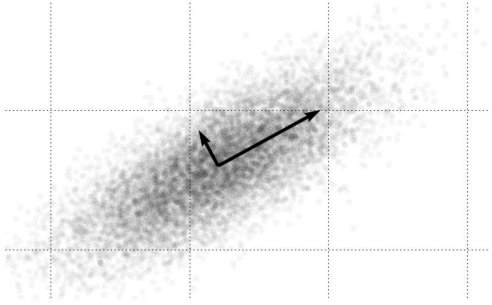
ETH zürich

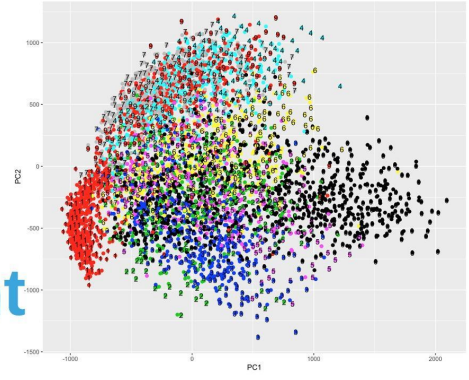
Seminar on Statistical Learning with Sparsity: Sparse Multivariate Methods

Gregor Bachmann and Emil Kr. Feet

Overview

- Principal Component Analysis and its Sparse Variants
- Autoencoders and some Extensions
- Hierarchical Clustering
- K-Means
- Convex Clustering





Principle Component Analysis

$$X = \underbrace{\begin{pmatrix} & \\ & \\ & \end{pmatrix}}_{X \in \mathbb{R}^{n \times p}} = \underbrace{\begin{pmatrix} | & & | \\ u_1 & & u_n \\ | & & | \end{pmatrix}}_{U \in \mathbb{R}^{n \times n}} \underbrace{\begin{pmatrix} D_{11} & & & \\ & D_{22} & & \\ & & \dots & \\ & & & D_{pp} \\ 0 & & & \end{pmatrix}}_{D \in \mathbb{R}^{n \times p}} \underbrace{\begin{pmatrix} - & v_1 & - & - \\ & & & \\ - & v_p & - & - \end{pmatrix}}_{V^t \in \mathbb{R}^{p \times p}} \cdot \Rightarrow \tilde{X} = \underbrace{\begin{pmatrix} | & & | \\ u_1 & & u_q \\ | & & | \end{pmatrix}}_{\tilde{U} \in \mathbb{R}^{n \times q}} \underbrace{\begin{pmatrix} D_{11} & & \\ & \dots & \\ & & D_{qq} \end{pmatrix}}_{\tilde{D} \in \mathbb{R}^{q \times q}} \underbrace{\begin{pmatrix} - & v_1 & - & - \\ & & & \\ - & v_q & - & - \end{pmatrix}}_{\tilde{V}^t \in \mathbb{R}^{q \times p}} \cdot$$

Data and Preprocessing

- **Data**: We are given a data matrix $X \in \mathbb{R}^{N \times p}$ consisting of N observations with p-dimensional features ($p \leq N$):

$$X = \begin{bmatrix} - & - & x_1^T & - & - \\ & & \vdots & & \\ - & - & x_N^T & - & - \end{bmatrix}$$

  where each vector $x_i \in \mathbb{R}^p$ corresponds to one observation.

- **Preprocessing**: Center and scale each column to variance one.

Central Ideas of PCA

- **Goal**: Find lower dimensional representation with as much information as possible.
- **Special Case**: 2d embedding for visualization.
- Do this in a **linear fashion** by using linear combinations of the predictors to form new ones.
- Also allow for **reconstruction** of the original data.

First Approach: Maximize Variance

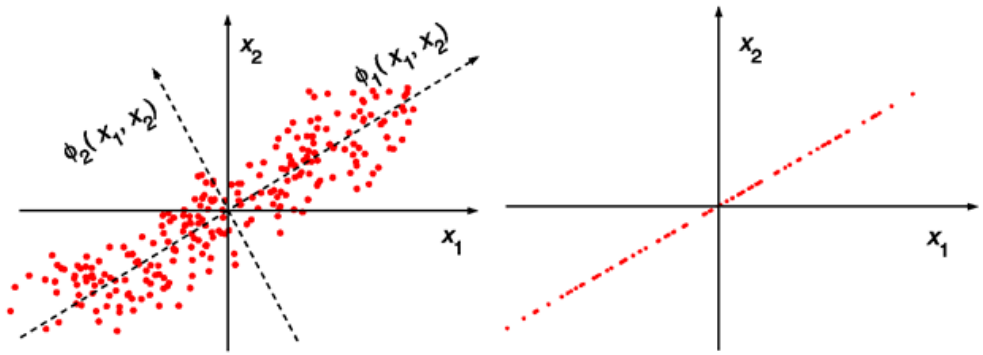
- Use **sample variance** in the data as measure of information.
- Find direction v_1 that maximizes spread in the data:

