Kovács László

Változószelekciós algoritmusok vizsgálata általánosított additív modellekben

Számítástudományi Tanszék

Témavezetők:
Dr. Láng Blanka Klára, PhD
Dr. Racskó Péter, CSc

Budapesti Corvinus Egyetem
Közgazdasági és Gazdaságinformatika Doktori Iskola

Változószelekciós algoritmusok vizsgálata általánosított additív modellekben
Egy új, hibrid metaheurisztika elemzése

Doktori értekezés

Kovács László

Budapest, 2021

Tartalomjegyzék

Ábrák jegyzéke

Táblázatok jegyzéke

A dolgozatban használt jelölések jegyzéke

$n \in \mathbb{Z}$	Megfigyelt minta elemszáma
$y_i \in \mathbb{R}$	Az eredményváltozó értéke a minta i-edik elemén
$\hat{y}_i \in \mathbb{R}$	Az eredményváltozóra adott regressziós becslés értéke a minta i-edik elemén
$x_{ji} \in \mathbb{R}$	A j-edik magyarázóváltozó értéke a minta i-edik elemén
$\varepsilon_i \in \mathbb{R}$	A regressziós modell hibatagja a minta i-edik elemén
$Y = [y_1, y_2, \dots, y_n]^T \in \mathbb{R}^n$	Az eredményváltozó vektora
$\hat{Y} = [\hat{y}_1, \hat{y}_2, \dots \hat{y}_n]^T \in \mathbb{R}^n$	A regresszióból becsült eredményváltozó vektora
$X_j = \left[x_{j1}, x_{j2}, \dots, x_{jn}\right]^T \in \mathbb{R}^n$	A j-edik magyarázóváltozó vektora
$\varepsilon = [\varepsilon_1, \varepsilon_2, \dots, \varepsilon_n]^T \in \mathbb{R}^n$	A regressziós modell hibavektora
$h(\cdot) \in \mathbb{R} \to \mathbb{R}$	Az általánosított lineáris/additív modell link függvénye
$E(\xi)$	Egy tetszőleges ξ valószínűségi változó várható értéke
$m \in \mathbb{Z}$	A lehetséges magyarázóváltozók száma
$p \in \mathbb{Z} \leq m$	Egy konkrét regressziós modellben szereplő magyarázóváltozók száma
$X = \{X_1, X_2, \dots, X_m\}$	A lehetséges magyarázóváltozók halmaza
$\tilde{X} = \{X_1, X_2, \dots, X_p\} \subseteq X$	Egy konkrét regressziós modellben szereplő magyarázóváltozók halmaza
$\beta_0 \in \mathbb{R}$	Konstans tag egy általánosított lineáris modellben
$\beta_j \in \mathbb{R}$	A j-edik magyarázóváltozó együtthatója egy általánosított lineáris modellben
$\boldsymbol{X} \in \mathbb{R}^{n \times m}$	Az összes lehetséges magyarázóváltozó oszlopos egymás mellé illesztéséből kapott $n \times m$-es mátrix
$\boldsymbol{X^{(i)}} \in \mathbb{R}^{1 \times m}$	Az $\boldsymbol{X}$ mátrix i-edik sora
$f_j(\cdot) \in \mathbb{R} \to \mathbb{R}$	A j-edik magyarázóváltozó transzformációs függvénye egy általánosított additív modellben
$\ell(\cdot)$	Egy általánosított lineáris/additív modellt a teljes mintán az argumentumban adott modellparaméterek mellett jellemző negatív feltételes log-likelihood függvény
$Bs(\cdot)$	Egy tetszőleges becslőfüggvény torzításának mértéke.
$SE(\cdot)$	Egy tetszőleges becslőfüggvény standard hibája.
$x \in \mathbb{R}$	Egy tetszőleges $\mathbb{R} \to \mathbb{R}$ függvény változója
$\mathcal{H}$	Reprodukáló magú Hilbert-tér (továbbiakban RMHT)
$r \in \mathbb{Z}$	Egy b-spline függvény rendje
$B_{i,r}(\cdot) \in \mathbb{R} \to \mathbb{R}$	Egy r-ed rendű b-spline függvény i-edik bázisfüggvénye
$k_i \in \mathbb{R}$	Egy b-spline függvény i-edik töréspontja a $[\min(x), \max(x)]$ intervallumon
$S_r(\cdot) \in \mathbb{R} \to \mathbb{R}$	Egy r-ed rendű b-spline függvény
$\alpha_i \in \mathbb{R}$	Egy r-ed renű b-spline függvény i-edik bázisfüggvényének együtthatója $S_r(\cdot)$-ben
$l\left(y_i, S_3\left(x_{ji}\right)\right)$	A minta i-edik elemén az eredményváltozó feltételes log-likelihood függvénye, ismert $E(y_i) = h^{-1}\left(S_3\left(x_{ji}\right)\right)$feltétel mellett
$\boldsymbol{\alpha}^{(j)} \in \mathbb{R}^r$	Egy a j-edik magyarázóváltozóra illesztett r-ed renű b-spline függvény együtthatóvektora
$f(\cdot), g(\cdot), \eta(\cdot) \in \mathcal{H}$	Tetszőleges függvények egy RMHT-ből
K, L: $\mathbb{R}^{n \times n} \to \mathbb{R} \in \mathcal{H}$	Egy RMHT pozitív definit magfüggvényei

$\lambda \in \mathbb{R}$	A thin plate spline illesztési feladatban a büntetőtag súlyát szabályozó paraméter
$\psi_i \in \mathbb{R}$	A thin plate spline feladat megoldásában szereplő bázisfüggvény együtthatója az i-edik illesztési pontban
$\Psi = [\psi_1, \psi_2, \dots, \psi_n] \in \mathbb{R}^n$	A thin plate spline feladat megoldásában szereplő bázisfüggvény együtthatóinak vektora
$\boldsymbol{E} = \{E_{ab} \coloneqq \eta_j(\|x_{ja} - x_{jb}\|)\} \in \mathbb{R}^{n \times n}$	A thin plate spline feladat megoldásában szereplő bázisfüggvény helyettesítési értékeiből felépített mátrix a j-edik magyarázóváltozóra
$\boldsymbol{E_k} = \boldsymbol{U_k D_k U_k^T} \in \mathbb{R}^{k \times k}$	Az $\boldsymbol{E}$ mátrix $k \in \mathbb{Z} \leq n$ legnagyobb sajátértéke melletti spektrálfelbontásból előálló $k \times k$ méretű mátrix
$\Psi_k = [\psi_1, \psi_2, \dots, \psi_k] \in \mathbb{R}^k$	Az $\boldsymbol{E_k}$-val felírt thin plate spline feladat megoldásában szereplő bázisfüggvény együtthatóinak vektora
$k_j \in \mathbb{Z}$	A j-edik magyarázóváltozóhoz meghatározott, optimális k dimenzió.
$\boldsymbol{A} \in \mathbb{R}^{n \times \sum_{j=1}^{p} k_j}$	Olyan mátrix, melynek oszlopaiban az egyes X_j magyarázóváltozók nem-lineáris f_j transzformációinak bázisfüggvényit reprezentáló $\boldsymbol{U_{k_j} D_{k_j}}$ mátrixok állnak
$\boldsymbol{S_j} \in \mathbb{R}^{\sum_{j=1}^{p} k_j \times \sum_{j=1}^{p} k_j}$	Olyan, a j-edik magyarázóváltozóhoz tartozó mátrix, melyben a főátló megfelelő k_j elemű szakaszán a megfelelő $\boldsymbol{D_{k_j}}$ diagonális mátrixok elemei állnak. Egyébként minden eleme 0.
$\widetilde{\Psi} = \left[\Psi_{k_1}, \Psi_{k_2}, \dots, \Psi_{k_p}\right] \in \mathbb{R}^{\sum_{j=1}^{p} k_j}$	A teljes GAM modell együtthatóvektora thin plate spline-ok alkalmazása mellett.
$\Psi_{k_j} \in \mathbb{R}^{k_j}$	$\widetilde{\Psi}$-nek a j-edik magyarázóváltozóhoz tartozó, k_j hosszú alvektora.
$p(\xi, \eta)$	A ξ és η valószínűségi változók együttes eloszlása
$p(\xi)$	A ξ valószínűségi változó egy peremeloszlása
$I(\xi, \eta)$	A ξ és η valószínűségi változók közös információtartalmának általános mértéke
$E_{\xi,\xi',\eta,\eta'}$	A $p(\xi, \eta)$ együttes eloszlásból vett független (ξ, η) és (ξ', η') párok várható értéke
$\boldsymbol{K_j} \in \mathbb{R}^{n \times n}$	A j-edik magyarázóváltozó Gram mátrixa
$\boldsymbol{L} \in \mathbb{R}^{n \times n}$	Az eredményváltozó Gram mátrixa

Köszönetnyilvánítás

Az értekezés elején szeretném kifejezni köszönetemet és hálámat azoknak a személyeknek, akik közvetve-közvetlenül segítséget vagy támogatást nyújtottak a disszertációm elkészítésében és a kutatói pályakezdés során.

Először szeretném kifejezni köszönetemet a két témavezetőmnek, Láng Blankának és Racskó Péternek. Láng Blankával BSc szakos hallgatóként kezdtük el a közös munkát, és ő vezetett be a metaheurisztikák területébe, így az értekezés során bemutatott algoritmus fejlesztésében végig támogatott, és folyamatos tanácsaival, észrevételeivel mindig tökéletesítette a munkám. Ezen kívül folyamatos támogatást nyújtott a kutatói pálya számtalan más területén is. Racskó Péter gyors és alapos útmutatásaira mindig számíthattam a dolgozat matematikai és gépi tanulással kapcsolatos részeiben. Továbbá, a tudományos publikációk műfajában való eligazodáshoz kaptam tőle felbecsülhetetlen értékű támogatást.

Hálás köszönettel tartozom szüleimnek, akik annak ellenére is folyamatosan támogattak és biztattak a disszertáció elkészítése során, hogy néhány alkalommal biztosan nagy kihívás volt elviselni a munkámmal járó kellemetlenségeket.

Szintén hálás vagyok minden oktatómnak és kollégámnak a Budapesti Corvinus Egyetem Informatika és Matematikai és Statisztikai Modellezés Intézeteiben, akik kellemes hangulatú szakmai közösséget teremtettek, amelyben öröm volt dolgozni. Külön kiemelném Balázsné Mócsai Andreát, akinek példamutató pedagógusi munkája nélkül biztosan nem fordultam volna ilyen lelkesedéssel a statisztika területe felé. Köszönöm Ágoston Kolosnak, Keresztély Tibornak, Mohácsi Lászlónak és Vékás Péternek, hogy a munkám különböző fázisaiban értékes észrevételeikkel segítették a kutatásom. Mindig élvezetes volt együtt dolgozni Burka Dáviddal, Csóka Imolával, Fodor Szabinával, Kovács Erzsébettel, Kő Andreával, Madari Zoltánnal és Vas Rékával is.

Köszönettel tartozom minden volt és jelenlegi hallgatómnak és szakdolgozómnak is, akik komoly szorgalmukkal, lelkiismeretes és pozitív hozzáállásukkal folyamatosan ösztönöztek arra, hogy kutatómunkám eredményeit és szakmai tapasztalataim próbáljam részben az oktatói tevékenységembe is beépíteni. A lelkes hallgatóság tanítása közben pedig magam is értékes felfedezéseket tettem a kutatott témámmal kapcsolatban.

Végül, de nem utolsó sorban barátaimnak szeretném megköszönni mindazt a türelmet, támogatás és biztatást, amit az utóbbi években nyújtottak. Külön kiemelném Nyisztor Nelli

folyamatos támogatását és türelmét. A közös beszélgetéseink többször is új megvilágításba helyeztek számomra egy-egy kérdést, problémát. Kiemelten hálás vagyok Bors Alexandrának is, aki mindig gyorsan a segítségemre sietett, ha szöveg- és ábraszerkesztési támogatásra vagy éppen egy átolvasásra szorultam. Köszönöm Barnóth Balázsnak, Csikai Leventének, Fehér Józsefnek, Nguyen Dorisnak, Sánta Sándornak, Sass Zoltánnak és Vig Ádámnak, hogy a munkám informatikai és matematikai kérdéseinek egy részét velük is megvitathattam. Hosszú éveken át kitartóan elviselt és támogatott kutatásaim során Fekete Edit, Havasi Fatime, Lampert Alex, Okulova Nyina, Pencz Kriszta, Vig Árpád és Zugor Valentina.

A felsoroltokon kívül még rengeteg családtagom, barátom és kollégám támogatását élveztem a disszertáció elkészítése során, így az itt szereplő lista biztosan nem teljeskörű. Ha valakit esetleg kifelejtettem volna a felsorolásból, akkor mentségemre egyedül az hozható fel, hogy ezt a súlyos hibát minden bizonnyal figyelmetlenségből, és semmiképpen sem rossz szándékkal követtem el.

1. Bevezetés

Különböző valószínűségi változók várható értékének becslése, előrejelzése más egyéb valószínűségi változók realizációinak ismeretében a statisztika egy nagyon régóta aktuális feladata. A legelső megoldást a feladatra Carl Friedrich Gauss adta 1821-ben *Theoria combinationis observationum erroribus minimis obnoxiae* címmel megjelent tanulmányában (Gauss, 1821 – G. W. Stewart 1995-ös angol fordítása) arra az esetre, ha az eredményváltozó (az a valószínűségi változó, aminek a várható értékét becsülni szeretnénk) normális eloszlású. A Gauss által bemutatott becslési eljárás ma legkisebb négyzetek módszere néven ismert, és a lineáris regressziószámítás alapja. Ebben az esetben az eredményváltozó feltételes várható értékét az ismert magyarázóváltozók lineáris kombinációjaként becsüljük meg. A Gauss-féle legkisebb négyzetek módszere megadja a lineáris kombinációban szereplő skalárok értékét úgy, hogy a becslés során elkövetett négyzetes hiba a „legkisebb", tehát minimális legyen. Gauss eredményeit 1900-ban Andrej Andrejevics Markov általánosította arra az esetre, amikor már csak azt követeljük meg a lineáris regressziós modelltől, hogy a hibavektor korrelálatlan, nulla várható értékű és azonos varianciájú valószínűségi változókból álljon (Plackett, 1949).

A lineáris regresszió és a legkisebb négyzetek segítségével végrehajtott becslések, előrejelzések a mai napig népszerűek, melynek az egyik oka, hogy a legkisebb négyzetek minimalizálási feladat megoldására létezik zárt formula. A XX. században a számítástechnika gyors fejlődésének köszönhetően a lineáris regressziós modell kiterjesztésre került olyan esetekre is, amikor zárt megoldás már nem adható, és a lineáris modell skalár paramétereinek becslésére iteratív numerikus módszerek alkalmazása szükséges. A legfontosabb ilyen eset a Bernoulli-eloszlású eredményváltozó esete, amire a Chester Bliss és John Gaddum nevéhez köthető, 1934-ben bemutatott probit modell (Gaddum, 1933) (Bliss, 1934) és a Joseph Berkson által 1944-ben javasolt logit modell (Berkson, 1944) ad megoldást. 1972-ben John Nelder és Robert Wedderburn egységes keretbe foglalták az exponenciális eloszláscsaládba tartozó eredményváltozók várható értékét különböző magyarázóváltozók lineáris kombinációjával történő becslését megvalósító modelleket általánosított lineáris modellek néven (Nelder – Wedderburn, 1972). A klasszikus legkisebb négyzetek elvű lineáris regresszió, valamint a probit és logit modellek is az általánosított lineáris modellek speciális esetei.

A XX. század második felében a számítástechnika gyors fejlődése nem csak a lineáris modellek kiterjesztését tette lehetővé különböző eloszlású eredményváltozók esetére, hanem nem-lineáris modellek alkalmazását is az eredményváltozó feltételes várható értékének becslésére. Ezzel gyakorlatilag a felügyelt gépi tanulás területe született meg az időszakban. Az első működő

többrétegű neurális hálózatot 1965-ben mutatta be Alexey Ivakhnenko (Ivakhnenko – Lapa, 1967). A neurális hálózatok tanításához elengedhetetlen hiba-visszaterjesztéssel (backpropagation of errors) kombinált gradiensereszkedés algoritmust 1974-ben publikálta Paul Werbos (Werbos, 1994). A támaszvektor-gépek már 1963-ban megjelentek Vladimir N. Vapnik és Alexey Ya. Chervonenkis munkájában (Cortes – Vapnik, 1995). A nyolcvanas évek során a legtöbb döntési fa algoritmus is kifejlesztésre került: CHAID (Kass, 1980), CART (Breiman et al., 1984) és az ID3 (Quinlan, 1986). 1990-ben Trevor Hastie és Robert Tibshirani továbbfejlesztette az általánosított lineáris modellek nem-lineáris hatások spline függvényekkel történő kezelésével. Ezzel bevezették az általánosított additív modellek fogalmát (Hastie – Tibshirani, 1990).

Ugyanakkor, a számítástechnika fejlődése a klasszikus regressziós modelleket is új kihívások elé állította. Az adattárolás hatékonyságának javulása egyre nagyobb adatbázisok létrejöttét eredményezte, aminek következtében a modellezés során elérhető magyarázóváltozók száma megnövekedett. A jelenség egyenes következménye, hogy a magyarázóváltozók körét már nem biztos, hogy szimplán szakértői szempontok alapján ki lehet választani a sok elérhető változó közül, mivel a lehetséges részhalmazok száma emberi módon kezelhetetlen méreteket ölthet. Emiatt a XX. század második felében elkezdtek fejlődni az automatikus változószelekciót megvalósító algoritmusok, először a lineáris regressziós modellek részeként. 1960-ban jelentek meg a lokális kereső Stepwise algoritmusok, amiket M. A. Efroymson javasolt (Efroymson, 1960). 1974-ben George Furnival és Robert Wilson mutatta be a leaps and bounds algoritmust, ami azt elsősorban vegyes egészértékű programozási feladatok megoldásában népszerű branch and bound stratégiát alkalmazza a változószelekciós feladat megoldására legjobb részhalmaz elven (Furnival – Wilson, 1974). Numerikus vizsgálatok alapján viszont a módszer csak 20 db lehetséges magyarázóváltozóig hatékony (Huo – Ni, 2007). Az először a lineáris modellekben megjelenő változószelekciós módszereket később a nem-lineáris keretben működő gépi tanuló algoritmusok is alkalmazni kezdték, hiszen az eredményváltozóval legszorosabb kapcsolatban álló magyarázóváltozók azonosítása ezekben a modellekben is fontos feladat az értelmezhetőség biztosítása és a túlillesztés elkerülése miatt.

A változószelekciós algoritmusok területén a következő nagy áttörés Robert Tibshirani 1996-os munkája jelentette, amiben a bemutatott Lasso algoritmus segítségével a lineáris modellek regularizált paraméterbecslésével valósít meg változószelekciót (Tibshirani, 1996). A 2000-es évek során a Lasso algoritmushoz hasonló regularizációs elven működő változószelekciós algoritmusok fejlesztése és tulajdonságainak elemzése egy nagyon népszerű kutatási iránnyá vált. A legfontosabb munkák: (Efron, 2004) (Tibshirani et al., 2005) (Zou – Hastie, 2005) (Yuan

– Lin, 2006) (Zhao – Yu, 2006) (Xie – Huang, 2009) (Jia – Yu, 2010). A regularizációs elv Reed és MarksII 1999-es munkájában megjelent a neurális hálózatok tanítása kapcsán is, mint egy védekezési technika a túlillesztés ellen (Reed – MarksII, 1999) és napjaink mélytanuló hálózataiban is gyakran alkalmazott módszer. Egy jó példa Ma et al. (2019).

A regularizációs becslések mellett a változószelekciós algoritmusok másik nagy csoportja szintén a 2000-es években vált népszerűvé: a metaheurisztikát alkalmazó legjobb részhalmaz elvű algoritmusok. A metaheurisztikák nagy előnye a legjobb részhalmaz elvű szelekció során a leaps and bounds algoritmussal szemben, hogy 20-nál több lehetséges magyarázóváltozó esetén is szolgáltatnak legalább egy „optimumtól elfogadható mértékben eltérő" megoldást (Yusta, 2009). Emiatt napjaink „big data" környezetében, amikor a lehetséges magyarázóváltozók köre a legtöbb esetben a 20-at bőven meghaladja, a metaheurisztikus megoldások kimondottan vonzónak bizonyulnak. Julie Hammon 2013-as doktori disszertációja alapján a 2000-es években a legtöbb gyakorlati alkalmazás a genetikus algoritmust implementálta változószelekcióra (Hammon, 2013).

Jelen doktori értekezésben egy új, hibrid genetikus-harmónia kereső metaheurisztikus algoritmus viselkedését vizsgáljuk meg általánosított additív modellek változószelekciós feladatának megoldására. Az értekezés harmadik fejezetében megfogalmazott K1-K5. kutatási kérdések elsősorban az algoritmus általánosított additív modellek körében történő alkalmazásának technikai lehetőségeit, az algoritmus által javasolt modellek minőségét, valamint az algoritmus várható futásidejének csökkentési lehetőségeit vizsgálják.

Az algoritmus kidolgozása során a fő motivációt az adta, hogy a változószelekció egyik legfontosabb feladata az értelmezhető gépi tanulás megvalósítása. Az értelmezhető gépi tanulás paradigmája szerint bizonyos gyakorlati szituációkban a gépi tanulás legfontosabb eredménye nem feltétlenül a minél pontosabb becslés elkészítése, hanem a becslés szempontjából legfontosabb magyarázóváltozók szűk körének azonosítása és az egyes magyarázóváltozók hatásának megállapítása. Például, egy banknak egyértelműen meg kell indokolnia, hogy mi alapján utasít el egy hitelkérelmet. Továbbá, ahhoz, hogy ne legyen a szabályrendszere a fogyasztó számára átláthatatlan, ehhez nem használhat tetszőlegesen sok változót. Ilyen esetekben nem előrejelző, hanem magyarázó modellek építése az elemző célja.

Az ismertetett motivációk figyelembevételével történt meg a doktori értekezésben vizsgált felügyelt gépi tanulási módszertan és változószelekciós paradigma kiválasztása is. Az általánosított additív modellek ugyanis az eredményváltozó várható értékét nem-lineáris módon becsülik, ám az egyes magyarázóváltozók hatásai még visszafejthetők. Ezzel egyensúlyt képviselnek a becslési pontosság és az értelmezhetőség között. A metaheurisztikus

algoritmusok egyrészt képesek legjobb részhalmaz elvű változószelekciót megvalósítani sok lehetséges magyarázóváltozó mellett is. Másrészt, annyira rugalmasan alkalmazhatók, hogy könnyen beépíthetők új korlátok a változószelekciós feladatba, amelyek segítségével biztosíthatjuk, hogy az általánosított additív modellek magyarázóváltozókkal szemben megkövetelt előfeltételei teljesüljenek a kiválasztott változóhalmazon.

A jelenlegi első fejezet történeti áttekintését és vázlatos témafelvetését követően az értekezés második fejezetében ismertetjük az értelmezhető gépi tanulás paradigmájának főbb koncepcióit.

A harmadik fejezetben korábbi munkáinkra támaszkodva bemutatjuk, hogy az értekezésben vizsgált genetikus-harmónia kereső algoritmus milyen eszközökkel igyekszik biztosítani az értelmezhető gépi tanulás megvalósítását lineáris modellek esetében. Megfogalmazásra kerülnek az értekezés konkrét kutatási céljai és legfontosabb elért eredményei, melyek a vizsgált algoritmus kiterjesztési lehetőségeit járják körbe általánosított additív modellekre.

A negyedik fejezetben formálisan is ismertetjük az általánosított lineáris modellek fogalmait és megadjuk a változószelekciós feladatot lineáris esetben. Erre a fejezetre építve fogjuk felvezetni az ötödik fejezetben az értekezés során részletesen vizsgált általánosított additív modellek fogalmait. Az ötödik fejezet további célja az additív modellek fogalmainak formális ismertetése mellett, hogy bemutassa milyen eszközöket kell alkalmazni a nem-lineáris hatások reprezentációja során, hogy a változószelekciós feladatot ne kelljen új döntési változókkal bővíteni additív modellek esetében.

A hatodik fejezetben ismertetjük a szakirodalom által javasolt és napjainkban népszerű változószelekciós algoritmusokat általánosított additív modellek esetében. Rámutatunk, hogy egyik vizsgált algoritmus sem képes a lehetséges magyarázóváltozók közötti esetleges redundanciákat maradéktalanul figyelembe venni a változószelekció során.

A hetedik fejezet során részletesen bemutatjuk az értekezésben vizsgált új, hibrid genetikus-harmónia kereső algoritmus működését lineáris és additív modellek keretében is. Megmutatjuk, hogy az algoritmusba épített extra korlátozó feltételek biztosítják, hogy a magyarázóváltozók közötti esetleges redundanciákat maradéktalanul kezelni tudjuk a változószelekció során. Végül kitérünk arra is, hogy az algoritmus paraméterezése során milyen elveket érdemes alkalmazni.

A nyolcadik fejezetben áttekintjük az értekezésben megfogalmazott kutatási eredmények alátámasztásául szolgáló numerikus kísérletek keretrendszerét. Továbbá, bemutatjuk az egyes modellek becslési pontosságát mérő teljesítménymetrikákat folytonos, illetve Bernoulli-eloszlású célváltozó esetében.

Az kilencedik és tizedik fejezetben a vizsgált algoritmusok működését összehasonlítjuk két valós adatbázison. A kilencedik fejezetben a feladat betongerendák nyomószilárdságának becslése. Ebben a fejezetben a genetikus-harmónia kereső algoritmus paramétereinek finomhangolására egy kisebb méretű adatbázist használunk. A tizedik fejezetben szereplő 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. A két fejezetben megmutatjuk, hogy a vizsgált metaheurisztikus algoritmus végső modelljeiből jobban azonosíthatók az eredményváltozóra ható, egymással nem szignifikánsan összefüggő magyarázóváltozók, mint a többi vizsgált algoritmus esetében. Az algoritmus futásideje viszont párhuzamosítás után is nagyságrendekkel nagyobb, mint a benchmarkként használt algoritmusoké.

A tizenegyedik fejezetben a genetikus-harmónia kereső algoritmus skálázhatóságát vizsgáljuk több, virtuális környezetben futtatott hardver architektúrán. A numerikus eredményeinket az Amdahl-törvény alapján számított maximális gyorsítás mértékéhez viszonyítjuk. Rámutatunk, hogy a numerikusan mért gyorsítás mértékének eltérése az elméleti maximumtól a hibrid genetikus-harmónia kereső algoritmus véletlen részeinek preferálása miatt adódik az optimális paraméterezésben.

A tizenkettedik fejezetben alaposabban kitérünk a kihagyott változó okozta torzítás jelenségére, mint a hibrid genetikus-harmónia kereső algoritmus legfontosabb korlátjára, valamint ismertetjük a lehetséges továbbfejlesztési irányokat.

Végezetül a tizenharmadik fejezetben összefoglaljuk a kutatás fő eredményeit, és tételesen is megválaszoljuk a második fejezetben ismertetett kutatási kérdéseket. A tapasztalatok alapján minősítjük az értekezésben vizsgált hibrid genetikus-harmónia kereső algoritmus gyakorlati alkalmazási lehetőségeit.

Az értekezés részét képezi a dolgozatban vizsgált és továbbfejlesztett hibrid genetikus-harmónia kereső metaheurisztikus algoritmus 332 soros forráskódja R nyelven, valamint az értekezés 9., 10. és 11. fejezeteiben szereplő numerikus eredményeket előállító, összesen 1176 soros R szkript. Az R szkriptek, a vizsgált algoritmusok futtatásához használt, *Rda* formátumú tanító és teszt adatbázisok, továbbá a numerikus eredményeket részleteiben tartalmazó *csv* és *xlsx* formátumú táblázatok a https://github.com/KoLa992/Hybrid-algorithm-for-GAMs tárhelyről érhetők el. Minden R nyelven írt szkript az R 3.5.3-as verziójában került futtatásra, 64 bites Windows 10 operációs rendszeren. A teszteléshez használt hardver

konfiguráció egy Intel Core i7-8750H 2,20 GHz processzorral (12 mag) és 8 GB 2666 MHz DDR4 RAM-mal rendelkező személyi számítógép.

2. Az értelmezhető felügyelt gépi tanulás kérdésköre

A fejezetben először egy intuitív példát mutatunk be James et al. (2013) alapján a felügyelt gépi tanulás motivációjának és alapfogalmainak ismertetésére.

Tegyük fel, hogy egy tanácsadó céget képviselünk, aminek aktuális feladata egy ügyfél konkrét terméke eladási bevételeinek növelése. Az ügyfél egy adatbázisban vezeti, hogy mennyi bevétele volt az elmúlt évben a szóban forgó termékből 200 különböző piacon. Ugyanezen a 200 piacon az ügyfél rögzíti a termék reklámkiadásait is három különböző platformon: TV, rádió, újság. Az ügyfél direkt módon nem feltétlenül képes a bevételeit növelni a termékből. Másrészről, viszont a három médiumra vonatkozó reklámkiadásait tudja szabályozni. Tehát, ha képesek vagyunk megállapítani, hogy milyen kapcsolat áll fenn a reklámkiadások és a termékből származó bevétel között, akkor módosítási javaslatokat tudunk tenni az ügyfél reklámköltségvetésében, ezzel indirekt módon növelve a bevételeket. Egy fokkal pontosabban fogalmazva azt mondhatjuk, hogy célunk egy olyan modell fejlesztése, amely a lehető legpontosabban meg tudja becsülni a vizsgált termékből származó bevételt a termék reklámköltségvetésének ismeretében.

A fenti példában statisztikai fogalmakkal élve azt mondhatjuk, hogy az eredményváltozó a termékből származó bevétel a különböző piacokon, a három magyarázóváltozónk pedig ugyan ezeken a piacokon a termék reklámkiadásai a TV, rádió és újság platformokon. Statisztikában jellemzően az eredményváltozót Y-al jelöljük, míg a magyarázóváltozókat X-el: Jellemzően a magyarázóváltozókat egy alsó indexszel különböztetjük meg jelöléseinkben. Tehát, jelen példánkban X_1 a TV reklám kiadások értéke a vizsgált piacokon, X_2 a rádióreklámokra fordított kiadás összege és X_3 az újsághirdetésekre szánt költség értéke.

Általánosabban vizsgálva a kérdést azt mondjuk, hogy megfigyelünk több független egyedre egy Y eredményváltozó és m darab magyarázóváltozó $X_1, X_2, \dots, X_m$ értékét. Továbbá feltételezzük, hogy létezik valamilyen kapcsolat Y és $X = (X_1, X_2, \dots, X_m)$ között, amely általánosan a következő formában írható fel: $Y = f(X) + \varepsilon$.

A képletben f egy rögzített, de ismeretlen többváltozós függvénye $X_1, X_2, \dots, X_m$-nek, és ε egy véletlen hibatag, amely X-től független és ε elemeinek átlaga 0. Ebben a formalizmusban f jelöli azt a szisztematikus információt, amelyet X elárul Y-ról. Lényegében a felügyelt gépi tanulás azon módszerek összességének tekinthető, amelyek használatával f pontos alakja megbecsülhető (Russell – Norvig, 2010).

A gyakorlati alkalmazások során f pontos alakjának ismeretére két okból lehet szükségünk: előrejelzésre és értelmezésre (James et al., 2013). A következő két alfejezetben mindkét okot röviden bemutatjuk.

2.1. Az f becslése előrejelzés céljából

Több gyakorlati alkalmazásban Y mérése nehéz, ám a magyarázóváltozók egy X halmazának értékei könnyen megfigyelhetők. Ekkor a fő célunk Y becslése $\hat{Y} = \hat{f}(X)$ módon, hiszen az ε hibatag átlagára 0-t feltételeztünk. Ebben a formulában $\hat{f}$ jelöli a becslésünk f-re, míg $\hat{Y}$ az $\hat{f}$-ből származó becsléseink (előrejelzéseink) értékét jelöli Y valós értékeire. Ebben az esetben $\hat{f}$ gyakran fekete dobozként kezelendő olyan értelemben, hogy nem vagyunk kíváncsiak $\hat{f}$ pontos alakjára, ha az pontos előrejelzéseket ad Y-ra.

Példaként vehetünk egy olyan esetet, amikor $X_1, X_2, \dots, X_m$ egy páciens vérmintáinak különböző jellemzőit jelölik és Y pedig a páciens válaszreakcióját jelöli egy adott gyógyszerre. Egy ilyen szituációban természetes, hogy Y-t szeretnénk minél pontosabban megbecsülni $X_1, X_2, \dots, X_m$ értékeinek függvényében. Hiszen, ezzel elkerülhető, hogy olyan pácienseknek adjuk be a szóban forgó gyógyszert, akiknél a gyógyszer káros mellékhatásokat okozhat.

$\hat{Y}$ pontossága, mint egy Y-ra adott előrejelzés, két tényezőtől függ, amelyeket csökkenthető, és nem-csökkenthető hibáknak nevez a szakirodalom (Russell – Norvig, 2010). Általánosságban elmondhatjuk, hogy $\hat{f}$ sosem lesz tökéletes becslése f-nek, és ennek a „tökéletlenségnek" a mértékét hívjuk csökkenthető hibának. Ez a hiba azért csökkenthető, mivel egyre bonyolultabb és bonyolultabb $\hat{f}$ függvényeket alkalmazva $\hat{f}$ egyre közelebb kerül a valós f-hez. Ugyanakkor, még ha sikerülne is f-et tökéletesen megbecsülni, és az előrejelzésünk Y-ra az $\hat{Y} = f(X)$ mennyiség lenne, a $\hat{Y}$ becslésünk még akkor is tartalmaz hibát! Hiszen, az eredeti modellünkben Y az ε függvénye is, ami a modell definícióból adódóan nem csökkenthető az X magyarázóváltozók használatával. Tehát, ε a nem-csökkenthető hiba.

A gyakorlatban a nem-csökkenthető hiba jelenlétének leggyakoribb oka, hogy ε olyan változók hatását tartalmazza Y-ra, amiket nem tudunk mérni, így nem szerepelnek az X-ek között. A korábbi példánknál maradva azt mondhatjuk, hogy egy páciens kockázata arra nézve, hogy egy gyógyszer káros mellékhatásokat vált ki nála, eltérő lehet különböző napokon, vagy a gyógyszer különböző gyártási technikákkal készült verziónak függvényében és a páciens vérmintáinak segítségével nem mérhető egyéb kockázati tényezők miatt.

A felügyelt gépi tanulási technikák közös jellemzője, hogy f-et a csökkenthető $f(X) - \hat{f}(X)$ hiba várható értékének minimalizálásával becsülik meg (Russell – Norvig, 2010).

Egy további szemléletes példa lehet a kizárólag előrejelzés céljából elvégzett f becslésre egy direkt marketing kampány. Ekkor a vállalat célja, hogy megtalálja azokat a potenciális ügyfeleket, akik pozitívan reagálnak különböző direkt marketing (e-mail, telefon, személyes) megkeresésre, különböző demográfiai változók ismeretében. Egy ilyen projekt esetében a vállalat nem feltétlenül érdekelt a demográfiai változók (magyarázóváltozók) és a marketing kampányra adott reakció (eredményváltozó) közötti kapcsolat pontos megismerésére. A marketing kampányért felelős szakember elsődleges célja egy minél pontosabb lista előállítása azokról az ügyfelekről, akik magas valószínűséggel pozitívan reagálnak a direkt marketing megkeresésre.

2.2. Az f becslése értelmezés céljából

A gyakorlati alkalmazásokban gyakran érdekelheti az elemzőt, hogy Y milyen módon reagál az $X_1, X_2, \dots, X_m$ értékek változására. Ebben a szituációban szintén f becslése a célunk, de nem feltétlenül azért, hogy előrejelzéseket adjunk Y-ra. Ehelyett Y és X kapcsolatát próbáljuk feltérképezni és megérteni. Konkrétan, arra vagyunk kíváncsiak, hogyan változik Y az $X_1, X_2, \dots, X_m$ értékeinek függvényeként. Ebben az esetben $\hat{f}$-ot nem kezelhetjük fekete dobozként, hiszen a célunk éppen $\hat{f}$ pontos alakjának megismerése. Ilyen jellegű gyakorlati példák esetében általában az alábbi három kérdés valamelyikét kívánja az elemző megválaszolni (James et al., 2013).

- *Mely magyarázóváltozók alakítják az eredményváltozó értékét?* Gyakorlati példákban sokszor előfordul, hogy az elérhető m db magyarázóváltozóknak csak egy kis részhalmaza van jelentős hatással Y alakulására. Egy kis elemszámú, de nagy hatású változóhalmaz azonosítása sok lehetséges magyarázóváltozó közül nagyban segíti Y alakulásának megértését több esetben is.
- *Milyen irányú kapcsolatban áll az eredményváltozó és egy konkrét magyarázóváltozó?* Bizonyos magyarázóváltozók egyirányú, mások ellentétes irányú kapcsolatban állnak az eredményváltozóval. Jól megválasztott $\hat{f}$ segítségével az eltérő irányú hatással bíró változók azonosíthatók. Egyirányú kapcsolata alatt azt értjük, hogy ha egy tetszőleges X_j változó értéke nő, akkor erre jellemzően Y is növekedéssel reagál. Ellentétes irányú kapcsolat esetén X_j növekedésére jellemzően Y csökkenéssel reagál. Bizonyos bonyolultabb $\hat{f}$ függvények esetén megengedett, hogy egy magyarázóváltozó hatása más magyarázóváltozók szintjétől vagy a saját szintjétől is függjön.
- *Lineárisnak tekinthető-e a kapcsolat az eredményváltozó és az egyes magyarázóváltozók között?* Történeti okokból az f-re adott becslési módszerek lineáris

függvényformát feltételeznek elsősorban az egyéb becslési módszerek számítási korlátai miatt. Bizonyos szituációkban a lineáris forma helytálló és kívánatos is. Ugyanakkor fontos felismerni, hogy a lineáris forma feltételezése mikor sérül meg látványosan, és mikor tudjuk kizárólag nem-lineáris alakban megadni a magyarázóváltozók és az eredményváltozók pontos kapcsolatát.

Ha a fenti három kérdést szeretnénk megválaszolni, fontos, hogy $\hat{f}$ alakja egyszerűen megadható, könnyen értelmezhető legyen. Amikor erre törekszünk, akkor úgynevezett értelmezhető gépi tanulási technikákat alkalmazunk f becslésére. Napjaink „big data" környezetében, amikor egy adott becslési feladathoz rengeteg potenciális magyarázóváltozó könnyen az elemző rendelkezésére áll, kiemelten fontossá válik, hogy az első felsorolt kérdésre válaszolni tudjunk, és kiszűrjük lehetséges magyarázóváltozóink közül azt a részhalmazt, amiknek a legjelentősebb hatása van az eredményváltozóra. Emiatt az értelmezhető gépi tanulási módszerekkel kapcsolatos átfogó szakirodalom az utóbbi években egyre bővül. A legjobb példák Molnar (2020) és Du et al. (2019) munkái.

A fejezet bevezető példája kapcsán a reklámköltségek és a termékből származó bevétel kapcsolatának vizsgálata során több olyan kérdést is feltehetünk, aminek megválaszolásához az értelmezhető gépi tanulás eszköztárát szükséges alkalmazni.

- Melyik platform (TV, rádió, újság) reklámköltségei hatnak a termék bevétel adataira?
- Melyik platformon elköltött egységnyi reklámköltség okozza a legnagyobb bevételnövekedést?
- Egy adott összegű TV reklám megvásárlása várhatóan mennyi bevételt fog generálni a reklámozott termékből?

Továbbá, az értelmezhető gépi tanulás módszerei lehetnek szükségesek olyan klasszikus közgazdasági kérdések megválaszolására is, mint egy keresleti függvény becslése. Legyen az eredményváltozó az, hogy egy konkrét ügyfél megvásárol-e egy adott terméket. Magyarázóváltozóként pedig használjunk olyan tényezőket, mint a termék ára, a bolt helye, a termékre adható kedvezmények mértéke, a konkurens termék ára stb. Ebben az esetben az elemzőt elsősorban az érdekli, hogy az egyes magyarázóváltozóként használt tényezők hogyan hatnak a termék megvásárlásának valószínűségére. Például, ha az egyéb tényezők nem változnak, akkor hogyan hat a termék árának változása annak megvásárlási valószínűségére?

Könnyen el tudunk képzelni olyan példát is, amikor a feladat egyaránt kívánja előrejelző és magyarázó modellek építését is. Például legyen a feladatunk lakásárak becslése olyan különböző környezeti magyarázóváltozók segítségével, mint a környék bűnözési rátája,

folyóparti-e a lakás, levegőminőség, környező iskolák minősége, környék lakóinak átlagos jövedelme, lakás területe stb. Egy ilyen helyzetben érdekes lehet az egyes magyarázóváltozók árakra gyakorolt hatásának becslése, pl. várhatóan mennyivel növeli egy lakás árát, ha az a folyóparton van. Ugyanakkor azt is kimondottan fontos lehet megállapítani, hogy egy konkrét lakás alul- vagy felülárazott-e. Ez utóbbi esetben a lakásárak minél pontosabb becslése az elemző célja.

Attól függően, hogy az elemző célja pusztán előrejelzés, értelemzés vagy a kettő kombinációja, különböző módszereket alkalmazhatunk f becslésére. A következő alfejezetben egy rövid áttekintést adunk különböző felügyelt gépi tanulást megvalósító algoritmusokról olyan szempontból is, hogy inkább előrejelzési vagy értelmezési feladatokra alkalmasak.

2.3. Néhány tanuló algoritmus elemzése becslési pontosság és értelmezhetőség szempontjából

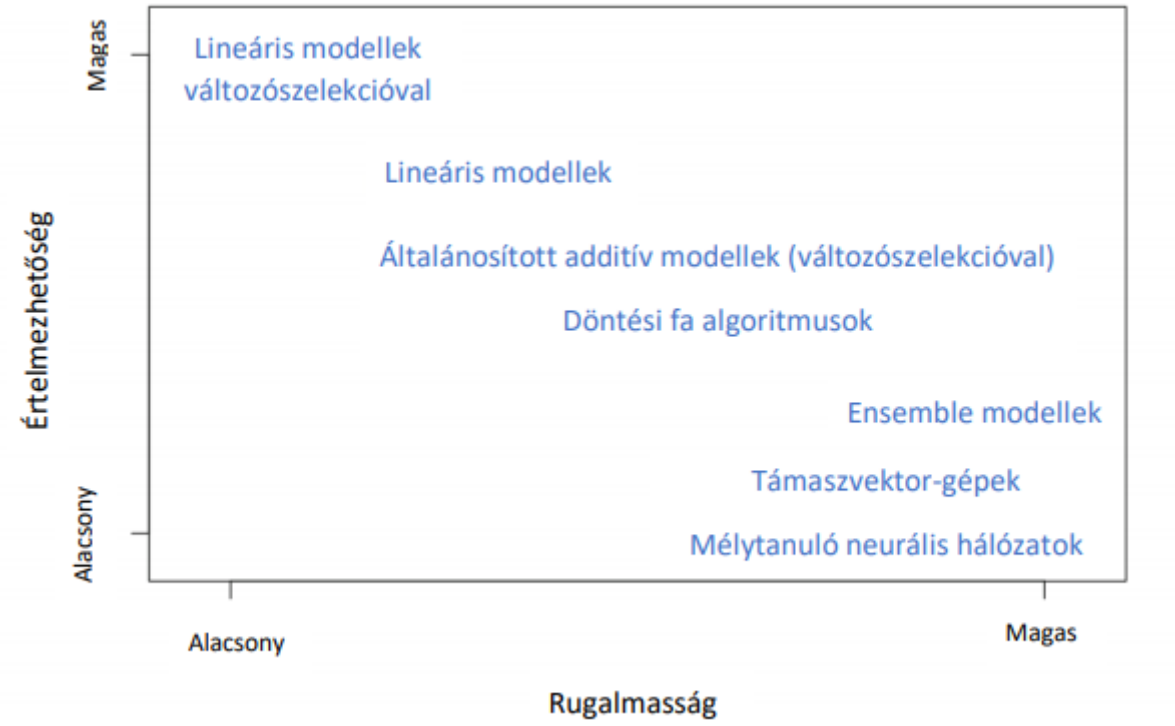
A felügyelt gépi tanulás módszerei közül bizonyosak elég rugalmatlannak, korlátosnak tekinthetők, mert f becslését elég szűken meghatározott függvények körében oldják meg. Nyilván egy lineáris regresszió (vagy az általánosított lineáris modellek) elég rugalmatlan módszernek tekinthető, hiszen $m + 1$ db változó kapcsolatát csak egy m dimenziós hipersík segítségével tudja leírni és a hipersík paramétereit becslő legkisebb négyzetek módszere a magyarázóváltozók függetlenségét teszi fel.

Amennyiben az eredményváltozó a magyarázóváltozók többváltozós, nem-lineáris függvénye, akkor a lineáris függvényformával történő $\hat{f}$ közelítés elég magas csökkenthető hibával rendelkező $\hat{Y}$ becsléseket eredményez. Ugyanakkor, megvan az az előnye, hogy egy tetszőleges X_j magyarázóváltozó marginális hatása Y-ra nagyon könnyen értelmezhetővé válik, hiszen lineáris esetben a $\frac{\partial \hat{f}(X)}{\partial X_j}$ parciális derivált konstans értéket vesz fel. Amennyiben változószelekciót is végrehajtunk akár regularizációs, pl. Lasso módszerrel (Tibshirani, 1996), akár valamilyen legjobb részhalmaz elvű szelekcióval (pl. leaps and bounds (Furnival – Wilson, 1974) vagy genetikus algoritmussal (Calcagno, 2019), akkor ugyan még jobban korlátozott modellt kapunk eredményül (hiszen a lehetséges magyarázóváltozók közül nem használunk fel mindent $\hat{f}$-ben), ám azonosításra kerül az Y alakulását legjobban meghatározó magyarázóváltozók részhalmaza. Ezzel az eredményváltozóra ható legfontosabb tényezők azonosításra kerülnek, és segítségükkel könnyebben lehet a modellek által azonosított összefüggéseket értelmezni és kommunikálni szélesebb közönség felé. Ugyanakkor, nem feltétlenül ezektől a szelektált lineáris modellektől várjuk a legjobb becslési pontosságot Y-ra.

Az általánosított additív modellek (Generalized Additive Model, röviden GAM) és a döntési fák kiterjesztik a lineáris függvényformát bizonyos értelemben. A GAM-ok esetében $\hat{f}$ az egyes magyarázóváltozókhoz tartozó nem-lineáris $\hat{f}_1, \hat{f}_2, \dots, \hat{f}_m$ függvények összegeként áll elő (Hastie – Tibshirani, 1990), míg a döntési fa módszerek elsősorban az egyes magyarázóváltozók közti interakciós hatások figyelembevételével lazítanak a lineáris modell megkötésein (Therneau – Atkinson, 2018). Jellemzően mindkét idézett módszer a lineárisnál pontosabb becslésekhez vezet Y-ra, ám az egyes változók marginális hatásai már nem adhatók meg konstansként. Ugyanakkor, az egyes változók marginális hatásai még jól vizualizálhatók a döntési fa hierarchikus gráfja és a GAM-ok különböző $\hat{f}_j$ függvényeinek koordinátarendszerben adott ábrái segítségével. Mindkét módszer értelmezhetősége javítható változószelekció alkalmazásával. A döntési fákat építő algoritmusok a modell becslése során eleve 0 súlyt rendelnek a nem releváns magyarázóváltozókhoz, míg a GAM-ok esetében külön algoritmusok valósítják meg a változószelekciót a lineáris modellekhez hasonlóan (James et al., 2013).

A teljesen rugalmas, nem-lineáris modellek (ensemble módszerek, mélytanuló neurális hálózatok, támaszvektor-gépek stb.) esetében a magyarázóváltozók marginális hatásai már gyakorlatilag egyáltalán nem visszafejthetők (Hastie et al., 2011). Bizonyos esetekben legfeljebb a magyarázóváltozók fontossági sorrendjének megállapítására van lehetőség. Pl. a véletlen erdő nevű ensemble módszer esetében. Ugyanakkor, fontos megjegyezni, hogy ezek a modellek a magyarázóváltozók fontosságát több kritérium alapján is meg tudják adni, pl. Gini-index vagy kereszt-entrópia alapján (Fawagreh et al., 2014). A különböző kritériumok alapján történő változórangsorok rangkorrelációinak vizsgálata további kutatások témája lehet. Azonban, a jelenség vizsgálata jelen értekezésnek nem tárgya.

A fejezetben említett módszerek közti viszony rugalmasság és értelmezhetőség szempontjából az 1. ábrán látható.



1. ábra: Különböző gépi tanuló algoritmusok értelmezhetősége rugalmasságuk függvényében. Általános tendencia, hogy a rugalmasabb algoritmusok kevésbé értelmezhetők.
Forrás: James et al., 2013, p.25. alapján saját szerkesztés.

3. A kutatási kérdések ismertetése és a kutatási módszertan áttekintése

Jelen fejezetben a doktori értekezésben végzett saját kutatásom öt megválaszolandó kérdését mutatom be. A saját munka kiemelése céljából ebben a fejezetben egyes szám első személyt alkalmazok, míg az összes többi fejezetben a tudományos publikációkban megszokottabb többes szám első személyt alkalmazom. Az értekezés kutatási kérdéseinek ismertetése után bemutatom a kutatás kivitelezése során alkalmazott Design Science módszertant Hevner et al. (2004) hét pontja és Wieringa (2014) alapján. Továbbá, részletezem, hogy az értekezésben bemutatott saját kutatásom hogyan felel meg a Hevner et al. (2004)-féle ajánlásoknak.

Az értekezésben bemutatott kutatás legfontosabb előzményeinek két társzerzős publikációm tekinthető: Láng – Kovács (2014) és Láng et al. (2017). Az idézett két tanulmányban gazdaságinformatikus szakos BSc-hallgatóként működtem közre. A publikációk BSc-hallgatóként önállóan készített korábbi TDK dolgozat és szakdolgozat továbbfejlesztése.

Ebben a két tanulmányban társszerzőimmel lineáris modellek keretében vizsgáljuk a különböző regularizációs és legjobb részhalmaz elvű változószelekciós algoritmusok hatékonyságát elsősorban a javasolt modelljeik értelmezhetőségének szempontjából. Az idézett tanulmányokban bemutatunk egy hibrid genetikus-harmónia kereső algoritmust (a továbbiakban röviden HGHK algoritmus) a változószelekciós feladat megoldására. Az algoritmus legjobb részhalmaz elvű szelekciót valósít meg. Legfontosabb újdonsága a szakirodalomban szereplő hasonló megoldásokhoz képest, hogy új korlátozó feltételek segítségével igyekszik a modellben szereplő magyarázóváltozók redundanciáját meggátolni, azaz függetlenségüket – amennyire lehetséges – biztosítani. Lineáris esetben ez a modellben szereplő magyarázóváltozók korrelálatlanságát jelenti, ami a lineáris modellek gyakorlati alkalmazásának egyik legfontosabb feltevése. Az algoritmus fejlesztése során saját hozzájárulásom a harmónia kereső és genetikus algoritmus ötvözésének ötlete volt. Az ötlet mögött az a felismerés állt, hogy a változószelekciós feladatban szükség van az egyedek nagyobb fokú véletlenségét biztosító rekombinációs operátorokra, amiket a harmónia kereső algoritmus tud biztosítani, ám futásidő szempontjából szükséges a genetikus algoritmus párhuzamos egyedkezelésének megőrzése is. Az algoritmus implementációját C# nyelven társszerzőimmel közösen végeztük. Az algoritmus első működő implementációja önálló munka eredménye, amely a gazdaságinformatikus BSc szakdolgozatom része. Társszerzőimmel a kód optimalizálásán működtünk közösen közre a két idézett publikáció elkészítése során.

Az idézett tanulmányokban megmutatjuk, hogy az algoritmus alkalmazásával elérhető a modellbe válogatott magyarázóváltozók megfelelő mértékű tapasztalati korrelálatlansága, ami nagyban segíti a magyarázóváltozók marginális hatásának értelmezhetőségét. Hiszen a $\frac{\partial \hat{f}(X)}{\partial X_j}$ parciális derivált, mint marginális hatás vizsgálata során feltesszük, hogy X_j változása esetén a modellben szereplő összes többi magyarázóváltozó értéke változatlan marad. Továbbá, a magyarázóváltozók közti korrelálatlanság sérülése hatással van a lineáris modellek paraméterbecslésének stabilitására is (Hastie et al., 2011).

Jelen értekezés közvetlen előzményének tekinthető az aktuárius MSc-hallgatóként végzett saját kutatásom alapján készült Kovács (2019) önálló publikációm. Ebben a tanulmányban a HGHK algoritmust kiterjesztettem a klasszikus lineáris modellekről a Cox-féle arányos hazárd modellekben végzett változószelekcióra is. A kiterjesztett algoritmussal életbiztosítási szerződések lemorzsolódási görbéit befolyásoló magyarázóváltozók körét határozom meg. A lineáris modellek esetéhez hasonlóan az arányos hazárd modellek esetében is érvényesült a HGHK azon tulajdonsága, hogy a modellkeretben alkalmazott egyéb változószelekciós algoritmusokhoz képest hasonló becslési pontosságú megoldást szolgáltat, ám ezt korrelálatlan magyarázóváltozóhalmaz segítségével éri el. Ennek köszönhetően egy életbiztosítási szerződésállományban előforduló lemorzsolódási okok könnyebben feltárhatóak voltak a HGHK segítségével, mint más változószelekciós algoritmus által szolgáltatott modelleket felhasználva. A kutatás legfontosabb hozadéka jelen értekezés szempontjából, hogy ekkor készítettem el a HGHK algoritmus R nyelvű implementációjának első változatát. Ez az implementáció a közvetlen kiindulása a jelen értekezéshez tartozó R nyelvű programkódoknak, melyek szintén teljes mértékben saját munka eredményei.

