Lecture 19: Unsupervised learning: dimension reduction

BIOS635

03/31/2020

Some slides taken/modified from other people's lectures online.

Unsupervised learning

Unsupervised vs Supervised Learning:

- Most of this course focuses on *supervised learning* methods such as regression and classification.
- In that setting we observe both a set of features $X_1, X_2, \ldots, X_p$ for each object, as well as a response or outcome variable Y. The goal is then to predict Y using $X_1, X_2, \ldots, X_p$.
- Here we instead focus on *unsupervised learning*, we where observe only the features $X_1, X_2, \ldots, X_p$. We are not interested in prediction, because we do not have an associated response variable Y.

The goals of unsupervised learning

- The goal is to discover interesting things about the measurements: is there an informative way to visualize the data? Can we discover subgroups among the variables or among the observations?
- We discuss two methods:
  - *principal components analysis*, a tool used for data visualization or data pre-processing before supervised techniques are applied, and
  - *clustering*, a broad class of methods for discovering unknown subgroups in data.

The challenge of unsupervised learning

- Unsupervised learning is more subjective than supervised learning, as there is no simple goal for the analysis, such as prediction of a response.
- But techniques for unsupervised learning are of growing importance in a number of fields:
  - subgroups of breast cancer patients grouped by their gene expression measurements,
  - groups of shoppers characterized by their browsing and purchase histories,
  - movies grouped by the ratings assigned by movie viewers.

Principal component analysis

- PCA produces a low-dimensional representation of a dataset. It finds a sequence of linear combinations of the variables that have maximal variance, and are mutually uncorrelated.
- Apart from producing derived variables for use in supervised learning problems, PCA also serves as a tool for data visualization.

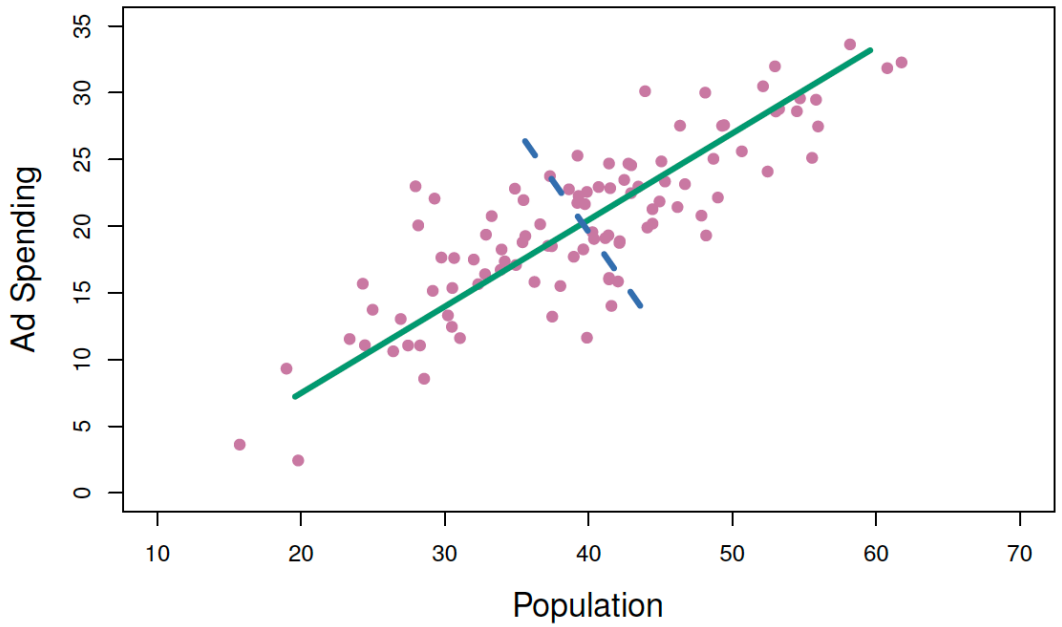
PCA details

- The *first principal component* of a set of features $X_1, X_2, \ldots, X_p$ is the normalized linear combination of the features

$$Z_1 = \phi_{11} X_1 + \phi_{21} X_2 + \ldots + \phi_{p1} X_p$$

  that has the largest variance. By *normalized*, we mean that $\sum_{j=1}^{p} \phi_{j1}^2 = 1$.
- We refer to the elements $\phi_{11}, \ldots, \phi_{p1}$ as the loadings of the first principal component; together, the loadings make up the principal component loading vector, $\phi_1 = (\phi_{11}\ \phi_{21}\ \ldots\ \phi_{p1})^T$.
- We constrain the loadings so that their sum of squares is equal to one, since otherwise setting these elements to be arbitrarily large in absolute value could result in an arbitrarily large variance.

PCA: example



The population size (pop) and ad spending (ad) for 100 different cities are shown as purple circles. The green solid line indicates the first principal component direction, and the blue dashed line indicates the second principal component direction.

Computation of principal components

- Suppose we have a $n \times p$ data set $\mathbf{X}$. Since we are only interested in variance, we assume that each of the variables in $\mathbf{X}$ has been centered to have mean zero (that is, the column means of $\mathbf{X}$ are zero).
- We then look for the linear combination of the sample feature values of the form

$$z_{i1} = \phi_{11} x_{i1} + \phi_{21} x_{i2} + \ldots + \phi_{p1} x_{ip} \qquad (1)$$

  for $i = 1, \ldots, n$ that has largest sample variance, subject to the constraint that $\sum_{j=1}^{p} \phi_{j1}^2 = 1$.
- Since each of the x_{ij} has mean zero, then so does z_{i1} (for any values of ϕ_{j1}). Hence the sample variance of the z_{i1} can be written as $\frac{1}{n} \sum_{i=1}^{n} z_{i1}^2$.

Computation of principal components, cont'd

- Plugging in (1) the first principal component loading vector solves the optimization problem

$$\underset{\phi_{11},\ldots,\phi_{p1}}{\text{maximize}} \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{p}\phi_{j1}x_{ij}\right)^2 \text{ subject to } \sum_{j=1}^{p}\phi_{j1}^2 = 1.$$

- This problem can be solved via a singular-value decomposition of the matrix $\mathbf{X}$, a standard technique in linear algebra.
- We refer to Z_1 as the first principal component, with realized values $z_{11}, \ldots, z_{n1}$

Geometry of PCA

- The loading vector ϕ_1 with elements $\phi_{11}, \phi_{21}, \ldots, \phi_{p1}$ defines a direction in feature space along which the data vary the most.
- If we project the n data points $x_1, \ldots, x_n$ onto this direction, the projected values are the principal component scores $z_{11}, \ldots, z_{n1}$ themselves.

