|  |  |
| --- | --- |
| Miskolci Egyetem  Gépészmérnöki és Informatikai Kar  Általános Informatikai Intézeti  *3515 Miskolc-Egyetemváros* |  |

**Diplomamunka**

*Feladat címe:*

Mesterséges intelligencia alapú oktatást támogató alkalmazás tervezése

*Készítette:*

Szilvási Péter

*MSc szintű mérnökinformatika szakos*

*Alkalmazásfejlesztő szakirányos hallgató*

*Témavezető:*

Dr. Kovács László

*tanszékvezető, egyetemi tanár*

**Miskolc, 2020.**

# Bevezetés

## Téma választás ismertetése

placeholder

## Miért kell támogatni az oktatást?

placeholder

## Oktatás fontossága

placeholder

## Mesterséges intelligencia és az oktatás

placeholder

# Gépi tanulás menete

**Gépi tanulás**

Ha elmélyülünk a gépi tanulás definíciójában, akkor azt találjuk, hogy „*A gépi tanulás az algoritmusok és statisztikai modellek tudományos vizsgálata, amelyet a számítógépes rendszerek egy adott feladat végrehajtásához használnak kifejezett utasítások nélkül, mintákra és következtetésekre támaszkodva*”. [1]

Ha elemezzük az előző definíciót, találunk néhány kiemelt szempontot. Hatalmas tudományos kutatás és erőfeszítés van a növekedés és fejlődés támogatására.

Algoritmusokra és statisztikai modellekre támaszkodik, amelyek azt mondják nekünk, hogy ha el akarjuk sajátítani a gépi tanulást, akkor az algoritmusok, a statisztikák és a valószínűségek átfogó megértését kell kialakítanunk. Célja egy meghatározott feladat végrehajtása, amely elmondja, hogy a gépi tanulási megoldásnak van egy bizonyos hatóköre, például előrejelzés és ajánlás.

Kifejezett utasítások nélkül, ami azt mondja nekünk, hogy a gépi tanulási megoldást nem szabad pontosan beprogramozni a dolgok megtanítására. Mintákra és következtetésekre támaszkodik. Tehát a gépi tanulási megoldásunk a történeti adatokból bizonyos mintákra támaszkodik a helyes működés megtanulásához.

**Megoldható problémák gépi tanulással** [2]

Most tárgyaljuk meg a gépi tanulás néhány érdekes, valós alkalmazását a mindennapokban.

* A képfelismerés. Hogyan tudja a mobiltelefon feloldani magát azzal, hogy ha a gazdája megmutatja az arcát a kamerának, miközben nem oldja fel magát más emberek számára? Egy képfelismerési technológiával érik el ezt a funkcionalitást. Ahhoz, hogy fel tudjuk ismerni a kívánt objektumot, a képi adatok (pixeleket) egy gépi tanulási algoritmuson kell kiterjeszteni.
* A rosszindulatú programok észlelése, ahol arra vagyunk kíváncsik, hogy az antivírusok hogyan tudják hatékonyan megismerni az új vírusokat, mielőtt frissítéseket kapnának. Ismét a gépi tanulás a motor, amely ezt működteti.
* A hangfelismerés. Egy másik jól ismert példa, amely a gépi tanulást használja. Legyen szó mobiltelefonról vagy okos otthonról, a gépi tanulás lehetővé teszi a felhasználó hatékony azonosítását.

A gépi tanulás terén szerzett kompetenciák és készségek fejlesztésével fantasztikus megoldásokat tudunk kidolgozni, amelyek a valós problémákkal foglalkoznak és értéket képviselnek az emberiség számára.

Végül érdemes megvitatni, hogy a gépi tanulás miért vált manapság ilyen vonzóvá. Nem is gondolná az ember, hogy már a 60-as évek óta létezik. Azt állíthatjuk, hogy az alapstatisztika és a valószínűség valamiféle gépi tanulás. Az elmúlt években azonban volt néhány mozgatóerő.

**Adat túlcsordulás**

Először is, a problémák nagy méretűvé váltak: ***nagy dimenziószám***. A sok dimenzióval rendelkező adat, amelynek túl sok jellemzője van, ami szinte lehetetlenné teszi a hagyományos programozással történő elemzést. A nagy dimenziós adatra példa a páciens egészségi állapota, ahol számos jellemző, például az immunrendszer állapota, a genetikai háttér, a táplálkozás, az operációk, a drog- és dohányfogyasztás.

Hatalmas adat túlcsordulás van, például hírcsatornák, webszolgáltatások, adatbázisok, e-mailek, felmérések adatforrásain. A vállalkozások a közelmúltban észrevették, hogy hatékony módszert kell találni ezen hatalmas adatok megértésére, amelyekből értéket kell teremteni.

A számítástechnika fejlődése, valamint a számítási és tárolási erőforrások folyamatos csökkenése, például a felhőalapú számítástechnikában hatékonyabbá tette a gépi tanulási algoritmusok megvalósítását. Általában a gépi tanulási algoritmusok erőforrás-igényesek. A kutatások jelentősen megnövekedtek az elmúlt évtizedben. A vállalkozások elkezdték megérteni ennek értékét, ezért számos algoritmust kutattak és fejlesztettek ki.



**1. ábra: A gépi tanulás mozgatórugói** [3]

Tehát, figyelembe véve korábbi információkat, magabiztosan kijelenthetjük, hogy:

*„Az adat az új üzemanyag.”* [4]

## Miben különbözik a gépi tanulás?

Nos, egy jó kérdés merülhet fel bennünk, hogy miért nem programozzuk be üzleti szabályainkat és tartomány (domain) ismeretünket a gépi tanulási modelljeinkbe úgy, mint a hagyományos programozásnál? Más szóval, mi a tényleges különbség a gépi tanulás és a hagyományos programozás között?

**Hagyományos programozás és gépi tanulás** [5]

A hagyományos programozás során a teljes tartományú üzleti logikát olyan üzleti szabályok formájában rögzítjük, amelyek kifejezetten programozhatók a kódunkba. Például, ha egy adott vállalat fizetését szeretnénk kiszámítani, akkor minden forgatókönyvre világos üzleti szabályunk lesz, és pontosan tudjuk, hogy mi fog történi. Teljes mértékben megértjük az üzleti területet.

Míg a gépi tanulás során nem teljesen értjük a tartománymodellt, valamennyire tudjuk, hogyan reagál a rendszer bizonyos bemenetekre és kimenetekre. Nem igazán értjük teljesen a mögöttes bonyolult kapcsolatokat a bemenet és a kimenet között. Például egy gépi tanulási algoritmus a vásárlási viselkedés hasonlósága alapján különböző csoportokba sorolhatja az ügyfeleket, annak ellenére, hogy nem értjük teljesen az alapul szolgáló komplex összefüggést. A feladat egy gépi tanulási algoritmusra van delegálva.

A hagyományos programozásban a rendszer kidolgozott. A változtatásokat kifejezetten a forráskód vagy a konfiguráció megváltoztatásával kell végrehajtani. Például, ha új szabályt szeretnénk hozzáadni a fizetési rendszerünkhöz, módosítanunk kell a forráskódot. De a gépi tanulásban a rendszer dinamikus és folyamatosan tanul, és a bemenetek és az új tanúsítványok alapján igazodik. Az ügyfelek ápolására szolgáló algoritmusunk idővel jobbá válhat, mivel több trendet tanul meg.

Egy másik különbség az, hogy a hagyományos programozásnál nincs szükségünk semmilyen történelmi adatra vagy tanulási fázisra. A logika pontosan tudja, mit kell tennie. Bérszámítási rendszerünk megfelelően működhet anélkül, hogy úgymond képzési fizetést biztosítanánk. Míg a gépi tanulás során sok történelmi adatra van szükségünk ahhoz, hogy tanulhassunk belőle. Például ügyfeleink megfelelő osztályozásához, a gépi tanulási algoritmusunknak elegendő mintát kell látnia az alapul szolgáló kapcsolatok megértéséhez. Ügyfélápoló algoritmusunknak történelmi adatokra van szüksége, hogy tanuljanak belőlük. Az embernél is hasonló a tanulás, múltbeli eseményekből vonnunk le következtetéseket, majd próbáljuk megjósolni a jövőt.

**Keretrendszerek [Citation from Python]**

A hagyományos programozásban a logika egyszerű és egyértelmű. Könnyen elolvashatjuk a forráskódot, hogy megtaláljuk az igazságot. A gépi tanulás homályos és bonyolult bizonyos matematikai algoritmusok alapján. Általában könyvtárak vagy olyan keretrendszerek, például a scikit-learn, tensorflow és a Keras segítségével elvonatkoztathatunk a gépi tanulás mögötti bonyolultságtól. A keretrendszerek absztrakciós rétegek, amelyek hozzájárulnak a gyors és egyszerű fejlesztéshez.

<<Kibővíteni, többet a keretrendszerekről>>

<<Összehasonlítani egyik keretrendszert a másikkal>>

## Főbb tanulási típusok

Miután megismertük, hogy mi a gépi tanulás és miben különbözik a tradicionális programozási megközelítéstől, fontos tisztában lenni a gépi tanulási algoritmusok különböző típusaival. Minden algoritmuscsoportnak megvan a maga módszere a tanulásra. Alapvetően három csoportot különíthetünk el: a felügyelt tanulási algoritmusok, a felügyelet nélküli tanulási algoritmusok és a megerősítő tanulási algoritmusok. Nézzük meg, hogyan működik mindegyikük!

**Felügyelt tanulás** [6]

Kezdjük a felügyelt tanulással. A felügyelt tanulási algoritmusok úgy működnek, hogy egy adott bemenetre () egy megfelelő kimenetet () eredményez. A bemeneti érték egy függvényen () keresztül leképezésre, ami egy kimeneti értéket produkál.

Tehát a kimenet függ a bemenettől. A cél az, hogy felfedezzük a (bemenet) és (kimenet) közötti összefüggéseket. Az függvényt leképezési függvénynek (mapping function) is nevezik, ami leírja az és kapcsolatát.

Vegyünk egy konkrét példát. Tegyük fel, hogy van egy olyan üzletünk, ahol van vásárlás szám, vásárlás dátuma és elköltött összege. Felügyelt gépi tanulással szeretnénk megtudni a várható értékesítési és vásárlói típusokat. A felügyelt algoritmus azokat a történelmi adatokat veszi fel, amelyek tartalmazzák az bemenetet. Azaz a vásárlások számát, a vásárlás dátumát és az elköltött összegeket. Az kimenet a probléma tartomány, a várható értékesítés vagy ügyféltípus. A felügyelt gépi tanulási algoritmus rétegek, és ennek eredménye egy gépi tanulási modell. Ezt követően a betanult modellnek új inputot adhatunk, és az új input alapján előre jelezhetjük a jövőbeni értékesítési és vásárlói profilokat. A felügyelt tanulásban előzetesen tudjuk a helyes választ, ami azt jelenti, hogy megfelelő eladásokkal és kategóriákkal rendelkezünk, amelyeket algoritmusunk tanulhat és fejleszthet belőle.



A felügyelt gépi tanulási algoritmusok két jól ismert típusa a ***regressziós algoritmus***, amikor az előre jelzett érték folyamatos jellegű, bármi egy bizonyos tartományban. Az eladási előrejelzések bármilyen értéket feltételezhetnek, 50, 100, 150 vagy akár milliót.

A másik típus az ***osztályozási algoritmusok***, amikor az előre jelzett érték diszkrét jellegű. Szóval az előre jelzett érték lehetséges opcióival rendelkezik. Például egy kiskereskedelmi üzlet minősítheti az ügyfeleket kiemelt, közepes és normális vásárlóknak.

**Felügyelet nélküli tanulás** [7]

Felügyelet nélküli algoritmusok azok, amikor csak bemeneti adatok vannak, és nincsenek kimeneti adatok. Nincs alapfeltevés igazság, tehát az adott bemenetre nincs hozzátartozó kimenet. A cél általában az adatszerkezet megértése az adatok helyes felhasználása érdekében. Ezeket az algoritmusokat önszerveződő algoritmusoknak is nevezzük. Tegyük fel, hogy van egy bizonyos adatkészletünk, amelyet A – L betűkkel jelölünk, és szeretnénk megismerni ezeknek az adatkészlet-tagoknak a kapcsolatát vagy asszociációit egy felügyelet nélküli tanulási algoritmus segítségével. Az algoritmus a következő csoportot hozhatja létre az általa kitalált más kapcsolatok vagy asszociációk alapján. [A, I, J]; [B, K, H]; [C, H, F]; [E, G, L]. Az, hogy miként szervezzük ezeket a kapcsolatokat és asszociációkat, az általunk használt, felügyelet nélküli tanulási algoritmus típusától függ.

