Дипломна работа

Диана Генева <dageneva@qtrp.org>

2019

Съдържание

Глава 1

Нулева зона

Бла, бла, бла, аз съм толкова емоционална. Не знам

Приложение А

Приложение за AdaBoost

Ще разгледаме алгоритъма AdaBoost в дискретния случай, като доказателството следва това в samme.

Задачата е следната: Дадено е K-брой класове. Имаме тренировъчни данни $M := (x_1, y_1), (x_2, y_2), \cdots (x_n, y_n), x_i \in X, y_i \in \{1 \dots K\}, i = 1 \dots n$ и множество $\mathscr{H}$ от класификатори от вида $h_i : X \times \{1 \dots K\} \to \{0, 1\}$, където

$$h_i(x, y) = \begin{cases} 1, & \text{ако класът на x е y} \\ 0, & \text{иначе} \end{cases}$$

и съответно $\sum\limits_{j=1}^{K} h_i(x, j) = 1$. При дадена константа T, търсим линейна комбинация на функции от $\mathscr{H}$:

$$H = \sum_{i=1}^{T} \alpha_i h_i$$

която минимизира загубата на точност. По-точно, искаме да оптимизираме следната функция на загубата:

$$L_{0,1}(g, x, y) = \begin{cases} 0, & g(x, y) = max(g(x, 1), \dots g(x, K)) \\ 1, & \text{иначе} \end{cases}$$

Тоест, имаме загуба 1, ако класификаторът ни не е познал правилно класа и 0, в противен случай. Това може да се запише и по следния начин:

$$L_{0,1}(g, x, y) = \begin{cases} 0, & y = argmax_j(g(x, j)) \\ 1, & \text{иначе} \end{cases}$$

съответно можем да добавим константа, която няма да промени минимизацията:

$$L_{0,1}(g, x, y) = \begin{cases} 0, & y = argmax_j \left(g(x, j) - \dfrac{1}{K} \left(\sum\limits_{k=1}^{K} g(x, k) \right) \right) \\ 1, & \text{иначе} \end{cases} = \begin{cases} 0, & g(x, y) - \dfrac{1}{K} \left(\sum\limits_{j=1}^{K} g(x, j) \right) > 0 \\ 1, & \text{иначе} \end{cases}$$

Нека означим $\tilde{g}(x,y) = g(x,y) - \frac{1}{K}\left(\sum_{j=1}^{K} g(x,j)\right)$. Тоест, искаме да оптимизираме:

$$L_{0,1}(g,x,y) = \begin{cases} 0, & \tilde{g}(x,y) > 0 \\ 1, & \text{иначе} \end{cases}$$

Тъй като тази функция не е диференцируема, можем за удобство да използваме експоненциалната функция на загубата, дефинирана по следния начин:

$$l(g,x,y) = e^{-\tilde{g}(x,y)}$$

Освен че тя е диференцируема, имаме, че ако $\tilde{g}(x,y) > 0, l(g,x,y) > 0$ и ако $\tilde{g}(x,y) \leq 0, l(g,x,y) \geq 1$, което означава, че $l(g,x,y)$ ограничава отгоре $L_{0,1}(g,x,y)$ и в някакъв смисъл е "по-песимистична".

AdaBoost алгоритъмът избира последователно функции h_i от $\mathcal{H}$ и намира за всяка тегло α_i.

С H_t ще означаваме линейната комбинация, получена от първите t избрани класификатори. Имаме, че:

$$\begin{aligned} H_t(x,y) &= \sum_{i=1}^{t} \alpha_i h_i(x,y) \\ &= H_{t-1}(x,y) + \alpha_t h_t(x,y) \end{aligned}$$

$$\begin{aligned} \tilde{H}_t(x,y) &= \\ &= H_{t-1}(x,y) + \alpha h_t(x,y) - \frac{1}{K}\left(\sum_{j=1}^{K} H_{t-1}(x,j) + \alpha h_t(x,j)\right) \\ &= H_{t-1}(x,y) - \frac{1}{K}\left(\sum_{j=1}^{K} H_{t-1}(x,j)\right) + \alpha\left[h_t(x,y) - \frac{1}{K}\left(\sum_{j=1}^{K} h_t(x,y)\right)\right] \\ &= \tilde{H}_{t-1}(x,y) + \alpha\tilde{h}(x,y) \qquad \text{(A.0.1)} \end{aligned}$$

$$H_0(x,y) = 0 \text{ и } \tilde{H}_0(x,y) = 0 \quad \forall x \in X, \forall y \in \{1 \dots K\} \qquad \text{(A.0.2)}$$

