SME3006 Machine Learning – 2025 Fall

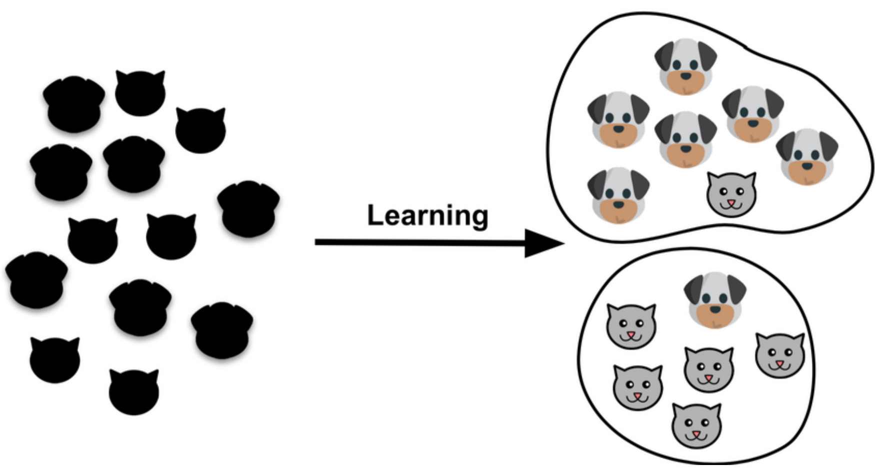
Support Vector Machine

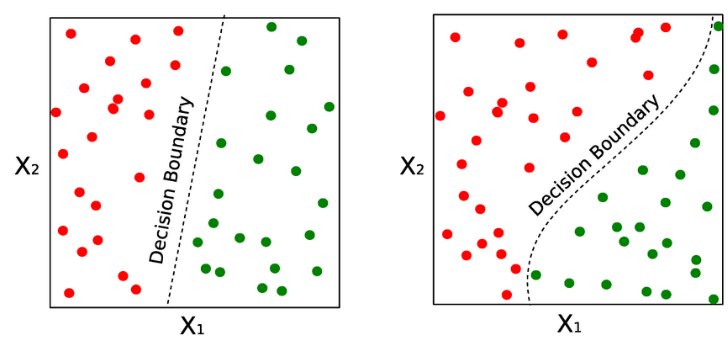
INHA UNIVERSITY

Support vector machines (SVM)

- Supervised method for binary classification (two classes)
  - maximal margin classifier: only applicable to linearly separable data
  - support vector classifier: can be applied to data that is not linearly separable. Decision boundary still linear

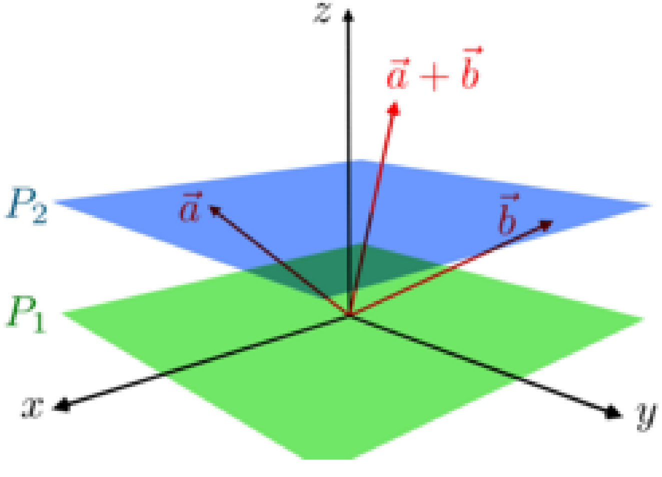
  - SVM: generalization of the above two. Non-linear decision boundary





Hyperplanes

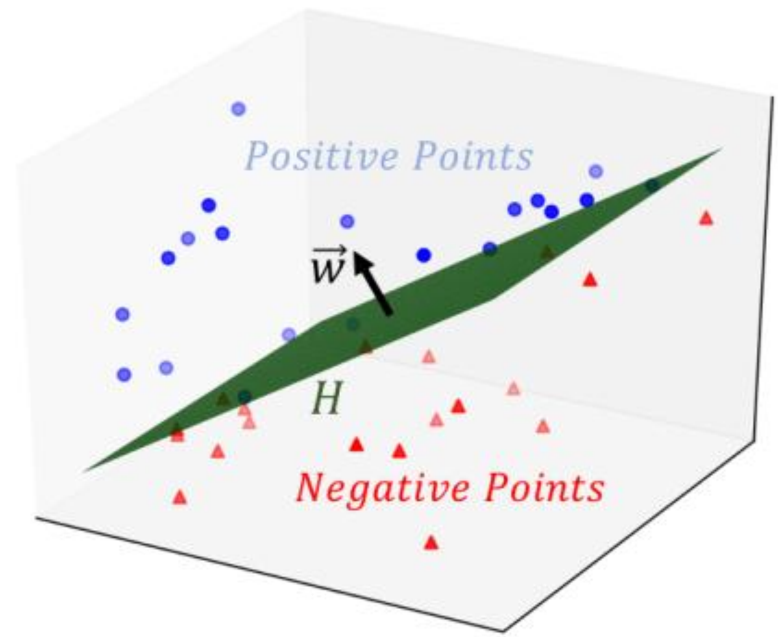
- A hyperplane is a flat, affine subspace of one dimension less than its ambient space
  - i.e., $(n-1)$-dimensional affine subspace in $\mathbb{R}^n$
  - recall: A subspace of a vector space is a subset that is closed under addition and scalar multiplication.
  - Affine subspace: not required to include zero vector, not closed



Hyperplanes

- Defined as set of points $x \in \mathbb{R}^n$ satisfying $w^\top \boldsymbol{x} + b = 0$
  - In 2D, $w_x x + w_y y + b = 0$ (line)
  - In 3D, $w_x x + w_y y + w_z z + b = 0$ (plane)
  - Any point satisfying the equation lies on the hyperplane

