CS 1810 Spring 2025 Section 5

Margin-Based Classification, SVMs

1 Motivation

In the past, with binary linear classifiers, we found a hyperplane that separated the data (or in cases when this was not possible separated a large amount of the data).

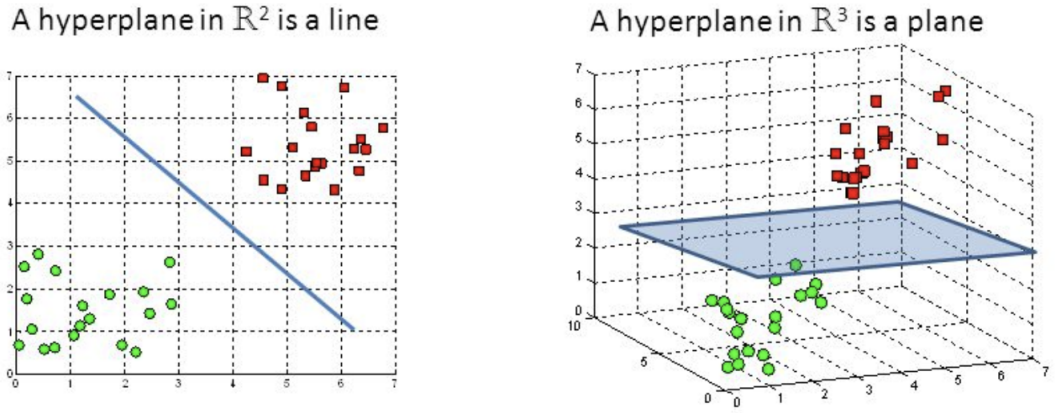
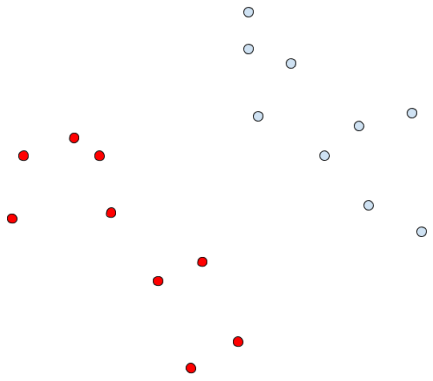


Figure 1: Source: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

However, if data is linearly separable, there may be many boundaries that work, so how can we know which is best?

Concept Question: Draw the "best" decision boundary for the data below (using your own definition of best). What would be a worse decision boundary? Why?



As your intuition told you above, the idea for Support Vector Machines is that, for all the linear hyperplanes that exist, we want one that will create the largest distance, or "margin", with the training data. At a high level, we define the margin as the minimum distance between a point and our boundary. Larger margins tend to improve generalization error.

Now we'll put this into math. To find a mathematical formula for the margin, we consider a hyperplane of the form

$$\mathbf{w}^\top \mathbf{x} + w_0 = 0.$$

The defining vector $\mathbf{w}$ is orthogonal to the hyperplane. To see this, first recall that the projection of some vector $\mathbf{a}$ onto the vector $\mathbf{b}$ is $\frac{\mathbf{a}\cdot\mathbf{b}}{\|\mathbf{b}\|}$. Then, if we consider the vector between two points $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(1)}$ on the hyperplane, the dot product with $\mathbf{w}$ gives

$$\mathbf{w}^\top(\mathbf{x}^{(1)} - \mathbf{x}^{(2)}) = \mathbf{w}^\top \mathbf{x}^{(1)} - \mathbf{w}^\top \mathbf{x}^{(2)} = -w_0 - (-w_0) = 0,$$

which means that the projection of $\mathbf{w}$ onto the plane must also be the 0 vector, or that $\mathbf{w}$ is orthogonal to the hyperplane.

So, the distance between the hyperplane and any point $\mathbf{x}$ will just be the length of the segment between the point and the hyperplane in the direction of $\mathbf{w}$. This gives us the perpendicular distance between $\mathbf{x}$ and the hyperplane. (*Note to think about:* Why do we want the perpendicular distance?)

