Support Vector Machines (SVMs)

SVM (Support Vector Machine) is a powerful supervised machine learning algorithm used for ***classification and regression*** tasks. It works by finding the ***optimal hyperplane*** that ***maximizes*** the margin between different classes in a dataset. Key concepts include:

- **Support Vectors**: Data points that define the decision boundary.
- **Margin**: The distance between the hyperplane and the nearest data points of each class.
- **Soft Margin vs. Hard Margin**: Soft margin allows some misclassification for better generalization, while hard margin strictly separates data.
- **Kernel Trick**: Allows SVM to work in higher-dimensional spaces by transforming data using functions like linear, polynomial, and RBF (Radial Basis Function).

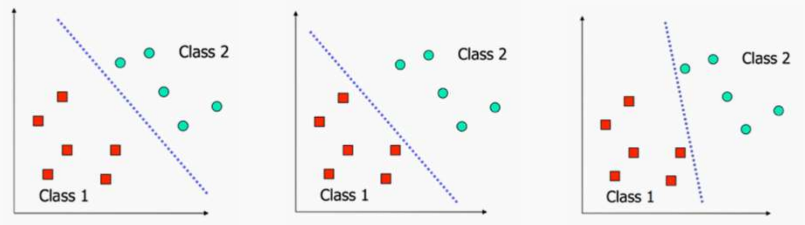
SVMs are widely used in applications like image classification, text categorization, and bioinformatics due to their robustness in high-dimensional spaces.

Decision Boundaries Revisited

In logistic regression, we learn a ***decision boundary*** that separates the training classes in the feature space.

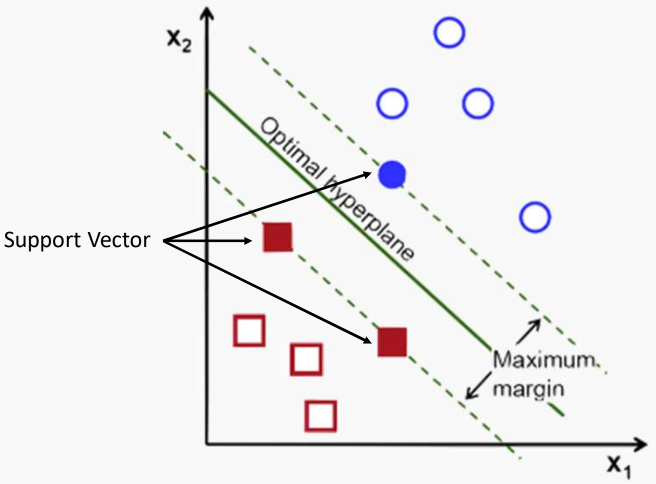
When the data can be perfectly separated by a linear boundary, we call the data ***linearly separable***.

In this case, multiple decision boundaries can fit the data. How do we choose the best?



Question: What happens to our logistic regression model when training on linearly separable datasets?

Illustration of an SVM



Decision Boundaries Revisited (cont.)

Constraints on the decision boundary:

- We may prefer a decision boundary that does not ‘**favor**’ any class (esp. when the classes are roughly equally populous).

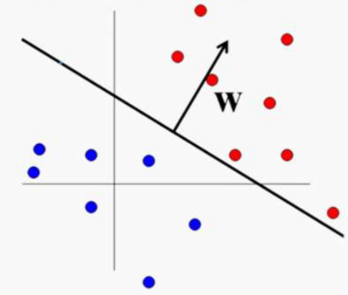
- Geometrically, this means choosing a boundary that maximizes the distance or ***margin*** between the boundary and both classes.

Geometry of Decision Boundaries

Recall that the decision boundary is defined by some equation in terms of the predictors. A linear boundary is defined by:

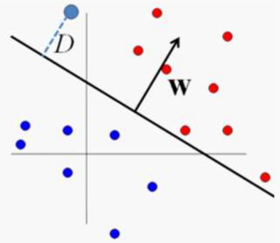
$w^\top x + b = 0$ (General equation of a hyperplane)

Recall that the non-constant coefficients, w, represent a ***normal vector***, pointing orthogonally ***away from the plane***



Geometry of Decision Boundaries (cont.)

