Lecture 7: Decision Trees

Instructor: Saravanan Thirumuruganathan

Outline

1. Geometric Perspective of Classification
2. Decision Trees

Geometric Perspective of Classification

Perspective of Classification

- Algorithmic
- Geometric
- Probabilistic
- ...

Geometric Perspective of Classification

- Gives some intuition for model selection
- Understand the distribution of data
- Understand the expressiveness and limitations of various classifiers

Feature Space[1]

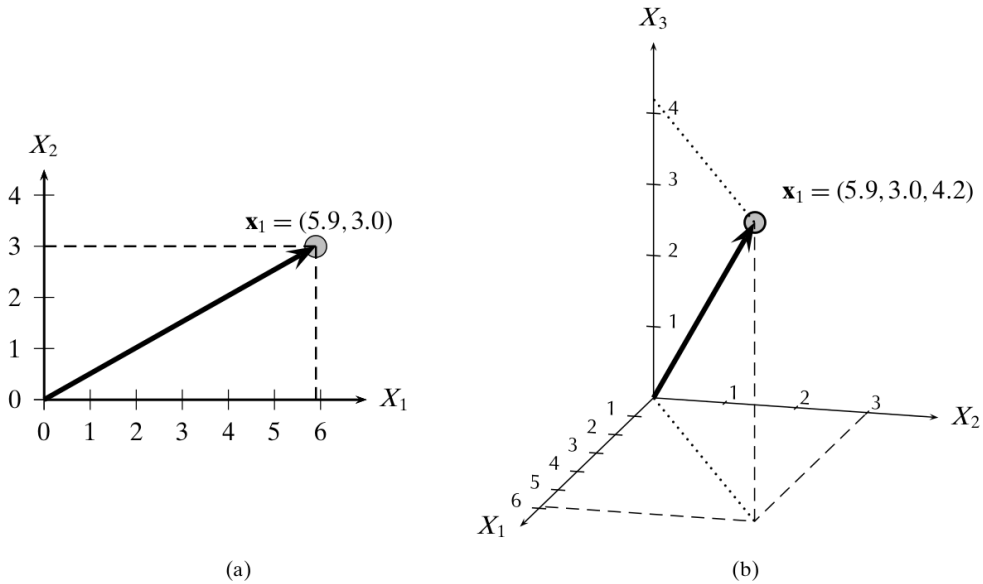
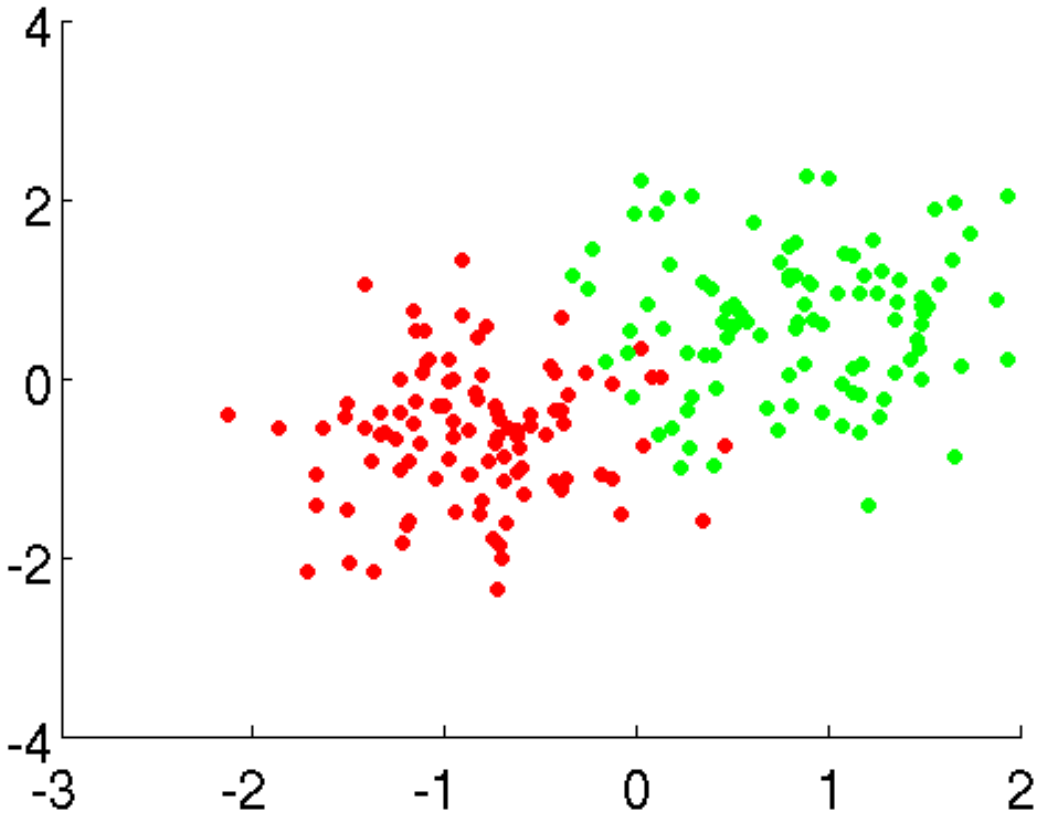


Figure 1.1. Row $\mathbf{x}_1$ as a point and vector in (a) $\mathbb{R}^2$ and (b) $\mathbb{R}^3$.

- **Feature Vector:** d-dimensional vector of features describing the object
- **Feature Space:** The vector space associated with feature vectors

[1] DMA Book

Feature Space in Classification



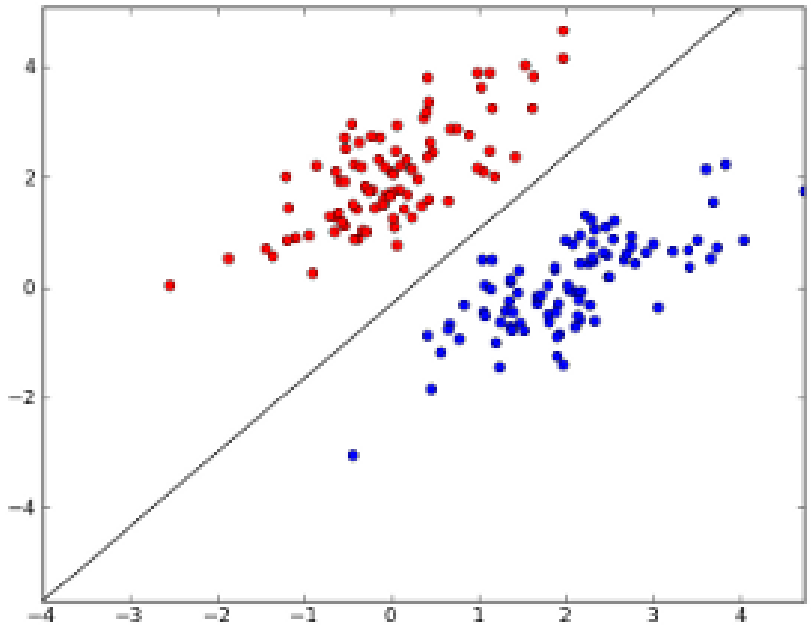
Geometric Perspective of Classification

- **Decision Region:** A partition of feature space such that all feature vectors in it are assigned to same class.
- **Decision Boundary:** Boundaries between neighboring decision regions

