Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Hálózati Rendszerek és Szolgáltatások Tanszék

Bereczki Norman Zoltán

VISSZAÉLÉSEK DETEKTÁLÁSA

gépi tanulás módszerekkel

KONZULENS

Dr. Wiandt Bernát

BUDAPEST, 2020

A kapott adatbázisok teljes mértékben tisztítottak és GDPR-nak megfelelők, így nem meghatározható belőlük, hogy az IBM Cloud világának mekkora szeletét fedik le, valamint az sem, hogy az adatok valójában mikor kerültek felvételre, illetve mennyire tükrözik a jelenlegi felhasználóhalmazt.

A dolgozatban felhasznált adatokat az IBM Budapest Lab oktatási céllal készítette el. Elem, illetve tulajdonságszámuk nem egyezik meg a valóban használt adatbázisokéval. Az értékek szintetikusak, csak eloszlásuk felel meg a valóságban rendelkezésünkre álló adatokkal.

Adatvédelmi szempontokból a tulajdonságok neveit, valamint pontos értékeket a dolgozat nem jelenít meg, mindenhol egy megközelítőleges érték van megadva.

Tartalomjegyzék

Összefoglaló

Az IBM Cloud egy piacvezető felhőalapú szolgáltatás cégek számára. A platform nem csak azért hasznos, mert erőforrás allokációt tesz lehetővé, hanem különböző szolgáltatásokat is a felhasználók rendelkezésére bocsát, ezzel is könnyítve, hatékonyabbá téve a munkát. Ilyen például a folytonos backup, valamint a fejlett Watson AI, mely egy áttörő mesterséges intelligencia rendszer.

Az IBM Cloud nyílt, bárki számára elérhető. Ebből kifolyólag elkerülhetetlen, hogy a szolgáltatást nemkívánatos, kártékony felhasználók is igénybe vegyék. Azokat a személyeket tekintjük kártékonynak, amelyek valamilyen módon (pornográf tartalom streamelésével, adatbányászattal, a cég hírnevében való kártétellel, valamint az allokált erőforrás utáni díjak kifizetésének elmulasztásával) kárt okoznak a cégnek.

Az IBM Cloudon regisztráló felhasználókról IBM által gyűjtött, valamint harmadik féltől származó adatok is rendelkezésre állnak. Ezeket használja fel a cég a regisztráció során a nemkívánatos felhasználók kiszűrésére. Jelenleg ez a munka manuális módon, elemzők által zajlik.

Munkám célja egy olyan automatizált rendszer fejlesztése, amely ezen adatokra támaszkodva képes felismerni a kártékony felhasználók viselkedésének mintáját, így már a regisztrációs folyamat során képes kiszűrni azokat, automatizálva ezzel a munkát.

Abstract

IBM Cloud is an industry leader in cloud based services for companies. The platform is useful, because it provides efficient resource allocation and several useful functions, like creating automated backups and Watson AI (a complex and scalable AI technology).
Registering on the IBM Cloud is open for everyone. Thus its unavoidable, that fraudulent registrations will happen. We label users as fraud, who somehow cause harm to the company: not paying their bills or engaging in illegal activities like storing / streaming child pornography or mining cryptocurrencies.

IBM collects data for each and every registration to its cloud platform.The data is mostly collected by IBM, but there are several third party services providing information about the users. The data collected is used to determine whether a registered user is likely fraudulent or not. Currently, this work is done manually by analysts.
My goal is to develop a decision support system, that can recognise anomalous behavioural patterns and classify accounts based on the available data.

1 Bevezetés

Az elmúlt évtizedben exponenciálisan ugrott meg az emberek által generált adatmennyiség az interneten és ennek a növekedésnek a mértéke egyre rohamosabban nő. A weben található adatok mintegy 90%-át az elmúlt két évben hoztuk létre. Manapság közel 2 milliárd aktív weboldalt tartunk számon, valamint egyidőben átlagosan 4.2 milliárd ember van online[14]. Az IBM becslései szerint naponta közel 2.5 trillió bájtnyi adatot generálunk. Ennyi adat feldolgozása nagyon erőforrásigényes, hiszen a velük végzett számítások nagyon sok lépésből állnak. Ugyan a technológiai innovációk is jelentősek voltak az elmúlt években, de a személyi számítógépek számítási teljesítményének fejlődése elmaradt az adatmennyiség növekedéséhez képest.

Ahhoz, hogy emberi időben kezelhessük ezeket az adatokat és kiszolgáljuk nagy erőforrásigényünket, erős, ugyanakkor drága hardware-ekre van szükségünk. Ezek a szuperszámítógépek nem elérhetőek bárki számára, valamint itt is fizikai korlátokba ütközhetünk. (Például egy adott méretű integrált áramkörre csak limitált mennyiségű tranzisztort tehetünk.) Erre a problémára nyújtanak megoldást a felhőszolgáltatások. Ezen fogalom alatt az igény szerinti számítógépes erőforrások allokálását lehetővé tevő szolgáltatásokat értjük. Ilyenek lehetnek például adattárolással kapcsolatos szolgáltatások, hosztolt alkalmazások, IaaS, PaaS, SaaS rendszerek. A legtöbb ilyen szolgáltató ’pay-as-you-go’ elven működik, mely azt jelenti, hogy csak az általunk igénybe vett szolgáltatások díját kell kifizetnünk. Jellemzőjük még a rugalmasság, a magas hibatűrés és biztonsági szolgáltatások nyújtása, illetve a skálázhatóság.

Az IBM Cloud-ra történő regisztráció ingyenes, ebből kifolyólag elkerülhetetlen, hogy nem kívánatos aktivitást folytató személyek is megjelenjenek a platformon. Jelenleg ezen személyek kiszűrése manuális módon, elemzők által zajlik. Dolgozatom célja egy olyan rendszer fejlesztése, mely ezt a munkát automatizálni tudja, ezzel segítve, javítva az elemzők munkáját.

1.1 A dolgozat részletes célja

A felhasználók közül az esetek túlnyomó részében már a regisztrációs folyamatuk során kiszűrhető az, hogy céljuk-e kártékony tevékenységet folytatni a platformon. Ez a

felhasználókról gyűjtött tulajdonságok szabályszerűségeiből, bizonyos mintázatot követő viselkedésükből érzékelhető. A regisztráló felhasználókról az IBM is gyűjt adatot a regisztrációs folyamat során, valamint harmadik féltől származó adatokat is használ a végső döntés meghozatalához.

A dolgozat készítése során két adathalmaz állt rendelkezésemre. Egy célváltozóval ellátott, valamint egy célváltozó nélküli. Előbbiben olyan esetek találhatók, melyeket már elemzők manuális módon értékeltek és a célváltozó jelöli, hogy az adott esetet anomáliának osztályozták-e, vagy sem. A célváltozóval ellátott adathalmaz mérete jelentősen kisebb, mint a célváltozó nélküli adathalmazé, mert az egyes esetek manuális vizsgálata idő és erőforrásigényes tevékenység.

Dolgozatom célja egy olyan rendszer elkészítése, mely a jelenleg manuálisan folyó munkát automatizálni, minőségét javítani tudja. Ehhez az adatok megfelelő előfeldolgozása, azok mélyebb megértése, analízise szükséges. Ez után gépi tanuláson alapuló modelleket illesztek az adatra, mely segít eldönteni egy adott esetről, hogy a felhasználó kártékony-e. Célom az elemzők által hozott döntések vizsgálata, a címkék javítása, majd ennek visszacsatolása a tanító adathalmazra. Ennek segítségével az elemzői munka és az osztályozó algoritmus pontosságának növelése. A végső cél egy olyan rendszer létrehozása, mely teljesen determinisztikus módon működik, szemben az elemzők heurisztikus döntéseivel, valamint megvizsgálni, hogy az elemzők által adott címkék helyesek-e és ha nem, akkor a kapott információt visszacsatolni, hogy a rendelkezésre álló tanítóhalmaz és ezáltal a tanított modellek minősége is javulhasson.

1.2 A feladat nehézségei, megoldandó problémái

Elsődlegesen megoldandó probléma a megfelelő adatbázis struktúra kialakítása. Az adatok IBM Cloudant-ban vannak tárolva, amely egy NoSQL alapú adattároló platform. Mivel az egyes regisztrációs esetek tulajdonságai több forrásból kerülnek összegyűjtésre, így struktúrájuk nem egyezik, nehezen kezelhetőek együtt.

A címkézett és a címkézetlen adathalmaz erősen eltérő elemszámmal rendelkezik, hozzávetőlegesen 2.7-szer több címkézetlen eset áll rendelkezésemre, így a tanító adathalmaz mérete kicsi. Ráadásul a címkézett esetek közül az anomáliának, valamint a nem anomáliáknak jelölt esetek aránya sem azonosan oszlik meg, az esetek mindössze 20%-a lett anomáliának megjelölve.

A dolgozat készítése során az IBM Cloudant-ból kinyert adatokkal dolgoztam. Az általam használt adatbázisok struktúrájuknak felépítése, szervezésük kialakítása, valamint a domain mélyebb megismerése a dolgozat elkészítési idejének egy jelentős hányadát tette ki, hiszen az adatbázison még senki sem végzett műveleteket, nekem kellett az egyes rekordokat kezelhető formára hozni, majd azokból információt kinyerni, olyan rendszerbe szervezni, hogy az a gépi tanulás eszköztárával felhasználható legyen. A címkék nem explicit mód meghatározottak az eredeti adaton, hanem különböző tulajdonságokból lehetett őket kinyerni.

Az címkézett adathalmazon található osztályzást az elemzők készítették. Ezzel egy osztályozó algoritmus pontosságának felső határába ütközünk, hiszen maximum olyan precíz algoritmust lehet ezen címkék alapján fejleszteni, amilyen precízen az elemzők dolgoztak, hiszen az ő munkájukat modellezi le.

2 Irodalmi áttekintés

2.1 Anomália detekció, fraud analízis

Az anomália detekció napjaink egyik sokat kutatott témája, hiszen széles körben, nagyon különböző területeken alkalmazzák. Többek között a kiberbiztonságban, bankkártya validálásnál, biztosítási esetek elbírálásakor, valamint egészségügyi képek feldolgozása során is. Kutatását már a 19. század elején megkezdték a matematikai statisztika területén és azóta fejlesztenek modelleket[18]. Egyes modellek specifikusak a felhasználási területre, mások egy általánosabb megoldást nyújtanak. Az anomália detekció lényege, hogy mintaszerűséget vegyünk észre azon adatok között, melyek viselkedése nem egyezik meg az elvárt viselkedési móddal. Az ilyen, más viselkedést tanúsító adatokat legtöbbször anomáliának, fraud adatnak, illetve kivételnek nevezzük. Példa lehet erre, ha egy MRI felvételen eldöntik egy daganatról, hogy az jó- vagy rosszindulatú, vagy egy szerver hűtési modulja a megszokott teljesítményének sokszorosán működik, melyből a szerver nagy igénybevételére, esetleges támadására következtethetünk. Fontos elkülöníteni az anomáliákat és a zajt egymástól. Zajként definiálhatjuk azokat az adatokat, melyek nem érdekesek a vizsgálat szempontjából. Adathalmazaim minden eleme célzottan lett gyűjtve, így azt feltételeztem, hogy zajjal az esetemben nem kellett számolnom.