Jelen doktori értekezésben megvizsgálom, hogy a HGHK algoritmus milyen módon terjeszthető ki a 2. fejezetben röviden már bemutatott GAM-ok körébe. A GAM-ok esetében a magyarázóváltozók függetlensége fontos feltétel, hiszen a modellben a $\hat{f}_1, \hat{f}_2, \dots, \hat{f}_m$ függvények additív módon kapcsolódnak egymáshoz. Ezzel a HGHK algoritmus korlátja, ami arra törekszik, hogy a modellben csak független (vagy legalábbis a függetlenség mérésére alkalmazott metrika szerint nem redundáns) magyarázóváltozók szerepeljenek, továbbra is hasznos lehet, ha az elemző célja egy GAM modellel inkább értelmező modell építése, és az eredményváltozóra nem-lineárisan ható legfontosabb tényezők azonosítása és marginális hatásaik megadása. Továbbá, a GAM-ok alkalmazásának előnye az 1. ábra alapján, hogy az értelmezhetőség megőrzése mellett pontosabb becsléseket szolgáltatnak Y-ra, mint a lineáris modellek.

3.1. Az értekezés kutatási kérdései

Az ismertetett kutatási előzmények alapján jelen értekezésben a K1-K5. kutatási kérdésekre keresem a választ a HGHK algoritmus GAM-ok körére kiterjesztett verziójával kapcsolatban. A K1-K5. kutatási kérdések elsősorban az algoritmus GAM keretre történő kiterjesztésének technikai lehetőségeit, az algoritmus által javasolt modellek minőségét, valamint az algoritmus várható futásidejének csökkentési lehetőségeit célozzák.

K1. Megvalósítható-e a HGHK algoritmus kiterjesztése lineáris modellekről GAM keretbe anélkül, hogy az algoritmusban lineáris esetben alkalmazott döntési változókon és bináris egyedreprezentáción változtatni kellene?

K2. Képes-e valós adatbázisokon a HGHK algoritmus (a magyarázóváltozók redundancia mentességét biztosítani kívánó korlát alkalmazása segítségével) olyan modelleket azonosítani, amelyek kevesebb és nem redundáns magyarázóváltozó segítségével nagyságrendileg hasonló becslési pontosságot nyújtanak, mint az értekezés során vizsgált egyéb korszerű változószelekciós algoritmusok GAM keretben?

K3. A HGHK algoritmusban az új populáció létrehozása során a véletlen vagy a kontrollált rekombinációs operátorokat érdemesebb előnyben részesíteni GAM keretben történő alkalmazás esetén?

K4. Milyen eszközzel érdemes a HGHK algoritmusban a kezdeti populáció jó minőségét biztosítani: több véletlen kezdőpopulációból lefuttatni a HGHK algoritmust kis populáció méret és alacsony maximális generációszám mellett, vagy inkább nagy populáció méretet és magas maximális generációszámot célszerű alkalmazni és az algoritmust csak egyszer lefuttatni?

K5. Milyen párhuzamosítási stratégiával javítható jobban a HGHK algoritmus futásideje: több egyedhez tartozó modell egyszerre történő, párhuzamos kiszámításával, vagy egy modell kiszámításának párhuzamosításával? A kiválasztott párhuzamosítási stratégiával az algoritmus milyen hatékonyan skálázható különböző teljesítményparaméterekkel rendelkező hardver architektúrákon? A párhuzamosítással elért empirikus gyorsítás a HGHK algoritmusban hogyan viszonyul a párhuzamosítás elviekben elérhető maximális gyorsításához?

3.2. A Design Science kutatási módszertan és alkalmazása az értekezés kutatási kérdéseinek megválaszolására

A K1-K5. kutatási kérdések megválaszolásához további kutatásomat a Design Science módszertan Hevner et al. (2004)-féle hét pontos ajánlása alapján tervezem meg.

A Design Science módszertan szerint megvalósított kutatás konkrét eredménytermékek tervezése és teljesítményük kivizsgálása éles környezetben. A kutatás explicit célja az eredménytermék funkcionális teljesítményének fejlesztése. A Design Science módszertan eredménytermékeinek körébe általában algoritmusok, ember/gép interfészek, folyamatmodellek és programnyelvek tartoznak. A módszertan alkalmazása leginkább a mérnöki és számítástudományi területeken elterjedt, ám nem korlátozott kimondottan ezekre a tudományágakra és egyéb területeken is alkalmazható (Kuechler – Vaishnavi, 2008).

A Design Science módszertan alkalmazása egyaránt jelenti informatikai eredménytermék kifejlesztését és kutatási kérdések megválaszolását. Az értekezésben a K1-K5. kutatási kérdéseinek megválaszolását egy saját algoritmus (HGHK) továbbfejlesztésével, implementációjával és annak numerikus hatékonyságvizsgálatával tervezem elérni, ami konkrét informatikai eredményterméknek tekinthető. Ebből adódóan logikus választás a Design Science módszertan a jelen értekezésben végzett kutatás kivitelezéséhez. A módszertan hét általános irányelve Hevner et al. (2004) alapján az alábbi:

1. Tervezés, mint termék: A Design Science módszertan szerint megvalósuló kutatás eredménye egy működő termék, tervezés, modellezés vagy eljárás formájában.
2. Releváns probléma: A kutatás célja, hogy egy technológia alapú megoldást fejlesszen egy fontos és releváns üzleti probléma megoldására.
3. Értékelés: A tervezett termék hasznosságát, minőségét és hatásosságát szigorúan kell értékelni egy megfelelő értékelési eljárás alapján.
4. Kutatási érték: A hatásos Design Science módszertan szerint megvalósuló kutatásnak világos és igazolható eredményeket kell felmutatnia a tervezett termék, a tervezés alapjai és módszerei esetében.
5. Kutatási szigor: A termék kialakításában és értékelésében egyaránt szigorú módszereket kell alkalmazni.
6. Tervezés, mint keresési folyamat: A hatásos eredmény megtalálása érdekében a keresés során szükséges a rendelkezésre álló eszközök olyan használata, mely során a kutatás megfelel a probléma környezetében fennálló törvényszerűségeknek.
7. Kutatás kommunikációja: A kutatás eredményeit hatásosan kell bemutatni a technológiában és a menedzsmentben jártas közönségnek egyaránt.

A jelen értekezésben elvégzett kutatásomban egy működő implementációját fejlesztem ki a HGHK algoritmusnak, ami képes a változószelekciós feladat megoldására GAM-ok esetén, valós adatbázisokon. Ezzel a 7. fejezetben részletesen bemutatott HGHK algoritmus megfelel a Design Science módszertan 1. ajánlásában szereplő termék kritériumainak.

A 4. és 5. fejezetben alaposan ismertetem a kutatásban vizsgált GAM-ok elméleti és módszertani kereteit a legfrissebb szakirodalmi források alapján. Kiemelem, hogy a GAM-ok paraméterbecslési eljárásai érzékenyek a modellben szereplő magyarázóváltozók redundanciájára, a concurvity jelenségre. Bemutatom, hogy a concurvity jelenség a modellek eredményeinek üzleti értelmezhetőségét is megnehezíti. A 6. fejezetben pedig a GAM keretben alkalmazható legkorszerűbb változószelekciós algoritmusok működésének elemzésével rávilágítok arra, hogy a GAM változószelekció során alkalmazott legnépszerűbb algoritmusok a concurvity jelenséget nem, vagy csak korlátozott mértékben veszik figyelembe. A 4., 5. és 6. fejezetekben tehát irodalomkutatás segítségével bemutatom, hogy az értekezésben bemutatott kutatás és annak eredményterméke fontos és releváns üzleti problémára nyújt megoldást a Design Science módszertan 2. ajánlásának megfelelően.

Az értekezés 8. fejezetében áttekintem a HGHK algoritmus teljesítményének kiértékelésére alkalmazott validációs módszerek és teljesítménymetrikák körét. Részletes irodalomkutatással alátámasztom, hogy a HGHK algoritmus teljesítményének numerikus összehasonlítása a 6. fejezetben bemutatott egyéb GAM keretben alkalmazható korszerű változószelekciós algoritmusok teljesítményével korszerű validációs eljárások és teljesítménymetrikák segítségével történik. Ezzel figyelembe véve a Design Science módszertan 3. ajánlását.

A 7. fejezetben bemutatom, hogy az általam javasolt HGHK algoritmus maradéktalanul figyelembe veszi a concurvity jelenséget a változószelekció során. Továbbá, a 9. és 10. fejezetekben két eltérő paraméterekkel rendelkező gyakorlati problémán alkalmazom a HGHK-t és az algoritmus teljesítményét a 6. fejezetben ismertetett eljárásokkal szemben összehasonlítható módon visszaméretem. A numerikus kísérletek megerősítik, hogy a HGHK képes elérni, hogy az eredményül kapott modell concurvity jelenségtől mentes legyen, továbbá a becslési pontossága is nagyságrendileg megközelíti a többi vizsgált algoritmus pontosságát a legtöbb, a 8. fejezetben javasolt metrika alapján. A 10. fejezet numerikus eredményei igazolják a HGHK elfogadható várható futásidejét is az algoritmus memóriájában/populációjában lévő egyedekhez tartozó GAM-ok párhuzamos kiszámítása esetén. Ezen numerikus eredmények alapján igazolom, hogy a kutatásom terméke, a HGHK algoritmus kutatási értéket hordoz, és ezzel kutatásom megfelel a Design Science 4. ajánlásának.

A HGHK algoritmus teljesítményének optimalizálását a 9. fejezetben a paraméterek szigorú ceteris paribus elvű érzékenységvizsgálatával végzem el. Az eredmények függvényében a HGHK algoritmust a 10. fejezetben már új módon, több, kisebb méretű kezdeti populációval alkalmazom a nagyobb méretű változószelekciós feladat hatékony megoldása érdekében. Ezen túl a 11. fejezetben vizsgálom a HGHK skálázhatóságát több párhuzamosított végrehajtásra alkalmaz virtuális hardver architektúra alkalmazásával. A HGHK különböző architektúrákon empirikusan mért teljesítményét mindig visszamérem a párhuzamosítással az adott architektúrán elérhető elvi maximális gyorsításhoz képest. Továbbá, az eredmények kiértékelése során mind a 9., 10. és 11. fejezetben szigorúan követem a 8. fejezetben bemutatott validációs módszertanokat. Mindezekkel biztosítom a megfelelést a Design Science módszertan 5. ajánlásában szereplő kutatási szigornak és az összes elérhető algoritmusfejlesztési eszköz-környezet törvényszerűségei megfelelő alkalmazásának (6. ajánlás).

A kutatás aktív kommunikációja (Design Science 7. ajánlás) során jelen doktori kutatás eredményeit bemutattam több hazai és nemzetközi konferencián is, valamint referált tudományos folyóiratokban is megjelentek publikációim. Továbbá, a kutatási eredmények bemutatásra kerültek a Budapesti Corvinus Egyetem „Adatelemzés a gyakorlatban” előadássorozatának részeként és a 2020 januári Kutatási héten is, ahol az eredményeket az Egyetem hallgatóival és olyan kutatókkal osztottam meg, akik a HGHK algoritmust saját kutatásaik során maguk is alkalmazni tudják.

Az értekezés kutatási céljainak és módszereinek bemutatása után a kutatási eredményeim ismertetését a GAM-ok elméleti és módszertani kereteinek tárgyalásával kezdem a 4. és 5. fejezetekben.

4. Az általánosított lineáris modellek (GLM) formális ismertetése

A fejezetet az általánosított lineáris modellek (továbbiakban GLM az angol Generalized Linear Model kifejezés alapján) tárgyalásával kezdjük Nelder – Wedderburn (1972) alapján. Az egyszerűbb, lineáris keretben ismertetjük a változószelekció alapfeladatát és ismertetjük a multikollinearitás fogalmát.

Az értekezés tárgya a GAM keretben értelmezhető változószelekciós algoritmusok köre, így a 5. fejezetben ezen modellkeret formális ismertetését adjuk meg. Viszont, az általánosított additív modellek az általánosított lineáris modellek közvetlen továbbfejlesztésének tekinthetők, így ezért foglalkozunk először jelen fejezetben a lineáris modellek formális leírásával. Mivel az 5. fejezetben a jelen fejezetben bevezetett fogalmak segítségével tudjuk bemutatni, hogy a nem-lineáris, GAM keretben milyen módosítások szükségesek a változószelekciós feladat megfogalmazásában a lineáris esethez képest.

Legyen adott egy n elemű minta. Legyen $Y = [y_1, y_2, \dots, y_n]^T \in \mathbb{R}^n$ egy exponenciális eloszláscsaládból származó valószínűségi változó megfigyelt értékeit tartalmazó vektor. Ekkor egy GLM segítségével Y várható értéke (1) módon becsülhető p db $X_j = \left[x_{j1}, x_{j2}, \dots, x_{jn}\right]^T \in \mathbb{R}^n$ magyarázóváltozó megfigyelt értékeinek segítségével.

$$h\big(E(Y)\big) = \varepsilon + \beta_0 + \sum_{j=1}^{p} \beta_j X_j \qquad (1)$$

Ahol $h(\cdot) \in \mathbb{R} \to \mathbb{R}$ a GLM link függvénye, $\varepsilon = [\varepsilon_1, \varepsilon_2, \dots, \varepsilon_n]^T \in \mathbb{R}^n$ a modell hibavektora és $\beta = \big(\beta_0, \beta_1, \dots, \beta_p\big)^T \in \mathbb{R}^p$ a modell együtthatóinak (paramétereinek) vektora. Ekkor β vektor maximum likelihood módszerrel megbecsülhető a rendelkezésre álló minta alapján az eredményváltozó eloszlásának figyelembevételével. Legyen $L_i(\beta)$ az i-edik megfigyelt mintaelem sűrűség értéke a feltételezett eredményváltozó eloszlás, a magyarázóváltozók i-edik megfigyelése és a β paramétervektor mellett. Ekkor a β vektort a $\max_{\beta} \sum_{i=1}^{n} \ln L_i(\beta)$ feladat megoldásával nyerjük. A megoldáshoz általában a Newton – Raphson módszert alkalmazunk. (James et al., 2013)

A következő gyakorlati probléma egy m db lehetséges $X = \{X_1, X_2, \dots, X_m\}$ magyarázóváltozót tartalmazó halmazból kiválasztani azt a $p \leq m$ db változót tartalmazó $\tilde{X} = \{X_1, X_2, \dots, X_p\} \subseteq X$ részhalmazt, ami a legjobb általánosító képességű modellt eredményezi. A regressziós modell általánosító képessége azt mutatja meg, hogy a modell mekkora pontossággal tudja az *n* elemű mintán kívüli populációt is jellemezni a minta alapján kinyert információk alapján. A megfelelő általánosító képesség eléréséhez kompromisszumra van szükségünk a mintainformációk

felhasználásának tekintetében. Ha túl kevés mintainformációt használunk fel, akkor nem nyerünk elég jó képet a valóságról. Ha túl sokat használunk fel, akkor túlságosan „ráfókuszálunk” a mintánkra, és ez rontja a mintán kívüli elemek leírásának pontosságát (Wooldridge, 2016).

Az általánosító képesség növelésére irányuló törekvések eredményeképpen született meg a magyarázóváltozók 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 érjük el a lehető legkevesebb magyarázóváltozó felhasználásával.

Tehát, a parszimónia elve tulajdonképpen a (2)-ben látható módon módosítja az eredeti $\max_{\beta} \sum_{i=1}^{n} \ln L_i(\beta)$ feladatot.

$$\min_{\beta} \sum_{i=1}^{n} (-\ln L_i(\beta)) + \lambda_0 p \qquad (2)$$

A (2)-ben szereplő $\sum_{i=1}^{n} -\ln L_i(\beta) = \ell(\beta)$ tagot, mint a modell negatív log-likelihood értékét felfoghatjuk a modell „hibatagjának” is, mivel a negatív előjel alkalmazása miatt $\ell(\beta)$ csökkenésével javul a modell illeszkedése a megfigyelt mintaadatokra a célváltozóban.

Különböző λ_0 értékek mellett, különböző minimalizálandó célfüggvények származtathatók az eredeti likelihood maximalizálási feladatból (2) alapján. Mindegyik „származtatott” célfüggvény a következő alakot követi: $\ell(\beta)$ + *büntetőtag*, ahol a büntetőtag mindig a modellben felhasznált magyarázóváltozók számának (*p*-nek) valamilyen függvénye. Azt, hogy milyen mértékben büntessük a $\ell(\beta)$ csökkenését egy új változó bevonásának (tehát *p* növelésének) hatására, azt a λ_0 paraméter szabályozza. λ_0 függvényében különböző modellszelekciós célfüggvények születtek, amik a parszimónia elvét is számításba veszik. Ezen célfüggvényeket összefoglaló néven információs kritériumoknak hívjuk.

Az információs kritériumokat Ramanathan (2002) alapján foglaljuk össze.

Az első az Akaike Információs Kritérium/Akaike Information Criterion (*AIC*):

$$AIC = \frac{\ell(\beta)}{n} + \frac{2p}{n}$$

A második célfüggvény a Schwarz- Bayes Információs Kritérium/Bayesian-Schwarz Information Criterion (*SBC*):

$$SBC = \frac{\ell(\beta)}{n} + \frac{2p}{n}\ln\left(\sqrt{n}\right)$$

A harmadik lehetséges célfüggvény pedig a Hannan-Quinn Információs Kritérium/Hannan-Quinn Information Criterion (*HQC*):

$$HQC = \frac{\ell(\beta)}{n} + \frac{2p}{n}\ln(ln(n))$$

Ezen kívül sokszor használatos a korrigált McFadden-féle pszeudo R-négyzet mutató alkalmazása. 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: $D_i = 2(\ln L_i(\text{telített}) - \ln L_i(\beta))$. 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 a McFadden-féle pszeudo R-négyzet:

$$R^2 = 1 - \frac{D}{D_0}$$

A modell túlillesztésének elkerülése érdekében a megfigyelt mintára vonatkozóan még korrigálni szükséges az aktuális modell magyarázóváltozóinak számával:

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

Az információs kritériumokkal ellentétben a $\bar{R}^2$ mutató egy maximalizálandó célfüggvény. Ugyanis, ez a mutató az információs kritériumokkal szemben, nem az átlagos egységnyi elkövetett – modellből származó – hibát $\left(\frac{\ell(\beta)}{n}\right)$ minimalizálja, hanem azt méri le, hogy mennyivel tud az aktuális modellünk jobb becslést adni az eredményváltozóra, mint a magyarázóváltozók nélküli nullmodell.

A túlillesztés problémáját el lehet kerülni a keresztvalidált $\ell(\beta)$ vagy pszeudo R^2 kiszámításával is, de ha az a célunk, hogy egyszerre több $\tilde{X}$ halmaz teljesítményét is kiértékeljünk, akkor a korrigált R^2 vagy az információs kritériumok alkalmazása a számítási kapacitásokkal történő takarékosság miatt preferált lehet.

A változószelekció során tehát az $\tilde{X}$ halmazt úgy szükséges megválasztani, hogy $\bar{R}^2$ maximális vagy a preferált információs kritérium minimális legyen. Másképp megfogalmazva azt is mondhatjuk, hogy a (2) feladat eredményéül kapott optimális $\hat{\beta}_j$-k az $X_j \in \tilde{X}$ változókra lesznek nemnullák. 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). Ezen kívül pedig az is fontos szempont, hogy az optimálisnak ítélt modellben csak szignifikáns változók szerepeljenek. Azaz, olyan változó ne szerepeljen a modellben, aminek az együtthatója (β_i) közel 0, tehát nem lehet egyértelműen eldönteni, hogy változása (ceteris paribus) növeli

vagy csökkenti-e az eredményváltozó értékét. Ezt hipotézisvizsgálatok segítségével tudjuk ellenőrizni, együtthatónként (β_i) külön-külön (Wald-féle LR tesztek). A vizsgálat során a nullhipotézis tartalmazza azt az állítást, hogy az adott magyarázó változó ($\tilde{X}_i$) együtthatója (β_i) nem szignifikáns, míg az alternatív hipotézis szerint a változó együtthatója szignifikáns. Tehát, a nullhipotézis elutasítását vizsgáljuk egy adott α szignifikancia szinten. Ha bármelyik magyarázó változó együtthatója nem szignifikánsnak bizonyul, nem tartjuk optimálisnak az adott modellt ilyen szempontból. (Fahrmeir et al., 2013) Ez az extra feltétel az optimalizálás során tulajdonképpen még inkább biztosítja, hogy a végső, optimálisnak ítélt regressziós modell megfeleljen a parszimónia elvének.

4.1. A multikollinearitás jelensége

A változószelekció során kiemelten figyelni kell a multikollinearitás jelenségére. A multikollinearitás azt jelenti, hogy a magyarázóváltozók redundánsak, azaz egymást magyarázzák. Ez rendkívül káros hatással tud lenni a változószelekciós feladatunkat megoldó algoritmusok futására, mivel a regressziós modellekben fontos feltétel, hogy a magyarázóváltozóink függetlenek legyenek. Az együtthatóvektor $\hat{\beta}_j$ elemeinek értelmezése is ceteris paribus elvű, ami értelmét veszíti, ha a magyarázóváltozóink korrelálnak, azaz együtt mozognak.

Ezen a ponton fontos megjegyezni, hogy a függetlenség és a korrelálatlanság nem azonos fogalmak. Tetszőleges n egész szám mellett az $X_1, X_2, \dots, X_n$ valószínűségi változók akkor és csak akkor függetlenek együttesen, ha $F_{X_1,\dots,X_n}$ együttes eloszlásfüggvényük a marginális F_{X_i} eloszlásfüggvényeik szorzataként állítható elő:

$$F_{X_1,\dots,X_n}(x_1, \dots, x_n) = F_{X_1}(x_1) \cdot \dots \cdot F_{X_n}(x_n), \forall x_1, \dots, x_n - \text{re}$$

A tökéletes multikollinearitás ellenben csak azt jelenti, hogy egy X_i valószínűségi változó előállítható $X_1, X_2, \dots, X_n$ változók lineáris kombinációjaként. Nem tökéletes, de káros multikollinearitás esetén X_i varianciájának egy nagy hányada magyarázható meg $X_1, X_2, \dots, X_n$ változók lineáris kombinációjával. Ebből következik, hogy az $X_1, X_2, \dots, X_n$ valószínűségi változók függetlensége esetén köztük multikollinearitás nem áll fenn. Ám abból, hogy az $X_1, X_2, \dots, X_n$ változók között a multikollinearitás mértéke 0, abból nem következik, hogy ezek a változók függetlenek is, hiszen nem-lineáris összefüggés még fennállhat közülük. Egyedül akkor tekinthető a multikollinearitás teljes hiánya ekvivalensnek a vizsgált valószínűségi változók függetlenségével, ha azok együttes eloszlása többdimenziós normális eloszlást követ (Dowdy et al., 2011).

Mindennek ellenére a multikollinearitás vizsgálata fontos a GLM-ekben, hiszen ennek jelenléte esetén a redundancia biztosan fennáll az alkalmazott magyarázóváltozók között, ami biztosan torzítja a paraméterbecslés standard mintavételi hibáját (Nelder – Wedderburn, 1972) és megnehezíti a marginális hatások értelmezését. Tehát, a GLM-ek megtisztítása káros multikollinearitástól javítja a modellek paramétereinek becslési stabilitását és értelmezhetőségét, de mivel nem jelent teljes függetlenséget, így tökéletessé nem teszi azokat.
A multikollinearitás jelensége mérhető az úgynevezett VIF_j mutatószámmal. A j index arra utal, hogy mindegyik magyarázóváltozóra ki lehet számítani ezt az értéket, és azt méri, hogy az adott magyarázóváltozót milyen mértékben magyarázza (lineárisan) a modellben lévő többi magyarázóváltozó.
Ehhez felírjuk a következő segéd regressziót:

$$h\left(E\left(\tilde{X}_j\right)\right) = \varepsilon + \hat{\beta}_0 + \hat{\beta}_1\tilde{X}_1 + \cdots + \hat{\beta}_{j-1}\tilde{X}_{j-1} + \hat{\beta}_{j+1}\tilde{X}_{j+1} + \cdots + \hat{\beta}_p\tilde{X}_p$$

Ezen regressziós modellnek kiszámoljuk a korrigálatlan R^2 mutatóját. A továbbiakban ezt R_j^2-vel jelölve, azaz a j-edik változóhoz kötjük jelölésben is. Ebből pedig a VIF_j mutató értéke:

$$VIF_j = \frac{1}{1 - R_j^2}$$

Multikollinearitás hiánya esetén a mutató értéke 1, mert ekkor $R_j^2 = 0$. A mutató 1-2 közötti értéke esetén sem beszélhetünk igazán multikollinearitásról. Ha VIF_j 2-5 között van, akkor a multikollinearitás jelensége jelen van, de még nem káros. 5 felett pedig káros multikollinearitásról beszélünk. (Kovács, 2008)
Változószelekció során érdemes lehet olyan $\tilde{X}$ halmazokat preferálni, ahol egyik változó VIF_j mutatója sem haladja meg a 2 vagy 5 határértékeket. Egy ilyen korlát beépítésével a változószelekciós feladatba biztosíthatjuk, hogy az eredményváltozó értékének alakulását befolyásoló lineárisan független tényezőket egyértelműen azonosítani tudjuk a szelektált végső modell alapján. 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).

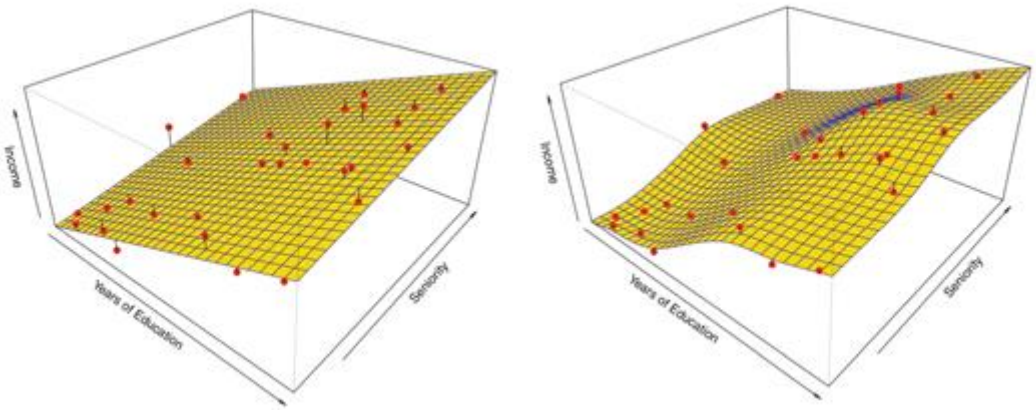
5. Az általánosított additív modellek (GAM) formális ismertetése

A fejezetben 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. Végül, bemutatjuk a concurvity jelenséget, ami a multikollinearitás fogalmának általánosítása nem-lineáris modellekre.

A GAM-ok az (1)-ben megadott lineáris modellt terjesztik ki, hogy úgy, hogy az X_j magyarázóváltozók nem csak egy egyszerű szorzót (β_j) kapnak, hanem valamilyen nem-lineáris f_j függvénnyel transzformálják őket a modellben (3). A modellek általános fogalmait Hastie – Tibshirani (1990) alapján ismertetjük.

$$h\big(E(Y)\big) = \varepsilon + \sum_{j=1}^{p} f_j\big(X_j\big) \tag{3}$$

Az (1) és (3) egyenletekkel adott modellek közötti különbség látványosan vizualizálható. James et al. (2013) példájában munkavállalók jövedelmét becsüljük iskolázottság és tapasztalat függvényeként. Ekkor a 2. ábrán látható, hogy a GLM egy síkra vetíti le a pontként ábrázolt munkavállalók z tengelyen szereplő jövedelmét (balra). Ezzel szemben, a GAM egy bonyolultabb felületet illeszt a jövedelem koordinátákákra (jobbra), ezzel nagyobb illesztési pontosságot ér el a megfigyelésekre a GLM-nél. Azonban a felület nem annyira összetett, hogy ne lehessen rajta azonosítani, hogy milyen tapasztalat és iskolázottság mellett lehet magasabb jövedelemre számítani.



2. ábra: Munkavállalók jövedelmének (*Income*) modellezése iskolázottság (*Years of Education*) és tapasztalat (*Seniority*) függvényeként. Forrás: James et al., 2013, p.22-23.

A (3) egyenlettel adott modell esetén a legfontosabb kérdés, hogyan tudjuk megadni az egyes magyarázóváltozókra illesztendő f_j transzformációs függvényeket. Hiszen itt is fontos az általánosító képesség és ennek kapcsán a túlillesztés kérdése. Ha túl egyszerű függvényt választunk, akkor a GAM nem fog pontosabb közelítést adni Y feltételes várható értékére, mint

a GLM. Ha túl bonyolult függvényt illesztünk, akkor ismét túlságosan „ráfókuszálunk” a mintánkra, mint abban az esetben, amikor túl sok magyarázóváltozót használunk a modellezéskor. Túl bonyolult függvény illesztése tehát csak a megfigyelt mintaadatokon javítaná a modell becslési pontosságát a lineáris esethez képest, a mintán kívüli populáció esetében könnyen elképzelhető, hogy egy ilyen függvény becslési pontossága még romlik is a lineáris modellhez képest.

Az egyszerűség érdekében az f_j függvények reprezentációjának kérdését először normális eloszlású eredményváltozóra mutatjuk be, ahol a link függvény az identitás. Ezzel GAM becslése az eredményváltozóra egyszerűen $\hat{Y} = \sum_{j=1}^{p} f_j(X_j)$ lesz, és az f_j függvények illesztése során a likelihood maximalizálás ekvivalens a $\min_{f_j} \left\| Y - \sum_{j=1}^{p} f_j(X_j) \right\|^2$ legkisebb négyzetek feladat megoldásával.

5.1. A bázis-spline függvények, mint transzformációs függvények alkalmazása

Leggyakrabban az f_j függvényeket egyszerű bázis-spline, röviden b-spline függvényeknek tekintik (Hastie – Tibshirani, 1990) (Hazewinkel, 2001). A b-spline függvények több, magasabb fokú polinomot (bázisfüggvényt) szakaszosan illesztenek össze a spline függvény értelmezési tartományán, vagyis a $[\min(X_j), \max(X_j)]$ intervallumon. Pl. 3 szakaszra illesztett másodfokú polinomok esetében:

$$B_1 = \frac{x^2}{2}, ha\ 0 \le x \le 1$$

$$B_2 = \frac{-2x^2 + 6x - 3}{2}, ha\ 1 \le x \le 2$$

$$B_3 = \frac{(3-x)^2}{2}, ha\ 2 \le x \le 3$$

A fenti példa egy harmadrendű b-spline, mivel egy r-ed rendű b-spline-nak mindig r szakasza van és a B_i függvények $r-1$ fokú polinomok. Ezt a tulajdonságot a b-spline függvények Cox de Boor rekurzív formulával történő megadása biztosítja:

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

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

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:

$$S_r(x) = \sum_i \alpha_i\, B_{i,r}(x)$$

Ahol i a megfelelő szakaszhatár (k_i) indexe x értelemzési tartományán.

Ha egy GAM-ban szeretnénk megbecsülni egy darab f_j függvényt b-splineok segítségével, akkor csak a következő egyszerű minimalizálási problémát kell megoldani:

$$\sum_{i=1}^{n} \left(y_i - S_r(x_{ji})\right)^2 = \|Y - S_r\|^2$$

Könnyen látható, hogy $r = 0$ esetben az alap lineáris modell alakját kapjuk vissza a spline függvények alkalmazásával.

A változószelekció során lineáris modellben csak arról kell dönteni, hogy szerepeltetjük-e X_j-t a modellben. Ez egy kisebb méretű, pl. 8 lehetséges magyarázóváltozót tartalmazó adatbázisban $2^8 - 1 = 255$ modell megvizsgálását jelenti, aminek következtében a globális optimum egyszerűen az összes részhalmaz megvizsgálásával megtalálható. Ellenben, ha GAM-mal modellezünk, és az f_j függvényeket b-spline-okkal reprezentáljuk, akkor a szelekció során dönteni kell a b-spline rendjéről és X_j értelmezési tartományának szakaszhatárairól is. A szakaszhatárokat általában egyenletesen osztják el X_j értelmezési tartományán, ám ez sokszor nem az optimális megoldás (Hastie – Tibshirani, 1990) (Zhou – Shen, 2001). Ezen felül, ha r-re egy ésszerű felső korlátot teszünk (r_{max}), akkor már m db lehetséges magyarázóváltozó esetén a megvizsgálandó részhalmazok száma $(r_{max} + 1)^m$. Hastie – Tibshirani (1990) javaslata a túlillesztés elkerülése érdékben $r_{max} = 6$, ami a kisméretű $m = 8$ adatbázisban is már $(6 + 1)^8 = 5\,764\,801$ db eset megvizsgálását jelenti. Ekkor pedig még nem is foglalkoztunk a k_i szakaszhatárok optimális elhelyezésével.

A fentiek miatt a klasszikus b-spline-ok alkalmazása nem javasolt nem-lineáris modellszelekció során, mivel a döntési változók megnövekedett száma miatt már a kis méretű adatbázisok is nagy keresési teret generálnak.

5.2. Thin plate spline függvények, mint transzformációs függvények alkalmazása

A b-spline függvények helyett a változószelekciós feladat során a Wood (2003), Wahba (1990) és Green – Silverman (1994) által javasolt thin plate spline függvényekkel érdemes az f_j függvények reprezentációját megoldani. Ebben az alfejezetben bemutatjuk, hogy thin plate spline függvények alkalmazása esetén nem kell foglalkozni a változószelekció során a csomópontok optimális elhelyezésével, és minimális számítási időt igényel a spline függvények rendjének megválasztása is.

5.2.1. Reprodukáló magú Hilbert-terek

A thin plate spline függvényekkel történő reprezentáció azt követeli meg az f_j függvényektől, hogy reprodukáló magú Hilbert-terek (továbbiakban röviden RMHT) elemei legyenek.

A reprodukáló magú Hilbert-tereket a XX. sz. közepén fedezték fel (Aronszajn, 1950), és az utóbbi évtizedekben terjedtek el széles körben a többváltozós statisztikában, nem-linearitások kezelésére, elsősorban a támaszvektor-gépek népszerűsége miatt (Boser et al., 1995). Lényegük, hogy adatainkat egy ún. reprodukáló magú Hilbert-térbe (RMHT) leképezve a szokásos lineáris faktor- és klaszteranalízis eljárások alkalmazhatók ahelyett, hogy az eredeti térben nem-lineáris módszereket hajtottunk volna végre. Magukat az adatokat nem is szükséges leképezni, ehelyett egy ún. magfüggvénnyel operálunk. A RMHT-ről definíciót és áttekintést Wahba (1990) alapján adunk.

Legyen X egy tetszőleges halmaz, és legyen $\mathcal{H}$ az X-en értelmezett valós függvények alkotta Hilbert-tér. A $\mathcal{H}$ Hilbert-teret alkotó függvényeken vett, úgynevezett kiértékelő L_x funkcionál, egy olyan lineáris funkcionál, ami minden függvényt kiértékel egy adott $x \in X$ pontban. Tehát, (4) szerint:

$$L_x: f \to f(x)\ \forall f \in \mathcal{H} \tag{4}$$

$\mathcal{H}$-t akkor mondjuk reprodukáló magú Hilbert-térnek minden x–re X–ben, ha L_x folytonos minden f-re $\mathcal{H}$–ban, vagy ekvivalensen: ha L_x korlátos operátor $\mathcal{H}$–n. Azaz, létezik olyan $M > 0$, hogy:

$$|L_x(f)| = |f(x)| \leq M\|f\|_{\mathcal{H}} \forall f \in \mathcal{H} \tag{5}$$

Bár (5) a leggyengébb feltétel, ami egyidejűleg biztosítja a skalárszorzat és a kiértékelő funkcionál létezését $\mathcal{H}$–n az értelmezési tartomány minden pontjában, de gyakorlati alkalmazások szempontjából nem a legkönnyebben megfogható definíció. A RMHT-k egy intuitívabb definíciója adható meg, ha megfigyeljük, hogy (5) garantálja, hogy a keresett kiértékelő L_x funkcionál reprezentálható egy $\mathcal{H}$–beli K_x függvény és f skalárszorzataként. Ez a K_x függvény az úgynevezett reprodukáló magfüggvénye $\mathcal{H}$–nak, amiről a RMHT-k a nevüket kapták.

Formálisabban nézve, a Riesz-féle reprezentációs tétel állítása szerint minden x–re X–ben egyértelműen létezik egy függvény $\mathcal{H}$–ból, ami rendelkezik a reprodukáló tulajdonsággal (6).

$$L_x(f) = f(x) = \langle f, K_x \rangle \forall f \in \mathcal{H} \tag{6}$$

Mivel K_y önmagában véve egy függvény $\mathcal{H}$–n, így minden y-ra X–ben igaz, hogy:

$$K_y(x) = \langle K_y, K_x \rangle$$

Ebből a tulajdonságból kiindulva definiálni tudjuk $\mathcal{H}$ reprodukáló K magfüggvényét az alábbi skalárszorzatos formában:

$$K: X \times X \to \mathbb{R}$$

$$K(x, y) = \langle K_y, K_x \rangle$$

A fenti definícióból következik, hogy $K: X \times X \to \mathbb{R}$ szimmetrikus és pozitív definit. Azaz,

$$\sum_{i,j=1}^{n} c_i c_j K(x_i, x_j) \geq 0$$

minden $n \in \mathbb{N}$-re, $x_1, \dots, x_n \in X$-re és $c_1, \dots, c_n \in \mathbb{R}$-re. Az Aronszajn (1950)-ben található Moore-Aronszajn tétel szerint, ha egy K függvény rendelkezik a fenti pozitív definit és szimmetria tulajdonságokkal, akkor létezik az X–en értelmezett x függvényeknek olyan Hilbert-tere, amin K egy reprodukáló magfüggvény.

5.2.2. A thin plate spline függvények illesztése és alkalmazásuk előnye a változószelekció során

A RMHT definíciójának áttekintése után, térjünk vissza thin plate spline függvények illesztéséhez. A thin plate spline-ok lényege, hogy a keresett f függvény egy reprodukáló magú Hilbert-tér (RMHT) eleme legyen és legyen megoldása (7) minimalizálási feladatnak.

$$\min_f \|Y - f\|^2 + \lambda \int f''(x)^2 dx \qquad (7)$$

Ahol λ választható paraméter, ami szabályozza az egyensúlyt a keresett f függvény pontos illeszkedése (első tag), és kellő simasága (második tag) között. Tehát, azzal, hogy a függvényillesztés feladatába f második deriváltján keresztül bevittünk egy büntetőtagot, szabályozzuk az illesztett függvény bonyolultságát. Jól megválasztott λ mellett hiába illeszkedik a célváltozóra a megfigyelt pontokban pontosan pl. egy 10-ed fokú polinom, a második deriváltja egy ilyen függvénynek túl magas lesz minden x pontban (nem lesz kellően sima a felülete, azaz túlilleszkedik), így ez nem lesz megoldása a fenti optimalizálási feladatnak. Helyette egy egyszerűbb f függvényt részesít a célfüggvény előnyben, ezzel elkerülve a spline túlillesztését a tanítóminta adataira. Az elv hasonló ahhoz, amikor lineáris modellszelekcióban információs kritériumokat alkalmazunk: a minimalizálandó kritériumban a modellhiba tag csökken új változó bevonásával, de a kritérium értéke nő a használt magyarázóváltozók számával arányosan.

A (7)-ben megadott függvényillesztési feladattal elértük, hogy az illesztett függvény fokszáma optimális legyen: pontosan illeszkedjen a megfigyelt adatokra, de ne legyen „túl bonyolult”,

azaz a második derivált minden pontban legyen kellően alacsony, ezzel elkerüljük a felesleges inflexiós pontokat.

(7) feladattal az egyetlen probléma csupán az, hogy a függvény értékét gyakorlatilag pontonként becsülni kell, tehát minden x pont egy illesztési pont is. Továbbá, λ becslését is meg kell oldani. Tehát, több paramétert kell becsülni, mint ahány adatpontunk van, ami nem-polinomiális idejű becslési algoritmusokhoz vezet (Hutchinson – de Hoog, 1995). Emiatt f-et kénytelenek vagyunk a b-spline-okhoz hasonlóan osztópontonként becsülni, amelyek optimális száma és helye x értelmezési tartományán továbbra is kérdéses.

A feladat megoldásához érdemes látni, hogy a (7) minimalizálási feladat megoldása megkapható (8) alakban, általánosan egy darab X_j magyarázóváltozóra.

$$f_j(X_j) = \sum_{i=1}^{n} \psi_i \eta_j\left(\left\|X_j - x_{ji}\right\|\right) \tag{8}$$

Ahol n a megfigyelt minta elemszáma és $\eta(v) = \frac{\Gamma(-3/2)}{16\pi^{1/2}} v^3$. $\Gamma(s) = \int_0^\infty t^{s-1}e^{-t}dt$, a gamma-függvény. A megoldás ezen alakjával tudunk definiálni egy $\boldsymbol{E} \in \mathbb{R}^{n\times n}$ mátrixot:

$$\{\boldsymbol{E}\}_{ab} = \eta_j\left(\left\|x_{ja} - x_{jb}\right\|\right)$$

Ezzel a (7) spline illesztési feladat átírható a (9) alakba.

$$\min_{\Psi} \|Y - \boldsymbol{E}\Psi\|^2 + \lambda \Psi^{\boldsymbol{T}} \boldsymbol{E} \Psi \tag{9}$$

Ahol $\Psi = [\psi_1, \psi_2, \dots, \psi_n]$ paramétervektor. Ahhoz, hogy ne $n+1$ paramétert kelljen meghatározni a minimalizálási feladat során, egyszerűen vegyük $\boldsymbol{E} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^{\boldsymbol{T}}$ spektrálfelbontását. Természetesen, a felbontás csupán szimmetrikus mátrixokra létezik (Golub – Van Loan, 1996). Viszont a fentiekben megadott definíció alapján láttuk, hogy $\boldsymbol{E} \in \mathbb{R}^{n\times n}$, így a spektrálfelbontás létezik. Tehát, $\boldsymbol{D}$ mátrix az $\boldsymbol{E}$ mátrix sajátértékeiből álló diagonális mátrix, és $\boldsymbol{U}$ mátrix oszlopaiban $\boldsymbol{E}$ mátrix sajátértékeihez tartozó sajátvektorok állnak. Válasszuk ki a $k < n$ legnagyobb sajátértéket $\boldsymbol{D}$-ből, és legyen $\boldsymbol{U_k}$ egy olyan mátrix, melynek oszlopai a kiválasztott k legnagyobb sajátértékhez tartozó sajátvektorok. Ezekből pedig megadható $\boldsymbol{E_k} = \boldsymbol{U_k}\boldsymbol{D_k}\boldsymbol{U_k^T}$ $n \times k$ dimenziós mátrix (ahol $\boldsymbol{D_k}$ az $\boldsymbol{E}$ mátrix k legnagyobb sajátértékeiből álló diagonális mátrix), ami a legjobb közelítése euklideszi normával $\boldsymbol{E}$-nek a megadott k dimenziós térben. Tehát, az előbbi módon megadott $\boldsymbol{E_k}$ minimalizálja a $\|\boldsymbol{E} - \boldsymbol{E_k}\|^2$ távolságot.

Ezek után oldjuk meg (10) minimalizálási feladatot, ami a legjobb közelítése (9)-nek a kisebb, k dimenziós térben (Wood, 2003).

$$\min_{\Psi_k} \|Y - \boldsymbol{U_k}\boldsymbol{D_k}\Psi_{\boldsymbol{k}}\|^2 + \lambda \Psi_k^T \boldsymbol{D_k} \Psi_{\boldsymbol{k}} \tag{10}$$

Ahol Ψ_k már csak egy k dimenziós paramétervektor, ami így polinomiális időben becsülhető. Gyakorlatilag a thin plate spline illesztését x értelmezési tartományának k db szakaszán végezzük el k db bázisfüggvény segítségével (melyek értéke a $\boldsymbol{U_k D_k} \in \mathbb{R}^{n \times k}$ mátrix oszlopaiban kerül tárolásra az egyes megfigyeléseinkre), és a szakaszhatárokat a spektrálfelbontás alkalmazásával úgy adtuk meg, hogy a lehető legjobban közelítsük meg azt a feladatot, amikor a spline függvény pontonként illesztjük. Ilyen értelemben tehát egy optimális osztópont rendszerhez jutottunk.

A thin plate spline függvények alkalmazásának legfontosabb előnye, hogy a változószelekciós feladatban az egyetlen dolog amiről dönteni kell, az a k értéke.

Viszont, csak arra szükséges figyelni, hogy k elég nagy legyen ahhoz, hogy a $\|\boldsymbol{E} - \boldsymbol{E_k}\|^2$ eltérés ne legyen szignifikáns. Ezt pedig formális statisztikai próbával tudjuk tesztelni Augustin et al. (2012) és Wood (2017) alapján. A két tanulmány által javasolt próbák nullhipotézise, hogy a $\|\boldsymbol{E} - \boldsymbol{E_k}\|^2$ eltérés nem szignifikáns.

Innentől kezdve változószelekciós szempontból csak annyi dolgunk van, hogy ha egy X_j magyarázóváltozót beveszünk a GAM modellbe, akkor kellően nagy k-t válasszunk hozzá. Ezt úgy érhetjük el, hogy alapértelmezés szerint $k = 10$-et választunk (Wood, 2017 alapján). Amennyiben a változónak kevesebb különböző lehetséges értéke van, akkor k az X_j magyarázóváltozó lehetséges értékek számával lesz egyenlő. Amennyiben az Augustin et al. (2012)-féle próbák p-értéke egy előre megadott α szignifikancia-szint alatti, akkor a k értékét pl. 5-ösével növeljük addig, amíg a változóhoz megfelelően nagy k-t nem választottunk.

Fontos megjegyezni, hogy a túl magas k választás csupán extra számítási kapacitásba kerül, túlilleszteni nem fogjuk a spline függvényt. Ugyanis az eredeti (7) feladat $\lambda \int f''(x)^2 dx$ büntetőtagja lineáris függvényformát fog preferálni egy olyan szakaszon X_j értelmezési tartományán, ahol a megfigyelt pontok csak valamilyen (általában normális eloszlásúnak feltételezett) zajjal térnek el a lineáris függvényformától. Ezzel a változószelekciós feladat bonyolultsága megmarad $2^m - 1$-nek, hiszen k választására nem kell új döntési változókat bevezetni.

5.2.3. GAM illesztése thin plate spline függvények használatával

Amennyiben az eredeti p db magyarázóváltozót használó GAM modell paraméterbecslési feladatát szeretnénk megfogalmazni a 5.2.2. fejezetben ismertetett thin plate spline-ok segítségével, akkor be kell vezetnünk két új mátrixot.

Legyen k_j a j-edik magyarázóváltozóhoz meghatározott és az Augustin et al. (2012)-féle próbával ellenőrzött, optimális k dimenzió. Legyen $\boldsymbol{A}$ mátrix olyan, melynek oszlopaiban

az egyes X_j magyarázóváltozók nem-lineáris f_j transzformációinak bázisfüggvényeit reprezentáló $\boldsymbol{U_{k_j} D_{k_j}}$ mátrixok állnak. Könnyű látni, hogy ezzel $\boldsymbol{A}$ mátrix $n \times \sum_{j=1}^{p} k_j$ dimenziós lesz. Továbbá legyenek $\boldsymbol{S_j}$-k $\sum_{j=1}^{p} k_j \times \sum_{j=1}^{p} k_j$ mátrixok, amelyek főátlóinak megfelelő k_j elemű szakaszán a megfelelő $\boldsymbol{D_{k_j}}$ diagonális mátrixok elemei állnak, és 0-k egyébként. $\tilde{\Psi}$ pedig legyen a Ψ_{k_j}-k egymás utáni sorozatából előálló $\sum_{j=1}^{p} k_j$ dimenziós együtthatóvektor. Ezen jelölésekkel a GAM illesztési feladat normális eredményváltozó mellett identitás link függvénnyel (11) alakban lesz felírható.

$$\min_{\tilde{\Psi}} \left\| Y - \boldsymbol{A}\tilde{\Psi} \right\|^2 + \sum_{j=1}^{p} \lambda_j \tilde{\Psi}^{\boldsymbol{T}} \boldsymbol{S_j} \tilde{\Psi} \tag{11}$$

Tetszőleges $h(\cdot)$ link függvény és bármely exponenciális eloszláscsaládba tartozó eredményváltozó esetén (11)-ben egyszerűen vissza kell térni a modellt az adott együtthatóvektor mellett jellemző negatív log-likelihoodra, $\ell(\tilde{\Psi})$-re (12). Hiszen, a thin plate spline függvények az $\boldsymbol{A}$-val reprezentált bázisfüggvények $\tilde{\Psi}$-vel vett lineáris kombinációnak segítségével adnak $\hat{Y} = \boldsymbol{A}\tilde{\Psi}$ módon becslést az eredményváltozó feltételes várható értékére.

$$\min_{\tilde{\Psi}} \ell(\tilde{\Psi}) + \sum_{j=1}^{p} \lambda_j \tilde{\Psi}^{\boldsymbol{T}} \boldsymbol{S_j} \tilde{\Psi} \tag{12}$$

Az egyetlen fennmaradt kérdés a λ_j-k megválasztása, amit Wood (2011) alapján korlátozott maximum likelihood (REML: REstricted Maximum Likelihood) módszerrel érdemes megtenni. A REML módszerben technikailag 10-szeres keresztvalidáció segítségével keressük azon λ_j-ket, melyek mellett (12)-t megoldva a legkisebb célfüggvény értéket kapjuk.

Fontos megjegyezni, hogy a thin plate spline-ok segítségével megadott GAM-ra alkalmazott információs kritériumok és az $\bar{R}^2$ képleteiben a modellben lévő magyarázóváltozók számát jelölő p helyett a modell összes paraméterének a számát, azaz $\sum_{j=1}^{p} k_j$-t kell venni.

5.2.4. A GAM illesztési feladat párhuzamosítása $\boldsymbol{QR}$ dekompozíció segítségével

Numerikus okokból, nagyobb adatbázisokon nem feltétlenül célszerű közvetlenül (12)-t megoldani pl. Newton – Raphson módszerrel. Helyette (11)-nek egy súlyozott változatát, (13)-t érdemes megoldani iteratívan súlyozott, büntetőtagos legkisebb négyzetek módszerrel, ami (12)-vel azonos megoldásra vezet (Wood et al., 2015).

$$\min_{\tilde{\Psi}} \left\| \boldsymbol{W}\boldsymbol{z} - \boldsymbol{W}\boldsymbol{A}\tilde{\Psi} \right\|^2 + \sum_{j=1}^{p} \lambda_j \tilde{\Psi}^{\boldsymbol{T}} \boldsymbol{S_j} \tilde{\Psi} \tag{13}$$

(13) feladatban a súlyok $\boldsymbol{W}$ mátrixának megadásához be kell vezetni újabb jelöléseket. Legyen $\hat{\mu}_i = y_i + \xi_i$, ahol ξ_i egy nagyon kicsi érték (legtöbbször 0), ami csak néhány speciális esetben szükséges $h(\hat{\mu}_i)$ létezésének biztosításához. Legyen $V(\cdot)$ függvény olyan, hogy az eredményváltozó varianciája $V(\hat{\mu}_i)$ ϕ konstansszorosa legyen minden i egyedre:

$$Var(y_i) = \phi V(\hat{\mu}_i)$$

Legyen továbbá

$$\boldsymbol{z} = \{z_i = h'(\hat{\mu}_i)(y_i - \hat{\mu}_i) + h(\hat{\mu}_i)\} \in \mathbb{R}^n$$

és

$$w_i = V(\hat{\mu}_i)^{-1/2} h'(\hat{\mu}_i)^{-1}$$

Végül, $\boldsymbol{W}$ pedig legyen a w_i-ből képzett $n \times n$-es diagonális mátrix.

(13) megoldásának előnye, hogy könnyen párhuzamosítható. Erre szükség is van, hiszen a λ_j-k kiszámítása 10-szeres keresztvalidációval (13)-ban (és (12)-ben is) elég költséges tud lenni futásidőben. Ugyanis, minden egyes megvizsgált λ_j értékhez (13)-t meg kell oldani 10-szer külön-külön a keresztvalidáció során.

Ezzel minden λ_j teszteléséhez a keresztvalidációban ki kell számolni és a memóriában kell tartani $\boldsymbol{WA}\widetilde{\Psi}$-t. A párhuzamos kiszámítást lehetővé tevő felírás megadásához Lang et al. (2014), Wood et. al. (2015) és Li – Wood (2019) eredményeit vesszük alapul. Az idézett tanulmányok azt az ötletet dolgozták ki és tesztelték le, hogyan lehet a GAM modelleket kisebb RAM terhelés mellett kiszámítani, hogy nagyobb adatmátrixra ($n = 10^8$, $\left|\widetilde{\Psi}\right| = 10^4$) is illeszteni lehessen GAM modelleket. A megoldásuk $\boldsymbol{WA}$ mátrix *QR* dekompozíciója, aminek alkalmazásával nem csak a GAM illesztés RAM igénye csökkenthető drasztikusan, hanem a minimalizálási feladat megoldása is könnyen párhuzamosíthatóvá válik.

A *QR* dekompozíció szerint: $\boldsymbol{WA} = \boldsymbol{QR}$, ahol $\boldsymbol{Q} \in \mathbb{R}^{n \times \sum_{j=1}^{p} k_j}$ és $\boldsymbol{R} \in \mathbb{R}^{\sum_{j=1}^{p} k_j \times \sum_{j=1}^{p} k_j}$. A felbontásban $\boldsymbol{Q}$ mátrix ortogonális, míg $\boldsymbol{R}$ felső háromszög mátrix. Fontos látni, hogy a felbontás csak nem szinguláris mátrixok esetén létezik. Az eljárás fontos korlátja, hogy a felbontás pontosságát nagyban befolyásolja a felbontandó mátrix kondíciószáma és egy rosszul kondicionált problémánál ez jelentős számítási hibát okozhat, amely az iterációk során halmozódhat (Golub – Van Loan, 1996).

