MACHINE LEARNING

Dimensionality Reduction + Clustering

Last Update: 22nd December 2022

Prof. Dr. Shadi Albarqouni

Director of Computational Imaging Research Lab. (Albarqouni Lab.)
University Hopsital Bonn | University of Bonn | Helmholtz Munich

STRUCTURE

1. Dimensionality Reduction Methods

1.1 Principle Component Analysis (PCA)

1.2 Variations

2. Clustering

3. Parametric, cost-based clustering

3.1 K-Means

3.2 Extensions

3.3 Comparison

4. Parametric, model-based clustering

4.1 Mixture Models

DIMENSIONALITY REDUCTION METHODS

WHY WE NEED SUBSPACE METHODS?

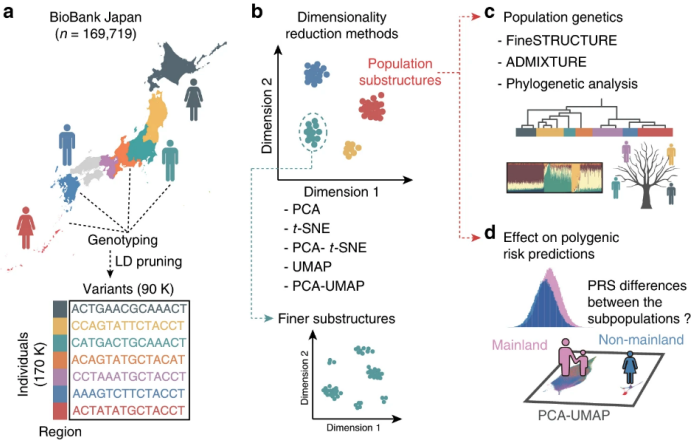
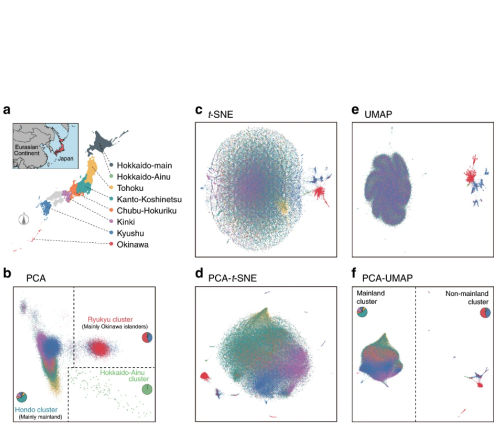


Image adopted from Sakaue, Saori, et al. "Dimensionality reduction reveals fine-scale structure in the Japanese population with consequences for polygenic risk prediction." Nature communications 11.1 (2020): 1-11.

WHY WE NEED SUBSPACE METHODS?



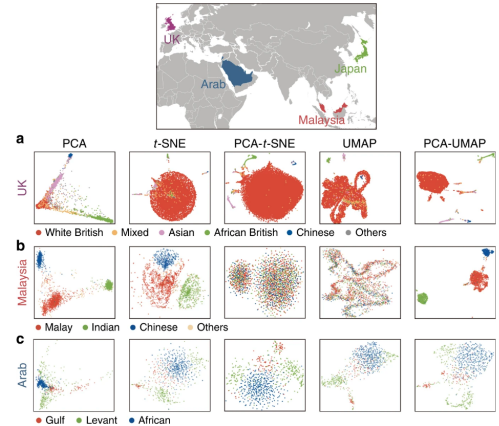


Image adopted from Sakaue, Saori, et al. "Dimensionality reduction reveals fine-scale structure in the Japanese population with consequences for polygenic risk prediction." Nature communications 11.1 (2020): 1-11.

NOTATION

$\mathcal{X}^T = \{x_1, x_2, ..., x_N\}^T \in \mathbb{R}^{d \times N}$ is the data set.

d is the feature dimension of x_i.

N is the number of instances.

