Regularization in Unsupervised Learning

Guille Canas, Lorenzo Rosasco

MIT, 9.520 Class 22

May 2, 2012

About this class

Goal To discuss some problems in unsupervised learning and in particular introduce a statistical learning framework for learning data representation/reconstruction under constraints, discussing the role played by regularization theory and regularized algorithms.

Plan

- From supervised learning to learning data representation
- Unsupervised Learning Algorithms
- Computations
- Final Comments

Supervised Learning

- we are given a trainining set $S = ((x_1, y_1), \ldots, (x_n, y_n))$ sampled i.i.d. with respect to $p(x, y)$.
- the problem is: *to predict the best possible output for* **new** *input*
- find

$$f(x_{new}) = y_{new}.$$

Statistical Learning Theory for Supervised Learning

The problem can be formalized fixing some concepts.

- loss function $V(y, f(x))$
- hypotheses space $\mathcal{H} : \{f \mid f : X \to \mathbb{R}\}$
- a learning algorithm maps the data into a function, i.e. $A(S) = f_S \in \mathcal{H}$

Statistical Learning Theory for Supervised Learning

The problem can be formalized fixing some concepts.

- loss function $V(y, f(x))$
- hypotheses space $\mathcal{H} : \{f \mid f : X \to \mathbb{R}\}$
- a learning algorithm maps the data into a function, i.e. $A(S) = f_S \in \mathcal{H}$

Generalization

A good algorithm is such that

$$\mathbb{P}(|I_S[f_S] - I[f_S]| \geq \epsilon)$$

is small,or

$$\mathbb{P}(I[f_S] - \inf_{f \in \mathcal{H}} I[f] \geq \epsilon)$$

is small, where $I[f] = \mathbb{E}[V(y, f(x))]$.

Empirical Risk and Regularization

Typically we use algorithm based on the (regularized) empirical risk minimization.

- Constrained minimization

$$\min_{f \in \mathcal{H}_\lambda} I_S[f_S]$$

- or penalized minimization

$$\min_{f \in \mathcal{H}} \{ I_S[f_S] + \lambda R(f) \}.$$

Unsupervised Learning

- we are given a trainining set $S = (x_1, \ldots, x_n)$ sampled i.i.d. with respect to $p(x)$.
- the problem is to extract from data "useful" information about p.....

Unsupervised Learning is not One but Many Problems

- density estimation
- dimensionality reduction
- clustering
- manifold learning
- correlation analysis
- association rules
- networks/graph analysis
- dictionary learning, vector quantizaton, **data representation/reconstruction**

Unsupervised learning is ubiquitous: many algorithms, no unified framework.

Learning Data Representation

Data representation is the preliminary step to supervised learning. Representation are often designed/engineered, but ideally should be *learned*.

We have discussed how a *good representation* should be discriminant and yet invariant to *task irrelevant* transformations.

Ideally such a good representation should reduce the sample complexity of subsequent supervised tasks.

Learning Data Representation (cont.)

Some questions before we even start..

- How can we know what is “task relevant” if we don’t have labels (we don’t know what’s the task)?
- What do we mean with learning a representation?
- What guarantees we have? (stability, overfitting...)

Plan

- From supervised learning to learning data representation
- Unsupervised Learning Algorithms
- Computations
- Final Comments

Approach

We are going to replace

$$\text{discrimination} \mapsto \text{reconstruction+constraints}$$

for the time being we are not going to consider the problem of invariance (see last lectures to see how this can be done).

Empirical Reconstruction Error

Given a training set S, we are going to consider algorithms that minimize the following empirical (reconstruction) error,

$$I_S[C] = \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, C)$$

where:

- C is a subset of X
- $d^2(x_i, C) = \min_{v \in C} \|x - v\|^2$ is the square distance of a point x to the set C

Regularization framework

We are going to consider (constrained) algorithms based on

$$\min_{C \in \mathcal{H}_k} I_S[C]$$

where $\mathcal{H}_k \subset \{C \mid C \subset X\}$ is a hypotheses space of suitable subsets of X.
The above algorithm returns an estimator $A(S) = C_S \in \mathcal{H}_k$.

