GÉPI TANULÁSI MÓDSZEREK LEHETSÉGES BIZTOSÍTÁSI ALKALMAZÁSA
KERESZTÉRTÉKESÍTÉSI LEHETŐSÉGEK FELTÉRKÉPEZÉSE

Csőke Marcell (BCE-ELTE Biztosítási és pénzügyi matematika MSc), csoke.marcell@gmail.com

ÖSSZEFOGLALÓ

Jelen tanulmány célja olyan értékesítést támogató modellek létrehozása, amelyekkel feltárhatjuk azon szegmenseket egy biztosító meglévő kötelező gépjármű-felelősségbiztosítás állományából, amelyekre nagyobb valószínűséggel történhet sikeres casco-keresztértékesítés. Az így kapott modellek valamelyikének gyakorlatban való alkalmazása lehetővé teszi, hogy hatékonyan feldolgozható címanyagot állítsunk elő a biztosító egy értékesítési csatornája számára. A modellezés során négy modelltípus kerül illesztésre: logisztikus regresszió, egy döntésifa-alapú ensemble-modell, támaszvektor-gép és neurális hálózat. Értékelésük a pontosság mellett az időtényezőt is figyelembe véve történik. Végül a kapott modelleredmények értelmezésén túl a gyakorlati felhasználhatóságukra és továbbfejlesztési lehetőségeikre is kitérek. A publikáció alapját a BCE-ELTE Biztosítási és pénzügyi matematika mesterszakon írt szakdolgozatom jelenti.

SUMMARY

The main purpose of this paper is to develop sales support models which are capable of identifying cross-selling opportunities in an insurance company's existing motor third party liability insurance portfolio for a Casco product. With the practical application of these models it is possible to generate efficiently processable leads for a sales channel of the insurance company. During the modeling four types of models are fitted: logistic regression, a decision tree based ensemble model, a support vector machine and a neural network. Besides the accuracy, the time factor of fitting the models is considered as well during the evaluation. Finally beyond the interpretation of the model results I will also discuss their practical usability and further development possibilities. The publication is based on my thesis written in the Actuarial and Financial Mathematics Master's program of Corvinus University of Budapest and Eötvös Loránd University.

Kulcsszavak: gépi tanulás, keresztértékesítés, címanyag-előállítás, biztosítás
Key words: machine learning, cross-selling, leadset generation, insurance

JEL: C45, C52, C53, G22

DOI: 10.18530/BK.2023.1-2.52
http://dx.doi.org/10.18530/BK.2023.1-2.52

Bevezetés

Az adatbányászat és ezzel a működéshatékonyság növelése évek óta a legaktuálisabb témája a hazai pénzügyi szektornak, relevanciája megkérdőjelezhetetlen. Ennek ellenére meglepően tapasztalható – ahogy Szabó (2015) is felhívja rá a figyelmet –, hogy a biztosítási szektor nemcsak a nemzetközi biztosítási piachoz képest, de hazai, iparágak közötti összevetésben is a 2010-es éveket némi lemaradással kezdte meg az adatbányászati eszközök elterjedtségének tekintetében. Ennek okai természetesen szorosan kapcsolódnak a biztosítási szektor sajátosságaihoz, más pénzügyi termékekkel való összevetésben, egy biztosítási termék esetében kevesebb tranzakció jön létre. A telekommunikációs szektorral kapcsolatban pedig nem túlzás kijelenteni, hogy egy-egy szerződéshez kapcsolódóan naponta több tranzakció generálódik, mint egy biztosítási szerződés kapcsán egy egész naptári évben. A nagyfokú digitalizáció előtt, a papíralapú szerződések korában ráadásul a változó adatminőség is akadályokat gördített az adatbányászat széles körű biztosítói alkalmazása elé. Továbbá a biztosítási üzem kockázatközösségekre épülő (és a kockázatokat ebben a közösségben szétporlasztó) szerkezetében szinte természetes a biztosítók (és értékesítőik) új ügyfélszerzésre, és nem a meglévő ügyfélkapcsolatok elmélyítésére koncentráló magatartása.

A téma aktualitását Horváth és Paulovits (2016) elemzése is jól szemlélteti, a szerzőpáros cikke alapján a biztosítással még nem rendelkező társadalmi szegmensek feltárása mellett (amely szintén komoly kihívásokat rejtő adatbányászati feladat) a keresztértékesítési potenciálok kiaknázása rejt további növekedési lehetőségeket a következő években a biztosítók számára.

Dolgozatom célja, hogy egy biztosító ügyfelei közül adatbányászati módszerekkel minél pontosabban feltérképezzem azokat, akik jó eséllyel nyitottak lehetnek egy másik biztosítási termékre vonatkozó szerződés megkötésére. Jelen dolgozat fókuszában két termék áll: a biztosító kötelező gépjármű-felelősségbiztosítás állományából a casco termékre való sikeres keresztértékesítést modellezem és elemzem, ennek megfelelően a modellezési feladat bináris klasszifikációs modellek illesztése lesz.

Az alaphalmazt egy hazai biztosító kötelező gépjármű-felelősségbiztosítás (továbbiakban röviden: kgfb) szerződéseinek azon része alkotja, amelyeket természetes személyek kötöttek személygépjárműre egy meghatározott időszakban a függő ügynöki értékesítési csatornán keresztül. Az elemzés során azt tekintem pozitív (azaz prediktálandó) esetnek, amikor a

kgfb-szerződés után (de legkorábban egyszerre) került megkötésre egy casco-szerződés is az adott gépjárműre. Azon megfigyelések, amelyeknél hamarabb jött létre casco, mint kgfb, nem kerültek bele az alaphalmazba.

A biztosító kötelező gépjármű-felelősségbiztosítás állományából a casco termékre való sikeres keresztértékesítést modellezem.

Dolgozatomban arra a kérdésre keresem a választ, hogy megalkotható-e egy olyan értékesítéstámogató-modell, amely az egyszerű, üzleti szabályok használatával előállítható, értékesítési hálózat által feldolgozandó címanyagoknál gazdaságosabb. A modellezés során négy modelltípust illesztek: logisztikus regressziót, egy döntési fán alapuló ensemble-modellt (ún. Gradient Boosting Machine, röviden GBM), támaszvektor-gépet (SVM) és előrecsatolt neurális hálózatot (FNN), majd ezek eredményeit hasonlítom össze és értékelem.

Bináris klasszifikáció

Mindenekelőtt érdemes röviden áttekinteni a különböző típusú bináris klasszifikációs modellek illesztéséhez és értékeléséhez használt keretrendszert. Az alaphalmazból előzetesen véletlen kiválasztással elkülönítésre került a teszt és a tanító halmaz (10-90% arányban), amelyekben a pozitív megfigyelések (sikeres keresztértékesítések) aránya megegyezik, ez hozzávetőlegesen 32%. A modellek optimális hiperparamétereinek megkeresése többszörös keresztvalidációval történik, ennek során a tanító halmaz további szétválasztásra kerül tanító és validációs halmazra, 80-20% arányban.

A klasszifikáció pontosságának mérésére, valamint a modelltípusok egymással való összehasonlítására alapvetően két mutatót használok (az előre elkülönített teszt-adathalmazon mérve), a találati arányokat tartalmazó mátrixot (klasszifikációs táblát), valamint a ROC-görbe alatti területet, vagyis az AUC mutatót. Alapvetően a ROC-görbe is a találati arányra épül: a hamis pozitív arányok[1] függvényében ábrázoljuk az igaz pozitív arányokat[2] . Ennek görbe alatti területe a modell pontosságát reprezentálja, melynek elméleti maximuma 1. Ebben az esetben a modell tökéletesen képes szeparálni a két csoportot, míg, ha az AUC értéke 0,5, akkor klasszifikációra alkalmatlan a modell, teljesítménye megegyezik a teljesen véletlenszerű besorolásával.

Fontos eleme a bináris klasszifikációs modelleknek az ún. küszöbérték (cut-off paraméter), megfelelő megválasztása különösen kulcsfontosságú, ha a pozitív mintaelemek aránya alacsony a mintán belül. Jelen adathalmaznál ugyan nem áll fenn ilyen extrém mértékű eset, mindenesetre érdemes a cut-off paraméter optimális értékét meghatározni, mivel még jelen körülmények között is számunkra kedvező irányba módosítja a predikciókat.

Az optimális cut-off paramétert meghatározó egyik módszer az ún. Youden-féle *J*-mutatón alapul, amely a klasszifikáció jóságának egy mutatószáma (Youden, 1950). A *J* mutató kiszámítása szorosan kapcsolódik a ROC görbéhez: ennek meghatározásához is a különböző cut-off paraméterek mentén előálló szenzitivitás és specificitás értékekre van szükség. A következőkben a klasszifikációs mátrix szokásos elemeire az egyszerűség kedvéért a bevett angol rövidítéseikkel hivatkozom (például true positives = *TP*):

$$sensitivity = \frac{TP}{TP+FN} \tag{1}$$

$$specificity = \frac{TN}{TN+FP} \tag{2}$$

Ezek segítségével a *J* mutató:

$$J=sensitivity+specificity-1 \tag{3}$$

A (3) összefüggésből némi átalakítás után megkaphatjuk, hogy

$$J=TPR-FPR \tag{4}$$

ahol a TPR az igaz pozitívok aránya (sensitivity), az FPR pedig a hamis pozitívok aránya (1-specificity). Az optimális cut-off (vagy más néven threshold) paramétert pedig az jelenti számunkra, amelynél a *J*-mutató értéke maximális, tehát:

$$threshold^* = \underset{threshold}{\operatorname{argmax}}\, J(threshold) \tag{5}$$

A módszer alkalmazása során a klasszifikációs táblában érdemes nyomon követni az első- és másodfajú hibákat, és az alapján dönteni az alkalmazásáról, hogy a modellezendő probléma szempontjából kedvezően alakulnak-e ezen hibák.