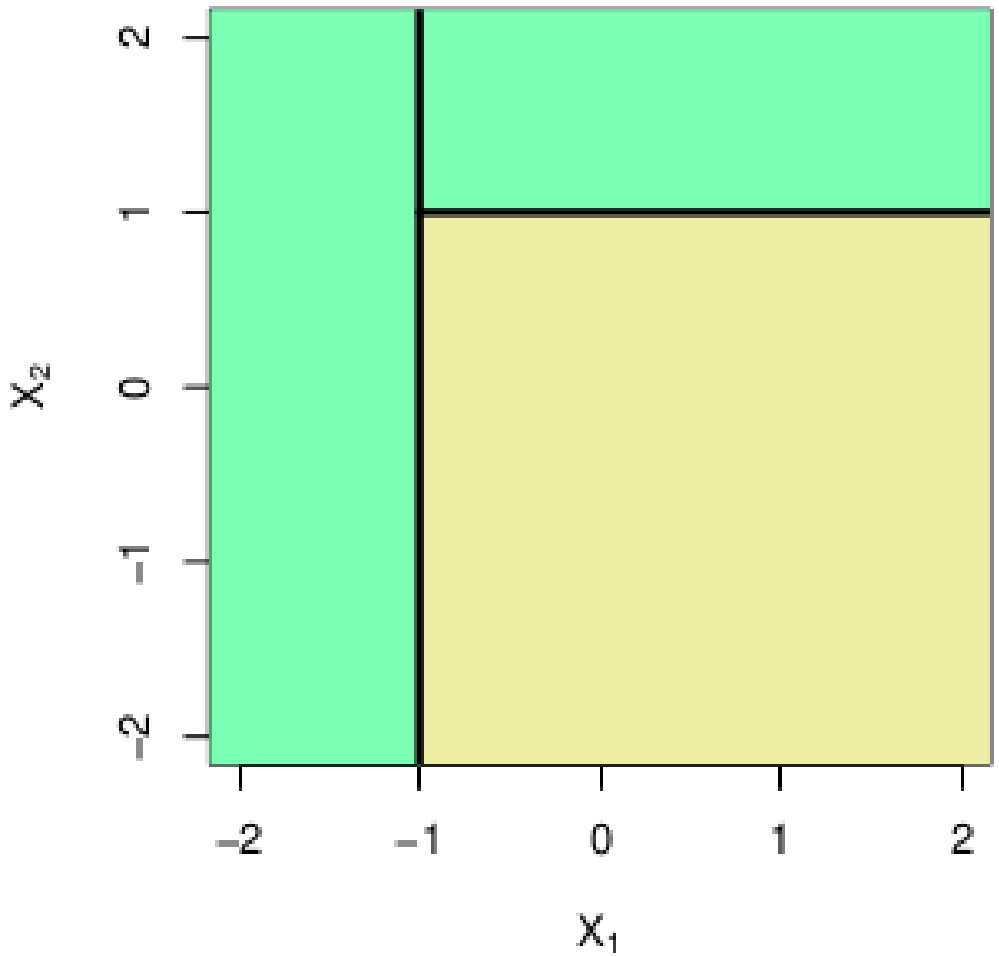
Geometric Perspective of Classification

- Objective of a classifier is to *approximate* the "real" decision boundary as much as possible
- Most classification algorithm has specific expressiveness and limitations
- If they align, then classifier does a good approximation

Linear Decision Boundary

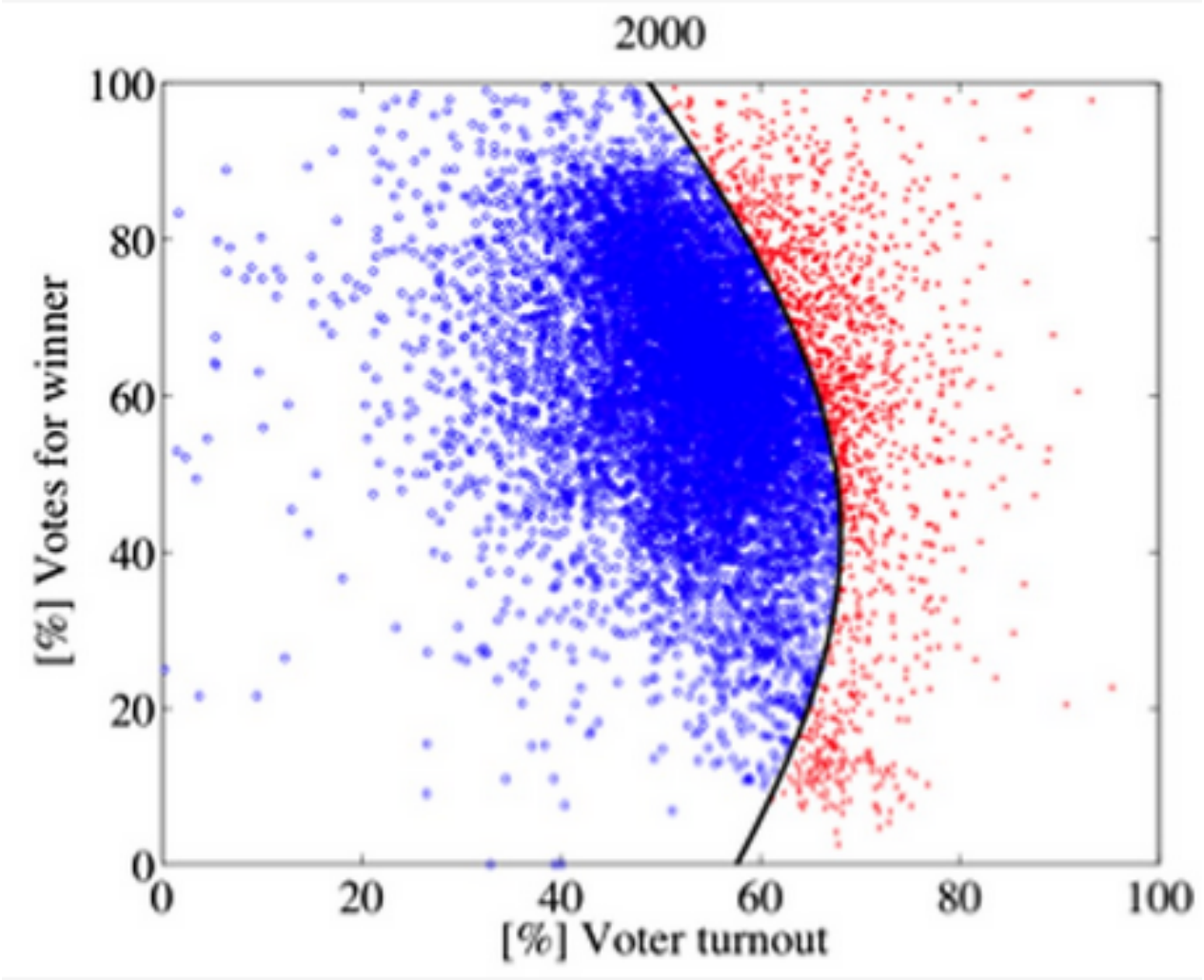


Piecewise Linear Decision Boundary[2]



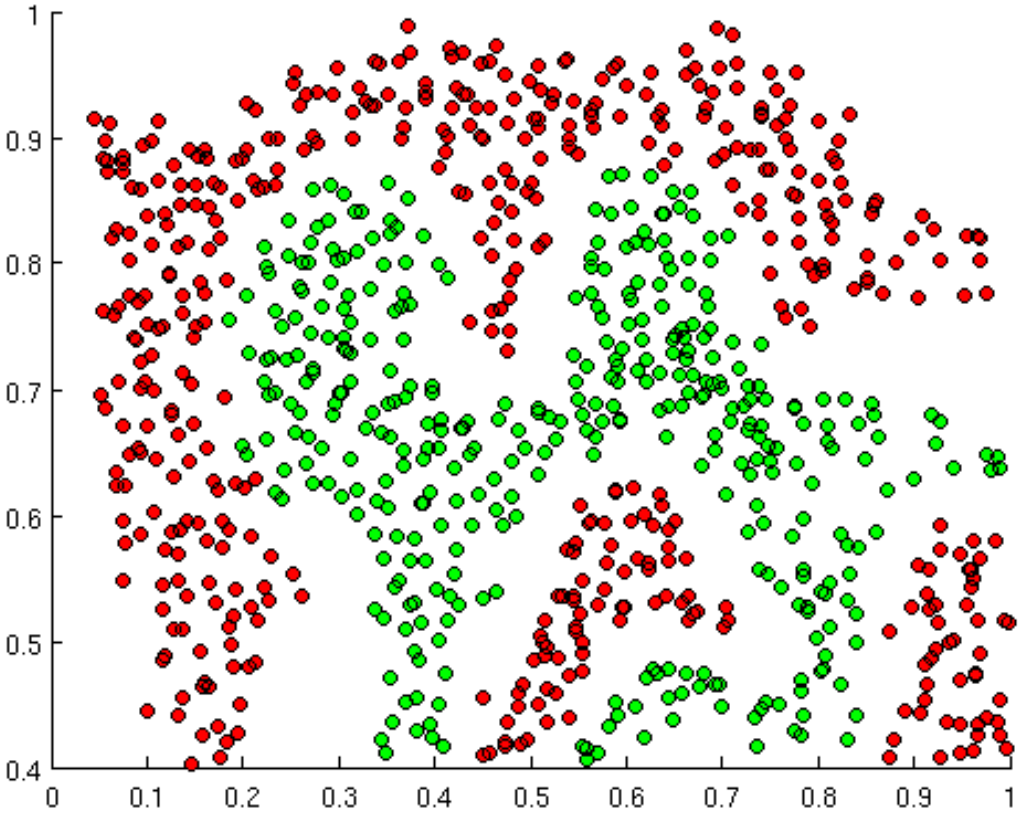
[2]ISLR Book

Quadratic Decision Boundary[3]



