K-means and Hierarchical Clustering

Note to other teachers and users of these slides. Andrew would be delighted if you found this source material useful in giving your own lectures. Feel free to use these slides verbatim, or to modify them to fit your own needs. PowerPoint originals are available. If you make use of a significant portion of these slides in your own lecture, please include this message, or the following link to the source repository of Andrew's tutorials: http://www.cs.cmu.edu/~awm/tutorials . Comments and corrections gratefully received.

Andrew W. Moore

Professor

School of Computer Science

Carnegie Mellon University

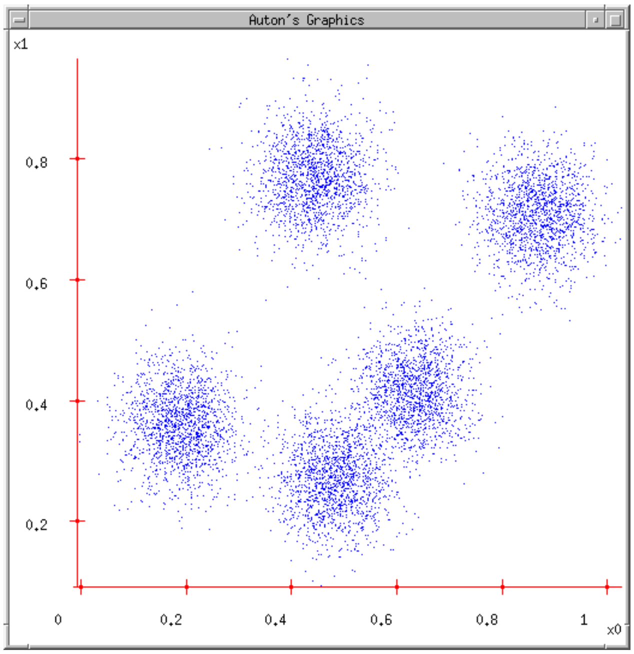
www.cs.cmu.edu/~awm

awm@cs.cmu.edu

412-268-7599

 Nov 16th, 2001

Some Data



This could easily be modeled by a Gaussian Mixture (with 5 components)

But let's look at an satisfying, friendly and infinitely popular alternative…

Lossy Compression

Suppose you transmit the coordinates of points drawn randomly from this dataset.

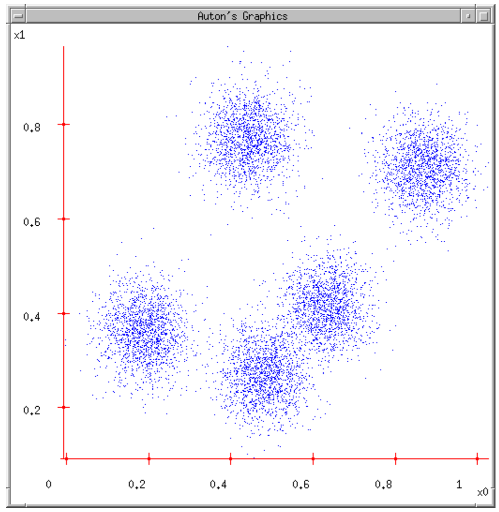
You can install decoding software at the receiver.

You're only allowed to send two bits per point.

It'll have to be a "lossy transmission".

Loss = Sum Squared Error between decoded coords and original coords.

What encoder/decoder will lose the least information?



Idea One

Suppose you transmit the coordinates of points drawn randomly from this dataset.

You can install decoding software at the receiver.

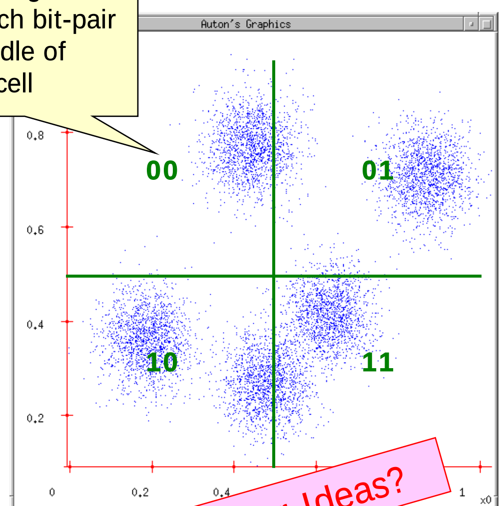
You're only allowed to send two bits per point.

It'll have to be a "lossy transmission".

Loss = Sum Squared Error between decoded coords and original coords.

What encoder/decoder will lose the least information?

Break into a grid, decode each bit-pair as the middle of each grid-cell



Any Better Ideas?

Idea Two

Suppose you transmit the coordinates of poi randomly from thi

You can install de software at the receiver.