Remarks

- the above error measure depends on the distribution, points that are more likely to be sampled contribute more to the error
- we can alternatively consider penalized algorithms

$$\min_{C \in \mathcal{H}} I_S[C] + R(C)$$

where $\mathcal{H} \subset \{C \mid C \subset X\}$ is an essentially unconstrained hypotheses space and $R(C)$ a regularizer that penalize *complex/large* sets.

Examples

The above framework is general enough to encompass a variety of algorithms.

- PCA
- Sparse coding
- K-Means
- K-Flats
- Non Negative Matrix Factorization
- ...

Each algorithm is defined by a suitable choice of $\mathcal{H}_k$.

Hypothesis Space

Hypothesis space

$$\mathcal{H} = \{C \subseteq \mathcal{X} : C = \{x : x = Tb = \sum_{j=1}^{k} b^j t_j, \text{ with } T \in \mathcal{T}, b \in \mathcal{B}\}\}$$

- $\mathcal{T}$ - linear transformations $\mathcal{B} \mapsto \mathcal{X}$, $Tb = \sum_{j=1}^{k} b^j t_j$, where t_j are the **code vectors** defining a **dictionary**.
- The choice of $\mathcal{B}$ determines the encoding $x \mapsto (b^1, \ldots, b^k)$ and the algorithm.

Hypothesis Space

Hypothesis space

$$\mathcal{H} = \{C \subseteq \mathcal{X} : C = \{x : x = Tb = \sum_{j=1}^{k} b^j t_j, \text{ with } T \in \mathcal{T}, b \in \mathcal{B}\}\}$$

- $\mathcal{T}$ - linear transformations $\mathcal{B} \mapsto \mathcal{X}$, $Tb = \sum_{j=1}^{k} b^j t_j$, where t_j are the **code vectors** defining a **dictionary**.
- The choice of $\mathcal{B}$ determines the encoding $x \mapsto (b^1, \ldots, b^k)$ and the algorithm.

Hypothesis Space

Hypothesis space

$$\mathcal{H} = \{C \subseteq \mathcal{X} : C = \{x : x = Tb = \sum_{j=1}^{k} b^j t_j, \text{ with } T \in \mathcal{T}, b \in \mathcal{B}\}\}$$

- $\mathcal{T}$ - linear transformations $\mathcal{B} \mapsto \mathcal{X}$, $Tb = \sum_{j=1}^{k} b^j t_j$, where t_j are the **code vectors** defining a **dictionary**.
- The choice of $\mathcal{B}$ determines the encoding $x \mapsto (b^1, \ldots, b^k)$ and the algorithm.

Hypothesis Space

- **Input**: samples S.
- **Output**: mapping T_S (determines set $C_S = T_S(\mathcal{B})$).

- **Dictionary**: T in coordinates $(t_1, \ldots, t_k)$.
- **Encoding**: given x, encoding is $\text{argmin}_{b \in \mathcal{B}} d^2(x, T(b))$
- $C = T(\mathcal{B})$ are encoded with no error

Hypothesis Space

- **Input**: samples S.
- **Output**: mapping T_S (determines set $C_S = T_S(\mathcal{B})$).

- **Dictionary**: T in coordinates $(t_1, \ldots, t_k)$.
- **Encoding**: given x, encoding is $\text{argmin}_{b \in \mathcal{B}} d^2(x, T(b))$
- $C = T(\mathcal{B})$ are encoded with no error

PCA:

- Let $\mathcal{B} = \mathbb{R}^k$
- Sets are of the form $C = T(\mathbb{R}^k)$, with T linear
  - C is a k-dimensional linear subspaces of $\mathcal{X}$

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(\mathbb{R}^k))$$

$$= \min_{\text{rank-}k \text{ linear projection } \pi} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, \pi x_i)$$

- **Free parameter**: k (maximum dimension of C)

PCA:

- Let $\mathcal{B} = \mathbb{R}^k$
- Sets are of the form $C = T(\mathbb{R}^k)$, with T linear
  - C is a k-dimensional linear subspaces of $\mathcal{X}$

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(\mathbb{R}^k))$$

$$= \min_{\text{rank-}k \text{ linear projection } \pi} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, \pi x_i)$$

- **Free parameter**: k (maximum dimension of C)

PCA:

- Let $\mathcal{B} = \mathbb{R}^k$
- Sets are of the form $C = T(\mathbb{R}^k)$, with T linear
  - C is a k-dimensional linear subspaces of $\mathcal{X}$

