Универзитет „Св. Кирил и Методиј“ – Скопје
Факултет за информатички науки и компјутерско инженерство

Интерактивен ансамбл на две нивоа за бинарна класификација со кумулативна колаборација

- Дипломска работа -

Ментор:
вон. проф. д-р Ласко Баснарков

Кандидат:
Мартин Павловски, 115048

Скопје, Октомври 2015

Abstract

The development of advanced algorithms for training classification models, which can autonomously make predictions related to certain problems, has been a big challenge in the field of machine learning in the past two decades. Combining them in order to obtain more general, but at the same time accurate predictions, has taken this discipline to a whole new level. Such combinations are known as committees or ensembles. One successful technique to realize the idea of classification ensembles involves successive training of weak learners, thus reducing bias, primarily, and also, variance. This technique is known as boosting.

In order to push this idea a little further, collaboration between these ensembles can be employed. With a coherent decision-making strategy taken into account, this approach can lead towards the creation of bi-level interactive ensemble. This diploma thesis proposes a new construction method and a learning algorithm for this kind of a complex ensemble. Additionally, it focuses on how the underlying base ensembles can improve their quality by cumulatively collaborating with each other and how can they reach a joint compromise. The experimental results show that these principles contribute to the reduction of the generalization error of the model.

Keywords: Machine Learning, collaborative classification, bagging, boosting, ensemble, combining models

Резиме

Развојот на напредни алгоритми за обука на модели за класификација, коишто можат самостојно да вршат предвидувања во врска со одредени проблеми, е голем предизвик во областа на машинско учење во изминатите две децении. Нивното комбинирање со цел да се постигнат поопшти, но во исто време точни предвидувања, ја има донесено оваа дисциплина на сосема ново ниво. Ваквите комбинации од модели се познати како комитети или ансамбли. Една успешна техника за реализација на идејата за ансамбли од класификатори вклучува последователна обука на слаби класификатори на начин кој првенствено ја намалува пристрасноста и варијансата. Оваа техника е позната под името boosting.

Со цел да се доведе оваа идеја малку подалеку, можно е да се примени соработка помеѓу овие ансамбли. Земајќи во предвид кохерентна стратегија за одлучување, овој пристап може да доведе до создавање на интерактивен ансамбл на две нивоа. Овој дипломски труд предлага нов метод за конструкција и алгоритам за учење на сложен ансамбл од ваков тип. Дополнително, трудот е фокусиран на тоа како базните ансамбли кумулативно соработувајќи помеѓу себе можат да го зголемат својот квалитет и како истите можат да постигнат заеднички компромис. Експерименталните резултати покажуваат дека примената на овие принципи допринесува за намалување на генерализациската грешка на моделот.

Клучни зборови: Машинско учење, колаборативна класификација, bagging, boosting, ансамбл, комбинирање на модели

Содржина

1 Вовед

Познато е дека математичкото моделирање како начин за опис на даден систем наоѓа голема примена во природните научки, инженерските дисциплини, како и социјалните науки. Природата на голем дел од проблемите кои се изучуваат во рамките на овие области ја наметнува потребата од развој на интелигентни модели кои би биле способни да учат, да создадат одредени концепти користејќи го знаењето кое го стекнале и користејќи ги истите на соодветен начин да вршат предвидувања во врска со одредени аспекти на проблемот за кој се наменети или пак во врска со идното однесување на системот кој го опишуваат.

Прашањето како да се изградат модели кои би биле способни да се однесуваат интелигентно или во поширока смисла да делуваат интелигентно е главниот двигател на вештачката интелигенција до 1990-тите години. Веднаш по овој период машинското учење, препознаено како посебна гранка на вештачката интелигенција, почнува поинтензивно да се развива. Имено, оваа подобласт ја видоизменува целта на вештачката интелигенција, ориентирајќи се кон справување со решливи проблеми од практична природа.

Во рамките на воведот е наговестена хипотезата за подобрувањето на перформансите при решавање на проблеми кои вклучуваат одредени предвидувања во случај на комбинирање на ансамбли од модели и овозможување на соработка помеѓу истите.

1.1 Комбинирање на несовршени модели

Моделите кои нецелосно опишуваат даден систем засебно не би можеле да извршуваат целосно точни предвидувања во врска со однесувањето на истиот. Од друга страна пак, доколку овие модели совршено добро го опишуваат системот, можно е предвидувањата направени со помош на истите да бидат премногу пристрасни кон примероците кои тие ги опсервирале. Таков пример се статистичките модели чии предвидувања е можно да станат премногу пристрасни кон податоците кои им се на располагање.

Идејата за комбинирање на несовршени модели е развивана и во области како климатологијата. Имено, во [1] е презентирана можноста да се искомбинираат повеќе климатски модели, кои можат континуирано да разменуваат информации помеѓу себе во текот на процесот на учење и предвидување, во таканаречени супермодели со цел да се засили способноста за симулација на опсервираната, историска еволуција на климата и да се добијат пореални проценки за идните климатски промени. Процесот на градење на овие модели се нарекува супер-моделирање.

Она што се претпоставува е дека оваа замисла може да биде прилагодена така

што ансамбли од модели кои комуницираат би можеле заедно да ги намалат нивните поединечни грешни одлуки притоа намалувајќи го бројот на грешни предвидувања на ансамблот кој го сочинуваат.

1.2 Учење со помош на ансамбли

Учењето со помош на ансамбли вклучува методи кои користат низа на алгоритми за учење на модели со цел да се добијат подобри перформанси при процесот на предвидување, кои инаку не би можеле да се постигнат поединечно од страна на некој од составните модели. Имено, учењето на еден ансамбл вклучува обука на секој од составните модели на истиот, додека носењето на финалната одлука се базира на постигнување компромис помеѓу моделите. Методот кој ќе биде предложен подоцна во текстот го опишува процесот на обука на сложен ансамбл составен од ансамбли од несовршени модели, како и начинот на кој истиот може да биде искористен за носење на одлуки.

1.3 Колаборативна класификација

Како што беше споменато погоре во текстот, моделите во рамки на еден ансамбл можат да соработуваат на тој начин што ќе комуницираат помеѓу себе. Во случај кога се работи за модели за класификација на одредени ненабљудувани примероци, истата се спроведува колаборативно. Имено, колаборативна класификација претставува класификација која се должи на соработката помеѓу група на модели од ваков тип. Класификацијата како процес е дефинирана и подетално објаснета во рамки на делот кој ги содржи теоретските основи за предложениот метод.

Поимот за колаборативна класификација може да биде обопштен со воспоставување на комуникација помеѓу низа ансамбли што би резултирало со формирање на сложен модел за колаборативна класификација базиран на интерактивен ансамбл на две нивоа.

2 Теоретски основи

2.1 Класификација

Класификацијата претставува процес на идентификација на припадноста на дадена опсервација во одредена категорија со помош на откривање на потенцијални релации помеѓу опсервации за кои се познати категориите во кои припаѓаат истите.

Нека со d-димензионалниот вектор $\mathbf{X} = (X_1, X_2, \ldots, X_d) \in \mathbb{R}^d$ е претставена една опсервација и нека со C е означена категоријата или класата на која припаѓа истата. Ако, $R_C = \{c_1, c_2, \ldots, c_m\}$ е множеството вредности кои може да ги прими C така што $P\{C = c_k\} \geq 0$ и $\sum_{k=1}^{m} P\{C = c_k\} = 1$, тогаш за дадени реални броеви $x_1, x_2, \ldots, x_d$, важи дека

$$P\{C = c_k | X_1 = x_1, X_2 = x_2, \ldots, X_d = x_d\} \tag{1}$$

е постериорната веројатност дека опсервацијата чиј вектор на вредности е $\mathbf{x} = (x_1, x_2, \ldots, x_d)$ припаѓа на класата c_k, каде $k = 1, \ldots, m$. Изразот (1) може да се запише и како $P\{C = c_k | \mathbf{X} = \mathbf{x}\}$.

Дефиниција 1. Класификација на дадена опсервација претставува одбирање на класа $c \in R_C$ така што

$$P\{C = c | \mathbf{X} = \mathbf{x}\} = \max_k P\{C = c_k | \mathbf{X} = \mathbf{x}\} \ , \tag{2}$$

каде $k = 1, \ldots, m$. Во случај кога $m = 2$ велиме дека станува збор за бинарна или биномна класификација. Во спротивно за $m > 2$, класификацијата се нарекува повеќекласна.

Дефиниција 2. Алгоритам кој имплементира класификација, особено ако се работи за конкретна имплементација на процесот на класификација, се нарекува класификатор. Терминот "класификатор" понекогаш се однесува и на математичко пресликување $f : \mathbb{R}^d \to R_C$, имплементирано од страна на алгоритам за класификација, кое ја пресликува секоја податочна инстанца $\mathbf{x}$ на влез во ознаката на класата на која припаѓа.

За класификатор кој спроведува бинарна класификација, велиме дека е бинарен.

Нека $\mathcal{D}_{train}$ е множество чии инстанци претставуваат вектори од вредности на опсервации $\mathbf{x}_i = (x_1^{(i)}, x_2^{(i)}, \ldots, x_d^{(i)}), i = 1, \ldots, N$. Ако за секоја инстанца $\mathbf{x}_i$ е позната класата со која е означена истата, тогаш множеството $\mathcal{D}_{train}$ се нарекува множество за обука на одреден модел за класификација. Спротивно, ако пак инстанците во множествата $\mathcal{D}_{validation} \subset \mathcal{D}_{train}$ и $\mathcal{D}_{test} (\mathcal{D}_{test} \cap \mathcal{D}_{train} = \emptyset)$ не се претходно набљудувани

и нивните класи се непознати, тогаш $\mathcal{D}_{validation}$ се нарекува множество за валидација, а $\mathcal{D}_{test}$ се нарекува множество за тестирање на моделот за класификација. Именно, еден статистички модел за класификација се обучува со помош на инстанците од множеството за обука, а потоа се тестираат неговите перформанси така што се врши проверка на неговата способност да ги предвиди класите со кои се означени инстанците од множеството за тестирање врз база на претходно извршената обука. Користењето на валидациско множество во фазата на обука е опционално.

Дефиниција 3. Нека е дадено множеството $\mathcal{D}_{train} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{train}}\}$ и соодветниот вектор на одговори $\mathbf{y} = (y_1, y_2, \ldots, y_{N_{train}})$, така што y_i е ознаката на класата со која е означена инстанцата $\mathbf{x}_i$, за секое $i = 1, \ldots, N_{train}$. Ако $\mathcal{D}_{train}$ е множеството со кое се обучува одреден класификатор, тогаш неговата грешка при обука се евалуира на следниот начин

$$\epsilon_{train} = \frac{\sum_{i=1}^{N_{train}} I(f(\mathbf{x}_i) \neq y_i)}{N_{train}} \ , \tag{3}$$

каде $I(A)$ претставува индикатор случајна променлива која може да прими една од вредностите 1 или 0 во зависност од тоа дали дадениот случаен настан A е вистинит или не, соодветно.
Аналогно, грешката ϵ_{test} на класификаторот на дадено множество $\mathcal{D}_{test}$, чии инстанци не се набљудувани од страна на истиот, се нарекува грешка при тестирање на класификаторот, позната и како генерализациска грешка.

Дефиниција 4. Нека Y е случајна променлива која го претставува излезот на даден класификатор за произволна инстанца на влез. Ако класификаторот ги третира сите класи $c_1, c_2, \ldots, c_m$ еднаквоверојатно т.е. ако за Y важи дека

$$P\{Y = c_k\} = \frac{1}{m} \ , \tag{4}$$

за секое $k = 1, \ldots, m$ односно

$$Y \sim U\big(\{c_1, c_2, \ldots, c_m\}\big) \ , \tag{5}$$

тогаш ваквиот класификатор го нарекуваме случаен класификатор или случаен погодувач.

Теорема 1. *Очекуваната вредност на грешката при обука на случаен класификатор за бинарна класификација е* $\frac{1}{2}$.

Доказ: Нека Z е случајна променлива која го претставува бројот на погрешно класифицирани инстанци од множеството за обука $\mathcal{D}_{train} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{train}}\}$, од страна на случаен класификатор. Ако е даден векторот со класите на кои припаѓаат истите

$\mathbf{y} = (y_1, y_2, \ldots, y_{N_{train}})$, $y_i \in \{-1, 1\}$ и ако со Y_i го означиме излезот на случајниот класификатор кога на влез е проследена инстанцата $\mathbf{x}_i$, за $i = 1, \ldots, N_{train}$, тогаш

$$Z = \sum_{i=1}^{N_{train}} I(Y_i \neq y_i) \ . \tag{6}$$

Можеме да забележиме дека равенството (6) дефинира Бернулиева шема од N_{train} експерименти од каде следува дека

$$Z \sim B(N_{train}, p) \ , \tag{7}$$

каде $p = P\{Y_i \neq y_i\} = \frac{1}{2}$, за секое $i = 1, \ldots, N_{train}$. Соодветно, грешката при обука

$$\frac{Z}{N_{train}} \sim B(1, p) \ . \tag{8}$$

Оттука, очекуваната вредност на грешката при обука на слабиот класификатор е

$$E\left[\frac{Z}{N_{train}}\right] = p = \frac{1}{2} \ . \tag{9}$$

Дефиниција 5. Слаб класификатор (или слаб "ученик") се нарекува оној класификатор чија грешка на обука ϵ_{train} е незначително помала, но сепак помала, од онаа на случајниот класификатор т.е.

$$\epsilon_{train} < \frac{1}{2} \ . \tag{10}$$

Спротивно на тоа, силен класификатор е оној класификатор чии предвидувања за класите се во голема корелација со вистинските класи на инстанците од множеството за обука.

