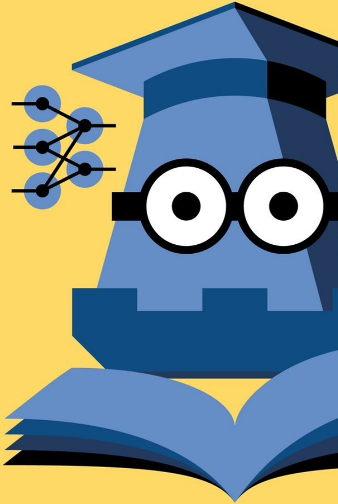
PETNICA SUMMER INSTITUTE
MACHINE LEARNING 6

# Basics of Unsupervised Learning



Mladen Nikolić

Faculty of Mathematics
University of Belgrade

Everseen

# Overview

Types of Unsupervised Learning

Dimensionality Reduction / Reprezentation Learning

Clustering

# Overview

Types of Unsupervised Learning

Dimensionality Reduction / Reprezentation Learning

Clustering

# Unsupervised learning

- Model should identify some relevant structure in the data
- Input data consists only of feature values, there are no target values
- Task to be learned is defined by the algorithm – for different kinds of tasks, different learning algorithms are formulated
- Typical tasks:
  - Dimensionality reduction
  - Representation learning
  - Clustering

# Dimensionality reduction

- Identification of subspaces (planes or manifolds) in which data lie
- PCA, autoencoders, *t*-SNE, ...
- Mostly used for data preprocessing and visualisation

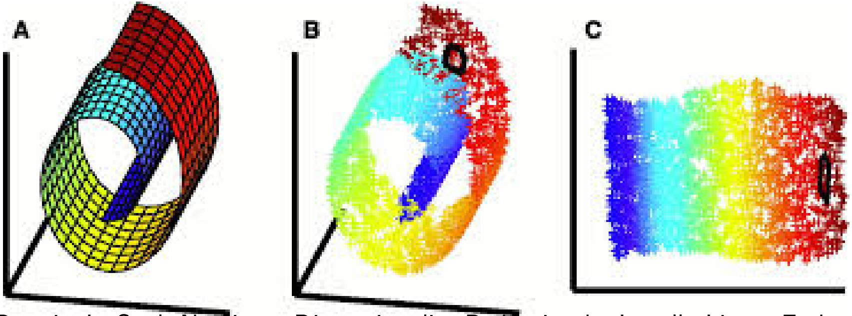



Figure: S. Roweis, L. Saul, Nonlinear Dimensionality Reduction by Locally Linear Embeddings

# Representation learning

- Finding representations in data which facilitate exploitation of relevant information
- Can include dimensionality reduction
- PCA, autoencoders, VAEs, GANs, word2vec,...
- Used for natural language understanding, semantic image manipulation, improvement of other algorithms...

# Clustering

- Identification of groups of data
- Grouping can be defined based on proximity, density, shape, ...
- $k$ means, DBSCAN, Gaussian mixture, agglomerative hierarchical clustering, ...
- Tasks like community detection in social networks, human genetic clustering, detection of different types of tissue in medical imaging, data reduction
- Interesting both in its own right and as a data preprocessing technique

