Search

Fork Sign in

Jhelum Chakravorty

1

Published Aug 10
Fork of Unsupervised Learning by Kris Sankaran

Unsupervised Learning

IFT6758, Fall 2020

Reading: ISLR sections 10.1, 10.2 and PDS pg. 462 - 476

What is unsupervised learning?

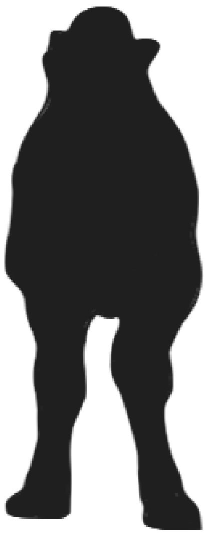
- Supervised learning: Learn some mapping $x_i \rightarrow y_i$
- Unsupervised learning
  - Usual definition: "Exploring the data x_i"
- Less orthodox interpretations
  - Learning with hidden labels (clusters: missing classes)
  - Data compression for human consumption
  - Generation of derived data for human consumption

Common Themes

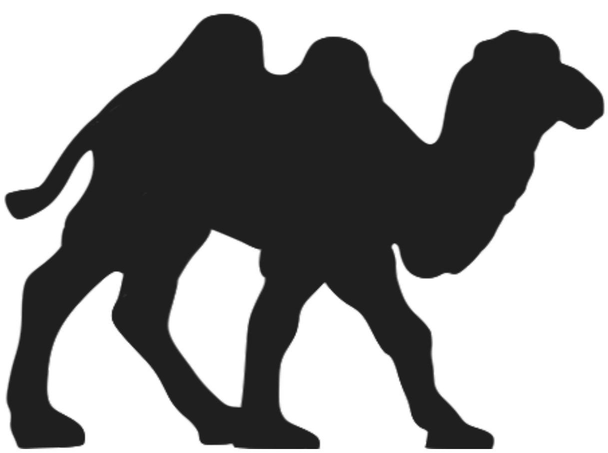
- Much harder to evaluate
- Most methods can be categorized as either,
  - Dimensionality reduction: Few features that summarize many
  - Clustering: Few "prototypes" that are representative of whole dataset
- We'll review canonical examples,
  - PCA (dimensionality reduction)
  - K-means and Hierarchical Clustering (clustering)

?

PCA
What is this?



PCA
What is this?



Credit for the idea: Prof. Julie Josse

PCA

Idea: Certain views of high-dimensional data are more informative than others.

Can you find a low-dimensional representation with as much variation as possible?

PCA

Can you find a low-dimensional representation with as much variation as possible?

To implement the idea,

- What will be the candidate family of low-dimensional representations?
- How will we choose one of the many candidates?

Candidates: Linear Mixings

Can you find a **low-dimensional representation** with as much variation as possible?

- For a representation, consider linear combinations of high-dimensional vectors,

$$\begin{aligned} z_i &= \sum_{j=1}^{p} \varphi_{1j} x_{ij} \\ &= \varphi_1^T x_i \end{aligned}$$

- φ_1 is a free parameter. E.g., if $\varphi_1 = \left(\frac{1}{n}, \dots, \frac{1}{n}\right)$, then we summarize x_i by averaging over its coordinates

Selection Criteria: Maximal Variance

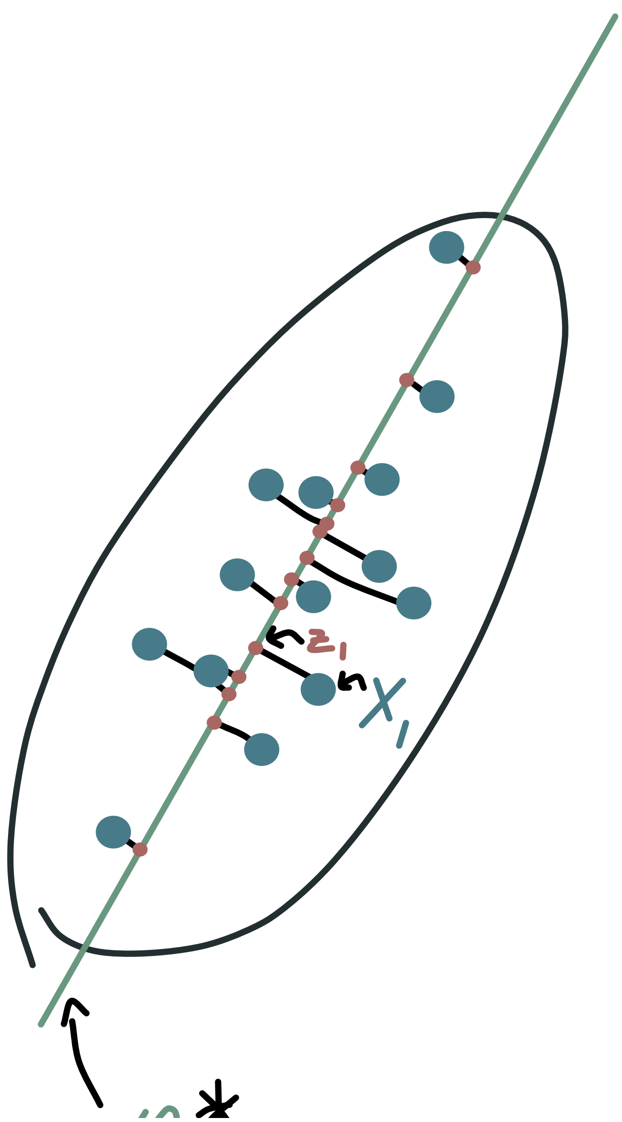
Can you find a low-dimensional representation with **as much variation** as possible?

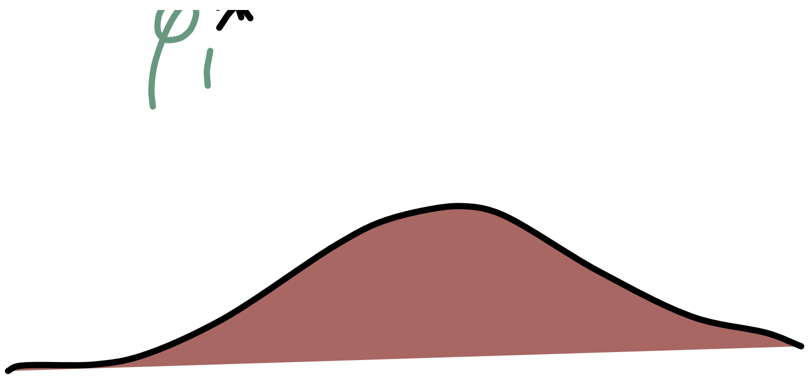
- The z_i's should be as spread out as possible:

$$\text{maximize}_{\varphi_1} \frac{1}{n}\sum_{i=1}^{n} z_i^2$$

  - Subject to constraint $\|\varphi_1\|^2 = 1$.

See the example here. The red arrow is φ_1 and the points along the 1D axes are the associated z_i's.





