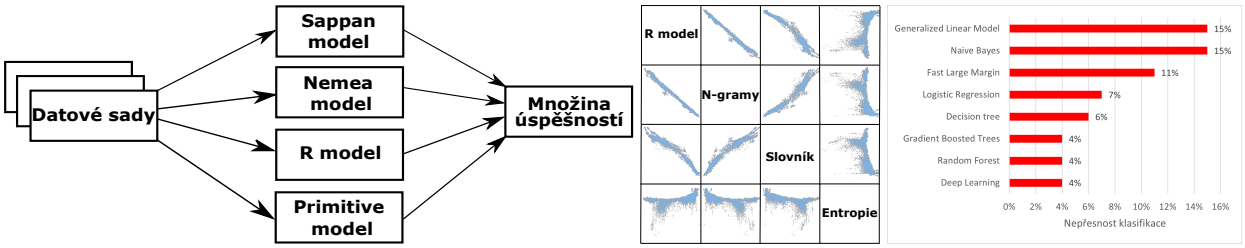
BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2021
http://excel.fit.vutbr.cz

Predikce úspěšnosti datových sad

Jiří Setinský*



Abstrakt

V síti často probíhá komunikace mezi útočníkem a nakaženými počítači. Nelegitimní komunikaci mezi řídícím serverem a botnetem chceme odhalit a filtrovat. Jejich komunikace je založená na generování umělých doménových jmen pomocí DGA algoritmů. Při detekci DGA adres je využito strojového učení. Pro přesnou klasifikaci je třeba vytvářet kvalitní trénovací datové sady. Jeden z přístupů, jak vytvořit datovou sadu je prostřednictvím genetických algoritmů. Při generování datových sad pomocí genetických algoritmů narážíme na problém s časovou náročností na určení kvality datové sady. Při velkém množství vygenerovaných datových sad přichází na řadu jejich otestování klasifikátorem. Cílem článku je najít vhodný způsob jak ohodnotit vygenerovanou datovou sadu bez toho, aniž by bylo potřeba použít klasifikátor pro otestování úspěšnosti. Budeme chtít najít parametry, jenž budou charakterizovat danou datovou sadu. Vyhodnocením parametrů pro více datových sad získáme trénovací datovou sadu, která bude sloužit pro natrénování klasifikátoru, který bude predikovat úspěšnost datové sady. Na vstupu klasifikátoru bude datová sada a na výstupu bude interval, který určí s jakou úspěšností bude datová sada ohodnocena.

Klíčová slova: DGA — strojové učení — genetické algoritmy — kvalita datových sad — predikce úspěšnosti

Přiložené materiály: *N/A*

*xsetin00@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Úvod

Generování datových sad pomocí genetických algoritmů je často velice časově náročné kvůli určování kvality vygenerovaných jedinců. Pro určení kvality jedinců nám slouží fitness funkce. V souvislosti s DGA (domain generation algorithm) je pro nás fitness funkce DGA detektor. Zrychlením výpočtu fitness funkce můžeme uvolnit hodně strojového času na její výpočet. Jsme schopni dosáhnout až desetinásobného zrychlení fitness funkce. Proto chceme rychle a efektivně ohodnocovat datové sady. Výsledkem ohodnocení bude přiřazená třída na základě předpokládané úspěšnosti datové sady po aplikování klasifikátoru. Cílem je vytvořit množinu parametrů charakterizující danou datovou sadu a na základě parametrů poté určit kvalitu datové sady. Díky získanému ohodnocení můžeme snadněji vytvářet kvalitní umělé datové sady s předem známou úspěšností.

Princip řešení spočívá ve využití strojového učení. Výsledkem bude natrénovaný klasifikátor, jehož výstupem bude interval vyjadřující procentuální úspěšnost. Kvůli konkrétní aplikaci na datové sady obsahující doménová jména neexistují podobná řešení, která by plnila podobný úkol. Specifický problém si vyžaduje

specializované řešení. Samozřejmě znalost DGA[1] a teorie, která se zabývá generováním datových sad pomocí genetických algoritmů (viz. sekce 2), je důležitou součástí pro pochopení problematiky. Práce, která se zabývá interpretací a mírou transparentnosti modelu je například článek CHIRPS: Explaining random forest classification [1], který je o vysvětlitelnosti random forest modelu. Článek je více teoretický a stejně jako my využívá definic při řešení problému. V rámci naší práce je také kladen důraz na zkoumání jak daný model funguje, ale náš přístup je nepřímý. Prostřednictvím datových sad, které dodáme modelu můžeme odhalit jeho slabé stránky a díky charakteristice sady i zjistíme jaké jsou důvody. Další článek, který řeší kvalitu modelů a jejich průhlednost je The Mythos of Model Interpretability [2]. V článku se pohybují na úrovní modelů a jejich porovnání a my chceme určovat kvalitu na úrovni datových sad. Poslední článek řešící podobnou problematiku, je AIMQ: a methodology for information quality assessment [3]. Jejich přístup spočívá v určení obecné kvality informací pomocí informačního zisku. Rozdělují data do 4 hlavních kategorií a popisují metodologii, jakým způsobem získat kvalitu informací. Náš přístup by měl vyústit v ohodnocení datové sady pomocí stanovených metrik.

Přístup k problému spočívá v tom, že budeme mít k dispozici velké množství datových sad, vygenerovaných pomocí genetického algoritmů. Datové sady obsahují seznam doménových jmen a příznak, zda se jedná o validní nebo DGA doménu. V první fázi přijde na řadu ohodnocení sad pomocí získaných klasifikátorů detekující DGA. Získáním úspěšností jednotlivých sad získáme množinu dat, která bude cílem predikce ohodnocujícího klasifikátoru. Následně se spočítají agregované parametry datové sady, které budou sloužit jako rozhodovací prvky při trénování klasifikátoru.

Účelem výsledného řešení je urychlit výpočet fitness funkce při generování umělých datových sad. Současné řešení je zdlouhavé, protože vyhodnocení datové sady DGA detektorem může být často časově náročné. Důvodem časové náročnosti je buď složitý rozhodovací model, nebo složitý a komplexní výpočet vektoru atributů. Stručně řečeno ušetříme spoustu času s generování kvalitních datových sad, protože díky aplikování ohodnocovacího klasifikátoru tento proces značně urychlíme.