Minden elemzést és modellezést a Python[3] nyílt forráskódú fejlesztői környezetben valósítottam meg. A logisztikus regresszió illesztését a „statsmodels", a GBM és az SVM modellek illesztését a „scikit-learn", míg az FNN modellt a „TensorFlow 2.0" nevű könyvtárak használatával készítettem.

Adathalmaz előállítása, előkészítése

A bevezetőben már definiálásra került, hogy az elemzés célja egy olyan értékesítéstámogató modell megalkotása, amely alkalmas a biztosító meglévő kgfb-állományában meghatározni azokat az ügyfeleket, akiket érdemes megkeresni a cég casco termékével. A modellel szembeni elsődleges elvárás, hogy tartósan minőségi címanyagot lehessen vele előállítani, amelyet a biztosító értékesítési csatornái (jelen esetben a függő ügynökei) feldolgozhatnak.

A modellek illesztéséhez az adatok elsődleges forrása a biztosító adatbázisa. Az adatbásból származó adatok részletes (leíró statisztikáig terjedő) bemutatása az adatok bizalmas jellege miatt nem lehetséges, de a következőkben a lehetőségekhez mérten igyekszem egy általános képet nyújtani a változók köréről.

A tanítóhalmazt a biztosító függő ügynökein keresztül kötött azon kgfb-szerződések alkotják, amelyek 2018 és 2021 között kerültek megkötésre. Ezt a halmazt üzleti megfontolások alapján leszűrtem természetes személyek olyan szerződéseire, ahol a biztosított gépjárműve személygépjármű, illetve további szűréseket kellett elvégezni a cég casco termékével összhangban a gépjármű korára és használati jellegére vonatkozóan. Pozitív mintaelemnek azok a szerződéseket tekintem, amelyekre a kgfb-szerződés megkötése után (vagy esetleg vele egyszerre) megkötésre került casco-szerződés is, miközben a kgfb-szerződés még élő.

A tanítóhalmazt azon kgfb-szerződések alkotják, amelyek 2018 és 2021 között kerültek megkötésre.

Az adatok előállítása az adatbázisból közvetlenül SQL-lekérdezésekkel történt, a cél pedig egy olyan adathalmaz előállítása volt, amely lényegében minden elérhető információt tartalmaz. A lekérdezett mezők alapvetően két csoportra bonthatók, egyrészt magára a kgfb-szerződésre vonatkozó adatokra (például a biztosítás díja, a gépjármű értéke vagy a díjfizetési mód és gyakoriság), másrészt pedig ügyféladatokra (például ügyfél neme, kora és lakhelye). Fontos kiemelni, hogy az előállított halmaz szerződésekre vonatkozóan egyedi, tehát ugyanaz az ügyfél többször is bekerülhetett, ha a vizsgált időszakban több megkötött kgfb-szerződése is volt az érintett értékesítési csatornán keresztül. Illetve a halmazban nem csak jelenleg élő szerződések szerepelnek (előfordulhat tehát olyan eset, hogy egy szerződés pozitív mintaelem, azonban már sem a kgfb-, sem a casco-szerződés nem élő, például, ha a gépjármű eladása miatt a szerződések érdekmúlással megszűntek). Az adatok egy része már azonnal a kellő formában a rendelkezésemre állt, másik részüket pedig még a lekérdezések folyamatában kellett származtatni az eredetileg tárolt adatok módosításával, aggregálásával. Ilyen származtatott mezők alatt elsősorban az ügyfelek különböző biztosítási statisztikáit kell érteni, mind a szóban forgó kgfb-, mind az egyéb nemélet és életbiztosítási szerződéseikre vonatkozóan.

Összesen 73 magyarázóból állt végül össze a tanítóhalmaz, amelyből nagyjából 45 numerikus változó, a fennmaradók pedig kategorikusak. A fentiek szerint kialakított halmazból a dolgozatomhoz egy véletlenszerűen kiválasztott részhalmazt használtam fel, amely megközelítőleg 11 ezer darab szerződésből áll. Ezek közül pedig nagyjából 3500 eset jelent pozitív mintaelemet. Ezt az alaphalmazt bontottam tehát két részre, tanító és teszt halmazokra.

Alapvetően a biztosító adatbázisában az adatminőség kiválónak mondható, összetett adattisztítási eljárásra egyetlen mező kivételével nem volt szükség. A kivételt képező mező a gépjármű értékének a meghatározása, ugyanis kgfb-szerződések kapcsán nincs eltárolva a gépjármű Eurotax-kódja. Így a lekérdezéshez egy olyan gépjárműértéket felbecsülő algoritmust kellett alkalmazni, amely a gépjármű néhány (kgfb-szerződésben már rögzített) paraméteréből karakteregyezőség alapján megállapítja a szóba jöhető Eurotax-kódok halmazát. E kódokhoz tartozó gépjárműárak átlaga lesz számunkra a becsült gépjárműérték. Illetve a lekérdezések legelején az adatminőség garantálása miatt kizárásra kerültek olyan kgfb-szerződések, amelyek valójában egy napot sem voltak élők (például adminisztrációs hiba miatt), de ezek száma szerencsére nagyon csekély volt.

Az adatok előkészítése kapcsán az egyik legfontosabb feladat a pénzben kifejezett értékek kezelése. Ehhez minden ilyen változót először idősorosan, időbélyeggel ellátva kérdeztem le. Ezt követően pedig jövőérték-számítással azonos időpontra transzformáltam az összegeket. A jövőérték-számításhoz használt szorzótényezőket változónként határoztam meg, nyilvánosan elérhető információkból. Például a szerződések kötéskori éves díját a magyar piaci átlagos árváltozással korrigáltam (1. táblázat).

1. táblázat: Átlagos kgfb-díjak és változásaik a magyar biztosítási piacon

Dátum	Átlagdíj (Ft)	Átlagos díjváltozás
2018.01.01	33 908	-
2019.01.01	39 791	17,35%
2020.01.01	47 296	18,86%
2021.01.01	49 109	3,83%
2022.01.01	44 754	-8,87%

Forrás: saját szerkesztés a Netrisk adatai alapján[5]

A szerződésekkel kapcsolatos kárkifizetések jövőértékét pedig a járműjavítási díjak változásai alapján számoltam ki (2. táblázat).

2. táblázat: Járműjavítások árváltozásai

Év	Átlagos árváltozás
2018	4,90%
2019	6,00%
2020	8,60%
2021	9,60%

Forrás: saját szerkesztés a KSH adatai alapján[6]

A jövőértéket minden pénzösszeg esetén 2021 végére határoztam meg, a számításokat pedig kamatos kamattal végeztem, azaz:

(6) $$FV(C, \boldsymbol{i}, T, \tau) = C \cdot (1 + i_0)^{\tau} \cdot \prod_{t=1}^{T}(1 + i_t)$$

ahol C a felmerült pénzösszeg (például kárkifizetés vagy befolyt díj), $\boldsymbol{i}$ az adott változóhoz tartozó korrekciós tényezőket tartalmazó vektor, T a felmerülés évének vége és 2021 között eltelt évek száma, míg τ a felmerülés évében az értéknaptól számított hátralévő idő aránya az adott év végéig a teljes évhez viszonyítva.

Az így összeállt adatbázis numerikus változóin *outlier*-elemzést végeztem el. A kiugró értékek azonosításához a Tukey-féle kvartilis módszert alkalmaztam (Tukey, 1977), amely az interkvartilis terjedelem és egy előre meghatározott k konstans alapján jelöli meg a kiugró megfigyeléseket:

(7) $$[Q_1\text{-}k(Q_3\text{-}Q_1),\ Q_3\text{+}k(Q_3\text{-}Q_1)]$$

A (7)-ben definiált intervallumon kívül eső megfigyelések számítanak kiugró értéknek. Tipikusan két k értékre is meg szokás vizsgálni az outlier-eket, 1,5-re és 3-ra. Utóbbi esetében a (7) intervallumon kívül eső megfigyeléseket távoli kiugró értéknek hívjuk.

A kiugró értékek elemzése közben az alaphalmazból a várakozásoknak megfelelően nem kellett sok megfigyelést kiszűrni, egyrészt néhány (valószínűleg autókereskedelemmel foglalkozó) egyéni vállalkozó került kiszűrésre, illetve néhány olyan gépjármű is, amely ugyan valamilyen oknál fogva személygépjárműként lett levizsgáztatva az adatbázis szerint, de bruttó tömege jócskán meghaladta a 3,5 tonnát.

Kategorikus változók esetében változónként megvizsgáltam a megfigyelések kategóriák közötti megoszlását azzal a céllal, hogy ahol az lehetséges és értelmes, az alacsony elemszámú kategóriákat összevonjam, végül egyetlen változónál került erre sor, a kgfb-szerződéskötés okánál.

A modellezés megkezdése előtt még a változók közötti összefüggőség elemzése következett, ez kiemelten fontos, hiszen eddig semmilyen szűrés nem történt a magyarázó változók szerepeltetésére vonatkozóan, csupán összeszedtem a lehető legszélesebb adatkört. Az elemzés két részre bomlott, egyrészt vizsgáltam, hogy van-e olyan magyarázó változó, amely a függő változóval túlságosan összefüggő lenne, másrészt a magyarázó változók között fennálló esetleges multikollinearitást elemeztem. Ebben a lépésben elsősorban az utóbbi, túlzott multikollinearitás miatt néhány magyarázó változó elhagyása mellett döntöttem. Ezek szinte kivétel nélkül azok a változók voltak, amelyeket a gépjármű értékét becsülő algoritmusnál felhasználtam a szóba jöhető Eurotax kódok megállapításához, az összefüggő párok közül maga a gépjármű értéke került végül megtartásra.

