VÁLTOZÓSZELEKCIÓ ÁLTALÁNOSÍTOTT ADDITÍV MODELLBEN METAHEURISZTIKA SEGÍTSÉGÉVEL[1]

KOVÁCS LÁSZLÓ
Budapesti Corvinus Egyetem

A tanulmányban egy hibrid genetikus – harmóniakereső metaheurisztikus algoritmus alkalmazását vizsgáljuk meg általánosított additív modellek változószelekciós feladatára. Ismertetjük az additív modellek legfontosabb alapfogalmait és segítségükkel megadjuk a változószelekció feladatát. A feladatba felveszünk egy olyan korlátot, ami a magyarázó változók közti redundancia elkerülését biztosítja. A korlátozott változószelekciós feladatot két valós adatbázison oldjuk meg. Az eredményeink alapján a metaheurisztikus megoldásunk hatékonyabban tudja kezelni a korlátot, mint a szakirodalom által javasolt két korszerű algoritmus. Az algoritmus hatékonyságát jelentősen befolyásolja a véletlen és szabályszerű szelekciós operátorok közti optimális kombináció megtalálása. Nagyobb adatbázison az algoritmus futásideje jelentős, ám párhuzamosítással egy véletlenerdő-alapú megoldással összemérhető.

Kulcsszavak: általánosított additív modell, változószelekció, spline függvények, genetikus algoritmus, harmóniakereső algoritmus.
Jel kódok: C31, C52, C63, C65

1 Bevezetés

A felügyelt gépi tanulás során célunk, hogy egy jól definiált eredményváltozóra minél nagyobb pontosságú becslést adjunk bizonyos magyarázó változók értékének ismeretében. Napjainkban a feladat számtalan összetett algoritmus segítségével megoldható, pl. mélytanuló neurális hálózatok, véletlen erdők, támaszvektor-gépek stb. Azonban egyre több szerző, pl. Molnar [2020] és Du et al. [2019] hívja fel a figyelmet arra, hogy a legpontosabb becslést szolgáltató modellekben a használt magyarázó változók hatásai az eredményváltozóra nehezen, vagy egyáltalán nem visszafejthetők. Viszont bizonyos gyakorlati szituációkban a gépi tanulás legfontosabb eredménye nem feltétlenül a minél pontosabb becslés elkészítése, hanem az egyes magyarázó változók hatásának megállapítása. Például egy banknak egyértelműen meg kell indokolnia, hogy mi alapján utasít el egy hitelkérelmet. Ilyen esetekben nem előrejelző, hanem magyarázó modellek építése az elemző célja.

Napjaink ,,big data" környezetében, amikor egy adott becslési feladathoz rengeteg potenciális magyarázó változó könnyen az elemző rendelkezésére áll,

[1] A szerző a Budapesti Corvinus Egyetem Gazdaságinformatika Doktori Iskola PhD hallgatója. Jelen kutatás az Új Nemzeti Kiválóság Program ÚNKP-19-3 – I. doktori hallgatói kutatói ösztöndíjának segítségével valósult meg. Beérkezett 2020. augusztus 18. E-mail: laszlo.kovacs2@uni-corvinus.hu.

még egy egyszerű lineáris regressziós modell alkalmazása esetén is problémás lehet a magyarázó változók hatásainak megállapítása. Molnar [2020] és James et al. [2013] egyik javaslata a probléma áthidalására és a különböző felügyelt tanulási modellek értelmezhetővé tételére a változószelekció.

Hall [1999] szerint a változószelekció legfontosabb alapelve, hogy a kiválasztott magyarázó változók szorosan korreláljanak a becsülendő eredményváltozóval, de egymáshoz képest legyenek függetlenek. Lineáris esetben az elv a káros multikollinearitás elkerülését jelenti. Ez az elv gyakorlatilag megegyezik az ökonometriában gyakran alkalmazott parszimónia elvével (Wooldridge [2016]). Hall [1999] javasol is egy algoritmust alapvetően a klasszikus lineáris korrelációs együtthatóra támaszkodva (Correlation based Feature Selection, CFS) a változószelekció feladatának megoldására. Az algoritmus gyakorlatilag egy legjobb részhalmaz elvű változószelekció, ahol a célfüggvény azokat a magyarázó változókat preferálja, amik szorosan korrelálnak az eredményváltozóval, de más magyarázó változókkal páronként nem korrelálnak káros mértékben.

A CFS algoritmus elvét nem-lineáris esetekre kiterjesztő megoldásokat dolgozott ki Song et al. [2012] és Climente-González et al. [2019] is. Mindkét tanulmány javaslata azonban továbbra is csak a magyarázó változók páronkénti függetlenségét ellenőrzi a változószelekció során. Viszont, a magyarázó változók függetlenségét az is sértheti, ha egy változó kifejezhető a többi változó többváltozós függvényével.

Korábbi munkáinkban (Láng et al. [2017] és Kovács [2019]) egy hibrid genetikus-harmóniakereső algoritmust (továbbiakban Hibrid algoritmus) javaslunk a változószelekciós feladat megoldására lineáris modellekben. Az algoritmus a magyarázó változók VIF értékén keresztül a szelekciós folyamat során nem csak a változók közti páronkénti káros korrelációkra szűr. Az idézett két tanulmányban megmutatjuk, hogy a Hibrid algoritmus segítségével olyan regressziós modellek építhetők, amelyek becslési pontosságban nem maradnak el jelentősen az egyéb algoritmusok segítségével épített modellektől, ám azokhoz képest lényegesen kevesebb magyarázó változót használnak ennek eléréséhez. Az ilyen „extrém módon" takarékos modellek természetesen magukban hordozzák a kihagyott változók miatti torzítás veszélyét, de segíthetnek az elemzőnek azonosítani az eredményváltozót alakító legfontosabb független hatásokat. A módszer előnye a hagyományos dimenziócsökkentési eljárások alkalmazásával szemben, hogy a végső modellben konkrétan megnevezhető változók szerepelnek, adott esetben nehezen értelmezhető faktorok helyett.

Jelen tanulmányban kiterjesztjük a Hibrid algoritmust a nem-lineáris modellek körében végzett változószelekcióra is. Ehhez az általánosított additív modellek (továbbiakban GAM, a Generalized Additive Model angol kifejezésből) keretrendszerét alkalmazzuk, mivel James et al. [2013] szerint ezek a modellek egyensúlyt képviselnek a modellek értelmezhetősége és a becslési pontosság között nem-lineáris esetben is. GAM-ok esetében magyarázó változók marginális hatásai az eredményváltozóra meghatározhatók (ellenben a mélytanuló neurális hálózatokkal és ensemble modellekkel), de nem köti az elemzőt

egy előre definiált lineáris, logaritmikus, négyzetes vagy egyéb függvényforma (mint a klasszikus lineáris regresszióban).

A tanulmány második fejezetében áttekintjük a GAM modellek alapvető matematikai keretét. Felhívjuk a figyelmet arra, hogy a nem-lineáris modellezés hogyan teszi bonyolultabbá a változószelekciós feladatot a lineáris esethez képest. Bemutatjuk a concurvity jelenséget, ami a multikollinearitás fogalmának általánosítása nem-lineáris modellekre. A harmadik fejezetben ismertetjük a CFS algoritmus általánosításait nem-lineáris esetre. Bemutatjuk, hogy a Song et al. [2012] által javasolt mRMR és a Climente-González et al. [2019] által javasolt HSIC-Lasso algoritmusok miért képesek csak korlátoltan kezelni a concurvity jelenséget a változószelekció során. A negyedik fejezetben a Hibrid algoritmus működésének ismertetése következik GAM keretben. Megmutatjuk, hogy az algoritmus kiterjesztésével sikeresen kezelni tudjuk a változószelekció során a concurvity jelenséget anélkül, hogy az algoritmus bináris egyedreprezentációját módosítani kellene.

Az ötödik fejezetben a vizsgált algoritmusok működését összehasonlítjuk két valós adatbázison. Az első példában a feladat betongerendák nyomószilárdságának becslése. Az algoritmus paramétereinek finomhangolására egy kisebb méretű adatbázist használunk. A másik példában, ahol a feladat hitelkártyaügyfelek csődvalószínűségének becslése, az algoritmus teljesítményét egy nagyobb adatbázison is vizsgáljuk. A vizsgált változószelekciós algoritmusokon túl döntési fa és véletlen erdő algoritmusokat alkalmazunk benchmarknak. Összességében elmondhatjuk, hogy a Hibrid algoritmus végső modelljeiből jobban azonosíthatók a vizsgált eredményváltozóra ható, egymással nem szignifikánsan összefüggő magyarázó változók, mint a többi vizsgált algoritmus esetében. Az algoritmus futásideje viszont párhuzamosítás után is nagyságrendekkel nagyobb, mint a benchmarkként használt modelleké. Végezetül, a hatodik fejezetben összefoglaljuk a kutatás fő eredményeit. Alaposabban kitérünk a Hibrid algoritmus korlátaira, valamint ismertetjük a lehetséges továbbfejlesztési irányokat.

2 Az általánosított additív modellek (GAM) alapfogalmai

Adott egy n elemű minta. Legyen $Y = [y_1, y_2, \ldots, y_n]^T \in \mathbb{R}^n$ egy exponenciális eloszláscsaládból származó valószínűségi változó megfigyelt értékeit tartalmazó vektor. Ekkor egy GAM segítségével Y várható értéke (1) módon becsülhető p db $X_j = [x_{j1}, x_{j2}, \ldots, x_{jn}]^T$ magyarázó változó megfigyelt értékeinek segítségével (Hastie – Tibshirani [1990]).

$$h(E(Y)) = \varepsilon + \sum_{j=1}^{p} f_j(X_j)\,, \tag{1}$$

ahol $h(\cdot)$ a GAM link függvénye, $\varepsilon = [\varepsilon_1,\ \varepsilon_2, \ldots, \varepsilon_n]^T$ a modell hibavektora, és $f_j(\cdot)$ a j-edik magyarázó változó transzformációs függvénye.

A GAM-ok formális definíciója után részletesen ismertetjük az f_j reprezentációját, és megadjuk a változószelekciós feladatot.

2.1 Az f_j transzformációs függvények reprezentációja

Legyenek az f_j függvények bázis-spline, röviden b-spline függvények. Hastie – Tibshirani [1990] és Hazewinkel [2001] alapján a b-spline függvények több polinomot (bázisfüggvényt) szakaszosan illesztenek össze a spline függvény értelmezési tartományán, vagyis a $[\min(X_j), \max(X_j)]$ intervallumon. Egy b-spline függvény legfontosabb paramétere a rendje. Egy r-ed rendű b-splinenak mindig r szakasza van, a B_i bázisfüggvények $(r-1)$-edfokú polinomok. Ezt a tulajdonságot a b-spline függvények Cox de Boor rekurzív formulával történő megadása biztosítja:

$$B_{i,d}(x) = \frac{x - k_i}{k_{i+d} - k_i} B_{i,d-1}(x) + \frac{k_{i+d+1} - x}{k_{i+d+1} - k_{i+1}} B_{i+1,d-1}(x)$$

és

$$B_{i,0}(x) = \begin{cases} 1, & \text{ha } x \in [k_i, k_{i+1}[\\ 0 & \text{egyébként.} \end{cases}$$

A Cox de Boor rekurzióval adott $B_{i,d}(x)$ függvények lineáris kombinációjával r-ed rendű b-spline függvényt állíthatunk elő, ha a rekurziós formulát $d = r$-re alkalmazzuk: $f(x) = S_r(x) = \sum_i \alpha_i B_{i,r}(x)$, ahol i a megfelelő szakaszhatár (k_i) indexe x értelemzési tartományán.

