KORSZERŰ ADATELEMZŐ ALGORITMUSOK ALKALMAZÁSA A KÖZIGAZGATÁSBAN

Racskó Péter

Nemzeti Közszolgálati Egyetem,
Budapest

SZÉCHENYI 2020

Magyarország Kormánya

Európai Unió
Európai Szociális Alap

BEFEKTETÉS A JÖVŐBE

KORSZERŰ ADATELEMZŐ ALGORITMUSOK ALKALMAZÁSA A KÖZIGAZGATÁSBAN

Projekt szakmai vezető
Kő Andrea PhD

Szerző:
Racskó Péter PhD

A kézirat lezárásának dátuma:
2018. 08. 15.

Kiadás éve:
xx

Kiadó:

Nemzeti Közszolgálati Egyetem
Közigazgatási Továbbképzési Intézet
www.uni-nke.hu

Felelős kiadó:

Prof. Dr. Kis Norbert rektorhelyettes
Címe: 1083 Budapest, Üllői út 82.

A kiadvány a
KÖFOP-2.1.2-VEKOP-15.
„A jó kormányzást megalapozó közszolgálat-fejlesztés” című projekt
keretében készült el és jelent meg.

© Racskó Péter PhD, 2020

© Nemzeti Közszolgálati Egyetem
Közigazgatási Továbbképzési Intézet, 2020

A mű szerzői jogilag védett. Minden jog, így különösen a sokszorosítás, terjesztés és fordítás joga fenntartva. A mű a kiadó írásbeli hozzájárulása nélkül részeiben sem reprodukálható, elektronikus rendszerek felhasználásával nem dolgozható fel, azokban nem tárolható, azokkal nem sokszorosítható és nem terjeszthető.

TARTALOMJEGYZÉK

1. BEVEZETÉS

Az adatok korát éljük. A társadalom minden szereplője, az emberek, a vállalkozások, a közszféra hatalmas mennyiségű adatot állít elő minden másodpercben. Megkérdezhetjük, hogy vajon hol voltak ezek az adatok korábban.

Nagyobb részben már léteztek, csak nem mértük és tároltuk azokat, mert vagy nem volt rájuk szükség, vagy nem volt erre lehetőség, kisebb részben pedig nem is léteztek. A nem létező adatokra példa a személyazonosító jel, amelyet egy 1992-es törvény határozott meg, ez előtt nem is létezett, vagy egy tweet, amely a Twitter létrejötte előtt nem jöhetett létre. Az informatikai eszközök, a kommunikációs hálózatok gyors fejlődésével és a rendelkezésre álló kapacitások exponenciális növekedésével ma már lehetővé vált korábban elképzelhetetlen mennyiségű adat megfigyelése, mérése, tárolása és továbbítása. A statisztikák szerint jelenleg 2,7 zettabájtnyi adat létezik[1], és 1,2 évente ez megduplázódik. A zettabájt az a szám, amely az egyes után 18 nullát tartalmaz. Az interneten számos leírást találunk arra vonatkozóan, hogy mindez hány mozifilmnek, tweetnek stb. felel meg. Soknak.[2]

Egyáltalán, mit jelent az, hogy egy adat létezik? Mi ezt a kérdést csak gyakorlati szempontból vizsgáljuk, egy adat akkor létezik gyakorlati szempontból, ha az interneten vagy lokálisan elérhető tároló eszközön tárolják.

Kérdés, hogy a hatalmas tárolt adatmennyiségből hogyan lesz használható adat, információ. A könyvben arra törekszünk, hogy ismertessük azokat a korszerű adatkezelési és adatelemzési módszereket, amelyek hasznos, a gyakorlatban használható információt állítanak elő a rendelkezésre álló adattömegből. Jelenleg kicsit hasonló a helyzet az 1990-es évek elejéhez, amikor nyilvánvalóvá vált, hogy az akkori hagyományos, döntően statisztikai módszerek már nem voltak alkalmasak a nagy és folyamatosan növekvő adatállományok értelmezésére és elemzésére. „Megfulladunk az információban, de tudásra éhezünk” – hangzik az 1990-es évek elejéről származó, E-O. Wilsonnak tulajdonított mondás (Wilson, 2018). A tudás iránti éhség motiválta az adattárházak és az adatbányászati eszközök kifejlesztését, lehetővé téve az akkori adatmennyiségek kezelését és elemzését.

A jelenlegi adatok mennyisége nagyságrendekkel[3] haladja meg a 25 éve előtti szintet, szerkezetükben is változtak, szemben a korábbi, jól strukturálható, többnyire adatbázisokban tárolt adatokkal a jelenlegi adatok nagy része kép, hang, videó, szöveges dokumentum formában létezik, amelyek feldolgozására a korábbi módszerek nem alkalmasak. Nem elhanyagolható az sem, hogy a feldolgozás sebessége iránti igények is változtak, sokszor egy-egy döntéshez nem elég a közelmúlt adatainak elemzése, hanem az elemzés eredményeire azonnal, az adatok beérkezésével közel egyidejűleg, azaz valós időben van szükség.

1 https://martech.zone/ibm-big-data-marketing/
2 https://mashable.com/2011/06/28/data-infographic/?europe=true#V6Jzs95HaOqy
3 Egy nagyságrend tízszeres mennyiséget jelent.

Ezt az új helyzetet a *big data* kifejezéssel szokás jelölni.

Könyvünk első fejezetében a big data adatok jellegét és közigazgatási szerepét tárgyaljuk, a második fejezetben a big data feldolgozásra létrehozott szoftverkörnyezetet mutatjuk be. A szoftverek fejlesztésében igen nagy szerepet játszottak a technikai óriásvállalatok, elsősorban a Google. A későbbiekben azután a kifejlesztett szoftvereket átadták nyílt forráskódú rendszereket fejlesztő és karbantartó projekteknek, így gyakorlatilag ma már minden big data szoftvereszköz ingyenesen elérhető és akár üzleti, akár közigazgatási, akár tudományos célra szabadon használható, természetesen a megfelelő szaktudás birtokában. A legtöbb big data szoftvernek létezik fizetős disztribúciója is, ezek mögött természetesen gyártói támogatás áll.

A könyv harmadik fejezetében a gazdasági életben széles körben használt ajánlórendszereket ismertetjük, kitérve az alkalmazási lehetőségekre a közszférában.

A korszerű adatelemzés fő irányzata a mesterséges intelligencia legjelentősebb részterülete, a gépi tanulás. A következő, negyedik fejezetben a felügyelt, míg az ötödikben a felügyelet nélküli gépi tanulás alapfogalmait és leggyakrabban használt eljárásait ismertetjük.

A mesterséges intelligencia fontos építőkövei a neurális hálózatok, amelyek megnevezésük szerint az emberi agyhoz hasonló struktúrával imitálnak és az agy elképzelt működését utánozva oldják meg a feladatokat. A neurális hálókról szól a hatodik fejezet.

A hetedik, a nyolcadik és a kilencedik fejezet kiemelten tárgyalja a közszférában a jövőben igen nagy potenciállal rendelkező területeket, a képfeldolgozást, a természetes nyelvi elemzést és a hálózatelemzést.

A tizedik fejezet nem újabb módszereket, hanem a mesterséges intelligencia, gépi tanulás és elemzés hatékonyságának és megbízhatóságának mérésére szolgáló eszközöket mutatja be.

A tizenegyedik fejezet a rendszerek hatékonyságának, pontosságának mérésére használt eszközöket ismerteti. Minthogy az alkalmazott módszerek sokszor igen komplexek, és az elemzők számára esetenként fekete dobozként működnek, így szükséges, hogy egységes, mindenki által elfogadott kritériumokat használjunk a hatékonyság mérésére. A mérőeszközök gyakran különböznek az általános alkalmazott statisztikai tesztektől, és sokszor nem is általánosak, hanem az alkalmazási területtől is függenek.

Reméljük, hogy a könyv elolvasásával az olvasó megfelelő áttekintéssel rendelkezik majd a korszerű elemző módszerek főbb tulajdonságairól és alkalmazási lehetőségeiről, elsősorban a közigazgatás területén.

2. A BIG DATA A KÖZIGAZGATÁSBAN

A big data fogalma

A big data fogalma nehezen meghatározható. A kifejezés nem csak a nagy adatmennyiségeket, hanem az adatok szerkezetének sajátosságait, az adatelemzéshez használt módszereket és eljárásokat, az elemzések eredményeinek alkalmazását is jelenti. A pontos meghatározást a big data tulajdonságainak leírásával helyettesítjük. A tulajdonságok jelölésére a szakirodalomban a „3V" jelzést használják, ami a big data alábbi tulajdonságait írja le:

Volume, azaz mennyiség

Az elektronikus tárolás költségének gyors csökkenésével lehetővé vált a korábbiakban elképzelhetetlen mennyiségű adat tárolása. Az interneten több milliárd gigabájt adat keletkezik naponta[4], ráadásul az adatok 90%-a az utóbbi két évben jött létre. 2017-ben percenként mintegy félmillió tweetet küldtek, a Google keresőjének sok millió kérdést tettek fel, a Facebookon félmillió komment született, százezernyi fényképet osztottak meg és a Youtube-ra 400 órányi videót töltöttek fel. Ezek az adatok korábban fizikailag nem léteztek. A továbbiakban az adatok információtartalmával és elemzésével is foglalkozunk.

Variety, azaz változatosság

Korábban az adatok elsődleges forrásai jól strukturált adatbázisok és fájlszerkezetek voltak. Az üzleti életben és a közszférában elsődlegesen a relációs adatbázisok szolgáltak adatforrásként. A relációs adatbázisokban az adatokat relációs táblákba rendezve tárolják, ahol a táblázat egy sora egy rekord, amit egy azonosító alapján lehet megtalálni. Egy rekordban egyféle adattípus szerepel, például egy ügyfél adatai. A táblázat oszlopai az ügyfél tulajdonságait reprezentálják, például egy oszlopban találjuk az ügyfél nevét, egy másikban a lakcímét stb. A relációs adatbázisokat a szervezet vagy a szerződött informatikai szolgáltatók számítógépein találjuk.

Jelenleg az adatok túlnyomó többségét nem relációs adatbázisokban, hanem a legkülönbözőbb formában, a legkülönbözőbb helyeken tárolják. Az üzleti szféra szervezeteinek jelentős része marketing vagy PR-célra nagymértékben hasznosítja a közösségi médiában elérhető adatokat, elemzi a rá vonatkozó tweeteket, arcfelismerő programokat futtat, de ez

4 https://public.dhe.ibm.com/common/ssi/ecm/wr/en/wrl12345usen/watson-customer-engagement-watson-marketing-wr-other-papers-and-reports-wrl12345usen-20170719.pdf

jellemző például a bűnmegelőzési tevékenységet folytató intézményekre is. Ezek az adatok – szöveges dokumentumok, e-mailek, videók, fényképek – nem jól strukturált relációs adatbázisokban vannak, hanem a legkülönbözőbb formában léteznek. Ma már egyetlen üzleti vállalkozás sem tekinthet el a nem, vagy csak részben strukturált adatok használatától, mert ezzel versenyhátrányba kerül, így egy ideje komoly fejlesztések folynak a nem strukturált adatok elemzési módszereinek kutatása és alkalmazása terén. Könyvünk legnagyobb része is ezzel a területtel foglalkozik.

Velocity, azaz sebesség

Korábban a strukturált adatok összegyűjtéséhez és feldolgozásához, előre megadott struktúrákba rendezéséhez idő kellett, az új adatokat csak késéssel lehetett elemezni és felhasználni a döntéshozatali vagy más folyamatokban. Jelenleg azonban az azonnali feldolgozás és felhasználás egyre erősödő követelményként jelentkezik az üzleti életben. Amikor valaki bejelentkezik egy elektronikus kereskedelemmel foglalkozó webes oldalon, azonnal testreszabott vásárlási ajánlatokat kap. A tőzsdei kereskedelemben a másodperc tört része alatt kell dönteni egy-egy tranzakcióról. A politikai kampányok során a politikusok folyamatosan, késleltetés nélkül szeretnének értesülni a közhangulatról. A gyorsaság tehát nem csak arra utal, hogy az adatok nagy mennyiségben és gyorsan keletkeznek, hanem arra is, hogy azokat gyorsan, a lehető legkisebb késleltetéssel kell feldolgozni.

A 3V-t a szakirodalomban újabb, természetesen V-vel kezdődő jellemzőkkel egészítették ki:

Veracity, azaz valódiság

Az interneten keletkezett adatok valóságtartalma sokszor kétséges. A feltölthető dokumentumok, vélemények, általában a tartalom minőségét és valóságtartalmát az interneten senki és semmi nem ellenőrzi, leszámítva a kifejezetten illegális tartalmakat, mint például a drogokkal való kereskedést, vagy gyermek pornográfiát. Így semmi nem áll útjában a hamis, megtévesztő, pontatlan információ terjesztésének és terjedésének. A lapos Föld elméletét éppen úgy elviseli az internet, mint egy valóságos asztrofizikai kutatás leírását. Így egy adat valóságtartalmáról a felhasználónak kell meggyőződnie, a hibás vagy hamis adatokat neki kell kiszűrnie. Ez sokszor nem egyszerű feladat, főleg, ha nem nyilvánvaló tények meghamisításáról vagy szándékos megtévesztésről van szó. Így egy adatelemzésnél az elemző felelőssége, hogy valósághű adatokat használjon.

Variability, azaz változatosság

Ezzel a tulajdonsággal azt fejezzük ki, hogy az adatok dinamikusan változnak. A gyors változás megértéséhez, az elemzésekhez, a folyamatosan érkező adatfolyamok értékeléséhez a hagyományos elemző módszerek általában nem alkalmasak, új módszerekre van szükség. Ilyen módszereket természetesen folyamatosan fejlesztenek, illetve a hagyományos, statikus adatokkal dolgozó módszereket teszik alkalmassá dinamikus adatok kezelésére.

Value, azaz érték

A sok adat legnagyobb része egy adott felhasználó számára értéktelen. Napi sok milliárd spam, a közösségi oldalakra feltöltött egyéni tartalom, sok blogbejegyzés az elemző számára legtöbbször közömbös. Az elemzőnek ki kell tudnia válogatni az elemzés céljából hasznos, értékes tartalmakat. Egyes szerzők megállapítják, hogy nem túl bonyolult algoritmusok nagy adatmennyiségeken igen jó eredményeket adhatnak. Ezt a jelenséget az „adatok ésszerűtlen hatékonyságának" is nevezik (Gillespie, 2014). Sokszor nehéz eldönteni, hogy mely adatok az értékesek.

Referenciakeret

A jellemzők leírása mellett hasznos egy referenciakeretet felállítani a big data közigazgatási célú használatához, amely keretbe foglalja az elvégzendő tevékenységeket:

- adatgyűjtés és -tárolás,
- adatelemzési technikák alkalmazása,
- az eredmények hasznosítása.

Az adatok gyűjtése, tárolása:

- nagy adatmennyiségekkel dolgozunk,
- az adatok részben strukturáltak, részben nem strukturáltak,
- az adatforrások és az adatformátumok sokfélék (szöveges dokumentumok, hang- és videófájlok, webes tartalmak stb.).

Az adatelemzési technikák alkalmazása:

- Az elemzés adatvezérelt, nincsenek előzetes feltételezéseink az adatok belső összefüggéseiről.
- Az elemzés a múltbéli adatok alapján elsősorban a jelenlegi (valós idejű elemzés, nowcasting) és a jövőbeni eseményekre, adatokra fókuszál (előrejelzés, predictive analysis).

Az eredmények hasznosítása:

- Az elemzés során a gyakorlati döntéshozatalban hasznosítható, érdekes eredmények elérésére, a cselekvéshez szükséges tudás megszerzésére törekszünk.

Biztonság és adatvédelem a big data korában

A big data talán legelterjedtebb közigazgatási alkalmazási területe az egyén és a társadalom biztonságával kapcsolatos elemzések, előrejelzések készítése. A biztonság mellett szükséges az egyén személyes adatainak védelme, mert a biztonság és az adatvédelem egymástól nem elválasztható fogalmak. Az adatvédelem egyformán vonatkozik az üzleti szféra és a közigazgatás adatkezelő és adatfeldolgozó tevékenységére.[5] A személyes adatok védelmének fontos elve, hogy egy személy megítélése nem történhet annak alap-

5 https://www.naih.hu/jogszabalyok.html

ján, hogy melyek a szándékai, mit szeret és mit nem, ugyanakkor a társadalmi és egyéni biztonság megköveteli, hogy az állam megvédje polgárait. Ehhez természetesen az állami szervek adatokat gyűjtenek és elemeznek, igyekeznek felkészülni a kockázatokra. Az adatok gyűjtésének, elemzésének és a felhasználás módjának meg kell felelni az adott országban érvényes adatvédelmi előírásoknak, és egyensúlyba kell hozni a biztonság és az adatvédelem szempontjait. Megjegyezzük, hogy a személyes adatok védelmére vonatkozó társadalmi elvárások és az ezekhez kapcsolódó szabályozások meglehetősen heterogének még az Egyesült Államok és az Európai Unió közötti különbségek is igen jelentősek.

A biztonságért felelős szervezetek számára elérhető személyes adatok mennyisége drasztikusan megnőtt. Kamerák, közösségi hálók és számos más adatforrás szolgál hasznosítható információval. A feldolgozásra használható módszerek folyamatosan fejlődnek. Az elemzők egyre több személyes adatot képesek felhalmozni az állampolgárokról és egyre jobb személyes profilokat tudnak előállítani. Kizárólag a megfelelően részletes, betartható és ellenőrzött adatvédelmi szabályozás biztosíthatja a személyes adatok megfelelő védelmét. Arra is fel kell készülni, hogy a technológia fejlődése olyan gyors, hogy a szabályozás nem tudja kellő rugalmassággal követni. Ilyen esetekben az adatkezelők és adatfeldolgozók önkéntes etikai normái segíthetnek. A témáról részletesebben is írunk a 12. fejezetben.

Adatelemzési technikák

A big data adatelemzés – mint említettük – általában nem előre megadott hipotéziseket vizsgál, mint a hagyományos statisztika, hanem előre nem látott összefüggéseket, mintákat keres, előrejelzéseket készít. Az alkalmazott módszereket gépi tanulásnak vagy mesterséges intelligencia eljárásoknak is nevezik, ami egyrészt azt fejezi ki, hogy a modelleket adatokon tanítják be a helyes következtetések levonására, másrészt a tanulási mód sokszor emlékeztet arra, ahogyan az emberi intelligencia működik, bár azt túlzás lenne állítani, hogy a kettő egyforma.

A gépi tanulásnak a felügyelt és a felügyelet nélküli módszereit különböztetjük meg.

A felügyelt gépi tanulás

A felügyelt tanulás esetén egy mintaadathalmazt használunk az algoritmus betanítására. A minta elemeit változókkal jellemezzük. Például, ha a minta személyekből áll, a változók a személyek adatai, életkora, neme, foglalkozása. A leggyakoribb feladat az, hogy olyan algoritmust keressünk, amely egyes változók értékeiből meg tudja becsülni más változók értékeit. Például egy banki hitelbírálati eljárásban a kérelmező demográfiai és más adataiból előre tudja jelezni, hogy az illető vissza fogja-e fizetni a hitelt. Ez tipikusan osztályozó algoritmus, ahol az egyik osztályba a hitelt visszafizetők, a másikba a nem fizetők tartoznak. Ha az algoritmus tévedési aránya elfogadható, akkor élesben is lehet alkalmazni. A betanítás múltbeli tényadatokon történik, tehát pontosan tudjuk, ki az, aki valóban visszafizette a hitelt és ki az, aki nem, vagyis a minta elemeit fel tudjuk címkézni a „visszafizette” és a „nem fizette vissza” címkékkel. Ezt a tanulási módot azért hívják felügyelt gépi tanulásnak, mert felügyelni tudjuk, hogy egy mintaelem melyik osztályba tartozzon.

A felügyelet nélküli gépi tanulás

A felügyelet nélküli tanulásnál nem tudjuk előre kijelölni, hogy a minta egy eleme mely kritériumoknak felel meg, sőt, kritériumokat sem határozunk meg előre. Azt várjuk, hogy a felügyelet nélküli algoritmus automatikusan alakítson ki csoportokat, találjon előre nem ismert összefüggéseket, asszociációkat, döntési stratégiákat, oldjon meg feladatokat. Egy vállalkozás ügyfeleinek szegmentálása tipikusan ilyen feladat.

A félig felügyelt gépi tanulás

A félig felügyelt tanulás a felügyelt és a felügyelet nélküli tanulás hibridje. A minta egy része címkézett, míg más része nem. Tipikus példa a természetes nyelvi elemzés, melynek során például webes tartalmakat osztályozunk aszerint, hogy relevánsak-e számunkra, vagy nem. A tanulás során mind címkézetlen, mind címkézett adatokat használunk.

A megerősítéses gépi tanulás

A megerősített tanulás – a legfiatalabb a gépi tanulási algoritmusok közül – nem tartozik egyik fenti kategóriába sem. Itt a tanulás folyamatában kap „jutalmat” vagy „büntetést” az algoritmus és a tanulás folyamán a legnagyobb jutalom elérése a cél. A megerősített tanuló algoritmusok meglepően jó teljesítményre képesek stratégiai játékok megtanulásánál anélkül, hogy a szabályrendszeren kívül bármit tudnának a tanulás kezdetén.

A mélytanulás

A mélytanuló algoritmusok szintén a gépi tanuló algoritmusok egy osztályát alkotják (Deng et al., 2014).

Mélytanuló algoritmusoknak a többrétegű, nem lineáris neurális hálókat szokás nevezni (ld. 6. fejezet). Minden réteg az előző réteg outputját használja inputként, amely ún. látens változókat is használhat. Jelenleg nyelvi modellezésre, képfelismerésre, kézírások felismerésére használják. Igen jó eredményeket képes produkálni, de a betanításhoz általában nagyon nagy mennyiségű adatra van szükség. További hátrányuk, hogy a modell sokszor olyan bonyolult szerkezeteket alakít ki, hogy az eredményeket nem lehet közérthető formában értelmezni.

Big data elemzések alkalmazása a közigazgatásban

A közszolgáltatásokban nagyon sok adat keletkezik. Az adatok elemzése lehetőséget ad arra, hogy a kormányzatok csökkentsék költségeiket. Az Egyesült Királyságban működő Policy Exchange Institute szerint a big data alkalmazásokkal az Egyesült Királyság Kormánya évi 33 Mrd GBP-t takaríthat meg. Ez a megtakarítási potenciál európai szinten 250 Mrd EUR.[6]

Miből származhatnak ezek a megtakarítások?

6 https://www.computerworlduk.com/it-vendors/government-could-save-33bn-year-using-big-data-says-think-tank-3367938/

Az átláthatóság és a döntéshozatal erősítése és a költségek csökkentése

A közigazgatásban gyakran előfordul, hogy olyan adatokat kérnek az állampolgároktól, amelyek a kormányhivatalok valamelyikében már léteznek. Erre jó példa az adóbevallás, amikor ugyanazokat a személyes vagy céges adatokat kell megadni minden évben, és olyan bevételi/kiadási adatokat kell megadni, amelyről a hatóságoknak legtöbbször amúgy is van információjuk. A hazai személyi jövedelemadó-bevallási rendszer jelenleg jó példát mutat arra, hogy a személyek bevallásának legnagyobb részét az adóhivatal úgy elő tudja készíteni, hogy a személynek ehhez legtöbbször nem kell új adatokkal szolgálnia, csak jóvá kell hagynia. Hasonló eljárást alkalmaz a svéd és a dán adóhatóság is.

A költségcsökkentés és hatékonyságnövelés könnyen megvalósítható, amennyiben az adatokhoz való hozzáférés centralizált abban az értelemben, hogy az igazgatási szervek egységes módon férnek hozzá a centralizáltan, vagy decentralizáltan, tárolt adatokhoz, természetesen a jogosultságok megfelelő szabályozása mellett. Az egységes adattárolás és hozzáférés jelentősen csökkenti a hibák és az inkonzisztencia arányát. Ehhez olyan szoftverek is használhatók, mint például az eredetileg NSA által kifejlesztett, jelenleg már nyílt forráskódú Apache Accumulo, amely a Hadoop HDFS architektúrára épül és biztosítja az államigazgatási szintű biztonságos hozzáférés menedzsmentet.

A közigazgatási adatok elérhetősége, az információ szabad áramlása hozzájárul az átláthatóság és az állampolgári bizalom erősítéséhez. Az állampolgárok megvizsgálhatják és megérthetik az államigazgatás által generált adatokat, és azt, hogy azokat mire használják. Az adatok ilyen jellegű megosztása innovatív új szolgáltatások bevezetésére ad lehetőséget az állampolgárokkal való közös fejlesztésekkel. Az átláthatóság biztosítja, hogy az állampolgárok figyelemmel kísérhetik, hogy a közigazgatásban hogyan költik a közpénzeket, és ez motiválja a döntéshozókat az ésszerű költésre.

A big data elemzésekkel azt is könnyen ki lehet deríteni, hogy például melyik tisztségviselő mennyit és hova utazott, milyen célból, kivel és milyen eredménnyel. Egy big data technológián alapuló intelligens utazás- és költségmenedzsment rendszer segít elkerülni a felesleges utazásokat és segít megérteni, hogy mely utazások fontosak és melyek nem.

Az állampolgári élmény

Az üzleti szférában ma már megszokott felhasználói élmény fogalmához (user experience) hasonlóan a közigazgatásban az állampolgári élmény (citizen experience) fogalma kezd elterjedni. A strukturálatlan és strukturált közösségi/nyilvános adatok elemzése segíti a közigazgatást abban, hogy gyorsan reagáljon a változó környezetre, amikor az állampolgárok valamilyen beavatkozást, segítséget várnak. Az állampolgárok szegmentálása és azonosítása segít megtalálni azokat, akik igényelnek valamilyen támogatást, mert például elvesztették a munkájukat vagy más okból kerültek nehéz helyzetbe. Az algoritmusok segítenek meghatározni, hogy milyen segítségre van szükségük. Az online adatok, vagy éppen az állampolgárok telefonhívásaiból készített hangelemzés, segítenek megérteni a lakosság véleményét, érzéseit, azt, hogy mit szeretnének, kezdve a közvetlen környezettől a városi és az állami szintig. A véleményelemzés segíthet megelőzni a tömeges elégedetlenség kialakulását is.

A perszonalizált megközelítés alkalmazható a választásoknál (ahogyan alkalmazzák is) annak megértésére, hogy a választók mit szeretnének és hogyan lehet eljuttatni hozzájuk a megfelelő üzeneteket. A big data alkalmazás és a perszonalizált megközelítés kiváló példája a 2012-es Obama-kampány az Egyesült Államokban[7].

Az adócsalások és a szociális támogatásokkal való visszaélések csökkentése

Az adózási rendszer minden országban nagy adatmennyiséget generál. Az adatok részletes elemzésével csökkenthetők a visszaélések. Az elemző algoritmusok képesek a csalásokra jellemző minták megtalálására és a minták alapján a csaló vagy jogosulatlan tranzakciók valós időben történő kiszűrésére. A rendelkezésre álló nagy, lokális és országos szintű adathalmazok összevetésével pontos következtetésekre lehet jutni az állampolgárok adózási szokásait illetően. Ki lehet szűrni a szokatlan viselkedési formákat, ami visszaéléseket jelenthet. A korábban felállított minták alapján meg lehet határozni a gyanús tranzakciók statisztikai paramétereit, és ezeket azután folyamatosan lehet figyelni. A szociális és demográfiai adatok felhasználásával el lehet dönteni, hogy a kiszűrt esetek valóban visszaéléseket jelentenek-e, például egy támogatásban részesülő személy valóban jogosult-e a segélyre.

Biztonság

A rendvédelem és az igazságszolgáltatás szempontjából fontos, hogy tudják, mikor és hol történik erőszakos bűncselekmény, vagy hol készülnek elkövetni ilyet. A világon a rendőri szervek sok helyen a múltbeli adatok és számos egyéb adatbázis felhasználásával olyan algoritmusokat alkalmaznak, amelyek előre jelzik a bűncselekmények bekövetkezésének körzeteit. Az információcentrikus megközelítés növeli a bűnüldözés hatékonyságát. Ez országos szinten is igaz, a kormányok a big data technológiával sikeresebben tudják előre jelezni a várható bűncselekmények, terrortámadások bekövetkezését. A legismertebb big data előrejelző rendszer az USA-ban alkalmazott PRISM, amely a Microsoft, Yahoo!, Google, Facebook, Paltalk, YouTube, AOL, Skype és az Apple felhasználói adataira építve készíti elemzéseit, nem kevés adatvédelmi vitát is kiváltva ezzel. A fenti tevékenységhez hasonló adatgyűjtést az EU adatvédelmi szabályozása nem tesz lehetővé, de az USA-ban érvényes szabályozás erre lehetőséget ad.[8]

Az USA-ban az utak mellett vagy felett elhelyezett, rendszámfelismerésre képes kamerák segítségével nyomon követhetővé válik gyakorlatilag bármelyik közúti jármű. Magyarországon az autópálya-matrica rendszer működtetéshez is szükség van erre, és a tervekben szerepelő úthasználat-arányos fizetéshez, vagy a működő EKÁER-hez a rendszámok automatikus felismerése elkerülhetetlen.

Általánosan elfogadott vélemény, hogy a big data kormányzati alkalmazása rendkívül hasznos, a fejlődés egyik motorja, ugyanakkor biztosítani kell az adatokhoz való hozzáférés és az azokból levont következtetések gyakorlati alkalmazásának ellenőrzését. Nem

7 https://www.theguardian.com/world/2012/feb/17/obama-digital-data-machine-facebook-election

8 https://www.techspot.com/news/52823-us-government-confirms-prism-surveillance-program-tech-companies-deny-involvement.html

engedhető meg, hogy valaki – csupán azon az alapon, hogy egy rendszer jelezte, hogy adatai megfelelnek egy tipikus bűnelkövetői mintának – gyanúsítottá váljon.

Egészségügy

Egyes országokban az egészségügyért felelős szervezetek a közösségi médiában is valós időben követik a betegségek terjedésének adatait. A Google éveken keresztül közölt influenza és dengue-láz trendeket számos országra, kizárólag a keresési adatok alapján. Később ezt beszüntették, jelenleg tanulmányozzák a fejlesztési lehetőségeket.[9] A kormányzati szervek azonban továbbra is közölnek heti szintű influenza terjedési adatokat az egész világra vonatkozóan.[10]

Az USA-ban 2012-ben indították útjára az egészségügy területén a Big Data to Knowledge (BD2K) programot, amely az adatközpontú gondolkodás és innováció támogatását célozta meg az egészségügy és a szociális szféra területén.

Gazdasági, pénzügyi tervezés

Gazdasági elemzések készítése: a sokféle adatforrásból nyert adatok segítik a közgazdászokat a pontosabb pénzügyi előrejelzések készítésében.

Okos városok, a dolgok internete (Internet of Things – IoT)

A közszféra egyre több olyan alkalmazást használ, amely szenzorok mérésein alapul, például forgalommérés, környezetszennyező anyagok koncentrációja, a szemetes konténerek telítettségi szintjének mérése, a közfeladatot ellátó járművek helyzete vagy a szokásostól eltérő szituációk, anomáliák. Az IoT-eszközökről származó adatok gyors elemzése potenciálisan hozzájárul a városgazdálkodás, a közlekedésszervezés és a biztonság színvonalának emeléséhez és az állampolgárok életminőségének javításához.

Kiberbiztonság

A kormányzati számítógépeken tárolt és feldolgozott szenzitív vagy kritikus adatokat is tartalmazó nagy adatállományok elemzése elősegíti kibertámadások megfigyelését és kivédését.

9 http://www.google.org/flutrends/about/
10 https://www.cdc.gov/flu/weekly/fluactivitysurv.htm

Az Európai Unió és a big data

Az EU számos dokumentuma szerint az adat a gazdaság egyik fő erőforrása.[11]

A geográfiai, statisztikai, időjárási, kutatási, közlekedési, energiafogyasztási, egészségügyi adatokban rejlő hasznos információ kinyerése és felhasználása azonban technológiai innovációt, új eszközöket és szaktudást tesz szükségessé. Az adatértéklánc felhasználása a jövő tudásalapú gazdaságának alapja. Az újfajta analitika a hagyományos gazdasági ágazatokban is újabb üzleti lehetőségeket teremt:

- innovatív informatikai szolgáltatások létrehozása,
- az üzleti intelligencia felhasználása a termelékenység javítására,
- a kutatás-fejlesztés hatékonyságának javítása,
- a személyre szabott szolgáltatások útján a költségek csökkentése,
- a közszolgáltatás hatékonyságának javítása.

Az EU 2014-ben kidolgozta az adatvezérelt gazdaság stratégiáját[12], 2017-ben megjelentette az *Európai Big Data Gazdaság Építése* című közleményt[13] és az ezt kísérő Munkadokumentumot[14].

A dokumentumok célul tűzik ki a kutatási és innovációs tevékenységek finanszírozását, új kutatások indítását. Az európai Ipari Digitalizáció öt fejlesztési prioritása közül az egyik az adatfeldolgozási technológia szabványosítása.

A *Connectivity for a European Gigabit Society* közlemény a konnektivitást az adatalapú gazdaság alapvető tényezőjének tekinti. Az adatalapú gazdasággal kapcsolatos EU-s politikát fejezi ki az Európai Parlament és a Tanács 2003/98/EK Irányelve a közszféra információinak további felhasználásáról.[15]

A dokumentum megállapítja: „A közigazgatási szervek gyakorlata a közszféra információinak felhasználására vonatkozóan nagyon eltérően alakult" az egyes országokban és ez hátráltatja gazdaság növekedését. Ennek a helyzetnek a kezelésére tesz javaslatot a dokumentum. Lényegében megfogalmazza az Open Data koncepcióját és a felhasználás szabályozását.

A dokumentumok szerint a big data a gazdasági növekedés, a versenyképesség, a munkahelyteremtés és a társadalmi fejlődés egyik legfőbb erőforrása. Az EU adatgazdaságának mérete 285 milliárd EUR volt 2015-ben, a teljes EU GDP majd 2%-a. A megfelelő szabályozási feltételek és az IKT-szektornak kedvező beruházási politika mellett az EU-ban ez 2020-ra 739 milliárd EUR-ra nőhet. Ehhez arra van szükség, hogy az adatok az EU-s határokon és a különböző szektorokon keresztül szabadon áramoljanak és az adatokat a releváns szereplők fel tudják használni. Az Egységes Digitális Piac stratégiájának megfelelően szükséges a koordinált európai megközelítés. Meg kell szüntetni az adatok helyhez

11 https://ec.europa.eu/digital-single-market/en/big-data
12 https://ec.europa.eu/digital-single-market/en/news/communication-data-driven-economy
13 https://ec.europa.eu/digital-single-market/en/policies/building-european-data-economy
14 https://ec.europa.eu/digital-single-market/en/news/staff-working-document-free-flow-data-and-emerging-issues-european-data-economy
15 https://eur-lex.europa.eu/legal-content/HU/TXT/PDF/?uri=CELEX:32003L0098&from=ENGeneral

kötöttségére vonatkozó indokolatlan és aránytalan korlátozásokat, biztosítani kell az adatok szabad áramlását és ki kell alakítani az adatok elérésére és továbbítására, az adatok hordozhatóságára és a nem személyhez köthető, gépi adatok felelősségére vonatkozó jogi szabályozást.

Az EU Bizottsága 2017-ben előkészített egy szabályozási tervezetet a nem személyes adatok EU-n belüli szabad áramlásáról.[16]

A tervezet szerint a szabályozás:

- biztosítani fogja a nem személyes adatok az EU-n belüli szabad áramlását, azaz a szervezetek bármely EU-tagországban tárolhatják ezeket az adatokat,
- kötelezővé teszi az adatok elérhetőségét a szabályozói ellenőrzésekhez, vagyis egy ország hatóságai ellenőrizhetik az adatokat akkor is, ha azok más országban vagy a felhőben vannak,
- arra ösztönzi a felhőalapú szolgáltatókat, hogy könnyítsék meg a szolgáltatóváltást önszabályozó alapon, azaz saját maguk alakítsanak ki olyan szabályozást, amely lehetővé teszi az egyszerű szolgáltatóváltást vagy a felhasználó saját adatainak könnyebb visszavételét a szolgáltatótól,
- teljes kiberbiztonsági megfelelőséget ír elő az EU területén és a felhőben.

A Bizottság 2018-ban újabb javaslatot készít a közszférában keletkező vagy közpénzből finanszírozott módon keletkező adatok elérésére és további használatára. A javaslat a 2003-as Direktíva 2013-as módosításának eredményeit tekinti át, amely azt javasolja a tagállamoknak, hogy a lehető legtöbb, a közszférában keletkező adatot tegyenek elérhetővé az állampolgárok és a vállalkozások számára.

Megjegyezzük, hogy a 2003-as direktíva óta eltelt 15 év alatt számos EU-s tagországban nem oldották meg kielégítően a közszolgálatai adatokhoz való felhasználóbarát hozzáférést, és az adatok szabványosítása is lassan halad, az elektronikus, továbbhasznosítható formában történő elérés elmaradt akár az üzleti szférában létező lehetőségektől, akár az Egyesült Államokban létező *data.gov* és más nyílt adatelérési szolgáltatásoktól.

A korszerű adatelemzési módszerek közigazgatási alkalmazása világszerte elmarad a gazdasági szférában tapasztalttól, sőt, a hagyományos adatbányászati módszerek alkalmazását sem tekinthetjük általánosan elterjedtnek. A várható haszon azonban egyre inkább tudatosul a közszféra döntéshozóiban is, ezért a korszerű elemzési módszerek, eszközök elterjedése a közeljövőben várható.

A big data eszközei

A big data adatkezelés és elemzés eszközkészlete az elosztott feldolgozás lehetőségeit használja ki. Az elosztott feldolgozás teszi lehetővé az esetenként igen nagymennyiségű, változó szerkezetű adat valós időben történő rögzítését és elemzését.

Az alábbiakban ismertetjük a big data feldolgozás „ökoszisztémáját”, azokat a szoftvereket, amelyeket az adatok tárolására, továbbítására és elemzésére használnak. Jelentős

16 https://ec.europa.eu/digital-single-market/en/free-flow-non-personal-data

részük nyílt forráskódú szoftver, amelyet eredetileg egy nagy technológiai vállalat, elsősorban a Google fejlesztett, majd adott át a nyílt forráskódú szoftvereket gondozó és karbantartó közösségnek, túlnyomó részben az Apache Projektnek. A vállalatok többnyire maguk is ezeket a nyílt forráskódú programokat használják saját rendszereikben vagy disztribúciójukban.

Az elosztott feldolgozás

A történészek szerint elosztott feldolgozás már a Római Birodalomban létezett, amikor Augustus császár az adózás ellenőrzésének céljából bevezette a népszámlálást.[17] A császár elhatározta, hogy mindenkit megadóztat a Birodalom területén, de ehhez először számba kellett venni az akkor már hatalmasra nőtt területen élő embereket. Észak-Afrika, a mai Spanyolország, Németország, Görögország, Perzsia, Szíria, Izrael a Birodalom részei voltak. A rómaiak hamar rájöttek, hogy a számlálást nem lehet belátható időn belül központilag végrehajtani, így kineveztek egy népszámláló testületet, melynek tagjai egy adott napon felkeresték a számukra kijelölt területet, végrehajtották a helyi számlálást, majd visszatértek Rómába, ahol az eredményeket összesítették. Ezzel sikerült hatékony megoldást találniuk a szétszórt, de nagy létszámú népesség összeszámolására, a számlálóbiztosokat küldték az adatokhoz, és nem az adatokat gyűjtötték be központi feldolgozásra.

A római népszámlálási elv szerint, ha nagymennyiségű adatot kell feldolgozni, akkor nem célravezető a processzálás centralizálása, hanem a feldolgozó egységeket kell az adatokhoz elvinni. Így a feldolgozható adatok mennyisége elvileg korlátlan. Ezt az elvet használják a big data architektúrák is.

A számítástechnikában nem újkeletű az elosztott (pl. a párhuzamos) feldolgozás. Már az 1950-es évek végén megszületett az a gondolat, hogy ha egy számítási feladat túl sokáig tart, de egymástól független részekre bontható, akkor ezek a független részek különböző eszközökön egymással párhuzamosan is végrehajthatók, majd a keletkező eredmények összesíthetők (Gill, 1958).

A párhuzamosításnak többféle szintje[18] is van, a továbbiakban a feladatok párhuzamosításával foglalkozunk, mert a big data ökoszisztémának ez az alapja. Lényegében ez azt jelenti, hogy ha hasonló adatokon ugyanazokat a műveleteket kell elvégezni, akkor a feladatok szétoszthatók több processzorra és a feldolgozás egyidejűleg is történhet. Nyilván ezzel a módszerrel felgyorsítható a feldolgozás. Természetesen nem minden feladat elosztása vezet a hatékonyság javulásához. Ha a részfeladatok csak meghatározott logikai sorrendben oldhatók meg, akkor egy feladat csak akkor indítható el, ha a megelőző már befejeződött, ez esetben a párhuzamosítás nem gyorsítja fel a megoldást.

Az informatikai feladatok között vannak jobban és kevésbé jól párhuzamosíthatók, és így az elosztott feladatmegoldás nem minden esetben vezet hatékonyságjavuláshoz.

17 Big Data and the Roman Census Approach. http://tdan.com/big-data-and-the-roman-census-approach/17244

18 Például többmagos processzorok, közös memóriával működő processzorok, többprocesszoros gépek, több számítógépre elosztott feldolgozás.

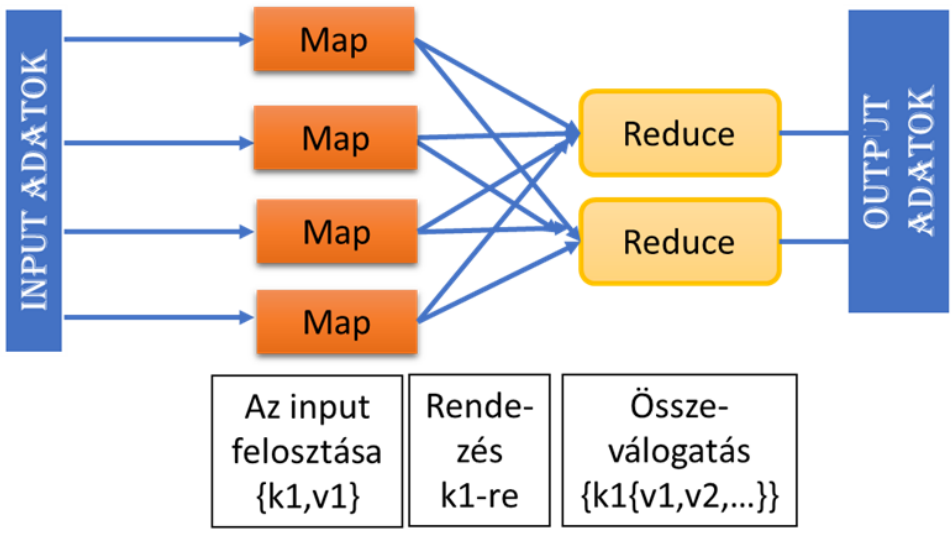
A Hadoop és elemei

Az eloszthatő feladatok tipikus alkalmazási példája a keresőmotorok működése. A keresőmotorok folyamatosan szkennelik az interneten számukra elérhető adatokat, az oldalakat indexelik és egy adott szintig el is tárolják, hogy amikor beütünk egy kereső kifejezést, akkor azonnal tudjanak válaszolni anélkül, hogy el kellene kezdeni keresni a neten. Sok százmillió webes oldallal végzik ugyanazt a műveletet. Nem véletlen, hogy a big data adatok párhuzamos feldolgozásának legjelentősebb úttörője a Google, de a Google-nál felmerülő feladatokhoz hasonlóak más technológiai vállalatoknál is jellemzők.

A Google saját fejlesztéseit publikussá tette, dokumentálta, átadta az Apache Projektnek és ezzel lerakta a nyílt forráskódú Hadoop ökoszisztéma alapjait. Ennek az ökoszisztémának a nagyobb elemei a Hadoop és MapReduce, a Hbase, a Mesos, a Zookeeper és a Drill, ezt kiegészíti számos kisebb programrendszer, amelyet vagy a Google, vagy más vállalkozás, vagy akadémiai intézet fejlesztett. Jelenleg azt látjuk, hogy a nyílt forráskódú big data ökoszisztéma nemzetközi fejlesztői együttműködésben gyorsan fejlődik, és a szoftvergyártó cégek legnagyobb része is a nyílt forráskódú szoftverek használatát preferálja.

A Hadoop–MapReduce koncepció és szoftver létrejöttét – amint az előzőekben írtuk – a robbanásszerűen növekvő webes tartalom indexelési igényének köszönheti. Az indexelési feladat motiválta a Google fejlesztőit a MapReduce megoldás kidolgozásában, amely alkalmas nagymennyiségű adat feldolgozására közönséges, kereskedelmi forgalomban kapható számítógépek klaszterein. Minthogy a Google szerver stratégiája egyértelműen az olcsó, kereskedelmi forgalomban is beszerezhető szerverek millióira épül, a fejlesztésnek is ez volt a fő irányvonala.

A MapReduce programozási modellje nagy adathalmazok feldolgozását támogatja. A felhasználók meghatároznak egy leképező függvényt (Map), amely a processzálást végző gép inputján megadott kulcs/érték párokat dolgoz fel, majd ebből új kulcs/érték párokat állít elő az outputon, és egy aggregáló függvényt (Reduce), amely összesíti az azonos output kulcsokhoz tartozó értékeket (ld. 1. ábra). Az ebben a rendszerben megírt programok párhuzamosan működnek, és sok gépből álló klaszteren futhatnak. Mindehhez azonban új adattárolási modellekre is szükség volt, mert a hagyományos relációs adatbázis-kezelők erre a feladatra nem hatékonyak. A Hadoop/MapReduce és hasonló feldolgozási megoldásokhoz kifejlesztették a NoSQL (Not only SQL) adatbázis-kezelő rendszereket (ld. a fejezet további részében).



1. ábra. A MapReduce munkafolyamat egyszerűsített sémája
(Forrás: Saját szerkesztés.)

A MapReduce modellt 2004-ben tették közzé (Dean et al., 2004). Ezután egy évvel létrejött az Apache Hadoop program és a Hadoop Infrastruktúra, amely jelenleg már messze túlmutat a webes tartalmak indexelésén, a legkülönbözőbb alkalmazási területeken, ahol nagymennyiségű, nem strukturált adatot kell gyorsan feldolgozni, a Hadoop a legelterjedtebb eszköz.

A Hadoop és a MapReduce működését az alábbi példán szemléltetjük.

A feladat szavak előfordulásának kiszámítása egy dokumentumban. A dokumentumot részekre – blokkokra – osztják, majd az egyes blokkokat eljuttatják egy feldolgozó egységhez, ahol a Map funkció minden szóhoz hozzárendel egy számértéket (kulcsot), majd a Reduce funkció minden kulcshoz hozzárendeli, hogy az adott szó hányszor fordult elő.

A Map és a Reduce programkódot a felhasználó készíti el, majd átadja a MapReduce keretrendszernek, amelyet C++-ban írtak. A felhasználói Map és Reduce programokat gyakorlatilag bármely ismertebb programozási nyelven meg lehet írni, a keretrendszer ezt kezelni tudja.

A MapReduce input és output oldalon is karaktersorozatokkal működik, a felhasználó dolga, hogy ezeket azután a saját igényeinek megfelelő adattípussá alakítsa.

Az eljárás leghatékonyabban a párhuzamosítható feladatokban alkalmazható. Ilyen feladat számos webes alkalmazás, mint például nagy adatállományok rendezése, vagy keresés nagy adatállományokban.

