Datuen birlaginketa adimentsua ikasketa automatikoan

Iñaki Albisua, Aritz Lasarguren, Javier Muguerza, Jesús M.ª Pérez

ALDAPA taldea, Konputagailuen Arkitektura eta Teknologia Saila,
Informatika Fakultatea (UPV/EHU)

Laburpena: Ikasketa automatikoa delako adimen artifizialaren arloa, esperientziarekin ikasten duten programak eraikitzen saiatzen da. Esperientzia hori datubaseetan gordetzen diren datuen bidez adierazi ohi da eta ez da beti erraza errealitatearen eredu bat ematen diguten datuak lortzea. Hori dela-eta, zenbait kasutan jasota dagoen informazioaren aurreprozesamendu bat egiten da ikasketa prozesua hasi aurretik. Aurreprozesamendu teknika horien artean erabilienetakoak birlaginketa metodoak dira. Lan honetan birlaginketa teknika ezagun batzuk aztertu eta hobekuntza-proposamen bat egingo dugu. Era berean, metodo horien aplikazioa ikusiko dugu adibide bati jarraiki.

Abstract: Machine learning is a branch of artificial intelligence that tries to develop programs capable of learning from experience. This experience is usually represented as data saved in databases but data being a good reflection of reality is not always easy to obtain. To deal with this problem, sometimes, before starting the learning process, a data preprocessing is required. Resampling methods are one of the most used preprocessing techniques. In this work we will analyze some well-known resampling methods and an improvement we proposed, and we will use an example to explain how they are applied.

1. SARRERA

Ikasketa automatikoak (*Machine Learning*) esperientziarekin ikasten duten programak nola gauzatu aztertzen du. Horretarako, esperientzia edo ezagutza lortzeko protokolo bat behar da, **ikasketa-algoritmoa** deritzona eta bai jasotako ezagutza ebaluatzeko neurri bat ere.

Ikasketa egin ahal izateko hasierako ezagutza bat behar da eta ezagutza hau datu-base batean bilduta dagoenean, **Datu Meatzaritza**-ko (*Data Mining*) kasu bat dugula esango dugu.

Arlo askotan erabili da ikasketa automatikoa, besteak beste medikuntzako diagnosian, planifikazioan, erroboten kontrolean, meteorologia, eta abarren aurreikuspenean, iruzurraren detekzioan, karaktere eta objektuen errekonozimenduan e.a...

Ikasketari dagokionez, **gainbegiratua** (*supervised*) ala **gainbegiratugabea** (*unsupervised*) izan daiteke. Lehenengoan, esperientzia adierazten duten datuekin batera kasu bakoitzari dagokion emaitza dugu; gaixotasun baten detekzioaren kasuan adibidez, aurretik izandako hainbat kideri egindako analisien ezaugarriekin batera, gaixotasun hori zuten ala ez ere adierazten da. Ikasketa gainbegiratugabean ordea, aurkitu nahi dugun baliorik ez dago, eta ditugun datuak aurki daitezkeen ezaugarri komunen arabera multzokatzea da helburua. Lan honetan ikasketa gainbegiratuaz arituko gara.

Adierazpide proposizional edo ezaugarri-balio adierazpidea deritzo jakintza hori adierazteko dagoen modurik erabilienari. Adierazpide honekin, datu bakoitza aldagai edo ezaugarri kopuru finko baten bidez adierazten da, aldagai bakoitzari dagokion balioaren bidez hain zuzen. Adibidez *n* ezaugarri eta *m* kasu dituen datu-base bat adierazteko hauxe dugu:

$$\overline{x}_1 = \langle x_{11}, x_{12}, \ldots, x_{1n} \rangle$$
$$\ldots$$
$$\overline{x}_m = \langle x_{m1}, x_{m2}, \ldots, x_{mn} \rangle$$

1. irudia. *n* ezaugarri eta *m* kasuko datu-basea adierazpide proposizionalean adierazita.

Kasu guztien multzoari **lagin** deritzo, populazio osoaren lagin bat da-eta. Ezaugarri bakoitzari **aldagai** deitzen zaio eta lehen esan bezala, gainbegiratutako ikasketan, ezaugarrien artean bada berezia den bat eta espero dugun emaitza erakusten duena. **Klase aldagai** deitzen zaio eta honek har dezakeen balio posible bakoitzari **klase**. Gaixotasunaren detekzioaren adibidean esate baterako, aldagaiak *adina*, *pisua*, *sukarra* duen ala ez, eta beste hainbat sintomari buruzko informazioa izan daitezke eta klase aldagaiak aztertzen ari garen gaixotasuna duen ala ez esango liguke, kasu honetan klase posibleak «bai» eta «ez» («gaixo» eta «osasuntsu») izanik.

Eta zein da helburua? Dugun informaziotik abiatuta ezagutza lortuko duen ikasketa-algoritmo bat erabili eta ondoren, ikasitakoarekin, kasu berriak agertzean zein klase dagokien aurreikustea. Gaixotasunaren adibidean, paziente berri bat etorrita, aldagaien balioak neurtu eta gaixotasuna duen ala ez esango liguke. Helburua kasu berri bakoitzari klaseetako bat egokitzea izanik, mota honetako problemei sailkapen problemak deritze (*classification problems*) eta algoritmoak erabili eta gero emaitza bezala sortzen diren programei berriz, sailkatzaile (*classifier*).

Eta nahikoa al da klase bat egokitzea? Gaixotasunaren kasuan adibidez, nahikoa da pazientea gaixo edo osasuntsu ote dagoen aurreikustea edo bestela, aurreikuspenarekin batera ondorio horretara iristeko egindako arrazonamendu edo azalpena behar al da? Lan honetan azalpena eskaintzen duten sailkatzaileak erabiliko ditugu, zehazki, hauen artean erabilienetakoak diren sailkapen zuhaitzak. Hauetan, ikasketa fasearen ondorioz zuhaitz motako egitura bat itzultzen da. Zuhaitzeko adabegi bakoitzak galdera edo erabaki bat adierazten du eta zuhaitzeko errotik hasita hosto baterainoko bidea eginez lortzen dira sailkapena eta sailkapenaren azalpena.