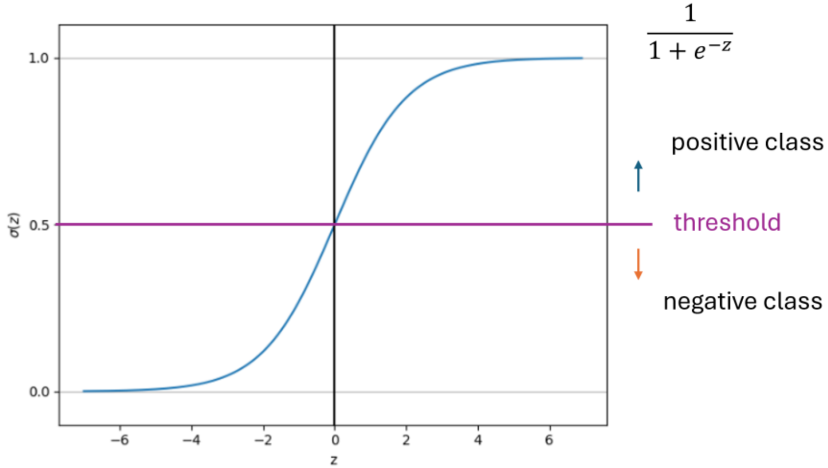
- Separating with a hyperplane
  - $w^\top \boldsymbol{x} + b > 0$ or $w^\top \boldsymbol{x} + b < 0$
  - point lies on either side of the hyperplane
  - it divides the n-dimensional space into two halves

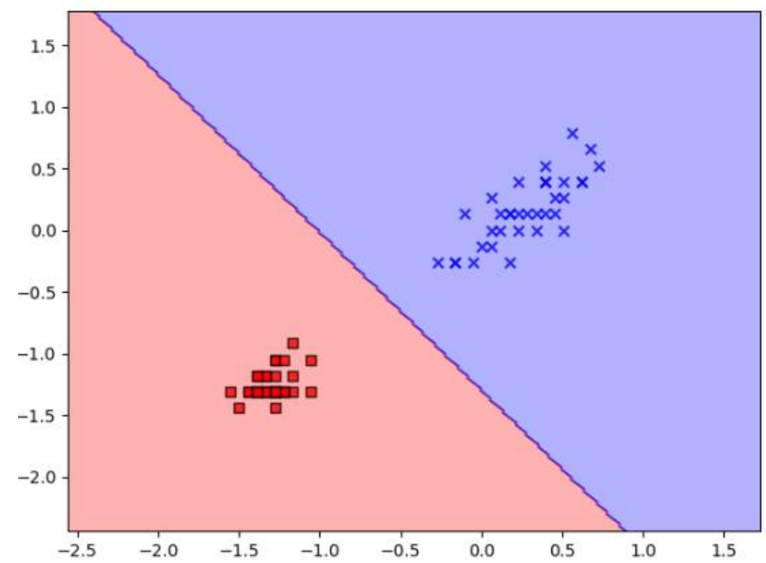


Hyperplanes

> Hyperplane classifier
  - use a separating hyperplane for binary classification
  - key assumption: classes can be separated by a linear decision boundary

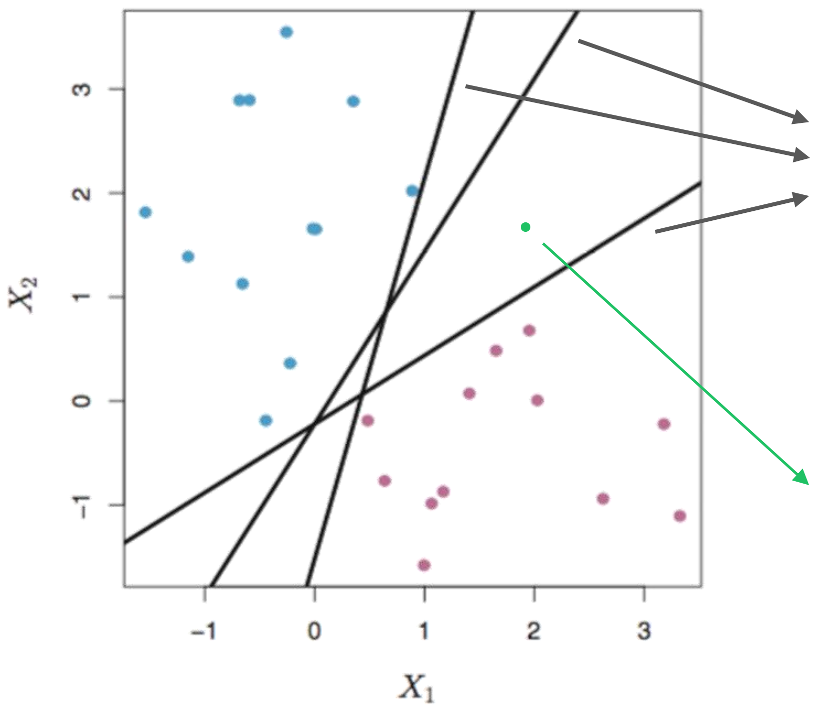
> recall: logistic regression
  - it effectively finds a separating hyperplane





Hyperplanes

> Note that for a linearly separable dataset, there are many possible separating hyperplanes (in fact, an infinite number)
  - they perform equally well on the training set
  - but show different results on the test set (generalization)



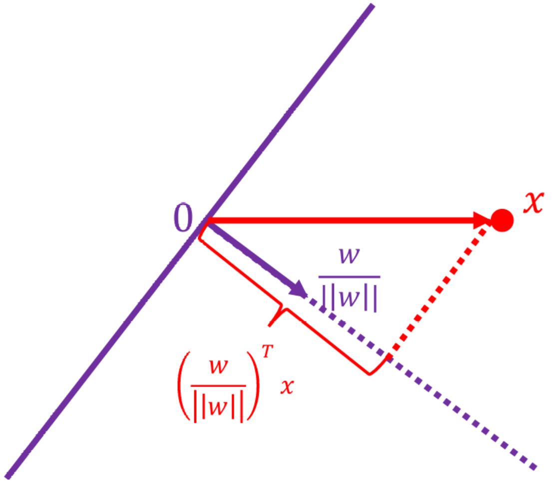
Which one would be an ideal classifier?

If we have a new data point (test set), where should it belong?