На всяка итерация t на алгоритъма, дефинираме разпределение върху тренировъчните данни, което ще означаваме с $\mathcal{D}_t$, където $\mathcal{D}_t(i)$ дава вероятност на i-тия пример. Идеята е да може да се даде по-голяма вероятност на тези примери, върху които предните $t-1$ избрани функции бъркат и да се избере тази функция, която се представя най-добре върху така претеглените данни. В началото на алгоритъма за $\mathcal{D}_1$ избираме равномерно разпределение, тоест:

$$\mathcal{D}_1(i) = \frac{1}{n}, i = 1 \cdots n$$

Алгоритъмът е следният:

1. $D_1(i) = \frac{1}{n}, i = 1 \cdots n$
2. $H_0 = \emptyset$
3. За всяко t от 1 до T се прави следното:
   3.1. $h_t = argmin_{h \in \mathcal{H}} P_{i \sim D_t} \left(\tilde{h}(x_i, y_i) \leq 0\right)$
   3.2. $\varepsilon_t = P_{i \sim D_t} \left(\tilde{h}(x_i, y_i) \leq 0\right)$
   3.3. $\alpha_t = ln \left(\frac{1 - \varepsilon_t}{\varepsilon_t}\right) + ln\,(K - 1)$
   3.4. $H_t = H_{t-1} \cup \{(h_t, \alpha_t)\}$
   3.5. За всяко i от 1 до n:
      3.5.1. $\mathcal{D}_{t+1}(i) = \frac{\mathcal{D}_t(i) e^{-\alpha_t \tilde{h}_t(x_i, y_i)}}{\sum\limits_{j=1}^{n} \mathcal{D}_t(j) e^{-\alpha_t \tilde{h}_t(x_i, y_j)}}$
4. $H = H_T$
5. Връщаме H

Целта на AdaBoost алгоритъмът е да се намери такова H, което минимизира експоненциалната грешка l.

За да видим, че горният алгоритъм намира такова H, ще докажем няколко твърдения.

Първо нека положим $w_{t,i} = e^{-\tilde{H}_{t-1}(x_i, y_i)}$ за удобство.

Твърдение 1. $\mathcal{D}_t(i) = \frac{w_{t,i}}{\sum\limits_{j=1}^{n} w_{t,j}}$

Доказателство: Ще го докажем по индукция.

- База $t = 1$

$$\begin{aligned}
\frac{w_{1,i}}{\sum\limits_{j=1}^{n} w_{1,j}} &= \frac{e^{-\tilde{H}_0(x_i, y_i)}}{\sum\limits_{j=1}^{n} e^{\tilde{H}_0(x_j, y_j)}} \\
&\overset{\text{A.0.2}}{=\!=} \frac{1}{n} \\
&\overset{\text{ред 1.}}{=\!=} \mathcal{D}_1(i) \text{ за всяко i}
\end{aligned}$$

- Нека твърденито е изпълнено за $\mathcal{D}_t(i)$ за всяко i

- $\mathcal{D}_t(i) = \dfrac{w_{t,i}}{\sum\limits_{j=1}^{n} w_{t,j}} \Rightarrow \mathcal{D}_{t+1}(i) = \dfrac{w_{t+1,i}}{\sum\limits_{j=1}^{n} w_{t+1,j}}$

$$\mathcal{D}_{t+1}(i) \overset{\text{рег 3.5.1.}}{=\!=} \frac{\mathcal{D}_t(i) e^{-\alpha_t \tilde{h}_t(x_i,y_i)}}{\sum\limits_{j=1}^{n} \mathcal{D}_t(j) e^{-\alpha_t \tilde{h}_t(x_j,y_j)}} \overset{\text{ИХ}}{=} \frac{\dfrac{w_{t,i}}{\sum\limits_{k=1}^{n} w_{t,k}} e^{-\alpha_t \tilde{h}_t(x_i,y_i)}}{\sum\limits_{j=1}^{n} \dfrac{w_{t,j}}{\sum\limits_{k=1}^{n} w_{t,k}} e^{-\alpha_t \tilde{h}_t(x_j,y_j)}}$$

$$\overset{\text{def}}{=} \frac{\dfrac{e^{-\tilde{H}_{t-1}(x_i,y_i)}}{\sum\limits_{k=1}^{n} e^{-\tilde{H}_{t-1}(x_k,y_k)}} e^{-\alpha_t \tilde{h}_t(x_i,y_i)}}{\sum\limits_{j=1}^{n} \dfrac{e^{-\tilde{H}_{t-1}(x_j,y_j)}}{\sum\limits_{k=1}^{n} e^{-\tilde{H}_{t-1}(x_k,y_k)}} e^{-\alpha_t \tilde{h}_t(x_j,y_j)}}$$