A b-spline függvényekkel felépített GAM ezzel gyakorlatilag lineáris modellként funkcionál, $\alpha = [\alpha_1^1, \alpha_2^1, \ldots, \alpha_{r_1}^1, \ldots, \alpha_{r_p}^p]$ a modell együtthatóinak vektora, ahol α_s^j a j-edik magyarázó változóra illesztett r_j rendű b-spline függvény (S_{r_j}) együtthatója a változó értelmezési tartományának s-edik szakaszán. Ha bevezetjük a $\mathbf{B}_j$ mátrixokat, amelynek oszlopai a $B_{s,r_j}(X_j)$ vektorok $\forall j$-re (ezzel $\mathbf{B}_j$-k $n \times r_j$ méretűek lesznek), és az $\alpha_j = [\alpha_1^j, \alpha_2^j, \ldots, \alpha_{r_j}^j]$ vektorokat, akkor a GAM modell (2) alakot veszi fel,

$$h(E(Y)) = \varepsilon + \sum_{j=1}^{p} \sum_{s=1}^{r_j} \alpha_s^j B_{s,r_j}(X_j) = \varepsilon + \sum_{j=1}^{p} \mathbf{B}_j \alpha_j \,. \tag{2}$$

A (2) alakú GAM modellben nem csak az α vektor becslése szükséges, hanem az egyes magyarázó változókhoz tartozó r_j rendek és a spline-ok szakaszhatárainak megválasztása is. A szakaszhatárokat általában egyenletesen osztják el X_j értelmezési tartományán, ám ez sokszor nem az optimális megoldás (Hastie – Tibshirani [1990], Zhou – Shen [2001], Schumaker [2015]). Viszont, ha a modellben szereplő f_j függvényeket thin plate spline-ként és nem klaszszikus b-spline-ként becsüljük meg Wood [2003], Wahba [1990] és Green – Silverman [1994] alapján, akkor nem kell foglalkozni a szakaszhatárok optimális elhelyezésével, és a spline függvények rendjének megválasztása is lényegesen egyszerűbbé válik.

Thin plate spline keretben először a (2) egyenletben annyi módosítást végzünk, hogy $r_j = n$-t választunk $\forall j$-re (ezzel $\mathbf{B}_j$-k $n \times n$ méretűek lesznek,

α_j-k pedig n elemű vektorok). Ezek után a (3) alakban adott minimalizálási feladatot írjuk fel.

$$\min_{\alpha} -\sum_{i=1}^{n} \ln L_i(\alpha) + \sum_{j=1}^{p} \lambda_j \alpha_j^T \mathbf{B}_j \alpha_j, \tag{3}$$

ahol λ_j választható paraméter, ami szabályozza az egyensúlyt a keresett f_j függvények pontos illeszkedése (első tag) és kellő simasága (második tag) között. Továbbá, $L_i(\alpha)$ az i-edik megfigyelt mintaelem sűrűségértéke a feltételezett eredményváltozó eloszlás, a magyarázó változók i-edik megfigyelése és a teljes α paramétervektor mellett. Természetesen ez a feladat így nem megoldható, hiszen $n \times p$ db paramétert kell becsülni.

Viszont minden $\mathbf{B}_j$ mátrixnak vehetjük $\mathbf{B}_j = \mathbf{U}_j \mathbf{D}_j \mathbf{U}_j^T$ klasszikus spektrálfelbontását. Tehát $\mathbf{D}_j$ mátrix a $\mathbf{B}_j$ mátrix sajátértékeiből álló diagonális mátrix, és $\mathbf{U}_j$ mátrix oszlopaiban a $\mathbf{B}_j$ mátrix sajátértékeihez tartozó sajátvektorok állnak. Válasszuk ki a k_j legnagyobb sajátértéket (figyeljünk, hogy $\sum_{j=1}^{p} k_j < n$ fennálljon) $\mathbf{D}_j$-ből, és legyen $\mathbf{U}_{k_j}$ egy olyan mátrix, melynek oszlopai a kiválasztott k_j legnagyobb sajátértékhez tartozó sajátvektorok. Ezekből megadható $\mathbf{B}_{k_j} = \mathbf{U}_{k_j} \mathbf{D}_{k_j} \mathbf{U}_{k_j}^T$ $n \times k_j$ dimenziós mátrix, ahol $\mathbf{D}_{k_j}$ a $\mathbf{B}_j$ mátrix k_j legnagyobb sajátértékeiből álló diagonális mátrix. Wood [2003] megmutatja, hogy a fenti módon megadott $\mathbf{B}_{k_j}$ minimalizálja a $\|\mathbf{B}_j - \mathbf{B}_{k_j}\|_2$ távolságot adott k_j érték mellett.

Ezek után megoldjuk a (4) minimalizálási feladatot, ahol a GAM modell (2) egyenletében $\mathbf{U}_{k_j} \mathbf{D}_{k_j}$-ket írunk $\mathbf{B}_j$-k helyébe, és ezekkel értékeljük ki az $L_i(\alpha)$-kat $\forall i$-re.

$$\min_{\alpha} -\sum_{i=1}^{n} \ln L_i(\alpha) + \sum_{j=1}^{p} \lambda_j \alpha_j^T \mathbf{D}_{k_j} \alpha_j \,. \tag{4}$$

(4)-ben már csak $\sum_{j=1}^{p} k_j < n$ db paraméter megbecslése szükséges, ami megoldható pl. Newton-Raphson módszerrel.

Gyakorlatilag a thin plate spline illesztését $\forall X_j$ értelmezési tartományának k_j db szakaszán végezzük el k_j db bázisfüggvény segítségével (melyek értéke a $\mathbf{B}_{k_j}$ mátrix oszlopaiban kerül tárolásra az egyes megfigyeléseinkhez). A szakaszhatárokat a spektrálfelbontás alkalmazásával úgy adtuk meg, hogy a lehető legjobban közelítsük meg azt a feladatot (3), amikor a spline függvényeket pontonként ($r_j = n$) illesztjük. Ilyen értelemben tehát egy optimális osztópontrendszerhez és ($r_j = k_j$) spline-rendekhez jutottunk.

Innentől kezdve a változószelekciós feladatban az egyetlen paraméter, amiről dönteni kell, az a k_j értéke. Viszont itt is csak az a fontos, hogy k_j elég nagy legyen ahhoz, hogy a $\|\mathbf{B}_j - \mathbf{B}_{k_j}\|_2$ eltérés ne legyen szignifikáns. Ezt a nullhipotézist formális statisztikai próbával tudjuk tesztelni Augustin et al. [2012] alapján.

A (3) és (4) minimalizálási feladatokban szereplő λ_j paramétereket Wood [2011] alapján korlátozott maximum likelihood (REML: REstricted Maximum Likelihood) módszerrel érdemes meghatározni. A REML módszerben

technikailag 10-szeres keresztvalidáció segítségével keressük azon λ_j-ket, melyek mellett (4)-et megoldva a legkisebb célfüggvény értéket kapjuk.

2.2 A változószelekciós feladat GAM keretben

A következő gyakorlati probléma egy m lehetséges $X = \{X_1, X_2, \ldots, X_m\}$ magyarázó változót tartalmazó halmazból kiválasztani azt a $p \leq m$ változót tartalmazó $\tilde{X} = \{X_1, X_2, \ldots, X_p\} \subseteq X$ részhalmazt, amely a legjobb általánosító képességű modellt eredményezi. A modell általánosító képessége leírja, hogy a modell mekkora pontossággal tud az n elemű mintán kívüli populációról is számot adni a minta alapján kinyert információk alapján. A megfelelő általánosító képesség eléréséhez kompromisszumra van szükségünk a mintaadatok felhasználásának tekintetében. Ha a túl kevés mintaadatot használunk fel, akkor nem nyerünk elég jó képet a valóságról. Ha túl sokat használunk fel, akkor túlságosan „ráfókuszálunk" a mintánkra (Wooldridge [2016]).

Az általánosító képesség növelésére irányuló törekvések következtében született meg a magyarázó változók szelekciójában a parszimónia, azaz a takarékosság elve. A parszimónia elve szerint a $\tilde{X} \subseteq X$ halmazt úgy kell megválasztanunk (szelektálnunk), hogy a lehető legjobb becslési pontosságot érjünk el a lehető legkevesebb magyarázó változó felhasználásával.

A modell általánosító képességét többféleképpen is tudjuk mérni. Jelen tanulmányban a korrigált McFadden-féle pszeudo R-négyzet mutatót alkalmazzuk. A mutatót McFadden [1974] alapján ismertetjük. A mutató megértéséhez fontos bevezetni a telített modell fogalmát. A telített modellnek annyi paramétere van, ahány megfigyelés a mintánkban, így minden megfigyelésre tökéletesen illeszkedik. A megfigyeléseink egyéni devianciája pedig az egyéni log-likelihood növekményének kétszeresét jelenti, ha az aktuális helyett a telített modellt használjuk: $D_i = 2 \ln L_i(\text{telített}) - \ln L_i(\alpha)$. A modell teljes devianciája mintán pedig nem más, mint $D = \sum_{i=1}^{n} D_i$. Legyen D_0 a nullmodell (a csak konstanst tartalmazó modell) teljes devianciája. Ezekkel a jelölésekkel: $R^2 = 1 - \frac{D}{D_0}$. Azért, hogy a modell túlillesztését elkerüljük a megfigyelt mintára vonatkozóan, még korrigálni szükséges az aktuális modell becsült paramétereinek számával:

$$\bar{R}^2 = 1 - \frac{n-1}{n - \sum_{j=1}^{p} r_j}(1 - R^2).$$

A túlillesztés problémáját el lehet kerülni a keresztvalidált pszeudo R^2 kiszámításával is, de ha a célunk, hogy egyszerre több $\tilde{X}$ halmaz teljesítményét is kiértékeljünk, akkor a korrigált R^2 alkalmazása a számítási kapacitásokkal történő takarékosság miatt preferált lehet. Különösen, ha figyelembe vesszük, hogy a REML paraméterbecslési módszer is keresztvalidációt alkalmaz.

A változószelekció során tehát az $\tilde{X}$ halmazt úgy szükséges megválasztani, hogy $\bar{R}^2$ maximális legyen. Azonban X összes részhalmazának megvizsgálása NP-nehéz feladat, hiszen az üres halmaz kizárásával is $2^m - 1$ db lehetséges megoldást kell vizsgálni (Huo – Ni [2007]).

Továbbá, a GAM modellek változószelekciója során alapesetben nem csak arról döntünk, hogy mely változók szerepeljenek a modellben. Azt is vizsgálni szükséges, hogy a bevont változókra milyen rendű b-spline függvényeket illesszünk, és hogy a bevont X_j értelmezési tartományát hol törjük szakaszokra. Viszont a thin plate spline-ok alkalmazásával ezekkel a döntési pontokkal nem kell foglalkoznunk, hiszen csak annyi a dolgunk, hogy egy kezdeti értéket adjunk minden r_j-nek, és ezt az értéket addig növeljük, amíg az Augustin et al. [2012]-féle próba nullhipotézisét el nem tudjuk fogadni a szokásos szignifikancia-szinteken. Ha r_j kezdeti értékét „túl nagynak" választottuk, az nem okoz problémát, mivel a (4) feladat második tagja véd minket attól, hogy az illesztett f_j függvény túl bonyolult legyen.

2.3 A concurvity jelenség GAM keretben

A végső modell értelmezhetősége miatt fontos figyelni a parszimónia-elvére is a változószelekció során. Emiatt jó, ha meg tudjuk állapítani, hogy mikor áll fenn a GAM-ban szereplő X_j magyarázó változók nem-lineáris transzformáltjai között zavaró mértékű összefüggés. Amennyiben a változószelekció eredményéül kapott GAM-ban szereplő változók redundanciától mentesek nem-lineáris értelemben is, akkor azt is biztosíthatjuk, hogy az eredményváltozó értékének alakulását befolyásoló független tényezőket egyértelműen azonosítani tudjuk a szelektált végső modell alapján. Így, ha az elemző célja egy kezdeti, takarékos értelmező modell építése előrejelző modellel szemben, érdemes lehet a változók közti redundancia szűrésére is erőforrásokat allokálni a változószelekció során. Az ilyen „extrém módon" takarékos modellek természetesen magukban hordozzák a kihagyott változók miatti torzítás veszélyét (Wooldridge [2016]), de segíthetnek az elemzőnek azonosítani az eredményváltozót alakító legfontosabb független hatásokat reprezentáló változókat. A módszer előnye a hagyományos dimenziócsökkentési eljárások alkalmazásával szemben, hogy a végső modellben konkrétan megnevezhető változók szerepelnek, adott esetben nehezen értelmezhető faktorok helyett (Jolliffe [1982]).

GAM keretben a változók közti redundancia szintjének megállapításához a multikollinearitás jelenségének nem-lineáris változatát, a concurvity-t (Wood [2017]) szükséges megmérnünk. A concurvity jelenség mérésére GAM modellekben, thin plate spline függvények használata esetén Wood [2017] javasol egy mutatót. A mutató alapötlete, hogy egy f_j függvényből a $\mathbf{B}_j$ mátrixszal reprezentált bázisfüggvényeinek $\mathbf{U}_{k_j}\mathbf{D}_{k_j}$ dekompozíciójával kinyerhető egy g_j függvény, ami más $f_{c\neq j}$ függvények $\mathbf{U}_{k_c}\mathbf{D}_{k_c}$ dekompozíciójának is része. A Wood-féle concurvity mérték ezt a redundancia hatást a $\|g_j\|^2/\|f_j\|^2$ hányados segítségével helyezi el egy $[0,1]$ skálán.

