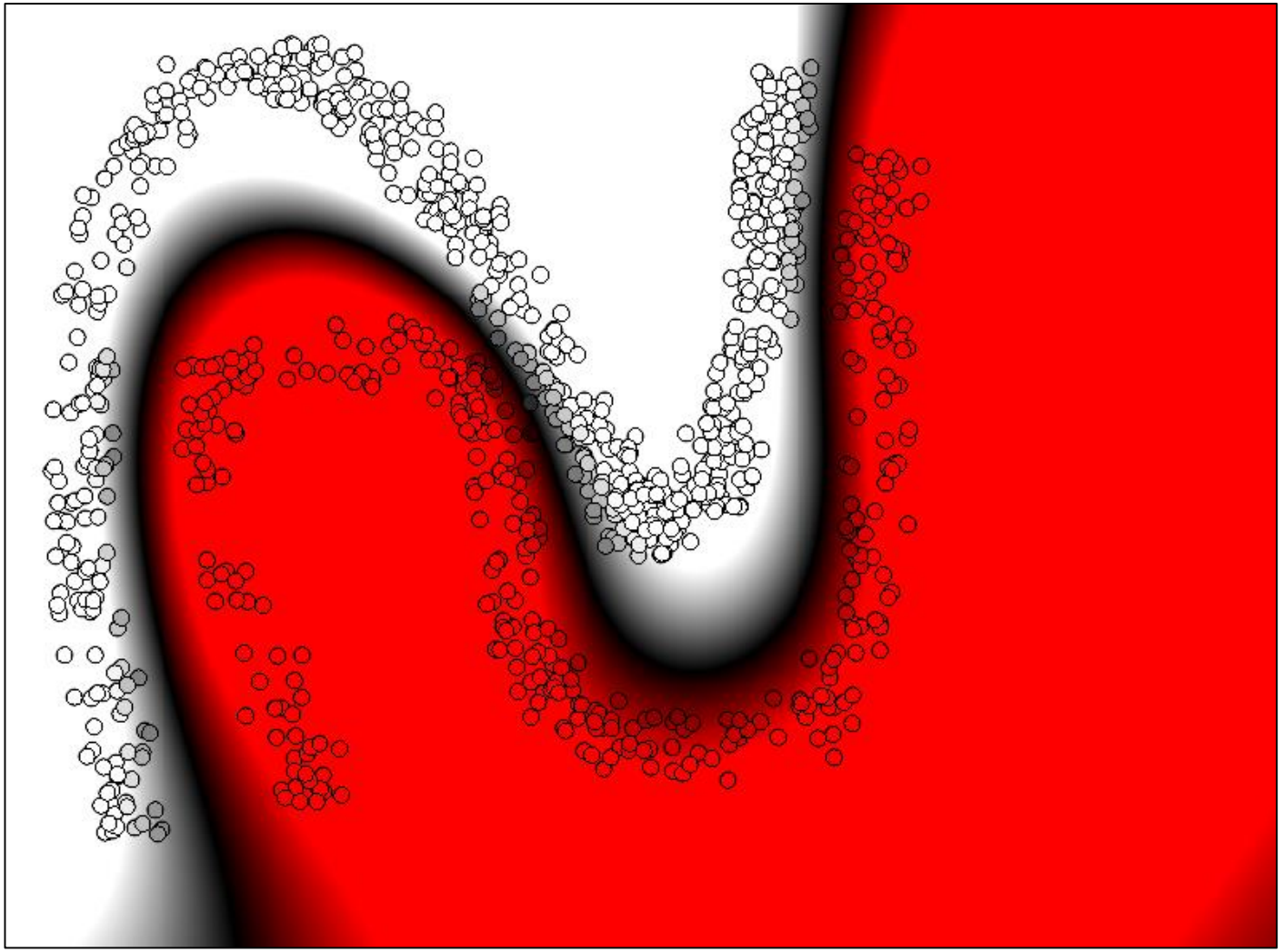
MACHINE LEARNING

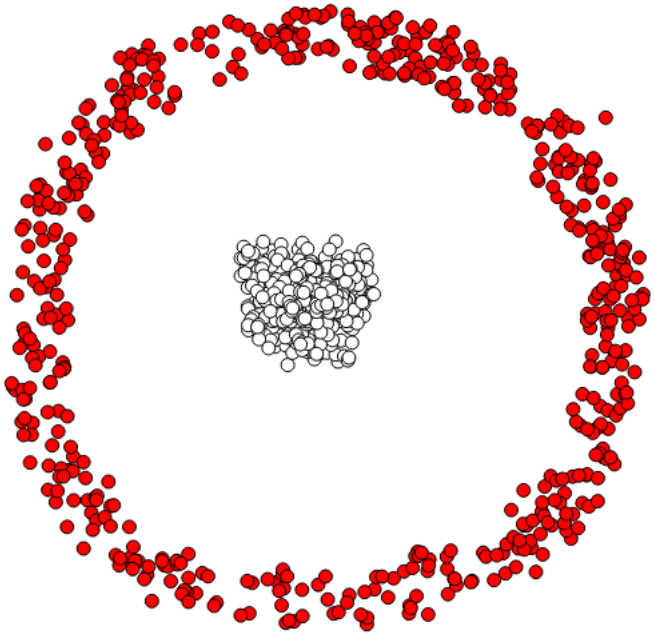
Support Vector Machine
For Classification

Part 2 – Non-Linear SVM

Non-linear classification



Non-linear classification

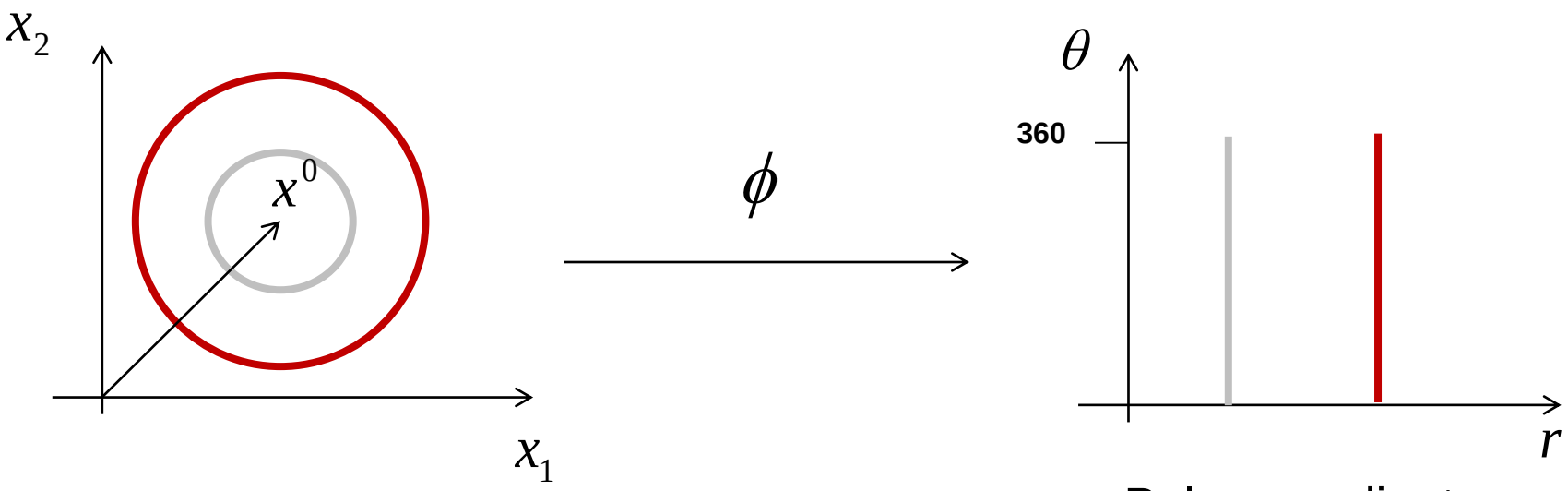


Can we separate these two groups linearly?

