

Support Vector Machines(SVMs)

Naeemullah Khan

naeemullah.khan@kaust.edu.sa



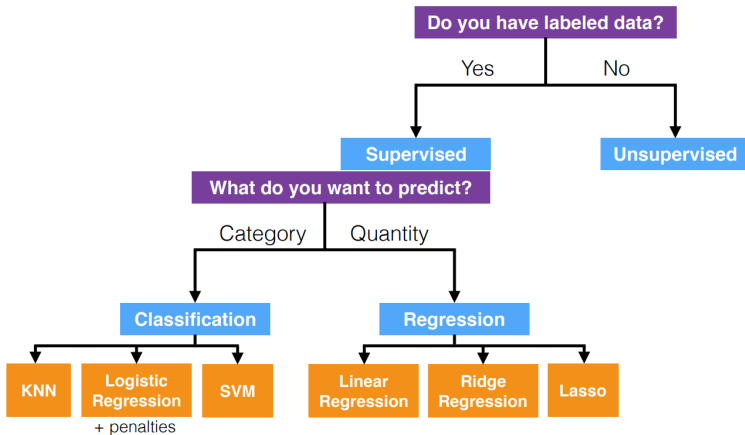
جامعة الملك عبد الله
للعلوم والتقنية
King Abdullah University of
Science and Technology

KAUST Academy
King Abdullah University of Science and Technology

July 25, 2025



- ▶ Hyperplanes
- ▶ Maximal margin classifier
- ▶ Support vector classifier
- ▶ Support vector machine

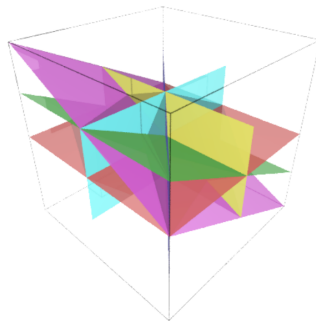


Support vector machine (SVM) is a supervised method for binary classification (two class). It is a generalization of 1 and 2 below.

1. **Maximal margin classifier:** only applicable to linearly separable data.
2. **Support vector classifier:** can be applied to data that is not linearly separable. Decision boundary still linear.
3. **Support vector machine:** non-linear decision boundary.

What is a hyperplane?

- ▶ In p -dimensional space, a hyperplane is a $(p - 1)$ -dimensional affine subspace.
- ▶ In 2D, a hyperplane is a flat 1D subspace, aka a line.
- ▶ In 3D, a hyperplane is a flat 2D subspace, aka a plane.



- ▶ A 2D hyperplane is defined by the equation:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$

- ▶ “Define” means any $\mathbf{X} = (X_1, X_2)$ for which the above equation holds is a point on the hyperplane.
- ▶ The above equation describes a line, which is a hyperplane in 2D.

- In p dimensions, a hyperplane is defined by the equation:

$$\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p = 0$$

- Similarly, any $\mathbf{X} = (X_1, X_2, \dots, X_p)$ for which the above equation holds is a point on the hyperplane.

- ▶ Instead of a point on the hyperplane, consider \mathbf{X} for which

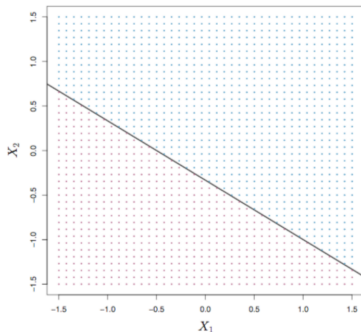
$$\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p > 0$$

- ▶ This point lies on one side of the hyperplane. An \mathbf{X} for which

$$\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p < 0$$

lies on the other side of the hyperplane.

- ▶ We can think of the hyperplane as dividing the p -dimensional space into two halves.



