Fontos jellemzők kiválasztása permutációk segítségével

Szakdolgozat

Dudás László

Témavezető: Lukács András
Számítógéptudományi tanszék

Eötvös Loránd Tudományegyetem
Természettudományi Kar

Tartalomjegyzék

Kivonat

Az adatbányászat egyik legfontosabb témája a klasszifikáció, ugyanolyan ismérvekkel rendelkező adatpontok osztályokba sorolása. Ennek a problémának a részfeladata azon jellemzők kiválasztása, amelyek alapján már nagy pontossággal be lehet sorolni az egyes egyedeket. Dolgozatunkban előbb részletesen bemutatjuk az erre a részfeladatra ismert megoldások típusait, majd empirikusan is megvizsgálunk egy új gyorsítási ötletet. A következtetések alapjául szolgáló eljárást egy adatbányászati keretrendszert kiegészítve valósítottuk meg. Tapasztalatunk szerint a módszer jelentős mértékben gyorsítja a kiválasztás folyamatát, és széles körben alkalmazható.

1. Bevezetés

1.1. Az adatbányászat kialakulása

A számítógépek elterjedése és a tárolóegységek kapacitásának rohamos növekedése hatalmas mennyiségű adattömeg keletkezését idézte elő. Sok üzleti érdek kötődik ahhoz, hogy a vállalatok nagy adatbázisaikból használható tudást nyerjenek. A hatalmas adatmennyiség miatt nem lehetséges az adatok pusztán emberi erővel történő elemzése, valamint a komplex összefüggéseket a hagyományos statisztikai módszerek sem mutatják ki. Ezt a problémát megoldandó jött létre az 1980-as években az adatbányászat a statisztika, a kombinatorika, a diszkrét matematika, az informatika, valamint a mesterséges intelligencia találkozásánál. Az adatok pusztán statisztikai elemzése még nem tekinthető adatbányászatnak. Sokan, sokféleképpen próbálták definiálni az adatbányászat fogalmát[1][4], így jelen dolgozatban nem célunk új meghatározás alkotása; azonban megnézzük mi a közös a legtöbb definícióban. A talált közös pontok:

- **Hasznos információ, mintázatok** - tehát olyasvalamit keresünk, amit később valaki fel tud használni, emberileg felfogható jelentéssel
- **kinyerése, felfedezése** - az adat rendelkezik információtartalommal
- **nagy adatbázisokból.** - az adat mérete nagy és strukturált. Nem érdemes az adatbányászat eszköztárát alkalmazni, hogyha csak egy-két változó van, mivel ilyenkor más (statisztikai, vagy vizualizációs) módszerek eredményesebbek.

1.2. Az adat struktúrája

Az adat, amivel dolgozunk relációs, táblázat jellegű: sorokból és oszlopokból áll. Az adattábla egy sora egy bejegyzésnek, egy adatpontnak, egy rekordnak; egy oszlop pedig egy tulajdonságnak, attribútumnak, jellemzőnek felel meg. A táblázat egy cellájába értelemszerűen a megfelelő rekord megfelelő tulajdonságértéke kerül. A rekordok tipikusan különböző egyedek, objektumok, vagy egy idősor adatai.

Többféle attribútum lehetséges[1]:

- **Kategória:** az attribútum csak diszkrét értékeket vehet fel, és az értékek között csak egyenlőséget tudunk vizsgálni. Speciális eset, amikor két kategória van. Ezeket a továbbiakban bináris attribútumnak hívjuk . Pl.: nemzetiség (kategória), nem (bináris)
- **Sorrend:** az attribútum értékkészletét sorba tudjuk rendezni, azaz rendezést adhatunk meg rajta. Pl.: tiszta–borús–esős
- **Intervallum:** nem csak rendezést hanem egy "+" műveletet is megadunk, amellyel csoportot alkotnak a változó értékei.
- **Arány:** olyan intervallum típusú jellemző, amely lehetséges értékei között van kitüntetett zérus elem. Pl.: idő

Bár az attribútumok nem mindig számértékűek, feltehetjük, hogy helyettesíthetjük őket valós, vagy természetes számokkal. Ha az attribútumokra, mint valószínűségi változókra tekintünk, akkor az alábbiak szerint definiálhatjuk az eddigi fogalmakat.
Az eseményteret jelöljük szokás szerint Ω-val! Feltesszük, hogy Ω minden eleme esemény is.

1.2.1. Definíció (Attribútum). *Egy $A\colon \Omega \to \Re$ valószínűségi változót attribútumnak hívunk.*

1.2.2. Definíció (Rekord). *Legyen $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$ attribútumok egy véges rendezett halmaza (a vizsgált jellemzők), valamint $\omega \in \Omega$! Ekkor az ω esemény $\mathcal{A}$-hoz tartozó rekordja $r = (A_1(\omega), A_2(\omega), \ldots, A_n(\omega))$.*

Mivel legtöbbször nyilvánvaló lesz, hogy mi az $\mathcal{A}$, gyakran azonosítjuk majd ω-t az ő rekordjával.

1.2.3. Definíció (Adattábla). *A R adattábla egy $\Re^{r\times n}$ mátrix amelynek minden sora egy-egy rekord, melyek ugyanazon A attribútumhalmazhoz tartoznak.*

Matematikai modellünkben feltettük, hogy az $A_i(\omega)$ mindig létezik. A valóságban a táblázatokban természetesen szerepelhetnek hiányzó vagy hibás értékek. Ezért már az előfeldolgozás során érdemes kiszűrni a kiugró adatokat, melyek valószínűleg hibásan szerepelnek, kitölteni a hiányzó értékeket, vagy akár elhagyni a hiányos oszlopokat vagy sorokat.

2. Klasszifikáció

2.1. Gépi tanulás

Az adatbányászat által vizsgált adat nagy mérete miatt csak számítógépes feldolgozás jöhet szóba. Az információ kinyeréséhez a gépi tanulás eszközeit használjuk. Gépi tanulás során a számítógép korábbi, meglévő adatok alapján kifejleszt egy adott modellt. Az adatbányászati alkalmazások esetében a modell megtanul felismerni mintázatokat, vagy döntést hoz egy adott kérdésben.

2.2. A klasszifikációs feladat

Gyakori feladat, hogy egy ismeretlen értékű, vagy előre nem megfigyelhető változó értékét megjósoljuk más, ismert változók tükrében. Ha az ismeretlen változó csak meghatározott diszkrét értékeket vehet fel, akkor klasszifikációnak, osztályba sorolásnak hívjuk ezt a feladatot. Például ilyen, ha egy telefontársaságnál szeretnénk jelezni a szolgáltatás felmondását fontolgató ügyfeleket, cukorbetegséget diagnosztizálni fiziológiai jellemzők alpaján, vagy csillagok típusát meghatározni egyes spektrumtartományaik intenzitásai alapján.