Logisztikus regresszió

Az alábbiakban a bináris logisztikus regressziót elsősorban Kovács (2014) alapján foglalom össze. A modell célja, hogy független változók lineáris kombinációjának segítségével leírjuk a csoportba tartozás valószínűségének logit transzformáltját, azaz

$$\ln\left(\frac{p}{1-p}\right) = logit(p) = \sum_{i=0}^{n} \beta_i x_i \qquad (8)$$

Amiből adódik

$$p = \frac{e^{\boldsymbol{\beta}^T x}}{1+e^{\boldsymbol{\beta}^T x}} \qquad (9)$$

A regressziós paraméterek becslése maximum likelihood módszerrel történik, ahol a likelihood függvény a Binom*(1,p)* (azaz Bernoulli) eloszlású változóra

$$L(p) = \prod_{i=1}^{n} p^{y_i}(1-p)^{(1-y_i)} \qquad (10)$$

A fentibe behelyettesítve (9)-et pedig

$$L(\beta_0, \beta_1, \dots, \beta_p) = \prod_{i=1}^{n} \left[\frac{\exp\left(\sum_{j=1}^{p} \beta_j x_{ij}\right)}{1+\exp\left(\sum_{j=1}^{p} \beta_j x_{ij}\right)}\right]^{y_i} \cdot \left[\frac{1}{1+\exp\left(\sum_{j=1}^{p} \beta_j x_{ij}\right)}\right]^{1-y_i} \qquad (11)$$

A β paraméterek meghatározására nincs zárt formula, az $L(\beta)$ függvény maximum-helyének kiszámítása numerikus módszerekkel történhet, például *Newton-Raphson* eljárással. Az illesztett modell becsült paramétereinek exponenciális transzformáltjai értelmezhetőek is, méghozzá: ceteris paribus x_i egységnyi növekedése hányszorosára változtatja meg a $\frac{p}{1-p}$-t (a p valószínűséghez tartozó *odds*-ot).

A logisztikus regressziós modell illesztése előtt minimális adattranszformációra volt még szükség: egyrészt a pénzben kifejezett változóknak a természetes alapú logaritmusát képeztem, másrészt pedig a kategorikus változókat dummy változókká alakítottam, változónként egy-egy kategória elhagyásával. Az 1. ábrán az összesítő egy részlete látható.

1. ábra: Részlet a logit-modell összesítőből

Optimization terminated successfully.
Current function value: 0.385678
Iterations 8

Results: Logit

Model:	Logit	Pseudo R-squared:	0.385
Dependent Variable:	target	AIC:	7642.1224
Date:	2022-04-12 17:41	BIC:	8252.2983
No. Observations:	9687	Log-Likelihood:	-3736.1
Df Model:	84	LL-Null:	-6077.1
Df Residuals:	9602	LLR p-value:	0.0000
Converged:	1.0000	Scale:	1.0000
No. Iterations:	8.0000		

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
X1	-0.3561	0.3238	-1.0997	0.2714	-0.9907	0.2785
X2	-1.4349	0.3052	-4.7016	0.0000	-2.0331	-0.8367
X3	-7.8314	1.2669	-6.1816	0.0000	-10.3145	-5.3483
X4	0.0000	0.0000	0.9410	0.3467	-0.0000	0.0000
X5	0.0005	0.0001	6.1985	0.0000	0.0004	0.0007
X6	0.0708	0.0687	1.0311	0.3025	-0.0638	0.2054

Forrás: saját szerkesztés

Mint látható, az optimalizáció során 8 iteráció alatt minimalizálta a loglikelihood függvényt az algoritmus. Szélsőérték-kereső eljárásnak a Newton-Raphson módszert választottam. Érdemes megjegyezni, hogy a dolgozat készítésekor nincs elérhető Python-implementáció a megszokott logit-illesztő metódusokra (változók bevonása/kihagyása például Wald-teszt, vagy likelihood arány alapján) az általam elérhető könyvtárakban, így ez az eljárás az összes magyarázó változót bevonja a modellbe, és nem szelektál. A modell teljesítményét a teszthalmazon a 2. ábra tartalmazza.

2. ábra: Logit-modell klasszifikációs táblázatai

Classes	True 0	True 1	Prediction precision
Pred 0	774	135	85.15%
Pred 1	83	272	76.62%
Class recall	90.32%	66.83%	

Overall accuracy on the test set (threshold=0.5): 82.75%

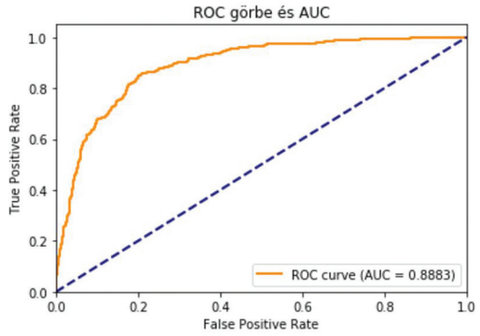
Classes	True 0	True 1	Prediction precision
Pred 0	679	60	91.88%
Pred 1	178	347	66.1%
Class recall	79.23%	85.26%	

Overall accuracy on the test set (optimal threshold=0.2859): 81.17%

Forrás: saját szerkesztés

Mint látható a 2. ábrán, a modell összpontossága valamelyest romlik a a kalkulált döntési (cut-off) paraméter alkalmazásával, azonban a helyesen beazonosított pozitívok aránya jelentősen megnő, de természetesen ennek ára van: a másodfajú hiba (hamis negatívak) csökkentésének árán az elsőfajú hibát (hamis pozitívok) növeltük. E változás megítélése teljesen az adott problémától függ, és mindig meg kell fontolni, hogy melyik hiba növekedése a rosszabb kimenet. Konkrétan a jelen esetben azzal állunk szemben, hogy az elsőfajú hiba növekedése esetén a címanyag „hígabb" (kevésbé minőségi) lesz, míg a másodfajú hiba emelkedése esetén több potenciális megkötethető szerződés „ragad be", mert nem kerülnek bele a címanyagba. Leegyszerűsítve reputációs kockázat (címanyag megítélése az ügynökhálózatban) áll szemben az elszalasztott új üzlettel. Jelen esetben úgy tűnik, hogy jobb döntés a másodfajú hibát csökkenteni, tehát alkalmazni az optimalizált cut-off paramétert, mert nagyobb arányban tudjuk növelni a potenciális új üzletek számát, mint amennyivel romlik a címanyag minősége. A 3. ábrán látható a modell ROC-görbéje és az AUC mutató, amely kerekítve 0,89. Ez az érték már-már kiváló minősítést jelent, kiváltképp egy logisztikus regressziós modellhez képest. A szignifikáns változók köréből megállapítható, hogy a modellt leginkább a szerződésadatok vezérlik, mintsem a különböző ügyfélstatisztikák.

3. ábra: Logit-modell ROC görbéje



Forrás: saját szerkesztés

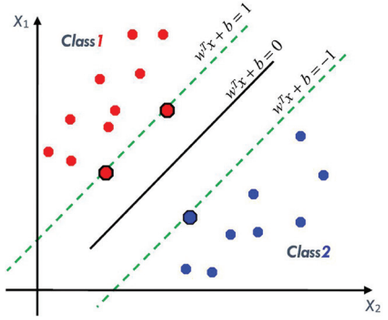
Támaszvektor-gép

A támaszvektor-gépek (továbbiakban röviden: SVM) egy gépi tanulási módszercsalád (Cortes & Vapnik, 1995), felhasználhatósága rendkívül széles körű, hiszen dimenziócsökkentésen kívül minden probléma esetén alkalmazható: regressziós, klaszterezési és klasszifikációs feladatra egyaránt. Megfelelően paraméterezve pedig felhasználhatóságának széleskörűségén túl a pontossága is vetekszik a bonyolult mélytanuló neurális

hálózatokéval. A módszercsalád viszonylag új, ugyan az alapjait már a ’60-as években lefektette Chervonenkis és Vapnik, de a napjainkban használt modell, a nemlineáris SVM puha határral, csak a ’90-es években alakult ki. Ahogy az előbb utaltam rá, kiterjeszthető regressziós problémára is, azonban ennek ismertetésétől el fogok tekinteni, jelen dolgozat fókusza a klasszifikációs eset. A bináris klasszifikációnál a megszokott 0-1 kódolás helyett ±1 jelöli a két kimeneti kategóriát.

Az SVM alapötlete a legegyszerűbb esettel könnyen szemléltethető, ekkor a p magyarázó változó által meghatározott $\mathbb{R}^p$ térben szeretnénk a célváltozó két kategóriája mentén lineárisan szeparálni egymástól az adatpontjainkat. Ezt egy hipersíkkal érjük el olyan módon, hogy a hipersíkhoz legközelebb eső pontok (a támaszvektorok) és a hipersík távolságát maximalizáljuk, az így kapott hipersíkot optimálisnak nevezzük, ezen optimális hipersík meghatározása a modellezés célja. A 4. ábrán egy egyszerű, kétdimenziós eset látható.

4. ábra: Optimális hipersík

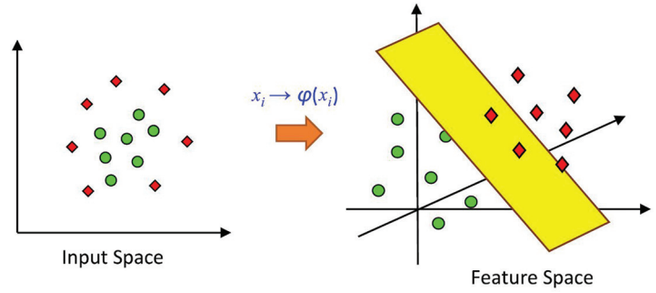


Forrás: Gholami & Fakhari (2017)

Azonban ez a feladat csak akkor oldható meg, ha lineárisan tökéletesen szeparálható a két csoport, ez természetesen túl erős feltevés, amely a gyakorlatban ritkán teljesül. Ezt a problémát hivatott kiküszöbölni a nemlineáris SVM.