Vegyünk egy példát. Tegyük fel, hogy három különböző autók vannak, hétköznapi autók, sportautók és elektromos autók. Az autókat betöltjük egy felügyelet nélküli tanulási algoritmusba, amely kiszámítja az egyes autókkal kapcsolatos összefüggéseket. És akkor ebből lesz egy gépi tanulási modellünk. A gépi tanulási modell lehet jövőbeli autók számára és felhasználható a jövőbeli minták előrejelzésére. Ennek értelmében a felügyelet nélküli algoritmusok ahelyett, hogy a visszajelzésekre támaszkodnának, mint a felügyelt algoritmusok, a felügyelet nélküli algoritmusok a közös vonások azonosításán alapulnak. Akár képes felismerni az adatkészletek közötti különbségeket is, bizonyos tulajdonságok hiányát vagy jelenlétét.



A gépi tanulási algoritmusok egyik jól ismert típusa a csoportosító algoritmusok. Ezt a típusú algoritmust akkor használják, ha egy adott adatbevitelt csoportosítani kívánunk. Például az ügyfelek csoportok szerinti szegmentálása, hogy a hirdetések megfelelő módon címezzék. A klaszterező algoritmus felismeri az ügyfél jellemzőit, mint például az életkor, a lakóterület, a jövedelem. Számos különböző csoportosító algoritmus létezik, mint például a k-csoportosítás. A másik algoritmus csoport az asszociáció. Az ilyen típusú algoritmusok célja, hogy asszociációkat hozzanak létre nagy adatbázis adatelemei között. Célja a változók közötti hasznos kapcsolatok feltárása.

**Megerősítő tanulás** [8]

A gépi tanulási algoritmus harmadik típusa a megerősítő tanulás, más néven célorientált tanulás. Az ilyen típusú algoritmusok célja egy közvetítő előállítása, más néven az ágens. Tanulási algoritmusok, amelyek az időben változó jutalmak felhasználásával megtanulják a környezetükből tanult feladatot. Az ágens úgy dönt, hogy végrehajt egy műveletet a környezetben, majd tanul ebből a műveletből. A tanulás visszavezethető környezeti állapot és jutalom formájában. Ennek felhasználásával az ágens törli és frissíti belső szabályrendszerét, hogy egyre több tapasztalatot szerez. Az ilyen típusú algoritmusok célja a sok lépéses komplex problémák megoldása, például egy összetett játék megoldása. A megerősítő tanulási algoritmusok üres állapotból indulnak ki. A megfelelő jutalmak alkalmazásával fokozatosan javítják a teljesítményt. A megerősítő algoritmusokat általában olyan autonóm rendszerekben használják, amelyek emberi útmutatás nélkül hoznak döntéseket. A megerősítő algoritmusokat szintén fejlett témának tekintik.



Végül hasznos összehasonlítani és szembe állítani a tanulási típusokat két különböző dimenzióból: objektív és tanulási forrásból. A felügyelt algoritmusok célja a jövőbeli értékek vagy kategóriák előrejelzése, és tanulási forrásuk a kimeneti adatkészlet. A felügyelet nélküli algoritmusok célja az adatok rendszerezése az alapul szolgáló struktúra alapján és a tanulási forrás az a bemeneti adat, amellyel visszatérünk, a csoportosított adat. A megerősítő algoritmusok a környezetből származó haszon és a környezeti állapot alapján adaptálódnak. A tanulási forrás természetesen a környező környezet és a haszon.

**Tanulási módok**

A gépi tanulás módjai két nagy csoportba válaszhatjuk szét [9]

* Kötegelt vagy offline tanulási módszerek. Ezek gépi tanulási módszerek, amelyeket végpontok közötti gépi tanulási rendszerekben alkalmaznak. Itt a modellt az összes rendelkezésre álló képzési adat felhasználásával képezik egyszerre.
* Az online képzési módszerek a kötegelt tanuláshoz képest más módon működnek. A képzés adatokat általában több növekményes kötegben teszik az algoritmusba. Ezeket a tételeket mini kötegeknek (batch) nevezzük. Dióhéjban az a különbség a kötegelt tanulás és az online tanulás között, hogy miként tápláljuk a képzési adatait.

## A gépi tanulás folyamata

Az előző bekezdésekben megértettük, hogy mi a gépi tanulás és annak különböző alkalmazásai. Majd részleteztem a különböző algoritmusok, amelyeket a különböző üzleti problémákra alkalmaznak. Itt az ideje, hogy egy kicsit elmélyüljünk és megértsük, hogyan valósítjuk meg valójában a gépi tanulást. A következő részben bemutatásra kerül a gépi tanulás folyamata.

**Gépi tanulás menete** [10]

A folyamat négy nagyobb szakaszra lehet osztani:

* ***Probléma meghatározás***: A folyamat egy probléma meghatározásával kezdődik. Itt azonosítjuk és elemezzük üzleti problémánkat.
* ***Adatszerzés***: Ezután folytatjuk az adatszerzés szakaszát, és előkészítjük az adatokat. Ezt követően el kell végeznünk néhány adatszegregációt.
* ***Modell tanítás***: Miután meg vannak a megfelelően előkészített adatok, neki láthatunk a modellünk betanítására. A tanítás befejezése után kiértékeljük a modellt, hogy mennyire felel meg az adott probléma megoldására.
* ***Modell telepítés***: Az utolsó fázis a modell telepítés. Ekkor kerül használatba a betanított modell. Ide tartozik a még a modell karbantartása is.



**Probléma meghatározása**

A probléma-meghatározási szakasz az első szakasz a gépi tanulási folyamatban. A gépi tanulás problémájának első és legfontosabb lépése az üzleti kérdés egyértelmű meghatározása, amelyre választ szeretnénk adni. Megpróbáljuk előre jelezni a következő hónap eladásait a készletelőzmények alapján? Vagy talán a demográfiai információk alapján kategorizálni tervezzük ügyfélkörünket.

Az adattudomány vagy a gépi tanulás öt féle típusú kérdésre tudd választ adni.

* Választási kérdések: Ez „A” vagy „B”? Ez az ügyfél vásárol vagy sem?
* Eldöntendő kérdések: Ez jó? Ez rossz? Egészséges? Van-e rendelleneség?
* Mennyiségre vonatkozó kérdések: Hány darab terméket fogok eladni a következő negyedévben?
* Csoportosítással kapcsolatos kérdések: Hogyan van ez megszervezve? Melyek a különböző vásárlói kategóriák?
* Előrejelzésre releváns kérdések: Mit tegyek ezután? Mi a következő lépés?

A probléma megfogalmazása egy kérdéssel kezdődik. A kérdésre való választ, pedig a gépi tanulás során fogjuk megkapni. Előfordul, hogy a folyamatosan fejlődő világban ugyan arra a kérdésre később más válasz társul. Esetemben több kérdés forma is megmutatkozik. „X” vagy „Y” tananyagot/forrás válaszam a tanulásra? Napi hány óra tanulást töltsek gyakorlattal vagy elmélettel? Milyen típusú tanuló vagyok? Az idő függvényébe ezek különböző válaszokat produkálhatnak.

**Adatszerzés** [9]

A második szakasz az adatforrás kapcsolatos lépésekkel foglalkozik. A gépi tanulási algoritmus olyan adatokon tevékenykedik, amelyeket megtanul és felhasznál a problémák megoldására. Ezért ezeket az adatokat egy gépi tanulási folyamatba kell táplálnunk. Tehát először be kell gyűjtenünk a probléma megoldásához szükséges adatokat.

Jellemzően a szervezeti tudás vagy a történeti adatok több különböző rendszerben vannak szétszórva. Például egy HR-rendszer, amely beszélhet REST-ről, egy 30 éves pénzügyi rendszer, amely csak a fájlintegrációt érti, egy CRM-rendszer, amely SOAP-t protokollon keresztül kommunikál. Tegyük fel egy konfigurációkezelő rendszer, amely az adatbázist közvetlenül SQL nézeteken keresztül teszi elérhetővé vagy akár érzékelők küldik az adataikat a saját eltérő formátumaiban.

Mindezen adattípusokat, amelyekről beszéltünk, strukturált adatoknak nevezünk, ami azt jelenti, hogy meghatározott elrendezésűek, általában táblázatosak. A gépi tanulási folyamat nem strukturális adatokon is működik, például orvosi feljegyzéseken, e-maileken, képeken és videókon.

Miután begyűjtöttük az adatokat a következő lépés, hogy előkészítésük azokat a tanulási algoritmus bemenetére. Az adatok előkészítése az egyik legfontosabb rész az adatszerzés folyamatába. Az előkészítés azért is nélkülözhetetlen, mivel az adatok eltérő forrásokból származnak. Különböző műveleteket hajtunk végre ebben a lépésben: [11]

* a hiányzó adatok azonosításával és kezelésével
* az adatokban lévő felesleges attribútumok eldobásával
* a kiemelkedő adatok azonosításával
* az adatok transzformációjával

Ennek a lépésnek a fő célja, hogy az adatokat a gépi tanulási algoritmusokhoz illesszük. A gépi tanulás algoritmusnak konkrét elvárásaik vannak a beléjük adott adatformátumok tekintetében. Például egyes gépi tanulási algoritmusok normálisan elosztott adatokat igényelnek. Meglepő módon ezt a lépést tartják a legidőigényesebb lépésnek egy gépi tanulási folyamatban. Néhány adatkutató azt állítja, hogy idejének 60-80% -át tölti az adatbányászattal és adatelőkészítéssel.

Amíg nem megfelelőek az adatok, addig nem érdemes elkezdeni a modell tanítását. Hiszen a nem egyértelmű vagy használhatatlan adatokkal való tanítás, értelmetlen és haszontalan modellt eredményezz. Informatikában gyakran használt kifejezés erre, hogy „Szemét be szemét ki”.

Előkészítést követő feladat az adatok szétválasztása. Az adatszegregáció abból áll, hogy az adatokat két halmazra osztjuk. Képzési készlet, amelyet a gépi tanulási algoritmus megtanulására és megtanítására használnak. A gépi tanulási algoritmus kiképzése az algoritmus bizonyos belső paramétereinek beállításának folyamata. Tesztkészlet, amelyet a gépi tanulási algoritmus teljes teljesítményének és általánosításának értékelésére használnak. A teszt adatokat elrejtjük a gépi tanulási algoritmus elől, hogy azokat később ki tudjuk értékelni. Ahhoz, hogy megtudjuk milyen jól teljesít a modell, olyan adatokra van szükség, amelyekkel még nem találkozott. Kétségtelen, hogy az algoritmus jobban fog teljesíteni a képzési adatokon mint a tesztadatokon. Voltaképp memorizálta a megoldást, nem pedig „megértette” a megoldáshoz vezető utat. A modell értékelése alapvető irányelv.

**Modell tanítás** [12]

Miután meg vannak az adatok következhet a modellképzése. A gépi tanulás folyamatának lényege a modell tanítás. A gépi tanulási algoritmusok itt illeszkednek a képzési adatainkhoz és előreláthatóan a megadott paraméterek alapján módosítják belső paramétereit. A gépi tanulási algoritmusok általában a polcon található receptek, amelyeket a megfelelő üzleti problémákra, például osztályozásra, lineáris regresszióra és klaszterezésre vonatkozó problémákra alkalmaz. Ez a lépés magában foglalhatja egynél több gépi tanulási algoritmus kipróbálását, mivel nem tudjuk előre, hogy melyik algoritmus illik valójában a mögöttes adat jellemzőkre.

A tanítás után történik a modell kiértékelése. Miután betanítottuk a gépi tanulási modellünket, itt az ideje, hogy teszteljük és értékeljük a teljesítményünket. Elég gyakori, hogy több gépi tanulási modellünk van, amelyek közül szeretnénk választani, és összehasonlítjuk teljesítményüket. Ezért hasznos, ha az adatot szétválasztjuk több validációs adatkészletre. Vannak bizonyos teljesítményértékelési mutatók, amelyek objektíven felhasználhatók a gépi tanulási algoritmusok teljesítményének összehasonlítására. Ilyen például az osztályozási táblázat és osztályozási algoritmusok, valamint az RSK (Robinson–Schensted–Knuth) és a lineáris regressziós algoritmusok.

**Modell telepítés** [10]

Az utolsó szakasz a modell telepítésé. Miután kiválasztottuk a gépi tanulási modellünket és elégedettek vagyunk a teljesítményével, eljött az ideje, hogy ezt a felhasználók és az üzleti élet számára is felhasználjuk. A gépi tanulási modell általában elfogadja az független rendszerek bemenetét (upstream), alkalmazza a gépi tanulási technikákat. A kimenetet és az eredményeket átadja a komplex rendszereknek (downstream).



Lehetőség van kiterjeszteni a gépi tanulási modelljeinket API-k, háttérfeldolgozási feladatok vagy bármilyen más megfelelő mechanizmus formájában.

