# Learning data representation and piece-wise estimation.

Sammy EL GHAZZAL

August, 28th 2012

# Contents

1. Background
2. Learning data representation
3. Kernel extensions

# Motivation

- Unsupervised learning: learning **without a teacher**. No labeled examples
- Unlabeled data is **cheap and abundant**
- Can be used as a **preprocessing step** before doing supervised learning

# Setting and objective

- Euclidean space $\mathcal{X} = \mathbb{R}^d$ with norm $\|\cdot\|$
- Samples $X_n = (x_1, \cdots, x_n)$

## Empirical reconstruction error

Given a closed set $S$, the empirical reconstruction error of $S$ is defined by:

$$\hat{\mathcal{E}}(S) = \frac{1}{n}\sum_{i=1}^{n} d_{\mathcal{X}}^2(x_i, S).$$

**Objective**: compute a set $\hat{S}$ that minimizes $\hat{\mathcal{E}}$.

# Unsupervised learning algorithms

- Necessity to **add constraints** on the problem: by setting $S$ to be the whole space,

$$\hat{\mathcal{E}}(S) = \frac{1}{n} \sum_{i=1}^{n} d_{\mathcal{X}}^{2}(x_i, S) = 0,$$

independently of the samples and the distribution.

- Unsupervised learning algorithm:

$$\begin{array}{rll} \mathcal{A}: & \chi^n & \to \mathcal{H} \subset \mathcal{P}(\mathcal{X}) \\ & X_n & \mapsto \hat{S} \end{array}$$

where $\mathcal{H}$ is the **hypothesis space**.

- **Objective**: compute the optimal set $\hat{S}$ such that

$$\hat{S} = \arg\min_{S \in \mathcal{H}} \hat{\mathcal{E}}(S).$$

# $k$-means

$$\mathcal{H} = \mathcal{P}_k \text{ where } \mathcal{P}_k \text{ is the class of sets of } k \text{ points,}$$

## Lloyd's algorithm [Lloyd,1982]

***Input***: a data set $X_n$, an integer $k$ (number of means).

***Output***: set of $k$ means.

1. Choose randomly the $(m_j)_{1 \leq j \leq k}$ among the $(x_i)_{1 \leq i \leq n}$ without replacement.
2. Assignment update:

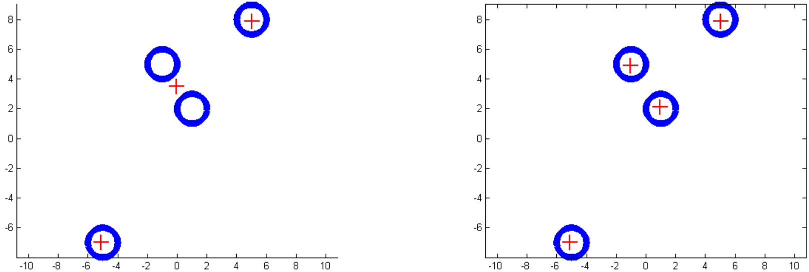
$$\forall i \in [\![1, n]\!],\ C(i) = \arg \min_{1 \leq j \leq k} \|x_i - m_j\|\,.$$

3. Means update:

$$\forall j \in [\![1, k]\!],\ m_j = \frac{\sum_{i \in \mathcal{C}_j} x_i}{n_j}.$$

4. Iterate steps 2. and 3. until convergence.

# Output of *k*-means



(a) 3-means

(b) 4-means

Figure: Output of $k$-means (in red) with $k = 3$ and $k = 4$ on a data set composed of four circles (in blue).