Further principal components

- The second principal component is the linear combination of $X_1, \ldots, X_p$ that has maximal variance among all linear combinations that are *uncorrelated* with Z_1.
- The second principal component scores $z_{12}, z_{22}, \ldots, z_{n2}$ take the form

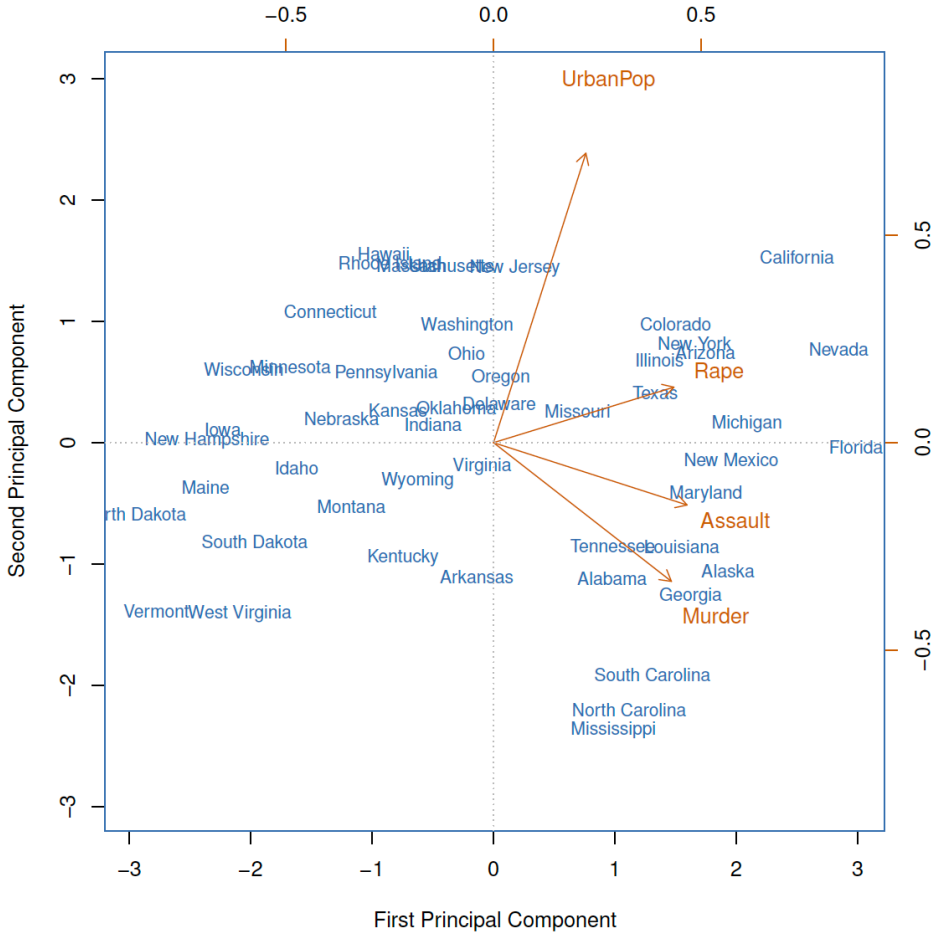
$$z_{i2} = \phi_{12} x_{i1} + \phi_{22} x_{i2} + \ldots + \phi_{p2} x_{ip},$$

where ϕ_2 is the second principal component loading vector, with elements $\phi_{12}, \phi_{22}, \ldots, \phi_{p2}$.

Further principal components, cont'd

- It turns out that constraining Z_2 to be uncorrelated with Z_1 is equivalent to constraining the direction ϕ_2 to be orthogonal (perpendicular) to the direction ϕ_1. And so on.
- The principal component directions ϕ_1, ϕ_2, ϕ_3, ... are the ordered sequence of right singular vectors of the matrix $\mathbf{X}$, and the variances of the components are $\frac{1}{n}$ times the squares of the singular values. There are at most $\min(n-1, p)$ principal components.