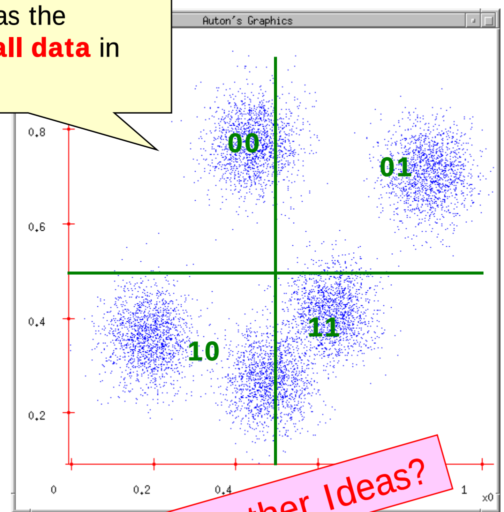
You're only allowed to send two bits per point.

It'll have to be a "lossy transmission".

Loss = Sum Squared Error between decoded coords and original coords.

What encoder/decoder will lose the least information?

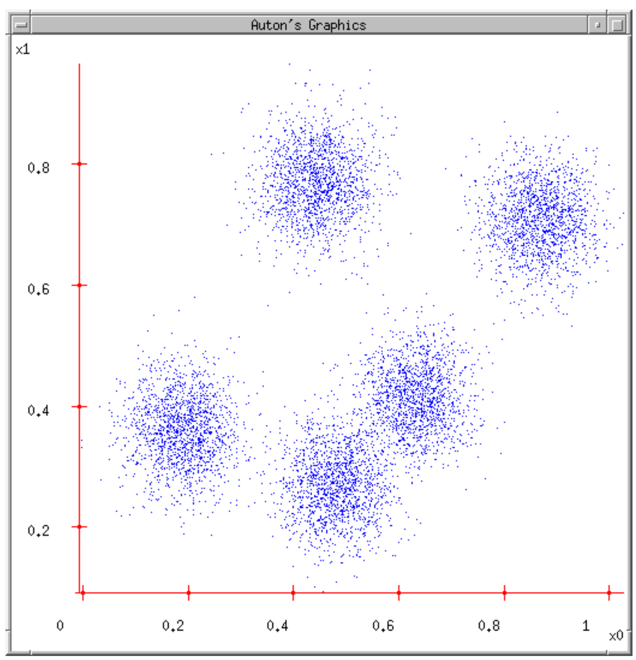
Break into a grid, decode each bit-pair as the **centroid of all data** in that grid-cell



Any Further Ideas?

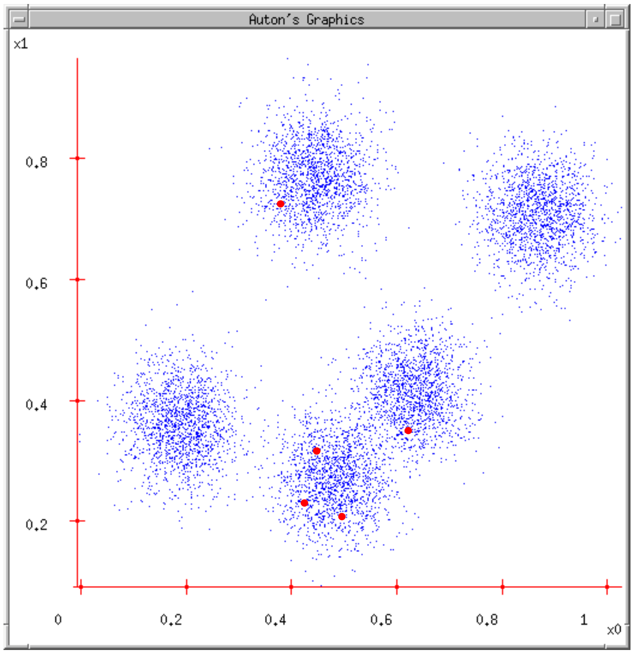
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*



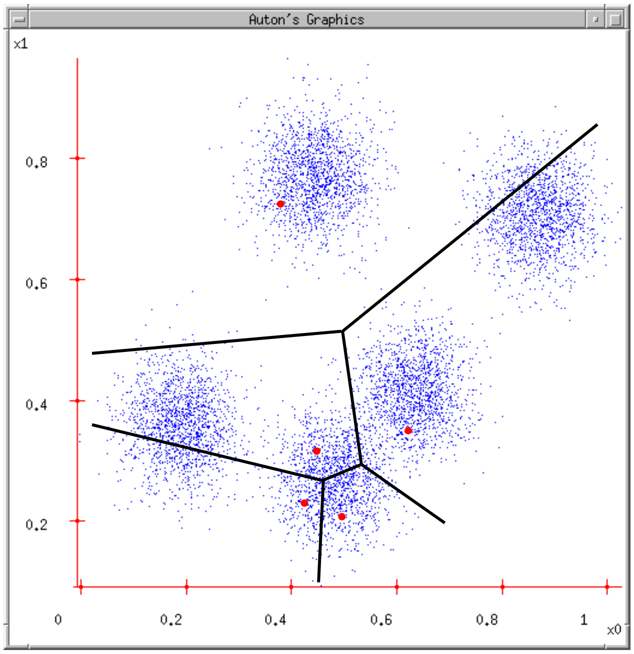
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations



