CS 536 : Perceptrons - Geometry and Convergence

16:198:536

In the usual way, we have a collection of data points $\{(\underline{x}^i, y^i)\}_{i=1,\dots,m}$ sampled from some underlying distribution. Each $\underline{x}$ is a vector of k-features, and y represents some binary class the data point belongs to (will purchase/won't purchase, is a dog/is not a dog). We take the classes to be tagged with $y = 1$ or $y = -1$ in this case. Our goal is that for any new feature vector $\underline{x}$, we want to be able to predict the class value y from those features. **Perceptrons** assume that the class is a function of the features, $y = f(\underline{x})$, where f is assumed to be of the following form:

$$f(\underline{x}) = \text{sign}(b + x_1 w_1 + x_2 w_2 + \ldots + x_k w_k), \tag{1}$$

for a bias value b and a vector of weights $\underline{w}$. Note, this can be expressed in a more 'decision-esque' framework as:

$$f(\underline{x}) = \begin{cases} 1 & \text{if } \underline{w}.\underline{x} + b \geq 0 \\ -1 & \text{else.} \end{cases} \tag{2}$$

The goal of perceptrons, as a learning problem, is to determine a bias value b and a vector of weights $\underline{w}$ such that each point in the data set is correctly classified: $f(\underline{x}^i) = y^i$ for each $i = 1, \dots, m$.

Geometry

What do perceptrons 'look like'?

In one dimension ($k = 1$), this reduces to $f(x) = \text{sign}(b + wx)$, or for $w > 0$,

$$f(x) = \begin{cases} 1 & \text{if } x \geq -b/w \\ -1 & \text{else.} \end{cases} \tag{3}$$

In this case, the perceptron represents a simple thresholding function - if x is above the threshold $-b/w$, x is classified as $y = 1$, if x is below the threshold then x is classified as $y = -1$. Perceptrons can model data where all the positive class points lie on one side of a value, and negative class points lie on the other.

In general, for higher dimensions, the 'cutoff' or threshold is the line / hyperplane defined by $\underline{w}.\underline{x} + b = 0$. Everything on one side of this *linear separator* is classified as the positive class, everything on the other side is classified as the negative class.

Constraints

This interpretation suggests some immediate constraints on what kinds of data perceptrons can model. For instance, in dimension $k = 1$, the data $(x = -1, y = -1)$, $(x = 0, y = 1)$, $(x = 1, y = -1)$ cannot be modeled with a perceptron. There is no threshold value $-b/w$ that separates this points. Similarly in $k = 2$ dimensions, the classic example is $((0,0),1), ((0,1),-1), ((1,0),-1), ((1,1),1)$ - there is no line that can be drawn in two dimensions that separates the positive and negative class points in this data set. In general, perceptrons can capture data points where the classes are *linearly separable.* Non-linear separators represent a problem for later focus.

Learning Algorithm

But can perceptrons be learned? In particular, given a data set, how can we determine a bias value b and weight vector $\underline{w}$ that correctly classify each data point? Given that there are uncountably many possible choices for each, this seems to be a considerable problem , but the following algorithm can be used to construct a linear separator (*if one exists!*) in a systematic way:

The Perceptron Learning Algorithm:

- Begin with $\underline{w} = 0, b = 0$
- Identify a point that is misclassified, i.e., $f(\underline{x}^i) \neq y^i$
- Update the weights and biases in the following way:

$$\begin{aligned} \underline{w} &\leftarrow \underline{w} + y^i \underline{x}^i \\ b &\leftarrow b + y^i. \end{aligned} \tag{4}$$

- Repeat until all points are correctly classified

The reasoning behind the algorithm is this: suppose that $\underline{w}, b$ misclassifies $(\underline{x}, y)$, that is, $\underline{w}.\underline{x} + b \geq 0$ and $y = -1$ or $\underline{w}.\underline{x} + b < 0$ and $y = 1$. Let $\underline{w}' = \underline{w} + y\underline{x}$ and $b' = b + y$. In that case, we have

$$\underline{w}'.\underline{x} + b' = (\underline{w} + y\underline{x}).\underline{x} + (b + y) = (\underline{w}.\underline{x} + b) + y\left(||\underline{x}||^2 + 1\right). \tag{5}$$

Hence, if $\underline{w}.\underline{x} + b$ is too small (negative when y is positive), $\underline{w}'.\underline{x} + b'$ is larger. If $\underline{w}.\underline{x} + b$ is too large (positive when y is negative), $\underline{w}'.\underline{x} + b'$ is smaller. Either way, if $\underline{x}, y$ is misclassified under $\underline{w}, b$, then it is closer to being correctly classified under $\underline{w}', b'$. In effect, the learning algorithm identifies an incorrectly classified point, and then drags the weight vector and bias value in a direction toward correctly classifying the point.

