Perceptron Learning Algorithm

Kelvin · Liang ziyoustep@gmail.com

August 26, 2018

1 Introduction to ML yes/no problem(Binary Classification)

Suppose there is a machine learning problem that we need to find out a final hypothesis f which can answer yes/no questions. For example: whether a bank want to approve a credit card for its customers. We will use credit card approval as an example in the following sections. The first thing we need to figure out is what's our hypothesis set. **Perceptron Learning Algorithm** is a way to come up with a fairy acceptable hypothesis set.

2 Perceptron Hypothesis Set

The main idea of Perceptron Hypothesis Set(PHS) is using **threshold** to classify the input. If the weighted score of a customer is greater than the threshold, the bank would approve a credit card to him/her, otherwise the bank would deny the application.

Perceptron Hypothesis Set is defined as:

$$h(X) = sign\left(\left(\sum_{i=1}^{n} w_i x_i\right) - threshold\right)$$

If the sign of $h(X)$ is $'+'$, the the bank would approve the application, otherwise the bank would deny if the sign of $h(X)$ is $'-'$.

We can simplify the form of PHS as follow: (Note that $w_0 = (-threshold)$

$$\begin{aligned} h(X) &= sign\left(\left(\sum_{i=1}^{n} w_i x_i\right) - threshold\right) \\ &= sign\left(\left(\sum_{i=1}^{n} w_i x_i\right) + (-threshold) \cdot (+1)\right) \\ &= sign\left(\sum_{i=0}^{n} w_i x_i\right) \\ &= sign\left(W \cdot X\right) \end{aligned}$$

Finally we get our PHS, the next step is to design an algorithm to select a hypothesis from this set. Note that the hypothesis in PHS may be good or bad, it depends on our luckiness. The algorithm we are going to propose is the Perceptron Learning Algorithm.

3 Perceptron Learning Algorithm (PLA)

Formal definition of PLA:

```
For t = 0,1,...
1. Find the next mistake of $W_t$ called $(X_i, y_i)$ where:
        $sign\,(W_t \cdot X_i) \neq y_i$
2. Correct the mistake by
        $W_{t+1} = W_t + y_iX_i$
...Until a full cycle not encountering any mistake
```

The PLA halts only if the input data set $\mathbb{D}$ is **linear separable**. It means that there exist a perfect W^* which makes no mistake for input $\mathbb{D}$. That is:

$$y_i = sign\,(W^* \cdot X_i)\ , for\, i \in \{1...n\}$$

What we need to do now is to prove that PLA will halt if $\mathbb{D}$ is linear separable. We start by mentioning the fact:
Fact. For perfect W^* the following inequality always holds:

$$y_iW^* \cdot X_i \geq \min_j\,(y_jW^* \cdot X_j) > 0\,, for\, j \in \{1...n\} \tag{1}$$

Claim 1. W_t gets more aligned with W^*
Proof: We will prove this by showing that the value of $W^* \cdot W_t$ is increasing at each update with (X_i, y_i).

$$\begin{aligned}
W^* \cdot W_{t+1} &= W^*(W_t + y_iX_i) \tag{2}\\
&= W^* \cdot W_t + y_iW^* \cdot X_i \tag{3}\\
&\geq W^* \cdot W_t + \min_j y_jW^* \cdot X_j\,, for\, j \in \{1...n\} \tag{4}\\
&> W^* \cdot W_t + 0 \tag{5}
\end{aligned}$$

So, we get $W^* \cdot W_{t+1} > W^* \cdot W_t + 0$ in the end. The increasing value of $W^* \cdot W_t$ comes from two sources. The first one is that the length of W_t gets larger. The second source is that W_t gets more aligned with W^*. To complete our proof of Claim 1, we need to prove Claim 2 first. After completing the proof of Claim 2, Claim 1 will automatically be true.
Claim 2. W_t does not grow too fast, which means the increasing value of $W^* \cdot W_t$ is mainly from the alignment between to two vectors. We are going to prove it by showing that the value of W_t doesn't increase much at each round.

$$\begin{aligned}
||W_{t+1}||^2 &= ||W_t + y_iX_i||^2 \tag{6}\\
&= ||W_t||^2 + 2y_iW^* \cdot X_i + ||y_iX_i||^2 \tag{7}\\
&\leq ||W_t||^2 + 0 + ||y_iX_i||^2 \tag{8}\\
&\leq ||W_t||^2 + \max_j ||y_jX_j||^2\,, for\, j \in \{1...n\} \tag{9}
\end{aligned}$$

Since $\max_j ||y_j X_j||^2$ is a constant. Intuitively, W_t is approaching to W^* steadily. We are going to prove that PLA will halt at some time concisely.

