Modèles statistiques pour l'image
Méthodes de classification

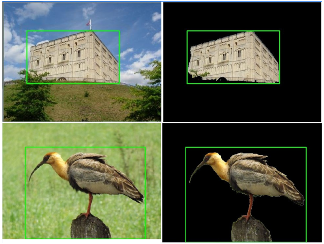
Julie Digne

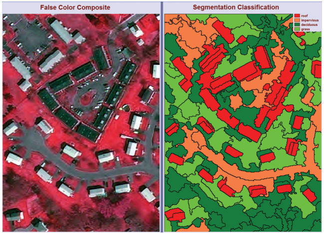
LIRIS - CNRS

16/09/2024

Outline

Objects to sort out in categories



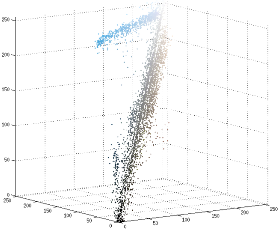


- pixels
- superpixels - image patches
- Entire images

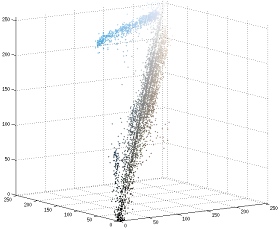
Classification Principle

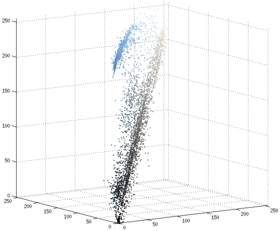
- *Describe* the objects to classify
- Natural description for pixels: A triplet $(R, G, B) \in \mathbb{R}^3$.
- But one can be more specific!

A color image in RGB

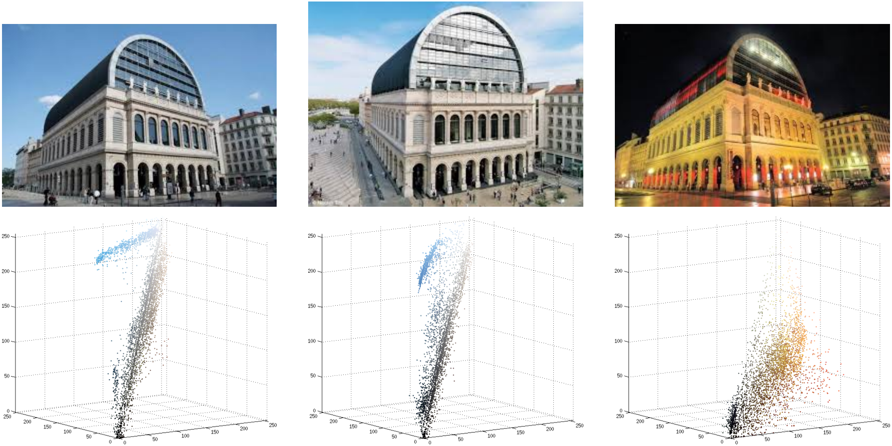


A color image in RGB





A color image in RGB



From the image domain to $\mathbb{R}^d$

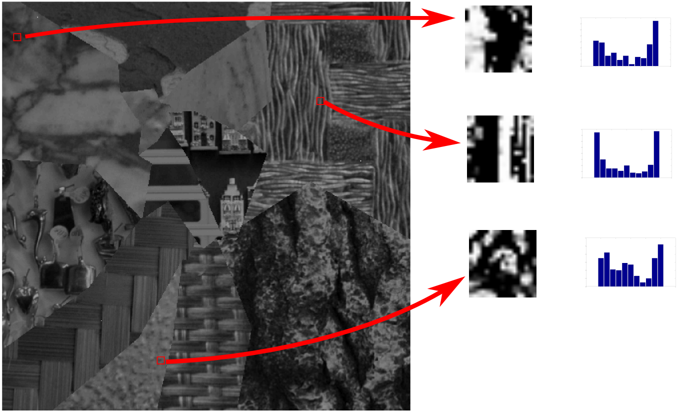
Data to classify

Recall that each pixel is representated as a vector in $\mathbb{R}^d$

Example: each pixel (i, j) of an image I can be encoded as:

- (i, j, r, g, b) in $\mathbb{R}^5$ (color image)
- $(\nabla_x I(i,j), \nabla_y I(i,j))$ in $\mathbb{R}^2$ (grayscale image)
- $(I(i-1,j-1), I(i,j-1), I(i+1,j-1), I(i-1,j), I(i,j), I(i+1,j), I(i-1,j+1), I(i,j+1), I(i+1,j+1))$ in $\mathbb{R}^9$ (grayscale image)

Descriptor example: local histograms



Histogram of gradient orientation

Descriptor example: response of the image to a set of filters

- Particularly well adapted for textures
- Each point is the set of responses of to a set of filters.
- Many filters have been proposed

For textures: Gabors filters

Gabor Filter

Measures the response to an oriented and localized filter. The filter writes:

$$G_{\theta,\sigma,\lambda} = \exp - \frac{x'^2 + y'^2}{2\sigma^2} \cos 2\pi\lambda \frac{x'}{\sigma}$$

with $x' = x \cos\theta + y \sin\theta$, $y' = x \sin\theta - y \cos\theta$

For textures: Gabors filters

Gabor Filter

Measures the response to an oriented and localized filter. The filter writes:

$$G_{\theta,\sigma,\lambda} = \exp - \frac{x'^2 + y'^2}{2\sigma^2} \cos 2\pi\lambda \frac{x'}{\sigma}$$

with $x' = x\cos\theta + y\sin\theta$, $y' = x\sin\theta - y\cos\theta$

- θ controls the filter orientation

For textures: Gabors filters

Gabor Filter

Measures the response to an oriented and localized filter. The filter writes:

$$G_{\theta,\sigma,\lambda} = \exp - \frac{x'^2 + y'^2}{2\sigma^2} \cos 2\pi\lambda \frac{x'}{\sigma}$$

with $x' = x\cos\theta + y\sin\theta$, $y' = x\sin\theta - y\cos\theta$

- θ controls the filter orientation
- σ controls the localization of the filter

For textures: Gabors filters

Gabor Filter

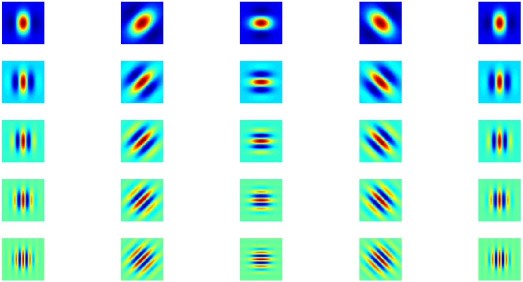
Measures the response to an oriented and localized filter. The filter writes:

$$G_{\theta,\sigma,\lambda} = \exp - \frac{x'^2 + y'^2}{2\sigma^2} \cos 2\pi\lambda \frac{x'}{\sigma}$$

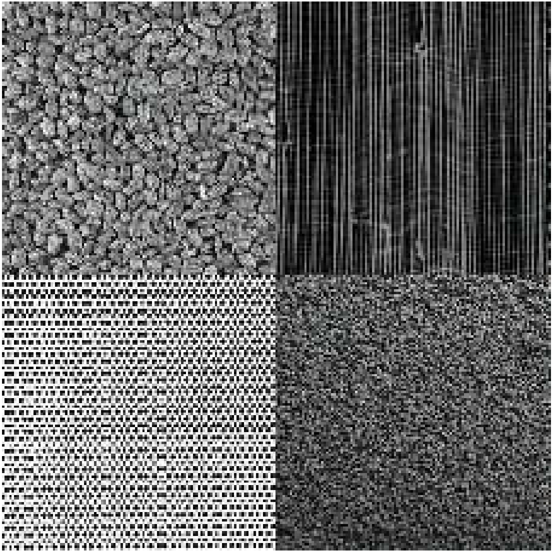
with $x' = x \cos\theta + y \sin\theta$, $y' = x \sin\theta - y \cos\theta$

