Машинное обучение
Теоретическое задание

Кириленко Елена

8 октября 2018 г.

1 *Задача 1*

$$f(x) = (\overline{x_1} \vee x_3) \wedge (x_1 \vee x_2 \vee x_3)$$

Из лекции знаем формулы для нейронной реализации логических функций. Отсюда можем представить $f(x)$ используя следующее последовательное применение функций :

$$\overline{x_1} = [-x_1 + \frac{1}{2} > 0]$$

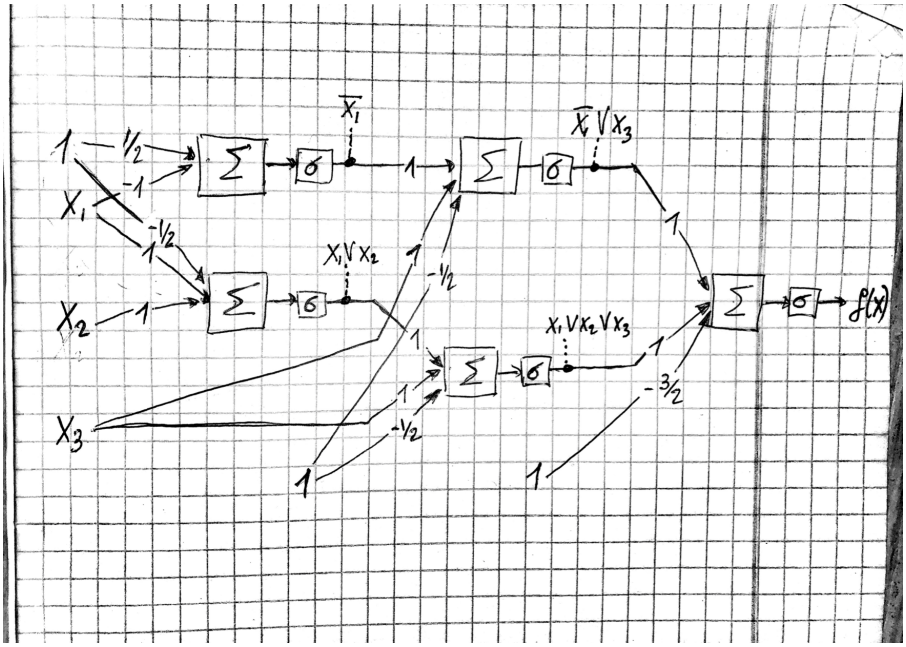
$$x_1 \vee x_2 = [x_1 + x_2 - \frac{1}{2} > 0]$$

$$\overline{x_1} \vee x_3 = [\overline{x_1} + x_3 - \frac{1}{2} > 0]$$

$$x_1 \vee x_2 \vee x_3 = [(x_1 \vee x_2) + x_3 - \frac{1}{2} > 0]$$

$$f(x) = [(\overline{x_1} \vee x_3) + (x_1 \vee x_2 \vee x_3) - \frac{3}{2} > 0]$$

Саму нейронную сеть можно увидеть на рисунке.



2 *Задача 2*

На лекции для удобства мы вводили обозначение u_h для выхода h-ого нейрона скрытого слоя. Поступим также.
Пусть мы рассматриваем объект x и x_i - его i-ый признак.
Обозначим

$$u_h(x) = \sum_{j=1}^{n} w_{jh} x_j$$

$$a_j(x) = \sum_{h=1}^{d} w'_{hj} u_h(x)$$

$$p_j(x) = \frac{exp(a_j)}{\sum_{k=1}^{m} exp(a_k)}$$

$$\mathcal{L}(W, W') = -\sum_{j=1}^{m} y_j log(p_j(x))$$

Начнем считать производные.

$$\frac{\partial \mathcal{L}}{\partial p_k} = \frac{y_k}{p_k}$$

Найдем $\frac{\partial p_k}{\partial a_j}$. Здесь есть 2 случая:

$$\frac{\partial p_k}{\partial a_k} = \frac{exp(a_k)}{\sum_{i=1}^{m} exp(a_i)} \left(1 - \frac{exp(a_k)}{\sum_{i=1}^{m} exp(a_i)}\right) = p_k(1 - p_k)$$

$$\frac{\partial p_k}{\partial a_j} = -\frac{exp(a_k)}{\sum_{i=1}^{m} exp(a_i)} \frac{exp(a_j)}{\sum_{i=1}^{m} exp(a_i)} = -p_k p_j \quad i \neq j$$