Az osztályozás során a kitüntetett C célattribútumot címkének is hívjuk, az osztályozás folyamatára pedig címkézésként, becslésként és jóslásként is hivatkozunk. C lehetséges értékei $c_1, c_2, \ldots, c_k$ ismertek. Vannak rekordjaink, melyekre tudjuk C értékét (ez a felügyelt gépi tanulás), feladatunk pedig, hogy becslést adjunk az ismeretlen célváltozókra.

Outlook	Temp	Windy	Friends Online	Days Until Exam	Activity (class)
rainy	16	true	6	6	play online
sunny	26	false	3	1	study
sunny	23	true	2	4	play outside
overcast	20	true	7	5	play online
rainy	13	false	4	2	study
overcast	19	false	4	5	play outside
overcast	25	false	5	1	study

1. ábra. Adattábla célváltozóval

2.2.1. Definíció (Klasszifikátor). *Ha $\mathcal{A}$ attribútumok egy halmaza, R_A pedig az $\mathcal{A}$-ban szereplő tulajdonságokkal rendelkező rekordok halmaza,valamint $C \notin \mathcal{A}$ osztálycímke értékkészlete $\bar{C} = \{c_1, c_2, \ldots, c_k\}$, akkor a $K : R_A \to \bar{C}$ függvényeket klasszifikátoroknak hívjuk.*

2.3. Klasszifikátor modellek

Az alábbiakban felsoroljuk a legfontosabb klasszifikátorcsaládokat, a teljesség igénye nélkül[1]:

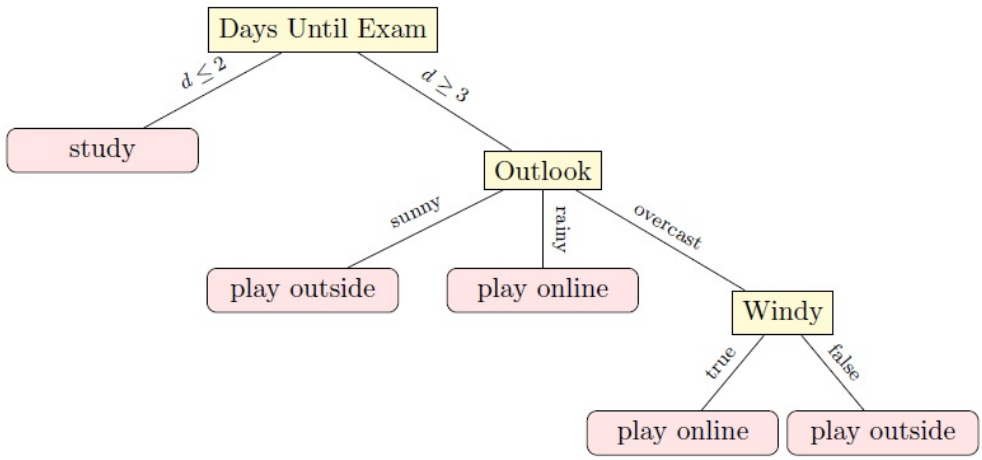
- Legközelebbi szomszéd módszerek
- Lineáris és logisztikus regresszió
- Neurális hálók
- Döntési szabályok, döntési fák
- Naiv-Bayes módszerek, Bayes-hálók
- Szupport vektor gépek
- Metaklasszifikátorok (ezek különböző klasszifikátorok eredményei alapján döntenek)

A modellek jellemzően legalább 2 lépcsőben dolgoznak. A rekordokat tanító és tesztelő részre osztják, a tanítók segítségével létrehoznak egy modellt, és a tesztrekordokkal ellenőrzik a modell minőségét.

A következőkben a dolgozatunk szempontjából fontos klasszifikátorokról ejtünk néhány szót.

2.3.1. Döntési fák

A döntési fák egyszerű – egy-egy attribútumon alapuló – döntések sorozatával osztályoznak. Minden döntési fának van egy gyökere. A gyökértől indulva minden csúcsban van egy feltétel egy attribútumra, a csúcs gyermekeibe vezető éleken pedig a lehetséges válaszok helyezkednek el. Tehát egy csúcsnak kategorikus attribútum esetén annyi gyereke van, ahány lehetséges válasz; többi esetben pedig, ahány intervallumra osztjuk a lehetséges értékeket. Ha mindig a pillanatnyi csúcsban található feltételnek megfelelően megyünk tovább, előbb-utóbb egy levélbe érünk. A levelekben megnézzük, hogy az adott levélbe érkező rekordok között melyik a leggyakoribb osztály; ez lesz a levél címkéje, a klasszifikáció kimenete. Sokféleképpen építhető döntési fa, a legismertebb módszerek az ID3 és a C4.5 (utóbbi előbbinek a továbbfejlesztett változata).



2. ábra. Egy döntési fa

2.3.2. Random forest

A Random forest[2] klasszifikátor egyszerre több kisebb méretű döntési fát is épít. Minden fa minden csúcsába k (konstans) kisorsolt változó közül valamilyen célfüggvény szerint legjobbat választja (az eredeti modell a Gini-index szerint dönt). A fák építését egy előre rögzített mélységig folytatja. Ha n db attribútumunk van célszerű $\log n$-t választani a fák mélységének, így a modellünk $O(n)$ változót használ majd összesen. Egy ismeretlen rekord osztályáról a fák többségi szavazással döntenek. Ez a Random forest eredeti modellje. Azóta több, a teljesítményt növelését célzó módszer is napvilágot látott: például a Gini-index lecserélése más, érzékenyebb mutatóra, vagy a fák szavazatainak súlyozása.[11]

2.4. A klasszifikátorok validálása

A klasszifikációs feladathoz hozzátartozik a klasszifikátor minőségének ellenőrzése, azaz validálása. Ilyenkor a modell számára ismeretlen, de számunkra ismert címkéjű adatpontokon végezzük a vizsgálatot.

A következőkben a klasszifikátorok validálásának lehetséges módjaiból mutatjuk be a legfontosabbakat. Nem tudjuk egy klasszifikátor általános minőségét mérni, hanem mindig csak egy adott tesztadatra szorítkozva beszélhetünk a modell teljesítményéről. Emiatt a tesztadatot előre rögzítettnek tekintjük.

2.4.1. Definíció (Pontosság). *A K klasszifikátor pontossága*

$$P(K) = \frac{\textit{a jó osztályba sorsolt rekordok száma}}{\textit{az összes rekord száma}}$$

A pontosság csak a helyesen klasszifikált rekordok arányát veszi figyelembe, az osztályok nagyságának arányait nem. Egy másik – szintén validálásra szolgáló – eszköz a keveredési mátrix. A keveredési mátrix ugyan jobban reprezentálja egy klasszifikátor teljesítményét, de önmagában még nem ad meg rendezést a klasszifikátorok között.

