Introduction Machine Learning Kinect Lab Course

Olivier Pauly, Loren Schwarz, and Diana Mateus

Computer Aided Medical Procedures
Technische Universität München

Outline

Introduction to Machine Learning

Motivation for Machine Learning

If you have...

- **Immense amount of data**
  Different information sources, sensors, databases...
- **Little or no domain knowledge**
  Complex phenomena may be influenced by a lot of different factors. Difficult to find an appropriate mathematical model?
- **Uncertainty**
  Data may be corrupted, noisy, redundant, irrelevant...

Motivation for Machine Learning

Then you can use Machine Learning to:

- **LEARN**
  - **Understand** the data,
  - find relations or **patterns** in data,
  - design **models** explaining the observations
- **ACT**
  - Make **decisions** (ex: classification),
  - Perform **predictions** (ex: regression)

What is Machine Learning?

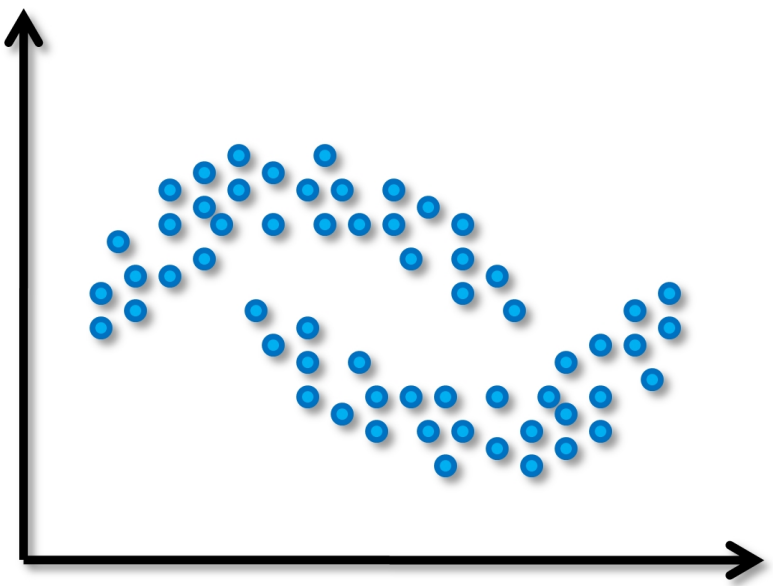
Main types of learning

- **Unsupervised Learning**
- **Supervised Learning**

What is Machine Learning?

Unsupervised Learning

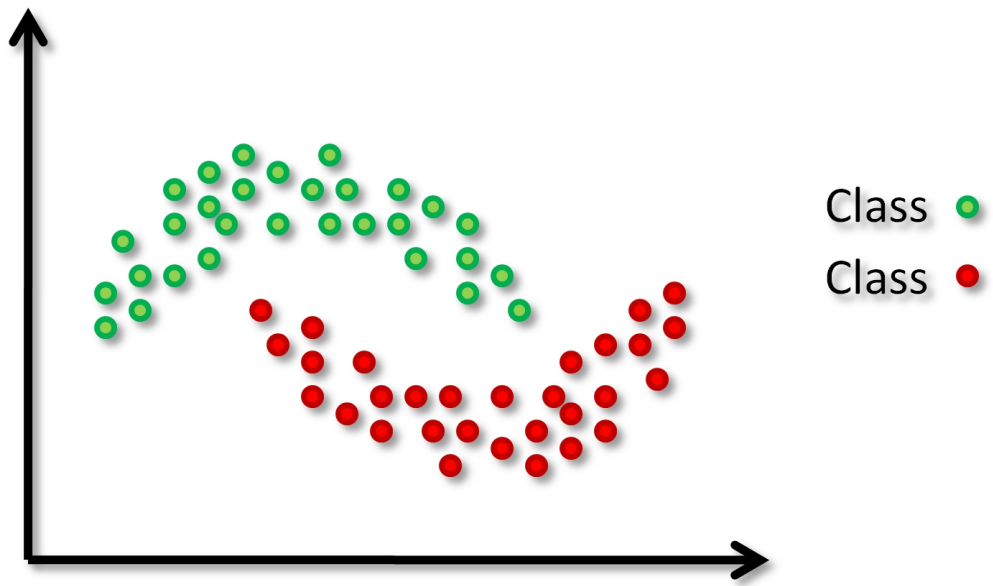
- **Goal**: discover groups of similar examples within observations (**clustering**) or estimate the distribution of the observations within the input space (**density estimation**)
- **Training data**: inputs only



What is Machine Learning?

Supervised Learning

- **Goal**: Make decisions or perform predictions based on observations (ex: classification, regression)
- **Training data**: inputs and corresponding outputs



Machine Learning

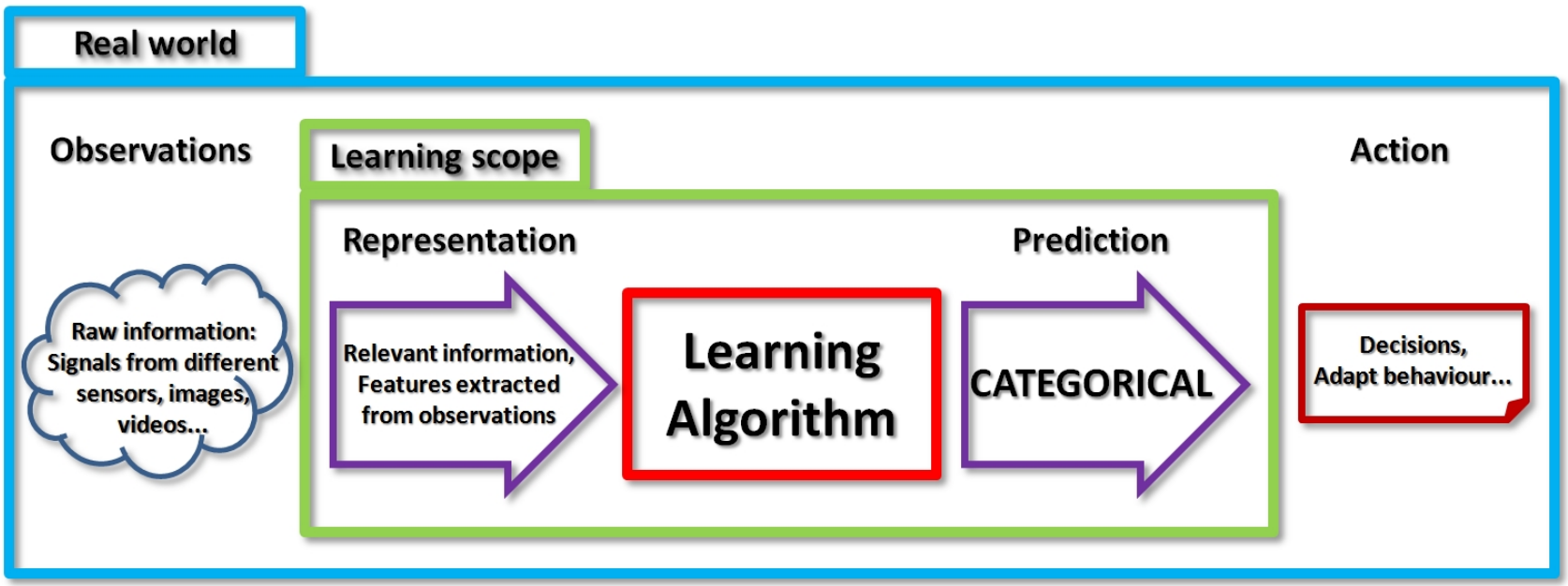
for Classification tasks

What is classification?

Definition

Method that **groups** observations or objects into classes or **categories** having **similar characteristics**

What is classification?



- Supervised Learning
- Classes: categorical outputs, labels
- 2 or more classes

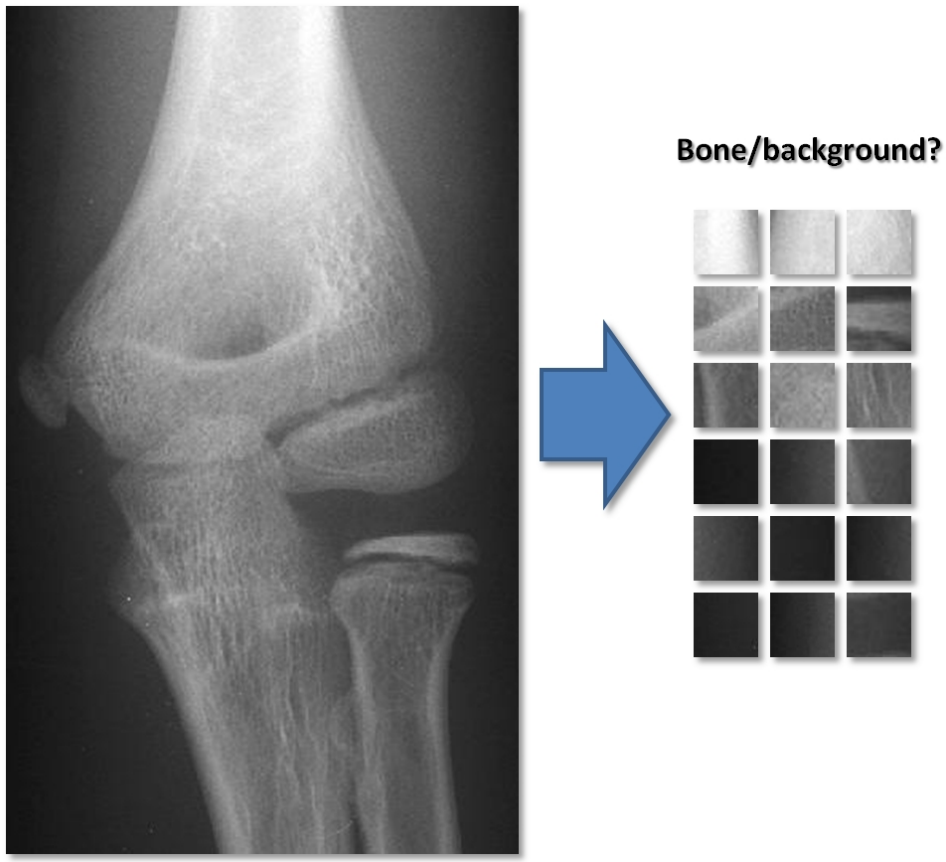
Example: Classify image patches

Observations: Image patches

Output: Bone/Background

To solve this task, we need:

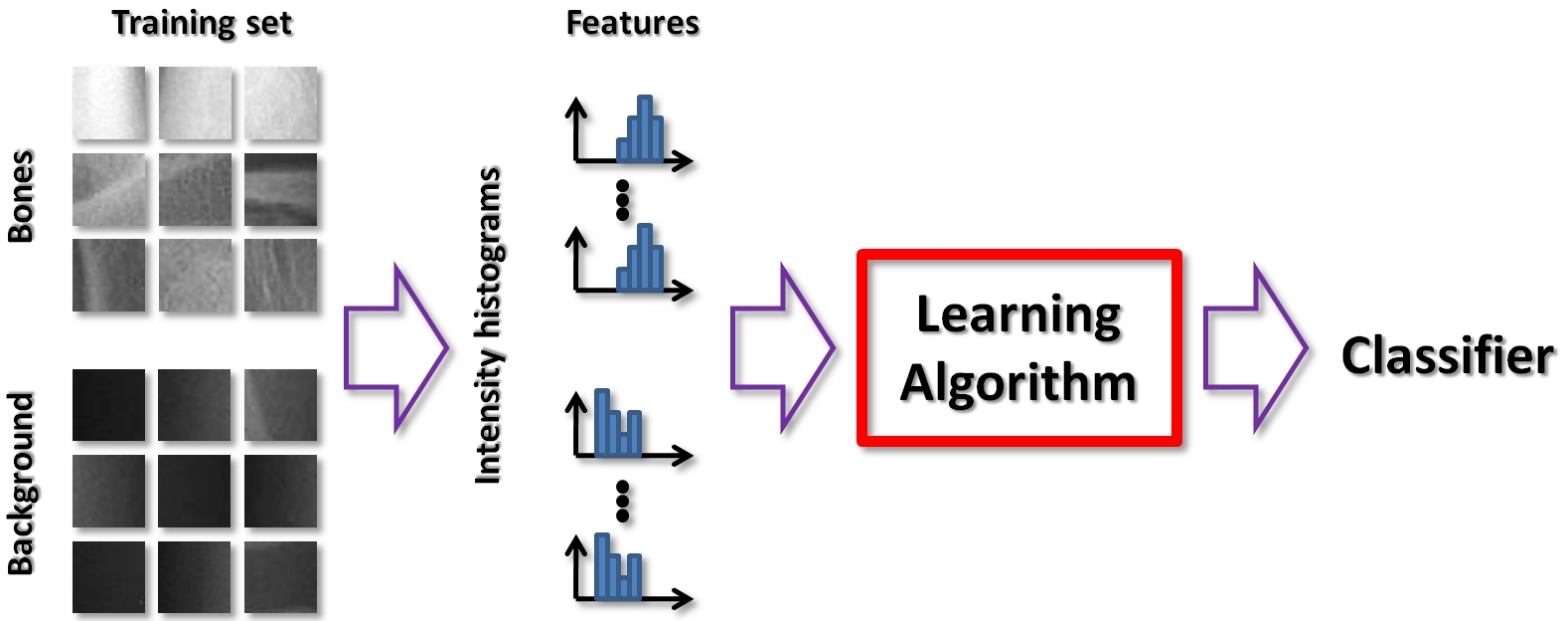
- Training set/Test set
- Feature representation
- Learning algorithm



Example: Classify image patches

Learning phase

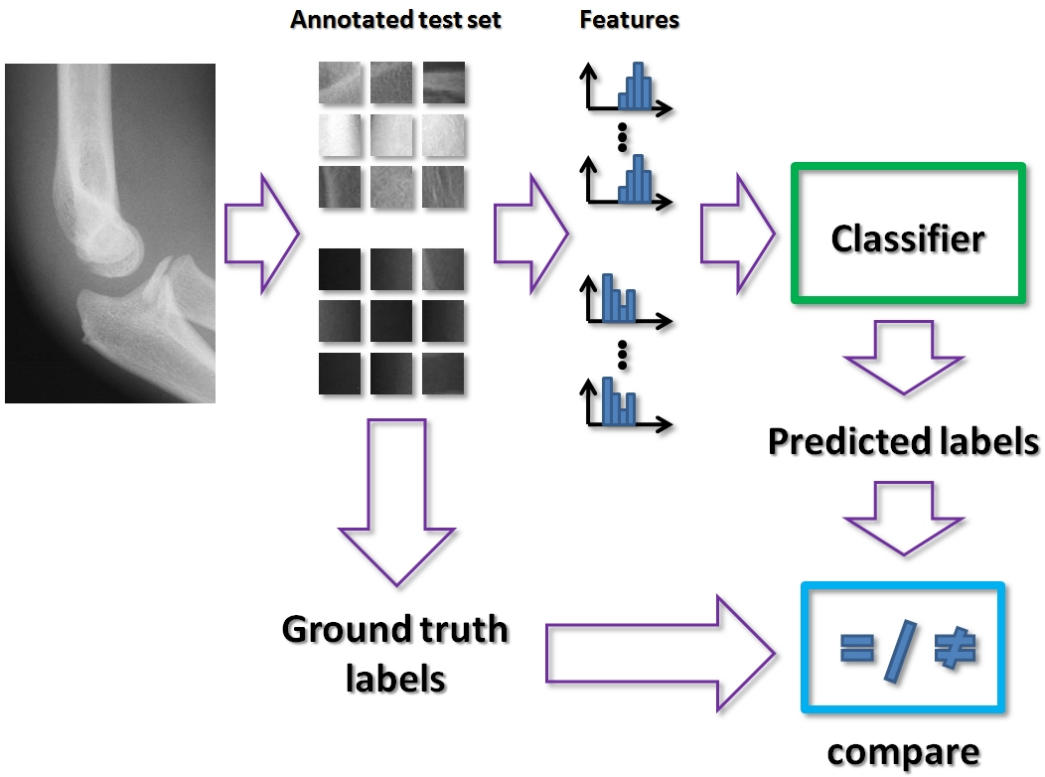
- Each patch is associated to a **label** Bone/Background
- **Intensity distribution** of the patch is chosen as a feature representation
- A **classifier** is created using a learning algorithm and the training set



Example: Classify image patches

Evaluation phase

- Create a test set
- Feed it into classifier
- Compare classifier outputs with ground truth



How to represent information:

Feature extraction

Sources of information

Input data can be:

- From different sources/types
- High-dimensional
- Redundant

What is a feature representation?

Image feature representation

- Describes **relevant** characteristics of the image: local or global, visual, statistical,...
- Is **compact**: feature dimensionality can be less than original information
- Is **discriminative**: permits to solve our classification problem

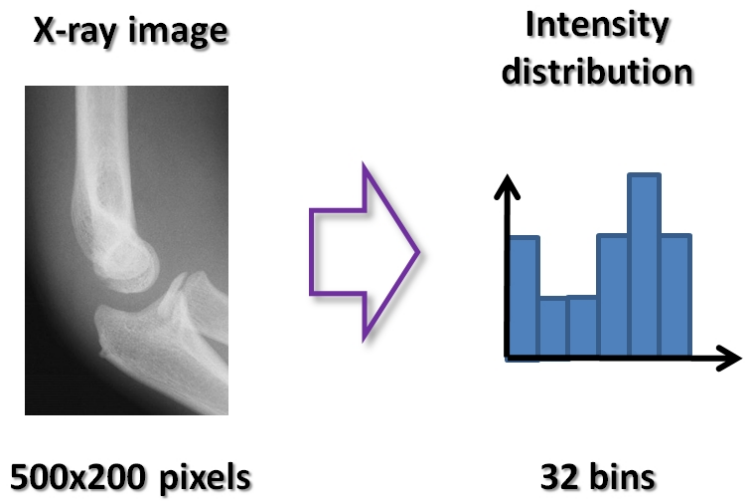


Image feature representation

Features can describe local or global properties of an image

- **Local** features:
  Intensities/Color, Texture, Context, Shape,
- **Global** features:
  Intensity/Color Statistics, Bag of visual words

Features can be:

- **Multi-scale**, to describe the information contained in different parts of the frequency spectrum
- **Invariant** under certain transformation, which increases their robustness

Image feature representation

Basic Local Features

- Pixel/Patch intensity/color
- Image derivatives (gradient,...)

