Pattern recognition

Classification/Clustering

GW – Chapter 12 (some concepts)

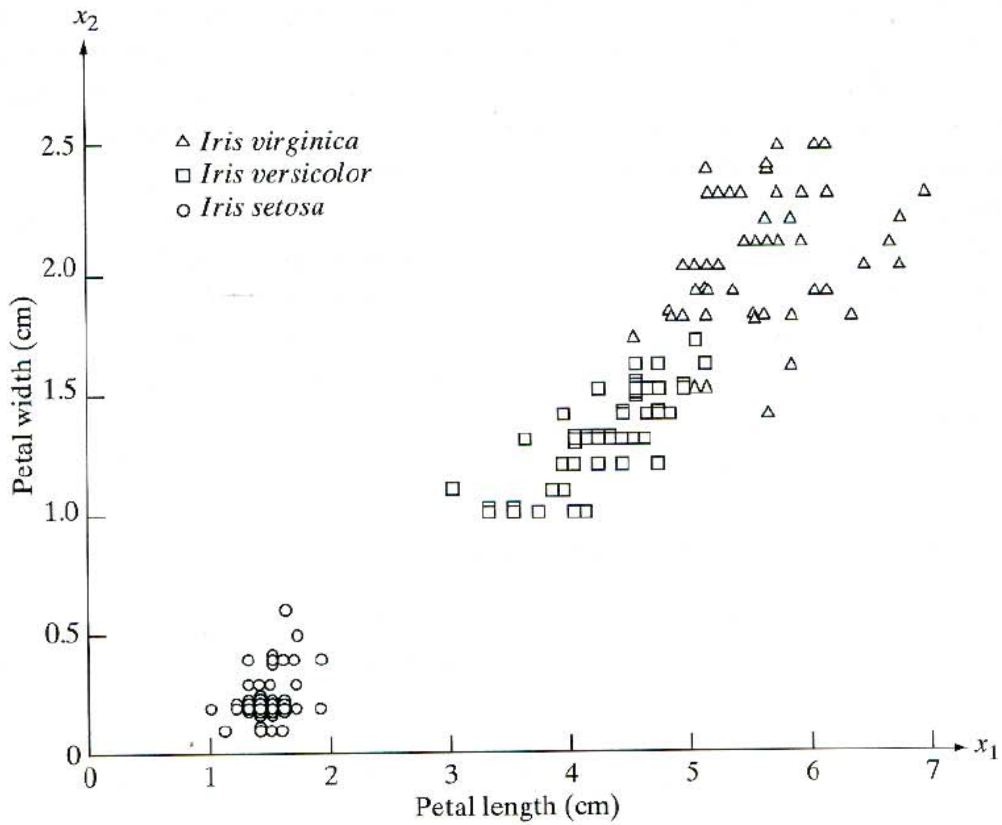
Textures

Patterns and pattern classes

- Pattern: arrangement of descriptors
- Descriptors: features
- Patten class: family of patterns that share common properties denoted by ω_1, ω_2, ω_3, …, ω_n
- Pattern recognition: assigning patterns to the respective classes
- Patterns are represented by vectors of descriptors called feature vectors
- $FV=[x_1,x_2,\ldots,x_n]^T$
- The nature of the features depends on the problem

Patterns and pattern classes

- Task: distinguish flowers based on the length and width of petals



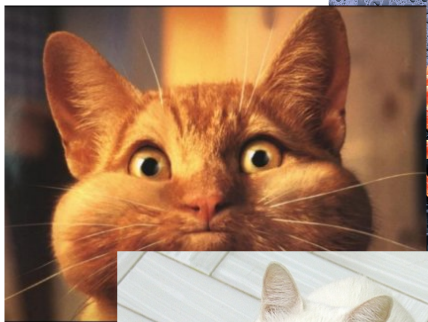
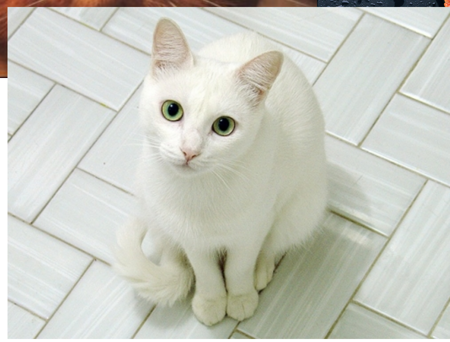
Feature selection problem

- Identify the features that lead to the *maximum separ*ability among classes
- Features vary both within and between classes. Classes are separable if istances of the same classes have features that are more similar among them then to the features of objects belonging to different classes. In this case, a boundary can be identified in the feature space separating objects belonging to different classes
- Goal of feature selection: minimize variability within classes and maximize separation among classes

Choose the good features!






Color

Number of hears

Choice of the features

- First statistical moments could be used as features
- Implies the use of a neighborhood which limits the resolution
- Typical for region-based processing
  - Texture features

PR based on decision theoretic methods

- Based on the use of a discriminant function
- Let x be a FV and $\omega_1, \omega_2, \omega_3, \ldots, \omega_n$ be n pattern classes. The basic problem consists in finding n decision functions di(x) with the property that if pattern x belongs to the class ωi then

$$d_i(x) > d_j(x) \quad i, j = 1, \ldots, n \quad i \neq j$$

- The decision boundary separating ω_i from ω_j is given by the values of x for which

$$d_i(x) = d_j(x)$$

PR based on decision theoretic methods

- Common practice consists in identifying decision boundaries by the single function

$$d_{i,j}(x) = d_i(x) - d_j(x)$$

- Thus

$$d_{ij}(x) > 0$$ For objects belonging to ωi

$$d_{ij}(x) < 0$$ For objects belonging to ωj

Classification

- Problem statement
  - Given a set of classes $\{\omega_i,\ i=1,\ldots N\}$ and a set of observations $\{\mathbf{x}_{i,k}, k=1,\ldots M\}$ determine the most probable class, given the observations. This is the class that maximizes the conditional probability:

$$\omega_{winner} = \max_k P(\omega_i | x_k)$$

Clustering and Classification

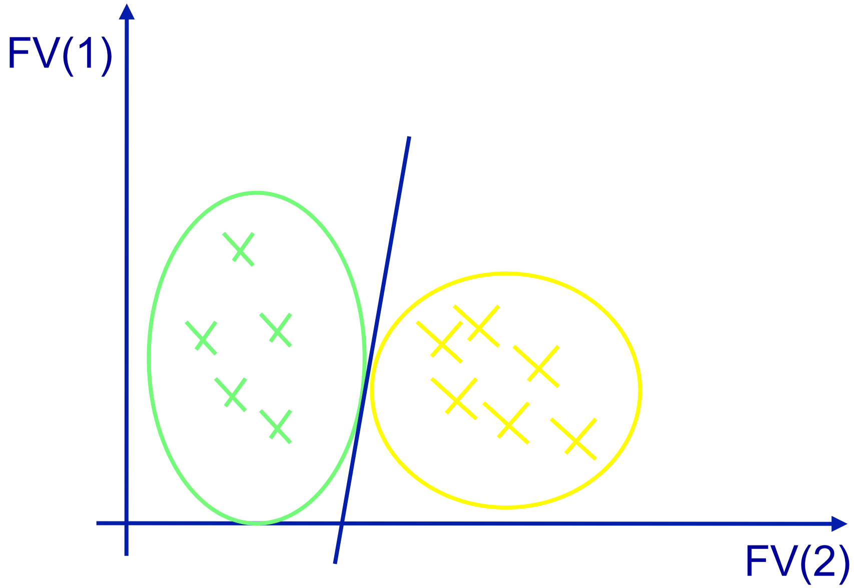
- Clustering
  - Putting together (aggregating) feature vectors based on a minimum distance criterion
  - Self-contained: no need to refer to other images or data samples
- Classification
  - Identification of the class a given feature vector belongs to based on a minimum distance criterion and based on a set of available "examples"
  - Uses a reference database of example images that identify the different classes

Hypothesis: the classes (textures) are separated in the feature space

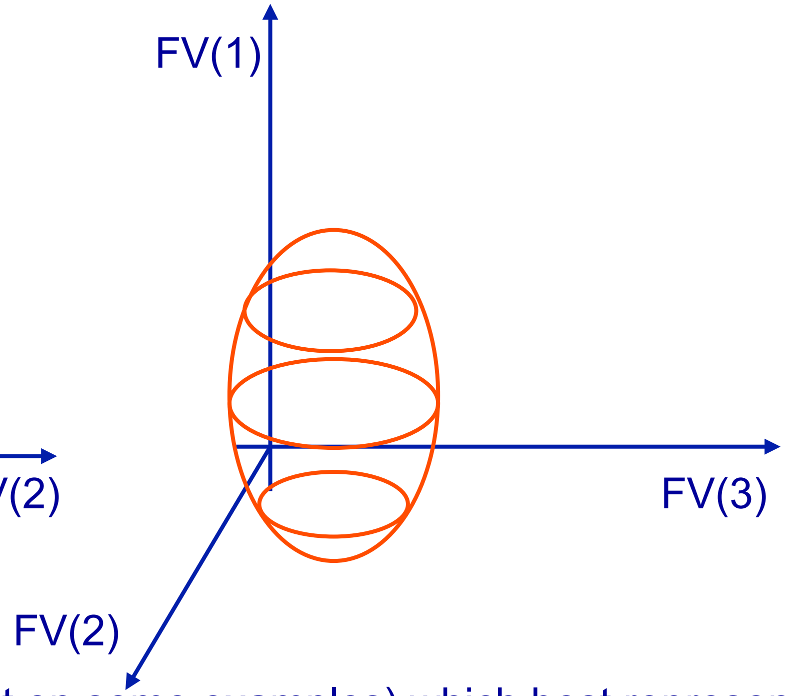
Both result in a segmentation map associating one class to each pixel

Condition: Separability in the Feature Space

Bi-dimensional feature space (FV of size 2)



Multi-dimensional feature space



Clustering: building the clusters

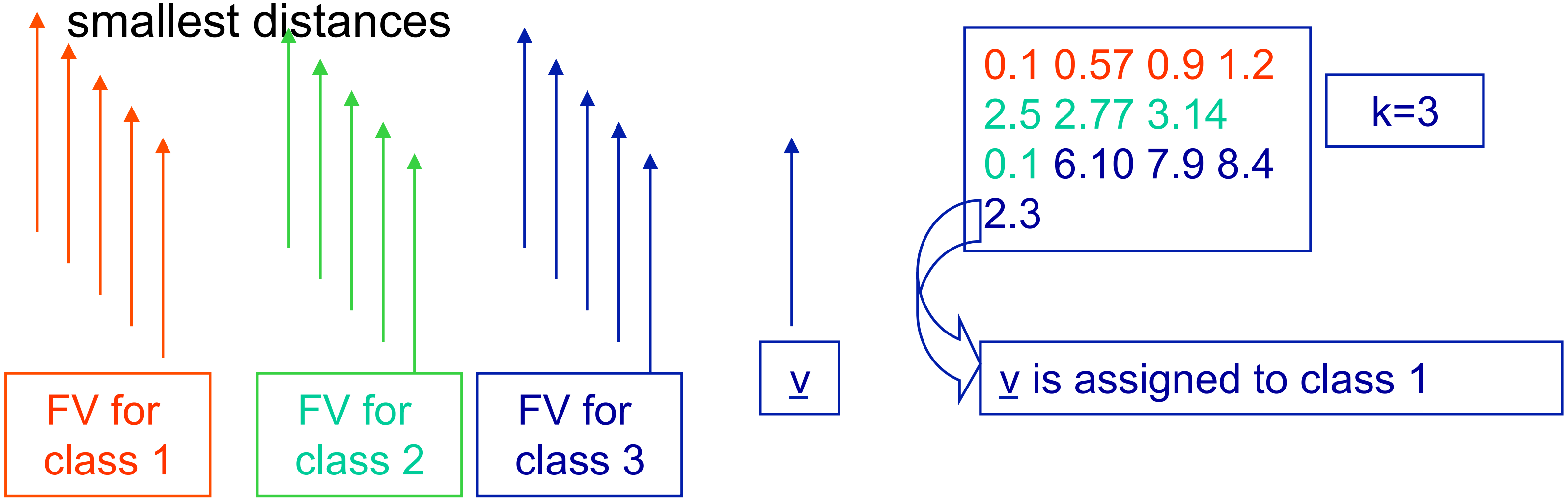
Classification: identification of the cluster (built on some examples) which best represents the vector according to the chosen distance measure