Ennek alapötlete az, hogy ha a p-dimenziós térben nem szeparálhatóak lineárisan az adatpontjaink, akkor alkalmazzunk egy ϕ: $\mathbb{R}^p \to \mathbb{R}^q$ (ahol q>p) nemlineáris transzformációt. Ez az ötlet a Cover-tételen alapul, amely értelmében minél nagyobb dimenziójú térbe transzformáljuk egy nemlineáris transzformációval az adathalmazunkat, annál valószínűbb, hogy lineárisan szeparálhatóvá válik (Cover, 1965). A transzformációt úgy választjuk meg (alapvetően közvetetten), hogy q minél nagyobb legyen (megfelelő választásnál akár $q=\infty$), hiszen a Cover-tétel alapján ekkor a legvalószínűbb, hogy lineárisan szeparálhatóak lesz az adathalmazok. Az 5. ábrán egy stilizált példa, ahol egy ϕ: $\mathbb{R}^2 \to \mathbb{R}^3$ nemlineáris transzformáció alkalmazása látható.

5. ábra: Transzformálás magasabb dimenziójú térbe



Forrás: Gholami & Fakhari (2017)

Azonban a ϕ transzformáló függvény kiválasztása általában nem közvetlenül történik, a feladat megoldható a duálisán keresztül, ami azért célszerű, mert ekkor mindössze skalárszorzatok elvégzésén keresztül megtalálható a szélsőérték, és nem kell a végtelen dimenziós térbe vetítő transzformáló függvénnyel számolni (amely egyébként általában nem is tehető meg közvetlenül). A duál feladatban ugyanis megjelenik az ún. magfüggvény, amely a transzformáló függvények skaláris szorzataként definiálandó: $\kappa:\mathbb{R}^{2p} \to \mathbb{R}, \kappa(\boldsymbol{u},\boldsymbol{v}) = \phi(\boldsymbol{u})\phi(\boldsymbol{v})$. A transzformáló függvény és a magfüggvény egyértelműen meghatározzák egymást, és a magfüggvény használatával már közvetlenül megoldható a feladat duálisa (ez a magfüggvény-trükk).

Az alábbi táblázatban (6. ábra) a leggyakrabban használt magfüggvények láthatók, amelyeket klasszifikációs problémák esetén alkalmaznak.

6. ábra: Népszerű magfüggvények

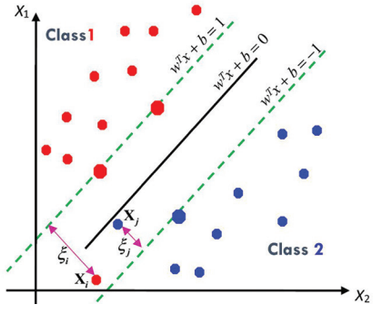
$\kappa(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u} \cdot \boldsymbol{v}$	Lineáris
$\kappa(\boldsymbol{u}, \boldsymbol{v}) = \gamma(\boldsymbol{u} \cdot \boldsymbol{v})^{\rho}$	Polinomiális
$\kappa(\boldsymbol{u}, \boldsymbol{v}) = \tanh(\gamma \boldsymbol{u} \cdot \boldsymbol{v} + \vartheta)$	Szigmoid
$\kappa(\boldsymbol{u}, \boldsymbol{v}) = \exp(-\gamma \lVert \boldsymbol{u} - \boldsymbol{v} \rVert^2)$	Gauss-féle RBF
$\kappa(\boldsymbol{u}, \boldsymbol{v}) = \frac{\sin((n + \frac{1}{2})(\boldsymbol{u} - \boldsymbol{v}))}{2 \sin((\boldsymbol{u} - \boldsymbol{v})/2)}$	Dirichlet

Forrás: saját szerkesztés Gholami & Fakhar (2017) alapján

Azonban a gyakorlatban a nemlineáris SVM-nél is gyakran előfordul, hogy a magasabb dimenziós térben továbbra sem tökéletesen lineárisan szeparálhatóak az adatpontok.

Ekkor egy határátlépési büntetési költség célfüggvénybe építésével már megvalósítható a szeparáció, ezt a módszert nevezzük puha határnak.

7. ábra: Határsértés nem tökéletes szeparáció esetén



Forrás: Gholami & Fakhari (2017)

Ahogy a 7. ábrán is látható, a határsértés mértékét ξ_i-vel fogjuk jelölni, a határsértés költségét pedig C-vel. Az SVM feladat ezek segítségével a következő alakban formalizálható:

$$\min_{w,b}(\|\boldsymbol{w}\| + C\sum_{i=1}^{n}\xi_i) : \quad y_i(\boldsymbol{w}^T\phi(\boldsymbol{x}) + b) \geq 1 - \xi_i \quad \xi_i \geq 0, \qquad ahol\ i = 1, \dots, n \tag{1}$$

ξ_i értéke 0, amennyiben nem történt határsértés, 0 és 1 közötti értéket vesz fel, ha az adatpont beleesik a határmezsgyébe, de a hipersík megfelelő oldalán van, és 1-nél nagyobb, ha ez utóbbi feltétel sem teljesül. A C határsértési költség pedig a modell hiperparamétere, amelynek meghatározása (a magfüggvény, és annak paramétereivel együtt) a modell illesztésének a része.

Az SVM modell tanításához (illetve általában a nem statisztikai alapú gép tanulási modellek esetében) az eredeti adathalmazt tanítás előtt transzformálni szükséges: egyrészt a numerikus változókat standardizálni kell, másrészt one-hot dummy változókat kell képezni, ami azt jelenti, hogy nem hagyunk el egy referenciának szánt kategóriát. Előzetes kísérletezgetések során azt tapasztaltam, hogy a magfüggvények közül egyedül a Gauss-féle RBF jöhet szóba, a többi magfüggvénnyel meg sem tudtam közelíteni az RBF függvénnyel elérhető modellpontosságot, így az optimalizálandó hiperparamétereket leszűkítettem a C paraméterre, valamint az RBF függvény γ paraméterére. Az optimális hiperparamétereket keresztvalidációs eljárással határoztam meg: kísérletezés útján választottam ki a hiperparaméterek egyes tartományait, amin vizsgálódni érdemes. A γ paraméternek 19 lehetséges értéket adtam meg (0,005 és 0,02 között minden érték 0,001 lépésközzel, valamint 3 egyedi, e tartományon kívül eső paraméter), a C büntetőparamétereknek pedig 16-ot (0,5-től 2-ig 0,1 lépésközzel). A hiperparaméter-kereső eljárás pedig az összes lehetséges kombinációval betanítja a modellt az előzetesen elkülönített tanító halmazon, tovább osztva azt validációs és tanító halmazokra, majd elmenti a modell teljesítményét. Azonban ezt az eljárást még bővíteni kellett, mivel az eredmények meglehetősen ingadozóak voltak, így jelenleg egy hiperparaméter-kombinációval többször (egészen pontosan az SVM esetében ötször) újratanul a modell, mindig más tanító-validációs halmaz felosztáson, az így kapott modellteljesítményeket pedig egyszerűen kiátlagolva menti el az adott paraméter-kombinációhoz. Az algoritmus a jelen bekezdésben ismertetett beállításokkal 3040 tanítást végez, ez nagyjából 8 órát vesz igénybe. Az így kapott optimális hiperparaméterek: C=1,2 és γ=0,015. A teszthalmazon mérve a következő eredményeket érte el az így kapott modell (8. ábra).

8. ábra: SVM modell klasszifikációs táblái

Classes	True 0	True 1	Prediction precision
Pred 0	777	134	85.29%
Pred 1	82	271	76.77%
Class recall	90.45%	66.91%	

Overall accuracy on the test set (threshold=0.5): 82.75%

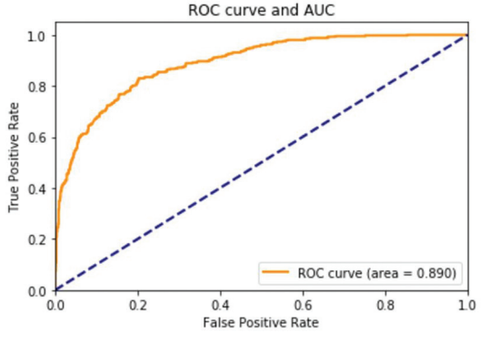
Classes	True 0	True 1	Prediction precision
Pred 0	654	64	91.09%
Pred 1	205	341	62.45%
Class recall	76.14%	84.2%	

Overall accuracy on the test set (optimal threshold=0.267): 78.72%

Forrás: saját szerkesztés

A modell ROC görbéje és AUC mutatója a 9. ábrán látható.

9. ábra: Az SVM modell ROC görbéje



Forrás: saját szerkesztés

Az AUC értéke 0,89, ami szintén szinte kiválónak nevezhető, viszont az interpretálhatóság elvesztése és a keresztvalidáció időigényessége e modell esetén nem nevezhető kifizetődőnek.

