POLITECNICO
DI TORINO

AI ML

Artificial Intelligence and Machine Learning

A.A. 2020/2021

Tatiana Tommasi

T.A. Silvia Bucci

T.A.: **Silvia Bucci**

E-mail*: **silvia.bucci@polito.it**

*actually via the slack channel you can get faster answers than via email

Supervised Learning with SVM

Summary :

- (x,y) : {data, labels}
- search for a linear classifier to separate y:+1 from y:-1

- differently from the perceptron: batch learning (not online) and here we include a large margin condition. So not all the linear classifiers are equally good.

- The best classifier is described in a sparse way -- only the support vectors are relevant, no need to keep the whole set of training data (fundamental difference wrt KNN).

- you already know
  - primal and dual optimization problems and what does it mean to solve them
  - what are hard and soft margins for non-linear separations
  - how to use the different kernels
  - how to go over binary classification (extensions to multiclass)

We can do more beside classification

Supervised Learning - SVM variants

- Support Vector Data Description (SVDD) and One class SVM (OSVM) - two different strategies to identify outliers
- Ranking SVM - learn to sort samples, not just annotate them

Unsupervised Learning

- Clustering (k-means algorithm)

Just to conclude - quickly back on two perceptron exercises with solution

Outlier / Novelty Detection

Inherently binary problem but with peculiarities

- we really know informations about one (positive) class
- issue: all positive samples are alike but each negative sample is negative in its own way
- desirable every time we do not have full control on what is negative

Applications
intrusion detection, fraud detection, fault detection, robotics, medical diagnosis, e-commerce, and more...

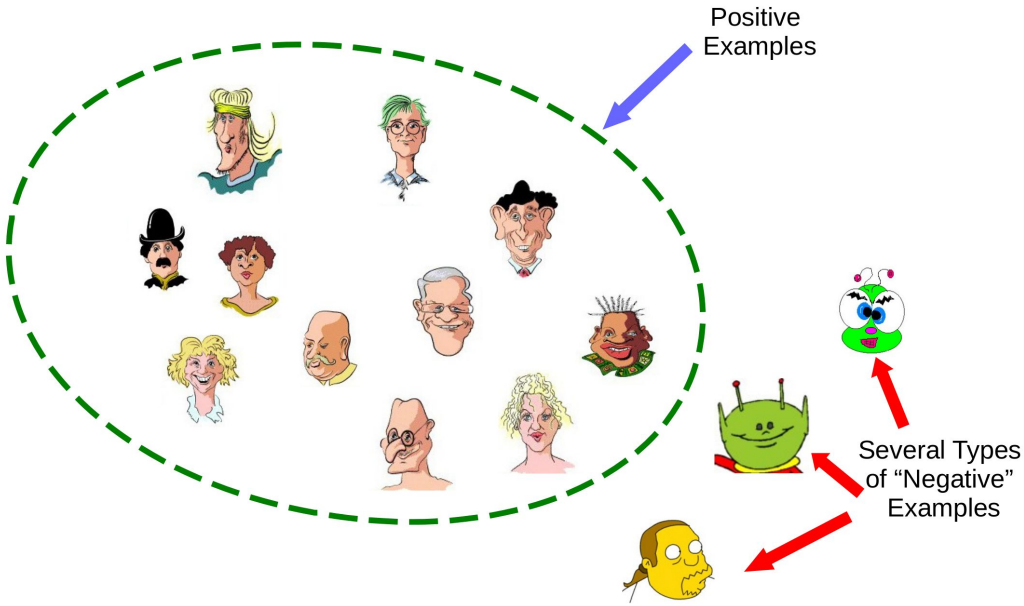


Figure Credit: Refael Chickvashvili

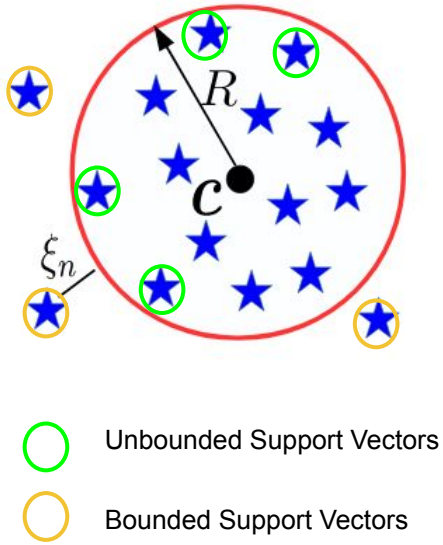
SVDD [Tax and Duin, ML 2004]

- Find the minimal circumscribing sphere around the data in the feature space
- Allow slacks (most of the points should lie inside the sphere, not all)

$$\text{center } c \in \mathbb{R}^d \qquad \text{radius } R \in \mathbb{R}^+$$

$$\min_{\substack{R\in\mathbb{R},c\in\mathbb{R}^d \\ \xi_1,\ldots,\xi_n\in\mathbb{R}^+}} R^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i$$

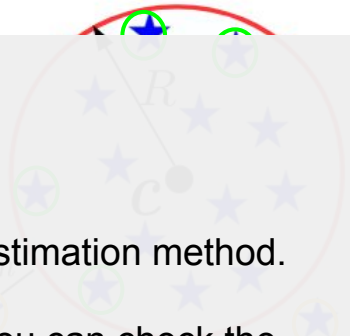
$$\text{subject to} \quad \|x_i - c\|^2 \leq R^2 + \xi_i \quad \text{for } i = 1,\ldots,n.$$



$\nu \in (0, 1]$ controls how many outliers are possible: the smaller v the fewer points will be outliers, since violating the constraint above will be penalized more heavily the optimization

SVDD [Tax and Duin, ML 2004]

- Find the minimal circumscribing sphere around the data in the feature space



- Allow slacks (most of the points should lie inside the sphere, not all)

$$\text{center } c \in \mathbb{R}^d \qquad \text{radius } R \in \mathbb{R}^+$$

$$\min_{c,R,\xi_1,\ldots,\xi_n \in \mathbb{R}^+} R^2 + \frac{1}{\nu n}\sum_{i=1}^{n} \xi_i$$

$\nu \in (0,1]$ controls how many outliers are possible: the smaller ν the fewer points will be outliers, since violating the constraint above will be penalized more heavily the optimization

Post-Video Note:

in the video Silvia mentions “Parzen Windows”. That is a non-parametric density estimation method.

We will not go over that method in the course this year. Still, if you are interested you can check the following references

Chapter 2.5.1 in “Pattern Recognition and Machine Learning” book by Bishop.
Chapter 4 in “Pattern Classification” book by Duda, Hart, Stork

SVDD kernel extension

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$$

Hilbert space $\mathcal{H}$ and feature map $\varphi : \mathcal{X} \to \mathcal{H}$

una generalizzazione dell'euclidian space ma con infinite dimensions

$$\min_{\substack{R \in \mathbb{R}, c \in \mathcal{H} \\ \xi_1, \ldots, \xi_n \in \mathbb{R}^+}} R^2 + \frac{1}{\nu n} \sum_{i=1}^{n} \xi_i$$

subject to

$$\|\varphi(x_i) - c\|_{\mathcal{H}}^2 \leq R^2 + \xi_i \quad \text{for } i = 1, \ldots, n.$$

SVDD kernel extension

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$$

Hilbert space $\mathcal{H}$ and feature map $\varphi : \mathcal{X} \to \mathcal{H}$

$$\min_{\substack{R \in \mathbb{R}, c \in \mathcal{H} \\ \xi_1, \ldots, \xi_n \in \mathbb{R}^+}} R^2 + \frac{1}{\nu n} \sum_{i=1}^{n} \xi_i$$

the slacks are positive values

subject to

$$\|\varphi(x_i) - c\|_{\mathcal{H}}^2 \leq R^2 + \xi_i \quad \text{for } i = 1, \ldots, n.$$

SVDD Lagrangian

vogliamo semplificare

$$\mathcal{L}\{R, c, \xi, \alpha, \mu\} = R^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i\{(\varphi(x_i) - c)^\top(\varphi(x_i) - c) - R^2 - \xi_i\} - \sum_{i=1}^{n}\mu_i\xi_i$$

Karush–Kuhn–Tucker (KKT) conditions

$$\frac{\partial\mathcal{L}}{\partial R} = 2R - 2R\sum_{i=1}^{n}\alpha_i = 0 \quad \Rightarrow \quad \sum_{i=1}^{n}\alpha_i = 1$$

$$\frac{\partial\mathcal{L}}{\partial c} = 2\sum_{i=1}^{n}\alpha_i c - 2\sum_{i=1}^{n}\alpha_i\varphi(x_i) \quad \Rightarrow \quad \boxed{c = \sum_{i=1}^{n}\alpha_i\varphi(x_i)}$$

$$\frac{\partial\mathcal{L}}{\partial\xi_i} = \frac{1}{\nu n} - \alpha_i - \mu_i = 0 \quad \Rightarrow \quad \alpha_i + \mu_i = \frac{1}{\nu n}$$

the sphere center is a linear combination of the transformed data points

SVDD Lagrangian

$$\mathcal{L}\{R, c, \xi, \alpha, \mu\} = R^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i\{(\varphi(x_i) - c)^\top(\varphi(x_i) - c) - R^2 - \xi_i\} - \sum_{i=1}^{n}\mu_i\xi_i$$

$$\mathcal{L}\{R, c, \xi, \alpha, \mu\} = R^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i k(x_i, x_i) - 2\sum_{i=1}^{n}\alpha_i\varphi(x_i)^\top c + c^\top c\sum_{i=1}^{n}\alpha_i + - R^2\sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}(\alpha_i + \mu_i)\xi_i$$

SVDD Lagrangian

$$\mathcal{L}\{R, c, \xi, \alpha, \mu\} = R^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i$$

$$+ \sum_{i=1}^{n}\alpha_i k(x_i, x_i) - 2\underbrace{\sum_{i=1}^{n}\alpha_i \varphi(x_i)}_{c}^{\top} c + c^{\top} c \underbrace{\sum_{i=1}^{n}\alpha_i}_{1} +$$

$$- R^2\sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}(\alpha_i + \mu_i)\xi_i$$

$$\mathcal{L}\{R, c, \xi, \alpha, \mu\} = \sum_{i=1}^{n}\alpha_i k(x_i, x_i) - c^{\top} c \quad = \sum_{i=1}^{n}\alpha_i k(x_i, x_i) - \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i \alpha_j k(x_i, x_j)$$

$$\text{s.t. } \alpha_i + \mu_i = \frac{1}{\nu n}, \quad \sum_{i=1}^{n}\alpha_i = 1, \quad \alpha_i \geq 0 \quad \forall i \qquad \Rightarrow 0 \leq \alpha_i \leq \frac{1}{\nu n}$$

SVDD Lagrangian

$$\mathcal{L}\{R, c, \xi, \alpha, \mu\} = R^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i$$

$$+ \sum_{i=1}^{n}\alpha_i k(x_i, x_i) - 2\underbrace{\sum_{i=1}^{n}\alpha_i \varphi(x_i)}_{c}{}^{\top} c + c^{\top} c \underbrace{\sum_{i=1}^{n}\alpha_i}_{1} +$$

$$- R^2\sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}(\alpha_i + \mu_i)\xi_i$$

$$\mathcal{L}\{R, c, \xi, \alpha, \mu\} = \sum_{i=1}^{n}\alpha_i k(x_i, x_i) - c^{\top} c \quad = \left\{\sum_{i=1}^{n}\alpha_i k(x_i, x_i) - \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i \alpha_j k(x_i, x_j)\right\}$$

maximize over α

$$\text{s.t. } \alpha_i + \mu_i = \frac{1}{\nu n}, \quad \sum_{i=1}^{n}\alpha_i = 1, \quad \alpha_i \geq 0 \quad \forall i \qquad \Rightarrow 0 \leq \alpha_i \leq \frac{1}{\nu n}$$

Difference wrt SVM

dual problem svm

nothing here

you have to know labels, che sono negative o positive

$$\max_\alpha \quad W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle.$$

kernel

$$\text{s.t.} \quad \alpha_i \geq 0, \quad i = 1, \ldots, n$$

$$\sum_{i=1}^{n} \alpha_i y^{(i)} = 0,$$

the support vector need to be balanced. composto da metà della classe negativa, metà della classe positiva

---

dual problem svdd

abbiamo solo un tipo di label che non appare nella formulazione

$$\mathcal{L}\{R, c, \xi, \alpha, \mu\} = \sum_{i=1}^{n} \alpha_i k(x_i, x_i) - c^\top c = \left\{ \sum_{i=1}^{n} \alpha_i k(x_i, x_i) - \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) \right\}$$

maximize over α

$$\text{s.t.} \;\; \alpha_i + \mu_i = \frac{1}{\nu n}, \quad \sum_{i=1}^{n} \alpha_i = 1, \quad \alpha_i \geq 0 \quad \forall i \qquad \Rightarrow 0 \leq \alpha_i \leq \frac{1}{\nu n}$$

Why passing through the Dual?

- Historically, the dual was introduced to use kernels, but it has been later proven that it is not necessary
- Yet, the dual is a convex quadratic problem with linear constraints: easy to optimize!
- Have a look at the SMO and the stochastic dual coordinate ascent algorithm if you want to know more (http://cs229.stanford.edu/notes/cs229-notes3.pdf)

$$\mathcal{L}\{R, c, \xi, \alpha, \mu\} = \sum_{i=1}^{n} \alpha_i k(x_i, x_i) - c^\top c = \left\{ \sum_{i=1}^{n} \alpha_i k(x_i, x_i) - \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) \right\}$$

maximize over α

$$\text{s.t. } \alpha_i + \mu_i = \frac{1}{\nu n}, \quad \sum_{i=1}^{n} \alpha_i = 1, \quad \alpha_i \geq 0 \quad \forall i \qquad \Rightarrow 0 \leq \alpha_i \leq \frac{1}{\nu n}$$

What are the next steps?