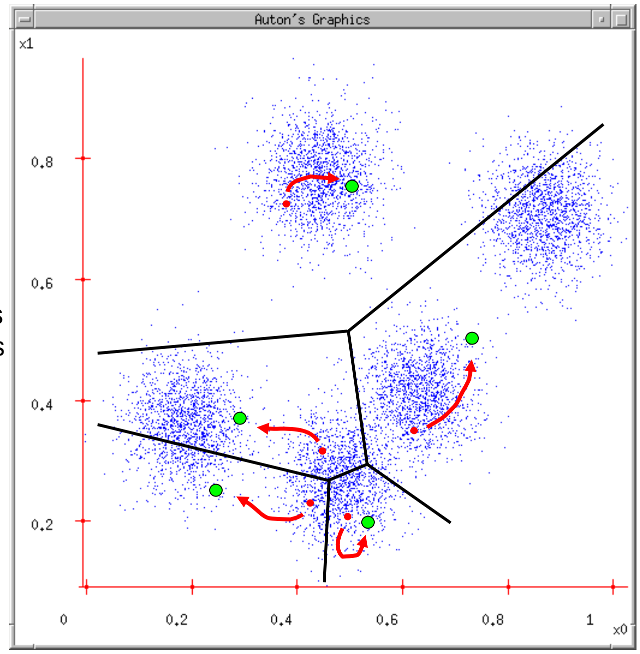
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to. (Thus each Center "owns" a set of datapoints)



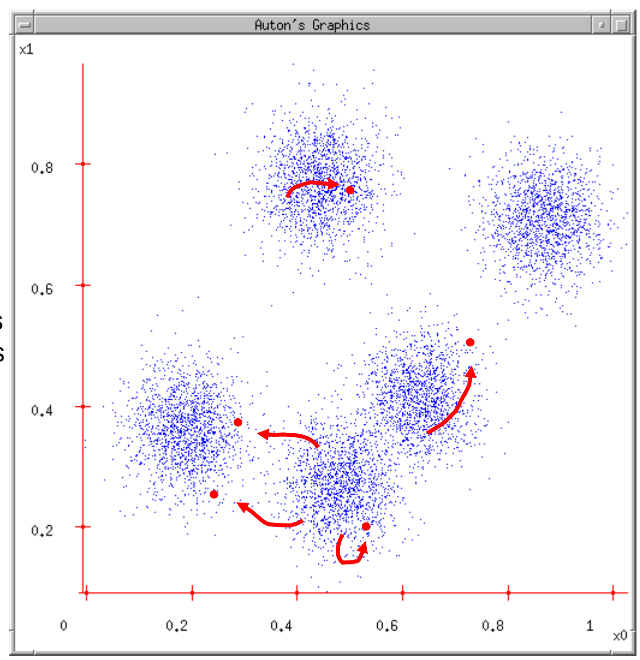
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns



K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns…
5. …and jumps there
6. …Repeat until terminated!

