Pēteris Račinskis
pr20015

Mājas darbs 2.6: Meta-algoritmi

Uzdevums 1

a. meta-Vote datu kopai "Diabetes"

Lai varētu izmantot *meta-Vote* algoritmu, nepieciešams izvēlēties bāzes klasifkatorus. Tā kā ir pieejami ļoti daudzi un dažādi klasifikatori, kuru precizitāte katrai konkrētai datu kopai var ievērojami atšķirties, katram klasifikatoram var būt vairāki parametri, kas katrs var pieņemt potenciāli bezgalīgi daudzas vērtības, turklāt pēc tam iegūtos rezultātus ir iespējams kombinēt vairākos veidos, ir grūti apgalvot, ka varētu kaut cik ticami vērtēt *meta-Vote* algoritma precizitāti kā funkciju no izmantoto algoritmu skaita. Teorētiski to varētu darīt, atrodot katram klasifikatoram labākos parametrus (vai vairākus kaut kādā ziņā labus, bet, ja iespējams, dažādus, parametru vektorus) dotajai datu kopai un tad caurskatot visas iespējamās klasfikatoru kombinācijas katram no bāzes modeļu skaitiem, taču tas nav praktiski izdarāms.

Līdz ar to nav īpaši lielas jēgas censties dziļi sistemātiski izvēlēties bāzes algoritmus - sasniegtie gala rezultāti principā ir stohastiska procesa rezultāts un tikai ļoti aptuveni sniedz priekštatu par *meta-Vote* lietderīguma sakarību ar izmantoto algoritmu skaitu. Dažādās klasifkatoru rezultātu kombinēšanas metodes pārsvarā ļoti lielu atšķirību rezultātos nesniedz, taču šķiet, ka *majority vote* labāk tiek galā ar situācijām, kad kāds no bāzes klasifikatoriem netiek galā ar uzdevumu - pārējās kopējais rezultāts parasti izskatās zemāks. Tāpēc arī tā ir izmantota. Kā bāzes modeļi citi meta-algoritmi izmantoti nav tikuši. Sākot ar 3 klasifikatoriem un katrā solī pieliekot jaunu:

- Pirmie trīs - SMO (SVM) ar RBF kerneli, gamma = 0.2; IBk (knn) ar k = 3; *NaiveBayes* - izvēlēti kā, autoraprāt, krietni atšķirīgi un līdz ar to potenciāli balsošanas procesā dažādi "domājoši" varianti. Modeļu parametri izvēlēti, datu kopai apmācot katru atsevišķi un atrodot šķietami labu rezultātu (visi ~70-77%). Intuitīvi šķiet, ka izvēloties līdzīgus algoritmus - piemēram, dažādus klasifkatorus, kas izmanto tuvākos kaimiņus kā k-nn ar nelielām variācijām, vai kas kaut kādos globālos, sarežģītos kontūros šķeļ Eiklīda telpu kā SVM - reāli balsošanā tiktu vairākas reizes atkārtoti ļoti līdzīgi rezultāti ar visiem tiem pašiem trūkumiem. Iegūtā precizitāte treniņa kopai ir 78.125%, krosvalidācijā 77.7344%.
- 4. izvēlēts lēmumu koks J48, jo, kaut gan plašos vilcienos varētu uzvesties nedaudz kā SVM un tāpēc pirmajā solī tika iešķirots pie

telpas ģeometriskajiem šķēlējiem, tas tomēr ir ļoti atšķirīgs. Izmantoti noklusējuma parametri, jo neko īpaši labāku atrast nav izdevies. Apvienotajam klasifikatoram - visai treniņa kopai precizitāte 80.599%, krosvalidācijā 78.2552%.