Maximal margin hyperplane

> Which hyperplanes should we choose?
  - maximal margin hyperplane: separating hyperplane that is the farthest from the training samples
  - margin: smallest distance between point x_i and the hyperplane

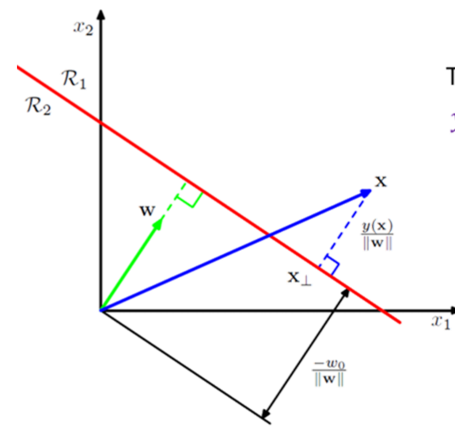
  - For hyperplane, $w^\top x = 0$ (unbiased)
    - w is orthogonal to the hyperplane
    - the unit direction is $w/\|w\|$
    - compute the projection of x_i
    - x_i has distance $|w^\top x_i|/\|w\|$



Maximal margin hyperplane

> Which hyperplanes should we choose?
  - maximal margin hyperplane: separating hyperplane that is the farthest from the training samples
  - margin: smallest distance between point x_i and the hyperplane

- For hyperplane, $w^\top x + b = 0$ (biased)
  - assume unbiased hyperplane HP_o (translated)
  - vectors $d(HP, x_i) + d(HP, HP_o) = d(x_i, HP_o)$
  - w is orthogonal to the hyperplane
  - $d(x_i, HP_o) = w^\top x_i / \|w\|$
  - $d(HP, HP_o) = b / \|w\|$
  - distance $|d(x_i, HP_o)| = |w^\top x_i + b| / \|w\|$



The notation here is:
$y(x) = w^T x + w_0$

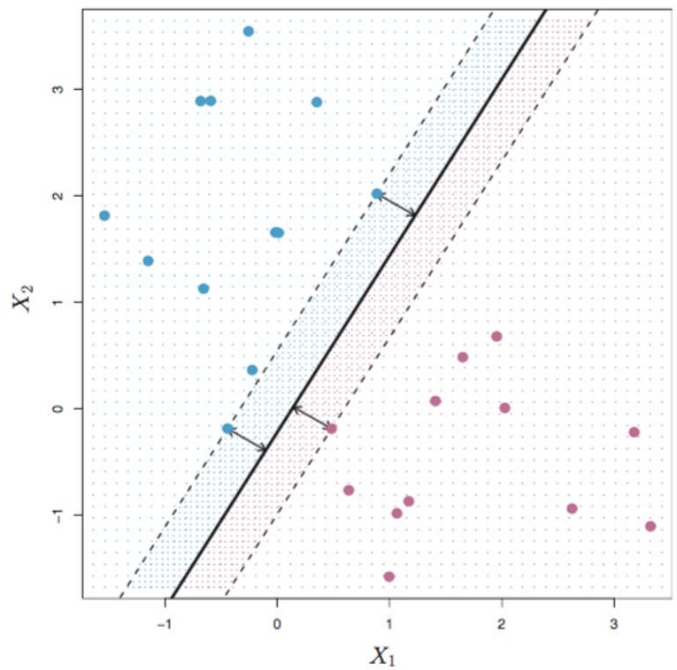
Maximal margin hyperplane

> Which hyperplanes should we choose?
  - maximal margin hyperplane: separating hyperplane that is the farthest from the training samples
  - margin: smallest distance between any point x_i and the hyperplane
  - support vectors: data points that have the distance equal to the margin

  - we want to maximize the margin
    - we have data points $(x_i, y_i),\ y_i \in \{-1, 1\}$
    - $\max_{w} M$

      $s.t.\quad \sum_{j=0}^{n} w_j^2 = 1$

      $$y_i(w^\top x_i + w_0) \geq M,\ \forall i$$
    - constraint necessary for well-defined optimization problem



Maximal margin hyperplane

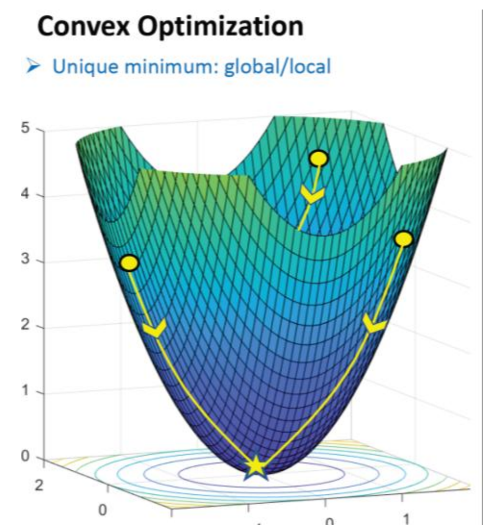
> Optimization problem

- $$\max_{w} |w^\top x + w_0|/\|w\|$$
  $$s.t.\quad \sum_{j=0}^{n} w_j^2 = 1$$
  $$y_i(w^\top x_i + w_0) \geq M,\ \forall i$$

$$\Rightarrow$$

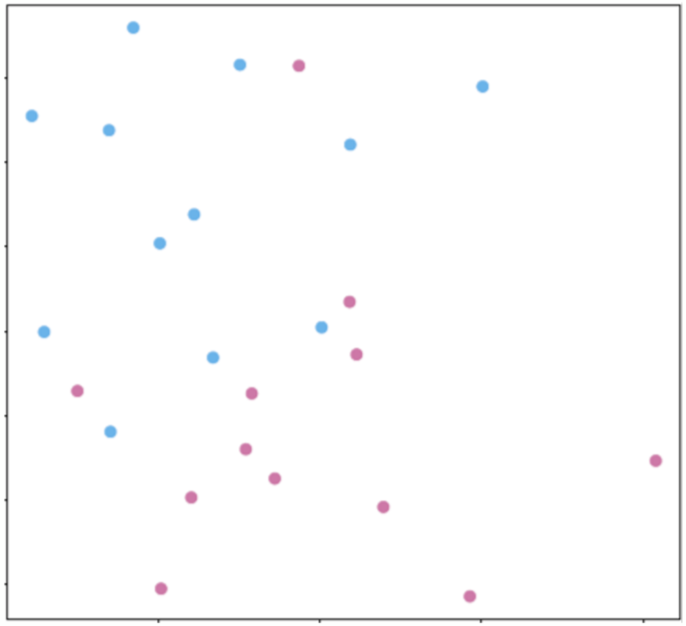
$$\min_{w} \|w\|^2$$
$$s.t.\quad y_i(w^\top x_i + w_0) \geq 1$$