Objective

Find a subspace that maximizing the variance among the data.

PRINCIPLE COMPONENT ANALYSIS(PCA)

Objective

To find a subspace that maximize the variance/covariance among the point cloud, we need to find a projection matrix $P \in \mathbb{R}^{r \times d}$ that maps the data $\mathcal{X}^T \in \mathbb{R}^{N \times d}$ into a lower dimensional space (subspace), $\mathcal{X}_{proj}^T \in \mathbb{R}^{N \times r}$,

$$\mathcal{X}_{proj} = P\mathcal{X},$$

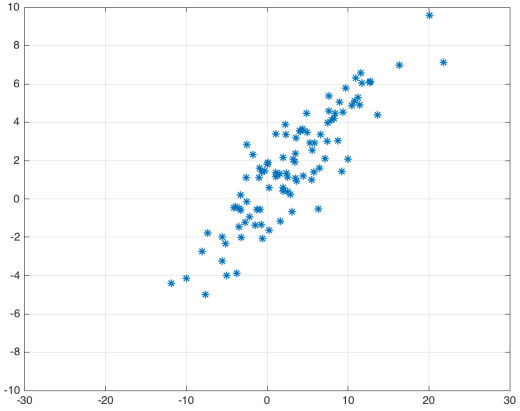
where $r << d$

P should fulfil a few conditions[1]:

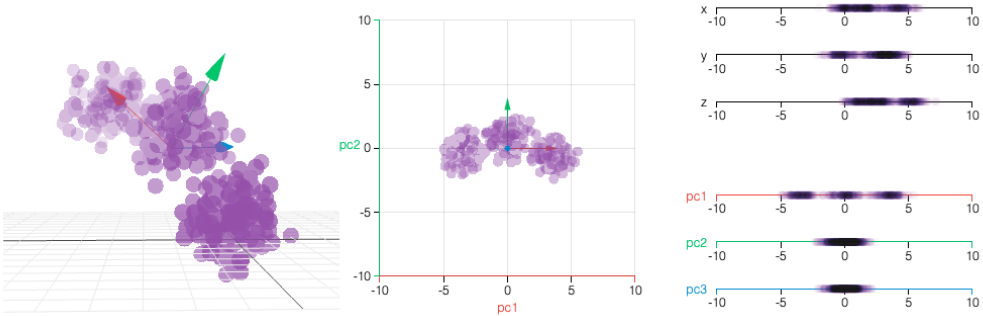
- P has orthonormal basis
- The covariance of the $\mathcal{X}_{proj}$ is diagonal

[1] Shlens, Jonathon. "A tutorial on principal component analysis." arXiv preprint arXiv:1404.1100 (2014).

EXAMPLE



EXAMPLE



Source: **https://setosa.io/ev/principal-component-analysis/**

SINGULAR VALUE DECOMPOSITION (SVD)

Singular Value Decomposition (SVD)[2]

Given a data matrix $X \in \mathbb{R}^{N \times d}$, where N is the number of samples (observations) and d is the feature dimension, the singular value decomposition (SVD) can be computed as follows:

$$X = U \Sigma V^T, \tag{1}$$

where $U \in \mathbb{R}^{N \times N}$ is the left-singular vectors, the diagonal elements of $\Sigma \in \mathbb{R}^{N \times d}$ are the singular values, and $V \in \mathbb{R}^{d \times d}$ is the right-singular vector. The eigenvectors are the same as the right-singular vector, where the eigenvalues are the diagonal elements of $\Sigma^T \Sigma$.

[2] **https://www.youtube.com/watch?v=HMOI_lkzW08**

EIGEN-DECOMPOSITION OF COVARIANCE MATRIX

Eigen-decomposition of Covariance Matrix

Given a covariance matrix $C \in \mathbb{R}^{d \times d}$, which can be computed from the data matrix as $C = X^T X$, the eigenvectors and eigenvalues can be computed as follows:

$$CV = \Lambda V, \tag{2}$$

where $V \in \mathbb{R}^{d \times d}$ is the eigenvectors matrix and the diagonal elements of $\Lambda \in \mathbb{R}^{d \times d}$ represent the eigenvalues.

