Support Vector Machines (SVMs) are statistical machine learning algorithms used, among other applications, to as a binary classifier. Classification algorithms such as the SVM, logistic regression, classification trees, linear discriminate analysis, and neural networks, "learns" to distinguish between classes of points in $\mathbb{R}^n$ given historic training data.

0.1 Maximal Margin Separating Hyperplane

For motivation, consider the two classes of points in $\mathbb{R}^2$ in Figure **??**. Note that there are infinitely many lines that could be drawn to separate the red from the green points. Any such line can be described by the equation

$$f(x) = \beta^T x + \beta_0 = 0 \tag{1}$$

for some $\beta \in \mathbb{R}^2$, $\beta_0 \in \mathbb{R}$, and $x \in \mathbb{R}^2$. Simple geometric arguments show that for the green points x, $f(x) < 0$; likewise, for any red point $f(x) > 0$.

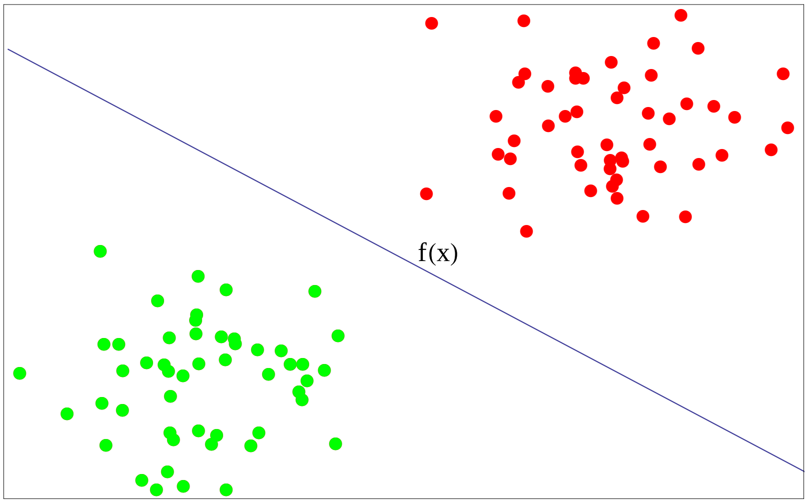

Figure 1: Linearly separable data

Moreover, for any two points x_1 and x_2 lying on the line L described by $f(x) = 0$,

$$\beta^T(x_1 - x_2) = -\beta_0 + \beta_0 = 0,$$

and thus $\beta^* = \beta / \|\beta\|$ is unit normal vector to L. The signed, normal distance from any point $x \in \mathbb{R}^2$ to L is $\frac{1}{\|\beta\|}(\beta^T x + \beta_0)$.

Suppose our feature set of N points is given by (x_i, y_i) where the class of $x_i \in \mathbb{R}^2$ is given by y_i such that $y_i = 1$ if x_i is red and $y_i = -1$ if x_i is green. This way, for any i, $y_i(\beta^T x_i + \beta_0)$ is the positive distance from x_i to L. We define the margin of the separating line to be the distance from L to the closest point in $\{x_i\}$.

The maximal margin separating hyperplane by solving the program:

$$\begin{array}{ll} \underset{\beta,\beta_0,\|\beta\|=1}{\text{maximize}} & M \\ \text{subject to} & y_i(\beta^T x_i + \beta_0) \geq M \\ & i = 1, ..., N. \end{array}$$

We can integrate the $\|\beta\| = 1$ constraint into inequality constraints as

$$\frac{1}{\|\beta\|} y_i(\beta^T x_i + \beta_0) \geq M$$

or

$$y_i(\beta^T x_i + \beta_0) \geq M \|\beta\| .$$

Note that for any β, β_0 satisfying the inequalities, any positive multiple also satisfies. Thus, we let $\|\beta\| = 1/M$ and give an equivalent program:

$$\begin{aligned} \underset{\beta,\beta_0}{\text{maximize}} \quad & \frac{1}{2}\|\beta\|^2 \\ \text{subject to} \quad & y_i(\beta^T x_i + \beta_0) \geq 1 \\ & i = 1, ..., N. \end{aligned} \tag{2}$$

Observe that the objective function in (**??**) is quadratic and the inequality constraints are nonlinear; thus, the program is convex and can be easily solved. The separating hyperplane program for arbitrary dimensional data can be naively solved in Mathematica:

```
SeparatingHyperplane[x_List, y_List] :=
 Module[
  {conditions, vars, b, w, ans,
   dim = Length[x[[1]]] (* Dimension of training data *),
   n = Length[x](* Size of training data *)},
  vars = Table[w[i], {i, 1, dim}];
  conditions = Join[
    {Norm[vars]^2} (* Cost function *),
    Table[y[[i]] (vars.x[[i]] + b) >= 1, {i, 1, n}]
    ];
  ans = Minimize[conditions, Join[vars, {b}]];
  ans = #[[2]] & /@ ans[[2]];
  {ans[[1;;-2]], ans[[-1]]}
  ]
```

The function **SeparatingHyperplane** returns $\{\beta_0, \beta\}$. The problem is more effectively solved with its Lagrangian dual, but we leave that development for the non-separable case below.