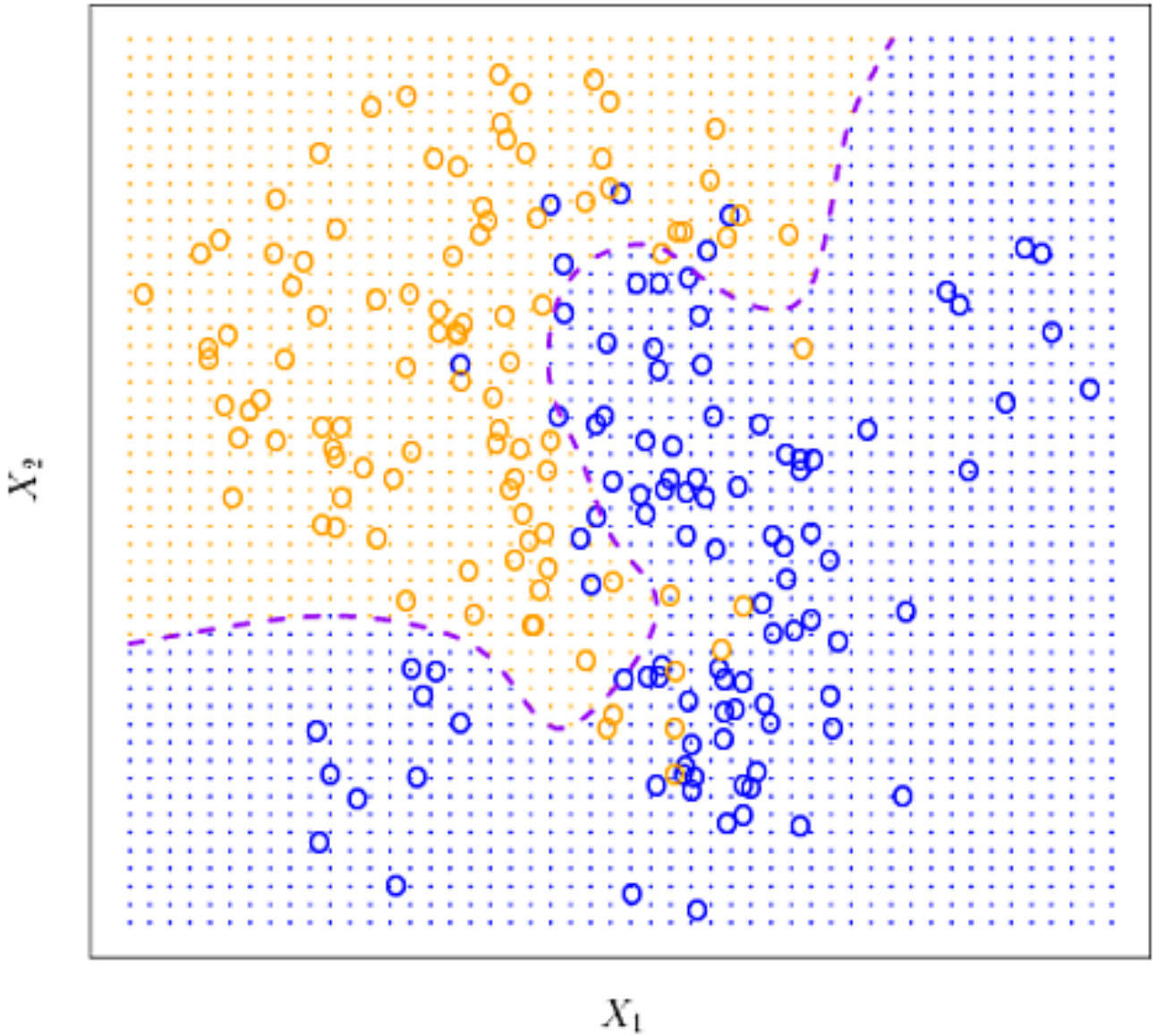
[3] Figshare.com

Non-linear Decision Boundary[4]



[4]ISLR Book

Complex Decision Boundary[5]

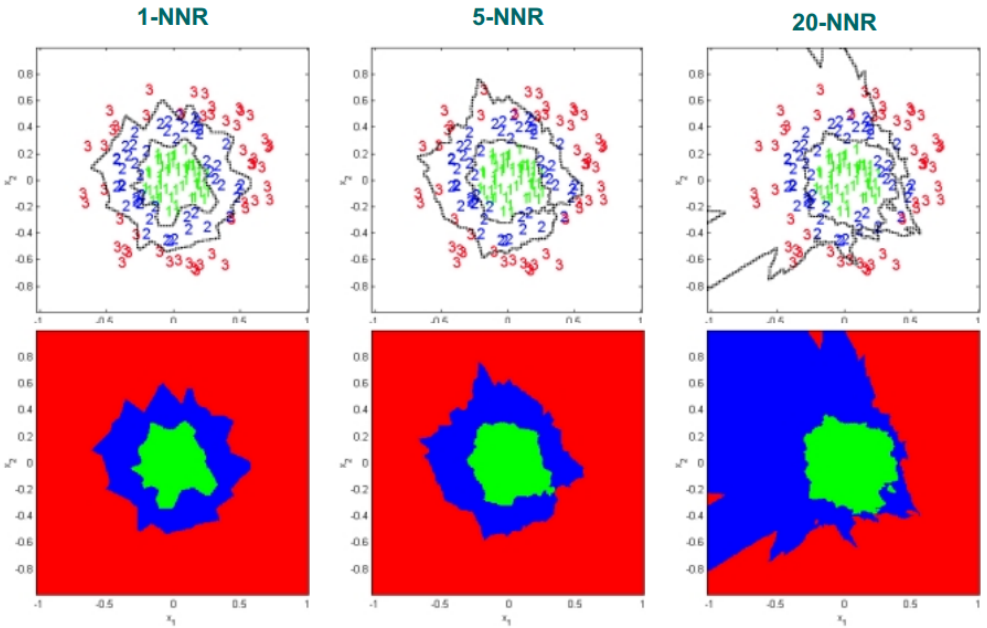


[5]

Classifier Selection Tips

- If decision boundary is linear, most *linear* classifiers will do well
- If decision boundary is non-linear, we sometimes have to use kernels
- If decision boundary is piece-wise, decision trees can do well
- If decision boundary is too complex, k-NN might be a good choice

k-NN Decision Boundary[6]



- Asymptotically Consistent: With infinite training data and large enough k, k-NN approaches the best possible classifier (Bayes Optimal)
- With infinite training data and large enough k, k-NN could approximate most possible decision boundaries