- θ controls the filter orientation
- σ controls the localization of the filter
- λ controls the filter frequency

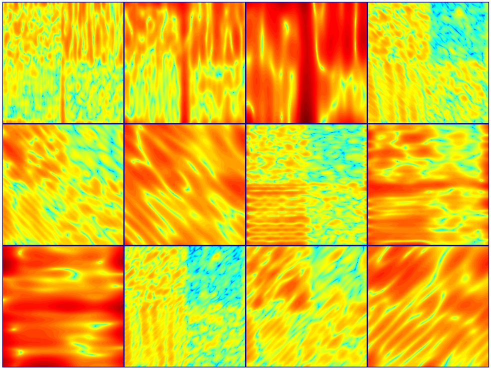
Gabor filters



Convolution by a Gabor filter



Convolution by a Gabor filter



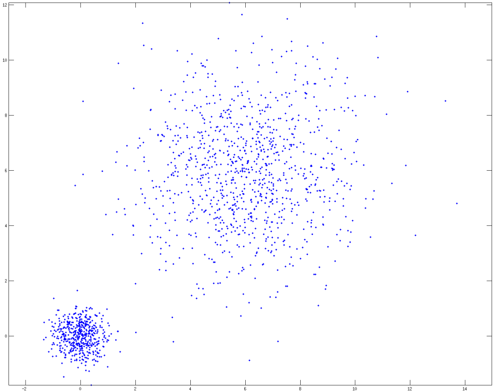
Classical segmentation algorithm

- Supervised classification / Unsupervised classification
- Data in $\mathbb{R}^d$ but we'll visualize 2D examples only.
- Classical examples we'll look at: K-means, mean-shift, Expectation Maximization

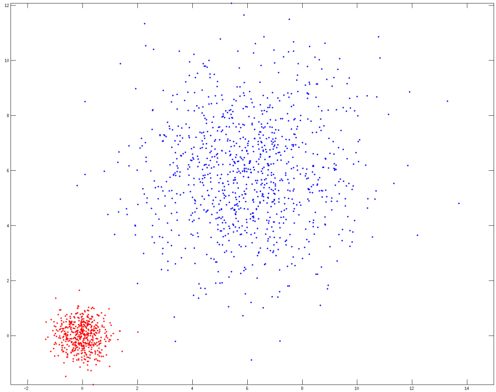
Recent advances

Deep Learning methods learn object descriptions (feature vectors). ImageNet Benchmark: AlexNet [Krizhevsky et al. 2012] ... [Chen et al. 2023]

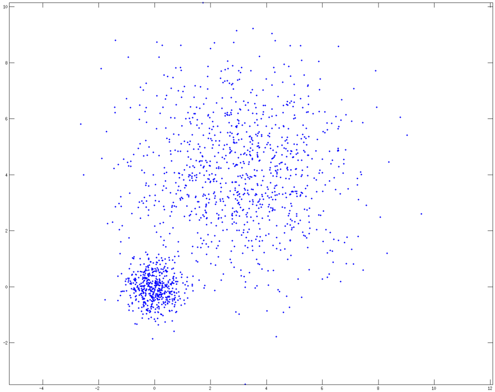
Classification in $\mathbb{R}^2$ (for easier visualization)



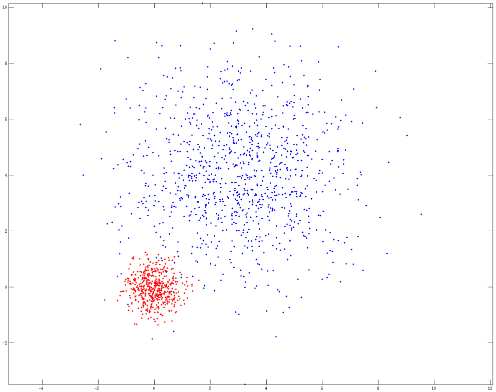
Classification in $\mathbb{R}^2$ (for easier visualization)



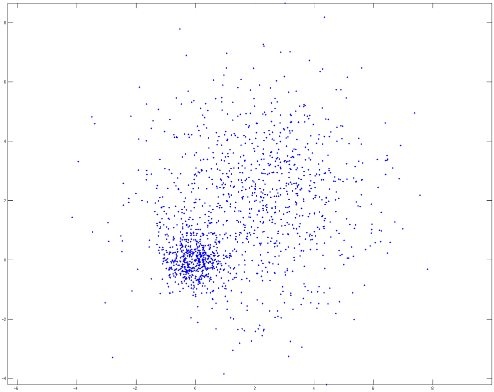
Classification in $\mathbb{R}^2$ (for easier visualization)



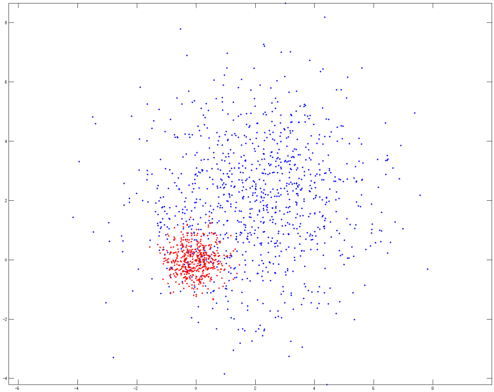
Classification in $\mathbb{R}^2$ (for easier visualization)



Classification in $\mathbb{R}^2$ (for easier visualization)



Classification in $\mathbb{R}^2$ (for easier visualization)



Outline

K-Means

- Goal: Extract classes (or *clusters*) from a set of points (Group the points into clusters)
- In this algorithm a class is represented by a special element called class representative of cluster center.

K-means

Principle

Let $(x_i)_{i=1\cdots n} \in \mathbb{R}^d$ a set of n points, K a given cluster number and $(y_k)_{k=1\cdots K}$ the cluster centers, then the label k_0 of a point x_i is:

$$k_0 = argmin_{k\in 1\cdots K}\|y_k - x_i\|^2$$

- Goal: Find the cluster centers y_k AND the labels of points x_i

Algorithm

- If we know the cluster centers, can we compute the labels?

Algorithm

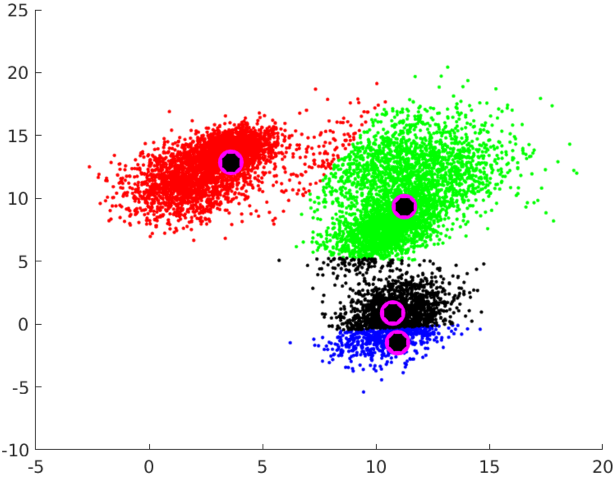
- If we know the cluster centers, can we compute the labels?
- If we know the labels, can we compute the cluster centers?

