STAT 535

11/22/22

Lecture 16

- SVM dual problem
- linear C-SVM
- kernel SVM

- Next lecture on Zoom
  Remote, recorded

Lecture V: Support Vector Machines

Marina Meilă
mmp@stat.washington.edu

Department of Statistics
University of Washington

November, 2022

Linear SVM's

- The margin and the expected classification error
- Maximum Margin Linear classifiers
- Linear classifiers for non-linearly separable data

Non linear SVM

- The "kernel trick"
- Kernels
- Prediction with SVM

Extensions

- L_1 SVM
- Multi-class and One class SVM
- SV Regression

Reading HTF Ch.: Ch. 12.1–3, Murphy Ch.: Ch 14 (14.1,14.2–14.2.4 kernels, 14.4 and equations (14.28,14.29) kernel trick, 14.5.1.–3 Support Vector Machines), Bach Ch.: 7.1–7.4, 7.7
Additional Reading: C. Burges - "A tutorial on SVM for pattern recognition"
These notes: Appendices (convex optimization) are optional.

Second idea

$\propto \frac{1}{||w||}$ = smallest margin

The **Second idea** is to formulate (10) as a **quadratic** optimization problem.

$$\min_{w,b} \frac{1}{2}||w||^2 \text{ s.t } y^i(w^T x^i + b) \geq 1 \text{ for all } i = 1 : n \quad (13)$$

constraints

This is the **Linear SVM (primal) optimization problem**

- This problem has a strongly convex **objective** $||w||^2$, and **constraints** $y^i(w^T x^i + b)$ linear in (w, b).
- Hence this is a convex problem, and can be studied with the tools of convex optimization.

Solved by off-the shelf optimizer. E.g. CVX

The distance of a point x to a hyperplane $H_{w,b}$

$$d(x, H_{w,b}) = \frac{|w^T x + b|}{||w||} \tag{14}$$

Intuition: denote

$$\tilde{w} = \frac{w}{||w||}, \ \tilde{b} = \frac{b}{||w||}, \ x' = \tilde{w}^T x. \tag{15}$$

Obviously $H_{w,b} = H_{\tilde{w},\tilde{b}}$, and x' is the length of the projection of point x on the direction of w.

The distance is measured along the normal through x to H; note that if $x' = -\tilde{b}$ then $x \in H_{w,b}$ and $d(x, H_{w,b}) = 0$; in general, the distance along this line will be $|x' - (-\tilde{b})|$.

Optimization with Lagrange multipliers

[2] The **Lagrangean** of (13) is

$$L(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_i \alpha_i [y^i (w^T x^i + b) - 1]. \quad (16)$$

[KKT conditions]

At the optimum of (13)

$$w = \sum_i \alpha_i y^i x^i \quad \text{with } \alpha_i \geq 0 \quad (17)$$

and $b = y^i - w^T x^i$ for any i with $\alpha_i > 0$.

- **Support vector** is a data point x^i such that $\alpha_i > 0$.
- According to (17), the final decision boundary is determined by the support vectors (i.e. does not depend explicitly on any data point that is not a support vector).

[2] The derivations of these results are in the Appendix

Karush Kuhn Tucker (at optimum w^*, b^*, α^*)

$$\frac{\partial L}{\partial w} = 0 = w - \sum_{i=1}^{n} \alpha_i y^i x^i \Rightarrow \boxed{w^* = \sum_{i=1}^{n} \alpha_i^* y^i x^i}$$