# Results on *k*-means

- Convergence ensured BUT **possibly towards a local minimum**
- No guarantees on "the proximity" to the optimal configuration
- Points are represented by their closest mean
- **Parameter *k* to be chosen**

# $k$-means++

## Algorithm [Arthur, Vassilvitskii, 2007]

***Input***: a data set $X_n$, an integer $k$ (number of means).
***Output***: set of $k$ means.

1. Randomly choose $m_1$ among the $(x_i)_{1 \leq i \leq n}$.
2. $\forall j \in [\![2, k]\!]$ choose $m_j = x_i$ with probability

$$\frac{D(x_i)^2}{\sum_{i=1}^{n} D(x_i)^2},$$

where $D(x)$ denotes the distance of $x$ to the closest mean already found.

- With this initialization: **guarantee in expectation on the proximity to the optimal configuration**

$$\mathbb{E}\left(\hat{\mathcal{E}}(S_{k++})\right) \leq 8(\ln k + 2)\hat{\mathcal{E}}(\hat{S}_k).$$

where $\hat{S}_k = \arg\min_{S \in \mathcal{P}_k} \hat{\mathcal{E}}(S)$

# $k$-flats

$\mathcal{H} = \mathcal{F}_{k,m}$ where $\mathcal{F}_{k,m}$ is the class of sets of $k$ affine spaces of dimension $m$ each.

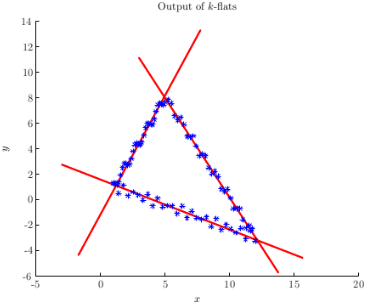
## Algorithm [Bradley, Mangasarian, 2000]

***Input***: a data set $X_n$, two integers $k$ (number of flats) and $m$ (dimension of the flats).

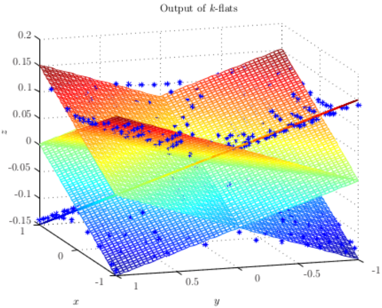
***Output***: set of $k$ affine spaces.

1. Initialize the assignment vector $C$.
2. $\forall j \in [\![1, k]\!]$ compute the $(F_j)_{1 \leq j \leq k}$ by finding the best $m$-dimensional ($1 \leq m \leq d$) flat (*i.e* the one that minimizes $\sum_{i \in C_j} d_{\mathcal{X}}(x_i, F_j)^2$).
3. Assignment: assign each data point to the closest flat, that is
$$\forall i \in [\![1, n]\!],\ C(i) = \arg\min_{1 \leq j \leq k} d_{\mathcal{X}}(x_i, F_j).$$
4. Repeat steps 2. and 3. until convergence.

# Output of *k*-flats



(a) Output of $k$-flats (in red) with $k = 3$ on a noisy triangle (in blue).



(b) Output of $k$-flats with $k = 4$ on an elliptic paraboloid (in blue).

# Results on *k*-flats

- Convergence ensured BUT towards a **local minimum**
- The points are represented by their projections on the closest flat
- 2 parameters to be chosen: **number of flats** *k* and **dimension of each flat** *m*

# Contents

1. Background

2. Learning data representation
   - Setting
   - Complexity regularization

3. Kernel extensions

# Framework

- $\mathcal{X}$: Hilbert space with norm $\|\cdot\|$ and inner product $\langle \cdot, \cdot \rangle$
- $p$ probability measure with support on a $d$-dimensional manifold $\mathcal{M}$. Density $\rho$ with respect to the volume measure on the manifold
- $X_n = (x_1, \cdots, x_n)$: samples drawn i.i.d. according to $p$

# Reconstruction error

## Reconstruction error

$$\mathcal{E}(S) = \int_{\mathcal{X}} d_{\mathcal{X}}^2(x,S)dp(x) = \int_{\mathcal{M}} \min_{m\in\mathcal{M}} \|x-m\|^2 \rho(x)d\mu_l(x)$$

where $\mu_l$ is the measure of volume on the manifold $\mathcal{M}$.

