Support Vector Machines From Scratch

Jonathan Huml

1 Introduction

This project implements a set of support vector machines from a pure nonlinear programming perspective, focusing on the underlying mathematics of the model instead of how to maximize results in a production setting. We implement and analyze six models: the hard-margin support vector machine and its dual problem, a compromised support vector machine, a combination of the compromised and hard-margin support vector machine, two types of kernel functions within the dual problem, and a quadratic surface support vector machine. While these sorts of problems are easily implemented in Scikit-Learn, for example, such simplicity tends to abstract away the underlying details. This paper will introduce intuitive motivations for the explicit nonlinear programs that will be introduced and computationally solved with data intended to show the weaknesses and strengths of certain models.

The support vector machine is a fairly simple program for the purpose of binary classification. We often start with the assumption that our data set is linearly separable, and attempt to separate our two classes with a hyperplane that maximizes the minimum distance between any given point and the supporting hyperplane (or line in $\mathbb{R}^2$, which is the case for most of our data in this paper). The larger the minimum distance, the more the model is thought to generalize to new instances of each class. Even for nonlinearly-separable classes, there may exist a hyperplane in higher dimensions that does separate the classes, which is the idea behind the kernel trick. For the quadratic surface model, we can completely discard the notion of linear separability in any dimension. Thus, the support vector machine is a highly fungible and diverse tool for solving binary classification problems.

Regardless of the data, each of the six models is a convex, quadratic programming problem, and all except for the last model have linear constraints. As such, we use the CVXPY and CVXOPT modules in Python, switching between each package as the latter makes more intuitive use of certain matrix forms, which is useful for the dual problem, as well as when we use the dual to implement the kernel trick. The simplicity of the support vector machine is seen here as well. For small to medium sized data sets, we examine the quick nature of computing an optimal solution.

2 Data

We analyze six total data sets. The first five exist in two dimensions, and the last exists in four dimensions. The sets able to be visualized without the aid of dimensionality reduction techniques are shown below. As we can see, only the first is truly, completely separable by a line. The rest are either imperfectly separable, or obviously nonlinear.

As stated in the introduction, the focus of this paper is mathematical, so very little preprocessing work is done on the data. Support vector machines are sensitive to the scale of the data, but these sets have been mostly normalized (i.e. the axes are of comparable numerical scale). They are also mostly balanced data sets; the classes are of similar sample size. We will abstain from any resampling or cross-validation for our models, for example. The main focus is whether or not the model visually captures the spirit of the data. As we can see below, some of the data sets are inherently nonlinear.

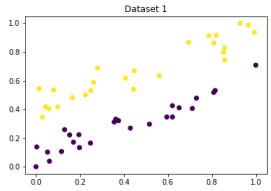

Figure 1: Scatterplot for Dataset 1

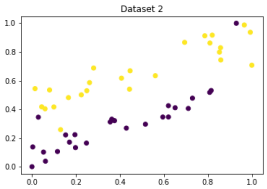

Figure 2: Scatterplot for Dataset 2

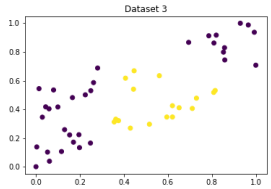

Figure 3: Scatterplot for Dataset 3

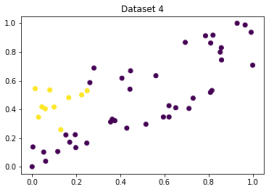

Figure 4: Scatterplot for Dataset 4

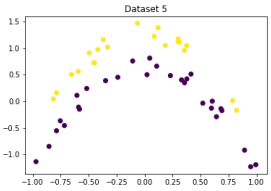

Figure 5: Scatterplot for Dataset 5

3 Approach

3.1 LSVM

The primal hard margin problem starts with the idea as stated in the introduction, to maximize the minimum distance between any given point and the margin. We need to find a weight vector $w \in \mathbb{R}^n$ such that the hyperplane $w^T x + b$ splits our data into two halves. The slope of this decision function is equal to the L_2 norm of w. Dividing this slope by 2 will place the points on either side of the line twice as far away from the boundary.