Végül a modell figyelése. A modell üzembe helyezése és működése után a történet még nem fejeződött be. A gépi tanulási modellek nem jelentenek kivételt azoktól az általános szoftveres megoldásoktól, amelyek karbantartást és üzemeltetési gondozást igényelnek. Ebben az utolsó lépésben a modell teljesítményünket a teljesítmény helyessége alapján mérjük. Például mennyire pontosak az előrejelzései, a válaszidő, a használt funkciók eloszlása ​​stb. Ez az értékes információ visszajuttatható gépi tanulási folyamatunkba annak jövőbeni teljesítményének javítása érdekében. A modellek kiépítését és felügyeletét együttesen gépi tanulás üzemeltetésnek (MLOps) nevezik. Célja, hogy növelje az automatizálást és javítsa a gyártás minőségét.

# Probléma meghatározása

És itt az ideje bemutatni azt az üzleti problémát, amelyet a gépi tanulási megoldásunk megvalósításához fogunk használni. Diákokkal kapcsolatos adatkészletet fogunk használni.

Az adatkészlet az Kaggle adattárból származik, amely a gépi tanulási közösség által használt empirikus elemzéshez használt adatbázisokat és vetélkedési lehetőségeket tartalmaz. Az adatállomány **<<ide\_a\_rekord\_szám>>** rekordot ír le az előrejelzésekről a diák teljesítményére mutatnak. **<<Mutatók bemutatása>>**.

A cél az, hogy az adatokból kinyerjük, hogy milyen kontextusba tanulnak a legmegfelelőben a diákok. Ha felidézi a korábban bevezetett gépi tanulási folyamatot, akkor ez a múltbéli adatokból való tanulás.

Különösen milyen típusú kérdésre próbálunk választ adni?

Milyen üzleti probléma típus? (regresszió, klasszifikáció)

Miután meghatároztuk problémánkat, az diákok adatkészlete képezi majd az alapot, amelyre a gépi tanulási folyamatot alkalmazzuk. A gépi tanulás folyamatát lépésről lépésre fogjuk használni az adatkészletünkön egy gépi tanulási megoldás megvalósításához.

Milyen tanulási algoritmus/algoritmusokkal?

## Az adatforrás a kezdetek kezdete

**Mik lehetnek az adatok?** [13]

Lehet bármilyen feldolgozatlan tény, érték, szöveg, hang vagy kép, amelyet nem értelmeznek és elemeznek. Az **adat**ok az összes adatelemzés, gépi tanulás, mesterséges intelligencia legfontosabb része. Adatok nélkül nem képezhetünk egyetlen modellt sem, és minden korszerű kutatás és automatizálás hiábavaló lesz. A nagyvállalkozások rengeteg pénzt költenek csak azért, hogy minél több biztos adatot gyűjtsenek.

Meg kell különböztetni az adat és információ közti különbséget. Az **információ** olyan adatok, amelyeket értelmeztek és manipuláltak, és amelyek most a felhasználók számára érdemi következtetéseket vonnak le. Tehát az adatunk valamilyen transzformáción és adattisztítási lépéseken esett át.

A következő lépcsőfok a **tudás**, ami az információ után következik. A megszerzett tudás következtetett információk, tapasztalatok, tanulás és meglátások kombinációja. Eredménye egy személy vagy szervezet koncepciójának kiépítése.



Vegyünk egy példát! Egy vásárlói központ tulajdonosa felmérést készített. Hosszú listát állított fel az ügyfelektől feltett kérdések és válaszok alapján. A kérdések és válaszok listája maga az adat. Minden alkalommal, amikor bármire következtetni akar, akkor az ügyfelek miden egyes kérdéseit és válaszait át kell nézni. Az összes adat végig nézése nem hasznos és nagyon időigényes folyamat. Az időveszteség csökkentése és a munka megkönnyítése érdekében nyújtanak segítséget a szoftverek, matematikai számítások, grafikonok. Ezek a támogató eszközök az adatokat manipulálják, megtisztítják, transzformálják. A manipulált adatokból levont következtetés az információ. Tehát az adatokból nyerjük az információkat. A tudásnak szerepe van abban, hogy különbséget tegyünk két egyforma információval rendelkező egyén között. A tudás valójában nem technikai tartalom, hanem az emberi gondolkodási folyamathoz kapcsolódik.

## Adatok strukturáltságának változatai

Az adatokat strukturáltság szempontjából három nagy kategóriába sorolhatjuk: a strukturált, a félig strukturált és a strukturálatlan adatok.

**Strukturált adatok** [2]

A strukturált adatok olyan információk, amelyeket formáztak és jól definiált adatmodellekké alakítottak. A nyers adatok előre megtervezett mezőkbe vannak leképezve, amelyeket később kibonthatnak és könnyedén beolvashatnak. Az SQL relációs adatbázisok, amelyek sorokból és oszlopokból álló táblákból állnak, tökéletes példa a strukturált adatokra.

A strukturált adatok relációs modellje kihasználja a memóriát, mivel ez minimalizálja az adatok redundanciáját. Ez azonban azt is jelenti, hogy a strukturált adatok jobban függenek egymástól és kevésbé rugalmasak.

A strukturált adatokat emberek és gépek egyaránt generálják. Számos példa található a gépek által létrehozott strukturált adatokra. Például egy bankszámlakivonat, amely tartalmazza a dátumot, az időt, az összeget stb. Hasonlóképpen, bárki, aki az adatokon dolgozik, életében egyszer használta a táblázatokat, ami az emberek által generált strukturált adatok klasszikus esete. A strukturált adatok szervezése miatt könnyebb elemezni, mint a félig strukturált és a strukturálatlan adatokat.

**Félig strukturált adatok** [2]

Adatai nem mindig lehetnek strukturáltak vagy strukturálatlanok - a félig strukturált adatok a kettő között egy másik, részben strukturált kategóriába tartoznak. Az ilyen adatok félig strukturáltak. A félig strukturált adatok olyan adattípusok, amelyek bizonyos következetes és határozott jellemzőkkel bírnak, nem korlátozódnak olyan merev struktúrába, mint amilyen a relációs adatbázisokhoz szükséges. Az olyan szervezeti tulajdonságokat, mint a metaadatok vagy a szemantikai címkék, félig strukturált adatokkal együtt kezelhetőbbé teszik, azonban továbbra is tartalmaz némi változékonyságot és következetlenséget.

A félig strukturált adatokra példa a tagolt fájlok. Olyan elemeket tartalmaz, amelyek külön hierarchiákra bonthatják az adatokat. Hasonlóképpen, a digitális fényképeken a képnek sincs előre meghatározott struktúrája. Ennek ellenére, ha okostelefonról veszik, akkor strukturált attribútumai vannak, például helyazonosító, eszközazonosító és dátum. Tárolás után a képekhez hozzárendelhetők olyan címkék is, mint a „háziállat” vagy a „kutya”, hogy struktúrát adjanak.

Bizonyos esetekben a strukturálatlan adatokat félig strukturáltnak minősítik, mert egy vagy több osztályozó attribútummal rendelkeznek.

Ilyenek lehetnek például a naplófájlok, JSON fájlok, érzékelő adatok, csv fájlok stb.

**Strukturálatlan adatok** [2]

Az abszolút nyers formában jelen lévő adatokat strukturálatlannak nevezzük. Ezeket az adatokat összetett elrendezése és formázása miatt nehéz feldolgozni. A strukturálatlan adatok sokféle formát ölthetnek, beleértve a közösségi média bejegyzéseket, csevegéseket, műholdas képeket, Internet of Things (IOT) érzékelő adatokat, e-maileket és prezentációkat.

Nem rendszerezett adat (strukturálatlan) nem illeszkedik a relációs adatbázis sorokba és oszlopokba. Például ilyenek lehetnek a szöveges fájlok, e-mailek, képek, videók, hangposták, hang fájlok stb.

**Különbségek a strukturált, félig strukturált és strukturálatlan adatok között** [14]

Az interjúk analógiája alapján különböztessük meg ezt a három típusú adatstruktúrát. Tegyük fel, hogy háromféle állásinterjú létezik: strukturálatlan, félig strukturált és strukturált interjúk.

Egy strukturálatlan formátumú interjúban a feltett kérdések teljesen az interjúztató választása. Ő eldöntheti azokat a kérdéseket, amelyeket fel akar tenni, és azok sorrendjét. A strukturálatlan kérdések népszerű példái: „mesélj magadról” és „írd le ideális szerepedet”.

Egy másik típus a strukturált interjú. Ebben az esetben a kérdező szigorúan követni fogja a HR osztály által létrehozott forgatókönyvet, és ugyanazt a forgatókönyvet fogja használni minden pályázó számára.

A harmadik típus félig strukturált. Egy félig strukturált interjúban az interjúkészítő egyesíti a strukturálatlan és a strukturált interjú elemeit. Ez magában foglalná a strukturált interjúhoz hasonlóan a mennyiségi és a következetességi elemeket. Ugyanakkor rugalmasan alkalmazhatja a kérdéseket a helyzetnek megfelelően, ami a strukturálatlan interjú egyik aspektusa. Megismételve azonban, a strukturálatlan és félig strukturált adatok közötti fő különbség az, hogy a strukturálatlan adatok nem követnek előre meghatározott formátumot, míg a félig strukturált adatok csak részben strukturálatlanok.

A következő pontok rámutatnak a strukturált, félig strukturált és strukturálatlan adatok közötti különbségekre:

* ***Szervezet***: A strukturált adatok jól szervezettek, ezért rendelkeznek a legmagasabb szintű szervezettséggel. Míg a félig strukturált adatok részben szervezettek, ezért a szervezés szintje alacsonyabb, mint a strukturált adatoké, de magasabb, mint a strukturálatlan adatoké. Végül a strukturálatlan adatokat egyáltalán nem rendezik.
* ***Rugalmasság és méretezhetőség***: A strukturált adatok relációs adatbázisoktól vagy sémáktól függenek, ezért kevésbé rugalmasak és nehezen méretezhetők. Míg a félig strukturált adatok rugalmasabbak és egyszerűbbek a méretezéshez, mint a strukturált adatok. A strukturálatlan adatoknak azonban nincs olyan sémájuk, amely a legrugalmasabbá és méretezhetőbbé tenné a másik kettő közül.
* ***Változatkészítés***: Mivel a strukturált adatok relációs adatbázison alapulnak, a verziószámcsomagok, sorok és táblák felett történik. Másrészről félig strukturált adatokban sorok vagy grafikonok lehetségesek, mivel csak egy részleges adatbázis támogatott. Végül a strukturálatlan adatokban a verziók valószínűleg teljes adatok, mivel az adatbázis nem támogatott.
* ***Tranzakciókezelés***: A strukturált adatokban az adatok egyidejűsége érhető el, ezért általában előnyben részesítik a többfeladatos folyamat során. Míg a félig strukturált adattranzakciók a DBMS-től adaptálódnak, de az adatok egyidejűsége nem áll rendelkezésre. Végül a strukturált adatokban sem a tranzakciókezelés, sem az adatok egyidejűsége nincs jelen.

## Hogy szerezünk adatokat?

**Az adatkészletek típusa** [15]

Az adattudományi projekt kialakítása során többen azt gondoljuk vagy feltételezük, hogy létre kell hozni az algoritmusok változatait. Majd megbecsülni a modell teljesítményét a képzési adatokon. Végül értelmezni az előrejelzett eredményeket. Mielőtt azonban ezeket a lépéseket elkezdnénk, nemcsak a megfelelő adatokat kell megszereznie, hanem azt is ellenőriznie kell, hogy az megfelelő címkézéssel van ellátva. Még akkor is, ha nincs szükség konkrét adatok gyűjtésére, sok időbe telhet egy olyan adatforrás megtalálása, amely a legjobban megfelel a projektnek.

Több ezer nyilvános adatkészlet különböző témákban elérhető online.

* ***Adatkészletek az általános gépi tanuláshoz***: Ebben az összefüggésben az „általános” gépi tanulás alatt a regressziót, az osztályozást és a csoportosítást értjük. Ezek a leggyakoribb gépi tanulási feladatok.
* ***Adatkészletek a mély tanuláshoz (deep learning)***: Az általános gépi tanulástól némileg különböző a mély tanulás. Alapvetően a mély tanulás olyan, mint a neurális hálózat. A különbség, hogy a neuronok itt számos rétegekbe szerveződnek. Leginkább a kép, szöveg vagy hangadatok feldolgozására használják a mély tanulást.
* ***Adatkészletek a természetes nyelv feldolgozásához***: A természetes nyelv feldolgozása (N.L.P.) a szöveges adatokról szól. Az olyan rendezetlen adatokhoz, mint a szöveg, különösen fontos, hogy az adatkészletek valós alkalmazásokból származzanak. Ennek érdekében könnyedén tudunk elvégezni az adat realitásának és életszerűségének az ellenőrzését (sanity check).
* ***Adatkészletek az idősor-elemzéshez***: Az idősor elemzéséhez időbélyeggel ellátott megfigyelésekre van szükség. Más szavakkal, minden adatrekordot az idő folyamán követnek nyomon és rögzítenek.
* ***Adatkészletek az ajánló rendszerek számára***: Az ajánló rendszerek átvették a szórakoztató és az e-kereskedelmi iparágakat. Remek példa az ajánló rendszerekre a Facebook, az Amazon, a Netflix és a Youtube.