Kétféle concurvity mértéket is rendelhetünk egy X_j változóhoz. Az egyik a megfigyelt concurvity, ami a modell paraméterbecslése során megkapott $\tilde{\alpha}$ vektorral előálló f_j függvények közötti concurvity-t méri. Ezzel szemben a pesszimista concurvity mérték megkeresi azt a $\tilde{\alpha}$ vektort, amivel a $\mathbf{U}_{k_j}\mathbf{D}_{k_j}$-ket súlyozva a legmagasabb concurvity mérték érhető el (Wood [2017]).

Intuitív értelmezésben a $\|g_j\|^2/\|f_j\|^2$ mérték megmutatja, hogy az f_j varianciájának hányadrészét adják olyan bázisfüggvények, melyek az egyéb $c \neq j$ változókra illesztett transzformációs függvényekben is szerepelnek. Ebből az értelmezésből adódik, hogy ha az X_j-hez rendelt concurvity mérték 0,5-nél nagyobb, akkor úgy vehetjük, hogy a változó hatásának értelmezését károsan befolyásoló concurvity van jelen a GAM modellben, mivel a változó hatását leíró függvénytranszformáció varianciájának több mint felét olyan bázisfüggvények adják, melyek más változókra illesztett transzformációs függvényeknek is részei.

Ez a 0,5-höz rendelt vágási pont analóg a lineáris eset $VIF_j > 2$ határával, hiszen ekkor a VIF_j képletben $R_j^2 > 0{,}5$ (Kovács [2008]). Természetesen a VIF_j mutató alapján a szakirodalom több vágási értéket is megad a káros multikollinearitás határaként. Mint például a $VIF_j > 5$ és $VIF_j > 10$, melyek rendre a $R_j^2 > 0{,}8$ és $R_j^2 > 0{,}9$ eseteknek felelnek meg (Hunyadi – Vita [2006]) (Kovács [2008]). Ennek megfelelően GAM-ok esetében a káros concurvity határait is megadhatjuk $\left(\frac{\|g_j\|}{\|f_j\|}\right)^2 > 0{,}8$ és $\left(\frac{\|g_j\|}{\|f_j\|}\right)^2 > 0{,}9$ feltételek formájában. A concurvity jelenségre adott korlát szigorúsága a változószelekció során GAM-ok esetében is a modellező egyéni preferenciájának függvénye.

Jelen tanulmányban a káros concurvity jelenség határának a $\left(\frac{\|g_j\|}{\|f_j\|}\right)^2 > 0{,}5$ vágást tekintjük minden j-re. Ez elsősorban abból az óvatossági megfontolásból adódik, hogy amennyiben a változószelekció eredményeként olyan modellt kapunk, ahol a változók között mért legmagasabb $\left(\frac{\|g_j\|}{\|f_j\|}\right)^2$ érték csak nagyon kevéssel marad el a 0,5-ös határértéktől (pl. 0,49-nek adódik), akkor is jóval a káros concurvity szempontjából megengedőbb határok (mint pl. 0,8 és 0,9) alatt helyezkedjen el. Általános esetben a káros concurvity mértékére megadott vágási érték a vizsgált probléma jellegétől és az elemző preferenciáitól függően megválasztható. Minél magasabb határérték felett tekintünk egy változót káros concurvity által érintettnek, a változószelekció során várhatóan annál több változó fog a végső modellben szerepelni, és ezzel a parszimónia elv sérülését kockáztatjuk (Wood [2017]).

3 Változószelekciós algoritmusok additív modellekben

A 2.2. fejezetben ismertetett, GAM keretben adott változószelekciós feladat megoldására több heurisztikus algoritmus is létezik a szakirodalomban. A legnépszerűbbek közé tartozik pl. Marra – Wood [2011], Schmid – Hothorn [2008] és Belitz – Lang [2008]. Jelen tanulmányban ezekkel az algoritmusokkal nem foglalkozunk, mivel a változószelekció során kísérletet sem tesznek a concurvity jelenség kontrollálására.

A 3.1. és 3.2. alfejezetekben két olyan korszerű algoritmust ismertetünk, ami nem-lineáris változószelekció esetén a célfüggvényben tekintettel van arra is, hogy a kiválasztott magyarázó változók között ne álljon fenn a végső mo-

dell értelmezhetőségét csorbító nem-lineáris összefüggés. Mindkét ismertetett algoritmus a Hall [1999]-féle CFS algoritmus általánosításának tekinthető nem-lineáris esetre.

3.1 Az mRMR algoritmus

Az algoritmus itt ismertetett formáját Song et al. [2012] dolgozta ki Peng et al. [2005] alapján. Az algoritmus neve egy angol kifejezés (minimum Redundancy – Maximum Relevance) rövidítése. Mint a módszer neve is sugallja, ez a változószelekciós eljárás egyszerre kísérel meg optimalizálni arra, hogy olyan változókat válogasson be a modellbe, amelyek szoros kapcsolatban állnak az eredményváltozóval, ám egymással nem állnak jelentős kapcsolatban. Az algoritmusban meg kell adni egy mértéket két változó közös információtartalmának mérésére. Song et al. [2012] ennek mértéknek a Gretton et al. [2005]-féle Hilbert-Schmidt függetlenségi kritériumot (HSIC) választja. Egy x és z valószínűségi változópár mellett a Hilbert-Schmidt kritérium az (5) alakot ölti.

$$\begin{aligned} HSIC(x,z) &= E_{x,x',z,z'}[K(x,x')L(z,z')] + \\ &+ E_{x,x'}[K(x,x')]E_{z,z'}[L(z,z')] - 2E_{x,z}[E_{x'}[K(x,x')]E_{z'}[L(z,z')]]\,, \end{aligned} \tag{5}$$

ahol $K, L : \mathbb{R}^{n\times n} \to \mathbb{R}$ pozitív definit magfüggvények, $E_{x,x',z,z'}$ a várható értéke a $p(x,z)$ együttes eloszlásból vett független (x,z) és (x',z') pároknak. $HSIC(x,z) = 0$, ha x és z függetlenek, egyébként pozitív. Az (5) definíció alapján látható, hogy a $HSIC$ mutató két valószínűségi változó együttes eloszlása alapján határozza meg az összefüggőség mértékét, ezzel a nemlineáris sztochasztikus kapcsolatokat is figyelembe veszi, ellentétben a CFS algoritmusban alkalmazott klasszikus Pearson-korrelációval. A kritérium részleteiről Gretton et al. [2005] ad mélyebb áttekintést. Az mRMR algoritmus és a 3.2. fejezetben tárgyalt HSIC-Lasso algoritmus implementációja is Gauss magfüggvényt

$$K(x,y) = e^{-\frac{\|x-y\|^2}{2\sigma^2}}$$

alkalmaz folytonos változókra, ahol $\sigma = 1$ a változók normalizálása miatt. Ha a változó bináris, akkor Delta magot alkalmaz az algoritmus:

$$K(x,y) = \begin{cases} 1/n_y, & \text{ha } x = y; \\ 0 & \text{egyébként,} \end{cases}$$

ahol n_y az y értékek elemszáma a mintában.

Az mRMR algoritmus alapötlete, hogy egy GAM-ba olyan X_j magyarázó változókat kell választani, amikre $HSIC(Y, X_j)$ magas, ám $HSIC(X_j, X_k)$ alacsony $\forall k \neq j$- re, ahol X_k már szerepel a modellben. Ezt a két szempontot az mRMR algoritmus egy legjobb részhalmaz elvű lineáris keresésben egyesíti.

Legyen $X = \{X_1, \ldots, X_m\}$ a lehetséges magyarázó változók halmaza, és egy GAM-ba keressük a legjobb $\tilde{X} \subseteq X$ részhalmazt a 2.2. fejezetben

megadott feladat szerint. Tegyük fel, hogy már p elemet kiválasztottunk X-ből $\tilde{X}$-be. Ezen a ponton az mRMR algoritmus azt az X_j magyarázó változót választja be $\tilde{X}$-be, amire

$$j = \arg\max_{k \in X \setminus \tilde{X}} \frac{HSIC(Y, X_k)}{\sum_{i \in \tilde{X}} HSIC(X_i, X_k)/|\tilde{X}|} .$$

Láthatjuk, hogy a maximalizálandó célfüggvény nő, ha X_k közös információtartalma a célváltozóval is nő. Azonban a célfüggvény csökken, ha a modellbe beválogatott más X_i magyarázó változókkal is nő X_k közös információtartalma.

Az eljárásból láthatjuk, hogy az mRMR algoritmus lineáris keresés, amihez szükséges előre megadni $|\tilde{X}|$-et, egyébként más kilépési kritérium hiányában a fentebb leírt eljárás X-ből minden magyarázó változót bevesz a modellbe.

3.2 A HSIC-Lasso algoritmus

Az mRMR algoritmust Yamada et al. [2018] fejlesztette tovább. A fejlesztés ötlete, hogy a $HSIC$ kritériumokhoz magyarázó változóként egy γ_j együtthatót hozzárendel. Ezzel gyakorlatilag egy lineáris Lasso-problémává (6) alakítható a változószelekciós feladat.

$$\max_{\Gamma \geq 0} \sum_{j=1}^{m} \gamma_j HSIC(X_j, Y) - \frac{1}{2} \sum_{j,k=1}^{m} \gamma_j \gamma_k HSIC(X_j, X_k) - \lambda \|\Gamma\|_1 \qquad (6)$$

$\lambda > 0$ egy paraméter a feladatban, ami keresztvalidáció segítségével úgy megválasztható, hogy a kiválasztott λ mellett a legnagyobb célfüggvényértéket kapjuk. Továbbá, $\Gamma = [\gamma_0, \gamma_1, \ldots, \gamma_m]$. A (6) feladat megoldásában az együtthatók L_1 normájával történő büntetés miatt a modellből elhagyható változók γ_j együtthatói 0-nak adódnak. A célfüggvényben korrigálunk a $HSIC(X_j, X_k)$, $j \neq k$ magyarázó változó-párok függetlenségi kritériumával, ezzel a több redundáns magyarázó változó kiválasztásának elkerülését ösztönözzük. (6) feladat bevezetésének lényege, hogy megoldásához nem szükséges $|\tilde{X}|$ ismerete.

(6) megoldása kimondottan memóriaigényes. Emiatt Climente-González et al. [2019] a HSIC-Lasso feladat megoldásához a megfigyelt mintát $B \ll n$ elemű blokkokra bontja fel, amelyekre külön ki lehet számítani a $HSIC$ kritériumokat, és ezek összegzésével kaphatunk a teljes mintára értelmezett $HSIC$ kritériumot. A blokkok használatával a HSIC-Lasso algoritmus futtatható nagyobb méretű adatbázisok esetén is.

Mind az mRMR, mind a HSIC-Lasso algoritmus esetében is érdemes észrevenni, hogy a HSIC kritérium alkalmazása a magyarázó változók közti kapcsolat szorosságának mértékét csak páronként vizsgálja. Az algoritmusok nem kezelik azt az esetet, amikor egy magyarázó változó több más magyarázó változó többváltozós függvényeként áll elő. Emiatt az mRMR-rel és a HSIC-Lassoval nyert GAM-ok esetében számítani lehet káros mértékű, az értelmezéseket torzító concurvity jelenségre.

