VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

ANALÝZA DAT SÍŤOVÉ KOMUNIKACE MOBILNÍCH ZAŘÍZENÍ

ANALYSIS OF MOBILE DEVICES NETWORK COMMUNICATION DATA

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. LUKÁŠ ABRAHAM**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. IVANA BURGETOVÁ, Ph.D.**
SUPERVISOR

BRNO 2020

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2019/2020

Zadání diplomové práce

Student: **Abraham Lukáš, Bc.**
Program: Informační technologie Obor: Informační systémy
Název: **Analýza dat síťové komunikace mobilních zařízení**
Analysis of Mobile Devices Network Communication Data
Kategorie: Data mining

Zadání:

1. Seznamte se s protokoly DNS a SSL/TLS.
2. Seznamte se se základními úlohami z oblasti dolování dat a s technikami pro předzpracování dat.
3. Prostudujte možnosti identifikace mobilních zařízení prostřednictvím informací zjištěných ze síťové komunikace těchto zařízení.
4. Seznamte se s dostupnými datovými sadami, které obsahují DNS a SSL/TLS data z reálné komunikace mobilních zařízení. Tyto datové sady vhodným způsobem předzpracujte pro další použití.
5. Po dohodě s vedoucí vyberte vhodnou metodu z oblasti dolování dat, kterou lze použít pro identifikaci mobilních zařízení na základě kombinace DNS a SSL/TLS dat obsažených v dostupných datových sadách.
6. Zvolenou metodu implementujte a otestujte na dostupných datových sadách.
7. Zhodnoťte dosažené výsledky.

Literatura:

- Peterson, Larry L., Davie, Bruce S.: *Computer Networks: A System Approach*, 4th edition, Amsterdam: Elsevier; Morgan Kaufmann, 2007, 806 p., ISBN 978-0-12-370548-8.
- Kurtz, A.,Gascon, H., Becker, T., Rieck, K., Freiling, F.: Fingerprinting Mobile Devices Using Personalized Configurations, In Proceedings on Privacy Enhancing Technologies, PoPETS, 2016 (1), p. 4-19.
- Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 3rd edition, Morgan Kaufmann Publishers, 2012, 703 p., ISBN 978-0-12-381479-1

Při obhajobě semestrální části projektu je požadováno:

- Body 1 až 5.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Burgetová Ivana, Ing., Ph.D.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2019
Datum odevzdání: 3. června 2020
Datum schválení: 24. října 2019

Abstrakt

Práce na svém začátku popisuje protokoly DNS a SSL/TLS, věnuje se hlavně komunikaci mezi zařízeními pomocí těchto protokolů. Poté si povíme něco o předzpracování dat a jejich čištění. Dále se práce zaobírá základními technikami pro dolování dat, jako jsou klasifikace dat, asociační analýza, vyhledávání dokumentů, regresní analýza a shluková analýza. V další kapitole si můžeme přečíst něco o tom, jak se dají identifikovat mobilní zařízení v síti. Zhodnotíme datové sady, které obsahují nasbíraná data z komunikace mezi protokoly DNS a SSL/TLS se kterými se bude pracovat v praktické části. Po té se konečně dostaneme k návrhu systému pro analýzu dat síťové komunikace. Popíšeme si použité knihovny a celou implementaci systému. Provedeme velké množství experimentů, které na konec ohodnotíme.

Abstract

At the beginning, the work describes DNS and SSL / TLS protocols, it mainly deals with communication between devices using these protocols. Then we'll talk about data preprocessing and data cleaning. Furthermore, the thesis deals with basic data mining techniques such as data classification, association rules, information retrieval, regression analysis and cluster analysis. The next chapter we can read something about how to identify mobile devices on the network. We will evaluate data sets that contain collected data from communication between the above mentioned protocols, which will be used in the practical part. After that, we finally get to the design of a system for analyzing network communication data. We will describe the libraries, which we used and the entire system implementation. We will perform a large number of experiments, which we will finally evaluate.

Klíčová slova

dolování dat, DNS, SSL, TLS, SSL/TLS, datové sady, mobilní zařízení, rozhodovací stromy, asociační pravidla, neuronové sítě, regresní analýza, shluková analýza, předzpracování dat, redukce dat, čištění dat, TF-IDF, MySQL databáze, neuron, python, vyhledávání dokumentů, klasifikace dat

Keywords

data mining, DNS, SSL, TLS, SSL/TLS, data sets, mobile device, decision tree, association rule learning, artificial neural network, regression analysis, cluster analysis, data preprocessing, data reduction, data cleansing, data cleaning, TF-IDF, MySQL database, neuron, python, information retrieval, statistical classification

Citace

ABRAHAM, Lukáš. *Analýza dat síťové komunikace mobilních zařízení.* Brno, 2020. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Ivana Burgetová, Ph.D.