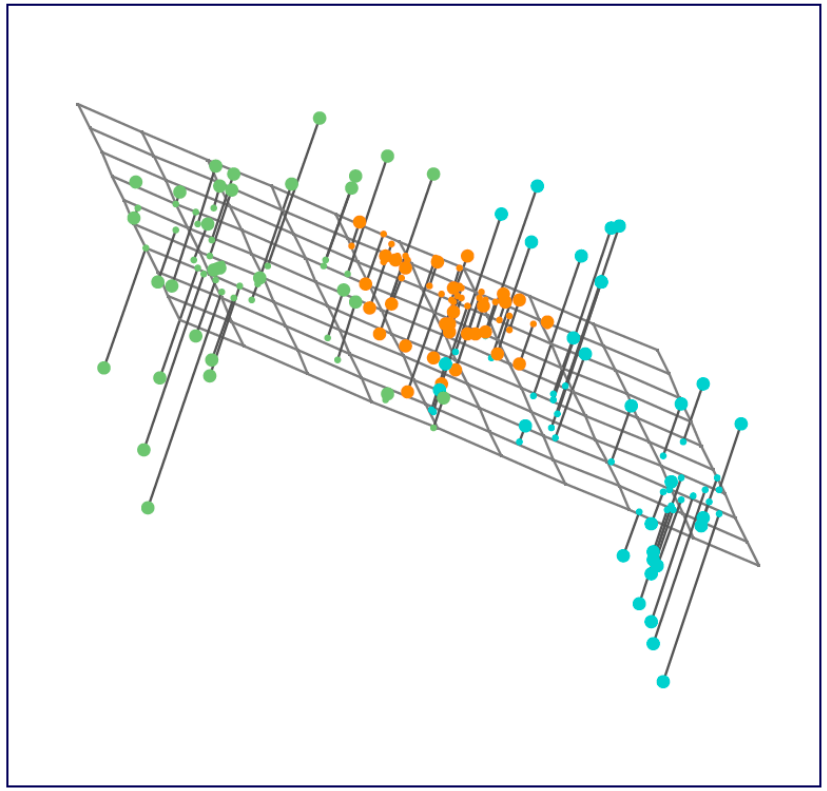
USAarrests data: PCA biplot

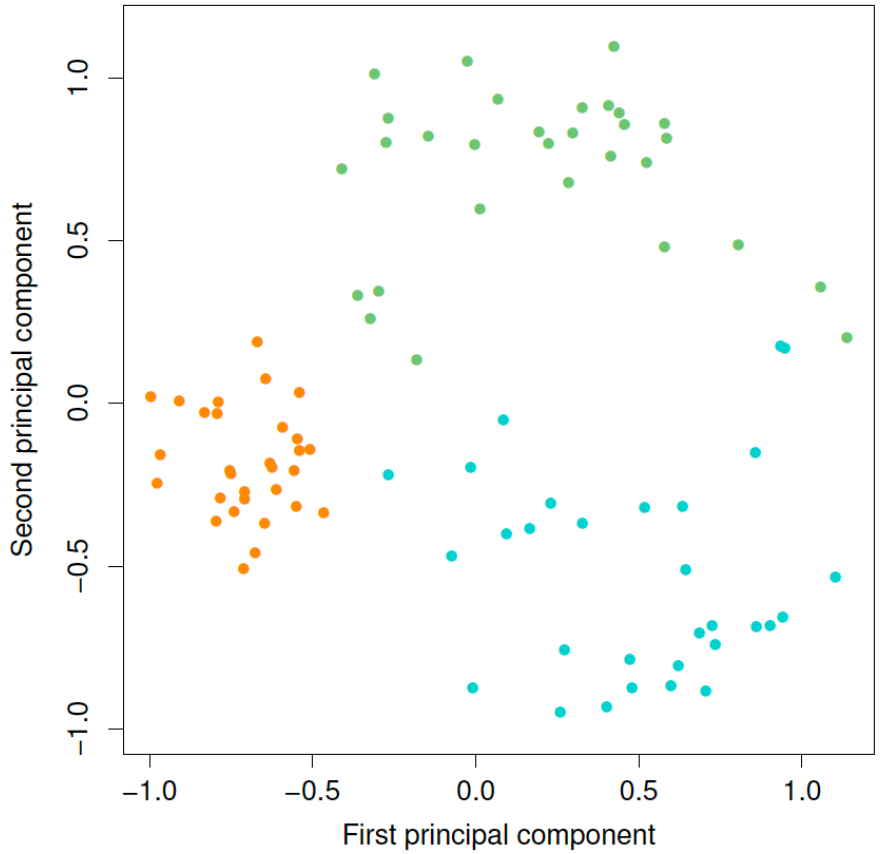


The first two principal components for the USArrests data.

- The blue state names represent the scores for the first two principal components.
- The orange arrows indicate the first two principal component loading vectors (with axes on the top and right). For example, the loading for `Rape` on the first component is 0.54, and its loading on the second principal component 0.17 [the word `Rape` is centered at the point $(0.54, 0.17)$].
- This figure is known as a *biplot*, because it displays both the principal component scores and the principal component loadings.

Another interpretation of principal component



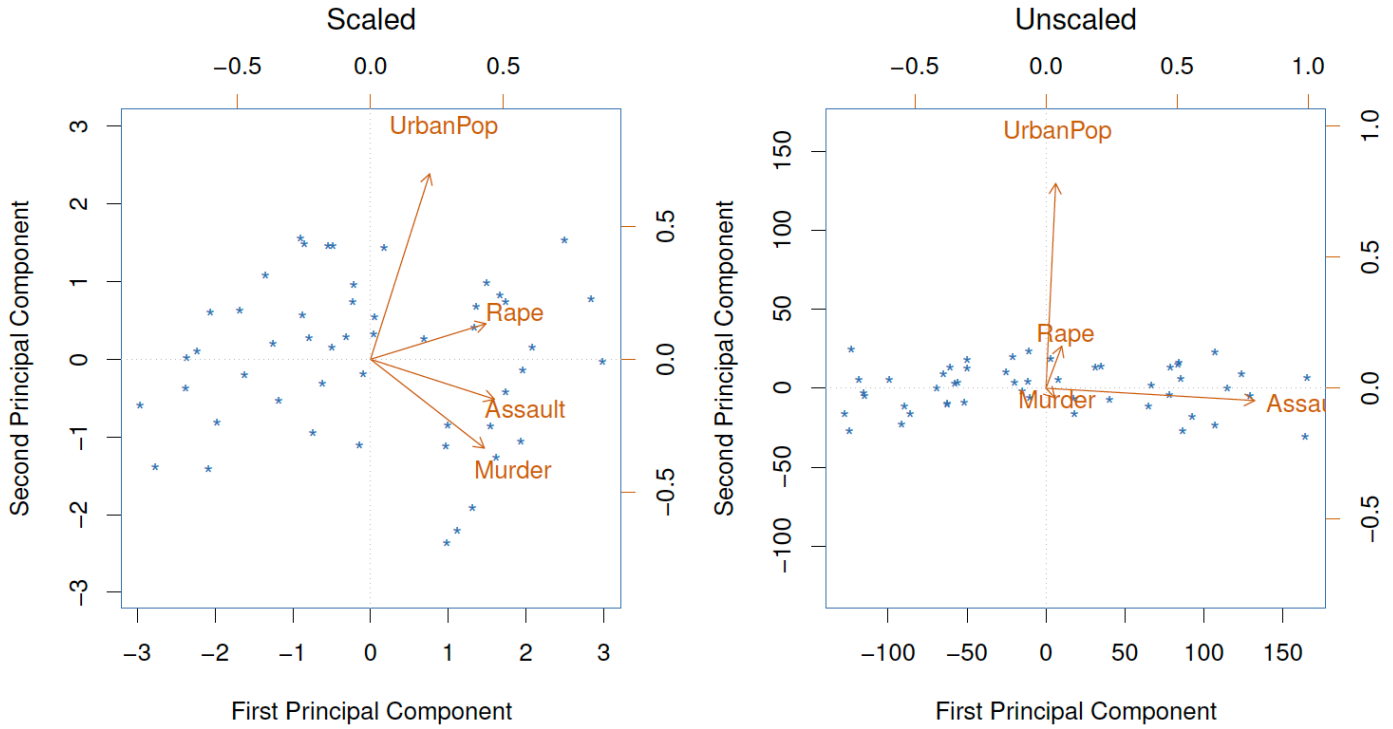


PCA finds the hyperplane closest to the observations

- The first principal component loading vector has a very special property: it defines the line in p-dimensional space that is *closest* to the n observations (using average squared Euclidean distance as a measure of closeness)
- The notion of principal components as the dimensions that are closest to the n observations extends beyond just the first principal component.
- For instance, the first two principal components of a data set span the plane that is closest to the n observations, in terms of average squared Euclidean distance.