- find alpha
- find c using $c = \sum_{i=1}^{n} \alpha_i \varphi(x_i)$
- find R $R^2 = ||\varphi(x) - c||^2$

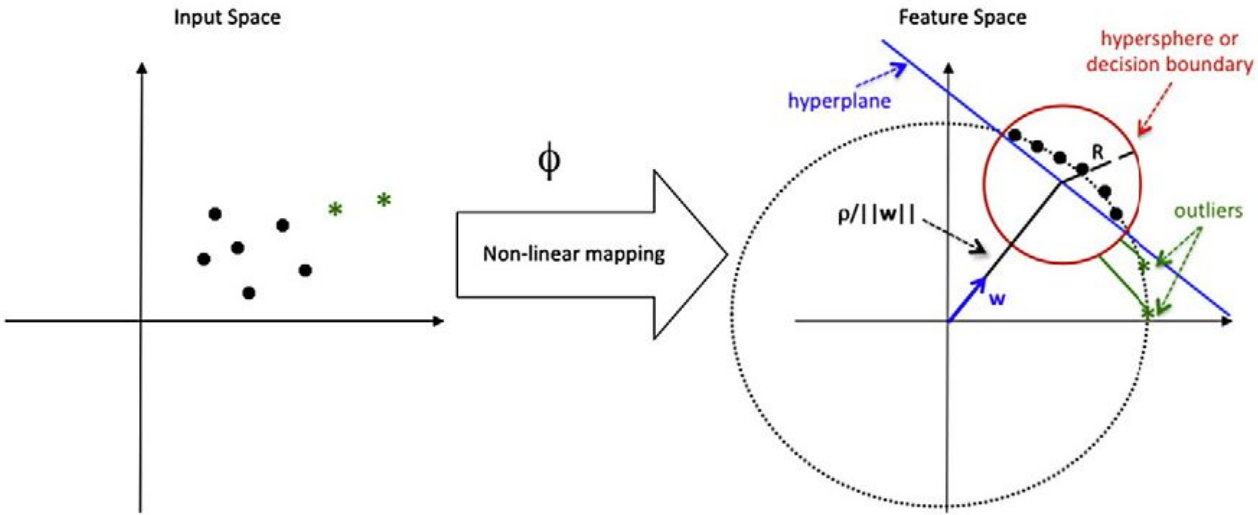
$$R^2 = k(x,x) - 2\sum_{i=1}^{n} \alpha_j k(x_i, x) + \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j)$$

- prediction rule $||\varphi(x^*) - c||^2 - R^2 < 0 \quad \Rightarrow y^* = +1$

per valutare se un sample è inlier or outlier

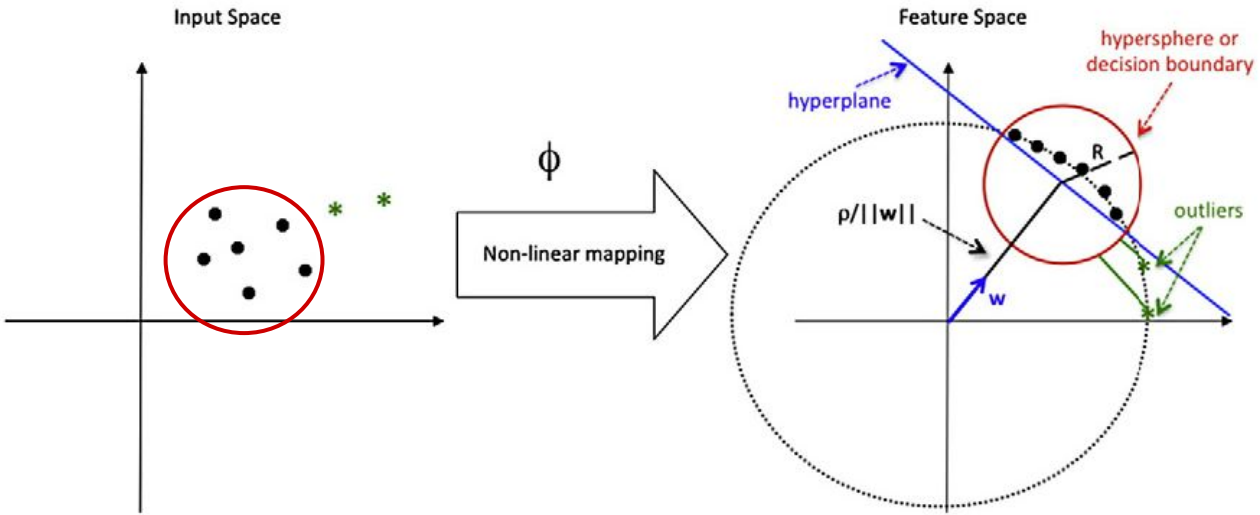
One-Class SVM [Schölkopf et al., 2001]

If all data points have the same feature space norm and can be separated linearly from the origin, finding the minimal enclosing sphere is equivalent to finding the maximal margin hyperplane between the data points and the origin.



One-Class SVM [Schölkopf et al., 2001]

If all data points have the same feature space norm and can be separated linearly from the origin, finding the minimal enclosing sphere is equivalent to finding the maximal margin hyperplane between the data points and the origin.



SVDD and OSVM are the same if all samples lie at the same distance from the origin and are linearly separable from it

One-Class SVM

same feature space norm = translation invariant kernels = k(x,x) is a constant

$$\mathcal{L}\{R, c, \xi, \alpha, \mu\} = \sum_{i=1}^{n} \alpha_i k(x_i, x_i) - \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j)$$

constant

maximize over α

all our observations lie on a sphere

Here the origin represents *all* the outliers (negatives) = low similarity to the training set

It corresponds exactly to SVM where we want to maximize the margin of positive samples from the origin.

One-Class SVM

$$\min_{\substack{R\in\mathbb{R},w\in\mathcal{H}\\ \xi_1,\ldots\xi_n\in\mathbb{R}^+}} \|w\|_{\mathcal{H}}^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi - \rho$$

subject to

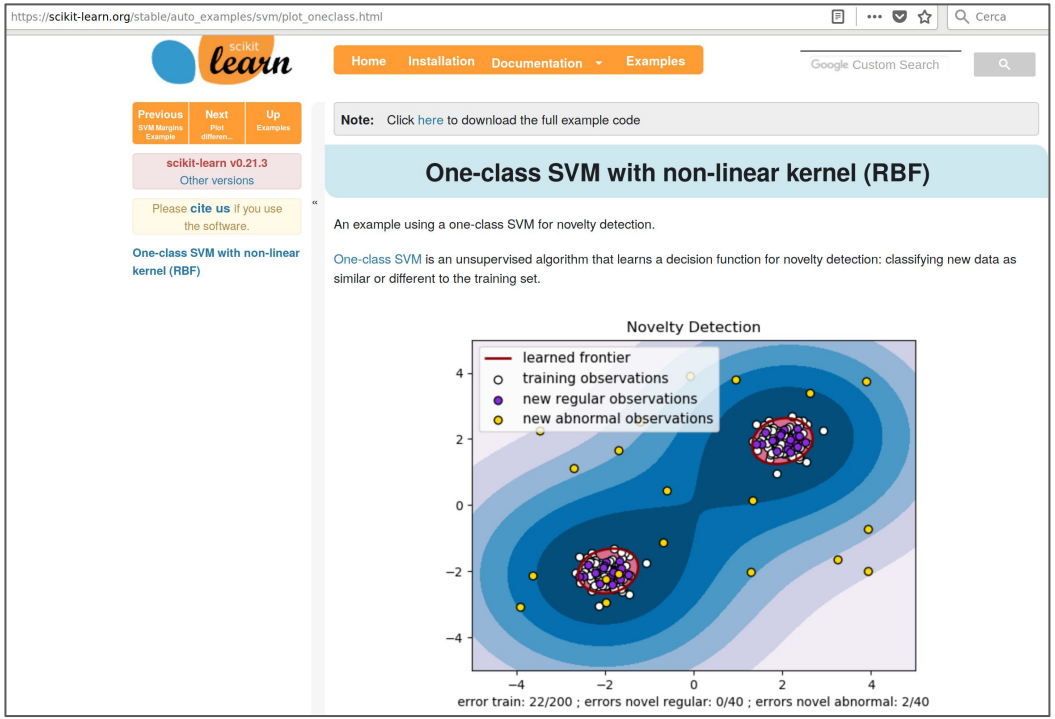
$$\langle w, \varphi(x_i)\rangle_{\mathcal{H}} \geq \rho - \xi_i, \qquad \text{for } i = 1,\ldots,n$$

Identical to SVM with bias term ρ.
Same decision function

$$w^\top \varphi(x^*) > \rho \quad \Rightarrow y^* = 1$$

$$\sum_{i=1}^{n} \alpha_i k(x_i, x^*) > \rho \quad \Rightarrow y^* = 1$$

Best thing: try it out



https://github.com/rmenoli/One-class-SVM/blob/master/One%20class%20SVM.ipynb

https://github.com/SatyaVSarma/anomaly-detection-osvm

Ranking

Ranking Task

In many cases data are not annotated with simple binary or multi-class labels.

The annotation can be more complicated and involve sample-relations, rather than individual single samples.

Example: any search engine is based on ranking
Results are lists. The output is an ordered sequence of samples

Different frameworks: pairwise and listwise algorithms

pair of documents at a time in the loss function: try and come up with the optimal ordering for that pair and compare it to the ground truth. The goal for the ranker is to minimize the number of inversions in ranking i.e. cases where the pair of results are in the wrong order relative to the ground truth.

list of documents: try to come up with the optimal ordering for all of them.

Pairwise Ranking SVM

Let us suppose to have images ordered on the basis of a descriptive attribute

Example: “smiling”

Example: “open”

Pairwise Ranking SVM

For each attribute a_m

Supervision is O_m:  , S_m: 

Learn a scoring function $r_m(\boldsymbol{x_i}) = \boldsymbol{w_m^T x_i}$ (Image features; Learned parameters)

that best satisfies constraints:

$$\forall (i,j) \in O_m : \boldsymbol{w_m^T x_i} > \boldsymbol{w_m^T x_j}$$

$$\forall (i,j) \in S_m : \boldsymbol{w_m^T x_i} = \boldsymbol{w_m^T x_j}$$

Pairwise Ranking SVM

Relax a bit the constraints, use a max-margin learning to rank formulation

O_m: $\{(\text{  } \succ \text{  }), \ldots\}$

S_m: $\{\{\text{  } \sim \text{  }\}, \ldots\}$

margin con un grado di libertà che è la slack variable

$$\boldsymbol{w_m^T}(\boldsymbol{x_i} - \boldsymbol{x_j}) \geq 1 - \xi_{ij}$$

$$\forall (i,j) \in O_m$$

$$|\boldsymbol{w_m^T}(\boldsymbol{x_i} - \boldsymbol{x_j})| \leq \gamma_{ij}$$

$$\forall (i,j) \in S_m$$

$$\xi_{ij} \geq 0; \gamma_{ij} \geq 0$$

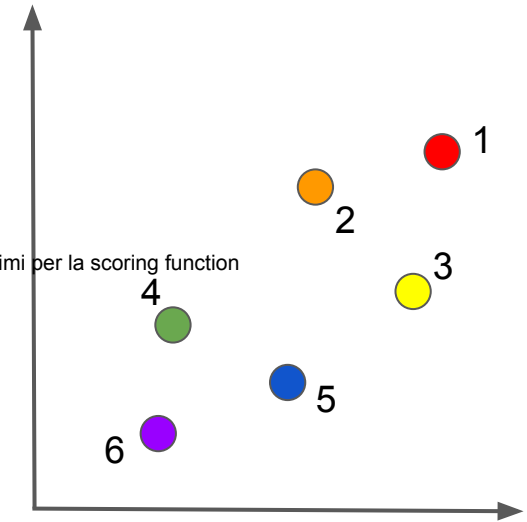
Max-margin learning to rank formulation

$$\min \quad \left(\frac{1}{2} ||\boldsymbol{w_m^T}||_2^2 + C \left(\sum \xi_{ij}^2 + \sum \gamma_{ij}^2 \right) \right)$$

Based on [Joachims, 2002]

Pairwise Ranking SVM

goal: trovare il max margin in order to find parameters (wm) ottimi per la scoring function

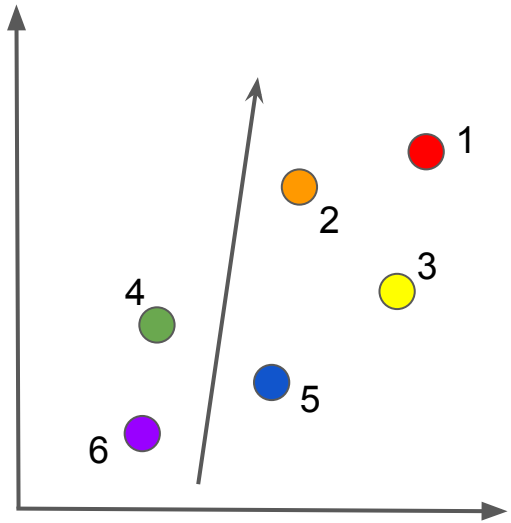


Max-margin learning to rank formulation

$$\min \quad \left(\frac{1}{2} ||\boldsymbol{w}_{\boldsymbol{m}}^T||_2^2 + C \left(\sum \xi_{ij}^2 + \sum \gamma_{ij}^2 \right) \right)$$

Based on [Joachims, 2002]