Eigen-decomposition of Covariance Matrix

Given a covariance matrix $C \in \mathbb{R}^{d \times d}$, which can be computed from the data matrix as $C = X^T X$, the eigenvectors and eigenvalues can be computed as follows:

$$CV = \Lambda V, \tag{2}$$

where $V \in \mathbb{R}^{d \times d}$ is the eigenvectors matrix and the diagonal elements of $\Lambda \in \mathbb{R}^{d \times d}$ represent the eigenvalues.

Connection between Covariance and SVD

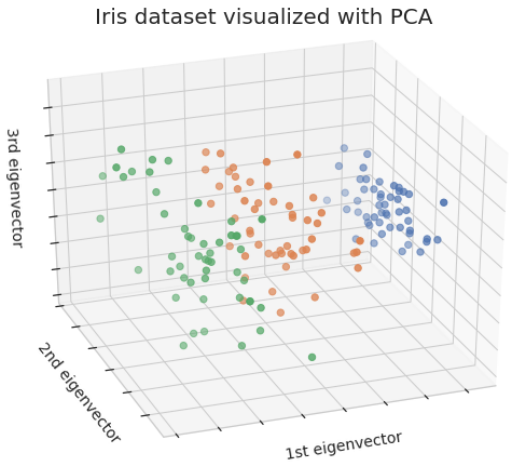
Let's start from Eq.(2), and substitute C with $X^T X$ as follows:

$$\begin{aligned} CV &= \Lambda V, \\ C &= V \Lambda V^T, \\ X^T X &= V \Lambda V^T, \\ (U \Sigma V^T)^T U \Sigma V^T &= V \Lambda V^T, \\ V \Sigma^T \Sigma V^T &= V \Lambda V^T, \end{aligned} \tag{3}$$

where $U^T U = V^T V = I$, and $\Sigma^T \Sigma = \Lambda$. It should be noted that data matrix X has column zero mean (features) and the projected data can be obtained by $X_{proj}^T = V^T X^T$.

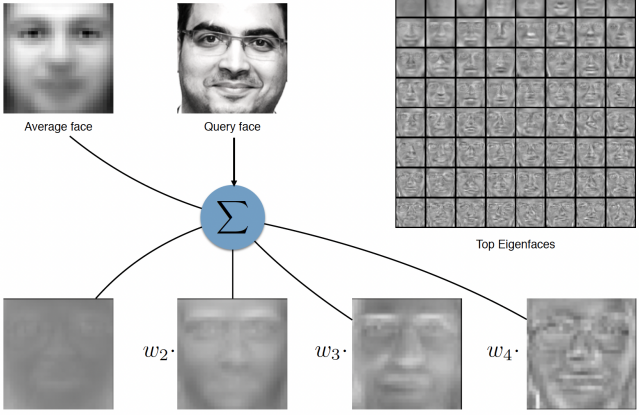
Note: To get consistent results from SVD and Covariance, i.e. for SVD: divide X^T by the $\sqrt{(N-1)}$, COV: divide the $X^T X$ by the $(N-1)$.

DEMO



EXAMPLE

Few example images
Each picture has 19x19 pixels (i.e. the feature space has 361 dims)



VARIATIONS

Kernel PCA

Linear Discriminative Analysis

Independent Component Analysis

Laplacian Eigenmap

CLUSTERING

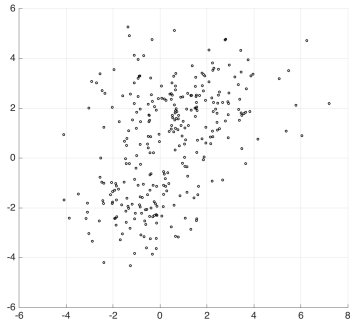
WHAT IS CLUSTERING?

