An Introduction to Statistical Machine Learning - Classical Models -

Samy Bengio

`bengio@idiap.ch`

Dalle Molle Institute for Perceptual Artificial Intelligence (IDIAP)

CP 592, rue du Simplon 4

1920 Martigny, Switzerland

`http://www.idiap.ch/~bengio`

Classical Models

1. Parametric or Not?
2. Histograms
3. Problem: Curse of Dimensionality
4. K Nearest Neighbors
5. Parzen Windows
6. Maximum Likelihood Approach
7. Bayes Decision and Bayes Classifiers
8. K-Means

Parametric or Not?

- The space $\mathcal{F}$ is often characterized to be parametric or not.
- Parametric: the space is very small, and characterized by a small number of parameters.
  - examples: a Gaussian distribution or a linear function
  - big prior on the solution
- Non-Parametric: the space is infinite, constrained only by the training data
  - examples: K nearest neighbors, Parzen Windows
  - small prior on the solution
- Semi-Parametric:
  - examples: most machine learning algorithms!
  - small prior on the solution, characterized by a large number of parameters

Histograms

- For classification or regression: $z = (x, y)$
- Let x be a $k-$dimensional vector
- For each dimension d, divide the possible values x_d into m_d bins
- Total number of bins $= \prod_{d=1}^{k} m_d$
- Model: compute average value (on the training set) of $\hat{y}$ corresponding to each bin
- Test: given a new example x, select the corresponding bin and output the associated $\hat{y}$
- Can be extended to classification and density estimation.
- Capacity controlled by the total number of bins.

Histograms (Graphical View)

- $x = \{x_1, x_2\}$
- estimated value of $\hat{y}$:

	$x_1 < 5$	$5 \leq x_1 < 7$	$7 \leq x_1$
$x_2 = \text{red}$	$\hat{y} = -3.2$	$\hat{y} = 1.5$	$\hat{y} = 3.2$
$x_2 = \text{blue}$	$\hat{y} = -3.2$	$\hat{y} = 0.1$	$\hat{y} = 0.37$

