ETDK-dolgozat

Bereczki-Orbán András

XXVI. reál- és humántudományi Erdélyi Tudományos Diákköri Konferencia (ETDK)

Informatika I.: elméleti módszerek, kísérletek, szimulációk szekció

Kolozsvár, 2023. május 18–21.

Kvízkérdések téma szerinti kategorizálása és nehézségének becslése

Szerzők:

Bereczki-Orbán András

Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar, Informatika szak, III. év

Témavezetők:

dr. Bodó Zalán, egyetemi docens

Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar

Tartalomjegyzék

1. Bevezető **1**

2. Kvízkérdések és kvízjátékok **2**

2.1. Jeopardy! 3

2.2. Legyen Ön is milliomos! 3

2.3. Honfoglaló 4

3. Osztályozó algoritmusok **6**

3.1. Naiv Bayes 6

3.2. Szupportvektor-gép 7

3.3. Osztályozó módszerek kiértékelése 8

4. Természetes nyelvű szövegek feldolgozása **9**

4.1. A jellemzővektor felépítése 9

4.1.1. TF tokenizálás 9

4.1.2. TF-IDF tokenizálás 10

4.2. Jellemzők kiválasztása 10

4.2.1. Chi négyzet kiválasztás 11

4.2.2. Mutual information kiválasztás 11

4.2.3. Stopszavak 12

4.3. A jellemzővektor kibővítése 12

5. Az adathalmazok **14**

5.1. Témakörökkel ellátott kvízkérdések adathalmaza 14

5.2. Nehézséggel ellátott kvízkérdések adathalmaza 15

5.3. Teszteléshez használt adathalmaz 15

6. Az elvégzett kísérletek **16**

6.1. Témakör szerinti osztályozás 16

6.2. Nehézség szerinti osztályozás 18

7. Következtetések és továbbfejlesztési lehetőségek **20**

1. Bevezető

Középiskolában barátaimmal több olyan alkalmazás használtunk és játékot játszottunk, amiben a beépített elemek mellett a játékosok által készített tartalmat is fel lehetett használni. Így a játékot saját baráti körünkre tudtuk szabni. Az általunk játszott játékok közül egy kivétel volt ilyen szempontból: a Honfoglaló. Innen jött az ötlet egy saját Honfoglaló-szerű kvízjáték megalkotására. Ez a projekt középiskolában még igen nagynak bizonyult, ám azóta több változata is használható volt. Most a negyedik verziója készül, melyben szükségessé vált a kvízkérdések téma szerinti kategorizálása és nehézségének becslése.

A dolgozat célja tehát olyan gépi tanuláson alapuló tanulási modell alkalmazása, melyek a kvízkérdések témakörét és nehézségét automatikus módon meghatározzák. Az így megszerzett információkat felhasználhatjuk akár az oktatás területén, illetve szórakozásra. Egy ilyen tanulási modell megalkotásának folyamatát és tesztelését írja le bővebben ez a dolgozat.

A dolgozat elején meg fogjuk határozni, hogy mi is egy kvízjáték, majd pár példát nézünk ezekre. A 3. fejezetben osztályozó algoritmusokról lesz szó, ezután a természetes nyelvű szövegek feldolgozásának, azaz a kvízkérdések feldolgozásának módszereit ismerteti a 4. fejezet. Az általunk elvégzett kísérleteket a 6. fejezet fogja leírni, majd a dolgozat utolsó részében levonjuk a következtetéseket, illetve felsorolunk néhány továbbfejlesztési lehetőséget.

2. Kvízkérdések és kvízjátékok

A kvízkérdések és az igény az elme próbatételére valamilyen formában mindig jelen voltak az ember történetében. Ezt bizonyítja a találós kérdések létezése. A találós kérdések is mind formájukban mind szerepükben hasonlítanak a mai kvízkérdésekre. Mindkét kérdésfajtánál csupán egy helyes válasz adható a feltett kérdésre, ám a találós kérdések és rejtvények esetében ez a válasz nem egyértelmű. Sok esetben a szokványostól eltérő gondolkodásmód vezet a megoldáshoz.

Ilyen találós kérdésekkel gyakran találkozunk az irodalomban. Az egyik közismert találós kérdés egy ókori görög mondából származik. A monda szerint a Szphinx Thébai városának lakóit tartotta rettegésben és aki nem tudott válaszolni a kérdésére azt elpusztította. A kérdése a következő volt: Reggel négy lábon, délben két lábon, este pedig három lábon jár [10]. A válasz a Szphinx kérdésére természetesen az ember. Nem kell azonban más kultúrák mondáiban keresnünk. Hasonló rejtvényekkel és furfangos kérésekkel gyakran találkozhatunk a magyar népmesék szövegeiben is. A kékfestőinas című mesében, például, a török vezér három kérdésére kell helyesen válaszolnia a királynak.

A meséken és mondákon kívül, a XIX. században megjelent az igény általános műveltségi kvízkérdések iránt. Ennek az igénynek a kielégítése végett elkezdtek kérdésgyűjtemények megjeleni. Az egyik első ilyen könyv az angol nyelven megjelent „QUIZZISM; AND ITS KEY. Quirks and Quibbles From Queer Quarters” [11]. A könyv 600 számozott, általános műveltségi kérdést tartalmaz természetesen válaszokkal együtt. A könyv céljának tudományok iránt az érdeklődés felkeltését tűzte ki.

A magyar értelmező szótár szerint [2], a kvíz vagy a kvízjáték szellemi vetélkedőt jelent. Ez összhangban van az eddig meghatározott értelmezésekkel és célokkal. A kvíz tehát az elme próbatétele, melynek célja a tudomány iránti érdeklődés felkeltése és alkalmanként kielégítése, ami vezethet a szokványostól eltérő gondolkodásmódhoz.

Kvízjátéknak nevezzük azokat az eseményeket, vetélkedőket és szórakoztató játékokat, amelyekben a játékosoknak több kvízkérdést kell megválaszolniuk egy adott szabályrendszer szerint. Ezekben a kvízjátékokban általában a nyertes az, aki a legtöbb helyes választ tudta adni a feltett kérdésekre. Sok kvízjáték stratégiát igényel, ezzel is nagyobb próbát, nyomást helyeznek a játékosok elméjére. A játékosoknak sokszor lehetőségük van a feltett kérdéseket befolyásolni például a kérdés nehézségének, témakörének vagy fajtájának megválasztásával. Emellett a válaszadásban is gyakran segítség kérhető. Ilyen például a megadott válaszlehető-

ségek csökkentése vagy a kérdés szövegének a kibővítése. Több sikeres kvízjáték készült, ám most hármat vizsgálunk meg közelebbről. Mindhárom kvízjáték az előbb felsorolt tulajdonságok közül választott játékmenete központjává.