Ahhoz, hogy a MapReduce modell valóban hatékony legyen, a futtatáshoz megfelelő számítógépes architektúra szükséges. Az architektúra lehet egy kisebb számítógép is, amelyen a memóriát osztják meg logikailag, de lehet egy igen nagy kapacitású, többprocesszoros számítógép, vagy számítógépek hálózata, a feladattól függően. A leghatékonyabb releváns architektúra a Google implementációja. A Google adatközpontjaiban a hardver a kereskedelmi forgalomban kapható PC-k kapacitásával összemérhető, egy-két processzoros gépek tízezreiből álló klaszterek, gépenként néhány GB-os RAM-mal és normál merevlemezzel.

A használt operációs rendszer a Debian, a Linux egy változata. A gépek hálózatban működnek, többnyire Gigabites Etherneten. Becslések szerint a Google adatközpontjaiban 2,5 millió ilyen szerver működik összesen. Az adatokat egy elosztott fájlrendszerben tárolják, redundáns módon, növelve a megbízhatóságot és elérhetőséget az átlagos megbízhatóságú hardveren.

A felhasználók a feladatokat egy ütemező szoftvernek küldik, amely a mappelést az alábbi módon hajtja végre az éppen szabad kapacitással rendelkező gépekre:

- Az input adatfájlt a MapReduce könyvtár 16–64MB nagyságú darabokra vágja szét (a méret paraméterezhető), majd a klaszter gépein futtatja a Map példányait.
- A példányok között van egy kitüntetett, a master, a többi a munkás (worker). A master osztja szét a feladatot a munkásoknak. Kétféle task van, a Map és a Reduce. A master figyeli, hogy melyik munkásnak nincs munkája, és ezeknek egy map vagy egy reduce feladatot ad.
- A map feladattal rendelkező munkás beolvassa a rá eső input blokkot, elemzi a kulcs/érték párokat, majd átadja a felhasználói Map programnak. A kimenő kulcs/érték párokat a memóriában tárolja, amelyeket időszakonként kiír a helyi lemezre, egy megfelelő partícióba. Minden Reduce példányhoz hozzá van rendelve egy partíció. A rendszer a lokális lemezre írt párok címét elküldi a masternek, amely továbbítja ezeket a Reduce munkásoknak.
- Amikor egy Reduce worker megkapja a párok címét, elolvassa azokat a lokális lemezekről. Amikor az összes adatot beolvasta, rendezi az állományt a kulcsok szerint, az azonos kulcshoz tartozó adatokat csoportosítja. Amikor egy kulcshoz kigyűjtötte az összes értéket, azokat átadja a felhasználó által írt Reduce programnak. Ez hozzáadódik a partícióhoz tartozó eredményfájlhoz.
- Amikor minden map és Reduce program példány befejezte a működését, a master „felébreszti” a felhasználói programot. A felhasználói program maga dönti el, hogy a Reduce példányok outputjait összesíti-e vagy továbbküldi egy másik MapReduce eljárásnak, vagy más olyan programnak, amely kezelni tudja a particionált eredményt.

Ez az algoritmus természetesen nem terjed ki a rendszer összes funkciójára. Számos segédfunkció és igen hatékony hibatűrő eljárás segíti a működést, például egy vagy több gép kiesése esetén más gépek azonnal át tudják venni a meghibásodott gépek munkáját.

A HBase adatbázis-kezelő rendszer

A Google működése során hatalmas adatmennyiséget olvas be a hálózaton elosztott szerverek százezreiről. Minthogy a relációs adatbázis-kezelő rendszerek nem hatékonyak ilyen mennyiségű tranzakció rövid idő alatt történő kezelésére, a feladatra új típusú, elosztott tárolási rendszert dolgoztak ki, amely alkalmas nagy mennyiségű, akár stukturálatlan adatok kezelésére is. Ezt Bigtable-nek nevezték. A rendszer több petabájtnyi adat kezelésére képes, az adatokat közönséges szerverek ezrein tárolja.

Számos Google alkalmazás használja a Bigtable megoldást, például a személyes keresések, de ilyen a Google Earth, a Google Analytics és a Google Finance is.

A Bigtable 2006-ban megjelent dokumentációja alapján készítette el az Apache Projekt a nyílt forráskódú verziót, a HBase-t a Hadoop rendszerhez.

A HBase NoSQL típusú adatbázis-kezelő, azaz nem csak relációs adatbázis-kezelő, nem támogatja teljes mértékben az SQL típusú lekérdezéseket. Számos, egy gépen használatos NoSQL típusú adatbázis-kezelő rendszer létezik, de a HBase ezektől abban különbözik, hogy kifejezetten támogatja az elosztott, mégpedig nagyon sok gépre elosztott tárolást. A HBase-t inkább adattárolónak, mint adatbázisnak lehetne nevezni, mert több, a relációs adatbázis-kezelőknél megszokott funkciót nem támogat. Nem támogatja például a másodlagos indexelést, a triggereket, a magas szintű lekérdező nyelveket.

Ezzel szemben a HBase támogatja mind a lineáris, mind a moduláris skálázást. A HBase klasztereket ún. regionális szerverek hozzáadásával lehet bővíteni, amelyek közönséges, kereskedelmi forgalomban kapható szerverek is lehetnek. Ha a regionális szerverek számát megduplázzuk, ez megduplázza a tárkapacitást és a feldolgozási kapacitást is. Ez a fajta skálázás a relációs adatbázis-kezelő rendszereknél általában korlátozott, egy adott méret után a bővítéshez már speciális típusú hardverek szükségesek.

Ha az adatmennyiség növekszik, a HBase automatikusan osztja tovább az adatokat a regionális szerverek között.

A HBase támogatja a tömegesen párhuzamosított feldolgozást MapReduce-szal, kiszolgálja mind az input, mind az output oldalt, vagyis a Mapreduce közvetlenül olvas és ír a HBase-ben. Rendelkezik JAVA API-val a JAVA programból történő eléréshez és Thrift/REST API-val más programozási nyelvekből történő híváshoz.

A HBase – számos jó tulajdonsága ellenére – nem minden probléma megoldására alkalmas. Előnyei nagy adatmennyiségeknél, százmilliós vagy milliárdos adatállományoknál jelentkeznek. Ha „csak” százezer vagy egymillió adatsorunk van, akkor egy relációs adatbázis-kezelő rendszer tökéletesen megfelel, ez esetben a HBase elosztó algoritmusa ugyanis nem ad munkát a klaszter nagyobbik részének. Ha egy RDBMS funkcionalitására van szükség (ld. hiányzó funkcionalitásokat), akkor a HBase nem jó választás, ui. az adatmigrálás egy RDBMS és a HBase között teljes újratervezést tesz szükségessé.

A Mesos erőforrás menedzsment rendszer

Az ökoszisztéma következő eleme a *Mesos*, amely a Google hatalmas szerverállománya és 30 körüli adatközpontja munkájának erőforrás menedzsere. A Mesos osztja szét a feladatokat az erőforrások között. A Google nem épített specializált klasztereket a különböző feladatokra, ahogyan ez a vállalati rendszerekben általánosan elfogadott stratégia, hanem minden klasztere minden feladatra alkalmas, a Mesos a feladatot a szabad erőforrásokra tudja küldeni. A rendszer fő jellegzetessége, hogy egy teljes adatközpontot egyetlen erőforrásként is tud kezelni. A Mesos nyílt forráskódú változatát (Apache Mesos) mások is használják, pl. a Twitter és az Airbnb.

A Zookeeper rendszermenedzser

A Google egyszerre sok online szolgáltatást nyújt, miközben a feladatokat kis darabkákra bontja és szétosztja a sok gépből álló hálózatra. Ehhez kialakították az akkor Chubby (magyarul: pufók, dundi) nevű szoftvert, amely szinkronizálta a gépek működését. Kezelte az

egyes gépek megnevezéseit, a konfigurációmenedzsmentet, az üzeneteket sorba rendezte, ill. figyelmeztetéseket küldött. A Chubby leírása és forráskódja a szokásos módon átadásra került a nyilvánosságnak, és Zookeeper néven önálló Apache Projektként vált ismertté. A Zookeeper eredeti fejlesztőjének többen a Yahoo-t tekintik (ld. Wikipedia), mindenesetre a Zookeeper ma már egységes és fontos Apache Projekt. Számos nagy cég is használja, pl. az eBay, a Yahoo.

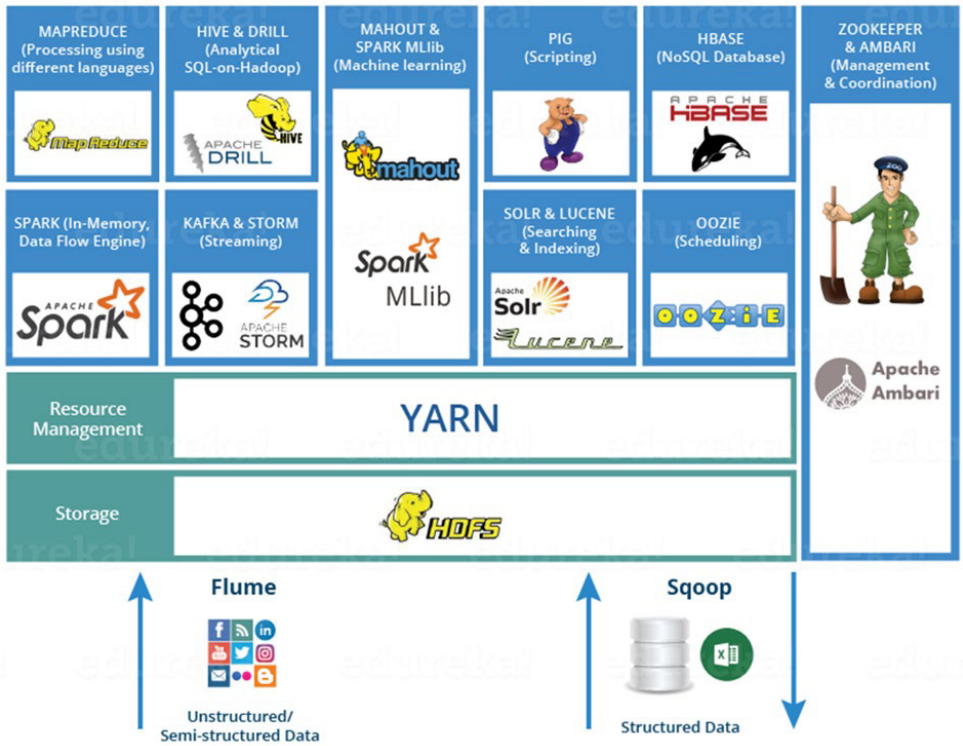
A Zookeeper skálázhatóvá teszi a rendszert, a bővítéshez elég új gépeket illeszteni a rendszerhez. A Zookeeper figyeli a gépek működését és meghibásodás esetén a hibás gépet egy működővel helyettesíti. Együttműködik a HBase-zel és más, elosztott rendszerekkel. Ismert felhasználói a Rackspace, Yahoo, eBay.

A Drill lekérdező

A Google eredetileg Dremel néven futó programja big data adattárak gyors lekérdezésére szolgál. A Google a Dremelt 2010-ben adta át az Apache Projektnek, jelenleg Apache Drill néven érhető el mint nyílt forráskódú program. Lényege az alacsony késleltési idő akár több ezer egyidejű lekérdezés esetén. Hadoop környezetben, NoSQL adatbázisok lekérdezésére alkalmas. Kezeli az önleíró adathalmazokat (JSON). Számos fájlformátumot és adatforrást képes feldolgozni. Kiemelkedő tulajdonsága, hogy számos NoSQL adattárral képes együttműködni (HBase, Google Cloud Storage, Amazon S3, Azure Blob Storage, HDFS stb.). Amint látjuk, nem csak a Hadoop ökoszisztémában, hanem a nagy szolgáltatók, mint az Amazon, a Google, a Microsoft, felhőiben is használható.

Kilépve a Google fejlesztési köréből, és általánosságban vizsgálva a jelenlegi big data infrastruktúrát, elmondhatjuk, hogy a Hadoop környezet a domináns, bár nem az egyetlen. A Hadoop környezet létrehozásához minden elem az Apache Projekt nyílt forráskódú termékeként dokumentációval együtt ingyenesen beszerezhető.

Az Apache Hadoop ökoszisztéma elemeit a 2. ábra szemlélteti.



2. ábra. A Hadoop ökoszisztéma
(Forrás: http://tdk.bme.hu/VIK/DownloadPaper/Alkalmazasfuggetlen-Big-Data-eroforras-elosztas)

A Hadoop további részei:

- A Hadoop alapcsomag (Hadoop Common Package), melynek részei az operációs rendszer és fájlrendszer szintű absztrakció, ez a gyakorlatban azt jelenti, hogy az operációs rendszert és a fájlrendszert illeszteni lehet a Hadoophoz, a MapReduce motorhoz, és a Hadoop Distributed File System-hez (HDFS). Tartalmazza még a Java Archive (JAR) fájlokat és a Hadoop futtatásához szükséges szkripteket. Az operációs rendszer szintű absztrakció lehetővé teszi, hogy a Hadoop ne csak natív Linux rendszeren fusson, hanem pl. Windows környezetben is.
- Hadoop Distributed File System (HDFS): elosztott fájlrendszer, amely az adatokat közönséges számítógépeken tárolja. A gépek között nagy sávszélességű hálózat működik.
- Hadoop YARN[19]: erőforrás menedzsment platform, ami a klaszterek erőforrásainak menedzseléséért és a feladatok elosztásáért felel.

19 A YARN a MapReduce továbbfejlesztett változata, vagy MapReduce 2.0.

A teljes ökoszisztéma megértéséhez szükséges a fenti három elem részletesebb ismertetése:

A HDFS fájlkezelő rendszer

A HDFS skálázható, portábilis, Java nyelven írt program. A HDFS-ről szóló szakirodalom a szerver gépeket node-oknak (egy gráf csúcsai) nevezi. Egy Hadoop alkalmazás tipikusan egy ún. namenode-ot és több datanode-ot tartalmaz, ezek együttesen alkotják a HDFS klasztert. A datanode-ok adatblokkokat tudnak küldeni a hálózatra egy HDFS specifikus protokollal. A fájlkezelő rendszer a TCP/IP réteget használja kommunikációra. A kliensek Remote Procedure Call (RPC)-t használnak az egymás közötti adatátvitelre.

A HDFS nagy fájlok tárolására készült, a gigabájt és terabájt közötti méretekben, ahol a tárolás sok számítógépen történik. A megbízhatósági követelmények miatt a HDFS alapbeállításban 3 másolatot készít minden adatblokkról, ezek közül az egyiket nem ugyanazon a racken lévő gépen tárolja, mint a másik kettőt. Ezzel a tárolási móddal kiváltják a jóval költségesebb RAID hibatűrő megoldást. Egy adatblokk 64 MB. A HDFS igyekszik minden adatblokkot más gépen tárolni.

A datanode-ok kommunikálnak egymással, megosztják adataikat és a másolatok számát a megadott szinten tartják minden adatnál. A HDFS nem teljesen felel meg a POSIX[20] szabványnak, mert az alkalmazási környezet és a célok lényegesen különböznek. A nagyobb teljesítmény kárpótolja a felhasználót a szabványtól való eltérésért.

A namenode meghibásodása esetén egy ún. másodlagos namenode veszi át a szerepét. A másodlagos namenode aktívan figyeli az elsődlegeset és megadott gyakorisággal menti a directory adatokat (snapshotot készít). Erről a snapshotról tud újra indulni a klaszter a másodlagos namenode irányításával, ha az elsődleges meghibásodik, nem kell a teljes folyamatot elölről kezdeni.

Mindennek ellenére a namenode a szűk keresztmetszete a rendszernek, ennek megoldására a HDFS fejlesztői egy több namenode-os megoldáson dolgoznak.

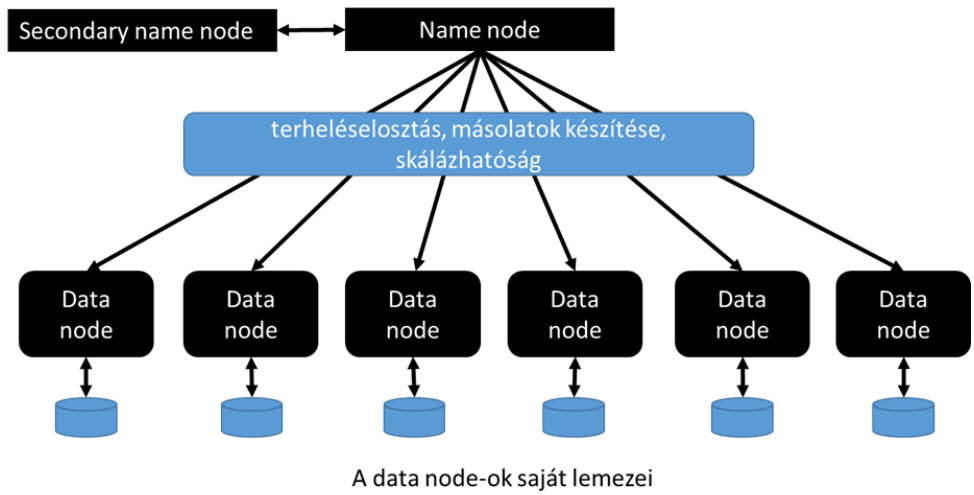
A HDFS nagy előnye a MapReduce-szal való szoros együttműködés. Ha az X datanode tartalmazza az x_1, x_2, x_3 adatot, és az Y datanode az y_1, y_2, y_3 adatot, akkor a rendszer az X-re küldi az x_1, x_2, x_3, és az Y-ra az y_1, y_2, y_3-t feldolgozó map és reduce feladatokat. Ez a megoldás csökkenti a hálózati terhelést.

Ha a Hadoophoz más fájlkezelőt használunk, ez az előny általában nincs meg.

A HDFS az olyan alkalmazásokban használható, ahol az adatokat egyszer írjuk és sokszor olvassuk. Ez jelentősen egyszerűsíti és gyorsítja a rendszer működését, hiszen nem kell gondoskodni a konkurens műveletek vezérléséről és az adatkonzisztenciáról. Ha az alkalmazás sok konkurens írási műveletet tartalmaz, akkor nem a HDFS a megfelelő fájlszerkezet.

20 A Portable Operating System Interface (POSIX) az IEEE Computer Society egy szabványcsaládja az operációs rendszerek közötti kompatibilitás kialakítására. Sokak szerint már nem felel meg a mai követelményeknek, különösen nagy mennyiségű adat feldolgozása esetén [22].

A HDFS nem működik együtt közvetlenül az operációs rendszerekkel, kivéve a UNIX-ot és a Linuxot, az adatok elérésre egyébként egy JAVA API szolgál, amelyet sokféle nyelvből lehet hívni (3. ábra).



3. ábra. A HDFS sematikus architektúrája
(Forrás: Saját szerkesztés.)

A megbízható működést szolgálja az ún. heartbeat jel, amelyet minden datanode rendszeresen küld a namenode-nak. Ha elmarad, ez azt jelzi a namenode-nak, hogy a datanode leállt vagy nem elérhető, és a namenode a datanode-ot kizárja a kommunikációból, és a feladatait átadja más datanode-oknak.

Összefoglalva, a HDFS olcsó számítógépek sokaságán képes megbízható módon nagy adatfájlokat tárolni. Használata akkor indokolt, ha az adatokat csak egyszer kell írni és többször kell olvasni.

A Hadoop/HDFS integrálható a HBase-zel, a HDFS-t tudja elosztott fájlrendszerként használni alapértelmezésként.

Ha nincs legalább 5 node, akkor nincs értelme HDFS-t használni, ui. legalább 3 replika kell minden adatblokkból különböző node-okon és egy namenode, plusz egy másodlagos namenode.

Mi a különbség a HBase és a HDFS között?

A HDFS kiválóan alkalmas nagy fájlok tárolására, de semmiképpen nem általános rendeltetésű fájlkezelő rendszer, nem teszi lehetővé egyedi rekordok gyors keresését. A HBase viszont a HDFS-re épül, és biztosítja a nagy táblákban való gyors keresést és módosítást. A HBase indexfájlokat készít a HDFS-en, hogy gyorsan lehessen egyedi rekordokat is keresni.

A továbbiakban a Hadoop ökoszisztémához vagy más Apache Projektekhez csatlakozó HIVE, PIG, Mahout és SPARK eszközökről lesz szó.

Egyéb big data szoftverek

A Hadoop ökoszisztéma része *az Apache Hive* adattárház szoftver, amelyben SQL szintaxist használva nagy, elosztott adatállományokat lehet írni, olvasni, menedzselni. A Hadoopra épül, és az SQL nyelv eszközeinek használatával a szokásos adattárház funkciókat látja el. (ETL[21], jelentéskészítés, adatelemzés). A HIVE alkalmas arra, hogy különféle adatformátumokat úgy kezeljen, hogy struktúrát épít rájuk (relációs táblákat készít), és nem teljesen strukturált adatokat is képes korszerű SQL eszközökkel kezelni, beleértve az SQL újabb adatelemző funkcióinak alkalmazását is. Közvetlenül eléri a HDFS fájlrendszert, vagy a HBase NoSQL adatbázis-kezelő adatait. A lekérdezéshez többféle Apache terméket tud használni, pl. a Sparkot vagy a MapReduce-t. A Hive bővíthető felhasználói funkciókkal is. Érdekessége, hogy nincs saját adatformátuma a tárolásnál, kapcsolókkal rendelkezik többféle fájlformátumhoz (pl. CSV, TSV textfájlok stb.), de saját felhasználói kapcsolókat is lehet definiálni.

Mire nem jó a Hive? Nem hatékony a hagyományos tranzakciófeldolgozó (Online Transaction Processing, OLTP) rendszerekben, kifejezetten adattárházas feldolgozásokra kevéssé alkalmas.

Minthogy változó big data környezetre készült, hibatűrő, skálázható, sokféle input formátumot képes kezelni és a teljesítménye nem romlik az adattömeg növekedésével.

Az Apache Pig szoftver

A Pig eredetileg a Yahoo fejlesztése, jelenleg már az Apache Projekt része, nyílt forráskódú eszköz, amely illeszkedik a Hadoop ökoszisztémába.

A Pig két fő összetevővel rendelkezik:

- programozási nyelv, amely adatfolyamok leírására alkalmas, ezeket Pig Latinnak[22] nevezik,
- a Pig Latin programok futtató környezete.

A futtató környezet kétféle, attól függően, hogy lokálisan, egy virtuális Java gépen kell-e végrehajtani a programot vagy egy Hadoop klaszteren.

A Pig Latin úgy képzelhető el, mint operátorok és transzformációk gyűjteménye, amelyek az input fájlokat outputtá alakítják. Az operátorok az adatfolyamokat írják le, a Pig futtató környezet ezeket végrehajtható formába transzformálja, majd végrehajtja.

Sokféle adattípust képes kezelni, beleértve összetett adattípusokat is.[23] Műveletei között megtalálható a rendezés, a join, a szűrők használata.

21 ETL – Extract, Transform, Load – az adattárházba kerülő adatok előkészítése és betöltése, az adattárház létrehozásának és működésének egyik alapvető összetevője.

22 A Pig Latin eredetileg egy játékos, angolul beszélő gyerekek által is használt mesterséges nyelv.

23 Az összetett adattípusok: Tuple – több típusú adat együttese, (pl. név, tajszám), bags – több, azonos típusú tuple együttese, map – kulcs/érték párok.

Legnagyobb felhasználói között megtaláljuk a Yahoo-t, amely MapReduce feladatainak 90%-át, és a Twittert, amely 80%-át Pigben fejlesztette, de ide sorolható a salesforce[24], a LinkedIn, a Nokia is.

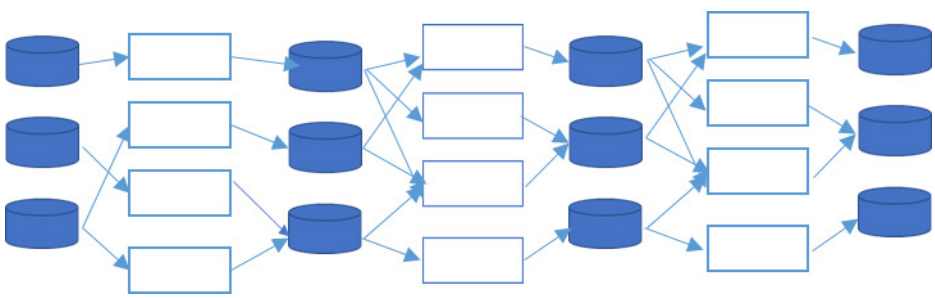
A Mahout gépi tanuló algoritmusokat tartalmazó szoftver, amely keretrendszert is biztosít skálázható gépi tanuló algoritmusok létrehozására.

A gépi tanuló algoritmusokat olyan adatelemzési feladatok esetén használjuk, ahol megfelelő, ismert adatokon „betaníthatjuk" a gépi modellünket arra, hogy a jövőre vonatkozó becsléseket tegyen. Például egy bank a korábbi hitelezőkről gyűjtött adatok alapján gépi tanuló algoritmus alkalmazásával tudja megbecsülni, hogy egy aktuális hitelkérelmező vissza fogja-e fizetni a hitelt, így a betanított modellt a hitelbírálatnál lehet használni.[25]

A Mahout számos gépi tanuló algoritmust, mint pl. a kollaboratív szűrést, a klaszterezést, a klasszifikációt és egyéb, a gazdaságban vagy a tudományban használható eljárást tartalmaz, melyekre itt nem térünk ki. Előnyös tulajdonsága, hogy egy parancssorral indíthatók a különböző, akár saját készítésű, algoritmusok is.

A *Spark* elosztott számítástechnikai környezetben működő, valós idejű adatfeldolgozásra kifejlesztett általános célú eszköz, melyet Scala nyelven írtak a Berkeley Egyetemen. Kissé kilóg a Hadoop ökoszisztémából, egyrészt nem a Google az eredeti fejlesztő, hanem a Berkeley Egyetem, másrészt sok átfedés van a Hadoop elemek és a Spark között.

Alapvető jellemzője, hogy a műveleteket főleg a memóriában hajtja végre, ezzel a MapReduce sebességének egyes esetekben akár a százszorosát is eléri. Hátránya, hogy a hatékonysághoz nagy memóriakapacitással rendelkező számítógépek szükségesek, ellentétben a Hadoop ökoszisztémával (4. ábra).



4. ábra Tipikus MapReduce működés a gyakorlatban. A lemezre írás lassú művelet (Forrás: Saját szerkesztés.)

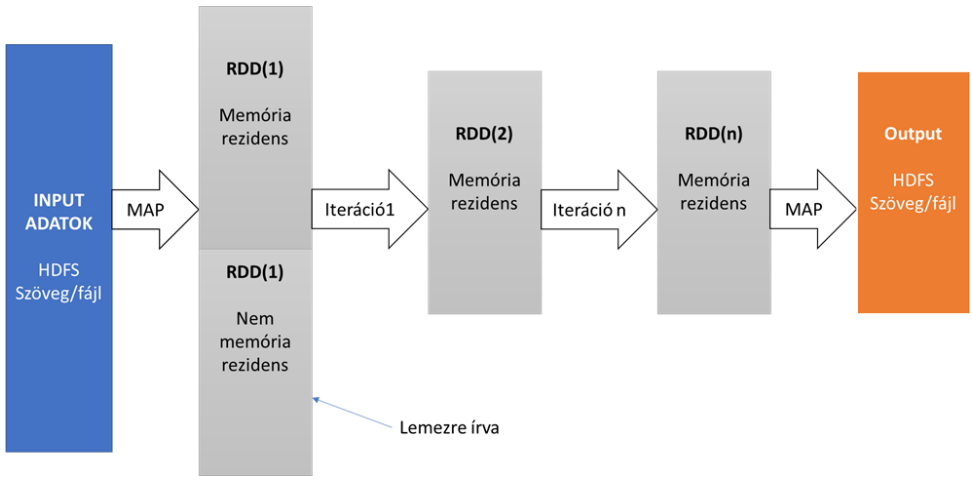
A Spark erősen kihasználja azt a tényt, hogy a memóriaműveletek kb. százszor gyorsabbak, mint a lemezműveletek. Definiálja a Resilient Distributed Datasets (RDDs) adathalmaz absztrakciót, amely a számítógép a klaszteren elosztott objektumokat egységes szerkezetben kezeli, függetlenül attól, hogy a memóriában vannak vagy a lemezen. A Spark programok

24 A salesforce a piacvezető felhőalapú CRM-szolgáltató.

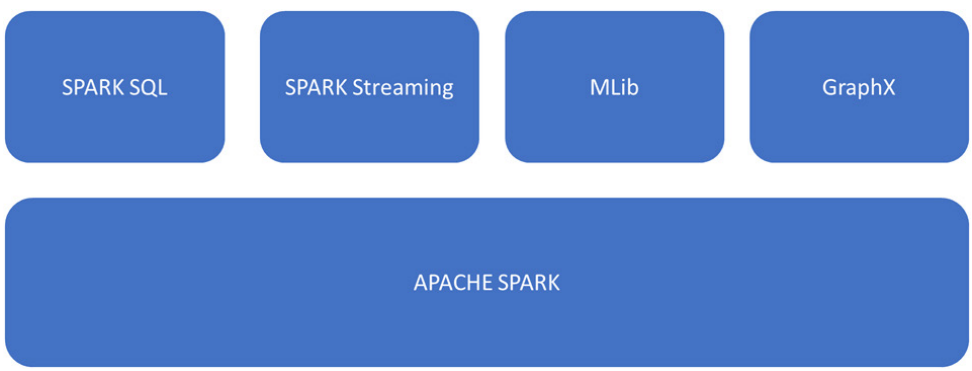
25 A gépi tanuló algoritmusok és a mesterséges intelligencia kifejezéseket sokszor szinonimaként használják. A gépi tanuló algoritmus azonban pontosan definiálható, míg a mesterséges intelligencia kifejezést sok mindenre lehet használni.

az RDD-objektumokon hajtják végre a műveleteket. Tipikus adatkezelő műveletei a map, filter, join, count, collect és több más, adatbázisok esetén ismert eljárás. Ha egy gép a Spark futtatása közben összeomlik, az RDD-k automatikusan újraképződnek. Az RDD-k megváltoztathatatlanok, a létrehozásuk után nem alakíthatók át (5. ábra).

A Spark keretrendszer biztosítja a párhuzamos és hibatűrő végrehajtást anélkül, hogy a felhasználónak ezzel foglalkoznia kellene.



5. ábra. Adatok beolvasása egy fájlból, feldolgozása Sparkban és kiírása egy HDFS fájlba
(Forrás: Saját szerkesztés.)



6. ábra. A Spark részei
(Forrás: Saját szerkesztés.)

A Spark több beépített könyvtárral rendelkezik, támogatja az SQL-t, az R nyelvet, a Pythont, Scalát, Javát stb. Ezek könnyen beilleszthetők a feldolgozási folyamatokba. A 6. ábrán látható egyéb szolgáltatások szintén részei a rendszernek.

A *Spark SQL* egy eredetileg SchemaRDD (újabban Dataframe) absztrakcióra épül, amely lehetővé teszi mind strukturált, mind részben strukturált adatok kezelését. A SchemaRDD-ket Javából, Pythonból és más nyelvekből is lehet használni a Sparkon keresz-

tül. Az SQL-t is támogatja, utasítás szintű interfészekkel rendelkezik, azaz a hagyományos SQL utasítások is végrehajthatók, ezenfelül nyílt forráskódú adatbázis API (ODBC) és Java API (JDBC) is létezik.

A *Spark Streaming* ütemező program jól alkalmazható adatfolyamok elemzéséhez, amikor is az elemzés nem utólagosan, hanem a bejövő adatokon valós időben, vagy közel valós időben, történik. Az ütemező mini-csomagokra bontja az adatfolyamot és az RDD transzformációkat a mini-csomagokon hajtja végre. A megoldás lehetővé teszi, hogy a nem adatfolyamokra kidolgozott programkódok itt újra hasznosíthatók legyenek.

Az *MLlib* elosztott gépi tanulási keretrendszer a *Spark*ban. Benchmarkok azt mutatják, hogy sokkal gyorsabb, mint más Hadoop gépi tanulási algoritmusok, amelyek nem használják a memóriaalapú megoldásokat. Az MLlib számos statisztikai és gépi tanulási algoritmust használ:

- alapvető statisztikai számítások, véletlen mintavétel, korrelációszámítás, hipotézisvizsgálatok,
- lineáris, logisztikus regresszió, döntési fák, naive Bayes,
- kollaboratív szűrés (ld. ajánlórendszerek),
- klaszterezés,
- dimenziócsökkentési eljárások (pl. főkomponens analízis)
- stb.

A *GraphX* elosztott működésű gráfkezelő program, amely gráfok kezeléséhez ad API-t a Pregel (a név a Parallel, a Graph és a Google szavakból származik) dokumentációjának megfelelően. Az alábbiakban ismertetjük a Pregelt mint a közösségi gráfok és számos egyéb gráfalkalmazás (pl. Page Ranking) alapját (Malewicz et al., 2010).

A GraphX eredetileg a Berkeley AMPLab fejlesztése, amely nagyméretű, több milliárd csúcsot és több ezer milliárd élt tartalmazó gráfok hatékony feldolgozására készült. A Google a Pregelt használja a Page Ranking algoritmushoz. A Pregel az Apache program alapja, amely viszont a Facebook eszköze a közösségi gráfok elemzésére. A Girafe természetesen együttműködik a Hadooppal, a HBase-zel és a HDFS-szel.

A Pregel működési modell azt feltételezi, hogy nagy gráfok esetén általában az egy-egy csúcson elvégzendő műveletek száma nem jelentős, de a nagy csúcs és élszám miatt a gráf nem fér be a lokális memóriába. (A webgráf kb. 20 milliárd weboldalt tartalmaz, 400 TB terjedelemben.)

A Pregel programozási modellje iterációk sorozatából áll, ezeket superstepeknek nevezik. Egy supertstep alatt a keretrendszer minden csúcsra elindít egy felhasználó által definiált függvényt, konceptuálisan párhuzamos módon. A függvény egy V csúcson egy S supertstep alatt végrehajtandó műveleteket határozza meg. Például elolvassa a V-nek az S-1-ben küldött üzenetet, vagy üzenetet küld más csúcsoknak, amelyeket azok S+1-ben kapnak meg, vagy módosítja V és a kimenő élek állapotát. Tipikusan az üzeneteket a kimenő éleken küldi a program a szomszédos csúcsokra, de bármely más, nem szomszédos csúcsnak is tud üzenetet küldeni, ha ismeri az azonosítóját.

A Pregel egyes feladatok esetén jóval hatékonyabb megoldás, mint a MapReduce, ezért sokan a Pregelben egy új, a Hadoopot leváltó ökoszisztéma egyik elemét látják. A Page Ranking algoritmus esetén a csúcsok relevanciáját számolja iteratív módon a MapReduce:

Input:

(id1[PR_m (1)(out11,out12⋯],(id2,[PR_m(2),out21,out22,⋯]),..

Output:

(id1[PR_{m+1}(1)(out11,out12⋯],(id2,[PR_{m+1}(2),out21,out22,⋯]),..

A gráf topológiája nem változik, de a MapReduce-ban minden iterációnál újra kell tölteni és feldolgozni, ezzel processzor- és hálózati kapacitást használunk feleslegesen. A Pregel megoldási modellje ezt kiküszöböli. Az 1. táblázat összehasonlítja a Pregel és a MapReduce tulajdonságait.

Szempont	**Hadoop–MapReduce**	**Pregel**
Programozási modell	Memória megosztás	Üzenetküldés
Számítási modell	Szinkron	Szinkron
Párhuzamosítási modell	Adatok szerint	Gráf csúcsok szerint
Architektúra modell	Master-slave	Master-slave
Feladat/csúcs ütemezési modell	Pull	Push
Alkalmazási kör	Lazán kapcsolódó, jellemzően párhuzamosítható feladatok	Szorosan kapcsolódó feladatok

1. táblázat. A Hadoop–MapReduce és a Pregel összehasonlítás (saját összeállítás)

Felmerül a kérdés, hogy a Spark jobb-e vagy a Hadoop, a Pregel vagy a MapReduce?

A helyes válasz az, hogy mindegyiknek más a funkciója. A Spark jobb a valós idejű feldolgozásoknál történő alkalmazásra, míg a Hadoop a nem strukturált adatok tárolásában és a nem valós idejű, kötegelt feldolgozásában a jobb. Egy többváltozós regresszió számításnál a Spark jobb megoldás, mert az adatokat csak az első iterációnál tölti be a memóriába, utána innen használja, ami a feldolgozási időt töredékére csökkenti.

Sok nagyvállalat a kettőt együtt alkalmazza a HDFS-ben tárolt big data elemzésre. Tipikus példa erre az eBay, ahol a Spark akár 200 gépes klaszteren is futhat, amelyeken összesen 100 TB RAM és 20 000 processzormag található. A Pregel a preferált megoldás az SSSP- (Single Source, Shortest Path Problem) feladatoknál, amelyekben egy gráf meghatározott csúcsából kell a legalacsonyabb költségű utakat megtalálni a többi csúcsba.

A MapReduce vs. Pregel összehasonlításban a gráfkezelési eljárások esetén a Pregelnek számos előnye van a MapReduce-szal szemben, míg például a népszámlálási típusú feladatban a MapReduce tűnik jobb megoldásnak.

A Hadoop alkalmazása – gyakorlati észrevételek

- A Hadoop ökoszisztéma sikere a fejlesztő gárdának és a nagy támogató szervezeteknek, mint a Facebooknak, Google-nak, Yahoo-nak, a Berkeley Egyetemnek stb. köszönhető. Bárki bekapcsolódhat a fejlesztői munkába vagy megtanulhatja az eszközök használatát.
- A programok ingyenesen letölthetők, és a tanulók számára sokféle, szintén ingyenes tankönyv áll rendelkezésre az interneten.
- Ha elakadunk, vagy egy hibát nem tudunk elhárítani, a fórumokon bátran kérdezhetünk a fejlesztői és alkalmazói közösségtől.
- A Hadoop rendszerben egy-két elem ismerete kevés egy hatékony megoldás létrehozásához. Egy konkrét feladathoz, vállalati környezethez több elemet kell összeválogatni. Egy megoldáshoz azonban nem szükséges az összes elem együttműködése, a megoldandó feladat, a gépi erőforrások és nem utolsósorban a megvalósítók szakértelme határozza meg, hogy a Hadoop mely elemeit alkalmazzák együtt.

NoSQL adatbázisok

Az utóbbi években a szervezetek és személyek által generált adatok mennyisége exponenciálisan nőtt. A növekedést hasonlóan gyors tempóban követte az adatbázisok kezelésére létrehozott technológiai megoldások száma. Minthogy a keletkezett adatok tömege sokszor meghaladja az egy számítógépen tárolható és feldolgozható mennyiséget, az új technológiák elosztott rendszereken működnek. A nagy adatmennyiség kezelhetősége érdekében az új megoldások egyszerűbbek, kevesebb hasznos funkciót tartalmaznak, mint a hagyományos adatbázis-kezelő rendszerek, ugyanakkor a hibatűrő képesség és a rendelkezésre állás tekintetében is másképpen viselkednek.

Miért van szükség a hagyományos relációs adatszerkezetektől eltérő, más jellegű adattárolási és feldolgozási módokra?

A relációs adatbázisok több olyan korlátozással bírnak, amelyek egyben alkalmazásukat is leszűkítik:

- Nem alkalmasak nagymennyiségű – petabájt méretű – többféle adattípus (videók, hangfájlok, képek, szöveges dokumentumok) kezelésére.
- Nem jól skálázhatók az adatmennyiség változásával.
- A memória mérete és a processzorkapacitás a volumen növelését korlátozza egy konkrét környezetben.
- Nem terjeszthetők ki korlátozás nélkül más gépekre, az írás és olvasás művelete cache függő.
- Az adatbázis szilánkosítása (sharding), azaz darabokra osztása a működésben hibakezelési problémákat okoz.
- A relációs adatbázis-kezelő rendszerek sokszor bonyolultak.
- A konzisztencia követelménye korlátozza a skálázhatóságot.

A NoSQL adatbázis-kezelők létrehozásának célja a relációs rendszereknél jobban skálázható, azokat meghaladó teljesítményű eszközök kifejlesztése.

A megvalósítás iránya a jelentős mértékben elosztott architektúra, a sok csomóponti gépen egyidejűleg futó programok, a relációs rendszerekben használt konkurens folyamatok vezérlésére használt, a folyamatos konzisztenciát biztosító mechanizmusok lazítása, skálázható replikáció és elosztás, ami lehetővé teszi akár több ezer gépre elosztott adatok feldolgozását, és a magasabb teljesítmény elérése.

A NoSQL megnevezés a „Not Only SQL”-ből származik, vagyis a rendszer nem zárja ki a relációs adatbázis-kezelést sem. A NoSQL strukturálatlan, időben változó, előre nem kiszámítható mennyiségű adat kezelésére alkalmas, amilyenek például a Google vagy a Facebook alkalmazásokból származó adatok, ahol a gyors adatkezelés, az azonnali elemzés fontos követelmény.

A NoSQL rendszereknél nincs szükség előzetes sémadefinícióra, ezért ezeket a rendszereket sémamentes adatbázis-kezelőknek is nevezik. A NoSQL adatbázis-kezelők az alábbi kategóriákba sorolhatók:

- Kulcs/érték adatbázisok
- Dokumentum adatbázisok
- Oszlop adatbázisok
- Gráf adatbázisok

A kulcs/érték adatbázisok

Az adatbázisban kulcs/érték adatpárokat tárolnak. Ez a tárolási mód a lehető legegyszerűbb, de a gyakorlatban igen hatékony. A kulcs az attribútum, az érték a tárolt hasznos adat. Az adatokat a kulcsok használatával lehet olvasni. A sorok szerkezete rugalmas, nem definiált előre, menet közben is változhat. A tárolási modell nagyon rugalmas, akár többszörös növekedést is lehetővé tesz a rendszer újratervezése nélkül. A műveletek gyorsak, a rendszer nagy adatmennyiségek kezelésére alkalmas.

A kulcs/érték adatbázisok kis komplexitású, változó méretű adathalmazok feldolgozására különösen alkalmasak, mint például a netes keresések, látogatók követése a neten, bevásárlói kosár adatok az ajánlórendszerekben.[26] Az ajánlórendszerekben a top-N ajánlatok tárolása is ebben a szerkezetben a leghatékonyabb.

Tipikus példák a Cassandra, az Amazon DynamoDB, az Azure Table Storage és az Oracle Berkeley DB.

Erősségek: sémamentes modell, nagyon gyors feldolgozás, nagy adatmennyiségek kezelése.

Gyengeségek: alacsony konzisztencia, aggregáció hiánya.

26 Ld. ECLAT algoritmus.

A dokumentum adatbázisok

A dokumentum adatbázisok az adatokat dokumentum formában tárolják. A dokumentumok több kulcs/érték párt vagy ezek komplexebb szerkezetét tárolják. Minden dokumentumnak van egy egyedi kulcsa, amellyel hivatkozhatunk rá. A kulcs mindig egy karaktersorozat, az érték bármely adattípus vagy tömb lehet, és kulcs/érték párt is tartalmazhat. A dokumentumokat szabványos formában tárolják (JSON, XML stb.).

Egy relációs adatbázisban a hiányzó adat üres hely, míg a dokumentum adatbázisban nincs szükség az üres hely tárolására, mert minden rekord más sémával rendelkezhet.

A dokumentum adatbázisok dokumentumok tárolására, visszakeresésére és menedzselésére használatosak, alkalmasak, mint például:

- Blogalkalmazások kommenteléssel
- Strukturálatlan logok tárolása, feldolgozása
- Gépi és szenzoradatok tárolása
- Megfigyelési adatok menedzselése

Tipikus példák: Mongo DB, Couche DB.

Erősségek: rugalmas séma, gyors, tömeges írási teljesítmény, skálázhatóság.

Gyengeségek: alacsony konzisztencia, nincs komplex lekérdezés.

Gráfadatbázisok

Az adatokat gráfszerkezetben értelmezi, tárolja a gráf csúcsai közötti kapcsolatokat és más relációkat. Az adatok reprezentálása és tárolása egy gráf csúcsainak és éleinek segítségével történik, az objektumok a csúcsok, az objektumok közötti relációk az élek. Minden csúcs és él további adatokat tartalmazhat kulcs/érték pár formájában. Az egymással sok szálon kapcsolódó objektumok tárolása, követése egy relációs adatbázisban nagyon lassú, tekintettel a sok tábla olvasására, a gráf tárolási modellben ez a művelet viszont igen hatékony. A gráfadatbázisok közül léteznek olyanok is, amelyek megfelelnek az ACID[27] követelményeknek.

A gráfadatbázisok természetes alkalmazási környezete a közösségi hálók, amelyekben a felhasználók és kapcsolataik gráfot alkotnak, minden csúcs és él meghatározott tulajdonságokkal rendelkezik. További tipikus alkalmazásuk a hálózati és felhőmenedzsment, biztonsági és hozzáférési kontroll, ajánlórendszerek[28] alkalmazása. Tipikus példák: Neo4J, Titan.

27 Ld. később.

28 Az ajánlórendszerek esetén a kollaboratív szűrés alkalmazásához.

Oszlopalapú adatbázisok

Az oszlopalapú adatbázisokban a rendszer az adatokat nem egy táblázat soraiban, hanem oszlopokban tárolja, ellentétben a relációs adatbázis-kezelő rendszerekkel. Minden oszlopot önállóan kezel, és az összetartozó adatok egy oszlopban és nem egy sorban helyezkednek el. Az oszlopos tárolás nagyobb skálázhatóságot és hatékonyságot biztosít olyan lekérdezéseknél, ahol kevés oszlop érintett, ilyen esetekben több oszlopot kell kombinálni, míg a soralapú tábla esetén ehhez több sort is el kell olvasni. Az adatokat nem táblázatokban, hanem elosztott architektúrán oszloponként tárolja. Az elosztott architektúrában a kulcsok oszlopokra vagy ezek csoportjára mutatnak. A tárolási mód gyors összesítésekre ad lehetőséget.

Egy relációs adatbázis-kezelőben a sorok így néznek ki:

azonosító	termék	ár	készlet	gyártási dátum
0001	AAAA	234,5	1000	2018.02.15
0002	BBBB	325,2	2000	2018.03.24.
0003	CCCC	123,6	1500	2018.01.12.
0004	DDDD	345,3	500	2017.03.12.

Oszlopalapú adatbázis esetén ugyanezek az adatok:

AAAA:0001;BBBB:0002;CCCC;0003;DDDD:0004
234,5:0001;325,2:0002,123,6:0004;343,3:0004
1000:0001;2000:0002;1500:0003;500:0004
2018.02.13:0001;2018.03.24:0002;2018.01.12:0003;2017.03.12;0004

Minden sorhoz tartozik egy belső azonosító, amellyel a rendszer hivatkozik az oszlopra. Az oszlopok tárolása folytonos. Az oszlopalapú adatbázisok egy része oszlopspecifikus fájlokban tárolja az adatokat, pl. egy oszlop egy fájl. Az egyes adatbázis-kezelő rendszerek általában a Google Big Table implementáción alapulnak.

A Google felhőszolgáltatásként nyújtja a Cloud Bigtable-t, mint NoSQL big data adatbázis szoftvert, de ezt használja saját szolgáltatásaihoz is, mint a kereséshez, analitikához, térképekhez és a gmailhez.

Általában gyakori az IoT-eszközök adatainak kezelésében, felhasználói analitikában és pénzügyi elemzésekben való alkalmazás. Tipikus példák: HBase és BigTable rendszerek.

Erősségek: nagy átbocsátóképesség big data esetén, jól skálázható, támogatja a particionálást.

Gyengeségek: komplex lekérdezéseket nem támogat, késleltetés a kérdésekre adott válaszokban.

