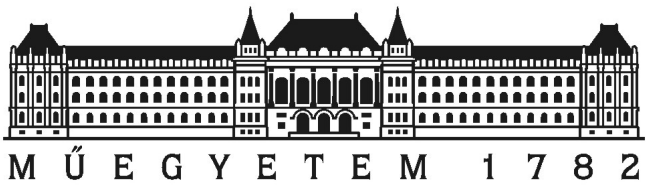

TDK DOLGOZAT

Elsőéves VBK hallgatók teljesítményének vizsgálata a Covid előtti és utáni időszakból

Köller Donát Ákos & Vlaszov Artúr
BME Matematikus MSc
Adattudomány szakirány

Témavezető: Szilágyi Brigitta
Geometria Tanszék

BME Matematika Intézet
Budapest
2022

Tartalomjegyzék

1. Bevezetés **1**

2. A kognitív tesztről **1**

3. Adatprepació **1**

3.1. Az adatok beszerzése, adattáblák jellemzése 1

3.2. Adattisztítás 1

4. Felderítő adatelemzés a két évben **4**

5. Prediktív analitika **4**

5.1. Modellek és metodológia 4

5.2. Osztályozó algoritmusok 5

5.2.1. Lineáris regresszió 5

5.2.2. Naive Bayes 5

5.2.3. Gradient Tree Boosting 5

5.2.4. Logisztikus regresszió 5

5.2.5. SVM 6

5.3. Implementálás és optimalizálás 6

6. Modellek kiértékelése **7**

1. Bevezetés

2. A kognitív tesztről

3. Adatprepació

3.1. Az adatok beszerzése, adattáblák jellemzése

A kutatáshoz használt adatokat Szilágyi Brigitta tanárnő bocsátotta rendelkezésünkre.

3.2. Adattisztítás

A felderítő és modellezési fázisban használt adattáblát a rendelkezésre álló táblák megfelelő mértékű tisztításából, majd ezt követő összeillesztéséből nyertük. Ezen műveleteket Python-ban valamint R-ben végeztük el.

A kezdeti adattáblák közül a matematika és kognitív eredményeket tartalmazó adatsor tisztítása során először egységesítettük a szakmegnevezést ("Vegyészmérnöki", "Biomérnöki", "Környezetmérnöki"), ahol pedig nem volt egyértelmű a szak, azt "UNKNOWN"-ra állítottuk. Ezt követően a teszten elért matematika és kognitív eredményt százalékos formátumról (tizedesjegy?) formátumra változtattuk, valamint létrehoztunk 3 új oszlopot, amely a matematikateszt 3 blokkjában helyesen megválaszolt kérdések számát mutatja. Végül a vizsgálataink szempontjából irreleváns oszlopokat eltávolítottuk, valamint azon oszlopokat is kiszűrtük, amelyek értéke a többiből egyértelműen kikövetkeztethető volt.

A kumulált átlagokat, felvételi pontszámokat illetve 0. ZH eredményeket tartalmazó tábláknál egy-egy sor eltávolításán kívül nem volt szükség adattisztításra, a matematika jegyeket tartalmazó tábla esetén pedig minden hallgató esetén meg kellett határozni azt az érdemjegyet, amellyel a tárgyat elvégezte.

A táblák összetűzése Neptun alapján történt belső illesztés alkalmazásával. Azonban voltak olyan hallgatók, akiről nem minden táblában volt adat (vagy azért, mert nem írt kognitív tesztet, vagy azért, mert az első félév vége előtt elhagyta a szakot), így az összeillesztés során 10-20 fős sorvesztéssel kellett számolnunk mindkét évben. Végül a két év rendre