Yes, but not in this space.

Making the problem linear

How to separate the red class from the grey class?



Polar coordinates

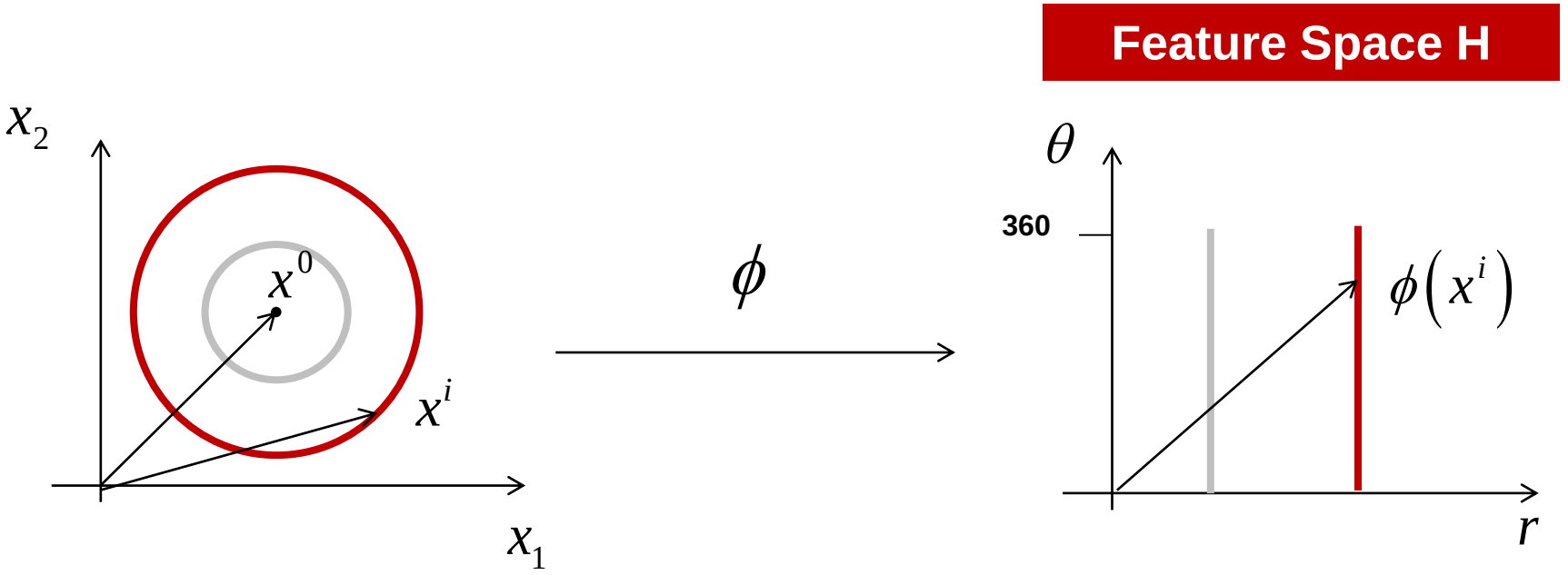
$$x = \left(r\sin(\theta), r\cos(\theta)\right) + x^0$$

Data become linearly separable

$$\phi: \mathbb{R}^2 \rightarrow \mathbb{R}^2$$

$$\phi(x) = (r, \theta), \quad r: \text{ radius}, \ \theta: \text{ angle}$$

Making the problem linear



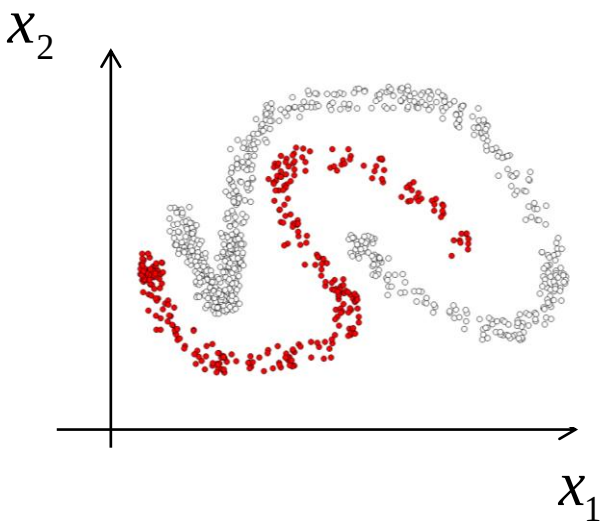
Idea: Send the data *X* into a *feature* space *H* through the nonlinear map ϕ.

$$X = \left\{ x^i \in \mathbb{R}^N \right\}^{i=1...M} \mapsto \phi(X) = \left(\phi(x^1), ..., \phi(x^M) \right)$$

In feature space, computation is simpler (e.g. perform linear classification)

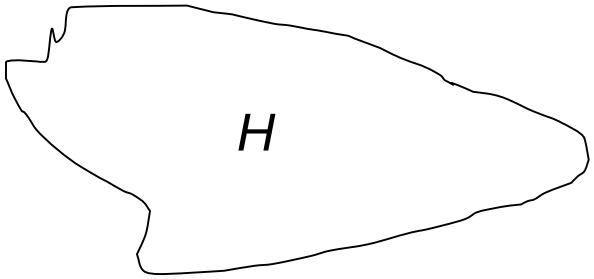
Finding the transformation

Original Space



What is ϕ?

