Support Vector Machine

Presented by: Wei Fan

School of the Gifted Young
University of Science and Technology of China

2020.10

Focus

SV Classifiers - Introduction

We consider the classification problem that seperate N points into two different classes:

- We want to seperate them by a linear boundary (hyperplane)
- In many cases, there're infinitely many choices of hyperplanes that seperate all the data correctly, as the figure below shows.
- Our intuition leads us thinking about the least square solution, however it does not do a perfect job in seperating the points. Misclassification may occur, also shown in the figure below.
- Therefore our target is to find a hyperplane that have the largest distance to the point that is closest.

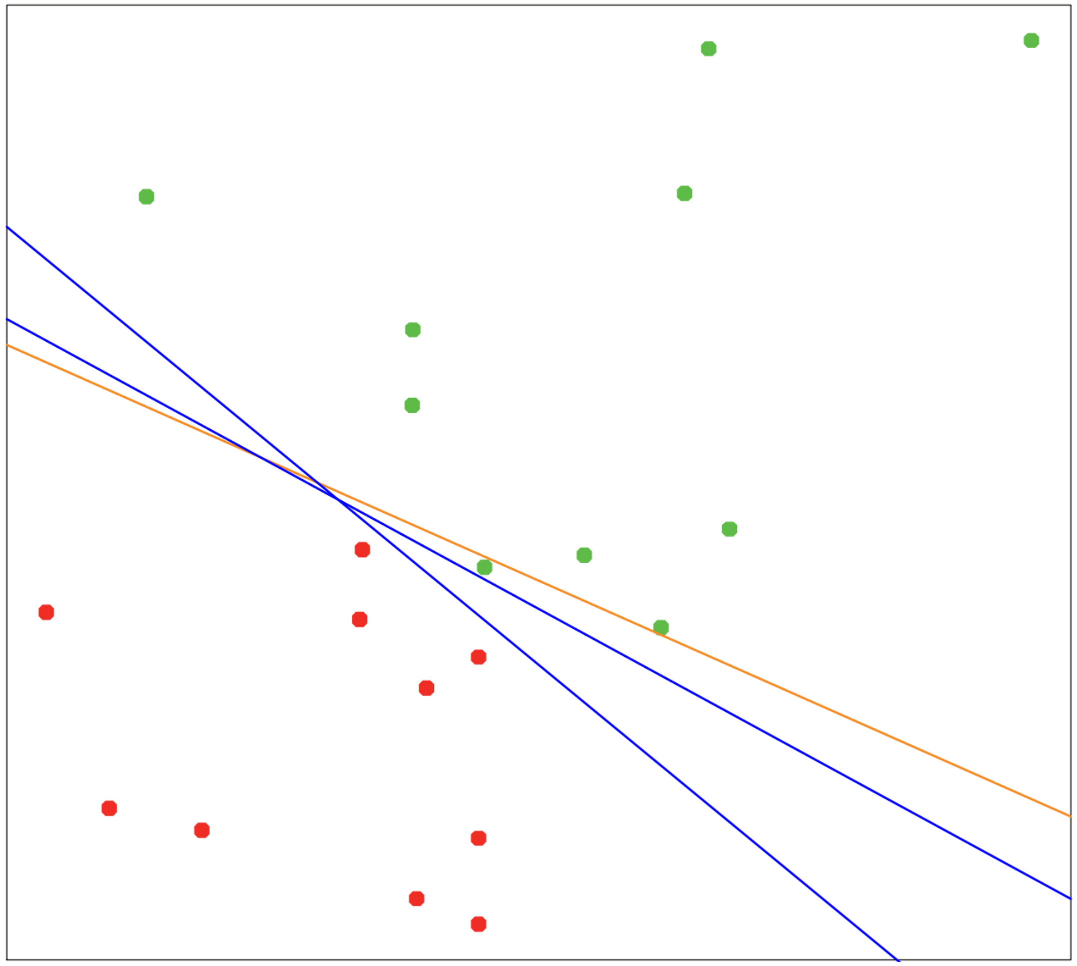


FIGURE 4.14. *A toy example with two classes separable by a hyperplane. The orange line is the least squares solution, which misclassifies one of the training points. Also shown are two blue separating hyperplanes found by the* perceptron learning algorithm *with different random starts.*

I.1. SV Classifier for Linear Seperable Case

- **Consider our data consisting of N pairs** $(x_1, y_1), (x_2, y_2) \cdots (x_N, y_N)$ **with** $x_i \in \mathbb{R}^p$ **and** $y_i \in \{-1, 1\}$**.**
- **We classify the data using a hyperplane** $\{x : f(x) = x^T\beta + \beta_0 = 0\}$ **where** $\|\beta\| = 1$**. It induces a classification rule** $G(x) =$ **sign**$[x^T\beta + \beta_0]$
- **Since the data is linear separable, we're able to find a function** $f(x) = x^T\beta + \beta_0$ **with** $y_i f(x_i) > 0\ \forall i$**. Hence our target is to find a function creates the biggest margin between the training points.**

Then we normalize the optimization problem as below:

$$\begin{cases} \displaystyle\max_{\beta,\beta_0,\|\beta\|=1} M \\ \textbf{subject to } y_i(x_i^T\beta + \beta_0) \geq M, i = 1, 2 \cdots N \end{cases} \tag{1}$$

Let $\|\beta\| = 1/M$ **instead of 1, the problem can be more conveniently repharased as:**

$$\begin{cases} \displaystyle\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 \\ \textbf{subject to } y_i(x_i^T\beta + \beta_0) \geq 1, i = 1, 2 \cdots N \end{cases} \tag{2}$$

We will characterize the solution below: The corresponding Lagrange primal function is:

$$L_P = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(x_i^T \beta + \beta_0) - 1] \tag{3}$$

Setting the derivative to 0, we have:

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i \tag{4}$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i \tag{5}$$

Therefore, the Lagrange Dual function is:

$$L_D = \sum_{i=1}^{N} \alpha_i - \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{1}{2} \alpha_i \alpha_k y_i y_k x_i^T x_k \tag{6}$$

Besides, the solution must satisfies Karush-Kuhn-Tucker conditions, to which we make a brief introduction below.

