CS2109S: Introduction to AI and Machine Learning

Lecture 4:
Intro to Machine Learning & Decision Trees

8 September 2023

Overview

You are here

"Classical" AI	Machine Learning	
Search Algorithms • Uninformed search: BFS, DFS, UCS, IDS • Informed search: Greedy best-first, A* • Local Search: hill-climbing, SA, Beam, GA • Adversarial search: Minimax, Alpha-beta	**ML Models & Techniques** • Decision Trees • Linear/Logistic Regression • Support Vector Machines • Neural Networks	**Miscellaneous** • Unsupervised Learning • AI & Ethics
Applied CS2040S, CS1231 Python	**Applied Linear Algebra, Calculus, Statistics & Probabilities** Numpy, Scikit-learn, PyTorch	

Outline

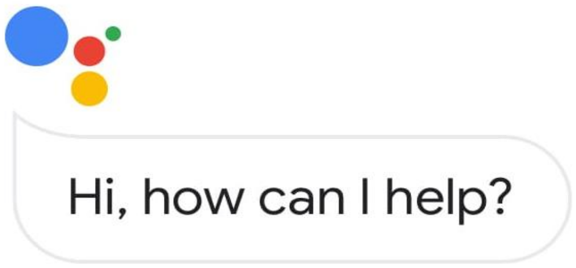
- Machine Learning
  - What is ML?
  - Types of Feedback
  - Supervised Learning
- Performance Measure
  - Regression: mean squared error, mean absolute error
  - Classification: correctness, accuracy, confusion matrix, precision, recall, F1
- Decision Trees
  - Decision Tree Learning (DTL)
  - Entropy and Information Gain
  - Different types of attributes
  - Pruning
  - Ensemble Methods

Outline

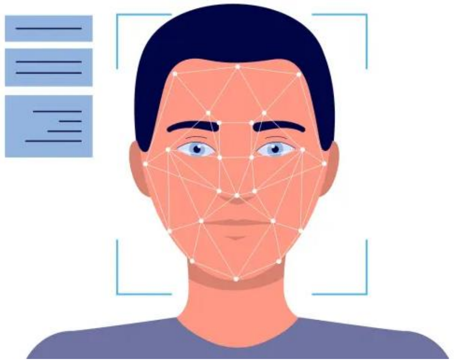
- **Machine Learning**
  - What is ML?
  - Types of Feedback
  - Supervised Learning
- Performance Measure
  - Regression: mean squared error, mean absolute error
  - Classification: correctness, accuracy, confusion matrix, precision, recall, F1
- Decision Trees
  - Decision Tree Learning (DTL)
  - Entropy and Information Gain
  - Different types of attributes
  - Pruning
  - Ensemble Methods

What is Machine Learning? Applications?

ChatGPT



Credit: Google



Credit: VentureBeat

Credit: Skyfish

Credit: Futuremind

What is Machine Learning?

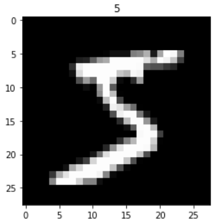
"The field of study that gives computers the **ability to learn without being explicitly programmed**"

- Arthur Samuel (1959)

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its **performance** at tasks in T, as measured by P, **improves with experience** E."

- Tom Mitchell

Why Machine Learning?



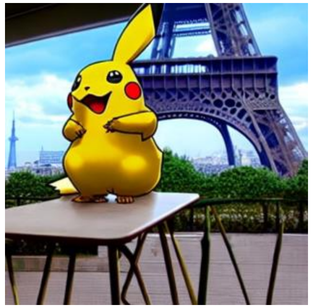
→ Function → 5

"What is AI?" → Function → "AI is …"

Credit: Chess.com

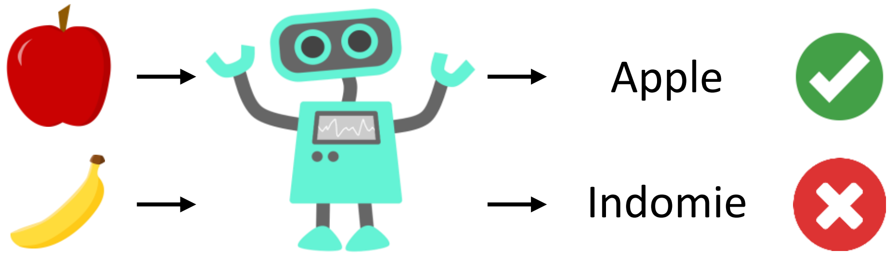
→ Function → Bb6 (chess move)

"Pikachu in the Eiffel tower" → Function →



How do we write this Function?

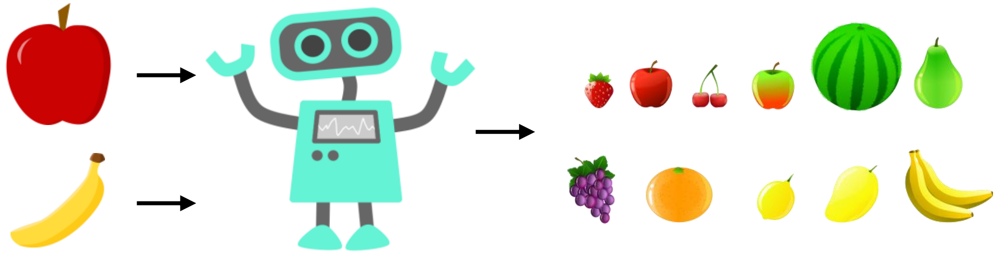
Types of Feedback



Supervised

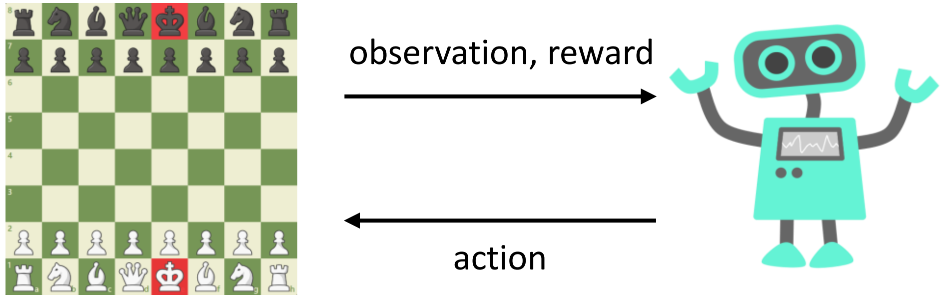
“Teacher’s feedback”

Semi/Weakly Supervised



Unsupervised

“No feedback”



Reinforcement

“Trial and error”

Supervised Learning

Learns from being given the **right answers**: $\boldsymbol{X} \rightarrow \boldsymbol{Y}$

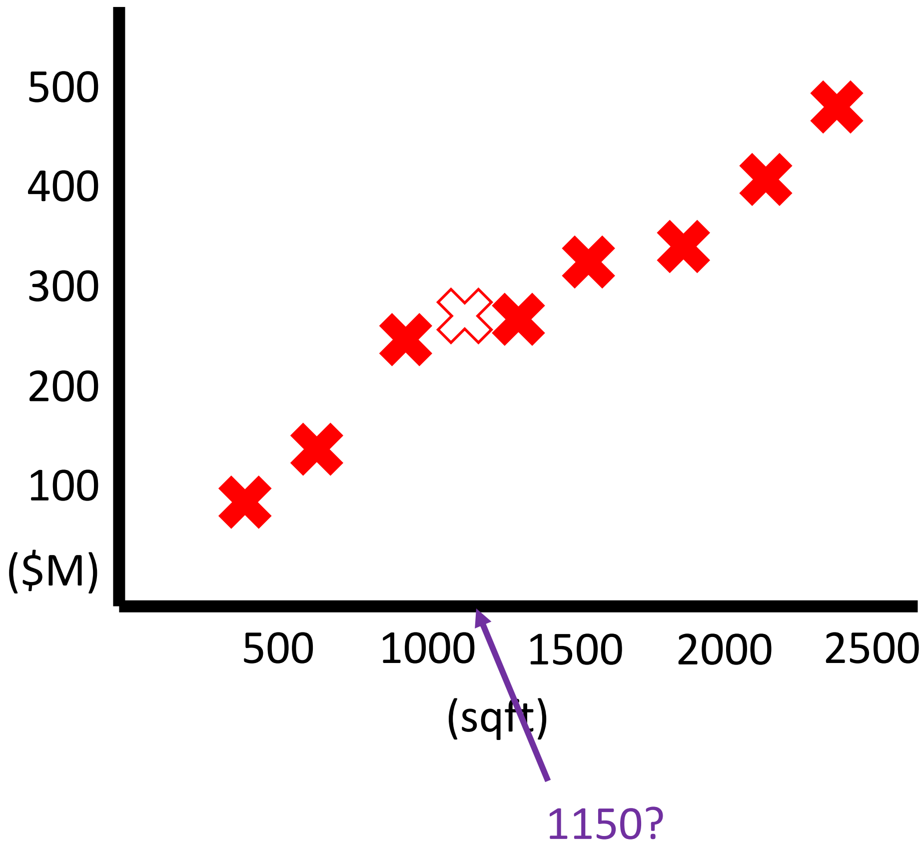
Input (X)	**Output (Y)**	**Application**
Email	Spam?	Spam filtering
Audio	Text transcripts	Speech recognition
English	Chinese	Machine translation
Image	Position of cars	Self-driving car