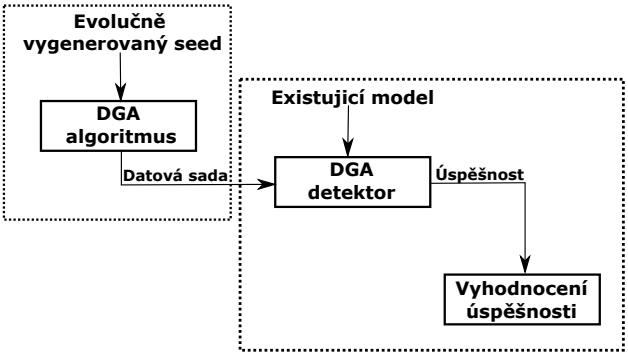
Zásadním problémem je výběr vhodných atributů, díky kterým budeme moci rychle a kvalitně predikovat ohodnocení datové sady. Musí se zvážit do kolika tříd se bude klasifikovat při udržení dostačující přesnosti. Pro ilustraci si představme, že máme datovou sadu a chceme predikovat její úspěšnost. Výsledkem můžou být 4 intervaly 0-25 %, 25-50 %, 50-75 %, 75-100 %. Když získáme kvalitní atributy a budeme mít k dispozici rozmanitou množinu datových sad, jenž bude obsahovat dostatečné zastoupení všech úspěšností, tak se nám může podařit tyto intervaly ještě rozmělnit, a zpřesnit tak klasifikaci. V rámci řešení bude také vyzkoušen regresní způsob predikce, při které se budeme snažit odhadnout už přímo numerickou hodnotu představující cílovou úspěšnost. Tímto přístupem se tak vyhneme vytváření klasifikačních tříd.

[1] https://zvelo.com/domain-generation-algorithms-dgas/

2. Generování datových sad

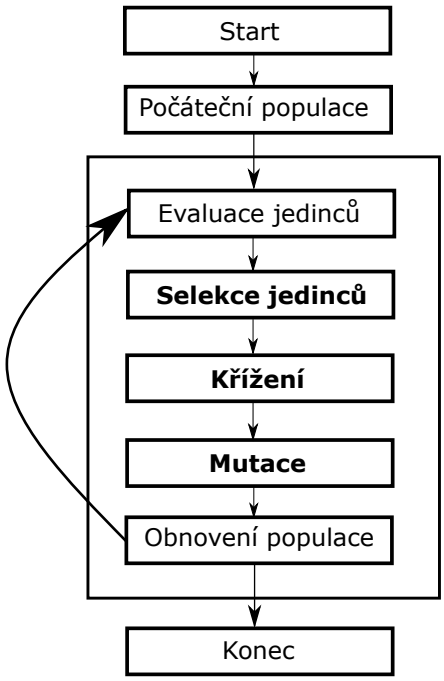
Hlavním důvodem, proč chceme generovat datové sady genetickým algoritmem, je získání kvalitní datové sady. V momentě, kdy máme k dispozici kvalitní trénovací datovou sadu, tak na ní můžeme natrénovat přesný klasifikátor. Často se můžeme setkat s klasifikátory, které nemají přesné výsledky jen kvůli tomu, že byly natrénovány na nevhodné datové sadě. Trénovací datová sada není dostatečně rozsáhlá a nemá rovnoměrné zastoupení predikovaných tříd. Pomocí genetického algoritmu se nám daří vytvářet datové sady, na kterých dosahuje klasifikátor nízké úspěšnosti. V našem případě při detekci DGA můžeme využitím umělých sad získat kvalitní datovou sadu zahrnující širokou škálu DGA algoritmů a získat tak přesnější klasifikátor.

Při generování datových sad využíváme takzvaných seedů, podle kterých vytváří DGA algoritmus konkrétní doménová jména. Seed nám v podstatě určuje jak bude výsledná datová sada vypadat. Zde přichází na řadu genetický algoritmus, pomocí kterého můžeme získat nejlepší seed a vytvořit tak dobrou datovou sadu. Na obrázku 1 můžeme vidět proces, jakým je vytvářena datová sada pomocí DGA algoritmu a její následné ohodnocení. V kontextu genetického algoritmu má DGA detektor roli fitness funkce, která určuje úspěšnost vygenerovaná sady na referenčním klasifikátoru.



Obrázek 1. Generování DGA sad a jejich ohodnocení

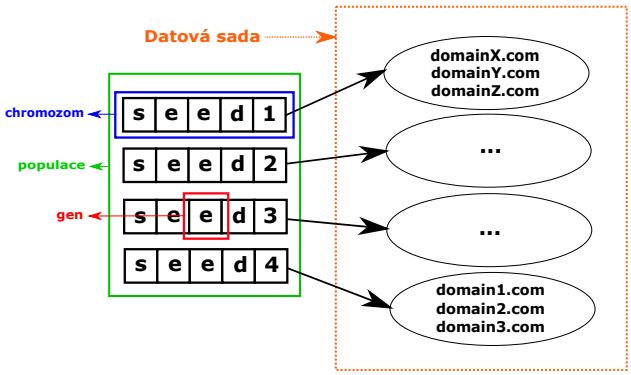
V následujícím textu bude stručně popsán postup, jak jsou tvořeny datové sady za pomocí genetického algoritmu. Celý proces je vizualizovaný na obrázku 2. Na počátku je náhodně vygenerovaná náhodná populace, která nastartuje evoluční vývoj.



Obrázek 2. Proces genetického algoritmu na generování DGA seedů.

V první fázi přichází na řadu ohodnocení populace. Jak bylo uvedeno výše, tak z populace, která se skládá ze seedů je vygenerovaná datová sada a ta je následně klasifikována. Výsledek klasifikace je právě zmiňovaná fitness funkce, která určí kvalitu populace. Poté následuje selekce nejkvalitnějších jedinců (seedů). Existují různé přístupy jak vybrat nejlepšího jedince. Například Roulette Wheel Selection, Tournament Selection, Rank Selection nebo Random Selection. Po získání nejlepších jedinců chceme vzájemné křížení, za účelem vzniku dalších kvalitních jedinců. Opět máme různé možnosti křížení, které jsou například One Point Crossover, Multi Point Crossover, Without Crossover. Na závěr je provedena mutace (například náhodným prohozením znaků v seedu) jedinců, díky které můžeme s určitou pravděpodobností dosáhnout dalšího kvalitního jedince. Výsledná populace (nová generace) je poté složena jen z jedinců vzniklých křížením a mutací. Výše uvedený popis se nadále opakuje, dokud nejsme spokojeni s kvalitou populace [4].

Pro uchopení věcí do souvislostí je na obrázku 3 uvedena analogie mezi pojmy DGA terminologie a terminologie genetického algoritmu. Populace se skládá z chromozomů (seedů) a chromozomy obsahují gen, jenž reprezentují znaky. Výsledná datová sada se potom skládá ze skupin doménových jmen, které byli vygenerovány pomocí stejného seedu. Každý seed může být na vstupu jiného DGA algoritmu a rozdílné skupiny doménových jmen, nám tak umožní diverzifikovat a zkvalitnit cílovou datovou sadu.