Consider our optimization problem:

$$\begin{cases} \textbf{Optimize } f(x) \\ \textbf{subject to } g_i(x) \geq 0, h_j(x) = 0 \textbf{ for } i = 1:m,\ j = 1:l \end{cases} \tag{7}$$

Then the Lagrange primal function becomes

$$L(x, \mu, \lambda) = f(x) - \mu^T g(x) - \lambda^T h(x) \tag{8}$$

If x^* is the local minimize of the optimization problem, it follows:

- **Stationarity**

$$\nabla f(x^*) = \sum_{i=1}^{m} \mu_i \nabla g_i(x^*) + \sum_{j=1}^{l} \lambda_j \nabla h_j(x^*) \tag{9}$$

- **Primal feasibility**

$$g_i(x^*) \geq 0, \ \textbf{for } i = 1, \cdots, m \tag{10}$$

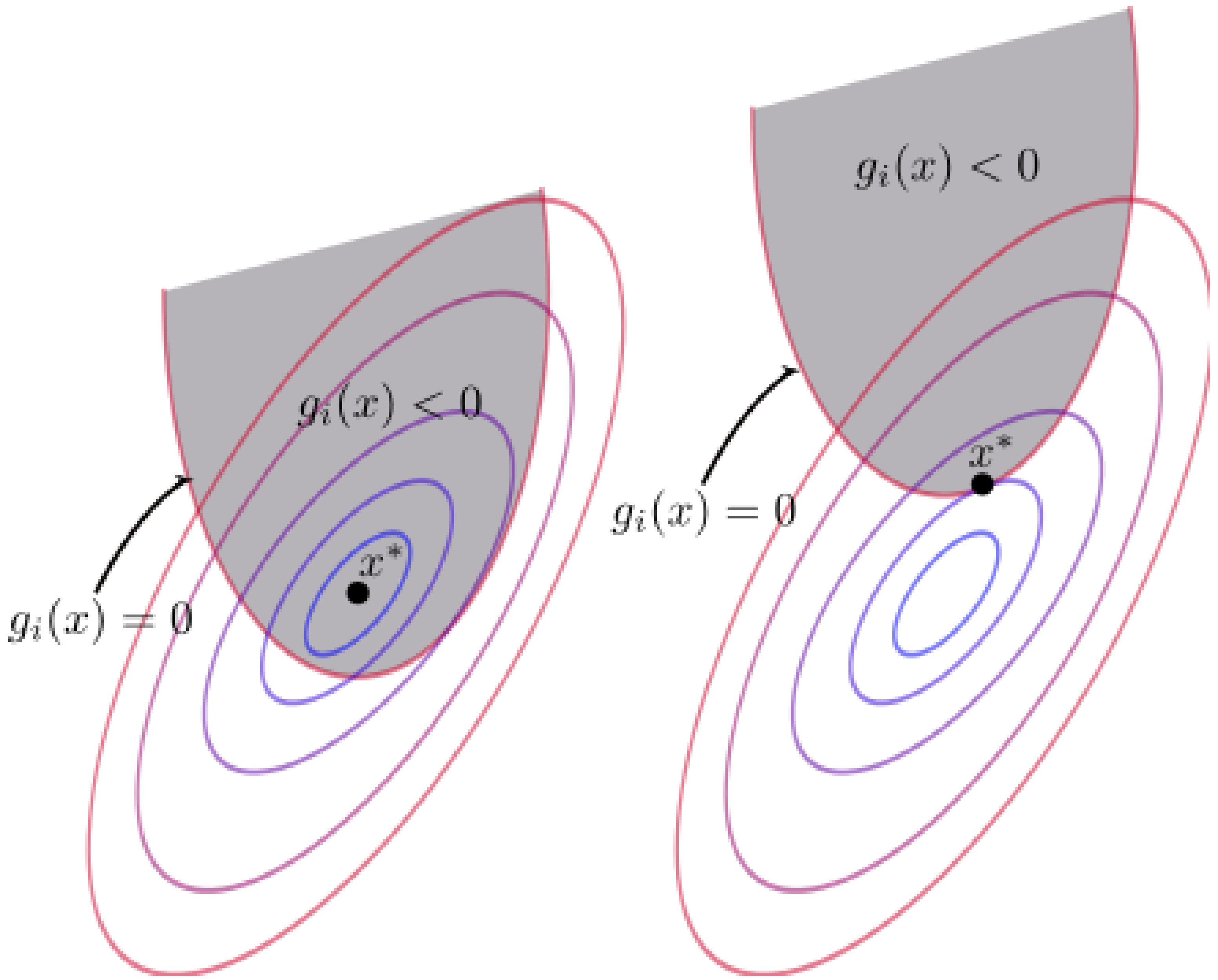
$$h_j(x^*) = 0, \ \textbf{for } j = 1, \cdots, l \tag{11}$$

- **Dual feasibility**

$$\mu_i \geq 0, \ \textbf{for } i = 1, \cdots, m \tag{12}$$

- **Complementary slackness**

$$\sum_{i=1}^{m} \mu_i g_i(x^*) = 0 \tag{13}$$


$g_i(x) < 0$
x^*
$g_i(x) = 0$
$g_i(x) < 0$
x^*
$g_i(x) = 0$

As a result, in this problem, the KKT condition includes (4), (5), (6) and

$$\alpha_i[y_i(x_i^T\beta + \beta_0) - 1] = 0 \ \forall i \tag{14}$$

We may see from that:

- **If** $\alpha_i > 0$**, then** $y_i(x_i^T\beta + \beta_0) = 1$**, implying that** x_i **is on the boundary**
- **If** $y_i(x_i^T\beta + \beta_0) > 1$**,** x_i **is not on the boundary, then** $\alpha_i = 0$

We see β **is a linear combination of** x_i**, which is on the boundary with** $\alpha_i > 0$**.**

