

# CPSC 340: Machine Learning and Data Mining

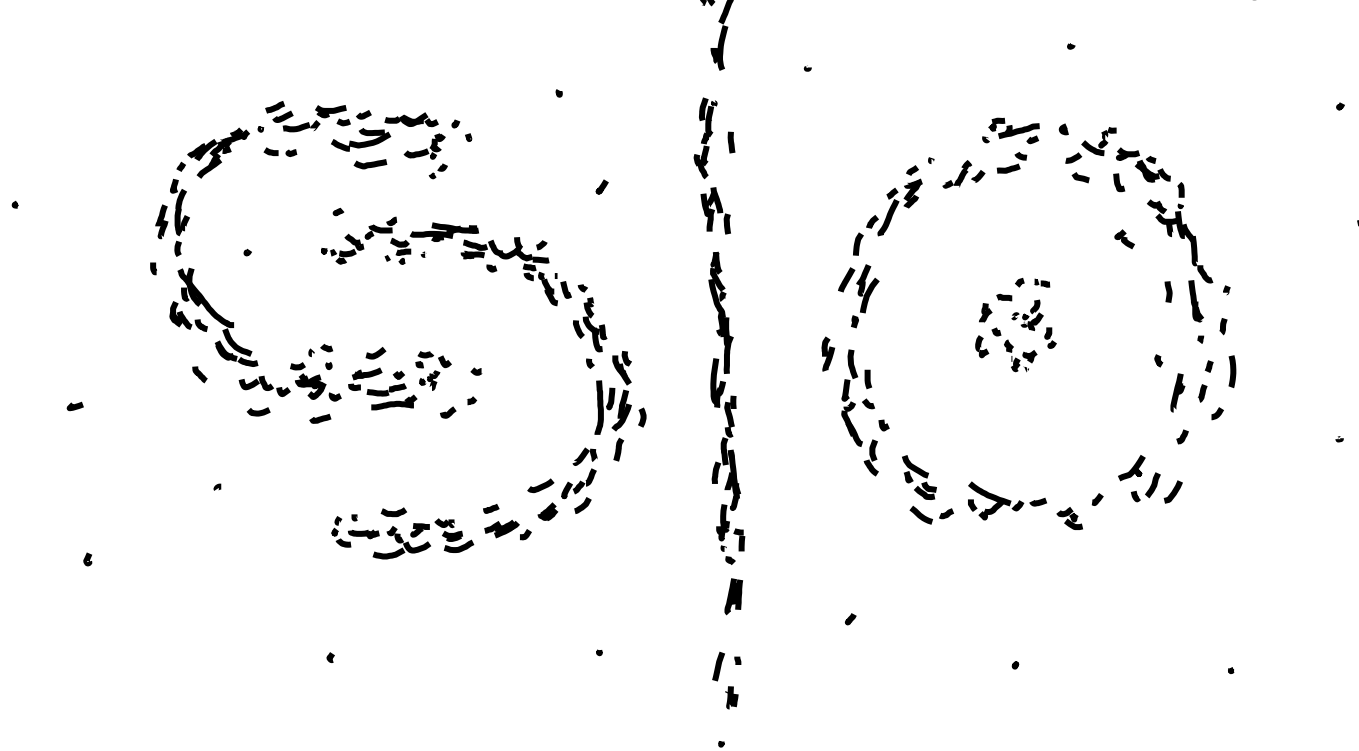
Hierarchical Clustering

# Admin

- Assignment 2 is out.
- I responded to your feedback (from last class) on Piazza
  - Past exams posted on github
  - Office hour scheduling is now “dynamic”
  - Feel free to ask for clarification on provided code
- I’m still looking at the online feedback system
  - There’s a link from the course website.

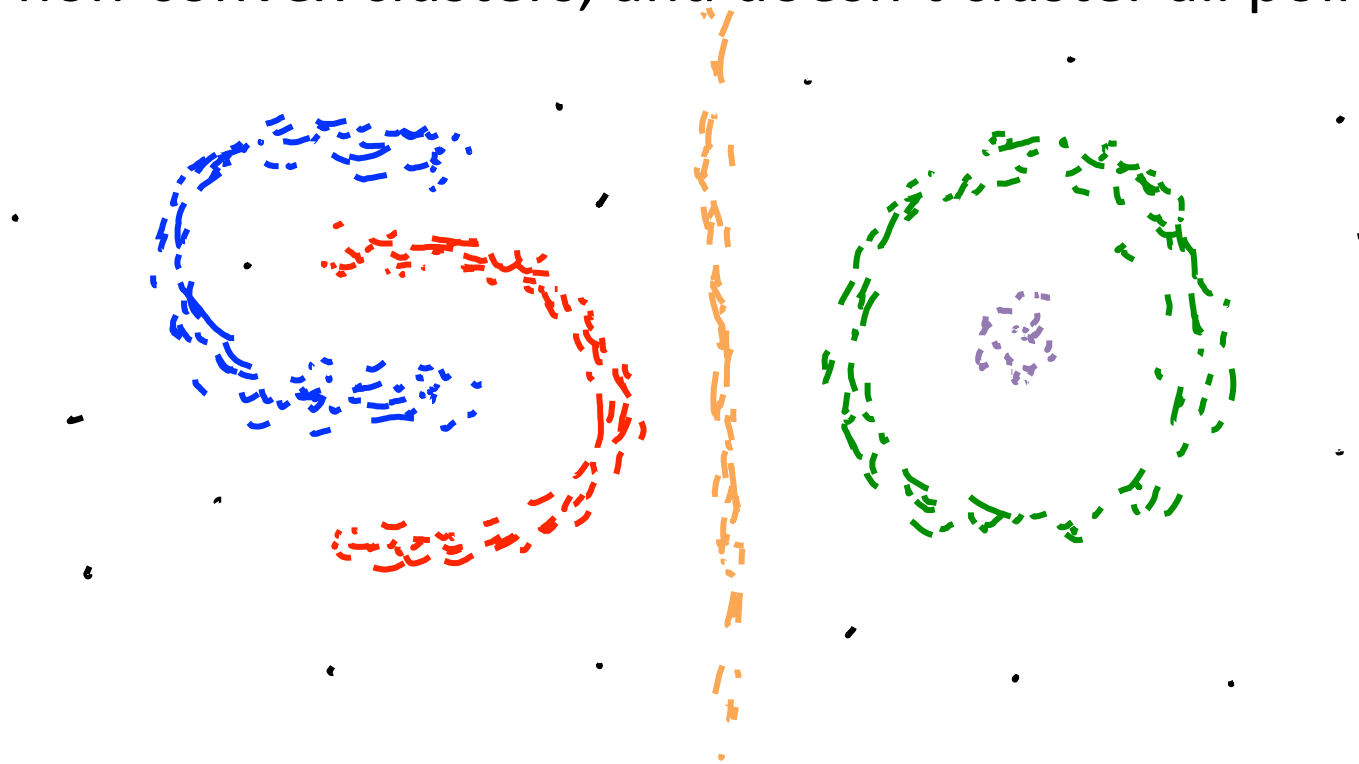
# Last Time: Density-Based Clustering

- We discussed **density-based clustering**:
  - **Non-parametric** clustering method.
  - Based on finding **connected regions of dense** points.
  - Can find non-convex clusters, and doesn't cluster all points.



# Last Time: Density-Based Clustering

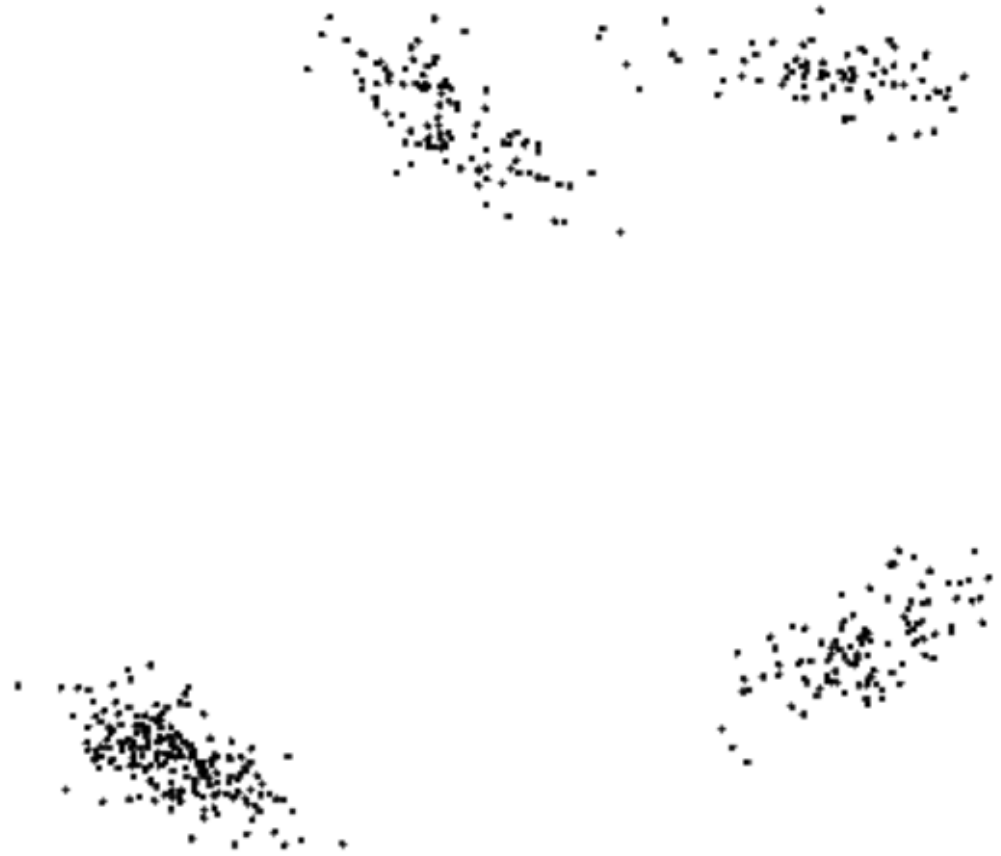
- We discussed **density-based clustering**:
  - **Non-parametric** clustering method.
  - Based on finding **connected regions of dense points**.
  - Can find non-convex clusters, and doesn't cluster all points.