Mindezen korlátok figyelembe vételével, ha bevezetjük a $f = \boldsymbol{Q^T W z}$ vektort és a $\|r\|^2 = \|\boldsymbol{Wz}\|^2 - \|f\|^2$ normanégyzetet, akkor a (13) feladat (14)-re módosul.

$$\min_{\widetilde{\Psi}} \left\| f - \boldsymbol{R}\widetilde{\Psi} \right\|^2 + \|r\|^2 + \sum_{j=1}^{p} \lambda_j \widetilde{\Psi}^{\boldsymbol{T}} \boldsymbol{S_j} \widetilde{\Psi} \qquad (14)$$

Tehát, nincs szükségünk a teljes $\boldsymbol{WA}$ mátrixra, elég egy $\sum_{j=1}^{p} k_j \times \sum_{j=1}^{p} k_j$ dimenziós részét használnunk, nincs szükség $\boldsymbol{Q}$–ra. Ráadásul, ha az n elemű mintánkat M db blokkra bontjuk, akkor lehetséges blokkonként párhuzamosan kiszámítani az $\boldsymbol{R}_i$ mátrixokat és f_i vektorokat. Végül, az alkotóelemeik párhuzamos kiszámítása után újra össze tudjuk állítani $\boldsymbol{R}$–t és f–et, egy újabb *QR* dekompozíciót alkalmazva:

$$\begin{pmatrix} \boldsymbol{R}_1 \\ \vdots \\ \boldsymbol{R}_M \end{pmatrix} = \tilde{\boldsymbol{Q}}\boldsymbol{R}$$

$$f = \tilde{\boldsymbol{Q}}^T \begin{pmatrix} f_1 \\ \vdots \\ f_M \end{pmatrix}$$

Tehát, a GAM illesztési feladat megoldása során M db különböző almintán $\boldsymbol{R}_i$ mátrixokat és f_i vektorokat tudunk M db processzormagon párhuzamosan kiszámítani. Ezzel párhuzamosítani tudjuk a (13)-mal adott GAM illesztési feladatot.

5.3. Concurvity jelenség

A végső GAM 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 magyarázóváltozók nem-lineáris transzformáltjai között zavaró mértékű összefüggés. Tehát, a multikollinearitás jelenségének nem-lineáris változatát, a concurvity-t (Wood, 2017) szükséges megmérnünk.

Fontos látni, hogy bár a concurvity jelenség egy általánosabb leírása a modell magyarázóváltozóinak összefüggéseire, mint a multikollinearitás, ám a concurvity hiányából sem következik a magyarázóváltozók közti függetlenség a 4.1. alfejezetben definiált módon, hiszen előfordulhat, hogy az összefüggés jellege nem írható le spline függvények segítségével. Valószínűségi változók között a teljesen általános függetlenség csak páronként mérhető különböző függetlenségi kritériumok segítségével, amik általában két valószínűségi változó együttes eloszlásának momentumait használják fel. Viszont, a többváltozós függetlenségre ezek a kritériumok nem általánosíthatók (Gallager, 2013). Egy ilyen függetlenségi kritérium jelen dolgozatban is bemutatásra kerül: a Hilbert-Schmidt függetlenségi kritérium a 6.7. alfejezetben. Tehát, a GLM-ekhez hasonló módon GAM-ok esetén sem mondhatjuk, hogy a concurvity jelenség megszüntetése tökéletesen ki fogja elégíteni a modell által feltételezett additív struktúrát a magyarázóváltozók között. Viszont, a modellek paramétereinek becslési stabilitása és értelmezhetősége javul (Wood, 2017).

A concurvity jelenség mérésére GAM modellekben, thin plate spline függvények használata esetén Wood (2017) javasol egy mutatót. A mutató alapötlete, hogy egy f_j függvény $\boldsymbol{U_{k_j}D_{k_j}}$ mátrixszal reprezentált bázisfüggvényeinek lineáris kombinációjával kinyerhető egy g_j függvény, ami más $f_{c\neq j}$ függvények $\boldsymbol{U_{k_c}D_{k_c}}$ reprezentációjának is része. A Wood-féle concurvity mérték ezt a redundancia hatást a $\left(\frac{\|g_j\|}{\|f_j\|}\right)^2$ hányados segítségével helyezi el egy $[0,1]$ skálán. Ha az X_j-hez rendelt concurvity mérték 0,5-nél nagyobb, akkor a változó hatásának értelmezését károsan befolyásoló concurvity van a GAM modellben. Ez analóg a lineáris eset $VIF_j > 2$ határával, hiszen ekkor a VIF_j képletben az $R_j^2 > 0{,}5$.

Kétféle concurvity mértéket is rendelhetünk egy X_j változóhoz. Az egyik a megfigyelt concurvity, ami a modell paraméterbecslése során megkapott Ψ vektorral előálló f_j függvények közötti concurvity-t méri. Ezzel szemben a pesszimista concurvity mérték megkeresi azt a $\widetilde{\Psi}$ vektort, amivel a $\boldsymbol{U_{k_j}D_{k_j}}$-ket súlyozva a legmagasabb concurvity mérték érhető el (Wood, 2017).

6. Változószelekciós algoritmusok GAM keretben

A GAM keretben működő változószelekciós algoritmusok három nagyobb csoportra bonthatók. Az első csoport a klasszikus stepwise logikát követi: egyszerre csak egy változót ad hozzá vagy vesz el a modellből, és ezt a műveletet addig folytatja, amíg a kiválasztott változószelekciós célfüggvény már nem javul szignifikánsan. A konkrét algoritmusok között a stepwise lépések finomságában találhatunk különbségeket. A klasszikus Stepwise algoritmus binárisan kezeli a változókat: egy változó vagy benne van $\tilde{X}$-ben vagy nem. A GAMBoost algoritmus viszont a 5.1. fejezetben megismert b-spline függvények alkalmazásával egy lépésben mindig azon változó „súlyát" növeli a modellben, amivel a legjobb változást tudja elérni a kiválasztott változószelekciós célfüggvényben. Az algoritmus végén kapott modellben a 0 súllyal (minden bázisfüggvény együtthatója 0) szereplő változók azok, amiket szelektált a módszer. Tehát ez az algoritmus is a stepwise logikát követi, hiszen egy lépésben csak egy változón módosít a modellben, csak ezt nem bináris, hanem folytonos módon teszik meg. A GAMBoost algoritmus alapötletét a módosított backfitting eljárás fejleszti tovább oly módon, hogy lehetővé teszi egy lépésben több változó „súlyának" növelését is a modellben, ezzel jól párhuzamosíthatóvá téve az algoritmust.

A második csoport a regularizációs módszerek családja. Ezek a módszerek a Tibshirani (1996)-féle Lasso algoritmus általánosításainak tekinthetők nem lineáris esetre. A regularizációs módszerek alapötlete, hogy a változószelekciót a modellek paramétereinek becslési eljárásába beépítik, mint extra korlátozó feltételeket. Ezen extra korlátok azt eredményezik, hogy bizonyos változókhoz tartozó bázisfüggvény együtthatók már a becslés során 0-nak adódnak. Az ötlet hasonló, mint a „folytonos" stepwise algoritmusok esetében, csak itt nem lépésenként változik egyszerre egy változó „súlya", hanem a becslés során megoldandó optimalizálási feladatba épített extra korlátok miatt adódnak 0 „súlyú" változók a modellekben.

Megjegyzendő, hogy sem a stepwise, sem a regularizációs módszerek nem biztosítják közvetlenül a multikollinearitás, vagy nem-lineáris esetben a concurvity jelenség elkerülését a modellben. Sőt, több szerző kimondottan felhívja a figyelmet arra, hogy a stepwise és regularizációs modellek konzisztenciája sérül multikollinearitás, vagy nem-lineáris esetben concurvity jelenség fennállása esetén a lehetséges magyarázóváltozók körében: Chong – Jun (2005), Zhao – Yu (2006), Signoretto et al. (2008), Jia – Yu (2010). A konzisztencia hiánya GAM esetben azt jelenti, hogy a változószelekció során olyan változók kerülnek be $\tilde{X}$, amikhez a teljes populációra felírt regresszióban minden bázisfüggvény 0 együtthatóval szerepel,

és olyan változók bázisfüggvényeinek lesz kivétel nélkül 0 súlya a végső modellben, amik a teljes populációra felírt modellben nem csupa 0 együtthatójú bázisfüggvényekkel szerepelnek. Mivel a stepwise és regularizációs módszerek érzékenyek a concurvity jelenségre, így érdemes lehet megvizsgálni a változószelekciós módszerek egy harmadik csoportját, ami az eredményváltozó és az egyes magyarázóváltozók közös információtartalma alapján végez változószelekciót. Az ide tartozó algoritmusok olyan szempontból speciálisak, hogy nem feltételeznek GAM (és semmilyen más modell) keretet a változószelekció során. Ezek az algoritmusok definiálnak egy $I(x,y)$ mértéket x és y valószínűségi változók közös információtartalmának mérésére, majd különböző optimalizáló algoritmusok segítségével maximalizálnak egy egyedi változószelekciós célfüggvényt. Ezek a célfüggvények arra az elvre épülnek, hogy olyan X_j változókat válasszanak be a modellbe (ami GAM mellett lehet akár véletlen erdő, támaszvektor-gép, neurális hálózat stb. is), melyekre $I(Y, X_j)$ magas, de a $k \neq j$ esetekre $I(X_k, X_j)$ mérték alacsony. Ez utóbbi szempont beemelésével ezek az algoritmus célfüggvényükben már védekeznek a modellbe válogatott változók közötti redundancia, tehát a multikollinearitás vagy a concurvity ellen. Ugyanakkor, az $I(x,y)$ jellegű mértékek alkalmazása a magyarázóváltozók közti kapcsolat szorosságának mértékét csak páronként vizsgálja. Az algoritmusok nem kezelik azt az esetet, amikor egy magyarázóváltozó több más magyarázóváltozó többváltozós függvényeként áll elő. Emiatt az mRMR-rel és a HSIC-Lassoval nyert GAM-ok esetében továbbra is fennállhat a concurvity jelenség. Mindkét ismertetett algoritmus a Hall (1999)-féle lineáris korreláció alapú változószelekciós algoritmus általánosításának tekinthető nem-lineáris esetre, így ennek az ismertetését is megtesszük az 6.5. fejezetben.

A további alfejezetekben a három algoritmus csoport konkrét tagjainak működését ismertetjük. A lista biztosan nem teljes körű. A részletesen vizsgált algoritmusok körét a hivatkozások száma és az R vagy Python nyelveken elérhető implementáció megléte határozta meg.

6.1. Regularizációs módszerek

A regularizációs módszerek tárgyalását először a klasszikus, lineáris modellek körében értelmezett Lasso módszerrel kezdjük. A Lasso-n keresztül bemutatjuk a regularizációs együtthatóbecslésen keresztül történő változószelekció alapgondolatait, majd ezeket felhasználva mutatjuk meg az eredmények általánosítását különböző nem-lineáris modellspecifikációkra.

A COSSO módszer (Lin – Zhang, 2006) a regularizációs változószelekciót az RMHT-k elméletét erősen felhasználó funkcionális ANOVA elven történő függvényreprezentációval

oldja meg. A 6.1.3. fejezetben bemutatott módszerek a thin plate spline-ok (12)-ben adott illesztési feladatára vezetnek be újabb regularizációs tagot a $\sum_{j=1}^{p} \lambda_j \tilde{\Psi}^{\boldsymbol{T}} \boldsymbol{S_j} \tilde{\Psi}$ tag mellé a változószelekció megvalósításához. A nemnegatív Garrotte módszer (Cantoni et al., 2011) pedig b-spline függvényeken keresztül alkalmazza a lineáris Lasso becslést.

6.1.1. Lasso

A Lasso módszer az első regularizációs együtthatóbecslési eljárás, ami változószelekcióra is alkalmazható. Tibshirani (1996)-ban jelent meg először, a módszer ismertetését is ez alapján végezzük.

A Lasso becslés az (1)-ben megadott GLM β együtthatóvektorát a (15) feladat megoldásával becsüli meg.

$$\hat{\beta} = \min_{\beta} \ell(\beta) + \lambda \|\beta\|_1 \tag{15}$$

Ahol $\ell(\beta)$ a GLM negatív log-likelihood függvénye, $\|\cdot\|_1$ az L_1 norma, λ pedig egy állítható paraméter $\lambda \in (0,1)$. Az új becslési eljárásban az együtthatók L_1 normájának, mint büntetőtag (vagy más kifejezéssel regularizáció) hozzáadása a negatív log-likelihood minimalizálási feladathoz azt eredményezi, hogy az együtthatók becslései torzítottak lesznek, ám a standard hibájuk olyan mértékben lecsökken, hogy a becslés átlagos négyzetes hibája kisebb, mint az eredeti negatív log-likelihood minimalizálásból kapott becsléseké. Átlagos négyzetes hibán (Mean Squared Error, MSE) a becslési variancia és a torzítottság foka négyzetének összegét értjük:

$$MSE(\hat{\beta}) = SE^2(\hat{\beta}) + Bs^2(\hat{\beta})$$

Az együtthatók L_1 normája, mint büntetőtag a negatív log-likelihood minimalizálásban ráadásul azt is eredményezi, hogy alkalmas λ választás mellett $\exists\, j: \hat{\beta}_j = 0$. Tehát, modellszelekciót hajtunk végre. Tibshirani (1996) azt is megmutatta, hogy ha λ-t pl. az információs kritériumok minimalizálásával választjuk ki 10-szeres keresztvalidációval, akkor az információs kritériumok tekintetében sokkal kedvezőbb eredményeket kapunk a klasszikus stepwise eljárásokhoz képest. Ráadásul, a becslés és λ megválasztása nem igényel kimondottan magas számítási kapacitást még $m > 20$ esetben sem, mivel a Lasso minimalizálási feladat könnyen visszavezethető a Newton-Raphson algoritmus egy változatára (Tibshirani, 1996).

Azonban, fontos megjegyezni, hogy Zhao – Yu (2006) állítása szerint a Lasso-módszer nem megfelelően kezeli a multikollinearitás jelenségének káros hatásait a modellszelekció során. Amennyiben a lehetséges magyarázóváltozók közül azon változók melyekre a sokaságban

$\beta_j = 0$ erősen korrelálnak azon változókkal, melyekre $\beta_j \neq 0$, akkor a (15) megoldásaként kapott $\hat{\beta}_j$ becsléseink a következő két tulajdonsággal rendelkeznek:

- $\beta_j \neq 0$ együtthatók $\hat{\beta}_j$ becslése extrém mértékben zsugorodni kezd (negatív irányban válik torzítottá), akár nullák is lehetnek, azaz kieshetnek a modellből.
- $\beta_j = 0$ együtthatók $\hat{\beta}_j$ becslése túl magas lehet (pozitív irányban válik torzítottá), így nem biztos, hogy kiesnek a modellből.

A Lasso módszert nem-lineáris esetekre is általánosítani lehet. A lehetőségek abban különböznek, hogy a (3)-al adott GAM-ban az f_j függvények reprezentációjára milyen megoldást alkalmazunk.

6.1.2. COSSO

A Lasso módszer elvére alapuló változószelekciós algoritmus GAM keretben először Lin – Zhang (2006)-os munkájában jelent meg. Az idézett munkában bemutatott COSSO módszer (COmponent Selection and Smoothing Operator) a RMHT-k elméletére építve végez változószelekciót nem-lineáris regresszióban, a Lasso módszerhez hasonlóan L_1 normával regularizálva a modell paraméterbecslési feladatát.

A COSSO-módszer az alábbi formában feltételezi, a nem-lineáris regressziós összefüggést:

$$h\big(E(Y)\big) = \eta(\boldsymbol{X}) + \varepsilon$$

Ahol, $\boldsymbol{X}$ $n \times m$-es mátrix oszlopait az X_j lehetséges magyarázóváltozók adják. $\eta(\cdot)$ többváltozós függvényt úgynevezett funkcionális ANOVA felbontásban adjuk meg spline forma helyett.

Egy d dimenziós η függvény funkcionális ANOVA felbontása (16) alakban adódik:

$$\eta(\boldsymbol{X}) = \eta_0 + \sum_{k=1}^{d} \eta_k(x_k) + \sum_{k<l} \eta_{k,l}(x_k, x_l) + \cdots + \eta_{1,\dots,d}(x_1, \dots, x_d) \qquad (16)$$

Ahol η_0 konstans tag, η_k-k a főhatások, $\eta_{k,l}$-ek a kétirányú interakciók, és így tovább egészen a d-tényezős interakcióig (Gu, 2013). Ezzel a COSSO módszer egy általánosabb nem-lineáris modellt feltételez, mint a GAM, hiszen az additív főhatások mellett nem-lineáris interakciókat is vizsgál legfeljebb d tényezőig.

Az η függvény komponenseire annyi kikötést teszünk, hogy egy $\mathcal{H}$ RMHT elemei legyenek. A fő hatásokat tartalmazó $\eta_k(x_k)$ függvényeket a

$$W^{(k)}[0,1] = \{f: f(t), f'(t)\ abszolút\ folytonos, f''(t) \in L_2[0,1]\}$$

másodrendű Szoboljev-térben becsüljük meg, ha x_k folytonos változó. A teret a (17) skalárszorzattal látjuk el.

$$\langle f, g \rangle = \int_0^1 f(t)\,dt \int_0^1 g(t)dt + \int_0^1 f'(t)\,dt \int_0^1 g'(t)dt + \int_0^1 f''(t)\,g''(t)dt \quad (17)$$

A (17) skalárszorzattal $W^{(k)}[0,1]$ egy RMHT, mégpedig a következő magfüggvénnyel:

$$K(s,t) = 1 + k_1(s)k_1(t) + k_2(s)k_2(t) - k_4(|s-t|).$$

Ahol,

$$k_1(s) = s - 0{,}5$$

$$k_2(s) = [k_1^2(s) - 1/12]/2$$

$$k_4(s) = \left[k_1^4(s) - \frac{k_1^2(s)}{2} + \frac{7}{24}\right]/24$$

Ez Wahba (1990) (10.2.4) összefüggésének egy speciális esete $m = 2$-re.

Vegyük észre a fentiek alapján, hogy $W^{(k)}$ felírható két ortogonális altér direkt összegeként. $W^{(k)} = 1^{(k)} \oplus W_1^{(k)}$, ahol $1^{(k)}$ az úgynevezett „átlag" tér, és $W_1^{(k)}$ az úgynevezett „kontraszt" tér, amit a következő magfüggvény generál a Moore-Aronszajn tétel szerint:

$$K_1(s,t) = K(s,t) - 1$$

Ha x_k kategorikus változó, akkor, ami a $\{1, \dots, L\}$véges elemű halmazból veszi fel a lehetséges értékeit, akkor $\eta_k(x_k)$ függvény valójában egy L hosszú vektor, és a függvény kiértékelése nem más, mint egyszerű koordináta kiemelés. Ekkor pedig $W^{(k)}$ dekompozíciója $1^{(k)} \oplus W_1^{(k)}$ alakban a következőképpen alakul:

$$1^{(k)} = \{f : f(1) = \cdots = f(L)\}$$

$$W_1^{(k)} = \{f : (1) + \cdots + f(L) = 0\}$$

A Moore-Aronszajn tételnek megfelelő magfüggvénnyel társítva:

$$K_1(s,t) = L\mathbb{I}_{(s=t)} - 1, \qquad s,t \in \{1, \dots, L\}$$

A (16) interakcióinak becslését a megfelelő egyváltozós függvények úgynevezett tenzorszorzatos[1] terében végezzük el. Egy ilyen tenzorszorzatos tér reprodukáló magfüggvénye, egyszerűen a tenorszorzat tényezőit adó két egyedi tér magfüggvényeinek szorzata. Tehát, pl. a $W_1^{(k)} \otimes W_1^{(l)}$ tér magfüggvénye a $K_1\left(s^{(k)}, t^{(k)}\right)K_1\left(s^{(l)}, t^{(l)}\right)$ szorzat eredménye. Ez az ábrázolási mód megkönnyíti majd a magyarázóváltozóink nem-lineáris hatásainak becslését is a COSSO módszer során magas dimenziójú magyarázóváltozó-vektor esetében (Lin – Zhang, 2006).

Az $\eta(\boldsymbol{X})$ (16) dekomozíciójának megfelelő tenzorszorzatokkal előállított, és a becsléshez használt teljes metrikus tér az alábbi formát ölti végül:

[1] A tenzorszorzatot tulajdonképpen diadikus szorzatként értelmezve.

$$\otimes_{k=1}^{d} W^{(k)} = \{1\} \oplus_{k=1}^{d} W_1^{(k)} \oplus_{k<l} \left\{ W_1^{(k)} \otimes W_1^{(l)} \right\} \oplus \dots$$

Magas d dimenziószámú magyarázóváltozó-vektor esetében (16)-ot gyakran korlátozzuk a csak kétirányú interakciókra, azaz csak kereszthatásokat tartalmazó többváltozós függvények tereire, mivel különben egy általános $\eta(x_1, \dots, x_d)$ függvény megbecslése az eddig ismertetett struktúrában is igen nagy mintaméretet igényel még relatíve kis d–k esetében is. Tehát, (16)-ot (18)-ra redukáljuk a legtöbb gyakorlati alkalmazás esetében (Huang et al., 2000).

$$\eta(\boldsymbol{X}) = \eta_0 + \sum_{k=1}^{d} \eta_k(x_k) + \sum_{k<l} \eta_{k,l}(x_k, x_l) \tag{18}$$

Általánosságban végül a (16)-nak (18)-ra redukált verziója a (19) formalizmussal írható fel m darab lehetséges magyarázóváltozó esetében:

$$\eta(\boldsymbol{X}) = \eta_0 + \sum_{\alpha=1}^{m} \eta_\alpha(\boldsymbol{X}) \tag{19}$$

A (19) felbontás pedig a következő m darab ortogonális altér direkt összegében értelmezhető:

$$\mathcal{H} = \{1\} \oplus_{\alpha=1}^{m} \mathcal{H}_\alpha$$

Ebben az esetben a korábbi jelöléseink alapján $K_\alpha(s,t)$ jelölje a $\mathcal{H}_\alpha$ altér reprodukciós kernelfüggvényét. Ebből adódóan $\mathcal{H}$ reprodukáló magfüggvénye a következőképpen áll elő (19)-ből:

$$K_{\mathcal{H}}(s,t) = 1 + \sum_{\alpha=1}^{m} K_\alpha(s,t)$$

A nem-lineáris regressziós modellek általános leírásának és funkcionális ANOVA elven történő reprezentációs elveinek áttekintése után felírjuk a nem-lineáris magyarázóváltozók közötti szelekcióra képes COSSO módszer optimalizálási feladatát.

A COSSO-módszer használata során a (20) feladatot oldjuk meg. A módszert Lin – Zhang (2006) és Storlie et al. (2011) alapján ismertetjük.

$$\min_{\eta \in \mathcal{H}} \ell\big(\eta(\boldsymbol{X})\big) + \tau \sum_{\alpha=1}^{m} \|P^\alpha \eta\| \tag{20}$$

Ahol $\mathcal{H} = \{1\} \oplus_{\alpha=1}^{m} \mathcal{H}_\alpha$ a megfelelő reprodukáló magú Hilbert-tér, és $\ell\big(\eta(\boldsymbol{X})\big)$ a modellt a $\eta(\cdot)$ függvény adott paraméterei mellett jellemző negatív log-likelihood (ekkor $\hat{Y} = \eta(\boldsymbol{X})$ módon adunk becslést az eredményváltozó feltételes várható értékére). Továbbá, $P^\alpha \eta$ az η többváltozós függvény projekciója $\mathcal{H}_\alpha$-ra. $\|\cdot\|$ pedig a $\mathcal{H}$ RMHT normája. $\tau \in (0,1)$ a Lasso módszerben használt λ-hoz hasonló állítható paraméter. Az alapelv tehát a GLM esetén változószelekcióra alkalmazott Lasso módszerhez hasonló. A becsülendő paraméterek

L_1 normájával büntetve (regularizálva) az $\ell\big(\eta(\boldsymbol{X})\big)$ negatív log-likelihood függvényt a minimalizálás során elérhető, hogy a becslésünk MSE értéke csökkenjen az egyszerű negatív log-likelihood minimalizáláshoz képest, és $\exists P^{\alpha}\eta \equiv 0$, tehát változószelekciót is végrehajtunk. Sőt, az is könnyen megmutatható, hogy a Lasso módszer a COSSO módszer egy speciális esete. Hiszen, ha a modellünk lineáris, akkor $\eta(\boldsymbol{X}) = \beta_0 + \sum_{j=1}^{m}\beta_j X_j$, és a modell $\mathcal{H}$ tere a következő alakban adható meg, a L_2 skalárszorzattal ellátva:

$$\{1\} \oplus \left\{X_1 - \frac{1}{2}\right\} \oplus \ldots \oplus \left\{X_m - \frac{1}{2}\right\}$$

A COSSO módszer $\sum_{\alpha=1}^{m}\|P^{\alpha}\eta\|$ büntetőtagja ebben az esetben $12^{-1/2}\sum_{j=1}^{m}|\beta_j|$-ként írható fel, ami pontosan a Lasso módszer klasszikus büntetőtagja: a lineáris együtthatóvektor L_1 normája. A becslési eljárásunk során (20) feladat megoldásához (20) egy alternatív felírásából, (21)-ből kell kiindulnunk.

$$\begin{gathered}\min_{\eta,\theta}\ell\big(\eta(\boldsymbol{X})\big) + \lambda_0\sum_{\alpha=1}^{m}\theta_{\alpha}^{-1}\|P^{\alpha}\eta\|^2 \\ fh\colon \sum_{\alpha=1}^{m}\theta_{\alpha} \le M, \theta_{\alpha} \ge 0\end{gathered} \tag{21}$$

Ahol $\theta = \big(\theta_1, \theta_2, \ldots, \theta_p\big)^T$ nemnegatív slack változók és λ_0 egy fix paraméter. M pedig egy finomhangolást végző, állítható paraméter, ami egy az egyben megfeleltethető a (20) feladat τ paraméterének. (21)-ben a θ-ra megadott korlát teszi lehetővé, hogy bizonyos θ_{α} slack változók becslése pontosan nulla legyen, ezzel azonosan nulla függvénykomponenseket eredményezve $\eta(\boldsymbol{X})$-ben.

Felhasználva $\mathcal{H}$ tér altereinek magfüggvényeit (K_{α}) (21) megoldása

$$\eta(\boldsymbol{X}) = b + \sum_{i=1}^{n}K_{\theta}\big(\boldsymbol{X}, \boldsymbol{X^{(i)}}\big)\,c_i$$

alakban kereshető, ahol $K_{\theta} = \sum_{\alpha=1}^{m}\theta_{\alpha}K_{\alpha}$. Az η függvény definiálatlansága miatt a b konstanst vagy egy konstans függvénynek kezeljük, ami a $\hat{Y} = \eta(\boldsymbol{X})$ becslés torzítását szabályozza, vagy ekvivalensen $b = 0$-nak tekintjük. Tehát, (21) megoldását az alábbi (22) alakban kereshetjük.

$$\eta(\boldsymbol{X}) = \sum_{i=1}^{n}\sum_{\alpha=1}^{m}\theta_{\alpha}K_{\alpha}\big(\boldsymbol{X}, \boldsymbol{X^{(i)}}\big)\,c_i \tag{22}$$

Nagyméretű adathalmazok esetében Lin – Zhang (2006) szerint érdemes a becslést különböző $\big(X_1^*, \ldots, X_q^*\big) \subseteq (X_1, \ldots, X_m)(q < m)$ részhalmazok által kifeszített alterein megbecsülni $\mathcal{H}$-nak. Ebben az esetben (21) megoldása $\eta(\boldsymbol{X}) = \sum_{i=1}^{n}\sum_{\alpha=1}^{m}\theta_{\alpha}K_{\alpha}\big(\boldsymbol{X}, \boldsymbol{X^{*(i)}}\big)\,c_i$ alakban keresendő.

Az alkalmazásaink során az $\boldsymbol{X}^*$ oszlopait adó részhalmaz kiválasztása véletlen mintavételezésekkel történik.

Habár (21) kifejezés minimalizálása (22) alapján történhet a θ és c vektorok értékei szerint szimultán is, Lin – Zhang (2006) szerint kifizetődőbb egy úgynevezett alternálva optimalizáló algoritmust használni, ami az iterációi során a Newton-Raphson eljárást felváltva alkalmazza θ-ra és c-re, a másik vektor előző iterációból kapott rögzített értékei mellett. Természetesen, az algoritmus M egy rögzített értéke mellett futtatandó. M optimális meghatározása a Lasso módszer λ paraméterének meghatározásához hasonlóan történik. Valamelyik tetszőleges információs kritérium minimalizálásával választjuk ki M-et 10-szeres keresztvalidáció végrehajtásával. Azt a M-et választjuk, ami minimalizálja keresztvalidált *AIC*-t. Ezzel az M választási technikával a Lasso módszerhez hasonló jóságú információs kritérium értékeket érhetünk el a hagyományosabb szelekciós módszerekhez képest.

A COSSO módszer alkalmazásához az értekezésben a *cosso* nevű R csomagot használjuk (Zhang – Lin, 2013).

6.1.3. Büntetőtagos thin plate spline illesztés

Amennyiben megtartjuk a (3)-ban megadott klasszikus GAM keretet, és az interakciók nem képezik részét a változószelekciónak, akkor az f_j függvények reprezentációjára thin plate spline-ok alkalmazhatók. Marra – Wood (2011) alapján egy a COSSO algoritmus elvéhez hasonló változószelekciós eljárás megadható thin plate spline függvények alkalmazásával is.

A spline illesztés alap minimalizálási feladata (13) formában felírva nem teszi lehetővé, hogy egy X_j magyarázóváltozónak $f_j \equiv 0$ függvényt válasszunk, mivel a legrosszabb esetben is a $\lambda_j \tilde{\Psi}^{\boldsymbol{T}} \boldsymbol{S_j} \tilde{\Psi}$ büntetőtag csak teljesen lineárissá teszi f_j-t, nem veszi azonosan nullának, azaz nem hagyja el X_j-t a modellből.

Ahhoz, hogy lehetséges legyen $f_j \equiv 0$ függvények illesztése is szükséges egy új büntetőtag (regularizáció) bevezetése a (13) feladatba, ami az $\boldsymbol{S_j}$ 0 sajátértékeihez tartozó sajátvektorokra alkalmaz célfüggvény növelést. Ezzel a célfüggvény értékét az is rontani tudja, ha „feleslegesen” lineáris függvényt becsülünk f_j-nek a GAM modellben, egy azonosan 0 függvény heylett.

Technikailag a GAM illesztési feladat (13)-ról (23)-ra módosul.

$$\min_{\tilde{\Psi}} \left\| \boldsymbol{W z} - \boldsymbol{W A} \tilde{\Psi} \right\|^2 + \sum_{j=1}^{p} \lambda_j \tilde{\Psi}^{\boldsymbol{T}} \boldsymbol{S_j} \tilde{\Psi} + \lambda_j^* \tilde{\Psi}^{\boldsymbol{T}} \boldsymbol{S_j^*} \tilde{\Psi} \qquad (23)$$

Ahol $\boldsymbol{S_j^* = U_j^* U_j^{*T}}$, amiben $\boldsymbol{U_j^*}$ oszlopai az eredeti $\boldsymbol{S_j}$ mátrix nulla sajátértékeihez tartozó sajátvektorok.

A nulla sajátértékekre bevezetett büntetést extra büntetőtag nélkül, az eredeti büntetőtag átírásával is meg lehet oldani. Az eredetei (13) feladatban $\boldsymbol{S_j}$-k helyett egy $\boldsymbol{\widetilde{S_j} = U_j D_j^* U_j^T}$ spektrálfelbontással adott mátrixot alkalmazunk, ami csak abban különbözik az eredeti $\boldsymbol{S_j = U_j D_j^* U_j^T}$ felbontástól, hogy $\boldsymbol{D_j}$-ben a nulla értékeket egy kellően kicsi ε értékkel írjuk felül $\boldsymbol{D_j^*}$-ban, ezzel pozitív büntetőtagot rendelve a nulla sajátértékekhez tartozó sajátvektorokhoz is a GAM illesztési feladatban. Ezzel a kettős büntetés alkalmazásához hasonlóan lehetővé válik $f_j \equiv 0$ függvények becslése is X_j-re.

A thin plate spline-ok alkalmazásával megvalósítható regularizációs elvű változószelekciót R nyelven az *mgcv* csomag segítségével tudjuk alkalmazni (Wood, 2017).

6.1.4. Nemnegatív Garotte módszer

Cantoni et al. (2011) közvetlen módon kísérli meg a lineáris modellek Lasso szelekcióját alkalmazni a GAM modellekre is. A nemnegatív garrote alapötlete, hogy ha a (3)-al adott GAM modellben az $f_j(X_j)$ tagokat már megbecsültük b-spline illesztéssel, akkor ezek a már ismert $f_j(X_j)$ értékek egy lineáris modellben, mint magyarázóváltozók kezelhetők. Emiatt, ebben az új lineáris modellben az együtthatók Lasso becslésével változószelekciót lehet végrehajtani, hiszen az eredményváltozó szempontjából nem releváns új $z_j \coloneqq f_j(X_j)$ magyarázóváltozók lineáris együtthatóit a Lasso képes 0-nak becsülni a lineáris együtthatóvektor L_1 normájára adott büntetőtaggal a célfüggvényben. Formálisan, a nemnegatív garrote a (24) feladatot oldja meg a klasszikus LARS algoritmussal.

$$\min_C \ell(\boldsymbol{F(X)}C) + \lambda \|C\|_1 \qquad (24)$$

Ahol $C = [c_0, c_1, \dots, c_m]$ és $\boldsymbol{F(X)} = \{1_n, f_1(X_1), \dots, f_m(X_m)\}$-k már eleve ismert, megbecsült α_i paraméterekkel rendelkező b-spline függvények. $\ell(\boldsymbol{F(X)}C)$ az eredményváltozó negatív log-likelihood függvénye, ha annak feltételes várható értékét $\hat{Y} = \boldsymbol{F(X)}C$ módon becsüljük.

Az algoritmus R implementációjához saját fejlesztésű szkriptet alkalmazunk, amelyben (24) megoldását a *ncvreg* csomag segítségével végezzük el (Breheny – Huang, 2011).

6.2. Stepwise módszer GAM modellek esetében

A statisztikusok körében legkorábban elterjedt modellszelekciós heurisztikák a Stepwise algoritmusok voltak. Napjainkban a legfontosabb hazai és külföldi statisztika és ökonometria témájú tankönyvekben alapvető tananyagként szerepelnek. Néhány példa: Kovács (2006),

Ramanathan (2002) valamint Hastie et al. (2011). A Stepwise algoritmusoknak két alapvető változata létezik: Forward és Backward szelekció.

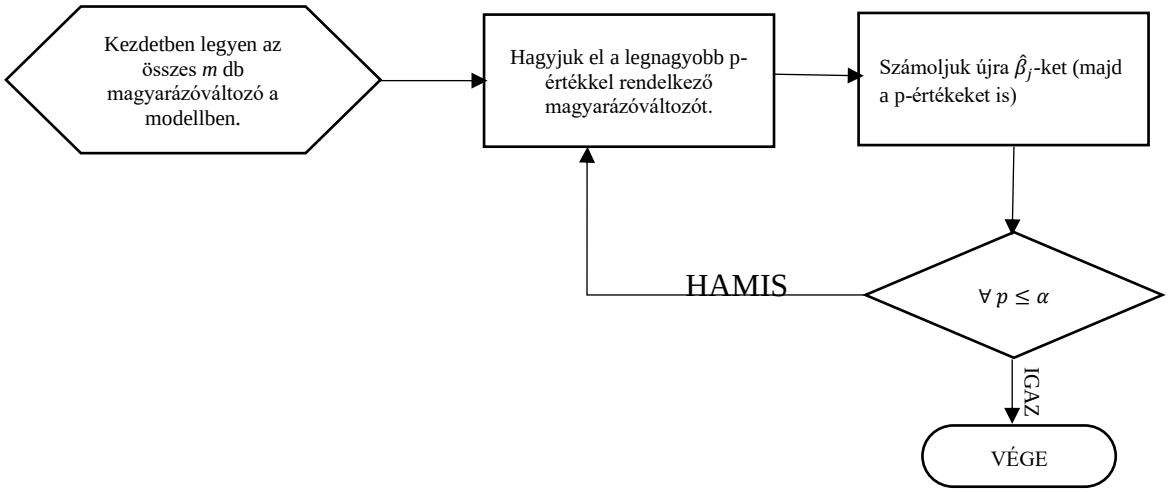
GLM esetén az algoritmusok legegyszerűbb variánsai a j-edik magyarázóváltozóhoz tartozó p-értéket használják fel a változószelekció során. A p-érték jelen esetben azt az empirikus szignifikancia szintet jelöli, melynél $\tilde{X}_j$ változó már éppen szignifikánsnak tekinthető. A p-értéket az $\tilde{X}_j$ változón végrehajtott parciális Wald-próba próbafüggvényéből számoljuk. Számértéke nem más, mint a próbastatisztikához igaz nullhipotézis esetén tartozó eloszlásfüggvény értéke. A használt Wald-próba szerint a $\left(\frac{\hat{\beta}_j}{SE(\hat{\beta}_j)}\right)^2$ próbastatisztika igaz nullhipotézis (tehát a változó inszignifikanciája esetén) χ_1^2 eloszlású.

Tehát, ha a j-edik változó szignifikanciáját egy adott α szignifikancia szinten vizsgáljuk, akkor az alábbi következtetések vonhatók le a p-érték alapján:

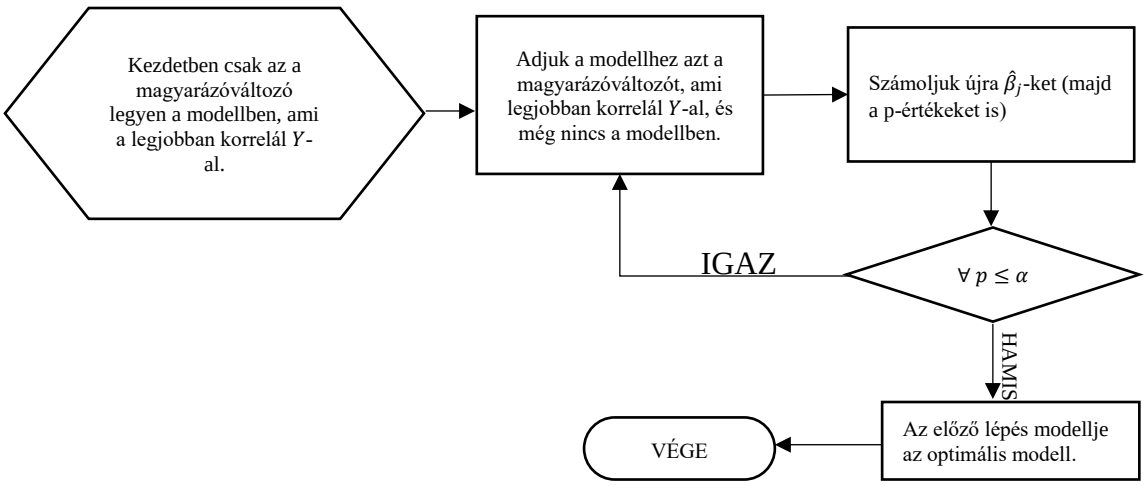
- ha $\tilde{X}_j$ változót vizsgálva $p > \alpha$, akkor $\tilde{X}_j$ nem szignifikáns magyarázóváltozó a modellben.
- ha $\tilde{X}_j$ változót vizsgálva $p \leq \alpha$, akkor $\tilde{X}_j$ szignifikáns magyarázóváltozó a modellben.

Tehát, minél kisebb az $\tilde{X}_j$-hez tartozó p-érték, $\tilde{X}_j$ annál szignifikánsabb. Illetve, minél nagyobb az $\tilde{X}_j$-hez tartozó p-érték, $\tilde{X}_j$ annál kevésbé szignifikáns. (Kovács, 2006)

A p-érték fogalmát felhasználva már leírhatjuk a Forward, és Backward algoritmusok folyamatábráját a 3. és 4. ábrák szerint.



3. ábra: A Backward elimináció folyamatábrája. Forrás: saját szerkesztés.



4. ábra: A Forward elimináció folyamatábrája. Forrás: saját szerkesztés.

A folyamatábrák alapján láthatjuk, hogy a Forward és a Backward algoritmusok elsődleges célfüggvénye az, hogy minden magyarázóváltozó szignifikáns legyen a modellben.

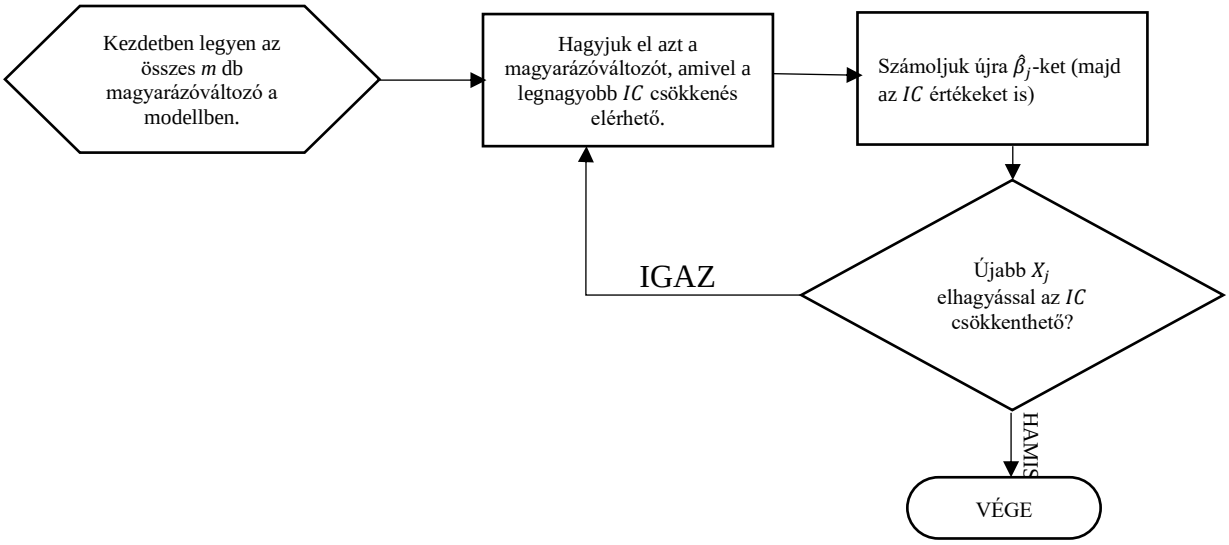
A folyamatábrák áttekintése után megállapíthatjuk azt is, hogy a Stepwise heurisztikák futása során az egyes regressziós modellek (a magyarázóváltozók adott részhalmazából képzett lehetséges megoldások) mindig csupán egy darab változóban különböznek (egy változót adunk hozzá vagy vonunk el). Tehát, a keresési tér nagy része feltérképezetlen maradhat, és az algoritmusunk lokális optimum felé konvergálhat a harmadik fejezetben ismertetett modellszelekciós célfüggvények ($AIC, SBC, HQC, \bar{R}^2$) értékét vizsgálva.

Ráadásul, a magyarázóváltozók p-értékének vizsgálata kimondottan félrevezető lehet multikollinearitás jelenléte esetén. Ugyanis, megmutatható, hogy ekkor az együtthatók standard hibái ($SE(\hat{\beta}_j)$-k) pont VIF_j-szeresükre nőnek. Ezzel pedig sok releváns magyarázóváltozó p-értéke irreálisan magasra nőhet, és az algoritmus szelektálja őket a modellből szimplán a multikollinearitás miatt.

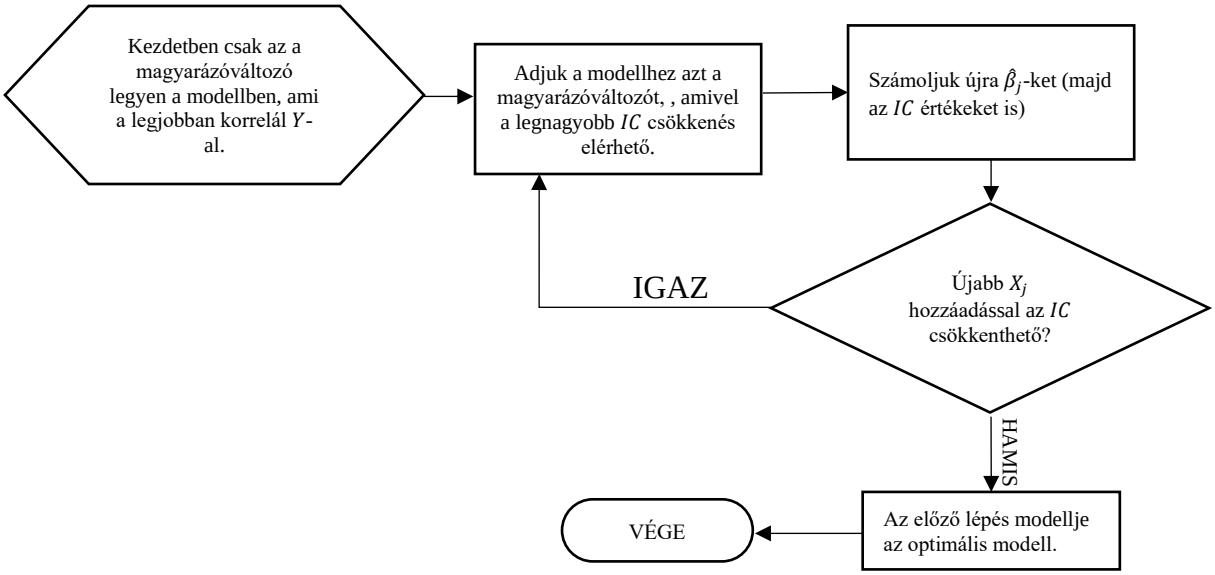
A klasszikus stepwise szelekciós algoritmust lehet alkalmazni GAM modellben, csupán ehhez definiálni szükséges, hogy mit jelent GAM keretben az, hogy egy X_j magyarázóváltozó nem szignifikáns. Amennyiben f_j-ket thin plate spline-ok segítségével modellezzük, akkor nem szignifikánsnak mondhatjuk X_j-t, ha a következő nullhipotézist nem tudjuk elutasítani: $\Psi_{k_j} = \mathbf{0}$. A nullhipotézis tesztelésére Marra – Wood (2011) ad meg egy χ^2-próbát.

A próbát az R *mgcv* csomagja segítségével tudjuk alkalmazni.

A stepwise algoritmusoknak létezik egy kifinomultabb változata is, amely nem a p-érték, hanem egy előre megadott információs kritérium (IC) alapján szelektálja a magyarázóváltozókat. Ezek az algoritmusok az 5. és 6. ábrákon látható módokon módosítják a 3. és 4. ábrákon adott Stepwise algoritmusok folyamatábráit.



5. ábra: Az információs kritérium alapú Backward elimináció folyamatábrája. Forrás: saját szerkesztés.



6. ábra: Az információs kritérium alapú Forward elimináció folyamatábrája. Forrás: saját szerkesztés.

Az információs kritériumok alkalmazásával a p-értékek helyett, az újabb Stepwise algoritmusok már mentesülnek a multikollinearitás okozta torzítottan magas p-értékek káros hatásai alól. Azonban az algoritmus lokális keresési elvén nem változtatnak, így a keresési tér nagy része továbbra is feltérképezetlen maradhat. Ebből kifolyólag viszont előnyük lehet a gyors futásidő.

Az információs kritérium alapján szelektáló Stepwise algoritmusok R nyelven a *MASS* csomag *stepAIC* függvénye segítségével alkalmazhatók GLM-re (Venables – Ripley, 2002). GAM keretben a *gam* csomag *step.Gam* függvénye implementálja az algoritmust (Hastie, 2018).

6.3. GAMBoost algoritmus

A likelihood alapú GAM boosting változószelekciós algoritmus Tutz et al. (2006) és Binder – Tutz (2008)-ban jelenik meg először. A végső, büntetőtagos verzió Schmid – Hothorn

(2008)-ban került publikálásra. Az alapötlet egy Schapire (1990) által osztályozó modellekre (pl. döntési fákra) kifejlesztett algoritmus alkalmazása GAM környezetben. Az ötlet lényege, hogy egy osztályozó modellt a tantóhalmazon iteratív módon tanítunk be, minden iterációban eltérő súlyokat helyezve a tanítóminta megfigyeléseire. A végső becslés a célváltozó besorolására a különböző tanítósúlyokkal vett modellek többségi szavazata lesz.

GAM modellek alkalmazásában a likelihood alapú boosting eljárás harmadrendűrendű b-spline függvényeket ($S_3(x_{ji}) = \sum_k \alpha_k B_{k,3}(x_{ji})$) alkalmaz tanuló algoritmusként egy X_j magyarázóváltozó értelmezési tartományán, egyenletesen elhelyezett k_i csomópontokkal. A boosting eljárás a magyarázóváltozókra illesztett spline függvényt büntetőtagos maximum likelihood elven becsüli. Először bevezetjük az $l(\cdot)$ jelölést az eredményváltozó eloszlásának loglikelihood függvényére (hogy ne keverjük össze a negatív log-likelihoodot jelölő $\ell(\cdot)$-el). Majd ennek segítségével megadjuk a minta loglikelihood függvényét, ahol a meghatározandó paraméterek a spline függvények α_i együtthatói: $l(\boldsymbol{\alpha}^{(j)}) = \sum_{i=1}^{n} l\left(y_i, S_3(x_{ji})\right)$. Ahol $l\left(y_i, S_3(x_{ji})\right)$ a minta i-edik elemén az eredményváltozó feltételes loglikelihood függvénye, ismert $\mathbb{E}(y_i) = h^{-1}(S_3(x_{ji}))$ feltétel mellett. $\boldsymbol{\alpha}^{(j)} \in \mathbb{R}^3$ pedig a j-edik magyarázóváltozóra illesztett harmadrendű b-spline függvény együtthatóvektora. Amennyiben a célváltozó normális eloszlású, $l(\boldsymbol{\alpha}^{(j)})$ épp a négyzetes euklideszi távolság.

Az algoritmus a spline becsléseknél (25)-t maximalizálja, ahol $\boldsymbol{\Lambda}$ úgy kerül megválasztásra, hogy a kifejezés 2. tagja a spline együtthatók elsőrendű differenciáira vonatkozó négyzetes büntetőtag ($\sum_i(\alpha_{i+1} - \alpha_i)^2$) legyen.

$$l_p(\boldsymbol{\alpha}^{(j)}) = l(\boldsymbol{\alpha}^{(j)}) - \frac{\lambda}{2}\boldsymbol{\alpha}^{(j)\boldsymbol{T}}\boldsymbol{\Lambda}\boldsymbol{\alpha}^{(j)} \qquad (25)$$

λ-t kellően nagyra kell választani, mivel az algoritmusnak kifejezetten célja, hogy egy spline becslés során a függvény illeszkedése ne legyen igazán pontos. A büntetőtag tehát azt biztosítja, hogy gyengén illeszkedő spline-okat kapjunk, nem pedig az a célja, hogy a b-spline rendjének választását szabályozza, mint a thin plate spline-ok esetében.

Az algoritmus lépéseit a következő pszeudo algoritmikus leírás segítségével ismertetjük.

0. lépés: Konstans modellt illesztünk Y-ra: $\hat{Y}_{(0)} = f_{(0)}(x) = \alpha_0$

ciklus $l = 1$-től L-ig

ciklus $j = 1$-től m-ig

1. lépés: X_j magyarázóváltozóra nézve meghatározzuk a legjobban illeszkedő spline függvényt a $Y - \hat{Y}_{(l-1)}$ célváltozóra illesztve (25) maximalizálásával.

2. lépés: Legyen ez az illesztett függvény $f_{j,(l)}$, és ezzel adjuk meg X_j magyarázóváltozóra a $\hat{Y}_{(l),j} = \hat{Y}_{(l-1)} + f_{j,(l)}(X_j)$ becslést

3. lépés: Nézzük meg $\hat{Y}_{(l),j}$ becslés devianciáját Y-tól: $D(\hat{Y}_{(l),j})$ jelöli a $\hat{Y}_{(l),j}$ becslés devianciáját a 4. fejezetben, a McFadden-féle pszeudo R-négyzet mutató definiálása során ismertetett deviancia fogalom szerint.

ciklus vége

4. lépés: keressük meg, hogy melyik $f_{j,(l)}$ függvény okozza a legnagyobb mértékű javulást az $l-1$-edik iterációban kapott modell devianciájához képest: $s = \underset{j}{\operatorname{argmax}}\{D(\hat{Y}_{(l-1)}) - D(\hat{Y}_{(l),j})\}$.

5. lépés: Legyen $\hat{Y}_{(l)} = \hat{Y}_{(l-1)} + f_{s,(l)}(X_s)$

ciklus vége

Mivel minden iterációban frissítjük a GAM modell $\hat{Y}$ becslését, így fontos elkerülni, hogy rögtön az első iterációkban tökéletes illesztést találjunk, mivel akkor a többi iteráció feleslegessé válik, és egy esetleges lokális optimumból nem tudja kimozdítani a keresést. Ezért szükséges a magas λ választása.

A külső ciklusban az iterációkat egy megadott L lépésszámig ismételjük. L megválasztására Tutz et al. (2006) azt javasolja, hogy $L \in [50,200]$, a végső értéket úgy adjuk meg, hogy a megadott intervallumban lévő L-ekkel dolgozva megnézzük, hogy a végső modellben mekkora az AIC értéke 10-szeres keresztvalidációval számolva, és azt az L-t válasszuk, ami minimalizálja AIC-t. Hasonló módszerrel tudjuk az optimális λ értéket is megválasztani a büntetőtagos maximum likelihood formulákhoz.