Types of PR algorithms

- Measuring the distance among a *class* and a *vector*
  - Each class (set of vectors) is represented by the mean (m) vector and the vector of the variances (s) of its components ⇒ the training set is used to build m and s
  - The distance is taken between the test vector and the m vector of each class
  - The test vector is assigned to the class to which it is closest
    - Euclidean classifier
    - Weighted Euclidean classifier
  - Example: k-means clustering
- Measuring the distance among of vectors
  - One vector belongs to the training set and the other is the one we are testing
  - Example: kNN classification

kNN

- Given a vector v of the test set
  - Take the distance between the vector v and ALL the vectors of the training set
  - (while calculating) keep the k smallest distances and keep track of the class they correspond to
  - Assign v to the class which is most represented in the set of the k smallest distances



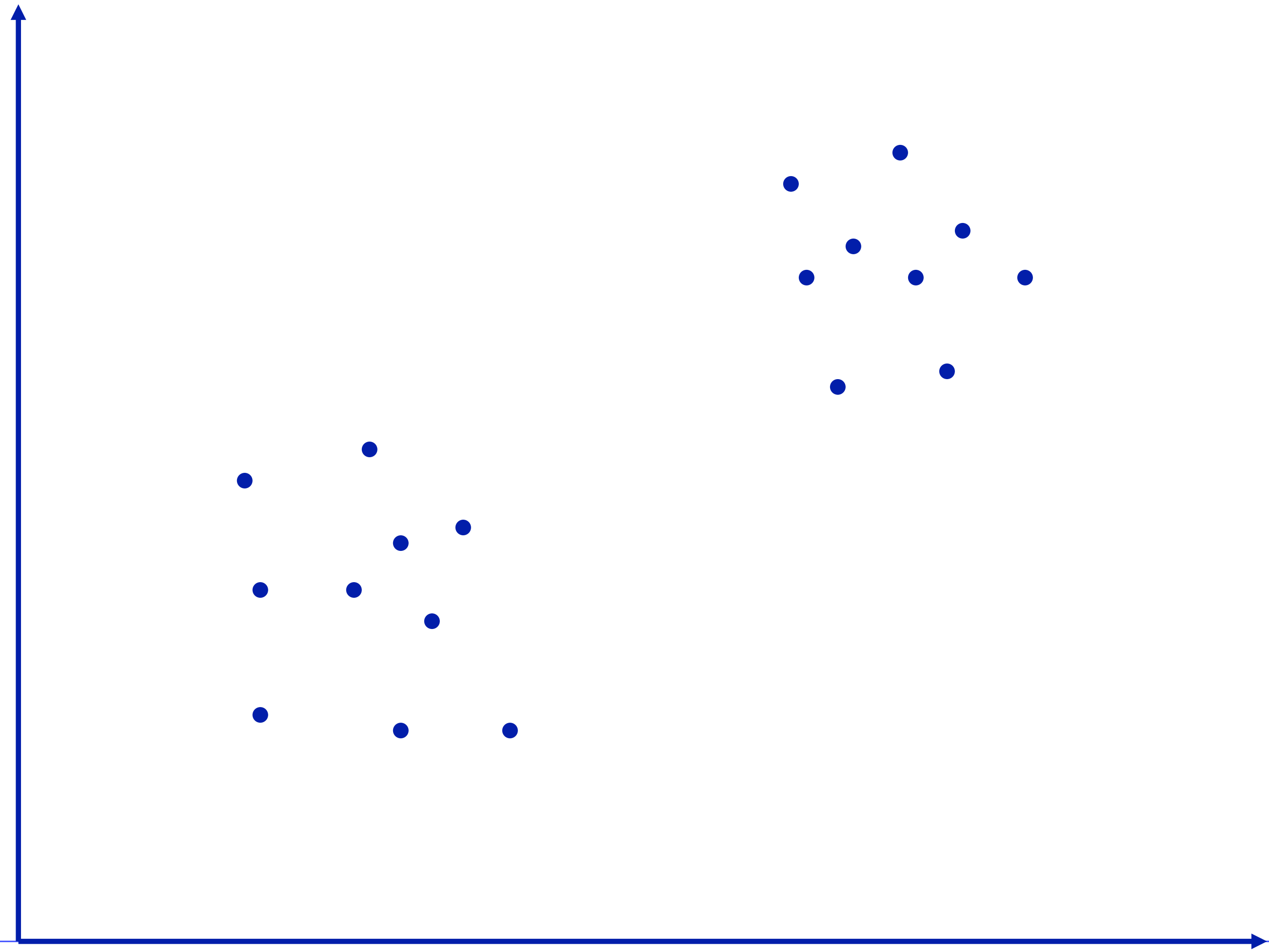
Confusion matrix

textures	1	2	3	4	5	6	7	8	9	10	% correct
1	841	0	0	0	0	0	0	0	0	0	100.00%
2	0	840	1	0	0	0	0	0	0	0	99.88%
3	2	0	839	0	0	0	0	0	0	0	99.76%
4	0	0	0	841	0	0	0	0	0	0	100.00%
5	0	0	88	0	753	0	0	0	0	0	89.54%
6	0	0	134	0	0	707	0	0	0	0	84.07%
7	0	66	284	0	0	0	491	0	0	0	58.38%
8	0	0	58	0	0	0	0	783	0	0	93.10%
9	0	0	71	0	0	0	0	0	770	0	91.56%
10	0	4	4	0	0	0	0	0	0	833	99.05%
				Average recognition rate							91.53%

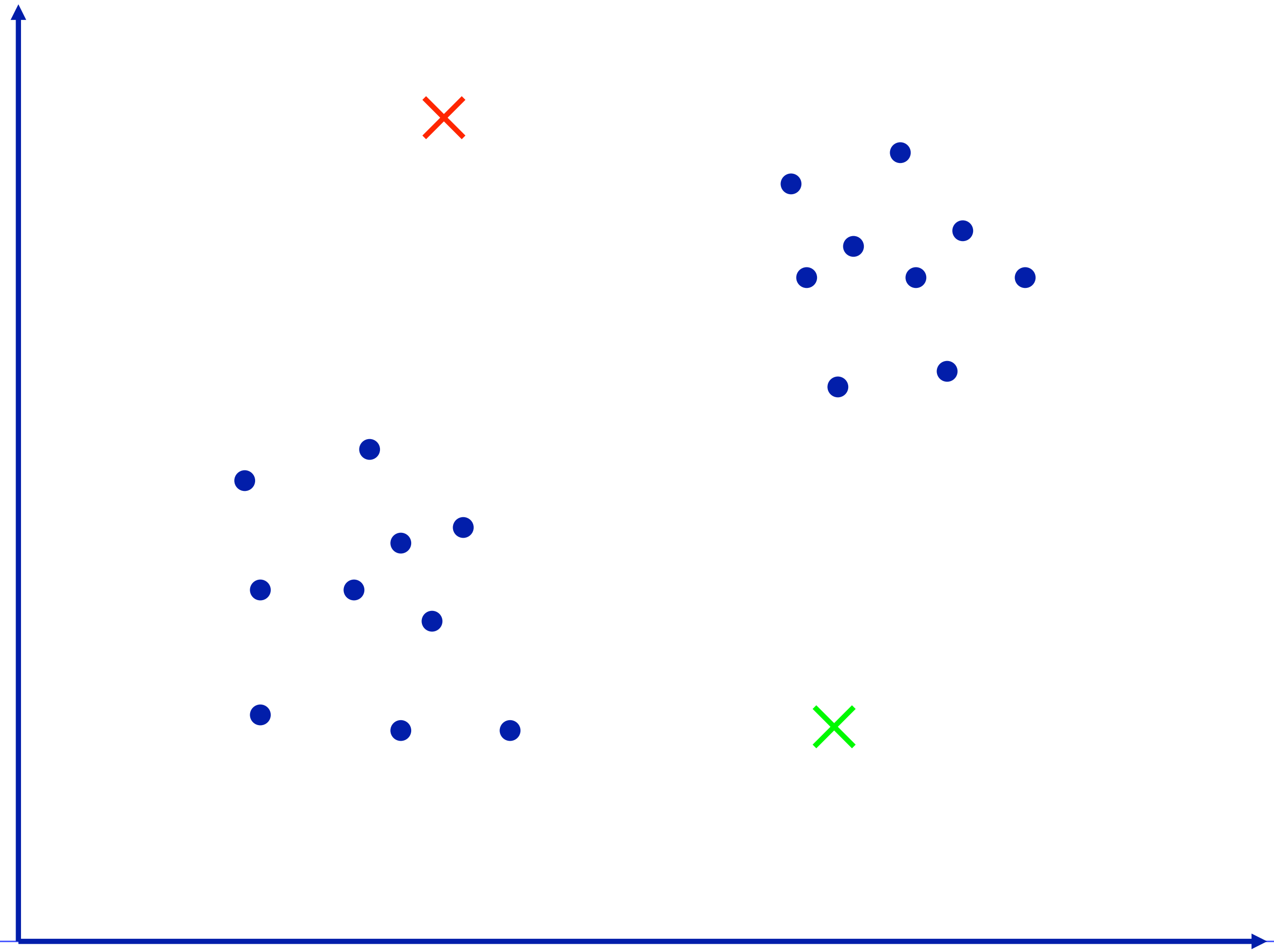
K-Means Clustering

1. Partition the data points into K clusters randomly. Find the centroids of each cluster.
2. For each data point:
   - Calculate the distance from the data point to each cluster.
   - Assign the data point to the closest cluster.
3. Recompute the centroid of each cluster.
4. Repeat steps 2 and 3 until there is no further change in the assignment of data points (or in the centroids).