Ahhoz, hogy minél kevesebb időt fordítsunk a megfelelő adatkészlet keresésére, tisztában kell lenni, hogy hol keressük meg. A továbbiakban részletezem, hogy hol tudunk a gépi tanulási projektekhez nyilvánosan elérhető adatforrást találni.

**Adatforrás gyűjtemények helye** [16]

1. A Google adatkészlet-keresője

A Google a keresőmotor óriás volt, és az összes gépi tanulással foglalkozó szakembert segítették azzal, hogy segítenek nekünk megtalálni az adathalmazokat. A keresőmotor nagyszerű munkát végez a kulcsszavakhoz kapcsolódó adatkészletek megszerzésében különböző forrásokból, ideértve a kormányzati weboldalakat, a Kaggle-t és más nyílt forráskódú adattárakat.

1. .gov adatkészletek

Mivel az Egyesült Államok, Kína és még sok más ország Mesterséges intelligencia nagyhatalommá válik, az adatok demokratizálódnak. Az ezen adatkészletekre vonatkozó szabályok és előírások általában szigorúak, mivel a nemzet különböző szektoraiból gyűjtött tényleges adatok. Ezért óvatos használat ajánlott. Van néhány olyan ország, amelyek nyíltan megosztják adataikat.

1. Kaggle adatkészletek

A Kaggle a gépi tanulás és a mély tanulási kihívások tárhelyéről ismert. A Kaggle relevanciája ebben az összefüggésben az, hogy adatkészleteket nyújt, és egyúttal a tanulók és az gépi tanulási szakemberek közösségét is biztosítja. Nagyban elősegíti a fejlődésüket a jövendő adattudósoknak. Minden kihívásnak külön adatkészlete van és általában megtisztítják az adatokat, hogy ne kelljen szükségszerűen a takarítás munkáját elvégeznünk. Helyett az algoritmus finomítására összpontosíthatunk. Az adatkészletek könnyen letölthetők. Az erőforrások részben előfeltételek és linkek találhatók a tananyaghoz, amely segít nekünk, amikor elakadunk akár az algoritmusnál, akár a megvalósításnál. A Kaggle egy fantasztikus webhely a kezdők számára, akik belevághatnak a gépi tanulás és a mélytanulás alkalmazásaiba. Továbbá részletes erőforráskészletet nyújt a gépi tanulás haladó gyakorlói számára is.

1. Amazon adatkészletek

Az Amazon nyilvántartásba vette a szervereiken elérhető néhány adatkészletet. Egyik nagy előnye, hogy amikor Amazon Web Services (AWS) erőforrásokat használunk a modellek kalibrálásához és módosításához, akkor lokális adatokat használunk. A helyileg elérhető adatkészleteknek a használata több tízszer gyorsítja az adatbetöltési folyamatot. A nyilvántartás számos adatkészletet tartalmaz az alkalmazások területe szerint, például műholdas képek, ökológiai erőforrások stb.

1. UCI Machine Learning Repository

Az UCI Machine Learning Repository könnyen kezelhető és tisztított adatkészleteket kínál. Ezek már régóta az egyik leggyakrabban használt adathalmazok a tudományos területeken.

**Hogyan keressünk könnyeden adatkészletet?** [17]

A mai világban sok tudományterület tudósai és egyre több elemző, újságíró gyűjt és állít elő adatokat. Az interneten sok ezer adattár található, amelyek milliónyi adatkészlethez nyújtanak hozzáférést. A helyi és nemzeti kormányok világszerte közzéteszik adataikat is. Az adatokhoz való könnyű hozzáférés érdekében elindították az adatkészlet keresést. Elősegítve, hogy a tudósok, az újságírók, az adatkezelők vagy bárki más megtalálhassa a munkájához és történeteihez szükséges adatokat, vagy csak egyszerűen kielégítse intellektuális kíváncsiságukat.

A Google Tudós működéséhez hasonlóan az adatkészlet keresés lehetővé teszi az adatkészletek megtalálását, bárhol is tárolják őket. Legyen szó kiadói webhelyről, digitális könyvtárról vagy a szerző személyes weboldaláról. Az adatkészlet keresés létrehozásához irányelveket dolgoztak ki az adatforrás szolgáltatók számára. Olyan értelemben írják le az adataikat, hogy a Google (és más keresőmotorok) jobban megértsék oldalaik tartalmát. Ezek az irányelvek kiemelkedő információkat tartalmaznak az adatkészletekről:

* Ki hozta létre az adatkészletet?
* Mikor jelent meg?
* Hogyan gyűjtötték az adatokat?
* Milyen feltételek vonatkoznak az adatok felhasználására?

Ezután összegyűjtött és összekapcsolt információkat elemzik, hogy ugyanazon verziók hol változtak. Találhatunk olyan publikációkat, amelyek leírják vagy megvitatják az adatkészletet. A megközelítés az információk leírásának nyílt szabvány alapú. Tehát bárki, aki adatokat publikál, leírhatja az adatkészletét így. Arra ösztönözve az adathalmaz-szolgáltatókat, hogy fogadják el ezt a közös szabványt, hogy minden adatkészlet része legyen ennek a robusztus ökoszisztémának.

Ebben az új kiadásban megtalálhatók a legtöbb környezeti és társadalomtudományi adatkészlet, valamint más tudományágak adatai. Ideértve a kormányzati és a hírszervezetek által szolgáltatott adatokat. Mivel egyre több adattár használja a közös szabványt az adatkészletek leírására. A felhasználók által az adatkészlet keresésben megtalált adathalmazok változatossága és lefedettsége a mai napig is növekszik.

Az adatkészlet keresés több nyelven működik, hamarosan további nyelvek támogatásával. Egyszerűen be kell írni, amit keresünk. Majd a keresőmotor segít a letárolt adatforrás megtalálásához.

Például, ha elemezni szeretnénk a diákok teljesítményére vonatkozó rekordokat, kipróbálhatjuk ezt a lekérdezést az adatkészlet keresésben:



Látni fogjuk a Kaggle által szolgáltatott adatait, valamint olyan tudományos adattárakat, mint például a Harvard Dataverse és a Statista. Számos adattárak célja, hogy támogassa a projektet és segítse, hogy számos adatkészletét kereshetővé tegyen ebben az eszközben.

Ez az adatkészlet keresés bevezetés azon kezdeményezések egyike, amelyek célja az adatforrásokat könnyebben beépítsük a projektjeinkbe és termékeinkbe. Míg ez a kezdeményezés inkább a hírszervezetekre és az adatújságírókra összpontosított, az adatkészlet keresés sokkal szélesebb közönség számára lehet hasznos, függetlenül attól, hogy tudományos adatokat, kormányzati adatokat vagy hírszervezetek által szolgáltatott adatokat keres.

Az ehhez hasonló keresőeszköz csak olyan jó, mint azok a metaadatok, amelyeket az adatkiadók hajlandók megadni. Remélhetőleg sokan a nyílt szabványokat használják az adatok leírására. Lehetővé téve a felhasználók ​​számára, hogy megtalálják azokat az adatokat, amelyeket keresnek.

## Saját adatforrásom ismertetése

Adat alapvetően: Ált leírás: null érték, zajok

Adathalmaz ismertetése

Átvezetés:

*Többet mond egy kép*

*mint száz bekezdés*

# **Reprezentációs módszerek**

## Miért hasznos az adatok megjelenítése?

Az adattudományban az egyik legfontosabb készség az adatok eloszlásának és összetettségének vizualizálása és megértése. Manapság azonban az online tanfolyamok nagy része elsősorban a gépi tanulásra és az algoritmus működésére összpontosít.

Ez nem azt jelenti, hogy a gépi tanulás ismerete elhanyagolandó. Nyilvánvalóan a legjobb adattudós könnyedén eligazodik a gépi tanulási algoritmusok nagy részével. Mindenesetre az adattudomány nem korlátozódik le a gépi tanulásra, inkább az egy képesség. A készség alatt azt kell érteni, hogy mennyire érted a probléma területet (domain), mennyire tudsz elmélyülni az adatokban. Elkapni a rejtett összefüggéseket és megtalálni az adatokban rejlő üzenetet. Az adat beszél magáról. A legjobb mód arra, hogy szóra bírjuk az ***adatmegjelenítés***. [18]

**Mi az adatmegjelenítés?**

Az adatmegjelenítés az információ (adatok) felvételének és vizuális kontextusba, például térképbe vagy grafikonba helyezésének módszere. A fő cél a nagy adatkészletek vizuális grafikába történő szűrése, hogy lehetővé tegye az adatokon belüli összetett kapcsolatok könnyű megértését. [11]

Tehát a vizualizáció választ ad azokra a kérdésekre, amelyeket nem tudunk.

**Miért fontos az adatmegjelenítés?**

Az adatmegjelenítést egyre inkább minden sikeres adatközpontú elemzési stratégia alapvető utolsó lépésének tekintik. Amikor az adattudósok egy összetett projekt közepén vannak, szükségük van egy módra az összegyűjtött adatok megértésére. Figyelemmel kell kísérni és módosítani folyamatot annak biztosítása érdekében, hogy megfelelően funkcionáljon. Az adatmegjelenítés megkönnyíti az adatcsoportok mintáinak, előítéleteinek és kiugró értékeinek észlelését is.

A célközönség vezetése arra, hogy az üzleti felismerésekre összpontosítson, hogy felfedezzék a figyelmet igénylő területeket. Korábban észrevétlen kulcsfontosságú tények feltárása az adatforrásokkal kapcsolatban, hogy a döntéshozók segítsenek adatelemzési jelentések elkészítésében. Segít az érdekelt feleknek és a csapat többi tagjának minőségi információkkal szolgálni azáltal, hogy hatalmas mennyiségű adatot könnyen érthető képpé és grafikává alakít.

**Az adatmegjelenítés előnyei** [18]

Figyelembe véve, hogy az adatok milyen hatást gyakorolnak a vállalkozás növekedésére, íme néhány előny

* Segít felismerni a legújabb fejlesztéseket a termék fejlesztése és az üzleti nyereség növelése érdekében.
* Az adatmegjelenítések megkönnyítik a kis és nagy adatok megértését az emberi agy számára, ami jobb elemzéshez vezet.
* Segít megérteni a történetet - Az emberi agy nem képes egyszerre nagy mennyiségű számot vagy szöveget megérteni, sőt csak elképzelni. Szüksége van egy vizuális ábrázolásra, hogy értelmezze őket, és ennek következtében a nyers adatokat kézzelfogható fogalommá alakítsa.

## Ábrázolási formák

**<<Ide\_mehet a Data sience from Stratch-ből (Visualizing Data>>**

**bevezetés**: történet, gépek előtt, hogy ábrázoltunk, mit ábrázoltunk (barlangokra strigulát stb.). Az ember maximum három dimenziót képes értelmezni. Az adatelemzésnél a dimenziót úgy kell felfogni, mint a tér leírására használtat. Szélesség, hosszúság és magasság helyett adatok tulajdonsága mentén ábrázolunk. Egy csillag esetében például: hőmérséklet, tömeg és fényesség. Az adatok természetéből kifolyólag számtalan dimenzió megjelenítés lehetséges.

**különböző tulajdonságok**at ugyanazon a koordinátán való ábrázolása

**2d**

**3d**

**Hogyan tudunk háromnál több dimenziót ábrázolni?**

...

## Saját adathalmazon bemutatása

**Saját adathalmazon ábrázolása**: Milyen megjelenítést használtam, több adathalmaz bemutatása, észrevételek, következtetések stb.

**Megjelenítési forma bemutatása**: Miért ezt választottam. Mi mit jelent rajta. Hány db adat. Színek, csoportok leírása stb.

**Mit mond az ábra?** Ábráról leszűrt következtetések. Célszerű stratégiák megfogalmazása: outlier eltávolítása, dimenzió csökkentés/növelés, adat csökkentés/növelés stb.

**Az ábrázolás az érme egyik oldala**: ábrázolás + statisztika. Az ábrázolás lehet megtévesztő. Mikor lehet megtévesztő és miért? Átvezetés: Nem szabad hinni mindig a szemünkben. Az ember képes beleesni az optikai illúziók csapdáiban. Ennek érdekében nyújthat segítséget az adatok statisztikai elemzése.

# Adatelemzés lépései és adathalmazok ismertetése

## Adat a gépi tanulásban

**Az adat** [19]

A gépi tanulási módszerek példákból tanulnak. Fontos, hogy jól megértsük a bemeneti adatokat és az adatok leírásakor használt különféle terminológiát. Ebben a szakaszban megismerhetjük a gépi tanulásban használt terminológiát, amikor az adatokra hivatkozik.

