Learning From Data
Lecture 19
A Peek At Unsupervised Learning

k-Means Clustering
Probability Density Estimation
Gaussian Mixture Models

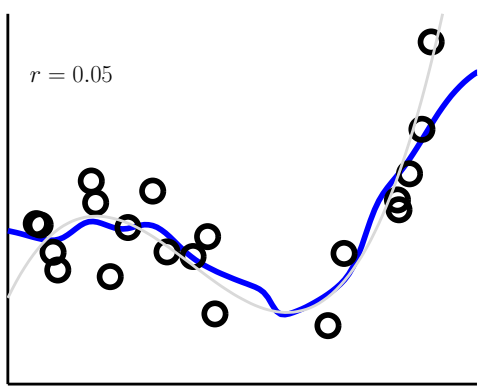
M. Magdon-Ismail
CSCI 4100/6100

RECAP: # Radial Basis Functions

Nonparametric RBF

$$g(\mathbf{x}) = \sum_{n=1}^{N} \left(\frac{\alpha_n(\mathbf{x})}{\sum_{m=1}^{N} \alpha_m(\mathbf{x})} \right) \cdot y_n$$

$$\alpha_n(\mathbf{x}) = \phi\left(\frac{\|\mathbf{x}-\mathbf{x}_n\|}{r}\right) \qquad \text{(bump on } \mathbf{x}\text{)}$$



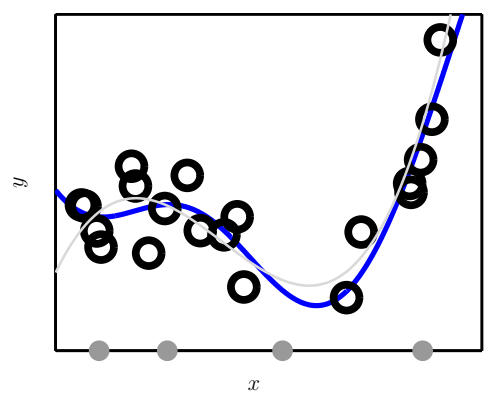
No Training

Parametric k-RBF-Network

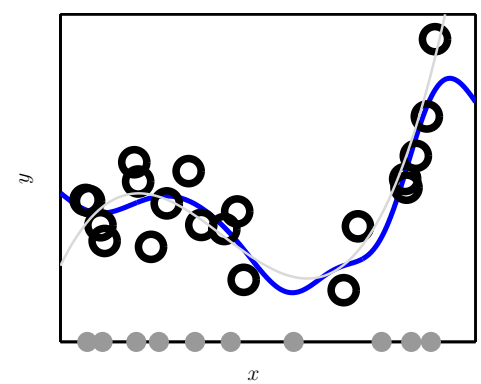
$$\begin{aligned} h(\mathbf{x}) &= w_0 + \sum_{j=1}^{k} w_j \cdot \phi\left(\frac{\|\mathbf{x} - \boldsymbol{\mu}_j\|}{r}\right) \\ &= \mathbf{w}^{\mathrm{T}} \boldsymbol{\Phi}(\mathbf{x}) \end{aligned}$$

(bump on $\boldsymbol{\mu}_j$)

linear model given $\boldsymbol{\mu}_j$
choose $\boldsymbol{\mu}_j$ as centers of k-clusters of data



$k = 4, r = \frac{1}{k}$



$k = 10$, regularized

Unsupervised Learning

- Preprocessor to organize the data for supervised learning:

  Organize data for faster nearest neighbor search

  Determine centers for RBF bumps.

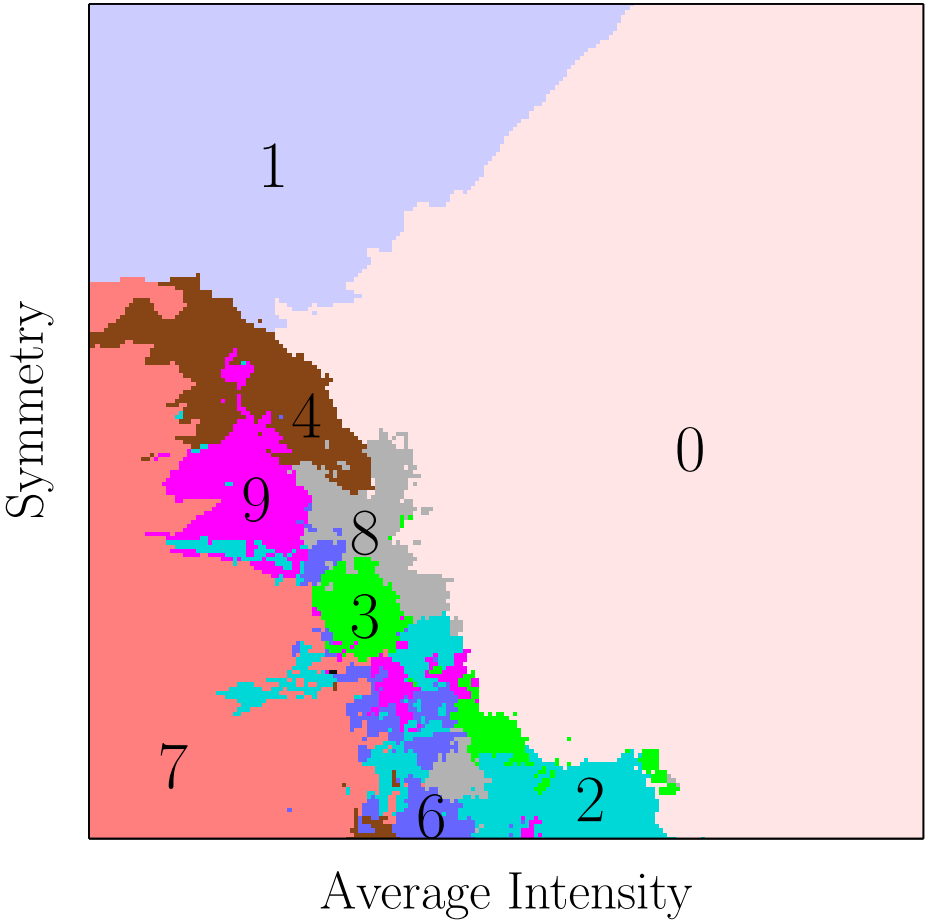
- Important to be able to organize the data to identify patterns.