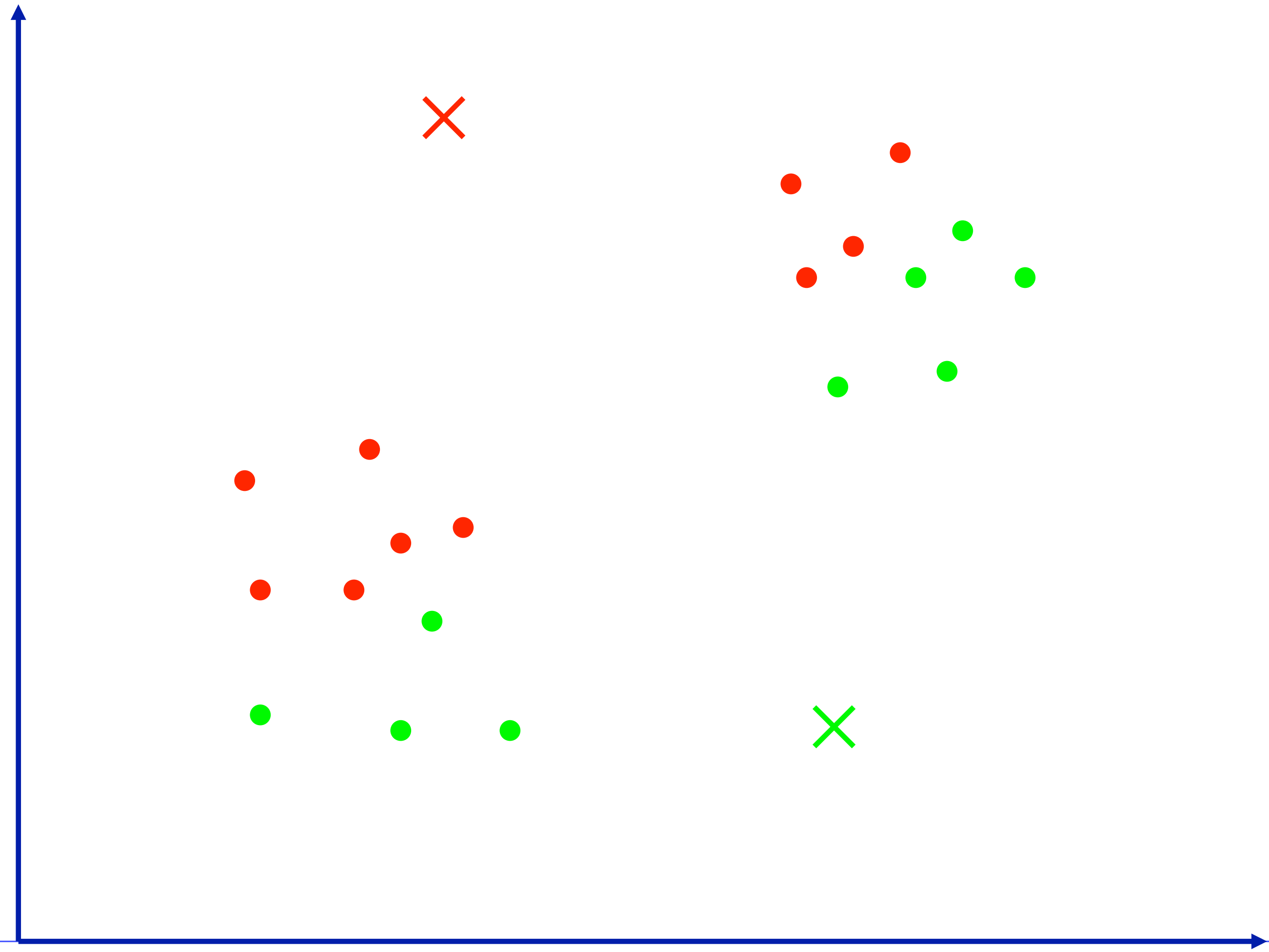
K-Means Clustering



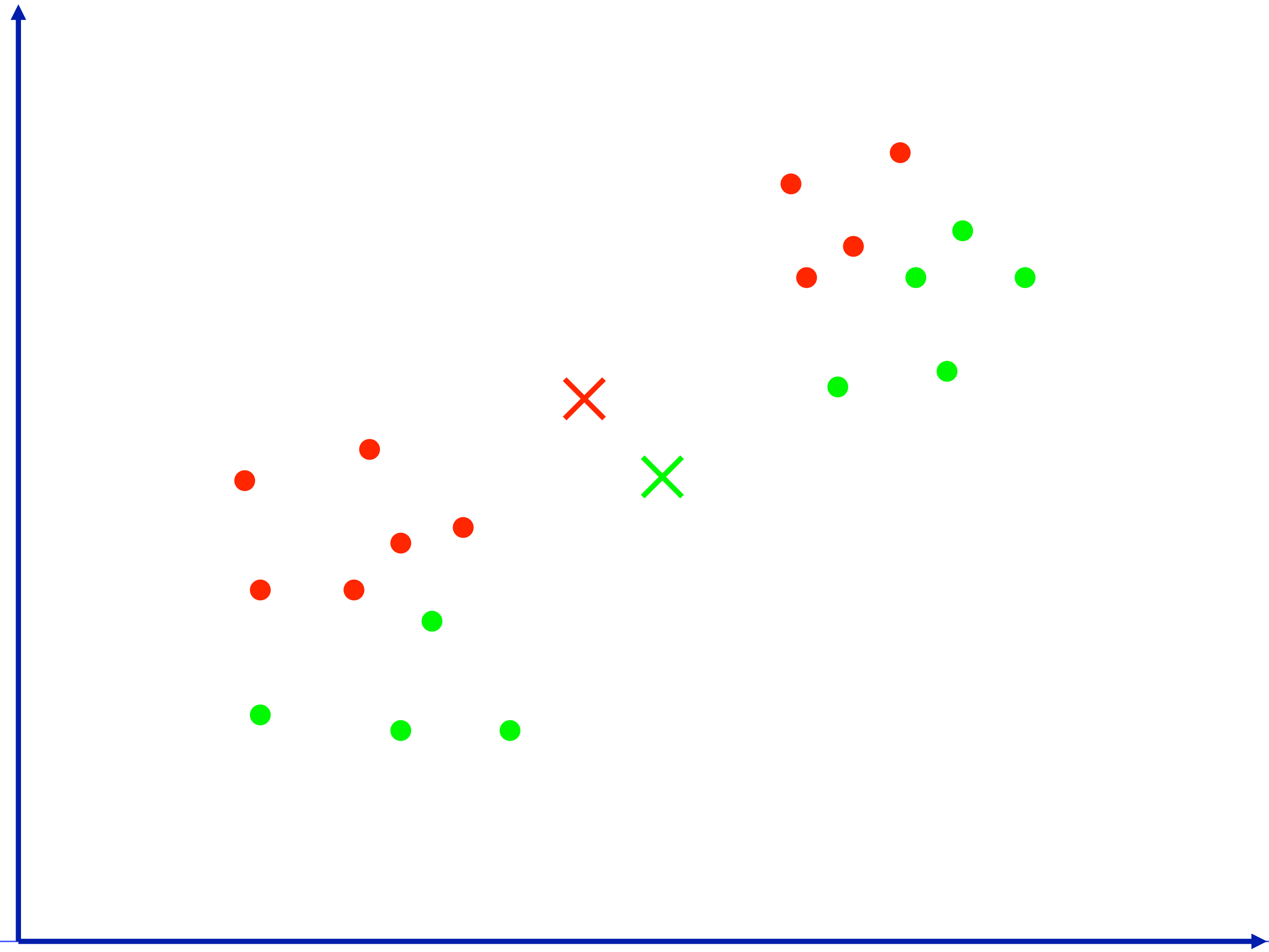
K-Means Clustering



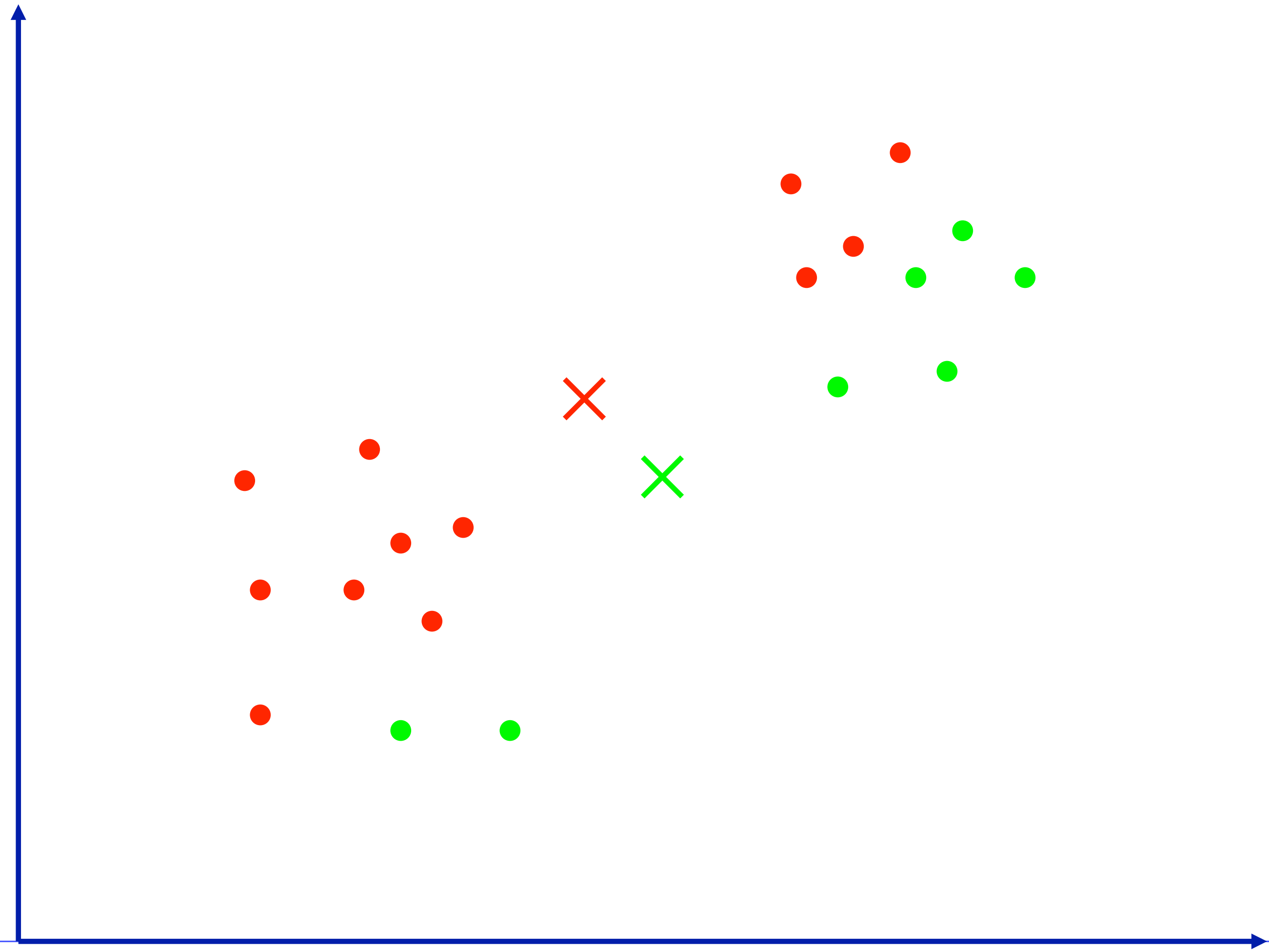
K-Means Clustering



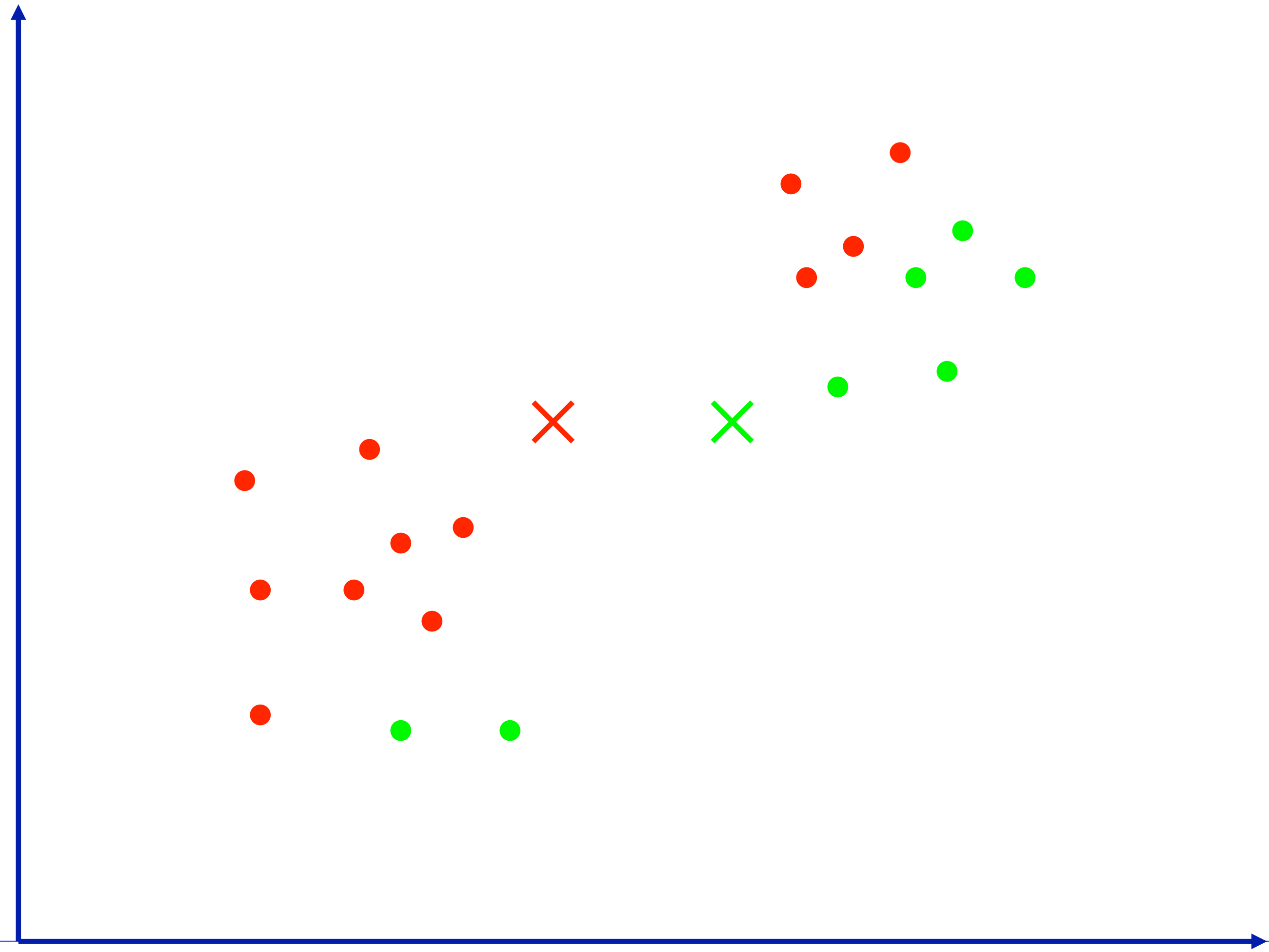
K-Means Clustering



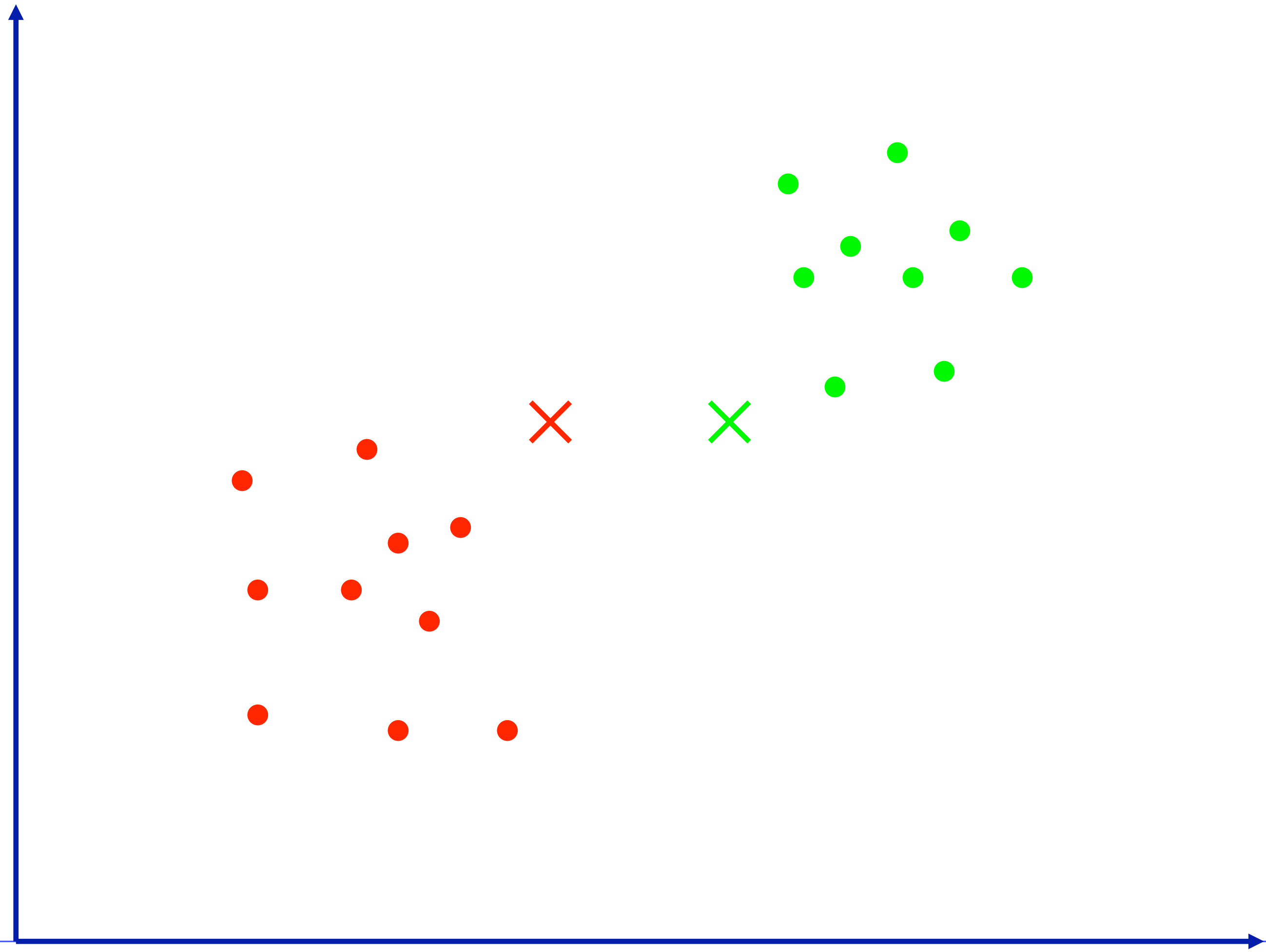
