Principles of Machine Learning

Lecture 6: Classification with Support Vector Machines and Tree-based Methods

Sharif University of Technology
Dept. of Aerospace Engineering

April 13, 2025

Table of Contents

Overview of Support Vector Machines

- Support Vector Machines (SVMs) are classification methods developed in the 1990s.
- SVMs generalize the maximal margin classifier to handle non-linear boundaries.
- The maximal margin classifier assumes classes are separable by a linear boundary.
- SVMs are designed for binary classification (e.g., personal vs junk emails).
- SVMs provide a geometric perspective on supervised learning.

Separating Hyperplanes

- In a p-dimensional space, a *hyperplane* is a flat affine subspace of dimension $p - 1$.
- In two dimensions, a hyperplane is a line.
- In three dimensions, a hyperplane is a plane.
- In $p > 3$ dimensions, it can be hard to visualize a hyperplane, but the notion of a $(p - 1)$-dimensional flat subspace still applies.
- The p-dimensional hyperplane is defined as

$$b + w_1 x_1 + w_2 x_2 + \cdots + w_p x_p = 0$$

where b and $\mathbf{w} \in \mathbb{R}^p$ are the parameters and based on the definition, the point $\mathbf{x} = (x_1, x_2, \cdots, x_p)^T$ lies on the hyperplane.

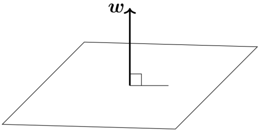
Separating Hyperplanes

- We can define the hyperplane by defining the function $f\colon \mathbb{R}^p \to \mathbb{R}$

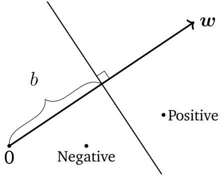
$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = 0$$

- $\mathbf{w}$ is a normal vector to the hyperplane:

$$\begin{aligned} f(\mathbf{x}_a) - f(\mathbf{x}_b) &= \langle \mathbf{w}, \mathbf{x}_a \rangle + b - (\langle \mathbf{w}, \mathbf{x}_b \rangle + b) \\ &= \langle \mathbf{w}, \mathbf{x}_a - \mathbf{x}_b \rangle = 0 \end{aligned}$$



(a) Separating hyperplane in 3D



(b) Projection of the setting in (a) onto a plane

Separating Hyperplanes

- Construct a hyperplane to perfectly separate training observations by class labels.
- For binary classification: $y_i = 1$ (positive) and $y_i = -1$ (negative).
- Geometrically: positives lie "above" and negatives "below" the hyperplane.
- The separating hyperplane satisfies:

$$
\begin{aligned}
f(\mathbf{x}_i) &= \langle \mathbf{w}, \mathbf{x}_i \rangle + b \geq 0 \quad \text{if} \quad y_i = +1, \\
f(\mathbf{x}_i) &= \langle \mathbf{w}, \mathbf{x}_i \rangle + b < 0 \quad \text{if} \quad y_i = -1,
\end{aligned}
$$

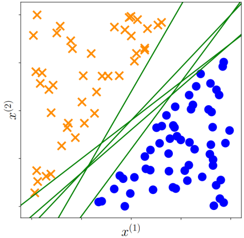
or equivalently:

$$
y_i \left(\langle \mathbf{w}, \mathbf{x}_i \rangle + b \right) \geq 0,
$$

for all $i = 1, \ldots, n$.

Separating Hyperplanes

- Classify test observation $\mathbf{x}^*$ based on the sign of $f(\mathbf{x}^*)$.
- If $|f(\mathbf{x}^*)|$ is large, $\mathbf{x}^*$ is far from the hyperplane, indicating confident classification.
- If $|f(\mathbf{x}^*)|$ is small, $\mathbf{x}^*$ is near the hyperplane, leading to less certainty in classification.
- Multiple linear classifiers (green lines) can separate orange crosses from blue discs. How to choose?

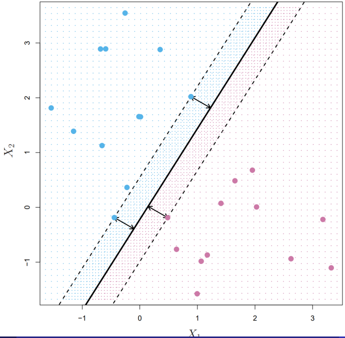


The Maximal Margin Classifier

- The maximal margin hyperplane maximizes the distance (margin) from training observations.
- The **margin** is the minimal distance from observations to the hyperplane.
- But, the maximal margin classifier may overfit when p (features) is large.