A kognitív eredményeket tartalmazó adattábla részletesen tartalmazott információt egyrészt minden hallgatóról (hova valósi, emelt érettségit tett-e matematikából, reál tagozatos volt-e, milyen szakra és tankörbe jár), másrészt a hallgató teszteredményéről is (mennyi idő alatt töltötte ki a tesztet, mely kérdéseket válaszolta meg jól, milyen lett a nyelvi és matekos teljesítménye), illetve tartalmazott néhány, a teszthez kapcsolódó egyéb információt is (például a teszthez használt edubase jelszó, felhasználónév). Természetesen nekünk ennyi adat nem kell, úgyhogy ebből az adattáblából jó pár irreleváns oszlopot ki kellett szűrnünk. Amelyeket meghagytunk, azok az alábbiak: a hallgató neve (ez már elég volt, hogy csak ez alapján fűzzük össze a táblákat) és Neptun kódja; emelt érettségit tett-e matematikából; reál/matematika tagozatos volt-e; szak és tankör; az elért pont és százalékos teljesítmény a nyelvi és matekos részben, valamint összességében. Problémát jelentett még, hogy a 'Szak' mezőben mindenki másképp írta be azt, hogy melyik szakon tanul, így ezt szabványosítani kellett, ha később szakok szerint akartunk vizsgálódni. Erre a feladatra külön Python kódot írtunk, és ha volt olyan mező, ahol nem tudtuk eldönteni, hogy mi lenne az oda tartozó érték (például mert 'VBK' volt odaírva), arra bevezettünk egy globális 'UNKNOWN' változóértéket, azonban szerencsére ilyenből kevés volt. Kicsit még tisztítani kellett a 'Tankör' értékeken is, de mivel ilyenből kevés volt, ezt manuálisan is meg tudtuk tenni. A 0. ZH eredményeket tartalmazó tábla szerencsére ennél jóval kisebb volt, csak a hallgató nevét, Neptun kódját, képzés nevét, illetve kódját, felvétel évét, valamint a ZH eredményt tartalmazta. Ebből értelemszerűen csak a névre és az eredményre volt szükségünk, a többi elhagyható.

A felvételi pontszámokat bontva (hozott pont, érettségi, többletpont) tartalmazó tábla

hallgatók nevén és születési dátumán kívül tartalmazott még pár, a felvételi eljáráshoz és felvételi döntéshez kapcsolódó adatot, illetve a ponthatárt. Ebből a táblából csak a név és a pontokat tartalmazó oszlopok kellettek, a többit elvethettük.

A Matematika A1a jegyeket tartalmazó táblával már több dolgunk volt, mint az előző kettő esetben. Először is minden személyhez több rekord tartozhatott, legalább egy az A1a jegyhez, lehetett még egy az A2c jegyhez (nyilván csak azoknak, akik elvégezték az A1a-t, és ott maradtak az egyetemen), illetve, ha egy korábbi, nem a végleges jegyet eredményező vizsgán egy hallgató megbukott, akkor ahhoz is tartozott egy rekord. Az attribútumok között a hallgató nevén, Neptun kódján és az osztályzatán kívül szerepelt még a felvétel éve, képzés neve, kódja, státusz ID (Aktív, elbocsátott stb.), Pénzügyi státusz (állami/önköltséges), a tantárgy neve, kódja, kreditértéke, jegy típusa, bejegyzés dátuma, illetve, hogy elismert és hogy érvényes-e az adott jegy. Ezekből az adatokból nekünk csak a hallgató nevére és jegyértékeire volt szükségünk, ráadásul olyan formában, hogy minden sor egy hallgatóhoz tartozzon, és az oszlopok a tantárgyakból szerzett jegyeket tartalmazzák. Ehhez először Pythonban kiszűrtük az irreleváns oszlopokat, majd 'crosstab'-eléssel a kívánt formára hoztuk az adatokat, ahol még ügyelni kellett arra, hogy a korábbi vizsgajegyek ne kerüljenek bele, tehát minden hallgatóhoz tárgyanként csak egy jegy tartozzon. Ezen kívül még, mivel az érdemjegyek szövegesen voltak megadva', azokat számszerűvé alakítottuk, hogy majd a későbbiekben könnyebb legyen velük dolgozni.

Egy külön adattábla tartalmazta még a kognitív eredményeknél a matekos eredményt blokkokra lebontva, amely valójában az elsőként tekintett adattáblának volt egy egyszerűsített, kevesebb attribútummal bíró változata. Ebből az adattáblából csak a hallgatók nevére, illetve a blokkonkénti teljesítményre volt szükségünk, a többit elhagytuk.

