Fundamentals of Algorithm Design and Machine Learning Scribe

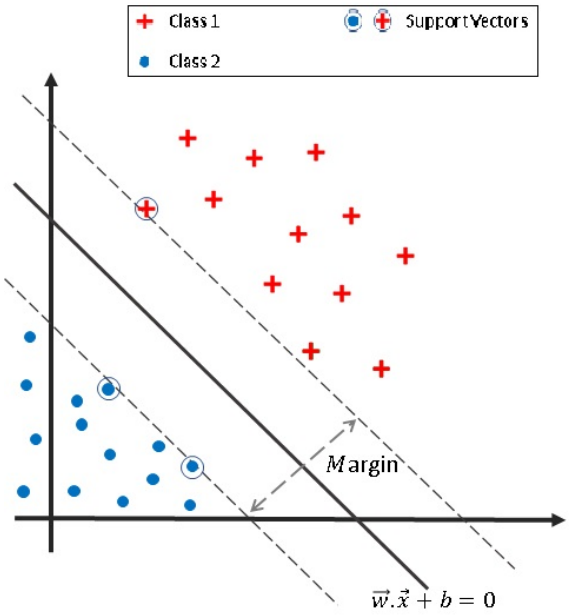
Professor Name: Aritra Hazra

Name: **Sankalp Davi**

Roll no: **24BM6JP48**

24th March 2025

1 Recap - SVM Optimization Problem



For SVM, the PRIMAL optimization problem is:

$$\min \frac{1}{2} W^T W \tag{1}$$

subject to constraints

$$y_i(W^T x_i + b) \geq 1 \tag{2}$$

$$\forall (x_i, y_i) \tag{3}$$

where W is the coefficient of the hyperplane of SVM, b is the bias and (x_i, y_i) is the coordinate of the training data.

Using the KKT theorem, the dual optimization problem of the above-mentioned constraint (according to the Lagrange multiplier) is given by

$$minL_p = \frac{1}{2}W^T W + \sum_{i=1}^{n} \alpha_i \left[1 - y_i(W^T x_i + b)\right] \tag{4}$$

subject to constraints

$$\alpha_i \geq 0 \tag{5}$$

The value L after taking the derivative of L with respect to W and b and setting them to zero becomes :

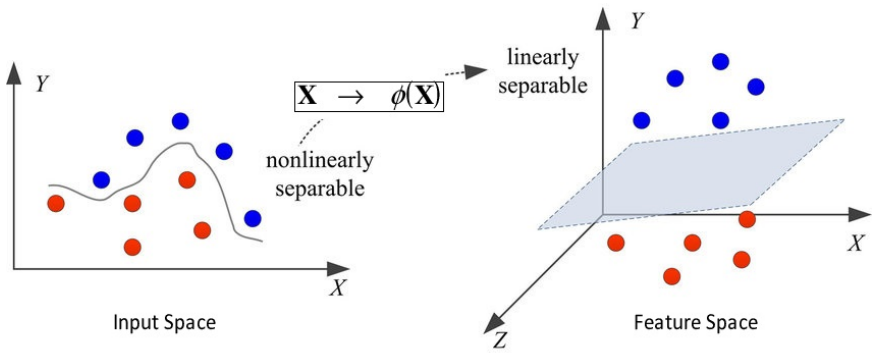
$$L = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (Z_i, Z_j) \tag{6}$$

α_i 's are Lagrange's Multipliers

The calculation of the term $\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (Z_i, Z_j)$ is more straightforward if we compute $y_i y_j (Z_i, Z_j)$ corresponding as the Hessian matrix. This matrix is precomputed and stored as H

2 Transformation

When no linear classification is possible in X-dimensional space, i.e., to work with non-linear decision boundaries, the key idea is to transform x_i to a higher-dimensional space Z using a transformation function $\phi(x)$ so that in this new space, the samples can be linearly separable. finally, upon getting a linear separable boundary, we transformed back again using $\phi^{-1}(x)$



Transformation of X into Z space using $\phi(x)$ which is the dot product

$$X = (1, x_1, x_2)^T \tag{7}$$

$$X' = (1, x'_1, x'_2)^T \tag{8}$$

$$X.X' = 1 + x_1 x'_1 + x_2 x'_2 \tag{9}$$

In our example,

$$Z = \phi(X) = (1, x_1, x_2, x_1^2, x_2^2, x_1 x_2) \tag{10}$$

$$Z = \phi(X') = (1, x'_1, x'_2, x'^2_1, x'^2_2, x'_1 x'_2) \tag{11}$$

This is a complete transformation process of Z from the x where x is two attribute point, i.e., $x = (x_1, x_2)$

Here $Z^T Z$ is nothing but $K(X, X')$
Then,

$$Z^T Z = K(X, X') = (1, x_1 x_1', x_2 x_2', x_1^2 x_1'^2, x_2^2 x_2'^2, x_1 x_2 x_1' x_2') \tag{12}$$

3. Kernel Trick

The trick is to compute $K(X, X')$ directly in the input space instead of taking a dot product for each point, which implicitly corresponds to a dot product in a higher-dimensional space. The kernel function allows us to work in a high-dimensional space without ever computing $\phi(x)$ explicitly.