Types:

- **Regression**: predict continuous output (e.g., temperature: 0.567)
- **Classification**: predict discrete output (e.g., animal: cat vs dog)

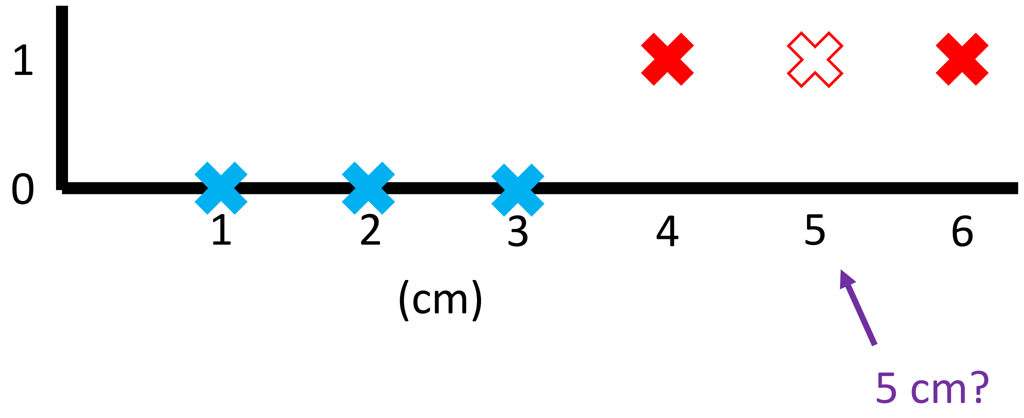
Supervised Learning: Regression

Housing price prediction



Supervised Learning: Classification

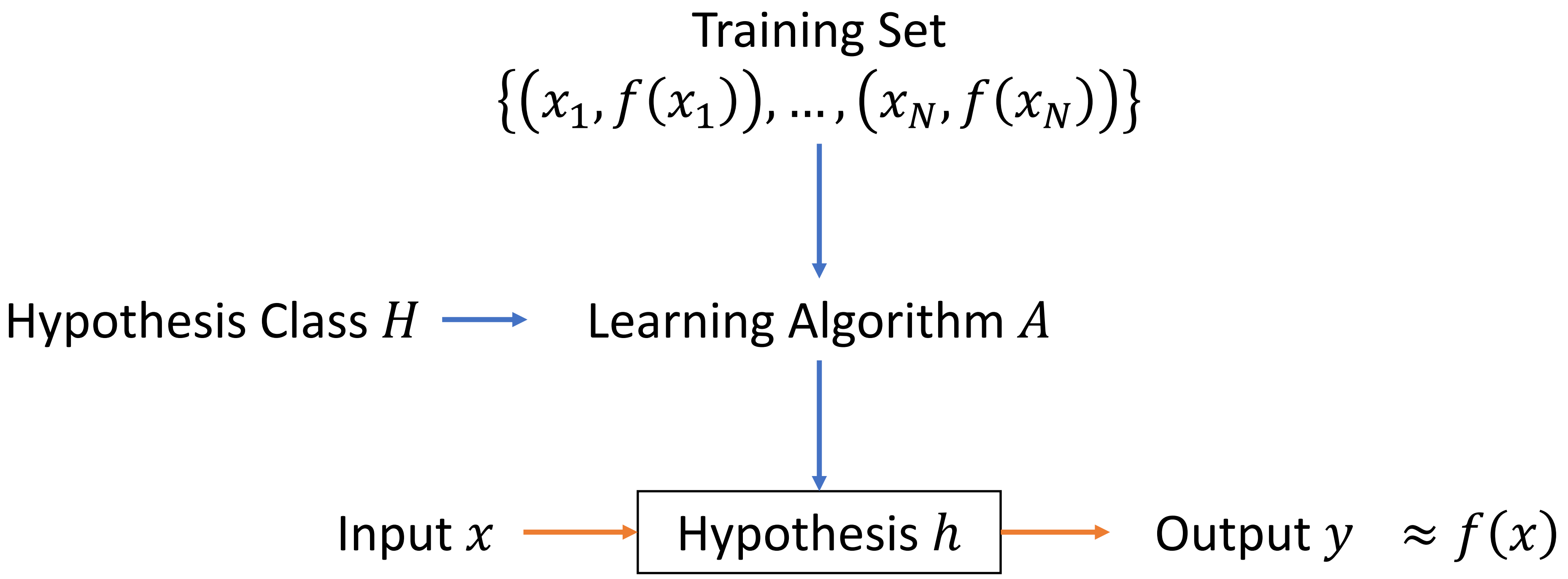
- Cancer prediction: benign (0), malignant (1)



Formalism

- In supervised learning, we assume that y is generated by a **true mapping function** $f: x \to y$.
- We want to find a **hypothesis** $h: x \to \hat{y}$ (from a **hypothesis class** H) s.t. $h \approx f$ given a **training set** $\{(x_1, f(x_1)), \ldots, (x_N, f(x_N))\}$.
  - We use a **learning algorithm** to find this hypothesis

Formalism (Illustrated)



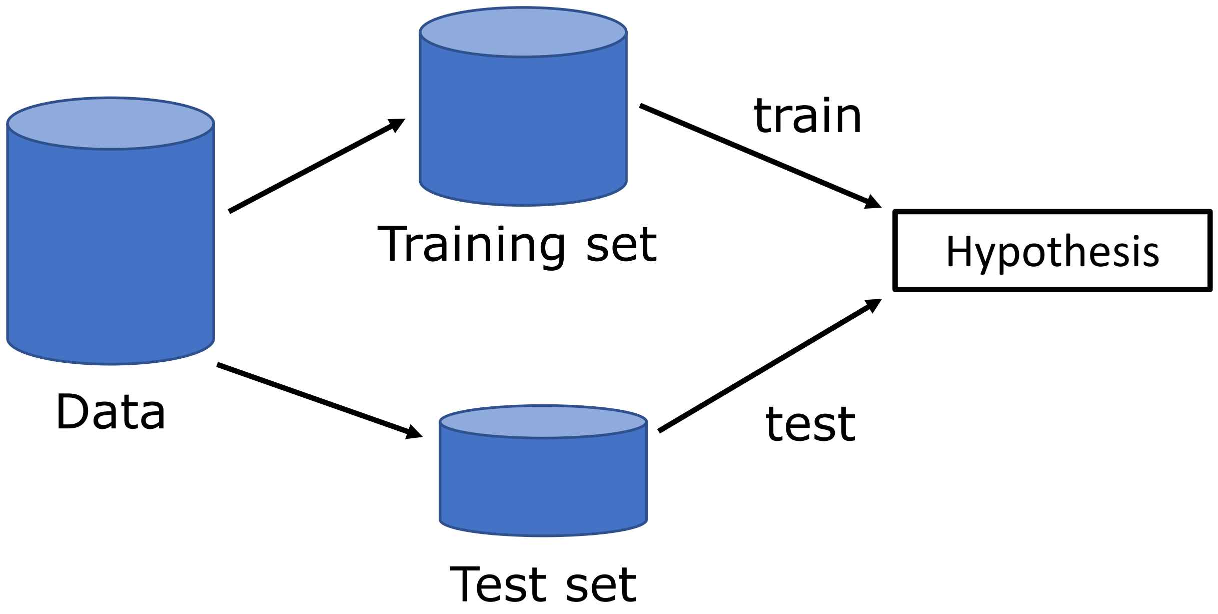
Outline

- Machine Learning
  - What is ML?
  - Types of Feedback
  - Supervised Learning
- **Performance Measure**
  - Regression: mean squared error, mean absolute error
  - Classification: correctness, accuracy, confusion matrix, precision, recall, F1
- Decision Trees
  - Decision Tree Learning (DTL)
  - Entropy and Information Gain
  - Different types of attributes
  - Pruning
  - Ensemble Methods

Performance Measure

How do we know that our hypothesis is good? i.e., $h \approx f$

1. Use **theorems** from statistical learning theory
2. **Try** the hypothesis on a new set of examples (**test set**)



Regression: Error

If the output of the hypothesis is a **continuous** value, then we can measure its **error**. For an input x with a true output y, we can compute:

$$Absolute\ Error = |\hat{y} - y|$$

$$Squared\ Error = (\hat{y} - y)^2$$

Where $\hat{y} = h(x)$.