Selection Criteria: Maximal Variance

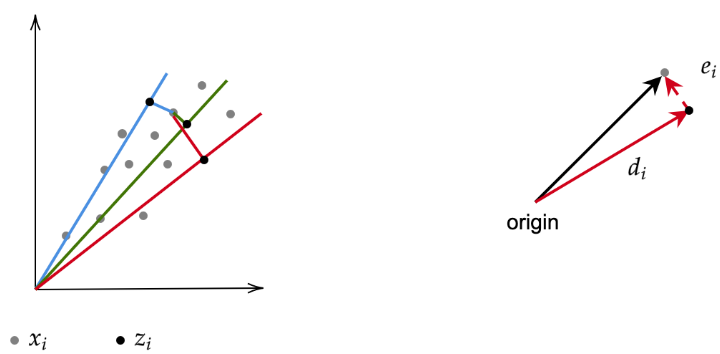
Can you find a low-dimensional representation with **as much variation** as possible?

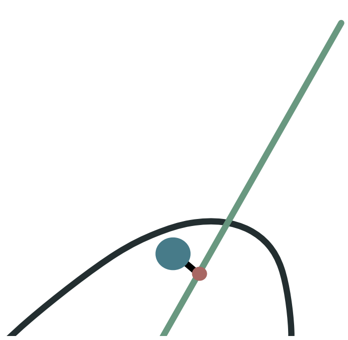
- The z_i's should be as spread out as possible:

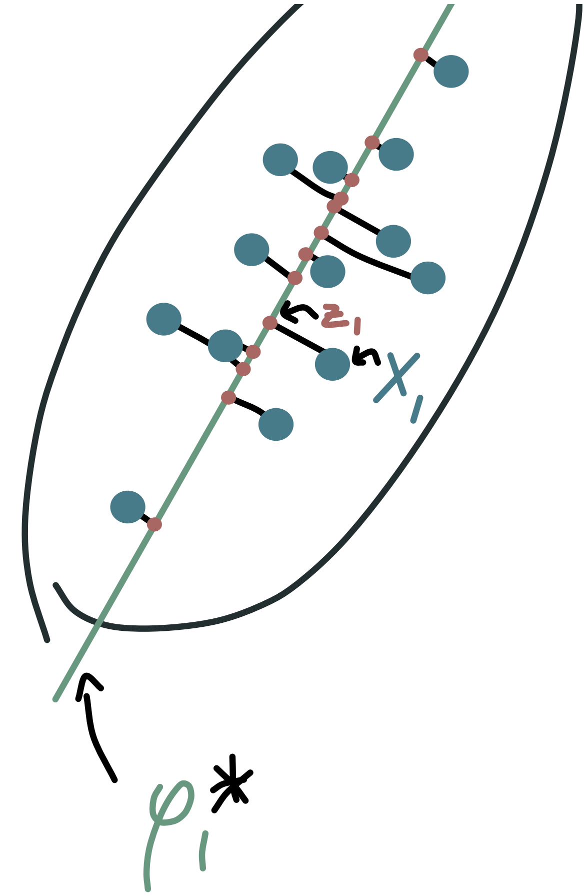
$$\text{maximize}_{\varphi_1} \frac{1}{n} \sum_{i=1}^{n} z_i^2$$

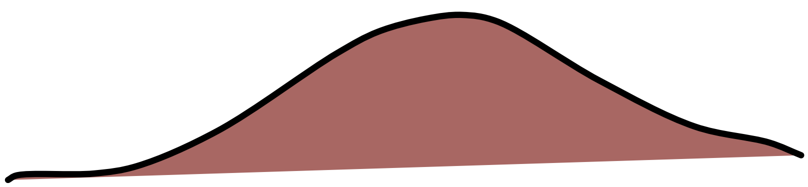
  - Subject to constraint $\|\varphi_1\|^2 = 1$.

See the example here. The red arrow is φ_1 and the points along the 1D axes are the associated z_i's.



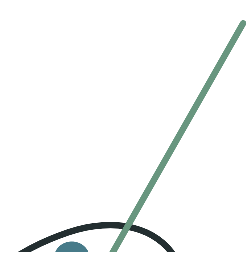


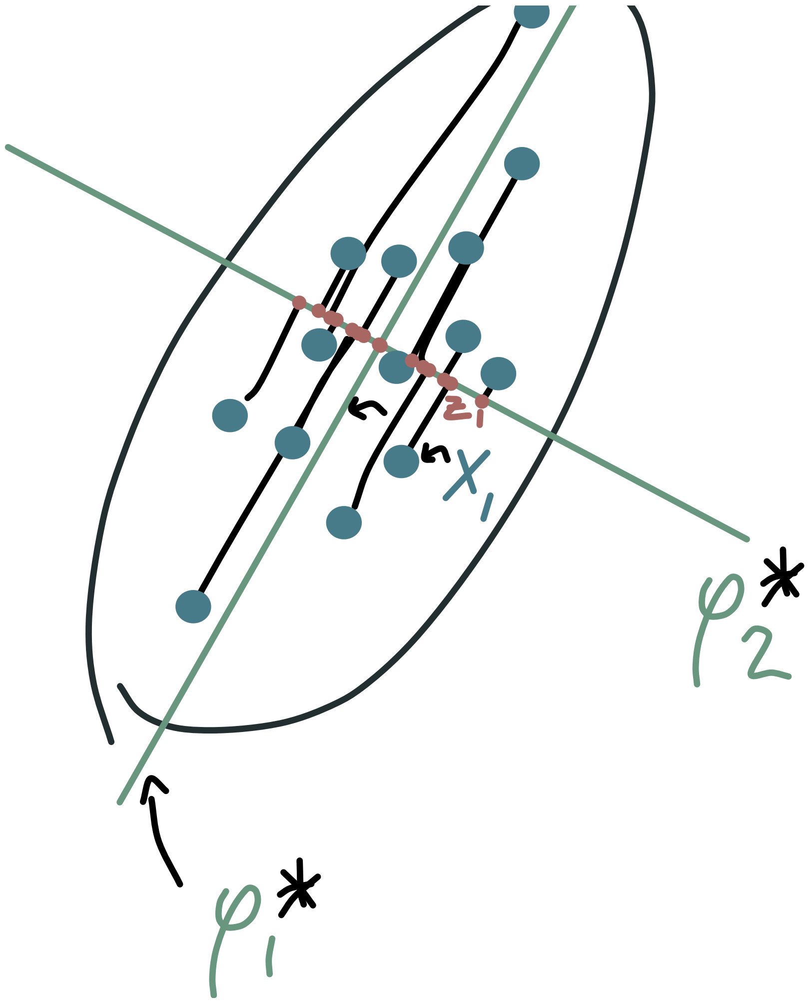




Second, third, ... PCA directions

- Once you find φ_1, you can find a "second" direction φ_2
- Found by solving the exact same optimization, but with a new constraint that it's at 90 degrees to the previous directions
- Interpretation: PCA is finding a new, better coordinate system for your data





