SVM

David Rajchman, Matthew-Blake Foust

December 8, 2025

Abstract

1 Why SVMs

When given a dataset of two classes that can be binarily classified, we can most easily separate the classes by a hyperplane using a Perceptron and assuming the classes are linearly separable. But we must ask ourselves, what if our data isn't linearly separable? If it is, will we have one exact solution found in a reasonable time? What if we have more than two classes?

This is where the Soft-Margin Support Vector Machine(SVM) answers our machine learning prayers. The problem with the Perceptron algorithm is it will not converge if our data is not linearly separable. If our data is linearly separable we will end up with inconsistent solutions based on the hyperparameter of the learning rate and how our data is sorted. SVM's are able to combat these problems by using support vectors to find an optimal boundary, margin and unique solution that finds the smallest classification error. If the data is mostly linearly separable we can use a Soft-Margin to generalize our hyperplane.

2 The concept of the Margin

When we draw our decision boundary, we want the boundary to be drawn as far away from each class. We call this, maximizing our margin, where the distance from the first class is equally as far from the boundary as the second class. Our margin distance is defined as,

$$r = \frac{1}{\|w\|}, \quad \text{and we also define } \|w\| = 1.$$

Normalizing $\|w\|$ helps us derive this equation, $y_n(\langle w, x_n\rangle + b) \geq r$, where $y_n \in \{-1, 1\}$ determines if x_n is on the negative or positive side of the linear boundary. The closest point(s) that can be defined as part of a class will be the "Support Vector" that defines our margin.

However when we have outliers it becomes problematic because our decision boundary and margin will be redefined by this outlier from one class and the a point closest in the other class. If we can see how the graph is drawn with an outlier we will see that our margin is drawn incorrectly and that maybe the point that is an outlier should be removed. This is where the concepts of Hard vs Soft margins come into play.

3 Hard VS Soft

Sometimes data is not linearly separable, but we would still like to classify it with an SVM. We know that hard margin SVMs enforce a decision boundary that will not accept any margin errors when classifying data, where everything on one side of the boundary gets classified as $+1$ and everything on the other side gets classified as -1. We use the following optimization problem to define our hard margin:

$$\min_{w,b} \frac{1}{2}\|w\|^2 \quad \text{subject to} \quad y_n(\langle w, x_n\rangle + b) \geq 1, \quad n = 1, \ldots, N$$

On the other hand, soft margin SVMs are able to draw a decision boundary that separates two classes while maximizing the margin, even with some misclassified data points. To calculate our margin, we add a "slack" variable ξ_n, and introduce a regularization parameter $C > 0$, which trades off the size of the margin and the total amount of slack:

$$\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_{n=1}^{N} \xi_n \quad \text{Where } y_n(\langle w, x_n\rangle + b) \geq 1 - \xi_n, \quad \xi_n \geq 0$$

If the regularization parameter is large, then it is implied that there is low regularization as we give the slack variables a larger weight. A large C heavily penalizes any $\xi_n > 0$. Thus, when there is any heavy penalty our model will try to correctly classify outliers by shrinking the margin. From this equation, we can also see that w is regularized to prevent overfitting, but b is not so the boundary can shift without worrying about the penalties.

Here a large value of C implies low regularization, as we give the slack variables larger weight,hence giving more priority to examples that do not lie on the correct side of the margin. For every outlier, we add a penalty, and depending on how far the outlier is from the boundary, we increase the size of the penalty.

4 Hinge Loss

For a binary loss function we must count the number of mismatches between our predictions and the actual labels. So we must determine whether our x_n when used in our model, $f(x_n)$, gives us the incorrect label y_n.

We see that when using the hinge loss,

$$\ell(t) = \max\{0, 1 - t\}, \quad \text{where } t = yf(x) = y(\langle w, x\rangle + b)$$

we will return a zero when our $f(x_n)$ is correctly labeled and outside of the margin. When it is correctly labeled but inside of the margin area, It returns a t, $0 < t < 1$. If it is completely mislabeled then it returns a high loss value.

Using the hinge loss, our goal is to minimize the total loss while applying regularization to the margin, as described in Section 3.

$$\min_{w,b} \underbrace{\frac{1}{2}\|w\|^2}_{\text{regularizer}} + C\underbrace{\sum_{n=1}^{N} \max\{0, 1 - y_n(\langle w, x_n\rangle + b)\}}_{\text{error term}}$$

From this equation, we have our margin maximization and we have the error term.

4.1 How to add Citations and a References List

All equations of math are from Mathematics for Machine Learning [1].

References

[1] Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong. *Mathematics for Machine Learning*. Cambridge University Press, 2020.