11.05.2015

WYKŁAD

Algorytm AdaBoost

Yoav Freund, Robert Schapire

Artykuł w 1997, nagroda w 2003

To jest praca z dziedziny learningu.
Powiedzmy, że chcemy zaprojektować klasyfikator. Będziemy się zajmować klasyfikatorami binarnymi, czyli takimi, które po prostu odpowiadają TAK lub NIE. Taki klasyfikator to funkcja $f: X \to \{-1, 1\}$, gdzie X to jakiś zbiór obiektów. Możemy sobie wyobrazić, że X to zbiór drużyn piłkarskich i chcemy poznać odpowiedź, czy wygrają następny mecz. Mamy wiele tzw. słabych klasyfikatorów $f_1, f_2, f_3, \ldots : X \to \{-1, 1\}$ typu: „czy wygrali ostatni mecz", „czy mają dobrego trenera", „czy są z kraju, który zdobył kiedyś mistrzostwo świata" itp. Idea jest taka, by z nich zbudować dobry klasyfikator, taki, który myli się rzadko.

Można by zorganizować głosowanie między $f_1, f_2, \ldots, f_k$.
Czyli zrobić $f = f_1 + f_2 + \ldots + f_k$ i sprawdzić, czy $f(x) \geq 0$ dla pewnego $x \in X$.

Jest jednak lepszy pomysł. Mianowicie można zorganizować głosowanie z wagami.

Weźmiemy $f = \alpha_1 f_1 + \alpha_2 f_2 + \dots + \alpha_n f_n$

A potem odpowiedź to $sgn(f(x))$.

Pytanie tylko jak dobrać wagi α_i.

Model jest taki: mamy pewien zbiór (słabych) klasyfikatorów $f_1, f_2, \dots, f_n$ oraz zbiór, na którym możemy się uczyć $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$, gdzie $x_i \in X$, $y_i \in \{-1, 1\}$.

Chcemy znaleźć $F = \sum_{j=1}^{k} f_{i_j} \alpha_j$, by był w pewnym sensie dobry. Oczywiście wtedy $F: X \to [-1, 1]$

AdaBoost — adaptive boosting. Boosting jest chyba od sumowania, adaptive od tego, że będziemy dobierać po kolei. Okazuje się, że w praktyce to bardzo dobrze działa, choć algorytm jest dość prosty.

Tu tak naprawdę są 2 decyzje do podjęcia:

- które $f_{i_1}, \dots, f_{i_k}$ wybrać?
- jakie $\alpha_1, \dots, \alpha_k$ im dać?

Będziemy za każdym razem dobierać optymalny f_{i_j}, w pewnym sensie zachłannie i przydzielać mu wagę.

Zamiast $f_{i_1}, f_{i_2}, \dots, f_{i_k}$ będziemy oznaczać $f_1, f_2, \dots, f_k$ wybrane klasyfikatory, dla uproszczenia notacji.

Niech $F_j(x_i) = \sum_{l=1}^{j} \alpha_l \cdot f_l(x_i)$

Powiedzmy, że mamy już $f_1, \dots, f_{m-1}$ razem z $\alpha_1, \dots, \alpha_{m-1}$ i chcemy wybrać f_m oraz α_m.

Będziemy chcieli tak wybrać, by błąd

$$E = \sum_{i=1}^{N} e^{-y_i \cdot F_m(x_i)}$$

był jak najmniejszy.

Zauważmy, że

$$\sum_{i=1}^{N} e^{-y_i F_m(x_i)} = \sum_{i=1}^{N} e^{-y_i F_{m-1}(x_i)} \cdot e^{-y_i \alpha_m f_m(x_i)}$$

Możemy więc myśleć, że jest to takie wybranie f_m, α_m, by błąd $e^{-y_i \alpha_m f_m(x_i)}$ z wagami $e^{-y_i F_{m-1}(x_i)}$ był jak najmniejszy. Zobaczmy, że $y_i \cdot F_{m-1}(x_i)$ bliskie 1, gdy $F_{m-1}(x_i)$ bliskie $y_i \in \{-1, 1\}$. A więc wtedy waga x_i to około e^{-1}. Gdy ~~np.~~ natomiast np. $y_i = 1$, a $F_{m-1}(x_i) = -0{,}8$, to waga to $e^{0,8}$, czyli spora.

Oznaczmy wagi $w_i^{(m)} = e^{-y_i F_{m-1}(x_i)}$ dla $m > 1$, $w_i^{(1)} = 1$.

przypadek skrajny

Mamy $E = \sum_{i=1}^{N} w_i^{(m)} e^{-y_i \alpha_m f_m(x_i)}$

Rozpisujemy

$$E = \sum_{i=1}^{N} w_i^{(m)} e^{-y_i \alpha_m f_m(x_i)} = \sum_{y_i = f_m(x_i)} w_i^{(m)} e^{-\alpha_m} +$$

$$+ \sum_{y_i \neq f_m(x_i)} w_i^{(m)} e^{\alpha_m} = \sum_{i=1}^{N} w_i^{(m)} e^{-\alpha_m} +$$

$$+ \sum_{y_i \neq f_m(x_i)} w_i^{(m)} \left(e^{\alpha_m} - e^{-\alpha_m} \right)$$

Zobaczmy, że dla ustalonego α_m suma zależy tylko od tego jak duża jest $\sum_{y_i \neq f_m(x_i)} w_i^{(m)}$.

Czyli po prostu trzeba wybrać f_m, który ma najmniejszy ważony błąd względem $w_i^{(m)}$.

Jak już mamy f_m to teraz pytanie brzmi jakie α_m wybrać. Żeby to zrobić to po prostu zróżniczkujmy E po α_m.

$$\frac{dE}{d\alpha_m} = -\sum_{y_i = f_m(x_i)} w_i^{(m)} e^{-\alpha_m} + \sum_{y_i \neq f_m(x_i)} w_i^{(m)} e^{\alpha_m}$$

Aby $\frac{dE}{d\alpha_m} = 0$ mamy

$$\sum_{y_i = f_m(x_i)} w_i^{(m)} e^{-\alpha_m} = \sum_{y_i \neq f_m(x_i)} w_i^{(m)} e^{\alpha_m}$$

$$\Downarrow$$

$$e^{2\alpha_m} = \frac{\sum_{y_i = f_m(x_i)} w_i^{(m)}}{\sum_{y_i \neq f_m(x_i)} w_i^{(m)}}$$

Czyli w efekcie dostajemy

$$\alpha_m = \frac{1}{2} \ln \left(\frac{\sum_{y_i = f_m(x_i)} w_i^{(m)}}{\sum_{y_i \neq f_m(x_i)} w_i^{(m)}} \right)$$

Dla ozn. $\varepsilon_m = \sum_{y_i \neq f_m(x_i)} w_i^{(m)} \Big/ \sum_{i=1}^{N} w_i^{(m)}$ mamy

$$\alpha_m = \frac{1}{2} \ln \left(\frac{1 - \varepsilon_m}{\varepsilon_m} \right)$$

Czyli po prostu ~~[illegible]~~ szukamy f_m z min. sumy

$\sum_{y_i \neq f_m(x_i)} w_i^{(m)}$, a jak tą mamy, to banalnie obliczamy

ε_m, a z tego α_m.

W praktyce (chyba) możemy wyprodukować wiele losowych (pewnie słabych) klasyfikatorów i z nich zrobić dobry używając AdaBoost. Podobno AdaBoost jest dość odporny na tzw. overfitting, czyli zbytnie dopasowanie do danych treningowych.