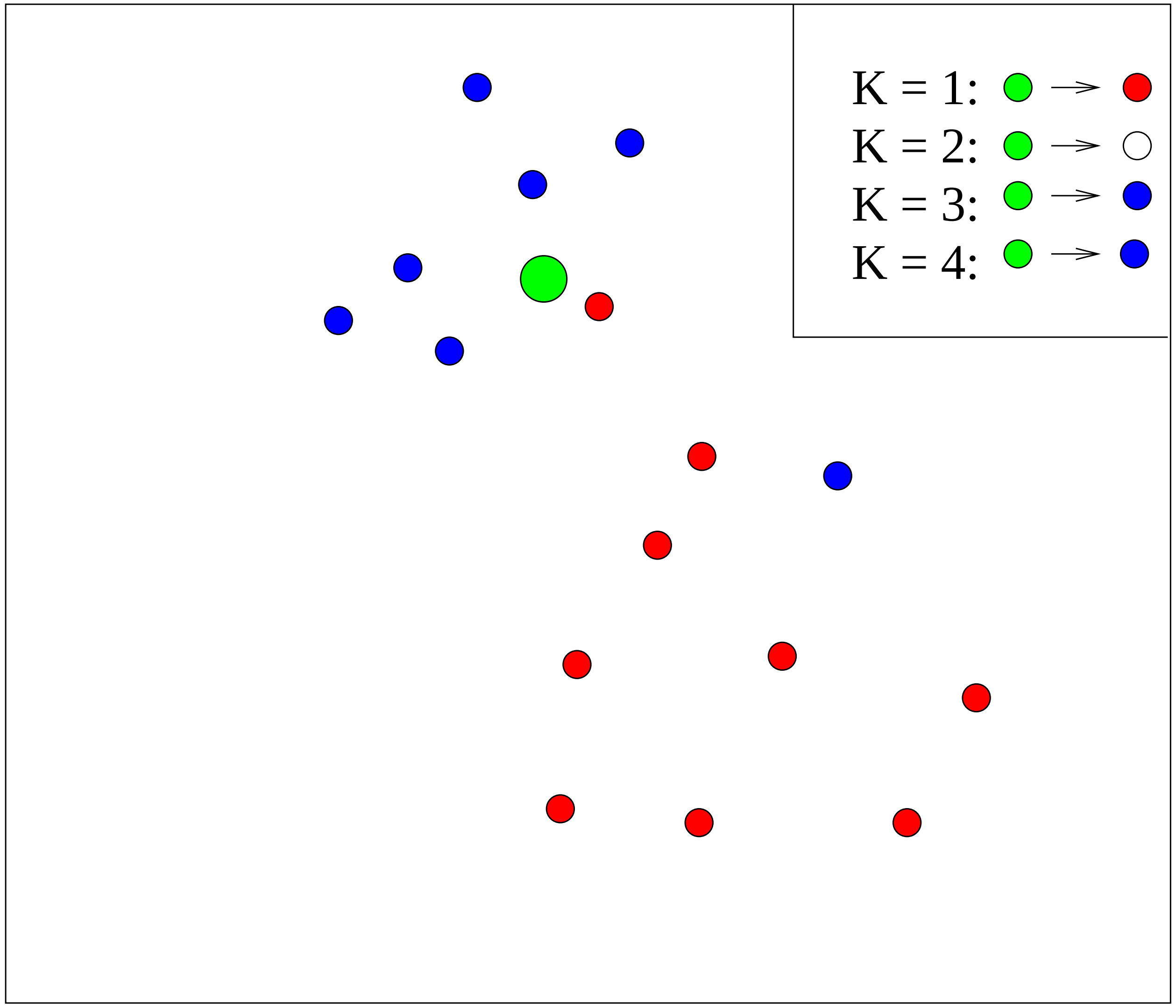
Problem: The Curse of Dimensionality (1)

- First view: combinatorial explosion
  - What happens when the number of input dimensions grows?
  - The number of bins grows exponentially faster!
  - Most bins will get no representative training example
  - How can we estimate a new example that is in one of those bins????
  - In fact, even the bins with some training examples are probably not correctly estimated...

K Nearest Neighbors

- Very simple method, no training necessary
- Needed:
  - a training set $D_n = \{z_1, z_2, \cdots, z_n\}$ with $z_i = (x_i, y_i)$
  - a distance function $L(x_1, x_2)$. For instance, $(x_1 - x_2)^2$
  - a parameter K
- For each test point x
  - select in D_n the K examples that are nearest to x according to $L(x, x_i)$ and keep their index (from D_n) in $\{s_1, \cdots, s_K\}$
  - decision:
    - regression: $\hat{y} = \frac{1}{K}\sum_{i=1}^{K} y_{s_i}$
    - classification: $\hat{y} = \text{sign}\left(\frac{1}{K}\sum_{i=1}^{K} y_{s_i}\right)$
- Capacity controlled by K.

K-NN (Graphical View)



KNN - Some Remarks

- What does it mean to be nearest to an example?
- Often used metric: Euclidean distance