Obrázek 3. Analogie mezi DGA a genetickým algoritmem.

Celý proces generování datových sad není výsledkem článku. Pouze byla převzata teorie [4] od mého vedoucího, který na generování sad pracuje. Jelikož se používají datové sady, které byly vygenerovány zmíněným způsobem, tak je vhodné osvětlit proces, který stojí za jejich vznikem.

3. Trénovací datová sada

Pokud chceme ohodnocovat datovou sadou, musíme využít strojového učení. Manuálním pozorování datové sady bychom ničeho nedosáhli. Jde nám o to, abychom určili vhodné atributy, které budou charakterizovat zkoumanou datovou sadu. Atributy musí věrohodně reprezentovat datovou sadu. Jednotlivé atributy budou tedy často výsledkem průměru nad určitým atributem, jenž popisuje jedno doménové jméno. Atributy se musí rychle počítat, aby to mělo smysl. Většinou půjde o statické atributy jako je průměrná délka doménového jména, průměrná entropie, průměrný počet numerických znaků, průměrný počet souhlásek, zastoupení validních n-gramů a anglických slov. Kromě atributů bude vstupem pro trénovací sadu množina atributů, jenž vykazují úspěšnost datové sady nad různými klasifikátory.

Více klasifikátorů zajistí jakousi vzájemnou kontrolu výsledku. Budeme chtít ověřovat, jestli zvolený princip ohodnocování opravdu funguje nehledě na použitý klasifikační model. Bude následovat stručný popis zvolených klasifikátorů.

3.1 Použité klasifikátory

Sappan model je klasifikátor implementován pomocí neuronové sítě v prostředí Pythonu. Model by vyvinut v rámci RWTH Aachen University.

Nemea model je mnou vytvořený model detekující DGA jména, který byl implementovaný v jazyce C v rámci NEMEA systému. Je založen na rozhodovacím stromu, na jehož vstupu je množina atributů, které charakterizují analyzované jméno. Výsledkem je binární klasifikace, jestli je doménové jméno validní nebo DGA.

R model volně dostupný model[2], který implementuje rozhodovací model pomocí Random forest algoritmu. Model je implementovaný ve vysokoúrovňovém jazyce R.

Primitive model je opět můj model, který je založený na manuálním pozorování a rozhoduje se na základě určeného prahu, který je hranicí pro legitimnost domény. Jednotlivé atributy mají ručně přiřazené váhy a pomocí aritmetických operací je vypočítána hodnota, která je porovnána s prahem.

3.2 Architektura

K dispozici budeme mít množinu vygenerovaných datových sad. Samotná množina nebyla vytvořena mnou, ale byla mi poskytnuta v rámci většího projektu. Vygenerované datové sady budou na vstupu mého skriptu v jazyce Python. Obecná struktura, jak bude u jednotlivých sad vypočítaná úspěšnost můžeme vidět na obrázku 4.

Obrázek 4. Princip počítání úspěšnosti datových sad.

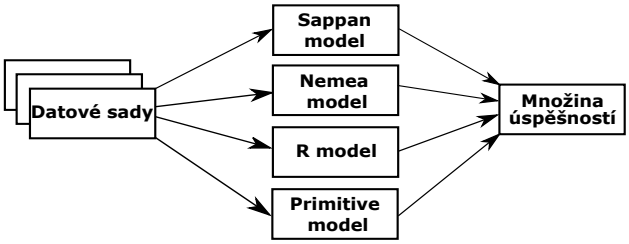
Skript v jazyce Python projde na začátku adresář s vygenerovanými datovými sadami a do seznamu si uloží všechny cesty nalezených csv souborů. Následně začne iterovat přes jednotlivé cesty obsahující datové sady. V každém cyklu je předána datová sada zmiňovaným klasifikátorům. Na obrázku 5 je navržen způsob zpracování jednotlivých sad. Datová sada je reprezentována ve formátu csv souboru. První sloupec obsahuje doménové jméno a druhý sloupec příznak, zda je adresa validní nebo DGA. Ilustrování datové sady je možné vidět v tabulce 1.

Soubor je načten do prostředí Pythonu, kde doménová jména jsou uložená v listové struktuře `domains` a příznaky v seznamu `labels`. Dvě zmíněné struktury se stávají vstupy pro klasifikátory. Výjimkou jsou klasifikátory implementované v rámci NEMEA. NEMEA požaduje na vstupu svou vlastní datovou strukturu s názvem unirec. Proto je potřeba překonvertovat

[2] https://github.com/jayjacobs/dga

Tabulka 1. Ukázka struktury datové sady

domain	type
9qmmvy.sk	1
rjvcvy.sk	1
mqzdvy.sk	1
iazovy.sk	1
pbghvy.sk	1
pdnjvy.sk	1



Obrázek 5. Architektura pro výpočet úspěšností.

csv soubor do formátu unirec. Pro převod můžeme použít NEMEA modul `logreplay`. Pro volání programů mimo prostředí Pythonu se může použít modul `os.popen('command')`. Po převedení souboru na unirec formát, tak je NEMEA modelu předána jeho cesta. Na druhou stranu u R modelu je využit balíček `rpy2`, jenž vytváří v Pythonu rozhraní pro práci s jazykem R, takže můžeme snadno spouštět R model přímo v Pythonu. Postupně se projde všech 9600 doménových jmen a u každé sady se provede klasifikace zvolenými modely.

Úspěšnosti klasifikátorů jsou vyextrahovány do csv souboru. Výslednou strukturu množiny úspěšností je možné vidět v tabulce 2. Výsledná data budou sloužit jako třídy, které budeme chtít při trénování ohodnocovacího klasifikátoru predikovat.

Tabulka 2. Ukázka úspěšností jednotlivých klasifikátorů

Datová sada	gid38_acc035.46.csv
Nemea model	27.12 %
Primitive model	67.24 %
R model	75.84 %
Sappan model	35.46 %

Poté co vypočítáme úspěšnosti klasifikátorů nad datovými sadami, tak máme vytvořenou první polovinu trénovací datové sady.

Druhá fáze zahrnuje výpočet atributů, jenž budou charakterizovat celou datovou sadu. Musíme zvolit atributy, které věrohodně reprezentují datovou sadu. Půjde o množinu statických atributů, které budou počítány pro celou datovou sadu. Budeme počítat atributy u jednotlivých doménových jmen, které následně zprůměrujeme pro celou datovou sadu. Bude možné využít i rozptyl daných atributů. Hlavní roli bude hrát následující atributy.