Semantics

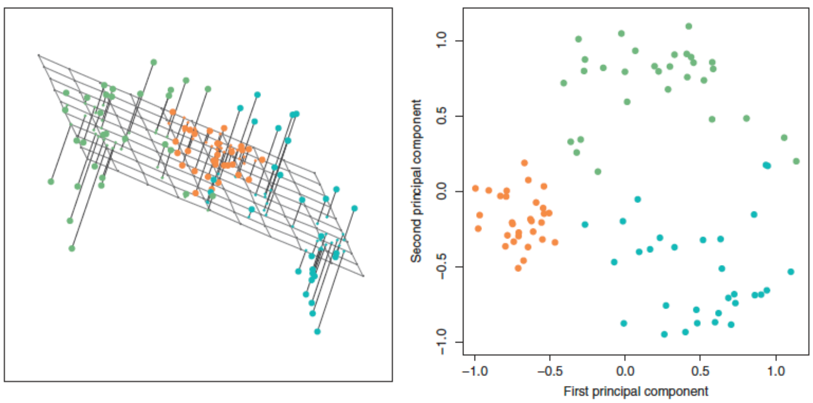
- The φ_k are the PCA "directions" or "components."
- The z_i are called "scores."
  - Interpreted as coordinates with respect to the directions φ_k.

Alternative Interpretation: Linear Approximation

- The first K directions in PCA find the best K-dimensional linear approximation (using sum of squared error to measure approximation quality).
- This means it's fair to say

$$x_{ij} \approx \sum_{k=1}^{K} z_{ik} \varphi_{kj}$$

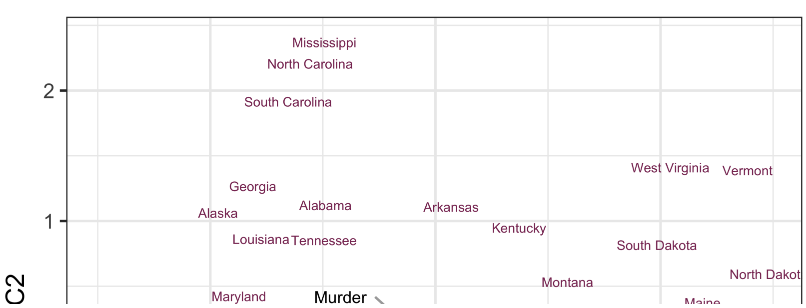
- Or, in matrix notation, $x_i \approx \Phi z_i$, where Φ concatenates the φ_k's vertically
    - x_i is a mixture of the components φ_k with weights z_{ik}.

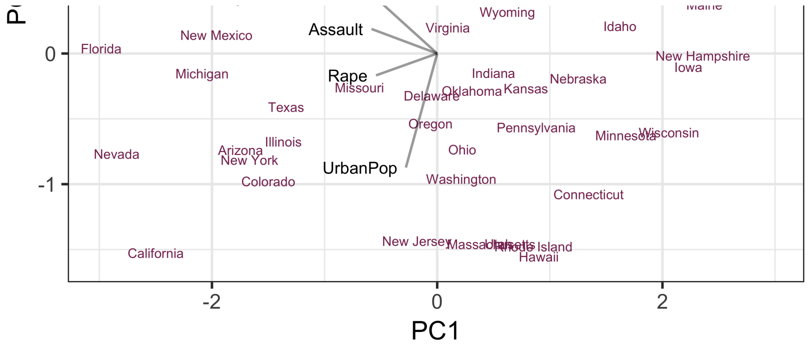


Example with two principal components

We'll practice reading PCA plots using `USArrests` --- a little dark but you will see more of such *breaking bad* themed examples in ISLR

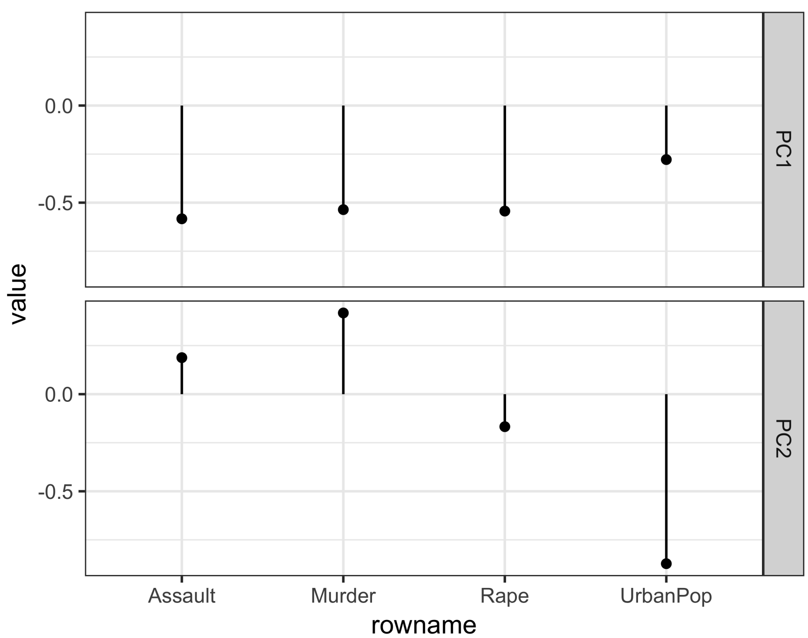
- z_i's give the states' coordinates
- Can interpret components by looking at how variables contribute. Variable j is plotted at $(\varphi_{1j}, \varphi_{2j})$.
    - E.g., the second PC mostly captures variation related to urban population





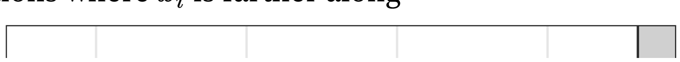
Biplots

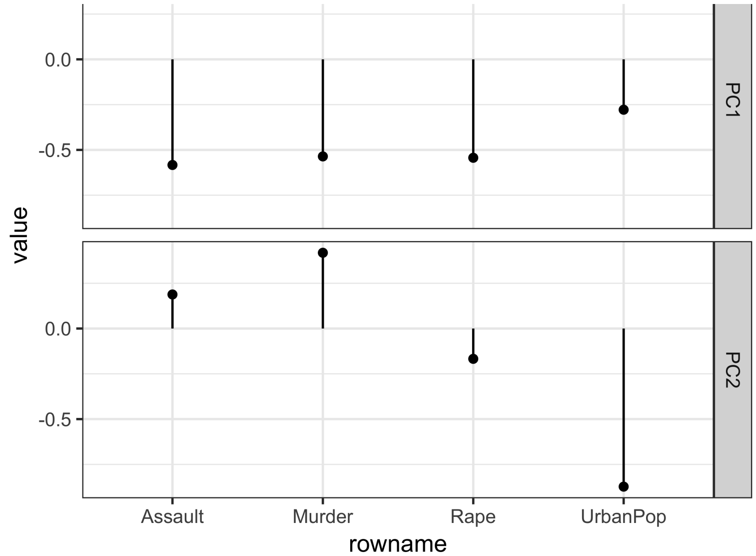
Arrows come from the φ_1 and φ_2. The (x, y)-coordinate of the arrows comes from viewing these PCs in 2D.