Let r signify the distance between a point and the hyperplane. Let $\mathbf{x}_p$ denote the projection of the point p onto the hyperplane. Then, we can decompose the point $\mathbf{x}$ as

$$\mathbf{x}_p + r\frac{\mathbf{w}}{||\mathbf{w}||} = \mathbf{x},$$

or the sum of its projection $\mathbf{x}_p$ onto our hyperplane and the perpendicular vector from $\mathbf{x}_p$ to $\mathbf{x}$.

After left multiplying this decomposition by $\mathbf{w}^\top$, we get

$$\begin{aligned}\mathbf{w}^\top \mathbf{x}_p + r\frac{\mathbf{w}^\top \mathbf{w}}{||\mathbf{w}||} &= \mathbf{w}^\top \mathbf{x} \\ \Rightarrow r &= \frac{\mathbf{w}^\top \mathbf{x} + w_0}{||\mathbf{w}||}.\end{aligned}$$

So, our scalar r then gives the signed, normalized distance between a point and the hyperplane.

When considering this in the context of classifying datapoints $(\mathbf{x}^{(n)}, y^{(n)})$ with a hyperplane defined by $\mathbf{w}^\top \mathbf{x} + w_0 = 0$ in a SVM, we note that for correctly classified data, we have $y^{(n)} = +1$ when this distance is positive and $y^{(n)} = -1$ when it is negative. So, we can obtain a positive distance for both kinds of examples by multiplying by $y^{(n)}$ (which will not change the magnitude since $\|y^{(n)}\| = 1$). Then, we can define the margin of the dataset as the minimum such distance over all of our $\mathbf{x}^{(n)}$ in our dataset:

$$\min_n \frac{y^{(n)}(\mathbf{w}^\top \mathbf{x}^{(n)} + w_0)}{||\mathbf{w}||},$$

Concept Question: Before optimizing a model, it's important to make sure you understand how it works. Give a new data point $\mathbf{x}$, what would a SVM predict (assume you know w^*)?

We now want to figure out how to find the optimal $\mathbf{w}$. As we discussed in motivating this model, we want the $\mathbf{w}$ and w_0 that maximize the margin:

$$\arg\max_{\mathbf{w},w_0} \frac{1}{||\mathbf{w}||} \min_n y^{(n)}(\mathbf{w}^\top \mathbf{x}^{(n)} + w_0)$$

When discussing margin classifiers, we consider both hard and soft margin classifiers. In the hard-margin training problem, we know that the data is linearly separable and therefore any margin (including the optimal) must be greater than 0 (in the soft-margin problem, we do not make this assumption; this case is more complex so we deal with it later):

$$\min_n \frac{y^{(n)}(\mathbf{w}^\top \mathbf{x}^{(n)} + w_0)}{||\mathbf{w}||} > 0$$

We can observe that $\mathbf{w}$ and w_0 are invariant to changes of scale. Because of this, it is without loss of generality to impose $\min_n y^{(n)}(\mathbf{w}^\top \mathbf{x}^{(n)} + w_0) \geq 1$ (we prove this in the exercises of these notes). This lets us write the optimization problem as:

$$\arg\max_{\mathbf{w},w_0} \frac{1}{||\mathbf{w}||} \quad \text{s.t. } \forall n \; y^{(n)}(\mathbf{w}^\top \mathbf{x}^{(n)} + w_0) \geq 1$$

We can invert $\mathbf{w}$ to change the max to a min:

$$\arg\min_{\mathbf{w},w_0} \frac{1}{2}||\mathbf{w}||^2 \quad \text{s.t. } \forall n \; y^{(n)}(\mathbf{w}^\top \mathbf{x}^{(n)} + w_0) \geq 1$$

Informally, this is the same because the constraint is binding for the examples closest to the decision boundary. For these examples we have $y^{(n)}(\mathbf{w}^\top \mathbf{x}^{(n)} + w_0) = 1$. The distance on these examples is $1/||\mathbf{w}||$, and is maximized by minimizing $||\mathbf{w}||^2$. Mathematically, the min of $\frac{1}{||\mathbf{w}||}$ must be the max of $\frac{1}{2}||\mathbf{w}||^2$ since otherwise, we could just pick the proposed better maximum of $\mathbf{w}^*$ and find something eve smaller than our preexisting min.