Mint láthatjuk, az algoritmus során nem feltétlenül kerül minden j kiválasztásra az s meghatározásakor, így a boosting módszerrel illesztett GAM modellben változószelekciót is végrehajtunk.

Az algoritmus R nyelven a *GAMBoost* csomag segítségével alkalmazható (Binder – Tutz, 2008).

6.4. Módosított backfitting eljárás

A Belitz – Lang (2008)-ban közölt módosított backfitting eljárás algoritmusa nagyon sok pontjában a GAMBoost módszerhez hasonlít, sőt konkrétan, meg is lehet mutatni, hogy két egyszerű módosítással átalakítható azzá.

Az algoritmus alapötlete, hogy a GAMBoost eljárásban megismert büntetőtagos maximum likelihood elvű spline illesztést minden X_j magyarázóváltozóra M_j db $\lambda_{j1} > \cdots > \lambda_{jM_j}$

paraméter mellett elvégezzük az iterációkban, és minden X_j esetében a legjobb becslési pontosságot biztosító λ_{js}-val ($s \in [1, M_j]$) vett spline illesztést tekintjük az aktuális becslésnek Y-ra X_j függvényében. Fontos megjegyezni, hogy λ_{js} értékeket úgy kell megválasztani, hogy a $df_{js} = s$ egyenlőség fennáljon. df_{js} a j-edik magyarázóváltozóra λ_{js} büntetőtag szorzóval illesztett spline szabadságfoka, ami nem más, mint a $\boldsymbol{Z_j}\left(\boldsymbol{Z_j^T Z_j} + \lambda_j \boldsymbol{\Lambda_j}\right)^{-1}\boldsymbol{Z_j^T}$ mátrix nyoma. Ahol $\boldsymbol{Z_j}$ az X_j magyarázóváltozó $B_{i,3}$ bázisfüggvény értékeinek mátrixa, $\boldsymbol{\Lambda_j}$ pedig a (25)-ben szereplő $\boldsymbol{\Lambda}$ mátrix a j-edik magyarázóváltozóra.

Az algoritmus lépéseit a következő pszeudo algoritmikus leírás segítségével ismertetjük.

0. lépés: $l = 0$-ban konstans modellt illesztünk Y-ra, minden X_j magyarázóváltozó esetében: $\hat{Y}_{j(0)} = f_{j(0)}(x) = \alpha_0$.

amíg $\hat{Y}_{(l)} \neq \hat{Y}_{(l-1)}$

ciklus $j = 1$-től m-ig

1. lépés: X_j változóra meghatározzuk a $f_{j,s,(l)}$ splinefüggvényeket a λ_{js} értékek mellett. Az illesztés célváltozója $Y - \hat{Y}_{[j](l-1)}$, ahol $\hat{Y}_{[j](l-1)}$ az előző iterációból kapott becslés Y-ra az X_j magyarázóváltozóhoz tartozó tag elhagyásával a GAM modell egyenletéből.

2. lépés: Megadjuk X_j esetén a becslést Y-ra az l-edik iterációban illesztett spline függvényekkel: $\hat{Y}_{j,s,(l)} = \hat{Y}_{[j](l-1)} + f_{j,s,(l)}(X_j)$.

3. lépés: Kiválasztjuk X_j-re a legkisebb $AIC\left(\hat{Y}_{j,s,(l)}\right) = D\left(\hat{Y}_{j,s,(l)}\right) + 2df_{js}$ értéket szolgáltató λ_{js}-t: $q_j = \underset{s}{\operatorname{argmin}}\, AIC\left(\hat{Y}_{j,s,(l)}\right)$.

ciklus vége

4. lépés: Az l-edik iteráció végső becslése Y-ra $\hat{Y}_{(l)} = \sum_j \hat{Y}_{j,q_j,(l)}$ lesz.

5. lépés: $l = l + 1$

A módosított backfitting illesztési eljárás során az X_j-re adott különböző függvényformák között egy olyan definícióval adott AIC kritériummal választunk, ami bünteti a felesleges szabadságfok növekedését az illesztett $f_{j,s,(l)}$-nek. Emiatt, változószelekció is megvalósulhat, ha valamely változó tekintetében AIC alapján nem éri meg a konstans modell keretékből kilépni még a lineáris esetbe ($df_{j1} = 1$) sem.

Érdemes észrevenni, hogy ha csak egy λ_j-t használunk minden X_j magyarázóváltozóra, a spline illesztés célváltozóját $Y - \hat{Y}_{(l-1)}$-nak választjuk $Y - \hat{Y}_{[j](l-1)}$ helyett, és egy l iterációban csak egy X_j magyarázóváltozónak frissítjük az $f_{j,(l)}$ függvényét a devianciajavítás alapján, akkor

a GAMBoost eljárást kapjuk vissza. Ezen módosítási pontokból pedig látszik, hogy a GAMBoost eljárásnál a módosított backfitting algoritmus numerikusan gyorsabban megoldható, mivel ebben az eljárásban minden l iterációban minden X_j magyarázóváltozóra adott $f_{j,(l)}$ függvény becslést felülvizsgáljuk, így az algoritmus implementálása során ez a rész minden X_j-re párhuzamosan kiszámítható. Azaz, nem feltétlenül csak egy X_j magyarázóváltozó nem-lineáris hatásának becslésén módosítunk egy iterációban, hanem adott esetben többén is. Ez a különbség a két algoritmus között nagyon látványos lesz már a 9. fejezetben bemutatott kisebb méretű vizsgált adatbázison is.

A módosított backfitting eljárást R nyelven a *R2BayesX* csomag implementálja (Umaluf et al., 2015).

6.5. A CFS algoritmus

Az 6.1., 6.2., 6.3. és 6.4. alfejezetek során megfigyelhetjük, hogy az ezekben ismertetett algoritmusok a változószelekció során kísérletet sem tesznek a concurvity jelenség kontrollálására.

Az 6.6. és 6.7. alfejezetekben két olyan korszerű algoritmust ismertetünk, ami nem-lineáris változószelekció esetén a célfüggvényben tekintettel van arra is, hogy a kiválasztott magyarázóváltozók között ne álljon fenn a végső modell értelmezhetőségét csorbító nem-lineáris összefüggés. Mindkét ismertetett algoritmus a Hall (1999)-féle lineáris korreláció alapú változószelekciós algoritmus (Correlation based Feature Selection, CFS a továbbiakban) általánosításának tekinthető nem-lineáris esetre. Emiatt először ebben a fejezetben a CFS algoritmust ismertetjük.

Az alapötlet a parszimónia elvének egy leegyszerűsített megfogalmazására vezethető vissza: egy magyarázóváltozó akkor igazán informatív az eredményváltozó szempontjából, ha azzal korrelál, ám más magyarázóváltozóval a modellben nem. Ezek alapján adja Hall a (26) képlettel definiált mutatót.

$$\frac{p\overline{r_{zl}}}{\sqrt{p + p(p-1)\overline{r_{ll}}}} \tag{26}$$

Ahol $\overline{r_{zl}}$ az átlagos korreláció az adott modellben szereplő magyarázóváltozók és az eredményváltozók között, $\overline{r_{ll}}$ az átlagos lineáris korreláció az adott modellben szereplő magyarázóváltozók között és p az adott modellben lévő magyarázóváltozók száma. Hall (1999) szerint (26) maximalizálásával megtalálható a magyarázóváltozók egy olyan részhalmaza, ami a parszimónia elvének megfelelő osztályozó modellt biztosítja. Természetesen, attól, hogy a (26) érték egy modellszelekciós probléma célfüggvénye a probléma továbbra is NP-nehéz

marad, így a megoldására különböző (meta)heurisztikák alkalmazása szükséges a CFS algoritmusban is.

Az *FSelector* nevű R csomag legújabb verziójában (Romanski – Kotthoff, 2018) elérhető a CFS algoritmus egy sztochasztikus hegymászó metaheurisztikával kombinálva, amivel nem kell az algoritmus alkalmazását lokális kereső eljárásokkal vagy egy előre megadott $\tilde{X}$ elemszámra, azaz $|\tilde{X}|$-re korlátozni.

Viszont, mivel a (26) célfüggvény csupán a változók közti lineáris korrelációt alkalmazza, két valószínűségi változó közös információtartalmának mérésére, így az algoritmus ilyen formában nem-lineáris modellek változószelekciós feladatának megoldására nem használható. Ehhez fejlesztéseket kell végrehajtani az algoritmuson.

Továbbá, az algoritmus nem kezeli lineáris modellek esetén a multikollinearitás minden esetét. Hiszen (26) nem veszi figyelembe a multikollinearitásnak azon esetét, amikor az nem magyarázóváltozók közötti páronkénti korrelációban jelentkezik, hanem abban, hogy bizonyos magyarázóváltozók lineáris kombinációjaként kifejezhető egy másik magyarázóváltozó. A VIF_j mutatók viszont ezt a lehetőséget is figyelembe veszik a 4.1. fejezetben adott definíció alapján.

6.6. Az mRMR módszer

Az mRMR (minimum redundancy – maximum relevance) változószelekciós keretrendszer Ding – Peng (2005)-ben jelent meg. Mint a módszer neve is sugallja, ez a változószelekciós eljárás egyszerre kísérel meg optimalizálni arra, hogy olyan változókat válogasson be a modellbe, amik szoros kapcsolatban állnak az eredményváltozóval, ám egymással nem állnak szignifikáns kapcsolatban. Ennek mérésére a szerzők egy saját mértéket adnak két változó közös információtartalmának mérésére, ami nagyon hasonló a CFS algoritmusában használt mértékhez, sőt folytonos változók esetén szintén a lineáris korrelációval méri a két változó együtt mozgását. Emiatt még nem tekinthető tisztán nem-lineáris változószelekciós algoritmusnak az mRMR, de a kategorikus változók esetén már tesz lépéseket efelé.

Kategorikus változók (pl. x és z) esetében algoritmus a nem-linearitást a változók együttes eloszlásának ($p(x,z)$) és peremeloszlásaiknak ($p(x)$ és $p(z)$) figyelembevételével kezeli. Két kategorikus változó közös információtartalma az algoritmusban az alábbi formula segítségével adott:

$$I(x,z) = \sum_{ij} p(x_i, z_j) \log \frac{p(x_i, z_j)}{p(x_i)p(z_j)}$$

Amennyiben x egy folytonos változó, míg z egy kategorikus változó K db kategóriával, akkor az mRMR-ben a közös információtartalmat egyszerűen az ANOVA F statisztikájával mérik a szerzők:

$$I(x,z) = \left[\left(\sum_k n_k(\bar{x}_k - \bar{x})^2\right)/(K-1)\right] \Big/ \left[\left(\sum_k (n_k - 1)\sigma_k^2\right)/(n-K)\right]$$

Ahol $\bar{x}_k$ az x folytonos változó átlaga z k-adik kategóriáján belül, $\bar{x}$ az x folytonos változó teljes átlaga, σ_k^2 pedig x varianciája z k-adik kategóriáján belül és n_k z k-adik kategóriájában lévő elemek száma és természetesen $n = \sum_k n_k$

Folytonos x és z változók között a közös információtartalom mértéke az mRMR-ben egyszerűen a Pearson-féle korreláció abszolút értéke, tehát nem-lineáris kapcsolatokat folytonos változók között a módszer nem kezel:

$$I(x,z) = |r(x,z)|$$

Ezen közös információtartalmi mértékek mellett akarunk egy regressziós modellben olyan X_j magyarázóváltozókat választani, amikre $I(Y, X_j)$ magas, ám $I(X_j, X_k)$ alacsony $\forall k$-ra, ahol X_k a regressziós modellben lévő egyéb magyarázóváltozókat jelöli.

Az mRMR algoritmus a fenti kettő kritériumot egy lineáris kereséssel oldja meg. Legyen $X = \{X_1, \dots, X_m\}$ a lehetséges magyarázóváltozók halmaza, és a regressziós modellünkbe keressük a fenti két kritérium szerinti legjobb $\tilde{X} \subseteq X$ részhalmazt. Tegyük fel, hogy már p db elemet kiválasztottunk X-ből $\tilde{X}$-be. Ezen a ponton az mRMR algoritmus azt az X_j magyarázóváltozót választja be $\tilde{X}$-be, amire $j = \underset{k \in X \setminus \tilde{X}}{\operatorname{argmax}} I(Y, X_k) \Big/ \frac{1}{|\tilde{X}|}\sum_{i \in \tilde{X}} I(X_i, X_k)$.

Láthatjuk, hogy a maximalizálandó célfüggvény nő, ha X_k közös információtartalma az eredményváltozóval is nő. Azonban, a célfüggvény csökken, ha a modellbe beválogatott más X_i magyarázóváltozókkal is nő X_k közös információtartalma.

Az eljárásból láthatjuk, hogy az mRMR algoritmus egy lineáris keresés, amihez szükséges előre megadni $\left|\tilde{X}\right|$-et, mivel egyébként más kilépési kritérium hiányában a fentebb leírt eljárás minden magyarázóváltozót bevesz a modellbe X-ből. Az algoritmus *R*-ben a *mRMRe* csomagban került implementálásra (De Jay et al., 2013), ahol az alapértelmezett beállítás szerint $\left|\tilde{X}\right| = 6$.

6.7. HSIC-Lasso módszer

Song et al. (2012) továbbfejlesztették az mRMR algoritmust azzal, hogy egy mérési skálára egységes és nem-linearitást is kezelő asszociációs mértéket használnak két változó közös információtartalmának mérésére, mégpedig a Gretton et al. (2005)-féle Hilbert-Schmidt

függetlenségi kritériumot ($HSIC$). Egy x és z valószínűségi változópár mellett a Hilbert-Schmidt kritérium a (27) alakot ölti:

$$HSIC(x,z) = E_{x,x',z,z'}[K(x,x')L(z,z')] + E_{x,x'}[K(x,x')]\mathbb{E}_{z,z'}[L(z,z')] - 2E_{x,z}\left[E_{x'}[K(x,x')]E_{z'}[L(z,z')]\right] \quad (27)$$

Ahol K és L $\mathbb{R}^{n\times n} \to \mathbb{R}$ magfüggvények RMHT-ben. $E_{x,x',z,z'}$ 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.

A gyakorlatban Song et al. (2012) javaslata alapján egy normalizált $HSIC_v$ kritériumot érdemes számolni, ha egy X_j magyarázóváltozóra és az Y célváltozóra adottak a $\boldsymbol{K_j}$ *és* $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ Gram mátrixok. A Gram mátrixok elemei egyszerűen a magfüggvények értékeit tartalmazzák minden megfigyelés párosra. Tehát, $\{\boldsymbol{K_j}\}_{ab} = K(x_{ja}, x_{jb})$ és $\{\boldsymbol{L}\}_{ab} = L(y_a, y_b)$ módon számíthatók. Bármely $\boldsymbol{K} \in \mathbb{R}^{n\times n}$ Gram mátrixra: $\bar{\boldsymbol{K}} = \frac{\boldsymbol{HKH}}{\|\boldsymbol{HKH}\|}$, ahol $\boldsymbol{H} \in \mathbb{R}^{n\times n}$ egy centráló mátrix, $\{\boldsymbol{H}\}_{ij} = \delta_{ij} - \frac{1}{n}$ elemekkel, melyben $\delta_{ij} = 1$, ha $i = j$ és 0 egyébként. Ennek ismeretében:

$$HSIC_v(X_j, Y) = \mathrm{tr}(\bar{\boldsymbol{K}}_{\boldsymbol{j}}\bar{\boldsymbol{L}})$$

Song et al. (2012) a $HSIC_v$ kritérium alapján javasolt az mRMR algoritmushoz hasonló lineáris keresést a változószelekcióra ismert $|\tilde{X}|$-el. Ezen az eljáráson fejlesztett Yamada et al. (2014) és Yamada et al. (2018) azzal, hogy a $HSIC_v$ kritériumokhoz magyarázóváltozóként egy α_j együttható hozzárendel a $HSIC_v(X_j, Y)$ kritériumokhoz, és ezekkel az együtthatókkal a nemnegatív Garrote módszerhez hasonlóan egy lineáris Lasso problémává alakítja a változószelekciós feladatot. Ezzel feloldva a Song et al. (2012)-féle ismert $|\tilde{X}|$ feltevést. Továbbá, a célfüggvényben korrigál a $HSIC_v(X_j, X_k)$ magyarázóváltozó párok függetlenségi kritériumával, ezzel a több redundáns magyarázóváltozó kiválasztásának elkerülését ösztönözi a (28) módon adott Lasso feladatban.

$$\max_{\Gamma\geq 0} \sum_{j=1}^{m} \gamma_j HSIC_v(X_j, Y) - \frac{1}{2}\sum_{j,k=1}^{m} \gamma_j\gamma_k\, HSIC_v(X_j, X_k) - \lambda\|\Gamma\|_1 \quad (28)$$

(28)-ban $\lambda > 0$ egy paraméter, ami 10-szeres keresztvalidáció segítségével megválasztható pl. AIC minimalizálásával. Továbbá, $\Gamma = [\gamma_0, \gamma_1, \dots, \gamma_m]$. (28) megoldásában a nemnegatív Garrote módszerhez hasonlóan a modellből elhagyható változók γ_j együtthatói 0-nak adódnak.

A vektorizálás operátorral ($\mathrm{vec}(\cdot)\colon \mathbb{R}^{n\times n} \to \mathbb{R}^{n^2}$) egy numerikusan könnyebben kezelhető alakban adható meg a feladat a Gram mátrixok segítségével:

$$\min_{\Gamma\geq 0} \|\mathrm{vec}(\bar{\boldsymbol{L}}) - [\mathrm{vec}(\bar{\boldsymbol{K}}_{\boldsymbol{1}}), \dots, \mathrm{vec}(\bar{\boldsymbol{K}}_{\boldsymbol{m}})]\Gamma\|_2^2 + \lambda\|\Gamma\|_1$$

A HSIC-Lasso megoldása viszont még ebben a vektorizált alakban is nagyon memóriaigényes, konkrétan $O(mn^2)$ az m db Gram mátrix tárolása miatt.

Emiatt Climente-González, et al. (2019) a HSIC Lasso feladat megoldásához a tanulóhalmazt $B \ll n$ elemű blokkokra bontja fel, amelyekből gyorsabban ki lehet számítani $HSIC_v$ kritériumokat, és ezek összegzésével kaphatunk egy, az egész tanulómintára értelmezett $HSIC_b$ kritériumot:

$$HSIC_b(X_j, Y) = \frac{B}{n}\sum_{l=1}^{B/n} HSIC_v\left(X_j^{(l)}, Y^{(l)}\right)$$

Ahol $X_j^{(l)}$ és $Y^{(l)}$ a j-edik magyarázóváltozó és a célváltozó részhalmazai az n elemű minta l-edik blokkján. Egy $HSIC_v\left(X_j^{(l)}, Y^{(l)}\right)$ kiszámítása $O(B^2)$ memóriaigényű, így a teljes $HSIC_b$ számítás memóriaigénye $O(nB^2) \ll O(mn^2)$. Mivel a feladat eredménye függhet a blokkok kialakításától, így a blokkokat érdemes lehet Q db különböző permutációban létrehozni, és azokat kiátlagolni:

$$HSIC_{bb}(X_j, Y) = \frac{1}{Q}\sum_{q=1}^{Q}\frac{B}{n}\sum_{l=1}^{B/n} HSIC_v\left(X_j^{(l,q)}, Y^{(l,q)}\right)$$

Ahol $X_j^{(l,q)}$ és $Y^{(l,q)}$ a j-edik magyarázóváltozó és a célváltozó részhalmazai az n elemű minta l-edik blokkján a q-adik permutációban. A (28) Lasso feladat $HSIC_{bb}$-k használatával $HSIC_v$-k helyett hatékonyabban megoldható.

Vegyük észre, hogy a $HSIC$ kritérium alkalmazása a változók közti kapcsolat szorosságának mértékét csak páronként vizsgálja, így továbbra sem számolunk azzal az esettel, amikor egy magyarázóváltozó más egyéb magyarázóváltozók többváltozós függvényeként előáll. Ezzel a HSIC-Lasso módszer által adott modellekben is számítani lehet káros mértékű, az értelmezéseket torzító concurvity jelenségre.

A HSIC-Lasso algoritmusnak jelenleg csak Python implementációja létezik (Climente-González, et al., 2019). Ez utóbbi implementációt a Repl.it online fejlesztőkörnyezetben futtattuk. Az implementáció Gauss magfüggvényt $K(x, y) = e^{-\frac{\|x-y\|^2}{2\sigma^2}}$ alkalmaz folytonos változókra. Ahol $\sigma = 1$, mivel az implementációban a változók normalizálásra kerülnek. Ha a változó bináris, akkor Delta magot alkalmaz az algoritmus:

$$K(x, y) = 1/n_y\,, ha\ x = y; 0\ egyébként$$

Ahol n_y az y-al kódolt értékek elemszáma a mintában.

7. A hibrid genetikus-harmónia kereső (HGHK) algoritmus

Korábbi munkáinkban (Láng et al., 2017) (Kovács, 2019) egy hibrid genetikus – harmónia kereső algoritmust (HGHK algoritmus) javaslunk a változószelekciós feladat megoldására lineáris modellek esetében. Az algoritmus a magyarázóváltozók VIF értékén keresztül a szelekciós folyamat során nem csak a változók közti páronkénti káros korrelációkra szűr.

A változószelekciós algoritmusok friss irodalmában elég gyakori a metaheurisztikus megoldások alkalmazása a feladat megoldására. Néhány példa a közelmúltból: Wang et al. (2015), Krömer – Platoš (2016), Mafarja – Mirjalili (2017), Sayed et al. (2019) és Abdel-Basset et al. (2020).

Az alkalmazott metaheurisztikák tárháza széles. Wang et al. (2015) harmónia kereső algoritmust alkalmaz, Krömer – Platoš (2016) genetikus algoritmust. Sayed et al. (2019) egy újabb metaheurisztikát, a „Dragonfly" algoritmust (Mirjalili, 2016) alkalmazza a feladatra. Abdel-Basset et al. (2020) szintén egy friss megoldást, a „szürke farkas" algoritmust (Mirjalili et al., 2014) implementálja a változószelekciós feladatra. Mafarja – Mirjalili (2017) megoldása az általunk javaslotthoz hasonlóan egy hibrid megoldás: szimulált hűtést ágyaznak be egy bálna optimalizáló algoritmusba.

Az idézett tanulmányok közös pontja, hogy a változószelekciós feladatban a modellt nem GLM-nek vagy GAM-nak tekintik, hanem valamilyen egyéb felügyelt gépi tanuló algoritmusnak (pl. k-NN, támaszvektor-gép, neurális hálózat stb.). Részben ebből adódóan az eredményváltozó szempontjából a változószelekciót csak osztályozó modellek körében értelmezik, mint a legjellemzőbb felügyelt tanulási feladatot.

Az alkalmazott modellek jellege miatt az eredményül kapott modellekben a magyarázóváltozók eredményváltozóra gyakorolt hatásának visszafejthetősége nem szempont az idézett munkákban. Ebből adódóan az algoritmusok csak a (2)-vel adott változószelekciós célfüggvény valamilyen speciális, osztályozási feladatra szabott változatát alkalmazzák. Az idézett tanulmányok leggyakoribb választása egy $\alpha\gamma_R(D) + (1-\alpha)\frac{p}{m}$ alakú minimalizálandó célfüggvény alkalmazása. Ahol $\gamma_R(D)$ a vizsgált osztályozó algoritmus félreosztályozási aránya és korábbi jelöléseink alapján $\frac{p}{m}$ a magyarázóváltozók kiválasztási aránya. $\alpha \in [0,1]$ egy választható paraméter, ami a modell pontossága és az a kiválasztott magyarázóváltozók száma közti egyensúlyt szabályozza.

Az idézett tanulmányokban egyéb korlátok (pl. multikollinearitás vagy concurvity jelenség elkerülésére a végső modellben) megfogalmazására nem kerül sor a változószelekció során. Ez részben ismét annak tudható be, hogy az idézett munkákban a magyarázóváltozók

hatásainak visszafejtése nem szempont. Továbbá, a tanulmányokban alkalmazott tanuló algoritmusok többségében a magyarázóváltozók függetlensége nem feltétel. Tudomásunk szerint az általunk javasolt HGHK algoritmus az egyetlen metaheurisztikus megoldás a változószelekciós feladatra, ami korlátozó feltételként a végső modellben szereplő magyarázóváltozók többdimenziós korrelálatlanságát is biztosítja. Ahogyan a 2. fejezetben ismertettük, ez a feltétel elsősorban magyarázó modell építése esetén releváns GLM vagy GAM keretben, ám léteznek további osztályozó algoritmusok, amelyek megkövetelik a fenti feltétel teljesülését (pl. naiv Bayes modell).

Jelen fejezetben először bemutatjuk a HGHK algoritmus alapjául szolgáló két metaheurisztikát, a genetikus és harmónia kereső algoritmusokat, és ismertetjük az algoritmus működését lineáris modellek esetében. Ezek után bemutatjuk az algoritmus kiterjesztését GAM keretre, thin plate spline-ok segítségével reprezentálva a magyarázóváltozók f_j transzformáló függvényeit.

7.1. A genetikus algoritmus

A genetikus algoritmust 1975-ben Holland fejlesztette ki, majd 1989-ben Goldberg továbbfejlesztette. (Goldberg, 1989) Az algoritmus alapgondolata a biológiai genetikán és Darwin „természetes kiválasztódás” elvén alapul. Az egyedek az egyes megoldások, melyeket az algoritmus többelemű populációba szervez, ezzel biztosítja, hogy a keresési tér minél nagyobb részét járjuk be. Ráadásul képes egy, az egyed alatti szintet is definiálni, a gének szintjét, amikből az egyedek felépülnek, ezzel kiváló lehetőséget biztosítva a megoldások (egyedek) finom, aprólékos változtatására, azaz a keresési tér még pontosabb bejárására. A géneket általában bitsorozattal ábrázoljuk, mint a legáltalánosabb megoldást, de lehetséges valós értékeket is alapul venni.

A három szint (populáció, egyed, gének) megválasztása után az algoritmus minden lépésében egy új populációt hoz létre a célfüggvény, azaz fitnesz értékek alapján (a fitnesz elnevezés alkalmazása a „célfüggvény érték” kifejezés helyett ebből az algoritmusból ered, hiszen Darwin szerint: „Survival of the fittest”), így folyamatosan halad az optimális megoldás (egyed) felé a populációnk. Ezt a folyamatot három ún. genetikus operátor képviseli: szelekció, rekombináció, mutáció.

A szelekciós eljárás segítségével választjuk ki az egyes populáció legjobb egyedeit, melyeket továbbviszünk a következő populációba, és amelyek a rekombinációs eljárás bemenetei lehetnek. Alapvetően három szelekciós operátor közül választhatunk:

- **Rulett kerék**: A kiválasztás valószínűsége a fitnesz értékkel arányos, olyan módon, hogy egy megoldás kiválasztásának valószínűsége annál nagyobb, minél nagyobb

a rátermettsége a populáció rátermettségi átlagához képest. A kiválasztás úgy működik, mintha az egyedeket egy olyan rulett kerék egy-egy mélyedéséhez (cikkelyéhez) rendelnénk, ahol a mélyedés nagysága arányos az egyed rátermettségével.

- **Verseny szelekció**: Egy csoportot választunk ki a populációból és a fitnesz értékük alapján versenyeztetjük őket. Az optimumhoz közelebb eső fitnesz értékű egyed győz.
- **Elitista**: Az eddig kapott legjobb $\varphi * c$ db egyedet mindig áthelyezi az új populációba, ahol c a populáció mérete, φ pedig az elitek aránya, és $\varphi \in [0, 1]$.

A rekombinációs operátor két egyedből hoz létre új egyedet. A biológiában ez egy biológiai utód létrehozásának felel meg. Egyik leggyakoribb kombináció az egypontos keresztezés. Ennek során a műveletben résztvevő egyedeket egy véletlenszerűen kiválasztott vagy egy előre rögzített génnél elvágjuk. A vágás előtti géntulajdonságokat örökli az új egyed az egyik szülőtől, a vágás utániakat a másik szülőtől.

A mutációs operátor egyetlen megoldást igényel, és ebből állít elő egy újat, az eredeti egy variánsát, mutációját. A mutáció célja a populáció frissítése. Ezzel a művelettel biztosítható, hogy egy eljárás képes legyen kitörni a lokális megoldás környezetéből. A mutációt általában úgy oldják meg, hogy rendszerint az egyed (megoldás) egy véletlen kiválasztott génje változik meg. De olyan megoldás is elképzelhető, ahol egy kellően kis valószínűséggel akár az összes gén módosulhat.

A legelterjedtebb megoldásokban elitista szelekciót használunk, és a mutáció során az összes gén módosulhat kellően kis valószínűség mellett. (Horváth, 1998)

Az genetikus algoritmus egyik legnépszerűbb alkalmazása a magyarázóváltozók szelekciójára lineáris modellekben az R *glmulti* csomagjában található (Calcagno, 2019). A *glmulti*-ban implementált algoritmus bináris kódolást használ az egyedek reprezentációjára. Ha az adatbázis j-edik változó szerepel a modellben, akkor az m (lehetséges magyarázó változók száma) elemű bitsorozat j-edik tagja 1 értéket vesz fel, ha nem, akkor 0-át.

A *glmulti* csomag dokumentációja alapján rekonstruált algoritmus pontos folyamatábráját a mellékletben szemléltetjük.

Az algoritmus futása akkor állítjuk le, ha a célfüggvény (AIC, SBC, stb.) értéke nem változik jobban, mint $0{,}05$ a populáció legjobb egyedének esetében.

7.2. A harmónia kereső algoritmus szelekciós operátorainak alkalmazása a genetikus algoritmusban – A HGHK algoritmus működése

Láng et al. (2017) bemutatja a genetikus algoritmus egy továbbfejlesztését lineáris regresszió esetében.

A fejlesztés lényege az algoritmus gyorsítása volt a rekombinációs operátorok cseréjének segítségével. Ugyanis, a tapasztalat azt mutatja, hogy a genetikus algoritmus keresztezési operátora alapvetően olyan problémák esetén alkalmazható nagy sikerrel, 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 nincsenek csak az adott egyed génsorozatának elején vagy végén kialakuló megfelelő génmintázatok. Ráadásul, egyetlen változó bevétele vagy elhagyása modellből drasztikusan megváltoztathatja a célfüggvényünk ($AIC, SBC,$ stb.) értékét. Tehát, szükségünk van az egyedek nagyobb fokú véletlenségét biztosító rekombinációs operátorokra. Ezeket pedig egy másik evolúciós algoritmusból, a fejlesztett harmónia kereső algoritmusból kölcsönözzük.

Mahdavi et al. (2007) szerint a harmónia keresés a zenei improvizáció világából meríti a rekombinációs operátorait.

A zenészek improvizációs folyamata alapvetően három szabály alapján zajlik:

- A zenészek emlékezetből lejátszanak egy ismert dallamot.
- Más hangfekvésben vagy más ritmusban játszanak le egy már lejátszott dallamot.
- Teljesen ötletszerűen, érzéseikre hagyatkozva játszanak le egy új dallamot.

Hasonló elven működik az improvizáció a harmóniakereső algoritmusban, bár a három alapszabály egy kicsit más:

- A repertoárból (korábbi populáció) lejátszunk egy már ismert dallamot (egyedet).
- Egy ismert dallamon (korábbi populációból egy egyedet) módosítunk.
- Teljesen véletlen dallamot (egyedet) generálunk.

Ezen három alapszabály alkalmazását kell lefordítanunk a matematika nyelvére. Erre a feladatra két valószínűséget fogunk bevezetni.

Az egyik a $HMCR$ (Harmony Memory Consideration Rate), ami a memóriából való választás valószínűségét adja meg. A memória kifejezés a harmónia kereső algoritmus terminológiájában a populációnak felel meg. Annak a valószínűsége, hogy teljesen véletlen megoldást generálunk $1 - HMCR$. Ha a $HMCR$ túl alacsony értékű, akkor csak kevés dallam kerül kiválasztásra a repertoárból és az algoritmus lassan konvergál. Ha ez az érték túl magas, közel 1, akkor majdnem az összes hang felhasználásra kerül a repertoárból, s ekkor csökken más dallamok felfedezésének lehetősége.

A következő valószínűség a dallam módosításának (hangmagasságának megváltoztatása) esélyét határozza meg. Tehát, ha már a memóriából választottunk egy dallamot, akkor ezzel a PAR (Pitch Adjusting Rate) valószínűséggel módosítunk rajta, és $1 - PAR$ valószínűséggel hagyjuk változatlanul.

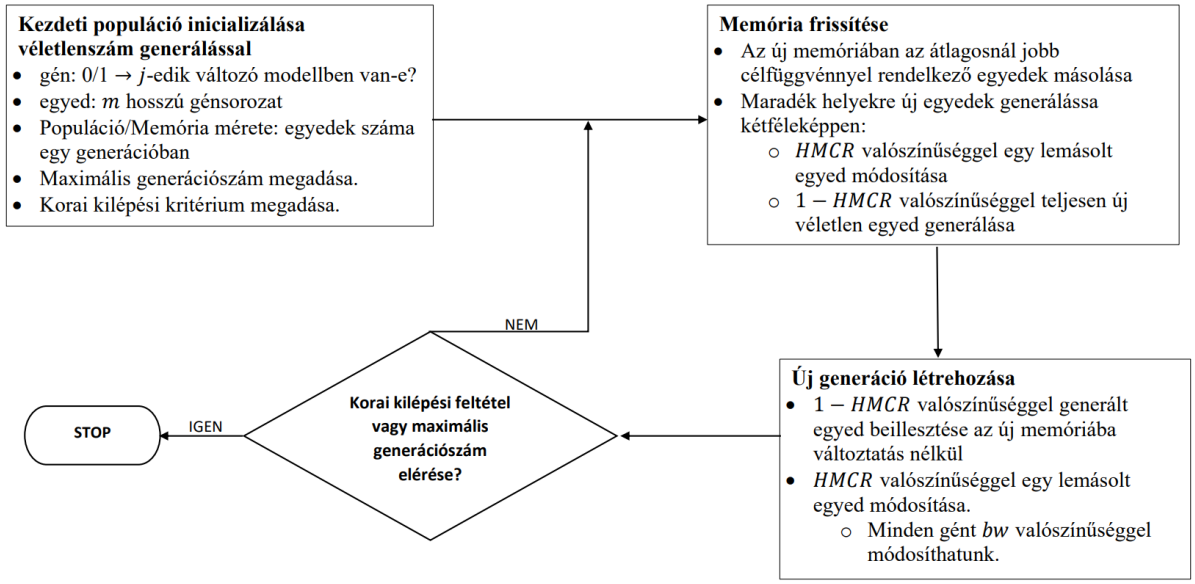
A hangmagasság megváltoztatása a zenei gyakorlatban lineárisan történik. Ezt a lépést (29) írja le.

$$x_{új} = x_{régi} + bw \cdot r \tag{29}$$

A (29) képletben $x_{új}$ a keletkezett új dallam, $x_{régi}$ pedig a memóriából kiválasztott már lejátszott dallam. A sávszélességet (bandwidth), azaz a keresési tér nagyságát a bw paraméter jelöli. Végezetül, r egy egyenletes eloszlású véletlen szám a $(-1, 1)$ intervallumban.

A változószelekciós feladatban a génjeink bináris változók, tehát (29)-t nem alkalmazhatjuk rájuk. Ebben a feladatban a (29) műveletet genetikus algoritmus mutációs operátorként értelmezzük. Ezen kívül a $HMCR$ valószínűség használatát emeljük át a genetikus algoritmusunkba.

Ezt a két az új rekombinációs operátort bevezetve megadjuk ennek az új hibrid harmónia kereső – genetikus algoritmusnak a folyamatábráját a 7. ábrán. A $HMCR$ valószínűség a futás során nő, míg a módosítás mértéke (bw) a futás során csökken. Ezzel azt a hatást váltjuk ki, hogy az algoritmus futásának elején minél agresszívebb legyen 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 pedig az algoritmus futása során egyre inkább közelebb kerülünk az optimumhoz, a keresési tér egyre kisebb részét kell bejárnunk az új egyedek generálása során. A HGHK algoritmust akkor állítjuk le, ha az elért egy előre megadott maximálisiterációszámot vagy teljesít egy korai kilépési feltételt. A korai kilépési feltételt akkor elégítjük ki, ha az utolsó valahány lépésben (ez paraméterként megadható) a célfüggvény ($AIC, SBC,$ $\bar{R}^2$stb.) értéke nem változik a memóriánk (populációnk) legjobb egyedének esetében. Az algoritmus a korábbi memória (populáció) legrátermettebb egyedeit mindig átviszi a következő generációba a genetikus algoritmus szelekciós operátorainak megőrzése miatt. Így a legjobb megoldás (aminek változatlanságát a korai kilépési feltétel vizsgálja) nem fog romlani.



7. ábra: A HGHK algoritmus folyamatábrája. Forrás: saját szerkesztés.

Az algoritmusnak fontos tulajdonsága, hogy a populáció/memória frissítése során megőrizzük a genetikus algoritmusból az egyedek párhuzamos kezelését. Tehát, az egyedek generálása az új populációba történhet párhuzamosan is.

Az új rekombinációs operátorok és a párhuzamos egyedgenerálás miatt a futásidőben elért javulás olyan jelentős, hogy kettő extra korlátot is hatékonyan be tudunk építeni a változószelekciós algoritmusba. Az első korlát értelmében csak olyan modelleket fogadunk el optimálisnak, amelyekben minden magyarázóváltozó szignifikáns egy előre adott α szignifikancia-szinten. A második korlátban pedig biztosítani tudjuk, hogy a VIF mutató alkalmazásával az algoritmus az eredményül adott modell esetében zárja ki a magyarázóváltozók közti multikollinearitást. Azaz egy egyedhez tartozó modellben minden X_j-re $VIF_j < K$-nak fenn kell állnia, ahol K egy felhasználó által megadott korlátozó érték a VIF mutatóra. A 4.1. fejezet alapján K értékét 2-nek vagy 5-nek érdemes választani a multikollinearitással kapcsolatos tolerancia függvényében.

A korlátok technikai megvalósítása a populáció egy egyedét leíró két új bináris változó bevezetésével történik. Ha az i-edik egyed teljesíti a multikollinearitásra vonatkozó korlátot, akkor a C_i bináris változó 1 értéket vesz fel, egyébként 0-t. Ha a felhasználó által megadott szignifikancia-szinten minden változóra elutasítható a $\beta_j = 0$ nullhipotézis az i-edik egyedben, akkor S_i bináris változó 1 értéket vesz fel, egyébként 0-t.

Amikor a memória frissítése során kiválogatjuk az átlagosnál jobb egyedeket az aktuális memóriából, akkor a memória átlagos célfüggvény értékére a (30) formulával adott súlyozott

átlagot alkalmazzuk és csak olyan i egyed minősülhet átlagosnál jobb egyednek, amire $C_i = S_i = 1$.

$$\bar{F} = \frac{\sum_{i=1}^{N} F_i \cdot C_i \cdot S_i}{\sum_{i=1}^{N} C_i \cdot S_i} \tag{30}$$

Ahol N a memória méretét, F_i az i-edik egyed célfüggvény értékét jelöli. Amennyiben $\sum_{i=1}^{N} C_i \cdot S_i = 0$, akkor $\bar{F}$ egyszerűen az F_i értékek számtani közepe és minden aktuális memóriában lévő egyed minősülhet átlagosnál jobb egyednek, nem csak azok, amikre $C_i = S_i = 1$.

Fontos észrevenni, hogy az algoritmusba épített korlátok miatt könnyen előfordulhat, hogy a véletlenszerűen generált kezdeti memóriában nem lesz olyan egyed, amit tovább engedünk 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 memória mérettel futtatni, vagy úgy, hogy egy futásidőben könnyen kezelhető memória méretet választunk, és többször lefuttatjuk az algoritmust egy alacsonyabb maximális generációszám mellett. A több futási eredmény legjobb célfüggvény értékkel rendelkező megoldást tekintjük optimumnak. Mint a 10. fejezetben látni fogjuk, az utóbbi stratégia kimondottan jól működik GAM keretben, nagyobb méretű adatbázisok esetében.

Láng et al. (2017) lineáris regressziós modellek esetében megmutatta, hogy szimulált, és valós, multikollinearitást tartalmazó adatbázisokon a HGHK algoritmus az egyetlen olyan megvizsgált modellszelekciós algoritmus (a tanulmányban a HGHK mellett a Leaps and Bound, GARS, Stepwise, PLS, Elastic Net, VIF alapú Backward és Egymásba ágyazott becslések módszerei kerültek összehasonlításra), amely a végső modellekből teljes mértékben eliminálja a multikollinearitást, és a modell magyarázóereje sem csökken le jelentősen az olyan modellekhez képest, melyek magyarázóváltozói között zavaró vagy káros multikollinearitás tapasztalható.

Kovács (2019) tanulmányban alkalmazzuk a HGHK algoritmust egy olyan esetre, amikor a lineáris modellünk eredményváltozója élettartamot jelöl. Ez a Cox-regresszió esete. Élettartam adatok többváltozós statisztikai elemzése során a következő n elemű mintával szembesülünk: $\left(t_1, \boldsymbol{x}^{(1)}, d_1\right) \dots \left(t_n, \boldsymbol{x}^{(n)}, d_n\right)$. Ahol $\boldsymbol{t} = (t_1, \dots, t_n)^T$ a vizsgált eltelt időtartam egy adott esemény/meghibásodás bekövetkezéséig, $\boldsymbol{x}^{(i)} = \left(x_{i1}, \dots, x_{ip}\right)$ a magyarázóváltozók p elemű vektora az i-edik mintaelem esetén. $\boldsymbol{d} = (d_1, \dots, d_n)^T$ pedig az úgynevezett

állapotváltozókat tartalmazó vektor. Az i-edik mintaelem d_i állapotváltozója 1 értéket vesz fel, ha a vizsgált esemény (nekünk törlés) bekövetkezett, míg 0 értéket vesz fel, ha az esemény még nem következett be, tehát a megfigyelt t_i időtartamunk cenzorált. Kovács (2019)-ben a t_i időtartamok életbiztosítási szerződések megkötésétől a törléséig eltelt időt jelentik hónapokban mérve

Élettartamadatok elemzése során fő célunk a $G(t) = 1 - F(t) = P(T \geq t)$ túlélésfüggvény becslése, ahol T a modellezett időtartam (túlélési idő), mint valószínűségi változó. A túlélésfüggvény tehát, minden egyes $t \geq 0$ időtartamra megadja annak a valószínűségét, hogy t idő eltelte előtt egy egyed még nem lépett ki a vizsgálatból (esetünkben: szerződés még életben van). Cox-regresszió során pedig azt vizsgáljuk, hogy ezen $G(t)$ túlélésfüggvényre hogyan hatnak különböző $\boldsymbol{x} = \left(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n)}\right)^T$ magyarázóváltozók ismert realizációi. A leírást Cox (1972), Cox (1975) alapján készítettük el. A multikollinearitás jelensége Cox-regressziós modellekben is problémákat vet fel, mivel az együtthatóvektor ekkor is ceteris paribus elven értelmezendő. (Xue et al, 2007)

A HGHK algoritmus eredményei itt is hasonlóak voltak, mint a klasszikus lineáris regresszió esetében. A hagyományos modellszelekciós algoritmusok (Stepwise, Lasso stb.) mellett a HGHK algoritmus képes volt arra, hogy a magyarázóerő minimális csökkenése mellett olyan változóhalmazt adjon megoldásként, ami multikollinearitástól mentes. Ennek köszönhetően egy életbiztosítási szerződésállományban előforduló törlés miatti lemorzsolódási okok könnyebben feltárhatóak voltak a HGHK algoritmus segítségével, mint más változószelekciós algoritmus által szolgáltatott modelleket felhasználva. Például, az egyik fő megmaradást segítő tényező az életbiztosítások mellé köthető műtéti térítésre és kórházi napidíjra szóló kiegészítő fedezetek megléte, ám a hagyományos modellszelekciós algoritmusok a baleseti rokkantságra szóló fedezet meglétét tulajdonították egy fontos megmaradást segítő tényezőnek, mivel korrelált az előbbi két kiegészítő fedezet meglétével.

7.3. A HGHK algoritmus GAM keretben

A 7.1. és 7.2. alfejezetekből, valamint a 7. ábrán látható, hogy a HGHK 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, az 5.2.

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 5.2.2. alfejezetben felvetett ötlet alapján HGHK algoritmusban azt a technikát követjük, hogy ha egy X_j magyarázóváltozót beveszünk a GAM modellbe, akkor az alapértelmezés szerint $k_j = 10$. Amennyiben X_j értékkészlete kisebb, akkor k_j az X_j magyarázóváltozó lehetséges értékek számával lesz egyenlő. Amennyiben az Augustin et al. (2012)-féle χ^2-próbák p-értéke egy előre megadott α szignifikancia-szint alatti, akkor a k_j értékét 5-ösével növeljük addig, amíg a változóhoz megfelelően nagy k_j-t nem választottunk.

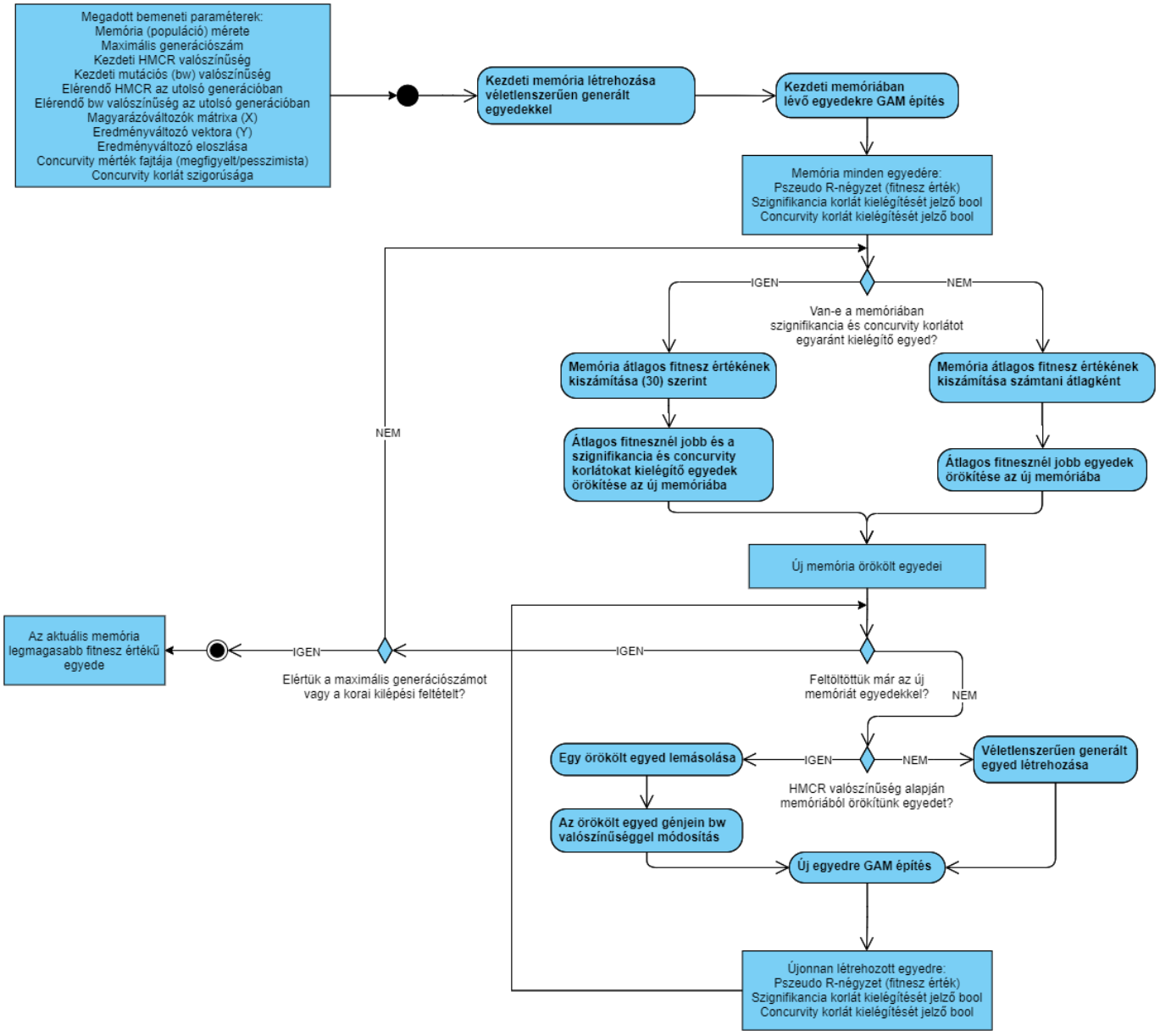
A túl magas k_j választás csupán extra számítási kapacitásba kerül, túlilleszteni nem fogjuk a spline függvényt. Ugyanis az eredeti (7) illesztési feladat $\lambda \int f''(x)^2 dx$ büntetőtagja lineáris függvényformát fog preferálni egy olyan szakaszon X_j értelmezési tartományán, ahol a megfigyelt pontok csak valamilyen zajjal térnek el a lineáris függvényformától. Ezzel a változószelekciós feladat bonyolultsága megmarad $2^m - 1$-nek, hiszen a k_j-k választására nem kell új döntési változókat bevezetni.

Fontos változtatás még az algoritmusban, hogy a populáció i-edik egyedéhez tartozó C_i bináris változók GAM keretben már a concurvity jelenségre vonatkozó korlátozásnak való megfelelést jelentik. Tehát, azt szeretnénk elérni, hogy a memória frissítése során csak olyan megoldásokat vigyünk át az új memóriába, melyekre a 5.3. 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.

Ezen túl az S_i bináris változók mögötti technikai lépés is változik. A változó értéke akkor vehet fel 1-et, ha a populáció i-edik egyedéhez tartozó modellben a felhasználó által megadott szignifikancia-szinten minden változóra elutasítható a $\Psi_{k_j} = \mathbf{0}$ nullhipotézis. A 6.2. fejezet alapján ekkor mondhatunk thin plate spline-okat alkalmazó GAM keretben egy X_j-t, szignifikánsnak. A teszteléshez a Marra – Wood (2011) által megadott χ^2-próbát alkalmazzuk.

A K1 kutatási kérdésre tehát pozitív válasz adható a thin plate spline-ok jelen fejezetben bemutatott alkalmazási módjának segítségével, a GAM modellekben a spline függvények rendjének és a szakaszhatárok helyének megválasztása automatizálható. Ezzel pedig a lineáris esetben alkalmazott döntési változókon és bináris egyedreprezentáción nem szükséges változtatni.

A jelen fejezetben bemutatott GAM keretben dolgozó HGHK algoritmus működése UML 2 szabványnak megfelelő tevékenységdiagramon (Jibitesh – Ashok, 2011) a 8. ábrán tekinthető meg.



8. ábra: A HGHK algoritmus UML tevékenységdiagramja. Forrás: saját szerkesztés

A 9. és 10. fejezetekben két valós adatbázison vizsgáljuk az itt ismertetett HGHK algoritmus hatékonyságát különböző paraméterezések mellett. A következő, 8. fejezetben ezen numerikus hatékonyságvizsgálatok keretének tervezését tekintjük át.

8. Numerikus hatékonyságvizsgálatok tervezése

A 7. fejezetben bemutatott saját fejlesztésű HGHK algoritmus hatékonyságvizsgálatát különböző paraméterezések függvényében, és a teljesítményének összevetését a 6. fejezetben ismertetett GAM keretben működő változószelekciós algoritmusokéval két valós adatbázison mutatjuk be.

Az első vizsgált 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 magyarázóváltozók összes részhalmazának legenerálásával. Az adatbázis használatának a célja, hogy megvizsgáljuk, hogy az ismert globális optimumot milyen hatékonysággal azonosítják a vizsgált algoritmusok. Az adatbázisra a továbbiakban „betongerendák adatbázis" néven hivatkozunk.

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 magyarázóváltozóval 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 referencia pont. Az adatbázisra a továbbiakban „banki ügyfelek adatbázis" néven hivatkozunk.

Mindkét adatbázis esetében adattisztítási lépéseket végzünk, amennyiben szükséges. Ridzuan – Zainon (2019) ajánlásainak megfelelően megvizsgáljuk és kezeljük mindkét adatbázis esetében az alábbi adatminőségi problémákat:

- Érvényességi hibák
    - Adattípus hibák: pl. string típusban tárolt szám
    - Értékhatár hibák: pl. nagyobb érték, mint az elméleti maximum (1-nél nagyobb arányszám)
    - Duplikáció: ismétlődő egyedi azonosítók
    - Karakterhibák: pl. „o"-ként tárolt 0 érték egy számban (6o33)
    - Hibás besorolások: pl. férfi keresztnévhez „nő" nem mező tartozik
- Változók közötti kapcsolat hibái: pl. komplementer események arányait leíró változók összege nem 1 bizonyos rekordok esetében

- Modellek paraméterbecslési eljárásait nagy mértékben befolyásoló kiugró értékek jelenléte
- Hiányzó értékek egy vagy több változóban

A fenti adatminőségi hibáktól megtisztított adatbázisokat 70% - 30% arányban osztjuk fel tanító- és tesztmintákra. A tanító mintán futtatjuk a vizsgált változószelekciós algoritmusokat, és az esetleges keresztvalidációs lépések is mindig a tanító adatbázison történnek.

A tesztmintán az algoritmusok által javasolt modellek Y célváltozóra adott $\hat{Y}$ becsléseinek pontosságát vizsgáljuk a szakirodalomban javasolt teljesítménymetrikák segítségével.

