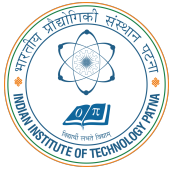
# Introduction to Data Science

## Decision Trees




**Arijit Mondal**

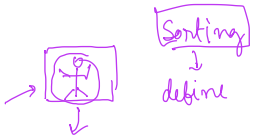
**Dept. of Computer Science & Engineering**

**Indian Institute of Technology Patna**

arijit@iitp.ac.in

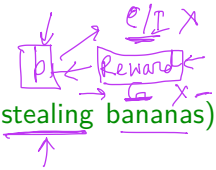
# Learning

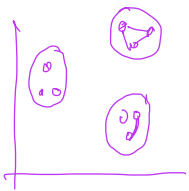
- An agent is learning if it improves its performance on future tasks after making observation about the world
- Why would an agent learn?
  - Designers cannot anticipate all possible situations
  - Designers cannot anticipate all changes over time
  - Sometime, people have no idea how to program a solution
- Inductive learning - Learning a general function or rule from specific input-output pairs
- Analytical / deductive learning - Going from a known general rule to a new rule that is logically entailed



# Paradigms of learning

- These are based on the types of feedback
- Supervised learning
  - Both inputs and outputs are given
  - The outputs are typically provided by a friendly teacher
- Reinforcement learning → control / Cranes
  - The agent receives some evaluation of its actions (such as a fine for stealing bananas), but is not told the correct action (such as how to buy bananas)
- Unsupervised learning
  - The agent can learn relationships among its percepts, and the trend with time



# Supervised learning

- A set of labeled examples $\langle x_1, x_2, \ldots, x_n, y \rangle$
  - $x_i$ are input variables
  - $y$ output variable
- Need to find a function $f: X_1 \times X_2 \times \ldots X_n \rightarrow Y$
- Goal is to minimize error/loss function
  - Like to minimize over all dataset
  - We have limited dataset

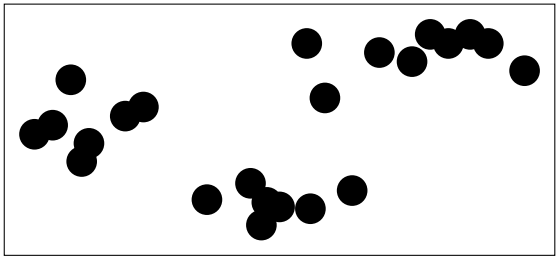
$f: X_i \rightarrow C$

$X_j \rightarrow IC$

Quiz-4
21st April
2pm

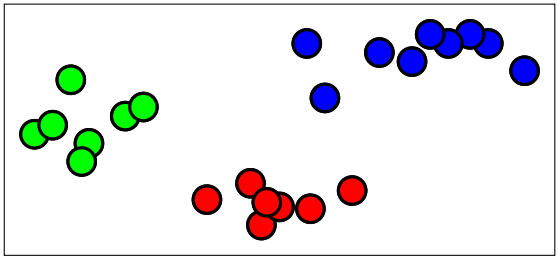
# Unsupervised learning

- Learns useful properties of the structure of data set
- Unlabeled data
  - Tries to learn entire probability distribution that generated the dataset
  - Examples
    - Clustering, dimensionality reduction PCA



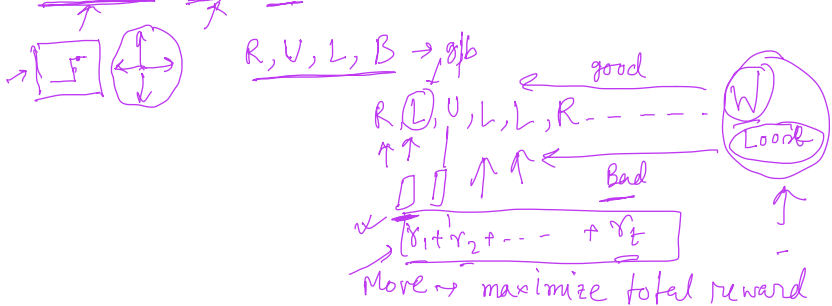
# Unsupervised learning

- Learns useful properties of the structure of data set
- Unlabeled data
  - Tries to learn entire probability distribution that generated the dataset
  - Examples
    - Clustering, dimensionality reduction