A fentiek alapján azt gondolhatnánk, hogy a „nagy” technológiai cégek (Facebook, LinkedIn, Google stb.) alapvetően NoSQL adatbázis-kezelő rendszereket használnak, valójában a hagyományos relációs adatbázis-kezelők szerepe legalább akkora, mint a NoSQL megoldásoké. Ez utóbbiak szerepe a viszonylag egyszerű, nagyon nagy mennyiségű, strukturálatlan adatok feldolgozásában nélkülözhetetlen, de egy hagyományos tranzakciós rendszerben, mint egy fizetési rendszer, emberierőforrás-menedzsment, pénzügyi nyilvántartás stb. a relációs adatbázis-kezelők szerepe változatlan. A NoSQL rendszerek nem váltják le az SQL megoldásokat, hanem azokkal együttműködnek és pótolják azok hiányosságait.

3. AJÁNLÓRENDSZEREK ALKALMAZÁSA A KÖZSZFÉRÁBAN

Az ajánlórendszerek kialakulása

A korszerű adatelemzés fontos területe és egyik legfontosabb mozgatórugója a gazdaságban az ügyfelek fogyasztási igényeinek előrejelzése, az előrejelzés alapján olyan ajánlatok összeállítása, amelyek feltehetően közel állnak az ügyfelek elképzeléseihez. A cél, hogy az ügyfelek minél nagyobb arányban megvásárolják az ajánlott termékeket vagy vegyék igénybe a felajánlott szolgáltatásokat.

Több felmérés egyértelműen bizonyítja, hogy az utóbbi években már az emberek fogyasztásának jelentős és egyre növekvő része, legyen szó akár fizikailag létező, vagy az online világban elérhető szolgáltatásokról, részben vagy egészében a hálózaton keresztül kommunikált ajánlatok alapján valósul meg (Ricci et al., 2010).

A már hagyományosnak tekinthető online hirdetések helyett az ügyfelek egyre inkább a kereskedő által készített ajánlások alapján tájékozódnak, ill. vásárolnak. A személyre szabott ajánlások rendszere korábban nem létezett, még az online marketing tevékenység sem irányult személyekre, legfeljebb ügyfélcsoportokra. Az ajánlórendszerek viszont személyre szabottak.

Az ajánlatok személyre szabási lehetősége a nagymennyiségű adat felhalmozódásának és a számítási kapacitások exponenciális növekedésének köszönhető. A gazdasági vállalkozások sokkal több adattal rendelkeznek ma ügyfeleik személyes adatairól, vásárlási szokásairól, preferenciáiról, az ügyfelek hasonlóságáról és kapcsolatrendszeréről, mint korábban. Tudják, ki mikor látogatott el a weboldalra, mennyi időt töltött ott, esetleg írt-e értékelést a szolgáltatásokról. A közösségi oldalakon, fórumokon, blogokon tájékozódhatnak az ügyfelek preferenciáiról.

A hatalmas adatmennyiség egyben nagy gazdasági potenciált is jelent, ha megfelelően hasznosítják. Az ügyfelek adataiból tesz szert hatalmas nyereségre a Facebook, a Google és sok más nagy technológiai vállalkozás.

Chris Anderson a *The Long Tail* című könyvében (Anderson, 2004) azt állította: „Most hagyjuk el az információ korszakát és lépünk be az ajánlások korába". Ezzel azt a helyzetet jellemezte, amikor nem mi keressük magunknak a szolgáltatásokat, barátokat, nem is hirdetések alapján fogyasztunk, hanem egy számítógépes algoritmus javasolja nekünk. Ehhez a meglátáshoz 2004-ben még vizionáriusi képességek kellettek, de jelenleg ez már nyilvánvaló.

Az első ajánlórendszert, az Information Lense nevű amerikai online szolgáltató készítette (Malone et al., 1987). Az Information Lense könyvekről, dalokról, filmekről gyűjtött be kvantitatív értékeléseket az oldal használóitól és ezek alapján készített ajánlásokat A termékek, szolgáltatások kvantitatív és kvalitatív értékelése azóta is az ajánlórendszerek

egyik legfőbb inputja. Minthogy a legtöbb ügyfél nem készít számszerű értékelést, az ajánlórendszerek implicit adatokat is használnak, értékelésnek tekintik, ha egy ügyfél megvásárolt egy terméket, elolvasta a termék leírását, kívánságlistára tette vagy bármely más személyes relációba került a termékkel.

Az ajánlórendszerek algoritmusai

Az ajánlórendszerek legegyszerűbb algoritmusa azt használja ki, hogy egy ügyfélhez demográfiai adataiban, földrajzi helyében, jövedelmében és egyéb tulajdonságaiban hasonló ügyfelek mit kedvelnek, ill. mit vásárolnak meg. Ezeket az adatokat kiegészítik a közösségi hálók adataival, esetenként adatokat vásárolnak ún. adataggregátor[29] vállalkozásoktól. Ezeknek az adatoknak alapján tesznek ajánlatot az ügyfélnek. Az ajánlórendszerek alkalmazásának legfőbb nehézségét az okozza, hogy nagyon nagy adatállományokkal kell dolgozni, amelyek azonban viszonylagosan kevés hasznos információt tartalmaznak.

Az ajánlórendszerek alkalmazása

A gazdasági területen kialakított ajánlórendszerek alkalmasak arra is, hogy a közszféra és az állampolgárok közötti kommunikációt elősegítsék és bevonják az állampolgárokat és a vállalkozásokat a helyi és az országos döntéshozatalba. A közszféra és az állampolgárok közötti kapcsolatokat három csoportba szokás sorolni:

- Kormányzat–állampolgár reláció (Government-to-citizens G2C), amely a helyi vagy országos kormányzat az állampolgároknak szóló információs és általában elektronikus szolgáltatásait jelenti. A jól működő G2C rendszer erősíti a kormányzat és az állampolgárok kapcsolatát azzal, hogy könnyen elérhetővé teszi a hivatalos dokumentumokat, csatornát nyit az állampolgári kezdeményezéseknek, és lehetővé teszi a gyors elektronikus tranzakciókat, mint például az adóbevallást vagy a személyes adatokban bekövetkezett változások bejelentését.
- Kormányzat–vállalat reláció (Government-to-businesses G2B) a kormányzat, a vállalkozások és a magánszektor más szervezetei közötti információcserét jelenti, amely segíti a vállalkozások alapítását és működtetését. A G2B szolgáltatások lehetővé teszik a vállalkozásoknak, hogy online érjék el a gazdasági szabályozásra vonatkozó információt, a működéshez szükséges elektronikus űrlapokat, a közbeszerzési rendszereket stb.
- Kormányzat–kormányzat reláció (Government-to-government G2G) a kormányzati szervek közötti elektronikus kommunikáció, amely jelentős költségcsökkentést, papírmegtakarítást és létszámcsökkentést tesz lehetővé.

Az ajánlórendszerek alkalmazása a közszférában elsősorban a G2C és a G2B kapcsolatban merül fel, mindkét esetben a kormányzat által az állampolgároknak és az üzleti vállalkozásoknak nyújtott szolgáltatások kapcsán. A szakirodalomban több tanulmány is bemutatja az

29 Az adataggregátor vállalkozások személyes adatokat gyűjtenek az interneten és ezeket online marketing célra értékesítik (https://datafox.com/keywords/data%20aggregation).

állampolgári elektronikus szolgáltatások szoftverrendszereinek lehetséges terveit (Janssen et al., 2003). Ezek a tervek specifikálják és ütemezik azokat a szolgáltatásokat, amelyekre az állampolgároknak leginkább szükségük van.

Az okos városok projektekben különösen nagy figyelmet szentelnek az e-government szolgáltatásokra, hiszen az okos városok egyik alapelve a szolgáltató önkormányzat (CitRec'17, August 2017, Como, Italy[30]).

Az ajánlórendszerek technológiájában és az alkalmazások céljában az utóbbi évtizedben jelentős változások következtek be. De Meo 2005-ös cikkében (De Meo et al., 2005) az elektronikus szolgáltatások működőképességének szimulációját jelölik meg célként, egy jól definiált mesterséges környezetben. Olyan e-government szolgáltatásokat modelleznek, amelyek figyelembe veszik mind az állampolgárok preferenciáit, mind az általuk használt kommunikációs eszközöket. A modellezés célja a rendszer működésének és hatékonyságának vizsgálata. A modellt szimulációs kísérletben vizsgálták meg 30 felhasználóval 90 szolgáltatás esetén. Ún. multiágens technológiát alkalmaztak, ahol az egyes ágensek:

- Felhasználói eszköz ágens
- Felhasználói profil ágens
- Szolgáltatás ajánló ágens
- Kormányzati ágens

A felhasználói eszköz ágens az ügyfelek elérési eszközeinek (számítógépeinek, okostelefonjainak stb.) képességeit (processzálási kapacitás, hálózati sávszélesség) modellezte, a felhasználói profil ágens, amely leginkább megfelel a mai ajánlórendszereknek, a felhasználók demográfiai adatait, a szolgáltatások korábbi igénybevételét és múltbeli elégedettségi mutatóit adta meg múltbeli adatbázisok alapján, a szolgáltatás ajánló ágens a felhasználói ágens által szolgáltatott felhasználói profil adatokból a rendelkezésre álló szolgáltatások[31] közül kiválogatta azokat, amelyek a modell szerint a felhasználó számára a legfontosabbak vagy legérdekesebbek. A szolgáltatások köre kiterjedt az egészségügyre, társadalombiztosításra, oktatásra, közösségi közlekedésre stb. A kormányzati ágens lényegében a felhasználói interfész szerepét töltötte be a szolgáltatások esetén.

A szimulációs kísérletben az egyes tényezők szerepét vizsgálták a rendszer előrejelző képességének pontosságát a felhasználó perszonalizációs szintje és a rendelkezésükre álló eszközök függvényében. (A gépi tanulórendszerek pontosságának mérésével kapcsolatban ld. a 11. fejezetet.) A szimulációs kísérletek rámutattak a felhasználói profilok meghatározásának fontosságára és arra, hogy melyek a fontos és kevésbé fontos szolgáltatások egy adott felhasználói csoportban.

A jelenlegi webes szolgáltatások fejlesztői már a rendszerek tervezésénél figyelembe veszik a felhasználók által használt eszközöket, és reszponzív[32] rendszereket fejlesztenek. A jelenleg használt okostelefonok, asztali PC-k, tabletek és laptopok és az azokat kiszolgáló

30 https://dblp.org/db/conf/recsys/citrec2017
31 A http://www.italia.gov.it oldalról.
32 A képernyő (böngészőablak) szélességére reagáló, rugalmasan változó grafikai megjelenésű weblap.

hálózatok az elektronikus közszolgáltatásokhoz elegendő sávszélességgel rendelkeznek, így ma már a felhasználói eszközök nem korlátozzák a szolgáltatások igénybevételét.

A jelenlegi ajánlórendszerek a perszonalizált felhasználói profilok gyors és pontos meghatározására és az ezekre épülő hatékony ajánlatkészítésre fókuszálnak.

Az alábbiakban néhány alkalmazási területet mutatunk be (Cortés-Cediel et al., 2017 cikke alapján):

1. Személyre szabott értesítések és ajánlások.[33] Mind a kollaboratív, mind a tartalomalapú szűrési stratégia[34] alkalmazásával történik, az állampolgár egyedi profiljának, mind a hozzá hasonló más állampolgárok profiljának, vizsgálatával. Az e-government szolgáltatásoknál a kereskedelmi alkalmazásokhoz képest sokkal nagyobb szerepe van a kontextusalapú értesítéseknek, mint például egy okmány lejárati idejéről, egy adott lakókörzetben végrehajtott útlezárásról küldött e-mailnek.
2. Az állampolgárok részéről a kormányzat számára fontos problémaleírások, vélemények a kormányzat által működtetett platformokon és a közösségi hálókban, blogokban. Az ajánlórendszerek a kormányzati tervezés hasznos eszközei lehetnek, sok potenciális problémát tesznek elkerülhetővé.

Ez esetben a természetes nyelvi feldolgozást és a véleménybányászati technikákat alkalmazzák (ld. 9. fejezet) arra, hogy a problémák, vélemények alapján a szükséges döntéseket előkészítsék.

3. Az állampolgárok támogatása a releváns javaslatok, vitafórumok, közösségek megtalálásában közösségi aktivitásuk fokozása érdekében. A támogatás az explicit módon kifejezett vagy kommentekből, közösségi kapcsolatokból meghatározott adatok alapján történik. Ez esetben a közösségi alapú ajánlórendszerek alkalmazása a jellemző, amely a felhasználók közösségi hálózati struktúráját vizsgálja.
4. G2B ajánlórendszerek a vállalkozások számára szükséges szolgáltatások elérhetőségéről adnak eligazítást. Ajánlásokat tesznek a szükséges információ, a kitöltendő elektronikus űrlapok elérhetőségére, az engedélyek beszerzésének módjára stb. Az ajánlórendszer kialakítja és tárolja a vállalkozás profilját és ennek alapján folyamatosan tájékoztatást ad egyrészt az e-government szolgáltatásokról, az érvényes szabályozásról, másrészt egyéb, igénybe vehető erőforrásokról, pályázatokról és a kormányzati beszerzési szándékokról.
5. Üzleti partnerek keresése a kormányzati elektronikus szolgáltatásokban. Az üzleti partnereket szolgáltatásaik alapján kategorizálják és ez az alapja a szemantikus ajánlórendszereknek, amelyek a szolgáltatások tartalma alapján készítenek ajánlásokat. Ha rendelkezésre állnak a vállalkozásokról korábbi értékelési adatok, akkor a szemantikus ajánlásokat a kollaboratív szűréssel lehet kombinálni. Az ajánlások céljai itt elsősorban azok a kormányzati szervezetek, amelyek piaci szolgáltatásokat kívánnak vásárolni.
6. A vállalatok számára – profiljuknak megfelelő – online konzultáció jogi és eljárási ügyekben. A lényeg itt a vállalat egyedi profilja, azaz érvényesül az ajánlórendszerek legfőbb pozitívuma, vagyis a személyre szabottság.

33 Hazai példa a jogosítvány érvényességének lejáratáról szóló értesítés.
34 A kollaboratív szűrés, tartalomalapú szűrés, közösségi háló fogalmakat a fejezet további részében határozzuk meg.

7. A G2G relációban az ajánlórendszerek az egyes kormányzati szervezetek közötti adatcserét tudják segíteni azzal, hogy javaslatot tesznek arra, hogy mely szervezetek számára lehetnek fontosak egyes dokumentumok. Ehhez szükség van a szervezetek közötti adatintegrációra és interoperabilitásra, vagyis az információs rendszerek együttműködési képességének biztosítására.
8. A kormányzati szervezeteken belüli elektronikus rendszerű humánerőforrás-menedzsment javítása. Az ajánlórendszer a közszolgálatban dolgozók szakmai profilja, munkaképessége, teljesítménye alapján javaslatot tehet arra, hogy egyes pozíciók betöltésére kik lehetnek a legalkalmasabbak, akár a szervezetek határain is átlépve.
9. A kormányzati szervezetek munkatársainak számára perszonalizált ajánlások készítése szakmai eseményekről, megpályázható munkakörökről, képzésekről. Természetesen mindehhez szükség van a munkatársak szakmai profiljára, ami egyébként az emberierőforrás-rekordokban rendelkezésre áll, az ajánlórendszerek csupán egy újszerű felhasználási módot jelentenek.

Az alábbiakban áttekintjük az ajánlórendszerek elméletének és gyakorlati alkalmazásának alapjait.

A kollaboratív szűrésen alapuló ajánlórendszerek

A kollaboratív szűrésen (collaborative filtering, CF) alapuló ajánlórendszerek a legegyszerűbb, de ugyanakkor a leggyakrabban alkalmazott, sikeres ajánlóalgoritmusok. A CF rendszerek adott ügyfélcsoport ismert preferenciáinak alapján becsülik más ügyfelek ismeretlen preferenciáit. A szűrés a preferenciák alapján történik. Három különböző módszer tartozik ebbe a csoportba:

- a memóriaalapú,
- a modellalapú és
- a hibrid

eljárás. A szűrés történhet ügyfelek vagy termékek/szolgáltatások szerint.

Az ügyfélalapú kollaboratív szűrés

A CF elnevezést Goldberg (1992) vezette be.

A CF alapfeltételezése az, hogy ha az X és az Y ügyfél hasonlóan értékelnek egy termékeket, vagy hasonlóan viselkednek egy termékkel vagy szolgáltatással kapcsolatban (pl. vesznek igénybe szolgáltatásokat), akkor más termékeket és szolgáltatásokat is hasonlóan értékelnek vagy vesznek igénybe (Goldberg, 2001).

A CF módszer egy ügyfélpreferenciákból, azaz ügyfélprofilból álló adatbázisból indul ki.

Legyen az ügyfelek száma m, az ügyfelek halmaza: , a termékek száma n, a termékek halmaza: . Minden ügyfél értékeli a termékek egy) listáját, akár explicit módon, akár azzal, hogy igénybe vesz egy szolgáltatást, akár azzal, hogy érdeklődik/nem érdeklődik a szolgáltatás iránt. Az ügyfelek és termékek közötti kapcsolatot egy ügyfél–termék mátrixszal ábrázoljuk (7. ábra).

ügyfél/termék	i_1	i_2	i_3	i_4	i_5
u_1	3	NA	NA	NA	3
u_2	2	5	NA	3	NA
u_3	NA	2	2	NA	NA
u_4	5	3	3	4	NA

7. ábra. Ügyfél–termék mátrix, a 0–5 skálán történő explicit értékeléssel. NA – hiányzó adat, vagyis az illető nem értékelte az adott terméket
(Forrás: Saját szerkesztés.)

ügyfél\termék	i_1	i_2	i_3	i_4	i_5
u_1	1	0	0	0	1
u_2	1	1	0	1	0
u_3	0	1	1	0	0
u_4	1	1	1	1	0

8. ábra. Ügyfél–termék mátrix. Az 1 érték azt jelenti, hogy az ügyfél megvásárolta, vagy megnézte a terméket, a 0 azt, hogy nem. A mátrix ábrázolhatja a Like relációt is
(Forrás: Saját szerkesztés.)

Tipikus CF feladat, amikor az ügyfél számára az u_1, u_2, u_3 ügyfelek implicit vagy explicit i_1, i_2, i_3, i_4, i_5 szolgáltatás értékeléséből készítünk prognózist arra vonatkozóan, hogy az u_4 ügyfélnek tetszik-e majd az i_5 szolgáltatás.

A mátrix számos eleme nem tartalmaz adatot, ui. valós körülmények között, amikor az ügyfelek száma több százezer vagy több millió és a szolgáltatások száma is százas nagyságrendű, egy-egy ügyfél csak kevés szolgáltatással kerül kapcsolatba. Az akár több millió sorból és több száz oszlopból álló mátrix nullától különböző elemeinek száma igen alacsony. Nyilvánvaló, hogy ez speciális számítógépes big data megoldásokat tesz szükségessé.

A CF feladatok megoldásának számos gyakorlati problémája van[35]:

- az ügyfél-szolgáltatás mátrix ritka, amely egy valós feladatnál eredeti formájában nem fér be a RAM-ba,
- a skálázhatóság, azaz az ügyfelek és a szolgáltatások számának növekedésével a számítások teljesítményének fenntartása,
- valós idejű ajánlások készítése az ügyfél bejelentkezésekor,
- a hasonló, de különböző nevű termékek vagy a hasonló nevű, de különböző termékek kezelése,
- személyes adatokkal kapcsolatos szabályozásoknak való megfelelés,
- a „cold start” probléma – mit ajánljunk annak, aki most jelentkezett be először,
- az ún. schilling támadás, ami az értékelések szándékos eltérítését jelenti az ajánlórendszerek manipulálása céljából,
- a „black sheep” problémának nevezik azt, hogy egyes emberek konzisztensen szembe mennek az árral, azaz véleményük jellemzően ellentétes a többség véleményével,

35 https://www.hindawi.com/journals/aai/2009/421425/tab1/

- a „grey sheep” probléma, azok az emberek, akiknek a véleménye többnyire eltér a többség véleményétől, de az eltérés nem konzisztens. Mind a black sheep, mind a grey sheep probléma nem ismeretlen az elektronikus közszolgáltatások értékelésénél.

A CF rendszerek korai megvalósításában az alábbi egyszerű algoritmus szerint számították az ajánlásokat (9. ábra).

9. ábra. Egyszerű CF-algoritmus
(Forrás: Saját szerkesztés.)

Az ajánlás mindig az aktív ügyfél számára készül. Az aktív ügyfélhez hasonló ügyfelek kiválasztása kreatív feladat, az elemzőnek meg kell határoznia azt a hasonlósági kritériumot, amelynek alapján kiválasztja azokat az ügyfeleket, akik véleményét figyelembe veszi az ajánlásnál. Minthogy az ügyfelek száma igen nagy, gyakorlati okokból csak egy szűk, az aktív ügyfélhez valamely hasonlósági kritérium alapján kiválasztott, TopN-csoportot határoznak meg. A legegyszerűbb módszer a TopN értékeléseinek átlagolása, de ha figyelembe vesszük, hogy a TopN tagjai is eltérő módon hasonlítanak az aktív ügyfélre, célszerű az egyszerű átlag helyett valamilyen súlyozott átlagot használni. Minthogy a módszer a teljes ügyfél és termék adatbázist használja, memóriaalapú CF-nek is nevezik. Itt legrosszabb esetben (m*n) nagyságrendű számításra van szükség, hiszen minden szolgáltatás minden ügyféllel való kombinációját meg kell vizsgálni. Tekintettel arra, hogy a termék–ügyfél mátrix igen ritka, a jó algoritmusok ezt kihasználva elérhetik az (m+n) számítási művelethez közeli nagyságrendet.

Gyakran előfordul, hogy vannak olyan vásárlók, akik a teljes terméklista jelentős része iránt érdeklődnek, így náluk a számítási igény n többszöröse, tehát a teljes számítási igény nem csökkenthető lényegesen az (m+n) nagyságrend alá. Milliós m és n értékeknél ez komoly számítási költséget vagy teljesítményproblémát jelent. Általánosan használt megoldás a vizsgált adatok mennyiségének csökkentése. Erre több módszert is használnak egyenként vagy kombinálva:

- véletlen mintavétel
- a kevés cikket fogyasztók kizárása
- a legnépszerűbb vagy legnépszerűtlenebb termékek kizárása
- termék/téma kategóriánkénti elemzés
- klaszteranalízis, főkomponens analízis, mint dimenziócsökkentési eljárások

(ld. Goldberg et al., 2001).

Hogyan választják ki az aktív ügyfélhez legközelebbi TopN ügyfelet, akik explicit vagy implicit ajánlása alapján ajánlatot lehet tenni az aktív ügyfél számára?

Először is meg kell határozni az ügyfelek távolságának mérési módszerét, tekintetbe véve azt, hogy az ügyfél többféle módon értékelhet:

1. folytonos skálán történő értékelés (pl. -5 és +5 között bármely, akár nem egész érték is adható)
2. intervallum értékelés (1–5 pont közötti egész számok)
3. kategória skálán történő értékelés (pl. nagyon nem tetszett, nem tetszett, tetszett, nagyon tetszett)
4. bináris értékelés (tetszett–nem tetszett)
5. unáris értékelés (a fogyasztó lájkol vagy megvásárol valamit).

Az unáris skála abban különbözik a bináris skálától, hogy míg a bináris skálán mindkét érték véleményt fejez ki, addig az unáris skálán az, hogy valaki nem lájkol valamit, vagy nem vásárol meg egy terméket, az még nem jelenti azt, hogy a termék neki nem tetszik. Ha két ügyfél ugyanazokat a termékeket lájkolja, az hasonlóságot jelez, de ha ugyanazokat nem lájkolja, az nem jelent hasonlóságot.

Az ügyfelek közötti távolság mérésére elvileg többféle mérték is szóba jöhetne, mint az euklideszi távolság, az ügyfelek mint vektorok által bezárt szög koszinusza és a Pearson-féle korrelációs koefficiens. Az euklideszi távolság a nem folytonos skálákon azonban nem értelmezhető, így nem használják. A leggyakrabban használt mérték a korreláció, a továbbiakban ezt ismertetjük.

A mind az *u,* mind a *v* ügyfél által explicit vagy implicit módon értékelt szolgáltatáslista az

$$I(u,v) = I(u) \cap I(v)$$

halmaz, a korrelációt csak ezen a halmazon számoljuk.
Legyen az *u* ügyfél átlagos értékelése, ahol r_{uk} az *u* ügyfél értékelése a *k* szolgáltatásra.

$$\mu_u = \frac{\sum_{k \in I(u)} r_{uk}}{|I(u)|}, \quad \forall u \in \{1,2 \dots m)\}$$

Az átlag értékét – ha pontosak szeretnénk lenni – akkor csak azokra a termékekre számolnánk, amelyek közösek a között a két ügyfél között, akiknek a távolságát mérjük, de ez számítási szempontból igen költséges és időigényes lenne (emlékezzünk, hogy az ajánlást valós időben kell megtenni), ezért egy ügyfélhez egy adott μ_u értéket rendelünk, amit offline is lehet számolni. Természetesen az átlag kiszámítása minden konkrét ügyfélpár esetén pontosabb eredményt ad, ezért az ilyen eljárásokra is van példa.

Az *u* és a *v* közötti hasonlóság mértéke:

$$SIM(u,v) = Pearson(u,v) = \frac{\sum_{k \in I(u,v)}(r_{uk} - \mu_u)(r_{vk} - \mu_v)}{\sqrt{\sum_{k \in I(u,v)}(r_{uk} - \mu_u)^2}\sqrt{\sum_{k \in I(u,v)}(r_{vk} - \mu_v)^2}}$$

A távolságok számításánál figyelembe kell venni, hogy minden ember másképpen értékel, van, akinél az átlag feletti értékelés a jellemző, és van, akinél az ez alatti. A korreláció számítás előtt ezért nullára központosítjuk az értékeléseket.

Legyen a központosított értékelés az *u* ügyfélre és a *j* termékre:

$$s_{uj} = r_{uj} - \mu_j \quad \forall u \in \{1, \dots m\}.$$

Legyen $P(u_N)$ azoknak a TopN ügyfeleknek a halmaza, akik értékelték a *j* terméket.

A becsült értékeléshez a végén levont átlagot majd hozzá kell adni, hogy korrigált, perszonalizált értéket kapjunk.

Ekkor az *u* ügyfélkorrigált becsült értékelése a *j* termékre:

$$\hat{r}_{uj} = \mu_u + \frac{\sum_{v \in P(u_n)} SIM(u,v) s_{vj}}{\sum_{v \in P(u_n)} |SIM(u,v)|}$$

A központosítás helyett egyes rendszerekben az értékeléseket sztenderdizálják, azaz nem csak az átlaggal csökkentik a tényleges értékelési „osztályzatot", hanem elosztják a minta szórásával is. A Pearson-féle korreláció számos variánsát is használják a távolság mérésére, egyes rendszerekben a *SIM(u,v)* helyett $[SIM(u,v)]^{\alpha}$ értéket alkalmazzák a hasonlóság értékének hangsúlyozására, vagy éppen csökkentésére az α paraméter megfelelő megválasztásával.

A módszer más módosításait is használják, ha a konkrét szaktarületi igények ezt szükségessé teszik. Erről a szakirodalomban találunk további információt (Regi et al., 2013).

Ha az ügyfelekről csak annyit tudunk, hogy használtak-e vagy megnéztek-e egy szolgáltatást, a Pearson-féle korrelációs koefficiens nem használható távolságuk mérésére. Ugyanis, ha két ügyfél nem használt egy szolgáltatást, akkor nem tudjuk, hogy esetleg nem tetszett nekik vagy nem is ismerték. Ezért, ha mindkettőnél a 0 szerepel egy szolgáltatásnál, ez nem jelent hasonlóságot. Ugyanígy, ha az egyik ügyfél használta a szolgáltatást, a másik nem, ez sem jelenti feltétlenül a különbözőséget, mert aki nem vásárolta meg, az esetleg nem is ismerte a terméket. Két ügyfél távolságáról csak azoknál a szolgáltatásoknál állíthatunk valamit, amelyeket használtak, ezeknél a távolságuk nulla. Így ez esetben a Pearson-féle korreláció nem lehet a távolság mértéke.

Erre az esetre szolgál a Jaccard-távolság, amely az azonos választások arányát tekinti két ügyfél távolságának:

$$Sim_{Jaccard} = \frac{|I(u) \cap I(v)|}{|I(u) \cup I(v)|},$$

ahol *I(u)* és *I(v)* azoknak a szolgáltatásoknak a halmaza, amelyeknél az *u,* ill. a *v* ügyfélnél az 1-es érték áll.

Amennyiben az értékeléseknél a kategóriaváltozóknak kettőnél több értékük lehet, akkor bináris változókká transzformáljuk azokat. Ha például az X kategóriaváltozó az *a, b* és *c* értéket veheti fel, akkor három bináris vektorváltozót vezetünk be, X1 = {i_1,0,0}, X2 = {0,i_2,0}, X3 = {0,0,i_3} értékekkel, ahol i_1 = 1, ha X = a, egyébként 0, i_2 = 1, ha X = b, egyébként 0, i_3 = 1, ha X = C, egyébként 0. Ezzel a transzformációval a kategóriaváltozók esetén is a Jaccard-távolságot tudjuk használni.

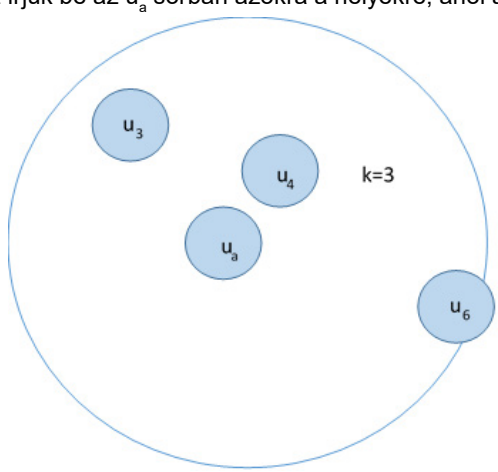
Példa ügyfélalapú CF-re:

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
u_1	NA	3	3	2	1	1	NA	NA
u_2	4	NA	NA	NA	4	5	NA	NA
u_3	3	NA	NA	3	2	2	NA	3
u_4	4	NA	NA	2	1	1	2	1
u_5	1	1	NA	NA	NA	NA	NA	1
u_6	NA	1	NA	NA	1	1	NA	1
u_a	NA	NA	4	3	NA	1	NA	3
r_a	3,5	1	4	3	1,3	1	2	3

10. ábra. Ordinális változó leképezése bináris változókká. NA – hiányzó érték (Forrás: Saját szerkesztés.)

A 10. ábra az u_1–u_6 ügyfelek és az u_a aktuális ügyfél értékeléseit tartalmazza az 1–5 skálán. Az NA azt jelöli, hogy nincs adat, vagyis az adott ügyfél az adott szolgáltatást nem értékelte.

Az u_a ügyfélhez a fenti értékelések alapján az u_1, u_3 és u_4 állnak közel (11. ábra), így ezek értékelésének átlagát írjuk be az u_a sorban azokra a helyekre, ahol az értékelés hiányzik.



11. ábra. Közel állók a termékalapú kollaboratív szűrés esetén (Forrás: Saját szerkesztés.)

Az aktív *u* ügyfél számára a rendszer úgy készít ajánlást az *i* termékről, hogy először meghatározza azoknak a termékeknek az *S* halmazát, amelyek hasonlóak az *i* termékhez, majd meghatározza az *u* ügyfél által készített értékeléseket az *S* halmazban. Ezeknek az értékeléseknek a súlyozott átlaga lesz az *i* termék becsült értékelése az *u* számára.

Az ügyfélalapú és a termék/szolgáltatásalapú kollaboratív szűrés közötti különbség az, hogy az ügyfélalapú szűrésnél a hasonló ügyfelek értékelése alapján készül az ajánlás, míg a szolgáltatásalapú szűrésnél az aktív ügyfél saját, hasonló szolgáltatásokra leadott értékeléseit alkalmazzuk (Agarwal et al., 2016).

Technikailag az első esetben az ügyfél–termék értékelési mátrix sorai közötti hasonlóságot, míg a termékalapú szűrésnél az oszlopai közötti hasonlóságot vizsgáljuk.

A két módszer, bár hasonló egymáshoz, számítási szempontból mégis különböző. Az ügyfélalapú megközelítés közvetlenül kapcsolható a webes használathoz. A konkrét számításokat az ügyfélalapú CF szerint, de az ügyfél–termék értékelési mátrix transzponáltján, azaz a termék–ügyfél értékelési mátrixon végezzük, ezért itt nem részletezzük.

A hibrid ajánlórendszerek

A hibrid ajánlórendszerek általában kombinálják a tudásalapú rendszereket a kollaboratív szűrő rendszerekkel. A hasznosságon alapuló és a tudásalapú rendszerek nem használják a múltbeli vásárlói adatokat, így elkerülik az azoknál jelentkező cold start problémát, azt, hogy az új ügyfélről vagy az új termékről még nincs információ. Elkerülhető az ún. „felzárkózási időszak", ami alatt egy új ügyfélnél vagy szolgáltatásnál a kevés adat miatt nem jó minőségű ajánlatok készülnek és a hiányzó adatok is sokkal kevésbé jelentenek problémát ebben az esetben.

A hasznosságon alapuló rendszereknél szükség van arra, hogy minden ügyfél rendelkezzen egy hasznossági függvénnyel, amely megmutatja, hogy egy adott terméket érdemes-e ajánlani az ügyfélnek. A függvény előállítása nem automatikus, szükséges az ügyféllel való interakció, ugyanakkor lehetőség van az termékhez csak közvetve kapcsolódó paraméterek figyelembevételére, mint például a szállítási határidők, a garanciafeltételek, minőségi mutatók stb. Sokszor ezek a feltételek döntőek lehetnek egy vásárláskor, pl. egy ajándék vásárlásakor egy nevezetes dátumra. Nem állandó ügyfelek vagy változó preferenciák esetén a módszer nyilvánvalóan kevésbé jól működik. A hasznosság alapján minden ügyfél számára előállítható egy TopN preferencia lista.

A tudásalapú rendszerekhez szükség van jól használható tudásbázisra. Az ajánlórendszerek általában az alábbi tudásbázisokat használják:

- *Szolgáltatásjegyzék,* amely tartalmazza a szolgáltatások leírását. A jegyzéket egységes ontológiára célszerű építeni, hogy a fogalmak tiszták legyenek.
- *Funkcionális tudás*, amely leképezi az ügyfél szükségleteit a szolgáltatásokra. Ehhez szükség van egy egyértelmű leképezésre az ügyféligények és a szolgáltatástulajdonságok között.
- *Az ügyfél ismerete* – a rendszer jobb ajánlást tud tenni, ha ismeri az ügyfelet, ismeri a demográfiai adatait és más speciális körülményeket.

A tudásalapú rendszerek számos előnnyel járnak, nem követelnek annyi interakciót az ügyféltől, mint a hasznosságalapú rendszerek, és nincs szükség az átmeneti adatgyűjtési időszakra sem.

A hibrid rendszerek, kombinálva a CF megoldásokat és a hasznosság-, ill. tudásalapú rendszerekkel, az egyféle algoritmussal működő rendszerek számos hiányosságát tudják pótolni.

Továbbra is fennmarad azonban néhány megoldatlan probléma:

- viszonylagos rugalmatlanság – mind a CF, mind a hibrid rendszerekben a TopN lista összeállításához szükséges számítások legnagyobb részét offline kell elvégezni, az aktuális ügyfél aktuális kérdései, és az, hogy milyen szolgáltatásokat vesz igénybe, nincsenek hatással a kimenetelre,
- mind a CF, mind a hibrid módszerek a RAM-ban működnek, így skálázhatóságuk korlátozott.

A modellalapú rendszerek

A modellalapú rendszerek célja a CF-alapú és a hibrid rendszerek korlátainak, a rugalmatlanságnak és a memóriafüggőségnek a kiküszöbölése. A modellalapú rendszerek a múlt adataira építenek, de statisztikai módszereket, gépi tanulási algoritmusokat és látens szemantikus faktorok felderítését alkalmazzák az ajánlatok előállításához.

A modellalapú rendszerek előnyei:

- Pontosabbak, mint a CF vagy a hibrid módszerek.
- A figyelembe vett paraméterek súlyát gépi tanulással határozzák meg, nem manuálisan.
- Rejtett mintákra is fényt deríthetnek.

A statisztikai modellek

A statisztikai modellek alkalmazása esetén kiszámítják a termékek és ügyfelek tulajdonságainak statisztikai jellemzőit, majd megkeresik azt a TopN terméklistát, amelyet az aktuális ügyfél a legnagyobb eséllyel választ. Az egyszerűsége és jó teljesítménye miatt népszerű modell a Naive Bayes (ld. 4. fejezet).

Gyakran használják a dimenzió csökkentésére a közelítő mátrix faktorizációs módszert és a gépi tanulás klasszifikációs algoritmusait:

- regressziószámítást,
- a KNN-módszert és
- a döntési fákat.

Ajánlórendszerek a gyakorlatban

Az elektronikus kereskedelemben az ajánlórendszerek minősége fontos, meghatározó tényezője a versenyképességnek. A legsikeresebb ajánlórendszerek pontos működése ezért mindenütt féltve őrzött üzleti titok. Ismert, hogy a vállalkozások nem egy, hanem több algoritmust használnak saját ügyfél- és termékkörüknek, működési módjuknak és üzleti stratégiájuknak megfelelően. A megfelelő algoritmus „mix" összeállítása általában komoly szakértelmet, tapasztalatot és kísérletezést igényel.

Az egyik legsikeresebb ajánlórendszert alkalmazó cég, a Netflix – ahol a filmletöltések kétharmada ajánlások lapján történik – 1 millió $ díjazással járó versenyt hirdetett 2006-ban az ajánlások 10%-os hatékonyságának javítására. A versenyt a BellKor's Pragmatic Chaos team nyerte meg 2009-ben azzal, hogy a Netflix saját algoritmusának hatékonyságához képest 10,06%-os javulást értek el az értékelések becslésének pontosságában. Megjegyezzük, hogy a versenyt egy ideig a Budapesti Műszaki Egyetem Gravity teamje vezette, de nem érték el a 10%-ot (ld. Wikipedia). A versenyt az eredeti szándékok alapján nem folytatták, tekintettel az adatvédelmi aggályokra. A Netflix versenyéről a cég honlapján tájékozódhatunk.[36]

A teljesítményt és a teljesítményjavulást az egyszerűség kedvéért a négyzetes közép hiba RMSE-értékkel (RMSE: root mean square error) mérték, ami ugyan könnyen számítható, de tekintettel arra, hogy az ajánlások a valóságban nem a teljes listát, hanem csak TopN listát tartalmazzák, erősen torzíthat. Ezért a valóságban többféle mértéket is használnak, ugyanúgy, mint a gépi tanuló algoritmusoknál általában.

Az ajánlórendszerek legtöbbször az algoritmusok által becsült preferenciákon kívül egyéb, üzleti szempontból fontos termékekkel „manuálisan" is bővítik a TopN listákat, például:

- magas profittartamú termékek, ha „organikus" úton nem is kerülnek be a listába,
- új termékek, amelyeket be kívánnak vezetni a piacra,
- szezonális termékek.

Az ajánlórendszerek speciális programozási eszközei általában nem egy konkrét rendszer létrehozását segítik, hanem a különféle algoritmusok, koncepciók kipróbálását, az ezekkel való kísérletezést támogatják.

Az ajánlórendszerek fejlesztését támogató eszközök

Számos, az ajánlórendszerek fejlesztését támogató eszköz áll rendelkezésre, pl. az R nyelvben, a Pythonban és sok egyéb platformon. Ezek legtöbbször egy kísérletezésre alkalmas keretet biztosítanak a fejlesztőnek, ahol a kipróbálhatja saját algoritmusait. Minden, eddig ismertetett és sok egyéb eljárás is tesztelhető.

36 https://www.netflixprize.com/

A megalkotott algoritmusok hatékonyságának mérésére leggyakrabban az alábbi mértékeket használjuk:

- MSE (mean squared error),
- RMSE (root mean squared error),
- MAE (mean absolute error).
- A TopN lista esetén: valós pozitív arány/hamis pozitív arány (TruePositiveRate/FalsePositiveRate, TPR/FPR),
- vevő működési karakterisztika (Receiver Operating Characteristic ROC),
- precision
- recall

A gépi tanulási algoritmusok teljesítménymérésének kérdéseivel a 11. fejezetben foglalkozunk.

A rendszerekhez tesztelési protokollokat is készítettek, hogy sztenderd összehasonlításokat lehessen végezni a teljesítményük között.

Az értékelési eljárások metodikájához Breese et al. (1998) négy kísérleti protokollt vezettek be, a Given 2, a Given 5, a Given 10 és az All-but-1 módszert. Ezt később Given-n-re és All-but-x-re általánosították.

A Given x protokollban az ügyfél értékelésből x véletlenszerűen kiválasztott terméket használnak a tanuló algoritmusban, a többit visszatartják a modell teszteléséhez. Az All-but-x protokoll esetén a tanulásnál visszatartanak x terméket tesztelési célból, a többit felhasználják a tanulásnál.

Az ajánlómodellek értékelése és az ajánlóalgoritmus elemzése

Szemléltetésképpen létrehozunk egy vizsgálati sémát, amely a Jester5k adatait egy tanulóhalmazra (90%) és teszthalmazra (10%) osztja. A séma célja a módszer hatékonyságának elemzése.

A teszthalmazból 15 cikket a tesztelésre, a maradékot a hiba számítására használunk. Ezután kiszámoljuk a tesztadatok ismert részére (ügyfelenként 15 cikk) az értékelést az UBCF és az IBCF algoritmussal. Csak az 5 és az 5 feletti értéket tekintjük jó értékelésnek, ami ez alatt van, azzal nem számolunk. Végül kiszámítjuk a becslés és a tesztadatok ismeretlen része közötti hibát a két módszer esetén. Az eredményt a 2. táblázat tartalmazza.

	RMSE	MSE	MAE
UBCF	4.478531	20.05724	3.514909
IBCF	5.103639	26.04713	3.988540

2. táblázat: Az UBCF- és IBCF-módszerek hibái
(Forrás: Saját szerkesztés.)

A 2. táblázatból látjuk, hogy az UBCF kisebb becslési hibával jár ezen az adathalmazon.

A sémában lehet meghatározni a kísérleti protokollt, ami vagy Given-n vagy All-but-x típusú.

A 3. táblázatban a Given-n protokollt alkalmazva n = 1,3,5,10,15,20 termékre lefolytatott, a POPULAR típusú modellt kiértékelő séma konfúziós mátrixát látjuk. A táblázat oszlopainak értelmezését ld. a 11. fejezetben.

	TP	FP	FN	TN	precision	recall	TPR	FPR
1	0.5264	0.4736	20.166	73.833	0.5264	0.03406	0.03406	0.00602
3	1.4912	1.5088	19.201	72.798	0.49706	0.09024	0.09024	0.01918
5	2.4832	2.5168	18.209	71.790	0.49664	0.15184	0.15184	0.03200
10	4.7392	5.2608	15.953	69.046	0.47392	0.28264	0.28264	0.06695
15	6.8240	8.1760	13.868	66.131	0.45493	0.39625	0.39625	0.10497
20	8.3992	11.600	12.293	62.706	0.41996	0.47367	0.47367	0.14965

3. táblázat. Az értékelési séma alkalmazásával kapott eredmények (Forrás: Saját szerkesztés.)

Az ajánlórendszerek prototípusai

Egy rendszer létrehozásához nem csak maga az ajánló algoritmus szükséges, hanem egyéb támogató rendszerek is. Lássunk egy példát, egy intelligens üzleti partnerkereső ajánlórendszert (Lu et al., 2009).

Az ajánlórendszer prototípusát Intelligent Business Partner Locator (IBPL)-nak nevezték el. A rendszer releváns potenciális üzleti partnerek keresésére lehet használni. A rendszernek három komponense van:

1. Adatgyűjtő alrendszer, amely az üzleti partnerek preferenciáira és profiljára vonatkozó adatokat gyűjti össze.
2. Adatbázis rendszer, amely három adatbázist tartalmaz: termék relevancia adatbázis, üzleti profil és felhasználói értékelés adatbázis
3. Ajánlómotor, amely egyedi ajánlati listát készít egy vállalkozás számára a lehetséges üzleti partnerekről a vállalkozás profilja és preferenciái alapján. Az ajánlómotor tartalmaz egy termék/szolgáltatásalapú CF hasonlósági elemzőt a partnerek hasonlósági mátrixának számítására az aktív vállalat preferenciái alapján, egy tartalmi elemzőt a hasonló partnerek kiválasztására és egy ajánlatgeneráló részt. Az ajánlatgeneráló program becsüli meg a hasonlóság alapján kiválasztott vállalkozások értékelési számait a két hasonlósági mérték (CF és tartalmi mértékek) alapján és készíti el a TopN listát az aktív vállalat számára.

Közbeszerzési rendszer

Az elektronikus közbeszerzés (eGov Procurement, eGP) a közszféra fontos eszköze az állami, és helyi kormányzatok beszerzési költségeinek csökkentésében és az áttekinthetőség növelésében. Az alábbiakban egy ajánlórendszer-alapú eGP konceptuális tervét ismertetjük (Zhang et al., 2013).

A módszer lényegében a CF algoritmusra épül. A termékek és szolgáltatások hasonlóságának meghatározása az ún. fuzzy hasonlósági mérték alapján kerül kiszámításra, ami tükrözi, hogy a gazdasági döntések során nem áll rendelkezésre minden információ, a döntéseket legtöbbször hiányos információ alapján kell meghozni. A CF első szakaszában a rendelkezésre álló információ hiányossága miatt sokszor csak hozzávetőlegesen határozhatók meg az értékelési kritériumok. Ezek alapján történik a potenciális szállítók kiválasztása. A második szakasz a tényleges szűrés, amikor a termék/szolgáltatásalapú kollaboratív szűrési algoritmussal vagy más módszerrel kiválasztják a TopN potenciális beszállítót. Az aktív felhasználó itt a beszerzést kiíró hivatal.

Az utolsó lépés a TopN lista alapján a minőségi és más kritériumoknak (ár, teljesítmény, szolgáltatásminőség) legjobban megfelelő beszállító kiválasztása.

4. FELÜGYELT GÉPI TANULÁS – KLASSZIFIKÁCIÓS MÓDSZEREK

A felügyelt gépi tanulás

A felügyelt gépi tanulás – angolul supervised learning – a gépi tanulás módszereinek egy fajtája. Felügyelt gépi tanulásnál a rendelkezésre álló adatok alapján olyan modellt hozunk létre, amely képes megfelelő pontossággal előre jelezni, hogy egy új, még nem ismert objektum mely előre meghatározott osztályba fog tartozni vagy milyen tulajdonságokkal fog rendelkezni.

Jelöljük X-szel az input, Y-nal az output változókat, az

$$Y = f(X)$$

függvényt szeretnénk meghatározni egy ismert (X, Y) adathalmaz alapján, amely megfelelő pontossággal képes előállítani az ismeretlen adatok esetén az

$$\tilde{Y} = f(\tilde{X})$$

outputot. Az output itt vagy klasszifikációt, vagy regressziót jelent, azaz vagy az objektum egy meghatározott osztályba tartozását, vagy egy tulajdonságát határozzuk meg.

A jellemzően felügyelt tanulásra használt algoritmusok az alábbiak:

- klasszifikációs eljárások:
- tartó-vektor gépek, angolul Support Vector Machines, SVM,
- Naiv Bayes-módszer
- döntési fák,
- neurális hálók,
- egyes klaszterezési eljárások,
- logisztikus regresszió;

regressziós modellek:

- lineáris regresszió,
- generalizált lineáris regresszió.