2.2 Теорија на ансамбли

Дефиниција 6. Од аспект на машинско учење, за група на алгоритми кои имплементираат модели за класификација велиме дека претставува комитет или ансамбл доколку истата може да биде обучена така што ќе биде способна да врши предвидувања кои ги инкорпорираат предвидувањата на секој од составните модели.

Еден ансамбл по своето обучување претставува единствена хипотеза. Сепак, не значи дека оваа хипотеза е содржана во рамки на просторот на хипотези на моделите од кои е изградена истата. Поради тоа, се покажува дека ансамблите можат да имаат поголема флексибилност во однос на функциите кои ги претставуваат.

Класичен проблем на мнозинско гласање:
Да претпоставиме дека $n \geq 2$ независни индивидуи одлучуваат по пат на мнозинско гласање. Ако секоја индивидуа може да донесе или правилна или погрешна одлука и ако истата со веројатност p ја носи правилната одлука, тогаш веројатноста дека мнозинската одлука ќе биде онаа правилната може да биде пресметана користејќи Бернулиева шема т.е.

$$P_k(n) = \sum_{i=k}^{n} \binom{n}{i} p^i (1-p)^{n-i} \ , \tag{11}$$

каде вредноста на k се пресметува на следниот начин

$$k = \begin{cases} \frac{n}{2} + 1, & \text{ако } n \text{ е парен број} \\ \frac{n+1}{2}, & \text{ако } n \text{ е непарен број} \end{cases} . \tag{12}$$

Овој проблем, неговите екстензии и примена се подетално дискутирани во [2].

Теорема 2. *(Теорема на Кондорсе) [3]: Да претпоставиме дека група од $n \geq 3$ индивидуи, каде n е непарен број, треба да направи избор помеѓу една од две опции со помош на мнозинско гласање. Само една од двете опции е точна и се претпоставува дека секоја индивидуа гласа независно од другите и непристрасно кон одредена опција. Ако p претставува веројатноста дека една индивидуа ќе го направи вистинскиот избор, тогаш потребниот број на гласачи во рамки на групата за да истата со помош на мнозинско гласање ја одбере точната опција зависи од вредноста на p, т.е.*

1. *Ако $p > \frac{1}{2}$, тогаш $P_k(n)$ монотоно расте со зголемувањето на вредноста на n т.е. $P_k(n) \to 1$ кога $n \to \infty$*
2. *Ако $p < \frac{1}{2}$, тогаш $P_k(n)$ монотоно опаѓа со зголемувањето на вредноста на n т.е. $P_k(n) \to 0$ кога $n \to \infty$*
3. *Ако $p = \frac{1}{2}$, тогаш $P_k(n) = \frac{1}{2}$, за секое $n = 2, 3, 4, \ldots$*

Теоремата важи и за парен број на индивидуи под претпоставка дека ако n е парен број немаме нерешена ситуација во однос на гласовите за двете опции.

Доказ: Со цел да ја избегнеме ситуацијата на еднаков број на гласови за двете опции, претпоставуваме дека n е непарен број.
Нека $n_c \leq n$ е бројот на индивидуи кои гласале точно. Да забележиме што се случува кога се зголемува бројот на индивидуи во групата. Додавање на индивидуи во групата, притоа запазувајќи вкупниот број на индивидуите во истата да остане непарен, е можно само доколку во секој чекор се додаваат по две индивидуи одеднаш. Оттука,

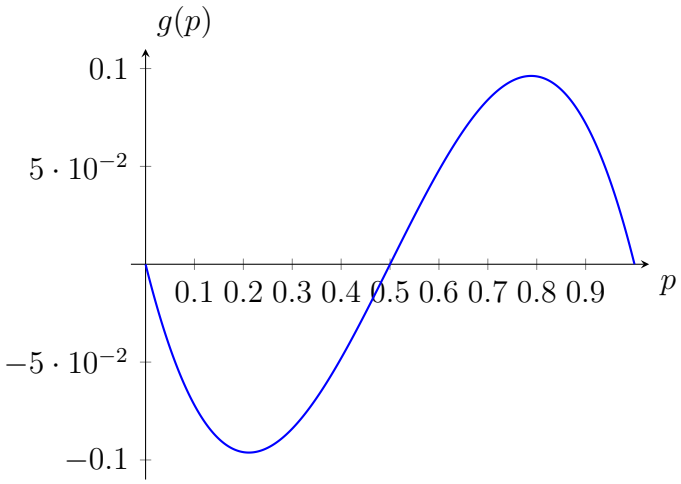
во секој таков чекор постојат два случаи за менување на мнозинскиот исход:
- ако двете нови индивидуи гласале точно, а пред нивното додавање во групата бројот на индивидуи кои гласале точно бил $\frac{n-1}{2}$
- ако двете нови индивидуи гласале неточно, а пред нивното додавање во групата бројот на индивидуи кои гласале точно бил $\frac{n-1}{2}+1$

Доколку се работи за чекор кој не опфаќа ниту еден од горенаведените случаи, тогаш сигурно се работи за случај кога едната од двете додадени индивидуи во тој чекор гласала точно, а другата неточно, или обратно, со што мнозинската одлука ќе остане непроменета.

Сега, да претпоставиме дека n-те индивидуи веќе го дале својот глас. Од првите $n-1$ индивидуи, половина гласале неточно, додека другата половина гласале точно. Во ваквата состојба, мнозинската одлука ќе биде точна само ако n-тата индивидуа гласа точно што значи дека p е веројатноста за истата да биде точна. Со додавање на две нови индивидуи во групата, веројатноста неточна мнозинска одлука да стане точна е $p_{correct} = (1-p)p^2$, додека веројатноста дека двата нови гласа ќе допринесат точна мнозинска одлука да стане неточна е $p_{incorrect} = p(1-p)^2$. Мнозинската одлука ќе биде точна ако $p_{correct} > p_{incorrect}$. Во продолжение е поедноставено неравенството.

$$\begin{aligned} (1-p)p^2 &> p(1-p)^2 \\ p^2 - p^3 &> p(1-2p+p^2) \\ p^2 - p^3 &> p - 2p^2 + p^3 \\ -2p^3 + 3p^2 - p &> 0 \ . \end{aligned} \tag{13}$$

Во продолжение графички е прикажана функцијата $g(p) = -2p^3+3p^2-p, 0 \leq p \leq 1$.

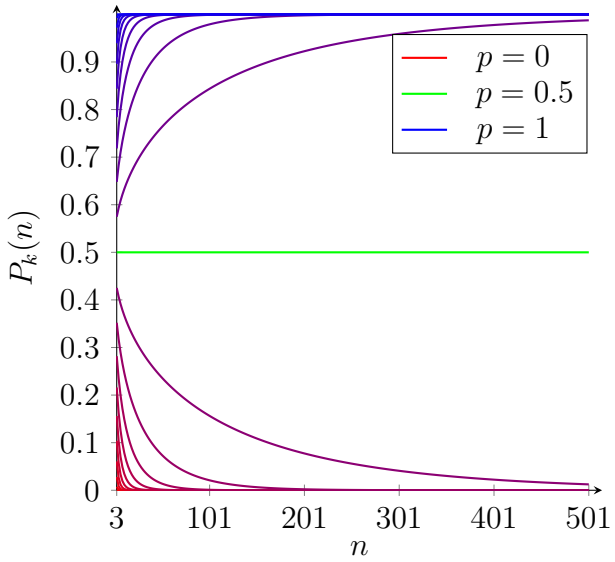


Слика 1: $g(p)$, за $p \in [0,1]$

Она што може да се забележи е дека $g(p)$ е позитивна на интервалот $\left(\frac{1}{2}, 1\right]$. Со други зборови, за $p \in \left(\frac{1}{2}, 1\right]$ веројатноста дека додавањето на две нови индивидуи во

групата ќе допринесе мнозинската одлука да биде точна е поголема од веројатноста дека истото ќе допринесе мнозинската одлука да биде неточна.

Оттука следува дека со последователно додавање на доволно голем број на индивидуи во групата, притоа додавајќи по две индивидуи во секој чекор, мнозинската одлука на групата ќе стане точна ако и само ако $p > \frac{1}{2}$.
Со ова е докажан случајот *1.* на теоремата. Симетрично се докажува и случајот *2.*, додека случајот *3.* природно следува од доказите на *1.* и *2.* .



Слика 2: Графички приказ на исказите под *1.* (со сина боја), *2.* (со црвена боја) и *3.* (со зелена боја), за $p = 0, 0.1, 0.2, \ldots, 1$

Последица 1. *Нека е даден ансамбл од M независни слаби бинарни класификатори кои се обучуваат на множеството $\mathcal{D}_{train} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{train}}\}$ и нека со $f_1(\mathbf{x}_i), f_2(\mathbf{x}_i), \ldots, f_M(\mathbf{x}_i)$ се означени излезите на M-те класификатори кога на влез е проследена податочната инстанца $\mathbf{x}_i$, за секое $i = 1, \ldots, N_{train}$. Ако излезот на ансамблот се базира на мнозинско гласање на составните класификатори, т.е.*

$$F(\mathbf{x}_i) = \text{sign}\left(\sum_{m=1}^{M} f_m(\mathbf{x}_i)\right) \ , \tag{14}$$

за секое $i = 1, \ldots, N_{train}$, тогаш грешката при обука на ансамблот $\varepsilon_{train} \to 0$ кога $M \to \infty$.

Доказ: Нека $\mathbf{y} = (y_1, y_2, \ldots, y_{N_{train}})$, $y_i \in \{-1, 1\}$ е вектор кој ги содржи класите на инстанците од $\mathcal{D}_{train}$. За грешката при обука на ансамблот важи следното

$$
\begin{aligned}
P\{\varepsilon_{train}=0\} &= P\{1-\varepsilon_{train}=1\} \\
&= P\left\{1-\frac{\sum_{i=1}^{N_{train}} I(F(\mathbf{x}_i)\neq y_i)}{N_{train}}=1\right\} \\
&= P\left\{\frac{N_{train}-\sum_{i=1}^{N_{train}} I(F(\mathbf{x}_i)\neq y_i)}{N_{train}}=1\right\} \\
&= P\left\{\frac{\sum_{i=1}^{N_{train}} 1-\sum_{i=1}^{N_{train}} I(F(\mathbf{x}_i)\neq y_i)}{N_{train}}=1\right\} \\
&= P\left\{\frac{\sum_{i=1}^{N_{train}} 1- I(F(\mathbf{x}_i)\neq y_i)}{N_{train}}=1\right\} \\
&= P\left\{\frac{\sum_{i=1}^{N_{train}} I(F(\mathbf{x}_i)= y_i)}{N_{train}}=1\right\} \\
&= P\left\{\left[\sum_{i=1}^{N_{train}} I(F(\mathbf{x}_i)= y_i)\right]=N_{train}\right\} \\
&= P\{F(\mathbf{x}_1)=y_1,\ldots,F(\mathbf{x}_{N_{train}})=y_{N_{train}}\} \overset{\text{нез.}}{=} \prod_{i=1}^{N_{train}} P\{F(\mathbf{x}_i)=y_i\} \\
&= \prod_{i=1}^{N_{train}}\sum_{s=k}^{M}\binom{M}{s}(P\{f_s(\mathbf{x}_i)=y_i\})^s(1-P\{f_s(\mathbf{x}_i)=y_i\})^{M-s} \ ,
\end{aligned}
\tag{15}
$$

каде вредноста на s се пресметува на следниот начин

$$
s=\begin{cases}\frac{M}{2}+1, & \text{ако } M \text{ е парен број}\\ \frac{M+1}{2}, & \text{ако } M \text{ е непарен број}\end{cases} \ . \tag{16}
$$

Без губење на општоста, на веројатноста $P\{f_s(\mathbf{x}_i)=y_i\}$ може да се гледа како на веројатност дека инстанцата $\mathbf{x}_i$ е една од точно класифицираните инстанци од страна на s-тиот слаб класификатор, а бидејќи неговата грешка при обука претставува процент на погрешно класифицирани инстанци, оваа веројатност може да се претстави како

$$
P\{f_s(\mathbf{x}_i)=y_i\}=1-\epsilon_{train}^{(s)} \ . \tag{17}
$$

Притоа, јасно е дека $P\{f_s(\mathbf{x}_i)=y_i\}>\frac{1}{2}$ бидејќи $\epsilon_{train}^{(s)}<\frac{1}{2}$, па според случајот *1.* во теоремата 2, следува дека

$$
\lim_{M\to\infty}\sum_{s=k}^{M}\binom{M}{s}(P\{f_s(\mathbf{x}_i)=y_i\})^s(1-P\{f_s(\mathbf{x}_i)=y_i\})^{M-s}=1 \ . \tag{18}
$$

Оттука, со замена се добива дека

$$\lim_{M\to\infty}\prod_{i=1}^{N_{train}}\sum_{s=k}^{M}\binom{M}{s}(P\{f_s(\mathbf{x}_i)=y_i\})^s(1-P\{f_s(\mathbf{x}_i)=y_i\})^{M-s}=$$
$$=\prod_{i=1}^{N_{train}}\lim_{M\to\infty}\sum_{s=k}^{M}\binom{M}{s}(P\{f_s(\mathbf{x}_i)=y_i\})^s(1-P\{f_s(\mathbf{x}_i)=y_i\})^{M-s}=\prod_{i=1}^{N_{train}}1=1\ . \tag{19}$$

$\Rightarrow P\{\varepsilon_{train}=0\}\to 1$ кога $M\to\infty\Rightarrow\varepsilon_{train}\to 0$ кога $M\to\infty$.