Contextual/Textural Local Features

- Local binary patterns (LBP)
- Histograms of oriented gradients (HOG)
- Haar-like features
- Shape Context

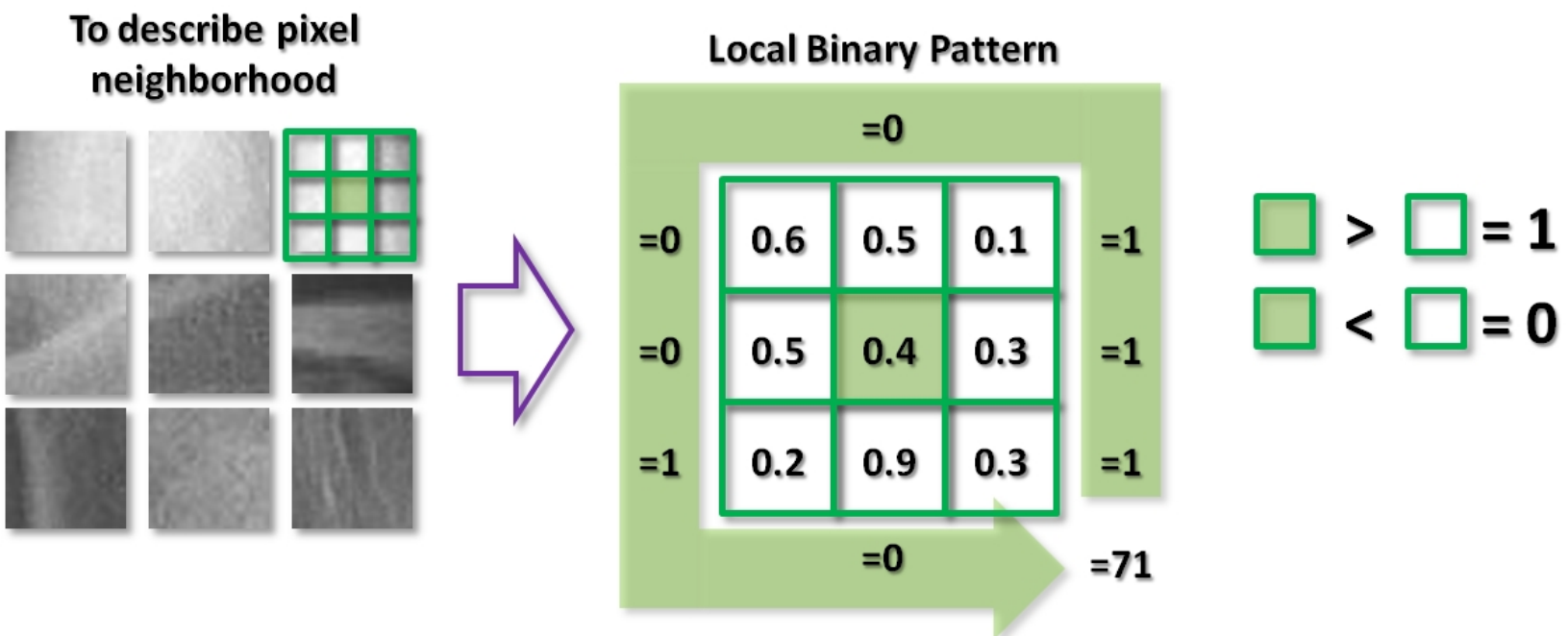
Global Features

- Intensity/Color histograms
- Bag of visual words

Image feature representation

Local Binary Patterns (LBP)

Describes the local intensity variations



$$1 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + 0 \times 2^3 + 0 \times 2^4 + 0 \times 2^5 + 1 \times 2^6 + 0 \times 2^7 = 71$$

Image feature representation

2D and 3D Histogram of oriented gradients (HOG)

Describes the local distribution of gradient orientation weighted by the gradient magnitude