Délka, číslice, souhlásky jsou základní parametry, které se budou počítat. U číslic a souhlásek je způsob výpočtu založen na poměrném zastoupení těchto atributů v doméně. Vždy je získána jejich četnost a je podělena celkovou délkou domény. Atributy jsou vypočítány pro veškerá jména v datové sadě a na závěr jsou hodnoty zprůměrovány, a tak jsou získány výsledné atributy.

Entropie vyjadřuje míru neuspořádanosti řetězce. Méně pravděpodobný výskyt řetězce bude mít větší míru náhodnosti (entropie) a může tak naznačovat, že se jedná o DGA adresu. Na závěr jsou hodnoty entropie opět zprůměrované. Konkretně Shannova entropie může být vypočítána pomocí vztahu 1, který můžeme vidět níže.

$$H(X) = -\sum_{i=1}^{n} \frac{count_i}{N} \log_2 \frac{count_i}{N} \qquad (1)$$

kde řetězec je reprezentovaný náhodnou veličinou **X**, kde **N** udává délku řetězce a obsahuje **n** různých znaků s četností **count** [5].

N-gramy vyjadřují všechny možné posloupnosti znaků v rámci řetězce o velikosti N. Existuje seznam, jenž shromažďuje nejnavštěvovanější doménová jména. Databáze se nazývá Alexa[3]. Ze seznamu Alexa jsou extrahovány n-gramy o velikosti 3,4,5. Při počítání zastoupení validních n-gramů v doméně je právě využíváno shodnosti s databází n-gramů extrahovaných z Alexa seznamu. Výsledný atribut je tak průměr všech poměrných zastoupení n-gramů z Alexy s v rámci konkrétní doménové adresy.

Shoda ve slovníku vyjadřuje počet shod v anglickém slovníku v závislosti na délce domény. Za shodu ve slovníku se považuje existující podřetězec domény, jenž je součástí anglického slovníku. Výsledný parametr je opět výsledkem průměru pro celou datovou sadu.

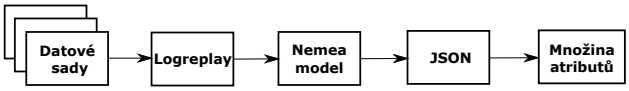
Rozptyl může být vhodným ukazatelem z jakých dat se datová sada skládá. Proto bude vypočítán rozptyl u všech dříve zmíněných atributů, které přispějí k vytvoření kvalitativních parametrů datové sady. Rozptyl je počítán pomocí následujícího vztahu 2.

$$\sigma^2 = \frac{\sum x^2}{N} - \mu^2 \qquad (2)$$

kde **x** je prvek množiny, **N** celkový počet prvků množiny a μ je průměr množiny [6].

[3] http://s3.amazonaws.com/alexa-static/top-1m.csv.zip

Způsob výpočtu atributů byl zvolen na základě rychlosti. Nejrychlejší způsob bylo využít Nemea model. Implementace v jazyce C umožňuje rychlý výpočet atributů, i přes režii s konvertování datové sady do unirec záznamu. Na obrázku 6 je možné vidět průběh výpočtu atributů. Na výstupu Nemea modelu jsou data ve formátu JSON, který je následně zpracován skriptem `Features.py`.



Obrázek 6. Způsob výpočtu atributů datové sady.

Výsledné atributy jsou v rámci skriptu opět serializovány do csv souboru. Ukázka vypočtených atributů, jenž charakterizují celou datovou sadu jsou zobrazeny v tabulce 3. Můžeme si všimnout, že rozptyl délky je nulový. Důvodem je stejná délka doménových jmen napříč všemi datovými sadami. Pro trénování ohodnocovacího modelu se tento atribut zatím neuplatňuje. Do budoucna se plánuje vygenerovat datové sady s rozdílně dlouhými doménovými jmény a tím atribut délky získá na důležitosti.

Tabulka 3. Ukázka atributů charakterizující datovou sadu

Datová sada	gid38_acc035.46.csv
Průměrná entropie	1.647
Průměrná délka	6.000
Průměrné zastoupení validních n-gramů	0.154
Průměrné zastoupení anglických slov	0.477
Průměrné zastoupení číslic	0.103
Průměrné zastoupení souhlásek	0.667
Rozptyl entropie	0.031
Rozptyl délky	0.000
Rozptyl zastoupení validních n-gramů	0.036
Rozptyl zastoupení anglických slov	0.068
Rozptyl zastoupení číslic	0.013
Rozptyl zastoupení souhlásek	0.033

V momentě, kdy získáme vektor atributů, tak se můžeme podívat na časovou náročnost jejich výpočtu. Je důležité, aby čas na výpočet atributů zabral méně času než původní fitness funkce v podobě R modelu, jinak by to postrádalo efekt. Dále je potřeba vybrat dostatečně rychlý rozhodovací model, jenž bude rozebrán v následující sekci. V tabulce 4 vidíme porovnání časů různých přístupů. První řádek tabulky vyjadřuje čas strávený čistě nad výpočtem atributů, v druhém řádku je celkový čas spotřebovaný celým Nemea modulem (inicializace datových struktur), který zahrnuje i čistý čas výpočtů atributů. Poslední řádek je čas R modelu v roli fitness funkce, který klasifikoval jednotlivé datové sady. Testování bylo prováděno na datových sadách s počtem 800 domén. Časové údaje jsou přepočítány na dobu pro 1 000 datových sad.

Tabulka 4. Srovnání časů pro výpočet atributů

	Doba výpočtu (1 000 sad)
Čistý výpočet atributů	4.34 s
Celkový čas Nemea modulu	125.39 s
Klasifikace R modelu	1323.87 s

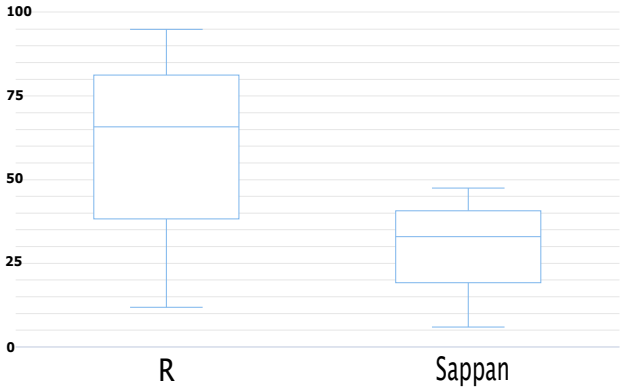
Z tabulky můžeme vidět, že celkový čas na výpočet atributů i s inicializací Nemea modelu je 125,39 s. V porovnání s dobou klasifikace R modelu získáme atributy více něž 10x rychleji. V této fázi můžeme říct, že určitě má smysl predikovat úspěšnost datové sady, pokud by to urychlilo výpočet fitness funkce až desetkrát.