Eta zertan datza datuen birlaginketa? Ikasketa-algoritmo gehienek datuen artean klase banaketak nahiko orekatuak daudenean ematen dituzte emaitzarik onenak. Zenbait egoeratan ordea, ohizkoa da klase bateko kasuak bestekoak baino askoz gutxiago izatea. Iruzurraren detekzioan adibidez, bezero askoren kasuak edukita ere gutxi batzuetan detektatzen da iruzurra, eta oso gaixotasun arraro baten kasuan, gehiengoak ez du gaixotasun hori izango. Halako egoeretan sailkapen-algoritmo estandarrek ez dute emaitza onik ematen eta ikasketa egin aurretik datuak aurreprozesatzea izaten da irtenbide erabilienetako bat, kasu berriak sortu edo dauden batzuk kenduz klaseen proportzioak orekatu eta ikasketak emaitza hobeak eman ditzan. Birlaginketa metodoak dira aurreprozesamendu teknika horien artean erabilienetakoak.

Hemendik aurrera esandakoan sakontzen joango gara eta horretarako adibide batean oinarrituko gara, gero aipatzen goazena hobeto ulertzeko laguntza bila. Datu-base hau helburu didaktikoekin maiz erabili den *PlayTennis* datu-basearen moldaketa bat da; bertan terminoei euskal kutsua emateaz gain, hozberoari dagokion informazioa gradutan jarri da (jatorrizkoan hiru balio posible bakarrik daude), zenbakizko aldagaiak ere edukitzeko.

Datu-base hau Tom Mitchelek erabili zuen lehenbizikoz bere «Machine Learning» [1] liburu ospetsuan.

Demagun hainbat egunetako eguraldiaren ezaugarrien arabera eskupilotan jokatzera joan garen ala ez aztertu eta informazioa datu-base batean

bildu dugula. Helburu gisa hartuko dugu aurrerantzean eguraldiaren ezaugarri horiek ezagututa jokatzera joango garen ala ez iragartzea. Ondoko taulan (1. taula) ikus dezakegu laginak zein itxura izango duen:

1. taula. Esku-pilotan jokatuko den ala ez aurreikusteko orain arteko esperientzia jasotzen duen taula.

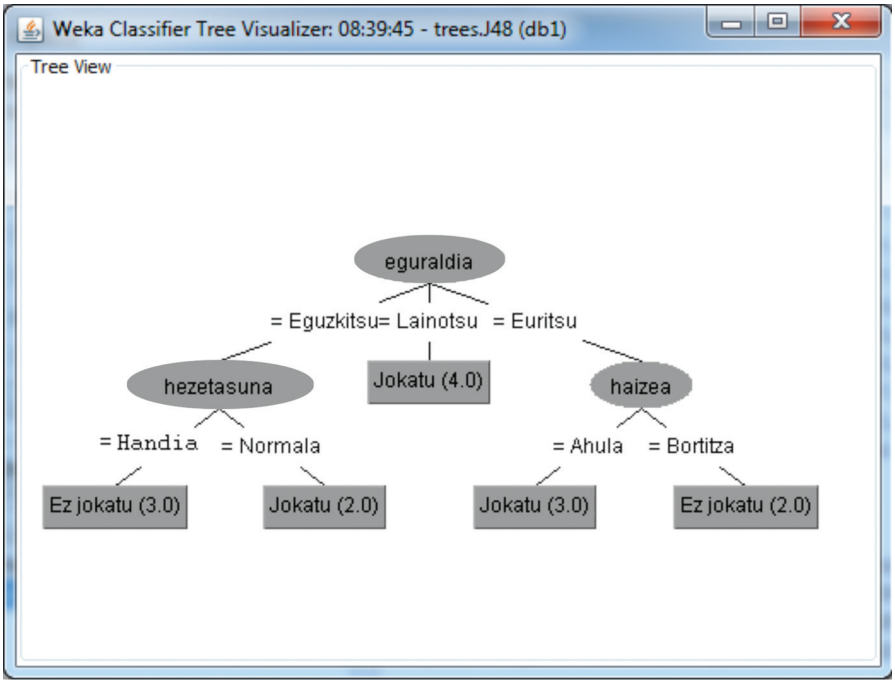
Eguna	Giroa	Hozberoa (gradutan)	Hezetasuna	Haizea	Jokatu
1	Eguzkitsu	30	Handia	Ahula	Ez
2	Eguzkitsu	32	Handia	Bortitza	Ez
3	Lainotsu	28	Handia	Ahula	Bai
4	Euritsu	22	Handia	Ahula	Bai
5	Euritsu	11	Normala	Ahula	Bai
6	Euritsu	14	Normala	Bortitza	Ez
7	Lainotsu	15	Normala	Bortitza	Bai
8	Eguzkitsu	19	Handia	Ahula	Ez
9	Eguzkitsu	13	Normala	Ahula	Bai
10	Euritsu	19	Normala	Ahula	Bai
11	Eguzkitsu	20	Normala	Bortitza	Bai
12	Lainotsu	20	Handia	Bortitza	Bai
13	Lainotsu	29	Normala	Ahula	Bai
14	Euritsu	21	Handia	Bortitza	Ez

Kasu honetan, *Giroa*, *Hozberoa*, *Hezetasuna* eta *Haizea* aldagaiak dira, **Jokatu** berriz klase-aldagaia da eta ikus daitekeenez klase posibleak bi dira: Ez eta Bai.

2. **IKASKETA FASEA**

Ikasketa fasea dugun laginari ikasketa algoritmoa ezartzean datza; bertan, sailkatzaile bat gauzatzen da, hots kasu berriak zein klaseri dagozkion iragarriko duen programa. Lehen esan bezala, sailkapen-zuhaitzak dira guk

erabiltzen ditugun sailkatzaileak, eta kasu honetan, Quinlanek [2] proposatutako C4.5 ezaguna erabiliko dugu. Zehazkiago, Weka [3] datu-meatzaritzarako software irekiak eskaintzen duen C4.5-en J48 aldaera. Ikus dezagun gure adibidean oinarrituta J48 algoritmoak sortzen duen zuhaitz motako sailkatzailea, Wekak berak erakusten digun formatuan (2. irudia):



2. irudia. Wekako J48 algoritmoarekin gauzatutako sailkatzailea.