Now, using some geometry, we can compute the **distance** between any point to the decision boundary using *w* and *b*.



The signed distance from a point $x \in \mathbb{R}^n$ to the decision boundary is

$$D(x) = \frac{w^\top x + b}{\|w\|} \quad \text{(Euclidean Distance Formula)}$$

Maximizing Margins

Now we can formulate our goal - find a decision boundary that maximizes the distance to both classes - as an optimization problem:

$$\begin{cases} \max\limits_{w,b} M \\ \text{such that } |D(x_n)| = \frac{y_i(w^\top x_n + b)}{\|w\|} \geq M, \; n = 1, \ldots, N \end{cases}$$

where M is a real number representing the width of the 'margin' and y_i = ±1. The inequalities $|D(x_n)| \geq M$ are called ***constraints***.

The constrained optimization problem as present here looks tricky. Let's simplify it with a little geometric intuition.

Maximizing Margins (cont.)

Notice that maximizing the distance of ***all points*** to the decision boundary, is exactly the same as maximizing the distance to the ***closest points***.

The points closest to the decision boundary are called ***support vectors***.

For any plane, we can ***always scale*** the equation:

$$w^{T}x + b = 0$$

so that the ***support vectors*** lie on the planes:

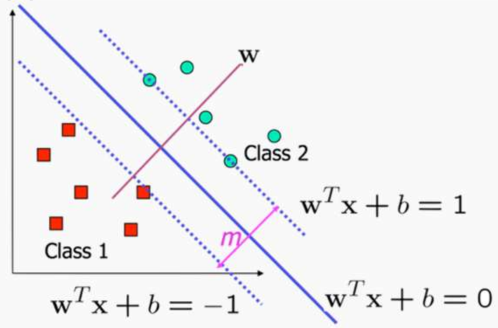
$$w^{T}x + b = \pm 1,$$

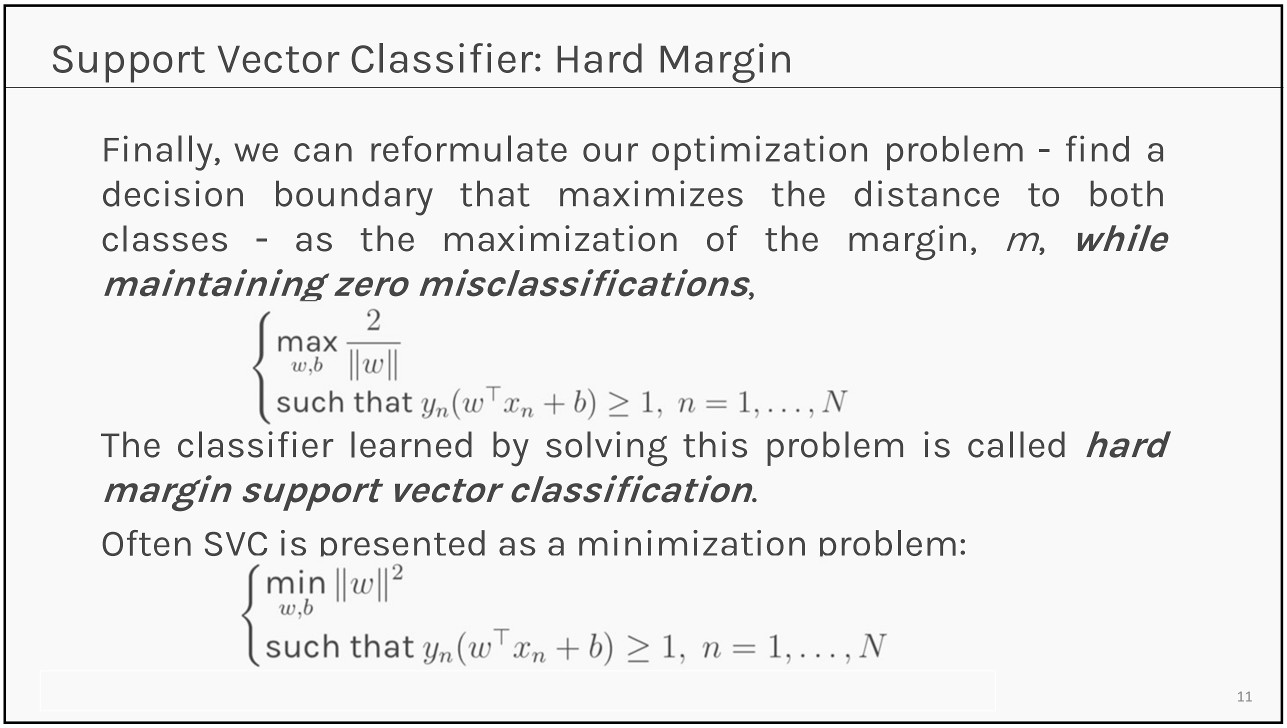
depending on their classes.