Pairwise Ranking SVM

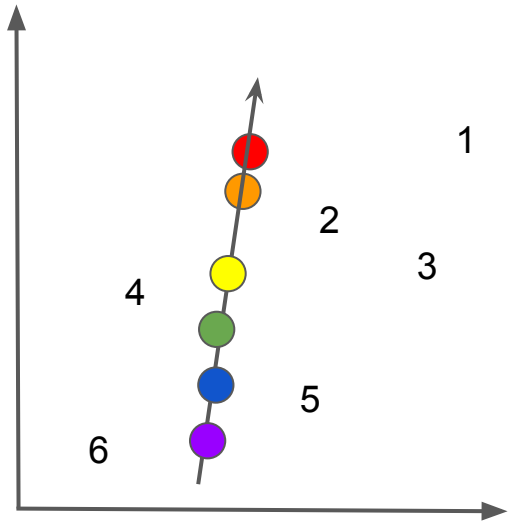


Max-margin learning to rank formulation

$$\min \quad \left(\frac{1}{2} ||\boldsymbol{w}_{\boldsymbol{m}}^T||_2^2 + C \left(\sum \xi_{ij}^2 + \sum \gamma_{ij}^2 \right) \right)$$

Based on [Joachims, 2002]

Pairwise Ranking SVM

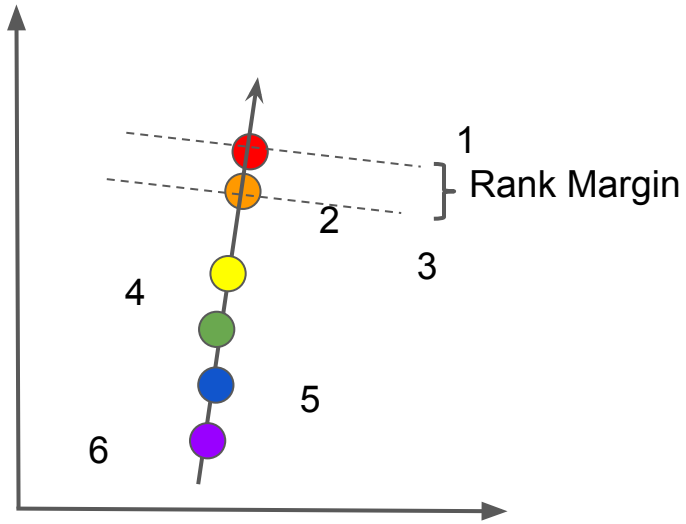


Max-margin learning to rank formulation

$$\min \quad \left(\frac{1}{2} ||\boldsymbol{w}_{\boldsymbol{m}}^T||_2^2 + C \left(\sum \xi_{ij}^2 + \sum \gamma_{ij}^2 \right) \right)$$

Based on [Joachims, 2002]

Pairwise Ranking SVM



distance between the closest ranked points is the rank margin -- what we want to maximize in this formulation

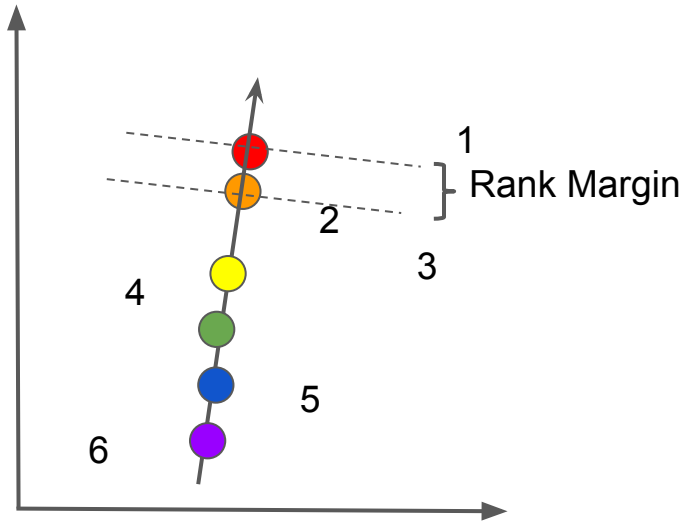
Max-margin learning to rank formulation

$$\min \quad \left(\frac{1}{2} ||\boldsymbol{w}_{\boldsymbol{m}}^T||_2^2 + C \left(\sum \xi_{ij}^2 + \sum \gamma_{ij}^2 \right) \right)$$

Based on [Joachims, 2002]

Pairwise Ranking SVM

At test time: given an image and its feature, predict its relative attribute



distance between the closest ranked points is the rank margin -- what we want to maximize in this formulation

Max-margin learning to rank formulation

$$\min \quad \left(\frac{1}{2}||\boldsymbol{w}_{\boldsymbol{m}}^T||_2^2 + C\left(\sum \xi_{ij}^2 + \sum \gamma_{ij}^2\right)\right)$$

Based on [Joachims, 2002]

Pairwise Ranking SVM

Relax a bit the constraints, use a max-margin learning to rank formulation

O_m: $\{(\text{  } \succ \text{  }), \ldots\}$

$$\boldsymbol{w}_{\boldsymbol{m}}^T(\boldsymbol{x_i} - \boldsymbol{x_j}) \geq 1 - \xi_{ij}$$

$$\forall (i,j) \in O_m$$

S_m: $\{\{\text{  } \sim \text{  }\}, \ldots\}$

$$|\boldsymbol{w}_{\boldsymbol{m}}^T(\boldsymbol{x_i} - \boldsymbol{x_j})| \leq \gamma_{ij}$$

$$\forall (i,j) \in S_m$$

$$\xi_{ij} \geq 0; \gamma_{ij} \geq 0$$

Max-margin learning to rank formulation

$$\min \left(\frac{1}{2} ||\boldsymbol{w}_{\boldsymbol{m}}^T||_2^2 + C \left(\sum \xi_{ij}^2 + \sum \gamma_{ij}^2 \right) \right)$$

Based on [Joachims, 2002]

Pairwise Ranking SVM

ICCV 2011: Marr Prize Paper
Project Page: https://www.cc.gatech.edu/~parikh/relative.html

Application Automatic Relative Image Descriptions

Density  Novel image 

Conventional binary description: *not dense*

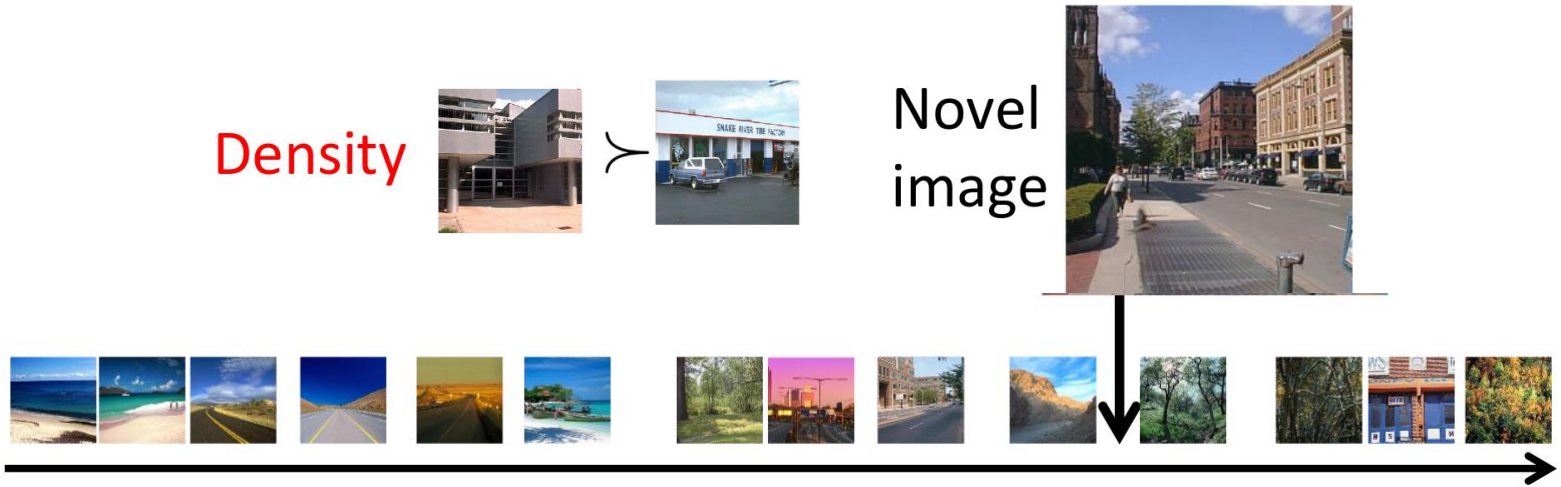
Dense: 

Not dense: 

Pairwise Ranking SVM

ICCV 2011: Marr Prize Paper
Project Page: https://www.cc.gatech.edu/~parikh/relative.html

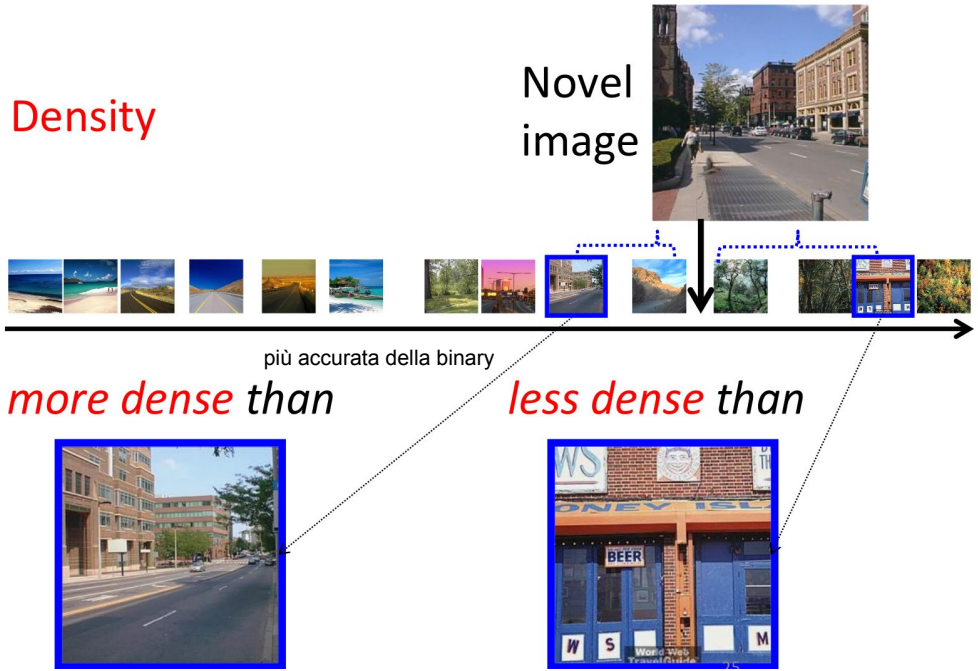
Application Automatic Relative Image Descriptions



Pairwise Ranking SVM

ICCV 2011: Marr Prize Paper
Project Page: https://www.cc.gatech.edu/~parikh/relative.html

Application Automatic Relative Image Descriptions



Pairwise Ranking SVM

ICCV 2011: Marr Prize Paper
Project Page: https://www.cc.gatech.edu/~parikh/relative.html

Application Automatic Relative Image Descriptions

Unsupervised Learning

today: clustering

...but (most probably) you already learned an unsupervised method...which one?

Unsupervised Learning

today: clustering

...but (most probably) you already learned an unsupervised method...which one?

Principal Component Analysis - we will not go over this subspace projection strategy in this course but we remind that it is a method based only on data, not on their labels. Thus it is unsupervised. If you want to know more, check Chapter 23.1 in “Understanding Machine Learning: From Theory to Algorithms” book by Shalev-Shwartz and Ben-David or Chapter 12 in “Pattern Recognition and Machine Learning” book by Bishop.