2.1. Jeopardy!

A Jeopardy![1] egy 1964-ben indult amerikai tévéműsor, kvízjáték. A vetélkedőben minden adásban 3 résztvevő versenyzik a legnagyobb pénzösszeg megszerzéséért. A játék központi eleme egy táblázat, aminek celláiban a feladványok találhatóak. A táblázat oszlopaiban egy témakörhöz tartozó kérdések vannak, míg egy sor a nyerhető pénzösszeget tartalmazza. A játék során egy résztvevő ki kell válaszon egy témakört és egy pénzösszeget, ezzel meghatározva egy feladványt. Ezután felolvassák a feladvány megoldását és a játékosoknak ki kell találnia a megadott válaszra találó és témakörbe illő kérdést. A játékos, aki a leghamarabb képes a helyes kérdést megtalálni az nyeri a megnevezett pénzösszeget.

Ebben a vetélkedőben jól megfigyelhető a játékosok általi kérdésbefolyásolás, hiszen a feltett kérdések nehézségét és témakörét is a játékosos szabják meg. Természetesen, egy idő után a lehetőségek csökkennek, de mindig van a játékosok kezében döntési lehetőség.

FOLYÓK ÉS TAVAK	TALÁLMÁNYOK	ÁLLATOK	IDEGEN KONYHA	SZÍNÉSZEK	SZÁMOT KÉREK
100	100	100	100	100	100
200	200	200	200	200	200
300	300	300	300	300	300
400	400	400	400	400	400
500	500	500	500	500	500

1. táblázat. A Jeopardy! játék egy lehetséges táblázata.

2.2. Legyen Ön is milliomos!

A Legyen Ön is milliomos[2] (angolul *Who Wants to Be a Millionaire?*) egy brit tévés kvízjáték, melyet története során több országban is adaptáltak. Első adását 1998-ban, míg magyarországi változatát 2000-ben sugározták.

[1] https://www.jeopardy.com/
[2] https://hu.wikipedia.org/wiki/Legyen_%C3%96n_is_milliomos!

A játékmenet egyszerű volt. A játékosnak helyesen kellett válaszolnia 15 darab, egyre nehezedő kérdésre. Ahogyan a játékos haladt előre a kérdések megválaszolásában, úgy növekedett a helyes válasz értéke. A játékos dönthetett úgy, hogy nem játszik tovább és elviszi a nyereményét vagy megpróbál egy újabb kérdésre válaszolni és minél nagyobb pénzösszeget összegyűjteni. A kérdések feleletválasztósak voltak, minden kvízkérdésre négy lehetséges válasz közül lehetett választani. Minden játékosnak 3 segítség állt rendelkezésére:

1. A válaszlehetőségek felezése
2. Telefonos segítség
3. A közönség megkérdezése.

A vetélkedő nehézségéből adódóan nagy hangsúly került a segítségek használatára. A játékos a megfelelő segítséget kellett jókor felhasználja a sikeres továbbjutáshoz, hiszen rossz válasz esetén nyeremény akár el is veszhetett. A játékosnak lehetősége volt ezeknek a segítségeknek a felhasználásával megbizonyosodni válasza helyességéről, de akár új gondolatmenetre is térhetett felhasználásukkal.

2.3. Honfoglaló

A Honfoglaló[3] egy 2002-ben indult magyar fejlesztésű kvízjáték. Indulásakor egy böngészőből elérhető játék volt, ám azóta mobilos és társasjáték formájában is elérhető. 2017-ben egy háromévados tévéműsor készült belőle[4].

A játék menete egyszerű: a játékosok célja a legtöbb győzelmi pont megszerzése, úgy, hogy közben nem veszti el saját várát. A játéktábla Magyarország térképe, 19 megyére felosztva. Minden játékos egy megyét válasz az ő várának, ez 1000 győzelmi pontot ér majd a játék végén, míg minden egyéb terület 200-at. A játékosos először a még el nem foglalt területeket osztják fel egymás között, úgy, hogy helyesen válaszolnak egy feleletválasztós kérdésre, majd megjelölnek egyet a saját területükkel szomszédos megyék közül. Miután minden megye valamelyik játékosé, utána hat fordulón keresztül a minden játékosoknak van lehetősége az ellenségek megyéit elfoglalni. Az elfoglalni kívánt megye a támadó játékos területével szomszédos kell legyen. Azé a játékosé lesz a megtámadott terület, aki egy feleletválasztós kérdésre helyesen tudott válaszolni. Amennyiben a támadó és a védő játékos is helyes választ adott, egy

[3] https://triviador.com/
[4] https://hu.wikipedia.org/wiki/Honfoglal%C3%B3_(kv%C3%ADzm%C5%B1sor)

1. ábra. A Honfoglaló játék egy állapota 2016 környékén. (Saját forrás)

tippelős kérdés következik. Ebben az esetben a tipp pontossága és a válaszidő is szerepet játszik a „csata” eldöntésében.

A kvízjáték fontos eleme a stratégia, mivel a játékososok úgy próbálnak meg területekhez jutni, hogy a saját várukat bizonyságban tartsák. A játékosok fel kell mérjék egy esetleges rossz válasz vagy elvesztett terület kockázatát a saját céljaik elérése érdekében. Ezen kívül a játék során a játékosok egymással egyességei is formálni fogják a játékosos stratégiáját, ezzel változtatva a játék végeredményén.

3. Osztályozó algoritmusok

A szövegosztályozás célja és szöveg vagy dokumentum egy kategóriába való besorolása, bizonyos rendszerezés szerint. Ezt a folyamatot nevezhetjük felcímkézésnek is, mivel a használt kategóriákat elképzelhetjük címkeként is. Ha jelenlevő kategóriákról azt feltételezzük, hogy egymástól függetlenek, végezhetünk egycímkés osztályozást, amikor minden dokumentumot egy osztályba kell besorolnunk. Ennek a speciális esete a bináris osztályozás, amikor egy osztály és az osztályon kívüli szövegeket akarjuk szétválasztani egymástól. Emellett léteznek többcímkés osztályozó algoritmusok is, ilyenkor egy szöveg több címkével is rendelkezhet [1].