Biplots

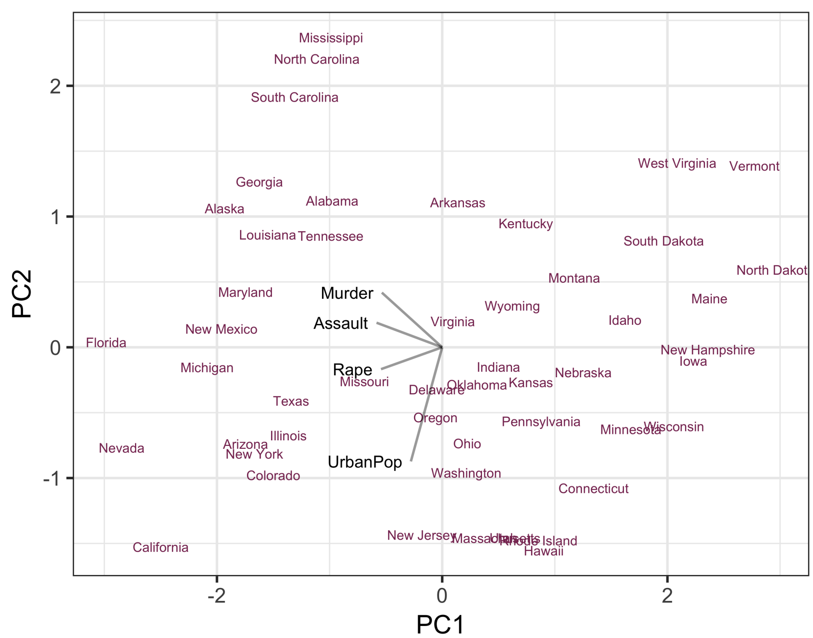
- The coordinate of x_i on the biplot is (z_{i1}, z_{i2})
- Since $x_i \approx z_{i1}\varphi_1 + z_{i2}\varphi_2$, they have large values for variables with large loadings in the coordinate directions where x_i is farther along





Biplots

- For example, California $\approx -2.5\varphi_1 - 1.52\varphi_2$
- Since φ_1 puts negative weight on the crimes, California has more than the average # of crimes $(- \times - = +)$
- Since φ_2 puts negative weight on urban population, California has larger than the average population

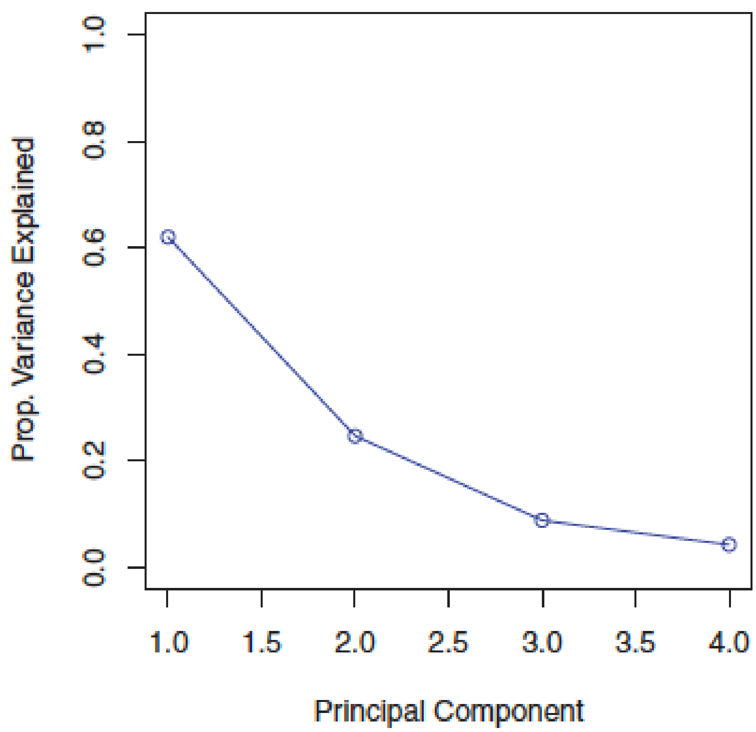


Explained Variation

- Amount explained by k^{th} component,

$$\frac{\sum_{i=1}^{n} \|z_{ki}\|^2}{\sum_{i=1}^{n} \|x_i\|^2}$$

- If no directions are preferred, get $\approx \frac{1}{p}\%$ everywhere
- How to compute?
    - Consider the covariance matrix of the data. Check total variance. Now look at the transformed covariance matrix and find the ratio of variances of each component.
- Exercise: What would the plot below look like if the data were shaped like...
    - a pancake (two long directions, one short one)
    - a cigar (one long direction, two short ones).



Things to watch out for

- Even though the method is easy to run, there are lots of potential issues,
    - Variables might be at different scales, and there might be ambiguity about whether to rescale them
    - The directions are only unique up to sign
- Choosing K is tricky (although might not be crucial)

Clustering

Idea: Partition the observations, so that those that are similar to each other appear together

Look for homogeneous subgroups in your heterogeneous data.

Formalization

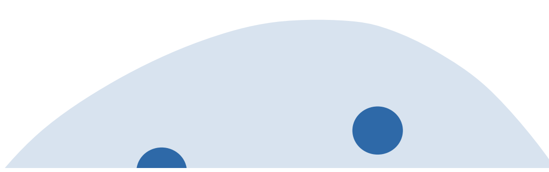
A partition $C_1, \dots, C_k$ is a collection of subsets satisfying,

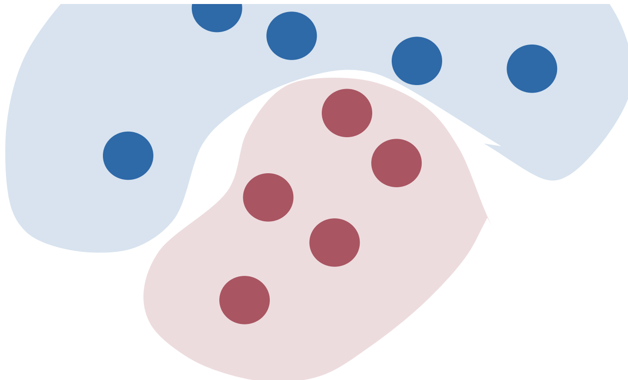
- Each sample is in a subset:

$$\cup_{k=1}^{K} C_k = \{1, \dots, n\}$$

- Subsets are disjoint: For any pair,

$$C_k \cap C_{k'} = \emptyset$$





Formalization

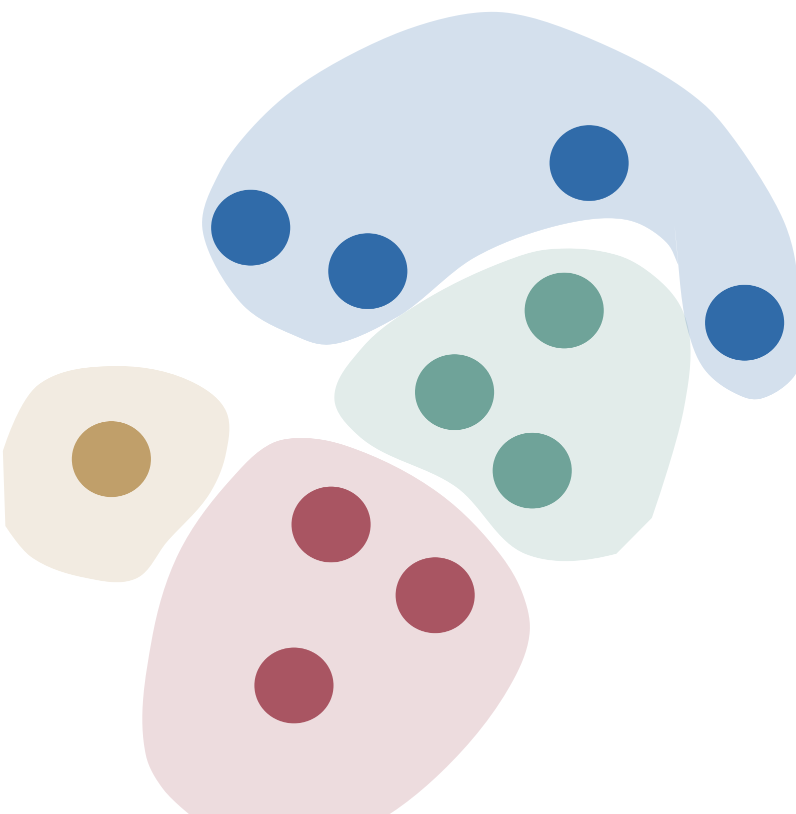
A partition $C_1, \ldots, C_k$ is a collection of subsets satisfying,

- Each sample is in a subset:

$$\cup_{k=1}^{K} C_k = \{1, \ldots, n\}$$

- Subsets are disjoint: For any pair,

$$C_k \cap C_{k'} = \emptyset$$



Formalization

A partition $C_1, \ldots, C_k$ is a collection of subsets satisfying,

- Each sample is in a subset:

$$\cup_{k=1}^{K} C_k = \{1, \ldots, n\}$$

- Subsets are disjoint: For any pair,

$$C_k \cap C_{k'} = \emptyset$$

Formalization

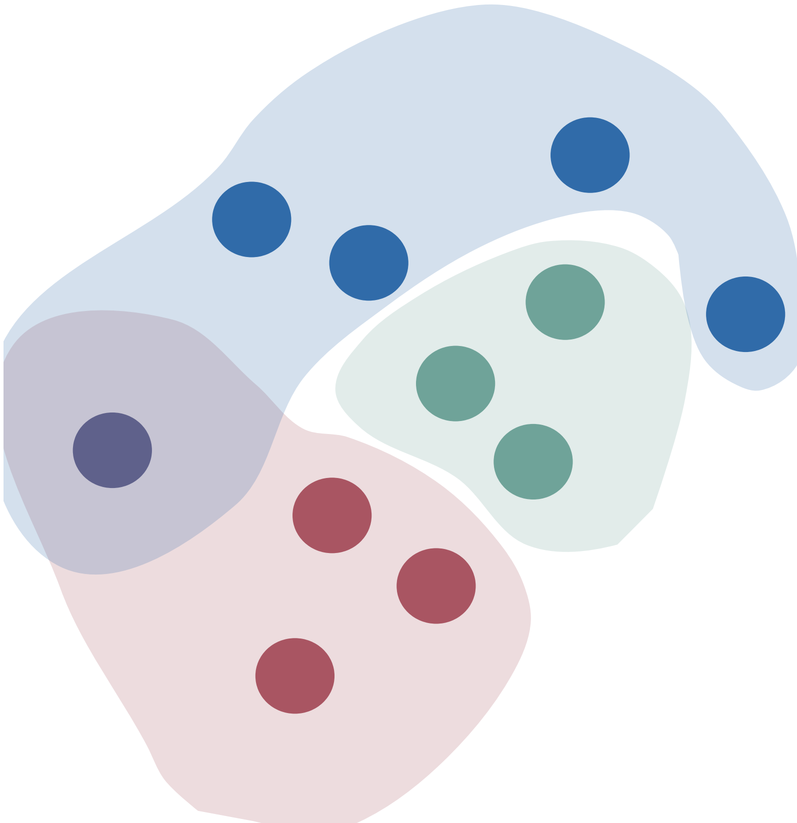
A partition $C_1, \ldots, C_k$ is a collection of subsets satisfying,

- Each sample is in a subset:

$$\cup_{k=1}^{K} C_k = \{1, \ldots, n\}$$

- Subsets are disjoint: For any pair,

$$C_k \cap C_{k'} = \emptyset$$

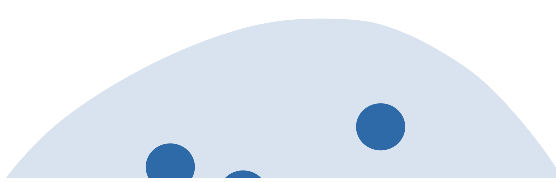


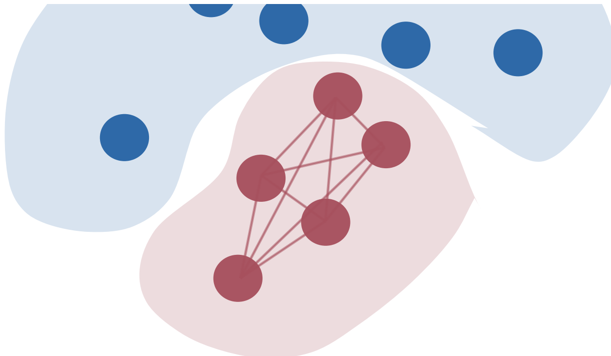
Formalization

We don't want just any partition, but the one that minimizes within group variation, which we'll call W.

$$\underset{C_k}{\text{minimize}} \sum_{k=1}^{K} W\left(C_k\right)$$

Usually, we use $W\left(C_k\right) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \|x_i - x_i'\|^2$.





Algorithm

This is a combinatorial optimization problem, and finding the global optimum is computationally challenging.

However the following algorithm usually finds good local optima,

1. Arbitrarily assign each x_i to one of the clusters, $C_1, \ldots, C_K$.
2. Iterate until convergence,

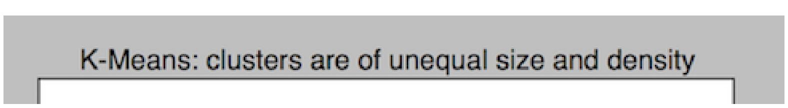
   a. Compute the mean $\bar{x}_k$ of the points in C_k.