$$v_1 = \underset{w:||w||_2=1}{\text{argmax}} \hat{Var}(Xw) = \underset{w:||w||_2=1}{\text{argmax}} \frac{1}{N} w^T X^T X w$$

- Turns out that v_1 is given by the eigenvector of $\frac{1}{N} X^T X$ corresponding to the largest eigenvalue.
- **Note**: This is not the new feature yet, only the direction!

Finding the resulting feature

- Use now this best direction to project the data into a one dimensional space.
- Concretely, our resulting 1d embedding is given by $\hat{z} = Xv_1 \in \mathbb{R}^{N \times 1}$



(a) PCA basis (b) PCA reduction to 1D

Figure: Projection of 2D data to 1D with PCA, taken from **what-when-how.com**

Find more directions

- Extend this idea by iteratively finding more directions.
- Find now the second best vector v_2 orthogonal to the previous one:

$$v_2 = \underset{w:||w||_2=1 \text{and} v_1^T w=0}{\text{argmax}} \frac{1}{N} w^T X^T X w$$

- Again, v_2 is given by the eigenvector corresponding to the second largest eigenvalue.
- Continuing like this gives us a set of directions $v_1, \ldots v_r$ with $r \leq p$, also called **loading vectors**.

Find more features

- Arrange now all directions $v_1, \ldots v_r$ as columns in a matrix $V_r \in \mathbb{R}^{p \times r}$, our projection matrix.
- To now find our r-dimensional embedding we simply calculate $Z = XV_r \in \mathbb{R}^{N \times r}$, also called the **principal components**.
- **Note**: If one chooses $r = p$, we get a rotation of the data that aligns the axes with the directions of biggest variance.
- Turns out that this iterative search is equivalent to solving the joint problem: $V_r = \text{argmax}_{A:A^TA=1_r} \, trace(A^T X^T XA)$

Reconstruction of data

- In order to **approximate** the original, calculate $\hat{X} = ZV_r^T \in \mathbb{R}^{N \times p}$.
- Again note that when one uses all eigenvectors for V, we obtain a loss-free solution.
- Make sure you understood distinction between **embedding** $Z \in \mathbb{R}^{N \times r}$ and the **reconstruction** $\hat{X} \in \mathbb{R}^{N \times p}$.

Alternative way of obtaining solutions

- Apply **SVD**: $X = UDV^T$ where $U \in \mathbb{R}^{n \times n}$, $D \in \mathbb{R}^{n \times p}$ and $V \in \mathbb{R}^{p \times p}$
- **Truncate** all the matrices to r rows or columns for reconstruction:

$$\hat{X} = \begin{bmatrix} | & & | \\ u_1 & - & u_r \\ | & & | \end{bmatrix} \begin{bmatrix} D_{11} & & \\ & \ddots & \\ & & D_{rr} \end{bmatrix} \begin{bmatrix} - & v_1^T & - \\ & \vdots & \\ - & v_r^T & - \end{bmatrix} = \tilde{U}\tilde{D}\tilde{V}^T$$

  with $\tilde{U} \in \mathbb{R}^{n \times r}$, $\tilde{D} \in \mathbb{R}^{r \times r}$ and $\tilde{V} \in \mathbb{R}^{p \times r}$.
- The projection is given by $Z = \tilde{U}\tilde{D}$.

Alternative View on PCA

- View PCA as an **optimal reconstruction** algorithm.
- Consider the minimization problem of the reconstruction error:

$$\underset{A_r \in \mathbb{R}^{p \times r} \text{ s.t. } A_r^T A_r = 1_r}{\operatorname{argmin}} \sum_{i=1}^{N} \left\| x_i - \hat{x}_i(A_r) \right\|_2^2 =$$

$$\underset{A_r \in \mathbb{R}^{p \times r} \text{ s.t. } A_r^T A_r = 1_r}{\operatorname{argmin}} \sum_{i=1}^{N} \left\| x_i - A_r A_r^T x_i \right\|_2^2$$

- Again the solution is taking the r eigenvectors corresponding to the r largest eigenvalues.

