Manually Calculating an SVM's Weight Vector

Jan 11, 2016

Note: This post assumes a level of familiarity with basic machine learning and support vector machine concepts.

Let's say that we have two sets of points, each corresponding to a different class. Let's call these classes *positive* and *negative* (but really they could be any binary class - *pink*, or *not pink*, *cucumber* or *not cucumber*, and so on).

For simplicity, I'm assuming two dimensions, but really the only requirement is that your data is separable by a hyperplane.

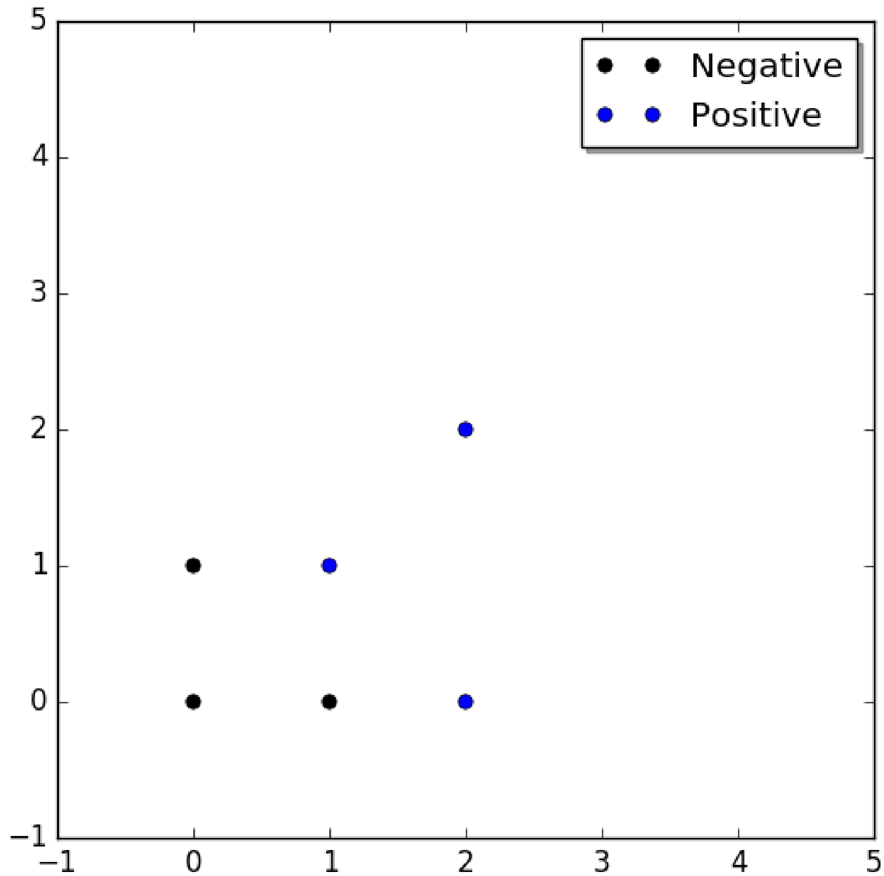
So, let's make up some input data:

$$positive = [(1,1),(2,2),(2,0)]$$

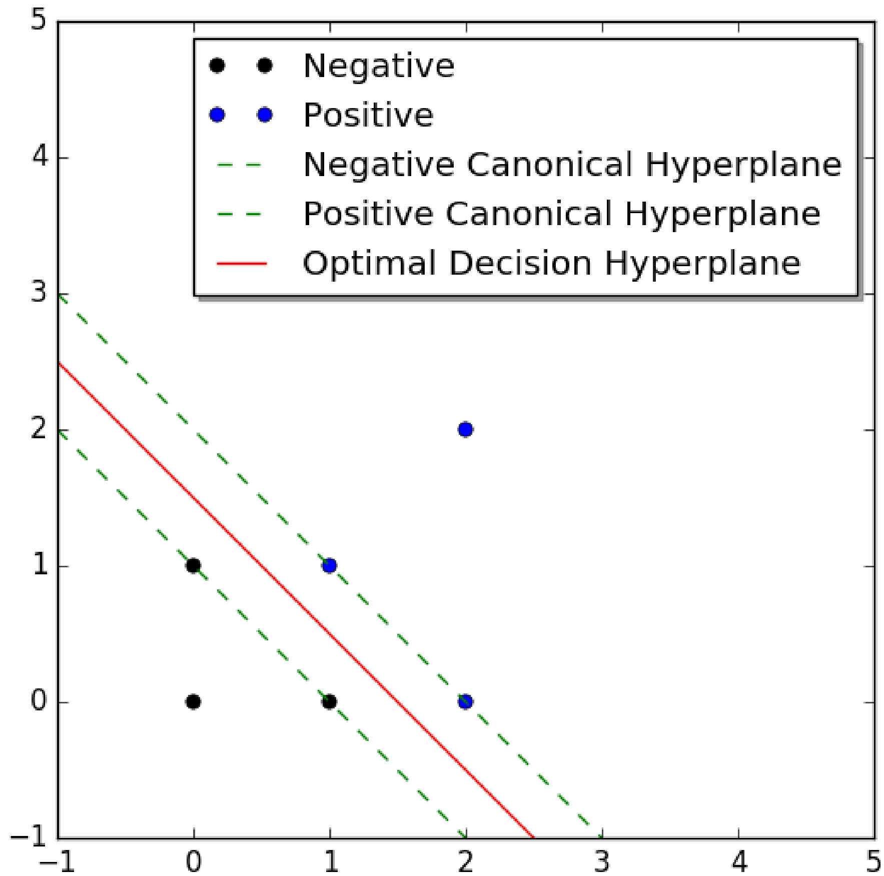
$$negative = [(0,0),(1,0),(0,1)]$$

Let's say that we want to train a support vector machine on these input classes and then determine the *weight vector* of the optimal decision hyperplane ("optimal decision hyperplane" is SVM-speak for "the line that separates the two input classes with the biggest distance between them" - this will become clear in a minute). The *weight vector* is simply a vector perpendicular to the optimal decision hyperplane.

Let's plot our points.

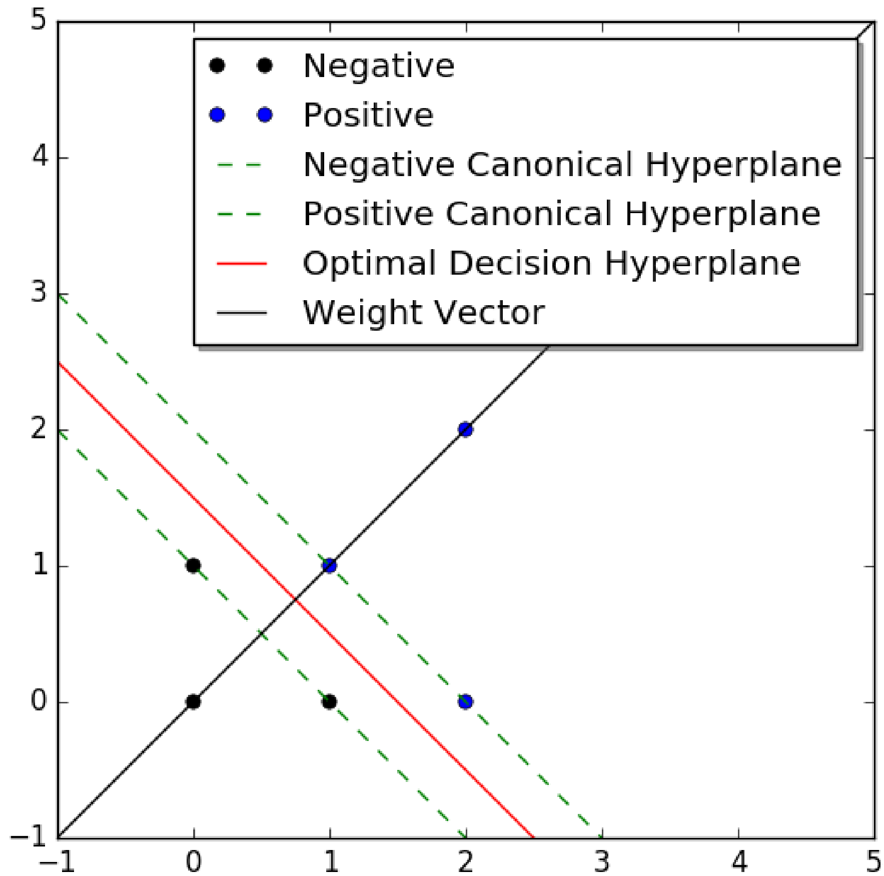


Are they linearly separable? (Yes, they are). Let's draw the positive and negative canonical hyperplanes, and then fit our optimal decision hyperplane.



Notice that the red line (the optimal decision hyperplane) *maximises* the perpendicular distance between the positive and negative classes.

So, what’s our weight vector? On this example, it’s actually pretty straightforward to determine by inspection (we’re simply looking for the normal vector to the optimal decision hyperplane), but that’s no fun. However, it’s good to know what result we’re after - so to give you an intuition, I’ve gone ahead and plotted it.



We're looking for the vector that passes through (0, 0) and (1, 1). So now we know what we're looking for, let's go ahead and figure it out.

