DSSP 14 : Unsupervised Learning

December 5, 2019

Outline

Motivation

- **Marketing:** finding groups of customers with similar behavior given a large database of customer data containing their properties and past buying records;
- **Biology:** classification of plants and animals given their features;
- **Libraries:** book ordering;
- **Insurance:** identifying groups of motor insurance policy holders with a high average claim cost; identifying frauds;
- **City-planning:** identifying groups of houses according to their house type, value and geographical location;
- **Internet:** document classification; clustering weblog data to discover groups of similar access patterns.

Marketing

- **Data:** Base of customer data containing their properties and past buying records
- **Goal:** Use the customers *similarities* to find groups.
- **Two directions:**
  - **Visualization:** propose a representation of the customers so that the groups are *visible*
  - **Clustering:** propose an explicit *grouping* of the customers

Source: Unknown

Dimension Reduction

- How to view a high-dimensional dataset?
- High-dimension: dimension larger than 2!
- *Projection* in a 2D space.

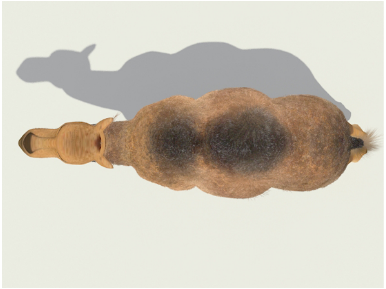
Source: F. Belardi

Dimension Reduction

- How to view a high-dimensional dataset?
- High-dimension: dimension larger than 2!
- *Projection* in a 2D space.

Source: F. Belardi

Dimension Reduction



- How to view a high-dimensional dataset?
- High-dimension: dimension larger than 2!
- *Projection* in a 2D space.

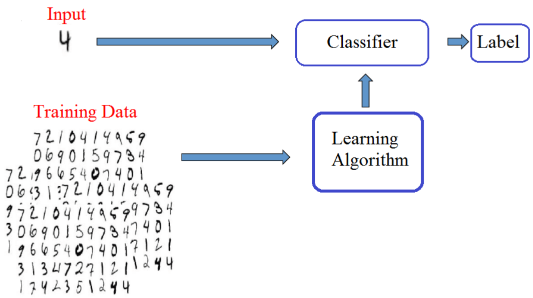
Source: F. Belardi

Dimension Reduction

- How to view a high-dimensional dataset?
- High-dimension: dimension larger than 2!
- *Projection* in a 2D space.

Source: F. Belardi

Machine Learning



A definition by Tom Mitchell (`http://www.cs.cmu.edu/~tom/`)

A computer program is said to learn from **experience E** with respect to some **class of tasks T** and **performance measure P**, if its performance at tasks in T, as measured by P, improves with experience E.

Source: Unknown

Supervised Learning

Experience, Task and Performance measure

- **Training data** : $\mathcal{D} = \{(\underline{X}_1, Y_1), \dots, (\underline{X}_n, Y_n)\}$ (i.i.d. $\sim \mathbb{P}$)
- **Predictor**: $f : \mathcal{X} \to \mathcal{Y}$ measurable
- **Cost/Loss function**: $\ell(f(\underline{X}), Y)$ measure how well $f(\underline{X})$ *predicts* Y
- **Risk**:
$$\mathcal{R}(f) = \mathbb{E}\left[\ell(Y, f(\underline{X}))\right] = \mathbb{E}_X\left[\mathbb{E}_{Y|\underline{X}}\left[\ell(Y, f(\underline{X}))\right]\right]$$

- Often $\ell(f(\underline{X}), Y) = \|f(\underline{X}) - Y\|^2$ or $\ell(f(\underline{X}), Y) = \mathbf{1}_{Y \neq f(\underline{X})}$

Goal

- Learn a rule to construct a **classifier** $\widehat{f} \in \mathcal{F}$ from the training data $\mathcal{D}_n$ s.t. **the risk** $\mathcal{R}(\widehat{f})$ is **small on average** or with high probability with respect to $\mathcal{D}_n$.

Unsupervised Learning

Experience, Task and Performance measure

- **Training data** : $\mathcal{D} = \{\underline{X}_1, \ldots, \underline{X}_n\}$ (i.i.d. $\sim \mathbb{P}$)
- **Task**: ???
- **Performance measure**: ???

- No obvious task definition!

Tasks for this lecture

- **Dimension reduction:** construct a map of the data in a **low dimensional** space without **distorting** it too much.
- **Clustering (or unsupervised classification):** construct a **grouping** of the data in **homogeneous** classes.

Dimension Reduction

- **Training data** : $\mathcal{D} = \{\underline{X}_1, \ldots, \underline{X}_n\} \in \mathcal{X}^n$ (i.i.d. $\sim \mathbb{P}$)
- Space $\mathcal{X}$ of possibly high dimension.

Dimension Reduction Map

- Construct a map Φ from the space $\mathcal{X}$ into a space $\mathcal{X}'$ of **smaller dimension**:

$$\begin{aligned} \Phi : \quad & \mathcal{X} \to \mathcal{X}' \\ & \underline{X} \mapsto \Phi(\underline{X}) \end{aligned}$$

- Map can be defined only on the dataset.

Motivations

- Visualization of the data
- Dimension reduction before further processing

Dimension Reduction

- Need to control the **distortion** between $\mathcal{D}$ and $\Phi(\mathcal{D}) = \{\Phi(\underline{X}_1), \ldots, \Phi(\underline{X}_n)\}$

Distortion(s)

- Reconstruction error:
  - Construct $\widetilde{\Phi}$ from $\mathcal{X}'$ to $\mathcal{X}$
  - Control the error between $\underline{X}$ and its reconstruction $\widetilde{\Phi}(\Phi(\underline{X}))$
- Relationship preservation:
  - Compute a *relation* $\underline{X}_i$ and $\underline{X}_j$ and a *relation* between $\Phi(\underline{X}_i)$ and $\Phi(\underline{X}_j)$
  - Control the difference between those two *relations*.

- Leads to different constructions....

Clustering

- **Training data** : $\mathcal{D} = \{\underline{X}_1, \dots, \underline{X}_n\} \in \mathcal{X}^n$ (i.i.d. $\sim \mathbb{P}$)
- Latent groups?

Clustering

- Construct a map f from $\mathcal{D}$ to $\{1, \dots, K\}$ where K is a number of classes to be fixed:

$$f : \quad \underline{X}_i \mapsto k_i$$

- Similar to classification except:
  - no ground truth (no given labels)
  - label only elements of the dataset!

Motivations

- Interpretation of the groups
- Use of the groups in further processing

Clustering

- Need to define the **quality** of the cluster.
- No obvious measure!

Clustering quality

- Inner homogeneity: samples in the same group should be similar.
- Outer inhomogeneity: samples in two different groups should be different.

- Several possible definitions of similar and different.
- Often based on the distance between the samples.
- Example based on the euclidean distance:
  - Inner homogeneity = intra class variance,
  - Outer inhomogeneity = inter class variance.
- **Beware:** choice of the number of cluster K often complex!

Outline

Dimension Reduction

- **Training data** : $\mathcal{D} = \{\underline{X}_1, \ldots, \underline{X}_n\} \in \mathcal{X}^n \quad$ (i.i.d. $\sim \mathbb{P}$)
- Space $\mathcal{X}$ of possibly high dimension.