Atributy, které jsou zde uvedeny nemusí být finálním ani správným řešením. Postupně se bude pracovat na zdokonalení výběru kvalitních atributů. Čím kvalitnější získáme atributy, tím lepší a přesnější budou výsledky natrénovaného klasifikátoru, který bude datové sady hodnotit. Do budoucna se na výběru atributů bude nadále pracovat a zdokonalovat tak výsledný model.

4. Tvorba klasifikátoru

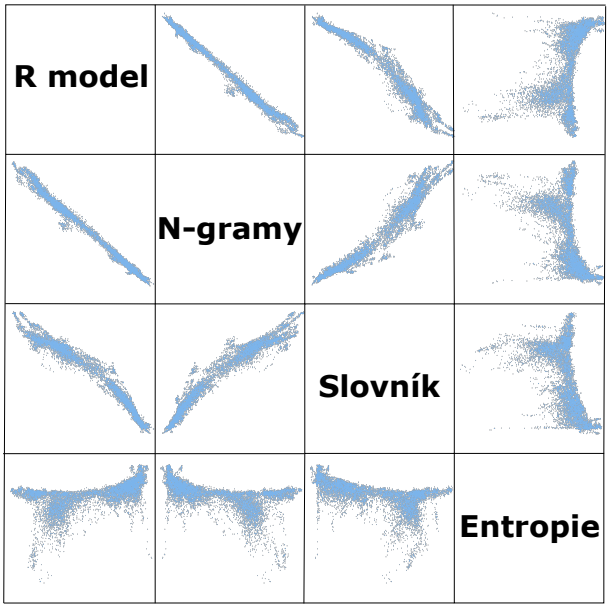
Když máme k dispozici vytvořenou trénovací datovou sadu, tak můžeme přejít ke trénování klasifikátoru a jeho následnému testování. V první fázi půjde o to vybrat vhodný rozhodovací model. Vhodným nástrojem pro prvotní srovnání rozhodovacích modelů je nástroj Rapidminer[4], který nám rychle a snadno umožní srovnat a vizualizovat rozhodovací modely. Jakmile zvolíme výsledný typ klasifikátoru, přijde na řadu implementace modelu v jazyce Python.

Součástí trénovací datové sady je množina úspěšností, které byly spočítány referenčními klasifikátory 3.1. Musíme zvolit úspěšnost jakého modelu chceme predikovat. Predikci nejde dělat obecně, protože každý klasifikátor vykazuje rozdílné chování. Je proto nutné pro každý model vygenerovat specializovanou datovou sadu. Vezmeme cílový model, vygenerujeme k němu dostatečný počet datových sad, které kategorizujeme a získáme tak výslednou trénovací sadu. Pro porovnání můžeme na obrázku 7 vidět distribuci jednotlivých úspěšností pro R a Sappan model. Z obrázku jde vidět, že datové sady vygenerované přímo pro R model mají mnohem rovnoměrnější rozložení v rámci intervalu úspěšnosti (0-100 %). Důvodem rovnoměrného rozdělení úspěšností R modelu je využití R úspěšnosti jako fitness funkce při generování datových sad, a proto nejde vytvořit obecný model pro více klasifikátorů. Hlavním účelem generování umělých datových sad je zpřesnění klasifikátoru. Zlepšení dosáhneme dodáním kvalitní trénovací datové sady. Když budeme chtít predikovat výsledky nějakého klasifikátoru, tak je dobré mít rovnoměrné zastoupení úspěšností v trénovací datové sadě. V rámci našeho prvotního řešení rozdělíme úspěšnost do 4 tříd. Z obrázku 7 lze jasně určit, že mnohem vhodnější pro predikování je model R, díky jeho rovnoměrnějšímu rozdělení úspěšností nad trénovací datovou sadou.



Obrázek 7. Porovnání distribuce úspěšností R a Sappan modelu.

Před samotným výběrem vhodného rozhodovacího modelu se podíváme na vzájemnou korelaci některých atributů. Obrázek 8 vyjadřuje grafovou matici s korelacemi mezi vybranými atributy. Pro přehlednost nejsou uvedeny všechny atributy.



Obrázek 8. Grafová matice korelací mezi vybranými atributy.

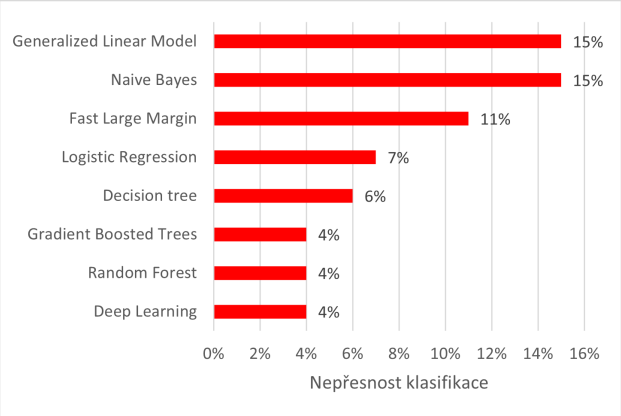
Můžeme si všimnout, že úspěšnost R modelu a průměrné zastoupení n-gramů vykazují zápornou korelaci. Z toho můžeme usoudit, že pokud je poměrné zastoupení validních n-gramů nižší, tak je pro R model snazší predikovat DGA adresu. Podobně to je u shody

[4] Software na analýzu dat a aplikování strojového učení

ve slovníku. U průměrné entropie je korelace s výslednou úspěšností horší, ale lze pozorovat lehkou tendenci ke kladné korelaci, což nám říká, že čím vyšší neuspořádanost řetězce (entropie), tak tím se zvyšuje pravděpodobnost správné predikce DGA jména.

Po prozkoumání atributů začneme nad datovou sadou aplikovat množinu algoritmů strojového učení. Jak bylo zmíněno dříve, tak pro rychlé porovnání rozhodovacích modelu je vhodné použít Rapidminer. V Rapidmineru jsme nad datovou sadou aplikovali všechny dostupné modely, které budou predikovat úspěšnost R modelu do 4 tříd. Třídy vzniknou rozdělením rozsahu úspěšností na 4 části. V rámci pozdější implementace modelu budou třídy rozděleny přesně do intervalů 0-25 %, 25-50 %, 50-75 %, 75-100 %. Po výpočtu všech modelů se zobrazilo jejich porovnání.