Problem of PCA

- As soon as $p >> N$, PCA can run into problems.
- Sample covariance $\hat{Cov}(X)$ is **not converging** to the population covariance when N and p grow on the same scale. As a result the eigenvectors can be wrong as well.
- To fix this, we need to regularize and impose **sparseness** on the loading vectors $v_1, ... v_r$.
- Usually no way to interpret resulting axes/features of PCA. With sparseness this gets easier.

Sparse PCA: First Optimization Task

- Focus on first principle component.
- **First idea**:

$$\underset{v:\|v\|_2=1}{\text{argmax}}\, v^T X^T X v \text{ subject to } \|v\|_0 \leq t$$

- Objective is not convex in two ways...
- **Relax** the objective to so called **SCoTCLASS** procedure:

$$\underset{v:\|v\|_2=1}{\text{argmax}}\, v^T X^T X v \text{ subject to} \|v\|_1 \leq t$$

Reformulation: Penalized Matrix Criterion

- Use that PCA is closely related to SVD and apply the **penalized matrix criterion** that we saw last week:

$$\underset{\|u\|_2=\|v\|_2=1}{\text{argmax}} \; u^T X v \text{ subject to } \|v\|_1 \leq t$$

- For fixed v, the optimal u is given by $u = \frac{Xv}{\|Xv\|_2}$
- This cost function is **biconvex** in (u, v) and every solution $\hat{v}$ to this problem is a solution to the SCoTCLASS problem.
- We can apply the same algorithm seen last week to solve the problem, using alternating minimization.

Brief Repetition of Last Week

Algorithm 8.1 ALTERNATING ALGORITHM FOR RANK ONE SPARSE PCA.

1. Initialize $v \in \mathbb{R}^p$ with $\|v\|_2 = 1$.
2. Repeat until changes in $\mathbf{u}$ and v are sufficiently small:
   (a) Update $\mathbf{u} \in \mathbb{R}^N$ via $\mathbf{u} \leftarrow \frac{\mathbf{X}v}{\|\mathbf{X}v\|_2}$.
   (b) Update $v \in \mathbb{R}^p$ via

$$v \leftarrow v(\lambda, \mathbf{u}) = \frac{\mathcal{S}_\lambda(\mathbf{X}^T\mathbf{u})}{\|\mathcal{S}_\lambda(\mathbf{X}^T\mathbf{u})\|_2}, \tag{8.9}$$

where $\lambda = 0$ if $\|\mathbf{X}^T\mathbf{u}\|_1 \leq t$, and otherwise $\lambda > 0$ is chosen such that $\|v(\lambda, \mathbf{u})\|_1 = t$.

Figure: Taken from Trevor Hastie Robert Tibshirani and Martin Wainwright, *Statistical Learning with Sparsity* (2015), page 205

Extending to higher rank

- We saw that in the non-sparse case, a simple iterative method was also solving the high rank problem.
- We can adapt the penalized matrix criterion in the following way:
  - Calculate the first solution (u_1, v_1, d_1) with $d_1 = u_1^T X v_1$
  - Subtract the solution : $X' = X - d_1 u_1 v_1^T$
  - Calculate the next solution (u_2, v_2, d_2) using X'
  - Iterate the procedure r times.
- Does not solve the multirank problem as in PCA case.
- There exists other methods derived from the reconstruction view.

How can we extend PCA?

- We saw that PCA produced new features by linear combinating the old ones.
- Extend this by also allowing for non-linear relationships.
- One could also consider more than one consecutive transformation of the features, leading to more "layers".
- All these extensions are used in the autoencoder.

Architecture of an Autoencoder

- **Input**: One observation $x \in \mathbb{R}^p$
- **Encoder**: Linear activation $W \in \mathbb{R}^{p \times r}$ followed by a non-linear activation function σ:

$$h = \sigma(W^T x)$$

- **Decoder**: Same linear activation $W \in \mathbb{R}^{r \times p}$:

$$\hat{x} = Wh$$

- **In total**: If we denote the autoencoder by f, we can compactly write:

$$\hat{x} = f(x) = W\sigma(W^T x)$$

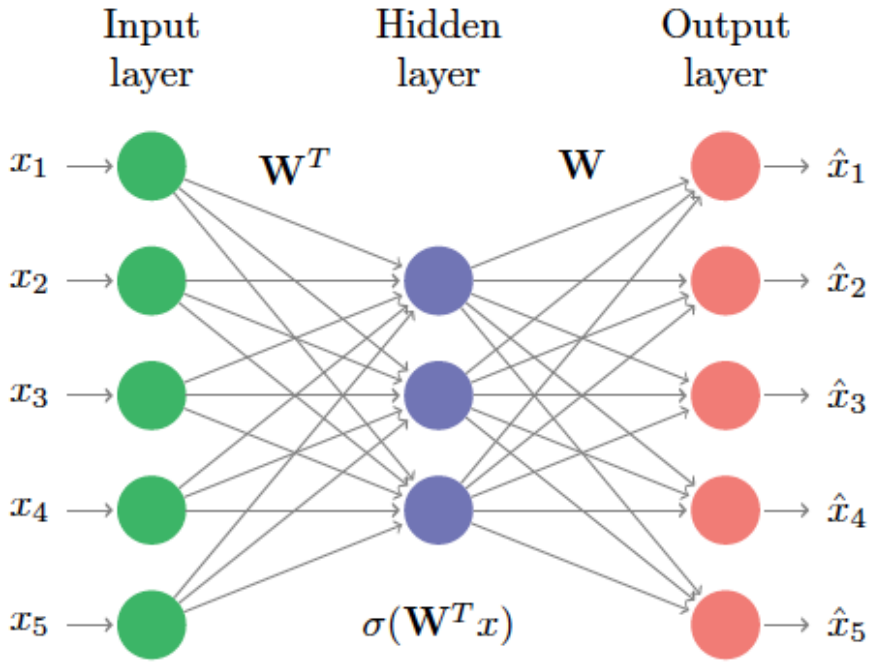


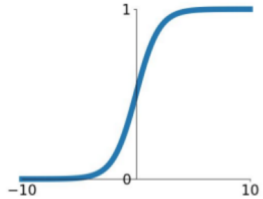
Figure: Architecture of the one hidden layer autoencoder. Taken from Trevor Hastie, Robert Tibshirani and Martin Wainwright, *Statistical Learning with Sparsity* (2015), page 210

Some activation functions

- Taking σ as the identity gives PCA.
- List of others:

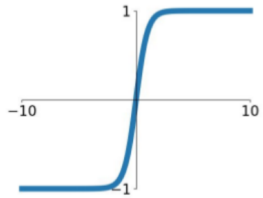
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



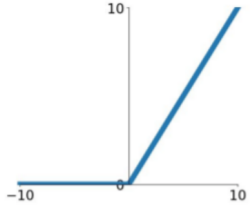
tanh

$$\tanh(x)$$



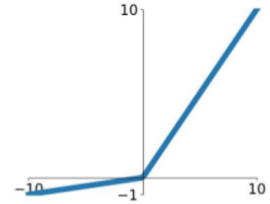
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$



Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

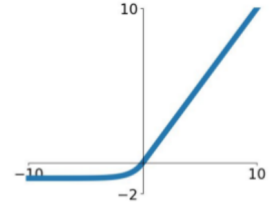


Figure: Taken from **medium.com**

Optimization for Autoencoders

- **Goal**: Reconstruct input as well as possible by choosing good weights W.
- Optimize the **quadratic loss**:

$$\underset{W \in \mathbb{R}^{r \times p}}{\text{minimize}} \frac{1}{2} \sum_{i=1}^{N} \|x_i - \hat{x}_i\|_2^2 = \underset{W \in \mathbb{R}^{r \times p}}{\text{minimize}} \frac{1}{2} \sum_{i=1}^{N} \|x_i - W\sigma(W^T x_i)\|_2^2$$

- This problem is **not convex** but is usually solved by iterative methods such as **stochastic gradient descent**.

Autoencoders as a denoising method

- Instead of feeding the true observations into the autoencoder, add **noise** to them and pass these noisy samples.
- **Trick**: Still evaluate the error comparing to the true observation.
- The idea is that the autoencoder learns how to **denoise** the data.