Egy osztályozó algoritmus betanításához a megfelelő metaparaméterek beállítása mellett szükséges egy tanulási adathalmaz. Ez az adathalmaz olyan dokumentumokat, esetünkben kvízkérdéseket tartalmaz, melyeknek osztályát ismerjük. Mivel ezek az osztályozó algoritmusok általánosak, nem csak szövegek felcímkézésére szolgálnak, ezért nem szavakkal, hanem jellemzőkkel foglalkoznak. A tanulási adathalmazból tehát ki kell emeljük a jellemzőket. Ezzel a jellemző előállítással foglalkozik a 4. fejezet, most csak azt kell tudnunk, hogy egy osztályozó algoritmus tanulásához két bemeneti paraméter szükséges. Egy jellemzővektor halmazt, amit egy mátrix formában tárolunk el és egy a jellemzővektoroknak megfelelő címkéket vektor alakban.

3.1. Naiv Bayes

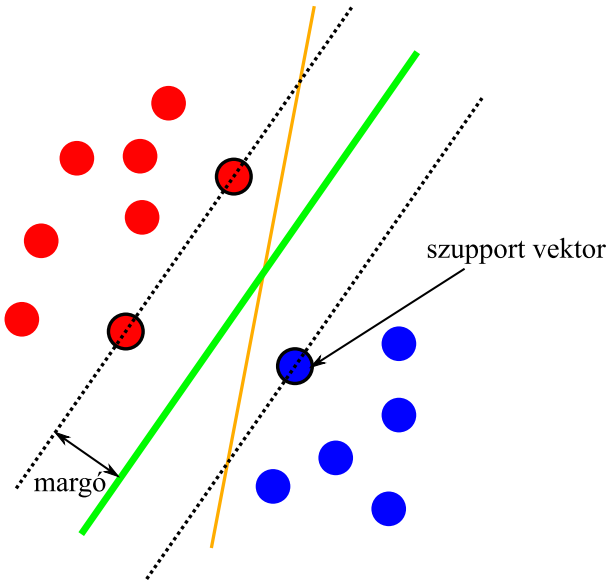
Legyen F egy adathalmazban fellelhető össze jellemző halmaza, és $f \in F$ egy jellemző (*feature*). Jelölje F_n az adathalmaz n. elemének jellemzőinek értékét. Hasonlóképpen, legyen L a jelenélevő osztályok / címkék halmaza, $l \in L$ egy címke (*label*) és N az osztályozandó szövegek száma. Ebben az esetben a Bayes tételre épülő Naiv Bayes osztályozás a következő módon írható fel az adathalmaz egy n. tagjára [3, 4. fejezet.]:

$$l_n^* = \arg\max_{l \in L} P(l) \cdot \prod_{f \in F_n} P(f|l) \cdot$$

A képletben a $P(l)$ a címke előfordulásának előzetes valószínűsége. Esetünkben nem rendelkezünk előzetes megfigyelésekkel, ezért az adathalmazban szereplő l címkéjű dokumentumok számát osztva N-el vettük $P(l)$ k-ként. Az osztályozó algoritmus a naiv jelzőt eztért kapta, mivel azt feltételezzük, hogy a jellemzők egymástól teljesen függetlenek.

3.2. Szupportvektor-gép

Szupportvektor-gép (*support vector machine, SVM*) osztályozás során minden dokumentumot – a jellemzői alapján – egy vektornak képzelünk el. Amennyiben egy dokumentumhoz tartozó vektor dimenziója p, a feladatunk a dokumentumokhoz tartozó adatpontokat egy $(p-1)$ dimenziójú hipersíkkal elválasztani egymástól az adatpontok osztálya szerint. Mivel több ilyen elválasztó hipersík is létezhet, a szupportvektor-gépeken alapuló osztályozás azt a hipersíkot keresi/választja, melynek margója a legnagyobb. A margó az elválasztó hipersík távolsága a legközelebbi adatpontokhoz. Ezeket a legközelebbi adatpontokat, vektor reprezentációjuk miatt hívjuk szupportvektoroknak vagy tartóvektoroknak [7, 15. fejezet.]. A 2. ábrán a szupportvektorokat a feketével kiemelt pontokkal jelöltük.



2. ábra. Szupportvektor-gép osztályozás két dimenzióban. A narancssárga és a zöld vonal is helyesen elválasztja a két osztályt, de a zöldnek nagyobb a margója-

Az osztályozó algoritmus tulajdonságai alapján csupán két osztályt tud egymástól megkülönböztetni. Amennyiben több mint két osztállyal dolgozunk, szükséges lesz annyi SVM osztályozót betanítani, ahány osztályunk van: ilyenkor a második, azaz negatív osztályban minden, egyéb címkével rendelkező dokumentum kap helyet. Egy ilyen modell tesztelésénél azt a címkét fogjuk választani, amelyik osztályhoz tartozó SVM osztályozó hipersíkja és az adatpont közti táv a legnagyobb. Ezt *one-versus-rest* osztályozásnak nevezzük.

		Valós címkék			
		l_1	l_2	$\dots$	l_n
Jósolt címkék	l_1	m_{11}	m_{12}	$\dots$	m_{1n}
	l_2	m_{21}	m_{22}	$\dots$	m_{2n}
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	l_n	m_{n1}	m_{n2}	$\dots$	m_{nn}

3. ábra. Tévesztési mátrix n osztállyal.

3.3. Osztályozó módszerek kiértékelése

Ahhoz, hogy el tudjuk dönteni egy osztályozó algoritmus teljesítményét, jóságát, szükséges a már betanított algoritmusokon teszteket végezni. Ilyenkor rendelkeznünk kell egy tesztelési adathalmazzal, melyben a tanulási adatokhoz hasonló dokumentumok vannak, de egy sem szerepelhet a tanulási adathalmazban. Természetesen ebben az adathalmazban is ismernünk kell a dokumentumok címkéit. Ezután a betanított algoritmus kísérletet tesz minden, a tesztelési adathalmazban levő dokumentum felcímkézésére. Az így kapott adatokból felírhatunk egy ún. tévesztési mátrixot (3. ábra).

Egy tévesztési mátrixban minden sor megfelel egy jósolt címkének, míg az oszlopok a valós címkéket mutatják. Ezzel a felosztással, az osztályozó algoritmus lefuttatása után a tesztelési adathalmazból minden dokumentumot be tudunk illeszteni a tévesztési mátrix egy cellájába. Egy cellában található dokumentumok számából a következő jósági metrikákat képezhetjük: pontosság (*precision*), fedés (*recall*), F_1 érték [3, 4.7. fejezet.]. Egy cellában található dokumentumok számát jelölje m_{ij}, ahol i a mártix egy sorára utal (egy jósolt címkére), míg j a mátrix egy oszlopára (egy valós címkére). Amennyiben az adathalmazban levő osztályok száma n akkor egy l_k osztályra a jósági értékek felírhatóak a következőképen:

$$pontosság(l_k) = \frac{m_{kk}}{\sum_{i=1}^{n} m_{ki}}$$

$$fedés(l_k) = \frac{m_{kk}}{\sum_{i=1}^{n} m_{ik}}$$

$$F_1(l_k) = \frac{2 \cdot pontosság(l_k) \cdot fedés(l_k)}{pontosság(l_k) + fedés(l_k)}$$

Ha az adathalmazban levő összes osztályra kiszámítjuk valamelyik jósági értéket, akkor belőlük az egész osztályozó algoritmus jóságát megkaphatjuk átlagszámítással. Súlyozott átlagot is számíthatunk amiben a súlyok egy osztály valós címkéinek száma lesz.