Döntési fák és ensemble-technikák

A döntési fa több algoritmus összefoglaló neve, egy olyan adatbányászati módszer, amely használható mind regressziós, mind klasszifikációs esetben. Az alapjait Quinlan (1986) és Breiman és társai (1984) fektették le, előbbi az ID3, majd a C4.5, utóbbiak pedig a CART algoritmusok leírásával. Az ID3 algoritmus csak klasszifikációra alkalmazható, és magyarázó változóként csak kategorikus változókat adhatunk meg. Ennek továbbfejlesztése, a C4.5 pedig már numerikus változókat is kezel, viszont továbbra is csak klasszifikációra alkalmazható, a CART segítségével tudunk regressziós döntési fákat építeni. A különböző algoritmusok alapvetően hasonló logika mentén működnek, az egyes magyarázó változókat elágaztatják többfelé (bináris fák esetén kettőfelé) a felvett értékek mentén úgy, hogy az így kapott részsokaságok valamilyen előre meghatározott mérték (ún. *tisztasági mérték*) szerinti homogenitása a lehető legnagyobb mértékben növekedjen. Az elágaztatást pedig többször elvégezzük. Arra is többféle módszer létezik, hogy végül hol állunk meg, és hogyan kapjuk meg a végső fát. A döntési fák ereje leginkább interpretálhatóságukban rejlik, szinte könnyebben értelmezhetők és bemutathatók, mint a klasszikus lineáris regresszió. Ráadásul hatékonyan képesek detektálni az adatokban rejlő nemlineáris struktúrákat is, és kisebb (adatbányászati mércével mérve, tehát néhány ezres) mintákon is jó eredményeket lehet velük elérni. Hátrányuk viszont, hogy könnyen túltanulnak, ez a probléma főként akkor jelentkezik, ha túl nagyra engedjük nőni a fát, mert ekkor már csak az adott adathalmaz sajátosságaira, a zajra tanul rá. Ez önmagában akár megfelelő keresztvalidációval, visszametszéssel vagy büntetőparaméterek bevezetésével orvosolható, de ennek ellenére is a gyakorlati tapasztalat azt mutatja, hogy kevésbé robusztus és kevésbé pontos modern számítástudományi módszer (Bhavsar & Ganatra, 2012), kiváltképp a többi, általam alkalmazott módszerhez képest. Ami miatt mégis indokolt az ismertetése, mert különböző ensemble-technikákat alkalmazva már kiváló modell építhető, az általam alkalmazott modellek egyike is egy ilyen típusú faalapú módszer.

Az ensemble-technikák alkalmazásával ugyan nagymértékben javíthatók az esetlegesen gyenge faalapú modelljeink, de ezzel éppen az egyik legnagyobb előnyük sérül: az interpretálhatóságuk. Az általam alkalmazott módszer a boosting módszercsaládba tartozik, ez több különböző algoritmus gyűjtőneve (a teljesség igénye nélkül közülük néhány konkrét: AdaBoost, Gradient Boosting Machine (GBM), XGBoost), részletes bemutatásuk azonban túlmutat jelen publikáció keretein. Ezen algoritmusok esetében több modell tanítására kerül sor, az egyes modellek eredményei pedig iteratívan kerülnek felhasználásra. A t-edik modell hibáját egy előre definiált veszteségfüggvénnyel megmérjük a validációs adathalmazon, majd a hibásan besorolt esetekre nagyobb súlyt alkalmazva új mintát készítünk, a t+1-edik modell pedig már ezen az új tanulóhalmazon tanul. A veszteségfüggvény általában algoritmusra jellemző, például az AdaBoost exponenciális veszteségfüggvényt használ (Zhang, 2004):

$$L(\hat{y}, y) = \exp(-\hat{y} \cdot y) \tag{2}$$

Az általam alkalmazott Gradient Boosting Machine (Hastie & Tibshirani & Friedman, 2009) esetén pedig a keresztentrópia a veszteségfüggvény:

$$L(\hat{y}, y) = -y \cdot \ln(\hat{y}) - (1 - y) \cdot \ln(1 - \hat{y}) \tag{3}$$

A veszteségfüggvényen túl további különbséget jelent az algoritmusok között például a súlyozás meghatározásának a módja is, GBM esetében például ez gradiens ereszkedés módszerével történik.

A számtalan szóba jöhető döntésifa-alapú ensemble technika közül azért a GBM-re esett a választás, mert az egyszerre az egyik leggyorsabban illeszthető és legpontosabb algoritmus. Az SVM-hez hasonlóan ez esetben is standardizált numerikus és one-hot eljárással átkódolt kategorikus változókkal dolgoztam, valamint az előző alfejezetben ismertetett hiperparaméter-optimalizáló eljárás itt is alkalmazásra került. A GBM-modellek esetében a legfontosabb hiperparaméterek a modellbe bekerülő fák száma, a tanulási ráta, a minimális mintaméret, amit egy node-nak tartalmaznia kell (továbbiakban röviden min_split), a minimális mintaméret, amit egy levél node-nak tartalmaznia kell (továbbiakban röviden min_leaf), valamint a fa maximális mélysége. Kísérletezés útján arra a megállapításra jutottam, hogy a tanulási rátát a 0,1-es alapértelmezett értékről nem érdemes megváltoztatni jelen modellezési feladatnál. Illetve a scikit-learn könyvtár legtöbb felügyelt tanuló algoritmusa rendelkezik az ún. early-stopping nevű beépített metódussal, amely a túltanulási elleni védelem egyik legfontosabb eszköze. A GBM esetében a modellbe bekerülő fák számát kontrollálja az early-stopping, tehát egészen addig fog a modell egy újabb fával bővülni, amíg a teszthalmazon mért hibafüggvény értéke csökken. Így sem ezt, sem pedig a tanulási ráta paramétereket nem szükséges a hiperparaméter-kereső algoritmussal optimalizálni, csupán egy maximális értéket adtam meg a fák számának, amely 200 lett, de ezt az értéket az early-stopping használata mellett sosem érték el a különböző modellek. A fák maximális mélységét 3 és 9 közötti egész számokon, a min_split és min_leaf paramétereket 7 és 13 közötti egész számokon optimalizáltam. Az ötszörös keresztvalidáció ezen hiperparaméterekkel kevésnek tűnt, emiatt ezt felemeltem tízszeresre. Ez összesen 3430 tanítást jelentett, ami kicsivel több, mint 3 óra alatt futott le. Az optimális maximális mélység 5, a min_split 11, míg a min_leaf értéke 8 lett, míg az early-stoppinggal kontrollált fák száma átlagosan 100 körül alakult. Az így kapott modell teljesítménye pedig a 10. és 11. ábrán látható.

10. ábra: GBM-modell klasszifikációs táblái

Classes	True 0	True 1	Prediction precision
Pred 0	769	92	89.31%
Pred 1	90	313	77.67%
Class recall	89.52%	77.28%	

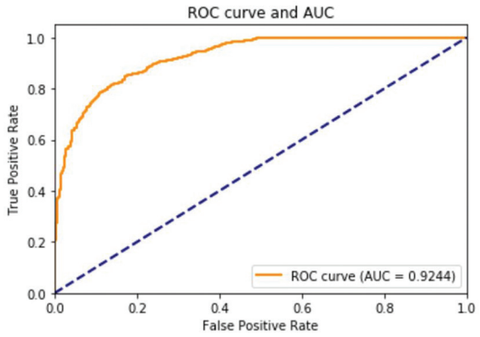
Overall accuracy on the test set (threshold=0.5): 85.6%

Classes	True 0	True 1	Prediction precision
Pred 0	707	60	92.18%
Pred 1	152	345	69.42%
Class recall	82.31%	85.19%	

Overall accuracy on the test set (optimal threshold=0.3553): 83.23%

Forrás: saját szerkesztés

11. ábra: GBM-modell ROC görbéje



Forrás: saját szerkesztés

A modell az előzetes várakozásoknak megfelelően kiválóan teljesít, használatával mind a logit, mind az SVM modelleknél pontosabb eredmény érhető el. Valamint a ***J***-mutató alapján számolt optimális cut-off paraméterrel is itt érhető el eddig a legmagasabb összesített modellpontosság, miközben a pozitívnak prediktált esetek 69,4%-a valóban pozitív mintaelem, amely az eddigi legmagasabb minőségű címanyag előállítását jelenti.

Neurális hálózat

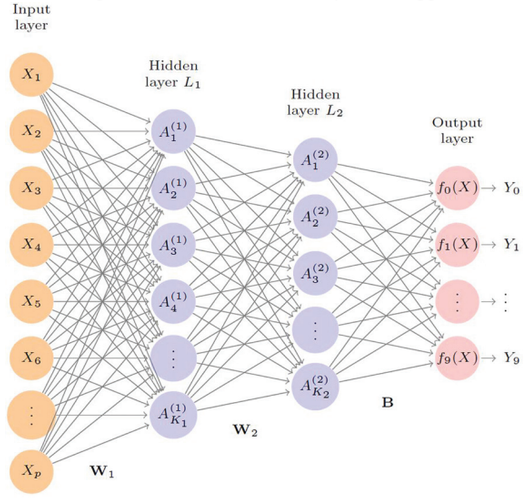
A mesterséges neurális hálózatok csomópontok és irányított kapcsolatok strukturált halmazából állnak, ahol a csomópontokat neuronoknak nevezzük. A hálózat csomópontjait három nagy csoportba soroljuk: bemeneti réteg (input layer), rejtett réteg (hidden layer, ez jelenthet akár több réteget is) és a kimeneti réteg (output layer). Az egyes neuronok (a bemeneti réteg elemeinek kivételével) a hálózat számításokat végző elemei. Ha egy hálózat több rejtett réteggel is rendelkezik, akkor mélytanuló hálónak *(deep learning)* is nevezzük. A neurális hálók további csoportosítási szempontja a neuronok közötti irányított kapcsolatok (élek) szervezési elvének mentén is történhet, ha egy hálózatban csak előre haladó élek vannak, akkor előrecsatolt hálózatnak *(feed-forward neural network)*, ellenkező esetben pedig rekurrens hálózatnak *(recurrent neural network)* nevezzük. Jelen dolgozatban kizárólag előrecsatolt mesterséges neurális hálózatokat használok, így a további ismertetés során kizárólag ezekre fogok koncentrálni.