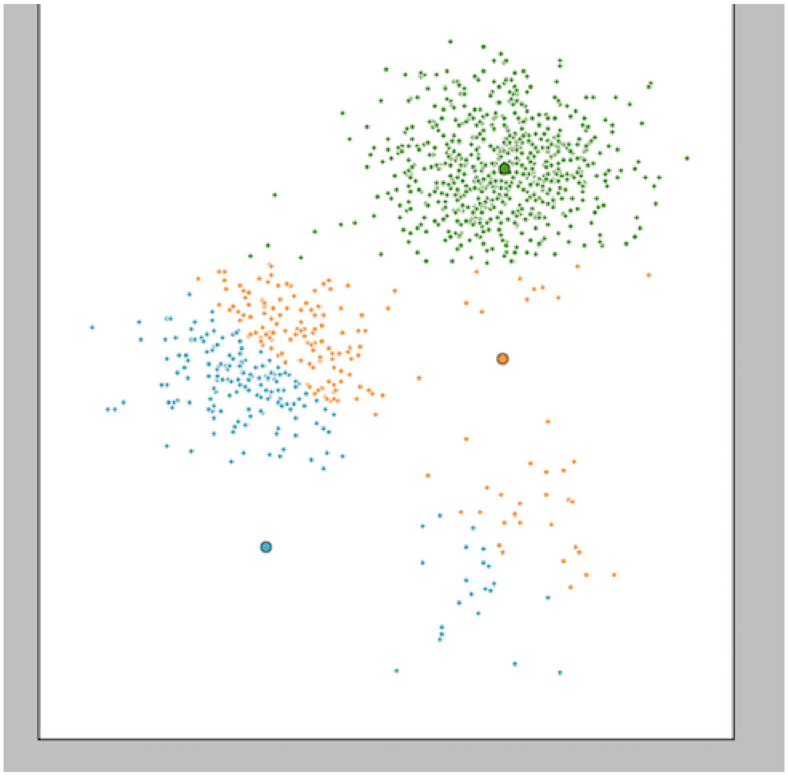
   b. Reassign the points x_i, so they are put in the cluster whose centroid they are closest to.

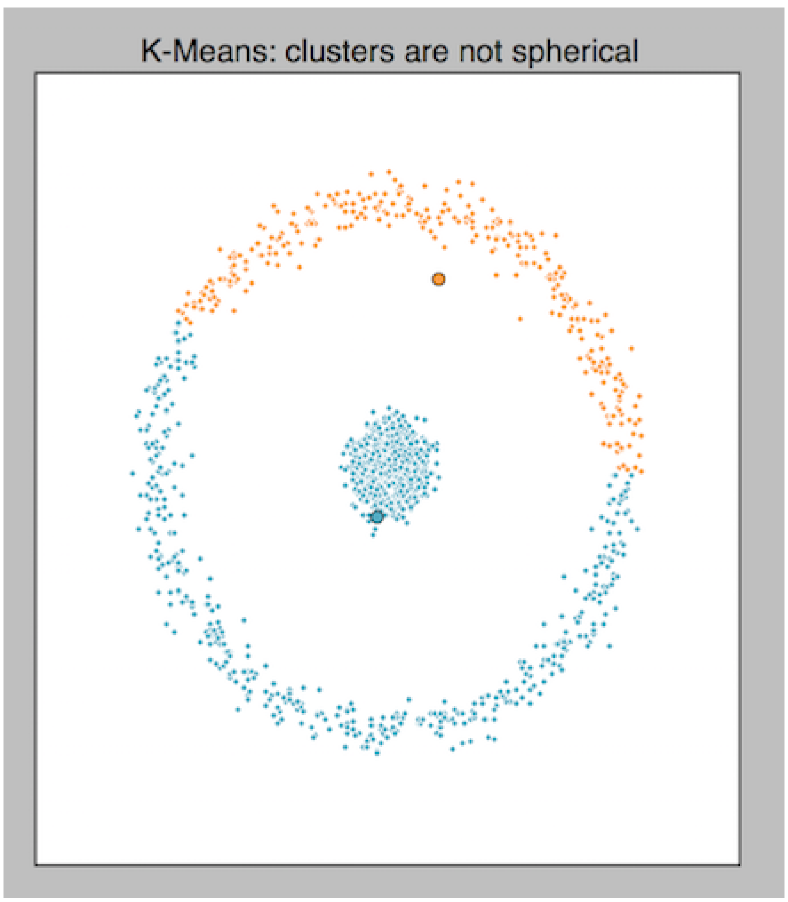
Here is a nice demo. The procedure reduces the criterion under study at each step, which means we will converge to a local optimum.

Limitations of K-means

K-means often does not do a good job in clustering when there are variations in density, nonspherical shapes of clusetrs and outliers.







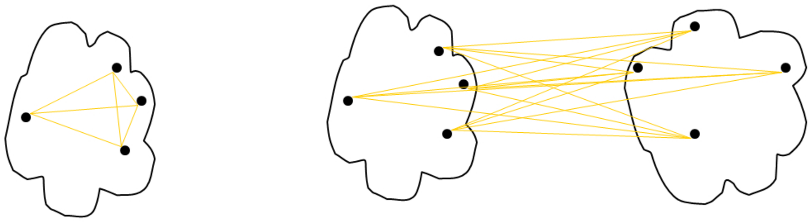
How to evaluate K-means results?

- Supervised evaluation: use a pre-classified dataset as a benchmark
    - Given the knowledge of the ground truth class assignments of the samples, use some metric, e.g., *homogeneity, completeness, v-score*, to evaluate the

homogeneity, completeness, v-score, to evaluate the goodness of the the clustering (higher score is better)

  - Python: from **sklearn** import **metrics**; *metrics.homogeneity_score(labels_true, labels_pred)*

- Unsupervised evaluation: minimize intra-cluser distance (maximize *cohesiveness*) and maximize inter-clusetr distance (*separation*)



cohesion separation

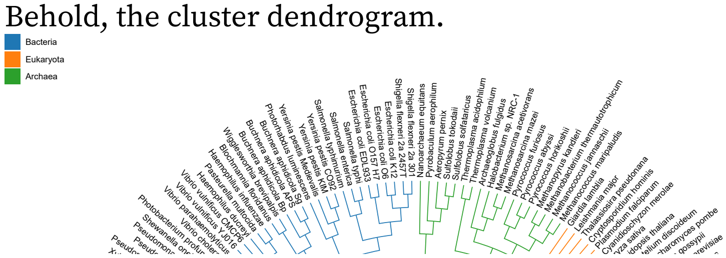
Details: Cluster Analysis Using K-means Explained