Scaling of the variables matters

- If the variables are in different units, scaling each to have standard deviation equal to one is recommended.
- If they are in the same units, you might or might not scale the variables.



Proportion variance explained

- To understand the strength of each component, we are interested in knowing the proportion of variance explained (PVE) by each one.
- The *total variance* present in a data set (assuming that the variables have been centered to have mean zero) is defined as

$$\sum_{j=1}^{p} \mathrm{Var}(X_j) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2,$$

  and the variance explained by the mth principal component is

$$\mathrm{Var}(Z_m) = \frac{1}{n} \sum_{i=1}^{n} z_{im}^2.$$

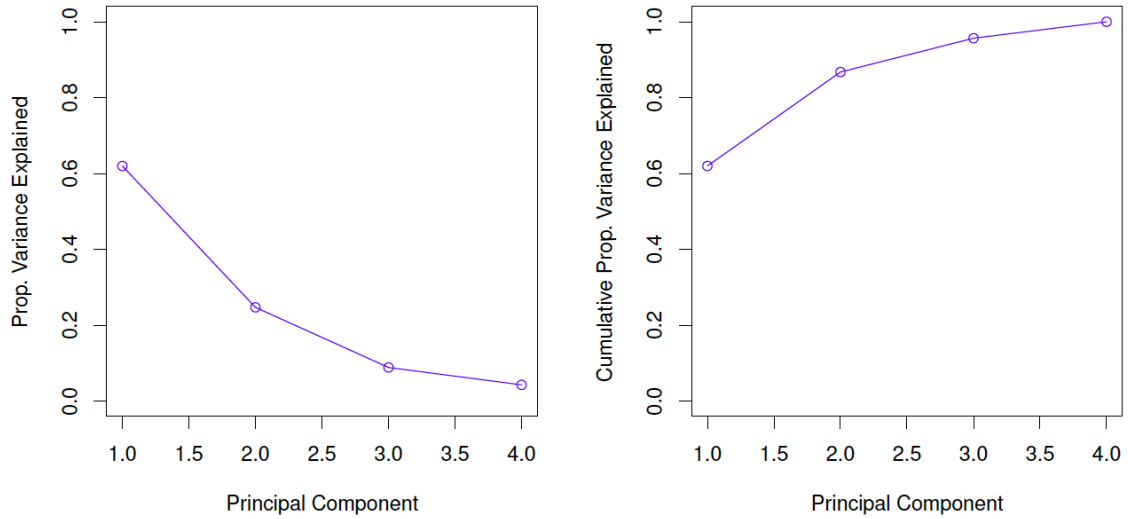
- It can be shown that $\sum_{j=1}^{p} \mathrm{Var}(X_j) = \sum_{m=1}^{M} \mathrm{Var}(Z_m)$, with $M = \min(n-1, p)$.

Proportion of variance explained, cont'd

- Therefore, the PVE of the mth principal component is given by the positive quantity between 0 and 1

$$\frac{\sum_{i=1}^{n} z_{im}^2}{\sum_{j=1}^{p} \sum_{i=1}^{n} x_{ij}^2}.$$

- The PVEs sum to one. We sometimes display the cumulative PVEs.



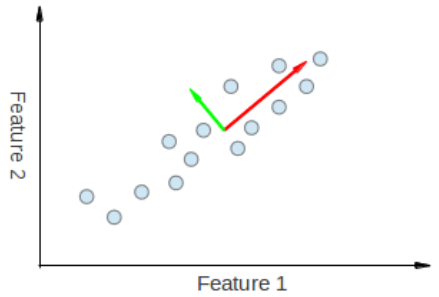
How many principal components?

If we use principal components as a summary of our data, how many components are sufficient?

- No simple answer to this question, as cross-validation is not available for this purpose.
  - *Why not?*
  - When could we use cross-validation to select the number of components?
- the "scree plot" on the previous slide can be used as a guide: we look for an "elbow".

PCA, recap

- PCA is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components (or PC scores)



First principal component: $\mathbf{Z}_1 = \phi_{11}\mathbf{X}_1 + \phi_{21}\mathbf{X}_2 + ... + \phi_{p1}\mathbf{X}_p$

PC loading vector: $\boldsymbol{\phi}_1 = \{\phi_{11}, \phi_{21}, ..., \phi_{p1}\}^T$, $\sum_{j=1}^{p} \phi_{j1}^2 = 1$

$\hat{\boldsymbol{\phi}}_1 = \underset{||\boldsymbol{\phi}_1||=1}{\text{argmax}}\{\boldsymbol{\phi}_1^T \mathbf{X}^T \mathbf{X} \boldsymbol{\phi}_1\}$ maximizes the variance of $\mathbf{Z}_1$.

$\boldsymbol{\phi}_1$ is the eigenvector corresponding to the largest eigenvalue of $\mathbf{X}^T\mathbf{X}$.

$\mathbf{Z}_2$ is restrained to be uncorrelated with $\mathbf{Z}_1$. Equivalently, $\boldsymbol{\phi}_2$ is orthogonal to $\boldsymbol{\phi}_1$.

Sparse PCA

- Disadvantage of ordinary PCA: PCs are usually linear combinations of all input variables. Sparse PCA overcomes this by finding linear combinations that contain just a few input variables, i.e., results in sparse loadings.