Dimension Reduction Map

- Construct a map Φ from the space $\mathcal{X}$ into a space $\mathcal{X}'$ of **smaller dimension**:

$$\begin{aligned} \Phi : \quad \mathcal{X} &\to \mathcal{X}' \\ \underline{X} &\mapsto \Phi(\underline{X}) \end{aligned}$$

Criterion

- Reconstruction error
- Relationship preservation

Outline

Dimensionality Curse

High Dimension Geometry Curse

- Folks theorem: In high dimension, everyone is alone.
- Theorem: If $\underline{X}_1, \dots, \underline{X}_n$ in the hypercube of dimension d such that their coordinates are i.i.d then

$$d^{-1/p}\left(\max \|\underline{X}_i - \underline{X}_j\|_p - \min \|\underline{X}_i - \underline{X}_j\|_p\right) = 0 + O\left(\sqrt{\frac{\log n}{d}}\right)$$

$$\frac{\max \|\underline{X}_i - \underline{X}_j\|_p}{\min \|\underline{X}_i - \underline{X}_j\|_p} = 1 + O\left(\sqrt{\frac{\log n}{d}}\right).$$

- When d is large, all the points are almost equidistant...
- Nearest neighbors are meaningless!

Inertia

- $\underline{X}_1, \ldots, \underline{X}_n \in \mathbf{R}^d$
- $m = \frac{1}{n} \sum_{i=1}^{n} \underline{X}_i$

Two views on inertia

- Inertia:

$$\begin{aligned} I &= \frac{1}{n} \sum_{i=1}^{n} \|\underline{X}_i - m\|^2 \\ &= \frac{1}{2n^2} \sum_{i,j} \|\underline{X}_i - \underline{X}_j\|^2 \end{aligned}$$

- 2 times the mean squared distance to the mean = Mean squared distance between individual

- Heuristic: a good representation is a representation with a large inertia
- Large dispersion $\sim$ Large average separation!

Inertia and Projection

- What if we replace $\underline{X}$ by its projection $\widetilde{\underline{X}} = P(\underline{X} - m) + m$?

Two views on inertia

- Inertia:

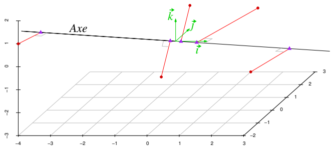
$$\begin{aligned}\widetilde{I} &= \frac{1}{n}\sum_{i=1}^{n} \|\widetilde{\underline{X}}_i - m\|^2 \\ &= \frac{1}{2n^2}\sum_{i,j} \|\widetilde{\underline{X}}_i - \widetilde{\underline{X}}_j\|^2\end{aligned}$$

- Inertia:

$$\begin{aligned}\widetilde{I} &= I - \frac{1}{n}\sum_{i=1}^{n} \|\widetilde{\underline{X}}_i - \underline{X}_i\|^2 \\ &= I - \frac{1}{2n^2}\sum_{i,j} \left(\|\underline{X}_i - \underline{X}_j\|^2 - \|\widetilde{\underline{X}}_i - \widetilde{\underline{X}}_j\|^2\right)\end{aligned}$$

- Four different way to obtain a large inertia!

First Component of the PCA



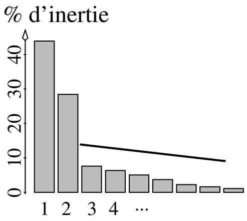
- 1D case: $\underset{\sim}{\widetilde{X}} = m + a^\top(\underline{X} - m)a$ with $\|a\| = 1$
- Inertia: $\widetilde{I} = \frac{1}{n}\sum_{i=1}^{n} a^\top(\underline{X}_i - m)(\underline{X}_i - m)^\top a$

Principal Component Analysis : optimization of the projection

- Maximization of $\widetilde{I} = \frac{1}{n}\sum_{i=1}^{n} a^\top(\underline{X}_i - m)(\underline{X}_i - m)^\top a = a^\top \Sigma a$ with $\Sigma = \frac{1}{n}\sum_{i=1}^{n}(\underline{X}_i - m)(\underline{X}_i - m)^\top$ the empirical covariance matrix.
- Explicit optimal choice given by the eigenvector of the largest eigenvalue of Σ.

Source: E. Matzner-Løber

PCA



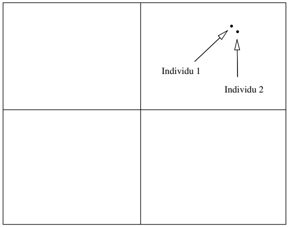
Principal Component Analysis : optimization of the projection

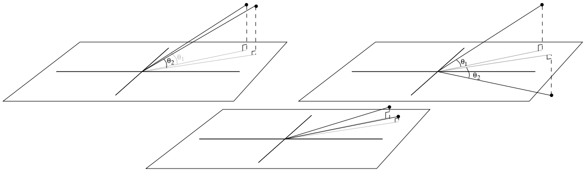
- Explicit optimal solution obtain by the projection on the eigenvectors of the largest eigenvalues of Σ.
- Projected inertia given by the sum of those eigenvalues.

- Often fast decay of the eigenvalues: some dimensions are much more important than other.
- Not exactly the curse of dimensionality setting...
- Yet a lot of *small* dimension can drive the distance!

Source: E. Matzner-Løber

PCA and Distances





Close projection doesn't mean close individuals!

- Same projections but different situations.
- Quality of the projection measured by the angle!

Source: E. Matzner-Løber

Outline

Reconstruction Error Approach

Goal

- Construct a map Φ from the space $\mathcal{X}$ into a space $\mathcal{X}'$ of **smaller dimension**:
$$\begin{aligned} \Phi : \quad & \mathcal{X} \to \mathcal{X}' \\ & \underline{X} \mapsto \Phi(\underline{X}) \end{aligned}$$
- Construct $\widetilde{\Phi}$ from $\mathcal{X}'$ to $\mathcal{X}$
- Control the error between $\underline{X}$ and its reconstruction $\widetilde{\Phi}(\Phi(\underline{X}))$

- Canonical example for $\underline{X} \in \mathbb{R}^d$: find Φ and $\widetilde{\Phi}$ in a parametric family that minimize
$$\frac{1}{n}\sum_{i=1}^{n} \|\underline{X}_i - \widetilde{\Phi}(\Phi(\underline{X}_i))\|^2$$

Principal Component Analysis

- $\mathcal{X} \in \mathbb{R}^d$ and $\mathcal{X}' = \mathbb{R}^{d'}$
- Affine model $\underline{X} \sim m + \sum_{l=1}^{d'} \underline{X}'^{(l)} V^{(l)}$ with $(V^{(l)})$ an orthonormal family.
- Equivalent to:
$$\Phi(\underline{X}) = V^\top(\underline{X} - m) \quad \text{and} \quad \widetilde{\Phi}(\underline{X}') = m + V\underline{X}'$$
- Reconstruction error criterion:
$$\frac{1}{n}\sum_{i=1}^{n} \|\underline{X}_i - (m + VV^\top(\underline{X}_i - m)\|^2$$
- **Explicit solution:** m is the empirical mean and V is any orthonormal basis of the space spanned by the d' first eigenvectors (the one with largest eigenvalues) of the empirical covariance matrix $\frac{1}{n}\sum_{i=1}^{n}(\underline{X}_i - m)(\underline{X}_i - m)^\top$.

Principal Component Analysis

PCA Algorithm

- Compute the empirical mean $m = \frac{1}{n} \sum_{i=1}^{n} \underline{X}_i$
- Compute the empirical covariance matrix $\frac{1}{n} \sum_{i=1}^{n} (\underline{X}_i - m)(\underline{X}_i - m)^\top$.
- Compute the d' first eigenvectors of this matrix: $V^{(1)}, \dots, V^{(d')}$
- Set $\Phi(\underline{X}) = V^\top(\underline{X} - m)$

- Complexity: $O(n(d + d^2) + d'd^2)$
- Interpretation:
  - $\Phi(\underline{X}) = V^\top(\underline{X} - m)$: coordinates in the restricted space.
  - $V^{(i)}$: influence of each original coordinates in the ith new one.
- **Scaling:** This method is not invariant to a scaling of the variables! It is custom to normalize the variables (at least within groups) before applying PCA.

Multiple Factor Analysis

- PCA assumes $\mathcal{X} = \mathbb{R}^d$!
- How to deal with categorical values?
- MFA = PCA with clever coding strategy for categorical values.

