Sparse Kernel Methods

TTT4185 Machine Learning for Signal Processing

Giampiero Salvi

Department of Electronic Systems
NTNU

HT2021

Outline

Outline

Two-Class Classification Problem

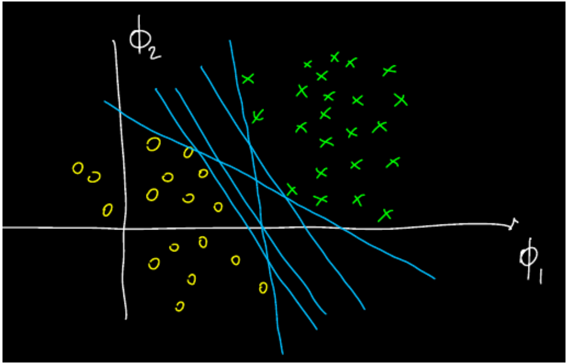
Model

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

Training data (assume linearly separable)

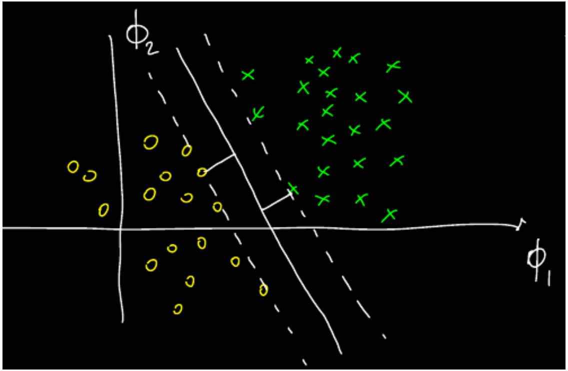
$$\{\mathbf{x}_1, \dots, \mathbf{x}_N\}, \{t_1, \dots, t_N\}$$

how to choose the best solution?



Maximum Margin

- Goal: minimize generalization error
- Problem: we can not use the test data
- Solution (Heuristics): choose decision boundary as far as possible from data



Optimization

- Only interested in solutions with no misclassifications:

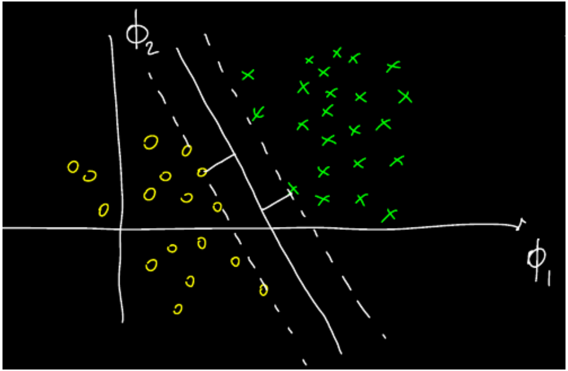
$$t_n y(\mathbf{x}_n) > 0$$

- if $\mathbf{x}$ on the decision boundary then

$$y(\mathbf{x}) = 0$$

- Distance between a point and the decision boundary:

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

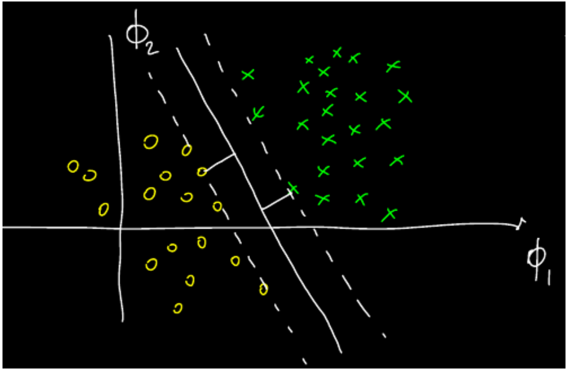


Optimization

- We can rewrite the distance as

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

- we want to maximize the distance of the closest point:



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

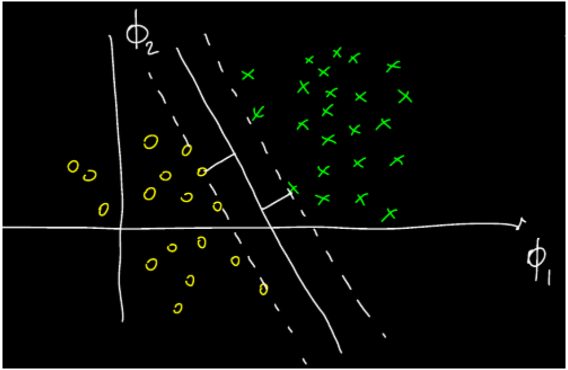
Canonical Representation

- Rescaling $\mathbf{w}, b$ does not change distances
- Define $\mathbf{w}, b$ such that:

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

  for the point that is closest to the decision boundary
- then

$$t_n \left(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) + b \right) \geq 1, \forall n \in [1, N]$$



Optimization Problem (Quadratic Programming)

$$\arg\max_{\mathbf{w},b} \left\{ \frac{1}{||\mathbf{w}||} \right\} = \arg\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2$$

subject to the constraints

$$t_n \left(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) + b \right) \geq 1, \forall n \in [1, N]$$

- can be sloved with Lagrange multipliers

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

Dual Representation

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

subject to

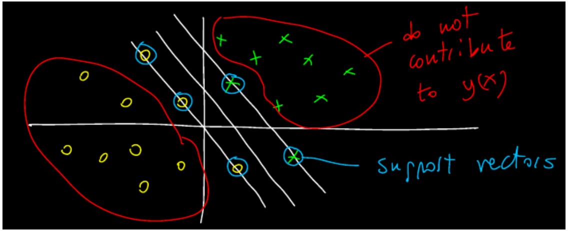
$$a_n \geq 0$$

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

Dual Representation for Prediction

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

subject to the constraints (Karish-Kuhn-Tucker, KKT)

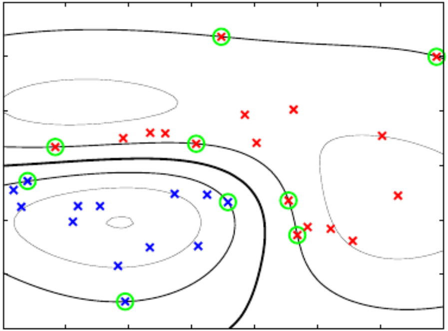


$$a_n \geq 0$$
$$t_n y(\mathbf{x}_n) - 1 \geq 0$$
$$a_n \{t_n y(\mathbf{x}_n) - 1\} = 0$$

Properties

- complexity of quadratic programming in M variables: $O(M^3)$
- with dual representation we have N variables (usually $N >> M$)
- but is convex optimization: global optimum
- and we can work in arbitrary large dimensions

Example



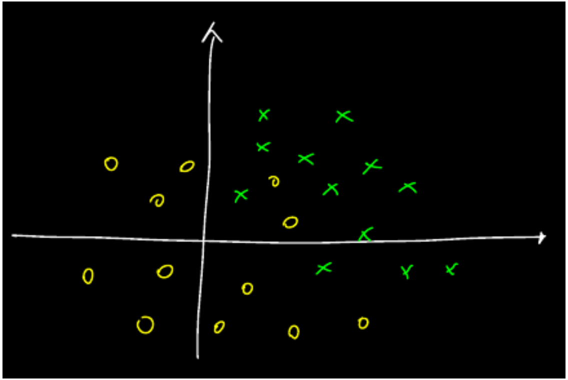
Example

https://playground.tensorflow.org

Outline

Overlapping Classes

- we need to allow some misclassifications



Loss Function

- previously assumed zero errors + canonical representation
- $\Rightarrow$ loss function is $\frac{1}{2}||\mathbf{w}||^2$
- equivalent to

$$\sum_{n=1}^{N} E_\infty(t_n y(\mathbf{x}_n) - 1) + \lambda||\mathbf{w}||^2,$$

where

$$E_\infty(z) = \begin{cases} 0 & \text{if } z \geq 0 \\ \infty & \text{otherwise.} \end{cases}$$

- we need to modify this to allow for finite errors

Slack Variables

$$\begin{aligned} \xi_n &\geq 0, && \forall n \in [1, N] \\ \xi_n &= 0, && \text{inside the margin}^* \\ \xi_n &= |t_n - y(\mathbf{x}_n)|, && \text{outside the margin}^* \end{aligned}$$

we substitute the constraint $t_n y(\mathbf{x}_n) \geq 0$ with:

$$t_n y(\mathbf{x}_n) \geq 1 - \xi_n, \qquad \forall n \in [1, N]$$

* with respect to the class

Loss Function

Goal: maximize margin by minimizing error:

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

- C controls trade-off between slack variable penalty and margin
- for misclassified points: $\xi_n > 1$
- $\Rightarrow \sum_{n=1}^{N} \xi_n$ upper bound to # errors
- $C \to \infty$ gives same solution as for separable data

Lagrange Multipliers

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

KKT conditions:

$$\begin{aligned}
a_n &\geq 0 \\
t_n y(\mathbf{x}_n) - 1 + \xi_n &\geq 0 \\
a_n(t_n y(\mathbf{x}_n) - 1 + \xi_n) &= 0 \\
\mu_n &\geq 0 \\
\xi_n &\geq 0 \\
\mu_n\xi_n &\geq 0
\end{aligned}$$

Dual Representation

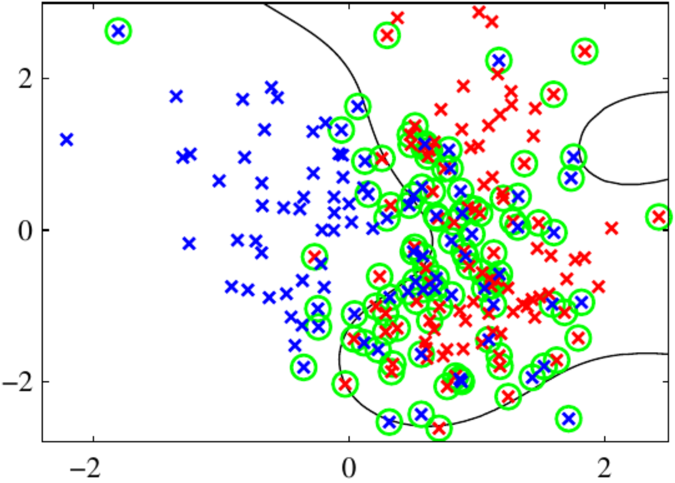
$$\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m)$$

Same as before but with different constraints

$$0 \leq a_n \leq C$$

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

Example



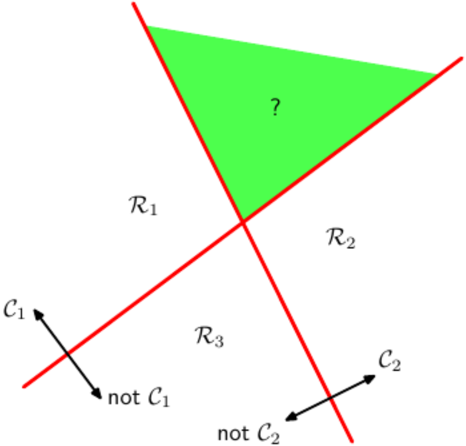
Outline

Multi-Class SVM

- SVM is a two-class classifier
- Several approaches to solve the K-class problem

Multi-Class SVM: Vapnik 1998

- Construct K different classifiers $y_k(\mathbf{x})$
- for each use data from C_k as positive examples and remaining classes as negative
- known as one-versus-the-rest approach
- problem in figure



Multi-Class SVM: Possible Solution

- Instead of one-versus-the-rest use:

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

- new problem: every $y_k(\mathbf{x})$ is trained on a different task
- no guarantee that have same scale
- other problem: if e.g. $K = 10$, for each $y_k(\mathbf{x})$, 10% positive and 90% negative examples

Multi-Class SVM: Possible Solution

- Instead of one-versus-the-rest use:

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

- new problem: every $y_k(\mathbf{x})$ is trained on a different task
- no guarantee that have same scale
- other problem: if e.g. $K = 10$, for each $y_k(\mathbf{x})$, 10% positive and 90% negative examples
- Solution (Lee et al 2001): scale targets
    - $+1$ for the positive class
    - $\frac{-1}{K-1}$ for the negative class

Multi-Class SVM: Weston and Watkins 1999

- objective function for training all $y_k(\mathbf{x})$ simultaneously
- but computationally expensive:
- instead of K problems over N data points $O(KN^2)$
- single problem of size $(K-1)N$ which is $O(K^2N^2)$

Multi-Class SVM: one-versus-one

- train $\frac{K(K-1)}{2}$ classifiers (one-versus-one)
- this is used in `sklearn.svm.SVC` (assignment 2)