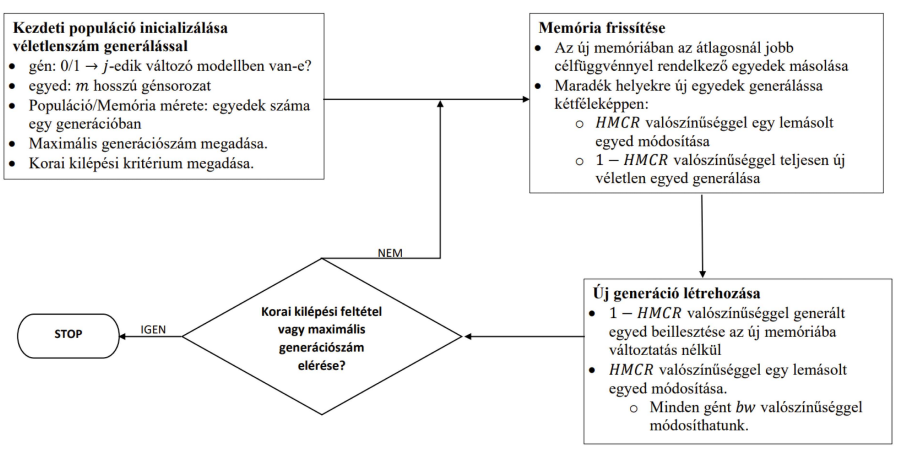
4 Hibrid genetikus-harmóniakereső algoritmus változószelekcióra GAM keretben

Korábbi munkáinkban (Láng et al. [2017] és Kovács [2019]) egy hibrid genetikus – harmóniakereső algoritmust (továbbiakban Hibrid algoritmus) javaslunk a változószelekciós feladat megoldására lineáris modellek esetében. Az algoritmus a magyarázó változók VIF értékén keresztül a szelekciós folyamat során nem csak a változók közti páronkénti káros korrelációkra szűr. Ebben a fejezetben megadjuk az algoritmus kiterjesztését GAM keretre, thin plate spline-ok segítségével reprezentálva a magyarázó változók f_j transzformáló függvényeit.

A Hibrid algoritmus folyamatábráját az *1. ábrában* adjuk meg. Célfüggvény jelen tanulmányban a McFadden-féle $\bar{R}^2$ mutató.

Az 1. ábrán látható, hogy a Hibrid algoritmus működésében megőrizzük a genetikus algoritmus párhuzamosítható populáció (a harmóniakereső algoritmus terminológiájában memória) kezelését, ám az algoritmus keresztezés nevű rekombinációs operátorát lecseréljük a harmóniakereső algoritmus valószínűségi alapú rekombinációs operátorára.

A cserére azért van szükség, mivel a genetikus algoritmus keresztezési operátora alapvetően olyan problémák esetén alkalmazható hatékonyan, ahol az egyedek minőségét érdemes részenként, géncsoportonként javítani, és az egyes részmegoldások keresztezésével új megoldásokat létrehozni. Azonban, mivel az adatbázisok többségében a változók sorrendje véletlenszerű, így általában nincsenek csak az adott egyed génsorozatának elején vagy végén kialakuló megfelelő génmintázatok. Ráadásul egyetlen változó bevétele vagy elhagyása a modellből drasztikusan megváltoztathatja a célfüggvényünk értékét. Tehát szükségünk van az egyedek nagyobb fokú véletlenségét biztosító rekombinációs operátorokra. Ezeket pedig a harmóniakereső algoritmusból tudjuk kölcsönözni.



1. ábra. A Hibrid algoritmus folyamatábrája. *Forrás:* saját szerkesztés.

A korábbi memória átlagosnál jobb egyedeiből történő választás valószínűsége ($HMCR$ az angol harmony memory consideration rate kifejezésből) a futás során nő, míg a mutáció (módosítás) valószínűsége (bw) a futás során csökken. Ezzel a finomhangolással azt a hatást váltjuk ki, hogy az algoritmus futásának elején minél agresszívebb lesz a populáció fennmaradó helyeire az egyedek generálása. Inkább a teljesen új, véletlen egyed generálását, vagy egy korábbi egyed nagy tartományban történő módosítást támogatjuk a korai szakaszban. Ahogy az algoritmus futása során egyre inkább közelebb kerülünk az optimumhoz, a keresési tér egyre kisebb részét kell bejárnunk az új egyedek generálása során. A Hibrid algoritmus korai kilépési feltétele akkor érvényesül, ha az utolsó valahány lépésben a célfüggvény értéke nem változik a populációnk legjobb egyedének esetében.

Az 1. ábrán látható, hogy az algoritmus a változószelekciós feladat során a lehetséges megoldásokat egy m hosszú bitsorozat segítségével reprezentálja. Tehát csak arról döntünk, hogy egy változót beemeljünk-e a modellbe vagy sem. Az algoritmus akkor alkalmazható GAM keretben is, ha ez nem változik. Amennyiben a változóra illesztett spline rendjét és a szakaszhatárokat is meg kell határozni, akkor már összetettebb reprezentáció szükséges a gének szintjén. Szerencsére, a 2.1. alfejezetben ismertetett thin plate spline-ok alkalmazásával az egyedek bináris reprezentációja és a változószelekciós feladat keresési tere nem változik meg a lineáris esethez képest.

Az algoritmusban azt a technikát követjük, hogy ha egy X_j magyarázó változót beveszünk a GAM modellbe, akkor az alapértelmezés szerint $k_j = 10$. Amennyiben X_j értékkészlete kisebb, akkor k_j az X_j magyarázó változó lehetséges értékei számával lesz egyenlő. Amennyiben az Augustin et al. [2012]-féle próbák p-értéke $\alpha = 0{,}01$ alatti, akkor a $k_j$értékét 5-ösével növeljük addig, amíg a változóhoz megfelelően nagy k_j-t nem választottunk.

Az algoritmusnak nagyon fontos tulajdonsága, hogy a változószelekció folyamán a concurvity jelenséget teljes egészében szűrni akarja a modellekből. Emiatt szeretnénk elérni, hogy a memória frissítése során csak olyan megoldásokat vigyünk át az új memóriába, melyekre a 2.2. alfejezetben meghatározott concurvity mérték 0,5 alatti. Az algoritmusban paraméterként megadható, hogy a megfigyelt vagy a pesszimista concurvity mértékre vonatkozzon-e ez a korlát. Ha az i-edik egyed teljesíti a korlátot minden változóra, akkor a C_i bináris változó 1 értéket vesz fel, egyébként 0-t. Továbbá az is szempont, hogy olyan modelleket preferáljunk, amikben csak olyan változók szerepelnek, amelyek spline függvényében elutasíthatjuk az $\alpha_j = \mathbf{0}$ nullhipotézist. A megadott nullhipotézis tesztelésére Marra – Wood [2011] ad meg egy χ^2-próbát, az algoritmusban ezt alkalmazzuk. Ha a felhasználó által megadott szignifikanciaszinten minden változóra elutasítható az $\alpha_j = \mathbf{0}$ nullhipotézis az i-edik egyedben, akkor S_i bináris változó 1 értéket vesz fel, egyébként 0-t.

Technikailag a korlátok beépítése úgy valósul meg, hogy amikor a memória frissítése során kiválogatjuk az átlagosnál jobb egyedeket az aktuális memóriából, akkor a memória átlagos célfüggvény értékére a (7) formulával adott súlyozott átlagot alkalmazzuk, és csak olyan i egyed minősülhet átlagosnál

jobb egyednek, amire $C_i = S_i = 1$.

$$\bar{R}_M^2 = \frac{\sum_{i=1}^{N} \bar{R}_i^2 \cdot C_i \cdot S_i}{\sum_{i=1}^{N} C_i \cdot S_i}, \tag{7}$$

ahol N a memória méretét, $\bar{R}_i^2$ az i-edik egyed célfüggvény értékét jelöli. Amennyiben $\sum_{i=1}^{N} C_i \cdot S_i = 0$, akkor $\bar{R}_M^2$ egyszerűen az $\bar{R}_i^2$ értékek számtani közepe, és minden aktuális memóriában lévő egyed minősülhet átlagosnál jobb egyednek.

Fontos észrevenni, hogy az algoritmusba épített korlátok miatt könnyen előfordulhat, hogy a véletlenszerűen generált kezdeti memóriában nem lesz olyan egyed, amit továbbengedünk az új memóriába a frissítés során és megfelel a $C_i = S_i = 1$ korlátoknak. Így könnyen lehet, hogy több generációig tart, míg találunk olyan megoldásokat, amelyek minden korlátot kielégítenek. Tehát az algoritmus futásideje a kezdeti memória minőségétől függ. Emiatt érdemes lehet az algoritmust vagy nagy memóriamérettel futtatni, vagy úgy, hogy egy futásidőben könnyen kezelhető memóriaméretet választunk, és többször lefuttatjuk az algoritmust egy alacsonyabb maximális generációszám mellett. A több futási eredmény legjobb célfüggvényértékkel rendelkező megoldását tekintjük optimumnak. Mint az 5. fejezetben látni fogjuk, az utóbbi stratégia kimondottan jól működik nagyobb méretű adatbázisok esetében.

5 Numerikus eredmények két valós adatbázison

A Hibrid algoritmus összevetése a mRMR és HSIC-Lasso algoritmusokkal két valós adatbázison történik.

Az első adatbázis forrása Yeh [1998], és 1030 db betongerenda 9 ismérvét tartalmazza. Feladatunk a gerendák nyomószilárdságának becslése a gerendák hét összetevője és a koruk függvényében. A változószelekciós feladat kicsi ($m = 8$), így a globális optimum könnyen megadható a lehetséges magyarázó változók összes részhalmazának legenerálásával. Az adatbázis használatának a célja, hogy megvizsgáljuk, hogy az ismert globális optimumot milyen hatékonysággal azonosítják a vizsgált algoritmusok.

A második vizsgált adatbázisban az a feladatunk, hogy egy tajvani bank ügyfeleire becsüljük meg, hogy az ügyfél a lekérdezés időpontjától számított 1 hónapos időtartamon belül csődöt fog-e jelenteni hitelkártya adósságára. Az adatbázis 30 000 rekordot tartalmaz, 26 lehetséges magyarázó változóval a kategorikus változók dummy kódolása után. Az adatok forrása Yeh – Lien [2009]. A változószelekciós feladat jelen esetben az összes részhalmaz generálásával már nem oldható meg, az alkalmazott algoritmusok legjobb modelljeit csak önmagukban tudjuk vizsgálni, nincs referenciapont.

A két vizsgált adatbázis változóinak listáját az 1. táblázat tartalmazza.

Betongerendák adatbázis változói
Cement – a betongerenda cementtartalma kg/m3-ben
BlastFurnaceSlag – a betongerenda salaktartalma kg/m^3-ben
FlyAsh – a betongerenda pernyetartalma kg/m^3-ben
Water – a betongerenda víztartalma kg/m^3-ben
Superplasticizer – a betongerenda folyósítóanyag-tartalma kg/m^3-ben
CoarseAggregate – a betongerenda durva aggregátumtartalma kg/m^3-ben
FineAggregate – a betongerenda finom aggregátumtartalma kg/m^3-ben
Age – a betongerenda kora napokban = gyártástól eltelt napok száma
CompressiveStrength – a betongerenda nyomószilárdsága MegaPascalban

Banki ügyfelek adatbázis változói
LIMIT_BAL – Az ügyfél hitelkerete a kártyáján (ezer tajvani dollárban)
SEX – Az ügyfél neme (férfi; nő)
EDUCATION – Iskolai végzettség (1 = általános iskola; 2 = érettségi; 3 = egyetem; 4 = egyéb)
MARRIAGE – Családi állapot (1 = házas; 2 = egyedülálló; 3 = egyéb)
AGE – Életkor (években)
PAY_X – az ügyfél törlesztési státusza, a lekérdezés előtt X hónappal $X \in \{0, 2, 3, 4, 5, 6\}$ (-1 = egy hónap túlfizetés; 0 = pontos fizetés; 1 = egy hónapnyi késedelem; 2 = 2 hónap késedelem stb.)
BILL_AMTX – A fennálló kártyatartozás összege (ezer tajvani dollárban), X hónappal a lekérdezés előtt $X \in \{1, 2, 3, 4, 5, 6\}$
PAY_AMTX – A lekérdezés előtt X hónappal fizetett törlesztőrészlet összege (ezer tajvani dollárban) $X \in \{1, 2, 3, 4, 5, 6\}$
default_nextMonth (1 = az ügyfél csődöt jelentett hitelkártya adósságára egy hónappal az adatgyűjtés után; 0 = az ügyfél nem jelentett csődöt hitelkártya adósságára egy hónappal az adatgyűjtés után)

1. táblázat. A vizsgált adatbázis változói, az eredményváltozó dőlttel jelölve.
Forrás: saját szerkesztés.

Az adatbázisokon minimális adattisztítási lépéseket kellett elvégezni. A betongerendák esetében az adatbázis tartalmazott kilógóan öreg gerendákat. Ezeket eltávolítottuk az elemzésből, így 916 megfigyelés maradt.