Regression: Mean Squared Error

For a set of N examples $\{(x_1, y_1), \dots, (x_N, y_N)\}$ we can compute the average (mean) **squared error** as follows.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2$$

Where $\hat{y}_i = h(x_i)$ and $y_i = f(x_i)$.

Regression: Mean Absolute Error

For a set of N examples $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ we can compute the average (mean) **absolute error** as follows.

$$MAE = \frac{1}{N} \sum_{i=1}^{N} |\hat{y}_i - y_i|$$

Where $\hat{y}_i = h(x_i)$ and $y_i = f(x_i)$.

Classification: Correctness & Accuracy

Classification is correct when the prediction $\hat{y} = y$ (true label).

For a set of N examples $\{(x_1, y_1), \dots, (x_N, y_N)\}$ we can compute the average **correctness** (**accuracy**) as follows.

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N}[\hat{y}_i - y_i]$$

Where $\hat{y}_i = h(x_i)$ and $y_i = f(x_i)$.

Classification: Confusion Matrix

Inst.	Actual y	Predicted $\hat{y}$	
1	Cancer	Cancer	TP
2	Cancer	Cancer	TP
3	Cancer	Benign	FN
4	Cancer	Benign	FN
5	Cancer	Benign	FN
6	Benign	Benign	TN
7	Benign	Benign	TN
8	Benign	Benign	TN
9	Benign	Benign	TN
10	Benign	Cancer	FP

$$\text{Accuracy} = \frac{\mathrm{TP+TN}}{\mathrm{TP+FN+FP+TN}}$$

FP: Type I error

FN: Type II error

Can we combine the best of both metrics?

F1 Score

$$F1 = \frac{2}{\frac{1}{P} + \frac{1}{R}}$$

Actual Label

Predicted Label	Cancer	Benign
Cancer	2 True Positive	1 False Positive
Benign	3 False Negative	4 True Negative

Precision

P = TP / (TP+FP)

How many **selected** items are relevant?

How **precise** were the positive predicted instances?

Maximize this if false positive (FP) is very costly.

E.g., email spam, satellite launch date prediction

Recall

R = TP / (TP+FN)

How many relevant items are **selected**?

How many positive instances can be **recalled** (predicted)?

Maximize this if false negative (FN) is very dangerous.

E.g., cancer prediction but not music recommendation

Outline

- Machine Learning
  - What is ML?
  - Types of Feedback
  - Supervised Learning
- Performance Measure
  - Regression: mean squared error, mean absolute error
  - Classification: correctness, accuracy, confusion matrix, precision, recall, F1
- **Decision Trees**
  - Decision Tree Learning (DTL)
  - Entropy and Information Gain
  - Different types of attributes
  - Pruning
  - Ensemble Methods

Life's Difficult Decision: **What to Eat?**

Decide whether to wait for a table at a restaurant based on the following input attributes:

1. Alternate: is there an alternative restaurant nearby?
2. Bar: is there a comfortable bar area to wait in?
3. Fri/Sat: is today Friday or Saturday?
4. Hungry: are we hungry?
5. Patrons: number of people in the restaurant (None, Some, Full)
6. Price: price range ($, $$, $$$)
7. Raining: is it raining outside?
8. Reservation: have we made a reservation?
9. Type: kind of restaurant (French, Italian, Thai, Burger)
10. WaitEstimate: estimated waiting time in minutes (0-10, 10-30, 30-60, >60)

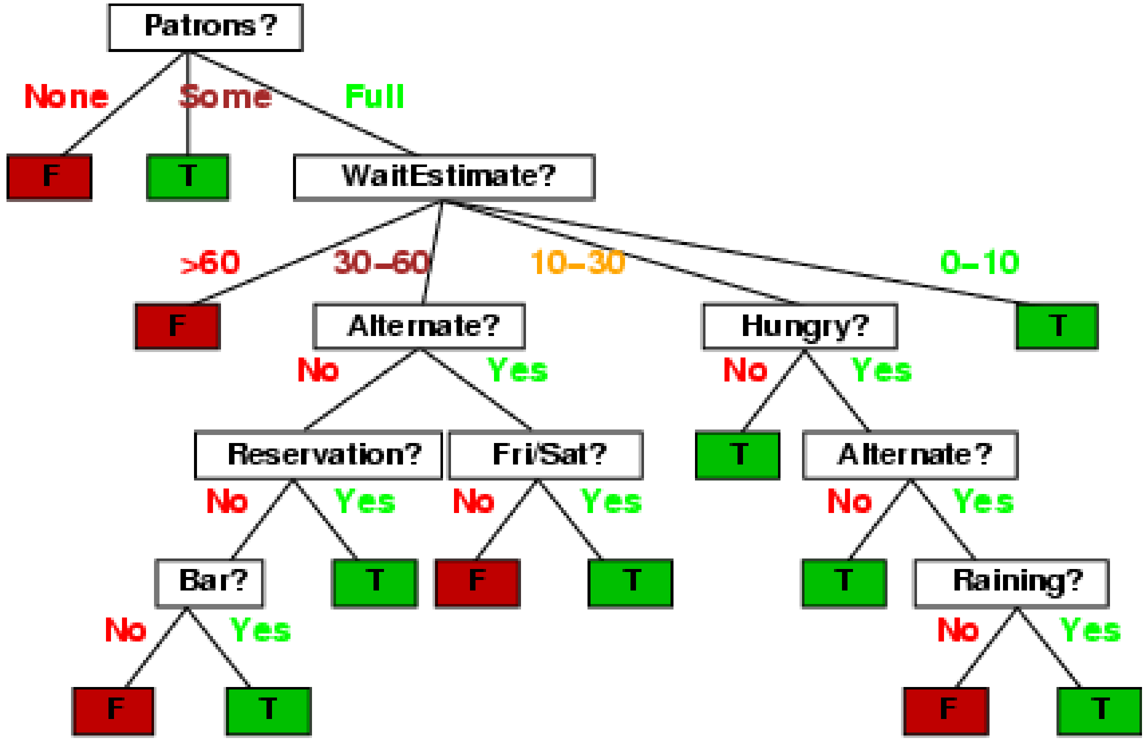
Life's Difficult Decision: **What to Eat?**

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

You are hungry, it's Friday, raining, …; **Should you wait?**

Decision Trees

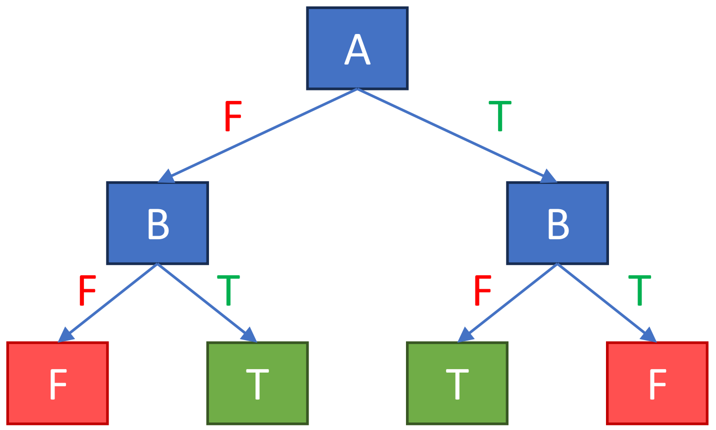
One possible representation for hypotheses



Expressiveness

- Decision trees can express **any function** of the input attributes.
- Example: Boolean functions, each row → path from root to leaf

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



- Trivially, there is a consistent decision tree for <u>any</u> training set, but probably **won't generalize to new examples**.

Prefer a more "compact" tree

The Size of the Hypothesis Class

How many distinct decision trees with n Boolean attributes?