The Maximal Margin Classifier

- The maximal margin hyperplane is the centerline of the widest "slab" separating two classes.
- **Support vectors** are the data points closest to the separating hyperplane.
- These points lie on the margin boundaries and determine the position of the hyperplane.
- Shifting a support vector would alter the hyperplane's position.



Construction of the Maximal Margin Classifier

- Having a set of n training observations $x_1, \cdots, x_n \in \mathbb{R}^p$ and associated class labels $y_1, \cdots, y_n \in \{-1, 1\}$.
- The maximal margin hyperplane solves the optimization problem

$$\max_{\mathbf{w},b,r} \quad r$$

$$\text{subject to} \quad \underbrace{y_i\left(\langle \mathbf{w}, \mathbf{x}_i \rangle + b\right) \geq r,}_{\text{data fitting}} \ \underbrace{||\mathbf{w}|| = 1}_{\text{normalization}}, r > 0, \ \text{for all } i = 1, \ldots, n,$$

attempting to maximize the margin r while ensuring that the data lies on the correct side of the hyperplane.

- If there is no solution with $r > 0$, no separating hyperplane exists, and so there is no maximal margin classifier.

Construction of the Maximal Margin Classifier

- Instead of choosing that the parameter vector is normalized, we could choose a scale for the data.
- We choose this **scale** such that the value of the predictor is 1 at the closest example $\mathbf{x}_a$, i.e. $\langle \mathbf{w}, \mathbf{x}_a \rangle + b = 1$.
- If $\mathbf{x}'_a$ is the orthogonal **projection** of $\mathbf{x}_a$ onto the hyperplane, we have $\langle \mathbf{w}, \mathbf{x}'_a \rangle + b = 0$.
- We can write $\mathbf{x}'_a$ based on $\mathbf{x}_a$ as

$$\mathbf{x}_a = \mathbf{x}'_a + r\frac{\mathbf{w}}{||\mathbf{w}||}.$$

- Thus, we have

$$\left\langle \mathbf{w}, \mathbf{x}_a - r\frac{\mathbf{w}}{||\mathbf{w}||} \right\rangle + b = 0 \rightarrow \langle \mathbf{w}, \mathbf{x}_a \rangle + b - r\frac{\langle \mathbf{w}, \mathbf{w} \rangle}{||\mathbf{w}||} = 0$$

Construction of the Maximal Margin Classifier

- The first term is 1 by our assumption of scale, i.e., $\langle \mathbf{w}, \mathbf{x}_a \rangle + b = 1$.
- Also $\langle \mathbf{w}, \mathbf{w} \rangle = ||\mathbf{w}||^2$. Thus, we get $r = \frac{1}{||\mathbf{w}||}$.
- The optimization problem can be written as

$$\begin{aligned} \max_{\mathbf{w}, b} \quad & \frac{1}{||\mathbf{w}||} \\ \text{subject to} \quad & y_i \left(\langle \mathbf{w}, \mathbf{x}_i \rangle + b \right) \geq 1, \text{ for all } i = 1, \ldots, n, \end{aligned}$$

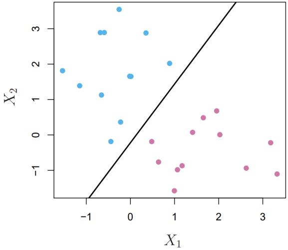
or equivalently

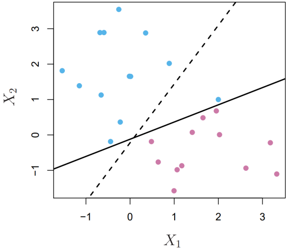
$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2} ||\mathbf{w}||^2 \\ \text{subject to} \quad & y_i \left(\langle \mathbf{w}, \mathbf{x}_i \rangle + b \right) \geq 1, \text{ for all } i = 1, \ldots, n, \end{aligned}$$

Known as **hard margin SVM** because the formulation does not allow for any violations of the margin condition.

