Lecture 9-10

Object Categorisation

Bag of Words

Maximum Margin Classifier

Tae-Kyun Kim

Object Categorisation - Challenges

Illumination

Poses/orientations

Clutter

Occlusions

intra-class variations

View-points

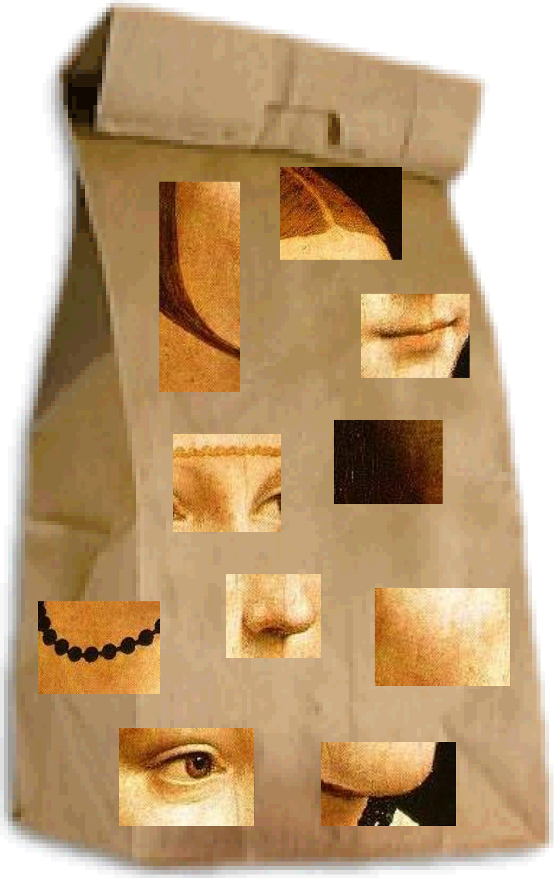
Image Representation
- Bag of Words

A good image representation is required.
Image are often represented as finite dimensional vectors.
Later, the vectors are classified into object categories.

image $\rightarrow$ $\mathbf{x} \in \mathbb{R}^D$

Bag of Words Model

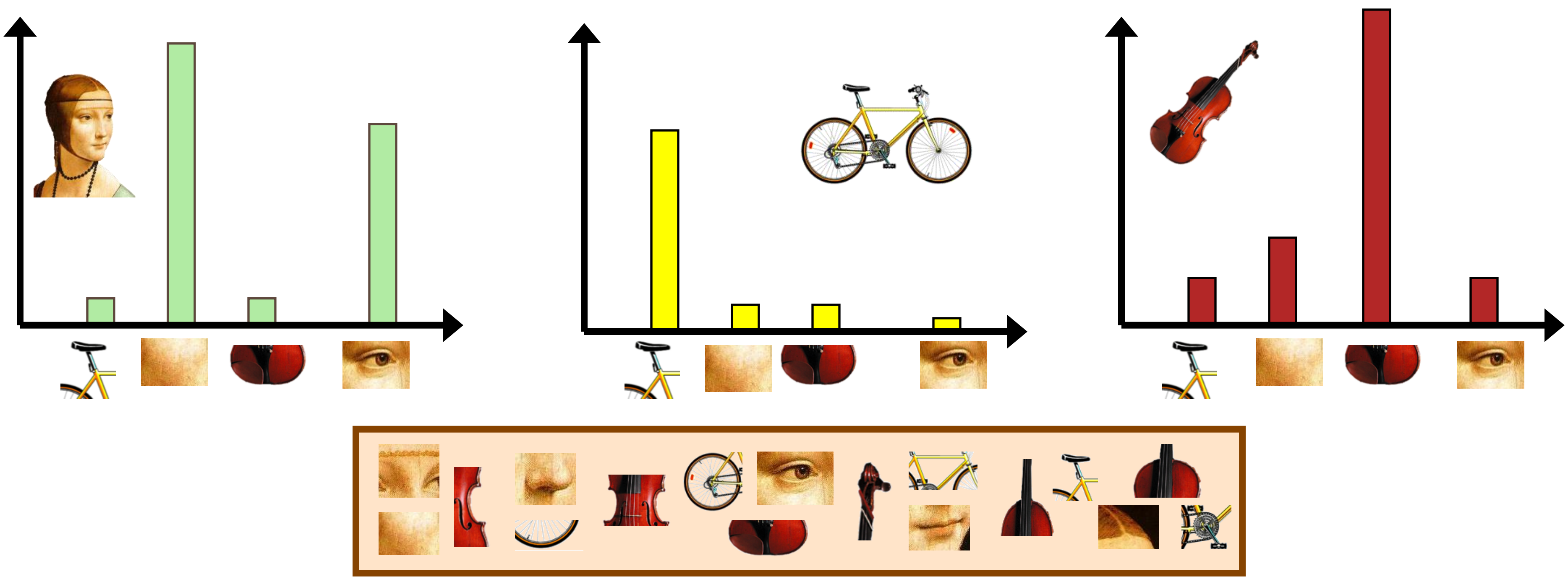
Object → **Bag of 'words'**



From L. Fei-Fei, 2009

Bag of Words

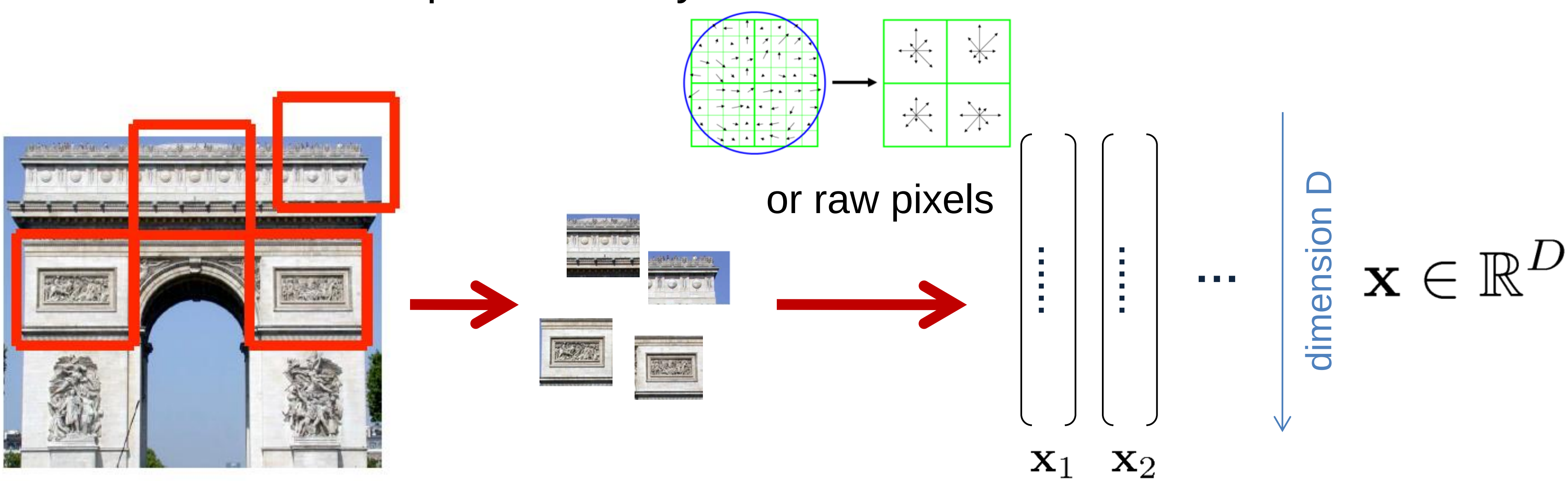
- Independent features (code words)
- Histogram representation



