

most ml algorithms are similarity-based



- **clustering \approx similarity-based \approx non-relational \approx easy**
 - 1NN is as good (1% diff) as C4.5 on some UCI tests!
 - k-means clustering (batch variant)
 - randomly divide examples e into n sets and compute mean (center) of each set
 - * reassign each e to set with nearest center
 - ** recompute centers of all sets
 - repeat * and ** until no more reassignments happen

hope that the data naturally divide into n clusters....

similarity is proximity

on-line k-means clustering



randomly initialize n data points: *centers*

- * find center closest to 1. example and move it closer to this example
- ** repeat * with next example, until all examples are considered
- repeat * and ** until all examples captured by each center remain the same

normally, *centers* are **attracted** towards actual cluster centers but there may be too many centers (centers without data points) or random center initialization prevents some centers from ever capturing data points (data points without centers)

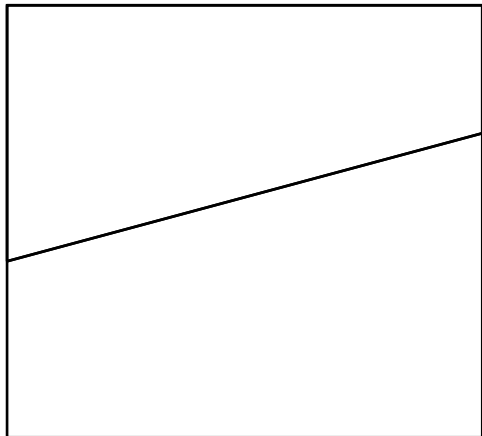
after clustering, associate unique output label with each cluster;
given new input, find its nearest cluster and output its label

note: these clusterers don't need to store all data

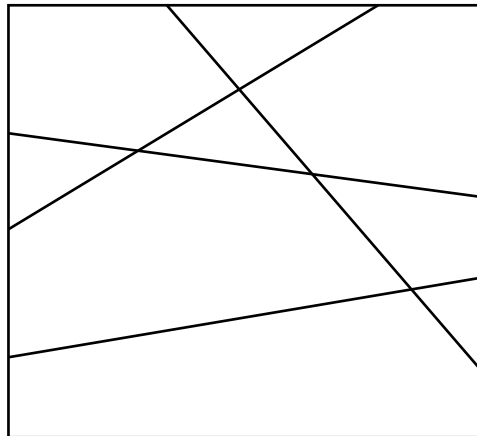
(syntactic) similarity-based learners



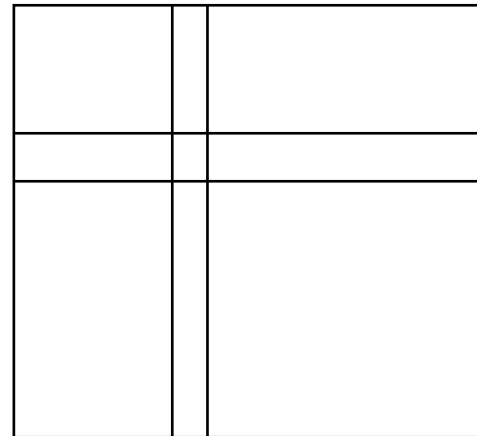
- * assume: uniformly labeled data points lie in simple regions of input space, **bounded by simple shapes** (circles, rectangles ..)
- * assume: **similarity is a major form of regularity**
- * there is a limited, given number of such *bounding constructs*
- * such learners are **interpolative**



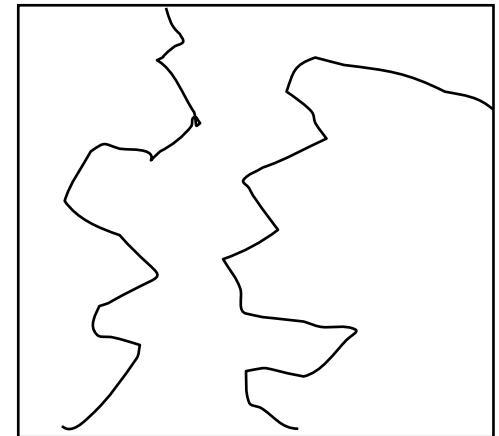
perceptron/LMS



backprop/ANN



C5/Cart



NN/LinVectQuantiz

(syntactic) similarity-based learners, cont.

- they work only to degree to which data with same output cluster together, i.e. degree to which input is *geometrically separable*
- generalize *linear separability* to measure proportion of data points whose NearestNeighbors have same output

$$\text{geomSeparability}(f) = \frac{\sum_{i=1}^n f(x_i) + f(x'_i) + 1 \bmod 2}{n}$$

f is target function, x a data point, x' its NN

similarity-based learners require hi geometric separability

relational learning: hard


- prime example of **relational** problems: **parity problems (checker board!)**

- output = 1 if there is an odd # of 1's, else 0
- note: every point has a NN with a different output label!

- try this

13	2	2	3	8	2	2	1	2	4	-->	4
8	3	6	4	6	2	8	3	8	1	-->	5
12	1	5	2	3	3	3	2	3	1	-->	4
13	4	13	3	8	2	8	1	8	3	-->	5
9	3	10	1	11	2	12	1	13	4	-->	7
10	4	10	3	1	3	1	4	10	2	-->	5
13	4	11	4	11	3	13	4	13	4	-->	5
9	2	4	2	5	2	13	2	10	2	-->	6
7	4	12	4	12	2	4	2	12	1	-->	4
13	2	8	2	1	3	1	3	1	4	-->	4
10	3	10	1	5	2	13	2	10	2	-->	4
11	3	11	4	13	1	13	1	13	3	-->	?
8	2	2	2	9	2	11	2	13	2	-->	?

relational learning: deep similarities...

- 
- surface similarity among data is not enough
 - required: **construction of a bias** to help in looking for hidden relationships among data --> leads to new concepts
 - new concepts are used in further learning
 - example: poker hands
 - bias for equality among arbitrary values leads to *n-of-a-kind* hands
 - given *n-of-a-kind* hands, it's easy to find *full-house* hands
 - bias for equality among specific variables leads to *flushes*
 - given *flushes*, it's easy to find *straight-flushes*
 - note: biased search for hidden relations produces **new concepts**, à la **theoretical concepts**, ==> creativity
 - e.g. ascending sequences of pairs, flushes with alternating suits