A felügyelt gépi tanulás lépései:

1. A feladat meghatározása, például osztályozni szeretnénk-e az objektumokat, vagy egy idősorból trendet szeretnénk számolni.
2. Az algoritmus kiválasztása a feladathoz. Az objektumok osztályozására többféle eljárást is használhatunk, mint például az SVM-et, a döntési fákat vagy a neurális hálókat stb. A választás a feladat jellegétől függ, ehhez ismerni kell az egyes eljárások előnyeit, hátrányait és azt, hogy milyen típusú adathalmazon melyik eljárás a leghatékonyabb.
3. A nyers adatok vizsgálata. Az elemzéshez képet kell kapnunk arról, hogy vannak-e hiányzó adatok, ill. mekkora a hiányzó adatok mennyisége, milyen az adatok struktúrája, létezik-e egyáltalán meghatározható struktúra, vannak-e felesleges adatelemek. Például a természetes nyelvi szövegek elemzésénél általában nem feltételezhetünk létező adatszerkezeteket, lehetnek hiányzó szavak, sok az írásjel és más, nem természetes nyelvi szimbólum.
4. A nyers adatok vizsgálata alapján adattisztítást hajtunk végre, kiszűrjük vagy pótoljuk a hiányzó adatokat, kiszűrjük a felesleges karaktereket és jól kezelhető adatstruktúrákat hozunk létre. Lehet, hogy a nyers adatok strukturálatlanok, de a felügyelt gépi tanulás algoritmusai strukturált adatokkal működnek. Az, hogy milyen struktúrát kell előállítani, az algoritmustól és a feladattól függ.
5. A már strukturált adathalmazt két részre bontjuk, az egyik a tanuláshoz használt (training dataset), a másik a teszteléshez használt (test dataset) adatok. Például, ha hitelbírálat készítéséhez osztályozni szeretnénk a hitelkérelmet benyújtókat (jogosult vs. nem jogosult) és rendelkezésre áll 1000 korábbi hitelfelvevő adata (X vektor változó), valamint az, hogy ezek közül ki fizette vissza és ki nem a hitelét (Y bináris változó), akkor kiválasztjuk 700 személy adatát, és ez lesz a betanulásra használt adathalmaz, a maradék 300 személy adatai pedig a betanított modell verifikálására használt tesztadatok. Itt azzal a feltételezéssel élünk, hogy az adatok statisztikai jellemzői a tanuló és a teszt adathalmazban azonosak. Ezért célszerű a tanuló- és a tesztadatok szétválasztásánál a személyeket véletlenszerűen besorolni, mert lehetséges, hogy a nyers adatok valamilyen rendszer szerint készültek, és ez torzítaná az eredményt.
6. Az adott algoritmus paraméterezése. Minden algoritmushoz be kell állítani néhány paramétert, amelyekkel „hangolni” tudjuk az algoritmust a megoldandó feladat és az adatok szerint. Például a neurális hálóknál meg kell határozni a rejtett rétegek számát, és még több más paramétert (ld. a 6. fejezetben).
7. A tanuló adathalmazon betanítjuk a kiválasztott, paraméterezett modellt.
8. Megvizsgáljuk, hogy a tanuló algoritmuson megfelelő teljesítményt nyújtott-e modell (a teljesítménymérést ld. a 11. fejezetben). Ha igen, akkor tovább megyünk a 9. pontra, ha nem, akkor visszatérünk a 6. pontra.
9. A modellt lefuttatjuk a tesztadatokon. Ha az eredményeket megfelelőnek ítéljük meg, akkor továbbmegyünk a 10. pontra. Ha nem, akkor visszatérünk a 6. pontra.
10. A modellt használatba vesszük.
11. A modell működése során folyamatosan vizsgáljuk a teljesítményét és szükség szerint újabb tanulóadatokon újra tanítjuk (visszatérünk a 4. ponthoz)

A klasszifikációs eljárások

A klasszifikáció, az osztályba sorolás az adatelemzés egyik leggyakrabban felmerülő feladata. A bankok egy hitelkérelem elbírálásánál jó adós vs. nem jó adós osztályba sorolják a kérelmezőt, a felsőoktatásba történő felvételnél a pontszámok alapján felvett és fel nem vett osztályokba sorolják a jelentkezőket, a ruházati cikkeket méret szerint teszik a polcokra az üzletben, egy vállalat osztályokba sorolja ügyfeleit az ügyféladatok alapján, a biológia fontos területe a taxonómia, vagyis az élőlények hierarchikus osztályozása, egy spamszűrőnek el kell tudni döntenie, hogy egy beérkező e-mail spam-e vagy nem, a számítógépes behatolást ellenőrző rendszereknek el kell tudni dönteni, hogy érte-a támadás a rendszert, egy orvosnak el kell döntenie, hogy a páciensnek milyen betegsége van és még sorolhatnánk a példákat.

Az osztályozás a besorolandó objektumok valamely tulajdonságai alapján történik. Ez sokszor egyszerű feladat, egy iskolai tanulónál az életkor alapján az esetek többségében el tudjuk dönteni, hogy hányadik osztályba jár, de például a jó adós–nem jó adós osztályozás ennél sokkal összetettebb feladat, hiszen az, hogy valaki majd a jövőben időben vissza fogja-e fizetni a hitelét, számos jelenlegi és jövőbeli paramétertől és eseménytől függ.

A klasszifikáció igen jelentős szerepet játszik a körülöttünk lévő világ megértésében, a fogalmak tisztázásában, a tudomány fejlődésében és nem utolsósorban az üzleti életben. Az osztályokba sorolás – különösen sok változó esetén – nem egyszerű feladat, a nagy, sokszor áttekinthetetlen adatmennyiség miatt emberi erővel nem is lehetséges.

A számítógépek elterjedése azonban lehetővé tette a humán osztályozási eljárások helyett hatékonyabb gépi eljárások kifejlesztését.

A gépi klasszifikációs eljárásokat az alábbi fő csoportokra oszthatjuk (ismét egy klasszifikáció!):

- felügyelt,
- felügyelet nélküli,
- részben felügyelt klasszifikáció.

A *felügyelt tanulásnál* pontosan tudjuk, hogy mely objektum mely osztályba tartozik, és célunk olyan modell létrehozása, amely nem ismert, vagy a tanuló adathalmazban nem szereplő, objektumokat is hatékonyan be tud sorolni osztályokba.

A *felügyelet nélküli* tanulásnál előzetesen nem tudjuk, hogy milyen osztályok léteznek, az algoritmusra bízzuk ezek létrehozását és az objektumok besorolását.

A *részben felügyelt* tanulás az előző két módszer kombinációja, ismerjük a célosztályokat, de az objektumoknak csak egy részénél tudjuk előzetesen eldönteni, hogy mely osztályba tartoznak.

A big data világában az adatok mérete, szerkezete és a valós idejű feldolgozás igénye új, nem hagyományos klasszifikációs módszerek létrehozását is szükségessé tette, és ez a fejlesztési irány folytatódik. Ilyen új módszereket látunk az adatfolyamokról szóló részben, ahol a felügyelet nélküli osztályozás – klaszterezés – adatfolyamokra kidolgozott változatát láthatjuk.

A fejezetben a továbbiakban a felügyelt gépi tanulás algoritmusaival foglalkozunk, a felügyelet nélküli, klaszterezési eljárások megismerésére az 5. fejezet szolgál. Rövid, általános jellegű leírást találunk Kis-Tóth et al. (2013) és formálisabb leírást Bolla et al. (2010) elektronikus tananyagaiban.

A továbbiakban az alábbi feltételezésekkel élünk:

- Véges objektumhalmazokat vizsgálunk. Az objektumok tulajdonságait változóknak nevezzük.
- A változók lehetnek kvantitatívak vagy kvalitatívak, a kvalitatív változók értéktartománya véges.
- A klasszifikáció valamely változó alapján történik, a klasszifikáló (osztályozó) változó értéke szerint soroljuk be az objektumot egy osztályba. Azt is mondjuk, hogy az objektumokat felcímkézzük, a címke jelöli meg azt az osztályt, ahova az objektum tartozik.

Az osztályba sorolás sok esetben bináris besorolást jelent, azaz az objektum vagy beletartozik egy osztályba, vagy nem. Például, visszafizette-e az adósságát egy hitelfelvevő, vagy nem.

A gépi tanulás során a felcímkézett objektumok adatainak felhasználásával olyan algoritmus – modell – létrehozására törekszünk, amely a címke ismerete nélkül képes hatékonyan osztályozni az új, még nem ismert objektumokat. A hatékonyság fogalmát a 11. fejezetben ismertettük.

Számos módszer létezik az osztályozó modellek felügyelt gépi tanulással történő létrehozására. Az alábbiakban kiemelünk néhány alapeljárást. Ezeken kívül még számos más megközelítés is létezik (ld. Bodon, 2010.) A neurális hálók is a gyakran használt osztályozó eljárások közé tartoznak, azokat a 6. fejezetben ismertettük.

Alap eljárások:

- a döntési fák módszere,
- logisztikus regresszió,
- támaszvektor-gép (Support Vector Machines, SVM).

A döntési fák módszere

A döntési fák módszerét egy gyakorlati példán keresztül szemléltetjük. A vállalatoknál sokszor okoz gondot a képzett munkaerő elvándorlása. Az emberi erőforrással foglalkozó szakemberek fontos feladata, hogy előre tudják jelezni, ha egy dolgozó szándékában áll a kilépés, lehetőleg még azelőtt, hogy a dolgozó kilépne. Ha előre tudják jelezni a szándékot, akkor olyan körülményeket teremthetnek, hogy sikerül megtartani a munkaerőt.

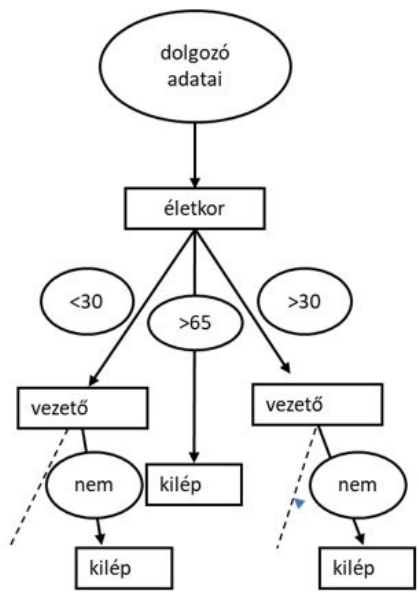
Az EF-szakemberek általában nem tudják, hogy egyik vagy másik dolgozó éppen ki szeretne lépni, csak akkor értesülnek erről, ha már beadta a felmondását. Ugyanakkor számos adat áll rendelkezésre a korábban már kilépettekről és természetesen mindenkiről, aki nem lépett ki. Feladatuk, hogy a rendelkezésre álló adatok alapján megjósolják, hogy kinek áll szándékában kilépni.

A munkatársakról rendelkezésre álló adatok sokfélék, milyen régen dolgozik a vállalatnál, hány éves, milyen nemű, milyen messze lakik, melyik részlegnél dolgozik, milyen beosztásban van, mennyi a fizetése, előléptették-e az utóbbi időben, javul-e, vagy romlik a teljesítménye stb. A már kilépett munkatársak esetén a kilépés ténye is egy bináris változóban rögzített. A legtöbb esetben nem egy, hanem több változó együttesen határozza meg, hogy egy dolgozó ki fog-e lépni vagy nem.

A gépi tanulás célja, hogy kiválassza azokat a változókat, amelyek lényegesen befolyásolják a munkatárs döntését, és azt, hogy ezek mely értékeinél várható a kilépés. A tanuláshoz a már kilépett és a nem kilépett dolgozók adatait lehet felhasználni. Az elemzéssel meg lehet mondani, hogy mi hat jelentősen arra, hogy egy dolgozó kilép-e vagy nem. A szervezeti egység? A vállalatnál eltöltött idő? Az életkor? A beosztás?

A döntési fák

A döntési fák módszere a fenti kérdésekre ad választ. A döntéseket egy fa gráfon ábrázoljuk. A fa gyökerén lépnek be az adott dolgozóra vonatkozó adatok. Minden egyes nem-levél csúcs egy döntést jelent, ahol a dolgozó adataiból eldől, hogy a csúcsból mely ágon megy tovább. A fa minden levele egy végső „kilép" vagy „nem lép ki" döntést ábrázol. Az, hogy egy dolgozó melyik levélre kerül, a megelőző döntések sorozata határozza meg. A döntési fa algoritmus a döntési fa felépítését végzi. A 12. ábrán egy döntési fa szerkezetét illusztráljuk.



12. ábra. Egy döntési fa szerkezete (példa)

(Forrás: Saját szerkesztés.)

12. ábra. Egy döntési fa szerkezete (példa)
(Forrás: Saját szerkesztés.)

Egy feladat döntési fájának létrehozására számos algoritmus ismert. Az egyik leggyakrabban használt és igen hatékony eljárás a C5.0, melynek elődei az ID3 és a C4.5 (Witten et al., 2011). Az algoritmus eredeti szerzője Ross Quinlan (Quinlan, 1993).

A C5.0 algoritmus az „oszd meg és uralkodj” elvet alkalmazza, a döntési fákat ún. rekurzív particionálással hozza létre, azaz az adatokat egyre kisebb, hasonlósági osztályokra bontja.

A teljes adathalmazt képviselő gyökérnél az algoritmus azt az tulajdonságot választja ki, amelyik a célosztályt legjobban meghatározza.

Ezután az objektumokat a fenti attribútum különböző értékei szerint osztályokba sorolja, ez a fa első elágazása. Az algoritmus ezután folytatja a módszert minden csúcsban, kiválasztva a legjobb osztályozó tulajdonságot, amíg egy leállási kritériumot el nem ér.

A leállási kritérium többféle is lehet:

- Egy csúcson minden vagy majdnem minden objektum egy osztályba sorolódik.
- Nem maradt olyan attribútum, amely alapján folytatni lehetne.
- A fa elért egy előre megadott méretet.

A kérdés az, hogy hogyan történik az egyes csúcsokon az osztályba sorolás, melyik változó adja a legjobb felosztást, általában egy adott szinten mit jelent a legjobb felosztás.

Ha egy szegmens csak egy osztály elemeit tartalmazza, akkor tisztának nevezzük. Nyilván minél tisztább egy osztály, annál sikeresebb a felosztás. Többféle mértéket is használhatunk a tisztaság mérésére és a felosztó tulajdonság kiválasztására.

A C5.0 az entrópiát alkalmazza a tisztaság mérésére. Az entrópia azt mutatja, hogy mennyire heterogén egy osztály. Példánkban a heterogenitáson azt értjük, hogy egy osztályon belül mennyire keverednek a kilépő és a nem kilépő munkatársak. Ha az entrópia értéke 0, akkor az osztály homogén, ha 1, akkor maximálisan rendezetlen, azaz azonos számú kilépő és nem kilépő munkatársat tartalmaz.

$$Entrópia(S) = \sum_{i=1}^{n} -p_i * log_2(p_i)$$

Az *S* a teljes halmazt jelenti, *n* az osztályokat, p_i az egyes osztályokba tartozás valószínűségét.

Az algoritmus a tisztasági mérték alapján dönti el, hogy mely változó alapján osztályozzon. Az entrópia mutatja meg, hogy az egyik felosztás jobb-e, mint a másik. Ha „információt nyerünk”, akkor jobb, egyébként nem jobb.

Egy X változó információs nyeresége a felosztás előtti osztály (*S1*) entrópiája és a felosztás utáni osztályok (*S2*) entrópiájának különbsége.

Információs nyereség (X) = Entrópia(S1) – Entrópia(S2)

Az *Entrópia (S2)* számításánál már több részosztályunk van a felosztás következtében, ezért a több részosztály entrópiájának súlyozott összegével számolunk. A w_i súly az adott részosztályba eső adatok aránya

$$Entrópia(S2) = \sum_{i=1}^{k} Entrópia(P_i),$$

ahol P_1, P_2, ... P_k az *S2* részosztályai.

Minél nagyobb az információs nyereség, annál jobb a felosztás. Ha az információs nyereség nulla, akkor nincs entrópiacsökkenés a felosztásnál, így az adott változó nem segít a klasszifikációban.

Az információs nyereség maximuma megegyezik a felosztás előtti entrópiával, ha ezt elérjük, akkor nem lehet már javítani ezen az ágon az osztályozást, mert az entrópia a felosztás után nulla, tehát teljesen homogén csoportokat kaptunk.

A döntési fák nominális és numerikus attribútumok esetén is használhatók. A nominális változók esetén az értékeket diszjunkt értéktartományokra bontják, és így nominális értékekre képezik le.

Az információs nyereséget könnyű számolni, így az adott döntési pontban legnagyobb információs nyereséget adó változót is könnyen ki lehet jelölni. Az algoritmus akkor fejeződik be, ha már egyetlen döntési ágon sincs olyan változó, amely egy megadott küszöbértéknél nagyobb információs nyereséget adna (Mingers, 1989).

A túlzottan sok szintet tartalmazó döntési fa nem használható jól az osztályozásban (ld. túlillesztés probléma[37]), ezért sokszor szükség van arra, hogy a fa méretét korlátozzuk, azaz csak a legnagyobb információs nyereséget hozó döntéseket vegyük figyelembe. Ez a „metszés” (pruning) művelete.

A döntési fa metszése arra irányul, hogy csökkentse a méretet és elég általános maradjon a felosztás ahhoz, hogy az ismeretlen objektumok klasszifikációjára is alkalmas legyen.

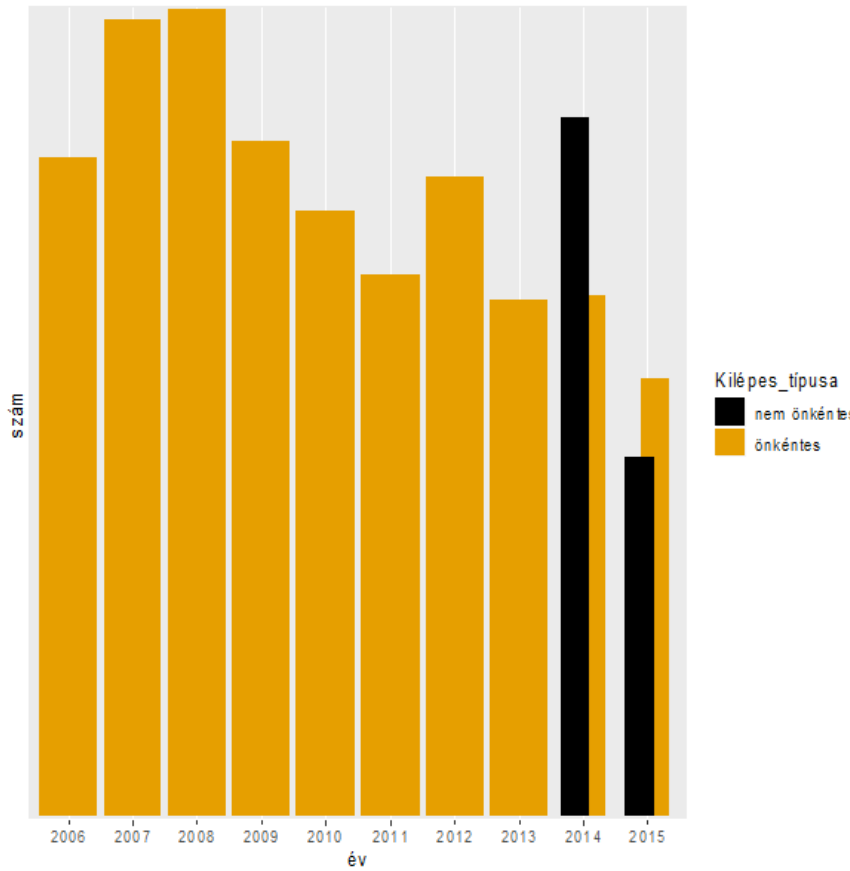
Példaképpen bemutatjuk a munkahelyről történő kilépés modelljét. A feladatot R nyelven programoztuk, a tanulóadatfájlt a Kaggle oldaláról[38] vettük. A 4. táblázatban a kilépők részlegek szerinti számát mutatjuk.

	részlegek	**kilépők**
1	központ_kilépett	59
2	központ_aktív	411
3	bolt_kilepett	1416
4	bolt_aktív	47652
5	központ_%	12.55
6	bolt_%	2.89

4. táblázat. A kilépő és az aktív dolgozók száma
(Forrás: Sajátszerkesztés.)

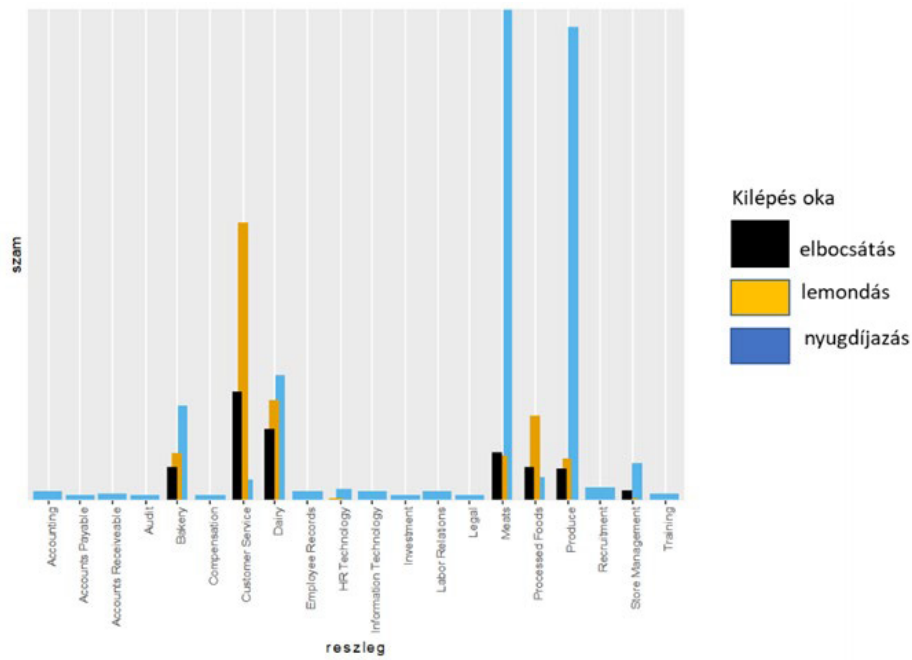
37 A túlillesztés azt jelenti, hogy a modell jó eredményeket szolgáltat a tanuló adatokon, de a teszt és az éles adatokon nem, mert nem általános érvényű, hanem csak a betanulásához használt adatokra megfelelő.

38 https://www.kaggle.com/lyndonsundmark/attrition-analysis-sample-script/data

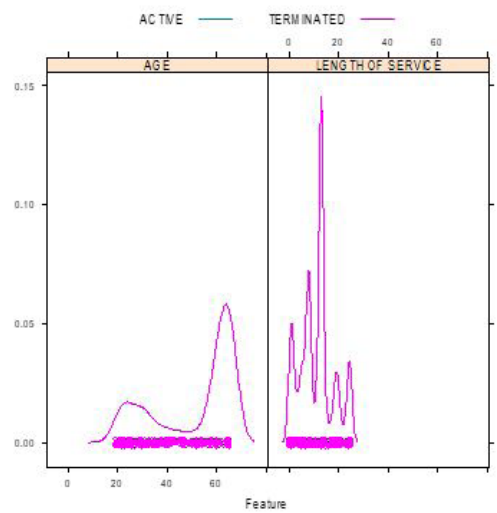


13. ábra. A kilépés típusának grafikonja (sárga – az alkalmazott kezdeményezésére, fekete – nem az alkalmazott kezdeményezésére)
(Forrás: Saját szerkesztés.)

A 13. ábrán a kilépők számának változását látjuk éves bontásban. A nem önkéntes kilépés azért csak 2014-től különbözik nullától, mert addig nem volt elbocsátás.



14. ábra. A kilépők száma részlegenként (A fekete az elbocsátást, a sárga a kilépést, a kék a nyugdíjba vonulást jelenti.) (Forrás: Saját szerkesztés.)



15. ábra. Az aktív és a kilépett dolgozók kor és szolgálati idejének gyakorisági grafikonja (Forrás: Saját szerkesztés.)

Az adatok megismerése után elkészítjük a modellt. Körülbelül 5000 dolgozó adatai állnak rendelkezésre, ebből véletlenszerűen leválogatunk 4500 adatot a modell betanítására, a maradékot a tesztelésre tartalékoljuk.

A modell ezen az adatmennyiségen kb. 1 mp alatt lefut. A kapott döntési fa szerkezete:

Döntési fa:

```
city_name in {Blue River,Cortes Island,Dawson Creek,Dease Lake,Fort Nelson,
    Grand Forks,Haney,New Westminister}: TERMINATED (127)

    city_name in {Abbotsford,Aldergrove,Bella Bella,Burnaby,Chilliwack,Cranbrook,
    Fort St John,Kamloops,Kelowna,Langley,Nanaimo,Nelson,
    New Westminster,North Vancouver,Ocean Falls,Pitt Meadows,
    Port Coquitlam,Prince George,Princeton,Quesnel,Richmond,Squamish,
    Surrey,Terrace,Trail,Valemount,Vancouver,Vernon,Victoria,
    West Vancouver,White Rock,Williams Lake}:
    :...age > 64: TERMINATED (46)
       age <= 64:
       :...length_of_service > 1: ACTIVE (4132/1)
          length_of_service <= 1:
          :...age <= 20: ACTIVE (106)
             age > 20: TERMINATED (89/40)
```

Látjuk, hogy a legnagyobb információs nyereséget adó döntési kritérium a földrajzi hely, utána az, hogy elérte-e a nyugdíjkorhatárt (64 év), majd az, hogy 20 évnél idősebb-e, vagy nem.

A döntési fát szabályrendszerré konvertálhatjuk az alábbi módon[39]:

```
Rules:
    Rule 1: (4327/50, lift 1.0)
        age <= 64
        city_name in {Abbotsford, Aldergrove, Bella Bella, Burnaby, Chilliwack,
                Cranbrook, Fort St John, Kamloops, Kelowna, Langley,
                Nanaimo, Nelson, New Westminster, North Vancouver,
                Ocean Falls, Pitt Meadows, Port Coquitlam, Prince George,
                Princeton, Quesnel, Richmond, Squamish, Surrey, Terrace,
                Trail, Valemount, Vancouver, Vernon, Victoria,
                West Vancouver, White Rock, Williams Lake}
        -> class ACTIVE  [0.988]
    Rule 2: (127, lift 20.0)
        city_name in {Blue River, Cortes Island, Dawson Creek, Dease Lake,
                Fort Nelson, Grand Forks, Haney, New Westminister}
        -> class TERMINATED  [0.992]
    Rule 3: (48, lift 19.8)
        age > 64
        -> class TERMINATED  [0.980]
    Default class: ACTIVE
```

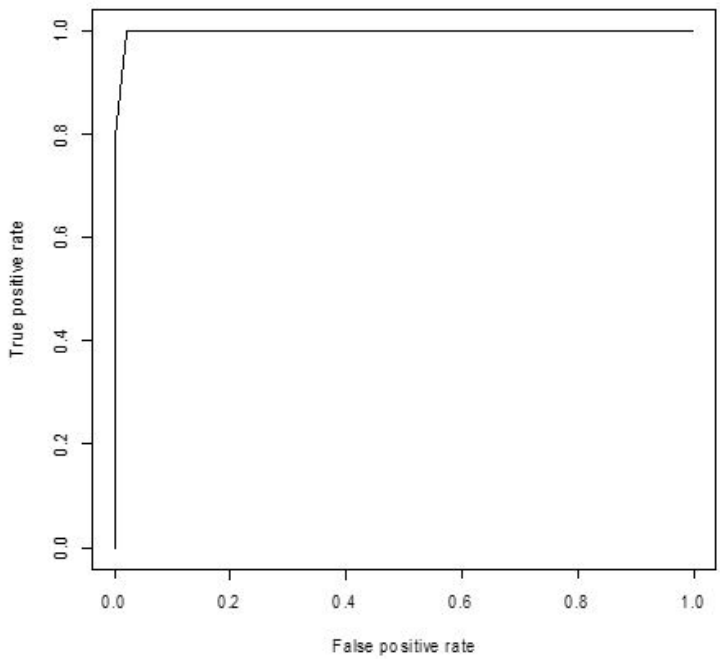
A szabályrendszer könnyebben kezelhető egy EF-munkatárs számára, mint egy döntési fa. Az első szabály azt mutatja, hogy először azt kell megnézni, hogy a munkatárs elérte-e a nyugdíjkorhatárt, majd azt, hogy melyik városban dolgozik és sorban a többi szabályt. Ha az EF-munkatárs egy konkrét dolgozó adatait behelyettesíti a szabályrendszerbe, becslést kap arra, hogy a dolgozó ki akar-e lépni vagy nem.

		Modell szerinti besorolás	
		Active	Terminated
Tényleges	Active	4237	40
	Terminated	1	222

5. táblázat. A példában kapott konfúziós vagy tévedési mátrix (ld. 11. fejezet) (Forrás: Saját szerkesztés.)

A konfúziós mátrixban látjuk, hogy a 4317 aktív dolgozóból 4237-et helyesen, míg 40-et helytelenül sorolt be a program. A kilépőknél a helyes találatok aránya a 223-ból 222, míg 1-nél tévedett az algoritmus. (A nagy pontosság annak köszönhető, hogy itt a konfúziós mátrixot a tanulóadatokból nyertük.)

39 A szabályt a "->" nyíllal jelöltük.



16. ábra. A példa modell ROC-görbéje
(Forrás: Saját szerkesztés.)

A pontosság javítása érdekében alkalmazhatjuk az ún. boosting eljárást, amely többször is lefuttatja modellt, és a végső eredmény több futás összegzésével készül. Egy 10 kísérletből álló sorozat eredményének konfúziós mátrixa a tesztadatokon (6. táblázat).

		Modell szerinti besorolás	
		Active	Terminated
Tényleges	Active	667	13
	Terminated	0	25

6. táblázat. A boosting eljárással futtatott modell konfúziós mátrixa
(Forrás: Saját szerkesztés.)

A teljes adatmennyiségre lefuttattunk egy logisztikus regressziós modellt is, amely az alábbi eredményeket adta (7. táblázat):

A regressziós koefficiensek:

	Becsült érték	std. hiba	hiba z érték	P(>|z|)
Konstans	-23.2	2.16	-10.72	<2e-16
Kor	1.01	0.11	9.58	<2e-16
Szolg. idő	-2.11	0.22	-9.68	<2e-16

7. táblázat. A logisztikus regressziós egyenlet koefficiensei a példában (Forrás: Saját szerkesztés.)

Látjuk, hogy eszerint a leginkább meghatározó paraméter a kor, míg negatív előjellel a szolgálati idő.

A logisztikus regresszió alkalmazása klasszifikációs feladatokra

A logisztikus regressziót akkor használjuk, ha a függő változónk dichotóm, azaz két értéket vehet fel, például 0-t vagy 1-et. A klasszifikációs modelleknél éppen erről van szó, ha a független változó értéke 1, ez azt mutatja, hogy az adott minta beletartozik egy célosztályba, míg a 0 érték azt mutatja, hogy nem. A logisztikus regresszió a függő bináris változó és egy vagy több nominális, ordinális, intervallum vagy numerikus változó közötti kapcsolatot írja le.

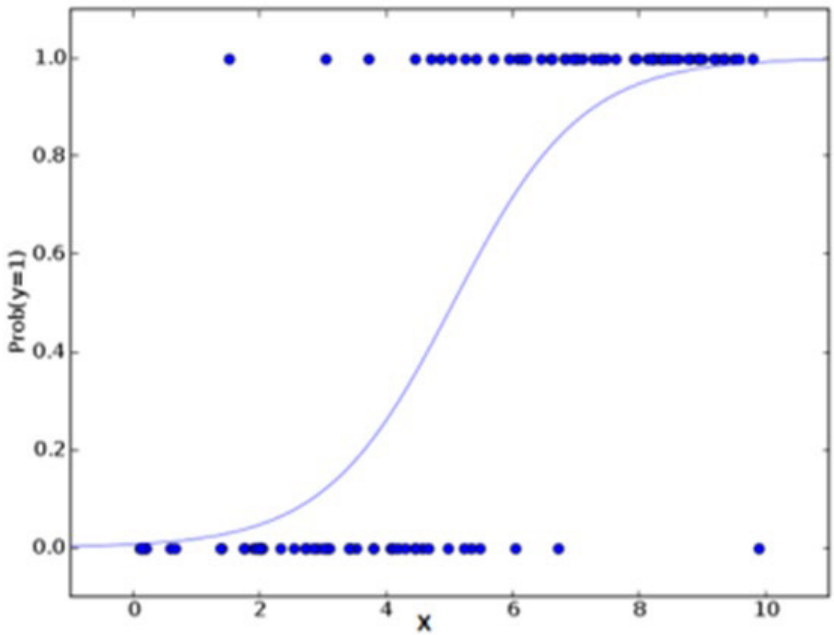
A bináris logisztikai regresszió ún. általánosított lineáris modell (Generalized Linear Model, GLM), mert a regressziós egyenlet jobb oldalán a független változók lineáris kombinációja található. Az egyenlet bal oldalán a bináris változó két értéke valószínűsége hányadosának logaritmusa található. Az előző, munkahelyről való kilépés modelljében jelöljük Y-nal azt a bináris valószínűségi változót, amely 1 értéket vesz fel, ha a dolgozó kilép, 0-t, ha nem. Legyen $p = P(Y = 1)$, ekkor $P(Y = 0) = 1-p$. Jelöljük X_1, X_2, ..., X_k-val a független változókat.

Ekkor a logisztikus egyenlet az alábbi:

$$\ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \cdots + \beta_k * X_k.$$

Az $\ln\left(\frac{p}{1-p}\right)$ értéket logit(p)-nek is nevezzük, a $\frac{p}{1-p}$ -t pedig esélynek (odds).

A logisztikus regressziós egyenlet jobb oldalán folytonos vagy nominális változók is lehetnek, összegük nem feltétlenül lesz a [0,1] intervallumban. Ezért vezetjük be a bal oldalon a logit függvényt, amelynek értéke 0 és 1 között van (ld. 17. ábra).



17. ábra. Logit görbe és a tényleges 0 vagy 1 értékek
(Forrás: Saját szerkesztés.)

A logisztikus regressziót alkalmazhatjuk, ha például:

- modellezni szeretnénk a függő változó valószínűségi értékeit a független változók függvényében (például a siker valószínűségét egy többfordulós felvételi vizsgán),
- két csoportba szeretnénk osztani személyeket tulajdonságaik alapján (például meg szeretnénk határozni azokat a jó ügyfeleket, akik gyakran látogatnak el egy weboldalra és többször vásárolnak).

A logisztikus regressziós egyenletet átalakítjuk az alábbi formára:

$$p = \frac{e^{\beta_0+\beta_1*X_1+\beta_2*X_2+\cdots+\beta_k*X_k}}{1+e^{\beta_0+\beta_1*X_1+\beta_2*X_2+\cdots+\beta_k*X_k}},$$

ahol *p* a becsült output, β_0, β_1, ... β_k a regressziós paraméterek.

Összességében, a regressziós egyenlet továbbra is lineáris, de a bal oldali érték nem az Y értéke, hanem a p/(1-p) esély logaritmusa.

A logisztikus regresszió β paramétereit a tanulóadatokból becsüljük meg a maximum likelihood (legnagyobb valószínűségek) módszerrel. A módszer lényege azoknak a paramétereknek a megtalálása, amelyek mellett a regressziós függvény minimalizálja az osztályba tartozás valószínűségeinek a becslési hibáit. Az optimalizálást valamely numerikus módszerrel (gradiens módszerrel vagy az ún. kvázi-Newton algoritmussal stb.) végzik.

A paraméterek meghatározása után már könnyű a klasszifikáció, csak a számokat (vagy nominális értékeket) kell behelyettesíteni és megkapjuk, hogy mekkora a valószínűsége az egyes osztályokba tartozásnak.

A logisztikus regressziós modellben két numerikus változót használtunk, az eredményeket már bemutattuk a 7. táblázatban. A táblázatból kiolvasható, hogy a leginkább meghatározó paraméter a kor, míg negatív előjellel a szolgálati idő.

A regressziós függvény értéke tehát:

-23,2+1,01*(életkor)-2,11* (szolgálati idő)

Legyen az életkor = 40 év, a szolgálati idő 8 év.

A regresszió jobb oldalának értéke 0,32 és p = 0,58, így annak a valószínűsége, hogy a 40 éves, 8 év munkaviszonnyal rendelkező dolgozó nem lép ki, 0,58.

Az értékelésnél eldönthetjük, hogy mekkora valószínűség felett tekintjük az objektumot egy osztályba tartozónak. Például, ha p ≥ 0,5, akkor a nem kilépők osztályába tartozik.

Nem véletlenül választottuk a fentiekben csak a numerikus változókat, mert nominális változók esetén a logisztikus regresszió végrehajtásához át kell alakítani a változókat. A nominális változókat az algoritmus automatikusan bináris változókká alakítja a számolásnál. Ha például a kategóriaváltozónak 3 különböző értéke van, akkor ehelyett 3 darab bináris változót használ, egyet minden kategóriához. Ezek értéke 1, ha a kategóriaváltozó értéke a releváns érték, egyébként nulla.

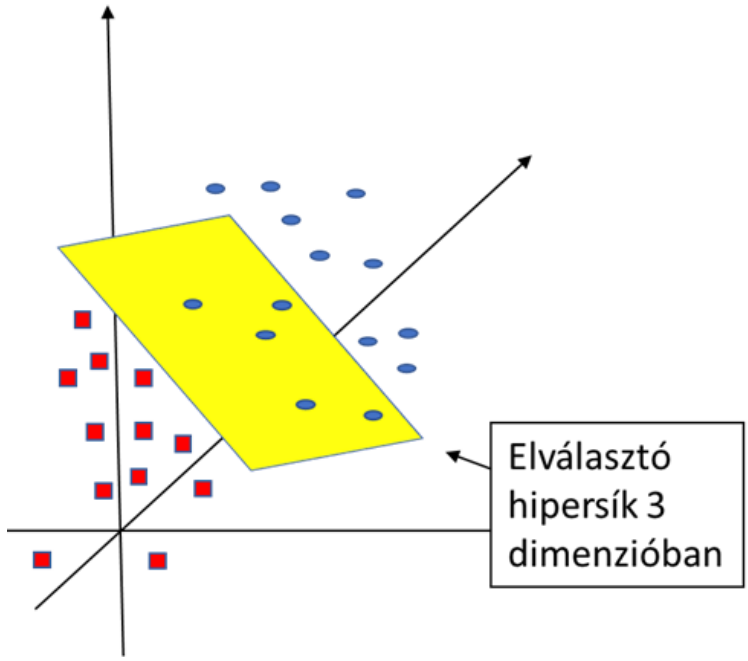
Megjegyezzük, hogy a logisztikus regresszióhoz használt adatok minősége lényeges, az eljárás nagyon érzékeny az adatok hibáira. Lényeges szempontok:

- Az adatok nem lehetnek zajosak, a bemenő adatok közül ki kell szűrni a kiugró értékeket és a hibás adatokat.
- A logisztikus regresszió lineáris algoritmussal működik, a végén egy nemlineáris transzformációval. Feltételezi, hogy az input változók lineáris viszonyban vannak az outputtal. Léteznek olyan transzformációs módszerek, amelyek jobban kifejezik a lineáris viszonyt, pl. a változók logaritmusának használata, a Box-Cox-módszer stb. Ezek pontosabb modell kialakítását teszik lehetővé.
- A korrelált inputok kivétele – ha az input adatok között magas a korreláció, akkor a modell túlillesztett lesz. Célszerű kiszámítani a változók közötti páronkénti korrelációt és eltávolítani a magasan korrelált változópárok egyikét.
- Az MLE-módszer egyes esetekben nem konvergál. Ez például akkor történhet, ha az input adatok erősen korreláltak vagy ha az input adatok ritkák, azaz sok a nulla.

Az SVM-algoritmus

A támaszvektor-gép olyan felület, amely egy többdimenziós tér pontjaiként ábrázolt objektumokat választja el egymástól. Az SVM-eljárás célja olyan hipersík létrehozása, amelynek mindkét oldalán a pontok homogén, vagy közel homogén, halmazt alkotnak. Az SVM tehát klasszifikációs eljárás, amely a geometriailag pontként ábrázolt objektumokat felügyelt tanulással két vagy több osztályba sorolja, hasonlóan a döntési fákhoz és a logisztikus regresszióhoz.

Az SVM az esetek nagy részében igen hatékony módszernek bizonyult az utóbbi időben, számos gépi tanulási versenyt nyertek az elemzők SVM-algoritmussal. Különösen jó tapasztalatok vannak a mintafelismerési algoritmusok terén, pl. kézzel írott szövegek felismerésében.



18. ábra. A ponthalmazokat elválasztó hipersík
(Forrás: Saját szerkesztés.)

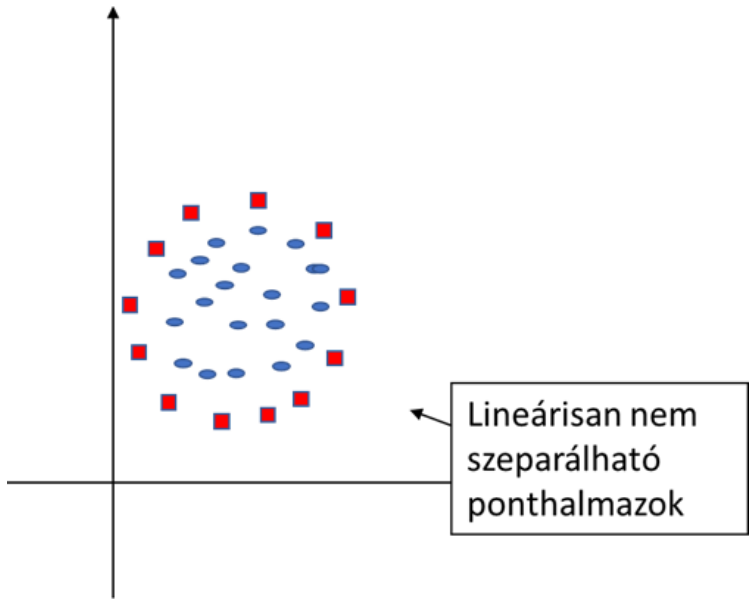
Jelentős alkalmazási területek:

- Genetika. A „microarray” („chip”) módszer az egyidejűleg vizsgálható gének számának növelésével szerkezeti és funkcionális információ tömegét képes nyújtani, az SVM-algoritmus segít a genetikai alapú betegségek és a rák azonosításában.
- Természetes nyelvi szövegek kategorizálása nyelvek vagy tartalom szerint.
- Ritka, de fontos események felismerése, mint például egy számítógépes támadás, földrengés, vagy ritka betegségek felismerése.

Az SVM-technikát legegyszerűbb a bináris klasszifikáción megérteni, ezért ennek ismertetésével kezdjük, majd a későbbiekben foglalkozunk a többosztályos klasszifikációval és az SVM használatával a predikciós feladatokban.

Klasszifikáció hipersíkokkal

A 18. ábrán háromdimenziós térben elhelyezkedő pontokat választottunk szét egy kétdimenziós síkkal. Az ábrán azt az ideális esetet ábrázoltuk, amikor ez könnyen megtehető, végtelen sok olyan síkot tudunk definiálni, amely szeparálja a két ponthalmazt. Ha létezik ilyen hipersík (és most többdimenziós térről beszélünk), akkor a ponthalmazok lineárisan szeparálhatók. A 19. ábra a kétdimenziós térben lineárisan nem szeparálható ponthalmazokat ábrázol. A későbbiekben erre az esetre is találunk majd megoldást.



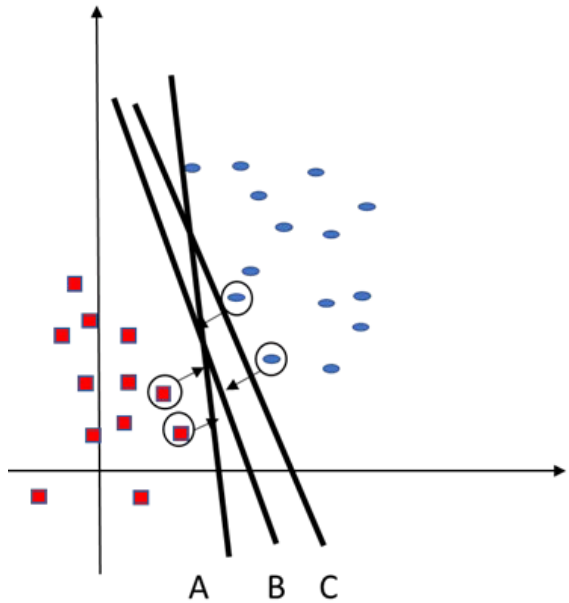
19. ábra. Lineárisan nem szeparálható ponthalmazok
(Forrás: Saját szerkesztés.)

Az SVM célja a szeparálható osztályokat elválasztó hipersík megtalálása. A 19. ábrán látható, hogy végtelen sok ilyen hipersík létezik, hiszen a pontok száma véges, így a sík mozgatható anélkül, hogy az osztályok elemei keverednének.

Hogyan választ az SVM-algoritmus az A, B és C egyenesek közül? Az eljárás a maximális margójú hipersíkot (Maximum Margin Hyperplane, MMH) keresi meg. Az MMH-sík választja el „legjobban" a halmazokat.

A 20. ábrán az A, B és C egyenesek egyaránt szétválasztják a ponthalmazokat, azt látjuk, hogy a B egyenes távolsága a legnagyobb a halmazok pontjaitól. A B egyeneshez legközelebbi pontok a támaszvektorok (Support Vectors), az eljárás neve is innen származik. Az ábrán a támaszvektorokat bekarikáztuk. Minden osztály kell, hogy tartalmazzon legalább egy támaszvektort, de természetesen több is létezhet.

Az MMH kiválasztása mögött az az intuitív meggondolás áll, hogy a jövőbeni adatok osztályba sorolásánál a legnagyobb MMH-val rendelkező hipersík a legbiztosabb, mert így a legkisebb annak az esélye, hogy az új objektum „átesik" a hipersík másik oldalára. A támaszvektorok alapján az MMH egyértelműen meghatározható. Ez a tulajdonság nagyon fontos az SVM-algoritmusnál, a klasszifikációs modell alkalmazásához elegendő a támaszvektorokat tárolni, a többi pontra nincs szükség.



20. ábra. A két osztályt elválasztó hipersíkok
(Forrás: Saját szerkesztés.)

Az SVM kiszámításának algoritmusát már az 1990-es években publikálták (ld. pl. Cortes, 1995). Az azóta elért eredményeket is tartalmazza Steinwart et al. (2008 cikke.

Lineárisan szeparálható adatok esetén az MMH a lehető legtávolabb helyezkedik el a halmazok határától, matematikai nyelven a ponthalmaz konvex burkától[40].
Az MHH formális meghatározása:
Vegyünk egy *n* dimenziós euklideszi teret. A

$$\vec{w} * \vec{x} + b = 0$$

egyenlet egy hipersíkot határoz meg. A nyilak azt jelölik, hogy *w* és *x* vektorok, vagyis $w=(w_1,w_2,..w_n)$, $x=(x_1,x_2,..x_n)$ és b konstans.

Az algoritmus célja, hogy találjon egy olyan w vektort, amely meghatároz két olyan hipersíkot, amelyekre:

$$\vec{w} * \vec{x} + b \geq +1$$

$$\vec{w} * \vec{x} + b \leq -1$$

40 Egy ponthalmaz halmaz konvex burka az a legkisebb konvex halmaz, amely a ponthalmazt tartalmazza.

A két hipersíkkal szemben további elvárás, hogy a lineárisan szeparálható ponthalmazok a hipersíkok különböző oldalaira kerüljenek úgy, hogy a két hipersík között ne legyenek halmazelemek. Ilyen hipersíkok a lineárisan szeparálható ponthalmazok esetében léteznek.