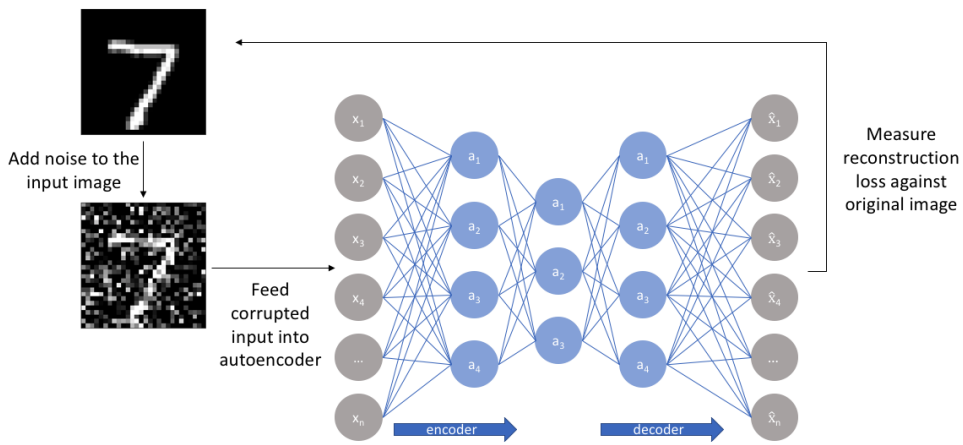


Figure: Illustration of denoising process for MNIST data. Taken from **jeremyjordan.me**

Variational Autoencoder

- Not covered in the book, but I found it so interesting that I will present it here.
- Another extension of the autoencoder, but now with the aim to **generate new images**!!
- The main difference is that we will include **randomness** in the hidden layer to be able to sample new pictures.
- The whole thing gets a bit more complicated, but bear with me (or just enjoy the visual results).
- If you are interested: Check out **towardsdatascience.com** All figures I use are taken from there.

Architecture of Variational Autoencoder

- **Idea**: Let the encoder produce two vectors, a mean vector μ and a variance vector σ^2.
- Create a new **random** vector h by sampling from a normal distribution with mean μ and variance σ^2.
- This h is then passed to the decoder which again tries to produce the original input.

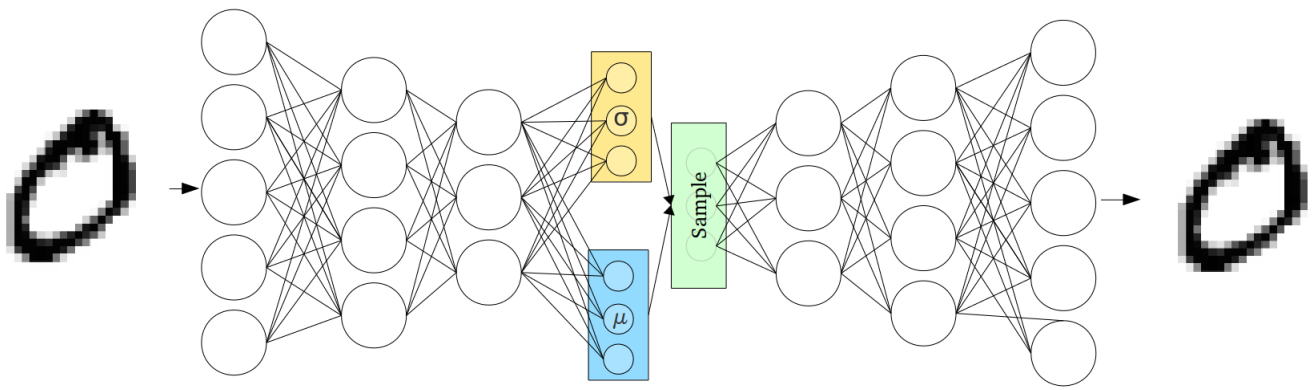


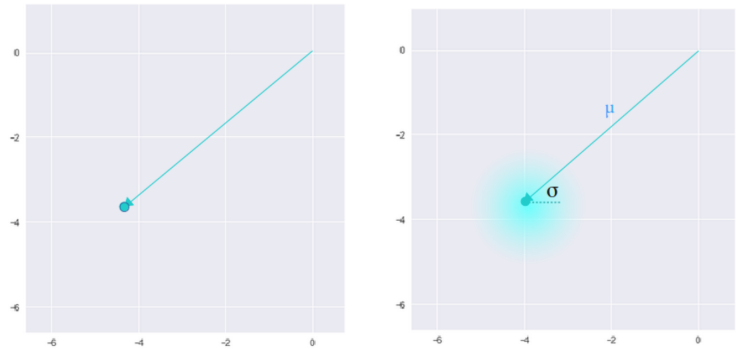
Figure: Architecture of Variational Autoencoder

Interpretation of ideas

- For normal autoencoders, hidden space is very sparsely populated.
- By inducing the randomness we populate the space more (making it more continuous), because we assign the input to different hidden vectors.
- To avoid very distinct clusters in the hidden space the cost function is adapted to force centers to be close to the origin.