K-Means Clustering



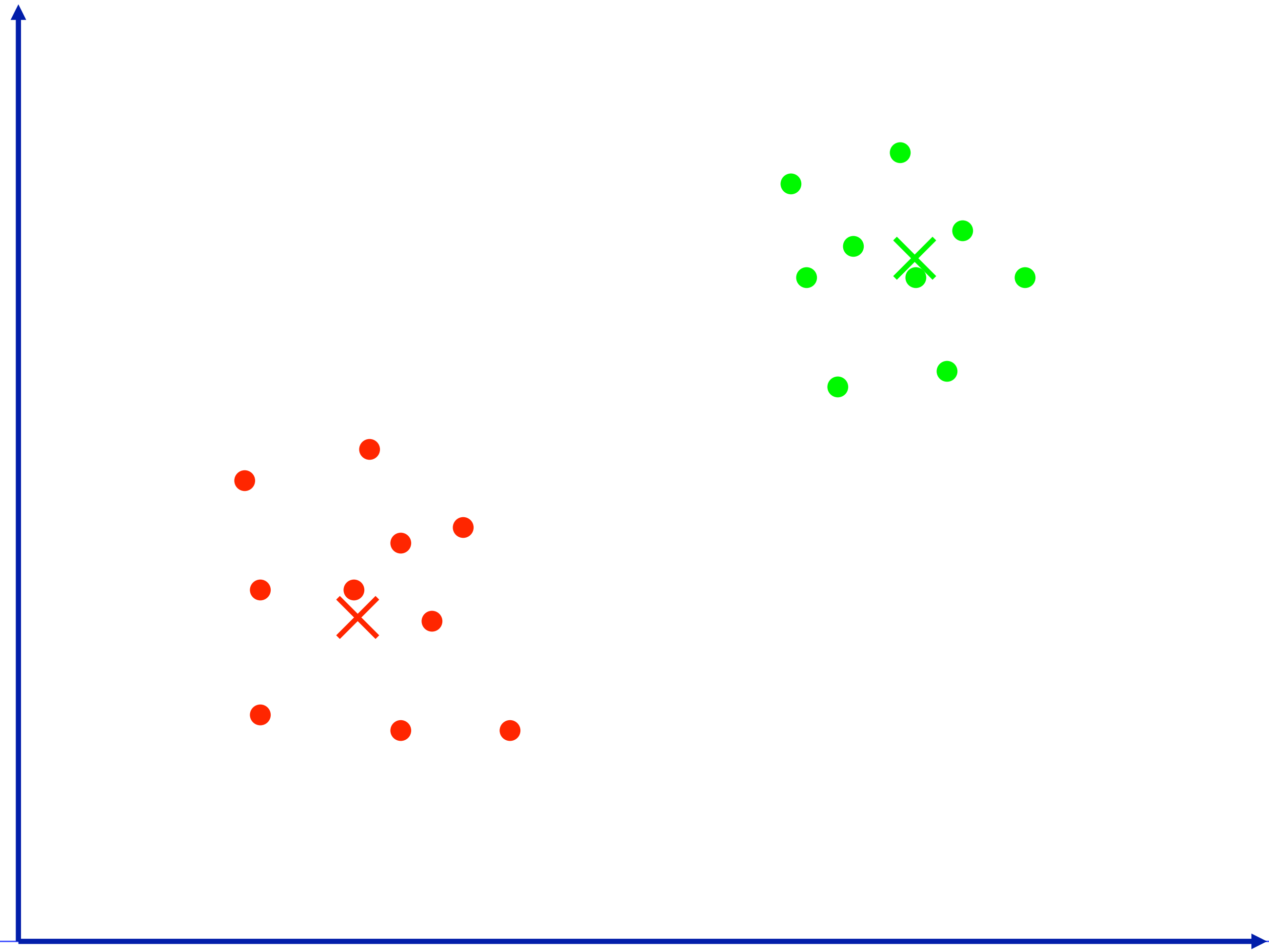
K-Means Clustering



K-Means Clustering

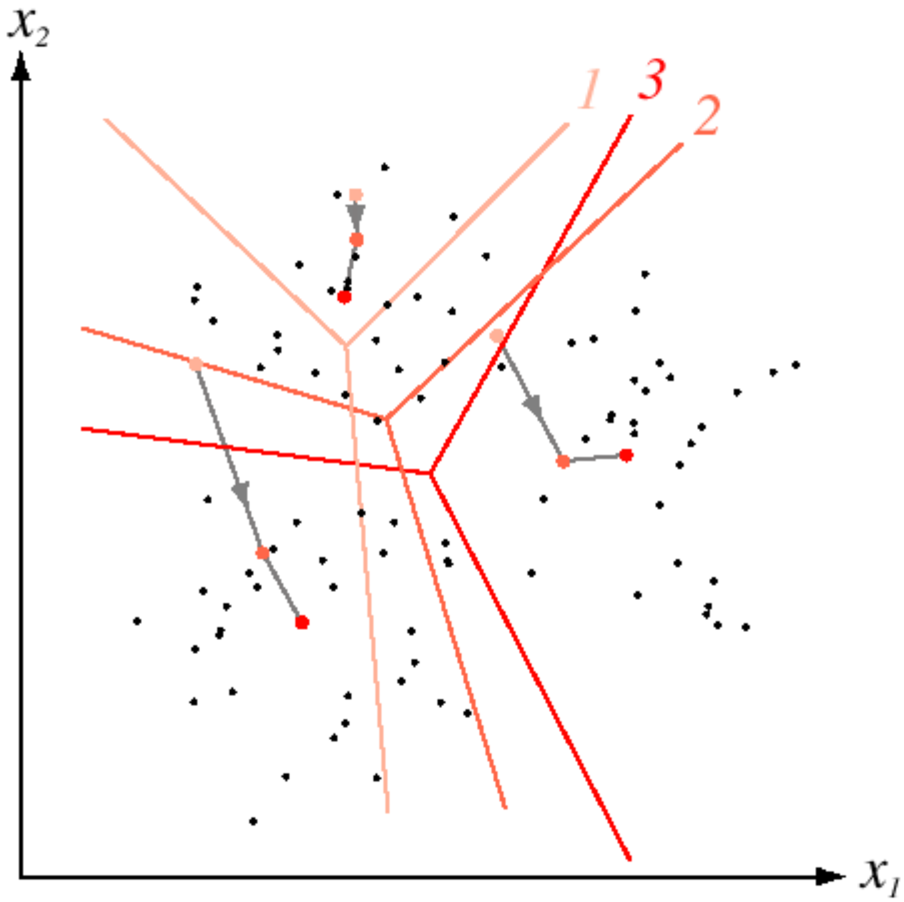


K-Means Clustering



K-Means Clustering

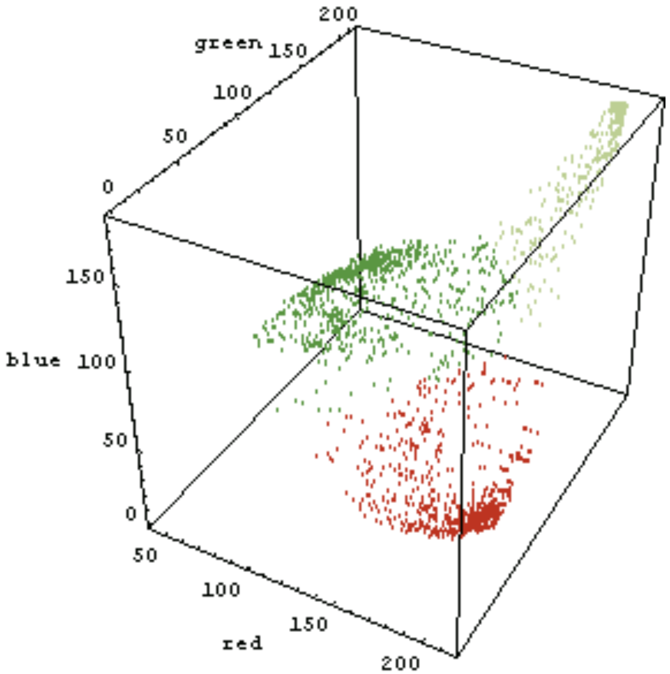
- Example



Duda et al.