The 2-norm is convex on $\mathbb{R}^n$. We know by the Triangle Inequality that $||x + y|| \leq ||x|| + ||y||$, and that, for $k \in \mathbb{R}$, $||kx|| = |k| \cdot ||x||$. Thus we have:

$$||\theta x + (1 - \theta)y|| \leq ||\theta x|| + ||(1 - \theta)y|| = \theta||x|| + (1 - \theta)||y||$$

Unfortunately, L_2 is not everywhere differentiable, which complicates our approach. We do know that $||w|| \geq 0$ for all $w \in \mathbb{R}^n$. Now a composition of functions $f \circ g$ is convex if f non-decreasing and convex and g convex. The function $f(x) = x^2$ on $[0, \infty)$ is increasing and convex, so $f \circ L_2$ is convex, as well as differentiable. We thus have a simple convex programming problem.

The primal hard-margin classifier is given as the quadratic problem below, for $y_i \in \{-1, 1\}$:

$$\begin{aligned} &\text{minimize } \frac{||w||_2^2}{2} \\ &\text{subject to } y_i(w^T x_i + b) \geq 1 \text{ for each } i \in \{1, \ldots, N\} \\ &w \in \mathbb{R}^n, b \in \mathbb{R}. \end{aligned}$$

3.2 DLSVM

The dual is given by the Lagrangian of the primal problem. In the primal problem of §3.1, we have a constraint for each individual data point. For small data sets, this is often perfectly fine, but can become computationally unwieldy as N grows. We will not explicitly derive the dual here. Instead, we will give its matrix form. This is useful not only for the computational formatting in Python, but in §3.5, we will make only a slight modification to our dual problem given here, and get as free of a lunch as one can get.

Let $X = [x_1, \ldots, x_N] \in \mathbb{R}^{n \times N}$, $\mathbf{1}_N = [1, \ldots, 1]^T \in \mathbb{R}^N$, $y = [y_1, \ldots, y_N]^T$, $\alpha = [\alpha_1, \ldots, \alpha_N]^T$ and $H = \mathbf{Diag}(y)X^T X\mathbf{Diag}(y)$. Then the formulation is given by:

$$\begin{aligned} \text{minimize } & \frac{1}{2}\alpha^T H\alpha - \mathbf{1}_N^T\alpha \\ \text{subject to } & \alpha^T y = 0 \\ & \alpha \in \mathbb{R}_+^N. \end{aligned}$$

The matrix H can be verified as positive semidefinite, making this a very recognizable sort of quadratic program.

3.3 Compromised LSVM

In practice, it is often not possible to completely separate two classes by a hyperplane. Outliers and not-completely-linear boundaries can throw a wrench in our approach. It may also be of interest to make a more generalizable and flexible model even if the hard margin classifier gives acceptable preliminary results.

The following model gives a soft margin that allows points of either class to lie on the incorrect side of the hyperplane as a technique of overall robustness.

Let $t_i = \max\{0, 1 - y_i(w^T x_i + b)\}$

$$\begin{aligned} \text{minimize } & \sum_{i=1}^{N} t_i \\ \text{subject to } & 1 - y_i(w^T x_i + b) \leq t_i \text{ for each } i \in \{1, \ldots, N\} \\ & t_i \geq 0, w \in \mathbb{R}^n, b \in \mathbb{R} \end{aligned}$$

3.4 Linear Soft LSVM

The fourth model is a mere linear combination of §3.2 and §3.3 with a parameter C that regularizes the overall quadratic formulation. As $C \to 0$, $||w||$ will become smaller, and thus the hard margin will become wider as the soft margin decreases.