Suppose the optimal seperating hyperplane has function $\hat{f}(x) = x^T\hat{\beta} + \hat{\beta}_0$**, which gives a classification function:**

$$\hat{G}(x) = \textbf{sign}[\hat{f}(x)] \tag{15}$$

I.2. SV Classifier for Non-Seperable Linear Case

- **Situation: Suppose now the classes overlap in feature space**
- **Method: Define the slack variable:** $\xi = (\xi_1, \xi_2, \ldots, \xi_N)$
- **There are two ways to modify the constraint:**

$$y_i(x_i^T \beta + \beta_0) \geq M - \xi_i \tag{16}$$

$$or$$

$$y_i(x_i^T \beta + \beta_0) \geq M(1 - \xi_i) \tag{17}$$

with limitation that $\forall i\ \xi_i \geq 0$, $\sum_{i=1}^{N} \xi_i \leq C$ **for some constant C.**

The first choice seems more natural, however it results in a nonconvex optimization problem. The second is convex. Therefore we use the method from then on. The idea of the formulation has following meanings:

- **Bounding** $\sum \xi_i$ **means bounding the total amount that the prediction falls on the wrong side.**
- **Misclassification occurs when** $\xi_i > 1$, **so bounding** $\sum \xi_i$ **also bounding the total number of misclassifications at K.**

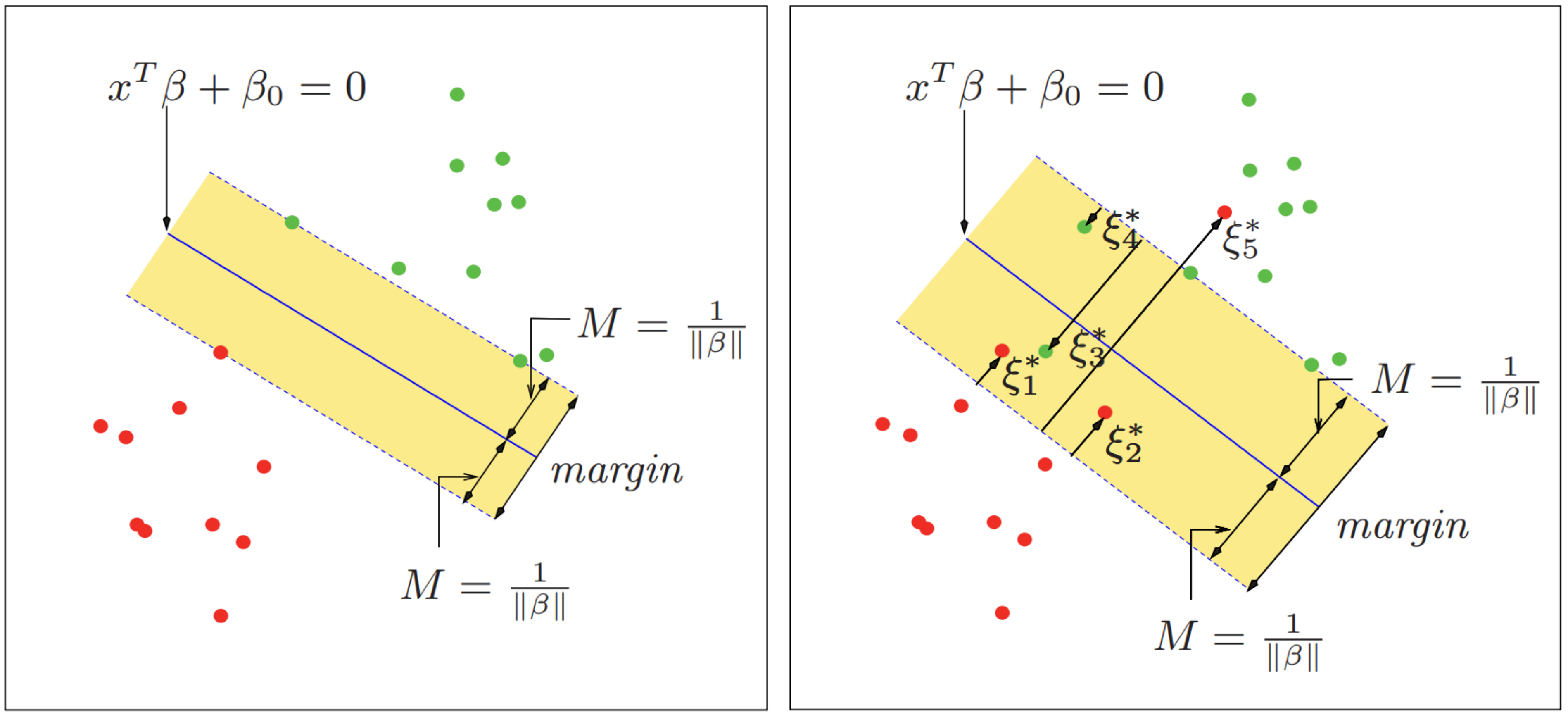


FIGURE 12.1. *Support vector classifiers. The left panel shows the separable case. The decision boundary is the solid line, while broken lines bound the shaded maximal margin of width $2M = 2/\|\beta\|$. The right panel shows the nonseparable (overlap) case. The points labeled ξ_j^* are on the wrong side of their margin by an amount $\xi_j^* = M\xi_j$; points on the correct side have $\xi_j^* = 0$. The margin is maximized subject to a total budget $\sum \xi_i \leq$ constant. Hence $\sum \xi_j^*$ is the total distance of points on the wrong side of their margin.*

We may express the optimization problem as below:

$$\begin{cases} \min\limits_{\beta,\beta_0} \|\beta\| \\ \textbf{subject to } y_i(x_i^T\beta+\beta_0) \geq 1-\xi_i,\ \ \xi_i \geq 0,\ \ \sum \xi_i \leq C\ \forall i \end{cases} \tag{18}$$

Computationally it's easier to solve the problem in the following form:

$$\begin{cases} \min\limits_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 + K\sum_{i=1}^{N}\xi_i \\ \textbf{subject to } \xi_i \geq 0,\ y_i(x_i^T\beta+\beta_0) \geq 1-\xi_i\ \forall i \end{cases} \tag{19}$$

