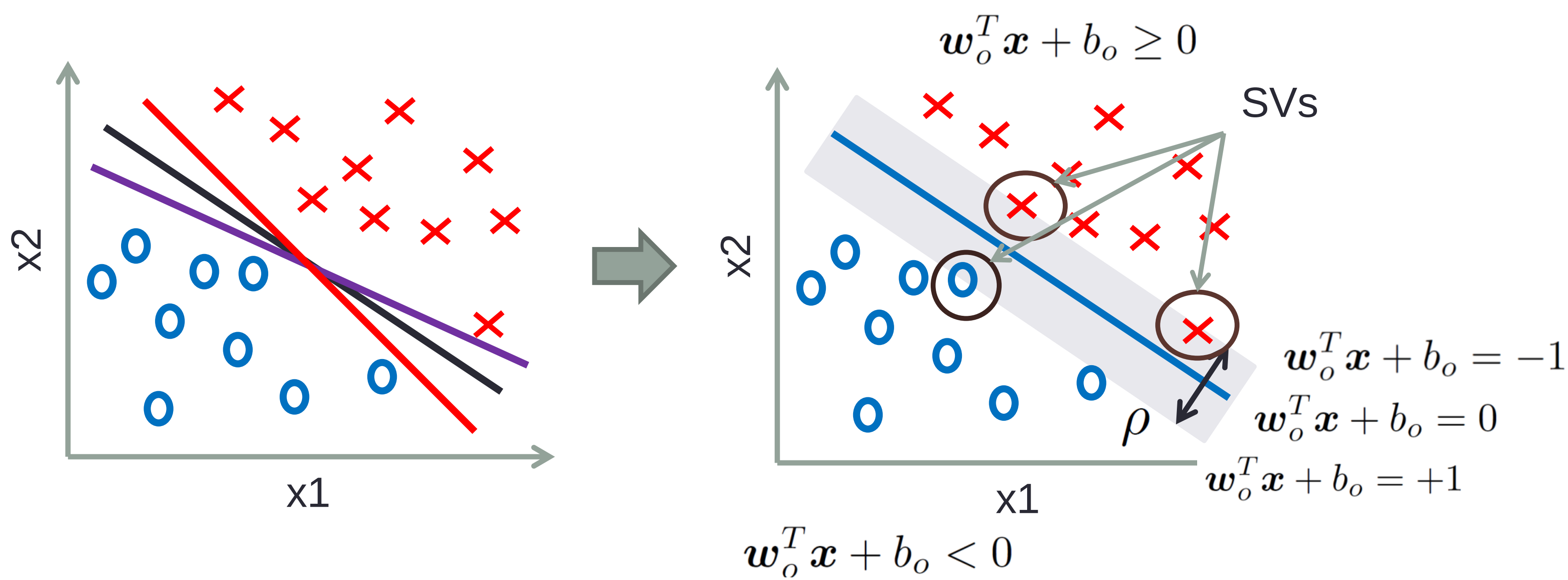
Support Vector Machine

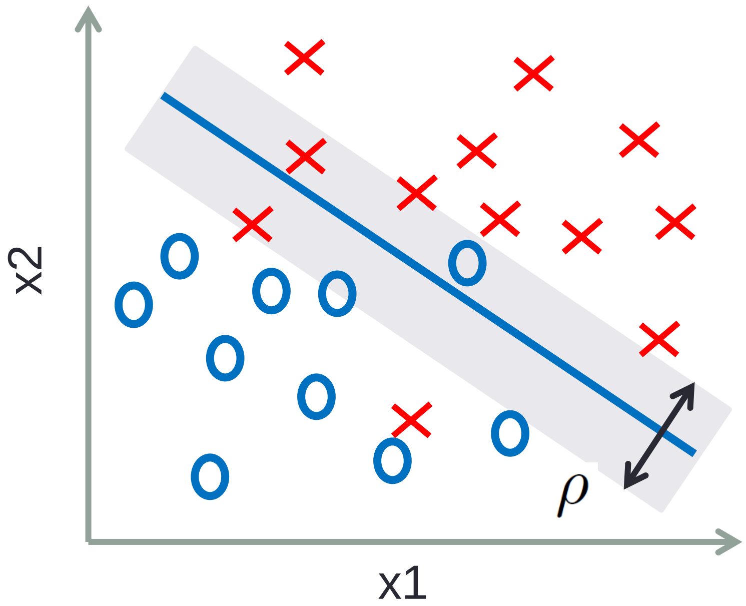
- Consider a training set consisting of *m* samples:

$$\langle \boldsymbol{x}^{(1)}, y^{(1)} \rangle \ldots \langle \boldsymbol{x}^{(m)}, y^{(m)} \rangle \quad \text{where} \quad y^{(i)} \in \{-1, 1\}$$