Categorical value code for a single variable

- Classical redundant dummy coding:
$$\underline{X} \in \{1, \ldots, V\} \mapsto P(\underline{X}) = \left(\mathbf{1}_{\underline{X}=1}, \ldots, \mathbf{1}_{\underline{X}=V}\right)^\top$$
- Compute the mean (i.e. the empirical proportions): $\overline{P} = \frac{1}{n}\sum_{i=1}^n P(\underline{X}_i)$
- *Renormalize* $P(\underline{X})$ by $1/\sqrt{(V-1)\overline{P}}$:
$$P(\underline{X}) \mapsto P^r(\underline{X})$$
$$(\mathbf{1}_{\underline{X}=1}, \ldots \mathbf{1}_{\underline{X}=V}) \mapsto \left(\frac{\mathbf{1}_{\underline{X}=1}}{\sqrt{(V-1)\overline{P}_1}}, \ldots, \frac{\mathbf{1}_{\underline{X}=V}}{\sqrt{(V-1)\overline{P}_V}}\right)$$

- χ^2 type distance!

Multiple Factor Analysis

- PCA becomes the minimization of

$$\frac{1}{n}\sum_{i=1}^{n}\|P^r(\underline{X}_i) - (m + VV^\top(P^r(\underline{X}_i) - m))\|^2$$

$$= \frac{1}{n}\sum_{i=1}^{n}\sum_{v=1}^{V} \frac{\left|\mathbf{1}_{\underline{X}_i=v} - (m' + \sum_{l=1}^{d'} V^{(l)\top}(P(\underline{X}_i) - m')V^{(l,v)})\right|^2}{(V-1)\overline{P}_v}$$

- Interpretation:
  - $m' = \overline{P}$
  - $\Phi(\underline{X}) = V^\top(P^r\underline{X} - m)$: coordinates in the restricted space.
  - $V^{(l)}$ can be interpreted s as a probability profile.
- Complexity: $O(n(V + V^2) + d'V^2)$
- Link with Correspondence Analysis (CA)

Multiple Factor Analysis

MFA Algorithm

- Redundant dummy coding of each categorical variable.
- Renormalization of each block of dummy variable.
- Classical PCA algorithm on the resulting variables

- Interpretation as a reconstruction error with a rescaled/χ^2 metric.
- Interpretation:
  - $\Phi(\underline{X}) = V^\top(P^r(\underline{X}) - m)$: coordinates in the restricted space.
  - $V^{(l)}$: influence of each modality/variable in the ith new coordinates.
- **Scaling:** This method is not invariant to a scaling of the continuous variables! It is custom to normalize the variables (at least within groups) before applying PCA.

Non Linear PCA

PCA Model

- PCA: Linear model assumption

$$\underline{X} \simeq m + \sum_{l=1}^{d'} \underline{X}^{\prime,(l)} V^{(l)} = m + V\underline{X}'$$

- with
  - $V^{(l)}$ orthonormal
  - $\underline{X}^{\prime,(l)}$ without constrains.

- Two directions of extension:
  - Other constrains on V (or the coordinates in the restricted space): ICA, NMF, Dictionary approach
  - PCA on a non linear image of $\underline{X}$: kernel-PCA
- Much more complex algorithm!

Non Linear PCA

ICA (Independent Component Analysis)

- Linear model assumption
- with
$$\underline{X} \simeq m + \sum_{l=1}^{d'} \underline{X}'^{,(l)} V^{(l)} = m + V\underline{X}'$$
  - $V^{(l)}$ without constrains.
  - $\underline{X}'^{,(l)}$ independent

NMF (Non Negative Matrix Factorization)

- (Linear) Model assumption
- with
$$\underline{X} \simeq m + \sum_{l=1}^{d'} \underline{X}'^{,(l)} V^{(l)} = m + V\underline{X}'$$
  - $V^{(l)}$ non negative
  - $\underline{X}'^{,(l)}$ non negative.

Non Linear PCA

Dictionary

- (Linear) Model assumption

$$\underline{X} \simeq m + \sum_{l=1}^{d'} \underline{X}'^{,(l)} V^{(l)} = m + V \underline{X}'$$

- with
  - $V^{(l)}$ without constrains
  - $\underline{X}'$ sparse (with a lot of 0)

kernel PCA

- Linear model assumption

$$\Psi(\underline{X} - m) \simeq \sum_{l=1}^{d'} \underline{X}'^{,(l)} V^{(l)} = V \underline{X}'$$

- with
  - $V^{(l)}$ orthonormal
  - $\underline{X}'_l$ without constrains.

Link with SVD

- Linear model assumption:

$$\underline{X} \simeq m + \sum_{l=1}^{d'} \underline{X}'^{,(l)} V^{(l)} = m + V \underline{X}'$$

- Vector rewriting

$$\underline{X}^\top \simeq m^\top + \underline{X}'^\top V^\top$$

Matrix Rewriting and Low Rank Factorization

- Matrix rewriting

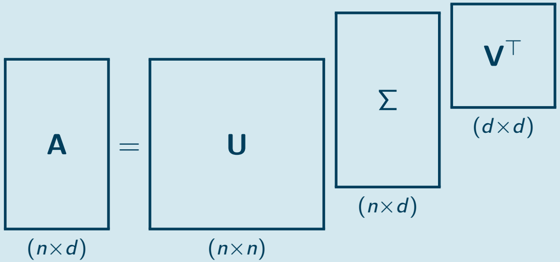
$$\underset{(n \times d)}{\boxed{\begin{array}{c} \underline{X}_1{}^\top - m^\top \\ \vdots \\ \vdots \\ \underline{X}_n{}^\top - m^\top \end{array}}} \simeq \underset{(n \times d')}{\boxed{\begin{array}{c} \underline{X}_1{}'^\top \\ \vdots \\ \vdots \\ \underline{X}_n{}'^\top \end{array}}} \underset{(d' \times d)}{\boxed{\mathbf{V}^\top}}$$

- Low rank matrix factorization! (Truncated SVD solution...)

SVD

SVD Decomposition

- Any matrix $n \times d$ matrix A can de decomposed as

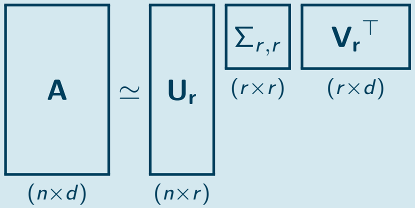


with U and V two orthononormal matrices and Σ a *diagonal* matrix with decreasing values.

SVD

Low Rank Approximation

- The best low rank approximation or rank r is obtained by restriction of the matrices to the first r dimensions:



for both the operator norm and the Frobenius norm!

- PCA: Frobenius norm, $d' = r$ and

$$\begin{pmatrix} \underline{X}_1{}^\top - m^\top \\ \vdots \\ \vdots \\ \underline{X}_n{}^\top - m^\top \end{pmatrix} \leftrightarrow A, \quad \begin{pmatrix} \underline{X}_1{}'^\top \\ \vdots \\ \vdots \\ \underline{X}_n{}'^\top \end{pmatrix} \leftrightarrow \mathbf{U_r}\Sigma_{r,r}, \quad \mathbf{V}^\top \leftrightarrow \mathbf{V_r}^\top$$

Auto Encoder

Deep Auto Encoder

- Construct a map Φ with a **NN** from the space $\mathcal{X}$ into a space $\mathcal{X}'$ of smaller dimension:

$$\begin{aligned}\Phi: \quad & \mathcal{X} \to \mathcal{X}' \\ & \underline{X} \mapsto \Phi(\underline{X})\end{aligned}$$

- Construct $\widetilde{\Phi}$ with a **NN** from $\mathcal{X}'$ to $\mathcal{X}$
- Control the error between $\underline{X}$ and its reconstruction $\widetilde{\Phi}(\Phi(\underline{X}))$:

$$\frac{1}{n}\sum_{i=1}^{n} \|\underline{X}_i - \widetilde{\Phi}(\Phi(\underline{X}_i))\|^2$$

- Optimization by gradient descent.
- NN can be replaced by another parametric function...

