Demystifying Support Vector Machines on Linearly Separable Data

Rishabh Iyer

March 19, 2024

1 Introduction

Support Vector Machines (SVMs) represent one of the most robust and accurate methods in the realm of supervised learning. Developed initially for binary classification, the scope of SVMs has expanded over the years to include applications in regression and outlier detection. At their core, SVMs seek to find the optimal separating hyperplane between classes in a feature space, maximizing the margin between the closest points of the classes, known as support vectors. This fundamental property allows SVMs to excel in a variety of tasks, offering significant advantages over simpler linear classifiers, such as perceptrons.

1.1 Solving the Limitations of Perceptrons

While perceptrons are instrumental in understanding the basics of linear classification, they fall short in several aspects, notably in their inability to select the most robust classifier among many that can separate the data and the lack of guarantees for non-separable data. SVMs address this by not just finding any separating hyperplane but the one that maximizes the margin between the classes, leading to models with better generalization on unseen data.

2 Deep Dive into SVMs: Linearly Separable Case

This section provides a comprehensive exploration of Support Vector Machines (SVMs) when applied to datasets that are linearly separable. We examine the process from initial data preparation to the final evaluation of the model.

2.1 Step 1: Data Preparation and Gathering

The first step involves collecting a dataset that consists of input features (x) and binary target labels (y), where $y \in \{-1, 1\}$. For SVMs, especially in the linearly separable scenario, it's crucial to ensure that the data is indeed linearly separable for the theoretical guarantees of SVM to hold. This part is very similar to perceptrons (**CITE article**).

2.2 Step 2: Model

In the linearly separable case, the SVM model seeks to find the optimal separating hyperplane that divides the data into two classes with the maximum margin. The model is represented as:

$$f(x) = \text{sign}(w^T x + b)$$

where w is the weight vector perpendicular to the hyperplane, and b is the bias term that offsets the hyperplane from the origin. Again, the linear model is similar to the perceptron case (**CITE article**).

2.3 Step 3: Loss Function or Formulation

The SVM formulation for the linearly separable case involves finding the w and b that maximize the margin between the two classes. This is achieved by minimizing the following objective function subject to certain constraints:

$$\begin{aligned} &\min_{w,b} \frac{1}{2}\|w\|^2 \\ &\text{subject to } y^{(i)}(w^T x^{(i)} + b) \geq 1 \text{ for all } i = 1, \cdots, M \end{aligned} \tag{1}$$

The formulation of Support Vector Machines (SVMs) is grounded in the concept of creating a model that is not only accurate on the training data but also generalizes well to unseen data.

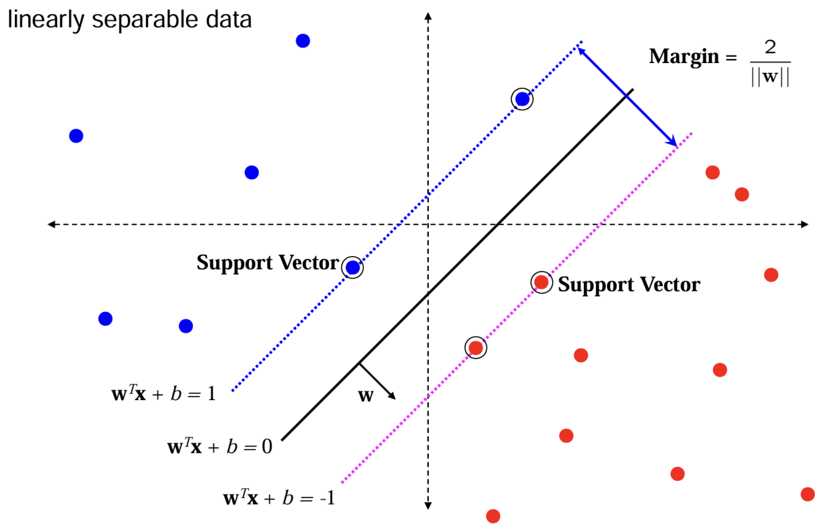
linearly separable data
Margin = $\frac{2}{||\mathbf{w}||}$
Support Vector
Support Vector
$\mathbf{w}$
$\mathbf{w}^T\mathbf{x} + b = 1$
$\mathbf{w}^T\mathbf{x} + b = 0$
$\mathbf{w}^T\mathbf{x} + b = -1$

Figure 1: Illustrating Support Vector Machine Formulation in Linearly Separable data.