Ha az adatokra gondolok, akkor sorokra és oszlopokra gondolok, például egy adatbázis táblára vagy egy Excel táblázatra. Ez egy hagyományos adatstruktúra, és ez általános a gépi tanulás területén. Egyéb adatok, például képek, videók és szöveg, az úgynevezett strukturálatlan adatok egyelőre nem vesszük figyelembe.



***Példány***: Egyetlen adatsort nevezünk példánynak. Ez egy megfigyelés az adatkészletből.

***Jellemző***: Egyetlen adatoszlopot nevezünk jellemzőnek. Ez egy megfigyelés összetevője, és adatpéldány attribútumának is nevezik. Egyes jellemzők lehetnek bemenetek egy modellhez, mások pedig kimenetek vagy a megjósolható jellemzők.

***Adattípus***: A jellemzők adattípussal rendelkeznek. Lehetnek valós vagy egész értékűek, vagy lehetnek kategorikus vagy sorszámértékek. Lehetnek karakterláncai, dátumai, időpontjai és összetettebb típusai is. Ezek tipikusan valós vagy kategorikus értékekre redukálódnak a hagyományos gépi tanulási módszerek használatánál.

**Címke**: A célváltozó vagy a címke az az érték, amelyet a modellünk megjósol. Például a mi esetünkben a matek dolgozat eredménye, az olvasási készség pont, vizsgán elért eredmény stb.

***Adatkészletek***: A példányok gyűjteménye egy adatkészlet, és amikor gépi tanulási módszerekkel dolgozunk, általában néhány adatkészletre van szükségünk különböző célokra.

***Képzési adatkészlet***: Adatkészlet, amelyet a gépi tanulási algoritmusunkba töltünk be modellünk betanításához.

***Tesztelési adatkészlet***: Olyan adatkészlet, amelyet a modellünk pontosságának ellenőrzésére használunk, de nem használjuk a modell képzésére. Nevezhetjük értékelési adatkészletnek.

***Validációs adatkészlet***: Az adatnak az a része, amelyet a modell gyakori kiértékeléséhez használnak. Illeszkedik a képzési adatkészletbe, valamint javítja az érintett hiper-paramétereket (kezdetben beállított paraméterek a modell tanulás előtt). Ezek az adatok akkor játszanak szerepet, amikor a modell ténylegesen a tanulás fázisába van.



Előfordulhat, hogy példányokat kell gyűjtenünk az adatkészletek létrehozásához, vagy kapunk egy véges adatkészletet, amelyet fel kell osztanunk részadatokra.

**Az adatok tulajdonságai** [12]

***Méret***: Adatok skálája. A világ népességének és technológiájának a növekedése óriási adatok produkálnak minden másodpercben. A felhőalapú számítástechnikai, az Internet of Things (IoT) szolgáltatások, a mobilforgalomnak tovább növelik az adatok mennyiségét. Mostanra már az adat méreteinek redukálására kell helyezni a hangsúlyt.

***Változatosság***: Különböző adatformák - egészségügy, képek, videók, hangfelvételek. Strukturált, félig strukturált és strukturálatlan adatokra utal, különféle adatforrások következtében, akár emberek, akár gépek generálják.

***Sebesség***: Az adatok halmozódásának sebességére vonatkozik. Ennek oka elsősorban az IoT, a mobil adat, a közösségi média stb.

***Érték***: Az adatok értelmessége olyan információk szempontjából, amelyekből a kutatók következtetni tudnak. Csak azért, mert rengeteg adatot gyűjtöttünk, nincs értéke, hacsak nem gyűjtünk belőle némi betekintést. Az érték arra utal, hogy az adatok mennyire hasznosak a döntéshozatalban. Megfelelő elemzéssel kell kinyernünk az adat értékét.

***Valódiság***: Az adatok minőségének, integritásának, hitelességének és pontosságának biztosítására utal. Mivel az adatokat több forrásból gyűjtik össze, ellenőriznünk kell az adatok pontosságát, mielőtt üzleti ismeretekre felhasználnánk őket.

## Adatelemzési lépések

**Bevezetés**

**Egyváltozós numerikus elemzés**

**Kétváltozós numerikus elemzés**

**Leíró statisztika**

## Hogyan osszuk szét az adatokat?

**Az adatok szétválasztásának szükségessége** [20]

Az adatelőkészítés után az adatok szétválasztásával kell foglalkozni. Felmerül a kérdés, hogy miért kell szétválasztani az adatokat. Megvan a gépi tanulási algoritmusunk, amelyet egy adott üzleti kérdés megválaszolásához szeretnénk használni. A cél az, hogy sikeresen általánosítson egy nem ismert adatbevitelre. Nevezzük a mintán kívüli adatoknak. Mielőtt azonban felhasználhatnánk az algoritmust vagy a modellt, először be kell tanítanunk az adatkészletünkbe. A tanítás után is szeretnénk kipróbálni, hogy megnézzük annak teljesítményét.

A képzési és tesztelési fázisok elvégzéséhez van egy adatkészlet. Az adatkészletet tekinthetjük már elkészítettnek. Az adatokat szét fogjuk osztani egy olyan képzési halmazra, amely képezni fogja algoritmusunkat. A másik rész pedig mint tesztkészlet lesz felosztva, amelyet az algoritmusunk tesztelésére használunk. A tesztelési eredményekből olyan mutatókat kapunk, amelyek segítenek eldönteni, hogy egy adott modell hogyan teljesítene a mintán kívüli adatokon. Ezen mutatók alapján eldönthetjük, melyik modell lesz alkalmas a választott problémánkra. Az algoritmus jellegétől függően sok mutató van. Azonban az egymillió forintos kérdés a gépi tanulásban az, hogy hogyan fogjuk felosztani az adatkészletünket?

**Előítéletes adat**

Érdekes, hogy ez történelem a gépi tanulásban, és két szempont van az adatok felosztásakor. Először is szeretnénk **randomizálni** az adatkészletünket, hogy a hasonló értékek ne oszlanak fel ugyanazon halmazra. Nem akarjuk megosztani az adatkészletünket úgy, hogy a férfiak vizsgáinak a pontszáma a képzési készletben legyen, míg az nők pontszámai a tesztkészletben. Ellenkező esetben előítéletet (bias) vezethetünk be. [12]

**Rosszul illeszkedő adat** [9]

Másodszor, van egy kompromisszum. Minél nagyobb a képzési készletünk, annál nagyobb a bizalom a modell tanulás iránt. Minél nagyobb a tesztkészletünk, annál nagyobb a bizalom a modelltesztelés iránt. Tehát az, hogy hogyan kell kettéválasztanunk, egy másik probléma. Lehetséges, mégis logikus kérdés, amelyet fel lehet tenni, hogy miért van gondunk az adatkészlet felosztásával? Miért nem csak a gépi tanulási algoritmust képezzük ki az egész adathalmazba, és így garantáljuk az algoritmusunk számára a lehetséges teszt tanulást.

Az ugyanazon az adatkészleten végzett képzés és tesztelés az úgynevezett túlillesztés (overfitting) eredményét fogja eredményezni. Ekkor egy modell inkább a tanító adatait kezdi megjegyezni, mintsem általánosítani. Úgy is értelmezhetjük, mint egy olyan hallgató, aki megjegyzi az oktatóanyagokat, de nem képes általánosítani. Azokat a vizsga kérdéseket, amelyeket még nem látott nem tudja megválaszolni. Egyszerűen bemagolja a tananyagot a vizsgára, ami nem jó. De a gépi tanulási modellt és a hallgatót új, eddig nem látott adatokkal kell tesztelni, hogy biztonságosan megállapíthassuk, hogy megfelelően tanultak-e.

Mivel elkezdtük megvitatni a túlillesztést, be kell vezetnünk az alul illesztést a teljesség érdekében. Az alul illesztést akkor következik be, amikor a gépi tanulási algoritmusok nem tanultak eléggé a bemeneti adatokra, ezért nem tudják megfelelően megbecsülni a jövőbeli bemeneti értékeket. Elképzelhetjük úgy, mint egy hallgató, aki nem tanulmányozta megfelelően a korábbi vizsga kérdéseit.

A megfelelő illesztésre törekszünk. Amikor a gépi tanulási algoritmus éppen annyit tanul, hogy a bemeneti adatokból általánosítson. Mégsem jegyzi meg a modellünket. Ekkor megjegyezhetjük, hogy gépi tanulás szépen követi a bemeneti adatok trendjét anélkül, hogy pontosan át kellene menni az összes adatpéldányon. A hasonlat itt egy olyan hallgató lesz, aki épp annyira tanulmányozza a vizsga kérdéseit, hogy sikeresen megértse a vizsga stílusát, stratégiáját és irányzatát, anélkül, hogy energiáját csak a tiszta memorizálásra fordítaná.



**Képzés és tesztkészletre felosztás**

Két módszer létezik az adathalmazunk elkülönítésére.

* Az első technikát képzés és teszt felosztásnak nevezzük, ahol az adatokat két különböző képzés és tesztkészletre osztjuk.
* Míg a második technikát K-Fold keresztellenőrzésnek nevezik, ahol a hiányokat csoportokra és foltokra osztjuk, és minden csoportot külön tanítunk.

Az első adatszegregációs technika, amelyet megvizsgálunk, a képzés és teszt felosztás. A képzési halmaz, amely egy nagyobb részt vesz az adatkészletünkből. A tesztkészletet, amely kisebb arányt vesz fel az összes adatból. A képzési teszt általában 70-80% -ot vesz igénybe az adatkészletünkből, míg a tesztkészlet általában 20-30% -ot vesz igénybe. [10]

**Diák adatok felosztása**

Vegyünk egy vizuális példát a képzés és teszt felosztás szakaszhoz és nézzük meg annak fő korlátját. Tegyük fel, hogy van a diák által tanulásra töltött ideje és az eredménye. Az általános tendencia az, hogy a tanulásra ráfordított idő növekedésével jobb eredmény érünk el. Kétféle felosztással képeztem ki az adatkészletemet. Más-más tanulási és teszt kombinációval. Nézzük meg, hogyan különbözik az algoritmus teljesítménye! A következő képen látható a tanító adatok eloszlása.



A gépi tanulási algoritmusra egy egyszerű lineáris regressziós algoritmust választottam. A lineáris regressziós vonal megközelítőleg az adatsorunk közepén halad el, és megmondja, hogyan fog kinézni a jövőbeni előrejelzése. Az alábbi tesztelési adatkészlettel teszteljük a betanított gépi tanulási algoritmust.



A hibát a betanított gépi tanulási algoritmusunk által becsült tényleges érték és a tesztelési adatok közötti távolság határozza meg. Számos technika létezik egy betanított algoritmus hibájának kiszámításához. A hiba kiszámításának egyik technikája a hibák négyzetének összege. A hibákat négyzetre osztjuk, így eltávolítjuk a negatív hibák hatását, mivel csak a hiba nagysága érdekel, nem pedig az iránya. Most pedig próbáljuk meg más módon felosztani az adatkészletünket.



Teszteljük az adatsort a második kombinációban. Jól észrevehető, hogy más lesz az értéke a hibaösszegnek.

**<<ide mehet az error\_outputs.txt megfigyelései>>**

Nagy különbség a hibaösszegek között nem ígéretes a modellünk szempontjából. Ez azt jelenti, hogy becslésünk pontossága csak szerencse kérdése. Az, hogy hogyan osztjuk fel adatkészletünket tanító és tesztkészletre, jelentős hatással van a modell pontosságunkra. Azt állítják, hogy ha a tanító és teszt felosztás nagy szórással rendelkezik, akkor nagyon érzékeny lesz a tanító adatainkra. Ennek alapvető oka az, hogy gépi tanulásunkat olyan adathalmazokkal tesztelik, amelyekről fogalma sincs. Jobb megközelítés, ha minden adatsort felhasználunk mind tesztelésre, mind tanításra, majd kiszámoljuk az egyes csoportok átlagos pontosságát. Ez az úgynevezett K-Fold keresztellenőrzés.

**K-Fold keresztellenőrzés**

A K-szeres keresztellenőrzés olyan képzési és tesztelési technika, amelynek célja az adatkészlet használatának optimalizálása a jobb és stabilabb tanítás pontosság elérése érdekében. A következő lépésekből áll [7]:

1. Adatkészletünket K csoportokra osztjuk. Az egyik csoportot tesztkészletként, a többi csoportot pedig edzéskészletként.
2. Képezzük és kiszámoljuk a modell pontosságát a választott képzési és tesztkészletek alapján.
3. Tesztkészletként a következő csoportot, tanítókészletként a többi csoportot választjuk, és megismételjük.
4. Miután kiszámoltuk az összes csoport pontosságát, a tanítórészekből kiszámítjuk az átlagos pontosságot.

