# DATA 572: Supervised Learning

2023W2

Shan Du

# Support Vector Machines

- We first discuss a general mechanism for converting a linear classifier into one that produces non-linear decision boundaries.
- We then introduce the support vector machine, which does this in an automatic way.

# Classification with Non-Linear Decision Boundaries

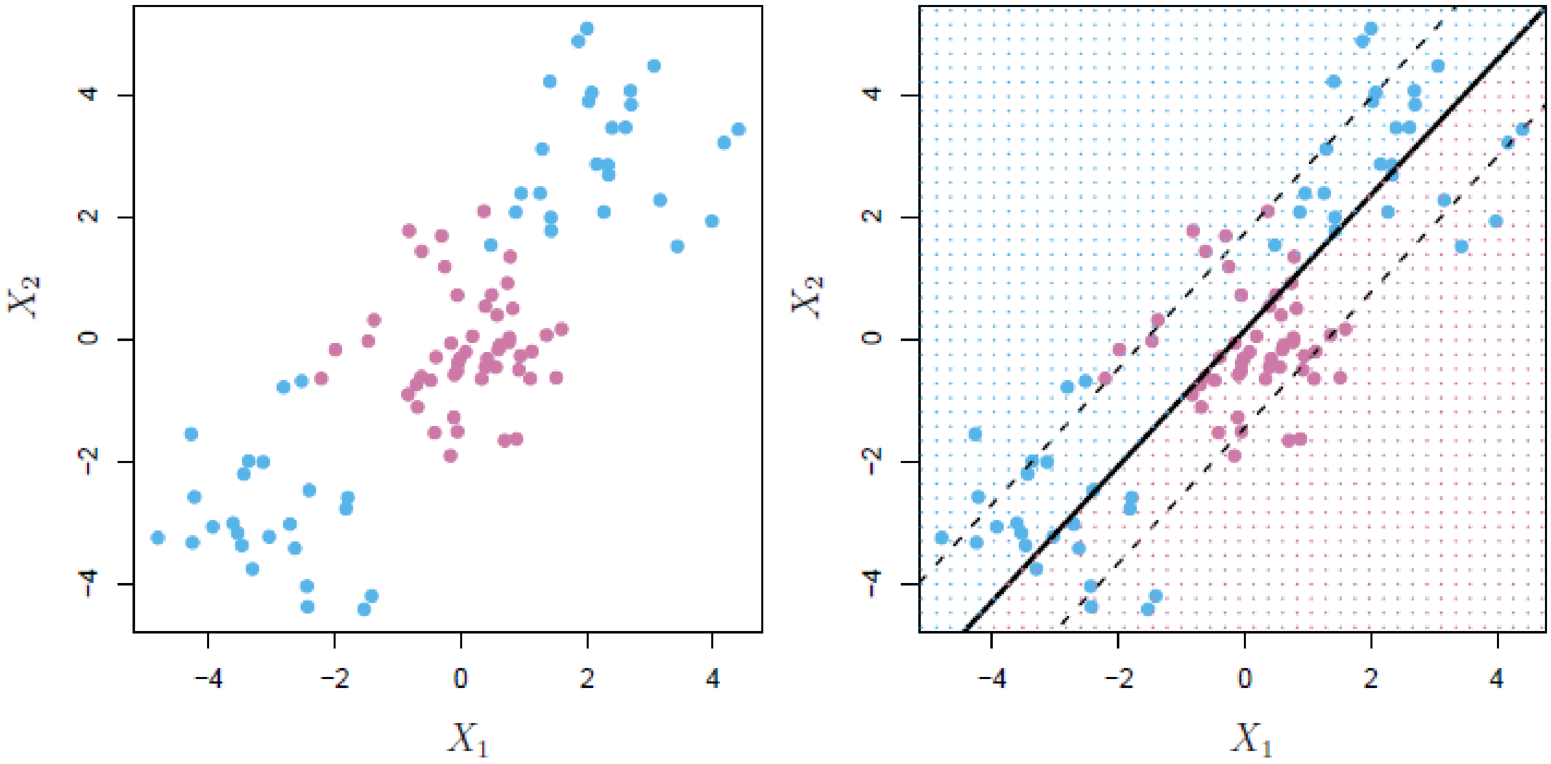


**FIGURE 9.8.** Left: *The observations fall into two classes, with a non-linear boundary between them.* Right: *The support vector classifier seeks a linear boundary, and consequently performs very poorly.*

# Classification with Non-Linear Decision Boundaries

- When there is a nonlinear relationship between the predictors and the outcome, in linear regression, we consider enlarging the feature space using functions of the predictors, such as quadratic and cubic terms, in order to address the non-linearity.
- In the case of the support vector classifier, we could address the problem of possibly non-linear boundaries between classes in a similar way, by enlarging the feature space using quadratic, cubic, and even higher-order polynomial functions of the predictors.