A két hipersík távolsága

$$\frac{2}{\|\vec{w}\|},$$

ahol $\|\vec{w}\|$ a $\vec{w}$ vektor hossza. Ezt az értéket kell maximalizálni az MMH meghatározásához valamely hatékony optimumkereső eljárással. A problémát az alábbi formában szokás kezelni:

$$min\frac{1}{2}\|\vec{w}\|^2$$

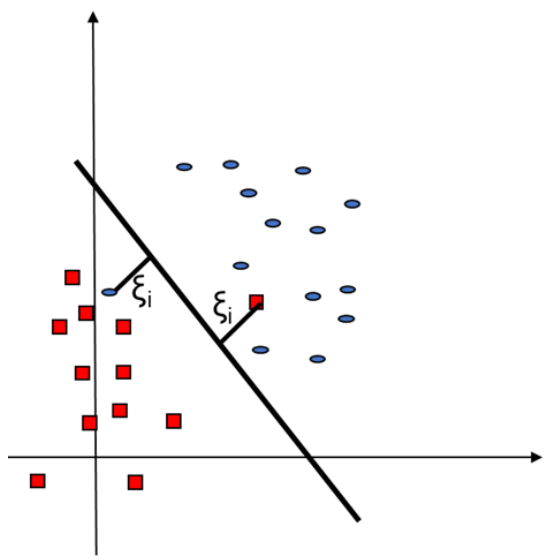
úgy, hogy az osztályba tartozást jelző, +1 vagy -1 értékű y_i-re teljesüljön, hogy

$$y_i(\vec{w} * \vec{x_i} - b) \geq 1$$

minden $\vec{x_i}$ esetén.

Az SVM alkalmazása lineárisan nem szeparálható halmazok esetén

Az alábbi, 21. ábrán két, lineárisan nem szeparálható halmazt látunk.



21. ábra. Két, lineárisan nem szeparálható halmaz
(Forrás: Saját szerkesztés.)

A hipersík „rossz oldalán” elhelyezkedő pontok esetén bevezetünk egy ún. kiegyenlítő (slack) változót, amely a pontnak a hipersíktól mért távolságát jelöli. Ezzel megengedjük, hogy egyes pontok a hipersík másik oldalán helyezkedjenek el, de ennek „költségét” figyelembe vesszük a számításoknál. Így az optimalizálandó függvény módosított formája:

$$\min \left(\frac{1}{2}\|\vec{w}\|^2 - C * \sum_{i=1}^{n} \xi_i\right) ,$$

ahol

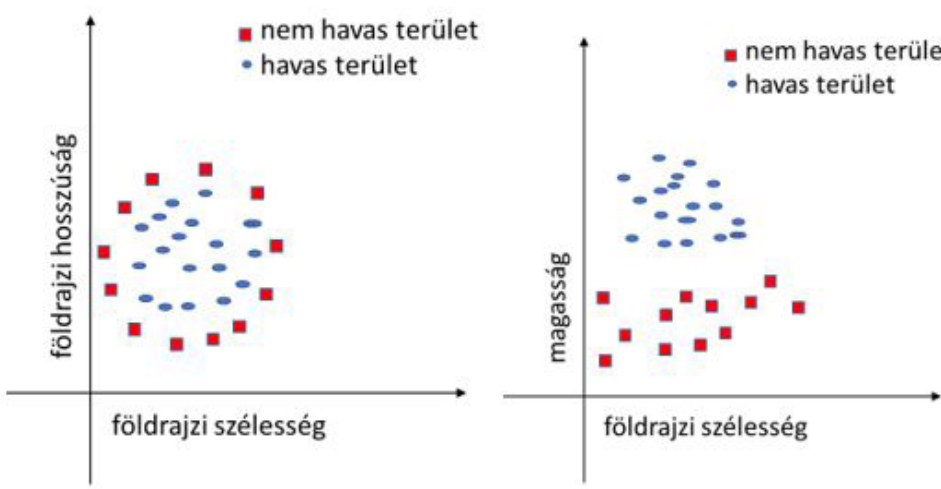
$$y_i(\vec{w} * \vec{x_i} - b) \geq 1\text{-}\xi_i$$

minden $\vec{x_i}$-re és $\xi_i \geq 0$-ra.

A C költségtényezővel befolyásolhatjuk, hogy milyen szeparátor hipersíkot szeretnénk kapni. Ha C nagy, nehéz elérni a 100% közeli szeparációt, ha kisebb, akkor szélesebb a margó. Az adatelemző feladata, hogy a megfelelő egyensúlyt megteremtse a két véglet között.

Az SVM alkalmazása nemlineáris összefüggések esetén

A 22. ábra baloldali része tipikusan nemlineáris összefüggést mutat. A vízszintes tengelyen a földrajzi szélességet, a függőlegesen a földrajzi hosszúságot ábrázoljuk, a kék pontok jelentik a hóval borított területeket, a pirosak pedig a hóval nem borított helyeket.



22. ábra. Lineáris szeparálhatóság létrehozása újabb dimenzió bevezetésével (Forrás: Saját szerkesztés.)

A 22. ábra jól szemlélteti, hogy hogyan lehet lineárisan nem szeparálható halmazokból egy (vagy több) dimenzió hozzáadásával szeparálható halmazokat előállítani. Az eljárást kernel-trükknek is nevezik a szakirodalomban.

A példában egy új változó, a magasság bevonásával szeparálhatóvá tettük a halmazokat. Nyilvánvaló, hogy a magasság a földrajzi szélesség és hosszúság függvénye, tehát a transzformációt a rendelkezésre álló adatok alapján végre lehet hajtani.

A kernel-trükk eljárást komplex nemlineáris esetekben is sikerrel lehet alkalmazni.

Példa az SVM alkalmazására

A példát R nyelven programoztuk és a szakirodalomban egyik leggyakrabban használt adathalmazt, az írisz mintaadathalmazt használjuk. Az írisz-adatok az írisz virág (magyarul nőszirom) háromféle fajának, egyenként 50 x 4 mérési adatát tartalmazza. Az adatok R. Fishertől, a 20. század egyik legnagyobb statisztikai kutatójától származnak (Fischer, 1950). Az írisz-adatokat majd minden klasszifikációs feladat tesztelésénél használják, így ez jó referencia az egyes algoritmusok hatékonyságának összehasonlítására is. Az egyes osztályok között vannak lineárisan szeparálhatók és nem szeparálhatók.

23. ábra Az írisz virága

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5,10	3,50	1,40	0,20	setosa
2	4,90	3,00	1,40	0,20	setosa
3	4,70	3,20	1,30	0,20	setosa
4	4,60	3,10	1,50	0,20	setosa
5	5,00	3,60	1,40	0,20	setosa

8. táblázat. Minta az írisz adatokból
(Forrás: Saját szerkesztés.)

A változók:
sepal length in cm (a csészelevél hossza)
sepal width in cm (a csészelevél szélessége)
petal length in cm (a virágszirom hossza)
petal width in cm (a virágszirom szélessége)

a faj neve: (Iris Setosa, Iris Versicolour, Iris Virginica)

Az SVM algoritmust használjuk a klasszifikátor modell létrehozására, amely azután egy ismeretlen írisz fajt be tud sorolni a fenti fajok valamelyikébe. [41]

A konfúziós mátrix:

		tényleges osztályba tartozás		
		setosa	versicolor	virginica
a modell besorolás	setosa	50	0	0
	versicolor	0	48	2
	virginica	0	2	48

9. táblázat. AZ SVM futtatása során kapott konfúziós mátrix
(Forrás: Saját szerkesztés.)

41 Az írisznek a fenti három fajon kívül még számos egyéb faja is létezik, a feladatban ezt nem vesszük figyelembe.

Látjuk, hogy az algoritmus négy esetben tévedett, amikor 2 versicolort virginicának és két virginicát versicolornak sorolt be.

A modell javítását többszöri futtatással és a paraméterek, valamint a kernel-függvény tuningolásával lehet megkísérelni, de szinte bizonyos, hogy a fentinél jobb konfúziós mátrixot nem kaphatunk.

A naiv Bayes osztályozó eljárás

A naiv Bayes hatékony, könnyen betanítható klasszifikációs modell, amely a Bayes-tétel alapján meghatározza egy kimenetel feltételes valószínűségét adott feltételek mellett. A feltételes valószínűségeket invertálva az osztályba tartozást a mérhető paraméterek függvényeként határozzuk meg.

A modell igen egyszerű, és a „naiv” jelző nem azt jelenti, hogy a módszer kevéssé hatékony, hanem azt, hogy abból az alapvető feltételezésből indulunk ki, hogy vannak olyan tényezők, amelyek meghatározzák az osztályba tartozást.

A naiv Bayes sokoldalú klasszifikációs módszer, számos területen alkalmazzák, de különösen ott hatékony, ahol egy osztályba tartozás valószínűségét a mérhető tényezők valószínűsége határozza meg. Például a természetes nyelvi feldolgozásban gyakran használják, a szöveg egy meghatározott egységét egy szótári bejegyzés előfordulásának tekinthetjük és a szöveg egységeinek relatív gyakorisága elég információval szolgál arról, hogy az adott egység melyik osztályba tartozik.

A Bayes-tétel

Tekintsünk két valószínűségi eseményt, A-t és B-t. Előfordulhat, hogy ha az egyik esemény már bekövetkezett, akkor ez megváltoztatja a másik esemény bekövetkezésének valószínűségét az eredeti valószínűséghez képest. Legyen például az A esemény az, ha egy kockával 6-ost dobunk, a B esemény pedig az, ha két kockát feldobva a pontok összege 12. Ha feldobjuk az első kockát és a kijövő érték kisebb, mint 6, akkor a B bekövetkezésének valószínűsége nulla, míg, ha a kijövő érték 6, akkor a B bekövetkezésének valószínűsége 1/6.

Legyen P(A) és P(B) az A és B események valószínűsége, P(A|B) az A feltételes valószínűsége a B feltétellel, P(B|A) pedig a B feltételes valószínűsége az A feltétellel.

A Bayes-tétel azt mondja ki, hogy:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}.$$

Egyszerűsége mellett a Bayes-tétel, amellett, hogy a statisztika egyik alapköve, komoly filozófiai következtetést is tartalmaz.[42] Vegyük a P(A) valószínűséget, ez az érték az A esemény előfordulásának valószínűségét fejezi ki, például *P(spam)* annak a valószínűsége, hogy egy számítógépes levél spam. Ezt a priori valószínűségnek nevezzük, mert vagy matematikai úton, vagy az előfordulás gyakorisága alapján tudunk rá következtetni. Egy spamszűrőt szeretnénk készíteni, és ehhez összegyűjtünk 100 e-mailt, amelyből 30 spam, 70 nem spam. Így P(spam) = 0,3.

42 A tétel alapgondolata az, hogy egy hipotézis helyességét egy annak valószínűségét mutató adathalmazzal lehet alátámasztani. Ez a szubjektivista filozófia sarokköve.

A szűréshez szeretnénk használni egy e-mailre jellemző kritériumot, pl. „egy e-mail spam, ha kevesebb, mint 60 karakter". A Bayes-tétel szerint:

$$P(spam|szöveg < 60\ karakter) = \frac{P(szöveg < 60\ karakter|spam)P(spam)}{P(szöveg < 60\ karakter)}.$$

Az egyenlet baloldali része a spam valószínűsége az adott feltétel teljesülése mellett, azaz, tudjuk, hogy a hossza kevesebb, mint 60 karakter. Ez az esemény a posteriori valószínűsége. Ha például a 60 karakternél rövidebb e-mailek száma 35, akkor *P(szöveg < 60 karakter) = 0,35*. Minthogy a 100 e-mailt már kézzel spam és nem spam osztályokba soroltuk, megszámoljuk, hogy hány spam hossza kisebb, mint 60 karakter. Legyen ez 25.

Így *P(szöveg < 50karakter|spam) = 25/30=0,83* és a jobb oldal értéke *0,83*0,3/0,35 = 0,71.* Ha tehát az e-mail hossza kevesebb, mint 60 karakter, akkor annak a valószínűsége, hogy spam, 0,71. Ez azt jelenti, hogy ha egy bejövő e-mail rövidebb, mint 60 karakter, akkor 71%-os a valószínűsége annak, hogy a spam osztályba tartozik.

Látjuk, hogy ha van megfelelő tanuló adathalmazunk és az objektumokat előzetesen felcímkézzük, akkor ki tudjuk számolni, hogy egy adott objektum (amelyet nem címkéztünk fel) milyen valószínűséggel fog az egyes osztályokba tartozni.

A képlet nevezőjében lévő érték normalizáló tényező, amit α-val is szoktak jelölni:

$$P(A|B) = \alpha P(B|A)P(A).$$

Ha a besorolás nem egyetlen, hanem több, C_1, C_2, ..., C_n tulajdonság alapján történik, amelyekről feltételezzük, hogy egymástól függetlenek, akkor

$$P(A|C_1 \cap C_2 \cap \ldots \cap C_n) = \alpha P(C_1|A)P(C_2|A) \ldots P(C_n|A)P(A).$$

A függetlenség feltétele sokszor nem teljesül. A spam-példában, ha a hosszúság feltétele mellett még feltételnek tekintjük, hogy a szövegben szerepeljenek olyan kulcsszavak, mint „fogyás", „gyors nyelvtanulás", „gyors gyógyulás" stb. akkor a két feltétel nem független egymástól. A legtöbb esetben a függőség azonban nem rontja le lényegesen a hatékonyságot (Zhang, 2004).

A kulcsszavak kijelölésénél természetesen figyelni kell a kétértelműségekre, pl. az angol „Are you free tomorrow?" és a „Free pills" közül az első nem spam, a második nyilvánvalóan az.

Az alábbi példán a naiv Bayes alkalmazását mutatjuk be sms-ek spamszűrésére. Az adathalmazt[43] egy spam/nem spam gyűjteményből vesszük, amelyet az Almeida et al. (2013) cikk írói hoztak létre.

Az adatbázis 5559 sms-t tartalmaz, felcímkézve a ham (legális mail) és a spam címkékkel. 4812 a legális sms-ek, 747 a spamek száma.

Az sms-ekből létrehozunk egy szövegkorpuszt, amellyel azután betanítjuk a naiv Bayes-algoritmust. (a korpusz fogalmával és a természetes nyelvi elemzéssel a 9. fejezet-

43 http://www.dt.fee.unicamp.br/~tiago/smsspamcollection/

ben foglalkozunk). Ezután megtisztítjuk a korpuszt a stopszavaktól, felesleges szóközöktől számoktól, írásjelektől (ld. a 9 fejezetet).

Például a „Just checking in the AAA airlines” sms-ből a „just checking aaa airlines” szöveget kapjuk a tisztítás után.

A tisztított korpuszból létrehozzuk a Document-Term-Matrixot, amely az egyes szavak előfordulási gyakoriságát tartalmazza sms-enként. A mátrix sorai az sms-eknek, az oszlopai a szavaknak felelnek meg, egy sor és egy oszlop közös elemében egy szám szerepel, amely megmutatja, hogy az adott sms-ben az adott szó hányszor szerepel. A mátrix sorainak száma 5599, az oszlopok száma 7000. A mátrix elemeinek legnagyobb része természetesen nulla. Ezzel a tipikusan strukturálatlan adatokat, illetve azok jellemzőit struktúrába rendeztük.

Ezután a felügyelt gépi tanulás módszertana szerint kiválasztjuk a tanuló- és a teszt-adatokat. Előbbibe mintegy 4200, utóbbiba mintegy 1400 sms-t sorolunk. Ilyenkor célszerű ellenőrizni, hogy a tanuló- és a tesztadatokban a célosztály (spam) aránya ugyanaz-e, mint az eredeti korpuszban. Ha ugyanis lényeges eltérést tapasztalunk, akkor a tanuló- és a tesztadatok szétválasztásával torzítottuk a statisztikákat. Ez olyankor fordulhat elő például, ha az eredeti korpuszban már rendezett adatok szerepelnek, mert például egy rendezett adatbázisból töltöttük le, ahol a célosztály címkéje (spam) volt az egyik rendező szempont.

Ilyen esetekben szétválasztás előtt az adathalmazt randomizálni szükséges. A tanuló adatbázisból készítünk egy szófelhőt:

24. ábra. Szófelhő a tanulóadatokból
(Forrás: Saját szerkesztés.)

Vizsgáljuk most meg, hogy hogyan különbözik a ham és a spam osztály szófelhője (25. ábra):

25. ábra. A bal oldali ábra a spam, a jobb oldali a ham osztály szófelhője (Forrás: Saját szerkesztés.)

Látjuk a gyakori szavak közötti különbségeket, ennek alapján a modell betanítható a spam és a ham elkülönítésére.

A tanuló- és a teszt-DTM-ek (Document Term Matrix) is ritka mátrixok, ezért célszerű kizárni a ritkán előforduló szavakat, ezzel csökkentjük a mátrix méretét is. A minimális előfordulási küszöböt 5 szóban határoztuk meg, így a tanuló-DTM mérete mintegy 4000 x 1200-ra, a teszt-DTM mérete 1300 x 1200-ra csökkent.

A mátrixot átalakítjuk oly módon, hogy ha az x elem értéke > 0, akkor az érték 1, amennyiben nulla, akkor a mátrixban marad a nulla. Ezután már csak azt vizsgáljuk, hogy egy szó szerepel-e egy sms-ben vagy sem, ezek az osztályozó változók. A tanuló adathalmazon betanítjuk a Bayes-osztályozót, amely ezután el tudja dönteni egy sms-ről, hogy spam-e vagy nem.

A teszt adatbázison a valós pozitív találatok, vagyis a helyesen kiszűrt spamek aránya TP = 0,97, a false pozitívok aránya FP = 0,03, a valós negatívok, azaz helyesen a ham kategóriába sorolt sms-ek aránya 0,98, a helytelenül besoroltaké 0,02.

Az osztályozó modell teljesítménye tovább növelhető, ha az osztályozásnál nagy súllyal vesszük figyelembe azokat a szavakat, amelyek soha nem fordulnak elő a ham vagy a spam kategóriában.

A naiv Bayes több osztály esetén is használható a csoportosításra. Ekkor feltételezik, hogy a paraméterek normális eloszlású változók minden osztályban. A modell betanítása után kiszámítjuk az egy osztályba tartozó értékek várható értékét és szórását, és az új megfigyeléseket a meglévő eloszlásokkal jellemezzük. Így az osztályba sorolás valószínűségi alapon történik. A naive Bayes-eljárás igen egyszerűen implementálható R nyelven, Pythonban vagy számos más, gépi tanulásra alkalmas számítógépes környezetben.

Regressziós eljárások

A regressziós modelleket általában prediktív feladatokra alkalmazzuk, amikor az f függvény az X vektor változót egy numerikus (egész vagy valós értékű) Y output változóra képezi le.

Tekintsünk egy példát: egy város egy adott kerületében eladtak 100 lakást. Azt szeretnénk megbecsülni, hogy mennyi egy újonnan a piacra kerülő lakás ára, azaz egy számszerű adatot akarunk megbecsülni.

Ehhez összegyűjtjük a már eladott lakások adatait (méret, tájolás, építés éve, korszerűsítés éve stb.), ezek egy része kategória, más része numerikus változó.

A regressziós modell paramétereinek a felügyelt gépi tanulással történő kiszámítása után meg kell vizsgálni a modell teljesítményét. Erre a legáltalánosabb módszer az RMSE (root mean squared error).

Példa:
a betanított regressziós modellel öt becslést készítettünk a tesztadatok felhasználásával:

1,2; 1,3; 1,1; 1,2; 1,4
és a tényleges érték 1,2, akkor

$$RMSE = \sqrt{\frac{(1{,}2-1{,}2)^2+(1{,}2-1{,}3)^2+(1{,}2-1{,}1)^2+(1{,}2-1{,}2)^2+(1{,}4-1{,}2)^2}{5}} = \sqrt{0{,}012} = 0{,}11.$$

Az RMSE előnye, hogy a hiba mértékegysége azonos a regresszióban számított értékkel.

Megjegyezzük, hogy a logisztikus regresszió nem prediktív, hanem klasszifikációs modell, mint azt már az előzőekben láttuk. A klasszifikáció lényegében egy diszkrét osztálycímke előrejelzése, míg a regresszió egy numerikus változó előrejelzése.

A klasszifikáció és a regresszió azonban nem teljesen diszjunkt eljárások, egy klasszifikációs algoritmus egy osztálycímke valószínűségét is megadhatja, de egy regressziós algoritmus is képes egész számok vagy kategóriaváltozók előrejelzésére.

A klasszifikációs és az előrejelző algoritmusok értékelése azonban különböző módszerekkel történik, ld. a 11. sz. fejezetet.

5. A FELÜGYELET NÉLKÜLI TANULÁS – KLASZTERANALÍZIS

A felügyelet nélküli tanulás

A felügyelet nélküli tanulást olyan esetekben alkalmazzuk, amikor nincs előfeltételezésünk az adatok osztályszerkezetéről, vagy egy előrejelzés során a magyarázó változók súlyáról és egymás közötti függőségeiről. A felügyelet nélküli tanulás tipikus példája a klaszteranalízis, melynek célja az adatok rejtett struktúráinak feltárása, az objektumok csoportszerkezetének feltárása a tulajdonságokat leíró adatok alapján. Olyan adatelemző technika, amely segít megérteni az adatok által meghatározott, de közvetlenül nem látható csoportstruktúrákat, klasztereket. Az objektumokat többdimenziós adatvektorokkal írjuk le, és az adatvektorok hasonlósága alapján keressük az egy csoportba tartozó objektumokat. Szemben a döntési fákkal, a neurális hálókkal, az SVM-mel vagy más, pl. a Random Forest algoritmussal történő osztályozással szemben, itt nincs előzetes feltételezésünk sem arról, hogy hány csoportot szeretnénk képezni (bár erre vonatkozó technikai megkötések lehetségesek), sem arra, hogy egy objektum melyik osztályba fog tartozni. Tehát az adatok szerkezetét szeretnénk feltárni és nem egy osztályozó algoritmus létrehozása a cél.

A klaszteranalízis algoritmusai nem újak, mondhatjuk, hogy több évtizede léteznek, újszerűvé akkor válik a feladat, amikor vagy nagyon nagy mennyiségű adatot kell osztályoznunk, vagy az adatok strukturálatlanok, vagy az osztályozást valós időben, az éppen akkor beáramló adatokon kell végeznünk, még mielőtt az adatok mind megérkeztek volna.

A fejezet további részében áttekintjük a klaszteranalízis alkalmazásait, mind a hagyományos, mind a big data eszközeit. A klaszterezést a gyakorlatban számos feladat megoldására alkalmazzák. Fontos megjegyezni, hogy a felügyelet nélküli tanulásnál az eljárás hatékonyságát és pontosságát nem lehet a 11. fejezetben leírt mértékekkel kifejezni, mert egy felügyelet nélküli klaszterezésnél létrejövő osztályszerkezetet nincs mihez hasonlítani. Így legtöbbször az értékelésnél az elemzők szakmai tapasztalatukra hagyatkoznak. Természetesen ez esetben nincs értelme a tanuló- és tesztadatok megkülönböztetésének, hiszen, ha a tanulóadatokon létrehozott klasztereket alkalmazzuk a tesztadatokra, nem tudjuk eldönteni, hogy valójában hova kellene tartozniuk.

Egyes alkalmazásoknál azért preferálják a felügyelet nélküli klaszterelemzést, mert a vizsgált objektumok címkézése csak manuálisan valósítható meg (pl. beszédfelismerés esetén), és ez nagy objektumhalmaz esetén költséges. Ezekben az esetekben jöhet szóba a félig felügyelt tanulás, ahol a felügyelet nélkül létrehozott klasztereket össze lehet hasonlítani a manuálisan felcímkézett osztályokkal, és ez esetben kvantitatív mérésekkel vizsgálhatjuk az algoritmus hatékonyságát.

A klaszteranalízis alkalmazási területei

Marketing

A vásárlók és ügyfelek csoportosítása és profilozása, ami a célzott marketing akcióknak az alapja. Az ügyfelek klaszterezése és az ennek alapján történő profilozás segíti a marketingeseket abban, hogy üzeneteiket hatékonyabban juttassák el a potenciális vásárlókhoz, ill. információt szolgáltatnak arra vonatkozóan, hogy termékeiket hogyan fejlesszék. Fontos azt is tudni, hogy mely csoportok mely marketing kommunikációs csatornákat használják leggyakrabban (mobil eszközök, asztali PC-k, televízió, nyomtatott sajtó).

Fontos terület a termékek csoportosítása az ajánlórendszereknél, valamely szempontból hasonló termékek csoportjainak kijelölése. A Google vagy a Facebook a célzott hirdetési szolgáltatásaihoz a hirdető által meghatározott tulajdonsággal rendelkező és profilú csoportnak jeleníti meg a hirdetéseket, az Amazon és sok más kereskedelmi vállalat adott ügyfélnek az ajánlásokat az ügyfélcsoportra jellemző tulajdonságok alapján készíti el.

A klaszterezés fontos eszköze a szövegbányászatnak is, része a szövegelemzés eszköztárának. Segít abban, hogy a szövegből jó minőségű információt nyerjünk ki oly módon, hogy „kibányássszuk” a szövegben lévő mintákat, trendeket. A szövegbányászat a nyers input szöveg strukturálásával, a szintaktikai elemzéssel, egyes nyelvi sajátosságok keresésével, mások kizárásával és a kapott eredmények struktúrába szerkesztésével kezdődők, majd az elemzéssel és értékeléssel folytatódik. A szövegbányászatban a jó minőség a relevancia, újdonságérték és az érdekesség kombinációja. A tipikus szövegbányászati eljárások a szövegek kategorizálása, szövegek klaszterezése, koncepció/entitás kivonatolása, taxonómia létrehozása, véleményelemzés, dokumentumok kivonatolása, az entitások közötti kapcsolatok modellezése. Az adatbányászat célja rejtett szövegmintákban lévő információ kinyerése nagy szöveg-adat gyűjteményekből (ld. 9. fejezet).

Értékelési modellek

A klaszteranalízis hasznos eszköze az értékelési modellek kialakításának és javításának is. Akár emberek teljesítményének méréséről, akár hitelképességének vizsgálatáról, akár tanulók osztályozásáról van szó, egy adott csoportba tartozás hasznos prediktív adat lehet egy értékelő modellben. Hasznos lehet továbbá a csoporttagok közötti interakció feltárása, vagy az, ha például klaszterenként eltérő értékelési modellt alkalmaznak.

Tekintsünk egy példát. Az iskolai oktatásban a tanulók képesség szerinti csoportosítása segít a tanításban és a tanulásban, a jó verbális készséggel rendelkező tanulók csoportja más megközelítést igényel, mint például a művészi beállítottsággal rendelkező tanulóké. Sok iskola teljesítményét nem csak azzal mérik, hogy a végzett hallgatók mennyire sikeresek, hanem azzal is, hogy az iskolába való belépés és az iskola befejezése közötti időben mennyit fejlődtek.

A klaszteranalízis sajátosságai

- Nem feltételezi az adatok statisztikai eloszlásának ismeretét.
- Nem alkalmazhatók a szokásos teljesítménymérési eszközök, nincsenek olyan egzakt mérőszámok, amelyek objektíven megmutatnák, hogy egy csoportosítás mennyire sikeres. Ezt az elemzőnek a szakterületre jellemző hasonló elemzések tapasztalatára építve kell eldöntenie (ld. az előzőekben a félig felügyelt tanulásról írottakat).
- A klaszterek igen jól, intuitív módon szemléltethetők, a grafikus ábrázolás jelentősen segíti a megértést.
- A klaszterezési algoritmusok érzékenyek a szélsőséges értékekre és a kezdeti, az elemző által megadott kiinduló pontokra (ld. később).
- Egy adathalmazt többféle módon is lehet klaszterezni, és az eredmények eltérők lehetnek, függenek az algoritmus paramétereinek megválasztásától.
- A nem numerikus (kategorikus vagy ordinális) változókon történő klaszterezés előfeldolgozást igényel.
- Az eredmény függ attól, hogy az elemző hogyan határozza meg az objektumok hasonlóságát.

A gyakorlatban a klaszterezési eljárások két nagy csoportra oszthatók, a hierarchikus és a nemhierarchikus eljárásokra. A hierarchikus klaszterezés során „fákat” alakítunk ki. Ezt vagy felülről lefelé haladva a csoportok felosztásával (felosztó módszer), vagy alulról felfelé haladva a csoportok egyesítésével végezzük (összevonó módszer) egészen addig, ameddig el nem érjük a kívánt klaszterszámot (Tóthné, 2011).

Az agglomeratív hierarchikus módszerben, ha N a csoportosítandó elemek száma, akkor induláskor minden elemet egy önálló klaszternek tekintünk, majd a klaszterek számát 1-gyel csökkentjük oly módon, hogy a leginkább hasonló két elemet egy klaszterbe vonjuk össze. Az eljárást addig folytatjuk, amíg nem érjük el a kívánt klaszterszámot.

Az elosztó módszerek ezzel ellentétes irányban működnek, kezdetben az összes objektumot egyetlen klaszternek tekintjük, majd ezt a megadott távolsági kritériumok alapján addig osztjuk, ameddig meg nem kapjuk a várt klaszterszámot, vagy a felosztás nem tesz eleget egy más, előre megadott kritériumnak.

A nemhierarchikus klaszterezési technikák az objektumokat olyan K csoportra osztják, amelyek nem tartalmaznak közös elemeket, úgy, hogy a csoportokon belüli távolságok összegét minimalizálják. A K értékét vagy előre definiáljuk, vagy azt az algoritmus maga határozza meg.

A nemhierarchikus klaszterezés jóval nagyobb adathalmazok kezelésére alkalmas, mint a hierarchikus.

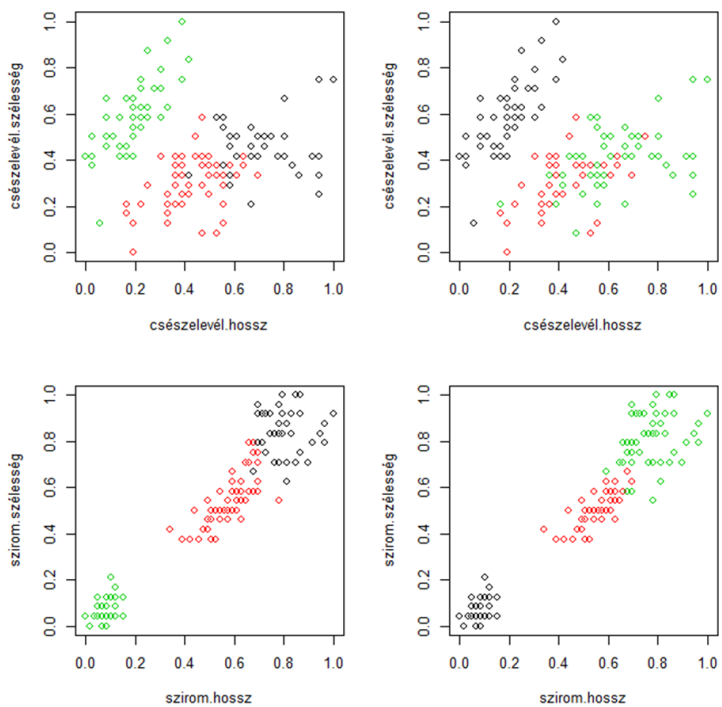
A leggyakrabban alkalmazott nemhierarchikus módszer a K-közép algoritmus. Ez először K kezdeti klaszterhez rendeli az összes objektumot. Ezután minden iterációban kiszámítja az objektumok távolságát a klaszter középpontjától. Az objektumot ezután áthelyezi abba a klaszterbe, amelynek középpontjától mért távolsága a legkisebb. Ha ez az a klaszter, amelyhez az előző iterációban tartozott, akkor nem változik a besorolása. Az iteráció addig folytatódik, amíg van lehetőség az átsorolásra.

A K-közép algoritmus nagyon egyszerű és általában gyors, hiányossága, hogy „gömb alakú” klasztereket feltételez, amelyek úgy szeparálhatók, hogy egy klaszterben lévő pontok konvergálnak a klaszter középpontjához.

A klaszterezési algoritmusok gyakorlati alkalmazása nem mindig egyszerű, mert bár az algoritmusok nem bonyolultak, a kapott eredmények elfogadhatóságának elemzése nem egyszerű. A kapott megoldást csak akkor lehet elfogadni és alkalmazni, ha az alkalmazási terület logikája szerint ez indokolt.

A klaszterezésnél az mtcars adatbázisban található gépkocsik adatait használtuk. Az adathalmazban 32 gépkocsi 11 paramétere található (fogyasztás, hengerek száma stb.). A dendogramot bárhol elvágjuk egy vízszintes vonallal, klasztereket kapunk. Minél magasabban vágjuk el a dendogramot, annál kisebb számú, aggregált klasztert kapunk. Ha az 1-es egyenes mentén vágunk, akkor kettő, ha a 2-es mentén, akkor négy klasztert kapunk. A program R nyelven készült, és az euklideszi távolságot használtuk.

A K-közép-módszer alkalmazására nézzük a referenciaként gyakran használt írisz virág adatokat, ahol a csészelevél szélessége (sepal.Width) és hosszúsága, valamint a szirom szélessége és hosszúsága alapján szokás az adatokban reprezentált háromféle fajtát (virginica, setosa, versicolor) csoportosítani.



26. ábra. A K-közép klaszterek. A kaszterezés 4 paraméter alapján történt. A baloldali ábrák a klaszter besorolást, a jobboldaliak az eredeti (ismert, de az algoritmusban nem használt) fajtacsoportokat mutatják
(Forrás: Saját szerkesztés.)

Az objektumok távolságának meghatározása

A legtöbb klasszikus klaszterezési eljárás az objektumok közötti hasonlóság/távolság meghatározásával kezdődik. A hasonlósági értékeket egy mátrixban lehet tárolni:

$$D = \begin{pmatrix} 0 & \cdots & d_{1n} \\ \vdots & \ddots & \vdots \\ d_{n1} & \cdots & 0 \end{pmatrix}$$

27. ábra. Az n objektum hasonlósági mátrixa. Minden oszlop és sor egy objektumnak felel meg. Az i-edik oszlop és a j-edik sor metszete az i-edik és a j-edik objektum távolsága (Forrás: Saját szerkesztés.)

A távolság mértékének meghatározása függ a konkrét alkalmazási területtől és a változók adattípusától. Az alábbiakban bemutatunk néhány távolságmeghatározást.

Euklideszi távolság

Az euklideszi távolság a geometriában alkalmazott távolság. Vegyünk két objektumot $\boldsymbol{x} = (x_1, x_2, \dots x_n)$ és $\boldsymbol{y} = (y_1, y_2, \dots y_n)$ vektorváltozókkal. Euklideszi távolságuk:

$$d(Euklidesz) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}.$$

Koszinusz távolság

Továbbra is használva az előző jelöléseket:

$$d(koszinusz) = \frac{\boldsymbol{x} * \boldsymbol{y}}{\|\boldsymbol{x}\|.\|\boldsymbol{y}\|},$$

ahol x és y a két objektum koordináta vektora, $\boldsymbol{x} * \boldsymbol{y}$ a két vektor skalár szorzata, $\|\boldsymbol{x}\|$ és $\|\boldsymbol{y}\|$ a vektorok hossza. Ez lényegében a két változó közötti standard korreláció.

Jacquard-hasonlóság

A Jacquard-távolság a klaszteranalízisben általában bináris változók távolságának meghatározására szolgál. A bináris 1-es azt jelenti, hogy egy tulajdonság jelen van, míg a nulla azt, hogy nincs jelen. Például ha egy személy tulajdonságai között szerepel, hogy tud-e angolul, akkor 1-gyel jelöljük, hogy tud, 0-val, hogy nem. Ha két személy közül mindkettő tud angolul, akkor hasonlítanak, ha egyikük tud, míg a másik nem, akkor különböznek, ha viszont egyikük sem tud angolul, ez nem jelent hasonlóságot.

A Jacquard-hasonlóság ezt a viszonyt írja le:

$$d(Jacquard) = \frac{\sum_{i=1}^{n} x_i\, y_i}{n},$$

ahol az x_i és az y_i értékei nullák vagy egyesek.

Manhattan-távolság

A Manhattan-távolság a nevét onnan kapta, hogy ekkora távolságot kell megtennünk New York Manhattan negyedében, ahol az utcák egymásra merőlegesek, vagy párhuzamosak ahhoz, hogy adott pontból egy másikba érjünk úgy, hogy csak az utcákon közlekedünk.

$$d(Manhattan) = \sum_{i=1}^{n} |x_i - y_i|$$

Chebysev-távolság

A Chebyshev-távolság a paraméterek távolságának maximuma. A koordináták közötti távolságok abszolút értékének maximuma

$$d(Chebyshev) = \max_k |x_k - y_k|.$$

Minkowsky-távolság

A Minkowski-távolság ún. általánosított távolságmérték. Abban különbözik az euklideszi távolságtól, hogy a kitevő nem feltétlenül 2, hanem bármely pozitív egész lehet. Ha , akkor megkapjuk a Chebyshev-távolságot. Mind ordinális, mind kvantitatív változók távolságának mérésére megfelelő.

$$d(Minkowsky) = \left(\sum_{i=1}^{n} |x_k - y_k|^{\frac{1}{p}} \right)^p.$$

A távolságmetrikák sora ezzel nem ér véget, a fenti mértékek számos további variánsa is létezik. Ha az objektumok minden jellemzője numerikus, akkor a legáltalánosabban alkalmazott mérték az euklideszi távolság vagy a koszinusz hasonlósági mérték. Ha ordinális adataink vannak, akkor a mérték választása a konkrét feladattól függ. Nem mondhatjuk például, hogy egy olimpiai első és harmadik hely között kétszer akkora a távolság, mint egy harmadik és negyedik hely között, így itt az euklideszi távolság vagy a koszinusz távolság nem jó mérték.

A klaszteranalízis, bár segítségével konkrét feladatokban sok hasznos információt tudunk kinyerni az adatokból, nem egzakt módszer.

Felsorolunk néhány technikai problémát:

- Az alkalmazott hasonlósági mértékek legtöbbször az objektumok távolságán alapulnak. A távolságalapú mértékek azonban nem mindig alkalmasak az objektumok korrelációjának kimutatására. Valószínű, hogy az adathalmazok között léteznek hasonló minták, akkor is, ha a távolság a választott metrika szerint nagy.
- A nemhierarchikus eljárásoknál előzetesen meg kell adni egyes paramétereket, mint például a klaszterek számát. Ez statikus változó, a programban nem változtatható.
- Ha a klaszterezett objektumok előzetesen rendelkeznek osztálybesorolással (osztálycímkével), akkor a felügyelt és a felügyelet nélküli osztályozás eltérő eredményeket adhat.
- A klaszteranalízis algoritmusai nem tudják eldönteni, hogy egy változó mennyire releváns az elemzés szempontjából, az eljárás nem tesz különbséget a változók között, ezért a klaszterezéshez használt változókat az elemzőnek kell kiválasztania a vizsgált terület szakmai szempontjai alapján. Ha ez nem történik meg, akkor a módszer félrevezető eredményekhez vezethet.

A fentiekben leírt klaszterezési eljárásoknál világosan kell látni, hogy egyik sem valós idejű, hanem batch eljárások. Az osztályozást csak akkor lehet elkezdeni, ha az ehhez szükséges minden adatot összegyűjtöttünk, megtisztítottunk, és – ha szükséges – struktúrába rendeztük és betöltöttük a memóriába. Az algoritmusok ezután többször is használják az adathalmazt.

Számos esetben, például egyes nyelvi értelmezési feladatoknál, fordításnál, szövegek lényegének kiemelésénél, képek, képrészletek csoportosításánál, akusztikus jelek értelmezésénél és egyes, a gyorsan változó piacra vonatkozó üzleti döntéseknél viszont szükség lehet a valós idejű klaszterezésre, vagyis a csoportosítást a beérkező adatfolyamon kell végezni, dinamikusan változó klaszterekkel.

A valós idejű klaszterezés módszere viszonylag új, algoritmusai különböznek a hagyományostól, hiszen a futáskor még nem ismerünk minden adatot és legtöbbször egy adatot csak egyszer lehet elolvasni, a korábbiak törlődnek.

Valós idejű klaszterezési eljárások

A valós idejű klaszterezésnél tipikus, hogy az adatok nagy mennyiségben adatfolyamként érkeznek, ami nem teszi lehetővé a hagyományos batch feldolgozó algoritmusok alkalmazását. Az algoritmus egyenként dolgozza fel a beérkező mintákat és azonnal elemzi azokat, korlátozott memóriaterületet használ, gyorsan szolgáltat eredményt, és bármely pillanatban értékelni kell tudnia a beérkezett adatokat s ezek alapján modellt kell felállítani. Az algoritmus lényegében tanuló (stream learning) eljárás, folyamatosan tökéletesíti magát a beérkező adatok alapján.

Stream learning algoritmusok léteznek klasszifikációs, regressziós, klaszterező és mintakereső feladatok megoldására.

Az első nyílt forráskódú program, amely nagy mennyiségű adatfolyam feldolgozására, illetve a hasonló programok fejlesztésére hoztak létre, a MOA (Massive On-line Analysis). Alkalmas többosztályos klasszifikációra, részgráf minták bányászatára, klaszterezésre (Bifet et al., 2018).

Csak az adatfolyam klaszterezést mutatjuk be, a MOA ehhez az alábbi, az elemző által választható algoritmusokat tartalmazza.

StreamKM++: Az adatfolyam kisebb, súlyozott részére alkalmazza a K-közép algoritmus egy változatát, randomizált klaszter kezdőértékekre. A kis mintával végzett számításokhoz a coreset közelítést alkalmazza (Agarwal et al., 2005).

CluStream: Az adatokból kinyert statisztikai információt mikroklaszterekben tárolja. A mikroklaszterek lényegében a klaszterek tulajdonságait tárolják pillanatfelvételszerűen. Bármely pillanatban vissza lehet keresni a korábbi mikroklasztereket és a folyamat végén elő lehet állítani a teljes folyamra vonatkozó makroklasztereket a mikroklaszterek összevonásával.

ClusTree: Paraméter nélküli algoritmus, amely automatikusan képes adaptálódni az adatfolyam sebességének változásához, észleli a konceptuális változásokat, a kiugró értékeket.

További algoritmusok: *DenStream, D-Stream, CobWeb.*

Az adatfolyamok az alábbi forrásokból származhatnak: SQL-adatbázisok, Hadoop, Storm, Hive, .csv fájlok, egyszerű fájlok stb. A MOA-eljárások 2014 óta R-ből is elérhetőek.

Az adatfolyam többdimenziós adatvektorok rendezett sorozata:

$$X = (\bar{X}_1, \bar{X}_2, \ldots \bar{X}_n, \ldots).$$

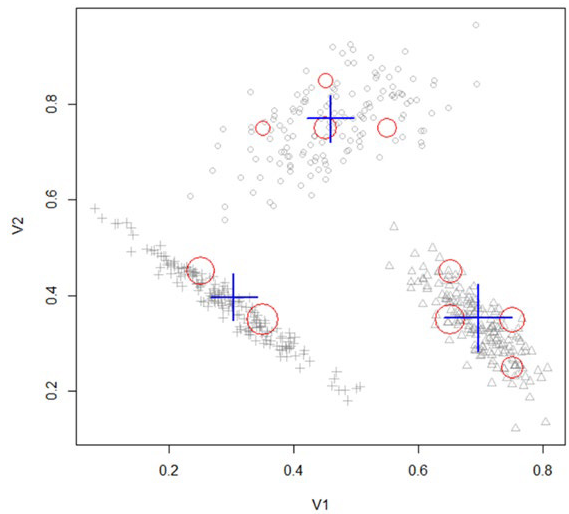
A rendezést az indexelés vagy időbélyegzés[44] jelenti.

Az egyes adatvektorok nominális, ordinális vagy komplexebb adatokat (pl. gráfokat vagy strukturálatlan adatokat) is tartalmazhatnak. A sorozat elvileg nem véges, bár a gyakorlatban nem találkozhatunk végtelen sorozatokkal.

A 28. ábrán mikro- és makroklasztereket látunk egy adott időpillanatban. A piros körök az időben változó mikroklaszterek középpontjai egy adott időpillanatban, a kék keresztek

44 Az időbélyegzés az adat keletkezésének időpontja.

ugyanebben az időpillanatban a mikroklaszterek integrációjából származó klaszterek középpontjai. Az R-program futása során a mikroklaszterek száma és helyzete folyamatosan változhat, a makroklaszterek száma 3-ban rögzített. A kék keresztek helyzete is változhat.



28. ábra. Mikro- és makroklaszterek
(Forrás: Saját szerkesztés.)

6. A NEURÁLIS HÁLÓK ÉS ALKALMAZÁSUK

A neurális hálók

A neurális hálók felügyelt gépi tanulási algoritmusok. Nem abban különböznek más gépi tanulási algoritmusoktól, hogy más jellegű feladatot oldanak meg, hanem abban, hogy másképpen és nagyon sok esetben hatékonyabban, nagyobb pontossággal oldják meg a feladatokat. A neurális hálók algoritmusai igen jól teljesítenek például a képek vagy arcok felismerésénél (ez lényegében osztályozási feladat), kézírás felismerésénél (ez is osztályozási feladatra vezethető vissza), projektek költségeinek becslésénél, értékbecsléseknél (ez tipikusan előrejelzés feladat), bűnözési minták leírásánál (ez szintén osztályozási feladat), stratégiai játékoknál (ez előrejelzési és osztályozási feladat).

A neurális hálókkal végzett elemzések az alábbi típusokba sorolhatók:

- objektumok bináris osztályozása,
- objektumok kettőnél több csoportba sorolása,
- előrejelzések, becslések.

Ezek az eljárások számos gyakorlati feladat megoldására alkalmasak, mint például:

- pénzügyi idősorok előrejelzése,
- hitelbírálat,
- képfelismerés, képek kategorizálása,
- kézírás-felismerés,
- adócsalás felderítése,
- eltérő értékek kiválasztása audit esetén,
- aláírás, értékpapírok verifikálása,
- gazdasági események előrejelzése,
- csődelőrejelzés,
- vállalati alkalmazottak teljesítményének előrejelzése,
- vállalati összeolvadások, felvásárlások előrejelzése,
- országkockázat előrejelzése,
- vásárlói viselkedés előrejelzése,
- értékesítési prognózis,
- célzott marketing,
- szállítási útvonal megtervezése,
- munkafolyamat tervezése.

Közigazgatási alkalmazások

A neurális hálókkal történő osztályozási és előrejelzési alkalmazások a közszolgáltatások színvonalának emelésében is szerepet játszanak. Az alkalmazások közül kiemelhetők az alábbiak.

Közlekedés

Mesterséges neurális hálózatot alkalmaztak Granada (Garido et al., 2014) város buszközlekedésénél az utazóközönség a szolgáltatásminőséggel való elégedettségének elemzésére. Azt vizsgálták, hogy a szolgáltatás egyes paraméterei mennyire befolyásolják az elégedettség szintjét. Az értékeléshez egy előzetesen lefolytatott kérdőíves elégedettségi felmérés adatait használták. Az elemzés célja azoknak a paramétercsoportoknak a beazonosítása volt, amelyek lényegesen befolyásolják a felhasználói elégedettséget. A kapott eredmény azt mutatta, hogy a felhasználói elégedettséget legnagyobb mértékben a járatok gyakorisága határozza meg, de más paraméterek, mint a sebesség és a megállók közelsége is befolyással van a minőségre.

A közösségi közlekedéssel való elégedettség mérésére számos más kutatási projekt is kidolgozott különféle módszereket. Az Ojo-cikk (2017) összesen 85 megjelent publikációt sorol fel ebben a témában.

Okos otthon

Az okos otthonok működtetéséhez okos szoftver megoldásokra van szükség. A Teich és munkatársai (2013) cikkükben személyre szabott szolgáltatáshoz kidolgozott szoftvert mutatnak be a szerzők.