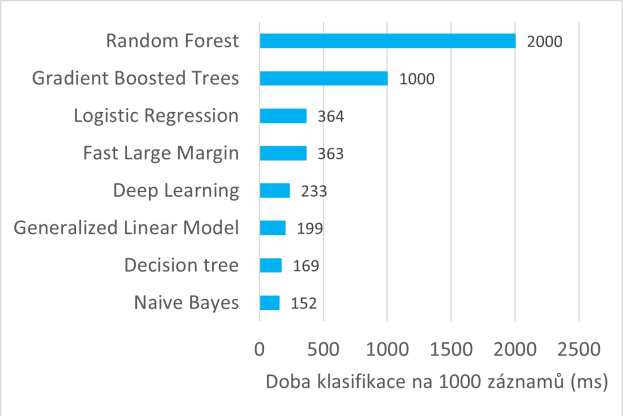
Srovnání bylo vyjádřeno dvěma grafy. První graf 9 vykresloval procentuální nepřesnost srovnávaných modelů a druhý graf 10 jejich časovou náročnost klasifikace. Srovnání bylo prováděno na datové sadě velikosti 4000. Výsledný model by měl být hlavně rychlý při klasifikování, jinak by poztrácelo celé řešení smysl, pokud by ohodnocovací klasifikátor byl pomalejší než původní DGA detektor.



Obrázek 9. Procentuální nepřesnost srovnávaných modelů.

Z grafu 9 vyplývá, že mezi nejpřesnější modely patří Gadient Boosted Trees, Random Forest, Deep Learning. Nepřesnost modelů se pohybuje kolem 4 % a hned za nimi je Rozhodovací strom s 6 % nepřesností.

Pokud zvážíme znalosti z dalšího grafu 10 s časovou náročností klasifikace, tak zjistíme, že nejvhodnější model je právě rozhodovací strom. Rozhodovací strom dokáže klasifikovat 1000 záznamů za 169 ms. Ostatní přesné modely zaostávají právě v době klasifikace, což je pro nás rozhodující faktor. Deep Learning je jediný srovnatelný s rozhodovacím stromem, ale jeho nevýhodou je složitost. Když dáme dohromady čas výpočtu atributů 4 a dobu klasifikace rozhodovacího stromu, tak jsme schopni predikovat úspěšnost 1 000



Obrázek 10. Časová náročnost srovnávaných modelů.

datových sad za 125.55 s. Tímto přístupem můžeme zrychlit výpočet fitness funkce desetkrát za cenu odchylky predikované úspěšnosti. Další část textu bude obsahovat právě implementaci rozhodovacího stromu v prostředí Python.

Prvním krokem v rámci implementace bylo zvolení knihovny, která umožňuje implementovat rozhodovací strom. Jednou z mnoha knihoven pro strojové učení je `scikit-learn`[5], která bude při implementaci využita. Dále přišlo na řadu načtení a zpracování trénovací sady. Pro práci s daty byly použity knihovny `pandas`[6] a `numpy`[7], které urychlily zpracování dat. Před samotným aplikováním klasifikátoru bylo potřeba rozdělit sadu na data, která chceme predikovat (úspěšnost R modelu) a atributy, podle kterých se klasifikátor bude rozhodovat. Predikovaná data se musely podrobit reklasifikaci a rozdělit úspěšnosti do 4 tříd. Reklasifikace je naznačena níže.

- Třída 1 — 0-25 %
- Třída 2 — 25-50 %
- Třída 3 — 50-75 %
- Třída 4 — 75-100 %

Jakmile máme vytvořené pole s predikovanými třídami a vícerozměrné pole s atributy, tak můžeme přejít k samotnému trénování a testování rozhodovacího stromu. Pro otestování kvality rozhodovacího stromu byly použity dva způsoby validace. První způsob spočívá v běžném rozdělení datové sady na trénovací a testovací sadu v poměru 2:1. Druhý způsob je založený na křížové validaci, kdy se v 5 iteracích vytvoří vždy nová trénovací a testovací datová sada. Výsledkem křížové validace je směrodatnější úspěšnost výsledného modelu díky zprůměrování více individuálních trénovacích procesů.

[5] https://scikit-learn.org/stable/
[6] https://pandas.pydata.org/
[7] https://numpy.org/

Nyní si podrobněji rozebereme výsledky prvního způsobu rozdělení datové sady. Na rozdělení byla použita funkce `train_test_split()`, která stratifikovala datovou sadu podle predikovaných tříd. Jinými slovy při rozdělení zajistila rovnoměrné zastoupení tříd. Takže po rozdělení máme dvě třetiny původních dat určené k trénování (6 720) a zbylou třetinu pro testování (2 880). Po natrénování modelu můžeme přejít k interpretaci výsledků. V tabulce 5 máme zobrazenou matici vyjadřující výsledky klasifikace. Řádky vyjadřují, jak daný klasifikátor predikoval příslušnost záznamů k jednotlivým třídám. A na druhou stranu sloupce zastupují opravdovou příslušnost záznamů ke třídám. Tabulka přehledně ukazuje, kolik záznamů bylo správně klasifikováno. Můžeme si všimnout, že při chybné klasifikaci vždy došlo jen k zařazení do vedlejší kategorie úspěšnosti. Je to pro nás pozitivní znamení a můžeme říct, že model funguje podle očekávání a nepredikuje diametrálně jinou kategorii.

Tabulka 5. Confusion matrix

	Skutečnost			
	1	2	3	4
Pred. 1	181	32	0	0
Pred. 2	17	951	14	0
Pred. 3	0	17	516	22
Pred. 4	0	0	20	1110

Z tabulky 5 můžeme vypočítat další ukazatele [7] určující kvalitu klasifikátoru. Mezi ně patří úspěšnost, přesnost, senzitivita a F1 skóre. Úspěšnost je základním ukazatelem vyjadřující podíl mezi správně klasifikovanými záznamy a celkovým počtem záznamů. Přesnost udává poměr mezi správně predikovanými záznamy a všemi záznamy predikované do konkrétní třídy. Senzitivita je podílem mezi správně predikovanými záznamy a všemi záznamy, jenž jsou doopravdy prvky predikované třídy. F1-score je potom váženým průměrem přesnosti a senzitivity. Všechny tyto metriky byly exportovány do tabulky 6. V tabulce je jsou navíc uvedeny četnosti tříd, celkový počet testovaných záznamů a průměry jednotlivých metrik. Můžeme vidět, že úspěšnost modelu je 96 %.