# Cost of Density-Based Clustering

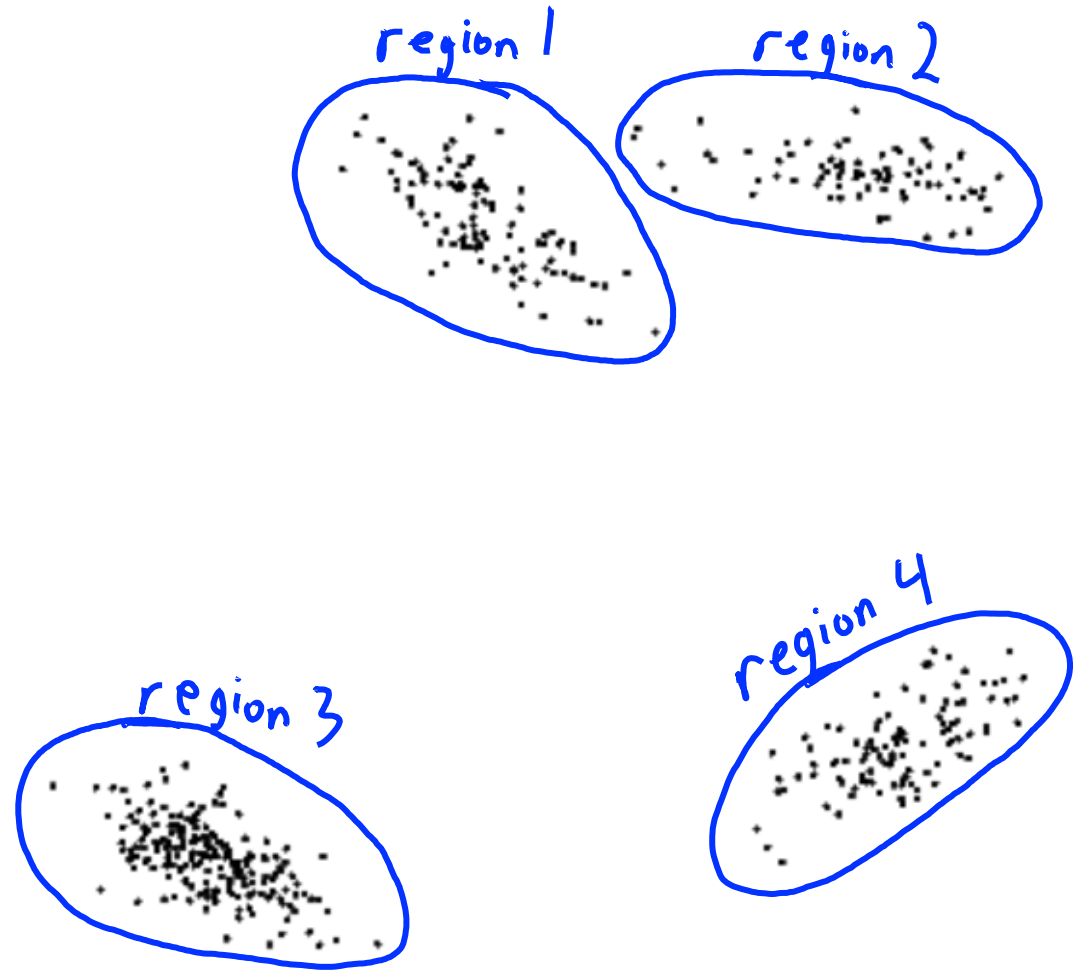
- Density-based clustering needs **all points within “radius” distance**.
  - What is the cost of this?
- Naïve approach to do this for all points:
  - Computing distance between 2 examples is  $O(d)$ .
  - We have ‘n’ training examples, so there are  $O(n^2)$  combinations.
  - Computing distance between all examples is  $O(n^2d)$ .
- **Not feasible if ‘n’ is large.**
- How can we **answer “closest points” queries when ‘n’ is large?**

# Distance-Based Pruning



# Distance-Based Pruning

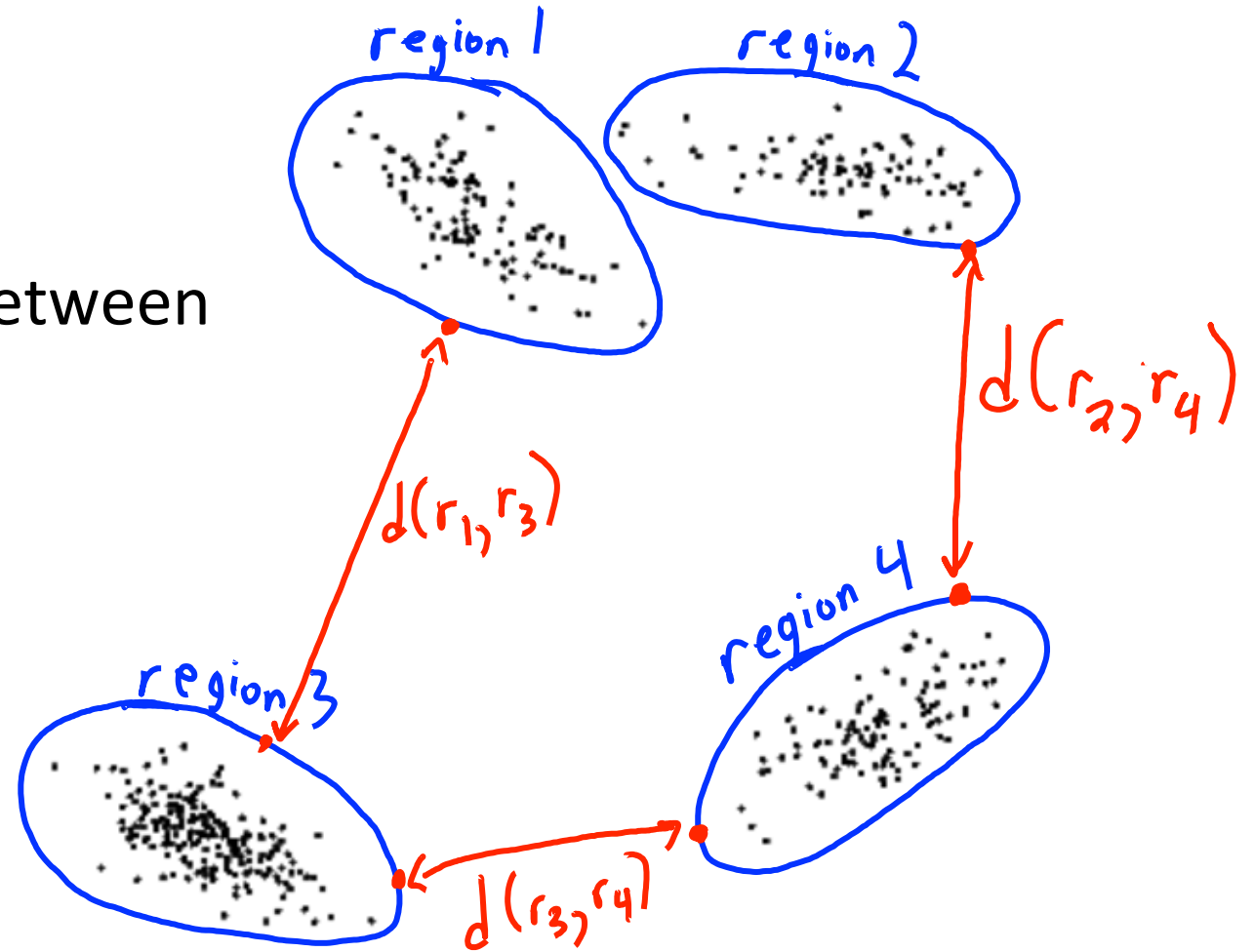
Divide objects into regions  $r_c$ .



# Distance-Based Pruning

Divide objects into regions  $r_c$ .

Let  $d(r_i, r_j)$  be minimum distance between regions  $r_i$  and  $r_j$ .



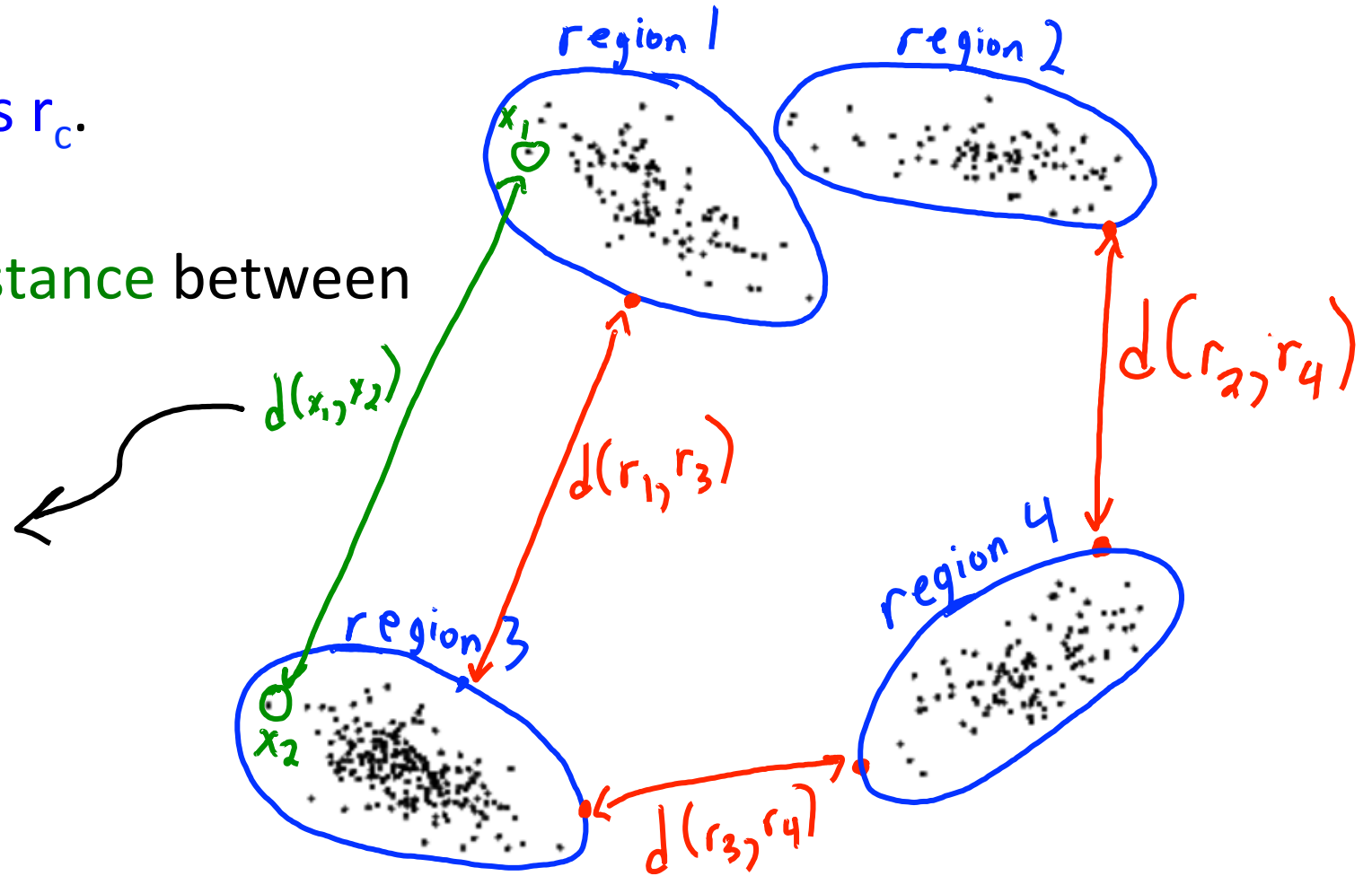


# Distance-Based Pruning

Divide objects into regions  $r_c$ .

Let  $d(r_i, r_j)$  be minimum distance between regions  $r_i$  and  $r_j$ .