= number of Boolean functions

= number of distinct truth tables with 2^n rows

= 2^{2^n}

A	B		A <op> B
F	F		T / F
F	T		T / F
T	F		T / F
T	T		T / F

With **6 Boolean attributes**, there are 18,446,744,073,709,551,616 trees

Decision Tree Learning

Greedy, top-down, recursive algorithm.

```
def DTL(examples, attributes, default):
   if examples is empty: return default
   if examples have the same classification:
      return classification
   if attributes is empty:
      return mode(examples)
   best = choose_attribute(attributes, examples)
   tree = a new decision tree with root best
   for each value v_i of best:
      examples_i = {rows in examples with best = v_i}
      subtree = DTL(examples_i, attributes – best, mode(examples))
      add a branch to tree with label v_i and subtree subtree
```

Need to define this!

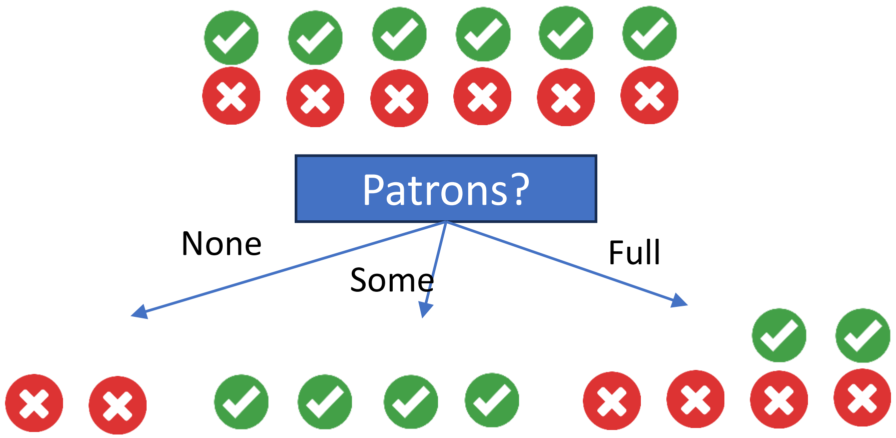
Choosing an Attribute

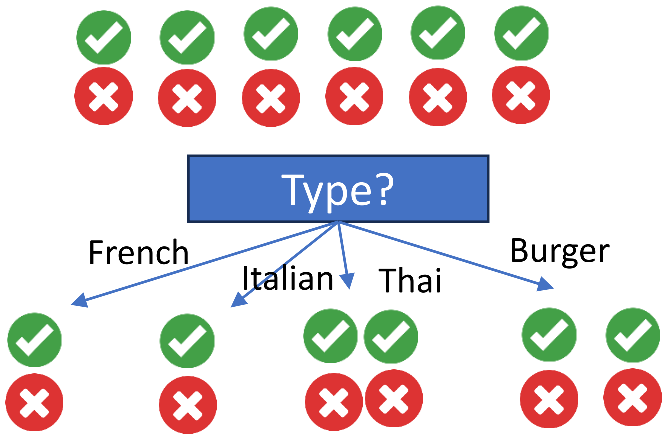
How do we choose an attribute?

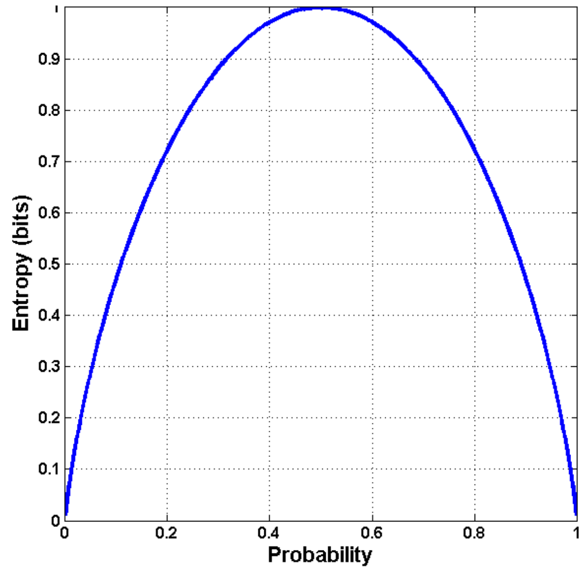
Ideally: we want to select an attribute that split the examples into "all positive" or "all negative"

Entropy

- Entropy is a measure of randomness:
  - $I\big(P(v_1), \ldots, P(v_n)\big) = -\sum_{i=1}^{n} P(v_i) \log_2 P(v_i)$
- For a data set containing p positive and n negative examples:
  - $I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = -\frac{p}{p+n} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n}$







Information Gain

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

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

- Information Gain (IG) or reduction in entropy

$$IG(A) = \boxed{I\left(\frac{p}{p+n}, \frac{n}{p+n}\right)} - \boxed{remainder(A)}$$

Entropy of this node

Entropy of children nodes

Decision Tree Learning

with Information Gain

```
def DTL(examples, attributes, default):
    if examples is empty: return default
    if examples have the same classification:
        return classification
    if attributes is empty:
        return mode(examples)
    best = choose_attribute(attributes, examples)
    tree = a new decision tree with root best
    for each value v_i of best:
        examples_i = {rows in examples with best = v_i}
        subtree = DTL(examples_i, attributes – best, mode(examples))
        add a branch to tree with label v_i and subtree subtree
```

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

(2,4,6) (2,2,4,4)

Patrons: (2,4,6)

2 x F 4 x T 2 x T, 4 x F

$$IG(Patrons) = 1 - [\frac{2}{12}I(0,1) + \frac{4}{12}I(1,0) + \frac{6}{12}I(\frac{2}{6},\frac{4}{6})] = .0541 \text{ bits}$$

Type: (2,2,4,4)

T+F T+F 2T, 2F 2T, 2 F

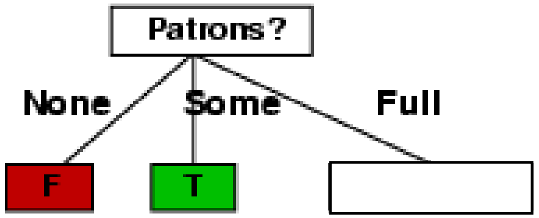
$$IG(Type) = 1 - [\frac{2}{12}I(\frac{1}{2},\frac{1}{2}) + \frac{2}{12}I(\frac{1}{2},\frac{1}{2}) + \frac{4}{12}I(\frac{2}{4},\frac{2}{4}) + \frac{4}{12}I(\frac{2}{4},\frac{2}{4})] = 0 \text{ bits}$$

Decision Tree Learning

with Information Gain

def **DTL**(examples, attributes, default):
    if examples is empty: return default
    if examples have the same classification:
        return classification
    if attributes is empty:
        return **mode**(examples)
    best = **choose_attribute**(attributes, examples)
    tree = a new decision tree with root best
    for each value v_i of best:
        $examples_i$ = {rows in examples with best = v_i}
        subtree = **DTL**($examples_i$, attributes – best, **mode**(examples))
        add a branch to tree with label v_i and subtree subtree

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T



Decision Tree Learning

with Information Gain

def **DTL**(examples, attributes, default):

  if examples is empty: return default

  if examples have the same classification:

    return classification

  if attributes is empty:

    return **mode**(examples)

  best = **choose_attribute**(attributes, examples)

  tree = a new decision tree with root best

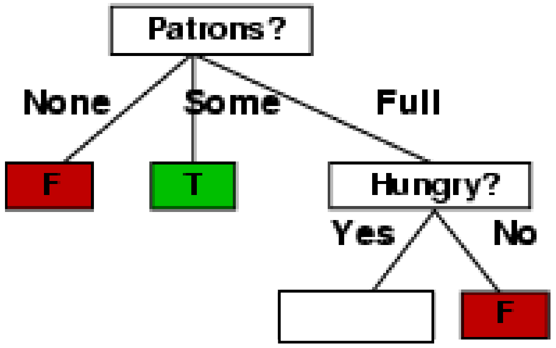
  for each value v_i of best:

    $examples_i$ = {rows in examples with best = v_i}

    subtree = **DTL**($examples_i$, attributes – best, **mode**(examples))

    add a branch to tree with label v_i and subtree subtree

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	[redacted]										
X_2	T	F	F	T	[redacted]	$	F	F	Thai	30–60	F
X_3	[redacted]										
X_4	T	F	T	T	[redacted]	$	F	F	Thai	10–30	T
X_5	T	F	T	F	[redacted]	$$$	F	T	French	>60	F
X_6	[redacted]										
X_7	[redacted]										
X_8	[redacted]										
X_9	F	T	T	F	[redacted]	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	[redacted]	$$$	F	T	Italian	10–30	F
X_{11}	[redacted]										
X_{12}	T	T	T	T	[redacted]	$	F	F	Burger	30–60	T