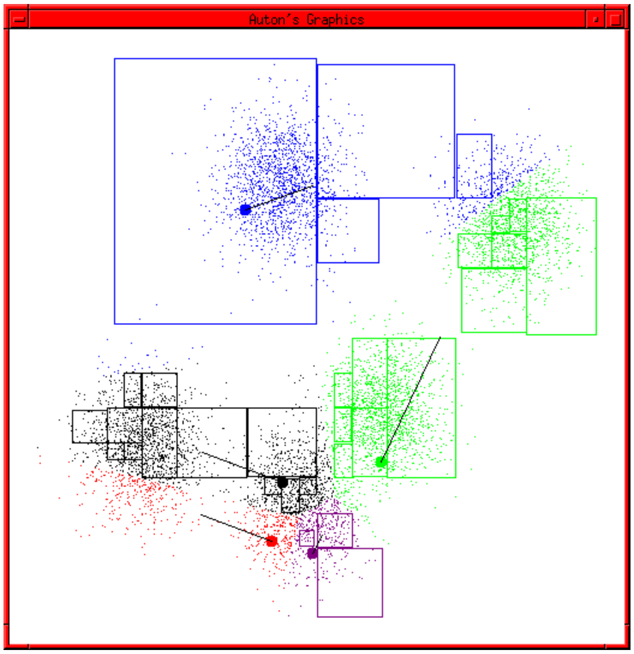


K-means Start

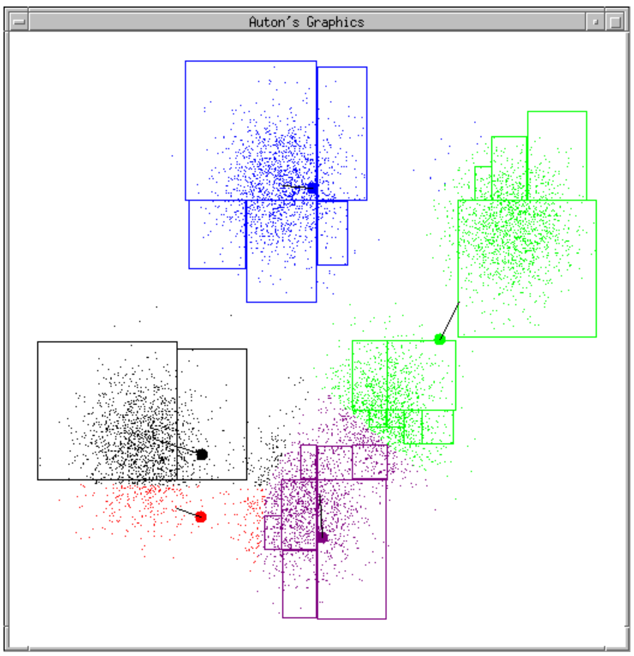
Advance apologies: in Black and White this example will deteriorate

Example generated by Dan Pelleg's super-duper fast K-means system:

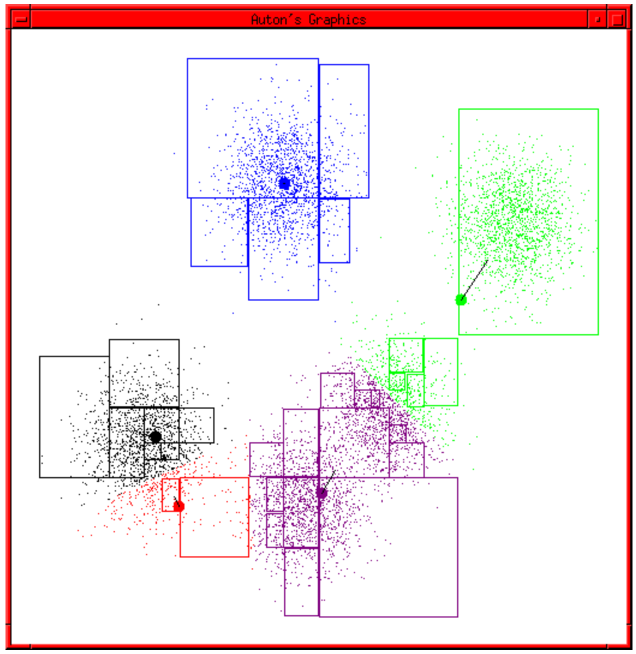
Dan Pelleg and Andrew Moore. Accelerating Exact k-means Algorithms with Geometric Reasoning. Proc. Conference on Knowledge Discovery in Databases 1999, (KDD99) (available on www.autonlab.org/pap.html*)*