Given a parameter $C > 0$:

$$\begin{aligned} \text{minimize } & \frac{||w||_2^2}{2} + C\sum_{i=1}^{N} t_i \\ \text{subject to } & y_i(w^T x_i + b) \geq 1 - t_i \text{ for each } i \in \{1, \ldots, N\} \\ & w \in \mathbb{R}^n, b \in \mathbb{R}, t \in \mathbb{R}_+^N \end{aligned}$$

3.5 Kernalized SVM

The idea behind the kernel trick is that a data set may not be linearly separable when viewed as a subset of $\mathbb{R}^m$, but may be linearly separable embedded in higher dimensional space. We can define a nonlinear feature

map to this higher dimensional space. In this paper, we use the radial basis function (with parameter $\sigma > 0$) and the sigmoid kernel (with parameters $\beta > 0, \theta$) defined as:

$$\text{Radial basis function: } K(x_i, x_j) = \exp\left(\frac{-||x_i - x_j||^2}{2\sigma^2}\right)$$
$$\text{Sigmoid: } K(x_i, x_j) = \tanh\left(\beta x_j^T x_i + \theta\right)$$

The price we pay, besides computing the kernel functions, is that we also need to fine-tune the parameters in a generalizable way. There are many techniques to do this in a less sporadic fashion than the simple process we will use in this paper (we take a simple linear iteration over an interval on the real line), but this is beyond the scope of this paper. The limitation of implementing the kernel SVM as a quadratic program from scratch is that we do not immediately get all of the nice features of a package like Scikit-Learn.

However, in the quadratic solver, we can use the exact same formulation as §3.2, merely redefining H as $\mathbf{Diag}(y)\mathcal{K}\mathbf{Diag}(y)$ with $\mathcal{K}_{ij} = K(x_i, x_j)$. The matrix $\mathcal{K}$ is clearly symmetric as the inner product and norm are symmetric for $K(x_i, x_j)$. If we unpack the matrix formulation back to the Lagrangian dual, we have:

$$\sum_{i=1}^{m}\sum_{j=1}^{m} a_i a_j K(x_i, x_j)$$

Of course, $a_{i,j} > 0$ uniformly, so we can easily see that $\mathcal{K}$ is symmetric positive definite for the radial basis function or Gaussian kernel, which guarantees the quadratic nature of our problem. The sigmoid kernel is only conditionally positive semidefinite, yet still often performs well in practice. As can be seen in the code, we actually run into singular KKT matrix in the latter instance.

3.6 Quadratic Surface SVM

A data set is linearly separable if $w^T x_i + b > 0$ or $w^T x_i + b < 0$ for all $i \in \{1, \ldots, N\}$. A data set is quadratically separable if $\frac{1}{2}x_i^T W x_i + w^T x_i + c > 0$ or $\frac{1}{2}x_i^T W x_i + w^T x_i + c > 0$ for all $i \in \{1, \ldots, N\}$. This formulation does not need a kernel function, and thus no parameter tuning is needed. This saves quite a bit of time and effort on the user end. We find a matrix W and a vector $(b, c) \in \mathbb{R}^{m+1}$ such that:

$$\text{minimize } \sum_{i=1}^{n} ||Wx_i + b||_2^2$$
$$\text{subject to } y_i\left(\frac{1}{2}x_i^T W x_i + b^T x_i + c\right) \geq 1 \text{ for each } i \in \{1, \ldots, n\}$$
$$W = W^T \in \mathbb{R}^{m\times m}, (b, c) \in \mathbb{R}^m \times \mathbb{R}$$

The objective function can be shown to be of the quadratic form $\frac{1}{2}z^T G z$ [2]. There also exists an L_1 penalization form of the QSSVM that is equivalent to the hard margin SVM for finite penalization criteria ("large-enough") λ. Thus, we expect roughly the same results for this formulation as §3.1. Much like the hard margin SVM, other modifications of the quadratic surface SVM exist. However, this paper only explores the hard margin version of the quadratic form.