Unsupervised Learning

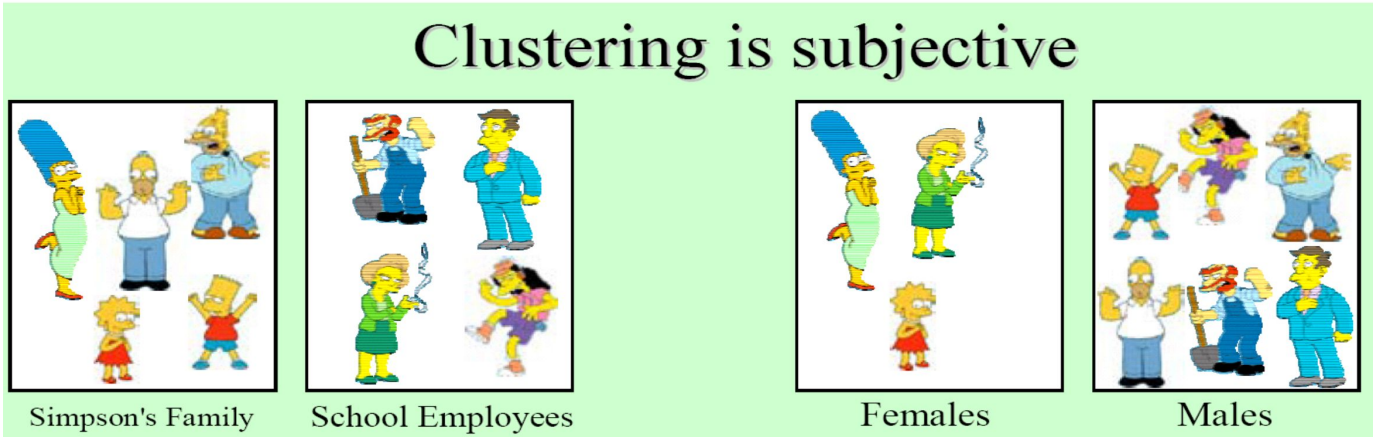
Clustering:

The process of grouping a set of objects into classes of similar objects

–high intra-class similarity

–low inter-class similarity

–It is the commonest form of unsupervised learning



Slide Credit: A.Smola, B. Póczos

But similarity is a difficult concept

Hard to define! *But we know it when we see it*

The real meaning of similarity is a philosophical question. We will take a more pragmatic approach: think in terms of a **distance** (rather than similarity) between random variables.

k-means algorithm

Assumptions

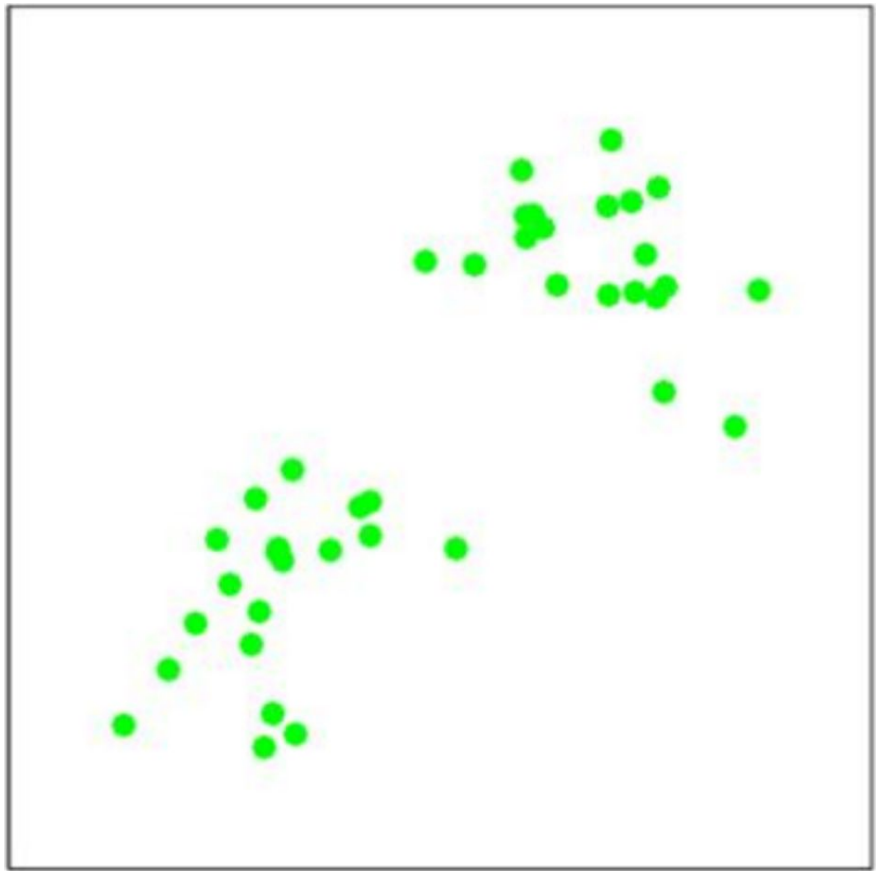
- Assume Euclidean space / distance
- Start by picking k, the number of clusters (groups of data)
- Initialize the clusters by picking one point per cluster
- For the moment we assume to pick k points at random

A very simple procedure

1. for each point of our dataset, place it in the cluster whose current centroid is the nearest
2. After all points are assigned, update the location of centroids of the k clusters
3. Reassign all points to the closest centroid (move points between clusters)

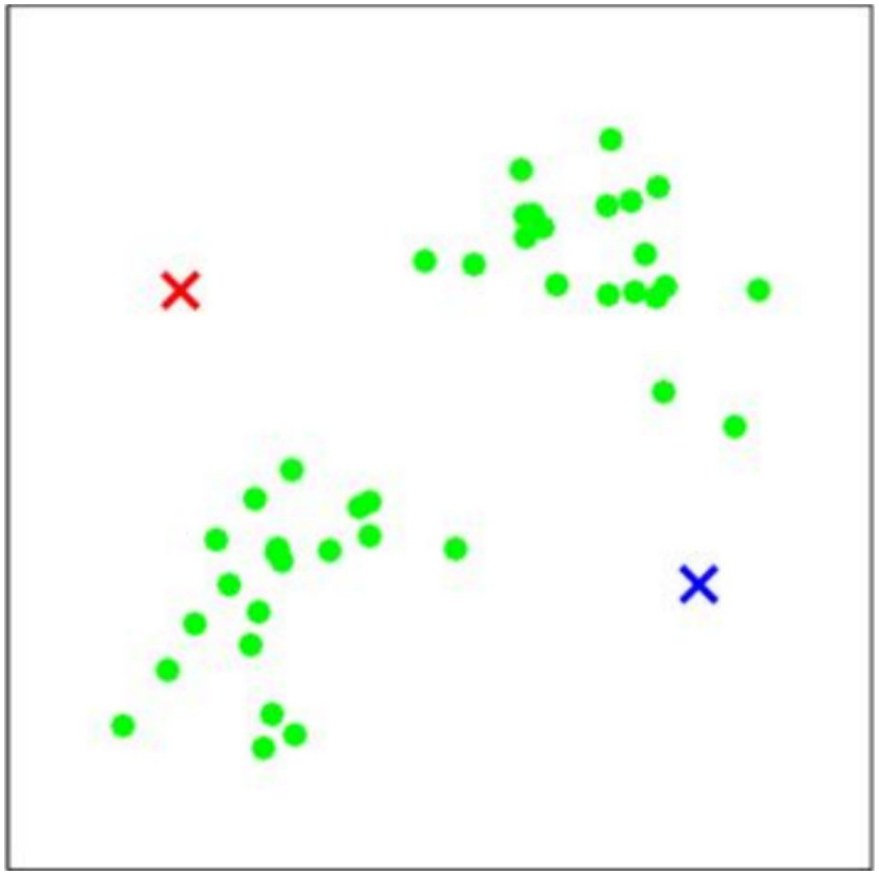
Repeat step 2 and 3 until convergence: all centroids stabilize, the points do not move between clusters anymore

Example k = 2 (starting point)



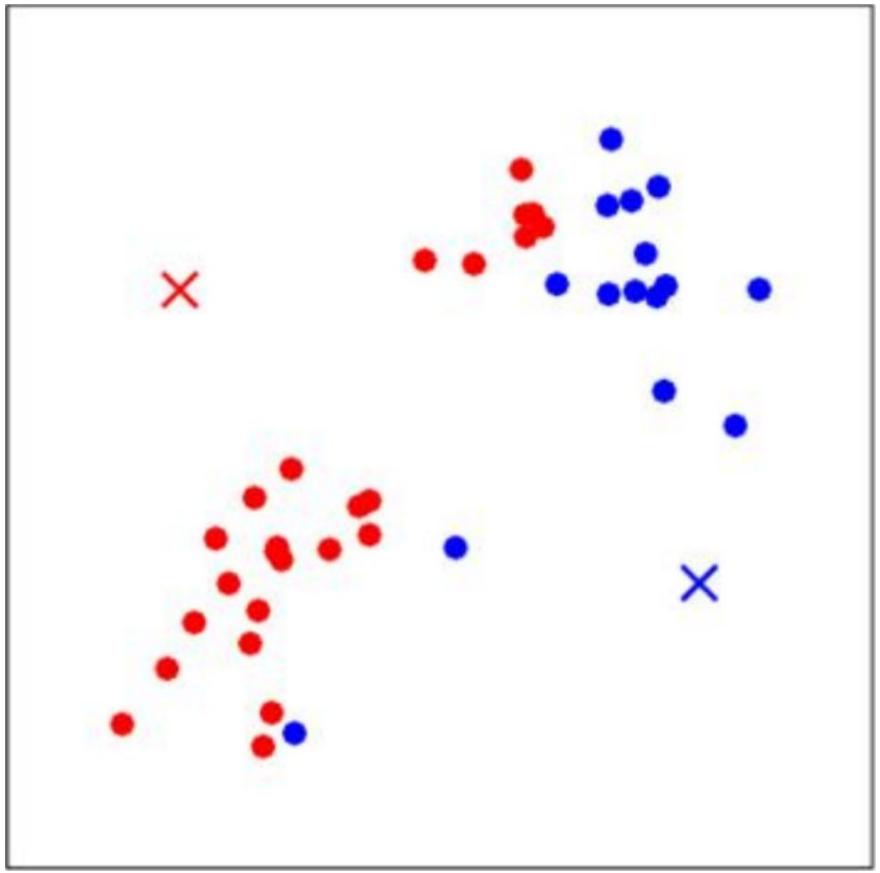
Slide Credit: A.Ng

Example k = 2 (random choice of centers)



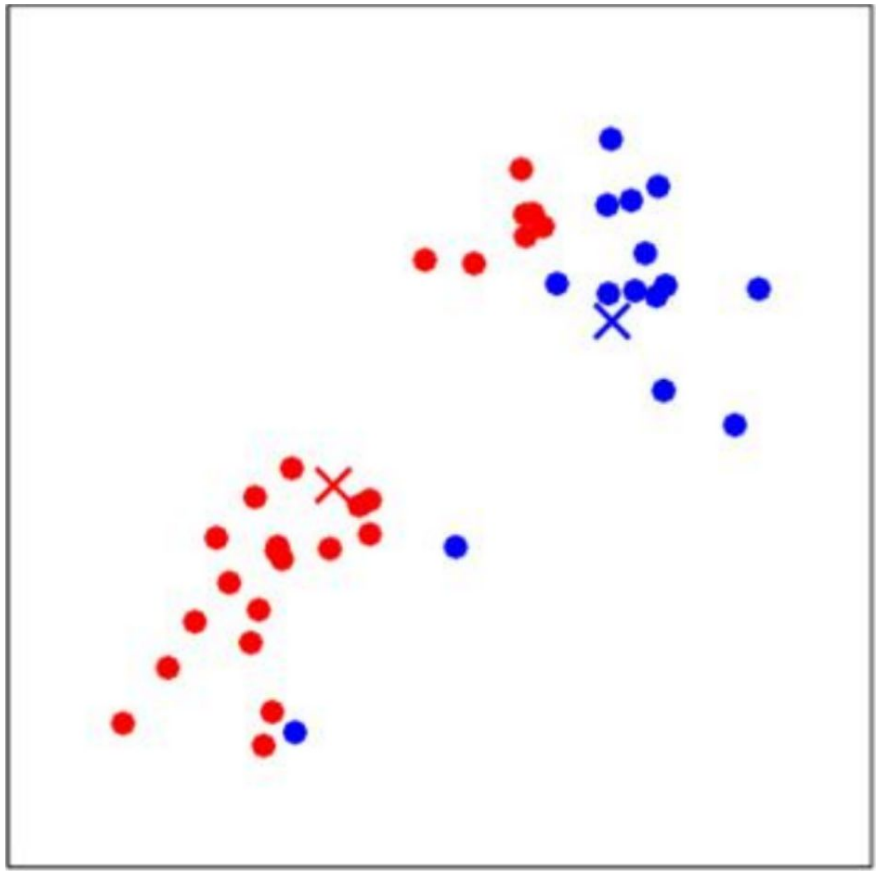
Slide Credit: A.Ng

Example k = 2 (assign points to centers)



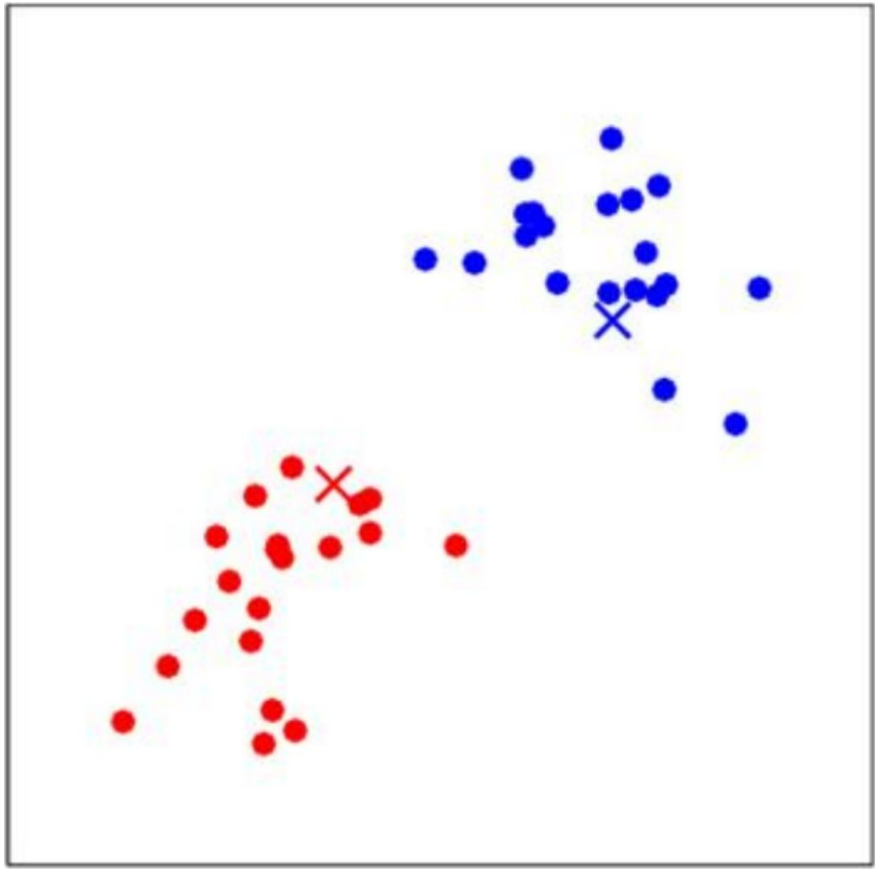
Slide Credit: A.Ng

Example k = 2 (update the centers)