0.2 Non-Separable Data

Any hyperplane of parameters β ($\|\beta\| = 1$) and β_0 given by

$$\left\{x \mid f(x) = \beta^T x + \beta_0 = 0\right\}$$

induces a binary classification rule given by

$$G(x) = \operatorname{sign}\left[\beta^T x + \beta\right].$$

We now consider a set $\{(x_i, y_i)\}$ of N points where $x_i \in R^n$ where each x_i is of class $y_i \in \{-1, 1\}$. Unlike our previous example, this data may not be separable by a hyperplane. Nevertheless, we want to find a hyperplane that, in some sense, optimally classifies the data. The program given above cannot handle non-separable data, because misclassifications cause it to be unbounded.

To construct a new program, we permit misclassifications by introducing positive slack variables $\xi = (\xi_1, \xi_2, \ldots, \xi_N)$ and change our constraint to

$$y_i(\beta^T x_i + \beta_0) \geq M(1 - \xi_i).$$

Since we want to minimize the number of misclassifications, we penalize the cost function for misclassified test points. Our resulting program is

$$\begin{aligned} \underset{\beta,\beta_0}{\text{minimize}} \quad & \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i \\ \text{subject to} \quad & y_i(\beta^T x_i + \beta_0) \geq 1 - \xi_i \\ & \xi_i \geq 0 \\ & i = 1, ..., N \end{aligned}$$

for some penalizing constant C. This is again a convex problem.

0.2.1 Dual Formulation

We introduce Lagrange multipliers α_i, μ_1 (for $i = 1, \ldots, N$) to give the Lagrangian

$$\phi = \frac{1}{2} \left\| \beta^2 \right\| + C \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i \left[y_i(\beta^T x_i + \beta_0) - (1 - \xi_i) \right] - \sum_{i=1}^{N} \mu_i \xi_i .$$

We compute the Lagrange dual by setting the gradient with respect to the primal variables equal to zero:

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i \tag{3}$$

$$0 = \sum_{i=1}^{N} a_i y_i \tag{4}$$

$$\alpha_i = C - \mu_i, \; \forall\, i. \tag{5}$$

Substituting this into ϕ, we obtain the Lagrangian dual

$$\phi(\alpha, \mu) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j .$$

Equations (**??**) and (**??**) give constraints for minimizing the dual function. Since $\mu_i \geq$, we minimize the dual subject to (**??**) and $0 \leq a_i \leq C$. Because the primal problem is convex, the dual optimal value is also primal optimal.

At the global optimal solution, the KKT conditions guarantee the gradient of the Lagrangian being zero (equations (**??**)-(**??**)). Also, primal feasibility and complementary slackness:

$$y_i(x_i^T\beta + \beta_0) + (1 - \xi_i) \geq 0 \tag{6}$$

$$\alpha_i \left[y_i(x_i)^T\beta + \beta_0 - (1 - \xi_i) \right] = 0 \tag{7}$$

$$\mu_i \xi_i = 0, \tag{8}$$

for $i = 1, \ldots, N$.

Note that the dual problem only requires the training data in the form of an inner product. This inner product can be replaced by generalized (Mercer) kernel functions which transform the data into other spaces without significant computational difficulty. The dual formulation even gives a closed formed solution for transforming the data into certain infinite dimensional spaces. Further discussion of kernel methods is outside the scope of this project.

Given a optimal solution to the dual problem $\left\{\hat{\alpha}_i, \hat{\xi}_i\right\}$, the primal solution is

$$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i,$$

and β_0 is given by (**??**) for any i such that $\alpha_i = 0$.

0.3 Sequential Minimal Optimization Algorithm

In 1998 John Platt of Microsoft Research published a paper [**?**] describing his Sequential Minimal Optimization Algorithm for solving the dual problem. The SMO algorithm is a variation on the more general stochastic gradient descent algorithm and is very fast for solving the SVM.

Because of the linear dependence given by (**??**), the parameters α_i cannot be optimized independently as in stochastic gradient descent. Platt recognized that the linear dependence requires optimization of two parameters α_i and α_j together.

I have included a **Mathematica** implementation of Andrew Ng's Simplified SMO algorithm described in [**?**]. The simplified SMO does not guarantee under every possible training set. However, except in pathological cases, it provides a satisfactory SVM.

In Figure **??**, I used my algorithm to find the separating hyperplane for 25 red points sampled from $N((1,1),(1.5,1.5))$ and 25 green points from $N((-1,-1),(1.5,1.5))$. The data has significant overlap and is thus not linearly separable; 45 points are correctly classified, and 5 are misclassified.