ISL (8th printing, 2017)

- ▶ This hyperplane in 2 dimensions is the line

$$1 + 2X_1 + 3X_2 = 0$$

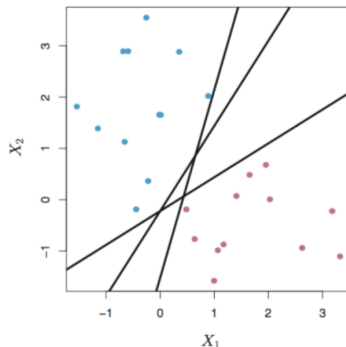
- ▶ The blue region is the set of points for which

$$1 + 2X_1 + 3X_2 > 0$$

- ▶ The purple region is the set of points for which

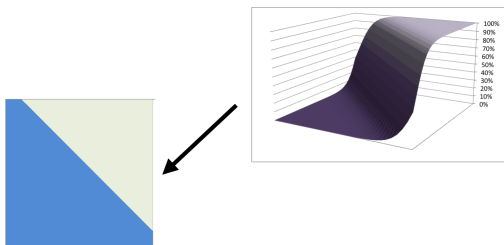
$$1 + 2X_1 + 3X_2 < 0$$

- **Idea:** Use a separating hyperplane for binary classification.
- **Key assumption:** Classes can be separated by a linear decision boundary.



ISL (8th printing, 2017)

- ▶ **Aside:** Logistic regression effectively finds a separating hyperplane.

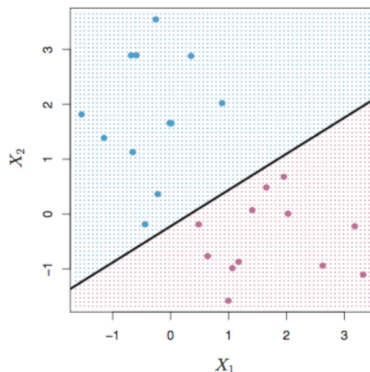


- ▶ Maximal margin classifiers and SVMs do this differently.

- ▶ **To classify new data points:**
- ▶ Assign class by location of new data point with respect to the hyperplane.

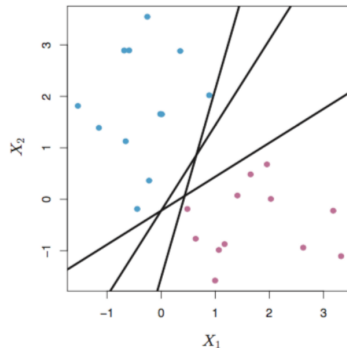
$$\hat{y} = \text{sign}(\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p)$$

- ▶ The farther away a point is from the separating hyperplane, the more confident we are about its class assignment.



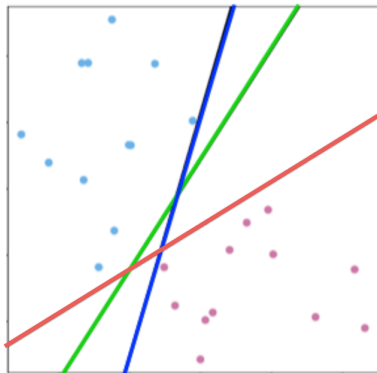
Separating Hyperplane Classifier

- Notice that for a linearly separable dataset, there are many possible separating hyperplanes that divide the dataset into two classes (in fact, an infinite number).

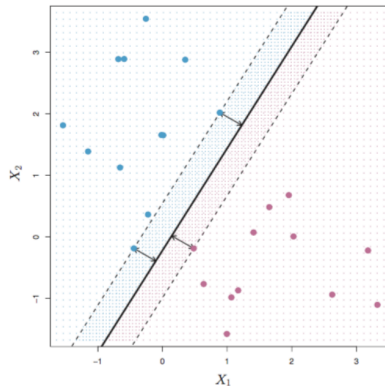


Which decision boundary?

- ▶ Multiple decision boundaries can perfectly separate the data.
- ▶ Which one should we choose?
- ▶ Some boundaries may generalize better than others.

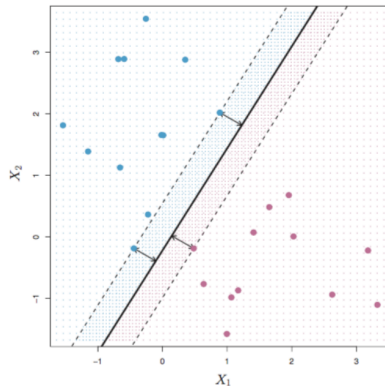


- ▶ Which of the infinite separating hyperplanes should we choose?
- ▶ A natural choice is the **maximal margin hyperplane**.
- ▶ It is the separating hyperplane that is farthest from the training samples.



Maximal Margin Hyperplane

- ▶ Margin: smallest distance between any training observation and the hyperplane.
- ▶ Support vectors: training observations whose distance to the hyperplane is equal to the margin



Why is it called a support vector?