- $\mathcal{E}$ measures how well $S$ represents the distribution ρ
- $\mathcal{E}$ cannot be computed
- Algorithm computes the set $\hat{S}$ such as:

$$\hat{S} = \arg\min_{S\in\mathcal{H}} \hat{\mathcal{E}}(S).$$

- **Objective**: what we are interested is to represent the **true distribution** and so measures the error of $\hat{S}$ on this distribution:

$$\mathcal{E}(\hat{S}).$$

# Free parameters

- Free parameters:
  - *k*-means: number of means *k*.
  - *k*-flats: number of flats *k* and dimension of the flats *m*.
- **Choice** to minimize the reconstruction error. For instance, find (for *k*-means) $\hat{k}$ such that:

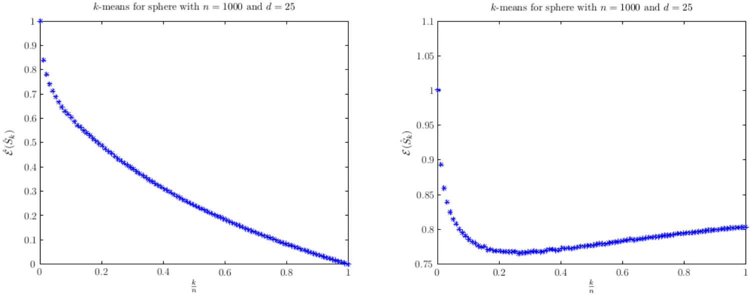
$$\hat{k} = \arg\min_{1 \leq k \leq n} \mathcal{E}(\hat{S}_k)$$

$$\text{where } \hat{S}_k = \arg\min_{S \in \mathcal{P}_k} \mathcal{E}(S).$$

- Impossible to compute $\hat{k}$ as $\mathcal{E}$ **cannot be computed**.

# Non-trivial choice

- The training error decreases with the number of means.
- But, for the **testing error**, in general, $\hat{k} \neq n$ !



(c) $d$=25, training

(d) $d$=25, testing

Figure: Training and testing error on a 25-dimensional unit-sphere.

# How to explain the tradeoff ?

- Decomposition of the reconstruction error:

$$\mathcal{E}(\hat{S}) \leq 2 \underbrace{\sup_{S\in\mathcal{H}} |\mathcal{E}(S) - \hat{\mathcal{E}}(S)|}_{\text{Statistical error}} + \underbrace{\mathcal{E}(S^*)}_{\text{Approximation error}} .$$

- Bounds:
  - Approximation error [Gruber, 2002]

$$\lim_{k\to\infty} \mathcal{E}_k . k^{\frac{2}{d}} \leq C_d \left( \int_{\mathcal{M}} \rho(x)^{\frac{d}{d+2}} d\mu_I(x) \right)^{\frac{d+2}{d}} .$$

  - Statistical error:
    - ★ Lower bound [Bartlett, Linder, Lugosi, 1997]

$$\exists \rho,\ \mathbb{E}\left(\mathcal{E}(\hat{S}_k)\right) - \mathcal{E}^* \geq C\sqrt{\frac{k^{1-4/d}}{n}}.$$

    - ★ Upper bounds [Bartlett, Linder, Lugosi, 1997]

$$\mathbb{E}\left(\mathcal{E}(\hat{S}_k)\right) - \mathcal{E}^* \leq C\sqrt{\frac{k^{1-2/d} d \log n}{n}}.$$