2.3 Методи за конструкција и учење со ансамбли

Во областа на машинско учење, овие методи претставуваат методи со помош на кои се конструираат ансамбли кои потоа користат низа алгоритми за учење со цел да постигнат подобри перформанси при предвидување од оние кои би се постигнале од страна на некој од составните алгоритми поединечно. За разлика од методите за учење со ансамбли од други области кои вклучуваат теоретски бесконечен број на составни модели, методите за машинско учење со ансамбли се однесуваат на групи од конечен број на модели за класификација.

Според [4] под поимот "метод за учење со ансамбл" се подразбира Баесовото усреднување, меѓутоа постојат и пософистицирани методи како што се bagging и boosting, кои вклучуваат поправка на грешката на излезот на ансамблот. Важно е да се направи разграничување помеѓу зависните рамки за градење на ансамбли кои можат да користат boosting како метод за нивна конструкција и учење и независните кои се базираат на bagging. Имено, кај зависните рамки излезот на секој класификатор се користи за градење на класификаторот кој следува во рамки на ансамблот. На тој начин знаењето стекнато во итерациите кои изминале може да биде искористено за учење во итерациите кои следат. Од друга страна пак, кај независните рамки класификаторите се независно изградени и нивните одлуки се комбинираат на соодветен начин. Ваквиот начин на поделба на рамките за градење на ансамбли е презентиран во [5].

2.3.1 Bagging (Bootstrap aggregating)

Опис на методот bagging и неговата цел

Bootstrap aggregating или само bagging претставува метод за конструкција и учење со ансамбли составен од две фази: генерирање на повеќе верзии на еден модел со земање на bootstrap примероци и агрегирање на нивните предвидувања во една конечна одлука [6]. Различните верзии на моделот се генерираат со поделба на множеството за обука на повеќе подмножества така што секое од нив добива улога на

множество за обука на една од верзиите на моделот. Овие подмножества всушност ги претставуваат bootstrap примероците.

Кај проблемите на регресија bagging-от претставува корисен метод за избор на репрезентативен примерок од дадено множество [7]. Од друга страна пак, кај проблемите на класификација, bagging-от вклучува генерирање на повеќе верзии на еден класификатор и тука клучен фактор е нестабилноста на класификаторот бидејќи ако поделбата на множеството за обука предизвикува значителни промени во предвидувањата од страна на неговите верзии, тогаш можно е подобрување на стабилноста и точноста на истиот [6]. Дополнително, bagging-от ја намалува и варијансата и помага да се избегне проблемот на преголема пристрасност кон инстанците од множеството за обука познат под името "overfitting" [8, p. 39,79].

Дефиниција 7. (Bagging) Нека е даден ансамбл составен од M модели и нека $\mathcal{D}_{train}^{(1)}$, $\mathcal{D}_{train}^{(2)}, \ldots, \mathcal{D}_{train}^{(M)}$ претставуваат примероци од множеството за обука $\mathcal{D}_{train}$ така што m-тиот составен модел се обучува на подмножеството $\mathcal{D}_{train}^{(m)}$, каде $m = 1, \ldots, M$. Излезот на ансамблот се пресметува преку усреднување на излезите на составните модели ако се работи за проблем на регресија или со помош на гласање ако се работи за проблем на класификација на инстанци чии класи имаат нумерички вредности. Процедурата која вклучува земање на примероци од множеството за обука и комбинирање на моделите кои се обучуваат на примероците се нарекува "bootstrap aggregating", позната и под името "bagging".

Од оваа дефиниција може да се забележи дека во суштина bagging-от кај проблеми на класификација претставува специјален случај на пристапот со усреднување на модели, но понатаму во текстот ќе се задржиме само на фазата од bagging-от која вклучува поделба на множеството за обука на повеќе подмножества.

Варијанти на bagging

Во зависност од начинот на земање на bootstrap примероците, нивната распределба и корелираност со множеството за обука, во продолжение се дефинирани повеќе варијанти на bagging.

- *Bagging со независно земање на примероци*
  Bagging со независно земање на примероци претставува варијанта на bagging која вклучува градење на примероци $\mathcal{D}_{train}^{(m)} \subseteq \mathcal{D}_{train} (m = 1, \ldots, M)$ со случајно влечење на инстанци од множеството за обука $\mathcal{D}_{train}$, притоа пропорционално запазувајќи ја неговата распределба кај секој од примероците. Случајното влечење на инстанците се изведува така што кога дадена инстанца ќе биде доделена во рамки на некој примерок, истата не се отстранува од множеството за обука. На овој начин се обезбедува независност помеѓу класификаторите кои ќе се обучуваат на добиените подмножества.

Влез: $\mathcal{D}_{train}$ - множество за обука
$\mathcal{L}$ - алгоритам за обука на базниот класификатор
$f : \mathbb{R}^d \to \{-1, 1\}$ кое го дефинира базниот класификатор
M - број на итерации за учење (број на класификатори)
Процедура:
за секое $m = 1, \ldots, M$:
$\mathcal{D}_{train}^{(m)} = \textit{IndependentSampling}(\mathcal{D}_{train})$ ▷ генерирање на примерок
$f_m \leftarrow \mathcal{L}(\mathcal{D}_{train}^{(m)})$ ▷ m-тиот класификатор се обучува на примерокот
Излез: $F(\mathbf{x}) = \text{sign}\Big(\sum_{m=1}^{M} f_m(\mathbf{x})\Big)$ ▷ комбинирање на излезите од верзиите на класификаторот преку гласање

Алгоритам 1: Алгоритам за bagging со независно земање на примероци [9] во случај на бинарна класификација

- *Идеален bagging* [10]
  Велиме дека bagging методот е идеален ако излезот на даден класификатор $f(\mathbf{x})$, обучен на множеството $\mathcal{D}_{train}$, кога на влез е проследена инстанца $\mathbf{x}$, може да се претстави како очекуваниот излез на неговата верзија обучена на случаен примерок $\mathcal{D}_{train}^{(m)} \subset \mathcal{D}_{train}$, т.е.

$$f(\mathbf{x}) = E[f_m(\mathbf{x})] \ , \tag{20}$$

  за секоја негова верзија т.е. за секое $m = 1, \ldots, M$.

- *Реален bagging*
  Реалниот bagging во суштина претставува конечна апроксимација на идеалниот bagging.

2.3.2 Boosting

Опис на методот boosting и неговата цел

Boosting претставува метод за инкрементално градење на ансамбл од модели преку обука на секој модел притоа нагласувајќи ги инстанците кои биле погрешно класифицирани од страна на моделите кои му претходеле. Целта на овој метод е да овозможи група од слаби класификатори да создаде еден силен класификатор. Примарното решение по ова прашање е презентирано во [11] и истото претставувало главен двигател на понатамошниот развој на овој метод.

Главната тенденција на boosting-от е, првенствено, намалувањето на пристрасноста, но исто така и на варијансата т.е. сензитивноста на мали флуктуации во множеството за обука. Во некои случаи boosting може да обезбеди поголема точност од

bagging, но сепак постои опасноста од постигнување на преголема пристрасност кон инстанците од множеството за обука, што не е случај кај bagging методот.

Алгоритми за boosting

Кога станува збор за имплементација на boosting, тогаш AdaBoost (скратен запис за Adaptive boosting) [12] претставува можеби историски најзначајниот boosting алгоритам. AdaBoost овозможува секвенцијална обука на група од M слаби класификатори кои во некои литератури може да се сретнат и под називот "базни" класификатори. Во секој чекор од алгоритамот се обучува по еден слаб класификатор на тој начин што истиот се стреми точно да ги предвиди класите на инстанците кои биле погрешно класифицирани од претходно обучените класификатори во секвенцата. Овие инстанци се инстанците кои во тој чекор имаат најголеми тежини. Имено, за дадено множество за обука $\mathcal{D}_{train} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{train}}\}$, на секоја инстанца $\mathbf{x}_i$ од истото и се доделува тежина ω_i, за секое $i = 1, \ldots, N_{train}$ и секој од класификаторите после својата обука ги менува овие тежини со помош на соодветен механизам за ажурирање на тежините и инстанците со така променетите тежини му ги проследува на следниот класификатор во секвенцата. Дополнително, по обучувањето на m-тиот класификатор се пресметува и неговата значајност α_m како функција од неговата грешка при обуката ϵ_m, за секое $m = 1, \ldots, M$. На самиот крај, по обучувањето на целата секвенца, се донесува конечната одлука на силниот класификатор како тежинска сума од излезите на слабите класификатори.

Негативност на AdaBoost е неговата сензитивност на шум и outlier-и, додека негова предност е тоа што во одредени случаи е помалку подложен на overfitting во споредба со други алгоритми за учење иако грешката при обука на целиот ансамбл $\varepsilon_{train} \to 0$ после одреден број на итерации M. Уште повеќе, во [12, p. 14] е докажано дека ε_{train} ќе добие вредност 0 експоненцијално брзо, односно после многу мал број на итерации.

Дополнителна предност на AdaBoost е можноста неговите перформанси значително да се подобрат во случај кога како базни класификатори во рамки на ансамблот се користат дрва на одлучување. Подетален опис на предностите и недостатоците на AdaBoost може да се најде во [13, p. 5] и [14, p. 658].

Покрај стандардниот AdaBoost, во зависност од начинот на кој се обезбедува boosting-от при процесот на обука, постојат повеќе алгоритми за постигнување на оваа цел. Може да се каже дека на секој од овие алгоритми може да се гледа како на одредена варијанта на првоопишаниот AdaBoost. Некои од овие алгоритми како што се AdaBoostM1 (во случај на две класи), LogitBoost, GentleBoost и RobustBoost се наменети за бинарна класификација, од друга страна пак AdaBoostM2 е наменет за повеќекласна класификација, додека постојат и некои алгоритми наменети и за двата типа на класификација, меѓу кои се вбројуваат LPBoost, TotalBoost, RUSBoost

и други. Во продолжение ќе биде ставен акцент само на дел од boosting алгоритмите наменети за бинарна класификација.

- *AdaBoost M1*

  AdaBoostM1 претставува прва екстензија на стандардниот AdaBoost. Во суштина, разликите помеѓу проширената верзија и стандардниот AdaBoost се занемарливи. Главната разлика е во начинот на пресметка на грешката при обука на секој базен класификатор. Имено, според AdaBoost пресметувањето на грешката при обука на m-тиот класификатор вклучува пресметување на апсолутната разлика $|f_m(\mathbf{x}_i) - y_i|$, додека кај AdaBoostM1 за таа цел се користи изразот $I(f_m(\mathbf{x}_i) \neq y_i)$, за секоја инстанца $\mathbf{x}_i \in \mathcal{D}_{train}, i = 1, \ldots, N_{train}$. Исто така, за дадена инстанца на влез, во случај кога се користи AdaBoostM1, излезот на ансамблот претставува ознака на класа која ја максимизира тежинската сума од излезите на базните класификатори за таа инстанца.