K-Means Clustering

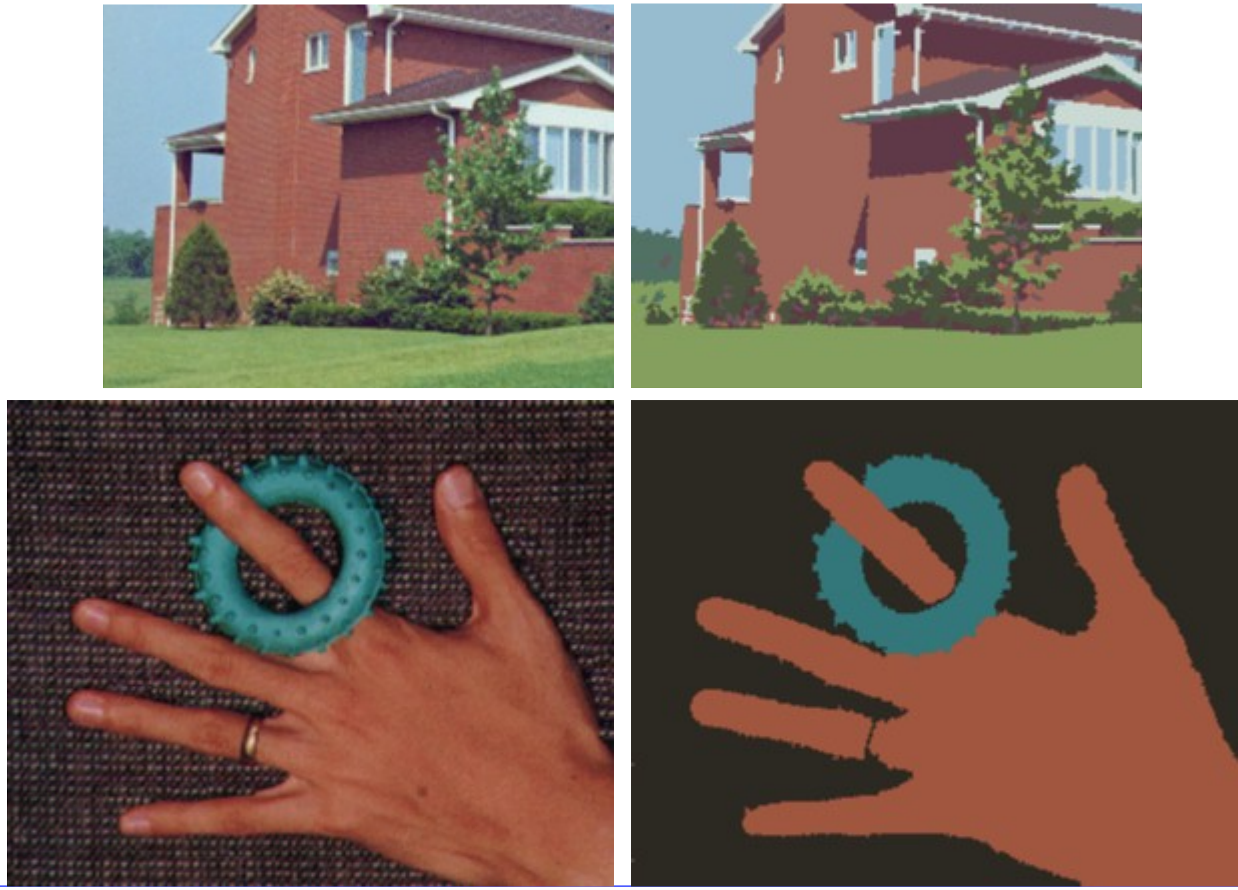
- RGB vector



K-means clustering minimizes $$\sum_{i \in \text{clusters}} \left\{ \sum_{j \in \text{elements of i'th cluster}} \left\| x_j - \mu_i \right\|^2 \right\}$$

Clustering

- Example



D. Comaniciu and P. Meer, *Robust Analysis of Feature Spaces: Color Image Segmentation*, 1997.

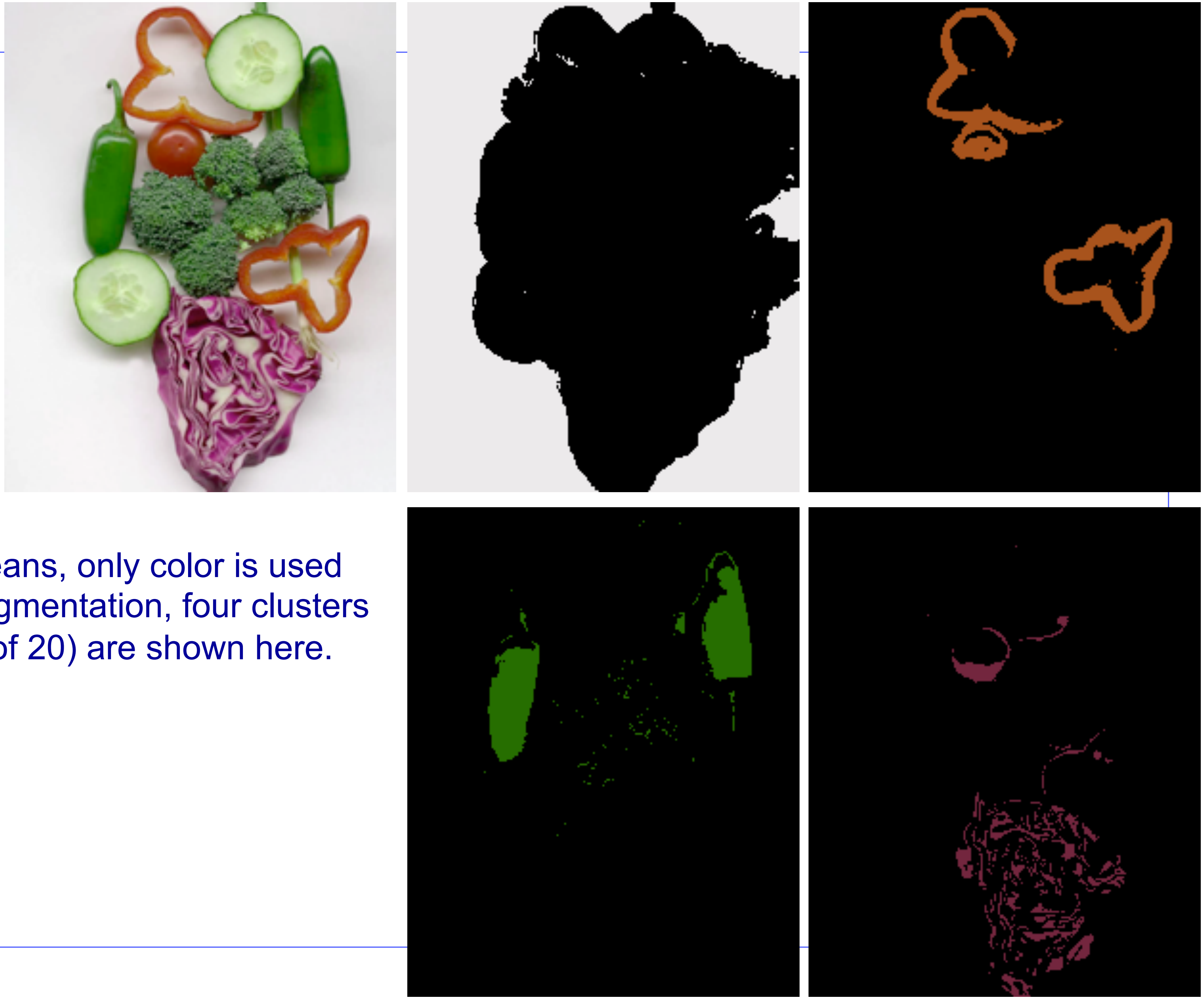
K-Means Clustering

- Example

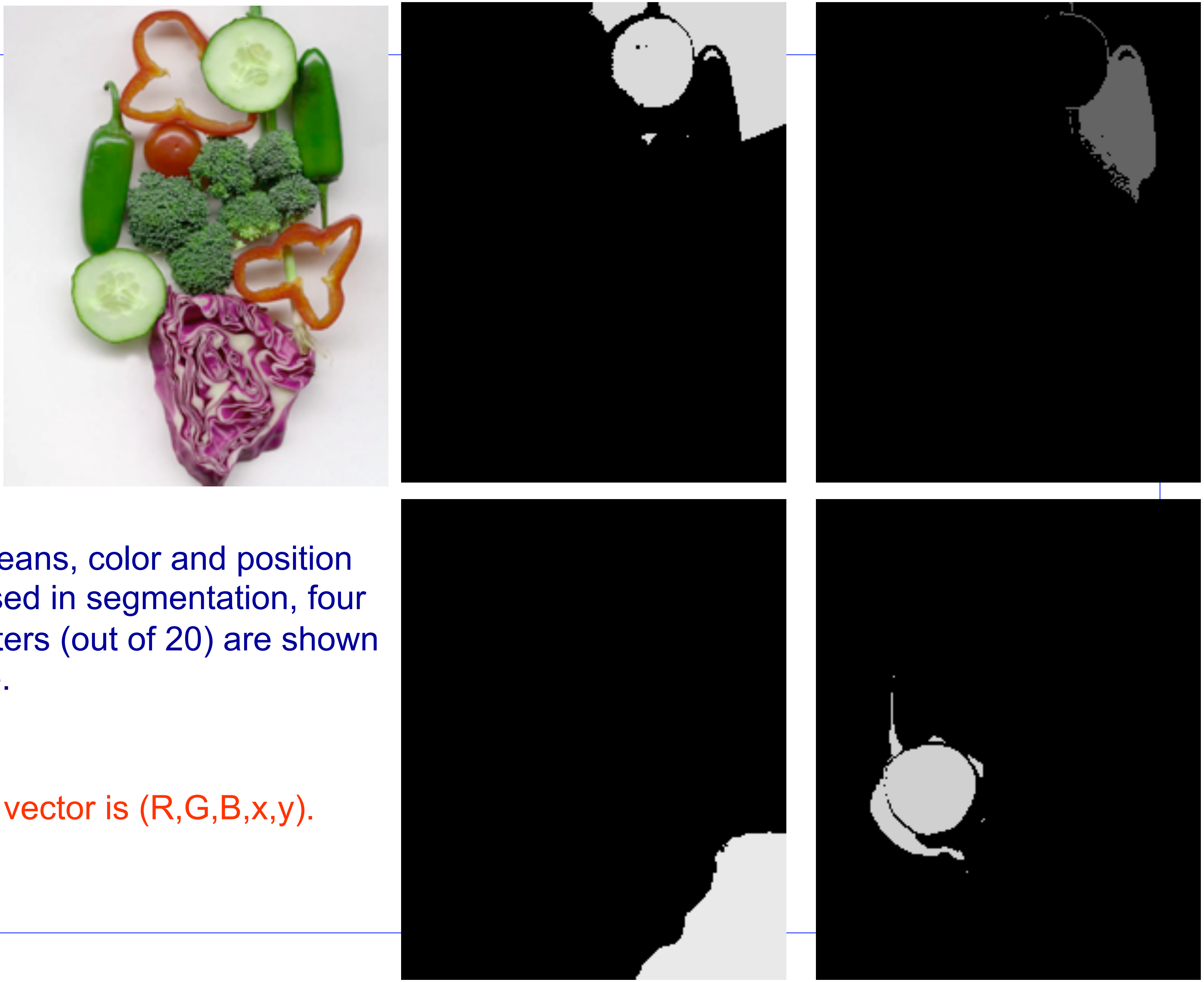
Original

K=5

K=11



K-means, only color is used in segmentation, four clusters (out of 20) are shown here.



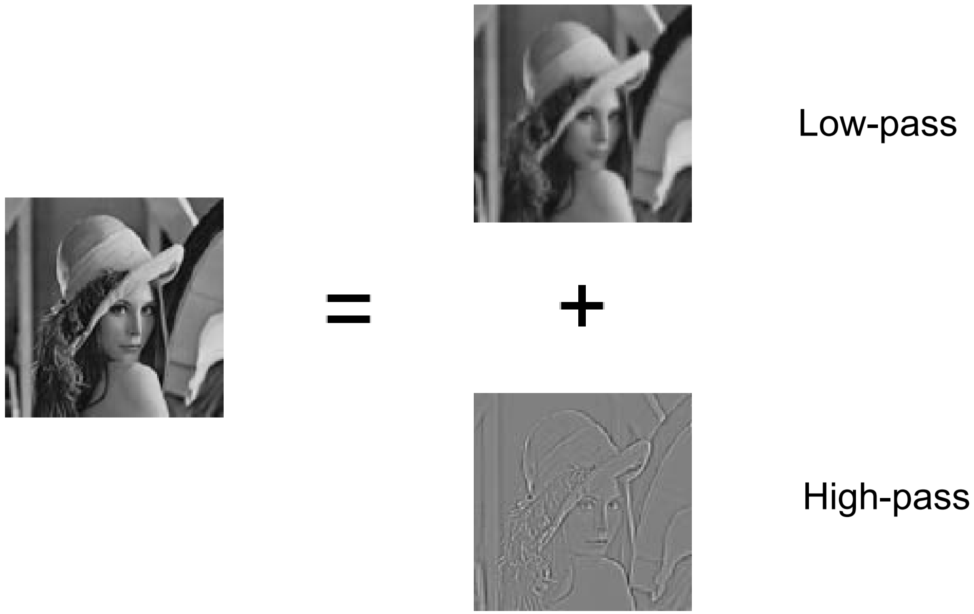
K-means, color and position is used in segmentation, four clusters (out of 20) are shown here.

Each vector is (R,G,B,x,y).

K-Means Clustering: Axis Scaling

- Features of different types may have different scales.
  - For example, pixel coordinates on a 100x100 image vs. RGB color values in the range [0,1].
- Problem: Features with larger scales dominate clustering.
- Solution: Scale the features.