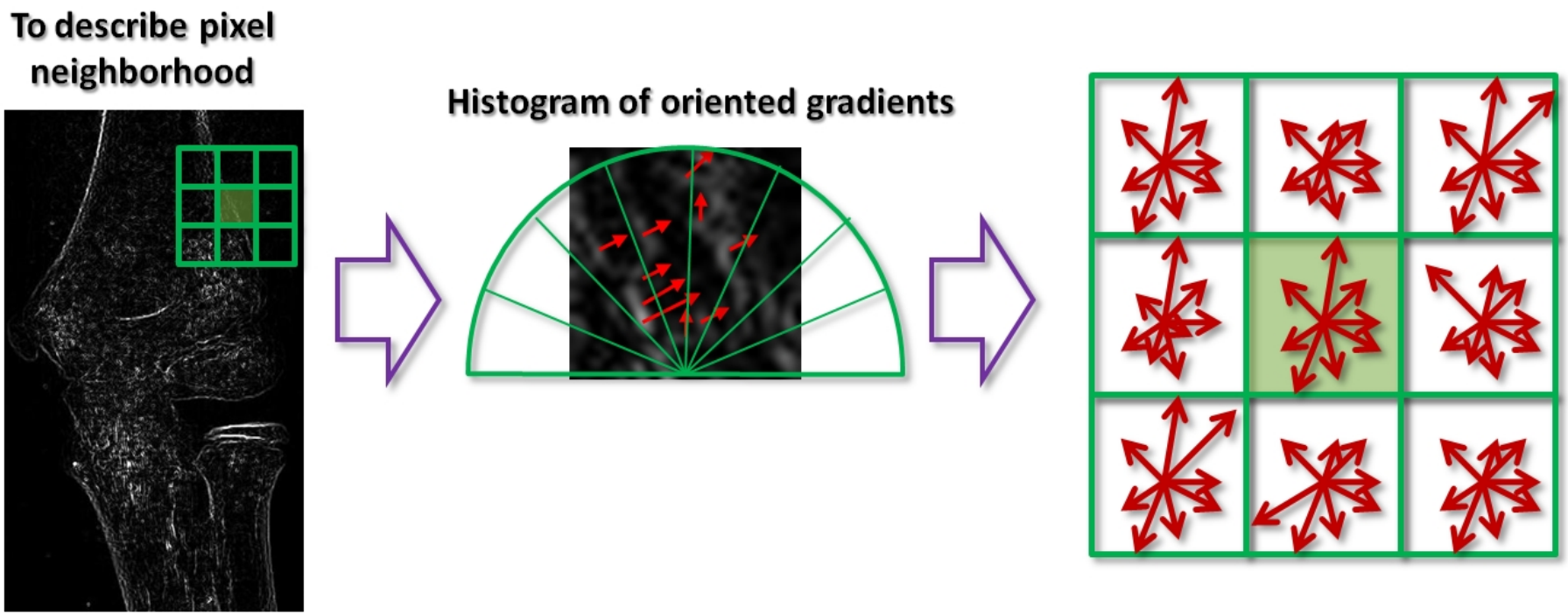


Image feature representation

Haar-like features

Describes the local context by using a dictionary of haar wavelets

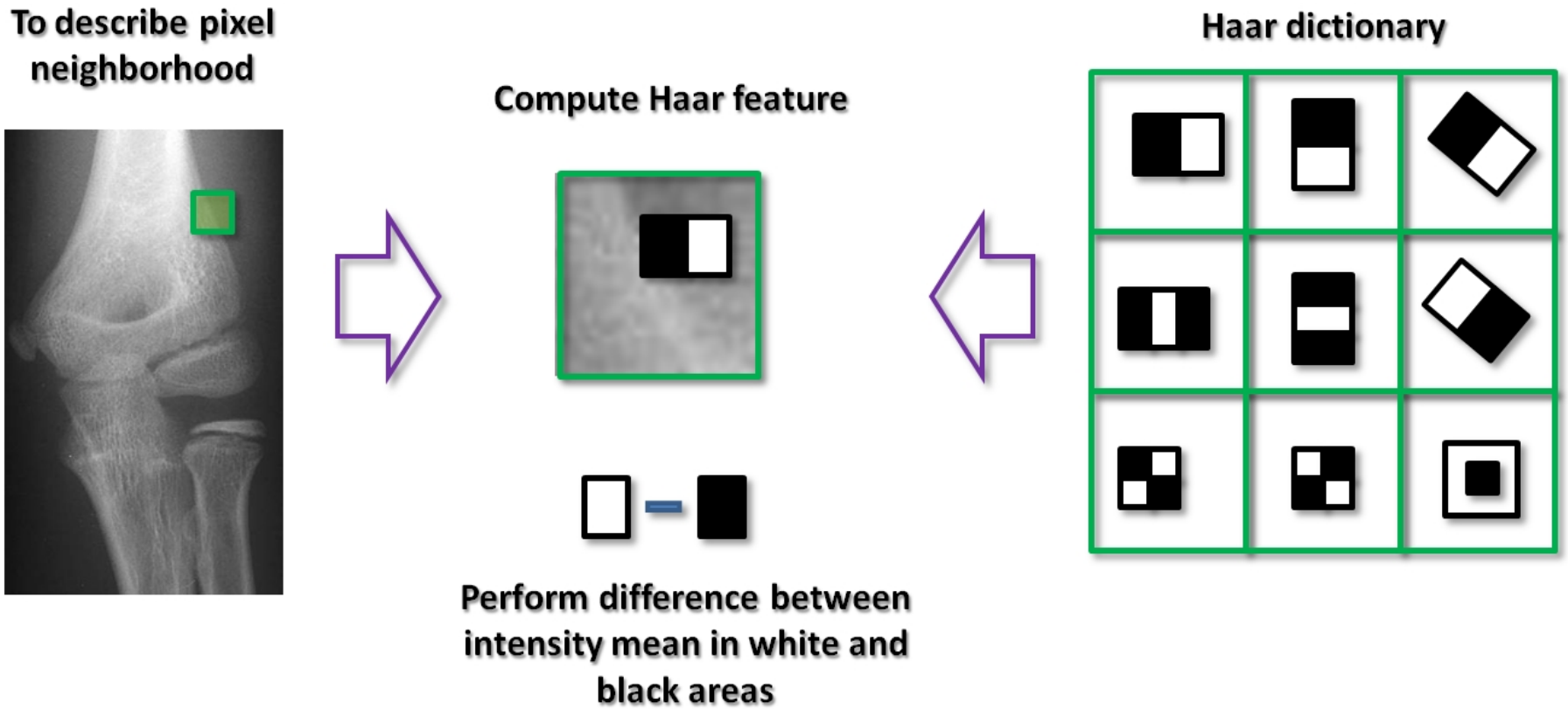


Image feature representation

Haar-like features

Describes the local context by using a dictionary of haar wavelets

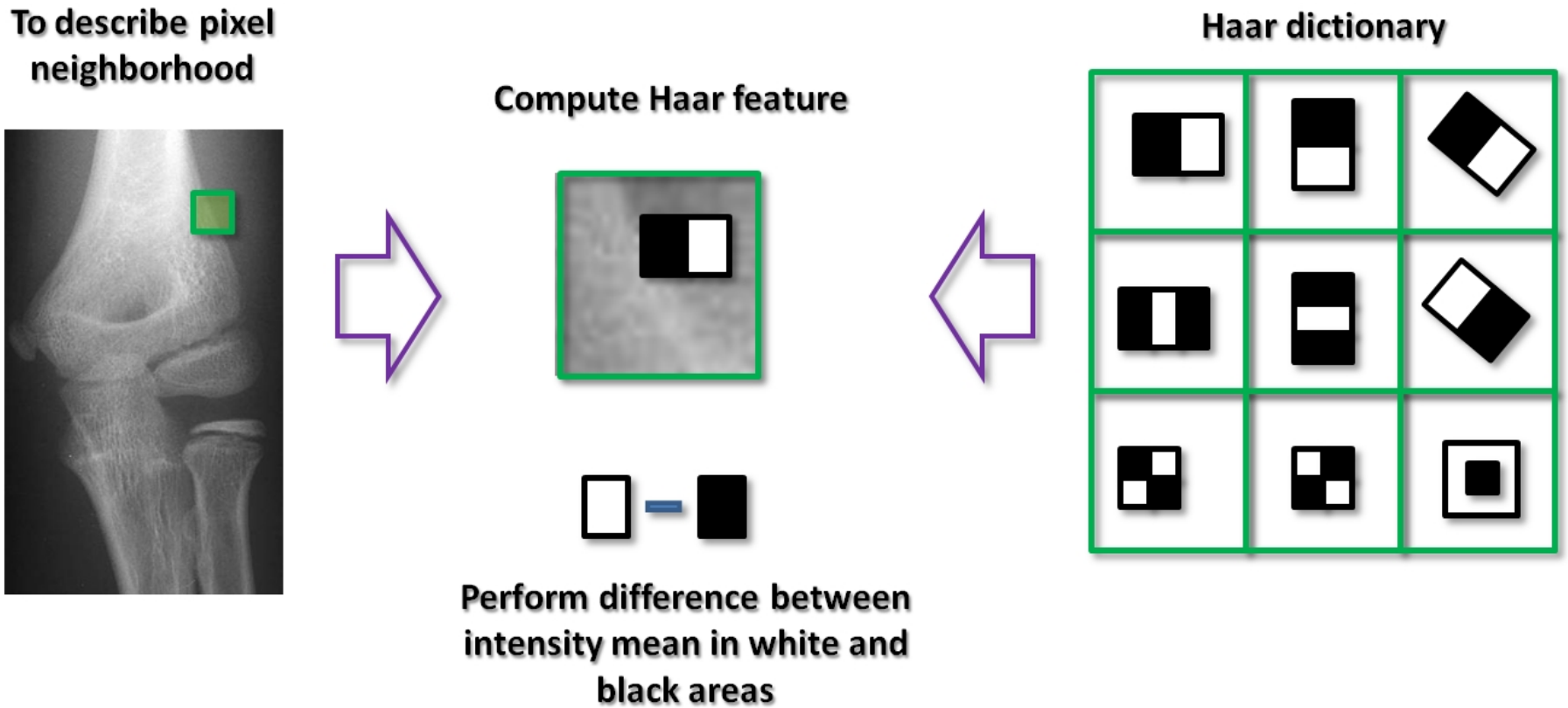


Image feature representation

2D and 3D Shape Context

- Describes a shape defined as a point cloud by approximating the point **distribution** observed from each point.

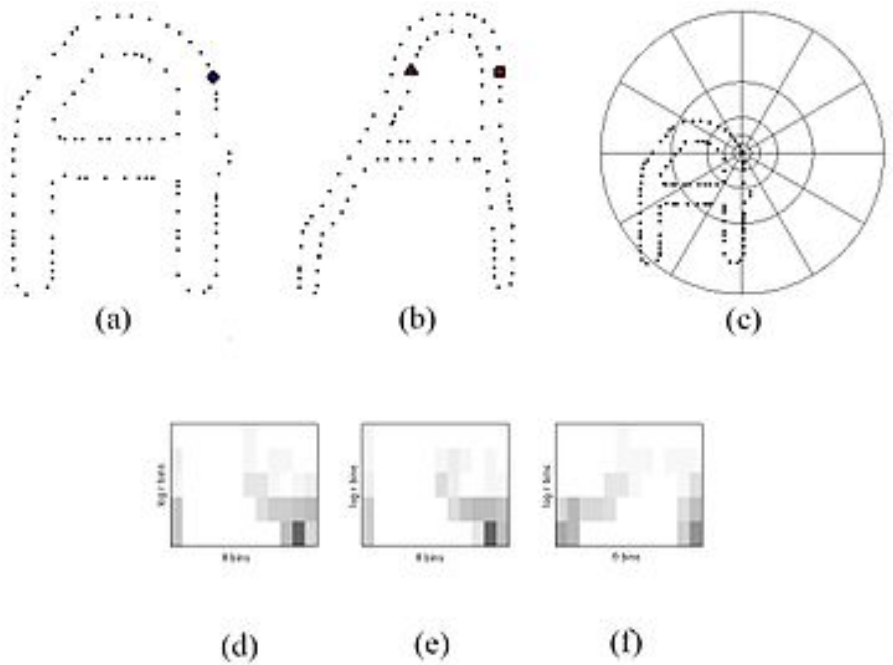


Image feature representation

Multi-scale analysis

Describes the information contained in different frequency bands using Gaussian filter.

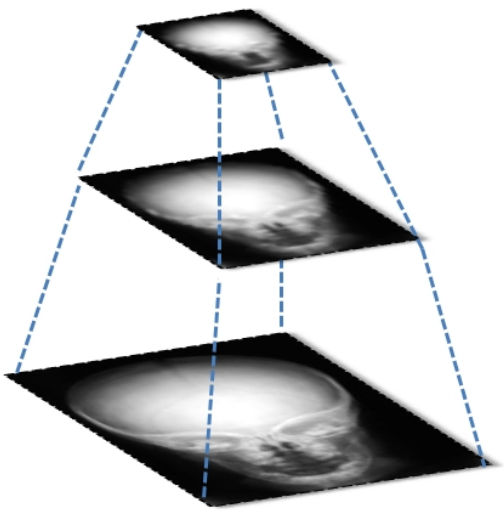
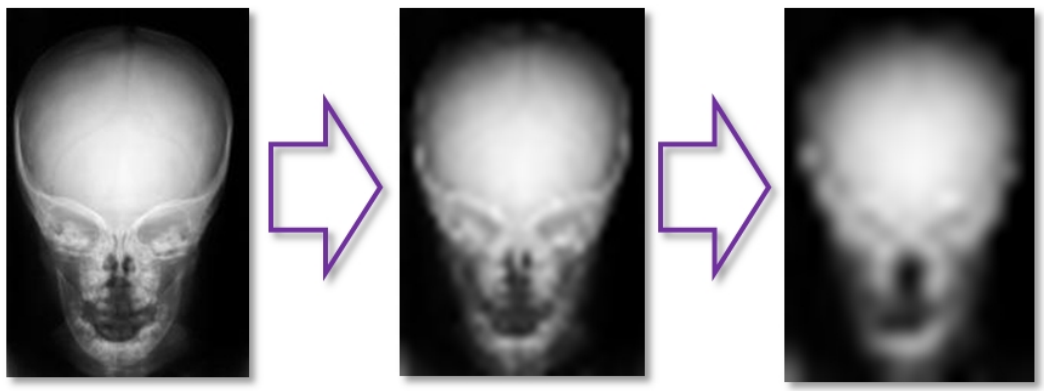


