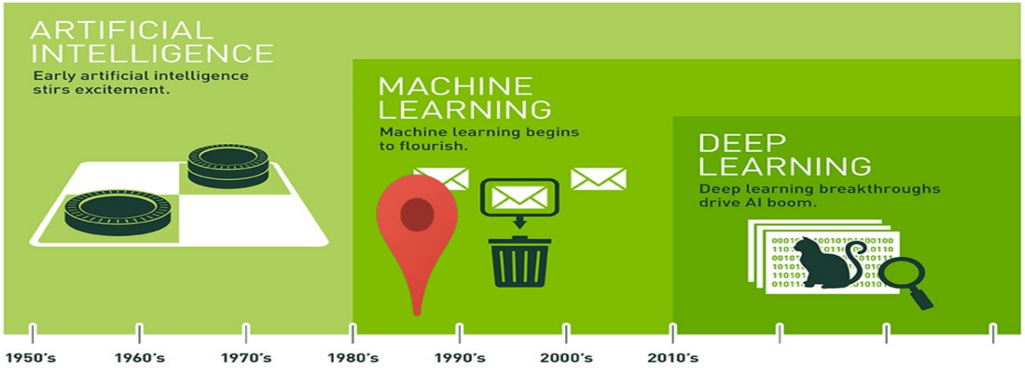
Artificial Intelligence: ML: Decision Trees & k-means Clustering

- Russell & Norvig: Sections 18.3, 18.4

Today

1. Introduction to ML (contd.)
2. Decision Trees
3. Evaluation (contd.)
4. Unsupervised Learning: k-means Clustering



https://medium.com/machine-learning-for-humans/neural-networks-deep-learning-cdad8aeae49b

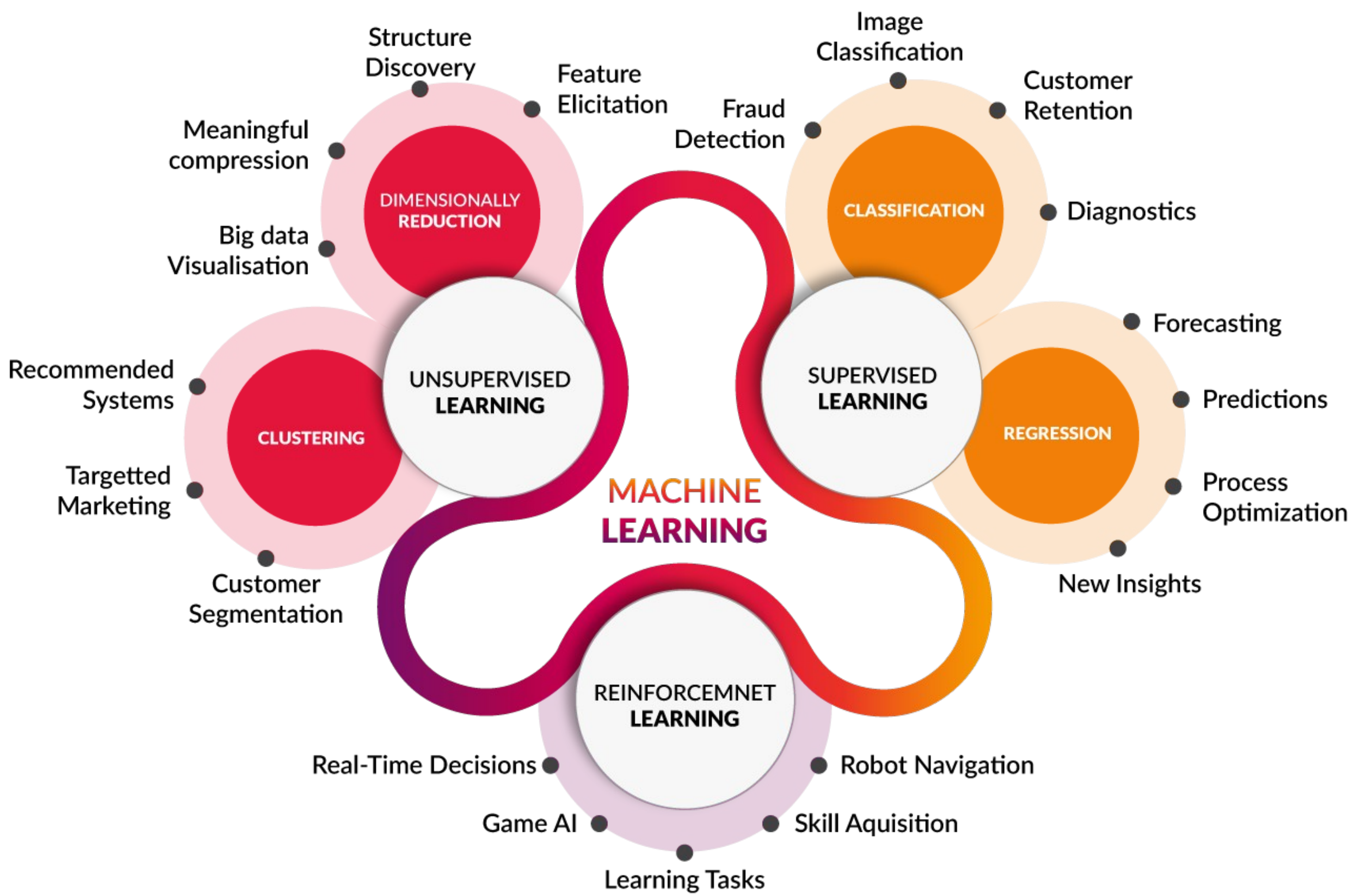
Applications

- Too many to list here!
  - Recommender systems (eg. Netflix)
  - Pattern Recognition (eg. Handwriting recognition)
  - Detecting credit card fraud
  - Computer vision (eg. Object recognition)
  - Discovering Genetic Causes of Diseases
  - Natural Language Processing (eg. Spam filtering)
  - Speech Recognition / Synthesis
  - Medical Diagnostics
  - Information Retrieval (eg. Image search)
  - Learning heuristics for game playing
  - …
  - *Oh… I'm out of space*

What is Machine Learning?

- Learning = crucial characteristic of an intelligent agent
- ML
  - Constructs algorithms that learn from data
  - i.e., perform tasks that were not explicitly programmed and improve their performance the more tasks they accomplish
    - generalize from given experiences and are able to make judgments in new situations

Types of Machine Learning

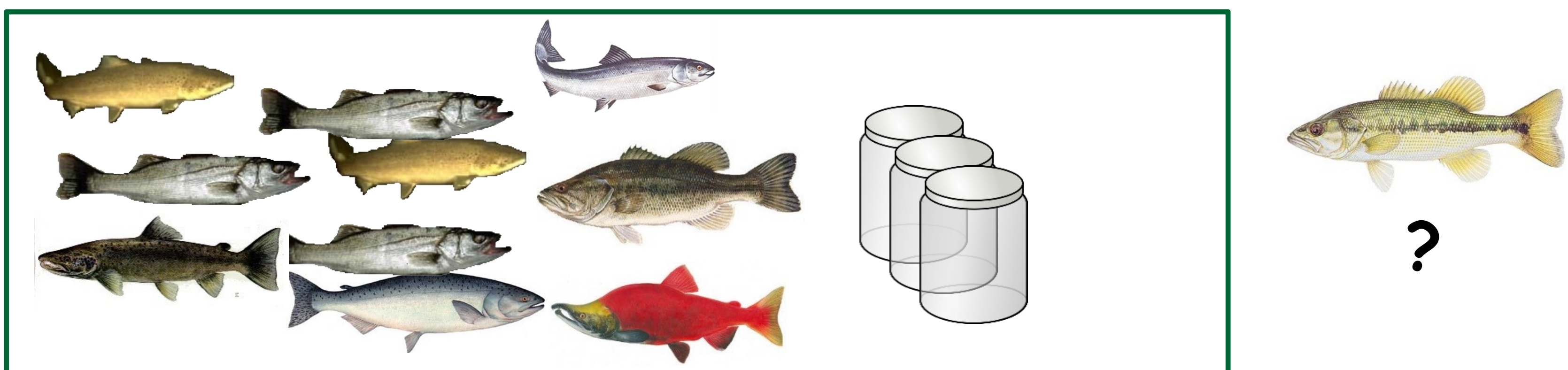


http://www.cognub.com/index.php/cognitive-platform/

Types of Machine Learning

- **Supervised learning**
  - We are given a training set of (X, f(X)) pairs
  - X = <color, length>

- **Unsupervised learning**
  - We are only given the Xs - not the corresponding f(X)



Types of Learning

- In **Supervised** learning
  - We are given a training set of (X, f(X)) pairs

big nose	big teeth	big eyes	no moustache	f(X) = not person
small nose	small teeth	small eyes	no moustache	f(X) = person
small nose	big teeth	small eyes	moustache	f(X) = ?

- In **Reinforcement** learning
  - We are not given the (X, f(X)) pairs

small nose	big teeth	small eyes	moustache	f(X) = ?

  - But we get a reward when our learned f(X) is right, and we try to maximize the reward
  - Goal: maximize the nb of right answers
- In **Unsupervised** learning
  - We are only given the Xs - not the corresponding f(X)

big nose	big teeth	big eyes	no moustache	*not given*
small nose	small teeth	small eyes	no moustache	*not given*
small nose	big teeth	small eyes	moustache	f(X) = ?

  - No *teacher* involved / Goal: find regularities among the Xs (clustering)
  - Data mining

Logical Inference

- Inference: process of deriving new facts from a set of premises

- Types of logical inference:
  1. Deduction
  2. Abduction
  3. Induction

Deduction

- aka Natural Deduction
- Conclusion follows necessary from the premises.
- From *A* B and *A*, we conclude that B
- We conclude from the general case to a specific example of the general case
- Ex:

  All men are mortal.

  Socrates is a man.

  Socrates is mortal.

Abduction

- Conclusion is one hypothetical (most probable) explanation for the premises
- From $A \Rightarrow B$ and B, we conclude A
- Ex:

  Drunk people do not walk straight.
  John does not walk straight.
  ____
  John is drunk.

- Not sound… but may be most likely explanation for B

- Used in medicine…
  - in reality… disease $\Rightarrow$ symptoms
  - patient complains about some symptoms… doctor concludes a disease

Induction

- Conclusion about all members of a class from the examination of only a few member of the class.

From $A \wedge C \Rightarrow B$ and $A \wedge D \Rightarrow B$, we conclude $A \Rightarrow B$

- We construct a general explanation based on a specific case.
- Ex:

All CS students in COMP 6721 are smart.

All CS students on vacation are smart.

All CS students are smart.

- Not sound
- But, can be seen as hypothesis construction or generalisation

Inductive Learning

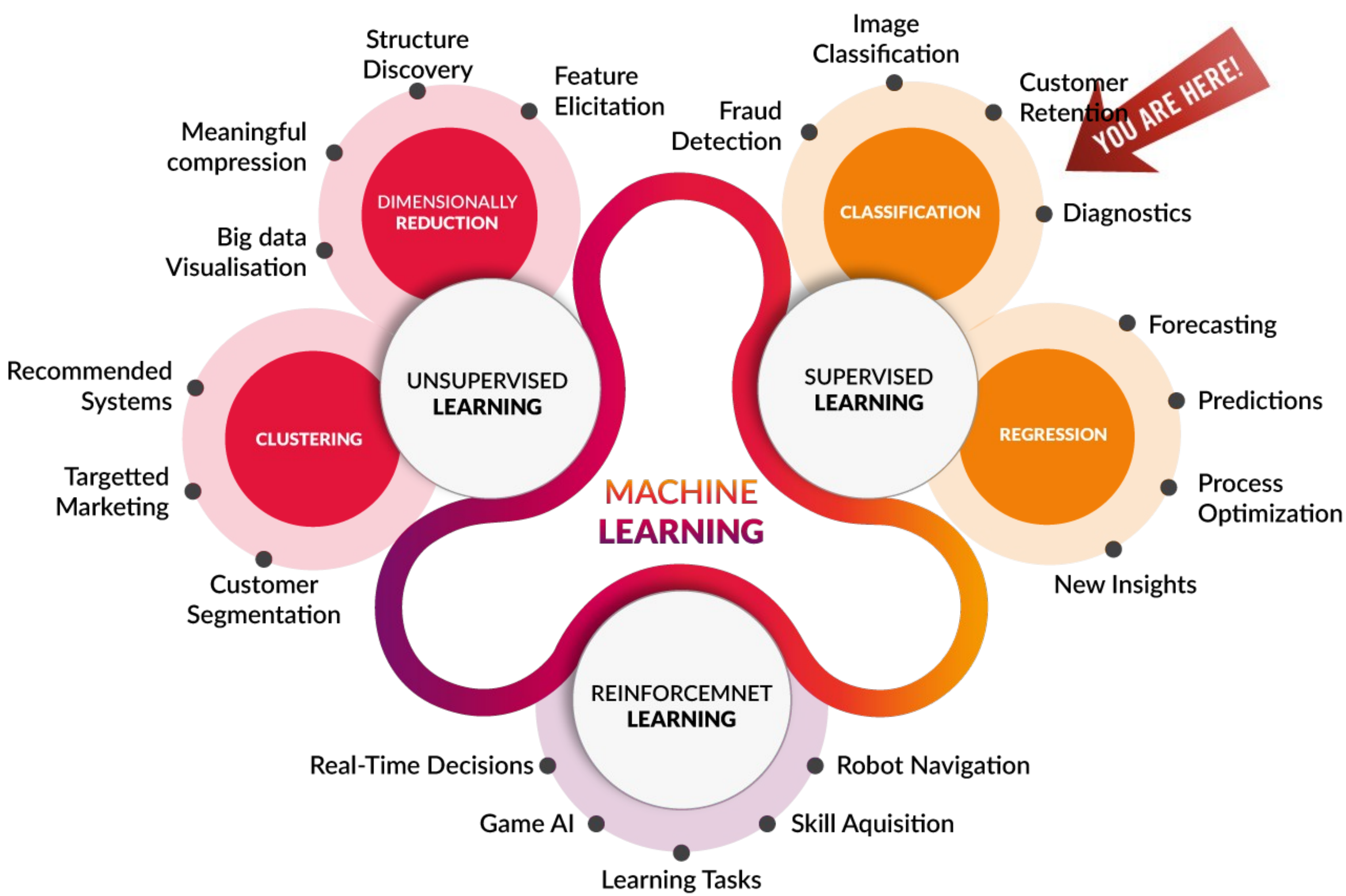
- = learning from examples
- Most work in ML
- Examples are given (positive and/or negative) to train a system in a classification (or regression) task
- Extrapolate from the training set to make accurate predictions about future examples
- Can be seen as learning a function
- Given a new instance X you have never seen
- You must find an estimate of the function f(X) where f(X) is the desired output
- Ex:

small nose	big teeth	small eyes	moustache	f(X) = ?