[6] ISLR Book

Decision Trees

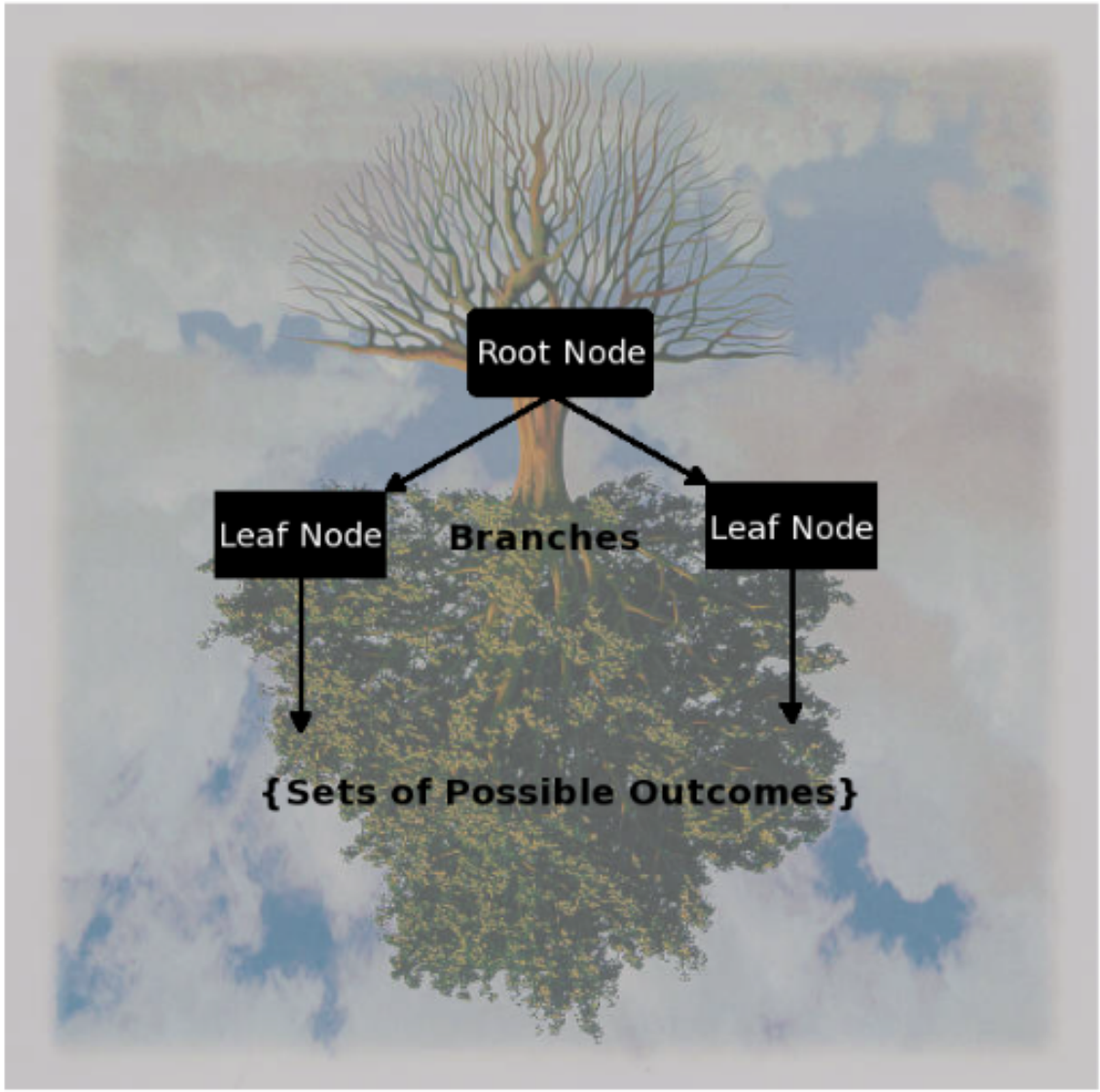
Strategies for Classifiers

- **Parametric Models:** Makes some assumption about data distribution such as density and often use explicit probability models
- **Non-parametric Models:** No prior assumption of data and determine decision boundaries directly.
  - k-NN
  - Decision tree

Tree[7]

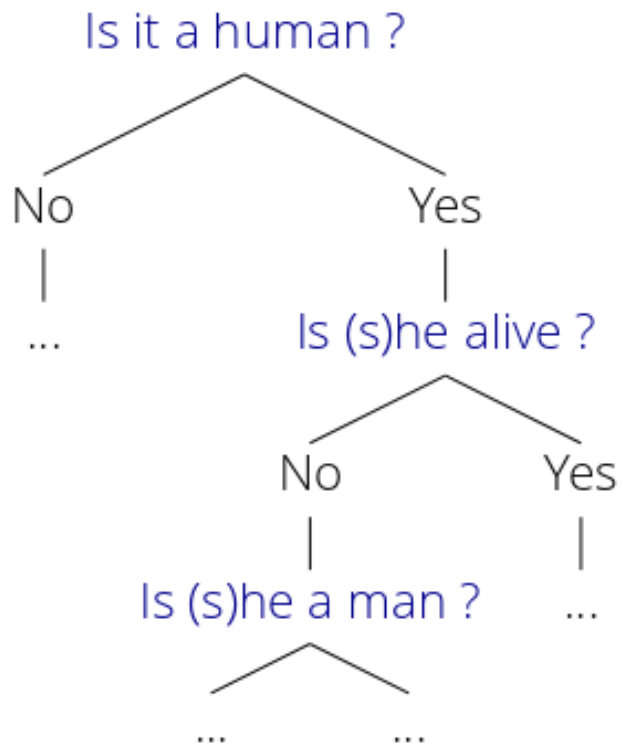
[7] http:

Binary Decision Tree[8]



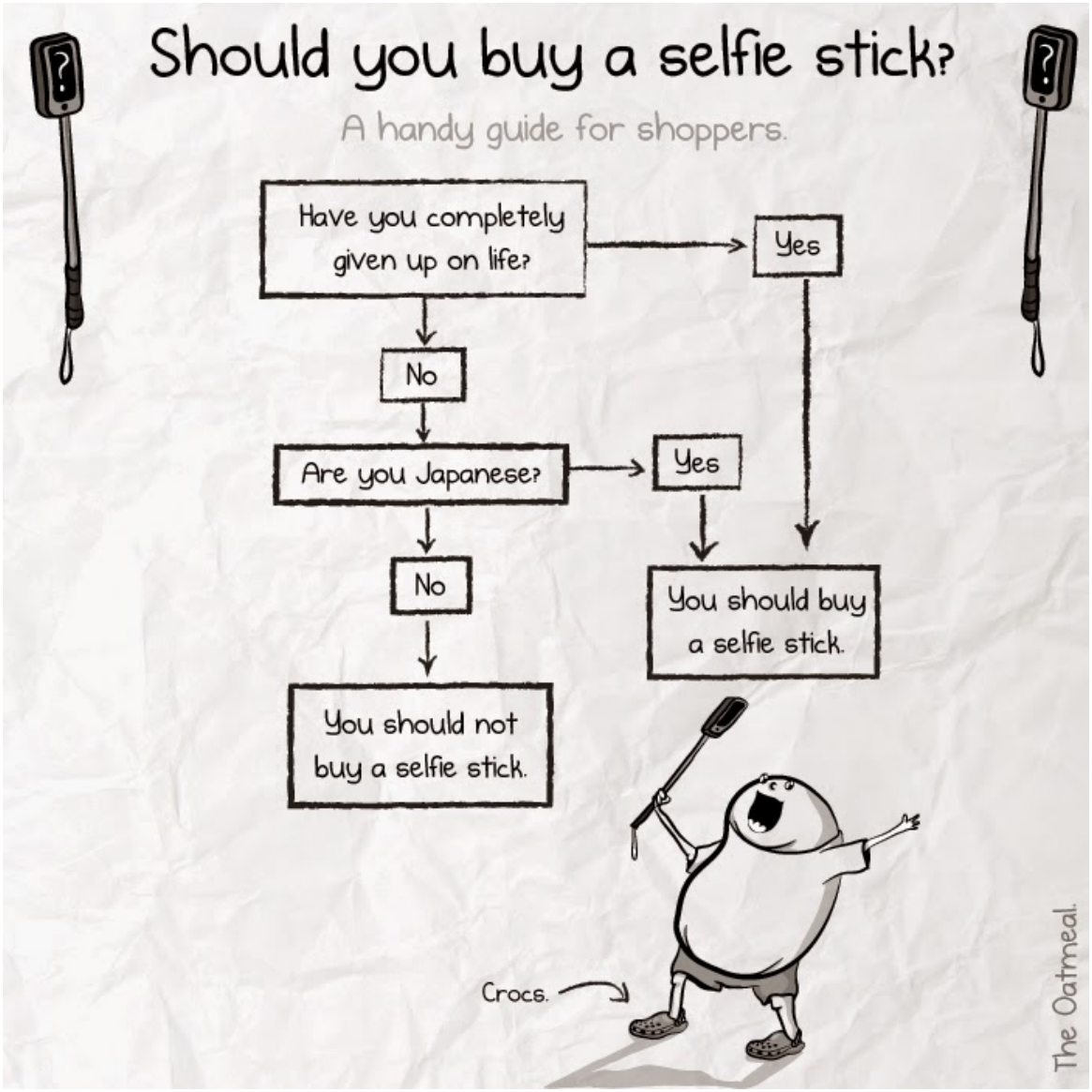
[8] http:

20 Question Intuition[9]



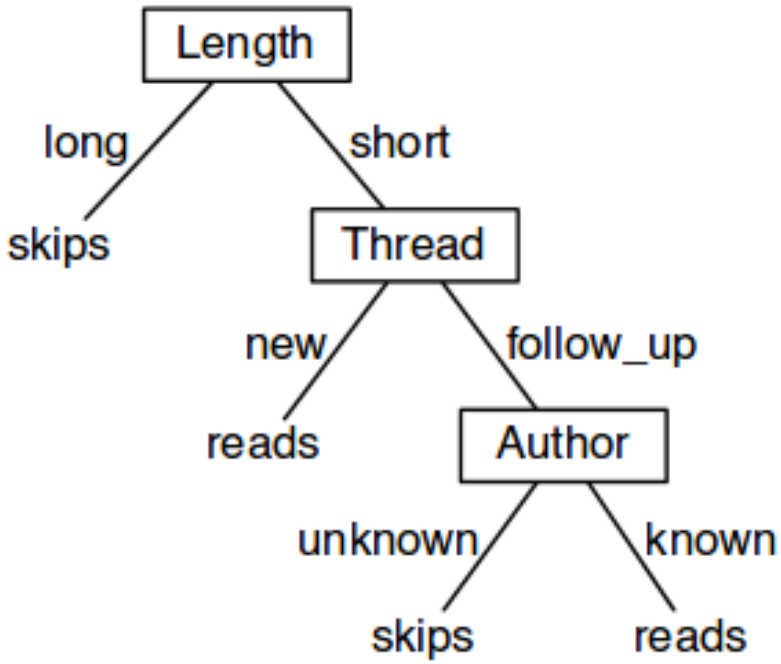
[9] http://www.idiap.ch/~fleuret/files/EE613/EE613-slides-6.pdf

Decision Tree for Selfie Stick[10]