From L. Fei-Fei, 2009

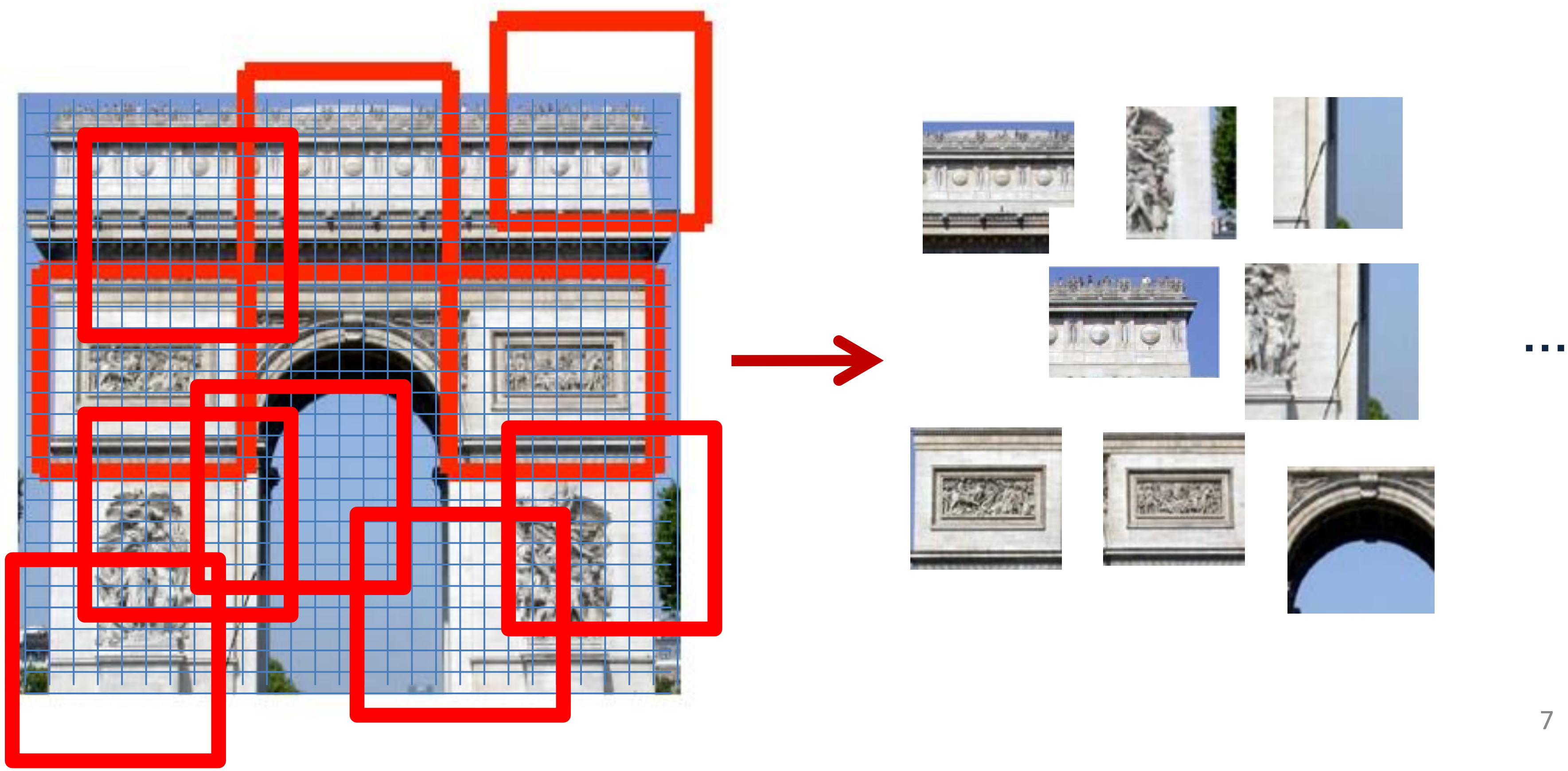
Visual Words

- Visual words are base elements to describe an image.
- Interest points are detected from an image.
  - E.g. Corners, Blob detector, SIFT (Scale-Invariant Feature Transform) detector (http://www.cs.ubc.ca/~lowe/keypoints/)
- Image patches are collected and represented by descriptors.
  - SIFT or raw pixel intensity



Visual Words

- Instead of a sparse Interest point detector, we can use a *dense grid*.
- Image patches are collected around *all points* on the grid.

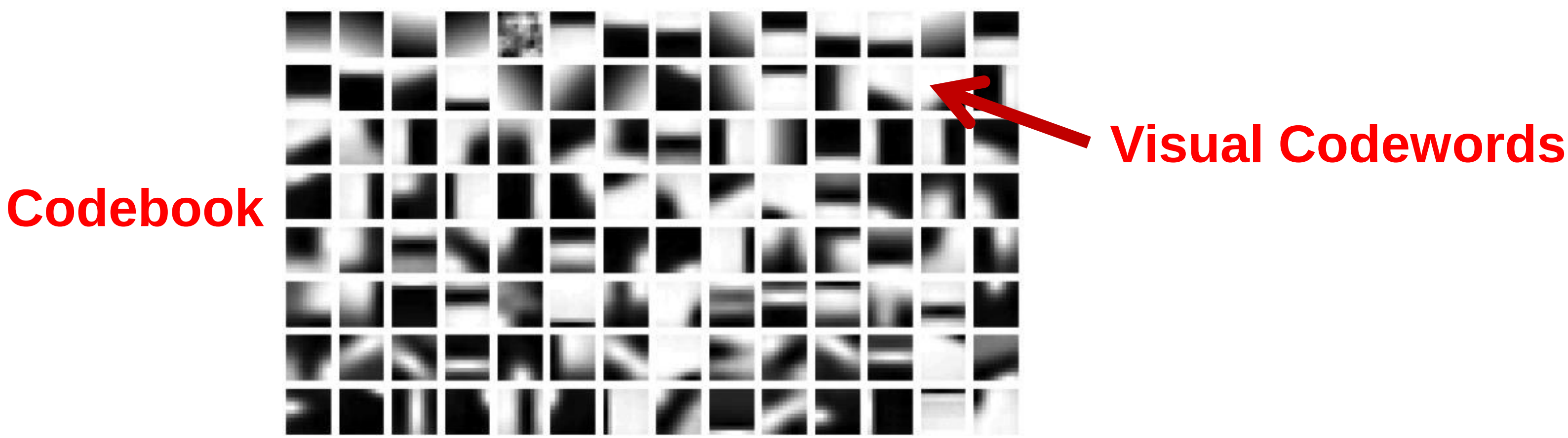


Building a Visual Codebook (Dictionary)