2.4.2. Definíció (Keveredési (Confusion) mátrix). *Egy n osztályos K klasszifikátor $K \in \Re^{n\times n}$ keveredési mátrixa az R rekordhalmazra nézve:*

$$Kev_{ij} = |\{r_k \in C_j | K(r_k) = i\}|, C_j \textit{ a } j\textit{. osztályba tartozó rekordok halmaza.}$$

Példa keveredési mátrixra:

		Valós osztály		
		a	b	c
Jósolt osztály	a	10	0	1
	b	2	5	0
	c	0	3	9

Ha kétosztályos klasszifikátorunk van az egyes mezőkre külön jelölést vezetünk be angol elnevezésük szerint:

		Valós osztály		
		1	0	$\sum$
Jósolt osztály	1	TP	FP	P
	0	FN	TN	N

Az egyes betűk jelentései: T = True; F = False; P = Positive; N = Negative.
A „true positive rate" a pozitív osztálybeliekre vett pontosság.

2.4.3. Definíció (True Positive Rate). $TPR = \dfrac{TP}{TP + FN}$

A „false positive rate" a negatív osztálybeliekre vett hiba (1-pontosság).

2.4.4. Definíció (False Positive Rate). $FPR = \dfrac{FP}{FP + TN}$

2.4.1. A ROC-görbe és a görbe alatti terület

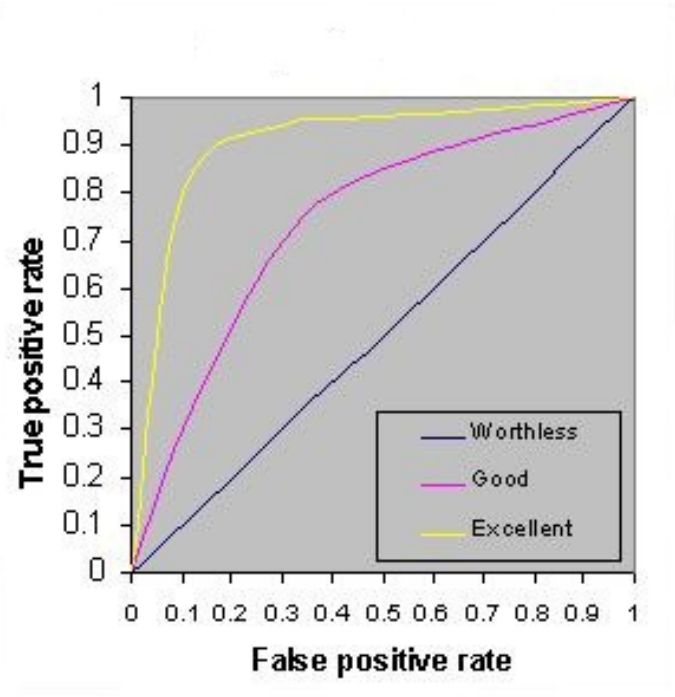
Legtöbb esetetben a K klasszifikátor nem csak egy osztályt ad vissza, hanem egy eloszlást is: a modell szerint mekkora valószínűséggel tartozik az adott rekord az egy egyes osztályokba. Mivel a valószínűségek összege 1 kell legyen, ezért bináris célváltozó esetén egy számmal is jellemezhető a feltételezett eloszlás: az „1" osztályba tartozás modell szerinti valószínűségével. Jelöljük $K^*(x)$-szel ezt a függvényt! A $K(x)$ így a következő alakú lesz:

$$K(x) = \begin{cases} 1 & \text{ha } K^*(x) \geqq t \\ 0 & \text{ha } K^*(x) < t \end{cases}, \text{ ahol } t \text{ konstans (a klasszifikátor választja ki).}$$

Ezt a t-t hívjuk thresholdnak. Ha elég nagynak választjuk, akkor minden egyedet 0-osztályúnak klasszifikálunk, így $FPR = 0, TPR = 0$ lesz. Ha viszont megfelelően kicsinek, akkor mind a TPR, mind az FPR érték 1 lesz. A $[0,1] \times [0,1]$ négyzetben ábrázolhatjuk ennek a két mutatónak az alakulását a t változtatásával, az FPR értéket az x, a TPR értéket az y tengelyen jelölve. Mivel mind az FPR, mind a TPR csak véges sok értéket vehet fel (a nevezők konstansok, míg a számlálók egészek, legalább 0 és legfeljebb n nagyságúak), ezért véges sok pontot kapunk. Ezeket t szerint sorban, egyenes szakaszokkal összekötve kapjuk a ROC (receiver operating characteristic) görbét. Ehhez a görbéhez nagyon hasonló görbét úgy is kaphatunk, hogy sorba rendezzük a rekordokat K^* értékeik szerint, és végigmegyünk a legnagyobbtól kezdve a legkisebbig. A görbét az origóból indítjuk, majd mindig ha egy pozitív osztályba tartozó rekord következik, akkor felfelé lépünk $1/P$-t, ha pedig negatív, akkor jobbra $1/N$-t. Minden lépés után összekötjük azt, ahol állunk a görbe aktuális végpontjával. Előfordulhat, hogy negatív és pozitív elemek is szerepelnek ugyanakkora jósolt értékkel. Mikor ilyenekhez érünk, akkor az összes szerint lelépjük a megfelelő lépést, és csak utána húzzuk be a következő vonalat. Mivel P-szer lépünk felfele $1/P$-t, és N-szer $1/N$-t, így az utolsó lépésben az $(1,1)$ pontba érkezünk.

A ROC-görbe alatti területet AUC-vel (area under curve) jelöljük. Ha egy klasszifikátor a k^* értékek szerint rendezet lista elejére helyezi az „1”-osztályba tartozó egyedeket, és végére a „0”-osztályba tartozókat, akkor lesz olyan t érték, amely mentén határvonalat húzva tökéletesen osztályozhatjuk a rekordokat. Az AUC érték ilyenkor 1 lesz, mivel először csak felfele, majd csak jobbra lépünk. Ha megfordítanánk a jóslatunkat, vagyis k^* helyett $1 - k^*$-ot vennénk, akkor a sorrend megfordulna. Ez pont a ROC-görbe a négyzet középpontja körüli 180 fokos elforgatásának felel meg; az új AUC érték 1 mínusz a régi érték lesz. Ebből következik, hogy ha egy klasszifikátor teljesítménye kevesebb, mint 0.5, akkor érdemesebb a megfordítását venni. Egy olyan klasszifikátornak, amely egyforma valószínűséggel osztályoz a két osztályba, az AUC értéke várhatóan $0,5$ lesz, mivel minden rangsornak az ellenkezőjét is egyforma valószínűséggel állítja elő, és minden ilyen pár átlag AUC értéke $0,5$. A 0.5 AUC érték tehát azt jelenti, hogy nem sikerült megtanulni az adatot. A ROC-görbe lehetőséget nyújt még más

vizsgálatokra is. Például, ha az érdekel bennünket, hogy a lista tetején mennyire jól teljesít egy klasszifikátor nézhetjük egy origóhoz közeli függőleges sávban a görbe alatti területet.