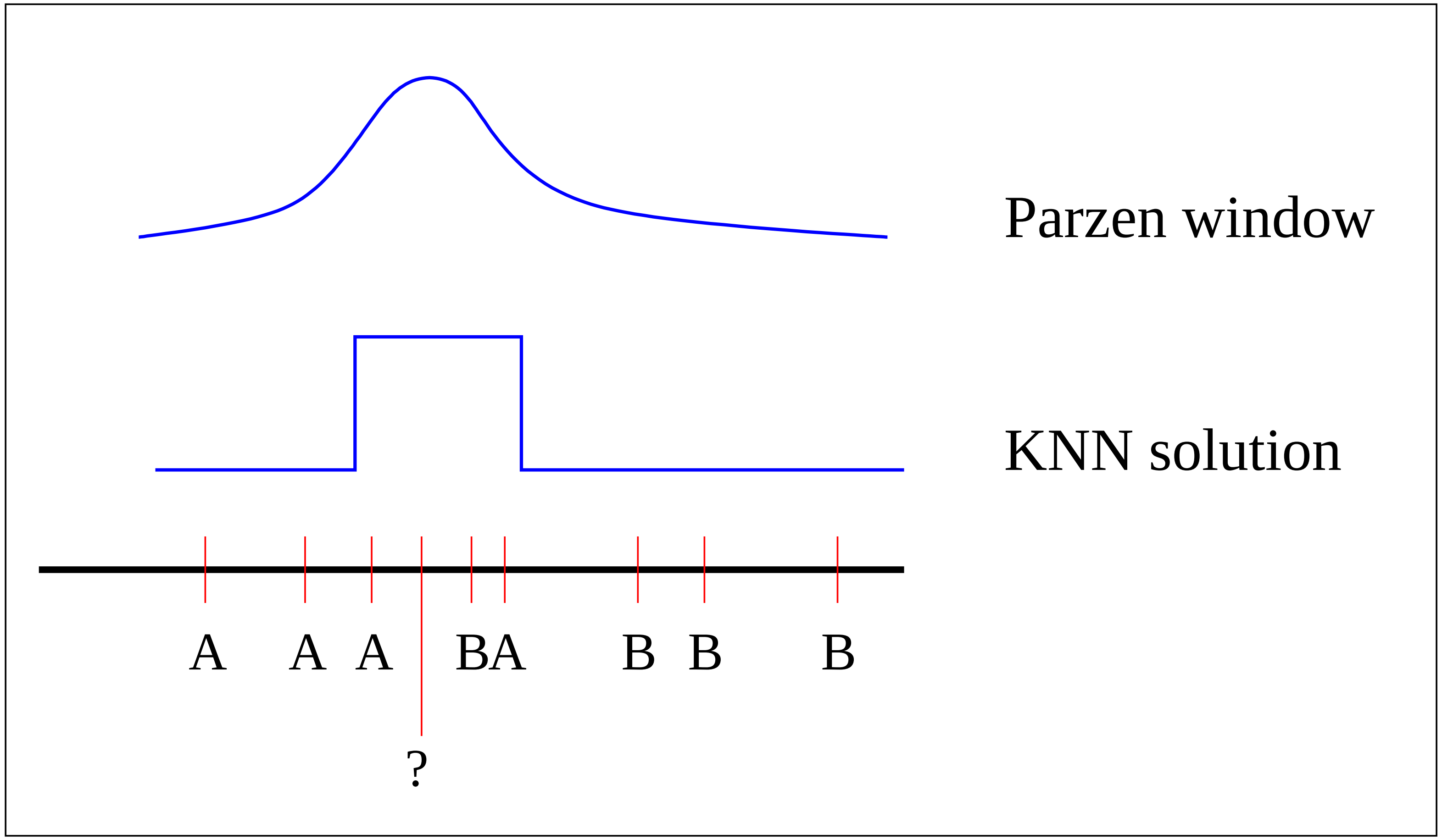
$$d = \sqrt{\sum_i (x_i - t_i)^2}$$

- For KNN, $\sqrt{\cdot}$ is not necessary
- How to select K ???
- Reminder: K controls the capacity...
- Hence, we can use a model selection technique

Problem: The Curse of Dimensionality (2)

- Second view: Euclidean distance
  - In high dimensional spaces, the Euclidean distance between any two random points converges to the same value!
  - Moreover, all points are at the boundary of the hypersphere containing the points.
  - Hence, all methods based on such distance are bound to work on small dimensions only.