A banki ügyféladatbázisban Yeh – Lien [2009] alapján megállapíthattuk, hogy magas a hibásan kódolt kategorikus változókkal rendelkező ügyfelek száma. A MARRIAGE változóban 0, míg EDUCATION változóban 0, 5, 6 érvénytelen kódok fordultak elő. A Tukey-féle külső kerítések BILL_AMT4-ben felfelé extrém kilógó értékeket, míg a BILL_AMT6 változó esetében lefelé kilógó értékeket (tehát extrém magas túlfizetése van az ügyfélnek a törlesztőrészleteire) jeleztek, amiket leszűrtünk az adatbázisból. Ezzel 22 165 megfigyelésünk maradt.

Mindkét megtisztított adatbázist 70%-30% arányban osztottuk fel tanító- és tesztmintákra. Az esetleges keresztvalidációk mindig a teljes tanító adatbázison történtek. Az adattisztítási lépések és a vizsgált algoritmusok implementálása és futtatása az R 3.5.3 verziójában történt. Ez alól kivételt képez a HSIC-Lasso algoritmus, aminek jelenleg csak Python implementációja létezik (Climente-González et al. [2019]). Ez utóbbi implementációt a Repl.it online fejlesztőkörnyezetben futtattuk. A GAM modellek becslését az R *mgcv* csomagjával végeztük el (Wood [2017]), míg az mRMR algoritmust az R *mRMRe* csomagjának segítségével alkalmaztuk (De Jay et al. [2013]). A Hibrid algoritmus esetén a concurvity korlátot a pesszimista mérték alapján vizsgáljuk, és az $\alpha_j = \mathbf{0}$ nullhipotézis vizsgálatához választott szignifikancia-szint 5%. Az

elkészült tanító- és tesztminták (*Rda* formátumban), R szkriptfájlok és futási eredmények a https://github.com/KoLa992/Hybrid-algorithm-for-GAMs linken találhatók meg mindkét adatbázisra.[2]

5.1 A betongerendák adatbázis eredményei

Az adatbázisban a célváltozó folytonos és közel normális eloszlású (Yeh [1998]). Ezzel a GAM link függvényének az identitás alkalmazható. A modellek értékeléséhez egyszerűen a tesztmintán mért $R^2 = \text{cor}(Y, \hat{Y})$ értéket használhatjuk. A globális optimum ismert. Azon modellek közül, melyekre $C_i = S_i = 1$, a maximális R^2 értéket a tesztmintán a *Cement + Blast Furnace Slag + Water + Age* változókombináció adja.

A Hibrid algoritmust a véletlen elemei miatt 30-szor futtatjuk le az adatbázison. A memória mérete 15, az induló $HMCR$ 0,2 és az induló mutációs (bw) valószínűség 0,9. A maximális lépésszám szintén 15, mivel az összes lehetséges eset száma $2^8 = 256$. Amennyiben kb. $15 \times 15 = 225$ modell[3] megvizsgálásával nem találjuk meg a globális optimumot, akkor már felesleges tovább folytatni az iterációkat. A korai kilépés változatlan legjobb célfüggvényre vonatkozó feltétele 5. A Hibrid algoritmus által legtöbbször ($12/30 = 0{,}4 = 40\%$) adott megoldás a globális optimum.

A vizsgált algoritmusok megoldásai a *2. táblázatban* találhatók. Az mRMR algoritmus esetében felhasználjuk azt az a priori tudást, hogy a globális optimumban $|\hat{X}| = 4$.

Algoritmus	Kiválasztott változók neve	Kiválasztott változók száma	Concurvity korlátot sértő változók	Maximális concurvity mérték	R^2 (Teszt) %
mRMR	Cement, BlastFurnaceSlag, Superplasticizer, Age	4	-	0,4876	81,580
HSIC-Lasso	Cement, BlastFurnaceSlag, Water, Superplasticizer, Age	5	Water, Superplasticizer	0,8656	86,382
Hibrid alg.	*Cement, BlastFurnaceSlag,* Water, Age	*4*	-	*0,4233*	*84,363*

2. táblázat. A vizsgált algoritmusok eredményei a betongerendák adatbázison. *Forrás:* saját szerkesztés.

Látható, hogy a HSIC-Lasso megoldása nem teljesíti a concurvity korlátokat. A korlátokat a Water és a Superplasticizer változók sértik meg. Ennek az oka, hogy az építőipari betongerendákban felhasznált folyósítóanyag (Superplasticizer) mennyisége a víz/cement arány függvénye (Muhit [2013] és Plank et al. [2009]). Tehát a Superplasticizer változó nagy pontossággal közelíthető a Cement és a Water változók többváltozós függvényeként. Ezt a jelenséget a HSIC-Lasso algoritmus figyelmen kívül hagyja, mivel a magyarázó változók közti redundanciát csak páronként szűri. Ellenben a Hibrid algoritmus a concurvity jelenséget közvetlenül szűri. Ezzel pedig meg tudja szűrni a szelekció

[2] A futtatáshoz használt hardver konfiguráció: Intel Core i7-8750H 2,20 GHz processzor, 8 GB 2666 MHz DDR4 RAM.

[3] Nem számolva a legjobb modellek ismétlődésével az egyes populációk között.

végén kapott változóhalmazt a káros többváltozós együttmozgásoktól. Jelen esetben, mivel az mRMR esetében felhasználjuk azt az a priori tudást, hogy a globális optimumban $|\tilde{X}| = 4$, így az algoritmus végső megoldásában nem jelentkezik káros concurvity jelenség, hiába szűri a magyarázó változók redundanciáját ez az algoritmus is csak páronként. Ellenben a megoldás R^2 értéke elmarad a globális optimumtól, és a legmagasabb concurvity mérték is magasabb, mint a Hibrid algoritmus által azonosított optimális megoldásban.

A betongerendák adatbázis kis mérete lehetőséget ad a Hibrid algoritmus paramétereinek finomhangolásához. Az algoritmus a globális optimumot a 30 futtatás átlagában az összes lehetőség 39%-ának átvizsgálásával meg tudja találni, ha a korai kilépési feltétellel áll le az algoritmus. Ha azt feltételezzük, hogy ahol az algoritmus nem találja meg 15 generációból az optimumot, ott mind a 256 modellt meg kell vizsgálni, akkor azt mondhatjuk, hogy a Hibrid algoritmus átlagosan az összes lehetőség 75%-nak átvizsgálásával meg tudja találni a globális optimumot. Ezen a paraméterek állításával lehet még javítani.

Az algoritmusban a *3. táblázatban* szereplő paraméterek ceteris paribus optimalizálását végezzük el.

Paraméter	Optimális érték
Memória (Populáció) mérete	20
Induló $HMCR$ valószínűség	5%
Induló mutációs (bw) valószínűség	90%
Maximális $HMCR$ valószínűség	35%
Minimális mutációs (bw) valószínűség	10%

3. táblázat. A Hibrid algoritmus optimalizált paraméterei a betongerendák adatbázison. *Forrás:* saját szerkesztés.

A vizsgálat során maximális lépésszámot mindig úgy állítjuk be, hogy az összes lehetőség kb. 75-78%-nak átvizsgálása után álljon le az algoritmus. A korai kilépés feltétele nem változik. Ez a 256 lehetséges részhalmaz esetén kb. 192-200 modell megvizsgálását jelenti[4].

Az adott paraméterérték hatékonyságát úgy mérjük, hogy megvizsgáljuk, 30 futtatásból hányszor találja meg az algoritmus a globális optimumot vagy a második legjobb megoldást. A második legjobb megoldásban is a pesszimista concurvity mértékek maximuma 0,461. Tesztmintán $R^2 = 81,555\%$ (az mRMR megoldással gyakorlatilag egy szinten van). A többi vizsgált algoritmussal összevetve ekkor is még mindig használható eredményt kapunk: nem jelentősen alacsonyabb az R^2, és nincs káros concurvity jelenség.

A 3. táblázat eredményei alapján elmondható, hogy a Hibrid algoritmusban a teljesen véletlen új egyed generálásra érdemes nagyobb súlyt helyezni, de nem érdemes teljesen elhagyni a memóriából történő kontrollált egyedgenerálást sem (5-ről 35%-ig emelhető a $HMCR$). Ezt a megfigyelést támasztja alá az a tény is, hogy a memóriából történő új egyed előállítás esetén is érdemes az algoritmus első lépéseiben magas mutációs (bw) valószínűséget

[4]Természetesen a legjobb egyedek örökítése miatt a valóságban ennél valamivel kevesebb különböző modellt vizsgál meg a Hibrid algoritmus.

alkalmazni (90%), ám az induló érték nagyon alacsonyra (10%) csökkentése az iterációk során szintén kifizetődő.

További fontos tapasztalat, hogy az optimális paraméterek alkalmazása mellett a futtatások valamivel több, mint harmadában (12/30 esetben) úgy találja meg az algoritmus a globális optimumot vagy a második legjobb megoldást, hogy ez a megoldás már az algoritmus 5. iterációja előtt a memóriába került (20-as memóriaméret mellett). Azaz a keresési térnek kb. $\frac{5 \cdot 20}{2^8} \approx 0{,}4$ részét vizsgálja csak át az algoritmus, mire megtalálja a két legjobb megoldás egyikét. A tapasztalat így arra enged következtetni, hogy előnyösebb a 4. fejezetben alkalmazott második stratégiát alkalmazni a kezdeti memóriák roszszabb minőségének kezelésére: egy futásidőben könnyen kezelhető memóriaméretet választunk, és többször lefuttatjuk az algoritmust egy alacsonyabb maximális generációszám mellett.

Az optimális paraméterek ceteris paribus keresésének részletes eredményei a mellékletekben találhatók meg.

5.2 A banki ügyfelek adatbázis eredményei

Az adatbázisban a célváltozó Bernoulli eloszlású (Yeh – Lien [2009]), így a GAM link függvénye a logit, és modellek értékeléséhez a tesztmintán mért ROC görbe alatti területet (*AUC*) használjuk.

Jelen adatbázison a globális optimum nem ismert. Meghatározni nem is lehet az összes lehetséges részhalmaz meghatározásával, mivel egy átlagos modell kiszámításának költsége magas. 100 véletlenszerű egyedet (változókombinációt) szimulálva egy modell átlagos kiszámítási idejének 95%-os alsó konfidencia-intervalluma 0,35 perc. Ezzel számolva a 2^{26} lehetséges megoldás generálásához 95%-os megbízhatósággal várhatóan legalább $\frac{0{,}35 \cdot 2^{26}}{60 \cdot 24 \cdot 365{,}25} = 44{,}72$ év szükséges. A szimuláció során ráadásul alkalmazzuk Wood et al. [2015] javaslatát, és egy GAM kiszámítását párhuzamosítjuk a (2)-ben szereplő $\mathbf{U}_{k_j}\mathbf{D}_{k_j}$ mátrixok QR dekompozíciójának segítségével.

Egy modell kiszámítását megkísérelhetjük tovább gyorsítani azzal, hogy nem a teljes tanító adatbázison, hanem csak egy FAE módon kiválasztott részhalmazán végezzük el a számítást. A részhalmaz méretének szignifikánsan kisebbnek kell lennie a tanító adatbázis méreténél, hogy a futásidőt javítsa, ám kellően nagynak kell lennie ahhoz, hogy a modell jellemzői ne térjenek el jelentősen a teljes tanító adatbázisra számolt értékektől. Jelen esetben $n = 5000$-et alkalmazunk. Ekkor 100 véletlen minta generálása után azt tapasztaljuk, hogy a teljes, minden változót tartalmazó modell McFadden-féle korrigált R-négyzet értékének mintavételi eloszlása $N(0{,}217; 0{,}011)$. A Shapiro-Wilk normalitásteszt p-értéke 0,9125-nek adódik. A teljes tanító adatbázison $\bar{R}^2 = 0{,}217$. Tehát 95%-os valószínűséggel egy 5000 elemű almintán felépített GAM $\bar{R}^2$ értéke legfeljebb csak $2 \cdot 0{,}011 = 0{,}022$-vel fog eltérni a teljes tanító adatbázison számított modell $\bar{R}^2$ értékétől. Egy $n = 5000$ alminta alkalmazása esetén egy modell átlagos kiszámítási idejének 95%-os alsó konfidencia-intervalluma 0,207 perc. Egy modell kiszámítása továbbra is párhuzamosított. Ezzel számolva az összes lehetséges megoldás

előállításához 26,46 év szükséges.