The seperable case corresponds to $C = \infty$ The Lagrange primal function is:

$$L_P = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i[y_i(x_i^T\beta+\beta_0) - (1-\xi_i)] - \sum_{i=1}^{N}\mu_i\xi_i \tag{20}$$

Setting the derivative to 0,we get:

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i \tag{21}$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i \tag{22}$$

$$\alpha_i = C - \mu_i \tag{23}$$

We substitue these 3 equations into Lagrange primal function,obtaining the Lagrangian dual objective function:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j \tag{24}$$

In addition, the Karush-Kuhn-Tucker conditions include the constraints:

$$\alpha_i [y_i (x_i^T \beta + \beta_0) - (1 - \xi_i)] = 0 \tag{25}$$

$$\mu_i \xi_i = 0 \tag{26}$$

$$y_i (x_i^T \beta + \beta_0) - (1 - \xi_i) \geq 0 \tag{27}$$

Combining the equations above, we see that the solution for β has the form:

$$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i \tag{28}$$

We have following observations about the solution:

- $\hat{\alpha}_i \neq 0 \iff y_i(x_i^T\beta + \beta_0) - (1 - \xi_i) = 0$**, which means the constraints are exactly met. We call those $\hat{\alpha}_i$ support vectors.**
- **Among the support vectors, if $\hat{\xi}_i = 0$ then $0 < \hat{\alpha}_i < c$; if $\hat{\xi}_i > 0$ then $\hat{\alpha}_i = c$.**
- **Any of the margin points $(\hat{\alpha}_i > 0, \hat{\xi}_i = 0)$ can be used to solve for β_0**

Given the solution to $\hat{\beta}_0$ and $\hat{\beta}$, the decision function may be written as:

$$\hat{G}(x) = \mathbf{sign}[x^T\hat{\beta} + \hat{\beta}_0] \tag{29}$$

Focus

II. SVM and Kernels - Introduction

- **We make the procedure more flexible with basis expansion (Chapter 5).**
- **Generally the linear boundaries in the enlarged space achieve better training-class seperation, i.e. it may make some non-seperable data in the original space seperable, then translate to the nonlinear boundaries in the original space.**
- **We use basis function $h(x_i)$ as input features to fit the function, producing the non-linear function $\hat{f}(x) = h(x)^T\hat{\beta} + \hat{\beta}_0$.**
- **The classifier is $\hat{G}(x) = \textbf{sign}(\hat{f}(x))$ as before.**

II.1. SVM for Classification

Using basis function $h(x_i)$, the Lagrange dual function has the form:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \langle h(x_i), h(x_j) \rangle \tag{30}$$

Thus the solution function $f(x)$ can be written as:

$$f(x) = h(x)^T \beta + \beta_0 = \sum_{i=1}^{N} \alpha_i y_i \langle h(x), h(x_i) \rangle + \beta_0 = \sum_{i=1}^{N} \alpha_i y_i K(x, x_i) + \beta_0 \tag{31}$$

In fact, we need not specify the transformation $h(x)$ at all, require only the knowledge of kernel function

$$K(x, x^{'}) = \langle h(x), h(x') \rangle \tag{32}$$

Three popular choices of K in SVM literature:

- **dth-degree polynomial:** $K(x, x^{'}) = (1 + \langle x, x' \rangle)^d$
- **Radial basis:** $K(x, x^{'}) = \exp(-\eta \|x - x'\|^2)$
- **Neural network:** $K(x, x') = \tanh(\kappa_1 \langle x, x' \rangle + \kappa_2)$

Example

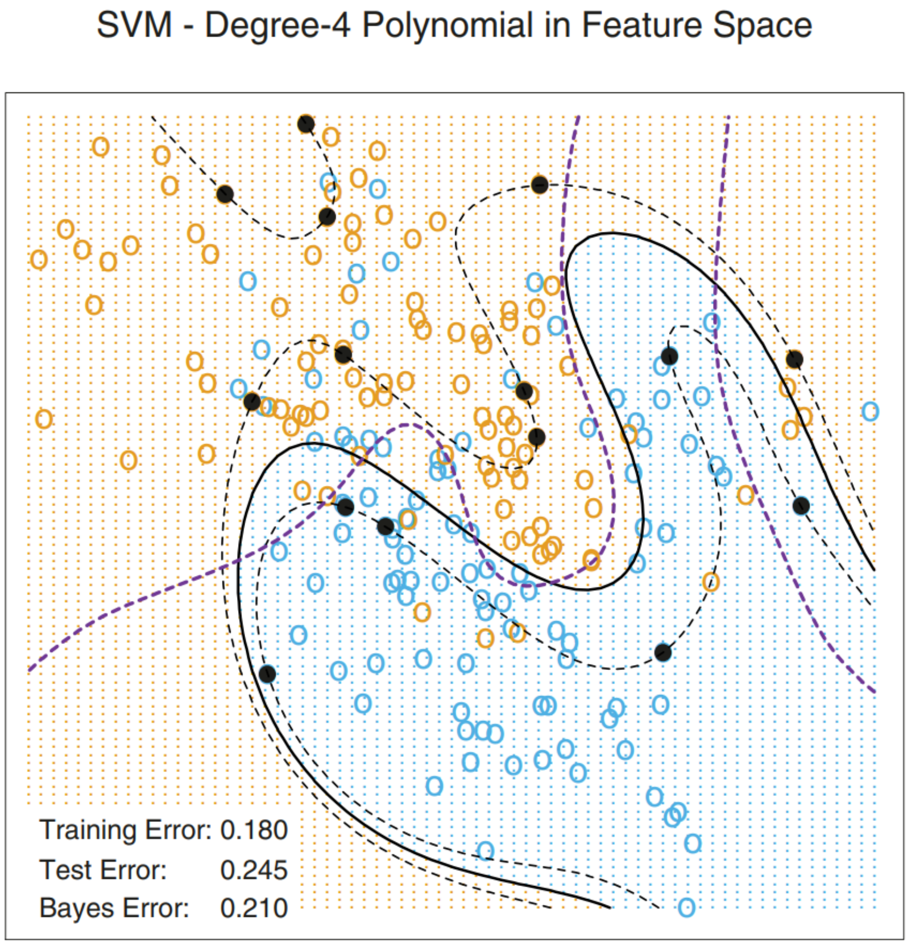
Consider a polynomial kernel of degree 2