X

  - X = features of a face (ex. small nose, big teeth, …)
  - f(X) = function to tell if X represents a human face or not

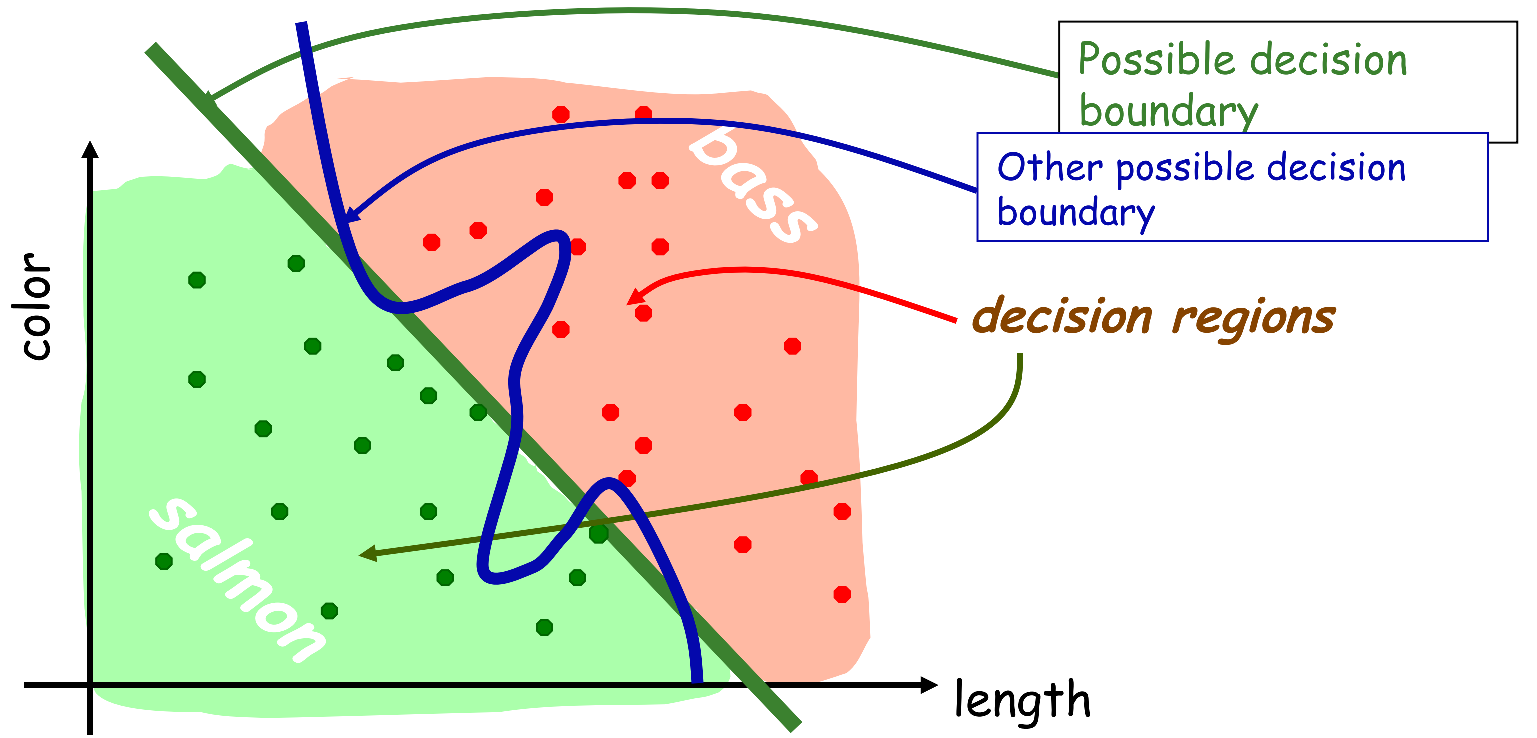
Types of Machine Learning



http://www.cognub.com/index.php/cognitive-platform/

Example

- Given pairs (X,f(X)) (the training set – the data points)
- Find a function that fits the training set well
- So that given a new X, you can predict its f(X) value



- Note: choosing one function over another beyond just looking at the training set is called **inductive bias** (eg. prefer "smoother" functions)

Inductive Learning Framework

- Input data are represented by a vector of features, X
- Each vector X is a list of (attribute, value) pairs.
  - Ex: `x = [nose:big, teeth:big, eyes:big, moustache:no]`
- The number of attributes is fixed (positive, finite)
- Each attribute has a fixed, finite number of possible values
- Each example can be interpreted as a point in a n-dimensional feature space
  - where n is the number of attributes

Note: *attribute == feature*

Example

has-hair?	has-scales?	has-feathers?	flies?	lives in water?	lays eggs?	
1	0	0	0	0	0	Dog
1	0	0	0	0	0	Cat
1	0	0	1	0	0	Bat
1	0	0	0	1	0	Whale
0	0	1	1	0	1	Canary
0	0	1	1	0	1	Robin
0	0	1	1	0	1	Ostrich
0	1	0	0	0	1	Snake
0	1	0	0	0	1	Lizard
0	1	0	0	1	1	Alligator

Real ML applications typically require hundreds, thousands or millions of examples

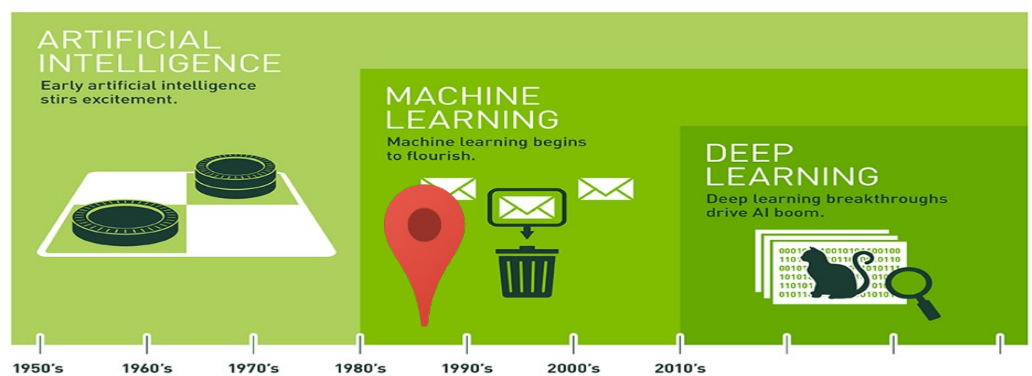
source: Alison Cawsey: The Essence of AI (1997).

Techniques in ML

- Probabilistic Methods
  - ex: Naïve Bayes Classifier
- Decision Trees
  - Use only discriminating features as questions in a big if-then-else tree
- Neural networks
  - Also called parallel distributed processing or connectionist systems
  - Intelligence arise from having a large number of simple computational units
- …

Today

1. Introduction tc  ıtd.)
2. Decision Trees
3. Evaluation (contd.)
4. Unsupervised Learning: k-means Clustering



https://medium.com/machine-learning-for-humans/neural-networks-deep-learning-cdad8aeae49b

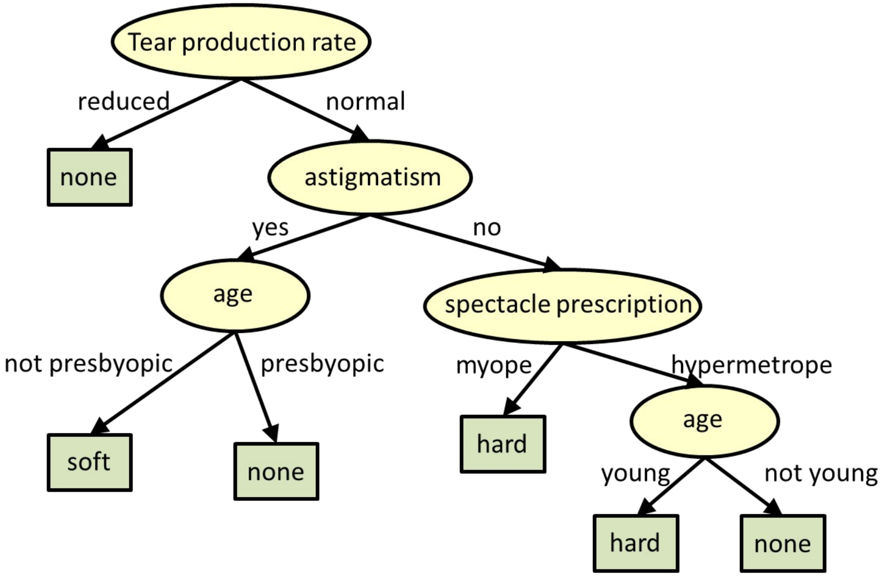
Guess Who?

Decision Trees

- Simplest, but most successful form of learning algorithm
- Very well-know algorithm is ID3 (Quinlan, 1987) and its successor C4.5
- Look for features that are very good indicators of the result, place these features (as questions) in nodes of the tree
- Split the examples so that those with different values for the chosen feature are in a different set
- Repeat the same process with another feature

ID3 / C4.5 Algorithm

- Top-down construction of the decision tree
- Recursive selection of the "best feature" to use at the current node in the tree
  - Once the feature is selected for the current node, generate children nodes, one for each possible value of the selected attribute
  - Partition the examples using the possible values of this attribute, and assign these subsets of the examples to the appropriate child node
  - Repeat for each child node until all examples associated with a node are classified



http://www.rulequest.com/Personal/

Example

Info on last year's students to determine if a student will get an 'A' this year

	Features (X)				Output f(X)
Student	**'A' last year?**	**Black hair?**	**Works hard?**	**Drinks?**	**'A' this year?**
X1: Richard	Yes	Yes	No	Yes	**No**
X2: Alan	Yes	Yes	Yes	No	**Yes**
X3: Alison	No	No	Yes	No	**No**
X4: Jeff	No	Yes	No	Yes	**No**
X5: Gail	Yes	No	Yes	Yes	**Yes**
X6: Simon	No	Yes	Yes	Yes	**No**

→ Worksheet #4 (Decision Tree)

Example 2: The Restaurant

- Goal: learn whether one should wait for a table
- Attributes
  - Alternate: another suitable restaurant nearby
  - Bar: comfortable bar for waiting
  - Fri/Sat: true on Fridays and Saturdays
  - Hungry: whether one is hungry
  - Patrons: how many people are present (none, some, full)
  - Price: price range ($, $$, $$$)
  - Raining: raining outside
  - Reservation: reservation made
  - Type: kind of restaurant (French, Italian, Thai, Burger)
  - WaitEstimate: estimated wait by host (0-10 mins, 10-30, 30-60, >60)

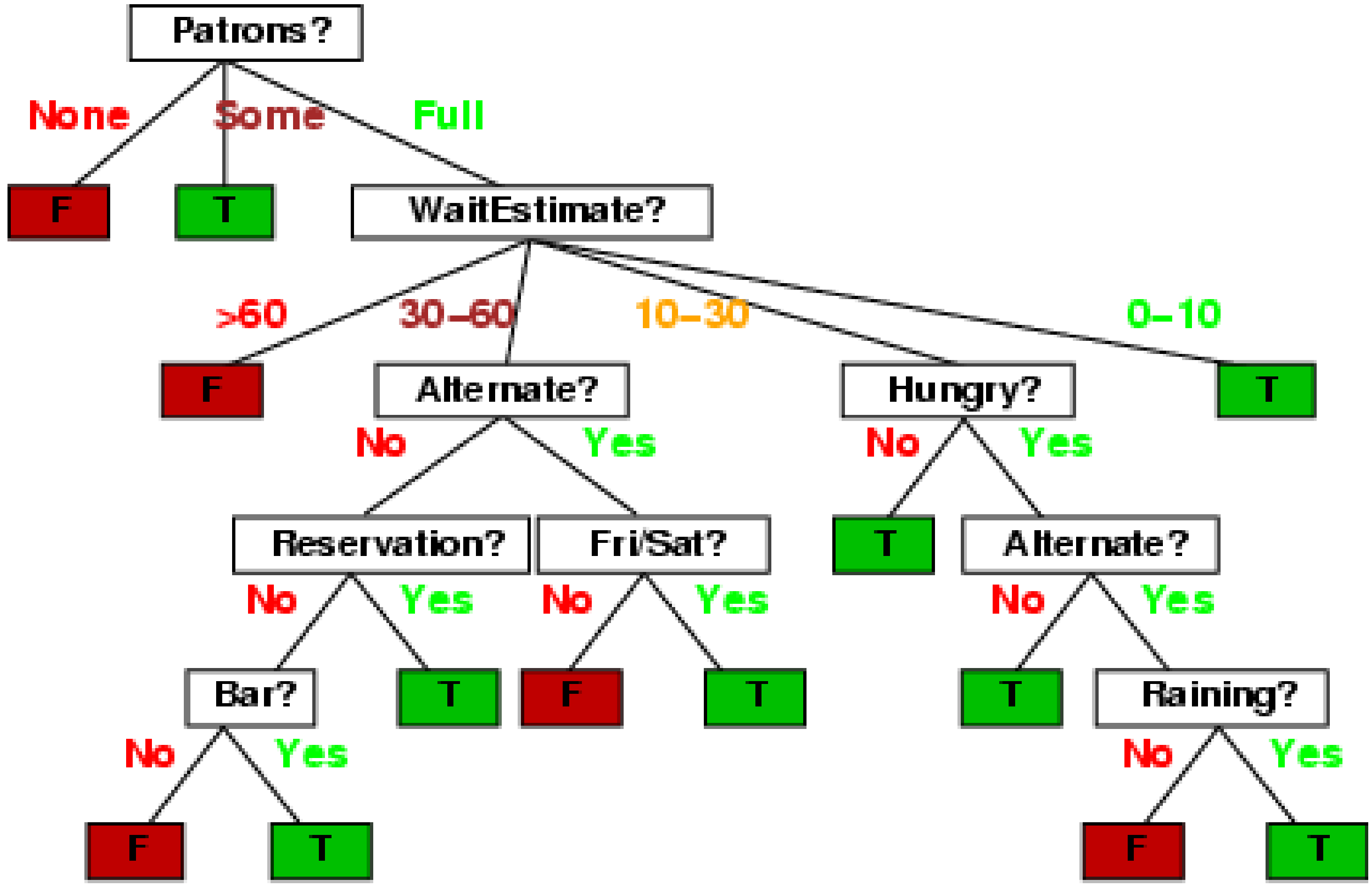
Example 2: The Restaurant

- Training data:

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

source: Norvig (2003)

A First Decision Tree



- But is it the best decision tree we can build?

source: Norvig (2003)

Ockham's Razor Principle

"It is vain to do more than can be done with less… Entities should not be multiplied beyond necessity."
[Ockham, 1324]

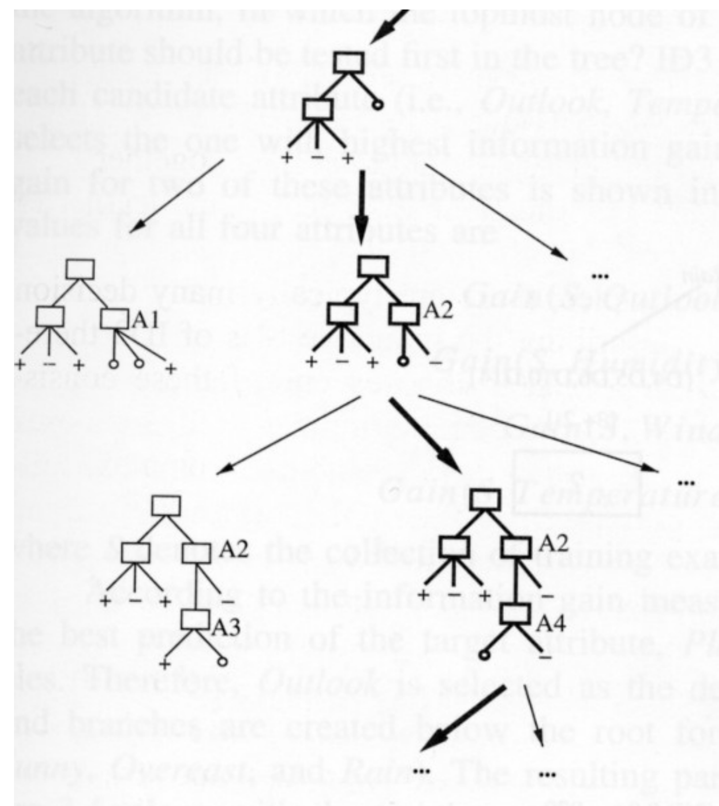
- In other words… always favor the simplest answer that correctly fits the training data
- i.e. the smallest tree on average

- This type of assumption is called inductive bias
  - inductive bias = making a choice beyond what the training instances contain

Finding the Best Tree

- can be seen as searching the space of all possible decision trees
- Inductive bias: prefer shorter trees on average
- how?
- search the space of all decision trees
  - always pick the next attribute to split the data based on its "discriminating power" (information gain)
  - in effect, steepest ascent hill-climbing search where heuristic is information gain

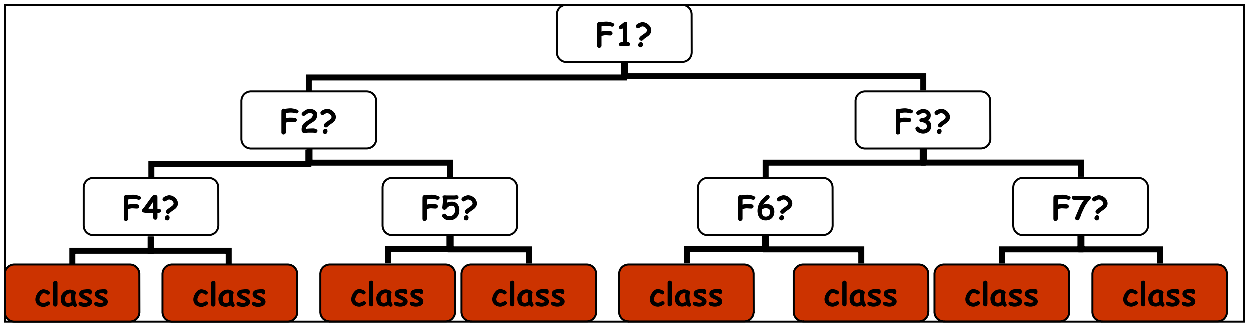
empty tree

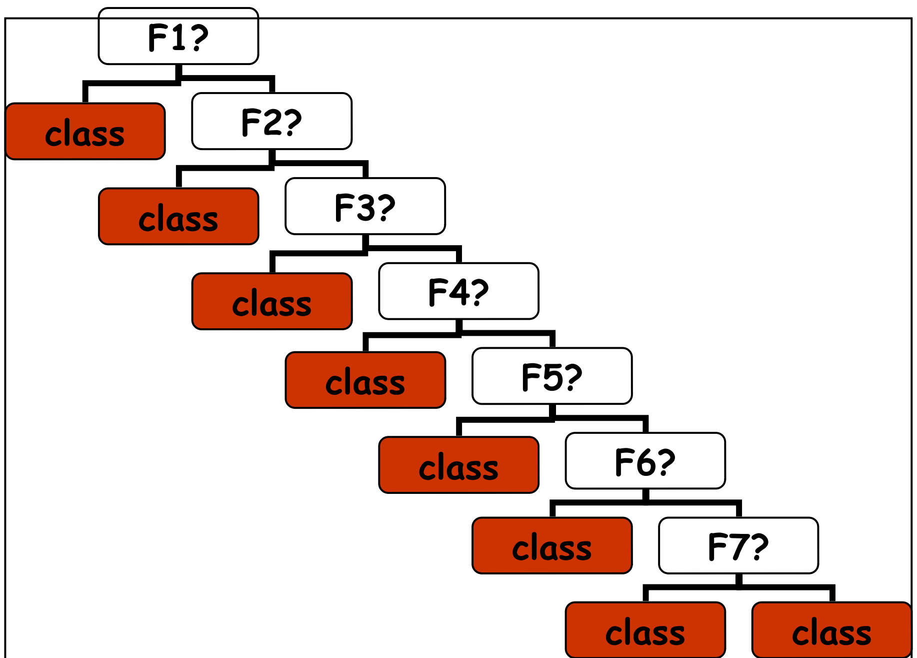


complete tree

source: Tom Mitchell, Machine Learning (1997)

Which Tree is Best?