$$\frac{\partial \mathcal{L}}{\partial a_j} = \sum_{k=1}^{m} \frac{\partial \mathcal{L}}{\partial p_k}\frac{\partial p_k}{\partial a_j} = -\frac{y_j}{p_j}(p_j(1-p_j)) + \sum_{k=1\ k\neq j}^{m} -\frac{y_k}{p_k}p_k p_j = -y_j(1-p_j) - \sum_{k=1\ k\neq j}^{m} y_k p_j =$$

$$= -y_j + 2y_j p_j - \sum_{k=1}^{m} y_k p_j = -y_j + 2y_j p_j - p_j$$

Далее найдем $\frac{\partial a_s}{\partial u_h}$

$$\frac{\partial a_s}{\partial u_h} = \frac{\partial \sum_{i=1}^{d} w'_{is} u_i(x)}{\partial u_h} = w'_{hs}$$

Отсюда получим, что

$$\frac{\partial \mathcal{L}}{\partial u_h} = \sum_{s=1}^{m} \frac{\partial \mathcal{L}}{\partial a_s}\frac{\partial a_s}{\partial u_h} = \sum_{s=1}^{m} \frac{\partial \mathcal{L}}{\partial a_s} w'_{hs} = \sum_{s=1}^{m} (-y_s + 2y_s p_s - p_s) w'_{hs}$$

Ну и наконец посчитаем производные по весам.

$$\frac{\partial \mathcal{L}}{w_{jh}} = \frac{\partial \mathcal{L}}{\partial u_h}\frac{\partial u_h}{\partial w_{jh}} = \sum_{s=1}^{m} (-y_s + 2y_s p_s - p_s) w'_{hs} x_j$$

$$\frac{\partial \mathcal{L}}{w'_{hs}} = \frac{\partial \mathcal{L}}{\partial a_s}\frac{\partial a_s}{\partial w'_{hs}} = (-y_s + 2y_s p_s - p_s) u_h$$

3 Задача 3

а) Известно, что $x_i \sim N(0,1)$ и все x_i независимы. Тогда вектор $x = (x_1, x_2, ..., x_n) \sim N(0, I)$, где I - единичная матрица размера $n * n$. Тогда

$$Wx + b \sim N(b, WIW^T) = N(b, WW^T)$$

Мы хотим, чтобы все $h_i \sim N(0, \sigma^2)$, то есть хотим, чтобы $h \sim N(0, \Sigma)$, где Σ - некоторая матрица ковариаций, в которой на главной диагонали стоят σ^2 (а в остальных местах что-то).

Тогда получим условия : $b = 0 \quad WW^T = \Sigma$.
Второе условие означает, что на главной диагонали WW^T должны стоять σ^2.
Заметим, что на главной диагонали WW^T стоят $w_{11}^2 + w_{12}^2 + ... + w_{1n}^2$, ..., $w_{j1}^2 + w_{j2}^2 + ... + w_{jn}^2$, ..., $w_{m1}^2 + w_{m2}^2 + ... + w_{mn}^2$. Мы хотим, чтобы все эти суммы одновременно равнялись σ^2
Рассмотрим более простой случай - матрица W состоит из констант, то есть $W = cI$. Тогда второй условие можно записать как $nc^2 = \sigma^2$. Отсюда получаем, что $c = \sqrt{\frac{\sigma^2}{n}}$ и $W = \sqrt{\frac{\sigma^2}{n}} I$.

В итоге, поняли, что подходит $b = 0 \ \ W = \sqrt{\frac{\sigma^2}{n}} I$

4 Задача 4

Вспомним, как выражаются веса на $N+1$ итерации, чем веса на предыдущей итерации.

$$w_i^{N+1} = \widehat{w}_i^N exp(-\alpha_N y_i b_N(x_i))$$

+ Нормировка :

$$\widehat{w}_i^{N+1} = \frac{\widehat{w_i^N} exp(-\alpha_N y_i b_N(x_i))}{\sum_{i=1}^{l} \widehat{w}_i^N exp(-\alpha_N y_i b_N(x_i))}$$

Теперь распишем взвешенную ошибку базового классификатора относительно весов со следующего шага.

$$\sum_{i=1}^{l} \widehat{w}_i^{N+1}[b_N(x_i) \neq y_i] = \frac{\widehat{w_i^N} exp(-\alpha_N y_i b_N(x_i))[b_N(x_i) \neq y_i]}{\sum_{i=1}^{l} \widehat{w}_i^N exp(-\alpha_N y_i b_N(x_i))} =$$

