Objectives

- Understand SVM margin and importance
- Learn SVM objective function
- Role of support vectors in SVM
- Lagrange multipliers in SVM
- Gradient descent optimization methods

---

1 Review

1.1 SVM Margins

In the context of Support Vector Machines (SVMs), the margin refers to the distance between the hyperplane and the closest data points from each class. These closest data points are called *support vectors*, and they are critical in determining the optimal decision boundary.

For a binary classification problem, the goal of an SVM is to find the hyperplane that maximizes this margin, ensuring the greatest possible separation between the two classes. A larger margin typically leads to better generalization and reduced risk of overfitting, as it implies a more confident separation of the classes.

1.2 Objective Function

Mathematically, the margin can be expressed as:

$$\text{Margin} = \frac{1}{\|\mathbf{w}\|}$$

where $\mathbf{w}$ is the weight vector normal to the hyperplane. The margin is inversely proportional to the magnitude of $\mathbf{w}$, meaning that to maximize the margin, we need to minimize $\|\mathbf{w}\|$. This is achieved by finding the hyperplane that satisfies the optimal separation between the two classes while keeping the margin as large as possible.

Note that while this is not the same as minimizing the vector $\mathbf{w}$, the *argmax* of $\frac{1}{\|\mathbf{w}\|}$ is the same as the *argmin* of $\|\mathbf{w}\|$. For instance, if you have:

$$\max(\frac{1}{10}, \frac{1}{20}, \frac{1}{30}), \quad \min(10, 20, 30)$$

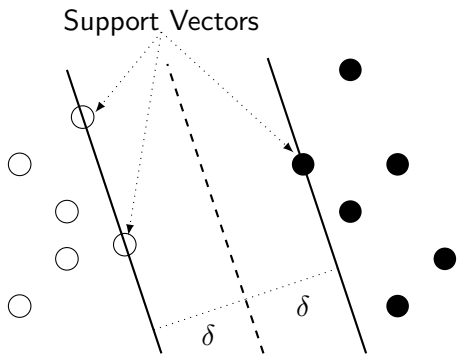

Figure 1: The dashed line is the optimal hyperplane with maximum margins on either side defined by the support vectors.

then the *argmax* of $\max(\frac{1}{10}, \frac{1}{20}, \frac{1}{30})$ is the index of the maximum value, and the *argmin* of $\min(10, 20, 30)$ is the index of the minimum value.

Therefore, the *argmax* and *argmin* refer to the positions where the maximum and minimum values occur, respectively.

2 Objective Function and Constraints

The objective function in SVM is to minimize the squared Euclidean norm of the weight vector $\mathbf{w}$:

$$\min \frac{1}{2}\|\mathbf{w}\|^2$$

The factor of $\frac{1}{2}$ is introduced for convenience, as it simplifies the derivative calculations. This is a differentiable function, which allows us to apply optimization techniques like gradient descent.

In SVM, the optimization problem is subject to the following constraint:

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant 1, \quad \forall i$$

where y_i is the class label (either 1 or -1), $\mathbf{x}_i$ is the feature vector, and b is the bias term.

3 Weighted Sum of Support Vectors

The support vectors are the points that lie closest to the margin boundaries. The weight vector $\mathbf{w}$ is a linear combination of these support vectors:

$$\mathbf{w} = c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \cdots + c_k\mathbf{x}_k$$

where $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k$ are the support vectors, and $c_1, c_2, \ldots, c_k$ are coefficients associated with each support vector.

Additionally, the constant coefficients $c_1, c_2, \ldots, c_k$ represent the relative importance of each support vector in the formation of the hyperplane. For example, if a support vector has a larger c_k, it has a

greater influence on the orientation of the hyperplane. On the other hand, say for instance that a support vector is an outlier, meaning it is far away from the decision boundary, then it follows that its corresponding c_k value may be small and have less influence on the orientation of the hyperplane.

4 Lagrange Multipliers and Optimization Constraints

To incorporate the constraints into the objective function, we use Lagrange multipliers.