## Proposition [Canas, Rosasco, 2012]

For $\delta \leq \frac{1}{e}$, there are constants $C_d$ and $\gamma_d$ dependent only on $d$, and a sufficiently large $N$ such that, by setting

$$k_n = n^{\frac{d}{2(d+2)}} \cdot \left( \frac{C_d}{24\sqrt{\pi}} \right)^{\frac{d}{d+2}} \cdot \int_{\mathcal{M}} \rho(x)^{\frac{d}{d+2}} d\mu_I(x),$$

and $\hat{S}^* = \hat{S}_{k_n}$, it is

$$\forall n \geq N, \ \mathbb{P}\left( \mathcal{E}(\hat{S}^*) \leq \int_{\mathcal{M}} \gamma_d . \rho(x)^{\frac{d}{d+2}} d\mu_I(x) \sqrt{\ln\left(\frac{1}{\delta}\right)} n^{-\frac{1}{d+2}} \right) \geq 1 - \delta,$$

where $C_d \sim d$ and $\gamma_d$ grows sublinearly with $d$.

- **Problem**: $k_n$ depends on unknown quantities ($\rho$ in particular).

# Complexity regularization

- **Idea**: penalize models with **high complexity**

- Complexity for *k*-means: number of means.

## Penalized reconstruction error

$$\underbrace{\tilde{\mathcal{E}}(S)}_{\text{penalized}} = \underbrace{\hat{\mathcal{E}}(S)}_{\text{empirical}} + \underbrace{p(k,n)}_{\text{penalty}}.$$

# How to choose $p(k,n)$ ?

## Proposition [Maurer, Pontil, 2010]

Let ρ be such that $\mathrm{supp}(\rho) \subseteq \mathcal{B}(0,1)^d$. Then:

$$\mathbb{P}\left(\sup_{S\in\mathcal{P}_k} |\mathcal{E}(S) - \hat{\mathcal{E}}(S)| \leq \frac{k\sqrt{18\pi}}{\sqrt{n}} + \sqrt{\frac{8\log 1/\delta}{n}}\right) \geq 1-\delta.$$

- We choose

$$p(k,n) = \frac{k\sqrt{18\pi}}{\sqrt{n}} + \underbrace{4\sqrt{\frac{\ln k}{n}}}_{\text{for convergence}} .$$

# Does it work?

$$\tilde{k} = \arg\min_{k \in [\![1,n]\!]} \tilde{\mathcal{E}}(\hat{S}_k)$$
$$\tilde{S} = \hat{S}_{\tilde{k}}$$

## Proposition

$$\forall \varepsilon > 0,\ \mathbb{P}\left(\mathcal{E}(\tilde{S}) > \min_{1 \le k \le n}\left(\mathcal{E}(S_k) + p(k,n) + 4\sqrt{\frac{\ln k}{n}}\right) + \varepsilon\right) \le \frac{\pi^2}{3} e^{-\frac{n\varepsilon^2}{32}}.$$

- However, in practice it does not work very well. Example of $d$-dimensional spheres. The **penalty** $p(k,n)$ **does not depend on** $d$, and the training error does not change that much for varying $d$ $\Rightarrow$ the optimal value $\tilde{k}$ of $k$ would roughly be the same for each $d$, which is not what we observe in practice.

# Contents

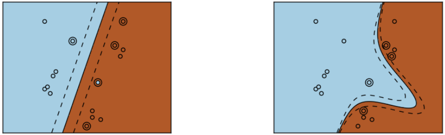
1 Background

2 Learning data representation

3 Kernel extensions

# Motivation

- Supervised learning: kernels are used to get nonlinear classification boundaries



(a) Without kernel.

(b) Polynomial kernel.

- **Idea**: by using kernels, we may be able to clusterize datasets that are **nonlineraly separable**

# Notions on kernels

- Data are mapped to a **high-dimensional Hilbert space** $H$ using a function $\phi : \mathcal{X} \rightarrow H$
- $\phi$ **is not known**: we have at our disposal only a **kernel function** $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$. It is such that:

  $$k(x,y) = \langle \phi(x), \phi(y) \rangle_H$$

  Example: gaussian kernel

  $$k(x,y) = \exp\left( -\frac{\|x-y\|^2}{2\sigma^2} \right).$$

- **Kernel matrix**:

  $$K_{ij} = k(x_i, x_j),\ i,j \in [\![1,n]\!]$$

# Kernel $k$-means

$\mathcal{H} = \mathcal{P}_k^{(f)}$ where $\mathcal{P}_k^{(f)}$ denotes the sets of $k$ points from the feature space.