A következőkben a Hastie és szerzőtársai (2021) által használt jelölésrendszert fogom alkalmazni. Jelölje egy hálózatban a rejtett rétegek számát *i* (i=1,…,I) és k (k=1,…,K_i) a neuronok számát az egyes rejtett rétegekben (magyarázó változóinkat pedig x_1,…,x_p jelöli). Ekkor az *i*-edik rétegben lévő *k*-adik neuront a következő leképezéssel írjuk le:

$$A_k^{(i)} = \begin{cases} g\left(w_{k0}^{(i)} + \sum_{j=1}^{p} w_{kj}^{(i)} x_j\right), & ha\ i = 1 \\ g\left(w_{k0}^{(i)} + \sum_{j=1}^{K_i} w_{kj}^{(i)} A_k^{(i-1)}\right) & különben. \end{cases} \quad (4)$$

Ahol g az ún. aktivációs függvény, w_{k0} pedig a torzítás. Tehát egy neuron kimeneti értéke nem más, mint a bemeneti értékek lineáris kombinációjának valamilyen eltolására alkalmazott aktivációs függvény értéke. A 12. ábrán egy két rejtett réteggel rendelkező előrecsatolt neurális hálózat látható ($\boldsymbol{w}_1$,$\boldsymbol{w}_2$ és $\boldsymbol{B}$ a súlyokat és eltolásokat tartalmazó mátrixok).

12. ábra: Előrecsatolt mesterséges neurális hálózat két rejtett réteggel



Forrás: Hastie és szerzőtársai (2021)

Az aktivációs függvény szerepe rendkívül fontos, egyrészt ennek segítségével jelenik meg a nemlinearitás a modellben. Másrészt, ha ún. szigmoid típusú függvényt választunk, akkor belátható, hogy egy mélytanuló neurális hálózat univerzális approximátor, sőt, univerzális osztályozó is (Altrichter és szerzőtársai, 2006). A 13. ábra a legnépszerűbb aktivációs függvények közül tartalmaz néhányat, a teljesség igénye nélkül.

13. ábra: Gyakori aktivációs függvények

Aktivációs függvény neve	Definíció
Logisztikus	$g(x) = \frac{1}{1+e^{-x}}$
Hiperbolikus tangens	$g(x) = \tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$
ReLU	$g(x) = \max(0, x)$
Leaky-ReLU	$g(x) = \max(ax, x)\ ahol\ a \in (0,1)$
ELU	$g(x) = \begin{cases} a(e^{x}-1), & ha\ x < 0 \\ x, & ha\ x \geq 0 \end{cases}$
Softplus	$g(x) = \ln(1+e^{x})$

Forrás: saját szerkesztés a TensorFlow 2.0-ban megtalálható aktivációs függvények alapján

Azonban nem csak az aktivációs függvény felel azért, hogy az általunk modellezni kívánt (feltételezhetően többdimenziós és nemlineáris) függvényt megfelelő mértékben megközelítsük, maga a háló architektúrája is hozzájárul ehhez, ezért a háló általában több rejtett réteget is tartalmaz. A rejtett rétegek száma, illetve az egyes rejtett rétegek által tartalmazott neuronok száma és a hozzájuk tartozó aktivációs függvények típusa a modell hiperparaméterei.

A kimeneti réteg neuronjainak számát a modellezendő probléma meghatározza, regressziós esetben egy neuronból áll, klasszifikációnál pedig annyi neuront tartalmaz, ahány kategóriája van a modellezett változónknak (reprezentálva az egyes kategóriákat). Utóbbi esetben az alapján dönthetünk a kategóriába-sorolásról, hogy melyik kategóriát reprezentáló kimeneti neuron értéke lett a legmagasabb. Kivételt képezhet ez alól a bináris klasszifikáció esete, ekkor ugyanis elégséges egyetlen neuron használata is, mégpedig, ha logisztikus aktivációs függvényt használunk a kimeneti rétegben, hiszen ekkor 0 és 1 közötti valós számmá transzformáljuk a modell eredményét, majd egy „cut-off" paraméter alkalmazásával sorolunk be az adott kategóriába.

A hálózat tanítása során a $\boldsymbol{w}^{(i)}$ (rétegenként a súlyokat és torzításokat tartalmazó) mátrixokat változtatjuk (finomhangoljuk) olyan módon, hogy a bemeneti réteget egymás után többször végigáramoltatjuk a hálózaton. A kezdeti súlyokat és eltolásokat általában véletlenszerűen határozzuk meg, az egyes végigáramoltatások (epoch-ok) során kapott predikciók hibáját egy előre definiált veszteségfüggvénnyel mérjük meg (természetesen egy előre elkülönített teszthalmazon). Majd úgy módosítjuk a $\boldsymbol{w}^{(i)}$ mátrixokat, hogy a lehető legnagyobb mértékben csökkenjen a veszteségfüggvény. Alapvetően a modellezendő probléma (regresszió vagy klasszifikáció) determinálja, hogy milyen veszteségfüggvényekből választhatunk. Bináris klasszifikáció esetében használható akár a klasszifikációs hiba, vagy a keresztentrópia is (Altrichter és szerzőtársai, 2006). A hálózat tanítása során tehát végső soron a veszteségfüggvény globális minimumának megtalálása a cél, ami numerikus módszerekkel történhet. Ez a gyakorlatban a gradiens ereszkedés továbbfejlesztett változataival[7] lehetséges a neurális hálózatokban.

A neurális hálózatok illesztése a legbonyolultabb és legidőigényesebb az általam alkalmazott módszerek közül a modellparaméterek lehetséges megválasztásának széleskörűsége okán. Abból a célból, hogy némileg egyszerűsítsem ezt a feladatot, a kísérletezések során arra jutottam, hogy három nagy csoportra érdemes bontani a keresztvalidálásra épülő hiperparaméter-keresést: az aktivációs függvény, valamint a rejtett rétegek és neuronjaik számának meghatározására és a tanulási ráta megválasztására. Mielőtt e három lépést ismertetném, érdemes röviden áttekinteni az általános hálózatarchitektúrát, amellyel dolgoztam. Alapvetően csak előrecsatolt mélytanuló hálózatokat illesztettem, a különböző mélységű és neuronszámú hálóknak pedig az alábbi közös tulajdonságai voltak:

- a kimeneti réteg aktivációs függvénye a logisztikus függvény
- minden rejtett réteg után az adatpontokat normalizálja az algoritmus
- minden rejtett rétegnél a beáramló inputok véletlenszerűen kiválasztott 5%-a nem vesz részt tovább a tanításban az adott epoch-ban (ún *dropout-layer* alkalmazása)
- a tanítás során a bináris keresztentrópia veszteségfüggvényt használtam, optimalizáló eljárásként pedig az Adam (Ba & Kingma, 2014) algoritmust.

A logisztikus függvény használatát a kimeneti rétegben a bináris klasszifikációs probléma teszi szükségessé. Az adatpontok normalizálása a futási időt csökkenti, mivel gyorsítja a konvergenciát (Ioffe & Szegedy, 2015). A dropout rétegek alkalmazása pedig a túltanulás megelőzését segíti, amely problémára mélytanuló neurális hálózatok esetén különös figyelmet kell fordítani (Srivastava & szerzőtársai, 2014). A túltanulás megakadályozásának másik eszköze az early-stopping, amely a neurális hálózatok esetén a tanítás során futtatandó epoch-ok számát szabályozza, azaz addig tanul a hálózat, amíg a validációs halmazon mért hiba csökken.

Először a hálózat rejtett rétegeiben használt aktivációs függvény kiválasztását végeztem el. Erre egységesen egy olyan architektúrát használtam, amelyben a kimeneti, logisztikus aktivációs függvényt használó rétegen kívül 4 rejtett réteg található, rétegenként 100 neuronnal. Összesen hatféle aktivációs függvényt adtam meg: logisztikus, ReLU, ELU, Leaky-ReLU, Softplus és Hiperbolikus tangens. Az aktivációs függvény meghatározása során a tanító halmaz 15%-át használtam validációs halmazként, egyazon aktivációs függvénnyel pedig 30 különböző tanító-validáló felosztáson történt a tanítás. A maximális epoch-szám 200 volt,

amelyet 5-*türelmi paraméterű*[8] early-stopping segítségével korlátoztam. A validációs halmazon mérve átlagosan az ($\alpha=1$ paraméterű) ELU aktivációs függvény esetén volt a legkisebb a hiba, így ezzel dolgoztam tovább (14. ábra).

14. ábra: Keresztvalidációval tesztelt aktivációs függvények

model	patience	loss	val_loss
ELU	5	0.2884	0.3440
Softplus	5	0.2906	0.3448
Sigmoid	5	0.3038	0.3599
Leaky-ReLU	5	0.2572	0.3663
Tanh	5	0.2841	0.3678
ReLU	5	0.2588	0.3766

Forrás: saját szerkesztés

Ezután öt különböző neurális hálózatot definiáltam, amelyekből kettő-kettő azonos rétegszámmal rendelkezik, de a neuronok számában eltérnek. A 15. ábra részletesen tartalmazza a különböző modellek felépítését.

