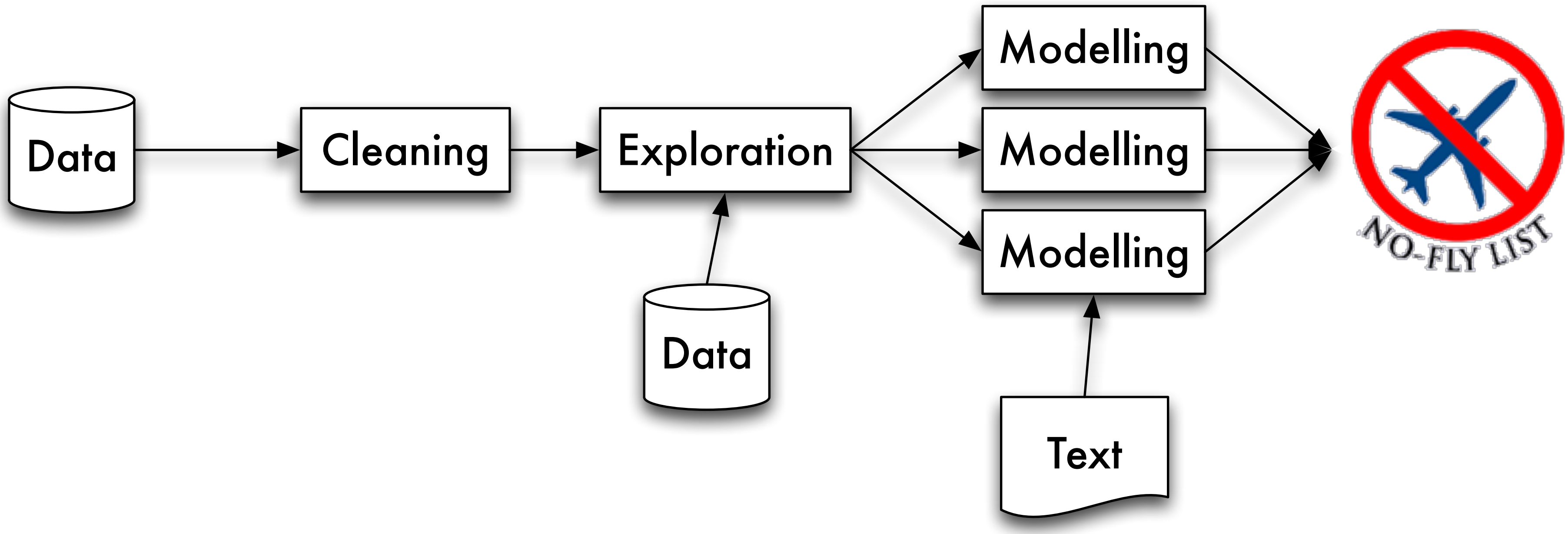
Power to the points: Local certificates for clustering

SIMONS INSTITUTE
for the Theory of Computing

Suresh Venkatasubramanian
University of Utah

Joint work with Parasaran Raman

Data Mining Pipeline

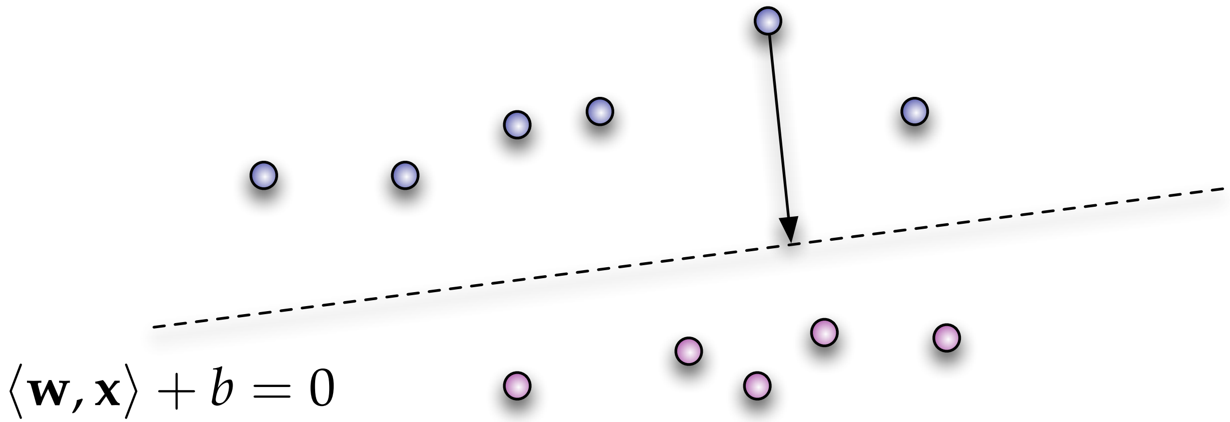


Learning algorithms ensure (global) quality of inference process

But what about the (local) labels assigned to data ?

Can we find LOCAL and SUCCINCT certificates that validate correctness of data labels ?

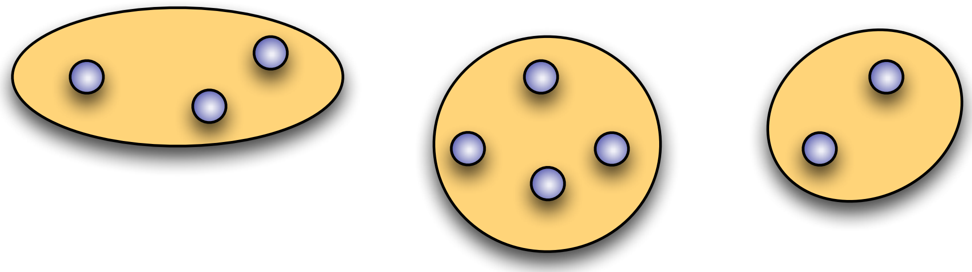
Local Validation in Classification



Platt scaling (P99): $p(y = 1 \mid \mathbf{x}, \mathbf{w}, b) = \dfrac{1}{\exp(A(\langle \mathbf{w}, \mathbf{x} \rangle + b) + B)}$

Parameters are estimated using ML

Clustering Data



<u>Group objects into meaningful *clusters*</u>

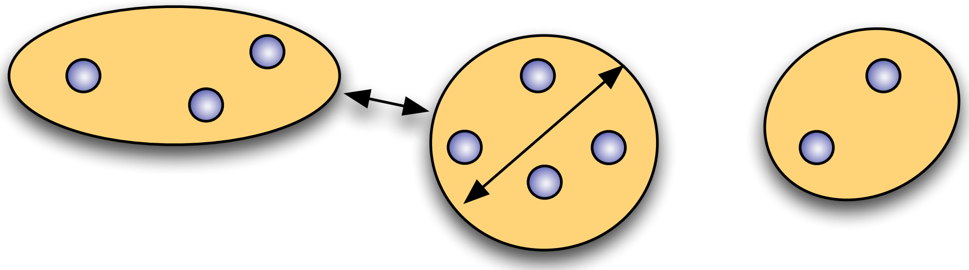
Different methods produce different answers

- k-means/medoids, HAC, spectral clustering, subspace clustering, correlation clustering, information bottleneck, ...

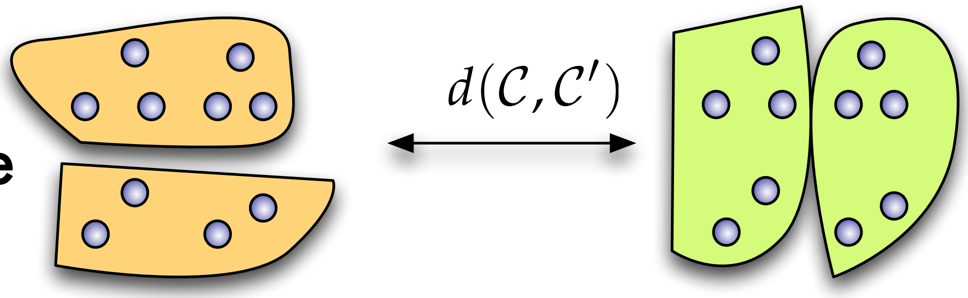
How do we know if an answer is good ?