- ▶ “Support”: maximal margin hyperplane only depends on these observations.
- ▶ “Vector”: points are vectors in p -dimensional space.
- ▶ If support vectors are perturbed, then MM hyperplane will change.
- ▶ If other training observations perturbed (provided not perturbed within margin distance of hyperplane), then MM hyperplane not affected.

- ▶ To find the maximal margin hyperplane on data $(\vec{x}^{(i)}, y^{(i)})$, where $y^{(i)} \in \{-1, 1\}$, solve:

$$\max_{\beta_0, \dots, \beta_p} M$$

- ▶ maximize the margin, M

$$\text{subject to } \sum_{j=0}^p \beta_j^2 = 1$$

- ▶ constraint necessary for well-defined optimization problem

$$y^{(i)} \left(\beta_0 + \beta_1 x_1^{(i)} + \dots + \beta_p x_p^{(i)} \right) \geq M, \quad \forall i$$

- ▶ all training points must be at least distance M from hyperplane

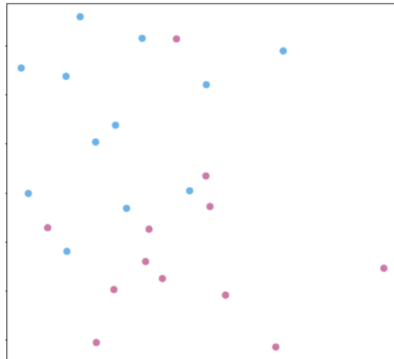
- ▶ To find the maximal margin hyperplane on data $(\vec{x}^{(i)}, y^{(i)})$, where $y^{(i)} \in \{-1, 1\}$, solve:



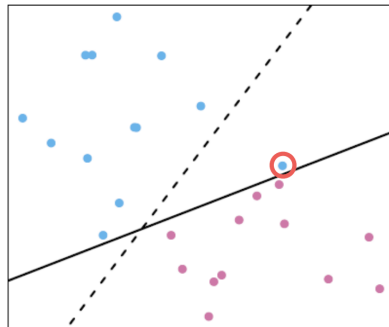
$$\max_{\beta_0, \dots, \beta_p} M$$

- ▶ subject to $\sum_{j=0}^p \beta_j^2 = 1$
- ▶ $y^{(i)} \left(\beta_0 + \beta_1 x_1^{(i)} + \dots + \beta_p x_p^{(i)} \right) \geq M, \quad \forall i$
- ▶ Can be written as a convex optimization problem.
- ▶ We know how to solve convex optimization problems efficiently to find M and β .

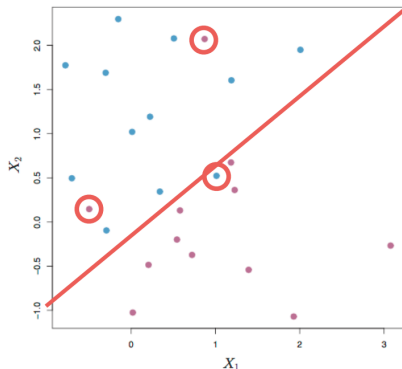
- ▶ Recall the assumption: Classes can be separated by a linear decision boundary.
- ▶ What if there is no separating hyperplane?



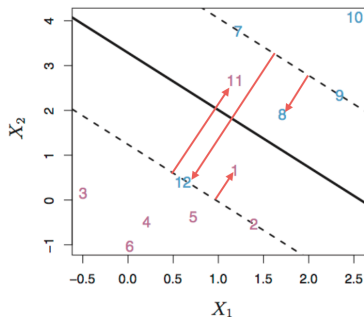
- Furthermore, notice a disadvantage of the maximal margin classifier:
 - Can be sensitive to individual observations
 - May overfit training data



- ▶ Like the maximal margin classifier, it looks for a hyperplane to perform classification.
- ▶ However, training samples are allowed to be on the “wrong side” of the margin or hyperplane.
- ▶ This hyperplane *almost* separates the classes using a “soft margin”.

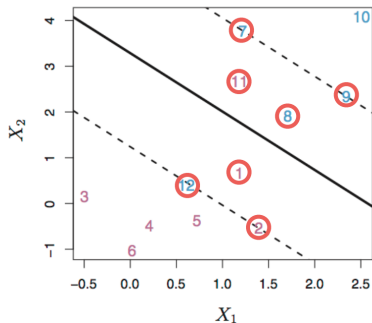


Support Vector Classifier



- Some points are allowed to violate the margin.