# Classification with Non-Linear Decision Boundaries

- Rather than fitting a support vector classifier using $p$ features $X_1, X_2, \dots, X_p$, we could instead fit a support vector classifier using $2p$ features $X_1, {X_1}^2, X_2, {X_2}^2, \dots, X_p, {X_p}^2$.
- Then the optimization problem would be

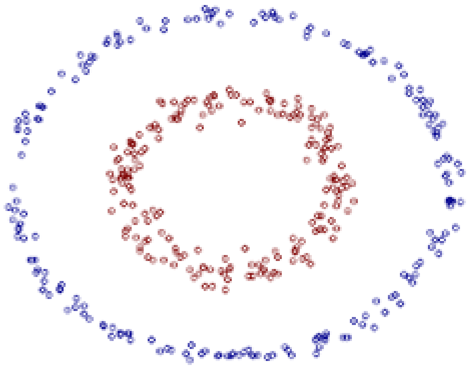
$$\underset{\beta_0, \beta_{11}, \beta_{12}, \dots, \beta_{p1}, \beta_{p2}, \epsilon_1, \dots, \epsilon_n, M}{maximize} M$$

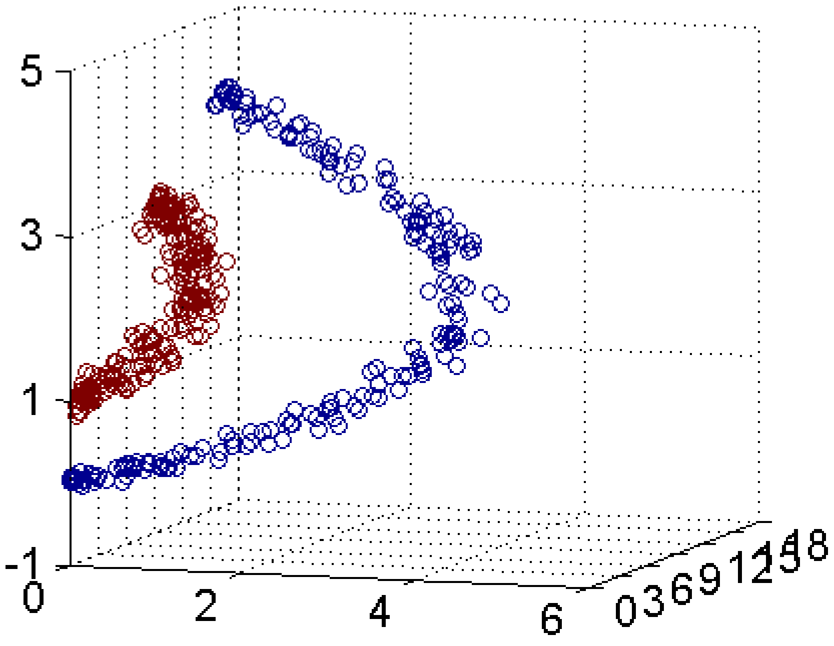
subject to $y_i\left(\beta_0 + \sum_{j=1}^{p} \beta_{j1} x_{ij} + \sum_{j=1}^{p} \beta_{j2} {x_{ij}}^2\right) \geq M(1-\epsilon_i), \sum_{i=1}^{n} \epsilon_i \leq C, \epsilon_i \geq 0, \sum_{j=1}^{p} \sum_{k=1}^{2} \beta_{jk}^2 = 1$

# Classification with Non-Linear Decision Boundaries

- Why does this lead to a non-linear decision boundary? In the enlarged feature space, the decision boundary is in fact linear. But in the original feature space, the decision boundary is of the form $q(x) = 0$, where $q$ is a quadratic polynomial, and its solutions are generally non-linear.

# Classification with Non-Linear Decision Boundaries





# The Support Vector Machine

- The support vector machine (SVM) is an extension of the support vector classifier that results from enlarging the feature space in a specific way, using *kernels*.
- We may want to enlarge our feature space in order to accommodate a non-linear boundary between the classes. The kernel approach is simply an efficient computational approach for enacting this idea.

# The Support Vector Machine

- The solution to the support vector classifier problem involves only the *inner products* of the observations (as opposed to the observations themselves).
- The inner product of two $r$-vectors $a$ and $b$ is defined as $\langle a, b\rangle = \sum_{i=1}^{r} a_i b_i$.
- Thus, the inner product of two observations $x_i$, $x_{i'}$ is given by $\langle x_i, x_{i'}\rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$.