Validating Clusterings

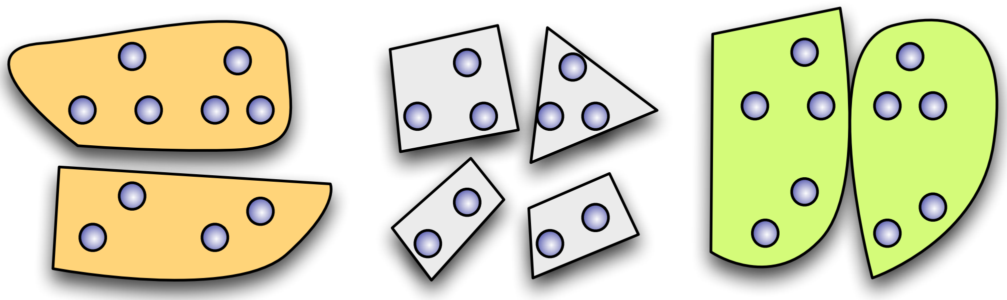
Internal validation:
intra- vs inter-cluster distance



External validation:
compare to a reference clustering



Relative validation/ stability:
compare different runs of algorithm



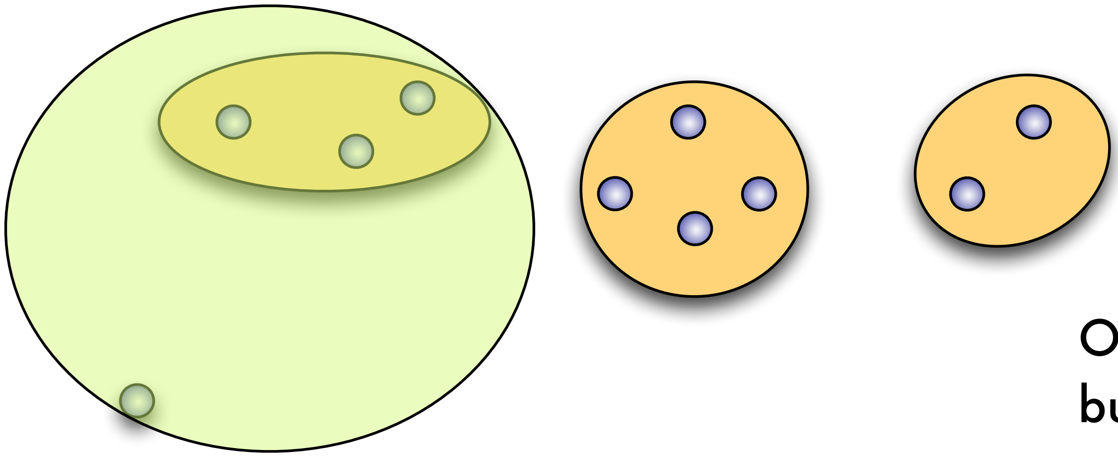
Power to the points

Given a clustering of data, determine confidence scores for the label assigned to a point.

Desiderata:

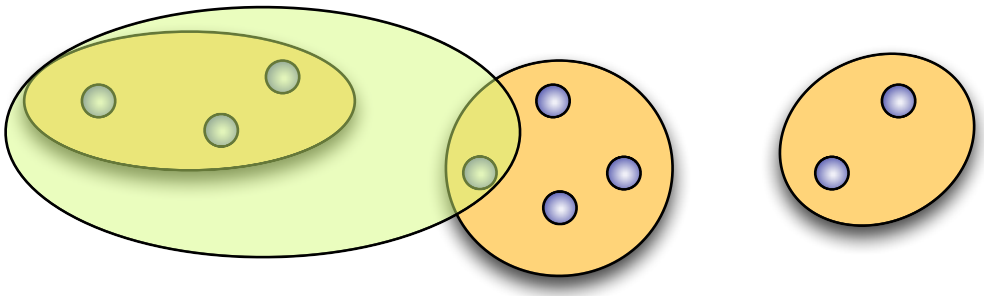
1. Data-independent scale.
2. Agnostic to the method by which the clustering was made.
3. Works for a single clustering...
4. but can be used to compare different clusterings.

Outlier Detection vs Local Validation



$$\min_{S \subset P, |S| \geq (1-\epsilon)|P|} \min_{\mathcal{C}(S)} f(\mathcal{C})$$

Outlier changes cost function
but not the structure of the answer



Locally unstable points change the
structure of the answer, but not the cost

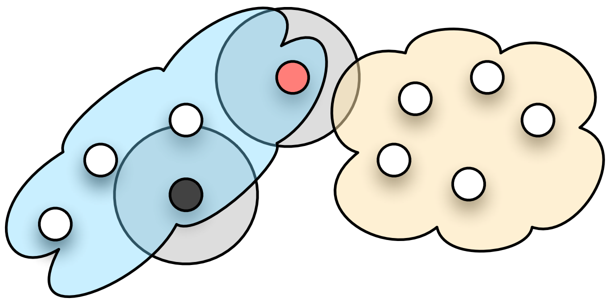
Power to the points

Given a clustering of data, determine confidence scores for the label assigned to a point.

Desiderata:

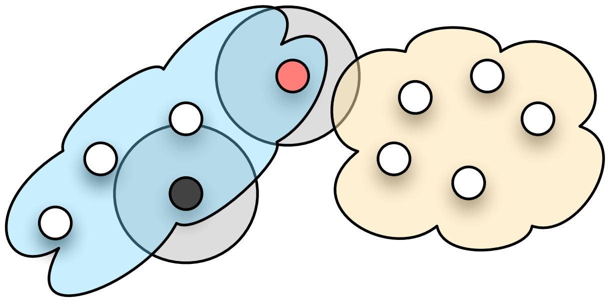
1. Data-independent scale.
2. Agnostic to the method by which the clustering was made.
3. Works for a single clustering...
4. but can be used to compare different clusterings.

Regions of influence



A point should be in a cluster if its region of influence overlaps the cluster region of influence

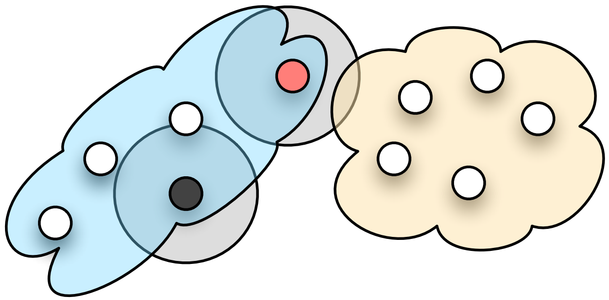
Regions of influence



A point should be in a cluster if its region of influence overlaps the cluster region of influence

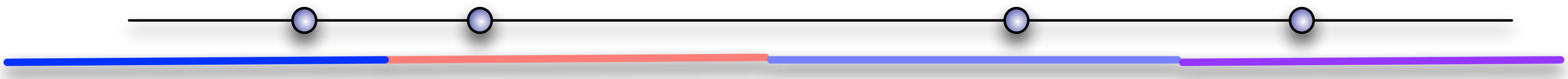
To estimate a point's affinity for a cluster, add it as a singleton "cluster" and see how much area it "steals" from neighboring clusters

Regions of influence

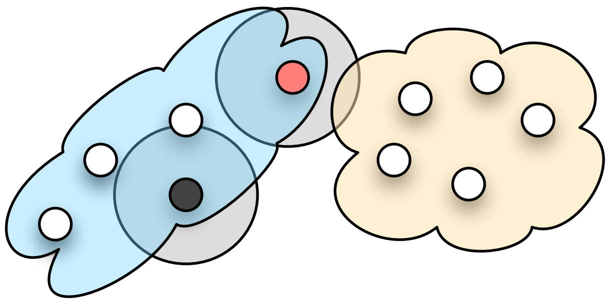


A point should be in a cluster if its region of influence overlaps the cluster region of influence

To estimate a point's affinity for a cluster, add it as a singleton "cluster" and see how much area it "steals" from neighboring clusters

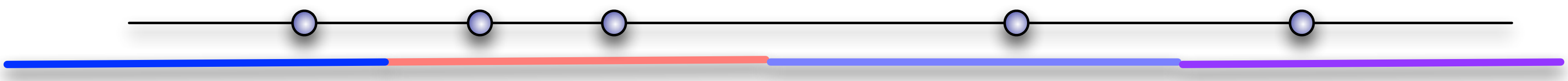


Regions of influence

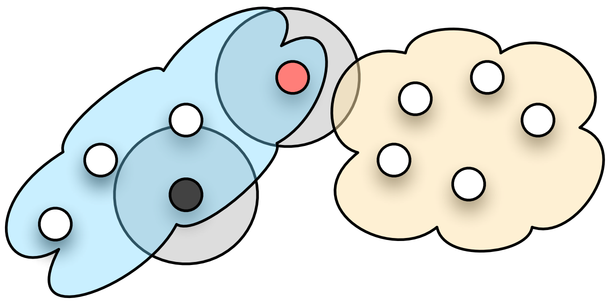


A point should be in a cluster if its region of influence overlaps the cluster region of influence

To estimate a point's affinity for a cluster, add it as a singleton "cluster" and see how much area it "steals" from neighboring clusters