Feature Space H



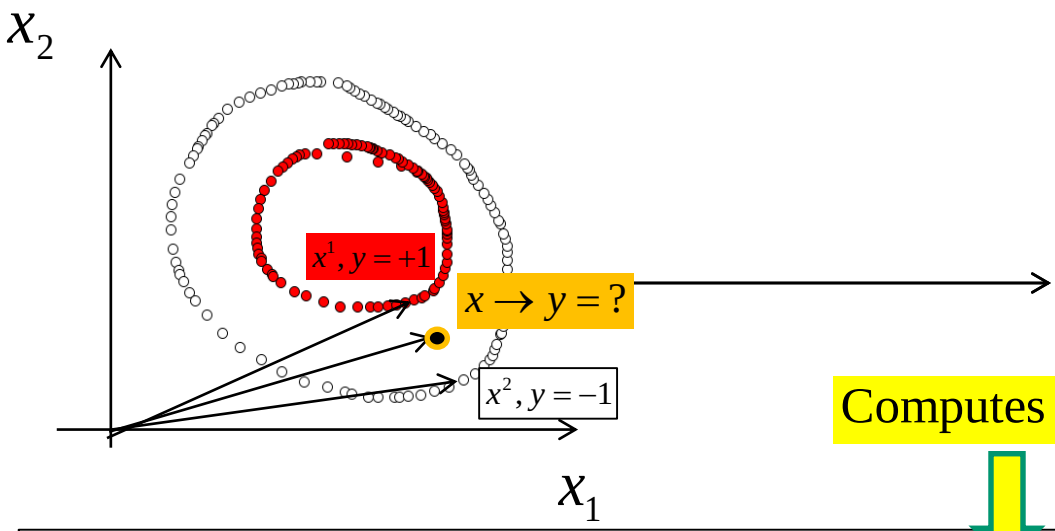
While the dimension of the original space is N, the dimension of the feature space may be greater than N! ➔ *X is lifted onto H*

Determining ϕ is difficult and sometimes impossible

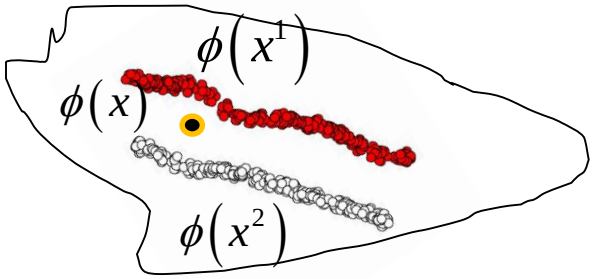
No need to compute ϕ, sufficient to compute distance in feature space

Kernels: Intuition

Original Space



Feature Space H



Computes distance

The kernel function

$k : X \times X \rightarrow \mathbb{R}$

$k(x^i, x^j) \rightarrow \langle \phi(x^i), \phi(x^j) \rangle.$

SVM Decision function $f(x) = \text{sgn}\left(\sum_{i=1}^{M} \alpha_i y^i \langle x, x^i \rangle + b \right)$

SVM Decision function $f(x) = \text{sgn}\left(\sum_{i=1}^{M} \alpha_i y^i \langle \phi(x), \phi(x^i) \rangle + b \right)$

No need to compute ϕ, sufficient to compute distance in feature space

Rephrase linear SVM

Linear problem expressed in feature space:

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

Under constraints: $y^i \left(\left\langle w, \phi\left(x^i\right) \right\rangle + b \right) \geq 1$, i=1,2,....,M.

w lives in feature space!

The constraints can be relaxed to avoid overfitting.

This can be solved using the Lagrange method for inequality constraints:

$$L(w,b,\alpha) \equiv \frac{1}{2}\|w\|^2 - \sum_{i=1}^{M} \alpha_i \left(y_i \left(\left\langle w, \phi\left(x^i\right) \right\rangle + b \right) - 1 \right)$$

with $\alpha_i \geq 0$

Rephrase linear SVM

We obtain the Dual Form :

Inner product

$$\max_{\alpha} L_D(\alpha) \equiv \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \left\langle \phi(x^i), \phi(x^j) \right\rangle$$

Subject to these constraints:

$$0 \le \alpha_j \le \frac{C}{M} \quad \forall j = 1,\ldots,M \qquad \sum_{j=1}^{M} a_j y_j = 0$$

The computation relies only on inner products across image of the points in feature space → replace with the kernel.

Dual Optimization

With the kernel, the dual becomes :

kernel

$$\max_{\alpha} L_D(\alpha) \equiv \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j k\left(x^i, x^j\right)$$

Subject to these constraints:

$$0 \le \alpha_j \le \frac{C}{M} \quad \forall j = 1,\ldots,M \qquad \sum_{j=1}^{M} a_j y_j = 0$$

The decision function becomes:

$$f(x) = sgn\left(\sum_{i=1}^{M} \alpha_i y_i k\left(x, x^i\right) + b\right)$$

See supplement for derivation

The kernel

RBF (Gaussian) kernel: $k\left(x, x^i\right) = e^{-\frac{\left\|x - x^i\right\|}{2\sigma^2}}, \quad \sigma \in \mathbb{R}.$

Kernel width

Order p of the polynomial and c

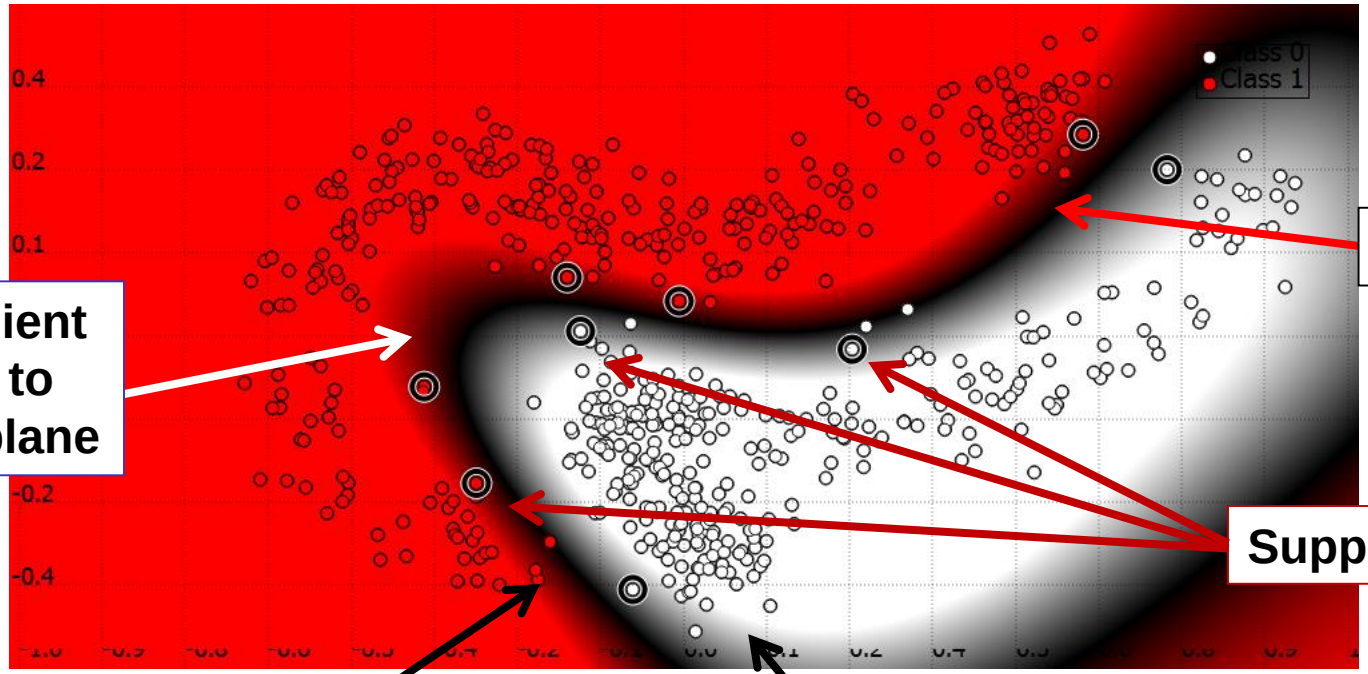
Inhomogeneous polynomial kernel: $k\left(x, x^i\right) = \left(\left\langle x, x^i \right\rangle + c\right)^p, \quad p \in \mathbb{N}, \; c \geq 0$

Each kernel has hyperparameters that need to be determined by hand (grid search).