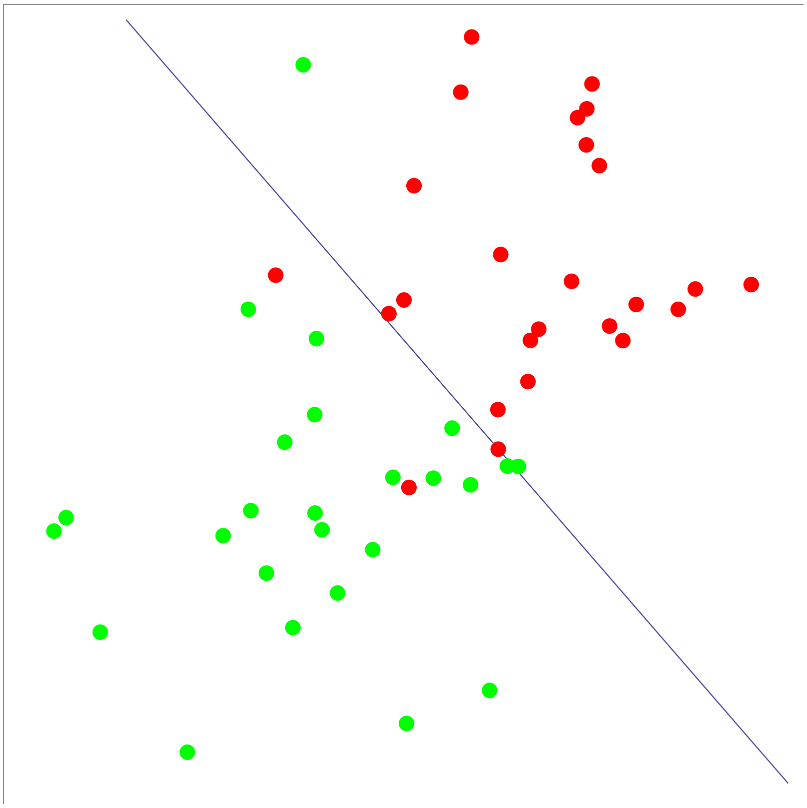

Figure 2: Non-separable data with linear separator

My code generalizes to data in arbitrary dimension and arbitrary Mercer kernels. In figure **??**, I used the polynomial kernel $K(x, x') = (1 + \langle x, x' \rangle)^2$ on the same data as **??**; 47 points are correctly classified, and 3 are misclassified. Of course, kernel methods are prone to overfitting.

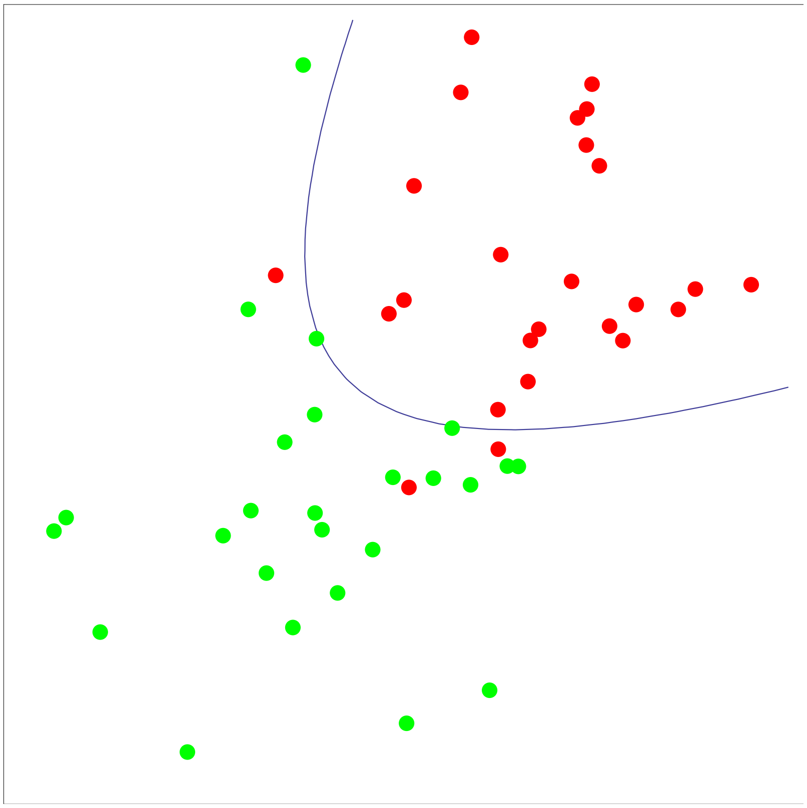

Figure 3: Non-separable data with quadratic kernel