# The Support Vector Machine

- The linear support vector classifier can be represented as

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle$$

where there are $n$ parameters $\alpha_i, i = 1, \ldots, n$, one per training observation.

- To estimate the parameters $\alpha_1, \ldots, \alpha_n$ and $\beta_0$, all we need are the inner products between all pairs of training observations.

# The Support Vector Machine

- For a new point $x$, to evaluate the function $f(x)$, we need to compute the inner product between the new point and each of the training points $x_i$.
- However, it turns out that $\alpha_i$ is nonzero only for the support vectors in the solution—that is, if a training observation is not a support vector, then its $\alpha_i$ equals zero.

# The Support Vector Machine

- So if $S$ is the collection of indices of these support points, we can rewrite

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i \langle x, x_i \rangle$$

- To summarize, in representing the linear classifier $f(x)$, and in computing its coefficients, all we need are inner products.

# The Support Vector Machine

- We can use a *generalization* of the inner product of the form

$$K\langle x_i, x_{i'}\rangle$$

where $K$ is some function that we will refer to as a *kernel*. A kernel is a function that quantifies the similarity of two observations. For instance, we could simply take

$$K\langle x_i, x_{i'}\rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

which would just give us back the support vector classifier.

# The Support Vector Machine

- This is a linear kernel because the support vector classifier is linear in the features; the linear kernel essentially quantifies the similarity of a pair of observations using Pearson (standard) correlation.
- But one could instead choose another form. For instance, one could replace every instance of $\sum_{j=1}^{p} x_{ij}x_{i'j}$ with the quantity

$$K\langle x_i, x_{i'}\rangle = (1 + \sum_{j=1}^{p} x_{ij}x_{i'j})^d$$

This is a *polynomial kernel of degree $d$,* where $d$ is a positive integer.

# The Support Vector Machine

- Using such a kernel with $d > 1$, instead of the standard linear kernel, in the support vector classifier algorithm leads to a much more flexible decision boundary.
- It essentially amounts to fitting a support vector classifier in a higher-dimensional space involving polynomials of degree $d$, rather than in the original feature space.

# The Support Vector Machine

- When the support vector classifier is combined with a non-linear kernel, the resulting classifier is known as a *support vector machine*.
- Note that in this case the (non-linear) function has the form

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i K\langle x, x_i \rangle$$

# The Support Vector Machine

- Another popular choice is the *radial kernel*, which takes the form

$$K\langle x_i, x_{i'} \rangle = exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i' j})^2)$$

where $\gamma$ is a positive constant.