Analýza dat síťové komunikace mobilních zařízení

Prohlášení

Prohlašuji, že jsem tento semestrální projekt vypracoval samostatně pod vedením paní Ing. Ivany Burgetové Ph.D.. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Lukáš Abraham
3. června 2020

Poděkování

Chtěl bych poděkovat hlavně vedoucí své semestrální práce Ing. Ivaně Burgetové Ph.D. za skvělé vedení a poskytnutí odborné pomoci a korekce textu této diplomové práce, kdykoliv bylo potřeba.

Obsah

Kapitola 1

Úvod

Tato práce se zabývá hlavně analýzou datových sad, které jsou získány z komunikací v síti. Jedná se o dva, lépe řečeno tři protokoly, a to DNS a SSL/TLS, kde TLS je nástupcem SSL. Ze získaných datových sad se pokusíme určit mobilní zařízení a najít ho v dalších datových sadách. Dále se podíváme na dolování dat a možnosti identifikaci mobilních zařízení. Seznámíme se blíže s datovými sadami a pokusíme se co možná nejlépe určit zařízení z těchto datových sad.

Jak už bylo zmíněno výše v první čísti se zaměříme na síťové protokoly DNS a SSL/TLS. Popíšeme si, k čemu jsou dobré a jak u nich probíhá komunikace, ze které vycházejí datové sady. V další části se zaměříme na dolování dat, a to přesněji na základní praktiky v tomto odvětví informatiky a také na to, jak se vstupní data upravují. Je totiž potřeba v některých případech data před jejich dolováním upravit. Nejdůležitější částí této teoretické části je metoda TF-IDF, která nám poslouží jako základní stavební kámen pro určení mobilního zařízení z datových sad. Popíšeme si, jak se dají v dnešní době identifikovat mobilní zařízení prostřednictvím komunikace jejich uživatelů. Zaměříme se hlavně na síťovou komunikaci, jelikož ta patří v dnešní době mezi tu nejčastější a nejvíce využívanou. Seznámíme se blíže s datovými sadami. Zjistíme, jaké informace jsou v nich obsaženy.

V praktické části si představíme, jak správně uložit datové sady a jak s nimi pracovat. Zaměříme se na naprogramování metody TF-IDF a její použití v praxi. Získané výsledky zhodnotíme a vyzkoušíme jaký vliv na výsledek má využití kombinace protokolů DNS a SSL/TLS. Pokud výsledky nebudou dostatečně dobré zkusíme navrhnout i nějaké rozšíření a úpravu této metody pro daný problém a následně tyto úpravy zaintegrovat do řešení a výsledky zhodnotit. Pokud se nám vše podaří podle plánu bude výsledný program schopen po zadání určitého počtu dat schopen s velkou přesností určit dané zařízení i v jiných datových sadách. V nejlepším možném případě pokaždé najít dané zařízení v jakékoliv komunikaci.

Kapitola 2

Sítové protokoly DNS a SSL/TLS

V této kapitole se zaměříme na popis dvou síťových protokolů. Je důležité pochopit, jak pracují a k čemu slouží, abychom mohli lépe určit, jak důležité jsou jednotlivé části dat, které jsme získali z komunikace mezi zařízeními, které tyto protokoly využívají. Znalost nám také pomůže vyčistit získaná data. Může se totiž stát, že se nám do získaných datových sad dostala i data, která nesouvisí s těmito protokoly. Mohou se nám však ve vstupních sadách objevit i zajímavá data, která jsou jen výjimečná a mohou nám pomoci k identifikaci zařízení. Více o datech, které jsme získali ze síťové komunikace protokolů DNS a SSL/TLS, si povíme v kapitole 6.

2.1 Systém DNS

V této kapitole a všech podkapitolách jsem čerpal ze zdrojů: [28] [6] [7].

DNS (Domain Name System) je hierarchický a decentralizovaný jmenný systém pro počítače, zařízení, služby a další zařízení, ke kterým se připojujeme ať už skrze internet nebo lokální síť. Slouží primárně k překladu doménových jmen na IP adresy nebo k opačnému procesu. Dnes si už nedokážeme představit běžnou komunikaci mezi uživateli bez této služby. Jsme schopni během chvíle poznat, že tento systém nefunguje, pokud se nejsme schopni připojit na žádnou webovou stránku. Bez DNS taky nejsme schopní odesílat ani přijímat emailové zprávy. Tato služba však zahrnuje daleko více webových služeb. V dnešní době se hojně používá i na IP telefonii, sdílení zdrojů a lokalizaci.