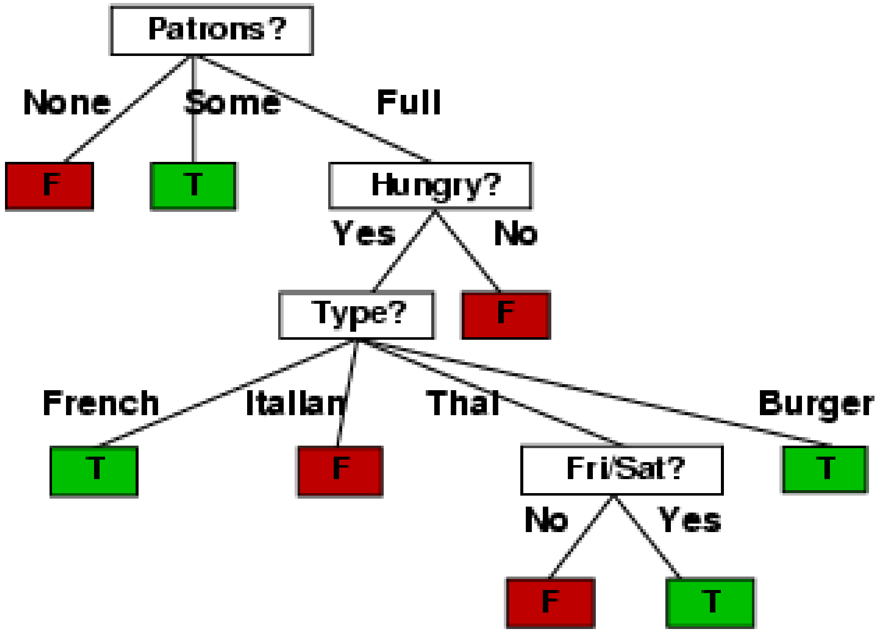


Decision Tree Learning

with Information Gain

def **DTL**(examples, attributes, default):
    if examples is empty: return default
    if examples have the same classification:
        return classification
    if attributes is empty:
        return **mode**(examples)
    best = **choose_attribute**(attributes, examples)
    tree = a new decision tree with root best
    for each value v_i of best:
        $examples_i$ = {rows in examples with best = v_i}
        subtree = **DTL**($examples_i$, attributes – best, **mode**(examples))
        add a branch to tree with label v_i and subtree subtree

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	[redacted]										
X_2											
X_3											
X_4											
X_5											
X_6											
X_7											
X_8											
X_9											
X_{10}											
X_{11}											
X_{12}											



Dealing with Attributes with Many Values

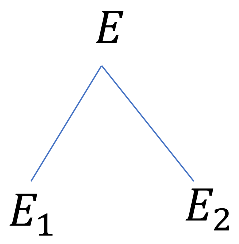
Information gain will select attribute with many values (e.g., dates, phone numbers) because it splits the data “well”. In the **extreme case**, **each branch** will have a **single example**, so “all positive” or “all negative”.

Balance the information gain with the **number of branches**.

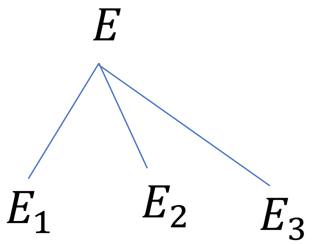
$$GainRatio(A) = \frac{IG(A)}{SplitInformation(A)}$$

$$SplitInformation(A) = -\sum_{i=1}^{d} \frac{|E_i|}{|E|} \log_2 \frac{|E_i|}{|E|}$$

Examples



SplitInformation = 1



SplitInformation = 1.6

Dealing with Attributes with Differing Costs

Certain attributes may have costs (e.g., in medical diagnosis). They may vary significantly in their costs, both in terms of monetary cost or patient comfort. Example: biopsy, blood test, etc.

Make decision trees to **use low-cost attributes** where possible using **Cost-Normalized-Gain**:

$$\frac{IG^2(A)}{Cost(A)}$$

$$\frac{2^{IG(A)} - 1}{(Cost(A) + 1)^w}, w \in [0,1]$$

Determine the importance of cost

Dealing with Continues-valued Attributes

Define a discrete-valued input attribute to **partition the values into a discrete set of intervals**.

Examples:

- Estimated waiting time (minutes): 0-10, 10-30, 30-60, >60
- Age (year): 0-12, 12-25, 25-40, 40-60, 60-80, >80
- …

Dealing with Missing Values

What if some values are missing?

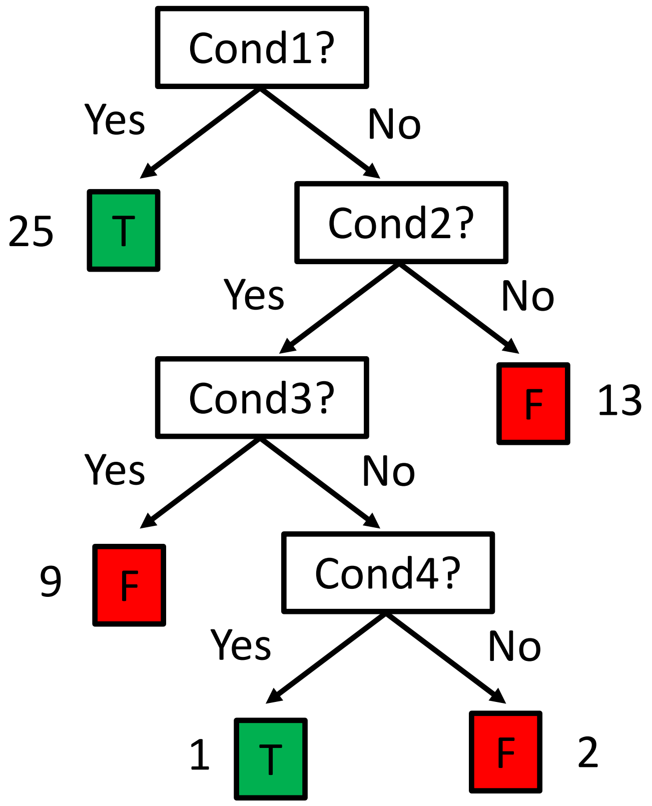
- Assign the most common value of the attribute
- Assign the most common value of the attribute with the same output
- Assign probability to each possible value and sample
- Drop the attribute
- Drop the rows
- …

Overfitting

- Decision Trees performance is <u>perfect on training data</u>, but worse on test data
  - DT captures the data perfectly, including the noise
- Occam's Razor
  - Prefer short/simple hypotheses
  - In favor:
    - **Short/simple** hypothesis that fits the data is **unlikely** to be **coincidence**
    - **Long/complex** hypothesis that fits the data **may be coincidence**
  - Against:
    - Many ways to define small sets of hypotheses (e.g., trees with prime number of nodes that uses attribute beginning with "Z")
    - Different hypotheses representations may be used instead

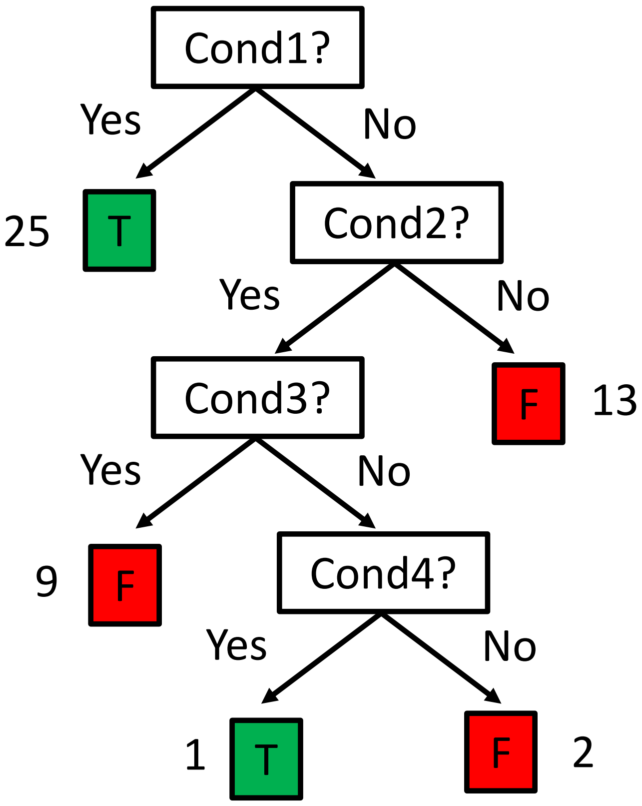
Pruning

Prevent nodes from being split even when if fails to cleanly separate examples.