- Visual words (real-valued vectors) can be compared using Euclidean distance:

$$E(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\sum_d (\mathbf{x}_1^d - \mathbf{x}_2^d)^2}$$

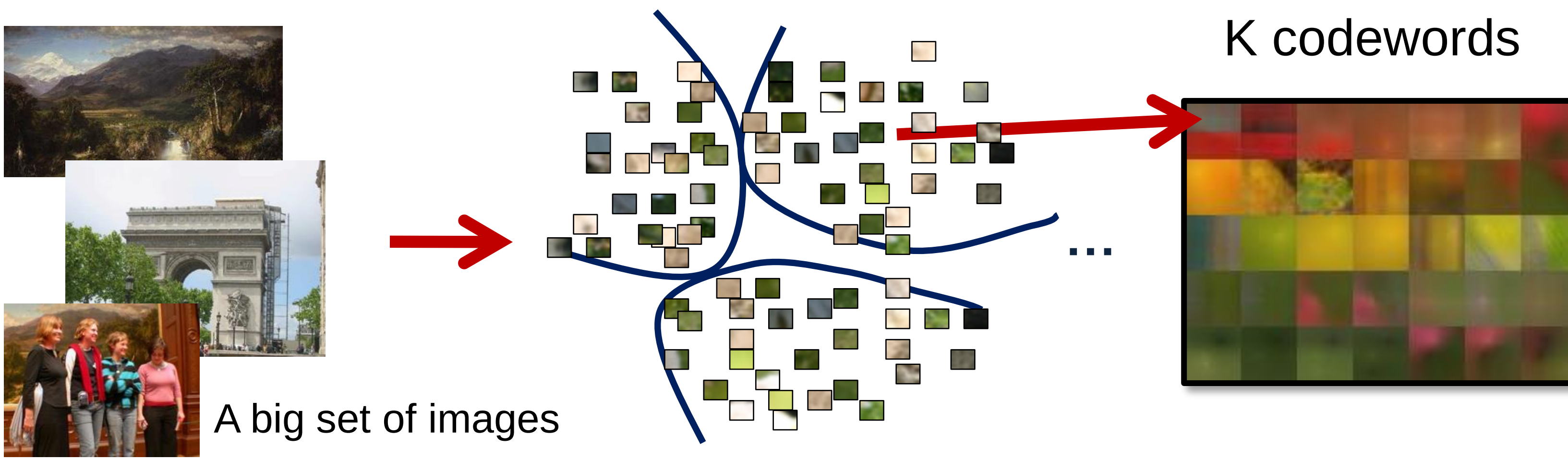
- These vectors are divided into groups which are similar, essentially clustered together, to form a codebook.



K-means Clustering for Codebook

Lecture 3-4

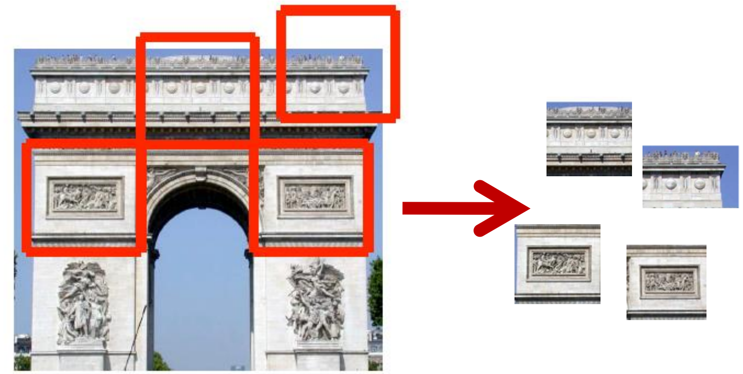
- A large number of image patches are collected from a representative set of images.
- The two steps repeat until no change in membership:
  1. Compute the center of each cluster as the data mean of the respective cluster
  2. Reassign each data point to the cluster whose center is nearest
- The cluster centers (mean vectors) form a visual dictionary.



Histogram of Visual Words

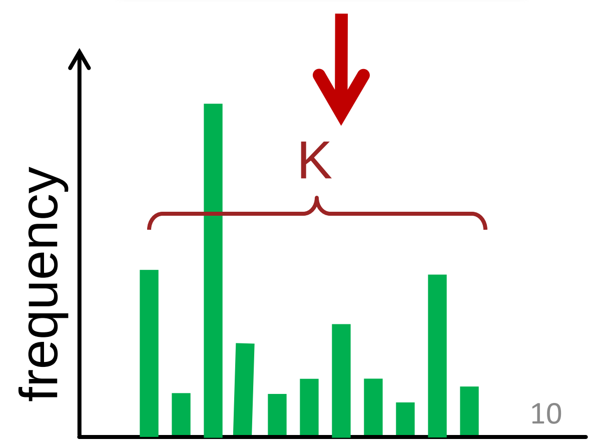
- Every visual word is compared with codewords and assigned to the nearest codeword.

- Histogram bins are codewords and each bin counts the number of words assigned to the codeword.