- quadratic cost function and linear constraints: quadratic program (convex)
- Quadratic program (QP)
  - $$\underset{x}{\text{minimize}} \frac{1}{2} x^\top P_0 x + q_0^\top x + c_0$$
    $$s.t. Ax = b$$
    $$\mathrm{x} \geq 0$$



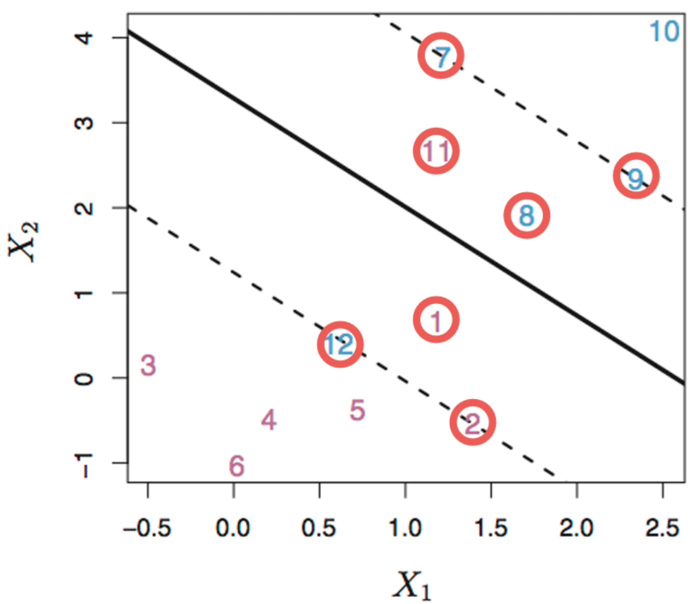
Maximal margin hyperplane

- What if there is no separating hyperplane?
- In addition,
  - it can be sensitive to individual data points
  - it may overfit training data
  - one outlier could ruin the algorithm



Support vector classifier

- Like a maximal classifier, it looks for a hyperplane to perform clasification
- Training samples are allowed to be on the 'wrong side'
- It separates the classes using a soft margin
  - maximal margin hyperplane is a hard-margin SVM
  - now, margin is not the closest distance, it is determined by the parameter
  - support vectors are points within the margin or on the wrong side



Support vector classifier

- We want to maximize the margin
  - we have m data points (x_i, y_i), $y_i \in \{-1, 1\}$
  - $\max_{w} \; M$
    $s.t. \;\; \sum_{j=0}^{n} w_j^2 = 1$ $\longrightarrow$ constraint necessary for well-defined optimization problem

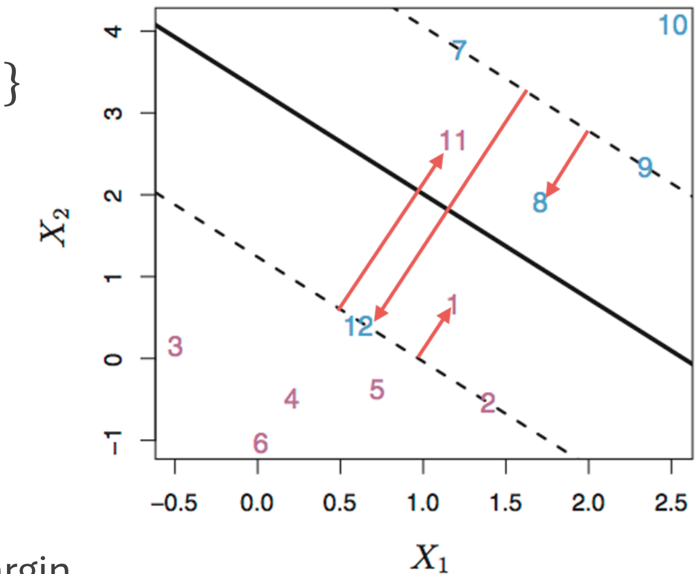
    $$y_i(w^\top x_i + b) \geq M(1 - \epsilon_i), \;\forall i$$
    $$\sum_{i=1}^{m} \epsilon_i \leq C \,, \;\; \epsilon_i \geq 0, \forall i$$

  - points must be at least distance M from hyperplane, or pay a penalty ϵ_i
  - there is a limit on the total penalties C
  - i.e., you can violate the margin, but only by a total amount C

Support vector classifier

- We want to maximize the margin
  - we have m data points $(x_i, y_i),\ y_i \in \{-1, 1\}$
  - $\max_{w}\ M$
    $s.t.\ \ \sum_{j=0}^{n} w_j^2 = 1$

    $$y_i(w^\top x_i + b) \geq M(1 - \epsilon_i),\ \forall i$$
    $$\sum_{i=1}^{m} \epsilon_i \leq C\ ,\ \epsilon_i \geq 0, \forall i$$



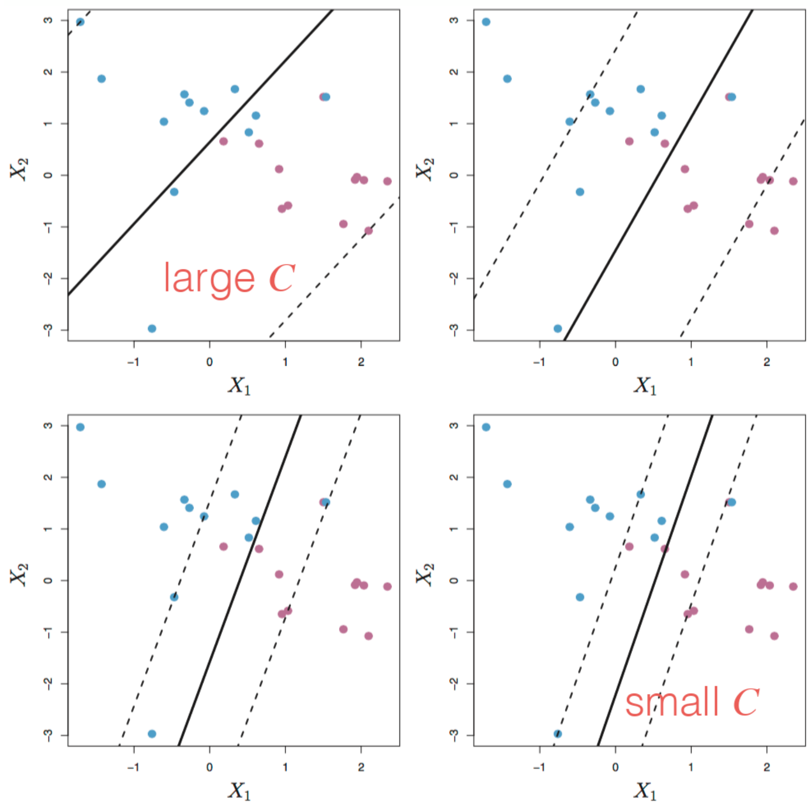
  - Penalties
    - $\epsilon_i = 0$: training point is on correct side of margin
    - $\epsilon_i > 0$: training point violates the margin
    - $\epsilon_i > 1$: training point is misclassified (wrong side of hyperplane)