Az értekezésben vizsgált teljesítménymetrikák köre két nagy csoportra bontható: a folytonos célváltozó és a Bernoulli-eloszlású célváltozó esetén alkalmazott metrikák körére. Erre azért van szükség, mert a betongerendák adatbázisban a célváltozó (szakítószilárdsága) folytonos és közel normális eloszlású (Yeh, 1998). Ezzel ellentétben, a banki ügyfelek adatbázisban a célváltozó (csődesemény bekövetkezett-e a lekérdezés után 1 hónappal vagy sem) Bernoulli eloszlású (Yeh – Lien, 2009). Ráadásul a csődesemény célváltozó kiegyensúlyozatlan is, mivel a vizsgált hitelkártya ügyfeleknek mindössze 24%-a jelentett csak csődöt hitelkártya tartozására. Emiatt kimondottan a kiegyensúlyozatlan Bernoulli eloszlású célváltozó esetén javasolt teljesítménymetrikákat szükséges a banki ügyfelek adatbázison alkalmazni.

Bizonyos algoritmusok erősen heurisztikus jellege miatt minden algoritmust 30-szor futtattunk az adatbázisokon, hogy az eredményül kapott végső modellek stabilitását is vizsgálni tudjuk. Azon algoritmusok esetén, ahol a végső modell nem volt mind a 30 futtatásban azonos, a legjobb célfüggvény[2] értékkel rendelkező megoldást jelenítjük meg az eredmények között.

Az algoritmusok által javasolt modellek becslési pontosságának vizsgálata mellett vizsgáljuk, hogy az egyes algoritmusok hány változót választanak ki a végső modellbe a teljes lehetséges X magyarázóváltozóhalmaz méretéhez képest. Továbbá, elemzésre kerül, hogy az algoritmusok által javasolt modellben szereplő változók közül hányat és melyeket érint a 5.3. fejezetben bemutatott káros concurvity jelenség.

Ezen felül a vizsgált algoritmusok várható futásidejét és a futásidők szórását is megvizsgáljuk a 30 futtatás tekintetében, hogy képet kapjunk az algoritmusok várható számításigényéről egy átlagosnál valamivel jobbnak tekinthető személyi számítógépen (az 1. fejezetben megadott konfiguráción), ami egy fontos szempont lehet a szoros határidőjű gyakorlati feladatok megoldása során.

[2] A célfüggvény mindig az algoritmus leírásában megadott célfüggvényt jelenti, és a tanító mintán megadott legjobb érték alapján választjuk ki a modelleket.

A HGHK algoritmus skálázhatóságának vizsgálata a nagyobb méretű, banki ügyfelek adatbázis esetén Microsoft Azure Data Science Virtual Machine (DSVM) virtuális számítógép különböző (az 1. fejezetben megadottnál erősebb) hardverkonfigurációin történt.

A numerikus hatékonyságvizsgálatok konkrét, adatbázisonként eltérő paramétereit a 9. fejezet tartalmazza a betongerendák, míg a 10. fejezet a banki ügyfelek adatbázis esetében.

A 8.1. és 8.2. alfejezetekben bemutatjuk a folytonos és kiegyensúlyozatlan Bernoulli-eloszlású célváltozó esetén a teszthalmazon vett kiértékelés során alkalmazott teljesítménymetrikákat.

8.1. Teljesítménymetrikák folytonos célváltozó esetén

A szakirodalomban több szerző (James et al, 2013) (Ramanathan, 1992) (Raschka – Mirjalili, 2017) (Wooldridge, 2016) is egyetért abban, hogy folytonos célváltozóval bíró modellek becslési pontosságának kiértékelésében a legnépszerűbb és legszemléletesebben értelmezhető teljesítménymetrika a klasszikus determinációs együttható, az R^2.

8.1.1. A klasszikus R-négyzet mutató és a négyzetes hibafüggvény

Az R^2 mutató a vizsgált felügyelt tanuló modell összes hibáját négyzetes skálán méri egy n elemű teszthalmazon.

Legyen $\hat{Y}$ változó egy folytonos Y célváltozóra adott becslés, tetszőleges felügyelti gépi tanuló modell alapján. Ekkor a modell összesített négyzetes (azaz L_2 normával vett) hibája (Sum of Squared Errors, SSE) (31) alakban írható fel.

$$SSE = \left\| Y - \hat{Y} \right\|^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{31}$$

Az R^2 ezt az SSE-t arányítja egy olyan „nullmodell" összes négyzetes becslési hibájához, ami Y-t, magyarázóváltozók nélkül, annak tapasztalati átlagával, $\bar{Y}$-al becsüli. Ennek a „nullmodellnek" az SSE-je az eredményváltozó teljes négyzetösszege (Sum of Squared Totals, SST), ami az eredményváltozó teljes varianciájának (összesen megmagyarázható információmennyiségének) is tekinthető: $SST = \|Y - \bar{Y}\|^2$. Az R^2 klasszikus definíciója (32) azt vizsgálja meg, hogy Y teljes varianciájának mekkora hányadát nem teszi ki a modellhiba (SSE). Ezzel gyakorlatilag a modell $[0,1]$ skálán mozgó, „százalékos" magyarázóerejeként értelmezhető.

$$R^2 = 1 - \frac{SSE}{SST} \tag{32}$$

Alternatív számítási módként az Y és $\hat{Y}$ közti Pearson-féle korreláció négyzeteként is számítható: $R^2 = \mathrm{cor}\left(Y, \hat{Y}\right)^2$.

8.1.2. Az RMSE mutató

Az R^2 mutató mellett több egyéb folytonos célváltozó esetén alkalmazott metrika is épül az SSE-re. Az egyik leggyakrabban alkalmazott metrika az átlagos négyzetes hiba gyöke (Root Mean Square Error, RMSE), ami (33) módon adott.

$$RMSE = \sqrt{\frac{SSE}{n}} \tag{33}$$

A (33) formula előnye, hogy egy átlagos megfigyelési egységen elkövetett modellhibát ad vissza, az eredményváltozó mértékegységében. Gyakran nevezik reziduális szórásnak is (Ramanathan, 1992).

8.1.3. Koszinusz-hasonlóság

Az R^2 mutató $\mathrm{cor}\left(Y, \hat{Y}\right)^2$ módon adott definícióját alapul véve a koszinusz-hasonlóság teljesítménymetrikához jutunk el, ami lényegében a $\mathrm{cor}\left(Y, \hat{Y}\right)$ mutató nem 0-ra centrált verzióját jelenti (Raschka – Mirjalili, 2017).

$$Cos(Y, \hat{Y}) = \frac{\sum_{i=1}^{n} y_i \cdot \hat{y}_i}{\sqrt{\sum_{i=1}^{n} {y_i}^2} \cdot \sqrt{\sum_{i=1}^{n} {\hat{y}_i}^2}} \tag{34}$$

A (34) formulában nem történik meg az egyes megfigyelésekből az átlag levonása, tehát a változók 0-ra centrálása, azonban a változók vektortérben vett hosszával normálás megy végbe (34)-ben. Ezzel a mutató a $[-1, +1]$ intervallumon adja meg a modell becslési pontosságát, de a gyakorlatban Y és $\hat{Y}$ ellentétes irányú mozgásának, így negatív korrelációnak és koszinusz-hasonlóságnak elhanyagolható az előfordulása (Raschka – Mirjalili, 2017).

8.1.4. Az RMSPE mutató

Az $RMSE$ mutató egy relativizált változata az átlagos négyzetes relatív hiba gyöke (Root Mean Square Percentage Error, RMSPE). Az $RMSPE$ definíciója (35) lényegében abban tér el a klasszikus $RMSE$ mutatótól, hogy az SSE-ben az y_i és $\hat{y}_i$ értékek eltérését nem különbségvétellel, hanem hányadosképzéssel hasonlítja össze.

$$RMSPE = \sqrt{\frac{\sum_{i=1}^{n} \left(\frac{\hat{y}_i}{y_i} - 1\right)^2}{n}} \tag{35}$$

A (35) módon adott teljesítménymetrika így egy átlagos egyeden elkövetett relatív modellhibát ad meg, amit a legtöbbször százalék formátumban jelenítünk meg (Shcherbakov et al., 2013).

8.1.5. A MAE mutató és az abszolút érték, mint hibafüggvény

Az összes eddig taglalt teljesítménymetrika (R^2, $RMSE$, koszinusz-hasonlóság, $RMSPE$) közvetlenül vagy közvetetten kötődik a modell SSE értékéhez, ami a hibákat L_2 norma segítségével aggregálja. Ennek a megoldásnak hátránya, hogy így a hibamérés kimondottan érzékennyé válik a kiugró értékekre, hiszen a hibanövekedést négyzetesen bünteti. Ezzel könnyen előfordulhat, hogy az SSE-re épülő teljesítménymetrikák csak azért ítélnek nem elfogadhatónak egy modellt, mert relatíve kevés megfigyelésen kiugróan nagy hiba, ám a többi esetben kimondottan jól illeszkedik. Tekintsük azt az extrém esetet, hogy két modellt hasonlítunk össze egy olyan tesztmintán, ahol egy megfigyelésen mindkét modellnek kiugróan magas a hibája. Ekkor az SSE-alapú metrikák csak az egy kiugró pontra koncentrálhatnak olyan mértékben, hogy a két modell közti választás során a többi megfigyelést teljesen figyelmen kívül hagyják (Tangian – Gruber, 2002).

Az SSE-alapú teljesítménymetrikák kiugró hibaértékekre vett érzékenységét könnyen kiküszöbölhetjük az abszolút érték (L_1 norma) alkalmazásával a négyzetes hiba (L_2 norma) helyett (Wooldridge, 2016) hibaaggregálás során. A két norma közti váltást legkönnyebben az $RMSE$ mutatóban lehet megoldani (36) módon.

$$MAE = \frac{\left\|Y - \hat{Y}\right\|}{n} = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n} \tag{36}$$

A (36) formulával kapott átlagos abszolút hiba (Mean Absolute Error, MAE) jelentéstartalma az $RMSE$-vel hasonlatos: egy átlagos megfigyelési egységen elkövetett modellhibát ad vissza, az eredményváltozó mértékegységében. Azonban a modell illeszkedésének mérése során 1-2 kiugróan magas vagy alacsony hibára robusztusabban tud reagálni, hiszen $(y_i - \hat{y}_i)^2 \gg |y_i - \hat{y}_i|$ minden i-re (Shcherbakov et al., 2013).

8.1.6. A MAPE mutató

A MAE metrikából kiindulva könnyen készíthetünk egy relatív abszolút hiba mutatót is (Mean Absolute Percentage Error, MAPE). Azonban, ahogy (37)-ben is látszik az $RMSPE$-től eltérően a $MAPE$ metrikában a hibát nem hányadosképzés segítségével határozzuk meg, hanem a különbségként képzett hibát viszonyítjuk a ténylegesen megfigyelt y_i értékhez.

$$MAPE = \frac{\sum_{i=1}^{n} \left|\frac{y_i - \hat{y}_i}{y_i}\right|}{n} \tag{37}$$

Jelentéstartalomban viszont a $MAPE$ az $RMSPE$-vel azonosnak tekinthető: egy átlagos egyeden elkövetett relatív modellhibát ad meg (Shcherbakov et al., 2013). A különbség jelen esetben is abszolút hibának köszönhető: a $MAPE$ kevésbé érzékeny a szélsőséges esetek modellhibáira.

8.1.7. A Huber-féle teljesítménymetrika

Ugyanakkor bizonyos esetekben az L_1 és L_2 normára épülő teljesítménymetrikák alkalmazása sem célravezető. Tekintsünk egy olyan esetet, ahol a megfigyeléseink 90%-nak a valódi eredményváltozó értéke 150 és a maradék 10%-nak y_i értékei 0 és 30 között ingadoznak. Ebben a példában egy L_1 elvű metrika olyan modellt preferálna, ahol $\hat{y}_i = 150$ minden i-re. Ez abból is következik, hogy egy magyarázóváltozók nélküli „nullmodellben" a MAE minimalizálása az eredményváltozó mediánját adná eredményül (Tangian – Gruber, 2002). Ugyan ebben a példában az L_2 normára építő metrika rengeteg $\hat{y}_i$ becslést adna a $(0{,}30)$ intervallumban, hiszen a kiugró értékek felé torzít. Mindkét metrika-család nem kívánatos eredményekhez vezet. A probléma áthidalására vezették be a Huber-féle teljesítménymetrikát, ami (38) formulával adott (Natekin – Knoll, 2013).

$$e_i = \begin{cases} \frac{1}{2}(y_i - \hat{y}_i)^2, & |y_i - \hat{y}_i| \leq \delta \\ \delta|y_i - \hat{y}_i| - \frac{1}{2}\delta^2, & |y_i - \hat{y}_i| > \delta \end{cases}$$
$$Huber = \sum_{i=1}^{n} e_i \tag{38}$$

(38) alapján elmondhatjuk, hogy a Huber-metrika alapötlete, hogy L_1 elvű hibamérést alkalmaz egészen addig, amíg a megfigyelt hiba elég kicsivé nem válik, és onnantól L_2 normával számítja be a hibákat a metrikába. Az „elég kicsi hiba" értéket egy δ paraméter határozza meg, amit finom hangolni lehet például keresztvalidáció segítségével (Natekin – Knoll, 2013). Gyakorlatilag a δ paraméterben határozhatjuk meg, hogy milyen hibaértéket tekintünk kiugrónak. A Huber-metrikának nem tudunk olyan szemléletes jelentéstartalmat adni, mint az $RMSE$ vagy MAE értékeknek. Ugyanakkor, mivel a metrika a modellhibákat aggregálja, az igaz, hogy a kisebb metrika értékkel rendelkező modell a preferált.

8.1.8. A Log – Cosh teljesítménymetrika

Mivel a Huber-metrika teljesítménye nagyban függ a δ paraméter megválasztásától, így felmerült az igény egy olyan teljesítménymetrikára, ami megőrzi a Huber-metrika kedvező tulajdonságait (nagy hibák esetén L_1, kisebb hibák esetén L_2 normát alkalmaz) anélkül, hogy szükséges lenne egy szabadon megválasztható paraméter finomhangolása az alkalmazás során. Erre az igényre válaszként született meg a $Log - Cosh$ teljesítménymetrika (Raschka – Mirjalili, 2017).

$$Log - Cosh = \sum_{i=1}^{n} \ln(\cosh(y_i - \hat{y}_i)) \tag{39}$$

A metrika lényege (39) alapján, hogy a megfigyelésenkénti modellhibák koszinusz-hiberbolikusz transzformáltjának logaritmusát összegzi. A $\ln(\cosh(x))$ függvény nagyságrendileg az $\frac{x^2}{2}$-t közelíti kis x-ek esetén, míg az $|x| - \ln 2$ függvényt nagy x-ekre. Tehát, a $Log - Cosh$ teljesítménymetrika rendelkezik a Huber-metrika kedvező tulajdonságaival (alapvetően négyzetes hibaként viselkedik, de a ritkán előforduló nagy hibaértékek nem torzítják el az értékét), és szabad paraméter finomhangolását sem igényli. Szemléletes jelentéstartalmat a Huber-metrikához hasonlóan nem tudunk hozzá társítani, de (39) formulából adódóan ez is egy minimalizálandó metrika, hiszen a modellhibákat aggregálja.

8.1.9. Az RMSLE mutató

Egy további lehetséges alternatíva az SSE alapú teljesítménymetrikák kiugró érték érzékenységének kezelésére a megfigyelésenkénti hibák logaritmus transzformációja (Tangian – Gruber, 2002). Ezt az elvet alkalmazza az átlagos négyzetes logaritmikus hiba gyöke (Root Mean Squared Log Error, RMSLE).

$$RMSLE = \sqrt{\frac{\sum_{i=1}^{n}(\ln(y_i + 1) - \ln(\hat{y}_i + 1))^2}{n}} \tag{40}$$

A (40) formulában fontos feltételezés, hogy $y_i \geq 0$ és $\hat{y}_i \geq 0$ minden i-re. A $+1$ tagok biztosítják az $= 0$ esetek megengedését. A feltételezés a legtöbb gazdasági alkalmazásban helytálló, hiszen a legtöbb folytonos eredményváltozó pénzben kifejezett (jövedelem, bevétel, költség stb.), ahol a negatív értékek nem megengedettek. Könnyen észrevehető, hogy a $\ln(y_i + 1) - \ln(\hat{y}_i + 1) = \ln\left(\frac{y_i+1}{\hat{y}_i+1}\right)$ azonosság alkalmazásával az $RMSLE$ az átlagos relatív hiba logaritmusaként is felfogható (Tangian – Gruber, 2002).

Az $RMSLE$ mutató a logaritmus alkalmazásával robusztus a kiugró hibaértékekre, és egy üzleti alkalmazásokban sokszor hasznos tulajdonsággal rendelkezik: aszimmetrikusan bünteti az alá és felé becsléseket. Konkrétan, az eredményváltozó felülbecslését kevésbé bünteti, mint az alulbecslését (Shcherbakov et al., 2013). Tekintsük az $y_i = 1000$ érték esetét. Erre az értékre az első modellünk $\hat{y}_i = 600$, míg második modellünk $\hat{y}_i' = 1400$ becslést ad. $RMSE$-t vizsgálva mindkét becslés hibája $\sqrt{(1000 - 600)^2} = 400 = \sqrt{(1000 - 1400)^2}$. Ellenben az $RMSLE$ logikáját követve az első modell becslésének hibája $\sqrt{(\ln(1000 + 1) - \ln(600 + 1))^2} = 0{,}510$, míg a második (a felülbecslő) modell esetén a hiba $\sqrt{(\ln(1000 + 1) - \ln(1400 + 1))^2} = 0{,}336$. Az $RMSLE$ aszimmetrikus büntetése

a felülbecslő modellek javára kimondottan hasznos tud lenni olyan gazdasági életben gyakori felügyelt tanulási problémák esetén, ahol az eredményváltozó valamiféle költséget kifejező mennyiség. Például, ha egy ételkiszállítással foglalkozó vállalatnál kívánjuk megbecsülni egy kiszállítás várható idejét, akkor felülbecslés esetén a hibának annyi következménye van, hogy a futár kap némi extra pihenőidőt. Ellenben, alábecslés esetén a futár nem szállítja ki időben a rendelést, így a hiba ügyfélelégedetlenséghez vezethet.

8.1.10. A folytonos célváltozó esetén alkalmazott teljesítménymetrikák alkalmazása a numerikus kísérletek során

Jelen értekezésben a szakirodalomban legnépszerűbb, és szemléletes jelentéstartalmú (relatív magyarázóerő) R^2 teljesítménymetrika alapján értékeljük ki elsősorban a folytonos eredményváltozóval bíró betongerendák adatbázisra épített modelleket. A fejezetben szereplő többi teljesítménymetrika segítségével az eredmények robusztusságát vizsgáljuk. Ugyanis fontos megvizsgálni, hogy a HGHK algoritmus által javasolt concurvity jelenségtől mentes modellek becslési pontossága mely metrikákban mennyire közelíti meg a magyarázóváltozóban kevésbé takarékos modellekét.

Mivel az adattisztítás során a kiugró értékektől megtisztítjuk az adatbázist, így várhatóan az L_1 és L_2 normákat alkalmazó teljesítménymetrikák preferenciái között nem lesz érdemi különbség. Ugyanakkor, a biztonság kedvéért érdemes minden metrika alapján összevetni a modelleket, hiszen az eredményváltozó kiugró értékeinek szűrése még nem garantálja, hogy a modellhibák esetén sem fog néhány megfigyelés kiugró értéket produkálni.

Érdemi különbséget várunk viszont az $RMSPE$ és $RMSLE$ metrikák által preferált modellek körében a többi vizsgált mutatóhoz képest. Mivel ez a két teljesítménymetrika a becsült és valós eredményváltozó értékek relatív eltéréséből, $\frac{\hat{y}_i}{y_i}$-ből és nem azok különbségéből ($y_i - \hat{y}_i$) indul ki.

8.2. Teljesítménymetrikák Bernoulli-eloszlású célváltozó esetén

Egy Bernoulli-eloszlású célváltozónak csak kétféle lehetséges realizációja lehetséges: 1 (bekövetkezett a vizsgált esemény) vagy 0 (nem következett be a vizsgált esemény). Ebből adódóan a Bernoulli-eloszlású Y változók esetén a felügyelt tanulást megvalósító modellek közvetlenül csak a $P(Y = 1|\tilde{X}) = p$ feltételes valószínűségre adnak becslést (James et al., 2013). Tehát, megadják, hogy mi az $Y = 1$ esemény bekövetkezési valószínűsége, feltéve, hogy ismerjük a $\tilde{X} = \{X_1, X_2, \dots, X_p\}$ halmazban lévő magyarázóváltozók értékét a teszthalmazban szereplő megfigyelésekre.

Következésképpen, a Bernoulli-eloszlású célváltozó esetén az $\hat{Y}$ becsült célváltozót csak egy vágási érték (cut-off) megadásával lehet előállítani. Jelölje most ezt a vágási értéket c. A c vágási érték ismeretében azt mondhatjuk, hogy

$$P(y_i = 1|\tilde{X}) > c \rightarrow \hat{y}_i = 1$$

és

$$P(y_i = 1|\tilde{X}) \leq c \rightarrow \hat{y}_i = 0$$

Ezzel a definícióval pedig előáll az $\hat{Y}$ becsült célváltozó (Raschka – Mirjalili, 2017).

8.2.1. A klasszifikációs mátrix és az ebből közvetlenül származtatható mutatók

A modellek kiértékeléséhez alkalmazott teljesítménymetrikák megadásához be kell vezetnünk a klasszifikációs vagy konfúziós mátrixot, ami nem más, mint az Y és $\hat{Y}$ változók kétdimenziós gyakorisági táblája (kontingenciatáblázat) a teszthalmazon (Powers, 2011). A mátrix általános alakját az 1. táblázat szemlélteti.

$\hat{Y}/Y$	**1**	**0**
1	tp	fp
0	fn	tn

1. táblázat: Egy klasszifikációs/konfúziós mátrix általános szerkezete. Forrás: Saját szerkesztés.

Az 1. táblázatban alkalmazott jelölések értelmezése mögött egy olyan elnevezési konvenció áll, mely szerint a Bernoulli-eloszlás 1 értékeit hívjuk „pozitív” osztálynak, míg 0 értékeit „negatív osztálynak (Powers, 2011). Ezzel a konvencióval az 1. táblázat jelölései az alábbi jelentéstartalommal bírnak:

- tp: „true positive” esetek száma. A modell által helyesen pozitív osztályba sorolt elemek gyakorisága. $\#(\hat{Y} = 1, Y = 1)$
- tn: „true negative” esetek száma. A modell által helyesen negatív osztályba sorolt elemek gyakorisága. $\#(\hat{Y} = 0, Y = 0)$
- fp: „false positive” esetek száma. A modell által tévesen pozitív osztályba sorolt elemek gyakorisága. $\#(\hat{Y} = 1, Y = 0)$
- fn: „false negative” esetek száma. A modell által tévesen negatív osztályba sorolt elemek gyakorisága. $\#(\hat{Y} = 0, Y = 1)$

Az összes valóságban is pozitív eset számát tesztmintán a következőképpen állíthatjuk elő a klasszifikációs mátrixból: $\#(Y = 1) = P = tp + fn$. Hasonlóan az összes valójában negatív eset száma: $\#(Y = 0) = N = tn + fp$.

A klasszifikációs mátrix alapján több alapvető teljesítménymetrika is megadható a Bernoulli-eloszlású eredményváltozóra adott becslések pontosságának kiértékelésére, melyeket Powers (2011) alapján foglalunk össze az alábbiakban.

- ACC (Accuracy): $ACC = \frac{tp+tn}{P+N}$. A modell teljes pontossága (angolul accuracy-ja). Megadja, hogy a tesztmintában szereplő megfigyelések hányad részét sorolja a modell helyesen pozitív, illetve negatív osztályba.
- Precision: A modell precizitását (angolul precision) pozitív osztályra $\frac{tp}{tp+fp}$, míg negatív osztályra $\frac{tn}{tn+fn}$ módon számítjuk. Megadja, hogy a modell pozitív (negatív) osztályú $\hat{Y}$ becsléseinek hányad része tartozik Y alapján is a pozitív (negatív) osztályba.
- Recall: A modell „visszaemlékezését” (angolul recall) pozitív osztályra $\frac{tp}{P}$, míg negatív osztályra $\frac{tn}{N}$ módon számítjuk. Megadja, hogy az Y alapján is a pozitív (negatív) osztályba tartozó megfigyelések hányad részét azonosítja a modell helyesen pozitív (negatív) osztályba tartozónak. Az angol szakirodalomban a pozitív osztály recall értéke gyakran szerepel True Positive Rate (TPR, magyarul valós pozitív ráta), míg a negatív recall értéke True Negative Rate (TNR, magyarul valós negatív ráta) néven is.
- FPR (FNR): A téves pozitív (negatív) ráta (false positive (negative) rate, FPR (FNR))) megadja, hogy az Y alapján pozitív (negatív) osztályban szereplő megfigyelések hányad részét sorolja a modell tévesen negatív (pozitív) osztályba. Formula szintjén mindez a $FPR = \frac{fp}{N} = 1 - TNR$ és $FNR = \frac{fn}{P} = 1 - TPR$ összefüggéseket jelenti.

A definíciók alapján a modellek összehasonlításakor az ACC, Precision és Recall teljesítménymetrikák maximalizálandók, míg a FPR és FNR metrikák minimalizálandók. Mindegyik metrika egy arányszámként értelmezhető, így értékeik a $[0,1]$ intervallumban mozoghatnak.

8.2.2. A kiegyensúlyozott Accuracy mutató

Kézenfekvőnek adódhat egy modell általános teljesítményét az ACC metrikával értékelni, hiszen jelentéstartalma nagyon könnyen rokonítható a folytonos változók esetén leggyakrabban alkalmazott R^2 metrikával: a modell relatív magyarázóerejét adja meg. Ugyanakkor az Accuracy mérték használata kiegyensúlyozatlan eredményváltozó esetén kimondottan félrevezető, torzított eredményre vezet (Williams, 2021). Vizsgáljunk meg egy olyan példát, ahol a valódi Y értékek 95%-a negatív, 5%-a pozitív osztályba tartozik. Minden megfigyelés negatív osztályba sorolása 95%-os ACC metrikát eredményez. Ám a pozitív osztály recall

értéke 0-nak és a FPR mérték 1-nek adódik (precision mutató pedig nem is értelmezhető a pozitív osztályra).

Kiegyensúlyozatlan esetekben érdemes a modellek teljesítményét az üzleti szituációban prioritást élvező osztály precision és/vagy recall metrikák alapján kiértékelni. Ugyanakkor, ha mindenképpen a modell teljes és nem csak egy kitüntetett osztályban mért relatív becslési pontosságáról szeretnénk képet kapni, akkor a szakirodalom két további gyakran alkalmazott mutatót javasol: a kiegyensúlyozott pontosságot (balanced Accuracy, bACC) és az F_1 mutatót (Williams, 2021).

A kiegyensúlyozott pontosság metrika egyszerűen az eredményváltozó mindkét osztályára vonatkozó recall metrikák számtani átlaga: $bACC = \frac{TPR+TNR}{2}$. A $bACC$ metrika értelmezhető a modell általános relatív pontosságaként, hiszen megadja, hogy átlagosan az Y pozitív és negatív osztályainak elemeit hányad részben azonosítják helyesen az $\hat{Y}$ becslések. Ugyanakkor, a mutató nem fogja a korábbi példánkban a csak negatív osztályú $\hat{Y}$ becsléseket adó modellt preferálni. Egy ilyen esetben a $bACC = \frac{0+1}{2} = 0{,}5$ eredmény adódik, ami a véletlen találgatás pontosságával ekvivalens az egyszerű ACC metrika esetén kiegyensúlyozott eredményváltozóra. A $bACC$ metrika kiterjeszthető súlyozás segítségével: nem feltétlenül szükséges a TPR és TNR mutatókat azonos mértékben preferálnunk az átlagolás során. Mivel a súlyok megválasztása általában az alkalmazási terület szakértői szempontjai alapján történik (Williams, 2021), így ennek vizsgálatától jelen értekezés módszertani fókusza miatt eltekintünk.

8.2.3. Az F_1 teljesítménymetrika

Az F_1 teljesítménymetrika definíció szerint a pozitív osztály precision és recall értékeinek harmonikus átlaga (Raschka – Mirjalili, 2017) (41).

$$F_1 = \frac{2}{\frac{1}{recall} + \frac{1}{precision}} = 2 \cdot \frac{precision \cdot recall}{precision + recall} = \frac{tp}{tp + 0{,}5 \cdot (fp + fn)} \quad (41)$$

Az F_1 metrika kihasználja, hogy pozitív osztály precision és recall metrikája jellemzően egymás rovására növelhető: ha a modell $\hat{Y}$ becslései a valós Y szerinti pozitív osztályú megfigyelések magas hányadát helyesen azonosítják, akkor várhatóan a pozitív osztályú $\hat{Y}$ becslések kisebb hányada lesz Y szerint is pozitív osztályú (a pozitív osztály precision metrikája csökken). Tehát TPR növekedésével az FPR is jellemzően nő, mivel várhatóan az $\hat{Y}$ becslések közül több lesz a pozitív osztályú, és a többletnek nem minden része lesz Y szerint is pozitív osztályú. Az F_1 metrika így közvetetten e két szempont közötti egyensúlyt törekszik megteremteni:

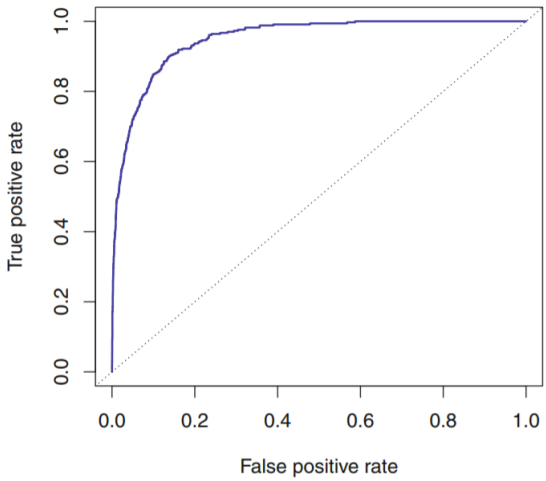
magas TPR elérése a lehető legkisebb FPR növekedéssel (Powers, 2011). Ezzel kontrolálva a kiegyensúlyozatlan célváltozó esetére. A korábbi példánk esetében az F_1 metrika 0 (vagy nem értelmezhető) lenne. Hiszen, a 95%-5% pozitív-negatív megoszlás során adott tiszta negatív osztályú $\hat{Y}$ becslések esetében a pozitív osztály recall értéke 0, míg a precision mutató nem is értelmezhető.

Az F_1 mutató is általánosítható súlyozás segítségével, hiszen a harmonikus átlag képzése során az F_1 formulájában a precision és recall mértékek azonos súlyt képviselnek. Az F_β metrika segítségével eltérő súlyokat rendelhetünk a két szempontnak. mivel a súlyok megválasztása a $bACC$ esetéhez hasonlóan az alkalmazási terület szakértői szempontjai alapján történik, így az értekezés módszertani fókusza miatt a különböző súlyozási sémák vizsgálatától az F metrika esetében is eltekintünk.

8.2.4. A ROC görbe

Az eddig áttekintett teljesítménymetrikák Bernoulli célváltozó esetére egyaránt egy fix c vágási érték mellett előálló klasszifikációs mátrix alapján kerültek meghatározásra. Ebből adódik, hogy szükséges egy módszert adni c optimális meghatározására. A feladat megoldására a szakirodalomban legnépszerűbb módszer a ROC (Receiver Operating Characteristic) görbe (Powers, 2011) (Williams, 2021).

A ROC görbe működését Powers (2011) alapján ismertetjük. A módszer alapötlete, hogy egy felügyelt gépi tanuló modellből a tesztminta megfigyeléseire közvetlenül nyert $P(Y = 1|\tilde{X})$ feltételes valószínűségek tapasztalati eloszlásainak percentiliseit, mint potenciális c vágási értékeket tekintjük. Ezen a lehetséges c értékek mindegyikére előállítjuk a klasszifikációs mátrixot, majd meghatározzuk belőle a TPR és FPR mutatókat. Ezeket az eredményeket pedig ábrázoljuk egy diagramon, ahol az x koordinátákat a különböző vizsgált c-k melletti FPR, míg az y koordinátákat a TPR értékek adják (9. ábra).



9. ábra: Egy tetszőleges felügyelt gépi tanuló modell ROC görbéje. Forrás: James et al., 2013, p.148.

A 9. ábrán látható, hogy a ROC görbe mellett egy 45 fokos egyenest is megjelenítünk a pontdiagramon. Ez egy olyan modell ROC görbéje, amely minden megfigyelésre $P(Y = 1|\tilde{X}) = 0{,}5$ becsléseket ad. Tehát, ez az egyenes a „véletlen találgatás" ROC görbéje. Ezzel szemben a valamilyen c mellett tökéletes osztályozást végző modell ROC görbéjének át kell mennie a $(0{,}1)$ ponton, hiszen ebben a pontban $TPR = 1$ és $FPR = 0$, azaz minden egyedet tökéletesen osztályozott a modell pozitív és negatív osztályokba.

Ebből adódóan a ROC-görbe alapján azt a c értéket tekinthetjük optimálisnak, amely alapján adódó ROC görbe pont a lehető legközelebb van a $(0{,}1)$ ponthoz. Hiszen ennél a c-nél érjük el a legmagasabb TPR-t minimális FPR mellett. Itt találjuk meg $\hat{Y}$ becsléseinkkel a pozitív osztályú Y-ok legnagyobb hányadát anélkül, hogy a negatív osztályú Y-ok közül az $\hat{Y}$ becslések túl nagy hányadot sorolnának tévesen pozitív osztályba. Ezzel lényegében a ROC görbe vizsgálata során közvetlenül optimalizálunk az F_1 metrikában közvetetten megjelenő TPR és FPR közti ellentét feloldására. Ebből adódóan pedig a ROC görbe alkalmazható kiegyensúlyozatlan eredményváltozó esetén is az F_1 metrikához hasonlóan.

Az optimális c vágási érték megválasztása mellett a ROC görbe használható egy vágási ponttól független teljesítménymetrika megalkotására is. Hiszen, egy teljesen tökéletesen osztályozó modell ROC görbéjén minden c érték mellett kapott pontja a $(0{,}1)$ pontban kell, hogy elhelyezkedjen. Ekkor pedig a ROC görbe egy egységnyi oldalú négyzetet alkot, melynek területe 1. Két Bernoulli-eloszlású célváltozóra adott modell közül tehát az tekinthető jobbnak, amely ROC görbéje alatti terület minél közelebb van az egységnégyzet területéhez. Tehát, a ROC görbe alatti terület (Area Under the ROC Curve, AUC) egy vágási ponttól független modellteljesítményt minősítő metrikaként alkalmazható. Az AUC metrika a $[0{,}1]$ tartományban mozog, és maximalizálandó. A gyakorlatban azonban ritka, hogy értéke 0,5 alá essen, hiszen

ahogy a 9. ábrán is láttuk: a „véletlen találgató modell” ROC görbéje egy 45 fokos egyenes, amely alatti terület az egységnégyzet területének fele, azaz 0,5. A gyakorlatban pedig ritka a véletlen találgatásnál pontatlanabbul becslő felügyelt gépi tanuló modell.

8.2.5. A kereszt-entrópia

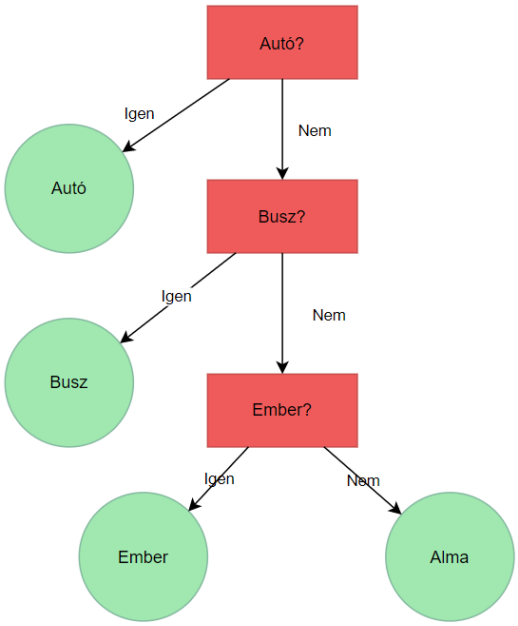
Az AUC metrika mellett a szakirodalomban népszerű, vágási értéktől függetlenül alkalmazható teljesítménymetrika Bernoulli célváltozó esetén a kereszt-entrópia (Williams, 2021). A metrika a Shannon-féle entrópia fogalmára támaszkodik. A kereszt-entrópia működését Borda (2011) és Murphy (2012) alapján ismertetjük.

Az entrópia fogalma a fizikából eredeztethető, és egy rendszer instabilitásának fokát fejezi ki. Külső energiabevitel nélkül egy fizikai rendszer entrópiája növekszik („a rendszer a káosz felé halad”). Például, egy téglakupacból és több zsák cementből az út szélén nem épül ház spontán módon. Ugyanakkor, egy ház lassan elpusztul karbantartás nélkül, míg a végén csak cementpor és téglatörmelék marad a helyén.

Shannon vezette be az entrópia fogalmát az információ „mennyiségének” kifejezésére. Legyen A egy halmaz a következő elemekkel:

$$A = \{autó, autó, autó, autó, autó, autó, busz, busz, busz, busz, ember, alma\}$$

A halmazban 12 elem van, és ha véletlenszerűen visszatevéses módon húzunk a halmazból egy elemet, akkor az egyes elemek kihúzási valószínűsége a következő valószínűségi eloszlás szerint alakul: $P(autó) = \frac{1}{2}$, $P(busz) = \frac{1}{3}$, $P(ember) = \frac{1}{12}$, $P(alma) = \frac{1}{12}$. Tegyük fel, hogy egy húzás során tudjuk milyen elemek szerepelnek a halmazban, ám a kihúzott elemet nem látjuk. Viszont, kérdéseket tehetünk fel annak eldöntésére, hogy milyen elemet húztunk A-ból. Egy ilyen kérdéssorozat lehet például a 10. ábrán látható struktúra.



10. ábra: Egy lehetséges kérdéssorozat az A halmazból húzott elem meghatározására. Forrás: saját szerkesztés.

A 10. ábrán látható kérdésstruktúrában egy kérdésből kitaláljuk, ha autót húztunk, két kérdésből, ha buszt és három kérdésből jöhetünk rá, ha embert vagy almát húztunk ki. Ha ezeket a kérdésszámokat az egyes elemek kihúzási valószínűségével súlyozva összegezzük, akkor megkapjuk a 10. ábrán látható kérdésstruktúrában az egy tetszőleges húzás során feltett kérdések várható számát: $1 \cdot \frac{1}{2} + 2 \cdot \frac{1}{3} + 3 \cdot \frac{1}{12} + 3 \cdot \frac{1}{12} = 1{,}666667$.

Természetesen a kérdéseket más sorrendben is feltehetjük, mint ahogy a 10. ábrán szerepel. A Shannon-féle entrópia a különböző kérdéssorendek szerinti várható kérdésszám alsó határát adja meg (42).

$$H(A) = -\sum_{i=1}^{k} [p_i \log_2(p_i)] \tag{42}$$

(42)-ben p_i az i-edik elem kihúzási valószínűségét jelöli az A halmazból, míg k az A halmaz elemeinek számát. Ez a példánkban szereplő A halmazra $-\left(\frac{1}{2} \cdot \log_2\left(\frac{1}{2}\right) + \frac{1}{3} \cdot \log_2\left(\frac{1}{3}\right) + \frac{1}{12} \cdot \log_2\left(\frac{1}{12}\right) + \frac{1}{12} \cdot \log_2\left(\frac{1}{12}\right)\right) = 1{,}625815$. Tehát ennyi a minimálisan várhatóan szükséges kérdések száma a húzott elem meghatározásához.

(42) alapján $0 \leq H(A) \leq \log_2 n$. Hiszen, az entrópia akkor 0, ha létezik $p_j = 1$ és minden $p_{i \neq j} = 0$. Tehát a halmazból történő húzás során 1 valószínűséggel mindig ugyanazt az elemet húzzuk. Ekkor a húzásnak nincs bizonytalansága, azaz entrópiája. Az entrópia akkor a lehető

legnagyobb ($\log_2 n$), ha minden $p_i = \frac{1}{n}$, azaz minden elemet egyenlő valószínűséggel húzunk ki.

A kereszt-entrópia a Shannon-féle entrópia általánosításának tekinthető két halmazra. A kereszt-entrópia két halmaz eloszlásának távolságát adja meg. Ezt egy felügyelt tanuló modell kiértékelésére Bernoulli-eloszlású célváltozó esetén úgy tudjuk alkalmazni, hogy megnézzük a teszthalmaz Y és $\hat{Y}$ változói által adott halmazok eloszlásának távolságát. Tekintsünk egy egyszerű, 3 elemű teszthalmazt, ahol $Y = (0; 1; 0)$. Az $\hat{Y}$ becsléseket most tekintsük közvetlenül a modellünk által a teszthalmaz elemeire adott $P(Y = 1|\tilde{X})$ feltételes valószínűségek halmazának: pl. $\hat{Y} = (0{,}1; 0{,}8; 0{,}1)$. Ekkor a kategorikus kereszt-entrópia (43) formula segítségével meghatározható.

$$H(Y, \hat{Y}) = -\sum_{i=1}^{n} [y_i \ln(\hat{y}_i)] \tag{43}$$

A természetes alapú logaritmus használata csak kényelemből történik, hiszen a logaritmus azonosságok miatt $\log_2 n = \frac{\ln n}{\ln 2}$. Ezzel (43) csak egy konstans ($\ln 2$) nevezővel térne el az általunk megadott képlettől.

Minél kisebb a két halmaz eloszlása közti entrópia, annál közelebbnek tekinthető a két halmaz elemeinek eloszlása egymáshoz. Az érték akkor lesz 0, ha a két eloszlás teljesen megegyezik. Ebből adódóan a kereszt-entrópia egy minimalizálandó teljesítménymetrika. Előnye az AUC-hoz hasonlóan, hogy anélkül teszi összehasonlíthatóvá az alkalmazott modellek becslési pontosságát, hogy külön vágási értéket kelljen alkalmazni. Emiatt a metrika alkalmazható kiegyensúlyozatlan célváltozó esetén is.

8.2.6. A Bernoulli-eloszlású célváltozó esetén alkalmazott teljesítménymetrikák alkalmazása a numerikus kísérletek során

Mivel a banki ügyfelek adatbázis célváltozója kiegyensúlyozatlan (24% jelentett csődöt hitelkártya tartozására), így jelen értekezésben elsődleges teljesítménymetrikaként a ROC görbe alatti területet (AUC) alkalmazzuk a Bernoulli-eloszlású célváltozó esetén. Ugyanis, ez a metrika a szakirodalomban a legnépszerűbb kiegyensúlyozatlan esetben (Williams, 2021). Ennek oka, hogy a metrika nem igényli vágási érték alkalmazását, és a kereszt-entrópiával ellentétben a modell becslési pontosságát egy jól határolt $[0{,}1]$ intervallumon belül fejezi ki. Továbbá, a metrika mögött álló ROC görbe alkalmas optimális vágási érték megválasztására a többi bemutatott teljesítménymetrikához.

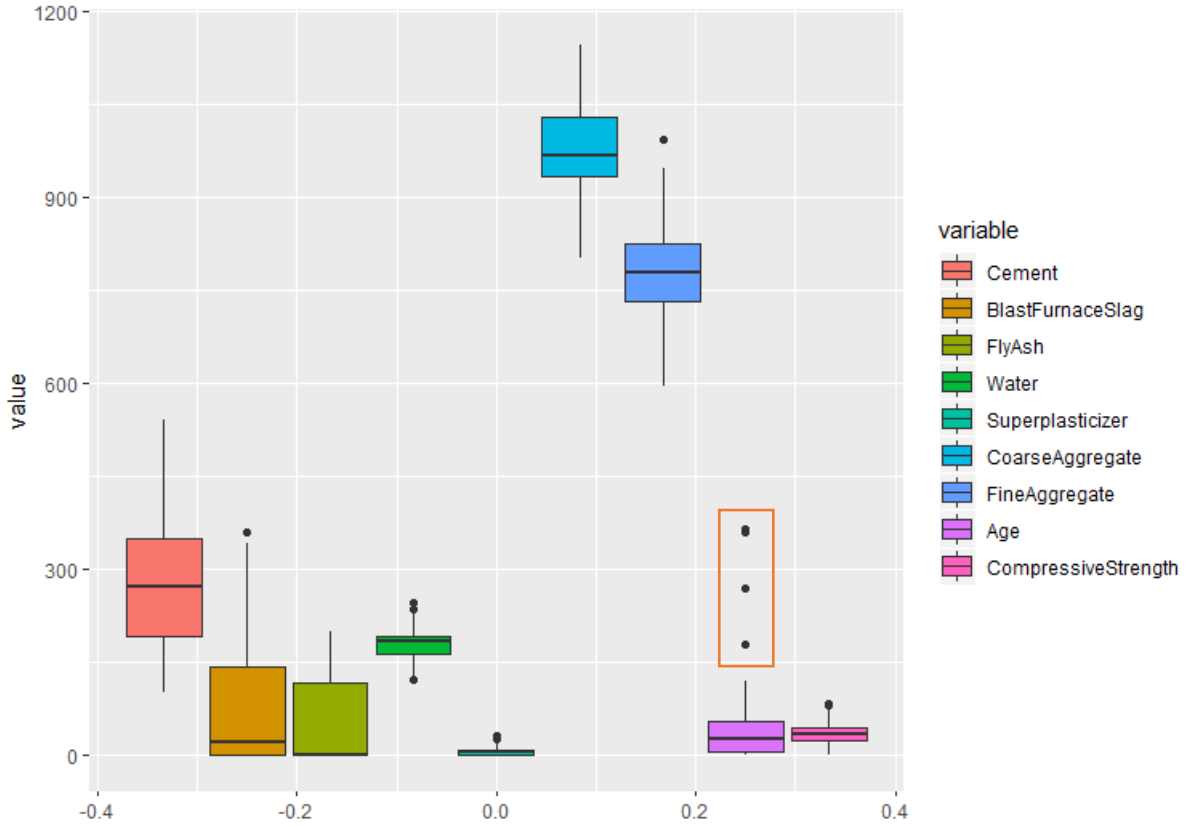
A modellek vizsgálatát nem hanyagolhatjuk el az AUC-on túli metrikák szerint sem. Hiszen a csődelőrejelzés üzleti alkalmazásaiban kimondottan fontos információ lehet a banki vezetés szempontjából, hogy a jövőben csődöt jelentő ügyfelek mekkora hányadát képes azonosítani a modellünk (recall a csőd osztályra), vagy, hogy a potenciálisan csődöt jelentő ügyfélnek becsült megfigyelések esetében mekkora a „hamis riasztások" aránya (precision a csőd osztályra). A modellműködés több teljesítménymetrika szerinti összehasonlításából származó hasonló információk nélkül a vezetőség nem tudja megadni az optimális akciótervet a potenciálisan csődöt jelentő ügyfélkör kezelésére (Moula, 2017).

9. A HGHK algoritmus paramétereinek finomhangolása egy kis méretű feladaton

A betongerendák adatbázis változóinak jelentése az alábbi:

- Cement – a betongerenda cement tartalma kg/m^3-ben
- Blast Furnace Slag – a betongerenda salak tartalma kg/m^3-ben
- Fly Ash – a betongerenda pernye tartalma kg/m^3-ben
- Water – a betongerenda víztartalma kg/m^3-ben
- Superplasticizer – a betongerenda folyósítóanyag tartalma kg/m^3-ben
- Coarse Aggregate – a betongerenda durva aggregátum tartalma kg/m^3-ben
- Fine Aggregate – a betongerenda finom aggregátum tartalma kg/m^3-ben
- Age – a betongerenda kora napokban = gyártástól eltelt napok száma
- Compressive Strength – a betongerenda nyomószilárdsága MegaPascalban (célváltozó)

Az adatbázisokon minimális adattisztítási lépéseket kellett elvégezni. A változók dobozábrája (11. ábra) alapján egy-két látványosan „öreg” gerendát fedezhetünk fel. Ezeket a gerendákat elhagyjuk, mint kiugró értékek, így 916 megfigyelés maradt az adatbázisban.



11. ábra: A betongerendák adatbázis változóinak doboz ábrája. A kor (Age) változóban azonosított kiugró értékek keretezve. Forrás: saját szerkesztés.

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 $R^2 = \mathrm{cor}(Y, \hat{Y})$ 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.

Az adattisztítási lépések és a vizsgált algoritmusok implementálása és futtatása az R 3.5.3 verziójában történt a 6. fejezetben hivatkozott csomagok segítségével. 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).

Az mRMR algoritmus esetében felhasználjuk azt az a priori tudást, hogy a globális optimumban $|\tilde{X}| = 4$.

A HGHK algoritmus esetén a célfüggvény a korrigált McFadden-féle pszeudo R^2 mutató. A concurvity korlátot a pesszimista mérték alapján vizsgáljuk és a $\Psi_{k_j} = \mathbf{0}$ nullhipotézis vizsgálatához választott szignifikancia-szint 5%. A k_j értékek szabályozásához használt Augustin et al. (2012)-féle próbák esetén a választott szignifikancia-szint 1%. 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.

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

9.1. Az vizsgált hagyományos algoritmusok futási eredményei

A 6. fejezetben ismertetett, GAM keretben működő változószelekciós algoritmusok és a HGHK algoritmus eredményeit a kezdeti paraméterekkel a betongerendák adatbázison a 2. táblázat tartalmazza.

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

Algoritmus	Kiválasztott változók	Concurvity korlátot sértő változók	R^2 (Tesztminta)	Átlagos futásidő (sec)	Futásidők szórása (sec)
Teljes modell	mindegyik	mindegyik, kivéve Age	88,119%	2,663	0,046
COSSO	Cement, BlastFurnaceSlag, FlyAsh, Water, Age	mindegyik, kivéve Age	85,045%	3,983	0,515
Büntetőtagos thin plate spline (23) formulával	mindegyik	mindegyik, kivéve Age	88,077%	9,902	0,253
Büntetőtagos thin plate spline $\tilde{S}_j$ mátrixszal	mindegyik	mindegyik, kivéve Age	87,694%	3,313	0,114
Nemnegatív Garotte módszer	mindegyik	mindegyik, kivéve Age	78,200%	0,270	0,120
Stepwise (AIC)	mindegyik	mindegyik, kivéve Age	88,119%	2,794	0,064
Backward (szignifikancia)	mindegyik kivéve CoarseAggregate	mindegyik, kivéve Age	88,107%	3,889	0,093
GAMBoost	mindegyik	mindegyik, kivéve Age	89,239%	507,104	18,506
Módosított backfitting	Cement, BlastFurnaceSlag, Superplasticizer, CoarseAggregate, Age	Cement, Superplasticizer	87,952%	0,835	0,527
mRMR	Cement, BlastFurnaceSlag, Superplasticizer, Age	-	81,580%	0,068	0,037
HSIC-Lasso	Cement, BlastFurnaceSlag, Water, Superplasticizer, Age	Water Superplasticizer	86,382%	0,404	0,095
HGHK algoritmus	*Cement, Blast Furnace Slag, Water, Age*	-	*84,363%*	*111,026*	*17,460*

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

Továbbá, a Mellékletekben található 1. táblázat alapján elmondható, hogy a HGHK algoritmus által legtöbbször (12/30 = 0,4 = 40%) adott megoldás a globális optimum a szignifikancia és concurvity korlátokkal adott változószelekciós feladatra.

A legtöbb változószelekciós algoritmus a teljes modellt preferálja, vagy csupán egy változót hagy el. Ennek következtében mindegyik változó ezekben a modellekben megsérti a concurvity korlátokat a gerenda kora (Age) kivételével. Tehát, ezekből a modellekből nem igazán lehet elkülöníteni, hogy a gerendák különböző kémiai összetevőinek alakulása hogyan befolyásolja azok nyomószilárdságát. A teljes modelleket preferáló algoritmusok tesztmintán mért teljesítményeiben jelentős különbséget nem fedezhetünk fel. Egyedül a b-spline függvényeket klasszikus módon illesztő nemnegatív Garotte módszer teljesítménye marad el látványosan (kb. 10 százalékponttal) a többi teljes modellt preferáló algoritmus teljesítményétől.