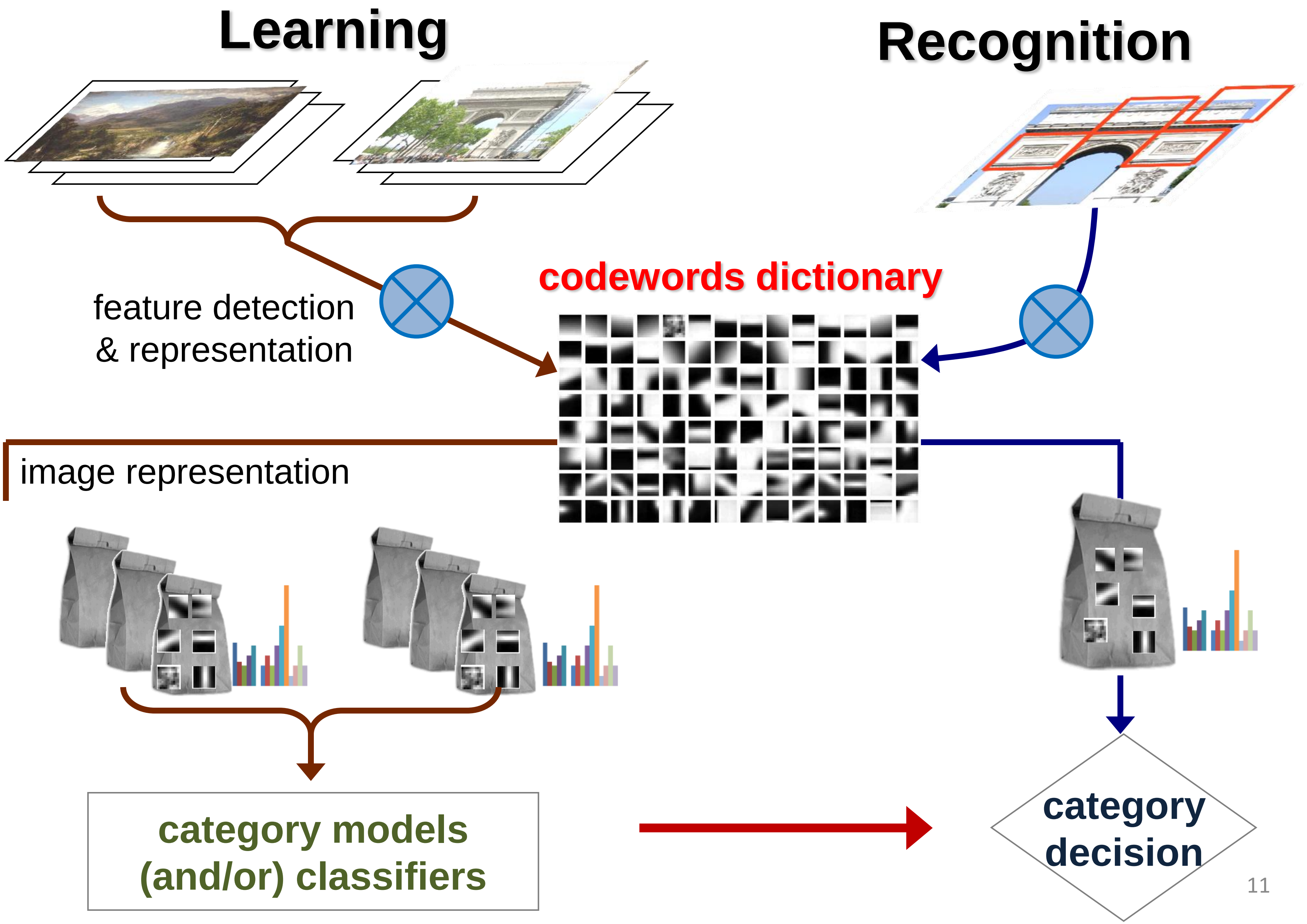


Nearest Neighbour Matching

“bag of words”



Categorisation



Summary of K-means Codebook

- The histogram is K-D, a highly compact and robust representation of images.
- The histogram representation greatly facilitates modelling images and categorising them.
- Codeword assignment (quantisation process) by K-means is time-demanding.
- K-means is an unsupervised learning method.

Sparse Kernel Machine (Maximum Margin Classifier or Support Vector Machine)

$$f : \mathbf{x} \in \mathbb{R}^D \rightarrow t = \{1, \ldots, n\}$$

Linear Discriminant Functions

- A discriminant function maps an input vector **x** into one of K classes, denoted C_k. For simplicity, consider two classes.

- *Linear* discriminant function takes the form of

$$y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$$

where **w** is a *weight vector* and w_o a *bias*.

- A vector **x** is assigned to C_1 if y(**x**) ≥ 0, and C_2 otherwise.

- The *decision boundary* is defined by $y(\mathbf{x}) = 0$, which is a (D-1)-dimensional hyperplane in the D-dimensional input space.

- Consider $\mathbf{x}_A$ and $\mathbf{x}_B$ on the decision surface.

$$\rightarrow \quad y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0$$

$$\rightarrow \quad \mathbf{w}^T(\mathbf{x}_A - \mathbf{x}_B) = 0$$

The vector **w** is orthogonal to the decision surface. **w** determines the ***direction*** of the decision surface.

- For **x** on the decision surface, $y(\mathbf{x}) = 0$, so the normal distance from the origin to the decision surface is

$$\frac{\mathbf{w}^T\mathbf{x}}{||\mathbf{w}||} = -\frac{w_0}{||\mathbf{w}||}$$

Thus, w_0 determines the ***location*** of the decision surface.

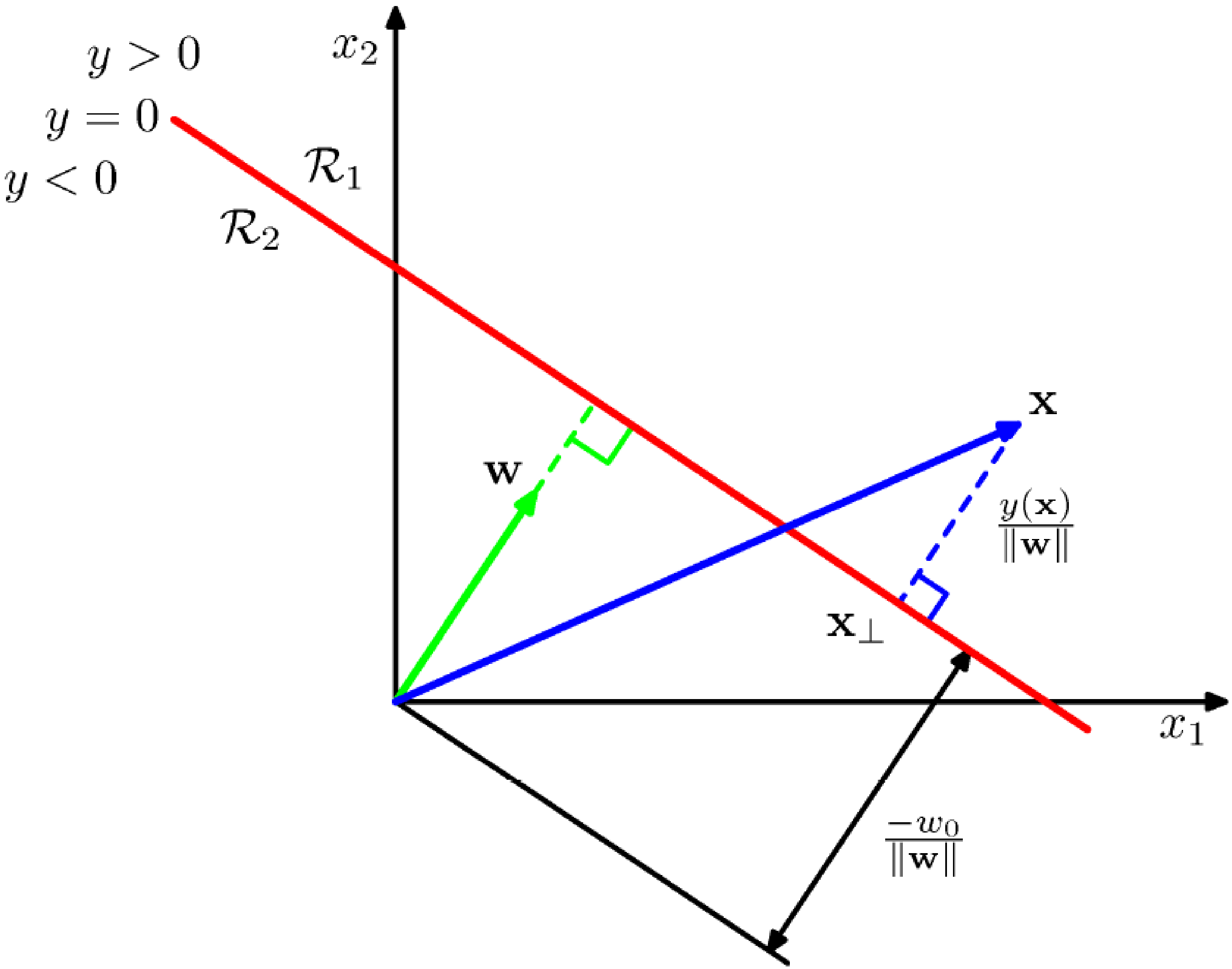
- For an arbitrary point $\mathbf{x}$, its orthogonal projection onto the decision surface $\mathbf{x}_\perp$ is such that