3. ábra. Tipikus Roc-görbék

2.4.2. A keresztvalidáció

A keresztvalidáció elfogadott, és gyakorta alkalmazott kiértékelési eljárás. A keresztvalidáció során az ismert címkéjű rekordokat k egyenlő részre osztjuk . Majd minden részt pontosan egyszer különvesszük, a maradék adaton tanítunk egy modellt, és a különvett részen teszteljük azt. A keresztvalidáció előnye, hogy minden rekord pontosan egyszer szerepel tesztrekordként (és olyankor nem szerepel a tanítóhalmazban).

3. Fontos jellemzők kiválasztása

3.1. Motiváció

A gépi tanulásos modellek sikerét sok minden befolyásolja, de leginkább az adat információtartalma. Gondolhatnánk, hogy több változó több információt hordoz, ezzel szemben a gyakorlat azt mutatja, hogy a tanulási módszerek könnyebben boldogulnak, ha kevesebb, de nagyobb predikciós értékű változóval kell dolgozni, mintha irreleváns vagy nagyon sok kis információt tartalmazó változó is található az adatban. Ez az egyik motiváló tényező az attribútumok szűrésére. A változók szelektálása továbbá segíthet elkerülni a túltanulást is, amikoris a modell a tanító halmaz egyéni sajátosságait tanulja meg az általános vonások helyett. Ha nincs olyan sok változó, egyszerűbbek lesznek a klasszifikációs szabályok és a modell sem tud annyira specifikálódni[8]. Más okok miatt is fontos, vagy akár elkerülhetetlen lehet a jellemzők számának csökkentése. Ilyen ok például, ha túl nagy költséggel (idő, pénz) jár egy-egy attribútum vizsgálata. Ekkor kevés adatponton végezzük el a széles spektrumú adatfelvételt, és ezek alapján kiválogatjuk azokat a változókat, amelyekkel már hatékonyan lehet klasszifikálni a többi rekorodot is.

A fontos jellemzők kiválasztása valójában két megközelítést foglal magában: egyrészt az attribútumok, másrészt az attribútumok részhalmazainak rangsorolását. Ennek alapján a kiválasztás problémájára adott módszerek két nagy csoportba oszthatók: a magukat az attribútumokat kiértékelő módszerek es az attribútumok részhalmazait kiértékelő módszerek csoportjaiba. A teljesség kedvéért megjegyezzük, hogy létezik egy harmadik csoport is: a beágyazott rendszerek. Ez utóbbiak olyan módszerek, amelyek egyéb gépi tanulási folyamatok során melléktermékként elvégzik a változók kiválasztását. Például megvizsgálhatjuk, milyen változókat használ egy döntési fa.

Előfordulhat, hogy két attribútum korrelál egymással és az osztálycímkével is. Ilyenkor mindkét jellemzőt fontosnak találhatjuk, de felesleges lenne mindkettőt kiválasztani, mivel a második hozzávételével már nem jutunk jelentősen több információhoz. Míg az attribútumokat önmagukban vizsgáló eljárások nem kínálnak megoldást erre a problémára, a részhalmaz-értékelő módszerek ezzel szem-

ben kezelni tudják a jellemzők egymáshoz való kapcsolatát, korrelációjukat vagy együttesen kifejtett hatásukat, következésképpen az ilyen módszerek bonyolultabbak és lassabbak.

3.2. Többszintű relevancia

Mint láttuk különböző szempontok vezérelhetik a tulajdonságok számának csökkentését, ezért is nincs olyan eljárás, ami az összes többit felülmúlná. A helyes módszer kiválasztasának megkönnyítéséhez az attribútumoknak 3 relevanciaszintjét vezetjük be[5]: irreleváns, gyengén releváns, és erősen releváns. Ha tudjuk, hogy milyen változókat keresünk, könnyebben kiválaszhatjuk a megfelelő eljárást is.

A következő definíciókban feltesszük, hogy $\mathcal{A}$ egy n elemű attribútumhalmaz, $A_i \in \mathcal{A}$, $S_i = \{A_1, A_2, \ldots, A_{i-1}, A_{i+1}, \ldots, A_n\}$, és C pedig az osztályváltozó.

3.2.1. Definíció (Erős relevancia). *Az A_i attribútum erősen releváns, amennyiben létezik olyan c,a_i, és s_i, hogy ha $p(A_i = a_i, S_i = s_i) > 0$, akkor*

$$p(C = c|A_i = a_i, S_i = s_i) \neq p(C = c|S_i = s_i).$$

Tehát egy változó erősen releváns, ha van olyan információ, amit csak az hordoz.

3.2.2. Definíció (Gyenge relevancia). *Az A_i attribútum gyengén releváns, ha nem erősen releváns és létezik olyan c,a_i, és $S'_i \subset S_i$, hogyha $p(A_i = a_i, S'_i = s'_i) > 0$, akkor*

$$p(C = c|A_i = a_i, S'_i = s'_i) \neq p(C = c|S'_i = s'_i).$$

Tehát egy változó gyengén releváns, ha hordoz információt, de azt más attribútum(ok) is hordozzá(k).

3.2.3. Definíció (Irrelevancia). *Egy A_i attribútum irreleváns, ha nem erősen és nem gyengén releváns.*

4. Attribútumrangsoroló módszerek

Minden attribútumrangsoroló módszernél egy, az attribútumokon értelmezett függvényt nézünk. Mivel az attribútumok valós eloszlását nem ismerjük, ezért mindenhol a tapasztalati gyakoriságokkal tudunk csak számolni. Feltesszük, hogy attribútumaink csak véges sok értéket vesznek fel (a folytonos változóknál intervallumokra osztással diszkretizálhatunk). A következő definíciókban az A attribútum értékkészletét jelölje $V = \{v_1, v_2, \ldots, v_k\}$, és $p(A = v_i) = p_i$!