  Learn the patterns in data, e.g. the patterns in a language before getting into a supervised setting.

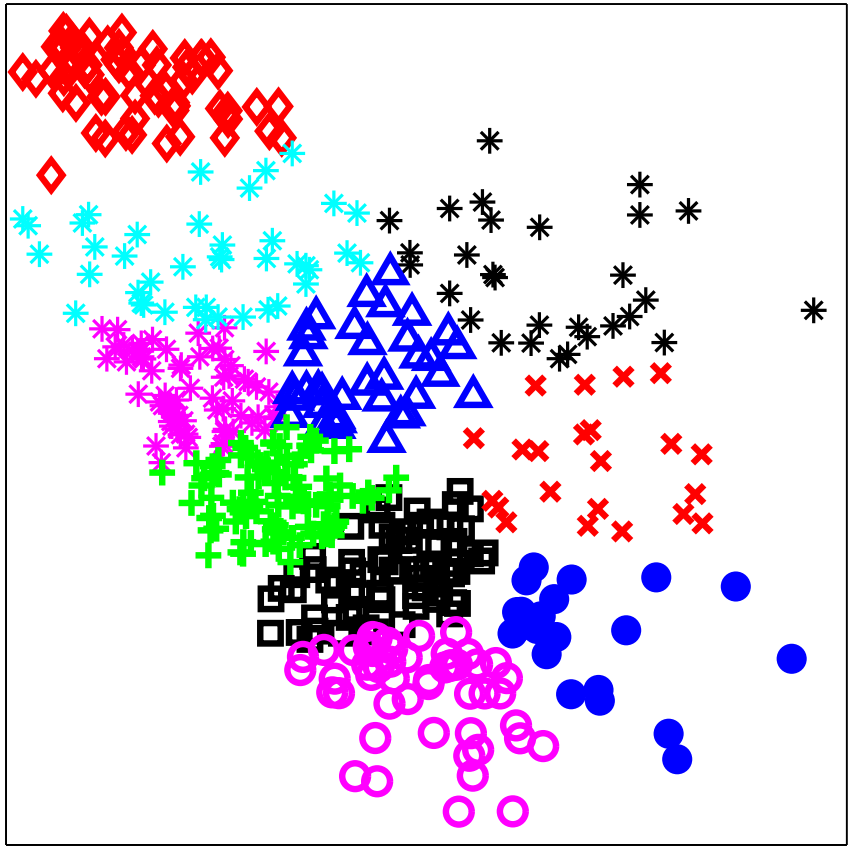
  amazon.com organizes books into categories

Clustering Digits

21-NN rule, 10 Classes



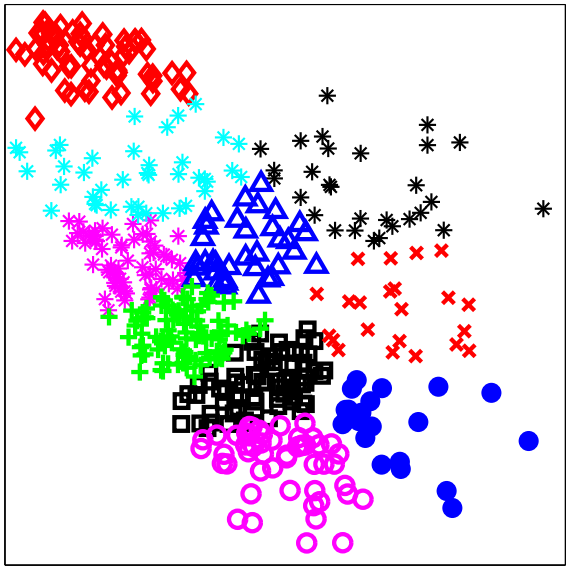
10 Clustering of Data



Clustering

A *cluster* is a collection of points S

A k-clustering is a partition of the data into k clusters $S_1, \ldots, S_k$.



$$\cup_{j=1}^{k} S_j = \mathcal{D}$$

$$S_i \cap S_j = \emptyset \qquad \text{for } i \neq j$$

Each cluster has a center $\boldsymbol{\mu}_j$

How good is a clustering?

Points in a cluster should be similar (close to each other, and the center)

Error in cluster j:

$$E_j = \sum_{\mathbf{x}_n \in S_j} \| \mathbf{x}_n - \boldsymbol{\mu}_j \|^2.$$

k-Means Clustering Error:

$$\begin{aligned} E_{\text{in}}(S_1, \ldots, S_k; \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k) &= \sum_{j=1}^{k} E_j \\ &= \sum_{n=1}^{N} \| \mathbf{x}_n - \boldsymbol{\mu}(\mathbf{x}_n) \|^2 \end{aligned}$$

$\boldsymbol{\mu}(\mathbf{x}_n)$ is the center of the cluster to which $\mathbf{x}_n$ belongs.

k-Means Clustering

You get to pick $S_1, \ldots, S_k$ *and* $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$ to minimize $E_{\text{in}}(S_1, \ldots, S_k; \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k)$

If centers $\boldsymbol{\mu}_j$ are known, picking the sets is easy:

Add to S_j all points closest to $\boldsymbol{\mu}_j$

If the clusters S_j are known, picking the centers is easy:

Center $\boldsymbol{\mu}_j$ is the centroid of cluster S_j

$$\boldsymbol{\mu}_j = \frac{1}{|S_j|} \sum_{\mathbf{x}_n \in S_j} \mathbf{x}_n$$

Lloyd's Algorithm for k-Means Clustering

$$E_{\text{in}}(S_1, \ldots, S_k; \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k) = \sum_{n=1}^{N} \| \mathbf{x}_n - \boldsymbol{\mu}(\mathbf{x}_n) \|^2$$

1: **Initialize** Pick well separated centers $\boldsymbol{\mu}_j$.