$$\mathbf{x} = \mathbf{x}_\perp + r\frac{\mathbf{w}}{||\mathbf{w}||}.$$

- Multiply both sides by $\mathbf{w}^T$ and add w_0, and

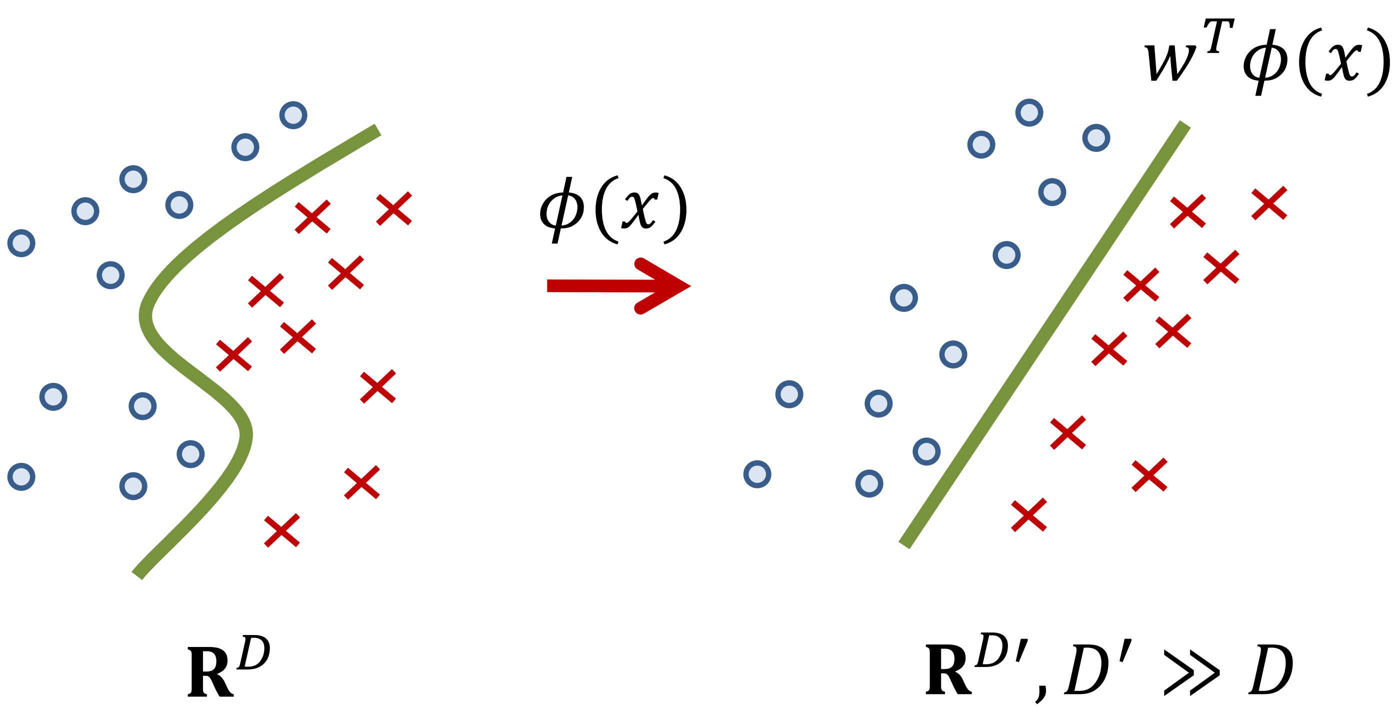
use $y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$ and $y(\mathbf{x}_\perp) = \mathbf{w}^T\mathbf{x}_\perp + w_0 = 0$, we have

$$r = \frac{y(\mathbf{x})}{||\mathbf{w}||}.$$


$y > 0$
$y = 0$
$y < 0$
x_2
$\mathcal{R}_1$
$\mathcal{R}_2$
$\mathbf{x}$
$\mathbf{w}$
$\frac{y(\mathbf{x})}{\|\mathbf{w}\|}$
$\mathbf{x}_\perp$
x_1
$\frac{-w_0}{\|\mathbf{w}\|}$

Kernel Trick

- We often need a *nonlinear* decision boundary.
- The input space is transformed into *a high-dimensional feature space* by a nonlinear mapping $\phi : \mathbf{x} \rightarrow \phi(\mathbf{x})$, where a linear decision boundary can separate all data points.



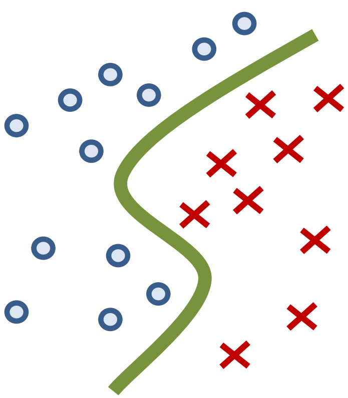
Kernel trick:

The concept of a kernel formulated as an inner product in a feature space allows to build nonlinear extensions of many ML algorithms.

- Many ML problems can be recast into an equivalent *dual representation*,
  where the predictions are based on linear combinations of a *kernel function* evaluated at training data points.

$$k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T \phi(\mathbf{x}')$$

- The kernel is a symmetric function $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$



$w^T \phi(x)$

$\in \mathbf{R}^{D'}, D' \gg D$

$D' \times N$

a form of $\phi(x)^T \phi(x')$

$N \times N$

D : input dimension
D': kernel space dimension
N : number of data points

- An example is

$$k(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^T \mathbf{z})^2.$$

If **x** = (x_1 ,x_2), **z** = (z_1 ,z_2),

$$k(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^T \mathbf{z})^2 = (x_1 z_1 + x_2 z_2)^2$$

$$= x_1^2 z_1^2 + 2x_1 z_1 x_2 z_2 + x_2^2 z_2^2$$

$$= \left(x_1^2, \sqrt{2}x_1 x_2, x_2^2\right) \left(z_1^2, \sqrt{2}z_1 z_2, z_2^2\right)^T$$

$$= \phi(\mathbf{x})^T \phi(\mathbf{z}).$$

The feature mapping takes the form $\phi(\mathbf{x}) = (x_1^2, \sqrt{2}x_1 x_2, x_2^2)^T$ comprising all second order terms.