# The Support Vector Machine

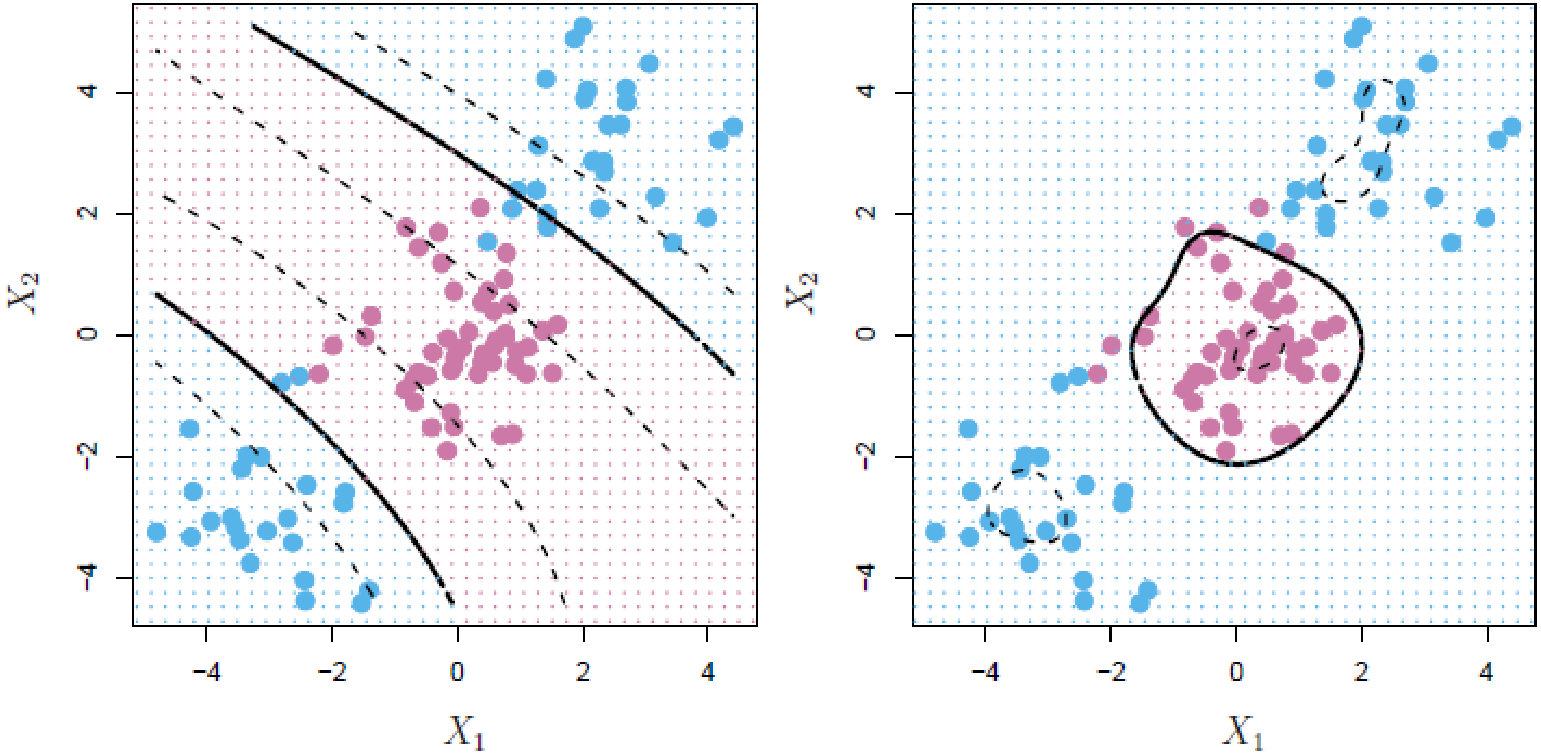


**FIGURE 9.9.** Left: *An SVM with a polynomial kernel of degree 3 is applied to the non-linear data from Figure 9.8, resulting in a far more appropriate decision rule.* Right: *An SVM with a radial kernel is applied. In this example, either kernel is capable of capturing the decision boundary.*

# The Support Vector Machine

- How does the radial kernel actually work?

If a given test observation $x^* = (x_1^*, \ldots, x_p^*)^T$ is far from a training observation $x_i$ in terms of Euclidean distance, then $\sum_{j=1}^{p}(x_j^* - x_{ij})^2)$ will be large, and so
$K\langle x^*, x_i\rangle = exp(-\gamma \sum_{j=1}^{p}(x_j^* - x_{ij})^2)$ will be tiny.

This means that $x_i$ will play virtually no role in $f(x^*)$.

# The Support Vector Machine

- Recall that the predicted class label for the test observation $x^*$ is based on the sign of $f(x^*)$. In other words, training observations that are far from $x^*$ will play essentially no role in the predicted class label for $x^*$.
- This means that the radial kernel has very local behavior, in the sense that only nearby training observations have an effect on the class label of a test observation.

# The Support Vector Machine

- What is the advantage of using a kernel rather than simply enlarging the feature space using functions of the original features?

One advantage is computational, and it amounts to the fact that using kernels, one need only compute $K\langle x_i, x_{i'}\rangle$ for all distinct pairs $x_i, x_{i'}$. This can be done without explicitly working in the enlarged feature space. This is important because in many applications of SVMs, the enlarged feature space is so large that computations are intractable. For some kernels, such as the radial kernel, the feature space is implicit and infinite-dimensional, so we could never do the computations there anyway!

# SVMs with More than Two Classes

- The two most popular proposals for extending SVMs to the *K*-class are the *one-versus-one* and *one-versus-all* approaches.

# One-Versus-One Classification

- Suppose that we would like to perform classification using SVMs, and there are $K > 2$ classes. A *one-versus-one* or *all-pairs* approach constructs $\binom{K}{2}$ SVMs, each of which compares a pair of classes.
- For example, one such SVM might compare the $k$th class, coded as $+1$, to the $k'$th class, coded as $-1$.
- 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.

# One-Versus-All Classification

- The *one-versus-all* approach (also referred to as *one-versus-rest*) is an alternative procedure for applying SVMs in the case of $K > 2$ classes. We fit $K$ SVMs, each time comparing one of the $K$ classes to the remaining $K - 1$ classes.
- Let $\beta_{0k}, \beta_{1k}, \ldots, \beta_{pk}$ denote the parameters that result from fitting an SVM comparing the $k$th class (coded as +1) to the others (coded as −1). Let $x^*$ denote a test observation. We assign the observation to the class for which $\beta_{0k} + \beta_{1k}x_1^* + \beta_{2k}x_2^* + \cdots + \beta_{pk}x_p^*$ is largest, as this amounts to a high level of confidence that the test observation belongs to the $k$th class rather than to any of the other classes.