4. Természetes nyelvű szövegek feldolgozása

Ahhoz, hogy megfelelő módon tudjuk osztályozni a kvízkérdéseket, szükséges a természetes nyelven megfogalmazott szövegüket feldolgozzuk és az osztályozó algoritmusok számára felhasználható alakba hozzuk. Ezt az alakot már láthattuk az osztályozó algoritmusokat bemutató 3. fejezetben. Tehát a célunk minden természetes nyelven megforgalmazott szöveghez egy őt jellemző vektort rendelni.

Egy kvízkérdésnél vagy bármilyen természetes nyelvű szövegnél nem nehéz kitalálnunk, hogy a szöveget felépítő szavak és n-gramok lesznek a legfontosabb jellemzőink. Egy n-gram egy n egymásutáni szövegrészt jelent [3]. Tehát egy bigram (2-gram) két egymásutáni szó kapcsolata, egy trigram (3-gram) pedig háromé.

Azt feltételeztük a jellemzővektor (*feature vector*) felépítésekor, hogy a szavak sorrendje nem rendelkezik számottevő információval és hogy a szövegben előforduló szavak és n-grammok egymástól függetlenek. Ezt a módszert *bag-of-words* (szózsák) reprezentációnak hívják [3].

4.1. A jellemzővektor felépítése

4.1.1. TF tokenizálás

A legegyszerűbb módon úgy építhetjük fel a jellemzővektort, ha megszámoljuk a szövegben előforduló szó vagy n-gram előfordulásait a szövegben. Esetünkben egy kvízkérdéshez kell egy jellemzővektort rendelnünk. Amennyiben egy adathalmazban levő összes kvízkérdésre elvégezzük a megfeleltetést lehetőségünk lesz az egész adathalmazt egy kétdimenziós mátrixként feldolgozni. Ebben a mátrixban egy oszlop megfelel egy jellemzőnek, míg egy sor egy osztályozandó szövegnek.

Az adathalmaznak megfelelő mátrix felépítése során alkalmazhatunk normalizálást is. A mátrixot normalizálhatjuk oszlopok (jellemzők) szerint, vagy ami számunkra hasznosabb, sorok (osztályozandó szövegek) szerint. Az egyik leggyakrabban használt ilyen módszer az L_2 normalizálás. Emiatt nem a puszta előfordulás, hanem a gyakoriság marad a jellemzővektorban. Így arányosabban fel tudunk dolgozni különböző hosszúságú szövegeket [3].

4.1.2. TF-IDF tokenizálás

Egy jellemzővektor felépítésének egy másik megközelítése a TF-IDF tokenizálás [7]. Ez a fajta jellemzővektor felépítési módszer minden szóhoz rendel egy relevanciaértéket, majd súlyoz vele. Ahogyan neve is mutatja, egy jellemző értékének kiszámítása két részletből áll. Az első a $TF(f)$ (*term frequency*, jellemző előfordulás). Ezt már láthattuk az egyszerű jellemzővektor felépítésénél, csupán a szövegben előforduló szavak megjelenési számára utal.

A második komponens az $IDF(f)$ (*inverse document frequency*, inverz dokumentum-előfordulás). Amennyiben megszámoljuk, hogy egy adathalmaz hány tagjában szerepel egy adott jellemző, akkor megkapjuk az adott jellemző dokumentum-előfordulását $DF(f)$ az adathalmazra nézve. Amennyiben N az adathalmazbeli dokumentumok számát jelöli, ha elvégezzük a következő műveletet $IDF(f) = \log \frac{N}{DF(f)}$, megkapjuk az inverz dokumentum-előfordulást. Ezt az IDF lesz egy jellemző relevanciaértéke. A két kiszámított tagot összeszorozva: $TFIDF(f) = TF(f) \cdot IDF(f)$ megkaphatjuk a jellemzővektor értékeit.

4.2. Jellemzők kiválasztása

Amennyiben az előzőleg felsorolt módszerek egyikével előállítjuk a jellemzővektorokat, észrevehetjük, hogy ezek a vektorok nagyon nagyok lesznek a sok egyedi szó miatt. Emellett ezekben a vektorokban a legtöbb érték 0. A 3. fejezetben leírt osztályozó algoritmusok időbonyolultsága mindig összefüggésben áll a bemeneti mátrix méretével, ebből következik, hogy amennyiben csökkenteni tudjuk a bemeneti mátrix méretét, azaz a jellemzők számát, akkor az osztályozó algoritmusainkat jelentősen gyorsítani tudjuk. Mindezt a gyorsítást úgy akarjuk elérni, hogy az osztályozás minőségben ne romoljon. Szeretnénk tehát csupán a legtöbb információt hordozó jellemzőket, szavakat megtartani ezáltal a bemeneti adatokban jelenlevő zajt is kiszűrni. A zaj kiszűrése az osztályozó algoritmus pontosságának növekedéséhez is hozzájárulhat. Ennek érdekében minden jellemzőhöz / szóhoz f rendelni fogunk egy relevanciaértéket. Nevezzük el ezt $Rel(f)$-nek. Miután kiszámítottuk minden jellemző relevanciaértékét kiválogathatjuk a legnagyobbal rendelkezőket így csökkentve a jellemzővektorok méretét. Használhatnánk a relevanciaértékek gyanánt a már megismert $DF(f)$-et vagy $IDF(f)$-et, ám ezeknél vannak jobban működő szűrők is.