Smaller trees are better

What's the size of a tree?

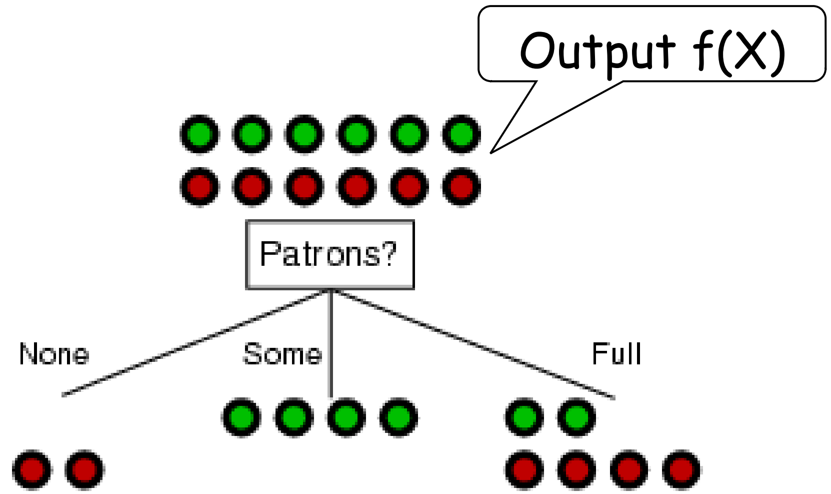
- Number of leaves
- Height of the tree
  - Longest path in the tree from the root to a leaf
- External Path Length
  - Start at leaf, go up to the root and count the number of edges
  - Do this for every leaf and add up the numbers
- Weighted External Path Length
  - Idea: not all paths are equally important/likely
  - Use the training data to computed a weighted sum

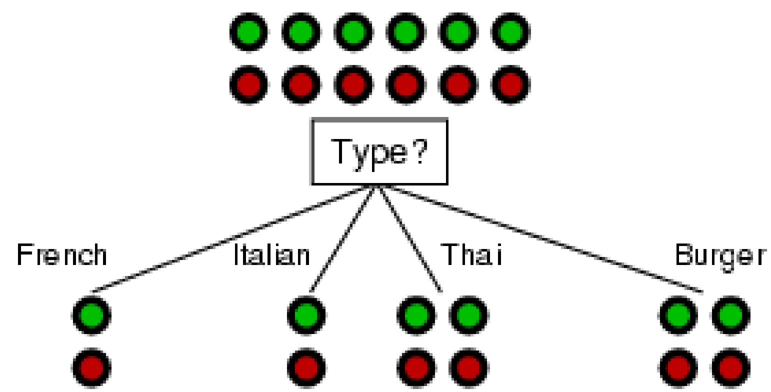
Choosing the Next Attribute

- The key problem is choosing which feature to split a given set of examples
- ID3 uses Maximum Information-Gain:
  - Choose the attribute that has the largest information gain
    - i.e., the attribute that will result in the smallest expected size of the subtrees rooted at its children
    - information theory

Intuitively…

- *Patron*:
  - If value is *Some*… all outputs=*Yes*
  - If value is *None*… all outputs=*No*
  - If value is *Full*… we need more tests
- *Type*:
  - If value is *French*… we need more tests
  - If value is *Italian*… we need more tests
  - If value is *Thai*… we need more tests
  - If value is *Burger*… we need more tests
- …
- So *patron* may lead to shorter tree…





source: Norvig (2003)

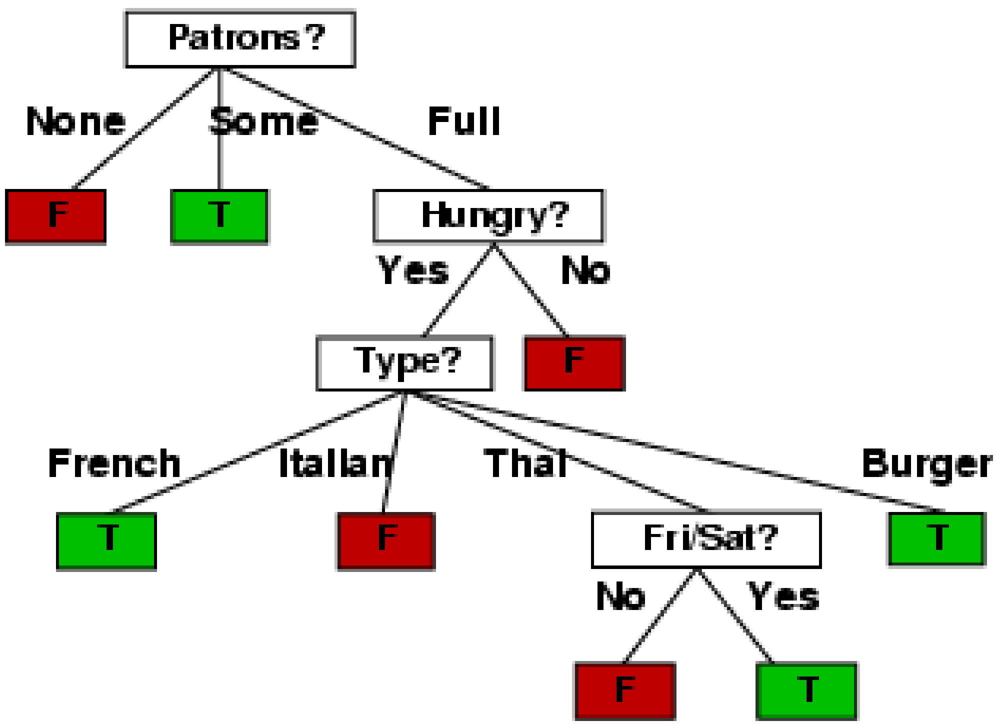
Next Feature…

- For only data where *patron = Full*
- *hungry:*
  - If value is *Yes*… we need more tests
  - If value is *No*… all output= *No*
- *type:*
  - If value is *French*… all output= *No*
  - If value is *Italian*… all output= *No*
  - If value is *Thai*… we need more tests
  - If value is *Burger*… we need more tests
- …

- So *hungry* is more discriminating (only 1 new branch)…

A Better Decision Tree

- 4 tests instead of 9
- 11 branches instead of 21



source: Norvig (2003)

Essential Information Theory

- Developed by Shannon in the 1940s
- Notion of entropy (information content)
- Measure how "predictable" a RV is...
  - If you already have a good idea about the answer (e.g. 90/10 split)
    → low entropy
  - If you have no idea about the answer (e.g. 50/50 split)
    → high entropy

Choosing the Next Attribute

- The key problem is choosing which feature to split a given set of examples
- Most used strategy: information theory

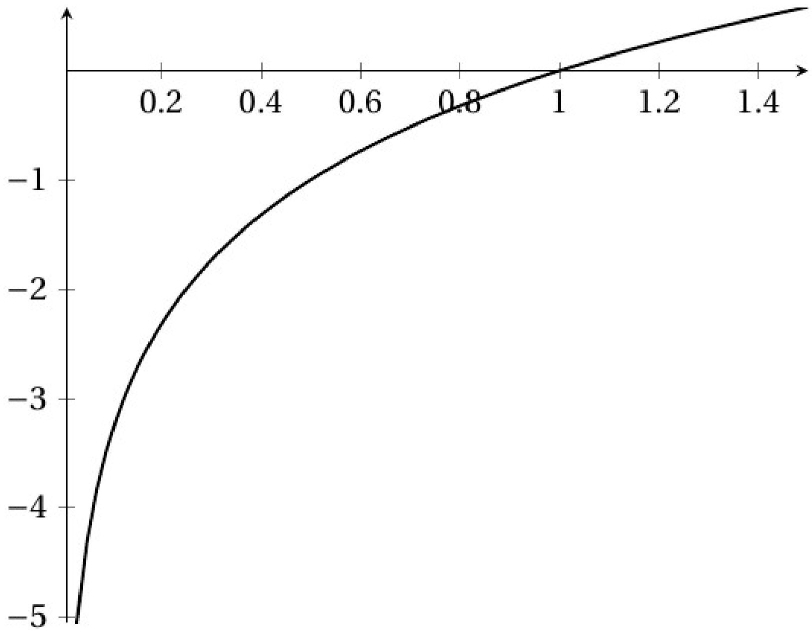
$$H(X) = -\sum_{x_i \in X} p(x_i)\log_2 p(x_i)$$ Entropy (or information content)

$$H(\text{fair coin toss}) = -\sum_{x_i \in X} p(x_i)\log_2 p(x_i) = H\left(\frac{1}{2}, \frac{1}{2}\right)$$

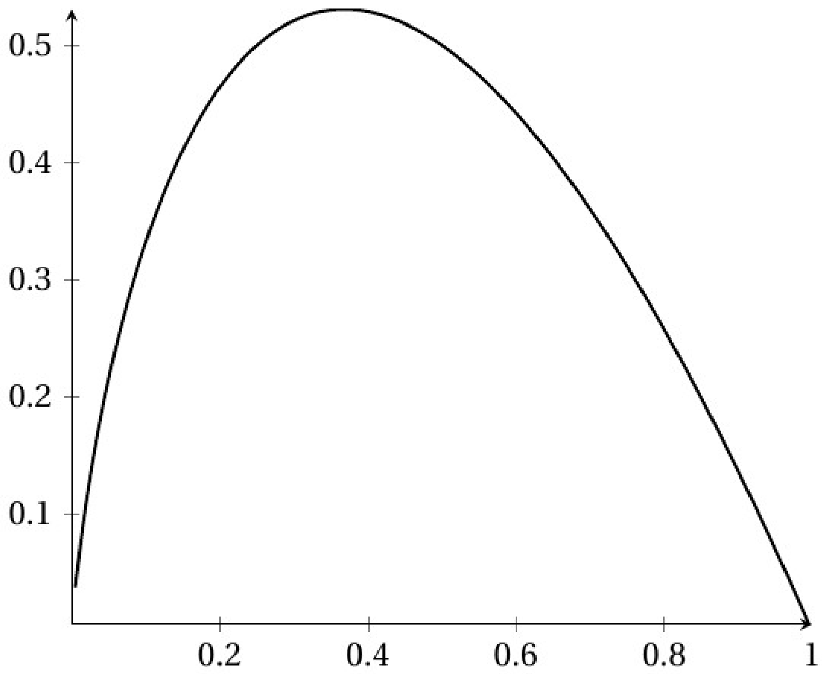
$$= \left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right) = 1 \text{ bit}$$

entropy of a fair coin toss (the RV) with 2 possible outcomes, each with a probability of 1/2

Why $-p(x)\cdot\log_2(p(x))$



$f(x) = \log_2(x)$



$f(x) = -x \cdot \log_2(x)$

Entropy

- Let X be a discrete random variable (RV) with *i* possible outcomes x_i
- Entropy (or information content)

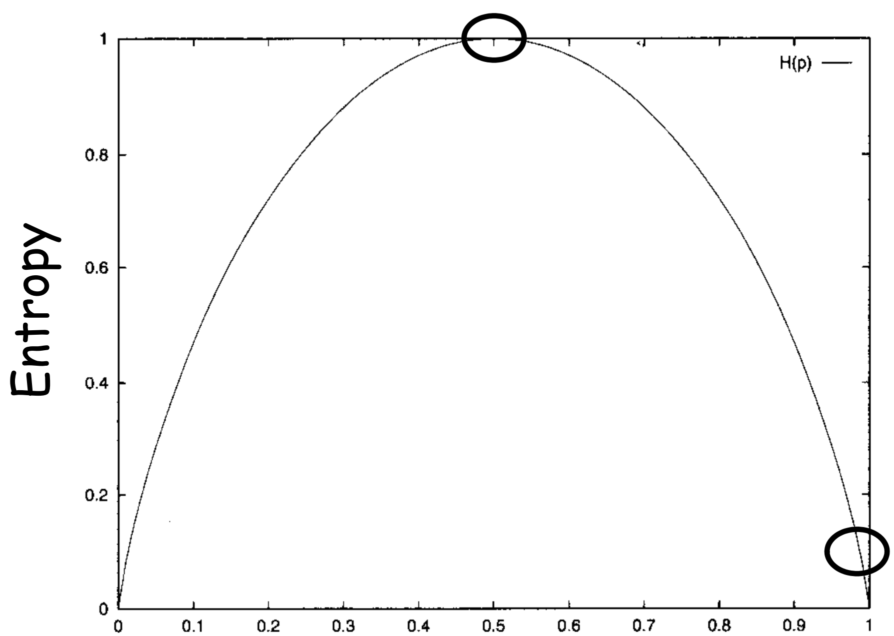
$$H(X) = -\sum_{i=1}^{n} p(x_i)\log_2 p(x_i)$$

- measures the *amount of information* in a RV
  - *average uncertainty* of a RV
  - *the average length of the message* needed to transmit an outcome x_i of that variable
- measured in bits
- for only 2 outcomes x_1 and x_2, then $1 \geq H(X) \geq 0$

→ *Worksheet #4 (Information Content)*

Example: The Coin Flip

- Fair coin: $H(X) = -\sum_{i=1}^{n} p(x_i)\log_2 p(x_i) = -\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right) = 1\text{ bit}$
- Rigged coin: $H(X) = -\sum_{i=1}^{n} p(x_i)\log_2 p(x_i) = -\left(\frac{99}{100}\log_2\frac{99}{100} + \frac{1}{100}\log_2\frac{1}{100}\right) = 0.08\text{ bits}$



fair coin -> high entropy

rigged coin -> low entropy

P(head)

→ ***Worksheet #4 (Entropy)***

Choosing the Best Feature (con't)

- The "discriminating power" of an attribute A given a data set S
- Let Values(A) = the set of values that attribute A can take
- Let S_v = the set of examples in the data set which have value v for attribute A (for each value v from Values(A))

information gain
(or entropy reduction)

$$gain(S, A) = H(S) - H(S \mid A)$$

$$= H(S) - \sum_{v \in values(A)} \frac{|S_v|}{|S|} \times H(S_v)$$

Some Intuition

Size	Color	Shape	Output
Big	Red	Circle	+
Small	Red	Circle	+
Small	Red	Square	-
Big	Blue	Circle	-

- *Size* is the least discriminating attribute (i.e. smallest information gain)
- *Shape* and *color* are the most discriminating attributes (i.e. highest information gain)

A Small Example (1)

Size	**Color**	**Shape**	**Output**
Big	Red	Circle	+
Small	Red	Circle	+
Small	Red	Square	-
Big	Blue	Circle	-