Soft Margin Classifier (SVM)

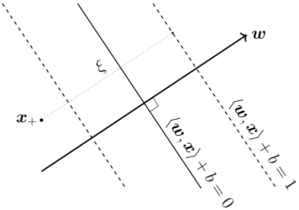
- A classifier based on a separating hyperplane will necessarily perfectly classify all of the training observations.
- This can lead to **sensitivity** to individual observations.
- For example, after the addition of a single observation, the Margin is not satisfactory because it has a very small margin.





Soft Margin Classifier (SVM)

- Let's misclassify a few training observations in order to have
  - a greater robustness to individual observations, and
  - a better classification of most of the training observations.
- The model that allows for some classification errors is called the **soft** margin classifier (SVM).
- The key geometric idea is to introduce a **slack variable** ξ_i for each sample $(\mathbf{x}_i, y_i)$.
- ξ measures the distance of a positive example $\mathbf{x}_+$ to the positive margin hyperplane $\langle \mathbf{w}, \mathbf{x} \rangle + b = 1$ when $\mathbf{x}_+$ is on the wrong side.



Soft Margin Classifier (SVM)

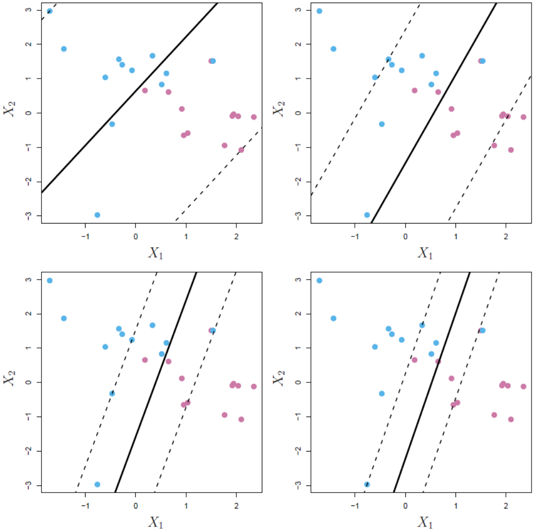
- Optimization problem for the soft margin classifier:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n} \xi_i,$$

$$\text{subject to} \quad \begin{cases} y_i\left(\langle \mathbf{w}, \mathbf{x}_i\rangle + b\right) \geq 1 - \xi_i,\ \xi_i \geq 0,\ \forall i. \\ \sum \xi_i < A \end{cases}$$

- Parameter $C > 0$ balances margin size and the tolerance for slack variables.
- Larger C reduces regularization, forcing the model to fit the training data more closely.
- Smaller C increases regularization, allowing more margin violations for better generalization.
- An alternative: impose an **upper bound** on the sum of slack variables instead of including it in the cost function.

Effect of changing upper bound of slack variables



Soft Margin Classifier (SVM)

- The soft margin classifier can be derived using a loss function perspective.

Soft Margin Classifier (SVM)

- The soft margin classifier can be derived using a loss function perspective.
- The **zero-one loss** counts mismatches between predictions and labels:

$$\mathbf{1}(f(\mathbf{x}_i) \neq y_i), \quad \text{where loss} = 0 \text{ if labels match, and 1 otherwise.}$$

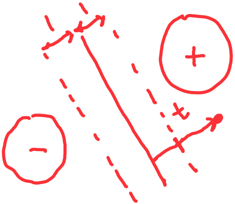
- Zero-one loss leads to a combinatorial optimization problem, which is difficult to solve.

Soft Margin Classifier (SVM)

- The soft margin classifier can be derived using a loss function perspective.
- The **zero-one loss** counts mismatches between predictions and labels:

$$\mathbf{1}(f(\mathbf{x}_i) \neq y_i), \quad \text{where loss} = 0 \text{ if labels match, and } 1 \text{ otherwise.}$$

- Zero-one loss leads to a combinatorial optimization problem, which is difficult to solve.
- The **hinge loss** provides a tractable alternative:



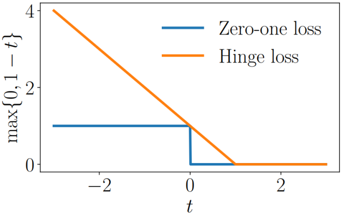
$$\ell(t) = \begin{cases} 0 & \text{if } \ t \geq 1, \\ 1 - t & \text{if } \ t < 1. \end{cases}$$

or equivalently:

$$\ell(t) = \max\{0, 1 - t\}, \quad t = yf(\mathbf{x}) = y(\langle \mathbf{w}, \mathbf{x} \rangle + b),$$

Soft Margin Classifier (SVM)

- We pay a penalty once we are closer than the margin to the hyperplane, even if the prediction is correct, and the penalty increases linearly.
- The hinge loss is a convex upper bound of zero-one loss.