4.2.1. Chi négyzet kiválasztás

A χ^2 statisztika egy jellemző és egy kategória közti függőséget vizsgálja. Minden osztály-jellemző párosra felépíthetünk egy táblázatot, amiben megjeleníthetjük a kettőjük kapcsolatát egy adott adathalmazban. Jelölje L az előforduló osztályok halmazát és a l egy osztályt ebből a halmazból. Ebben az esetben a kapcsolatot leíró táblázat, M_{fl} , a következő:

	f szerepel a szövegben	f nem szerepel a szövegben
l a szöveg osztálya	A_{fl}	C_{fl}
l nem a szöveg osztálya	B_{fl}	D_{fl}

Ezt a jelölést használva leírhatjuk a χ^2 statisztika kiszámítási módját, majd kiszámíthatjuk egy jellemző relevanciaértékét egy adathalmazban: [12]:

$$\chi^2(f,l) = \frac{N \cdot (A_{fl}D_{fl} - C_{fl}B_{fl})^2}{(A_{fl} + C_{fl}) \cdot (B_{fl} + D_{fl}) \cdot (A_{fl} + B_{fl}) \cdot (C_{fl} + D_{fl})}$$

$$Rel(f) = \sum_{l \in L} \frac{A_{fl} + C_{fl}}{N} \chi^2(f,l)$$

4.2.2. Mutual information kiválasztás

A mutual information statisztika $MI(f,l)$, hasonlóan a χ^2 statisztikához egy jellemző és egy osztály közötti függőséget vizsgálja. Amennyiben egy jellemző előfordulása teljesen független egy osztálytól a statisztika értéke 0 lesz. Hátránya, hogy érzékeny a ritkán előforduló jellemzőkre. Az előző alfejezetben használt jelölést használva a relevanciaérték kiszámítható a következő képen [12]:

$$MI(f,l) = \log \frac{N \cdot A_{fl}}{(A_{fl} + C_{fl}) \cdot (A_{fl} + B_{fl})}$$

$$Rel(f) = \sum_{l \in L} \frac{A_{fl} + C_{fl}}{N} MI(f,l)$$

4.2.3. Stopszavak

Egy másik megközelítés a jellemzővektor méretének csökkentésére az úgynevezett stopszavak használata. A stopszavak a jellemzők egy listája, esetünkben egy szólista, amelyeket teljes mértékben figyelmen kívül hagyunk a feldolgozás és ebből következőleg, az osztályozás során. Amennyiben a feldolgozandó kvízkérdés szövegében találkozunk egy stopszóként megjelölttel, akkor azt egyszerűen törölni fogjuk belőle. A stopszavak listája általában egy nyelv leggyakrabban előforduló, így számunkra valószínűleg kevés információértékkel rendelkező szavakat tartalmaz. Nekünk egy ilyen stop szólistában a *yes, no, true, false* szavak jelenléte volt a legfontosabb, mivel ezek gyakori válaszlehetőségek.

4.3. A jellemzővektor kibővítése

A kvízkérdések rövidségéből adódóan a belőlük épített jellemzővektorok kevés jellemzőt tartalmaznak az összes jelenlevőhöz képest. Emiatt még a vektor felépítése előtt szükségünk van a kvízkérdések szövegének kibővítésére, így a belőlük képezett jellemzővektorok „sűrűbbek” lesznek. Ezt az eljárást *query expansion*-nak hívjuk [7, 9.2.2 fejezet.]. A kibővítés során a kvízkérdés szövege végére illesztünk olyan szót, amiről azt feltételezzük, hogy az osztályozás során nagy relevanciaértékkel rendelkeznek. Azért szükséges még a vektorfelépítés előtt elvégezni ezt a lépést, mivel a bag-of-words reprezentáció miatt elveszítjük a szavak sorrendiségét, ami a megfelelő bővítményszavak meghatározása során fontos. Azonban pont emiatt a reprezentáció miatt tehetjük meg a szavak szöveghez való hozzáillesztését mindenféle nyelvtani szabályok betartása nélkül.

Az általunk használt query expansion alapja a WordNet tezaurusz [8]. Egy kibővítendő szövegben a főneveket, az igéket a mellékneveket és a határozószókat próbáluk meg kibővíteni ezeknek a rokon értelmű szavaival. A jellemzők kibővítése során el kell döntenünk, hogy a felsoroltak közül mely szófajokat bővítjük ki és hogy egy szó hány rokon értelmű szavát használjuk.

Mivel a tezaurusz egy szó összes lehetséges jelentését és a jelentésnek megfelelő szinonimákat tartalmazza, gyakran előfordul, hogy olyan szavakkal terjesztünk ki, melyeknek nincs köze a kibővítendő szöveghez. Ez ronthatja az osztályozás pontosságát. Emiatt szükséges meghatároznunk a kiterjesztendő szó szövegbéli jelentését. A megfelelő jelentés meghatározására Michael Lesk 1987-ben leírt módszerét használtuk [5].

1. Listing. Különböző kibővítések

```

1. Kibővített szöveg, csak főneveket felhasználva,
   szavanként az összes szinonimával és
   jelentésmeghatározás nélkül.

    What type of animal is a lamprey? character eccentric
       case animate_being beast brute creature fauna
       lamprey_eel lamper_eel

2. Kibővített szöveg, mind a négy szófajt felhasználva,
   szavanként az összes szinonimával és
   jelentésmeghatározással.

    What type of animal is a lamprey? animate_being beast
       brute creature fauna embody be personify lamprey_eel
       lamper_eel

3. Kibővített szöveg, mind a négy szófajt felhasználva,
   szavanként az egy szinonimával és
   jelentésmeghatározással.

    What type of animal is a lamprey? animate_being
       personify lamprey_eel
```

5. Az adathalmazok

A témakörmeghatározó és nehézségbecslő modellek betanításához és teszteléséhez négy kérdéscsomagot használtunk fel. A kérdéscsomagokban levő kvízkérdéseket témakörük vagy nehézségük szerint külön fájlokban tároltuk el előfeldolgozás után. Előfeldolgozás során egy kérdésnek csak a helyes válaszát tartottunk meg (amennyiben több válaszlehetőség is rendelkezésre állt). Emellett egyes kérdéscsomagokban előfordult, hogy egy kvízkérdés képre vagy esetleg hangra utalt. Ezeket a kvízkérdéseket kiszűrtük a végleges fájlokból. Az így létrejött szöveges állományokban egy kérdés formája a következő:

2. Listing. Egy adathalmazbeli kvízkérdés általános formája.

```
#Q Kérdés szövege
^ Helyes válasz
```

3. Listing. Példa egy helyes formára hozott kvízkérdésre.

```
#Q What is the capital and largest city of Hawaii, the 50th
   US state?
^ Honolulu

#Q Which continents are entirely in the Southern Hemisphere?
^ Australia and Antarctica
```

Az így feldolgozott kérdéscsomagokból három adathalmazt készítettünk az elvégzett kísérleteknek megfelelően.