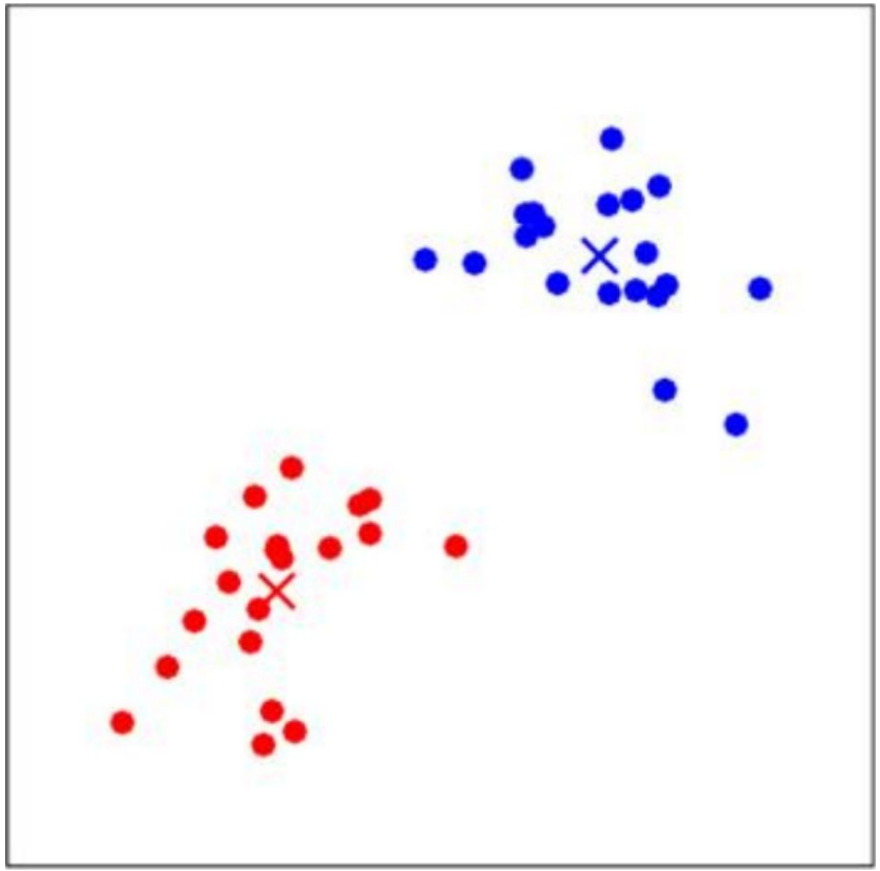
Slide Credit: A.Ng

Example k = 2 (update of point assignments)



Slide Credit: A.Ng

Example k = 2 (update the centers again…)

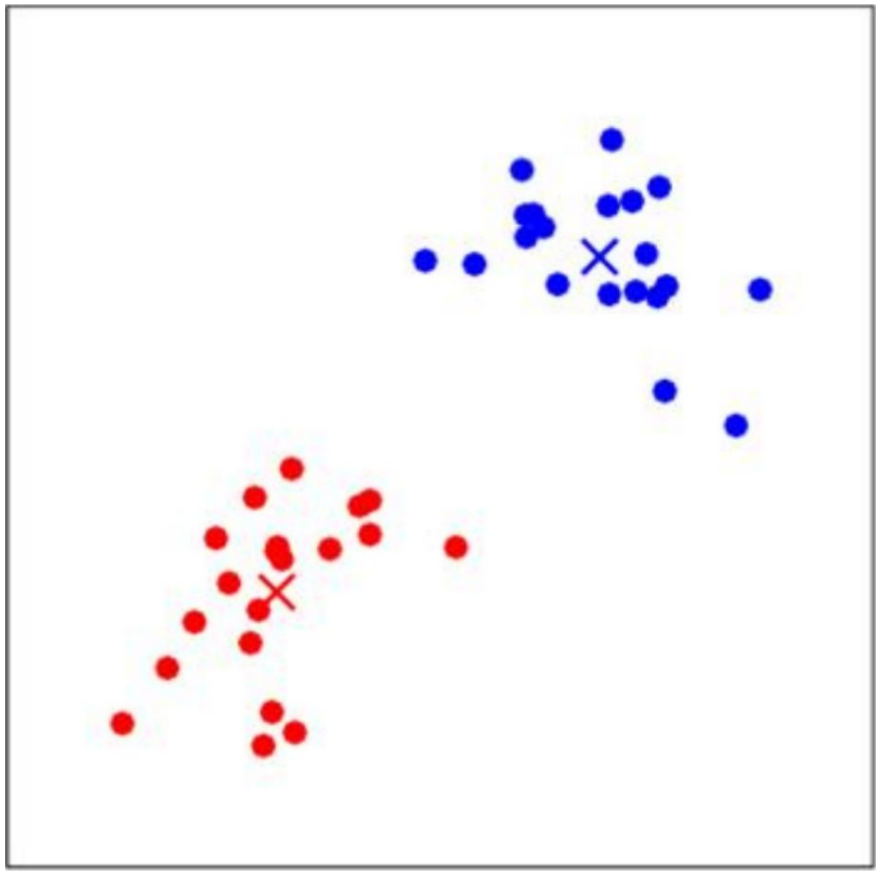


Slide Credit: A.Ng

Example k = 2 (update the centers again…)

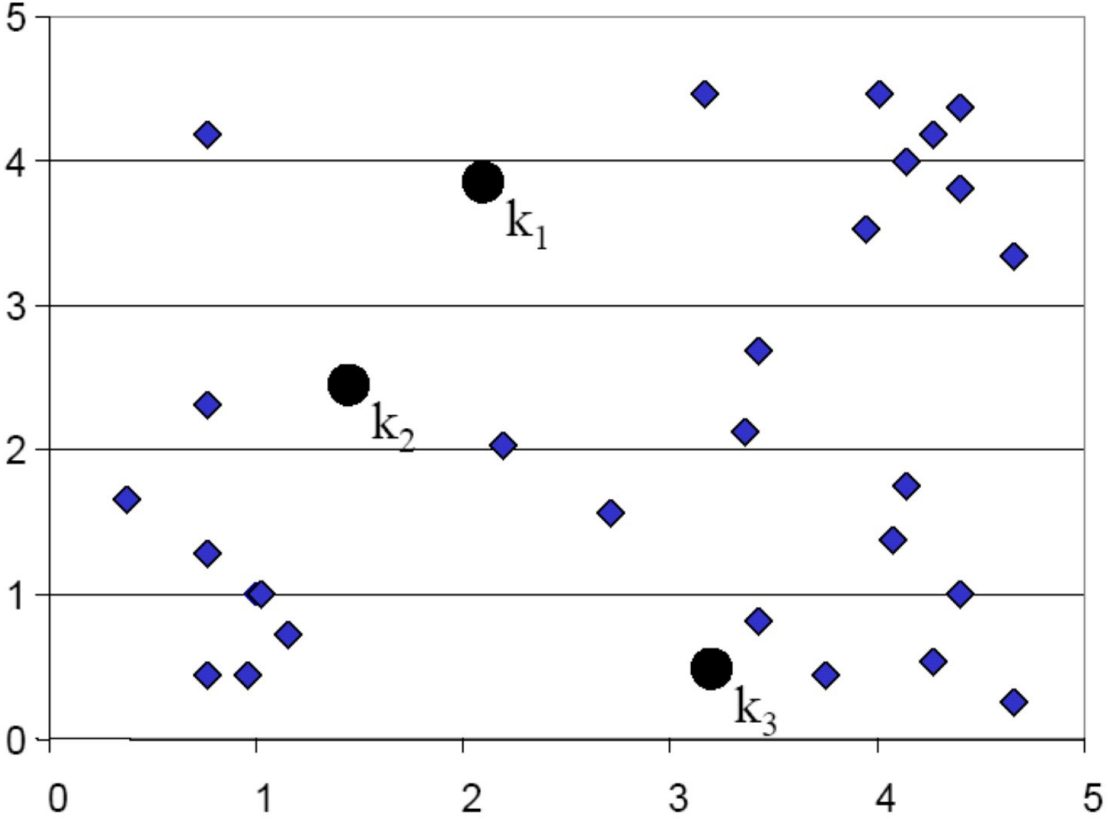
Quite reasonable clustering.

We are done!