2.1.1 Historie DNS

Protože je pro nás daleko jednodušší zapamatování jmen než číselných kombinací, tak jsme schopni datovat začátky DNS už při vzniku ARPANETu. Kdy mělo každé zařízení svůj vlastní lokální soubor HOSTS.TXT, který byl vytvořen jen na jednom zařízení a následně rozeslán mezi všechny ostatní zařízení pomocí služby FTP. Tento soubor spravovala organizace NIC (Network Information Center). V Unixových systémech tento soubor stále nalezneme a můžeme ho využít například k upřednostnění některého DNS serveru oproti jinému atd.

Hlavním důvodem pro vznik DNS bylo vytvoření konzistentních databází, které budou jednoduše rozšiřitelné a primárně se použijí na identifikaci síťových cílů, či zdrojů. Aby byl výkon co nejlepší tak se dají částí databází ukládat do lokálních databází na každém zařízení, takže dané zařízení nemusí pro každý překlad doménového jména komunikovat se serverem.

2.1.2 Služba DNS

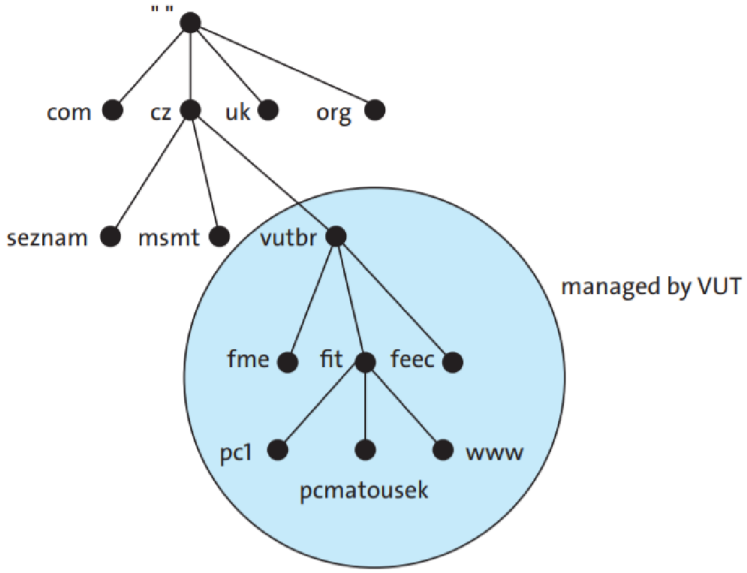
Mezi všeobecně známý a hlavní úkol DNS patří překlad doménové adresy (např. www.seznam.cz) na IP adresu (77.75.75.172). Tento proces můžeme též nazvat mapováním, či převodem. Jak už je zmíněno výše, pro lidský mozek není jednoduché zapamatovat si řadu čísel. Vezměte si, že byste si měli pamatovat místo všech webových stránek, které běžně používáte, jen čísla oddělená tečkou. Bylo by to velice nepraktické avšak na druhou stranu pro počítač a internetovou komunikaci je daleko lepší využít číselnou hodnotu pro přesnou identifikaci zdroje. Pokud chceme pro nějaké doménové jméno zjistit danou IP adresu můžeme využít příkaz "nslookup", který zobrazí pro zadané doménové jméno IP adresu. Tento příkaz funguje i obráceně.

```
C:\Users\Lukíno>nslookup www.seznam.cz
Server:  UnKnown
Address:  fe80::1

Non-authoritative answer:
Name:    www.seznam.cz
Addresses:  2a02:598:4444:1::1
          2a02:598:4444:1::2
          2a02:598:3333:1::1
          2a02:598:3333:1::2
          77.75.75.172
          77.75.75.176
          77.75.74.172
          77.75.74.176


C:\Users\Lukíno>nslookup 77.75.75.172
Server:  UnKnown
Address:  fe80::1

Name:    www.seznam.cz
Address:  77.75.75.172
```

Obrázek 2.1: Ukázka příkazu nslookup v příkazové řádce operačního systému Windows.