[10] The Oatmeal Comics

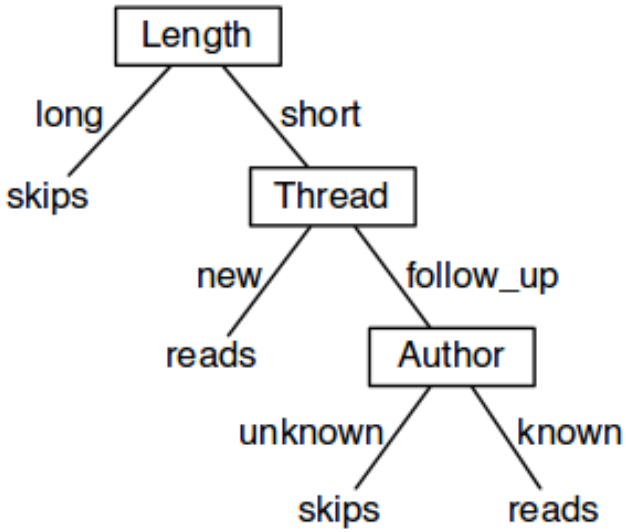
Decision Trees and Rules[11]



[11] http://artint.info/slides/ch07/lect3.pdf

Decision Trees and Rules[12]

- long $\rightarrow$ skips
- short $\wedge$ new $\rightarrow$ reads
- short $\wedge$ follow Up $\wedge$ known $\rightarrow$ reads
- short $\wedge$ follow Up $\wedge$ unknown $\rightarrow$ skips



[12] http://artint.info/slides/ch07/lect3.pdf

Building Decision Trees Intuition[13]

Horsepower	Weight	Mileage
95	low	low
90	low	low
70	low	high
86	low	high
76	high	low
88	high	low

Table: Car Mileage Prediction from 1971

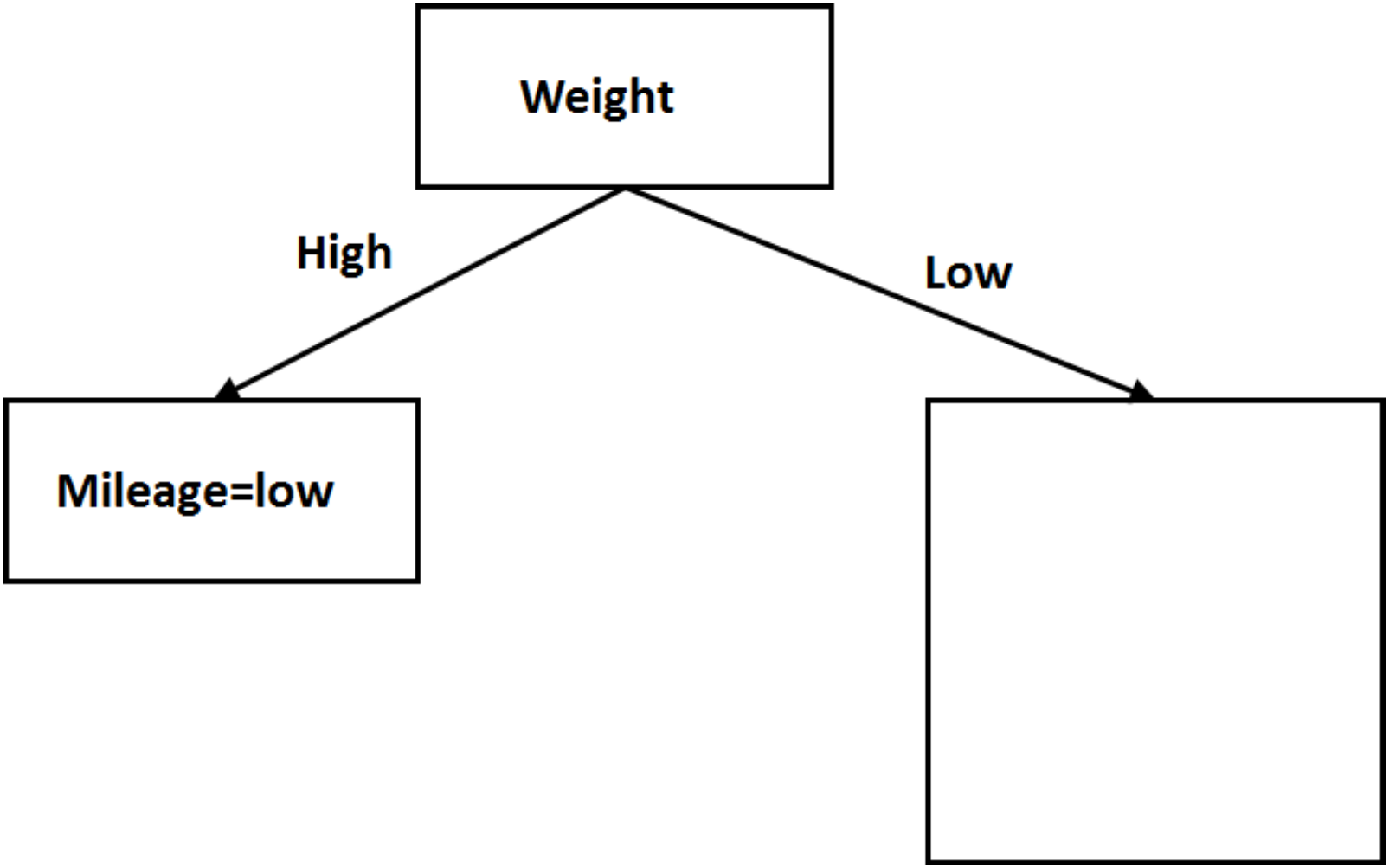
[13] http://spark-summit.org/wp-content/uploads/2014/07/Scalable-Distributed-Decision-Trees-in-Spark-Made-Das-Sparks-Talwalkar.pdf

Building Decision Trees Intuition

Horsepower	Weight	Mileage
95	low	low
90	low	low
70	low	high
86	low	high
76	high	low
88	high	low