The decision function becomes:

$$f\left(x\right) = sgn\left(\sum_{i=1}^{M} \alpha_i y_i k\left(x, x^i\right) + b\right)$$

How to read out the result of SVM



Color gradient = distance to the hyperplane

Hyperplane

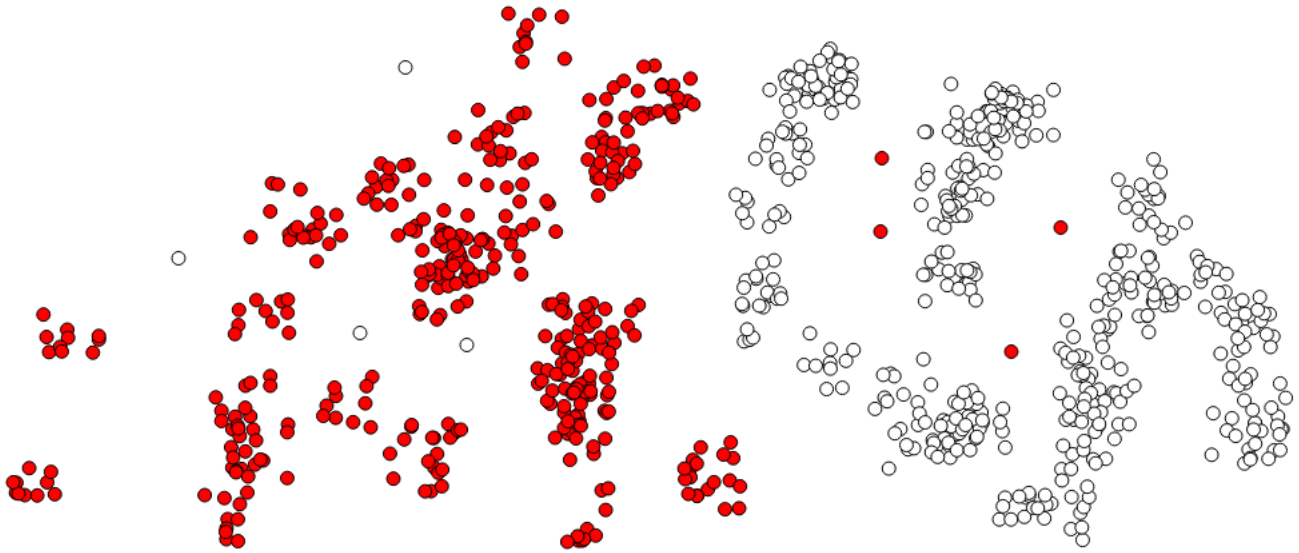
Support vectors

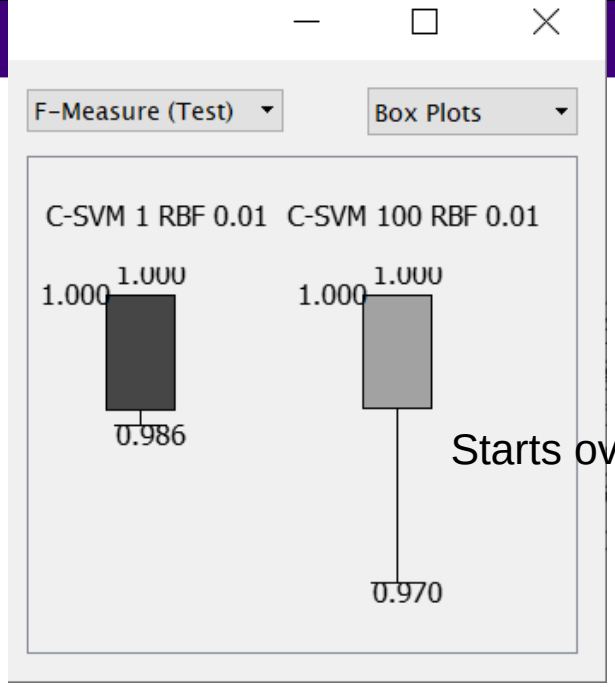
The margin: region of equal distance on both side of hyperplane

Effect of the hyperparameters on performance in classification with SVM

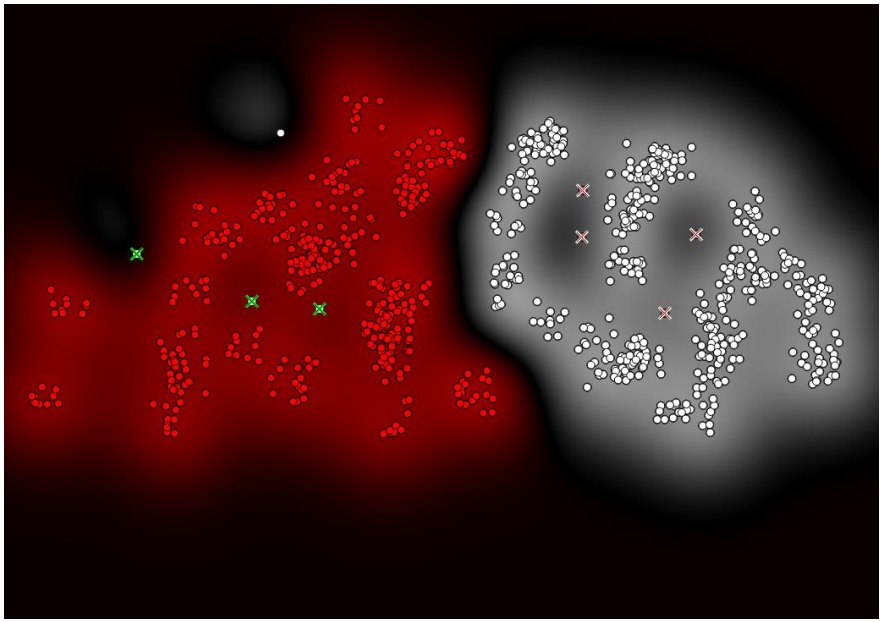
$$\min_{w,\xi}\left(\frac{1}{2}\|w\|^2 + \frac{C}{M}\sum_{j=1}^{M}\xi_j\right)$$