Влез: $\mathcal{D}_{train} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{train}}\}$ - множество за обука
$\mathbf{y} = (y_1, y_2, \ldots, y_{N_{train}}) \in \{-1, 1\}^{N_{train}}$ - вектор на одговори
D - распределба на инстанците во $\mathcal{D}_{train}$
$\mathcal{L}$ - алгоритам за обука на слаб класификатор
M - број на итерации (број на класификатори)

Иницијализација: $\omega_i^{(1)} = D(i), i = 1, \ldots, N_{train}$ ▷ иницијализација на тежините на инстанците

Процедура:
  за секое $m = 1, \ldots, M$:
    $\mathbf{p}^{(m)} = \mathbf{w}^{(m)} / \sum_{i=1}^{N_{train}} \omega_i^{(m)}$
    $f_m \leftarrow \mathcal{L}(\mathcal{D}_{train}, \mathbf{p}^{(m)})$ ▷ обука на m-тиот класификатор со $\mathcal{L}$, притоа земајќи ја во предвид распределбата $\mathbf{p}^{(m)}$
    $\epsilon_m = \sum_{i=1}^{N_{train}} p_i^{(m)} I(f_m(\mathbf{x}_i) \neq y_i)$ ▷ пресметување на грешката при обука на m-тиот класификатор
    ако $\epsilon_m > (1/2)$ **тогаш**
      $M = m - 1$
      терминирање на циклусот
    инаку
      $\beta_m = \epsilon_m / (1 - \epsilon_m)$
      за секое $i = 1, \ldots, N_{train}$:
        $\omega_i^{(m+1)} = \omega_i^{(m)} {\beta_m}^{1 - I(f_m(\mathbf{x}_i) \neq y_i)}$ ▷ ажурирање на тежините

Излез: $F(\mathbf{x}) = \text{sign}\Big(\sum_{m=1}^{M} \ln(1/\beta_m) f_m(\mathbf{x}) \Big)$, за секое $\mathbf{x} \in \mathbb{R}^d$

Алгоритам 2: AdaBoostM1 во случај на бинарна класификација [12, p. 21]

- *LogitBoost*

  LogitBoost претставува варијанта на AdaBoost која користи логистичка регресија [15, Ch. 6] за пресметување на излезот на секој базен класификатор. Дополнително, тежинската грешка на секој класификатор не се минимизира така што бројот на инстанците кои биде точно класифицирани од негова страна да биде максимален. Имено, m-тиот класификатор се одбира на таков начин што истиот ја минимизира Њутн-Рафсоновата апроксимација на грешката на подобност. Повеќе информации во врксa со алгоритамот LogitBoost се презентирани во [16].

Влез: $\mathcal{D}_{train} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{train}}\}$ - множество за обука
$\mathbf{y} = (y_1, y_2, \ldots, y_{N_{train}}) \in \{-1, 1\}^{N_{train}}$ - вектор на одговори
Иницијализација: $\omega_i = 1/N_{train}, i = 1, \ldots, N_{train}$ ▷ униформна иницијализација на тежините на инстанците
$G(\mathbf{x}) = 0$
$p(\mathbf{x}_i) = 1/2, i = 1, \ldots, N_{train}$
Процедура:
  за секое $m = 1, \ldots, M$:
    за секое $i = 1, \ldots, N_{train}$:
      $z_i = (y_i - p(\mathbf{x}_i))/(p(\mathbf{x}_i)(1 - p(\mathbf{x}_i)))$
      $\omega_i = p(\mathbf{x}_i)(1 - p(\mathbf{x}_i))$ ▷ ажурирање на тежините на инстанците
      Одреди го $f_m(\mathbf{x})$ со помош на тежинска least-squares регресија на z_i како функција од $\mathbf{x}_i$, користејќи ги тежините ω_i
      $G(\mathbf{x}) \leftarrow G(\mathbf{x}) + (1/2) f_m(\mathbf{x})$
      $p(\mathbf{x}) \leftarrow (e^{G(\mathbf{x})})/(e^{G(\mathbf{x})} + e^{-G(\mathbf{x})})$
Излез: $F(\mathbf{x}) = \text{sign}[G(\mathbf{x})] = \text{sign}\Big[\sum_{m=1}^{M} f_m(\mathbf{x})\Big]$, за секое $\mathbf{x} \in \mathbb{R}^d$

Алгоритам 3: LogitBoost [16, p. 351]

- *Gentle AdaBoost*

  Gentle AdaBoost, или само GentleBoost, претставува варијанта на AdaBoost според која излезот на секој слаб класификатор се одбира така што истиот ја минимизира тежинската сума $\sum_{i=1}^{N_{train}} \omega_i (y_i - g_m(\mathbf{x}_i))^2$, за секое $m = 1, \ldots, M$. На тој начин, во случај кога перформансите на m-тиот слаб класификатор се приближно совршени, ваквиот пристап овозможува за дадена инстанца $\mathbf{x}$ која припаѓа на класа y, да се избере $g_m(\mathbf{x}) = \alpha_m f_m(\mathbf{x})$ така што $I(g_m(\mathbf{x}) = y) = 1$.

Влез: $\mathcal{D}_{train} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{train}}\}$ - множество за обука

$\mathbf{y} = (y_1, y_2, \ldots, y_{N_{train}}) \in \{-1, 1\}^{N_{train}}$ - вектор на одговори

Иницијализација: $\omega_i = 1/N_{train}, i = 1, \ldots, N_{train}$ ▷ униформна иницијализација на тежините на инстанците

$G(\mathbf{x}) = 0$

Процедура:

  за секое $m = 1, \ldots, M$:

    за секое $i = 1, \ldots, N_{train}$:

      Одреди го $f_m(\mathbf{x})$ со помош на тежинска least-squares регресија на z_i како функција од $\mathbf{x}_i$, користејќи ги тежините ω_i

      $G(\mathbf{x}) \leftarrow G(\mathbf{x}) + (1/2) f_m(\mathbf{x})$

      $\omega_i \leftarrow \omega_i e^{-y_i f_m(\mathbf{x}_i)}$ ▷ ажурирање на тежините на инстанците

    за секое $i = 1, \ldots, N_{train}$:

      $\omega_i \leftarrow \omega_i / \sum_{i=1}^{N_{train}} \omega_i$ ▷ нормализација на тежините

Излез: $F(\mathbf{x}) = \text{sign}[G(\mathbf{x})] = \text{sign}\left[\sum_{m=1}^{M} f_m(\mathbf{x})\right]$, за секое $\mathbf{x} \in \mathbb{R}^d$

Алгоритам 4: Gentle AdaBoost [16, p. 353]

3 Предложен метод

Во рамките на овој дел од дипломската работа е предложен метод за конструкција на интерактивен ансамбл на две нивоа за бинарна класификација базиран на кумулативна соработка помеѓу неговите компоненти, проследен со соодветен алгоритам за негово учење. Имено, образложена е структурата и организацијата на компонентите во ваквиот тип на сложен ансамбл и опишан е начинот на кој истиот може да биде обучен.

3.1 Интерактивен ансамбл на две нивоа

Во 1.2 беше споменато дека целта на учењето со помош на еден ансамбл од модели е да се добијат подобри перформанси при процесот на предвидување, кои инаку не би можеле да се постигнат поединечно од страна на некој од составните модели. Но, сепак главното прашање во суштина е дали ваквиот пристап може да се обопшти. Имено, обопштувањето на овој концепт генерално се базира на начинот на организирање на составните модели и начинот на комбинирање на нивните излези. Ова подразбира групирање на моделите на начин кој ќе обезбеди комбинациите од нивните излези да водат кон постигнување на подобри перформанси.

Во случај на проблем на бинарна класификација, водејќи се според горенаведениот пристап, можно е да се конструираат сложени ансамбли од модели за бинарна класификација. Овие посложени ансамбли во суштина претставуваат обопштување на поимот за ансамбл за класификација. Наспроти ансамблите чии предвидувања се базираат на слабите класификатори во нивниот состав, во рамките на предвидувањата на сложените ансамбли влегуваат одлуките на повеќе стандардни ансамбли составени од слаби класификатори.

Да претпоставиме случај на бинарна класификација кога на располагање имаме L слаби класификатори и множество за обука $\mathcal{D}_{train}$. Еден природен пристап кон градење на хипотеза за податоците во $\mathcal{D}_{train}$ се базира на организација на L-те класификатори во рамки на еден ансамбл и обучување на истиот со помош на некој од методите предложени во 2.3. Она што се предложува како алтернатива претставува комбинација од ваквите методи. Имено, доколку со помош на некоја од варијантите на bagging методот се земат K примероци од $\mathcal{D}_{train}$, но веднаш потоа наместо да се обучува по еден слаб класификатор на еден примерок и за ненабљудувани инстанци да одлучуваат K слаби класификатори, можно е L-те слаби класификатори да се групираат во K ансамбли така што j-тиот ансамбл да биде составен од M слаби класификатори и да се обучува на примерокот $\mathcal{D}_{train}^{(j)}$, за секој $j = 1, ..., K$, а конечната одлука всушност да претставува сублимат од одлуките на ансамблите. Притоа, мора да се запази равенството $L = KM$. Дополнително, доколку за обука на секој

од K-те ансамбли се користи boosting методот, во тој случај процесот на обука на ваквиот сложен ансамбл претставува комбинација од методите bagging и boosting. Оваа организација на класификаторите дефинира две нивоа на класификатори. Како класификатори на прво ниво ги вбројуваме L-те слаби класификатори, додека K-те ансамбли формирани од L-те класификатори можеме да ги разгледуваме како класификатори на второ ниво.

Клучната карактеристика на сложениот ансамбл претставува можноста за колаборација помеѓу ансамблите на второто ниво. Да претпоставиме дека е даден еден сложен ансамбл од овој тип, составен од K ансамбли на второ ниво, секој од нив составен од M слаби бинарни класификатори и обучен со помош на даден boosting алгоритам $\mathcal{B}$. Дополнително, нека постои соодветен начин на колаборација помеѓу ансамблите на второ ниво за време на обуката, што би резултирало со формирање на интерактивен ансамбл на две нивоа за бинарна класификација. За сложениот ансамбл дефиниран на овој начин, се очекува преку колаборативна класификација на претходно ненабљудувани инстанци да обезбеди помала генерализациска грешка во однос на онаа добиена со класификација од страна на единечен ансамбл од KM слаби класификатори обучен со алгоритамот $\mathcal{B}$.

3.2 Алгоритам за конструкција и учење на интерактивен ансамбл на две нивоа за бинарна класификација со кумулативна колаборација

Претходно опишаниот модел за бинарна класификација базиран на интерактивен ансамбл на две нивоа може да биде конструиран и обучен со помош на соодветен алгоритам. Во продолжение ќе биде опишан предлог алгоритам за конструкција и учење на сложен ансамбл од овој тип преку колаборација меѓу неговите составни ансамбли на второ ниво која се одвива кумулативно.

Влезни параметри:
K - број на ансамбли (број на подмножества)
M - број на слаби класификатори по ансамбл
μ - приближен однос помеѓу кардиналноста на j-тото подмножество и кардиналноста на множеството за обука, каде $j = 1, \ldots, K$ т.е. ваквиот параметар означува колкав дел, приближно, од множеството за обука би влегол во рамките на секое од K-те подмножества
$\mathcal{B}$ - алгоритам врз база на кој секој од K-те ансамбли се обучува и ја евалуира својата тежинска одлука (можните варијанти се 'AdaBoostM1', 'LogitBoost' и 'GentleBoost')

Излез: Модел за бинарна класификација со кој е претставен класниот атрибут како функција од останатите атрибути на дадена податочна инстанца.

Чекори на алгоритамот:

1. **Поделба на множеството за обука**

   Поделба на податочното множество за обука $\mathcal{D}_{train} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{train}}\}$ на K bootstrap примероци $\mathcal{D}_{train}^{(1)}, \mathcal{D}_{train}^{(2)}, \ldots, \mathcal{D}_{train}^{(K)}$ чии елементи се избрани на случаен начин (можни се преклопувања помеѓу подмножествата) притоа запазувајќи ги следните ограничувања:

   - распределбата на класите во секое подмножество мора да биде запазена во согласност со распределбата на класите во множеството за обука
   - кардиналноста на секое подмножество $|\mathcal{D}_{train}^{(j)}|$ мора да е приближно еднаква на μN_{train}, каде N_{train} е кардиналноста на множеството за обука

   Може да се забележи дека за поделба на $\mathcal{D}_{train}$ се користи независно земање на примероци кое е опишано во 2.3.1, притоа запазувајќи ја распределбата на класите. Изборот на методот за поделба се должи на [17, p. 206], каде се покажува дека независното земање на примероци обезбедува помала генерализациска грешка во однос на стандардниот bagging каде не се запазува вистинската распределба на множеството за обука.