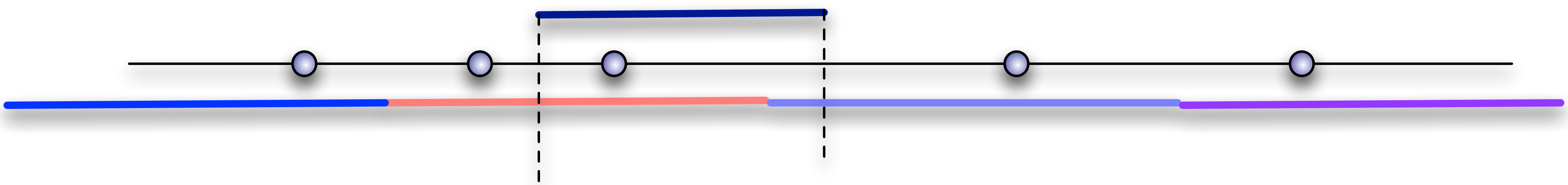


Regions of influence



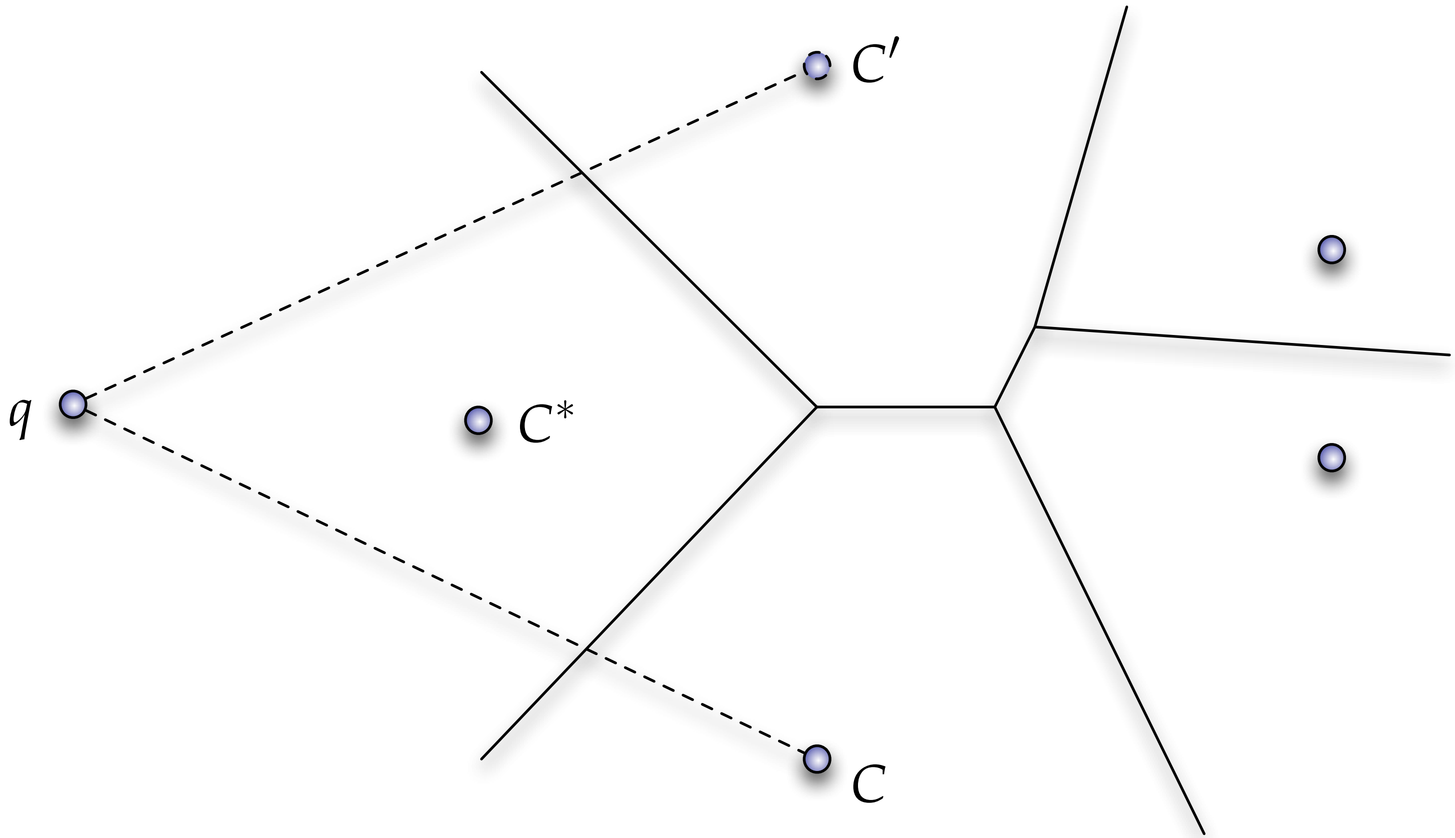
A point should be in a cluster if its region of influence overlaps the cluster region of influence

To estimate a point's affinity for a cluster, add it as a singleton "cluster" and see how much area it "steals" from neighboring clusters



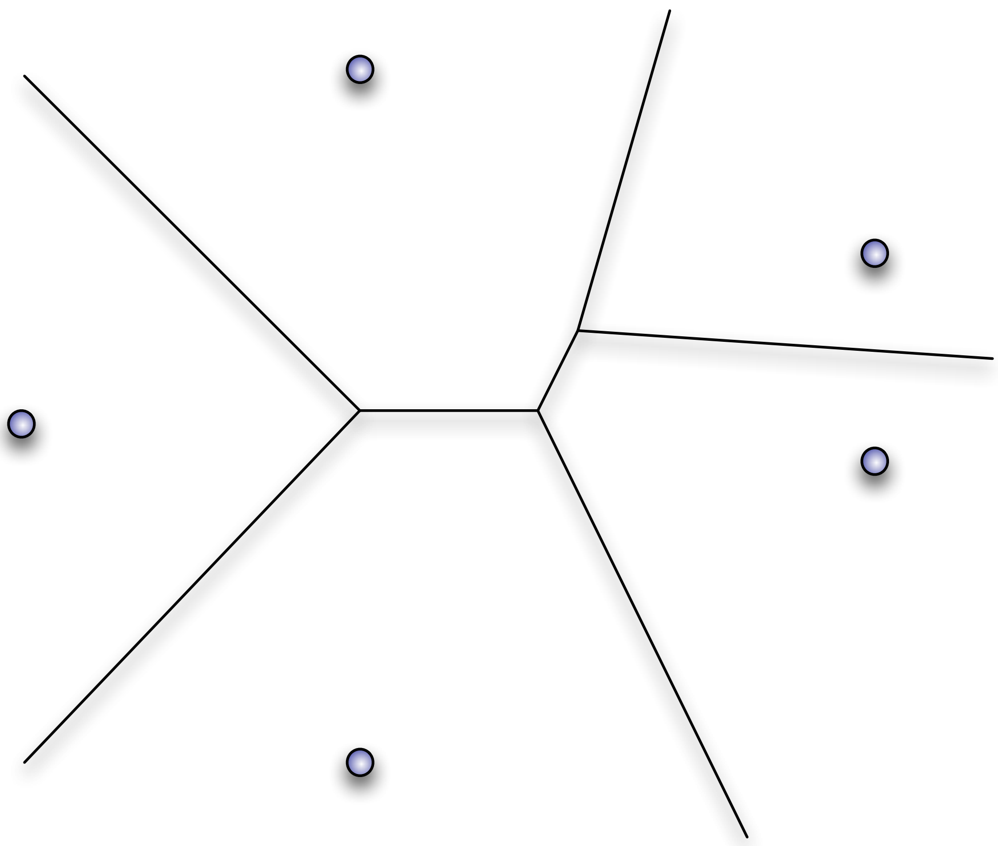
Distances are "one-dimensional" measures of influence

Regions can be shielded



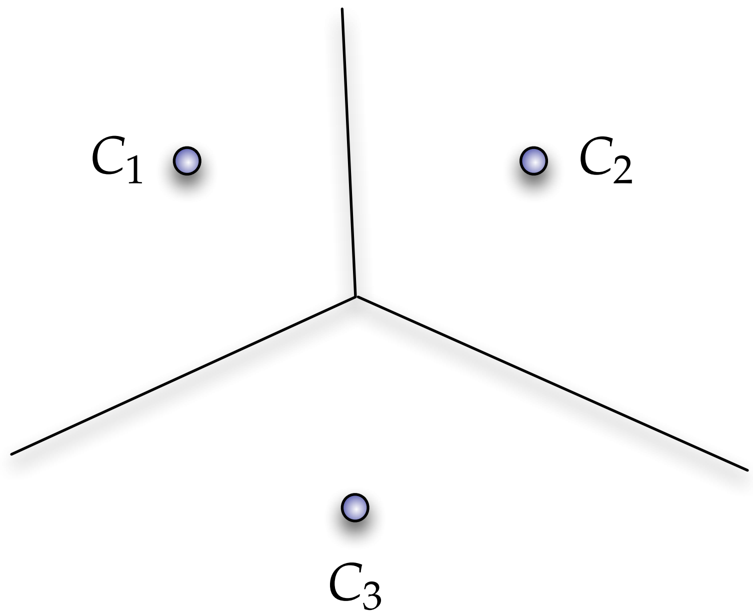
q is equidistant from C and C' (and half the distance from C*), and by distance estimation alone should have same chance of being assigned to either as to C*

