UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH
PRÍRODOVEDECKÁ FAKULTA

GENERATÍVNE SÚPERIACE SIETE A DETEKCIA ANOMÁLIÍ V OBLASTI BANKOVÝCH PODVODOV

2022 **Bc. Laura VIŠTANOVÁ**

UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH
PRÍRODOVEDECKÁ FAKULTA

GENERATÍVNE SÚPERIACE SIETE A DETEKCIA ANOMÁLIÍ V OBLASTI BANKOVÝCH PODVODOV

DIPLOMOVÁ PRÁCA

Študijný program:	Analýza dát a umelá inteligencia
Pracovisko (katedra/ústav):	Ústav informatiky
Vedúci diplomovej práce:	RNDr. Ľubomír Antoni, PhD.

Košice 2022

Bc. Laura VIŠTANOVÁ

Univerzita P. J. Šafárika v Košiciach
Prírodovedecká fakulta

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta:	Bc. Laura Vištanová
Študijný program:	analýza dát a umelá inteligencia (medziodborové štúdium, magisterský II. st., denná forma)
Študijný odbor:	Informatika, Matematika
Typ záverečnej práce:	Diplomová práca
Jazyk záverečnej práce:	slovenský
Sekundárny jazyk:	anglický

Názov: Generatívne súperiace siete a detekcia anomálií v oblasti bankových podvodov

Názov EN: Generative adversarial networks and detection of anomalies in bank fraud

Cieľ:
1. Spracovať prehľad metód a algoritmov na detekciu anomálií.
2. Popísať metódy generatívnych súperiacich sietí a ich možnosť použitia pre štruktúrované údaje.
3. Navrhnúť a implementovať model generatívnych súperiacich sietí a navrhnúť jeho použitie pri detekcii anomálií v oblasti bankových podvodov.
4. Vyhodnotiť a porovnať presnosť implementovaných modelov na reálnych dátach.

Literatúra:
1. AGGARWAL, C. C. Data Mining: The Textbook. New York: Springer, 2015.
2. GOODFELLOW, I., et al. Generative adversarial nets. Advances in neural information processing systems 27, 2014.
3. CHEN, T., GUESTRIN, C. Xgboost: A scalable tree boosting system [online]. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 2016. p. 785-794.

Kľúčové slová: detekcia anomálií, odhaľovanie podvodov, generatívne súperiace siete

Vedúci: RNDr. Ľubomír Antoni, PhD.
Oponent: RNDr. Andrej Gajdoš, PhD.
Ústav : ÚINF - Ústav informatiky
Riaditeľ ústavu: doc. RNDr. Ondrej Krídlo, PhD.

Dátum schválenia: 29.04.2022

doc. RNDr. Ondrej Krídlo, PhD.
riaditeľ Ústavu informatiky

Poďakovanie

Týmto by som sa chcela poďakovať svojmu vedúcemu práce RNDr. Ľubomírovi Antonimu, PhD. za odborné vedenie, jeho cenné rady a ochotu kedykoľvek pomôcť. Za štylistickú a gramatickú úpravu by som sa chcela poďakovať Mgr. Lucii Kekeňákovej.

Abstrakt

Generatívne súperiace siete (GAN siete) patria medzi nové metódy v oblasti analýzy dát a umelej inteligencie. Tieto siete boli pôvodne navrhnuté pre generovanie umelých obrázkov, avšak s postupom času sa ich užitočnosť preukázala aj pri generovaní iných typov dát, ako napr. tabuľkových dát. V tejto práci popisujeme všeobecnú štruktúru GAN sietí, ako aj špeciálne GAN siete vhodné na generovanie tabuľkových dát. Podrobnejšie sa venujeme sieťam TGAN a CTGAN. Značná časť práce je venovaná anomáliám a detekcii anomálií. V poslednej kapitole prezentujeme nový postup pre riešenie problému detekcie anomálií pomocou GAN sietí, ktorý aplikujeme na reálne dáta z oblasti detekcie podvodov v bankových transakciách. Porovnávame rôzne výbery atribútov z pôvodnej dátovej sady, ako aj rôzne algoritmy pre detekciu anomálií. V závere práce prezentujeme výsledky, ktoré potvrdzujú užitočnosť použitia GAN sietí pre problém odhalenia podvodníkov, nakoľko sa nám podarilo pomocou GAN sietí na tých istých dátach dosiahnuť lepšie výsledky ako bez nich.

Kľúčové slová: metódy detekcie anomálií, generatívne súperiace siete, odhaľovanie podvodníkov, umelé anomálie

Abstract

Generative adversarial networks (GAN networks) represent the novel methods in data analysis and artificial intelligence. These networks were proposed to generate artificial images. However, their possible applications were recently shown for generating other types of data, e.g., tabular data. In this thesis, we describe the general structure of GAN networks and their alternative architecture appropriate for generating tabular data. We focus on the TGAN and CTGAN networks in more detail. The important part of this thesis is devoted to anomalies and their detection. In the last chapter, we present a new method for solving the problem of anomaly detection using GAN networks, which we apply to real data in the field of fraud detection in banking transactions. We compare different attribute selections from the original dataset, as well as different anomaly detection algorithms. At the end of this thesis, we present the results, which confirm the effect of using GAN networks for the issue of detecting fraudsters. We achieved better results using GAN networks on the same data than without them.

Keywords: anomaly detection methods, generative adversarial networks, fraud detection, artificial anomalies

Obsah

Zoznam ilustrácií

Zoznam tabuliek

Úvod

V posledných rokoch sa na oblasť analýzy dát a umelej inteligencie upriamuje čoraz väčšia pozornosť. Je to spôsobené zvyšujúcou sa informatizáciou vo väčšine oblastí nášho života, ktorá nám umožňuje zhromažďovanie veľkého množstva dát.

Zbieranie a uchovávanie dát síce máme k dispozícii, máme dostatočné množstvo dát, avšak ozajstnou výzvou ostáva ich spracovanie. Vďaka ich rôznorodosti a zdrojov, z ktorých pochádzajú, sa v posledných rokoch vyvinuli rôzne metódy a prístupy analýzy dát, pričom takmer každý deň vznikajú nové prístupy. Neexistuje jedinečný spôsob, akým extrahovať informácie z dát a to robí odvetvie analýzy dát takým zaujímavým. Všetky prístupy musíme prispôsobiť prostrediu, z ktorých pochádzajú dáta, štruktúre dát a ich vlastnostiam. Napriek tomu, že dnes je už známych mnoho algoritmov a prístupov na ich spracovanie, často nestačí iba aplikovať jednotlivé metódy. Potrebujeme ich mierne modifikovať, aby sme z vhodne predspracovaných dát získali čo najviac poznatkov. Nie je nezvyčajné, aby predtým, ako nájdeme správny postup pre danú dátovú sadu, viaceré zlyhali. Aj takéto zlyhania nás však vedia posunúť dopredu, ak dokážeme aspoň čiastočne odhaliť ich príčinu. V oblasti analýzy dát sa často používajú aj rôzne kombinácie algoritmov, o čo sme sa pokúsili aj v tejto diplomovej práci.

Detekcia anomálií je rozšírenou podoblasťou analýzy dát. Využíva sa takmer v každom odvetví, ktoré si vieme predstaviť. V tejto práci sme sa zamerali na detekciu podvodníkov v reálnych bankových dátach. Keďže podvodných transakcií je výrazne menej ako tých platných, podvodné transakcie môžeme uvažovať ako anomálie. Dáta sme prevzali zo súťaže vyhlásenej na kaggle.com a aplikovali sme na ne nový prístup detekcie anomálií.

Najväčším problémom detekcie anomálií je fakt, že anomálie nie sú časté v dátach, a teda je ťažké sa ich učiť. Hlavnou myšlienkou tejto práce bolo aplikovanie GAN sietí, ktoré umožňujú generovanie umelých dát na základe reálnych. Takto vieme doplniť dátovú sadu o umelých podvodníkov, vďaka čomu zvýšime ich pomer v dátach a tým uľahčíme učenie anomálií.

V tejto práci prezentujeme nami navrhnutý postup detekcie anomálií, ako aj podrobnejší popis jednotlivých častí postupu. V prvej kapitole predstavíme viaceré definície anomálií a ich rôzne typy. V druhej kapitole sa venujeme algoritmom na vyhľadávanie anomálií, ktoré sme aplikovali na naše dáta. Popisujeme najmä 3 metódy,

a to rozhodovacie stromy, náhodný les a metódu XGBoost. Tretia kapitola je venovaná GAN sieťam, ich základnej myšlienke, rôznym typom sietí. V štvrtej kapitole rozvíjame GAN siete pre tabuľkové dáta. V nej popisujeme ťažkosti spojené s generovaním tabuľkových dát a dva modely TGAN a CTGAN. V závere práce je predstavená dátová sada, popísaný presný postup detekcie anomálií s medzivýsledkami a následným porovnaním konečných výsledkov.

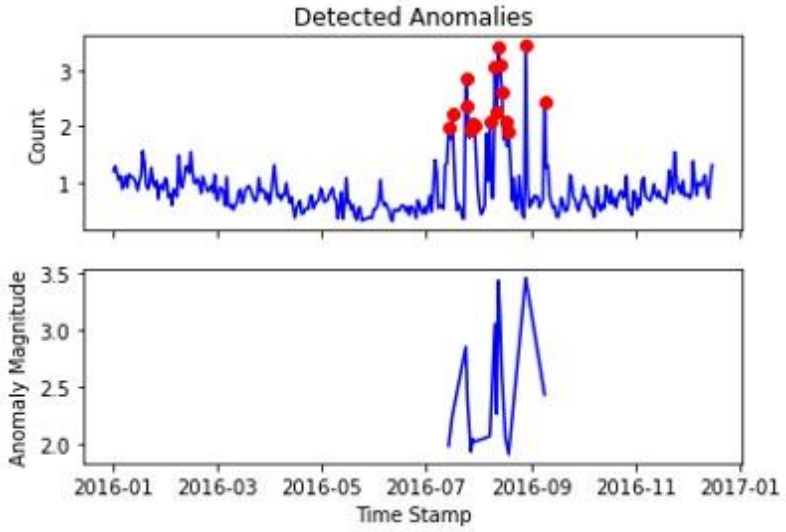
1 Metódy detekcie anomálií

Pojem anomália má dôležité postavenie v oblasti analýzy dát, strojového učenia a umelej inteligencie. V tejto kapitole definujeme pojem anomálie, prezentujeme rôzne typy anomálií a poskytujeme prehľad základných metód na ich detekciu [1] [13].

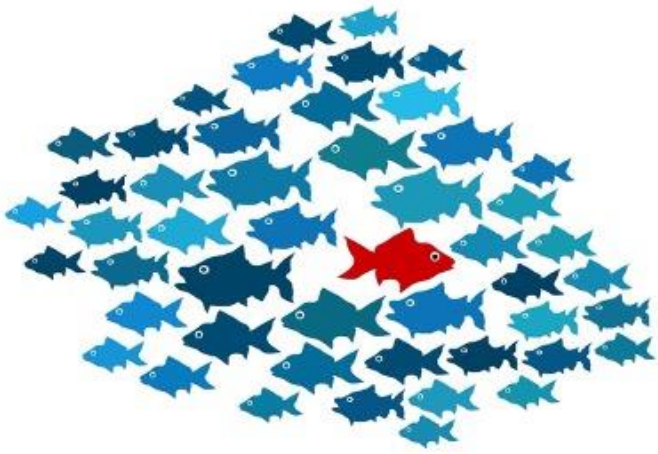
1.1 Definícia anomálie

Anomáliou nazývame také pozorovanie v dátach, ktoré vnímame ako ťažko vysvetliteľnú odchýlku od normálneho stavu. Táto definícia je ťažkopádna a vágna. Anomália vo výsledku EEG vyšetrenia bude vyzerať inak, ako anomália v spotrebe domu, keďže aj normálne správanie týchto systémov je výrazne odlišné.

Na obrázku 1 a 2 vidíme dve rôzne situácie. V oboch znamená anomália niečo iné. Na obrázku 1 máme priebeh spotreby domu v jednej miestnosti, červené body sa vyhodnotili ako anomálie. Voľným okom vidíme, že v tých bodoch nadobúda modrá funkcia vyššie hodnoty ako vo väčšine bodov. Na druhom obrázku (prevzatom z [15]) sa anomália dá jednoducho odhaliť, je to červená rybka plávajúca opačným smerom ako ostatné.



Obrázok 1 Anomálie v spotrebe domu



Obrázok 2 Ďalší príklad anomálie [15]

Anomáliu podľa Hawkinsa môžeme definovať nasledovne: *„Anomália je také pozorovanie, ktoré sa natoľko líši od ostatných pozorovaní, až to vzbudzuje podozrenie, že bolo generované iným mechanizmom.“* Na anomálie sa vieme tiež pozrieť ako na opak klastrov. Pri klastrovaní sa totiž snažíme podobné dáta spojiť do klastrov, kým pri hľadaní anomálií sa snažíme odhaliť individuálne dáta, ktoré sa výrazne líšia od ostatných.

Anomálie sa často objavujú v rôznych odvetviach analýzy dát, ako napríklad:

- *predspracovanie dát:* v dátach často nachádzame šum, ktorý zvyšuje chybu modelov, preto je vhodné ho odstrániť. Keďže takéto pozorovania zodpovedajú definícii anomálie, tak použitie metód pre detekciu anomálií sa javí ako užitočný nástroj;
- *odhaľovanie podvodníkov:* správanie podvodníkov sa často výrazne líši od správania bežných ľudí, preto sa tiež dá charakterizovať ako anomália;
- *medicína:* odchýlky vo výsledkoch rôznych vyšetrení môžu indikovať ochorenia, preto môžu metódy detekcie anomálií posunúť úroveň diagnostiky na vyššiu úroveň.