Služba DNS tedy obsahuje databázi všech doménových adres a k nim příslušných IP adres. Tato databáze má více kopií a je umístěna na vícero počítačích ve všech koutech světa, pro rychlejší získání daného překladu. Počítačům, které obsahují tyto databáze říkáme nameservery, doménové servery (jmenné servery) nebo také servery DNS. Tyto servery zajišťují zprávu databází, jejich synchronizaci a také odpovídání na dotazy od uživatelů. Více o DNS serverech naleznete v další kapitole 2.1.3.

Jelikož jsou tyto databáze veliké, vzniklo specifické uložení daného doménového jména. Toto uložení napomáhá lepšímu a efektivnímu vyhledávání. Systém DNS tedy tvoří databáze hierarchicky uspořádaná jako kořenový strom doménových jmen. Tato struktura velice připomíná uspořádání adresářové struktury Unixových systémů. Z pohledu algebry se jedná o acyklický graf. Příklad uspořádání doménových jmen je možné si prohlédnout na obrázku 2.2. Jednotlivé části domény mohou mít až 63 znaků. Celková délka doménového jména je maximálně 255 znaků a může být rozdělena až na 127 úrovní.



Obrázek 2.2: Ukázka uspořádání doménových jmen v DNS [28]

Mezi základní služby, které poskytuje systém DNS, patří:

- Překlad doménových adres na IP adresy (s využitím záznamů typu A pro IPv4, AAAA pro IPv6).
- Překlad IP adres na doménové adresy (s využitím záznamů typu PTR).
- Překlad tzv. aliasů jmen počítačů na tzv. kanonické doménové jméno (s využitím záznamů CNAME).
- Určení poštovního serveru pro danou doménu (s využitím záznamů typu MX).
- Sdílení informace v rámci globálního prostoru jmen.

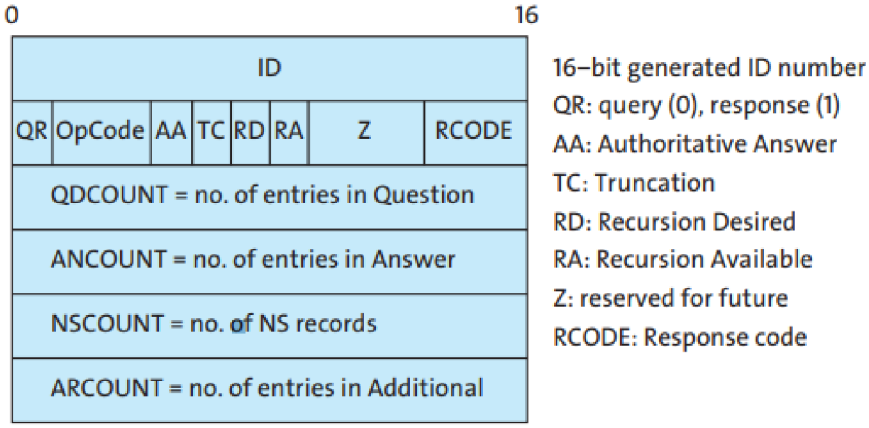
2.1.3 DNS server

DNS server je aplikace, která se stará o data z prostoru doménových jmen. Jak už bylo zmíněno výše, prostor doménových jmen je příliš velký proto, aby se uchovávala všechna data na jednom místě. Proto se tento prostor rozděluje do zón. Tyto zóny jsou dále rozdělené mezi jednotlivé servery DNS. Hlavním úkolem těchto serverů je odpovídat na dotazy. Dotaz může pocházet buď od klienta nebo jiného serveru. Autoritativní servery odpovídají na dotazy pouze z dat, která byla nakonfigurována původním zdrojem, mezi které patří správce domény, nebo je získal pomocí dynamických DNS metod. Autoritativní servery se dále dělí na dva typy, a to primární a sekundární. Servery, které nejsou autoritativní uchovávají kopie částí různých autoritativních serverů, ale jen po určitou dobu, pak je zahazují a aktualizují. Těmto serverů se říká rekurzivní (caching only) servery.

2.1.4 Přenos dat a komunikace v DNS

Pro přenos DNS paketů po síti se využívá transportního protokolu UDP. Tento protokol nezajišťuje bezpečné doručení, což znamená že DNS server nepotvrzuje přijetí daného DNS

paketu. To nám však vůbec nevadí, protože se jedná o rychlou komunikaci a pokud nedostaneme do určitého času odpověď, pošleme daný paket znovu. Maximální velikost paketu je omezena standardem na 512 bytů. Pokud chceme poslat delší zprávu musíme ji rozdělit do více paketů. Pro přenos dat mezi jednotlivými zónami je transportní protokol UDP nahrazen TCP, který zaručí, že citlivá data budou doručena. Na následujícím obrázku 2.3 si můžete prohlédnout hlavičku DNS paketu. Veškerá komunikace v systému DNS je popsána standardem RFC 1035 [15].