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(\mathbb{R}^k))$$

$$= \min_{\text{rank-}k \text{ linear projection } \pi} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, \pi x_i)$$

- **Free parameter**: k (maximum dimension of C)

PCA:

- Let $\mathcal{B} = \mathbb{R}^k$
- Sets are of the form $C = T(\mathbb{R}^k)$, with T linear
  - C is a k-dimensional linear subspaces of $\mathcal{X}$

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(\mathbb{R}^k))$$

$$= \min_{\text{rank-}k \text{ linear projection } \pi} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, \pi x_i)$$

- **Free parameter**: k (maximum dimension of C)

K-Means

K-Means:

- Let $\mathcal{B} = \{e_1, \ldots, e_k\}$
- Sets of the form $C = T(\mathcal{B}) \subseteq \mathcal{X}$, with T linear mapping
  - Arbitrary sets $C \subset \mathcal{X}$ of size $|C| = k$

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(\{e_1, \ldots, e_k\}))$$

$$= \min_{c=\{m_1,\ldots,m_k\}} \frac{1}{n} \sum_{i=1}^{n} \min_{j=1}^{k} d^2(x_i, m_j)$$

- **Free parameter**: k (size of C)

K-Means

K-Means:

- Let $\mathcal{B} = \{e_1, \ldots, e_k\}$
- Sets of the form $C = T(\mathcal{B}) \subseteq \mathcal{X}$, with T linear mapping
  - Arbitrary sets $C \subset \mathcal{X}$ of size $|C| = k$

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(\{e_1, \ldots, e_k\}))$$

$$= \min_{c=\{m_1,\ldots,m_k\}} \frac{1}{n} \sum_{i=1}^{n} \min_{j=1}^{k} d^2(x_i, m_j)$$

- **Free parameter**: k (size of C)

K-Means

K-Means:

- Let $\mathcal{B} = \{e_1, \ldots, e_k\}$
- Sets of the form $C = T(\mathcal{B}) \subseteq \mathcal{X}$, with T linear mapping
  - Arbitrary sets $C \subset \mathcal{X}$ of size $|C| = k$

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(\{e_1, \ldots, e_k\}))$$

$$= \min_{c=\{m_1,\ldots,m_k\}} \frac{1}{n} \sum_{i=1}^{n} \min_{j=1}^{k} d^2(x_i, m_j)$$

- **Free parameter**: k (size of C)

K-Means

K-Means:

- Let $\mathcal{B} = \{e_1, \ldots, e_k\}$
- Sets of the form $C = T(\mathcal{B}) \subseteq \mathcal{X}$, with T linear mapping
  - Arbitrary sets $C \subset \mathcal{X}$ of size $|C| = k$

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(\{e_1, \ldots, e_k\}))$$

$$= \min_{C=\{m_1,\ldots,m_k\}} \frac{1}{n} \sum_{i=1}^{n} \min_{j=1}^{k} d^2(x_i, m_j)$$

- **Free parameter**: k (size of C)

K-Flats

K-Flats:

- Let $\mathcal{B} \subset \mathbb{R}^{m \times k}$
- $\mathcal{B} = \cup_{j=1}^{k} \text{span}\left(e_{j,1}, \ldots, e_{j,m}\right) \neq \mathbb{R}^{m \times k}$
- Sets are collections of k m-dimensional linear subspaces of $\mathcal{X}$ (m-flats)

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(\mathcal{B}))$$

$$= \min_{\substack{C = \{\pi_1, \ldots, \pi_k\} \\ \pi_j \text{ rank-}k \text{ linear projection}}} \frac{1}{n} \sum_{i=1}^{n} \min_{j=1}^{k} d^2(x_i, \pi_j x_i)$$

- **Free parameters**: k, m (number and dimension of flats)

K-Flats

K-Flats:

- Let $\mathcal{B} \subset \mathbb{R}^{m \times k}$
- $\mathcal{B} = \cup_{j=1}^{k} \text{span}\left(e_{j,1}, \ldots, e_{j,m}\right) \neq \mathbb{R}^{m \times k}$
- Sets are collections of k m-dimensional linear subspaces of $\mathcal{X}$ (m-flats)

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(\mathcal{B}))$$

$$= \min_{\substack{C=\{\pi_1,\ldots,\pi_k\} \\ \pi_j \text{ rank-}k \text{ linear projection}}} \frac{1}{n} \sum_{i=1}^{n} \min_{j=1}^{k} d^2(x_i, \pi_j x_i)$$