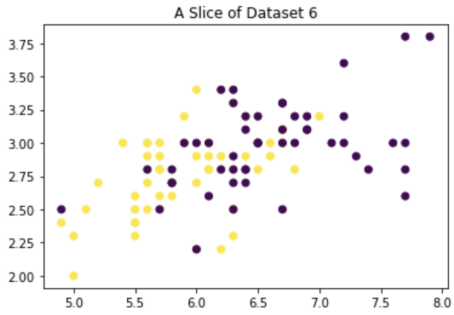

Figure 6: The Iris data set has overlapping classes

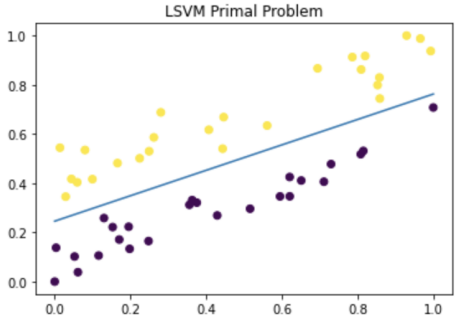

Figure 7: The primal problem is only feasible for the first (linearly separable) data set

4 Results

4.1 LSVM

The primal problem is not feasible for any of the data sets except for the very first one. This is because no planes exist that completely separate the other sets. In data set 2, a few points spill over beyond the upper half of a possible line. In data set 3, we would need two lines in $\mathbb{R}^2$ to separate the two classes. Data sets 4 and 5 would need a nonlinear boundary. While we cannot look directly at data set 6, we can look at a two-dimensional slice (see Fig. 6) to conclude that it is not linearly separable.

This is the major downside of a hard margin. The formulation is quite simple and intuitive, but not very practical with real data. For the first and only data set, we obtain $w = [-9.45\ 18.29]^T$ and $b = -4.49$. We can recover the line as a function of x_1 with $x_2 = -\frac{w_1^*}{w_2^*}x_1 - \frac{b^*}{w_2^*}$, which gives $x_2 = 0.52x_1 + 0.25$. We show the hard margin results in Fig. 7.

4.2 DLSVM

We recognized earlier that the primal problem is a linearly constrained convex quadratic programming problem. We also note that our set w exists on the convex set $\mathbb{R}^n$. By conjugate duality theory, we expect that its dual problem should exist without any duality gap. This is exactly what we observe in the code (although it is interesting to note that CVXPY immediately gives an infeasible solution, while CVXOPT iterates until the objective function is sufficiently large, on the order of 10^6, which we have to infer as infeasible). Thus we obtain exactly the same results as §4.1, with all infeasible except for the first data set. We obtain the same line as shown in Fig. 7.

4.3 Compromised LSVM

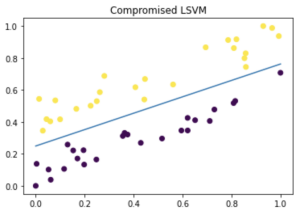

Figure 8: Compromised LSVM for data set 1

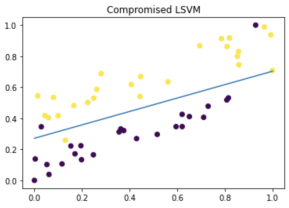

Figure 9: Compromised LSVM for data set 2

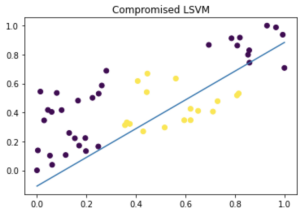

Figure 10: Compromised LSVM for data set 3

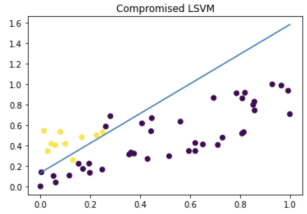

Figure 11: Compromised LSVM for data set 4

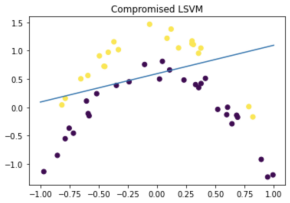

Figure 12: Compromised LSVM for data set 5

The compromised LSVM, where able to be visualized, performs impressively on our data sets, most notably for the first, second, and fourth sets. Here, we obtain accuracies of 100, 98, and 94 %, respectively (as far as accuracy is an appropriate statistic here: in the fourth set, we have a heavily imbalanced set of classes, so 94 % is not all that impressive alone. We can still see that the program seems to capture the overall trend, however). When we relax our margins, we not only gain more generalizable answers, we obtain an answer in the first place. By visual inspection, in data set 2, there should be no reason why two simple outliers should confound our search for a hyperplane when the sets are mostly separable.

