Assignment 2
Foundations of Machine Learning

Name: Ahmik Virani
Roll Number: ES22BTECH11001

Contents

1 Question 1

- We begin by explaining our choice of 1 as the constant. When we choose 1, the distance between the margins becomes $\frac{2}{\|w\|}$, leading to the optimization problem of minimizing $\frac{\|w\|^2}{2}$. The factor of 2 is beneficial for differentiation, allowing us to eliminate unnecessary constants when working with partial derivatives in the Duality Problem and using Lagrange Multipliers.

  It is important to note that multiplying the equation by a constant does not affect our results, as we are dealing with the **functional margin**.

  Let's define the functional margin as:

  $$\text{Functional Margin} = y_i \cdot (w^T x_i + b).$$

  The **geometric margin** is defined as:

  $$\text{Geometric Margin} = \frac{y_i \cdot (w^T x_i + b)}{\|w\|}.$$

  This means that, regardless of the scale of our functional margin, the geometric margin remains constant.

  We can express the geometric margin as:

  $$\text{Geometric Margin} = \frac{\text{Functional Margin}}{\|w\|}.$$

  Instead of 1, we can choose γ. Although this may appear to change the margin size, we are changing the functional margin. The optimization problem still remains the same i.e. on geometric margin, which remains unaffected.

  If we define the margin boundaries as $w^T x_i + b = \gamma$ and $w^T x_i + b = -\gamma$, we can divide both sides of these equations by γ, resulting in:

  $$\frac{w^T x_i + b}{\gamma} = 1 \quad \text{and} \quad \frac{w^T x_i + b}{\gamma} = -1.$$

  This shows that the solution is simply rescaled by a factor of γ, this rescaling does not affect the geometric margin, which is still the same:

  $$\text{Geometric Margin} = \frac{\text{Functional Margin}}{\|w\|}.$$

  By subtracting the two equations, we obtain:

  $$w_{\text{new}}^T (x_1 - x_2) = 2\gamma.$$

  Thus, the distance between the two margins is given by $\frac{2\gamma}{\|w_{\text{new}}\|}$.

  The optimization problem then transforms into minimizing $\frac{\|w_{\text{new}}\|^2}{2\gamma^2}$. However, we know that $\|w_{\text{new}}\| = \gamma\|w\|$.

In other words, after dividing by γ, both w and b are scaled accordingly, but the optimization problem still minimizes $\frac{\|w\|^2}{2}$, ensuring that the same maximum margin hyperplane is found, with no change in the final result.

In summary, while the functional margin may change due to scaling, the optimization process relies on the geometric margin, which remains invariant. Thus, the final solution for the maximum margin hyperplane remains unchanged.

2 Question 2

- Consider the half-margin of the maximum-margin SVM defined by ρ, i.e. $\rho = \frac{1}{\|w\|}$. We want to show that

  $$\frac{1}{\rho^2} = \sum_{i=1}^{N} \alpha_i$$

  where α_i are the Lagrange multipliers given by the SVM dual.

  We start by considering the optimization problem for the maximum-margin SVM, which involves minimizing $\frac{\|w\|^2}{2}$ subject to the constraints $y_i(w^T x_i + b) \geq 1$. Using the method of Lagrange multipliers, we can form the Lagrangian:

  $$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i \left(y_i(w^T x_i + b) - 1\right)$$

  where $\alpha_i \geq 0$ are the Lagrange multipliers corresponding to each constraint.

  Taking the partial derivative of the Lagrangian with respect to w and equating it to zero gives:

  $$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{N} \alpha_i y_i x_i = 0$$

  Thus, we have:

  $$w = \sum_{i=1}^{N} \alpha_i y_i x_i$$

  Now, we compute $\|w\|^2$ by taking the dot product of w with itself:

  $$\|w\|^2 = w^T w = \left(\sum_{i=1}^{N} \alpha_i y_i x_i\right)^T \left(\sum_{j=1}^{N} \alpha_j y_j x_j\right)$$

  Expanding this, we get:

  $$\|w\|^2 = \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (x_i^T x_j)$$

  This expression is used in the dual formulation of SVM optimization.