Image pyramids: concept



Multiresolution signal analysis
(computer vision)

Image pyramids

Idea: Represent NxN image as a "pyramid" of 1×1, 2×2, 4×4,…, $2^k \times 2^k$ images (assuming $N=2^k$)

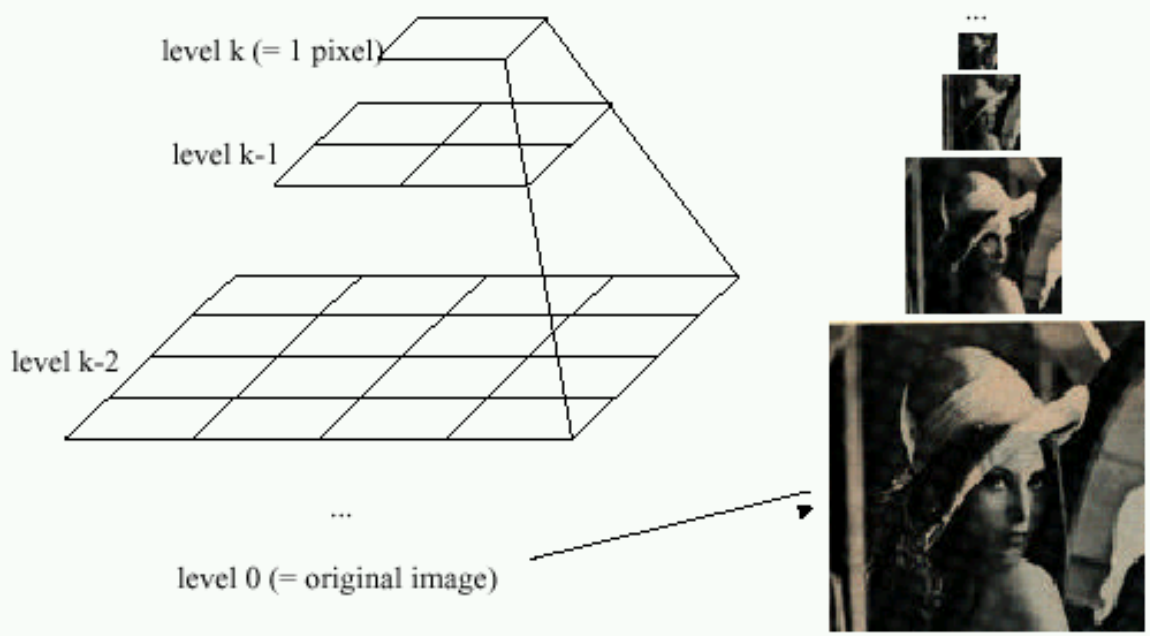


Image pyramid: feature extraction

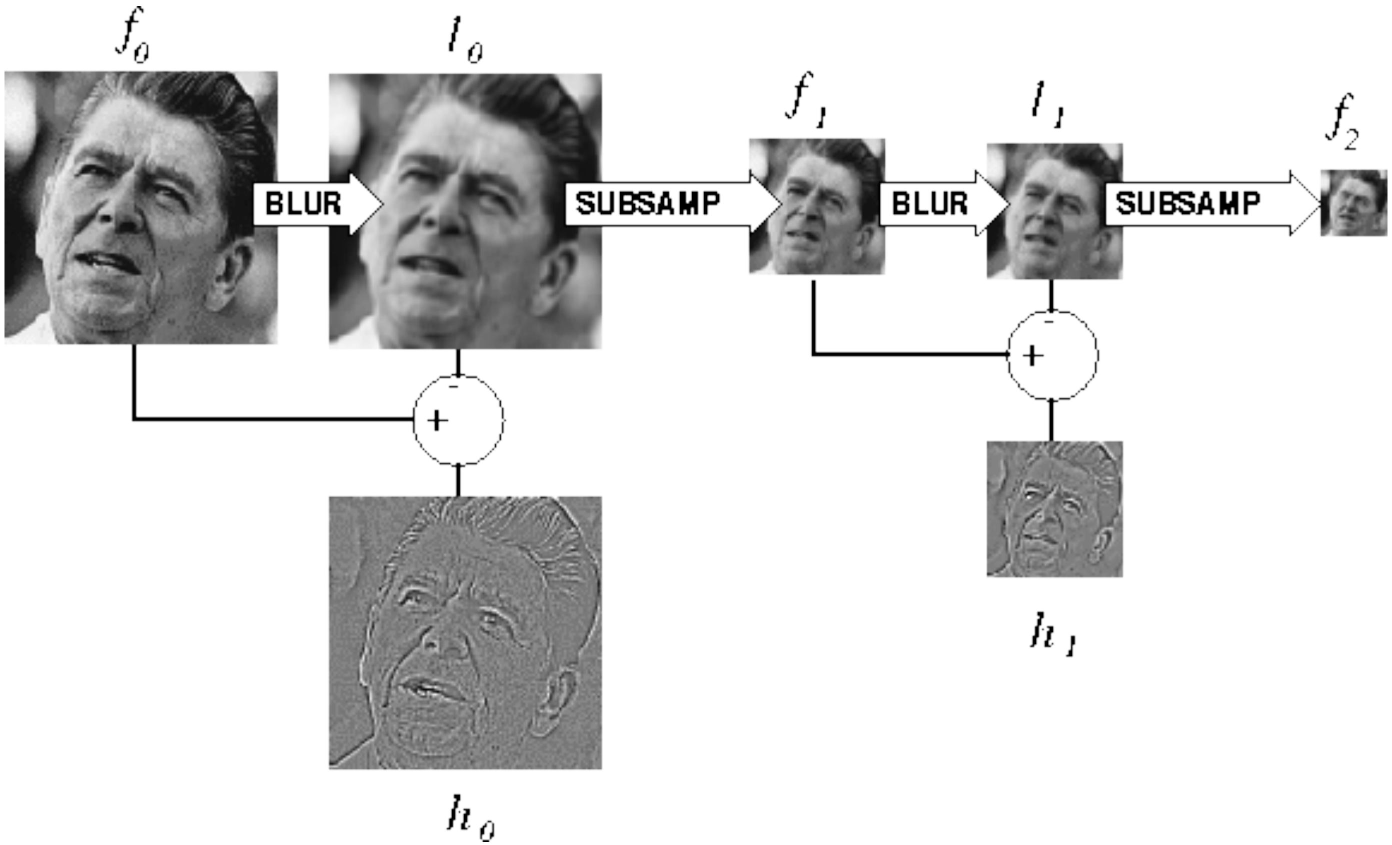
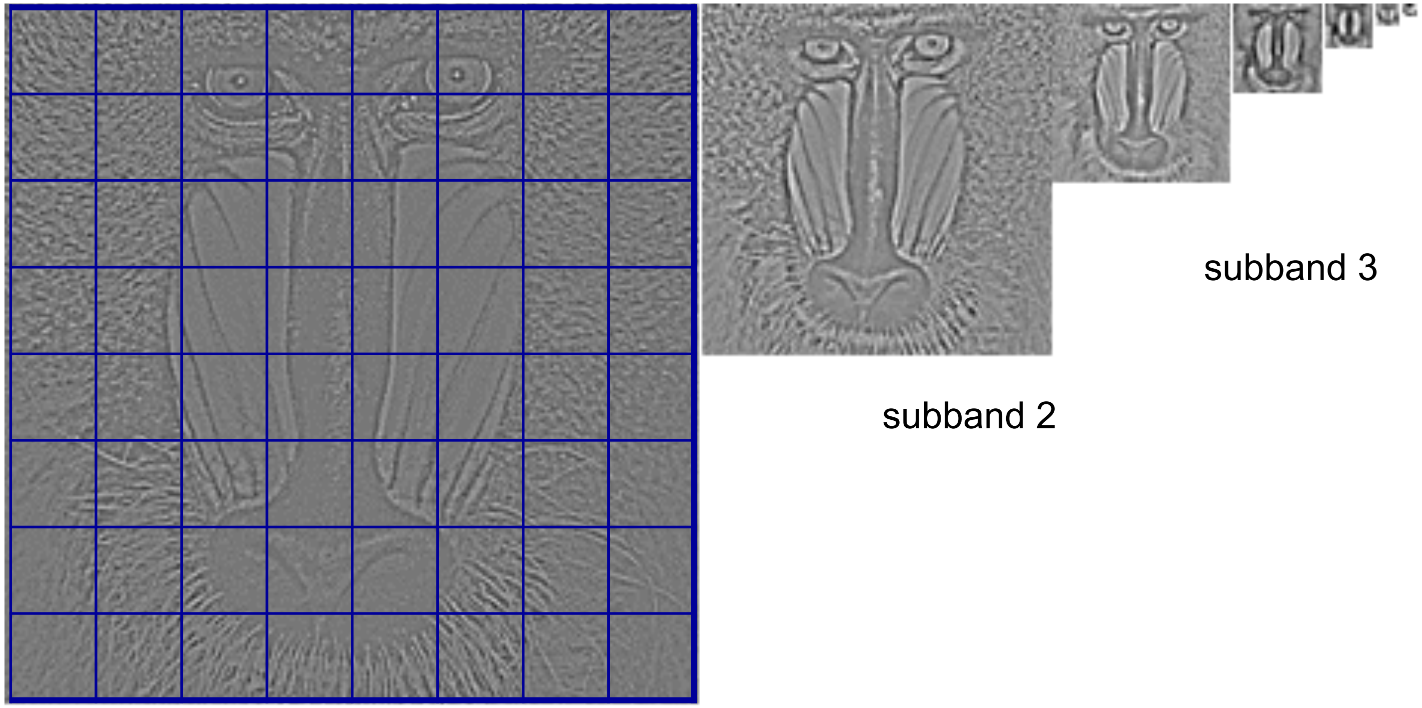
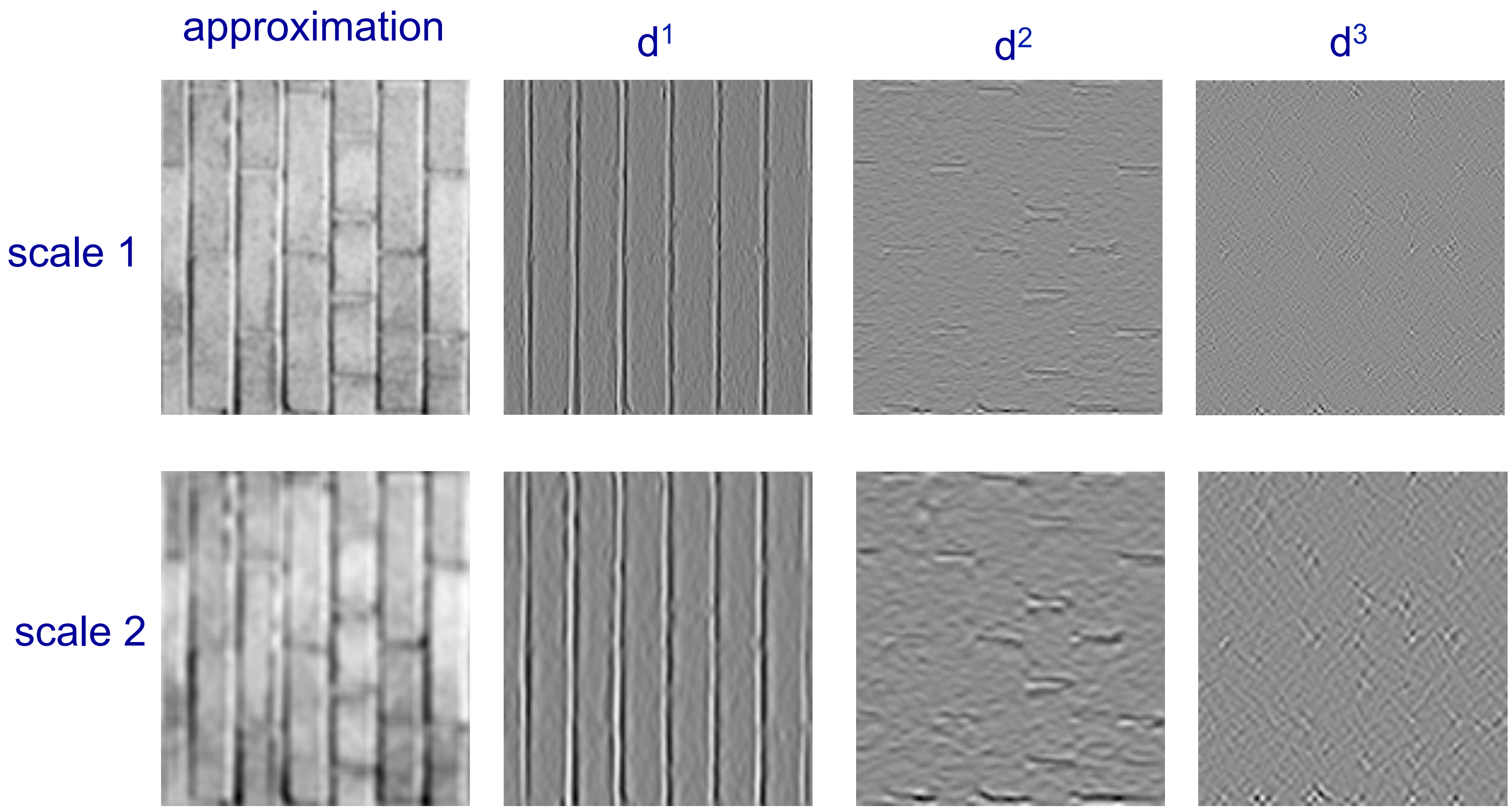