Notice that  
 $d(x_1, x_2) \geq d(r_1, r_3)$

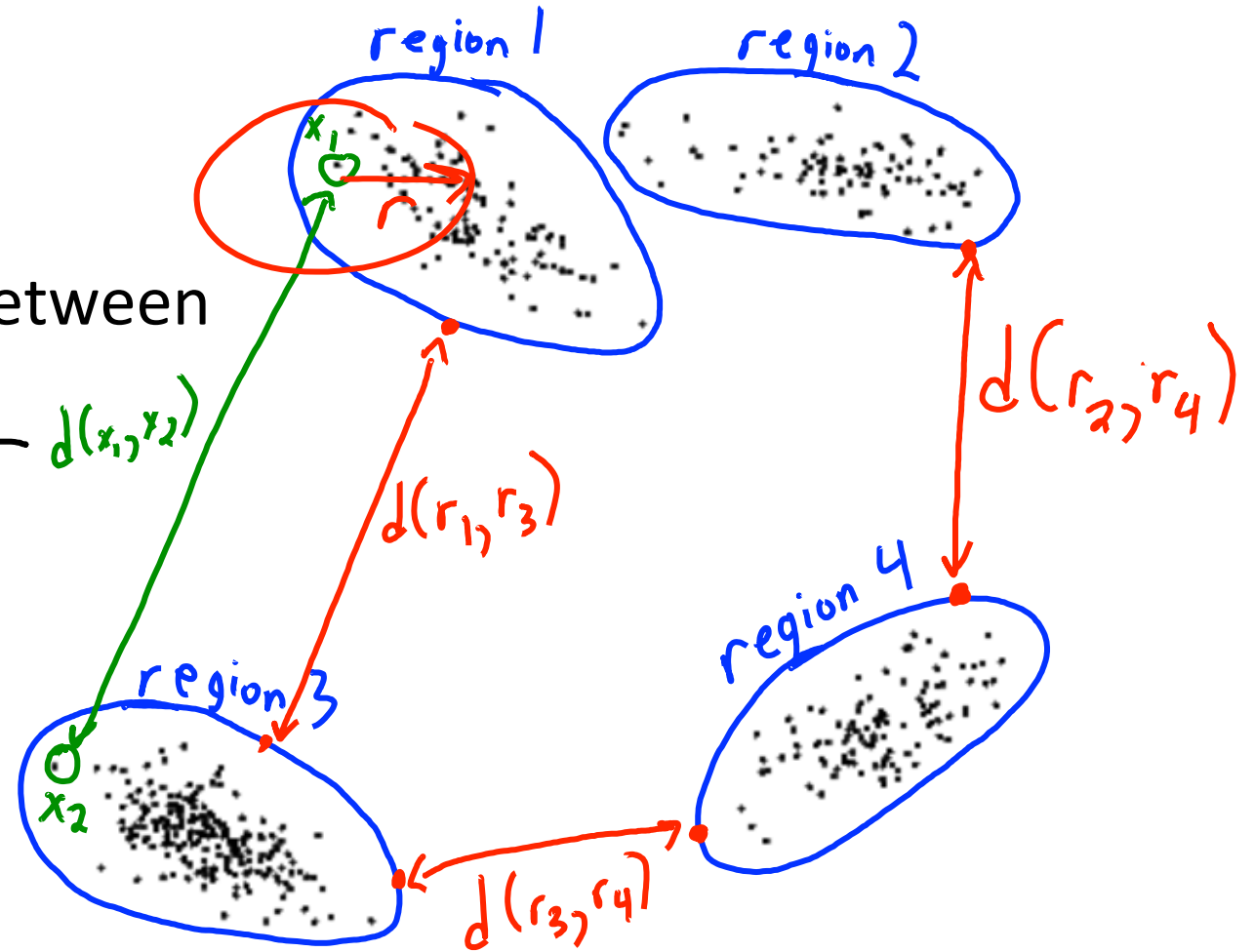


# Distance-Based Pruning

Divide objects into regions  $r_c$ .

Let  $d(r_i, r_j)$  be minimum distance between regions  $r_i$  and  $r_j$ .

Notice that  
 $d(x_1, x_2) \geq d(r_1, r_3)$



All points within radius “ $r$ ” are:

- Points within same region.
- Points in “close” regions where  $d(r_i, r_j) \leq r$ . (skip comparing points in far-apart regions)

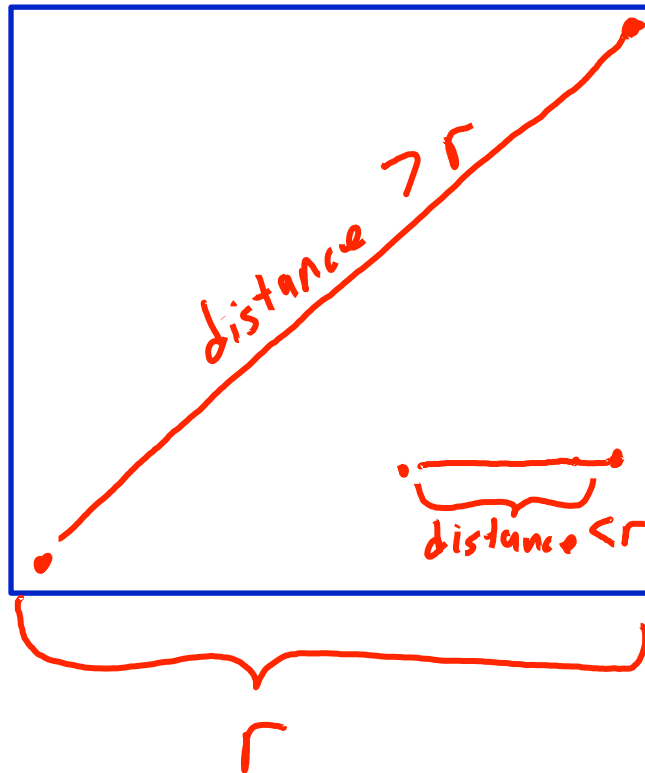
# Square-Shaped Regions

- A computationally-nice way to define “regions” for 2D data:
  - Squares with side length of ‘ $r$ ’.



# Square-Shaped Regions

- A computationally-nice way to define “regions” for 2D data:
  - Squares with side length of ‘ $r$ ’.

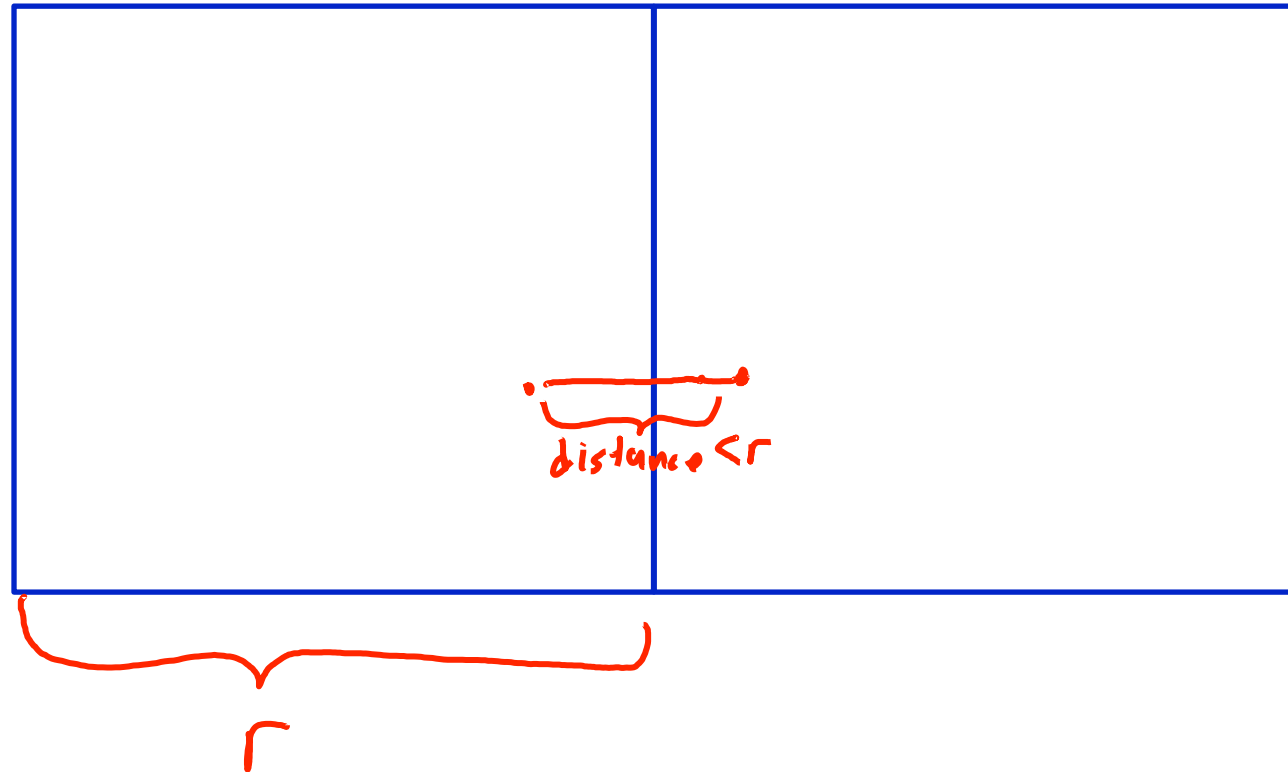


Points in **same square** can have distance less than ' $r$ '.

# Square-Shaped Regions

- A computationally-nice way to define “regions” for 2D data:
  - Squares with side length of ‘ $r$ ’.