Obrázek 2.3: Hlavička paketu DNS [28]

2.2 Protokol SSL/TLS

Protokoly Secure Sockets Layer (SSL) a Transport Layer Security (TLS) se starají o zabezpečení, lépe řečeno o šifrování zpráv při komunikaci mezi klientem a serverem. Tyto protokoly tedy zabraňují odposlouchávání či falšování zpráv v internetové komunikaci. Poskytují zabezpečení pro internetové služby jako WWW, elektronická pošta, internetový fax a mnoho dalších. Jedná se tedy o dva protokoly, které spolu úzce souvisí, jelikož SSL protokol je předchůdce protokolu TLS. Mezi poslední verzí SSL a první verzí TLS, konkrétně SSL 3.0 a TLS 1.0, jsou jen drobné rozdíly, tedy jedná se o skoro totožné protokoly [18] [35] [20].

2.2.1 Historie SSL/TLS

Historii těchto protokolů jsme schopni dopátrat, až k vzniku využívání internetu širokou veřejností. Už v první polovině 90. letech, kdy vznikal veřejný internet bylo jasné, že některé zprávy je zapotřebí šifrovat. Jednalo se původně jen o spojení protokolem HTTP, který dříve nejvíce využívaly banky. Kvůli těmto, ale i dalším důvodům vyvinula společnost Netscape protokol Secure Sockets Layer (SSL). Cílem bylo vytvořit zašifrovanou komunikaci mezi serverem a klientem, která bude nezávislá na operačním systému. SSL bylo vydáno v roce 1994, ale po pěti letech už začalo být nahrazováno novým protokolem TLS. Tento protokol nyní vydává a standardizuje IETF (Internet Engineering Task Force). Protokol TLS je tedy už standardizovaný a může ho využít jakákoliv společnost. Všechny jeho verze se dají nalézt v RFC standardech. Navíc TLS protokol již počítá s porty POP3, SMTP a IMAP [35] [20] [8].

2.2.2 Verze protokolu SSL/TLS

Jak už bylo zmíněno v předchozích kapitolách, oba protokoly mají více verzí. Liší se především tím, že s každou novou verzí je zabezpečení komunikace bezpečnější a hůře prolomitelné. Hlavními rozdíly při přechodu z SSL na TLS je standardizace a rozšíření protokolů o nové porty, které jsou zmíněny v předchozí kapitole. Pro přehlednost je možné si v následující tabulce 2.1 prohlédnout všechny aktuálně vydané verze včetně RFC, jimiž jsou popsány [8] [21].

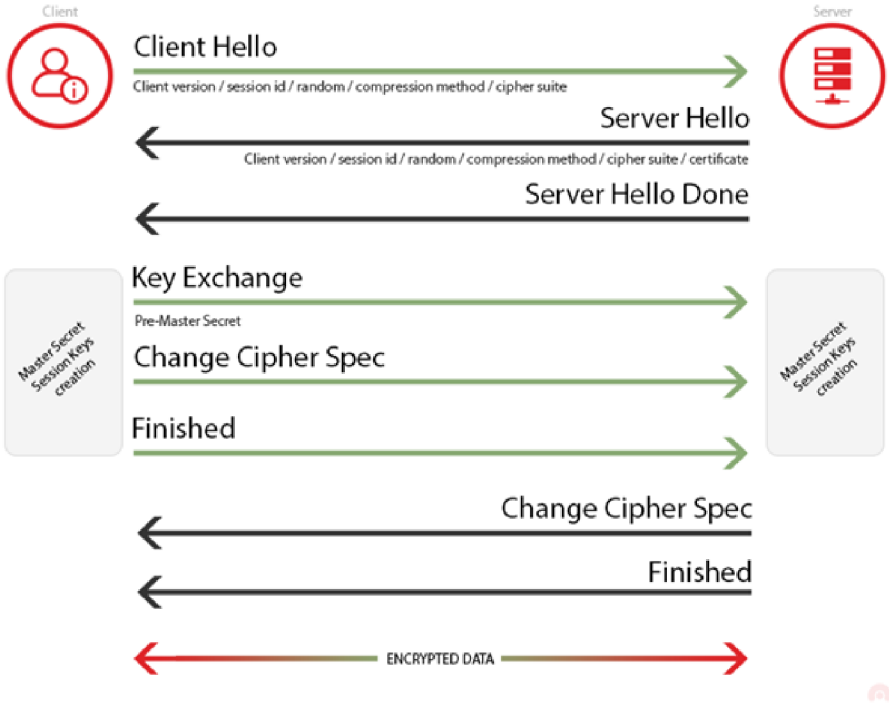
Protokol	Vydán	Odmítnut	Poznámka
SSL 1.0	Neuveřejněný	Neuveřejněný	Nebyla vydána oficiální verze
SSL 2.0	1995	2011	Zakázáno v RFC 6176
SSL 3.0	1996	2015	Publikováno v RFC 6101. Ukončeno v RFC 7568
TLS 1.0	1999	2020	Definováno v RFC 2246
TLS 1.1	2006	2020	Definováno v RFC 4346
TLS 1.2	2008	-	Definováno v RFC 5246
TLS 1.3	2018	-	Definováno v RFC 8446