- **Free parameters**: k, m (number and dimension of flats)

K-Flats

K-Flats:

- Let $\mathcal{B} \subset \mathbb{R}^{m \times k}$
- $\mathcal{B} = \cup_{j=1}^{k} \text{span}\left(e_{j,1}, \ldots, e_{j,m}\right) \neq \mathbb{R}^{m \times k}$
- Sets are collections of k m-dimensional linear subspaces of $\mathcal{X}$ (m-flats)

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(\mathcal{B}))$$

$$= \min_{\substack{C=\{\pi_1,\ldots,\pi_k\} \\ \pi_j \text{ rank-}k \text{ linear projection}}} \frac{1}{n} \sum_{i=1}^{n} \min_{j=1}^{k} d^2(x_i, \pi_j x_i)$$

- **Free parameters**: k, m (number and dimension of flats)

Sparse Coding

- Let $\mathcal{B} = B_1(\lambda) = \{y \in \mathbb{R}^k : \|y\|_1 \leq \lambda\}$
- $\mathcal{T} = \{ \text{ linear } T : \|Te_i\| \leq \gamma, 1 \leq i \leq k\}$

- Sets are of the form $C = T(B_1(\lambda))$
  (Typically sparse) linear combinations of k points

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(B_1(\lambda)))$$

- **Free parameter**: λ, controls sparsity.

Sparse Coding

- Let $\mathcal{B} = B_1(\lambda) = \{y \in \mathbb{R}^k : \|y\|_1 \leq \lambda\}$
- $\mathcal{T} = \{$ linear $T : \|Te_i\| \leq \gamma, 1 \leq i \leq k\}$

- Sets are of the form $C = T(B_1(\lambda))$
  (Typically sparse) linear combinations of k points

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(B_1(\lambda)))$$

- **Free parameter**: λ, controls sparsity.

Sparse Coding

- Let $\mathcal{B} = B_1(\lambda) = \{y \in \mathbb{R}^k : \|y\|_1 \leq \lambda\}$
- $\mathcal{T} = \{ \text{ linear } T : \|Te_i\| \leq \gamma, 1 \leq i \leq k\}$

- Sets are of the form $C = T(B_1(\lambda))$
  (Typically sparse) linear combinations of k points

$$I_{S,k} = \min_{T \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, T(B_1(\lambda)))$$

- **Free parameter**: λ, controls sparsity.

Plan

- From supervised learning to learning data representation
- Unsupervised Learning Algorithms
- Computations
- Final Comments

Unsupervised Learning: K-Means Problem

K-means problem: find set $C = \{m_1, \ldots, m_k\}$ of size k minimizing

$$I_S[C] = \frac{1}{n} \sum_{i=1}^{n} \min_{j=1}^{k} d^2(x_i, m_j)$$

- Non-linear, non-convex.
- NP-hard even for $k = 2$!

K-Means Problem

$$I_S[C] = \frac{1}{n} \sum_{i=1}^{n} \min_{j=1}^{k} d^2(x_i, m_j)$$

- Given means, optimal partition given by assignment function:

$$a(x_i) = \text{argmin}_{j=1}^{k} d^2(x_i, m_j)$$

- Given a partition, optimal means given by centers of mass:

$$m_j = \sum_{i:a(x_i)=j} x_i \; / \sum_{i:a(x_i)=j} 1$$

K-Means with Lloyd's Algorithm

$$I_S[C] = \frac{1}{n} \sum_{i=1}^{n} \min_{j=1}^{k} d^2(x_i, m_j)$$

Lloyd's algorithm:

- Initialize $\{m_i : i = 1, \ldots, k\}$ randomly in $S = \{x_1, \ldots, x_n\}$ (without replacement)
- Repeat until convergence:
  - find partition given means,

$$a(x_i) = \text{argmin}_{j=1}^{k} d^2(x_i, m_j)$$

  - update means given the above partition,

$$m_j = \sum_{i:a(x_i)=j} x_i \; / \sum_{i:a(x_i)=j} 1$$

K-Means with Lloyd's Algorithm

$$I_S[C] = \frac{1}{n} \sum_{i=1}^{n} \min_{j=1}^{k} d^2(x_i, m_j)$$

Lloyd's algorithm:

- Initialize $\{m_i : i = 1, \ldots, k\}$ randomly in $S = \{x_1, \ldots, x_n\}$ (without replacement)
- Repeat until convergence:
  - find partition given means,

$$a(x_i) = \text{argmin}_{j=1}^{k} d^2(x_i, m_j)$$

  - update means given the above partition,

$$m_j = \sum_{i:a(x_i)=j} x_i \; / \sum_{i:a(x_i)=j} 1$$

K-Means with Lloyd's Algorithm (cont.)