Algorithm

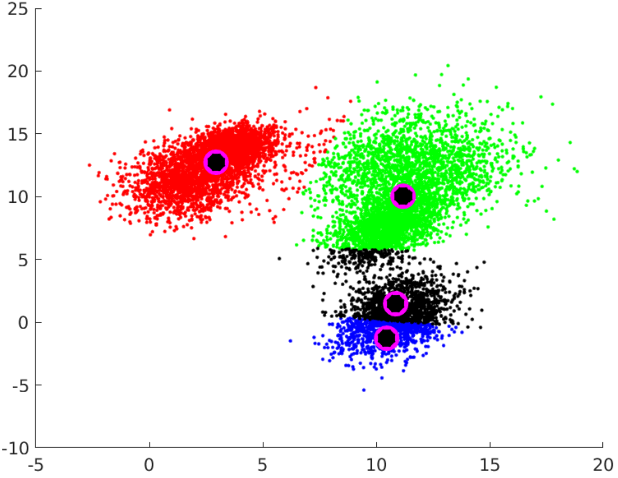
Algorithm 1: Algorithme K-Means

Data: $(x_i)_{i=1\cdots n} \in \mathbb{R}^d$, a number of classes K
Result: An assignment for $(l_i)_{i=1\cdots n} \in \{1 \cdots K\}$ and representatives $(y_k)_{i=1\cdots K}$

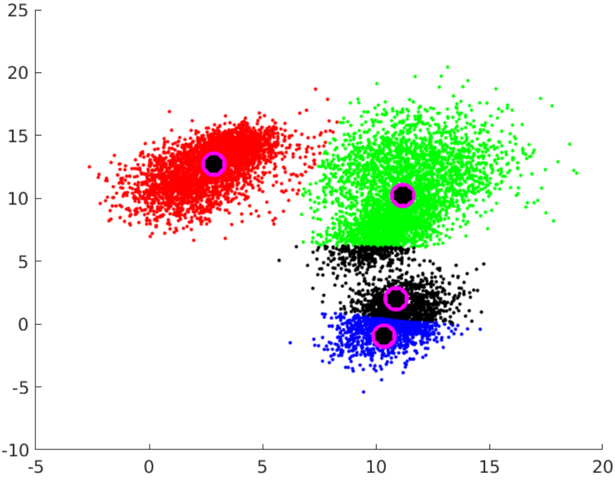
Start with random y_k drawn from x_i;
do
    Assign to each x_i the label corresponding to its nearest y_k;
    For each k, update y_k as the barycenter of the x_i with label k;
Until *Convergence*;



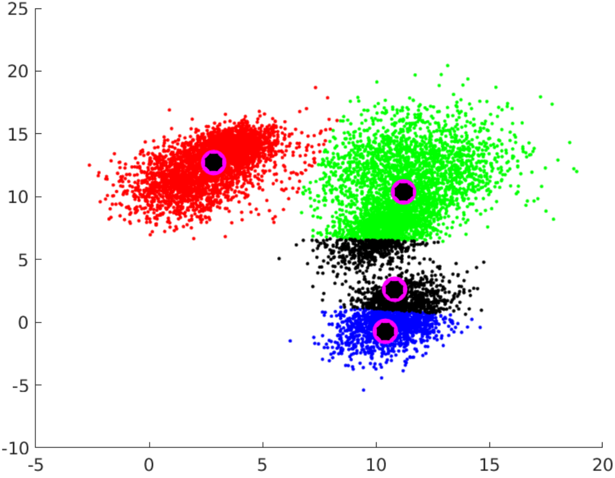
Iteration 1



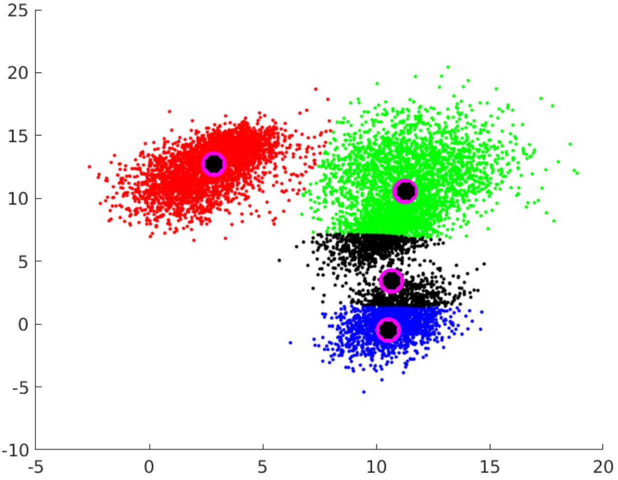
Iteration 2



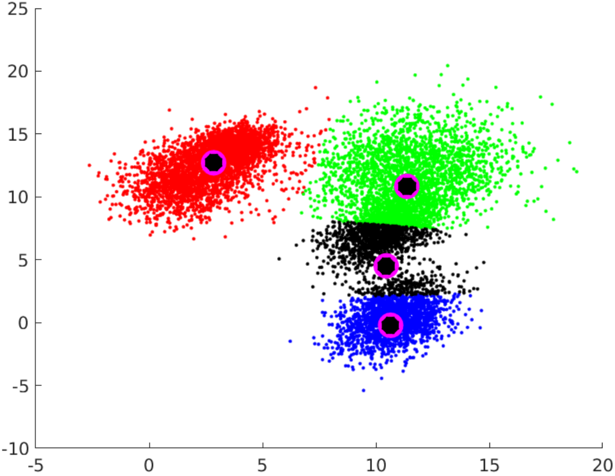
Iteration 3



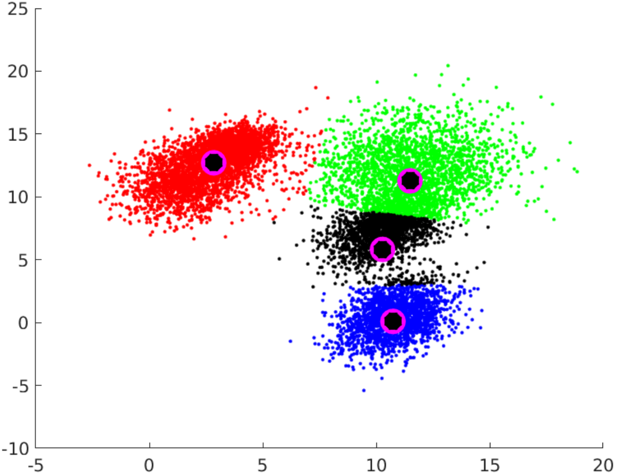
Iteration 4



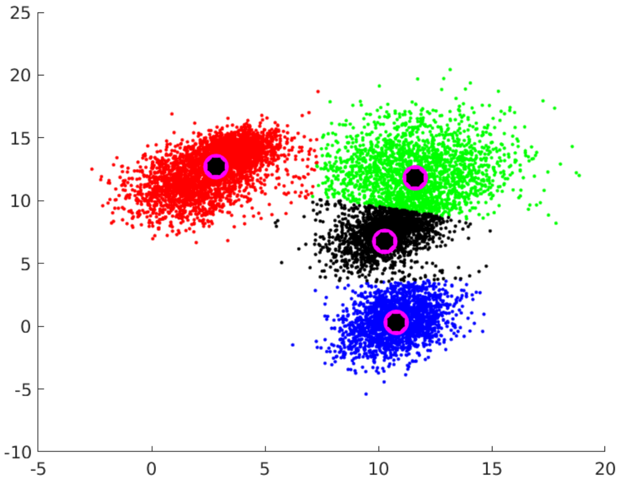
Iteration 5



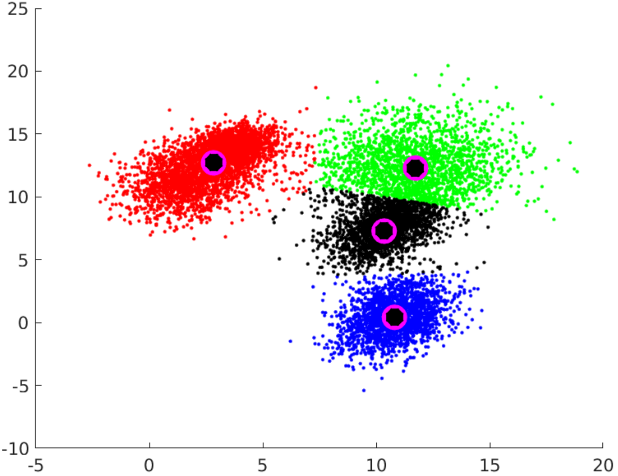
Iteration 6



Iteration 7



Iteration 8

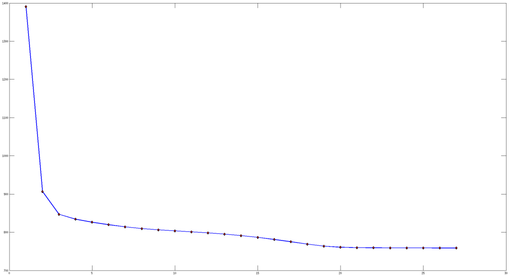


Iteration 9

Algorithm convergence?