Voronoi property of clusterings

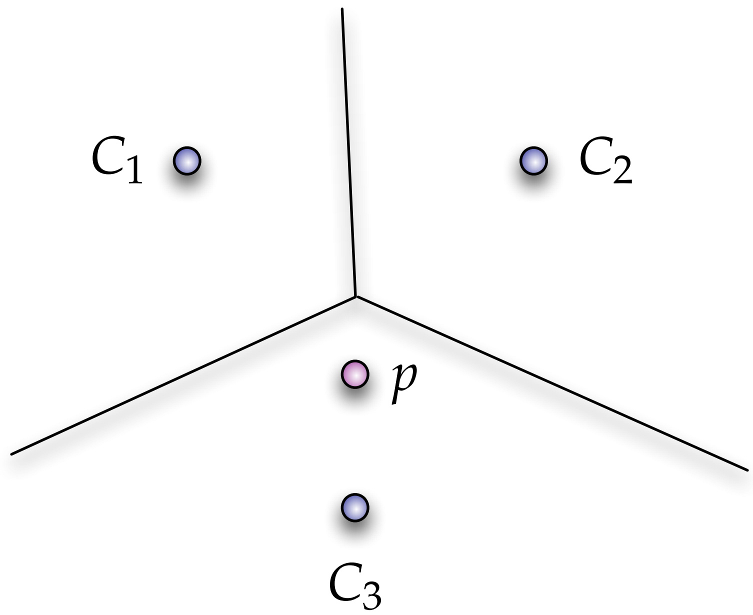


It's always better to assign a point to its nearest neighbor

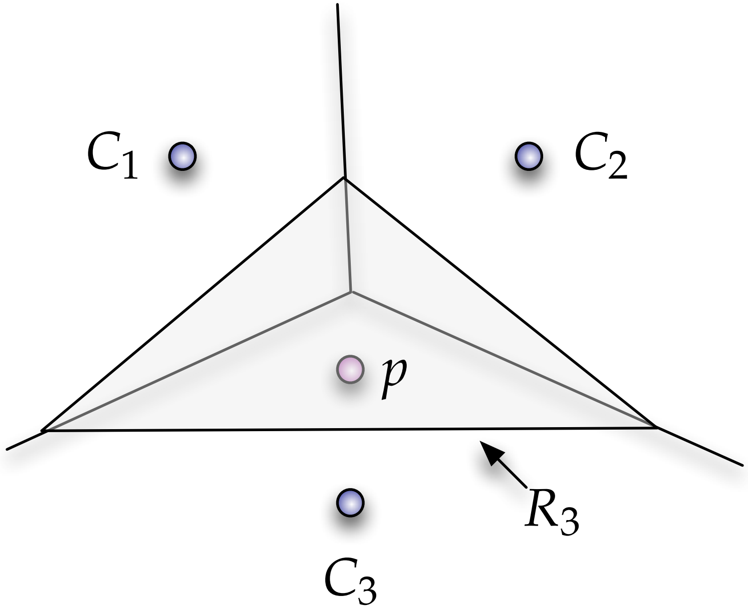
Voronoi Regions of Influence



Voronoi Regions of Influence



Voronoi Regions of Influence



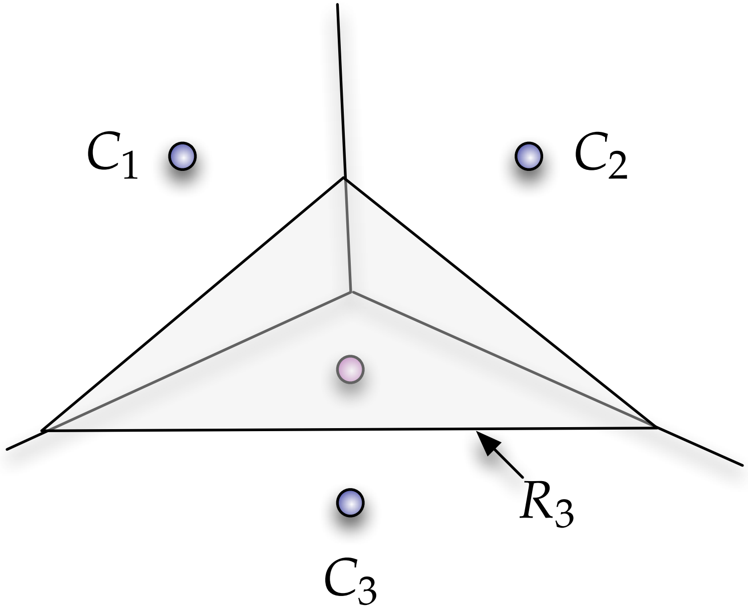
$$\alpha_i = \frac{\mathrm{Vol}(R_i)}{\mathrm{Vol}(R)}$$

Affinity of a point for a cluster is the fractional area stolen from it

$$\boldsymbol{\alpha}(p) = (\alpha_1, \ldots, \alpha_k)$$

$$\sum \alpha_i = 1$$

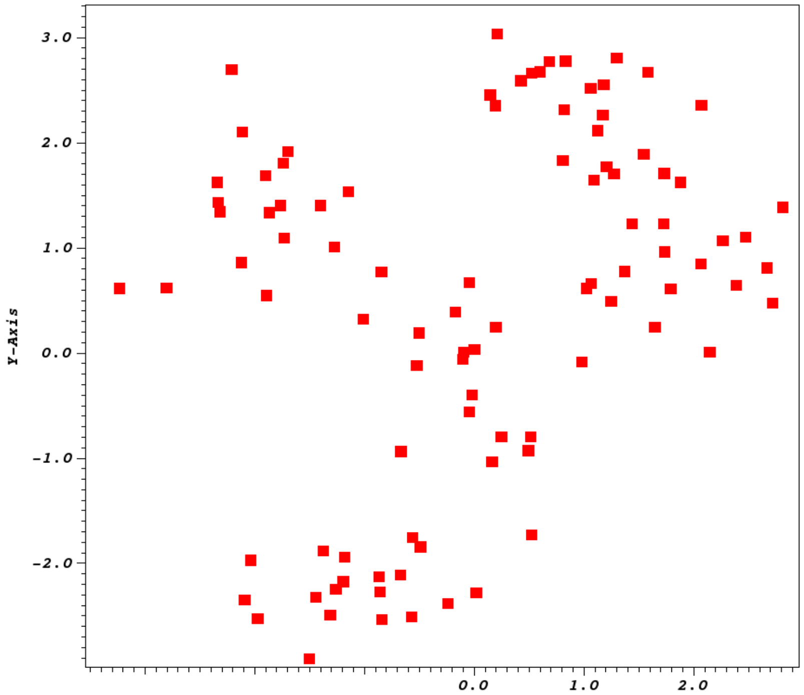
Voronoi Regions of Influence

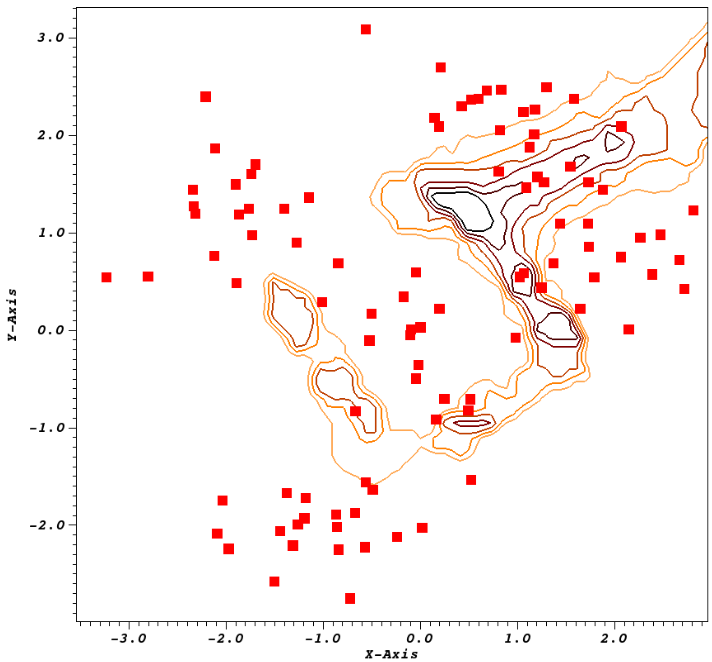


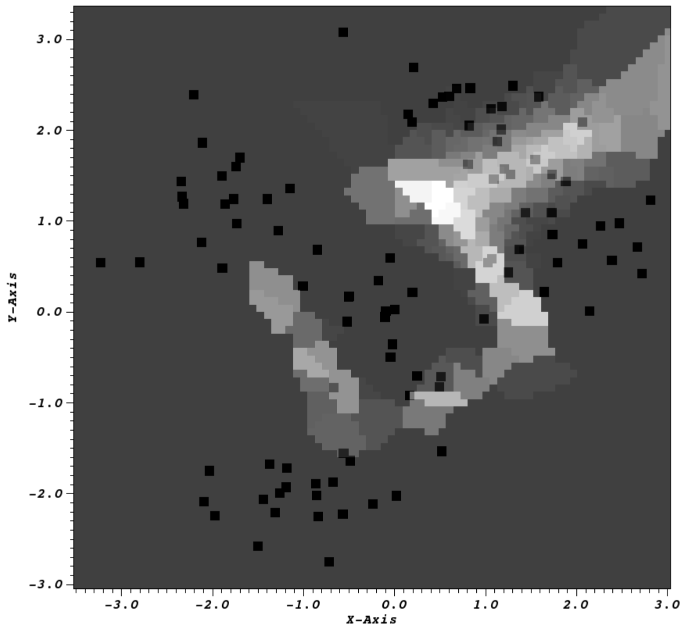
$$\alpha_i = \frac{\mathrm{Vol}(R_i)}{\mathrm{Vol}(R)}$$

Affinity of a point for a cluster is the fractional area stolen from it

- A point is "stable" if the maximum affinity is more than 0.5:
- Maximum affinity is a continuous scalar function
- This idea was first used for doing interpolation of a scalar field (natural neighbor interpolation)