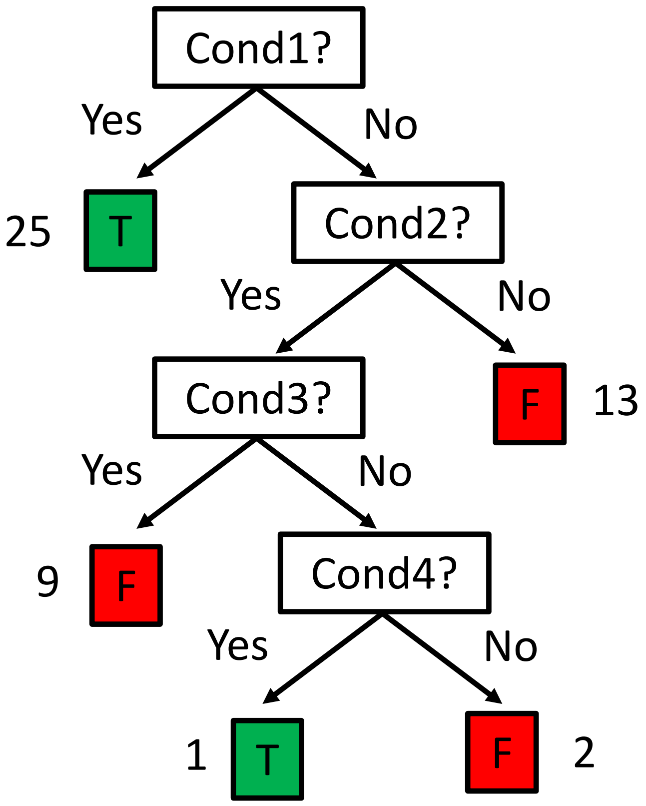


Pruning

Prevent nodes from being split even when if fails to cleanly separate examples.

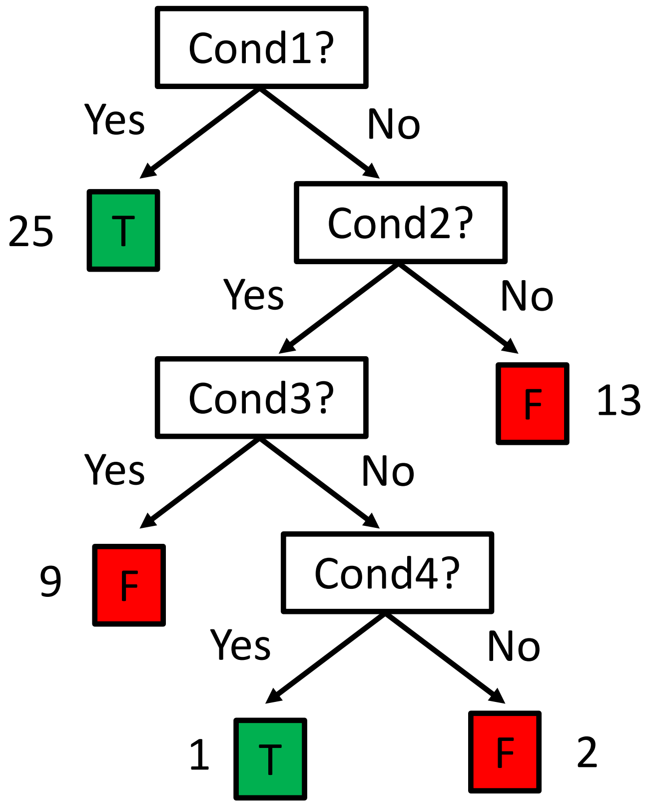


Min-sample = 5

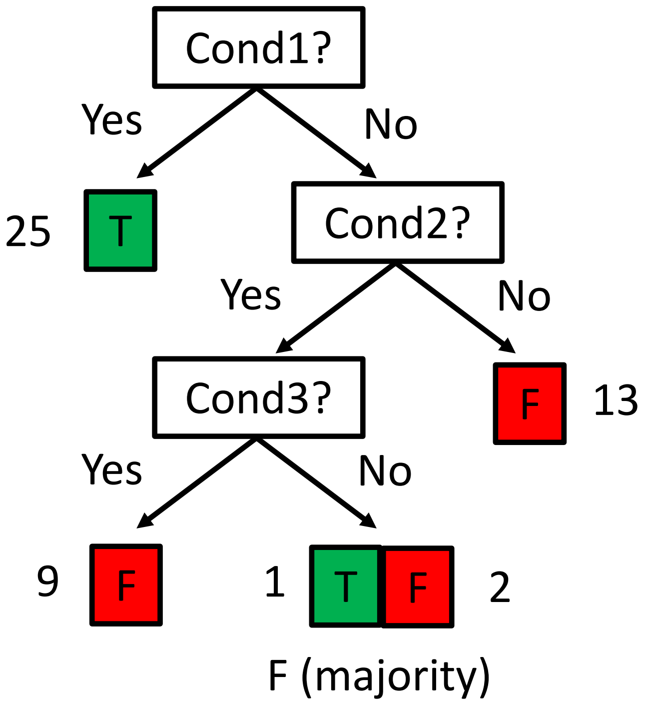


Pruning

Prevent nodes from being split even when if fails to cleanly separate examples.

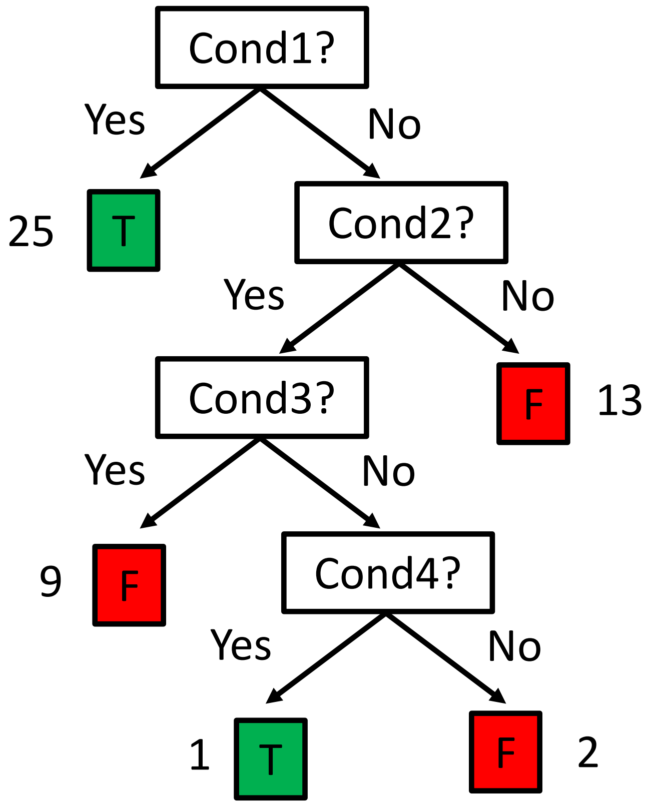


Min-sample = 5

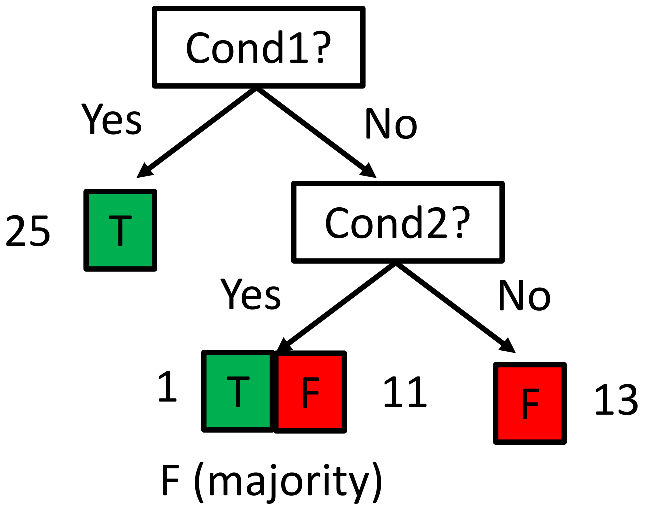


Pruning

Prevent nodes from being split even when if fails to cleanly separate examples.