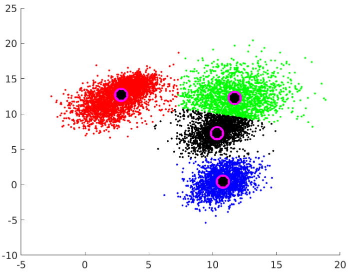
Measure the time when the clusters (or the labels) do not change

- Average motion of the cluster center is close to 0
- Better: No labeling is changed ($\rightarrow$ the cluster center will not move at the next iteration)

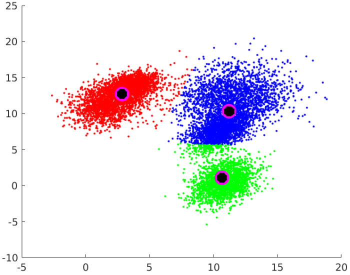


Algorithm initialization

- A random choice in the set of x_i
- A random choice in the *domain* of the x_i?

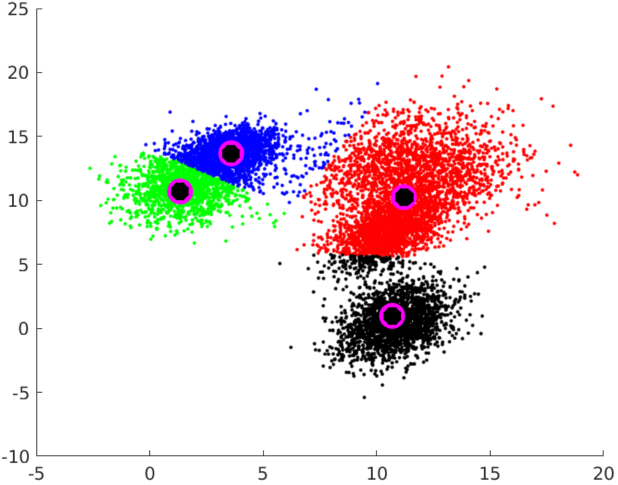


Random from the set



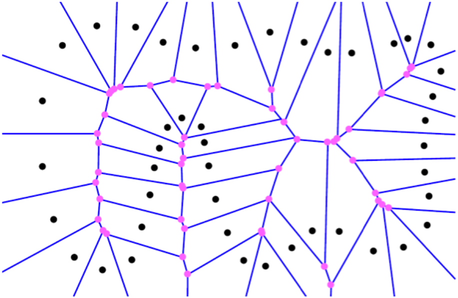
Random in the domain

Convergence... To a local minimum



A small detour by Computational Geometry

- The Voronoi Diagram of S is a partition of space into regions $V(p)$ ($p \in S$) such that all points in $V(p)$ are closer to p than any other point in S.
- For a vertex, we can draw an empty circle that just touches the three points in S around the vertex.
- Each Voronoi vertex is in one-to-one correspondence with a Delaunay triangle



Link between K-means and the Voronoi Diagram

Voronoi Diagram

In $\mathbb{R}^d$ using the L^2 distance, the boundary of a cell is a hyperplane.

K-means

- The assignation step assigns each point to the center (seed) of their Voronoi cell.
- The positions of the seeds are then recomputed.