Теперь заметим, что если индикатор $[b_N(x_i) \neq y_i] \neq 0$, то $b_N(x_i)y_i = -1$. В связи с этим фактом можем переписать предыдущее выражение.

$$= \frac{\widehat{w_i^N} exp(\alpha_N)[b_N(x_i) \neq y_i]}{\sum_{i=1}^{l} \widehat{w}_i^N exp(-\alpha_N y_i b_N(x_i))} =$$

Теперь разложим знаменатель на 2 суммы добавив индикаторы.

$$= \frac{\widehat{w_i^N} exp(\alpha_N)[b_N(x_i) \neq y_i]}{\sum_{i=1}^{l} \widehat{w}_i^N exp(-\alpha_N y_i b_N(x_i)) \ \ ([b_N(x_i) \neq y_i] + [b_N(x_i) = y_i])} =$$

$$= \frac{\widehat{w_i^N} exp(\alpha_N)[b_N(x_i) \neq y_i]}{\sum_{i=1}^{l} \widehat{w}_i^N exp(-\alpha_N)[b_N(x_i) \neq y_i] + \sum_{i=1}^{l} \widehat{w}_i^N exp(\alpha_N)[b_N(x_i) = y_i])} =$$

Теперь вспомним выражения для P и N:

$$P(b) = \sum_{i=1}^{l} u_i[b(x_i) = y_i]$$

$$N(b) = \sum_{i=1}^{l} u_i[b(x_i) \neq y_i]$$

Тогда можем сократить предыдущее выражение:

$$= \frac{exp(\alpha_N)N}{exp(-\alpha_N)N + exp(\alpha_N)P} =$$

Вспомним теорему с лекции про то, что в AdaBoost оптимальным α_N, то есть минимизирующим функционал $\widehat{Q}_N$ является $\alpha_N = \frac{1}{2}ln(\frac{P}{N})$. Тогда подставив, получим:

$$= \frac{\sqrt{\frac{P}{N}}N}{\sqrt{\frac{P}{N}}N + \sqrt{\frac{N}{P}}P} = \frac{\sqrt{PN}}{2\sqrt{PN}} = \frac{1}{2}$$

Все, доказали.

5 Задача 5

1) Обозначим $f_{T,i}$ - приближение на итерации T. То есть $f_{T,i} = \sum_{t=1}^{T} \alpha_t b_t(x_i)$.
Мы минимизируем следующий функционал

$$Q(\alpha, b, X^l) = \sum_{i=1}^{l} \mathcal{L}(f_{T-1,i} + \alpha_T b_T(x_i),\ y_i) \to min_{b_T, \alpha_T}$$

Вспомним, что в градиентном бустинге мы сроим бызовый алгоритм так, чтобы он приблидат вектор антиградиента - $-g$.

$$b_T = argmin_b \sum_{i=0}^{l} (b(x_i) + g_i)^2$$

где

$$g_i = \mathcal{L}'(f_{T-1,i},\ y_i)$$

Давайте рассмотрим в качестве функции потерь квадратичную функцию.
Тогда, получим, что $g_i = \left(\frac{1}{2}(f_{T-1,i} - y_i)^2\right)' = (f_{T-1,i} - y_i)$ И в итоге получим, что

$$b_T = argmin_b \sum_{i=0}^{l} (b(x_i) + (f_{T-1,i} - y_i)) = argmin_b \sum_{i=0}^{l} (b(x_i) - (y_i - f_{T-1,i}))$$

То есть то, что требовалось.

2) Аналогично предыдущему пункту :

$$b_T = argmin_b \sum_{i=0}^{l} (b(x_i) + g_i)^2$$

В нашем случае

$$f_{T-1} = (5, 10, 6, 3, 0) \ \ y = (6, 8, 6, 4, 1)$$

Найудем вектор градиента : $g_i = \mathcal{L}'(f_{T-1,i},\ y_i)$ В нашем случае :

$$g_i = 4(f_{T-1,i} - y_i)^3$$

Подставив значения, получим:

$$g = 4((5-6)^3, (10-8)^3, (6-6)^3, (3-4)^3(0-1)^3) = (-4, 32, 0, -4, 4)$$

То есть получим, что следующий базовый алгоритм будет настраиваться на вектор $g = (-4, 32, 0, -4, 4)$