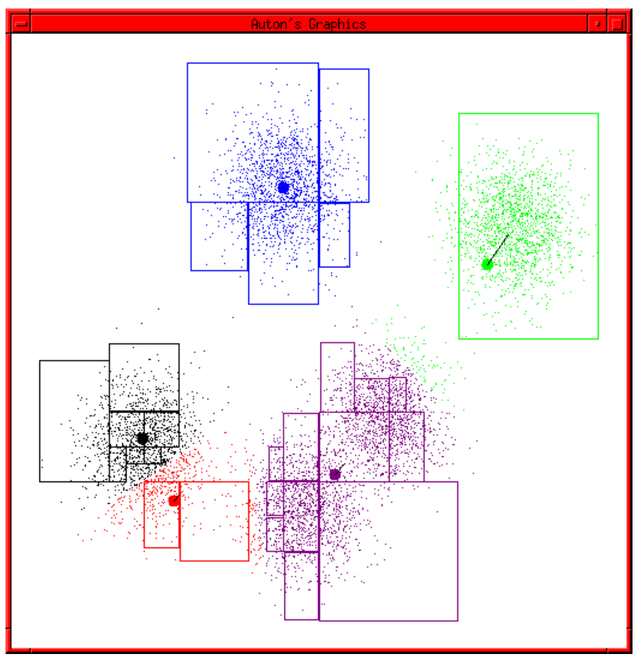
K-means continues …



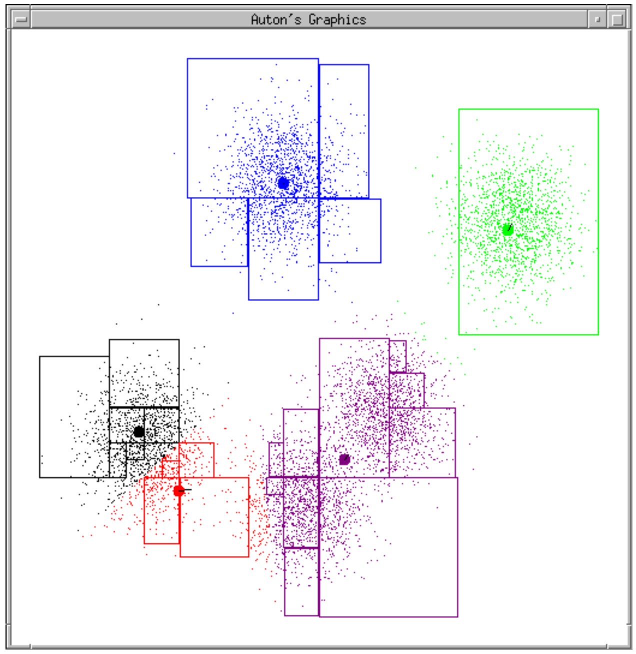
K-means continues …



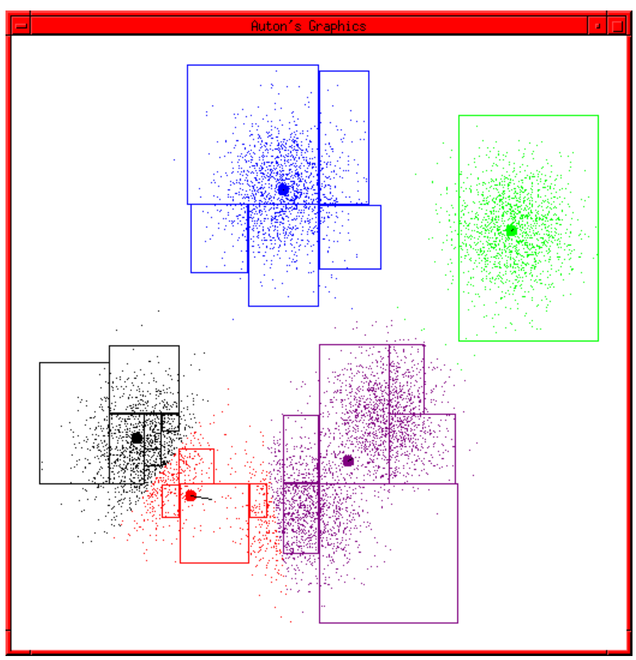
K-means continues …



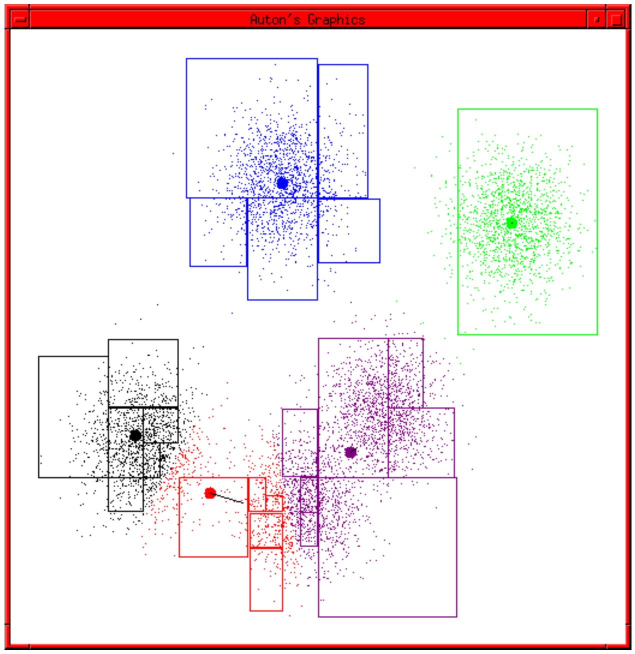
K-means continues …



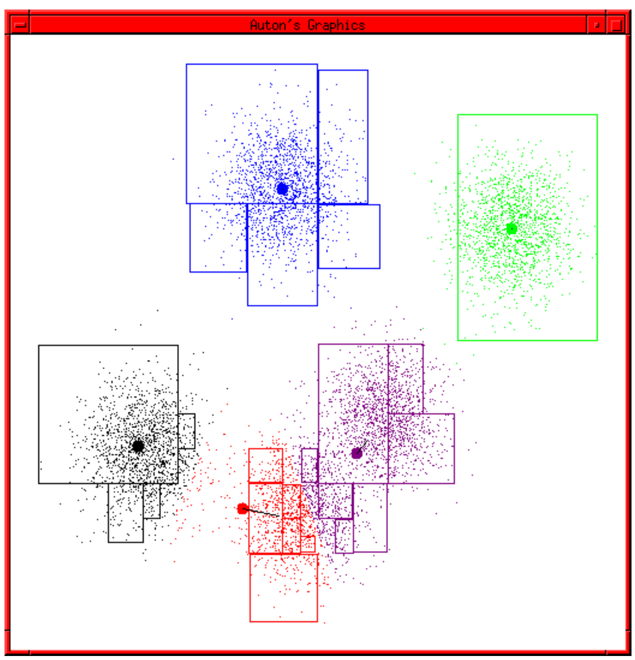
K-means continues …



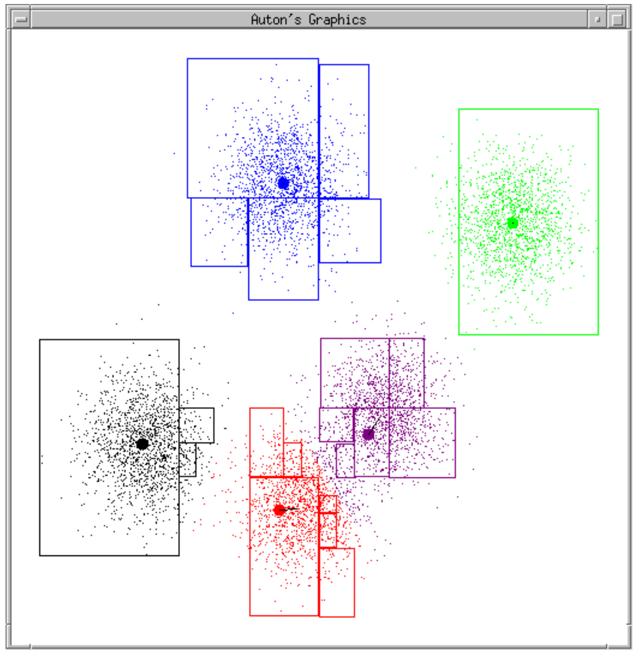
K-means continues …



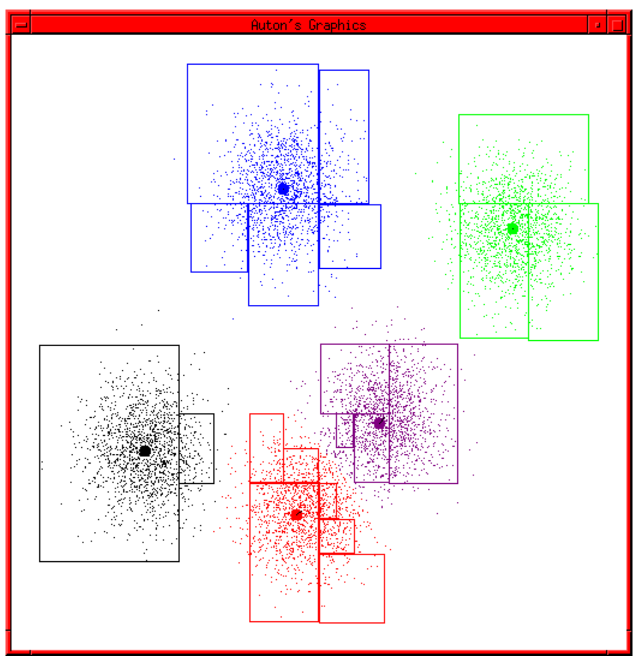
K-means continues …



K-means continues …



K-means terminates



K-means Questions

- What is it trying to optimize?
- Are we sure it will terminate?
- Are we sure it will find an optimal clustering?
- How should we start it?
- How could we automatically choose the number of centers?

....we'll deal with these questions over the next few slides

Distortion

Given..

•an encoder function: ENCODE : $\Re^m \rightarrow [1..k]$

•a decoder function: DECODE : $[1..k] \rightarrow \Re^m$

Define...

$$\text{Distortion} = \sum_{i=1}^{R} \left(\mathbf{x}_i - \text{DECODE}[\text{ENCODE}(\mathbf{x}_i)]\right)^2$$

Distortion

Given..

•an encoder function: ENCODE : $\Re^m \rightarrow [1..k]$

•a decoder function: DECODE : $[1..k] \rightarrow \Re^m$

Define...

$$\text{Distortion} = \sum_{i=1}^{R} \left(\mathbf{x}_i - \text{DECODE}[\text{ENCODE}(\mathbf{x}_i)]\right)^2$$

We may as well write

$$\text{DECODE}[j] = \mathbf{c}_j$$

so $$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

