Book exercises solution

Mayra A. Vega Niño

Exercise 1.2
Suppose that we use a perceptron to detect spam messages. Let's say that each email message is represented by the frequency of occurrence of keywords, and the output is +1 if the message is considered spam.

a) Can you think of some keywords that will end up with a large positive weight in the perceptron?

b) How about keywords that will get a negative weight?

c) What parameter in the perceptron directly affects how many borderline messages end up being classified as spam?

Solution

a) Since spam messages are assigned the label +1, the most frequent words in this type of email will end up with a large positive weight in the perceptron algorithm. Examples of these words (phrases) can be: free, earn, change (password), money, take action, winner, own boss.

b) On the other side, the most frequent words in non-spam emails will end up with a negative weight. Examples of these words (phrases) can be: thank you, sincerely, regards, information.

c) The bias term b is the parameter responsible of the amount of messages that end up being classified as spam since it represents the threshold to decide if an email is spam or not.

Exercise 1.3
The weight update rule $\mathbf{w}(t+1) = \mathbf{w}(t) + y(t)\mathbf{x}(t)$ has the nice interpretation that it moves in the direction of classifying $\mathbf{x}(t)$ correctly.

a) Show that $y(t)\mathbf{w}^T(t)\mathbf{x}(t) < 0$. *[Hint: $\boldsymbol{x}(t)$ is misclassified by $\boldsymbol{w}(t)$.]*

b) Show that $y(t)\mathbf{w}^T(t+1)\mathbf{x}(t) > y(t)\mathbf{w}^T(t)\mathbf{x}(t)$. *[Hint: use $\boldsymbol{w}(t+1) = \boldsymbol{w}(t) + y(t)\boldsymbol{x}(t)$.]*

c) As far as classifying $\mathbf{x}(t)$ is concerned, argue that the move from $\mathbf{w}(t)$ to $\mathbf{w}(t+1)$ is a move "in the right direction".

Solution

a) Let $\tilde{y}(t) = sign(\mathbf{w}^T\mathbf{x})$ and $y(t)$ the sign of the true label of $\mathbf{x}(t)$. Since $\mathbf{x}(t)$ is misclassified we have that $y \neq \tilde{y}$. Suppose that $y = +1$, then,

$$\tilde{y}\mathbf{w}^T\mathbf{x} < 0, \qquad \text{since } \mathbf{x}(t) \text{ is misclassified.} \tag{1}$$

Likewise, if $y = -1$, then,

$$\tilde{y}\mathbf{w}^T\mathbf{x} < 0, \qquad \text{since } \mathbf{x}(t) \text{ is misclassified.} \tag{2}$$

From 1 and 2 we have that $y(t)\mathbf{w}^T(t)\mathbf{x}(t) < 0, \forall \mathbf{x}(t)$ that is misclassified.

b) We have that the update rule in the perceptron algorithm is $\mathbf{w}(t+1) = \mathbf{w}(t) + y(t)\mathbf{x}(t)$, from here we have the following,

$$\begin{gathered} y(t)\mathbf{w}(t+1) = y(t)\mathbf{w}(t) + y^2(t)\mathbf{x}(t) \\ \Rightarrow y(t)\mathbf{w}(t+1)\mathbf{x}(t) = y(t)\mathbf{w}(t)\mathbf{x}(t) + \underbrace{y^2(t)\mathbf{x}^2(t)}_{\geq 0}, \end{gathered}$$

Thus, $y(t)\mathbf{w}^T(t+1)\mathbf{x}(t) > y(t)\mathbf{w}^T(t)\mathbf{x}(t)$.

c) Let $\mathbf{w}(t)$ be the separating hyperplane in the perceptron algorithm. Suppose we have a sample $\mathbf{x}(t)$ which is misclassified, e.g., its true sign is $+1$ but is classified as $y(t) = -1$. Since $\mathbf{x}(t)$ is misclassified, in the update rule we will have $\mathbf{w}(t) + \underbrace{y(t)}_{<0}\mathbf{x}(t)$, so $\mathbf{w}(t+1)$ will be closer to $\mathbf{x}(t)$ than $\mathbf{w}(t)$ since we are moving it towards $\mathbf{x}(t)$. As we see, the update rule 'moves' in the right direction because it will be closer to $\mathbf{x}(t)$ so it has more chances to classify it correctly.

Exercise 1.11
We are given a data set D of 25 training examples from an unknown target function $f : X \to Y$, where $X = \mathbb{R}$ and $Y = \{-1, +1\}$. To learn f, we use a simple hypothesis set $H = \{h_1, h_2\}$ where h_1 is the constant $+1$ function and h_2 is the constant -1.

We consider two learning algorithms, S smart and C crazy. S chooses the hypothesis that agrees the most with D and C chooses the other hypothesis deliberately. Let us see how these algorithms perform out of the sample from the deterministic and probabilistic points of view. Assume in the probabilistic view that there is a probability distribution on X, and let $\mathbb{P}[f(x) = +1] = p$.

a) Can S produce a hypothesis that is *guaranteed* to perform better than random on any point outside D?.

b) Assume for the rest of the exercise that all the examples in D have $y_n = 1$. It is *possible* that the hypothesis that C produces turns out to be better than the hypothesis that S produces.

c) If $p = 0.9$, what is the probability that S will produce a better hypothesis than C?

d) Is there any value of p for which it is more likely than not that C will produce a better hypothesis than S?

Solution

a) No, neither S nor C can tell us something certain about f outside D.

b) Yes, because we don't know what is the distribution of the data outside D.

c) We are asked for $\mathbb{P}[E_{out}(S) < E_{out}(C)]$. If $p = 0.9$, then $\mathbb{P}[f(x) = +1] = 0.9$ and $\mathbb{P}[f(x) = -1] = 0.1$. Since all the points in D have $y_n = +1$, S will pick the hypothesis $h_1 = +1$ and C will pick $h_2 = -1$. Outside D, 90% of the points will have $f(x) = +1$, so the hypothesis h_1 will have at most a 10% error, on the other hand, the hypothesis h_2 will have at most a 90% error. Since $E_{out}(S) < E_{out}(C)$, $\mathbb{P}[E_{out}(S) < E_{out}(C)] = 1$.

d) If $p < 0.5$, we will have that $E_{out}(S) \leq 1 - p$ and $E_{out}(C) \leq p$, thus $E_{out}(C) \leq E_{out}(S)$.

Exercise 1.12
A friend comes to you with a learning problem. She says the target function f is *completely* unknown, but she has 4000 data points. She is willing to pay you to solve her problem and produce for her a g which approximates f. What is the best that you can promise her among the following:

a) After learning you will provide her with a g that you will guarantee approximates f well out of sample.

b) After learning you will provide her with a g, and with a high probability the g which you produce will approximate f well out of sample.

c) One of two things will happen.

   (i) You will produce a hypothesis g;

   (ii) You will declare that you failed.

   If you do return a hypothesis g, then with high probability the g which you produce will approximate f well out of sample.

Solution
The option *c*) is the best one since only after the learning process we will know if we have found a g that approximates f well out of sample. In case of having found a g with $E_{in}(g)$ small, the Hoeffding inequality guarantees us that $E_{in}(g) \approx E_{out}(g)$ with high probability thanks to the amount of data points we have.