15. ábra: Keresztvalidációval tesztelt hálózatarchitektúrák

Modell	Rejtett rétegek száma	Első és utolsó rejtett réteg neuronjainak száma	Fennmaradó rejtett rétegek neuronjainak száma
1. modell	7	200	500
2. modell	7	50	200
3. modell	5	100	300
4. modell	5	30	100
5. modell	10	50	50

Forrás: saját szerkesztés

A keresztvalidáció során az egyes modelleket többféle türelmi paraméterrel futtattam le. Az eredmények alapvetően elég hasonlóak lettek, a néhány legjobb látható a 16. ábrán (utolsó két oszlop a validációs halmazon mérve).

16. ábra: Architektúra-választás eredményei

model	p	loss	val_loss	auc
2	8	0.2679	0.3553	0.9207
4	5	0.2941	0.3390	0.9223
4	8	0.2772	0.3423	0.9210
5	5	0.2952	0.3455	0.9223
5	8	0.2826	0.3473	0.9220

Forrás: saját szerkesztés

A 16. ábra alapján megállapítható, hogy az utolsó, 5. modell teljesítménye a legstabilabb és legpontosabb, így ezzel dolgoztam tovább. Ezután az optimális tanulási rátát határoztam meg az eddig is alkalmazott keresztvalidációs eljárással. Empirikus úton arra jutottam, hogy a türelmi paramétert nem veszem fel az optimalizálandó hiperparaméterek közé, mivel így arányaiban nagyobb mértékben csökkenthettem a tanítási időt, mint amennyit modellpontosságban ezzel nyerni lehetett. Érdemes megjegyezni, hogy a hiperparaméter-keresés legfontosabb részei az aktivációs függvény és az architektúra megválasztásával már megtörténtek, a cél inkább a modell robusztusságának növelése, mintsem a pontosság érdemi javítása. A kezdeti tanulási ráta végül 0,03 lett. Tehát az alkalmazásra kerülő modell a következőképpen néz ki: 10 rejtett rétegből áll, rétegenként 50 neuronnal, amelyek mindegyikének aktivációs függvénye az 1-paraméterű ELU függvény. A 17. és 18. ábra az így kapott modelleredményeket tartalmazza, amelyeket a teszthalamazon ért el a modell.

17. ábra: FNN-modell klaszifikációs mátrixai

Classes	True 0	True 1	Prediction precision
Pred 0	766	108	87.64%
Pred 1	93	297	76.15%
Class recall	89.17%	73.33%	

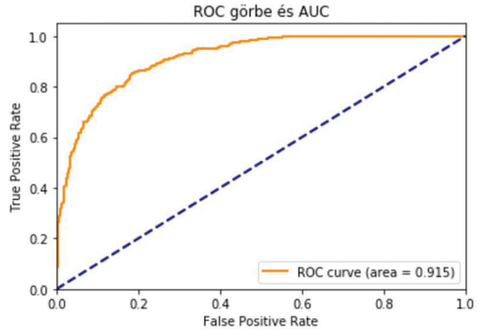
Overall accuracy on the test set (threshold=0.5): 84.1%

Classes	True 0	True 1	Prediction precision
Pred 0	727	81	89.98%
Pred 1	132	324	71.05%
Class recall	84.63%	80.0%	

Overall accuracy on the test set (optimal threshold=0.3626): 83.15%

Forrás: saját szerkesztés

18. ábra: FNN-modell ROC-görbéje



Forrás: saját szerkesztés

Az AUC mutató kismértékben elmarad a GBM-modelléhez képest, mivel az elsőfajú hibája magasabb ennek a modellnek. Cserébe a másodfajú hiba viszont tovább csökkent, a J-mutató alapján számolt cut-off paraméterrel már a pozitív mintaelemek 71%-át képes a modell helyesen beazonosítani. Összességében a hiperparaméterek meghatározásához közel 600 különböző tanításra volt szükség, ezek együttesen, tisztán a futási időket mérve, nagyjából 18 órát vettek igénybe.

Modelleredmények értékelése

A keresztértékesítési lehetőségeket feltáró modellezési folyamat sikeresnek tekinthető, ugyanis az így kapott modellek pontossága bőven elégséges ahhoz, hogy minőségi címanyag legyen általuk előállítható. A következőkben röviden áttekintem és összegzem a modellek eredményeit, és a gyakorlati felhasználás szempontjából is értékelem. Végül néhány lehetséges továbbfejlesztési lehetőséget is felvázolok.

A 19.ábra a négy modellt foglalja röviden össze.

19. ábra: Modellek teljesítménye

Modell	AUC	Tanítások száma	Tanítási idő (óra)
Logit	0.888	1	0
SVM	0.890	3040	8
GBM	0.924	3430	3.25
FNN	0.915	600	18

Forrás: saját szerkesztés

Ugyan az egyes modellek teljesítményeinek különbségei nem túl nagyok, de mindent figyelembe véve egyértelműen a GBM-modell lett a legjobb, hiszen ez adja a legpontosabb eredményt, és a modell hiperparamétereinek megkeresése sem vesz túlságosan sok időt igénybe. A tanítási idők figyelembevételét a címanyagok előállításának tervezhetősége indokolja, ugyanis éles üzleti környezetben rendszeres időközönként érdemes finomhangolni és újratanítani a modellt friss adatokon, hiszen nincs garancia a modell által feltárandó döntési szabály időbeli állandóságára. Felmerülhet a kérdés, látva, hogy a probléma könnyen modellezhető: érdemes-e egyáltalán gépi tanulási modelleket alkalmazni. Véleményem szerint igen, bár nem drasztikusak a modellek közötti különbségek, de érdemben tudunk csökkenteni a logisztikus regresszióhoz képest a másodfajú hibán, ezáltal „tisztább” címanyag kiadását lehetővé téve.

Ezen túl felmerülhet, hogy vajon a pozitív mintaelemeink nem szeparálhatóak-e más módszerrel, ugyanilyen hatékonysággal. E kérdés kiderítése érdekében több különböző klaszterelemzési módszert is kipróbáltam[9] , és minden módszer esetében arra jutottam, hogy nem lehetséges. Ugyan kialakulnak elkülönülő, értelmezhető méretű klaszterek, de a pozitív mintaelemek relatív gyakorisága nem különbözött jelentősen egyik klaszterben sem. E klaszterelemzési eredményeket végül ennél részletesebben nem ismertetem, egyrészt a továbbiakban felhasználható eredmények hiánya, valamint terjedelmi okok miatt. A klaszterelemzési kísérlet tanulsága, hogy felügyelt tanuló módszerekkel érdemes feltárni e keresztértékesítési potenciálokat.

Érdemes elgondolkozni azon, hogy az így kapott modellek pontosan hogyan is értelmezhetők, hiszen ez alapvetően meghatározza a felhasználhatóságukat. Ugyan léteznek eszközök az összetett gépi tanulási modellek interpretációjára, de a felhasznált adatkör bizalmas jellege, illetve terjedelmi okok miatt jelen publikációban ez nem került fókuszba. Természetesen üzleti környezetben a címanyag-generáló modell előállításán túl az interpretáció is kiemelt jelentőségű. Így a logisztikus regresszió illesztése önmagában emiatt is fontos volt, hiszen ebből kiderült, hogy a 10%-on szignifikáns változók elsősorban a szerződésadatok, azon belül is a gépjárműre vonatkozó kockázati információk. A tanításhoz használt szerződések kizárólag a biztosító függő ügynökei által megkötöttek voltak, ami indukálja is, hogy pontosan mit tanultak meg a különböző gépi tanulási modellek: leginkább az ügynökhálózat keresztértékesítési „profilját”, nem pedig ügyfélaffinitást. Tehát azon ügyfelek (vagyis sokkal inkább gépjárművek) körét, amelyekre a függő ügynökök jóval nagyobb eséllyel tudnak cascót értékesíteni a meglévő kgfb-szerződés mellé. Ez persze a modellek gyakorlati felhasználhatóságát is meghatározza, ugyanis, ha függő ügynöki kgfb-állományon tanítunk, és utána függő ügynöki kgfb-állomány tanításra nem használt részhalmazából szeretnénk címanyagot előállítani, akkor tulajdonképpen csupán a modell másodfajú hibája kerülne kiadásra. Hiszen ekkor ki kéne szűrni azokat az eseteket, amelyekhez már van kötve casco (ezek ~70%-át, mint látható volt, helyesen be is azonosítaná a modell), a fennmaradó rész pedig a másodfajú hiba. Ugyan e szerződések körét is tekinthetjük akár értelmes címanyagnak, hiszen valamilyen értelemben mégiscsak hasonlítanak azon szerződésekhez, amelyekhez már van kötve casco, de ehelyett inkább értékesítési csatornák között érdemes a modellt alkalmazni véleményem szerint, a minél magasabb értékesítési ráta elérésének érdekében. Méghozzá olyan módon, hogy egy kiválasztott értékesítési csatorna állományán modellezünk, egy másik értékesítési csatorna állományán alkalmazzuk a modellt, majd az így kapott címanyagot kiadhatjuk azon értékesítési csatorna számára feldolgozásra, amelyen a modellezést végeztük. Mivel a modell tanításakor akaratlanul mégiscsak azzal a feltételezéssel élünk, hogy a múltban megtapasztalt keresztérékesítési arány a mérvadó: amely kgfb-szerződés mellé ez idáig nem lett casco is értékesítve, ahhoz nem is lehetséges. Tehát általánosan, ha azon értékesítési csatorna állományát vesszük tanító halmaznak, ahol a legmagasabb a kgfb-casco keresztértékesítési arány, majd ezt az egyéb (például digitális) értékesítési csatornák cascóval nem rendelkező kgfb-állományára alkalmazzuk, akkor ezzel elő is áll egy, az eredeti értékesítési csatorna számára várhatóan magas hatékonysággal feldolgozható címanyag.