5.1. Témakörökkel ellátott kvízkérdések adathalmaza

Az adathalmaz megépítése során két kérdéscsomagot használtunk fel. Az első, nagyobb kérdéscsomag az OpenTriviaQA [5], ami megközelítőleg 50000 egyedi kvízkérdést tartalmaz 22 témakörre osztva. Ezek közül a témakörök közül 10-et tudtunk felhasználni a kísérleteink során. A témakörök kiválasztásáról és a csökkentéséről a 6.1. alfejezetben lehet olvasni. A másik felhasznált kérdéscsomag az adathalmaz építése során a 2.1. alfejezetben már bemutatott Jeopardy! kvízjátékból származnak. A kvízjáték nem tiszta kategóriái miatt nagyjából 3500 használható kvízkérdést adott hozzá az adathalmazhoz. Összesen tehát a témakörökkel ellátott kvízkérdések adathalmaza 34 285 elemű.

[5] https://github.com/uberspot/OpenTriviaQA

5.2. Nehézséggel ellátott kvízkérdések adathalmaza

Ebbe az adathalmazba egy kérdéscsomag került. Ez a kérdéscsomag[6] Wikipedia szócikkeket használt fel kérdései alapjaként. Amellett, hogy minden kérdésről tudjuk, mely szócikkből származik, a legtöbb kérdést két nehézségi értékkel is ellátja. A két nehézségi érték két szemszögnek felel meg: a kérdező és a válaszadó szemszögének. Az előfeldolgozás során, ha egy kérdés két nehézségi értéke megegyezett, akkor besorolása egyértelmű volt. Amennyiben ez a két érték különbözött vagy valamelyik hiányzott, akkor a válaszadó szemszögéből ítélt nehézségi kategóriába soroltuk a kvízkérdést.

Az adathalmaz csupán 2000 kvízkérdést tartalmaz, mivel gyakran többször is előfordult egy kérdés szövege, csupán a helyes válasz hosszúsága és minősége változott. Ezekben az esetében a legrövidebb helyes válasszal rendelkező változatot használtuk a kérdések kvízjátékban való felhasználására gondolva.

5.3. Teszteléshez használt adathalmaz

A tanulási modellek végső teszteléséhez a 2.3. alfejezetben ismertetett Honfoglaló társasjáték magyar nyelven megjelent kérdéseit használtuk fel. Az adathalmaz 234 angol nyelvre fordított kvízkérdést tartalmaz, célja, hogy a kísérletek végén egy összehasonlíthassuk a betanított algoritmus pontosságát. Emellett az elkészült kvízjáték-alkalmazás első használt kérdései is ezek lesznek.

[6] https://www.kaggle.com/datasets/rtatman/questionanswer-dataset

6. Az elvégzett kísérletek

Egy kvízkérdés témakörének és nehézségének meghatározására a 3. fejezetben leírt mindhárom osztályozó algoritmussal, és az algoritmusok számára a bemenetet előállító természetesnyelv feldolgozási módszerekkel kísérleteztünk. Egy ilyen kételemű tanulási modell jóságát a 3.3. alfejezetben bemutatott F_1 metrikával határoztuk meg. Ezeknek a modelleknek a megvalósítására a scikit-learn [9] és NLTK [6] Python könyvtárakat használtuk.

6.1. Témakör szerinti osztályozás

Első lépésben meg kellett határoznunk, hogy milyen témakörökbe osztályozzunk a kvízkérdéseket. Ehhez az alapot a Honfoglaló kvízjáték adta, ott 10 témakörbe vannak sorolva a kvízkérdések. Az OpenTriviaQA 22 témakörét összevetve a Honfoglaló témaköreivel 7 egyezést találtunk (biológia, földrajz, irodalom, sport, szórakozás, történelem, tudományok), ezekből indultunk ki. A hét pontos egyezésen kívül, a 22 témakörben találtunk pár rokon témakört is, így létrehoztuk a hét végleges osztályt, a zárójelben az osztályt felépítő témakörök vannak megjölölve:

- szórakozás (szórakozás, filmek, televízió),
- művészetek (irodalom, zene),
- földrajz (földrajz, a világ),
- biológia,
- történelem,
- sport,
- tudományok.

A zene társkategóriát volt a legnehezebb besorolni. Először saját osztályban szerepelt egyedül. Ebben az esetben a szórakozás, művészet és zene osztályoknak mind alacsony F_1 értéket számoltunk. Nem volt ez meglepő, hiszen nagyon hasonlóan vannak a kvízkérdések megfogalmazva ezekben a témakörökben és tematikájuk is hasonló. Ezután a szórakozás osztályba soroltuk ám az így megmaradt két témakörosztály még mindig alacsony F_1 értékkel rendelkezett (60%). Végül a legjobb eredményt az irodalommal kombinálva alkotta (72%).

Egy másik kérdés, amire válaszolnunk kellett a nagyvolumenű kísérletek előtt, az, hogy amennyiben rendelkezésre áll több válaszlehetőség egy kvízkérdésre, használjuk-e fel őket, és ha igen, akkor mindet vagy csak a helyest? A kérdés megválaszolása során kizártuk azt a lehetőséget, hogy az összes válaszlehetőséggel együtt osztályozzunk. Azért, mivel a helytelen válaszlehetőségek nem kell feltétlenül a kérdéshez kapcsolódjanak, emiatt könnyen egy kvízkérdés rossz osztályozásához vezethetnek. Ezután egy gyors kísérlet végrehajtása után megállapítottuk, hogy amennyiben a helyes válasz is a kvízkérdés szövege után van illesztve, akkor a tanulási modellek F_1 értéke átlagosan 1%-kal növekszik. Ezentúl tehát minden tesztben szerepel a kvízkérdés helyes válasza is.

A következő lépésben kimerítő tesztelést végeztünk el. A legjobb előfeldolgozásból és osztályozó algoritmusból álló tanulási modellt szeretnénk megtalálni. Ennek érdekében az általunk előállítható összes modellre elvégeztünk 10 kísérletet, amelyek jóságából (F_1 értékeiből) várható értéket és szórást számoltunk. A 4.2. fejezetben felsorolt jellemző kiválasztási módszereken kívül a scikit-learn alapbeállításaként szolgáló F-value kiválasztással is kísérleteztünk. A kiválasztás módszerén kívül a kiválasztott jellemzők számát is vizsgáltuk, amit az összes jellemzők számának egy bizonyos százalékával határoztunk meg. Stopszólistából háromfélét használtunk: egy üreset, a scikit-learn beépítettjét és az Apache Lucene-ben[7] találhatót kibővítve a 4.2.3. fejezetben leírt 4-gyel.