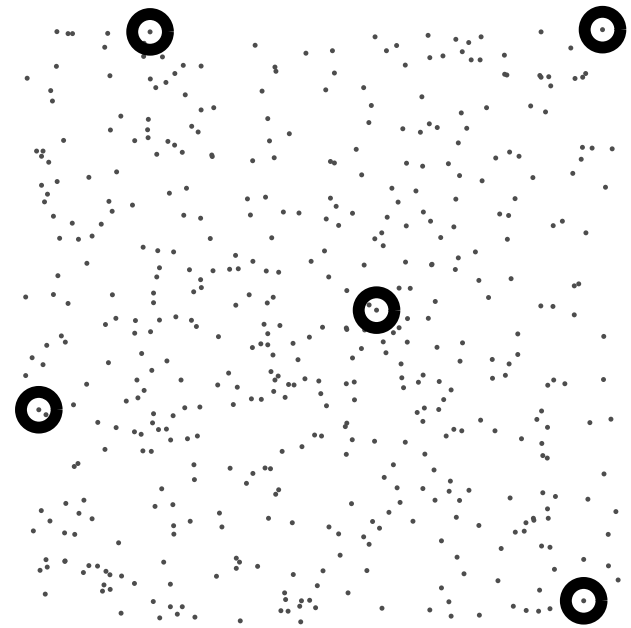
2: **Update** S_j to be all points closest $\boldsymbol{\mu}_j$.

$$S_j \leftarrow \{\mathbf{x}_n : \| \mathbf{x}_n - \boldsymbol{\mu}_j \| \leq \| \mathbf{x}_n - \boldsymbol{\mu}_\ell \| \text{ for } \ell = 1, \ldots, k\}.$$

3: **Update** $\boldsymbol{\mu}_j$ to the centroid of S_j.

$$\boldsymbol{\mu}_j \leftarrow \frac{1}{|S_j|} \sum_{\mathbf{x}_n \in S_j} \mathbf{x}_n$$

4: Repeat steps 2 and 3 until E_{in} stops decreasing.



Lloyd's Algorithm for k-Means Clustering

$$E_{\text{in}}(S_1, \ldots, S_k; \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k) = \sum_{n=1}^{N} \| \mathbf{x}_n - \boldsymbol{\mu}(\mathbf{x}_n) \|^2$$

1: **Initialize** Pick well separated centers $\boldsymbol{\mu}_j$.

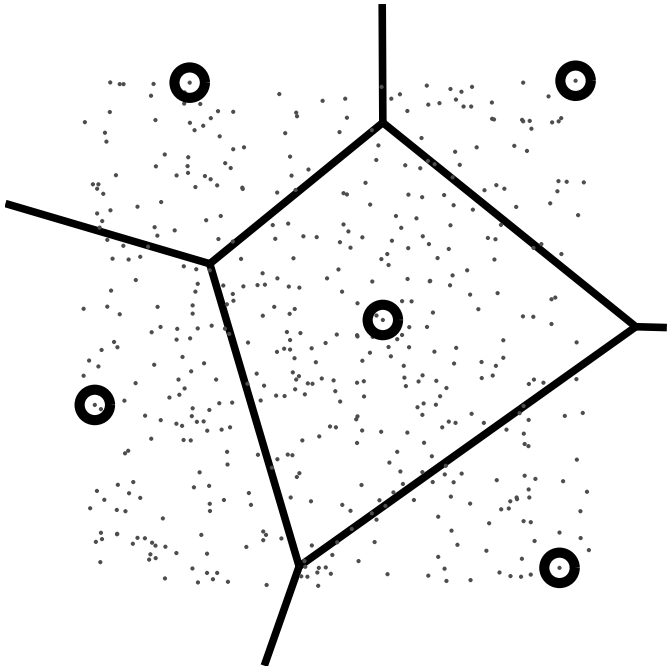
2: **Update** S_j to be all points closest $\boldsymbol{\mu}_j$.

$$S_j \leftarrow \{\mathbf{x}_n : \| \mathbf{x}_n - \boldsymbol{\mu}_j \| \leq \| \mathbf{x}_n - \boldsymbol{\mu}_\ell \| \text{ for } \ell = 1, \ldots, k\}.$$

3: **Update** $\boldsymbol{\mu}_j$ to the centroid of S_j.

$$\boldsymbol{\mu}_j \leftarrow \frac{1}{|S_j|} \sum_{\mathbf{x}_n \in S_j} \mathbf{x}_n$$

4: Repeat steps 2 and 3 until E_{in} stops decreasing.



Lloyd's Algorithm for k-Means Clustering

$$E_{\text{in}}(S_1, \ldots, S_k; \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k) = \sum_{n=1}^{N} \| \mathbf{x}_n - \boldsymbol{\mu}(\mathbf{x}_n) \|^2$$

1: **Initialize** Pick well separated centers $\boldsymbol{\mu}_j$.