The Minimal Distortion

$$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

What properties must centers c_1 , c_2 , ... , c_k have when distortion is minimized?

The Minimal Distortion (1)

$$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

What properties must centers c_1 , c_2 , … , c_k have when distortion is minimized?

(1) x_i must be encoded by its nearest center

….why?

$$\mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)} = \underset{\mathbf{c}_j \in \{\mathbf{c}_1, \mathbf{c}_2, \ldots \mathbf{c}_k\}}{\arg\min} (\mathbf{x}_i - \mathbf{c}_j)^2$$

..at the minimal distortion

The Minimal Distortion (1)

$$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

What properties must centers c_1 , c_2 , … , c_k have when distortion is minimized?

(1) x_i must be encoded by its nearest center

….why?

Otherwise distortion could be reduced by replacing ENCODE[x_i] by the nearest center

$$\mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)} = \underset{\mathbf{c}_j \in \{\mathbf{c}_1, \mathbf{c}_2, \ldots \mathbf{c}_k\}}{\arg\min} (\mathbf{x}_i - \mathbf{c}_j)^2$$

..at the minimal distortion

The Minimal Distortion (2)

$$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

What properties must centers c_1 , c_2 , ... , c_k have when distortion is minimized?

(2) The partial derivative of Distortion with respect to each center location must be zero.