Sailkatzailea gauzatuta, etorkizunean jokatzera joango garen ala ez aurreikusi nahi badugu, lehenbizi eguraldiaren iragarpena ikusiko dugu. Demagun *giro* eguzkitsua iragarrita dagoela. Orduan, ezkerreko adarretik joko dugu eta ondoren, *hezetasuna* aztertuko dugu. Demagun orain hezetasun handia egongo dela. Hau jakinda, erabakia ez jokatzearena izango dela aurreikusiko du sistemak eta erabakiaren azalpena nabaria da: ez dugu jokatuko giro eguzkitsua eta hezetasun handia egongo delako. Hostoetan, *Jokatu* ala *Ez jokatu* klaseekin batera, entrenamenduko kasuen artean hosto bakoitzari zenbat kasu dagozkion adierazten da parentesi artean. Ikus daitekeen bezala, kasu honetan sortutako zuhaitzean ez da *hozberoa* aldagaia agertzen. Orokorrean, gerta daiteke atributu bat eta kla-

searen arteko erlazioa esanguratsua ez izatea eta ondorioz, sailkatzailean aldagai hori kontuan ez hartzea.

Datu-basean ditugun 14 kasuekin froga egiten badugu, guztietarako asmatzen duela ikusiko dugu. Baina nola jakin kasu berriak iristean asmatuko duen ala ez?

Horretarako hainbat estimazio-teknika daude eta esperimentuaren ezaugarrien arabera komenigarria gerta daiteke bat edo beste erabiltzea. Orokorrean denek antzeko ideiari eusten diote: ikasteko ditugun datuak bi multzotan banatu eta batzuk ikasketarako erabili eta gero, besteak sailkatzaileak ondo edo gaizki sailkatuko lituzkeen aztertzeko erabili. Hainbat aldiz errepikatzen da prozesu hau eta batez besteko emaitzetan oinarrituta, etorkizuneko datuekin izango litzatekeen portaeraren estimazio bat egiten da. Kasu honetan, *10-fold cross validation* teknika erabili dugu, bera baita honelako lanetan gehien erabiltzen den tekniketako bat (Wekak berak aukera bezala eskaintzen du); hala ere, orokorrean kasu gehiagoko datu-baseekin erabiltzen da. Teknika honetan, labur azalduta, dugun lagina 10 zati disjuntutan banatzen da. Honela, zatietako bakoitzarekin ondokoa egiten da: beste zati guztietako kasuak ikasteko erabili eta zati honetakoak kasu berriak balira bezala, sailkatzaileak ondo sailkatuko lituzkeen ala ez aztertu. Gero, 10 zatiekin lortutako emaitzen batez bestekoak kalkulatu eta horiek hartzen dira sailkatzailearen etorkizuneko errendimenduaren estimatzaile bezala.

Eta zer diote egindako estimazioek gure adibidean? *10-fold cross validation* bidez eginiko estimazioen arabera, sailkatzaileak kasu berrien erdian asmatu eta beste erdian huts egingo du eta hori, oso asmatze tasa baxua da. Kasu honetan ordea, espero zitekeen zerbait da. Orokorrean kasu asko beharko da ikasketa bat aurrera eramateko eta kasu honetan, ikasteko kasuak gutxiegi dira. Jarraian (3. irudia) ikus daiteke Wekak emaitza hauek nola erakusten dizkigun:

Classifier output

```
Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances           7               50      %
Incorrectly Classified Instances         7               50      %
```

3. irudia. Weka softwarearekin *10-fold cross validation* baten ondoren lortutako emaitzak.

Baina kasu kopurua handia izanda ere, gerta daiteke klaseetako bati dagozkion kasuetatik besteari dagozkionak baino askoz gehiago izatea eta desoreka horrek ere sailkatzailea gaizki ikastera eramatea; ondorioz, emaitza txarrak lortuko dira. Emaitza txarrak lortzea ez da bakarrik asmatze tasa txikia lortzea, gerta daiteke oso asmatze tasa handia lortu baina emaitzak txarrak izatea. Eta nola uler daiteke hori? Demagun populazioaren % 0,1ak besterik ez duen gaixotasun baten detekzioan gabiltzala lanean. % 99,9ak ez du gaixotasun hori eta hori dela-eta, adimenik gabeko sailkatzaile batek beti osasuntsu gaudela esango baligu, % 99,9ko asmatze tasa izango luke. Baina emaitzak ontzat jo ditzakegu? Hain justu garrantzitsuenak diren kasuetan egiten du huts sailkatzaileak.

Eta nola eman dakieke irtenbidea honelako arazoei? Datuen birlaginketa da modurik erabilienetako bat; [4]n daukagu honi buruzko lan interesgarri bat; bestela esanda, klase ezberdinei dagozkien kasuen proportzioa edo kopurua aldatuz kasu berriak sortu edo existitzen direnak ezabatzea, adibidez.

Jarraian birlaginketa metodoen artean ezagunenetakoak azalduko ditugu.

3. **BIRLAGINKETA METODOAK**

Birlaginketa teknikak bi multzo nagusitan bana ditzakegu. Alde batetik, azpilagunketa teknikek ezabatu egiten dituzte existitzen diren kasuak; beste alde batetik, badaude gainlaginketa teknikak, kasu berriak sortzen dituztenak.

Batzuen zein besteen artean, kasuak ausaz edo kalkuluren bat eginez aukeratzen dituzten teknikak daude. Azken hauei, birlaginketa teknika adimentsu deritze eta kasuen arteko antzekotasunaren neurketan oinarritzen dira. Baina nola neur daiteke kasuen arteko antzekotasuna? Distantzia-funtzio bat erabiltzen da; zenbat eta bi kasuren arteko distantzia txikiagoa izan, esango dugu orduan eta antzekoagoak direla bi kasuak.

3.1. **Kasuen arteko distantzia**

Bi kasuren arteko distantzia edo diferentziaren kalkulua ez da gauza hutsala. Aldagai guztiak zenbakizko aldagaiak balira, ez legoke zailtasun handirik, eta distantzia euklidear normalizatua erabiltzea da ohikoena halako kasuetan. Zergatik normalizatua? Demagun bi zenbakizko aldagai ditugula: *adina* eta *zenbat diru duen kontu korrontean*. Adinaren balioak 0 eta 100 bitartekoak izango dira gutxi gorabehera, baina diruari dagokionez, balio tartea askoz ere handiagoa da. Eta zeintzuk ote dira antzekoagoak, adinari dagokionez 2 eta 80 urte ala diruari dagokionez 100.000 eta 100.500 euro? Hasiera batean adinaren balioetan desberdintasun nabarmenagoa dela dirudien arren, normalizatu ezean, bigarrenak distantzian eragin handiagoa izango du. Izan ere, lehen bien arteko diferentzia 78 den bitartean, bigarren bikotearen arte-

koa 500 da. Hori dela-eta, aldagai bakoitzak har dezakeen balio tartearekiko edo desbideratze estandarrarekiko normalizatzen dira balioak, eta horrela, aldagai guztiek eragin berdina dute distantziaren kalkuluan.