Lloyd's algorithm:

- a) Given means $\Rightarrow$ optimize partition.
- b) Given partition $\Rightarrow$ optimize means.

Greedy **block-coordinate descent**. Does it converge?

- Steps a) and b) **strictly** decrease $I_S[C]$, until convergence.
- $\Rightarrow$ no partition is repeated (never twice in the same state).
- Number of different partitions of S is $\leq k^n$

$\Rightarrow$ must converge to **local minimum** in finite number of steps.

How close to **global optimum**?: we don't know.

K-Means with Lloyd's Algorithm (cont.)

Lloyd's algorithm:

- a) Given means $\Rightarrow$ optimize partition.
- b) Given partition $\Rightarrow$ optimize means.

Greedy **block-coordinate descent**. Does it converge?

- Steps a) and b) **strictly** decrease $I_S[C]$, until convergence.
- $\Rightarrow$ no partition is repeated (never twice in the same state).
- Number of different partitions of S is $\leq k^n$

$\Rightarrow$ must converge to **local minimum** in finite number of steps.

How close to **global optimum**?: we don't know.

K-Means with Lloyd's Algorithm (cont.)

Lloyd's algorithm:

- a) Given means $\Rightarrow$ optimize partition.
- b) Given partition $\Rightarrow$ optimize means.

Greedy **block-coordinate descent**. Does it converge?

- Steps a) and b) **strictly** decrease $I_S[C]$, until convergence.
- $\Rightarrow$ no partition is repeated (never twice in the same state).
- Number of different partitions of S is $\leq k^n$

$\Rightarrow$ must converge to **local minimum** in finite number of steps.

How close to **global optimum**?: we don't know.

K-Means++

Choose seeding adaptively:

- m_1 uniform random in $S = \{x_1, \ldots, x_n\}$.
- For $j = 2, \ldots, k$, let

$$\mathbb{P}[m_j = i\,] \propto \min_{l=1,\ldots,j-1} d^2(x_i, m_l)$$

(pick means far from previously-inserted means)

Properties of K-Means++

K-Means++

- m_1 uniform random in $S = \{x_1, \ldots, x_n\}$.
- $\mathbb{P}[m_j = i\,] \propto \min_{l=1,\ldots,j-1} \; d^2(x_i, m_l), \quad j = 2, \ldots, k$

Guarantees:

- $O(\ln k)$-Approximation **randomized** algorithm: after seeding it is

$$\mathbb{E}\left\{I_S[\{m_1, \ldots, m_k\}]\right\} \leq 8(\ln k + 2) \cdot \min_{|C|=k} I_S[C]$$

Properties of K-Means++

K-Means++

- m_1 uniform random in $S = \{x_1, \ldots, x_n\}$.
- $\mathbb{P}[m_j = i\,] \propto \min_{l=1,\ldots,j-1} \; d^2(x_i, m_l), \quad j = 2, \ldots, k$

Guarantees:

- $O(\ln k)$-Approximation **randomized** algorithm: after seeding it is

$$\mathbb{E}\left\{I_S[\{m_1, \ldots, m_k\}]\right\} \leq 8(\ln k + 2) \cdot \min_{|C|=k} I_S[C]$$

Complexity of K-Means++

K-Means++

- m_1 uniform random in $S = \{x_1, \ldots, x_n\}$.
- $\mathbb{P}[m_j = i\,] \propto \min_{l=1,\ldots,j-1} \; d^2(x_i, m_l), \quad j = 2, \ldots, k$

Complexity: $O(kn)$.