Support vector classifier

- C determines the total budget for violations
  - it is a hyperparameter that we tune using cross-validation
  - if $C = 0$, we recover the maximal margin classifier
  - As C goes from small to large, there is bias-variance tradeoff

    - Large C
      high bias, low variance
      large margin
      many support vectors

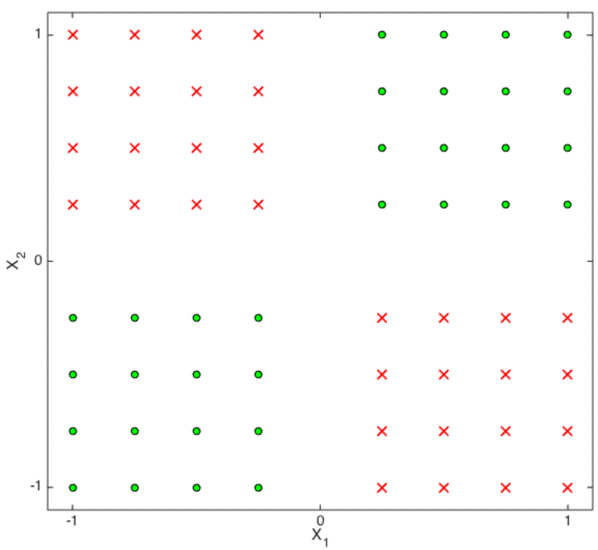
    - Small C
      low bias, high variance
      small margin
      few support vectors



Support vector classifier

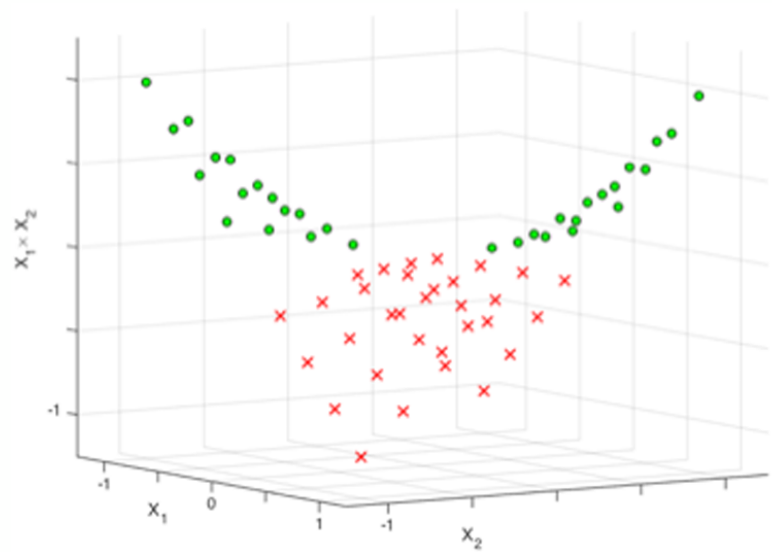
- We are still using a linear decision boundary
  - Some datasets are not linearly separable, but linearly separable when transformed into a higher dimensional space