Így már rendelkezésünkre állt az összes tábla, egyenként tisztítva, és már csak az összefűzés volt hátra, amit R-ben könnyen meg tudtunk tenni, valamint még a végén rendeztük az oszlopok sorrendjét, hogy az adathalmaz logikus szerkezetű legyen. Fontos megjegyezni azonban, hogy nem minden elsőéves írt abban az évben kognitív tesztet, így összefűzés során (ahol valójában 'inner-join'-oltunk) kevesebb sorunk lett, mint ahányan abban az évben a BME VBK karára felvételt nyertek. A legvégén az így kapott adathalmaz 231 rekorddal és 21 oszloppal rendelkezett, amelyek között még esetlegesen szűrtünk különböző algoritmusok használata során.

4. Felderítő adatelemzés a két évben

5. Prediktív analitika

5.1. Modellek és metodológia

Következő lépésként azt vizsgáltuk, hogy a két évben külön a bemeneti adatok alapján mennyire pontosan lehet előrejelezni egyes teljesítménymutatók értéket, illetve hogy a predikciókra nézve a különböző bemeneti változók milyen és mekkora szereppel bírnak. Ehhez többféle modellen többféle *gépi tanuláson alapuló osztályozó algoritmus* használatára volt szükség. A kutatás során kétféle eredmény alaposabb vizsgálatára összpontosítottunk mindkét évben: a "Matematika A1a - Analízis" tárgyból kapott érdemjegyre illetve az első félév végén megállapított kumulált tanulmányi átlagra. Ezen feladatok a felügyelt gépi tanulási problémák közé esnek, ahol a kitüntetett célváltozót szeretnénk a többi bemeneti változóval előrejelezni. Ekkor a gépi tanulás során a teljes adathalmazt tanító és teszthalmazra bontjuk, a tanító adathalmazon betanítjuk az algoritmusokat úgy, hogy a tanítóhalmazbeli adatok célváltozóját minél pontosabban prediktálják. Ezt követően valamilyen metrika szerint kiválasztjuk a betanított algoritmusok közül a legjobbat, majd az általánosítóképesség ellenőrzése végett a teszthalmazon is visszamérjük a teljesítményét.

A két vizsgált probléma közül az előbbi alapvetően egy osztályozási probléma öt osztállyal, míg az utóbbi egy regressziós feladat. Mivel azonban viszonylag kevés adat állt a rendelkezésünkre, ezért az érdemjegyek prediktálásánál az adatrekordok esetén a pontos érdemjegy helyett érdemjegy csoportok előrejelzésére koncentráltunk. A matematika érdemjegycsoportok prediktálására így kétféle modell került felvázolásra: egy *3 csoport modell*, illetve egy *2 csoport modell*.

A 2 csoport modell esetén a két csoportot a {5,4,3} illetve {2,1} osztályok adják, míg a 3 csoportnál az osztályok {5,4}, {3,2} illetve {1} módon alakultak. Az előbbi esetben az osztályok intuitívan a lemorzsolódási veszélyeztetettség szerint formálódtak, míg az utóbbiban egy általánosabb "jól teljesítő", "rosszul teljesítő", "lemorzsolódott" csoporthármast kívántunk elérni. Mindkét modell esetén külön vizsgáltuk a teljesítményt összes hallgatóra vonatkozóan illetve szétbontva vegyészmérnök és biomérnök hallgatókra egyaránt (a környezetmérnökökről nem állt rendelkezésre elég adat, így őket a szakonkénti bontásban kihagytuk).

Az egyes modellek és almodellek esetén a tanítás előtt főkomponens analízist (Principal Component Analysis) is alkalmaztunk. Az eljárás lényege, hogy az attribútumok súlyozott kombinációjával új attribútumokat hozunk létre, kevesebb számút mint az eredeti attribútumok száma, oly módon, hogy az információveszteség minimális legyen. Ennek következtében az adatok egy kisebb dimenziós térbe transzformálódnak át, s az ehhez használt megfelelő súlyozás pedig az attribútumok tapasztalati kovarianca mátrixának

segítségével határozható meg. PCA használata során az algoritmusok gyorsabban tanulnak a kisebb dimenziószám miatt, és olykor jobb eredményt is érnek el. Jelen kutatásban másrészt azért is döntöttünk a PCA alkalmazása mellett, mert a korábbi, ugyanezen problémakört vizsgáló, általunk átnézett tanulmányok nem használtak PCA-t még nagyobb attribútum szám esetén sem, ugyanakkor mi a korai tanító-tesztelési fázisnál azt tapasztaltuk, hogy az áttranszformált adaton jobban teljesítenek az osztályozó algoritmusok, így potenciálisan megéri alkalmazni.