- The loss corresponding to the hard margin SVM is defined as

$$\ell(t) = \begin{cases} 0 & \text{if } \; t \geq 1 \\ \infty & \text{if } \; t < 1 \end{cases},$$

where this loss can be interpreted as never allowing any examples inside the margin.

Soft Margin Classifier (SVM)

- Using the hinge loss gives us the unconstrained optimization problem

$$\min_{\mathbf{w},b} \quad \underbrace{\frac{1}{2}||\mathbf{w}||^2}_{\text{regularizer}} + \underbrace{C\sum_{i=1}^{n} \max\{0, 1 - y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)\}}_{\text{error (loss) term}}.$$

- Margin maximization can be interpreted as regularization.
- This optimization problem and previous one based on slack variables are equivalent.

Dual Support Vector Machine

- In the primal view, the number of parameters (dimension of $\mathbf{w}$) grows linearly with the number of features.
- The dual view reformulates the problem, making it independent of the number of features.
- Instead, the number of parameters in the dual view depends on the training set size.
- This approach is ideal for cases with more features than training examples.
- The dual SVM also enables the use of **kernel**s seamlessly.

Convex Duality via Lagrange Multipliers

- The primal variables: $\mathbf{w}$, b, and ξ.
- The corresponding Lagrangian to the optimization problem is

$$
\begin{aligned}
\mathcal{L}(\mathbf{w}, b, \xi, \alpha, \gamma) = & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i \\
& \underbrace{-\sum_{i=1}^{n}\alpha_i\left(y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) - 1 + \xi_i\right)}_{\text{correct classification constraint}} \\
& \underbrace{-\sum_{i=1}^{n}\gamma_i\xi_i}_{\text{non-negativity of the slack variables}}
\end{aligned}
$$

where $\alpha_i \geq 0$ and $\gamma_i \geq 0$ are the Lagrange multipliers.

Convex Duality via Lagrange Multipliers

- By differentiating the Lagrangian with respect to the three primal variables $\mathbf{w}$, b, and ξ respectively, we obtain

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \mathbf{w}^T - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i^T,$$

$$\frac{\partial \mathcal{L}}{\partial b} = -\sum_{i=1}^{n} \alpha_i y_i,$$

$$\frac{\partial \mathcal{L}}{\partial \xi_i} = C - \alpha_i - \gamma_i.$$

- We can find the maximum of the Lagrangian by setting each of these partial derivatives to zero.

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i,$$

which states that the optimal weight vector in the primal is a linear combination of the example.

Convex Duality via Lagrange Multipliers

- The previous expression for $\mathbf{w}$ also provides an explanation of the name "support vector machine."
- The examples $\mathbf{x}_i$, for which the corresponding parameters $\alpha_i = 0$, do not contribute to the solution $\mathbf{w}$ at all.
- The other examples, where $\alpha_i > 0$, are called support vectors since they "support" the hyperplane.
- The constraint obtained by setting $\frac{\partial \mathcal{L}}{\partial b}$ to zero implies that the optimal weight vector is an affine combination of the examples.
- By setting $\frac{\partial \mathcal{L}}{\partial \xi_i}$ we obtain that $C = \alpha_i + \gamma_i$.