Values(Color) = {red,blue}

Color

Red: 2+ 1- | **Blue: 0+ 1-**

$$\text{gain}(S, \text{Color}) = H(S) - \sum_{v \in \text{values(Color)}} \frac{|S_v|}{|S|} \times H(S_v)$$

$$H(S) = -\left(\frac{2}{4}\log_2\frac{2}{4} + \frac{2}{4}\log_2\frac{2}{4}\right) = 1$$

for each v of Values(Color)

$$H(S \mid \text{Color} = \text{red}) = H\left(\frac{2}{3}, \frac{1}{3}\right) = -\left(\frac{2}{3}\log_2\frac{2}{3} + \frac{1}{3}\log_2\frac{1}{3}\right) = 0.918$$

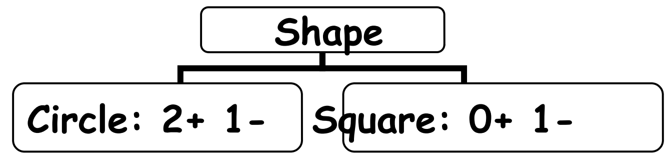
$$H(S \mid \text{Color} = \text{blue}) = H(1,0) = -\left(\frac{1}{1}\log_2\frac{1}{1}\right) = 0$$

$$H(S \mid \text{Color}) = \frac{3}{4}(0.918) + \frac{1}{4}(0) = 0.6885$$

$$\text{gain}(\text{Color}) = H(S) - H(S \mid \text{Color}) = 1 - 0.6885 = 0.3115$$

A Small Example (2)

Size	Color	Shape	Output
Big	Red	Circle	+
Small	Red	Circle	+
Small	Red	Square	-
Big	Blue	Circle	-



$$H(S) = -\left(\frac{2}{4}\log_2\frac{2}{4} + \frac{2}{4}\log_2\frac{2}{4}\right) = 1$$

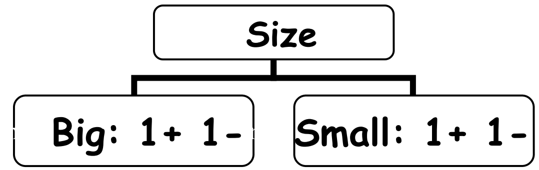
Note: by definition,
- $\log 0 = -\infty$
- $0\log 0$ is 0

$$H(S \mid Shape) = \frac{3}{4}(0.918) + \frac{1}{4}(0) = 0.6885$$

$$gain(Shape) = H(S) - H(S \mid Shape) = 1 - 0.6885 = 0.3115$$

A Small Example (3)

Size	Color	Shape	Output
Big	Red	Circle	+
Small	Red	Circle	+
Small	Red	Square	-
Big	Blue	Circle	-



$$H(S) = -\left(\frac{2}{4}\log_2\frac{2}{4} + \frac{2}{4}\log_2\frac{2}{4}\right) = 1$$

→ Worksheet #4 ("Information Gain")

A Small Example (4)

Size	**Color**	**Shape**	**Output**
Big	Red	Circle	+
Small	Red	Circle	+
Small	Red	Square	-
Big	Blue	Circle	-

$$gain(Shape) = 0.3115$$

$$gain(Color) = 0.3115$$

$$gain(Size) = 0$$

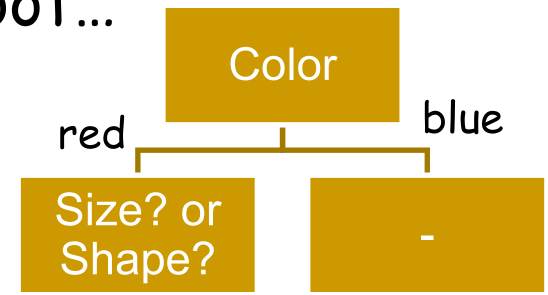
- So first separate according to either *color* or *shape* (root of the tree)

A Small Example (5)

- Let's assume we pick *Color* for the root…

Size	Color	Shape	Output
Big	Red	Circle	+
Small	Red	Circle	+
Small	Red	Square	-
Big	Blue	Circle	-

S_2



$$H(S_2) = -\left(\frac{2}{3}\log_2\frac{2}{3} + \frac{1}{3}\log_2\frac{1}{3}\right)$$

for each v of Values(Size)

$$H(S_2 \mid Size = big) = H\left(\frac{1}{1}, \frac{0}{1}\right) = 0$$

$$H(S_2 \mid Size = small) = H\left(\frac{1}{2}, \frac{1}{2}\right) = 1$$

$$H(S_2 \mid Size) = \frac{1}{3}(0) + \frac{2}{3}(1)$$

$$gain(Size) = H(S_2) - H(S_2 \mid Size)$$

for each v of Values(Shape)

$$H(S_2 \mid Shape = circle) = H\left(\frac{2}{2}, \frac{0}{2}\right) = 0$$

$$H(S_2 \mid Shape = square) = H\left(\frac{0}{1}, \frac{1}{1}\right) = 0$$

$$H(S_2 \mid Shape)$$

$$gain(Shape) = H(S_2) - H(S_2 \mid Shape)$$

Back to the Restaurant

- Training data:

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

source: Norvig (2003)

The Restaurant Example

$gain(alt) = \ldots \quad gain(bar) = \ldots \quad gain(fri) = \ldots \quad gain(hun) = \ldots$

$$gain(pat) = 1 - \left(\frac{2}{12} \times H\left(\frac{0}{2}, \frac{2}{2}\right) + \frac{4}{12} \times H\left(\frac{0}{4}, \frac{4}{4}\right) + \frac{6}{12} \times H\left(\frac{2}{6}, \frac{4}{6}\right) \right)$$

$$= 1 - \left(\frac{2}{12} \times - \left(\frac{0}{2} \log_2 \frac{0}{2} + \frac{2}{2} \log_2 \frac{2}{2} \right) + \frac{4}{12} \times - \left(\frac{0}{4} \log_2 \frac{0}{4} + \frac{4}{4} \log_2 \frac{4}{4} \right) + \ldots \right) \approx 0.541 \text{bits}$$

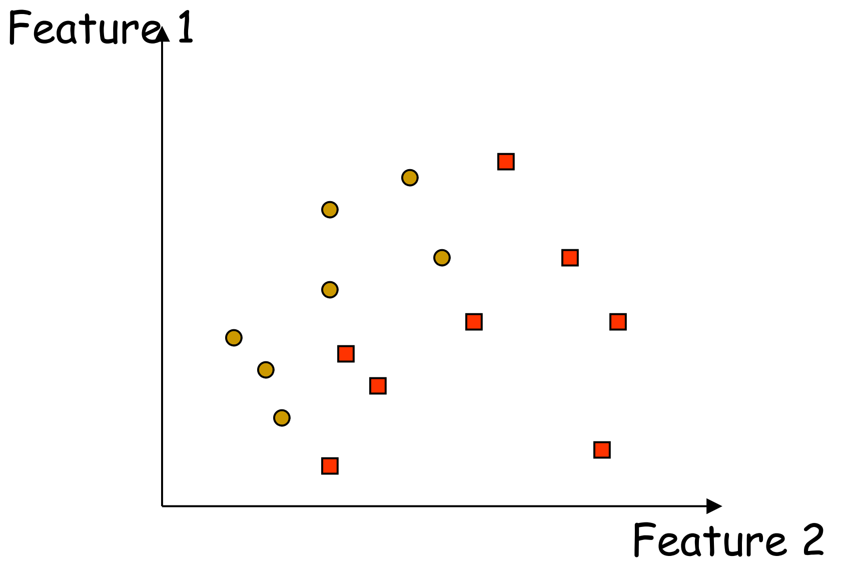
$gain(price) = \ldots \quad gain(rain) = \ldots \quad gain(res) = \ldots$

$$gain(type) = 1 - \left(\frac{2}{12} \times H\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{2}{12} \times H\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{4}{12} \times H\left(\frac{2}{4}, \frac{2}{4}\right) + \frac{4}{12} \times H\left(\frac{2}{4}, \frac{2}{4}\right) \right) = 0 \text{ bits}$$

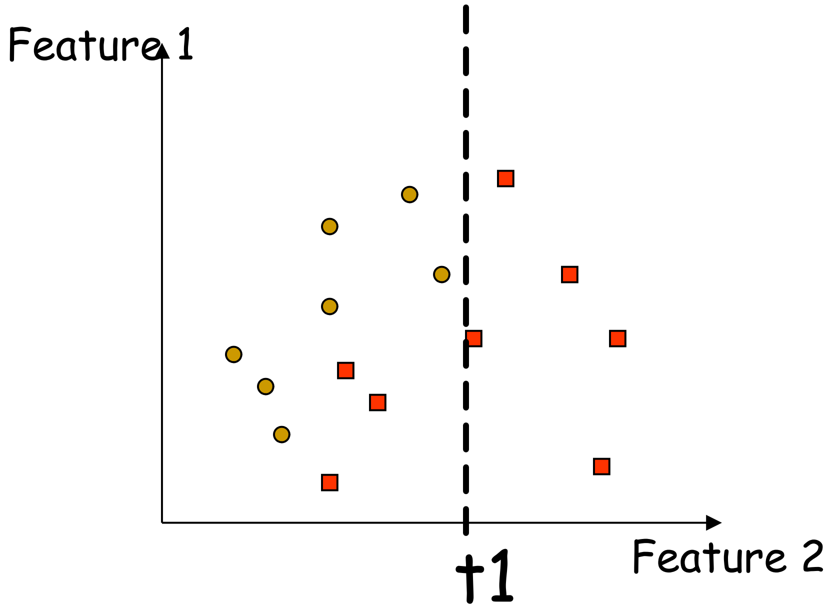
$gain(est) = \ldots$

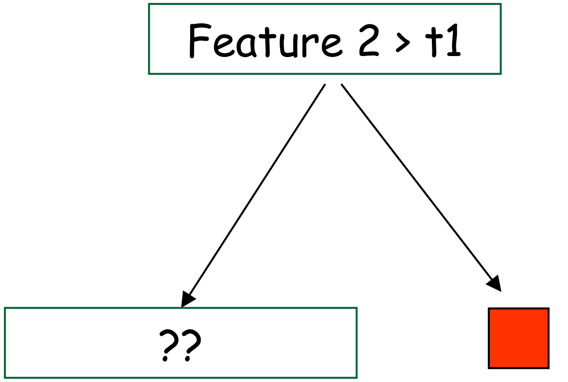
- Attribute pat (Patron) has the highest gain, so root of the tree should be attribute *Patrons*
- do recursively for subtrees

Decision Boundaries of Decision Trees

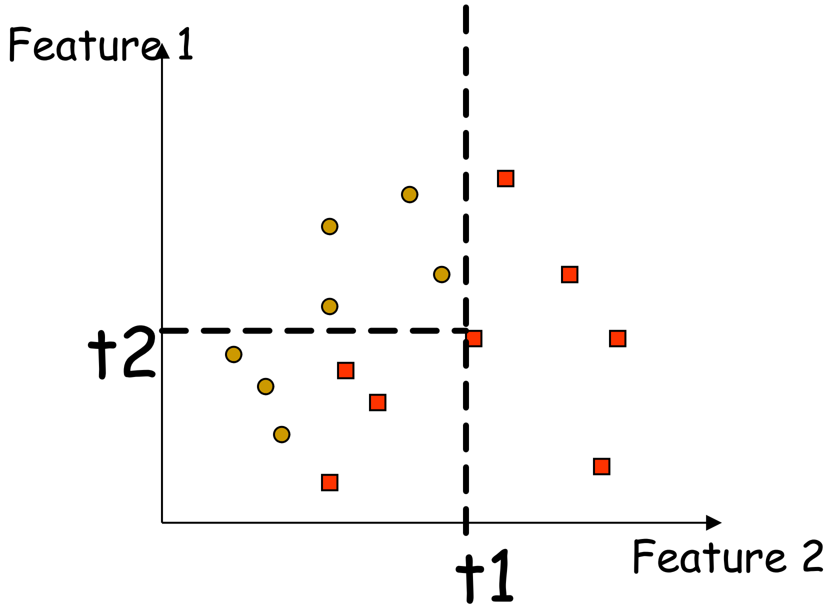


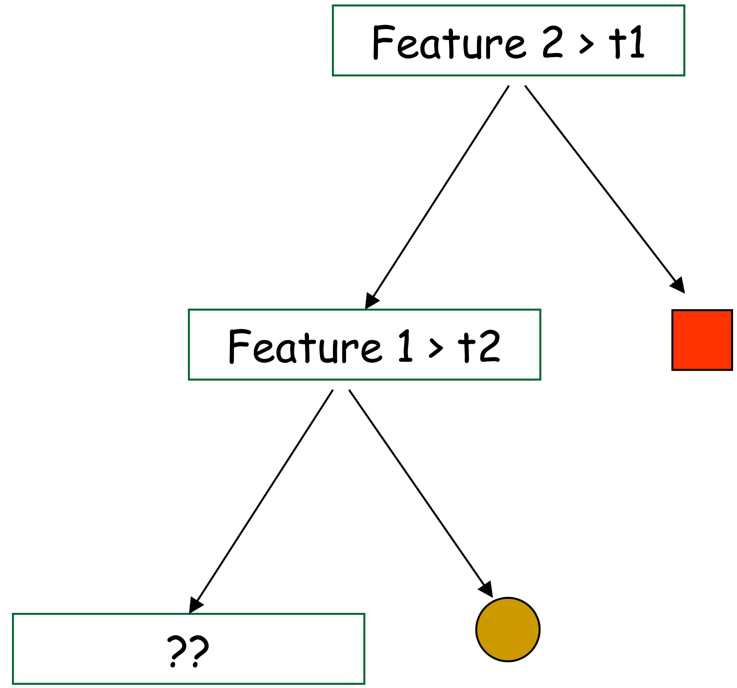
Decision Boundaries of Decision Trees



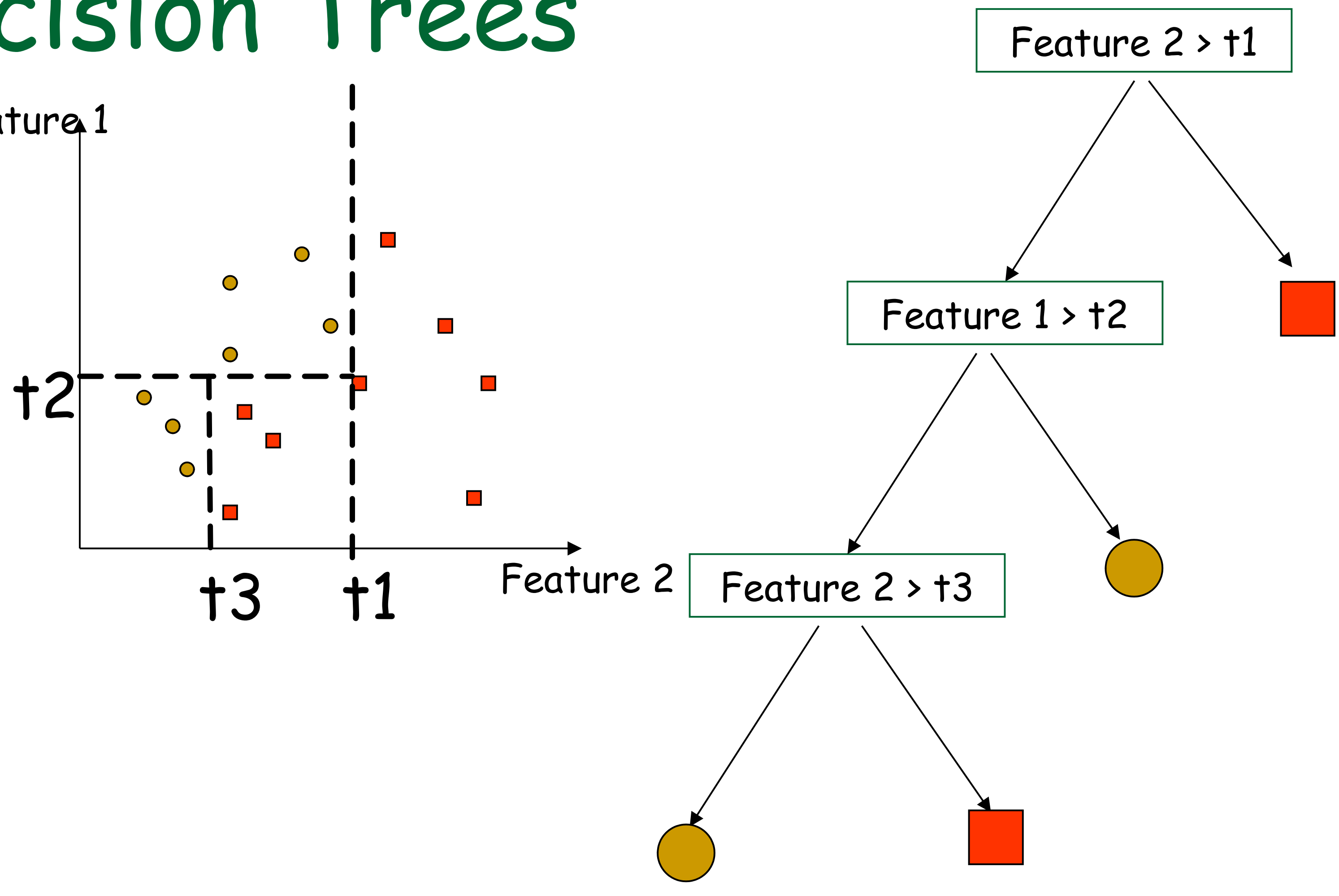


Decision Boundaries of Decision Trees





Decision Boundaries of Decision Trees

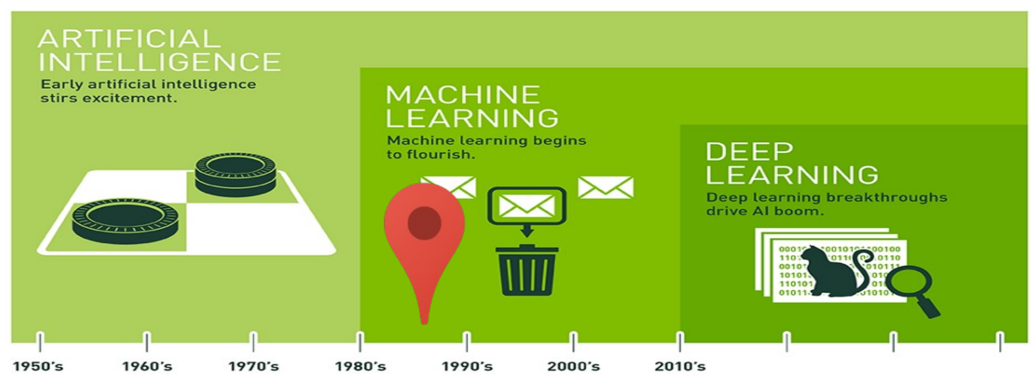


Applications of Decision Trees

- One of the most widely used learning methods in practice
  - Fast, simple, and traceable
- Can out-perform human experts in many problems