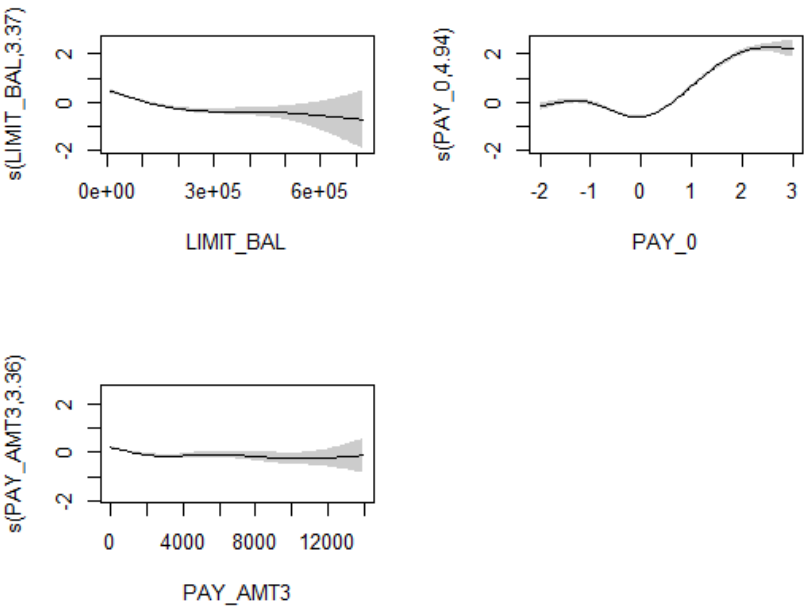
A részhalmazok átvizsgálását tovább gyorsíthatjuk azzal, hogy nem egy modell kiszámítását párhuzamosítjuk, hanem több részhalmazhoz tartozó modellt számítunk ki egyszerre párhuzamosan. Egy ilyen párhuzamosítást az R *foreach* (Weston [2019a]) és *doParallel* (Weston [2019b]) csomagjainak segítségével tudunk implementálni. Ez a megoldás a 100 szimulált egyed teljes kiszámítási idejét az $n = 5000$ elemű almintán tapasztalt 21,373 percről 3,374 percre csökkenti. Ezzel várhatóan az összes lehetséges megoldás generálásához szükséges idő 4,18 évre csökken. Ez jelentős csökkenés, ám még így sem reális, hogy belátható időn belül meg tudjuk találni a változószelekciós feladat globális optimumát. Ráadásul a futásidő nyereséget biztosan túl is becsüljük, hiszen 100 egyed párhuzamos kiszámítása könnyebben megoldható, mint pl. 10 000 egyed párhuzamos kiszámítása.

A 100 szimulált egyed esetén vizsgált GAM modell számításból a legfontosabb tanulság, hogy jobban kifizetődik több egyedhez tartozó modell egyszerre történő kiszámítása, mint egy modell kiszámításának párhuzamosítása. Ezt a tapasztalatot tudjuk hasznosítani a Hibrid algoritmus implementációjakor jelen feladatra. A 4. fejezetben bemutattuk, hogy a Hibrid algoritmus a genetikus algoritmusból örökölt egyedkezelése miatt képes a memóriában lévő megoldások párhuzamosított kiszámítására. Ezen kívül a futásidőt tovább gyorsítjuk egy $n = 5000$ elemű alminta alkalmazásával is.

A betongerendák adatbázison szerzett tapasztalatokat is tudjuk hasznosítani a Hibrid algoritmus implementációja során. Mivel az algoritmus gyakran talál egy elég jó megoldást a kezdeti iterációkban, így a konkrét kísérletünkben 60 elemű populációval dolgozunk, fixen 6 iteráción keresztül. Az algoritmus futtatását 5-ször ismételjük meg, és kiválasztjuk a legjobb modellt. A $HMCR$ és mutációs valószínűségek beállításaihoz a betongerendák adatbázison tapasztalt legjobb beállításokat alkalmazzuk. A $HMCR$ valószínűség 5%-ről nő a 6 iteráció során 35%-ra, míg a mutációs (bw) valószínűség 90%-ról csökken 10%-ra. Ezen paraméterek mellett futtatjuk le a Hibrid algoritmust 20-szor és vizsgáltuk meg az eredményül kapott modelleket és futásidőket.

A 20 futtatás átlagos futásideje 2,45 óra, 0,25 óra szórással. A 20 eredményből mindegyik megoldás megfelel a concurvity és szignifikancia korlátoknak. Továbbá, csak 3 esetben azonosíthatunk olyan modellt, aminek a tanító mintán mért $\bar{R}^2$ értéke jelentősen eltér a teljes modell 0,217-es értékétől (jelentős eltérésnek a 0,19-nél kisebb $\bar{R}^2$-eket vesszük). A 20 futtatás során talált legjobb megoldásban $\bar{R}^2 = 0{,}209$, és 3 változót használ: LIMIT_BAL, PAY_0 és PAY_AMT3. A kiválasztott változók hatásai a hitelkártyacsőd valószínűségének logit transzformáltjára a *2. ábrán* találhatók.

A 20 futtatás során tapasztalt legjobb modell eredményeit vetjük össze a teszt adatbázison az mRMR és HSIC-Lasso algoritmusok eredményeivel. Ezen kívül benchmarkként alkalmazunk a feladatra döntési fa és véletlen erdő modelleket is. A döntési fát az *rpart* R csomaggal (Therneau – Atkinson [2018]), míg a véletlen erdőt a *caret* R csomaggal (Recurvive Feature Elimination, RFE változószelekcióval kombinálva) (Kuhn et al. [2019]) implementáltuk.



2. ábra. A Hibrid algoritmus végső modelljében a magyarázó változókra illesztett spline függvények a banki ügyfelek adatbázison, 95%-os konfidenciaintervallummal.
Forrás: saját szerkesztés.

A futásidők összehasonlíthatósága miatt minden alkalmazott módszert 20-szor futtatunk, és az adatbázis nagyobb mérete miatt vizsgáljuk a futásidők átlagát és szórását is az eredményül kapott modellek teszt adatbázison mért *AUC* értéke mellett. Az eredményeket a *4. táblázat* tartalmazza.

Algoritmus	Kiválasztott változók neve	változók száma	Concurvity korlátot sértő változók	Maximális concurvity mérték	AUC (Teszt)	Futásidők átlaga (perc)	Futásidők szórása (perc)
mRMR	AGE, PAY_0+2+5, PAY_AMT1+6	6	PAY_2	0,8729	0,7441	1,629	0,035
HSIC-Lasso	AGE PAY_0+2+ 3+4	5	PAY_0+2+3	0,9038	0,7409	0,446	0,048
Hibrid alg.	*LIMIT_BAL PAY_0, PAY_AMT3*	*3*	-	*0,2267*	*0,7488*	*147,257*	*15,079*
Döntési fa	PAY_0	1	-	-	0,6413	0,0127	0,0004
Véletlen erdő (RFE szel.)	mindegyik	26	LIMIT_BAL, PAY_0+2+ 3+4+5+6, BILL_AMT1+ 2+3+4+5+6, PAY_AMT1+ 2+3+4+5+6,	0,9834	0,7562	132,543	0,758

4. táblázat. A vizsgált algoritmusok eredményei a banki ügyfelek adatbázison. Forrás: saját szerkesztés.

A 4. táblázat alapján elmondható, hogy a véletlen erdő algoritmusban nem kifizetődő változószelekciót végezni, és ez az algoritmus adja a legjobb teljesítményt a tesztadatokon. Azonban a szelektált GAM-ok nem maradnak el jelentősen a véletlen erdő teljesítményétől, és lényegesen kevesebb magyarázó változót használnak fel a hasonló pontosság eléréséhez. A döntési fa modell túl takarékos, csak egy magyarázó változót használ fel, így a teljesítménye látványosan elmarad a többi vizsgált modelltől.

Megfigyelhetjük, hogy a PAY_0 változót (ügyfél törlesztési státusza a lekérdezés hónapjában) mindegyik modellben érdemes szerepeltetni, ám a változó értéke korábbi időszakokban már redundanciát (így a concurvity korlátok megsértését) okoz a modellekben. Ennek oka, hogy a fizetési státusz a lekérdezés hónapjában jól közelíthető a korábbi hónapok fizetési státuszának többváltozós függvényével. Azaz a lekérdezés hónapjának fizetési státusza a korábbi hónapok fizetési státuszára vonatkozó információt is magában hordozza, így ezek szerepeltetése a modellben nem csökkenti érdemben a csődesemény előrejelzésének bizonytalanságát. Ezt a jelenséget a redundanciát csak páronként vizsgáló mRMR és HSIC-Lasso algoritmusok nem detektálják. Az mRMR algoritmus modelljében a PAY_2 változó jól közelíthető a PAY_0 és PAY_5 változók kétváltozós függvényeként. Ez egy érthető jelenségnek tűnik, hiszen a lekérdezés előtt 2 hónappal mért fizetési státuszban feltehetően az 5 hónappal korábbi állapot hatása „még megjelenik", a 2 hónappal későbbi státusz hatása pedig „már megjelenik". Hasonló jelenség okozza a HSIC-Lasso által preferált modellben tapasztalt redundanciákat is. Ellenben, a Hibrid algoritmus segítségével rájöhetünk, hogy a lekérdezés előtt 3 hónappal fizetett törlesztőrészlet összegének (PAY_AMT3) szerepeltetésével a modellben új, nem-redundáns információhoz juthatunk az ügyfelek csődvalószínűségéről. Tehát azt mondhatjuk el, hogy a lekérdezés időpontja előtti hónapok fizetési fegyelmére vonatkozó új információt a 3 hónappal korábban fizetett törlesztőrészlet összege hordoz, a lekérdezés hónapjában fennálló hátralékos hónapok száma mellett. A hátralékos hónapok számának korábbi értékei a csődvalószínűségre nem szolgáltatnak érdemi új információt.

A futásidők vizsgálata során egyértelmű hátrányban vannak a lehetséges magyarázó változók halmazának több részhalmazát is megvizsgáló algoritmusok (Hibrid és RFE véletlen erdővel) a lineáris kereső mRMR-vel és a regularizációs becslési feladatot megoldó HSIC-Lasso-val szemben, nem is beszélve az egyszerű döntési fáról. Azonban, ha az elemző célja egy takarékos magyarázó modell építése, és a futásidőre nincsenek kimondottan szűk korlátok, akkor érdemes lehet a Hibrid algoritmust alkalmazni például a véletlen erdővel kombinált RFE algoritmussal szemben. Hiszen a Hibrid algoritmus hasonló futásidő és tesztadat teljesítmény mellett egy olyan megoldást tud biztosítani, melyben a magyarázó változók hatása jobban visszafejthető. Természetesen ezen kezdeti eredmények még további vizsgálatokat igényelnek több, a jelen tanulmányban vizsgáltaktól eltérő viselkedésű adatbázisokon is. Ezen túl a Hibrid algoritmus eredményeinek érzékenységvizsgálata is fontos eredményekre vezethet az alkalmazott próbákhoz választott szignifikanciaszint és a concurvity mértékre szabott korlát szigorúságának függvényében.

6 Összegzés

Jelen tanulmányban bemutattunk egy hibrid genetikus – harmóniakereső algoritmust általánosított additív modellek változószelekciós feladatának megoldására.

Munkánkban áttekintettük a GAM-ok alapfogalmait, külön kitérve a magyarázó változók nem-lineáris transzformációját megvalósító függvények reprezentációjára. Bemutattuk, hogy thin plate spline függvények alkalmazásával automatizálható a spline függvények rendjének és a bázisfüggvények szakaszhatárainak megválasztása. Ezzel a változószelekciós feladat komplexitása megőrizhető: továbbra is csak a modellben szerepelő magyarázó változók köréről szükséges döntést hozni. Felhívtuk rá a figyelmet, hogy GAM változószelekció során kerülendő a concurvity jelenség a végső modellben, ha célunk egy jól értelmezhető modell kialakítása, amiben a változók hatásai visszafejthetők. A thin platse spline-ok fogalmainak segítségével bemutattunk egy mértéket, amivel mérhető a concurvity jelenség, azaz megadható, hogy egy magyarázó változó mennyire jól kifejezhető más magyarázó változók nem-lineáris függvényeként.

A tanulmányban ismertettünk két olyan korszerű változószelekciós algoritmust, amelyek célfüggvényükön keresztül törekednek elérni, hogy a végső modellben a concurvity jelenség ne lépjen fel. Ugyanakkor, mindkét algoritmus a concurvity-t csak magyarázóváltozó-páronként vizsgálja. Ezzel figyelmen kívül hagyják az olyan eseteket, amelyekben egy magyarázó változó a modellben szereplő egyéb változók többváltozós függvényeként fejezhető ki.