Aldagai guztiak zenbakizkoak balira, x eta y kasuen arteko distantzia euklidearra, desbideratze estandarrarekiko normalizatua, honela kalkulatuko litzateke:

$$d(x,y)=\sqrt{\sum_{i=1}^{n}\frac{1}{\sigma_i^2}(x_i-y_i)^2}.$$

Non x_i eta y_i, x eta y kasuek i-garren aldagairako hartzen dituzten balioak diren eta σ_i, i-garren aldagaiko balioen artean dagoen desbideratze estandarra den.

Baina zer gertatzen da aldagaiak zenbakizkoak ez badira? Zein da adibidez *eguraldia lainotsu* eta *eguzkitsu* izatearen arteko distantzia?

Mota honetako aldagaiei aldagai diskretuak deritze eta hauen arteko distantzia kalkulatzeko bi modu ikusiko ditugu.

— ***Overlap* edo gainezarmen distanzia**: Aukerarik errazena da. Bi kasuk aldagai diskretu baterako dituzten balioak berdinak badira, distantzia 0 da, eta bestela, 1.

— **VDM distantzia (*Value Difference Metric*)**: Stanfill eta Waltzek [5]-en aurkeztutako distantzia honen kalkulua zailagoa da. Bi balio antzekoagotzat jotzen dira sailkatzerako orduan, emaitza antzekoagoak ematen badituzte. v aldagai baten a eta b balioen arteko VDM distantzia honela kalkulatzen da:

$$vdm_v(a,b)=\sum_{k=1}^{K}\left|P_{v,a,k}-P_{v,b,k}\right|^q,$$

non K klase kopurua den, q balio konstante bat (orokorrean 1 edo 2) eta $P_{v,a,k}$, k klasea izateko probabilitatea v atributuaren balioa a denean. Hau da,

$$P_{v,a,k}=\frac{N_{v,a,k}}{N_{v,a}},$$

$N_{v,a}$ izanik entrenamendurako laginean v aldagaiarentzat a balioa duten kasuen kopurua, eta $N_{v,a,k}$, entrenamendurako laginean v aldagaiarentzat a balioa izanda k klaseari dagozkion kasuen kopurua.

Gure adibidera itzulita, demagun *eguzkitsu* eta *lainotsu*-ren arteko diferentzia kalkulatu nahi dugula. Kasu honetan, *v* aldagaia *eguraldiaren iragarpena* da, *a* balioa *eguzkitsu* eta *b* balioa *lainotsu*. Klase posibleak berriz bi dira, *Jokatu* eta *Ez jokatu*. Gauzak honela, egin ditzagun kalkuluak:

$$vdm_{\text{eguraldiaren iragarpena}}(eguzkitsu, lainotsu) = \left(\frac{3}{5} - \frac{0}{4}\right)^2 + \left(\frac{2}{5} - \frac{4}{4}\right)^2 = 0,72.$$

Honen arabera, *eguzkitsu* eta *lainotsu* nahiko balio ezberdinak dira, diferentzia 1etik gertuago baitago 0tik baino.

Amaitzeko, nola kalkulatu bi kasuren arteko distantzia aldagai batzuk diskretuak eta beste batzuk zenbakizkoak direnean? Atributuz atributu diferentzia kalkulatzen da, zenbakizkoen kasuan distantzia euklidear normalizatua eta diskretuenean, *overlap* erabiliz; kasu honetan, HEOM distantzia deritzo edo *vdm* (HVDM distantzia). Era formalean adierazita, hauxe dugu *n* atributuko *x* eta *y* kasuen arteko distantzia:

$$HVDM(x,y) = \sqrt{\sum_{a=1}^{n} d_a^2(x_a, y_a)},$$

non $d_a^2(x_a, y_a)$ distantzia euklidearra den, zenbakizko aldagaien kasuan eta VDM distantzia aldagai diskretuenean.

Eta distantzia edo diferentzia nola kalkulatzen den ikusita, ikus ditzagun azpilaginketa eta gainlaginketa direlako teknika adimentsuak.

3.2. **Azpilaginketa metodoak**

— **Ausazko azpilaginketa** (*random subsampling*)**:** Dagoen laginketarik sinpleena da. Klase bakoitzetik behar adina aldiz kasu bat zoriz aukeratu eta ezabatzen du. Demagun gure adibideko datu-baseko bi klaseetan kasu kopuru berdina egotea nahi dugula. *Ez jokatu* klaseko 5 elementu eta *Jokatu* klaseko 9 ditugula jakinik, *Jokatu* klaseko 4 kasu ausaz aukeratu eta kenduko lirateke.

— **ENN** (*Edited Nearest Neighbor*)**:** Wilsonek [6] proposatu zuen teknika hau 1972. urtean. Azpilaginketa teknika bada ere, ez du aukerarik ematen zenbat kasu kendu erabakitzeko. Datu guztien artean «okerrak» direnak erabaki eta ezabatzen ditu eta horregatik, garbiketa teknika bat dela ere esaten da. Eta nola erabakitzen da kasu bat okerra den ala ez? Kasu bakoitzerako datu-baseko beste kasu guztiekiko

distantziak kalkulatu eta gertuen dauden hirurak aukeratzen dira. Hiru hauetako biren edo gehiagoren klasea eta uneko kasuarena ezberdinak badira, uneko kasua okertzat jo eta ezabatu egiten da. Teknika honek beraz ez du klase-banaketa jakin bat lortzeko balio eta klaseak orekatu nahi ditugunean beste teknikaren batekin konbinatzen da.

— **Tomek links:** Tomekek [8] proposatutako hau ere garbiketa teknika bat da eta *Tomek* lotura batean parte hartzen duten kasuak dira kentzen direnak (loturan parte hartzen duten bi kasuak edo soilik kasu gehien duen klaseari dagokiona ken daitezke). Zer da *Tomek* lotura bat? Bi kasuren artean *Tomek* lotura bat dagoela esaten da baldin eta bi kasu horiek klase ezberdinekoak badira eta biak elkarrengandik gertuen dauden kasuak badira, hau da, ez badago beste kasurik batetik edo bestetik gertuago.