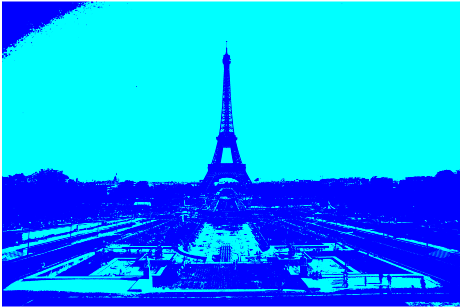
Color image segmentation

Color clouds

Original

Color image segmentation

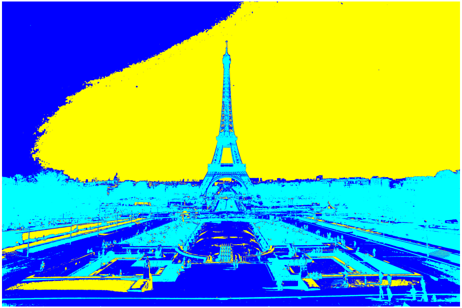
Color clouds



2 classes

Color image segmentation

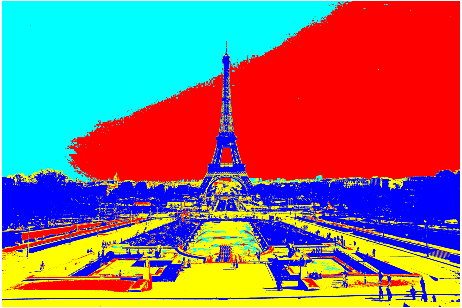
Color clouds



3 classes

Color image segmentation

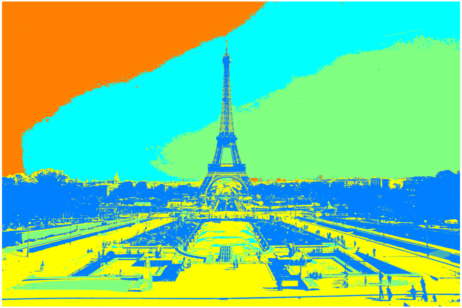
Color clouds



4 classes

Color image segmentation

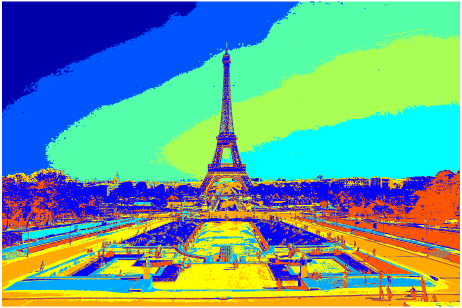
Color clouds



5 classes

Color image segmentation

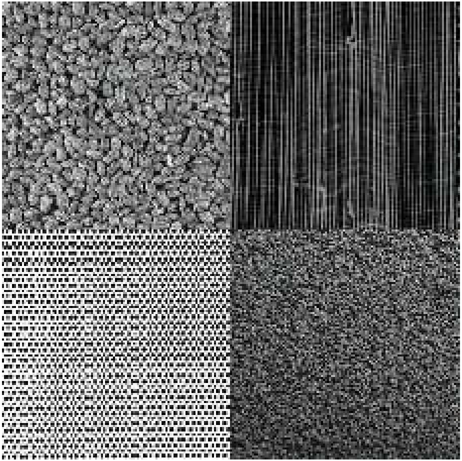
Color clouds



10 classes

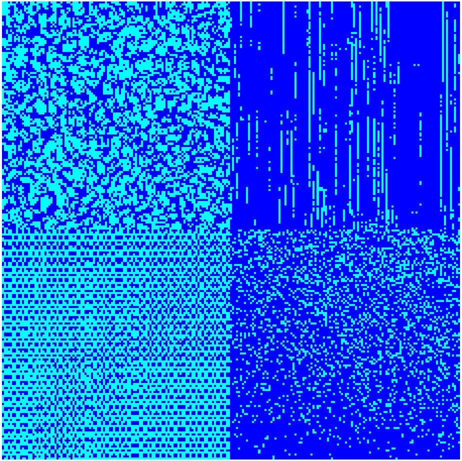
On textures

Color clouds



On textures

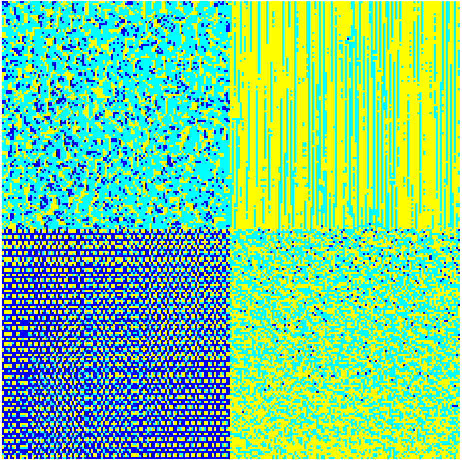
Color clouds



2 classes

On textures

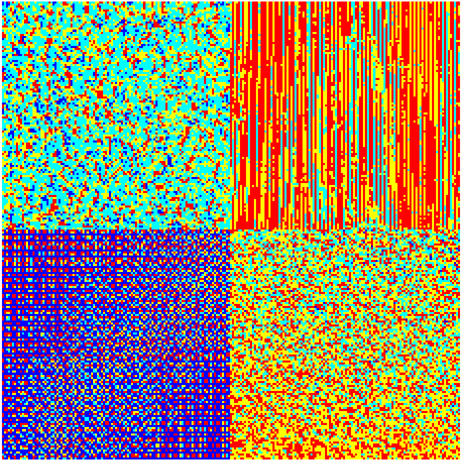
Color clouds



3 classes

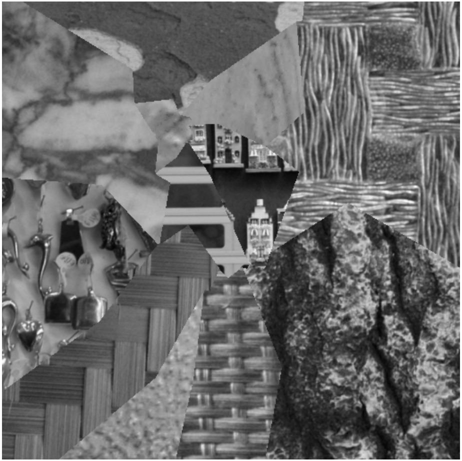
On textures

Color clouds



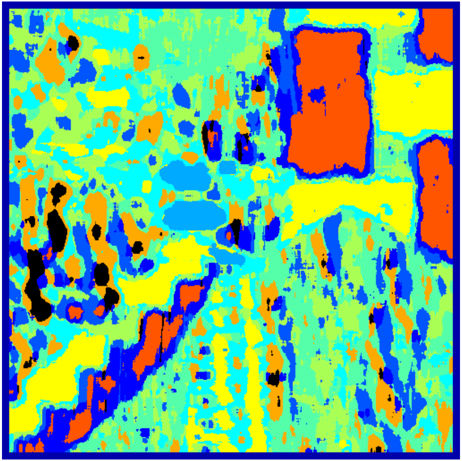
4 classes

On textures



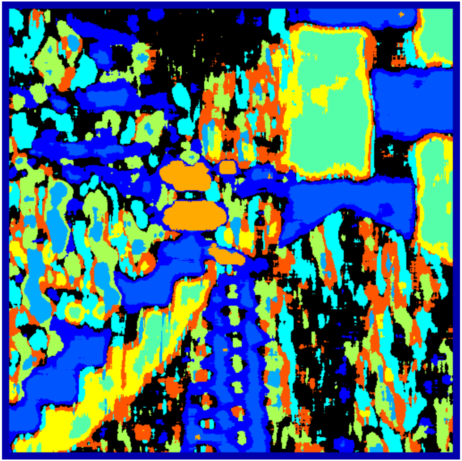
On textures

With local histograms of gradient orientations (size 16x16)



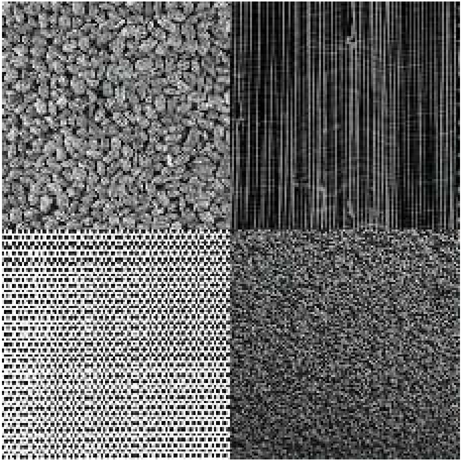
On textures

With local histograms of gradient orientations (size 32x32)



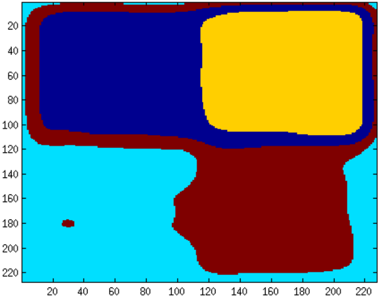
On textures

With Gabor filters



On textures

With Gabor filters



Conclusion on K-means

- It is necessary to know the number of classes K
- Strong dependency on the initialization
- Assumes that classes can be separated by an hyperplane.

Dropping the hyperplane assumption

- Embed the data in a space where the classes will be indeed separated by hyperplanes (*kernel trick*)
- Use the K-means algorithm in this space.

Outline

Mean-shift

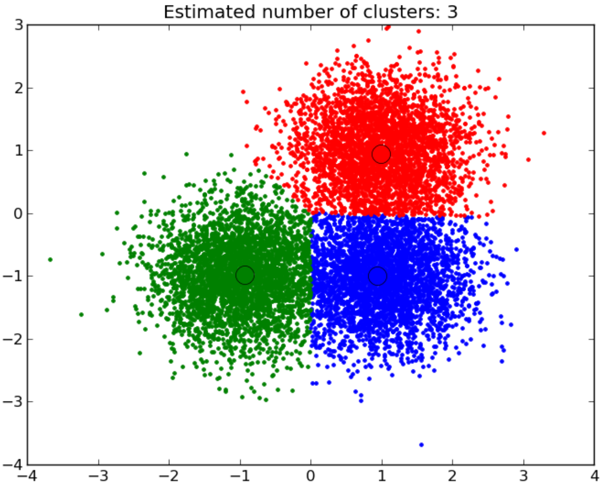


Figure: Data Example

Mean-shift

- Idea: clusters correspond to high point densities areas
- Points will *evolve* and *be attracted* towards high density areas
- When the convergence is reached we'll deduce the classification

"Particle filter"

Points are particles moving in $\mathbb{R}^d$

Mean-shift

Definition

Let $(x_i)_i$ be a set of observations in $\mathbb{R}^d$ Let K be a *kernel*, an estimator of the local point density at x:

$$f(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K(\frac{x - x_i}{h})$$

A word on kernels

A kernel K is a function defined on $\mathbb{R}^d$ with values in $\mathbb{R}$ iff there exists a function $k : \mathbb{R}^+ \rightarrow \mathbb{R}$ such that:

- $K(x) = k(\|x\|^2)$
- k is nonnegative
- k is decreasing
- k is piecewise continuous and $\int_{\mathbb{R}^+} k(x)dx < \infty$

We will assume that $\int_{x \in \mathbb{R}^d} k(x)dx = 1$, and:

$$K(x) = k(\|x\|^2)$$

Kernel examples:

- Gaussian Kernel $K(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp -(\frac{\|x\|^2}{2\sigma^2})$
- Flat Kernel: $K(x) = \mathbb{1}_{\|x\|^2 < r^2}(x)$

Computing the density extrema

- Need to solve for $\nabla f(x) = 0$:
- Let $g \equiv -k'$

Density gradient

$$\nabla f(x) = [\frac{2}{cnh^{d+2}} \sum_{i=1}^{n} g((\frac{\|x - x_i\|}{h})^2)](\frac{\sum_{i=1}^{n} g((\frac{\|x-x_i\|}{h})^2)x_i}{\sum_{i=1}^{n} g((\frac{\|x-x_i\|}{h})^2)} - x)$$

- The gradient expression can be understood easily

Computing the density extrema

- Need to solve for $\nabla f(x) = 0$:
- Let $g \equiv -k'$

Density gradient

$$\nabla f(x) = [\frac{2}{cnh^{d+2}} \sum_{i=1}^{n} g((\frac{\|x - x_i\|}{h})^2)](\frac{\sum_{i=1}^{n} g((\frac{\|x-x_i\|}{h})^2)x_i}{\sum_{i=1}^{n} g((\frac{\|x-x_i\|}{h})^2)} - x)$$

- Module

Computing the density extrema

- Need to solve for $\nabla f(x) = 0$:
- Let $g \equiv -k'$

Density gradient

$$\nabla f(x) = [\frac{2}{cnh^{d+2}} \sum_{i=1}^{n} g((\frac{\|x - x_i\|}{h})^2)](\frac{\sum_{i=1}^{n} g((\frac{\|x-x_i\|}{h})^2) x_i}{\sum_{i=1}^{n} g((\frac{\|x-x_i\|}{h})^2)} - x)$$

- Weighted average of the neighbors

Computing the density extrema

- Need to solve for $\nabla f(x) = 0$:
- Let $g \equiv -k'$

Density gradient

$$\nabla f(x) = [\frac{2}{cnh^{d+2}} \sum_{i=1}^{n} g((\frac{\|x - x_i\|}{h})^2)](\frac{\sum_{i=1}^{n} g((\frac{\|x-x_i\|}{h})^2)x_i}{\sum_{i=1}^{n} g((\frac{\|x-x_i\|}{h})^2)} - x)$$

- Vector from x to the weighted average of the neighbors

Algorithm

Algorithm 2: Mean-Shift

Data: A set of points x_i, kernel size h, threshold ε

Result: A set of clusters c_i and labels l_i

for $j = 1 \cdots n$ **do**

  $x_j^0 = x_j$;

$t = 0$;

while *error* $> \varepsilon$ **do**

  for $j = 1 \cdots n$ **do**

    $m(x_j^t) = \frac{\sum_{i=1}^n g((\frac{\|x_j^t - x_i\|}{h})^2) x_i}{\sum_{i=1}^n g((\frac{\|x_j^t - x_i\|}{h})^2)}$;

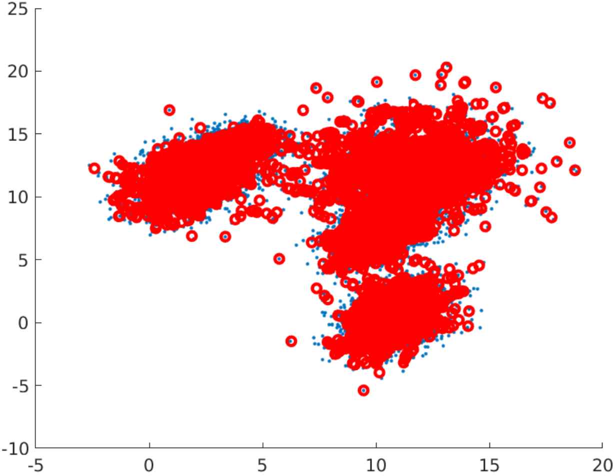
    $x_j^{t+1} = m(x_j^t)$;

  $error = \frac{1}{n} \sum_j \|m(x_j^t) - x_j^t\|$;

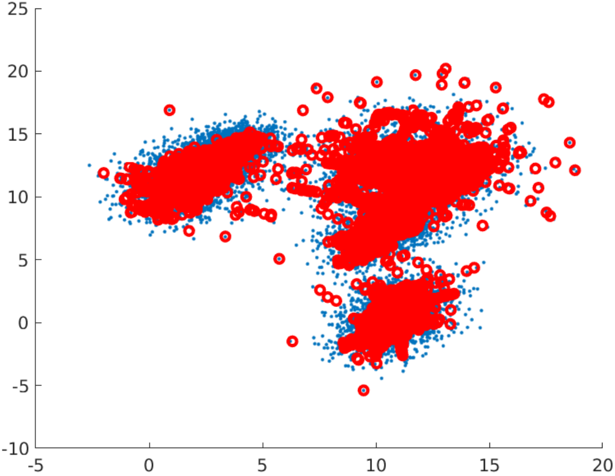
  $t = t + 1$;

Group x_i^T by position;

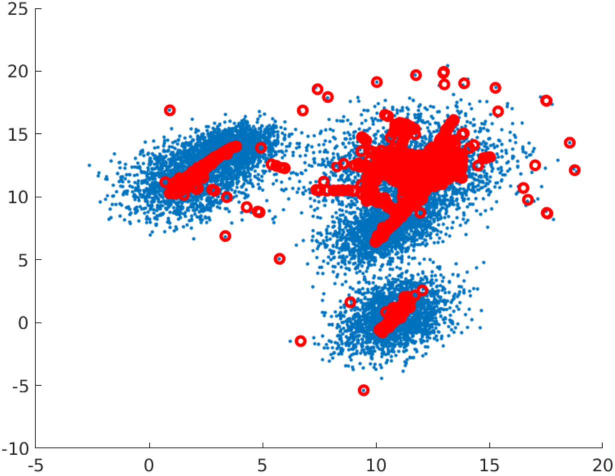
Assign x_i to the cluster of x_i^T;