(2) The partial derivative of Distortion with respect to each center location must be zero.

$$\begin{aligned}
\text{Distortion} &= \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2 \\
&= \sum_{j=1}^{k} \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j)^2
\end{aligned}$$

OwnedBy(c_j) = the set of records owned by Center c_j .

$$\begin{aligned}
\frac{\partial \text{Distortion}}{\partial \mathbf{c}_j} &= \frac{\partial}{\partial \mathbf{c}_j} \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j)^2 \\
&= -2 \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j) \\
&= 0 \text{ (for a minimum)}
\end{aligned}$$

(2) The partial derivative of Distortion with respect to each center location must be zero.

$$\begin{aligned}\text{Distortion} &= \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2 \\ &= \sum_{j=1}^{k} \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j)^2\end{aligned}$$

$$\begin{aligned}\frac{\partial \text{Distortion}}{\partial \mathbf{c}_j} &= \frac{\partial}{\partial \mathbf{c}_j} \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j)^2 \\ &= -2 \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j) \\ &= 0 \text{ (for a minimum)}\end{aligned}$$

Thus, at a minimum: $\mathbf{c}_j = \frac{1}{|\text{OwnedBy}(\mathbf{c}_j)|} \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} \mathbf{x}_i$

At the minimum distortion

$$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

What properties must centers c_1 , c_2 , ... , c_k have when distortion is minimized?

(1) x_i must be encoded by its nearest center

(2) Each Center must be at the centroid of points it owns.

Improving a suboptimal configuration...

$$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

What properties can be changed for centers $\boldsymbol{c}_1$, $\boldsymbol{c}_2$, ... , $\boldsymbol{c}_k$ have when distortion is not minimized?

(1) Change encoding so that $\boldsymbol{x}_i$ is encoded by its nearest center

(2) Set each Center to the centroid of points it owns.

There's no point applying either operation twice in succession.

But it can be profitable to alternate.

...And that's K-means!

Easy to prove this procedure will terminate in a state at which neither (1) or (2) change the configuration. Why?

Improving a suboptimal configu...on...

There are only a finite number of ways of partitioning R records into k groups.

So there are only a finite number of possible configurations in which all Centers are the centroids of the points they own.

If the configuration changes on an iteration, it must have improved the distortion.

So each time the configuration changes it must go to a configuration it's never been to before.

So if it tried to go on forever, it would eventually run out of configurations.

What p... $\boldsymbol{c}_k$
have w...

(1) Cha... center

(2) Set ...

There's ...ssion.

But it ca... profitable to alternate.

...And that's K-means!

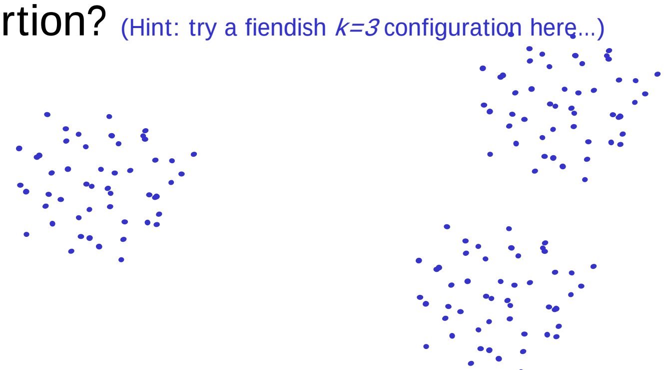
Easy to prove this procedure will terminate in a ...ate at which neither (1) or (2) change the configuration. Why?

Will we find the optimal configuration?

- Not necessarily.
- Can you invent a configuration that has converged, but does not have the minimum distortion?

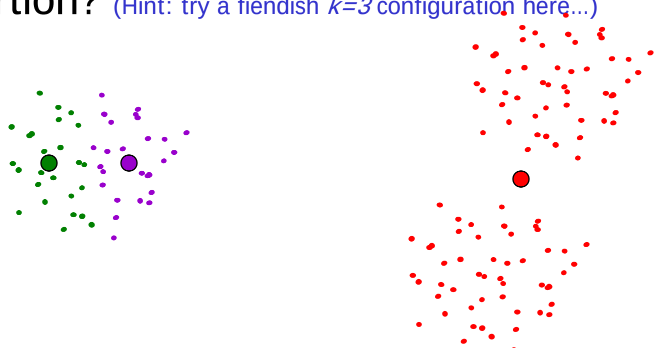
Will we find the optimal configuration?

- Not necessarily.
- Can you invent a configuration that has converged, but does not have the minimum distortion? (Hint: try a fiendish *k=3* configuration here...)



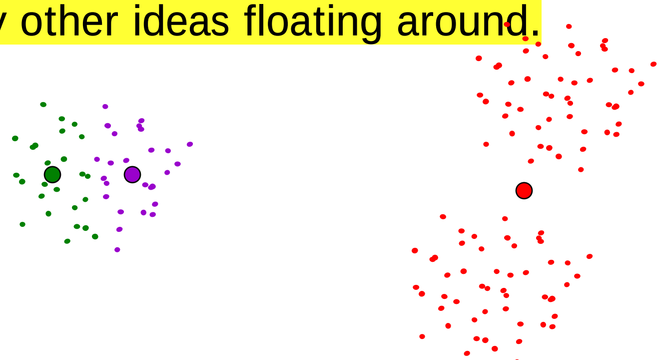
Will we find the optimal configuration?