**Diák adatok K-Fold keresztellenőrzéssel**

Lássuk ezt működés közben. Vegyünk példát egy négyszeres keresztellenőrzési képzéssel. K-értéket 4-nek választottam az egyszerűség kedvéért, de kiválaszthatnánk bármilyen értéket.



Az első tanításnál az első három csoportot választjuk tanítókészletnek, az utolsó csoportot pedig tesztkészletnek. A kör pontosságát Pontosság[1]-ként fogjuk kiszámítani, míg a második tanítókörnél az első, a második és a negyedik csoportot választjuk tanítókészletnek és a harmadik csoportot tesztkészletnek választjuk. A kör pontosságát Pontosság[2]-ként fogjuk kiszámítani. A harmadik körben az első, a harmadik, a negyedik csoportot választjuk tanítókészletnek és a második csoportot tesztkészletnek. A pontosságot Pontosság[3]-ként számítjuk ki. Végül a negyedik körben az első csoportot választjuk tesztkészletnek, míg a második, a harmadik és a negyedik csoportot tanítókészletnek. A kör pontosságát Pontosság[4]-ként számoljuk.

A modell általános pontossága az 1. pontosság, a 2. pontosság, a 3. pontosság és a 4. pontosság átlaga.

**<<Ide mehet pár megfigyelés error\_outputs.txt KFOLD résznél>>**

Az adatszegregáció alkalmazásakor kevés szempontot érdemes megemlíteni. Az adatszegregációs paraméterek kiválasztásakor két szabály van érvényben. A tanító és teszt felosztás használata esetén ajánlott az úgynevezett Pareto elv követése. Amely szerint az adatok 20% a tesztelésre és 80% a tanításra kell osztani. Szokták még 30% tesztelésre és 70% a tanításra is szétosztani. Míg a k-szeres keresztellenőrzés használatakor a 10-es k értéket ajánlják kísérletezéssel.

Előfordulhat, hogy a hasonló adatkészletek szomszédosok, például a szekvenciális rekordok bejegyzésénél. A rekordok egymáshoz közel vannak, mivel ugyanabban az időkeretben helyezkednek el. Ha az adatkészletet ugyanabban a sorrendben vesszük, fennáll annak a veszélye, hogy az adatkészletet előítéletes módon osztjuk fel. Tehát célszerű az adatkészlet véletlenszerű kiválasztása, hogy elkerüljük az előítéletes felosztást.

Van néhány hátul ütője, ha a K-szeres keresztellenőrzést választjuk. Igaz, hogy a K-szeres keresztellenőrzés valószínűleg képzettebb modellt eredményez a tanító és teszt osztott módszerhez képest. Ez azonban k-szer lassabban megy végbe, mivel a tanítást k-szer hajtjuk végre. A valóban jelentős időbeli különbség a nagyméretű adatoknál jelentkezik.

# Adattisztítási lépések

## Az adatokkal való problémák

**Adat előkészítési problémák**

Általában, ha adatelemzésről beszélünk, akkor elmondhatjuk, hogy az adatok előkészítése rendkívül hosszú ideig tart. Továbbá rendkívül fontos lépés. Ebben a fejezetben részletezni fogom, hogy milyen fontos ellátni jó adatokkal a gépi tanulási modellünket. Hiszen a híres mondás még mindig állja a helyét: „Szemét be szemét ki”.

Ha a gépi tanulási modellbe betöltött adatok rossz minőségűek, maga a modell is rossz minőségű lesz. Ezeket az adatokat a modell tanításához, valamint az előrejelzéshez is felhasználják. Semmilyen módon nem lehet jó gépi tanulási modellt felépíteni, függetlenül attól, hogy milyen algoritmust használunk, ha az adatai nincsenek megfelelően előkészítve. [21]

Rengeteg probléma merül fel, amikor a valós adatokkal dolgozunk. Lehet, hogy éppen nincs elegendő adat ahhoz, hogy elegendő legyen a modell tanításhoz. Lehetséges, hogy túl sok adat áll rendelkezésre, amelyek közül nem mindegyik releváns. Lehet, hogy nem reprezentatív adatok alapján tanítjuk a modellt. Tehát rossz tulajdonságokat táplálunk a gépi tanulási modellünkbe. Ezenkívül a valós világ adatai nyers adatok, nem biztos, hogy mindig tiszták. Előfordulhat, hogy hiányoznak a fontos mezők és rekordok. Megtörténhet, hogy duplikátumok vannak az adatokban, amelyekkel foglalkoznia kell. Végül lehetséges, hogy az adatokat hibák tarkítják, kiugró értékek vagy hiányzó adatok formájában. Tehát egyértelműen vannak problémák az adatokkal.

Hogyan kezeljük ezeket?

### Nem elegendő adat

Olyan modelleknél, amelyek nem elegendő adatokkal vannak kiképezve gyengén teljesítenek az előrejelzésben. Tehát nincs annyi minta, amelyből a modell tanulhatna. Ha csak néhány rekord van a gépi tanulás modellhez, az kétféleképpen végződhet. [12]

* ***Túlillesztés***: Ilyenkor a modell túl sokszor olvas be a kevés adatból és egyszerűen megjegyzi az adatokban meglévő mintákat. A modell memorizál nem pedig általánosít.
* ***Alul illesztés***: Lehetséges, hogy kevés adat esetén a modell alul illeszt az adatokon. Itt a modell túlságosan leegyszerűsödhet, ami azt jelenti, hogy nem igazán értette meg az adatokban létező mintákat.

A nem elegendő adat problémája a valós projektek során gyakori küzdelem. Megeshet, hogy a releváns adatok egyszerűen nem állnak rendelkezésre. Ha mégis előállnak az adatok, akkor gyűjtésének tényleges folyamata időigényes és rendkívül nehéz. A modellek tanításához szükséges adatok hiányával küzdés során nincs nagyszerű megoldás. Egyszerűen több adatforrást kell találni. Várni kell tovább, amíg az adatok előálltak vagy saját kézzel kell begyűjteni azokat.

Van néhány megközelítés, amellyel megoldhatjuk a kis adatmennyiség problémáját. A technika, amely részletezésre kerül, nem minden esetben alkalmazható széles körben. Mit tegyünk, ha nincs elég adatunk, ha kis adathalmazzal foglalkozunk?

* ***Modell leegyszerűsítés***: Választhatunk egy egyszerűbb modell felépítését. Az egyszerűbb modellek kevesebb adat mellett jobban működnek.
* ***Átviteli tanulás (transfer learning)***: Ha neurálishálózatokkal vagy mély tanulási technikákkal dolgozunk, akkor használhatjuk az átviteli tanulást. Egy előre felépített modellt használunk, amelyet aztán módosítunk a meglévő kis adatkészleten.
* ***Adatbővítés***: Megpróbálhatja növelni az adatmennyiséget, amellyel dolgozik, az adatbővítési technikák használatával. Ezt a képadatokkal meglehetősen gyakran használják. Egyszerűen módosítja a meglévő képeket, hogy új képeket kapjon.
* ***Adatszintézis***: És még egy utolsó lehetőség lehet: megértette, hogy milyen típusú adatokra van szüksége a modell felépítéséhez, és ezen adatok statisztikai tulajdonságait használja szintetikus mesterséges adatok előállításához.

**Modell leegyszerűsítés** [9]

Minden gépi tanulási algoritmusnak megvan a saját paraméterkészlete. Figyelembe kell venni, hogy milyen modellt használunk és mennyi adat elérhető. Ha kevesebb adat van, akkor egyszerűbb modellt érdemes választani. Az egyszerűbb modellek kevésbé hajlamosak az adatok túlillesztésére és az adatok mintázatának memorizálására. A túlillesztésre hajlamos modellek jól teljesítenek az tanító adatkészleten, de a való világban gyengén. Nem tanultak az adatokból, egyszerűen megjegyezték a mintákat.

Egy másik lehetőség az, hogy együttes tanítás (ensemble training) alkalmazunk a kis adatkészleten. Az együttes tanítás nem támaszkodnak egyetlen gépi tanulási modellre, sok egyéni modellt tanítanak és a modell végső előrejelzése az egyes modellek összesített előrejelzése. Mivel az együttes tanítás sok egyéni előrejelzés eredményeit összesítik, gyakran jobb teljesítményt nyújtanak, mint az egyéni gépi tanulási modellek. Például logisztikai regresszióra, Random Forest, Naiv Bayes osztályozóra stb. Összevonjuk az egyes modellek eredményeit, hogy megkapjuk a az együttes végeredményt.



**Átviteli tanulás**

Ha neurális hálózatokkal dolgozunk és nincs túl sok adata a modell betanításához, akkor az átviteli tanulás egy lehetőség. Az transzfertanulás azonban csak azokra a használati esetekre alkalmazható, amelyek nagyon gyakoriak. Az átviteli tanulás magában foglalja egy képzett neurális hálózat újra felhasználását, amely megoldja a hasonló problémát.

Például képosztályozásnál szükség van egy modellre, amely képes képosztályozásra. Újra felhasználjuk az architektúrát, valamint a modell paramétereit és egyszerűen elvégezünk egy kis tanítást. Átveszük a modell részét képező ismereteket, majd betanítjuk a modellt az új adatkészletére. Az adatkészlet nem feltétlenül hatalmas, de a modell magában foglalja azt az ismeretet, amelyet az eredeti adatkészletből nyert, amely általában nagyobb. Majd a tanítás után a modell illeszkedik az új adatokra is.



**Adatbővítés**

Térjünk át a következő technikára, amelyet alkalmazhatunk, ha nincs elegendő adatunk: az adatok bővítése. Ilyenkor a meglévő mintákat veszünk és valamilyen módon megváltoztatjuk. A módosítás során új mintákat jönnek létre. Az adatbővítési technikák lehetővé teszik a tanító adatkészlet számának növelését. Általában képadatokhoz használják. Új képek előállításához az eredeti képeket kell manipulálni. Például zajosíthatjuk, méretezhetjük, forgathatjuk. Ezekkel további mintákat is lehet készíteni, amelyekkel taníthatjuk a hálózatunkat.

**Adatszintézis** [10]

Az utolsó lehetőség a szintetikus, mesterséges adatok előállítása. A szintetikus adatok saját problémakörrel járnak. Alapvetően mesterségesen előállított mintákat utánozzák a való világ adatait. Ennek érdekében meg kell értenie az adatok jellemzőit, amelyekre jó modellek felépítéséhez van szükség. Túlmintázhatjuk a meglévő adatmintákat, hogy új adatokat kapjunk vagy más technikákat használhatunk mesterséges adatok előállításához. Lehetnek néhány buktatók is. Bevezethetünk az adatkészletbe a való világban nem létező előítéletet, ami előítéletes modellt eredményez.

### Túl sok adat

A szervezetek tudják, hogy fontosak az adatok, viszont gyakran nem tudják, milyen adatok lesznek fontosak. Végül túl sok adatot gyűjtenek be. Furcsának tűnhet, hogy a túl sok adat is problémát jelent. Az adatok, amelyekkel dolgozunk, kétféleképpen lehetnek túlzottak.

**Dimenzió átka** [9]

Az első a dimenzió átka. A sok dimenzió az jelenti, hogy a példányaink az adatkészletben túl sok jellemzője van. Más szóval az adatrekordoknak túl sok oszloppal rendelkezik. Ha túl sok dimenzióval rendelkezik az adatunk, akkor lényegtelen jellemzők vannak. Nem fontos jellemzők nem igazán segítenek a modellek tanításában.

A dimenzió átka hatalmas téma, amelyet az adattudósok részletesen tanulmányoznak. Két konkrét probléma merül fel, ha túl sok dimenzió áll rendelkezésre.

* Az első az, hogy valamilyen módon el kell döntenie, hogy mely adatok relevánsak valójában. Ez magába foglalhatja a jellemzők statisztikai technikákkal történő kiválasztását vagy a dimenziócsökkentést. Alapvetően oszlopok megszűntetéséről lenne szó.
* Lehetséges, hogy az adatok túlságosan szemcsések ahhoz, hogy az ML modellek felvegyék a mintákat. Ilyenkor a nagyon alacsony szintű adatokat hasznos jellemzőkbe kell összesítenie. Tehát oszlopokat összevonni értelmes módon.

**Elavult adatok** [21]

Lehetséges, hogy elavult történelmi adatokkal kell dolgozni. Túl sok minta vagy rekord van, amelyek közül sok lényegtelen. A történeti adatokkal való munka egyfajta kétélű kard. Mennyire fontosak a történelmi adatok? Ha túl sok olyan történelmi adata van, amely nem igazán jelentős, akkor találkozhatunk az úgynevezett koncepció áramlással. Az idő múlásával a bemeneti adatok (X) és a címkék (Y) között változik a kapcsolat. A gépi tanulási modellek nem tudnak lépést tartani az új bemenetekkel, mert túl sok történelmi adat van. Következésképpen a modell teljesítménye hanyatlik. A koncepció áramlás lényegében azt jelenti, hogy gépi tanulási modell folyamatosan figyeli az elavult, már nem jelentős adatrekordokat. Az elavult tudás felhasználva nem képes jó előrejelzésekre az új mintákra. Tehát, az elavult történelmi adatokkal körültekintően kell használni. A történelmi adatok fontosak, de valamiféle ítéletet kell használnia, hogy kiderítse, valójában milyen rekordok relevánsok.