First principal component: $\mathbf{Z}_1 = \phi_{11}\mathbf{X}_1 + \phi_{21}\mathbf{X}_2 + ... + \phi_{p1}\mathbf{X}_p$

PC loading vector: $\boldsymbol{\phi}_1 = \{\phi_{11}, \phi_{21}, ..., \phi_{p1}\}^T$, $\sum_{j=1}^{p} \phi_{j1}^2 = 1$

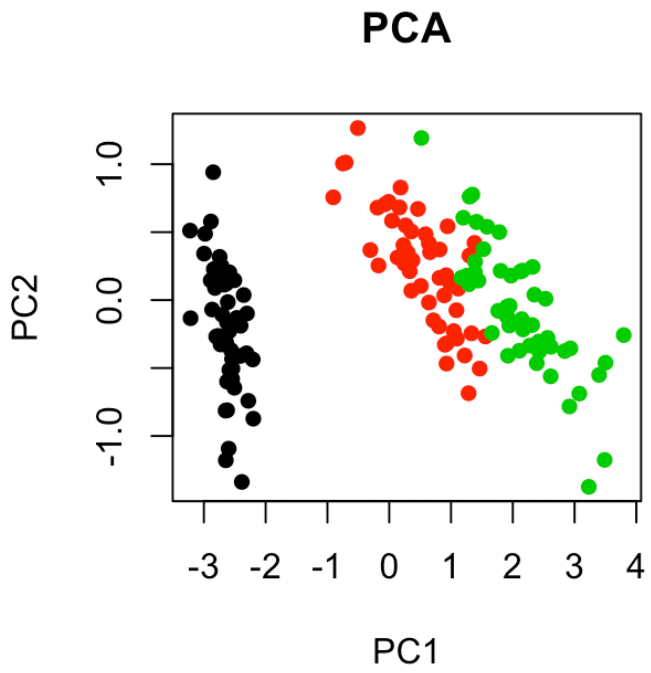
$\hat{\boldsymbol{\phi}}_1 = \underset{||\boldsymbol{\phi}_1||=1, ||\boldsymbol{\phi}_1||_0 \leq k}{\text{argmax}} \left\{\boldsymbol{\phi}_1^T \mathbf{X}^T \mathbf{X} \boldsymbol{\phi}_1\right\}$ maximizes the variance of $\mathbf{Z}_1$.

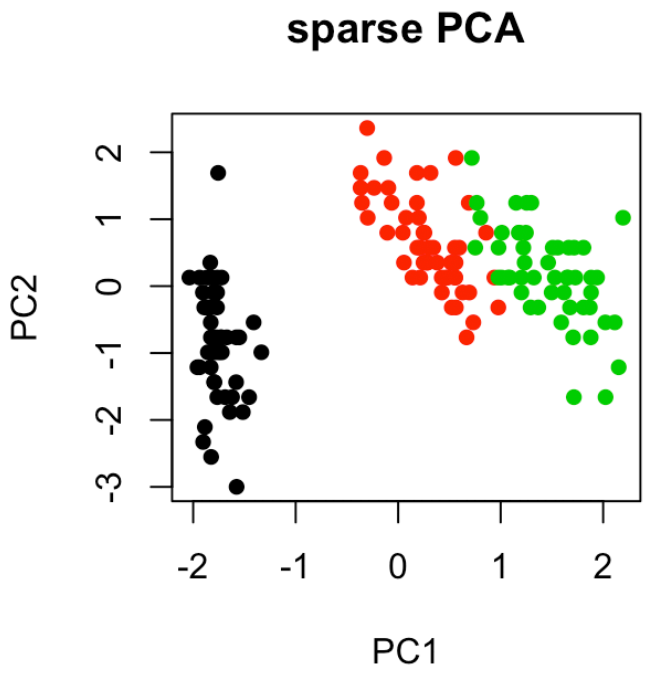
If $k = p$, this reduces to ordinary PCA.

PCA v.s. sparse PCA

```
> pca1$rotation
                    PC1         PC2         PC3        PC4
Sepal.Length  0.36138659 -0.65658877  0.58202985  0.3154872
Sepal.Width  -0.08452251 -0.73016143 -0.59791083 -0.3197231
Petal.Length  0.85667061  0.17337266 -0.07623608 -0.4798390
Petal.Width   0.35828920  0.07548102 -0.54583143  0.7536574
```

```
> spca$loadings
             [,1]       [,2]      [,3] [,4]
[1,] 0.00632416  0.0000000 0.9725313    0
[2,] 0.00000000 -0.9737092 0.0000000    0
[3,] 0.55960678  0.0000000 0.0000000    0
[4,] 0.81983460  0.0000000 0.0000000    0
```





Kernel PCA

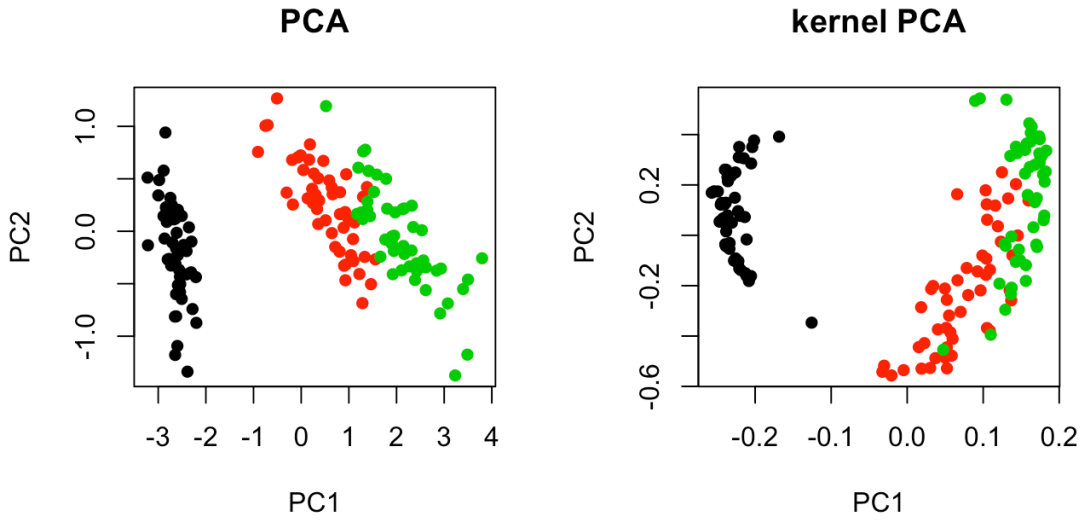
- Map each data point to nonlinear feature space
- Extract PC in the transformed space, which is non-linear in the original data space.