Table: Car Mileage Prediction from 1971

Building Decision Trees Intuition



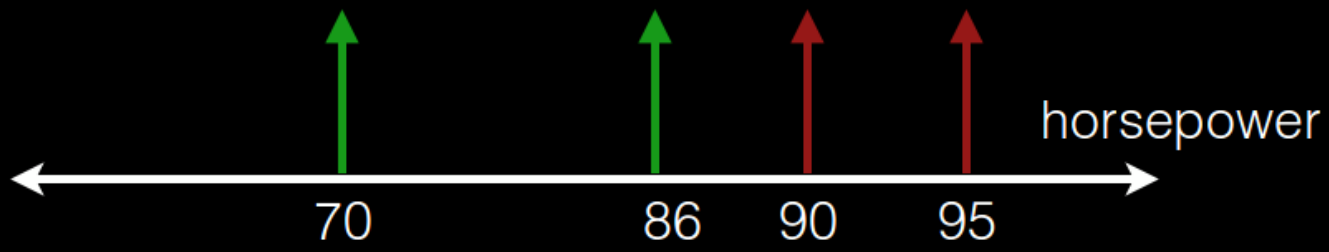
Building Decision Trees Intuition

Horsepower	Weight	Mileage
95	low	low
90	low	low
70	low	high
86	low	high

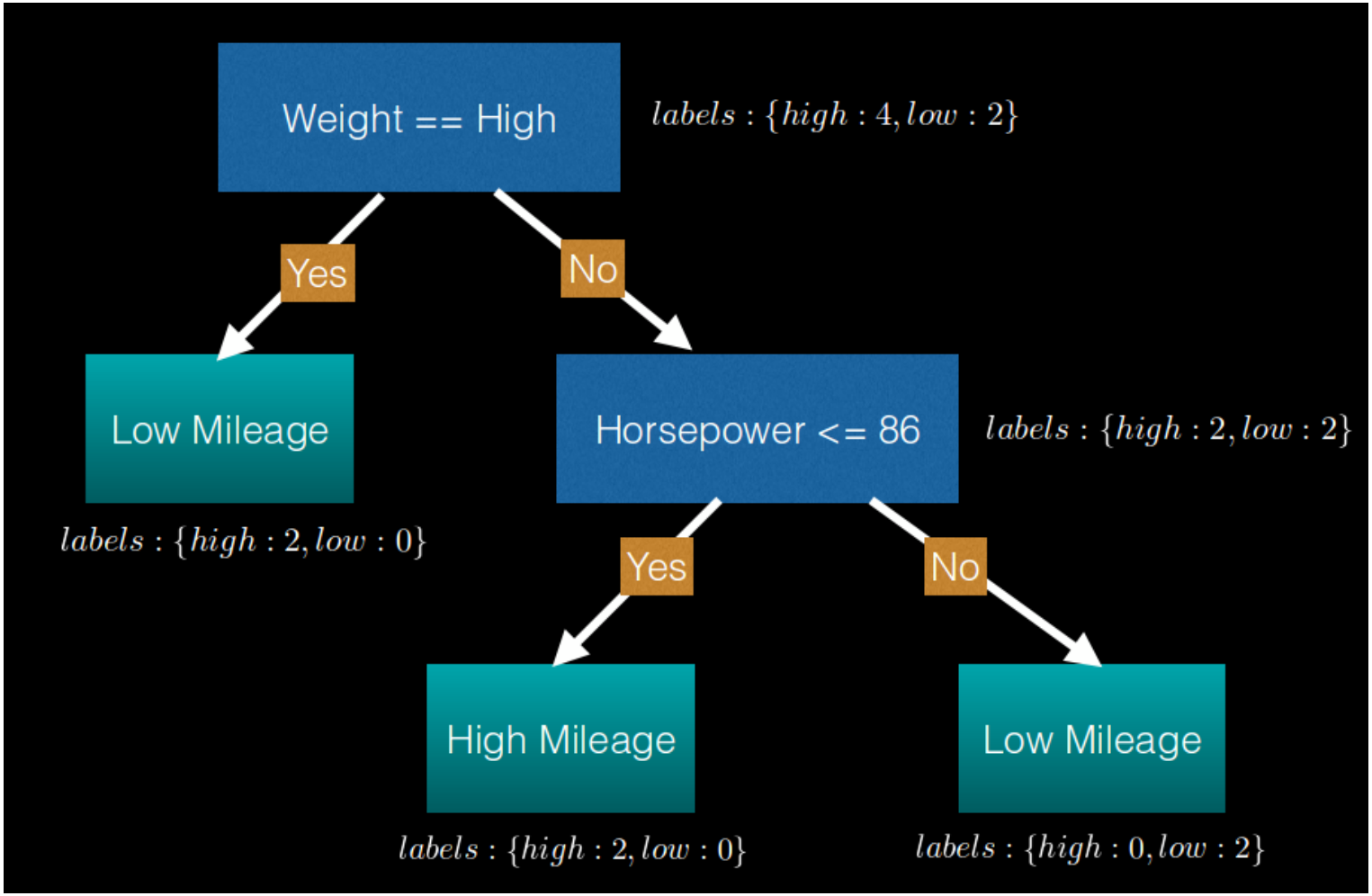
Table: Car Mileage Prediction from 1971

Building Decision Trees Intuition

horsepower	weight	mileage
95	low	low
90	low	low
70	low	high
86	low	high



Building Decision Trees Intuition



Building Decision Trees Intuition

Prediction:

horsepower	weight	mileage prediction
90	high	
80	low	
70	high	

Building Decision Trees Intuition

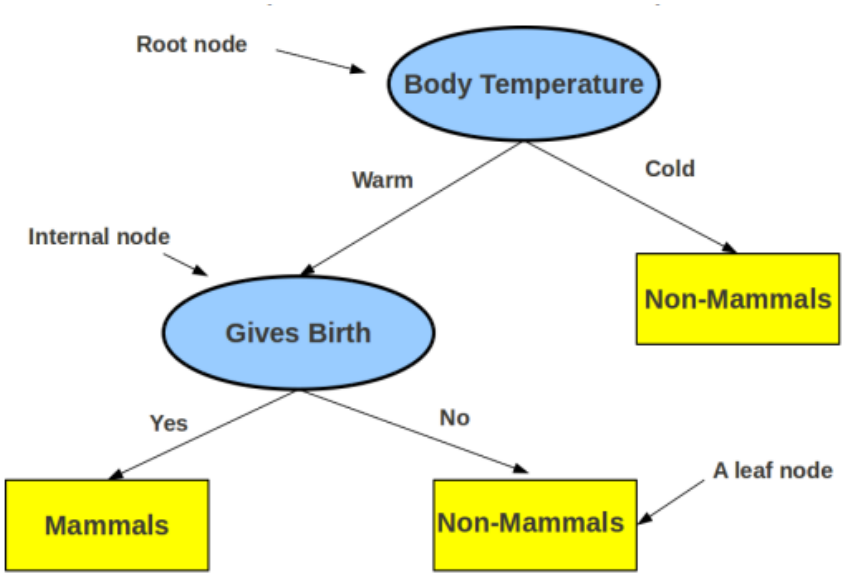
Prediction:

horsepower	weight	mileage prediction	
90	high	low	Correct!
80	low	low	Correct!
70	high	high	Wrong!

Learning Decision Trees

Decision Trees

- Defined by a **hierarchy** of rules (in form of a tree)



- Rules form the internal nodes of the tree (topmost internal node = root)
- Each rule (internal node) tests the value of some property the data
- Leaf nodes make the prediction

Decision Tree Learning

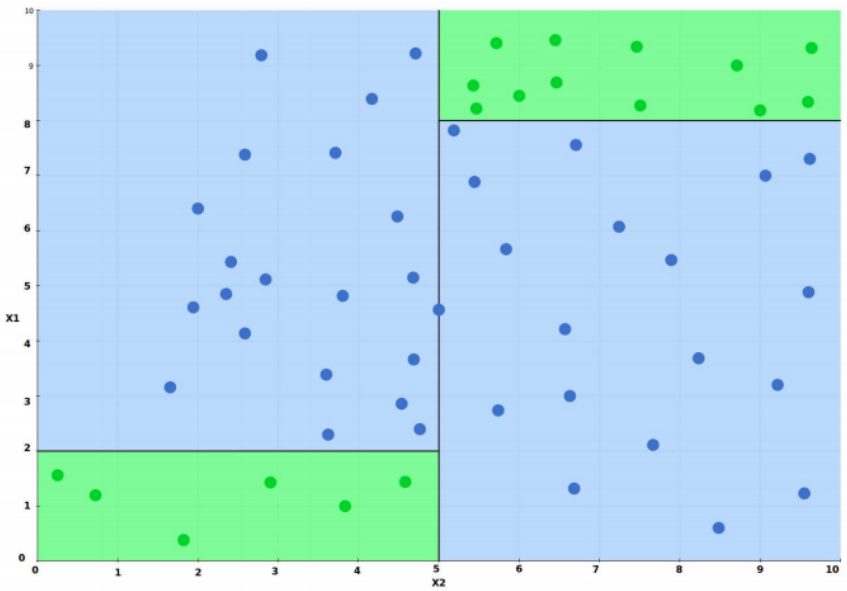
Objective:

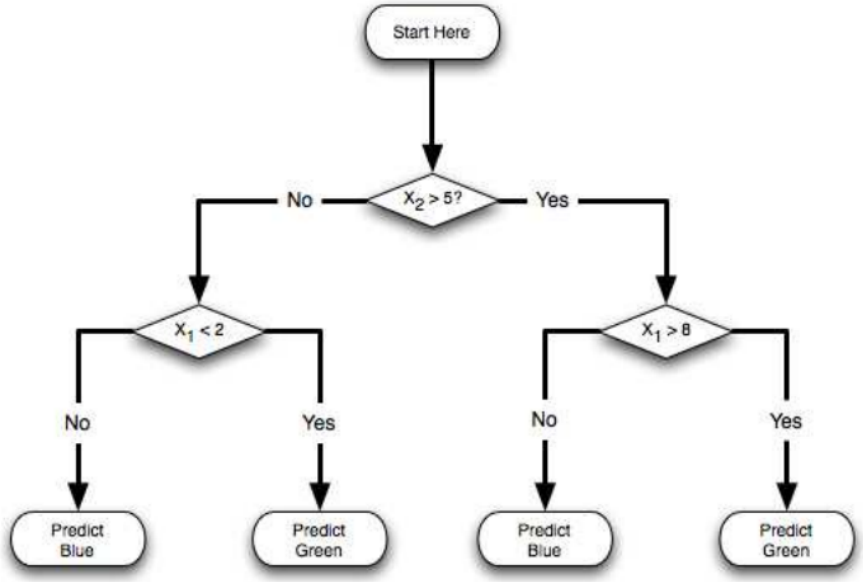
- Use the training data to construct a good decision tree
- Use the constructed Decision tree to predict labels for test inputs