We see that our infeasible results for the primal and dual problems are not reasonable with most data sets, especially in the real world where pure linear separability is usually not a valid assumption. The Iris data set performs well under the compromised LSVM model. Although we cannot visualize the results directly, we gain 98 % accuracy with the model, which is fairly impressive given the even split of the two classes. There exists definite predictive power with this model.

4.4 Linear Soft LSVM

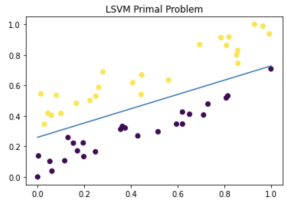

Figure 13: Linear Soft LSVM for data set 1

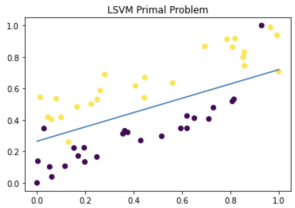

Figure 14: Linear Soft LSVM for data set 2

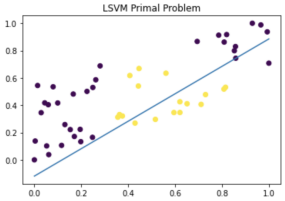

Figure 15: Linear Soft LSVM for data set 3

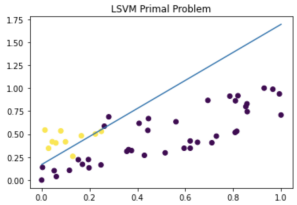

Figure 16: Linear Soft LSVM for data set 4

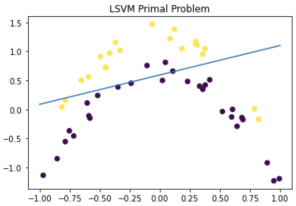

Figure 17: Linear Soft LSVM for data set 5

Given that the linear soft LSVM is a mere linear combination of the original primal problem and the soft margin problem, we should expect very little difference between the two problems. We even find that the objective function is quite insensitive to the parameter C, which in these figures is given as $C = 0.95$.

4.5 Kernalized SVM

4.5.1 Radial Basis Function/Gaussian Kernel

We find that the first and second data sets are very sensitive to small $\sigma > 0$. For $\sigma < 1$, the hyperplane becomes highly inaccurate, and pushes upwards. As $\sigma \to \infty$, the hyperplane roughly approaches that of $\S 4.3 - 4$, which is the correct positioning.

For the third data set, there is not much hope for linear separability in $\mathbb{R}^2$. The best we can do is to set $\sigma \to 0$ for $\sigma > 0$, which vertically halves the data.

Another interesting note for data set 4 is that for any $\sigma > 1$, the sign of our lines in $\S 4.3 - 4.4$ switches from positive to negative. However, this seems to be less accurate than a line for which $\frac{w_1^*}{w_2^*} < 0$, so we see little to no improvement with more work for tuning our parameter σ.

For $\sigma < 1$, the hyperplane also switches signs for data set 5. However, for $\sigma >> 1$, the lines approaches a flat line. Like data set 3, there is little hope for linear separability in $\mathbb{R}^2$.

The Iris data set behaves very much like the first and second data sets in terms of accuracy and sensitivity to σ. We therefore suspect that this data set is also mostly able to be separated linearly in $\mathbb{R}^4$.

4.5.2 Sigmoid Kernel

The sigmoid kernel presents many problems due to its conditional positive semidefinite properties we found in §3.5. In fact, for many combinations of β, θ, the quadratic solver is unable to solve for the objective because the rank of the matrix H is not full. Thus, we have to restrict our parameters on very small subsets of $\mathbb{R}$, which is quite difficult to do for so many data sets. Although this function is indeed possible to use, it may not be as practical as the Gaussian or a polynomial kernel of even degree.