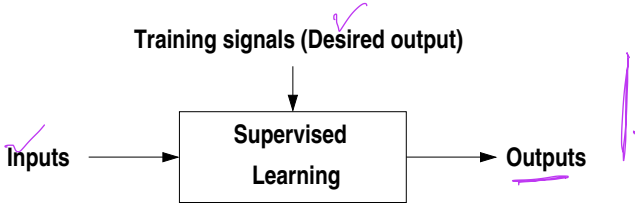


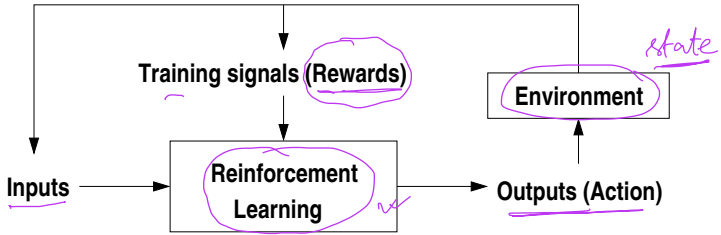
# Reinforcement learning

- Set of actions that the learner will make in order to maximize its profit
- Action may not only affect the next situation but also subsequent situation
  - Trial and error search
  - Delayed reward
- A learning agent is interacting with environment to achieve a goal
- Agent needs to have idea of state so that it can take right action
- Three key aspects — **observation, action, goal**



# Reinforcement vs supervised learning



# Decision trees

- A decision tree takes as input an object or situation described by a set of properties, and outputs a yes/no "decision"
- A list of variables which potentially affect the decision on whether to wait for a table at a restaurant.
  - Alternate: whether there is a suitable alternative restaurant
  - Lounge: whether the restaurant has a lounge for waiting customers
  - Fri/Sat: true on Fridays and Saturdays
  - Hungry: whether we are hungry
  - Patrons: how many people are in it (None, Some, Full)
  - Price: the restaurant's rating (*, **, ***)
  - Raining: whether it is raining outside
  - Reservation: whether we made a reservation
  - Type: the kind of restaurant (Indian, Chinese, Thai, Fastfood)
  - WaitEstimate: 0-10 mins, 10-30, 30-60, >60.

# Observations

| Example | Input Attributes | | | | | | | | | | Goal |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | *Alt* | *Bar* | *Fri* | *Hun* | *Pat* | *Price* | *Rain* | *Res* | *Type* | *Est* | *WillWait* |
| $\mathbf{x}_1$ | *Yes* | *No* | *No* | *Yes* | *Some* | *$$$* | *No* | *Yes* | *French* | *0–10* | $_1 = Yes$ |
| $\mathbf{x}_2$ | *Yes* | *No* | *No* | *Yes* | *Full* | *$* | *No* | *No* | *Thai* | *30–60* | $_2 = No$ |
| $\mathbf{x}_3$ | *No* | *Yes* | *No* | *No* | *Some* | *$* | *No* | *No* | *Burger* | *0–10* | $_3 = Yes$ |
| $\mathbf{x}_4$ | *Yes* | *No* | *Yes* | *Yes* | *Full* | *$* | *Yes* | *No* | *Thai* | *10–30* | $_4 = Yes$ |
| $\mathbf{x}_5$ | *Yes* | *No* | *Yes* | *No* | *Full* | *$$$* | *No* | *Yes* | *French* | *60* | $_5 = No$ |
| $\mathbf{x}_6$ | *No* | *Yes* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Italian* | *0–10* | $_6 = Yes$ |
| $\mathbf{x}_7$ | *No* | *Yes* | *No* | *No* | *None* | *$* | *Yes* | *No* | *Burger* | *0–10* | $_7 = No$ |
| $\mathbf{x}_8$ | *No* | *No* | *No* | *Yes* | *Some* | *$$* | *Yes* | *Yes* | *Thai* | *0–10* | $_8 = Yes$ |
| $\mathbf{x}_9$ | *No* | *Yes* | *Yes* | *No* | *Full* | *$* | *Yes* | *No* | *Burger* | *60* | $_9 = No$ |
| $\mathbf{x}_{10}$ | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$$$* | *No* | *Yes* | *Italian* | *10–30* | $_{10} = No$ |
| $\mathbf{x}_{11}$ | *No* | *No* | *No* | *No* | *None* | *$* | *No* | *No* | *Thai* | *0–10* | $_{11} = No$ |
| $\mathbf{x}_{12}$ | *Yes* | *Yes* | *Yes* | *Yes* | *Full* | *$* | *No* | *No* | *Burger* | *30–60* | $_{12} = Yes$ |