linear combination of x^i ← only if $\alpha_i^* \neq 0$ → x^i SUPPORT VECTOR

w^* = data

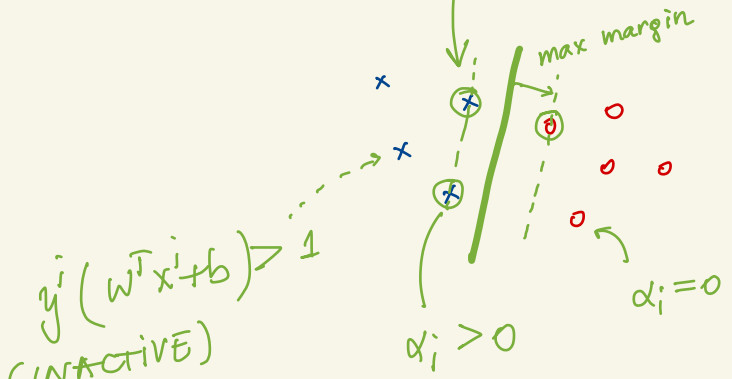
Non-parametric

$$\frac{\partial L}{\partial b} = 0 = -\sum \alpha_i y^i$$

replace b^* with constraint

$i = 1:n$ ← one dual variable / constraints

$$L = \frac{1}{2}\|w\|^2 + \sum \alpha_i \left[1 - y^i (w^T x^i + b)\right]$$



Dual optimization Pb

$$\max_{\alpha \in \mathbb{R}^n} \; g(\alpha) \quad \text{s.t} \quad \alpha_i \geq 0$$

$g(\alpha)$ ← dual function; $\alpha_i \geq 0$ ← simpler + constraints

$$g(\alpha) = L(w^*, b^*, \alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y^i y^j (x_i^T x^j)$$

Solution $\Rightarrow \alpha^* \Rightarrow w^* \Rightarrow b^* \ldots$

$$\|w\|^2 = w^T w = \sum_{i,j=1}^{n} \alpha_i \alpha_j y^i y^j (x^i)^T x^j$$

$$\sum_{i=1}^{n} (\alpha_i y^i x^i) w^T = \text{—“—}$$

$$\left(\sum \alpha_i y^i\right) b = 0$$

$$g \equiv L(w^*, b^*, \alpha) =$$

Finding b^*

1) pick i support vector

2) $(w^*)^T x^i + b = y^i \Rightarrow b = y^i - (w^*)^T x^i$

$$g(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y^i y^j (x^i)^T x^j = 1^T \alpha - \frac{1}{2} \alpha^T \bar{G} \alpha$$

$$1 = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbb{R}^n$$

$$\bar{G}_{ij} = y^i y^j (x^i)^T x^j, \quad \bar{G} = \begin{bmatrix} y^i & & \\ & \ddots & \\ & & \end{bmatrix} G \begin{bmatrix} y^i & & \\ & \ddots & \\ & & \end{bmatrix}$$

Gram matrix

$$G_{ij} = (x^i)^T x^j \Rightarrow G = XX^T \quad n \times n$$

$$X = \begin{bmatrix} -(x^i)^T- \\ - \quad - \end{bmatrix}$$

uncentered $\Sigma = X^T X$ $d \times d$

Dual SVM optimization problem

- Any convex optimization problem has a **dual** problem. In SVM, it is both illuminating and practical to solve the dual problem.
- The dual to problem (13) is

$$\max_{\alpha_{1:n}} \sum_i \alpha_i - \frac{1}{2}\sum_i \alpha_i \alpha_j y^i y^j x^{i\,T} x_j \text{ s.t } \alpha_i \geq 0 \text{ for all } i \text{ and } \sum_i \alpha_i y^i = 0. \tag{18}$$

- This is a quadratic problem with n variables on a convex domain.
- Dual problem in matrix form
  - Denote $\alpha = [\alpha_i]_{i=1:n}$, $y = [y^i]_{i=1:n}$, $G_{ij} = x^{i\,T} x_j$, $\bar{G}_{ij} = y^i y^j x^{i\,T} x_j$, $G = [G_{ij}] \in \mathbb{R}^{n\times n}$, $\bar{G} = [\bar{G}_{ij}] \in \mathbb{R}^{n\times n}$.

$$\max_{\alpha\in\mathbb{R}^n} 1^T\alpha - \frac{1}{2}\alpha^T \bar{G}\alpha \quad \text{s.t } \alpha \succeq 0 \text{ and } y^T\alpha = 0. \tag{19}$$

- $g(\alpha) = 1^T\alpha - \frac{1}{2}\alpha^T\bar{G}\alpha$ is the **dual objective function**
- G is called the **Gram matrix** of the data. Note that $\bar{G} = \operatorname{diag} y^{1:n\,T} G \operatorname{diag} y^{1:n}$.
- At the dual optimum
  - $\alpha_i > 0$ for constraints that are satisfied with equality, i.e. **tight**
  - $\alpha_i = 0$ for the **slack** constraints

Compare with Primal ~ $w\in\mathbb{R}^d, b\in\mathbb{R}$ ← n large

Dual $\alpha\in\mathbb{R}^n$ ← n small, d large

- Linear classifier, separable data ✓
- ——"——, any data ← ✓
- Non-linear, —"— ← Kernel trick