Jelen modellstruktúrát többféle módon is tovább lehetne fejleszteni, ennek irányát a címanyagok értékesítési hálózat által való feldolgozásának a tapasztalatai is kijelölhetik. Egy irány lehet például, ha jobban ügyfélaffinitás irányába szeretnénk eltolni a modellt, ehhez azonban valószínűleg gyökeres átalakításokra lenne szükség. Kezdve a magyarázó változók körével, valószínűleg minden konkrét szerződésadatot mellőzni kellene, és kizárólag az ügyfélről elérhető információk alapján megpróbálni modellt építeni. Azonban a csak kgfb-szerződéssel rendelkezőkről kétséges, hogy elegendő adat állna rendelkezésre egy ilyen modell elkészítéséhez.

Inkább értékesítési csatornák között érdemes a modellt alkalmazni.

További érdekes kérdés, hogy lehetséges lenne-e más biztosítási termékre való modellt előállítani (akár csak a mostani, értékesítési hálózatot modellező keretrendszerben), hiszen a kgfb- és a casco-szerződések között fennálló szoros és könnyen modellezhető kapcsolat magától értetődő. Viszont egy lakásbiztosítás vagy akár egy életbiztosítási termék kapcsán már közel sem mondható triviálisnak a kérdés, tehát ilyen irányba is érdemes lehet tovább vizsgálódni.

Összefoglalás

Dolgozatomban a gépi tanulási algoritmusok egy lehetséges biztosítási alkalmazását kívántam megvizsgálni. Egy olyan modell létrehozását tűztem ki célul, amely egy hazai biztosító kötelező gépjármű-felelősségbiztosítás- és casco-állományai között fennálló keresztértékesítési kapcsolatot hivatott leírni. Az elemzési folyamat és a modellezés által megkaphattunk egy olyan modellt, amely alkalmas a biztosító meglévő állományából beazonosítani olyan ügyfeleket, akiket érdemes célzottan megkeresni egy casco-szerződéssel. A modellezési feladat ennek fényében egy bináris klasszifikációs probléma, ahol azokat a szerződéseket tekintettük pozitív esetnek, amelyeknél megvalósult a casco-keresztértékesítés a meglévő kötelező biztosításra.

A modellezéshez négy modelltípust alkalmaztam: a logisztikus regressziót, a döntésifa-alapú módszereket, a támaszvektor-gépet, valamint a mesterséges neurális hálózatokat. A faalapú módszerek közül végül a gyakorlatban a Gradient Boosting Machine nevű algoritmust választottam pontossága és gyorsasága okán, a támaszvektor-gépek közül pedig szintén a legmodernebb variánst, a puha határral kiegészített nemlineáris SVM-et használtam. Neurális hálózat esetében pedig egy olyan előrecsatolt, többrétegű és teljesen összefüggő hálózatot használtam, melynek kimeneti rétegében egy logisztikus aktivációs függvényű neuron van.

Már a logisztikus regresszió illesztése kapcsán kiderült, hogy a problémakör meglehetősen pontosan modellezhető. Illetve e modell alapján kiderült, hogy leginkább a gépjárműadatok köre alkotja a szignifikáns változókat a modellben. Azonban a fejlettebb gépi tanulási módszerek alkalmazása sem volt hiába való, hiszen mind a GBM, mind az FNN modellek pontosabbak tudtak lenni. Elsősorban a másodfajú hiba volt csökkenthető általuk, ami alkalmazási szituációtól függően akár még előnyösebb is lehet, mint az elsőfajú hiba csökkenése, mivel így tisztább, minőségibb címanyag állítható elő az adott értékesítési csatorna számára. A gépi tanulási modellek illesztésekor különös figyelmet fordítottam a modellparaméterek meghatározására, amelyeket n-szeres keresztvalidációval határoztam meg.

Négy modelltípust alkalmaztam: a logisztikus regressziót, a döntésifa-alapú módszereket, a támaszvektor-gépet, a mesterséges neurális hálózatokat.

A logisztikus regressziónak köszönhetően leszűrhető volt az elemzés egyik legfontosabb tanulsága, miszerint az így kapott modellek sokkal inkább az érintett értékesítési csatornát modellezik, mintsem az ügyfélmagatartást. Potenciális értékét ez önmagában nem csökkenti, viszont a felhasználhatóság módját alapjaiban meghatározza. Így azt a következtetést vontam le, hogy a modellt két teljesen elkülönülő (és eltérő keresztértékesítési rátájú) értékesítési csatorna által megkötött szerződések halmazán érdemes alkalmazni, az egyiken betanítjuk a modellt, míg a másikon tudjuk alkalmazni, ezzel várhatóan magas értékesítésisiker-rátájú címanyagot generálva.

Végezetül továbbfejlesztési ötleteket ismertettem, azonban ezek az irányvonalak már jóval bonyolultabb modelltípusokat is igényelhetnek, illetve közel sem biztos, hogy egy biztosító rendelkezik az ezekhez szükséges adatkörrel. Mindenesetre a jelen dolgozatban használt modellstruktúra más biztosítási termék párosokon alkalmazva is komoly kihívásokat jelenthet, és a bennük rejlő lehetőségek felmérése további elemzéseket igényel.

HIVATKOZÁSOK

[1]False positive rate: tévesen 1-nek prediktált esetek aránya a valójában 0-ba tartozókhoz viszonyítva
[2]True positive rate: helyesen 1-nek prediktált esetek aránya a valójában 1-be tartozókhoz viszonyítva
[3]Van Rossum, G., & Drake, F. L. (2009) Python 3 Reference Manual. CreateSpace.
[4]Például ügyfélszintű kárstatisztikák. A részletes felsorolás az adatok bizalmas jellege miatt nem lehetséges.
[5]https://www.netrisk.hu/kgfb-index.html
[6]https://www.ksh.hu/stadat_files/ara/hu/ara0041.html
[7]Például GDM, RMSProp vagy az Adam nevű algoritmusokkal
[8]A paraméterben megadott számú epoch még lefut azután is, miután már növekedés tapasztalható a validációs halmaz hibafüggvényében. Az oszcillálás kezelésére miatt szükséges ez a módszer.
[9]K-közép és hierarchikus klaszterelemzést, valamint a DBSCAN nevű algoritmust is. Utóbbi kettőt kétféleképpen: euklideszi, valamint az ún. Gower-távolság alapján is.

IRODALOMJEGYZÉK

Altrichter, M. – Horváth, G. – Pataki, B. et al (2006) Neurális hálózatok. Panem Könyvkiadó Kft.

Ba, J. – Kingma, D. (2014) Adam: A Method for Stochastic Optimization.
https://doi.org/10.48550/arXiv.1412.6980 Letöltés: 2023.04.05.

Bhavsar, H. – Ganatra, A. (2012) A Comparative Study of Training Algorithms for Supervised Machine Learning. International Journal of Soft Computing and Engineering (IJSCE) 2(4). pp. 74–81.

Breiman, L. – Friedman, J. – Olshen, R. et al (1984). Classification And Regression Trees.
https://doi.org/10.1201/9781315139470 Letöltés: 2023.04.05.

Cortes, C. – Vapnik, V. (1995) Support-vector networks. Machine Learning 20. pp. 273–297.
https://doi.org/10.1007/BF00994018 Letöltés: 2023.04.05.

Cover, T. M. (1965) Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition. IEEE Transactions on Electronic Computers 14(3). pp. 326–334.
https://doi.org/10.1109/PGEC.1965.264137 Letöltés: 2023.04.05.

Gholami, R. – Fakhari, N. (2017) Support Vector Machine: Principles, Parameters, and Applications. In Handbook of Neural Computation, pp. 515–535.
https://doi.org/10.1016/B978-0-12-811318-9.00027-2 Letöltés: 2023.04.05.

Hastie, T. – James, G. – Tibshirani, R. et al (2021) An Introduction to Statistical Learning with Applications in R, Second Edition.
https://doi.org/10.1007/978-1-0716-1418-1 Letöltés: 2023.04.05.

Hastie, T. – Tibshirani, R. – Friedman, J. (2009) The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition. Springer.

Horváth, A. – Paulovits, M. (2016) Biztosítási piac és szolgáltatások a lakosság szemével. Biztosítás és Kockázat 3 (4). pp. 86–109.
https://doi.org/10.18530/BK.2016.4.86 Letöltés: 2023.04.05.

Ioffe, S. – Szegedy, C. (2015) Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.
https://doi.org/10.48550/arXiv.1502.03167 Letöltés: 2023.04.05.

Kovács, E (2014) Többváltozós adatelemzés. TypoTex Kiadó. pp. 126–147.

Quinlan, J. R: (1986) Introduction of decision trees. Machine Learning 1. pp. 81–106. https://doi.org/10.1007/BF00116251 Letöltés: 2023.04.05.

Srivastava, N. – Hinton, G. – Krizhevsky, A. et al (2014) Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research 15. pp. 1929–1958.

Szabó, D. (2015) Adatbányászat a biztosítási szektorban. Biztosítás és Kockázat 2 (4). pp. 62–77.
https://doi.org/10.18530/BK.2015.4.62 Letöltés: 2023.04.05.

Tukey, W. (1977) Exploratory Data Analysis. Addison-Wesley.
https://doi.org/10.1007/978-0-387-32833-1_136 Letöltés: 2023.04.05.

Youden, W. J. (1950) Index for rating diagnostic tests. Cancer 3 (1). pp. 32–35.
https://doi.org/10.1002/1097-0142(1950)3:1<32::AID-CNCR2820030106>3.0.CO;2-3
Letöltés: 2023.04.05.

Zhang, T. (2004) Statistical behavior and consistency of classification methods based on convex risk minimization. The Annals of Statistics. 32 (1). pp. 56–85.
https://doi.org/10.1214/aos/1079120130 Letöltés: 2023.04.05.