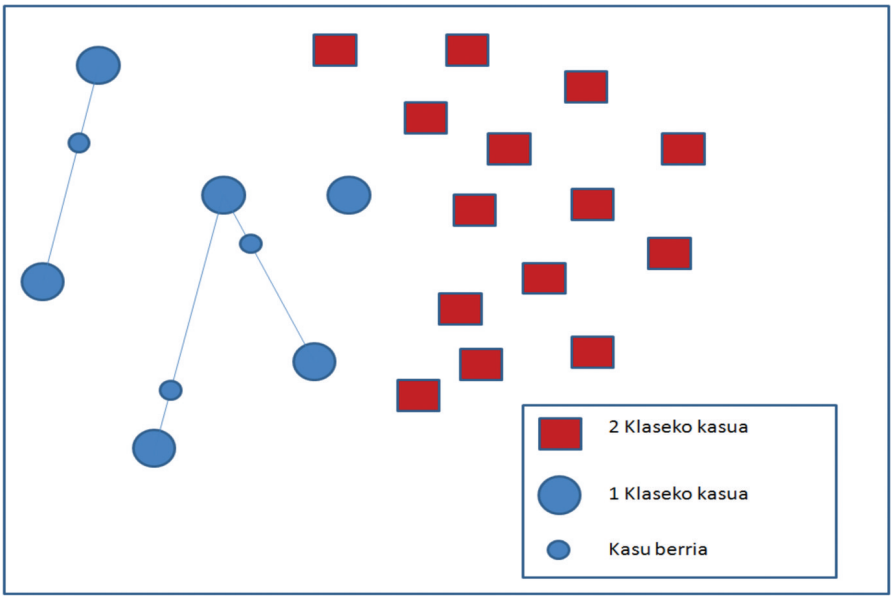
— **SKT:** ENNren antzeko ideia jarraituz baina ezabatzeko erabakia findu nahian gure taldeak [7] proposatutako metodoa da. Bere izena euskal kirola den Sokatiratik dator, ideia ere handik izan genuen-eta. Klase bakoitza tiraka dabilen taldea balitz bezala ulertuta, kasu bakoitzerako berarengandik gertuen dauden *k* kasuak kalkulatu eta klase bakoitzekoak alde batera tira egiten dutela suposatzen da. Kasu bakoitzak egiten duen indarra aztertzen ari garen kasuarekiko duen distantziarekiko alderantziz proportzionala da (gero eta gertuago, indar handiagoz erakartzen du). *k* kasuen artean klase batekoek zein bestekoek egiten dituzten indarren baturak alderatuta beste klasekoek gutxieneko diferentzia batekin irabazten badute, kasua ezabatu egiten da. Bestela, kasua bere horretan uzten mantentzen da.

3.3. **Gainlaginketa metodoak**

— **Ausazko gainlaginketa** (*Random oversampling)*: Meto honetan, kasu bat nahi adina aldiz aukeratzen da zoriz. Demagun berriro gure adibideko datu-baseko bi klaseetan kasu kopuru berdina egotea nahi dugula. *Ez jokatu* klaseko 5 elementu eta *Jokatu* klaseko 9 ditugula jakinik, *Ez jokatu* klaseko 4 kasu ausaz aukeratu eta errepikatuko lirateke. Honekin, atributuek klase aldagaiarekiko duten erlazioa aldatzen da, eta horrela sailkatzaile ezberdin bat lortzen da.

— **SMOTE** (*Synthetic Minority Oversampling TEchnique*): Gainlaginketa teknika honek jadanik badauden kasuak errepikatu beharrean, haietan oinarrituta kasu berriak sortzen ditu sintetikoki. Chawla *et al*-ek [9] proposatu zuten metodo honetan, nahi adina aldiz aukeratzen da ausaz kasu gutxien duen klaseko kasu bat; ondoren bera eta berarengandik gertuen dagoen bere klaseko *k* bizilagunen arteko bat

lotzen dituen zuzenaren punturen batean kasu berri bat sortzen da. Horretarako, aukeratutako kasua eta bizilagunaren arteko diferentzia bektorea kalkulatzen da (atributu bakoitzerako diferentzia); ondoren, 0 eta 1 arteko ausazko zenbaki batekin biderkatu eta azkenik, gure kasuaren ezaugarri bektoreari gehitzen zaio, lortzen den bektorea kasu berria izanik. Aldagai diskretuen kasuan, gertueneko bizilagunen artean gehien errepikatzen den balioa egokitzen zaio kasu berriari. Jarraiko irudian ikus daiteke (4. irudia) SMOTEren bidez 3 kasu berrien sorkuntza. Bertan, kasurik gutxien duen klaseko elementuak borobilen bidez adierazten dira, eta borobil txikiagoz, irudikatzen dira sortzen diren kasu berriak. Hauek klase bereko elementuen artean kokatuak egongo dira beti.



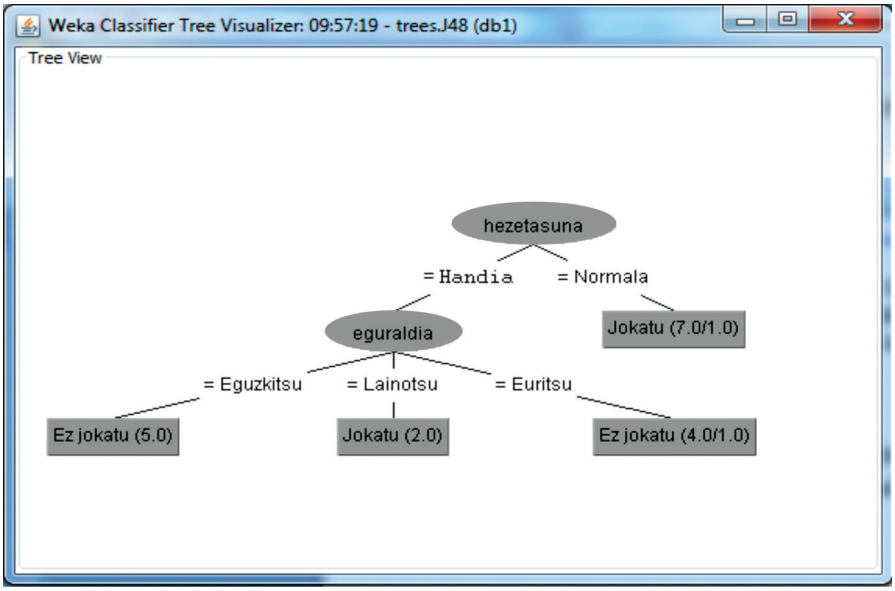
4. irudia. SMOTE erabiltzean aukeratutako kasua eta klase berekoak diren gertueneko *k* kasutako bat lotzen dituen lerroko punturen batean sortzen da kasu berria.