Outline

Pairwise Relation

- Different point of view!
- Focus on pairwise relation $\mathcal{R}(\underline{X}_i, \underline{X}_j)$.

Distance Preservation

- Construct a map Φ from the space $\mathcal{X}$ into a space $\mathcal{X}'$ of **smaller dimension**:

$$\begin{aligned} \Phi: \quad & \mathcal{X} \to \mathcal{X}' \\ & \underline{X} \mapsto \Phi(\underline{X}) = \underline{X}' \end{aligned}$$

- such that

$$\mathcal{R}(\underline{X}_i, \underline{X}_j) \sim \mathcal{R}'(\underline{X}'_i, \underline{X}'_j)$$

- Most classical version (MDS):
  - Scalar product relation: $\mathcal{R}(\underline{X}_i, \underline{X}_j) = (\underline{X}_i - m)^\top (\underline{X}_j - m)$
  - Linear mapping $\underline{X}' = \Phi(\underline{X}) = V^\top (\underline{X} - m)$.
  - Euclidean scalar product matching:

$$\frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \left| (\underline{X}_i - m)^\top (\underline{X}_j - m) - (\underline{X}'_i)^\top \underline{X}'_j \right|^2$$

- Φ often defined only on $\mathcal{D}$...

MultiDimensional Scaling

MDS Heuristic

- Match the *scalar* products:
$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left|(\underline{X}_i - m)^\top(\underline{X}_j - m) - \underline{X}_i{'}^\top \underline{X}_j'\right|^2$$
- Linear method: $\underline{X}' = U^\top(\underline{X} - m)$ with U orthonormal

- **Beware:** $\underline{X}$ can be unknown, only the scalar products are required!
- Resulting criterion: minimization in $U^\top(\underline{X}_i - m)$ of
$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left|(\underline{X}_i - m)^\top(\underline{X}_j - m) - (\underline{X}_i - m)^\top UU^\top(\underline{X}_j - m)\right|^2$$
without knowing explicitly $\underline{X}$...
- Explicit solution obtained through the eigendecomposition of the know Gram matrix $(\underline{X}_i - m)^\top(\underline{X}_j - m)$ by keeping only the d' largest eigenvalues.

MultiDimensional Scaling

- In this case, MDS yields the same result than the PCA (but with different inputs, distance between observation vs correlations)!
- **Explanation:** Same SVD problem up to a transposition:
  - MDS
  $$\underline{\overline{X}}_{(n)}{}^\top \underline{\overline{X}}_{(n)} \sim \underline{\overline{X}}_{(n)}{}^\top U U^\top \underline{\overline{X}}_{(n)}$$
  - PCA
  $$\underline{\overline{X}}_{(n)} \underline{\overline{X}}_{(n)}{}^\top \sim U^\top \underline{\overline{X}}_{(n)} \underline{\overline{X}}_{(n)}{}^\top U$$
- Complexity: PCA $O((n+d')d^2)$ vs MDS $O((d+d')n^2)$...

Generalized MDS

- Preserving the scalar products amounts to preserve the euclidean distance.
- Easier **generalization** if we work in term of distance!

Generalized MDS

- Generalized MDS:
  - Distance relation: $\mathcal{R}(\underline{X}_i, \underline{X}_j) = d(\underline{X}_i, \underline{X}_j)$
  - Linear mapping $\underline{X}' = \Phi(\underline{X}) = V^\top(\underline{X} - m)$.
  - Euclidean matching:

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left|d(\underline{X}_i, \underline{X}_j) - d'(\underline{X}'_i, \underline{X}'_j)\right|^2$$

- Strong connection (but no equivalence) with MDS when $d(x, y) = \|x - y\|^2$!
- **Minimization:** Simple gradient descent can be used (can be stuck in local minima).

ISOMAP

- MDS: equivalent to PCA (but more expensive) if $d(x, y) = \|x - y\|^2$!
- ISOMAP: use a *localized* distance instead to limit the influence of very far point.

ISOMAP

- For each point $\underline{X}_i$, define a neighborhood $\mathcal{N}_i$ (either by a distance or a number of points) and let

$$d_0(\underline{X}_i, \underline{X}_j) = \begin{cases} +\infty & \text{if } \underline{X}_j \notin \mathcal{N}_i \\ \|\underline{X}_i - \underline{X}_j\|^2 & \text{otherwise} \end{cases}$$

- Compute the shortest path distance for each pair.
- Use the MDS algorithm with this distance

Random Projection

Random Projection Heuristic

- Draw at random d' unit vector (direction) U_i.
- Use $\underline{X}' = U^\top(\underline{X} - m)$ with $m = \frac{1}{n}\sum_{i=1}^{n} \underline{X}_i$

- **Property:** If $\underline{X}$ lives in a space of dimension d'', then, as soon as, $d' \sim d'' \log(d'')$,
$$\|\underline{X}_i - \underline{X}_j\|^2 \sim \frac{d}{d'}\|\underline{X}'_i - \underline{X}'_j\|^2$$
- Do not really use the data!

Locally Linear Embedding

LLE Heuristic

- For each point $\underline{X}_i$, define a neighborhood $\mathcal{N}_i$ (either by a distance or a number of points).
- Compute some weights $W_{i,j}$ such that
$$W_{i,j} = 0 \quad \text{if } \underline{X}_j \notin \mathcal{N}_i$$
$$\underline{X}_i \sim \sum_j W_{i,j} \underline{X}_j$$
- Find some $\underline{X}'_i$ in a space $\mathcal{X}'$ of **smaller dimension** such that
$$\underline{X}'_i \sim \sum_j W_{i,j} \underline{X}'_j$$

- LLE: use a least square metric for the fits.

t-Stochastic Neighbor Embedding

SNE heuristic

- From $\underline{X}_i \in \mathcal{X}$, construct a set of conditional probability:
$$P_{j|i} = \frac{e^{-\|\underline{X}_i - \underline{X}_j\|^2/2\sigma_i^2}}{\sum_{k\neq i} e^{-\|\underline{X}_i - \underline{X}_j\|^2/2\sigma_i^2}} \qquad P_{i|i} = 0$$
- Find $\underline{X}'_i$ in $\mathbb{R}^{d'}$ such that the set of conditional probability:
$$Q_{j|i} = \frac{e^{-\|\underline{X}'_i - \underline{X}'_j\|^2/2\sigma_i^2}}{\sum_{k\neq i} e^{-\|\underline{X}'_i - \underline{X}'_j\|^2/2\sigma_i^2}} \qquad Q_{i|i} = 0$$
is close from P.

- **t-SNE:** use a Student-t term $(1 + \|\underline{X}'_i - \underline{X}'_j\|^2)^{-1}$ for $\underline{X}'_i$
- Minimize the Kullback-Leibler divergence $(\sum_{i,j} P_{j|i} \log \frac{P_{j|i}}{Q_{j|i}})$ by a simple gradient descent (can be stuck in local minima).
- Parameters σ_i such that $H(P_i) = -\sum_{j=1}^{n} P_{j|i} \log P_{j|i} = \text{cst}$.

UMAP

- Topological Data Analysis inspired.