**Megoldási technikák** [9]

Elavult történelmi adatokkal és a sodródás fogalmával elég nehéz foglalkozni. A dimenzió átka könnyebben megoldható probléma.

* ***Jellemző kiválasztás*** (feature selection): Statisztikai technikák segítségével kiválasztjuk azokat a jellemzőket, amelyek relevánsak számunkra.
* ***Jellemző tervezés*** (feature engineering): A jellemző tervezéssel összevonhatjuk az alacsony szintű és részletezett jellemzőket olyan hasznos jellemzőkbe, amelyek kevésbé szemcsések. Kombinálhatjuk a funkciókat is, hogy javítsuk a prediktív erejüket.
* ***Dimenzió csökkentés***: Elvégezhetünk dimenziócsökkentést. Itt csökkenthetjük adatainak bonyolultságát anélkül, hogy elveszítenénk az információkat. Ennek egyik módja az, ha adatokat új tengely mentén kerül átirányításra. Ezáltal elérjük az adatokban meglévő maximális varianciát.

Amikor ilyen jellegű fejlesztést végezünk az adatok összetettségének csökkentése érdekében, akkor találkozhatunk a koncepció hierarchia kifejezéssel. Ez egy olyan leképezés, amely nagyon alacsony szintű jellemzőket kombináljuk általánosabban használható jellemzőkbe. A koncepció hierarchia magában foglalja az adatok gyűjtését is, hogy az információk kevésbé legyenek túl részletezett formátumban.

### Nem reprezentatív adatok

A nem reprezentatív adatoknak számos megnyilvánulása van. Az egyik az, hogy csak rossz tulajdonságokat táplálunk be a modellbe, de vannak más megnyilvánulásai is. Lehetséges, hogy az összegyűjtött adatok hibákat tartalmaznak. Bizonyos szempontból pontatlan és a hibák olyanok, hogy jelentős hatással lehetnek a modellre. Feltétlenül fontos az adatok tisztítása és feldolgozása azért, hogy az adatok jó állapotba kerüljenek.

Előfordulhat, hogy az adatok nem reprezentatívak, mert előítélettel rendelkeznek. Tegyük fel, hogy öt érzékelőből gyűjt adatokat, öt különböző országban, és van egy olyan érzékelő, amely nem működik mindig. Az adatok előítéletesek, mivel nincsenek arányos adatai az egyik érzékelőtől. Ha előítéletes adatokkal dolgozunk, akkor az előítéletes gépi tanulási modellt eredményez. Ezek a modellek a gyakorlatban gyengén teljesítenek, mivel nem tartják szem előtt az összképet. Enyhíthetünk a túl- és az almintázás segítségével. Tehát, ha kevesebb adata van az egyik érzékelőtől, akkor felülmintázhatjuk azokat az adatokat. Így reprezentatív mintát érünk el. A túl- vagy az alul mintavétel saját elfogultságot eredményezhet, ezért erre ügyelnie kell. [9]

### Hiányzó, kiugró értékek

További problémák, amelyekkel találkozhatunk, az adatok hiánya és a kiugró értékek jelenléte. Az adattisztítási eljárások jelentősen mérsékelhetik mind a hiányzó adatok, mind a kiugró értékek hatását. Egy későbbi fejezetben részletezem azokat a konkrét technikákat, amelyekkel kezelhetjük a hiányzó adatokat és a kiugró értékeket. Folytassuk és beszéljünk az ismétlődő adatokról.

### Másolatok

Ha adatokat gyűjtünk, előfordulhat, hogy vannak duplikátumok. Ha az adatokat másodpéldányként lehet megjelölni, a problémát nagyon könnyű megoldani. Egyszerűen csak másolatot kell alkalmaznia az adatokra, mielőtt betáplálnánk őket egy modellbe, de a világ nem ilyen egyszerű. A másolatokat nehéz lehet azonosítani bizonyos alkalmazásokban, konkrétan a valós idejű streaming alkalmazásoknál. Mivel az idő folyamatosan változik nem tudjuk érzékelni a másolatokat.

## Hiányzó adatok kezelése

Adatok gyűjtése és azokkal való munka során előfordulhat, hogy hiányzó adatok vannak a mezők hiányzó értékei formájában. Illetve olyan kiugró értékeket tartalmazhat, amelyeknek valójában nincs sok értelme. Kétféle megközelítést követhetünk, mikor a hiányzó adatokról beszélünk. [22]

* ***Törlés***: Ezek közül az első a törlés, ahol megszabadul az adatoktól, amelyek hiányzó mezőket tartalmaznak,
* ***Kitöltés***: A másik a kitöltés, ahol a hiányzó értékeket valamilyen technikával kikövetkeztetjük, majd beírjuk a hiányzó helyre.



Először beszéljünk a törlésről. A törlést listás törlésként is nevezik. Itt egy teljes rekord kerül törlésre, amely megfelel az adatkészlet egy sorának. Ha hiányzik egyetlen egy érték egy jellemzőből (oszlop), akkor az egész sor törölni kell. Ez egy egyszerű, problémamentes technika a hiányzó értékektől való megszabaduláshoz. Ugyanakkor előítélethez vezethet, mert megszabadulunk egy teljes rekordoktól, még akkor is, ha csak egy irreleváns mezőnek hiányzik az értéke. A listán belüli törlés a gyakorlatban a legelterjedtebb módszer, mert egyszerű, viszont ez több problémához is vezethet.

Nagymértékben csökkenhet a minta mérete. Ha nincs sok rekord, akkor olyan helyzetbe kerülhet, hogy nincs elegendő adat a gépi tanulási modell betanításához. Például egy jellemző értékei mind hiányoznak, mert az tegyük fel egy másik érzékelő jegyzi fel azokat. Miután kidobunk minden rekordot a bizonyos érzékelőről az jelentős torzítást eredményez.

Tehát világos, hogy a teljes rekordok eldobása, amelyekből hiányzik néhány mező, nem nagyszerű megoldás, ezért a kitöltésre térünk át. Ilyenkor a hiányzó oszlopértékeket egészítjük ki, ahelyett, hogy a rekordokat törölnénk. A hiányzó értékek a már rendelkezésre álló adatokból lehet következtetni. Számos módszer közül választhatunk, amelyek a nagyon egyszerűtől a nagyon összetettig terjednek. [22]

* ***Oszlop átlag***: A lehető legegyszerűbb módszer az oszlopátlag használata. Feltételezheti, hogy a hiányzó érték lényegében megegyezik az oszlopban vagy az adott szolgáltatásban szereplő átlagértékkel.
* ***Oszlop medián***: További nagyon hasonló lehetőségek az adott oszlop mediánértékének vagy az adott oszlop módjának használata.
* ***Interpoláció***: A hiányzó értékek beszámításának másik módja az interpoláció más közeli értékekből. Ez a technika akkor hasznos, ha a rekordokat valamiféle eredendő sorrendben rendezik.

**Egyváltozós kitöltés**

Az egyváltozós kitöltés csak ugyanazon tulajdonság vagy ugyanazon oszlop ismert értékeire támaszkodik.

**Többváltozós kitöltés**

A többváltozós kitöltés viszont minden ismert adatot felhasznál, amire az adatokból hiányzó értékekre kell következtetnie. Például érdemes az adatok más oszlopaiból regressziós modelleket készíteni, hogy előre jelezze az adott oszlop hiányzó értékeit. Ez egy példa a többváltozós kitöltés. Ezt iteratív módon megismétli az összes oszlopban, amelyek azonos értékeket tartalmaznak.

**Kritérium alapú kitöltés**

Vannak más technikák, amelyek a hiányzó értékek kitöltésére is alkalmazhatók, például a kritérium alapú kitöltés. Az összes bejegyzést fontos kritériumok alapján rendezük. Minden hiányzó értéknél felhasználhatjuk közvetlenül a kritérium szerinti előző értéket. Ezt az utoljára átvitt megfigyelésnek nevezzük. Tehát a rekordok rendezése után kitöltjük a hiányzó értékeket az előző értékkel. Ez különösen hasznos az idősoros adatoknál, ahol az időbeli előrehaladásnak van értelme. [22]

**Átlag helyettesítés**

Egy általános technika, amelyet gyakran használnak egyváltozós kitöltésre, hogy minden hiányzó érték helyett az összes rendelkezésre álló érték átlagát helyettesíti. Az átlagos helyettesítés gyengíti az összefüggéseket az adatok oszlopai között. Amikor lényegében azt mondod, hogy ez egy átlagos adatpont, akkor nincs semmi különös, gyengíted a korrelációkat. Ez problematikus lehet, ha kétváltozós elemzést vagy elemzést végzünk két változó kapcsolatának meghatározására. [22]

**Hiányzó adat előrejelzés**

Ha intelligensen módon meg akarjuk jósolni az adatok hiányzó értékeit, érdemes egy gépi tanulás modell segítségéhez fordulni. A modell előre jelzi a hiányzó oszlopokat más oszlopértékek alapján. Ennek a technikának az alkalmazása megmutatja az adatokban meglévő összefüggéseket. Lényegében látni fogjuk, hogy ez az oszlop függ a többi oszloptól. Így a regressziónak és az átlagos helyettesítésnek gyengítő hatásai vannak. Tisztában kell lennie a technikák alkalmazásának árnyalataival, és meg kell határoznia az adott felhasználási esetet.

## Kiugró értékek kezelése

Az adatainak feltárása és vizualizálása során előfordulhat, hogy az adatkészletben vannak kiugró értékek. Ez egy olyan adatpont, amely jelentősen eltér az ugyanazon adatkészlet többi adatpontjától. Előfordulhat, hogy a teljes rekord valamilyen módon kiugró érték, vagy vannak olyan mezők, amelyek kiugró értékekkel rendelkeznek.

Két lépést tudunk elkülöníteni, mikor kiugró adatokkal dolgozunk. Az első lépés az adatokban létező kiugró értékek azonosítása. A második lépés ezeknek a kiugró értékeknek a kezelése.

Csakúgy, mint vannak gépi tanulási algoritmusok, vannak olyan speciális algoritmusok is, amelyeket a kiugró értékek észlelésére építettek. A legalapvetőbb azonosítása a kiugró értékeknek, hogy megmérjük az adott adatpont távolságát az adatok átlagától. Másik módszer, hogy vesszük az adatpont és a modell által illesztett vonal távolságát. Miután azonosítottuk a kiugró értékeket, három tág technika segítségével megbirkózhatunk a kiugró értékekkel. Eldobhatjuk a kiugró adatokat, korlátozhatja a kiugró értékeket vagy beállíthatja a szélső értékeket az átlagos értékre. [22]



**Észlelés**

Kezdjük azzal, hogy megbeszéljük, hogyan tudjuk azonosítani a kiugró értékeket a két megvitatott technika segítségével.

**Távolság az átlagtól** [23]

Az adatok bármely jellemzőjének átlaga alapvetően a központi irányvonal mérőszáma. Ez az a pont, amely körül a fennmaradó pontok csoportosulnak. Egy adatpont, amelynek értéke messze van az átlagtól, azt kiugrónak tekinthető.

Ha gyorsan összefoglalni szeretnénk az összes olyan adatsort, amellyel dolgozunk, akkor az első mérőszám az adatok átlaga. Bármely adat átlagos értéke az a szám, amely az összes adatpontot a legjobban képviseli. Bármely adatpont átlaga lényegében az összes szám összege elosztva a darabszámmal.

Az átlaggal együtt azonban az adatokban meglévő variáció is fontos. A variáció annak mérése, hogy az adatpontok mekkora tartományba mozognak. Az adatok változásának egyik fontos mértéke a tartomány, amely egyszerűen a maximális és a minimális érték levonása. Azonban a tartomány teljesen figyelmen kívül hagyja az átlagot, ezért gyakran alkalmaznak egy másik mérőszámot, amely a variancia.

Az adatok második legfontosabb mérőszám, amely összefoglalja az összes pontkészletet a variancia. A variancia annak mérése, hogy az adatok hogyan változnak az átlagtól. Röviden összefoglalják a számok halmazát, bármilyen számkészletet. A varianciával együtt egy másik kifejezés, amellyel találkozhatunk, a szórás.

A szórás nem más, mint a variancia négyzetgyöke, és az adatok változásának mértéke. A szórás segít kifejezni, hogy egy adott adatpont milyen messze van az átlagtól.

Azokat a pontokat, amelyek háromnál több szórást jelentenek az átlagtól, gyakran kiemelkedőnek tekintünk. A kiugró értékek standard eltérési küszöbértéke gyakran a felhasználási eseten alapul.