Bemutattuk, hogyan építhető be a thin plate spline függvények alkalmazása a lineáris változószelekciót már kezelő Hibrid algoritmusba. Megmutattuk, hogy a Hibrid algoritmus képes kezelni közvetlenül a concurvity mértékre vonatkozó korlátokat is a változószelekciós feladatban. Kiemeltük, hogy az algoritmus hatékony működésének kulcsa a kezdeti memória (populáció) jó minőségének biztosítása. Erre két lehetséges megoldást is javasoltunk: a nagy memóriaméret alkalmazását és az algoritmus többszöri futtatását kis maximális generációszám mellett.

A bemutatott algoritmusokat alkalmaztuk két valós adatbázison. Mindkét esetben a Hibrid algoritmus végső modellje a legnagyobb becslési pontossággal rendelkező modell, amely mentes a concurvity jelenségtől. A kisebb méretű, betongerendák adatbázison történő futtatások során vizsgáltuk a Hibrid algoritmus optimális paramétereinek beállítását. A vizsgálat alapján elmondhattuk, hogy az algoritmus hatékonyabb, ha a véletlenszerű szelekciós operátorai dominálnak, de a szabályszerű elemek teljes elhagyása nem kifizetődő. A kezdeti populáció minőségének biztosításához hatékonyabb módszer az algoritmus többszöri futtatása kis maximális generációszám mellett. A nagyobb méretű, banki ügyfelek adatbázison bemutattuk, hogy a memóriában lévő egyedekhez tartozó modellek párhuzamos kiszámításával az algoritmus várható futásideje rövidíthető. Ugyanakkor a Hibrid algoritmus alkalmazásakor nagyobb adatbázis esetében a futásidő így is jelentős, bár hasonló nagyságrendű, mint egy véletlen erdő algoritmus RFE változószelekcióval kombinálva.

Összességében az eredményeink alapján elmondható, hogy ha a cél egy takarékos, magyarázó jellegű modell építése az eredményváltozóra, és a futásidő sem komoly korlát, akkor a Hibrid algoritmus alkalmazása javasolt az egyéb vizsgált változószelekciós algoritmusokkal szemben. Későbbi munkáinkban tervezzük a Hibrid algoritmus eredményeinek érzékenységvizsgálatát az alkalmazott próbákhoz választott szignifikancia-szint és a concurvity mértékre szabott korlát szigorúságának függvényében. Továbbá tervezzük elemezni a Hibrid algoritmus viselkedését a jelen tanulmányban vizsgált adatbázisokhoz képest eltérő viselkedésű, új adathalmazokon is.

Irodalom

1. Augustin, N. H., Sauleau, E. A., & Wood, S. N. [2012]. On quantile quantile plots for generalized linear models. *Computational Statistics & Data Analysis,* 56(8), 2404–2409.
2. Belitz, C., & Lang, S. [2008]. Simultaneous selection of variables and smoothing parameters in structured additive regression models. *Computational Statistics & Data Analysis,* 53(1), 61–81.
3. Climente-González, H., Azencott, C. A., Kaski, S., & Yamada, M. [2019]. Block HSIC Lasso: model-free biomarker detection for ultra-high dimensional data. *Bioinformatics,* 35(14), i427–i435.
4. De Jay, N., Papillon-Cavanagh, S., Olsen, C., El-Hachem, N., Bontempi, G., & Haibe-Kains, B. [2013]. mRMRe: an R package for parallelized mRMR ensemble feature selection. *Bioinformatics,* 29(18), 2365–2368.
5. Du, M., Liu, N., & Hu, X. [2019]. Techniques for interpretable machine learning. *Communications of the ACM,* 63(1), 68–77.
6. Green, P. J. and Silverman, B. W. [1994] *Nonparametric Regression and Generalized Linear Models.* London: Chapman and Hall.
7. Gretton, A., Bousquet, O., Smola, A., & Schölkopf, B. [2005]. Measuring statistical dependence with Hilbert-Schmidt norms. In *International conference on algorithmic learning theory,* Springer, Berlin, Heidelberg, 63–77.
8. Hall, M. A. [1999]. Correlation-based feature selection for machine learning. Doktori értekezés. University of Waikato.
9. Hastie, T. J. and Tibshirani, R. J. [1990] *Generalized Additive Models.* London: Chapman and Hall.
10. Hazewinkel, M. [2001]. Spline interpolation. Encyclopedia of Mathematics, 1.
11. Hunyadi, L., & Vita, L. [2006]. *Statisztika közgazdászoknak.* Központi Statisztikai Hivatal, Budapest.
12. Huo, X., & Ni, X. [2007]. When do stepwise algorithms meet subset selection criteria?. *The Annals of Statistics,* 870–887.
13. James, G., Witten, D., Hastie, T., & Tibshirani, R. [2013]. *An introduction to statistical learning: with applications in R.* New York: Springer.
14. Jolliffe, I. T. [1982]. A note on the use of principal components in regression. *Journal of the Royal Statistical Society: Series C (Applied Statistics),* 31(3), 300–303.
15. Kovács, L. [2019]. Applications of Metaheuristics in Insurance. *Society and Economy,* 41(3), 371–395.

16. Kovács, P. [2008]. A multikollinearitás vizsgálata lineáris regressziós modellekben. *Statisztikai Szemle* 86(1), 38–67.

17. Kuhn., M., Wing, J., Weston, S., Williams, A., Keefer, C., Engelhardt, A., Cooper, T., Mayer, Z., Kenkel, B., the R Core Team, Benesty, M., Lescarbeau, R., Ziem, A., Scrucca, L., Tang, Y., Candan, C., & Tyler Hunt. [2019]. caret: Classification and Regression Training. R package version 6.0-84. https://CRAN.R-project.org/package=caret

18. Láng, B., Kovács, L., & Mohácsi, L. [2017]. Linear Regression Model Selection Using a Hybrid Genetic – Improved Harmony Search Parallelized Algorithm. *SEFBIS Journal* 11(1), 2–9.

19. Marra, G., & Wood, S. N. [2011]. Practical variable selection for generalized additive models. *Computational Statistics & Data Analysis,* 55(7), 2372–2387.

20. McFadden, D. [1974]. Conditional logit analysis of qualitative choice behaviour, in: P. Zarembka (ed.), *Frontiers in Econometrics,* Academic Press, New York, 105–142.

21. Molnar, C. [2020]. Interpretable machine learning. Leanpub.

22. Muhit, I. B. [2013]. Dosage limit determination of superplasticizing admixture and effect evaluation on properties of concrete. *International Journal of Scientific & Engineering Research,* 4(3), 1–4.

23. Peng, H., Long, F., & Ding, C. [2005]. Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis & Machine Intelligence,* 8, 1226–1238.

24. Plank, J., Schroefl, C., Gruber, M., Lesti, M., & Sieber, R. [2009]. Effectiveness of polycarboxylate superplasticizers in ultra-high strength concrete: the importance of PCE compatibility with silica fume. *Journal of Advanced Concrete Technology,* 7(1), 5–12.

25. Schmid, M., & Hothorn, T. [2008]. Boosting additive models using component-wise P-splines. *Computational Statistics & Data Analysis,* 53(2), 298–311.

26. Schumaker, L. L. [2015]. *Spline functions: computational methods.* Society for Industrial and Applied Mathematics.

27. Song, L., Smola, A., Gretton, A., Bedo, J., & Borgwardt, K. [2012]. Feature selection via dependence maximization. *Journal of Machine Learning Research,* 13(1), 1393–1434.

28. Therneau, T., & Atkinson, B. [2018]. rpart: Recursive Partitioning and Regression Trees. R package version 4.1-13. https://CRAN.R-project.org/ package=rpart

29. Wahba, G. [1990]. *Spline models for observational data.* CBMS–NSF Regl Conf. Ser. Appl. Math., 59.

30. Weston, S. [2019a]. foreach: Provides Foreach Looping Construct. R package version 1.4.7. https://CRAN.R-project.org/package=foreach

31. Weston, S. [2019b]. doParallel: Foreach Parallel Adaptor for the 'parallel' Package. R package version 1.0.15. https://CRAN.R-project.org/package=doParallel

32. Wood, S. N. [2003]. Thin plate regression splines. *Journal of the Royal Statistical Society: Series B (Statistical Methodology),* 65(1), 95–114.

33. Wood, S. N. [2011]. Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology),* 73(1), 3–36.

34. Wood, S. N., Goude, Y., & Shaw S. [2015] Generalized additive models for large datasets. *Journal of the Royal Statistical Society, Series C,* 64(1): 139–155.

35. Wood S. N. [2017] *Generalized Additive Models: An Introduction with R* (2nd edition). Chapman and Hall/CRC Press.

36. Wooldridge, J. M. [2016]. *Introductory econometrics: A modern approach.* Nelson Education.

37. Yamada, M., Tang, J., Lugo-Martinez, J., Hodzic, E., Shrestha, R., Saha, A., & Radivojac, P. [2018]. Ultra high-dimensional nonlinear feature selection for big biological data. *IEEE Transactions on Knowledge and Data Engineering,* 30(7), 1352–1365.

38. Yeh, I. C. [1998]. Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete Research,* 28(12), 1797–1808.

39. Yeh, I. C., & Lien, C. H. [2009]. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications,* 36(2), 2473–2480.

40. Zhou, S., & Shen, X. [2001]. Spatially adaptive regression splines and accurate knot selection schemes. *Journal of the American Statistical Association,* 96(453), 247–259.

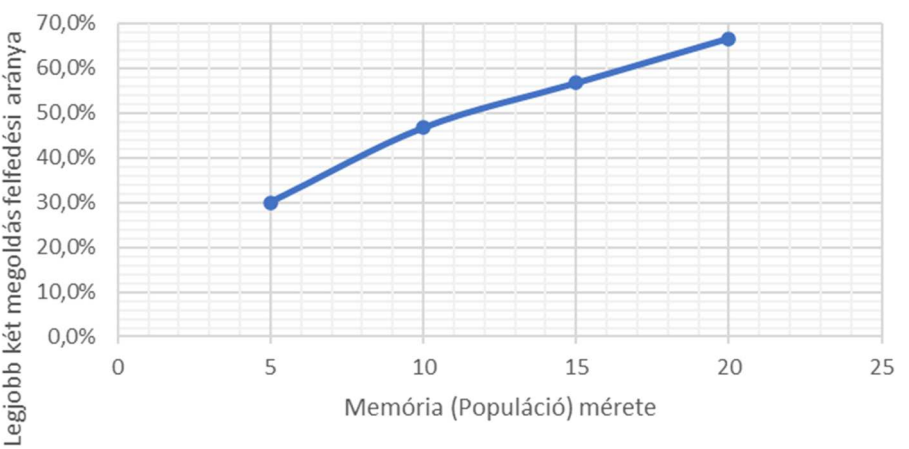
Mellékletek

Az 5.1. fejezetben bemutatott feltételek mellett a Hibrid algoritmusra elvégzett ceteris paribus elvű paraméteroptimalizálás részletes eredményeit a betongerendák adatbázison jelen mellékletben ismertetjük.

1. A Memória (Populáció) méret keresése

Fontos megjegyezni, hogy a memóriaméret esetében a lehető legnagyobb érték az optimális. 20 fölé nem emeltük az értéket, mivel akkor az algoritmusban már 5-nél kevesebb lépés kellett volna a megállási kritérium kielégítéséhez, így az egyedek öröklődését szabályozó szelekciós operátorok lényegében semmilyen szerepet nem kaptak volna a futtatás során.

Változatlanul hagyott paraméterek: induló $HMCR = 0{,}2$; maximális $HMCR = 0{,}6$; induló mutációs valószínűség $= 0{,}9$; minimális mutációs valószínűség $= 0{,}2$; A korai kilépés változatlan legjobb célfüggvényre vonatkozó feltétele $= 5$.

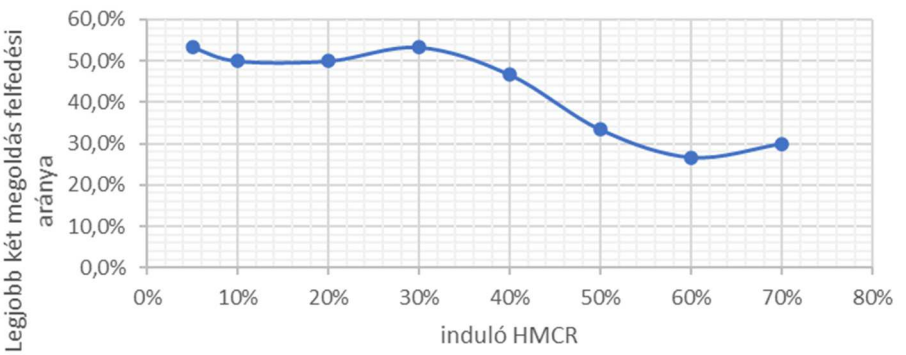


3. ábra. A Hibrid algoritmus hatékonysága a populációméret függvényében. *Forrás:* saját szerkesztés.