Slide Credit: A.Ng

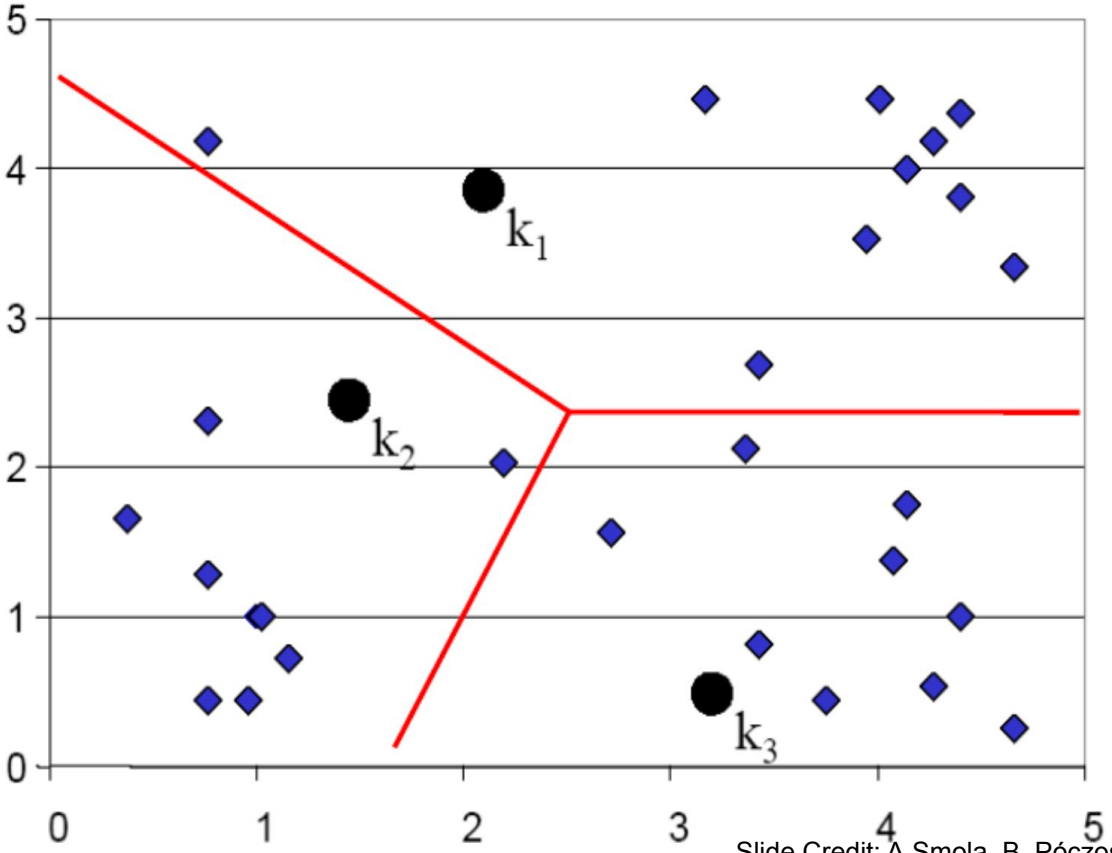
Example k = 3



Slide Credit: A.Smola, B. Póczos

Example k = 3

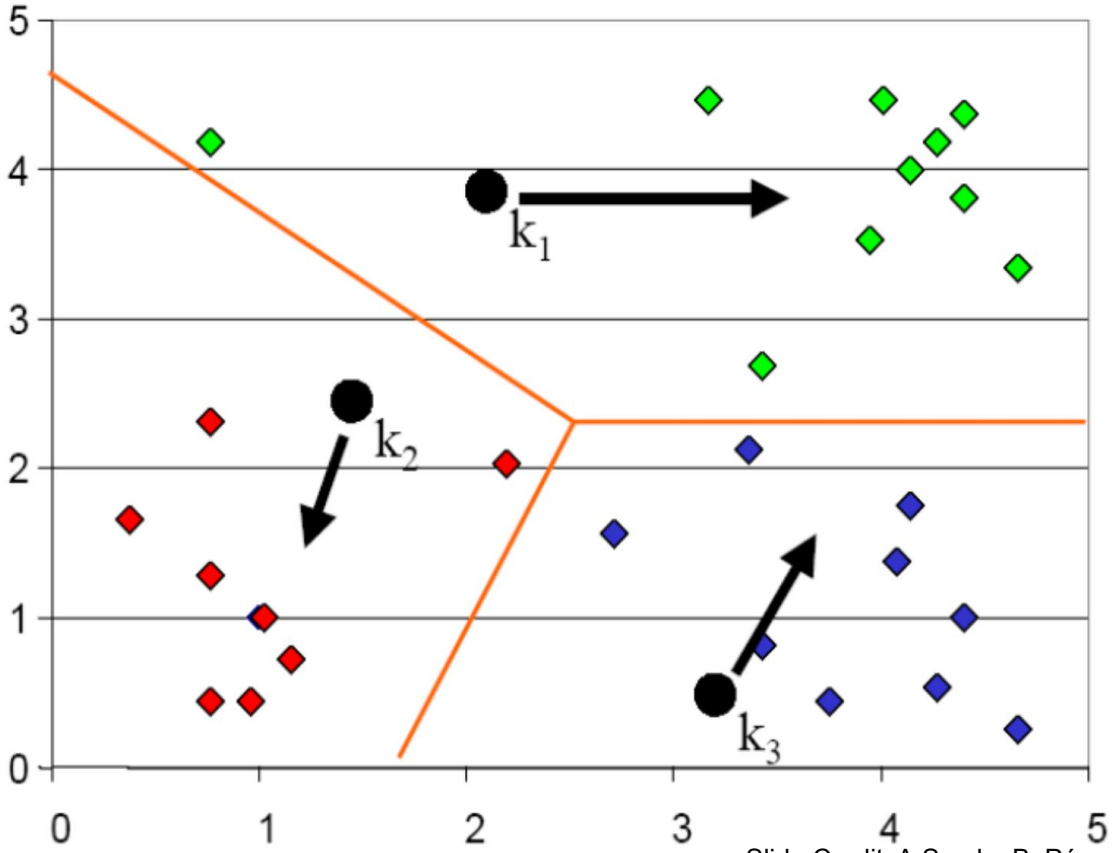
red lines: voronoi lines



Slide Credit: A.Smola, B. Póczos

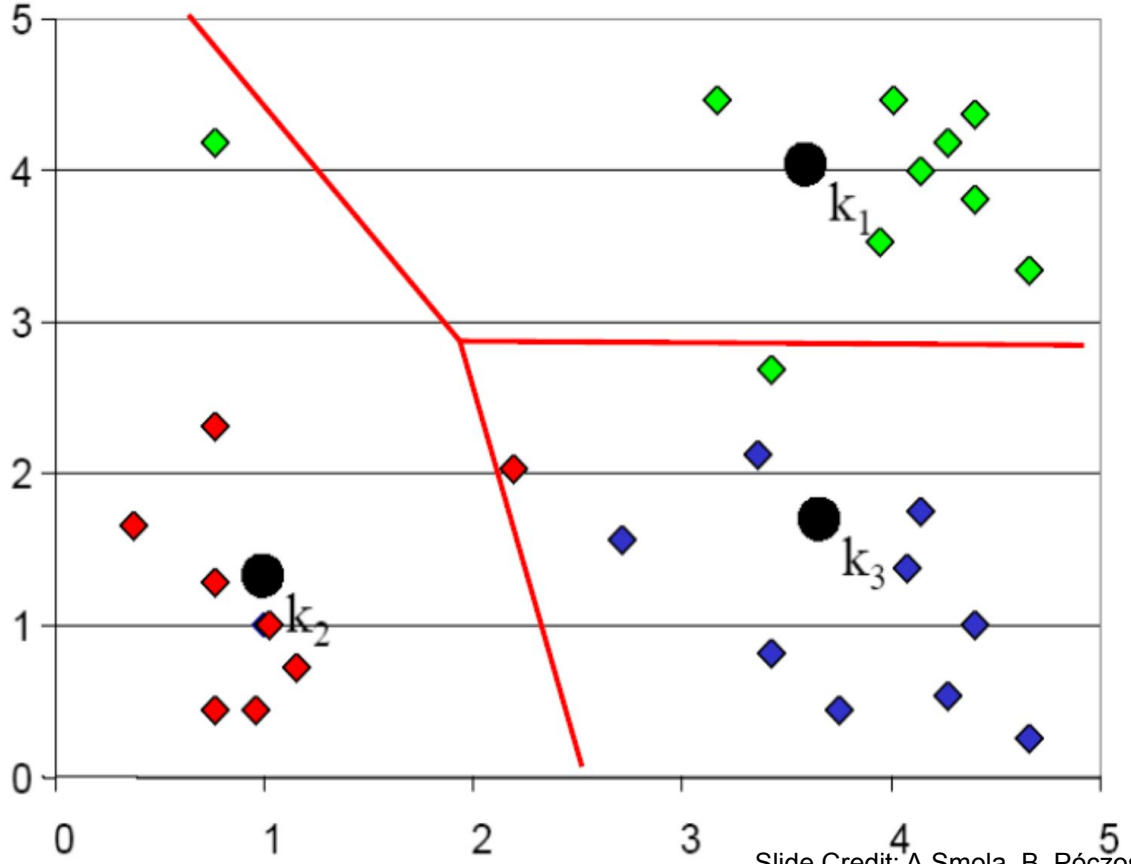
Example k = 3

ogni data point è associato al nearest centroid



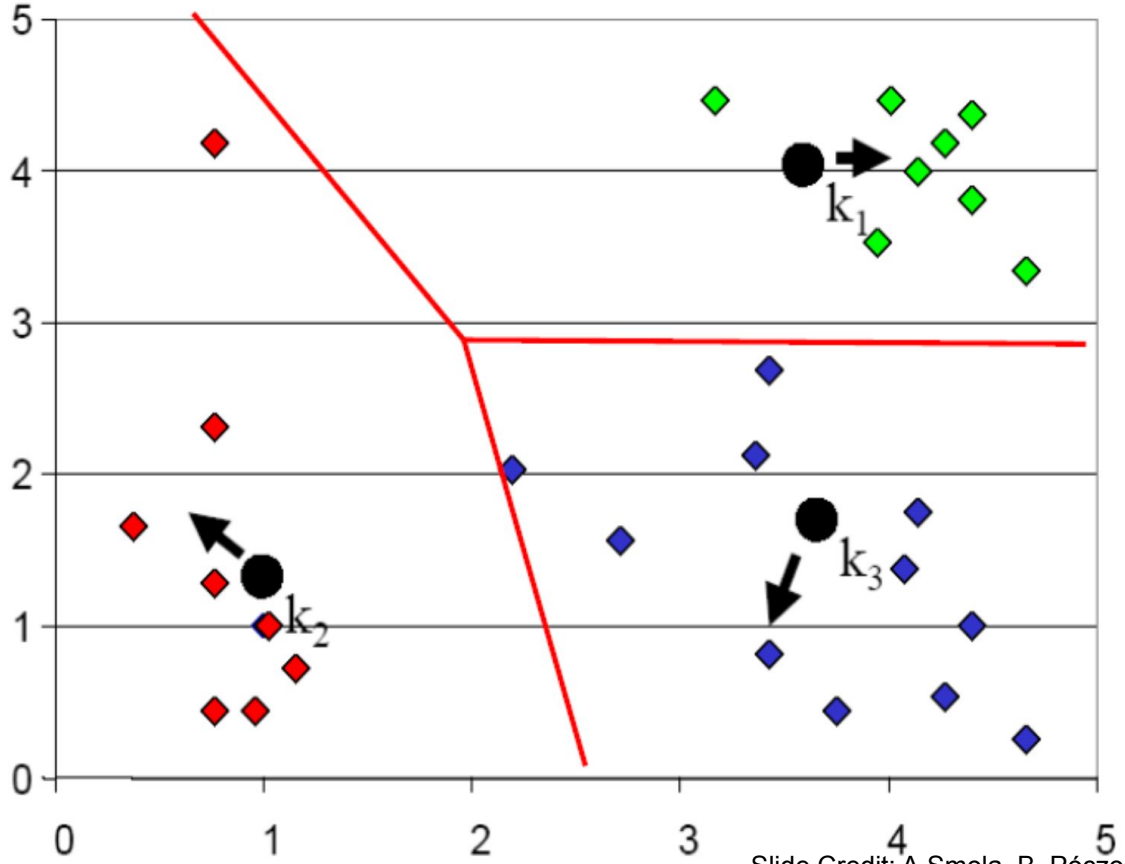
Slide Credit: A.Smola, B. Póczos

Example k = 3



Slide Credit: A.Smola, B. Póczos

Example k = 3

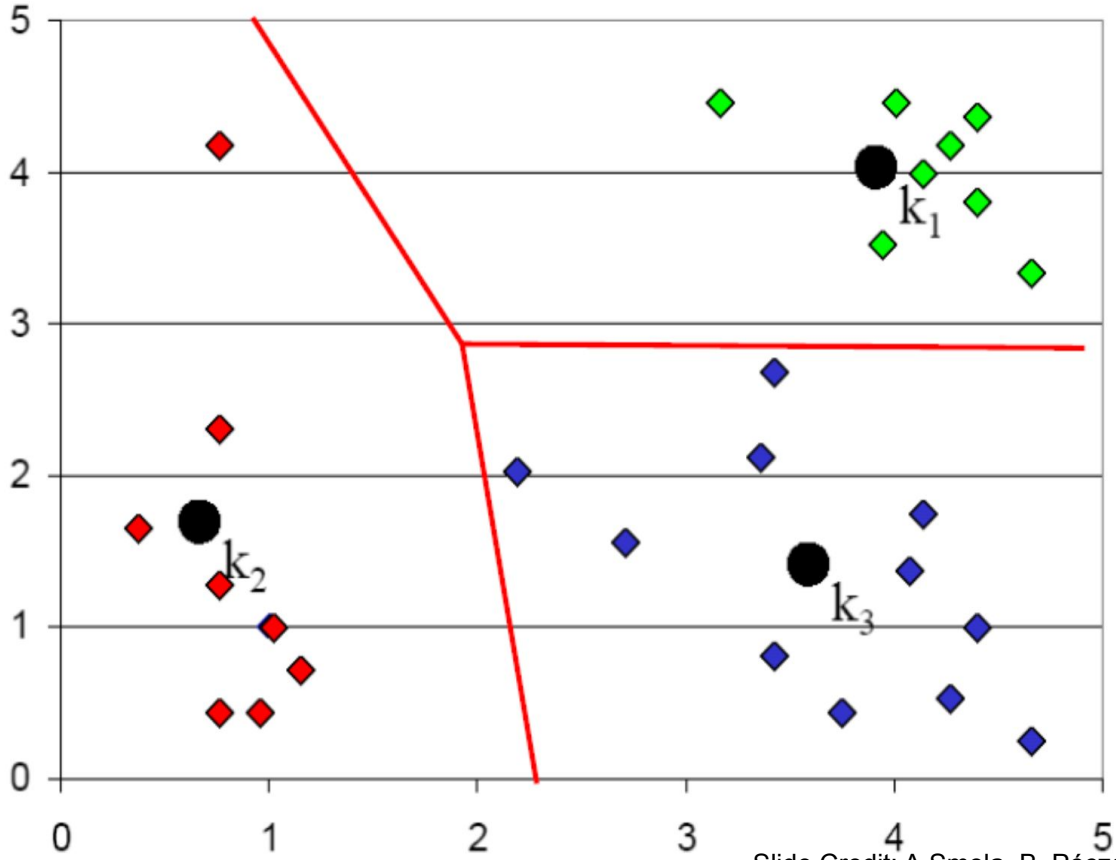


Slide Credit: A.Smola, B. Póczos

Example k = 3

Quite reasonable clustering.

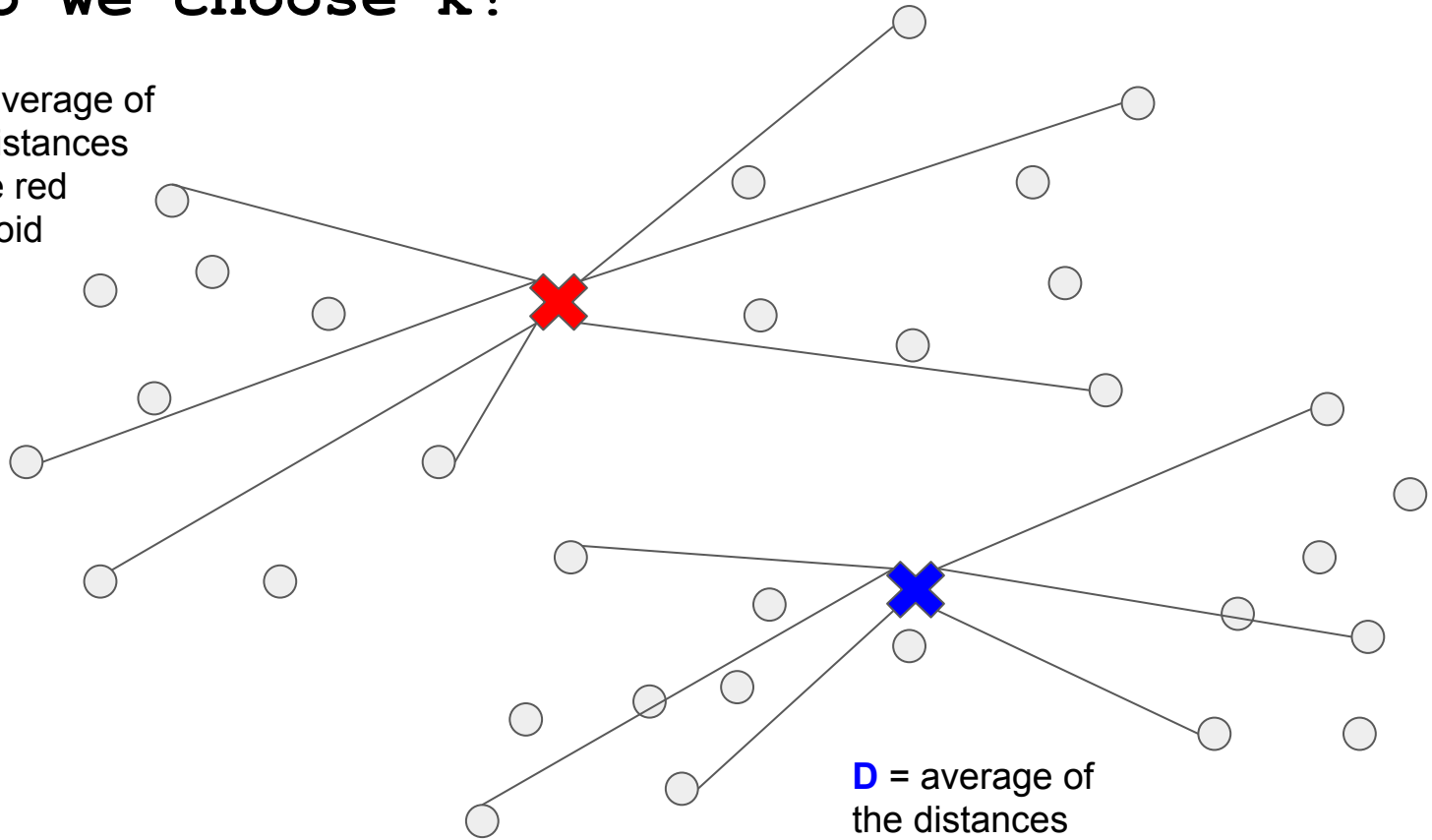
We are done!



