Review of SVM and Kernel Trick

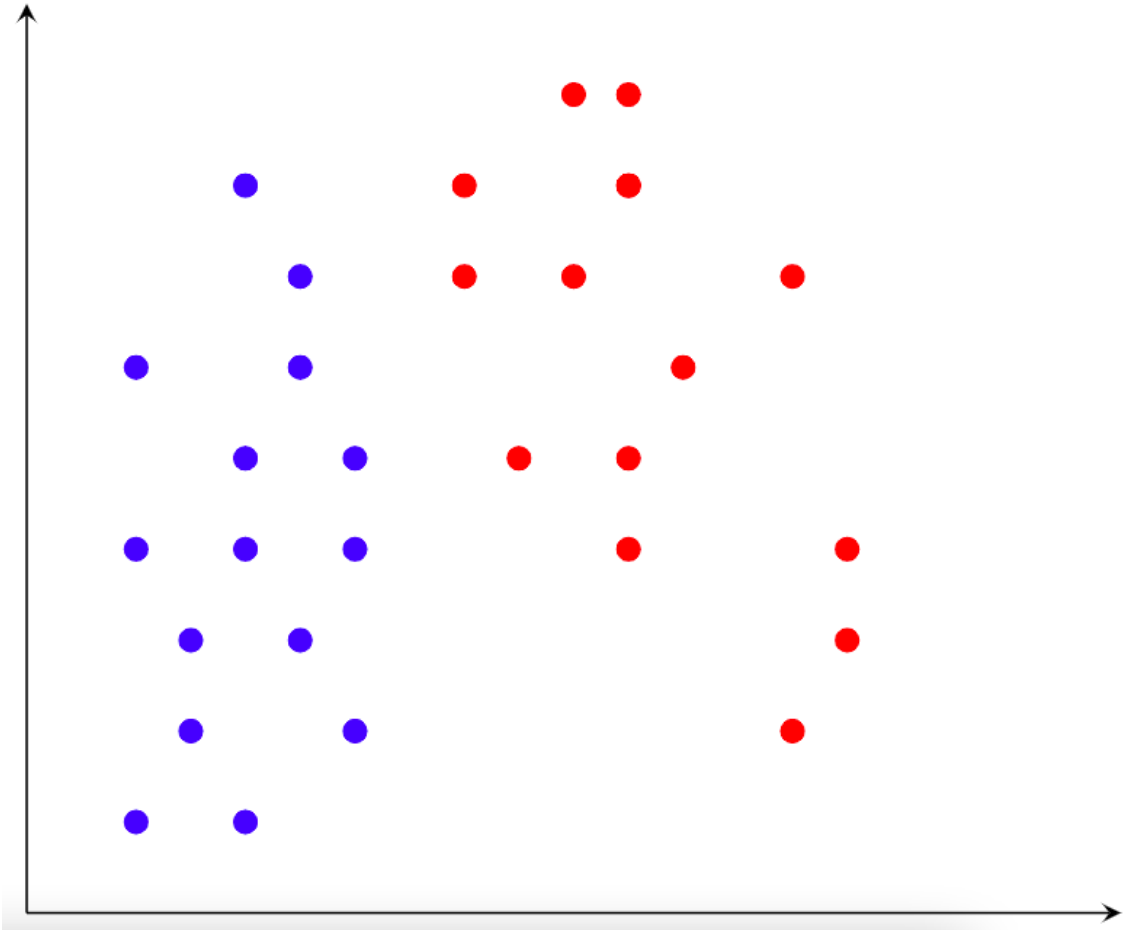
Teaching team of STA314

Department of Statistical Sciences
University of Toronto

Monday November 25th, 2024