Today

1. Introduction to ML (contd.)
2. Decision Trees
3. Evaluation (contd.)
4. Unsupervised Learning: k-means Clustering



https://medium.com/machine-learning-for-humans/neural-networks-deep-learning-cdad8aeae49b

Metrics, revisited

- Accuracy
  - % of instances of the test set the algorithm correctly classifies
  - when all classes are equally important and represented
- Recall, Precision & F-measure
  - when one class is more important and the others

Confusion Matrix

- Not all errors are equal
  - Type I error (FP) might be worse than Type II error (FN) (depends on the application, e.g., spam filtering)
  - *"It is better to risk saving a guilty man than to condemn an innocent one."* (Voltaire)

Model says…	In reality, the instance is… in class *C*	Is not in class *C*
instance is in class *C*	True Positive (TP)	False Positive (FP)
instance is NOT in class *C*	False Negative (FN)	True Negative (TN)

$$\text{Precision} = \frac{TP}{TP+FP} \qquad \text{Recall} = \frac{TP}{TP+FN} \qquad \text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

Evaluation: A Single Value Measure

- cannot take mean of P&R
  - if R = 50% P = 50% M = 50%
  - if R = 100% P = 10% M = 55% (not fair)
- take harmonic mean

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

HM is high only when both P&R are high

if R = 50% and P = 50% HM = 50%

if R = 100% and P = 10% HM = 18.2%

- take <u>weighted</u> harmonic mean

w_r: weight of R w_p: weight of P $a = 1/w_r$ $b = 1/w_p$

$$WHM = \frac{a+b}{\left(a/R + b/P\right)} = \frac{(a+b)/b}{\left(a/Rb + b/Pb\right)} = \frac{a/b + 1}{\left(a/bR + 1/P\right)}$$

  - let $\beta^2 = a/b$

$$WHM = \frac{\beta^2 + 1}{\left(\beta^2/R + 1/P\right)} = \frac{(\beta^2 + 1)\,PR}{\beta^2 P + R}$$

… which is called the F-measure

Evaluation: the F-measure

- A weighted combination of precision and recall

$$F = \frac{(\beta^2 + 1)PR}{(\beta^2 P + R)}$$

- β represents the relative importance of precision and recall
  - when $\beta = 1$, precision & recall have same importance
  - when $\beta > 1$, precision is favored
  - when $\beta < 1$, recall is favored

Example

	Target	system 1	system 2	system 3
	X1	X1	X1	X1
	X2	X2	X2	X2
	X3	X3	X3	X3
	X4	X4	X4	X4
	X5	X5	X5	X5
	X6	X6	X6	X6
	X7	X7	X7	X7
	…	…	…	…
	…	…	…	…
	X500	X500	X500	X500
Accuracy		495 /500 = 99 % !	498/500 = 99.6%	498/500 = 99.6%
Precision		0/0	3/3 = 100%	5/7 = 71%
Recall		0/5 = 0%	3/5 = 60%	5/5 = 100%
F1 -measure (B=1)		Undef	2PR/P+R =	2PR/P+R =

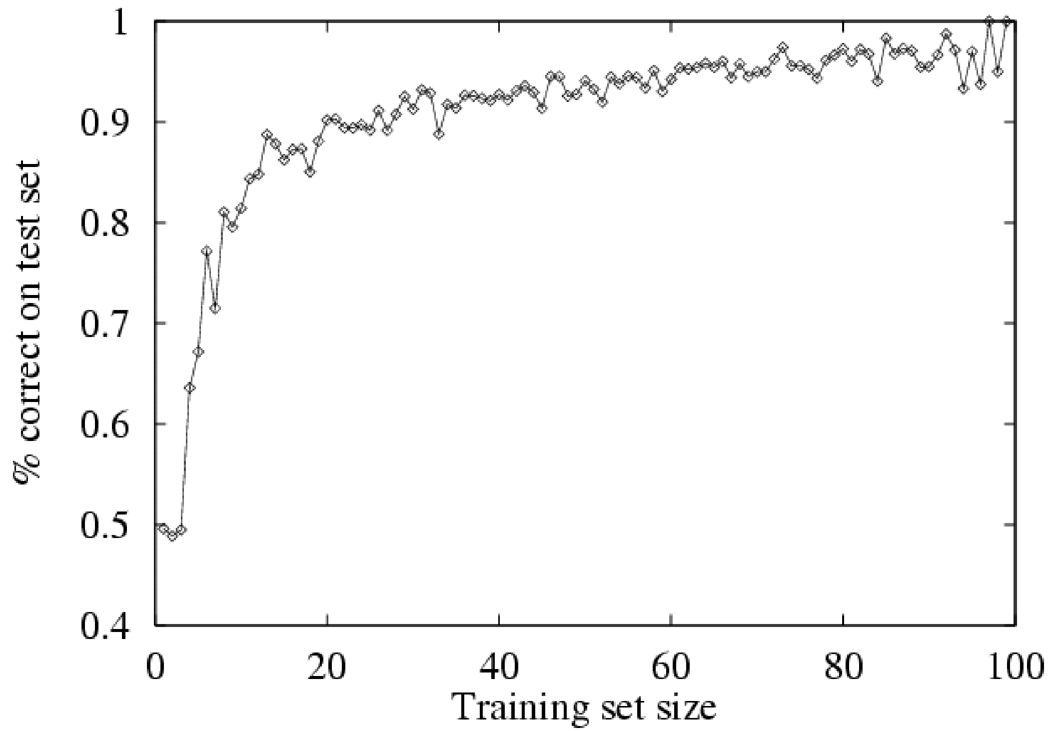
→ ***Worksheet #4 (F-Measure)***

Error Analysis

- Where did the learner go wrong ?
- Use a confusion matrix / contingency table

correct class (that should have been assigned)	classes assigned by the learner							
	C1	**C2**	**C3**	**C4**	**C5**	**C6**	…	**Total**
C1	94	3	0	0	3	0		100
C2	0	93	3	4	0	0		100
C3	0	1	94	2	1	2		100
C4	0	1	3	94	2	0		100
C5	0	0	3	2	92	3		100
C6	0	0	5	0	10	85		100
…								

A Learning Curve



- Size of training set
  - the more, the better
  - but after a while, not much improvement…

source: Mitchell (1997)

Some Words on Training

- In all types of learning… watch out for:
  - Noisy input
  - Overfitting/underfitting the training data

Noisy Input

- In all types of learning… watch out for:
  - Noisy Input:
    - Two examples have the same feature-value pairs, but different outputs

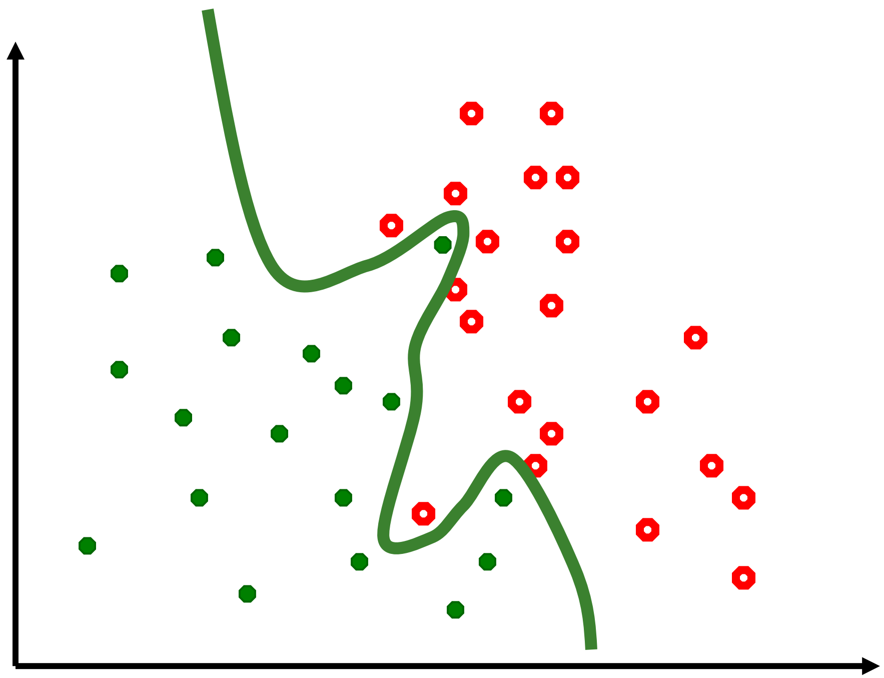
Size	Color	Shape	Output
Big	Red	Circle	+
Big	Red	Circle	-

    - Some values of features are incorrect or missing (ex. errors in the data acquisition)
    - Some relevant attributes are not taken into account in the data set

Overfitting

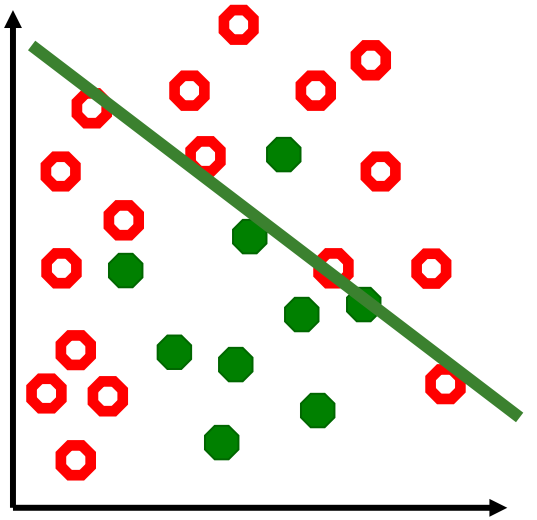
- If a large number of irrelevant features are there, we may find meaningless regularities in the data that are particular to the training data but irrelevant to the problem.
- Complicated boundaries *overfit* the data
- they are too tuned to the particular training data at hand
- They do not *generalize* well to the new data
- Extreme case: "rote learning"

- Training error is low
- Testing error is high



Underfitting

- We can also underfit data, i.e. use too simple decision boundary
- Model is not expressive enough (not enough features)
- There is no way to fit a linear decision boundary so that the training examples are well separated



- Training error is high
- Testing error is high

Cross-validation

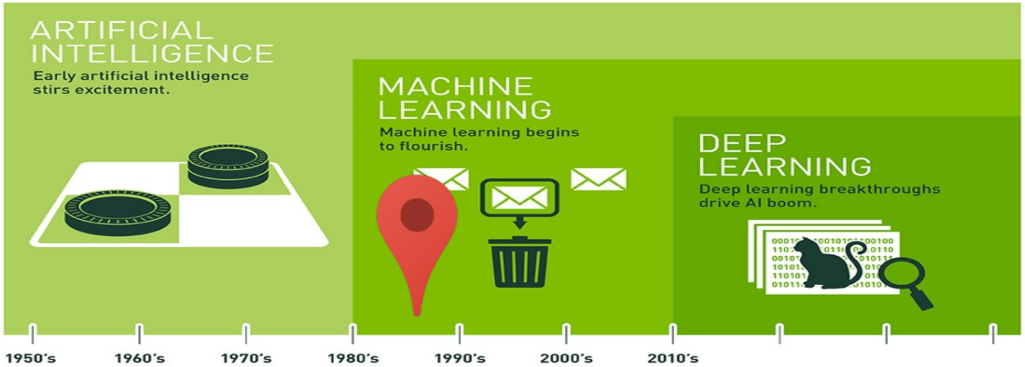
- K-fold cross-validation
  - run k experiments, each time you test on 1/k of the data, and train on the rest
  - than you average the results

- ex: 10-fold cross validation
  1. Collect a large set of examples (all with correct classifications)
  2. Divide collection into two disjoint sets: training (90%) and test (10% = 1/k)
  3. Apply learning algorithm to training set
  4. Measure performance with the test set
  5. Repeat steps 2-4, with the 10 different portions
  6. Average the results of the 10 experiments

<table>
<tr><td>exp1:</td><td colspan="9">train</td><td>test</td></tr>
<tr><td>exp2:</td><td colspan="8">train</td><td>test</td><td>train</td></tr>
<tr><td>exp3:</td><td colspan="7">train</td><td>test</td><td colspan="2">train</td></tr>
<tr><td>…</td><td>…</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