A rendszer adaptív, az épület sajátosságai, a lakók profilja és a külső tényezők figyelembevételével alkalmazkodik a lakók igényeihez. A rendszer támogatja például a lakás energiahatékony módon történő fűtését. A rendszer folyamatosan tanul, a visszajelzésekből újratanítja a gépi tanulómodelleket, és azonnali visszacsatolást is lehetővé tesz.

Neurális hálókat használnak az időseket támogató okosotthon-alkalmazásoknál, az egészségügyben vagy a katasztrófaelhárításban, lakókörzetek energiaszükségletének megtervezéséhez alkalmazott döntéstámogató rendszerekben, gyártási folyamatok energiaszükségletének minimalizálásában és számos más területen.

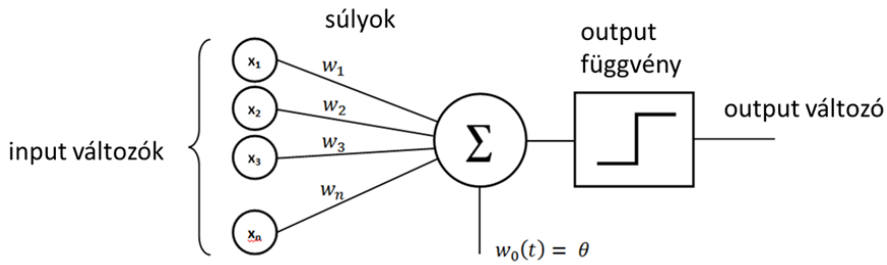
A neurális hálók felépítése

A neurális hálóknak nevezett gépi tanulómodelleket az emberi idegrendszer működésével való bizonyos hasonlóság miatt nevezik neurális hálóknak. A neurális hálók példákból tanulnak, nem használnak feladatspecifikus modelleket. Az algoritmusokat példákon keresztül tanítják be a feladatok megoldására. Az eljárás gondolata az 1940-es években merült fel először, amikor McCulloch et al. (1943) leírták egy neurális háló számítási modelljét. A számítás algoritmusát küszöb-logikának nevezték. A felépített modell neuronnak nevezett számítási egységekből állt, amelyek kapcsolatban voltak a többi neuronnal, és szignálokat bocsátottak ki, ha a bemenő szignálok elértek egy adott küszöbértéket. Ebben az időben

még nem léteztek elektronikus számítógépek, amelyek ezzel a módszerrel gyakorlati számításokat tudtak volna végezni. Az elméleti modell célja az idegműködés szimulációja volt.

A mai értelemben vett neurális hálók létrejöttét a legtöbben Rosenblatt perceptronának megjelenésével azonosítják (Rosenblatt, 1958).

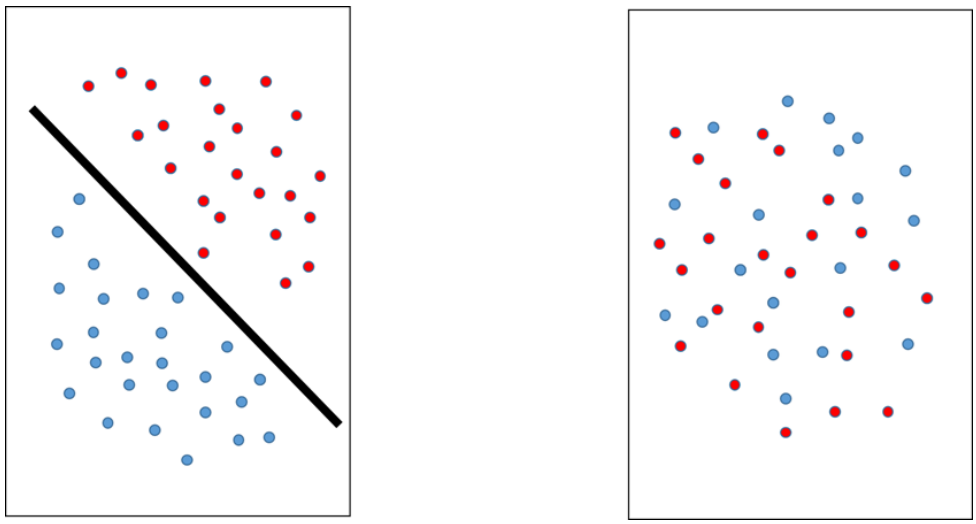
A Rosenblatt-féle perceptront a 29. ábrán mutatjuk be:



29. ábra. A Rosenblatt-féle perceptron
(Forrás: Saját szerkesztés.)

A Rosenblatt-modellben a neuron inputját több numerikus változó alkotja, ezek értékei a külvilágból származnak. A neuronban kiszámoljuk az input értékek a (w_1, w_2, ..., w_n) súlyvektorral súlyozott összegét és hozzáadunk egy (t) = θ értéket. Az így kapott skalárt az előre meghatározott, rögzített output függvény (aktivációs függvény)[45] átalakítja egy output változóvá. Ezután megvizsgáljuk, hogy az output függvény értéke mennyire van közel az elvárt outputhoz, például egy bináris klasszifikáció esetén a perceptron milyen arányban osztályozza helyesen az objektumokat. Ha az eredménnyel elégedetlenek vagyunk, akkor a w_i súlyok változtatásával próbáljuk javítani a kimenő függvény osztályozó képességét, és ezt az eljárást addig folytatjuk, ameddig az osztályozás nem lesz elfogadható minőségű.

45 Az aktivációs függvény megnevezés abból ered, hogy megfelelő bemenő szignálok esetén a neuron kimenő szignálokat aktivál.

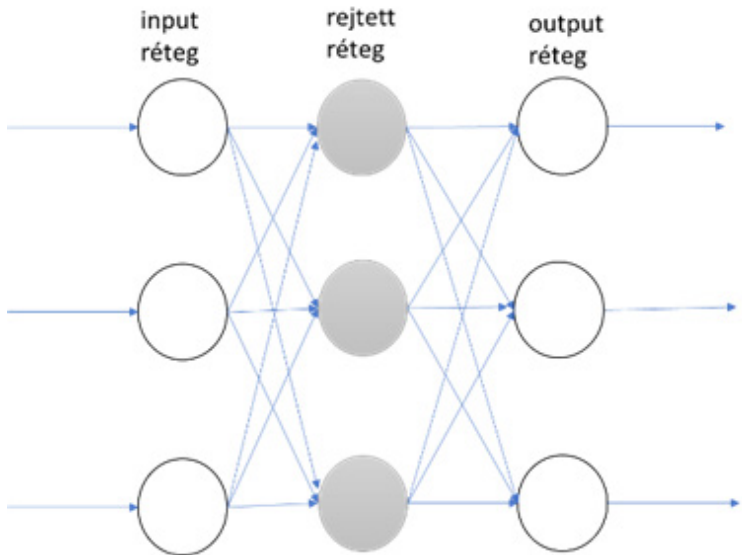


30. ábra. Az (a) – lineárisan szeparálható,
(b) – lineárisan nem szeparálható objektumok halmaza
(Forrás: Saját szerkesztés.)

A neurális hálókkal történő gépi tanulás elméletében és gyakorlatában komoly előrelépést jelentett, amikor Rosenblatt fenti cikkében bizonyította, hogy az egyszerű perceptron fentiekben leírt, ún. előrecsatolásos tanulási folyamat meghatározott tanuló algoritmusok esetén konvergál. A tétel csak a lineárisan szeparálható, azaz síkkal szétválasztható objektumhalmaz esetén igaz. Viszont egy egyszerű perceptronnal ennél többet nem is tudunk elérni (30. [a] ábra).

Rosenblatt perceptronának megjelenését követően a fejlesztések a több, egymással összekapcsolt perceptronból álló hálók létrehozására és kutatására irányultak, de hosszabb idő telt el, amíg a több perceptronból álló hálók alkalmazásával a lineárisan szeparálható halmazok szeparálási problémájának megoldásán kívül egyéb feladatok is megoldhatóvá váltak, ugyanis hosszú ideig nem találtak megfelelő eljárást a több, összekapcsolt perceptron súlyvektorainak megfelelő változtatására, az egy perceptronnál működő előrecsatolásos módszer ui. itt nem használható.

A neurális hálókban a számítási egységeket – neuronokat – rétegekbe rendezik. Az első rétegben lévő neuronok a környezetből származó adatokat kapnak (ezek a gépi tanulási algoritmusnál a tanulóadatok), míg az utolsó rétegben lévő neuronok az eredményadatokat közlik a külvilággal. Az első – input – réteg és az utolsó – output – réteg között több, ún. rejtett réteg is lehetséges, a modell bonyolultságának megfelelően. A rejtett rétegeknek nincs közvetlen kapcsolatuk a külvilággal, csak belső adatokat dolgoznak fel és továbbítanak (31. ábra).



31. ábra. Neurális háló egy rejtett réteggel
(Forrás: Saját szerkesztés.)

Az algoritmus minden réteg minden egyes neuronjának bemenő adataihoz súlyokat rendel és iterációk során keresztül ezeket úgy változtatja, hogy a kimenő adatokból képzett, a modellezett folyamatot vagy állapotot leíró mérték megfelelő legyen. Például a legkisebb négyzetes eltérés ne haladjon meg egy meghatározott szintet.

Többrétegű neurális hálók súlyainak optimalizálására találták ki 1975-ben az ún. visszaterjesztéses folyamatot (angolul: backpropagation), ami azt jelenti, hogy a háló végén számított hibát visszaterjesztik az előző rétegekre, és az előző rétegeken minimalizálják a hibát. A visszaterjesztéses módszerrel működő modellt 1986-ban építettek (Rumelhart et al., 1986).

A visszaterjesztéses módszer

A visszaterjesztéses módszer egymásba ágyazott függvények optimalizálása oly módon, hogy egy tanulási menet után a kimeneti értékek és az elvárt értékek közötti hibát a megelőző rétegekre „visszaterjesztjük”[46]. A visszaterjesztésnél az első lépés a kimeneti rétegen mért hiba deriválása az utolsó réteg súlyai szerint, majd a deriváltak további deriválása a megelőző rétegekre az adott réteg súlyai szerint. Így az összetett függvények deriválási szabálya szerint deriváltak láncolatát kapjuk egészen az input rétegig, és visszafelé haladva tudjuk optimalizálni az egyes rétegek hibáit. Ez az algoritmus része számos neurális hálót tervező és létrehozó programnak (pl. a Tensorflow-nak), mindezt nem szükséges az alapoktól programozni.

46 A visszaterjesztéses módszer matematikai alapjainak megértéséhez szükséges az optimalizálásnál alkalmazott gradiens módszer ismerete. A fejezet további része enélkül is érthető.

Tegyük fel, hogy az input vektorok alapján kiszámítottuk a súlyokat. Az összesített hibát jelöljük *E*-vel:

$$E = \frac{1}{2}\sum_{c}\sum_{j}(y_{j,c} - d_{j,c}).$$

A hibát az output paraméterekre (j) index és az esetekre (c-case) összegeztük. A hiba minimalizálásához a gradiens módszert alkalmazva ki kell számolni az *E* minden súly szerinti parciális deriváltját. Egy súly szerinti parciális derivált az összes input-output eset súly szerinti parciális deriváltjainak összege. Egy adott esetre kiszámítjuk a hiba súlyok szerinti parciális deriváltjait. A pontos matematikai leírást az idézett Rumelhart et al. cikkben olvashatjuk.

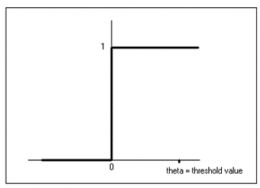
A hiba visszaterjesztéses módszer sikeres alkalmazása lehetővé tette, hogy a neurális háló alkalmassá vált komplex feladatok ellátására is, olyanokéra, amelyek megoldására az eredeti perceptron algoritmus elméletileg sem volt alkalmas, még a többrétegű neurális hálók esetén sem. A visszaterjesztéses módszer és a neurális hálók azóta történt számos fejlesztése azt eredményezte, hogy a neurális hálók igen széles körben alkalmazott, hatékony gépi tanuló algoritmusokká fejlődtek.

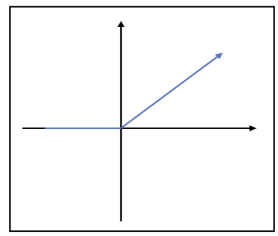
Az egyre intelligensebb robotok elterjedése, a természetes nyelvi fordítás, a beszédfelismerés, a képfelismerés, az arcfelismerés legnagyobb részben neurális hálókon alapszanak.

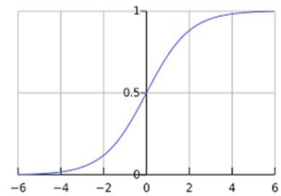
A továbbiakban azokat a fogalmakat, összefüggéseket és modelleket ismertetjük, amelyek már az elemzők mindennapi munkaeszközévé váltak, az előremutató, de a gyakorlatban még nem használt tudományos eredményeket itt nem tárgyaljuk.

Az aktivációs függvények

Az egyes neuronok kimenetét az aktivációs függvény határozza meg a bemenő adatok és a súlyok alapján. Az aktivációs függvények általában egyszerűek, 0-1 értékűek, lineárisak vagy szigmoid függvények (ld. **.** ábra).







(a) küszöbfüggvény függvény

$$f(x) = \begin{cases} 0 \; ha \; x \leq 0 \\ 1 \; ha \; x > 0 \end{cases}$$

(b) szigmoid

$$f(x) = 1 \; / \; (1 + e^{-x})$$

(c) rectified linear unit (RELU)

$$f(x) = \max(0, x)$$

32. ábra. Tipikus aktivációs (output) függvények (Forrás: Saját szerkesztés.)

Ha egy neuron több bemeneti adatot kap és több kimeneti adatot állít elő, akkor az aktivációs függvény egy vektort állít elő. A leggyakrabban használt ilyen függvény a softmax:

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=0}^{k} e^{x_j}}.$$

Ez lényegében a szigmoid függvény általánosítása többváltozós esetre.

Egyáltalán miért van szükség az aktivációs függvényekre? Amennyiben a neuronok csak a bemenő szignálok súlyozott összegét továbbítanák, akkor csak lineáris függvényeket lehetne a hálóval modellezni. A cél azonban az, hogy a neurális háló tetszőlegesen bonyolult függvényt le tudjon írni. Matematikailag bizonyítható, hogy az egy rejtett réteggel rendelkező neurális hálók egy szigmoid függvény segítségével tetszőleges pontossággal tudnak közelíteni folytonos függvényeket, feltéve, hogy nem szabjuk meg a neuronok számát és a súlyok értékét. Elméletileg tehát a neurális hálók modellezési képessége nem korlátozott, az alkalmazásokban azonban több gyakorlati problémát is meg kell oldani. A visszaterjesztéses módszernél fontos, hogy az aktivációs függvények differenciálhatók legyenek, ehhez a gradiens módszer alkalmazásához van szükség.

Elméleti jelentőségük ellenére a szigmoid aktivációs függvények nem mindig alkalmasak a visszaterjesztéses optimalizálásra. A legfőbb probléma az eltűnő gradiensek jelensége, amikor a gradiens értéke egy iterációban olyan közel van a nullához, hogy a súlyok értéke nem változik, mert a súlyok módosításának mértéke a gradiensek értékével arányos és a tanulás ezért idő előtt leáll. Ezen a problémán segít a RELU aktivációs függvény. A szigmoid függvények másik problémája a lassú konvergencia, azaz a szigmoid aktivációs függvények mellett a hálózat lassan tanul, nagy tanuló adatbázis szükséges a megfelelő hatékonyság eléréséhez.

A szigmoid függvény helyett szokták a hiperbolikus tangens[47] függvényt alkalmazni, amely alakjában hasonlít a szigmoid függvényre, de értéke a (-1, 1) intervallumban változik, és a függvény értéktartománya nagyobb, mint a szigmoid függvényé, a gradiens kevésbé „tűnik el".

A tapasztalatok szerint a leghatékonyabb a legegyszerűbb megoldás a RELU-függvény. Az eltűnő gradiens itt is problémát okozhat, ez ellen úgy védekeznek, hogy a negatív változó tartományban a konkrét értéket randomizálják. A RELU viszont csak a rejtett rétegekben alkalmazható, az input és output rétegben nem, mert ha mindenütt lineáris függvényt használunk, akkor a neurális háló nem lesz képes a nemlineáris osztályozásra vagy becslésre. A nem rejtett rétegekben ilyenkor a softmax alkalmazása a legáltalánosabb módszer.

A veszteségfüggvény (loss function)

A neurális háló által előállított adatok és a céladatok közötti különbséget kifejező függvényt nevezzük veszteségfüggvénynek, vagy költségfüggvénynek. A veszteségfüggvény nagymértékben függ attól, hogy milyen feladatról van szó, és attól, hogy milyen a neurális hálózat struktúrája. Egy regressziós feladatnál a legáltalánosabb veszteségfüggvény a négyzetes hiba, a klasszifikációs feladatnál a keresztentrópia. A négyzetes hiba a regressziós becslés és a becsült függvény távolságát méri, a keresztentrópia azt fejezi ki, hogy ha az objektumokat a modellel soroljuk osztályokba, mennyire „tiszták" ezek az osztályok, azaz mekkora a nem a megfelelő helyre sorolt objektumok aránya.

$$SE(\hat{y}) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2,$$

A négyzetes hiba (squared error) számítása:

ahol *y* a tényleges, y a neurális hálóval kapott becsült érték. A neurális hálóval képzett regressziós becslés esetén a négyzetes hiba a leggyakrabban használt veszteségfüggvény.

A neurális hálókkal végzett osztályozási feladatok esetén a cél az, hogy a modell az objektumokat egymást kölcsönösen kizáró osztályokba sorolja. A modell akkor tekinthető megfelelőnek, ha az objektumokat nagy valószínűséggel sorolja abba az osztályba, amelybe tartoznak. Például, ha kézzel írott számjegyeket kell osztályoznunk, akkor megfelelőnek értékeljük a modellt, ha minden számjegyet nagy valószínűséggel a saját osztályába sorol, az 1-et az egyesek osztályába, a 2-t a kettesek osztályába stb. Minthogy a modell tanításához felhasznált objektumok címkézettek, azaz tudjuk, hogy egy adott objektum ténylegesen hova tartozik, azt is tudjuk, hogy ezt az objektumot a modell nagy valószínűséggel sorolta-e a megfelelő osztályba. A keresztentrópia segítségével mérhetjük a modell megfelelőségét. Egy példán szemléltetjük a keresztentrópia kiszámítását.

Tudjuk, hogy $a \in A$; azaz az *a* objektum az *A* osztályba tartozik.

Ekkor

$$P(a \in A) = 1, \qquad P(a \in B) = 0, \qquad P(a \in C) = 0, \quad P(a \in D) = 0.$$

Ha a modell által szolgáltatott valószínűségek az alábbiak:

$$Pm(a \in A) = 0{,}6 \;\; Pm(a \in B) = 0{,}2 \;\; Pm(a \in C) = 0{,}1 \quad Pm(a \in D) = 0{,}1,$$

47 $\tanh(x) = \frac{e^{2x}-1}{e^{2x}+1}$

akkor a keresztentrópia:

$$H(p,q) = -\sum_{x} p(x) * logq(x),$$

ahol *p(x)* a valódi, ismert valószínűség, *q(x)* a modellben számított valószínűség. Esetünkben H(p,q)=-1xln(0,6). A 0,51 érték a tényleges és a képzett osztályozás távolságát méri. Pm(a∈A)=1 esetén a távolság nulla lenne, és a modell tökéletesen mutatná, hogy az *a* objektum az *A* osztályba tartozik.

A neurális hálókkal történő modellezés

A fentiekben ismertettük a neurális hálók alapmodelljét, amely egy bementi és egy kimeneti réteggel és nulla vagy több rejtett réteggel rendelkezik, a veszteségfüggvény optimalizálása pedig kétrétegű háló esetén közvetlen súlyválasztással, rejtett rétegek jelenléte esetén a visszaterjesztéses módszerrel történik, ahol a hibát visszavezetjük a rejtett rétegeken keresztül és a gradiens módszerrel (vagy a számítási volumen csökkentése érdekében mintavételezéssel, a sztochasztikus gradiens módszerrel) minimalizáljuk.

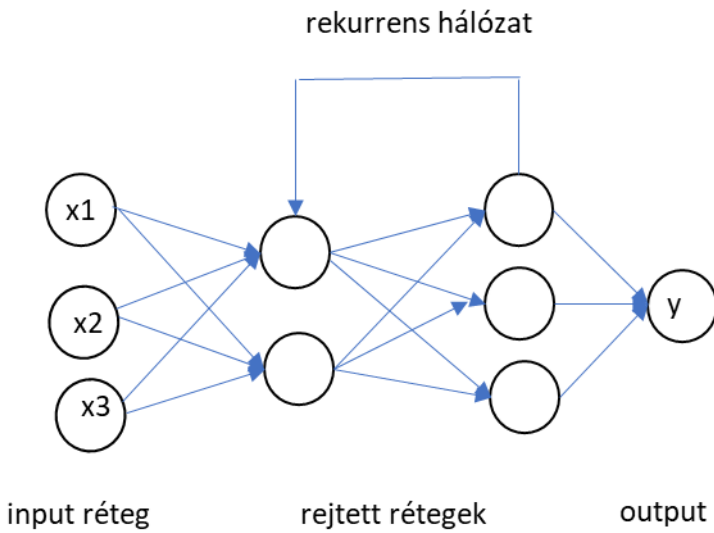
Az alapmodell azonban a gyakorlati osztályozási és regressziós modellek jelentős részére közvetlenül nem alkalmazható. Az egyik tipikus példa, amikor az alapmodell nem működik, a képfelismerési feladat, amely nem más, mint a kép tartalma vagy más tulajdonságai alapján történő osztályozás. A képen szereplő személy vagy tárgy jellemzőinek felismerése ui. osztályba sorolási feladat.

A digitális képábrázolásban a képet általában képpontok, pixelek halmazaként tároljuk és dolgozzuk fel. Egy jó minőségű kép esetén azonban a pixelek száma igen nagy lehet, akár több millió. Minthogy egy pixel a neurális háló bemeneti rétegén egy neuronnak felel meg, a feldolgozáshoz hatalmas méretű hálóra és sok millió súlyparaméterre lenne szükség az neurális hálózati alapmodellben, ami számítási szempontból kezelhetetlen. A képfeldolgozási feladatokhoz fejlesztették ki az ún. konvolúciós modellt (ld. később), amelyet azután más területeken, például hangfelismerésben is sikerrel alkalmaztak.

Az alapmodell – amelyben nincsenek hurkok, nincs visszacsatolás – problémája az állapotmentesség. Az állapotmentesség azt jelenti, hogy a háló betanításának bármely pontján a korábbi információ nem érhető el, mert azt nem tároljuk, a súlyok pillanatnyi értéke csak integráltan tartalmazza az összes korábbi információt. Ezzel a módszerrel viszont nem lehet például sorozatmintákat észlelni, vagy osztályozni, vagy nem lehet akusztikus szignálokból a folytatásra következtetni. Más természetű példaként tekintsünk egy részvénypiaci trendet. Ha szeretnénk előre jelezni a részvényárfolyamok alakulását, akkor nem csak a részvények aktuális árfolyamát, hanem az előző időszak árfolyamalakulását is figyelembe kell venni.

Rekurrens hálók

Amennyiben a háló tartalmazhat visszacsatolást, vagyis a háló visszacsatolja a kimeneteit a bemenetekre, rekurrens hálóról beszélünk (33. ábra). Ezáltal a rendszer dinamikussá válik, az aktivációs szintek stabil állapotot vagy ciklikus viselkedést is mutathatnak, és a kaotikus, azaz hosszabb távon megjósolhatatlan viselkedés sem kizárt.



33. ábra. Rekurrens háló. Az egyes rétegek „visszafelé" is kommunikálnak, nem csak az előző rétegből kapott adatokat dolgozzák fel
(Forrás: Saját szerkesztés.)

A rekurrens háló esetén a háló egy bemenetre adott válasza nem csak a pillanatnyi állapottól és a bemenettől függ, hanem a kezdeti állapotoktól is, a háló „emlékszik" a korábbi bemenetekre. Ezt a képességet rövid távú memóriának tekintjük, mert a háló „emlékszik" arra, hogy az előző tanulási ciklusban hogyan döntött, de nem emlékszik a teljes kontextusra.

Az emlékezés azonban nem korlátlan, lássunk egy példát a rekurrens háló korlátaira.

Azt a mondatot, hogy a

„A fű színe ..."

egy rekurrens háló egyszerűen ki tudja egészíteni:

„A fű színe zöld."

amennyiben az előző mondatban megadtuk, hogy a növények általában zöldek.

De ha az alábbi szöveget adjuk meg:

„István az Egyetemen angol szakot végzett, majd évekig Ázsiában dolgozott"
„István jól beszél ... "

Egy ember tudja, hogy István jól beszél angolul. Egy rekurrens hálózat azonban erre a kérdés feltevésekor már nem emlékszik, mert egyéb információ jött közbe.

Matematikai nyelven megfogalmazva, ez az eltűnő gradiens problémája. Az alapmodellben a súlyok aktualizálása a tanulási sebességnek, az előző réteg hibaértékének és a réteg inputjának szorzata. Így a hiba értéke az aktuális rétegen az előző rétegek hibáinak szorzatától függ. A szigmoid és a hasonló aktivációs függvények kis értékű deriváltjai, amelyek beépülnek a hibafüggvénybe, összeszorzódnak a visszaterjesztés során, ahogyan a kezdeti rétegek felé haladunk. Így a gradiensek szinte eltűnnek (nullához közelítenek), és így a régebbi adatok súlya egyre kisebb. A rekurrens hálóknál hasonló a helyzet, a háló csak rövid ideig emlékszik egy adatra, mert, ha a tanulásban sok szót adunk meg a hálónak, a korábbi adat súlya egyre csökken.

A rekurrens hálók rövid távú memóriával kapcsolatos problémáinak kiküszöbölésére fejlesztették ki az ún. hosszú-rövid távú memóriával rendelkező hálókat[48] (Greff et al., 2016).

Az LSTM-hálók

A rekurrens neurális hálózatok egyforma súllyal kezelnek minden adatot. Míg egy ember képes súlyozni a beérkező információt, például képes eldönteni, hogy egy már megbeszélt találkozó fontosabb-e, mint egy már lekötött másik esemény, és az új adat felülírja-e a régit, a rekurrens neurális háló minden új információ megjelenésekor az összes addigi információt átírja. Nincs fontos vagy kevésbé fontos. Az LSTM ezzel szemben csak kisebb módosításokat hajt végre az információn (szorzásokkal és hozzáadásokkal). Az LSTM-hálózatban az információ egy cellaállapotok nevű mechanizmuson halad át, ami lehetővé teszi azt, hogy az LSTM szelektíven jegyezzen meg, vagy felejtsen el, adatokat. Egy adott cellaállapotban három különböző függőséget különböztetünk meg, amit egy példán szemléltetünk.

Szeretnénk megjósolni egy részvény árfolyamát. Az árfolyam az alábbi tényezőktől függ:

- az előző napi trend, ami emelkedő, stagnáló vagy süllyedő,
- az előző napi árfolyam,
- a mai árfolyamot befolyásoló tényezők, mint például a vállalati jelentések, hírek, profit előrejelzés, céges vezetőváltások.

Ezek a tényezők az alábbiak szerint csoportosíthatók:

- az előző cellaállapot, azaz az előző időpont után a memóriában jelenlévő információ,
- az előző rejtett állapot, azaz az előző cella outputja,
- az input az aktuális időpontban, azaz az aktuális időpontban meglévő új információ.

Az LSTM-et gyakran hasonlítják egy szállítószalaghoz. A szállítószalag a termékeket szállítja a különböző munkafázisokhoz, ugyanezt végzi az LSTM az információval. Az egyes munkafázisok során újabb információt adhatunk a hálózatba, módosíthatunk vagy törölhetünk információt, ahogyan a szállítószalag keresztülviszi azokat a különböző rétegeken.

48 Angolul: Long Short-Term Memory Networks, LSTM.

A nem rekurrens neurális hálók esetén a működést az alábbi matematikai jelöléssel írhatjuk le kompakt módon egyetlen rejtett réteg esetén.

Legyen

$$h_j = f\left(\sum_{i=1}^{n} x_i * w_{i,j}\right),$$

ahol h_j a rejtett réteg *j*-edik neuronjának kimenő értéke, x_i $i = 1,...$ a rejtett neuron bemeneti értékei, $w_{i,j}$ a *j* -edik neuron *i*-edik bemenetének súlya. Feltételezzük, hogy az egyik súly felelős az eltolásért, *f* a rejtett neuron kimeneti függvénye.

Legyen a *j*-edik output neuron kimenő értéke

$$y_j = \left(\sum_{i=1}^{n} h_i * v_{i,j}\right).$$

A modell minden újabb mintaelem feldolgozása után a tanulás során módosítja a súlyokat, javítja a pontosságot, de ugyanakkor a korábbi információegységekhez rendelt súlyokat elfelejti.

Az LSTM-algoritmusok (és általában a rekurrens háló) ezeket a súlyokat megjegyzik és hasznosítják a későbbiekben. Ehhez arra van szükség, hogy a rejtett rétegek ne legyenek állapotmentesek, azaz ne felejtsenek. Az időt jelöljük *t*-vel, ekkor a és az alábbiak szerint módosulnak:

$$h_t = f(W * x_t + U h_{t-1})$$

$$y_t = V * h_t$$

A *t-1* időpontban eltárolt információ visszatér a *t*-edik időpontban. Az *U* a rejtett rétegek közötti átviteli mátrix (transition matrix), ami meghatározza, hogy mely korábbi adatok mennyire fontosak.

A folyamat során a sok információ összegyűlik, egymás súlyát csökkenti, és a modell egy idő után kaotikussá válik. Az LSTM-eljárást ennek a problémának a megoldására találták ki. Az LSTM az alábbi plusz tulajdonságokkal rendelkezik:

1. Képes felejteni. Egy bejövő új információnál el kell tudni döntenie, hogy mely régebbi információk relevánsak, melyeket kell megőrizni és melyeket lehet törölni. A modellnek meg kell tanulnia a felejtés és az információ megjegyzésének algoritmusát.
2. Képes tárolni a releváns információt.
3. Tudja, hogy a tárolt információt mikor kell felhasználni. Ez a fókuszálás.

Amíg az RNN szabályok alkalmazása nélkül írja át az „emlékeit", az LSTM ezt teljesen szabályozott módon teszi, és meg tudja különböztetni, hogy mely információk relevánsak és melyek nem azok, így sokkal több hasznos adatot tud hosszabb ideig tárolni.

A fenti tulajdonságokat az LSTM kisebb neurális hálókkal tanulja meg. A pontos működést itt nem tárgyaljuk, mert meghaladja a könyv kereteit, de egy közérthető leírást találunk itt.[49]

Az LSTM működését egy példán szemléltetjük. A feladat az LSTM algoritmus alkalmazása egy chatrobot betanítására a Facebook által létrehozott bAbI-adatokon.[50] Az adathalmaz szintetikusan előállított valós világbeli történetekből áll.

A letölthető adatok a Facebook bAbI-projektjéhez tartozó 20 feladat első adathalmazát tartalmazzák. Az adatok szövegek értelmezésére és magyarázatára kialakított tanuló és tesztállományok. A projekt célja a kérdés–válasz (QA) rendszerek kutatása, a különböző tanuló algoritmusok hatékonyságának vizsgálata. Az adatokat a Facebook szabad felhasználásra tette közzé a chatrobotok fejlesztésének támogatására. A feladat fontosságát támasztják alá az olyan rendszerek elterjedése, mint a Siri, Cortana, Google voice assistant, Alexa, a kép–szöveg, a szöveg–kép és videó–szöveg fordítóprogramok, amelyek mind a számítógéppel természetes nyelven történő kommunikáció eszközei.

A QA-rendszerek régóta kutatott területnek számítanak, részterületei az adatfeldolgozás, információkeresés, egyes fókuszpontok kinyerése a szövegből. Jelenleg a QA-rendszereket chatbotok fejlesztésére és az emberi beszéd szimulációjára használják.

Hagyományosan a QA nyelvészeti alapon nyugvó NLP- (Natural Language Processing) technikákat, részesítették előnyben, mint a nyelvi elemzés, beszédrészek címkézése, koreferenciák kezelése. Ezt használja pl. az IBM Watson-rendszere.

A jelenleg használatos RNN-, LSTM-rendszerek azonban hatékonyabbak, hosszabb szövegeket tudnak kezelni, mint a természetes nyelvi technikák. A memóriával rendelkező hálók lehetővé teszi a leginkább releváns tényekre történő fókuszálást.

A hasonló adathalmazok két típusúak, nyíltak és zártak. A nyíltak a megadott adathalmazon kívül bármilyen adatokat használhatnak a világból a válasz kialakítására, míg a zártak csak a megadott adathalmazt.

Az Allen AI Science[51] például nyílt, míg a bAbI zárt QA-adathalmaz. A nyílt adathalmazok természetesen tágabb kutatási lehetőséget biztosítanak, de sok olyan nyelvi technológia alkalmazását teszik szükségessé, amelyek megnehezítik a feladatot. Ezért használjuk a bAbI-adatokat. A bAbI 20 feladatának mindegyike számos tartalom–kérdés–válasz hármast tartalmaz, minden feladat egy-egy értelmező szempont tesztelésére használható, és a modell tulajdonságainak validálására is alkalmas.

Az alkalmazott szótár és a mondatszerkezetek nagyon leegyszerűsítettek. Az egyszerűsítés megkönnyíti a kutatást. A generált történet mellett a szöveg mutatókat tartalmaz a releváns tényekre (supporting facts), azaz azokra a mondatokra, amelyek szükségesek a kérdések megválaszolásához. Így a gépi tanulásnak mind az erősen felügyelt, mind a gyengén felügyelt változatát használhatjuk. Az erősen felügyelt változatnál figyelembe vesszük a

49 http://colah.github.io/posts/2015-08-Understanding-LSTMs/
50 https://research.fb.com/downloads/babi/
51 https://www.kaggle.com/c/the-allen-ai-science-challenge/data

supporting facts mutatókat, míg a gyengén felügyelt tanulásnál csak azt, hogy egy mondat a történet része, kérdés vagy válasz.

Az adatok angol és hindi nyelven érhetők el. Nyelvenként két részből állnak. Egy-egy részben 1000, egy feladatnál összesen 10 000 példa szerepel. Itt a kisebb, ezres angol nyelvű adatokat használjuk. A bAbI-hoz hasonló adatbázist hozott létre a Microsoft is, a neve MCTest. Ez is kontextust, kérdést és válaszokat tartalmaz. Az MCTest fiktív történetekből áll, amelyeket emberi munkával hoztak létre az Amazon Mechanical Turk[52] segítségével, a hétéves gyerekek olvasási szintjén. Az MCTest egy többválasztós kérdéses feladat. 660 történetet és 2000 választ tartalmaz. A kis adatbázis nem igazán alkalmas ilyen jellegű algoritmusok tanítására.

A továbbiakban használt bAbI-file tartalma:

ID szöveg
ID szöveg
ID szöveg
ID kérdés[tab]válasz[tab]supporting fact IDs.
...

Az egymás után következő mondatok egy sztorit alkotnak. Egy sztorihoz az ID-k 1-től kezdődnek és egyesével növekednek. Ha az ID értéke 1 a fájlban, akkor a következő mondatok egy új sztori kezdetét jelentik.

A supporting fact ID-k a sztorin belüli hivatkozások.

Példa[53]:
1 Mary moved to the bathroom.
2 John went to the hallway.
3 Where is Mary? bathroom 1
4 Daniel went back to the hallway.
5 Sandra moved to the garden.
6 Where is Daniel? hallway 4
7 John moved to the office.
8 Sandra journeyed to the bathroom.
9 Where is Daniel? hallway 4
10 Mary moved to the hallway.
11 Daniel travelled to the office.
12 Where is Daniel? office 11
13 John went back to the garden.
14 John moved to the bedroom.
15 Where is Sandra? bathroom 8

52 https://docs.aws.amazon.com/AWSMechTurk/latest/AWSMechanicalTurkRequester/IntroductionArticle.html
53 A példát eredeti angol nyelven közöljük a reprodukálhatóság érdekében.

1 Sandra travelled to the office.
2 Sandra went to the bathroom.
3 Where is Sandra? bathroom 2

Az LSTM alkalmazásával a bAbI-adatokon betanítjuk a modellt.

Az eredeti számítógépes program a githubon található.[54] A szakirodalom szerint a betanított modell 98,6%-os pontosságot ér el a single_supporting_fact_10k adatokon 120 futtatás után, azaz a betanított modell a kérdések 98,6%-ára helyes választ ad. Ezeket az eredményeket egy korszerű laptop számítógépen 10 perces futási időn belül reprodukálni tudjuk.

Az igen jó eredmények értékelésénél nem hagyhatjuk figyelmen kívül, hogy a feladatot kissé leegyszerűsítettük:

1. A bAbI-adatbázis zárt a QA-feladat szempontjából, azaz a program nem használ más adatokat.
2. A sztorikban, kérdésekben és válaszokban használt szavak egy korlátozott szókészletből származnak.
3. A mondatok szerkezete a természetes nyelvi mondatokhoz képest egyszerűsített.
4. A „supporting facts” manuális címkézés eredménye.

Az adatbázis és az eljárás sokkal inkább kutatási célokat szolgál, mint egy természetes nyelvi környezetben működő modell létrehozását. Az adatokkal jól ellenőrzött körülmények között tudunk kísérletezni, és ki tudjuk dolgozni a feladathoz és a környezethez leginkább illeszkedő modelleket. Az így megszerzett tapasztalatok alapján tudunk valós körülmények között működő modellt építeni.

54 https://github.com/rstudio/keras/blob/master/vignettes/examples/babi_memnn.R
(Jason Weston, Antoine Bordes, Sumit Chopra, Tomas Mikolov, Alexander M. Rush, “Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks”.)

7. KÉPEK ELEMZÉSE A KOMMUNIKÁCIÓBAN

A képek szerepe az információ közvetítésében

A vizuális szimbólumok egyre nagyobb szerepet játszanak a közszféra és az állampolgárok közötti kommunikációban, elsősorban a politikai üzenetek közvetítésében. A nyomtatott sajtó jelentőségének csökkenésével a televízió és az internet az információ elsődleges forrása, ezeken a médiumokon pedig a képi megjelenítés dominál. A képi kommunikáció és az egyes politikai vagy más célú üzenetekhez kapcsolódó vizuális megjelenítés a gépi tanulás tartalmának elemzése egyre népszerűbbé váló kutatási terület (Anastasopoulos et al., 2017).

A képek központi szerepet játszanak a közvélemény formálásában, például a közszereplők imázsának alakításában. Az idézett cikk is ezt a kérdést vizsgálja a neurális hálók alkalmazásával a Facebookról az USA kongresszusának és szenátusának tagjairól letöltött közel 300 ezer fénykép alapján.

A képek hatásának elemzésével számos tudományos, pszichológiai és szociológiai tanulmány is foglalkozik (Tingley et al., 2014).

A továbbiakban azt vizsgáljuk, hogy a gépi tanulási algoritmusok hogyan tudják megkülönböztetni egymástól a különböző szervezetek üzeneteit, hogyan lehet egy-egy tájékoztatási kampány képeit általános ismérvek lapján megkülönböztetni egymástól, vagyis osztályozni. Az eredmények alkalmazása hozzájárulhat egy-egy üzenet hatékony eljuttatásához a címzettekhez.

A 6. fejezetben bevezettük a neurális hálózatok elméletét, a továbbiakban a neurális hálók képfeldolgozásra történő alkalmazását vizsgáljuk részletesebben.

Képfeldolgozás konvolúciós hálóval

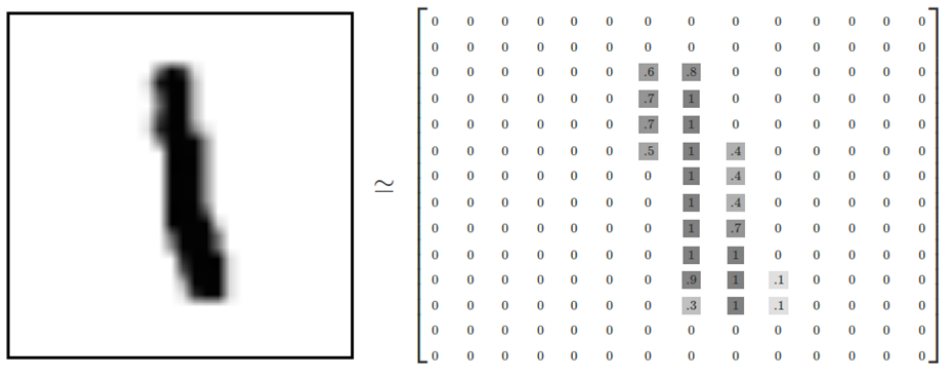
A digitális képek képpontokból, ún. pixelekből állnak.[55] Amennyiben a kép „fekete-fehér", vagy a szürke árnyalataiból áll, akkor egy-egy pixelt egy 0–255 közötti egész számmal jellemzünk, azaz minden pixelt egy bájttal írunk le. Ha színes képről van szó, akkor egy pixel megadásához általában három bájtot használunk, egyet a piros, egyet a zöld, egyet a kék szín intenzitásának jelelemzésére. Ez az RGB (Red, Green, Blue) színábrázolási módszer. A piros, zöld és kék szín kombinációja adja meg egy pixel színét a képernyőn.[56]

---

55 Az elemzéshez a vektorgrafikával készült képeket először raszteres képpé – pixelekké – alakítják.

56 Megjegyezzük, hogy a nyomtatásban más színábrázolást alkalmaznak, mert fehér papíron az RGB színekből nem lehet jó minőségű feketét nyomtatni.

A fenti ábrázolással egy fekete-fehér képet egy mátrixszal, egy színes képet három mátrixszal (vagy másképpen: egy háromdimenziós tömbbel) ábrázolunk.



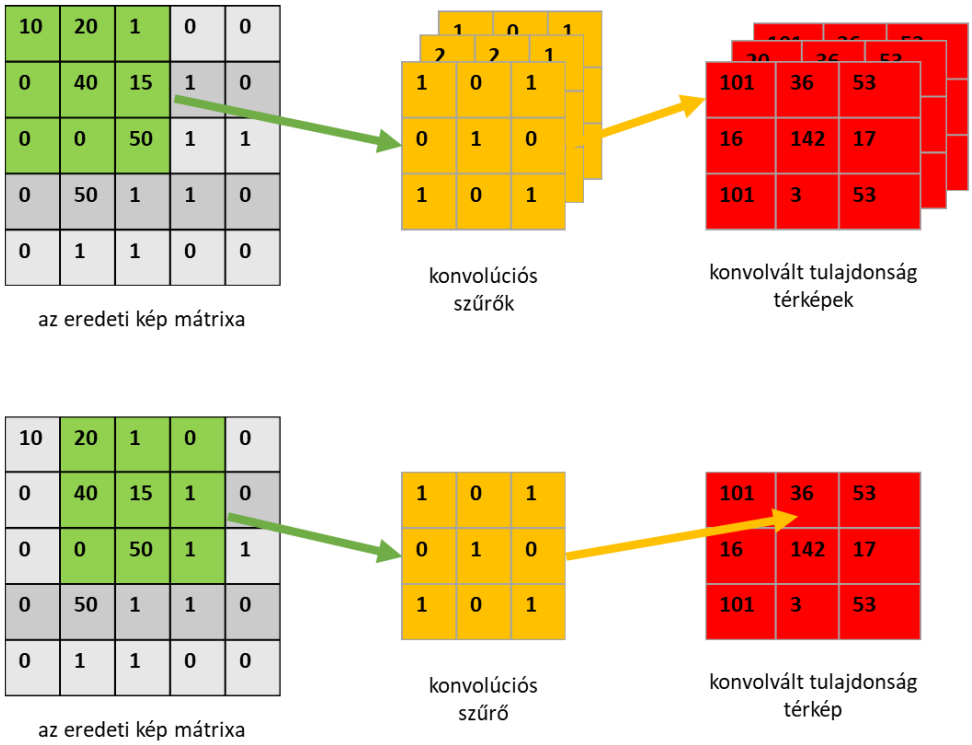
34. ábra. Egy 28 x 28 pixelen ábrázolt, szürke árnyalatos kézzel írt számjegy leképezése egy 28 x 28-as mátrixszá a nyilvánosan használható MNIST-adatbázisból[57]

Ha egy neurális hálót szeretnénk megtanítani a 34. ábrán látható objektumok osztályozására, vagyis kézzel írott számjegyek felismerésére, akkor az input rétegben 28 x 28 = 784 neuront kell alkalmaznunk. Ha nagyobb a kép mérete és a kép színes, akkor ez a szám sokkal nagyobb lehet, és a neurális hálók alapmodellje nagy, színes képek osztályozására közvetlenül nem használható.

A digitális képfelismerés méretproblémájának megoldására kifejlesztették a konvolúciós neurális hálókat, melyek kifejezetten erre a célra kialakított többrétegű (mély) hálók. Egy viszonylag kis méretű, 100 x 100 pixeles színes kép esetén egyetlen neuron inputján 100 x 100 x 3 = 30 000 súly jelenik meg, ami a kép méretének növekedésével skálázhatatlan módon nő. A konvolúciós hálók a képet kisebb, kezelhető részenként olvassák be és dolgozzák fel. Ehhez három réteget alkalmaznak:

1. Konvolúciós réteg, amely egy szűrősorozattal szkenneli a képet és egy ún. tulajdonságsíkra képezi le. A szűrő egy súlymátrix, amellyel a kép éppen beszkennelt részét súlyozzuk. A konvolúciós rétegben egy vagy több szűrő szerepelhet, amelyek más-más tulajdonságokat vonnak ki. Minden szűrő azonos számú neuront tartalmaz. Minden szűrőhöz a súlyokon kívül tartozik még egy eltolás (offszet) érték. Egy konvolúciós szűrő alkalmazásának eredménye egy ún. tulajdonságtérkép (feature map), több szűrő több tulajdonságtérképet állít elő (35. ábra).

57 https://tensorflow.rstudio.com/tensorflow/articles/tutorial_mnist_beginners.html



35. ábra. Fent több, lent egy konvolúciós szűrő működése
(Forrás: Sztenderd mátrixok.)

A gyakorlatban olyan szűrőket használnak, amelyek a képek legjellemzőbb tulajdonságait gyűjtik ki, mint az azonosság, kontúrok detektálása, kontrasztok (36. ábra)

36. ábra. A gyakorlatban használt néhány tipikus szűrő
(Forrás: Sztenderd mátrixok.)

A tulajdonságtérképeken a konvolúciós hálózati algoritmus alkalmaz egy kimeneti, általában a RELU-függvényt, így alakulnak ki az ún. aktivációs térképek.

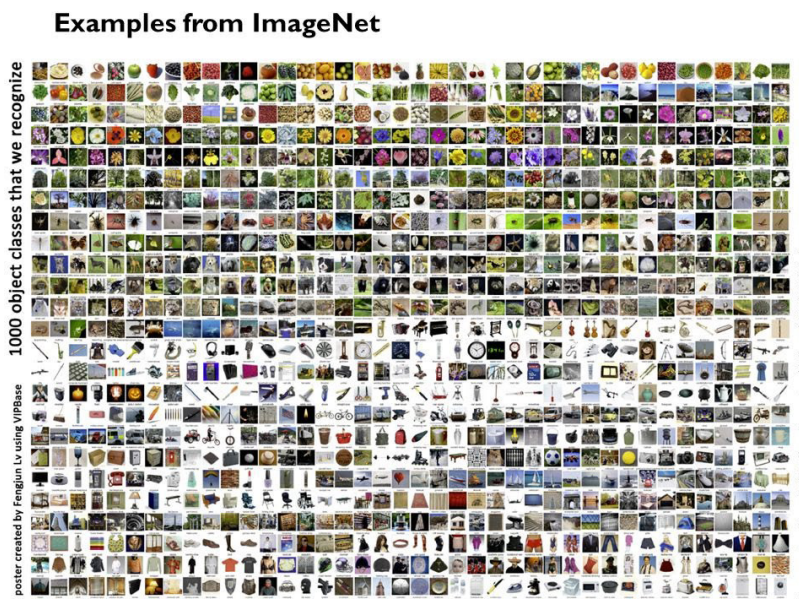
2. Mintavételi rétegek (pooling layers). A szűrés nem csökkenti lényegesen a méreteket, a méretek csökkentése céljából az aktivációs térképből egy kisebb méretű mintát vesznek. A mintavétel úgy történik, hogy a térkép megadott méretű négyzetes részmátrixaiból képeznek egy számot (maximumot, átlagot vagy összeget – max pooling, average pooling, sum).