Maximizing Margins Illustration

For points on planes $w^{T}x + b = \pm 1$, their distance to the decision boundary is ±1/‖w‖.

So we can define the ***margin*** of a decision boundary as the distance to its support vectors, $m = 2/\| w\|$.


$\mathbf{w}$
Class 2
$\mathbf{w}^T\mathbf{x} + b = 1$
m
Class 1
$\mathbf{w}^T\mathbf{x} + b = -1$
$\mathbf{w}^T\mathbf{x} + b = 0$



Support Vector Classifier: Hard Margin

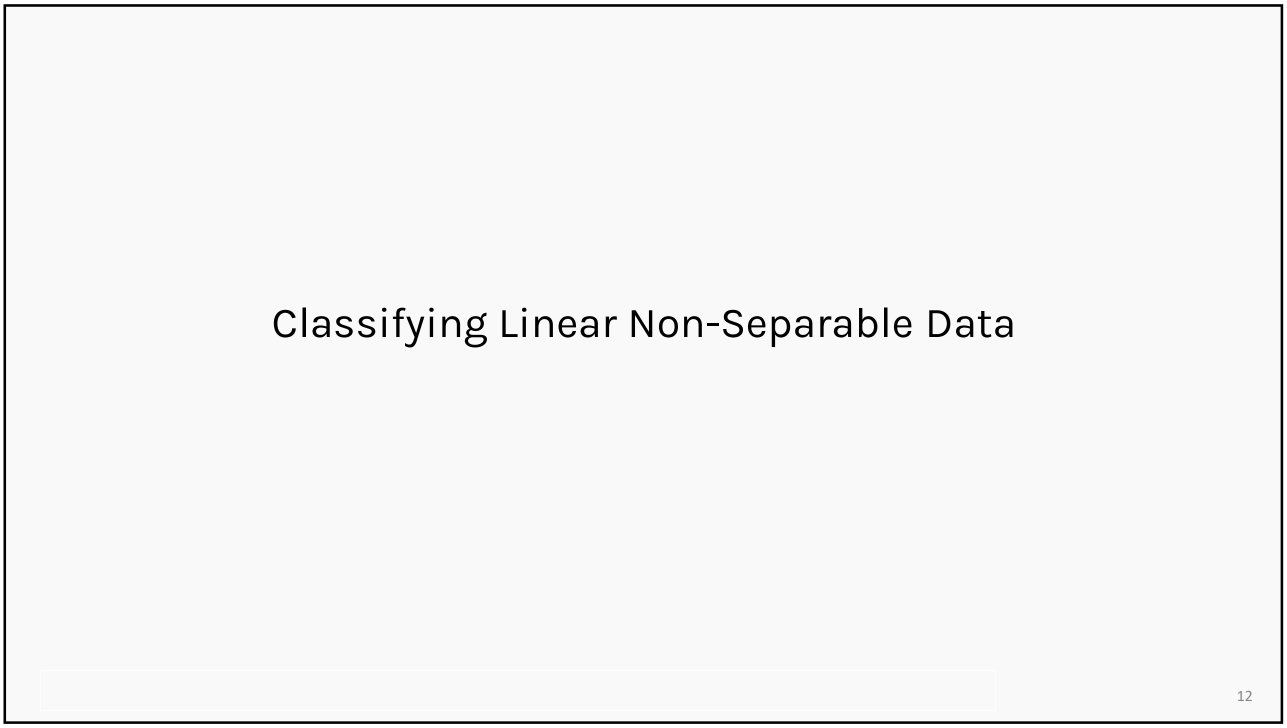
Finally, we can reformulate our optimization problem - find a decision boundary that maximizes the distance to both classes - as the maximization of the margin, *m*, ***while maintaining zero misclassifications***,