# Clustering illustration

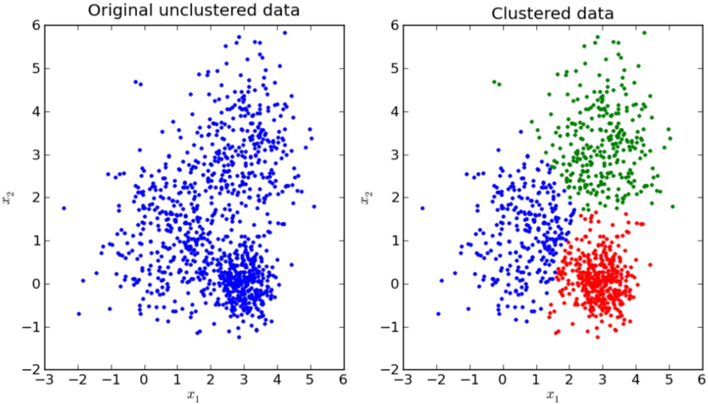



Figure: `https://towardsdatascience.com/k-means-data-clustering-bce3335d2203`

# Overview

Types of Unsupervised Learning

Dimensionality Reduction / Reprezentation Learning

Clustering

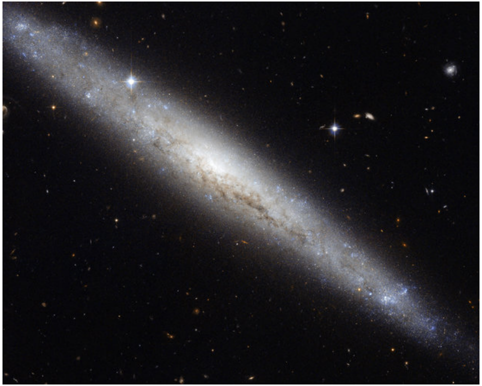
# Towards principal component analysis

- Data usually lie in a small dimensional surface within the feature space
- Consider faces
- Assume a linear surface – a plane
- Data need not lie perfectly on that plane, but we can find the best one and project the data to it
- Some information may be lost

# Data variability

- Variable values change as we switch from instance to instance
- The way they change together reflects the dependencies among variables
- If they were constant, there would be no information about dependencies of variables
- Variability is important!

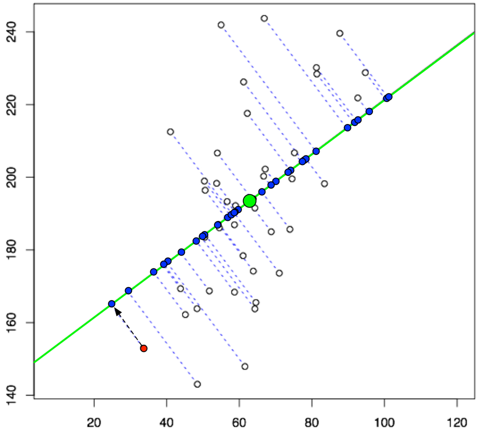
# How to choose a plane?



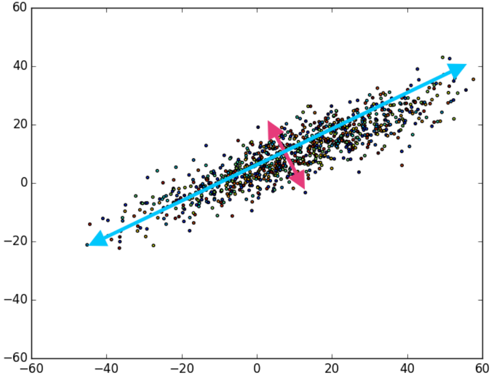
# How to choose a plane?

- Among planes of some dimension, choose one which preserves the most variation when data is projected to it
- How many dimensions?
- Let's start with a line

# Variability along the line



# Principal components



## Variance and covariance

- For sample of values $x_1, \dots, x_n$, variance is defined by:

$$Var[X] = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \overline{x})^2$$

- Covariance of two variables measures how they vary together

$$Cov[X, Y] = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \overline{x})(y_i - \overline{y})$$

- For variables $X_1, \dots, X_n$ we can define covariance matrix $\Sigma$ such that

$$\Sigma_{ij} = Cov[X_i, X_j]$$

# Variance of the data

- Variation of the data:

$$Var[X] = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2$$

- Variation of the data projected on unit vector $d$:

$$Var[Xd] = \frac{1}{N-1}\sum_{i=1}^{N}(x_i \cdot d - \overline{x} \cdot d)^2$$

- If we center the data (s.t. $\overline{x}=0$) prior to computation:

$$Var[Xd] = \frac{1}{N-1}\sum_{i=1}^{N}(x_i \cdot d)^2$$

# First principal component

- Find the direction of maximal variance:

$$\arg\max_{\|d\|=1} \sum_{i=1}^{N} (x_i \cdot d)^2 =$$

$$\arg\max_{\|d\|=1} \|Xd\|_2^2 =$$

$$\arg\max_{\|d\|=1} d^T X^T X d =$$

$$\arg\max_{\|d\|=1} d^T \Sigma d$$

# Properties of $\Sigma$

- If it holds $Av = \lambda v$ for $v \neq 0$, $v$ is an eigenvector of matrix $A$ and $\lambda$ is its corresponding eigenvalue
- $\Sigma = X^T X$ is a symmetric matrix with orthonormal eigenvectors $v_i$ corresponding to different eigenvalues $\lambda_i$ which are real and nonnegative
- Assume $\lambda_i$ are sorted in decreasing order
- We can use them to form an orthonormal basis of the space in which data lie and decompose