Väčšina metód detekcie anomálií vytvára model normálneho správania a následne definuje anomálie ako pozorovania, ktoré sa do daného modelu nehodia. Mieru, nakoľko sa pozorovanie nehodí do modelu, nazývame mierou anomality. Pre každé pozorovanie môžeme vyjadriť mieru anomality jedným z dvoch spôsobov:

- **reálne číslo**: vyjadruje, nakoľko sa dané pozorovanie môže považovať za anomáliu, vyššia hodnota indikuje vyššiu mieru anomality;

- **binárna hodnota**: indikuje, či je pozorovanie anomália alebo nie. Napriek tomu, že takýto výstup poskytuje menší objem informácií ako predchádzajúci, najčastejšie požadujeme práve takýto výstup, keďže v konečnom dôsledku potrebujeme iba informáciu, či je pozorovanie anomália alebo nie. Ak sa totiž pozrieme na prípad podvodníkov, tak je pre nás dôležitá iba informácia, či dané pozorovanie zodpovedá podvodníkovi alebo nie. Informácia, či je jeden podvodník väčší podvodník ako iný je pre nás irelevantná, my potrebujeme zakročiť proti všetkým.

1.2 Typy anomálií

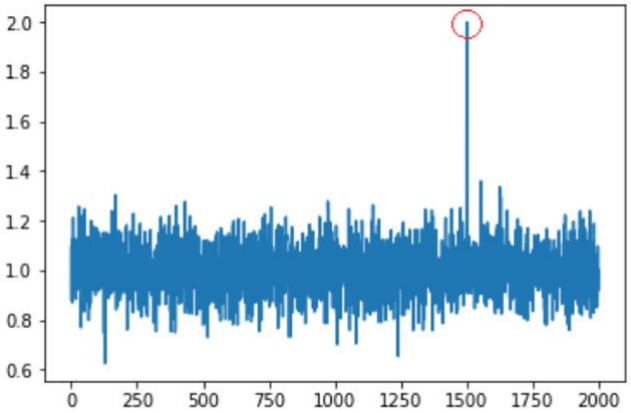
Anomálie môžeme podľa ich typu kategorizovať podľa viacerých kritérií. Najčastejšie delíme anomálie do troch skupín:

- bodové anomálie,
- kontextuálne anomálie,
- kolektívne anomálie.

1.2.1 Bodové anomálie

Bodovou anomáliou nazývame pozorovanie, ktoré sa dá považovať za anomáliu v porovnaní s ostatnými dátami. Najčastejšie sa skúmajú práve anomálie tohto typu. Na obrázku 3 môžeme vidieť systém, ktorého namerané hodnoty normálne nepresahujú hodnotu 1,4. V bode 1500 vidíme pozorovanie, v ktorom je nameraná hodnota 2,0. Výrazne presahuje normálne namerané hodnoty, a teda tento bod nazveme anomáliou.

Ako príklad z reálneho sveta si vieme predstaviť platobné transakcie. Napríklad stále platíme kartou menej ako 100 eur. Ak nám však pribudne platba v hodnote 500 eur, tak ju považujeme za bodovú anomáliu, ktorá môže poukázať či už na odcudzenie karty, alebo výnimočný nákup.

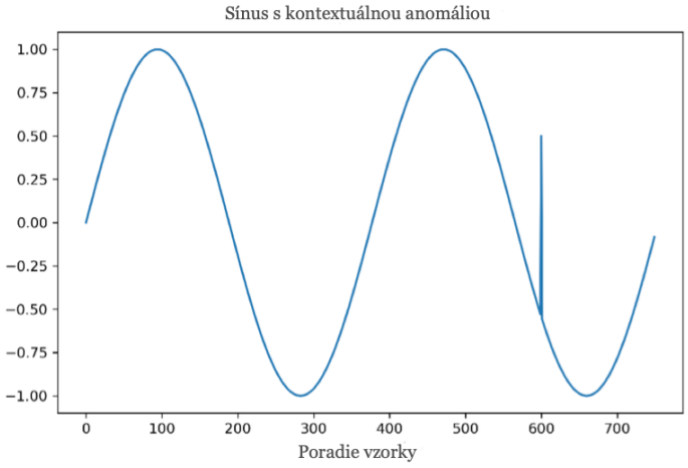


Obrázok 3 Príklad bodovej anomálie

1.2.2 Kontextuálne anomálie

Kontextuálnou anomáliou nazývame pozorovanie, ktoré síce nie je anomáliou v porovnaní s ostatnými dátami, ale je anomáliou v kontexte dát. Ak sa pozrieme na obrázok 4, tak môžeme vidieť, že nameraná hodnota v bode 600 nie je hodnotovo vyššia alebo nižšia ako v iných bodoch, ale nezapadá do kontextu dát. Ak by sme totiž sledovali priebeh hodnôt, tak by sme očakávali nameranú hodnotu okolo –0,5 namiesto nameraných 0,5.

Takéto chyby môžu byť aj výsledkom zlého merania, ale pozrime sa, ako by takáto anomália vyzerala pri platbách. Predstavme si, že každý týždeň chodíme v piatok na veľký nákup v hodnote 100 eur a v utorok na malý nákup v hodnote 30 eur. Ak prídu sviatky a my nakúpime v utorok za 70 eur, tak táto hodnota bude kontextuálnou anomáliou. Hodnota nášho nákupu síce nepresiahla zvyčajné maximum 100 eur, ale v ten deň sme očakávali nákup za 30 eur.

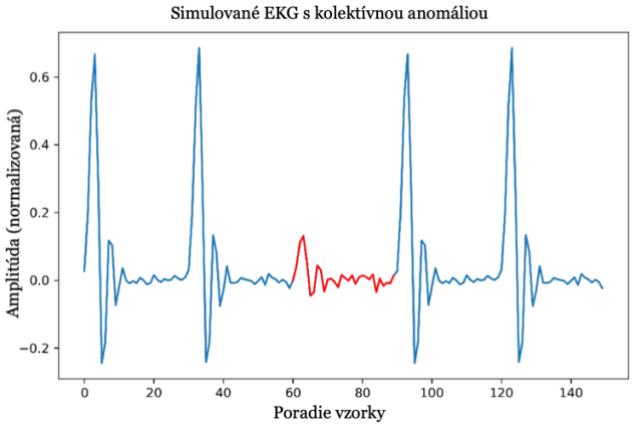


Obrázok 4 Príklad kontextuálnej anomálie

1.2.3 Kolektívne anomálie

Postupnosť bodov nazývame kolektívnou anomáliou, ak je táto postupnosť anomáliou v porovnaní s ostatnými dátami. Jednotlivé body tejto postupnosti nemusia byť samostatne anomáliami, ale ich spoločný výskyt z nich robí kolektívnu anomáliu.

Ako príklad z reálneho sveta si môžeme predstaviť výsledok EKG vyšetrenia, ktorý je uvedený na obrázku. Ako môžeme vidieť, červená množina bodov je v tomto prípade kolektívnou anomáliou, pretože táto postupnosť nezodpovedá normálnemu správaniu systému (modré časti), aj keď jednotlivé body nemusia byť anomáliami.



Obrázok 5 Príklad kolektívnej anomálie

1.3 Metódy detekcie anomálií

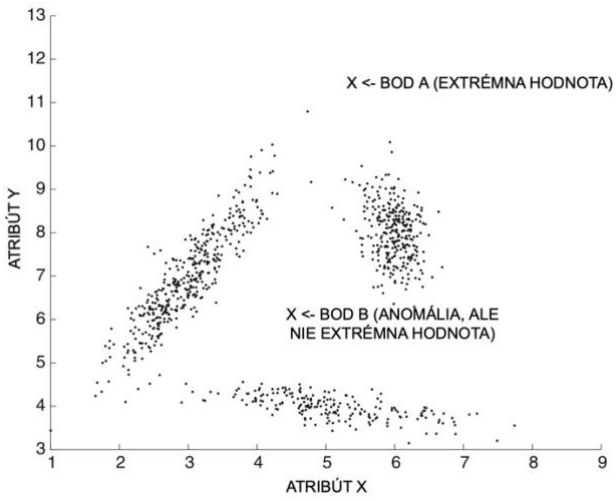
Anomálie majú jednu významnú vlastnosť, ktorá spôsobuje základný problém v ich detekcii. Anomálie sú z definície také pozorovania, ktoré sa výrazne odlišujú od normy, z čoho vyplýva, že tieto pozorovania sa v dátach objavujú iba zriedka. Keďže ich zastúpenie je v dátach veľmi malé, je náročné sa naučiť ich odhaliť. V tejto podkapitole si predstavíme základné prístupy k detekcii anomálií. Rozlišujeme 6 základných prístupov:

- analýza extrémnych hodnôt,
- zhlukovacie modely,
- modely založené na vzdialenosti,
- modely založené na hustote,
- pravdepodobnostné modely,
- informačno-teoretické modely.

V nasledujúcich podkapitolách si stručne predstavíme niektoré z nich.

1.3.1 Analýza extrémnych hodnôt

Základnou myšlienkou analýzy extrémnych hodnôt je označiť za anomálie tie dátové body, ktoré ležia na okraji distribúcie pravdepodobnosti hodnôt. Je ale dôležité si uvedomiť, že takýmto spôsobom dokážeme odhaliť iba niektoré anomálie v dátach. Platí totiž, že síce všetky extrémne hodnoty môžeme považovať za anomálie, opak však nie je pravdou. Nie každá anomália je extrémna hodnota. Jednoduchým jednorozmerným príkladom je množina {1, 3, 3, 3, 50, 97, 97, 97, 100}. Hodnoty 1 a 100 môžeme považovať za extrémne hodnoty a teda aj anomálie, zaujímavá je však hodnota 50. Priemer všetkých hodnôt v množine je 50, teda túto hodnotu nebudeme považovať za extrémnu. Avšak práve táto hodnota je najviac izolovaná od zvyšku pozorovaní, preto by sme ju mali považovať za anomáliu. Ak sa pozrieme na nasledujúci obrázok, tak môžeme vidieť dvojrozmerný príklad, v ktorom je bod A extrémnym bodom. Bod B z hľadiska rozdelenia pravdepodobnosti nie je extrémny bod, avšak je výrazne izolovaný od ostatných bodov, teda ho môžeme považovať za anomáliu.



Obrázok 6 Príklad viacrozmerných extrémnych hodnôt

Základnou myšlienkou tohto prístupu je nájsť vhodný model pre rozdelenie pravdepodobnosti a podľa neho definovať mieru anomality pre jednotlivé pozorovania. Najčastejšie používaným modelom je normálne rozdelenie s hustotou v tvare:

$$f(x) = \frac{1}{\sigma \cdot \sqrt{2\pi}} \cdot e^{\frac{-(x-\mu)^2}{2 \cdot \sigma^2}}.$$

Často pracujeme so štandardným normálnym rozdelením, v ktorom sú hodnoty parametrov $\mu = 0$ a $\sigma = 1$. V prípade, že máme k dispozícii veľké množstvo pozorovaní, dokážeme pomerne presne odhadnúť tieto hodnoty. Potrebujeme ešte mieru anomality pre jednotlivé pozorovania, ktorú si definujeme ako Z-hodnotu náhodnej premennej. Pre pozorovanie x_i definujeme hodnotu z_i nasledujúcim vzťahom:

$$z_i = \frac{x_i - \mu}{\sigma}$$

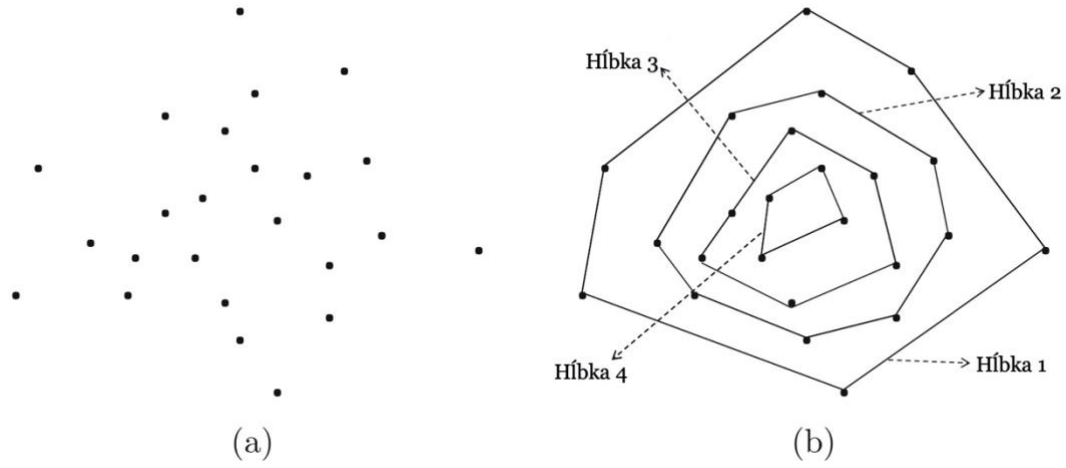
Vyššie hodnoty indikujú horný okraj rozdelenia, nižšie hodnoty zase dolný okraj rozdelenia. Štatistici v tomto prípade používajú ako prah hodnotu 3. To znamená, že ak je hodnota z_i pre pozorovanie x_i v absolútnej hodnote aspoň 3, tak sa dané pozorovanie vyhodnotí ako anomália. Dá sa ukázať, že v tomto prípade označíme ako anomálie nanajvýš okolo 0,01 % dát.

V prípade malého množstva n pozorovaní sa používa Studentovo t-rozdelenie s n stupňami voľnosti, ktoré konverguje k normálnemu rozdeleniu. V prípade viacrozmerných dát je postup podobný, podrobnejší popis sa nachádza na stranách 242 – 243 v knihe [1].

Zaujímavým prístupom, ktorý patrí do tejto kategórie, je aj nasledujúca metóda založená na hĺbke. Pozostáva z nasledujúcich krokov, ktoré sa opakujú, kým máme k dispozícii dátové body:

- vytvorí sa konvexný obal na dátach,
- odstránia sa tie dáta, ktoré tvoria konvexný obal a priradí sa im poradové číslo iterácie.

Za mieru anomality v tomto prípade považujeme priradené poradové číslo, teda čím skôr bolo pozorovanie odstránené, tým viac ho môžeme považovať za anomáliu. Nasledujúci obrázok znázorňuje, v akom poradí sa budú jednotlivé body odstraňovať.