- 5. pievienots *logistic regression* algoritms, kas nav pazīstams, taču ātri uzmetot aci *Wikipedia* rakstam, šķiet, veic lineāru regresiju modelim, kas ir logaritmiski saistīts ar datu kopu, un klasifikācijā tiek izmantots nelineārā formā. Tātad vispārīgi kaut kas aptuveni līdzīgs SVM, taču pietiekami atšķirīgs, lai nebūtu uzreiz pilnībā saprotams. Parametrus mainot precizitāti acīmredzamos veidos izmainīt neizdodas, tāpēc izmantots ar noklusējuma konfigurāciju. Apvienotajam klasifikatoram - visai treniņa kopai precizitāte 79.9479%, krosvalidācijā 78.125%. Tātad sliktāk nekā iepriekš. Intereses pēc izmetot pa vienam no iepriekšējiem klasifikatoriem un salīdzinot rezultātus krosvalidācijā:
  - J48 - 77.2135%
  - SVM - 76.9531%
  - IBk - 77.6042%
  - NaiveBayes - 77.8646%

  Tātad šķiet, ka pievienojot šo klasifikatoru iepriekšējiem nekādus uzlabojumus vienkārši nevar sasniegt.
- 6. pievienots K* algoritms, ko piedāvā paši Weka izstrādātāji. Tas ir principā analogs k-nn tipa algoritmiem, bet izmanto kaut kādi ar entropiju saistītu kaimiņu distances mēru. Visu treniņa kopu tagad var klasificēt ar 83.8542% precizitāti, bet krosvalidācijā rezultāts krities uz 77.6042%.
- 7. un pēdējais pievienots neironu tīkls (*multilayer perceptron*), jo tādu iespēju Weka piedāvā. Visu parametru izvēle atstāta Weka heiristiku ziņā, un rezultātā - visai kopai 83.4635%, krosvalidācijā - 77.9948%.

Secinājumus izdarīt īpaši liela pamatojuma nav, kā jau skaidrots augstāk, taču galvenā atziņa, ko varētu gūt, ir tas, ka balsošanas metode sevišķus uzlabojumus pār katra atsevišķā klasifikatora precizitāti nesniedz, vismaz šai datu kopai.

Uzdevums 2

a. RandomForest - iterāciju skaits

Pēc noklusējuma - iterāciju skaits = 100. Treniņa datu kopai klasifikators pielāgots ar 100% precizitāti, taču atkārtojot eksperimentu ar iterāciju skaitu = {20, 50, 100, 200, 500} krosvalidācijas rezultāti ir sekojošie:

- 20, 74.8698%;
- 50, 75.7813%;
- 100, 75.7813% (jā, tā nav rakstu kļūda, rezultāti ir identiski);
- 200, 75.651% (par vienu instanci sliktāk);
- 500, 75.7813% (atkal tas pats, 582 pareizi).

Kā redzams, krosvalidācijā algoritms konverģē jau pie 50 iterācijām. kaut gan treniņa kopā vienu no 768 instancēm klasificē nepareizi. Interesanti, ka pie 200 rezultāts krosvalidācijā ir sliktāks, kā abpus tam esošām vērtībām. Treniņa kopa, protams, apgūta 100% precīzi.

b. RandomForest - nejaušo izlašu izmēri

Līdzīgā veidā var mainīt nejaušās izlases izmēru = {10%, 20%, 40%, 60%, 80%, 100%}. Mazākām izlasēm ir grūtāk pilnīgi apgūt treniņa datu kopu, bet rezultāti krosvalidācijā:

- 10%, 75.9115%;
- 20%, 76.3021%;
- 40%, 75.5208%;
- 60%, 75.9115%;
- 80%, 75.520%;
- 100%, 75.7813%

Labākais rezultāts sasniegts pie 20%, taču tas ir tikai par 4 instancēm no 768 pārāks par 100% variantu, tāpēc ar šo datu korpusu nevarētu teikt, ka šim parametram ir liela nozīme.

c. RandomForest - lēmumā izmantoto atribūtu skaits

Pēc noklusējuma 8-dimensionālai datu kopa tiek izmantoti 4 atribūti. Mainot diapazonā {1,2,4,8}:

- 1, 76.1719%;
- 2, 76.8229%;
- 4, 75.7813%;
- 8, 75.1302%.

Rezultāti ir nedaudz labāki pie mazākiem atribūtu skaitiem. Arī pie *numFeatures*=1 treniņa datu kopa tiek pilnīgi apgūta. Taču atšķirības ir nelielas, un šī datu kopa jau daudzas reizes sevi pierādījusi kā klasifikatoriem izteikti nedraudzīga - līdz ar to nevarētu pārliecinoši apgalvot, ka pieņemtā heiristika atribūtu skaita izvēlei noteikti ir nepareiza.