Original feature space



variables x_1, x_2

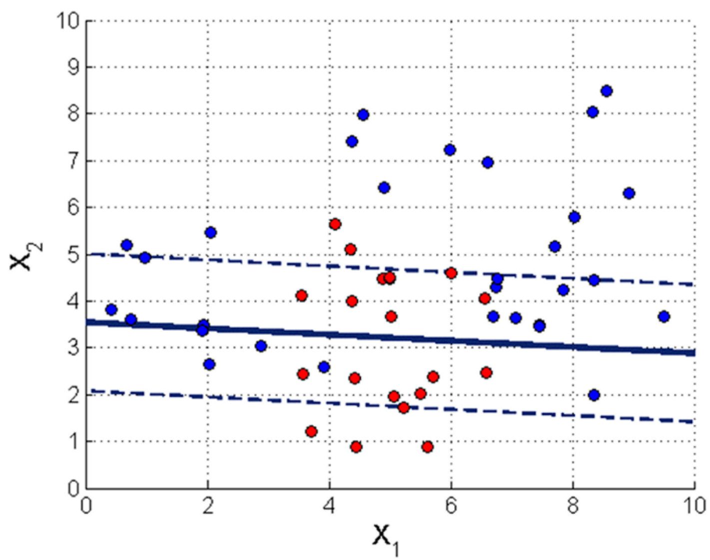
New feature space

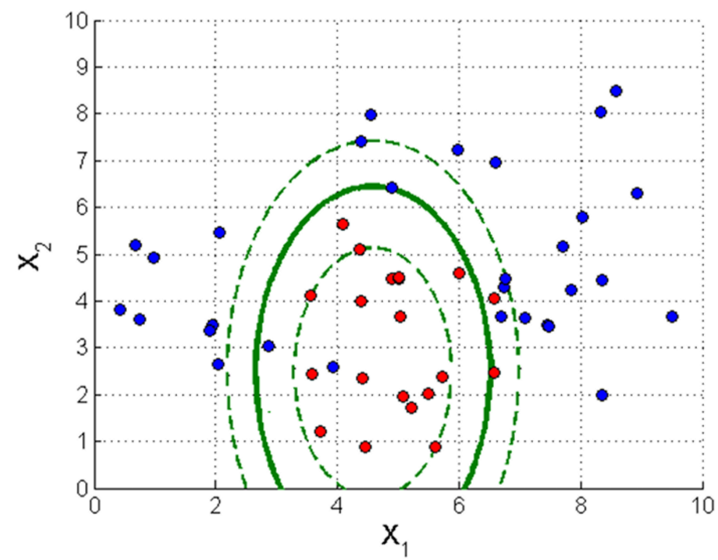


variables x_1, x_2, x_1x_2

Expanding feature space

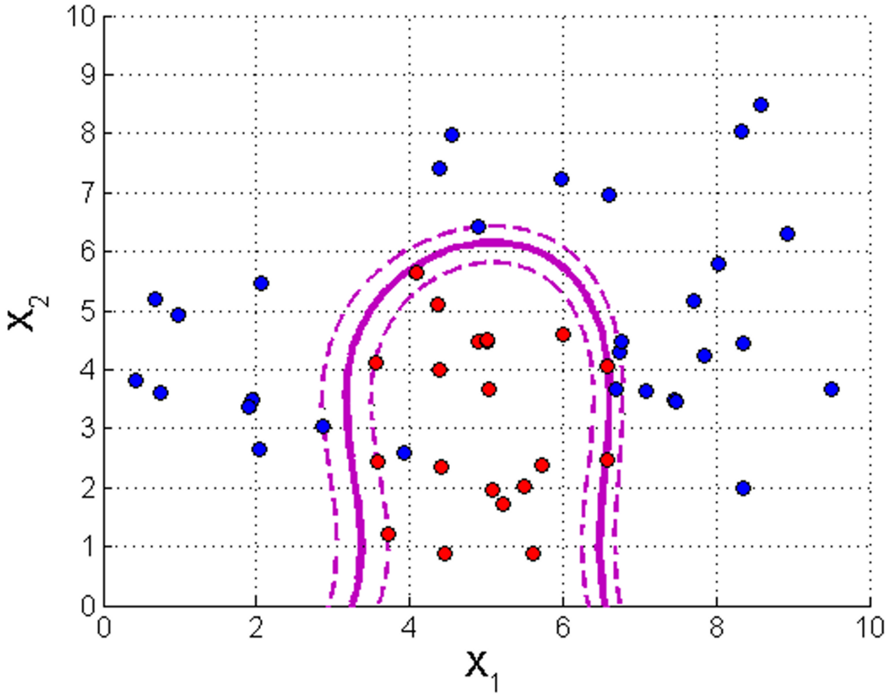
- We are lifting the data into a new feature space
  - original data
    $x = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$
  - lifted data
    e.g., $\tilde{\mathrm{x}} = (x_1, x_2, \ldots, x_n, x_1^2, x_2^2, \ldots, x_n^2) \in \mathbb{R}^{2n}$
  - support vector classifier will find a hyperplane in $2n$ dimensions
  - hyperplane will be nonlinear in the original space
    - in this case, it is an ellipse





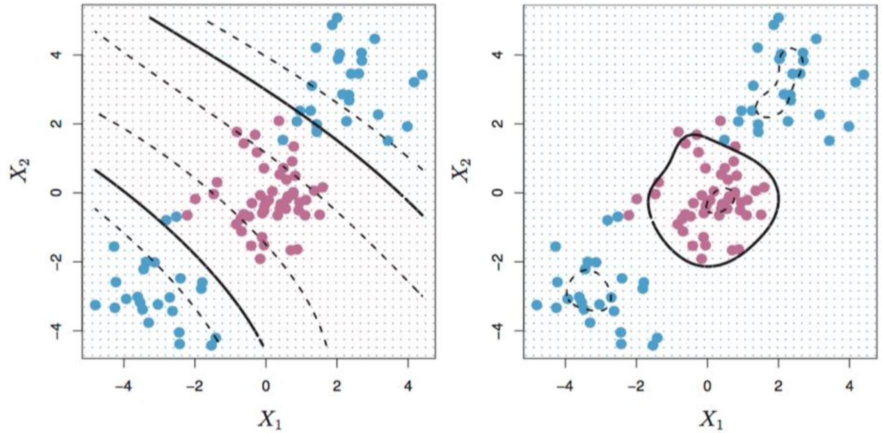
Expanding feature space

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

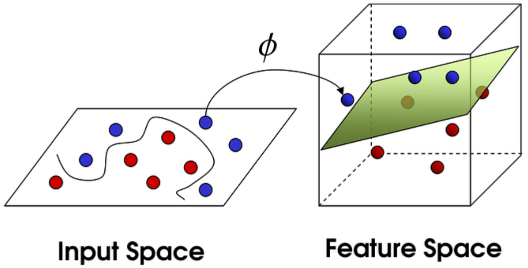


SVM

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



  - recall: kernel function $K(x_i, x_j) = \phi(x_i)^\top \phi(x_j)$
  - they implicitly map data into higher-dimensional space



SVM

- Support vector classifier (linear SVM)
  - hyperplane: $w^\top x + b$,
  - If we solve the optimization problem, $w = \sum_i \alpha_i y_i x_i$ (from the optimality condition of Lagrangian)
  - classifier: $f(x) = w^\top x + b - M \quad \rightarrow \quad f(x) > 0 \;\; or \;\; f(x) < 0$

$$= \Big(\sum_i \alpha_i y_i \underbrace{x_i^\top x}_{\text{inner product}}\Big) + b - M$$

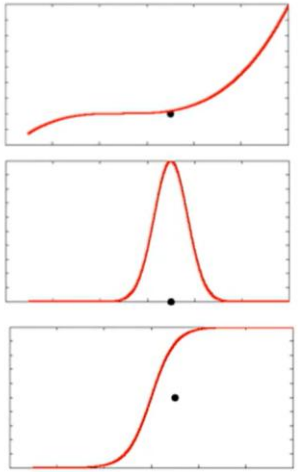
- General SVM
  - we replace the inner product with some kernel function
  - classifier: $f(x) = (\sum_i \alpha_i y_i K(x_i, x)) + b - M$

SVM

- Properties of kernels
  - generalization of inner product without explicitly computing the mapping
    - $\phi\colon\ X \to X^{\phi}$
    - $x \ \mapsto \phi(x)$
    - $K(x, x') = \phi(x)^{\top}\phi(x')$
  - Symmetric: $K(x, x') = K(x', x)$
  - Kernel matrix is positive semidefinite: $K \geq 0$

SVM

- Popular kernels
  - Polynomial: $K(x_i, x_j) = (x_i^\top x_j + c)^d$
  - RBF (Gaussian): $K(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2})$
  - Sigmoid: $K(x_i, x_j) = \tanh(\alpha x_i^\top x_j + c)$
  - Fisher, Neural tangent, Laplacian, Bessel, ...