2. **Обука на моделот**

   За секое $j = 1, \ldots, K$ се применува следната постапка:

   - Се одредува излезот на j-тиот ансамбл F_j, така што како слаби класификатори во неговите рамки се користат дрва на одлучување кои по секој од атрибутите на дадена инстанца се разгрануваат само до своето прво ниво.
   - Ако $j > 1$, тогаш од подмножеството за кое е одговорен j-от ансамбл се формира множество $\mathcal{M}^{(j)}$ кое ги содржи оние инстанци $\mathbf{x}_i (i = 1, \ldots, |\mathcal{M}^{(j)}|,$ $|\mathcal{M}^{(j)}| < \frac{1}{2}|\mathcal{D}_{train}^{(j)}|)$ кои биле погрешно класифицирани од страна на истиот. Потоа j-от ансамбл соработува со секој од ансамблите кои му претходат, така што за секое $s = j - 1, \ldots, 1$ се применува следната постапка:
     - инстанцата $\mathbf{x}_i \in \mathcal{M}^{(j)}$ само привремено се отстранува од множеството на j-тиот ансамбл и се додава во множеството на s-тиот ансамбл
     - j-тиот и s-тиот ансамбл се обучуваат повторно земајќи ги во предвид промените од претходниот чекор
     - доколку по учењето на инстанцата $\mathbf{x}_i$, или грешката на s-тиот ансамбл или грешката на j-тиот ансамбл се намалува или и двете, тогаш $\mathbf{x}_i$ перманентно се отстранува од множеството на j-от ансамбл и се додава во рамките на множеството за кое е одговорен s-тиот ансамбл, во спротивно инстанцата $\mathbf{x}_i$ останува во множеството на j-от ансамбл и не се додава во множеството за

кое е одговорен s-тиот ансамбл

- Постапката дефинирана на овој начин се нарекува "кумулативна колаборација" и истата се повторува се додека не се изминат сите ансамбли-претходници или пак додека не се испразни множеството на погрешно класифицирани инстанци од страна на тековниот ансамбл.

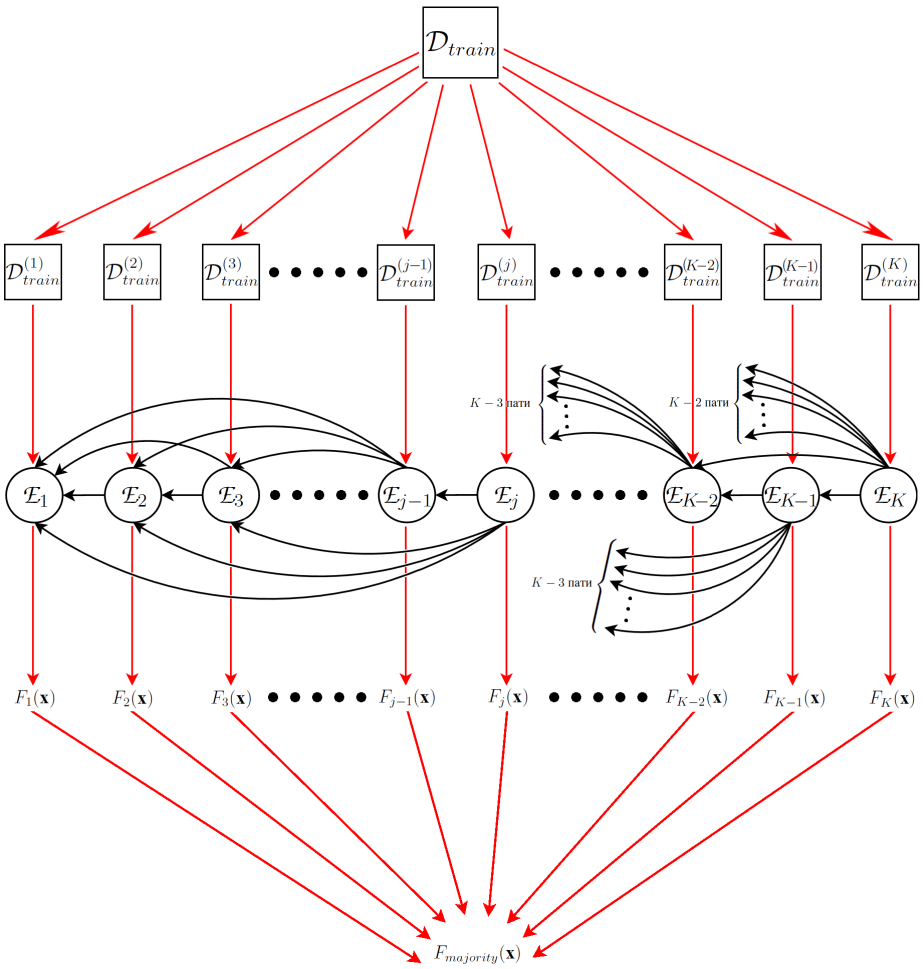
3. Одлучување

Нека одлуките на ансамблите за дадена инстанца **x** ги означиме со $F_1(\mathbf{x})$, $F_2(\mathbf{x})$, $\ldots$, $F_K(\mathbf{x})$. Конечната одлука во врска со класата на инстанцата се базира на мнозинско гласање, т.е.

$$F_{majority}(\mathbf{x}) = S\left(\sum_{j=1}^{K} F_j(\mathbf{x})\right) \ , \tag{21}$$

каде за секој реален број n, функцијата $S(n)$ се пресметува на следниот начин

$$S(n) = \begin{cases} -1, & \text{ако } n < 0 \\ 1, & \text{инаку} \end{cases} \ . \tag{22}$$



Слика 3: Организација, обука и начин на одлучување на интерактивниот ансамбл

Со цел да се обезбеди подетален опис на алгоритамот за конструкција и учење на интерактивниот ансамбл, во продолжение е прикажан и конкретниот редослед на операциите кои се извршуваат во рамки на неговите чекори.

1: **Влез:** $\mathcal{D}_{train} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{train}}\}$ - множество за обука
2: $\mathbf{y} = (y_1, y_2, \ldots, y_{N_{train}}) \in \{-1, 1\}^{N_{train}}$ - вектор на одговори
3: K - број на ансамбли на второ ниво (број на подмножества)
4: M - број на слаби класификатори по ансамбл (на прво ниво)
5: $\mathcal{B}$ - алгоритам за boosting на второ ниво
6: μ - фракција од $\mathcal{D}_{train}$ наменета за генерирањето на секој примерок
7: **Процедура:**
8: **за секое** $j = 1, \ldots, K$:
9: $\mathcal{D}^{(j)}_{train} = \textit{IndependentSampling}(\mathcal{D}_{train}, \mu)$ ▷ генерирање на примерок
10: **за секое** $j = 1, \ldots, K$:
11: $F_j \leftarrow \mathcal{B}(\mathcal{D}^{(j)}_{train})$ ▷ j-тиот ансамбл се обучува на $\mathcal{D}^{(j)}_{train}$
12: **ако** $j > 1$ **тогаш**
13: **за секое** $\mathbf{x}_i \in \mathcal{D}^{(j)}_{train}$:
14: **ако** $F_j(\mathbf{x}_i) \neq y_i$ **тогаш**
15: $\mathcal{M}^{(j)} \leftarrow \mathcal{M}^{(j)} \cup \{\mathbf{x}_i\}$
16: ▷ 17 - 30: колаборација на j-тиот ансамбл со неговите претходници
17: **за секое** $\mathbf{x} \in \mathcal{M}^{(j)}$:
18: **за секое** $s = j - 1, \ldots, 1$:
19: **ако** $\mathbf{x} \notin \mathcal{D}^{(s)}_{train}$ **тогаш**
20: $\mathcal{D}^{(j)}_{train} \leftarrow \mathcal{D}^{(j)}_{train} - \{\mathbf{x}\}$
21: $\mathcal{D}^{(s)}_{train} \leftarrow \mathcal{D}^{(s)}_{train} \cup \{\mathbf{x}\}$
22: $F'_j \leftarrow \mathcal{B}(\mathcal{D}^{(j)}_{train})$
23: $F'_s \leftarrow \mathcal{B}(\mathcal{D}^{(s)}_{train})$
24: **ако** $\varepsilon^{(j')}_{train} \geq \varepsilon^{(j)}_{train} \vee \varepsilon^{(s')}_{train} \geq \varepsilon^{(s)}_{train}$ **тогаш**
25: $\mathcal{D}^{(j)}_{train} \leftarrow \mathcal{D}^{(j)}_{train} \cup \{\mathbf{x}\}$
26: $\mathcal{D}^{(s)}_{train} \leftarrow \mathcal{D}^{(s)}_{train} - \{\mathbf{x}\}$
27: **инаку**
28: $F_j \leftarrow F'_j$
29: $F_s \leftarrow F'_s$
30: врати се на линија број 17.
31: **Излез:** $F_{majority}(\mathbf{x}) = S\Big(\sum_{j=1}^{K} F_j(\mathbf{x})\Big)$ ▷ комбинирање на излезите од
32: ансамблите на второ ниво преку мнозинско гласање

Алгоритам 5: Алгоритам за конструкција и учење на интерактивниот ансамбл на две нивоа за бинарна класификација преку кумулативна колаборација

Сложеност на алгоритамот

Поради природата на проблемот за кој е наменет алгоритамот, истиот може да биде разгледуван како целина од повеќе подпроблеми. Конечната комплексност на алгоритамот може да биде пресметана со одредување на бројот на операции потребни за секоја негова фаза која е наменета за одреден подпроблем. Имено, за дадени K, M и μ на влез, за поделба на множество за обука со кардиналност N се потребни $K\mu N$ операции, веднаш потоа за обука на секој слаб класификатор на прво ниво се потребни $d\mu N$ операции, што би значено дека за обука на секој од K-те ансамбли на второ ниво ќе бидат потребни $d\mu NM$ операции. Дополнително, за пронаоѓање на погрешно класифицираните инстанци од страна на j-тиот ансамбл се потребни μN операции, додека за колаборација помеѓу истиот и неговите претходници се потребни $\mu N(j-1)2d\mu NM$ операции, за секое $j = 1, \ldots, K$.

Оттука, сложеноста на процесот за обука кој ги вклучува чекорите **1.** и **2.**, е

$$\begin{aligned} &O\big(K\mu N + K(d\mu NM + \mu N) + \mu N[(1+2+\ldots,K-1)2d\mu NM]\big) = \\ &= O\big(K\mu N + Kd\mu NM + K\mu N + \mu N\frac{(K-1)K}{2}2d\mu NM\big) = \\ &= O\big(d\mu NMK + d\mu^2N^2MK^2\big)\ . \end{aligned} \tag{23}$$

Во однос на чекорот **3.**, за дадена инстанца $\mathbf{x} \in \mathbb{R}^d$, асимптотското однесување на функцијата $F_{majority}(\mathbf{x})$ се опишува со

$$O\big(KM\big) \tag{24}$$

бидејќи предвидувањето од страна на еден ансамбл на второ ниво има линеарна сложеност т.е. $O\big(F_j(\mathbf{x})\big) = O\big(M\big)$, за секое $j = 1, \ldots, K$.

Имплементација

Софтверската имплементација на предложениот алгоритам е изведена во програмскиот пакет MATLAB. За секој од алгоритмите за конструкција и учење на ансамблите на второ ниво се искористени нивните официјални MATLAB имплементации [18]. Целосното програмско решение може да биде преземено од [19].

3.3 Валидација на интерактивниот ансамбл

Во случај кога за обука на интерактивниот ансамбл на две нивоа се користи множество чиј број на инстанци е од редот 10^5 или поголем, дури и кога бројот на атрибути е мал, алгоритамот може да бара многу пресметки. Во овие ситуации корисно е интерактивниот ансамбл да се обучува само на дел од множеството за обука, а притоа да има доволно знаење за податоците со кое инаку би се здобил кога би бил обучен земајќи го во предвид целото множество за обука. Овој ефект може да биде

постигнат со поделба на оригиналното множество за обука $\mathcal{D}_{train}$ на намалено множество за обука $\mathcal{D}'_{train}$ и множество за валидација $\mathcal{D}_{validation}$, така што да важи дека $\mathcal{D}'_{train} \cup \mathcal{D}_{validation} = \mathcal{D}_{train} \wedge \mathcal{D}'_{train} \cap \mathcal{D}_{validation} = \emptyset$. Сега, интерактивниот ансамбл може да биде обучен на множеството $\mathcal{D}'_{train}$ што би резултирало со намалување на бројот на операции при извршување на алгоритамот за негова обука, додека множеството за валидација може да биде искористено за одредување на $\gamma_1, \gamma_2, \ldots, \gamma_K$, каде значајноста на j-тиот ансамбл на второто ниво е γ_j. Оваа промена би овозможила тежинско вреднување на излезите од ансамблите на второ ниво во рамки на конечната одлука на следниот начин

$$F_{majority}(\mathbf{x}) = S\Big(\sum_{j=1}^{K} \gamma_j F_j(\mathbf{x})\Big) \ . \tag{25}$$

Со минимизација на сумата од квадратните отстапувања на вредноста на класата на секоја инстанца $\mathbf{x}_i \in \mathcal{D}_{validation} (i = 1, \ldots, N_{val} = |\mathcal{D}_{validation}|)$ од тежинската сума на излезите од ансамблите на второто ниво т.е. со минимизација на

$$\sum_{i=1}^{N_{val}} \Big(y_i - \sum_{j=1}^{K} \gamma_j F_j(\mathbf{x}_i)\Big)^2 \ , \tag{26}$$

по секое γ_j, се добива системот од линеарни равенки

$$\begin{cases} \sum_{j=1}^{K} \gamma_j \sum_{i=1}^{N_{val}} F_1(\mathbf{x}_i) F_j(\mathbf{x}_i) = \sum_{i=1}^{N_{val}} y_i F_1(\mathbf{x}_i) \\ \sum_{j=1}^{K} \gamma_j \sum_{i=1}^{N_{val}} F_2(\mathbf{x}_i) F_j(\mathbf{x}_i) = \sum_{i=1}^{N_{val}} y_i F_2(\mathbf{x}_i) \\ \quad \vdots \\ \sum_{j=1}^{K} \gamma_j \sum_{i=1}^{N_{val}} F_K(\mathbf{x}_i) F_j(\mathbf{x}_i) = \sum_{i=1}^{N_{val}} y_i F_K(\mathbf{x}_i) \end{cases} . \tag{27}$$

Оттука значајностите $\gamma_1, \gamma_2, \ldots, \gamma_K$ можат да бидат пресметани со решавање на системот (27).