- Choose m_1 randomly.
  Let $p_i \leftarrow d^2(x_i, m_1), i = 1, \ldots, n$ $\Theta(n)$
- For $j = 2, \ldots, k$: $k\times$
  - Draw $m_j \propto [p_1, \ldots, p_{j-1}]$ $\Theta(n)$
  - $p_i \leftarrow \min\{p_i, d^2(x_i, m_j)\}$ $\Theta(n)$

Complexity of K-Means++

K-Means++

- m_1 uniform random in $S = \{x_1, \ldots, x_n\}$.
- $\mathbb{P}[m_j = i\,] \propto \min_{l=1,\ldots,j-1} \; d^2(x_i, m_l), \quad j = 2, \ldots, k$

Complexity: $O(kn)$.

- Choose m_1 randomly.
  Let $p_i \leftarrow d^2(x_i, m_1), i = 1, \ldots, n$ $\Theta(n)$
- For $j = 2, \ldots, k$: $k\times$
  - Draw $m_j \propto [p_1, \ldots, p_{j-1}]$ $\Theta(n)$
  - $p_i \leftarrow \min\{p_i, d^2(x_i, m_j)\}$ $\Theta(n)$

K-Means++ Addenda

Additionally:

- May still run Lloyd algorithm after.
- **Incremental**: computes all intermediate solutions ($k' = 1, \ldots, k$).
- Proof does not (trivially) extend to K-Flats.

Generalization Bounds

Can we quantify the generalization properties of the previous unsupervised learning algorithms?

Recall that we defined,

- **Empirical reconstruction error**: $I_S[C] = \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, C)$
- **Expected reconstruction error**: $I[C] = \mathbb{E}[d^2(x, C)]$

Generalization Bounds (cont.)

A good algorithm is such that

- generalization

$$\mathbb{P}(|I_S[C_S] - I[C_S]| \geq \epsilon)$$

  is small,or

- consistency (excess risk bounds)

$$\mathbb{P}(I[C_S] - \inf_{C \in \mathcal{H}_k} I[C] \geq \epsilon)$$

  is small.

Many algorithms can be extended using kernels.
Given a feature map $\Phi : X \to \mathcal{H}$, the idea is to consider

$$I_S[C] = \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, C)$$

where:

- C is a subset of $\mathcal{H}$
- $d^2(x_i, C) = \min_{v \in C} \|\Phi(x) - v\|^2_{\mathcal{H}}$ is the square distance of the feature map $\Phi(x)$ of a point x to the set C

Kernel K-Means

For example in the case of kernel k-means, the algorithms will compute (implicitly) means in the feature space

$$m_j = \frac{1}{\ell_j} \sum_{i=1}^{\ell_j} \Phi(x_i), \quad \ell_j \leq n.$$

Indeed the distance of a point to a such mean can be computed explicitly

$$\left\| \Phi(x) - m_j \right\|_{\mathcal{H}}^2 = K(x,x) - \frac{1}{\ell_j} \sum_{i=1}^{\ell_j} K(x, x_i) + \frac{1}{{\ell_j}^2} \sum_{i=1}^{\ell_j} K(x_i, x_i)$$

Summary and Questions

- Learning data representation can be described in a regularization framework.
- Different Hypotheses spaces induce different algorithms.
- Computations are considerably more complicated (lack of convexity).

- Sample/approximation tradef-offs?
- Partial supervision (semi-supervised, time, constraints– symmetry etc.)

K-Flats Problem

Modified Lloyd algorithm:

- Initialize flats "randomly" $\{\pi_i : i = 1, \ldots, k\}$.
- Repeat until convergence:
  - Given flats, optimal partition given by assignment function:

$$a(x_i) = \text{argmin}_{j=1}^{k} d^2(x_i, \pi_j x_i)$$

  - Given partition, optimal flat π_j given by top eigenvectors (PCA) of

$$S_j S_j^t = \sum_{i:a(x_i)=j} x_i x_i^t$$

- Similar guarantees: convergence to **local minimum**.

K-Flats Problem

Modified Lloyd algorithm:

- Initialize flats "randomly" $\{\pi_i : i = 1, \ldots, k\}$.
- Repeat until convergence:
  - Given flats, optimal partition given by assignment function:

$$a(x_i) = \text{argmin}_{j=1}^{k} d^2(x_i, \pi_j x_i)$$

  - Given partition, optimal flat π_j given by top eigenvectors (PCA) of

$$S_j S_j^t = \sum_{i:a(x_i)=j} x_i x_i^t$$

- Similar guarantees: convergence to **local minimum**.