## Kernel $k$-means [Dhillon, Guan, Kulis, 2004]

***Input***: a data set $X_n$, an integer $k$ (number of means), and a kernel $k(\cdot,\cdot)$.
***Output***: assignment vector $C$.

1. Initialize the assignment vector $C$ with values between 1 and $k$.
2. Compute the distances from the data points to the means (see equation (**??**) for details), that is compute:

$$d_H(\phi(x_i), m_j), \ \ \forall i \in [\![1,n]\!], \ \forall j \in [\![1,k]\!].$$

3. Assign data points to their closest mean:

$$\forall i \in [\![1,n]\!], \ C(i) = \arg\min_{1\leq j\leq k} \|\phi(x_i) - m_j\|_H .$$

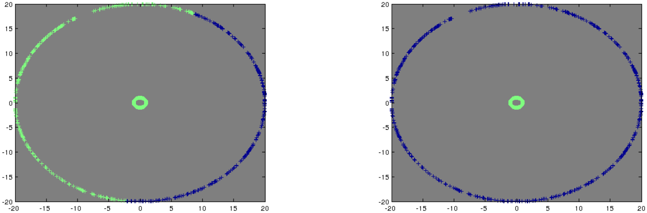
4. Repeat steps 2. and 3. until convergence.

# Comparison with standard *k*-means

- Same guarantees as *k*-means: convergence towards a **local minimum**
- Computation of the distances uses the kernel matrix:

$$\|\phi(x_i) - m_j\|^2 = K_{ii} + \frac{1}{n_j^2}\sum_{l,l'\in\mathcal{C}_j} K_{ll'} - \frac{2}{n_j}\sum_{l\in\mathcal{C}_j} K_{il}$$

- the output cannot be a set of means as $\phi$ is not known. Instead, kernel *k*-means outputs the **assignment vector** *C*



(c) *k*-means

(d) Kernel *k*-means

Figure: Comparison between *k*-means and kernel *k*-means.

# Kernel $k$-flats

$\mathcal{H} = \mathcal{F}_{k,m}^{(f)}$ where $\mathcal{F}_{k,m}^{(f)}$ is the class of sets of $k$ vector spaces of dimension $m$ in the feature

## Kernel $k$-flats

***Input***: a data set $X_n$, two integers $k$ (number of means) and $m$ (number of flats), a kernel $k(\cdot,\cdot)$.

***Output***: assignment vector $C$.

1. Initialize the assignment vector $C$ with values between 1 and $k$.
2. $\forall j \in [\![1,k]\!]$, compute the distances from the data points to the flats, that is compute:
$$d_H(\phi(x_i), F_j)^2, \;\; \forall i \in [\![1,n]\!], \; \forall j \in [\![1,k]\!].$$
3. Assignment: assign each data point to the closest (in the feature space) flat:
$$\forall i \in [\![1,n]\!], \; C(i) = \arg\min_{1\leq j\leq k} d_H(\phi(x_i), F_j).$$
4. Repeat steps 2. and 3. until convergence.

# Conclusion and further work

- Method for choosing the number of means in *k*-means.
  - What about the **number of flats and the dimension of flats** in *k*-flats?
  - More generally, **bounds** on the statistical error for *k*-flats and **tightness** of the existing bounds.
- Kernel extensions.
  - What exactly happens in the **input space** and how to rationalize the intuition that kernel extensions perform better on nonlineraly separable datasets ?
  - Further study the **output of kernel *k*-flats** ?