5.2. Osztályozó algoritmusok

5.2.1. Lineáris regresszió

A lineáris regresszió jellegéből adódóan alapvetően folytonos célváltozó prediktálására alkalmas, ugyanakkor kategorikus, ordinális címkéjű adatok osztályozására is fel lehet használni. Az algoritmus lényege, hogy a magyarázóváltozók mindegyikéhez egy-egy súlyt rendelünk, majd az adatpont attribútumértékeinek vesszük a súlyokkal való súlyozott összegét, és esetleg egy bias tagot hozzáadva az így kapott szumma lesz a prediktált címkeérték. A cél a tanítás során a súlyok optimális megtalálása, miközben a tanítóhalmazon minimalizáljuk a predikciós hibát.

5.2.2. Naive Bayes

A Naive Bayes algoritmus működése mögött álló alapelv az, hogy feltesszük az attribútumok feltételes függetlenségét, amennyiben a célváltozó értéke ismert. Osztályozás során azt vizsgáljuk, hogy mely címkeérték mellett a legnagyobb a valószínűsége annak, hogy az adott adatrekord attribútumai éppen a felvett értékeket kapták. A megfelelő valószínűségeket a tanítóhalmazbeli adatok attribútumértékeinek különböző címkék melletti relatív gyakoriságai adják.

5.2.3. Gradient Tree Boosting

A Gradient Boosting algoritmus egy Ensemble típusú osztályozó, amelyek lényege, hogy sok gyenge teljesítményű prediktor (*weak learner*) eredményét felhasználva hoz egy erős predikciót.

5.2.4. Logisztikus regresszió

A logisztikus regresszió alapvetően bináris osztályozási problémák megoldására alkalmas, de kiterjeszthető másfajta feladatok megoldására is. Lényege, hogy a lineáris regresszióhoz hasonlóan az adatrekord attribútumértékeinek súlyozott összegét használjuk egy szigmoid[1] függvény bemeneteként, amely azt utána leképzi az $(0, 1)$ intervallumra, és amennyiben

[1] A szigmoid függvény: $\sigma(z) = \frac{1}{e^{-z}}$, ahol $z = x_1w_1 + x_2w_2 + \cdots + x_nw_n + b$ a súlyozott összeg.

az output 0.5-nél nagyobb, úgy a pozitív osztályba soroljuk az adott rekordot.

5.2.5. SVM

Az SVM (Support-Vector-Machine) egy lineáris szeparálást használó bináris osztályozó algoritmus, amely egyéb problémákra is kiterjeszthető. Lényege, hogy különböző magfüggvénye segítségével az adatrekordokat egy magasabb dimenziós térbe képzi le, ahol olyan szeparáló hipersíkot keres, amely maximalizálja a vele párhuzamos, adatpontot nem tartalmazó térrészt. A cél a megfelelő magfüggvény és a szeparáló hipersík megtalálása, amellyel a különböző címkéjű adatpontok lineárisan szeparálhatóak.

5.3. Implementálás és optimalizálás

A 2 és 3 csoport modellnél a célváltozó-értékeket rendre 1, 0-ra illetve 2, 1, 0-ra módosítottuk. Az osztályozó algoritmusok közül a Naive Bayes, kNN és Gradient Tree Boosting alapból jól kezeli a kategorikus változókat, az SVM-nél valamint a regressziós eljárásoknál viszont minimális változtatásra volt szükség. Az SVM és logisztikus regresszió mivel csak bináris osztályozásra alkalmas, így ott *One-Vs-Rest* elvű osztályozást használtunk, lineáris regressziónál pedig az címkeértékek ordinális jellege miatt egyszerűen a prediktált értéket kerekítettük a legközelebbi csoportcímkére. Kumulált átlag előrejelzésénél csupán lineáris regressziót alkalmaztunk.