Kiemelendő a GAMBoost algoritmus teljesítménye, ami a legjobb R^2 értéket adja a tesztmintán, bár ez csak egy százalékponttal magasabb a többi algoritmus értékénél, ami a teljes modellt preferálja. Ellenben az algoritmus futásideje látványosan magas. Ennek az oka elsősorban az, hogy egy l iterációban csak egy X_j magyarázóváltozónak frissítjük az $f_{j,(l)}$

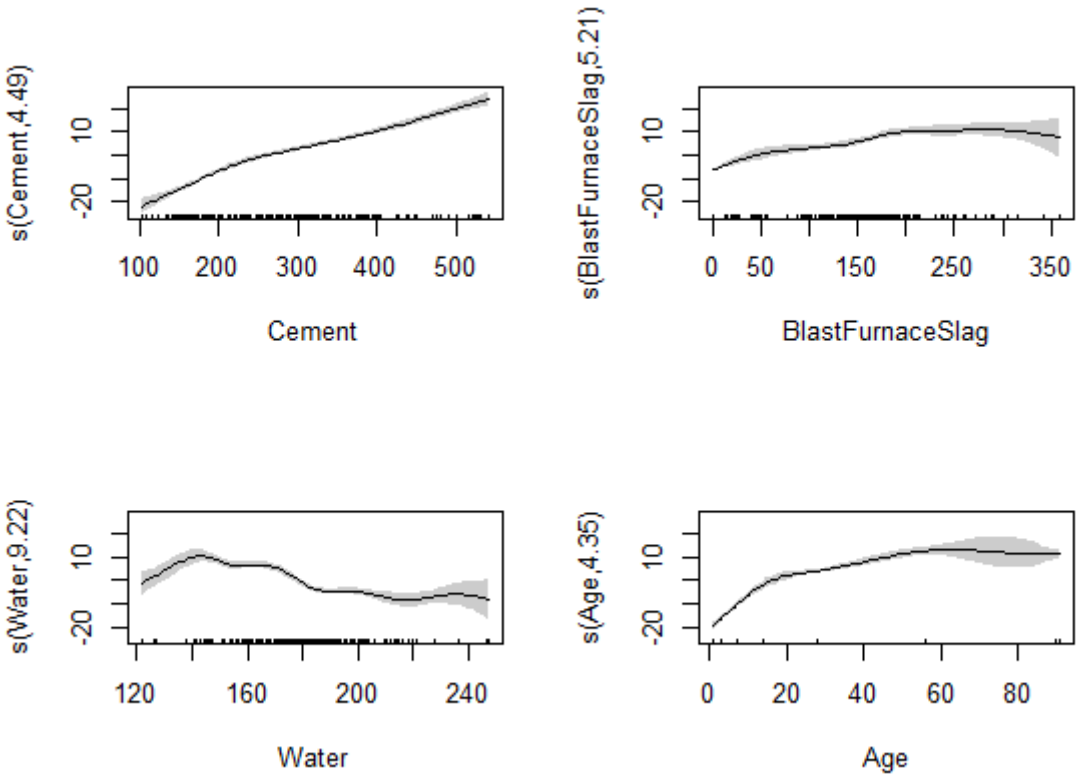
függvényét a devianciajavítás alapján. Ebből adódóan a nagyon hasonló működési elvű, de egy l iterációban akár párhuzamosan több $f_{j,(l)}$ függvényt is frissítő módosított backfitting eljárással ellentétben a GAMBoost algoritmusban nem jól párhuzamosíthatók az egy l iterációban végrehajtott műveletek. Továbbá, több L maximális lépésszám mellett kell lefuttatni az algoritmust, hogy végül 10-szeres keresztvalidációs eljárással kiválasszuk a legjobban illeszkedő modellt. A módosított backfitting esetében elég a $\hat{Y}_{(l)} = \hat{Y}_{(l-1)}$ kritérium teljesülésekor rögtön megállni, ami ráadásul jóval kevesebb iterációból is elérhető amiatt, hogy az eljárás egy iterációjában egyszerre több X_j magyarázóváltozónak is frissíthetjük az $f_{j,(l)}$ függvényét. Ez a két különbség látványosan megjelenik a két algoritmus futásidejében. A GAMBoost várhatóan több, mint 8 percig fut, a módosított backfitting várható futásideje pedig a 95%-os konfidencia intervallum felső határa esetén is mindössze $0{,}835 + t_{0{,}975}^{30-1} \cdot \frac{0{,}527}{\sqrt{30}} = 1{,}032$ másodperc.

A COSSO, módosított backfitting, mRMR és HSIC-Lasso algoritmusok szűkebb modelleket preferálnak, ám az mRMR megoldása kivételével mindegyik modell változói között concurvity jelenség áll fenn. A legrosszabb helyzetben a COSSO algoritmus modellje van, aminek az Age kivételével minden változója érintett a concurvity jelenségben, és a tesztmintán mért R^2 értéke is észrevehetően kisebb, mint a concurvity szempontjából hasonlóan rossz viselkedésű modelleké.

A módosított backfitting és HSIC-Lasso modellek a COSSO-hoz hasonlóan 5 db magyarázóváltozót használnak, de ezekből nem 4, hanem csak 2-2 produkál káros concurvity jelenséget. Továbbá, a modellek R^2 értékei meghaladják a COSSO modell értékét. Különösen figyelemre méltó a módosított backfitting teljesítménye, ami 5 változóval felülmúlja a 8 változót használó, $\widetilde{\boldsymbol{S}}_{\boldsymbol{j}}$ mátrixszal illesztett thin plate spline-okat alkalmazó GAM-ot.

A módosított backfitting és a HSIC-Lasso modellek alaposabb vizsgálatából megérthető, hogy elsősorban mi okozza a káros concurvity jelenséget a vizsgált betongerendák kémiai összetevőit leíró változók között. A HSIC-Lasso modellben korlátokat a Water és a Superplasticizer változók sértik meg, míg a módosított backfitting modellben a Cement és Superplasticizer változók. Ennek az oka, hogy az építőipari betongerendákban felhasznált folyósítóanyag (Superplasticizer) mennyisége a víz/cement arány függvénye (Muhit, 2013) (Plank et al., 2009). Tehát, a Superplasticizer változó nagy pontossággal közelíthető a Cement és a Water változók többváltozós függvényeként. Ezt a jelenséget pedig a HSIC-Lasso algoritmus figyelmen kívül hagyja, mivel a magyarázóváltozók közti redundanciát csak páronként szűri.

A HGHK algoritmus a concurvity jelenséget közvetlenül szűri. Ezzel pedig meg tudja szűrni a szelekció végén kapott változóhalmazt a káros többváltozós együttmozgásoktól. Jelen esetben, mivel az mRMR esetében felhasználjuk azt az a priori tudást, hogy a globális optimumban $\left|\tilde{X}\right| = 4$, így az algoritmus végső megoldásában nem jelentkezik káros concurvity jelenség, hiába szűri a magyarázóváltozók redundanciáját ez az algoritmus is csak páronként, mint a HSIC-Lasso. Ellenben, a megoldás R^2 értéke elmarad a HGHK algoritmus modelljétől.
A HGHK algoritmus modelljének legnagyobb előnye, hogy mivel a modell concurvity jelenségtől mentes, így a benne szereplő változók ceteris paribus marginális hatásai értelmezhetők, a változókra illesztett spline függvények alapján. Egy ilyen jellegű értelmezést a szintén concurvity jelenségtől mentes mRMR modell alapján is meg lehet tenni, de érdemesebb a HGHK modellt alkalmazni, mivel annak teljesítménye magasabb a tesztmintán az mRMR teljesítményénél. Tehát, az mRMR modellnél a HGHK algoritmus a gerendák nyomószilárdságával jobban összefüggő és concurvity mentes változóhalmazt tudott azonosítani.
A modell alapján tehát elmondhatjuk, hogy a betongerendák nyomószilárdságát azok cement, víz, salak tartalma és koruk befolyásolják legjobban, mint nem-lineárisan is korrelálatlan tényezők. A 12. ábra alapján leolvasható ezen változó marginális hatása: a cement tartalom majdnem lineáris összefüggésben van a nyomószilárdsággal. Salak és víztartalomból egyértelműen megállapítható egy maximális nyomószilárdságot eredményező mennyiség: kb. 250 kg/m^3 salak esetében és kb. 140 kg/m^3 a víz esetében. A magasabb életkor logaritmikus ütemben növeli a nyomószilárdságot, kb. 55 évnél idősebb gerendák esetében nem nő jelentősen a nyomószilárdság.



12. ábra: A HGHK algoritmus végső modelljében a magyarázóváltozókra illesztett spline függvények a betongerendák adatbázison, 95%-os konfidencia-intervallummal. Forrás: saját szerkesztés.

A HGHK algoritmus futásideje látványosan különbözik a többi vizsgált algoritmusétól. Az 1-2 másodperces nagyságrendekhez képest a HGHK algoritmus várható futásideje 2 perc köré tehető. Ennek oka, hogy az algoritmus több változóhalmazhoz tartozó GAM-ot épít fel és vizsgál meg, így hátrányban van futásidő szempontjából a folytonos optimalizációt megoldó regularizációs módszerekkel vagy a lokális kereső stepwise jellegű algoritmusokkal szemben. A GAM modelleket nem is építő mRMR és HSIC-Lasso algoritmusok futásidejéhez képest pedig különösen magas a HGHK algoritmus futásideje. Ugyanakkor a GAMBoost algoritmus, több mint 8 percig tartó várható futásidejét vizsgálva a HGHK algoritmus eléggé gyorsnak tűnik. Mindezek alapján elmondhatjuk, hogy ha az elemző célja inkább egy magyarázó modell építése, ahol a cél a legfontosabb magyarázóváltozók hatásának visszafejtése az eredményváltozóra, nem pedig a minél nagyobb becslési pontosság elérése, akkor a kezdeti tapasztalatok alapján a HGHK algoritmus ígéretes alternatívája lehet az egyéb vizsgált változószelekciós algoritmusoknak.

9.2. A vizsgált algoritmusok által javasolt modellek kiértékelése több teljesítménymetrika szerint

A betongerendák adatbázison futtatott változószelekciós algoritmusok által javasolt GAM modellek becslési pontossága a teszthalmazon a 8.2. fejezetben bemutatott teljesítménymetrikák szerint a 3. táblázatban található.

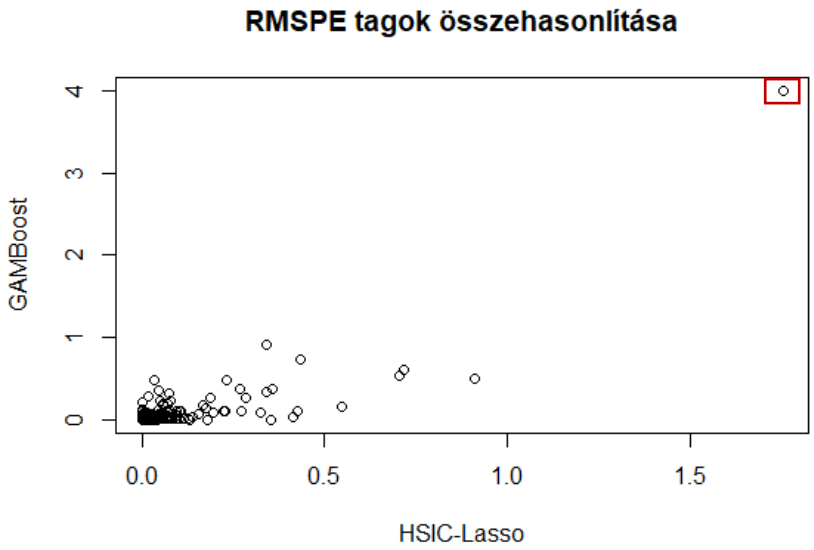
Algoritmus	R^2	Koszinusz Hasonlóság	RMSE	RMSPE	RMSLE	MAE	MAPE	Log-Cosh
Teljes modell	88,12%	0,988	5,918	26,94%	0,192	4,547	17,07%	3,910
COSSO	86,22%	0,985	6,617	30,72%	0,251	5,297	20,36%	4,643
Büntetőtagos thin plate spline (23) formulával	88,08%	0,988	5,928	26,64%	0,191	4,546	16,99%	3,911
Büntetőtagos thin plate spline $\widetilde{S_j}$ mátrixszal	87,69%	0,987	6,021	26,45%	0,192	4,619	17,03%	3,982
Nemnegatív Garotte módszer	78,20%	0,976	8,467	36,65%	0,269	6,485	24,28%	5,839
Stepwise (AIC)	88,12%	0,988	5,918	26,94%	0,192	4,547	17,07%	3,910
Backward (szignifikancia)	88,11%	0,988	5,921	26,88%	0,191	4,543	17,05%	3,907
GAMBoost	89,24%	0,989	5,632	25,50%	0,195	4,333	16,44%	3,709
Módosított backfitting	87,95%	0,987	5,963	25,51%	0,198	4,535	16,79%	3,901
mRMR	81,58%	0,981	7,357	28,27%	0,243	5,574	19,75%	4,925
HSIC-Lasso	86,38%	0,986	6,335	24,49%	0,201	4,808	16,94%	4,183
HGHK algoritmus	84,36%	0,984	6,799	26,72%	0,214	5,412	19,04%	4,771

3. táblázat: A vizsgált algoritmusok becslési pontosságának kiértékelése több teljesítménymetrika alapján a betongerendák adatbázison. Minden metrika oszlopában a legrosszabb pontosságot jelentő értéket piros, a legjobb értéket zöld színnel jelöljük. Forrás: saját szerkesztés.

A 3. táblázat alapján a 8.2. fejezetben megfogalmazott előzetes várakozásaink teljesültek. A négyzetes és abszolút hibafüggvényen alapuló teljesítménymetrikák a kiugró értékek szűrése miatt nem eredményeznek érdemben eltérő modellpreferencia sorrendet. Tehát, a legtöbb esetben a 2. táblázatban, az R-négyzet alapján kialakult sorrend marad meg a modellek között becslési pontosság alapján. Ellenben, a modellhibákat egy megfigyelésen hányadossal és nem különbséggel összehasonlító $RMSPE$ és $RMSLE$ metrikák által javasolt legjobb modell eltérő a többi metrikához képest.

Kiemelendő, hogy az $RMSPE$ által javasolt modell a HSIC-Lasso algoritmus modellje, ami csupán 5 változót használ a 8-ból. Ugyan, csak minimálisan, egy százalékponttal jobb $RMSPE$ szerint a HSIC-Lasso teljesítménye a legtöbb metrika által preferált GAMBoost-hoz képest, de az a teljes változókészletet felhasználja az eredmény eléréséhez.

Azonban fontos látni, hogy a HSIC-Lasso algoritmus modelljének kiemelkedő teljesítményét az $RMSPE$ metrikában elsősorban egy kiugró értékként viselkedő pont magyarázza (13. ábra).



13. ábra: A HSIC-Lasso és GAMBoost modellek $(\hat{y}_i/y_i - 1)^2$ hányadosainak alakulása a betongerendák adatbázis teszthalmazán. A GAMBoost modell alapján kiugróan magas hányadosok kerettel jelölve. Forrás: saját szerkesztés.

A 13. ábrán piros keretezéssel jelölt pontban a GAMBoost modell alapján számolt $\hat{y}_i$ becslések arányaiban sokkal nagyobb mértékben különböznek a valós y_i értékektől, mint a HSIC-Lasso modellből adott $\hat{y}_i$ becslések. Konkrétan, a HSIC-Lasso becslés és a valós érték hányadosa közel 2,32-nek adódik, ám a GAMBoost becslés erre az értékre már a valós érték nagyjából 3-szorosa. Ezek az eltérések pedig az $RMSPE$-be négyzetes skálán kerülnek beszámításra (35) alapján. Viszont, az $RMSLE$ metrikában a logaritmus alkalmazása a becsült és valós Y értékek hányadosára (40) szerint, megszünteti a kiugróan magas GAMBoost eltérés hatását. Mivel ez a hiba csak relatív mértékben nagy a GAMBoost esetén, különbségeket vizsgálva nem, így a többi klasszikus SSE alapú metrikában nem jelentkezik kiugró értékként.

Az $RMSLE$ metrika alapján a változók szignifikanciája szerint szelektáló Backward algoritmus és a (23) formula szerinti büntetőtagos thin plate spline által javasolt modellek nyújtják a legjobb teljesítményt. Ezek a modellek már a legtöbb metrika alapján preferált GAMBoost modellhez hasonlítanak olyan szempontból, hogy a teljes változókészletet a modellben hagyják. Ez alól a Backward algoritmus modellje képez kivételt, de itt is csak egy változó kerül elhagyásra (2. táblázat). Innen látszik, hogy a hányados alapú hibamérés egyetlen kiugró hibamértékének hatása jelentősen mérséklődik, hiszen az $RMSLE$ a GAMBoost-hoz hasonló viselkedésű modelleket preferál. Továbbá, a GAMBoost és a legjobb modell $RMSLE$ értékének hányadosa csak 1,02, az $RMSPE$ esetben tapasztalt 1,04-hez képest.

A HGHK algoritmus szempontjából elmondhatjuk, hogy a MAE és $Log - Cosh$ metrikák esetén kialakított preferencia sorrendben az algoritmus által javasolt modell az R^2 alapján

kialakított sorrendben elfoglalt pozícióját őrzi meg a nemnegatív Garotte és mRMR algoritmusok modelljei előtt. Azonban, a Koszinusz Hasonlóság, $RMSE$, $RMSLE$ és $MAPE$ metrikák esetében egy hellyel előre lép, mivel a COSSO módszernél is jobb teljesítményt nyújt. Az $RMSPE$ metrikát vizsgálva pár tized százalékponttal ugyan, de a Teljes, Stepwise és Backward modelleket is maga mögé szorítja. Bár ez a teljesítmény csak a HSIC-Lasso modell vizsgálata során azonosított három ponton nyújtott jó relatív becslési pontosság eredménye.

Összességében elmondhatjuk, hogy a HGHK által javasolt GAM modell becslési pontossága robusztusnak tekinthető a vizsgált metrikák körében, az egyetlen másik concurvity korlátot nem sértő mRMR modellhez képest minden metrikában pontosabb becsléseket szolgáltat. Tehát nem sérül az a 9.1. fejezetben tett megállapítás, hogy az mRMR modellnél a HGHK algoritmus a gerendák nyomószilárdságával jobban összefüggő és továbbra is concurvity mentes változóhalmazt tudott azonosítani.

Összefoglalva a 9.1. és 9.2. alfejezetek eredményeit a K2 kutatási kérdés kapcsán kijelenthetjük, hogy HGHK algoritmus modelljeinek tesztmintákon mért becslési pontossága nagyságrendileg nem marad el az értekezésben vizsgált egyéb algoritmusok által javasolt modellek teljesítményétől a betongerendák adatbázison. Egyedül az mRMR algoritmus képes concurvity jelenségtől teljesen mentes változóhalmaz azonosítására (2. táblázat), ám a javasolt modell becslési pontossága elmarad a HGHK algoritmus modelljétől, valamint alkalmaztuk azt az a priori ismeretet, hogy a korlátoknak megfelelő globális optimumban a magyarázóváltozók száma négy. A betongerendák adatbázis esetében az eredmények robusztusnak tekinthető a vizsgált teljesítménymetrikák körében. Az egyetlen másik concurvity korlátot nem sértő mRMR modellhez képest minden vizsgált metrikában (tehát nem csak R^2-ben) pontosabb becsléseket szolgáltat a HGHK által javasolt GAM.

9.3. HGHK algoritmus viselkedése, optimális paraméterezés azonosítása

A betongerendák adatbázis kis mérete lehetőséget ad a HGHK algoritmus paramétereinek finomhangolásához. Az algoritmus a globális optimumot a 30 futtatás átlagában az összes lehetőség 39%-nak 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 HGHK algoritmus átlagosan az összes lehetőség 75%-nak átvizsgálásával meg tudja találni a globális optimumot. Ezen a paraméterek állításával lehet még javítani.

Az algoritmusban a 4. 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%

4. táblázat: A HGHK 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 4. táblázat eredményei alapján elmondható, hogy a HGHK algoritmusban a teljesen véletlen új egyed generálásra érdemes nagyobb súlyt helyezni, de nem érdemes teljesen elhagyni a memóriából történő kontrollált egyed generálást sem (5-ről 35%-ig emelhető a $HMCR$). A vizsgálat eredményei alapján a $HMCR$ választása nem egyértelmű. Ceteris paribus az 5% induló értékkel azonos hatékonyságot nyújt a 30%-is induló érték választása is. Viszont, mivel maximális $HMCR$ paraméter ceteris paribus vizsgálata során egyértelműen a 35%-os érték bizonyult a leghatékonyabbnak, így az azonos hatékonyságú induló értékek közül a kisebb, 5%-os értéket választottuk optimálisnak.

Az algoritmus véletlen elemeit preferáló, de az irányított elemeket is némiképp megőrző paraméterezés hatékonyságát támasztja alá az a tény is, hogy a memóriából történő új egyed előállítás esetén is érdemes az algoritmus első lépéseiben magas mutációs (bw) valószínűséget alkalmazni (90%), ám az induló érték nagyon alacsonyra (10%) csökkentése az iterációk során szintén kifizetődő.

Jelen alfejezet eredményei alapján tehát a K3 kutatási kérdés kapcsán kijelenthető, hogy GAM keret esetén a HGHK algoritmusban a teljesen véletlen új egyed generálásra érdemes nagyobb

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

súlyt helyezni, de nem érdemes teljesen elhagyni a memóriából történő kontrollált egyed generálást sem. Csupán ezen kontrollált elemek súlyát kell leszorítani a rekombináció során.

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 memória méret mellett). Azaz a keresési térnek kb. $\frac{5*20}{2^8} = 0{,}4$ részét vizsgálja csak át az algoritmus, mire megtalálja a két legjobb megoldás egyikét. A tapasztalat így arra enged következtetni, hogy előnyösebb az 5. fejezetben bemutatott második stratégiát alkalmazni a kezdeti memóriák rosszabb minőségének kezelésére: egy futásidőben könnyen kezelhető memória méretet választunk, és többször lefuttatjuk az algoritmust egy alacsonyabb maximális generációszám mellett.

Az előző bekezdésben ismertetett eredmények alapján választ tudunk adni a K4 kutatási kérdésre is: a GAM keretből adódó rosszabb kezdeti memória (populáció) minőséget az optimális paraméterezésben az algoritmus több véletlen kezdeti memóriából történő futtatásával preferált kezelni az egyszer, ám nagy memóriamérettel történő futtatással szemben.

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

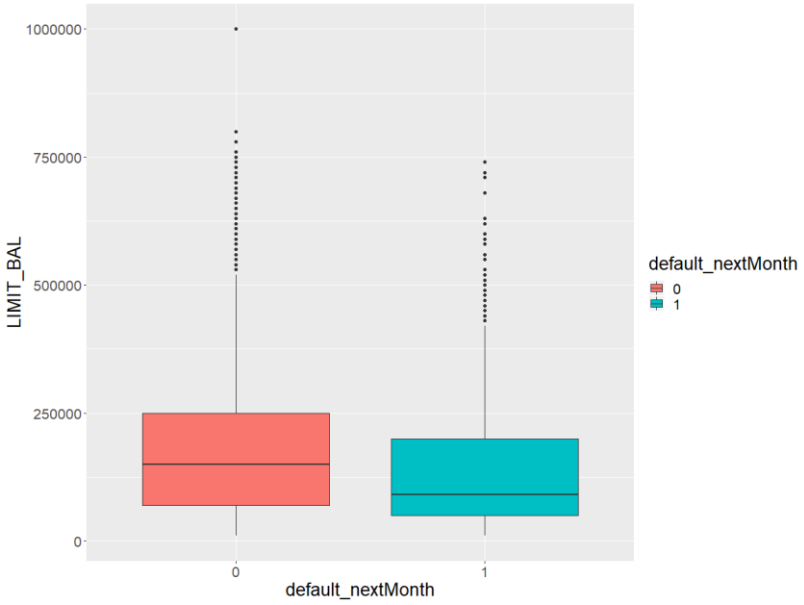
10. A HGHK algoritmus viselkedése nagy méretű feladat esetén

A banki ügyfelek adatbázis változóinak jelentése az alábbi:

- LIMIT_BAL – Az ügyfél hitelkerete a kártyáján (ezer tajvani dollárban): beleértendő az egyéni fogyasztói hitelkeret és a családtagoknak szóló kiegészítő hitelkeretösszeg is.
- 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;…)
- 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 – A célváltozó:
    - 1 = az ügyfél csődöt jelentett hitelkártya adósságára az adatgyűjtés utáni egy hónapban;
    - 0 = az ügyfél nem jelentett csődöt hitelkártya adósságára az adatgyűjtés utáni egy hónapban

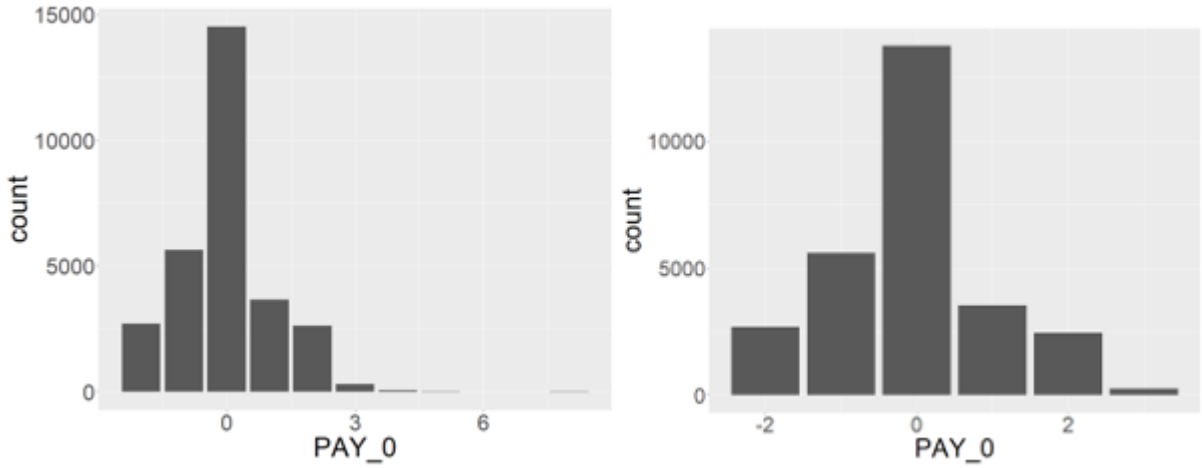
Az adatbázisban Yeh – Lien (2009) alapján megállapíthattuk, hogy magas a hibásan kódolt kategorikus változókkal rendelkező ügyfelek száma. A MARRIAGE változóban 0, míg EDUCATION változóban 0, 5, 6 érvénytelen kódok fordultak elő. Ezeket a megfigyeléseket leszűrtük az adatbázisunkból. Ezzel 399 ügyfelet veszítve, 29 601 megfigyelésünk maradt.

A LIMIT_BAL változóban egy extrém módon kiugró (Tukey-féle felső külső kerítésnél magasabb) ügyfelünk volt, 1 000 000 ezer dolláros hitelkerettel (14. ábra). Ezt az ügyfelet eltávolítva 29 600 megfigyelésünk maradt.



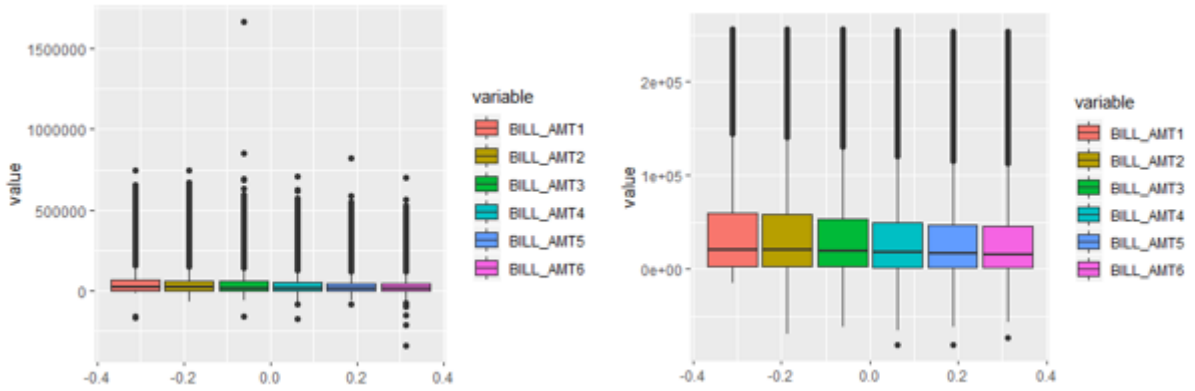
14. ábra: A banki ügyfelek adatbázisban a LIMIT_BAL változó dobozábrája. Forrás: saját szerkesztés.

A késedelmes hónapok száma változókban (PAY_X, $\forall X - re$) a 3 hónapnál többet késő ügyfelek erősen alacsony gyakorisága miatt 3 hónapnál többet késő ügyfeleket is leszűrtük, mivel a 3-nál magasabb értékek előfordulása az eloszlásban 0 közelinek tekinthető (15 ábra).



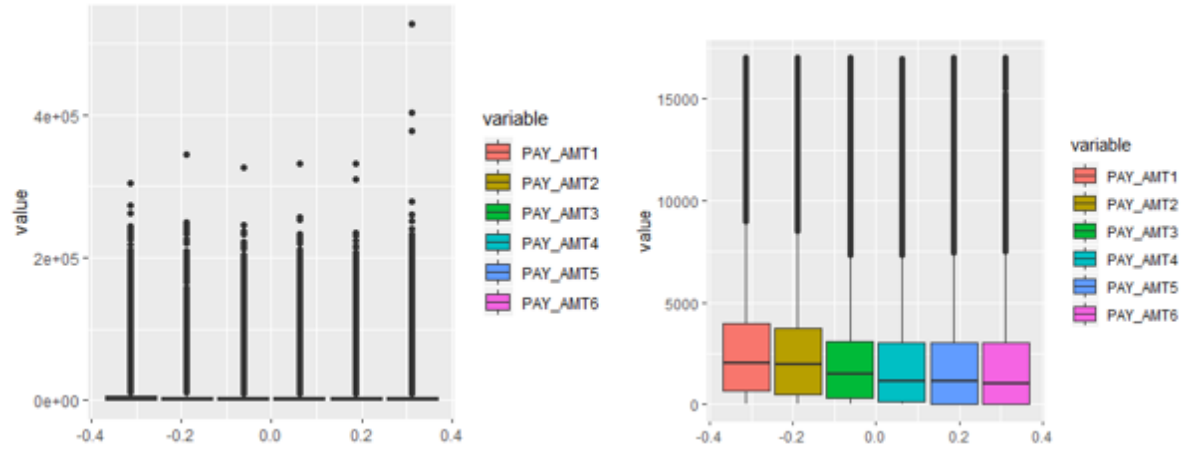
15. ábra: A banki ügyfelek adatbázisban a PAY_0 változó hisztogramja a kiugró értékek szűrése előtt (balra) és után (jobbra). Forrás: saját szerkesztés.

A BILL_AMTX változók esetében a Tukey-féle külső kerítések BILL_AMT4-ben felfelé extrém kiugró értékeket, míg a BILL_AMT6 változó esetében lefelé kiugró értékeket (tehát extrém magas túlfizetése van az ügyfélnek a törlesztőrészleteire) jeleztek, amiket leszűrtünk az adatbázisból (16. ábra).



16. ábra: A banki ügyfelek adatbázisban a BILL_AMT*X* változók doboz ábrája a kiugró értékek szűrése előtt (balra) és után (jobbra). Forrás: saját szerkesztés.

A PAY_AMTX változók esetében csak felső irányban fordultak elő extrém módon kiugró értékek. Elsősorban a PAY_AMT6 változóban. Ezen szélsőértékeket is eltávolítottuk az adatbázisból. Ezzel minden egyedi esettől megtisztítottuk az adatokat, és az extrém módon elnyúló eloszlásokat korrigáltuk, extrémérték eloszlások illesztése nem lesz szükséges (17. ábra).



17. ábra: A banki ügyfelek adatbázisban a PAY_AMT*X* változók doboz ábrája a kiugró értékek szűrése előtt (balra) és után (jobbra). Forrás: saját szerkesztés.

Bár a végső adatbázisban a dobozábrák alapján maradtak még a Tukey-féle belső kerítések szerint kiugró értékek, ezek a külső kerítéseket vizsgálva nem extrém módon kiugró, a regressziós modellek becsléseit befolyásoló megfigyelések. Továbbá, értékeik a változóik értékkészletének természetes folytatásai, nincs az eloszlásban törés a doboz ábrák alapján. A végső adatbázisban így 22 165 megfigyelés maradt. Ezt osztottuk fel 70%-30% arányban tanító és tesztmintára.

10.1. A GAM keretben adott változószelekciós feladat párhuzamosítási lehetőségeinek elemzése

Az adatbázisban a célváltozó Bernoulli eloszlású (Yeh – Lien, 2009), így a GAM link függvénye a logit.

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 az 5.2.4. alfejezetben ismertetett QR dekompozíció 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. Hasonlóan, ahogy a COSSO módszer esetében Lin – Zhang (2006) javasolja. 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 normalitás teszt 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 $26{,}46 \cdot \frac{3{,}374}{21{,}373} = 4{,}18$ 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.

Jelen alfejezet eredményei alapján a K5 kutatási kérdés első részére tudunk választ adni. A QR dekompozíció segítségével megvalósított párhuzamosítás esetén az összes lehetséges megoldás előállításához várhatóan 26,46 év szükséges. Ez a várható idő kb. 0,16-szorosára (azaz kb. 4,18 évre) csökkenthető, ha több részhalmazhoz tartozó modellt számítunk ki egyszerre párhuzamosan. Tehát, a HGHK algoritmusban jobban kifizetődik több egyedhez tartozó modellek egyszerre történő kiszámítása, mint egy modell kiszámításának párhuzamosítása.

Az adattisztítási lépések és a vizsgált algoritmusok implementálása és futtatása a 9. fejezethez hasonlóan az R 3.5.3 verziójában történt. Minden egyéb technikai paraméter és a tanító- és tesztminták megadási módja, valamint az R szkriptek tárhelye (https://github.com/KoLa992/Hybrid-algorithm-for-GAMs) szintén megegyezik a 9. fejezetben megadottakkal. Annyi kivétellel, hogy az mRMR algoritmus esetében már nincs a priori ismeretünk $\left|\tilde{X}\right|$-re, így a *mRMRe* csomag alapértelmezett $\left|\tilde{X}\right| = 6$ beállításán nem változtatunk.

10.2. Benchmark modellek futási eredményei

A vizsgált adatbázisra Yang – Zhang (2018) alkalmaz több korszerű gépi tanuló algoritmust is: GLM (Logit link függvénnyel), neurális hálózat, támaszvektor-gép, Xgboost és LightGBM. A tanulmányban a cél csupán az osztályozási pontosság maximalizálása, így a szerzők változószelekciót nem alkalmaznak.

Az idézett tanulmányban a legmagasabb AUC érték a tesztmintán a LightGBM algoritmus éri el, ami 0,7904. A legrosszabb eredmény a GLM-é 0,7228-cal. Ezek az eredmények szolgálnak benchmarkként a változószelekciós algoritmusok által javasolt GAM modellek teljesítményének mérésére. Mindezek mellett a benchmark modellek listáját kibővítjük két modellel: döntési fával (CART algoritmus) és véletlen erdővel. A véletlen erdő algoritmust egy egyszerű változószelekciós algoritmussal, az RFE-vel (Recursive Feature Elimination) kombináljuk.

Kuhn et al. (2019) alapján az RFE algoritmus egy legjobb részhalmaz elvű változószelekciós algoritmus. Az algoritmusnak meg kell adni $q \leq m$ darab különböző p_i értékeket $\left|\tilde{X}\right|$-re. Továbbá egy err változószelekciós hibafüggvény megadása is szükséges a (2)-höz hasonló

alakban, ám a (2)-ben szereplő $\ell(\beta)$ tag cserélhető az RFE eljárással kombinált gépi tanuló algoritmus függvényében más értékre (pl. Gini-index, kereszt-entrópia, félreosztályozási ráta, stb.). Az RFE a hibafüggvény 10-szeres keresztvalidációval számított értékét használja.

Az RFE algoritmus minden $i \in \{1,2,\dots,q\}$ esetben megvizsgálja, hogy milyen változókombináció eredményezi a legkisebb választott hibafüggvény értéket az összes lehetséges $\tilde{X}_{p_i} = \{X_1, X_2, \dots, X_{p_i}\} \subseteq X$ részhalmaz megvizsgálásával. Ezek után különböző p_i értékekhez tartozó optimális 10-szeres keresztvalidációval számított hibafüggvény értékek (err_i) közül az algoritmus egyszerűen kiválasztja a legjobb értéket, és az ehhez az értékhez tartozó $\tilde{X}_{p_i}$ részhalmaz adja a végső modell magyarázóváltozóinak halmazát. Mindez formálisan a $p_L = \underset{i}{\operatorname{argmin}}\, err_i$ feladat megoldását jelenti.

A banki ügyfelek adatbázison csak a $26,25,20,15,10,5,3,1$ db magyarázóváltozót tartalmazó részhalmazokat vizsgáljuk meg, hogy a feladat ne legyen a 2^m lehetőség megvizsgálásával ekvivalens. Az algoritmust véletlen erdő tanulási algoritmussal kombináljuk, és az R *caret* csomagjának segítségével futtatjuk (Kuhn et al., 2019). A választott hibafüggvény Yang – Zhang (2018) alapján a 7. fejezetben megismert $\gamma_R(D)$ félreosztályozási ráta. A véletlen erdő algoritmus sztochasztikus elemei miatt az algoritmus futtatását 30-szor replikáljuk, és a legjobb megoldást vizsgáljuk. Ezen kívül vizsgáljuk az algoritmus futásidejének átlagát és szórását a 30 esetben.

A döntési fa algoritmust a banki ügyfelek adatbázisra R nyelven az *rpart* csomag segítségével futtatjuk (Therneau – Atkinson, 2018). A választott vágási függvény a CART algoritmushoz a Gini-index. A futásidők vizsgálata miatt a döntési fa algoritmus futtatását is 30-szor megismételjük a banki ügyfelek adatbázison. Mivel az algoritmus paraméterezésén nem változtatunk, így a modell nem fog változni a 30 futtatás során.

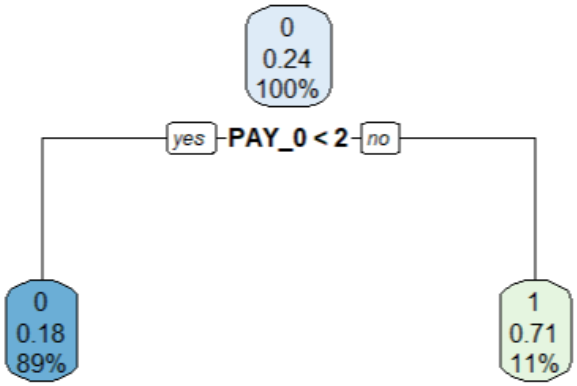
A benchmark modellek eredményeit a banki ügyfelek adatbázison az 5. táblázat tartalmazza.

<table>
<tr><th>Algoritmus</th><th>Kiválasztott változók</th><th>Concurvity korlátot sértő változók</th><th>AUC (Tesztminta)</th><th>Átlagos futásidő (perc)</th><th>Futásidők szórása (perc)</th></tr>
<tr><td>Döntési fa</td><td>PAY_0</td><td>-</td><td>0,6413</td><td>0,0127</td><td>0,0004</td></tr>
<tr><td>Véletlen erdő (RFE szelekcióval)</td><td>mindegyik</td><td>LIMIT_BAL, PAY_0+2+3+4+5+6, BILL_AMT1+2+3+4+5+6, PAY_AMT1+2+3+4+5+6</td><td>0,7562</td><td>132,543</td><td>0,758</td></tr>
<tr><td>GLM</td><td colspan="2" rowspan="5">Yang – Zhang (2018)-ban nincs változószelekció, így mindegyik modell az összes magyarázóváltozót használja.</td><td>0,7228</td><td colspan="2" rowspan="5">Yang – Zhang (2018)-ban nincs futásidő vizsgálat.</td></tr>
<tr><td>Neurális hálózat</td><td>0,7735</td></tr>
<tr><td>Támaszvektor-gép</td><td>0,7230</td></tr>
<tr><td>Xgboost</td><td>0,7792</td></tr>
<tr><td>LightGBM</td><td>0,7904</td></tr>
</table>

5. táblázat: A benchmarkként alkalmazott algoritmusok futási eredményei a banki ügyfelek adatbázison. Forrás: saját szerkesztés.

A 3. táblázat alapján elmondható, hogy a véletlen erdő algoritmusban nem kifizetődő változószelekciót végezni, és ez az algoritmus adja a negyedik legjobb teljesítményt a tesztadatokon. Yang – Zhang (2018) és a véletlen erdő eredményei alapján elmondható, hogy ha a feladat szimplán az eredményváltozó becslési pontosságának növelése, akkor változószelekció nélküli, bonyolult neurális hálózatok és ensemble modellek (Xgboost, LightGBM) alkalmazása a kifizetődő.

A döntési fa döntési szabálya nagyon egyszerű lett, annak ellenére is, hogy a Gini-index jellemzően terebélyes döntési fákat épít (Steinberg – Colla, 2009). Esetünkben a döntési szabály annyi lett, hogy amennyiben PAY_0 változó értéke 2-nél kisebb (azaz a lekérdezés hónapjában az ügyfélnek legfeljebb 1 hónapnyi hátraléka volt), akkor csődösnek kell osztályoznunk az ügyfelet, egyébként nem (18. ábra).



18. ábra: A CART algoritmus alapján felépített döntési fa a banki ügyfelek adatbázison. Forrás: saját szerkesztés.

A döntési fa tesztmintán mért AUC értéke viszont nagyságrendekkel alacsonyabb még a minden változót használó GLM-nél is. Tehát, hiába az egyszerű és könnyen értelmezhető döntési szabály a csődveszélyes ügyfelek azonosítására, a becslés túlságosan pontatlan ebből a modellből. Tehát, a változószelekcióval kombinált GAM-ok alkalmazásának a célja, hogy találjunk egy olyan modellt, aminek a tesztmintán mért AUC értéke eléri legalább a Yang – Zhang (2018)-féle GLM szintjét, és a magyarázóváltozók hatásai visszafejthetők. Továbbá, célul tűzzük ki, hogy a változószelekciós algoritmus futásideje ne haladja meg nagyságrendileg a véletlen erdővel kombinált RFE algoritmus kb. 133 perces várható futásidejét.

10.3. A COSSO és GAMBoost algoritmusok implementációjának néhány technikai kérdése

Mielőtt részletesen áttekintjük a vizsgált algoritmusok eredményeit a banki ügyfelek adatbázison, szót kell ejtenünk bizonyos algoritmusok implementációjában felmerülő kihívásokról, korlátokról.

A COSSO és GAMBoost algoritmusokat nem lehet a teszteléshez használt konfigurációval a teljes adatbázison lefuttatni, mivel memóriaigényük túl magas.

A COSSO-nak a nem-lineáris függvények megfigyelésenként (pontonként) történő ábrázolása miatt akár 46 GB méretű mátrixokat kellene a memóriában tárolnia, amik leírják minden magyarázóváltozó RMHT-beli nem-lineáris reprezentációját a tanító halmazon.

A GAMBoost esetében pedig a magyarázóváltozók függvényformájának magas λ melletti harmadfokú bázis spline becsléseit el kell tárolni a memóriában minden lépésben, hiszen ezeket minden l lépésben ki kell értékelni, hogy kiválasztható legyen melyik $f_{j,(l)}$ transzformáció legyen hozzáadva $\hat{Y}_{(l-1)}$-hez. Ráadásul ezeket az információkat a módosított backfitting eljáráshoz képest több kilépési kritérium, több L érték mellett is el kell tárolni, hogy végül 10-szeres keresztvalidációs eljárással kiválasszuk a legjobban illeszkedő modellt. A módosított backfitting esetében elég a $\hat{Y}_{(l)} = \hat{Y}_{(l-1)}$ kritérium teljesülésekor rögtön megállni, így nem áll fenn reális veszélye annak, hogy olyan sok nem-lineáris $f_{j,(l)}$ transzformációt kell tárolni, hogy ne legyen rá elég RAM egy átlagos személyi számítógépben (a teszteléshez használt konfigurációban 8 GB RAM volt) egy 22165×27-es adatmátrix esetén. A több L megvizsgálása mellett kapott legjobb eredmény javíthat a GAMBoost algoritmus pontosságán, amit a betongerendákkal foglalkozó adatbázis példáján láttuk is. Azonban, a több kilépési feltétel melletti modellek eltárolása és 10-szeres keresztvalidációja már egy ilyen közepes méretű adatbázison is RAM hiányhoz vezethet.

Az R több csomagja is kínál lehetőséget a számítógép RAM kapacitásánál nagyobb méretű adatmátrixok kezelésére. A módszereket Walkowiak (2016) és a CRAN weboldalán 2019. 10. 31-én elérhető R csomagok dokumentációja alapján foglaljuk össze.

Az *ff* és *ffbase* csomagok segítségével nagy mátrixainkat almátrixokra bontva a merevlemezen tárolhatjuk azokat az almátrixokat, amelyeken műveletet éppen nem végzünk. Ezzel a módszerrel a számítások elvégezhetővé válnak a RAM kapacitásnál nagyobb méretű mátrixokra is, ám a várható futásidő számottevően megnő a folyamatos merevlemez írás-olvasási műveletek miatt. Esetleg SSD alkalmazása javíthat az írás-olvasás időkön, ám a módszernek egy komolyabb technikai korlátja is van számunkra.

R környezetben egy speciális típusú objektumban, *ff dataframe*-ben (*ffdf* röviden) tudjuk csak tárolni azt az adatmátrixot, amit nem szeretnénk teljes egészében RAM-ban tartani, hanem almátrixait a merevlemezen kívánjuk tárolni (Adler et al., 2018). Az *ffdf* típusú adatmátrixok kezelésére pedig koránt sincsen minden R függvény felkészítve. A CRAN könyvtárak átvizsgálása után azt tapasztalhatjuk, hogy alapvetően a *ffdf* objektumok lineáris modellek kezelésére vannak felkészítve. Az általánosított lineáris modellek (*biglm* vagy *biglmm* függvény) és a Lasso legnépszerűbb megoldó algoritmusa a LARS (*biglars* függvény) került implementálásra *ffdf* objektumokra is (Lumley, 2013) (Lumley, 2018) (Seligman et al., 2011). Nem-lineáris modellek közül egyelőre csak a döntési fa és véletlen erdő algoritmusoknak létezik *ffdf* implementációja (*bigrf* függvény), ám már ez a függvény nem elérhető az R legújabb verzióiban[5].

A COSSO és a Boosting algoritmusok nem képesek jelenleg *ffdf* típusú objektumok kezelésére. Az implementáció megvalósítása túlmutatna jelen értekezés keretein. Érdekes feladat lehet a későbbiekben a fenti két algoritmus *ffdf* objektumokat is kezelő implementációk elkészítése.

Egy alternatív megoldás R-ben a RAM-nál nagyobb méretű mátrixok kezelésére a *bigmemory* csomag. A csomag segítségével adatmátrixainkat *big.matrix* objektumként tudjuk R környezetben eltárolni. A *big.matrix* objektumok képesek osztott memória használatra. Tehát az adatmátrixaink memory-mapped fájlként működnek, így hozzáférnek más folyamatokhoz allokált, de fizikailag szabad RAM erőforrásokhoz is (Kane et al., 2013).

Ez a megközelítés sajnos a mi problémánkra eleve nem képes megoldást szolgáltatni, mivel a COSSO algoritmus futtatásához szükséges 46 GB méretű, nem-lineáris transzformációkat pontonként ábrázoló mátrixokat akkor sem tudjuk a 8 GB RAM-ban kezelni, ha az R-nek hozzáférést engedünk a teljes memóriatartományhoz. A csomag előnye, ha több számítógépet klaszterbe tudnánk kötni, akkor a klaszter teljes RAM tartományához osztott hozzáférést tudunk adni a *big.matrix* objektumoknak. Azonban, az értekezés megírásakor egy közös memória klaszter létrehozásához nem állt rendelkezésünkre a megfelelő hardveres erőforrás.

Továbbá, a *big.matrix* objektumokat szintén nem képes kezelni jelenleg a COSSO és GAMBoost implementáció R nyelven. Jelenleg a CRAN dokumentációk alapján *big.matrix* objektumokat általánosított lineáris modellekben (*bigglm.big.matrix* függvény, ami már a korábban említett *biglm* függvényt is használja), valamint k-közép klaszterezés során (*bigkmeans* függvény) tudunk alkalmazni. Ezen kívül alapvető lineáris algebrai transzformációk (pl. főkomponensek képzése) végezhetők el *big.matrix* objektumokon

[5] https://cran.r-project.org/web/packages/bigrf/index.html
Letöltve: 2019.10.31.

a *bigalgebra* csomag függvényei segítségével. A COSSO és a GAMBoost algoritmusok nem képesek jelenleg *big.matrix* típusú objektumok kezelésére sem.

Ezen technikai megkötések miatt egy másik megközelítésben kell alkalmaznunk a COSSO és a GAMBoost algoritmusokat. A fejezet elején bemutatott, véletlenszerűen kiválasztott almintát alkalmazó megközelítés nem csak a HGHK algoritmus gyorsítására, hanem a két algoritmus RAM korlátjainak feloldására is alkalmazható. Tehát, mindkét algoritmust a tanítóminta egy véletlenszerűen kiválasztott $n = 5000$ elemű almintáján futtatjuk, és az eredményül kapott modellekkel az elkülönített tesztmintán végzünk előrejelzést. A rekordok számának csökkentésével a teszteléshez használt konfiguráció RAM kapacitása már elégséges az algoritmusok futtatásához. A fejezet elején elvégzett vizsgálatok alapján pedig tudjuk, hogy az algoritmus végső GAM becslési pontosságát mérő $\bar{R}^2$ 95%-os megbízhatósággal legfeljebb 0,022-vel fog eltérni a teljes tanító adatbázison számított modell $\bar{R}^2$ értékétől.

10.4. A HGHK algoritmus paraméterezésének kérdései

A fejezet elején bemutatott 100 szimulált egyed (változókombináció) esetén vizsgált GAM modell számításból a legfontosabb tanulság, hogy jobban kifizetődik több egyedhez tartozó modellek egyszerre történő kiszámítása, mint egy modell kiszámításának párhuzamosítása. Ezt a tapasztalatot tudjuk hasznosítani a HGHK algoritmus implementációjakor jelen feladatra. A 7. fejezetben bemutattuk, hogy a HGHK 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 COSSO és a GAMBoost algoritmusok esetében bemutatott módon (10.3. alfejezet).

A betongerendák adatbázison szerzett tapasztalatokat is tudjuk hasznosítani a HGHK 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. Ez az 5. fejezetben bemutatott második stratégia a kezdeti memóriák rosszabb minőségének kezelésére.

A $HMCR$ és mutációs valószínűségek beállításaihoz a betongerendák adatbázison tapasztalt legjobb beállításokat alkalmazzuk. A $HMCR$ valószínűség 5%-ról nő a 6 iteráció során 35%-ra, míg a mutációs (bw) valószínűség 90%-ról csökken 10%-ra.

A megadott paraméterek mellett lefuttatjuk a HGHK algoritmust 20-szor (az algoritmus sztochasztikus elemei miatt) és vizsgáltuk meg az eredményül kapott modelleket és futásidőket.

A célfüggvény a korrigált McFadden-féle pszeudo R^2 mutató ($\bar{R}^2$). A concurvity korlátot a pesszimista mérték alapján vizsgáljuk és a $\Psi_{k_j} = \mathbf{0}$ nullhipotézis vizsgálatához választott szignifikancia-szint 5%. A k_j értékek szabályozásához használt Augustin et al. (2012)-féle próbák esetén a választott szignifikancia-szint 1%.

10.5. A vizsgált változószelekciós algoritmusok futási eredményeinek elemzése

A banki ügyfelek adatbázison futtatott benchmark és GAM változószelekciós algoritmusok részletes eredményei a 6. táblázatban tekinthetők meg. A táblázatban a Yang – Zhang (2018) által futtatott algoritmusok közül csak a legrosszabb (GLM) és a legjobb (LightGBM) eredményt nyújtó algoritmusok kerültek megjelenítésre.

Algoritmus	**Kiválasztott változók**	**Concurvity korlátot sértő változók**	AUC **(Tesztminta)**	**Átlagos futásidő (perc)**	**Futásidők szórása (perc)**
Teljes modell	mindegyik	mindegyik kivéve AGE, SEX, EDUCATION, MARRIAGE	0,7689	0,305	0,041
COSSO	LIMIT_BAL, EDUCATION, PAY_0+2+3+4+6	PAY_0+2+3+4+6	0,7011	64,272	20,979
Büntetőtagos thin plate spline (23) formulával	mindegyik kivéve BILL_AMT5+6, PAY_AMT5	mindegyik kivéve AGE, SEX, EDUCATION, MARRIAGE	0,7681	6,624	0,105
Büntetőtagos thin plate spline $\widetilde{\boldsymbol{S}}_{\boldsymbol{J}}$ mátrixszal	mindegyik kivéve BILL_AMT6, PAY_AMT5	mindegyik kivéve AGE, SEX, EDUCATION, MARRIAGE	0,7667	5,183	0,065
Nemnegatív Garotte módszer	mindegyik kivéve PAY_6, BILL_AMT5	mindegyik kivéve AGE, SEX, EDUCATION, MARRIAGE	0,7671	0,670	0,003
Stepwise (AIC)	mindegyik	mindegyik kivéve AGE, SEX, EDUCATION, MARRIAGE	0,7689	0,309	0,045
Backward (szignifikancia)	LIMIT_BAL, SEX, MARRIAGE, PAY_0+2+3+4+6, BILL_AMT1+3, PAY_AMT1+2+4	mindegyik kivéve LIMIT_BAL, SEX, MARRIAGE	0,7683	2,584	0,548
GAMBoost	LIMIT_BAL, SEX, MARRIAGE, AGE, PAY_0+2+3+4+6. PAY_AMT1+4+5+6	mindegyik kivéve LIMIT_BAL, AGE, SEX, MARRIAGE, PAY_AMT6	0,7609	692,237	9,497
Módosított backfitting	LIMIT_BAL, AGE, SEX, EDUCATION, MARRIAGE, PAY_0+2+3+4+5+6, BILL_AMT1, PAY_AMT1+2+4+6	mindegyik kivéve LIMIT_BAL, AGE, SEX, EDUCATION, MARRIAGE	0,7649	0,443	0,214
mRMR	AGE, PAY_0+2+5, PAY_AMT_1+6	PAY_2	0,7441	1,629	0,035
HSIC-Lasso	AGE, PAY_0+2+3+4	PAY_0+2+3	0,7409	0,446	0,048
HGHK algoritmus	*LIMIT_BAL, PAY_0, PAY_AMT3*	-	*0,7488*	*147,257*	*15,079*
Döntési fa	PAY_0	-	0,6413	0,0127	0,0004
Véletlen erdő (RFE szelekcióval)	mindegyik	LIMIT_BAL, PAY_0+2+3+4+5+6, BILL_AMT1+2+3+4+5+6, PAY_AMT1+2+3+4+5+6	0,7562	132,543	0,758
GLM	Yang – Zhang (2018)-ban nincs változószelekció, így mindegyik modell az összes magyarázóváltozót használja.		0,7228	Yang – Zhang (2018)-ban nincs futásidő vizsgálat.	
LightGBM			0,7904		

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

A 6. táblázat alapján elmondható, hogy a szelektált GAM modellek megfelelnek a feléjük támasztott elvárásoknak. A bizonyos mértékű concurvity korlátok figyelembevétele mellett szelektált GAM-ok (mRMR, HSIC-Lasso és HGHK) AUC-ban meghaladják a Yang – Zhang (2018)-féle GLM AUC értékét. Továbbá, ezek a modellek nem maradnak el jelentősen a véletlen erdő és a Yang – Zhang (2018)-féle LightGBM teljesítményétől sem. Ezt az eredményt

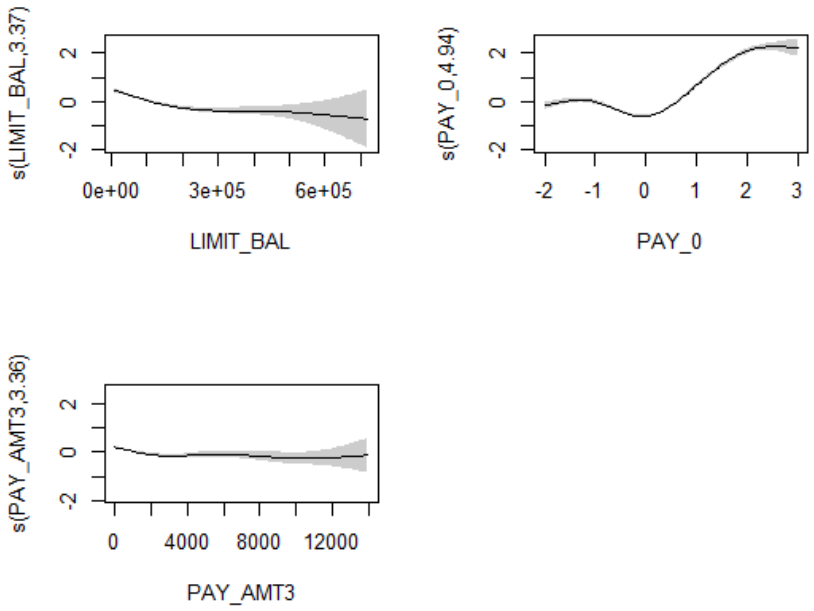
lényegesen kevesebb magyarázóváltozó mellett érik, mint a többi GAM keretben működő változószelekciós algoritmus, amik a COSSO kivételével alig hagynak el magyarázóváltozókat a modellből. Ilyen szempontból az eredmények hasonlóak a betongerendák adatbázison tapasztaltakhoz.