So far, we've discussed SVMs without defining what a support vector actually is. A **support vector** is a data point on the margin boundary of the optimal solution. As such, we have that

- $y^{(n)}(\boldsymbol{w}^\top \boldsymbol{x}^{(n)} + w_0) = 1$
- The margin is exactly $\frac{1}{||\boldsymbol{w}||_2}$

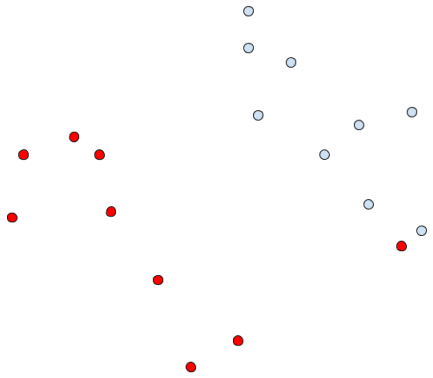
Note that we must have at least 2 support vectors, one for each class.

2 Soft Margin Formulation

For the hard margin formulation, we have been assuming that the data is linearly separable. However, this is not always true, and even if the data is linearly separable, it may not be best to find a separating hyperplane. In optimizing generalization error, there is a tradeoff between the size of the margin and the number of mistakes on the training data.

Concept Question: Let's illustrate the tradeoff between size of the margin and number of mistakes on training data. Assuming that you have sufficient basis transformations to draw any

boundary (since a boundary linear on the transformed data may be nonlinear on the original dimensions), draw what you feel would be the most generalizable SVM classifier for the below data?



For the soft margin formulation, we introduce a slack variable $\xi^{(n)} \geq 0$ for each i to relax the constraints on each example.

$$\xi^{(n)} \begin{cases} = 0 & \text{if correctly classified and not inside margin region} \\ \in (0, 1] & \text{if correctly classified but inside margin region} \\ > 1 & \text{if incorrectly classified} \end{cases}$$

We can now rewrite the training problem for a soft margin formulation to be

$$\arg\min_{\mathbf{w}, w_0} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{n=1}^{N} \xi^{(n)}$$

$$\text{s.t. } \forall i \; y^{(n)}(\mathbf{w}^\top \mathbf{x}^{(n)} + w_0) \geq 1 - \xi^{(n)}$$

$$\xi^{(n)} \geq 0$$

We add a regularization parameter C, that controls how much we penalize violating the hard margin constraints. A large C penalizes these violations and thus "respects" the data closely and has small regularization. A small C does not penalize the sum of slack variables as heavily, relaxing the constraint. This is increasing the regularization.

The *support vectors* in the soft margin formulation are again those data points that lie on the margin boundary, in addition to any points inside the margin region or any points that are misclassified.

3 Dual Form of SVM Training

Let's return to our training problem, from the first part. Our original hard-margin training problem, which looks for weights that maximize the margin on the training data is quite difficult. Recall the training problem below

$$\mathbf{w}^\star, w_0^\star = \arg\min_{\mathbf{w},w_0} \frac{1}{2}||\mathbf{w}||^2 \quad \text{s.t.} \quad y^{(n)}(\mathbf{w}^\top \mathbf{x}^{(n)} + w_0) \geq 1, \quad \forall n \in \{1,\ldots,N\}\,. \tag{1}$$