C that determines the costs associated to incorrectly classifying datapoints is an open parameter of the optimization function

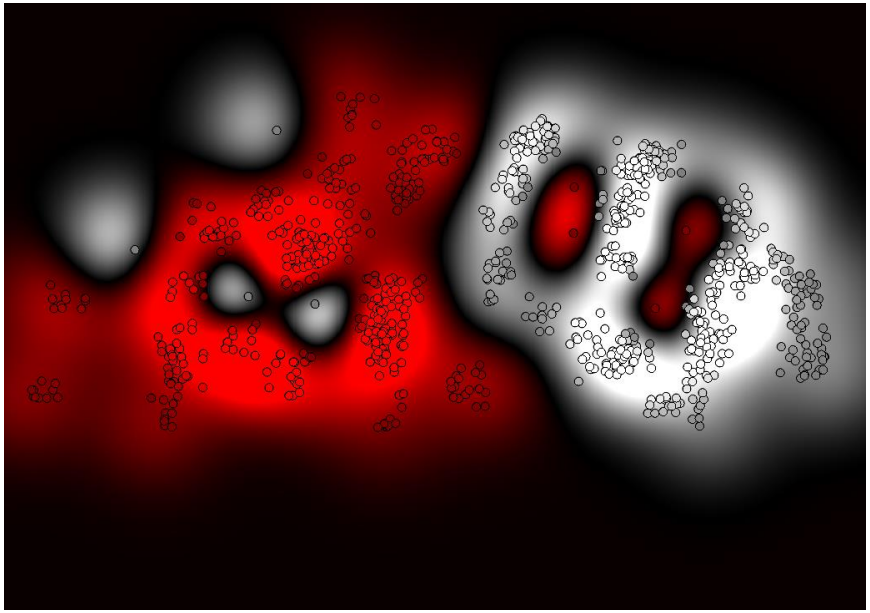




Starts overfitting



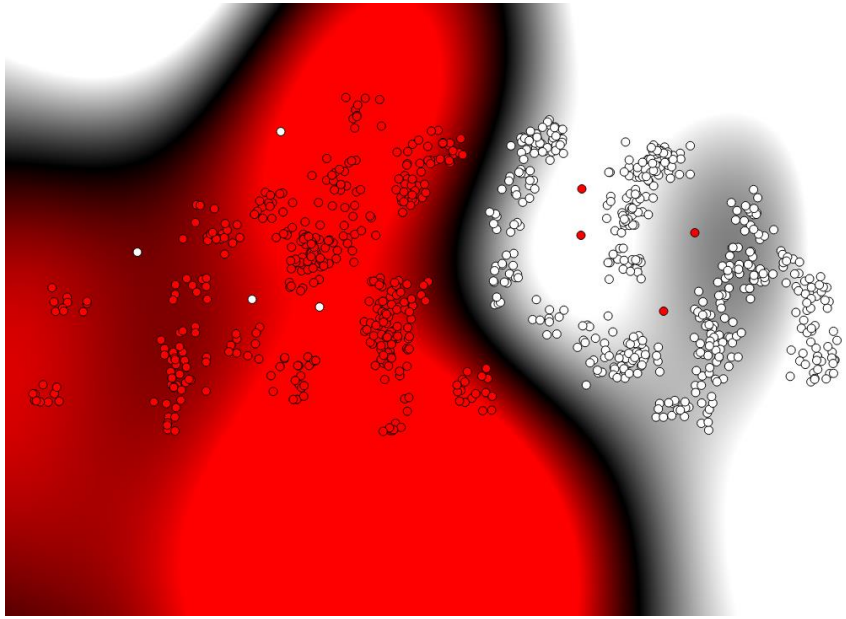
C=1; several misclassified datapoints



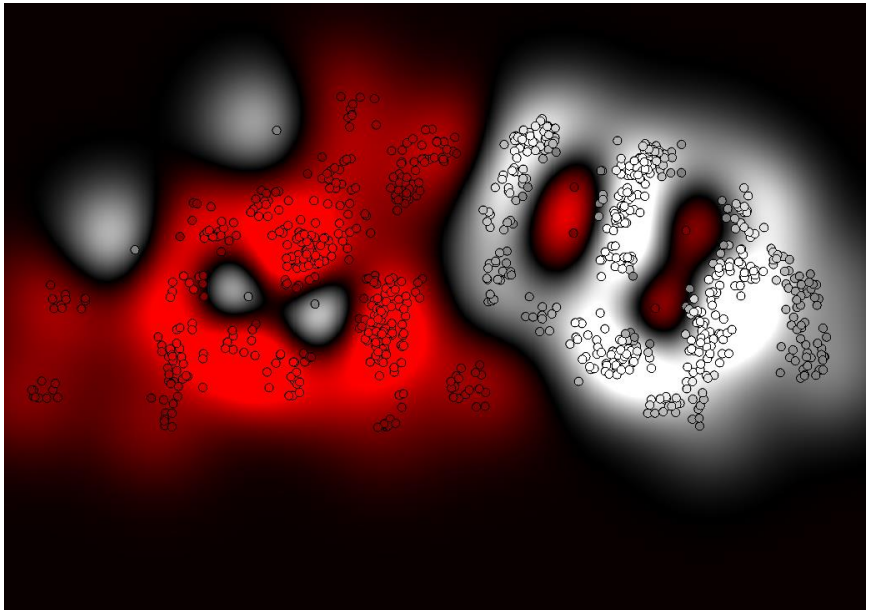
C=100; all points correctly classified

Kernel width

RBF (Gaussian) kernel: $k\left(x, x^i\right) = e^{-\frac{\left\|x - x^i\right\|}{2\sigma^2}}, \quad \sigma \in \mathbb{R}.$