  Now we know that the second term of the Lagrangian must be zero to satisfy the KKT conditions. Hence, we have:

  $$\sum_{i=1}^{N} \alpha_i \left(y_i(w^T x_i + b) - 1\right) = 0$$

Expanding this, we get:

$$\sum_{i=1}^{N} \alpha_i \left(y_i \left(\left(\sum_{j=1}^{N} \alpha_j y_j x_j \right)^T x_i + b \right) - 1 \right) = 0$$

This simplifies to:

$$\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (x_i^T x_j) - b \sum_{i=1}^{N} \alpha_i y_i + \sum_{i=1}^{N} \alpha_i = 0$$

Now, since $\sum_{i=1}^{N} \alpha_i y_i = 0$ holds, we can further simplify this to:

$$\sum_{i=1}^{N} \alpha_i = \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (x_i^T x_j)$$

Substituting back into our expression for the geometric margin, we can express it as:

$$\sum_{i=1}^{N} \alpha_i = \|w\|^2$$

Thus, substituting $\|w\|^2$ into the definition of ρ:

$$\rho^2 = \frac{1}{\sum_{i=1}^{N} \alpha_i}$$

Finally, we arrive at the desired result:

$$\frac{1}{\rho^2} = \sum_{i=1}^{N} \alpha_i$$

This concludes the derivation, demonstrating the relationship between the half-margin ρ and the Lagrange multipliers α_i.

3 Question 3

3.1 Part (a)

$k(x,z) = k_1(x,z) + k_2(x,z)$ Let $k_1(x,z) = \phi_1(x) \cdot \phi_1(z)$ and $k_2(x,z) = \phi_2(x) \cdot \phi_2(z)$. Now, we can write $k_1(x,z) + k_2(x,z)$ as follows:

$$\begin{bmatrix} \phi_1(x) & \phi_2(x) \end{bmatrix} \cdot \begin{bmatrix} \phi_1(z) \\ \phi_2(z) \end{bmatrix} = \phi_1(x) \cdot \phi_1(z) + \phi_2(x) \cdot \phi_2(z)$$

Because expanding will just give the original equation i.e. $\phi(x) \cdot \phi(z)$. Hence, this is a valid kernel function because it can be written as a dot product of feature mappings, and the sum of two valid kernels is also a valid kernel.

3.2 Part (b)

$k(x,z) = k_1(x,z)k_2(x,z)$ Now, let's prove the two necessary properties:

(i) Symmetry: We need to show that $k(x,z) = k(z,x)$.

$$k(x,z) = k_1(x,z)k_2(x,z)$$

$$k(z,x) = k_1(z,x)k_2(z,x)$$

Since k_1 and k_2 are valid kernel functions, they are symmetric, i.e., $k_1(x,z) = k_1(z,x)$ and $k_2(x,z) = k_2(z,x)$. Thus,

$$k(z,x) = k_1(x,z)k_2(x,z) = k(x,z)$$

Hence, $k(x,z)$ is symmetric.

(ii) Positive Semidefiniteness: Since k_1 and k_2 are positive semidefinite, for any function $f(x)$ where $\int f(x)^2 dx$ is finite, the following inequality holds:

$$\int\int f(x)k_1(x,y)f(y)\,dx\,dy \geq 0$$

and

$$\int\int f(x)k_2(x,y)f(y)\,dx\,dy \geq 0$$

For the product $k(x,z) = k_1(x,z)k_2(x,z)$, the positive semidefiniteness is also guaranteed because the product of two positive semidefinite kernels is positive semidefinite.
Hence, $k(x,z) = k_1(x,z)k_2(x,z)$ is a valid kernel function.

3.3 Part (c)

- $k(x,z) = h(k_1(x,z))$, where h is a polynomial function with positive coefficients.

  Let $h(x)$ be of the form:

$$h(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

Therefore,

$$h(k_1(x,z)) = a_0 + a_1 k_1(x,z) + a_2 k_1(x,z)^2 + \cdots + a_n k_1(x,z)^n$$

This expression can be viewed as the sum of products of kernel functions. Let us first solve the problem for a simpler case:
If $k_1(x,z) = \phi(x) \cdot \phi(z)$ is a valid kernel, then so is $k(x,z) = c \cdot k_1(x,z)$ for any constant $c > 0$.

We can express this as:

$$k(x,z) = \sqrt{c}\,\phi(x) \cdot \sqrt{c}\,\phi(z)$$

This shows that multiplying a valid kernel by a positive constant still results in a valid kernel.