Az anomália detekció során számos nehézségbe ütközhetünk, megfontolt tervezést, a szabályok pontos definiálását igényli. Sok esetben nem egyértelműen meghatározható, hogy mi a szabályszerű viselkedés, a nem szabályos minták nem mindig mutatnak egyértelmű eltérést az elvárt működéstől. Gondot okoz még a trendek folytonos változása, így lehet, hogy amit egyszer szabályosnak ítéltünk meg, az idő előrehaladtával már nem mondható annak. Az anomáliák, kivételek gyakran adaptálódnak az alkalmazott szűrő szabályaihoz, hogy az osztályozás alapján ők is megfelelő viselkedést tanúsítsanak. Egy anomália detekciós algoritmus megalkotásához rengeteg paramétert figyelembe kell venni, mint például a viselkedési szabályokat, a rendelkezésre álló adat mennyiségét, az adat természetét, valamint meg kell határozni, hogy milyen kimenetet várunk. Léteznek rendszerek, melyek egy adott skálán az egyes tulajdonságok alapján egy pontszámot rendelnek a rekordokhoz. Az általam választott módszer is ilyen, azonban a pontszám

alapján nem egy valószínűséget, hanem egy definiált küszöbérték segítségével egy címkét ad eredményül, mely megmondja egy adott esetről, hogy az anomáliának számít-e vagy sem.

Címkézettség szempontjából három féle anomália detekciós módszerről beszélhetünk[18]:

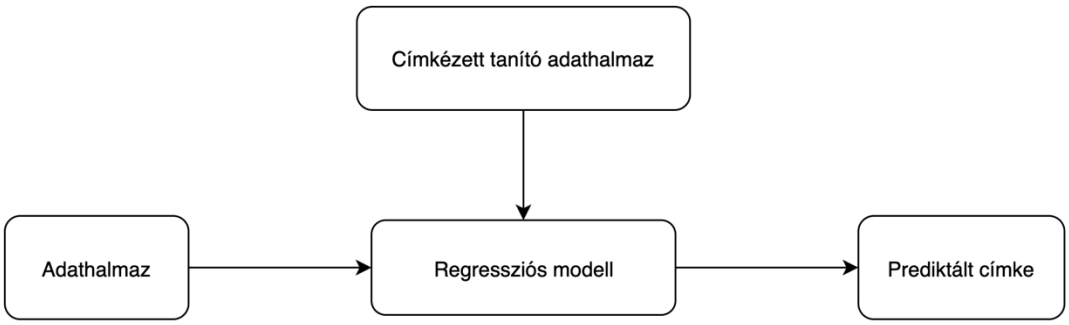
- Felügyelt anomália detekció: Olyan adatokhoz alkalmazható, ahol az anomália, valamint a normális viselkedést mutató címkék is rendelkezésünkre állnak. Az én adataim egyik fele is ebbe a kategóriába tartozik, hiszen az egyik adatbázis teljesen címkézett.
- Félig felügyelt anomália detekció: Olyan adathalmazokhoz alkalmazható, ahol csak a normális viselkedést jelző címke áll rendelkezésünkre. Az anomália mivoltot jelző címke hiánya nem feltétlenül jelenti, hogy az adott eset nem illeszkedik a megfelelő mintába. Jelentheti azt is, hogy ezt még nem sikerült igazolni.
- Felügyelet nélküli anomália detekció: Olyan adathalmazokon használatos, melyek esetében nincs információnk arról, hogy a benne lévő adatok anomáliák-e, vagy sem. Ilyenkor az esetek túlnyomó részében valamiféle tulajdonságok menti csoportosulásokat vizsgálunk.

Az anomália detekció egyik ága a fraud analízis. Ez egy kártékony tevékenységek detektálására szakosodott terület, melyet a leggyakrabban banki szférában, utalásoknál, biztosításoknál, valamint kritikus rendszereknél használnak valamiféle, a szokványostól eltérő viselkedésforma detektálására. Főként adatbázis kezeléssel, statisztikával, adatelemzői tevékenységekkel, valamint gépi tanulással kapcsolatos tudásokra épít. Kezdetben a fraud analízis csak az adatok karakterisztikájának elemzésével történt. Mára, a gépi tanuláson alapuló technológiák elterjedte lehetővé tette, hogy nem csak emberek által, manuálisan elemezzünk adatokat. Ezek a technológiák az adat mélyebb, jobb eredményeket produkáló elemzését teszik lehetővé. Mind felügyelt és felügyelet nélküli gépi tanulásos modellek is használatosak a területen. Jellemzően felhasználói profil készítésével/pontozással, csoportosulások észlelésével, klaszterezéssel, regresszióval, valamint klasszifikálással detektáljuk a csaló tevékenységeket.

2.1.1 Regresszió

A regresszióanalízis célja a függőségek feltárása és mértékének megbecsülése a tulajdonságok, illetve a célváltozó között. A tulajdonságok független, míg a célváltozó függő változók. Egy változót akkor tekintünk függőnek, ha értéke több, független változó segítségével valamilyen hipotézis segítségével meghatározható.

A regressziós modellek célja egy olyan függvény megtalálása, mely a neki megadott tulajdonságokból minél pontosabban képes kiszámítani a keresett célváltozó értékét.



1. ábra: Regressziós modell működése

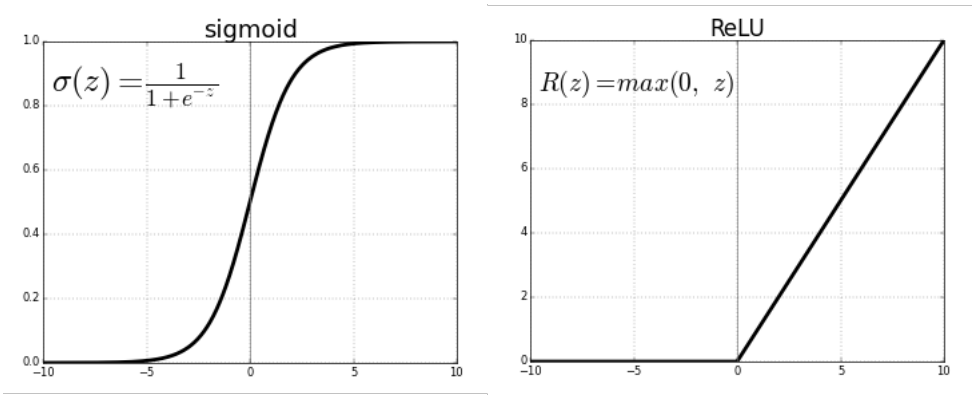
A regressziót két területen alkalmazzák leginkább. Az egyik, mikor adott tulajdonságok alapján meg kell jósolnunk egy bizonyos tulajdonság értékét. (Ilyen példa lehet, ha rendelkezésünkre állnak egy házról különböző adatok, például mérete, szobák száma, a környék biztonsága 1-5-ig pontozva, stb., ebből pedig meg szeretnénk becsülni a ház értékét. Az időjárás előrejelzések is regressziós modellek segítségével készülnek.) Másik terület, amikor függő, illetve független változók kapcsolatát szeretnénk kiszámítani. Ez az úgynevezett korrelációanalízis. Általa ismereteket kaphatunk arról, hogy egy adott változó, vagy változóhalmaz megváltozása milyen mértékben határozza meg a célváltozó értékét. A regressziós modellek egy tanító adathalmazt alapul véve az egyes tulajdonságok lineáris kombinációját alkotják meg, mely a bemenő adaton alkalmazva visszatér egy értékkel, melyből a prediktált változót kiszámítja (lásd 1. ábra). A regresszió két általam használt fajtája a lineáris, illetve a logisztikus regresszió. A lineáris regresszió csak lineáris összefüggések kezelésére képes.

Dolgozatomban a logisztikus regresszió módszerét fogom alkalmazni.

2.1.1.1 Logisztikus regresszió

A logisztikus regresszió célja egy olyan függvény meghatározása, mely egy adott esemény bekövetkeztének valószínűségét határozza meg. Egy gyakori felhasználási területe a különböző orvosi képfeldolgozó módszerekben van, segítségével ajánlást kaphatnak arról, hogy egy daganat mekkora eséllyel rosszindulatú. Ehhez egy döntési határértéket kell jól definiálnunk, amely segítségével meghatározható, hogy a logikai függvény adott értéke milyen kimenetet jelöl.

Ez a valószínűségi függvény leggyakrabban valami féle szigmoid-függvény, bár egyre elterjedtebb a ReLU függvény is (lásd 2. ábra), hiszen a szigmoid függvény hátrányát, vagyis azt, hogy a két szélsőérékhez közel felvett értékei esetében csak nagyon kicsit képes változni, kiküszöböli. Ráadásul a ReLU 0 feletti pontjaiban a függvény deriváltja konstans, ez is elősegíti a gyorsabb tanulást.



2. ábra: A két leggyakrabban használt aktivitási függvény képe

2.1.2 Klaszteresés

Klaszterezés során gyakran nem áll rendelkezésünkre célváltozó, ilyenkor felügyelet nélküli tanulásról beszélünk. Feladata, hogy bizonyos tulajdonságok melletti csoportosulásokat detektáljon és ez alapján klaszterekbe rendezze az adatpontokat.

Klaszterezés alapú anomália detekciós algoritmusok lényege, hogy megkeressük azokat a csoportosulásokat, melyek normális viselkedést tanúsítanak, így láthatjuk, hogy mely rekordok nem illeszkednek bele ebbe a mintába. Egy másik mód, hogy egy igazoltan anomália adatponthoz keresünk közeli adatokat és azokat vizsgáljuk.

2.1.2.1

2.1.2.2 Random Forest

A random forest algoritmus a döntési fák működésének elvét veszi alapul. A döntési fa egy olyan, döntések modellezésére használatos eszköz, melynek vizuális reprezentációja egy fa gráf. Egyes pontjai állapotokat, élei pedig az állapotból való döntést reprezentálják. Tartalmazhatnak még az élek bekövetkezést jelző valószínűségi változókat, valamint pozitív, illetve negatív megerősítéses tanuláshoz pontokat.