$$\begin{cases} \max\limits_{w,b} \dfrac{2}{\|w\|} \\ \text{such that } y_n(w^\top x_n + b) \geq 1,\ n = 1, \dots, N \end{cases}$$

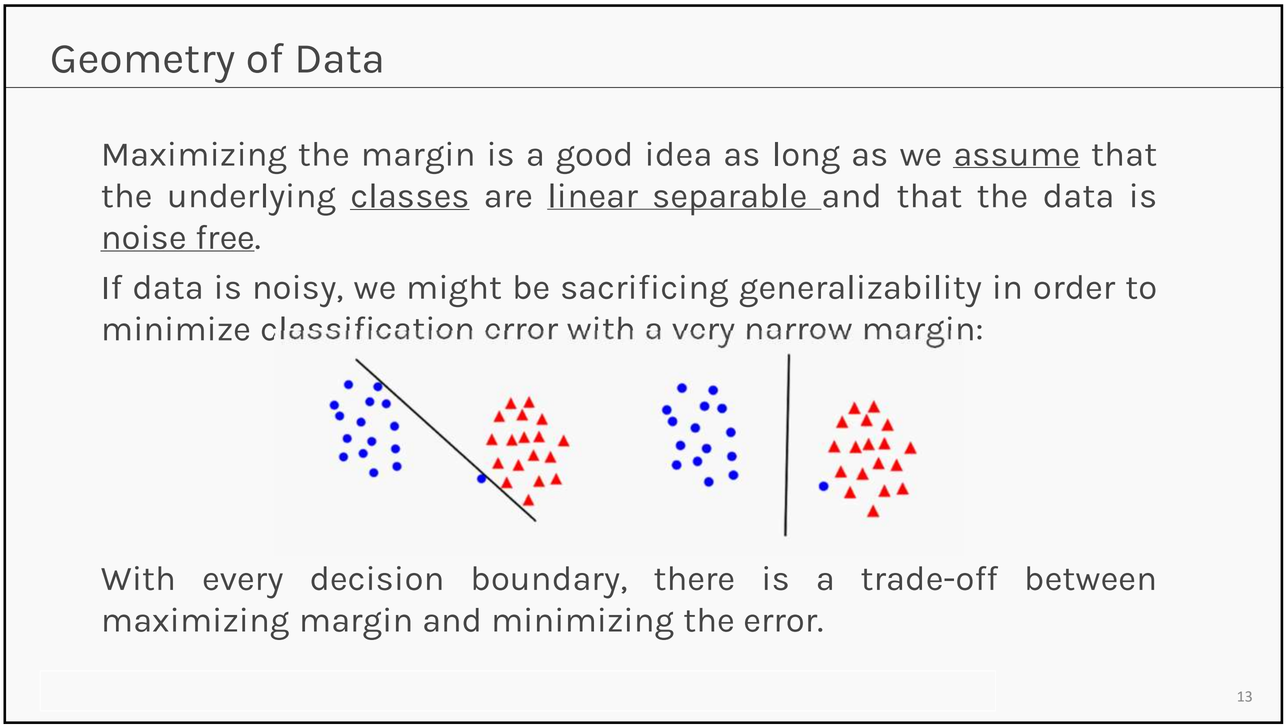
The classifier learned by solving this problem is called ***hard margin support vector classification***.