binary

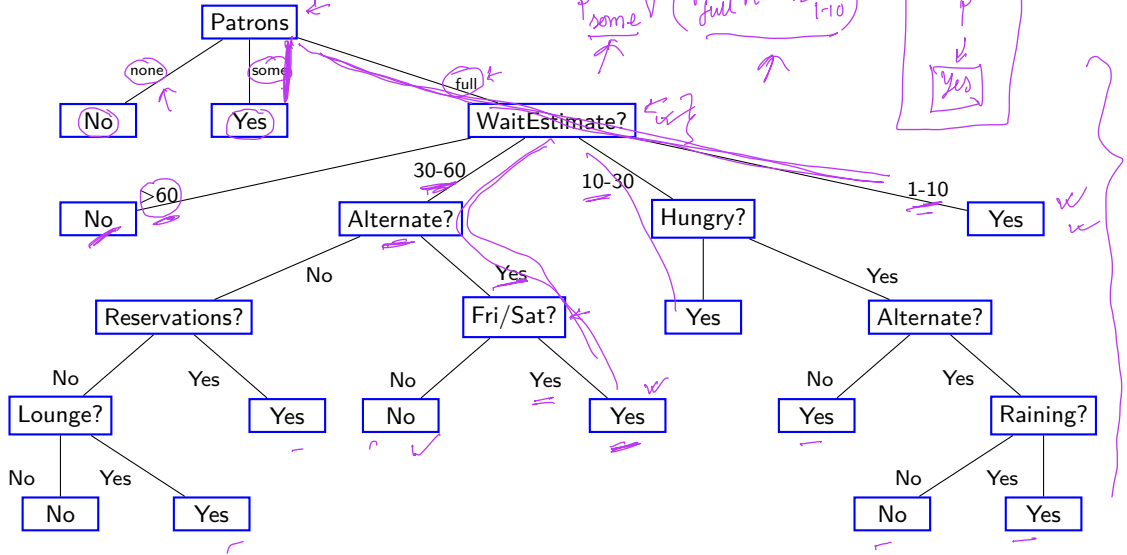
$2^{10}$

→ Yes/No



Image source: AI by Russel & Norvig

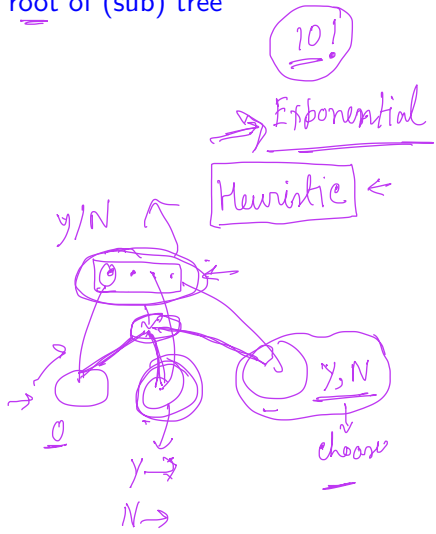
# Sample decision tree



# Decision Tree Learning

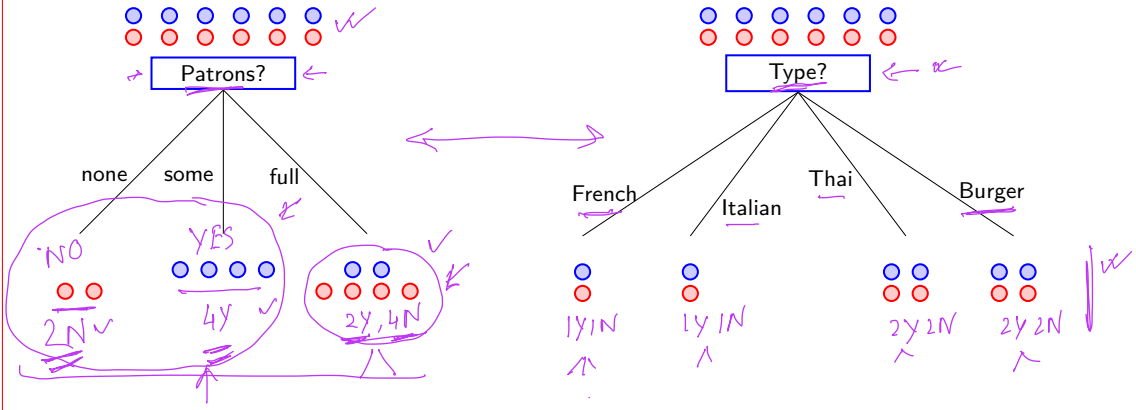
- Aim: find a small tree consistent with the training examples
- Idea: (recursively) choose "most significant" attribute as root of (sub) tree