Valamennyi modellnél a teljes adathalmazt felbontottuk tanító- és teszthalmazra 70-30 arányban, továbbá a két adathalmazban a folytonos változók kvantilis alapú 0-1 skálázásnak lettek alávetve, hogy a távolság alapú osztályozók jobban teljesítsenek. Az algoritmusok hiperparamétereinek optimális megválasztására 5-szörös kersztvalidációt alkalmaztunk. Fontosabb optimalizációs lépések csak a 2-3 csoport modellek esetén történtek. A Naive Bayes és lineáris regresszió algoritmusoknál jellegük és/vagy implementálásuk végett nem történt optimalizálás, a többi algoritmus esetén az alábbi hiperparaméterek kerültek változtatásra:

- **kNN**:
  - Távolságmérték (euklédeszi, Mahalanobis)
  - Szomszédok száma (5 és 15 között változtatva)
  - Szomszéd címkeértékének súlyozása (uniform, távolság reciprokra, távolság reciproknégyzete)
- **Gradient Tree Boosting**:
  - Tanulóráta (0.01 és 5 között változtatva 0.1-es lépésközzel)
  - Boosting fázisok száma (5 és 100 között változtatva 5-ös lépésközzel)
  - Vágási feltétel (négyzetes hiba, Friedman MSE)
  - Maximális famélység (3,4 és 5)

- **Logisztikus regresszió**:
  - Regularizációs paraméter (0.05 és 5 között változtatva 0.05-ös lépésközzel)
  - Optimalizálási módszer ("SAG", "SAGA")

- **SVM**:
  - Regularizációs paraméter (0.1 és 5 között változtatva 0.1-es lépésközzel)
  - Magfüggvény (lineáris, legfeljebb 3-ad fokú polinomiális, RBF)

A PCA esetében a főkomponensek számát 2 és 8 között változtattuk 2-es lépésközzel, az algoritmusokat minden főkomponensszám mellett 5-szörös keresztvalidációval optimalizáltuk, majd a legjobb modelleket visszamértük a teszthalmazon.

6. Modellek kiértékelése

			Osztályozó algoritmusok				
			Grad. Boost.	Naive B.	Log. reg.	SVM	Lin. reg.
3 csoport	Összesített	2 PC	66.67	62.67	52.00	58.67	61.33
		4 PC	**70.67**	60.00	57.33	53.33	65.33
		6 PC	65.33	62.67	54.67	68.00	64.00
		8 PC	64.00	68.00	53.33	62.67	64.00
	Szakonként	2 PC					
		4 PC					
		6 PC					
		8 PC					
2 csoport	Összesített	2 PC	59.42	69.57	69.57	72.46	69.57
		4 PC	59.42	71.01	**73.91**	69.57	73.91
		6 PC	63.77	69.57	73.91	71.01	73.91
		8 PC	68.12	69.57	71.01	73.91	72.46
	Szakonként	2 PC					
		4 PC					
		6 PC					
		8 PC					

1. táblázat. A 2019-es adatsor eredményei

			Osztályozó algoritmusok				
			Grad. Boos.	Naive B.	Log. reg.	SVM	Lin. reg.
3 csoport	Összesített	2 PC	53.74	72.97	67.75	**73.30**	56.80
		4 PC	61.86	70.49	65.82	66.63	51.91
		6 PC	68.27	68.53	67.73	70.80	64.00
		8 PC	66.31	70.20	67.21	67.17	64.00
	Szakonként	2 PC					
		4 PC					
		6 PC					
		8 PC					
2 csoport	Összesített	2 PC	69.45	67.89	69.77	69.77	67.89
		4 PC	72.74	72.90	78.23	76.51	77.91
		6 PC	67.89	72.90	79.80	79.80	76.19
		8 PC	68.05	74.62	79.96	**83.24**	79.63
	Szakonként	2 PC					
		4 PC					
		6 PC					
		8 PC					

2. táblázat. A 2021-es adatsor eredményei

Hivatkozások