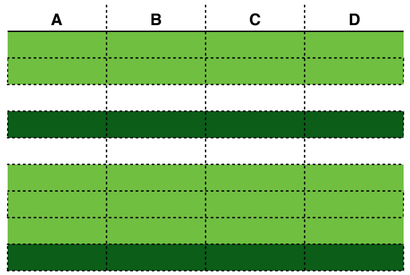
- Single tree might easily overfit and might be unstable
- Combine multiple trees
- Should be sufficiently “different”
    1. Randomly select features
    2. Randomly select samples (with replacement)

Random Forest: Construct a large number of trees using randomly selected objects and features and combine their decisions

Bagging (Bootstrap Aggregation)







$$h_1(\mathbf{x})$$

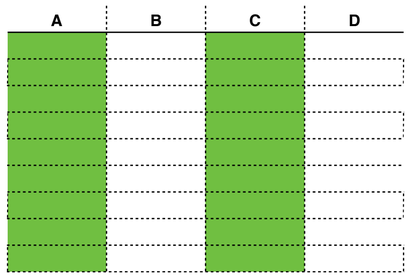
$$h_2(\mathbf{x})$$

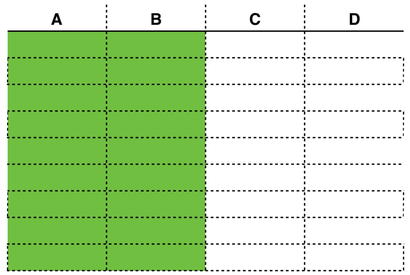
$$h_B(\mathbf{x})$$

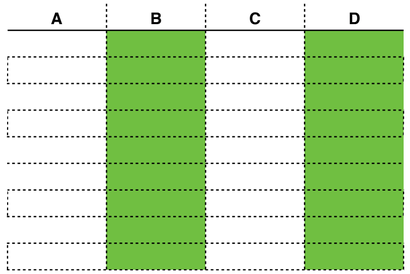
$$\frac{1}{B}\sum_{b=1}^{B} h_b(\mathbf{x})$$

Breiman, L. (1996). Bagging predictors. Machine Learning, 140, 123–140.