Example :

$$K(X, X') = (1 + x_1 x_1' + x_2 x_2')^2 \tag{13}$$

Expanding the expression:

$$(1 + x_1 x_1' + x_2 x_2')^2 = 1 + x_1^2 x_1'^2 + x_2^2 x_2'^2 + 2x_1 x_1' + 2x_2 x_2' + 2x_1 x_2 x_1' x_2' \tag{14}$$

$$(1 + x_1 x_1' + x_2 x_2')^2 = (x_1^2, x_2^2, \sqrt{2}x_1 x_2, \sqrt{2}x_1, \sqrt{2}x_2, 1)^T \cdot (x_1'^2, x_2'^2, \sqrt{2}x_1' x_2', \sqrt{2}x_1', \sqrt{2}x_2', 1) \tag{15}$$

This corresponds to the feature mapping:

$$\phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1 x_2, \sqrt{2}x_1, \sqrt{2}x_2, 1) \tag{16}$$

and

$$\phi(x') = (x_1'^2, x_2'^2, \sqrt{2}x_1' x_2', \sqrt{2}x_1', \sqrt{2}x_2', 1) \tag{17}$$

Thus:

$$K(X, X') = (X \cdot X' + 1)^Q = \phi(x) \cdot \phi(x') = (1 + x_1 x_1' + x_2 x_2' + ... + x_d x_d') \tag{18}$$

These are called **Polynomial Kernels**

Now , Our optimization problem is modified as (Z_i, Z_j) gets changed to $K(X, X')$ after using the kernel trick. So, the value of L is:

$$L = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (K(X_i, X_j)) \tag{19}$$

The benefit of using a kernel trick is that it allows us to visit any higher-dimensional space and return with the desired result using no extra memory and with a minimal effect on computation time. Furthermore, we avoid the burden of explicitly computing the feature mappings and their dot products in the infinite-dimensional space. Rather, the kernel trick suggests that we do not have to visit all these dimensions to compute this. This computation only requires knowledge of the kernel.

In the kernel trick, when we transfer data to a higher-dimensional space, we may assume that our generalization boundary is becoming too severe. However, this isn't always the case because, in some cases, the discriminant is limited to support vectors, and a generalization bound to a smaller number of support vectors severely restricts our dimension.

$$E[E_{\text{out}}] \leq \frac{E[\text{No. of support vectors}]}{(N-1)}, \tag{20}$$

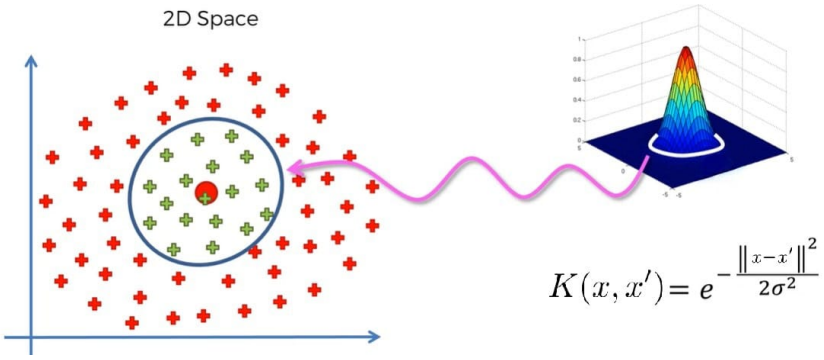
where N = number of points

4. Radial Basis Function (RBF)

The RBF kernel, also known as the Gaussian kernel, is defined as follows:

$$K(x, x') = e^{-\frac{\|x-x'\|^2}{2\sigma^2}} \tag{21}$$

The RBF kernel maps data into an infinite-dimensional space, making it very flexible for capturing complex patterns.



The parameter γ controls the width of the Gaussian curve, which affects how localized the influence of each data point is. γ If it is large, then the Gaussian curve will be much steeper, and if it is small, it will be much smoother.