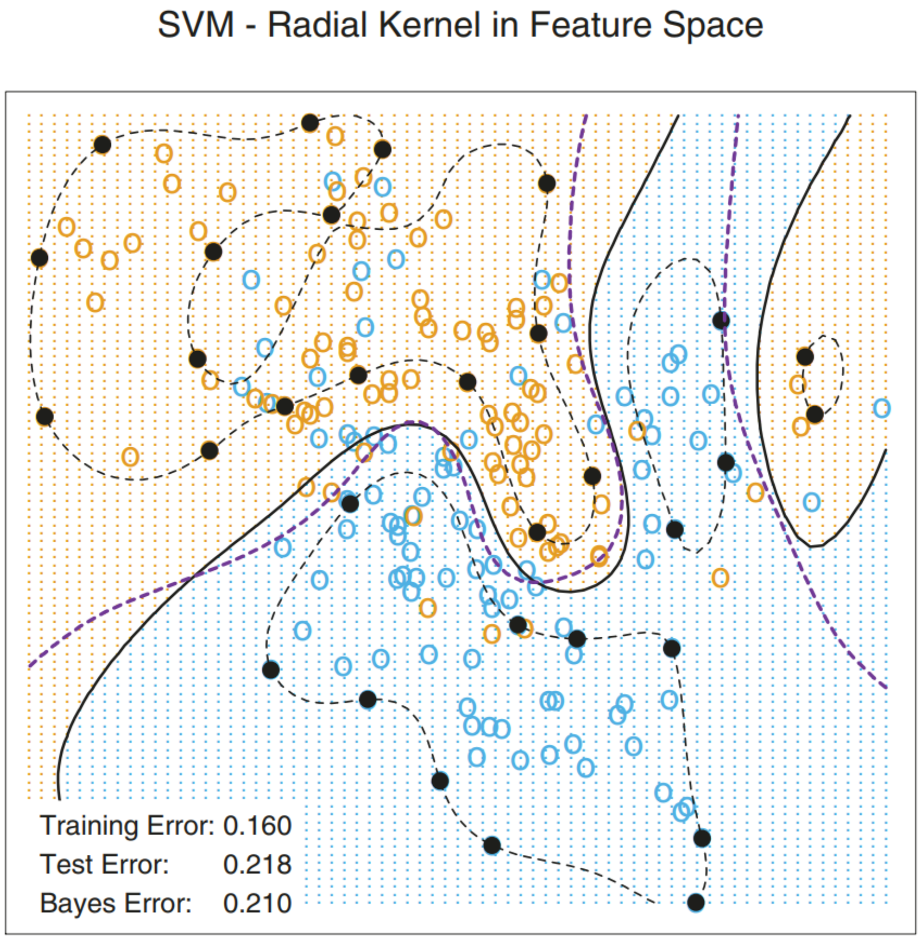
- **Choose** $h_1(X) = 1$, $h_2(X) = \sqrt{2}X_1$, $h_3(X) = \sqrt{2}X_2$, $h_4(X) = X_1^2$, $h_5(X) = X_2^2$ **and** $h_6(X) = \sqrt{2}X_1X_2$
- **The kernel function:** $K(X, X^{'}) = \langle h(X), h(X^{'}) \rangle$
- **The solution can be written as:** $\hat{f}(x) = \sum_{i=1}^{N} \hat{\alpha}_i y_i K(x, x_i) + \hat{\beta}_0$

The role of parameter C:

- **Large value of C: Discourage any positive** ξ_i**, leading to overfit wiggly boundary**
- **Small value of C: Encourage small value of** $||\beta||$**, the boundary become smoother**

The figure below shows two nonlinear support vector machine applied to mixture example in Chapter 2.


SVM - Degree-4 Polynomial in Feature Space
Training Error: 0.180
Test Error: 0.245
Bayes Error: 0.210


SVM - Radial Kernel in Feature Space
Training Error: 0.160
Test Error: 0.218
Bayes Error: 0.210

II.2. The SVM as a Penalization Method

With $f(x) = h(x)^T\beta + \beta_0$, we consider the following problem:

$$\min_{\beta,\beta_0} \sum_{i=1}^{N} [1 - y_i f(x_i)]_+ + \frac{\lambda}{2}\|\beta\|^2 \tag{33}$$

Next we'll show that this optimization problem is the same as (19):

Proof. With $\lambda = 1/C$, $f(x) = h(x)^T\beta + \beta_0$, we get:

$$\min_{\beta_0,\beta} C \sum_{i=1}^{N} [1 - y_i(h(x_i)^T\beta + \beta_0)]_+ + \frac{1}{2}\|\beta\|^2 \tag{34}$$

However, this is non-differentiable. Therefore we need to introduce slack variable ξ_i, then we can show that the problem is equivalent to solving:

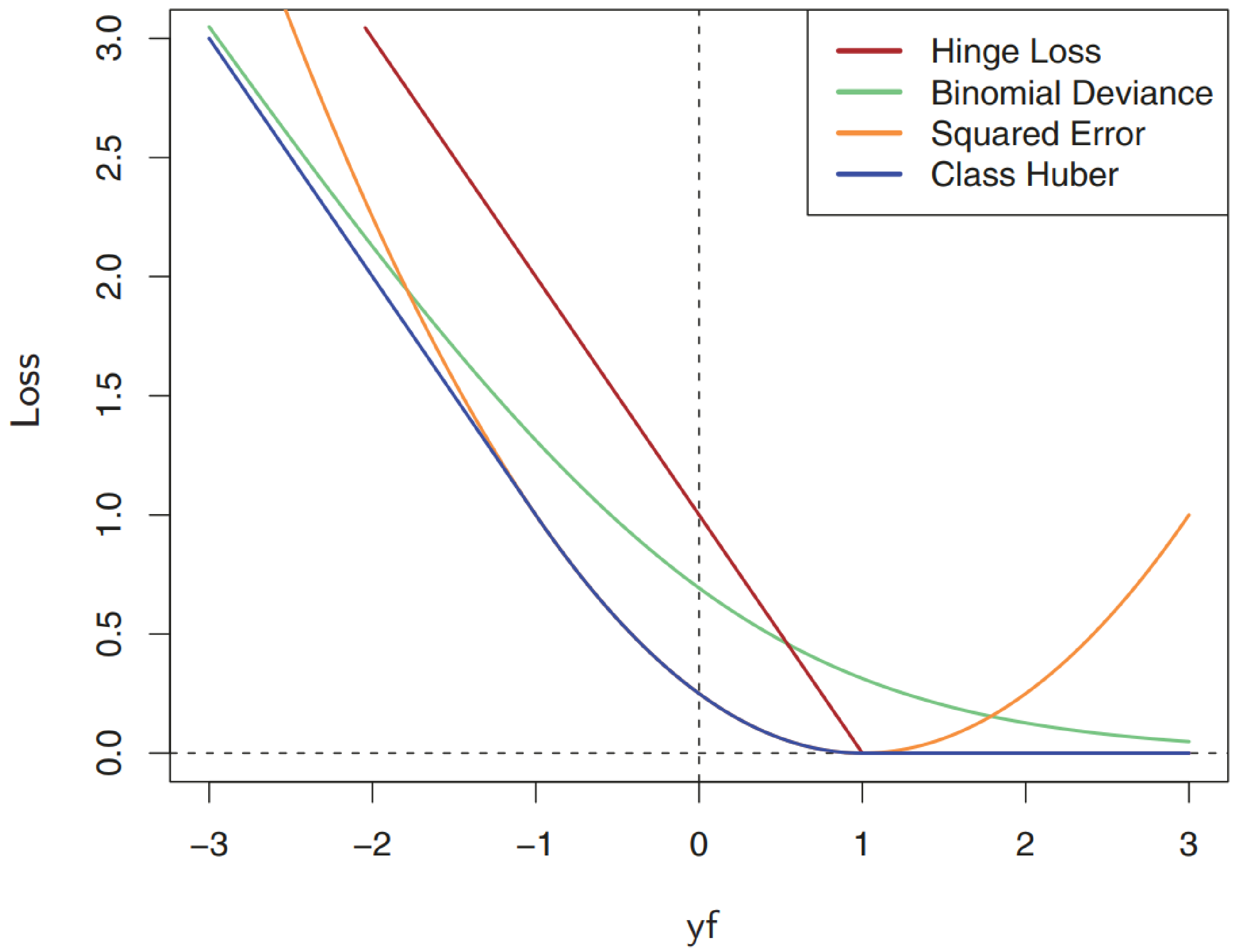
$$\min_{\beta_0,\beta} C \sum_{i=1}^{N} \xi_i + \frac{1}{2}\|\beta\|^2 \quad \text{subject to} : \xi_i \geq 0,\ y_i(h(x_i)^T\beta + \beta_0) \geq 1 - \xi_i \tag{35}$$

The problem is same as (19), only difference in x and $h(x)$, the solution of which is illustrated before. ■

We have following obeservations about ”hinge” loss function:

- **The ”hinge” loss function is not differentiable, so it’s difficult to optimize the target function directly.**
- **To figure out the dilemma, we usually set a bound for the ”hinge” loss function (e.g. bounding that $1 - y_i(x^T\beta + \beta_0) \geq \xi_i \geq 0$ in the previous example). We call the function with a bound surrogate loss function.**
- **The ”hinge” loss function has a higher demand to learning, only when the classification is right and credible can the loss function equals 0.**
- **The ”hinge” loss function is more reasonable for two-class classification.**

We compare ”hinge” loss function with other kinds of loss functions in the table below:


Hinge Loss
Binomial Deviance
Squared Error
Class Huber
Loss
0.0
0.5
1.0
1.5
2.0
2.5
3.0
−3
−2
−1
0
1
2
3
yf

Focus

A.1. SVMs and the Curse of Dimensionality

- **Shortcomings of SVM: Not easy to detect the structure of data.**
- **Example: Consider the polynomial of degree 2, then all term of $2X_iX_j'$ are given an equal weight.**
- **If the dimension p were large and the class seperation occur only in the linear subspace spanned by X_1 and X_2, the kernel would have to search over many dimensions.**
- **Target: Build knowledge about the subspace into the kernel, we first need to discover the structure of the data.**

