Rosenblat Perceptron

Yulan van Oppen (s2640325)
Jarvin Mutatiina (s3555631)

December 11, 2018

1 Introduction

Many applications of machine learning involve binary classification systems, e.g. the spam filter in your email. The basic processing element in such a system is the linear threshold classifier called the perceptron. We very briefly introduce some elementary concepts

A *dichotomy* is a partition of a set into two disjoint subsets. We denote a dichotomy by $\mathfrak{D}_N^P := \{\boldsymbol{\xi}^\mu, S^\mu\}_{\mu=1}^P$, where $\boldsymbol{\xi}_\mu$ is an N-dimensional point and $S^\mu \in \{\pm 1\}$ is the corresponding class label, for $\mu = 1, \ldots, P$. The class label as a function of the data points $\boldsymbol{\xi}^\mu$ defines a function $S(\boldsymbol{\xi})$. Suppose this function is *homogeneously linearly separable*, that is, there exists an N-dimensional vector $\mathbf{w}$ such that

$$\text{sign}(\mathbf{w} \cdot \boldsymbol{\xi}^\mu) = S^\mu \quad \text{for} \quad \mu = 1, \ldots, P. \tag{1}$$

The *perceptron (storage) problem* is to find, for such a dichotomy, a vector such that (1) holds.

[ADDITIONAL STUFF]

I THINK THE PIECE IN RED SHOULD BE A LITTLE MORE RIGOROUS, THE PROPOSED CORRECTION IS IN BLUE, ABOVE (AFTER AN VERY SHORT INTRODUCTION). ALSO, WHAT ARE LINEARLY SEPARABLE CLASSES? THE ABOVE CORRECTION WOULD REQUIRE SOME SMALL CHANGES IN NOTATION

Consider a *dichotomy* with a set of observations that are homogeneously linearly separable; then we can deduce that all the observations fall within the following categories defined by weight vector(w) and the value of the observation, ξ.

$$f(x) = \begin{cases} -1 & w \cdot \xi > 0 \\ 0, & w \cdot \xi = 0 \\ 1 & w \cdot \xi < 0 \end{cases}$$

With this definition/hypothesis, we can assume that any new observations can comfortably fall within the existing partitions. Also Rosenblatt states that if the observation are indeed sampled from two linearly separable classes (??) then, a separating decision hyper-plane is formed between these two classes. The hyperplane is defined as;

$$\sum_{i=1}^{n} w_i \cdot x_i + b = 0$$

where n is the total number of observations and b is the bias. The bias is mainly meant to move the decision boundary away from the origin to cater for correct classification.
Given the distinction between the two classes in the sample space, the Perceptron's problem is to find optimal weights for the input vector that fully classify the inputs into one of the two available classes. In cases of misclassification, the weight vector is adapted until a correct classification is found; as will be described in the next section.

Describe the problem

- Goal of the perceptron is to correctly classify a set of input into one of two classes
- **Very brief illustration of a real-life application**
- Learning rules- criteria for modifying the weights in scenarios of misclassification

2 Notation

Maybe put this in the intro?

3 Methodology

This is rough; needs refinement. Feel free to change everything

Algorithm 1 Rosenblatt Perceptron algorithm

```
Variables:
$\mathbf{D} = \{\xi^{\mu}, S^{\mu}\}_{\mu=1}^{p}$        ▷ $\xi^{\mu}$ represents the input vector and $S^{\mu}$ is the corresponding class label
$w(t) = [w_1(t), w_2(t), w_3(t), \cdots]$        ▷ weight vector

1. Initialisation w(0) =0;
For time steps t=1,2,3,$\cdots$

$E^{\mu} = w \cdot \xi^{\mu} S^{\mu}$        ▷ At every time step compute $E^{\mu}$

If $E^{\mu} > 0$
w(t+1) = w(t)
else if $E^{\mu} \leq 0$
w(t+1) = w(t) + $\frac{1}{N}\xi^{\mu(t)} S^{\mu(t)}$

Repeat for all timesteps until optimal weights are found.
```

- Describe the equations and break down the sections eg the hebian learning
- **Brief mention of convergence?**
- Pseudo code

4 Results

In the following section, the number of points P in any given dichotomy will be related to the dimension N of the space under consideration through $P = \alpha N$. We employ the algorithm discussed in the previous section to obtain the fraction of successful runs $Q_{l.s.}$, as a function of α.

Unless specified otherwise, for each figure referred to below, we generate $n_D = 50$ dichotomies $\mathfrak{D}_N^P := \{\boldsymbol{\xi}^{\mu}, S^{\mu}\}_{\mu=1}^{P}$. For each dichotomy we allow for a maximum of $n_max = 100$ epochs when using the Rosenblatt algorithm. Moreover, the N-dimensional space we consider has default value $N = 20$. When checking whether the local potentials satisfy $E^{\mu} > c$, c is by default 0. It should be noted that in the "fraction of linearly separable dichotomies" y-axis label in each figure, linearly separable is meant as far as can be determined from running the Rosenblatt algorithm with the given settings. The actual fraction is always at least as large as the plotted values.

Holding everything else fixed, varying n_D yields the results depicted in Figure 1. Little variation can be observed. Varying n_{max}, however, significantly changes the results. This is shown in Figure 2. As we vary N, the graph of $Q_{l.s.}$ as a function of α approaches a step function. The point of discontinuity of the approached graph, however, does not appear to lie on the theoretical value $\alpha = 2$. A reason for this is discussed in the next section.