Image pyramid



Finer scale

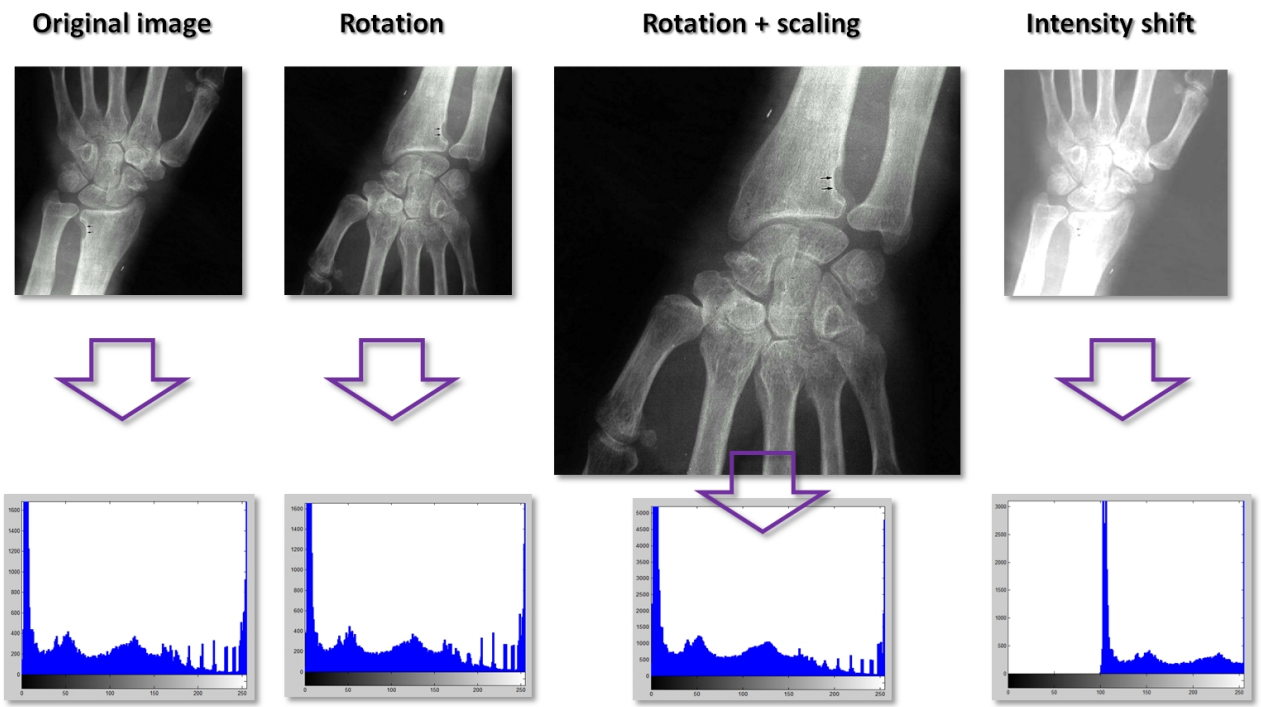
Coarser scale

Image feature representation

Invariance

Transformations impact feature representations

Ex: Intensity histograms



Feature representation: the dimensionality

If features are:

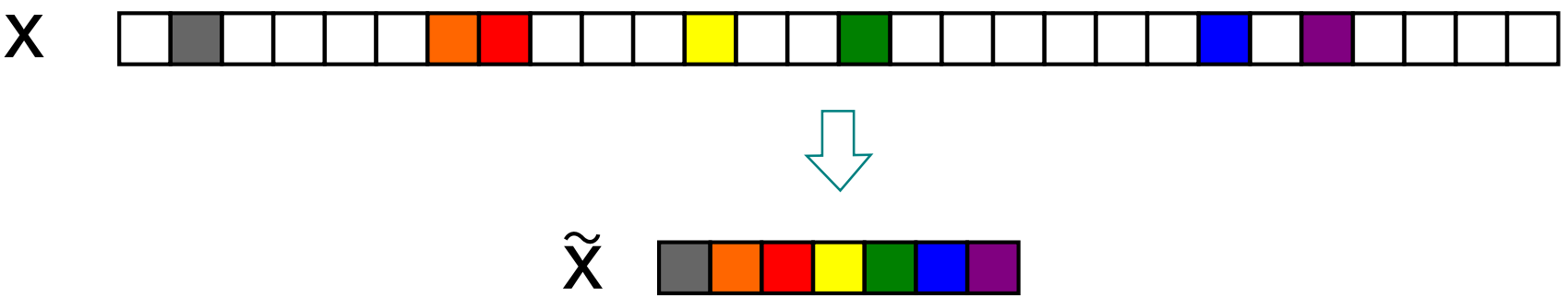
- **High dimensional**
  - better representation of the data
  - processing (e.g. classification) is slow
- **Low dimensional**
  - fast processing
  - poor representation of the data

Feature representation: the dimensionality

If features are:

- **High dimensional**
  - better representation of the data
  - processing (e.g. classification) is slow
- **Low dimensional**
  - fast processing
  - poor representation of the data

Intermediate solution is to **reduce the dimensionality** of high dimensional features trying to keep the important information.



Dimensionality Reduction Methods

Map the original data to the space with reduced dimensionality

Linear methods use a linear transformation $\mathbf{T}$,

$$\tilde{\mathbf{X}} = \mathbf{X}\mathbf{T}$$

Ex: Principal Component Analysis (PCA), Independent Component Analysis (ICA).

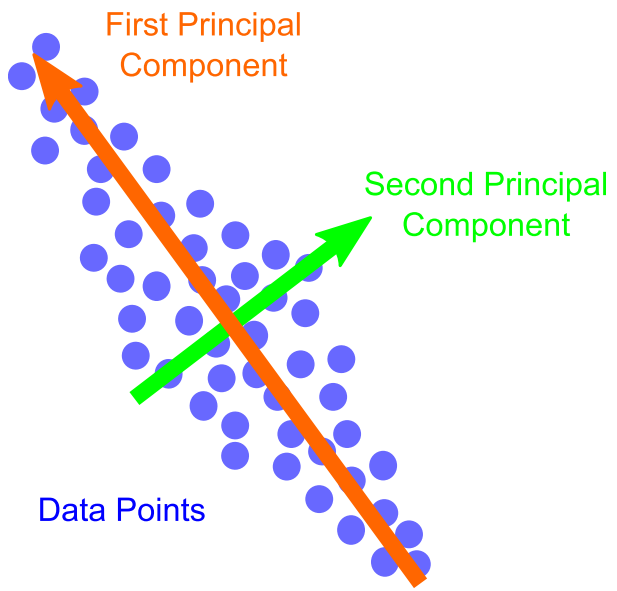
Non-linear methods use a non-linear map $\tilde{\mathbf{X}} = \Phi(\mathbf{X})$.

Ex: Isomap, Locally-linear Embedding (LLE), Laplacian Eigenmaps.

Linear methods: Principal Component Analysis

The **Principal Components** are

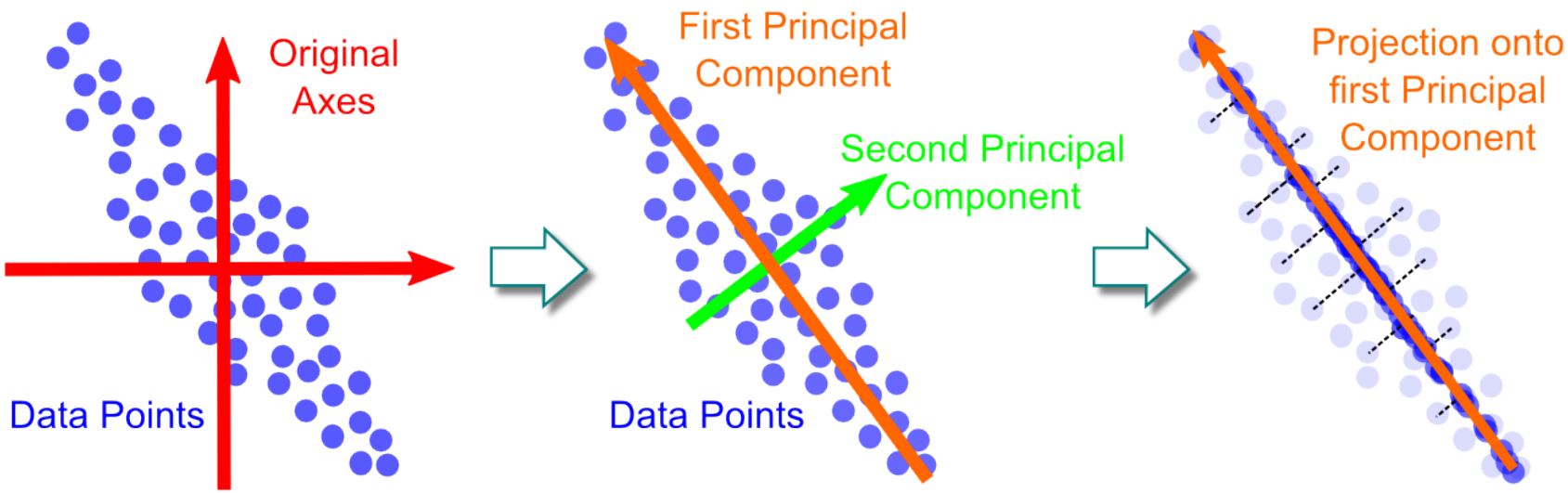
- the **directions of maximum variance** in the input space.
- **ordered** such that principal axis 1 has the highest variance, axis 2 has the next highest variance, ... , etc.
- **uncorrelated** (orthogonal) to each other. The covariance between each pair of principal axes is zero.