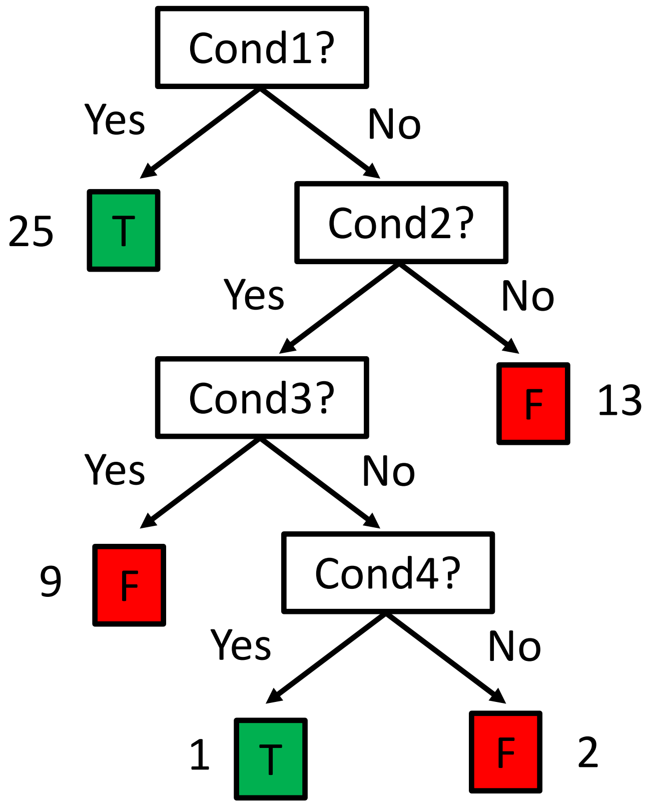
Min-sample = 5



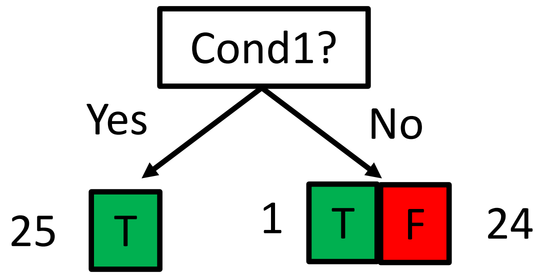
Done!
But can be simplified…

Pruning

Prevent nodes from being split even when if fails to cleanly separate examples.

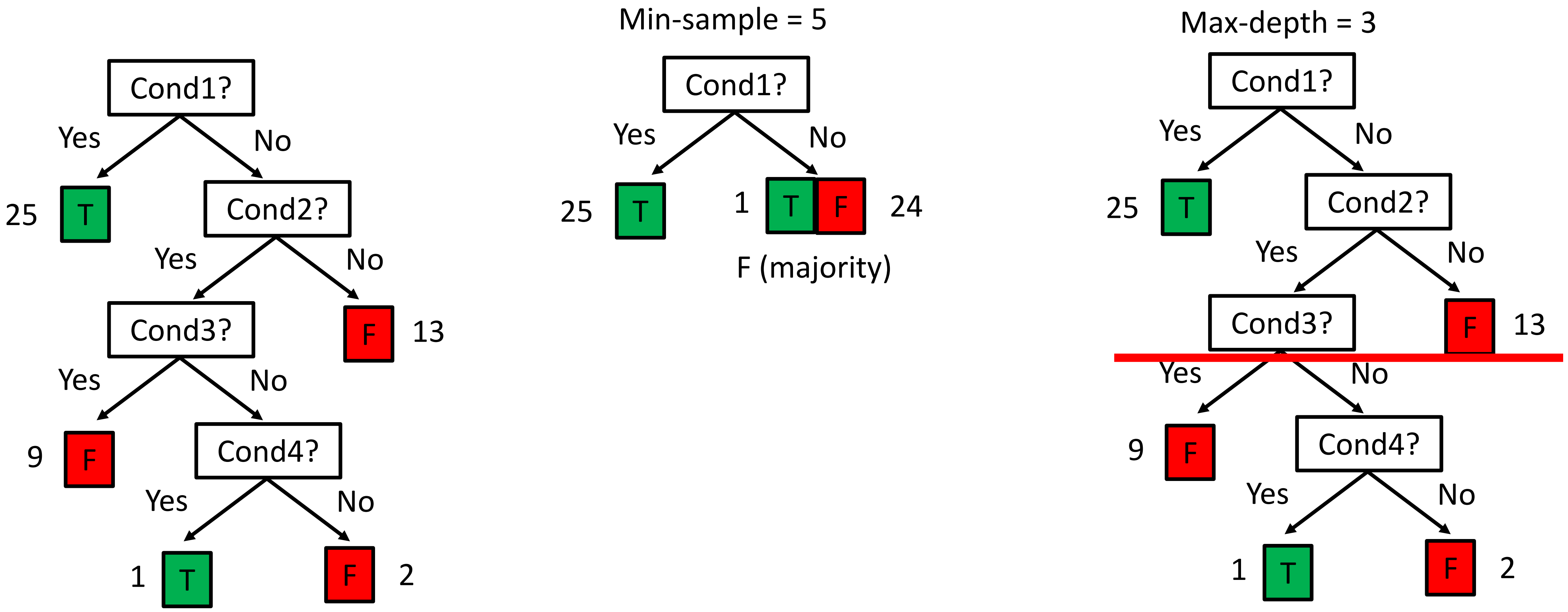


Min-sample = 5



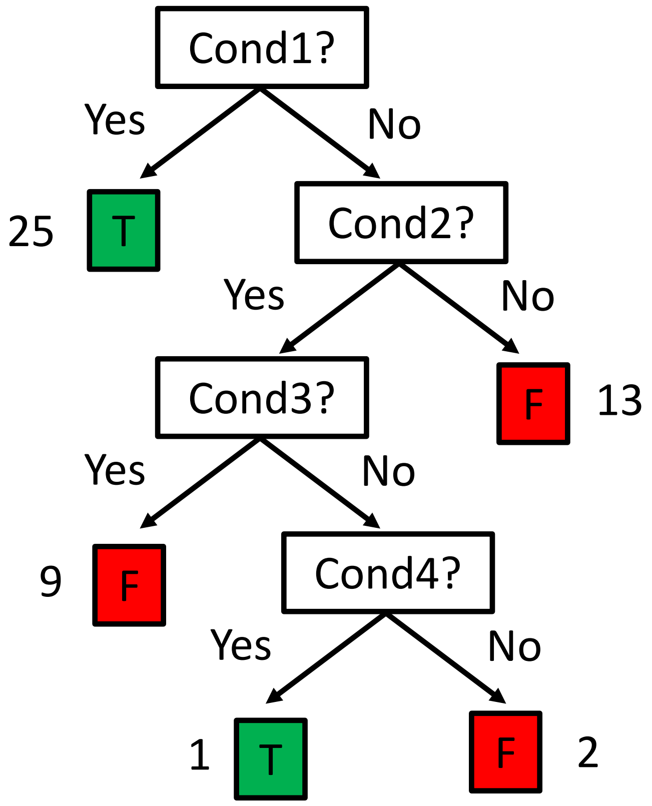
Pruning

Prevent nodes from being split even when if fails to cleanly separate examples.

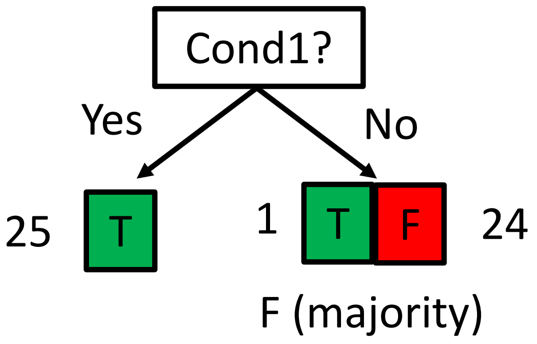


Pruning

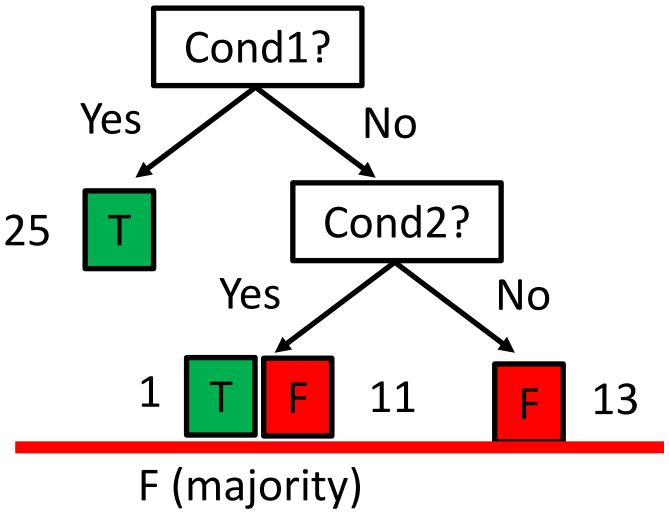
Prevent nodes from being split even when if fails to cleanly separate examples.