$$d = \sum_{i=1}^{n} \alpha_i v_i$$

# First principal component

- Find the direction of maximal variance:

$$\arg\max_{\|d\|=1} d^T \Sigma d =$$

$$\arg\max_{\|d\|=1} \left( \sum_{i=1}^{n} \alpha_i v_i \right) \Sigma \left( \sum_{j=1}^{n} \alpha_j v_j \right) =$$

$$\arg\max_{\|d\|=1} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j v_i^T \Sigma v_j =$$

$$\arg\max_{\|d\|=1} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \lambda_j v_i^T v_j =$$

$$\arg\max_{\|d\|=1} \sum_{i=1}^{n} \alpha_i^2 \lambda_i$$

- How to maximize with respect to $d$?

# First principal component

- Find the direction of maximal variance:

$$\arg\max_{\|d\|=1} \sum_{i=1}^{n} \alpha_i^2 \lambda_i$$

- $\alpha_1, \ldots, \alpha_n$ are constrained by the budget $\|d\| = 1$:

$$1 = \|d\|^2 = d^T d = \left(\sum_{i=1}^{n} \alpha_i v_i\right)^T \left(\sum_{i=1}^{n} \alpha_i v_i\right) = \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j v_i^T v_j = \sum_{i=1}^{n} \alpha_i^2$$

- Spend the whole budget at $\lambda_1$ since it is the greatest
- Therefore, eigenvector of $\Sigma$ is the first principal component

# How much variation?

- Fraction of variation captured by the plane spanned by first $k$ eigenvectors is

$$\sum_{i=1}^{k} \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j}$$

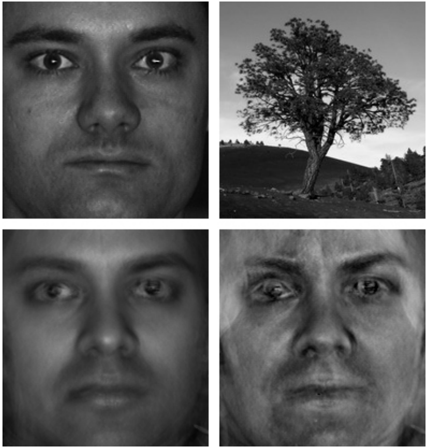
# Dimensionality reduction

- Pick dominant eigenvectors to explain enough variance and arrange them as columns of matrix V
- Transform the data to new feature space by: $X' = XV$
- If needed, data can be returned to original feature space by: $X'V^T$

# Face detection via PCA

# Eigenfaces

# Projection of images to eigenface space

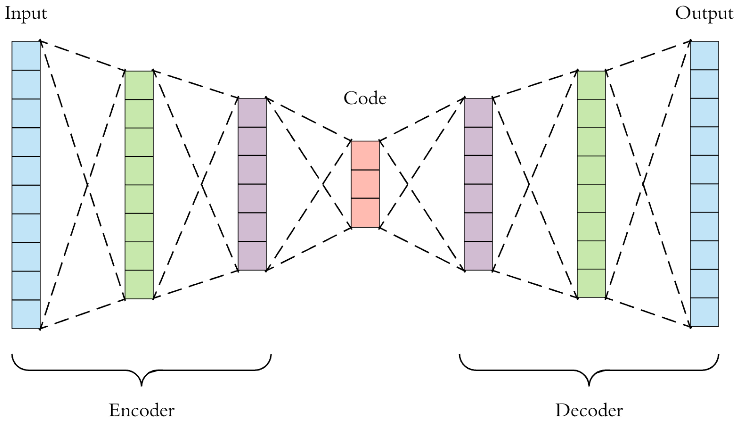


# Autoencoder

- Nonlinear cousin of PCA
- Neural network of specific architecture
- Minimizes reconstruction loss:

$$\min_{w} \sum_{i=1}^{N} \|x_i - f_w(x_i)\|_2^2$$

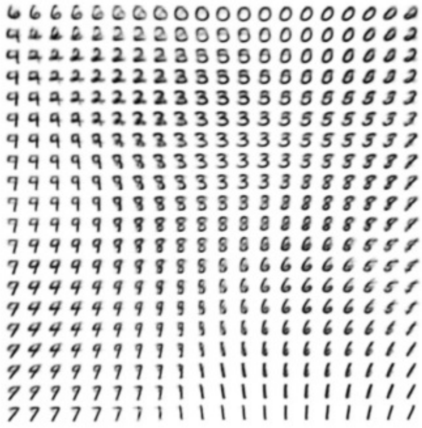
# Autoencoder architecture



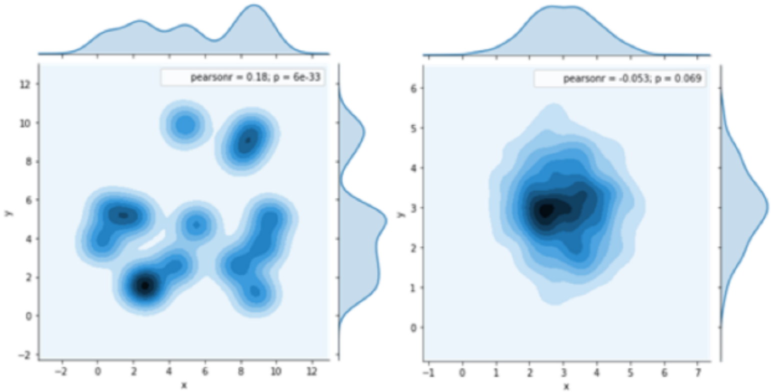
# Latent space

- Bottleneck principle
- Encoder maps from feature space to latent space
- Decoder maps from latent space to feature space, thus parametrizing the data surface
- By moving around latent space and decoding, we move around data surface
- Similarities are preserved to some degree

# Latent space



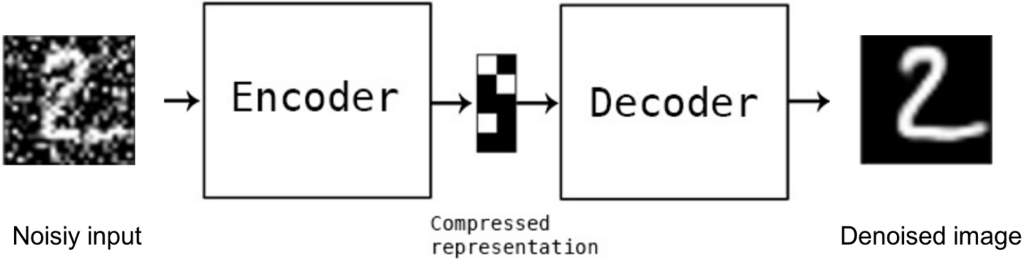
# Latent space



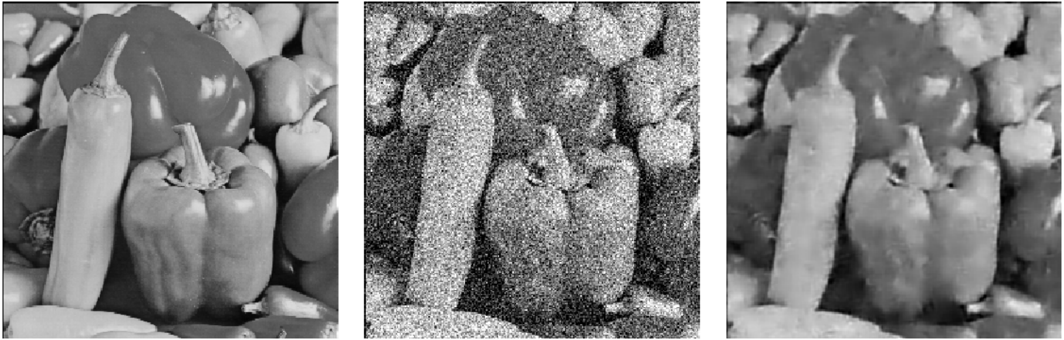
# Autoencoder applications

- Dimensionality reduction
- Denoising
- Outlier detection
- Search

# Denoising autoencoder



# Denoising autoencoder



# Overview

Types of Unsupervised Learning

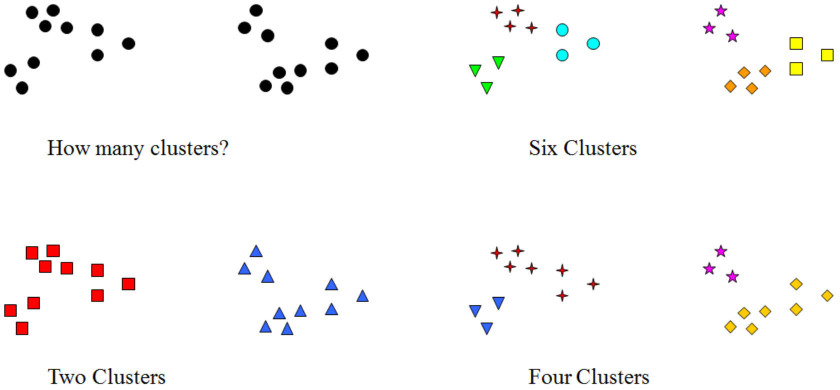
Dimensionality Reduction / Reprezentation Learning

Clustering

# Kinds of clusters

- Globular
- Well separated
- Dense
- Hierarchical
- Connected clusters
- ...