Need to center in the feature space $\tilde{\Phi}(\mathbf{x}_i) = \Phi(\mathbf{x}_i) - \frac{1}{n}\sum_{k=1}^{n}\Phi(\mathbf{x}_k)$.

Construct the centered kernel matrix: $\tilde{\mathbf{K}} = \mathbf{K} - \mathbf{1}_{1/n}\mathbf{K} - \mathbf{K}\mathbf{1}_{1/n} + \mathbf{1}_{1/n}\mathbf{K}\mathbf{1}_{1/n}$.

Solve an eigenvalue problem: $\tilde{\mathbf{K}}\boldsymbol{\alpha}_i = \lambda_i\boldsymbol{\alpha}_i$.

For any data point, we can represent it as $y_j = \sum_{i=1}^{n}\alpha_{ij}\tilde{\mathrm{K}}(\mathbf{x}, \mathbf{x}_i), j = 1, ..., d$.



tSNE & UMAP

- tSNE: t-Distributed Stochastic Neighbor Embedding

  Paper: https://lvdmaaten.github.io/publications/papers/JMLR_2008.pdf
  Presentation: https://www.youtube.com/watch?v=RJVL80Gg3lA

- UMAP: Uniform Manifold Approximation and Projection

  Paper: https://arxiv.org/pdf/1802.03426.pdf
  Presentation: https://www.youtube.com/watch?v=nq6iPZVUxZU

tSNE: t-Distributed Stochastic Neighbor Embedding

- Constructs probability distribution over pairs of high-dimensional objects in such a way that similar objects have a high probability of being picked whilst dissimilar points have an extremely small probability of being picked.

$X_1, \ldots, X_N$ are N high-dimensional objects

$$P_{j|i} = \frac{\exp(-||X_i - X_j||^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-||X_i - X_k||^2/2\sigma_i^2)}$$

Conditional probability that X_i would pick X_j as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centered at X_i.

Bandwidth of Gaussian kernels σ_i: smaller σ_i for denser part of the data space.

$$P_{ij} = \frac{P_{j|i} + P_{i|j}}{2}$$

tSNE: t-Distributed Stochastic Neighbor Embedding

- Defines a similar probability distribution over the points in the low-dimensional map, and it minimizes the KL divergence between the two distributions with respect to the locations of the points in the map.

Learn d-dimensional map $Y_1, \ldots, Y_N$, $Y_i \in \mathbb{R}^d$ that reflects the similarity P_{ij} as well as possible.

$$q_{ij} = \frac{(1 + ||Y_i - Y_j||^2)^{-1}}{\sum_{k \neq l}(1 + ||Y_k - Y_l||^2)^{-1}}$$

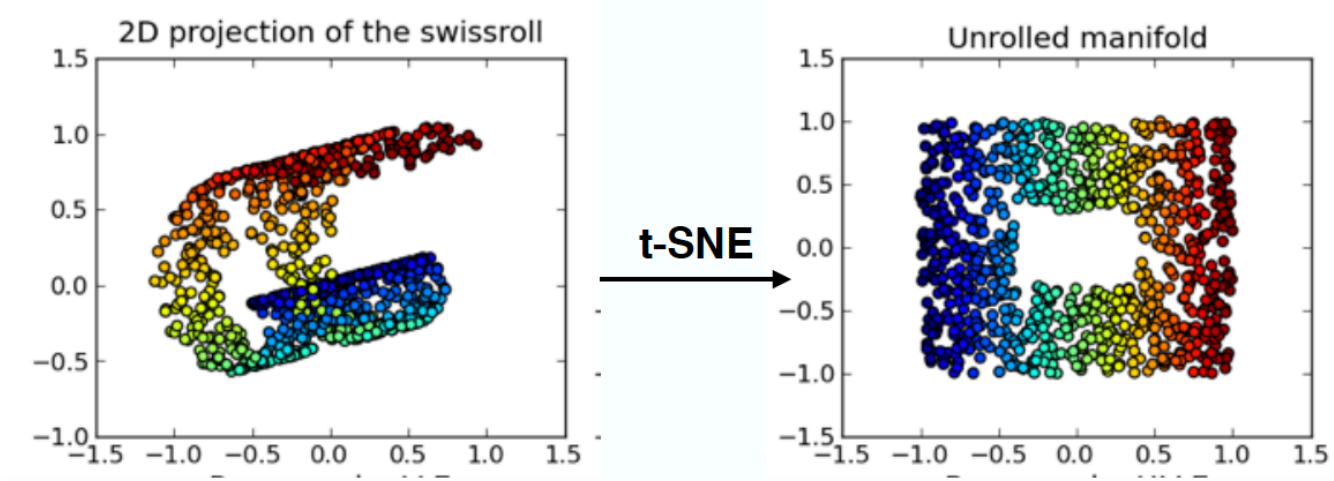
Heavy-tail t-distribution is used to measure similarities between low-dimensional points in order to allow dissimilar objects to be modeled far apart in the map.