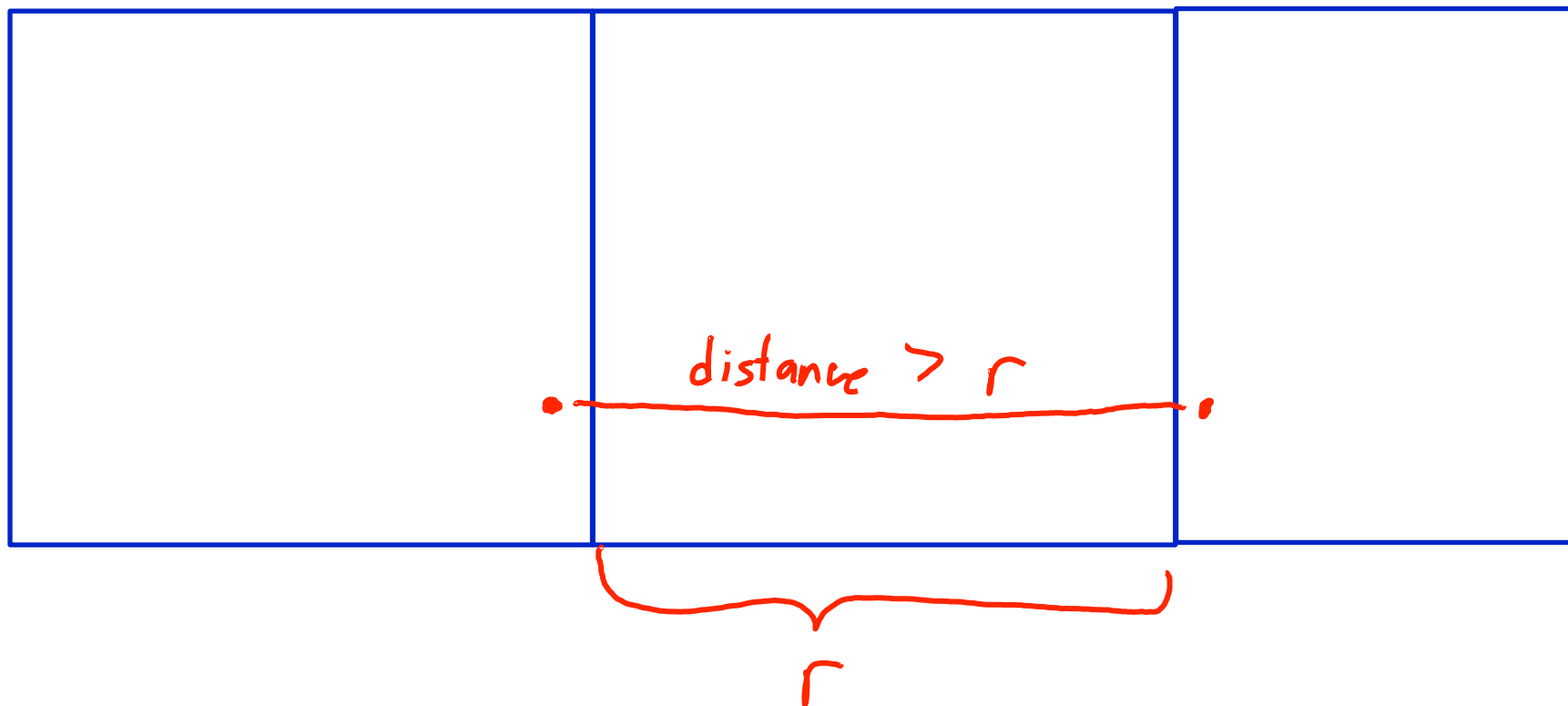
Points in **adjacent squares** can distance less than distance ‘ $r$ ’.



# Square-Shaped Regions

- A computationally-nice way to define “regions” for 2D data:
  - Squares with side length of ‘ $r$ ’.

Points in **non-adjacent squares** must have distance **more than ‘ $r$ ’**.

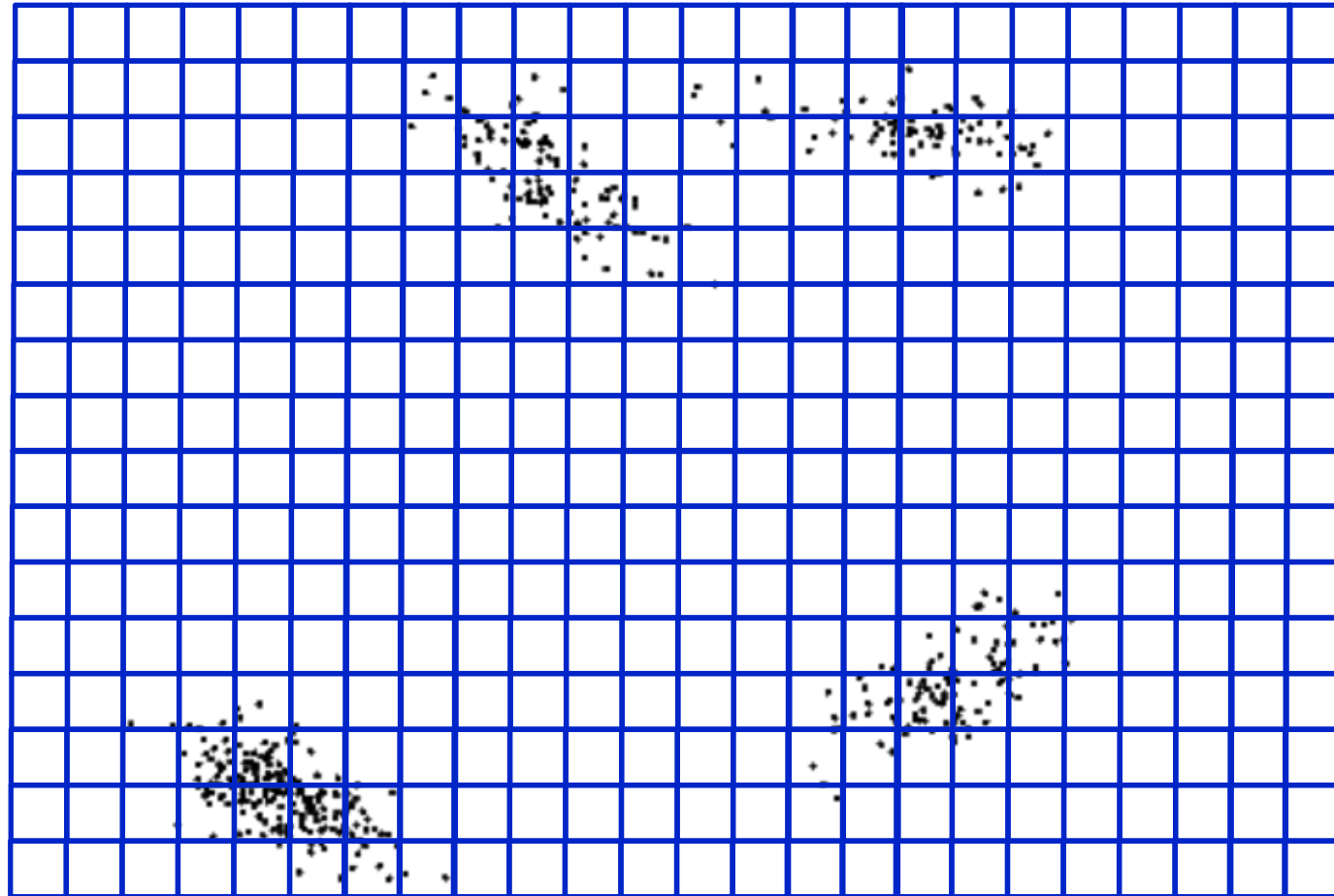


# Grid-Based Pruning

- Assume we want to find objects within a distance of 'r'.

Divide space  
into **squares**  
of length  $r$ .

Assign (“hash”)  
each object to its  
square region.



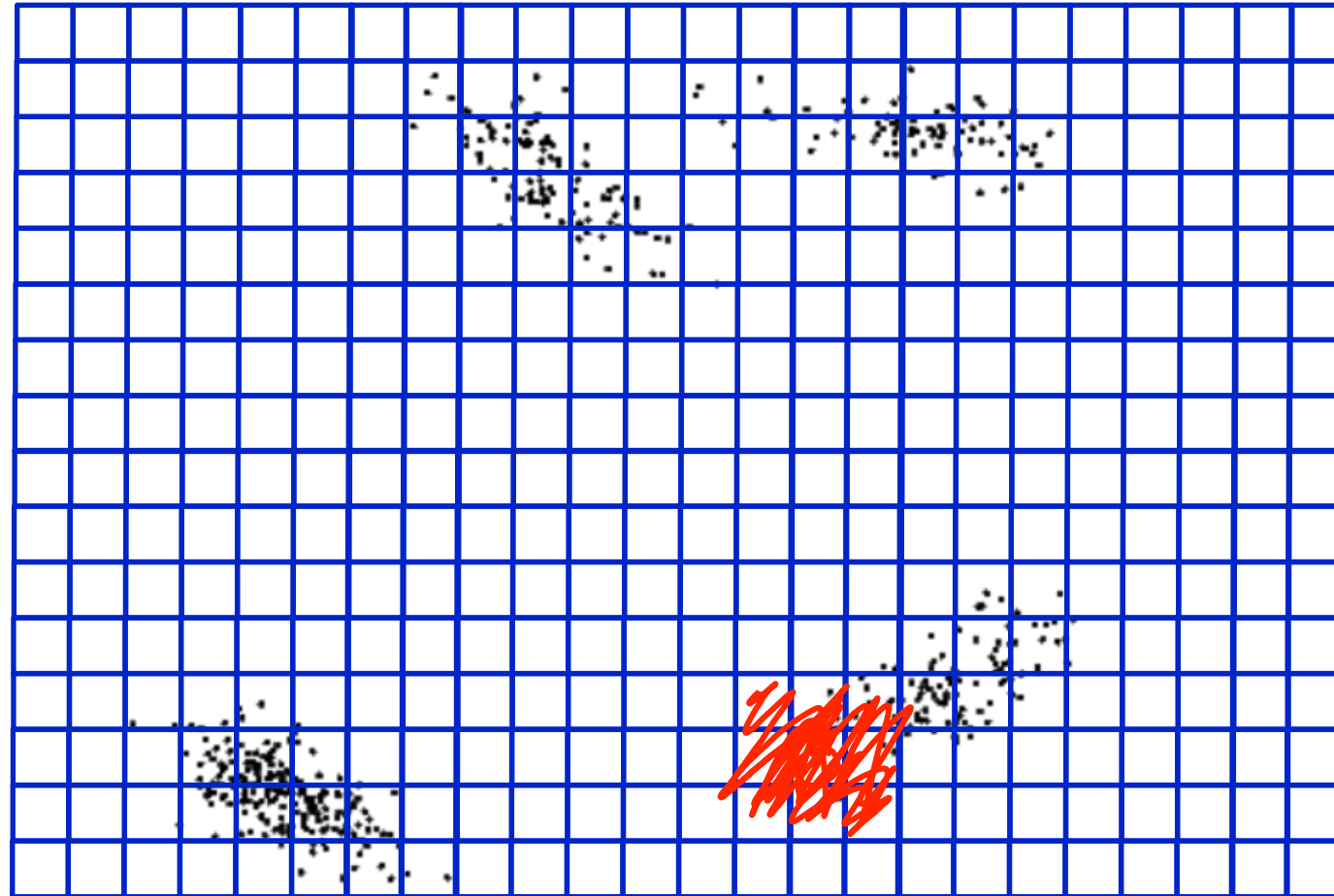
# Grid-Based Pruning

- Assume we want to find objects within a distance of 'r'.

Divide space  
into **squares**  
of length  $r$ .

Assign (“hash”)  
each object to its  
square region.

- Only check objects in **same**  
**and adjacent** regions.





# Grid-Based Pruning Discussion

- Grid-based pruning also works for  $L_1$ -norm and  $L_\infty$ -norm.
- Similar ideas can be used for other “closest point” calculations:
  - K-nearest neighbours (finding closest training examples).
- But we again have the “curse of dimensionality”:
  - Hyper-cubes of length ‘r’ are tiny: **few points will fall into each cube.**
  - **Number of adjacent cubes increases exponentially:**
    - 8 in 2D, 26 in 3D, 80 in 4D, 242 in 5D,  $3^d - 1$  in d dimensions.
- Better “regions” in high-dimensions (but still bad in worst case):
  - Quad-trees, R-trees, ball-trees.
- Approximate methods:
  - Locality sensitive hashing (random transformation to low-dimensional problem).