- Not necessarily.
- Can you invent a configuration that has converged, but does not have the minimum distortion? (Hint: try a fiendish *k=3* configuration here...)



Trying to find good optima

- Idea 1: Be careful about where you start
- Idea 2: Do many runs of k-means, each from a different random start configuration
- Many other ideas floating around.



Trying to find good optima

- Idea 1: Be careful about where you start
- Idea 2: D ny runs of k-means, each
  fro
- Ma

Neat trick:

Place first center on top of randomly chosen datapoint.

Place second center on datapoint that's as far away as possible from first center

:

Place j'th center on datapoint that's as far away as possible from the closest of Centers 1 through j-1

:

Choosing the number of Centers

- A difficult problem
- Most common approach is to try to find the solution that minimizes the Schwarz Criterion (also related to the BIC)

$$\text{Distortion} + \lambda\,(\#\text{parameters}) \log R$$

$$= \text{Distortion} + \lambda m k \log R$$

m=#dimensions

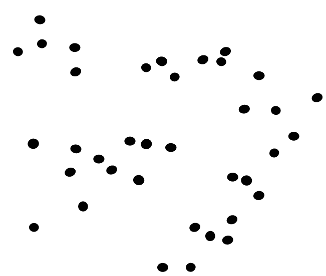
k=#Centers

R=#Records

Common uses of K-means

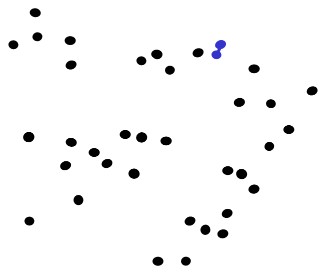
- Often used as an exploratory data analysis tool
- In one-dimension, a good way to quantize real-valued variables into k non-uniform buckets
- Used on acoustic data in speech understanding to convert waveforms into one of k categories (known as Vector Quantization)
- Also used for choosing color palettes on old fashioned graphical display devices!

Single Linkage Hierarchical Clustering



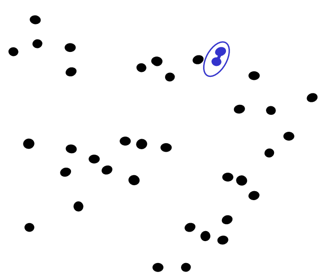
1. Say "Every point is its own cluster"

Single Linkage Hierarchical Clustering



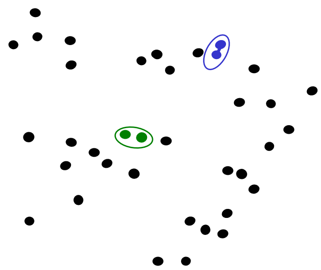
1. Say “Every point is its own cluster”
2. Find “most similar” pair of clusters

Single Linkage Hierarchical Clustering



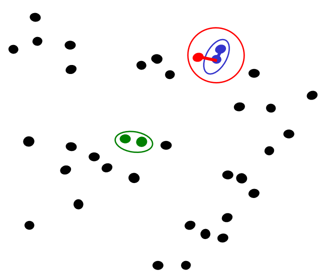
1. Say “Every point is its own cluster”
2. Find “most similar” pair of clusters
3. Merge it into a parent cluster

Single Linkage Hierarchical Clustering

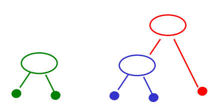


1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat

Single Linkage Hierarchical Clustering



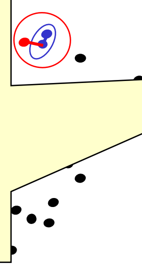
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat



Single Linkage Hierarchical
ustering

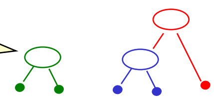
How do we define similarity between clusters?

- Minimum distance between points in clusters (in which case we're simply doing Euclidian Minimum Spanning Trees)
- Maximum distance between points in clusters
- Average distance between points in clusters



1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat...until you've merged the whole dataset into one cluster

You're left with a nice dendrogram, or taxonomy, or hierarchy of datapoints (not shown here)



Also known in the trade as Hierarchical Agglomerative Clustering (note the acronym)

Single Linkage Comments

- It's nice that you get a hierarchy instead of an amorphous collection of groups
- If you want k groups, just cut the (k-1) longest links
- There's no real statistical or information-theoretic foundation to this. Makes your lecturer feel a bit queasy.

What you should know

- All the details of K-means
- The theory behind K-means as an optimization algorithm
- How K-means can get stuck
- The outline of Hierarchical clustering
- Be able to contrast between which problems would be relatively well/poorly suited to K-means vs Gaussian Mixtures vs Hierarchical clustering