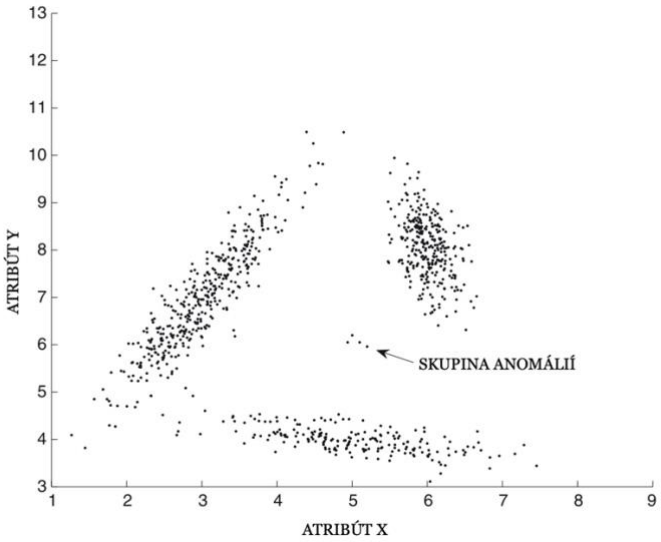


Obrázok 7 Príklad metódy založenej na hĺbke

1.3.2 Zhlukovacie metódy

Ako už bolo spomenuté, na hľadanie anomálií sa môžeme pozerať ako na komplementárny problém k hľadaniu zhlukov v dátach. Pri zhlukovaní totiž platí, že dané pozorovanie buď patrí do niektorého zhluku, alebo sa považuje za anomáliu. Z toho dôvodu sa ako vhodný spôsob detekcie anomálií javí použitie algoritmov na zhlukovanie, ktorých vedľajším produktom sú práve anomálie. Aj v tomto prípade potrebujeme definovať mieru anomality pre jednotlivé dátové body. Jednoduchý spôsob, ako to urobiť, je definovať mieru anomality ako vzdialenosť bodu od centra najbližšieho klastra.

Anomálie zvyknú vytvárať malé zhluky, keďže sa chyba môže rovnakým spôsobom viackrát zopakovať. Príklad takého malého zhluku anomálií môžeme vidieť aj na nasledujúcom obrázku.



Obrázok 8 Príklad malého zhluku anomálií

Mohli by sme sa preto obávať, že takéto anomálie algoritmus vyhodnotí ako zhluk. Algoritmy pre zhlukovanie väčšinou berú do úvahy iba väčšie zhluky, preto sa očakáva, že vo výstupe budú tieto pozorovania aj napriek formovaniu zhluku vyhodnotené ako anomálie. Táto vlastnosť nie je zachovaná pri použití metód založených na hustote, preto keď očakávame menšie skupiny anomálií, je vhodnejšie použiť zhlukovacie metódy.

Použitie metód zhlukovania sa teda javí ako užitočný spôsob pre detekciu anomálií. Nesmieme však zabudnúť, že tieto metódy nie sú optimalizované na riešenie nášho problému, preto ich výstup nemusí vždy zodpovedať realite.

1.3.3 Metódy založené na hustote

Metódy založené na hustote sú založené na podobnom princípe ako metódy pre zhlukovanie podľa hustoty. Vyhľadávajú sa riedke oblasti v dátach, ktoré sa potom označia ako anomálie. Na tento účel sa môže použiť histogram, ktorým takéto oblasti dokážeme odhaliť. V jednorozmernom prípade si ľahko vieme predstaviť histogram, ktorý pozostáva z obdĺžnikov rôznej veľkosti nad osou x. Mieru anomality pre jednotlivé pozorovania vieme definovať ako počet pozorovaní, ktoré sa s ním nachádzajú v jednom obdĺžniku. Analogický prístup použijeme aj vo viacrozmernom prípade. Každú dimenziu rozdelíme na ekvidištančné intervaly a určíme si maximálny počet pozorovaní vo viacrozmernom obdĺžniku pre anomálie.

Treba podotknúť, že táto metóda sa veľmi nerozšírila, keďže so stúpajúcou dimenziou dát je čoraz náročnejšie zvoliť vhodné delenie rozsahu dát pre jednotlivé dimenzie a tiež prahy pre počet pozorovaní v jednom obdĺžniku. Aj preto sme v predchádzajúcej kapitole spomínali, že aplikácia metód založených na hustote nie je vhodná na dátach ako na obrázku 8. Pri d dimenziách totiž budeme mať aspoň 2^d obdĺžnikov, teda počet pozorovaní, ktoré padnú do jedného obdĺžnika exponenciálne klesá s narastajúcou dimenziou. Môže sa preto stať, že algoritmus vyhodnotí ako anomálie oveľa väčšiu časť dát.

2 Vybrané algoritmy strojového učenia na detekciu anomálií

V tejto kapitole prezentujeme vybrané algoritmy strojového učenia, ktoré je možné aplikovať na detekciu anomálií. Popisujeme rozhodovacie stromy, náhodné lesy [1] a novú metódu XGBoost [20], [14].

2.1 Rozhodovacie stromy

Rozhodovacie stromy patria medzi známe metódy na riešenie klasifikačných úloh. Sú jednou z najrozšírenejších metód aj vďaka tomu, že pomocou nich sa dá jednoducho popísať proces klasifikácie.

Táto metóda vytvára na základe tréningovej množiny strom. V strome sa nachádzajú dva typy vrcholov – vnútorné vrcholy a listy (koreň sa považuje za vnútorný vrchol). V každom kroku vytvárania stromu sa dáta delia podľa deliaceho pravidla. Deliace pravidlo nám určuje podmienku, ktorú majú spĺňať dáta v ľavom a pravom podstrome. Ak máme dáta o ľuďoch, tak deliť ich môžeme napríklad podľa pohlavia, a teda do ľavého podstromu zaradiť všetky dáta zodpovedajúce ženám a do pravého podstromu zaradiť všetky dáta zodpovedajúce mužom. Následne môžeme podstromy deliť podľa veku a tak nám v podstromoch vzniknú ďalšie dva podstromy. Ak si zvolíme hranicu 40 rokov, tak sa nám dáta rozdelia do štyroch skupín: ženy do 40 rokov, ženy nad 40 rokov, muži pod 40 rokov, muži nad 40 rokov. Keďže rozhodovacie stromy chceme použiť na klasifikáciu, potrebujeme cieľový atribút, ktorý nám dáta rozdelí do viacerých skupín. Uvažujme najjednoduchší prípad, a to binárnu klasifikáciu. Budeme sa snažiť určiť, či daná osoba získa hypotéku alebo nie. Ak dáme údaje o osobe na vstup, tak v každom vnútornom vrchole vieme určiť, do ktorého podstromu osoba patrí (podľa deliaceho pravidla) a takto sa vieme dostať k listu. V listoch sa uchováva informácia o tom, do ktorej triedy sa daná osoba klasifikuje, ak skončí v danom liste. Aby sme nemali príliš veľký strom a aby nedošlo k preučeniu, potrebujeme pri vytváraní stromov uvažovať aj kritérium zastavenia. Je zrejmé, že keď sa nám do niektorého vrcholu dostanú iba dáta z rovnakej triedy, tak tieto dáta už nepotrebujeme ďalej deliť. Z daného vrcholu urobíme list, ktorý bude vstupy klasifikovať do tej triedy, do ktorej patria všetky dáta v ňom.

Pozrime sa teraz bližšie na spôsoby delenia dát. Deliaci bod vyberáme tak, aby sme čo najviac zvýšili homogenitu vzniknutých podmnožín príkladov. Homogenitu vieme vypočítať rôznymi spôsobmi, jedným z nich je určiť hodnotu Giniho indexu. Giniho index je vhodné použiť pre kategoriálne atribúty, čím nižšia je hodnota indexu, tým lepšie vie daný atribút rozdeliť príklady do tried.

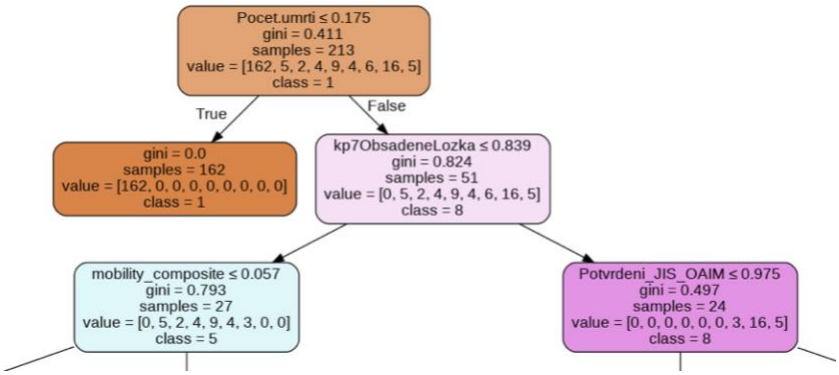
Giniho index v uzle vieme vyjadriť v tvare:

$$G(A) = \sum_{a \in A} p_{A,a,i}^2$$

kde A označuje atribút, a je možná hodnota atribútu a $p_{A,a,i}^2$ označuje pravdepodobnosť, že hodnota atribútu A v i-tom príklade tréningovej množiny bude a. Množinu budeme teda deliť podľa toho atribútu, ktorého hodnota Giniho indexu bude najmenšia.

Ukážeme si definované pojmy ešte na jednom príklade, kde bola úloha klasifikovať počet potrebných pľúcnych ventilácií v nemocniciach počas pandémie koronavírusu. V tomto prípade sa nepoužila regresia, keďže nás nezaujímal presný počet potrebných prístrojov, ale klasifikácia, ktorá určila interval potrebného počtu prístrojov. Závažnosť situácie je teda priamo úmerná triedam v klasifikácii, pretože trieda 1 vyjadrovala najnižší potrebný počet prístrojov a trieda 10 predstavovala množstvo, ktoré by zdravotníctvo už nezvládlo.

Na obrázku môžeme vidieť časť rozhodovacieho stromu. Na vrchole stromu je koreň, v ktorom sa ako deliaci atribút vybral atribút s názvom Pocet.umrti. V danom strome sa homogenita počítala podľa už spomenutého Giniho indexu, ktorý dáta rozdelil do dvoch skupín. Z obrázku vieme tiež prečítať, že väčšina dát sa dostala do ľavého podstromu, v ktorom sa nachádza 162 príkladov z tréningovej množiny (z celkového počtu 213 príkladov) a všetky sa klasifikujú do triedy 1. Môžeme tiež vidieť, že v prípade, keď atribút Pocet.umrti bol nanajvýš 0.175, tak sa vstupné dáta automaticky vyhodnotili ako patriace do triedy 1. V prípade, že bola táto hodnota vyššia, sa pozeralo na hodnotu 7-dňového kĺzavého priemeru obsadených lôžok v nemocniciach, ktorý nám opäť rozdelil dáta na dva podstromy. Takto sa pokračuje ďalej, kým nie je splnené kritérium zastavenia.



Obrázok 9 Ukážka z rozhodovacieho stromu

2.2 Náhodný les

Náhodný les vznikne generovaním viacerých nezávislých rozhodovacích stromov, ide teda o rozšírenie predchádzajúcej metódy. Rozhodovacie stromy, ktoré tvoria náhodný les, sa vytvárajú na rôznych tréningových dátach s náhodným výberom atribútov. Výsledná klasifikácia sa uskutoční tak, že každý rozhodovací strom bude klasifikovať a za celkový výsledok zoberieme ten, ktorý sa objavil u najviac stromov.

2.3 Metóda XGBoost

V poslednej podkapitole popíšeme novú metódu XGBoost [20], [14] a princípy, na ktorých je založený jej algoritmus.

2.3.1 Metódy boostingu a baggingu

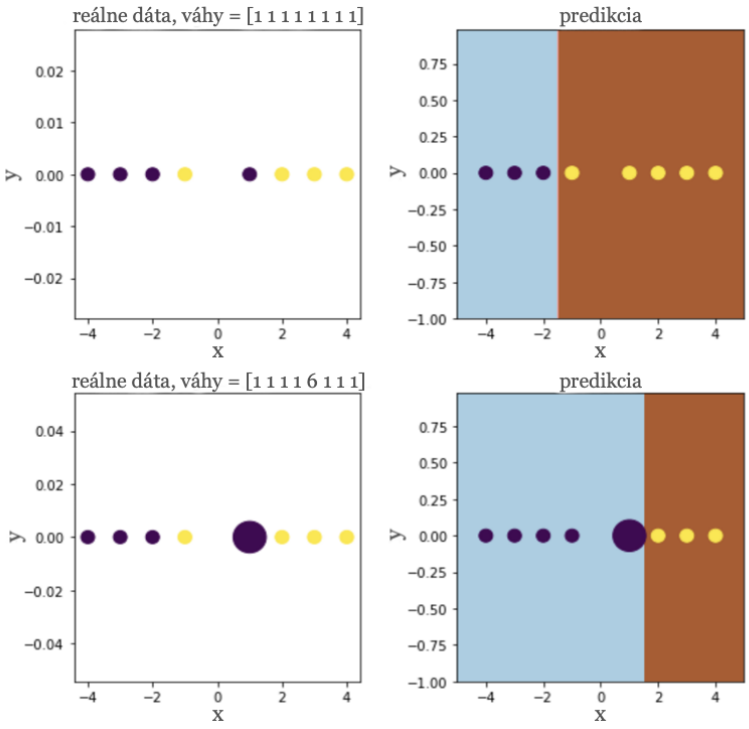
Metódy boostingu a baggingu patria medzi ensemble metódy strojového učenia. Hlavnou myšlienkou v oboch prístupoch je kombinácia viacerých slabých klasifikátorov, čím vytvoríme jeden silný klasifikátor. Metódy sa však líšia v tom, ako vytvárajú jednotlivé klasifikátory. V prípade baggingu sa tréningová množina rozdelí na podmnožiny a na každej z nich sa vytvoria rozhodovacie stromy. Takto máme viacero stromov, ktoré klasifikujú vstupné dáta, pričom celkový výstup sa vytvorí pomocou výstupov jednotlivých stromov. Chyba takého modelu je nižšia, ako by bola chyba jedného rozhodovacieho stromu.

Boosting sa skladá tiež z viacerých klasifikátorov, ale vytvárajú sa v postupnosti. Kým v prípade baggingu boli jednotlivé rozhodovacie stromy budované nezávisle od

seba, boosting vytvára stromy tak, aby ďalší vytvorený strom opravil chybu predchádzajúceho stromu.