Illustration of the ideas

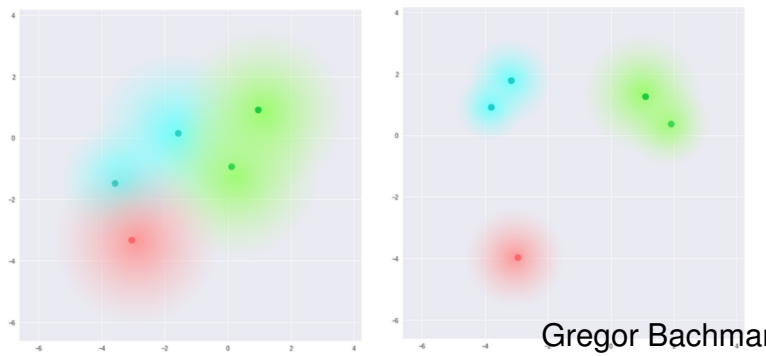
Input gets mapped to a region instead of a single point.



Standard Autoencoder
(direct encoding coordinates)

Variational Autoencoder
(μ and σ initialize a probability distribution)

Enforce clusters to be close together in order to be able to sample from mixed regions for interpolations.



What we require

What we may inadvertently end up with

What is possible with more time, better coding skills and GPU

Figure: Generating Anime images, taken from **mc.ai**

Sparse Clustering

N observations of $X_i \in R^p$ where $p >> N$

- Goal: Group similar observations based on features
- Why do we need sparsity?
  - Not all features are relevant for clustering
- We will consider:
  - Sparse Hierarchical Clustering
  - Sparse K-means
  - Convex clustering

Agglomerative Hierarchical Clustering

$X \in \mathbb{R}^{N \times p}$. With corresponding dissimilarity matrix D, where

$$D_{i,i'} = \sum_{j=1}^{p} d_{i,i',j}$$

- $d_{i,i',j}$ is some dissimilarity measure between observation i and i' on feature j. e.g. Euclidean.

Algorithm:

1. Split observations in to N groups, i.e. each group is a singleton.
2. Measure *similarity* between each group.
3. Join the two which are most *similar*.
4. Repeat step 2-3 until there is only one group containing all observations.

Hierarchical Clustering: measure of similarity

- Complete linkage: $\max_{i \in C_k, i' \in C_{k'}} D_{i,i'}$
  - Good in situations where clusters are compact and close.
- Average linkage: $\frac{1}{|C_k||C_{k'}|} \sum_{i \in C_k, i' \in C_{k'}} D_{i,i'}$
  - generally okay measure.
- Single linkage: $\min_{i \in C_k, i' \in C_{k'}} D_{i,i'}$
  - Good when clusters are stretched, e.g. circles.

Hard to judge which is best in a high dimensional setting.

Hierarchical Clustering

R Example

Sparse Hierarchical Clustering

Problem: High influence of variables that are irrelevant for clustering.

Solution: Let

$$\Delta = \begin{bmatrix} d_{1,1,1} & \cdots & d_{1,1,p} \\ d_{1,2,1} & \cdots & \\ \vdots & \ddots & \vdots \\ d_{N,N,1} & \cdots & d_{N,N,p} \end{bmatrix} \in \mathbb{R}^{N^2,p}$$

Define $\tilde{D} = \Delta w$, which means that $\tilde{D}_{i,i'} = \sum_{j=1}^{p} w_j d_{i,i',j}$. We seek a w such that $\tilde{D}$ "captures" as much of Δ as possible. This can be seen as a sparse PCA problem.

Sparse Hierarchical Clustering: finding *w*

Theorem

Consider data matrix $X \in \mathbb{R}^{n \times p}$ *with* $rank(X) = K \leq min(n, p)$. *Let* $U \in \mathbb{R}^{n \times K}$ *and* $V \in \mathbb{R}^{p \times K}$, *both orthogonal, and D a diagonal matrix with elements* d_k. *Then*

$$\frac{1}{2}||X - UDV^T||_F^2 = \frac{1}{2}||X||_F^2 - \sum_{k=1}^{K} u_k{}^T X v_k + \frac{1}{2}\sum_{k=1}^{K} d_k^2$$

- See Theorem 2.1 in *Witten, D. M., Tibshirani, R., Hastie, T. (2009). A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. Biostatistics (Oxford, England), 10(3), 515-34.*

Sparse Hierarchical Clustering: finding w

$\Delta \in \mathbb{R}^{N^2 \times p}$ is a matrix with column j consisting of N^2 pairwise dissimilarities for feature j.
Using previous theorem we now see that finding w is equivalent with:

$$\begin{aligned} &\textit{maximize}_{u \in \mathbb{R}^{N^2}, w \in \mathbb{R}^p} \{u^T \Delta w\} \\ &\text{subject to } ||u||_2 \leq 1,\ ||w||_2 \leq 1, ||w||_1 \leq s, w \succeq 0 \end{aligned}$$

- $\Delta w = \tilde{D} \in \mathbb{R}^{N^2}$, reshaped to $\tilde{D} \in \mathbb{R}^{N \times N}$. Then apply hierarchical clustering.

