VU MIF

Duomenų analizė

„AdaBoost“ algoritmas

Parengė: Tomas Anbinderis

2004 Vilnius

“AdaBoost” metodo esmė

“Paskatinimas” (toliau - “Boosting”) tai viena iš žymiausiu “mokymo” idėjų paskelbtų per paskutinius 10 metų. Iš pradžių „boosting“ metodai buvo kuriami klasifikavimo užduotims, tačiau jie nesunkiai gali būti išplesti ir regresijos uždaviniams spręsti. „Boosting“ metodų pagrindinė idėja yra tame, kad „boosting“ metodai sujungia (kombinuoja) daugelio „silpnųjų“ (bazinių) kasifikatorių išeitis (toliau „output“), tam kad suformuoti „galinga“ bendra išeitį.

1995 metais Freund ir Schapire pasiūlė vieną iš šiuo metų populiariausių klasifikavimo algoritmų - „AdaBoost.M1“. Tarkime turime klasifikavimo užduotį, kur output kintamieji priklauso Y ir gali prilygti vienai iš dviejų reikšmių: -1 arba +1. Įeičiai (toliau „input“) paduodamas vektorius X. Klasifikatorius $G(X)$ duoda prognozę (viena iš dviejų reikšmių: -1 arba +1). Akivaizdu jog klaidos rodiklis mokomajam (toliau „training“) pavyzdžiui:

$$\text{err} = 1/N \left(\sum_{i=1}^{N} I(y_i \neq G(x_i)) \right).$$

Silpnasis klasifikatorius – tai toks klasifikatorius, kurio klaidos rodiklis yra nežymiai mažesnis negu spejimo atveju (t.y. klaidos rodiklis nežymiai mažesnis už 50%). „Boosting“ idėja yra nuosekliai taikyti silpnajį klasifikavimo algoritmą pastoviai modifikuojamoms duomenų versijoms. Tokiu būdu rezultate gaunama silpnųjų klasifikatorių seka $G_m(x)$, $m = 1,2,\ldots,M$. Toliau visų šių klasifikatorių prognozės yra kombinuojamos per svertinį balsų daugumą (angl.: „weighted vote majority“), tam kad suformuoti galutinę prognozę:

$$G(x) = \text{sign} \left(\sum_{m=1}^{M} \alpha_m G_m(x) \right).$$

Čia $\alpha_1,\alpha_2,\ldots,\alpha_M$ yra apskaičiuojami „boosting“ algoritmu ir nurodo kiekvieno atitinkančio klasifikatoriaus $G_m(x)$ indelį į galutinį rezulatatą (prognozę). Efektas yra tame kad tikslesniems klasifikatoriams suteikiamas didesnis svoris, dėl to skaičiuojant galutinę prognozę tikslesnieji klasifikatoriai turi didesnį poveikį galutiniam rezultatui.

„Training“ duomenys sudaro poros (x_i,y_i), $i = 1,2,...,N$, kur x_i – input ir y_i – output kintamieji. Duomenų modifikavimas kiekviename „boosting“ žingsnyje įvykdomas pritaikant svorius $w_1, w_2,...,w_N$ kiekvienai porai (x_i,y_i), $i = 1,2,...,N$. Iš pradžių visi svoriai prilyginami $w_i = 1/N$, todėl pirmas žingsnis „moko“ klasifikatorių kaip ir paprastai (t.y. neatsižvelgiant į svorius). Kiekvienai sekančiai iteracijai $m = 2,3,...,M$ poru (x_i,y_i) svoriai yra individualiai modifikuojami ir klasifikavimo algoritmas pritaikomas „svertoms“ poroms (x_i,y_i). Svoriai padydėja toms poroms, kurios buvo klaidingai suklasifikuotos, tuo metu kai teisingai suklasifikuotų porų svoriai sumažėja. Tokiu budu, kai iteracijos tęsiaisi, tom porom kurias sunku teisingai klasifikuoti, suteikiama didesnė įtaka. T.y. kiekvienas sekantis klasifikatorius priverstas sukoncentruoti savo dėmesį į tas poras, kurios buvo klaidingai klasifikuotos prieš tai einančiais klasifikatoriais.