Non-linearly separable problems and their duals

The C-SVM

$$\begin{aligned} \text{minimize}_{w,b,\xi} \quad & \frac{1}{2}||w||^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & y^i(w^T x^i + b) \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned} \tag{20}$$

In the above, ξ_i are the slack variables. Dual[3]:

$$\begin{aligned} \text{maximize}_{\alpha} \quad & \sum_i \alpha_i - \frac{1}{2}\sum_i \alpha_i \alpha_j y^i y_j x^{i\,T} x_j \\ \text{s.t.} \quad & C \geq \alpha_i \geq 0 \text{ for all } i \\ & \sum_i \alpha_i y^i = 0 \end{aligned} \tag{21}$$

$\Rightarrow$ two types of SV

- $\alpha_i < C$ data point x^i is "on the margin" $\Leftrightarrow$ $y^i(w^T x^i + b) = 1$ (original SV)
- $\alpha_i = C$ data point x^i cannot be classified with margin 1 (**margin error**) $\Leftrightarrow$ $y^i(w^T x^i + b) < 1$

[3] Lagrangean $L(w, b, \xi, \alpha, \mu) = \frac{1}{2}||w||^2 + C\sum_i \xi_i - \sum_i \alpha_i[y^i(w^T x^i + b) - 1 + \xi_i] - \sum_i \mu_i \xi_i$ with $\alpha_i \geq 0,\ \xi_i \geq 0,\ \mu_i \geq 0$

The ν-SVM

$$\text{minimize}_{w,b,\xi,\rho} \quad \frac{1}{2}||w||^2 - \nu\rho + \frac{1}{n}\sum_i \xi_i \tag{22}$$

$$\text{s.t.} \quad y^i(w^T x^i + b) \geq \rho - \xi_i \tag{23}$$

$$\xi_i \geq 0 \tag{24}$$

$$\rho \geq 0 \tag{25}$$

where $\nu \in [0, 1]$ is a parameter.
Dual[4]:

$$\text{maximize}_{\alpha} \quad -\frac{1}{2}\sum_i \alpha_i \alpha_j y^i y^j x^{i\,T} x^j \tag{26}$$

$$\text{s.t.} \quad \frac{1}{n} \geq \alpha_i \geq 0 \text{ for all } i \tag{27}$$

$$\sum_i \alpha_i y^i = 0 \tag{28}$$

$$\sum_i \alpha_i \geq \nu \tag{29}$$

Properties If $\rho > 0$ then:

- ν is an upper bound on #margin errors/n (if $\sum_i \alpha_i = \nu$)
- ν is a lower bound on #(original support vectors + margin errors)/n
- ν-SVM leads to the same w, b as C-SVM with $C = 1/\nu$

[4] Lagrangean $L(w, b, \xi, \rho, \alpha, \mu, \delta) = \frac{1}{2}||w||^2 - \nu\rho + \frac{1}{n}\sum_i \xi_i - \sum_i \alpha_i[y^i(w^T x^i + b) - \rho + \xi_i] - \sum_i \mu_i \xi_i - \delta\rho$ with $\alpha_i \geq 0,\ \delta \geq 0,\ \mu_i \geq 0$

A simple error bound

$$L_{01}(f_n) \leq E\left[\frac{\#\text{support vectors of } f_{n+1}}{n+1}\right] \tag{30}$$

where f_n denotes the SVM trained on a sample of size n.

Exercise Use the Homework 6 to prove this result.

Non-linear SVM

How to use linear classifier on data that is not linearly separable?