A véletlen erdők ilyen döntési fákból állnak. Sok, egymástól relatív független fát tartalmaz. Egy ilyen rendszer kulcsa, hogy sok különálló, egymással kis korrelációban álló fát tartalmazzon, hiszen így nagyobb általánosítóerővel rendelkezik a modell. Ez azért van, hiszen több, kisebb leíróerővel bíró modellt együtt kiértékelve csökkenthető a torzítás[9].

2.1.3 Gépi tanulás algoritmus pontosságának mérése

A gépi tanuló algoritmusok pontosságának mérésére nem elég a helyes becslések/összes becslés arány bevezetése. Egyrészt nagyon kevés információt biztosít arról, hogy az adott modell milyen arányban vét hibákat, valamint milyen hibákat vét. (Fals pozitív, illetve fals negatív osztályozás megkülönböztetése fontos lehet. Ha egy daganatos elváltozásról szeretnénk megmondani, hogy jóindulatú-e, vagy sem, akkor nem árt, ha tisztában vagyunk vele, hogy az algoritmus milyen irányba hibázik.)

	Prediktált negatív	Prediktált pozitív
negatív	Valós negatív (VN)	Fals pozitív (FP)
pozitív	Fals negatív (FN)	Valós pozitív (VP)

3. ábra: Igazságmátrix

Másrészről belátható, hogy ezzel a méréssel 0.5-nél rosszabb modellt nem tudunk alkotni, hiszen, ha a címkék kevesebb, mint fele egyezik meg az elvárttal, akkor egyszerűen negáljuk a kimenetet. Ezért vezetjük be a valós pozitív, negatív, illetve fals

pozitív, fals negatív fogalmát (lásd 3. ábra). Ezek segítségével valósabb képet kaphatunk az algoritmus pontosságáról. A dolgozatban az egyes modellek kiértékeléséhez használt mérőszámok a következők:

- Accuracy (pontosság): Az összes valós negatív, illetve pozitív eredmény és az összes eredmény hányadosa. (VN + VP)/(VN+VP+FN+FP)
- Precision (precízió): A valós pozitív esetek száma, osztva az összes pozitív eset számával. Megadja, hogy az anomáliának értékeltek közül mennyi volt valóban anomália. (VP)/(FP+VP)
- Recall(érzékenység): A valós negatív esetek és a valós negatív, valamint fals pozitív esetek hányadosa. Megadja, hogy az összes anomália közül mennyiről mondtuk meg, hogy valóban anomália. (VP)/(FN+VP)
- F1-score: A precision és recall súlyozott átlagát adja meg. 2*(Precision*Recall)/(Precision+Recall)

2.2 Anomália detekció a gyakorlatban

Az anomáliák, kártékony tevékenységek detektálása szinte minden szektorban elengedhetetlen, hogy megelőzzünk vele károkat. Az online webshopok nagy része szűri a hamis vásárlásokat. Statisztikai alapon képet alkot arról, hogy egy vásárló milyen összegben és milyen tételszámban vásárol általában és ha valami nagyon eltérő ettől, letiltja a vásárlást. Sok esetben mérik azt is, hogy egy adott időegység alatt (jellemzően óránként) maximum hány vásárlás történhet. Így, ha valaki egy bot segítségével hamis vásárlások tömkelegét generálná, nem sikerülne neki[10]. A legtöbb online-bolt keretrendszer (pl. Shopify) tartalmazza ezeket a szolgáltatásokat.

A regisztrációk során a botok kiszűrésére a Captcha módszert alkalmazzák, mely egy olyan tesztet generál, amit csak ember képes megoldani. Ilyen lehet például bizonyos tárgyak bejelölése egy képen, vagy egy torzított szöveg leolvasása. Léteznek Captcha-k, melyek nem igénylik plusz művelet végzését a felhasználótól, hiszen az oldalon történt egérmozgatásból következtetik ki, hogy nem automata rendszer vezérli a kurzort. Ez üzleti szempontból fontos lehet, hiszen növelheti a felhasználói elégedettséget[8].

Léteznek statiszikai alapú anomália detekciós módszerek is. Erre egy jó példa Benford törvénye, mely egy statisztikai megfigyelésen alapul. Kimondja, hogy a természetben előforduló nagy numerikus adathalmazokban az egyes rekordok első számjegyei nem egyenletes, hanem logaritmikus eloszlásúak, így a világban előforduló számok első számjegyeinek kb. 50%-át az 1-2-3 számjegyek teszik ki[11]. A 80-as években több bank is alkalmazta, valamint a könyvelések érvényességét is vizsgálták vele. Egyik legismertebb fraud detektálás, melyet ezzel a törvénnyel vittek véghez, a 2009-es iráni választásokhoz köthető[16][19]. Sokan Benford törvényét tartják a természet szabályszerűségének. Érdekességképp az adatok folytonos változóira én is implementáltam a törvényt. Az első számjegyek eloszlása nagyban követte a törvényt, ugyanakkor voltak számjegyek, melyek nem illeszkedtek a Benford törvény által elvárt görbére. Ennek oka lehet, hogy az adathalmaz nagyon kicsi. Ez az eltérés számomra, az egyes entitások kártékonyságáról nem hordoz információt.

Egy nagyon elterjedt fraud detektáló módszer az AVS (Adress Verificaton Service), azaz a cím verifikáló szolgáltatás használata[13]. Ezek a szolgáltatások kapcsolatban állnak a bankokkal és egy regisztrációs folyamat során visszajelzést kaphatunk tőlük, hogy a regisztráció során megadott számlázási cím megegyezik-e a bankkártya tulajdonosához rögzített címmel. Ez egy erős indikátora lehet annak, hogy kiderüljön, hogy a regisztrációt nem a kártya tulajdonosa kezdeményezi, ugyanakkor fontos megemlíteni, hogy egyezés során sem mondhatjuk egyértelműen, hogy a vásárlás nem fraud. A szolgáltatástól egy kódot kapunk vissza, mely jelöli, hogy egyezik-e az irányítószám, a cím, vagy esetleg rendelkezésre áll-e a felhasználóról ez az adat.

A CVV (Card Verification Value) egy, a kártyára (jellemzően bankkártyára) nyomtatott számot kér be, majd ellenőrzi, hogy a megadott szám megegyezik-e a fizikai kártyán találhatóval. Ehhez szükséges, hogy a kártya fizikailag nálunk legyen[7].

Léteznek szolgáltatások, melyek kifejezetten arra szakosodtak, hogy harmadik félként ajánlást adjanak cégeknek, hogy az általuk ellenőrizni kívánt adat valós-e, vagy sem. Ilyenek pl a Emailage, vagy a Clearsale.

Az anomália detekció témaköre nagyon komplex, erősen domainfüggő, így nincs egy általánosan elfogadott, minden helyzetben jól működő módszer. A legtöbb cég nem hozza nyilvánosságra az általa kifejlesztett módszert, hiszen ezzel a kártékony tevékenységek viselkedése adaptálódhatna a szűrőjükhöz, így könnyen átmehetne rajta.

Az anomália detekció nagyon fontos egy szolgáltatás hírnevére vonatkozóan is, hiszen azt nagyban meghatározza az, hogy mennyire biztonságos. Minden szolgáltató érdeke, hogy a lehető legkevesebb káros tevékenység kötődjön a nevéhez.

Sok cég manuálisan, elemzők segítségével végzi ezt az analízist. Ez nem egy hatékony mód, hiszen egy nemdeterminisztikus modell, sok múlik az elemző pillanatnyi döntésén, melyet befolyásol érzelmi állapota, fáradtsági szintje, valamint tapasztalata. Egy jól működő modell üzletileg nagy előnyt jelenthet, hiszen adott idő alatt, több adatot tud feldolgozni, valamint determinisztikus működést biztosít, mely konzisztens osztályozáshoz vezet. Továbbá az egyes döntésekbe nem számítanak bele olyan, egyénenként eltérő emberi tényezők, mint a fáradtság, valamint kivédhető a tapasztalat hiánya.

3 Tárgyalás

3.1 Alkalmazott technológiák

Az adatokat az IBM Cloudant NoSQL adatbázisából értem el. Ezeket a hozzá tartozó API segítségével ideiglenes fájlokba töltöttem le. Az általam használt programozási nyelv a Python 3.7. Olvashatósága, valamint a számtalan analízist, data science tevékenységet segítő modulja miatt választottam a feladatra. Ezen belül az adatok tárolására, manipulálására a Pandas könyvtárat használtam. Ez egy gyors, nyílt forráskódú, python nyelvre épülő könyvtár, mely táblázatos adatok gyors olvasását, írását, módosítását teszi lehetővé. A legtöbb gépi tanulási modellt tartalmazó könyvtár jól kezeli a formátumát, bonyolultabb konverzió nélkül adhatók át számukra a Pandas-ban tárolt adatok. Dimenzióredukáláshoz az UMAP könyvtárat vettem igénybe, részletesen egy későbbi alfejezetben fejtem ki működését. A gépi tanuláson alapuló modelleket Scikit learn segítségével illesztettem. Az adatok vizualizációjára a Matplotlib könyvtárat használtam. Az AutoEncoder-t a Pyod könyvtárból illesztettem.

3.2 Definíciók, alapvető szabályok

Ahhoz, hogy a kártékony tevékenységeket sikeresen detektálni tudjuk, fontos, hogy először jól definiáljuk, hogy mi számít fraudnak. Ebben a dolgozatban *minden olyan tevékenységet, mely a cégnek kárt okoz, akár anyagi, akár hírnév vonatkozásban, vagy törvénnyel ellenes, kártékonynak tekintünk.* Azon felhasználókat definiáljuk anomáliaként, akik kártékony tevékenységet terveznek folytatni a szolgáltatás igénybevételének segítségével.

Üzletpolitikai kérdés, hogy mennyire fektessünk nagy hangsúlyt a regisztrációs folyamat során ezen ügyfelek kiszűrésére. Amennyiben alapos átvizsgálásnak vetjük alá az egyes eseteket, mindenképp pontosabb detekcióra leszünk képesek, de ez a folyamat erőforrásigényes, ezért költségesebb és lassú, ami pedig a felhasználói elégedettséget csökkentheti. Egy másik opció, ha nem fektetünk nagy hangsúlyt a regisztráció során történő vizsgálatra. Ezzel az elemzés olcsóbbá, gyorsabbá válik, viszont elkerülhetetlenül több anomália fog megjelenni a felhasználók világában, amely hosszú távon a cég szempontjából negatív hatásokkal járhat. Az IBM Cloud kifejezetten a biztonságra fekteti

a hangsúlyt. Ebből kifolyólag a kezdő hozzáállás az, hogy minden eset fraud, addig, míg ez meg nem dől. Amennyiben egy esetről nem dönthető el egyértelműen, hogy nem kártékony, részletesebb vizsgálatnak vetik alá.

