**Support Vector Machine (SVM) - A Detailed Theory**

Support Vector Machine (SVM) is a supervised learning algorithm used primarily for classification and regression tasks. It works by finding the optimal hyperplane that separates data points belonging to different classes, with the maximum margin (distance) between them. The margin is defined by two planes: the **marginal planes**, which are parallel to the hyperplane but located on either side of it, and the **support vectors**, which are the data points closest to these planes.

**Core Concepts of SVM**

**1. Hyperplane and Margins**

In the case of linearly separable data, SVM tries to find the **optimal hyperplane** that maximizes the margin between two classes.

* **Hyperplane (decision boundary)**: A decision boundary that separates the feature space into two parts. In the case of two dimensions, this is a line; in three dimensions, it’s a plane, and in higher dimensions, it’s a hyperplane.
* **Margin**: The distance between the hyperplane and the nearest data points from either class, called the **support vectors**.

**2. Mathematical Formulation**

For a set of data points (xi,yi)(x\_i, y\_i), where xi∈Rnx\_i \in \mathbb{R}^n and yi∈{+1,−1}y\_i \in \{+1, -1\} (for binary classification), the goal of SVM is to find a hyperplane of the form:

wTx+b=0w^T x + b = 0

Where:

* ww is the normal vector to the hyperplane.
* bb is the bias term.

The **margin** MM between the two classes is maximized, which can be formulated as:

M=2∥w∥M = \frac{2}{\|w\|}

Thus, the goal of SVM is to **minimize** ∥w∥\|w\| subject to the constraint that all data points are correctly classified, i.e.,:

yi(wTxi+b)≥1for all iy\_i (w^T x\_i + b) \geq 1 \quad \text{for all } i

**3. Hinge Loss Function**

To handle non-linearly separable cases, SVM uses the **hinge loss function** to penalize misclassifications. The hinge loss for a single data point is defined as:

L(yi,f(xi))=max⁡(0,1−yi(wTxi+b))L(y\_i, f(x\_i)) = \max(0, 1 - y\_i (w^T x\_i + b))

Where:

* f(xi)=wTxi+bf(x\_i) = w^T x\_i + b is the decision function.
* yiy\_i is the true class label.

The hinge loss function is **zero** when the data point is correctly classified and at least **1 unit away** from the hyperplane, and increases linearly for points that violate the margin.

**4. Regularization Parameter (C)**

The parameter **C** in SVM controls the trade-off between achieving a large margin and minimizing the classification error (or misclassification).

* **High C**: A high value for CC gives a **high penalty** for misclassifying training points. This may lead to **overfitting** because the model tries to fit the training data too well, even at the cost of generalizing poorly to new data.
* **Low C**: A low value for CC gives a **low penalty** for misclassification, leading to a **larger margin** but potentially higher misclassification on the training data. This could lead to **underfitting**.

The objective of SVM can be written as a **regularized optimization problem**:

Minimize:12∥w∥2+C∑i=1Nmax⁡(0,1−yi(wTxi+b))\text{Minimize:} \quad \frac{1}{2} \|w\|^2 + C \sum\_{i=1}^{N} \max(0, 1 - y\_i (w^T x\_i + b))

Where the first term represents the margin maximization, and the second term represents the hinge loss function.

**5. Dual Formulation and Kernels**

SVM can also be formulated in its **dual** form, which leads to the **kernel trick**. In the dual form, the decision function depends on the inner product between the data points. For non-linearly separable data, SVM uses **kernels** to map the data into a higher-dimensional feature space where the data becomes linearly separable.

* **Linear Kernel**: ⟨xi,xj⟩\langle x\_i, x\_j \rangle
* **Polynomial Kernel**: (⟨xi,xj⟩+c)d(\langle x\_i, x\_j \rangle + c)^d
* **Gaussian (RBF) Kernel**: exp⁡(−∥xi−xj∥22σ2)\exp\left( - \frac{\|x\_i - x\_j\|^2}{2 \sigma^2} \right)

Using these kernels, SVM can effectively handle **nonlinear** decision boundaries.

**SVM in Action - Visualization**

Let’s now look at a few visual representations.

**1. Linearly Separable Case**

For linearly separable data, SVM finds the **best hyperplane** that maximizes the margin between two classes.

* **Margin**: The distance between the hyperplane and the closest support vectors.
* **Support Vectors**: The data points closest to the hyperplane, which define the margin.

Class 1 | Class 2

X | X

X | X

--------------------

Support vectors

**2. Non-linearly Separable Case (using kernel trick)**

In non-linearly separable cases, SVM maps the data into a higher-dimensional space where the data becomes separable. For example, using the **Gaussian kernel** can map points in 2D to a higher-dimensional space where they become linearly separable.

**Original data (2D space):**

Class 1 | Class 2

X | X

X | X

**After mapping using a kernel (higher-dimensional space):**

Mapping to higher dimension

X

X

X

----------------------------

Hyperplane

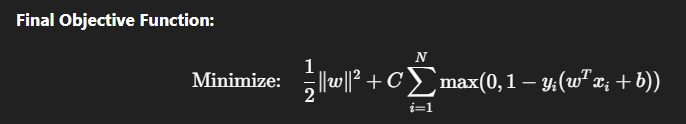
**Trade-off Between C and Margin**

A key point to note is the trade-off between the margin and classification error controlled by **C**.

| **C Value** | **Effect on Margin** | **Effect on Overfitting/Underfitting** |
| --- | --- | --- |
| **High C** | Smaller margin | Overfitting: Fits closely to training data |
| **Low C** | Larger margin | Underfitting: May have higher error on training data |

**Summary of SVM Objectives and Behavior**

* **Minimize** the **norm of the weight vector ww**, which is inversely related to the margin between classes.
* **Penalty term**: The hinge loss term penalizes misclassifications.
* **Parameter CC**: Controls the trade-off between the margin size and classification error (misclassifications).

****

This formulation provides a way to adjust the balance between fitting the training data and maintaining a large margin to improve generalization.