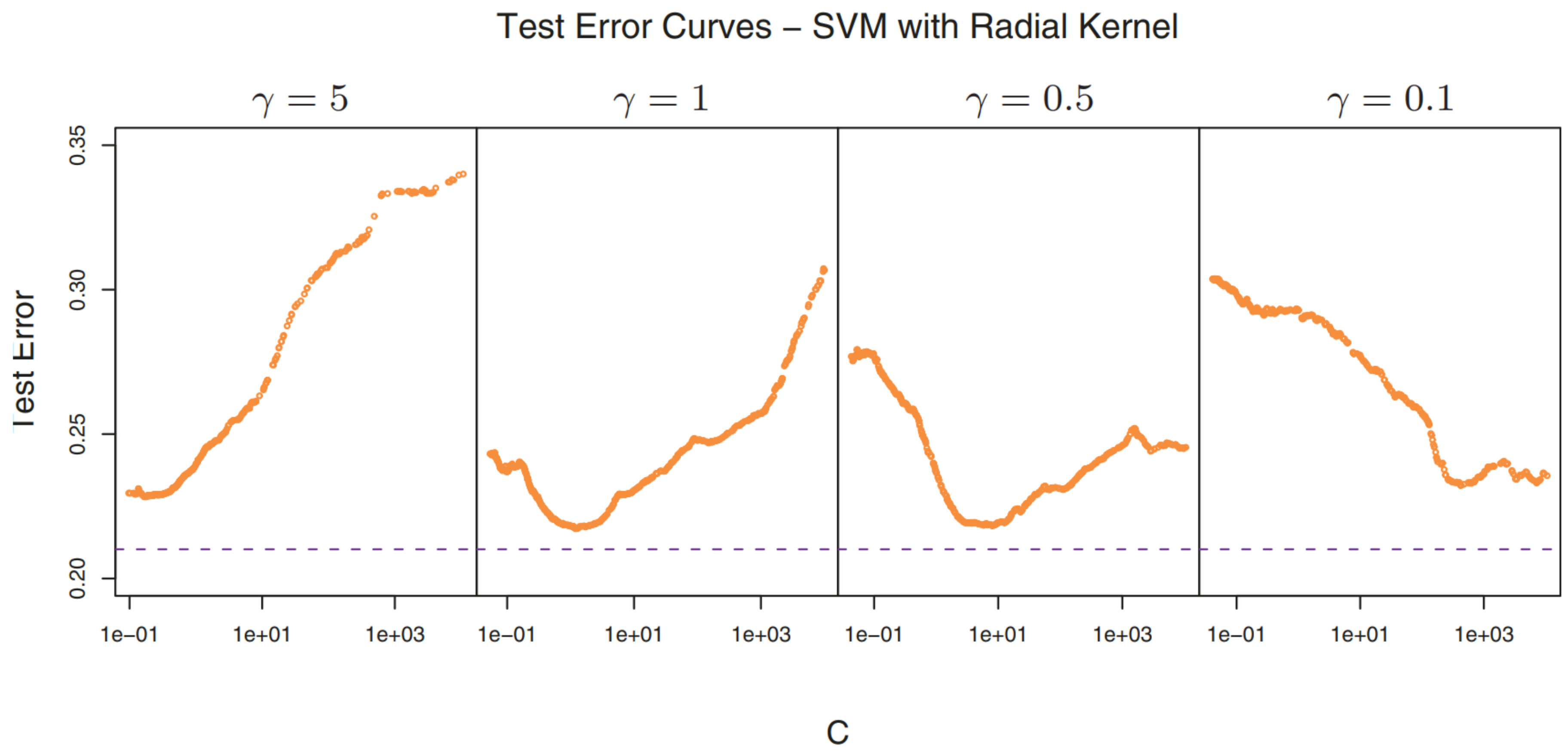
In following example we have some obeservations that support the theory:

- **The SV classifiers does poorly since the hyperplane cannot seperate classes.**
- **The polynomial SVM made substantial improvement.**
- **However the high-degree polynomial does worse because of the curse of dimensionality.**

TABLE 12.2. *Skin of the orange: Shown are mean (standard error of the mean) of the test error over* 50 *simulations. BRUTO fits an additive spline model adaptively, while MARS fits a low-order interaction model adaptively.*

	Method	Test Error (SE)	
		No Noise Features	Six Noise Features
1	SV Classifier	0.450 (0.003)	0.472 (0.003)
2	SVM/poly 2	0.078 (0.003)	0.152 (0.004)
3	SVM/poly 5	0.180 (0.004)	0.370 (0.004)
4	SVM/poly 10	0.230 (0.003)	0.434 (0.002)
5	BRUTO	0.084 (0.003)	0.090 (0.003)
6	MARS	0.156 (0.004)	0.173 (0.005)
	Bayes	0.029	0.029

A.2. A Path Algorithm for the SVM Classifier



The figure above shows test error of the data with different choice of C and γ in the radial kernel. It is shown that as γ=5, small C is required, while as $\gamma = 1$ an intermediate value of C is required.

Now we describe a path algorithm for efficiently fitting the entire sequence of SVM models by varying C:

- **Method: Use the loss+penalty formulation (20), along with the figure in pg.16**
- **It leads to a solution for β at given value of λ:**

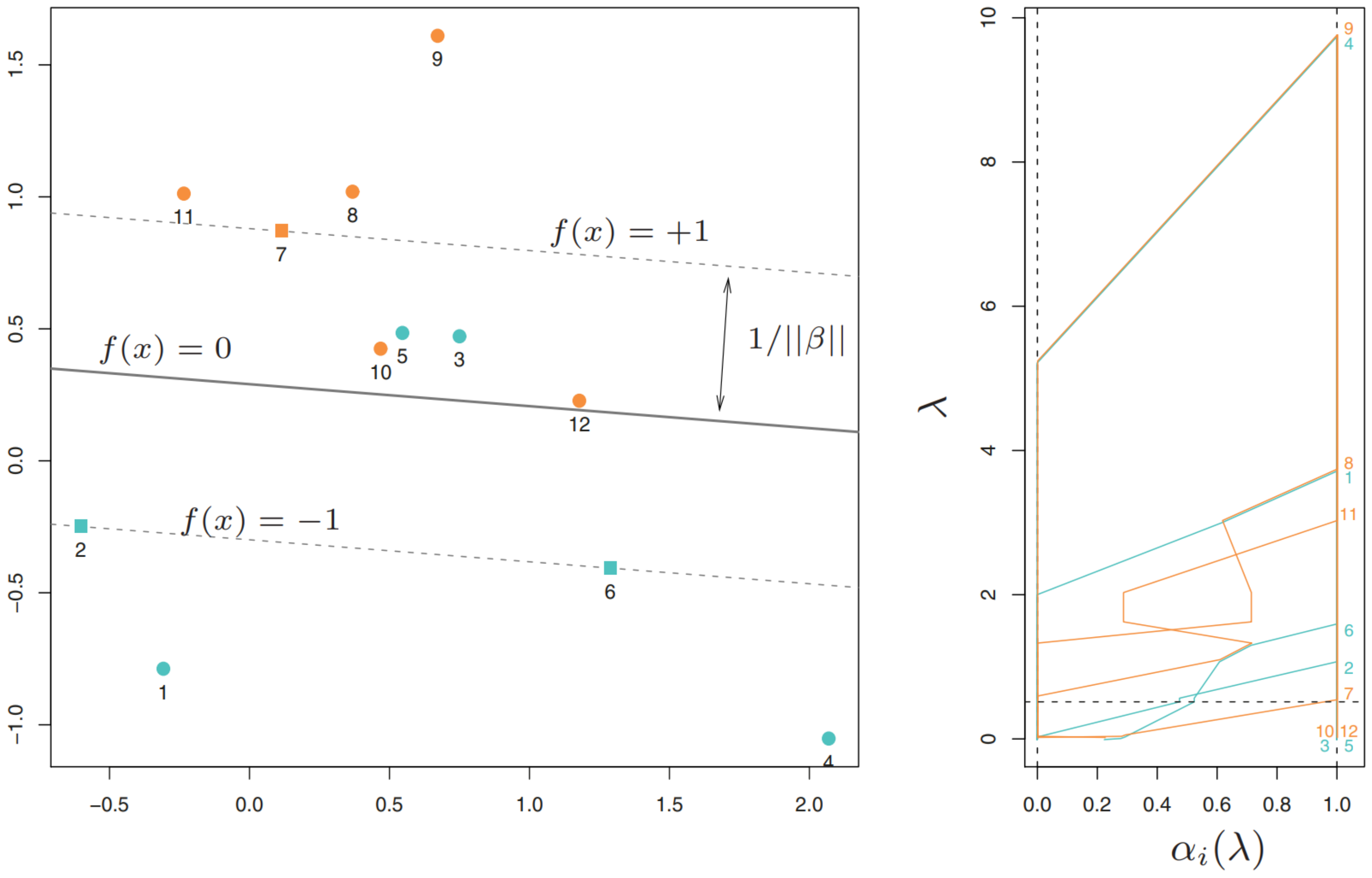
$$\beta_\lambda = \frac{1}{\lambda}\sum_{i=1}^{N}\alpha_i x_i y_i \tag{36}$$