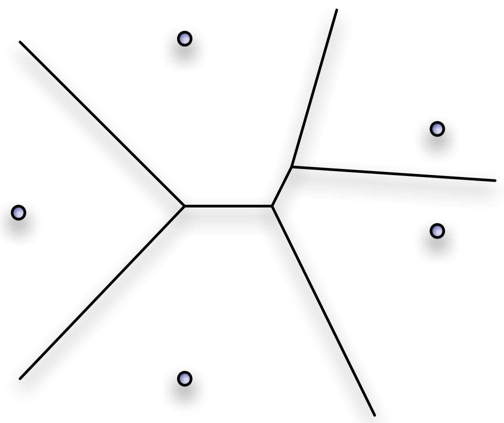

Y-Axis
3.0
2.0
1.0
0.0
-1.0
-2.0
0.0
1.0
2.0


Y-Axis
3.0
2.0
1.0
0.0
-1.0
-2.0
-3.0
-2.0
-1.0
0.0
1.0
2.0
X-Axis

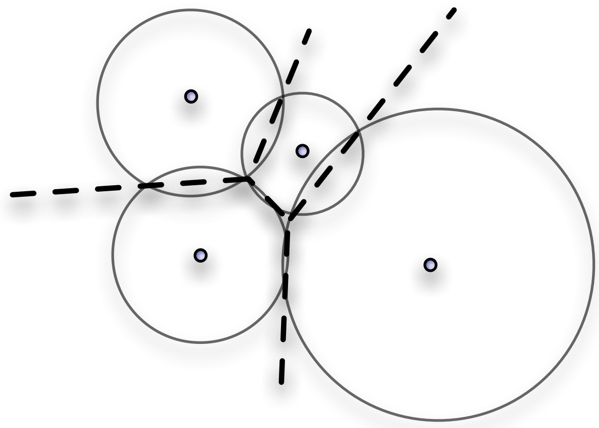

Y-Axis
3.0
2.0
1.0
0.0
-1.0
-2.0
-3.0
-3.0
-2.0
-1.0
0.0
1.0
2.0
3.0
X-Axis

Incorporating cluster density

If Voronoi diagram has polyhedral cells, then all relevant volumes are polyhedral cells.

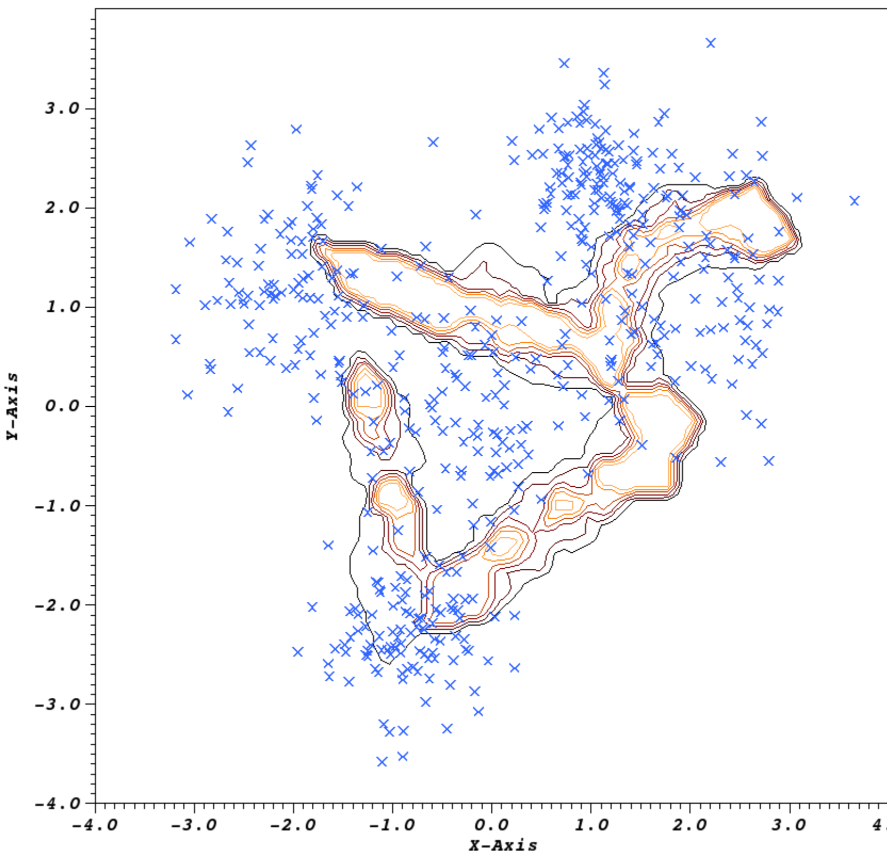


$$d(p, x) = \|p - x\|^2$$

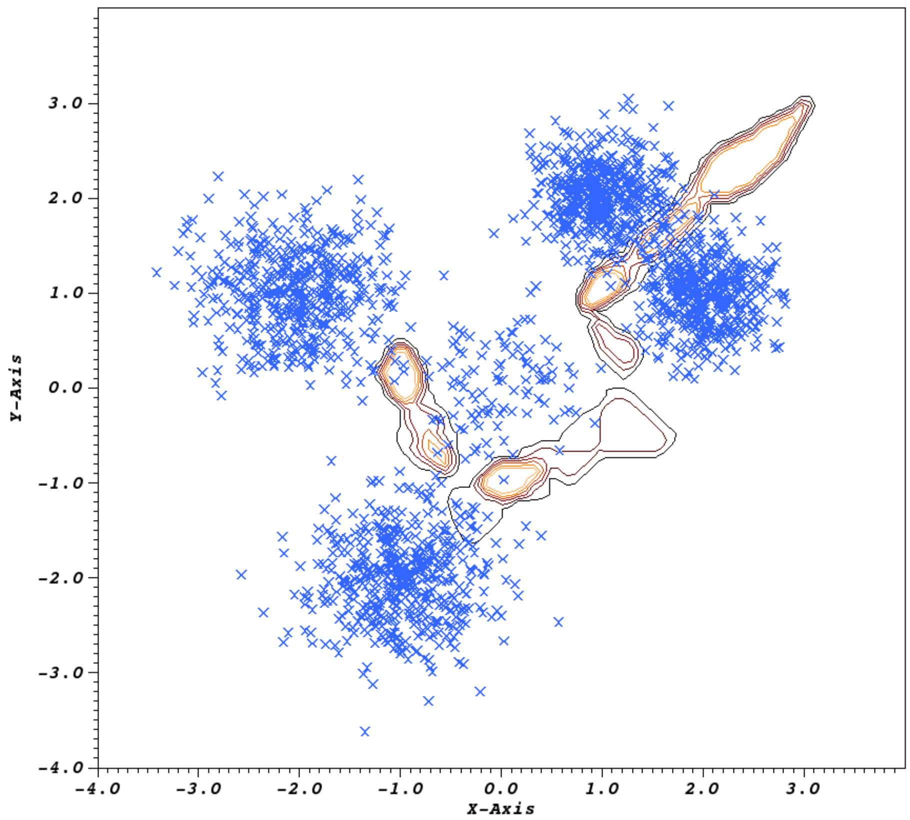


$$d(p, x) = \|p - x\|^2 - w_x$$

(power diagram)



Y-Axis
X-Axis
-4.0
-3.0
-2.0
-1.0
0.0
1.0
2.0
3.0



Y-Axis
X-Axis
-4.0
-3.0
-2.0
-1.0
0.0
1.0
2.0
3.0

Generalizing to other distance spaces

Bregman divergences: Kullback-Leibler, Itakura-Saito, ...

$$B_\phi(x, y) = \phi(x) - \phi(y) - \langle \nabla\phi(y), x - y \rangle$$

$$d(p, x) = B_\phi(p, x) - w_x$$

$$d(\mathbf{p}, x) = d(\mathbf{p}, y) \equiv c + \langle \nabla\phi(y) - \nabla\phi(x), \mathbf{p} \rangle = 0$$

Kernel distances: graphs, strings, ...

$$d(p, x) = \|\Phi(p) - \Phi(x)\|^2 - w_x$$

Computing affinity vectors

In 2D:

- Computing Voronoi diagram is $O(k \log k)$
- Intersection of two convex polygons takes $O(k)$ time
- k-vertex polygon can be triangulated in $O(k)$ time
- Area of a triangle can be computed in $O(1)$ time.

Overall: $O(k \log k)$ time per query

In 3D:

- Voronoi diagram takes $O(k^2)$ time.
- Intersection of convex polyhedra takes $O(k)$ time
- Tetrahedralization can be done in $O(k)$ time.

Overall: $O(k^2)$ time per query

Computing affinity vectors

In 2D:

- Computing Voronoi diagram is O(k log k)
- Intersection of two convex polygons takes O(k) time
- k-vertex polygon can be triangulated in O(k) time
- Area of a triangle can be computed in O(1) time.

Overall: O

Voronoi diagram in d dimensions has complexity $O(k^{d/2})$

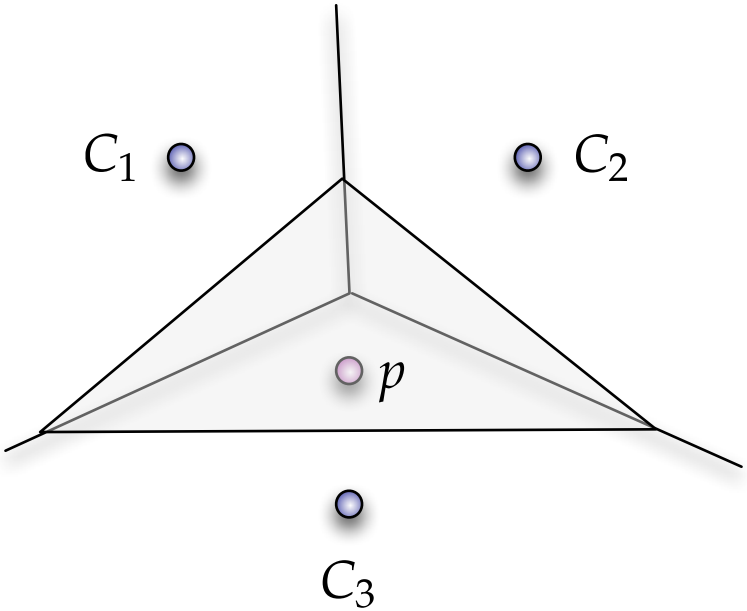
In 3D:

- Voronoi diagram takes $O(k^2)$ time.
- Intersection of convex polyhedra takes O(k) time
- Tetrahedralization can be done in O(k) time.