Rozlišujeme dva základné spôsoby boostingu. V prípade AdaBoostu sa stromy budujú vždy z modifikovanej tréningovej množiny, ktorá vznikne upravením váh jednotlivých príkladov. Ak sa pozrieme na aktuálne budovaný strom, tak sa vyhodnotia príklady, na ktorých bola vysoká chyba klasifikácie. Váha týchto príkladov sa zvýši a na vytvorenie nového stromu sa posielajú dáta s modifikovanými váhami.

Na obrázku 10 vľavo hore môžeme vidieť tréningovú množinu ôsmich bodov, ktorá obsahuje dáta z dvoch tried (žltá a fialová farba). Za ním nasleduje obrázok, ktorý znázorňuje výstup prvého stromu, teda rozdelenie dát do dvoch množín. Ako môžeme vidieť, tento prvý strom sa pomýlil iba v jednom prípade, a to v prípade piateho fialového bodu, preto sa váha tohto príkladu zvýši v tréningovej množine. Zo zoznamu nad obrázkom vidíme, že sa zvolila váha 6 namiesto pôvodnej váhy 1. Následne ďalší klasifikátor kvôli vysokej váhe piateho príkladu delil dáta iným spôsobom, a teda opravil chybu predchádzajúceho stromu.



Obrázok 10 Prvé kroky boostingu na príklade

Druhý spôsob sa nazýva gradientný boosting. V ňom sa nemenia váhy jednotlivých príkladov tréningovej množiny, ale pomocou stratovej funkcie sa pozerá na

rozdiel medzi výstupom modelu a ozajstnou hodnotou cieľového atribútu. Táto metóda je vhodná ako pre regresiu, tak aj pre klasifikáciu.

V nasledujúcej časti popisujeme pseudokód gradientného boostingu so zvolenou stratovou funkciou najmenších štvorcov. Vidíme, že stromy sa budujú postupne, pričom pri každom strome sa určí hodnota stratovej funkcie, vypočíta sa záporný gradient stratovej funkcie a buduje sa model, ktorý ako cieľový atribút používa gradienty. Záporný gradient počítame z dôvodu, že chceme určiť smer, v ktorom stratová funkcia klesá najrýchlejšie. Na konci sa tieto estimátory sčítajú a tak dostaneme výsledok modelu.

nastav prvý strom F^1

pre každý strom z [1,M]:

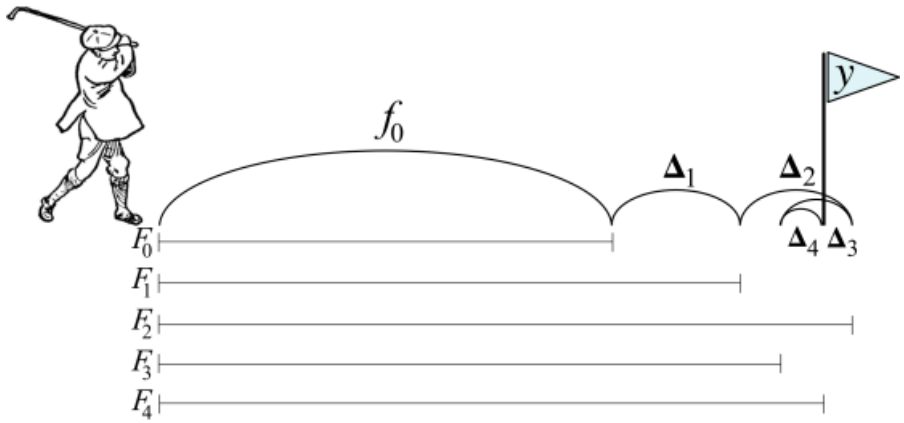
$\text{strata}^i = \sum_{j=1}^{n}(Y_j - F^i(X_j))^2$ //vypočítame stratu

$\text{neg}_{\text{gradient}} = -\frac{\partial \text{strata}^i}{\partial X_j} = -\frac{2}{n} * \left(Y_j - F^i(X_j)\right)$ pre každé i

nastav slabý strom H^i na $(X, \frac{\partial strata}{\partial X})$

Výsledná predikcia: $F^m(X) = F^i(X) + \varrho * H^i(X) = \mathrm{F}^1 + \varrho * \sum_{i=1}^{m} H^i(X)$

Tento algoritmus v skutočnosti používame aj v reálnom svete. Na obrázku 11 môžeme vidieť hru golfu, počas ktorej na začiatku hráč urobí nejaký odhad f_0, ktorý potom postupne zlepšuje a snaží sa aproximovať hodnotu y.



Obrázok 11 Aplikácia boostingu v reálnom svete

3 Generatívne súperiace siete (GAN siete)

Generatívne súperiace siete (GAN siete) patria medzi pomerne nové metódy v oblasti strojového učenia.

Prvýkrát boli popísané v roku 2014 Ianom Goodfellowom v článku [9] a odvtedy sa veľkým tempom zlepšovali a vyvíjali. GAN siete slúžia na vytvorenie umelých dát na základe tréningovej sady. Pôvodne sa používali na generovanie obrázkov, vďaka čomu vieme generovať neexistujúce ľudské tváre. Aby bol rýchly pokrok v tejto oblasti ešte viditeľnejší, uvádzame obrázok (ľudskú tvár), ktorú vygenerovala jedna z prvých GAN sietí v roku 2014 (obrázok 12) a pre porovnanie ďalší vygenerovaný obrázok pomocou StyleGAN z roku 2018 (obrázok 13).

Obrázok 12 Obrázok generovaný GAN sieťou z roku 2014

Obrázok 13 Obrázok generovaný StyleGAN sieťou z roku 2018

Vďaka GAN sieťam už v súčasnosti dokážeme generovať umelé obrázky, ktoré sú pre ľudí nerozlíšiteľné od pravých – tak, ako to môžeme vidieť na obrázku. Aby čitateľ

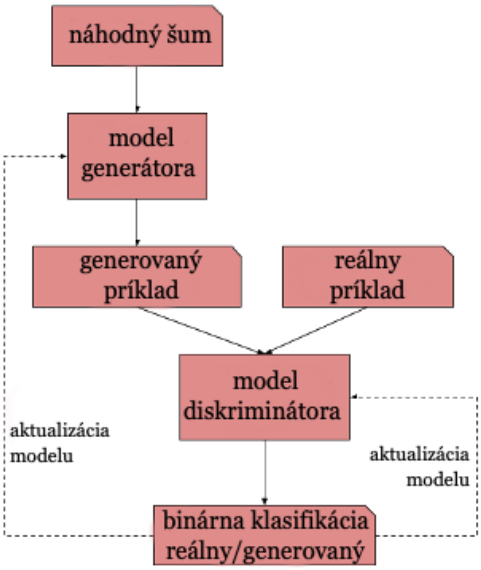
videl, že tieto siete dokážu generovať akékoľvek reálne vyzerajúce obrázky a nie len ľudské tváre, uvádzame príklad ďalších obrázkov so zvieratami, jedlom a prírodou.

Obrázok 14 Ďalšie príklady umelo vytvorených fotografií pomocou GAN sietí

V tejto kapitole predstavíme princíp fungovania GAN sietí. Vzhľadom na to, že GAN siete boli prvotne vytvorené na generovanie obrázkov a väčšina sietí sa zaoberá generovaním obrázkov, v nasledujúcej kapitole predstavíme GAN siete pre tabuľkové dáta. Popíšeme, v čom sa líšia od pôvodných sietí pre obrázky a v čom spočívali ťažkosti ich navrhnutia. Obrázky a text v tejto časti práce sú spracované na základe článku [3].

3.1 Popis GAN sietí

GAN siete [6] pozostávajú z dvoch neurónových sietí, ktoré súťažia proti sebe v hre s nulovým súčtom (výhra jednej siete je prehrou druhej). Prvá zo sietí sa nazýva *generátor*. Jej úlohou je generovanie umelých dát. Druhá sieť sa nazýva *diskriminátor* a jej úlohou je rozlíšiť umelo vygenerované vzorky od tých reálnych. Na vstupe dostáva obrázky z tréningovej sady aj tie, ktoré vytvoril generátor. Pomocou klasifikačných metód sa snaží obrázky rozdeliť do dvoch množín, a to na pravé (z tréningovej sady) a umelé (vygenerované generátorom) [6].



Obrázok 15 Štruktúra GAN sietí

Prácu GAN sietí vysvetlíme v nasledujúcej časti (obrázok 15). Trénujú sa dve siete (generátor a diskriminátor), pričom ich trénovanie prebieha separátne, teda buď trénujeme generátor, alebo diskriminátor. Váhy neurónových sietí sa aktualizujú pomocou spätnej propagácie chyby. Trénovanie diskriminátora prebieha v nasledujúcich krokoch:

I. diskriminátor dostane na vstupe obrázky z tréningovej sady a obrázky vygenerované generátorom;
II. klasifikuje vstupné obrázky do dvoch tried – pravé obrázky (z tréningovej sady) a umelo vygenerované obrázky (výstup generátora);
III. vyhodnotí sa chyba diskriminátora a spätnou propagáciou chyby sa aktualizujú váhy diskriminátora.

Trénovanie generátora je podobné. Najväčší rozdiel spočíva v tom, že váhy neurónovej siete sa upravujú na základe chyby diskriminátora. Kroky trénovania sú nasledovné:

I. generátor dostane na vstupe náhodný vektor;
II. generátor generuje umelé obrázky;
III. kvalitu vygenerovaných obrázkov určuje diskriminátor, ktorého presnosť rozhoduje o tom, či sú vygenerované obrázky dostatočne dobré;
IV. na základe chyby diskriminátora sa spätnou propagáciou aktualizujú váhy generátora.

Ako už bolo spomenuté, jednotlivé siete sa trénujú zvlášť. Z popisu jednotlivých krokov tréningu je zrejmé, že generátor postupne generuje obrázky čoraz viac podobné tréningovej sade, kým diskriminátor má čoraz väčší problém rozlíšiť, odkiaľ obrázky pochádzajú. Otázkou teda ostáva, dokedy má zmysel tieto siete trénovať. Trénovanie končí, keď obrázky generované generátorom nedokáže diskriminátor odlíšiť od ozajstných obrázkov, čiže keď už len tipuje výsledok klasifikácie (teda presnosť klasifikácie je okolo 50 %).

4 GAN siete pre tabuľkové dáta

V predchádzajúcej kapitole boli predstavené GAN siete a spôsob, akým generujú obrázky. GAN siete boli síce pôvodne navrhnuté pre potreby generovania obrázkov, ale v posledných rokoch sa výskum orientuje aj smerom generovania iných štruktúr, akými sú napríklad tabuľkové dáta. Ak chceme generovať tabuľkové dáta podobné reálnym dátam, tak sa potrebujeme vysporiadať s novými problémami, ktoré pri generovaní obrázkov neboli prítomné:

- rôzne typy atribútov;
- rôzne tvary rozdelení v dátach;
- nevyvážené kategoriálne premenné.

Pri vytváraní GAN sietí pre tabuľkové dáta bolo potrebné vyriešiť tieto problémy, aby sme dokázali efektívne generovať reálne, celočíselné alebo kategoriálne dáta. Je prirodzenou požiadavkou, aby vzniknuté dáta mali podobné štatistické vlastnosti ako tie reálne. Preto by mali tieto algoritmy vedieť generovať dáta z ľubovoľného rozloženia dát.

V nasledujúcich podkapitolách predstavíme vybrané GAN siete, ktoré generujú tabuľkové dáta a ktoré sme aplikovali aj na našu dátovú sadu. Treba podotknúť, že generovanie tabuľkových dát sa objavilo až pár rokov po predstavení GAN sietí. Preto je v čase písanie tejto práce zatiaľ relatívne málo článkov a GAN sietí, ktoré sa dajú použiť na generovanie tabuľkových dát. Obsah tejto kapitoly vznikol na základe prác [4], [18] a [17].

4.1 Tabuľkové generatívne súperiace siete (TGAN siete)

Predstavme si tabuľku, v ktorej máme numerické (diskrétne alebo spojité) a kategoriálne dáta s nejakým rozdelením dát X. Chceme vytvoriť model, ktorý bude generovať riadky z tohto rozdelenia tak, aby pre novovytvorené dáta platili nasledujúce dve podmienky:

I. ľubovoľný model strojového učenia má podobnú presnosť na nových dátach ako na pôvodných;
II. vzťah medzi ľubovoľnými dvomi atribútmi sa zachová aj v nových dátach.

Neurónové siete dokážu efektívne generovať hodnoty z rozdelenia centrovaného na (–1,1) pomocou aktivačnej funkcie *tanh* a tiež multinomické rozdelenie pomocou

aktivačnej funkcie *softmax*. Využijeme túto vlastnosť a urobíme na dátach transformáciu, aby boli splnené predpoklady. Tieto transformácie musia spĺňať jednu dôležitú vlastnosť – musia byť reverzibilné. Je to spôsobené tým, že na výstupe chceme mať tabuľku (riadok tabuľky), ktorá je na nerozoznanie od reálnej tabuľky. Preto ak urobíme transformáciu na dátach, tak sa musíme vedieť spätne vrátiť do pôvodného tvaru. Transformácie popíšeme zvlášť pre numerické a kategoriálne premenné.

4.1.1 Numerické premenné

Pri transformácii numerických premenných sa používa Gaussov model zmesí (GMM, Gaussian Mixture Model), ktorý je zovšeobecnením normálneho rozdelenia. Dáta totiž relatívne často nemajú normálne rozdelenie, ale vieme ich vyjadriť pomocou viacerých normálnych rozdelení. To znamená, že ak atribút vyjadríme pomocou m normálnych rozdelení, tak získame $\mu_1, \dots, \mu_m$ stredných hodnôt a $\sigma_1, \dots, \sigma_m$ štandardných odchýlok. Pre danú hodnotu atribútu c následne určíme pre každé z m rozdelení normalizovanú pravdepodobnosť, že c pochádza práve z toho rozdelenia. Získame tak m ďalších hodnôt $u_1, \dots, u_m$. Vyberieme najväčšiu pravdepodobnosť, označme ju u_k a normalizujeme hodnotu c nasledujúcim vzťahom:

$$v = \frac{c - \mu_k}{2\sigma_k} \qquad (*)$$

Hodnotami $u_1, \dots, u_m$ a v následne prezentujeme pôvodnú hodnotu c. V článku autori rozkladali atribúty na 5 normálnych rozdelení. Ak je ich menej, niektoré z nich sa takmer vynulujú, a teda dostaneme dobré rozdelenie pre daný atribút.