  With α_i be Lagrange multipliers, in this case they're all in [0,1].

- **The Figure below shows the setup.**

It can be shown that KKT optimality conditions imply that (x_i, y_i) fall into three distinct groups:

- **Observations outside their margins: They have $y_i f(x_i) > 1$, and Lagrange multipliers $\alpha_i = 0$. Examples are points 1,4,8,9,11.**
- **Obeservations sitting on their margins: They have $y_i f(x_i) = 1$, with Lagrange multipliers $\alpha_i \in [0, 1]$. Examples are points 2,6,7.**
- **Observations inside their margins: They have $y_i f(x_i) < 1$, with Lagrange multipliers $\alpha_i = 1$. Examples are points 3,5,10,12.**



$f(x) = +1$
$f(x) = 0$
$f(x) = -1$
$1/||\beta||$
λ
$\alpha_i(\lambda)$

Now we show the idea for the path algorithm:

- **Set initially large λ, therefore the margin $1/\|\beta_\lambda\|$ is large and all points have $\alpha_i = 1$.**
- **Decrease λ, therefore the margin gets narrower. Some points move to outside the margin, thus α_i change from 1 to 0.**
- **By continuity of $\alpha_i(\lambda)$, some points will linger on the margin during transition.**
- **By (23) we sees that points with $\alpha_i = 1$ made fixed contribution to β_λ, while those points with $\alpha_i = 0$ had no contribution. So all that changes as λ decreases are those points on the margin.**
- **Since $y_i f(x_i) = 1$, we may use certain linear functions to describe how $\alpha_i(\lambda)$ and β_λ changes, as shown in the Figure above.**

For nonlinear model, we may apply exactly the same idea:

$$f_\lambda(x) = \frac{1}{\lambda}\sum_{i=1}^{N}\alpha_i y_i K(x, x_i) \tag{37}$$

A.3. SVM for Regression

Now we want to discuss linear regression model $f(x) = x^T\beta + \beta_0$ and then handle nonlinear genralizations.

To estimate β, consider the minimization of

$$H(\beta, \beta_0) = \sum_{i=1}^{N} V(y_i - f(x_i)) + \frac{\lambda}{2}\|\beta\|^2 \tag{38}$$

We have following two ways in choosing function V:

$$V_\epsilon(r) = \begin{cases} 0 & |r| \leq \epsilon \\ |r| - \epsilon & |r| > \epsilon \end{cases} \tag{39}$$

$$V_H(r) = \begin{cases} r^2/2 & |r| \leq c \\ c|r| - c^2/2 & |r| > c \end{cases} \tag{40}$$

- The first choice ignores errors size less than ϵ, so it's a rough analogy, the "low error" points are the ones with small residuals.
- The second choice is used in robust regression, since it reduces quadratic to linear for residuals greater than c, so it's made less sensitive to outliers. It differs from the first choice in not flattening small residuals.

Let $\hat{\beta}$, $\hat{\beta}_0$ be the minimizers of H, the solution functions have the form:

$$\hat{\beta} = \sum_{i=1}^{N} (\hat{\alpha}_i{}^* - \hat{\alpha}_i) x_i \tag{41}$$

$$\hat{f}(x) = \sum_{i=1}^{N} (\hat{\alpha}_i{}^* - \hat{\alpha}_i) \langle x, x_i \rangle + \beta_0 \tag{42}$$

With $\hat{\alpha}_i$, $\hat{\alpha}_i{}^*$ are positive and solve following optimization problem:

$$\min_{\alpha_i, \alpha_i^*} \epsilon \sum_{i=1}^{N} (\alpha_i + \alpha_i^*) - \sum_{i=1}^{N} y_i (\alpha_i^* - \alpha_i) + \frac{1}{2} \sum_{i,i'=1}^{N} (\alpha_i^* - \alpha_i)(\alpha_{i'}^* - \alpha_{i'}) \langle x_i, x_{i'} \rangle \tag{43}$$

subject to the constraints:

$$0 \le \alpha_i, \alpha_i^* \le 1/\lambda \tag{44}$$

$$\sum_{i=1}^{N} (\alpha_i^* - \alpha_i) = 0 \tag{45}$$

$$\alpha_i \alpha_i^* = 0 \tag{46}$$

- **Usually only a subset of the values $(\hat{\alpha}_i{}^* - \hat{\alpha}_i)$ are nonzero, and associated data values are called support vectors.**
- **We may generalize the methods to richer spaces by defining an appropriate inner product (e.g. Polynomial/Radial basis/Neural network).**