Linear methods: PCA

PCA finds a map $\tilde{\mathbf{X}} = \mathbf{XT}$ such that:

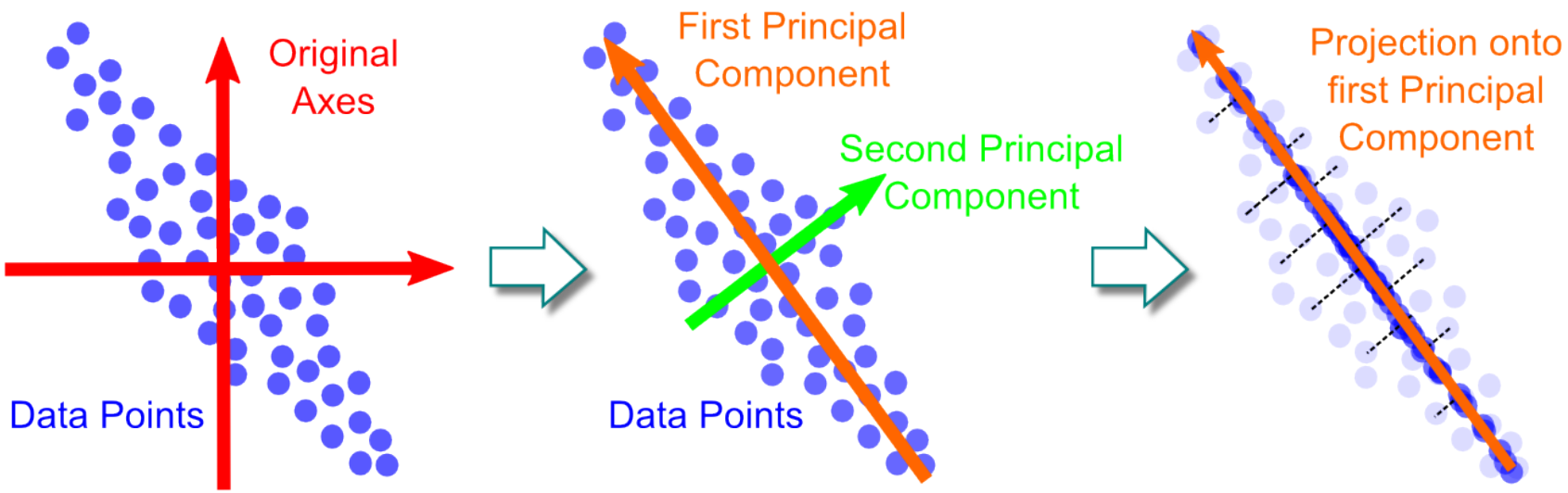
- $\mathbf{T}$ is an **orthogonal linear** transformation.



Linear methods: PCA

PCA finds a map $\tilde{\mathbf{X}} = \mathbf{XT}$ such that:

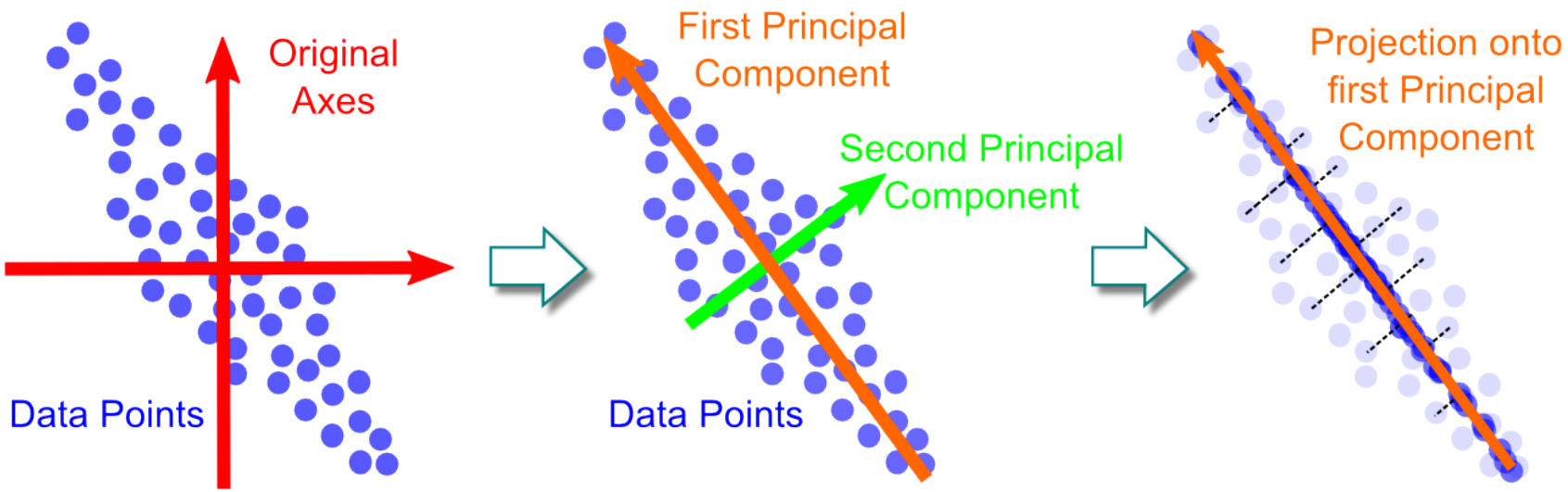
- $\mathbf{T}$ rotates the data to a **new coordinate system**, in order to **align** coordinate axis with the principal components.



Linear methods: PCA

PCA finds a map $\tilde{\mathbf{X}} = \mathbf{XT}$ such that:

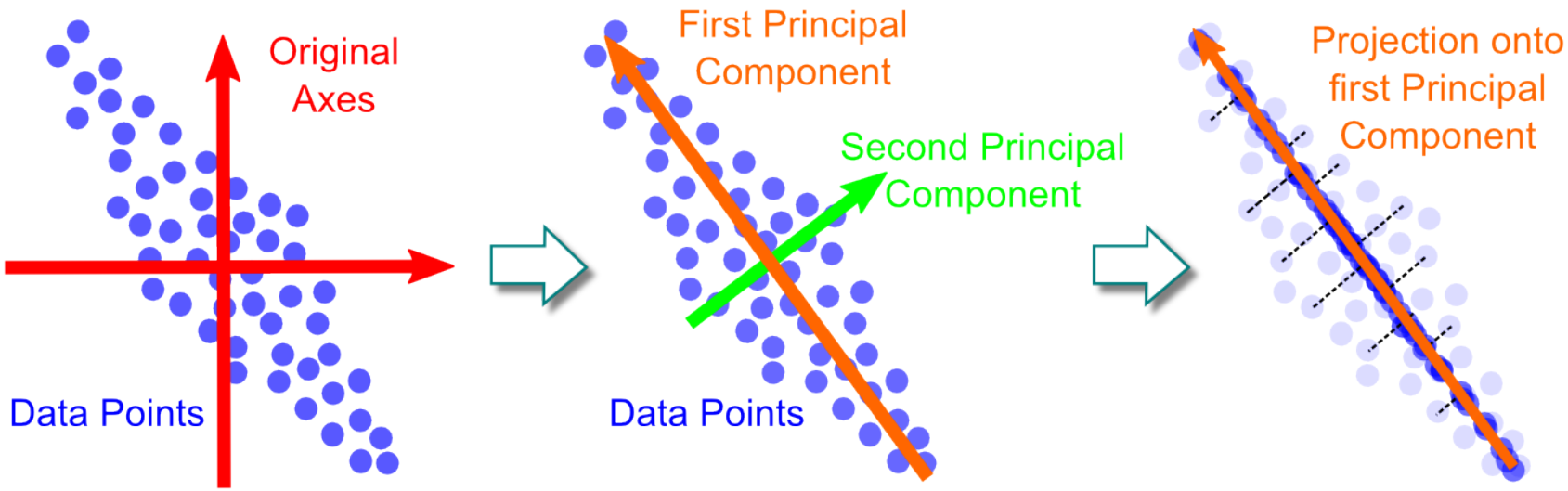
- After the rotation, most of the variance lies in the first few dimensions.



Linear methods: PCA

PCA finds a map $\tilde{\mathbf{X}} = \mathbf{XT}$ such that:

- The values in the remaining dimensions, tend to be small and may be dropped with minimal loss of information. PCA is used in this manner for dimensionality reduction.



Outline

Learning Algorithms for Classification

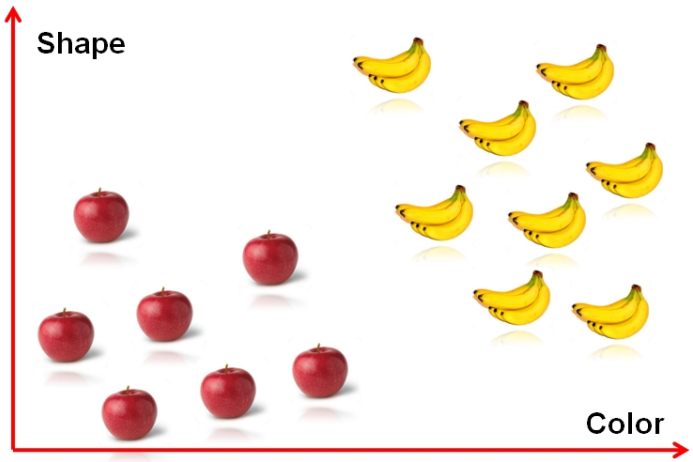
From linear classifiers to Support Vector Machines

A classical binary classification problem...