Often SVC is presented as a minimization problem:

$$\begin{cases} \min\limits_{w,b} \|w\|^2 \\ \text{such that } y_n(w^\top x_n + b) \geq 1,\ n = 1, \dots, N \end{cases}$$



Classifying Linear Non-Separable Data



Geometry of Data

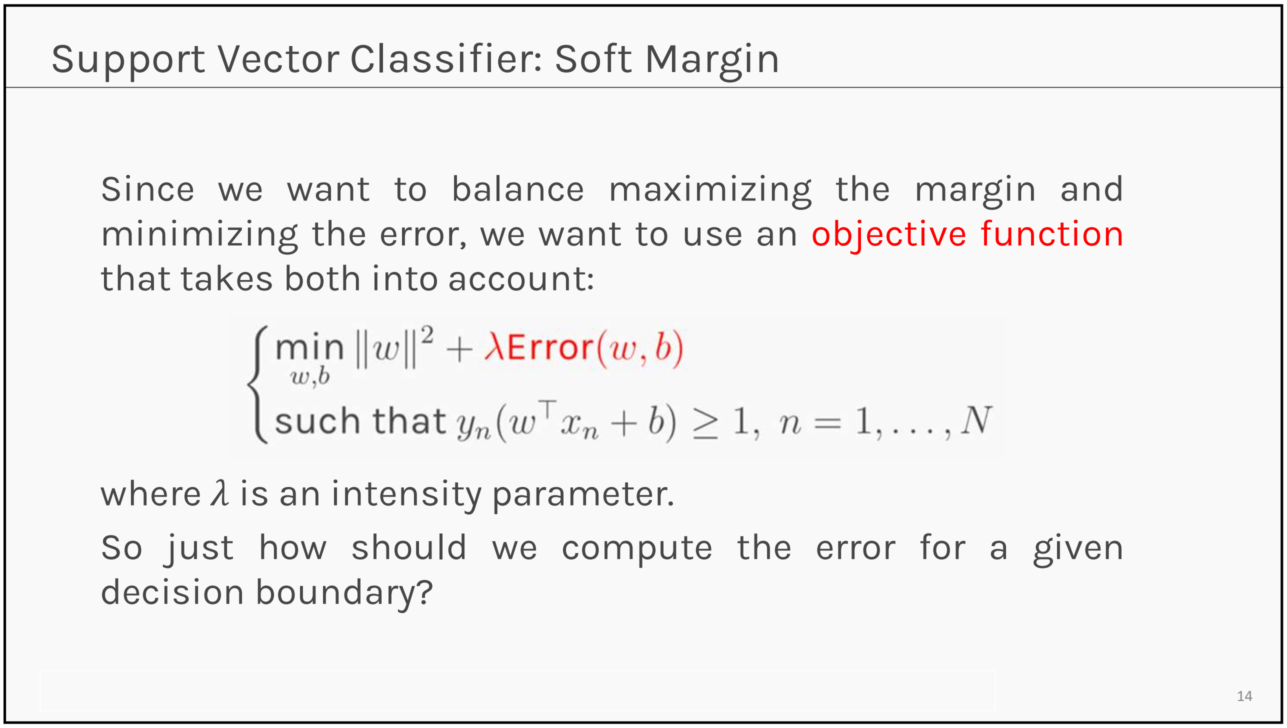
Maximizing the margin is a good idea as long as we assume that the underlying classes are linear separable and that the data is noise free.

If data is noisy, we might be sacrificing generalizability in order to minimize classification error with a very narrow margin:

With every decision boundary, there is a trade-off between maximizing margin and minimizing the error.

13

13



Support Vector Classifier: Soft Margin

Since we want to balance maximizing the margin and minimizing the error, we want to use an objective function that takes both into account:

$$\begin{cases} \min\limits_{w,b} \|w\|^2 + \lambda \text{Error}(w,b) \\ \text{such that } y_n(w^\top x_n + b) \geq 1, \; n = 1, \ldots, N \end{cases}$$

where λ is an intensity parameter.