Manuális elemzők jelenleg több felületen dolgoznak, ahol látják az egyes esetekhez tartozó tulajdonságokat, valamint kapcsolati gráfot látnak az új regisztrációkról. Ennek segítségével detektálják a kialakuló trendeket. Trendnek nevezzük azon időszakosan megjelelő, kártékony regisztrálók sorozatát, melyek között, bár gyakran nehezen, de egyértelműen felfedezhető, közös attribútumok vannak. (Ilyen például egy adott szerverparkról, ugyanazon ip címről érkező, adott idő alatt gyakran bejövő regisztrációs kérelem.)

A dolgozat készítése során két adatbázis állt rendelkezésemre. Az egyik egy címkézett, kb. 8000 rekorddal rendelkező adatbázis volt, ahol a címke a célváltozó volt, azaz az elemzők által adott értékelés. A másik adatbázis egy jóval nagyobb, kb. 200 ezer rekorddal rendelkező volt, célváltozó nélkül.

3.3 Az adatok letöltése, előfeldolgozása

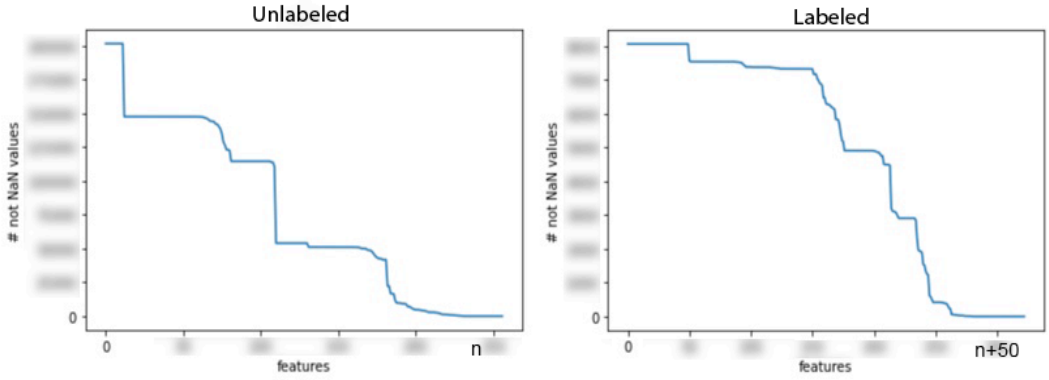
Az adatokat a Cloudant API segítségével töltöttem le. Ez egy NoSQL alapú adatbázis, amely az egyes rekordokat ún. documentekben tárolja. Ezek letöltése úgy zajlott, hogy egy függvény kiadott a megfelelő paraméterekkel ellátott HTTP kérést, melyre válaszként visszaérkeztek metaadaok az adatbázisról (hány elemet tartalmaz, stb), majd minden letöltendő elemre indult egy-egy HTTP kérés, melyre válaszként érkezett meg az ahhoz a rekordhoz tartozó JSON állomány. A Cloudant szerver limitálja az adott idő alatt beérkező HTTP kérések számát, így egy kis szünetet kellett implementálni a kérések elküldése közé.

A JSON állományokat ezután a Python JSON könyvtárának beépített JSON Flatten (laposítás) funkciójával Pandas DataFrame-mé konvertáltam. A rekordok tulajdonságai között volt diszkrét, folytonos, illetve bináris, valamint időbeli adat is. Mindkét adathalmaz hozzávetőlegesen 200-250 tulajdonsággal rendelkezett.

Mivel több, azonos nevű tulajdonság is megjelent így a DataFrame objektumban, szükségessé vált a JSON fileok részletes tanulmányozása. Erre vim-et használtam, hiszen a grafikus megjelenítők, valamint modern szövegszerkesztők (VSCode) nem tudták megfelelően megjeleníteni az adatot a mérete miatt.

Következő lépésként a tulajdonságok nevének konzisztenssé tételével foglalkoztam. A két adatbázisban más-más JSON hierarchia volt kialakítva, más szinteken voltak megtalálhatóak az adatok, így a JSON – Pandas DataFrame konvertálás során más neveket kaptak az egyes tulajdonságok. A tulajdonságok egységes elnevezésére egy függvényt írtam, amely a tulajdonságok nevéből a különböző hierarchia miatti felesleges részeket előre beállított offszetek segítségével levágta. Így a két adatbázis azonos nevű tulajdonságokkal rendelkezett már, ami fontos mérföldkő volt a későbbi elemzésekhez.

Következő lépés a megfelelő tulajdonságbázis kiválasztása volt. A tulajdonságok összegyűjtése több forrásból történik, ráadásul a különböző országok máshogy szabályozzák azon adatokat, melyeket elérhetővé tesznek 1-1 felhasználóról, így több olyan tulajdonság is található az adatbázisokban, melyekben a hiányzó adatok aránya igen magas. Az egyes változók kitöltöttségi arányát mutatja a 4. ábra. Ezekkel a későbbiekben nehéz lesz dolgozni, hiszen az adatok pótlása nagyon körülményes művelet lenne, ezért megvizsgáltam minden tulajdonságra a kitöltöttség arányát.



4. ábra: A címke nélküli és a címkézett adatbázis tulajdonságai kitöltöttségének aránya

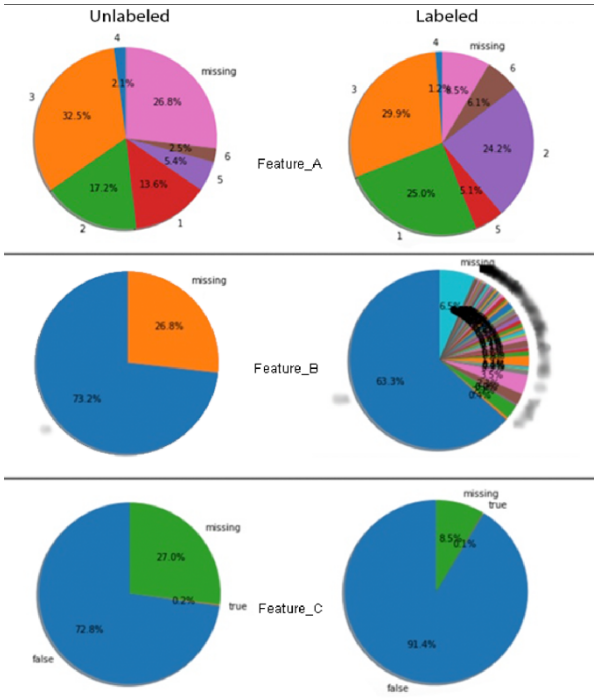
A célváltozó nélküli adatbázis esetében az egyes tulajdonságok kitöltöttségi arányát kirajzolva jól látható volt 50 és 100 darab tulajdonság között egy letörés. Az ott megkapott arányt határoztam meg küszöbértéknek, majd az az alatti kitöltöttséggel rendelkező tulajdonságokat eldobtam mindkét adathalmazból. A Cloudant adatbázist előttem más célra használók több MapReduce funckiót is eltároltak, mint entitást, így ezektől a felesleges, számomra jelentéstartalom nélküli adatoktól is megszabadultam.

3.3.1 Konzisztencia vizsgálata

Ahhoz, hogy a címkézett adatokon tanított modellek alkalmazhatóak legyenek a jóval bővebb, címkézetlen adathalmazra is elengedhetetlen megvizsgálni, hogy a két adatbázisban található tulajdonságok statisztikailag megegyeznek-e.

3.3.1.1 Diszkrét változók vizsgálata

A diszkrét változók mindegyikéhez egy kördiagrammot rajzoltam ki, mely reprezentálja a tulajdonság egyedi értékeinek előfordulási gyakoriságát. Három tulajdonsághoz kirajzolt diagrammot mutat be az 5. ábra.



5. ábra: Három diszkrét változó értékének eloszlása mindkét adatbázisban

Minden vizsgált tulajdonság esetén igaz az, hogy a címkézett adathalmazban arányaiban kevesebb hiányzó adat van. Ennek oka valószínűleg az, hogy ez az adatbázis kitöltöttség szempontjából felülreprezentált. Az ebbe kerülő esetek nem egyenletes eloszlással kerültek kiválasztásra, hiszen azok a felhasználók, akik fizetős szolgáltatást terveznek igénybe venni, nagyobb valószínűséggel adják meg adataikat. Az *"A"* tulajdonság esetében az értékek eloszlása nagyjából megegyezik, kis torzulást csak az adatok hiánya okoz a célváltozó nélküli adathalmazban. A *"B"* tulajdonság egy olyan adat, melyet az IBM nem bocsátott rendelkezésére a címkézetlen adathalmazzal foglalkozóknak, így számomra is elérhetetlen. Ezeket a

tulajdonságokat is eldobtam (0-10 db közötti ilyen tulajdonság volt az adatbázisban), hiszen a címkézetlen adathalmazban nagyon kevés információtartalommal bír. A *"C"* tulajdonság egy logikai változó, mely értékeinek előfordulásában ugyan látható kis eltérés, de nem számottevő, okozhatja a címkézett adathalmaz jóval kisebb elmeszáma is.

3.3.1.2 Folytonos változók vizsgálata

A folytonos változók konzisztenciáját statisztikai próbákkal vizsgáltam. Egy statisztikai próba olyan műveletsorozat, mely meghatározza, hogy a vizsgált változók értéke között van-e statisztikai kapcsolat/összefüggés, képet ad eltérésük mértékéről. A legtöbb ilyen módszer azt vizsgálja, hogy két adathalmaz között megdönthető-e egy előre meghatározott nullhipotézis.

Ez egy olyan statisztikai alapfelvetés, ami azt mondja ki, hogy két vizsgált változó között nem áll fenn összefüggés. Ezt mindaddig tényállásként kezeljük, amíg be nem tudjuk bizonyítani az ellenkezőjét. Ez a leggyakrabban úgy történik meg, hogy megfogalmazunk egy alternatív hipotézist, (amely feltételezi, hogy fennáll a két változó között valami féle statisztikai összefüggés,) majd ezt bebizonyítjuk.