Egy általunk előállítható modell részei és a részek lehetséges értékei, fajtái:

- jellemzővektor előállító módszer (vektorizáló): {TF, TF-IDF}
- stopszólista: {Nincs, Beépített, Lucene}
- használt n-gram-ok: {1-1, 1-2, 1-3, 2-2, 2-3, 3-3}
- jellemző kiválasztási módszer (kiválasztás): $\{\chi^2$, MI, F-value$\} \times \{10, 20, \ldots, 100\}$
- normailázló: {Nincs, L_2}
- osztályozó algoritmus (osztályozó. alg.): {Naiv Bayes, SVM}

Miután a felsorolt elemek összes lehetséges értékeivel előállítottunk egy tanulási modellt, majd leteszteltük a megfelelő adathalmazon, kiválasztottunk a legjobb 5 modellt, hogy jellemzővektor kibővítés után is végezzünk velük kísérleteket. Minden kísérlet eredményét egy dokumentumalapú adatbázisban tároltunk el az általa nyújtott rugalmasság miatt.

nr.	vektorizáló	stopszólista	n-gram	kiválasztás	normalizáló	osztályozó alg.
1.	TF-IDF	Nincs	1-2	χ^2, 40	Nincs	Naiv Bayes
2.	TF-IDF	Lucene	1-2	χ^2, 20	Nincs	Naiv Bayes
3.	TF	Lucene	1-2	χ^2, 20	L_2	SVM
4.	TF-IDF	Beépített	1-2	MI, 20	L_2	SVM
5.	TF	Beépített	1-2	F-value, 10	L_2	SVM

2. táblázat. Témakör szerinti osztályozás szempontjából az általunk kiválasztott 5 tanulási modell paraméterei.

modell	F_1 várható értéke	F_1 szórása
1.	0.88038	≈ 0
2.	0.86914	0.00001
3.	0.85473	0.00001
4.	0.85146	0.00001
5.	0.84793	0.00001

3. táblázat. A vizsgált modellek F_1 várható értékei és szórásai jellmzőkiterjesztés előtt.

A megfigyelve a 2-es és a 3-es táblázatokat és kapott eredmények alapján levonhatunk pár következtetést. Az első, hogy amennyiben jelen van egy jellemző kiválasztási módszer, akkor a stopszavak jelenléte vagy hiánya szinte semmiben sem befolyásolja a modell jóságát. A második észrevétel is a jellemző kiválasztáshoz kapcsolódik. Jellemzően a 10 és 40 százalék közötti kiválasztás a leghatásosabb. Az eredményekből azt a következtetést is levonhattuk, hogy mivel a TF-IDF jellemzővektor előállítás során bizonyos szintű normalizálás is végbemegy, emiatt nem szükség továbbiakat alkalmaznunk. N-gramok terén leolvashatjuk, hogy a szavak bigrammokkal való kombinációja a legjobb.

A jellemzővektorok kibővítése után a kiválasztott 5 tanulási modell jósága növekedett, bár nagyon keveset. A legjobb eredményt a 2-es táblázat 1-es számú modellével értük el, ha használtunk jelentésmeghatározást, csupán a főneveket bővítettük ki és csak egy, a leghasználtabb szinonimával. Az így kapott modell F_1 várható értéke: 0.88075, a szórás továbbra is tart a nullához.

6.2. Nehézség szerinti osztályozás

A nehézség szerinti osztályozás munkamenete teljes mértekben hasonlít az előző alfejezetben már leírtakra, a különbség csupán annyi, hogy nem 7, hanem csak 3 osztállyal dolgozunk:

[7] https://lucene.apache.org/

- könnyű (easy),
- közepes (medium),
- nehéz (hard).

nr.	vektorizáló	stopszólista	n-gram	kiválasztás	normalizáló	osztályozó alg.
1.	TF	Nincs	1-3	F-value, 20	L_2	SVM
2.	TF	Nincs	1-3	χ^2, 30	L_2	SVM
3.	TF	Nincs	1-3	MI, 30	L_2	SVM
4.	TF	Lucene	1-3	F-value, 30	L_2	SVM
5.	TF	Lucene	1-2	χ^2, 10	Nincs	Naiv Bayes
6.	TF-IDF	Nincs	1-1	χ^2, 10	L_2	SVM

4. táblázat. Nehézség szerinti osztályozás szempontjából az általunk kiválasztott 6 tanulási modell paraméterei.

modell	F_1 várható értéke	F_1 szórása
1.	0.87207	0.00033
2.	0.85526	0.00022
3.	0.84599	0.00024
4.	0.82158	0.00041
5.	0.78142	0.00021
6.	0.77580	0.00069

5. táblázat. A vizsgált modellek F_1 várható értékei és szórásai jellmzőkiterjesztés előtt.

Megfigyelhetjük, hogy a nehézség szerinti osztályozásban, ezen adathalmazon a TF vektorizációval, L_2 normalizálással és SVM osztályozó algoritmussal rendelkező modellek teljesítettek a legjobban. Az 5. kiválasztott tanulási modell a legjobb, amit Naiv Bayes algoritmussal értünk el, itt viszont az L_2 normalizáló sokat rontana az F_1 értéken, nagyjából 10%-t. A Ebben az esetben is igaz a témakör szerinti osztályozásnál megfigyelt kapcsolat a jellemzőkiválasztás és a stopszavak között. Az adathalmaz kis mérete miatt számítottunk a nagyobb szórásra.

A jellemzők kibővítése után a legjobb tanulási modell a 4. táblázatban 1-es számú lett. Ebben az esetben is, ahogyan a témakör szerinti osztályozás esetében, a legnagyobb F_1 várható érték jelentésmeghatározással, egy szinonimával és csak főnevek kibővítésével elérhető. Ám itt magénak a jelentésmeghatározásnak nem tűnt nagy szerepe, de ennek biztossága még további kísérleteket igényel. Ennek a modellnek a F_1 várható értéke: 0.88493 és szórása: 0.00025.

7. Következtetések és továbbfejlesztési lehetőségek

A kísérletekkel foglalkozó fejezetben már levontunk pár következtetést, ezek közül pár itt is meg fog jelenni az összegzés kedvéért. A legelső következtetés, amit levonhatunk az, hogy egy kvízkérdés osztályozásánál segít, ha a kérdés helyes válasza is szerepel a feldolgozandó szövegben. A jellemzők kiválasztásánál a χ^2 kiválasztás stabilan jól teljesített minden tesztesetben, a megmaradó jellemzők száma az eredeti 10–30% között lesz ideális. A stopszavak hatása elhanyagolható amennyiben jelen van egy ilyen jellemzőkiválasztási módszer és a jellemzővektor felépítése során a bigramok, de esetenként a trigramok jelenléte elősegíti a helyes osztályozást.