A konvolúció és a mintavételezés egymást követően többször is történhet, a konvolúciós és a pooling rétegek számát a modell betanítása során kell meghatározni.

3. Teljesen összekötött réteg (fully connected, FC). A teljesen összekötött réteg hasonló a nem konvolúciós többrétegű neurális hálózatok struktúrájához, az utolsó pooling réteg minden neuronja összeköttetésben van az utolsó, kimenő réteg minden neuronjával. A bemeneti rétegének minden eleme össze van kötve az FC-réteg minden neuronjával, mint egy klasszikus többrétegű perceptronban. Tipikusan utolsó rétegként szokás használni a konvolúciós neurális hálóban. A konvolúciós rétegek a jellemzők kiemelését végzik, az FC-rétegek pedig pl. az osztályozást. A kimenő rétegben az aktivációs függvény vagy RELU, vagy softmax típusú.

A képfelismerési – osztályozási – algoritmusok fejlesztésének támogatására közösségi részvétellel létrehozták a 22 ezer kategóriát és jelenleg több, mint 14 millió képet tartalmazó weboldalt, a www.image-net.org-et. 2012-ben a közösség által meghirdetett verseny, az ImageNet Challenge forradalmi fejlődést hozott a képfelismerés területén.

A versenyt 2010 óta évente hirdették meg ImageNet Large Scale Visual Recognition Challenge (ILSVRC) néven. A versenyben a kutató teamek feladata minél nagyobb pontosság elérése többféle képfelismerési feladatban egy előre megadott adathalmazon. A versenyen használt adathalmaz 1000 kategóriát tartalmaz az eredeti 22 ezerből, például 90 kutyafajt. 2011-ben a hibás felismerés aránya 25% volt, 2012-ben egy mély konvolúciós hálózattal 16%-ot értek el, a következő években a tévedés aránya néhány százalékra esett.



37. ábra. Az 1000 objektum osztály az ILSVRC-versenyen (Forrás: https://www.google.com/search?q=ILSVRC&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiC9NeR3JvjAhVRlYsKHSNUB3AQ_AUIECgB&biw=1358&bih=710#imgrc=0PeS9V4U5Cn5PM)

A 2012-es áttörés nem új algoritmusoknak volt köszönhető, hanem a már meglévő eljárások együttes felhasználásának, ez jelentette a mesterséges intelligencia képfelismerésre történő használatának ipari méretű kezdetét.

2015-re a képfelismerő szoftver képességei a versenyen egy-egy szűkebb területen már meghaladták az emberi intelligenciát. Nem szabad ugyanakkor elfelejteni, hogy itt csak 1000 kategóriát kellett megkülönböztetni, az ember ennél sokkal több kategória felismerésére képes.[58]

2017-ben a 38 versenyző csapat közül 29 kevesebb, mint 5%-os hibaarányt ért el.

2017 novemberében a Google AutoML, új neurális hálózatok kutatására irányuló projektje létrehozta a NASNet hálózatot, amelyet részben az ImageNetre optimalizáltak. A Google szerint a NASNet teljesítménye meghalad minden korábbi, az ImageNettel kapcsolatos eredményt (Sulleyman, 2017).

A képfelismerés nem csak teljes képekre, hanem képek részleteire is kiterjed, azaz egy több személyt és tárgyat tartalmazó képről a személyek és a tárgyak egyenkénti felismerése is lehetséges.

A fejezet elején hivatkozott Anastasopoulos et al. (2017) cikkben a kérdés az volt, hogy az USA kongresszusi képviselőiről és szenátorairól letöltött 300 ezer kép elemzése alapján megállapíthatók-e különbségek a demokrata és a republikánus képviselők, ill. szenátorok képei között. Lényegében két vizsgálatot folytattak le, az egyik az illetővel a képen együtt szereplő más személyek faji hovatartozásának, a másik a képen szereplő tárgyak, lakberendezési elemek összehasonlítása. Például egyértelműen kimutatható volt, hogy egyes képviselők mellett több, míg mások mellett kevesebb afroamerikai személy volt a képeken, és hasonlóan, az otthonukban készült képeken lévő berendezési tárgyak is tematikusan különböztek.

Az interneten fényképek milliárdjai tölthetők le szabadon. Az ImageNeten vagy más referencia adatbázison betanított neurális hálózati algoritmusokkal számos kutatást lehet végezni, mint például:

- emberek csoportosítása nem, bőrszín vagy más tulajdonságok alapján,
- embereket körülvevő más emberek és így kapcsolati hálók keresése,
- személyek csoportosítása a képen szereplő helyszínek, tárgyak alapján,
- személyek azonosítása és ezt követő profilkészítés.

A képfeldolgozás fontos területe az egészségügy. A korszerű diagnosztikához elengedhetetlenek a különböző képalkotó eljárások. A létrehozott képek értékelése azonban sokszor nem egyszerű feladat a számos zavaró tényező miatt. De egészen biztosan mondhatjuk, hogy az egészségügyben használt képalkotó berendezések a nem távoli jövőben önálló, mesterséges intelligencia alapú komplex értékelési szoftverrel rendelkeznek majd, amelyek hozzájárulnak az orvosi döntések gyorsaságához és pontosságához.

58 https://www.newscientist.com/article/dn28206-forget-the-turing-test-there-are-better-ways-of-judging-ai/

8. A TERMÉSZETES NYELVI ELEMZÉS

A természetes nyelvi elemzés a közigazgatásban

A közszolgálat digitalizációjának természetes velejárója az állampolgárok és a hivatalok közötti kommunikáció gyors növekedése. A digitális csatornákon az állampolgárok könnyen megírhatják és elküldhetik kéréseiket, véleményüket, értékelésüket a közigazgatási szervezeteknek, hivataloknak. A kommunikáció intenzitásának fokozására irányulnak az Európai Unió egyes programjai is, például a közvetlen állampolgár–kormányzati kommunikációt ösztönző ImproveMyCity, amelyek szorgalmazzák az állampolgárok részvételét a helyi és az országos kormányzásban.[59]

Az ImproveMyCity, hasonlóan a közszféra–**állampolgár kommunikáció egyéb programjaihoz,** három pilléren nyugszik: a bejelentésen, az intézkedésen és az elemzésen. Az állampolgárok közvetlenül jelentik a helyi önkormányzatoknak a helyi problémákat, a bejelentések automatikusan átküldésre kerülnek a megfelelő hivatalba, ahol megtörténik az intézkedés. Az ezt követő elemzés során különböző vizualizációs technikák alkalmazásával segítik a munkatársakat abban, hogy megfelelő képet kapjanak a működésről és segítsék a vezetést a döntéshozatalban.

Az állampolgárok és a kormányzat közötti kommunikáció több csatornán folyik Magyarországon is, telefonon, interneten, papíralapú beadványok formájában stb.

A gyors és hatékony kommunikáció alapvető feltétele a nagyfokú automatizáció, hiszen, ha a digitális csatornákon folytatott kommunikáció minden lépéséhez emberi beavatkozás szükséges, akkor a folyamat lassú és a digitalizáció haszna korlátozott. Az automatizálás alapvető feltétele, hogy a természetes nyelven beérkező szövegeket ne kelljen emberi munkával értelmezni, mert ez lassúvá és költségessé teszi az ügyek intézését. Természetszerűleg adódik a természetes nyelvi elemző módszerek használatának igénye.

A közigazgatás egyik fontos feladata az állampolgári elégedettség mérése és nyomon követése. Az elégedettségről az állampolgárok közvetlen vagy közvetett visszajelzéseiből lehet képet kapni. A visszajelzések legnagyobb részben természetes nyelven írt közvetlen üzenetek, fórumokon kifejtett vélemények, észrevételek, közösségi hálón szövegek formájában jelennek meg. Így a természetes nyelvi elemző eszközök használatával lehet gyors és pontos visszajelzést kapni arról, hogy az állampolgárok mennyire elégedettek a közszolgáltatásokkal (Kowalski et al., 2017).

59 https://ec.europa.eu/futurium/en/egovernment4eu/direct-citizen-government-communication-and-collaboration

A természetes nyelvi feldolgozás gépi eszközei

A természetes nyelven írt szövegek számítógépes feldolgozása (Natural Language Processing – NLP) a gépi tanulás és a nyelvtudomány közös területe. A természetes nyelv gépi eszközökkel történő „megértése" bonyolult feladat, nem elég a szavak és mondatok jelentésének, hanem a szövegkörnyezet és a kommunikáló felek ismerete is fontos. Tudni kell, hogy a szavak és mondatok milyen környezetben, milyen feltételezett szándékkal hangzottak el, kitől és kinek szólt a közlés. Az elemzést megnehezítik az élő beszédben használt többértelmű szavak, a szleng és az emberi beszéd pontatlansága, amelyek nem befolyásolják azt, hogy a beszédet egy másik ember megértse, de lényeges problémát jelentenek a gépi szövegértésben.

A természetes nyelvi feldolgozás leggyakoribb alkalmazásai a számítógépen tárolt vagy továbbított, írott vagy beszélt szövegek megértése, a szövegben lévő értékes információ kinyerése és elemzése, a fordítás, a nyelvfelismerés, a szövegben történő intelligens keresés, valamint az ember–gép természetes nyelvi kapcsolatok fejlesztése (Chowdhury, 2003).

A természetes nyelvi feldolgozás építő elemei:

- tokenizálás, azaz elemekre (szavakra, mondatrészekre) bontás,
- a szöveg mondatokra bontása,
- a beszéd elemeinek megcímkézése. Nem csak szavakat, hanem több elemből álló rendezett beszédrészeket (ún. N-gramokat) is tekinthetünk nyelvi elemeknek,
- információ kinyerése, szöveg kivonatolása,
- tulajdonnévvel rendelkező nyelvi elem felismerése és kategorizálása (személynevek, földrajzi helyek, egyéb, azonosítóval ellátott objektumok stb.).

A tokenizálás, a mondatok particionálása, a címkézés, a szavak, mondatok, kifejezések beazonosítására, szerkezetük és határaik elemzésére szolgáló eljárások viszonylag egyszerűbbek, míg az információ kinyerése és a nevesített entitások felismerése bonyolultabb feladat (Mitkov, 2009), (Prakash et al., 2011).

A közigazgatási szférában jó példa az információ kivonatolására a járművekre, bűntényekre stb. vonatkozó adatok strukturálatlan forrásokból történő kibányászása.

Pinhero et al. (2010) cikke bűntények színhelyeit és típusát bányászta ki online szövegekből 72–87%-os pontossággal.[60] A cikkben leírt alkalmazásban a szövegekből az információ kinyerésére egyrészt előre meghatározott szintaktikai és szemantikai szabályokat[61], másrészt statisztikai módszereket használnak, de ha a tanuló adathalmaz elég nagy, akkor a statisztikai módszerek jobbak. Szabályalapú eljárásokat tartalmaz például Ananthanarayanan et al. (2008), statisztikai módszereket Guangpu et al. (2011) cikke.

A tulajdonnévvel rendelkező elemek felismerése az információ kinyerési feladat része. Lényegében a megfelelő tulajdonneveket kell megtalálni egy szövegben (személynevek, földrajzi megnevezések, szervezetek nevei) és ezeket kell kategorizálni. A feladatot nehezíti, hogy beszédfelismerésnél az azonosítókat sokszor zajos szövegkörnyezetben kell felismerni.

60 Itt a „pontosságot" az angol „precision", és nem az „accuracy", értelmében használjuk.
61 A szabályalapú és a statisztikai elemzést ld. a továbbiakban.

A szabályok korlátai és a statisztikai típusú természetes nyelvi elemzés térnyerése[62]

A természetes nyelvi elemzések hőskorában a kutatók arra törekedtek, hogy mindenre kiterjedő szabályrendszert konstruáljanak a feldolgozáshoz, megpróbálták a nyelvtani és a szemantikai szerkezeteket teljes mértékben formalizálni, a természetes nyelvi szövegeket strukturálni, lényegében úgy tekintettek az emberi nyelvre, mint egy nagyon bonyolult, de megfelelő szabályrendszerrel leírható nyelvre, mint például egy programozási nyelvre. Ha ez sikerült volna, akkor a gépi fordítás problémáját már fél évszázada megoldották volna. A természetes nyelvek hatalmas terjedelme, a szigorú nyelvi korlátozások hiánya, a szavak, kifejezések esetleges többértelműsége, a nyelv fejlődése során kialakult nyelvtani és értelmezési kivételek, a kontextus meghatározó szerepe azonban nem teszik lehetővé a teljesen szabályalapú leírást. Az előre rögzített szabályok alapján történő elemzés okozta problémák két fő kategóriába sorolhatók:

1. Az NLP a szöveg jelentését, „szemantikáját" kell, hogy meghatározza, míg a mondatrészek – alany, állítmány, tárgy stb. – közötti formális nyelvtani szabályok elsősorban a szintaktikát írják le. Természetesen a pontosítás érdekében sok szemantikus szabályt is lehet alkalmazni (pl. az „eszik" állítmány csak ehető tárgyra vonatkozhat), de egy idő után a szabályok mennyisége kezelhetetlenné válik, ellentmondásokra vagy többértelműségekre vezet. A többértelműség igen gyakran a szóviccek alapja. (A halakat *kifogástalan* állapotban találta a halőr.)

2. A szakterületi szövegeket, amelyeket az ott dolgozók kiválóan értenek, például az államigazgatásban használt szövegeket, vagy az orvosi szakzsargont, a formális nyelvtanok nem tudják hatékonyan értelmezni. Például: „Az Összegzett Terheléses Score egy félkvantitatív megközelítés, mely összegzi a húsz myocardialis perfúziós szegmens automatikus értékeit a terheléses perfúziós vizsgálatnál."[63] Ez az orvosi megállapítás nehezen fejthető meg formális eszközökkel.

Tekintettel a fenti problémákra a természetes nyelvi elemzéssel foglalkozó kutatásokban a 80-as években a szabályrendszer tökéletesítésére irányuló törekvések háttérbe szorultak és előtérbe került a statisztikai megközelítés. Az egyszerű, de stabil statisztikai közelítő módszerek kiszorították a szabályalapú mély analízist, és a sztochasztikus gépi tanuló algoritmusok váltak jellemzővé a kutatásban. A statisztikai elemzésen alapuló gépi tanuló algoritmusok nagy, annotált szövegtesteket, ún. korpuszokat használnak a betanuláshoz, amelynek eredményeképpen megtanulják a szövegek értelmezését. Az annotáció a felügyelt tanuláshoz szükséges előzetes osztályba sorolásokat adja meg.

62 https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3168328/

63 http://tudasbazis.sulinet.hu/hu/magyar-nyelv-es-irodalom/magyar-nyelv/magyar-nyelv/2/fogalomgyujtemeny/szakzsargon

A kezdetben mondatokra épülő elemzésnél hatékonyabbnak bizonyult a tematikus megközelítés. A tematikus megközelítés abból indul ki, hogy egy adott korpusz „témáját" az abban leggyakrabban előforduló szavak határozzák meg. A gyakoriság meghatározásához először le kell csupaszítani a szavakat, azaz elő kell állítani a szótöveket (stemming)[64], vagy értelmes alapszavakat (lemmatization)[65], hogy a toldalékok ne befolyásolják az eredményt. A szótövek gyakoriságát egy szógyakorisági mátrixban lehet jól ábrázolni, ahol a mátrix egy sora egy dokumentumnak, egy oszlopa pedig egy szótőnek felel meg, a mátrix elemei pedig az egyes szótövek előfordulási gyakorisága egy dokumentumban (Term-Document-Matrix, TDM vagy transzponáltja, a Document-Term-Matrix, DTM). A szakirodalomban elfogadott ökölszabály szerint a 10% feletti előfordulási gyakoriság határozza meg a dokumentum témáját. A TDM nem csak szótöveket, hanem szókapcsolatokat vagy más nyelvi egységek gyakoriságát is tartalmazhatja, az elemző szándékától függően. A TDM lényegében a teljesen strukturálatlan dokumentumokat egy jól strukturált leírással reprezentálja.

Amennyiben az elemző előre megadott tartalmi kategóriákat és az ehhez kapcsolódó szavakat, szókapcsolatokat állít fel, akkor a TDM alapján egy felügyelt osztályozó algoritmussal be lehet sorolni a dokumentumokat egy adott osztályba, vagy másképpen mondva, meg lehet adni egy dokumentum osztályba tartozásának valószínűségi eloszlását.

Lássunk egy példát:

Az USA kormány az Open Data kezdeményezés keretei között létrehozta és üzemelteti a Socrata honlapot[66], amely számos szakterületről tartalmaz elemezhető adatokat. Az egyik ilyen adathalmaz a banki ügyfelek panaszainak gyűjteménye. Ez az adatállomány részben strukturált adatokat tartalmaz, ui. minden egyes panasznak egy rekord felel meg, amelyet elláttak egy azonosítóval, tartalmazza a benyújtás dátumát, az érintett bankot, a szóban forgó termék azonosítóját, reagált-e bank a panaszra, mennyi ideig tartott a panaszkezelés stb., de tartalmazza a panasz kapcsán az ügyfél és a bank között lezajlott levelezést is, ami természetes nyelven írott szövegek sorozata és így nem strukturált. Az elemzéshez – tekintettel a rendelkezésre álló számítógépes kapacitásokra – csak 1 millió rekordot töltöttünk le egy Excel-fájl formájában.

A strukturált adatokból elkészíthetjük a számunkra érdekes statisztikákat, pl. mely bankok szerepelnek leggyakrabban a panaszokban, mely időszakokban keletkezett a legtöbb panasz vagy mely bankok zárták le leghamarabb a panaszokat. Ezek a statisztikák egyszerű műveletek, akár az Excelben (az egymilliós korlát alatt), akár egy statisztikai programmal könnyen végrehajthatók.

A panaszok természetes nyelven történő leírása ennél jóval bonyolultabb. Az elemzésben a panaszok szövegeit önálló dokumentumoknak tekintettük és létrehoztunk egy korpuszt, amely közel egymillió dokumentumot tartalmazott. Ezután megtisztítottuk a korpuszt azoktól a szavaktól, amelyek gyakran előfordulnak, de az elemzés szempontjából

64 Magyar nyelvi szótövező pl. a Hunspell- és a Hunmorph-program (http://people.mokk.bme.hu/%7Ehp/papers/lncshp.pdf). A stemming nem mindig szolgáltat értelmes szótövet.

65 A lemmatizálás szótár segítségével történő, értelmes eredményt előállító eljárás.

66 https://dev.socrata.com/data/

nincs jelentőségük, ezek az ún. stopwordök (pl. and, a, an, one). Az angol nyelv stopword szótárát megtaláljuk például az R nyelv tm (text mining) programcsomagjában. Az angol nyelv egy másik stopword gyűjteményét megtalálhatjuk például githubon[67] is, ez 127 szót tartalmaz.[68] Amennyiben olyan szavakat is tartalmaz a dokumentum, amelyeket eleve ki szeretnénk zárni az elemzésből, mert az adott helyzetben nem hordoznak információt (pl. a dokumentum létrehozójának a sokszor ismételt neve) akkor ezeket is eltávolíthatjuk az elemzendő korpuszból.

A stopwordök és egyéb felesleges szavak eltávolítása után elhagyjuk a számokat, az írásjeleket, a felesleges szóközöket, majd a nagybetűket kisbetűkké alakítottuk. Ezzel kiemeltük az egyes panaszok tartalmára jellemző szavakat. A korpusz tovább finomítható a szótövezéssel.

A kapott korpuszból elkészítettük a TDM-et, amely az egyes szavak gyakoriságát tárolja. A TDM matematikailag ritka mátrix, ui. egy-egy dokumentumban a mátrixban lévő szavak csak egy kis része, 1-2%-a fordul elő, a mátrix kitöltöttsége ezért igen alacsony. Megjegyezzük, hogy a természetes nyelvi algoritmusok a TDM-ek tárolására speciális, ritka mátrix eljárást használnak, hogy a nagyobb korpuszokból készített TDM-ek is elférjenek a memóriában. Példánkban a nemüres dokumentumok száma az átalakítás után közel 900 ezer, a korpuszban szereplő szavak száma 214. A mátrix több, mint 188 millió elemet tartalmaz, ebből mintegy 3 millió különbözik a nullától. A mátrix sűrűsége (kitöltöttsége) kb. 2%.

A leggyakrabban előforduló szavak: „loan” (230 ezer előfordulás), „credit” (175 ezer), „incorrect” (130 ezer), „information”, (131 ezer), „debt” (111 ezer).

A TDM-ből kiindulva számos elemzést végezhetünk most már hagyományos statisztikai módszerekkel, ui. a TDM az eredeti, strukturálatlan dokumentumok egy strukturált reprezentációja.

Mint említettük, a statisztikai NLP átvette a hagyományos, szabályalapú elemzést. Ez különösen igaz azokra az esetekre, ahol az elemzéshez nagy korpuszok állnak rendelkezésre, amelyek elegendőek a modell betanítására. A statisztikai elemzők valószínűségeken alapuló, kontextusfüggetlen szabályokat (Context-Free Grammar, CFG) használnak, de a szabályok nem determinisztikusak, hanem azok valószínűségét a modell az annotált korpuszokból tanulja meg. Például a C5.0 döntési fa modelljét be lehet tanítani arra, hogy mikor melyik lehetőséget válassza az elemzésben. Így kevesebb, általánosabb érvényű szabályt használnak a sok elaprózott, egyedi eset helyett. Az elemző végeredményben mindig a legvalószínűbb felbontást adja. A statisztikai modellek kontextusfüggők, egy orvosi korpuszokon betanított elemző jól működik az egészségügyben, de nem használható pl. egy pénzügyi szöveg elemzésére (Manning et al., 1999).

A statisztikai modellek akkor adnak jó eredményt, ha elég nagy korpuszok állnak rendelkezésre, ezért az eljárást adatvezéreltnek nevezzük.

67 https://gist.github.com/sebleier/554280

68 Magyar nyelvű stopword szótárat találunk a http://members.unine.ch/jacques.savoy/clef/hungarianST.txt címen.

Adatvezérelt természetes nyelvi feldolgozó eszközök

- Nyílt forráskódú NLP könyvtárak: a könyvtárak NLP algoritmikus építőköveket tartalmaznak. Például az Algorithmia[69] részben ingyenes API-kat ad számos NLP algoritmus elérésére, anélkül, hogy szervereket vagy infrastruktúrát állítanánk fel. Az Algorithmia lényegében egy algoritmus piactér, ahol előre megírt algoritmusokkal kísérletezhetünk.
- Apache OpenNLP: gépi tanuló eszközök (karaktersorozatok szétdarabolása, mondat szegmentációja, beszédrészek bejelölése, nevesített entitások kiemelése, szótövező, elemző, referenciák megtalálása stb.).
- Natural Language Toolkit (NLTK): szövegfeldolgozó modulokat tartalmazó Python könyvtár, osztályozó, szétválasztó, szótövező, címkéző, elemző funkciókkal.

A fenti felsorolás távolról sem teljes, sok egyéb számítógépes megoldás is létezik.

Az N-gramok

Az N-gramok N nyelvi elemből álló sorozatok. Az elemek betűk, szavak, fonémák. A természetes nyelvi szövegekben egyes elempárok, tripletek, quadrupletek vagy még több elemből álló sorozatok gyakrabban fordulnak elő, mint mások. Az angol nyelvben pl. a Q-t mindig U követi, egy szókezdő T-t soha nem követi a K, a portugál nyelvben a Ç-t mindig egy magánhangzó követi. Ha elég sok adatunk van, akkor az N-gramokhoz statisztikailag releváns gyakorisági eloszlást lehet számolni. Minthogy N növekedésével a lehetséges N-gramok száma exponenciálisan nő, ezért a gyakorlatban nem célszerű túlságosan nagy N-et használni, mert nagy N-ek esetén az empirikus gyakoriságok előállításához csak kis minták állnak rendelkezésre, így ezek nem megbízhatók. A Google például elkészítette az angol szavak 5-gramjainak jegyzékét a Web-adatok és a Google Books adatbázis alapján.[70]

Amennyiben ismert egy korpusz vagy korpuszegyüttes N-gramjainak gyakorisági eloszlása, akkor ezt már többféle alkalmazásban is használhatjuk.

- Szavak és mondatok automatikus kiegészítése pl. keresésnél, mint pl. a Google interfészben.
- Automatikus javítás a szókörnyezet szavak alapján.
- Beszédfelismerés – az azonos hangzású szavak megkülönböztetése a szomszédos szavak alapján.
- Kétértelműség megszüntetése a korpuszokban annotált korrekt jelentéskörnyezet segítségével.

Az N-gram szótárak természetesen terjedelmesek, például a Google N-gram adatbázisa 28 GB, de minthogy a tár olcsó, ez nem játszik lényeges szerepet. Az N-gramok keresését speciális N-gram indexelési eljárások gyorsítják meg.

69 https://algorithmia.com/
70 https://books.google.com/ngrams/

Az eljárás előnye, hogy N-gram-alapú osztályozókat bármilyen nyelvészeti vagy szakterületi tudás nélkül is lehet alkalmazni nyers szövegekre. Az eljárás önmagában is hatékony, de a szakterületi tudás felhasználásával lényeges mértékben pontosítható.

A szózsák (Bag of Words – BoW) modell

A BoW-modell szöveges adatok reprezentálásának egy formája a gépi tanuló algoritmusokban. A BoW-modell egyszerű, könnyű implementálni és hatékony a nyelvi modellezésben és a dokumentumok osztályozásában.

Ahogyan az előzőekben már láttuk, a gépi tanuló algoritmusok nem képesek közvetlenül használni a természetes nyelvi szövegeket, a feldolgozáshoz át kell alakítani azokat strukturált formába. Ez általában egy numerikus vektor, amely jellemző a szövegre. A transzformáció eszköze a jellegkiemelés (feature extraction). A jellegkiemelés egyik egyszerű módszere a BoW-modell.

A BoW megadja, hogy a szavak milyen gyakorisággal fordulnak elő egy dokumentumban. Ehhez két dolog kell:

- az ismert szavak szótára,
- az ismert szavak jelenlétének mértéke.

A „zsák” megnevezés azt jelenti, hogy ellentétben az N-gramokkal, az elemzésnél eltekintünk a szavak sorrendjétől, a szószerkezetektől. A modell csak azt veszi figyelembe, hogy egy szó jelen van-e a dokumentumban, azt, hogy hol, azt nem. Itt a szöveg jellege a szavak gyakorisága.

A szózsák modell azt feltételezi, hogy a dokumentumok hasonlóak, ha a tartalmazott szavak gyakorisága hasonló és a tartalmazott szavak gyakoriságából következik a dokumentum tartalma. A szózsák tetszőleges bonyolultságú lehet, ha ezt látjuk célszerűnek. Megadhatjuk, hogy mely szavak szerepeljenek a szótárban, és hogy hogyan mérjük a szavak előfordulását.

Ha a korpusz nagy, esetleg több ezer kötetnyi, akkor az előfordulási vektor sok tízezer pozíciót tartalmazhat (ld. Term-Document-Matrix). Ugyanakkor egy adott dokumentum csak viszonylag kevés szót tartalmaz a teljes szótárból, így az előfordulások helyén sok a nulla érték, a vektor pedig ritka (sparse). A ritka vektorokkal való műveletekhez ugyan speciális eljárásokat is használhatunk, de célszerű a szótár méretének a lehető legkisebbre vétele. Ehhez sokféle szövegtisztítási technikát alkalmaznak, mint például:

- A kis és nagybetűket nem különböztetik meg
- A központozást ignorálják
- A stopwordöket kihagyják
- A hibásan beírt szavakat kijavítják
- A szavakat a szótővel helyettesítik

Bonyolultabb szótárakat szócsoportokból (N-gramok) is lehet képezni. Az N-gram szótárak jobban tükrözik a dokumentum jelentését. A szótár célszerűen csak a dokumentumban szereplő N-gramokat tartalmazza.

Példaképpen tekintsünk egy dokumentumot:

Jött értem a fekete hajó[71]
Jött értem fekete vízen.

A szavak és előfordulásuk:
jött – 2
értem – 2
a – stopword, figyelmen kívül hagyjuk
fekete – 2
hajó – 1
vízen – 1

A BoW-modell:
Szótár: {jött, értem, fekete, hajó, vízen}
BoW-szóvektor: (2,2,2,1,1)
bigramok:
{jött értem}, {értem, fekete} {fekete hajó}, {fekete, vízen}
bigramok előfordulása: (2,2,1,1}

A fenti korpusz igen kis méretű, ezért a modell alapján nem célszerű bármilyen következtetést levonni.

A BoW-modell korlátai

A modell egyszerű és hatékony, sikeresen használják szövegelőrejelzési és -osztályozási algoritmusokban, ugyanakkor vannak hiányosságai:

- A szótár megszerkesztése bonyolult feladat, a nagy méretek és a dokumentum reprezentációs vektorok kis sűrűsége miatt gondos tervezést igényel számítástechnikai szempontból.
- Kis sűrűség: a ritka vektorokkal és mátrixokkal történő reprezentáció nem csak számítástechnikai szempontból kezelhető nehezen, hanem az információ értelmezése szempontjából is, ui. kevés információt kell kinyerni egy nagyon nagy reprezentációs térből.
- Jelentés: a BoW-modell figyelmen kívül hagyja a szavak sorrendjét, a kontextust és így a szavak pontos jelentését is. A sorrend és a kontextus ugyanakkor komoly szerepet játszhat a szavak értelmezésében. (Pl. „Ez a könyv érdekes.” és „Érdekes ez a könyv?” vagy „régi bútordarab” és „régi bútor”.)

71 Karinthy Frigyes: Így irtok ti. http://mek.oszk.hu/00700/00716/html/

Nyelvi eszközök a vélemény- és hangulatelemzésben

A digitális médiában megjelenő vélemények elemzése mind az üzleti világban, mind a közszolgáltatásokban fontos feladat. A döntéshozatali folyamatokban fontos tudni, hogy mit gondolnak az emberek, tetszik-e nekik egy termék, egy szolgáltatás, egy ötlet, egy szabályozási tervezet vagy nem. A webes korszak előtt is megkérdeztük ismerőseinket, barátainkat, hogy tudnak-e egy jó autószerelőt javasolni, milyen volt a tavalyi balatoni nyaralásuk, hogyan vált be a nemrég vásárolt mosógépük. A web általánossá válásával lehetővé vált, hogy sok, nagyrészt általunk nem is ismert ember és függetlennek tekinthető szakember véleményét és értékelését is figyelembe vegyük egy-egy döntés meghozatalakor. Megfigyelhetjük, hogy azok száma, akik önként megosztják véleményüket az interneten, folyamatosan nő, míg azok száma, akik egy kérdőíves felmérésre válaszolnak, egyre csökken.

Egy korábbi amerikai felmérés szerint[72] a vásárlók 81%-a egy termék megvásárlása előtt tájékozódik az interneten, a turizmussal összefüggő szolgáltatásokkal kapcsolatos online vélemények jelentős szerepet játszanak az utazások megtervezésénél, és az interneten kiváló értékelést kapott termékekért a vásárlók 20–99%-kal többet hajlandók fizetni. Pang et al. (2008) könyve a vélemények elemzését mutatja be tudományos részletességgel.

A véleményelemzés egyre nagyobb szerepet kap a politikusok eszköztárában is. Egy-egy beszéd, cselekedet, program hatására sok ember közli véleményét közösségi oldalakon, a Twitteren, értékelő fórumokon vagy más internetes médiumon, és ezek elemzése a közvetlen lekérdezéses kutatásnál gyorsabb és pontosabb eredményt adhat sokkal kisebb költséggel. Természetesen a véleményüket közlők nem tekinthetők reprezentatív mintának, mert az internetet nem használók, vagy azok, akik nézeteiket nem teszik publikussá, nem szerepelnek egy ilyen mintában. Tekintettel a hagyományos közvélemény-kutatásokban általában használt minták korlátozott elemszámára és arra, hogy az emberek a globális statisztikák szerint egyre kevésbé adják meg valós véleményüket, az internetes adatokból származó véleményelemzés minősége nem marad el a hagyományos kutatásokban mért adatokétól[73], ugyanakkor az eredményekre nem kell várni, alacsony költséggel számíthatók, a minta nagysága pedig sokkal nagyobb is lehet.

Véleményelemzés tweetekkel

Azokban az országokban, ahol az emberek sokat tweetelnek, a tweetek elemzéséből is valós idejű információt lehet kapni arról, hogy hogyan vélekednek egy-egy termékről, állami intézkedésről, politikusi állásfoglalásról. A Twitter viszonylag egyszerű módon bárki számára lehetővé teszi korlátozott számú (alkalmanként egy-háromezer) tweet letöltését. A letöltésnél szűrhetünk kulcsszavakra, a tweet írójának földrajzi koordinátáira, a tweetek nyelvére stb. A letöltött tweetek ezután tisztítás és szótövezés után készen állnak az elemzésre. A legegyszerűbb vélemény- vagy hangulatelemzési módszer az, amikor a tweetek szavait vagy N-gramjait összehasonlítjuk egy előzetesen összeállított listával, amely tartalmazza

72 comScore/the Kelsey group. Online consumer-generated reviews have significant impact on offline purchase behavior. Press Release, November 2007. http://www.comscore.com/press/release.asp?press=1928.

73 Ezt az állítás a szerző véleménye, nem kvantitatív kutatás eredménye.

a nyelv pozitív és negatív véleményt tükröző szavait vagy N-gramjait. Amennyiben egy tweetben találnuk egy pozitív tartalmú szót, akkor +1, ha negatívat, akkor -1 pontot adunk a tweet véleményértékéhez, majd ezeket a pontszámokat tweetenként összesítjük, így minden tweetről megállapítjuk, hogy negatív, pozitív vagy semleges tartalmat fejez ki attól függően, hogy a pontértéke negatív, pozitív vagy nulla. A pontszám abszolút értéke azt is megmutatja, hogy a tweetelő véleménye mennyire erős. A pontszámokból hisztogram rajzolásával a vélemények empirikus eloszlására vonatkozó információt is kapunk, megállapítható, hogy az eredmények mennyire szórnak.

Példa:
2018.06.07-én letöltöttünk 1500 angol nyelvű tweetet, amely tartalmazta a Facebook kulcsszót. Az összesített pontszám 35 volt, ami azt jelenti, hogy a Facebook megítélése ekkor egy kicsit jobb volt a semlegesnél.

Magyar nyelven a Facebook kulcsszóval csak 16 tweetet találtunk. Néhányat idézünk, rövidített formában.
@facebook A facebooknak fel sem tűnik hogy az igazság van leírva? De 2016-os az a bejegyzés amiért
@facebook @Imamofpeace Boycott Facebook”
@facebook Boycott Facebook”
@facebook mvmg”
@facebook Nem tűnik fel önöknek hogy 2016-os kommentárt tiltottak? Önöknek az a lényeg hogy mi
@facebook ➡ Nazibook https://t.co/VX1amB695Z

Azért idéztük a fenti tweeteket, hogy lássuk, hogy a véleményelemzés egyszerű szótárral való összehasonlítása nem adhat mindig megbízható eredményt. A legelső tweet nyilvánvalóan negatív véleményt fejez ki, de nem tartalmaz egyértelműen negatív szavakat és a fenti véleményelemzésben semleges lesz.

A negatív és pozitív szavakkal történő összehasonlítás érzéketlen az ironikus kifejezésekre. Például: „Módomban volt órákig nézegetni a pályaudvar gyönyörű mennyezetét, mert a vonatom sokat késett”. Ez a mondat összességében nem lesz negatív az elemzésben, pedig tartalmilag erősen az.

A szótáras elemzéshez természetesen megfelelő szótárak kellenek, ilyenek angol nyelven rendelkezésre állnak az interneten. A SentiWordnet[74] mondatrészeket tartalmaz és nem egyszerűen negatív vagy pozitív besorolást ad, hanem ezeket egy erősségi skálán helyezi el. Hu és Liu[75] pozitív és negatív szavakat tartalmazó szótára az angol nyelv pozitív és negatív jelentésű szavait sorolja fel, mindegyikből néhány ezret. Ezt a szótárat használtuk a fenti angol nyelvű tweetek elemzésnél.

74 http://sentiwordnet.isti.cnr.it/
75 https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html

Bár a fenti módszerek kvantitatív értelemben távolról sem pontosak, ha megfelelő számú tweet áll rendelkezésre, akkor kvalitatív módon el lehet dönteni, hogy valaminek a társadalmi megítélése többnyire pozitív, negatív vagy semleges. Mindezt percek alatt és olcsón.

A véleményelemzést nem csak rövid üzenetek, hanem hosszabb szövegek esetén is alkalmazhatjuk. Minél jobban tudjuk szűkíteni a vizsgált témakört, annál pontosabb szótárakat tudunk szerkeszteni az elemzéshez. A pontosságot természetesen befolyásolja a szavak jelentésének módosulása különböző szövegkörnyezetben. Így a szótáralapú elemzéseknél az eredményeket célszerű tájékoztató jellegűnek felfogni. Az alábbi példán egy más jellegű, mélyebb klasszifikáción alapuló, szintén a tweetekből kiinduló elemzést mutatunk be.

Hangulatelemzés gépi tanulással

Corallo és munkatársai (2015) cikke a „Lecce – 2019 – European Capital of Culture" témával kapcsolatos twitteres hangulatelemzést mutatja be. Az elemzéshez használt tweeteket 2014 őszén gyűjtötték össze a Twitterben elérhető API segítségével.[76] A szerzők kifejtik, hogy az állampolgárok véleményének összegyűjtése egyszerűbb, biztonságosabb és költséghatékonyabb módszer, mint a hagyományos felmérés, ahol a hamis állítások kockázata jelentős. A közösségi hálók elemzésével az állampolgárok véleményének szélesebb körű felmérésére van mód, lehetőség van a vélemények és fontos események automatikus észlelésére, valamint az állampolgárok közszolgáltatásokkal kapcsolatos véleményének azonnali megismerésére Az információ megismerése segíti a közintézményeket a gyors helyzetfelmérésben és a döntéshozatalban.

Lecce az EU kulturális fővárosa című pályázatában a „Reinventing Eutopia" jelmondatot fogalmazta meg, és az „európai álom" politikai, társadalmi, kulturális és gazdasági perspektíváit írja le.[77]

A tweeteket a „# Lecce 2019" és „#noisiamolecce2019" hashtagekre szűrték, ezzel mintegy 5000 tweetet sikerült begyűjteni. A duplikátumok és retweetek kiszűrése után 1700 értékelhető tweet maradt. A tweeteket három független annotátor értékelte, akik a tweeteket pozitív, negatív és semleges kategóriába sorolták. Az annotátorok véleménye 80%-ban megegyezett. Ahol eltérés volt, ott egy vezető sorolta be a tweeteket a megfelelő osztályba. Ezzel létrejött egy manuálisan annotált, a tweetek további értékeléséhez használható modell betanítására alkalmas korpusz.

A korpuszt ezután feldolgozható formára alakították:

- kiemelték a jellemzőket (feature extraction). Ebben a fázisban az unigramokat, bigramokat és trigramokat vették figyelembe az elemzők, az ennél hosszabb N-gramok már nem voltak érdekesek,
- megtisztították a szövegeket a stopszavaktól, írásjelektől, hashtagektől, egy-két betűs szavaktól, emotikonoktól, elvégezték a szótövezést stb. és
- kijelölték a beszédrészeket.

76 A tweetek letöltésének technikájáról részletes leírást találunk számos internetes forrásban, pl. itt: https://www.credera.com/blog/business-intelligence/twitter-analytics-using-r-part-1-extract-tweets/

77 https://www.lecce2019.it/2019/utopie.php

Az előkészítés után a megtisztított tweetekből kiválasztották a tanuló- és a teszthalmazt, amelyek 1000, ill. 700 tweetet tartalmaztak. A tanulóhalmazt ezután 8 különböző klasszifikációra használták az alábbi, már strukturált adatokon:

(1) uni-gram, (2) bi-gram, (3) uni-gram+bi-gram, (4) uni-gram+stop szavak törlése+ismételt betűk törlése, (5) az előző+szótövezés, (6) az előző+emoticonok figyelembevétele, (7) az előző+hashtag a # jel nélkül, (8) ez előző+beszédrészek.

Minden klasszifikációt 10-szer ismételtek meg különböző tanulóhalmazokon (100, 200, 300, ... ,1000 tweeten), majd az eredményeket a tesztadathalmazon validálták.

Az előzőekben említettük a szótárakon alapuló véleményelemzés módszereinek gyengeségeit, elsősorban a szövegkörnyezet befolyásoló hatását, amelyet a módszer figyelmen kívül hagy, ezért nem lehet általános érvényűen meghatározni egy érzelmi tartalmat hordozó szó pozitív vagy negatív súlyát. Ne felejtsük, hogy viszonylagos pontatlansága ellenére a módszer igen hasznos, mert valós idejű és olcsó, mert nem igényli a manuális annotálást, míg a gépi tanuláson alapuló eljárás a modell betanítása miatt nem valós idejű és manuális feldolgozást is igényel.

A hangulatelemzésben használt felügyelt gépi tanuló algoritmusok alkalmazása lehetővé teszi, hogy a modell érzékelje a szövegkörnyezetet, és ebben a szövegkörnyezetben pontos eredményeket szolgáltasson. Természetesen, ha a modellt eltérő tematikájú szövegekre alkalmazzuk, az eredmény pontossága lényegesen romlik. A fent idézett cikk szerzői két különböző klasszifikációs megközelítést alkalmaztak:

- Az egyedi szövegdokumentumok osztályozása.
- A teljes adathalmaz osztályozása.

Az egyedi dokumentumok osztályozására a Naive Bayes és az SVM általános gépi tanuló klasszifikációs eljárásokat használták, amelyeket a *Klasszifikációs eljárások* című fejezetben ismertetünk, míg a teljes adathalmaz szintjén történő osztályozáshoz King és Hopkins politikai véleményeket tartalmazó blogok cikkeit elemző komplex eljárását követték (King et al., 2010), amely az N-gramok helyett a BoW-típusú strukturálást alkalmazza. Ezt a módszert az idézett cikkben ismerhetjük meg részletesen.

Az eljárások pontosságát az RMSE-értékkel mérték (ld. 11. fejezet). Az első három tanulási szcenárió gyakorlatilag egyforma pontos eredményt adott, a hibák négyzetes összege átlagának négyzetgyöke 0,1 körüli érték volt, a pontosság 78%. A Naive Bayes és az SVM osztályozónál ez alig függött a tanulóhalmaz méretétől. A King és Hopkins cikkében leírt módszer pontosabb eredményt adott, és ott a tanulóadatok számának növekedésével a pontosság is nőtt, de nem lényeges mértékben.

Összefoglalva, a vélemény- és hangulatelemzés egyszerű módszere a megtisztított szövegek összehasonlítása egy előre megadott szótárral vagy N-gram-gyűjteménnyel és a pozitív/negatív/semleges találatok összesítése. Ez a módszer valós időben hajtható végre és számítási igénye sem jelentős. Ugyanakkor pontossága korlátozott.

Alternatív módszer egy korpusz összeállítása és manuális annotálása, majd egy felügyelt gépi osztályozó modell betanítása az adott tematikus környezetben a szövegekben leírt vélemény/hangulat elemzésére. Ez nem valós idejű, költségesebb, de pontosabb módszer. Az elemző feladata, hogy kiválassza, hogy az adott célnak milyen eljárás felel meg legjobban.

A chatbotok

A chatbotok olyan programok, amelyek üzenetekre, kérdésekre válaszolnak, ill. saját maguk kezdeményeznek üzentváltásokat. A kommunikáció folyhat írásban vagy hangüzenetek formájában és lehetnek írásos vagy hangüzenetek. A chatbotok lehetnek egyszerűbbek, amelyek mindig ugyanazt válaszolják a hasonló kérdésekre, bonyolultabbak, amelyek a választ a kérdésben lévő kulcsszavak alapján alakítják ki, és léteznek öntanuló chatbotok, amelyek az emberekkel történő kommunikáció során a gépi tanulás eszközeinek segítségével alkalmazkodnak a szituációhoz. Sokféle médiumon képesek működni, SMS-en, weboldalak chatablakaiban vagy a közösségi üzenetközvetítő hálózatokon, pl. a Twitteren és a Facebookon. Gyakorlati alkalmazásuk igen gyors ütemben növekszik, hiszen napi 24 órában képesek kiszolgálni az ügyfeleket, immunisak az ügyfélszolgálatosokat érő stresszhatásokra és igen költséghatékonyak.

A Facebook szerint[78] a Messengeren havonta több mint 2 milliárd üzenetet váltanak egymással az emberek és a vállalkozások, az automatikus és az emberek által kezdeményezett üzeneteket is számítva. A Messenger segítségével egyszerre sok embernek lehet üzenetet küldeni, majd külön-külön lehet folytatni a kommunikációt. Ha egy marketinges egy terméket akar megismertetni a potenciális ügyfelekkel, a Messengeren automatikus chatbottal küldött üzenetek ehhez ideálisak lehetnek. A cég adatai szerint az emberek 53%-a szívesebben vásárol olyan vállalkozástól, amelyikkel közvetlen üzenetet válthat. Az emberek 56%-a telefonálás helyett szívesebben vált üzenetet az ügyfélszolgálattal. A Facebook természetesen felismerte a Messengeren működő chatbotok üzleti értékét és támogatja a felhasználókat saját chatbot létrehozásában, amelyek a Messenger kommunikációs mechanizmusára épülnek.[79]

A Facebook létrehozott egy chatbot fejlesztőknek szóló kézikönyvet is, ami lépésenként vezet végig egy chatbot felállításának néhány órás folyamatán.[80] Azonban a chatbot minősége, azaz hogy mennyire intelligensen tud válaszolni a kérdésekre, és milyen kérdésekre tud válaszolni, igen széles határok között mozoghat, a nagyon fókuszált, egyszerű megoldásoktól a bonyolult, önálló tanulásra is képes mesterséges intelligencia algoritmusokig.

A Facebook chatbotok számára már világversenyt is rendeztek[81], ahol 7 kategóriában hirdettek győztest. Ezek nem általános célú chatbotok, hanem valamely témához, az utazáshoz, közösségi kommunikációhoz, szórakoztatáshoz stb. kötődnek. Ezek a botok angol nyelvűek, használatukhoz facebookos bejelentkezés szükséges, és tárolnak minden, a felhasználó által megadott adatot. Itt is, mint minden más „ingyenes” Facebook-szolgáltatásnál a felhasználó személyes adataival „fizet”.

78 https://www.facebook.com/business/products/messenger-for-business
79 https://developers.facebook.com/videos/f8-2016/introducing-bots-on-messenger/
80 Messenger Bot Tutorial: Step-by-Step Instructions for Building a Basic Facebook Chat Bot blog: https://blog.hartleybrody.com/fb-messenger-bot/
81 http://www.fbchatbot.hu/cikk/7-chatbot-amit-mindenkinek-ismernie-kene

A chatbotok alkalmazása a közigazgatásban sok előnnyel jár.[82]

Az állampolgárok számára nyújtott előnyök:

- Az informatikai ismeretekkel nem rendelkező állampolgárok is könnyen hozzáférhetnek a nyilvános adatokhoz.
- Panaszaikat, észrevételeiket természetes nyelven is megfogalmazhatják és online küldhetik el.
- A formai előírásokat előíró beadványok is beküldhetők a chatbotokon keresztül.
- Online fizethetnek.
- Segítséget kérhetnek természetes nyelven.
- Csökken az emberi részvételt igénylő, ezért költségesebb és lassúbb telefonos és e-mail-kommunikáció.