# Ensemble Clustering

? question ☆

stop following 23 views

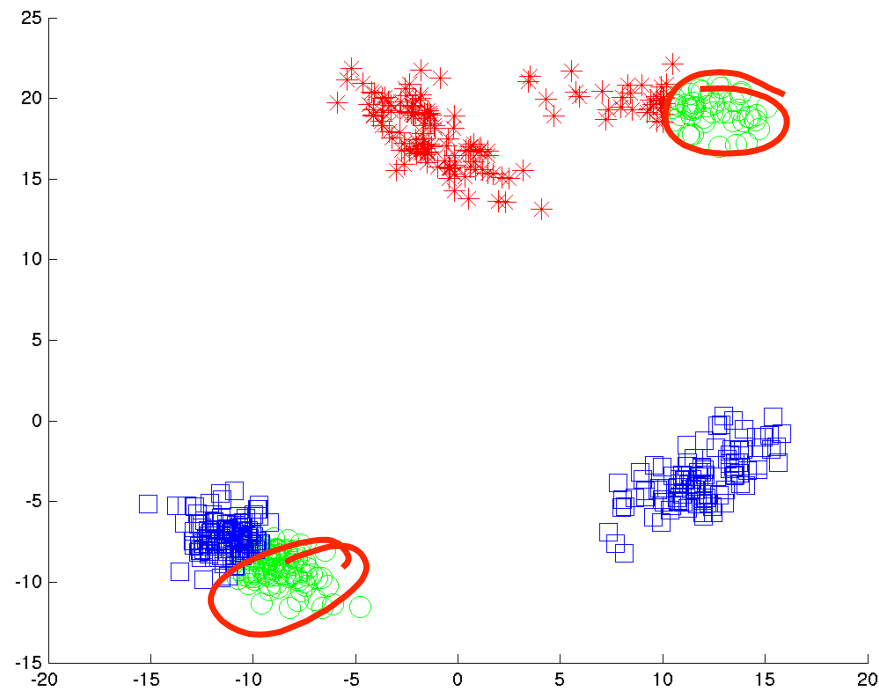
## Multiple random runs of K means

I was wondering how running K Means (original version, not K means ++ ) several times with random initializations can help us make an accurate model. K Means outputs the class labels of all the samples. We definitely can't use mode of all the labels it got in different runs because class labels from different runs don't make any sense when compared. We somehow have to see what points are coming in the same cluster in a lot of runs..I am not sure, how do we do it?

- We can consider **ensemble methods** for clustering.
  - “Consensus clustering”
- It's a good/important idea:
  - **Bootstrapping** is widely-used.
  - “Do clusters change if the data was slightly different?”
- But we **need to be careful** about how we combine models.

# Ensemble Clustering

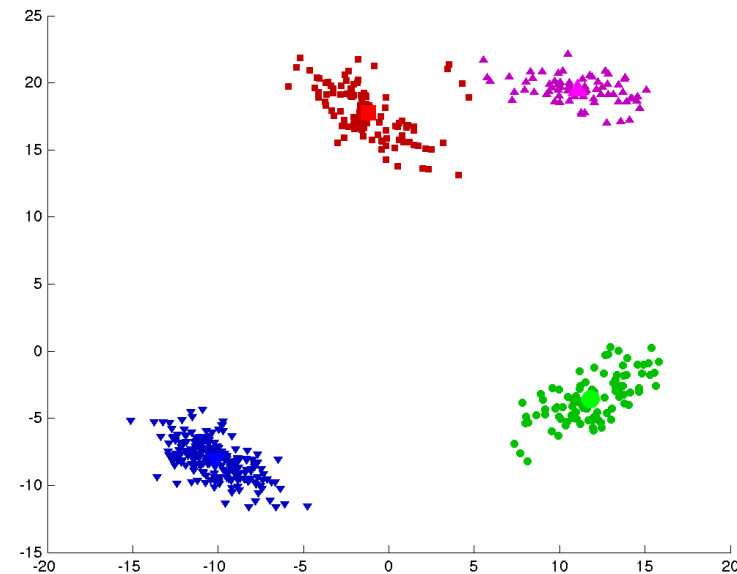
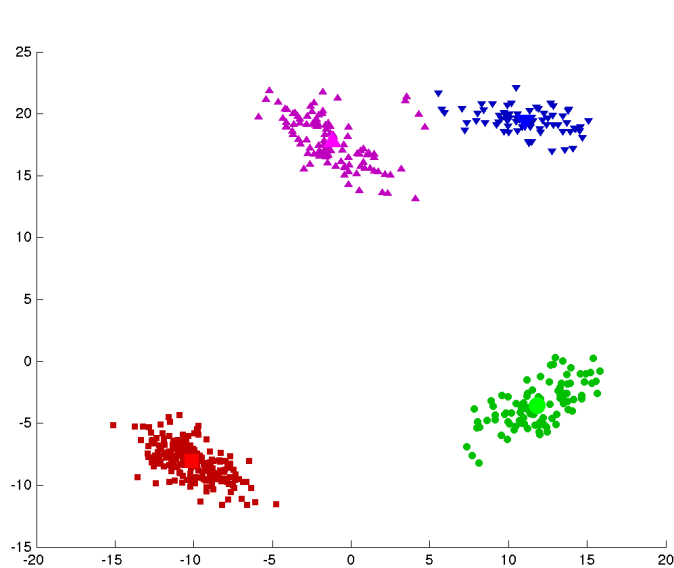
- E.g., run k-means 20 times and then cluster using the mode.
- Normally, averaging across models doing different things is good.



- But this is a bad ensemble method: **worse than k-means on its own.**

# Label Switching Problem

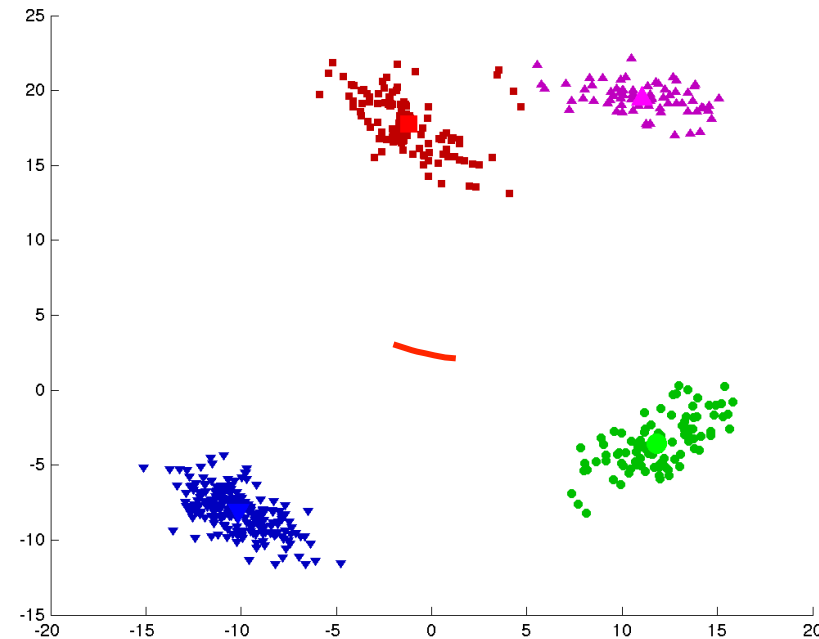
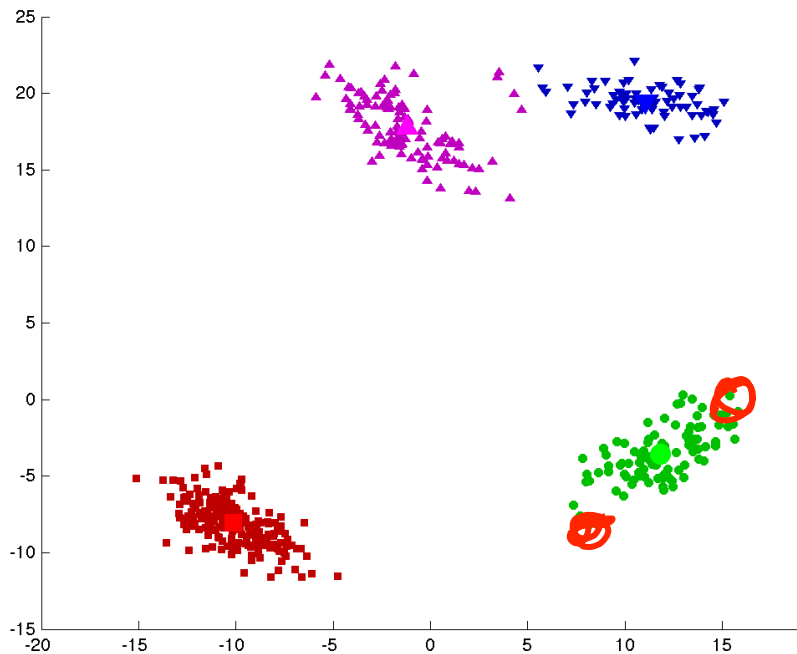
- This doesn't work because of **label switching**:
  - The **cluster labels are meaningless**.
  - We could get **same clustering with permuted labels**:



- All **clusters become equally likely** as number of initializations increases.

# Addressing Label Switching Problem

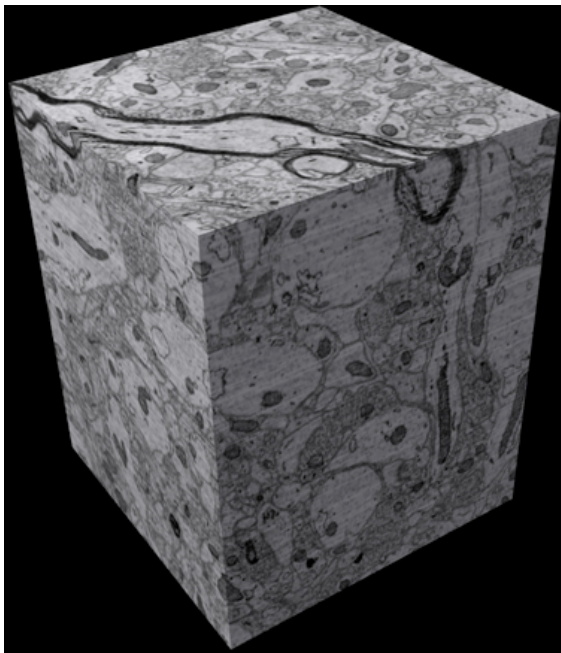
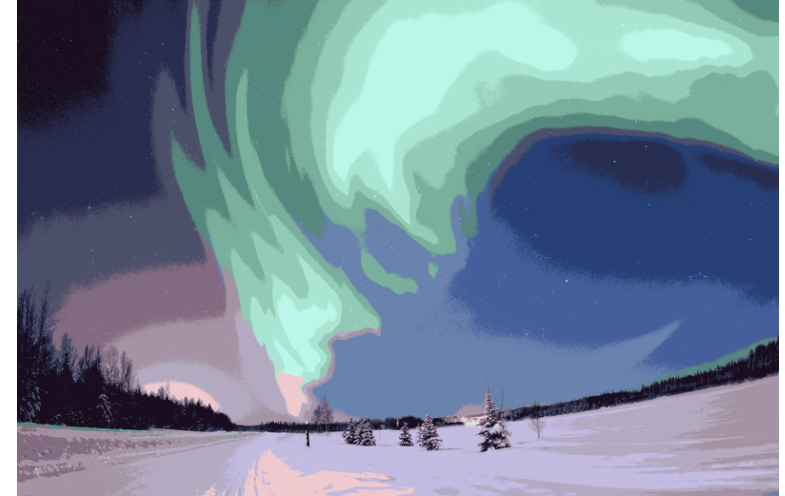
- Ensembles can't depend on label "meaning":
  - Don't ask "is point  $x_i$  in red square cluster?", which is meaningless.
  - Ask "is point  $x_i$  in the same cluster as  $x_j$ ?", which is meaningful.