An old trick

1. Map the data $x^{1:n}$ to a higher dimensional space

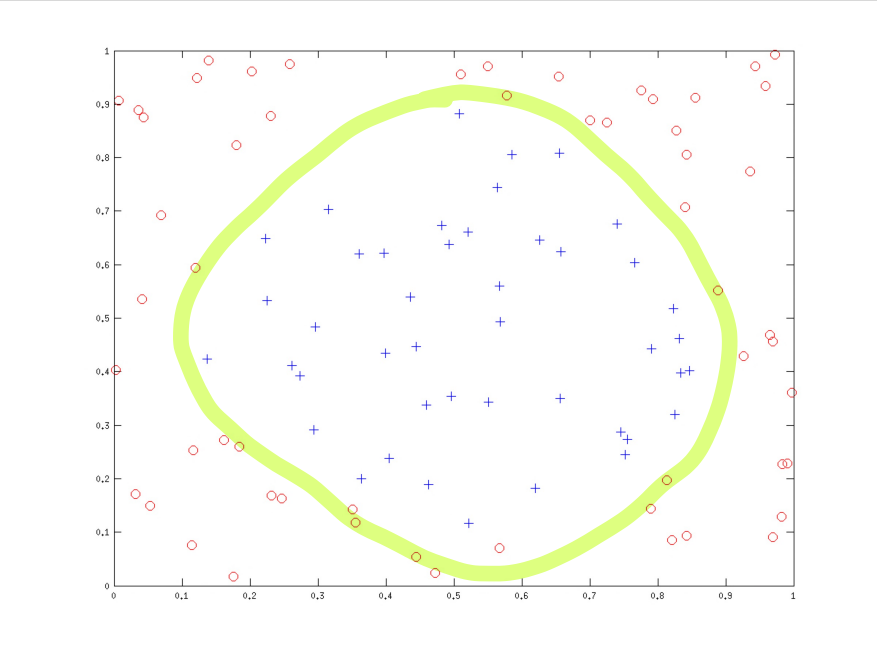
$$x \rightarrow z = \phi(x) \in \mathcal{H}, \text{with dim } \mathcal{H} >> n.$$

$\dim w = \dim \varphi(x) \equiv \dim \mathcal{H}$

2. Construct a linear classifier $w^T z + b$ for the data in $\mathcal{H}$

   In other words, we are implementing the non-linear classifier

$$f(x) = w^T \phi(x) + b = w_1\phi_1(x) + w_2\phi_2(x) + \ldots + w_m\phi_m(x) + b \quad (31)$$



Example

- Data $\{(x, y)\}$ below are not linearly separable

x		y	z		
-1	-1	1	-1	-1	1
-1	1	-1	-1	1	-1
1	-1	-1	1	-1	-1
1	1	1	1	1	1

- We map them to 3 dimensions by

$$z = \phi(x) = [x_1 \; x_2 \; x_1 x_2].$$

- Now the classes can be separated by the hypeplane $z_3 = 0$ (which happens to be the maximum margin hyperplane). Hence,
  - $w = [0\ 0\ 1]$ (a vector in $\mathcal{H}$)
  - $b = 0$
  - and the classification rule is $f(\phi(x)) = w^T \phi(x) + b$.
- If we write f as a function of the original x we get

$$f(x) = x_1 x_2$$

a quadratic classifier.

Non-linear SV problem

- Primal problem minimize $\frac{1}{2}||w||^2$ s.t $y^i(w^T\phi(x^i)+b)-1\geq 0$ for all i.
- Dual problem

$$\max_{\alpha_{1:n}} \sum_i \alpha_i - \frac{1}{2}\sum_i \alpha_i\alpha_j \underbrace{y^i y_j \phi(x^i)^T\phi(x_j)}_{\bar{G}_{ij}} \text{ s.t. } \alpha_i \geq 0 \text{ for all } i \text{ and } \sum_i y^i\alpha_i = 0 \quad (32)$$

$$G_{ij} = \phi(x^i)^T\phi(x^j) \text{ and } \bar{G} = y^T G y \quad (33)$$

- $\bar{G}_{ij}$ has been redefined in terms of ϕ
- Dual problem

$$\max_{\alpha} 1^T\alpha - \frac{1}{2}\alpha^T\bar{G}\alpha \quad \text{s.t. } \alpha_i \geq 0,\ y^T\alpha = 0 \quad (34)$$

- Same as (19)!

$\dim \mathcal{W} = \dim \mathcal{H}$

only on $(z^i)^T z^j$ — Kernel trick

only change in dual

$G \in \mathbb{R}^{n\times n}$

compute $(z^i)^T z^j$ in $O(d)$ time

$x \in \mathbb{R}^d$

φ = feature map

Want φ so that $K(x,x') = \varphi(x)^T\varphi(x')$

in $\mathbb{R}^d$ non-linear — in $\mathcal{H}$ bilinear

The "Kernel Trick"

d idea The result (34) is the celebrated **kernel trick** of the SVM literature. We can make the following remarks.