KNN versus Parzen Windows



Parzen Windows

- Very simple method, no training necessary
- Needed:
  - a training set $D_n = \{z_1, z_2, \cdots, z_n\}$ with $z_i = (x_i, y_i)$
  - a kernel function $K(x_1, x_2)$. For instance, $\exp(-\frac{||x_1 - x_2||^2}{2\sigma^2})$
- For each test point x (or z for density estimate)
  - decision:
    - regression: $\hat{y} = \dfrac{\sum_{i=1}^{n} y_i K(x, x_i)}{\sum_{i=1}^{n} K(x, x_i)}$
    - classification: $\hat{y} = \text{sign}\,(\text{regression estimate})$
    - density estimate: $\hat{p}(z) = \dfrac{1}{n} \sum_{i=1}^{n} \dfrac{1}{\sqrt{2\pi}\sigma} K(x, x_i)$
- Capacity controlled by σ

Maximum Likelihood for Density Estimation

- Given a set of examples $D_n = \{z_1, z_2, \cdots, z_n\}$
- Objective: find a distribution $p(Z)$ that maximizes the likelihood of future data
- Select a set of distributions $p(Z|\theta)$ with parameters θ.
- The likelihood of D_n (all examples are iid):

$$\mathcal{L}(D_n|\theta) = \prod_{i=1}^{n} p(z_i|\theta)$$

Hence we search for:

$$\begin{aligned} \theta^* &= \arg\max_{\theta} \prod_{i=1}^{n} p(z_i|\theta) \\ &= \arg\min_{\theta} -\sum_{i=1}^{n} \log p(z_i|\theta) \end{aligned}$$

Maximum Likelihood for Gaussians

- Family of one-dimensional Gaussians with $\theta = \{\mu, \sigma\}$

$$\hat{p}(z|\theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(z-\mu)^2}{2\sigma^2}\right)$$

- Maximum likelihood solution:
  - $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} z_i$
  - $\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(z_i - \hat{\mu})^2$

Bayes Decision

- Classification: $z = (x, y) \in \mathbb{R}^d \times \{-1, 1\}$
- Given: true posterior distribution $P(Y|X = x)$
- It can be shown that the decision

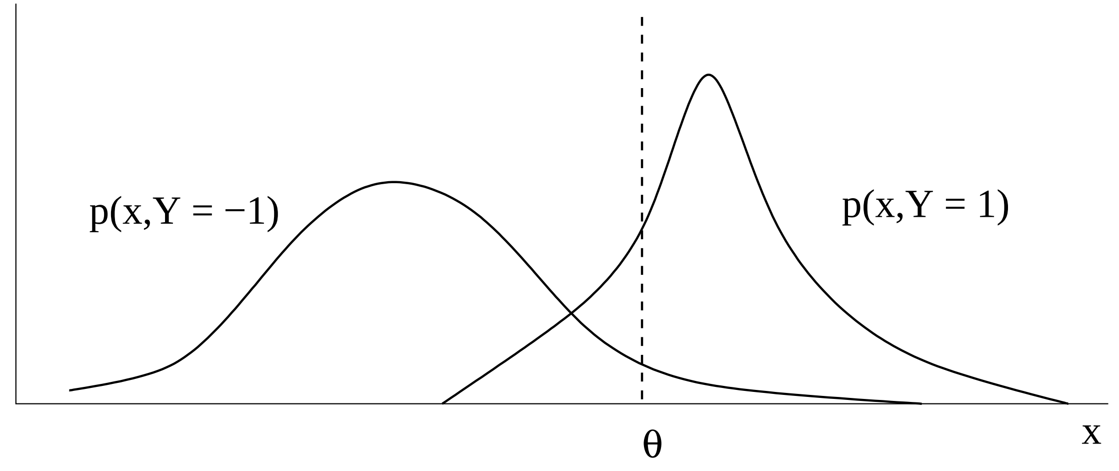
$$\hat{y} = \arg \max_{i \in \{1,-1\}} P(Y = i|X = x)$$

  is optimal in the sense that it minimizes the number of classification errors.
- This decision corresponds to the class maximum a posteriori (MAP) criterion