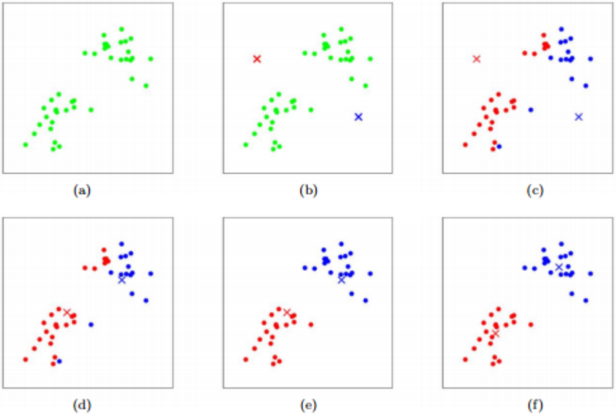
# Cluster granularity



# $K$ means

- Randomly initialize $K$ centroids by random sampling from the data
- Repeat until there is no change
  - Assign instances to nearest centroids to form clusters
  - Compute centroids as means of clusters

# $K$ means



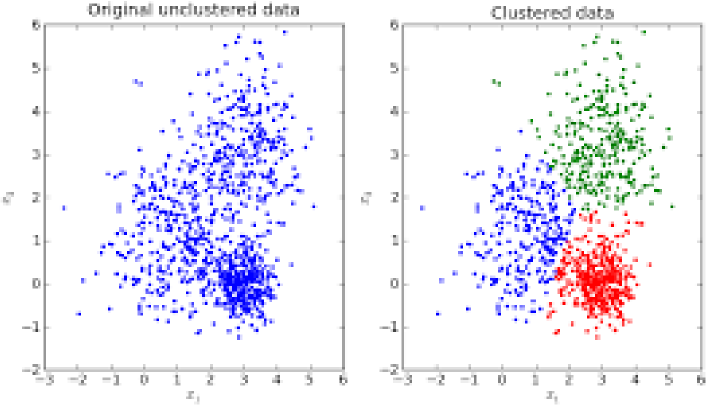
# $K$ means properties

- Guaranteed convergence
- Performs local minimization of

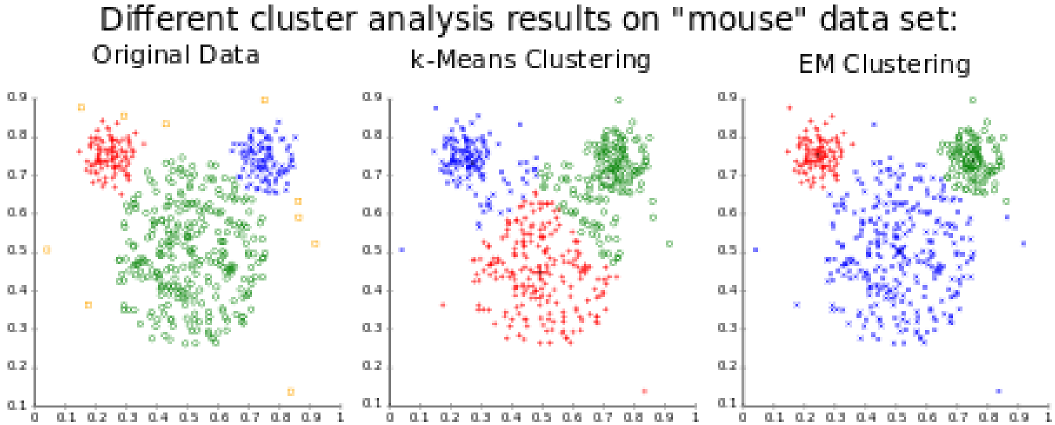
$$SSE(C_1, \ldots, C_K) = \sum_{i=1}^{K} \sum_{x \in C_i} \|x - \overline{x}_i\|_2^2$$

  over partitions $C_1, \ldots, C_K$ of the training set
- Prefers spherical clusters of similar volume and density
- Sensitive to outliers