3) Как и ранее

$$b_T = argmin_b \sum_{i=0}^{l} (b(x_i) + \mathcal{L}'(f_{T-1,i},\ y_i))^2$$

$$L'(y,\ z) = -(\frac{y}{z} - \frac{1-y}{1-z}) = \frac{z-y}{z(1-z)}$$

Тогда получим, что

$$b_T = argmin_b \sum_{i=0}^{l} \left(b(x_i) + \frac{f_{T-1,i} - y_i}{f_{T-1,i}\ \ (1 - f_{T-1,i})} \right)^2$$

$$\alpha_T = argmin_{\alpha>0} \sum_{i=1}^{l} \mathcal{L}(f_{T-1,i\alpha} b_T(x_i),\ \ y_i)) =$$

$$= argmax_{\alpha>0} \sum_{i=1}^{l} (y_i\ log(f_{T-1,i} + \alpha b_T(x_i)) + (1 - y_i)\ log(1 - f_{T-1,i} - \alpha b_T(x_i)))$$

6 Задача 6

1) Давайте вспомним как обновляются веса в AdaBoost.

$$w_i^{N+1} = w_i^N exp(-\alpha_N y_i b_N(x_i))$$

Видим, что у тех объектов, которые были распределены неверно, увеличивается вес, так как в этом случае $y_i b_N(x_i) = -1$.
У правильно классифицированных объектов вес наоборот уменьшается, так как $y_i b_N(x_i) = 1$. Кроме того, заметим, что на итерации N вес всех правильно классифицированных объектов домножается на одинаковое число : $e^{-\alpha_N}$, а вес всех неправильно классифицированных тоже домножается на одно и то же число, равное e^{α_N}.

Отсюда идея: взять такое множество алгоритмов B, что пороговый объект всегда классифицируется правильно. Из-за этого вес на пороговом объекте будет уменьшаться на каждой итерации.

Пусть у нас есть 2 класса на прямой. Объекты -1 класса лежат в отрицательных значениях, объекты 1 класса лежат в положительных. Кроме того есть выброс, который имеет положительную и очень большую координату C. Мы рассмотрим множество алгоритмов B, состоящее из двух алгоритмов : первый из них классифицирует $(x : x < 0)$ в класс -1, а $(x : x \geq 0)$ в класс 1, а второй $(x : x \leq C + 1)$ в класс -1, а $(x : x > C + 1)$ в класс 1.
Можно заметить, что такой класс удовлетворяет условию теоремы про AdaBoost, а именно для любого распределения весов взвешенное число ошибочных классификаций N больше $\frac{1}{2}$. Почему? Если суммарный вес хороших объектов больше веса выброса, то мы выбираем первый базовый алгоритм и классифицируем эти объекты верно, иначе выбираем второй алгоритм и и классифицируем выброс и все точки с отрицательными координатами хорошо.

Теперь возьмем в качестве порогового объекта объект с отрицательной координатой (можно поближе к 0). Обеими классификаторами он классифицируется верно, а поэтому его вес будет уменьшаться на каждой итерации. При этом мы будем периодически

выбирать то первый, то второй алгоритм так как то веса объектов из первого класса будут перевешивать, то вес выброса будет перевешивать. При этом заметим, что при выборе первого базового алгоритма вес выброса увеличивается так как он классифицируется неверно, а при выборе второго наоборот - уменьшается. Но во втором случае он уменьшается на ту же величину, что и в весе порогового объекта. Тогда получим, что если мы делаем неограниченное число итраций, то мы неограниченное число раз выбираем первый базовый алгоритм, а если мы выбираем первый алгоритм, то отношение веса выброса к весу порогового увеличивается на $e^{2\alpha_N}$. На самом деле у нас еще производится нормировка весов, чтобы все веса суммировались в единицу, но отношение веса выброса к весу порогового объекта от этого не менется.

2) В данном случае $\mathcal{L}(M) = log(1 + exp(-M))$
Найдем веса при такой функции потерь:

$$w_i = \mathcal{L}'(M_i) = \frac{exp(-M_i)}{1 + exp(-M_i)} = 1 - \frac{1}{1 + exp(-M_i)}$$

Рассмотрим пороговый объект. По определению у него маленький положительный margin. Ограничем margin для порогового объекта константой C. Тогда получим, что $M_i \leq C \Rightarrow w_i \geq 1 - \frac{1}{1+exp(-C)} = Const$

При этом для любого объекта мы можем ограничить margin сверху единицей: $\forall i w_i = 1 - \frac{1}{1+exp(-M_i)} < 1$. В том числе это верно и для выброса.

Тогда получим, что

$$\frac{w_{outlier}}{w_{threshold}} < \frac{1}{w_{threshold}} = \frac{1 + exp(-C)}{exp(-C)}$$

То есть получили, что бустинг с логистической функцией потерь устойчив к шуму.