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. Ez okozza, hogy a közel teljes modelleket preferáló algoritmusok által javasolt modellekben a demográfiai változók (SEX, EDUCATION, MARRIAGE, AGE) kivételével az összes, az ügyfelek múltbeli fizetési fegyelmét leíró változók (PAY, BILL_AMT, PAY_AMT változók) káros concurvity jelenség által érintettek. Viszont, ha megfigyeljük a GAMBoost algoritmus modelljét, amiben a demográfiai változók mellett a fizetési fegyelmet leíró változók egy szűkebb köre került csak kiválasztásra, akkor láthatjuk, hogy a PAY_AMT6 változó már nem okoz káros concurvity jelenséget. Tehát, ez alapján azt sejthetjük, hogy a lekérdezés hónapjában vett fizetési státusz (PAY_0) mellé még érdemes lehet a lekérdezés előtt 5-6 hónappal vett egyéb fizetési fegyelmet leíró változók (BILL_AMT, PAY_AMT) szerepeltetése a modellekben. Mivel ezen változók hordoznak az ügyfél korábbi fizetési fegyelméről olyan új információt, ami nem jelenik meg a lekérdezés hónapjához közel álló fizetési státusz változókban. Ugyanakkor, a COSSO algoritmus modelljében a concurvity korlátot megsértő változókat vizsgálva azt sejthetjük, hogy a PAY változó lekérdezés előtt fél évvel vett értékét (PAY_6) nem biztos, hogy érdemes szerepeltetni egy takarékos modellben, mivel a lekérdezés időpontjához közel álló PAY változókra (PAY_0 és PAY_2) már „öröklődik” a hat hónappal korábbi fizetési státuszban lévő információ is.

Ezeket a redundanciákat a változók között még a mRMR és HSIC-Lasso algoritmusok sem detektálják teljes mértékben, hiszen concurvity jelenséget csak páronként vizsgálják. Például az mRMR algoritmus modelljében a PAY_2 változó jól közelíthető a PAY_0 és PAY_5 változók kétváltozós függvényeként. Ellenben a HGHK 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.

A HGHK algoritmus végső modelljének alaposabb vizsgálata tanulságos lehet. Különösen, ha összevetjük az eredményeket a kimondottan egyszerű elven működő döntési fa modellel. A HGHK algoritmus által javasolt GAM elemzésével szemléletesen bemutatható, hogy mik

azok a kvázi függetlennek tekinthető tényezők, amik befolyásolják a banki ügyfelek csődkockázatát 1 hónapos időtávra nézve. Az alaposabb vizsgálat ezen tényezők hatásának jellegére is rávilágítanak. A kiválasztott változók hatásai a hitelkártyacsőd valószínűségének logit transzformáltjára a 19. ábrán találhatók.



19. ábra: A HGHK algoritmus végső modelljében a magyarázóváltozókra illesztett spline függvények a banki ügyfelek adatbázison, 95%-os konfidencia-intervallummal. Forrás: saját szerkesztés.

A 19. ábrán a PAY_0 változóra illesztett spline függvényt vizsgálva látható, hogy a döntési fa modell alacsony AUC értékének az oka. A döntési fa alapján amennyiben PAY_0 változó értéke 2-nél kisebb (azaz a lekérdezés hónapjában az ügyfélnek legfeljebb 1 hónapnyi hátraléka volt), akkor csődösnek kell osztályoznunk az ügyfelet, egyébként nem. Viszont, ez a döntési szabály nem mutatja meg, hogy a PAY_0 mínuszos értékei, azaz a túlfizetők is csődveszélyesebbek. Bár a HGHK algoritmus modellje sem feltétlenül osztályozza a túlfizetőket csődös ügyfélnek, ám a PAY_0-ra illesztett spline függvény felhívja a figyelmet arra, hogy a túlfizető ügyfelek valamivel kockázatosabbak, mint a pontosan fizetők. Ez abból adódhat, hogy ha túlfizetésben van az ügyfél, akkor a bank, és néha a pénzügyi felügyelet (esetleges csalás gyanúja miatt) is arra ösztönzik az ügyfelet, hogy a túlfizetést szüntesse meg. A megszüntetés preferált módja a bank és a pénzügyi felügyelet részéről is a levásárlás. Ugyanakkor, gyakran a túlfizetés hamis eszközzel történik, és a csaló abban bízik, hogy a banktól kompenzációt kap a túlfizetésre mielőtt kiderül, hogy a csekk érvénytelen (az USA-ban 2013-ban is előfordult egy ilyen eset 1 millió dolláros összegben). Viszont, az a helyzet is könnyen előállhat, hogy az ügyfél nincs

tisztában a túlfizetés mértékével, így a levásárlás során olyan adósságot halmoz hirtelen fel, amit nem tud kifizetni[6].

A futásidők vizsgálata során egyértelmű hátrányban vannak a lehetséges magyarázóváltozók halmazának több részhalmazát is megvizsgáló algoritmusok (HGHK és RFE véletlen erdővel) a lineáris kereső mRMR-el és a regularizációs becslési feladatot megoldó vagy Stepwise jellegű algoritmusokkal szemben, nem is beszélve az egyszerű döntési fáról. Ugyanakkor, a betongerendák adatbázis esetében tapasztaltakhoz hasonlóan jelen adatbázison is a GAMBoost algoritmus produkálja a legjelentősebb futásidőket a több L maximális lépésszám vizsgálata és az egyes l iterációk nem párhuzamosított magyarázóváltozó-kezelése miatt. Az algoritmus várható futásideje jelentősen meghaladja a 11 órát még az 5000 elemű alminta alkalmazása ellenére is. Hasonló jelenséget tapasztalhatunk a COSSO algoritmus esetében is, aminek a 20 futtatás alapján 1 óra feletti a várható futásideje, bár azok szórása is jelentős (20 perc).

A futtatások tapasztalatait összefoglalva kijelenthetjük, hogy amennyiben az elemző célja egy takarékos, magyarázó modell építése, és a futásidőre nincsenek kimondottan szűk korlátok, akkor érdemes lehet a HGHK algoritmust alkalmazni például a véletlen erdővel kombinált RFE algoritmussal szemben. Hiszen, a HGHK algoritmus hasonló futásidő és tesztadat teljesítmény mellett egy olyan megoldást tud biztosítani, melyben a magyarázóváltozók hatásai jobban visszafejthetők és a concurvity jelenség hiánya miatt egymástól jól elkülöníthetők egymástól. Természetesen, az eredmények még további vizsgálatokat igényelnek több, a jelen tanulmányban vizsgáltaktól eltérő viselkedésű adatbázisokon is. Ezen túl a HGHK algoritmus eredményeinek érzékenységvizsgálata is fontos eredményekre vezethet az alkalmazott próbákhoz választott szignifikancia-szint és a concurvity mértékre szabott korlát szigorúságának függvényében.

10.6. A vizsgált algoritmusok által javasolt modellek kiértékelése több teljesítménymetrika szerint

Az értekezésben vizsgált algoritmusok becslési pontosságát a 8.3. fejezetben bemutatott teljesítménymetrikák szerint a 7. táblázat tartalmazza. A Yang – Zhang (2018)-ban szereplő GLM és LightGBM modellek teljesítményének részletesebb értékelése a 7. táblázatban nem szerepel, mivel a tanulmány AUC alapján végzett kiértékelést, és nem vizsgálta teljesítménymetrikák széles körét.

[6] https://www.creditcards.com/credit-card-news/credit-balance-overpay-refund-1282.php
Letöltve: 2019. 11. 04.

Algoritmus	Optimális Vágási Érték	AUC	$bACC$	Precision (Csőd)	Recall (Csőd)	FPR (Csőd)	F_1	Kereszt-entrópia
Teljes modell	19,59%	0,7689	69,77%	42,86%	69,67%	30,13%	0,5307	0,4603
COSSO	35,77%	0,7011	65,39%	40,23%	59,42%	28,64%	0,4798	0,7682
Büntetőtagos thin plate spline (23) formulával	20,10%	0,7681	69,96%	44,05%	67,89%	27,98%	0,5343	0,4607
Büntetőtagos thin plate spline $\widetilde{S_j}$ mátixszal	19,94%	0,7667	69,87%	43,68%	68,32%	28,58%	0,5329	0,4610
Nemnegatív Garotte módszer	22,00%	0,7671	70,10%	46,02%	64,89%	24,70%	0,5385	0,4616
Stepwise (AIC)	19,59%	0,7689	69,77%	42,86%	69,67%	30,13%	0,5307	0,4603
Backward (szignifikancia)	21,28%	0,7683	70,28%	45,67%	66,05%	25,49%	0,5400	0,4610
GAMBoost	18,59%	0,7609	69,40%	43,76%	66,54%	27,74%	0,5280	0,4650
Módosított backfitting	19,49%	0,7649	69,65%	43,18%	68,57%	29,28%	0,5299	0,4622
mRMR	19,57%	0,7441	68,68%	43,76%	64,09%	26,73%	0,5200	0,4705
HSIC-Lasso	18,90%	0,7409	68,62%	43,29%	64,76%	27,52%	0,5189	0,4708
HGHK algoritmus	19,10%	0,7488	67,81%	42,01%	64,52%	28,90%	0,5088	0,4707
Döntési fa	44,59%	0,6413	64,13%	71,74%	32,41%	4,14%	0,4465	0,9312
Véletlen erdő	23,30%	0,7562	68,99%	43,42%	65,81%	27,82%	0,5232	1,0437

7. táblázat: A vizsgált algoritmusok becslési pontosságának kiértékelése több teljesítménymetrika alapján a banki ügyfelek adatbázison. Minden metrika oszlopában a legrosszabb pontosságot jelentő értéket piros, a legjobb értéket zöld színnel jelöljük. Forrás: saját szerkesztés.

A 7. táblázat alapján nem tapasztalható olyan egységes preferencia sorrend a Bernoulli-eloszlású célváltozóra alkalmazott teljesítménymetrikák esetében, mint folytonos esetben (9.3. fejezet) fennállt. Az AUC metrika alapján legjobban ítélt teljes GAM modell a csőd osztály recall mutatója és a kereszt-entrópia szerint bizonyul még legjobban preferált modellnek. Az eredmény érthető, hiszen a ROC görbe alatti terület növelése lényegében a kijelölt pozitív osztály (jelen esetben csődös ügyfelek) recall mutatójának (TPR) maximalizálását jelenti. Az FPR figyelembevétele inkább egy korlátozó tényezőként fogható fel a ROC görbe alatti terület maximalizálásakor. A kereszt-entrópia pedig az AUC-hoz hasonlóan egy vágási értéktől független teljesítménymetrika, ami a becsült csődvalószínűségek és a valós csődesemény bekövetkezési események eloszlásának hasonlóságát vizsgálja, így érthető, hogy hasonló modelleket preferálnak.

Kiemelendő, hogy a csőd osztály precision, valamint FPR mutatója szerint a döntési fa modell bizonyul a legjobb választásnak. Ennek oka, hogy az optimális vágási érték a döntési fa esetén a legmagasabb: csak 44,59%-os csődvalószínűség miatt osztályoz a modell egy ügyfelet csődveszélyesnek. Ez modell 17. ábrán (10.2. fejezet) megfigyelhető egyszerű osztályozási szabályrendszeréből adódik. Emiatt természetes, hogy az óvatosabb csődös osztályozás miatt

a csőd predikciók alacsonyabb arányban lesznek hibásak. Ám ennek az az ára, hogy a modell a tesztminta valóban csődös ügyfeleinek lényegesen alacsonyabb hányadát azonosítja helyesen. Tehát, lényegében az összes többi teljesítménymetrika szerint a döntési fa modell nyújtja a legrosszabb teljesítményt. A jelenség kezelésére alkalmazható a C5.0 algoritmus, ami a döntési fa rossz teljesítményén a fals negatív és a fals pozitív értékek súlyozásával kísérel meg javítani. Azonban, fontos megjegyezni, hogy a súlyok megadása során egy konkrét döntéshozó preferenciáit kell tükrözni (Quinlan, 1993).

A HGHK algoritmus AUC esetében a 10.5. fejezet alapján a döntési fa és COSSO algoritmusok modelljei mellett a páronkénti concurvity jelenségre kontrolláló mRMR és HSIC-Lasso eljárások által javasolt modelleknél is jobb becslési pontosságot nyújtott a concurvity korlátok megsértése nélkül. Ez a jó teljesítmény az csődös osztály recall és kereszt-entrópia mutatókra öröklődik. Ennek oka jelen esetben is a két mutató AUC metrikával vett hasonlósága, ami a korábban ismertetésre került. Viszont, ezt a jó teljesítményt a HGHK algoritmus modellje valamivel rosszabb precision és FPR értékek mellett produkálja, mint az mRMR és HSIC-Lasso modellek. Ráadásul, a döntési fa ebben a két mutatóban a legjobb teljesítménnyel bíró modell, így mind mutatókban a HGHK modellje csak a COSSO modellt múlja felül precisionben és a teljes modellt FPR-ben. Mivel a $bACC$ és F_1 mutatók 8.3. fejezetben ismertetett képletei részlegesen figyelembe veszik a csőd osztály precision mutatóját, így a HGHK rosszabb teljesítménye ezekre a mutatókra is öröklődik. Bár e két metrika alapján a HGHK modell jobb becslési pontossággal rendelkezik, mint a COSSO és döntési fa modellek, ám az mRMR és HSIC-Lasso teljesítményétől elmarad. Noha kisebb mértékben, mint a precision és FPR mutatók szerint, hiszen ezeket a $bACC$ és F_1 metrikák csak részben, 50%-os súllyal veszik figyelembe.

Amennyiben a két concurvity korlátokat nem sértő modellt a HGHK-t és a döntési fát vizsgáljuk, akkor elmondható, hogy becslési pontosság szempontjából kiegészítik egymást: a HGHK jobb „recall-jellegű” mutatókkal rendelkezik: a valódi csődveszélyes ügyfelek nagyobb hányadát jelöli meg a tesztmintán, míg a döntési fa osztályozás a „precision-jellegű” metrikákban nyújt jó teljesítményt: a modell által csődösnek jelölt ügyfelek nagyobb arányban lesznek a valóságban is csődöt jelentők, ám az összes valódi csődveszélyes ügyfélnek kisebb hányadát azonosítja a modell.

A csődkockázat modellezések során a döntéshozók általában a „recall-jellegű” teljesítménymetrikákat részesítik előnyben, mivel a hitelintézet számára nagyobb költség egy csődöt jelentő ügyfél fel nem fedezése, mint egy ügyfél téves megkeresése egy adósságrendező

akciótervvel (Moula, 2017). Tehát, a döntéshozóknak relevánsabb metrikák körében a HGHK algoritmus modellje alapján érdemes azonosítani a nem redundáns hitelkockázati tényezőket.

A 10.5. és 10.6. fejezetek eredményeit összefoglalva a K2 kutatási kérdésre a 9.1 és 9.2. alfejezetekben adott válaszok a következő megállapításokkal bővíthetők. A banki ügyfelek adatbázison tapasztalt eredmények megerősítik a K2 kutatási kérdésre adott korábbi válaszunkat, amely szerint a HGHK algoritmus modelljeinek tesztmintán mért becslési pontossága nagyságrendileg nem marad el az értekezésben vizsgált egyéb algoritmusok által javasolt modellek teljesítményétől. Sőt, a 6. táblázatban közölt eredmények alapján az is kijelenthető, hogy HGHK algoritmus modelljének banki ügyfelek adatbázison elért becslési pontossága a Yang – Zhang (2018)-féle változószelekció nélküli GLM modell teljesítményét meghaladja, és a szerzők által legpontosabbnak tartott LightGBM modell teljesítményétől sem marad el jelentősen. Továbbá, a HGHK algoritmus az egyetlen a vizsgált algoritmusok közül a banki ügyfelek adatbázison, ami a concurvity jelenségtől teljesen mentes változóhalmazt javasol. Ez utóbbi eredményt még a concurvity jelenségre kontrolláló mRMR és HSIC-Lasso algoritmusok sem tudják minden esetben biztosítani, mivel a jelenséget az algoritmusok csak magyarázóváltozó-páronként vizsgálják.

A banki ügyfelek adatbázison a teljesítménymetrikák függvényében változatosabb eredményeket tapasztaltunk, mint a 9. fejezet betongerendák adatbázisán. Itt a két concurvity korlátokat egyik változóban sem sértő modellt, a HGHK-t és a döntési fát vizsgáljuk, akkor elmondható, hogy becslési pontosság szempontjából kiegészítik egymást. Azonban, a banki csődkockázati elemzésekben a döntőshozóknak a „Recall-jellegű” metrikák a relevánsabbak, amiben a HGHK lényesen jobban teljesít, mint a döntési fa modell. Emiatt azt javasolhatjuk, hogy HGHK algoritmus modellje alapján érdemes azonosítani a nem redundáns hitelkockázati tényezőket.

A HGHK algoritmus megfelelő teljesítménye (concurvity jelenségtől mentes változóhalmaz mellett a teljesítménymetrikák körében hasonló pontosság elérése, mint a többi vizsgált algoritmus esetén és a várható futásidő elfogadhatósága a többi vizsgált algoritmuséhoz képest) a banki ügyfelek adatbázison megerősíti a K4 kutatási kérdésre adott válaszunkat is: érdemes a HGHK GAM keretben tapasztalt rossz minőségű kezdeti memóriáját (populációját) az algoritmus több véletlen kezdeti memóriából történő futtatásával kezelni kisebb generációszám mellett.

11. A HGHK algoritmus skálázhatóságának vizsgálata virtuális architektúrák segítségével

A skálázhatóság általános definíciója alapján a fogalom egy rendszer azon tulajdonságát jelenti, hogy képes megnövekedett feladatmennyiséget kezelni új erőforrások bevonásával vagy meglévő erőforrások elvételével (Bondi, 2000).

A számítástudomány területén a fenti skálázhatóság fogalom számítógépek hálózatok, algoritmusok, programok és alkalmazások egy jellemző tulajdonsága. Például egy keresőmotor azon tulajdonsága, hogy pótlólagos hardvererőforrások igénybevételével képes kiszolgálni egy folyamatosan növekvő felhasználószámot és indexálandó témamennyiséget (Kenmeth et al., 2008).

A skálázhatóság két csoportját különíti el a szakirodalom: horizontális és vertikális skálázhatóság (Michael et al., 2007).

Horizontális skálázás esetén több csomópontot adunk (vagy veszünk el) egy rendszerhez. Például, új számítógépet teszünk elérhetővé egy elosztott módon működő alkalmazásnak. Például egy webalkalmazást egy helyett három szervergépen futtatunk. Magas teljesítményigényű számítástudományi alkalmazások a szeizmikus hullámok elemzése és a biotechnológia területén horizontális skálázás segítségével valósítják meg olyan feladatok költséghatékony megoldását, amelyekhez korábban drága szuperszámítógépek alkalmazására volt szükség. A komplex társadalmi hálózatok elemzése jelenleg kimeríti a legnagyobb teljesítményű szuperszámítógépek kapacitását is, így a legtöbb probléma a területen csak skálázható rendszerek segítségével vizsgálható (Ali, 2019).

Ezzel szemben, vertikális skálázás esetén többleterőforrást juttatunk a vizsgált rendszer egy csomópontjához. Ez jellemzően egy személyi számítógép vagy munkaállomás processzormagjainak számának vagy RAM méretének növelését (vagy csökkentését) jelenti (Ali, 2019).

A többleterőforrással ellátott rendszerek alaposabb tervezést igényelnek, hiszen a rendszer különböző feladatait allokálni szükséges az új erőforrásokhoz. Ebből adódóan olyan jelenségeket is kezelni kell, mint az áteresztőképesség több processzormag esetén és az erőforrások részeredményeinek aggregálásából adódó késleltetés. Továbbá, azt is fontos szem előtt tartani, hogy bizonyos rendszerek horizontálisan nem is skálázhatók (Zomaya, 2021).

Jelen értekezésben a HGHK algoritmus vertikális skálázhatóságát vizsgáljuk meg, hiszen a memóriában lévő megoldásokhoz tartozó modellek szimultán kiszámításának hatékonysága függ az alkalmazott processzormagok számától az algoritmus futtatására használt személyi

számítógépen. Az algoritmus skálázhatóság-vizsgálatának kereteit a 11.1. alfejezetben ismertetjük, míg az elvégzett numerikus kísérletek eredményeit a 11.2. alfejezetben mutatjuk be.

11.1. A HGHK algoritmus skálázhatóság-vizsgálat keretei

A HGHK algoritmus vertikális skálázhatóságát a memóriában lévő megoldásokhoz tartozó modellek szimultán kiszámítására allokált processzormagok számának függvényében vizsgáljuk. A vizsgálatot a banki ügyfelek adatbázison végezzük el, hiszen a 9.1. és 10.5. alfejezetek eredményei alapján a várható futásidő egyértelműen ebben, a nagyobb méretű adatbázisban meghatározó kérdés a HGHK algoritmus hatékonysága szempontjából.

A 9. és 10. fejezetekben elvégzett numerikus kísérletek mögötti hardverkonfiguráció az 1. fejezetben megadott személyi számítógép volt, amelynek releváns paraméterei: 12 processzormag, 2,20 GHz alapsebességgel és 8 GB RAM. A skálázhatóság numerikus vizsgálatához használt hardvererőforrások körét Microsoft Azure Data Science Virtual Machine segítségével bővítjük. A platformon egyetemi környezetben elérhető hardverkonfigurációk segítségével 4, 8, 12 és 16 maggal és 2,60 GHz alapsebességgel rendelkező Intel Xeon Platinum 8272CL processzorok érhetők el, 24 GB RAM-mal kiegészítve (Etaati, 2019).

A numerikus kísérleteket a 12 processzormagos konfiguráció esetén is megismételjük, hiszen az eltérő RAM kapacitást is figyelembe kell venni a 9. és 10. fejezetekben használt hardverkonfigurációhoz képest a HGHK algoritmus párhuzamosított teljesítményeinek összehasonlítása során.

A skálázhatóság numerikus vizsgálata előtt elengedhetetlen megállapítani a legfontosabb viszonyítási alapot, a párhuzamosítás által elérhető maximális gyorsítás mértékét, korlátlan számú elérhető processzormag esetén. Ezen felső határ megállapításához Amdahl-törvénye alkalmazható (Amdahl, 1967) (Rodgers, 1985) (Bryant, 2016). Amdahl-törvénye a teljes feladat/algoritmus párhuzamosítás miatti sebességnövekedésének felső határát (44) módon határozza meg.

$$S = \frac{1}{(1-p) + \frac{p}{s}} \tag{44}$$

(44)-ben s jelöli a sebességnövekedést, ami a párhuzamosítással elérhető az algoritmus párhuzamos végrehajtásra alkalmas részében. Továbbá, p jelöli az algoritmus egy processzormagon történő végrehajtása során a futásidőnek azon hányadát, amit az algoritmus párhuzamosítható része kitesz. Ez egy egyszerű példán azt jelenti, hogy ha egy algoritmus

futásidejének 30%-a párhuzamos kiszámításra alkalmas, és a párhuzamosítással ezen rész az algoritmusban 2-szer gyorsabban végrehajtható, akkor Amdahl-törvényében $p = 0{,}3$ és $s = 2$. Ezek alapján pedig az algoritmus összességében legfeljebb $S = \frac{1}{(1-p)+\frac{p}{s}} = \frac{1}{(1-0{,}3)+\frac{0{,}3}{2}} =$ 1,18-szor gyorsabban végrehajtható a soros futtatáshoz képest.

A HGHK algoritmus 7. ábrán adott folyamatábrája, valamint a 7. és 10. fejezet alapján az algoritmus párhuzamosítható része az új egyedek generálása és az egyedekhez tartozó GAM-ok kiszámítása, ami a genetikus algoritmustól örökölt egyedkezelés miatt szimultán megoldható. A HGHK algoritmust nem párhuzamosított módon, 20-as replikációszámmal futtattuk banki ügyfelek adatbázison, a 10.4. fejezetben megadott paraméterekkel. A futtatásokból megállapítható, hogy a 20 futtatásban a teljes futásidőnek átlagosan 92,259%-át teszi ki a párhuzamosítható egyedgenerálás, 6,345%-pontnyi szórással. Ez alapján várhatóan a HGHK futásidejének legalább 89,3%-a párhuzamosítható, 95%-os konfidencia-intervallum alsó határa szerint. Tehát, a HGHK esetében $p = 0{,}893$.

Az algoritmus párhuzamosítható részében elérhető maximális sebességnövekedés mértéke (s) a HGHK-ban a nagyjából populáció méretével egyezhet meg első benyomásra. Hiszen, ha egy populációban szereplő minden egyedhez tartozó GAM kiszámítását külön processzormagon lehet végezni, akkor az összesített futásidő a leghosszabb ideig tartó GAM számítási idővel egyezik meg, míg teljesen soros feldolgozás esetén minden GAM kiszámítási ideje összeadódna, és ez az összeg várhatóan egy GAM átlagos számítási idejének populáció méretszerese lenne. Természetesen, ez csak nagyságrendi közelítés, hiszen a modellben lévő változók száma sztochasztikussá teszi a különböző egyedekhez tartozó GAM-ok kiszámítási idejét.

Ugyanakkor, nem szabad arról megfeledkeznünk, hogy a 2. iterációtól kezdve minden populációban lesznek egyedek, amik változtatás nélkül öröklődnek az előző iteráció populációjából a 7.2. fejezet (30) formulája alapján. Az örökölt egyedekhez tartozó GAM-okat nem számítja újra az algoritmus, így ezek nem lesznek részei a párhuzamosítás miatti gyorsításnak. Ebből adódóan s megadásához korrigálnunk kell a populáció méretét a populációban jelen lévő örökölt egyedek átlagos arányával egy futtatás során. Ennek becsléséhez a HGHK nem párhuzamosított, 20-as replikációval történő futtatása során (amely futtatások alapján alapján p-t is meghatároztuk) elmentettük minden populációban az örökölt egyedek számát, amit a teljes populáció méretéhez, 60-hoz arányosítottunk. Mivel egy replikáció során az algoritmust 5 véletlenszerű induló populációval is megfuttattuk, és egy futtatás 6 iterációig tartott, így egy $20 \times 5 \times 6 = 600$ elemű mintából meg tudjuk adni

az örökölt egyedek várható arányának 95%-os megbízhatóságú konfidencia-intervallumát. Az eredményekből a konfidencia-intervallum felső határát használjuk a populáció méret korrekcióhoz, hiszen a mi szempontunkból ez prudens becslés: a több ismétlődés rontja a párhuzamosítással elérhető gyorsítás mértékét.

A 600 elemű mintában az örökölt egyedek átlagos aránya 10,55%, 10,05%-pont szórással. Ezzel a 95%-os konfidencia-intervallum felső határa a várható arányra 11,36%. Ezzel a HGHK esetében az Amdahl-törvény szerinti $s = pop\_meret \cdot (1 - 0{,}1136)$. Ami a 10.4. fejezetben megadott 60-as populáció méret mellett $s = 60 \cdot (1 - 0{,}1136) = 53{,}2$-t jelent.

Összességében tehát a HGHK algoritmusban Amdahl-törvénye alapján maximálisan $S = \frac{1}{(1-p)+\frac{p}{s}} = \frac{1}{(1-0{,}893)+\frac{0{,}893}{53{,}2}} = 8{,}078 \approx 8$-szoros gyorsítás lenne elérhető korlátlan számú elérhető processzormag esetén a soros végrehajtáshoz képest.

Természetesen, mint korábban jeleztük ez a 8-szoros sebességnövekedés a párhuzamosítással egy elméleti maximum, amelynek eléréséhez szükséges erőforrások (kb. 53 processzormag) nem állnak a rendelkezésünkre.

Ugyanakkor, Amdahl-törvénye segítségével megbecsülhető a várható gyorsítás mértéke az általunk elérhető 4, 8, 12 és 16 mag mellett is. Hiszen, ezen esetekben az algoritmus párhuzamosítható részében elérhető maximális sebességnövekedés mértéke (s) az elérhető processzormagok számával fog megegyezni. Ekkor nem is szükséges korrigálni az örökölt egyedek átlagos arányával, mivel eleve nincs annyi erőforrásunk, hogy az alkalmazott pop_meret paraméternek (esetünkben 60) akár csak a felét kitevő egyedekhez tartozó GAM-okat külön processzormagokon számítsuk ki. Ezek alapján pl. 4 processzormag esetén az elérhető maximális gyorsítás mértéke Amdahl-törvénye alapján $\frac{1}{(1-0{,}893)+\frac{0{,}893}{4}} = 3{,}03$-szoros. A többi vizsgált magszám esetén az eredményeket a 8. táblázat tartalmazza.

Magok száma	**Gyorsítás mértéke (Amdahl)**
1	1,00
4	3,03
8	4,57
12	5,51
16	6,14

8. táblázat: A vizsgált processzormagok száma mellett elérhető maximális gyorsítás mértéke a HGHK algoritmusban Amdahl-törvénye szerint. Forrás: saját szerkesztés.

Láthatjuk, hogy a 16 mag esetén elérhető 6,14-szeres maximális gyorsítás is már elég közeli érték a 10.4. alfejezet beállításainak megfelelő 60-as populációméret mellett elérhető abszolút maximális 8,08-szoros gyorsításhoz is (amihez kb. 53 mag alkalmazása szükséges).

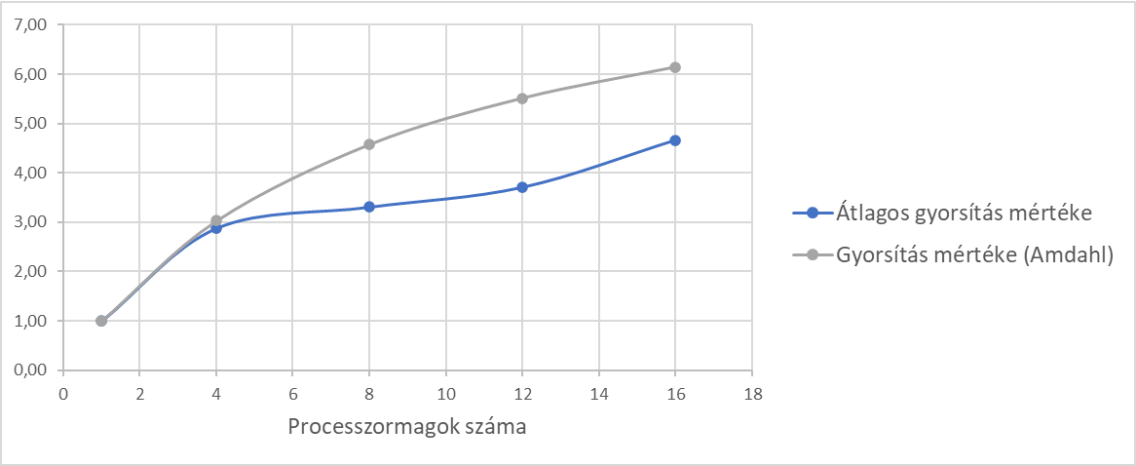
11.2. A HGHK skálázhatóság-vizsgálatának numerikus eredményei

A 11.1. fejezetben ismertetett elvek mentén elvégzett numerikus kísérletek eredményeit a 9. táblázat szemlélteti.

Magok száma	Átlagos futásidő (perc)	Futásidők szórása (perc)	Futásidők relatív szórása (%)	Átlagos gyorsítás mértéke
1	550,910	32,876	5,97%	1,00
4	191,560	12,587	6,57%	2,88
8	166,727	13,776	8,26%	3,30
12	148,633	16,293	10,96%	3,71
16	118,203	45,487	38,48%	4,66

9. táblázat: A HGHK skálázhatóság-vizsgálatának numerikus eredményei. Forrás: saját szerkesztés.

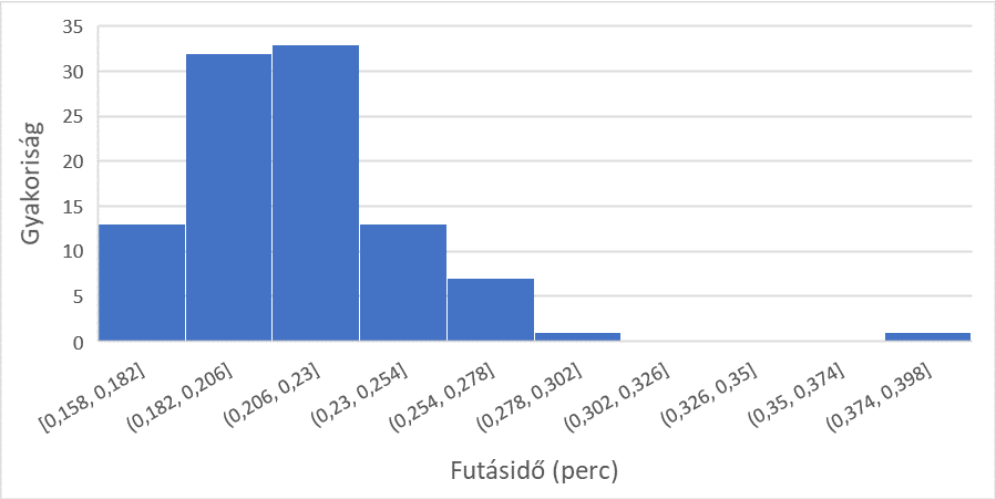
A 9. táblázat alapján a numerikus vizsgálatok megerősítik, hogy már 4 processzormag alkalmazása is majdnem 3-szoros gyorsítást képes elérni a HGHK algoritmus várható futásidejében. Viszont, a magok számának 8-ra, illetve 12-re növelése igazán látványos érdemi gyorsítást nem okoz, 12 mag esetén is csak 3,71-szeres gyorsítást érünk el a soros végrehajtáshoz képest, szemben az Amdahl-törvény alapján lehetséges 5,51-gyel. A következő jelentősebb gyorsulás a várható futásidőben 16 mag alkalmazása esetén következik be, amikor a soros végrehajtáshoz képest már 4,66-szoros gyorsítást tudunk elérni, és a várható futásidő már 2 óra (120 perc) alatti. Azonban, 16 mag esetén a futásidők relatív szórásában is jelentős emelkedést tapasztalunk. Az egyes konkrét futásidők a várható futásidő majdnem 40%-os környezetében ingadoznak átlagosan. Ez a 12 mag esetéhez képest majdnem 4-szeres bizonytalanság növekedést eredményez, ami szerint a 118 perces átlagos futásidőhöz képest egy 45 perces eltérés is még egy átlagos eltérésnek számít 16 mag esetén. Továbbá, a 4,66-szoros gyorsítás is messze van az Amdahl-törvény segítségével meghatározott 6,14-szeres maximumtól. A numerikus eredmények által kimutatott gyorsítás mértékeket a soros végrehajtáshoz képest érdemes külön ábrán összevetni az Amdahl-törvény alapján számolt maximális értékekkel (8. táblázat). Az összevetés eredményei a 20. ábrán láthatók.



20. ábra: A HGHK algoritmuson a párhuzamosítás által elért átlagos gyorsítás mértéke és az Amdhal-törvény segítségével számított maximális gyorsítás mértéke az alkalmazott processzormagok számának függvényében. Forrás: saját szerkesztés.

A 20. ábráról leolvasható, hogy míg 4 processzormag alkalmazása esetén a numerikusan mért várható gyorsítás lényegében megegyezik az Amdahl-törvény segítségével számolt maximummal, addig 8 mag esetén a két érték már jelentősen eltér. Viszont, 12 és 16 mag esetén a különbség az empirikus és a maximális érték között szűkülni kezd, noha ez a jelenség 16 mag esetén már nem elhanyagolható mértékű relatív szórás mellett történik. Emiatt a maximális gyorsítás mértékéhez történő közelítést 16 mag esetén nem szabad általános tendenciának tekinteni a magas szóródás miatt a vizsgált futtatások között. Továbbá, a numerikusan mért várható gyorsítások növekedési ütemének tendenciája is megtöri az Amdahl-törvény alapján várható logaritmikus függvényformát: 8 és 12, valamint 12 és 16 mag között a növekedés üteme magasabb, mint 4 és 8 mag között (bár 12 és 16 mag között ismét figyelembe kell venni a 16 magos eset magas relatív szórását).

A 8 magtól tapasztalt jelenségek (növekvő relatív szórás, logaritmikustól eltérő gyorsítás növekedési ütem és jelentősebb eltérések az empirikus és a maximális gyorsítás mértéke között) mögött egységesen a HGHK 9.3. alfejezetben megadott, az algoritmus véletlen elemeit preferáló optimális paraméterezés áll. Hiszen az optimális paraméterezés (alacsony kezdeti $HMCR$ és magas kezdeti bw valószínűségek) esetén a HGHK iterációi során egy populációban a nem az előző populációkból öröklődő egyedek többségének generálása során a véletlen hatások dominálnak. A 10.1. alfejezetben vizsgáltuk már 100 véletlenszerű egyedet (változókombinációt) szimulálva egy GAM átlagos kiszámítási idejét a HGHK-ban is alkalmazott 5000 elemű tanítóhalmaz esetén. A HGHK viselkedésének megértéséhez 8 mag feletti párhuzamosítás esetén a 100 szimulált egyedhez tartozó GAM-ok kiszámítási idejeinek eloszlását érdemes tanulmányozni a 21. ábrán található hisztogram segítségével.



21. ábra: A 10.1. fejezetben vizsgált 100 véletlenszerűen generált magyarázóváltozó részhalmazhoz tartozó GAM-ok kiszámítási idejének hisztogramja. Forrás: saját szerkesztés.

A 21. ábra hisztogramja alapján a különböző változókombinációk esetén a szimulációban vizsgált 100 GAM számítási ideje enyhén jobbra elnyúló eloszlású 1 kiugró pont kivételével, ahol a futásidő 0,391 perc, ami a futásidők 95. percentilisének (0,267 perc) majdnem 1,5-szöröse. Tehát, a szimuláció alapján azt mondhatjuk, hogy 1% körüli a tapasztalati valószínűsége annak, hogy egy jelentős mértékben kiugró futásidejű GAM-mal rendelkező változókombináció kerüljön be egy populáció egyedei közé. Ami azért tud problémákat okozni a párhuzamosítás során, mert azt a processzormagot, amin a kiugró futásidejű GAM számítása fut, a többi processzormagnak (amiken többnyire átlagos futásidejű GAM-ok számítása zajlott) „be kell várnia” mielőtt elkezdődik a szelekciós művelet a populációban. Mindez azért szükséges, hogy az átlagos célfüggvény érték (30) formulával megadható legyen a populációban. (30) viszont csak az összes egyed ismeretében számítható egy populációban.

A leghosszabb futásidővel rendelkező egyed bevárása miatt a HGHK minden iterációjában így lesz egy holtidő, amikor egy processzormag kivételével a többi nem dolgozik.

A holtidő miatti futásidő veszteség az alkalmazott processzormagok függvényében nő, mivel a kihasználatlan processzormagok számát növelik az extra magok. A holtidőben nem használt magok számának növekedésével pedig nő a távolság az empirikusan mért gyorsítás és az Amdahl-törvény alapján meghatározott maximális gyorsítás között. Az, hogy 16 mag esetén már újra közelíti az elméleti maximumot az empirikus gyorsítás a 20. ábrán, annak a következménye, hogy átlagosan többet nyer a HGHK a többlet magokkal abban a szakaszban, amikor mindegyik mag kihasználtan működik, mint amennyit a holtidőben elveszít. Ugyanakkor, ez az átlagok szintjén tapasztalt jelenség nem tekinthető általános tendenciának a magas relatív szórás miatt.

A magas relatív szórás oka 16 mag esetén az, hogy a holtidő hossza attól függ, hogy a populációban lévő kiugróan hosszú GAM számítási idővel bíró egyed mennyire számít kiugrónak. Ha nincs nagyon kiugró számítási idővel bíró egyed több iteráció populációjában sem, akkor a holtidők elég rövidek tudnak lenni, így az algoritmus egészének futásidejében meg lehet közelíteni az Amdahl-törvény szerinti maximális gyorsítást. Azonban, ha sok populációban található nagy mértékben kiugró számítási idejű egyed, akkor az ezek miatt bekövetkező holtidő kumulálva a teljes algoritmuson végzett gyorsítást is messzire teheti az elméleti maximumtól. Mivel a HGHK optimális paraméterezésében az algoritmus véletlenszerű elemei dominálnak, így a kiugró számítási idejű egyedek jelenléte a populációkban könnyen magas ingadozást mutathat. Ezen ingadozás hatása azért a 16 magos esetekben jelenik meg dominánsan, mert a kihasználatlan magok magas száma miatt itt a legnagyobb ezen véletlenszerű kiugró értékek miatti holtidő hatása a HGHK teljes futásidejére.

Jelen fejezetben csak olyan numerikus vizsgálatokat végeztünk, ahol a maximális gyorsítás mértéke Amdahl-törvénye segítségével jól becsülhető, azaz teljesül az a feltétel, hogy a vizsgált probléma mérete állandó. Ez esetünkben úgy értendő, hogy a HGHK-ban alkalmazott populációméret, tehát a HGHK párhuzamosítható részének mérete állandó, az algoritmus véletlen elemeit (az örökölt egyedek aránya egy populációban, a kiugró számítási idővel bíró egyedek jelenléte egy populációban) leszámítva. A numerikus gyorsítási eredmények eltérése az Amdahl-törvény alapján számítotthoz képest a HGHK véletlen elemeinek dominanciájával magyarázható az alkalmazott optimális paraméterezésben.

A kutatás továbbfejlesztése során vizsgálható a HGHK skálázhatósága a Gustafson – Barsis törvény alapján is, ami az Amdahl-törvénnyel ellentétben nem állandó problémaméretet feltételez, hanem azt, hogy a probléma mérete arányosan nő a bevont processzormagok számával, így nagyobb méretű feladatok oldhatók meg a futásidő növekedése nélkül (Gustafson, 1988). Ez a HGHK esetében a populációméret igazítását jelentené az elérhető processzormagok számához. A HGHK javasolt populációméreteinek megállapítása az elérhető processzormagok számának függvényében tehát további kutatások tárgyát képezheti. Egy optimális populációméret azonosításával lehetőség van a párhuzamosítás miatti gyorsítás elvi mértékének jobb megközelítésére is.

Jelen alfejezet eredményei alapján a K5 kutatási kérdés második, a HGHK skálázhatóságát érintő részét is meg tudjuk válaszolni. Az empirikus vizsgálatok alapján az Amdahl-törvénye szerint párhuzamosítással elérhető maximális gyorsítás mértékét a HGHK 4 mag esetén eléri, míg a többi esetben némileg elmarad tőle, aminek hátterében az algoritmus véletlen elemeit

preferáló optimális paraméterezés következtében bekövetkező holtidő áll. Ez a holtidő magas relatív szóródást is okoz a 16 magos hardver konfiguráción tapasztalt futásidőkben.

12. A HGHK algoritmus gyakorlati korlátai, további kutatási irányok

A HGHK algoritmus által javasolt GAM-ok legjelentősebb hátránya bizonyos alkalmazásokban a kihagyott változók okozta torzítás lehet. A jelenségre Wooldridge (2016) alapján adunk egy intuitív ismertetőt lineáris esetre.

Tegyük fel, hogy egy normális eredményváltozóval és identitás link függvénnyel adott GLM-et vizsgálunk, ahol tudjuk, hogy az eredményváltozót a következő modell írja le tökéletesen: $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \varepsilon$. Továbbá tegyük fel, hogy az is ismert, hogy X_2 magyarázóváltozó kifejezhető X_1 segítségével $X_2 = \delta_0 + \delta_1 X_1 + \zeta$ módon, ahol δ_0, δ_1 a regressziós modell paraméterei és ζ a modell hibatagja. Ebben az esetben tehát X_2 és X_1 között bizonyos mértékű multikollinearitás biztosan fennáll.

Vizsgáljuk meg mi történik, ha az eredeti, két magyarázóváltozót használó regressziós modell egyenletébe beírjuk a magyarázóváltozók közti összefüggést: $Y = \beta_0 + \beta_1 X_1 + \beta_2(\delta_0 + \delta_1 X_1 + \zeta) + \varepsilon = (\beta_0 + \beta_2\delta_0) + (\beta_1 + \beta_2\delta_1)X_1 + (\varepsilon + \beta_2\zeta)$. Az eredményül kapott regressziós modellben csak egy magyarázóváltozó, X_1 szerepel, melynek együtthatója $(\beta_1 + \beta_2\delta_1)$. Azaz X_2 magyarázóváltozó önálló marginális hatása Y-ra megjelenik X_1 regressziós együtthatójában. Amennyiben úgy építünk regressziós modellt ebben a példában, hogy X_2-t nem szerepeltetjük a magyarázóváltozók között, akkor a változó hatása a vele regressziós összefüggésben álló X_1 változó együtthatójába „költözik" a modellben, és a $\beta_2\delta_1$ tagot hívjuk az új X_1 együtthatóban a modellből kihagyott X_2 változó okozta torzításnak. Hiszen innentől kezdve X_1 együtthatója nem csak X_1 változó közvetlen Y-ra gyakorolt hatását hordozza, hanem az X_2 változón keresztül gyakorolt közvetett hatását is.

A kihagyott változó okozta torzítás a nem-lineáris GAM modellekben is fennállhat a concurvity jelenség 5.3. alfejezetben megfogalmazott definíciója miatt. Concurvity jelenség esetén ugyanis egy f_j függvény $\boldsymbol{U_{k_j} D_{k_j}}$ mátrixszal reprezentált bázisfüggvényeinek lineáris kombinációjával kinyerhető egy g_j függvény, ami más $f_{c\neq j}$ függvények $\boldsymbol{U_{k_c} D_{k_c}}$ reprezentációjának is része. Amennyiben a kérdéses $X_{c\neq j}$ változót elhagyjuk a GAM egyenletéből, akkor a $\boldsymbol{U_{k_c} D_{k_c}}$ mátrixszal reprezentált bázisfüggvények hatása $\boldsymbol{U_{k_j} D_{k_j}}$ mátrixhoz tartozó Ψ_{k_j} együtthatóvektor elemeiben „jelenik meg" a lineáris esethez hasonló módon.

A kihagyott változók okozta torzítása megjelenése a HGHK algoritmus által javasolt modellekben majdnem biztosra vehető, hiszen a concurvity jelenség szűrésével pont azt akarjuk elérni, hogy az egymással nem jelentős mértékben összefüggő, de az eredményváltozóval szoros kapcsolatban lévő magyarázóváltozókat automatikusan azonosítsuk. Azon

magyarázóváltozók, amik rendelkeznek ezzel a két tulajdonsággal, értelemszerűen bizonyos kihagyott változók eredményváltozóra gyakorolt hatását is reprezentálják. Gyakorlatilag pont ezért kerestük őket. Például a banki ügyfelek adatbázis vizsgálata esetében a HGHK algoritmus modelljében kifejezetten előnyös, hogy csak a PAY_0 változó jelenik meg, hiszen ez a változó az összes többi PAY_*X* változó hatását magában hordozza a csődvalószínűségre. Hasonlóan, a modellben szereplő PAY_AMT3 változó minden bizonnyal magában hordozza az összes többi PAY_AMT*X* és BILL_AMT*X* változó csődvalószínűségre gyakorolt hatását. De jelen vizsgálatban a modellnek ez a torzítás nem hátránya, hiszen azonosítottunk egy szűk változóhalmazt, amivel elég jó pontossággal és könnyen értelmezhető módon leírható az ügyfelek csődvalószínűsége. A kinyert információ segítségével pedig a bank ügyfélkapcsolati osztálya számára egyszerű szabályokat tudnak az elemzők megfogalmazni arra nézve, hogy mikor kell egy ügyfelet interveniálni amiatt, mert nagy valószínűséggel a következő hónapban csődöt jelent.

Viszont, vizsgáljunk meg egy másik, némileg sarkított példát. Tegyük fel, hogy szeretnénk megvizsgálni, hogy az USA különböző településein regisztrált koronavírus fertőzöttek számát a települések milyen egyéb tényezői magyarázzák. Lehet, hogy hallottunk arról az összeesküvés elméletről[7], hogy a koronavírus fertőzéseket valójában az 5G kommunikációra képes mobiltelefon tornyok által kibocsájtott elektormágneses sugárzás okozza, így a lehetséges magyarázóváltozók közé felvesszük azt is, hogy egy településen hány 5G képes mobiltelefon torony van. A lehetséges magyarázóváltozók között szerepel a települések népessége is, ami szorosan korrelál az 5G képes tornyok számával, hiszen a nagyobb népességet több toronnyal lehet kiszolgálni. Ugyanakkor, a népesség biztosan szorosan korrelál a településen regisztrált koronavírussal fertőzöttek számával is, hiszen nagyobb alappopuláció esetén várhatóan többen fertőződnek meg. Egy ilyen szituációban a HGHK algoritmus modellje szigorú concurvity mértékre szabott korlát mellett akár egy olyan modellt is eredményezhet a fertőzöttek számára, amiben az 5G tornyok száma szerepel, de a népesség nem. Ezzel a népesség hatása a fertőzöttek számára az 5G tornyok számában jelenhet meg, ezzel azt sugallhatva, hogy igazából az 5G tornyok számának növekedésével arányosan nő a fertőzöttek száma a vizsgált településeken.

A fenti példa, mint említettük is némileg sarkított, hiszen a jelenség könnyen kezelhető, ha egy főre vetítve vizsgáljuk a változókat, de akár abban is bízhatunk, hogy az 5G tornyok számának szerepeltetése a modellben a népesség helyett olyan mértékű csökkenést okoz a becslési

[7] https://medium.com/@gabrieltraveler/are-5g-cell-towers-causing-the-coronavirus-symptoms-31be3847b870 Letöltve: 2020. 08. 14.

pontosságban, hogy a célfüggvényünk már nem fogja ezt az opciót preferálni. A példával csupán azt kívántuk érzékeltetni, hogy a kihagyott változók okozta torzítás a HGHK algoritmus modelljeiben problémás lehet, ha az algoritmus az alkalmazott próbákhoz választott magas szignifikancia-szint és a szigorú concurvity korlátok miatt fontos kontrollváltozókat hagy el a modellből. Ugyanakkor, fontos megjegyezni, hogy ha a HGHK által javasolt modellek nem is adják meg a valós oksági kapcsolatokat, kiindulási alapot jelenthetnek a legfontosabb, nem-redundáns változók körének meghatározására, amelyeket az elemző kifinomultabb modellek építéséhez kiindulópontként felhasználhat.

A kihagyott változók okozta torzítás hatásának alapos vizsgálatát további kutatásainkban tervezzük megtenni. Először elemezni kívánjuk az alkalmazott próbákhoz választott szignifikancia-szint és a concurvity mértékre szabott korlát szigorúságának hatását a HGHK algoritmus által preferált modellek magyarázóváltozóinak számára. Vizsgáljuk, hogy a korlátok szigorúságának létezik-e egy olyan szintje, ami mellett az algoritmus nem hagyja el a fontos kontrollváltozókat, de az értelmezést valóban károsító concurvity jelenséget okozó változókat már kiszűri a modellből. Amennyiben a korlátoknak egy ilyen értelemben optimális szintje általánosan nem adható meg, a HGHK algoritmust továbbfejleszthetjük úgy, hogy a felhasználó megadhassa a magyarázóváltozóknak egy olyan körét, amelyeket az algoritmusnak mindenképpen szerepeltetnie kell a modellben. A kihagyott változók okozta torzítás hatásának vizsgálatára jó adatbázis lehet a KSH tájékoztatási adatbázisában elérhető éves településstatisztika. Az itt szereplő adatok alapján meg lehet vizsgálni például, hogy a magyar településeken az egy főre jutó praktizáló orvosok számát a település milyen egyéb tényezői befolyásolják. Egy ilyen kutatás keretében a modellben mindenképpen szerepeltetni kell bizonyos demográfiai és gazdasági kontrollváltozókat: pl. népsűrűség, munkanélküliségi ráta, SZJA adatok alapján becsülhető egy főre jutó jövedelem stb.