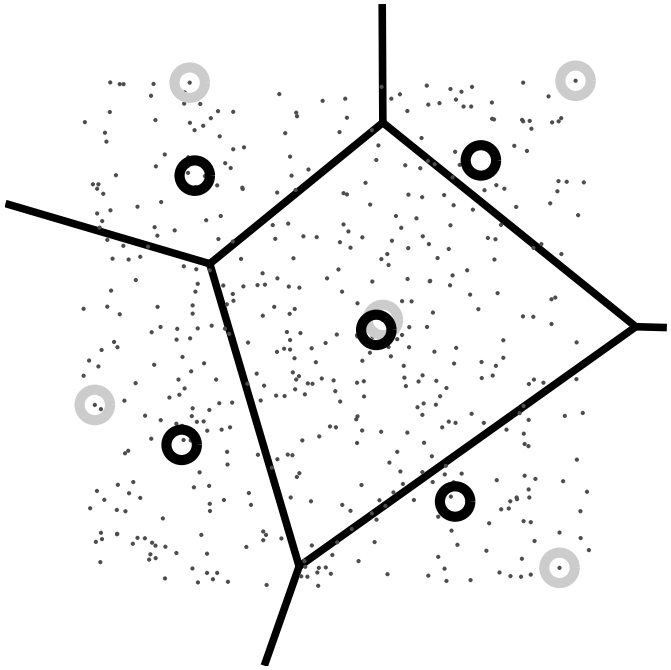
2: **Update** S_j to be all points closest $\boldsymbol{\mu}_j$.

$$S_j \leftarrow \{\mathbf{x}_n : \| \mathbf{x}_n - \boldsymbol{\mu}_j \| \leq \| \mathbf{x}_n - \boldsymbol{\mu}_\ell \| \text{ for } \ell = 1, \ldots, k\}.$$

3: **Update** $\boldsymbol{\mu}_j$ to the centroid of S_j.

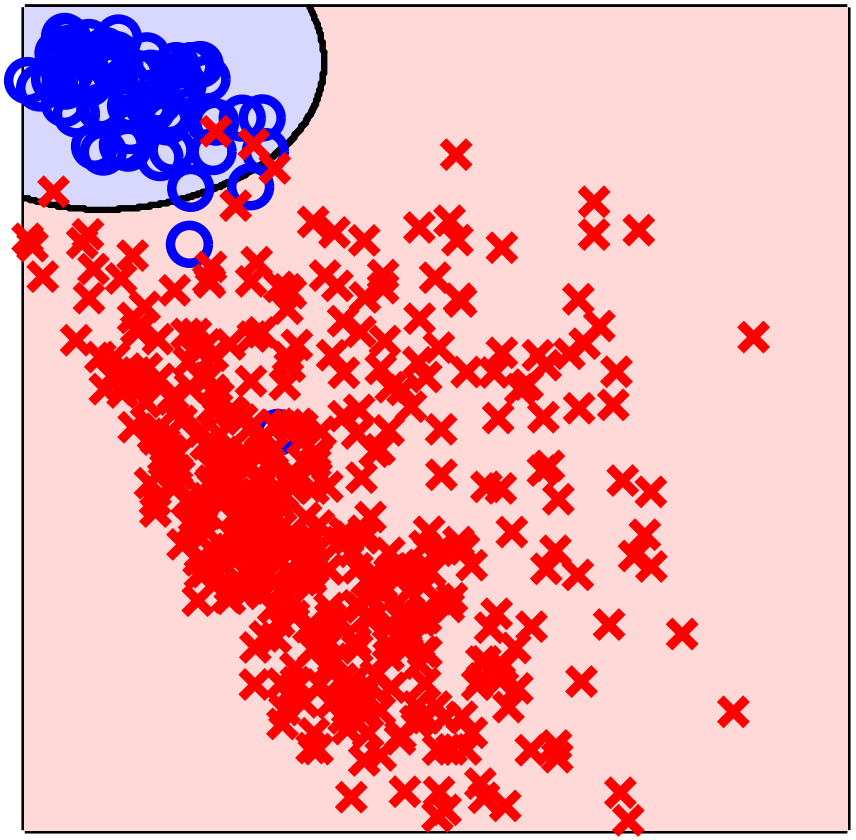
$$\boldsymbol{\mu}_j \leftarrow \frac{1}{|S_j|} \sum_{\mathbf{x}_n \in S_j} \mathbf{x}_n$$

4: Repeat steps 2 and 3 until E_{in} stops decreasing.