Tabulka 2.1: Verze protokolu SSL/TLS

2.2.3 Komunikace v protokolech SSL/TLS

V této kapitole jsem čerpal ze zdrojů: [20] [19] [17].

Komunikace v protokolech SSL/TLS je na rozdíl od DNS bezpečná a zajišťuje ji protokol TCP. Proto je komunikace oproti DNS složitější a nejlépe si jí ukážeme na diagramu 2.4. V Diagramu je vynechán takzvaný 3 way handshake běžně používaný v protokolu TCP, který komunikaci předchází.



Obrázek 2.4: Diagram komunikace v protokolu TLS [19]

Nyní si pojďme blíže popsat jednotlivé zprávy z diagramu:

1. Client Hello (klient -> server)

   - Na začátku komunikace pošle klient paket z následujícími informacemi:
     - Verze klienta (Client Version) - klient odešle seznam všech verzí protokolu TLS / SSL, které podporuje, přičemž preferovaná je první na seznamu. Preferována verze bývá z pravidla ta nejnovější.
     - Klientovo náhodné číslo (Client Random) - tato informace je 32byte náhodné číslo. Vygenerované číslo jak u klienta, tak na serveru, se později používá k vygenerování klíče pro šifrování.
     - Číslo spojení (Session ID) - číslo, které označuje dané spojení mezi klientem a serverem.
     - Metoda komprese (Compression Methods) - toto je seznam metod, které budou použity pro kompresi dat, a to před jejich šifrováním. Použitím komprese můžeme dosáhnout nižšího využití šířky pásma, a tedy i vyšších přenosových rychlostí.
     - Šifrovací sady (Cipher Suites) - šifrovací sady jsou kombinací kryptografických algoritmů. Každá šifrovací sada obvykle obsahuje jeden kryptografický algoritmus pro každou z následujících úkolů: výměna klíčů, ověřování, hromadné (datové) šifrování a ověřování zpráv. Klient odešle seznam všech šifrovacích sad, které podporuje, v pořadí podle preferencí. To znamená, že klient by v ideálním případě preferoval připojení navázané pomocí první odeslané šifrovací sady.

- Dále tento paket obsahuje volitelná rozšíření. Například se může jednat o tato rozšíření: Server Name, Status Request, Supported Groups, EC Point Formats, Signature Algorithms a pár dalších. Názvy rozšíření jsou zde uvedena v Anglickém jazyce jelikož po překladu by některé názvy ztrácely smysl.

2. Server Hello (server -> klient)

   - Poté co server příjme od klienta paket "Client Hello"na něj odpoví svým Hello paketem. Server Hello paket obsahuje buď vybrané možnosti (z těch, které byly navrženy během Client Hello), nebo obsahuje zprávu o selhání při pokusu navázat spojení. Tento paket obsahuje následující informace:
     - Verze serveru (Server Version) - server vybere preferovanou verzi protokolu SSL / TLS z těch, které předložil klient.
     - Náhodné číslo serveru (Server Random) - stejně jako u klienta se jedná 32byte náhodné číslo. Toto vygenerované číslo se později používá k vygenerování klíče pro šifrování.
     - Číslo spojení (Session ID) - pokud číslo spojení, které dostal server od klienta nebylo prázdné, server vyhledá dříve uložené relace v mezipaměti (cache) a pokud nalezne shodu, použije se toto Session ID k obnovení relace. Pokud bylo Session ID klienta prázdná, server vytvoří novou relaci a odešle ji klientovi.
     - Metoda komprese (Compression Methods) - server vybere jednu z metod, které mu zaslal klient.
     - Šifrovací sady (Cipher Suites) - server vybere šifrovací sadu ze sad zaslaných v Client Hello paketu.