Két statisztikai tesztet használtam a folytonos változók vizsgálatára. Az egyik a *két mintás t-próba*, mely azt vizsgálja, hogy két változó átlaga statisztikai szempontból mutat-e eltérést. A másik a *Kolmogorov-Szmirnov próba*, mely egy mintás t-próbát alkalmaz mindkét változón, azok statisztikai és elméleti eloszlásfüggvényük eltéréseinek maximuma alapján. Az így kapott értékből összehasonlíthatjuk a két vizsgált változó eloszlását.

```
Feature_E
Ttest_indResult(statistic=6.393964003046161, pvalue=1.6197744646316456e-10)
Ks_2sampResult(statistic=0.1360040083766928, pvalue=4.613827571080398e-125)

Feature_F
Ttest_indResult(statistic=28.993811455398895, pvalue=1.8279341448377948e-184)
Ks_2sampResult(statistic=0.2281138442489652, pvalue=0.0)
```

6. ábra: A folytonos változókon alkalmazott statisztikai tesztek eredménye

A korrelációs teszteredményekről leolvasható, hogy mindegyik változóra igazolták a statisztikai tesztek, hogy a közöttük található átlag, valamint szórásbeli különbségek mértéke nagyon kicsi (p érték < 0.05). A tesztek alapján nem bizonyítható, hogy lényeges

eltérés lenne a két adathalmaz ugyan azon tulajdonságainak eloszlásai között (lásd 6. ábra).

3.3.1.3 Kategorikus változók kezelése

Az általam használt gépi tanulás alapú modellek csak numerikus adatokkal képesek műveleteket végezni, így elengedhetetlen lépés a kategorikus változók megfelelő előfeldolgozása. Ennek elvégzésére többféle módszer is ismert, mint például a label encoding, mely lényege, hogy az adott tulajdonság minden értékéhez egy számot rendel, majd ezzel helyettesíti azt. Ezen módszer hátránya, hogy így a különböző tulajdonságokat egy számegyenesen elhelyezett ponttal reprezentáljuk, ami azt indukálja, hogy a tulajdonságoknak megjelenik egy egymástól mérhető távolsága. Így azt tanítjuk modellünknek, hogy egy

```
FEATURE_CARS: [’Audi’: 1, ’Mercedes’: 2, ’BMW’: 3 ...]
```

megfeleltetés esetén az Audi közelebb van a Mercedeshez, mint a BMW-hez, mely nem helyes és csökkentheti a modellünk leíróerejét.

Erre a problémára nyújt megoldást a One Hot Encoding, mely egy n elemmel rendelkező tulajdonsághalmazt egy n dimenziójú vektorral reprezentál és minden egyes elemhez egy különböző egységvektorát rendeli hozzá. Így minden elemnek egyediséget biztosít, valamint a kategóriák egymástól egyenlő távolságra vannak. Hátránya a módszernek, hogy egy 30 egyedi értékkel rendelkező tulajdonság 29 új tulajdonságot fog eredményezni az adathalmazban, ezzel nagyban növelve annak dimenziószámát. Az általam feldolgozott adatbázisok esetében azonban ezzel a technikával is kezelhető méretű maradt az adathalmaz.

Több tulajdonság is volt, melyeket első ránézésre kategorikus adatnak ítéltem meg, ilyenek például különböző külső szolgáltatóktól érkező, 1-5-ig terjedő pontozások. Jobban végig gondolva ezeket, folytonos változókként kezeltem, hiszen itt értelmezhető, sőt tartalommal bír az egyes pontok közti távolság.

3.4 Tulajdonságok definiálása, korrelációs vizsgálata

Az új tulajdonságok készítésének a célja, hogy a meglévő információkból olyan új, a célváltozóval erősen összefüggő tulajdonságokat alkossunk, melyek még nincsenek benne a modellben. A tulajdonságok korrelációs vizsgálata segíthet meghatározni, hogy

a későbbi modellillesztés során mely tulajdonságokat érdemes megtartani az adathalmazban és melyek a célváltozóval kapcsolatban információt nem hordozó tulajdonságok.

3.4.1 Location similarity tulajdonság

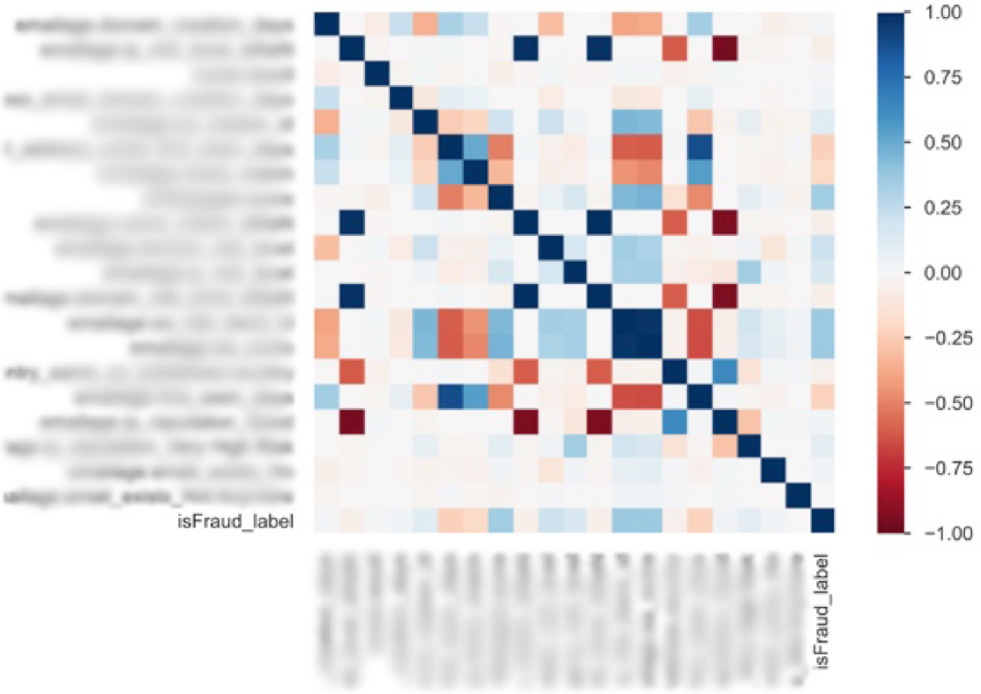
Különböző forrásokból rendelkezésemre állt az ip cím lokációja, hogy hová történik a számlázás, valamint az, hogy a felhasználó használ-e VPN-t, vagy egyéb, ip-t elfedő szervert.

Amennyiben ilyen használatára nem kerül sor, de az IP cím, valamint a számlázási cím nem egyezik meg, egy Location Unsimilar bitet 1-re állítok, mely jelzi, hogy a két hely nem egyezik. Ellenkező esetben az értéke 0. Ha hiányzó adat van, az értéke elővigyázatossági szempontokból szintén 1-re áll.

3.4.2 Korreláció analízis

Miután a tulajdonságokat a DataFrame objektumon belül megfelelő típusra állítottam és a one hot kódolás segítségével a kategorikus változók is átalakításra kerültek, megkezdtem a korreláció analízist. Két változó korrelációja megadja, hogy köztük milyen mértékű a függőségi kapcsolat. Azt vizsgáltam, hogy az egyes tulajdonságok milyen mértékben korrelálnak a célváltozóval, mennyire határozzák meg annak értékét. A célváltozóval nagy mértékben korrelációt mutató tulajdonságok meghatározzák annak értékét, így ezen tulajdonságokat érdemes használni a modellben.

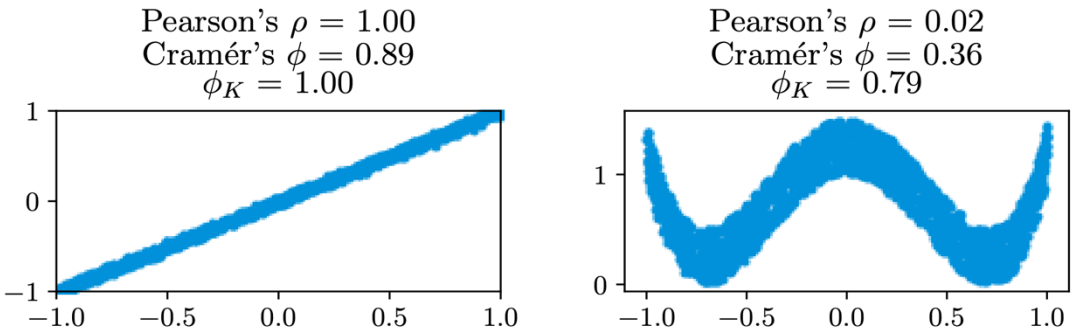
Két változó között a korreláció mértékét a Pearson-félre r korrelációs együtthatóval mértem. Ez egy 1 és -1 között mozgó szám. Minél közelebb van r abszolútértéke 1-hez, annál erősebb kapcsolatot feltételez a két változó között, valamint minél közelebb van 0-hoz, annál lazább közöttük a kapcsolat. 0 érték esetében a két vizsgált változó között nincs lineáris összefüggés. Amennyiben r előjele pozitív, a két vizsgált tulajdonság értékei között egyenes arányú, amennyiben negatív, fordított arányú kapcsolat áll fent. (Számomra ezeket nem kellett külön kezelni, hiszen a később implementált modellek ezt kezelik, nem kell a tanításnál jelölni ezt.)



7. ábra: Pearson korrelációs mátrix

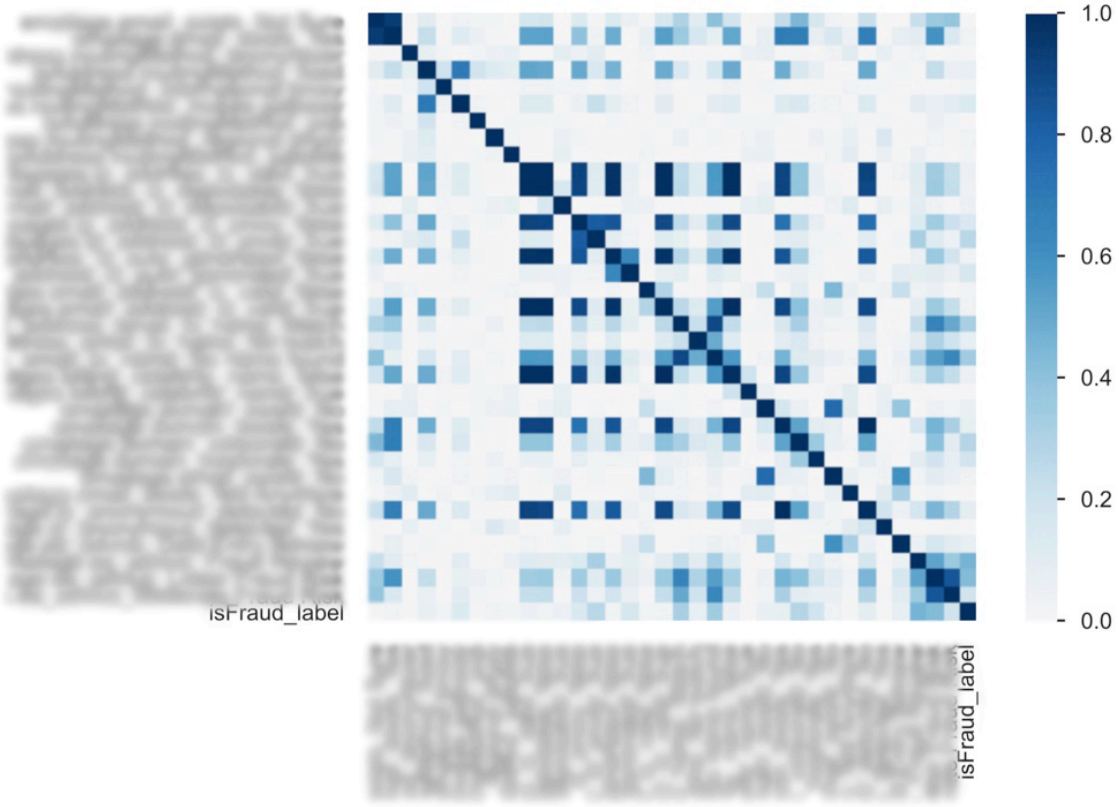
A 7. ábra a Pearson-féle korrelációs mátrix látható a tulajdonságok egy részhalmazára, illetve a célváltozóra tekintve. A mátrix ÉNY-DK-i főátlója csupa egyesből áll, hiszen ott minden változó saját magával számított összefüggésének értéke áll.