4.1.2 Kategoriálne premenné

Transformácia kategoriálnych premenných sa udeje v nasledujúcich krokoch:

- každú hodnotu d diskrétnej premennej transformujeme na one-hot vektor $\boldsymbol{d}$ takej dimenzie, koľko kategórií máme v danej premennej;
- ku každej dimenzii pridáme malý náhodný šum vyjadrený vzťahom: $\boldsymbol{d} = \boldsymbol{d} + \text{Uniform}(\gamma)$, pričom autori zvolili $\gamma = 0{,}2$;
- normalizujeme vektor $\boldsymbol{d}$: $\frac{\boldsymbol{d}}{\sum \boldsymbol{d}}$

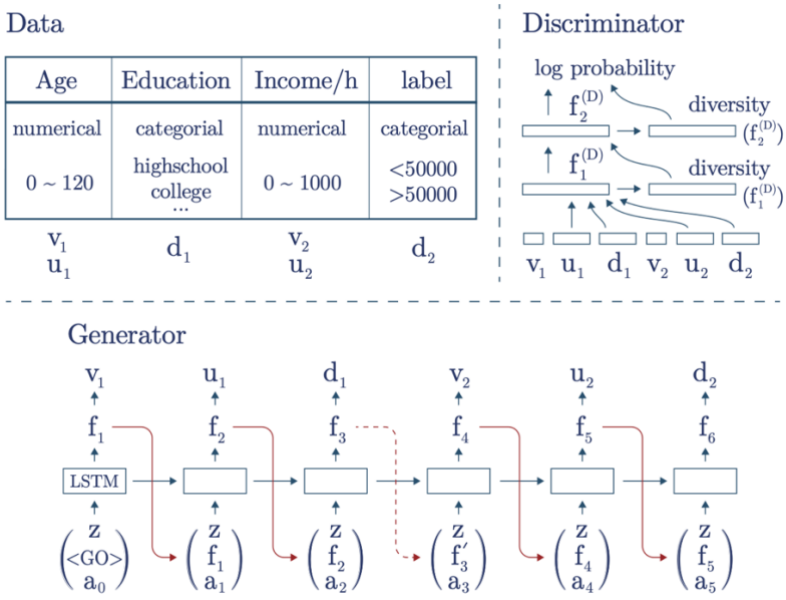
Týmto spôsobom máme popísanú vhodnú transformáciu pre numerické, ale aj pre kategoriálne premenné. GAN sieť následne generuje hodnoty $u_1, \dots, u_m$ a v pre

numerické dáta a vektor $\boldsymbol{d}$ pre kategoriálne dáta. Spätná transformácia pri numerických dátach sa robí pomocou vzťahu $(*)$, z ktorého vieme vyjadriť c. Pri kategoriálnych premenných sa z vektora $\boldsymbol{d}$ vyberie najpravdepodobnejšia kategória.

4.1.3 Model siete

V tejto časti popíšeme, ako vyzerá model diskriminátora a generátora. Aby bol popis zrozumiteľnejší, popíšeme ho na príklade z pôvodného článku.

Na obrázku vidíme tabuľku so štyrmi atribútmi (vek, vzdelanie, príjem, označenie). Dva atribúty sú kategoriálne (vzdelanie a označenie), zvyšné dva atribúty sú numerické (vek a príjem).



Obrázok 16 Príklad generovania tabuľkových dát pomocou TGAN

Generátor: Z transformácie dát vyplýva, že numerické dáta budeme generovať v dvoch krokoch (zvlášť vektor $\boldsymbol{u}$ a zvlášť skalár v), kategoriálne premenné budeme generovať v jednom kroku (vektor $\boldsymbol{d}$). Tabuľka na obrázku obsahuje dve numerické premenné a dve kategoriálne, teda generovanie jedného riadku tabuľky budeme robiť v šiestich krokoch. Generátor pozostáva z LSTM sietí, pričom na vstupe v ľubovoľnom kroku t má sieť nasledujúce tri prvky:

- náhodný vektor z;

- vektor f_{t-1} alebo f'_{t-1} (závisí od typu predchádzajúceho výstupu, či sme generovali numerický alebo kategoriálny atribút). Na začiatku nemáme vektor z výstupu, teda aplikujeme <GO> vektor, ktorý sa priebežne učíme;
- vektor a_{t-1}, v prvom kroku inicializujeme $a_0 = 0$.

Výstup LSTM je vektor h_t , ktorý použijeme pre určenie hodnoty $f_t = \tanh(W_t h_t)$, kde W_t je parameter siete, ktorý sa sieť učí. Následne použijeme vektor f_t, aby sme získali výstup. Výstup závisí od toho, aký typ premennej generujeme, preto platí:

- ak výstup je časťou v pre numerickú premennú, tak výstup určíme vzťahom: $v_i = \tanh(W_t f_t)$ a do ďalšieho kroku pošleme vektor f_t;
- ak výstup je časťou $\boldsymbol{u}$ pre numerickú premennú, tak výstup určíme vzťahom $u_i = \mathrm{softmax}(W_t f_t)$ a do ďalšieho kroku pošleme vektor f_t;
- ak výstup má byť diskrétna premenná, tak výstup určíme vzťahom $\boldsymbol{d}_i = \mathrm{softmax}(W_t f_t)$ a do ďalšieho kroku pošleme vektor daný vzťahom $f'_t = E_i[arg_k max \boldsymbol{d}_i]$, kde E je reprezentačná matica pre diskrétnu premennú.

Diskriminátor: Pozostáva z plne prepojenej siete s l vrstvami, pričom na vstupe má konkatenáciu výstupov $\boldsymbol{u}, v, \boldsymbol{d}$. Výpočet vo vrstvách sa robí pomocou nasledujúceho predpisu:

$$f_1^{(D)} = \mathrm{LeakyReLu}(\mathrm{BN}(W_1^{(D)}(v_{1:n_c} \oplus u_{1:n_c} \oplus \boldsymbol{d}_{1:n_c})))$$

$$f_i^{(D)} = \mathrm{LeakyReLu}\left(\mathrm{BN}\left(W_i^{(D)}\left(f_{i-1}^{(D)} \oplus \text{rôznorodosť}\left(f_{i-1}^{(D)}\right)\right)\right)\right), i = 2, \dots, l$$

Vidíme, že prvá vrstva má na vstupe iba vstupný vektor, všetky ostatné vrstvy používajú výstup predchádzajúcej vrstvy. BN(.) zodpovedá normalizácii dávok, aktivačná funkcia je LeakyReLU. Vysvetlíme ešte, čomu zodpovedá rôznorodosť(.).

Pri generovaní sa môže stať, že dôjde k tomu, že generátor začne generovať takmer rovnaké dáta pre celú dávku, čím sa snaží oklamať diskriminátor. Diskriminátor sa ale naučí, že tieto dáta sú umelé, preto núti generátor, aby sa naučil nejaké nové. Ten však opäť vyprodukuje skupinu rovnakých dát, teda nesplní požiadavku, že má generovať rôzne dáta z daného rozloženia. Tento jav sa nazýva Helvetica scenario a predchádza sa

mu tým, že sa sleduje rozdielnosť dát v jednej skupinke. Rôznorodosť(.) je teda vektor, ktorý určuje vzdialenosť vzorky od ostatných vzoriek v rámci dávky.

Stratová funkcia: Pri trénovaní modelu sa použil optimalizátor Adam. Kvôli zlepšeniu modelu sa k stratovej funkcii pridáva aj KL divergencia[1] pre vektor $\boldsymbol{d}$ a $\boldsymbol{u}$ a dostávame tak nasledujúce vzťahy:

Pre generátor máme stratovú funkciu definovanú ako:

$$\mathcal{L}_G = -\mathbb{E}_{z\sim\mathcal{N}(0,1)}\log D\big(G(z)\big) + \sum_{i=1}^{n_c} \mathrm{KL}(u_i', u_i) + \sum_{i=1}^{n_d} \mathrm{KL}(\boldsymbol{d}_i', \boldsymbol{d}_i)$$

Pre diskriminátor :

$$\mathcal{L}_{\mathcal{D}} = -\mathbb{E}_{v_{1:n_c}, u_{1:n_c}, \boldsymbol{d}_{1:n_d}\sim\mathbb{P}(T)}\log D\big(v_{1:n_c}, u_{1:n_c}, \boldsymbol{d}_{1:n_d}\big) + \mathbb{E}_{z\sim\mathcal{N}(0,1)}\log D\big(G(z)\big)$$

kde u_i' a d_i' zodpovedajú generovaným dátam a u_i a d_i zodpovedajú reálnym dátam.

4.2 Podmienené tabuľkové GAN siete (CTGAN siete)

Autori modelu TGAN sa spojili s ďalšími výskumníkmi a krátko po publikovaní článku o TGAN vydali článok o CTGAN, v ktorom prezentujú nový spôsob transformácie riadkov.

Kým diskrétne hodnoty vieme jednoducho prezentovať pomocou one-hot vektorov, pri numerických premenných je transformácia dát na vhodný tvar náročnejšia. Dáta môžu pochádzať z ľubovoľného rozdelenia, čo nám sťažuje generovanie dát. Pre numerické premenné je popísaný nasledujúci postup transformácie pre atribút C_i:

I. Pomocou VGM (Variational Gaussian Mixture Model) sa odhadne počet normálnych rozdelení zastúpených v rozdelení premennej C_i , tento počet označíme ako m_i.
II. Pre každú hodnotu $c_{i,j}$ atribútu C_i a každé nájdené normálne rozdelenie určíme pravdepodobnosť, že hodnota pochádza z daného rozdelenia, takto pre každú hodnotu máme m_i pravdepodobností tvoriacich vektor.
III. Z tohto vektora vyberieme niektoré náhodné rozdelenie (s danými pravdepodobnosťami). Povedzme, že sme vybrali k-te rozdelenie a definujeme

[1] KL divergencia zodpovedá Kullback-Leibler divergencii, ktorá určuje vzdialenosť medzi rozdeleniami na vstupe.

$\alpha_{i,j} = \frac{c_{i,j} - \mu_k}{4\sigma_k}$ a $\beta_{i,j} = [0, \dots, 1, \dots, 0]$, tento vektor má m_i prvkov s 1-kou na k-tom mieste a všade inde má nuly.

Takto získame reprezentáciu numerickej premennej pomocou hodnoty α a one-hot vektoru β.

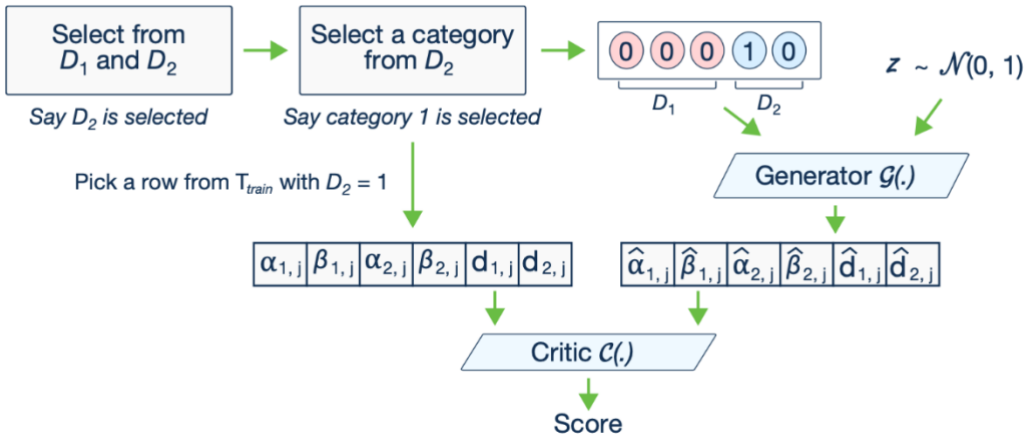
V tomto článku sa autori snažili vyriešiť problém generovania nevyvážených kategoriálnych premenných. V reálnych dátach sa často stretávame s nevyváženými kategoriálnymi premennými, čo pri vytváraní modelu môže skresliť výsledok. Ak totiž vyberáme tréningové dáta náhodne, tak sa môže ľahko stať, že vo vzorke nebudeme mať obsiahnuté všetky možné kategórie, a teda model sa ich nenaučí generovať. Z toho by potom vyplývalo, že generátor negeneruje vzorky z rovnakého rozdelenia, aké mali pôvodné dáta. Čiže by sme nesplnili jednu z dvoch podmienok, ktoré by mali tieto modely splniť. Túto podmienku vieme vyjadriť nasledovne: uvážme i-tu kategoriálnu premennú D_i a jej konkrétnu hodnotu k. Výstup generátora si označme ako $\hat{\boldsymbol{r}}$. Tieto dáta by mali splniť nasledujúcu podmienku: $\hat{\boldsymbol{r}} \sim P_{\mathcal{G}}(\text{row}|\, D_i = k)$, čo je podmienená pravdepodobnosť a odtiaľ pochádza aj názov modelu. To znamená, že diskriminátor má za úlohu naučiť sa podmienené rozdelenie dát, aby platilo $P_{\mathcal{G}}(\text{row}|D_i = k) = P(\text{row}|D_i = k)$, vďaka čomu vieme skonštruovať pôvodné rozdelenie dát vzťahom: $P(\text{row}) = \sum_{k \in D_i} P_{\mathcal{G}}(\text{row}| = k)P(D_i = k)$.

Aby sme toto mohli urobiť, potrebujeme reprezentáciu podmienok, teda potrebujeme vyjadriť vzťahy tvaru $D_i = k$.