3. Server Certificate (server -> klient)

   - Server nyní odešle podepsaný certifikát TLS / SSL, který prokazuje jeho totožnost klientovi. Tento paket také obsahuje veřejný klíč serveru.

4. Client Certificate (klient -> server)

   - Ve vzácných případech může server vyžadovat, aby byl klient ověřován pomocí klientského certifikátu. Pokud ano, klient poskytne serveru podepsaný certifikát. Tato zpráva je volitelná, což znamená, že není vždy vyžadována.

5. Server Key Exchange (server -> klient)

   - Paket o výměně klíče server odešle pouze v případě, že předchozí certifikát poskytnutý serverem nestačí, aby si klient mohl vyměnil informace se serverem o takzvaném "Pre-Master Secret". Pre-Master Secret bude blíže vysvětleno ke konci této kapitoly.

6. Server Hello Done (server -> klient)

   - Tato zpráva slouží k identifikaci konce sekvence "Server Hello" paketů.

7. Client Key Exchange (server -> klient)

- Zpráva Client Key Exchange je odeslána ihned poté, co klient příjme zprávu Server Hello Done. Pokud server požaduje klientský certifikát, odešle se poté výměna klientských klíčů. Během této fáze klient vytvoří pre-master klíč.

8. Client Change Cipher Spec (klient -> server)

    - V tomto okamžiku je klient připraven k přechodu do zabezpečeného šifrovaného prostředí. Protokol Change Cipher Spec se používá ke změně šifrování. Veškerá data odesílaná klientem od této zprávy budou šifrována pomocí symetrického sdíleného klíče.

9. Client Handshake Finished (klient -> server)

    - Poslední zpráva, která udává, že nyní má klient potřebné informace ke komunikaci v šifrované formě. Toto je také první šifrovaná zpráva zabezpečeného připojení.

10. Server Change Cipher Spec (server -> klient)

    - Server je také připraven přepnout do šifrovaného prostředí. Všechna data odesílaná serverem od nynějška budou šifrována pomocí symetrického sdíleného klíče.

11. Server Handshake Finished(server -> klient)

    - Nyní je i na straně serveru vše potřebné k šifrované komunikaci. Toto je první šifrovaná zpráva ze strany serveru.

Pre-Master Secret je tajemství vytvořeno klientem (způsob vytvoření závisí na šifrovací sadě) a poté je sdílen se serverem. Před odesláním tajemství na server jej klient zašifruje pomocí veřejného klíče serveru extrahovaného z certifikátu poskytnutého serverem. To znamená, že zprávu může dešifrovat pouze server.

Pre-Master Secret je zapotřebí k tomu, abychom mohli vytvořit Master Secret, neboli hlavní tajemství, které slouží jako klíč. Poté, co server přijme tajný klíč "Pre-Master Secret", dešifruje jej soukromým klíčem. Nyní klient a server vypočítávají hlavní tajný klíč (master secret key) na základě náhodně vyměněných hodnot (Client Random a Server Random) pomocí pseudonáhodné funkce (PRF). PRF je funkce používaná ke generování libovolného množství pseudonáhodných dat. Hlavní tajný klíč, který je dlouhý 48 bajtů, je pak použit jak klientem, tak i serverem k symetrickému šifrování dat pro zbývající část komunikace.

Kapitola 3

Techniky pro předzpracování dat

Před použitím výše zmíněných metod je zapotřebí nasbírat nějaká data, která chceme analyzovat. Data, která však získáme ať už z nějakých senzorů, či napsaná ručně lidmi, mají často podobu, která nám nevyhovuje. Jsou například neúplná nebo obsahují chyby, se kterými by se algoritmy pro dolování dat jen těžko vyrovnávaly. Pokud čerpáme vstupní data z více zdrojů, je také možné, že mají jiný formát. Nejen kvůli těmto důvodům je potřeba vstupní data upravit, odstranit z nich chyby, případně zredukovat jejich počet. Máme-li totiž málo kvalitní vstupní data dostáváme i méně kvalitní výsledek.