$$= \frac{e^{-(\tilde{H}_{t-1}(x_i,y_i)+\alpha_t \tilde{h}_t(x_i,y_i))}}{\cancel{\sum\limits_{k=1}^{n} e^{-\tilde{H}_{t-1}(x_k,y_k)}}} \frac{\cancel{\sum\limits_{k=1}^{n} e^{-\tilde{H}_{t-1}(x_k,y_k)}}}{\sum\limits_{j=1}^{n} e^{-(\tilde{H}_{t-1}(x_j,y_j)+\alpha_t \tilde{h}_t(x_j,y_j))}}$$

$$= \frac{e^{-(\tilde{H}_{t-1}(x_i,y_i)+\alpha_t \tilde{h}_t(x_i,y_i))}}{\sum\limits_{j=1}^{n} e^{-(\tilde{H}_{t-1}(x_j,y_j)+\alpha_t \tilde{h}_t(x_j,y_j))}} \overset{\text{A.0.1}}{=\!=} \frac{e^{-\tilde{H}_t(x_i,y_i)}}{\sum\limits_{j=1}^{n} e^{-\tilde{H}_t(x_j,y_j)}} \overset{\text{def}}{=} \frac{w_{t+1,i}}{\sum\limits_{j=1}^{n} w_{t+1,j}}$$

Твърдение 2. $P_{i\sim D_t}\left(\tilde{h}(x_i,y_i) \leq 0\right) = \sum\limits_{i:\tilde{h}(x_i,y_i)\leq 0} \dfrac{w_{t,i}}{\sum\limits_{j=1}^{n} w_{t,j}}$

Доказателство: Използвайки горното твърдение, имаме че:

$$P_{i\sim D_t}(\tilde{h}(x_i,y_i) \leq 0) = \sum_{i:\tilde{h}(x_i,y_i)\leq 0} \mathcal{D}_t(x_i) = \sum_{i:\tilde{h}(x_i,y_i)\leq 0} \frac{w_{t,i}}{\sum\limits_{j=1}^{n} w_{t,j}}$$

Твърдение 3. Изборът на $h_t = argmin_{h\in\mathcal{H}} P_{i\sim D_t}(\tilde{h}(x_i,y_i) \leq 0)$ от точка 3.2 на алгоритъма минимизира експоненциалната грешка на H_t върху тренировъчното множество, тоест:

$$h_t = argmin_{h\in\mathcal{H}} \left(\frac{1}{n} \sum_{i=1}^{n} l(H_{t-1} + Ch, x_i, y_i) \right),$$

където C е произволна константа.

Доказателство:

$$h_t = argmin_{h\in\mathcal{H}} \left(\frac{1}{n}\sum_{i=1}^{n} l(H_{t-1}+Ch, x_i, y_i)\right)$$
$$= argmin_{h\in\mathcal{H}} \left(\frac{1}{n}\sum_{i=1}^{n} e^{-(\tilde{H}_{t-1}(x_i,y_i)+C\tilde{h}(x_i,y_i))}\right)$$
$$= argmin_{h\in\mathcal{H}} \left(\frac{1}{n}\sum_{i=1}^{n} e^{-\tilde{H}_{t-1}(x_i,y_i)} e^{-C\tilde{h}(x_i,y_i)}\right)$$

Махаме константата $\frac{1}{n}$, тъй като тя не влияе на минимизацията, и заместваме с $w_{t,i}$.

$$= argmin_{h\in\mathcal{H}} \left(\sum_{i=1}^{n} w_{t,i} e^{-C\tilde{h}(x_i,y_i)}\right)$$

Сега можем да разделим сумата на две, в зависимост $\tilde{h}$ е сбъркал, тоест дали $\tilde{h}(x_i,y_i) \leq 0$. Ако това е изпълнено, то $\tilde{h}(x_i,y_i) = h(x_i,y_i) - \frac{1}{K}\sum_{j=1}^{K}(h(x_i,j)) = 0 - \frac{1}{K} = -\frac{1}{K}$ и е равно на $1-\frac{1}{K}$ в противен случай, тоест:

$$= argmin_{h\in\mathcal{H}} \left(\sum_{i:\tilde{h}(x_i,y_i)>0}^{n} w_{t,i} e^{-C(1-1/K)} + \sum_{i:\tilde{h}(x_i,y_i)\leq 0}^{n} w_{t,i} e^{C/K}\right)$$
$$= argmin_{h\in\mathcal{H}} \left(\left[\sum_{i=1}^{n} w_{t,i} e^{-C} - \sum_{i:h(x_i)\neq y_i}^{n} w_{t,i} e^{-C}\right] + \sum_{i:h(x_i)\neq y_i}^{n} w_{t,i} e^{C}\right)$$
$$= argmin_{h\in\mathcal{H}} \left(\sum_{i=1}^{n} w_{t,i} e^{-C} + \sum_{i:h(x_i)\neq y_i}^{n} w_{t,i}(e^{C} - e^{-C})\right)$$