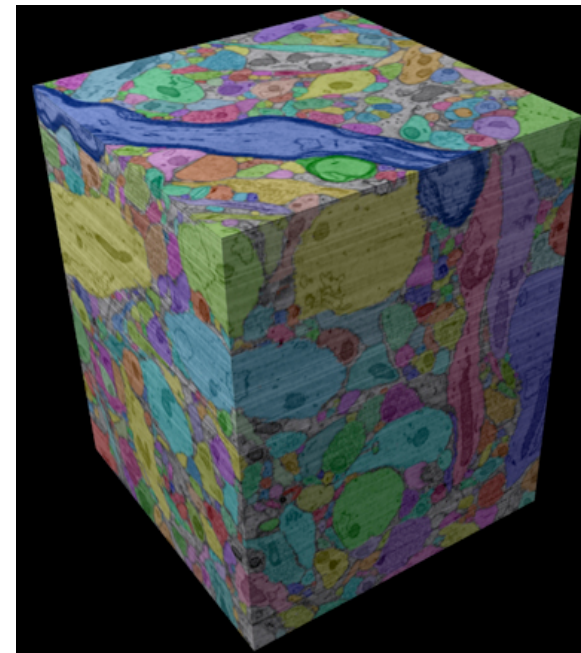
# Another application: image segmentation



k-means



supervised learning



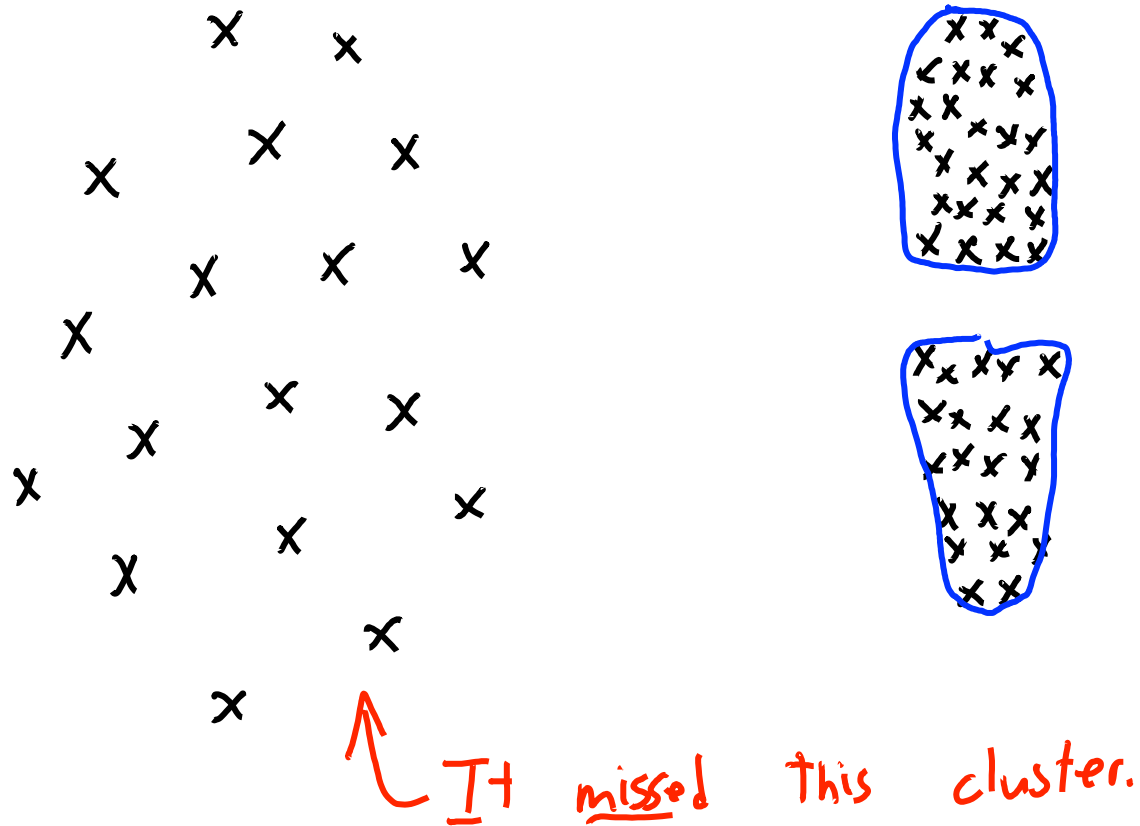
Images from Wikipedia and  
conectomethebook.com

# Supervised learning for clustering

- We are given a bunch of example *data sets* (images) and their true clustering (segmented images)
- Want to learn clustering (of pixels) based on data
- Each pixel is an object, we compute some features of the pixels
- We need an error metric to assess the quality of our predictions
- Again, we have the label switching problem
- One possible solution is the Rand index, which uses pairwise info
  - For a pair of objects, are they in the same cluster in clustering A? In clustering B? If they agree we call this “agreement”.
  - Rand index: fraction of pairs for which clustering A and B agree

# Differing Densities

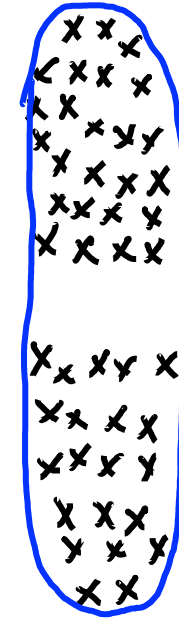
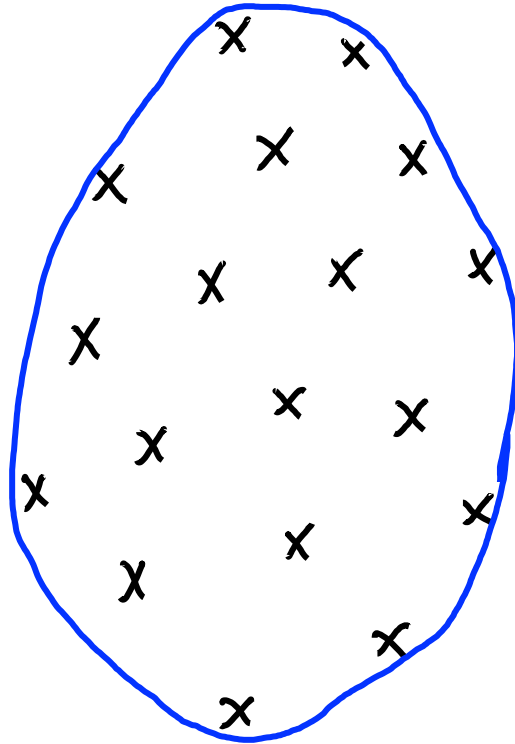
- Consider density-based clustering on this data:





# Differing Densities

- Increase the radius and run it again:

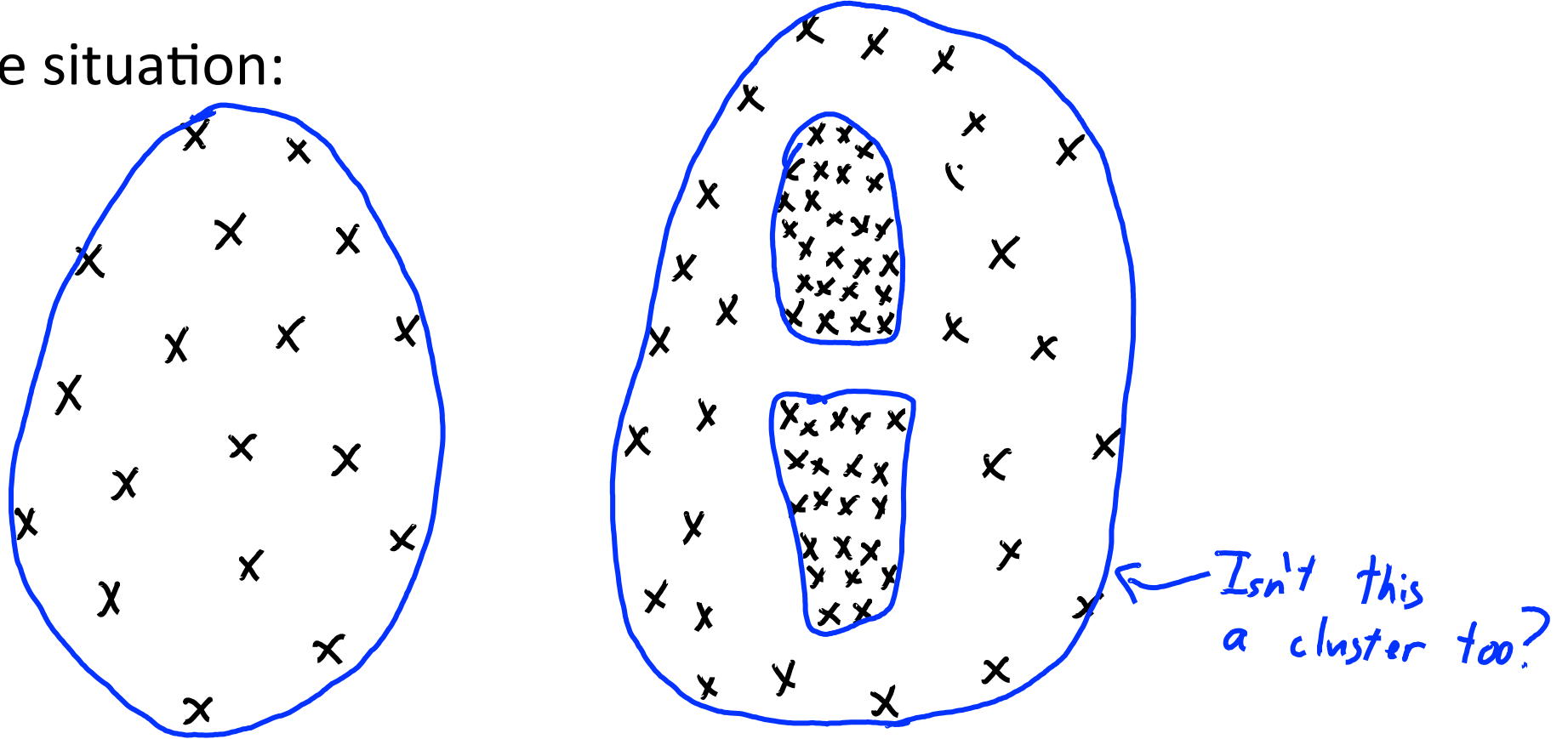


These 2 clusters  
are now "close."