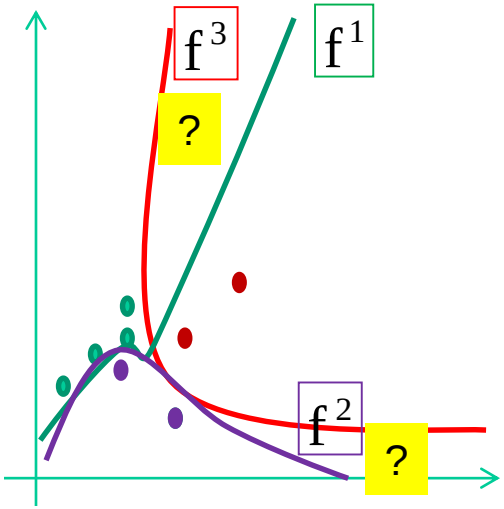
σ=0.2; smoother fit

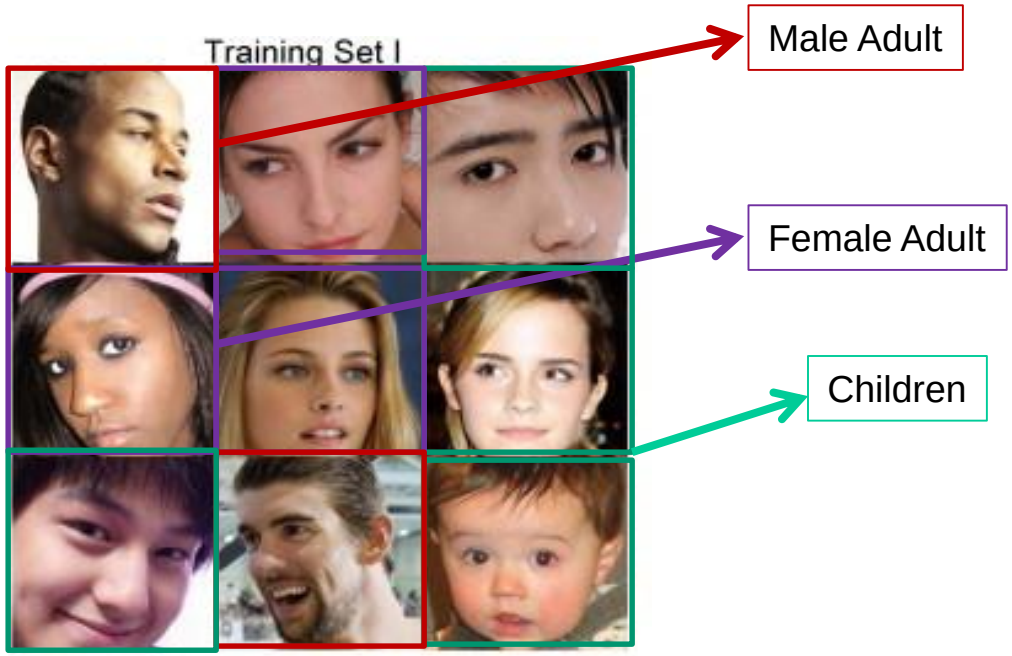


σ=0.01; too tight a fit

Multi-class classification with SVM

Multi-Class SVM





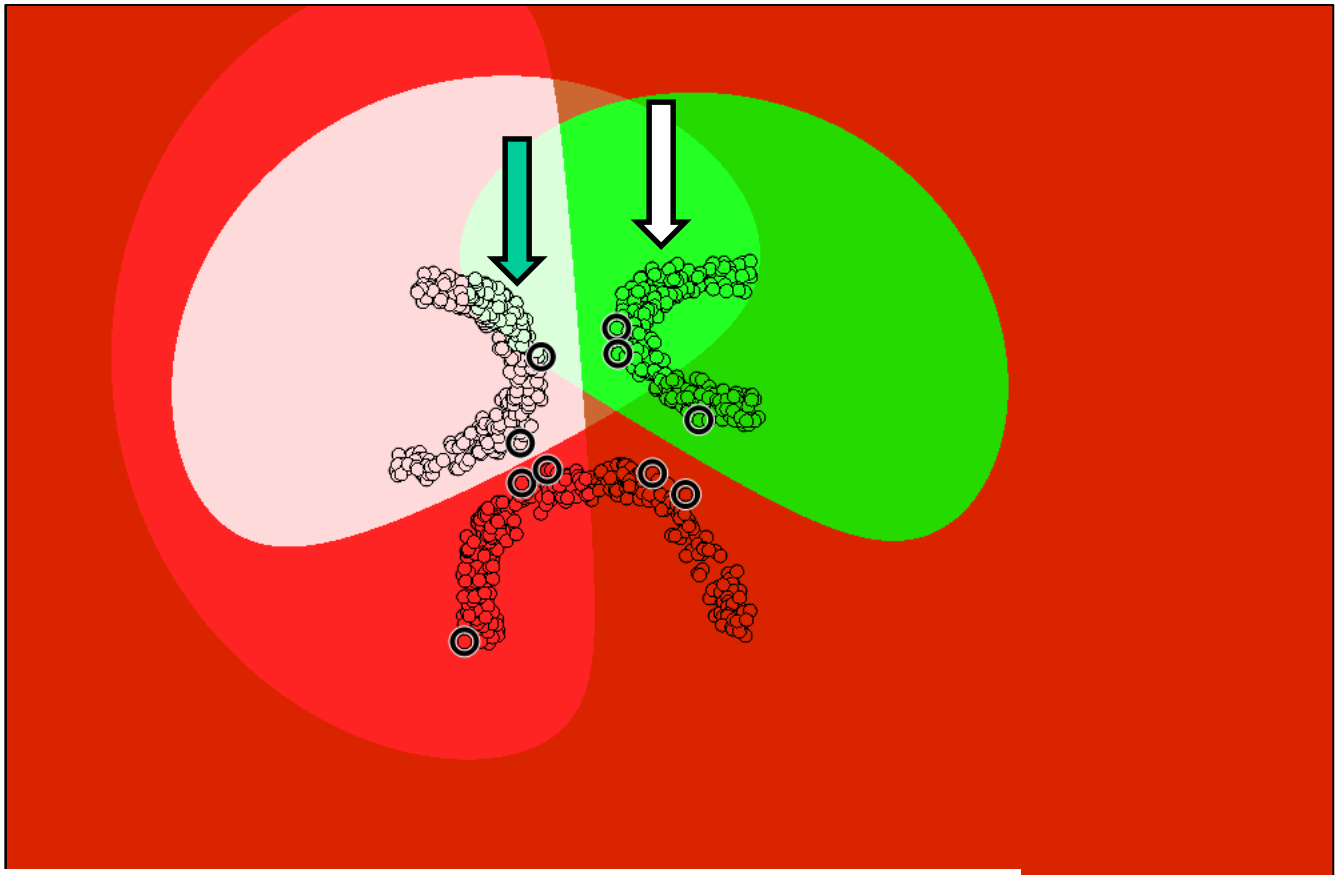
Construct a set of K binary classifiers $f^1,....,f^K$, each trained to separate one class from the rest.

Compute the class label in a winner-take-all approach:

$$j=\arg\max_{j=1,...K}\left(\sum_{i=1}^{M} y^i \alpha_i^j k\left(x, x^i\right)+b^j\right)$$

Sufficient to compute only K-1 classifier for K classes
But computing the K'th classifier may provide tighter bounds on the Kth class.