We introduce *Lagrange multipliers*, $\alpha_1,\ldots,\alpha_n \geq 0$, one for each inequality in Equation **??**, i.e., one per example, to obtain the Lagrangian function (Bishop appendix E does a good job of explaining Lagrange multipliers):

$$L(\mathbf{w}, w_0, \alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{n=1}^{N} \alpha^{(n)}(y^{(n)}(\mathbf{w}^\top \mathbf{x}^{(n)} + w_0) - 1) \tag{2}$$

Based on this, we now equivalently solve:

$$\mathbf{w}^*, w_0^* = \arg\min_{\mathbf{w},w_0} \max_{\boldsymbol{\alpha}\geq 0} L(\mathbf{w}, w_0, \alpha)$$

By strong duality (out of scope!), we can equivalently write this as:

$$\max_{\boldsymbol{\alpha}\geq 0} \min_{\mathbf{w},w_0} L(\mathbf{w}, w_0, \alpha)$$

This is useful because we can now solve analytically for the $\min_{\mathbf{w},w_0}[\cdot]$ part of this expression. Taking derivatives, we see:

$$\frac{\partial L}{\partial w_j} = w_j - \sum_{n=1}^{N} \alpha^{(n)} y^{(n)} \mathbf{x}^{(n)} = 0 \Rightarrow \mathbf{w}^* = \sum_{n=1}^{N} \alpha^{(n)} y^{(n)} \mathbf{x}^{(n)}$$

$$\frac{\partial L}{\partial w_0} = -\sum_{n=1}^{N} \alpha^{(n)} y^{(n)} = 0 \Rightarrow \sum_{n=1}^{N} \alpha^{(n)} y^{(n)} = 0$$

Plugging these optimal values into the Lagrangian function, we get (you should understand but not memorize this):

$$\begin{aligned}
L(\mathbf{w}, w_0, \alpha) &= \frac{1}{2}||\sum_{n=1}^{N} \alpha^{(n)} y^{(n)} \mathbf{x}^{(n)}||^2 - \sum_{n=1}^{N} \alpha^{(n)} y^{(n)} (\sum_{n'=1}^{N} \alpha^{(n')} y^{(n')} \mathbf{x}^{(n')})^\top \mathbf{x}^{(n)} - \sum_{n=1}^{N} \alpha^{(n)} y^{(n)} w_0 + \sum_{n=1}^{N} \alpha^{(n)} \\
&= \frac{1}{2} \sum_{n=1}^{N} \sum_{n'=1}^{N} \alpha^{(n)} \alpha^{(n')} y^{(n)} y^{(n')} {\mathbf{x}^{(n)}}^\top \mathbf{x}^{(n')} - \sum_{n=1}^{N} \sum_{n'=1}^{N} \alpha^{(n)} \alpha^{(n')} y^{(N)} y^{(n')} {\mathbf{x}^{(n)}}^\top \mathbf{x}^{(n')} \\
&\quad - w_0 \sum_{n=1}^{N} \alpha^{(n)} y^{(n)} + \sum_{n=1}^{N} \alpha^{(n)} \\
&= \sum_{n=1}^{N} \alpha^{(n)} - \frac{1}{2} \sum_{n=1}^{N} \sum_{i'=1}^{n} \alpha^{(n)} \alpha^{(n')} y^{(n)} y^{(n')} {\mathbf{x}^{(n)}}^\top \mathbf{x}^{(n')}
\end{aligned}$$

We then solve for α, maximizing L, subject to $\alpha \geq 0$. The support vectors are defined to be $Q = \{n : \alpha^{(n)} > 0\}$. For the soft-margin variation, these include examples that lie in the margin region in addition to on the margin boundary.
To classify a new example $\mathbf{x}$, we compute

$$\sum_{n=1}^{N} (\alpha^\star)^{(n)} \, y^{(n)} (\mathbf{x}^{(n)})^\top \mathbf{x} + w_0^\star$$

where $\alpha^\star$ and $w_0^\star$ solve the training problem. Based on this, we classify the example as $+1$ if this discriminant value is > 0, and -1 otherwise. The dual form is useful because the number of variables to maximize is linear in n (one for each training example), but there tends to be a small number of support vectors (data points that define the boundary) and thus the trained classifier can be interpretable and easier to calculate.