Uniform Manifold Approximation and Projection

- Define a notion of asymetric scaled local proximity between neighbors:
  - Compute the k-neighborhood of $\underline{X}_i$, its diameter σ_i and the distance ρ_i between $\underline{X}_i$ and its nearest neighbor.
  - Define
    $$w_i(\underline{X}_i, \underline{X}_j) = \begin{cases} e^{-(d(\underline{X}_i,\underline{X}_j)-\rho_i)/\sigma_i} & \text{for } \underline{X}_j \text{ in the } k\text{-neighborhood} \\ 0 & \text{otherwise} \end{cases}$$
- Symmetrize into a *fuzzy* nearest neighbor criterion
  $$w(\underline{X}_i, \underline{X}_j) = w_i(\underline{X}_i, \underline{X}_j) + w_j(\underline{X}_j, \underline{X}_i) - w_i(\underline{X}_i, \underline{X}_j) w_j(\underline{X}_j, \underline{X}_i)$$
- Determine the points $\underline{X}'_i$ in a low dimensional space such that

$$\sum_{i \neq j} w(\underline{X}_i, \underline{X}_j) \log \left(\frac{w(\underline{X}_i, \underline{X}_j)}{w'(\underline{X}'_i, \underline{X}'_j)} \right) + (1 - w(\underline{X}_i, \underline{X}_j)) \log \left(\frac{(1 - w(\underline{X}_i, \underline{X}_j))}{(1 - w'(\underline{X}'_i, \underline{X}'_j))} \right)$$

- Can be performed by local gradient descent.

Graph based

Graph heuristic

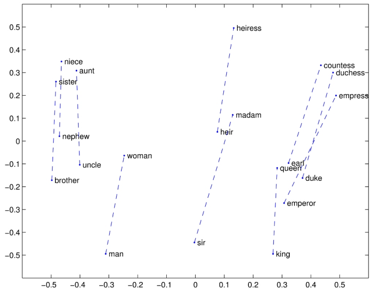
- Construct a graph with weighted edges $w_{i,j}$ measuring the *proximity* of $\underline{X}_i$ and $\underline{X}_j$ ($w_{i,j}$ large if close and 0 if there is no information).
- Find the points $\underline{X}'_i \in \mathbb{R}^{d'}$ minimizing

$$\frac{1}{n}\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{i,j}\|\underline{X}'_i - \underline{X}'_j\|^2$$

- Need of a constraint on the size of $\underline{X}'_i$...
- Explicit solution through linear algebra: d' eigenvectors with smallest eigenvalues of the Laplacian of the graph $D - W$, where D is a diagonal matrix with $D_{i,i} = \sum_j w_{i,j}$.
- Variation on the definition of the Laplacian...

Outline

Word Vectors



Word Embedding

- Map from the set of words to $\mathbb{R}^d$.
- Each word is associated to a vector.
- Hope that the relationship between two vectors is related to the relationship between the corresponding words!

Source: nlp.stanford.edu

Word And Context

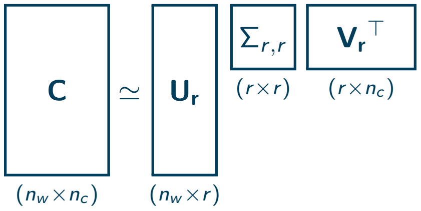
Look ! $\boxed{\text{A}}$ $\boxed{\text{single}}$ $\boxed{\textbf{word}}$ $\boxed{\text{and}}$ $\boxed{\text{its}}$ context

Word and Context

- **Idea:** characterize a word w through its relation with its context c...
- **Probabilistic description**:
  - Joint distribution: $f(w,c) = \mathbb{P}(w,c)$
  - Conditional distribution(s): $f(w,c) = \mathbb{P}(w|c)$ or $f(w,c) = \mathbb{P}(c|w)$.
  - Pointwise mutual information: $f(w,c) = \mathbb{P}(w,c)/(\mathbb{P}(w)\mathbb{P}(c))$
- Word w characterized by the vector $C_w = (f(w,c))_c$ or $C_w = (\log f(w,c))_c$.

- In practice, C is replaced by an estimate on large corpus.
- Very high dimensional model!

A (Naive) SVD Approach



Truncated SVD Approach

- Approximate the code matrix C using the truncated SVD decomposition (best low rank approximation).
- Use as a code

$$C'_w = U_{r,w}\Sigma^{\alpha}_{r,r}$$

with $\alpha \in [0, 1]$.

- Variation possible on C.
- State of the art results but computationally intensive...

A Least Square Approach

- All the previous models corresponds to
$$-log\mathbb{P}\left(w,c\right) \sim C'^t_w C''_c + \alpha_w + \beta_c$$

GloVe (Global Vectors)

- Enforce such a fit through a (weighted) least square formulation:
$$\sum_{w,c} h(\mathbb{P}\left(w,c\right)) \left\| -log\mathbb{P}\left(w,c\right) - \left(C'^t_w C''_c + \alpha_w + \beta_c\right) \right\|^2$$
with h a increasing weight.
- Minimization by alternating least square...

- Much more efficient than SVD.

A Learning Approach

Supervised Learning Formulation

- Couples (w, c) are positive examples.
- Artificially generate negative examples (w', c') (for instance by copying c and generating w' independently of c.)
- Model the probability of being positive given (w, c) as a (simple) function of the codes C'_w and C''_c

- Word2vec: logistic modeling
$$\mathbb{P}\left(1|w, c\right) = \frac{e^{C'^t_W C''_c}}{1 + e^{C'^t_W C''_c}}$$
- State of the art and efficient computation.
- Similar to a factorization of $-\log(\mathbb{P}\left(w, c\right) / (\mathbb{P}\left(w\right) \mathbb{P}\left(c\right)))$!

Outline

Clustering

- **Training data** : $\mathcal{D} = \{\underline{X}_1, \dots, \underline{X}_n\} \in \mathcal{X}^n$ (i.i.d. $\sim \mathbb{P}$)
- Latent groups?

Clustering

- Construct a map f from $\mathcal{D}$ to $\{1, \dots, K\}$ where K is a number of classes to be fixed:
$$f : \quad \underline{X}_i \mapsto k_i$$

Motivations

- Interpretation of the groups
- Use of the groups in further processing

- Several strategies possible!
- Can use dimension reduction as a preprocessing.

Outline

Partition based

Partition Heuristic

- Clustering is defined by a partition in K classes...
- that minimizes a homogeneity criterion.

K- Means

- Cluster k defined by a *center* μ_k.
- Each sample is associated to the closest center.
- Centers defined as the minimizer of $\sum_{i=1}^{n} \min_k \|\underline{X}_i - \mu_k\|^2$
- Iterative scheme (Loyd):
  - Start by a (pseudo) random choice for the centers μ_k
  - Assign each samples to its nearby center
  - Replace the center of a cluster by the mean on its assigned samples.
  - Repeat the last two steps until convergence.

Partition based

Partition based

- Other schemes:
  - McQueen: modify the mean each time a sample is assigned to a new cluster.
  - Hartigan: modify the mean by removing the considered sample, assign it to the nearby center and recompute the new mean after assignment.
- A good initialization is crucial!
  - Initialize by samples.
  - k-Mean++: try to take them as separated as possible.
  - No guarantee to converge to a global optimum: repeat and keep the best result!
- Complexity : $O(n \times K \times T)$ where T is the number of step in the algorithm.

Partition based

- k-Medoid: use a sample as a center
  - PAM: for a given cluster, use the sample that minimizes the intra distance (sum of the squared distance to the other points)
  - Approximate medoid: for a given cluster, assign the point that is the closest to the mean.
- Complexity:
  - PAM: $O(n^2 \times T)$ in the worst case!
  - Approximate medoid: $O(n \times K \times T)$ where T is the number of step in the algorithm.
- **Remark:** Any distance can be used...