أكاديمية كاوست
KAUST ACADEMY



- ▶ Support vector classifiers also have support vectors.
- ▶ They are points lying directly on the margin, or on the wrong side of the margin for their class.
- ▶ These observations affect the hyperplane.

To find the support vector classifier hyperplane, solve:

$$\max_{\beta_0, \dots, \beta_p, \epsilon_1, \dots, \epsilon_n} M$$

$$\text{subject to } \sum_{j=0}^p \beta_j^2 = 1$$

$$y^{(i)} \left(\beta_0 + \beta_1 x_1^{(i)} + \dots + \beta_p x_p^{(i)} \right) \geq M(1 - \epsilon_i), \quad \forall i$$

$$\sum_{i=1}^n \epsilon_i \leq C, \quad \epsilon_i \geq 0, \quad \forall i$$

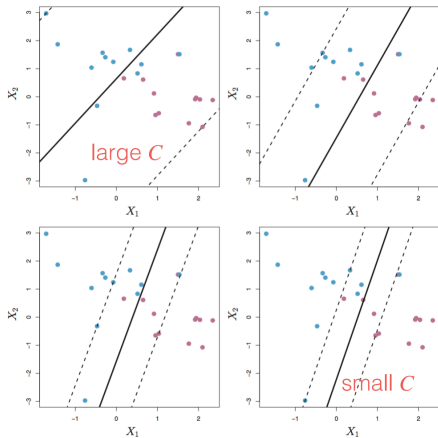
- ▶ Slack variables ϵ_i allow for violations of the margin.
 - $\epsilon_i = 0$: training point is on correct side of margin
 - $\epsilon_i > 0$: training point violates the margin
 - $\epsilon_i > 1$: training point is misclassified (wrong side of hyperplane)
- ▶ Penalty parameter C is the total “budget” for violations.
 - Allows at most C misclassifications on training set.

How do we choose C ?

- ▶ As with many things we don't know *a priori* in machine learning, C is a hyperparameter that we tune using cross-validation.
- ▶ Note that it must be non-negative.
- ▶ If $C = 0$, we recover the maximal margin classifier (if one exists).
- ▶ As C goes from small to large, there is a bias-variance tradeoff.

► Large C

- Large violation budget
- Large margin
- Many support vectors

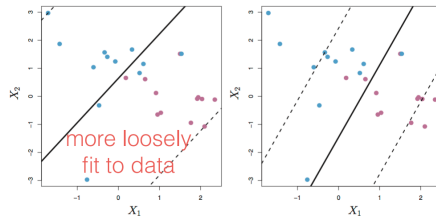


► Small C

- Small violation budget
- Small margin
- Few support vectors

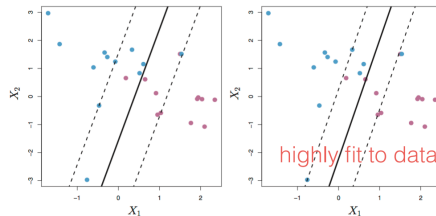
► Large C

- High bias
- Low variance

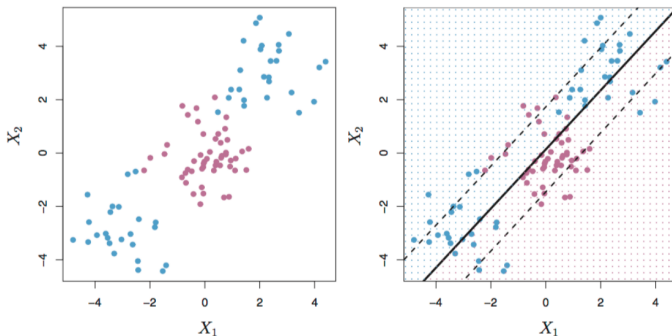


► Small C

- Low bias
- High variance

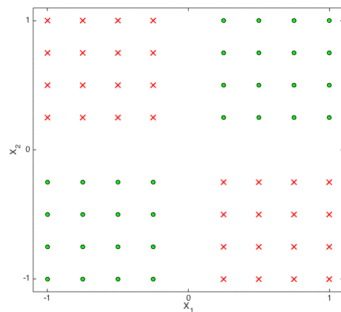


*We are still using a **linear** decision boundary.*

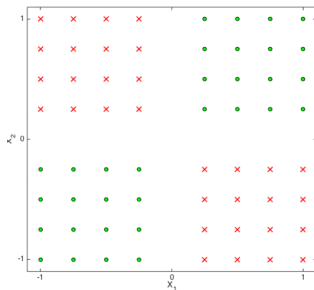


Some datasets are not linearly separable, but they *become* linearly separable when transformed into a *higher* dimensional space.