Why Class MAP Minimizes Error?

$$\begin{aligned}
\hat{y} &= \arg \max_{i \in \{1,-1\}} P(Y=i|X=x) \\
&= \arg \max_{i \in \{1,-1\}} \frac{p(x|Y=i) \cdot P(Y=i)}{p(x)} \\
&= \arg \max_{i \in \{1,-1\}} p(x|Y=i) \cdot P(Y=i) \\
&= \arg \max_{i \in \{1,-1\}} p(x, Y=i)
\end{aligned}$$

- Let us select a threshold for all our decisions $x = \theta$.



Why Class MAP Minimizes Error?

- The probabilities of error are

$$\begin{aligned} p(\text{error}|x > \theta, Y = -1) &= 1 - p(x > \theta, Y = -1) \\ &= \int_{x<\theta} p(x, Y = -1) \end{aligned}$$

$$\begin{aligned} p(\text{error}|x < \theta, Y = 1) &= 1 - p(x < \theta, Y = 1) \\ &= \int_{x>\theta} p(x, Y = 1) \end{aligned}$$

- Which θ corresponds to the *break-even* point?

$$p(x > \theta, Y = -1) = p(x < \theta, Y = 1) \Longrightarrow$$

$$p(x, Y = -1) = p(x, Y = 1)$$

Bayes Classifiers

- Goal: take the decision based on the MAP criterion:

$$\hat{y} = \arg \max_{i \in \{1,-1\}} p(x|Y=i) \cdot P(Y=i)$$

- Hence, you need to estimate:
  - the conditional density $p(x|Y=i)$ for each class i
  - the class prior $P(Y=i)$ for each class i
- Good: each class is estimated independently
- Bad: you learn unnecessary relations
- This technique is nevertheless often used in speech processing

Clustering by K-Means

- Given a set of examples $D_n = \{z_1, z_2, \cdots, z_n\}$
- Search for K prototypes μ_k of disjoint subsets S_k of D_n in order to minimize

$$L = \sum_{k=1}^{K} \sum_{j \in S_k} \sum_{i} (z_j^i - \mu_k^i)^2$$

where z_j^i is the i^{th} coordinate of example z_j, and μ_k is the mean of the examples in subset S_k:

$$\mu_k = \frac{1}{|S_k|} \sum_{j \in S_k} z_j$$

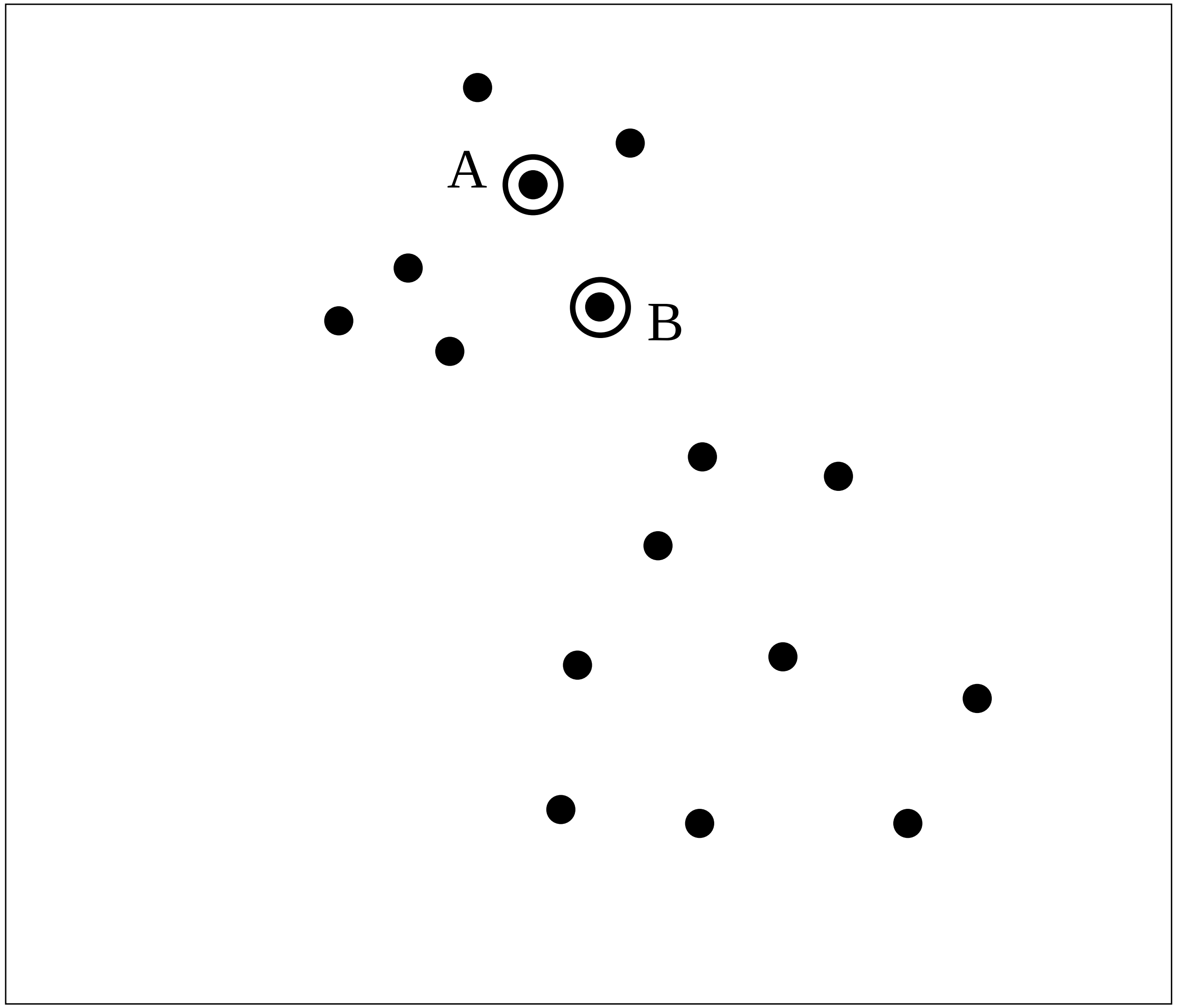
- We could also use another distance metric than Euclidean... (as long as it is a true distance!)