Outline

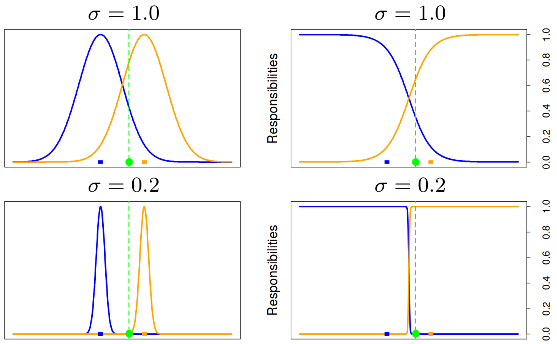
Model Based

Model Heuristic

- Use a generative model of the data:
$$\mathbb{P}\left(\underline{X}\right) = \sum_{k=1}^{K} \pi_k \mathbb{P}_{\theta_k}\left(\underline{X}|k\right)$$
where π_k are proportions and $\mathbb{P}_\theta\left(\underline{X}|k\right)$ are parametric probability models.
- Estimate those parameters (often by a ML principle).
- Assign each observations to the class maximizing the a posteriori probability (obtained by Bayes formula)
$$\frac{\widehat{\pi_k}\mathbb{P}_{\widehat{\theta_k}}\left(\underline{X}|k\right)}{\sum_{k'=1}^{K} \widehat{\pi_{k'}}\mathbb{P}_{\widehat{\theta_{k'}}}\left(\underline{X}|k'\right)}$$

- Link with Generative model in supervised classification!

Model Based



A two class example

- A mixture $\pi_1 f_1(\underline{X}) + \pi_2 f_2(\underline{X})$
- and the posterior probability $\pi_i f_i(\underline{X})/(\pi_1 f_1(\underline{X}) + \pi_2 f_2(\underline{X}))$
- Natural class assignment!

Model Based

Sub-population estimation

- A mixture $\pi_1 f_1(\underline{X}) + \pi_2 f_2(\underline{X})$
- Two populations with a parametric distribution f_i.
- Most classical choice: Gaussian distribution

Gaussian Setting

- $\underline{X}_1, \dots, \underline{X}_n$ independent
- $\underline{X}_i \sim \mathcal{N}(\mu_1, \sigma_1^2)$ with probability π_1 or $\underline{X}_i \sim \mathcal{N}(\mu_2, \sigma_2^2)$ with probability π_2
- We don't know the parameters μ_i, σ_i, π_i.
- We don't know from which distribution each $\underline{X}_i$ has been drawn.

Model Based

Maximum Likelihood

- Density: $\pi_1 \Phi(\underline{X}, \mu_1, \sigma_1^2) + \pi_2 \Phi(\underline{X}, \mu_2, \sigma_2^2)$
- log-likelihood:
$$\mathcal{L}(\theta) = \sum_{i=1}^{n} \log \left(\pi_1 \Phi(\underline{X}_i, \mu_1, \sigma_1^2) + \pi_2 \Phi(\underline{X}_i, \mu_2, \sigma_2^2) \right)$$
- No straightforward way to optimize the parameters!

What if algorithm

- Assume we know from which distribution each sample has been sampled: $Z_i = 1$ if from f_1 and $Z_i = 0$ otherwise.
- log-likelihood:
$$\sum_{i=1}^{n} Z_i \log \; \Phi(\underline{X}_i, \mu_1, \sigma_1^2) + (1 - Z_i) \log \Phi(\underline{X}_i, \mu_2, \sigma_2^2)$$
- Easy optimization
- but the Z_i are unknown...

Model Based

What if algorithm

- Assume we know from which distribution each sample has been sampled: $Z_i = 1$ if from f_1 and $Z_i = 0$ otherwise.
- log-likelihood:
$$\sum_{i=1}^{n} Z_i \log \ \Phi(\underline{X}_i, \mu_1, \sigma_1^2) + (1 - Z_i) \log \Phi(\underline{X}_i, \mu_2, \sigma_2^2)$$
- Easy optimization
- but the Z_i are unknown...

Bootstrapping Idea

- Replace Z_i by its expectaion given the current estimate.
- $\mathbb{E}\left[Z_i\right] = \mathbb{P}\left(Z_i = 1|\theta\right)$ (A posteriori probability)
- and iterate...
- Can be proved to be good idea!

Model Based

EM Algorithm

- (Random) initialization: μ_i^0, σ_i^0, π_i^0.
- Repeat:
  - Expectation (Current a posteriori probability):

$$\mathbb{E}_t\left[Z_i\right] = \mathbb{P}\left(Z_i = 1 | \theta^t\right) = \frac{\pi_1^t \Phi(\underline{X}_i, \mu_1^t, (\sigma_1^t)^2)}{\pi_1^t \Phi(\underline{X}_i, \mu_1^t, (\sigma_1^t)^2) + \pi_2^t \Phi(\underline{X}_i, \mu_2^t, (\sigma_2^t)^2)}$$

  - Maximization of

$$\sum_{i=1}^{n} \mathbb{E}_t\left[Z_i\right] \log\ \Phi(\underline{X}_i, \mu_1, \sigma_1^2) + \mathbb{E}_t\left[1 - Z_i\right] \log \Phi(\underline{X}_i, \mu_2, \sigma_2^2)$$

  to obtain μ_i^{t+1}, σ_i^{t+1}, π_i^{t+1}.

Model Based

- Large choice of parametric models.

Gaussian Mixture Model

- Use

$$\mathbb{P}_{\theta_k}\left(\underline{\vec{X}}|k\right) \sim \mathcal{N}(\mu_k, \Sigma_k)$$

with $\mathcal{N}(\mu, \Sigma)$ the Gaussian law of mean μ and covariance matrix Σ.

- Efficient optimization algorithm available (EM)
- Often some constrain on the covariance matrices: identical, with a similar structure...
- Strong connection with K-means when the covariance matrices are assumed to be the same multiple of the identity.

Model Based

Probabilistic latent semantic analysis (PLSA)

- Documents described by their word counts w
- Model:
$$\mathbb{P}\left(w\right) = \sum_{k=1}^{K} \mathbb{P}\left(k\right) \mathbb{P}_{\theta_k}\left(w|k\right)$$
with k the (hidden) topic, $\mathbb{P}\left(k\right)$ a topic probability and $\mathbb{P}\left(w|k\right)$ a multinomial law for a given topic.
- Clustering according to
$$\mathbb{P}\left(k|w\right) = \frac{\widehat{\mathbb{P}\left(k\right)}\mathbb{P}_{\widehat{\theta_k}}\left(w|k\right)}{\sum_{k'} \widehat{\mathbb{P}\left(k'\right)}\mathbb{P}_{\widehat{\theta_{k'}}}\left(w|k'\right)}$$

- Same idea than GMM!
- Bayesian variant called LDA.

Model Based

Parametric Density Estimation Principle

- Assign a probability of membership.
- Lots of theoretical studies…
- Model selection principle can be used to select K the number of class:
  - AIC / BIC /MDL penalization
  - Cross Validation is also possible!

- Complexity: $O(n \times K \times T)$

Outline

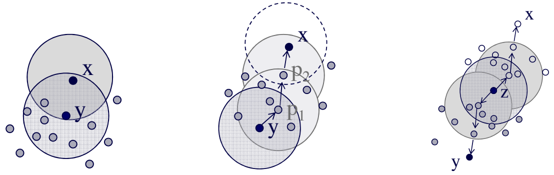
(Non Parametric) Density Based

Density Heuristic

- Cluster are connected dense zone separated by low density zone.
- Not all points belong to a cluster.

- Basic bricks:
  - Estimate the density.
  - Find points with high densities.
  - Gather those points according to the density
- Density estimation:
  - Classical kernel density estimate...
- Gathering:
  - Link points of high density and use the resulted component.
  - Move them toward top of density *hill* by following the gradient and gather all the points arriving at the same *summit*.

(Non Parametric) Density Based



- Examples:
  - DBSCAN: link point of high densities using a very simple kernel.
  - PdfCLuster: find connected zone of high density.
  - Mean-shift: move points toward top of density *hill* following an evolving kernel density estimate.
- Complexity: $O(n^2 \times T)$ in the worst case.
- Can be reduced to $O(n \log(n) T)$ if samples can be encoded in a tree structure (n-body problem type approximation).