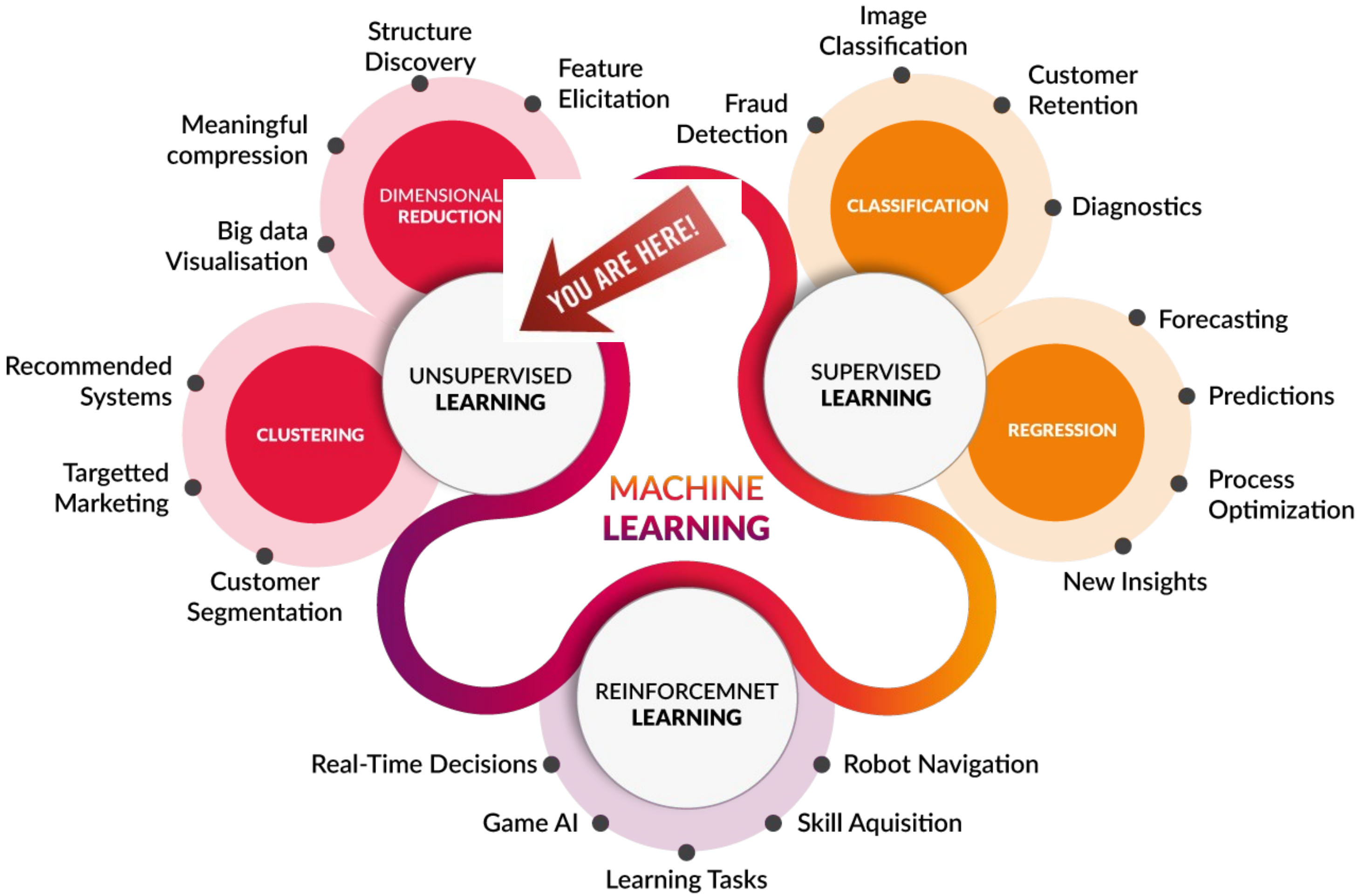
Today

1. Introduction to ML (contd.)
2. Decision Trees
3. Evaluation (contd.)
4. Unsupervised Learning: k-means Clustering



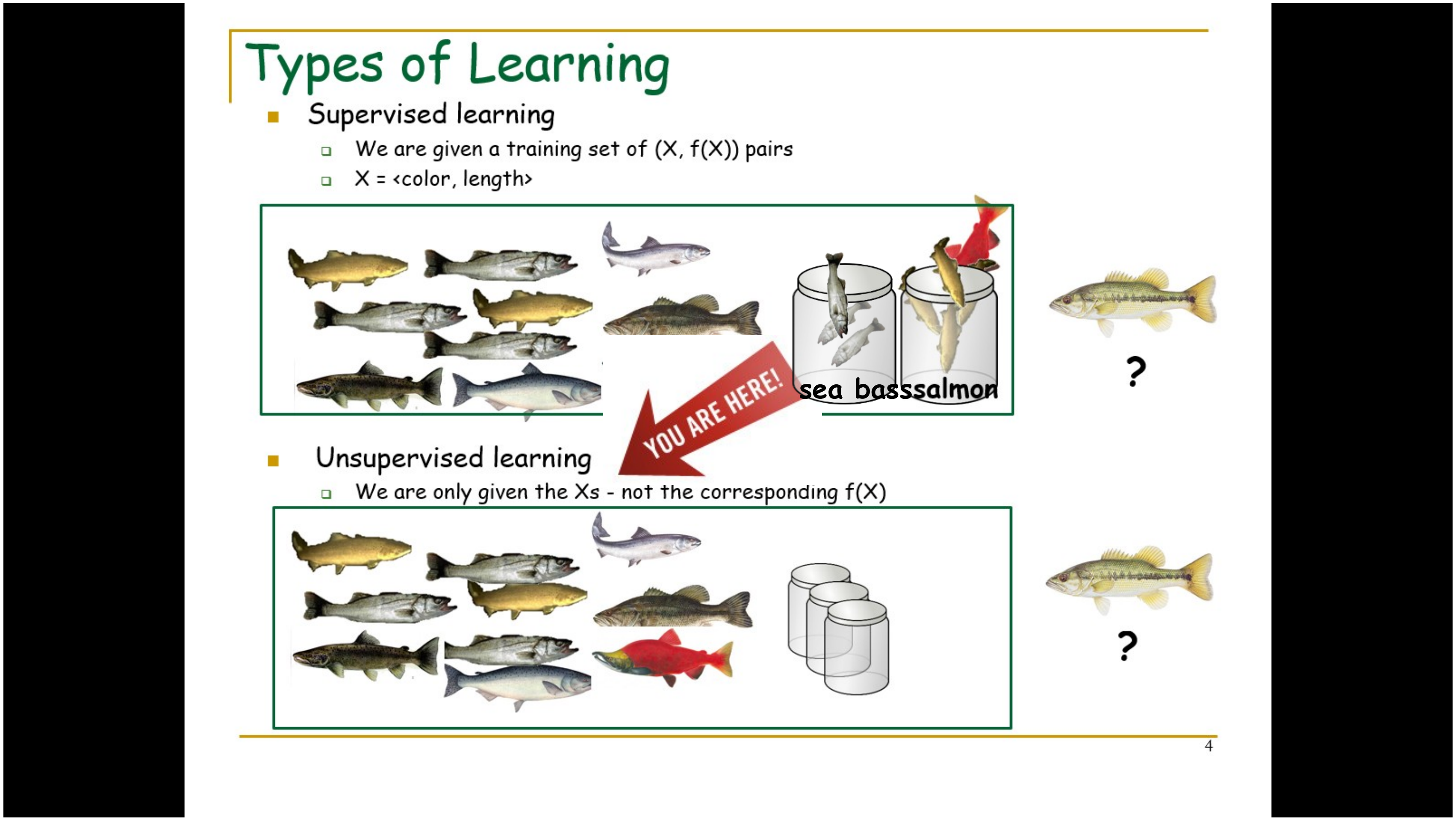
https://medium.com/machine-learning-for-humans/neural-networks-deep-learning-cdad8aeae49b

Types of Machine Learning



http://www.cognub.com/index.php/cognitive-platform/

Remember this slide?



Unsupervised Learning

- Learn without labeled examples
  - i.e. X is given, but not f(X)

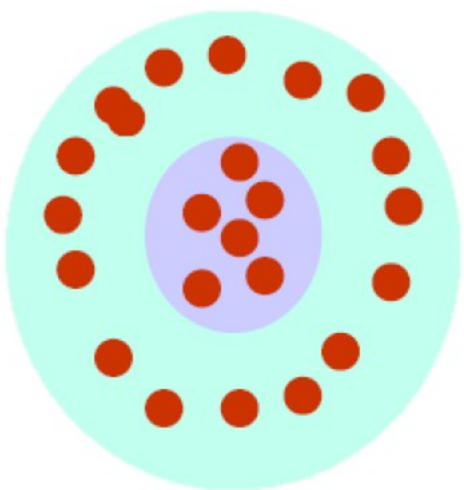
small nose	big teeth	small eyes	moustache	f(X) = ?

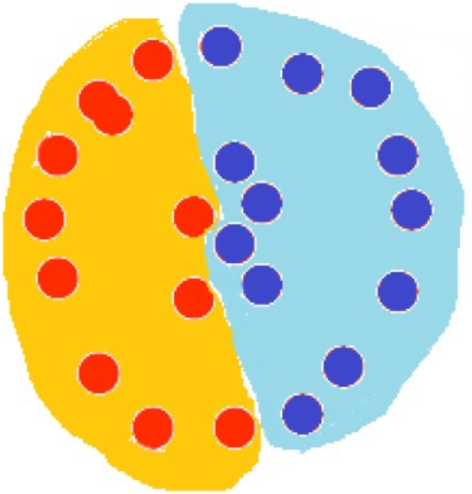
- Without a f(X), you can't really identify/label a test instance

- But you can:
  - Cluster/group the features of the test data into a number of groups
  - Discriminate between these groups without actually labeling them

What is Clustering

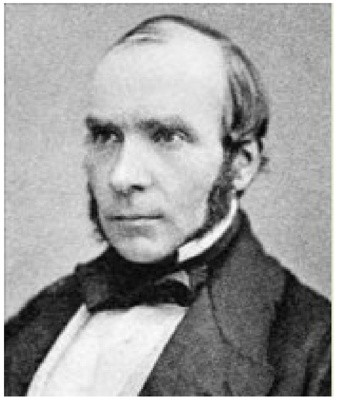
- The organization of unlabeled data into similarity groups called clusters.
- A cluster is a collection of data items which are "similar" between them, and "dissimilar" to data items in other clusters.

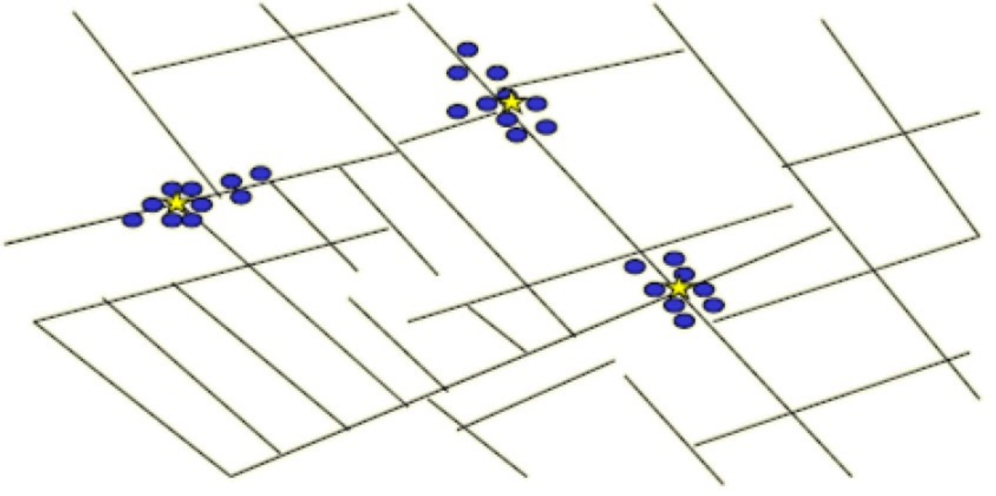




Historic Application of Clustering

- John Snow, a London physician plotted the location of cholera on a map during an outbreak in the 1850s.
- The locations indicated that cases were clustered arounds certain intersections where there were polluted wells – thus exposing both the problem and the solution.

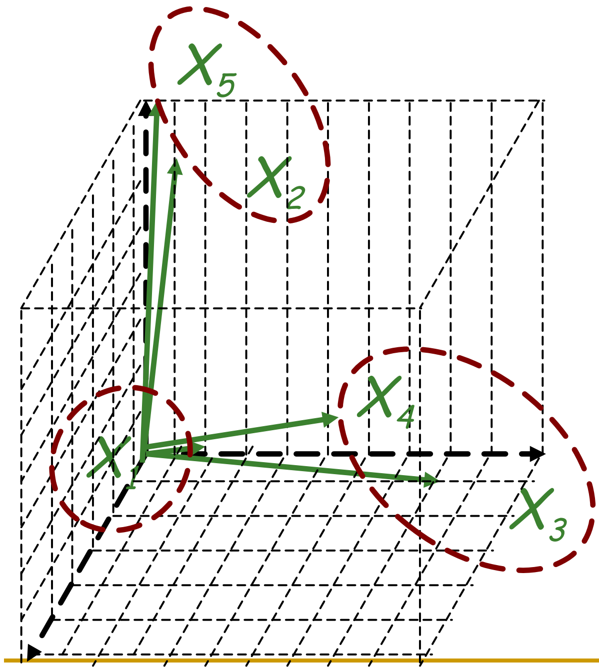




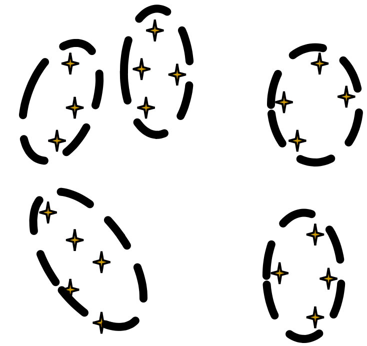
Clustering

- Represent each instance as a vector $\langle a_1, a_2, a_3, \ldots, a_n \rangle$
- Each vector can be visually represented in an *n*-dimensional space

	a_1	a_2	a_3	Output
X_1	1	0	0	?
X_2	1	6	0	?
X_3	8	0	1	?
X_4	6	1	0	?
X_5	1	7	1	?

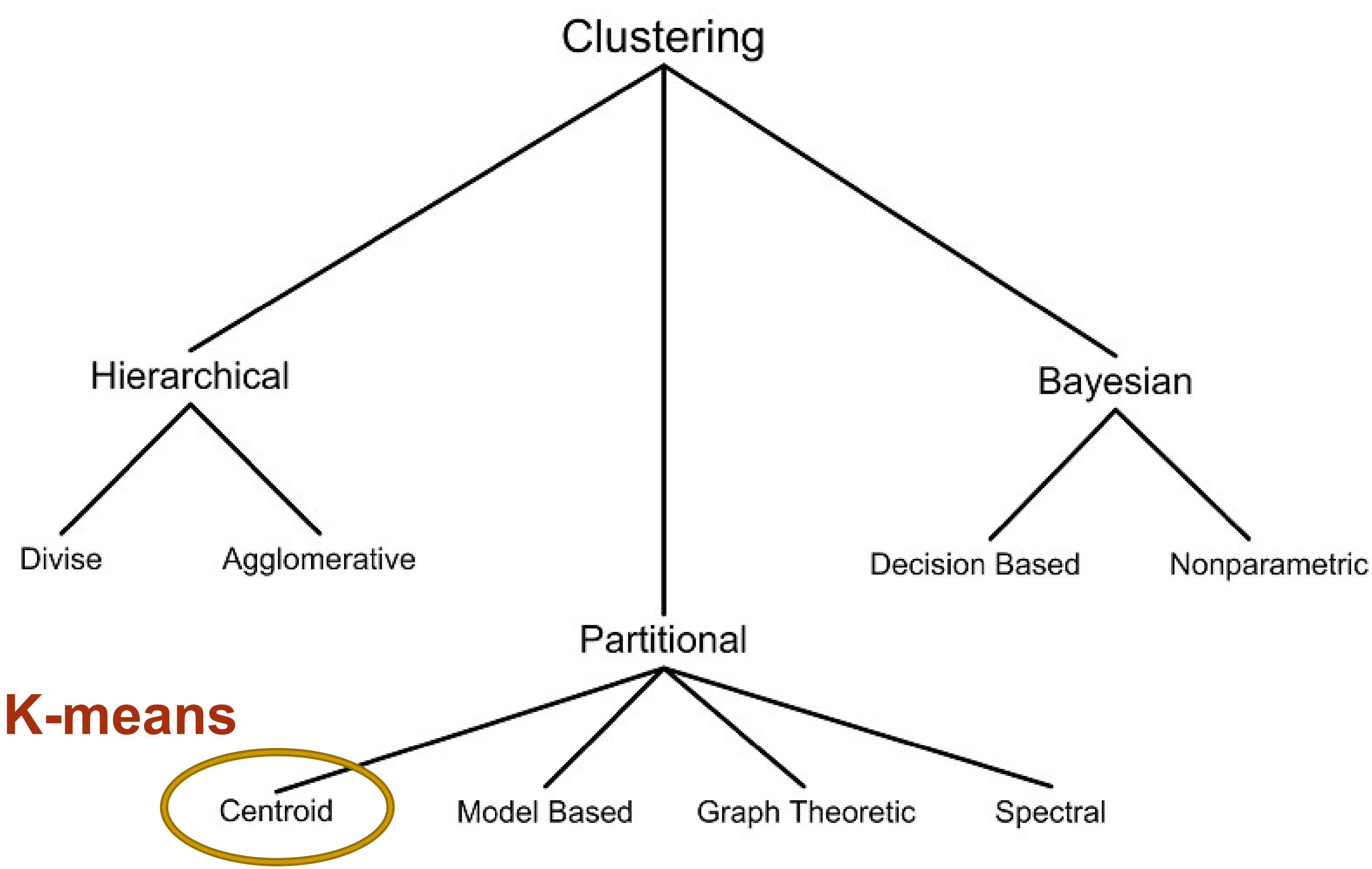


Clustering



- Clustering algorithm
  - Represent test instances on a n dimensional space
  - Partition them into regions of high density
    - How? ... many algorithms (ex. k-means)
  - Compute the centroïd of each region as the average of data points in the cluster

Clustering Techniques



k-means Clustering

- User selects how many clusters they want... (the value of k)