Goal: minimize $\frac{1}{2}||\boldsymbol{w}||^2$

Support Vector Machine with soft margin

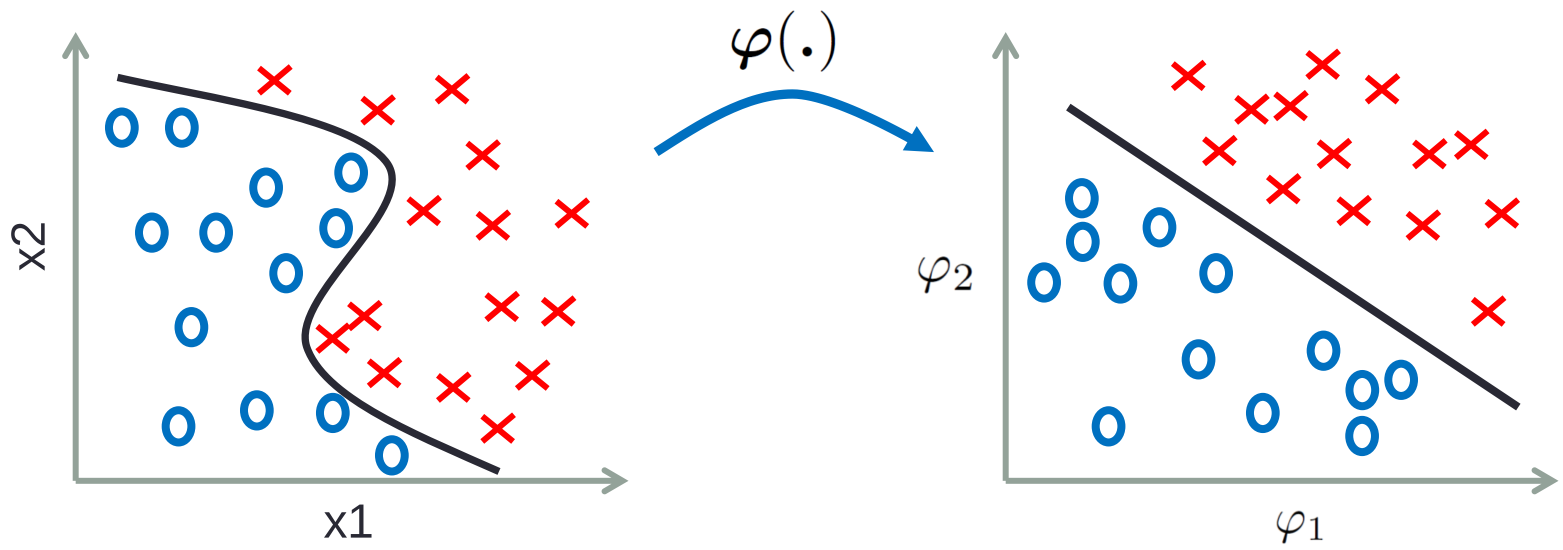


Goal:

$$arg \min_{\boldsymbol{w}} \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{m} \xi_i$$

- The free parameter C controls the relative importance of minimizing the norm $||\boldsymbol{w}||$ and satisfying the margin constraint for each sample