1. The ϕ vectors enter the SVM optimization problem only trough the Gram matrix, thus only as the scalar products $\phi(x^i)^T\phi(x_j)$. We denote by $K(x, x')$ the function

$$K(x, x') = K(x', x) = \phi(x)^T\phi(x') \tag{35}$$

   K is called the **kernel** function. If K can be computed efficiently, then the Gram matrix G can also be computed efficiently. This is exactly what one does in practice: we choose ϕ implicitly by choosing a kernel K. Hereby we also ensure that K can be computed efficiently.
2. Once G is obtained, the SVM optimization is independent of the dimension of x and of the dimension of $z = \phi(x)$. The complexity of the SVM optimization depends only on n the number of examples. This means that we can choose a very high dimensional ϕ without any penalty on the optimization cost.
3. Classifying a new point x. As we know, the SVM classification rule is

$$f(x) = w^T\phi(x) + b = \sum_{i=1}^{n} \alpha_i y^i \phi(x^i)^T\phi(x) = \sum_{i=1}^{n} \alpha_i y^i K(x^i, x) \tag{36}$$

   Hence, the classification rule is expressed in terms of the support vectors and the kernel only. No operations other than scalar product are performed in the high dimensional space H.

Kernels

The previous section shows why SVMs are often called **kernel machines**. If we choose a kernel, we have all the benefits of a mapping in high dimensions, without ever carrying on any operations in that high dimensional space. The most usual kernel functions are

$K(x, x') = (1 + x^T x')^p$ the polynomial kernel of degree p

$K(x, x') = \tanh(\sigma x^T x' - \beta)$ the "neural network" kernel

$K(x, x') = e^{-\frac{||x-x'||^2}{\sigma^2}}$ the Gaussian or **radial basis function** (RBF) kernel

it's ϕ is ∞-dimensional

Happy Thanksgiving !!

The Mercer condition

- How do we verify that a chosen K is is a valid kernel, i.e that there exists a ϕ so that $K(x, x') = \phi(x)^T \phi(x')$?
- This property is ensured by a positivity condition known as the Mercer condition.

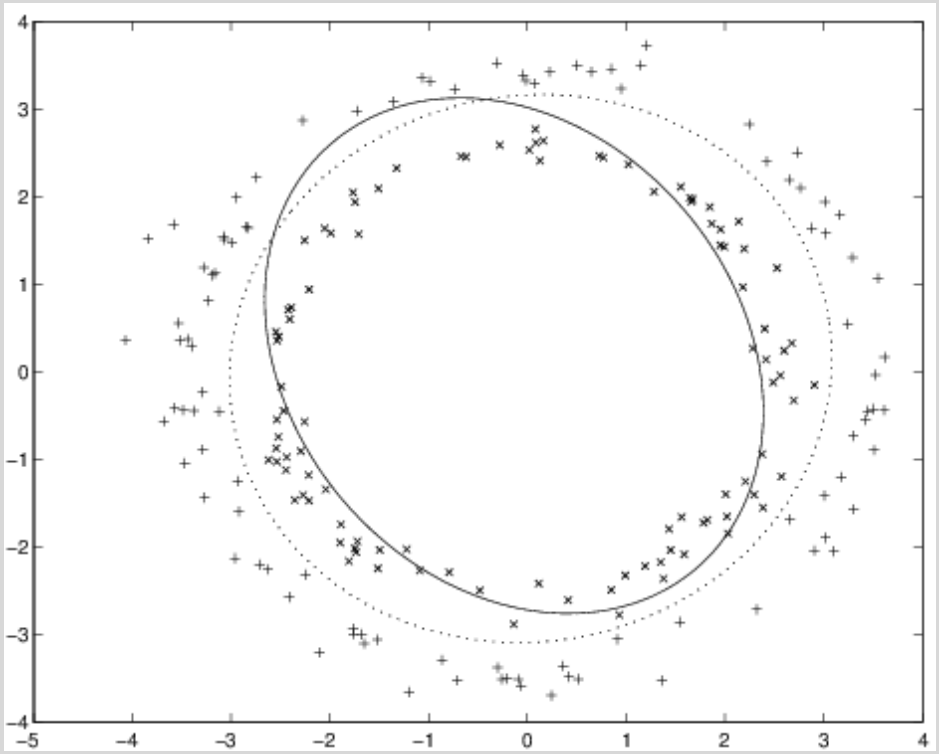
Mercer condition

Let $(\mathcal{X}, \mu)$ be a finite measure space. A symmetric function $K : \mathcal{X} \times \mathcal{X}$, can be written in the form $K(x, x') = \phi(x)^T \phi(x')$ for some $\phi : \mathcal{X} \to \mathcal{H} \subset \mathbb{R}^m$ iff

$$\int_{\mathcal{X}^2} K(x, x') g(x) g(x') d\mu(x) d\mu(x') \geq 0 \quad \text{for all } g \text{ such that } ||g(x)||_{L_2} < \infty \tag{37}$$