A Pearson-féle r korrelációs együttható hátránya, hogy csak lineáris összefüggéseket vizsgál, ezért a φK korrelációs együtthatót is megvizsgáltam. Jellegzetessége, hogy képes nemlineáris összefüggéseket is detektálni változók értékei között, így indikálhat bonyolultabb összefüggéseket is[15]. Egy ilyen esetet mutat be a 8. ábra 2. grafikonja is, ahogy az adatpontok között a Pearson korrelációs vizsgálat egy



8. ábra: A korrelációs együtthatók összehasonlítása lineáris, illetve nem lineáris összefüggés esetében [15]

nagyon alacsony együtthatóval tér vissza, hiszen az adatok között nem-lineáris összefüggés található. Skálája 0 és 1 között mozog. A korrelációs együttható értéke minél nagyobb, annál erősebb kapcsolatot jelez a két változó között. Az adat egy részhalmazára implementált φK vizsgálatát lásd 9. ábra.



9. ábra: φK korrelációs együtthataó mátrix

A mátrixok ugyan azon tulajdonságokat mutaják a címkével magasan korrelálónak. Ebből az következik, hogy a tulajdonságok és az osztályzás között főképp lineáris összefüggések vannak. Leolvasható, hogy több, one hot enkódolt, diszkrét változó érték, valamint folytonos változók is mutatnak magasabb korrelációt. Kiválasztottam 20, a célváltozóval legmagasabb korrelációban álló tulajdonságot. Továbbiakban ezeket használom a dimenzióredukció, valamint a modellek tanítása során. Az ezek alatt lévő tulajdonságok már csak nagyon kevés korrelációt mutatnak a címkével.

3.5 Dimenzióredukálás

A kiválasztott tulajdonságok alapján már be lehetne tanítani az algoritmust, ami a tanító adathalmazommal lehetséges is, hiszen elemszáma 10.000 alatti. A mai számítógépek kapacitásával ezek a műveletek belátható időn belül elvégezhetők lennének, de cél, hogy egy gyorsan működő modellt kapjak, valamint, ha később bővül a

tanító adathalmaz elemszáma, az lassíthatja a modell tanítását. Ez okból is fontos az adat dimenzió redukálása.

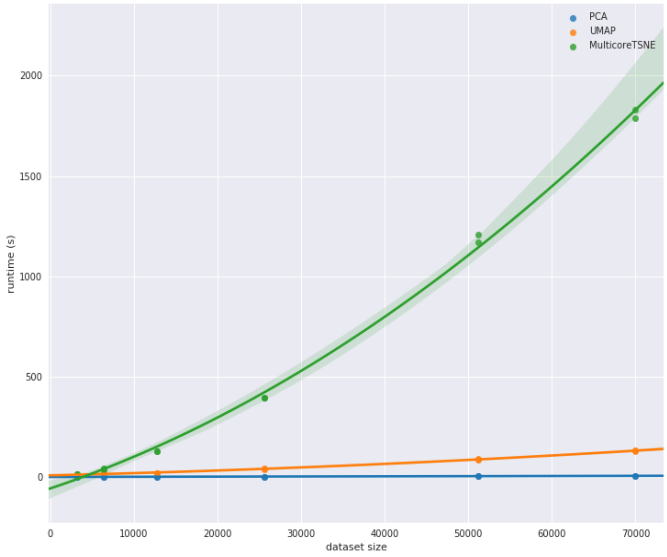
Ez egy olyan matematikai eljárás, mely egy magasabb dimenziószámú adatot valamilyen műveletekkel (pl. különböző projekciók) alacsonyabb dimenziószámúvá redukál, a lehető legminimálisabb entrópiaveszteséggel.

A dimenzió redukálásnak több előnye is van. Egyrészt, ahogy a fentebbi bekezdésben taglaltam, az adat mérete kisebb lesz, így gyorsabban végezhetők rajta műveletek. Mivel az adat kevesebb dimenzióban jelenik meg, lehetőségünk van vizuálisan reprezentálni, így egy 3-nál magasabb dimenziószámú adathalmaz is megjeleníthető lesz. A modellre matematikai szempontból is hatást gyakorol. Magasabb dimenziószámban a sűrűség elveszik, ezáltal a sűrűség alapú távolságmérés nem használható, mely anomália detekció során egy gyakran alkalmazott módszer. Amennyiben túl sok tulajdonság áll rendelkezésünkre, fennáll a túlillesztés veszélye. Ez azt jelenti, hogy egy adott modell a tanító adathalmazra ''túl jól'' rátanul, ám új esetekre pontatlan kimenetet fog adni. Ezt a túlságosan magas számban rendelkezésre álló tulajdonságok is okozhatják, hiszen a modellnek nem kell sok matematikai összefüggést megtanulnia, elég a tulajdonságok egy nagyon egyszerű kombinációját figyelni. A dimenzió redukálás erre is megoldást nyújt, hiszen alacsonyabb dimenzióban valószínűleg bonyolultabb összefüggéseket vehet észre az algoritmus, amik alapján eredményt számol.

Gyakran használt dimenzióredukáló módszerek a PCA, valamint a t-SNE. A PCA-t (főkomponens analízist) 1901-ben alkotta meg Karl Pearson, majd később, tőle függetlenül is megjelent egy 1933-ban íródott tanulmányban. Működési elve, hogy az adatot a lineáris algebra módszereivel vetíti kisebb dimenziószámú altérre[12]. Célja az eredeti tulajdonsághalmaz olyan lineáris transzformációja, mely az előzőtől kevesebb, olyan új tulajdonságokat hoz létre, amely legjobban tükrözik az eredeti tulajdonsághalmaz varianciáját. Hátránya, hogy csak lineáris összefüggéseket képes detektálni, így nagy mértékű információvesztéshez vezethet.

A t-SNE egy 2008-ban feltalált eljárás, a mai napig az egyik legnépszerűbb dimenzióredukáló módszer. Legfőbb eltérése a PCA-tól, hogy nemlineáris összefüggéseket is képes észrevenni, azonban a t-SNE nagy méretű adathalmaz esetében rosszul skálázódik[1].

Az adataim dimenzióredukáláshoz az UMAP (Uniform Manifold Approxiamtion and Projection for Dimension Reduction) technikát használom. Az UMAP egy 2018-ban bemutatott dimenzióredukáló technika. A t-SNE-vel szemben számos előnnyel rendelkezik. Az egyik legnagyobb a nagy elemszámú halmazokon történő dimenzióredukálás lefutási sebességének radikális csökkentése.



10. ábra: Dimenzióredukáló módszerek lefutása az adat méretének függvényében[4]

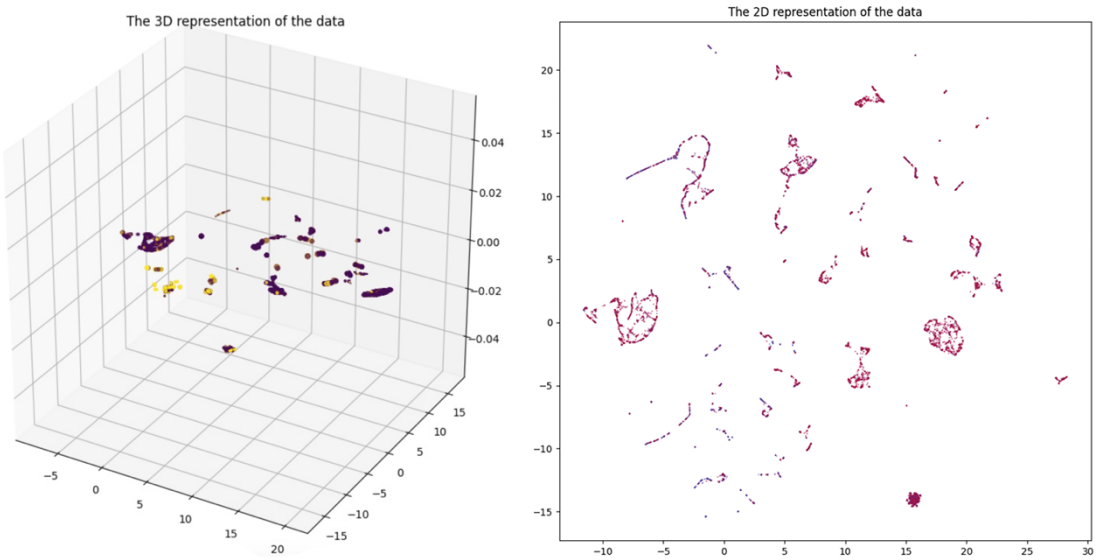
A 784x70.000 pontot tartalmazó MNIST adathalmazon az UMAP 3 perc alatt végzi el a dimenzió redukálást (3 dimenziósra), szemben ezzel az scikit-learn t-SNE algoritmusának ez 45 percet vesz igénybe[6]. A 10. ábra szemlélteti, hogy az UMAP mennyivel jobban skálázódik nagy adatokra, mint a MulticoreTSNE, ami sokkal gyorsabban fut le, mint az scikit-learn-ben található t-SNE algoritmus[3].

Előnye még az UMAP-nak, hogy a globális struktúrák mellett a lokális struktúrákat is jobban megtartja, mint a t-SNE. Ez azt jelenti, hogy a klasztereken belüli kapcsolatok is erősebbek, nagyobb jelentéssel bírnak, mint a t-SNE esetén[2].