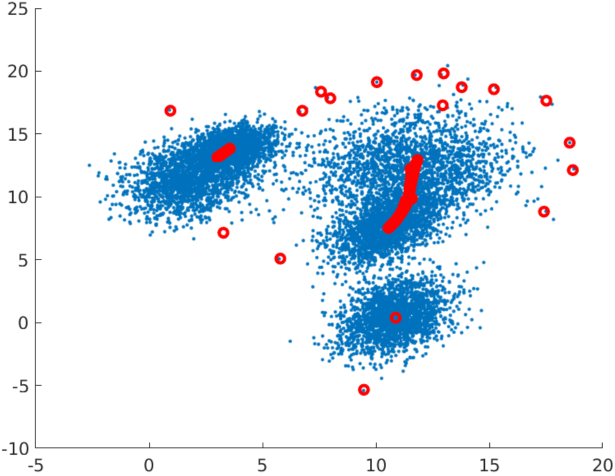
Iteration 2



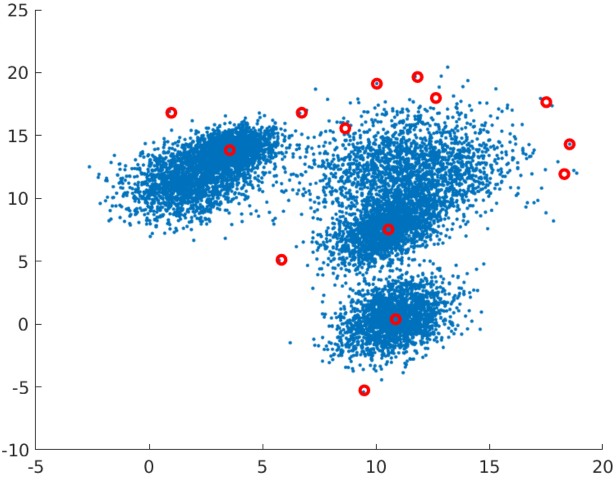
Iteration 3



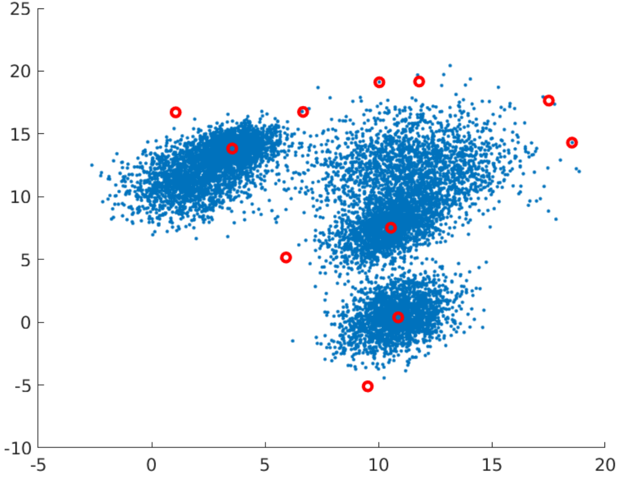
Iteration 5



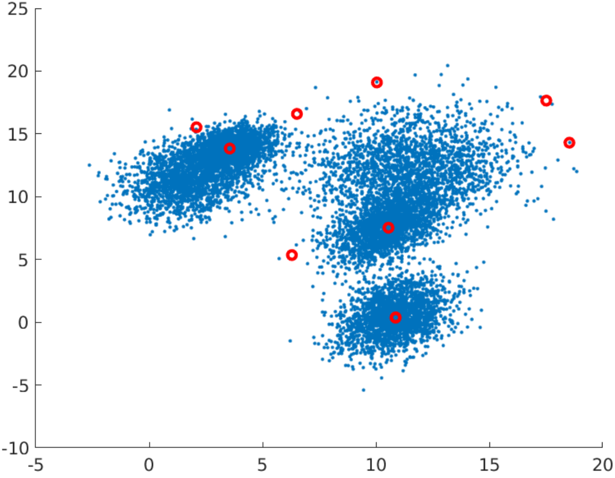
Iteration 8



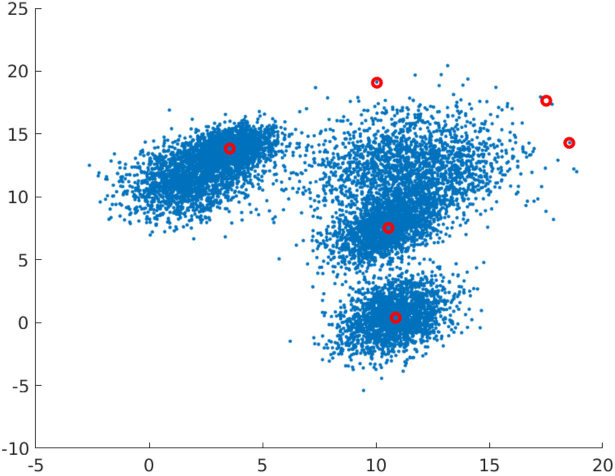
Iteration 11



Iteration 14



Iteration 17



Iteration 20

Analysis

- Pro: No need to choose the number of classes

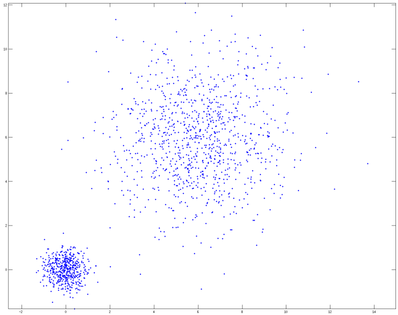
Analysis

- Pro: No need to choose the number of classes
- Pro: Guaranteed convergence to a density extrema

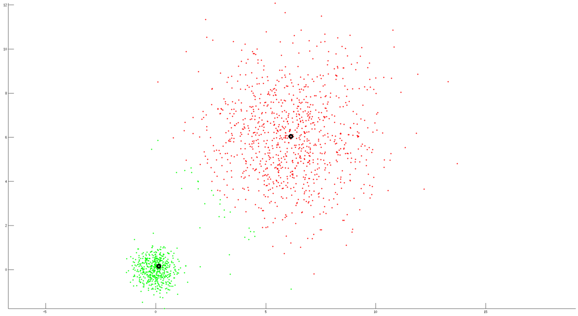
Analysis

- Pro: No need to choose the number of classes
- Pro: Guaranteed convergence to a density extrema
- Con: Needs post-filtering for small density extrema

Meanshift - K-means

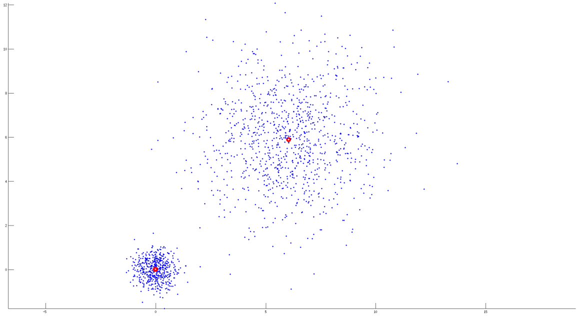


Meanshift - K-means



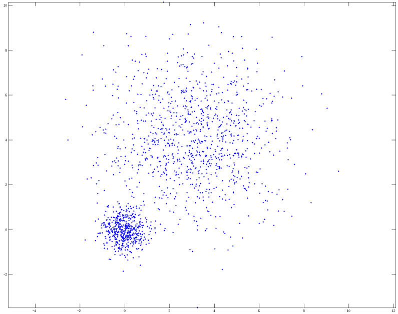
Kmeans

Meanshift - K-means

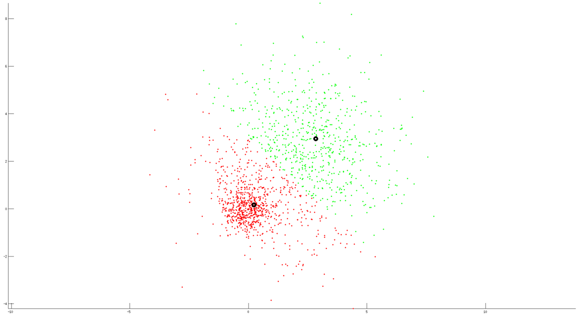


Meanshift

Meanshift - K-means

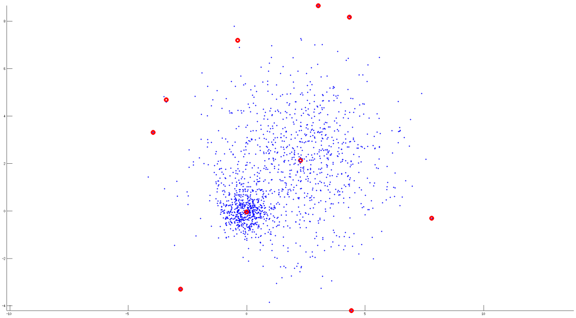


Meanshift - K-means



Kmeans

Meanshift - K-means



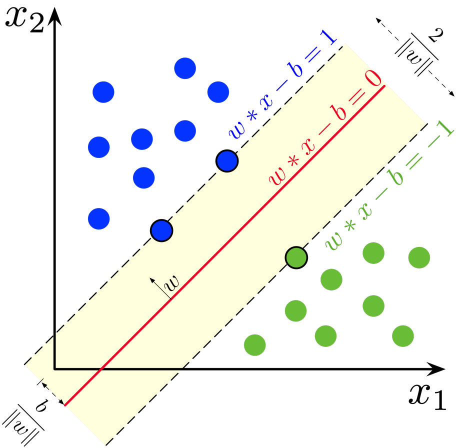
Meanshift

Outline

Support Vector Machine

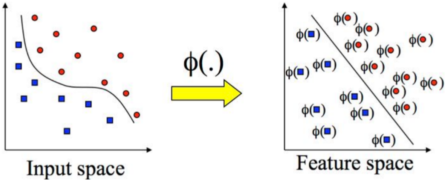
Large margin binary classifier

Find an hyperplane separating the two classes maximizing the *margin*



Support Vector Machine

- Works well when the classes are linearly separable. What if it's not the case?