Beti ere gure adibidean, bi klaseetatik kasu kopuru berdina eduki nahiko bagenu, lehen 5 kasu izanik lehenengo klasekoak (*Ez jokatu*) eta 9 kasu 2. Klasekoak (*Jokatu*), lehenengo klaseko 4 elementu berri sortu beharko genituzke, eta SMOTE aukeratzen badugu 4 kasu berri horiek sortzeko, honako hau (2. taula) da emaitza posibletako bat (SMOTEk zoria erabiltzen duenez, aplikatzen den bi alditan ez du zertan emaitza berdina eman beharrik).

2. taula. Jatorrizko laginari SMOTE erabili ondoren lortzen den lagina, azken lau kasuak izanik sortutako kasu berriak. Azken bi zutabeetan ondoren ikusiko ditugun adibideetarako informazioa dago ikusgai, garbiketa tekniketako bakoitzak zein kasu ezabatuko lukeen adierazita.

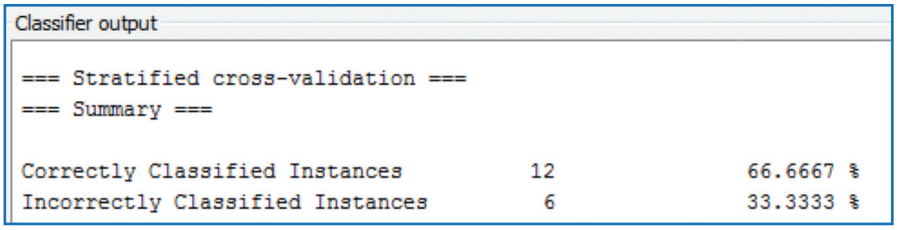
Eguna	Giroa	Hozberoa (gradutan)	Hezetasuna	Haizea	Jokatu	ENN	SKT
1	Eguzkitsu	30	Handia	Ahula	Ez		
2	Eguzkitsu	32	Handia	Bortitza	Ez		
3	Lainotsu	28	Handia	Ahula	Bai		
4	Euritsu	22	Handia	Ahula	Bai	✗	✗
5	Euritsu	11	Normala	Ahula	Bai		
6	Euritsu	14	Normala	Bortitza	Ez	✗	✗
7	Lainotsu	15	Normala	Bortitza	Bai		
8	Eguzkitsu	19	Handia	Ahula	Ez		
9	Eguzkitsu	13	Normala	Ahula	Bai		
10	Euritsu	19	Normala	Ahula	Bai		
11	Eguzkitsu	20	Normala	Bortitza	Bai	✗	✗
12	Lainotsu	20	Handia	Bortitza	Bai		✗
13	Lainotsu	29	Normala	Ahula	Bai		
14	Euritsu	21	Handia	Bortitza	Ez		
15	Eguzkitsu	17	Handia	Bortitza	Ez		
16	Euritsu	19	Handia	Bortitza	Ez		
17	Euritsu	18	Handia	Bortitza	Ez		
18	Eguzkitsu	30	Handia	Bortitza	Ez		

Behin datuen birlaginketa eginda, ikasketa-algoritmoa erabiltzen da eta ikus daitekeenez (5. irudia), lortzen dugun sailkatzailea aurretik lortutakoaren ezberdina da. Hostoetan parentesi artean agertzen diren balioei so eginez, ikus dezakegu eskuinaldeko bi hostoetako bakoitzean parentesi artean bi balio ageri. Horren arabera, entrenamenduko kasuekin froga egin eta hosto horietako bi kasuetan (4. eta 14. kasuetan) gaizki sailkatutako kasu bana dago, hau da, kasu honetan sailkatzaileak ez ditu entrenamenduko kasu guztiak ondo sailkatzen. Baina zer gertatuko ote da kasu berriak sailkatzean?



5. irudia. Hasierako laginari SMOTE erabili eta gero Wekako J48 algoritmoak eraikitzen duen sailkatzailea.

Berriro ere *10-fold cross validation* teknika aplikatu eta, ondoko irudian (6. irudia) erakusten den bezala, kasu berrien % 67-an asmatuko litzatekeela estimatzen da, lehen baino asmatze tasa handiagoa.



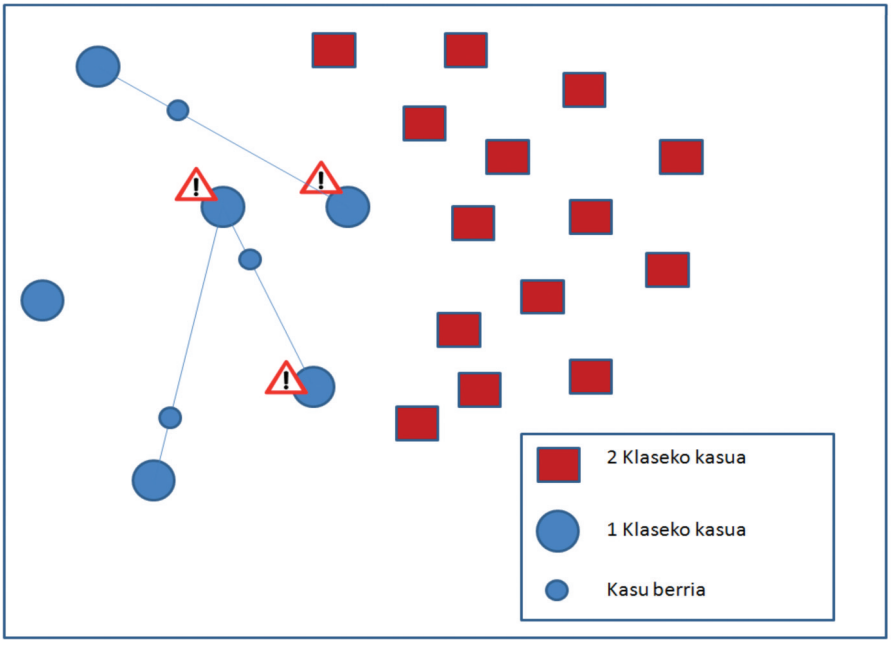
6. irudia. Weka softwarearekin *10-fold cross validation* baten ondoren lortutako emaitzak jatorrizko laginari SMOTE erabili ondoren.