Algoritmo AdaBoost.M1 žingsniai:

1. Inicializuoti porų (x_i,y_i) svorius $w_i = 1/N$, $i = 1,2,...,N$.
2. Kiekvienai iteracijai $m = 1$ iki M:
   (a) Pritaikyti klasifikatoriu $G_m(x)$ “training” duomenims naudojant svorius w_i
   (b) Apskaičiuoti:

$$\text{err}_m = \left(\sum_{i=1}^{N} w_i I(y_i \neq G(x_i)) \right) / \left(\sum_{i=1}^{N} w_i \right)$$

   (c) Apskaičiuoti: $\alpha_m = \log((1-\text{err}_m)/\text{err}_m)$
   (d) Nustatyti $w_i = w_i \exp[\alpha_m I(y_i \neq G_m(x_i)]$, $i = 1,2,\ldots,N$

3. Galutinis įvertinimas $G(x) = \text{sign}\left(\sum_{m=1}^{M} \alpha_m G_m(x)\right)$.

„AdaBoost.M1“ algoritmas taip pat yra žinomas kaip „Diskretus AdaBoost“, nes bazinis klasifikatorius $G_m(x)$ gražina diskrečią reikšmę. Jei bazinis (silpnasis) klasifikatorius gražina realias reikšmes (pvz. iš intervalo [-1,1]) „AdaBoost“ gali būti modifikuotas į „Real AdaBoost“ algoritmą (Friedman 2000).

Kaip baziniai klasifikatoriai dažniausiai naudojami klasifikavimo medžiai.

Darbo aprašymas ir rezultatai

Tam kad išsamiai ištirti AdaBoost algoritmo tinkamuma ivairiems duomenims buvo realizuotos 2 programos:

- „*AdaBoost*“ ir
- „*RealAdaBoost*“

Abejuose programuose realizuoti 3 klasifikavimo algoritmai: vieno artimiausio kaimyno algoritmas, klasifikavimo medis bei AdaBoost.

„*AdaBoost*“ programa leidžia pamatyti grafinę duomenų interpretaciją. Duomenis vaizduojami taškais XY koordinatėse. Po klasifikatoriaus paleidimo galime matyti klasifikavimo rezulatato sritys. Ieities(input) duomenų parametrų skaičius lygus dviem (ir atitinka XY koordinates). Išeitis(output) irgi lygu dviem(ir atitinka skirtingas spalvas).

„*RealAdaBoost*“ duomenis vaizduojami lentelėse. Galime keisti input parametrų skaičių, kaip ir klasifikavimo rezulatatų (output) skaičių.

„*RealAdaBoost*“ [Ieitys(input) – 13 parametrų, Išeitis(output) – 9]

Klasifikatorius	Test Error	Training Error
Vieno artimiausio kaimyno	0.70-0.75	0.02-0.1
Klasifikavimo medis	0.67-0.69	0.59-0.60
AdaBoost (10 iter.)	0.66-0.68	0.57-0.58
Atsitiktinis spejimas	0.88	

„*AdaBoost*“ [Ieitys(input) – 2 parametrai, Išeitis(output) – 2]

Klasifikatorius	Test Error	Training Error
Vieno artimiausio kaimyno	0.4-0.5	0.02-0.1
Klasifikavimo medis	0.3-0.4	0.2-0.3
AdaBoost (~30 iter.)	0.2-0.25	0.1-0.2
Atsitiktinis spejimas	0.5	

Rezultati yra apskaičiuoti vidutiniškai ir gali labai skirtis priklausomai nuo duomenų. Tai labai gerai matosi „AdaBoost“ programos pagalba. Jei pradiniai (training) duomenis yra išdėstomi „teisingai“ t.y. išdėstant vienos klasės duomenis greta vienas kito – t.y. vienoje srityje, tada visų trijų klasifikatorių rezultatai žymiai pagerėja.

Išanalizavęs rezultatus siūličiau AdaBoost algoritme kaip bazinį klasifikatorių naudoti kitą klasifikatorių vietoj klasifikavimo medžio. Nors Breiman (NIPS Workshop, 1996) teigia kad AdaBoost su medžiais yra geriausias klasifikatorius šiuo metų siūlomas[1] 293psl.

Vidutiniškai iteracijų skaičius reikalingas AdaBoost algoritme yra nuo iki 10 iki 30, nes AdaBoost klasikifatorių klaidą artėja link 0.5 ribos („*AdaBoost*“ atvėjų) bei 0.88 ribos „*RealAdaBoost*“.

Didėjant iteracijom *test error* išlieka tas pats *training error* mažėja.
Internete radau sekančias AdaBoost algoritmo realizacijas:
http://www1.cs.columbia.edu/~freund/adaboost/
http://www.cs.technion.ac.il/~rani/LocBoost/index.html
Rezultatai šiuose realizacijuose sutampa su rezulatais *„AdaBoost“* programoje.

Naudota literatūra

[1] Friedman J., Hastie T., Tibshirani R. „The elements of statistical learning“ (2001)
[2] Yoav Freund, Robert E.Schapire „A Short Introduction to Boosting“