Outline

Agglomerative Clustering

Agglomerative Clustering Heuristic

- Start with very small clusters (a sample by cluster?)
- Sequential merging of the most similar clusters...
- according to some *greedy* criterion Δ.

- Generates a hierarchy of clustering instead of a single one.
- Need to select the number of cluster afterwards.
- Several choice for the merging criterion...
- Examples:
  - Minimum Linkage: merge the closest cluster in term of the usual distance
  - Ward's criterion: merge the two clusters yielding the less inner inertia loss (k-means criterion)

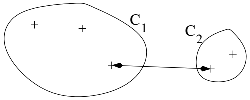
Agglomerative Clustering

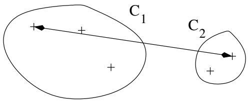
Algorithm

- Start with $(\mathcal{C}_i^{(0)}) = (\{\underline{X}_i\})$ the collection of all singletons.
- At step s, we have $n - s$ clusters $(\mathcal{C}_i^{(s)})$:
    - Find the two most similar clusters according to a criterion Δ:
      $$(i, i') = \operatorname*{argmin}_{(j,j')} \Delta(\mathcal{C}_j^{(s)}, \mathcal{C}_{j'}^{(s)})$$
    - Merge $\mathcal{C}_i^{(s)}$ and $\mathcal{C}_{i'}^{(s)}$ into $\mathcal{C}_i^{(s+1)}$
    - Keep the $n - s - 2$ other clusters $\mathcal{C}_{i''}^{(s+1)} = \mathcal{C}_{i''}^{(s)}$
- Repeat until there is only one cluster.

- Complexity: $O(n^3)$ in general.
- Can be reduced to $O(n^2)$
    - if only a bounded number of merging is possible for a given cluster,
    - for the most classical distances by maintaining a nearest neighbors list.

Agglomerative Clustering





Merging criterion based on the distance between points

- Minimum linkage:

$$\Delta(\mathcal{C}_i,\mathcal{C}_j) = \min_{\underline{X}_i\in\mathcal{C}_i}\ \min_{\underline{X}_{\in}\mathcal{C}_j} d(\underline{X}_i,\underline{X}_j)$$

- Maximum linkage:

$$\Delta(\mathcal{C}_i,\mathcal{C}_j) = \max_{\underline{X}_i\in\mathcal{C}_i}\ \max_{\underline{X}_{\in}\mathcal{C}_j} d(\underline{X}_i,\underline{X}_j)$$

- Average linkage:

$$\Delta(\mathcal{C}_i,\mathcal{C}_j) = \frac{1}{|\mathcal{C}_i||\mathcal{C}_j|}\sum_{\underline{X}_i\in\mathcal{C}_i}\ \sum_{\underline{X}_{\in}\mathcal{C}_j} d(\underline{X}_i,\underline{X}_j)$$

- Clustering based on the proximity...

Source: E. Matzner-Löber

Agglomerative Clustering

Merging criterion based on the inertia (distance to the mean)

- Ward's criterion:

$$\Delta(\mathcal{C}_i, \mathcal{C}_j) = \sum_{\underline{X}_i \in \mathcal{C}_i} \left(d^2(\underline{X}_i, \mu_{\mathcal{C}_i \cup \mathcal{C}_j}) - d^2(\underline{X}_i, \mu_{\mathcal{C}_i}) \right) + \sum_{\underline{X}_j \in \mathcal{C}_j} \left(d^2(\underline{X}_j, \mu_{\mathcal{C}_i \cup \mathcal{C}_j}) - d^2(\underline{X}_j, \mu_{\mathcal{C}_j}) \right)$$

- If d is the euclidean distance:

$$\Delta(\mathcal{C}_i, \mathcal{C}_j) = \frac{2|\mathcal{C}_i||\mathcal{C}_j|}{|\mathcal{C}_i| + |\mathcal{C}_j|} d^2(\mu_{\mathcal{C}_i}, \mu_{\mathcal{C}_j})$$

- Same criterion than in the k-means algorithm but greedy optimization.

Outline

Grid based

Grid heuristic

- Split the space in pieces
- Group those of high density according to their proximity

- Similar to density based estimate (with partition based initial clustering)
- Space splitting can be fixed or adaptive to the data.
- Examples:
  - STING (Statistical Information Grid): Hierarchical tree construction plus DBSCAN type algorithm
  - AMR (Adaptive Mesh Refinement): Adaptive tree refinement plus k-means type assignment from high density leaves.
  - CLIQUE: Tensorial grid and 1D detection.
- Linked to Divisive clustering (DIANA)

Others

Graph based

- Spectral clustering: dimension reduction + k-means.
- Message passing: iterative local algorithm.
- Graph cut: min/max flow.

- Kohonen Map,
- ...

Outline

Scalability

Large dataset issue

- When n is large, a $O(n^{\alpha} \log n)$ with $\alpha > 1$ is not acceptable!
- How to deal with such a situation?

- **Beware:** Computing all the pairwise distance requires $O(n^2)$ operations!

Ideas

- Sampling
- Online processing
- Simplification
- Parallelization

Sampling

Sampling heuristic

- Use only a subsample to construct the clustering.
- Assign the other points to the constructed clusters afterwards.

- Requires a clustering method that can assign new points (partition, model...)
- Often repetition and choice of the best clustering
- Example:
  - CLARA: K-medoid with sampling and repetition
- Two step algorithm:
  - Generate a large number n' of clusters using a fast algorithm (with $n' \ll n$)
  - Cluster the clusters with a more accurate algorithm.

Online

Online heuristic

- Modify the current clusters according to the value of a single observation.

- Requires compactly described clusters.
- Examples:
  - Add to an existing cluster (and modify it) if it is close enough and create a new cluster otherwise (k-means without reassignment)
  - Stochastic descent gradient (GMM)
- May leads to far from optimal clustering.

Simplification

Simplification heuristic

- Simplify the algorithm to be more efficient at the cost of some precision.

- Algorithm dependent!
- Examples:
  - Replace groups of observation (preliminary cluster) by the (approximate) statistics.
  - Approximate the distances by cheaper ones.
  - Use n-body type techniques.

Parallelization

Parallelization heuristic

- Split the computation on several computers.

- Algorithm dependent!
- Examples:
  - Distance computation in k-means, parameter gradient in model based clustering
  - Grid density estimation, Space splitting strategies
- Classical batch sampling not easy to perform as partitions are not easily merged...

Outline

Generative Modeling and Density Estimation

Generative Model

- Probabilistic model of the world.
- Allow to *generate* samples that mimics $\underline{X}$.
- Classical approaches are based on likelihood:
    - Parametric model,
    - Bayesian model.

Generative Algorithm

- Computational probabilistic model of the world.
- Allow to *generate* samples $G(Z)$ that mimic $\underline{X}$ from
    - a randomness source Z,
    - a computable function G.
- No explicit form of the likelihood!

- How to learn G?

A Clever Idea

$$G(Z) \sim \quad \underline{X} \; ?$$

- From estimation to...

A Clever Idea

$$\Phi(G(Z)) \sim \Phi(\underline{X})?$$

- From estimation to... discrimination

Discriminator (Goodfellow 14)

- Let
$$(\underline{\tilde{X}}, Y) = \begin{cases} (X, 1) & \text{with probability } 1/2 \\ (G(Z), 0) & \text{with probability } 1/2 \end{cases}$$
- Can we guess from $\underline{\tilde{X}}$ whether it comes from $\underline{X}$ or $G(Z)$?
- Discriminator loss = Classifier loss:

$$\mathcal{L}(D, G) = 1/2\mathbb{E}_{\underline{X}}\left[-\log D(\underline{X})\right] + 1/2\mathbb{E}_{G(Z)}\left[-\log(1 - D(G(Z)))\right]$$

Heuristic

- One can learn a discriminator from the data for a fixed G.
- The ideal generator is such that this problem is hard!