- A necessary and sufficient condition for a valid kernel $k(\mathbf{x}, \mathbf{x}')$:

$$\mathbf{x}^T\mathbf{K}\mathbf{x} \geq 0$$

where $\mathbf{K}$ is the Gram matrix whose elements are $k(\mathbf{x}_i, \mathbf{x}_j')$ i.e. $\in \mathbf{R}^{N \times N}$ (N is the number of training data points). **K** should be positive semidefinite.

- *Typical examples of Kernels:*

Linear kernel: $$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}'$$

Polynomial kernel: $$k(\mathbf{x}, \mathbf{x}') = \left(\mathbf{x}^T\mathbf{x}' + c\right)^M \text{ with } c > 0$$

Gaussian kernel: $$k(\mathbf{x}, \mathbf{x}') = \exp\left(-||\mathbf{x} - \mathbf{x}'||^2/2\sigma^2\right)$$

Maximum Margin Classifiers

- *Sparse Kernel Machine*: has sparse solutions i.e. the predictions for new inputs depend only on the kernel function evaluated at a subset of the training data points, which are called support vectors.

- *Support Vector Machine (SVM):* is a decision machine, which is obtained by maximising the margins, thus it is also called maximum margin classifier.

- We have N training data vectors $\mathbf{x_1}, \ldots, \mathbf{x}_N$ and their target values $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_N$ where $\boldsymbol{t}_N \in \{-1, 1\}$.

- **SVM** for the two-class problem takes the form of

$$y(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$$

where $\phi(\mathbf{x})$ denotes a feature mapping, b the bias.

- Data points $\mathbf{x}$ are classified by the sign of $y(\mathbf{x})$.

- Assume that the training data is linear separable in feature space. Thus, optimal **w** and b satisfy

$$t_n y(\mathbf{x}_n) > 0$$

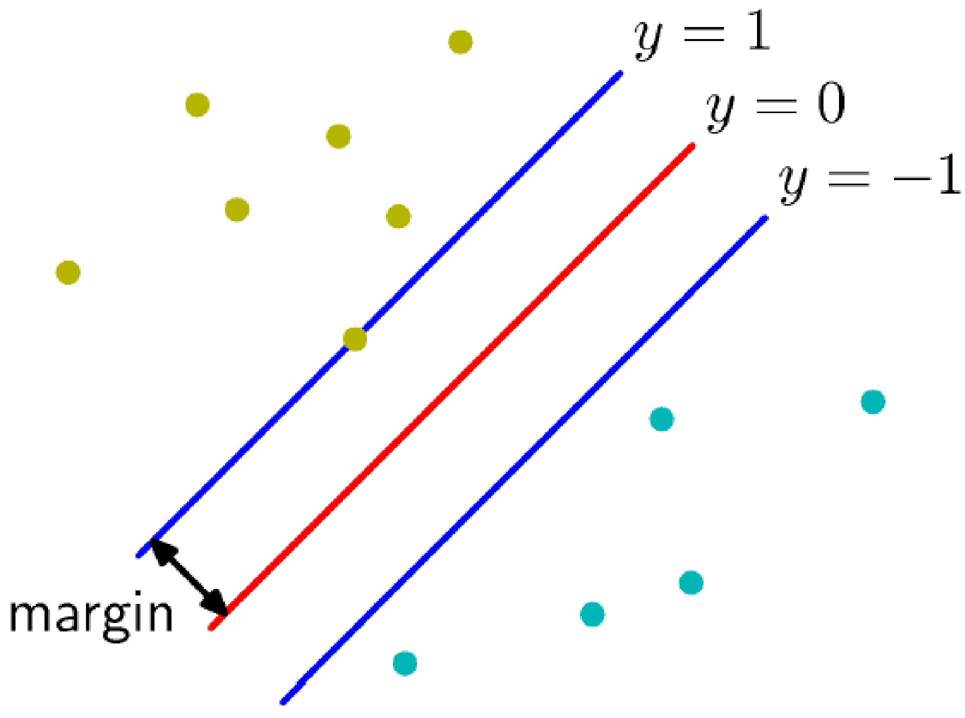
for all training data points.

- The perpendicular distance of a point **x** from a hyperplane $y(\mathbf{x})$ is $|y(\mathbf{x})|/ ||w||$.
- As we assume $t_n y(\mathbf{x}_n) > 0$ for all n, the distance of a point $\mathbf{x}_n$ to the decision surface is

$$\frac{t_n y(\mathbf{x}_n)}{||\mathbf{w}||} = \frac{t_n(\mathbf{w}^T \phi(\mathbf{x}_n) + b)}{||\mathbf{w}||}$$

- The margin is the minimum perpendicular distance.
- We find **w** and b that maximise the margin i.e.

$$\arg\max_{\mathbf{w},b} \left\{ \frac{1}{||\mathbf{w}||} \min_n [t_n(\mathbf{w}^T \phi(\mathbf{x}_n) + b)] \right\}$$


$y = 1$
$y = 0$
$y = -1$
margin

Dual representation

- Canonical representation of the decision hyperplane:

  Rescaling **w** -> *k***w** and *b* -> *kb* does not change the distance from any point $\mathbf{x}_n$ to the decision surface. We can therefore set

$$t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) = 1$$

for the point that is closest to the surface. All data points satisfy

$$t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) \geq 1, \quad n = 1, ..., N.$$

Dual representation

- The optimisation problem becomes to maximise $||w||^{-1}$. Equivalently, we have

$$\arg\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2$$

subject to the constraints $t_n(\mathbf{w}^T\phi(\mathbf{x})+b) \geq 1$

- The problem is solved by introducing Lagrange multipliers $a_n \geq 0$,

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{n=1}^{N} a_n\{t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) - 1\}$$

where a = $(a_1,...,a_N)^T$. Note the minus sign in front of the Lagrange multiplier term, as we are minimising the function.

max f(x) s.t. g(x)=0 $\rightarrow$ max f(x) + λg(x)

http://en.wikipedia.org/wiki/Lagrange_multiplier

Dual representation

- Setting the derivatives of *L*(**w**, *b*, a) w.r.t. **w**, *b* to zero, we obtain

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n) \qquad 0 = \sum_{n=1}^{N} a_n t_n.$$

Using the two conditions above, we eliminate **w** and *b* to get the ***dual representation*** as

$$\widetilde{L}(\mathbf{a}) = \sum_{i=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m \boxed{k(\mathbf{x}_n, \mathbf{x}_m)}$$

with the constraints $a_n \geq 0, \quad n = 1, ..., N$

$$\sum_{n=1}^{N} a_n t_n = 0.$$

where $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T \phi(\mathbf{x}').$

Dual representation

- The SVM decision function takes the form of

$$y(\mathbf{x}) = \sum_{i=1}^{N} a_n t_n \boxed{k(\mathbf{x}, \mathbf{x}_n)} + b.$$