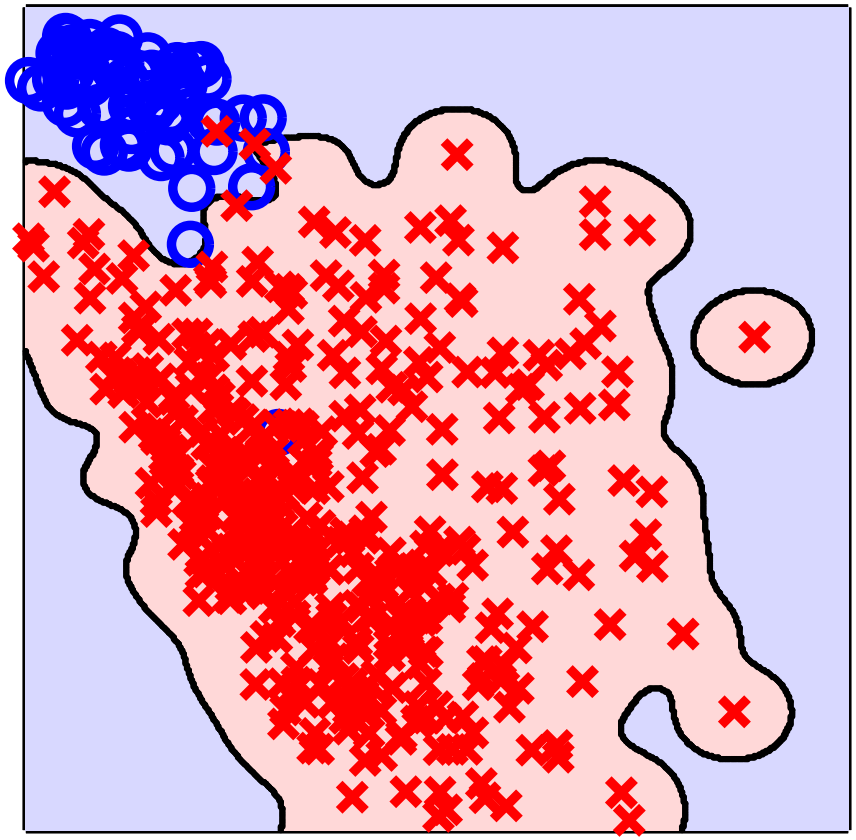


Application to k-RBF-Network

10-center RBF-network

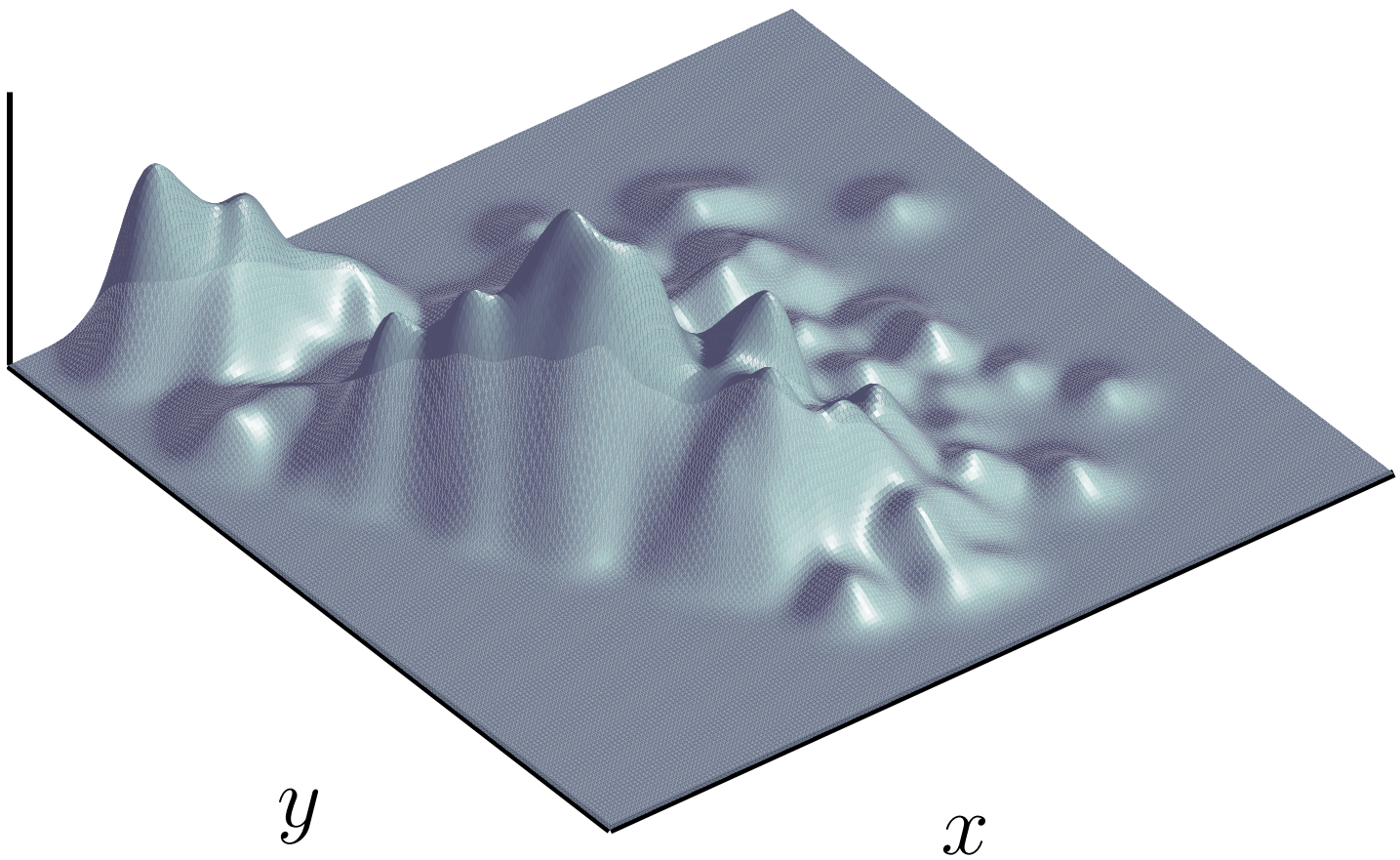


300-center RBF-network



Choosing k - knowledge of problem (10 digits) or CV.