Min-sample = 5



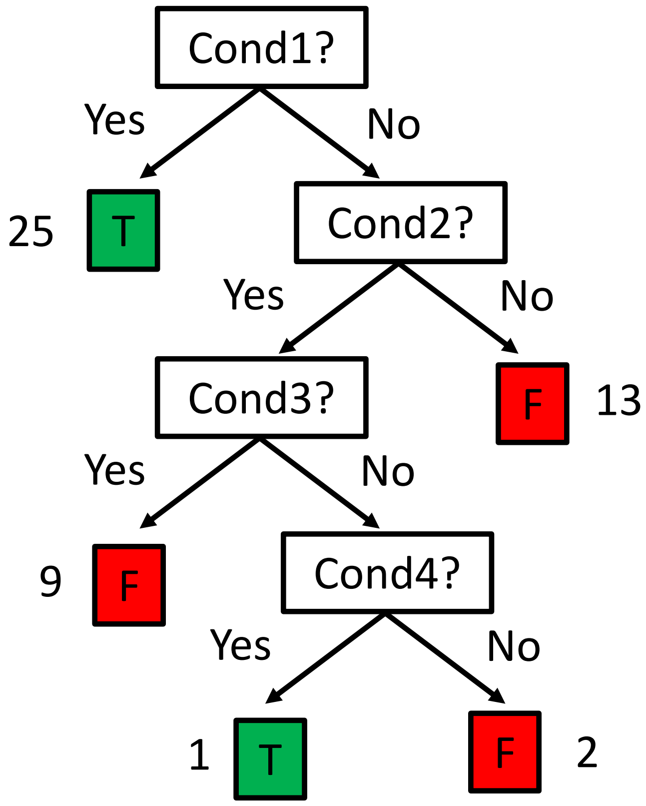
Max-depth = 3



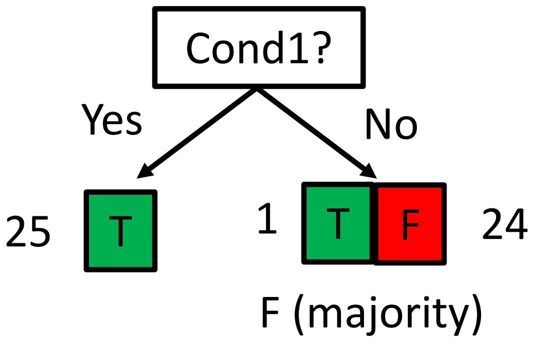
Done!
But can be simplified…

Pruning

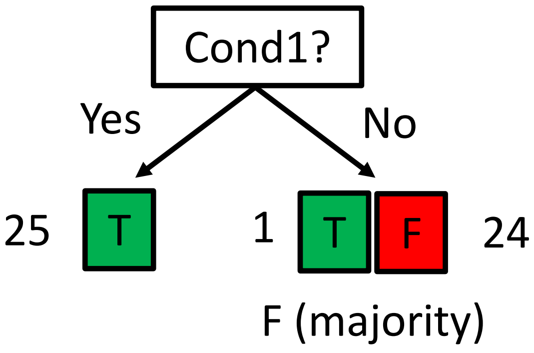
Prevent nodes from being split even when if fails to cleanly separate examples.



Min-sample = 5



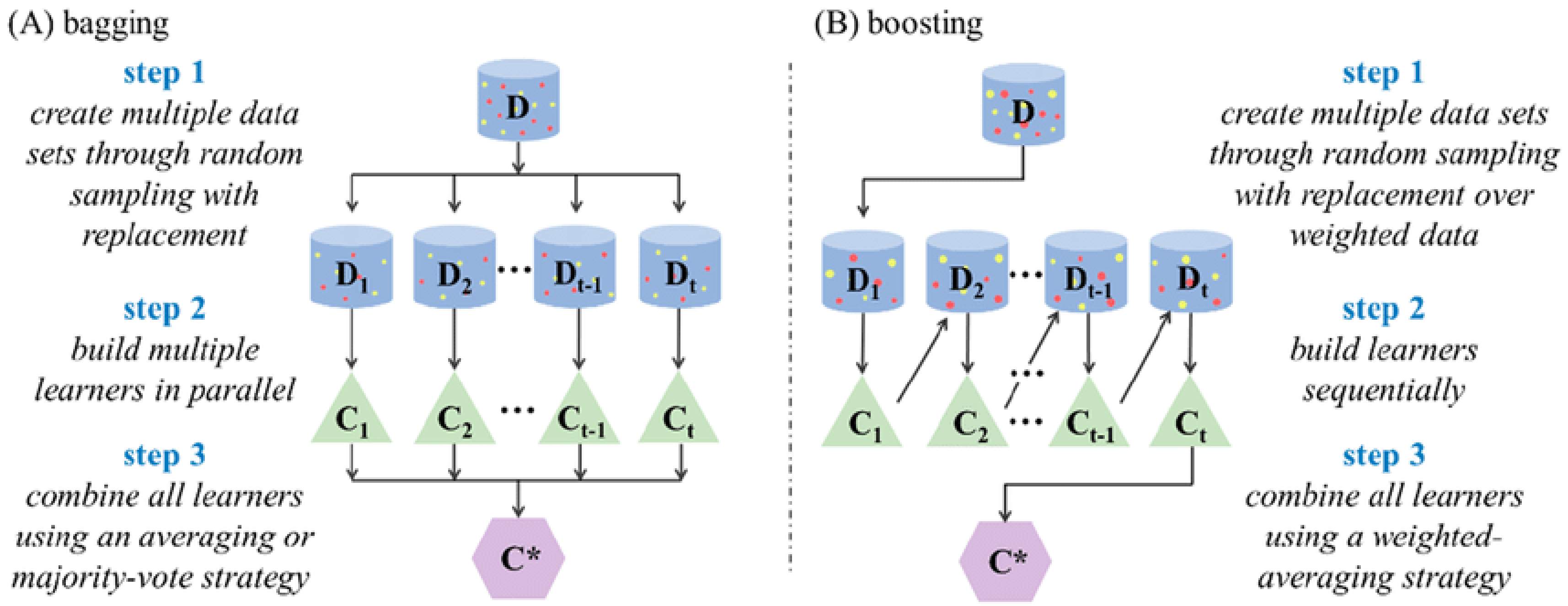
Max-depth = 3



The sample that is likely a **noise** (1T) is ignored!

Results in a **smaller tree** which may have **higher accuracy**

Ensemble Methods



- **Bootstrap Aggregation (Bagging)**
  - Random Forests
- **Boosting**
  - Adaboost, XGBoost

Image Credit: https://www.researchgate.net/figure/Illustrations-of-A-bagging-and-B-boosting-ensemble-algorithms_fig4_334404567

Summary

- Machine Learning
  - What is ML? – machine that learns through data
  - Types of Feedback: supervised, unsupervised, semi-supervised, reinforcement
  - Supervised Learning:
- Performance Measure
  - Regression: mean squared error, mean absolute error
  - Classification: correctness, accuracy, confusion matrix, precision, recall, F1
- Decision Trees
  - Decision Tree Learning (DTL): greedy, top-down, recursive algorithm
  - Entropy and Information Gain
  - Different types of attributes: many values, differing costs, missing values
  - Pruning: min-sample, max-depth
  - Ensemble Methods: bagging, boosting

Coming Up Next Week

- Linear Regression
  - Multiple Features
  - Polynomial Regression
- Optimization Algorithms
  - Gradient Descent
    - Variants of Gradient Descent
  - Normal Equation

To Do

- **Lecture Training 4**
  - +100 Free EXP
  - +50 Early bird bonus
- **Problem Set 2**
  - Due Tuesday, 12^{th} September (Extended!)
- **Problem Set 3**
  - Out today!
  - Contest +500 EXP