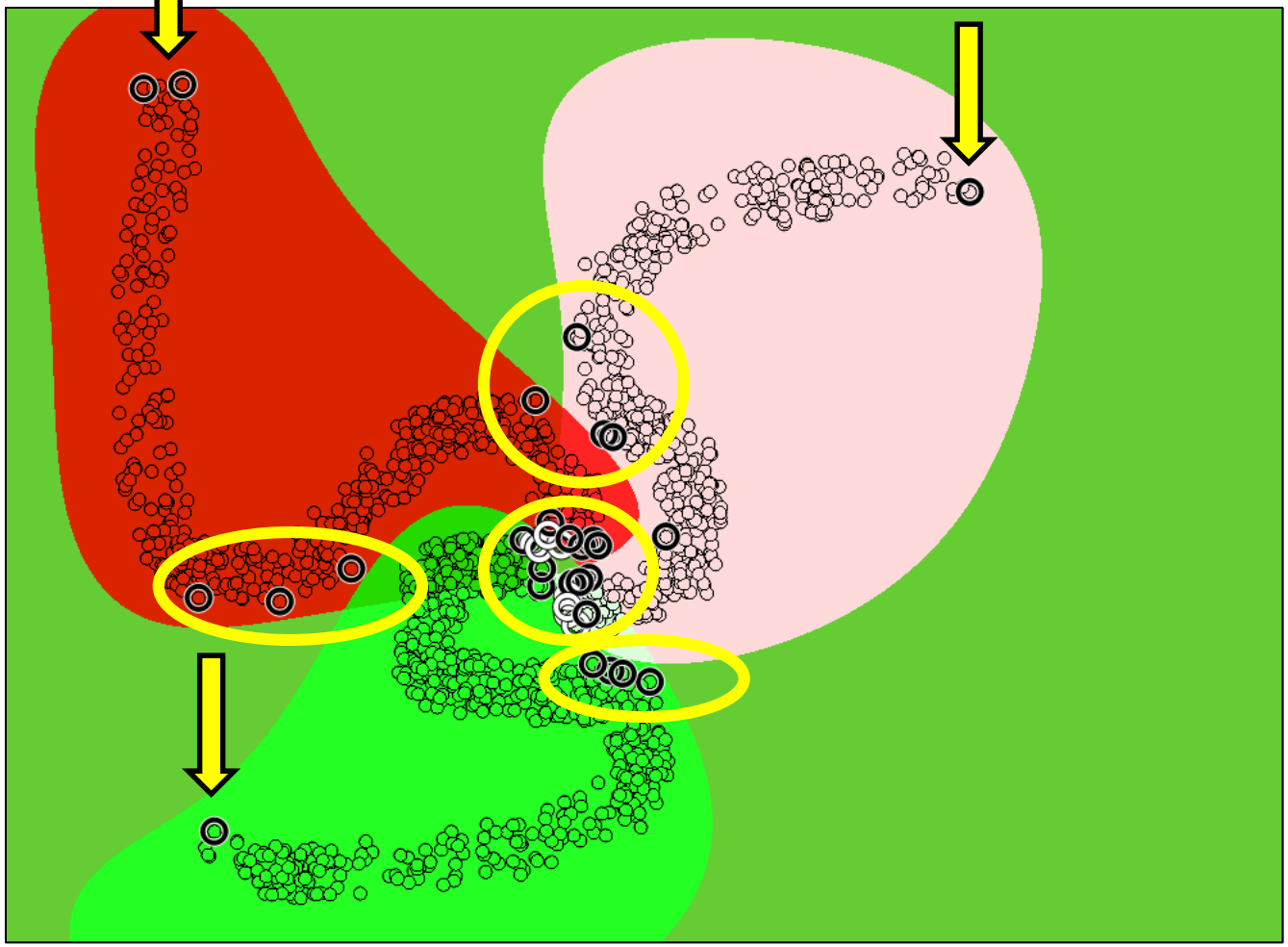
Multi-Class SVM



Compute the class label in a winner-take-all approach:

$$\mathrm{j}=\arg\max_{j=1,\ldots K}\left(\sum_{i=1}^{M} y^i \alpha_i^j k\left(x, x^i\right)+b^j\right)$$

Multi-Class SVM



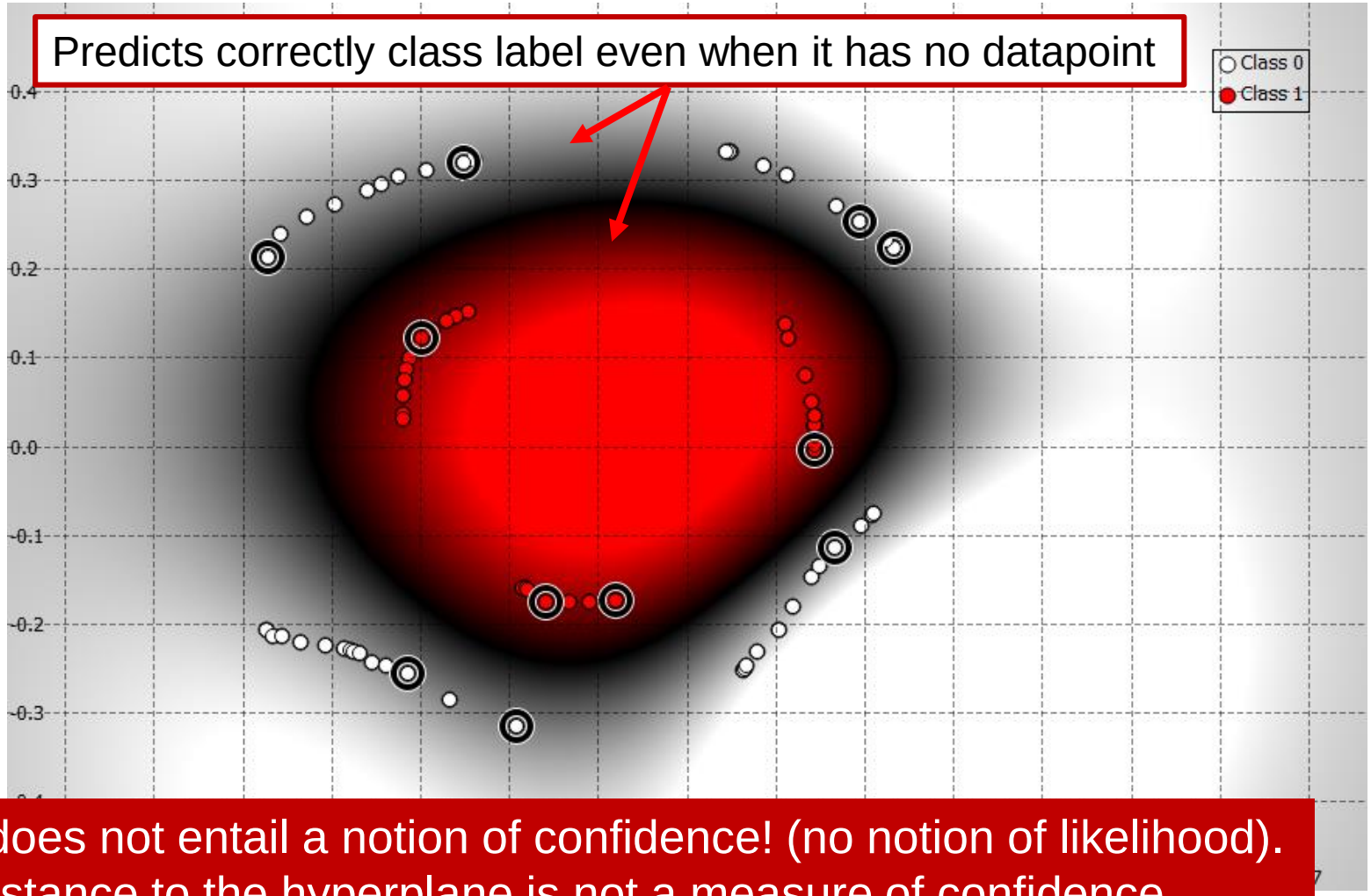
Summary

Summary: What is SVM?