4 Експериментални резултати

Во овој дел од дипломската работа се наведени експериментите извршени со помош на алгоритамот опишан во 3.2 и опишани се податочните множества кои се користени во рамки на истите. Дополнително, презентирани се резултатите добиени од извршените експерименти и толкувано е нивното значење преку соодветна дискусија.

4.1 Опис на користените податочни множества

Секој од експериментите кои ќе бидат опишани подоцна вклучува користење на одредено податочно множество. Во продолжение е наведен краток опис за секое од користените податочни множества.

BUPA Liver Disorders

Податочното множество BUPA Liver Disorders [20] содржи податоци за пациенти кај кои се испитуваат нарушувања на црниот дроб. Множеството содржи 345 инстанци така што секоја инстанца се однесува на еден пациент. Карактеристиките на пациентот се претставени преку 6-те атрибути на инстанцата со која е претставен и истите имаат целобројни или реални вредности. Класниот атрибут може да има една од вредностите -1 или 1.

Statlog (Heart)

Податочното множество Statlog (Heart) [21] содржи податоци за пациенти кои потенцијално страдаат од срцево заболување. Множеството содржи 270 инстанци, а секоја инстанца има 13 атрибути кои имаат реални вредности. Класниот атрибут прима една од вредностите -1 или 1 во зависност од отсуството, односно присуството на срцево заболување.

Breast Cancer Wisconsin (Original)

Податочното множество Breast Cancer Wisconsin (Original) [20] содржи податоци за пациенти кои боледуваат од карцином на дојка. Множеството содржи 683 инстанци, а секоја инстанца има 10 атрибути кои имаат целобројни вредности. Класниот атрибут прима една од вредностите -1 или 1 во зависност од тоа дали карциномот е бенигнен или малигнен, соодветно.

Се користи претпроцесирана верзија на множеството. Имено, отстранет е првиот атрибут кој претставува само идентификатор на инстанца. Дополнително се отстранети 16 инстанци кај кои недостасуваат некои од вредностите на нивните атрибути.

Pima Indians Diabetes

Податочното множество Pima Indians Diabetes [20] содржи податоци за пациенти кои потенцијално боледуваат од диабетес. Множеството содржи 768 инстанци, а секоја инстанца има 8 атрибути кои имаат целобројни или реални вредности. Класниот атрибут прима една од вредностите -1 или 1 во зависност од тоа дали тестот за диабетес бил негативен или позитивен, соодветно.

Leukemia

Податочното множество Leukemia [22] содржи податоци за пациенти кои потенцијално боледуваат од леукемија. Множеството содржи 72 инстанци, а секоја инстанца има 7129 атрибути кои имаат реални вредности. Класниот атрибут прима една од вредностите -1 или 1, во зависност од отсуството, односно присуството на леукемија кај пациентот.

Претпроцесирањето на множеството вклучува нормализација на вредностите на атрибутите по инстанца така што средната вредност на истите да е еднаква на 0, а нивната варијанса да е еднаква на 1. Дополнително е извршена и нормализација на вредностите на секој од атрибутите поединечно на истиот начин.

Fourclass

Податочното множество Fourclass [23] претставува вештачки создадено множество со 862 инстанци кои претставуваат координати на точки во дводимензионален, ограничен простор. Инстанците се изгенерирани така што секоја од нив може да припаѓа на една од 4 класи. Класите имаат неправилна распространетост низ просторот, региони на прекин и истите не се линеарно сепарабилни.

Множеството е претпроцесирано така што извршено е групирање на класите по парови што овозможува истото да биде користено за бинарна класификација.

Skin Segmentation

Податочното множество Skin Segmentation [20] содржи податоци за пиксели во слики од лица на луѓе од различен пол, возраст и боја на кожа. Множеството содржи 245057 инстанци, додека секоја инстанца има 3 атрибути кои претставуваат B,G,R вредности на случајно одбран пиксел од една од сликте. Класниот атрибут прима една од вредностите -1 или 1, во зависност од тоа дали разгледуваниот пиксел припаѓа или не припаѓа на регион во сликата каде има човечка кожа.

4.2 Опис на извршените експерименти

Експериментите кои се извршени вклучуваат бинарна класификација на инстанци од претходно опишаните податочни множества со помош на добро познатите алгоритми за boosting опишани во 2.3.2, како и со предложениот алгоритам. Секое податочно множество е поделено на множество за обука кое завзема 60% од инстанците во оригиналното множество и множество за тестирање кое ги содржи останатите 40% од инстанците. При поделбата на оригиналното множество запазена е распределбата на класите.

Во секој од експериментите за дадени влезни параметри на предложениот алгоритам K, M, μ и $\mathcal{B}$, истиот се обучува на дадено множество за обука $\mathcal{D}_{train}$ и дополнително со $\mathcal{B}$ се обучува единечен ансамбл од KM слаби класификатори. Откако ќе заврши процесот на обука во двата случаи, покрај грешките при обука, се пресметуваат и следните генерализациски грешки:

- Генерализациска грешка на интерактивниот ансамбл

$$E_{test} = \frac{\sum_{i=1}^{N_{test}} I(F_{majority}(\mathbf{x}_i) \neq y_i)}{N_{test}} \ , \tag{28}$$

  каде $\mathbf{x}_i$ преставуваат инстанци од множеството за тестирање $\mathcal{D}_{test} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{test}}\}$. Притоа, кај некои од експериментите колаборацијата помеѓу ансамблите на второ ниво е оневозможена.

- Минимална и максимална генерализациска грешка од секои M итерации на boosting алгоритамот со кој се врши споредбата

$$\begin{aligned} \varepsilon^{(K)}_{test_{min}} &= \min\left\{\varepsilon^{((K-1)M+1)}_{test}, \ldots, \varepsilon^{(KM)}_{test}\right\} \\ \varepsilon^{(K)}_{test_{max}} &= \max\left\{\varepsilon^{((K-1)M+1)}_{test}, \ldots, \varepsilon^{(KM)}_{test}\right\} \ . \end{aligned} \tag{29}$$

Соодветно, грешките при обука во рамки на експериментите се означени со E_{train}, $\varepsilon^{(K)}_{train_{min}}$ и $\varepsilon^{(K)}_{train_{max}}$.

Целта на експериментите е да се провери дали колаборацијата помеѓу ансамблите на второ ниво има значајна улога при обуката на интерактивниот ансамбл и дали перформансите во однос на можноста за генерализација од страна на истиот се подобри во споредба со оние кај познатите алгоритми за boosting.

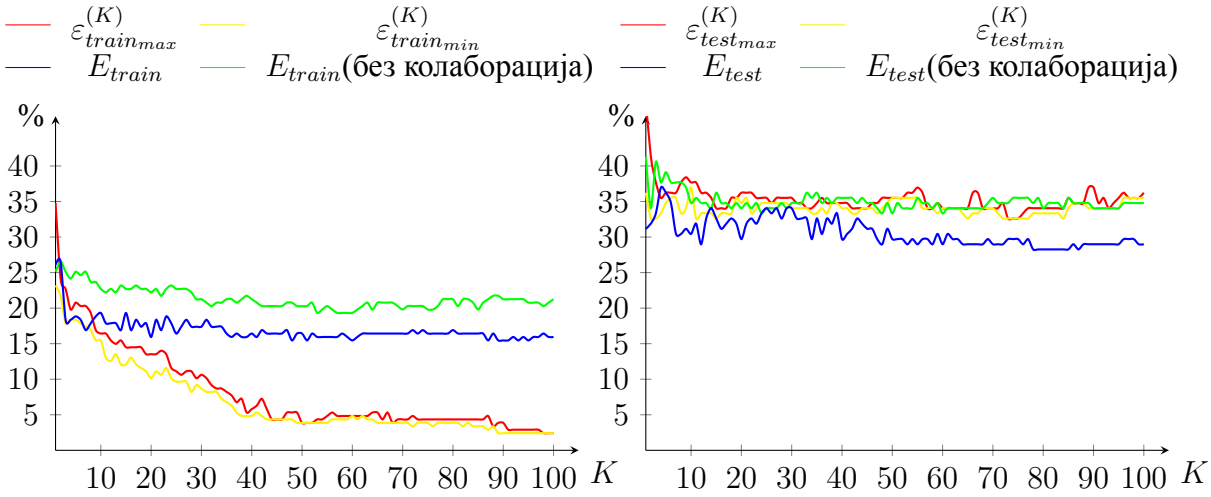
4.3 Резултати и дискусија

Во овој дел ќе бидат графички прикажани резултатите од извршените експерименти проследени со соодветна дискусија.

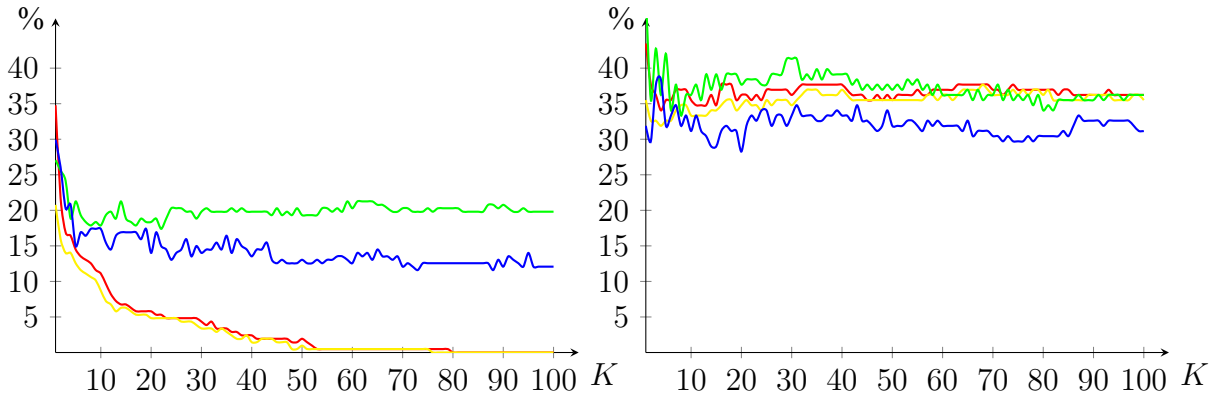
4.3.1 Утврдување на значајноста на колаборацијата

BUPA Liver Disorders

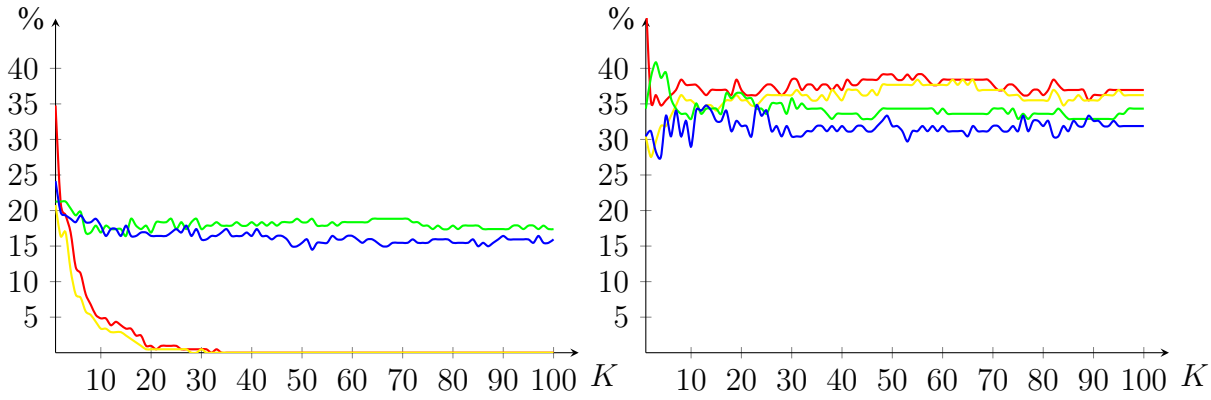
Во продолжение графички се прикажани резултатите од обуката (од лева страна) и тестирањето (од десна страна) на интерактивниот ансамбл со и без колаборација, како и на единечниот ансамбл.