**Távolság az illesztett vonaltól**

Az adataiban szereplő kiugró értékek azonosításának másik módja az, ha megmérjük azok távolságát a modell által illesztett vonaltól. Tegyük fel, hogy van egy kétdimenziós adathalmazunk. Ebben az esetben a modellünk megpróbál egy sort illeszteni valamilyen regressziós algoritmus segítségével. A kiugró értékek lényegében olyan adatpontok, amelyek nem távol vannak az illesztett vonaltól vagy görbétől.

**Kezelés** [23]

Miután azonosítottuk a kiugró értékeket, ki kell találni, hogyan akarunk velük megbirkózni. Igazából nincs konkrét megoldás, számos feltétel szabadhatja meg a kiugró adatok kezelését. Meg kell vizsgálnia és meg kell értenie az adatokban található kiugró értékeket. Ezek a kiugró értékek helytelen megfigyelések vagy az adatokban előforduló hibák miatt vannak? Ekkor érdemes lehet teljesen megszabadulnia ettől a rekordtól, ha a rekord összes attribútuma hibás.

Ha egy sorra vagy egy rekordra vonatkozik, ha úgy érzi, hogy csak egy attribútum van tévesen rögzítve, akkor érdemes ezt a kiugró értéket átlagnak beállítani, és nem kidobni a rekord egészét.

Feltéve, hogy modellt nem torzítják a kiugró értékek jelenléte, akkor hagyjuk úgy, ahogy van.

A kiugró értékek fontos információkat közvetítenek, amelyeket a modelljének esetleg fel kell ismernie. Ehhez szükség lehet arra, hogy először standardizálni az adatokat. A standardizálás az összes adatpontot kifejezze az átlagtól való eltéréssel. A standardizálás magában foglalja az összes értékből az átlag levonását, így az eredményül kapott érték skálázott. A felső kiugró értékeket pozitív három standard eltérésként határozhatjuk meg az átlagtól. Az alsó kiugró értékeket negatív három standard eltérés lehet.

**Az adatok kiegyensúlyozása (túl- és alul mintavételezés)**

placeholder (meg lehetne említeni?!?!?)

## Numerikus és kategorikus adatok

Az adatokat két nagy kategóriába lehet sorolni, a kategorikus és a numerikus adatok. Mi a különbség a kettő között?

**Kategorikus adatok** [22]

A kategóriák a megengedett értékek véges halmaza. A diszkrét értékekből vagy kategóriákból álló adatok további két alkategóriára oszthatók, a nevesített és a sorrendi adatokra. A sorrendi adatok rendezhetők a nevesített adatok, pedig nem.

A kategorikus adatokat numerikusan kódolni, mielőtt az modellekben felhasználható lenne. Két konkrét technika a címkekódolás és az egycsatornás kódolás.

A kategorikus adatok, amelyek kategóriákból állnak például: férfi, nő, az év hónapja, a hét napja.

A kategorikus értékek osztályozásra használhatóak, mint a K-Nearest Neighbor.

**Numerikus adatok** [22]

A numerikus adatok tetszőleges értéket vehetnek fel egy végtelen tartományból. Amikor a numerikus adatok különböző skálán vannak, akkor nem igazán működnek jól a gépi tanulási modellek. Különféle technikák segítségével méretezhetjük a numerikus adatokat.

A numerikus értékek mindig nagyságrend szerint rendezhetők, sajátos rendezéssel vagy rangsorolással rendelkeznek. Ezért a numerikus értékeket folyamatos adatoknak is nevezik. Például a súly (kg), a hőmérséklet (Fahrenheit), a lakásárak.

Numerikus adatok a folyamatos értékek előrejelzésére használható, mint a regressziós modellek.

Minden más típusú adatot, a szöveges adatokat, a kép adatokat, át kell alakítani ezen adattípusok egyikévé.

## Adattisztítás megvalósítása

### Hiányzó értékek kezelése

Az alábbi bekezdésbe egy olyan adatsorral fogunk foglalkozni, amelyre adattisztítási technikákat kell alkalmazni. Az adatsor kezdeti formába nem áll készen a gépi tanulásra. Megtisztítjuk ezt az adatsort, majd feltárjuk, mielőtt illesztenénk egy regressziós modellt.

Először be kell olvasni be az adatainkat tartalmazó csv fájlt. Adataim a 2019\_PublicDataFile\_Students.csv fájlban találhatók az adatkészletek mappában. Az adatkészlet eredeti forrás .gov oldalán található. Több mezőben hiányzó értékek találhatóak.

Ha megnézzük a betöltött adatkészletet, akkor láthatjuk, hogy 11 jellemzőből és 1829 rekordból áll. Meg kell állapítani meg a hiányzó értékeket, majd összegezni azokat. Az egyik jellemző összes rekordja üres, ezért azt kidobjuk.



Látható, hogy több mint több mint 1000 üres mező van. Különböző technikák segítségével kezelhetjük ezeket a hiányzó értékeket. Ha elegendő adat áll rendelkezésre, akkor ez egy teljesen érvényes lehetőség. Figyelembe véve az oszlopok hiányzó értékeinek számát a „Trust Score” jellemzőnek van a legkevesebb üres mezője. Ráadásul ez a jellemző függ legjobban az összes többi oszloptól. Továbbá ez egy célváltozó, egy jellemző, amit végeredményt reprezentálja. Tehát itt egyszerűen eldobom az üres mezőket tartalmazó rekordokat.

Egy másik érvényes stratégia, hogy minden hiányzó mezőt helyére az átlagot helyettesítem be. Ami a „Strong Family-Community Ties Score” oszlopot illeti, az összes hiányzó értéket az adatok jelenlegi átlagával fogom pótolni. Mivel nincs nagyon sok hiányzó mező, így az átlaggal való kitöltés kézenfekvő. Hasonló módon jártam el a „Collaborative Teachers Score” és az „Effective School Leadership Score” jellemzőknél.

A „Rigorous Instruction Score” oszlop esetén a leggyakrabban szereplő értéket helyettesítettem be.

Kifinomultabb megközelítés az IterativeImputer osztály használata, amely minden tulajdonságot hiányzó értékekkel modellez a többi jellemző függvényében, és ezt a becslést használja a kitöltéshez. Round-robin algoritmus szerint iterál egy megadott ismétlés számig.

Az üres értékek eldobása és kitöltése után 1694 rekord van, szemben a kezdeti 387 rekordunk van, szemben a kezdeti 1829 rekorddal. Tehát 135 rekord került ki az adatkészletből, ami nem annyira számottevő. Ennek ellenére nullára csökkentettük az üres értékek számát.



### Címkekódolás és egycsatornás kódolás

Hogyan lehet konvertálni a kategorikus adatokat numerikus formába a címkekódoló és az egycsatornás kódoló segítségével. Egy másik diák adathalmaz kategorikus adatok átalakítása következik. A konkrét adatkészlet eredeti forrása az alábbi URL-címen található: <https://www.kaggle.com/janiobachmann/math-students>.

Ha megnézzük az közelebbről, látható, hogy számos kategorikus jellemzővel rendelkezik. A nem, az internet hozzáférés, a család mérete. Az adatkészlet 395 rekordból és 33 jellemzőből áll.

**Címkekódolás** [24]

Vannak olyan kategorikus adatok, amelyek két diszkrét értéket vehetnek fel. Ilyen például: nem, családi támogatás, romantikus kapcsolat.

Ezeket egyszerű címkekódolással átalakítom numerikus adattá. A LabelEncoder osztály segítségével könnyedén transzformálhatók az adatok. Újra felhasználhatóság érdekében egy függvénybe szerveztem ki a címkekódolást. A címkekódoló 0-val és 1-gyel helyettesíti a kategóriákat.



**Egycsatornás kódolás** [7]

Egy másik kategorikus jellemző, a ’gyám’ (guardian). Ha megnézünk egy mintát, látható, hogy a gyám lehet apa, anya vagy egyéb. Ha az adatok kategóriáiban nincs implicit sorrend, akkor érdemes lehet egycsatornás kódolással használni (one-hot encoder). Ilyenkor különböző oszlopokba kerülnek a kategóriák, a mi esetünkbe: guardian\_father, guardian\_mother, guardian\_other. Ha a diák gyámja az apja házas, akkor a guardian\_father oszlop alatt 1-es érték és az összes többi oszlop értéke 0 lesz. Ezt követően az eredeti ’gyám’ oszlopot eldobhatjuk az adatkészletből. Majd beillesztjük a kódolt változatott.

Egyszerűbb módja az egycsatornás kódolásnak, ha a panda keretrendszerbe létrehozott „get\_dummies” függvényt használjuk. Csak át kell adnia az eredeti adatkeretet, és meg kell adnia azt az oszlopot, amelyet egyszerre szeretne kódolni. Ha nem szeretnék az összes kategorikus értéket egycsatornás módon kódolni, akkor elég, ha csak az adatkeretet adjuk át. Feltérképezi, majd hozzáadja a kódolt oszlopokat az adatkészlethez. A függvény gondoskodik az eredeti kategorikus oszlop eltávolításáról is.



### A jellemzők méretezése és standardizálása

### Jellemzők normalizálása

### Folyamatos értékek diszkretizálása

# Irodalomjegyzék

|  |  |
| --- | --- |
| [1] | T. M. M. Arthur Samuel, „Machine\_learning: Wikipedia,” 16 December 2020. [Online]. Available: https://en.wikipedia.org. |
| [2] | D. Cielen és A. Meysman, Introducing Data Science, New York: Manning Publications, 2016. |
| [3] | P. Blanes, „Search: Flaticon,” 2019. [Online]. Available: https://www.flaticon.com. |
| [4] | A. Adesina, „Data is the new oil: Medium,” 13 Nov 2018. [Online]. Available: https://medium.com/. |
| [5] | M. Heller, „What is machine learning? Intelligence derived from data: Infoworld,” 15 May 2019. [Online]. Available: https://www.infoworld.com/. |
| [6] | S. Weidman, Deep Learning from Scratch, USA: O’Reilly Media, 2019. |
| [7] | A. C. Müller és S. Guido, Introduction to Machine Learning with Python, USA: O’Reilly Media, 2017. |
| [8] | D. Silver, „Teaching: david silver,” University College London, 31 May 2015. [Online]. Available: https://www.davidsilver.uk. |
| [9] | A. Geron, Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow, USA: O'Reilly Media, 2019. |
| [10] | E. Ameisen, Building Machine Learning Powered Applications, USA: O'Reilly Media, 2020. |
| [11] | P.-N. Tan, M. Steinbach és V. Kumar, Introduction to Data Mining, Edinburgh Gate: Pearson Education, 2014. |
| [12] | J. Grus, Data Science from Scratch, USA: O'Reilly Media, 2015. |
| [13] | S. H. Mohit Gupta, „Introduction Data Machine Learning: Geeks for geeks,” 17 May 2020. [Online]. Available: https://www.geeksforgeeks.org/. |
| [14] | T. Naeem, „Blog: Astera website,” 4 November 2020. [Online]. Available: https://www.astera.com/type/blog/structured-semi-structured-and-unstructured-data/. |
| [15] | W. Badr, „Top Sources For Machine Learning Datasets: Towards Data Science,” 13 Jan 2019. [Online]. Available: https://towardsdatascience.com. |
| [16] | H. Timothy, „Blog: Great learning web site,” 11 May 2020. [Online]. Available: https://www.mygreatlearning.com/blog/sources-for-analytics-and-machine-learning-datasets/. |
| [17] | N. Noy, „Blog: Google,” 5 Sep 2018. [Online]. Available: https://www.blog.google/products/search/making-it-easier-discover-datasets/. |
| [18] | C. N. Knaflic, Storytelling with data, New Jersey: John Wiley & Sons, Inc., 2015. |
| [19] | J. Brownlee, „Data, Learning and Modeling: Machine learning mastery,” 6 January 2017. [Online]. Available: https://machinelearningmastery.com. |
| [20] | H. I. Rhys, Machine Learning with R, the tidyverse, and mlr, New York: Manning Publications, 2020. |
| [21] | Q. E. McCallum, Bad Data Handbook, USA: O’Reilly Media, 2013. |
| [22] | W. McKinney, Python for Data Analysis, USA: O’Reilly Media, 2017. |
| [23] | J. Brownlee, „How to Remove Outliers for Machine Learning: Machine learning mastery,” 25 April 2018. [Online]. Available: https://machinelearningmastery.com/. |
| [24] | D. Yadav, „Categorical Encoding using Label Encoding and One Hot Encoder,” 9 Dec 2019. [Online]. Available: https://towardsdatascience.com/. |
| [25] | T. Mills, „AI.io: Medium web site,” 3 Jul 2019. [Online]. Available: https://medium.com/ai-io/why-big-data-and-machine-learning-are-important-in-our-society-b4e708d2c654. |