Now, considering $h(k_1(x,z))$, it is just a sum of powers of $k_1(x,z)$ multiplied by positive coefficients a_i. Each term of the form $a_i k_1(x,z)^i$ can be interpreted as a kernel because:

- $k_1(x,z)$ is a valid kernel.
- The product of a valid kernel and a positive constant is also a valid kernel.
- The sum of valid kernels is a valid kernel (as proven in part (a)).
- The product of valid kernels is a valid kernel (as proven in part (b)).

Hence, $h(k_1(x,z))$ is a sum of products of kernel functions, each multiplied by positive constants. Since all terms are valid kernels, the result is a valid kernel function.

Thus, $k(x,z) = h(k_1(x,z))$ is a valid kernel.

3.4 Part (d)

- This is similar to part (c).

  The Taylor series expansion of $\exp(x)$ is:

  $$\exp(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots$$

  Thus, the expansion of $\exp(k_1(x,z))$ is:

  $$\exp(k_1(x,z)) = 1 + k_1(x,z) + \frac{k_1(x,z)^2}{2!} + \frac{k_1(x,z)^3}{3!} + \ldots$$

  This expression is a sum of powers of $k_1(x,z)$, where each term is multiplied by positive coefficients $\frac{1}{n!}$, which are constants. Since:

  - $k_1(x,z)$ is a valid kernel,
  - Powers of a valid kernel are also valid (as proven in part (c)),
  - The sum of valid kernels is a valid kernel,

  we can conclude that $\exp(k_1(x,z))$ is a valid kernel.

  Thus, the function $k(x,z) = \exp(k_1(x,z))$ is just a sum of products of valid kernels, each multiplied by positive constants. Hence, $k(x,z)$ is a valid kernel function.

- We can say that the exponential function is a polynomial function of infinite dimension.

3.5 Part (e)

- Given the kernel function:

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

We can expand the norm $\|x - z\|^2$ as:

$$\|x - z\|^2 = \|x\|^2 + \|z\|^2 - 2x^T z$$

Substituting this into the kernel expression, we get:

$$k(x, z) = \exp\left(-\frac{\|x\|^2 + \|z\|^2 - 2x^T z}{\sigma^2}\right)$$

This can be split into two parts:

$$k(x, z) = \exp\left(-\frac{\|x\|^2}{\sigma^2}\right) \cdot \exp\left(-\frac{\|z\|^2}{\sigma^2}\right) \cdot \exp\left(\frac{2x^T z}{\sigma^2}\right)$$

The first part:

$$\exp\left(-\frac{\|x\|^2}{\sigma^2}\right) \cdot \exp\left(-\frac{\|z\|^2}{\sigma^2}\right) = \exp\left(-\frac{\|x\|^2 + \|z\|^2}{\sigma^2}\right)$$

is a constant.

The second part, $\exp\left(\frac{2x^T z}{\sigma^2}\right)$, involves $x^T z$, which is a valid kernel function. Multiplying it by a positive constant $\frac{2}{\sigma^2}$ is also valid, as was proved earlier.

Taking the exponent of $x^T z$ is valid based on the results from part (d).

Thus, combining the two parts involves multiplying by some positive value (since e^x, where x is any real number, is always positive). Therefore, the entire expression is a valid kernel function.

4 Report for Q4

4.1 Part (a)

- **Performance Metrics:**
  - **Accuracy:** 0.9788
  - **Number of Support Vectors:** 28
  - **Support Vectors on Each Side:** [14, 14]
- **Model Evaluation:**

  The testing accuracy of **97.88%** indicates that the model generalizes well to unseen data. Also, a low number of support vectors are sufficient to caputure the decision boundary, this means that the model only seems few data points to define the boundary, which means that the model is well suited for linear seperation. The model's high accuracy and minimal reliance on support vectors demonstrate that the linear SVM is an effective classifier for this dataset.

4.2 Part (b)

- **Performance Metrics:**

Dataset Size	Accuracy	Number of Support Vectors
50	0.9811	2
100	0.9811	4
200	0.9811	8
800	0.9811	14

Table 1: Accuracy and support vectors for different dataset sizes

- **Analysis:**

  The accuracy for datasets of sizes 50, 100, 200, and 800 remains constant at **98.11%**.