Random Subspaces







$$h_1(\mathbf{x})$$

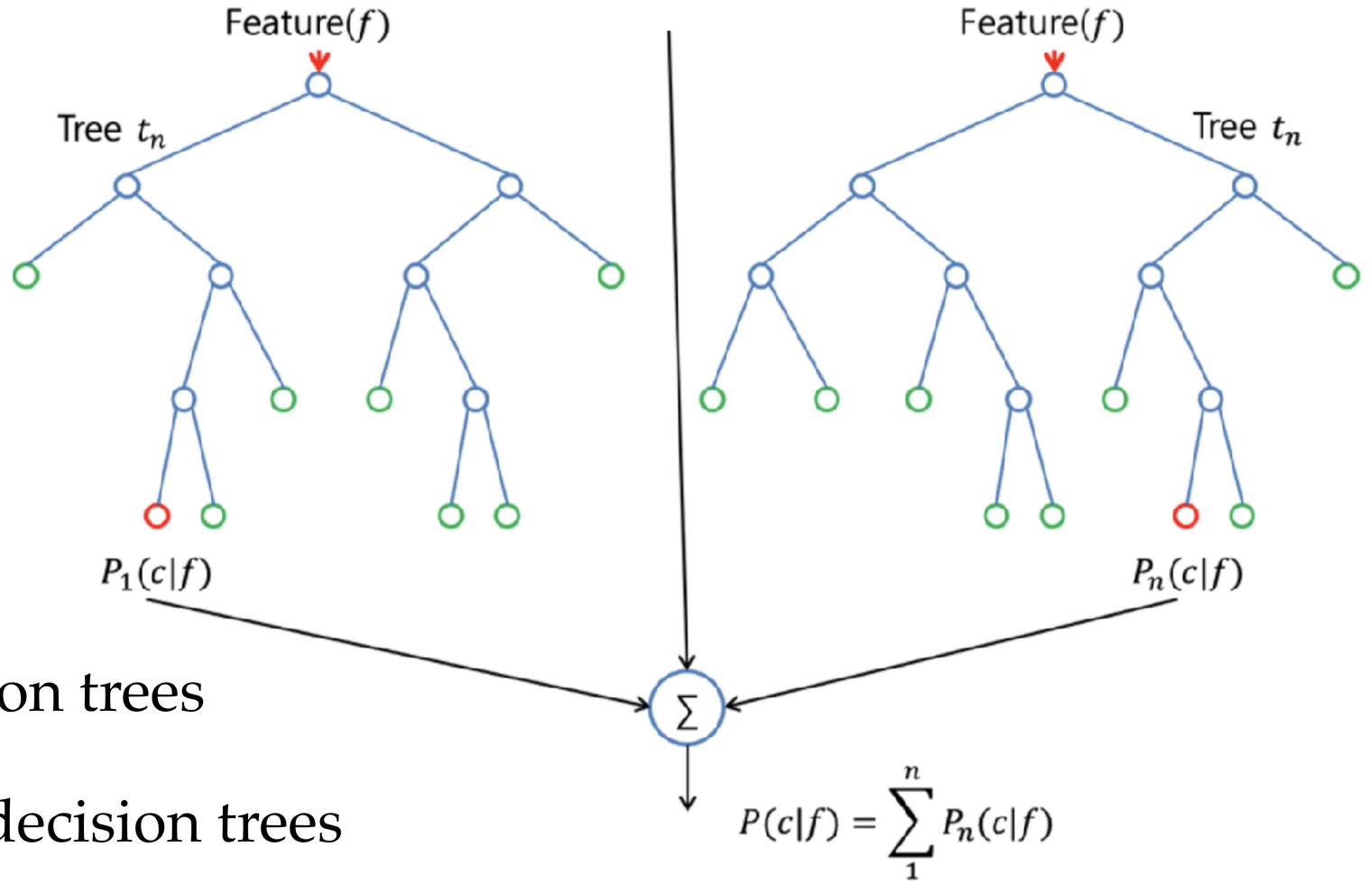
$$h_2(\mathbf{x})$$

$$h_B(\mathbf{x})$$

$$\frac{1}{B}\sum_{b=1}^{B} h_b(\mathbf{x})$$

Ho, Tin Kam. "The random subspace method for constructing decision forests." *IEEE transactions on pattern analysis and machine intelligence* 20.8 (1998): 832-844.

Random Forest



- Breiman: bagging with decision trees
- Ho: random subspaces with decision trees
- Random Forest in practice: bagging and random subspaces with decision trees

Parameters

TREES

- Depth
- Number of leaves
- Minimum number of objects per node
- Information criterion
- Pruning

FOREST

- Number of trees
- Size of subspaces considered

Pros and Cons

PROS

- Flexible/low bias
- Works for many different types of data
- Embarrassingly parallel
- Produces out-of-bag (OOB) estimates
- (Scale) invariant
- (Relatively) few hyperparameters

CONS

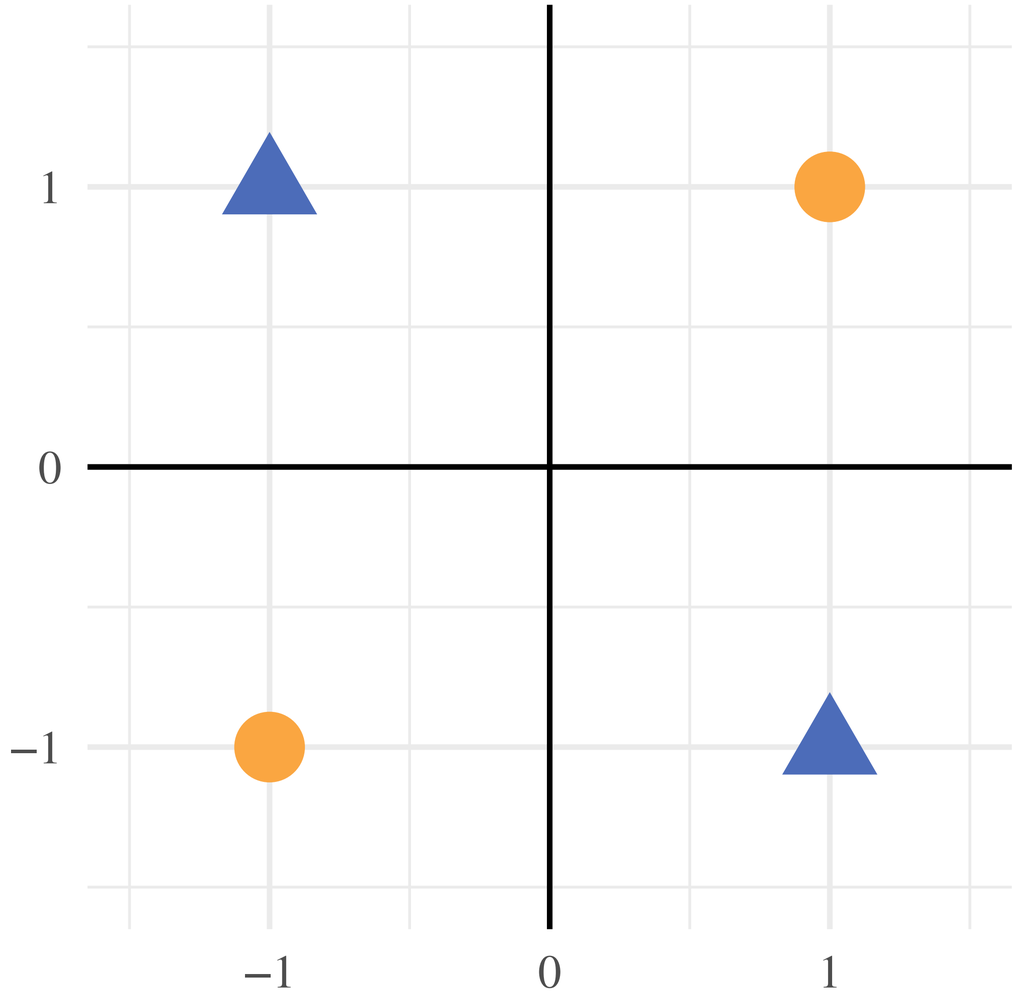
- Harder to interpret than single trees
- Computationally expensive


KINECT
SPORTS

Recap

- Decision trees: a greedy space partitioning approach to non-linear classification that leads to an “intuitive” classifier, but typically has high variance
- Classifier combining can lead to better predictions by combining multiple diverse classifiers
- Random Forests are a combination of decision trees constructed using random subsampling of objects and features.
- Next time: MLPs, a parametric non-linear discriminative classifier that acts as a “learned” combiner

Revisiting XOR



Can the decision tree learner covered today solve this?