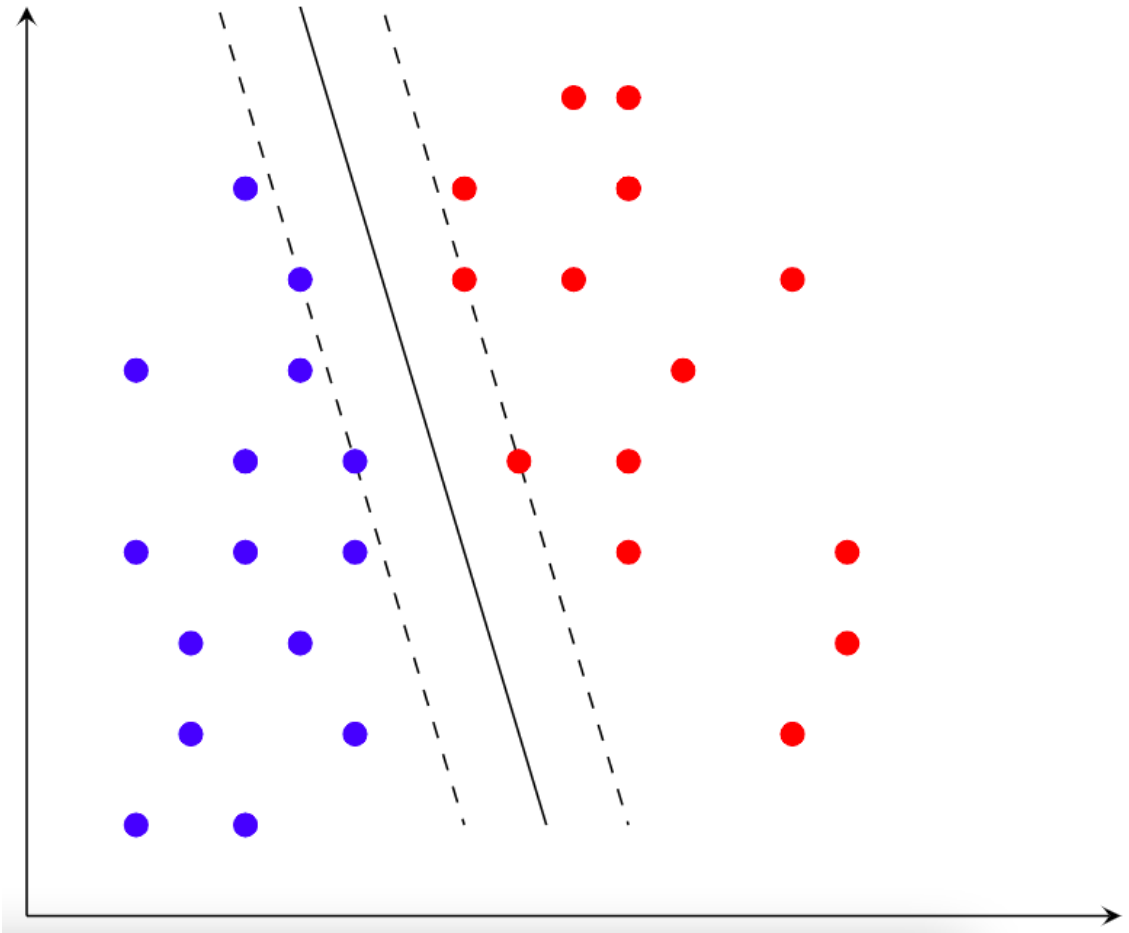
Problem Setting

Our task is to find to separate two sets of points from two classes blue and red with maximum margin between the points and the separating line