Az UMAP teljes mértékben beleillik az sklearn pipeline-ba, számára blackbox modulként viselkedik. Az UMAP használata:

```
standard_embedding =
umap.UMAP(random_state=42,n_components=3).fit_transform(X)

#ezután az UMAP egy 3 elemű listákat tartalmazó listával tér vissza, az
egyes pontok x,y,z koordinátáival
```



11. ábra: Az adat vizuális megjelenítése dimenzió redukálás után 3, illetve 2 dimenzióban

A 3 dimenziós ábrán a lila a nem kártékonynak, a sárga pedig a kártékonynak ítélt felhasználókat jelöli. A 2 dimenziós redukálásban ezek a piros, illetve kék színek. (lásd 11. ábra)

2 dimenzióban felfedezhetők ugyan nagyobb csoportosulások, láncolatok, de közöttük az entitás kártékony mivoltának szempontjából nincs észrevehető elkülönülés. 3 dimenziós redukálás során azonban jól látható, hogy 2 klaszter szinte csak kártékonynak címkézett felhasználókból áll, míg a többi túlnyomó részt nem.

3.6 Modell illesztése

Miután az adat már átesett dimenzióredukáláson is, készen állt, hogy gépi tanuláson alapuló modelleket illesszek rá. A tanító adathalmazomat tanító, illetve tesztelő halmazra szedtem szét, 75%-25%-os eloszlásban. Erre azért van szükség, hogy a tesztelő adathalmaz segítségével meg tudjam határozni a készített algoritmus pontosságát.

Az egyik modell, melyet az adatra illesztettem a logisztikus regresszió volt. A másik a random forest classifier. Korábbi félévekben elvégzett munka során már készítettem egy logisztikus regressziót a címkézett adathalmazra, mely során nem fektettem ekkora hangsúlyt a tulajdonságok megfelelő előfeldolgozására, így egy jóval pontatlanabb modellt kaptam. Ez is igazolja azt, hogy milyen fontos az adatok megfelelő előfeldolgozása.

3.6.1 Modellek értékelése

	Accuracy	Precision	Recall	F1-Score
Logisztikus regresszió	0.800	0.741	0.664	0.684
Random Forest	0.806	0.750	0.672	0.693
Régi logisztikus regresszió	0.751	0.651	0.524	0.486

12. ábra: Az egyes modellek értékelése

Az egyes modellek értékeléséből látható, hogy az adatok megfelelő előfeldolgozása egy nagyon fontos lépés, hiszen majdnem 0.2 javulást sikerült a két regressziós modell F1-Scorejában elérni (lásd 12. ábra).

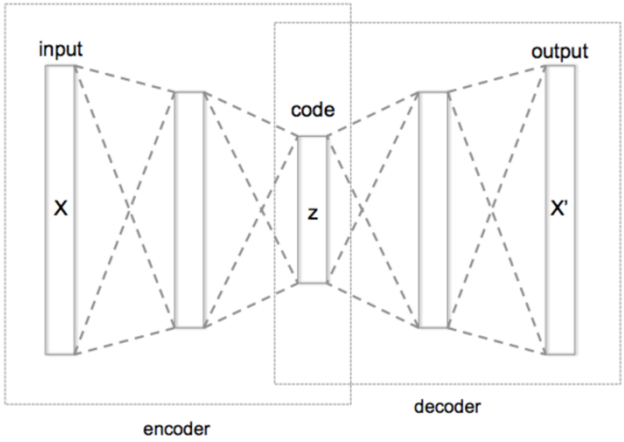
Valószínűsíthető, hogy a Random Forest klasszifikáló eredményei kis mértékben tovább javíthatók lennének hiperparaméter-optimalizálással, valamint keresztvalidációs adatrész használatával, de dolgozatom célja egy átfogó anomália detekciós folyamat bemutatása, a hangsúlyt nem a gépi tanuló modell optimalizálására helyezi.

3.7 Elemzői döntések vizsgálata

Ahogyan azt a dolgozat elején említettem, a címkék nem beigazolódott esetek alapján készültek, hanem az elemzők által adott értékeléseket reprezentálják. Előfordulhat azonban, hogy egy elemző kártékonynak ítél egy, valójában kártékony tevékenységet folytatni nem kívánó személyt, valamint ennek a fordítottja is. Ebből adódóan a címkézett adathalmaz alapján maximum csak olyan jó algoritmust vagyunk képesek alkotni, amennyire jól osztályoznak az elemzők. Erre a problémára nyújtana megoldást, ha valahogy ellenőrizni tudnánk az elemzők döntését.

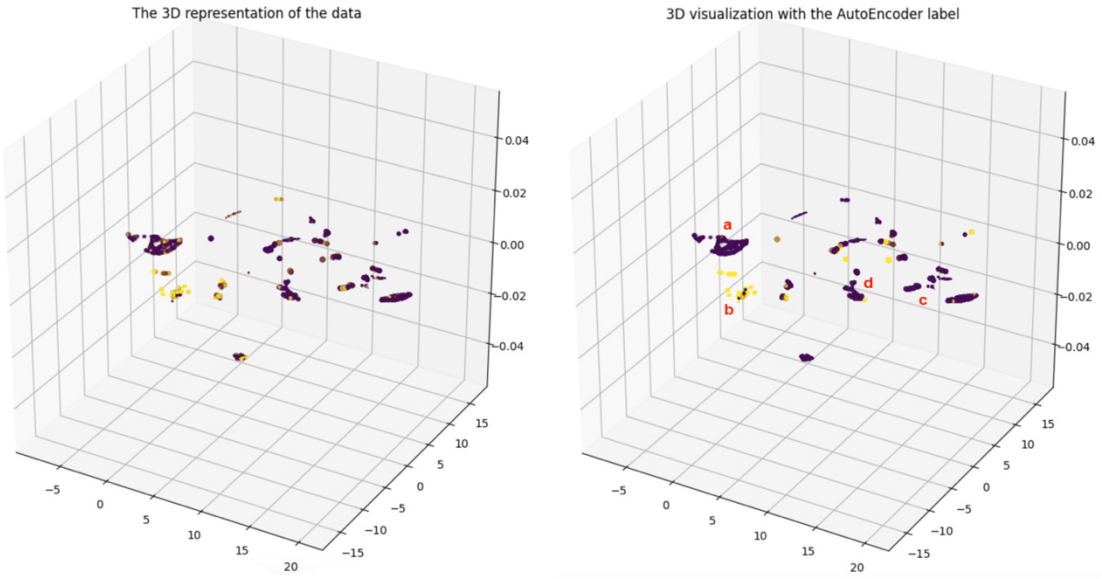
3.7.1 AutoEncoder

Erre a problémára nyújt egy lehetséges megoldást az AutoEncoder segítségével történő anomália detekció. Az AutoEncoder egy neurális háló architektúra, mely bemeneti, kimeneti, valamint rejtett réteg(ek)ből áll. Logikai szinten három része különül el. enkóder, kód, illetve dekóder[5]. Felépítését lásd: 13. ábra.



13. ábra: Az AutoEncoder felépítése (ábra forrás: Wikipedia)

Ez egy címkézetlen módszer, bemenetként csak egy tulajdonsághalmazt vár. Ezeket megkapva az enkóder rátanul a tulajdonságokra és egy kevesebb dimenziójú kódot állít elő belőlük, a lehető legkevesebb információveszteséggel. Amint a kód elkészül, a dekóder megpróbálja visszaállítani belőle az eredeti mintát. Ha ez megtörtént, az eredeti mintát és a visszaállítottat összehasonlítja, ebből egy hibapontot számít. Ha egy adott elemet nem megfelelően, nagy hibával tudott csak visszaállítani, akkor az adathalmaz szempontjából ez anomáliának számít[17].



14. ábra: Az adat az elemzők által adott címkékkel, valamint az AutoEncoder által generáltakkal

Az AutoEncoder által osztályozott címkék nagy mértékben megegyeznek az elemzők által adottakkal, azonban vannak apróbb eltérések. A 14. ábra ”*a*”-val jelölt csoportosulásban az AutoEncoder egyáltalán nem jelölt anomáliát, ahogyan a ”*d*”, illetve a ”*c*” csoportosulásokban is jóval kevesebbet, mint az elemzők. A ”*b*”-vel jelölt csoportosulásban viszont az elemzők jelöltek kevesebb anomáliát, ezeket az eseteket érdemes lehet számukra újra megvizsgálni.

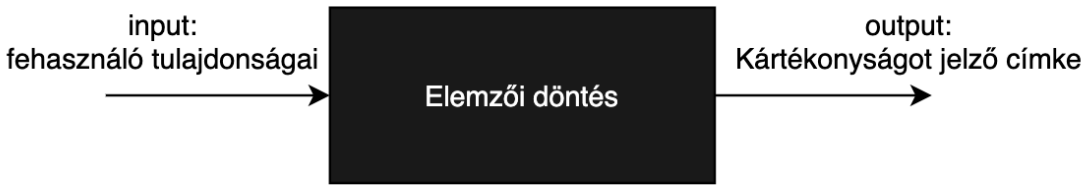
Az elemzők azokon a klasztereken belül is több anomáliát jelöltek meg, amelyek nagyrészt nem kártékony felhasználókat tartalmaztak. Ez viszont betudható a céges alapelvnek, miszerint minden eset, melyről nem egyértelműen eldönthető, hogy nem kártékony, azt kártékonyként kell kezelni. Az AutoEncoder objektum *contaminaiton* paraméterét a lehető legkisebb, 0.1 értékre állítottam.

3.7.1.1 Felhasználása

Az AutoEncoder egy determinisztikus működésű matematikai modell alapján keres az adatban anomáliákat, célváltozó nélkül. Az általa kapott eredmények visszacsatolhatók az elemzők által adott címkékre. Amennyiben egy adott esetet az elemzők nem jeleztek kártékonynak, de az algoritmus igen, érdemes megvizsgálni újra azt. Ha pedig egy adott esetet anomáliának jeleztek, de az algoritmus szerint nem lenne az, lehet, hogy az elemző nem jogosan zárt ki egy felhasználót. Az anomália detekció

egyik nagy kihívása, hogy a tanító halmazban alacsony arányban fordulnak elő anomáliák, így kevés olyan elem van, melyre jól rátanulhat az algoritmus. Ezzel a módszerrel javítható lenne a címkék minősége, így egy pontosabb tanító adathalmaz jöhetne létre. Ezzel a tanítható modellek pontossága is javíthatóvá válna.