# $K$ means illustrations



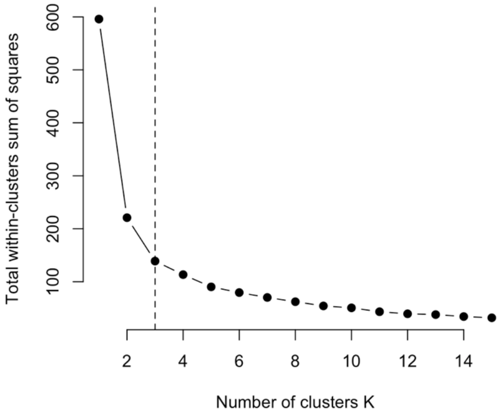
# $K$ means comparison



# How to select $K$?

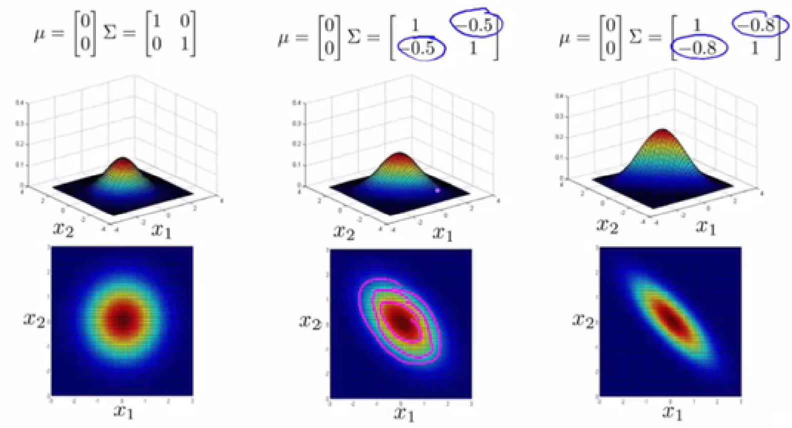
- There is no objective solution to the question of granularity!
- Still, some heuristics are often used in practice

# Elbow rule



Total within-clusters sum of squares
100
200
300
400
500
600
2
4
6
8
10
12
14
Number of clusters K

# Multivariate normal distirbution



# Limitations of $K$ means

Normal terms

- $K$ means is biased towards clusters which are
  - Spherical
  - Of similar radius
  - Of similar number of points
- Each point $x_i$ belongs strictly to one cluster $z_i$
- Clusters differ by their centroids

Probabilistic terms

- Clusters are distributed as $p(x|z=k) = \mathcal{N}(x; \mu_k, \Sigma_k)$
  - $\Sigma_k = \sigma_k I$
  - $\sigma_1 = \ldots = \sigma_K$
  - $p_k = 1/K$
- If $k = \arg\max_j p(z_i = j|x_i)$, then let $p(z_i = k|x_i) = 1$ and the rest be 0
- $\mu_k$ differ

# Generative model of the data for $K$ means

- Assume there is a stochastic mechanism which generated the data
- It first decides from which cluster to generate and then which point from that cluster to generate
- Then, probability density over point space is:

$$p(x) = \sum_{i=1}^{K} \frac{1}{K} p(x|z=i) = \sum_{i=1}^{K} \frac{1}{K} \mathcal{N}(x|\mu_i, \sigma^2 I)$$

- We don't really wan to generate the data, but to identify the mechanism which might have generated it and to obtain knowledge by inspecting such mechanism

# Can we generalize this?

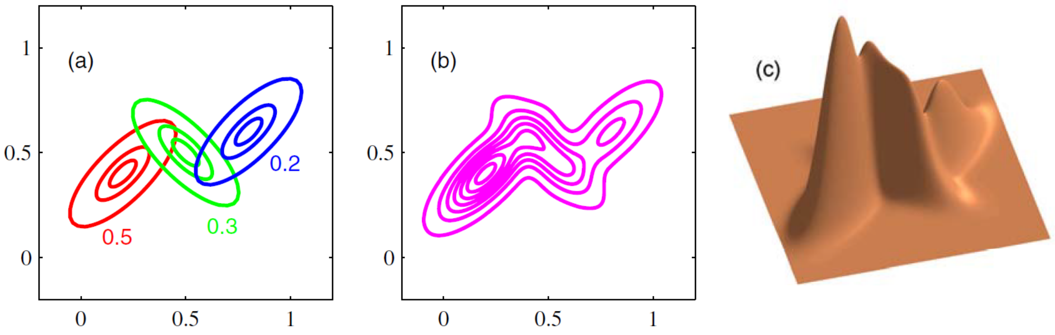
- Clusters may differ in cardinality – generalize $1/K$
- Clusters may differ in volume – generalize $\sigma^2$
- Clusters may differ in shape – generalize $I$
- Gaussian mixture model:

$$p(x) = \sum_{i=1}^{K} \pi_i \mathcal{N}(x|\mu_i, \Sigma_i)$$

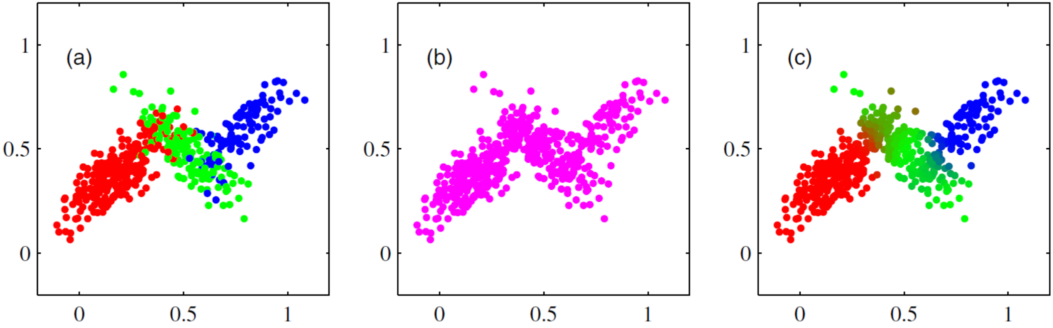
$$\sum_{i=1}^{K} \pi_i = 1$$

$$\pi_i \geq 0$$

# Gaussian mixture model



# Clustering by GMM



# How to learn model parameters?

- If we knew model parameters it would be easy to identify the clusters for instance $x_i$:

$$
\begin{aligned}
p(z = k|x_i) &= \frac{p(x_i|z = k)p(z = k)}{p(x_i)} \\
&= \frac{p(x_i|z = k)p(z = k)}{\sum_{k=1}^{K} p(x_i|z = k)p(z = k)} \\
&= \frac{\pi_k \mathcal{N}(x_i; \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)} \\
&\triangleq r_{ik}
\end{aligned}
$$

# How to learn model parameters?

- If we knew distribution over clusters for each instance, it would be easy to estimate the parameters:

$$\pi_k = \frac{1}{N} \sum_{i=1}^{N} r_{ik}$$

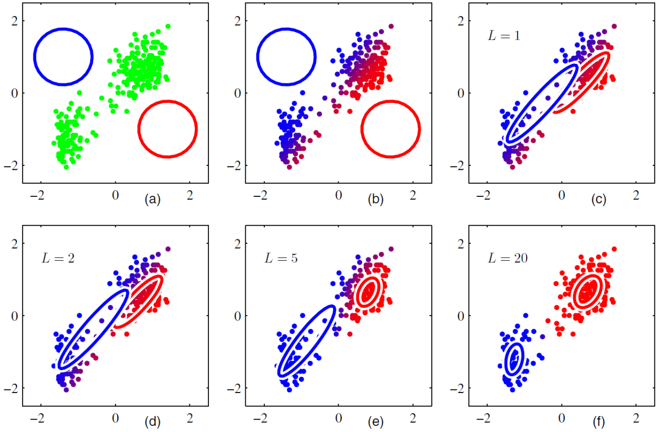
$$\mu_k = \frac{\sum_{i=1}^{N} r_{ik} x_i}{\sum_{i=1}^{N} r_{ik}}$$

$$\Sigma_k = \frac{\sum_{i=1}^{N} r_{ik} (x_i - \mu_k)^T (x_i - \mu_k)}{\sum_{i=1}^{N} r_{ik}}$$

# How to learn model parameters?

- Start with randomly initialized parameters
- Iterate cluster identification and parameter estimation
- This is an instance of much more general EM algorithm