Definition (Clustering)

Given n unlabelled data points, separate them into K clusters.

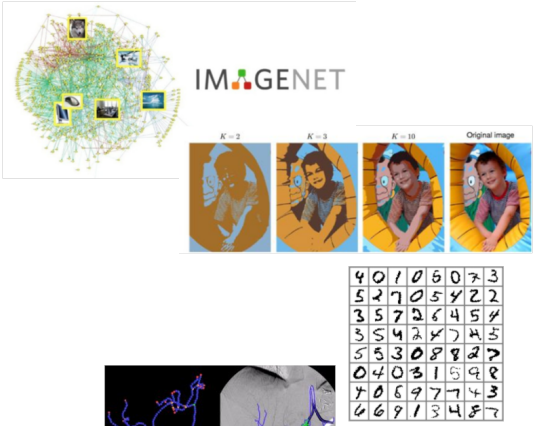
Dilemma! [8]

- What is a Cluster?
  (Compact vs. Connected)
- How many K clusters?
  (Parametric vs. Non-parametric)
- Soft vs. Hard clustering.
  (Model vs. Cost based)
- Data representation.
  (Vector vs. Similarities)
- Classification vs. Clustering.
- Stability [10].



APPLICATIONS

Image Retrieval
Image Compression
Image Segmentation
Pattern Recognition



NOTATION

$\mathcal{X}^T = \{x_1, x_2, ..., x_N\} \in \mathbb{R}^{d \times N}$ is the data set.

d is the feature dimension of x_i.

N is the number of instances.

K is the number of clusters.

$\nabla = \{C_1, C_2, ..., C_K\}$, where C_k is a partition of $\mathcal{X}$.

$c(x_i)$ is the label/cluster of instance x_i.

r_{nk} where n is the index of instance and k is the index of cluster.

Objective

Find the clusters ∇ minimizing the cost function $\mathcal{L}(\nabla)$.

PARAMETRIC, COST-BASED CLUSTERING

PARAMETRIC, COST-BASED CLUSTERING

Parametric: K is defined.
Cost-based: It is hard-clustering based on the cost function.
Selected Algorithms:

- K-Means [11].
- K-Medoids [15].
- Kernel K-Means [16].
- Spectral Clustering [14].

K-MEANS

K-Means algorithm:

Initialize: Pick K random samples from the dataset $\mathcal{X}^T$ as the cluster centroids $\mu_k = \{\mu_1, \mu_2, ..., \mu_K\}$.

Assign Points to the clusters: Partition data points $\mathcal{X}^T$ into K clusters $\nabla = \{C_1, C_2, ..., C_K\}$ based on the Euclidean distance between the points and centroids (searching for the closest centroid).

Centroid update: Based on the points assigned to each cluster, a new centroid is computed μ_k.

Repeat: Do step 2 and 3 until convergence.

Convergence: if the cluster centroids barley change, or we have compact and/or isolated clusters. Mathematically, when the cost (distortion) function $\mathcal{L}(\nabla) = \sum_{k=1}^{K} \sum_{i \in C_k} \|x_i - \mu_k\|^2$ is minimum.

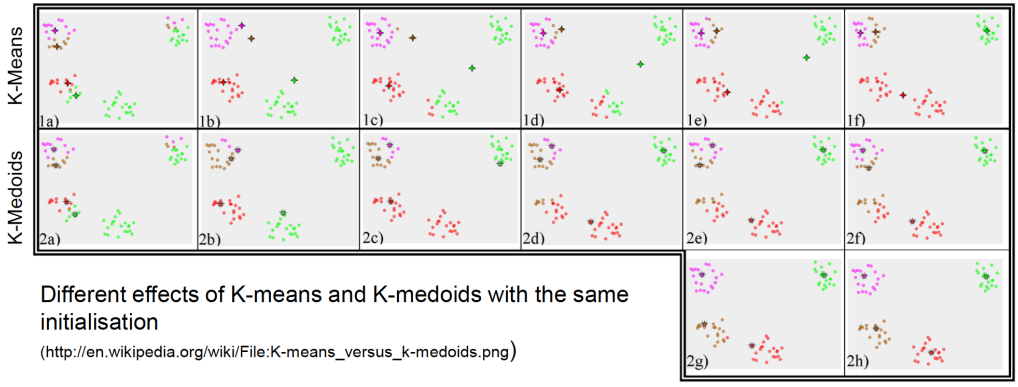
Practical issues:

a) The initialization. b) Pre-processing.

K-MEANS -- ALGORITHM

input : Data points $\mathcal{X}^T = \{x_1, x_2, ..., x_N\}$, number of clusters K
output: Clusters, $\nabla = \{C_1, C_2, ..., C_K\}$

Pick K random samples as the cluster centroids μ_k.
repeat
  for $i = 1$ to N do
    $c(x_i) = \min_{k \in K} \|x_i - \mu_k\|_2^2$ %Assign points to clusters
  end
  for $k = 1$ to K do
    $\mu_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i$ %Update the cluster centroid
  end
until convergence;



Different effects of K-means and K-medoids with the same initialisation

(http://en.wikipedia.org/wiki/File:K-means_versus_k-medoids.png)

EXTENSIONS

Alternative cost (distortion) function:

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\|x_i - x_j\|^2 = \underbrace{\sum_{k=1}^{K}\sum_{i,j\in C_k}\|x_i - x_j\|^2}_{\text{Intracluster distance}} + \underbrace{\sum_{k=1}^{K}\sum_{i\in C_k}\sum_{j\notin C_k}\|x_i - x_j\|^2}_{\text{Intercluster distance}}$$

Intracluster distance:

$$\mathcal{L}(\nabla) = \sum_{k=1}^{K}\sum_{i,j\in C_k}\|x_i - x_j\|^2 + constant$$

Interclsuter distance:

$$\mathcal{L}(\nabla) = -\sum_{k=1}^{K}\sum_{i\in C_k}\sum_{l\notin C_k}\|x_i - x_j\|^2 + constant$$

CONT.

Alternative Initialization:

- K-Means++ [2]
- Global Kernel K-Means [17]

On selecting K [3]:

- Rule of thumb: $K = \sqrt{N/2}$
- Elbow Method
- Silhouette

Soft clustering: Fuzzy C-Means [3]

Variant: Spectral Clustering [18]

Hierarchical Clustering

[3] **https://en.wikipedia.org/wiki/Determining_the_number_of_clusters_in_a_data_set**

COMPARISON

Algorithm	Data Rep.	Comp.	Out.	Cent.
K-Means	Vectors	Low	No	$\notin \mathcal{X}^T$
K-Medians	Vectors	High	No	$\notin \mathcal{X}^T$
K-Medoids	Similarity	High	Yes	$\in \mathcal{X}^T$
Kernel K-Means	Kernel	High	N/A	$\notin \mathcal{X}^T$
Spectral Clustering	Similarity	High	N/A	$\notin \mathcal{X}^T$

4

[4] Data Rep: Data Representation, Comp.: Computational cost, Out.: Handling outliers, Cent.: Centroids.

PARAMETRIC, MODEL-BASED CLUSTERING

PARAMETRIC, MODEL-BASED CLUSTERING

Parametric: K and the density function are defined (i.e. Gaussian)
Model-based: It is soft-clustering based on the mixture density $f(x)$.

$$f(x) = \sum_{k=1}^{K} \pi_k f_k(x), \qquad s.t. \qquad \pi_k \geq 0, \sum_{K} \pi_k = 1,$$

where $f_k(x)$ is the component of mixture. $f(x)$ is a Gaussian Mixture Model (GMM) when $f_k(x) \sim \mathcal{N}(x; \mu_k, \sigma_k^2)$.
Degree of Membership:

$$\gamma_{ki} = P[x_i \in C_k] = \frac{\pi_k f_k(x_i)}{f(x_i)}$$

GMM Parameter: $\theta = \{\pi_{1:K}, \mu_{1:K}, \sigma_{1:K}\}$.
Selected Algorithm to estimate the parameter: EM-Algorithm [6].