Image pyramid: feature extraction



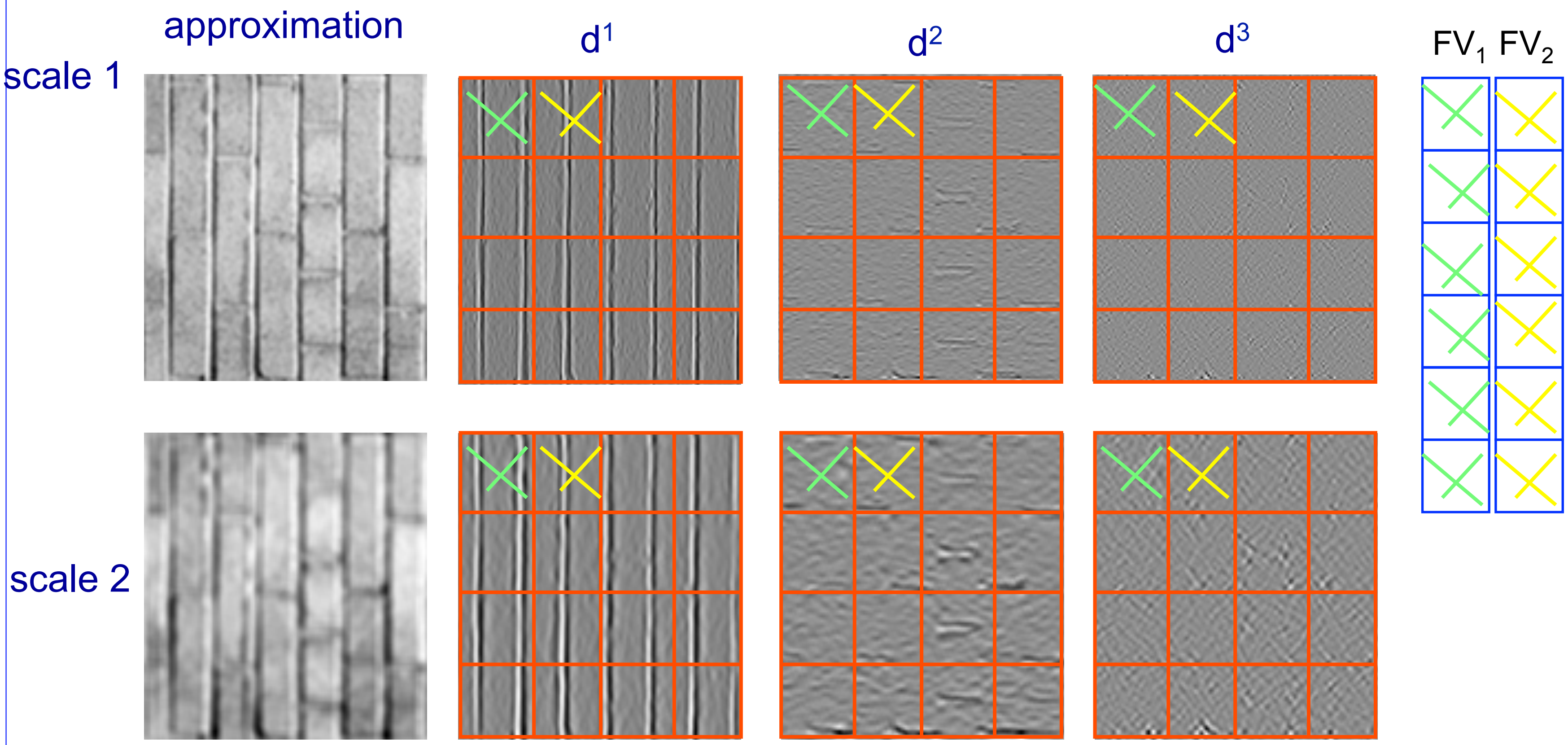
Texture features are claculated over the blocks and gathered into feature vectors.

Building the FV



Building the FV

elements of FV_1 of texture 1
elements of FV_2 of texture 1



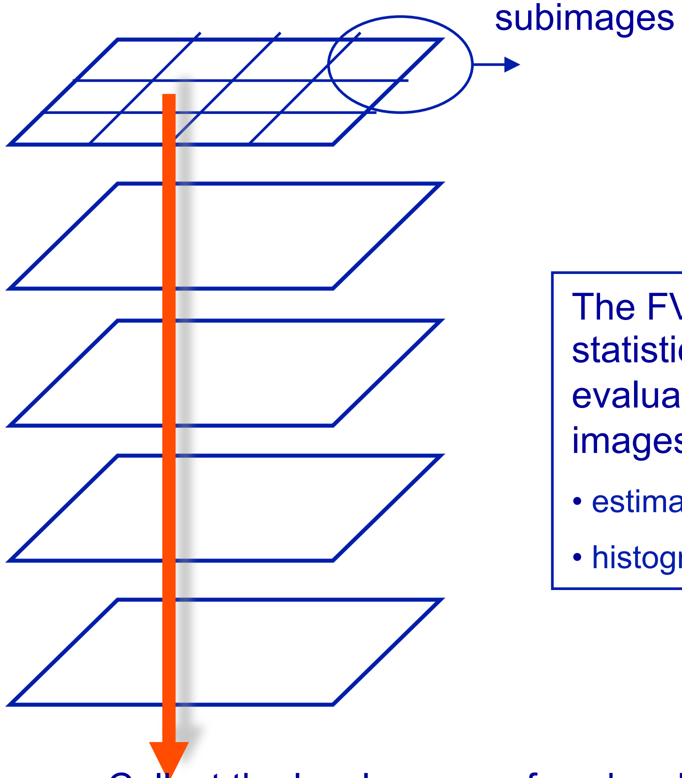
Feature extraction

- Step 2: extract features to form *feature vectors*

Intensity image
↓
Transformation
↓
For each sub-image
For each subband

Calculate the local energy (variance)

Fill the corresponding position in the FV
↓
One FV for each sub-image ⇨ Classification algorithm



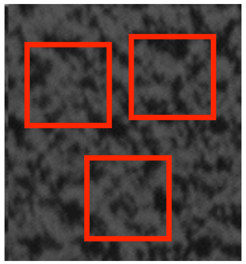
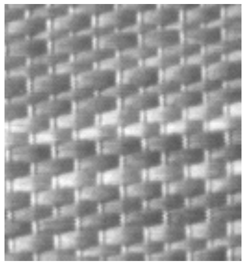
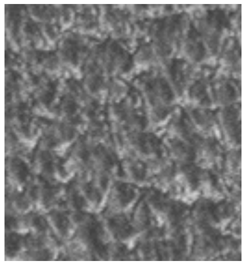
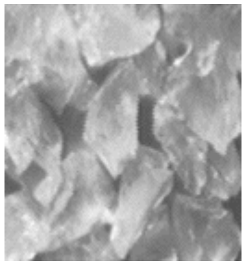
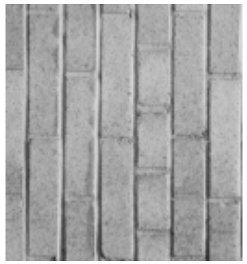
⇒ Collect the local energy of each sub-image in the different subbands in a vector

The FVs contain some statistical parameters evaluated on the subband images

- estimates of local variances
- histograms

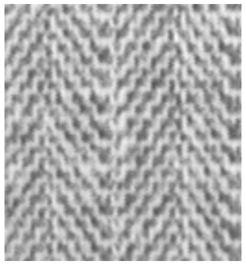
Texture features

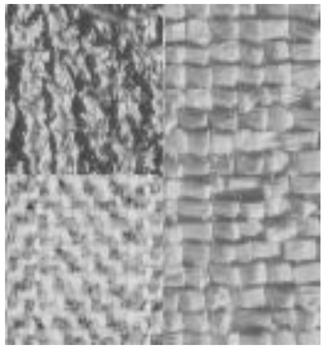
- No agreed reference definition
  - Texture is property of areas
  - Involves spatial distributions of grey levels
  - A region is perceived as a texture if the number of primitives in the field of view is sufficiently high
  - Invariance to translations
  - Macroscopic visual attributes
    - uniformity, roughness, coarseness, regularity, directionality, frequency [Rao-96]
  - Sliding window paradigm

Texture analysis

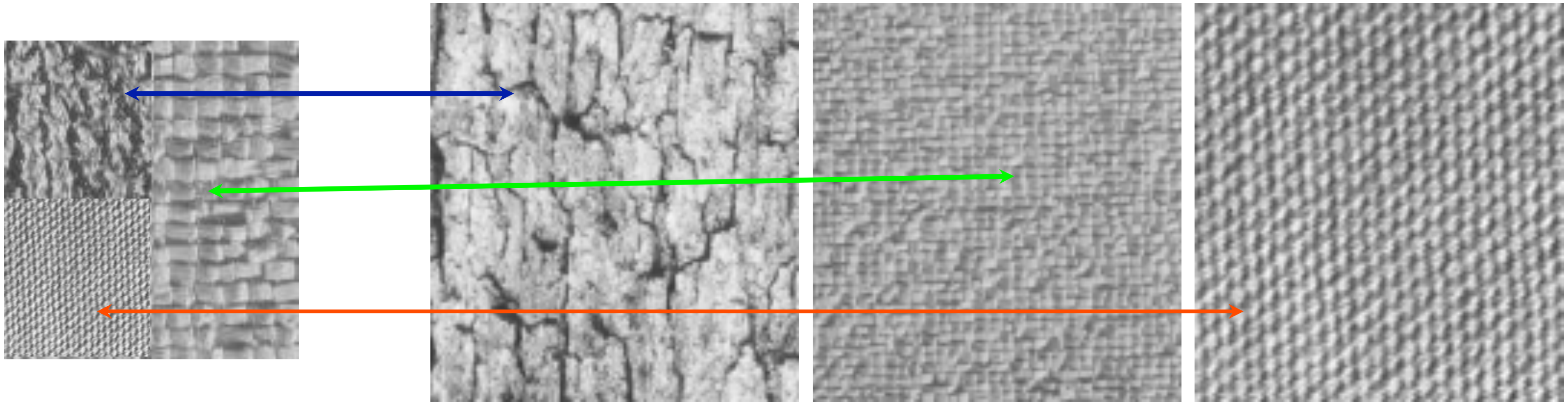
- Texture segmentation
  - Spatial localization of the different textures that are present in an image
  - Does not imply texture recognition (classification)
  - The textures do not need to be *structurally* different
  - *Apparent* edges
    - Do not correspond to a discontinuity in the luminance function
    - Texture segmentation ↔ Texture segregation





Texture analysis

- Texture classification (recognition)
  - Hypothesis: textures pertaining to the same class have the same visual appearance → the same *perceptual features*
  - Identification of the class the considered texture belongs to within a given set of classes
  - Implies texture recognition
  - The classification of different textures within a composite image results in a segmentation map