1. pick an attribute to split at a non-terminal node
2. split examples into groups based on attribute value
3. for each group:
   A. if no examples - return majority from parent
   B. else if all examples in same class - return class
   C. else loop to step 1



# Choosing an attribute

**Idea: A good attribute splits the examples into subsets that are (ideally) "all positive" or "all negative"**



# Attribute selection

- Information content (Entropy): $I(P(v_1), \ldots, P(v_n)) = \sum_{j=1}^{n} -P(v_j) \log_2 P(v_j)$
- For a training set containing $p$ positive examples and $n$ negative examples:

$$I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

- A chosen attribute $A$ divides the training set $E$ into subsets $E_1, \ldots, E_v$ according to their values for $A_i$ where $A$ has $v$ distinct values

$$\text{remainder}(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p+n} I\left(\frac{p_i}{p_i+n_i}, \frac{n_i}{p_i+n_i}\right)$$

- Information gain (IG) or reduction in entropy $IG(A) = I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) - \text{remainder}(A)$
- Choose the attribute with the largest IG

# Information gain: example

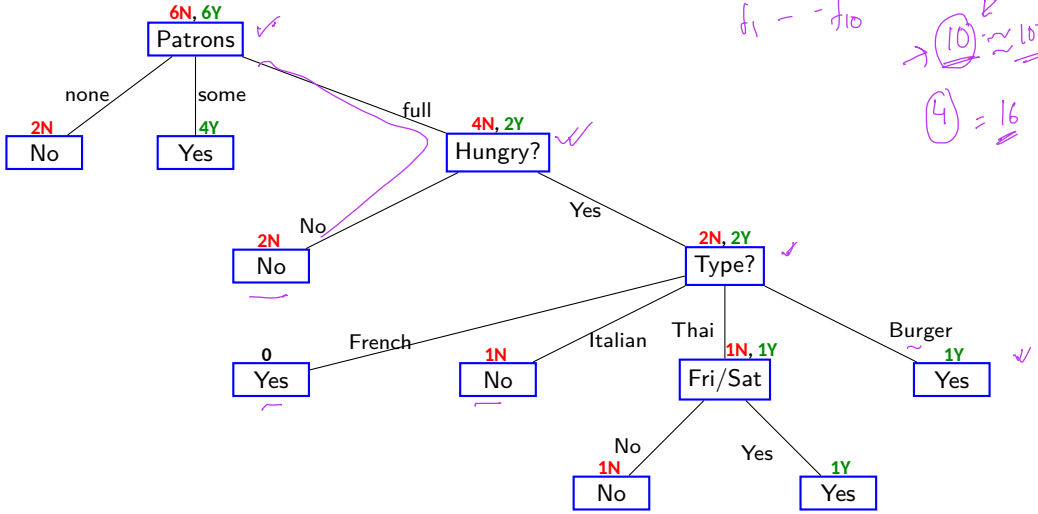
- For the training set $p = n = 6$, $I(\frac{6}{12}, \frac{6}{12}) = 1$ bit

  $IG(Patrons) = 1 - [\frac{2}{12} I(0,1) + \frac{4}{12} I(1,0) + \frac{6}{12} I(\frac{2}{6}, \frac{4}{6})] = 0.0541$

  $IG(Type) = 1 - [\frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4})] = 0$

- Patrons will be selected

# Final decision tree



# A good tree

- Not too small: need to handle important but possibly subtle distinctions in data
- Not too big:
  - Computational efficiency (avoid redundant, spurious attributes)
  - Avoid over-fitting training examples