We know that the equation of a line in an SVM follows these constraints;

$$w_1 x + w_2 y + w_3 = 0$$

Our weight vector is composed of *w1* and *w2*. If we examine our positive and negative canonical hyperplanes, we can see that there are two points that lie on each. These are the basis for our *support vectors* - the vectors that allow the SVM classifier to decide on the optimal decision hyperplane.

$$supportVectors_{positive} = [(1,1),(2,0)]$$

$$supportVectors_{negative} = [(1,0),(0,1)]$$

So, we can plug these points into our line equation, and solve linearly for *w1*, *w2*, and *w3*. Note that the *positive* examples are denoted with a sum of *1*, and the *negative* examples are denoted with a sum of *-1*. This is just SVM convention (it's arbitrary but it works).

$$pos_1 = 1w_1 + 1w_2 + w_3 = 1$$

$$pos_2 = 0w_1 + 2w_2 + w_3 = 1$$

$$neg_1 = 1w_1 + 0w_2 + w_3 = -1$$

$$neg_2 = 0w_1 + 1w_2 + w_3 = -1$$

We can try to solve this by subtracting *neg1* from *pos1*. The *w3* and *w1* terms cancel out leaving us with…

$$pos_1 - neg_1 \implies w_2 = 2$$

We can now substitute the value for *w2* into *neg2*…

$$2 + w_3 = -1 \implies w_3 = -3$$

We can now substitute the values for *w2* and *w3* into *pos1*…

$$w_1 + 2 - 3 = 1 \implies w_1 - 1 = 1 \implies w_1 = 2$$

This gives us our new line equation - which can be simplified down to…

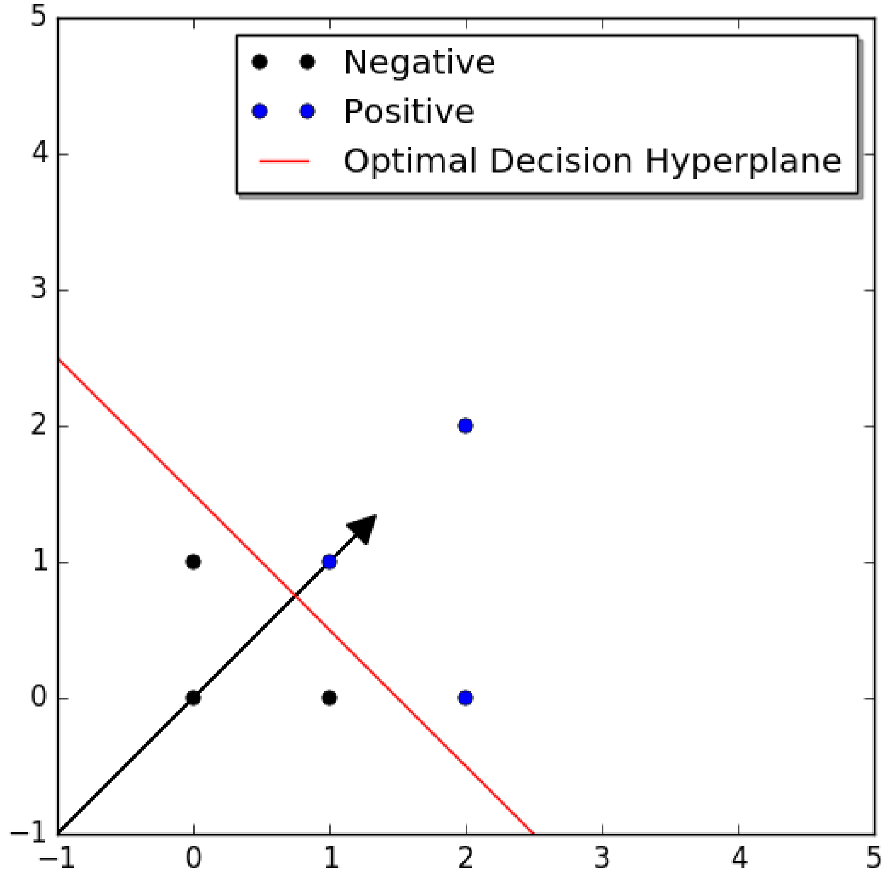
$$2x + 2y - 3 = 0 \implies x + y + \frac{3}{2} = 0$$

This implies that our vector passes through *(0, 0)* and *(1, 1)*, because the coefficients of *x* and *y* (the weight vector components) are 1. Note that you don't *need* to divide the line equation by 2, as above. Using *(1, 1)* or *(2, 2)* for the weight vector components are both perfectly sensible answers because the weight vector is *normal* to the optimal decision hyperplane - that is, it simply encodes a *direction*.

So we now have the components of our SVM weight vector. The SVM weight vector is comprised of *w1* and *w2*, so….

$$w = \begin{bmatrix} w_1 \\ \\ w_2 \\ \\ \end{bmatrix} \implies \begin{bmatrix} 1 \\ \\ 1 \\ \\ \end{bmatrix}$$

And you're done. Does this fit with what we predicted earlier? (Yes, it does).



0 Comments - *powered by utteranc.es*

Write Preview

Leave a comment

Styling with Markdown is supported

Sign in to comment

./backlog: Charlie's blog

Charles Newey
charlie.newey@gmail.com

charlienewey

perambulations through the world of data science