For definiteness, let's define it $\gamma = \frac{1}{2\sigma^2}$, so the kernel becomes:

$$K(x, x') = e^{-\gamma\|x-x'\|^2} \tag{22}$$

Expanding $\|x - x'\|^2$:

$$\|x - x'\|^2 = (x - x') \cdot (x - x') = \|x\|^2 + \|x'\|^2 - 2(x \cdot x') \tag{23}$$

Substitute this into the kernel:

$$K(x, x') = e^{-\gamma(\|x\|^2+\|x'\|^2-2(x\cdot x'))} = e^{-\gamma\|x\|^2} e^{-\gamma\|x'\|^2} e^{2\gamma(x\cdot x')} \tag{24}$$

For simplicity assume $\gamma = 1$ Using the Taylor Series expansion for the exponential function $e^z = \sum_{k=0}^{\infty} \frac{z^k}{k!}$, expand the term $e^{2\gamma(x\cdot x')}$:

$$e^{2\gamma(x\cdot x')} = \sum_{k=0}^{\infty} \frac{(2(x \cdot x'))^k}{k!} = \sum_{k=0}^{\infty} \frac{(2)^k (x \cdot x')^k}{k!} \tag{25}$$

Thus, the kernel becomes:

$$K(x, x') = e^{-\|x\|^2} e^{-\|x'\|^2} \sum_{k=0}^{\infty} \frac{(2)^k (x \cdot x')^k}{k!} \tag{26}$$

The expansion shows that the RBF kernel can be written as an infinite sum, where each term corresponds to a component in the feature space. The outcome can be interpreted as a dot product in an infinite-dimensional feature space:

$$K(x, x') = \phi(x) \cdot \phi(x') \tag{27}$$

Where the feature mapping $\phi(x)$ has components involving terms like $e^{-\gamma\|x\|^2} \frac{(2\gamma)^{k/2} x^k}{\sqrt{k!}}$ for $k = 0, 1, 2, \ldots$.

5. Conditions for a Function to be a Valid Kernel

In general K, the kernel $(x \cdot x')^K$ expands into a sum of terms, each corresponding to a component of the feature mapping $\phi(x)$.

$K(x_i, x_j)$ This matrix is symmetric and positive semi-definite.

$$\begin{bmatrix} K(x_1,x_1) & K(x_1,x_2) & \cdots & K(x_1,x_n) \\ K(x_2,x_1) & K(x_2,x_2) & \cdots & K(x_2,x_n) \\ \vdots & \vdots & \ddots & \vdots \\ K(x_n,x_1) & K(x_n,x_2) & \cdots & K(x_n,x_n) \end{bmatrix}$$

1.) Structural Consistency of Feature Mappings
The comparison is made with constant functions, which means the structure of the feature mappings $\phi(x)$ and $\phi(x')$ must be the same. For a kernel to exist, both $\phi(x)$ and $\phi(x')$ should have the same structure (e.g., the same dimensionality and form of the transformed features). This procedure ensures that the inner product $\phi(x) \cdot \phi(x')$ is well-defined and consistent across all pairs of points.

2.) Mercer's Theorem
It states that for a symmetric function $K(x, x')$ to be a valid kernel, the above matrix must be positive semi-definite. Specifically:

- **Symmetry**: The kernel must satisfy $K(x, x') = K(x', x)$ for all x, x'.
- **Positive Semi-Definiteness**: For any finite set of points $\{x_1, x_2, \ldots, x_N\}$, the kernel matrix (Gram matrix) K with entries $K_{ij} = K(x_i, x_j)$ must be positive semi-definite. This means that for any vector $c \in R^N$,
$$\sum_{i=1}^{N} \sum_{j=1}^{N} c_i c_j K(x_i, x_j) \geq 0$$

Mercer's Theorem guarantees that if $K(x, x')$ is symmetric and positive semi-definite, it can be represented as a sum of a convergent sequence of product functions, i.e., there exists a feature mapping ϕ such that $K(x, x') = \phi(x) \cdot \phi(x')$. This condition is the most critical mathematical requirement for a function to be a valid kernel.

3.) Avoiding Data Snooping
If we don't care whether the kernel function is valid or not, then this becomes a case of data snooping, as the kernel is tailored to the specific dataset (e.g., by inspecting the test data to choose a kernel that performs well), which can lead to overfitting and poor generalization. Instead, the kernel should be chosen based on prior knowledge, cross-validation on training data, or theoretical considerations, ensuring that the model remains unbiased with respect to unseen data.