- *Karush-Kuhn-Tucker (KKT) conditions* of the Lagrangian function are

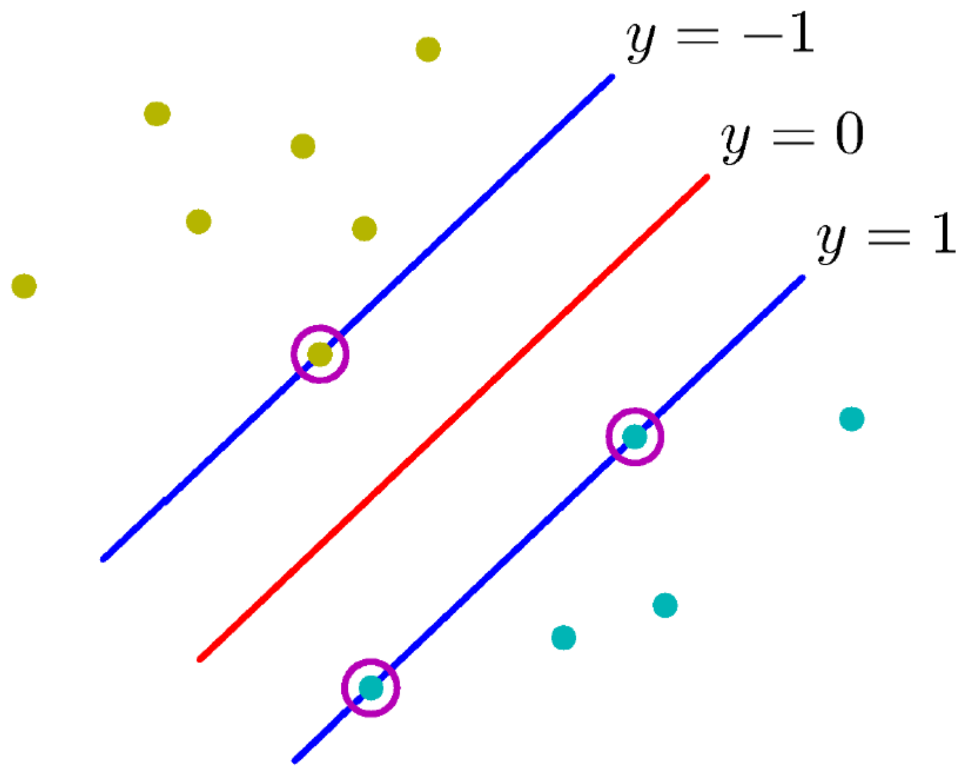
$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x}). \rightarrow \begin{aligned} g(\mathbf{x}) &\geq 0, \\ \lambda &\geq 0, \\ \lambda g(\mathbf{x}) &= 0. \end{aligned}$$

- The KKT conditions of the maximum margin problem are

$$a_n \geq 0$$
$$t_n y(\mathbf{x}_n) - 1 \geq 0$$
$$a_n \{t_n y(\mathbf{x}_n) - 1\} = 0.$$

$\rightarrow$

For every data point, a_n=0 or $t_n y(\mathbf{x}_n)$ =1.
Any data point for which a_n=0 plays no role in making predictions.
The remaining data points are called *support vectors*, and because they satisfy $t_n y(\mathbf{x}_n)$ =1, i.e. lying on the margin hyperplanes.