3.7.2 Az eddigi osztályzási folyamat

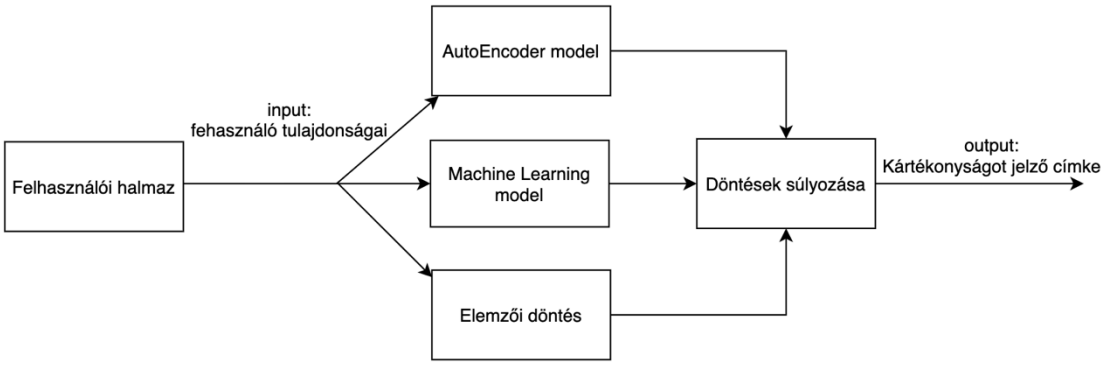


15. ábra: Manuális elemzőkkel történő döntés folyamata

Az eddigi munkafolyamat szerint egy eset beérkezett egy elemzőhöz. Ő megvizsgálta azokat a tulajdonságokat, melyeket fontosnak vélt és ha adott tulajdonságnál vagy tulajdonságoknál eltérést, gyanús viselkedés mintázatát észlelte, akkor az esetet kártékonynak vélte. Amennyiben ilyenre nem került sor és minden adat illeszkedett a normális viselkedés mintájára, az entitást nem ítélte kártékonynak és engedte számára a regisztrálást (lásd 15. ábra).

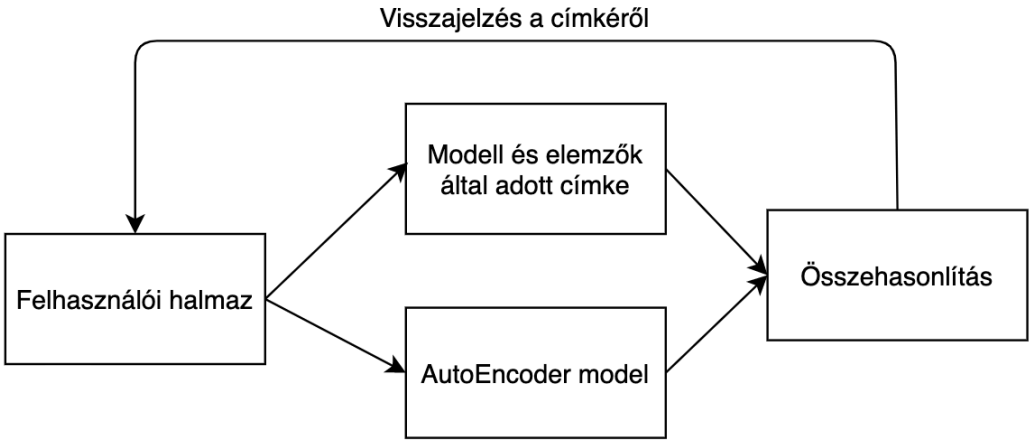
A modell szinte black-box szerűen működött. Bemenetként egy adott felhasználó tulajdonságait kapta meg, kimenetként pedig egy címkét ad, mely osztályozza a felhasználót anomália voltja szempontjából. Az osztályozás folyamata nem rögzített, nem megismételhető szekvencia, hiszen sok olyan változó (munkatárs tapasztalatai, szubjektív vélemény, fáradtság, stb.) közrejátszhatott, melyek befolyásolhatják az eredményt. Másik problémája ennek a folyamatnak, hogy nem tudjuk mérni a minőségét, nincs visszacsatolás. Modellem ezekre nyújt megoldást.

3.7.3 A modellt használó osztályozási folyamat



16. ábra: A dolgozat modelljét használó döntés folyamata

A döntés immár nem csak az elemzők által történik, hanem pontos matematikai modellek is segítik meghatározni, hogy az adott felhasználó kártékony-e. Ezzel kiküszöböljük az osztályozás esetlegességét. Így egy determinisztikus matematikai modell is segíti a döntést. Végül az implementált gépi tanuló algoritmus, neurális háló, valamint az elemző manuális döntésének a súlyozásából áll össze a pontosabb végeredmény (lásd 16. ábra).



17. ábra: A címke pontosságának javítása visszacsatolással

A tanító adathalmaz javítása is lehetővé válik modellem alkalmazásával. A modell az elemzők által adott címkét összeveti az AutoEncoder által adott címkével. Ezt a két változót összehasonlítva visszacsatolás kapható a tanító adathalmazban lévő rekordok

címkéjére (lásd 17. ábra). A címkék felülvizsgálhatóak, így az adathalmaz minősége javítható.

Ezzel a módszerrel az elemzők munkája is pontosítható, hiszen értékelni tudjuk, hogy az egyes elemekre adott predikciójuk mennyire volt pontos. Amennyiben valamit nem ítéltek kártékonynak, de az AutoEncoder anomáliának határozza meg, az adott felhasználó alaposabb átvizsgálása/felülvizsgálása indítható.

4 Befejezés

4.1 Elért eredmények, összefoglalás

A munkám végére egy olyan döntéstámogató rendszert készítettem el, mely viszonylag magas pontossággal képes előre jelezni egy regisztráló felhasználóról, hogy fog-e kártékony tevékenységet folytatni a platformon vagy sem. Ez az ajánlórendszer az elemzők számára egy erős támaszkodási lehetőséget biztosít. Későbbiekben a kézi munkát is felválthatja, ezzel sokkal gyorsabb regisztrációs folyamatot eredményezve, ugyanis nem kell kivárni, hogy az elemzők az eseteket kézzel, sok idő alatt megvizsgálják. Ez a szolgáltatással kapcsolatos felhasználói elégedettségi szintet fogja megnövelni. Az osztályozás sokkal determinisztikusabbá válik, az elemzők emberi tényezői nem lesznek akkora hatással a döntésre, így sokkal konzisztensebb lehet a szűrő.

Az adatbázisokban struktúrát alakítottam ki, így rajtuk már végezhető további feldolgozás, a későbbiekben több területen (pl. üzleti analízis) is felhasználhatóvá válnak. Részletes leírást készítettem az egyes tulajdonságokról, egy összeszedett dokumentáció született meg róluk.

A tulajdonság analízis során kapott eredményeket a cégnél többször demonstráltam az elemzőknek, akiknek rávilágítottam bizonyos, eddig nem olyan hangsúlyosan kezelt tulajdonságok fontosságára, ezzel javítva a munkájuk pontosságát, mely segítségével még több, az IBM Cloudot, illetve a céget megkárosító felhasználó szűrhető ki előzetesen.

Fontos eredmény, hogy az osztályozási folyamat megváltozott, bekerült egy visszacsatolás a rendszerbe a nem felügyelt anomália detekciós eljárásnak köszönhetően. Ennek segítségével a felhasználókkal kapcsolatos döntések és a historikus adathalmaz minősége felülvizsgálható, idővel javítható. Ez a visszacsatolás lehetővé teszi, hogy a bizonytalan minőségű, több forrásból érkező adathalmazon a címkék minőségét javítsunk, így a jövőben pontosabb modelleket tanítsunk.

Ugyan a működő rendszer így is egy viszonylag pontos eredményt ad, a munka lehetséges folytatása a gépi tanuló algoritmusok paraméteroptimalizálása, továbbá más modellek alkalmazása a már megtisztított és előkészített adathalmazra, valamint a

folyamatos változások, illetve trendek lekövetésének megoldása. Fontos lenne, hogy az adatok minőségét javítani képes visszacsatolás implementálásra kerüljön a cég belső működésében, ezáltal tapasztalatokhoz juthatnánk a módszer működőképességének tekintetében.

5 Irodalomjegyzék

[1] Laurens van der Maaten, G. H. (2008). Visualizing Data using t-SNE. *Journal of Machine Learning Research 9*, 2579-2605 . Forrás: JMLR.

[2] Leland McInnes, J. H. (18 Sep 2020). arXiv. *1802.03426v3 [stat.ML]* . Forrás: UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction: https://arxiv.org/pdf/1802.03426.pdf

[3] DmitryUlyanov. (2020. 10 28). *Multicore t-SNE.* Forrás: Github: https://github.com/DmitryUlyanov/Multicore-TSNE

[4] 9912773d, L. M. (2020. 10 28). *Performance Comparison of Dimension Reduction Implementations.* Forrás: UMAP: https://umap-learn.readthedocs.io/en/latest/benchmarking.html

[5] Aggarwal, C. C. (2015). Outlier Analysis. *Data Mining*, 237-263.

[6] Andy Coenen, A. P. (2020. 10 28). *Github*. Forrás: Understanding UMAP: https://pair-code.github.io/understanding-umap/

[7] *About - home page*. (2020. 10 28). Forrás: cvvnumber: https://www.cvvnumber.com/

[8] Ahn L., B. M. (2003). CAPTCHA: Using Hard AI Problems for Security. *Biham E. (eds) Advances in Cryptology — EUROCRYPT 2003. EUROCRYPT 2003. Lecture Notes in Computer Science, vol 2656.* , 294-311.

[9] Breiman, L. (2001). Random Forests. *Machine Learning 45*, 5-32.

[10] *Fighting return through better customer experience*. (2020). Forrás: Signifyd: https://www.signifyd.com/blog/2020/09/23/combating-return-fraud-with-cx/

[11] Hill, T. P. (1995). A Statistical Derivation of the Significant-Digit Law. *Statist. Sci. Volume 10, Number 4*, 354-363.

[12] Jackson, J. E. (1991). *A User's Guide To Principal Components.* ISBN 0-471-62267-2 .

[13] Jon T.S.Quah, M. (2008). Real-time credit card fraud detection using computational intelligence. *Expert Systems with Applications Volume 35, Issue 4*, 1721-1732.

[14] K.G Coffman, A. O. (1998). The Size and Growth Rate of The Internet. *First Monday*.

[15] M. Baak, R. K. (2019 March). A new correlation coefficient between categorical, ordinal and interval variables with Pearson characteristics. *arXiv:1811.11440v2 [stat.ME] 9 Mar 2019*.

[16] Mark J. Nigrini, J. T. (April 2012). In *Benford's Law: Applications for Forensic Accounting, Auditing, and Fraud Detection.* ISBN: 978-1-118-15285-0.

[17] Univeristy, S. (2020. 10 28). *Autoencoders.* Forrás: UFLDL Tutorial: http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/

[18] Varun Chandola, A. B. (2009). Anomaly Detection: A Survey. *ACM Computing Surveys*.

[19] Wendy K Tam Cho, B. J. (2012). Breaking the (Benford) Law Statistical Fraud Detection in Campaign Finance. *The American Statistician Volume 61, 2007 - Issue 3*, 218-223.