Supervised Learning: Perceptron Algorithm

April 12, 2021

1 Motivation

A college's acceptances are a yes/no decision. The factors the college considers are:
x_1 := test score / 10
x_2 := grades / 10

Can we predict whether a new (x_1, x_2) gets accepted or rejected?

2 Classification Problem

This is a classification problem, so our goal is not to fit a function to the data, but instead fit a function such that it splits the data into discrete, already defined categories. In the case of the college problem, we plot the points on an x_1 vs x_2 plot, and come up with the equation of a line $f(x_1.x_2) = \beta_2 x_2 + \beta_1 x_1 + \beta_0$ that splits the data in half. Then, for any given (x_1, x_2), our prediction equation is:

$$\hat{y} = \begin{cases} 0 & f(x_1, x_2) \leq 0 \\ 1 & f(x_1, x_2) > 0 \end{cases}$$

Generally, if each point on our plot has n coordinates, our dividers are going to be n-planes, or an n-1 subspace of $\mathbb{R}^n$, and our $f(x_1, x_2) = \beta^T X + \beta_0$, where both β and X are column vectors containing the respective elements. Just like linear regression, we can also bend our plane in polynomial shapes by adding new terms of the form x_i^n

3 Perceptron

3.1 Another Way

3.2 Logical Operators

Basically, if we plot points at $(0,0), (0,1), (1,0), (1,1)$, we need to come up with a dividing line or whatever such that all points above it, when plugged in, return true, and all other points return false. Try plugging in these points into equations of the form $f(x_1.x_2) = \beta_2 x_2 + \beta_1 x_1 + \beta_0$ that follow the rules listed for each logical operator and compare the output with the logical operator it's supposed to represent:
AND: $\beta_0 < 0, \beta_1 + \beta_0 < 0, \beta_2 + \beta_0 < 0, \beta_2 + \beta_1 + \beta_0 > 0$
OR: $\beta_0 < 0, \beta_1 + \beta_0 > 0, \beta_2 + \beta_0 > 0, \beta_2 + \beta_1 + \beta_0 > 0$
NOT: $\beta_0 > 0, \beta_1 + \beta_0 < 0$

You can compose the rest of these with one another to get other functions.

4 Finding the Constant Terms

Now that you know both the 2 variable and the general case of this algorithm, we can use what we've learned so far to figure out what the constant terms β are (ya know, the whole point of this algorithm).

4.1 Perceptron Trick

4.1.1 Intuition

For any misclassified point, we want to move the line closer to the misclassified points, because the line is the border between our classification 1 or 0, so crossing that line equates to changing the predicted value. Kind of like how we do the absolute trick, where we want the line to move to a place where it approximately fits with the data, we want to do a similar thing here. But here our line (or hyperplane) acts as a border between points, not as a representation of the data. So, we want the "broad side" of the line to move towards the points that are misclassified, so we flip all our $\pm$ signs to $\mp$ signs.

4.1.2 Equation

So for each misclassified point,
$f(\beta, \alpha, X, P) = (\beta - \alpha P)^T X$
Where:
$\beta :=$ Constants in vector form, including β_0
$\alpha :=$ learn rate
$X :=$ the data input (appended with a 1 for the constant term β_0)
$P :=$ the point that is misclassified $(p_1, p_2, ...)$

4.2 Algorithm

In general, the steps are as follows:

- Start with random weights
- figure out what points are misclassified (apply current linear eqn to the dataset and see if each corresponding data point matched with the actual data point's classification)
- For each point that is misclassified, change weights by the value of their corresponding x multiplied by the learn rate

Algorithm 1 Perceptron Algorithm

procedure STEP(X, β, y, α)
▷ X the data input, β the coefficients to the linear eqn, y the actual result values of the predictor output, α is the learning rate
$\hat{y} = \beta^T X$
for each point in (X, y) **do**

▷ Remember, when we read in the data, it's read in as (<input>, <output>), where X = input (n col matrix) and y = output (vector)

if $\hat{y} == 0$ and the corresponding y is different **then**:
$\beta+ = \alpha X$
end if

if $\hat{y} == 1$ and the corresponding y is different **then**:
$\beta- = \alpha X$
end if

end for
end procedure