(Note: Yes, higher dimension also increases chance of overfitting. But in some cases the tradeoff is worthwhile.)

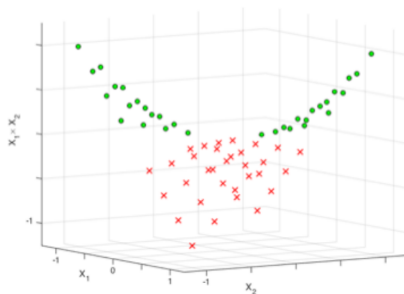


Original feature space



variables x_1, x_2

New feature space



variables x_1, x_2, x_1x_2

- ▶ In linear regression, we created new features to capture non-linearity of data.

- ▶ For example:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \epsilon$$

- ▶ We can apply the same technique to support vector classifiers.

- Suppose our original data has p features.

$$\vec{X} = (X_1, X_2, \dots, X_p)$$

- We can expand the feature space to include e.g. $2p$ features.

$$\vec{X} = (X_1, X_1^2, X_2, X_2^2, \dots, X_p, X_p^2)$$

$$= (\tilde{X}_1, \tilde{X}_2, \tilde{X}_3, \tilde{X}_4, \dots, \tilde{X}_{2p-1}, \tilde{X}_{2p})$$

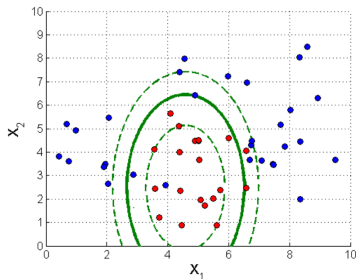
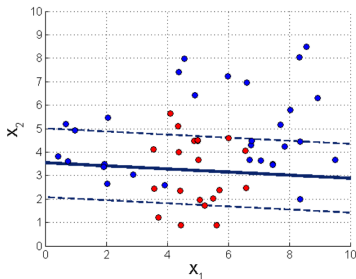
- Support vector classifier will find a hyperplane in $2p$ dimensions:

$$\beta_0 + \beta_1 \tilde{X}_1 + \beta_2 \tilde{X}_2 + \cdots + \beta_{2p-1} \tilde{X}_{2p-1} + \beta_{2p} \tilde{X}_{2p} = 0$$

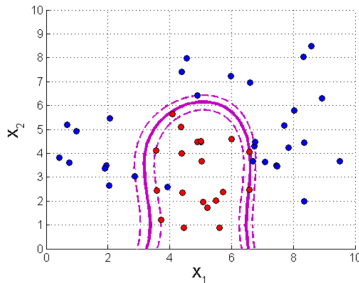
- Hyperplane will be non-linear in *original* feature space. In this case, it is an ellipse:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \cdots + \beta_{2p-1} X_p + \beta_{2p} X_p^2 = 0$$

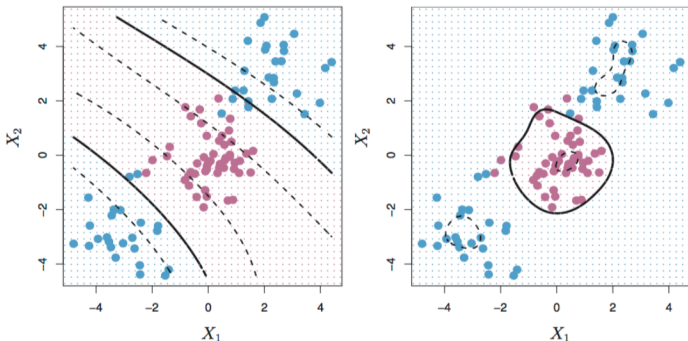
Non-linear Decision Boundary



- ▶ Can imagine adding higher-order polynomial terms, quotients, and more to expand feature set.
- ▶ Large number of features becomes computationally challenging.
- ▶ We need an efficient way to work with large number of features.



- Extends the support vector classifier by using **kernel functions** to achieve non-linear decision boundaries.