4.1. Gini-index

4.1.1. Definíció (Gini-index). *Az A attribútum Gini-indexe:*

$$G(A) = 1 - \sum_{i=1}^{k} {p_i}^2.$$

A Gini-index az egyik legegyszerűbb attribútumértékelő módszer. A Gini-index annál nagyobb, minél jobban „szétosztja" egy attribútum a rekordokat. Mivel gyorsan számolható, gyakran alkalmazzák döntési fák építésekor. Folytonos változatát országok jövedelemeloszlásának vizsgálatára használják.

4.2. Entrópia

Az entrópia fogalmát az információelméletben Shannon vezette be 1948-ban, hogy jelsorozatok információtartalmát mérhetővé tegye.

4.2.1. Definíció (Entrópia (Shannon-féle)). *Az A attribútum entrópiája:*

$$H(A) = -\sum_{i=1}^{k} p_i \log_2 p_i.$$

Az entrópia egy attribútum „bizonytalanságát" fejezi ki. Ha egy attribútum csak egyféle értéket vesz fel, akkor nincs bizonytalanság, és a képlet szerint az entrópia 0.

4.2.2. Definíció (Feltételes entrópia). *Legyenek A és B attribútumok. Értékkészletük rendre V és W. Az A attribútum B feltétellel vett feltételes entrópiája:*

$$H(A|B) = -\sum_{v \in V} \sum_{w \in W} p(A = v, B = w) \log_2 p(A = v|B = w).$$

Bebizonyítható, hogy $H(A|B) = H(AB) - H(B)$, ami azt jelenti, hogy a feltételes entrópia megmondja, mennyi bizonytalanság marad A-ban akkor, ha ismerjük B értékét. Az entrópia és feltételes entrópia fogalma nem önmaguk, hanem a következő két mérték miatt fontos vizsgálatunk szempontjából.

4.3. Information gain és gain ratio

Legyen C a kitüntetett célattribútum!

4.3.1. Definíció (Information gain).

$$IG(A) = H(C) - H(C|A).$$

Az "information gain" azt mutatja meg, hogy mennyivel csökkentette a célváltozó bizonytalanságát az A változó megismerése.

4.3.2. Definíció (Gain ratio).

$$GR(A) = \frac{H(C) - H(C|A)}{H(A)}.$$

A "gain ratio" az "information gain" korrigálva az A változó entrópiájával. Ha A maga is egy nagy bizonytalanságú változó, akkor könnyen meghatározható vele az osztály, de nem biztos, hogy A egy fontos változó. Például, ha A egy ember azonosítószáma, és C jelzi egy betegség meglétét vagy hiányát, akkor azonosítószám alapján egyértelműen kiválaszthatók a beteg egyedek, bár az azonosítónak semmi köze a megbetegedésekhez. Ezért ha találunk egy olyan változó, ami nem túl bizonytalan, de ismerete mégis csökkenti a betegség bizonytalanságát, az nagyobb GR értékkel fog rendelkezni.

4.4. χ^2-próba

Egy A változó az osztályváltozóval való korrelációjánk eldöntésére végezhetünk χ^2 próbát a változók függetlenségére. Szokás szerint jelölje N_{ij} azon R beli rekordok számát, melyekre $A = v_i$ és $C = c_j$, valamint $N_{i\cdot} = \sum_j N_{ij}$, és $N_{\cdot j} = \sum_i N_{ij}$!
A próbastatisztika:

$$k(\sum_{i=1}^{k}\sum_{j=1}^{l}\frac{N_{ij}^2}{N_{i\cdot}N_{\cdot j}} - 1).$$

Minél nagyobb a statisztika értéke annál jobban elentmondanak az adatok a két változó függetlenségének.

4.5. A RELIEF algoritmus

A RELIEF algoritmus[6] volt az egyik első hatékony megoldás a változókiválasztás problémájára, és azóta is alkalmazzák referenciaként. Ez az algoritmus azért különleges, mert a rekordok felől közelíti meg a rangsorolást. A kimenete egy W súlyvektor, mely az egyes attribútumok relevanciáját tartalmazza.

4.5.1. Az algoritmus működése

A rekordok halmazát jelöljük R-rel, $r \in R$ i-ik attribútumát pedig r_i-vel! Feltesszük, hogy ismerjük az egyes rekordok osztálycímkéjét, ami csak kétféle lehet. Definiálunk egy d távolságfüggvényt R elemein, külön-külön az attribútumok értékkészletén értelmezett d_i távolságok segítségével. Számértékű attribútumok esetén értelemszerűen definiáljuk a távolságot; kategorikus attribútumok esetén pedig 0 a távolság, ha az adott attribútuma a két adatpontnak megegyezik, különben 1. Tehát két rekord távolsága legyen $d(r,s) = \sqrt{\sum_{i=1}^{n} d_i^2(r_i, s_i)}$.

1. Kezdetben minden attribútum súlya legyen $W = 0$!
2. Válasszunk R-ből véletlenszerűen egy rekordot: ez lesz r.
3. Legyen R_m az r-rel megegyező osztályúak r kivételével, R_e pedig ellentétes osztályúak!

4. Nézzük az r-hez legközelebbi rekordot R_m-ből és R_e-ből is; ha több legközelebbi van, akkor véletlenszerűen az egyiket! Legyenek ezek r_m és r_e!

5. Frissítsük W koordinátáit a következő képlet szerint:

$$W_i := W_i - d_i^2(r_i, r_{m_i}) + d_i^2(r_i, r_{e_i})$$

6. Egy előre meghatározott lépésszámig ismételjük a $2-5$. lépéseket!

A RELIEF algoritmus hátránya, hogy csak bináris osztállyal működik (létezik egy továbbfejlesztés, mely már többosztályos rekordokat is kezel); valamint, hogy főként akkor működik jól, ha a releváns változók kevesen vannak és erősen relevánsak többi változónál, mivel a RELIEF a gyengén releváns változókat találja meg. A módszer előnye gyors, ha t-szer iteráljuk, akkor $O(|R|t)$ lépésig tart a keresés, és általában hatékonyan találja meg a gyengén releváns vűltozókat.

5. Részhalmaz rangsoroló módszerek

5.1. A FOCUS-algoritmus

A FOCUS algoritmus azt a minimális elemszámú attribútumhalmazt keresi, mellyel legjobban magyarázható az osztálycímke. Magyarázhatóság alatt itt azt értjük, hogy az adott attribútumokra leszűkített adattábla nem tartalmaz inkozisztens sorokat, rekordokat.