Corollary 1. After T mistake corrections

$$\cos\theta = \frac{W^* \cdot W_T}{||W^*|| \cdot ||W_T||} \geq \sqrt{T} \cdot C\,, for\ constant\ C$$

Proof: Following the same logic as equation (2) to (4) we have

$$\begin{aligned} W^* \cdot W_T &= W^*(W_{T-1} + y_i X_i) && (10)\\ &= W^* \cdot W_{T-1} + y_i W^* \cdot X_i && (11)\\ &\geq W^* \cdot W_{T-1} + \min_j y_j W^* \cdot X_j\,, for\, j \in \{1...n\} && (12)\\ \cdots & && (13)\\ &\geq W^* \cdot W_0 + T \min_j y_j W^* \cdot X_j\,, for\, j \in \{1...n\} && (14)\\ &\geq T \min_j y_j W^* \cdot X_j\,, for\, j \in \{1...n\} && (15) \end{aligned}$$

Note that $W_0 = (-threshold)$. For the simplicity of our proof, we choose $W_0 = \mathbf{0}$ when transiting from equation (14) to (15).
Following the same idea as equation (6) to (9) we have

$$\begin{aligned} ||W_T||^2 &= ||W_{T-1} + y_i X_i||^2 && (16)\\ &= ||W_{T-1}||^2 + 2y_i W^* \cdot X_i + ||y_i X_i||^2 && (17)\\ &\leq ||W_{T-1}||^2 + 0 + ||y_i X_i||^2 && (18)\\ &\leq ||W_{T-1}||^2 + \max_j ||y_j X_j||^2\,, for\, j \in \{1...n\} && (19)\\ \cdots & && (20)\\ &\leq ||W_0||^2 + T \max_j ||y_j X_j||^2\,, for\, j \in \{1...n\} && (21)\\ &= T \max_j ||y_j X_j||^2\,, for\, j \in \{1...n\} && (22) \end{aligned}$$

Again we choose $W_0 = \mathbf{0}$ here.
Combining equation (15) and (22), we get

$$\frac{W^* \cdot W_T}{||W^*|| \cdot ||W_T||} \geq \frac{T \min_j y_j W^* \cdot X_j}{||W^*|| \sqrt{T} \max_i ||y_i X_i||} = \sqrt{T} \cdot \frac{\min_j y_j W^* \cdot X_j}{||W^*|| \cdot \max_i ||y_i X_i||} \qquad (23)$$

Where the constant C is equal to $\frac{\min_j y_j W^* \cdot X_j}{||W^*|| \cdot \max_i ||y_i X_i||}$.

With equation (23) we can easily derive the upper bound of T

$$T \leq \frac{\max_i ||y_i X_i||^2 \cdot \left(W^* \cdot W^T\right)^2}{||W^T||^2 \left(\min_j y_j W^* \cdot X_j\right)^2} \qquad (24)$$

With all of the discussion above, we have proved that PLA will halt after many times of correction if $\mathbb{D}$ is linear separable.

But there are still many problems to be solve for PLA before it can be used in practice. One of the problems is that we are not sure about how long will it take for the algorithm to halt. Another problem is that PLA may not halt because of noisy input in the really world. To comfort this problem, we introduce another algorithm called Pocket Algorithm that is a refinement of PLA in the next section.

4 Refine PLA to Pocket Algorithm

```
Initialize pocket weights $\mathbf{W}$ For t = 0,1,...
1. Find a random mistake of $W_t$ called $(X_i, y_i)$ where:
      $sign\,(W_t \cdot X_i) \neq y_i$
2. Try to correct the mistake by
      $W_{t+1} = W_t + y_i X_i$
3. If $W_{t+1}$ makes fewer mistakes than $W$, replace $W$ with $W_{t+1}$
...Until enough of iterations
```

The idea behind this algorithm is that we always hold the best solution we have ever seen. The algorithm update W only if it encounter a better solution. The pros of this algorithm compared to PLA is that it's sure to halt after constant iteration specified by caller.

5 References

Almost all of the equations(except some proof) of this note are from Professor Hsuan-Tien Lin , NTU. If you wan to know more information about Machine Learning Foundation, please refer to Professor Lin's homesite.