Math 345 - Notes
Perceptron and PLA
September 10, 2018

The Learning Model

We try to estimate the *unknown* target function $f : \mathcal{X} \to \mathcal{Y}$ that takes the input $\mathbf{x}$ from the set $\mathcal{X}$, and maps it to y, an element of $\mathcal{Y}$. We use the *bold* notation to denote that $\mathbf{x}$ is a vector, since in most applications our input comes in with many categories/features.

Let $\mathcal{D}$ denote the data set used for training our model/ deciding on the best function to use. Every element $(\mathbf{x}, y)$ in $\mathcal{D}$ must satisfy $f(\mathbf{x}) = y$.

Let $\mathcal{H}$ denote the hypothesis set, that is, the set of all candidate formulas/models we consider to estimate f.

The learning algorithm picks a function $h : \mathcal{X} \to \mathcal{Y}$ from the hypothesis set $\mathcal{H}$ that appropriately approximates f. In other words, we want to optimize over all functions in $\mathcal{H}$ in order to find the best "fit" for f, so that $h \approx f$. The optimal h is picked when possible to best match the *training data* $\mathcal{D}$. The metric used to decide how close h is to f depends on the type of model we use and $\mathcal{D}$.

Perceptron

Keep the following application in mind: a bank wants to automate the decision to grant or deny credit to a customer. Suppose they have abundant data on past/current customers, which the bank can use to find a target function for a yes/no response to a customer application. The kind of data for each customer they might consider is: salary, time in residence, credit score, if the bank profited from past loans, if the customer was denied in previous applications etc. Suppose there are d categories that we want to consider as relevant to our decision. These factors should not have equal weight, in fact some should have negative weight. We can use a weighted sum to compute a "score" for each customer and if that score is above a certain threshold, we approve for credit, otherwise we deny it.

There are two ways to go about this problem: decide ahead of time on how the categories are weighted and what the threshold should be, or use the data to decide on a best set of weights and threshold to best match the data on previous/current customers. The first option leads to an analytic solution, without taking into consideration the data set. The second option uses data science to find / derive a model to fit the data. We will take the second approach.

The target function f is the *unknown* ideal formula for credit approval. It has a binary output (yes/no), which we will encode as $+1$ for "yes" and -1 for "no", so $\mathcal{Y} = \{-1, +1\}$. Suppose there are d categories we account for out input. Let w_i denote the weight associated to category i, for $1 \leq i \leq d$, and b the threshold for approval. Then we have our decision

$$\begin{aligned} \text{output} &= \begin{cases} +1 & , \quad \text{if } w_1x_1 + \cdots + w_dx_d > b \\ -1 & , \quad \text{if } w_1x_1 + \cdots + w_dx_d < b \end{cases} \\ &= \text{sign}\left(w_1x_1 + \cdots + w_dx_d - b\right). \end{aligned}$$

Recall that sign(x) equals $+1$ if $x > 0$ and it equals -1 if $x < 0$.

In order to be able to write the target function f in a more compact way, we introduce another category in the input vector, category 0, with $x_0 = 1$ for all possible input data. We let $w_0 = -b$ and define

$$\mathcal{X} = \{1\} \times \mathbb{R}^d = \{[x_0, x_1, \cdots, x_d]^T \,|\, x_0 = 1, x_1 \in \mathbb{R}, \cdots x_d \in \mathbb{R}\}.$$

$f : \mathcal{X} \to \{-1, +1\}$ as

$$\begin{aligned}
f(\mathbf{x}) = f(x_0, x_1, \cdots, x_d) &= \text{sign}\,(w_0 x_0 + \cdots + w_d x_d) \\
&= \text{sign}\left(\textstyle\sum_{k=0}^{d} w_k x_k\right) && \text{using summation notation} \\
&= \text{sign}(\mathbf{w} \cdot \mathbf{x}) && \text{sum is dot product of vectors } \mathbf{w} \text{ and } \mathbf{x} \\
&= \text{sign}(\mathbf{w}^T \mathbf{x}) && \text{matrix multiplication notation}
\end{aligned}$$

Here $\mathbf{v}^T$ stands for the transpose of the matrix $\mathbf{v}$ and $\mathbf{w} = [w_0, w_1, \cdots, w_d]^T$ represents the (column) vector encoding the weights for each category.

We must search for a function that is close to f in

$$\mathcal{H} = \{h : \mathcal{X} \to \mathcal{Y} \,|\, h(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x}),\ \mathbf{w} \in \mathbb{R}^{d+1}\}$$

Note that the function $h(\mathbf{x})$ is uniquely determined by $\mathbf{w}$, so finding h is equivalent to finding $\mathbf{w}$.