5.1.1. Definíció (Inkonzisztencia). *$\omega_i, \omega_j \in \Omega$ $\mathcal{A}$ attribútumhalmazhoz tartozó rekordok inkozisztensek, ha $\forall A_k \in \mathcal{A}$-ra $A_k(\omega_i) = A_k(\omega_j)$, de $C(\omega_i) \neq C(\omega_j)$, ahol C az osztályváltozó.*

A FOCUS a már látott RELIEF algoritmussal ellentétben csak a szükséges számú változót találja meg. Azonban ha az adatunk inkozisztens, akkor a FOCUS algoritmusnak nincs visszatérési értéke

A részhalmaz rangsoroló módszerek (egyes kivételektől eltekintve) a változók részhalmazainak terében végzik a keresést, ezért az alábbi 4 paraméter már meg is határozza az algoritmust:

- **Kezdeti részhalmaz:** Az első részhalmaz, amit megvizsgálunk. Általában vagy az üreshalmaz, vagy az összes attribútum halmaza.

- **Vizsgálat sorrendje:** n változó esetén a részhalmazok száma 2^n, így túl sokáig tartana az összes lehetséges részhalmazt végignézni, valamilyen módon szűkíteni kell a vizsgálandó részhalmazok körét. A vizsgálat sorrendje az eddig megvizsgált halmazok eredményei alapján meghatározza a következő megvizsgálandó részhalmazt.

- **Részhalmazok kiértékelésének módja:** Két nagy csoportjuk létezik: a filterek (szűrők) és a wrapperek (csomagolók). A filteres megközelítésnél a filter – az értékelő függvény – fix, független minden klasszifikátortól. A wrapper-módszerben egy klasszifikátort használunk a kiértékelésnél, azt amelynek teljesítményét növelni szeretnénk.

- **Megállási feltétel:** Általában az optimális részhalmaz elemszámához, vagy egy adott teljesítmény eléréséhez, vagy egy adott iterációszámhoz kötjük.

5.2. Bejárási stratégiák

Az alábbiakban bemutató bejárási stratégiák már magukban foglalják, a kezdőpontot vizsgálat sorrendjét és a megállási feltételt is.

- **Best first** Indulhat az üres vagy a teljes halmazból is, minden lépésben sorra veszi az összes változót. A legelső aminek hozzávétele/elhagyása javít azt hozzáveszi/elhagyja. Akkor állunk meg, ha már nem tudunk növelő irányba lépni.

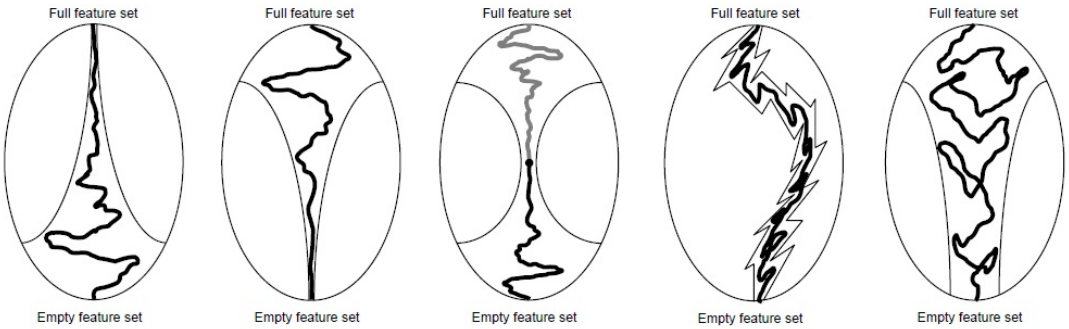
- **Szekvenciális előrefele keresés(SFS)** Kiinduló halmazunk az $\mathcal{A}_0 = \mathcal{A}$, s az $\mathcal{A}_{i+1}$ pedig a maximális jóságú halmaz lesz, ami úgy kapható, hogy elhagyjuk A_i egy elemét. Az üres halmaznál állunk meg.

- **Szekvvensiális visszafele keresés(SBS)** Kiinduló halmazunk $\mathcal{A}_0 = \emptyset$, s $\mathcal{A}_{i+1}$ pedig az a maximális halmaz, azok közül, mely úgy kapható, hogy A_i-hez hozzáveszünk egy elemet $A \setminus A_i$-ból. Ha $A_i = A$ akkor megállunk.

- **Kétirányú keresés (BDS)** Véletlenszerűen választunk egy $\lfloor n/2 \rfloor$ elemű részhalmazt, melyből egyszerre indítunk egy SFS, és egy SBS keresést.

- **l-t előre r-t vissza keresés (LRS)** Ha $l > k$, akkor az üreshalmazból, ha $l < r$ akkor $\mathcal{A}$-ból indulunk. Felvátva hajtunk végre k darab SFS lépést, valamint l darab lépését az SBS algoritmusnak.

- **Szekvenciális lebegő keresések (SFFS és SFBS)**[10] Csak az SFFS bejárást mutatjuk be, a SFBS ennek megfordítása. Induljunk ki az üres halmazból! Minden lépés során az aktuális részhalmazhoz vegyük hozzá azt az attribútumot, amelyikkel legjobban váltitatja meg a célfüggvény értékét, majd ha van olyan változó, amelynek elhagyásával növekszik a célfüggvény, akkor hagyjuk el ilyenek közül amelyikkel a célfüggvény legjobban nő. Ha olyan részhalmazba érkezünk, ahol már jártunk, akkor bővítéssel folytatjuk, hogy a végtelen hurkokat elkerüljük. Akkor állunk le, ha eljutottunk a teljes halmazhoz.



4. ábra. Bejárási stratégiák ábrái (balról jobbra: SFS, SBS, BDS, LRS, SFFS)

5.3. Különbségek a filter- és a wrapper-módszer között

A filter-módszerek nem függnek semmilyen klasszifikátortól, így csak általánosságban tudnak dönteni egy attribútum fontosságáról. Ha egy konkrét M modell hatékonyságának növelése a cél, akkor érdemesebb egy M-hez kötött mértéket használni. Ez az úgynevezett wrapper-módszer[7]. A wrapper-módszer esetében egy részhalmaz kiértékelése az adott attribútumhalmazra leszűkített tanítórekordokra épített klasszifikátor teljesítménye lesz a relevancia-függvényünk. (A szakirodalomban wrapper-módszereknél teljesítmény alatt többnyire a pontosságot értik, mi azonban végzünk AUC-vel is méréseket.)

A wrapper-módszer általában sokkal jobban teljesít, mint a filter-módszerek, de számítási igénye sokkal nagyobb, hiszen minden részhalmaz kiértékelésekor, újabb klasszifikátort kell építeni. Ehhez jön még hozzá, hogy a modell korrekt kiértékelésekor ha keresztvalidációt alkalmazunk, az megsokszorozza a modellépítések számát. A wrapper-módszer további hátránya, hogy a kiválasztott részhalmaz csak az M modell esetén növeli a hatékonyságot, arra nincs garancia, hogy más modellekre is pozitív hatással lesz az attribútumhalmaz csökkentése.