Overall: $O(k^2)$ time per query

Approximate Affinities

Given $\varepsilon > 0$ find $\tilde{\alpha}$ such that $|\tilde{\alpha} - \alpha| \leq \varepsilon$



Sampling algorithm:

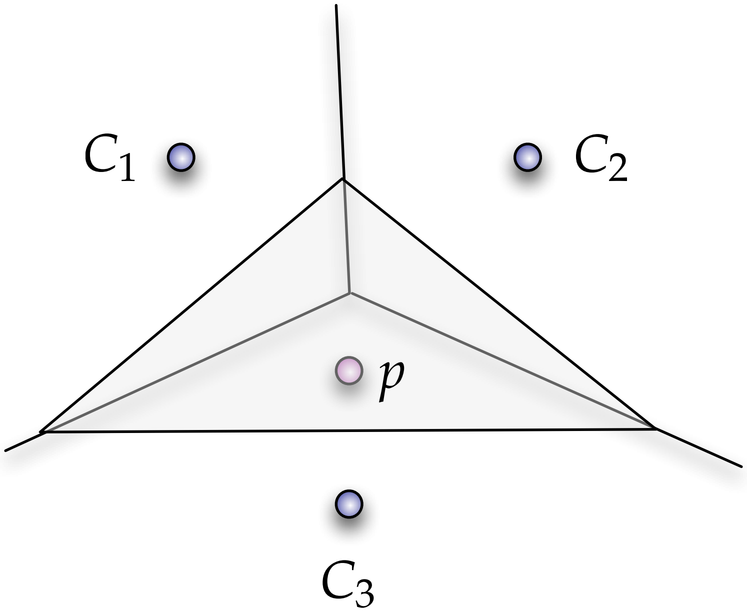
- Sample s from Voronoi cell of p
- Find second closest neighbor of s
- Increment count of that neighbor

- At end, return normalized counts.

Each sample is processed in O(k) time
Need to solve two problems:
1) How many samples to pick
2) How to sample from Voronoi cell of p

Approximate Affinities

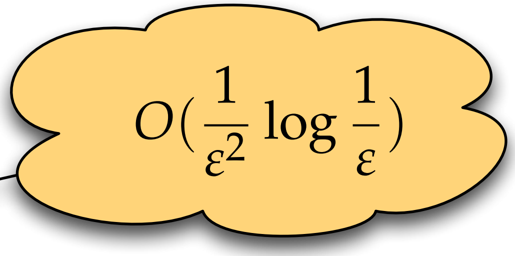
Given $\varepsilon > 0$ find $\tilde{\alpha}$ such that $|\tilde{\alpha} - \alpha| \leq \varepsilon$



Sampling algorithm:

- Sample s from Voronoi cell of p
- Find second closest neighbor of s
- Increment count of that neighbor

- At end, return normalized counts.



Each sample is processed in O(k) time
Need to solve two problems:
1) How many samples to pick
2) How to sample from Voronoi cell of p

Volume Sampling

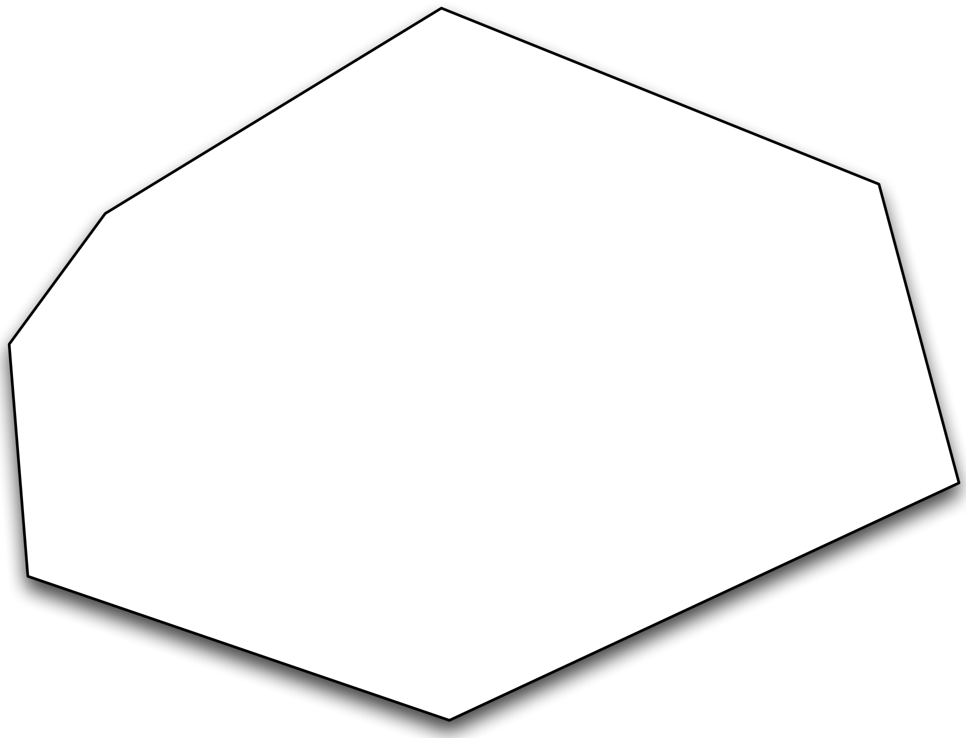
- Voronoi cell of p is a convex body
- Membership oracle is easy: "is sample nearer to p than to any other point"
- Use standard results for sampling from convex body with membership oracle

  $O^*(d^4)$ samples suffice [LV06].
- In practice, use hit-and-run sampling.

Volume Sampling

- Voronoi cell of p is a convex body
- Membership oracle is easy: "is sample nearer to p than to any other point"
- Use standard results for sampling from convex body with membership oracle

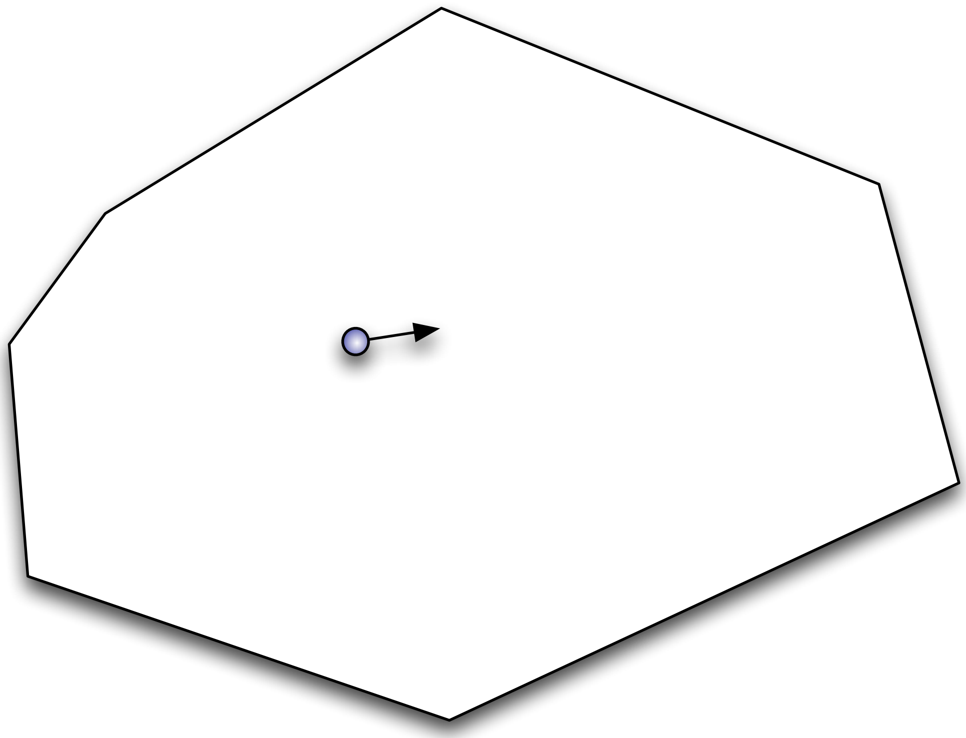
  $O^*(d^4)$ samples suffice.
- In practice, use hit-and-run sampling.