4 Kernel Trick

Additionally, the dual has the very nice property that if we use a basis function to map $\mathbf{x}$ to a higher dimensional space, this only comes in through the "kernel function." The training problem is as explained earlier, except where $(\mathbf{x}^{(n)})^\top \mathbf{x}^{(n')}$ appears, we use

$$K(\mathbf{x}^{(n)}, \mathbf{x}^{(n')}) = \phi(\mathbf{x}^{(n)})^\top \phi(\mathbf{x}^{(n')})$$

in its place. Similarly, we classify a new example $\mathbf{x}$ based on the value of discriminant

$$\sum_{n=1}^{N} \alpha^{\star(n)} \, y^{(n)} K(\mathbf{x}^{(n)}, \mathbf{x}) + w_0^*.$$

The reason that this is interesting is because we can directly compute the dot product $\phi(\mathbf{x})^\top \phi(\mathbf{x}')$ *without projecting to the higher-dimensional space!* This is known as the "kernel trick." As long as $K()$ is a valid kernel, the dual training problem can be solved without actually computing ϕ. Note that we can use this trick specifically because the dual gives us an equation for $\mathbf{w}^*$ in terms of $\phi(\mathbf{x}^{(n)})^T \phi(\mathbf{x}^{(n')})$.

5 Practice Problems

1. **Removing Support Vectors and Retraining (Berkeley, Fall '11)**

   Suppose that we train two SVMs, the first containing all of the training data and the second trained on a data set constructed by removing some of the support vectors from the first training set. How does the size of the optimal margin change between the first and second training data? What is a downside to doing this?

2. **Proof that margin is invariant to scalar multiplication**

   In reformulating our max margin

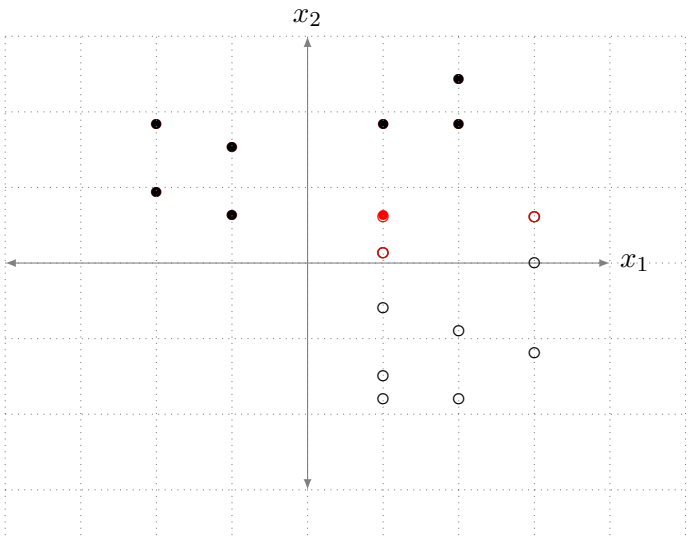
   $$\frac{y^{(n)}(\mathbf{w}^\top \mathbf{x}^{(n)} + w_0)}{||\mathbf{w}||}$$

   training problem, we use the fact that the margin is invariant to multiplying $(\mathbf{w}, w_0)$ by any $\beta > 0$. Show that this property is true.

3. **Draw Margin Boundary**

i) For the first example, you can assume the hard margin formulation. Draw the decision boundary as well as the two margin boundaries given that the red examples represent points with minimum margin.
ii) For the second example, you can assume the soft margin formulation and that all points are correctly classified with the optimal decision boundary. The decision boundary is already given. Draw the two margin boundaries given the information about the ξ_n slack values for the different examples shown. Assume the other points are outside of the margin region.

i)



ii)