1. Place k points into the space (ex. at random). These points represent initial group centroïds.

2. Assign each data point x_n to the nearest centroïd.

3. When all data points have been assigned, recalculate the positions of the K centroïds as the average of the cluster

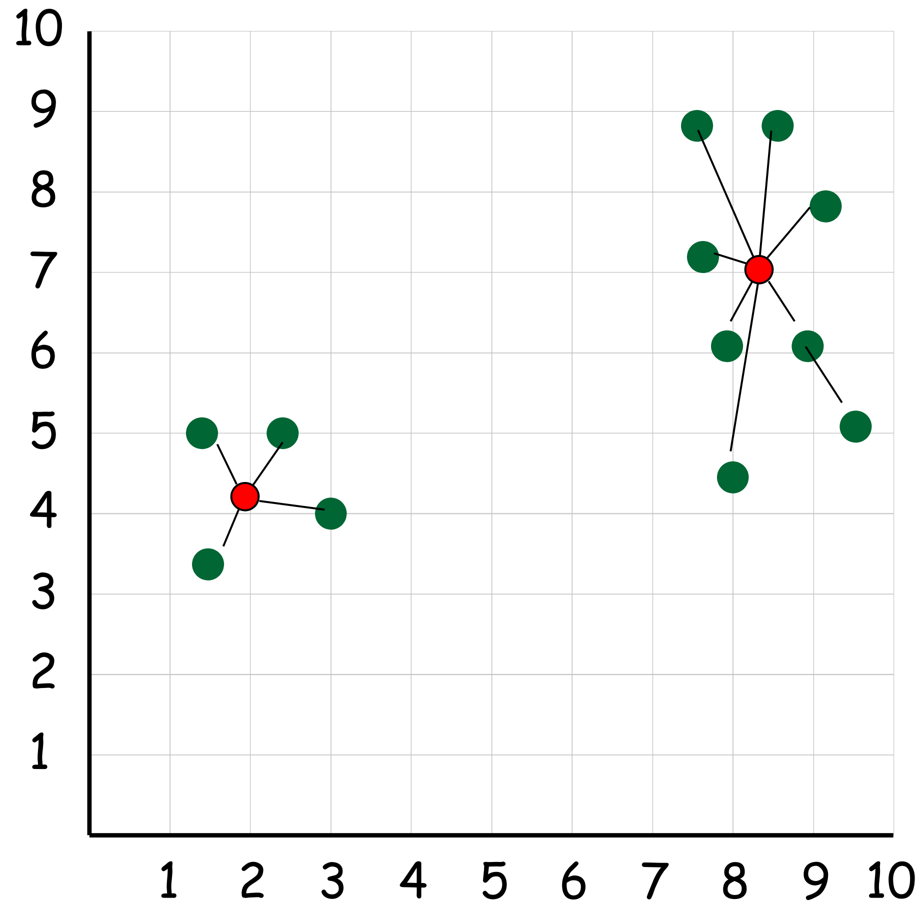
4. Repeat Steps 2 and 3 until none of the data instances change group.

Euclidean Distance

- To find the nearest centroïd…
- a possible metric is the Euclidean distance
- distance between 2 pts
  $p = (p_1, p_2, \ldots, p_n)$
  $q = (q_1, q_2, \ldots, q_n)$

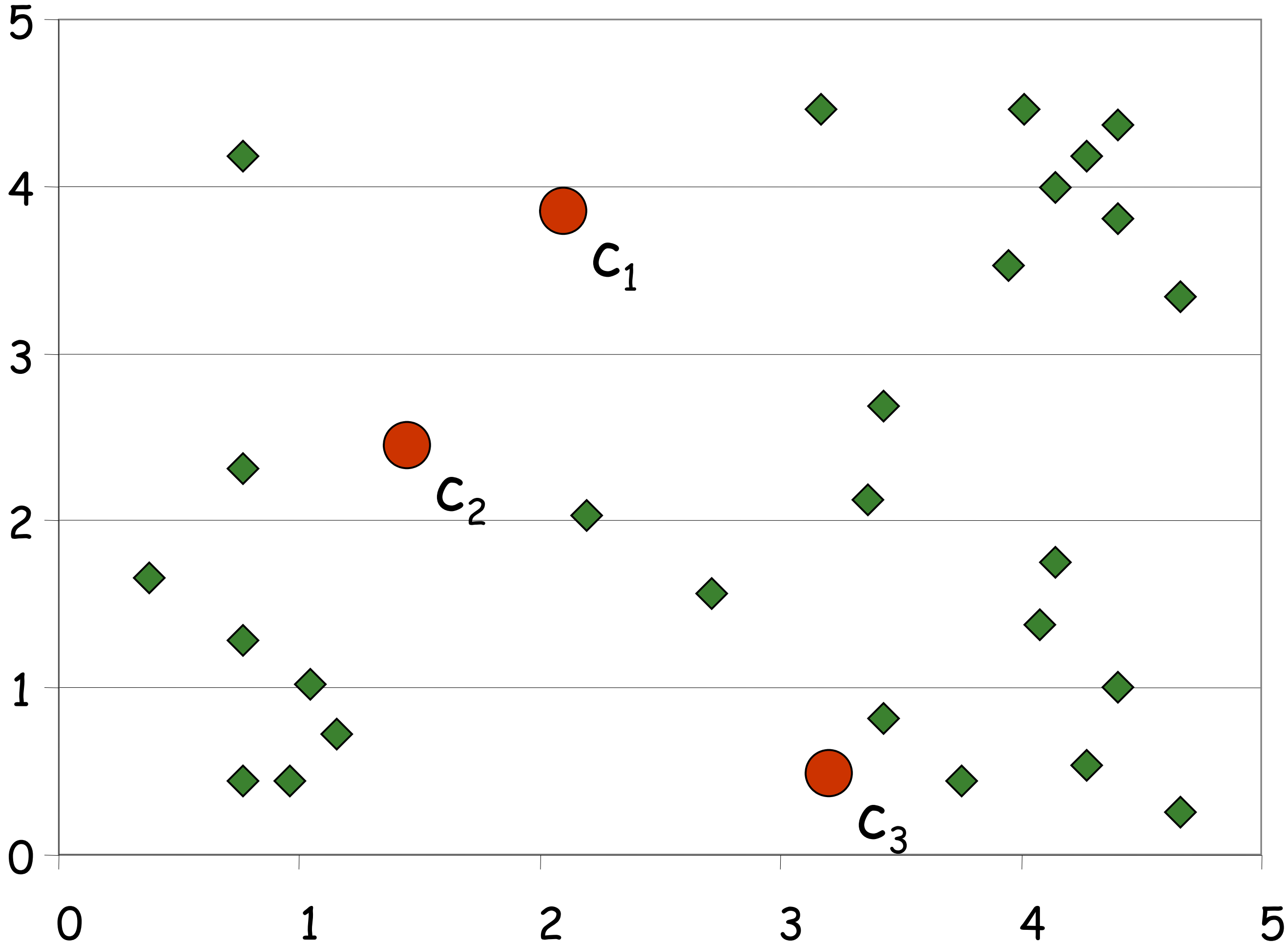
$$d = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

- where to assign a data point x?
- For all k clusters, chose the one where x has the smallest distance



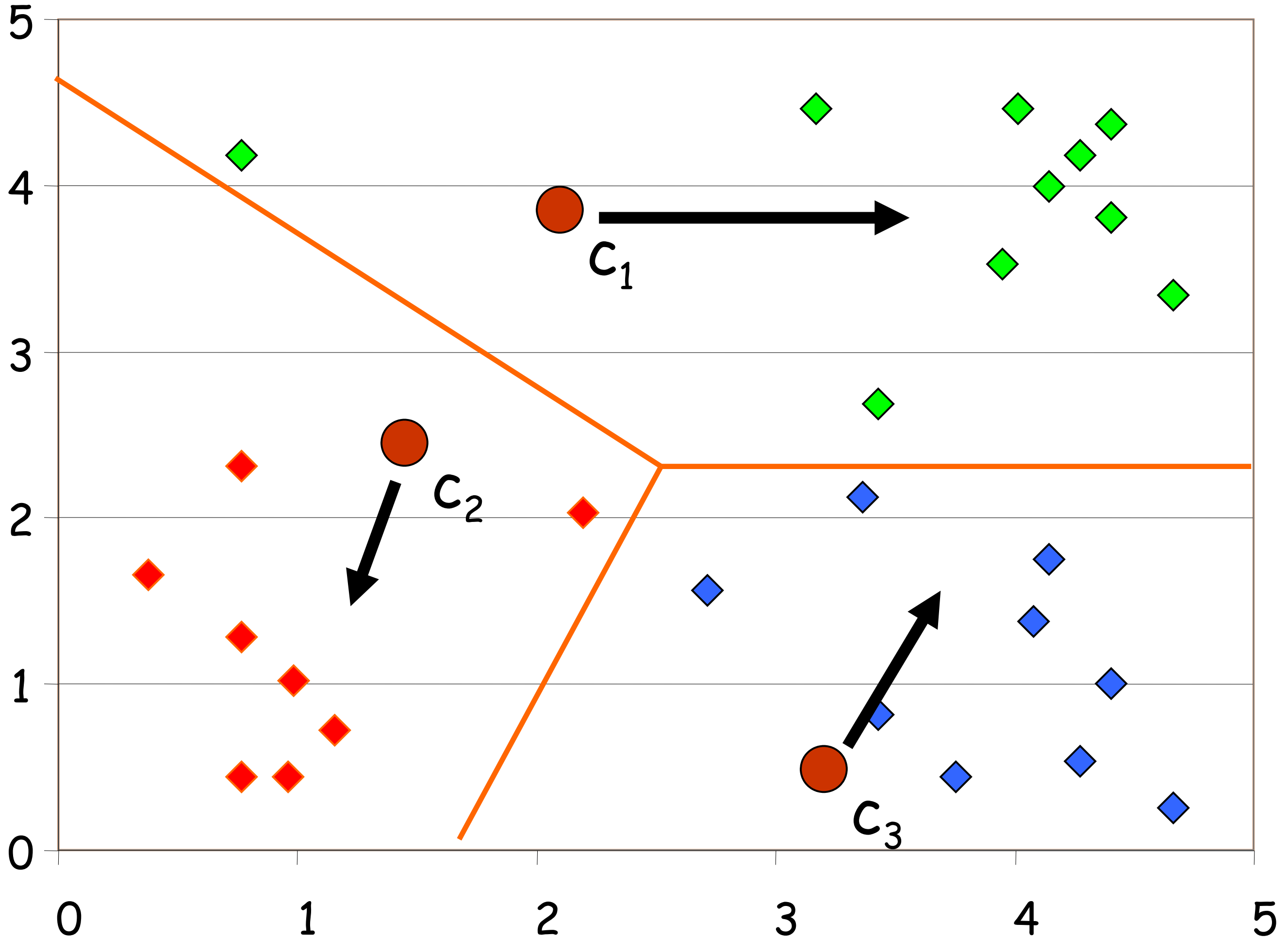
Example (in 2-D… i.e. 2 features)

initial 3 centroïds (ex. at random)