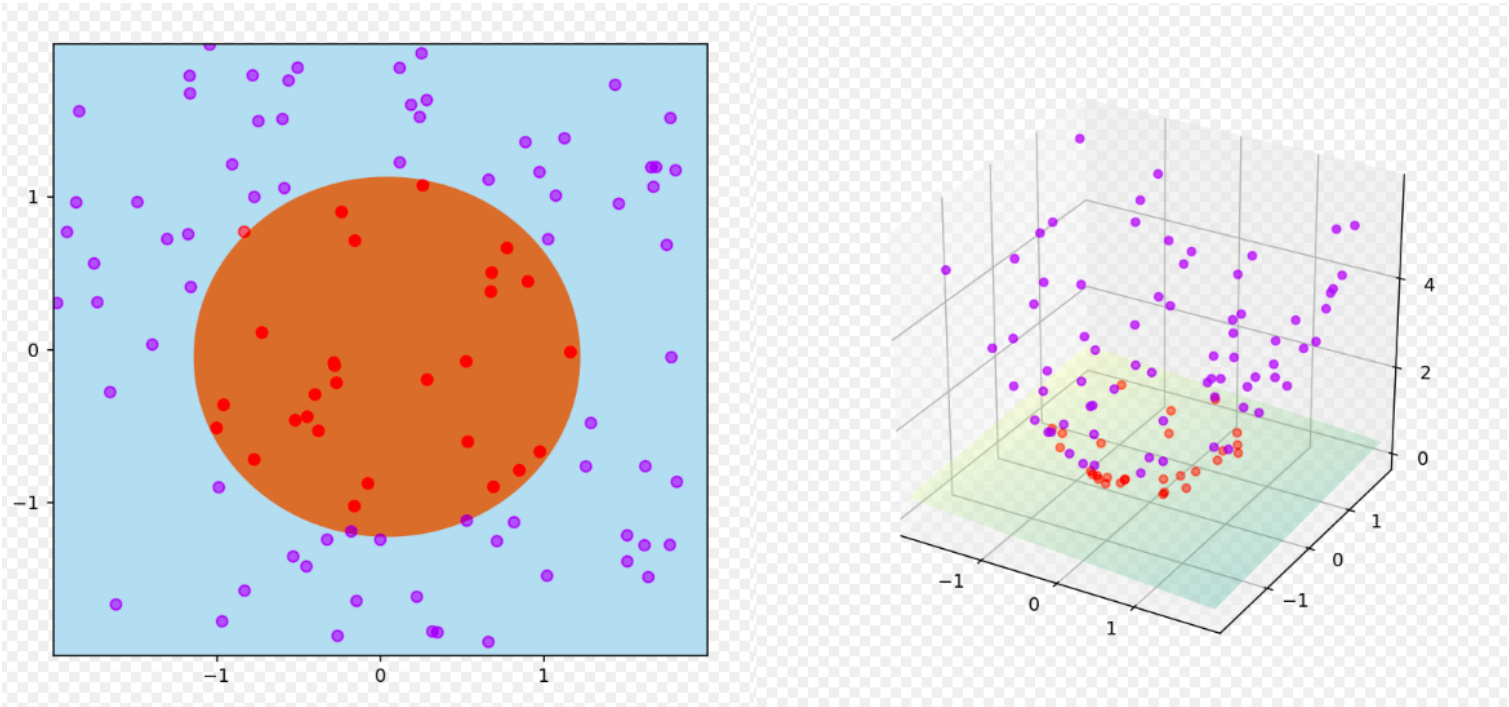
Problem Setting

Our task is to find to separate two sets of points from two classes blue and red with maximum margin between the points and the separating line



Problem Setting

Affine boundaries are often not enough to separate our points



But a clever transformation can make our data linearly separable

$$\phi : (a, b) \mapsto \phi(a, b) = (a, b, a^2 + b^2)$$

Hyperplane Separation

Let us encode our data by $f(\bullet) = 1$ and $f(\bullet) = -1$. We want to find the hyperplane $H : \langle \mathbf{w}, \mathbf{x} \rangle + b = 0$ that separates the two classes while maintaining a *maximum margin* m. (Recall that in a Euclidean space, $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^\top \mathbf{y}$ is a canonical example of an inner product)

$$\implies \max m, \text{ s.t. } f(\mathbf{x}_i)(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq m.$$

Maximizing the margin m equates to solving the constrained optimization problem

$$\max_{\mathbf{w}, b} 2/\|\mathbf{w}\|_2^2$$

$$\text{s.t. } y_i \left(\langle \mathbf{w}, \mathbf{x}_i \rangle + b\right) \geq 1.$$

Maximizing the objective function is equivalent to minimizing its reciprocal, so we may express the Lagrangian of this problem as

$$\mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2} \langle \mathbf{w}, \mathbf{w} \rangle - \sum_{i=1}^{n} \alpha_i \left(1 - f(\mathbf{x}_i)(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)\right).$$

Hyperplane Separation

When the points cannot be completely separated, we may modify the problem by introducing the slack variables ζ_i, giving us the *soft margin* formulation, where for $y_i = f(\mathbf{x}_i)$ we have

Constraints $f(x_i)(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \zeta_i; \quad$ for $\zeta_i \geq 0$.

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2 + \mu \sum_i \zeta_i$$
$$\text{s.t. } y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \zeta_i$$

Where the hyperparameter μ encodes how much we penalize examples that are incorrectly classified, and the Lagrangian would have this extra term as well.[1]

[1] For more details see the supplemental material, or Chapter 12.2.1 in ESL.