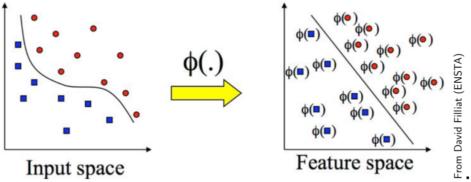
From David Filliat (ENSTA)

Support Vector Machine

- Works well when the classes are linearly separable. What if it's not the case?

Kernel trick

Find a function Φ such that the two classes of $(\phi(x_i), l_i)_i$ are linearly separable.

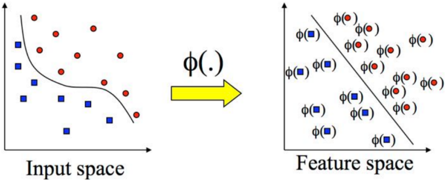


Support Vector Machine

- Works well when the classes are linearly separable. What if it's not the case?

Kernel trick

Find a function Φ such that the two classes of $(\phi(x_i), l_i)_i$ are linearly separable.



From David Filliat (ENSTA)

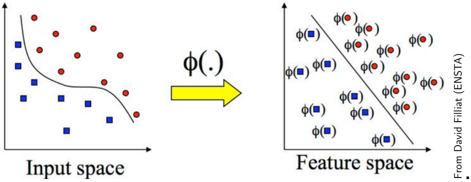
- Problem: how do we design Φ? Manually

Support Vector Machine

- Works well when the classes are linearly separable. What if it's not the case?

Kernel trick

Find a function Φ such that the two classes of $(\phi(x_i), l_i)_i$ are linearly separable.



- Problem: how do we design Φ? Manually or that's where Deep Learning methods come in handy.

Optimization

- Equation of the separating hyperplane: $w^T x + b = 0$
- If $w^T x_i + b > 0$ then $l_i = 1$, and if $w^T x_i + b < 0$ then $l_i = -1$,
- Decision function: $f(x) = sign(w^T x + b)$.

Maximal margin

Maximize the distance between hyperplanes $w^T x + b = \pm 1$. Decision function: $l_i = 1$ if $w^T x + b \geq 1$, $l_i = -1$ if $w^T x + b \leq 1$.

What is the size of the margin between the two hyperplanes?

Optimization (continued)

Optimization problem

$$Minimize_{w,b} \frac{1}{2} w^T w$$

subject to $\forall i, l_i(w^T x_i + b) \geq 1$

Optimization (continued)

Optimization problem

$$Minimize_{w,b} \frac{1}{2} w^T w$$

subject to $\forall i, l_i(w^T x_i + b) \geq 1$

Allow for some training errors: samples that violates the margin condition

Reformulation

$$Minimize_{w,b} \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i$$

subject to $\forall i, l_i(w^T x_i + b) \geq 1 - \xi_i$ and $\forall i, \xi_i \geq 0$

Optimization (continued)

- Notice that constraints are equivalent to setting $\xi_i = \max(0, 1 - l_i(w^T x_i + b))$

Optimization (continued)

- Notice that constraints are equivalent to setting $\xi_i = \max(0, 1 - l_i(w^T x_i + b))$
- set $x_i = [x_i; 1]$ and $w = [w; b]$ to simplify.

Optimization (continued)

- Notice that constraints are equivalent to setting $\xi_i = \max(0, 1 - l_i(w^T x_i + b))$
- set $x_i = [x_i; 1]$ and $w = [w; b]$ to simplify.

Reformulation

$$Minimize_w J(w) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \max(0, 1 - l_i(w^T x_i))$$

Optimization (continued)

- Notice that constraints are equivalent to setting $\xi_i = \max(0, 1 - l_i(w^T x_i + b))$
- set $x_i = [x_i; 1]$ and $w = [w; b]$ to simplify.

Reformulation

$$Minimize_w J(w) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \max(0, 1 - l_i(w^T x_i))$$

- Unconstrained optimization

Optimization (continued)

- Notice that constraints are equivalent to setting $\xi_i = \max(0, 1 - l_i(w^T x_i + b))$
- set $x_i = [x_i; 1]$ and $w = [w; b]$ to simplify.

Reformulation

$$Minimize_w J(w) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \max(0, 1 - l_i(w^T x_i))$$

- Unconstrained optimization
- Gradient descent $w_{t+1} = w_t - \nu_t \nabla_w J(w_t)$

Optimization (continued)

Stochastic gradient descent

Gradient computed per sample:

$$J(w, x_i, l_i) = \frac{1}{2} w^T w + C \max(0, 1 - l_i(w^T x_i))$$

- initialization $w_0 = 0$
- While not converged
  - For each training sample (x_i, l_i)
  - Compute $\nabla_w J(w_t, x_i, l_i)$
  - $w_{t+1} = w_t - \nu_t \nabla_w J(w_t, x_i, l_i)$
- Return w

Optimization (continued)

Problem

J is not differentiable!

Optimization (continued)

Problem

J is not differentiable! Strategy:

- $\nabla J(w, x_i, l_i) = w$ if $\max(0, 1 - l_i w^T x_i) = 0$
- $\nabla J(w, x_i, l_i) = w - C l_i x_i$ otherwise

Optimization (continued)

Problem

J is not differentiable! Strategy:

- $\nabla J(w, x_i, l_i) = w$ if $\max(0, 1 - l_i w^T x_i) = 0$
- $\nabla J(w, x_i, l_i) = w - C l_i x_i$ otherwise

- Initialization: $w_0 = 0$
- While not converged
  - For each training sample (x_i, l_i)
  - If $l_i w_t^T x_i \leq 1$, $w_{t+1} = (1 - \nu_t) w_t + \nu_t C l_i x_i$
  - Otherwise $w_{t+1} = (1 - \nu_t) w_t$
- Return w

Optimization (continued)

Problem

J is not differentiable! Strategy:

- $\nabla J(w, x_i, l_i) = w$ if $\max(0, 1 - l_i w^T x_i) = 0$
- $\nabla J(w, x_i, l_i) = w - C l_i x_i$ otherwise

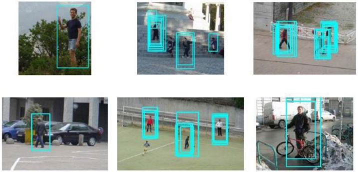
- Initialization: $w_0 = 0$
- While not converged
  - For each training sample (x_i, l_i)
  - If $l_i w_t^T x_i \leq 1$, $w_{t+1} = (1 - \nu_t) w_t + \nu_t C l_i x_i$
  - Otherwise $w_{t+1} = (1 - \nu_t) w_t$
- Return w

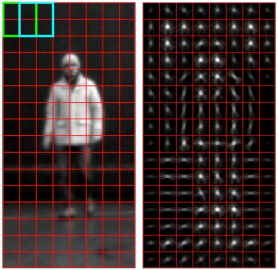
Stochastic Gradient Descent

Shuffle the training set before picking an example

What's wrong in the above derivation?

Pedestrian detection (Dalal & Triggs 2005)





- Descriptor of each image: Histogram of oriented gradients
- Classified using a linear svm (soft: allows for some margin violation during training).

Conclusion

- A way to classify information encoded in various ways
- The choice of the encoding is crucial (color? color and localization? Filter Bank Response?)