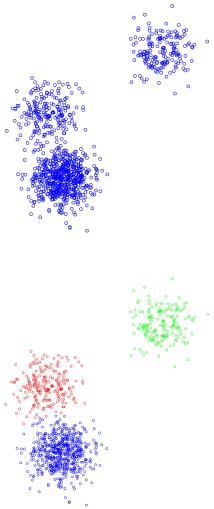
EXPECTATION-MAXIMIZATION (EM) ALGORITHM

Given data points $\mathcal{X}^T$ sampled i.i.d from an unknown distribution f

We need to model the distribution using Maximum Likelihood (ML) principle (log-likelihood):

$$l(\theta) = \ln f_\theta(\mathcal{X}) = \sum_{i=1}^{N} \ln f_\theta(x_i) \triangleq \sum_{i=1}^{N} \ln \sum_{k=1}^{K} \pi_k f_k(x_i)$$

The objective: $\theta^{ML} = \arg\max_\theta l(\theta)$

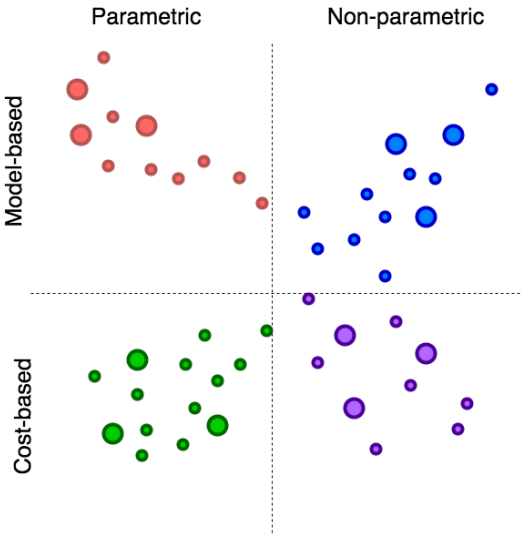


EM -- ALGORITHM

input : data points $\mathcal{X}^T$, number of clusters K
output: Parameters, $\theta^{ML} = \{\pi_{1:K}, \mu_{1:K}, \sigma_{1:K}\}$

Initialize the parameters θ at random.
repeat
  for $i = 1$ to N do
    for $k = 1$ to K do
      $\gamma_{ik} = \frac{\pi_k f_k(x_i)}{f(x_i)}$ %E-Step
    end
  end
  for $k = 1$ to K do
    $\pi_k = \frac{1}{N} \sum_{i=1}^{N} \gamma_{ik}$ %M-Step
    $\mu_k = \frac{1}{N\pi_k} \sum_{i=1}^{N} \gamma_{ik} x_i$
    $\sigma_k = \frac{1}{N\pi_k} \sum_{i=1}^{N} \gamma_{ik} (x_i - \mu_k)(x_i - \mu_k)^T$
  end
until convergence;

SUMMARY



REFERENCES

- Michal Aharon, Michael Elad, and Alfred Bruckstein.
  K-svd: An algorithm for designing overcomplete dictionaries for sparse representation.
  Signal Processing, IEEE Transactions on, 54(11):4311–4322, 2006.
- David Arthur and Sergei Vassilvitskii.
  k-means++: The advantages of careful seeding.
  In Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.
- James C Bezdek.
  Pattern recognition with fuzzy objective function algorithms.
  Springer Science & Business Media, 2013.
- Christopher M Bishop.
  Pattern recognition