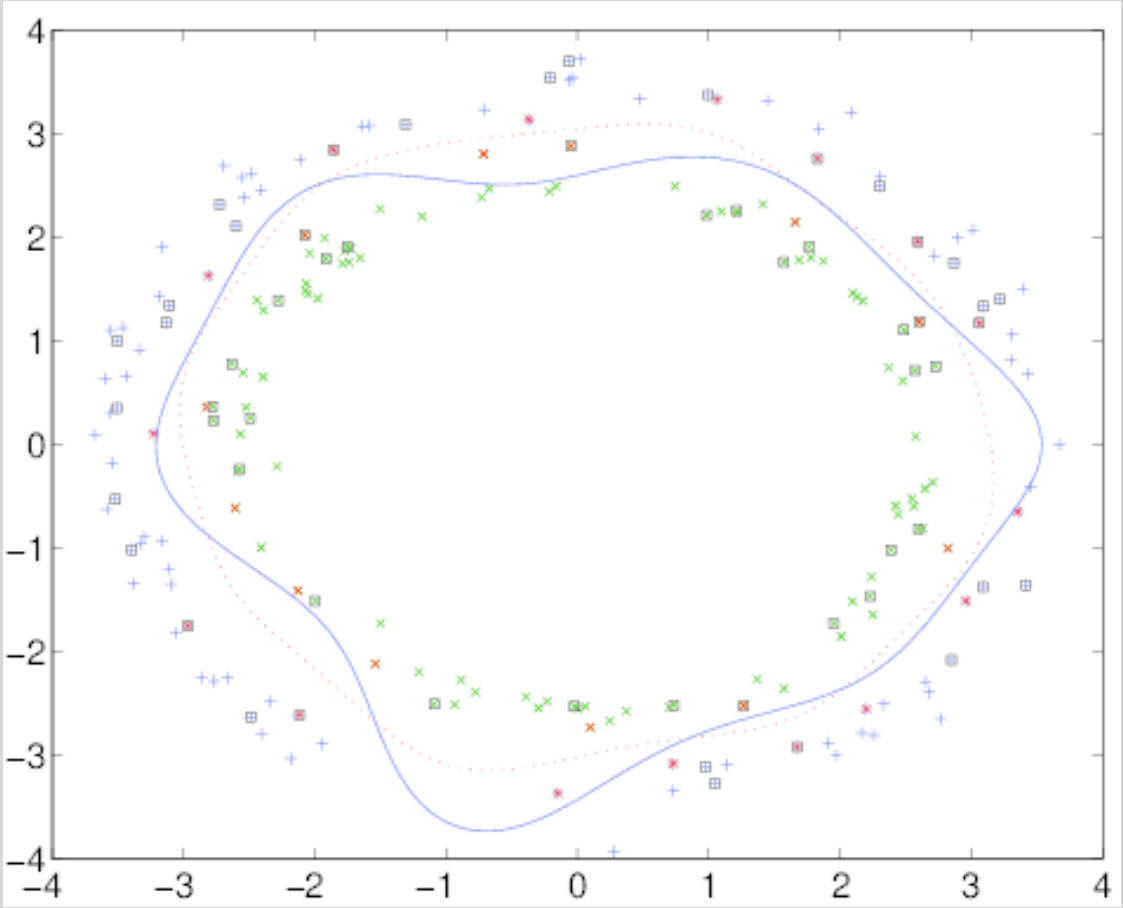
- In other words, K must be a positive semidefinite operator on L_2.
- If K satisfies the Mercer condition, there is no guarantee that the corresponding ϕ is unique, or that it is finite-dimensional.

Quadratic kernel



- C-SVM, polynomial degree 2 kernel, $n = 200$, $C = 10000$
- The two ellipses show that a constant shift to the data ($x^i \leftarrow x^i + v$, $v \in \mathbb{R}^n$) can affect non-linear kernel classifiers.