- Creating more complicated kernels
  - $K(x_i, x_j) = K_1(x_i, x_j) + K_2(x_i, x_j)$
  - $K(x_i, x_j) = \alpha K_1(x_i, x_j)$
  - $K(x_i, x_j) = K_1(x_i, x_j) K_2(x_i, x_j)$

More about RBF kernel

> RBF (Gaussian): $K(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2})$

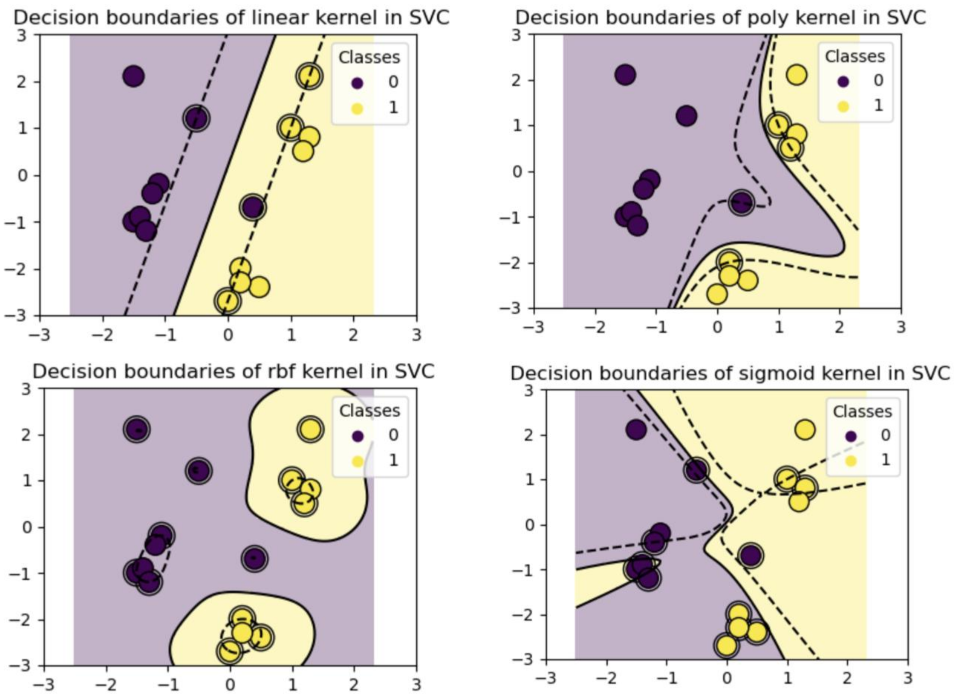
- for $d = 1$, $\phi(x) = \exp\left(-\frac{x^2}{2\sigma^2}\right)\left[1, \frac{x}{\sigma\sqrt{1!}}, \frac{x^2}{\sigma\sqrt{2!}}, \frac{x^3}{\sigma\sqrt{3!}}, \dots\right]^{\mathsf{T}}$
- this is an infinite vector, nobody actually uses this value

- Very popular in practice
  - gives very smooth hypothesis
  - behaves somewhat like k-nearest neighbors, but smoother
  - oscillates less than polynomials

- choose σ by validation
  - σ trades off bias vs. variance
  - larger σ $\rightarrow$ wider Gaussians & smoother h $\rightarrow$ more bias & less variance

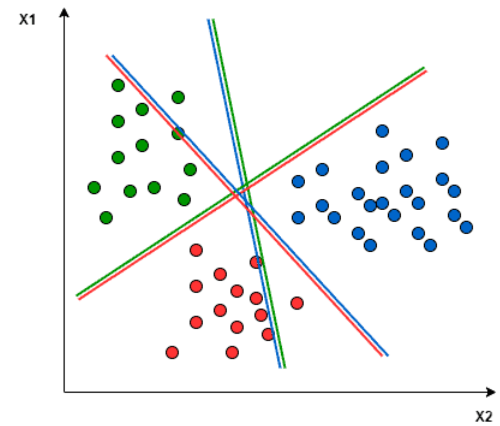
SVM

- Decision boundaries in Kernel SVMs
  - linear: best for linearly separable data; simplest and most interpretable
  - polynomial: captures curved decision boundary, flexibility increases with deg.
  - RBF: highly flexible, local decision boundary
  - sigmoid: inspired by NN, but rarely used due to instability

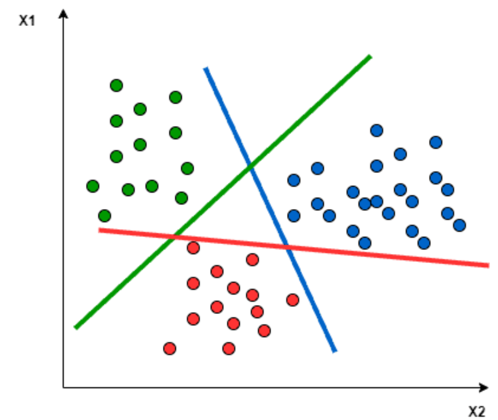


SVM with 3+ classes

> Adapt SVMs to perform classification for more than 2 classes
  - One vs. one
    - construct an SVM for each pair of classes
    - for k classes, this requires training and testing $k(k-1)/2$ SVMs
    - computationally expensive for large k



  - One vs. all
    - construct an SVM for each class against the $k-1$ other classes pooled together
    - for k classes, this requires training and testing k SVMs
    - may exacerbate class imbalances, distance to hyperplane may not correspond well to confidence