- ▶ **Kernel function:** generalization of inner product. It takes in two arguments and *implicitly* computes their inner product in some feature space.
- ▶ Kernels are an efficient computational approach to create non-linear decision boundaries.
- ▶ They (implicitly) map data into a higher-dimensional space.
- ▶ We then apply a support vector classifier in this high-dimensional space with a linear decision boundary (hyperplane).

It can be shown that a support vector classifier can be represented as:

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

- ▶ $f(x) > 0$ is one class, $f(x) < 0$ is another
- ▶ S is the set of support vectors
- ▶ $\langle x, x_i \rangle$ is the **inner product**

Where:

$$\langle \vec{u}, \vec{v} \rangle = \sum_{j=1}^p u_j v_j$$

It can be shown that a support vector classifier can be represented as:

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

In SVM, we replace the inner product with a kernel function:

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

► **Generalization of inner product:**

for an explicit feature map $\phi : \mathcal{X} \rightarrow \mathcal{X}^\phi, \quad x \mapsto \phi(x)$

$$K(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{X}^\phi}$$

► **Symmetric:** $K(x, x') = K(x', x)$

► **Gives a measure of similarity** between X and X'

- If X and X' are close together, then $K(X, X')$ is large
- If X and X' are far apart, then $K(X, X')$ is small

For a more formal definition: <http://mlweb.loria.fr/book/en/constructingkernels.html>

- ▶ **Linear kernel**

$$K(x, x') = \langle x, x' \rangle$$

- ▶ **Polynomial kernel** (degree p)

$$K(x, x') = (1 + \langle x, x' \rangle)^p$$

- ▶ **Radial basis kernel**

$$K(x, x') = \exp(-\gamma \|x - x'\|^2)$$

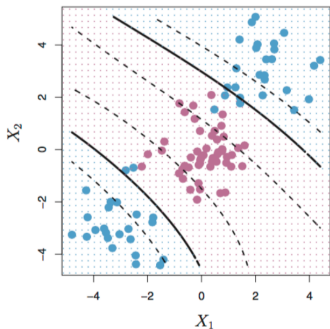
(an infinite-dimensional feature map!)

- ▶ Why use kernels instead of explicitly constructing a larger feature space?
- ▶ Computational advantage:

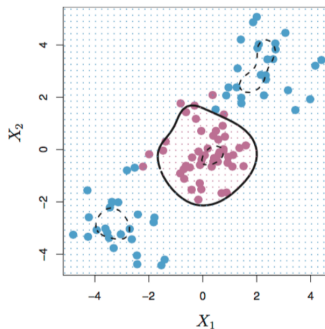
$$\phi : \mathbb{R}^p \rightarrow \mathbb{R}^P, \quad p \ll P$$

$$K(x, x') = \langle \phi(x), \phi(x') \rangle \quad \text{in } \mathcal{O}(p)$$

Cubic polynomial kernel



Radial kernel



Pros:

- ▶ Regularization parameter C helps avoid overfitting
- ▶ Use of kernel gives flexibility in shape of decision boundary
- ▶ Optimization problem is convex — unique solution

Cons:

- ▶ Must tune hyperparameters (e.g. C , kernel function)
- ▶ Must formulate as binary classification
- ▶ Difficult to interpret

- ▶ SVMs are designed for binary classification, given the nature of a separating hyperplane.
- ▶ We can adapt SVMs to perform classification when we have more than 2 classes.
- ▶ Popular approaches:
 - One-versus-one
 - One-versus-all

- ▶ Construct an SVM for each pair of classes.
- ▶ For k classes, this requires training $k(k - 1)/2$ SVMs.
- ▶ To classify a new observation:
 - Apply all $k(k - 1)/2$ SVMs to the observation.
 - Take the most frequent class among the pairwise results as the predicted class.
- ▶ **Con:** computationally expensive for large k .

- ▶ Construct an SVM for each class against the $k - 1$ other classes pooled together.
- ▶ For k classes, this requires training k SVMs.
- ▶ Distance to separating hyperplane is a proxy for classification confidence.
- ▶ To classify a new observation:
 - Choose the class with the highest confidence (i.e., farthest from the hyperplane).
- ▶ **Con:** may exacerbate class imbalances; distance to hyperplane may not correspond well to confidence.

- [1] Sherrie Wang et al. *CME250: Introduction to Machine Learning*. Stanford University, Winter 2019.
<https://web.stanford.edu/class/cme250/>.