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant 1$$

First, subtract 1 from both sides:

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 \geqslant 0$$

Next, introduce a Lagrange multiplier α_i for each data point x_i. The Lagrange multiplier enforces the constraint during the optimization process, ensuring that the constraint is met for the support vectors. Thus, we write the Lagrange term as:

$$\alpha_i \left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1\right) = 0$$

Now, we can write the objective function to be minimized. The original objective function in SVM is to minimize the squared Euclidean norm of the weight vector:

$$\min \frac{1}{2}\|\mathbf{w}\|^2$$

To incorporate the constraint using Lagrange multipliers, we combine the objective function with the Lagrange multiplier terms. This leads to the Lagrangian:

$$L = \min \frac{1}{2}\|\mathbf{w}\|^2 - \sum \alpha_i \left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1\right)$$

The key property of the Lagrange multipliers is that most of them turn out to be zero. This is because only the support vectors contribute to the weight vector w, making the corresponding multipliers for non-support vectors zero and minimizing their influence on the optimization process.

When discussing the constraint, it's important to recognize that the sign of the expression $y_i(\mathbf{w}^T\mathbf{x}_i + b)$ ensures that the data points are classified correctly. Specifically, if $(\mathbf{w}^T\mathbf{x}_i + b)$ is positive and the label y_i is also positive, the product will be positive, which indicates that the point is correctly classified on the positive side of the margin. Similarly, if $(\mathbf{w}^T\mathbf{x}_i + b)$ is negative and y_i is negative, the product will be positive, confirming the point is on the negative side of the margin. This relationship ensures that each data point is on the correct side of the margin boundary. The critical point here is that the margin must always be non-zero, i.e. there needs to be some "space" between the support vectors and the hyperplane to maintain proper separation between the two classes.

5 Partial Derivatives and Gradients

When tackling an optimization problem, the typical approach is to minimize a function. This involves computing the derivative of the objective function, setting it equal to zero, and solving for the variable (in this case, $\mathbf{w}$). However, in the presence of constraints, such as $y_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant 1$, we cannot directly apply this procedure.

To work with constraints, we adjust our approach. Instead of directly solving for $\mathbf{w}$ using just the objective function, we modify the problem to include both the objective function and the constraints. This results in a new equation where the constraint is "bundled" into the objective, allowing us to proceed with the minimization. The equation becomes:

$$L = \min \frac{1}{2}\|\mathbf{w}\|^2 - \sum \alpha_i \left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1\right)$$

The reason this approach works lies in the structure of the problem: the term $\frac{1}{2}\|\mathbf{w}\|^2$ is treated as the function $f(\mathbf{w})$, while the summation term acts as $g(\mathbf{w})$. This setup allows us to take the gradients of both functions, which boil down to:

$$\nabla f = \alpha \nabla g$$

This means the gradients of the objective function $f(\mathbf{w})$ and the constraint term $g(\mathbf{w})$ are parallel, and we can find the optimal solution by adjusting the weights based on these gradients.

In machine learning, the weight vector $\mathbf{w}$ can contain thousands or even billions of components. This is where multivariable calculus and gradient methods become particularly handy. For a function $f(x_1, x_2, \ldots, x_m)$, we can represent the gradient as a vector of the partial derivatives like so:

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_m} \end{bmatrix}$$

Here is an example to see how to calculate partial derivatives by differentiating with respect to one variable while treating the other variables as constants. This helps us understand how the function changes in each direction. For instance, for the simple 3D parabola $f(x, y) = x^2 + y^2$, the partial derivatives are computed by holding one variable constant at a time. Specifically, when taking the partial derivative with respect to x, we treat y as a constant, and when taking the partial derivative with respect to y, we treat x as a constant, then take the derivative as usual:

$$\frac{\partial f}{\partial x} = 2x, \quad \frac{\partial f}{\partial y} = 2y$$

Thus, the gradient vector is:

$$\nabla f = \begin{bmatrix} 2x \\ 2y \end{bmatrix}$$

The gradient points in the direction of the steepest ascent, and we use it in gradient descent to find the minimum by moving in the opposite direction.

6 Gradient Descent

Once we have calculated the gradient, we can use various gradient descent techniques to optimize the weights and biases. Some common types of gradient descent include:

- **Regular Gradient Descent:** Uses the full dataset to compute the gradient at each iteration.
- **Stochastic Gradient Descent (SGD):** Uses a single randomly selected data point for each iteration.
- **Mini-Batch Gradient Descent:** Uses a small batch of data points for each iteration.
- **Momentum:** Adds a fraction of the previous update to the current update to accelerate convergence.
- **AdaGrad:** Adapts the learning rate based on the past gradients.
- **RMSProp:** An adaptive learning rate algorithm that divides the gradient by an exponentially decaying average of squared gradients.
- **Adam (Adaptive Moments):** A combination of Momentum and RMSProp, incorporating both the first and second moments of the gradients.