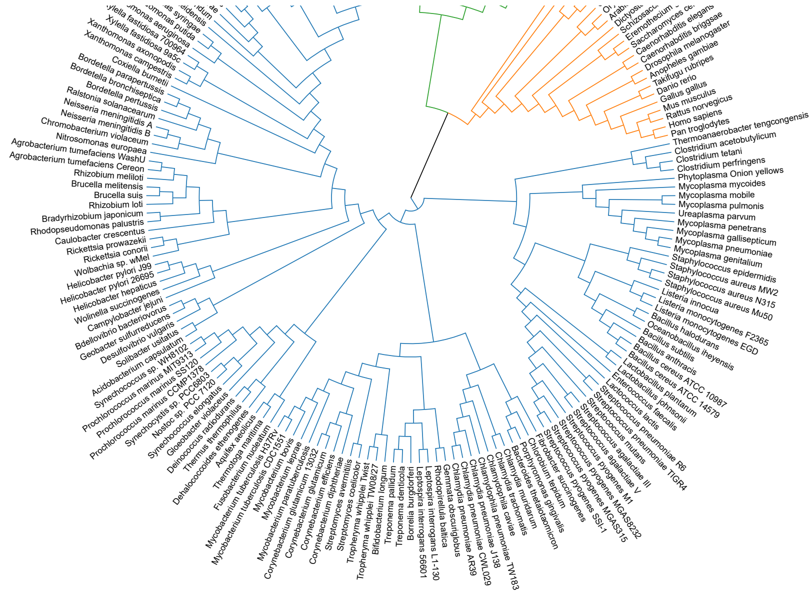
Hierarchical Clustering

- K controls the "magnification" at which we do the clustering.
- What if we could do the clustering at many different scales, all at once?

Hierarchical Clustering

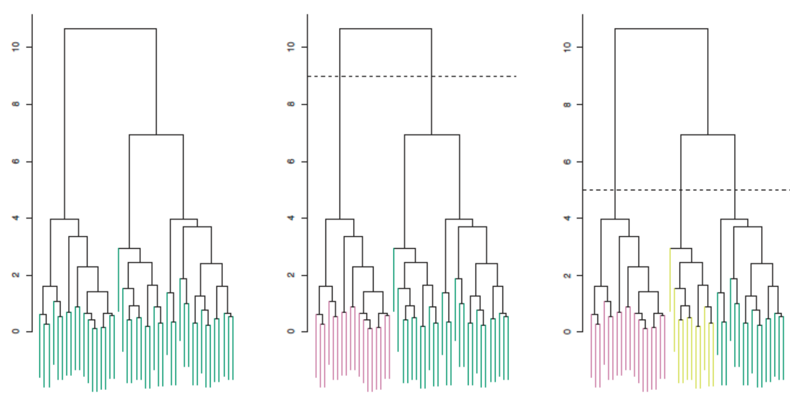
Behold, the cluster dendrogram.





Interpretation

- Samples which are similar to each other are put on the same subtree.
- Pairs of samples that are very similar to one another share very recent common ancestors
  - Beware: Samples can be close by at the leaves without being close in the subtree sense
- You can get a standard clustering by "cutting" tree at some horizontal level

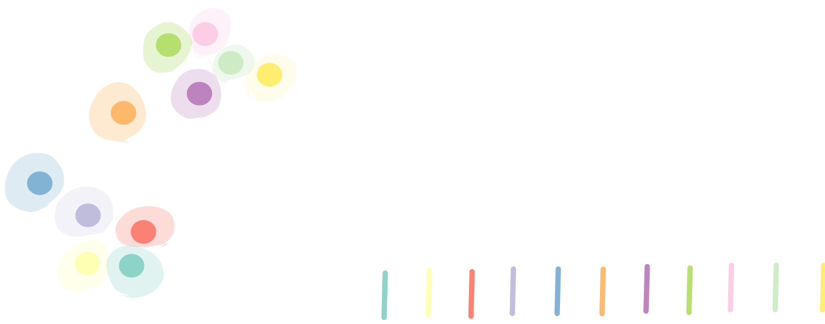


That's cool, how do I make it?

- These trees are informative. We'd like an automated procedure for creating them.
  - Agglomerative: *bottom-up*
  - Divisive: *top-down*

Agglomerative

a. Initialize: Associate each point with a cluster $C_i := \{x_i\}$
b. Iterate until only one cluster: Look at all pairs of clusters.
Merge the pair $C_k, C_{k'}$ which are the most similar.



That's cool, how do I make it?

- These trees are informative. We'd like an automated procedure for creating them.

Agglomerative

a. Initialize: Associate each point with a cluster $C_i := \{x_i\}$
b. Iterate until only one cluster: Look at all pairs of clusters.
Merge the pair $C_k, C_{k'}$ which are the most similar.

?

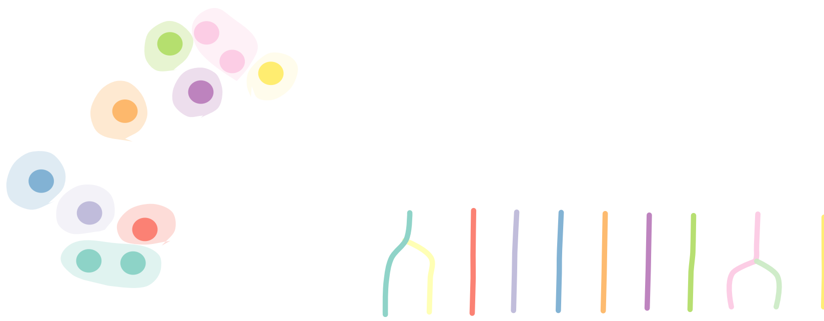
That's cool, how do I make it?

That's cool, how do I make it?

- These trees are informative. We'd like an automated procedure for creating them.

Agglomerative

a. Initialize: Associate each point with a cluster $C_i := \{x_i\}$
b. Iterate until only one cluster: Look at all pairs of clusters.
Merge the pair $C_k, C_{k'}$ which are the most similar.

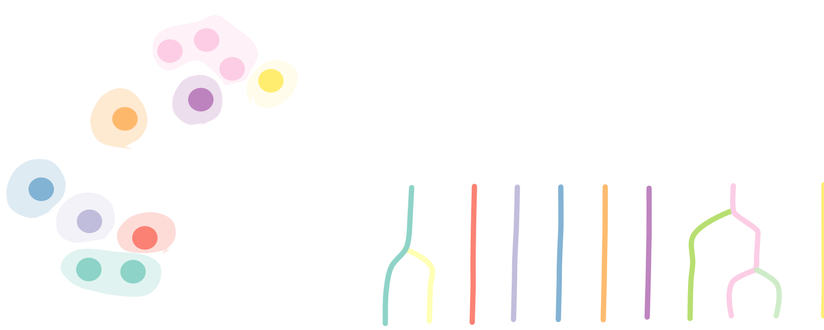


That's cool, how do I make it?

- These trees are informative. We'd like an automated procedure for creating them.

Agglomerative

a. Initialize: Associate each point with a cluster $C_i := \{x_i\}$
b. Iterate until only one cluster: Look at all pairs of clusters.
Merge the pair $C_k, C_{k'}$ which are the most similar.