Slide Credit: A.Smola, B. Póczos

How do we choose k?

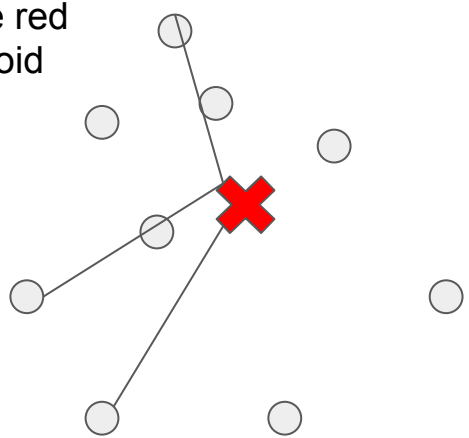
D = average of the distances to the red centroid



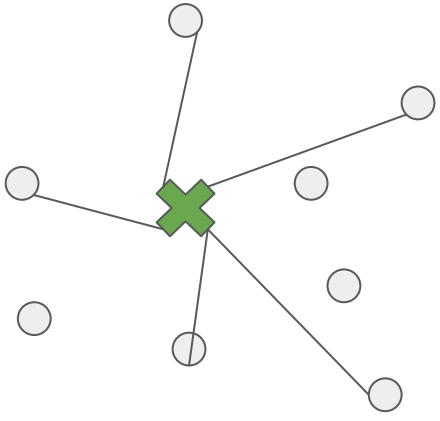
D = average of the distances to the blue centroid

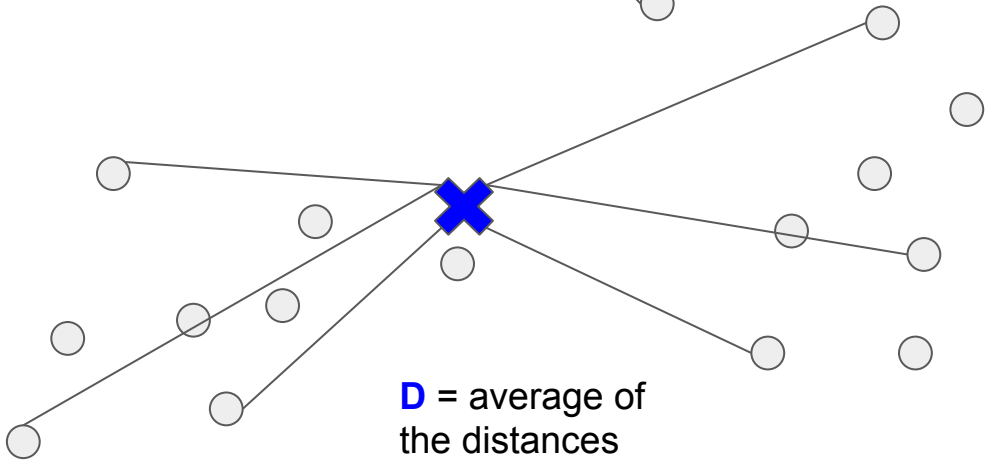
How do we choose k?

D = average of the distances to the red centroid



D = average of the distances to the green centroid

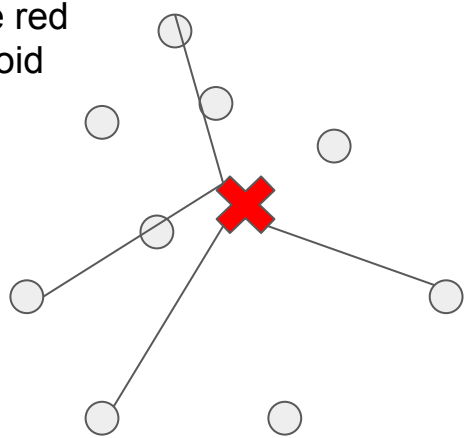




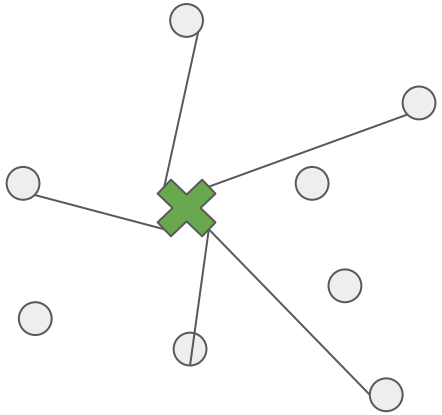
D = average of the distances to the blue centroid

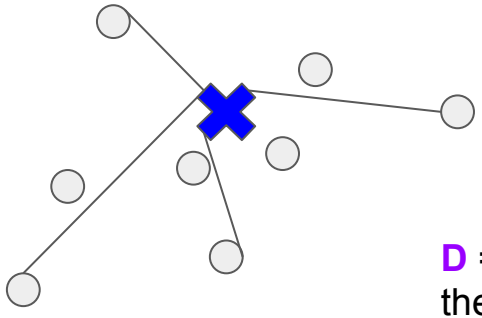
How do we choose k?

D = average of the distances to the red centroid

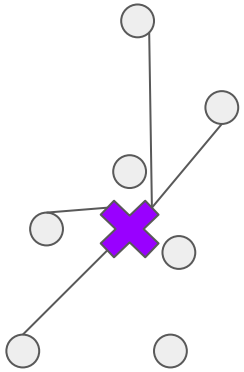


D = average of the distances to the green centroid





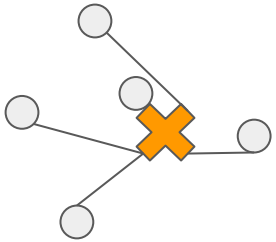
D = average of the distances to the blue centroid



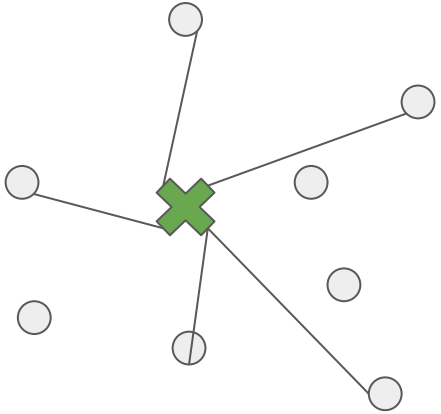
D = average of the distances to the purple centroid

How do we choose k?

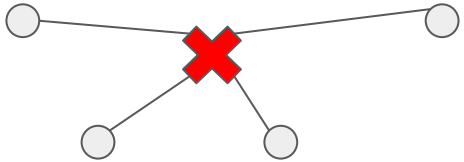
D = average of the distances to the orange centroid



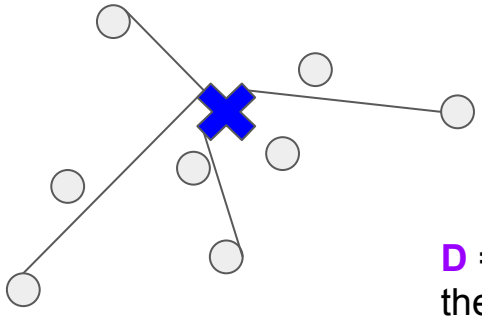
D = average of the distances to the green centroid



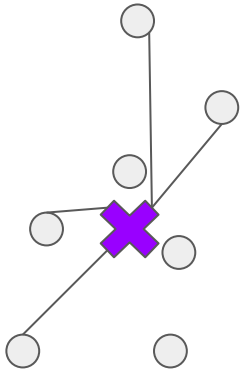
D = average of the distances to the red centroid



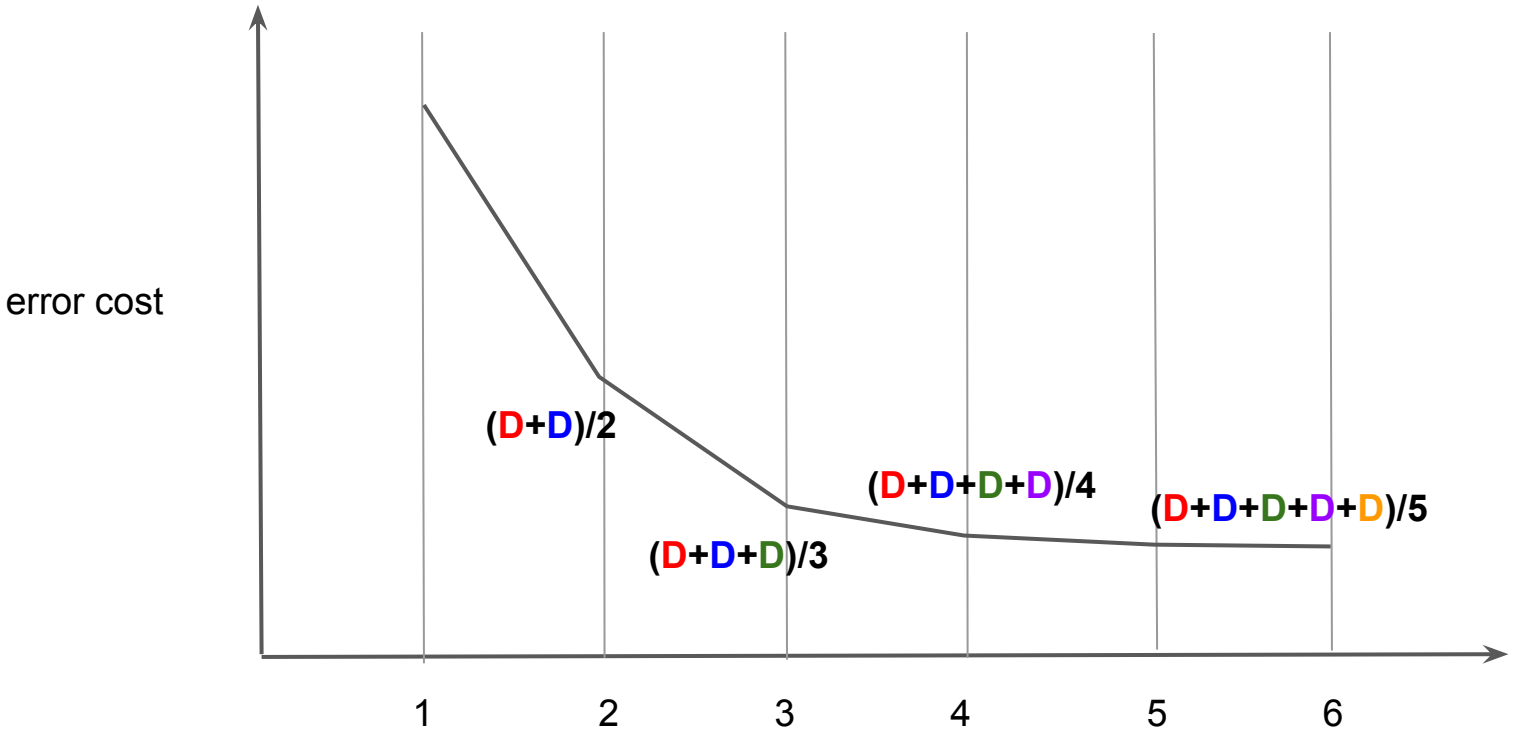
D = average of the distances to the blue centroid



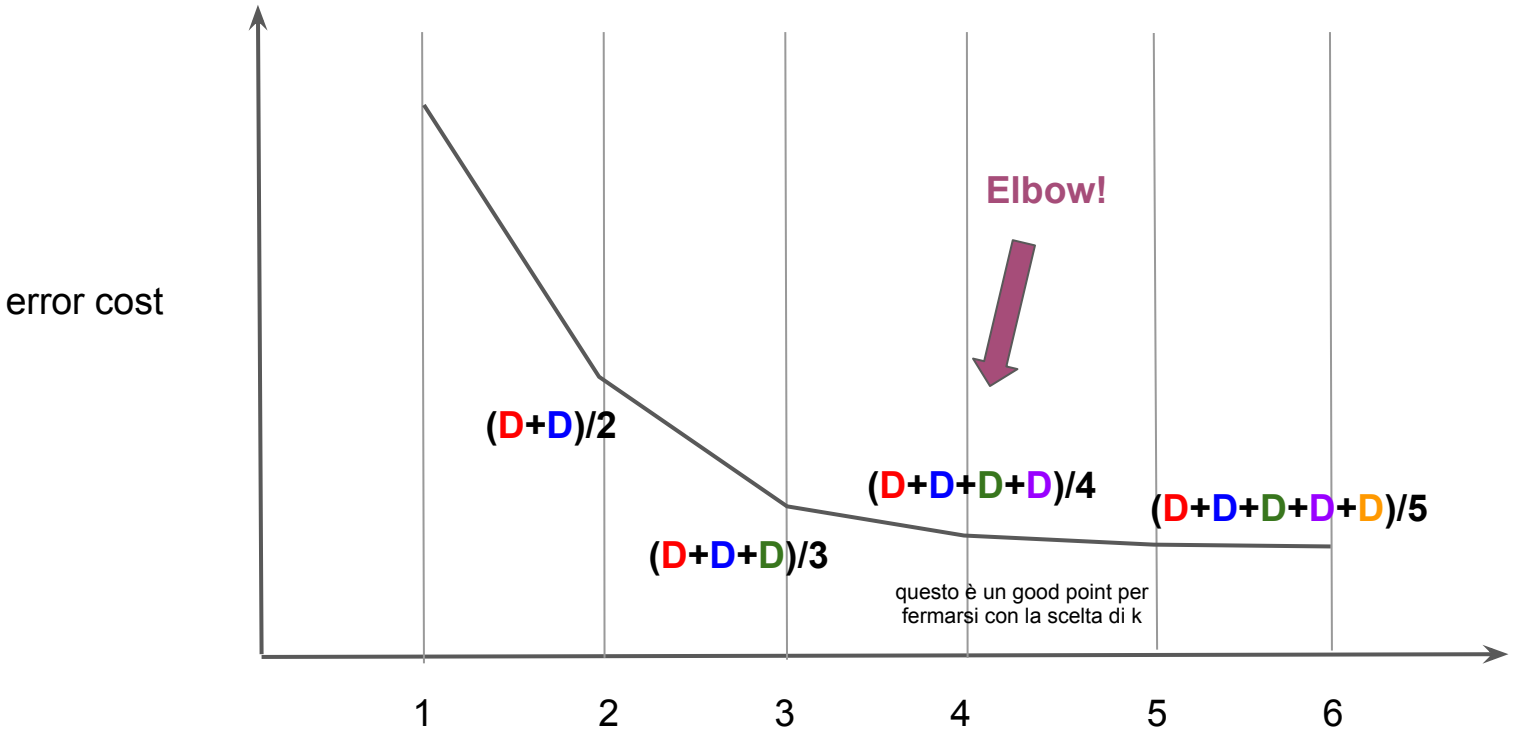
D = average of the distances to the purple centroid



How do we choose k?