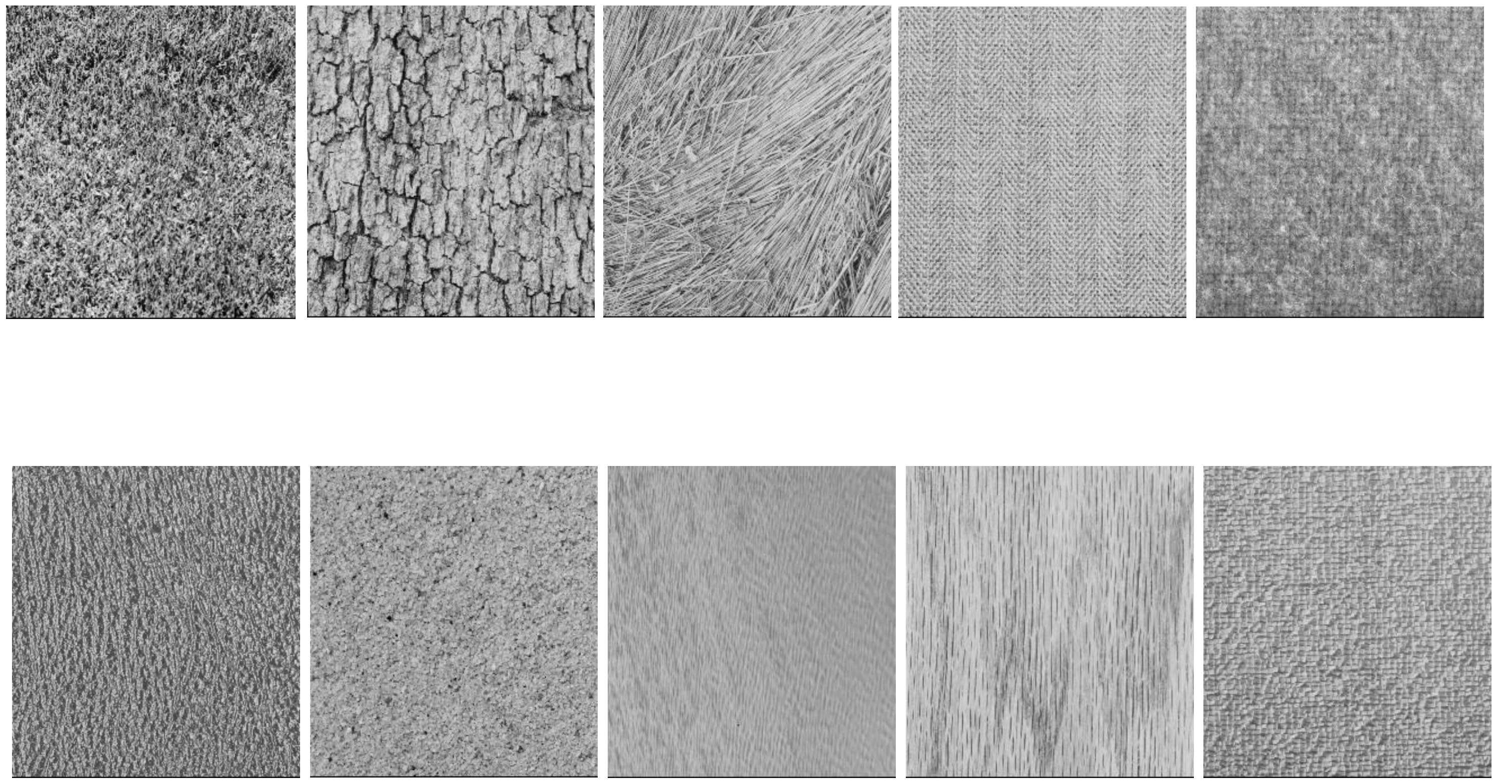


Texture classification

- Method
  - Describe the texture by some *features* which are related to its *appearance*
    - Texture → class → ω_k
    - Descriptors → Feature Vectors (FV) → $\boldsymbol{x}_{i,k}$
  - Define a distance measure for FV
  - Choose a *classification rule*
    - Recipe for comparing FV and choose 'the winner class'
  - Assign the considered texture sample to the class which is the *closest* in the feature space

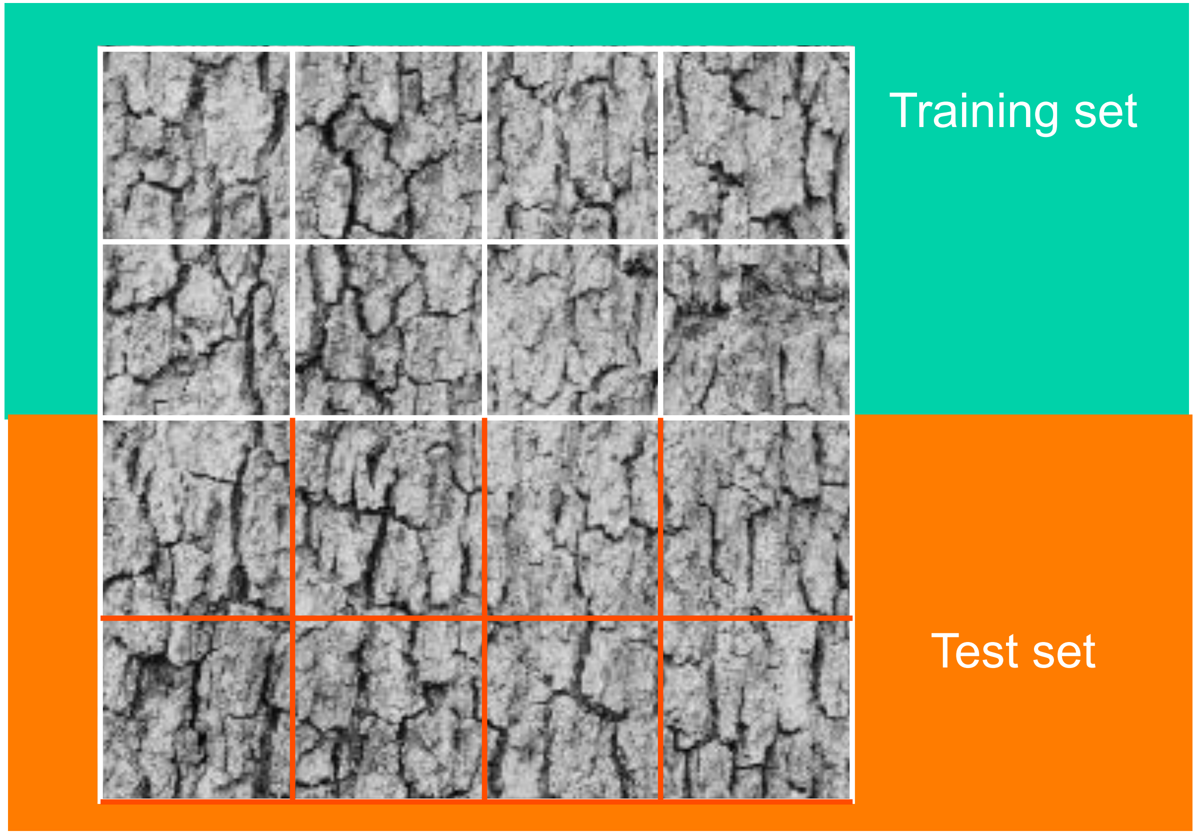
Exemple: texture classes

ω_1 ω_2 ω_3 ω_4



FV extraction

- Step 1: create independent texture instances



Co-occurrence matrix

- A co-occurrence matrix, also referred to as a co-occurrence distribution, is defined over an image to be the *distribution of co-occurring values at a given offset*.
- Mathematically, a co-occurrence matrix $C_{k,l}[i,j]$ is defined over an NxM image I, parameterized by an offset (k,l), as:

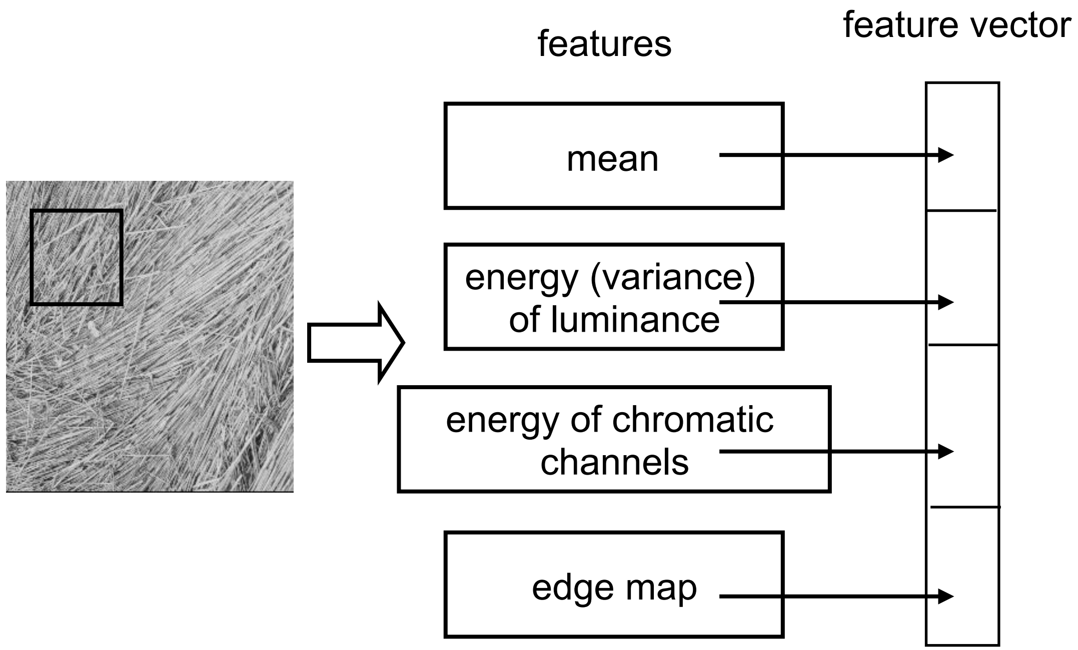
gray level values

$$C_{k,l}[i,j] = \sum_{p=1}^{N}\sum_{q=1}^{M}\begin{cases}1, \text{ if } I(p,q)=i \text{ and } I(p+k,q+l)=j \\ 0, \text{ otherwise}\end{cases}$$

- The co-occurrence matrix depends on (k,l), so we can define as many as we want

Feature extraction

- Step 2: extract features to form *feature vectors*



One FV for each sub-image ⇨ Classification algorithm

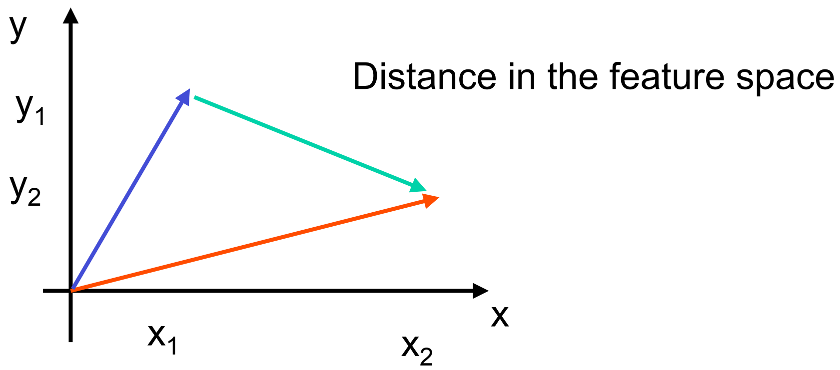
Feature vector distance

- Step 3: definition of a distance measure for feature vectors
  - Euclidean distance

$$d\left(\vec{v}_1,\vec{v}_2\right)=\sqrt{\left(x_1-x_2\right)^2+\left(y_1-y_2\right)^2+\ldots+\left(z_1-z_2\right)^2}$$

$$\vec{v}_1=\left\{x_1,y_1,\ldots,z_1\right\}$$

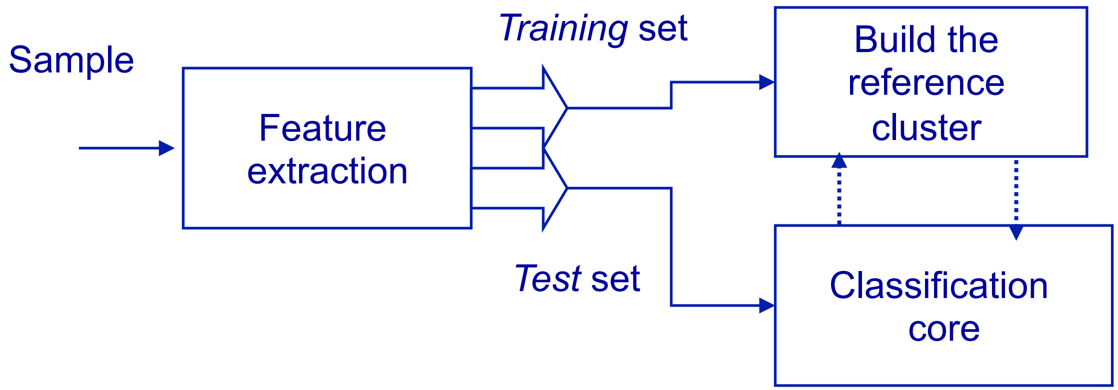
$$\vec{v}_2=\left\{x_2,y_2,\ldots,z_2\right\}$$



Classification steps

- Step 4: Classification
  - Phase 1: Training
    - The classification algorithm is provided with many examples of each texture class in order to build clusters in the feature space which are representative of each class
    - Examples are sets of FV for each texture class
    - Clusters are formed by ***aggregating vectors*** according to their "distance"
  - Phase 2: Testing
    - The algorithm is fed with an example of texture ω_i (vector $\boldsymbol{x}_{i,k}$) and determines which class it belongs to as the one to which it is "closest" in the feature space

Classification



Conclusions

- No golden rule exists for clustering/classification
- Major issues:
  - Feature selection
  - Definition of a metric for measuring distances
  - Definition of a representative training set
  - Choice of the classification/clustering strategy
- Possible solution: classifier fusion
  - Consider a set of different classifiers and combine their results