Enforcing a nonzero minimal stability on the solution, i.e. setting $c > 0$, intuitively decreases $Q_{l.s.}$ for any value of α. Figure 4 supports this intuition. Only for these computations, n_D (which was held fixed) was set to 100. This was in an effort to get rid of large oscillations in the graphs of $Q_{l.s.}$ as a function of α, although some are still present.

Modify the algorithm in order to find also inhomogeneous perceptron solutions by adding a clamped input to all feature vectors. Does Ql.s. change significantly?

5 Discussion

A higher value of n_D yields results that more accurately depict the true mean of the fraction of successful runs of the Rosenblatt algorithm. Figure 1 showed varying n_D had little effect on the graph of $Q_{l.s.}(\alpha)$. This suggests that (for most purposes) $n_D = 50$ considers enough different dichotomies for trustworthy results.

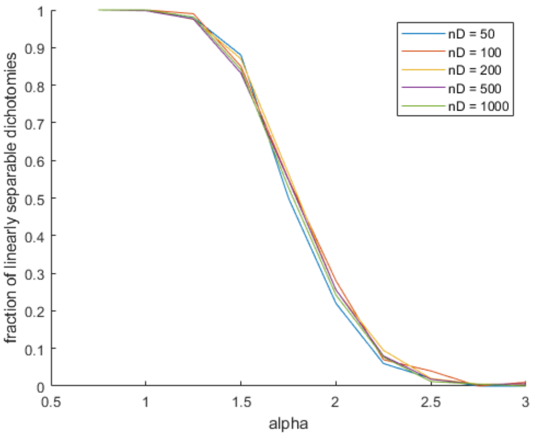

Figure 1: Plots of the fraction of linearly separable dichotomies for several values of n_D. Here $N = 20$, $n_{max} = 100$, $c = 0$, and $\alpha = 0.75, 1, 1.25, \ldots, 3$.

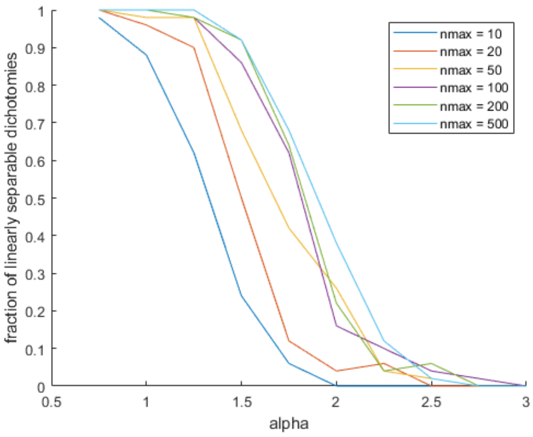

Figure 2: Plots of the fraction of linearly separable dichotomies for several values of n_{max}. Here $N = 20$, $n_D = 50$, $c = 0$ and $\alpha = 0.75, 1, 1.25, \ldots, 3$.

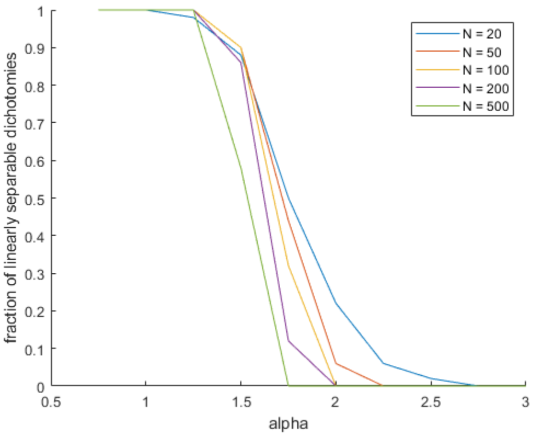

Figure 3: Plots of the fraction of linearly separable dichotomies for several values of N. Here $n_{max} = 100$, $n_D = 50$, $c = 0$, and $\alpha = 0.75, 1, 1.25, \ldots, 3$.

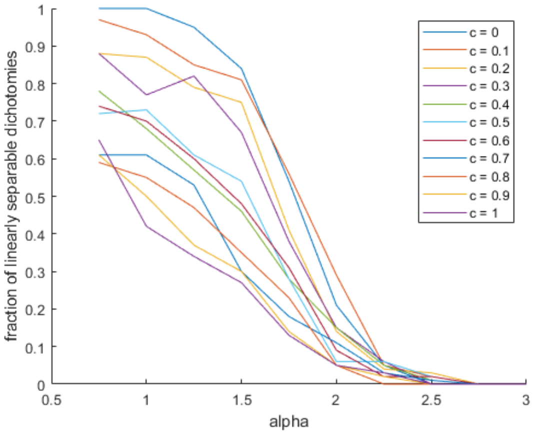

Figure 4: Plots of the fraction of linearly separable dichotomies for several values of c. Here $N = 20$, $n_{max} = 100$ and $n_D = 100$, $c = 0$ and $\alpha = 0.75, 1, 1.25, \ldots, 3$.

We know from theory that the Rosenblatt algorithm converges for linearly separable dichotomies in finitely many iterations, and for $\alpha < 2$ this is the case with probability 1. Finitely many, however, can turn out to be quite a lot. Increasing n_{max} increases the probability of convergence within n_{max} iterations. For $n_{max} = 100$ (our default value), we see the graph of $Q_{l.s.}(\alpha)$ in Figure 2 has generally lower values than its theoretical counterpart. This difference vanishes as n_{max} is sufficiently increased.

This observation moreover explains why the step function approached by the graphs in Figure 3 does not seem to have a discontinuity at $\alpha = 2$, but rather at $\alpha \approx 1.5$.

References