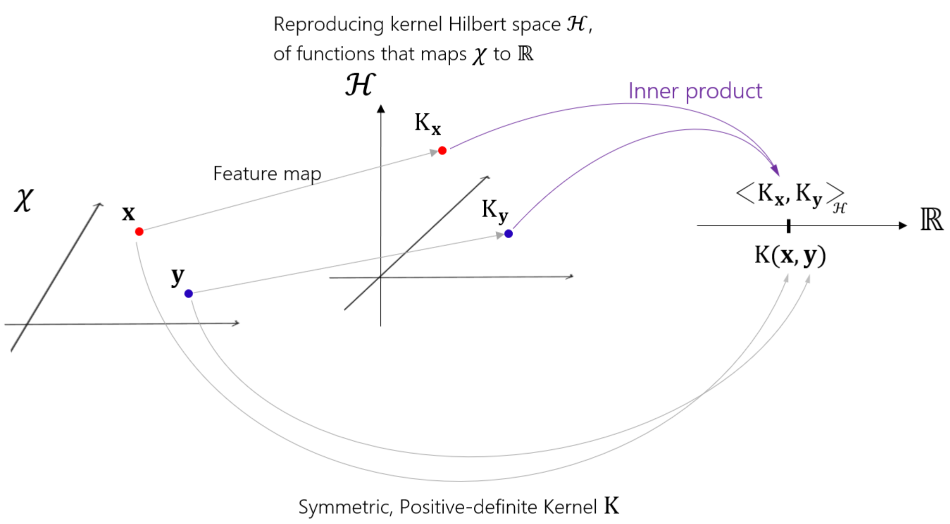
SVM

- Summary
  - regularization parameter C helps avoid overfitting
  - use of kernel gives flexibility in shape of decision boundary
  - optimization problem is convex – unique solution

  - must tune hyperparameters (e.g., C, kernel function)
  - must formulate as binary classification
  - difficult to interpret

RKHS (deeper look into kernel methods)

- Introduction to Reproducing kernel Hilbert space (RKHS)
  - In kernel trick, we don't know the feature map explicitly
  - We're still doing optimization, regularization, and learning in this unknown space
    - Is it okay? Is this optimization mathematically sound?
  - A RKHS is a special Hilbert space of functions satisfying:
    - the kernel function acts as a feature
    - every function in this space satisfies the reproducing property



Hilbert space

> Inner product space containing Cauchy sequence limits (complete)
  - it is like a generalization of Euclidean space to functions

> Inner product: A function $<\cdot,\cdot>_{\mathcal{H}}: \mathcal{H} \times \mathcal{H} \rightarrow \mathbb{R}$
  - linear: $\langle \alpha_1 f_1 + \alpha_2 f_2,\ g\rangle_{\mathcal{H}} = \alpha_1\langle f_1, g\rangle_{\mathcal{H}} + \alpha_2\langle f_2, g\rangle_{\mathcal{H}}$
  - symmetric: $\langle f, g\rangle_{\mathcal{H}} = \langle g, f\rangle_{\mathcal{H}}$
  - $\langle f, f\rangle_{\mathcal{H}} \geq 0$ and $\langle f, f\rangle_{\mathcal{H}=} = 0$ if and only if $f = 0$

> Cauchy sequence
  - a sequence where the elements become arbitrarily close to each other as the sequence progresses
  - ex) 1, 1.4, 1.41, 1.414, 1.4142, … (approaching $\sqrt{2}$) is Cauchy
    In $\mathbb{Q}$ (rational number), its limit $\sqrt{2} \notin \mathbb{Q}$, so not complete
    In $\mathbb{R}$, $\sqrt{2} \in \mathbb{R}$, so $\mathbb{R}$ is a complete space

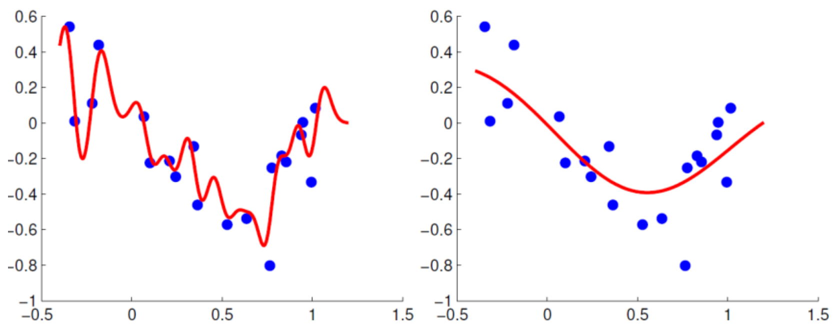
Reproducing property

- Obtain the value of a function $f(x)$ by taking an inner product between the function and the kernel
  - $f(x) = \langle f(\cdot), k(\cdot, x) \rangle_{\mathcal{H}}$ (reproducing property)
  - Imagine $f(\cdot)$ is hidden, but you want to know $f(x)$
  - RKHS is a Hilbert space full of functions
  - A positive semidefinite kernel function leads to the reproducing kernels (Moore-Aronszajn Theorem)
- Then we estimate $\hat{f}(\cdot) = \sum_{i=1}^{n} \alpha_i k(x_i, \cdot)$ (Representer Theorem)
  - In regularized empirical risk minimization problems over an RKHS, the optimal function $\hat{f}$ can always be expressed as a linear combination of kernel functions centered at the training data points
  - $\hat{f}(x) = \langle \sum_{i=1}^{n} \alpha_i k(x_i, \cdot), k(\cdot, x) \rangle_{\mathcal{H}} = \sum_{i=1}^{n} \alpha_i k(x_i, x)$

RKHS

> To sum up,
  - Any symmetric, PSD kernel defines an RKHS (Moore-Aronszajn theorem) (We don't have to define a function space – the kernel defines it)
  - In learning problems, we don't need to search over infinite-dimensional functions – we need a weighted sum of kernels at the data points (Representer theorem)

> RKHS norm $\|f\|_{\mathcal{H}}^2 = \sqrt{\langle f(\cdot), f(\cdot)\rangle_{\mathcal{H}}}$
  - the norm acts as a built-in measure of how complex a function is
  - small norm results in smooth functions -> better generalization



Reference

> SVM

- https://web.stanford.edu/class/cme250/files/cme250_lecture5.pdf
- https://www.cs.princeton.edu/courses/archive/spring16/cos495/slides/ML_basics_lecture4_SVM_I.pdf
- https://www.cs.princeton.edu/courses/archive/spring16/cos495/slides/ML_basics_lecture5_SVM_II.pdf

> RKHS

- https://www.gatsby.ucl.ac.uk/~gretton/coursefiles/rkhscourse.html