- There may no density-level that gives you 3 clusters.

# Differing Densities

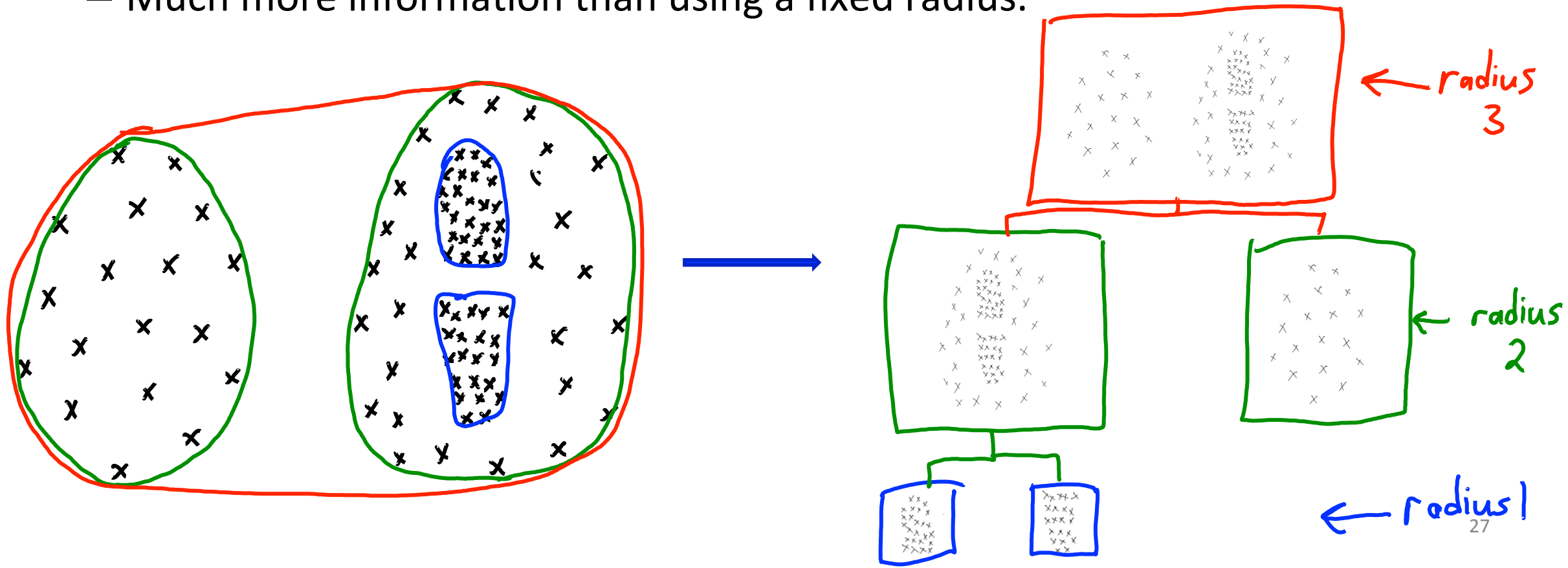
- Here is a worse situation:



- Now you need to choose between coarse/fine clusters.
- Instead of fixed clustering, we often want **hierarchical clustering**.

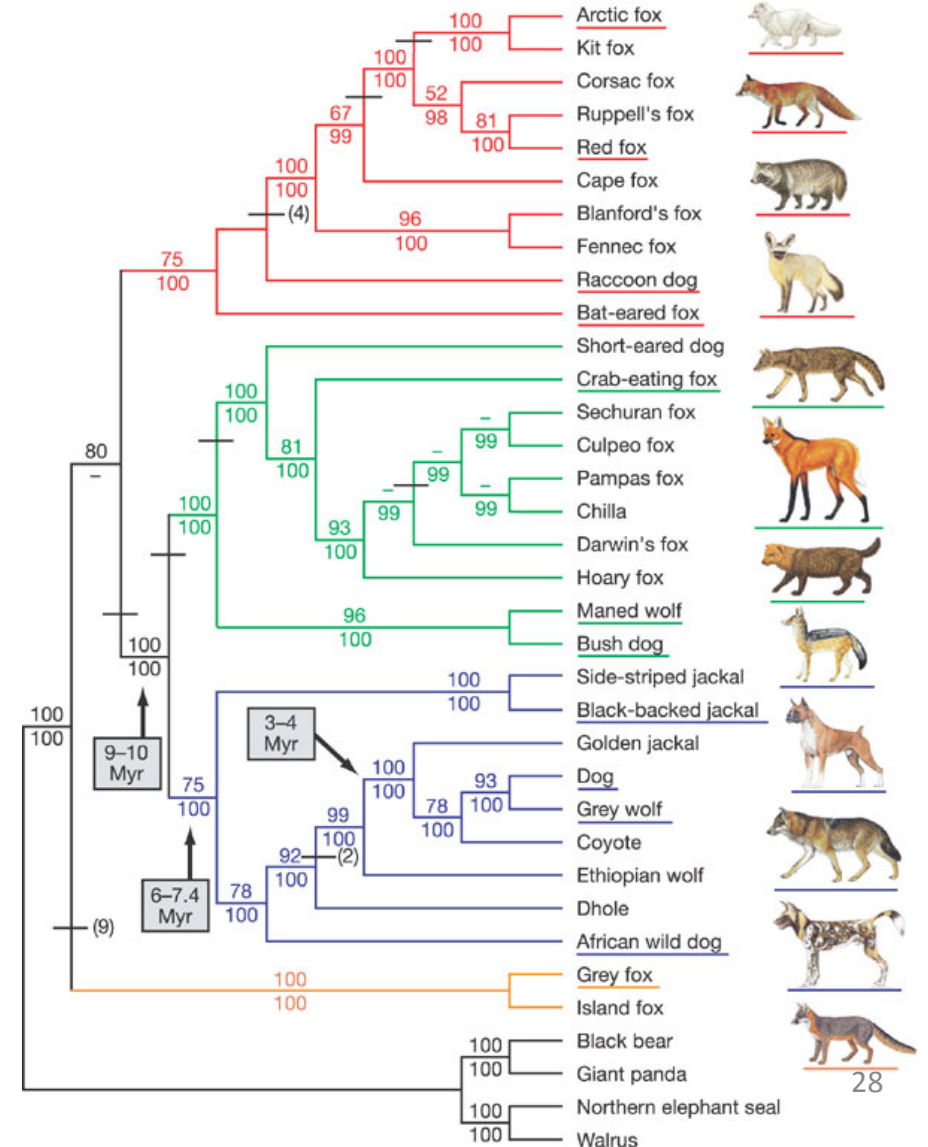
# Density-Based Hierarchical Clustering

- A simple way to make a **hierarchical DBSCAN**:
  - Fix minPoints, **record clusters as you vary the radius.**
  - Much more information than using a fixed radius.



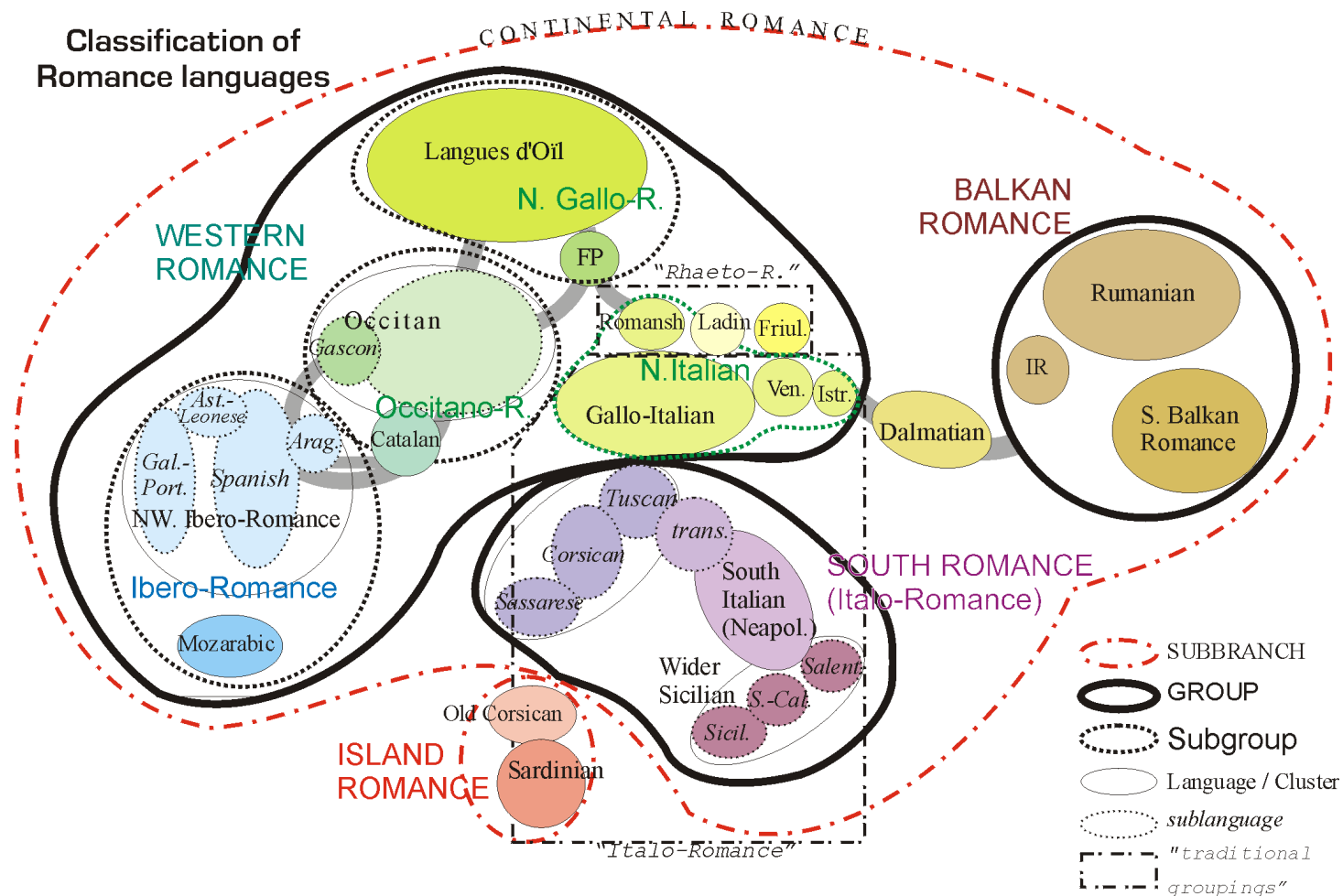
# Application: Phylogenetics

- We sequence genomes of a set of organisms.
- Can we construct the “tree of life”?
- Comments on this application:
  - On the right are individuals.
  - As you go left, clusters merge.
  - Merges are ‘common ancestors’.