- **Maximizing the Margin:** The core idea of SVMs is to find the hyperplane that separates the classes with the maximum margin. The margin is defined as the distance between the hyperplane and the nearest data point from either class. A larger margin implies a more confident classification, as it places the decision boundary far from the closest data points, reducing the model's sensitivity to small variations in the input data. The margin in the case of SVM is $2/\|w\|$ (as shown in Figure 1), and maximizing $2/\|w\|$ is essentially same as minimizing $\|w\|^2$ (the latter formulation makes the optimization problem a quadratic problem for which many solvers exist).
- **The Constraints:** The constraints in the SVM formulation, $y^{(i)}(w^T x^{(i)} + b) \geq 1$ for all i, ensure that all data points are correctly classified with a margin of at least 1. These constraints are critical for two reasons:
  1. **Correct Classification:** Each constraint enforces that data points are on the correct side of the decision boundary. For a data point $x^{(i)}$ with label $y^{(i)} = 1$, the constraint requires that $w^T x^{(i)} + b \geq 1$, placing it on the positive side of the hyperplane. Conversely, for $y^{(i)} = -1$, it ensures that $w^T x^{(i)} + b \leq -1$, placing it on the negative side. As we see in Figure 1, the blue points are are all above the $w^T x + b = 1$ line while the red points are all below the $w^T x + b = -1$ line.
  2. **Margin Bound:** The constraints also quantify the margin. By enforcing $y^{(i)}(w^T x^{(i)} + b) \geq 1$, the formulation ensures that the data points are not only on the correct side of the hyperplane but also at a distance that contributes to the margin. This distance is indirectly controlled by the norm of w, as the margin is inversely proportional to $\|w\|$. Thus, minimizing $\frac{1}{2}\|w\|^2$ maximizes the margin.
- **The Support Vectors:** Finally, the support vectors are the points closest to the decision boundary. As we see, the circled blue and red points are the support vectors in Figure 1. As we expect, the support vectors will lie on either the $w^T x + b = 1$ or $w^T x + b = -1$ lines.

2.4 Step 4: Optimization

The optimization problem at the heart of training a Support Vector Machine (SVM) for the linearly separable case is a quadratic programming (QP) problem. Quadratic programming refers to the process of minimizing (or maximizing) a quadratic objective function subject to linear constraints. In the context of SVMs, the objective function is quadratic due to the $\frac{1}{2}\|w\|^2$ term, and the classification constraints are linear with respect to the decision variables w and b.

Quadratic programming problems, such as the one formulated for SVMs, are typically solved using specialized numerical methods. The most common approaches include:

- **Interior Point Methods:** These methods iteratively approach the solution from within the feasible region defined by the constraints. They are highly effective for solving large-scale QP problems and are widely used in practice.
- **Sequential Minimal Optimization (SMO):** For SVMs specifically, the SMO algorithm breaks the large QP problem into a series of smallest possible QP problems. These smaller problems are then solved analytically, significantly reducing the computational complexity. SMO iterates over pairs of Lagrange multipliers, optimizing them while keeping the others fixed, and ensuring the constraints are satisfied.
- **Ellipsoid Algorithms:** Though less commonly used today for SVMs, ellipsoid algorithms iteratively refine an ellipsoid that contains the optimal solution, reducing its volume at each step until it converges to the optimal point.

2.5 Step 5: Evaluation

After training the SVM model, its performance is evaluated on a separate test dataset not seen during training. Recall that the model here are the learnt weights and bias terms. We denote the learnt model by (w, b). Now, given a test data instance x_{test}, the predicted label is simply:

$$y_{\text{pred}} = \text{Sign}(w^T x_{\text{test}} + b) \tag{2}$$

Given the predicted labels for every test data instance, we compare it to the true labels using the evaluation metrics. The evaluation metrics are very similar to the perceptron (**CITE Blog** so we just summarize it below.

- **Accuracy:** The proportion of correctly predicted instances in the test set.
- **Precision and Recall:** Especially important in imbalanced datasets, these metrics help understand the trade-offs between classifying positives correctly and avoiding false positives.
- **F1 Score:** The harmonic mean of precision and recall, providing a single metric to assess the balance between them.

Evaluation in the context of SVMs not only assesses how well the model generalizes to new data but also how effectively it leverages the maximum margin principle to ensure robustness and stability of the classification boundary.

3 Conclusion

In this blog article, we provide a formulation of support vector machines that solves a key limitation of the perceptron algorithm, i.e., its ability to differentiate between perfect classifiers, and its tendency to select models close to one class over the other. Thanks to the margin based formulation of SVM, i.e., its attempt to not just find the perfect separator between the classes, but amongst all perfect separators, pick the one that maximizes the margin, SVMs are much more robust compared to perceptrons in avoiding overfitting and dealing with noise in the training dataset.