Convex Duality via Lagrange Multipliers

- By substituting the expressions into the Lagrangian, we obtain the dual

$$
\begin{aligned}
\mathcal{D}(\xi, \alpha, \gamma) =& \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_{i=1}^{n} y_i \alpha_i \left\langle \sum_{j=1}^{n} y_j \alpha_j \mathbf{x}_j, \mathbf{x}_i \right\rangle \\
& + \underbrace{\sum_{i=1}^{n} (C - \alpha_i - \gamma_i)\, \xi_i}_{=0} - b \underbrace{\sum_{i=1}^{n} y_i \alpha_i}_{=0} + \sum_{i=1}^{n} \alpha_i \\
=& -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^{n} \alpha_i
\end{aligned}
$$

Convex Duality via Lagrange Multipliers

- Thus, the dual optimization problem of the SVM (dual SVM) is

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_{i=1}^{n} \alpha_i$$

$$\text{subject to} \quad \sum_{i=1}^{n} y_i \alpha_i = 0,\ 0 \leq \alpha_i \leq C,\ \text{for all } i = 1, \ldots, n.$$

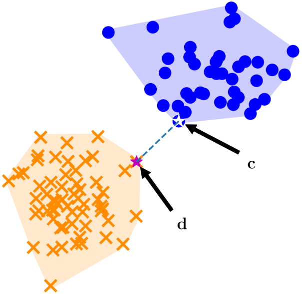
- Once we obtain the dual parameters $\boldsymbol{\alpha}^*$, we can recover the primal parameters $\mathbf{w}^*$ based on the combination of the examples expression.
- Also, if the example $\mathbf{x}_i$ lies on the margin's boundary, the parameter b^* can be obtained as

$$b^* = y_i - \langle \mathbf{w}^*, \mathbf{x}_i \rangle.$$

- If there is no examples that lie exactly on the margin, we should compute $|y_i - \langle \mathbf{w}^*, \mathbf{x}_i \rangle|$ for all support vectors and take the median value of this absolute value difference to be the value of b^*.

Convex Hull View

- **Convex hull** is a convex set that contains all the examples with the same label such that it is the smallest possible set.



- The convex hull can be described as the set

$$\mathrm{conv}(\mathbf{X}) = \left\{ \sum_{i=1}^{n} \alpha_i \mathbf{x}_i \right\} \text{ with } \sum_{i=1}^{n} \alpha_i = 1, \ \ \alpha_i \geq 0, \text{ for all } i = 1, \cdots, n.$$

Convex Hull View

- We form two convex hulls, corresponding to the positive and negative classes respectively.
- We pick a point $\mathbf{c}$, which is in the convex hull of the set of positive examples, and is closest to the negative class distribution.

$$\mathbf{c} = \sum_{i:y_i=+1} \alpha_i^+ \mathbf{x}_i.$$

- Similarly, we pick a point $\mathbf{d}$ in the convex hull of the set of negative examples, which is closest to the positive class distribution.

$$\mathbf{d} = \sum_{i:y_i=-1} \alpha_i^- \mathbf{x}_i.$$

- We define a difference vector between $\mathbf{c}$ and $\mathbf{d}$:

$$\mathbf{w} := \mathbf{c} - \mathbf{d}.$$

Convex Hull View

- Requiring **c** and **d** to be closest to each other is equivalent to minimizing the length/norm of **w**:

$$\arg\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 \rightarrow \arg\min_{\boldsymbol{\alpha}} \frac{1}{2} \left\| \sum_{i:y_i=+1} \alpha_i^+ \mathbf{x}_i - \sum_{i:y_i=-1} \alpha_i^- \mathbf{x}_i \right\|^2,$$

  where $\boldsymbol{\alpha}$ is the set of all coefficients $(\boldsymbol{\alpha}^+, \boldsymbol{\alpha}^-)$.
- Also, the constraints $\sum_{i:y_i=+1}^{n} \alpha_i^+ = 1$ and $\sum_{i:y_i=-1}^{n} \alpha_i^- = 1$ implies that

$$\sum_{i=1}^{n} y_i \alpha_i = 0$$

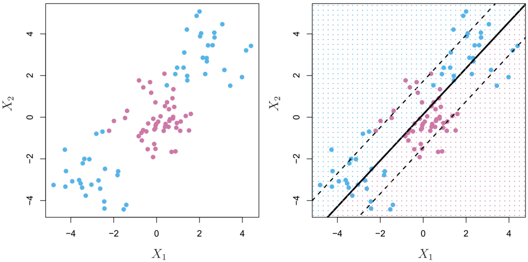
- The objective function and the above constraint, along with $\boldsymbol{\alpha} > 0$, give us a constrained (convex) optimization problem.
- This optimization problem can be shown to be the same as that of the dual **hard margin** SVM.