Kernel SVM

Now the dual formulation for the previous Lagrangian problem is given by

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left\langle \mathbf{x_i}, \mathbf{x_j} \right\rangle$$

$$\text{s.t.} \sum_{i=1}^{n} \alpha_i y_i = 0, \; 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, n.$$

Where the maximization is over $\boldsymbol{\alpha}$, and depends on the constant products $\left\langle \mathbf{x_i}, \mathbf{x_j} \right\rangle$. For non-affine boundaries, we looked to express $\mathbf{x}_i$ in a different coordinate system for which our data was linearly separable, namely we considered the mapping ϕ, yielding a non-linear boundary (in $\mathbf{x}_i$).

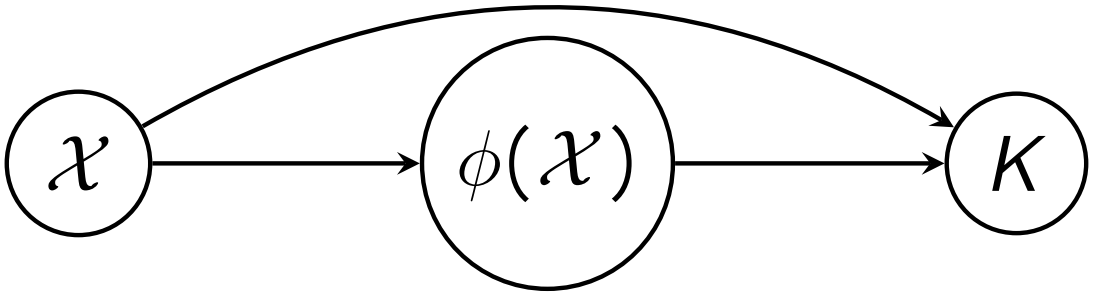
This modified the objective function to become

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left\langle \phi(\mathbf{x_i}), \phi(\mathbf{x_j}) \right\rangle.$$

Kernel Trick

But what is ϕ, and how expensive is it to calculate it?

As it turns out, we don't need to calculate ϕ explicitly!



$$\sum_{i=1}^{n} \alpha_i - \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \right\rangle$$

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$$

$$\boxed{k(\mathbf{x}, \mathbf{x}') = \left\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \right\rangle}$$

Kernel Function

We only need to find a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ which behaves as the inner product in the (typically unknown) feature space $\mathcal{H} = \mathrm{Im}(\phi)$ and thus must inherit the properties of an inner product

- Symmetric: $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$,
- Positive definite: $\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(\mathbf{x}, \mathbf{x}') \geq 0$.

Circle Example

For the circle example we had before we used the transformation

$$\phi : (a, b) \mapsto \phi(a, b) = (a, b, a^2 + b^2)$$

but we could just compute its corresponding kernel directly as

$$\begin{aligned}\left\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \right\rangle &= \left\langle (x_1, x_2, x_1^2 + x_2^2), (x_1', x_2', {x_1'}^2 + {x_2'}^2) \right\rangle \\ &= x_1 x_1' + x_2 x_2' + (x_1^2 + x_2^2)({x_1'}^2 + {x_2'}^2)\end{aligned}$$

So the corresponding kernel is given by

$$k(\mathbf{x}, \mathbf{x}') = \|x\|^2 \|x'\|^2 + \left\langle x, x' \right\rangle$$

Note how we can calculate $k(\mathbf{x}, \mathbf{x}')$ directly, without needing to compute $\phi(\mathbf{x})$ or $\phi(\mathbf{x}')$.

Example: Linear Regression

Kernels can also be useful for regression problems.

Recall that the problem of predicting y as a linear function of $\mathbf{x}$, from some samples $\{(\mathbf{x}_i, y_i)\}$ may be written as

$$\hat{y} = \mathbf{x}^\top \hat{\boldsymbol{\beta}} = \sum_{i=1}^{n} \alpha_i \langle \mathbf{x}, \mathbf{x}_i \rangle$$

Where $\hat{\boldsymbol{\beta}} = (X^\top X)^{-1} X^\top y$. Note that, assuming $n = p$ and the data matrix X is full rank (otherwise we would need the pseudo-inverse), we have

$$(X^\top X) X^\top = X^\top (X X^\top)$$

$$(X^\top X)^{-1} (X^\top X) X^\top (X X^\top)^{-1} = (X^\top X)^{-1} X^\top (X X^\top)(X X^\top)^{-1}$$

$$(X^\top X)^{-1} X^\top = X^\top (X X^\top)^{-1},$$

so we have $\implies \mathbf{x}^\top \hat{\boldsymbol{\beta}} = \mathbf{x}^\top X^\top (X X^\top)^{-1} y =: \mathbf{x}^\top X^\top \boldsymbol{\alpha}$.