So just how should we compute the error for a given decision boundary?

14

14

Support Vector Classifier: Soft Margin (cont.)

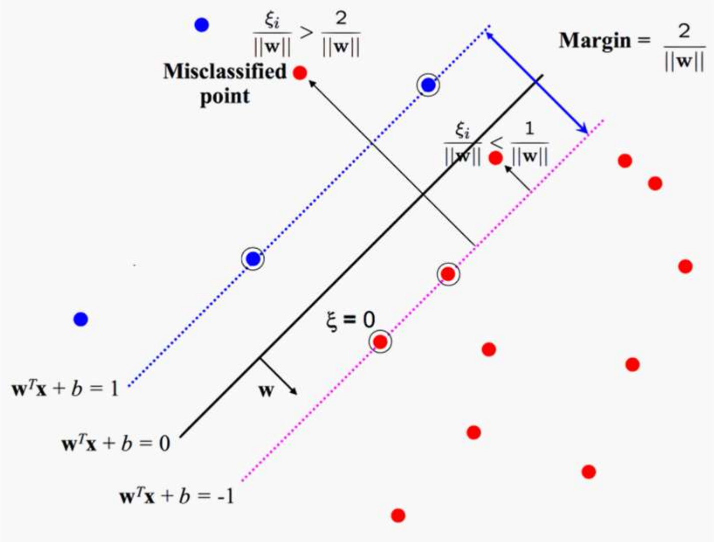
We want to express the error as a function of distance to the decision boundary.

Recall that the support vectors have distance $1/\|w\|$ to the decision boundary. We want to penalize two types of ‘errors’

- **(margin violation)** points that are on the <u>correct side of the boundary but are inside the margin</u>. They have distance $\xi\ /\|w\|$, where $0 < \xi < 1$.
- **(misclassification)** points that are on the <u>wrong side of the boundary</u>. They have distance $\xi/\|\mathrm{w}\|$, where $\xi > 1$.

Specifying a nonnegative quantity for ξ_n is equivalent to quantifying the error on the point x_n.

Support Vector Classifier: Soft Margin Illustration


$\frac{\xi_i}{\|\mathbf{w}\|} > \frac{2}{\|\mathbf{w}\|}$
Misclassified point
Margin = $\frac{2}{\|\mathbf{w}\|}$
$\frac{\xi_i}{\|\mathbf{w}\|} < \frac{1}{\|\mathbf{w}\|}$
ξ = 0
$\mathbf{w}^T\mathbf{x} + b = 1$
w
$\mathbf{w}^T\mathbf{x} + b = 0$
$\mathbf{w}^T\mathbf{x} + b = -1$

Support Vector Classifier: Soft Margin (cont.)

Formally, we incorporate error terms ξ_n 's into our optimization problem by:

$$\begin{cases} \min\limits_{\xi_n \in \mathbb{R}^+, w, b} \|w\|^2 + \lambda \sum_{n=1}^{N} \xi_n \\ \text{such that } y_n(w^\top x_n + b) \geq 1 - \xi_n,\ n = 1, \ldots, N \end{cases}$$

The solution to this problem is called ***soft margin support vector classification*** or simply ***support vector classification***.

Tuning SVC

Choosing different values for λ in

$$\begin{cases} \min\limits_{\xi_n \in \mathbb{R}^+, w, b} \|w\|^2 + \lambda \sum_{n=1}^{N} \xi_n \\ \text{such that } y_n(w^\top x_n + b) \geq 1 - \xi_n,\ n = 1, \ldots, N \end{cases}$$

will give us different classifiers. In general,

- small λ penalizes errors less and hence the classifier will have a large margin
- large λ penalizes errors more and hence the classifier will accept narrow margins to improve classification
- setting $\lambda = \infty$ produces the hard margin solution

Recall how the error terms ξ_n's were defined: the points where ξ_n = 0 are precisely the support vectors

Thank you