Probability Density Estimation



Probability Density Estimation

$$P(\mathbf{x})$$

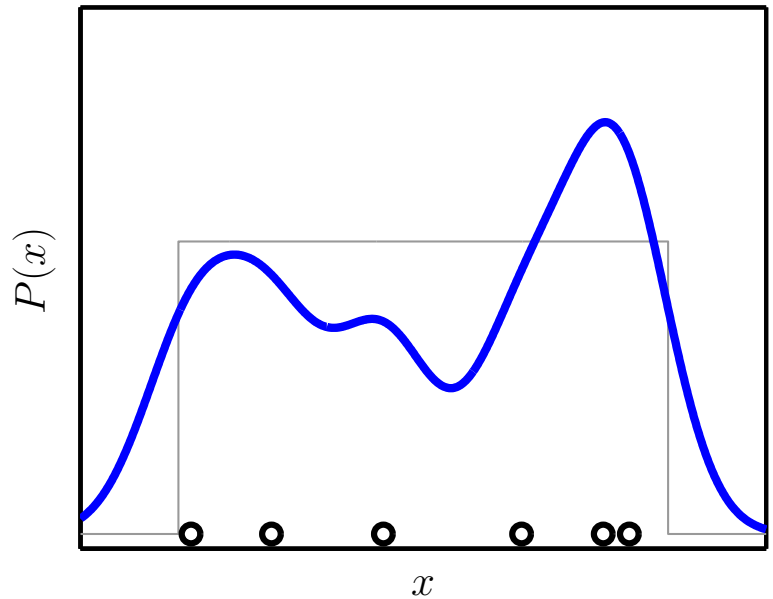
$P(\mathbf{x})$ measures how likely it is to generate inputs *similar* to $\mathbf{x}$.

Estimating $P(\mathbf{x})$ results in a 'softer/finer' representation than clustering

Clusters are regions of high probability.

Parzen Windows – RBF density estimation

Basic idea: put a bump of 'size' (volume) $\frac{1}{N}$ on each data point.

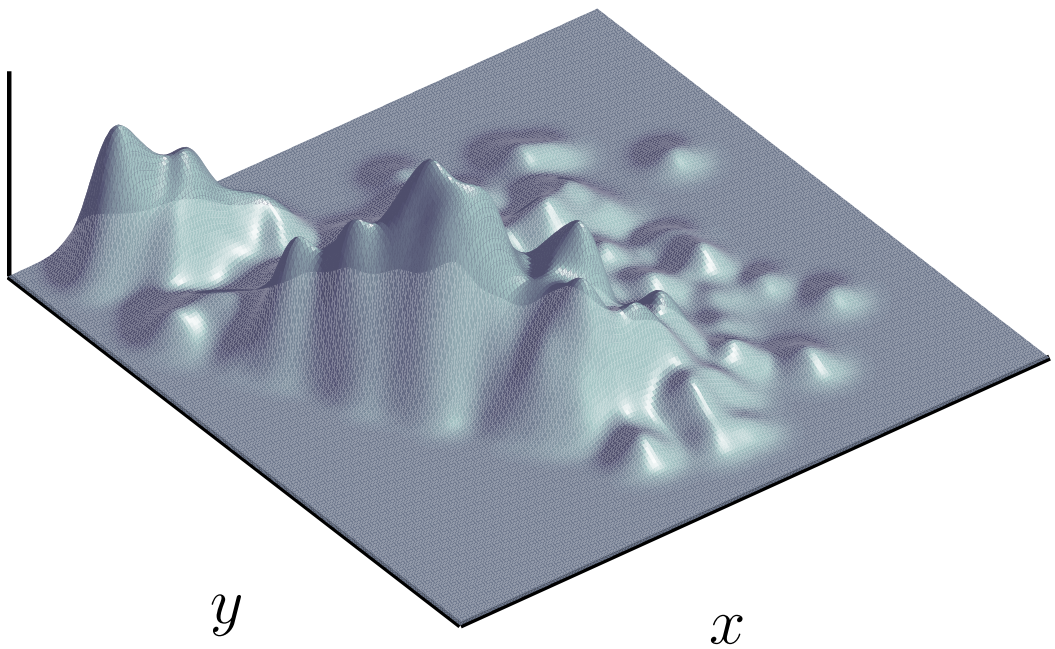


$$\hat{P}(\mathbf{x}) = \frac{1}{Nr^d} \sum_{i=1}^{N} \phi\left(\frac{\| \mathbf{x} - \mathbf{x}_i \|}{r}\right)$$

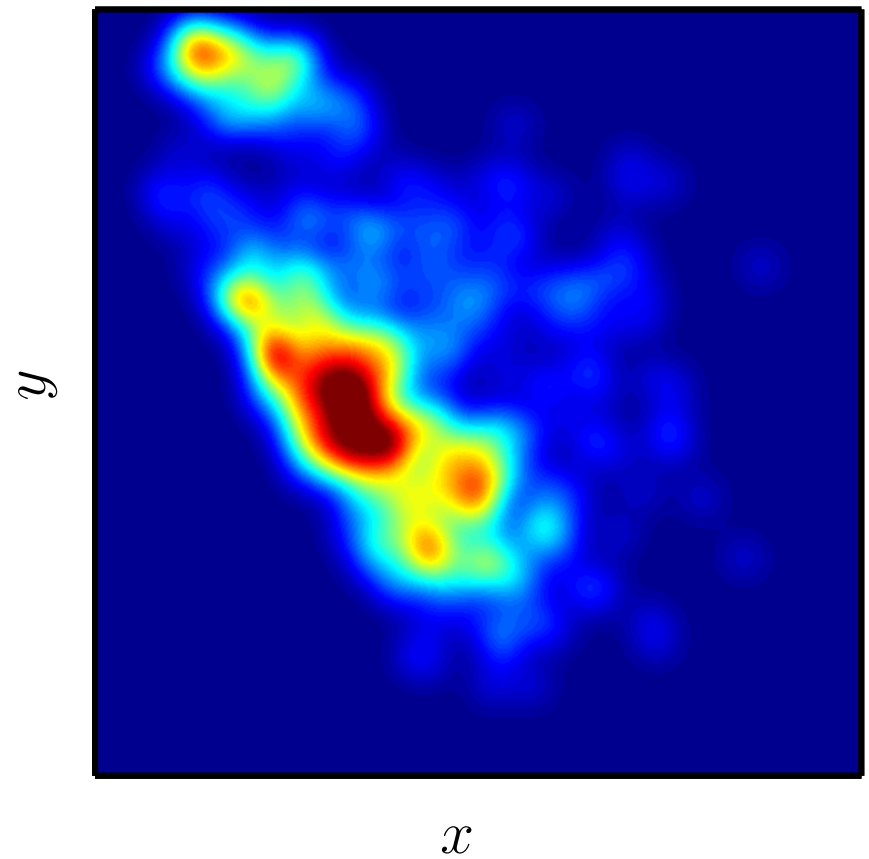
$\phi(z) = \frac{1}{(2\pi)^{d/2}} e^{-\frac{1}{2}z^2}$