Uvažujme kategoriálne premenné $D_1, \dots, D_n$ so všetkými hodnotami, ktoré môžu nadobudnúť. Vytvoríme nulový vektor pre každý atribút tak, aby dimenzia každého vektora zodpovedala počtu možných hodnôt atribútu. Ak chceme vyjadriť vzťah $D_i = k$, tak vo vektore pre atribút D_i na k-tej pozícii zapíšeme 1, čím indikujeme, že dané pozorovanie má túto hodnotu. Tieto vektory nakoniec spojíme, čím dostaneme jeden vektor popisujúci podmienku. Ukážme si tento postup na konkrétnom príklade.

Predstavme si, že máme 2 kategoriálne premenné $D_1 = \{1,2,3\}$ a $D_2 = \{1,2\}$ a chceme vyjadriť $D_2 = 1$. Pre atribút D_1 máme vektor $[0, 0, 0]$ a pre atribút D_2 máme vektor $[1, 0]$, ktorých spojením získame vektor, ktorý sa v článku označuje ako $cond = [0, 0, 0, 1, 0]$ a nazýva sa *mask* vektor.

V článku sa ďalej popisuje postup nazývaný training-by-sampling, ktorého kroky popíšeme a ukážeme na príklade.



Obrázok 17 Príklad k training-by-sampling

Daný príklad pracuje s tabuľkou, ktorá obsahuje dve numerické a dve kategoriálne (D_1, D_2) premenné. Vidíme to z vektorov, ktoré vstupujú do diskriminátora (na obrázku sa nazýva Critic).

Training-by-sampling:

I. pre každú kategoriálnu premennú vytvoríme nulový vektor popísaný vyššie;
II. náhodne vyberieme kategoriálnu premennú D_i. V príklade sa vybrala premenná D_2;
III. vytvoríme distribučnú funkciu pre vybranú kategoriálnu premennú tak, že hodnota distribučnej funkcie bude pre každú hodnotu rovná logaritmu frekvencie výskytu danej hodnoty;
IV. na základe distribučnej hodnoty náhodne zvolíme hodnotu pre vybraný atribút. V príklade sa zvolila hodnota 1;
V. máme podmienku, tú vyjadríme spôsobom popísaným vyššie. Takto vznikne vektor v príklade, ktorý má na štvrtej pozícii hodnotu 1 a inde 0.

Vzniknutý vektor následne spolu s náhodným vstupom vytvára vstup pre generátor.

Generátor aj diskriminátor majú podobnú štruktúru ako v prípade TGAN. Ako už bolo popísané, generátor má na vstupe vektor cond spolu s náhodným vektorom. V tomto prípade máme iba dve skryté vrstvy. Následne vytvárame výstup podľa toho, aký typ premennej generujeme (podobne ako pri TGAN).

Štruktúra generátora:

$$h_0 = z \oplus \text{cond}$$

$$h_1 = h_0 \oplus \text{ReLU}(\text{BN}(\text{FC}_{|\text{cond}|+|z|\to 256}(h_0)))$$

$$h_2 = h_1 \oplus \text{ReLU}(\text{BN}(\text{FC}_{|\text{cond}|+|z|+256\to 256}(h_1)))$$

$$\hat{\alpha}_i = \tanh\left(\text{FC}_{|\text{cond}|+|z|+512\to 1}(h_2)\right) \qquad 1 \le i \le N_c$$

$$\hat{\beta}_i = \text{gumbel}_{0.2}(FC_{|\text{cond}|+|z|+512\to m_i}(h_2)) \qquad 1 \le i \le N_c$$

$$\widehat{\boldsymbol{d}}_i = \text{gumbel}_{0.2}(FC_{|\text{cond}|+|z|+512\to |D_i|}(h_2)) \qquad 1 \le i \le N_c$$

FC je lineárna transformácia, ktorá transformuje u-dimenzionálny vstup na v-dimenzionálny výstup a gumbel je aktivačná funkcia Gumbel softmax.

Štruktúra diskriminátora:

$$h_0 = r_1 \oplus \ldots \oplus r_{10} \oplus \text{cond}_1 \oplus \ldots \oplus \text{cond}_{10}$$

$$h_1 = \text{drop}(\text{leaky}_{0.2}(\text{FC}_{10|r|+10|\text{cond}|\to 256}(h_0)))$$

$$h_2 = \text{drop}(\text{leaky}_{0.2}(\text{FC}_{256\to 256}(h_1)))$$

$$\mathcal{C}(\cdot) = \text{FC}_{256\to 1}(h_2)$$

Ako optimalizátor sa opäť používa Adam s učiacim pomerom $2 \cdot 10^{-4}$.

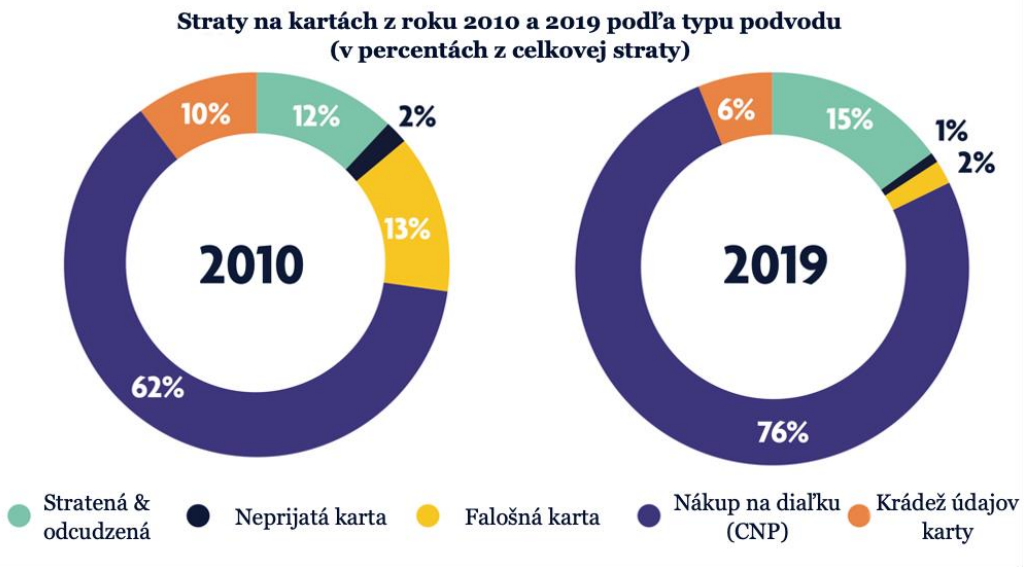
5 Dátová sada

V tejto kapitole si predstavíme dátovú sadu, ktorá je použitá v diplomovej práci. Je zameraná na odhaľovanie podvodníkov v bankových transakciách.

5.1 Odhaľovanie podvodníkov v bankových transakciách

Podvodné transakcie predstavujú každý rok vysoké straty ako pre klientov, tak aj pre obchody, ktoré sú zodpovedné za straty. Väčšina podvodov sa stane cez CNP transakcie. CNP transakcie (card-not-present transakcie) sú také platby, počas ktorých nie je fyzicky prítomná karta a vlastník karty. Typické CNP platby sú online platby alebo automaticky opakované mesačné platby. Pomer CNP platieb k ostatným druhom platieb v posledných rokoch neustále rastie, na obrázku 18 môžeme vidieť porovnanie z rokov 2010 a 2019 [16].

Rovnako každý rok rastú aj škody spôsobené podvodníkmi. Motivácia odhaliť podvodníkov pri platbách je vysoká, pretože za všetky straty sú zodpovedné obchody, do ktorých išla platba. Štatistiky z roku 2020 hovoria, že každý jeden dolár v podvodnej transakcii stál obchodníkov v priemere 3,36 dolárov. Je to spôsobené tým, že obchody musia vrátiť peniaze, ale aj prešetriť všetky transakcie, takže aj samotné analyzovanie a vyhodnotenie nahlásených transakcií ich niečo stojí.



Obrázok 18 Štatistiky podvodov za rok 2010 a 2019

Banky a obchodníci sa snažia rôznymi spôsobmi chrániť pred podvodníkmi. Bežne sa používa určenie hodnoty skóre rizika (risk score) v reálnom čase pre transakcie, na základe ktorého sa vyhodnotí, či je transakcia bezpečná. Rozvojom mobilných zariadení sa pridalo aj overenie majiteľa buď cez odtlačok prsta, overenie hlasu, alebo rozoznávanie tvárí. Ďalšia možnosť na ochranu pred podvodníkmi je použiť metódy strojového učenia, čo je aj cieľom tejto diplomovej práce.

5.2 Kaggle súťaž

V roku 2019 bola na stránke kaggle.com vypísaná súťaž [12] na odhaľovanie podvodníkov v CNP transakciách. Súťaž bola vypísaná spoločnosťou IEEE Computational Intelligence Society. Dáta o transakciách poskytla spoločnosť Vesta, ktorá sa zaoberá odhaľovaním podvodníkov v transakciách. Súťaž trvala pol roka, víťazom sa podarila dosiahnuť presnosť 94,6 %.

Vesta poskytla do súťaže dve dátové sady, ktoré majú spolu 434 atribútov. Medzi atribútmi sa nachádzajú informácie o kartách, adresách, vzdialenostiach a rôzne iné. Kvôli citlivosti dát nepoznáme význam všetkých atribútov, vieme ale, že niektoré atribúty vytvorila Vesta, pretože ich považovala za dôležité. Je podstatné poznamenať, že pri všetkých transakciách máme poznačené, či išlo o podvodnú transakciu alebo nie. V praxi máme často pozorovania bez toho, aby sme vedeli, ktoré sa považujú za anomálie a ktoré nie. Vtedy je ešte náročnejšie sa ich nejakým spôsobom naučiť.

Popísanú dátovú sadu sme použili pre účely tejto práce.

6 Návrh modelu a výsledky

V šiestej kapitole prezentujeme návrh modelu detekcie anomálií na zvolenej dátovej sade. Hlavným prínosom práce je generovanie umelých anomálií pomocou GAN sietí. Jedným z cieľov práce je porovnať výsledky na pôvodnej tréningovej sade bez umelých podvodníkov s výsledkami rovnakých algoritmov aplikovaných na tréningovú sadu doplnenú o umelých podvodníkov.

6.1 Návrh modelu

Postup, ktorý v tejto práci navrhujeme a aplikujeme na reálne dáta je možné zhrnúť do nasledujúcich bodov:

I. úprava dát,
II. výber atribútov,
III. generovanie umelých dát,
IV. vytvorenie novej tréningovej množiny,
V. klasifikácia dát (podvodná transakcia/platná transakcia).

6.1.1 Úprava dát

Pri práci s dátami je väčšinou prvým krokom úprava dát, preto nemohla chýbať ani v tomto prípade. Túto časť sme nerobili samostatne, v rámci súťaže boli totiž urobené viaceré predspracovania dát, ktorým predchádzali netriviálne úvahy a pozorovania. V popise súťaže navyše neboli kvalitne popísané atribúty dátovej sady, autori súťaže ozrejmovali niektoré fakty len po položení otázky v diskusiách, pričom kvôli citlivosti dát pri niektorých atribútoch ani neprezradili, čo predstavujú. Tento fakt viedol k zvýšenej náročnosti kvalitného predspracovania dát. Z uvedených dôvodov sme použili postup predspracovania, ktorý používal víťazný tím tejto súťaže.

6.1.2 Výber atribútov

Vybraná dátová sada obsahuje okolo 400 atribútov a stovky tisíc riadkov, preto by si práca s ňou vyžadovala veľa času. Kvôli zmenšeniu veľkosti dát sme sa rozhodli do postupu zakomponovať výber atribútov. Použili sme dva algoritmy na výber najdôležitejších atribútov, aby sme výsledky dokázali porovnať aj podľa toho, s ktorými atribútmi sme pracovali. Pre tento účel sme použili algoritmus SelectKBest a ExtraTreesClassifier.

6.1.2.1 SelectKBest

Prvá metóda, ktorú sme použili na výber atribútov je SelectKBest z knižnice scikit-learn. Táto metóda sa pozerá na vzťah jednotlivých atribútov a cieľovej premennej pri klasifikačnom probléme. Vyberá tie atribúty, ktoré najviac súvisia s cieľovou premennou. Spôsob, akým vyberá „najlepšie“ atribúty si popíšeme v tejto podkapitole.

Metóda vyžaduje 3 základné vstupy, a to:

X – množinu atribútov, z ktorých chceme vybrať tie „najlepšie“;

y – cieľová premenná, podľa ktorej klasifikujeme dáta do tried;

k – počet atribútov, ktoré chceme použiť.

SelectKBest počíta dve hodnoty, konkrétne *F*-hodnotu a *p*-hodnotu, a to pre každú dvojicu atribút-cieľová premenná. Na základe nich usporiada jednotlivé atribúty a následne metóda vráti prvých *k* atribútov. Čím vyššie je *F*-skóre, tým dôležitejší je uvažovaný atribút pri klasifikácii podľa cieľovej premennej.

Ak uvažujeme klasifikáciu do dvoch tried (+ a –), tak *F*-skóre i-teho atribútu je v [11] definované nasledovným vzťahom:

$$F(i) \equiv \frac{\left(\bar{\boldsymbol{x}}_i^{(+)} - \bar{\boldsymbol{x}}_i\right)^2 + \left(\bar{\boldsymbol{x}}_i^{(-)} - \bar{\boldsymbol{x}}_i\right)^2}{\frac{1}{n_+ - 1}\sum_{k=1}^{n_+}\left(x_{k,i}^{(+)} - \bar{\boldsymbol{x}}_i^{(+)}\right)^2 + \frac{1}{n_- - 1}\sum_{k=1}^{n_-}\left(x_{k,i}^{(-)} - \bar{\boldsymbol{x}}_i^{(-)}\right)^2},$$

kde

$\bar{\boldsymbol{x}}_i^{(+)}$ je priemer hodnôt i-teho atribútu v triede +,

$\bar{\boldsymbol{x}}_i^{(-)}$ je priemer hodnôt i-teho atribútu v triede – a

$\bar{x}_i$ je celkový priemer hodnôt atribútu.

Hodnota $x_{k,i}^{(+)}$ vyjadruje hodnotu i-teho atribútu pre k-ty príklad z triedy +, analogicky definujeme hodnotu $x_{k,i}^{(-)}$.