Let us consider 2 classes:  and 

- Given observations and their associated labels $\left\{\mathbf{x}^{(j)},\ \mathbf{y}^{(j)}\right\}_{j\in\{1,\cdots,N\}}$

$$\mathbf{x}^{(3)} = \text{  }$$
$$\mathbf{y}^{(3)} = \text{“apple”}$$



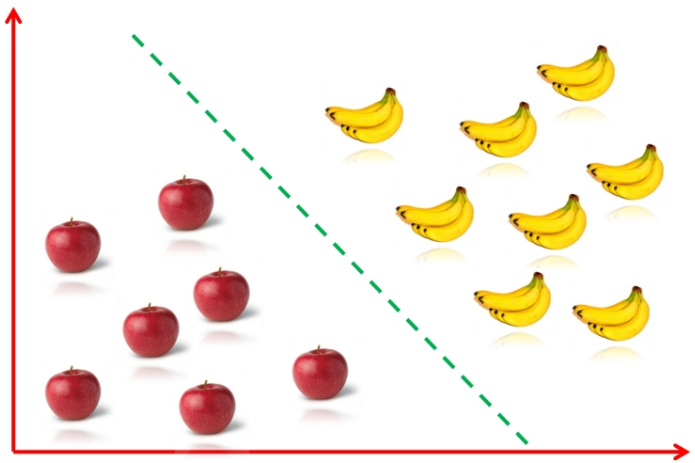
- **Goal**: find a **decision function** $\mathcal{F}$ so that

$$\mathcal{F}(\mathbf{x}^{(j)}) = \mathbf{y}^{(j)}$$

Generalized Linear Classifier

- Given training data
- Find a **hyperplane** separating your observations in 2 classes:

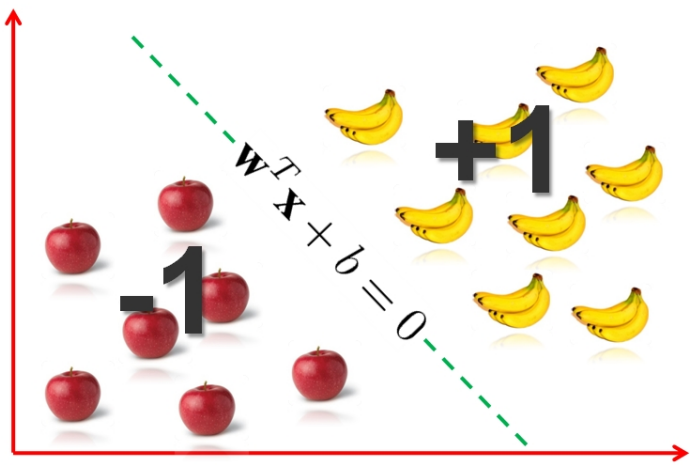
$$\mathbf{w}^\top \mathbf{x} + b = 0$$



Generalized Linear Classifier

- Given training data
- Find a hyperplane separating your observations in 2 classes:

$$\mathbf{w}^\top \mathbf{x} + b = 0$$



- Subject to constraints :

$$\text{if } \mathbf{y}^{(j)} = +1 \rightarrow \text{above}$$
$$\text{if } \mathbf{y}^{(j)} = -1 \rightarrow \text{bellow}$$

Generalized Linear Classifier

GOAL: Find the optimal parameters $\mathbf{w}_o$ and b_o which minimize the classification errors

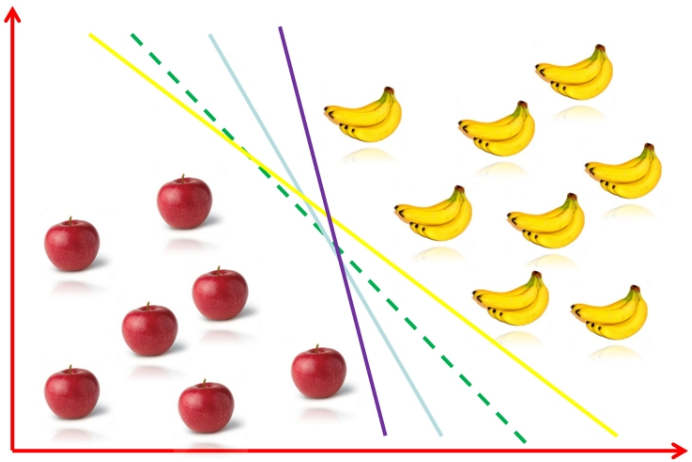
Can be solved if data are **linearly separable**, if not NP-hard!

OUTPUT:

Decision function

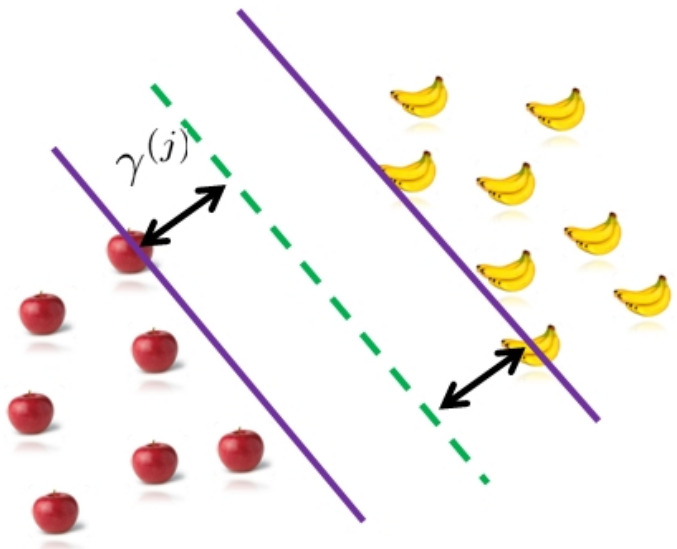
$$\mathcal{F}(\mathbf{x}) = \text{sign}(\mathbf{w}_o^\top \mathbf{x} + b_o)$$

A lot of solutions!



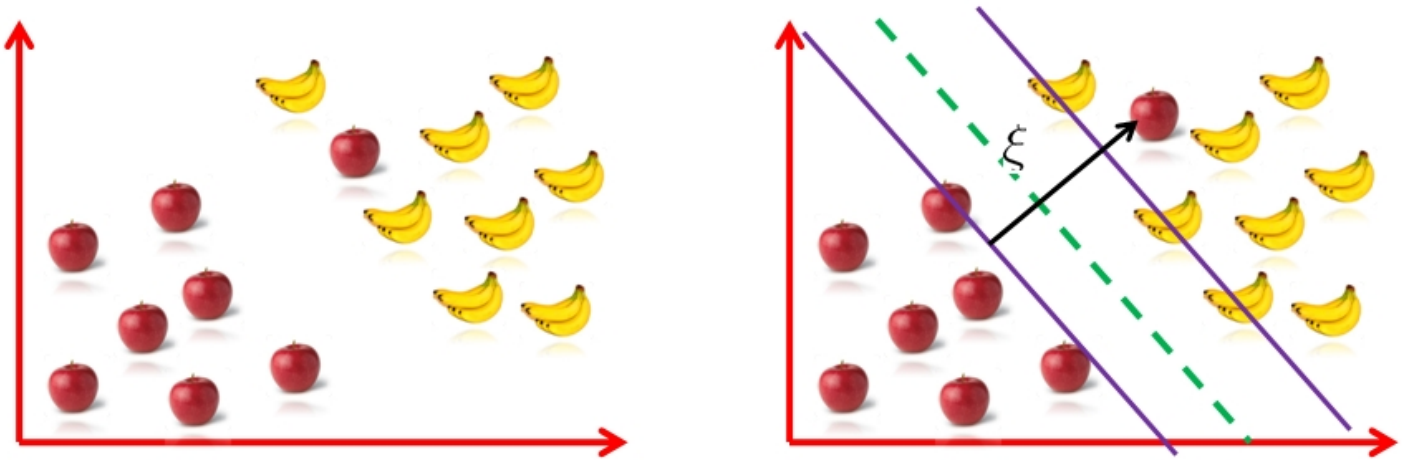
- A lot of hyperplanes solve this problem but which one is the best?
- Choose the best separating hyperplane which maximizes the margin!

A lot of solutions!



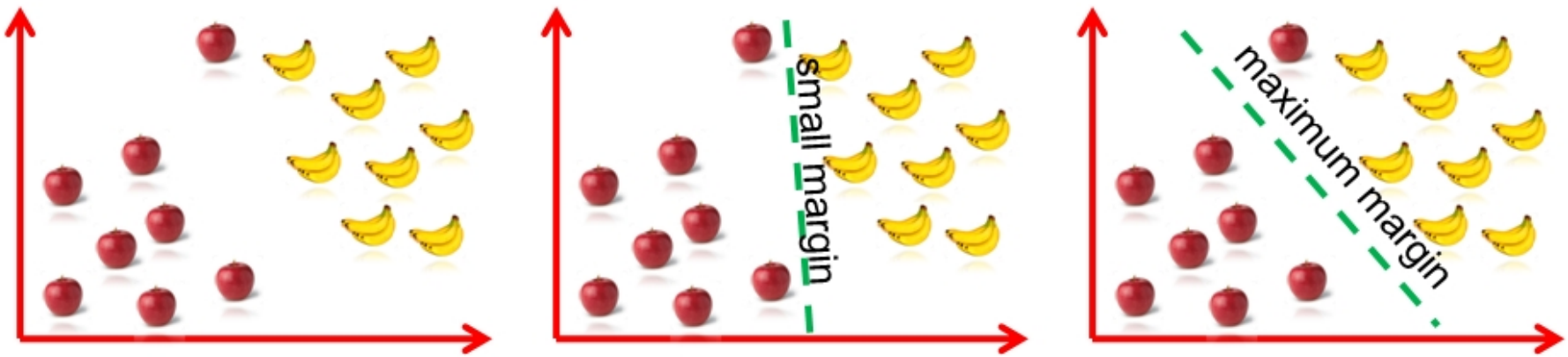
- A lot of hyperplanes solve this problem but which one is the best?
- Choose the best separating hyperplane which maximizes the margin!