6. Permutációs módszer az attribútumok kiválsztására

A wrapper-módszer alkalmazánál a szűk keresztmetszet legtöbbször a modellépítések száma; egy döntési modell megalkotása általában időigényesebb, mint az adatok végigolvasása vagy egy modell kiértékelése. Ezért az algoritmusaink során próbáljuk minimalizálni a modellépítések számát, vagy megpróbálhatjuk helyttesíteni a kevésbé időigényes lépésekkel a modellépítést, akkor is ha így csak közelítőleg jó eredményt kapunk. Leo Breiman a Random forestet bemutató cikkben[2] megemlít egy módszert, amellyel egy attribútum fontosságát szeretné becsülni az adott modellen belül.

A leírt módszer szerint, az A_i attribútum fontosságának becsléséhez megpermutáljuk a tesztrekordok között A_i értékeit,majd megvizsgáljuk a modell teljesítményének változását. A permutálás segítségével az attribútumot helyettesítettük

egy ugyanolyan eloszlású zajjal. Az információ csökkenésének következtében azt várjuk, hogy romlik a modell teljesítménye, főként mivel a model építésekor a változót még nem kevertük meg, így a modell is úgy kezelte, mint információ hordozót. Hipotézisünk szerint ha egy változó fontos a célváltozó meghatározásához, akkor az ő értékeinek megpermutálása nagyobb mértékben rontja a modell teljesítményét, mint egy olyan változó, mely kisebb súllyal járul hozzá.

Méréseink során ezzel a keveréses technikával gyorsítottuk a wrapper-módszer azon lépéseit, amikor egy változót kívánunk elhagyni. Azt a változót hagytuk el, melynek megkeverésével a legkevésbé romlott a teljesítmény. Így ha k elem közül kell választani, akkor k modellépítés helyett, csak egy modellt kell építenünk. Egy attribútum értékeinek különböző megpermutálásával természetesen különböző eredményeket is kaphatunk, így egy változót nemcsak egyszer, hanem többször is megkevertünk, így egy többelemű minta alapján adhattunk becslést a relevanciára. Mint torzítatlan becslést, a mintaelemek átlagával becsültük a várható értéket.

7. Mérések leírása és eredmények

7.1. A keretrendszer

A kísérletek elvégzéséhez a Waikatoi Egyetem (Új-Zeland) által fejlesztett 'Weka'[9] adatbányászati szoftver biztosította a keretrendszert. A Weka egy nyílt forráskódú Java nyelven íródott program, amely rendelkezik mind klasszifikációs, mind változókiválasztási módszerekkel, köztük a filter- és wrapper-módszerekkel is. Így már csak a gráfbejárást megvalósító osztályt, és az adott atrribútumot összekeverő algorimust, valamint a kiértékelő osztályokban alkalmazandó új függvényeket implementáltam. A méréseket egy korszerű Intel Xeon szerverprocesszoron futtatuk, de csak egy programszálon.

Módszerünk bemutatásakor érdemes külön figyelmet szentelni az attribútumok megkeverését elvégző algoritmusra, mivel nem triviális egyenletes valószínűséggel választani a k-elemű permutációk csoportjából. Már 70! is több mint 100 számjegyből áll, azaz ábrázolásához több, mint 300 bit kéne, amit a legtöbb

számítógép nem támogat. Feltételezve, hogy tudunk $j \leq k$-ra egyenletes valószínűséggel egészeket kisorsolni a $[0, j]$ intervallumból; k elem megpermutálására a következő eljárást választottuk:

1. Vegyünk egy $k+1$ hosszú tömböt, ennek az utolsó k darab helyére írjuk be az értékeket sorrendben!
2. Az üres helytől jobbra levők közül egyforma valószínűséggel sorsoljunk ki egyet!
3. A kisorsolt elemet helyezzük át az üres helyre!
4. A régi üres hely és az új üres hely közt lévő elemeket toljuk jobbra eggyel!
5. 2-4. pontokat még $k-1$-szer megismételjük!

7.1. Állítás. *Bármely két permutáció egyforma eséllyel fordul elő.*

Bizonyítás. Mivel minden lépés során egyenletes eloszlás szerint sorsoltunk a változók közül, így nem tettünk különbséget köztük. Tehát az x. helyre ugyanakorra eséllyel kerülhet az összes változó. Tehát minden permutáció egyforma eséllyel fordul elő.

Felhívjuk a figyelmet arra, hogy ez az algoritmus helyben elvégzi a keverést, nem szükséges a tömb elemeinek másolása. Valamint a lépésszáma is $O(n)$, feltéve, hogy véletlen számot konstans időben tudunk sorsolni.

7.2. A paraméterek

- Bejárási stratégia: Olyan módszert kerestünk, amelyben gyakran van szükség változók eliminálására. Emiatt és egyszerűsége miatt a felülről induló SBS-t választottuk.
- A klasszifikátor: Mivel az eredeti ötlet is a Random forestet bemutató cikkből jött, mi is ezt választottuk kiértékelő módszerünknek. További érvek még a Random forest mellett, hogy minden változót használ (a nagy modellben minden változó nagy valószínűséggel szerepel), jó teljesítményű, így

releváns méréseket folytathattunk, valamint a modell építésének ideje jelentősen hosszabb, mint a kiértékelés. A klasszifikátort úgy paramétereztük, hogy 50 $\log n$ mélységű fát építsen.

- A teljesítmény mérése: Mivel wrapper-módszerről van szó, a klasszifikátor célfüggvényét kellett kiválasztanunk. Két mértéket használtunk: a pontosságot, valamint a ROC-görbe alatti területet (AUC). Előbbit trivialitása miatt, utóbbit pedig érzékenysége miatt. 30-szor kevertünk meg egy változót, és utána a mintaelemek átlagát vettük, mivel feltettük, hogy mind két érték normális eloszláshoz tart, s a várható értékre torzítatlan becslés lesz a mintaátlag.