Ako môžeme vidieť, čitateľ obsahuje rozptyly jednotlivých tried (+ a –) v celom i-tom atribúte. Menovateľ sa pozerá na rozptyl v jednotlivých triedach. Čím menší je rozptyl vnútri tried, tým je hodnota menovateľa menšia, a teda hodnota zlomku väčšia. Preto hľadáme vysoké *F*-hodnoty.

Nevýhodou tohto prístupu je, že sa na jednotlivé atribúty pozerá zvlášť. Teda týmto spôsobom neodhalíme skupiny atribútov, ktoré spolu dokážu dobre klasifikovať dáta. Môže sa totiž stať, že dvojica atribútov má zvlášť nízke *F*-skóre, ale keby sme ich spojili, získali by sme výrazne lepší výsledok. Práve preto sa nemôžeme spoľahnúť na to, že týmto spôsobom získame najlepšiu možnú k-ticu atribútov, ale vieme, že aj napriek tomu bude dostatočne dobrá.

6.1.2.2 ExtraTreesClassifier

Metóda ExtraTreesClassifier je implementovaná tiež v knižnici scikit-learn a používa sa nielen na výber atribútov, ale najmä na klasifikáciu. Táto metóda je popísaná v [7]. Podobá sa na metódu náhodného lesu, ktorá bola popísaná v kapitole 2. Najväčšími rozdielmi sú:

- náhodný les nepracoval naraz s celou tréningovou vzorkou, iba s jej časťou, kým ExtraTreesClassifier pracuje s celou tréningovou vzorkou;
- pre náhodný výber atribútov v náhodnom lese sa pre každý atribút určil deliaci bod, ktorý zabezpečil najlepšiu možnú homogenitu dát. Následne sa zvolil atribút, pre ktorý bolo delenie najlepšie. V tomto prípade sa deliace body vyberajú náhodne, teda sa zrýchľuje výpočet.

Ako môžeme vidieť, v týchto algoritmoch je zabudované určenie dôležitosti jednotlivých atribútov vzhľadom na klasifikáciu dát, práve preto ich môžeme použiť, ak chceme určiť najdôležitejšie atribúty.

6.1.3 Generovanie umelých dát

Pomocou GAN sietí predstavených v kapitole 4 vytvoríme umelú dátovú sadu na základe reálnej. V tejto časti sme použili siete tabGAN a CTGAN s rôznymi

nastaveniami parametrov. Pri generovaní umelých dát môžeme skúsiť dva základné prístupy:

I. generovanie iba podvodných transakcií z dátovej sady, ktorá obsahuje iba podvodné transakcie;
II. generovanie celej dátovej sady s následným filtrovaním podvodných transakcií z výslednej dátovej sady.

V práci sme využívali najmä prístup II., keďže v prvom prípade strácame celkovú informáciu o dátach. To znamená, že prichádzame o informáciu, ktorá robí anomálie anomáliami medzi ostatnými dátami. Na druhej strane kvôli zriedkavému výskytu dát sa model nemusí naučiť tieto dáta správne, keďže predstavujú iba zlomok z celého súboru a do celkovej chyby modelu prispievajú iba minimálne. Môže sa nám stať, že model bude správne generovať nenahlásené transakcie, ale pri generovaní podvodných nebude dostatočne úspešný aj napriek tomu, že celkový výsledok modelu je dobrý.

6.1.4 Vytvorenie novej tréningovej množiny

Pri tejto časti je dôležité podotknúť, že nové transakcie pridávame iba do tréningovej množiny, testovacia sada ostáva nezmenená. Je to spôsobené tým, že chceme vytvoriť model, ktorý dokáže spoľahlivo odhaliť podvodné transakcie, ale chceme ho aplikovať na reálne dáta. Pri vytvorení modelu teda môžeme učeniu pomôcť tým, že pridáme navyše nejaké umelo vygenerované dáta, ale presnosť klasifikácie už testujeme na reálnych dátach.

Ďalšou zaujímavou otázkou je, ako veľa podvodných transakcií chceme doplniť do tréningovej množiny, teda v akom pomere chceme mať podvodné a platné transakcie.

6.1.5 Klasifikácia dát

Po vytvorení modifikovanej tréningovej množiny ostáva už len klasifikácia dát. Dáta klasifikujeme do dvoch tried, delíme ich na podvodné transakcie a platné transakcie. Použili sme algoritmy popísané v kapitole 2, a to náhodný les a XGBoost algoritmus.

7 Výsledky

V tejto kapitole prezentujeme výsledky, ktoré sme získali analýzou troch súborov:

- **top30_ExtraTreesClassifier.csv** – súbor s 30 atribútmi, ktoré boli vybrané na základe algoritmu ExtraTreesClassifier;
- **top30_SelectKBest_chi.csv** – súbor s 30 atribútmi, ktoré boli vybrané na základe algoritmu SelectKBest použitím funkcie chi2;
- **top30_SelectKBest_f.csv** – súbor s 30 atribútmi podľa SelectKBest použitím funkcie f_classif.

7.1.1 Základné výsledky

V každom z uvedených troch súborov je 590 540 riadkov. Z nich 20 663 je podvodných (čiže ich považujeme za anomálie), čo predstavuje približne 3,5 % anomálií v našich dátach. Aby sme mali s čím porovnávať výsledky na dátových sadách doplnených o umelých podvodníkov, pozreli sme sa na výsledky na pôvodných dátach. Pre všetky tri uvedené dátové sady sme získali výsledky klasifikácie pomocou náhodného lesa a algoritmu XGBoost. Výsledky uvádzame v nasledujúcej tabuľke (červeným sú zvýraznené najlepšie výsledky pre daný algoritmus):

Názov datasetu	XGBoost	Random Forest
top30_ExtraTreesClassifier.csv	0.9617	0.976883
top30_SelectKBest_chi.csv	0.9554	0.973255
top30_SelectKBest_f.csv	0.95301	0.971028

Tab. 1 Základné výsledky bez použitia umelých podvodníkov

Pre algoritmus XGBoost sme použili nasledujúce nastavenia:

- n_estimators = 2000,
- max_depth = 12,
- learning_rate = 0.02.

Otestovali sme rôzne nastavenia parametrov, najlepšie výsledky sa dosiahli pre tieto hodnoty.

Pre získanie výsledkov pomocou algoritmu náhodných lesov sme použili dopredu nastavené hodnoty parametrov, keďže ich zmenou sa nepodarilo zlepšiť výsledky. Na vyhodnotenie modelu sme použili metriku presnosť (accuracy), ktorá vyjadruje podiel správne klasifikovaných vzoriek zo všetkých klasifikovaných vzoriek, teda sa dá vyjadriť vzťahom:

$$\text{presnosť} = \frac{\text{správne klasifikované vzorky}}{\text{všetky klasifikované vzorky}}.$$

Túto metriku sme zvolili najmä z toho dôvodu, že sa ľahko interpretuje a porovnáva.

Pre získanie výsledkov pomocou algoritmu XGBoost sme použili metriku priemernej absolútnej chyby (MAE, Mean Absolute Error), ktorá sa dá vyjadriť vzťahom:

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

kde n je celkový počet vzoriek, x_i je reálna hodnota výstupnej premennej a y_i je hodnota, ktorú pre danú vzorku určil model, teda predikcia modelu. Vzhľadom k tomu, že v našich dátach nadobúda cieľová premenná hodnoty 0 alebo 1, tak vieme, že sčítance v čitateli budú mať tiež hodnoty 0 (ak máme správnu predikciu) alebo 1 (ak model nesprávne určil hodnotu výstupnej premennej). Vďaka tomu bude hodnota MAE zodpovedať podielu nesprávne klasifikovaných vzoriek zo všetkých klasifikovaných vzoriek, a teda bude doplnkom k vyššie popísanej presnosti. Platí teda nasledujúci vzťah: $\text{presnosť} = 1 - \text{MAE}$. Využijúc tento vzťah, aby sme tabuľky urobili čitateľnejšími, sme v stĺpci XGBoost uviedli hodnotu $1 - \text{MAE}$. Takto sa nám budú ľahšie porovnávať výsledky jednotlivých algoritmov.

Ako môžeme vidieť, výber atribútov pomocou ExtraTreesClassifier viedol k lepším výsledkom ako v prípade SelectKBest bez ohľadu na to, ktorý algoritmus sme použili na klasifikáciu. Tiež vidíme, že bez ohľadu na použitú dátovú sadu, algoritmus náhodný les mal lepšie výsledky ako algoritmus XGBoost. Výsledky poukazujú na to, akú presnosť vieme dosiahnuť bez použitia umelých podvodníkov, teda za predpokladu trénovania výlučne s reálnymi podvodníkmi. S týmito hodnotami teda budeme porovnávať naše nasledujúce výsledky, v ktorých sa pozrieme na to, ako sa zmenia hodnoty presnosti po doplnení umelých podvodníkov do tréningovej sady.

Na generovanie umelých podvodníkov sme použili dva algoritmy, a to CTGAN a tabGAN. Oba algoritmy sa mierne líšia svojimi výstupmi, preto v krátkosti opíšeme, ako vyzerá generovanie umelých dát v jednotlivých prípadoch.

7.1.2 Výsledky CTGAN

Najprv uvedieme algoritmus CTGAN. Na vstupe dostáva pôvodnú dátovú sadu a na výstupe nám dáva model, ktorý dokáže generovať dáta podobné tým, na ktorých bol model trénovaný. Po získaní modelu dokážeme generovať umelé dáta ľubovoľnej veľkosti, teda vieme určiť veľkosť vzorky pomocou parametra. Je potrebné spomenúť, že získanie modelu trvá dlhší čas (rádovo niekoľko hodín), ale práca so samotným modelom je už veľmi rýchla. Za pár sekúnd máme k dispozícii umelé dátové sady ľubovoľnej veľkosti.

Najprv sme vygenerovali výsledky siete so základnými nastaveniami a skúmali, ako sa zmení presnosť predikcie pri doplnení umelých podvodníkov z modelov so základnými nastaveniami (tabuľka 2).

Názov datasetu	XGBoost	Random Forest
top30_ExtraTreesClassifier.csv	0.94747	0.976416
top30_SelectKBest_chi.csv	0.94136	0.973055
top30_SelectKBest_f.csv	0.93744	0.970699

Tab. 2 Výsledky po použití umelých podvodníkov z CTGAN

Všetky základné nastavenia pre CTGAN vieme zistiť v dokumentácii, preto uvádzame iba niektoré:

- epochs = 300,
- batch_size = 500,
- generator_lr = discriminator_lr = 0.0002 (lr = learning rate – učiaci pomer) – keďže CTGAN (ako všetky GAN siete) pozostáva z dvoch neurónových sietí, potrebujeme nastaviť učiaci pomer pre každú zvlášť, v tomto prípade však bol zvolený na rovnakú veľkosť v oboch prípadoch.

Ako bolo spomenuté, pri generovaní umelých dát z modelu dokážeme určiť počet vzoriek, ktoré chceme, aby model generoval. Skúšali sme rôzne počty umelých podvodníkov, najlepšie výsledky sa dosiahli v prípade počtu okolo 8 000 – 10 000 podvodníkov (pre porovnanie, pôvodná dátová sada obsahovala 20 663 podvodníkov).

Ak sa pozrieme na výsledky, môžeme vidieť, že opäť boli dosiahnuté najlepšie výsledky pre dátovú sadu top30_ExtraTreesClassifier.csv a pre algoritmus náhodný les. Ak však tieto výsledky porovnáme s predchádzajúcimi, môžeme vidieť, že sa nám pri základných nastaveniach nepodarilo zlepšiť presnosť modelu. V prípade náhodného lesu je zhoršenie veľmi mierne, avšak pri XGBoost modeli presnosť modelu výrazne klesla. Ukazuje sa teda, že použitie umelých podvodníkov zo CTGAN modelu so základnými nastaveniami nie je pre náš problém výhodné.

7.1.3 Výsledky tabGAN

V prípade siete tabGAN máme tri vstupy a dva výstupy.

Vstupy pre algoritmus:

- train – tréningová dátová sada bez cieľovej premennej (v našom prípade bez stĺpca isFraud);
- target – cieľová premenná (v našom prípade stĺpec isFraud);
- test – testovacia dátová sada bez cieľovej premennej.

Výstupy algoritmu:

- new_train – umelá dátová sada bez cieľovej premennej;
- new target – cieľová premenná umelej dátovej sady.

V tomto prípade na výstupe nemáme model, ale umelú dátovú sadu. Počet umelých podvodníkov je teda fixný vo výstupe, avšak pomocou nastavenia parametra gen_x_times vieme vopred určiť, koľko nových dát chceme. Takto sme sa dostali k počtu okolo 14 000 – 16 000 umelých podvodníkov.

Výsledky, ktoré sme získali pri základných nastaveniach parametrov sú zobrazené v tabuľke 3.

Názov datasetu	XGBoost	Random Forest
top30_ExtraTreesClassifier.csv	0.92518	**0.977843**
top30_SelectKBest_chi.csv	0.91479	0.975354
top30_SelectKBest_f.csv	0.91224	0.972126

Tab. 3 Výsledky po použití umelých podvodníkov z tabGAN

Môžeme vidieť, že opäť boli dosiahnuté najlepšie výsledky na dátovej sade top30_ExtraTreesClassifier.csv a výrazne lepšie výsledky dosiahol opätovne algoritmus náhodný les. Ak tieto výsledky porovnáme s predchádzajúcimi tabuľkami, môžeme si všimnúť, že algoritmus XGBoost dosiahol oveľa horšie výsledky ako predtým. V prípade algoritmu náhodných lesov však získaná presnosť presiahla základ, ktorý sme si určili v prvej tabuľke. Zdá sa teda, že najvhodnejšia cesta je pracovať s dátovou sadou top30_ExtraTreesClassifier.csv, generovať nových podvodníkov pomocou algoritmu tabGAN a zamerať sa na algoritmus náhodný les.

7.1.4 Hlavné experimenty

Na základe pozorovaní z predchádzajúcich experimentov sme sa zamerali už len na jednu dátovú sadu (top30_ExtraTreesClassifier.csv) a jeden spôsob generovania umelých podvodníkov (tabGAN). V tejto finálnej fáze sme sa snažili ešte vylepšiť najvyššiu presnosť, ktorú sa nám podarilo získať v základnej fáze skúmania (hodnota presnosti 0.977843 pre náhodný les).