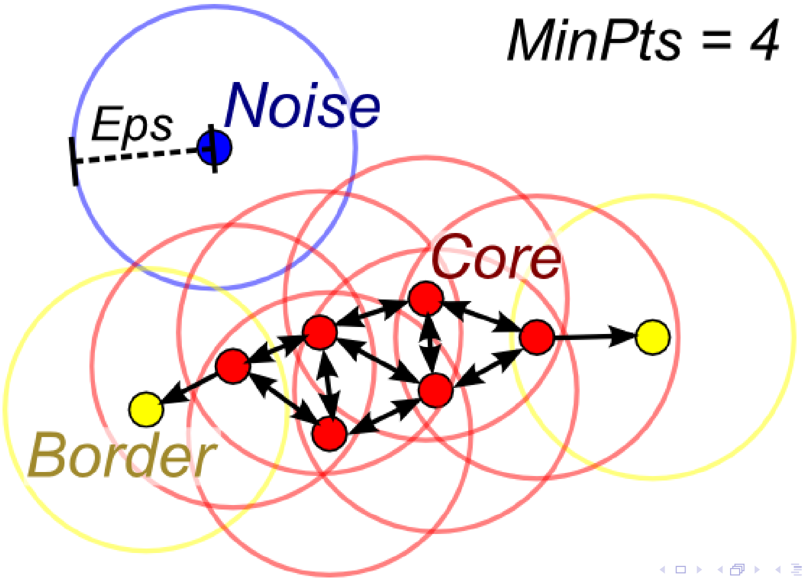
# GMM clustering process



# DBSCAN

- Extract *core points* which have at least *MinPts* points in its $\varepsilon$ neighbourhood
- Connect to them all points in their $\varepsilon$ neighbourhood and form a graph
- Return its connected components as clusters

# DBSCAN



# DBSCAN properties

- Prefers dense cluster
- Unsuitable in case of clusters of varying density
- Arbitrary cluster shapes
- Discards some instances as noise

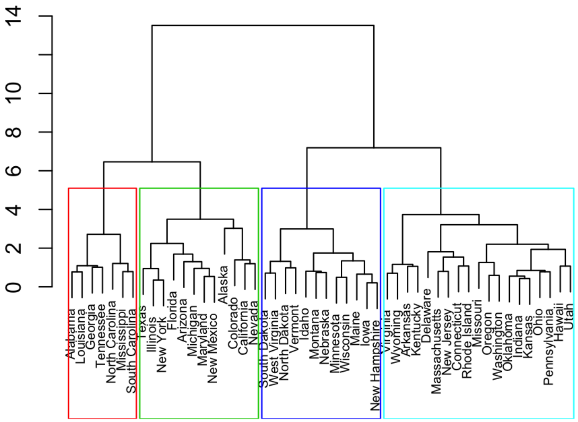
# Agglomerative clustering

- Initialize clusters to single instances
- Repeat until single cluster is left
  - Compute distances between all pairs of clusters
  - Merge two nearest clusters

# Cluster distance

- Minimum of distances of elements from each cluster – arbitrary shapes, but sensitive to noise
- Maximum – globular clusters
- Average – compromise

# Dendrogram



0
2
4
6
8
10
14
Alabama
Louisiana
Georgia
Tennessee
North Carolina
Mississippi
South Carolina
Texas
Illinois
New York
Florida
Arizona
Michigan
Maryland
New Mexico
Alaska
Colorado
California
Nevada
South Dakota
West Virginia
North Dakota
Vermont
Idaho
Montana
Nebraska
Minnesota
Wisconsin
Maine
Iowa
New Hampshire
Virginia
Wyoming
Arkansas
Kentucky
Delaware
Massachusetts
New Jersey
Connecticut
Rhode Island
Missouri
Oregon
Washington
Oklahoma
Indiana
Kansas
Ohio
Pennsylvania
Hawaii
Utah

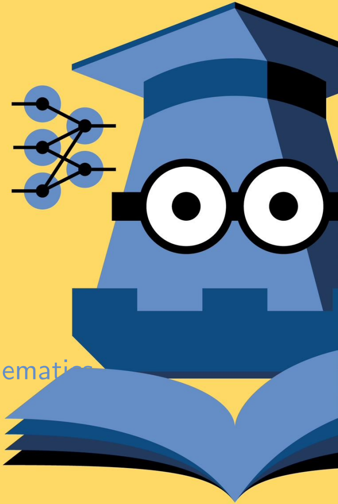
# Agglomerative clustering properties

- Cluster distance dependent
- Provide full clustering information
- Can provide required number of clusters afterwards

# Advanced topics

- Generative adversarial networks
- Variational autoencoders
- Self-supervised learning
- ...

# THANK YOU



Mladen Nikolić
nikolic@math.rs
Machine Learning and Applications Group at the Faculty of Mathemati
machinelearning.math.rs