How do we choose k?



How do we select the k points?

per l'inizializzazione

Few simple choices

1. at random

2. at random but multiple times: re-initialize the kmeans multiple times.
   At the end compare the distance curves of the previous slide and choose the one with the lowest elbow.

3. pick "dispersed set of points"

- pick the first centroid at random
- pick the second centroid to be the one as far as possible from the first one in the data
- pick the third centroid as the one as far as possible from the first two…
- iteratively: choose each centroid whose minimum distance from the previous ones is as large as possible

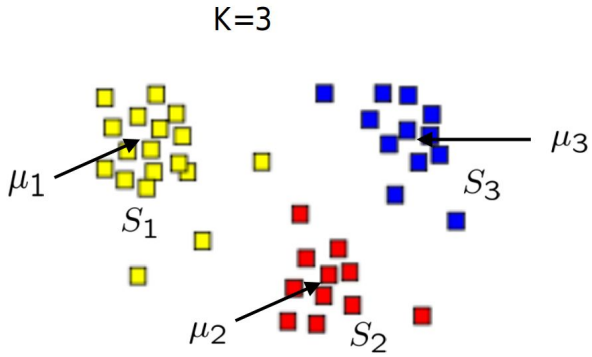
What are we really optimizing?

Given a set of observations $(x_1, x_2, \ldots, x_n)$

where $x_i \in \mathbb{R}^d$

- **Randomly initialize k centers**

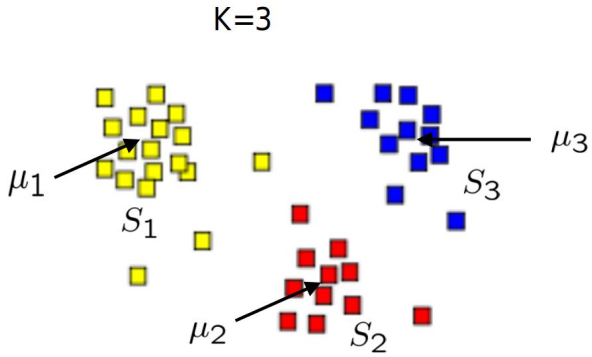
$$\mu^0 = (\mu_1^0, \ldots, \mu_K^0)$$



What are we really optimizing?

Given a set of observations $(x_1, x_2, \ldots, x_n)$

where $x_i \in \mathbb{R}^d$



- **Randomly initialize k centers**

$$\mu^0 = (\mu_1^0, \ldots, \mu_K^0)$$

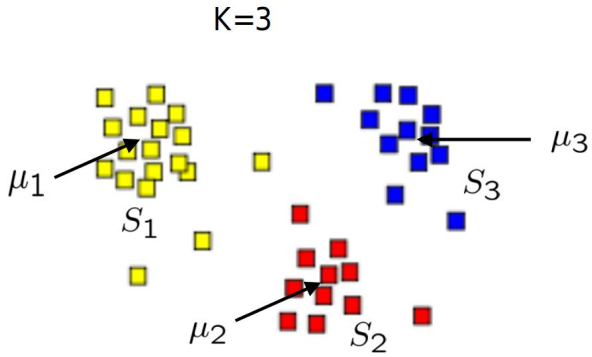
- **Classify**: At iteration t, assign each point j ∈ {1,...,n} to nearest center:

$$C^t(j) \leftarrow \arg\min_i \|\mu_i^t - x_j\|^2$$ Classification at iteration *t*

What are we really optimizing?

Given a set of observations $(x_1, x_2, \ldots, x_n)$

where $x_i \in \mathbb{R}^d$



- **Randomly initialize k centers**

$$\mu^0 = (\mu_1^0, \ldots, \mu_K^0)$$

- **Classify**: At iteration t, assign each point j ∈ {1,...,n} to nearest center:

$$C^t(j) \leftarrow \arg\min_i \|\mu_i^t - x_j\|^2$$ Classification at iteration *t*

- **Recenter**: μ_i is the centroid of the new sets:

$$\mu_i^{(t+1)} \leftarrow \arg\min_\mu \sum_{j:C^t(j)=i} \|\mu - x_j\|^2$$ Re-assign new cluster centers at iteration *t*

What are we really optimizing?

Define the following potential function F of centers μ and point allocation C

$$\mu = (\mu_1, \ldots, \mu_K)$$

$$C = (C(1), \ldots, C(n))$$

$$F(\mu, C) = \sum_{j=1}^{n} \|\mu_{C(j)} - x_j\|^2$$

squared distance fra un esempio cj e il suo corrispondente centro mu di cj

What are we really optimizing?

Define the following potential function F of centers μ and point allocation C

$$\mu = (\mu_1, \ldots, \mu_K)$$

$$C = (C(1), \ldots, C(n))$$

$$F(\mu, C) = \sum_{j=1}^{n} \|\mu_{C(j)} - x_j\|^2$$

$$= \sum_{i=1}^{K} \sum_{j:C(j)=i} \|\mu_i - x_j\|^2$$

Two equivalent versions

What are we really optimizing?

Define the following potential function F of centers μ and point allocation C

$$\mu = (\mu_1, \ldots, \mu_K)$$

$$C = (C(1), \ldots, C(n))$$

$$F(\mu, C) = \sum_{j=1}^{n} \|\mu_{C(j)} - x_j\|^2$$

$$= \sum_{i=1}^{K} \sum_{j:C(j)=i} \|\mu_i - x_j\|^2$$

Two equivalent versions

Optimal solution of the K-means problem:

$$\min_{\mu, C} F(\mu, C)$$

What are we really optimizing?

Optimize the potential function:

$$\min_{\mu,C} F(\mu, C) = \min_{\mu,C} \sum_{j=1}^{n} \|\mu_{C(j)} - x_j\|^2 = \min_{\mu,C} \sum_{i=1}^{K} \sum_{j:C(j)=i} \|\mu_i - x_j\|^2$$

What are we really optimizing?

Optimize the potential function:

$$\min_{\mu,C} F(\mu, C) = \min_{\mu,C} \sum_{j=1}^{n} \|\mu_{C(j)} - x_j\|^2 = \min_{\mu,C} \sum_{i=1}^{K} \sum_{j:C(j)=i} \|\mu_i - x_j\|^2$$

K-means algorithm:

(1) Fix μ, Optimize C

$$\min_{C(1),C(2),\ldots,C(n)} \sum_{j=1}^{n} \|\mu_{C(j)} - x_j\|^2 = \sum_{j=1}^{n} \min_{C(j)} \|\mu_{C(j)} - x_j\|^2$$

Exactly first step

Assign each point to the nearest cluster center

What are we really optimizing?

Optimize the potential function:

$$\min_{\mu,C} F(\mu, C) = \min_{\mu,C} \sum_{j=1}^{n} \|\mu_{C(j)} - x_j\|^2 = \min_{\mu,C} \sum_{i=1}^{K} \sum_{j:C(j)=i} \|\mu_i - x_j\|^2$$

K-means algorithm:

(1) Fix μ, Optimize C

$$\min_{C(1),C(2),\ldots,C(n)} \sum_{j=1}^{n} \|\mu_{C(j)} - x_j\|^2 = \sum_{j=1}^{n} \min_{C(j)} \|\mu_{C(j)} - x_j\|^2$$

Exactly first step

Assign each point to the nearest cluster center

(2) Fix C, Optimize μ

$$\min_{\mu_1,\ldots,\mu_K} \sum_{i=1}^{K} \sum_{j:C(j)=i} \|\mu_i - x_j\|^2 = \sum_{i=1}^{K} \min_{\mu_i} \sum_{j:C(j)=i} \|\mu_i - x_j\|^2$$

Exactly 2nd step (re-center)

What are we really optimizing?

Optimize the potential function:

$$\min_{\mu,C} F(\mu, C) = \min_{\mu,C} \sum_{j=1}^{n} \|\mu_{C(j)} - x_j\|^2$$

K-means algorithm: (coordinate descent on F)

(1) Fix μ, Optimize C **Expectation step**

(2) Fix C, Optimize μ **Maximization step**

Complexity

- At each iteration,
  - Computing distance between each of the n objects and the K cluster centers is $O(Kn)$.
  - Computing cluster centers: Each object gets added once to some cluster: $O(n)$.
- Assume these two steps are each done once for ℓ iterations: $O(\ell Kn)$.

Convergence

Can you prove that the K-means algorithm guaranteed to terminate?

The potential function that we are minimizing is not strictly convex.
It decreases but does not have a single minimum.

Terminates when we have very small oscillations around a local minimum which most of the times is a reasonable clustering solution.

- Some seeds can result in **poor convergence rate**, or convergence to **sub-optimal** clustering.
- K-means algorithm can get stuck easily in **local minima.**
  - Select good seeds using a heuristic (e.g., object least similar to any existing mean)
  - Try out **multiple** starting points (very important!!!)

Clustering Visual Application: Segmentation

K=2

Goal of Segmentation is to partition an image into regions each of which has reasonably homogenous visual appearance.

Original

Slide Credit: David Sontag

Clustering Visual Application: Segmentation

Slide Credit: David Sontag

Best thing: try it out

https://scikit-learn.org/stable/auto_examples/cluster/plot_color_quantization.html#sphx-glr-auto-examples-cluster-plot-color-quantization-py

Cerca

Previous (Demo of DBSCA...) | Next (Hierarchical ...) | Up (Examples)

scikit-learn v0.21.3
Other versions

Please **cite us** if you use the software.

Color Quantization using K-Means

Note: Click here to download the full example code

Color Quantization using K-Means

Performs a pixel-wise Vector Quantization (VQ) of an image of the summer palace (China), reducing the number of colors required to show the image from 96,615 unique colors to 64, while preserving the overall appearance quality.

In this example, pixels are represented in a 3D-space and K-means is used to find 64 color clusters. In the image processing literature, the codebook obtained from K-means (the cluster centers) is called the color palette. Using a single byte, up to 256 colors can be addressed, whereas an RGB encoding requires 3 bytes per pixel. The GIF file format, for example, uses such a palette.

For comparison, a quantized image using a random codebook (colors picked up randomly) is also shown.

Original image (96,615 colors)

Quantized image (64 colors, K-Means)

Best thing: try it out

https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_digits.html#sphx-glr-auto-examples-cluster-plot-kmeans-digits-py

Cerca

Previous (Comparison of...) | Next (Comparing dif...) | Up (Examples)

scikit-learn v0.21.3
Other versions

Please **cite us** if you use the software.

A demo of K-Means clustering on the handwritten digits data

Note: Click here to download the full example code

A demo of K-Means clustering on the handwritten digits data

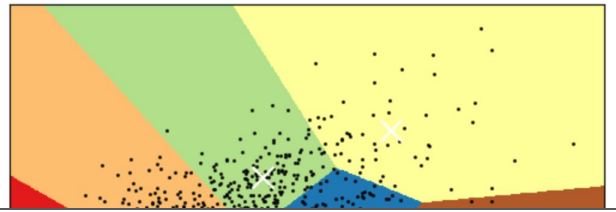
In this example we compare the various initialization strategies for K-means in terms of runtime and quality of the results.

As the ground truth is known here, we also apply different cluster quality metrics to judge the goodness of fit of the cluster labels to the ground truth.

Cluster quality metrics evaluated (see Clustering performance evaluation for definitions and discussions of the metrics):

Shorthand	full name
homo	homogeneity score
compl	completeness score
v-meas	V measure
ARI	adjusted Rand index
AMI	adjusted mutual information
silhouette	silhouette coefficient

K-means clustering on the digits dataset (PCA-reduced data)
Centroids are marked with white cross



Best thing: try it out

https://scikit-learn.org/stable/auto_examples/cluster/plot_face_compress.html#sphx-glr-auto-examples-cluster-plot-face-compress-py

Cerca

scikit learn

Home Installation Documentation Examples

Previous Online learni... | Next Agglomerativ e... | Up Examples

scikit-learn v0.21.3
Other versions

Please cite us if you use the software.

Vector Quantization Example

Note: Click here to download the full example code

Vector Quantization Example

Face, a 1024 x 768 size image of a raccoon face, is used here to illustrate how `k`-means is used for vector quantization.