Vytvorili sme tri ďalšie umelé dátové sady pomocou algoritmu tabGAN, z ktorých všetky prevýšili predchádzajúce výsledky pri algoritme náhodný les. V krátkosti popíšeme, aké nastavenia sme použili pri generovaní jednotlivých dátových sád:

- tabGAN_ETC_1.csv – všetky parametre nastavené podľa nastavení v príklade na (https://github.com/Diyago/GAN-for-tabular-data) (prípad tretieho modelu so všetkými parametrami);
- tabGAN_ETC_2.csv – rovnaké nastavenia ako v predchádzajúcom prípade, jediná zmena bola gen_x_times = 2 a only_generated_data=True. Zvýšili sme teda počet generovaných dát a ponechali sme si iba umelé dáta;
- tabGAN_ETC_3.csv – nastavili sa iba nasledujúce parametre pre siete: gan_params = {"batch_size": 500, "patience": 25, "epochs" : 750}.

Výsledky, ktoré sme získali z týchto troch modelov, sú znázornené v tabuľke 4.

Názov datasetu	XGBoost	Random Forest
tabGAN_ETC_1.csv	0.92505	0.977853
tabGAN_ETC_2.csv	0.92521	0.977889
tabGAN_ETC_3.csv	0.92502	**0.978299**

Tab. 4 Finálne výsledky

Ako sa dalo očakávať, algoritmus XGBoost nedosiahol výrazné zlepšenie, aj keď sa mu aspoň mierne podarilo zlepšiť predchádzajúci najlepší výsledok. Zaujímavejšie sú však výsledky algoritmu náhodný les, keďže sme sa sústredili najmä na zlepšenie výsledkov v tomto prípade. Ak sa pozrieme na výsledok tretieho pokusu, tak môžeme vidieť najlepší výsledok, aký sa nám podarilo získať. Na prvý pohľad sa síce môže zdať dosiahnuté zlepšenie ako minimálne, avšak treba si uvedomiť, že vo svete dátovej analýzy dokáže byť obrovským úspechom aj zlepšenie o jednu tisícinu. Nezabudnime, že problematika, ku ktorej tieto dáta patria je z bankového prostredia, v ktorom sa rozprávame o obrovskom objeme peňazí. Vezmime si štatistiky Európskej centrálnej banky z roku 2019, keď hodnota podvodných transakcií tvorila 1,5 miliardy eur [8]. Aj malé zlepšenie dokáže ušetriť veľké množstvo peňazí, čo je hlavným cieľom pri odhaľovaní podvodov.

Ak sa pozrieme na prvú tabuľku, môžeme vidieť, že presnosť, ktorú sme chceli zlepšiť, mala hodnotu 0.976883. Presnosť, ku ktorej sme sa dopracovali je 0.978299. To znamená, že sme zlepšili presnosť o viac ako jednu tisícinu a ako sme už spomínali, tento výsledok sa dá považovať za výrazné zlepšenie. Na serióznych medzinárodných súťažiach často o víťazoch rozhodujú ešte menšie rozdiely.

8 Diskusia a súhrn

V prechádzajúcej kapitole sme popísali výsledky, ktoré sme získali v diplomovej práci. Podarilo sa nám ukázať, že v prípade detekcie podvodníkov v bankových dátach má použitie GAN sietí zmysel a dokáže nám pomôcť v lepšom odhaľovaní podvodníkov. Aplikovali sme tri metódy na redukciu atribútov, z ktorých dosiahla najlepšie výsledky metóda ExtraTreesClassifier. Vo všetkých experimentoch sme dosiahli najlepšie výsledky práve na atribútoch vybraných týmto spôsobom. Pri generovaní umelých podvodníkov sme experimentovali s dvomi rôznymi GAN sieťami. V našom prípade dosiahla lepšie výsledky metóda tabGAN, preto sme väčšinu ďalších experimentov zamerali na túto sieť. Výslednú klasifikáciu podvodníkov sme porovnávali pomocou dvoch algoritmov, z ktorých dosiahol najlepšiu presnosť náhodný les.

Pomocou postupného zlepšovania nastavení pre GAN siete sa nám podarilo vytvoriť modifikovanú tréningovú množinu, vďaka ktorej sa presnosť náhodného lesu zlepšila oproti pôvodnej hodnote. Úspešne sme teda ukázali, že GAN siete vedia byť užitočné aj v prípade detekcie anomálií, konkrétne detekcie podvodníkov na bankových dátach. Nemáme vedomosť o tom, že by tento alebo podobný postup bol na daných dátach v literatúre testovaný. Preto ide o jedinečný prístup, ktorý sa osvedčil ako vhodný.

Je potrebné podotknúť, že v oblasti analýzy dát sa často kombinujú rôzne prístupy, keďže každý z nich vie potenciálne zlepšiť naše výsledky. Cieľom tejto práce bolo zoznámiť sa s GAN sieťami a použiť ich na problém detekcie anomálií. Je však zrejmé, že naša dosiahnutá presnosť by sa pravdepodobne dala vylepšiť. Uvedieme niekoľko možností pre ďalšiu prácu v tejto oblasti:

- zväčšenie počtu atribútov (my sme pracovali s 30 atribútmi, ale výsledky je možné otestovať aj pri väčšom počte atribútov);
- niektoré atribúty mali špeciálne rozdelenie dát, ktoré by sa mohli prípadne vynechať (v dátach sa nachádzajú stĺpce, v ktorých má viac ako 75 % pozorovaní hodnotu –1, ale aj niekoľko pozorovaní s hodnotou rádovo okolo 10-tisíc). Vyskúšali sme to v jednom prípade, ale neviedlo to k zlepšeniu výsledkov (práve naopak), preto sme neskôr tento smer zavrhli. V každom prípade by ale mohlo byť užitočné tieto atribúty podrobnejšie skúmať;
- vyskúšať iné algoritmy vhodné pre klasifikáciu;

- vyskúšať iné GAN siete (v tomto bode je potrebné poznamenať, že GAN siete vhodné pre tabuľkové dáta sú relatívne nové, nie je ich veľa, teda je potrebné sledovať aj vývoj nových GAN sietí);
- spojiť viacero algoritmov na hľadanie anomálií v dátach. Výsledné modely sa často skladajú z viacerých podmodelov, keďže sa môže stať, že na jednej podmnožine dát má jedna metóda najlepšie výsledky, kým na inej podmnožine dát má lepšie výsledky iná metóda. Ich spájanie teda môže viesť k ešte lepším výsledkom. Ak sa pozrieme na víťazov Kaggle súťaže, z ktorej sme čerpali dáta, tak si môžeme všimnúť, že aj oni používali tri rôzne algoritmy na detekciu anomálií práve z toho dôvodu, že všetky mali dobrý výkon na rôznych vstupoch.

Na záver by sme ešte dodali, že dosiahnuté výsledky nedokážeme porovnať s inými výsledkami. V súťaži zo stránky kaggle.com totiž existovali dve testovacie množiny, verejná a privátna. Súťažiaci sa snažili získať čo najlepšie výsledky práve na privátnej testovacej množine. My sme tieto množiny nepoužívali, používali sme iba tréningovú množinu a tú sme delili následne na menšiu tréningovú množinu a testovaciu množinu. Keďže k privátnej testovacej množine súťaže nemáme prístup, nedokážeme porovnať nami získané výsledky s víťazom súťaže, ktorému sa podarilo dosiahnuť presnosť okolo 94 %.

Záver

V práci sme predstavili nový spôsob detekcie anomálií na reálnych dátach. Základ postupu tvorilo použitie GAN sietí pre účely generovania umelých anomálií. Popísali sme pojem anomálie, typy anomálií a tiež algoritmy, pomocou ktorých vieme vyhľadávať anomálie. Podstatnú časť práce sme venovali predstaveniu GAN sietí. Popísali sme všeobecnú štruktúru GAN sietí, ktoré boli vytvorené za účelom generovania obrázkov. Predstavili sme nový typ GAN sietí, pomocou ktorých dokážeme generovať tabuľkové dáta, podrobne sme popísali modely TGAN a CTGAN, ktoré sme aplikovali na naše dáta.

Detekcia anomálií zohráva dôležitú úlohu v spracovaní dát, jej využitie nájdeme v rôznych príkladoch z reálneho sveta. Práve preto sme pri výbere dátovej sady dbali na to, aby sme mali reálne dáta, v ktorých by detekcia anomálií prinášala úžitok pre spoločnosť. Odhaľovanie podvodníkov patrí medzi reálne problémy, ktoré potrebujeme riešiť. Preto sa nám použitá dátová sada zdala ako vhodná pre účely tejto práce.

Jadro diplomovej práce tvorili GAN siete a spôsob, akým sme tieto siete použili pri odhaľovaní podvodníkov v bankových transakciách. Vychádzali sme zo základného problému detekcie anomálií, a to z faktu, že anomálie sa vyskytujú iba zriedkavo v dátach, a preto je náročné sa ich učiť. Počet anomálií v dátach sme zvýšili pomocou GAN sietí, ktoré generujú umelé dáta na základe reálnych.

V siedmej kapitole sme prezentovali výsledky práce. Uviedli sme výsledky základných modelov, ktoré nepracovali s umelými anomáliami a porovnávali ich s výsledkami modelov vytvorených na nových tréningových sadách. Porovnali sme rôzne výbery atribútov, rôzne algoritmy na detekciu anomálií a tiež rôzne GAN siete na generovanie umelých anomálií. Ukázali sme, že výberom atribútov pomocou algoritmu ExtraTreesClassifier, generovaním anomálií pomocou tabGAN a následnou klasifikáciou pomocou náhodného lesu dokážeme prekonať základné výsledky, čím sa nám podarilo ukázať, že aplikácia GAN sietí naozaj dokáže zlepšiť presnosť modelu.

GAN siete patria medzi relatívne nové modely v oblasti analýzy dát a umelej inteligencie. Neustále sa vyvíjajú nové, ktoré posúvajú ich možnosti na vyššiu úroveň. GAN siete pre tabuľkové dáta vznikli iba pred pár rokmi, ich výskum je ešte na začiatku, čo potvrdzuje aj nízky počet odborných článkov na túto tému. Napriek tomu sa už v niektorých odborných prácach, ale aj v tejto diplomovej práci preukázala ich sila a užitočnosť v aplikácii na reálne problémy. Je možné predpokladať, že ich postupným

vylepšovaním posunieme ich význam a využitie aj na riešenie mnohých ďalších praktických úloh v rôznych oblastiach nášho života.

Zoznam použitej literatúry

[1] AGGARWAL, C. C., 2015. *Data Mining: The Textbook*. New York: Springer. ISBN 978-3-319-14141-1.

[2] ANDREJKOVÁ, G. a ANTONI, Ľ., 2020. *Strojové učenie*. Univerzita Pavla Jozefa Šafárika v Košiciach. ISBN 978-80-8152-912-2

[3] ARASANIPALAI, A. U., 2019. Generative Adversarial Networks – The Story So Far [online]. Dostupné na internete: https://blog.floydhub.com/gans-story-so-far/#cgan

[4] ASHRAPOV, I., 2020. GANs for tabular data [online]. 2020. Dostupné na internete: https://towardsdatascience.com/review-of-gans-for-tabular-data-a30a2199342

[5] ASHRAPOV, I., 2020. Tabular GANs for uneven distribution. arXiv preprint arXiv:2010.00638.

[6] BROWNLEE, J., 2019. A Gentle Introduction to Generative Adversarial Networks (GANs). Dostupné na internete: https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/

[7] CEBALLOS, F., 2019. An Intuitive Explanation of Random Forest and Extra Trees Classifiers [online]. Dostupné na internete: https://towardsdatascience.com/an-intuitive-explanation-of-random-forest-and-extra-trees-classifiers-8507ac21d54b

[8] EUROPEAN CENTRAL BANK, 2021. Seventh Report on card fraud [online]. Dostupné na internete: https://www.ecb.europa.eu/pub/cardfraud/html/ecb.cardfraudreport202110~cac4c418e8.en.html. ISSN 2315-0033

[9] GOODFELLOW, I., et al. 2014. Generative adversarial nets. *Advances in Neural Information Processing Systems* 27. NuerIPS Proceedings.

[10] CHEN, T. a GUESTRIN, C., 2016. XGBoost: A scalable tree boosting system [online]. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pp. 785-794.

[11] CHEN, Y. W. a LIN, C. J., 2006. Combining SVMs with various feature selection strategies. In: *Feature Extraction*. Springer, Berlin, Heidelberg, p. 315-324.

[12] IEEE – CIS, 2019. Fraud Detection [online]. Dostupné na internete: https://www.kaggle.com/c/ieee-fraud-detection/overview

[13] PARALIČ, J., 2022. Detekcia anomálií. Prednášky z predmetu Objavovanie znalostí. Technická univerzita v Košiciach.

[14] PARR, T – HOWARD, J., 2022. Gradient boosting: Distance to target [online]. Dostupné na internete: https://explained.ai/gradient-boosting/L2-loss.html

[15] SANTOYO, S. A., 2017. Brief Overview of Outlier Detection Techniques [online]. Dostupné na internete: https://towardsdatascience.com/a-brief-overview-of-outlier-detection-techniques-1e0b2c19e561

[16] WOROBEC, K. 2020. Fraud – The Facts 2020. Dostupné na internete: https://www.ukfinance.org.uk/system/files/Fraud-The-Facts-2020-FINAL-ONLINE-11-June.pdf

[17] XU, L., et al. 2019. Modeling tabular data using conditional GAN. *Advances in Neural Information Processing Systems* 32. In: 33rd Conference on Neural Infmormation Processing Systems (NeurIPS 2019), Vancouver, Canada.

[18] XU, L., VEERAMACHANENI, K., 2018. Synthesizing tabular data using generative adversarial networks. arXiv preprint arXiv:1811.11264.

[19] YIU, T., 2019. Understanding Random Forest [online]. Dostupné na internete: https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[20] ZHANG, Z., 2019. Boosting Algorithms Explained [online]. Dostupné na internete: https://towardsdatascience.com/boosting-algorithms-explained-d38f56ef3f30

Prílohy

Príloha A: zdrojové kódy