Tabulka 6. Výsledky klasifikátoru na testovací sadě

	Přesnost	Senzitivita	F1-score	Četnost
Třída 1	0.85	0.91	0.88	198
Třída 2	0.97	0.95	0.96	1000
Třída 3	0.93	0.94	0.93	550
Třída 4	0.98	0.98	0.98	1132
Úspěšnost			0.96	2880
Průměr	0.93	0.95	0.94	2880
Vážený průměr	0.96	0.96	0.96	2880

Druhý způsob testování klasifikátoru spočíval v křížové validaci. Jak už bylo zmíněno dříve, tak křížová validace v každém kroku vytvoří novou trénovací a testovací datovou sadu a provede natrénování a otestování modelu. Bylo zvoleno 5 kroků. V každém kroku se vzala náhodně jedna pětina celkové datové sady a zbytek dat byl označen jako trénovací sada. Zmíněným způsobem ověříme, zda úspěšnost prvního způsobu testování nebyla jenom náhoda. V tabulce 7 jsou uvedeny úspěšnosti jednotlivých iterací. Můžeme vidět, že celková úspěšnost po křížové validaci je 94 %.

Tabulka 7. Výsledky křížové validace

	Úspěšnost
Iterace 1	0.938
Iterace 2	0.941
Iterace 3	0.936
Iterace 4	0.957
Iterace 5	0.948
Průměr	0.94
Směrodatná odchylka	0.01

Jelikož se nám v průběhu práce podařilo vygenerovat nové datové sady s cílem rovnoměrnějšího zastoupení do kategorií, tak můžeme zkusit rozmělnit predikované intervaly. Experimenty budou prováděny nad kategoriemi o velikosti 5 a 10. Výsledné třídy vzniknou vždy rozdělením intervalu 100 požadovaným číslem. Proces implementace modelů je totožný s předchozím případem se čtyřmi třídami. V tabulkách 8 a 9 jsou zobrazeny confusion matice pro testované intervaly.

Tabulka 8. Confusion matrix s 5 třídami

	Skutečnost				
	1	2	3	4	5
Pred. 1	676	81	45	44	0
Pred. 2	61	683	89	37	0
Pred. 3	38	53	614	90	0
Pred. 4	17	33	71	702	13
Pred. 5	0	0	0	16	848

Tabulka 9. Confusion matrix s 10 třídami

	Skutečnost									
	1	2	3	4	5	6	7	8	9	10
Pred. 1	137	7	4	2	11	5	7	0	0	0
Pred. 2	11	123	11	7	4	4	12	0	0	0
Pred. 3	10	4	137	19	1	2	7	0	0	0
Pred. 4	7	9	12	133	22	3	3	4	0	0
Pred. 5	9	6	4	14	110	15	8	5	0	0
Pred. 6	2	5	11	4	10	124	29	13	0	0
Pred. 7	10	9	1	4	10	20	75	27	0	0
Pred. 8	0	0	0	3	2	5	39	141	27	1
Pred. 9	0	0	0	0	0	0	0	25	141	10
Pred. 10	0	0	0	0	0	0	0	1	17	173

V momentě, kdy se snažíme intervaly rozmělnit, tak samozřejmě klesá i výsledná přesnost klasifikátoru. Z uvedených matic jde vidět, že většina predikcí se stále drží ve správné kategorii, ale ovšem chybné predikce už zasahují nejen do vedlejších tříd. Také si můžeme všimnout kvalitnější klasifikace datových sad s vysokou úspěšností. Metriky získané z natrénovaných modelů nad testovacími daty jsou opět vygenerované do tabulek 10 a 11. Datová sada pro 5 kategorií obsahuje 14 035 záznamů a sada pro 10 kategorií 6 091 záznamů. Testovací sady zastupovali 30 % z celkového počtu. Model pro 5 tříd dosahuje po křížové validaci úspěšnosti 83 %. Model pro 10 tříd dosáhl křížovou validací průměrné úspěšnosti 72 %. Můžeme sledovat klesající přesnost s narůstajícím počtem kategorií.

Tabulka 10. Výsledky klasifikátoru pro 5 tříd

	Přesnost	Senzitivita	F1-score	Četnost
0-20	0.80	0.85	0.83	792
20-40	0.79	0.80	0.79	850
40-60	0.77	0.75	0.76	819
60-80	0.84	0.79	0.81	889
80-100	0.98	0.98	0.98	861
Úspěšnost			0.84	4211
Průměr	0.84	0.84	0.84	4211
Vážený průměr	0.84	0.84	0.84	4211

Tabulka 11. Výsledky klasifikátoru pro 10 tříd

	Přesnost	Senzitivita	F1-score	Četnost
0-10	0.79	0.74	0.76	186
10-20	0.72	0.75	0.73	163
20-30	0.76	0.76	0.76	180
30-40	0.69	0.72	0.70	186
40-50	0.64	0.65	0.65	170
50-60	0.63	0.70	0.66	178
60-70	0.48	0.42	0.45	180
70-80	0.65	0.65	0.65	216
80-90	0.80	0.76	0.78	185
90-100	0.91	0.94	0.92	184
Úspěšnost			0.71	1828
Průměr	0.71	0.71	0.71	1828
Vážený průměr	0.71	0.71	0.71	1828

Doposud bylo přistupováno k problematice pomocí rozdělení úspěšností do tříd. Nyní budeme chtít provádět regresní predikci. Princip je založen na odhadování přímo numerické hodnoty. V praxi to znamená, že dodáme modelu vektor atributů a výsledkem bude přímo číslo, které bude reprezentovat predikovanou úspěšnost datové sady. Pro ověření, které modely jsou vhodné pro tento přístup byla už použita rovnou zmiňovaná knihovna sklearn. Knihovna poskytuje množinu regresních modelů, které splňují naše požadavky. Na základě příspěvku o predikování numerických hodnot [8], byly nad daty vyzkoušeny tyto modely:

- Linearní regresor
- Regresor rozhodovacího stromu
- Random forrest regresor
- Regresor neuronové sítě
- Regresor K-nejbližších sousedů

Po vyhodnocení všech regresních modelů, přišlo na řadu jejich porovnání. Metriky[8], jenž udávají kvalitu regresního modelů, se liší od třídních modelů. Mezi hlavní metriky patří průměrná absolutní chyba, jejíž hodnota udává o kolik se predikovaná úspěšnost průměrně liší o opravdovou hodnotu. Dále můžeme určit maximální chybu, která zachycuje největší rozdíl mezi predikovanou a reálnou hodnotou. Poslední zjišťovaná metrika je vysvětlený rozptyl (explained variance) vyjadřující rozptyl mezi predikovanými a reálnými hodnotami. Nejlepší hodnota metriky dosahuje jedničky a nižší hodnoty značí vyšší variaci mezi hodnotami dvou skupin. V tabulce 12 je zobrazeno porovnání jednotlivých modelů a zmíněných metrik.