Decision Tree Learning

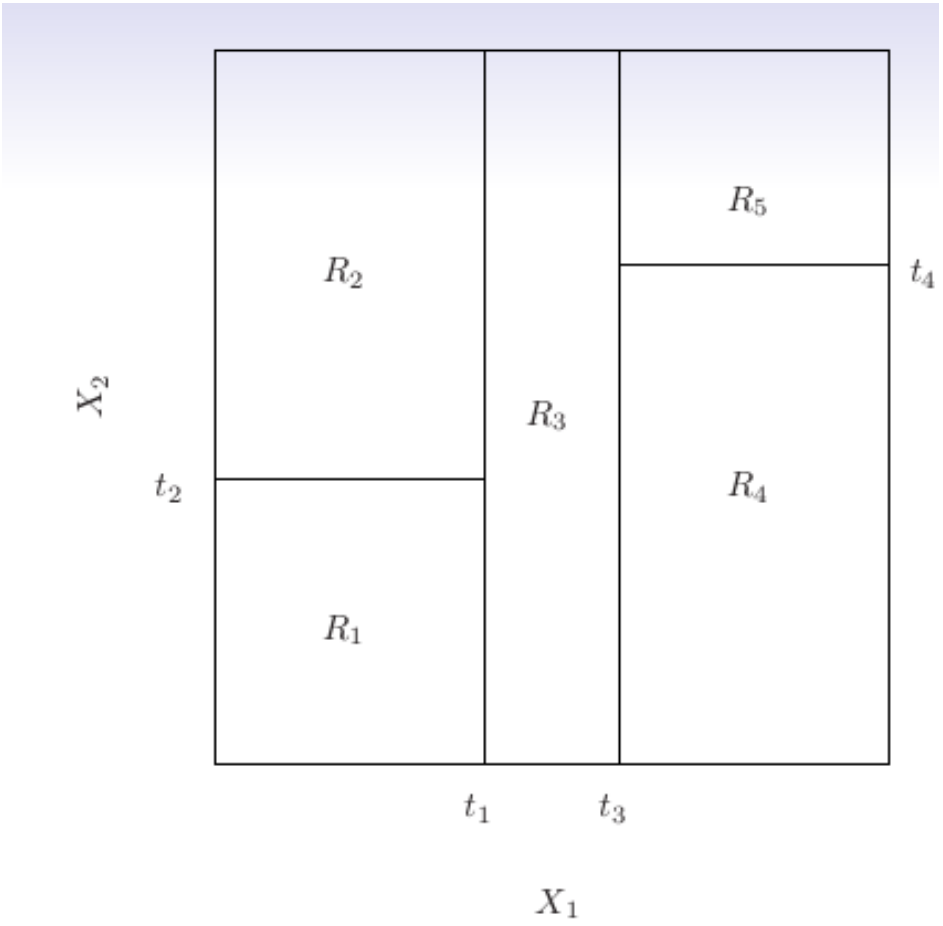
- Identifying the region (blue or green) a point lies in
  - A classification problem (blue vs green)
  - Each input has 2 features: co-ordinates $\{x_1, x_2\}$ in the 2D plane
- Once learned, the decision tree can be used to predict the region (blue/green) of a new test point