PLA (Perceptron Learning Algorithm)

The algorithm will determine what $\mathbf{w}$ should be, based on the data set $\mathcal{D}$, following the iterative process below. Suppose that the data set is linearly separable, that is, there is some vector $\mathbf{w}$ so that for all $(\mathbf{x}, y) \in \mathcal{D}$,

$$\text{sign}(\mathbf{w}^T \mathbf{x}) = y.$$

Suppose at time t in the iteration, the weights are given by $\mathbf{w}(t)$. The algorithm picks an element of $\mathcal{D}$ that is misclassified, call it $(\mathbf{x}(t), y(t))$ and uses it to update $\mathbf{w}$. The weight update rule in PLA is

$$\mathbf{w}(t+1) = \mathbf{w}(t) + y(t)\mathbf{x}(t)$$

Notes:

1. At time t, since $(\mathbf{x}(t), y(t))$ was misclassified, $y(t) \neq \text{sign}(\mathbf{w}(t)^T \mathbf{x}(t))$.
2. The rule moves the boundary in the direction of classifying $\mathbf{x}(t)$ correctly.
3. The algorithm continues for a finite number of iterations, until there are no misclassified examples in the data set.
4. We can initialize $\mathbf{w}(0)$ in any way we want. For example, we can let it be the zero vector.
5. At step t, we can choose any of the misclassified data points in order to update. This can be done by choosing at random, or cycling through the data set in a specific order.

The PLA algorithm ends by finding a vector $\tilde{\mathbf{w}}$ that works for all the data points in $\mathcal{D}$. Let the final hypothesis function be

$$h(\mathbf{x}) = \text{sign}(\tilde{\mathbf{w}}^T \mathbf{x}).$$

How well will h fit new data points? Or in other words, how "close" it is to the target function f?

Going back to our bank example, once h is found, it will be used to decide on approving or denying credit.

Example

We are given the data of the form $\begin{bmatrix} \mathbf{x} \\ y \end{bmatrix} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ y \end{bmatrix}$ with $x_0 = 1$ for all data points:

$$\mathcal{D} = \left\{ \begin{bmatrix} 1 \\ 1 \\ 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \\ 4 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 3 \\ 4 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \\ 1 \\ -1 \end{bmatrix}, \begin{bmatrix} 1 \\ 4 \\ 2 \\ -1 \end{bmatrix} \right\}.$$

We will use the PLA to approximate the line separating the data. We are looking into a set of weights $\mathbf{w}$ so that $\text{sign}(\mathbf{w}^T\mathbf{x}) = y$ for all points in the data set.

$t = 0:$ Set $\mathbf{w}(0) = \mathbf{0}$.

Check if points are misclassified, in order they are listed in $\mathcal{D}$:

$$\text{sign}\left([0, 0, 0] \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} \right) = 0 \neq 1$$

so the the data point is mislabeled. Call this data point $\begin{bmatrix} \mathbf{x(0)} \\ y(0) \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 2 \\ 1 \end{bmatrix}$

$t = 1:$ Let $\mathbf{w}(1) = \mathbf{w}(0) + y(0)\mathbf{x}(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + 1 \cdot \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}$

Check if points are misclassified, in order they are listed in $\mathcal{D}$:

$$\text{sign}\left([1, 1, 2] \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} \right) = 1$$

$$\text{sign}\left([1, 1, 2] \begin{bmatrix} 1 \\ 2 \\ 4 \end{bmatrix} \right) = 1$$

$$\text{sign}\left([1, 1, 2] \begin{bmatrix} 1 \\ 3 \\ 4 \end{bmatrix} \right) = 1$$

$$\text{sign}\left([1, 1, 2] \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \right) = 1 \neq -1$$

so the the data point is mislabeled. Call this data point $\begin{bmatrix} \mathbf{x(1)} \\ y(1) \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \\ -1 \end{bmatrix}$

$t = 2:$ Let $\mathbf{w}(2) = \mathbf{w}(1) + y(1)\mathbf{x}(1) = \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} + (-1) \cdot \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix}$

Check if points are misclassified, in order they are listed in $\mathcal{D}$:

$$\text{sign}\left([0, -1, 1] \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} \right) = 1$$

$$\text{sign}\left([0, -1, 1] \begin{bmatrix} 1 \\ 2 \\ 4 \end{bmatrix} \right) = 1$$

$$\text{sign}\left([0, -1, 1] \begin{bmatrix} 1 \\ 3 \\ 4 \end{bmatrix} \right) = 1$$

$$\text{sign}\left([0, -1, 1] \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \right) = -1$$

$$\text{sign}\left([0, -1, 1] \begin{bmatrix} 1 \\ 4 \\ 2 \end{bmatrix} \right) = -1$$

Since all data points are correctly classified, we output the set of weights $\mathbf{w} = \begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix}$, and

$$h(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x}) = \text{sign}(-x_1 + x_2).$$

Note that the target function can be geometrically described by the line where the sign function equals zero, that is where $-x_1 + x_2 = 0$ or $x_2 = x_1$.