4.6 Quadratic Surface

The quadratic surface SVM is only feasible for the first and last data sets. As stated in §3.6, this is the be expected as the hard margin SVM and quadratic surface SVM are equivalent for sufficiently large λ (where λ is the L_1 penalization criteria). Below in Figure [18], we show the results for the first data set; since the graph of the surface lives in $\mathbb{R}^3$, we show a level curve of the surface at $f(x_1, x_2) = 3.5$. At this point, the nonlinear surface approximates the line we found in §4.1.

The results presented here offer many further opportunities. Though the kernel happens in higher dimensional, nonlinear space, the separation given is still fundamentally linear and requires tuning. The quadratic surface gives nonlinear bounds, and we could easily change our model to accommodate a data set like data set 5. Though the hard margin problem is infeasible with this data set with the current formulation, we could add a slack variable to obtain a solution as the surface is *almost* quadratically separable. We note the same problem with data set 3: we needed at least two lines to separate the data in $\mathbb{R}^2$, which is nearly the same as finding some convex parabola on this particular problem.

We refer the reader to [1] for further reading on this topic.

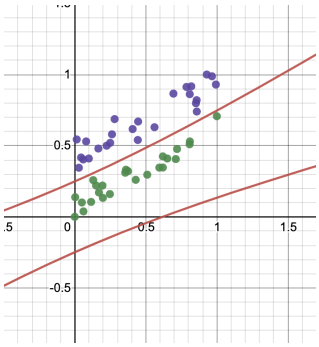

Figure 18: The quadratic line (shown here for a level curve at $f(x_1, x_2) = 3.5$) is nearly a flat hyperplane. We disregard the lower line as the level surface has two solutions on the quadratic problem.

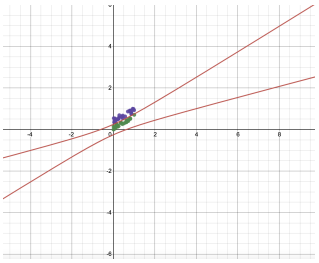

Figure 19: A nonlinear curve as we zoom out

5 Conclusions

The support vector machine is a light yet powerful tool in nonlinear programming and machine learning. Its mathematical basis is both intuitive and simple. Its convexity exists across several different forms, and thus will guarantee that our computational approaches will mostly work across several different types of data. The "No Free Lunch Theorem" states that all models will perform equally well across all types of problems, and we see this in action in this paper. We must let the data guide our model decisions, which is why the paper started with an exploration of how the data looked. Thankfully for support vector machine enthusiasts, there exist several modifications to the simple hard margin problem we started with that will work with nearly any data set. While the initial formulation of the problem is fundamentally linear, we saw that there even exists quadratic reformulations of the support vector machine. Though we solved for a hard-margin quadratic problem, we could easily accommodate a soft-margin reformulation as we did with

§3.1.

We saw the the initial hard-margin formulation is not a practical approach to real data in the face of outliers. However, the support vector machine performs quite well when we add slack variables. Even for highly nonlinear data sets like data set 5, we see predictive power (i.e. greater than 50% accuracy with a balanced distribution of classes). We also saw the difficulty and utility of the kernel trick. The sigmoid kernel can be useful, but is highly dependent upon the data. Mercer's condition states that we need $\mathcal{K}$ to be positive semidefinite, and this is violated for certain values of the parameters of the sigmoid function. Lastly, as shown by the quadratic surface SVM, this model does not have to be used for inherently linear data sets. It is flexible in its formulation, and still maintains convexity in the face of a nonlinear data set.

Most importantly, we see the ties between the long-standing field of optimization, and the more recent trends in machine learning. One field often implies the other, as most, if not all, machine learning models are nonlinear programs. The real difference is that we do not always attempt to find "the" optimal solution in machine learning, but the one that will generalize to the overall trend of the data. Regardless, it is odd to see how well continuous, convex programs seem to bifurcate real world data.

References

[1] Luo et al. "Soft Quadratic Surface Support Vector Machine for Binary Classification". In: *Asia-Pacific Journal of Operational Research* 33.6 (2016). DOI: 10.1142/S0217595916500469.

[2] Mousavi et al. "Quadratic Surface Support Vector Machine with L1 Norm Regularization". In: (2019). DOI: 10.3934/jimo.2021046.