A Clever Idea

Best Discriminator

- Bayes Discriminator D^*:

$$D^*(\underline{\tilde{X}}) = \mathbb{P}\left(Y = 1 | \underline{\tilde{X}}\right) = \frac{1/2 f_{\underline{X}}(\underline{\tilde{X}})}{1/2 f_{\underline{X}}(\underline{\tilde{X}}) + 1/2 f_{G(Z)}(\underline{\tilde{X}})}$$

- Optimal loss:

$$\begin{aligned}\mathcal{L}(D^*, G) = 1/2\mathbb{E}_{\underline{X}}&\left[- \log 1/2 + - \log \frac{f_{\underline{X}}(\underline{X})}{1/2 f_{\underline{X}}(\underline{X}) + 1/2 f_{G(Z)}(\underline{X})}\right] \\ &+ 1/2\mathbb{E}_G\left[- \log 1/2 + - \log \frac{f_G(G)}{1/2 f_{\underline{X}}(G) + 1/2 f_G(G)}\right] \\ &= -1/2 KL(f_{\underline{X}}, 1/2 f_{\underline{X}} + 1/2 f_{G(Z)}) \\ &\quad - 1/2 KL(f_{G(Z)}, 1/2 f_{\underline{X}} + 1/2 f_{G(Z)}) + \log 2 \\ &= -JKL_{1/2}(f_{\underline{X}}, f_{G(Z)}) + \log 2\end{aligned}$$

- Adversarial minimization:

$$\operatorname*{argmax}_G \min_D \mathcal{L}(D, G) = \operatorname*{argmin}_G JKL_{1/2}(f_{\underline{X}}, f_{G(Z)})$$

Generative Adversarial Network

$$G^* = \underset{G}{\operatorname{argmin}} \max_{D} \left[1/2\mathbb{E}_{\underline{X}} \left[\log D(\underline{X}) \right] + 1/2\mathbb{E}_{G(Z)} \left[\log(1 - D(G(Z))) \right] \right]$$

Generative Adversarial Network

- Replace the set of all possible G and D by a set of parametric functions, for instance some deep neural networks
- Replace the expectations by some empirical means.
- Alternate a maximization on D and a minimization on G.

- Z is often $\mathcal{U}[-1,1]$ or $\mathcal{N}(0,1)$.
- Not that easy to train:
  - hard to achieve Nash equilibrium (no guaranteed convergence)
  - mode collapse (restart required)
  - support issue of KL like divergence (add noise)
  - adding feature matching helps!

GAN and f-divergence

$$D_f(P,Q) = \int f\left(\frac{p(x)}{q(x)}\right) q(x)$$
$$= sup_T \mathbb{E}_{\underline{X}\sim P}\left[T(\underline{X})\right] - \mathbb{E}_{G\sim Q}\left[f^*(T(G))\right]$$

f-divergence and dual representation

- Defines a divergence for any convex f.
- Dual representation with $f^*(x) = \sup_u \langle x, u\rangle - f(u)$

$$\min_G \sup_T \mathbb{E}_{\underline{X}\sim P}\left[T(\underline{X})\right] - \mathbb{E}_Z\left[f^*(T(G(Z)))\right]$$

f-GAN

- Replace the set of all possible G and T by a set of parametric functions, for instance some deep neural networks
- Replace the expectations by some empirical means.
- Alternate a maximization on D and a minimization on G.

Classical GAN and f-GAN

$$JKL(P,Q) = sup_T \mathbb{E}_{\underline{X}\sim P}\left[T(\underline{X})\right] - \mathbb{E}_{G\sim Q}\left[-\log(2 - exp\, T(G))\right]$$

Classical GAN as a f-GAN

- JKL-divergence is a f divergence with $f(u) = -(u+1)\log\frac{1+u}{2} + u\log u$.
- Parameterize T by $\log 2 - \log(1+e^{-T'})$ so that

$$\begin{aligned}JKL(P,Q) = &\sup_{T'} \mathbb{E}_{\underline{X}\sim P}\left[\log 2 - \log(1+e^{-T'})\right]\\ &- \mathbb{E}_{G\sim Q}\left[\log(2 - 2/(1+e^{-T'})\right]\\ = &\, 2\log 2 + \sup_{T'} \mathbb{E}_{\underline{X}\sim P}\left[\log(1/(1+e^{-T'}))\right]\\ &+ \mathbb{E}_{G\sim Q}\left[\log(1 - 1/(1+e^{-T'}))\right]\end{aligned}$$

- GAN formulation up to the parameterization of T:

$$\begin{aligned}\min_G \max_{T'} &\, \mathbb{E}_{\underline{X}}\left[\log(1/(1+e^{-T'(\underline{X})}))\right]\\ &+ \mathbb{E}_{G(Z)}\left[\log(1 - 1/(1+e^{-T'(G(Z))}))\right]\end{aligned}$$

GAN and Wasserstein

$$\begin{aligned} W(P,Q) &= \inf_{\xi\pi(P,Q)} \mathbb{E}_{(p,q)\sim\xi}\left[\|p-q\|\right] \\ &= \frac{1}{K} sup_{\|f\|_L \leq K} \mathbb{E}_{\underline{X}\sim P}\left[f(\underline{X})\right] - \mathbb{E}_{G\sim Q}\left[f(G)\right] \end{aligned}$$

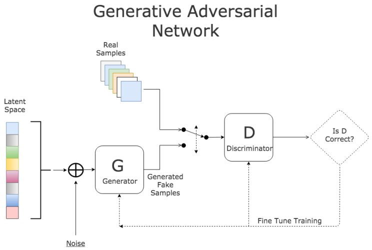
$$\min_G \sup_{\|f\|_L\leq 1} \mathbb{E}_{\underline{X}\sim P}\left[f(\underline{X})\right] - \mathbb{E}_Z\left[f(G(Z))\right]$$

WGAN

- Replace the set of all possible G and f by a set of parametric functions, for instance some deep neural networks
- Replace the expectations by some empirical means.
- Alternate a maximization on D and a minimization on G.

- Constrain on the Lipschitz norm is the most complex part:
  - clip on the network weights
  - or penalization of the gradient norm
- **Rk:** More a case of integral probability metric than optimal transport...

GAN



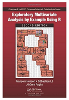
Generative Adversial Network

- Clever idea combined with state of the art NN architecture.
- Impressive results!

- Can it be used to perform clustering in the latent space?

Outline

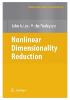
References


Exploratory Multivariate Analysis by Example Using R

SECOND EDITION

François Husson • Sébastien Lê • Jérôme Pagès

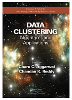
F. Husson, S. Le, and J. Pagès.
Exploratory Multivariate Analysis by Example Using R (2nd ed.)
Chapman and Hall/CRC, 2017


John A. Lee · Michel Verleysen

Nonlinear Dimensionality Reduction

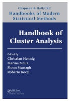
Springer

J. Lee and M. Verleysen.
Nonlinear Dimension Reduction.
Springer, 2009


DATA CLUSTERING Algorithms and Applications

Edited by Charu C. Aggarwal Chandan K. Reddy

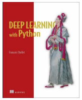
Ch. Aggarwal and Ch. Reddy.
Data Clustering: Algorithms and Applications.
Chapman and Hall/CRC, 2013


Chapman & Hall/CRC Handbooks of Modern Statistical Methods

Handbook of Cluster Analysis

Edited by Christian Hennig Marina Meila Fionn Murtagh Roberto Rocci

Ch. Hennig, M. Meila, F. Murtagh, and R. Rocci.
Handbook of Cluster Analysis.
Chapman and Hall/CRC, 2015


DEEP LEARNING with Python

François Chollet

F. Chollet.
Deep Learning with Python.
Manning, 2017

Licence and Contributors

Contributors

- S. Boucheron, A.K. Fermin, S. Gaiffas, A. Guilloux, Ch. Keribin, E. Le Pennec, E. Matzner, E. Scornet and X Exed team.