Tabulka 12. Porovnání regresních modelů

Model	Prům. chyba	Max. chyba	Vys. rozptyl
Linearní regrese	1.26	7.48	0.9946
Rozhodovací strom	1.15	8.46	0.9952
Random forrest	0.82	5.83	0.9976
Neuronová sít	1.02	5.54	0.9965
K-nejbližších sousedů	1.22	9.99	0.9945

Z tabulky můžeme vyčíst, že nejlepších výsledků dosahuje model Random forrest s průměrnou chybou 0.82. Dobře si stojí ale i ostatní modely, které dodávají také solidní výsledky. Nevýhodou Random forrestu může být jeho pomalejší klasifikace vyplývající z podstaty jeho implementace (delší čas z důvodu vyhodnocení jednotlivých stromů). Z toho důvodu by byl vhodnější spíše rozhodovací strom nebo neuronová síť, která ale není natolik transparentní.

Celkový úspěch implementace můžeme považovat za zdařilý. Když vezmeme v potaz, že do budoucna se budeme snažit získat kvalitnější množinu atributů, která umožní vytvořit kvalitnější rozhodovací model. Za zmínku také stojí různá kvalita klasifikátorů na základě trénovací sady. Trénovací datová sada má velký vliv na konečný výsledek. Když jsme aplikovali nad dvěma různými sadami stejný klasifikátor pro 4 třídy, tak výsledná úspěšnost se lišila až o 12 %.

Proti třídní klasifikaci jsme postavili kontinuální predikci numerických hodnot pomocí regresních tech-

[8] https://scikit-learn.org/stable/modules/model_evaluation.html#regression-metrics

nik. V porovnání s třídní klasifikací dosahovala přesnějších výsledků. Nejhorší regresivní model dosahoval maximální odchylky 9.99, což je v podstatě maximální chyba u klasifikace do 10 tříd v případě správné predikce. Nehledě na to, že při špatné klasifikaci může dojít k mnohem vyšší odchylce. V případě regrese hrála opět roli trénovací datová sada. V jednom případě jsme dosáhli průměrné odchylky kolem 1 a v tom horším případě kolem 5.

5. Závěr

Cílem článku bylo vytvořit klasifikátor, který bude predikovat ohodnocení datových sad s doménovými jmény. Předcházelo tomu osvětlení teorie o generování datových sad za pomocí genetického algoritmu. Následovalo tvoření trénovací datové sady, kterou můžeme rozdělit na dvě části. První část se skládá z množiny úspěšností referenčních klasifikátorů. Druhá část reprezentuje vektor atributů, který bude vstupem pro náš klasifikátor. Součástí tvoření sady byl také popis architektury, jakým způsobem trénovací sada vznikla. Dalším krokem byl výběr vhodného klasifikačního modelu. Po provedené analýze atributů a srovnání modelů jsme došli k závěru, že nejvhodnějším modelem je rozhodovací strom nebo neuronová síť (složitost) pro třídní i regresní klasifikaci. V rámci implementace byly uvedeny prostředky k realizaci klasifikátoru. Dále bylo provedeno testování a rozebrání výsledných statistik.

Výsledný model pro 4 kategorie dosahoval po provedení křížové validace průměrné úspěšnosti 94 % se směrodatnou odchylkou 0.01. Při chybné klasifikaci dojde vždy jen k zařazení do sousední kategorie úspěšnosti, a dokazuje to funkčnost a realizovatelný vývoj klasifikátoru. Model pro 5 kategorií dosáhl průměrné úspěšnosti 83 % a model pro 10 kategorií 72 %. U více kategorií už byla klasifikace méně přesná a chybné klasifikace zasahovaly nejen do vedlejší třídy. Regresní techniky v porovnání s třídním přístupem vykazovaly vyšší přesnost s průměrnou odchylkou 1 napříč různými algoritmy.

Hlavním přínosem práce je zrychlení generování datových sad, jenž obsahují doménové jména. Navržený způsob ohodnocovaní může až desetkrát urychlit výpočet fitness funkce. R model jako referenční fitness funkce dokáže zpracovat 1 000 datových sad za 1323.87 s. Ohodnocovací klasifikátor dokáže stejný počet sad ohodnotit za 125.55 s. Výsledný klasifikátor by tak mohl nahradit původní DGA detektor v roli fitness funkce. Dále získáme schopnost ohodnotit datovou sadu pomocí množiny parametrů a na jejich základě předpovědět úspěšnost budoucí klasifikace.

Výsledek práce je součástí většího projektu, na kterém pracuje můj vedoucí a po dalším zkvalitnění stávajících výsledků by mohla tato práce rapidně urychlit proces generování umělých datových sad pro DGA klasifikaci.

Poděkování

Rád bych poděloval svému vedoucímu Peterovi Tisovčíkovi za jeho pomoc.

Literatura

[1] Gaber M.M. & Azad R.M.A. Hatwell J. Explaining random forest classification. Artif Intell Rev 53, 6 2020. https://doi.org/10.1007/s10462-020-09833-6.

[2] Zachary Chase Lipton. The mythos of model interpretability. *CoRR*, abs/1606.03490, 2016. http://arxiv.org/abs/1606.03490.

[3] Yang W. Lee, Diane M. Strong, Beverly K. Kahn, and Richard Y. Wang. Aimq: a methodology for information quality assessment. *Information & Management*, 40(2):133–146, 2002. https://www.sciencedirect.com/science/article/pii/S0378720602000435.

[4] Peter Tisovčík. Odhalování kybernetických útoků s využitím technik strojového učení (dátové sady). online, 10 2020. https://docs.google.com/presentation/d/1s0REtKg5xJDsqds_scf4Io_k-m5k2kCiBPQtwD3_W2k/edit?us=sharing".

[5] Mroman. Entropy. online, 2 2013. https://rosettacode.org/wiki/Entropy.

[6] Rod Pierce. 'standard deviation and variance' math is fun. online, 12 2020. http://www.mathsisfun.com/data/standard-deviation.html.

[7] Exsilio Solutions. Accuracy, precision, recall & f1 score: Interpretation of performance measures. online, 9 2016. https://bit.ly/39tjN5M.

[8] Stacey Ronaghan. Machine learning: Trying to predict a numerical value. online, 7 2018. https://srnghn.medium.com/machine-learning-trying-to-predict-a-numerical-value-8aafb9ad4d36.