- An algorithm developed originally in the early 90's, by Vapnik and his coworkers, which could be applied in classification and regression;
- It does the following:
  - map original input space to higher dimension feature space, which is implemented implicitly by kernel function, such that "linear decision boundaries constructed in the high dimensional feature space correspond to nonlinear decision boundaries in the input space";
  - in feature space, an optimal separating hyperplane is constructed, which could be determined by solving an optimization problem;
  - Lagrange multipliers and dual theory can then be applied to convert this optimization problem into a convex quadratic program subject to linear constraints.

Pros and Cons of SVM

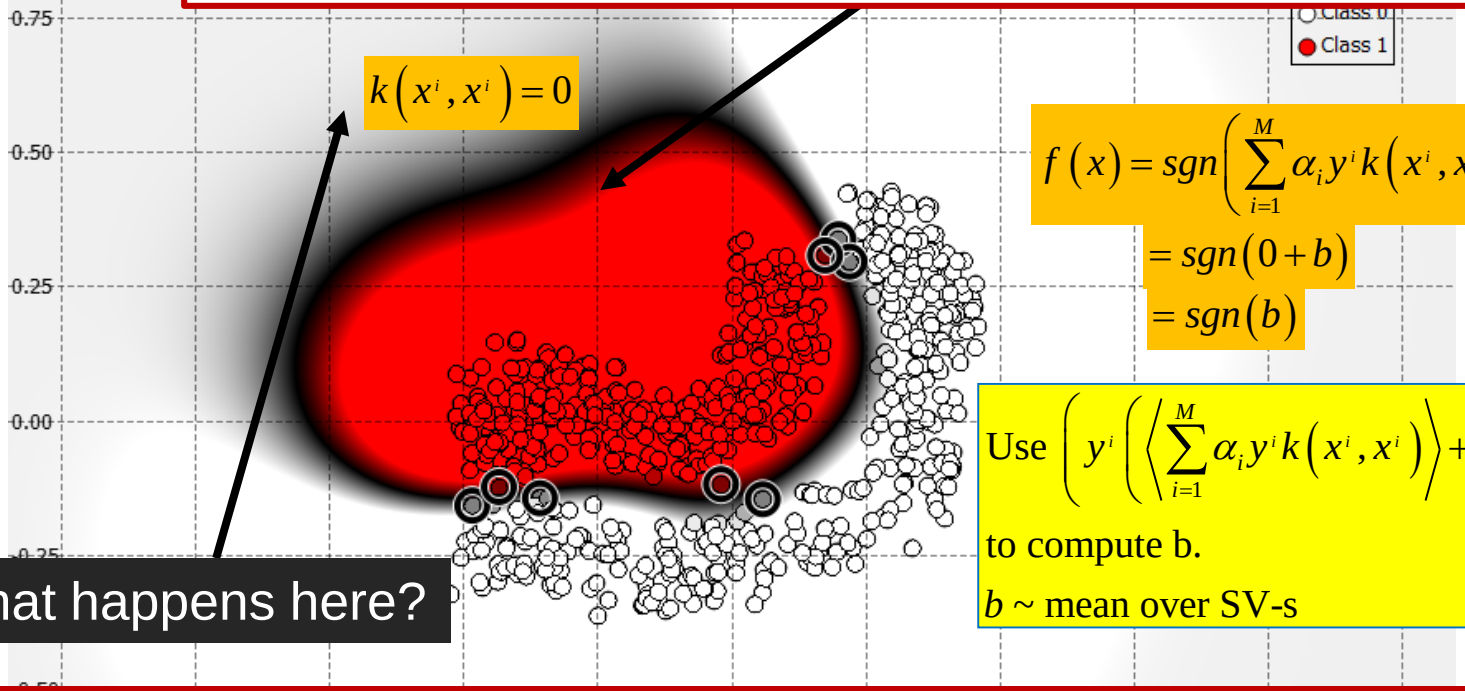
SVM – Generalization



SVM does not entail a notion of confidence! (no notion of likelihood).
The distance to the hyperplane is not a measure of confidence.
→ You cannot tell if prediction is correct or not!

SVM – no confidence in prediction

Generalization: Predicts class label even when it has no datapoint
Prediction could be completely incorrect



$$f(x) = sgn\left(\sum_{i=1}^{M} \alpha_i y^i k(x^i, x^i) + b\right)$$
$$= sgn(0 + b)$$
$$= sgn(b)$$

Use $\left(y^i\left(\left\langle\sum_{i=1}^{M} \alpha_i y^i k(x^i, x^i)\right\rangle + b\right) - 1\right) = 0$

to compute b.

b ~ mean over SV-s

Predicts by default sign of *b* when far from the training datapoints!!
→ This could lead to large amount of *false positives* for the class with same sign as *b!*

Doing crossvalidation would not prevent this effect as it would use only points at your disposal that are close to training datapoints.

SVM – confidence in prediction

- False positives must be treated with care.
- Imagine you classify images of cancer tissue and you want to predict if the tissue has a tumor (positive class) or no tumor (negative class); **you cannot afford false positive for the negative class.**
- To prevent this to happen, you should:
  - Verify that the sign of *b* is not the sign of the class you care about.
  - Run crossvalidation by generating a testing set from points never seen – far from your training set.
  - Compare distribution of training and testing sets with GMM to make sure they are different

SVM versus GMM and KNN

❖ Advantages of SVM over GMM and KNN:

- ➢ Build a model (compared to KNN) – boundary is an explicit function
- ➢ Guaranteed to find the global optimum (compared to GMM)

❖ Disadvantages of SVM :

- ➢ Computational costs at testing can be daunting O(M), M: number of datapoints
- ➢ It requires to choose two hyperparameters (C and kernel width) – compared to 1 for KNN and GMM.

Summary: When to use SVM?

- SVM have wide range application for all type of data (vision, text, handwriting, robotics).
- SVM is very powerful for large scale classification. Optimized solvers for the training stage. Rapid during recall.
- One issue is that the computation grows linearly with number of datapoints and the number of SV and the algorithm is not sparse in nb of SV-s
  → see ν–SVM and RVM in Advanced ML course videos (optional).
- Another issue is that SVM can predict only two classes. For multi-class classification, one needs to run several two-class classifiers.
- Finally, SVM does not have a notion of confidence. Extensions to SVM, such as RVM, can offer such probabilistic interpretation → see RVM in Advanced ML course video (optional).