Az algoritmus második kritikus pontja a várható futásideje nagyobb adatbázisokon. A 10. fejezetben az algoritmus futásideje nagyságrendileg nem hosszabb egy RFE változószelekciós algoritmussal kombinált véletlen erdőnél, sőt a 11. fejezet eredményei alapján 16 processzormag esetén várható értékben a HGHK futásideje kedvezőbb is a véletlen erdőnél (bár a szóródás magas). Viszont, a banki ügyfelek adatbázison a modellparaméterek jelentős változása nélkül tudtunk alkalmazni $n = 5000$ elemű almintákat a modellek tanítása során. Ez a megoldás nem mindig elérhető. Különösen akkor lehet problémás az alminták alkalmazása, ha a célunk egy olyan Bernoulli-eloszlású eredményváltozó modellezése, amiben a vizsgált esemény kimondottan ritka, bekövetkezési valószínűsége csak 5-10%. Ilyen esetben nem feltétlenül engedhetjük meg, hogy olyan megfigyeléseket ne szerepeltessünk a tanító

mintában, amelyeken az eredményváltozóban vizsgált esemény bekövetkezett. A tanító minta mesterséges kiegyensúlyozása sem feltétlenül vezet mindig a megfelelő eredményre (Weiss, 2004), így a tanításhoz szükséges az összes megfigyelés használata. Az ilyen esetek kezelése miatt fontos látni, hogy a HGHK algoritmus mekkora méretű adatbázisokat tud kezelni egy előre megadott várható futásidő korláton (pl. 24 óra) belül. Ritka események modellezésére elsősorban a biztosítási események vizsgálata során van szükség, így érdemes lehet pl. De Jong – Heller (2008) kötelező gépjármű felelősségbiztosítás káreseményeire vonatkozó adatbázisát vizsgálni további kutatásaink során. Az adatbázisban 67 856 megfigyelés található és a káresemények relatív gyakorisága 6,8%.

De Jong – Heller (2008) adatbázisa nagy mérete miatt a HGHK skálázhatóságának további vizsgálatára is alkalmas lehet. Az adatbázis nagy mérete igényli, hogy a HGHK populáció mérete minél magasabb legyen a több véletlenszerű kezdőpopulációból történő indítás mellett is. Egy ilyen szituációban pedig ideálisan vizsgálható a HGHK skálázhatósága a Gustafson – Barsis törvény alapján a 11.2. alfejezetben javasolt módon, ami a populációméret az elérhető processzormagok számához igazítását jelenti esetünkben.

13. Összefoglalás és diszkusszió

Jelen értekezésben bemutattunk egy hibrid genetikus – harmónia kereső algoritmust általánosított additív modellek változószelekciós feladatának megoldására. A vizsgált algoritmus szakirodalmi egyediségét, működési elvét, implementációját és numerikus hatékonyságvizsgálatát bemutató kutatásunkat a Design Science módszertan Hevner et al. (2004) hét pontos ajánlásának figyelembevételével valósítottuk meg.

Ebben a kutatásban korszerű forrásokra épülő irodalomkutatás segítségével áttekintettük a GAM-ok alapfogalmait, külön kitérve a magyarázóváltozók 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 magyarázóváltozók 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 plate spline-ok fogalmainak segítségével bemutattunk egy mértéket, amivel mérhető a concurvity jelenség, azaz megadható, hogy egy magyarázóváltozó mennyire jól kifejezhető más magyarázóváltozók nem-lineáris függvényeként.

Az értekezésben nyolc, GAM keretben alkalmazható változószelekciós algoritmust mutattunk be részletesen. Az algoritmusok áttekintése során megállapítottuk, hogy a regularizációs elven működő algoritmusok, valamint a Stepwise, GAMBoost és módosított backfitting eljárások egyáltalán nem kísérelnek meg a magyarázóváltozók közötti káros concurvity jelenségre kontrollálni a változószelekció során.

A Hall-féle CFS algoritmus kiterjesztéseként felfogható mRMR és HSIC-Lasso algoritmusok 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 magyarázóváltozó a modellben szereplő egyéb változók többváltozós függvényeként fejezhető ki.

Bemutattuk, hogyan építhető be a thin plate spline függvények alkalmazása a lineáris változószelekciót már kezelő HGHK algoritmusba. Megmutattuk, hogy a HGHK 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

memória méret alkalmazását és az algoritmus többszöri futtatását több véletlen kezdeti populációból kis maximális generációszám mellett.

A bemutatott algoritmusokat alkalmaztuk két valós adatbázison. Mindkét esetben a HGHK algoritmus végső modellje több (a feladatban releváns) teljesítménymetrika alapján is a legnagyobb becslési pontossággal rendelkező modell, amely teljesen mentes a concurvity jelenségtől. A kisebb méretű, betongerendák adatbázison történő futtatások során vizsgáltuk a HGHK 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 több véletlen kezdeti populációból történő 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 HGHK algoritmus alkalmazásakor nagyobb adatbázis esetében a futásidő így is jelentős, bár hasonló nagyságrendű, mint egy véletlen erdő algoritmus RFE változószelekcióval kombinálva. A HGHK skálázhatóságát a banki ügyfelek adatbázison négy virtuális hardver architektúrán vizsgáltuk, melyeken az algoritmus párhuzamosítását rendre 4, 8, 12 és 16 processzormagon hajtottuk végre. A numerikus eredmények alapján az Amdahl-törvénye alapján elérhető maximális gyorsítás mértékét a 4 magos architektúra erősen megközelíti, míg a 8, 12 és 16 magos architektúrákon nagyobb eltérések mutatkoznak. Ráadásul, a 16 magos architektúrán az eredmények relatív szórása is jelentős. Mindkét jelenség hátterében a HGHK algoritmus véletlen elemeit preferáló optimális paraméterezés következtében bekövetkező holtidő áll.

Mindezek alapján jelen értekezés K1-K5. kutatási kérdéseire tételesen is válaszolni tudunk. Ezen kutatási kérdésekre adott válaszok tekinthetők az értekezés téziseinek.

K1. Az első kutatási kérdés a HGHK algoritmus GAM keretre történő kiterjesztését vizsgálta a döntési változók és egyedreprezentáció szempontjából. Az 5.2. alfejezetben részletesen bemutatott thin plate spline függvények segítségével a 7. fejezetben megmutattuk, hogy GAM keretben a spline függvények rendjének és a szakaszhatárok helyének megválasztása automatizálható. Ezzel pedig a lineáris esetben alkalmazott döntési változókon és bináris egyedreprezentáción nem szükséges változtatni.

K2. A második kérdés a HGHK algoritmus által javasolt modellek minőségére koncentrált. A 9. és 10. fejezetben bemutatott két numerikus példa eredményei alapján elmondhatjuk, hogy a HGHK algoritmus modelljeinek tesztmintákon mért

becslési pontossága nagyságrendileg nem marad el az értekezésben vizsgált egyéb algoritmusok által javasolt modellek teljesítményétől. Sőt, a 6. táblázatban közölt eredmények alapján az is kijelenthető, hogy a HGHK algoritmus modelljének banki ügyfelek adatbázison elért becslési pontossága a Yang – Zhang (2018)-féle változószelekció nélküli GLM modell teljesítményét meghaladja, és a szerzők által legpontosabbnak tartott LightGBM modell teljesítményétől sem marad el jelentősen. Továbbá, a HGHK algoritmus az egyetlen a vizsgált algoritmusok közül, ami a concurvity jelenségtől teljesen mentes változóhalmazt javasol. Ez utóbbi eredményt még a concurvity jelenségre kontrolláló mRMR és HSIC-Lasso algoritmusok sem tudják minden esetben biztosítani, mivel a jelenséget az algoritmusok csak magyarázóváltozó-páronként vizsgálják. A 8. fejezetben bemutatott betongerendák adatbázison az mRMR algoritmus képes concurvity jelenségtől teljesen mentes változóhalmaz azonosítására (2. táblázat), ám a javasolt modell becslési pontossága elmarad a HGHK algoritmus modelljétől, valamint a betongerendák adatbázison az mRMR algoritmusban alkalmaztuk azt az a priori ismertet, hogy a korlátoknak megfelelő globális optimumban a magyarázóváltozók száma négy. A betongerendák adatbázis esetében az eredmények robusztusnak tekinthető a vizsgált teljesítménymetrikák körében. Az egyetlen másik concurvity korlátot nem sértő mRMR modellhez képest minden metrikában pontosabb becsléseket szolgáltat a HGHK által javasolt GAM. A banki ügyfelek adatbázison a teljesítménymetrikák függvényében változatosabb eredményeket tapasztaltunk. Ezen az adatbázison a két concurvity korlátokat nem sértő modellt a HGHK-t és a döntési fát vizsgáljuk, akkor elmondható, hogy becslési pontosság szempontjából kiegészítik egymást. Azonban, a banki csődkockázati elemzésekben a döntőshozóknak a „Recall-jellegű” metrikák a relevánsabbak, amiben a HGHK lényesen jobban teljesít, mint a döntési fa modell. Emiatt azt javasolhatjuk, hogy HGHK algoritmus modellje alapján érdemes azonosítani a nem redundáns hitelkockázati tényezőket.

K3. A harmadik kutatási kérdésben a HGHK algoritmus rekombinációs operátorainak vizsgálatát céloztuk. A 9.3. fejezet eredményei alapján elmondható, hogy GAM keret esetén a HGHK algoritmusban a teljesen véletlen új egyed generálásra érdemes nagyobb súlyt helyezni, de nem érdemes teljesen elhagyni a memóriából történő kontrollált egyed generálást sem. Az algoritmus paramétereinek ceteris paribus elvű érzékenységvizsgálata alapján a korábbi memóriából történő választás valószínűségét ($HMCR$) egy kimondottan alacsony értékről, 5%-ról érdemes növelni

35%-ig az algoritmus futtatása során. Tehát, a rekombináció során a korábbi memória egyedei közül történő választásra kis súlyt érdemes helyezni, de teljesen elhagyni nem érdemes ezt az elemet. Sőt, az algoritmus futtatása során a korábbi memóriából történő választást érdemes növelni is! Az algoritmus véletlen elemeit preferáló, de az irányított elemeket is némiképp megőrző paraméterezés hatékonyságát támasztja alá a 9.3. fejezet azon eredménye is, hogy a memóriából történő új egyed előállítás esetén is érdemes az algoritmus első lépéseiben magas mutációs (bw) valószínűséget alkalmazni (90%), ám az induló érték nagyon alacsonyra (10%) csökkentése az algoritmus futása során szintén kifizetődő.

K4. A negyedik kutatási kérdésünkben a HGHK algoritmus optimális memória (vagy populáció) méretét és kilépési feltételét vizsgáltuk. A 9.3. fejezetben, a kisebb méretű betongerendák adatbázison elvégzett ceteris paribus elvű érzékenységvizsgálat rávilágított arra, 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 a keresési térnek csak kb. 0,4 részét vizsgálja át. A tapasztalat így arra enged következtetni, hogy előnyösebb a 7. fejezetben bemutatott második stratégiát alkalmazni a kezdeti memóriák rosszabb minőségének kezelésére: Egy futásidőben könnyen kezelhető, kisebb memória méretet választunk, és több véletlen kezdeti memóriából (azaz populációból) lefuttatjuk az algoritmust egy alacsonyabb maximális generációszám mellett.

K5. Utolsó kutatási kérdésünkben a HGHK algoritmus párhuzamosítási stratégiáira helyeztük a fókuszt. A 10.1. fejezetben a nagyobb méretű, banki ügyfelek adatbázison az $n = 5000$ elemű alminták mellett elvégzett kísérletek eredménye alapján az 5.2.4. alfejezetben bemutatott QR dekompozíció segítségével megvalósított párhuzamosítás esetén az összes lehetséges megoldás előállításához várhatóan 26,46 év szükséges. Ez a várható idő kb. 0,16-szorosára (azaz kb. 4,18 évre) csökkenthető, ha több részhalmazhoz tartozó modellt számítunk ki egyszerre párhuzamosan. Tehát, a HGHK algoritmusban jobban kifizetődik több egyedhez tartozó modellek egyszerre történő kiszámítása, mint egy modell kiszámításának párhuzamosítása. Párhuzamosítás segítségével a HGHK algoritmus várható futásideje a nagyobb méretű banki ügyfelek adatbázison bő két órának tekinthető, ami egy RFE változószelekcióval kombinált véletlen erdő algoritmus várható futásidejével nagyságrendileg összemérető. Továbbá, jelentősen jobb teljesítményt

jelent a GAMBoost algoritmus 11 óra feletti várható futásidejénél. Az algoritmus párhuzamosításának skálázhatóságát 4, 8, 12 és 16 processzormaggal rendelkező virtuális architektúrákon végeztük el. Az empirikus vizsgálatok alapján az Amdahl-törvénye szerint párhuzamosítással elérhető maximális gyorsítás mértékét a HGHK 4 mag esetén eléri, míg a többi esetben némileg elmarad tőle, aminek hátterében az algoritmus véletlen elemeit preferáló optimális paraméterezés következtében bekövetkező holtidő áll. Ez a holtidő magas relatív szóródást is okoz a 16 magos hardver konfiguráción tapasztalt futásidőkben.

A HGHK algoritmus K1-K5. kérdések mentén történő vizsgálata során az algoritmus gyakorlati alkalmazásának két fontos korlátjára derült fény. Az algoritmus által javasolt modellek paramétereiben a kihagyott változók okozta torzítás tapasztalható a szigorú szignifikancia és concurvity korlátok miatt. Ez a torzítás értelmezési gondokat okozhat, ha az algoritmus fontos kontrollváltozókat is kihagyott a végső modellből. További korlát lehet a várható futásidő szempontjából, ha ritka eseményt leíró Bernoulli-eloszlású eredményváltozó modellezése a feladatunk. Ugyanis, ekkor nem feltétlen tudunk élni a $n = 5000$ elemű alminták alkalmazásával. A skálázhatóság (több processzormag alkalmazása) javít a várható futásidőn, ám magas magszám esetén a futásidők szóródásának növekedésével kell számolni, és az elvi maximális gyorsítástól való elmaradással.

Az azonosított korlátokat későbbi munkáinkban kívánjuk feloldani. Tervezzük a HGHK algoritmus eredményeinek érzékenységvizsgálatát az alkalmazott próbákhoz választott szignifikancia-szint és a concurvity mértékre szabott korlát szigorúságának függvényében. Továbbá, tervezzük elemezni a HGHK algoritmus viselkedését a jelen tanulmányban vizsgált adatbázisokhoz képest eltérő viselkedésű, gépjármű felelősségbiztosítási káreseményeket vizsgáló új adatbázison is. Ezek az új adatbázisok nagyobb méretük miatt lehetőséget adnak arra, hogy megvizsgáljuk a HGHK skálázhatóságát a Gustafson – Barsis törvény alapján is. Ez alapján megvizsgálhatjuk, hogy a populációméret processzormagok számához történő igazításával lehetséges-e a párhuzamosítás miatti gyorsítás elvi mértékének jobb megközelítése.

Eredményeink általános összevetése a szakirodalom eredményeivel nem könnyen kivitelezhető feladat abból adódóan, hogy a GAM keretben alkalmazható változószelekciós algoritmusokkal és azok hatékonyságával kapcsolatos szakirodalomban a kiválasztott magyarázóváltozók redundanciája jellemzően nem, vagy csak korlátozott mértékben szempont, ahogy a 6. és 7. fejezetek irodalomkutatásai során bemutattuk.

Általánosságban elmondható, hogy az értekezés irodalomkutatása során megvizsgált tanulmányokban szereplő GAM keretben működő változószelekciós algoritmusokban a feladat célfüggvénye a 4. fejezet (2) formulája szerinti alakú: modell illeszkedését törekszik javítani a változószelekció, ám a célfüggvény bünteti olyan magyarázóváltozók bevonását, amelyek a modell illeszkedését nem javítják eléggé. Egyéb korlátok (pl. multikollinearitás vagy concurvity jelenség elkerülésére a végső modellben) megfogalmazására jellemzően nem kerül sor a változószelekció során. Ez részben ismét annak tudható be, hogy az idézett munkákban a magyarázóváltozók hatásainak visszafejtése nem szempont. Egyedül az mRMR és HSIC-Lasso algoritmusok célfüggvénye védekezik a modellbe válogatott változók közötti redundancia, tehát a multikollinearitás vagy a concurvity ellen. Viszont, az algoritmusok nem kezelik azt az esetet, amikor egy magyarázóváltozó több más magyarázóváltozó többváltozós függvényeként áll elő.

Ezek az általános jellemzők nem csupán a GAM keretben működő változószelekciós algoritmusokra mondhatók el, hanem Hamon (2013), Phyu – Oo (2016) összefoglaló munkái alapján az általános, konkrét modelltől függetlenül működő változószelekciós algoritmusok esetén is hasonló tulajdonságok jellemzik az alkalmazott célfüggvényeket. Ugyanakkor, ezen algoritmusok esetében a magyarázóváltozók marginális hatásának meghatározása nem szempont, mivel az idézett tanulmányok szerint a legtöbbször eleve támaszvektor-gépekkel és k-legközelebbi szomszéd modellekkel alkalmazzák őket együtt, amelyek James et al. (2013) alapján az alacsony értelmezhetőségű modellek családjába tartoznak (1. ábra).

A kimondottan metaheurisztika alapú változószelekciós algoritmusok körében is a 4. fejezet (2) alakú célfüggvénye a legelterjedtebb, ahogy a 7. fejezet irodalomkutatása során tapasztaltak is mutatják, és a magyarázóváltozók közötti redundancia ezen algoritmusok körében sem jelenik meg, mint szempont. Ez ismét annak tudható be Agrawal et al. (2021) szintetizáló munkája alapján, hogy a legtöbb metaheurisztika alapú algoritmust is a James et al. (2013) alapján nem jól értelmezhető modellek (k-legközelebbi szomszéd, mélytanuló neurális hálózatok és támaszvektor-gépek) becslési pontosságának javítása céljából alkalmazzák. Ugyanakkor, Agrawal et al. (2021) rávilágít arra, hogy a szimplán becslési pontosságot maximalizáló algoritmusok körében is jól teljesítenek az olyan, több metaheurisztika elemeit ötvöző hibrid megoldások, mint a jelen értekezés tárgyát képező HGHK algoritmus.

A becslési pontosság maximalizálása során Hamon (2013), Phyu – Oo (2016) és Agrawal et al. (2021) numerikus eredményei alapján is azok az algoritmusok nyújtanak jellemzően jobb teljesítményt, amelyek jelen értekezés 9. és 10. fejezeteiben is kiemelkedően szerepeltek ebből a szempontból. Vagy a teljes változóhalmazt alkalmazó (szelekció nélküli) elsősorban

ensemble típusú modellek, mint a 10. fejezetben referenciaként használt LightGBM algoritmus (Yang – Zhang, 2018), vagy az olyan ensemble megoldások, amelyek a modellek paraméterbecslése során szimultán változószelekciót is végrehajtanak, mint a GAMBoost. Ezen ensemble eljárások közös vonása a saját kutatásunkban és a három idézett tanulmányban is, hogy jellemzően a teljes lehetséges magyarázóváltozó halmaz nagy részét a javasolt modellben tartják. Tehát, ezek a szakirodalomban népszerű modellek elsősorban előrejelző és nem magyarázó modellként funkcionálnak.

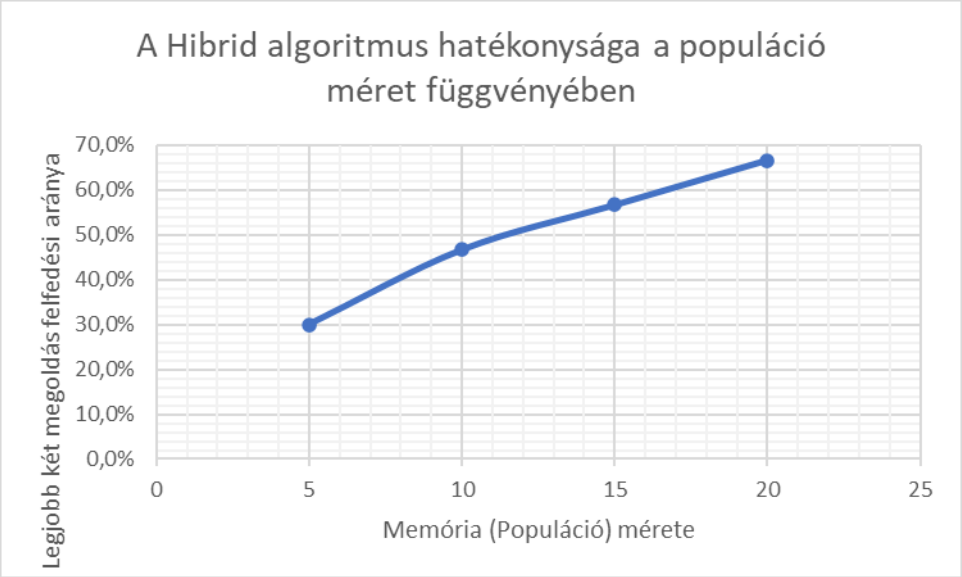
Az értekezésben elvégzett szakirodalmi áttekintés és diszkusszió alapján az általunk javasolt HGHK algoritmus az egyetlen metaheurisztikus megoldás a változószelekciós feladatra, ami GAM modellkeretben működik és korlátozó feltételként a végső modellben szereplő magyarázóváltozók redundancia mentességét többváltozós (és nem csupán páronkénti) összefüggéseket vizsgálva is biztosítja.

A HGHK algoritmus kapcsán GAM keretben az értekezés eredményei alapján elmondható, hogy ha a cél egy takarékos, magyarázó jellegű modell építése az eredményváltozóra és a futásidő sem komoly korlát, akkor a HGHK algoritmus alkalmazása javasolt az egyéb vizsgált változószelekciós algoritmusokkal szemben.

Melléklet

A 9.3. alfejezetben bemutatott feltételek mellett a HGHK algoritmusra elvégzett ceteris paribus elvű paraméter optimalizálás részletes eredményeit a betongerendák adatbázison jelen mellékletben ismertetjük.

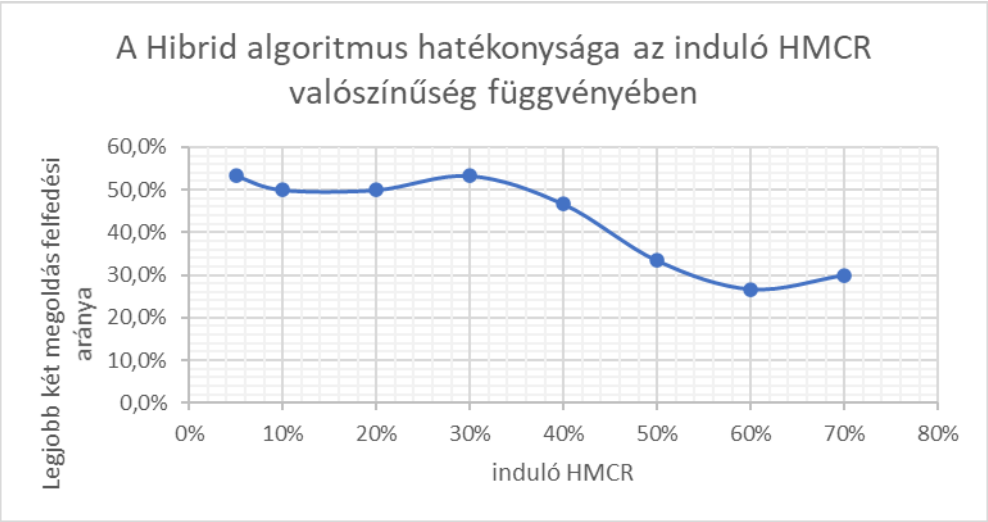
A Memória (Populáció) méret keresése:



Fontos megjegyezni, hogy a memória méret 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.

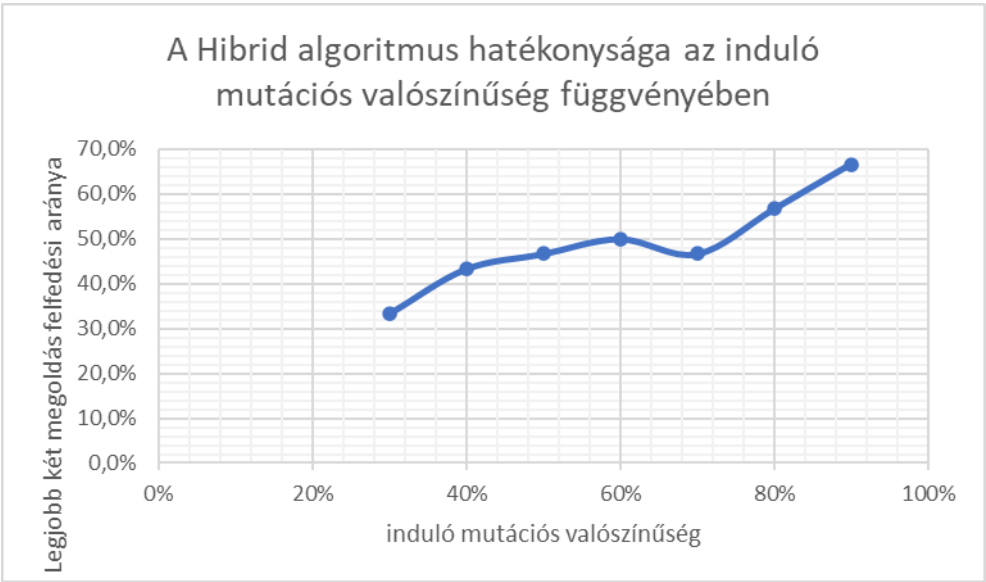
Változatlanul hagyott paraméterek: induló $HMCR$ = 0,2; maximális $HMCR$ = 0,6; induló mutációs valószínűség = 0,9; minimális mutációs valószínűség = 0,2; A korai kilépés változatlan legjobb célfüggvényre vonatkozó feltétele = 5

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

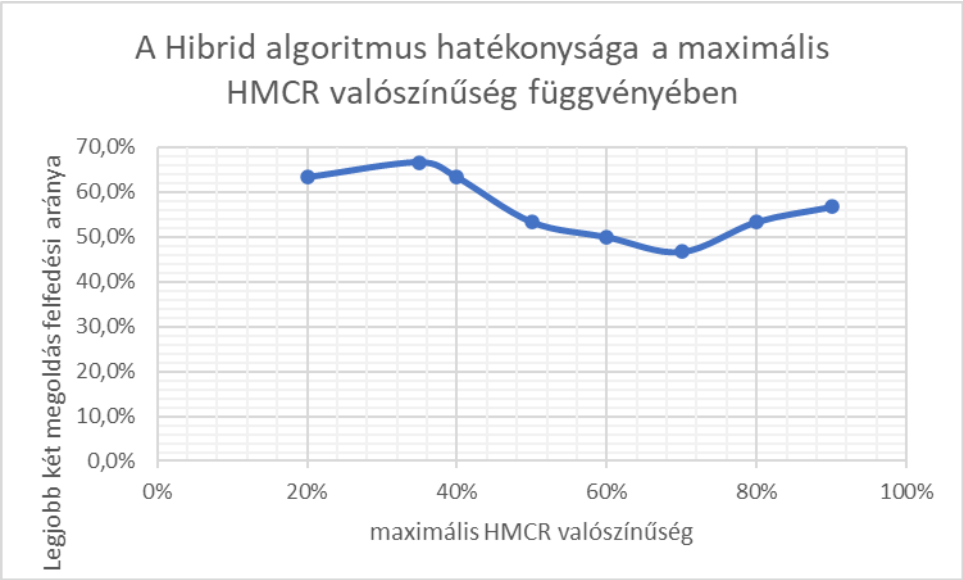
Az induló mutációs (bw) valószínűség keresése:



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

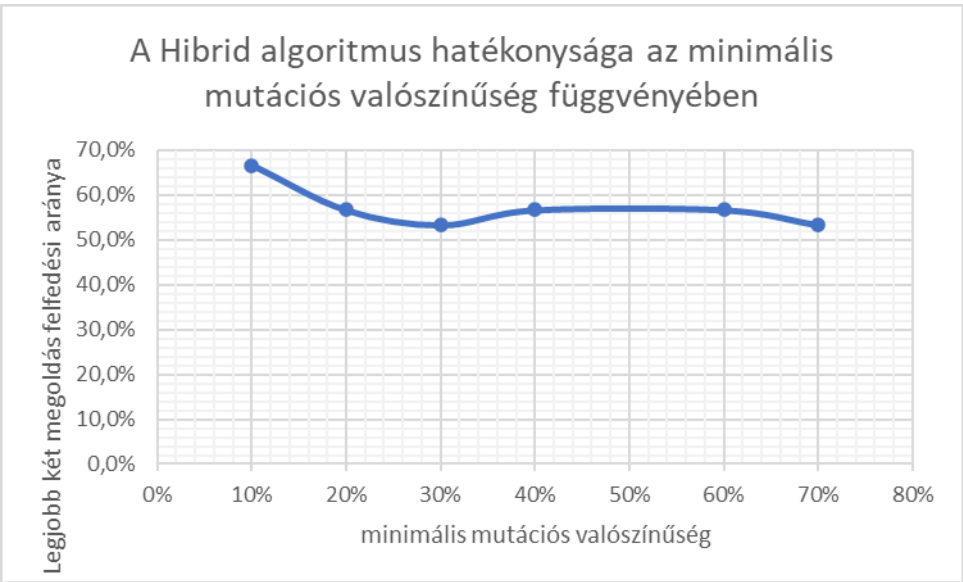
A maximális $HMCR$ valószínűség keresése:



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

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% felé emelkedni.

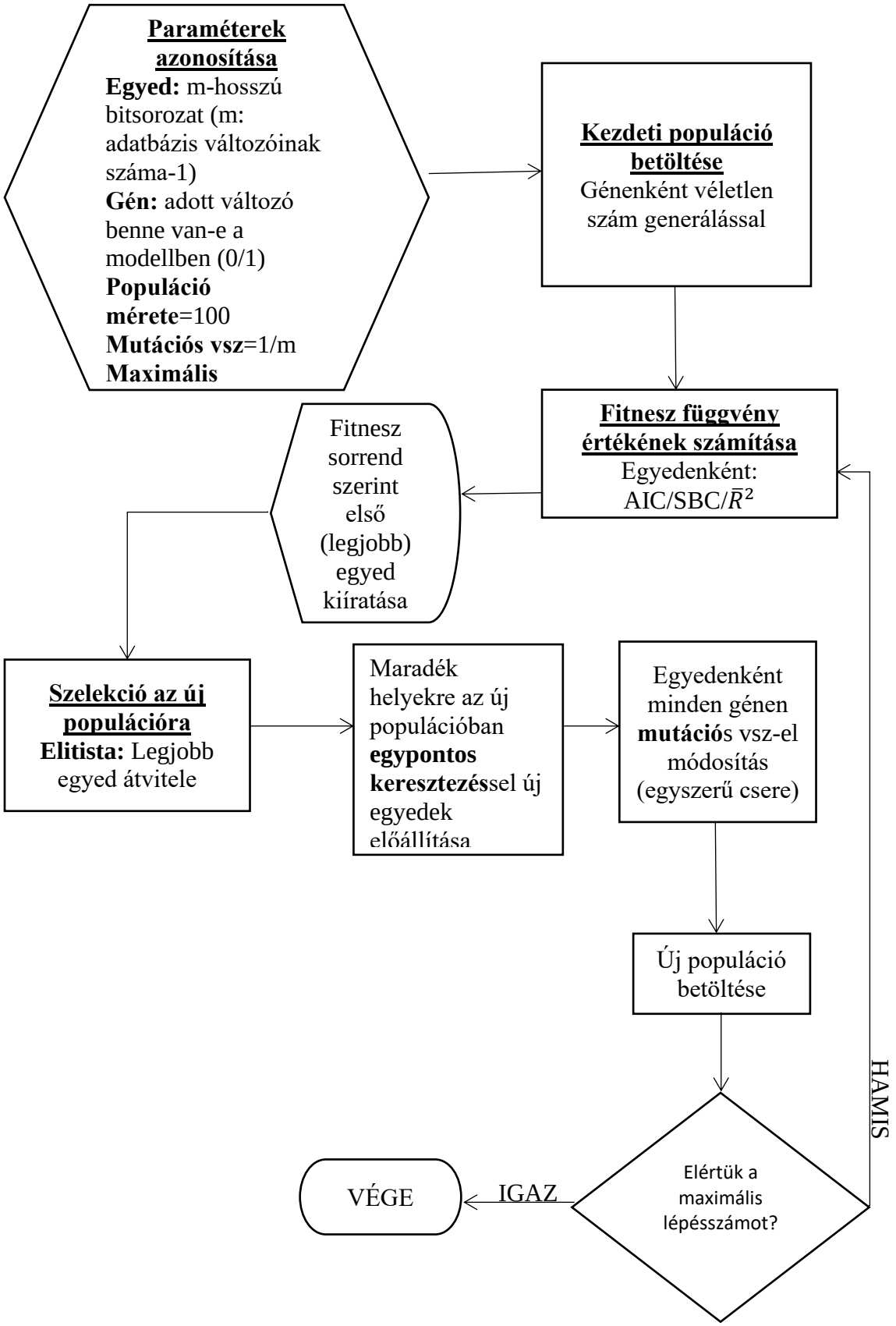
Változatlanul hagyott paraméterek: Memória mérete = 15; induló $HMCR$ = 0,2; maximális $HMCR$ = 0,6; induló mutációs valószínűség = 0,9; A korai kilépés változatlan legjobb célfüggvényre vonatkozó feltétele = 5

A 3. táblázatban adott optimális paraméter értékek mellett a globális optimum vagy a második legjobb megoldás megtalálásához szükséges lépésszámok eloszlása:

Mennyiség	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

A kiemelt gyakoriság értékek rávilágítanak, hogy az optimális paraméterek alkalmazása mellett a futtatások valamivel több, mint harmadában (12/30 esetben) úgy találja meg a HGHK algoritmus a globális optimumot vagy a második legjobb megoldást, hogy ez a megoldás már az algoritmus 5. iterációja előtt a memóriába került.

A rekonstruált genetikus algoritmus rekonstruált folyamatábrája, saját szerkesztés



Irodalomjegyzék

Abdel-Basset, M., El-Shahat, D., El-henawy, I., de Albuquerque, V. H. C., & Mirjalili, S. (2020). A new fusion of grey wolf optimizer algorithm with a two-phase mutation for feature selection. Expert Systems with Applications, 139, 112824.

Adler, D., Gläser, C., Nenadic, O., Oehlschlägel, J., & Zucchini, W. (2018). ff: Memory-Efficient Storage of Large Data on Disk and Fast Access Functions. R package version 2.2-14. https://CRAN.R-project.org/package=ff

Agrawal, P., Abutarboush, H. F., Ganesh, T., & Mohamed, A. W. (2021). Metaheuristic Algorithms on Feature Selection: A Survey of One Decade of Research (2009-2019). IEEE Access, 9, 26766-26791.

Ali, A. H. (2019). A survey on vertical and horizontal scaling platforms for big data analytics. International Journal of Integrated Engineering, 11(6), 138-150.

Amdahl, G. M. (1967, April). Validity of the single processor approach to achieving large scale computing capabilities. In Proceedings of the April 18-20, 1967, spring joint computer conference (pp. 483-485).

Aronszajn, N. (1950). Theory of reproducing kernels. Transactions of the American mathematical society, 68(3), 337-404.

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.

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.

Binder, H., & Tutz, G. (2008). A comparison of methods for the fitting of generalized additive models. Statistics and Computing, 18(1), 87-99.

Berkson, J. (1944). Application of the logistic function to bio-assay. Journal of the American statistical association, 39(227), 357-365.

Bliss, C. I. (1934). The method of probits. Science, 79, 38-39.

Bondi, A. B. (2000, September). Characteristics of scalability and their impact on performance. In Proceedings of the 2nd international workshop on Software and performance (pp. 195-203).

Borda, M. (2011). Fundamentals in information theory and coding. Springer Science & Business Media.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992, July). A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on Computational learning theory (pp. 144-152).

Breheny, P., & Huang, J. (2011) Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. Ann. Appl. Statist., 5: 232-253.

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). Classification and regression trees. CRC press.

Bryant, R. E., David Richard, O. H., & David Richard, O. H. (2016). Computer systems: a programmer's perspective. Third Edition. Upper Saddle River: Prentice Hall.

Buja, A., Hastie, T., & Tibshirani, R. (1989). Linear smoothers and additive models. The Annals of Statistics, 17(2), 453-510.

Cantoni, E., Flemming, J. M., & Ronchetti, E. (2011). Variable selection in additive models by non-negative garrote. Statistical modelling, 11(3), 237-252.

Calcagno, V. (2019). glmulti: Model Selection and Multimodel Inference Made Easy. R package version 1.0.7.1. https://CRAN.R-project.org/package=glmulti

Chong, I. G., & Jun, C. H. (2005). Performance of some variable selection methods when multicollinearity is present. Chemometrics and intelligent laboratory systems, 78(1-2), 103-112.

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.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.

Cox, D. R. (1972). Regression Models and Life-Tables. Journal of the Royal Statistical Society. Series B (Methodological) 34(2), 187-220.

Cox, D. R. (1975). Partial likelihood. Biometrika, 269-276.

Croxford, R. (2016): Restricted Cubic Spline Regression: A Brief Introduction; Institute for Clinical Evaluative Sciences.

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.

De Jong, P., & Heller, G. Z. (2008). Generalized linear models for insurance data. Cambridge Books.

Dowdy, S., Wearden, S., & Chilko, D. (2011). Statistics for research (Vol. 512). John Wiley & Sons.

Du, M., Liu, N., & Hu, X. (2019). Techniques for interpretable machine learning. Communications of the ACM, 63(1), 68-77.

Efron, B., Hastie, T., Johnstone, I., & Tibshirani, R. (2004). Least angle regression. The Annals of statistics, 32(2), 407-499.

Efroymson, M. A. (1960). Multiple regression analysis. Mathematical methods for digital computers, 191-203.

Etaati, L. (2019). Data Science Virtual Machine and AI Frameworks. In Machine Learning with Microsoft Technologies (pp. 273-285). Apress, Berkeley, CA.
Fahrmeir, L., Kneib, T., Lang, S., & Marx, B. (2013). Regression models. In Regression (pp. 21-72). Springer, Berlin, Heidelberg.
Fawagreh, K., Gaber, M. M., & Elyan, E. (2014). Random forests: from early developments to recent advancements. Systems Science & Control Engineering: An Open Access Journal, 2(1), 602-609.
Furnival, G. M., & Wilson, R. W. (1974). Regressions by leaps and bounds. Technometrics, 16(4), 499-511.
Gaddum, J. H. (1933). Reports on Biological Standards: Methods of Biological Assay Depending on a Quantal Response. HM Stationery Office.
Gauss, C. F. (1995). Theoria Combinationis Observationum Erroribus Minimis Obnoxiae, pars prior, 1821, translated by GW Stewart. Theory of the Combination of Observations Least Subject to Errors.
Gallager, R. G. (2013). Stochastic Processes Theory for Applications. Cambridge University Press.
Goldberg, D. E., (1989). Genetic Algorithms in Search, Optimization and Machine Learning. Boston, Kluwer Academic Publishers.
Green, P. J. and Silverman, B. W. (1994) Nonparametric Regression and Generalized Linear Models. London: Chapman and Hall.
Gretton, A., Bousquet, O., Smola, A., & Schölkopf, B. (2005). Measuring statistical dependence with Hilbert-Schmidt norms. In International conference on algorithmic learning theory (pp. 63-77). Springer, Berlin, Heidelberg.
Golub, G. H., & Van Loan, C. F. (1996). Matrix computations. Baltimore: Johns Hopkins University Press.
Gu, C. (1992). Diagnostics for nonparametric regression models with additive terms. Journal of the American Statistical Association, 87(420), 1051-1058.
Gu, C. (2013). Smoothing spline ANOVA models (Vol. 297). Springer Science & Business Media.
Gustafson, J. L. (1988). Reevaluating Amdahl's law. Communications of the ACM, 31(5), 532-533.
Hall, M. A. (1999). Correlation-based feature selection for machine learning. Doktori értekezés. University of Waikato.
Hamon, J. (2013). Optimisation combinatoire pour la sélection de variables en régression en grande dimension: Application en génétique animale. Doktori értekezés. Lille University of Science and Technology.
Hastie, T. (2018). gam: Generalized Additive Models. R package version 1.16. https://CRAN.R-project.org/package=gam
Hastie, T. J., & Tibshirani, R. J. (1990) Generalized Additive Models. London: Chapman and Hall.
Hastie, T. J., Tibshirani, R., & Friedman, J. (2011). The elements of statistical learning: Data mining, inference, and prediction (2nd ed.). NYC, NY: Springer.
Hazewinkel, M. (2001). Spline interpolation. Encyclopedia of mathematics, 1.
Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. Management Information Systems Quarterly, 28(1), 6.
Holland, J. H.,(1975). Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor.
Horváth, G. (1998). Neurális hálózatok és műszaki alkalmazásai. Budapest, Műegyetemi Kiadó.
Huo, X., & Ni, X. S.(2006). Regressions by Enhanced Leaps-and-Bounds via Additional Optimality Tests (LBOT). Georgia Institute of Technology.
Huo, X., & Ni, X. (2007). When do stepwise algorithms meet subset selection criteria?. The Annals of Statistics, 870-887.
Hutchinson, M. F. and de Hoog, F. R. (1985) Smoothing noisy data with spline functions. Numer. Math., 47, 99–106.
Ivakhnenko, A. G., & Lapa, V. G. (1967). Cybernetics and forecasting techniques. American Elsevier Publishing Company.
James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning: with applications in R. New York :Springer.
Jia, J., & Yu, B. (2010). On Model Selection Consistency of the Elastic Net. Statistica Sinica, 20, 595-611.
Jibitesh, M., & Ashok, M. (2011). Software Engineering. Pearson Education.
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.
Kane, M., J., Emerson, J., & Weston, S. (2013). Scalable Strategies for Computing with Massive Data. Journal of Statistical Software, 55(14), 1-19.
Kass, G. V. (1980). An exploratory technique for investigating large quantities of categorical data. Journal of the Royal Statistical Society: Series C (Applied Statistics), 29(2), 119-127.
Kenmeth, C., Landon, C., & Guercio, T. (2008). E-Commerce-Business, Technology. Pearson Prentice Hall/Pearson Education.
Kovács, E. (2006). Pénzügyi adatok statisztikai elemzése. Egyetemi Tankönyv, Budapesti Corvinus Egyetem Pénzügyi és Számviteli Intézet.

Kovács, L. (2019). Applications of Metaheuristics in Insurance. Society and Economy, 41(3), 371-395.
Kovács, P. (2008). A multikollinearitás vizsgálata lineáris regressziós modellekben. Statisztikai szemle 86(1), 38-67.
Krömer, P., & Platoš, J. (2016, July). Genetic algorithm for entropy-based feature subset selection. In 2016 IEEE Congress on Evolutionary Computation (CEC) (pp. 4486-4493). IEEE.
Kuechler, B., & Vaishnavi, V. (2008). On theory development in design science research: anatomy of a research project. European Journal of Information Systems, 17(5), 489-504.
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
Lang, S., Umlauf, N., Wechselberger, P., Harttgen, K., & Kneib, T. (2014). Multilevel structured additive regression. Statistics and Computing, 24(2), 223-238.
Láng, B., & Kovács, L. (2014). Linear regression model selection using improved harmony search algorithm. SEFBIS Journal, 9(1), 15-22.
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.
Lee, K. S., & Geem, Z. W. (2005). A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. Computer Methods in Applied Mechanics and Engineering, 194, 3902–3933.
Leng, C. and Zhang, H. H. (2006) "Model selection in nonparametric hazard regression", Nonparametric Statistics, 18, 417–429.
Li, Z., & S.N. Wood (2019) Faster model matrix crossproducts for large generalized linear models with discretized covariates. Statistics and Computing.
Lin, Y., & Zhang, H. H. (2006). Component selection and smoothing in multivariate nonparametric regression. The Annals of Statistics, 34(5), 2272-2297.
Lumley, T. (2013). biglm: bounded memory linear and generalized linear models. R package version 0.9-1. https://CRAN.R-project.org/package=biglm
Lumley, T. (2018). biglmm: Bounded Memory Linear and Generalized Linear Models. R package version 0.9-1. https://CRAN.R-project.org/package=biglmm
Ma, R., Miao, J., Niu, L., & Zhang, P. (2019). Transformed ℓ1 regularization for learning sparse deep neural networks. Neural Networks, 119, 286-298.
Mahdavi, M., Fesanghary, M., & Damangir, E. (2007). An improved harmony search algorithm for solving optimization problems. Applied Mathematics and Computation, 188, 1567-1579.
Mafarja, M. M., & Mirjalili, S. (2017). Hybrid whale optimization algorithm with simulated annealing for feature selection. Neurocomputing, 260, 302-312.
Marra, G., & Wood, S. N. (2011). Practical variable selection for generalized additive models. Computational Statistics & Data Analysis, 55(7), 2372-2387.
McFadden, D. (1974). Conditional logit analysis of qualitative choice behaviour, in: P. Zarembka (ed.), Frontiers in Econometrics, Academic Press, New York, 105-142.
Michael, M., Moreira, J. E., Shiloach, D., & Wisniewski, R. W. (2007, March). Scale-up x scale-out: A case study using nutch/lucene. In 2007 IEEE International Parallel and Distributed Processing Symposium (pp. 1-8).
Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. Advances in engineering software, 69, 46-61.
Mirjalili, S. (2016). Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Computing and Applications, 27(4), 1053-1073.
Molnar, C. (2020). Interpretable machine learning. Leanpub.
Moula, F. E., Guotai, C., & Abedin, M. Z. (2017). Credit default prediction modeling: an application of support vector machine. Risk Management, 19(2), 158-187.
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.
Murphy, K. P. (2012). Machine learning: a probabilistic perspective. MIT press.
Natekin, A., & Knoll, A. (2013). Gradient boosting machines, a tutorial. Frontiers in neurorobotics, 7, 21.
Nelder, J. A., & Wedderburn, R. W. (1972). Generalized linear models. Journal of the Royal Statistical Society: Series A (General), 135(3), 370-384.
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.
Phyu, T. Z., & Oo, N. N. (2016). Performance comparison of feature selection methods. In MATEC web of conferences (Vol. 42, p. 06002). EDP Sciences.
Plackett, R. L. (1949). A historical note on the method of least squares. Biometrika, 36(3/4), 458-460.

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.
Powers, D. M. W. (2011). Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation. Journal of Machine Learning Technologies, 2(1), 37–63.
Quinlan, J. R. (1986). Induction of decision trees. Machine learning, 1(1), 81-106.
Quinlan, J. R. (1993). C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers.
Ramanathan, R. (1992). Introductory to econometrics with applications (Vol. 4). Dryden Press.
Raschka, S., & Mirjalili, V. (2017). Python Machine Learning: Machine Learning and Deep Learning with Python. Scikit-Learn, and TensorFlow. Second edition ed. Birmingham: Packt Publishing, Limited.
Reed, R., & MarksII, R. J. (1999). Neural smithing: supervised learning in feedforward artificial neural networks. Mit Press.
Ridzuan, F., & Zainon, W. M. N. W. (2019). A review on data cleansing methods for big data. Procedia Computer Science, 161, 731-738.
Rodgers, D. P. (1985). Improvements in multiprocessor system design. ACM SIGARCH Computer Architecture News, 13(3), 225-231.
Romanski, P., & Kotthoff, L. (2018). FSelector. R package version 0.31. https://CRAN.R-project.org/package=FSelector
Russell, S., & Norvig, P. (2010). Artificial intelligence: a modern approach. Third Edition. Prentice Hall. ISBN 9780136042594.
Sayed, G. I., Tharwat, A., & Hassanien, A. E. (2019). Chaotic dragonfly algorithm: an improved metaheuristic algorithm for feature selection. Applied Intelligence, 49(1), 188-205.
Schapire, R. E. (1990). The strength of weak learnability. Machine Learning 5, 197–227.
Schmid, M., & Hothorn, T. (2008). Boosting additive models using component-wise P-splines. Computational Statistics & Data Analysis, 53(2), 298-311.
Schumaker, L. L. (2015). Spline functions: computational methods. Society for Industrial and Applied Mathematics.
Seligman, M., Fraley, C., & Hesterberg, T. (2011). biglars: Scalable Least-Angle Regression and Lasso. R package version 1.0.2. https://CRAN.R-project.org/package=biglars
Shcherbakov, M. V., Brebels, A., Shcherbakova, N. L., Tyukov, A. P., Janovsky, T. A., & Kamaev, V. A. E. (2013). A survey of forecast error measures. World Applied Sciences Journal, 24(24), 171-176.
Signoretto, M., Pelckmans, K., & Suykens, J. A. (2008). Functional ANOVA Models: Convex-concave approach and concurvity analysis (No. 08-203). Internal Report.
Song, L., Smola, A., Gretton, A., Bedo, J., & Borgwardt, K. (2012). Feature selection via dependence maximization. Journal of Machine Learning Research, 13(5), 1393-1434.
Steinberg, D., & Colla, P. (2009). CART: classification and regression trees. The top ten algorithms in data mining, 9, 179.
Storlie, C. B., Bondell, H. D., Reich, B. J. and Zhang, H. H. (2011) Surface Estimation, Variable Selection, and the Nonparametric Oracle Property. Statistica Sinica, 21, 679–705.
Tangian, A., & Gruber, J. (2002). Constructing and Applying Objective Functions. In Proceedings of the Fourth International Conference on Econometric Decision Models: Constructing and Applying Objective Functions. University of Hagen, Held in Haus Nordhelle, August (Vol. 2831, p. 2000).
Therneau, T., & Atkinson, B. (2018). rpart: Recursive Partitioning and Regression Trees. R package version 4.1-13. https://CRAN.R-project.org/package=rpart
Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B (Methodological), 58(1), 267-288.
Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., & Knight, K. (2005). Sparsity and smoothness via the fused lasso. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 67(1), 91-108.
Tutz, G., & Binder, H. (2006). Generalized additive modeling with implicit variable selection by likelihood-based boosting. Biometrics, 62(4), 961-971.
Umlauf, N., Adler, D., Kneib, T., Lang, S., & Zeileis, A. (2015). Structured Additive Regression Models: An R Interface to BayesX. Journal of Statistical Software, 63(21), 1-46. URL http://www.jstatsoft.org/v63/i21/.
Venables, W. N. & Ripley, B. D. (2002) Modern Applied Statistics with S. Fourth Edition. Springer, New York. ISBN 0-387-95457-0
Wahba, G. (1990) Spline models for observational data. CBMS–NSF Regl Conf. Ser. Appl. Math., 59.
Walkowiak, S. (2016). Big Data Analytics with R. Packt Publishing Ltd.
Wang, Y., Liu, Y., Feng, L., & Zhu, X. (2015). Novel feature selection method based on harmony search for email classification. Knowledge-Based Systems, 73, 311-323.
Weiss, G. M. (2004). Mining with rarity: a unifying framework. ACM Sigkdd Explorations Newsletter, 6(1), 7-19.

Werbos, P. J. (1994). The roots of backpropagation: from ordered derivatives to neural networks and political forecasting (Vol. 1). John Wiley & Sons.
Weston, S. (2019a). foreach: Provides Foreach Looping Construct. R package version 1.4.7. https://CRAN.R-project.org/package=foreach
Weston, S. (2019b). doParallel: Foreach Parallel Adaptor for the 'parallel' Package. R package version 1.0.15. https://CRAN.R-project.org/package=doParallel
Wieringa, R. J. (2014) Design science methodology for information systems and software engineering. Springer.
Williams, C. K. I. (2021). The Effect of Class Imbalance on Precision-Recall Curves. Neural Computation, 33 (4), 853–857.
Wooldridge, J. M. (2016). Introductory econometrics: A modern approach. Nelson Education.
Wood, S. N. (2003). Thin plate regression splines. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 65(1), 95-114.
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.
Wood, S. N. (2017) Generalized Additive Models: An Introduction with R (2nd edition). Chapman and Hall/CRC.
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.
Xie, H., & Huang, J. (2009). SCAD-penalized regression in high-dimensional partially linear models. The Annals of Statistics, 37(2), 673-696.
Xue, X., Kim, M. Y., & Shore, R. E. (2007). Cox regression analysis in presence of collinearity: an application to assessment of health risks associated with occupational radiation exposure. Lifetime data analysis, 13(3), 333-350.
Yamada, M., Jitkrittum, W., Sigal, L., Xing, E. P., & Sugiyama, M. (2014). High-dimensional feature selection by feature-wise kernelized lasso. Neural computation, 26(1), 185-207.
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.
Yang, S., & Zhang, H. (2018). Comparison of several data mining methods in credit card default prediction. Intelligent Information Management, 10(05), 115.
Yeh, I. C. (1998). Modeling of strength of high-performance concrete using artificial neural networks. Cement and Concrete research, 28(12), 1797-1808.
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.
Yuan, M., & Lin, Y. (2006). Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 68(1), 49-67.
Yusta, S. C. (2009). Different metaheuristic strategies to solve the feature selection problem. Pattern Recognition Letters, 30(5), 525-534.
Zhang, H. H., & Lin, C. Y. (2006) Component Selection and Smoothing for Noparametric Regression in Exponential Families. Statistica Sinica, 16, 1021–1041.
Zhang, H. H., & Lin, C. Y. (2013). cosso: Fit Regularized Nonparametric Regression Models Using COSSO Penalty.. R package version 2.1-1. https://CRAN.R-project.org/package=cosso
Zhao, P., & Yu, B. (2006). On model selection consistency of Lasso. The Journal of Machine Learning Research, 7, 2541-2563.
Zhou, S., & Shen, X. (2001). Spatially adaptive regression splines and accurate knot selection schemes. Journal of the American Statistical Association, 96(453), 247-259.
Zomaya, A. Y. (2021). Advanced computer architecture and parallel processing. John Wiley & Sons, Hoboken, New Jersey.
Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 67(2), 301-320.