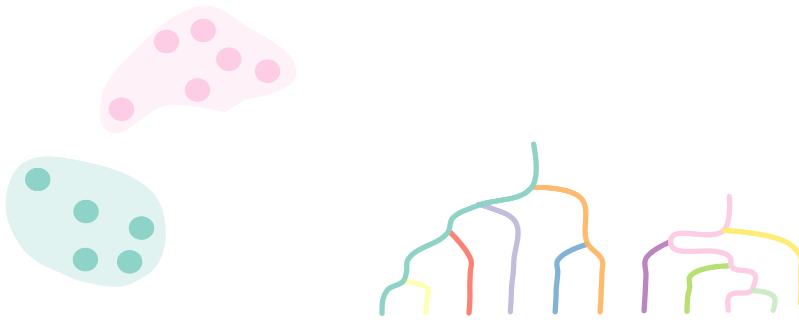


That's cool, how do I make it?

- These trees are informative. We'd like an automated procedure for creating them.

Agglomerative

a. Initialize: Associate each point with a cluster $C_i := \{x_i\}$
b. Iterate until only one cluster: Look at all pairs of clusters.
Merge the pair $C_k, C_{k'}$ which are the most similar.



That's cool, how do I make it?

- These trees are informative. We'd like an automated procedure for creating them.

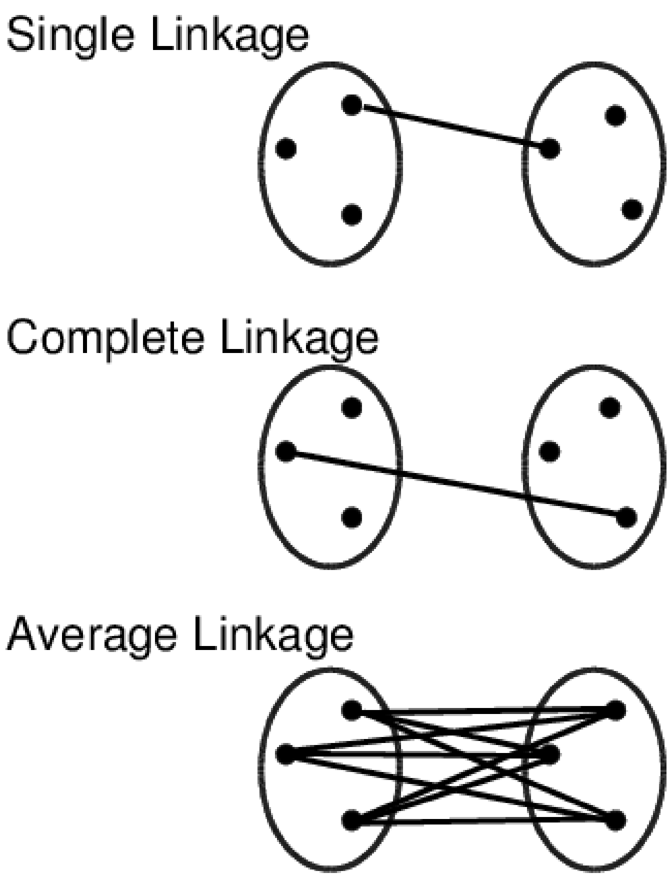
Agglomerative

a. Initialize: Associate each point with a cluster $C_i := \{x_i\}$
b. Iterate until only one cluster: Look at all pairs of clusters.
Merge the pair $C_k, C_{k'}$ which are the most similar.

Similarity between clusters

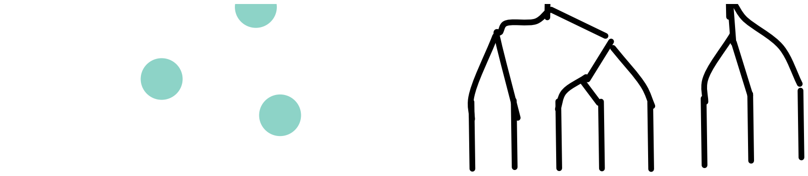
- The height on the dendrogram gives the similarity between descendants

- But what's a good distance between pairs of sets?
  - Single: Minimum distance between any pair of points
  - Complete: Maximum distance between any pair of points
  - Average: Average distance over all pairs of points



Similarity between clusters

- The distance between the clusters at any iteration can be visualized on the tree
- Merges lower on the tree --> Smaller intercluster distance



Similarity between clusters

- The distance between the clusters at any iteration can be visualized on the tree
- Merges lower on the tree --> Smaller intercluster distance

Comparison between K-means and hierarchical clustering

1. K-means can handle big data more efficiently compared to hierarchical method.

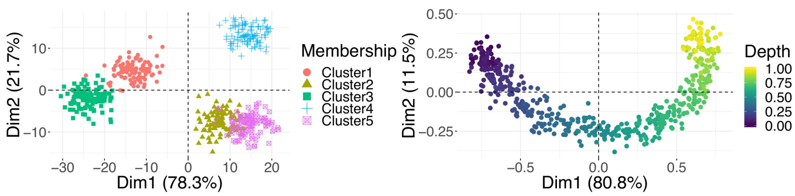
2. K-means clustering requires prior knowledge of K. But, you can stop at whatever number of clusters you find appropriate in hierarchical clustering by interpreting the dendrogram.

3. K Means is found to work well when the shape of the clusters is hyper spherical whereas hierarchical clustering works better in case of non-spherical data distribution.

4. Results are reproducible in Hierarchical clustering, whereas they might differ in K-means due to random choice of initial clusters.

Objective of dimesionality reduction

- Compress data while preserving most of the meaningful information
- What are the latent patterns in the data?
    - Discrete change in attributes: clustering
    - Gradual shift in pattern: gradients



(a) Clusters (b) Gradient

Ref: Ten quick tips for effective dimensionality reduction

What to watch out for

- The choice of distance is crucial.
    - Numeric data: e.g., Euclidean, Squared Euclidean, Manhattan, Mahalanobis, Minkowski
    - Text or other non-numeric data: Hamming, Levenshtein
- Don't try to find a "perfect" distance. Try many, and assess the sensitivity of your findings.

Some final remarks

Unsupervised learning is useful for more than "exploring your data"

We are both "summarizing" data and *creating more of it*

- The centroids in a clustering are new data, which help us understand the original data

These methods are critical submodules in removal of batch effects, interpretation of deep networks, anomaly detection, regularized regression and much much more.

```
import {chart} from @mbostock/tree-of-life
```

```
keypresscontrol = undefined
```

```
slide = ƒ()
```

```
hl = ▸ Object {highlight: ƒ(e, t, a, c), highlightAuto: ƒ(e, t)
```

```
<style>
```

```
import {mtex_block} from @krisrs1128/function-fitting
```

```
import {mtex} from @krisrs1128/function-fitting
```

Extra Material

- The relationship between PCA and the covariance matrix

$$
\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}\left(\varphi_1^T x_i\right)^2 &= \frac{1}{n}\sum_{i=1}^{n}\left(\varphi_1^T x_i\right)\left(x_i^T \varphi_1\right) \\
&= \varphi_1^T\left(\frac{1}{n}\sum_{i=1}^{n} x_i x_i^T\right)\varphi_1 \\
&= \varphi_1^T \hat{\Sigma} \varphi_1
\end{aligned}
$$

?