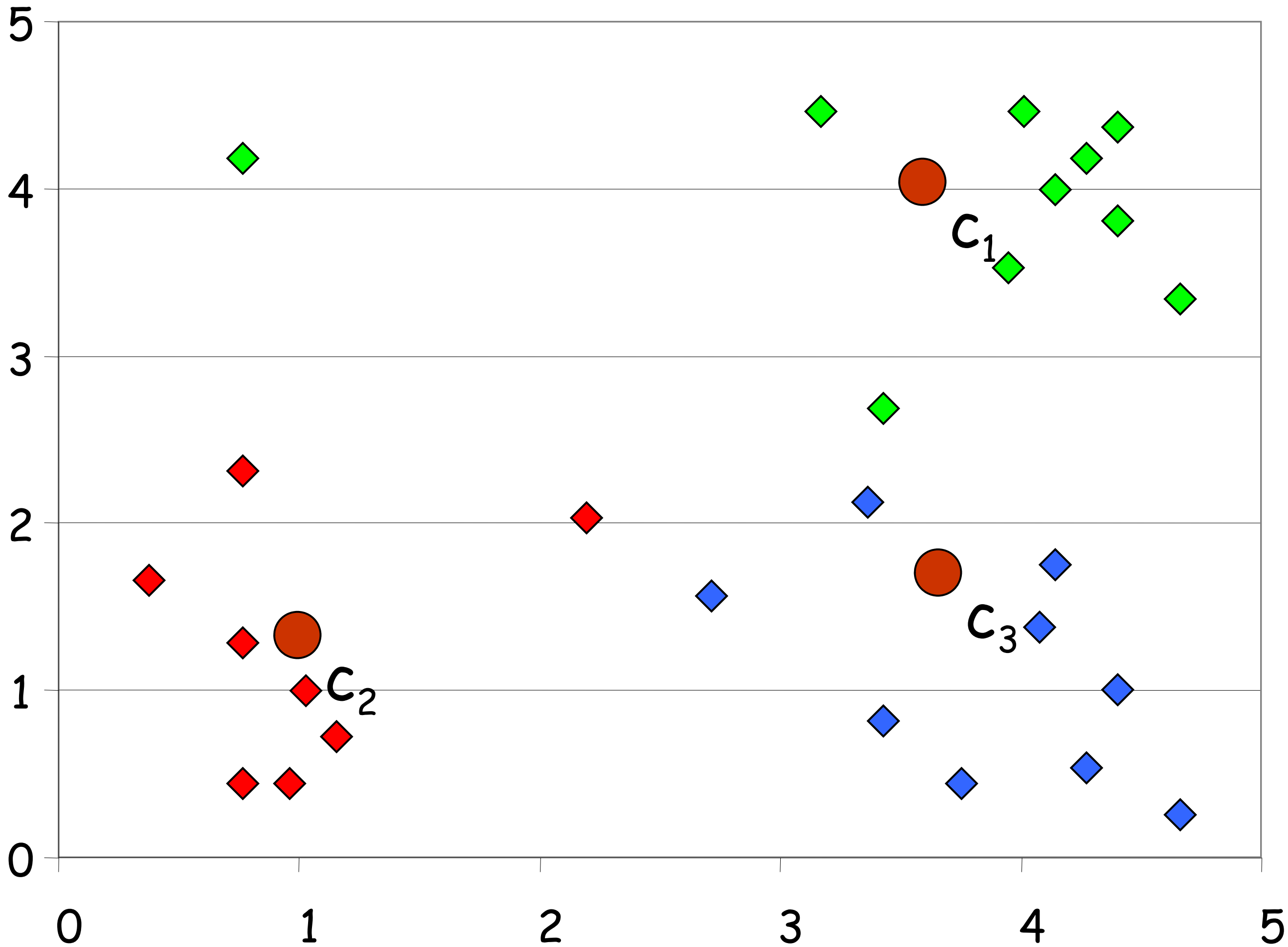
Example

partition data points to closest centroïd



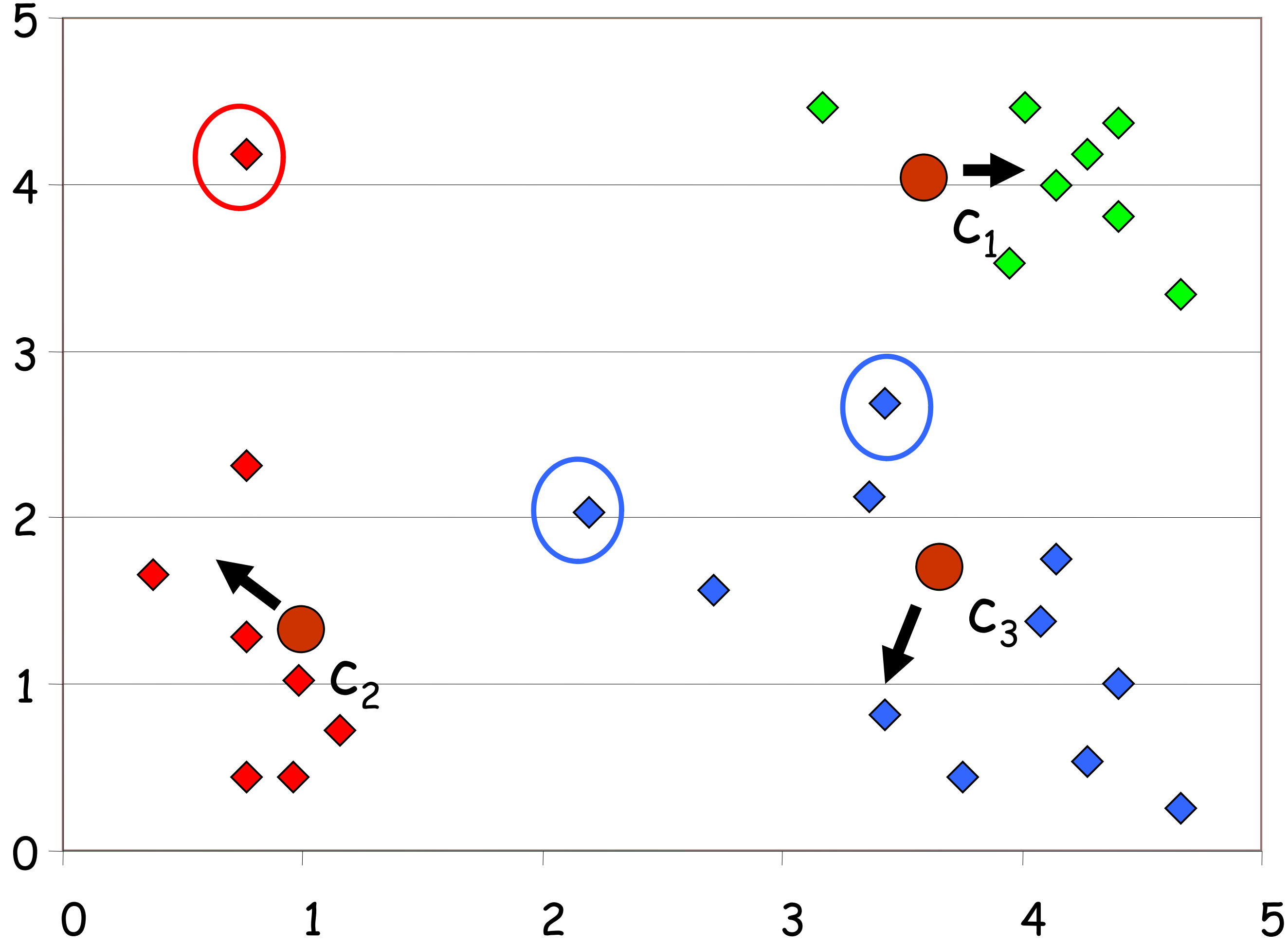
Example

re-compute new centroïds

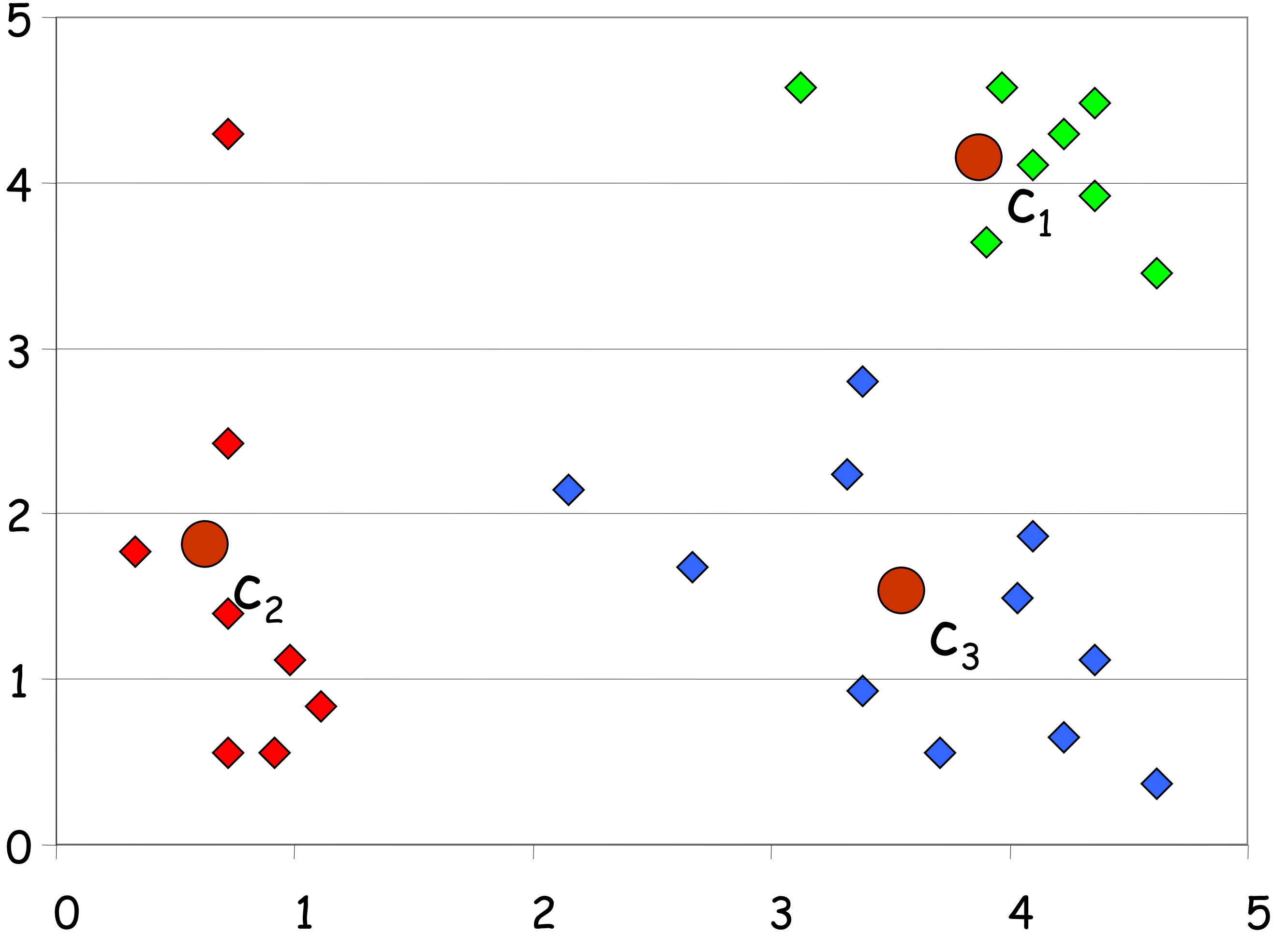


Example

re-assign data points to new closest centroïds



Example



→ ***Worksheet #4 (k-Means Clustering)***

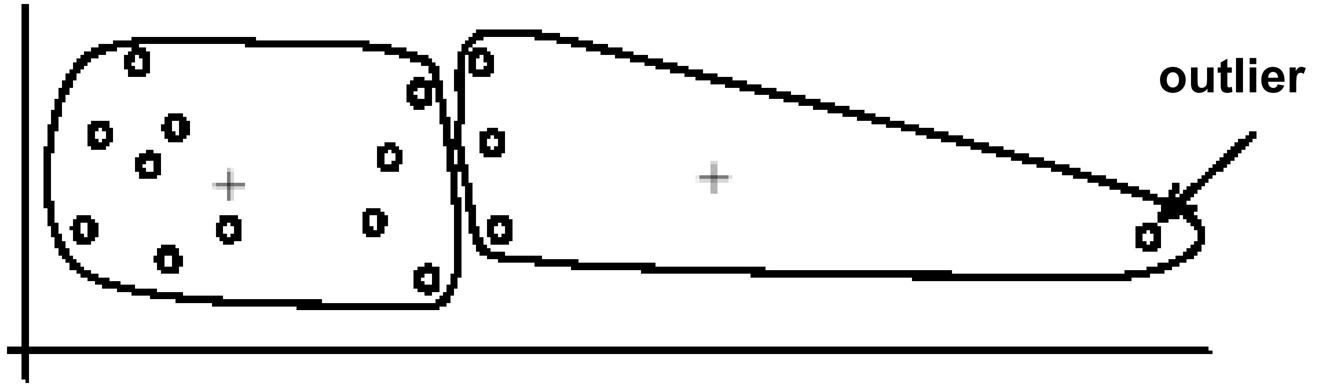
Why use k-means?

- Strengths:
  - Simple
    - Easy to understand and implement
  - Efficient: Time complexity $O(t \cdot k \cdot n)$
    - n number of data points
    - k number of clusters
    - t number of iterations
  - With small k and t, linear performance on practical problems

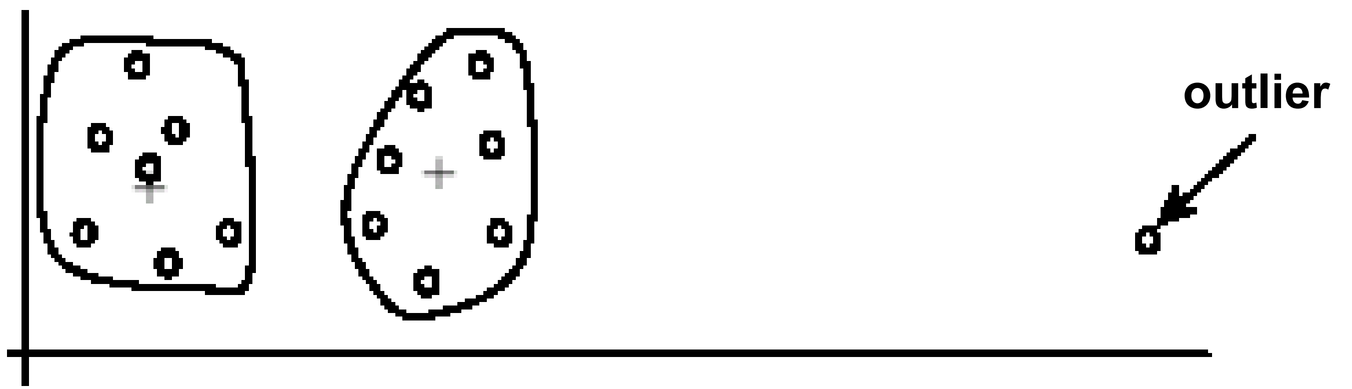
Weakness of k-means

- User needs to specify k
- Algorithm is sensitive to outliers
  - i.e., data points that are far away from others
  - Could be errors in the data or special data points with very different characteristics

Outliers

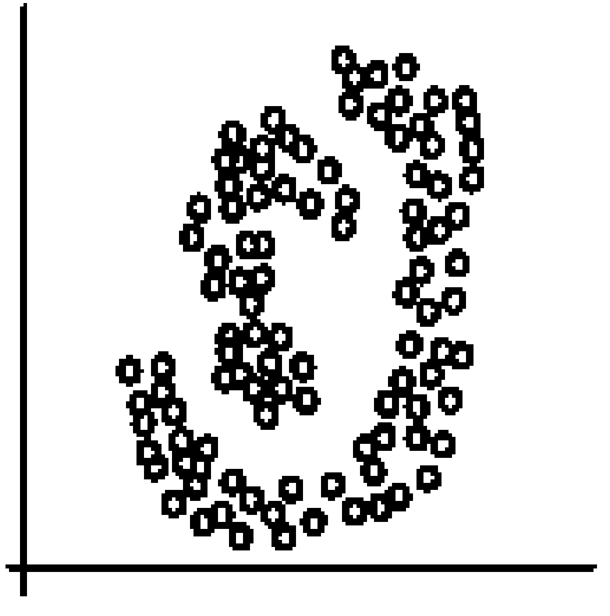


(A) Undesirable clusters

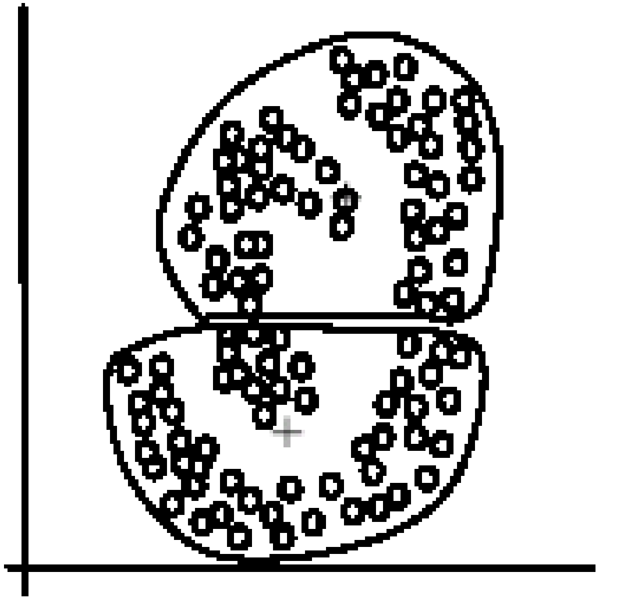


(B) Ideal clusters

Special data structures

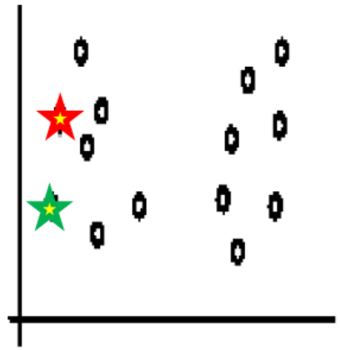


(A) Two natural clusters

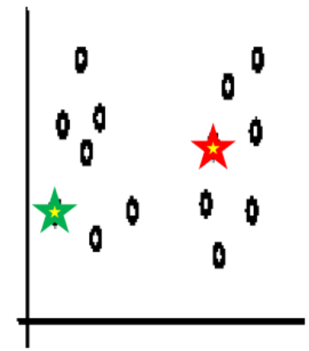


(B) k-means clusters

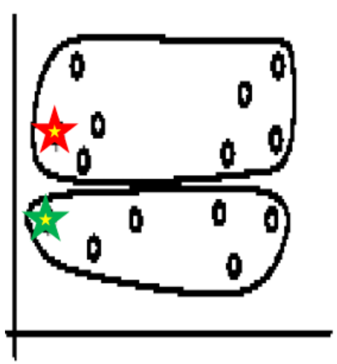
Sensitivity to initial seeds



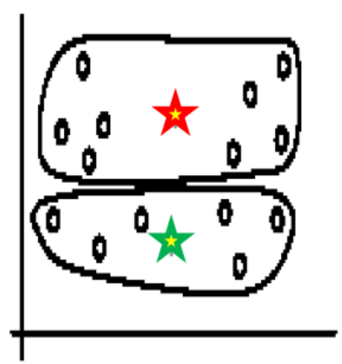
Random selection of seeds (centroids)



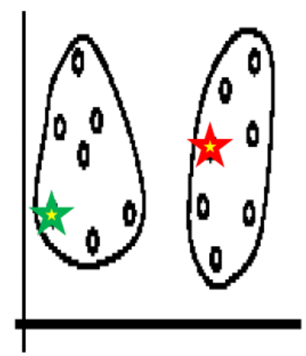
Random selection of seeds (centroids)



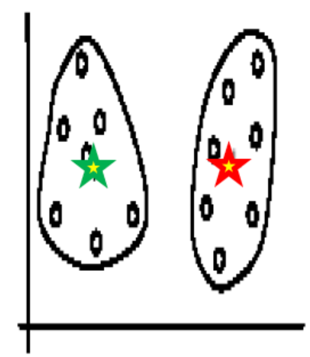
Iteration 1



Iteration 2



Iteration 1



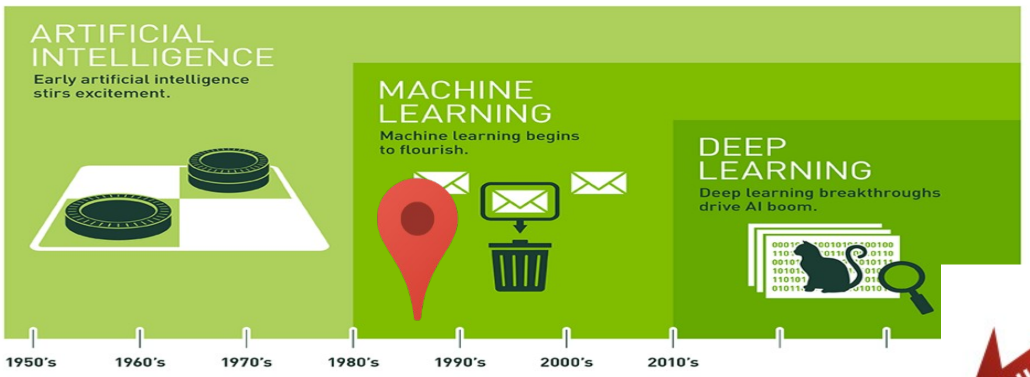
Iteration 2

K-means: Summary

- Despite weaknesses, k-means is still one of the most popular algorithms, due to its simplicity and efficiency
- No clear evidence that any other clustering algorithm performs better in general
- Comparing different clustering algorithms is a difficult task.
  No one knows the correct clusters!

Today

1. Introduction to ML (contd.)
2. Decision Trees
3. Evaluation (contd.)
4. Unsupervised Learning: k-means Clustering



https://medium.com/machine-learning-for-humans/neural-networks-deep-learning-cdad8aeae49b