While this suggests that each step of the algorithm moves the model closer to correctly identifying the chosen point, it is potentially surprising that it doesn't move the model *away* from classifying other points. In fact, we can make the following claim:

Theorem: If the data is linearly separable, then the perceptron learning algorithm terminates in finite time, producing a linear separator $\underline{w}.\underline{x} + b = 0$ that correctly classifies all points.

We put off the proof of this result for a moment to develop first some simplifications and additional concepts, to support the proof of a much stronger result.

Quantifying Convergence

It is convenient to define the following concept:

If the plane $\underline{w}.\underline{x} + b = 0$ is a linear separator for a data set, then the **margin** of this separator is defined to be the perpendicular distance from this plane to the nearest data point:

$$\gamma(\underline{w}, b) = \min_i \frac{|\underline{w}.\underline{x}^i + b|}{||\underline{w}||}. \tag{6}$$

Geometrically, this comes from the fact that the weight vector $\underline{w}$ is perpendicular to the plane defined by $\underline{w}.\underline{x} = 0$, so that for any general $\underline{x}$, $\underline{x} - \alpha\underline{w}$ will project $\underline{x}$ perpendicularly onto the plane $\underline{w}.\underline{x} + b = 0$ for some value of α.

Additionally, assume that each feature vector $\underline{x}$ is normalized, so that $||\underline{x}||^2 = 1$ - that is, the data cloud under analysis lies completely in the ball of radius 1. One reason this assumption is nice is the following - if every point lies within this ball, then the linear separators we are considering must cut through this ball. Geometrically, you can show that if the hyperplane $\underline{w}.\underline{x} + b = 0$ cuts through this ball, then $b^2/||\underline{w}||^2 \leq 1$. To see this, you can show that the point on the plane closest to the origin is given by $\underline{x} = -(b/||\underline{w}||^2)\underline{w}$, and in order for this point to lie within the ball we must have $||\underline{x}||^2 \leq 1$, which yields the inequality $b^2/||\underline{w}^2|| \leq 1$, or $b^2 \leq ||\underline{w}||^2$.

With this additional notion and assumption, we can state and prove a much stronger result:

Theorem: If a linear separator exists with weight vector $\underline{w}$, bias b, then the perceptron learning algorithm terminates in at most $4/\gamma^2(\underline{w}, b)$ steps.

This convergence result is in fact independent of both the ambient feature space dimension, and the number of data points - the only way the data enters into this bound is through the size of the margin.

Proof of Termination

We want to show that if there is a linear separator, the learning algorithm terminates, and yields a weight vector that acts as a linear separator. Note, the algorithm only terminates when all points are correctly classified and there are no more errors, so any weight vector the algorithm terminates on *must* be a linear separator.

Let $(b_0, \underline{w}_0), (b_1, \underline{w}_1), \ldots$ be the sequence of biases and weight vectors produced by the perceptron learning algorithm. We would like to get a handle on this progression, understanding how it evolves, and ideally, how it terminates. Note that by the update equations, we have that

$$\begin{aligned} \underline{w}_{t+1} &= \underline{w}_t + y^i \underline{x}^i \\ b_{t+1} &= b_t + y^i. \end{aligned} \tag{7}$$

where $\underline{x}^i, y^i$ is misclassified by $(b_t, \underline{w}_t)$. These relations yield a couple of important results. From the first, we get that

$$\underline{w}_{t+1}.\underline{w}_{t+1} = \underline{w}_t.\underline{w}_t + 2y^i\underline{x}^i.\underline{w}_t + \left(y^i\right)^2 \underline{x}^i.\underline{x}^i, \tag{8}$$

or, grouping things suggestively and observing that $y^i = \pm 1$ and the feature vectors are normalized,

$$||\underline{w}_{t+1}||^2 = ||\underline{w}_t||^2 + 2y^i(\underline{x}^i.\underline{w}_t + b_t) + 1 - 2y^i b_t. \tag{9}$$

From the second, we get that

$$b_{t+1}^2 = b_t^2 + 2b_t y^i + \left(y^i\right)^2 = b_t^2 + 2b_t y^i + 1. \tag{10}$$

Combining the two:

$$\left[||\underline{w}_{t+1}||^2 + b_{t+1}^2\right] = \left[||\underline{w}_t||^2 + b_t^2\right] + 2y^i(\underline{x}^i.\underline{w}_t + b_t) + 2. \tag{11}$$