Слика 4: Влезни параметри: $K = 100, M = 15, \mu = 0.6$. Boosting алгоритам: $AdaBoostM1$.



Слика 5: Влезни параметри: $K = 100, M = 15, \mu = 0.4$. Boosting алгоритам: $LogitBoost$.



Слика 6: Влезни параметри: $K = 100, M = 15, \mu = 0.8$. Boosting алгоритам: $GentleBoost$.

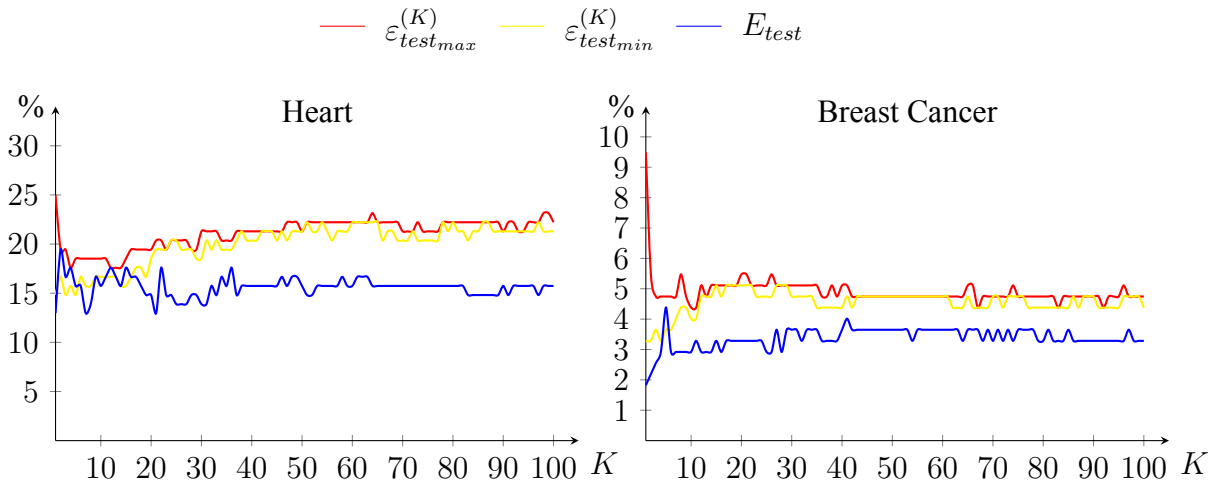
Од резултатите прикажани на сликите 4, 5 и 6 се забележува дека грешката при обука на интерактивниот ансамбл, со и без колаборација, е поголема од максималната грешка при обука на единечен ансамбл без разлика на boosting алгоритамот кој се користи. Но, иако навидум интерактивниот ансамбл се покажува полошо при обуката, истиот обезбедува помала генерализациска грешка при фазата на тестирање во сите три случаи на boosting и без разлика на вредноста на параметарот K.

Според последицата 1 од теоремата 2, грешката при обука на еден ансамбл тежи кон 0 кога неговите составни класификатори/ансамбли се независни и непристрасни. Тоа не е случај кај интерактивниот ансамбл бидејќи колаборацијата помеѓу ансамблите на второ ниво создава зависност помеѓу нив. Со тоа е прекршен првиот услов за теоремата 2 да важи што генерално допринесува грешката при обука на интерактивниот ансамбл да не достигне 0. Резултатите покажуваат дека ова води кон погенерални одлуки при фазата на тестирање. Дополнително, фактот дека секој ансамбл на второ ниво поминува низ мал број на итерации за обука т.е. содржи мал број на слаби класификатори претставува само уште една причина за помалата генерализациска грешка на сложениот ансамбл.

4.3.2 Споредба со стандардните алгоритми за boosting

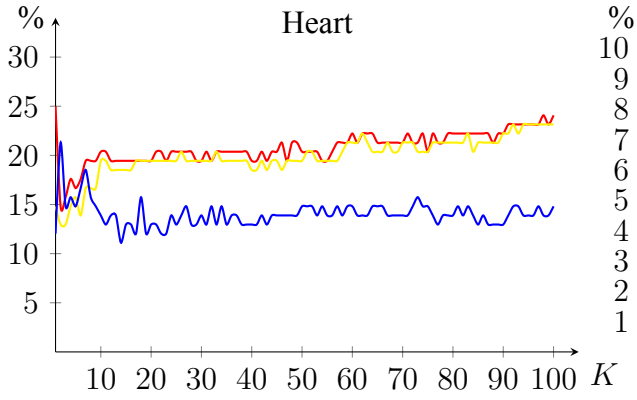
Споредба на способноста за генерализација на интерактивниот ансамбл на две нивоа со онаа на стандардните алгоритми за boosting е спроведена на уште неколку податочни множества. Соодветните резултати се прикажани во продолжение.

Statlog (Heart) и Breast Cancer Wisconsin (Original)

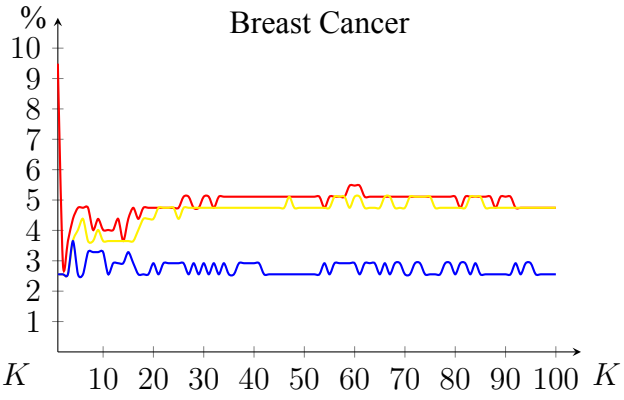


Слика 7: $K = 100, M = 30, \mu = 0.7$. Boosting алгоритам: $AdaBoostM1$.

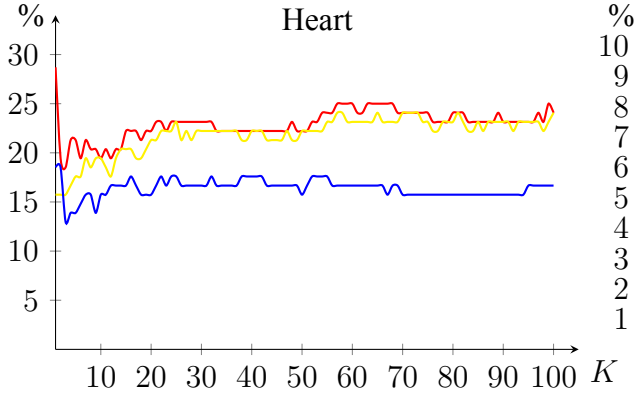
Слика 8: $K = 100, M = 10, \mu = 0.6$. Boosting алгоритам: $AdaBoostM1$.



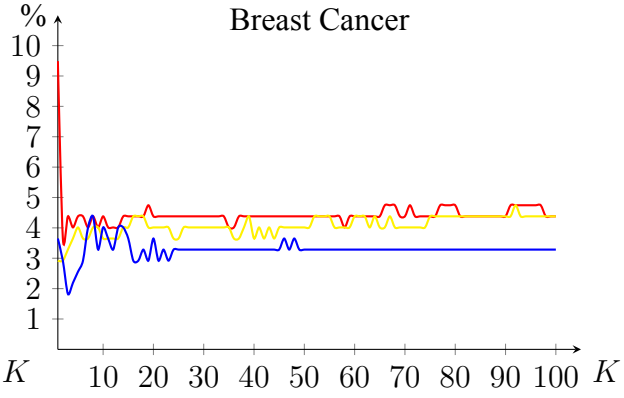
Слика 9: $K = 100, M = 30, \mu = 0.4$.
Boosting алгоритам: $LogitBoost$.



Слика 10: $K = 100, M = 10, \mu = 0.7$.
Boosting алгоритам: $LogitBoost$.

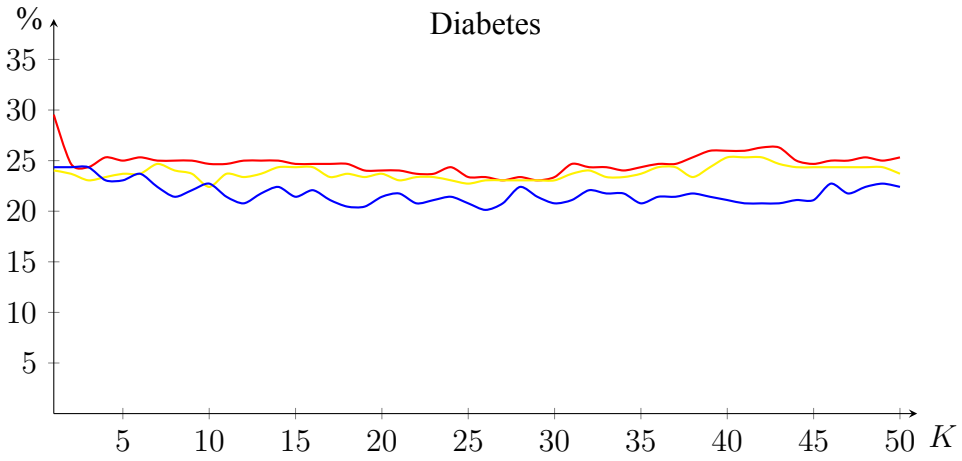


Слика 11: $K = 100, M = 30, \mu = 0.9$.
Boosting алгоритам: $GentleBoost$.

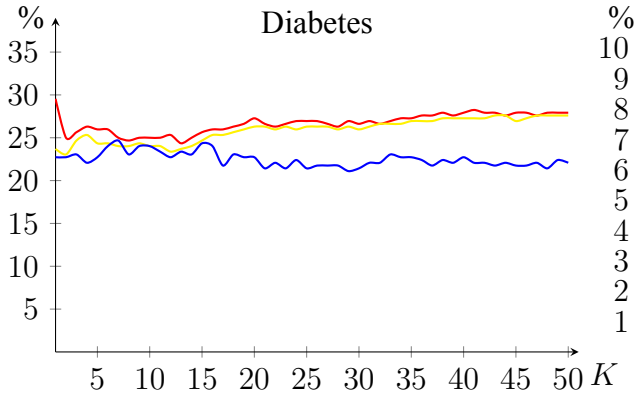


Слика 12: $K = 100, M = 10, \mu = 0.4$.
Boosting алгоритам: $GentleBoost$.

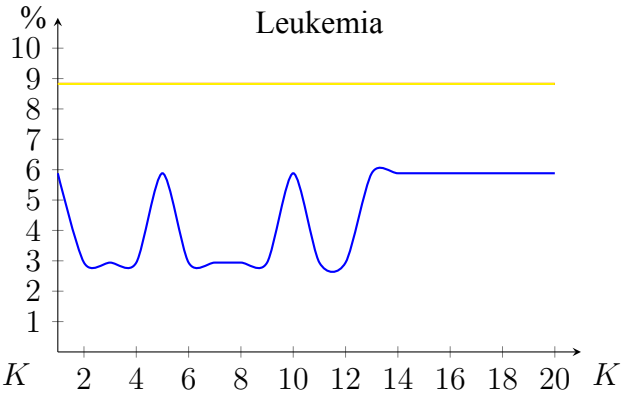
Pima Indians Diabetes и Leukemia



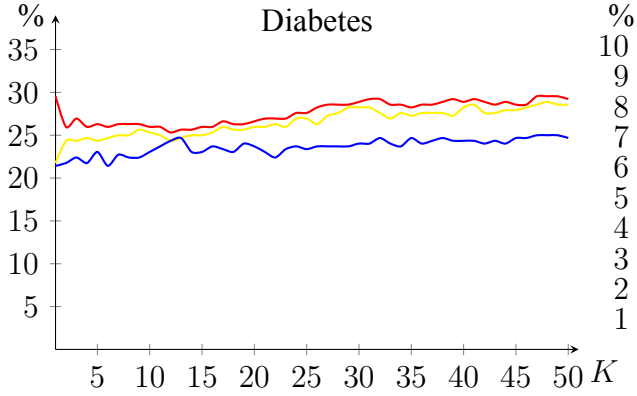
Слика 13: $K = 50, M = 20, \mu = 0.3$. Boosting алгоритам: $AdaBoostM1$.



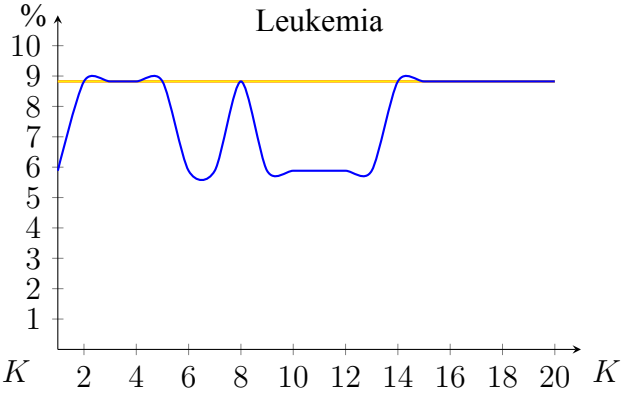
Слика 14: $K = 50, M = 20, \mu = 0.4$.
Boosting алгоритам: $LogitBoost$.