Linear separability



- Data are not always **linearly separable**
- Noise in the features? Mislabelled data? Outliers?
- **Idea**: allow **margin violations**, i.e. misclassification errors = **soft margin**

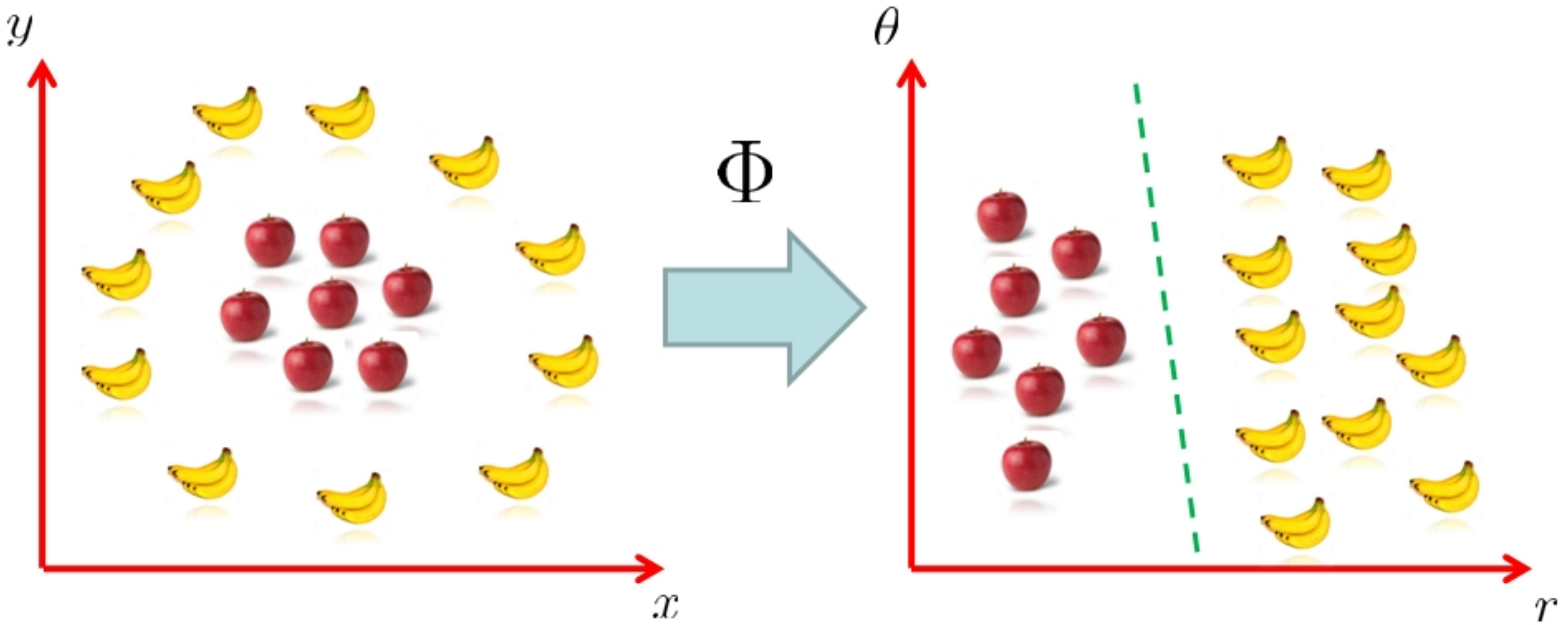
Generalization ability



Soft margin

- provides a better **generalization** ability
- prevents from **overfitting** the data, i.e. prevents the model of describing random errors (**outliers**) or **noise**.

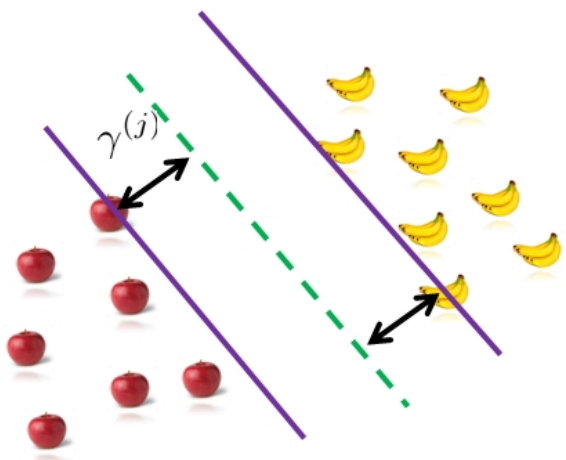
Non-linear separability



- Data are not always **linearly separable**, however...they may be separable in another (higher dimensional) space!
- **Idea**: find a **non-linear mapping** from the input space into a (higher dimensional) feature space in which data are separable.

Support Vectors

Include the **soft margin** and the **non-linear mapping** in the optimization to find the parameters of the **decision function**



$$\mathcal{F}(\mathbf{x}) = \text{sign}(\mathbf{w}_o^\top \Phi(\mathbf{x}) + b_o)$$

The solution $\mathbf{w}_o$ is a linear combination of the training data!

Solution

$$\mathbf{w}_o = \sum_{j=1}^{N} \alpha_j \mathbf{y}^{(j)} \Phi(\mathbf{x}^{(j)})$$

where:

- $\alpha_j \geq 0$ are weights
- the $\mathbf{x}^{(j)}$ for which $\alpha_j \neq 0$ are called **support vectors**.

Introducing a non-linear mapping

Problem

$$\mathcal{F}(\mathbf{x}) = \text{sign}(\mathbf{w}_o^\top \Phi(\mathbf{x}) + b_o)$$

Solution

$$\mathbf{w}_o = \sum_{j=1}^{N} \alpha_j \mathbf{y}^{(j)} \Phi(\mathbf{x}^{(j)})$$

The decision function can be written:

Decision function

$$\mathcal{F}(\mathbf{x}) = \text{sign}\left(\sum_{j=1}^{N} \alpha_j \mathbf{y}^{(j)} \Phi(\mathbf{x}^{(j)})^\top \Phi(\mathbf{x}) + b_o\right)$$

But we do not know the non-linear mapping $\Phi(\mathbf{x})$

Kernel Trick

Instead of defining the map Φ, define the corresponding **Kernel**:

Kernel

$$K(\mathbf{x}, \mathbf{z}) = \Phi(\mathbf{x})^\top \Phi(\mathbf{z})$$

Support Vector Machine (SVM)

$$\mathcal{F}(\mathbf{x}) = \text{sign}(\sum_{j=1}^{N} \alpha_j \mathbf{y}^{(j)} K(\mathbf{x}^{(j)}, \mathbf{x}) + b_o)$$

- $K(\mathbf{x}^{(j)}, \mathbf{x})$ can be interpreted as a **Similarity measure** in the feature space

Kernels...

Many possible choices for the kernel function:

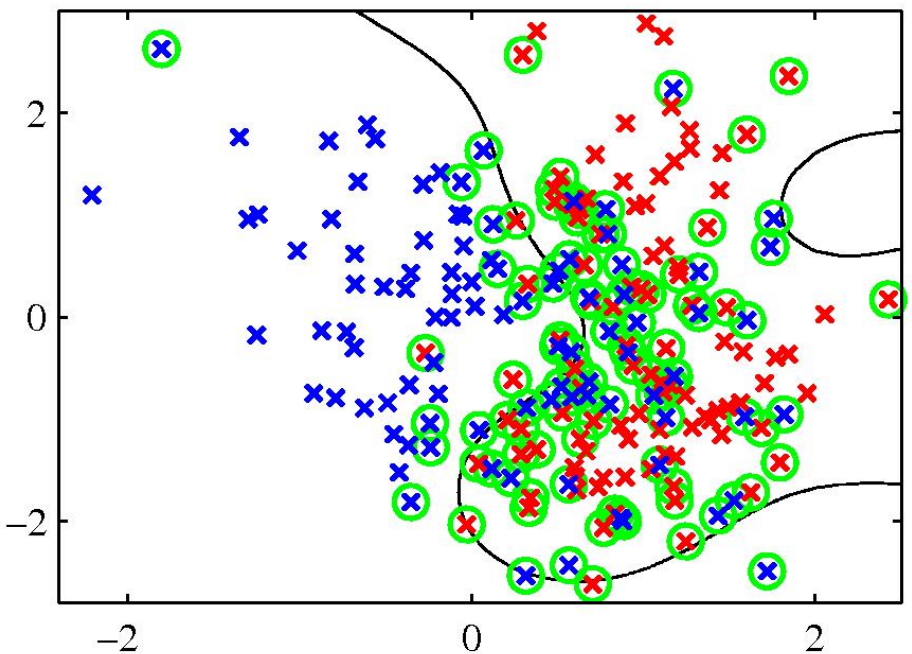
Kernel functions

- **Linear**:
  $K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}'$
- **Polynomial**:
  $K(\mathbf{x}, \mathbf{x}') = \left(\gamma \mathbf{x}^\top \mathbf{x}' + c\right)^k$
- **Radial Basis Function (RBF)**:
  $K(\mathbf{x}, \mathbf{x}') = \exp\left(-\gamma \left\|\mathbf{x} - \mathbf{x}'\right\|^2\right)$
- **Sigmoid**:
  $K(\mathbf{x}, \mathbf{x}') = \tanh\left(\gamma \mathbf{x}^\top \mathbf{x}' + c\right)$

Support Vector Machines (SVM)

Support Vector Machine (SVM)

$$\mathcal{F}(\mathbf{x}) = \text{sign}(\sum_{j=1}^{N} \alpha_j \mathbf{y}^{(j)} K(\mathbf{x}^{(j)}, \mathbf{x}) + b_o)$$



C.Bishop, 2006

Conclusion

Support Vector Machines

- SVMs find the **optimal** separating hyperplane which **maximizes** the margin between the classes
- Non-linear decision function can be modeled by using **Kernels**. They can project data in higher dimensional features space in which they become linearly separable.
- In case of non separable data, using a **soft margin** allows misclassification errors. This provides also better generalization ability in the cases of noisy data or outliers.

However: no formulation for multi-class classification...

Multi-class classification

Multi-class SVM?

- No multi-class formulation
- Several multi-class solutions based on combinations of several models:
  - **One-vs-one**: ensemble of binary SVMs trained to discriminate between pairs of classes.
  - **One-vs-rest**: ensemble of binary SVMs trained to discriminate between a class and all the rest.

Real multi-class solutions

- K-Nearest Neighbors classifier
- Decision trees, Random forests

Outline

1. Introduction to Machine Learning
2. Machine Learning for Classification tasks
3. How to represent information: Feature extraction
4. Learning Algorithms for Classification