  The number of support vectors increases as the dataset size increases, which means the the data is linearly seperable and even with small number of training points the SVM can effectively make a decision boundary.

  As the training set size increases, the number of support vectors naturally rises due to exposure to more data points near the decision boundary. Although the decision boundary does not change significantly (as the similar accuracy), the SVM requires more support vectors as the training set grows to maintain a well-defined boundary.

4.3 Part (c)

- **Evaluation for $Q = 2$:**
  - We evaluate the performance of the SVM model with varying C values. The results are summarized in Table 2.

    The testing error remains constant across different C values, indicating that the model generalizes well despite changes in C. Also, the training error decreases as

C Value	Testing Error	Training Error	Number of Support Vectors
0.0001	0.01650943396226412	0.008968609865470878	236
0.001	0.01650943396226412	0.004484304932735439	76
0.01	0.018867924528301883	0.004484304932735439	34
1	0.018867924528301883	0.0032030749519538215	24

Table 2: Effect of varying C values on errors and support vectors (Q = 2)

C increases, which meand that the model fits the training data well and stable testing error means no overfitting.

- **Evaluation for $Q = 5$:**
  - We evaluate the performance with $Q = 5$, and the results are presented in Table 3.

C Value	Testing Error	Training Error	Number of Support Vectors
0.0001	0.018867924528301883	0.004484304932735439	26
0.001	0.021226415094339646	0.004484304932735439	25
0.01	0.021226415094339646	0.0038436899423446302	23
1	0.021226415094339646	0.0032030749519538215	21

Table 3: Effect of varying C values on errors and support vectors (Q = 5)

Similar to the previous analysis, increasing C decreases the training error. The number of support vectors also decreases, reflecting that the model is becoming more selective in defining the decision boundary.

- **Conclusion:**
  Both analyses demonstrate that the model maintains a stable testing error across varying C values, indicating a good balance between model complexity and generalization ability.

Answering from the above data (i) False (ii) True (iii) False (iv) False

4.4 Part (d)

- **Performance Metrics:**

C Value	Testing Error	Training Error	Number of Support Vectors
0.01	0.0236	0.00384	406
1	0.0212	0.00448	31
100	0.0189	0.00320	22
10000	0.0236	0.00256	19
1000000	0.0236	0.00064	17

Table 4: Analysis of testing and training errors with high C values

- **Analysis:**
  The best performance occurs around $C = 100$, the model seems to generalize well without becoming overly complex.

As C increases to high values, training errors decrease. And, as testing error is not too high, model is not overfitting. The trend of decreasing support vectors with increasing C suggests the model is becoming more selective about which points define the decision boundary.

5 Report for Q5

5.1 Part (a)

- **Results:**
  - Training error: 0.0
  - Testing error: 0.024000000000000002
  - Number of support vectors: 1084
- **Analysis:**

  The training error of 0 indicates that the data is well-separated by a linear hyperplane which means that there are no misclassifications, leading to a loss function of 0, suggesting the model is forming a hard margin.

  This situation is not indicative of overfitting because the testing error is also very low. If the testing error was high, it could mean overfitting.

  The model shows high accuracy, suggesting it generalizes well on unseen data. The high number of support vectors (1084) indicates that many points lie near the decision boundary.

5.2 Part (b)

- **Results for RBF Kernel:**
  - Training error: 0.0
  - Testing error: 0.5
  - Number of support vectors: 6000
- **Analysis:**

  The training error is 0, while the testing error is 0.5, indicating clear overfitting, the model is not generalizing well enough, making it comparable to just guessing a value. This results in a very weak learner, suggesting it may not be the best model to use.

  The large number of support vectors (6000) implies that the decision boundary is very complex, the model attempts to fit every detail of the training data, which can lead to overfitting.

Results for Polynomial Kernel:

- Training error: 0.0004999999999999449
- Testing error: 0.020000000000000018
- Number of support vectors: 1332

Analysis:

The training error is very low, and the testing error is also quite low at just 2%, This indicates that the polynomial kernel model is a much better choice compared to the RBF kernel. The number of support vectors (1332) is significantly lower than in the RBF case (6000), this means a more balanced model that is less complex, contributing to better generalization.

Conclusion:

- The polynomial kernel yields the lowest testing error.
- Although the other two algorithms provide better training accuracy, the polynomial kernel of degree 2 best represents the data among the tests and generalizes the data effectively.