Digits Data

RBF Density Estimate



Density Contours



The Gaussian Mixture Model (GMM)

Instead of N bumps $\longrightarrow$ $k \ll N$ bumps. (Similar to nonparametric RBF $\longrightarrow$ parametric k-RBF-network)

Instead of uniform spherical bumps $\longrightarrow$ each bump has its own shape.

Bump centers:

$$\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$$

Bump shapes:

$$\Sigma_1, \ldots, \Sigma_k$$

Gaussian formula for the bump:

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_j, \Sigma_j) = \frac{1}{(2\pi)^{d/2}|\Sigma_j|^{1/2}} e^{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_j)^{\mathrm{T}} \Sigma_j{}^{-1} (\mathbf{x} - \boldsymbol{\mu}_j)}.$$

GMM Density Estimate

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_j, \Sigma_j) = \frac{1}{(2\pi)^{d/2}|\Sigma_j|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_j)^{\mathrm{T}}\Sigma_j{}^{-1}(\mathbf{x}-\boldsymbol{\mu}_j)}.$$

$$\hat{P}(\mathbf{x}) = \sum_{j=1}^{k} w_j \, \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_j, \Sigma_j)$$

(Sum of k weighted bumps).

$$w_j > 0, \qquad \sum_{j=1}^{k} w_j = 1$$

You get to pick $\{w_j, \boldsymbol{\mu}_j, \Sigma_j\}_{j=1,\ldots,k}$

Maximize Likelihood Estimation

Pick $\{w_j, \boldsymbol{\mu}_j, \Sigma_j\}_{j=1,\ldots,k}$ to best explain the data.

Maximize the likelihood of the data given $\{w_j, \boldsymbol{\mu}_j, \Sigma_j\}_{j=1,\ldots,k}$

(We saw this when we derived the cross entropy error for logistic regression)

Expectation-Maximization: The E-M Algorithm

A simple algorithm to get to the local minimum of the likelihood.

Partition variables into two sets. Given one-set, you can estimate the other

'Bootstrap' your way to a decent solution.

Lloyd's algorithm for k-means is an example for 'hard clustering'

Bump Memberships

Fraction of $\mathbf{x}_n$ belonging to bump j (a 'hidden variable')

$$\gamma_{nj}$$

Bump Memberships

Fraction of $\mathbf{x}_n$ belonging to bump j (a 'hidden variable')

$$\gamma_{nj}$$

$$N_j = \sum_{n=1}^{N} \gamma_{nj} \qquad \text{('number' of points in bump } j\text{)}$$

$$w_j = \frac{N_j}{N} \qquad \text{(probability bump } j\text{)}$$

$$\boldsymbol{\mu}_j = \frac{1}{N_j} \sum_{n=1}^{N} \gamma_{nj} \mathbf{x}_n \qquad \text{(centroid of bump } j\text{)}$$

$$\Sigma_j = \frac{1}{N_j} \sum_{n=1}^{N} \gamma_{nj} \mathbf{x}_n \mathbf{x}_n^{\mathrm{T}} - \boldsymbol{\mu}_j \boldsymbol{\mu}_j^{\mathrm{T}} \qquad \text{(covariance matrix of bump } j\text{)}$$

Re-Estimating Bump Memberships

$$\gamma_{nj} = \frac{w_j \, \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_j, \Sigma_j)}{\sum_{\ell=1}^{k} w_\ell \, \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_\ell, \Sigma_\ell)}$$

γ_{nj} is the probability that $\mathbf{x}_n$ came from bump j

probability of bump j: w_j
probability density for $\mathbf{x}_n$ <u>given</u> bump j: $\mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_j, \Sigma_j)$

E-M Algorithm

E-M Algorithm for GMMs:

1: Start with estimates for the bump membership γ_{nj}.

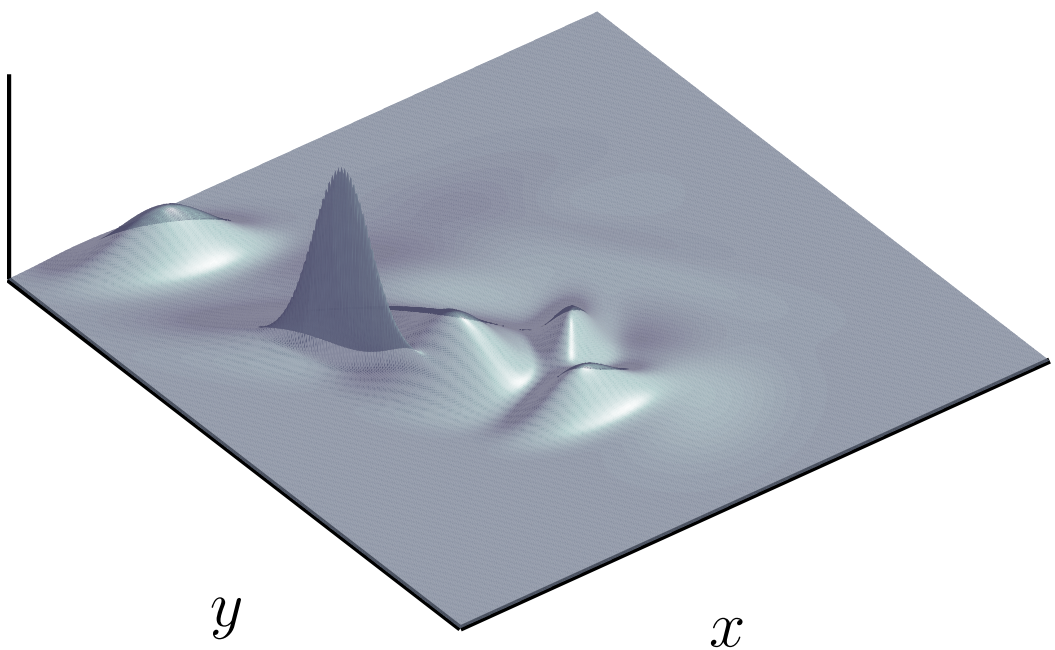
2: Estimate $w_j, \boldsymbol{\mu}_j, \Sigma_j$ given the bump memberships.