Sparse Hiearchical Clustering

R Example

K-means

Goal: Partition observations into K homogeneous groups based on minimizing cluster sum of squares:

$$W(\mathcal{C}) = \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k} ||x_i - \bar{x}_k||_2^2$$

- $\{\bar{x}_k\}_1^K$ *codebook* vectors. $\bar{x_k} = \sum_{i \in \mathcal{C}_k} \frac{x_i}{N_k}$, where $N_k = |\mathcal{C}_k|$.
- $\tau(i)$ encoder which assigns x_i to the cluster of closest centroid. $C_k = \{i \ : \ \tau(i) = k\}$

Algortithm for K-means

Algorithm 14.1 *K-means Clustering.*

1. For a given cluster assignment C, the total cluster variance (14.33) is minimized with respect to $\{m_1, \ldots, m_K\}$ yielding the means of the currently assigned clusters (14.32).

2. Given a current set of means $\{m_1, \ldots, m_K\}$, (14.33) is minimized by assigning each observation to the closest (current) cluster mean. That is,

$$C(i) = \underset{1 \le k \le K}{\operatorname{argmin}} ||x_i - m_k||^2. \qquad (14.34)$$

3. Steps 1 and 2 are iterated until the assignments do not change.

Figure: K-means Algorithm: Taken from *Elements of Statistical Learning*, Hastie et al. (2009), p. 510

Sparse K-means

It holds that

$$\sum_{i,i' \in \mathcal{C}_k} ||x_i - x_{i'}||_2^2 = 2N_k \sum_{i \in \mathcal{C}_k} ||x_i - \bar{x}_k||_2^2.$$

Thus the cost-function we try to minimize is

$$W(\mathcal{C}) = \sum_{k=1}^{K} \frac{1}{N_k} \sum_{i,i' \in \mathcal{C}_k} \sum_{j=1}^{p} d_{i,i',j}.$$

- This allows us to derive K-means for a dissimilarity matrix $D_{i,i'}$ - here Euclidean.
  - Note: Can also derive a similar algorithm for more general dissimilarity matrix $\tilde{D}$. Have to use k-medoids.

Sparse K-means

Again suppose not all variables are important for clustering:

$$\begin{aligned} &\textit{minimize}_{C, w \in \mathbb{R}^p} && \sum_{j=1}^{p} w_j \Big(\sum_{k=1}^{K} \frac{1}{N_k} \sum_{i,i' \in C_k} d_{i,i',j} \Big) \\ &\textit{subject to} && ||w||_2 \leq 1, \; ||w||_1 \leq s \, , \; w \succeq 0 \end{aligned}$$

- problem is convex in w.

Sparse K-means

Again suppose not all variables are important for clustering:

$$\begin{aligned} &\textit{minimize}_{\mathcal{C}, w \in \mathbb{R}^p} && \sum_{j=1}^{p} w_j \Big(\sum_{k=1}^{K} \frac{1}{N_k} \sum_{i,i' \in \mathcal{C}_k} d_{i,i',j} \Big) \\ &\textit{subject to} && ||w||_2 \leq 1, \; ||w||_1 \leq s \, , \; w \succeq 0 \end{aligned}$$

- problem is convex in w.
- However, it has the solution $\hat{w} = 0$.

Sparse K-means

$$\textit{minmize } W(\mathcal{C}) \iff \textit{maximize } B(\mathcal{C})$$

Can be seen from

$$\begin{aligned} T &:= \sum_{i=1}^{N} \sum_{i'=1}^{N} \left(\sum_{j=1}^{p} d_{i,i',j} \right) \\ &= \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k} \left(\sum_{i' \in \mathcal{C}_k} \sum_{j=1}^{p} d_{i,i',j} + \sum_{i' \notin \mathcal{C}_k} \sum_{j=1}^{p} d_{i,i',j} \right) \\ &= W(\mathcal{C})' + B(\mathcal{C})' \end{aligned}$$

Where T (total cost) is constant.