— **Borderline-SMOTE:** Mugaldeko SMOTE, aurretik ikusi dugun SMOTE beraren oso antzekoa da. Bi aldaera proposatu zituzten bere egileek [10]: Borderline-SMOTE1 eta Borderline-SMOTE2. Desberdintasun nagusi bat dute biek SMOTErekin; azken honek kasu berriak sortzerako orduan gutxiengo klaseko kasu guztiak hartzen ditu kontutan eta aurreko bi horiek berriz, bi klaseen arteko mugan

dauden kasuak bakarrik hartzen dituzte. Egileen hitzetan, mugako kasu hauek arriskuan daude eta *Danger* kasuak deitzen diete. Horretarako, kalkulatzen da gutxiengo klaseko kasuen artean arriskuan zeintzuk dauden. Kasu bat arriskuan dagoela onartzen da, baldin eta bere *m* gertueneko kasuen artean erdia edo gehiago beste klasekoak badira (*m*, metodoaren parametro bat izanik, *k*-ren ezberdina).

Orain, kasu berri bat sortzen da, nahi adina aldiz arriskuan daudenen artean bat zoriz aukeratuz lehen bezala, bera eta gertueneko kasutako bat lotzen dituen lerroan. Borderline-SMOTE1en kasuan, bere klasekoak diren gertueneko *k* kasuak hartzen dira kontutan, baina Borderline-SMOTE2k zein klasekoak diren aztertu gabe aukeratzen ditu gertueneko *k* kasu horiek eta azkenean, ausaz aukeratutakoa ez bada gure kasuaren klase berekoa, 0 eta 1 arteko zenbaki batekin bidertu ordez, 0 eta 0,5 arteko batekin bidertzen da, gure kasutik gertuago egon dadin.

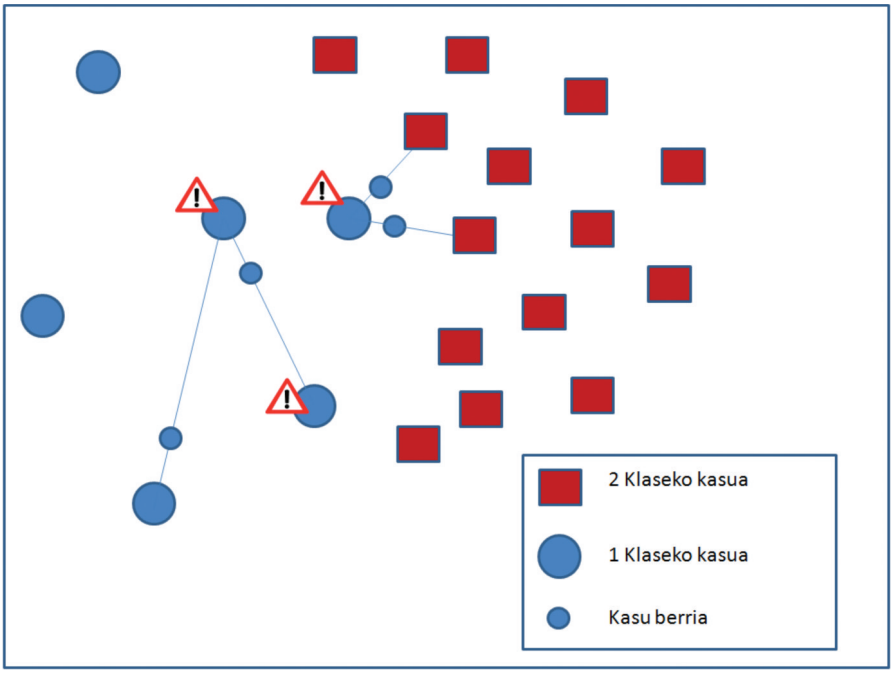
Jarraiko irudian (7. irudia) Borderline-SMOTE1 metodoaren aplikazioaren adibide bat ikus daiteke. Bertan, gutxiengo klaseko kasuak borobil batez adieraziak daude. Hauen artean arriskuan daudenei



7. irudia. Borderline-SMOTE1-en, lehenbizi gutxiengo klasean arriskuan dauden kasuak bilatzen dira. Ondoren, hauen eta klase bereko gertueneko kasuetako baten artean kasu berri bat sortzen da.

arrisku seinale bat jarri zaie eta kasu berriak (borobil txikiz adieraziak), kasu hauen eta bere klaseko gertuenetakoen artean sortzen dira.

Hurrengo irudian berriz (8. irudia) Borderline-SMOTE2 metodoaren aplikazioaren adibidea erakusten da. Berriro ere, gutxiengo klaseko kasuak borobil batez adieraziak daude eta arriskuan daudenek arrisku seinale bat dute alboan. Oraingo honetan ordea, kasu berriak, kasu hauen eta edozein klasetako gertuenetakoen artean sortzen dira. Hori bai, beste klaseari dagozkion kasuak aukeratzen diren kasuetan, gure kasutik beste klasekotik baino gertuago kokatzen da beti kasu berria.



8. irudia. Borderline-SMOTE2-n, lehenbizi gutxiengo klasean arriskuan dauden kasuak bilatzen dira. Ondoren, hauen eta edozein klaseko gertueneko kasuetako baten artean kasu berri bat sortzen da.

3.4. **Laginketa tekniken konbinazioa**

Ohiko joera bat da teknika bakar bat erabili ordez teknika ezberdinak konbinatzea. Esate baterako, gainlaginketa teknika batekin batera garbiketa teknika bat erabil daiteke. Honen adibide dira jarraian ikusiko ditugun biak, non SMOTE bezalako gainlaginketa metodoa eta ENN garbiketa teknika batera erabiltzen diren.

— **SMOTE-ENN**: Lehenbizi nahi adina kasu sortzeko SMOTE erabiltzen da eta ondoren ENN teknikak garbiketa egiten du okertzat jotzen dituen kasuak ezabatuz. Datu-base txiki eta oso desorekatuei aurre egiterako orduan, egile batzuek erreferentziatzat jotzen dute metodo hau beraien lanetan [11][12].