Batch and Stochastic K-Means

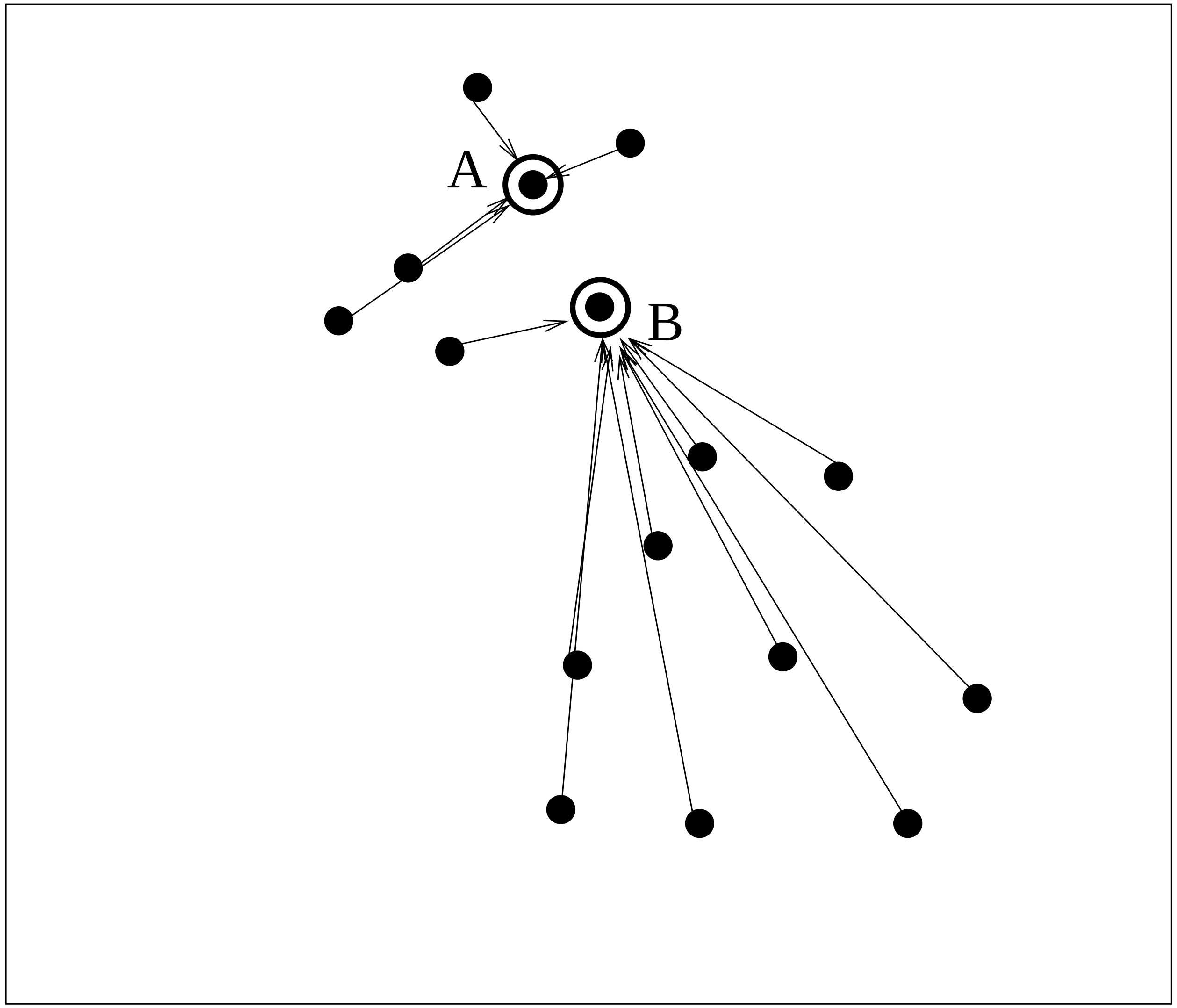
- Initialization: select randomly K examples z_j in D_n as initial values of each μ_k
- At each batch iteration:
  - For each prototype μ_k, put in the emptied set S_k the examples of D_n that are closer to μ_k than to any other $\mu_{j \neq k}$.
  - Re-compute the value of each μ_k as the average of the examples in S_k.
- The algorithm stops when no prototype moves anymore.
- It can be shown that the K-Means criterion will never increase.
- A stochastic version of K-Means can also be derived: given a small η, for each example z_j move the nearest μ_k as follows:

$$\mu_k = \mu_k + \eta(z_j - \mu_k)$$

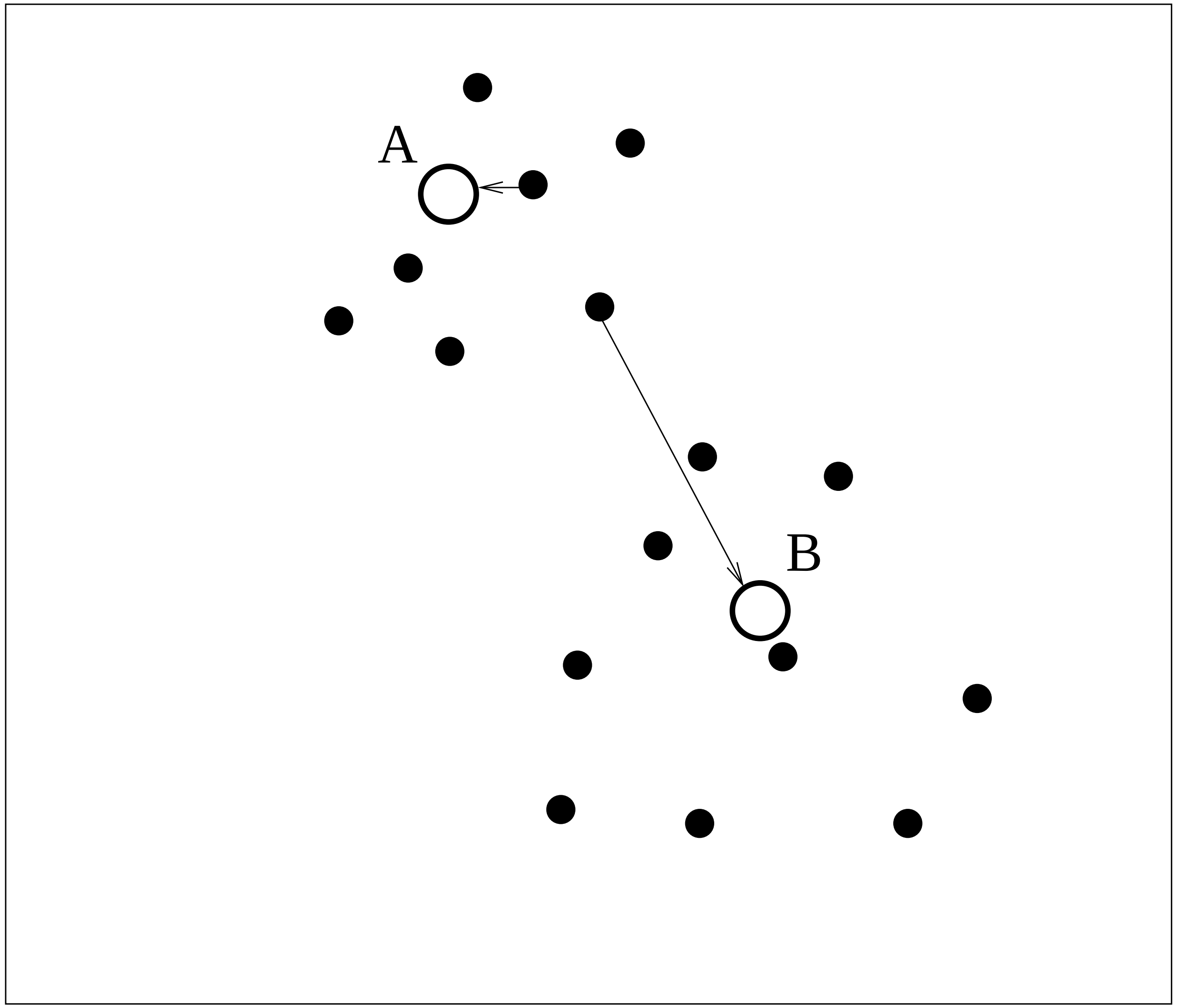
K-Means (Graphical View 1)



K-Means (Graphical View 2)



K-Means (Graphical View 3)



K-Means - Some Remarks

- As for KNN, we can change the metric
- For instance, we can normalize the data
- How to select K ???
- Reminder: as for KNN, K controls the capacity...
- Hence, we can use a model selection technique
- Note: K-Means is quite sensitive to initialization. Other heuristics exist, or you can retrain many times...
- Application: feature extraction

  represent each example z by the index of the closest prototype