A közszolgáltatások számára nyújtott előnyök:

- Természetes nyelvi, személyre szabott kommunikáció az állampolgárokkal.
- Egyes közszolgáltatások egyszerűsödése.
- Napi 24 órás, heti 7 napos folyamatos elérhetőség biztosítása.
- A válaszidők csökkenése, az emberi munkaerő részleges mentesítése egyes ügyfélszolgálati tevékenység alól.
- Az információs rendszerekkel való közvetlen integráció és menedzsment.
- Többnyelvű tájékoztatás és segítségnyújtás lehetősége.
- Több kommunikációs csatorna használatának lehetősége (SMS, írásos üzenet, hang).

Az intelligens chatbotok fejlesztésére jelenleg a legtöbb erőforrást a nagy, sok ember által beszélt nyelvi környezetekben fordítják, mert ezeken a területeken a legnagyobb a várható haszon is. Minthogy azonban a korszerű természetes nyelvi feldolgozás gépi tanulási módszerei adatvezéreltek, azaz az algoritmusok a tudást jellemzően nyelvi korpuszokból merítik, a módszertani alapokat más nyelvekre is lehet alkalmazni, amennyiben rendelkezésre állnak kézzel annotált nagy korpuszok az adott nyelven. Magyarországon is dolgoznak a kutatók ilyen korpuszok kialakításán[83], így várhatóan a magyar nyelvű chatbotok fejlesztése is együtt haladhat az angol, spanyol vagy kínai nyelvű fejlesztésekkel. Az interneten számos hazai chatbot fejlesztői csapatot találunk, akik egyszerűbb, vagy esetenként komplexebb, eszközök kialakítására is vállalkoznak. Üzleti szempontból a leggyorsabban megtérülő beruházások a célorientált marketing chatbotok, így ezek tömeges elterjedése várható a közeljövőben.

A közszférában a kommunikáció jellege különbözik az üzleti kommunikációtól, a tájékoztatásnak egyértelműnek és részleteiben is szabályozottabbnak kell lennie, hiszen a hivatali

82 https://chatbotsmagazine.com/how-chatbots-are-beneficial-to-government-agencies-6e21052e3ba4

83 Ld. pl. Szeged Korpusz és Treebank (https://www.inf.u-szeged.hu/rgai/kutatas/nyelvtech) vagy az MTA Nyelvtudományi Intézet Magyar Nyelvi Szövegtára (http://mnsz.nytud.hu/).

üzenetek akár hivatkozási alapul is szolgálhatnak a további eljárásokban. Ezért az üzleti életben alkalmazott chatbotok sokszor „lazább" kommunikációs stílusa nem igazán ültethető át a közigazgatási szférába. Az üzleti célra használt chatbotok sokszor a lazább stílussal oldják az esetleges konfliktusokat, például, amikor nem tudják a választ egy kérdésre. A magasabb pontosság, a tévedések minimalizálásának követelménye, a tévedések kizárásának igénye miatt a közigazgatási alkalmazások elterjedésének üteme természetszerűleg lassabb, mint az üzleti célú chatbotoké.

Néhány nemzetközi példa

1. Los Angeles CHIP (City Hall Internet Personality)

A Los Angeles Business Assistance Virtual Network napi mintegy 180 ember munkáját támogatja. A chatbot bevezetése 50%-kal csökkentette a levelezés volumenét. A rendszer a Los Angelesben tevékenykedő üzleti vállalkozások kérdéseire ad válaszokat, segíti a város és a vállalkozások közötti üzleti tevékenységet, miközben folyamatosan tanul. 2017-es indulásakor 200 kérdésre tudott válaszolni, ez a szám gyorsan 700-ra növekedett. A rendszer a Microsoft Azure felhőben működik. [84]

2. GovBot (Botty Bonn) Németországban

Adminisztratív kérdésekre ad választ integrált adminisztratív tudástár alapján. Jelenleg kísérleti, tanuló fázisban van, és néhány területre korlátozódik a válaszadási lehetősége, mint például a parkolási információ, időjárás vagy az adminisztratív ügyintézés a városban. Angolul is tud a németen kívül.[85]

3. Alex– az Ausztrál Adóhivatal chatbotja

Az adózással kapcsolatos kérdésekre ad választ. Jelenleg kísérleti verzióban működik.[86]

A legjobb chatbotok

A jó chatbotok – ugyanúgy, mint más jó minőségű szoftverek – jelentős piacra tehetnek szert. Ezért a fejlesztők igyekeznek objektív módon versenyeztetni a chatbot termékeket.[87]

Mitől jó egy chatbot? Nyilván attól, ha minél több kérdésre minél pontosabb választ tud adni és a kommunikáció közben néha el tudunk felejtkezni arról, hogy nem valódi emberrel csevegünk.

84 http://www.govtech.com/computing/Los-Angeles-Microsoft-Unveil-Chip-New-Chatbot-Project-Centered-on-Streamlining.html
85 https://govbot.bonn.de/
86 https://beta.ato.gov.au/Tests/Introducing-Alex--our-new-web-assistant
87 What Are The Best Intelligent Chatbots or AI Chatbots Available Online? https://chatbotsmagazine.com/which-are-the-best-intelligent-chatbots-or-ai-chatbots-available-online-cc49c0f3569d

Minden évben kiírják az ún. Loebner-díjat a legjobb chatbotok elismerésére. Az értékelésnél fontos szempont az emberhez hasonló viselkedés, minél inkább összetéveszthető a robot egy emberrel, annál jobb értékelést kap. A mérés eszköze a szabványos Turing-teszt. A teszt abból áll, hogy a tesztelő (ember) írásban kérdéseket tesz fel két tesztalanynak, akiket így se nem láthat, se nem hallhat. A két alany egyike ember, míg a másik egy gép. Ha a kérdező ötperces kommunikációs után sem tudja egyértelműen megállapítani, hogy a két alany közül melyik a gép, akkor a gép sikerrel teljesítette a tesztet.[88]

Amelyik chatbot a legközelebb áll a Turing-teszt teljesítéséhez, az kapja a Loebner-díjat. A díjat a Mitsuku nevű botchat három alkalommal nyerte el az elmúlt években.[89] Kipróbálható a hivatkozott oldalon, sajnos, csak angol nyelven.

Hogyan történik egy chatbot betanítása?

Az angol nyelven leggyakrabban használt chatbotok az Apple Siri, a Microsoft Cortana, a Google Assistant és az Amazon Alexa. Tájékoztatnak az időjárásról a világ bármely pontján, vezetés közben az útirányról, a sportesemények állásáról, felhívhatnak egy számot a telefonkönyvünkből vagy vásárolhatunk rajtuk keresztül egy cipőt. Mindezt akár hangüzenetek igénybevételével. De hogyan tanítják be a programokat? A következőkben az írott üzenetekkel foglalkozunk, a beszéd–szöveg és szöveg–beszéd konverzió kérdéseit nem vizsgáljuk.

A chatbotnak meg kell találnia a legjobb választ a feltett kérdésekre, releváns információt kell szolgáltatnia, ha ezt nem tudja, akkor vissza kell kérdeznie, és ésszerű módon folytatnia kell a beszélgetést. Mindehhez nem csak meg kell értenie a kérdező szándékát, hanem a helyes választ nyelvileg helyesen kell megfogalmaznia, figyelembe véve az adott nyelv nyelvtani és lexikai szabályait. Jelenleg azonban a fejlesztések még nem jutottak el oda, hogy a fenti követelményeknek maradéktalanul megfeleljenek. Egy egyszerű példa[90] mutatja, hogy egy chatbot könnyen hibázhat.

„You need to start understanding me, Siri”
I’ll make a note of that
„Yes, you better make a note of that”
Of that

A félreértés a „make a note” (magyarul: „megjegyez”, „feljegyez”) kétértelműségén alapul.

A gépi tanuló algoritmusok általában egy statisztikailag stabil adathalmazon dolgoznak. A természetes nyelvi szövegek nem ilyenek, *A neurális hálók* című fejezetben bemutattuk, hogy hogyan lehet az LSTM neurális háló modellel elérni, hogy ne felejtse el a múltbéli adatokat s válaszolni tudjon olyan kérdésekre is, amelyeknek a megválaszolásához szükséges a múlt ismerete is.[91]

88 Wikipedia: https://en.wikipedia.org/wiki/Turing_test
89 https://www.pandorabots.com/mitsuku/
90 https://www.kdnuggets.com/2017/08/deep-learning-train-chatbot-talk-like-me.html
91 https://research.fb.com/downloads/babi/

9. HÁLÓZATELEMZÉS A KÖZSZOLGÁLATBAN

A hálózatelemzés alkalmazási lehetőségei

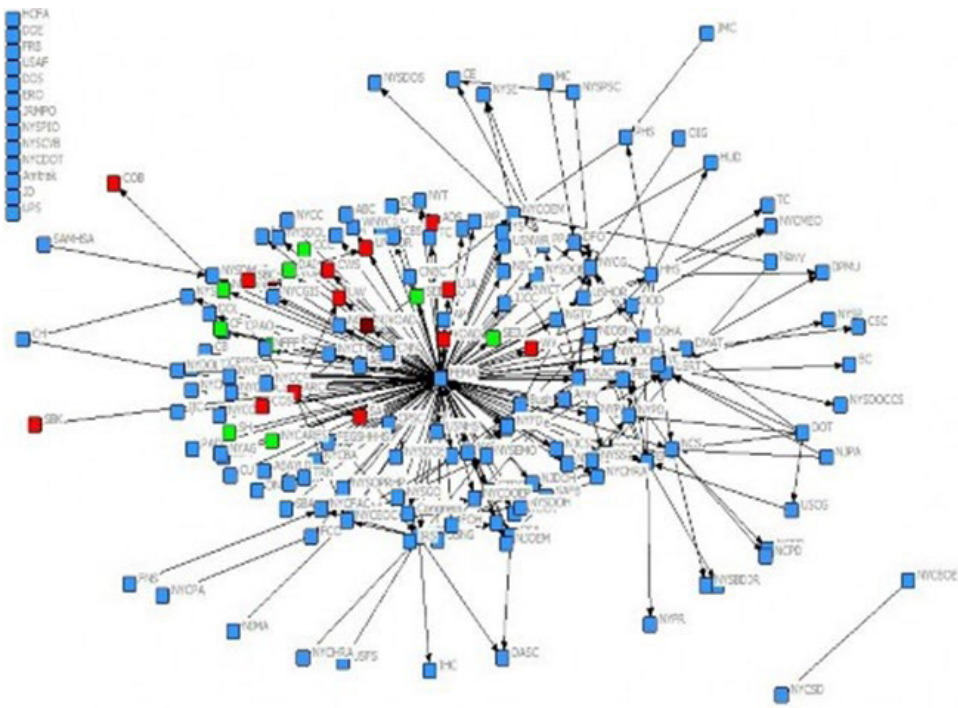
Az állami, ill. közszféra egyes szolgáltatásai, mint például a katasztrófaelhárítás, a terrorizmus elleni harc, a bűnüldözés, a tömegközlekedés szervezése, az infrastrukturális fejlesztések a közszféra intézményeinek, a gazdasági és társadalmi szervezeteknek hálózatszerű együttműködésében valósulnak meg.

Az államigazgatás különféle szervezetei is hálózati kapcsolatban állnak és dolgoznak. A hálózatra jellemző, hogy a résztvevők nemhierarchikus struktúrában, hanem együttműködő egységekként végzik a tevékenységüket. Az államok közötti együttműködés hatékonyságát kritikus helyzetekben az egymás közötti információ és erőforrás megosztása lényegesen javítja. A hálózatra ugyan nem a hierarchikus működés a jellemző, de ez nem jelenti azt, hogy a hatékony együttműködés során ne alakulnának ki a megfelelő struktúrák. A több szervezet együttműködésében megvalósuló szolgáltatások, a kollaboratív közszolgálat nem csak a szervezetek közötti információmegosztást és az erőforrások egyesítését, hanem új, közös elképzelések kialakítását, innovációt és gyakran az izolált működésből eredő hátrányok kiküszöbölését is jelenti.

Példaként bemutatjuk (38. ábra) a különböző igazgatási és nonprofit szervezetek együttműködését a New York-i 9/11-es terrortámadás után. Az ábrán a NVOAD (National Voluntary Organizations Active in Disaster) és más nonprofit szervezetek együttműködési hálózatát látjuk 2001. szeptember 11. után. A piros négyzetek a NVOAD-tagokat, a zöldek az egyéb nonprofit szervezeteket, a kékek az egyéb szervezeteket, a sötétvörösek más VOAD-szervezeteket jelölnek (Kapucu et al., 2017), (Koliba et al., 2010), (Rathemeyer et al., 2008).

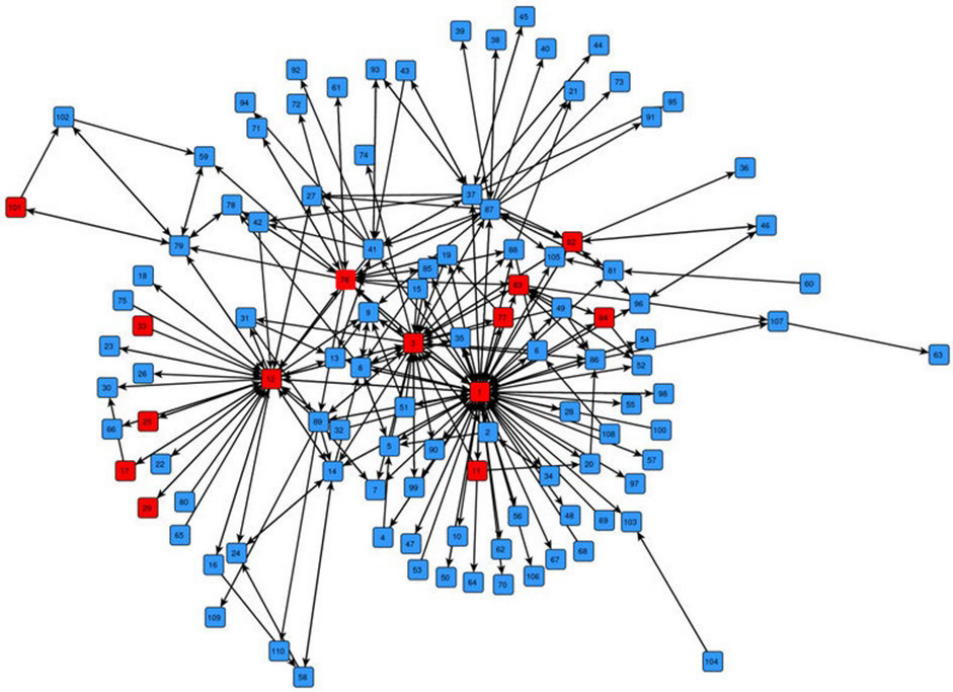
A leginkább központi szerepet játszó szervezetek a Szövetségi Katasztrófaelhárítási Ügynökség (FEMA), a New York Polgármesteri Hivatal, az USA Egészségügyi és Humán Szolgáltatások Minisztériuma, a FEMA helyi irodája, az USA hadseregének mérnöki testülete.

A 38. ábrán jól kirajzolódik a résztvevők együttműködési struktúrája. A hálózat központjában a FEMA (Federal Emergency Management Agency) helyezkedik el, és ez a szervezet tartja a kapcsolatot a legközelebbi, centrálisan elhelyezkedő egyéb szervezetekkel, amelyek azután másokat vonnak be a munkába. A hálózati struktúra elemzése jelentősen megnövelheti a hatékonyságot, gyorsíthatja az információáramlást és az erőforrások optimális allokálását.



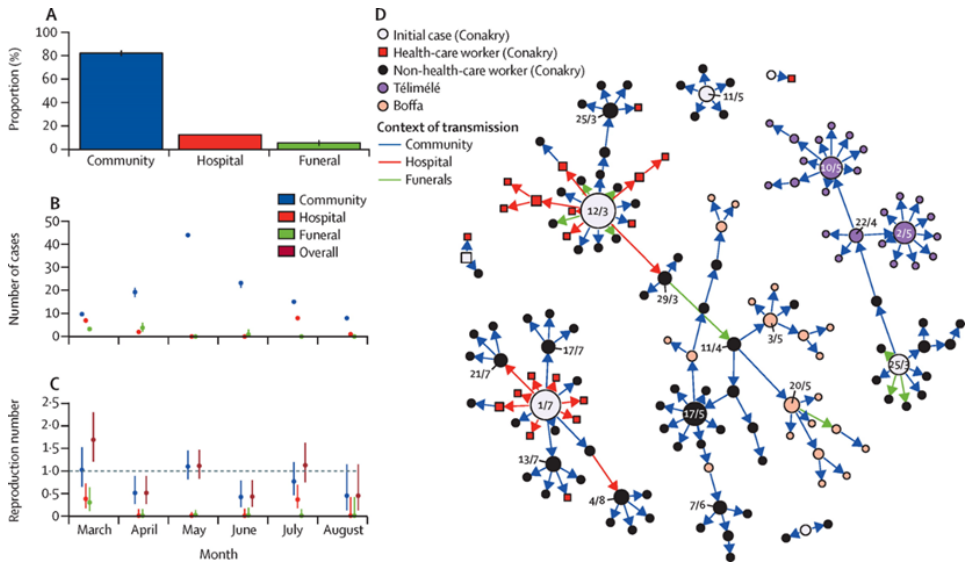
38. ábra. A szervezeti együttműködés hálózata a 2001. szeptember 11-i terrortámadás után az USA-ban (Forrás: https://ojs.tripaledu.com/index.php/jefa/article/view/33/32. Letöltés ideje: 2018. augusztus 1.)

Egy másik tipikus példa a hálózatelemzés alkalmazására a Morselli et al. (2013) cikkében leírt igazságszolgáltatási témájú esettanulmány, amelyben hálózati módszerrel jósolják meg a várható ítéleteket és a büntetések időtartamát. Az ítéletet és a büntetések várható időtartamának becslése már a rendőrségi intézkedések kezdetén is fontos lehet. A bemutatott példa a Caviar Network nevű kábítószer-terjesztő hálózat esete. A hálózaton belüli kapcsolatok elemzése komoly segítséget nyújthat a későbbi büntetések előrejelzésében. A lényeges hálózati ismérvek: kik a központi személyek a hálózatban, milyen a csoportszerkezet, milyen a csoportok közötti interakció, a teljes hálózat szerkezete, mi a hatása egy adott személy kivételének a hálózatból, melyek a hálózat információs csatornái. A kriminológusok számára egy bűnözői hálózat elemzése szintén komoly haszonnal jár, jól azonosíthatók a központ, az irányító személyek és a kevésbé fontos brókerek.



39. ábra. A Caviar Network bűnszervezet hálózati kommunikációs gráfja. A pirossal jelzett személyeket már elítélték. (Forrás: https://link.springer.com/article/10.1186/2190-8532-2-4. Letöltés ideje: 2018. augusztus 1.)

A hálózatelemzés igen fontos szerepet játszik a fertőző betegségek terjedésének leírásában és a járványok kialakulásának megakadályozásában. Az egészséges, de nem immunis, a fertőzött és az immunis személyek egy hálót alkotnak (SIR model – Susceptible to the disease, Infected, Recovered and immun). A fertőzés terjedése a fenti személyek kapcsolatrendszerétől függ. A 40. ábra a 2014-es ghánai Ebola-fertőzés terjedését mutatja. A 130 közeli fertőzött személy 90%-a csak egy forrással érintkezhetett az ábra szerint.



40. ábra. A 2014-es ghánai Ebola-fertőzés terjedése
(Forrás: Chains of transmission and control of Ebola virus disease in Conakry, Guinea, in 2014: an observational study O Faye, PY Boëlle, E Heleze, O Faye… – The Lancet Infectious …, 2015 – Elsevier. 2015 15, 320–326, Elsevier.) [92]

A hálózatok elemzése

A sokféle létező és működő hálózat, mint a közigazgatási vagy gazdasági szervezetek kapcsolatrendszerei, a járványok terjedési hálózata, az online közösségi hálók, a webes oldalakból és hiperlinkekből álló WWW, a routerekből, telekommunikációs csatornákból és számítógépekből álló internet vagy egy közösségi háló, amely emberekből és az őket összekapcsoló társadalmi viszonyokból áll – nagyon hasonló tulajdonságokkal rendelkezik (Barabási, 2017).

A hálózatok szerkezetének ismerete az élet számos területén lehetővé teszi a szervezetek és az egyének hatékonyabb működését és gyorsabb információcseréjét. A hálózatok kutatása a gráfelmélet eszközeivel történik. A továbbiakban szükségünk lesz az alábbi gráfelméleti fogalmakra:

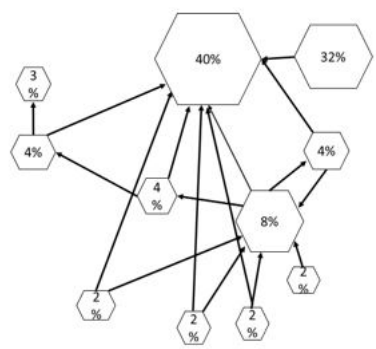
- A gráf csúcsok és élek halmaza. Egyes csúcsokat élek kötnek össze, míg más csúcsokat nem.
- Az egyszerű gráfban (és a továbbiakban csak ilyenekkel foglalkozunk) két csúcsot legfeljebb egy él köt össze.

92 Licence: Except as otherwise provided in any additional terms for a Service, you may print or download Content from the Services for your own personal, non-commercial, informational or scholarly use, provided that you keep intact all copyright and other proprietary notices.

- Egy csúcs fokának a rá illeszkedő élek számát nevezzük.
- Két csúcs szomszédos, ha van egy közös élük.
- Két, különböző csúcs közötti élek sorozatát útnak nevezzük, ha az egyik csúcsból a másikra az élek sorozatán keresztül el lehet jutni.
- Két csúcs közötti legrövidebb útnak azt az utat nevezzük, amely a legkevesebb élt tartalmazza.
- Összefüggő gráfnak nevezzük azt a gráfot, amelynek bármely csúcsából vezet út bármely más csúcsába.

A fenti gyakorlati alkalmazások esetén azt keressük, hogy kik a gráf központi szereplői, kik továbbítják az információt (a fertőző betegségek esetén kik terjesztik a járványt) és kik azok, akik a hálózatban izoláltak, ill. a periférián helyezkednek el. A gráfokban az alábbi módon értelmezhetjük egy csúcs fontosságát, ún. központiságát. Lássunk néhány példát:

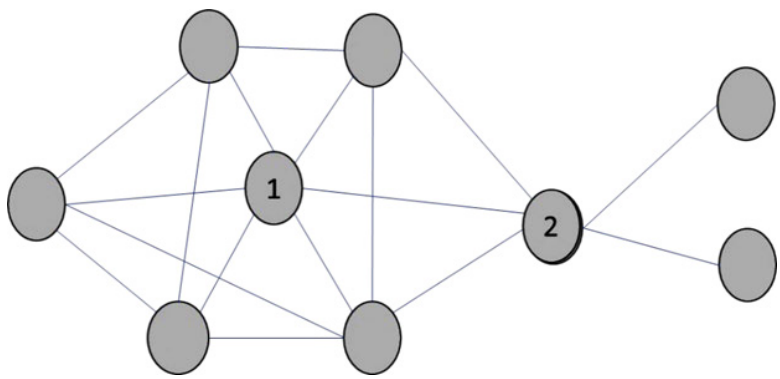
1. Mindnyájan használjuk a Google keresőjét. A keresőprogram alapja a PageRank algoritmus (Brin et al., 1998), amely egyébként a Google üzleti sikerének is a megalapozója. A PageRank (41. ábra) azt mutatja meg, hogy egy weboldal mennyire releváns egy adott keresésnél. A relevanciát a weboldalra mutató linkek számával és minőségével mérik, a magasabb relevanciájú találatok az értékesebbek, ezeket teszi a Google a találati lista elejére. Egy link minőségét azzal mérik, hogy a *hivatkozó* oldal mennyire releváns. A relevanciát úgy is felfoghatjuk, mint annak a mértékét, hogy a weboldal mennyire foglal el központi helyet, mennyire *centrális*:



41. ábra. Az egyes oldalak – gráf csúcsok – relevanciájának mértéke a Page Ranking algoritmus szerint. A relevancia mértékét a csúcs mérete szemlélteti

2. Az internet nem más, mint útvonalválasztók (routerek), számítógépek és kommunikációs csatornák által alkotott gráf. Az útvonalválasztók és a számítógépek a gráf csúcsai, a kommunikációs csatornák az élek. Azok a csúcsok, amelyeken több út megy át, jelentősebb szereppel bírnak a hálózat vezérlésében, mint azok, amelyeken kevesebb. Kiemelten kezeljük a csúcson átmenő legrövidebb utakat, és azokat a csúcsokat, amelyeken keresztül a legtöbb legrövidebb út halad át, a többi csúcs elérhetősége és irányíthatósága szempontjából központinak tekintjük.

3. Egy emberi közösségi kapcsolatrendszerben feltehetjük azt a kérdést, hogy hány ismerősön keresztül juthatunk el egymáshoz. Karinthy Frigyes így ír *Láncszemek* című novellájában 1929-ben: „Tessék egy akármilyen meghatározható egyént kijelölni a Föld másfél milliárd lakója közül, bármelyik pontján a Földnek [...] legföljebb öt más egyénen keresztül, kik közül az egyik neki személyes ismerőse, kapcsolatot tud létesíteni az illetővel, csupa közvetlen – ismeretség – alapon."Karinthy a novellában példákkal illusztrálja az állítást, melyekben az a közös, hogy mindegyik ismeretségi lánc eleme egy-két, sokak által jól ismert, központi személy, mint pl. a svéd király vagy Henry Ford. Karinthy állítása a modern kutatások fényében is teljes mértékben helytálló, a kapcsolatok jelentős része egy-egy központi csúcson keresztül működik. A nem közösségi hálók esetén is fontos gyakorlati kérdés, hogy hány lépésben lehet eljutni egy csúcsból egy másikba, és vannak-e olyan csúcsok, amelyek ebben kitüntetett szerepet játszanak, amelyek több csúcshoz vannak közel, mint mások (42. ábra).



42. ábra. A gráfban az 1-gyel jelölt csúcs fokszám-központisága a legmagasabb, a 2-vel jelölt csúcs közöttiség mértéke a legnagyobb

Definíciók:

1. Egy csúcs *fokszám-központiságának* (angolul: degree-centrality), a csúcs fokszámának a gráf fokszám-összegéhez viszonyított arányát nevezzük. A központiság azt fejezi ki, hogy mely csúcsnak van sok közvetlen kapcsolata szomszédos csúcsokkal.
2. Egy csúcs *közöttiség* mértékének (angolul: betweenness centrality) a csúcson átmenő legrövidebb utak számának és a csúcson átmenő legrövidebb utak végpontjait összekötő összes út számának hányadosát értjük. A közöttiség a közvetítő szerepre jellemző mérőszám, azt feltételezve, hogy a kommunikáció, az adatáramlás vagy éppen a fertőzés a legrövidebb úton megy végbe.
3. Egy csúcs normalizált *közeliség* (angolul: closeness) mértéke a csúcs és a gráf összes többi csúcsa közötti legrövidebb út átlagos hossza. Minél kisebb a csúcs közelisége, annál közelebb van a többi csúcshoz.

Egyéb mértékeket is használnak a hálók szerkezetének és csúcsainak jellemzésére, az egyszerűség kedvéért azonban a továbbiakban csak a fentiekkel foglalkozunk.

Hogyan alkalmazhatjuk a gyakorlatban a központiság mértékeit?

Egy közösségi hálózat legmagasabb fokszám-központiságú csúcsai hatékony segítséget nyújthatnak az információterjesztésben. A közösségi hálózat, mint marketing csatorna, akkor a leghatékonyabb, ha a marketingesek tudják, mely pontokra kell eljuttatni az információt ahhoz, hogy annak a legnagyobb hatása legyen.

Az egészségügyben a központi csúcsok a járvány terjedési hálójának központjai (ld. 42. ábra), elszigetelésük lassítja vagy megszünteti a járvány terjedését.

Egy számítógépes hálózat esetén a magas központiságú gépek fokozott védelme magasabb biztonságot nyújt, mint ha ugyanazokat az erőforrásokat az összes hálózati gép védelmére fordítanánk.

Egy szervezetközi együttműködésben a magasabb központiságú szervezeteknek a súlya magasabb, célszerű az erőforrásokat is eszerint elosztani.

A hálózatelemzés számítástechnikai kérdései

Nagy, sok millió csúcsot és élet tartalmazó hálózatok esetén a csúcsok központiságának kiszámítása a nagy méret miatt speciális, big data jellegű számítástechnikai megoldásokat tesz szükségessé. Ha a gráf egymillió csúcsot tartalmaz, akkor a csúcsok közötti élek leírásához sok milliárd adat is szükséges lehet. A Google keresője napi sok milliárd honlapot indexel és számítja ki a PageRanket a 2. fejezetben ismertetett eszközökkel.

10. AZ OSZTÁLYOZÓ ÉS A PREDIKTÍV GÉPI TANULÓ RENDSZEREK PONTOSSÁGA ÉS HATÉKONYSÁGA

Hatékonyságvizsgálat

A gépi tanuló algoritmusok alkalmazásának esetén csak az esetek egy részében van lehetőség arra, hogy hagyományos módon becsüljük meg a kapott modell megfelelőségét. Egy hagyományos lineáris regressziós modell esetén hipotézist állítunk fel arra, hogy az *x* magyarázó (független) változóknak nincs hatása az *y* magyarázott (függő) változóra. A legfontosabb értékelési eljárás ez esetben az a hipotézisvizsgálat, ahol a kiinduló, azaz H_0 hipotézis az, hogy az

$$y = \beta_0 + \beta_1 x_1 + \varepsilon$$

egyenlet B_1 paramétere nulla. Emellett feltételeztük, hogy az *y(x)* függvény lineáris, az hibaparaméter nulla várható értékű normális eloszlású valószínűségi változó. A hipotézist a meglévő adatokból statisztikai számításokkal tudjuk elvetni vagy nem elvetni egy meghatározott konfidenciaszint mellett. Ha nem sikerül a hipotézist elvetni, akkor úgy gondoljuk, hogy a független változónak van hatása a függő változóra, és a hatás mértékét a regressziós egyenletben a megfelelő koefficiens mutatja. A hipotézisvizsgálathoz azonban meghatározott, például a fentiekben leírt statisztikai feltételeknek kell teljesülniük.

A gépi tanulási algoritmusok alkalmazásakor azonban a legtöbb esetben nem ismerjük annyira az adatokat, hogy garantálni tudjuk, hogy előre meghatározott statisztikai feltételek teljesüljenek, vagy az adatok mennyisége és strukturálatlansága miatt nem is tudnánk lefolytatni azokat a statisztikai teszteket, amelyek bizonyítanák egy-egy feltétel meglétét vagy hiányát. Nehéz lenne például tweetek osztályozásához bármilyen stabil statisztikai jellemzőket meghatározni. Egy ajánlórendszer esetén az újonnan belépő termékek és vásárlók folyamatosan módosíthatják a korábbi termékek jellemzőinek statisztikai tulajdonságait.

A gépi tanuló algoritmusok hatékonyságát sokféleképpen mérik, nincs egyetlen, általános kritérium. Az, hogy milyen kritériumot alkalmazunk, függ a konkrét feladattól. Például, ha a bankkártyával történő visszaélések észlelésre készítünk egy klasszifikációs modellt, egy tranzakció visszaélés kategóriába sorolása, ha az nem visszaélés, kevesebb kárt okoz, mintha egy visszaélést nem észlel a rendszer.

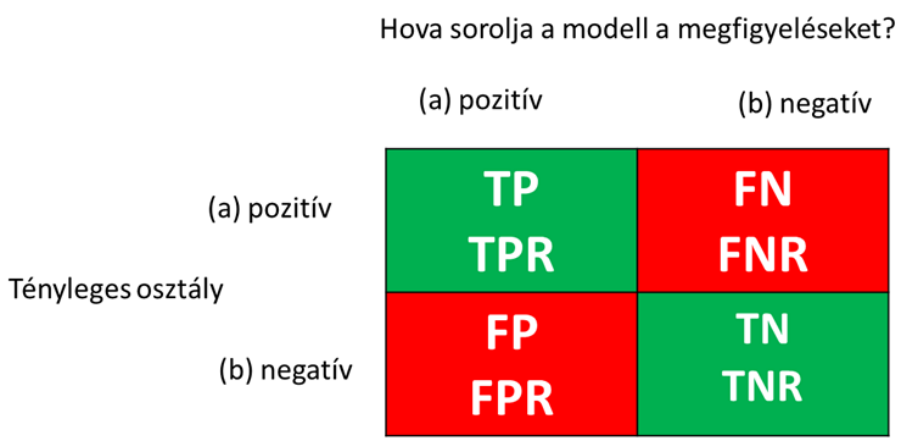
Az alábbiakban bemutatjuk a leggyakrabban használt teljesítménykritériumokat.

A konfúziós mátrix

A konfúziós mátrixot mind a klasszifikációs, mind a prediktív modelleknél alkalmazzuk. A klasszifikáció a gépi tanulás egyik alapfeladata, a prediktív modellek egy része is ebbe a kategóriába esik. Például, az ajánlórendszerek is felfoghatók klasszifikációs modellként, a termékek feloszthatók ajánlott és nem ajánlott termékekre, vagy erősen ajánlott, közepes mértékben ajánlott és nem ajánlott termékekre, az asszociációs modelleknél az összetartozó objektumokat osztályoknak is tekinthetjük. A konfúziós mátrix alkalmazása szélesebb körű, mint az osztályozó algoritmusok jellemzése.

A klasszifikáció során az objektumokat osztályokba soroljuk, a konfúziós mátrixszal azt mérjük, hogy tanulás során kialakított modell hány helyes és hány helytelen besorolást végez. A 43. ábra egy-két (pozitív és negatív) osztályba sorolás helyes és helytelen találatait összegzi. A jelölések:

- TP true positive – a célosztályba sorolt és valóban odatartozó objektumok száma,
- TN true negative – a nem célosztályba sorolt és valóban nem a célosztályba tartozó objektumok száma,
- FP false positive – a célosztályba sorolt, de valójában nem odatartozó objektumok száma,
- FN false negative – a nem célosztályba sorolt, de valójában odatartozó objektumok száma,
- TPR, TNR, FPR, FNR – TP, TN, FP, FN aránya.



43. ábra. A bináris osztályozás konfúziós mátrixa

Kvantitatív mutatók

A konfúziós mátrixból többféle komplex kvantitatív mutatót számolunk[93]:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Error\ rate = \frac{FP + FN}{TP + TN + FP + FN} = 1 - Accuracy$$

Az *Accuracy* a helyes találatok arányát mutatja az összes objektumhoz viszonyítva, míg az *Error rate* a helytelen találatok arányát az összes objektumhoz képest.

Az *Accuracy* és az *Error rate* nagyon általános mutatók, az alkalmazási terület sajátosságaira vonatkozóan nem mindig kellően informatívak. Nem informatív például az előbbi visszaélés detektálási feladatnál említett, a fals pozitív és a negatív találatok viszonyára vonatkozóan. Nem használható e két kritérium akkor sem, ha a célosztály mérete kicsi a teljes halmazhoz képest. Például egy ritka betegség esetén, ha 100 ezerből 10 személy tartozik a becsült osztályba, akkor is 99.99%-os *Accuracy*t kapunk, ha egyszer sem találjuk el a célosztályt.

$$Sensitivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{FP + TN}$$

A *Sensitivity*t valódi pozitív rátának is nevezik, mert a célosztályba helyesen besorolt objektumok arányát fejezi a ki az összes, valóban a célosztályba tartozó objektumhoz képest.

A *Specificity* pedig a valódi negatív ráta, mert a nem célosztályba helyesen besorolt objektumok arányát fejezi ki az összes valóban nem a célosztályba tartozó objektumhoz képest.

A *Sensitivity* a besorolás „agresszivitását” fejezi ki. Egy klasszifikáció „agresszív”, ha minden, kicsit is „gyanús” megfigyelést a célosztályba sorol (pl. egy e-mail-folyam spam/nem spam osztályozása, ahol minden, kicsit is spamgyanús levelet a modell a spam kategóriába sorol), vagy nagyon konzervatív, amikor túl sok objektum (spam) átmegy a szűrőn.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

A *Precision* és a *Recall* mutatók jelzik, hogy a modell által adott besorolás mennyire érdekes és releváns, vagy az eredmény nem különbözik lényegesen egy véletlenszerű osztályozástól.

93 Az angol megnevezéseket többféleképpen is fordítják magyarra, a pontosság érdekében a kritériumok angol nevét használjuk.

A *Precision* a modell predikciós értéke, azt mutatja, hogy a modell mennyire korrektül sorol be egy pozitív megfigyelést a pozitív célosztályba. Ha ez az érték alacsony, akkor túl sok a tévedés, pl. egy keresőmotor esetén.

A *Recall* azt mutatja, hogy az eredmény mennyire teljes. Értéke ugyanaz, mint a sensitivity értéke, csak más az értelmezése. Ha magas az értéke, akkor sok pozitív megfigyelés kap korrekt besorolást.

Például, egy keresőmotor sok jó találatot gyűjt.

A *Kappa statisztika:*

$$K = \frac{p_0 - p_e}{1 - p_e}$$

Ahol p_0 az *Accuracy*, p_e pedig annak valószínűsége, hogy egy kiválasztott megfigyelést véletlenszerűen a célosztályba sorolunk.

Minthogy $p_0 \leq 1$, $K \leq 1$. Ha K = 1, a modell tökéletes. Ez azt jelenti, hogy az *Accuracy* = *1*, vagyis az osztályozás tökéletes. *K = 0* esetben az osztályozó modell nem jobb, mint egy véletlen osztályozás.

Az egyes K értékekhez a szakértők minőségi kategóriákat rendelnek:

- Gyenge megfelelőség: K < 0.20
- Elégséges megfelelőség: 0.2 < K < 0.4
- Elfogadható megfelelőség: 0.4 < K < 0.6
- Jó megfelelőség: 0.6 < K < 0.8
- Kiváló megfelelőség: 0.8 < K < 1

A McNemar-statisztika

A McNemar-statisztikával azt vizsgálják, hogy van-e különbség két kategóriaváltozó között a χ^2-teszt alkalmazásával. Az egyik változó a valós osztályozás, a másik a modell. Ehhez egy 2 x 2-es kontingencia táblázatot hozunk létre a minta gyakoriságaiból és elvégezzük a χ^2 tesztet.

A konfúziós mátrix:

	modell besorolása	
valóság		
	TP	FN
	FP	TN

Alkalmazzuk a χ^2-tesztet, kiszámítjuk a

$$T = \frac{(FN - FP)^2}{(FN + FP)}$$

statisztikát és összehasonlítjuk az 1 szabadságfokú χ^2 eloszlás értékével adott szignifikancia mellett. Általában az α = 0,05 értéket használjuk. Ha a T értéke nagyobb, mint a χ^2 értéke, akkor nem vetjük el a nullhipotézist, vagyis a modell nem ad jobb besorolást, mint a véletlen.

Vizsgáljuk, hogy a modell által pozitívnak besorolt és a valóságban pozitív objektumok száma egyenlő-e és ugyanígy egyenlő-e a valós negatív objektumok száma a modell által negatívnak besoroltak számával, vagyis TP + FN = TP + FP és FP + TN = FN + TN.

Nézzünk egy számpéldát:

	FN=22
	TN=35

$$T = \frac{(FN - FP)^2}{(FN + FP)} = \frac{(22 - 15)^2}{22 + 15} = 1.32$$

$$\chi^2_{\alpha=0,05}=3,84$$

$$1,32 \leq 3,84$$

A nullhipotézis az, hogy a sorösszegek megegyeznek az oszlopösszegekkel. A hipotézist ezen a szinten nem utasítjuk el.

A McNemar-teszt nem feltételezi a változók meghatározott eloszlását, ugyanakkor kis elemszámoknál a χ^2-teszt nem használható.

Az F1 érték

A Precision és a Recall harmonikus átlaga, a pontosságot méri. Ha értéke 1, akkor a modell tökéletes, ha 0, akkor rossz.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Az F1 értéket gyakran használják a keresési, dokumentum klasszifikációs, lekérdezés klasszifikációs eljárásokban a teljesítmény mérésére.

A fenti kritériumokon kívül számos más mérőszámot is alkalmaznak a konkrét szakterület sajátosságainak megfelelően.

Az osztályozás megbízhatósága

A fenti mértékek valamilyen módon az osztályozó modell teljesítményét mérik, de fel kell tennünk azt a kérdést is, hogy vajon mennyire megbízható egy osztályozás. Ha például egy e-mail tartalmazza az „ingyenes” vagy a „fogyás” kifejezéseket, 99%-os eséllyel spam és az osztályozó ide is sorolja. Ha tartalmazza a „konferencia” kifejezést, csak 51% az esélye, hogy spam, de a spamszűrő ezt is kiszűri. Ilyenkor célszerű a 0–1 kategóriák helyett az egyes osztályokba tartozás valószínűségeit számolni (44. ábra).

Objektumok	P(nem tartozik a célosztályba)	P(célosztályba tartozik)
1	0.0808272	0.9191728
2	1.0000000	0.0000000
3	0.7064238	0.2935762
4	0.1962657	0.8037343
5	0.8249874	0.1750126
6	1.0000000	0.0000000

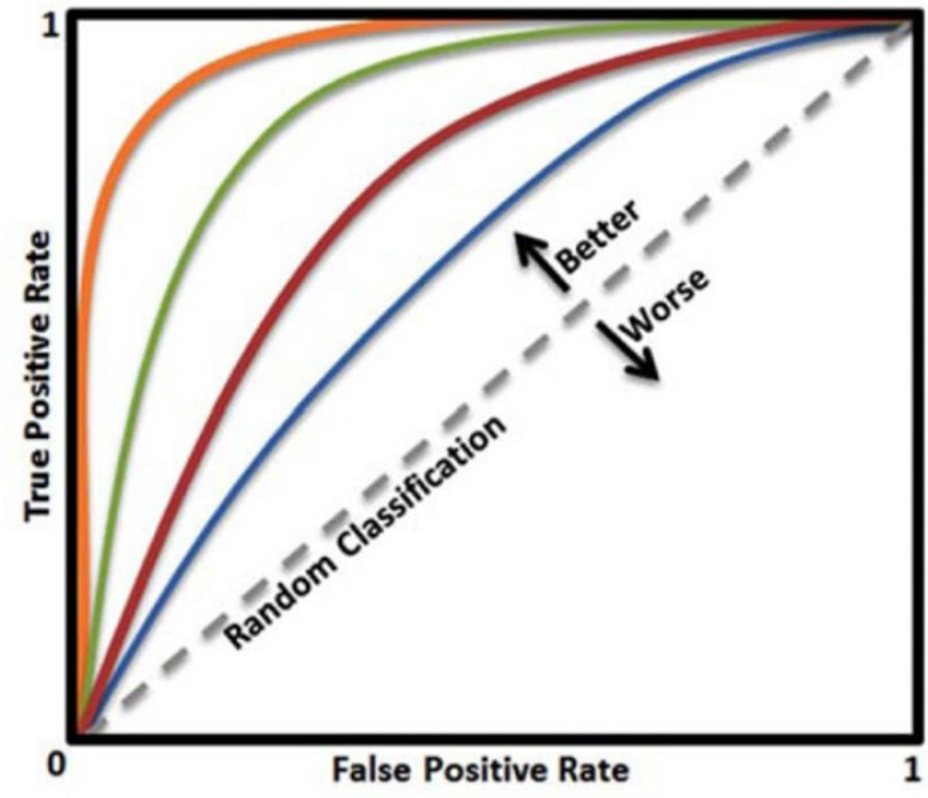
44. ábra. Az osztályba tartozás valószínűsége

A ROC-görbe

A ROC (Receiver Operating Characteristic) a valódi pozitívak aránya és a fals pozitívak aránya közötti kapcsolatot jellemzi. Az elnevezést a radartechnológiában vezették be, a valódi szignálok és a fals riasztások megkülönböztetésére szolgál.

Minthogy a tengelyeken használt mértékek ekvivalensek a *Sensitivity*vel és az (1-*Specificity*)-vel, *sensitivity/specificity* diagramnak is nevezik.

A görbe azt mutatja, hogy a független változók értékének, mint döntési paramétereknek változtatásával hogyan függ a *Sensitivity* a kiesők arányától (fall-out). Ha a változók nominálisak, a ROC-görbében a valószínűséget lehet használni. Ha a valódi pozitív besorolások és a fals pozitív besorolások valószínűségi eloszlása ismert, akkor a ROC-görbe a kumulatív eloszlásfüggvény. A görbe alatti terület (AUC – area under curve) a 0-tól egy felső határig a helyes besorolás valószínűsége.



45. ábra. A ROC-görbe

A 45. ábrán a 45°-os egyenes a random klasszifikációt jellemzi, ha egy modell ROC-görbéje ez alatt van, akkor a modell rosszabb osztályozást ad, mintha az objektumokat random módon sorolnánk be. Minél távolabb van a ROC-görbe a 45°-os egyenestől a felső háromszögben, annál hatékonyabbnak tartjuk a modellt.

A modell megfelelőségére jellemző az AUC (area under the curve) is, azaz a ROC-görbe alatti terület. Ha az AUC értéke 0.6, vagy kisebb, a modell gyakorlatilag nem ad jobb eredményt, mint a random osztályozás. Úgy tekintjük, hogy a modell jól használható, ha az AUC-érték 0.9–1 között van.

IRODALOMJEGYZÉK

1. Aggarwal, C. C. (2016). *Recommender Systems. The Textbook.* Springer International Publishing Switzerland.
2. Agarwal, P. K., Har-Peled, S., Varadarajan, Kasturi R. V. (2005). Geometric approximation via coresets. In Goodman, J. E.; Pach, J.; Welzl, E.: *Combinatorial and Computational Geometry.* Mathematical Sciences Research Institute Publications, 52, Cambridge Univ. Press, Cambridge, 1–30.

3. Almeida T. A., Gómez Hidalgo, J. M., Silva, T. P. (2013). Towards SMS Spam Filtering: Results under a New Dataset. *International Journal of Information Security Science (IJISS)*, 2(1), 1–18.

4. Anastasopoulos, L. J., Badani, D., Lee, C., Ginosar, S., Williams, J-R. (2017). Political image analysis with deep neural networks https://scholar.harvard.edu/files/janastas/files/neural-networks-preprint.pdf

5. Anderson, C. (2004). The Long Tail publ. Wired. https://www.wired.com/2004/10/tail/

6. Barabási A-L. (2017). A hálózatok tudománya. Libri Könyvkiadó.

7. Barbuceanu M., Fox, M. S. (1996). Coordinating multiple agents in the supply chain, in 'Proceedings of the Fifth Workshop on Enabling Technology for Collaborative Enterprises (WET ICE'96), Stanford University, CA. 134–141.

8. Bifet, A., Holmes, G., Pfahringer, B., Read, J., Kranen, P., Kremer, H., Jansen, T., Seidl, T. (2018). MOA: A Real-Time Analytics Open Source Framework. Available from: https://www.researchgate.net/publication/220699253_MOA_A_Real-Time_Analytics_Open_Source_Framework

9. Breese J.S., Heckerman, D., Kadie, C. (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In Uncertainty in Artificial Intelligence. Proceedings of the Fourteenth Conference, 43–52.

10. Brewer, E. (2012). CAP Twelve Years Later: How the „Rules" Have Changed. https://www.infoq.com/articles/cap-twelve-years-later-how-the-rules-have-changed .