Gure adibidean SMOTE erabili ondoren lortutako laginari ENN ezartzen badiogu, honek hiru kasu kentzeko erabakia hartzen du. SMOTE erabili eta gero lortutako lagina adierazten duen taulan (2. taula): ENN zutabean adierazita daude ENN aplikatuta ezabatuko liratekeen hiru kasuak. Eta lortzen den laginarekin lortutako sailkatzailearen errendimenduaren estimazioa eginez, % 86ko asmatze tasa lortzen da. Ondoko irudian (9. irudia) ikus daiteke Wekan egindako saioaren emaitza.

Classifier output

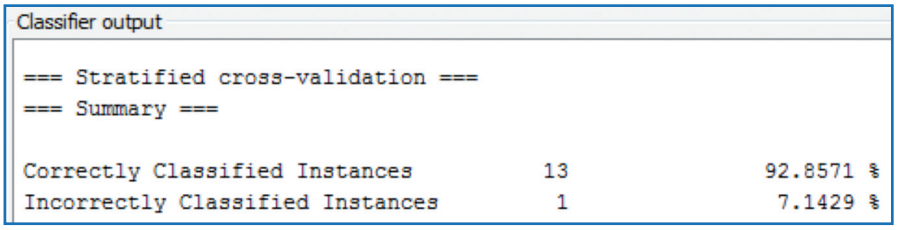
```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          13               86.6667 %
Incorrectly Classified Instances         2               13.3333 %
```

9. irudia. Weka softwarearekin *10-fold cross validation* baten ondoren lortutako emaitzak jatorrizko laginari SMOTE-ENN aplikatu ondoren.

— **ENN-SMOTE**: Antzeko ildoari jarraiki, kasu berriak sortu baino lehen lagineko kasuen artean «okertzat» jotzen direnak ezabatu eta behin garbiketa eginda SMOTEren bidez kasu berriak sortzea da ideia. Teknika hau gaur arte argitaratu gabe dago eta gure taldeak [7] egiten duen proposamena da. Gure esperimentuetan SMOTE-ENNek bezain emaitza onak eman ditu. Esperimentu horietan tamaina eta klase-banaketa ezberdinetako datu-baseen multzo handi batekin egin ditugu frogak eta antzeko emaitzak lortzeaz gain, SMOTE-ENN baino metodo azkarragoa dela ikusi dugu.

Hauetaz gain, gure ikerkuntza lanean **SMOTE-SKT** ere erabiltzen ari gara. Izenak argi adierazten duen bezala, behin SMOTE erabilita SKT garbiketa teknika erabiltzen dugu. ENN-rekin bezala, SMOTE erabili eta gero lortutako lagina adierazten duen taulan (2. taula), azken zutabean adierazita dago SKT teknikak zein kasu ezabatuko lituzkeen. Ondoko irudian (10. irudia) ikus daitekeenez, Wekarekin egindako *10-fold cross validation* baten araberako estimazioak % 92-ko asmatze tasa ematen du.



10. irudia. Weka softwarearekin *10-fold cross validation* baten ondoren lortutako emaitzak jatorrizko laginari SMOTE-SKT aplikatu ondoren.

Argi dago ikusi berri duguna adibide txiki bat besterik ez dela eta aipatutako teknikak ulertzeko erabili da, ez tekniken arteko konparazioa egin eta teknika egokiena finkatzeko. Tekniken arteko konparazioak egiteko datu-base askorekin egin behar dira saiakerak, teknikak baldintza ezberdinetan frogatzeko; behin saio guztiak eginda eta emaitzak aztertuta, froga estatistiko batzuk aplikatzen dira lortutako emaitzak esanguratsuak diren ala ez ikusteko.

4. **ONDORIOAK**

Ikasketa automatikoaren inguruan, beharrezkoa izaten da hainbat arazori aurre egiteko datuen aurre-prozesamendua. Datuen birlaginketa da aurreprozesamendua egiteko aukeren artean erabilienetako bat, sailkatzaile sortu baino lehen kasu berriak sortu edo existitzen direnak kentzen dituena.

Lan honetan, gaur egun erabiltzen diren birlaginketa metodo batzuen azterketa egin nahi izan dugu, gure proposamen pare batekin batera (SKT eta ENN-SMOTE), arlo honen inguruan lan nola egiten den erakusteko asmoz. Ikuspegi orokor bat ematea izan da gure helburua, eta jakin-mina piztuz gero, bibliografian zehar hainbat eta hainbat lan aurki daitezke honen guztiaren inguruan, lan honetan aipatzen diren lanetatik hasita.

5. **BIBLIOGRAFIA**

[1] MITCHEL T. 1997. *Machine Learning*. McGraw-Hill. New York.

[2] QUINLAN J. R. 1986. «Induction of decision trees». *Machine Learning*. **1**, 81-106

[3] HALL M., FRANK E., HOLMES G., PFAHRINGER B., REUTEMANN P., WITTEN I. H. 2009. «The WEKA Data Mining Software: An Update». *SIGKDD Explorations*. **11**.

[4] HE H., GARCÍA E. 2009. «Learning from Imbalanced». *IEEE Transactions on Data Knowledge and Data Engineering*. **21**, 1263-1284.

[5] STANFILL C., WALTZ D. 1986. «Toward memory-based reasoning». *Communications of the ACM* **29**, 1213-1228

[6] WILSON D. L. 1972. «Toward memory-based reasoning». *IEEE Transactions on Systems, Man and Cybernetics* **2**, 408-421

[7] ALDAPA taldea (ALgorithms, DAta mining & PArarelism). *http://www.sc.ehu.es/aldapa*

[8] TOMEK I. 1976. « Two Modifications of CNN ». *IEEE Transactions on Systems, Man and Cybernetics* **6**, 769-772.

[9] CHAWLA N. V., BOWYER K. W., HALL L. O., KEGELMEYER W. P. 2002. «SMOTE: Synthetic Minority Over-sampling Technique». *JAIR* **16**, 321-357.

[10] HAN H., WANG W., MAO B. 2005. «Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning». *International Conference on Intelligent Computing 2005*, 878-887.

[11] BATISTA G. E. A. P. A., PRATI R C., MONARD M. C. 2004. «A study of the behavior of several methods for balancing machine learning training data». *SIGKDD Explorations Newsletter, ACM*. **6**, 20-29.

[12] GARCÍA S., FERNANDEZ A., HERRERA F. 2009. «Enhancing the Effectiveness and Interpretability of Decision Tree and Rule Induction Classifiers with Evolutionary Training Set Selection over Imbalanced Problems». *Applied Soft Computing,* **9**, 1304-1314.