Kernels

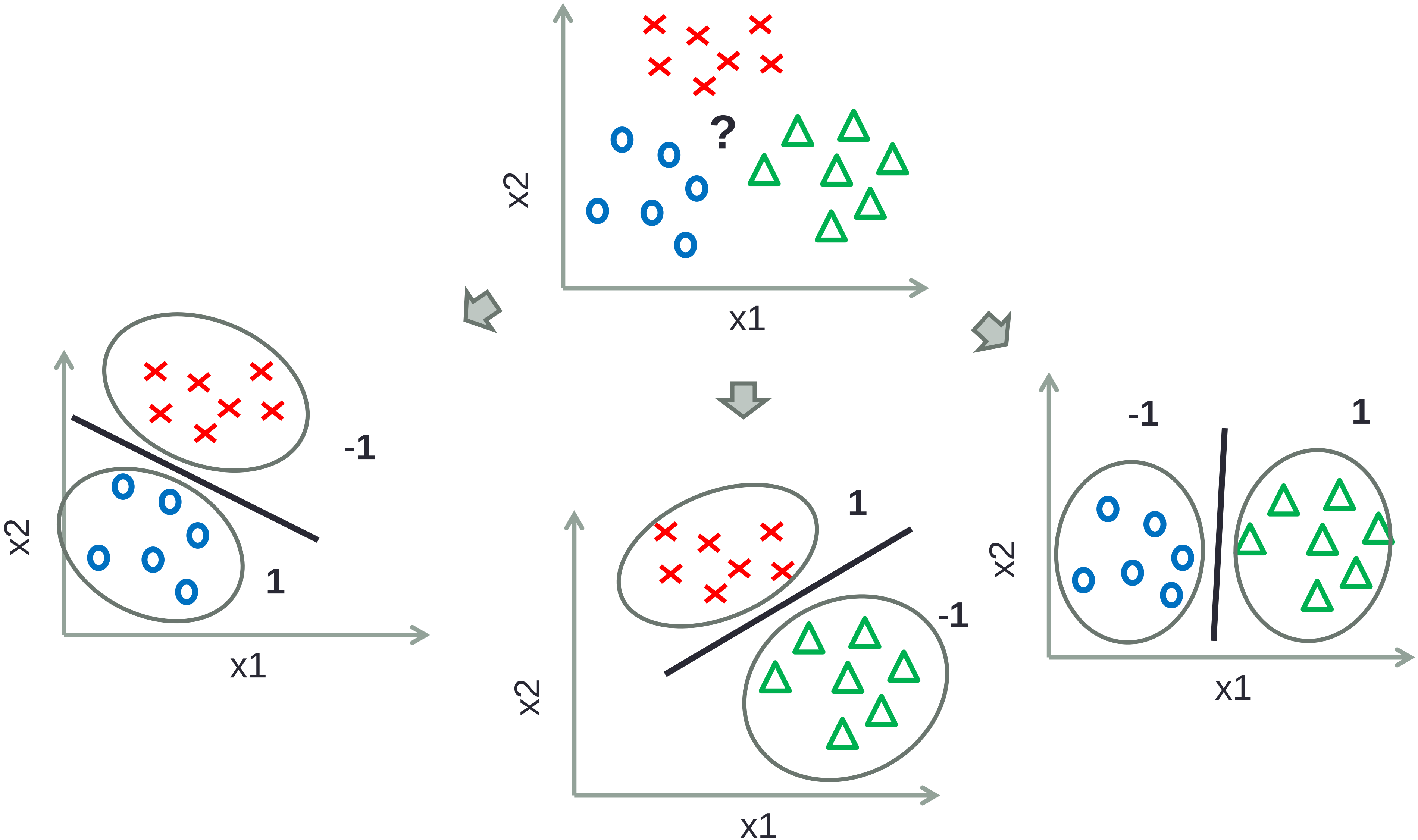


$$K(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{\varphi}(\boldsymbol{x}_1)^T \boldsymbol{\varphi}(\boldsymbol{x}_2)$$

RBF kernel:

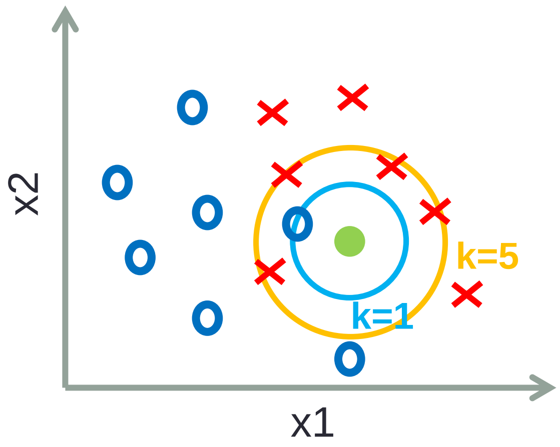
$$K(\boldsymbol{x}, \boldsymbol{y}) = exp(-\frac{||\boldsymbol{x} - \boldsymbol{y}||^2}{2\sigma^2})$$

Multiclass classification: One-vs-one



k-Nearest Neighbor (k-NN)

- The main idea:
  - For a new sample examine the ***k*** closest training samples according to some distance metric (most common: Euclidean distance)
  - Assign the new sample to the most frequently occurring class within those ***k*** samples



SVM & k-NN in MATLAB

- **SVM:**
  - boxconstraint is the free parameter C
  - Most important functions: svmtrain, svmclassify

svm = svmtrain (X, Y, ‘ShowPlot’, true, ‘boxconstraint’, 1, ‘Kernel_Function’,’linear‘);

prediction = svmclassify (svm, X, ‘ShowPlot’, true);

- **k-NN:**
  - k is the number of neighbors

prediction = knnclassify (Xtest, X, Y, k);

- See demo script on the course website