$\sum_{i=1}^{n} w_{t,i} e^{-C}$ е константа спрямо h, затова също не участва в минимизацията и тогава:

$$= argmin_{h\in\mathcal{H}} \left(\sum_{i:h(x_i)\neq y_i}^{n} w_{t,i}(e^{C} - e^{-C})\right)$$
$$= argmin_{h\in\mathcal{H}} \left((e^{C} - e^{-C}) \sum_{i:h(x_i)\neq y_i}^{n} w_{t,i}\right)$$

Тъй като и $(e^C - e^{-C})$ не зависи от h:

$$= argmin_{h \in \mathcal{H}} \left(\sum_{i:h(x_i) \neq y_i}^{n} w_{t,i} \right),$$

Можем да умножим по константата $\frac{1}{\sum_{j=1}^{n} w_{t,j}}$ и да получим:

$$= argmin_{h \in \mathcal{H}} \left(\frac{\sum_{i:h(x_i) \neq y_i}^{n} w_{t,i}}{\sum_{j=1}^{n} w_{t,j}} \right),$$

$$= argmin_{h \in \mathcal{H}} \left(\sum_{i:h(x_i) \neq y_i}^{n} \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}} \right),$$

$$\overset{\text{Твърдение 2}}{=} P_{i \sim D_t}(h(x_i) \neq y_i)$$

Твърдение 4. Изборът на α_t от точка 3.3 на алгоритъма минимизира експоненциалната грешка на H_t върху тренировъчното множество, тоест:

$$\alpha_t = argmin_\alpha (\frac{1}{n} \sum_{i=1}^{n} l(H_{t-1} + \alpha h_t, x_i, y_i))$$

Доказателство:

От точка 3.2 имаме, че $\varepsilon_t = P_{i \sim D_t}(h(x_i) \neq y_i) \overset{\text{Твърдение 2}}{=} \sum_{i:h(x_i) \neq y_i} \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$

От предишното твърдение знаем, че:

$$argmin_\alpha (\frac{1}{n} \sum_{i=1}^{n} l(H_{t-1} + \alpha h_t, x_i, y_i)) = argmin_\alpha (\sum_{i:h(x_i) = y_i} w_{t,i} e^{-\alpha} + \sum_{i:h(x_i) \neq y_i} w_{t,i} e^{\alpha})$$

Това, от което се интересуваме, е производната по α.

$$
\begin{aligned}
&\frac{\partial(\sum\limits_{i:h(x_i)=y_i} w_{t,i}e^{-\alpha} + \sum\limits_{i:h(x_i)\neq y_i} w_{t,i}e^{\alpha})}{\partial\alpha} = 0 && && \longleftrightarrow \\
&-\left(\sum_{i:h(x_i)=y_i} w_{t,i}\right)e^{-\alpha} + \left(\sum_{i:h(x_i)\neq y_i} w_{t,i}\right)e^{\alpha} = 0 && && \longleftrightarrow \\
&\left(\sum_{i:h(x_i)\neq y_i} w_{t,i}\right)e^{\alpha} = \left(\sum_{i:h(x_i)=y_i} w_{t,i}\right)e^{-\alpha} && && \longleftrightarrow \\
&e^{2\alpha} = \frac{\sum\limits_{i:h(x_i)=y_i} w_{t,i}}{\sum\limits_{i:h(x_i)\neq y_i} w_{t,i}} && \text{/логаритмуваме} && \longleftrightarrow \\
&2\alpha = ln\left(\frac{\sum\limits_{i:h(x_i)=y_i} w_{t,i}}{\sum\limits_{i:h(x_i)\neq y_i} w_{t,i}}\right) && && \longleftrightarrow \\
&2\alpha = ln\left(\frac{\sum\limits_{i=1} w_{t,i} - \sum\limits_{i:h(x_i)\neq y_i} w_{t,i}}{\sum\limits_{i:h(x_i)\neq y_i} w_{t,i}}\right) && \text{/умножаваме и делим на } \sum_{i=1}^{n} w_{t,i} && \longleftrightarrow \\
&2\alpha = ln\left(\frac{1 - \frac{\sum\limits_{i:h(x_i)\neq y_i} w_{t,i}}{\sum\limits_{i=1}^{n} w_{t,i}}}{\frac{\sum\limits_{i:h(x_i)\neq y_i} w_{t,i}}{\sum\limits_{i=1}^{n} w_{t,i}}}\right) && \text{/ заместваме с } \varepsilon_t && \longleftrightarrow \\
&\alpha = \frac{1}{2}ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)
\end{aligned}
$$

Което съвпада с избора на α от точка 3.3 на алгоритъма.

От Твърдение 3 и Твърдение 4 следва, че полученото в точка 5. от алгоритъма H ще е с минимална експоненциална грешка върху тренировъчните данни.