Volume Sampling

- Voronoi cell of p is a convex body
- Membership oracle is easy: "is sample nearer to p than to any other point"
- Use standard results for sampling from convex body with membership oracle

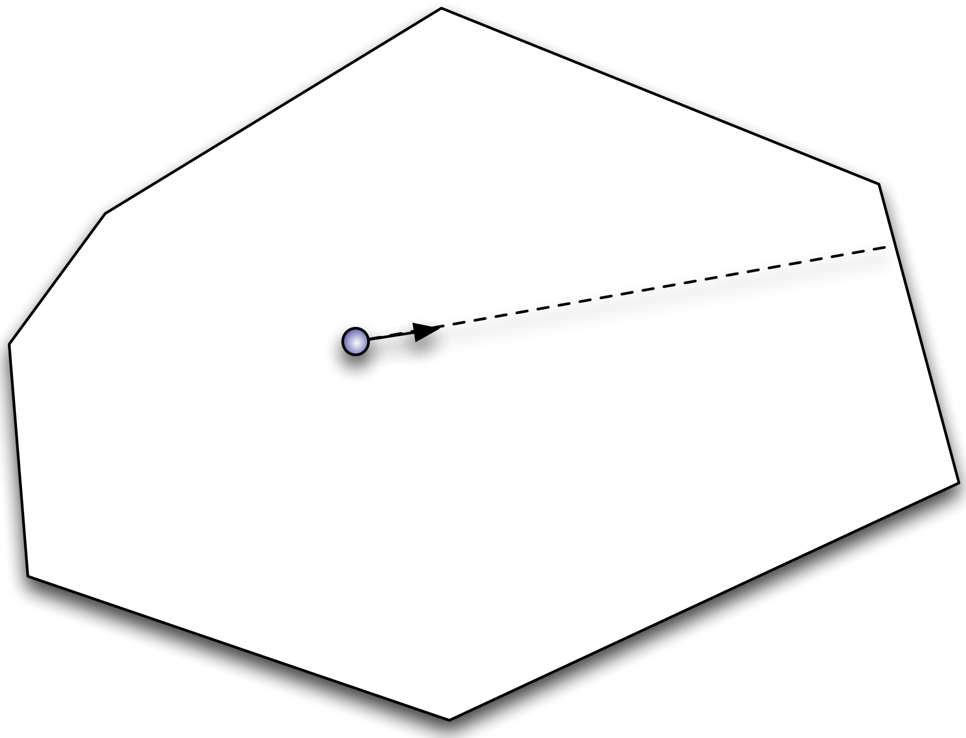
  $O^*(d^4)$ samples suffice.
- In practice, use hit-and-run sampling.



Volume Sampling

- Voronoi cell of p is a convex body
- Membership oracle is easy: "is sample nearer to p than to any other point"
- Use standard results for sampling from convex body with membership oracle

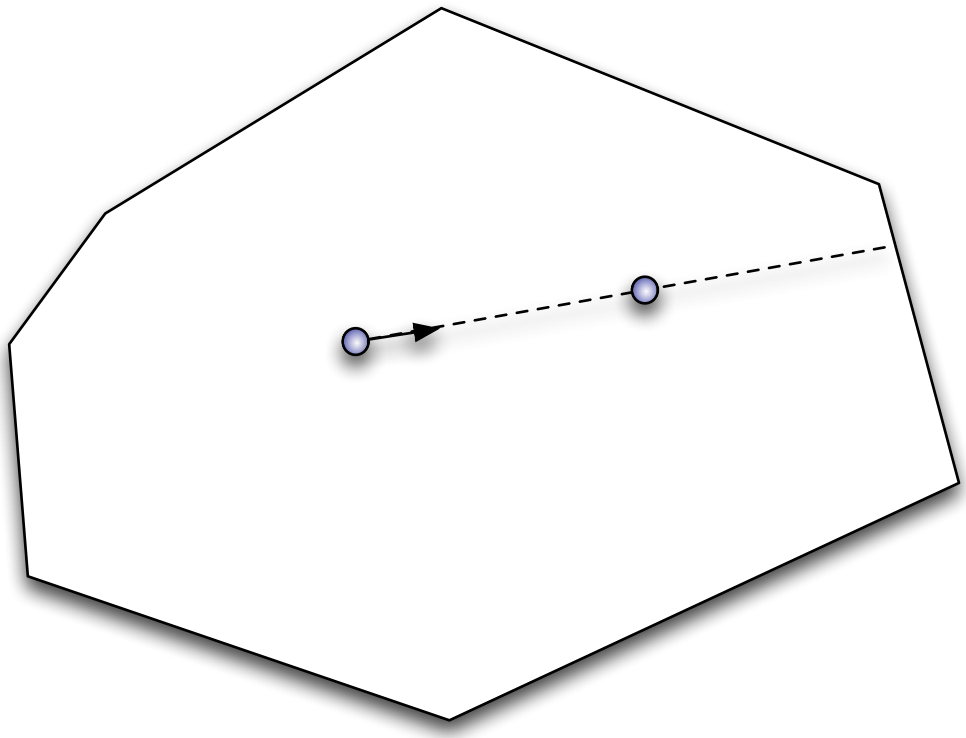
  $O^*(d^4)$ samples suffice.
- In practice, use hit-and-run sampling.



Volume Sampling

- Voronoi cell of p is a convex body
- Membership oracle is easy: "is sample nearer to p than to any other point"
- Use standard results for sampling from convex body with membership oracle

  $O^*(d^4)$ samples suffice.
- In practice, use hit-and-run sampling.



Dimensionality Reduction

Running time is polynomial in d (ambient space dimension).

Consider Euclidean distance:

$$d(x,y) = \|x - y\|^2$$

k clusters induce a k-1 dimensional space $\mathcal{H}$

$$x = u + w, u \in \mathcal{H}, w \perp u$$

$$d(x,x') = \|u - u'\|^2 + \|w - w'\|^2$$

Any Voronoi cell can be written as

$$V = V' + \mathcal{H}^{\perp}, V' \in \mathcal{H}$$

Volume ratios need only be measured in $\mathcal{H}$

Algorithm Summary

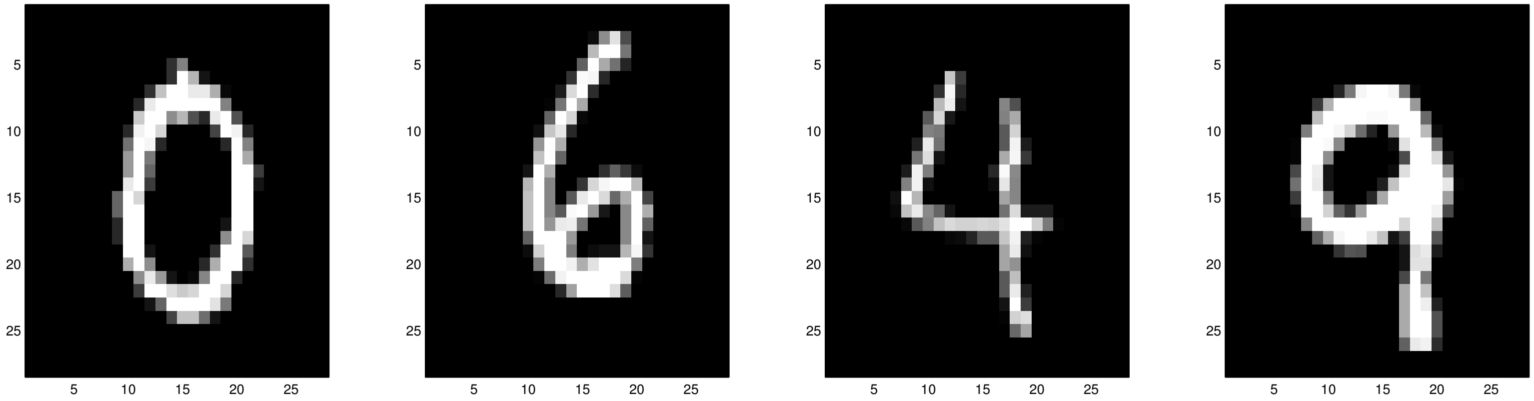
Given k clusters and query p

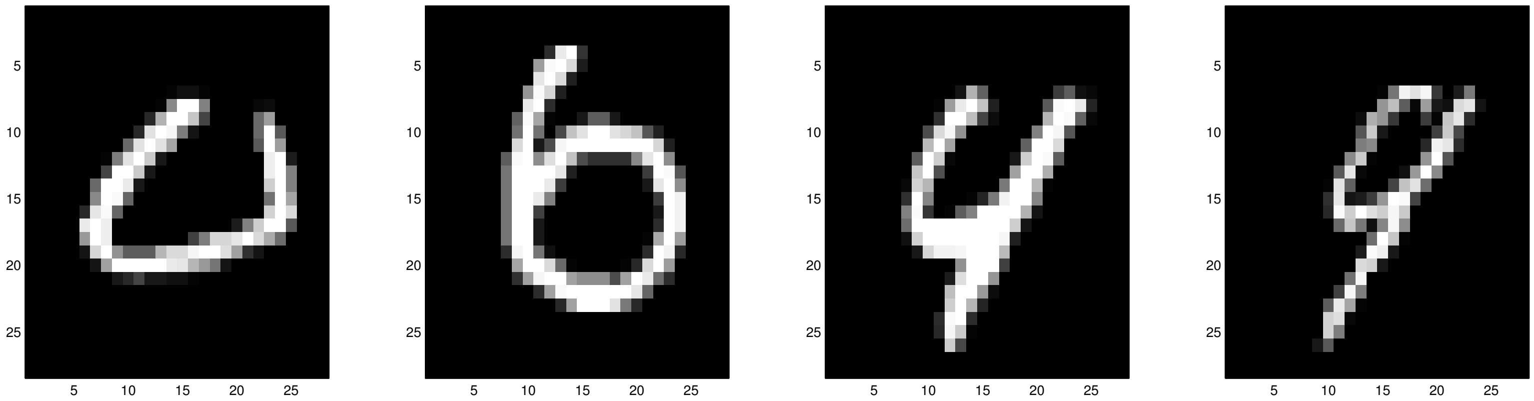
- Project clusters to k-dimensional space
- Sample uniformly from Voronoi cell of p
- Compute frequencies of second-nearest neighbors
- Return approximate affinity scores

Overall running time: poly(k, $1/\varepsilon$)
In practice: on the order of milliseconds/query.