7.3. Adatok

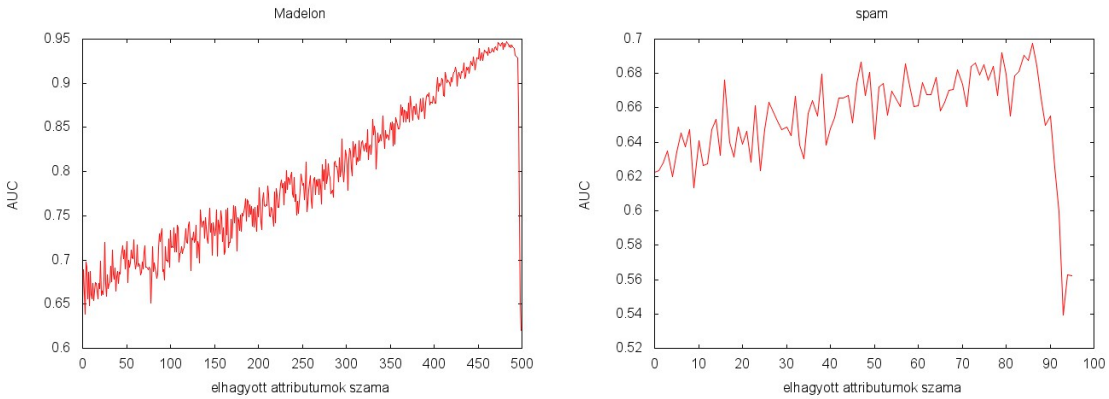
Az alábbi adatokon végeztük a tesztelést:

- Madelon: A madelon adatbázis az UCI[3] nyilvános adattárhátrházban megtalálható, szintetikus, generált adat, kifejezetten attribútumkiválasztási módszerek tesztelésére. Egy 5-dimenziós hiperkoca csúcsaiban találhatók a rekordok (kicsit szétszórva, hogy ne egyformák legynek). Mindegyik csúcsban véletlenszerűen döntöttek az osztályról, ami kétféle lehet; az ugyanzon csúcshoz tartozó rekordok osztálya megegyezik. Ezután generáltak 20 változót az 5 koordináta különös lineáris kombinációjaként. Végül 480 független változót is hozzáadtak. Így összesen 500 változó van, valamint 1000-1000 rekord osztályonként.

- Spam: Az ECML/PKDD 2010 Discovery Challenge[12] versenyen weboldalak klasszifikálása volt a feladat. Több címkét is megadtak az egyes weboldalakhoz, ezek közül 10-re kellett modellt építeni (spam, hír, hirdetés, adatbázis, szórakozás...). Több jellemzőcsoportot is megadtak, amelyek közül mi a tartalommal kapcsolatosakat használtuk.

7.4. Eredmények

Az alábbi grafikonokon az AUC értékek változása látható az elhagyott változók számának növekedésével. Baloldalt a madelonra, jobboldalt a spam adat „commercial" célváltozóra futtatott mérések eredményei láthatók.



Láthatjuk, hogy mindkét esetben javul a klasszifikátor minősége, a madelon esetében igen jelentősen, és az optimális részalmazok is kis méretűek. Valamit azt vehetjük észre, hogy a maximum után meredeken leesik a függvényérték. Ez azt mutatja, hogy az optimális halmazaink csak erősen releváns változókból állnak. Az időbeni gyorsulás mérésére referenciaként minden olyan részhalmazra, melyre keveréssel becslést adtunk modellt építettünk és kiértékeltünk, s így becsültük az SBS algoritmus idejét. Azt tapasztaltuk, hogy a permutációs módszer alkalmazásával majdnem felére csökkentek a futási idők. Megjegyeznénk, hogy míg a sima wrapper-módszer esetében a modellépítés az időigényes folyamat, a keveréses becslésünknél a többszörös mintavételezés miatt a kiértékelés lesz a szűk keresztetszet. Ha megkétszerezzük a keverések számát, akkor a futási idő is közel duplájára nő.

7.5. További lehetőségek

Nem vizsgáltuk meg a módszer számos lehetséges variációját. A bejárások, és maga a keveréses elhagyás is igen jól párhuzamosítható folyamatok. Másrészt a permutálást, nem csak az SBS bejárás során lehet alkalmazni, hanem minden

olyan algoritmusban, ahol változót hagyunk el, választhatjuk ezt a lehetőséget az elhagyandó változó kiválasztására. Ha tudjuk, hogy egymás után több attribútumot hagyunk el, akkor akár több attribútumot is megkeverhetünk egszerre, anélkül, hogy közben új modellt építenenénk.

8. Összefoglalás

Dolgozatunkban bemutattuk az adatbányászat és a gépi tanulás egyik legfontosabb problémáját a klasszifikácó kérdését. Majd a klasszifikáció egyik részfeladatára, a jelemzőkiválsztására adott eddigi megoldásokat csoportosítottuk és mutattuk be a legfontosabbakat. Végül egy attribútumok rangsorolására adott eljárást fejlesztettünk tovább, és megvizsgáltuk annak teljesítményét. Módszerünk jelentősen növelte a választott klasszifikátor teljesítményét, míg időigénye jóval kisebb, mint a hasonló módszereké.

Hivatkozások

[1] Bodon, Ferenc, Dr.(2010): *Adatbányászati algoritmusok*. URL: http://www.cs.bme.hu/ bodon/magyar/adatbanyaszat/tanulmany/ /adatbanyaszat.pdf. [Utolsó elérés dátuma: 2011.06.08.]

[2] Breiman, Leo – Schapire, E.(2001): Random forests. *Machine Learning*, 45. évf., 1. sz., 5–32. o.

[3] Frank, A. – Asuncion, A.(2010): *UCI Machine Learning Repository*. CA: University of California, School of Information and Computer Science, URL: http://archive.ics.uci.edu/ml [Utolsó elérés dátuma: 2011.06.08.]

[4] Han, Jiawei – Kamber, Micheline(2000): *Data Mining: Concepts and Techniques*, Morgan Kaufmann, San Francisco CA

[5] John, George H. – Ron, Kohavi – Pfleger, Karl(1994): Irrelevant Features and the Subset Selection Problem. *Machine Learning: Proceedings of the Eleventh International Conference*. Morgan Kaufmann, 121–129. o.

[6] Kira, Kenji – Rendell, Larry A.(1992): A practical approach to feature selection. *Machine Learning: Proceedings of the Ninth international Conference of Machine Learning*. Morgan Kaufmann, 249–256. o.

[7] Kohavi, Ron – John, George H.(1997): Wrappers for Feature Subset Selection. *Artificial Intelligence*, 97. évf., 1–2. sz., 273–324. o.

[8] Hall, Mark A.(1999): *Correlation-based Feature Selection for Machine Learning*, University of Waikato

[9] Hall, Mark A. – Frank, Eibe – Holmes, Geoffrey – Pfahringer, Bernhard – Reutemann, Peter – Witten, Ian H.(1999): The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11. évf., 1.sz.

[10] Pudil, P. – Novovicová, J. – Kittler J.(1994): Floating search methods in feature selection. *Pattern Recognition Letters*, 15. évf., 1119–1125. o.

[11] Robnik-Sikonja, Marko(2004): Improving Random Forests.*Machine Learning: ECML 2004*. Springer, 359–370. o.

[12] *ECML/PKDD 2010 Discovery Challenge versenykiírás* (2010). URL: http://datamining.sztaki.hu/?q=en/DiscoveryChallenge/rules, [Utolsó elérés dátuma: 2011.06.08.]