Sparse K-means

This leads us to the optimization problem

$$\begin{aligned} &\textit{maximize}_{\mathcal{C}, w \in \mathbb{R}^p} && \sum_{j=1}^{p} w_j \left(\frac{1}{N} \sum_{i=1}^{N} \sum_{i'=1}^{N} d_{i,i',j} - \sum_{k=1}^{K} \frac{1}{N_k} \sum_{i,i' \in \mathcal{C}_k} d_{i,i',j} \right) \\ &\textit{subject to} && ||w||_2 \leq 1, \; ||w||_1 \leq s \, , \; w \succeq 0 \end{aligned}$$

Iterative scheme:

- Hold $\mathcal{C}$ fixed. Maximize with respect to w.
- Hold w fixed. Optimize with respect to $\mathcal{C}$ (Weighted K-means).

Sparse Clustering: Choosing *s* is not trivial

Algorithm to select tuning parameter *s* for sparse *K*-means

1. Obtain permuted data sets $\mathbf{X}_1, \ldots \mathbf{X}_B$ by independently permuting the observations within each feature.
2. For each candidate tuning parameter value s:
   a. Compute $O(s)=\sum_j w_j(\frac{1}{n}\sum_{i=1}^{n}\sum_{i'=1}^{n} d_{i,i',j} - \sum_{k=1}^{K}\frac{1}{n_k}\sum_{i,i' \in C_k} d_{i,i',j})$, the objective obtained by performing sparse K-means with tuning parameter value s on the data $\mathbf{X}$.
   b. For $b = 1, 2, \ldots B$, compute $O_b(s)$, the objective obtained by performing sparse K-means with tuning parameter value s on the data $\mathbf{X}_b$.
   c. Calculate $\text{Gap}(s)=\log(O(s)) - \frac{1}{B}\sum_{b=1}^{B}\log(O_b(s))$.
3. Choose s^* corresponding to the largest value of Gap(s). Alternatively, one can choose s^* to equal the smallest value for which Gap(s^*) is within a standard deviation of log ($O_b(s^*)$) of the largest value of Gap(s).

Figure: Taken from *Witten, D. M., Tibshirani, R. (2010). A framework for feature selection in clustering. Journal of the American Statistical Association, 105(490), 713-726.*

Convex Clustering

Minimize for $u_i \in \mathbb{R}^p$

$$J(u_1, u_2, ..., u_N) = \frac{1}{2} \sum_{i=1}^{N} ||x_i - u_i||_2^2 + \lambda \sum_{i<i'} w_{i,i'} ||u_i - u_{i'}||_q$$

- Predefined weights: for example $w_{ii'} = \exp(-\phi||x_i - x_{i'}||)$.
- Convex for $q \geq 1$.
- For λ large "enough", many cluster-centers will be equal.

Convex Clustering: Example

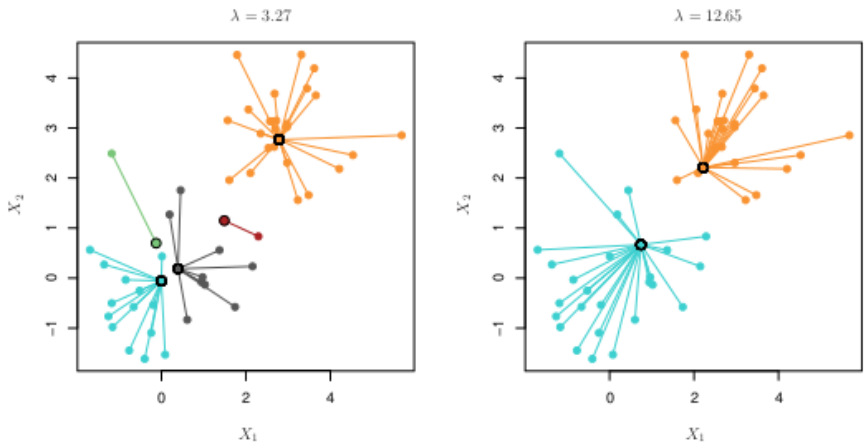


Figure: Taken from *Statistical Learning with Sparsity: The Lasso and Generalizations*, Hastie et al. (2015), p. 232

- Data from two spherical Gaussian populations, separated by three units in each direction.
- Note that cluster prototypes (black circles) cannot be interpreted as cluster centroids.
- Found from a path of 50 lambda values.

Convex Clustering: R-Example

Need package: cvxclustr