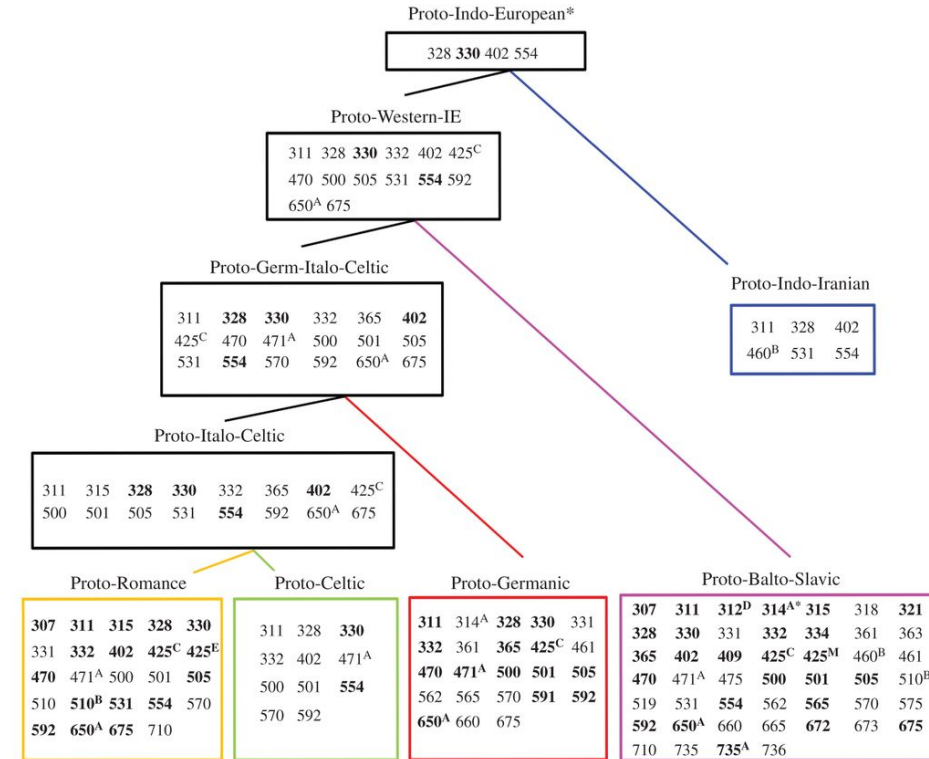
# Application: Phylogenetics

- Comparative method in linguistics studies evolution of languages:



# Application: Phylogenetics

- Recent paper on evolution of fairy tales.
  - Evidence that “Devil and the Smith” goes back to bronze age.
  - “Beauty and the Beast” published in 1740, but might be 2500-6000 years old.

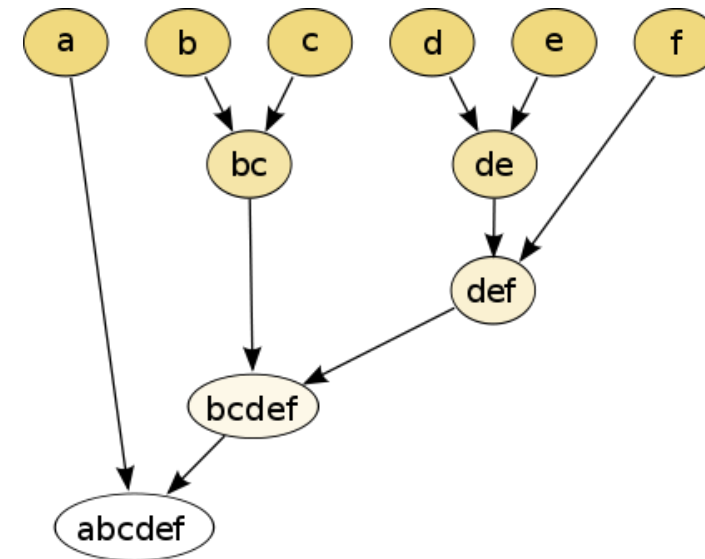
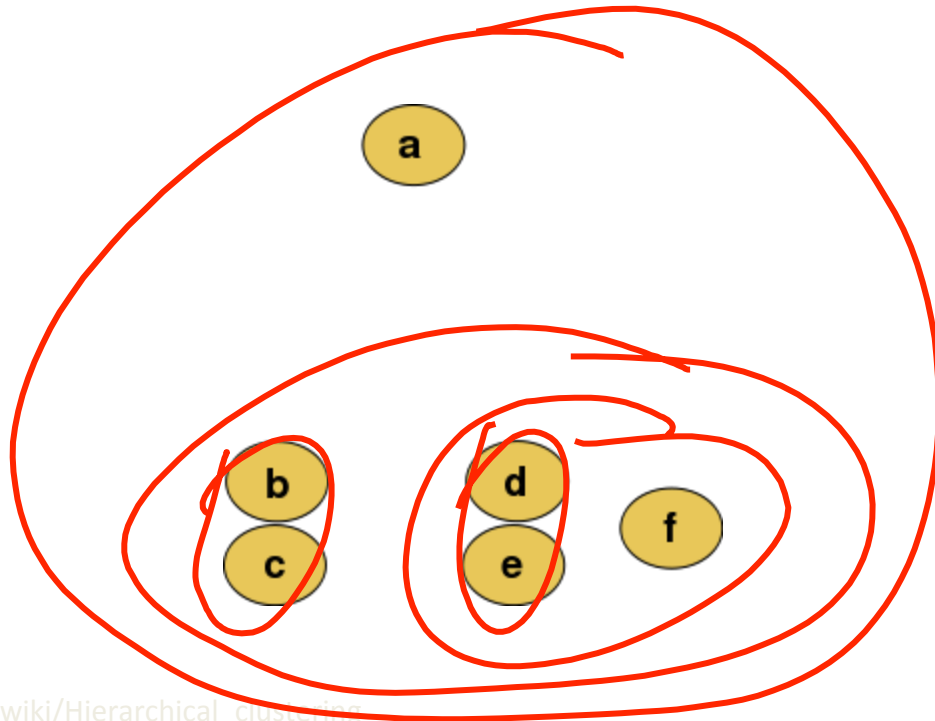


International tale types

307	The Princess in the Coffin	409	The Girl as Wolf	562	The Spirit in the Blue Light
311	Rescue by Sister	425C	Beauty and the Beast	565	The Magic Mill
312D	Rescue by the Brother	425E	The Enchanted Husband	570	The Rabbit-Herd
314A	The Shepherd and the Giants	425M	The Snake Bridegroom	575	The Prince's Wings
314A*	Animal Helper in the Flight	460B	The Journey	591	The Thieving Pot
315	The Faithless Sister	461	Three Hairs	592	The Dance Among Thorns
318	The Faithless Wife	470	Friends in Life and Death	650A	Strong John
321	Eyes Recovered from Witch	471A	The Monk and the Bird	660	The Three Doctors
328	The Boy Steals Ogre's Treasure	475	The Man as the Heater	665	The Man who Flew and Swam
330	The Smith and the Devil	500	Supernatural Helper	672	The Serpent's Crown
331	The Spirit in the Bottle	501	The Three Old Spinning Women	673	The White Serpent's Flesh
332	Godfather Death	505	The Grateful Dead	675	The Lazy Boy
334	Household of the Witch	510	Cinderella and Peau d'Âne	710	Our Lady's Child
361	Bear Skin	510B	Peau d'Âsne	735	The Rich and the Poor Man
363	The Corpse-Eater	519	The Strong Woman as Bride	735A	Bad Luck Imprisoned
365	The Dead Bridegroom	531	The Clever Horse	736	Luck and Wealth
402	The Animal Bride	554	The Grateful Animals		

# Agglomerative (Bottom-Up) Clustering

- More common hierarchical method: **agglomerative clustering**.
  1. Starts with each point in its own cluster.
  2. Each step merges the two “closest” clusters.
  3. Stop when everything is in one big cluster.



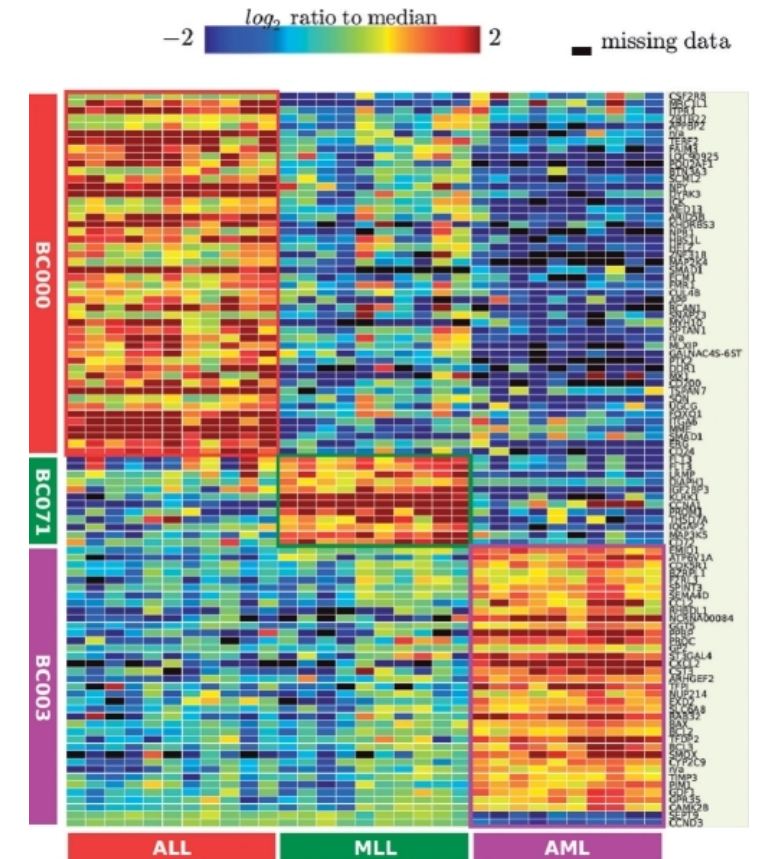
# Agglomerative (Bottom-Up) Clustering

- More common hierarchical method: [agglomerative clustering](#).
  1. Starts with each point in its own cluster.
  2. Each step merges the two “closest” clusters.
  3. Stop when everything is in one big cluster.
- How do you define “closest” clusters?
  - Average-link: average distance between points in clusters.
  - Single-link: minimum distance between points in clusters.
  - Complete-link: maximum distance between points in clusters.
  - Ward’s method: minimize within-cluster variance.
- Reinvented by different fields under different names (UPGMA).
  - Popular implementation: BIRCH.



# Other Clustering Methods

- Mixture models:
  - Probabilistic clustering.
- Mean-shift clustering:
  - Finds local “modes” in density of points.
- Bayesian clustering:
  - A variant on ensemble methods.
  - Averages over models/clustering, weighted by “prior” belief in the model/clustering.
- Biclustering:
  - Simultaneously cluster objects and features.
- Spectral clustering and graph-based clustering:
  - Clustering of data described by graphs.



# Summary

- **Region-based pruning**: speeds up “closest point” calculations.
- **Ensemble clustering**: combines multiple clusterings.
  - Can work well but need to account for **label switching**.
- **Hierarchical clustering**: more informative than fixed clustering.
- **Agglomerative clustering**: sequentially merges clusters.
- Next time: discovering a hole in the ozone layer.