Example: Linear Regression II

With the previous identity, we may express our linear regression as

$$\hat{y} = \mathbf{x}^\top \hat{\boldsymbol{\beta}} = \sum_{i=1}^{n} \alpha_i \langle \mathbf{x}, \mathbf{x}_i \rangle,$$

where $\boldsymbol{\alpha} = (XX^\top)^{-1} y$.

So we can implicitly map our variables into a different space, changing their coordinates, by using a kernel function:

$$\hat{y} = \hat{\boldsymbol{\beta}}^\top \mathbf{x} = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}, \mathbf{x}_i),$$

replacing the inner product. Now our data is not constrained to lie on an affine subspace.

More Kernel Examples

Consider a dataset $\{\mathbf{x}_i\}_{i=1}^n \subset \mathbb{R}^m$, and assume the data can be separated by a polynomial. The polynomial kernel is given by

$$k(\mathbf{x}, \mathbf{x}') = \left(\left\langle \mathbf{x}, \mathbf{x}' \right\rangle + c\right)^d$$

Which requires $m + 2$ operations to compute. If we were to compute the transformation ϕ such that $k(\mathbf{x}, \mathbf{x}') = \left\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \right\rangle$ for $d = 2$ we would have to calculate:

$$\phi(\mathbf{x}) = (x_1^2, \ldots, x_m^2, \sqrt{2}x_1x_2, \sqrt{2}x_1x_3, \ldots, \sqrt{2}x_{m-1}x_m, \sqrt{2c}x_1, \ldots, \sqrt{2c}x_m, c)$$

Which is a space of dimension $2m + \binom{m}{2} + 1$. The number of operations required would be $4m + 2\binom{m}{2}$ just for the outputs of ϕ, plus those of computing the inner product.

More Kernel Examples

Again for a dataset $\{\mathbf{x}_i\}_{i=1}^n \subset \mathbb{R}^m$, and consider the *cosine* kernel, given by:

$$k(\mathbf{x}, \mathbf{x}') = \frac{\langle \mathbf{x}, \mathbf{x}' \rangle}{\|\mathbf{x}\| \|\mathbf{x}'\|}.$$

Which requires $O(m)$ operations to calculate.

The corresponding feature map would be

$$\phi(\mathbf{x}) = \frac{\mathbf{x}}{\|\mathbf{x}\|},$$

which would have an identical computational cost of $O(m)$.

For this example, the cost is exactly the same, given the simplicity of the feature map, which makes the data scale invariant. This example is special, as the kernel trick usually makes computations a lot more efficient.

Example: RBF

The *Radial Basis Function* kernel has an expression

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

This kernel may be expanded by Taylor expansion as

$$\exp\left(-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right)\exp\left(-\frac{\|\mathbf{x}'\|^2}{2\sigma^2}\right)\left(1 - \frac{\langle \mathbf{x}, \mathbf{x}' \rangle}{\sigma^2} + \frac{\langle \mathbf{x}, \mathbf{x}' \rangle^2}{2!\sigma^2} - \frac{\langle \mathbf{x}, \mathbf{x}' \rangle^3}{3!\sigma^2} + \dots\right)$$

for which the corresponding feature map (taking $\mathbf{x} \in \mathbb{R}^1$ for simplicity) is

$$\phi(\mathbf{x}) = e^{-\mathbf{x}^2/2\sigma^2}\left(1, \sqrt{\frac{1}{1!\sigma^2}}\mathbf{x}, \sqrt{\frac{1}{2!\sigma^4}}\mathbf{x}^2, \sqrt{\frac{1}{3!\sigma^6}}\mathbf{x}^3, \dots\right)$$

What would be the output space of ϕ here? What is its dimension?