Convex Hull View

- To obtain the soft margin dual, we consider the reduced hull.
- The **reduced hull** is similar to the convex hull but has an upper bound to the size of the coefficients α.
- The bound on α shrinks the convex hull to a smaller volume.

Support Vector Machine with Kernels

- In practice we are sometimes faced with non-linear class boundaries.
- A support vector classifier or any linear classifier will perform poorly in not linearly separable datasets.



- We could address the problem of possibly non-linear boundaries between classes by enlarging the feature space using quadratic, cubic, and even higher-order polynomial functions of the predictors.

Support Vector Machine with Kernels

Definition (Kernel)

Kernels are by definition functions $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ for which there exists a Hilbert space $\mathcal{H}$ and $\phi : \mathcal{X} \to \mathcal{H}$ a feature map such that

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{H}}$$

- The inputs $\mathcal{X}$ of the kernel function can be very general and are not necessarily restricted to the input data dimension.
- Support vector classifiers use a linear kernel (inner product of the examples).
- The generalization from an inner product to a kernel function is known as the **kernel trick**.

Support Vector Machine with Kernels

- The matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$, resulting from the inner products or the application of $k(\cdot, \cdot)$ to a dataset, is called the **Gram matrix** (kernel matrix):

$$\mathbf{K}_{ij} := \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) := k(\mathbf{x}_i, \mathbf{x}_j)$$

- The kernel matrix $\mathbf{K}$ is symmetric and positive semi-definite for any examples:

$$\mathbf{z}^T \mathbf{K} \mathbf{z} \geq 0 \;\text{ for any } \mathbf{z} \in \mathbb{R}^n$$

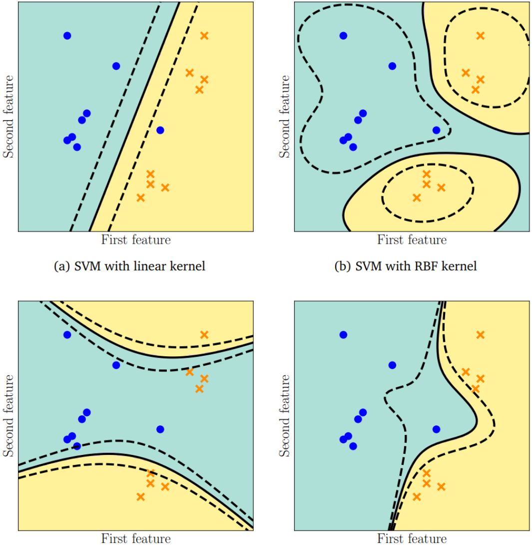
- Polynomial kernel of degree d

$$k(\mathbf{x}, \mathbf{x}') = \left(\mathbf{x}^T \mathbf{x}' + c\right)^d$$

- Radial (RBF) kernel

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-||\mathbf{x} - \mathbf{x}'||^2}{2\sigma^2}\right) = \exp\left(-\gamma ||\mathbf{x} - \mathbf{x}'||^2\right)$$

Support Vector Machine with Kernels



(a) SVM with linear kernel

(b) SVM with RBF kernel

(c) SVM with polynomial (degree 2) kernel

(d) SVM with polynomial (degree 3) kernel

Suppose that we would like to perform classification using SVMs, and there are $K > 2$ classes.

1 One-Versus-One Classification approach
   - This approach construct $\binom{K}{2}$ SVMs, each of which compares a pair of classes.
   - We classify a test observation using each of the $\binom{K}{2}$ classifiers, and we tally the number of times that the test observation is assigned to each of the K classes.
   - The final classification is performed by assigning the test observation to the class to which it was most frequently assigned in these $\binom{K}{2}$ pairwise classifications.

SVMs with More than Two Classes

Suppose that we would like to perform classification using SVMs, and there are $K > 2$ classes.

2 One-Versus-All Classification approach

- We fit K SVMs, each time comparing one of the K classes to the remaining $K - 1$ classes.
- Let $\mathbf{w}_k$ and b_k denote the parameters that result from fitting an SVM comparing the kth class (coded as $+1$) to the others (coded as -1).
- We assign the observation to the class for which $\mathbf{w}_k^T \mathbf{x}^* + b$ is the largest, as this amounts to a high level of confidence that the test observation belongs to the kth class rather than to any of the other classes.