A jellemzők kibővítését illetően levonhatjuk azt a következtetést, hogy a jelentésmeghatározás mindig javítja a kibővítés hatékonyságát. Emellett igaz az a mondás miszerint a kevesebb néha több. A szó összes szinonimáját, vagy csak egy jelentéséhez kapcsolódó összes formáját nem éri meg felhasználni, mert az az osztályozás jóságának kárára mehet. Elegendő a rokon értelmű szavak közül csak egyet megtartani, mégpedig a leggyakrabban használtat. Szófajok tekintetében a főnevek kibővítése tűnt a leghatékonyabbnak, de a többi szófaj kibővítésével is eredményesek lehetünk.

Egy kvízkérdés témakör szerinti osztályozásánál sokkal nagyobb jelentősége van a speciális, azaz a zsargon szavaknak. Ezek által a való életben is könnyen behatárolhatjuk egy szöveg vagy beszélgetés témáját. Ezzel ellentétben, a nehézségbecslésnél a kérdőszó és a válasz szerkezete van nagy hatással az osztályozásra. El tudjuk képzelni, hogy egy dologra többféleképpen is rákérdezhetünk: például a következő két kérdés két különböző nehézséggel bír: Igaz-e, hogy 1456-ban volt a nándorfehérvári csata, amely után Hunyadi János életét vesztette? / Mikor volt a nándorfehérvári csata és melyik híres személyiség hunyt el következében?

Jelenleg egy harmadik osztályozó algoritmussal, a logisztikus regresszióval végzünk kísérleteket. A 6. fejezetben vázolt szövegfeldolgozási módszerekkel együtt teszteljük az új osztályzó algoritmus jóságát. A logisztikus regresszión kívül lehetséges, hogy egy negyedik módszer, a döntési fával (decision tree) való osztályozás tesztelése is sorra fog kerülni.

Hamarosan a 6. fejezetben bemutatott kísérleteken kívül, azt a feltételezésünket is le fogjuk tesztelni, miszerint egy kvízkérdés nehézsége összefüggésben áll annak olvashatóságával. Hiszen, ha egy kérdés nincs egyértelműen és közérthetően megfogalmazva, nehezebbnek érezhetjük a kvízkérdést. Emiatt egy kvízkérdés nehézségét meg fogjuk vizsgálni annak Flesch–Kincaid olvashatósági értékével [4]. Ez a metrika egy szöveg mondatai, szavai és szótagjai számát felhasználva határozza meg annak olvashatósági értékét. A módszer 1975-ben lett

kidolgozva a katonaság számára, főleg újságcikkek és jelentések olvashatóságára lett hangolva. Ezek a szövegek jelentősen hosszabbak, mint egy kvízkérdés, így lehetséges, hogy nem a legmegfelelőbb metrika e célra. Emiatt előfordulhat, hogy egy kvízkérdés más olvashatósági értékeivel is próbálkozni fogunk.

Nem tudjuk kizárni azt a lehetőséget sem, hogy egy kvízkérdés olvashatósága mellett a témaköre is befolyással van a kérdés nehézségére. Amennyiben a témakör szerinti osztályozó algoritmus pontossága megfelelő lesz, a nehézségbecslés során egy kvízkérdés általunk meghatározott témakörével is végezhetünk kísérletek.

A jövőben el fog készülni egy a 2.3. alfejezetben bemutatott Honfoglaló kvízjátékra hasonlító webes alkalmazás, amely segítségével még több és relevánsabb információt tudunk gyűjteni a kvízkérdésekről. A tervek szerint a felhasználók feltölthetik majd saját kvízkérdéseket tartalmazó csomagjaikat, amiket a már bemutatott és finomított osztályozó algoritmusok feldolgoznak és javaslatokat adnak egy kvízkérdés témakörét és nehézségét illetően. A feltöltött kvízkérdésekkel majd lehetséges lesz játszani a kvízjátékkal. Az osztályozás során pontosan be lehet majd állítani, hogy milyen stílusú játékot szeretnénk játszani, így személyre szabható lesz minden játék és az élmény is, amit nyújt.

A kvízjáték során gyűjtött információkkal tudunk majd javítani az osztályozó algoritmusok jóságán. Emellett új módszerek használatát teszi lehetővé, a nehézségbecslés terén. A játékosok válaszai alapján is következtetni próbálhatunk a kérdés nehézségére. A becslésben fel tudnánk használni a helyes és helytelen válaszok számát, a válaszoló játékos értékelését, esetleg témakör szerinti profizmusát.

Hivatkozások

[1] Tikk Domonkos. *Szövegbányászat*. Typotex Kft, 2007.

[2] Juhász József és mások. *Magyar értelmező kéziszótár*. 8. kiadás. Akadémiai Kiadó, 1989.

[3] Dan Jurafsky és James H. Martin. *Speech and Language Processing. 3rd*. 2023. URL: https://web.stanford.edu/~jurafsky/slp3.

[4] J Peter Kincaid és mások. *Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel*. Technikai jelentés. Naval Technical Training Command Millington TN Research Branch, 1975.

[5] Michael Lesk. „Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone". *Proceedings of the 5th Annual International Conference on Systems Documentation*. SIGDOC '86. Toronto, Ontario, Canada: Association for Computing Machinery, 1986, 24–26. oldal. ISBN: 0897912241. DOI: 10.1145/318723.318728. URL: https://doi.org/10.1145/318723.318728.

[6] Edward Loper és Steven Bird. „Nltk: The natural language toolkit". *arXiv preprint cs/0205028* (2002).

[7] Christopher D. Manning, Prabhakar Raghavan és Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. URL: https://nlp.stanford.edu/IR-book/information-retrieval-book.html.

[8] George A. Miller. „WordNet: A Lexical Database for English". *Commun. ACM* 38.11 (1995. nov.), 39–41. oldal. ISSN: 0001-0782. DOI: 10.1145/219717.219748. URL: https://doi.org/10.1145/219717.219748.

[9] F. Pedregosa és mások. „Scikit-learn: Machine Learning in Python". *Journal of Machine Learning Research* 12 (2011), 2825–2830. oldal.

[10] Pseudo-Apollodorus. *Apollodorus, The Library*. Online. Utolsó hozzáférés dátuma: 2023. 04. 12. 16:19. URL: http://data.perseus.org/citations/urn:cts:greekLit:tlg0548.tlg001.perseus-eng1:3.5.8.

[11] Albert P. Southwick és A. M. *QUIZZISM; AND ITS KEY. Quirks and Quibbles From Queer Quarters.* New England Publishing Company, 1896. URL: https://babel.hathitrust.org/cgi/pt?id=uva.x001039632&view=1up&seq=7.

[12] Yiming Yang és Jan O Pedersen. „A comparative study on feature selection in text categorization". *Icml*. 97. kötet. Citeseer. 1997, 412–420. oldal.