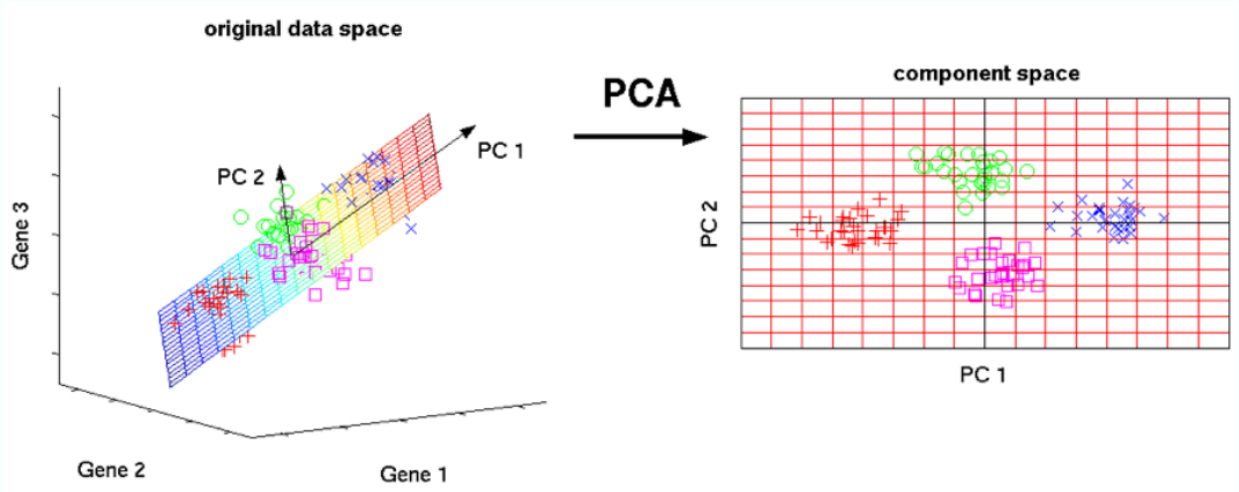
Minimize Kullback-Leibler divergence to get Y_i

$$KL(P||Q) = \sum_{i \neq j} P_{ij} log \frac{P_{ij}}{Q_{ij}}$$

The result of this optimization is a low-dimensional map that reflects the similarities between the high-dimensional input.

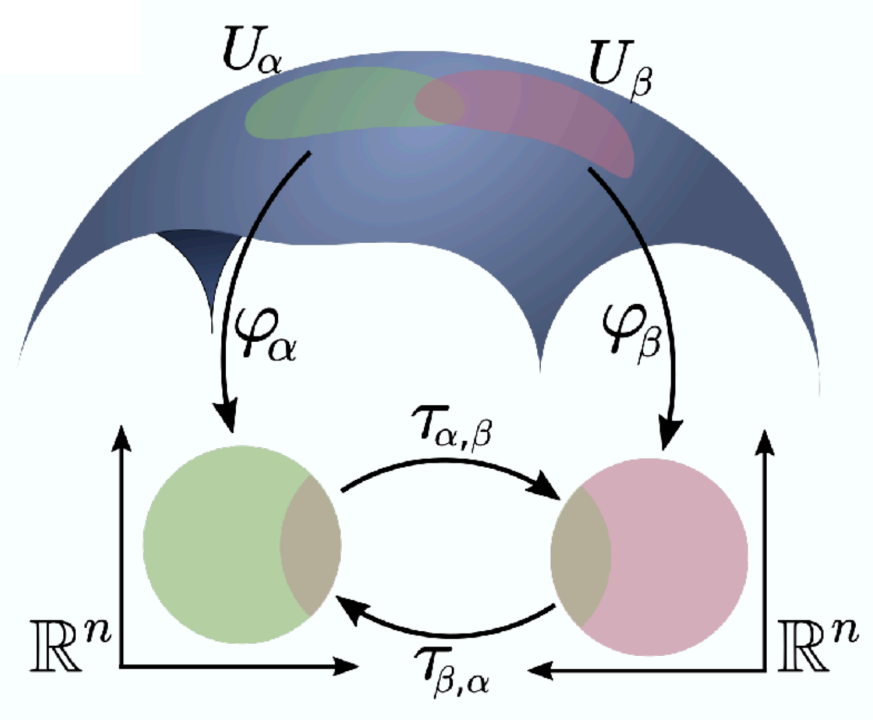
tSNE v.s. PCA


2D projection of the swissroll
Unrolled manifold
t-SNE
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.5 −1.0 −0.5 0.0 0.5 1.0 1.5


original data space
PC 1
PC 2
Gene 3
Gene 2
Gene 1
PCA
component space
PC 2
PC 1

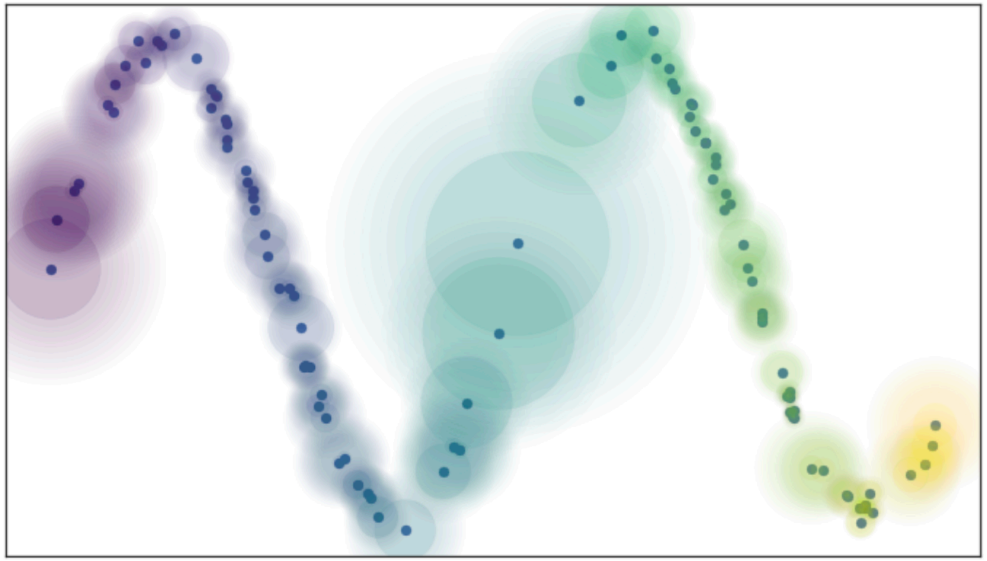
UMAP: Uniform Manifold Approximation and Projection

- Uniform Manifold Approximation and Projection (UMAP) is a dimension reduction technique that can be used for visualization similarly to t-SNE, but also for general non-linear dimension reduction. The algorithm is founded on three assumptions about the data
  - The data is uniformly distributed on a Riemannian manifold;
  - The Riemannian metric is locally constant (or can be approximated as such);
  - The manifold is locally connected.
- From these assumptions it is possible to model the manifold with a fuzzy topological structure. The embedding is found by searching for a low dimensional projection of the data that has the closest possible equivalent fuzzy topological structure.