2. Az induló $HMCR$ valószínűség keresése

Mivel a $HMCR$ valószínűséget az algoritmus futása során folyamatosan növeljük, így a két azonos hatékonyságú megoldás (5% és 30%) közül a kisebbet tekintjük optimálisnak. Hasonló okból kifolyólag az induló $HMCR$ valószínűség értékek maximuma 70%-ban lett megállapítva.

Változatlanul hagyott paraméterek: Memória mérete = 15; maximális $HMCR$ = 0,9; induló mutációs valószínűség = 0,9; minimális mutációs valószínűség = 0,2; A korai kilépés változatlan legjobb célfüggvényre vonatkozó feltétele = 5.

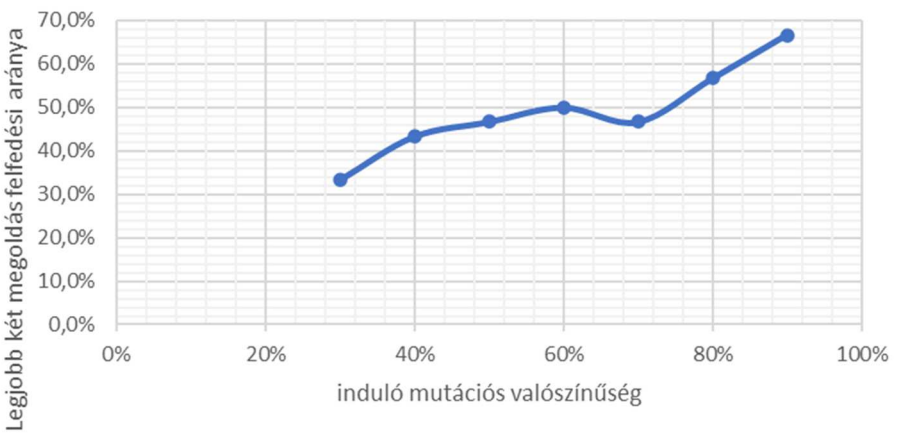


4. ábra. A Hibrid algoritmus hatékonysága az induló $HMCR$ valószínűség függvényében. *Forrás:* saját szerkesztés.

3. Az induló mutációs (bw) valószínűség keresése

Mivel az algoritmus futás során a mutációs valószínűség folyamatosan csökken, így az induló mutációs valószínűség értékek minimuma 30%-ban lett megállapítva.

Változatlanul hagyott paraméterek: Memória mérete = 15; induló $HMCR$ = 0,2; maximális $HMCR$ = 0,6; minimális mutációs valószínűség = 0,1; A korai kilépés változatlan legjobb célfüggvényre vonatkozó feltétele = 5.

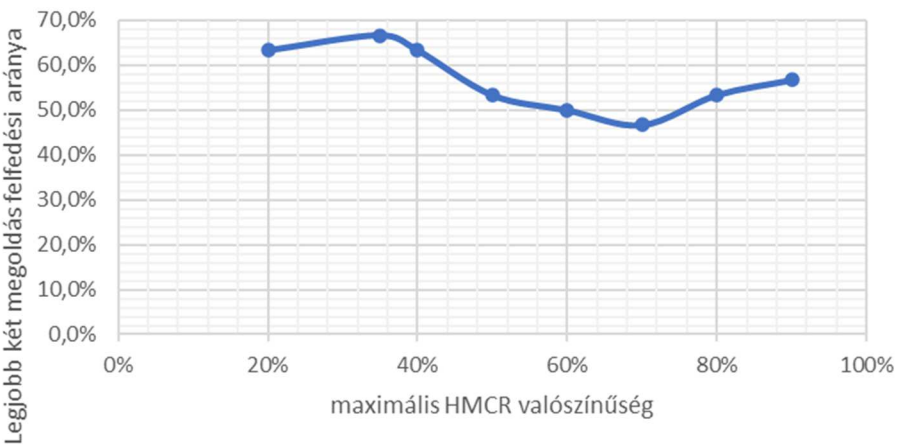


5. ábra. A Hibrid algoritmus hatékonysága az induló mutációs valószínűség függvényében. *Forrás:* saját szerkesztés.

4. A maximális $HMCR$ valószínűség keresése

Mivel a $HMCR$ valószínűséget az algoritmus futása során folyamatosan növeljük, így a maximális generációszám esetén érvényes értékét nem engedjük 20% alá csökkenni.

Változatlanul hagyott paraméterek: Memória mérete = 15; induló $HMCR$ = 0,1; induló mutációs valószínűség = 0,9; minimális mutációs valószínűség = 0,2; A korai kilépés változatlan legjobb célfüggvényre vonatkozó feltétele = 5.

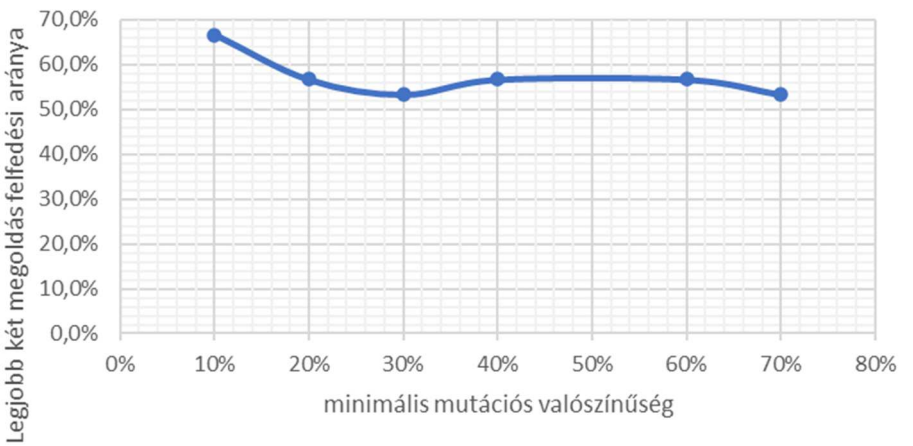


6. ábra. A Hibrid algoritmus hatékonysága a maximális $HMCR$ valószínűség függvényében. *Forrás:* saját szerkesztés.

5. A minimális mutációs (bw) valószínűség keresése

Mivel az algoritmus futás során a mutációs valószínűség folyamatosan csökken, így a maximális generációszám esetén érvényes értékét nem engedjük 70% fölé emelkedni.

Változatlanul hagyott paraméterek: Memória mérete = 15; induló $HMCR$ = 0,2; maximális $HMCR$ = 0,6; induló mutációs valószínűség = 0,9; A korai kilépés változatlan legjobb célfüggvényre vonatkozó feltétele = 5.



7. ábra. A Hibrid algoritmus hatékonysága a minimális mutációs valószínűség függvényében. *Forrás:* saját szerkesztés.

6. A globális optimum vagy a második legjobb megoldás megtalálásához szükséges lépésszámok eloszlása

A 3. táblázatban adott optimális paraméter értékek mellett a globális optimum vagy a második legjobb megoldás megtalálásához szükséges lépésszámok eloszlását mutatja az *5. táblázat*. A kiemelt gyakoriságértékek rávilágítanak, hogy az optimális paraméterek alkalmazása mellett a futtatások valamivel több mint harmadában (12/30 esetben) úgy találja meg a Hibrid algoritmus a globális optimumot vagy a második legjobb megoldást, hogy ez a megoldás már az algoritmus 5. iterációja előtt a memóriába került.

	Végső megoldás megjelenésének iterációja									
Hányadik legjobb megoldás?	2	3	4	5	6	7	8	9	10	Végösszeg
8					1					1
7									1	1
6	1									1
5			1						1	2
4				2			1			3
3				1				1		2
2	*2*	*1*	*1*	*1*		1		1		7
1	*3*		*1*	*3*	2	1	2		1	13
Végösszeg	6	1	3	7	3	2	3	2	3	30

5. táblázat. A megoldások megtalálásához szükséges lépésszámok eloszlása

FEATURE SELECTION IN GENERALIZED ADDITIVE MODELS WITH METAHEURISTICS

In supervised machine learning our aim is to predict a well-defined target variable as accurately as possible by utilizing the known values of several feature variables. Nowadays many complex algorithms are available to solve this task. On the other hand, algorithms that provide the most accurate estimates of the target variable are usually poor at determining marginal effects of the feature variables to the target. However, in certain practical applications, the most important result of supervised learning is not necessarily the accurate estimation of the target, but the discovery of each feature's marginal effect. For example, a bank has to offer a clear reasoning when declining a credit application. In our current big data environment, when the number of possible features is large, determining marginal effects can be challenging even for a linear regression model. One tool that can be utilized to make supervised learning models more interpretable is feature selection.

In this paper we examine the performance of the most recent feature selection algorithms that can be utilized in the context of Generalized Additive Models (GAMs). We chose the model framework of GAMs since they represent a balance between model interpretability and prediction accuracy. In GAMs, marginal effect of the features can be determined, and we are not bound by pre-defined linear, logarithmic, or any other closed functional forms when representing the non-linear effect of features. However, the model does assume an additive structure, so we should apply features that are uncorrelated in a non-linear sense.

Several feature selection algorithms can be applied in a GAM framework. However, the algorithms based on the Hilbert-Schmidt Independence Criterion (HSIC) aim to avoid selecting redundant features. The most recent examples are the mRMR

and the block HSIC Lasso. On the other hand, these algorithms only examine pairwise independence of the features, so they cannot tackle a case where one feature can be accurately estimated by the combination of several other features. Therefore, we propose a hybrid genetic-improved harmony search algorithm (HGHS for short) that applies thin plate splines to produce a best subset feature selector that is capable to find parsimonious models. In the HGHS, the classical feature selection problem is extended with an extra constraint that ensures that redundancy between the selected features is avoided. Numerical examples for the constrained feature selection problem are shown on two real world datasets. The first dataset contains 9 variables of 1030 concrete girders. The task is to estimate the comprehensive strength of concrete material as a non-linear function of age and ingredients. In the second dataset, the task is to estimate for clients in a Taiwanese bank if they are to report default on their credit card loans in one month from now. This dataset consists of 30000 observations and 26 possible features. The performance of the HA is compared with the mRMR and the HSIC Lasso on these datasets. On the second, larger dataset, CART Decision Tree and Recursive Feature Elimination (RFE) combined with a Random Forest learner is also applied as a benchmark algorithm that is not based on GAM learners. All the R scripts of the numerical experiments, the training and test datasets in Rda format, and the tables containing detailed numerical results in csv and xlsx formats are available on the https://github.com/KoLa992/Hybrid-algorithm-for-GAMs repository. Every R script was run in R version 3.5.3., on a 64-bit Windows 10 operating system.

Based on the numerical results, the HGHS can tackle the constrained feature selection problem more effectively than the mRMR and HSIC-Lasso. The HA is the only one of the examined algorithms that can propose models with feature sets that are completely free of redundancy on both examined real-world datasets. This result cannot be matched by the mRMR and the HSIC-Lasso since they only address the redundancy between variable pairs. In case of the Concrete Comprehensive Strength Dataset, the mRMR algorithm proposes a concurvity-free feature subset, but the prediction accuracy of the proposed model is smaller than that of the model proposed by the HGHS.

The performance of the HGHS metaheuristic is greatly affected by the balance between random and controlled selection operators. Results show that when applying the HGHS in a GAM framework, it is preferred to generate completely random new individuals during the iterations, but inheritance from the previous population should not be neglected. In fact, it is also preferred to continuously increase the probability of inheriting an individual from the better-than average individuals of the previous memory/population. Results of a ceteris paribus sensitivity analysis highlight that in case of the optimal parameter set of HGHS, the algorithm can find the global optima or the second-best solution before mapping 40% of the search space in more than third of trials. Based on this observation, we can conclude that the poor quality of initial memory/population should be addressed by running the algorithm from several initial random populations rather than running the algorithm once with large population/memory size. On a large dataset the runtime of the metaheuristic is substantial. However, with parallelization the runtimes can be decreased to the level of Random Forest algorithm combined with Recursive Feature Elimination algorithm.