RBF kernel and Support Vectors



Prediction with SVM

- Estimating b
  - For any i support vector, $w^T x^i + b = y^i$ because the classification is tight
  - Alternatively, if there are slack variables, $w^T x^i + b = y^i(1 - \xi_i)$
  - Hence, $b = y^i(1 - \xi_i) - w^T x^i$

  - For non-linear SVM, where w is not known explicitly, $w = \sum_j \alpha_j y^j \phi(x_j)$. Hence, $b = y^i(1 - \xi_i) - \sum_{j=1}^n \alpha_j y^j K(x^i, x^j)$ for any i support vector

- Given new x

$$\hat{y}(x) = \text{sgn}(w^T x + b) = \text{sgn}\left(\sum_{i=1}^n \alpha_i y^i K(x^i, x) + b\right). \quad (38)$$

L1-SVM

- If the regularization $||w||^2$, based on l_2 norm, is replaced with the l_1 norm $||w||_1$, we obtain what is known as the **Linear L1-SVM**

$$\min_{w,b} ||w||_1 + C\sum_i \xi_i \quad \text{s.t } y^i(w^T x^i + b) \geq 1 - \xi_i,\ \xi_i \geq 0 \text{ for all } i = 1:n \tag{39}$$

- The use of the l_1 norm promotes sparsity in the entries of w
- The **Non-linear L1-SVM** is

$$f(x) = \sum_i (\alpha_i^+ + \alpha_i^-) y^i K(x_i, x) + b \quad \text{classifier} \tag{40}$$

$$\min_{\alpha_\pm, b} \sum_i (\alpha_i^+ + \alpha_i^-) + C\sum_i \xi_i \quad \text{s.t } y^i f(x^i) \geq 1 - \xi_i,\ \xi_i, \alpha_i^\pm \geq 0 \text{ for all } i = 1:n \tag{41}$$

- This formulation enforces $\alpha_i^+ = 0$ or $\alpha_i^- = 0$ for all i. If we set $w_i = \alpha_i^+ - \alpha_i^-$, we can write $f(x) = \sum_i w_i y^i K(x^i, x) + b$, a linear classifier in the non-linear features $K(x^i, x)$.
- The L1-SVM problems are Linear Programs
- The dual L1-SVM problems are also linear programs
- The L1-SVM is no longer a Maximum Margin classifier

Multi-class and One class SVM

Multiclass SVM

For a problem with K possible classes, we construct K separating hyperplanes $w_r^T x + b_r = 0$.

$$\text{minimize} \quad \frac{1}{2}\sum_{r=1}^{K} ||w_r||^2 + \frac{C}{n}\sum_{i,r}\xi_{i,r} \tag{42}$$

$$\text{s.t.} \quad w_{y^i}^T x^i + b_{y^i} \geq w_r^T x^i + b_r + 1 - \xi_{i,r} \quad \text{for all } i = 1:n,\ r \neq y^i \tag{43}$$

$$\xi_{i,r} \geq 0 \tag{44}$$

One-class SVM This SVM finds the "support regions" of the data, by separating the data from the origin by a hyperplane. It's mostly used with the Gaussian kernel, that projects the data on the unit sphere. The formulation below is identical to the ν-SVM where all points have label 1.

$$\text{minimize} \quad \frac{1}{2}||w||^2 - \nu\rho + \frac{1}{n}\sum_{i}\xi_i \tag{45}$$

$$\text{s.t.} \quad w^T x^i + b \geq \rho - \xi_i \tag{46}$$

$$\xi_i \geq 0 \tag{47}$$

$$\rho \geq 0 \tag{48}$$

SV Regression

The idea is to construct a "tolerance interval" of $\pm\epsilon$ around the regressor f and to penalize data points for being outside this tolerance margin. In words, we try to construct the smoothest function that goes within ϵ of the data points.

$$\text{minimize} \quad \frac{1}{2}||w||^2 + C\sum_i(\xi_i^+ + \xi_i^-) \tag{49}$$

$$\text{s.t.} \quad \epsilon + \xi_i^+ \geq w^T x^i + b - y^i \geq -\epsilon - \xi_i^- \tag{50}$$

$$\xi_i^{\pm} \geq 0 \tag{51}$$

$$\rho \geq 0 \tag{52}$$

The above problem is a linear regression, but with the kernel trick we obtain a kernel regressor of the form $f(x) = \sum_i(\alpha_i^- - \alpha_i^+)K(x^i, x) + b$