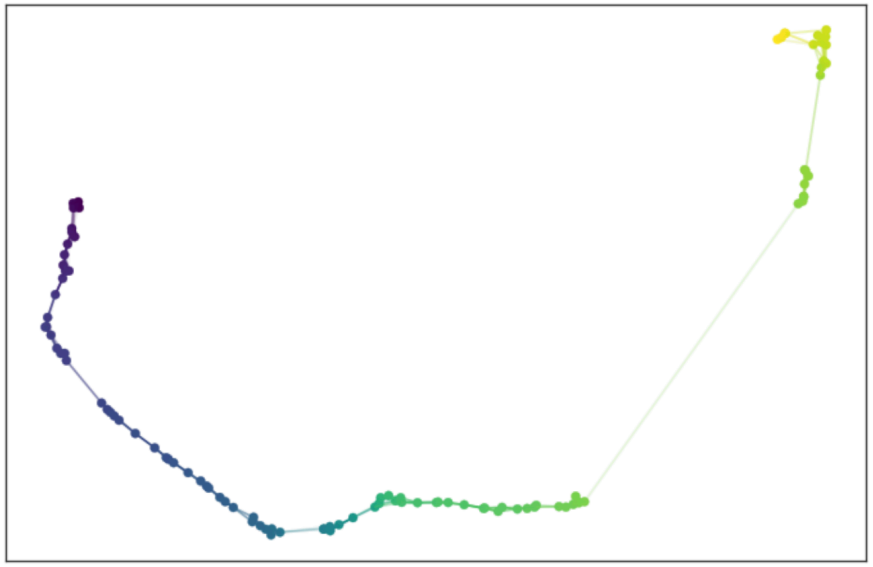


UMAP

Fuzzy topological structure



Low-dimensional representation

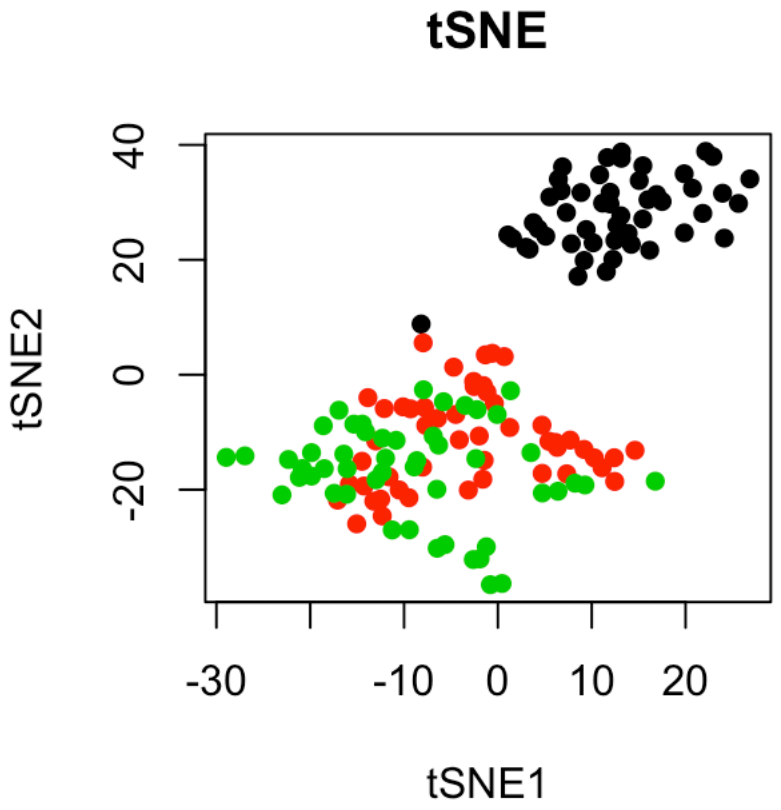


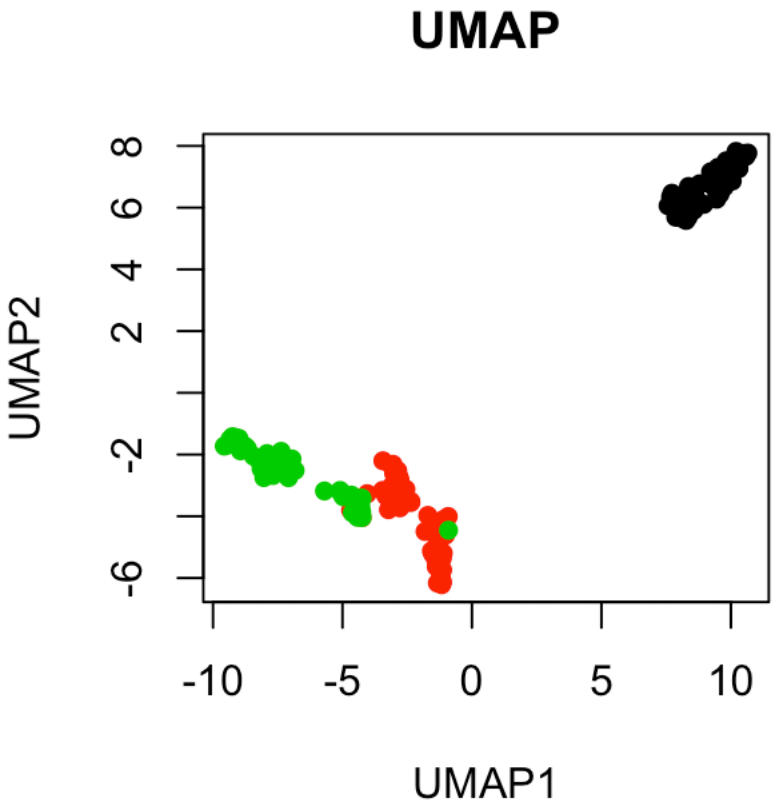
Get the clumps right

$$\sum_{a \in A} \mu(a) \log\left(\frac{\mu(a)}{\nu(a)}\right) + (1-\mu(a)) \log\left(\frac{1-\mu(a)}{1-\nu(a)}\right)$$

Get the gaps right

tSNE v.s. UMAP


tSNE
tSNE2
40
20
0
-20
-30
-10
0
10
20
tSNE1


UMAP
UMAP2
8
6
4
2
-2
-6
-10
-5
0
5
10
UMAP1