Слика 15: $K = 20, M = 20, \mu = 0.7$.
Boosting алгоритам: $LogitBoost$.



Слика 16: $K = 50, M = 20, \mu = 0.7$.
Boosting алгоритам: $GentleBoost$.



Слика 17: $K = 20, M = 20, \mu = 0.7$.
Boosting алгоритам: $GentleBoost$.

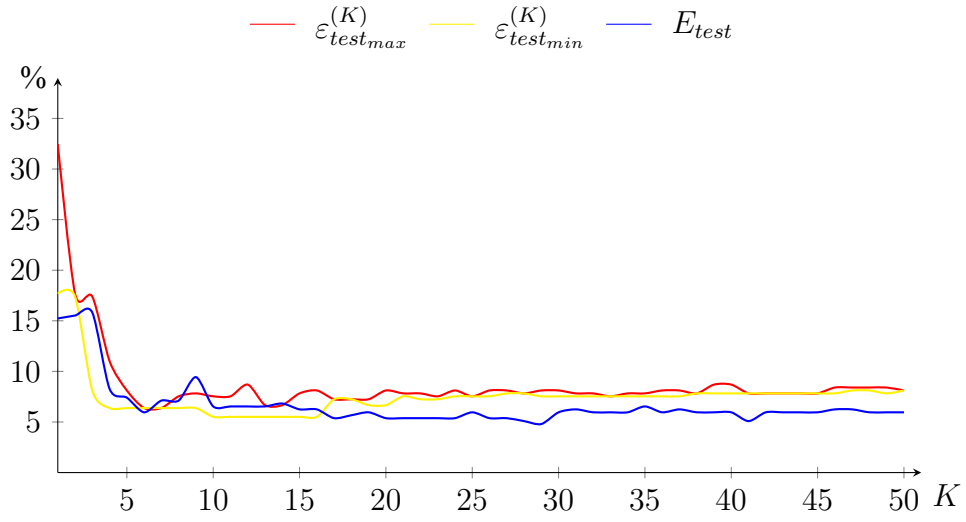
Горенаведените резултати ја потврдуваат тезата дискутирана во 4.3.1. Имено, за секое од податочните множества кои се користат за обука, интерактивниот ансамбл најголем дел од времето во фазата на тестирање (за најголем број на различни вредности на K) има помала генерализациска грешка од минималната грешка при тестирање на секој од boosting алгоритмите. Исклучок делумно се јавува само кај резултатот од Слика 17.

Во некои од случаите како што се Слика 9 и Слика 14 се забележува и благ тренд на разидување на генерализациската грешка на интерактивниот ансамбл и минималната генерализациска грешка на соодветниот boosting алгоритам со кој се споредува истиот.

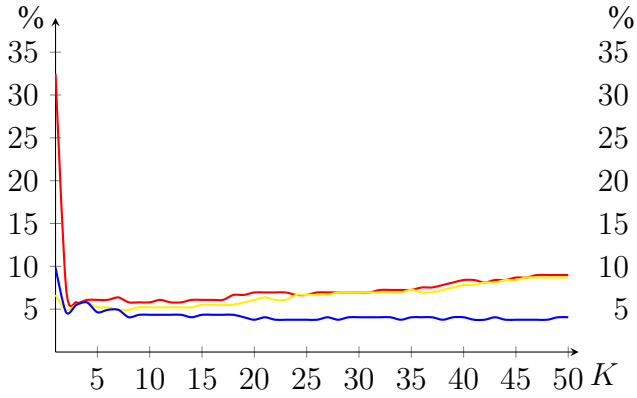
Потенцијален проблем може да се јави во случај кога ансамблите на второ ниво користат одреден алгоритам за boosting, но во нивните рамки постојат класификатори чија грешка при обука на соодветните подмножества е поголема од 50%. Ваков случај се јавува кога при обука на интерактивниот ансамбл со множеството Leukemia, на второ ниво се користи AdaBoostM1. Оттука, јасно е дека во овие ситуации потребно е да се одбере друг алгоритам за boosting на второ ниво.

4.3.3 Однесување на вештачки генерирани податоци

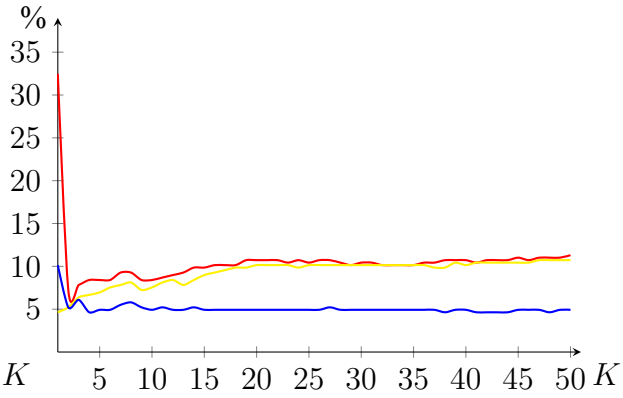
Fourclass



Слика 18: $K = 50, M = 60, \mu = 0.4$. Boosting алгоритам: $AdaBoostM1$.



Слика 19: $K = 50, M = 60, \mu = 0.6$. Boosting алгоритам: $LogitBoost$.

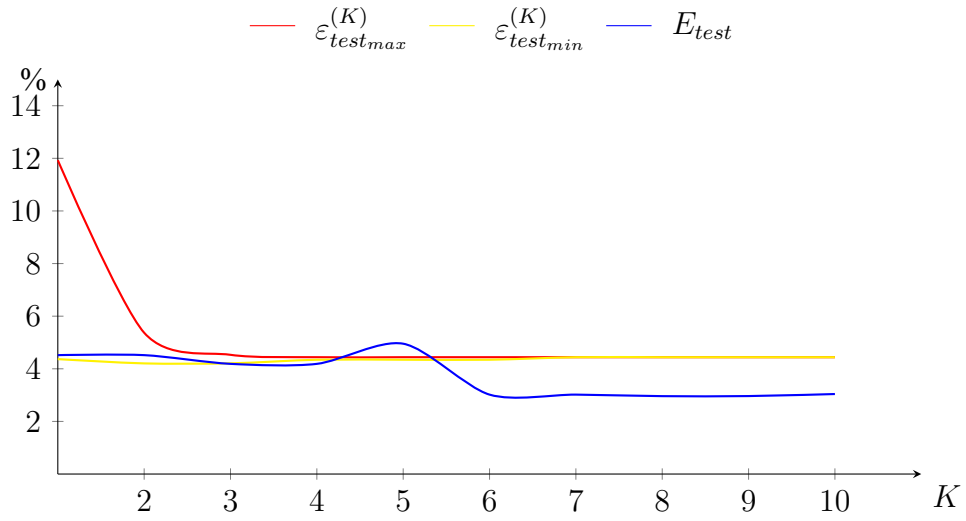


Слика 20: $K = 50, M = 60, \mu = 0.7$. Boosting алгоритам: $GentleBoost$.

Кај експериментите кои вклучуваат обука на интерактивниот ансамбл со вештачки генерирани податоци, повторно се забележува дека неговата генерализациска грешка е помала од минималната на boosting алгоритамот кој се користи за споредба. И во овој случај се јавува благ тренд на разидување на овие две грешки со зголемувањето на вредноста на параметарот K.

4.3.4 Употреба на множество за валидација

Skin Segmentation



Слика 21: $K = 10, M = 150, \mu = 0.8, \left[|\mathcal{D}_{validation}|/|\mathcal{D}_{train}|\right] = 0.3$. Boosting алгоритам: $AdaBoostM1$.

Резултатот од употребата на множество за валидација уште еднаш ја потврдува тезата дека интерактивниот ансамбл обезбедува помала генерализациска грешка во споредба со единечен ансамбл кој се обучува со boosting алгоритамот AdaBoostM1. Дополнително, помалата генерализациска грешка го потврдува тврдењето дека употребата на дел од множеството за обука на овој начин може да допринесе моделот да има знаење за податоците приближно на знаењето со кое би се здобил кога би бил обучен на целото множество за обука. Слични резултати се добиваат и при споредба со LogitBoost и GentleBoost.

5 Заклучок

Во рамките на оваа дипломска работа беше објаснета потребата од учење со помош на ансамбли од модели. Во случај на проблемот на класификација, користењето на ансамбли од класификатори претставува ефикасен пристап за подобрување на перформансите при процесот на предвидување. Недостаток на алгоритмите за обучување кои вклучуваат boosting претставува потенцијалната можност истите да доведат до преголема пристрасност на ансамблите кон податоците на кои се обучуваат. Сепак, овие алгоритми оставаат простор за нивно подобрување. Токму за оваа цел беше предложен алтернативен интерактивен ансамбл на две нивоа, проследен со алгоритам за негова конструкција и учење. Резултатите од извршените експерименти покажуваат дека кумулативната колаборација помеѓу составните ансамбли на второто ниво претставува главен двигател за донесување на погенерални одлуки во фазата на тестирање. Овој пристап обезбедува генерализациска грешка која е помала од минималната грешка при тестирање на некои од стандардните алгоритми за boosting.

6 Идна работа

Во иднина може да се испита сензитивноста на предложениот алгоритам на останатите параметри покрај параметарот кој се однесува на бројот на ансамбли кои се користат на второ ниво. Покрај структурата на самите ансамбли на второто ниво, може да се анализираат и алгоритмите за boosting кои се користат за обука на истите. Еден тип на алгоритми кои можат да бидат искористени претставуваат екстензиите на AdaBoostM1 за повеќекласна класификација. На овој начин, интерактивниот ансамбл ќе може да се оспособи и за проблеми кои вклучуваат повеќекласна класификација.

Во однос на колаборацијата помеѓу ансамблите на второто ниво, може да се испита од колкава важност би била истата во случај кога овие ансамбли се хетерогени и постои диверзитет помеѓу нив.

Референци

[1] Frank M Selten, GS Duane, W Wiegerinck, N Keenlyside, Jürgen Kurths, and Ljupco Kocarev. Supermodeling by combining imperfect models. *Procedia Computer Science*, 7:261--263, 2011.

[2] Suk Wah Louisa LAM. Theory and application of majority vote: From condorcet jury theorem to pattern recognition. 2000.

[3] Marie Jean Antoine Nicolas de Caritat et al. *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix*. L'imprimerie royale, 1785.

[4] Thomas G Dietterich. Ensemble methods in machine learning. In *Multiple classifier systems*, pages 1--15. Springer, 2000.

[5] Lior Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2):1--39, 2010.

[6] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123--140, 1996.

[7] Leo Breiman. Heuristics of instability in model selection. technique report. statistics department. *University of California at Berkeley*, 1994.

[8] Ethem Alpaydin. *Introduction to machine learning*. MIT press, 2014.

[9] Z.-H. Zhou. Ensemble learning. *S. Z. Li ed. Encyclopedia of Biometrics, Berlin*, pages 270--273, 2009.

[10] Giorgio Fumera, Fabio Roli, and Alessandra Serrau. A theoretical analysis of bagging as a linear combination of classifiers. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(7):1293--1299, 2008.

[11] Robert E Schapire. The strength of weak learnability. *Machine learning*, 5(2):197--227, 1990.

[12] Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23--37. Springer, 1995.

[13] Robert E Schapire and Yoav Freund. *Boosting: Foundations and algorithms*. MIT press, 2012.

[14] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[15] David A Freedman. *Statistical models: theory and practice*. cambridge university press, 2009.

[16] Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337--407, 2000.

[17] Ludmila I Kuncheva. *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2004.

[18] MathWorks. MathWorks ensemble methods, 1994-2015.

[19] GitHub. Bi-level interactive ensemble classifier. `https://github.com/MartinPavlovski/Bi-level-interactive-ensemble-classifier`, 2015. [Online; accessed 24-October-2015].

[20] David J Newman, Seth Hettich, Cason L Blake, and Christopher J Merz. {UCI} repository of machine learning databases. 1998.

[21] Donald Michie, David J Spiegelhalter, and Charles C Taylor. Machine learning, neural and statistical classification. 1994.

[22] Todd R Golub, Donna K Slonim, Pablo Tamayo, Christine Huard, Michelle Gaasenbeek, Jill P Mesirov, Hilary Coller, Mignon L Loh, James R Downing, Mark A Caligiuri, et al. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *science*, 286(5439):531--537, 1999.

[23] Tin Kam Ho and Eugene M Kleinberg. Building projectable classifiers of arbitrary complexity. In *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, volume 2, pages 880--885. IEEE, 1996.