Because $(\underline{x}^i, y^i)$ is misclassified at time t, we have that $y^i(\underline{x}^i.\underline{w}_t + b_t) \leq 0$, so

$$\left[||\underline{w}_{t+1}||^2 + b_{t+1}^2\right] \leq \left[||\underline{w}_t||^2 + b_t^2\right] + 2. \tag{12}$$

Unwrapping this recurrence, we have

$$||\underline{w}_T||^2 + b_T^2 \leq ||\underline{w}_0||^2 + b_0^2 + 2T = 2T. \tag{13}$$

From this, we see that the overall size of $\underline{w}_T, b_T$ are bound with respect to T. This must hold at every point in time. This makes sense - at any time during the progression of the algorithm, the current weights and bias can't jump too much from the previous, and therefore can't have wandered too far from the initial 0-weight, 0-bias model.

At this point, we haven't utilized the fact that there is a true linear separator. What this gives us is that for some $\underline{w}, b$, we have that for all $\underline{x}^i, y^i$ that $y^i(\underline{w}.\underline{x}^i + b) = |\underline{w}.\underline{x}^i + b|$. We will not show that the $\underline{w}_t, b_t$ converge to $\underline{w}, b$, but they are useful for bounding the overall behavior of T. Observe that, from the previous recurrence relations, we have that

$$\begin{aligned} \underline{w}.\underline{w}_{t+1} &= \underline{w}.\underline{w}_t + y^i \underline{w}.\underline{x}^i, \\ b * b_{t+1} &= b * b_t + b * y_i. \end{aligned} \tag{14}$$

or grouping suggestively,

$$\underline{w}.\underline{w}_{t+1} + bb_{t+1} = \underline{w}.\underline{w}_t + bb_t + y^i(\underline{w}.\underline{x}^i + b). \tag{15}$$

Utilizing the previous observation about $\underline{w}, b$ giving a linear separator, we have

$$\underline{w}.\underline{w}_{t+1} + bb_{t+1} = \underline{w}.\underline{w}_t + bb_t + |\underline{w}.\underline{x}^i + b|. \tag{16}$$

Bounding from below, we have

$$\underline{w}.\underline{w}_{t+1} + bb_{t+1} \geq \underline{w}.\underline{w}_t + bb_t + ||\underline{w}||\gamma(\underline{w}, b). \tag{17}$$

Unwrapping this recurrence as we did before,

$$\underline{w}.\underline{w}_T + bb_T \geq \underline{w}.\underline{w}_0 + bb_0 + T||\underline{w}||\gamma(\underline{w}, b) \geq T||\underline{w}||\gamma(\underline{w}, b). \tag{18}$$

Again, this result must hold at every time T.

By the Cauchy-Shwartz inequality, we have that

$$|\underline{w}.\underline{w}_T + bb_T| \leq \sqrt{||\underline{w}||^2 + b^2}\sqrt{||\underline{w}_T||^2 + b_T^2}, \tag{19}$$

and applying the previous result that any linear separator must pass through the unit sphere and therefore have $b^2 \leq ||\underline{w}||^2$,

$$|\underline{w}.\underline{w}_T + bb_T| \leq \sqrt{2||\underline{w}||^2}\sqrt{||\underline{w}_T||^2 + b_T^2}. \tag{20}$$

We can apply this here to combine these results and give us

$$T||\underline{w}||\gamma(\underline{w}, b) \leq \sqrt{2||\underline{w}||^2}\sqrt{2T}, \tag{21}$$

or

$$T \leq \frac{4}{\gamma^2(\underline{w}, b)}. \tag{22}$$

This says that *at any time during the run of the learning algorithm*, that current time is bounded. This gives us termination (if there is a linear separator) immediately. Further, however, this gives us that any time during the run of the learning algorithm is bounded by a quantity determined only the size of the margin. From the above, this upper bound on the runtime of the algorithm is independent of a) the dimensionality of the data, and b) the number of data points. The only way that the data itself enters into the picture at all is through the size of the margin.

The main takeaway from this result is that the perceptron learning algorithm terminates in finite time if there is a linear separator. However, it is worth noting the following: the upper bound on the number of steps *increases with the maximum margin of any linear separator.* That is, for data sets with *small margin separators*, we typically would expect (or at least cannot exclude) that the learning algorithm will take a large number of steps. This makes a lot of sense - if any separator has a small margin, small perturbations in the weight vector will cause the perceptron to misclassify points, pushing some data points over the boundary. It will be very difficult to identify a separating hyperplane, shifting the weight vector point by point as we do. The larger the largest margin, the easier to find and more stable any given linear separator will be.