Clustering digits

Highly stable points





Highly unstable points

Accelerating Clustering

"Active clustering": only pick points that inform true decision boundary

Idea: use affinity scores to identify points that might lie on boundary

- Use fast procedure to generate cluster centers (k-means++ initialization)
- Sample points with low affinity scores, as well as few points with high affinity scores.
- Cluster reduced sample.

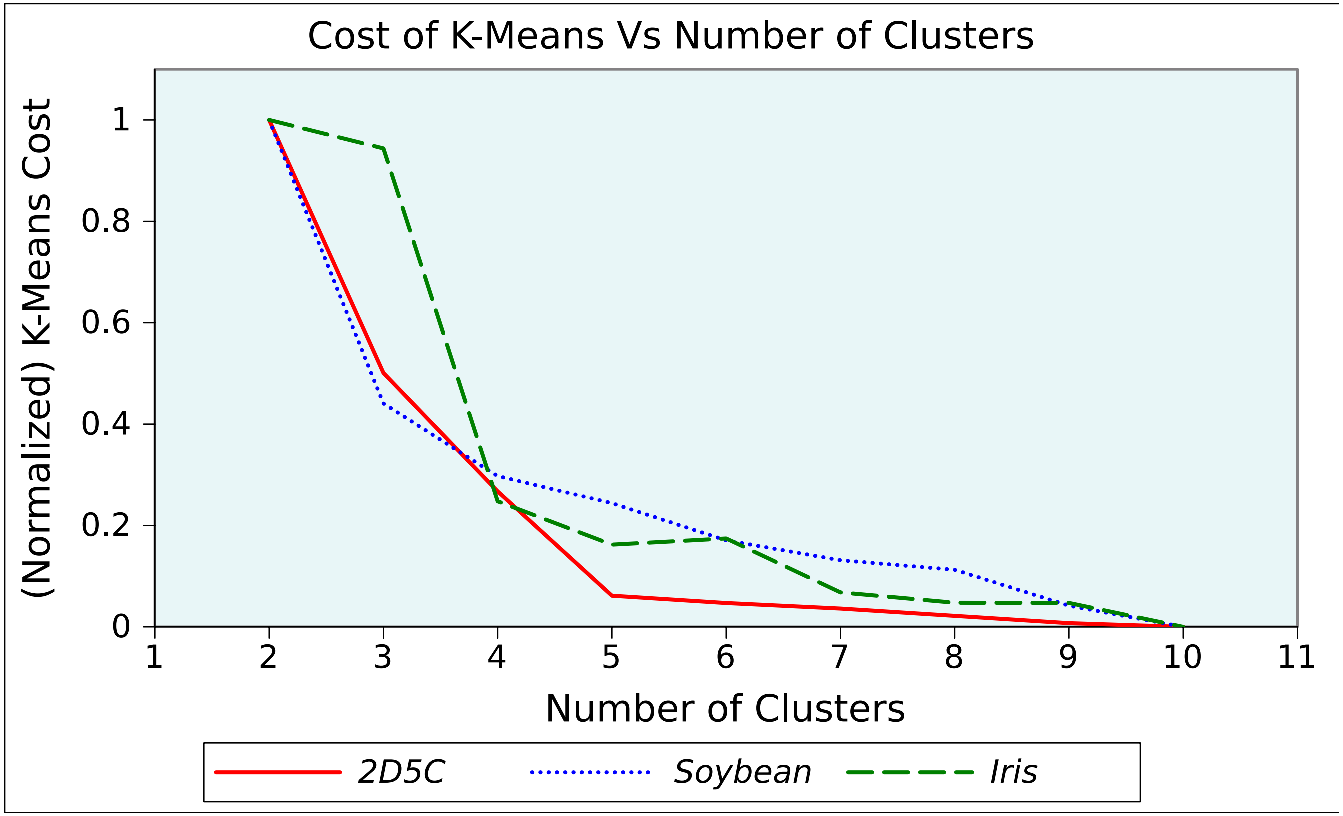
Result: comparable accuracy of clustering with orders of magnitude speedup

Work in progress (with Kilian Weinberger): speeding up classification algorithms using affinity scores.

Points with low affinity scores act as sparse "skeleton" of data set.

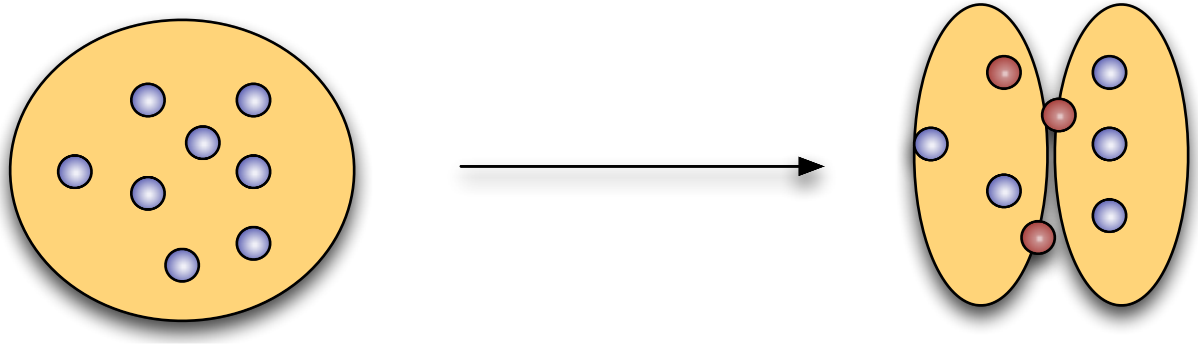
Model Selection

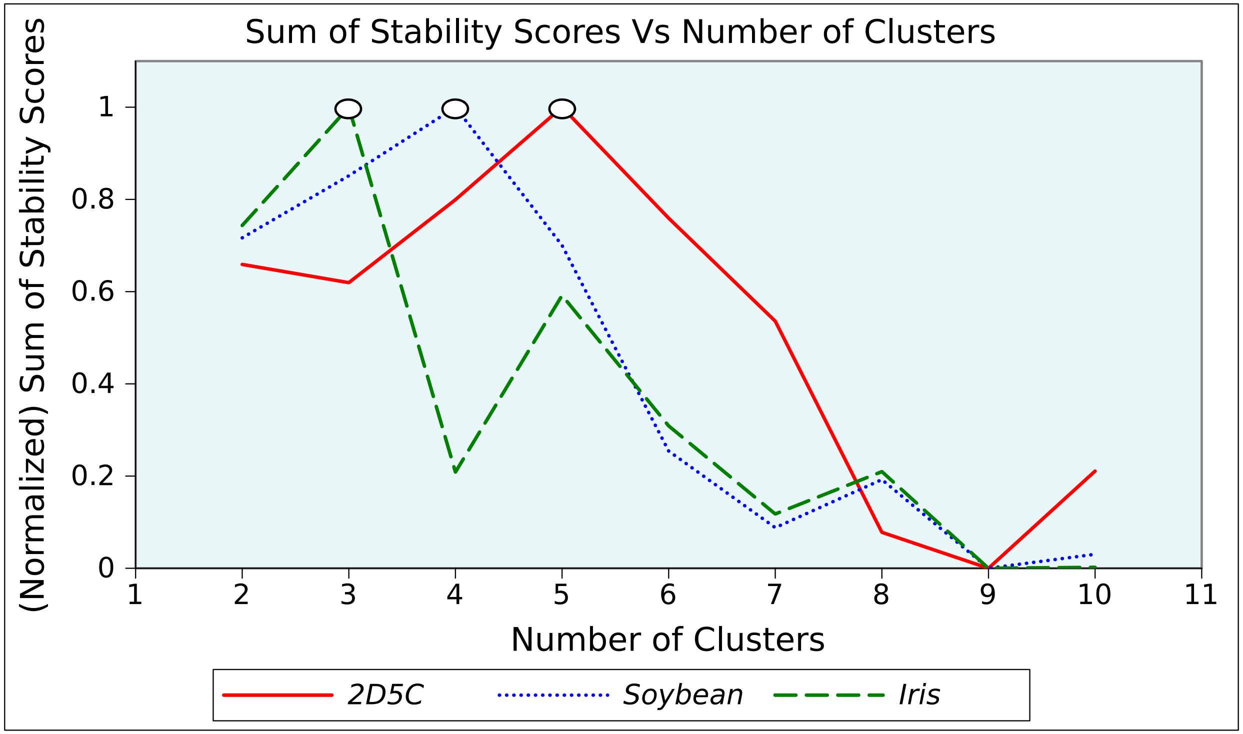
How do we choose the right "k" for a clustering with k centers ?



Model Selection

Average stability does not increase monotonically with increasing clusters





Questions

Can affinity scores be correlated with the probabilities extracted from a clustering model ?

The (maximum) affinities define a (scalar) field over the data. Can topological methods like persistence help to identify "interesting" parts of the space ?

Can we compute points of low affinity (the data skeleton) quickly (without exploring the entire space) ?

Are there other applications where affinity scores can be used as an accelerant ?