3: Update the bump memberships given $w_j, \boldsymbol{\mu}_j, \Sigma_j$;

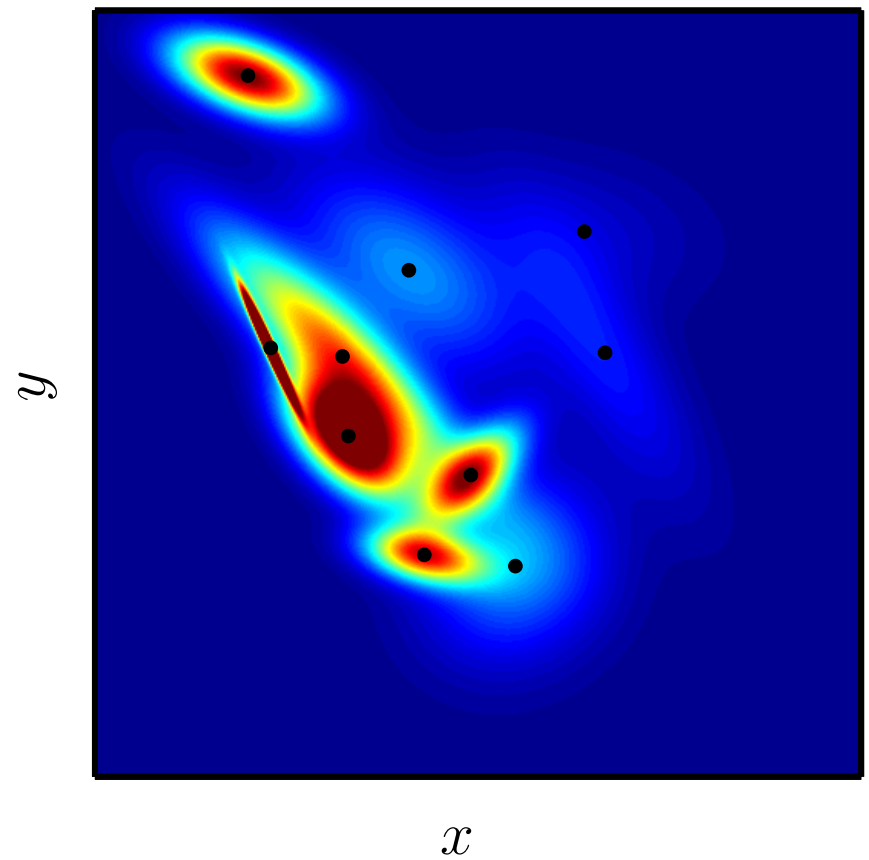
4: Iterate to step 2 until convergence.

GMM on Digits Data

10-center GMM

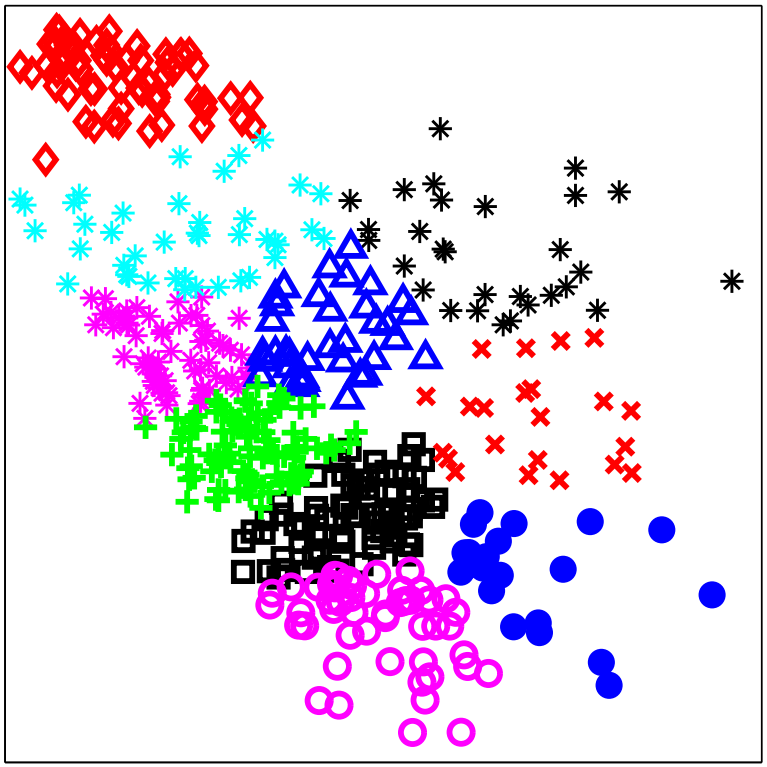


Density Contours



Compare K-Means and GMM

K-Means, $K = 10$



GMM, $K = 10$