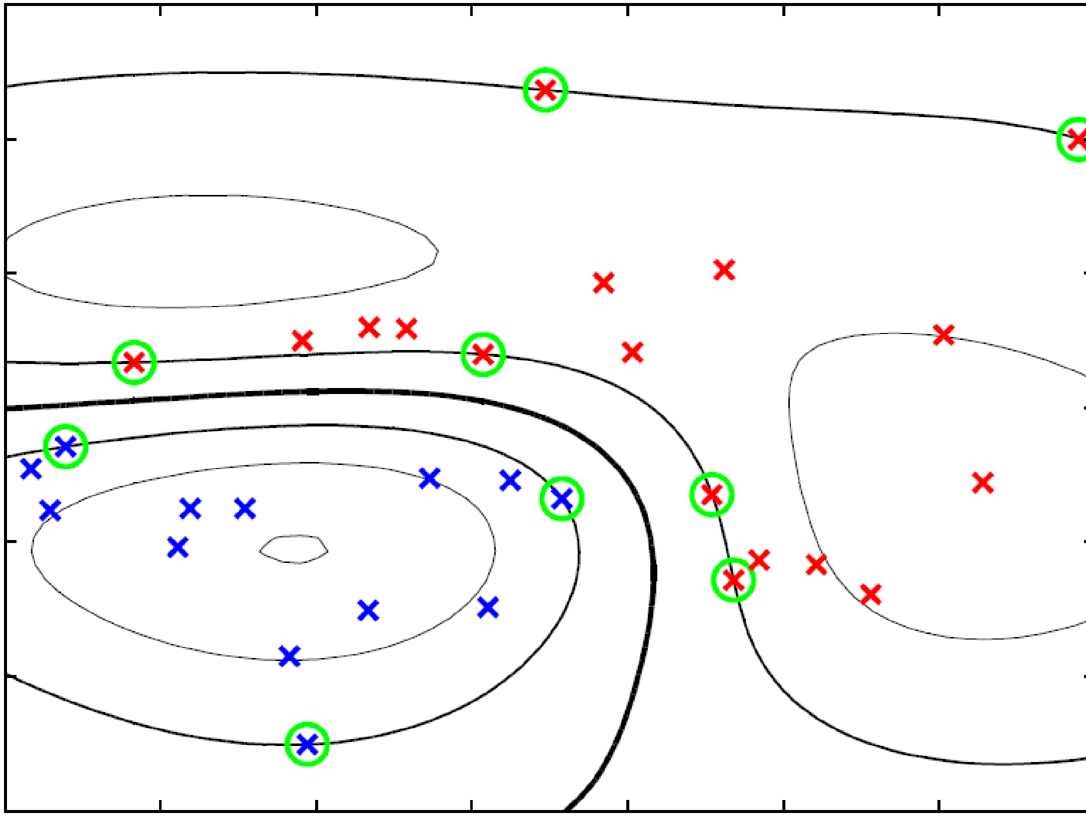

$y = -1$
$y = 0$
$y = 1$

Optimisation

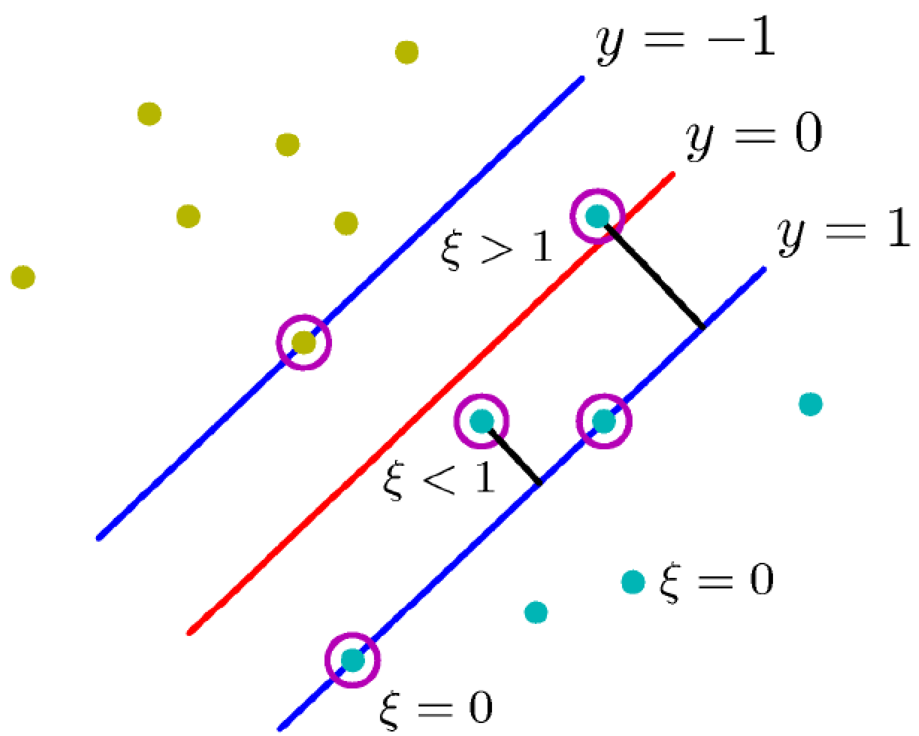
- *Sequential minimal optimisation (SMO):*

  see e.g.

CHRISTOPHER J.C. BURGES, A Tutorial on Support Vector Machines for Pattern Recognition (http://research.microsoft.com/pubs/67119/svmtutorial.pdf)



Overlapping class distributions



Overlapping class distributions

- *Slack variables*, $\xi_n \geq 0$, n=1,...,N are defined s.t.

$$t_n y(\mathbf{x}_n) \geq 1 - \xi_n, \quad n = 1, ..., N.$$

$\xi_n = 0$ for data points that are on or inside the correct margin boundary and $\xi_n = |t_n - y(\mathbf{x}_n)|$ for other points.

- We therefore minimise

$$C \sum_{n=1}^{N} \xi_n + \frac{1}{2}||\mathbf{w}||^2$$

where $C > 0$ is a trade-off constant and $\xi_n \geq 0$, subject to

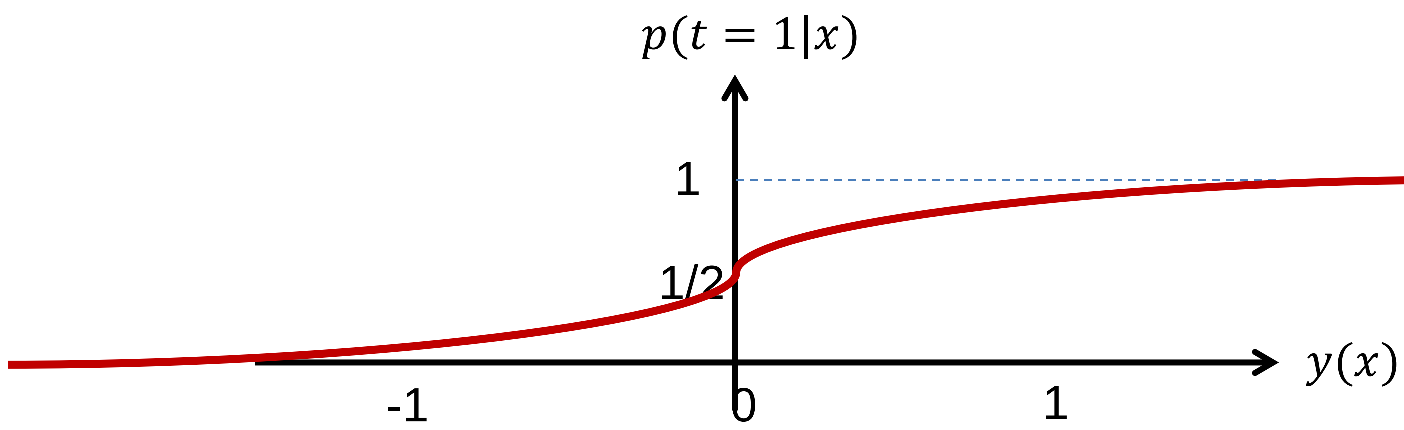
$$t_n y(\mathbf{x}_n) \geq 1 - \xi_n, \quad n = 1, ..., N.$$

Probabilistic output

- We fit a logistic sigmoid to the output of SVM. The conditional probability takes the form of

$$p(t = 1|\mathbf{x}) = \sigma(Ay(\mathbf{x}) + B)$$

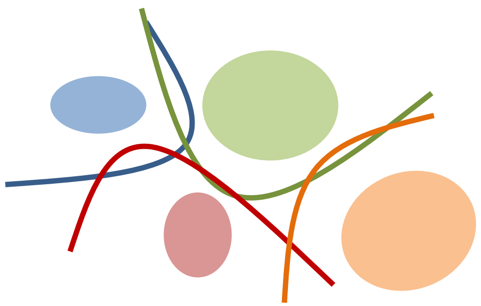
Where *A*, *B* are parameters and σ is the sigmoid function.



Multiclass SVMs 1)

One-versus-the-rest trains K separate SVMs, where $y_k(\mathbf{x})$ is trained by C_k as the positive class and the remaining K-1 classes as the negative class.

The prediction for a new input **x** is by



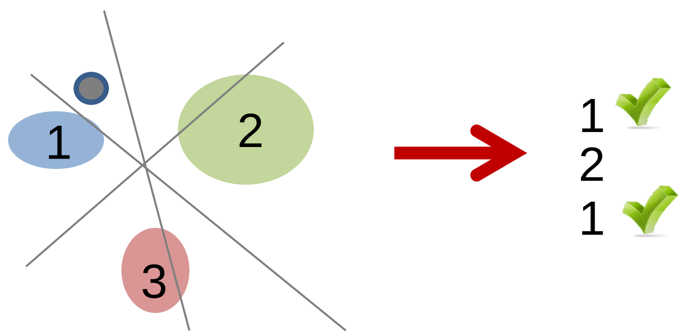
$$y(\mathbf{x}) = \max_k y_k(\mathbf{x}).$$

Problems: 1) the output values $y_k(\mathbf{x})$ for different classifiers have no appropriate scales. 2) the training sets are imbalanced.

Multiclass SVMs 2)

One-versus-one trains *K(K-1)/2* different 2-class SVMs on all possible pairs of classes.

Classification of a test point is by, which class has the highest number of 'votes'.



Problems: it requires more training time and evaluation time.

Statistical Pattern Recognition Toolbox for Matlab

http://cmp.felk.cvut.cz/cmp/software/stprtool/

BoW as input to SVM classifier

- Treat BoW histograms as feature vectors for standard classifiers
  – e.g SVM
- SVM for object classification
  – Csurka, Bray, Dance & Fan, 2004

Video-based Object Recognition

Imperial College, KAIST

Video-based Object Recognition
by
Sets of Sets