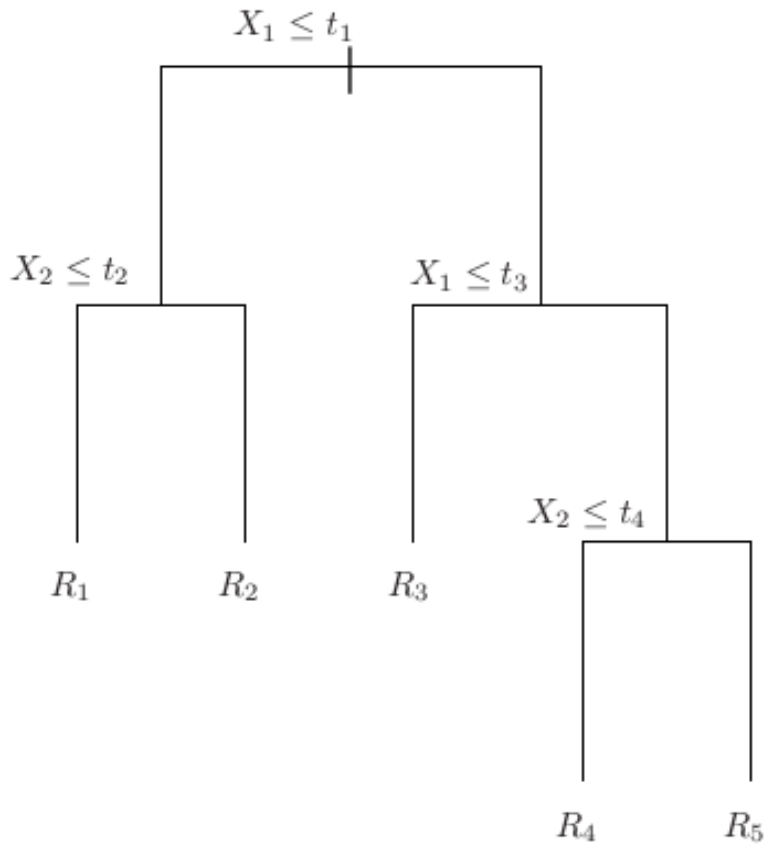
Decision Tree Learning





Expressiveness of Decision Trees

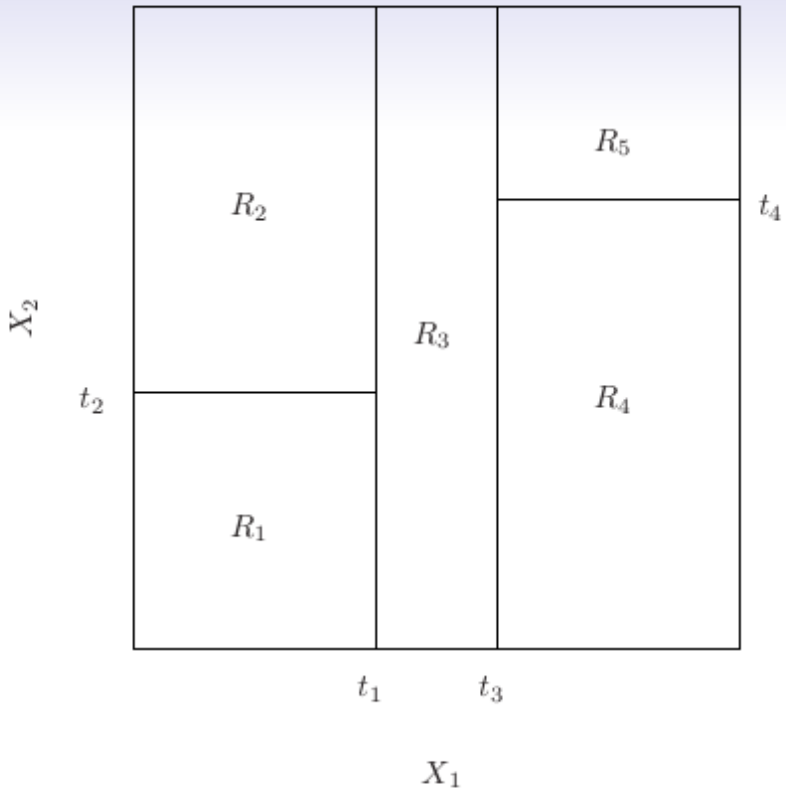


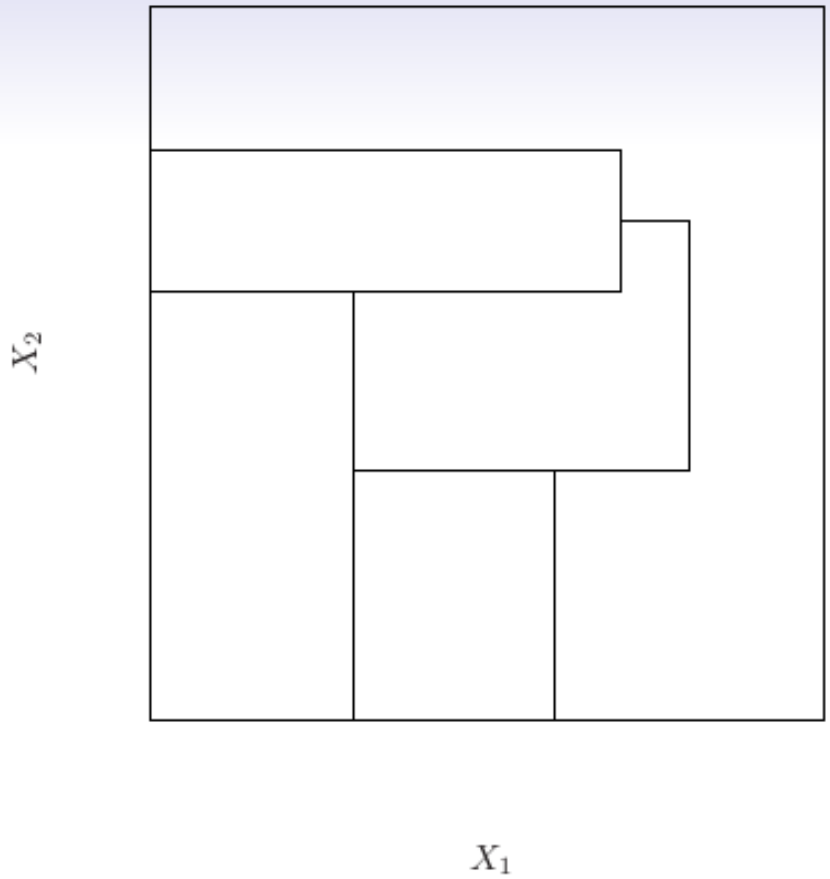


Expressiveness of Decision Trees

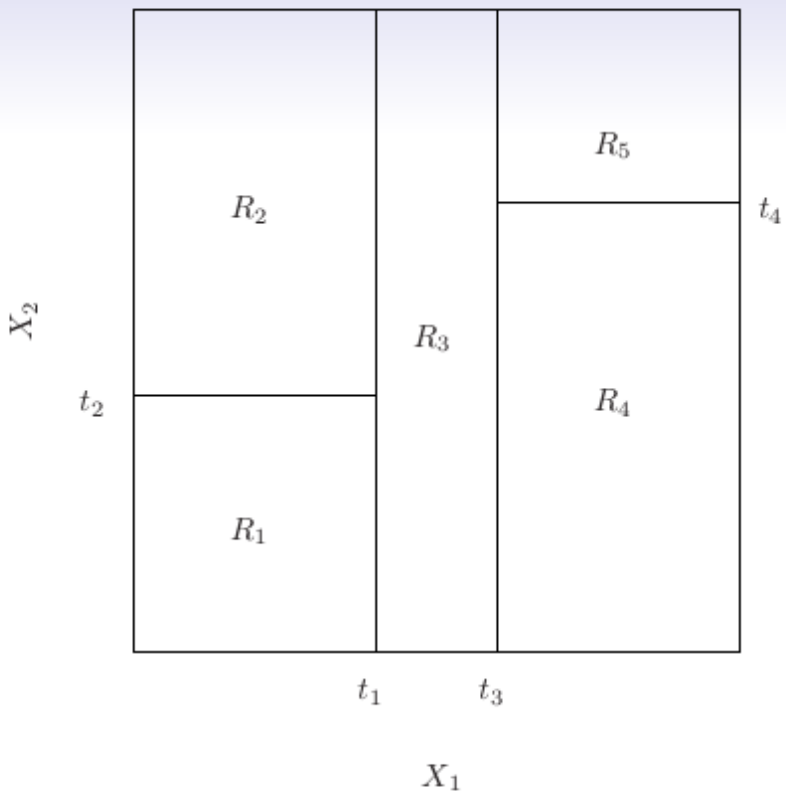
- Decision tree divides feature space into **axis-parallel** rectangles
- Each rectangle is labelled with one of the C classes
- Any partition of feature space by recursive binary splitting can be simulated by Decision Trees

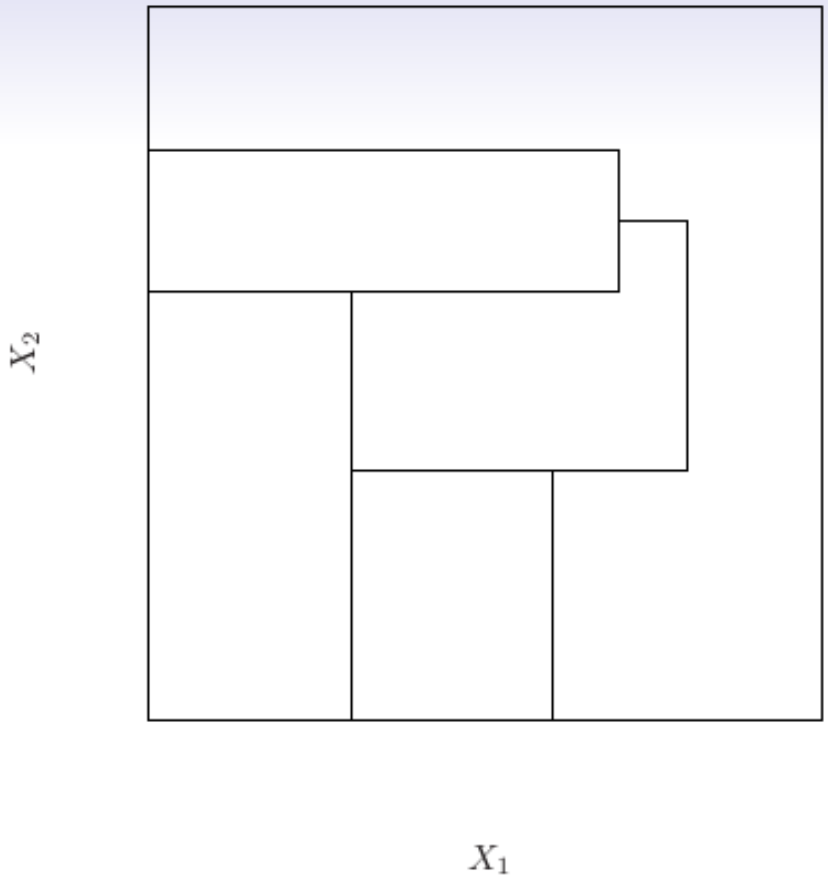
Expressiveness of Decision Trees





Expressiveness of Decision Trees



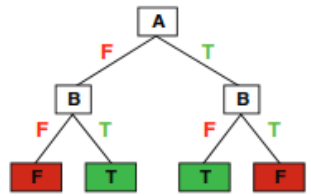


Feature space on left can be simulated by Decision tree but not the one on right.

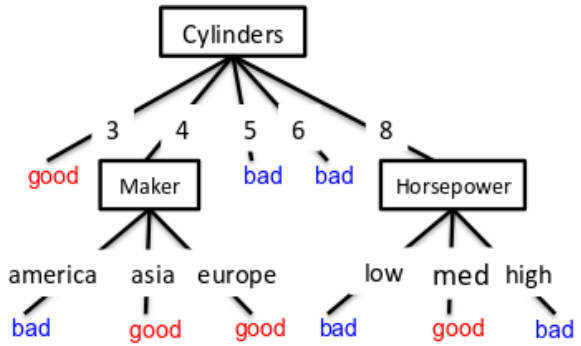
Expressiveness of Decision Tree

- Can express any logical function on input attributes
- Can express **any** boolean function
- For boolean functions, path to leaf gives truth table row
- Could require exponentially many nodes
- $cyl = 3 \vee (cyl = 4 \wedge (maker = asia \vee maker = europe)) \vee \ldots$

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



(Figure from Stuart Russell)



Hypothesis Space

- Exponential search space wrt set of attributes
- If there are d boolean attributes, then the search space has 2^{2^d} trees
  - If $d = 6$, then it is approximately 2×10^{18}
  - If there are d boolean attributes, each truth table has 2^d rows
  - Hence there must be 2^{2^d} truth tables that can take all possible variations
- NP-Complete to find optimal decision tree
- Idea: Use greedy approach to find a locally optimal tree

Decision Tree Learning Algorithms

- 1966: Hunt and colleagues from Psychology developed first known algorithm for human concept learning
- 1977: Breiman, Friedman and others from Statistics developed CART
- 1979: Quinlan developed proto-ID3
- 1986: Quinlan published ID3 paper
- 1993: Quinlan's updated algorithm C4.5
- 1980's and 90's: Improvements for handling noise, continuous attributes, missing data, non-axis parallel DTs, better heuristics for pruning, overfitting, combining DTs

Decision Tree Learning Algorithms

Main Loop:

1. Let A be the "best" decision attribute for next node
2. Assign A as decision attribute for node
3. For each value of A, create a new descendent of node
4. Sort training examples to leaf nodes
5. If training examples are perfectly classified, then STOP else iterate over leaf nodes

Decision Tree Learning

- Greedy Approach: Build tree, top-down by choosing one attribute at a time
- Choices are locally optimal and may or may not be globally optimal
- Major issues
  - Selecting the next attribute
  - Determining termination condition

Termination Condition

Stop expanding a node further when:

Termination Condition

Stop expanding a node further when:

- It consist of examples all having the same label
- Or we run out of features to test!

Recursive Algorithm for Learning Decision Trees

DT(*Examples*, *Labels*, *Features*):

- If all examples are positive, return a single node tree *Root* with label = +
- If all examples are negative, return a single node tree *Root* with label = -
- If all features exhausted, return a single node tree *Root* with majority label
- Otherwise, let F be the feature having the highest information gain
- $Root \leftarrow F$
- For each possible value f of F
  - Add a tree branch below *Root* corresponding to the test $F = f$
  - Let $Examples_f$ be the set of examples with feature F having value f
  - Let $Labels_f$ be the corresponding labels
  - If $Examples_f$ is empty, add a leaf node below this branch with label = most common label in *Examples*
  - Otherwise, add the following subtree below this branch:

    $$\mathrm{DT}(Examples_f, Labels_f, Features - \{F\})$$

  - Note: *Features* - $\{F\}$ removes feature F from the feature set *Features*

Summary

Major Concepts:

- Geometric interpretation of Classification
- Decision trees

Slide Material References

- Slides from ISLR book
- Slides by Piyush Rai
- Slides for Chapter 4 from "Introduction to Data Mining" book by Tan, Steinbach, Kumar
- See also the footnotes