3.1 Čištění dat

Prvním metodou, dalo by se říci i krokem pro předzpracování dat, je jejich čištění. Čištění dat nebo anglicky data cleansing, či data cleaning je proces, při kterém mažeme, upravujeme nebo opravujeme vstupní data. Tato data bývají často neúplná, nesprávná anebo duplikovaná. Čištění dat se provádí buď jednorázově, kdy před zahájením dolování data vyčistíme. V jiných případech se data čistí průběžně, tyto případy bývají nejčastěji při vkládání nových dat do datových sad nebo před jejich ukládáním do databází. Čištění dat můžeme provádět buďto ručně nebo automaticky. K automatickému čištění dat je k dispozici velké množství nástrojů. Z pravidla je nejlepší čistit data co možná nejblíže k jejich zdroji, aby se nám chybná data nešířila moc daleko systémem. Cílem čištění je tedy zlepšení konzistence dat a oprava chyb, které se v nich mohou nalézat [23].

3.2 Redukce dat

V některých datových skladech, které se snažíme zpracovat, se nachází velké množství dat, které chceme zpracovat. Analýza všech těchto dat by byla velice náročná ať už časově nebo výpočetně. Proto chceme objem vstupních dat co možná nejvíce zredukovat, ale tak, aby byl zachován co možná nejpřesnější výsledek. Dalším důvodem pro redukci dat je velké množství informací v databázi, kde nás zajímá jen jejich část. K těmto účelům slouží různé strategie pro redukci dat. Mezi nejznámější patří:

- Redukce dimenzionality
- Komprese dat
- Redukce datových objektů (shlukování, vzorkování)

Kapitola 4

Základní úlohy dolování dat (data mining)

Proces dolování dat za účelem odhalení skrytých spojení a předpovídání budoucích trendů na trhu má dlouhou historii. Někdy se tento proces nazývá „získávání znalostí v databázích“, pojem „dolování dat“ byl vytvořen až v 90.let 20. století. Samotné dolování dat zahrnuje tři vzájemně propojené vědecké disciplíny: statistiku (numerické studium datových vztahů), umělou inteligenci (inteligence podobná člověku zobrazovaná pomocí softwaru a / nebo strojů) a strojové učení (algoritmy, které se mohou z dat učit, aby předpovídaly). Technologie dolování dat se neustále vyvíjí, aby udržovala krok s neomezeným potenciálem velkých datových skladů a dostupným výpočetním výkonem, který neustále roste. Čím komplexnější jsou nashromážděná data, tím větší je potenciál odhalit relevantní informace, které hledáme. Dolování dat má velké využití v dnešním světě. Maloobchodníci, banky, výrobci, poskytovatelé telekomunikačních služeb, pojišťovna a spousty dalších, využívají dolování dat k objevování vztahů mezi různými daty, od cenotvorby, propagace a demografie až po vliv ekonomiky, rizika, konkurence a sociálních médií na jejich obchodní modely, příjmy, operace a vztahy se zákazníky [4] [22].

Nyní se zaměříme na základní úlohy dolování dat, mezi které patří:

- klasifikace dat
- asociační pravidla
- regresní analýza
- shluková analýza
- vyhledávání dokumentů

4.1 Klasifikace dat

Klasifikací dat, jak už samotný název hovoří, se rozumí rozdělení dané skupiny (množiny) objektů, jevů či procesů do určitého počtu menších skupin (podmnožin), které mají podobné vlastnosti. Tyto vlastnosti musí být dostatečně rozlišné, aby bylo možné správně vstupní data klasifikovat.Pomocí těchto vlastností vytváříme klasifikační kritéria, která vytvoří samotnou klasifikaci. Objekty, které mají společné vlastnosti, nám utváří klasifikační

třídu. Každá klasifikace musí být úplná, čili každý objekt musí patřit do jedné klasifikační třídy, ale zároveň nemůže být ve dvou či více klasifikačních třídách [26].

Samotný proces klasifikace dat se dělí do dvou kroků. Prvním z nich je tzv. učení. Toto učení provádí tvorbu klasifikačního modelu, kdy tento model je schopný klasifikovat data za pomoci cvičných dat. Tato data se skládají ze vzorku dat, u kterých známe klasifikační třídu, do které patří, a zkoumáme, jestli je klasifikace provedena správně. Druhým krokem je samotná klasifikace dat, kdy používáme vytvořený model z prvního kroku ke klasifikaci nových dat [34].

Před samotnou klasifikací dat se často provádí úprava nad daty. Jedná se o čištění, určení relevantnosti dat, či jejich transformace. Tyto metody jsou již popsané v předchozí kapitole 3.

Dále se podíváme na dva základní typy klasifikačních modelů. Tyto modely patří k těm nejznámějším a modelům klasifikačních úloh a jedná se o rozhodovací stromy a neuronové sítě.

4.1.1 Rozhodovací stromy

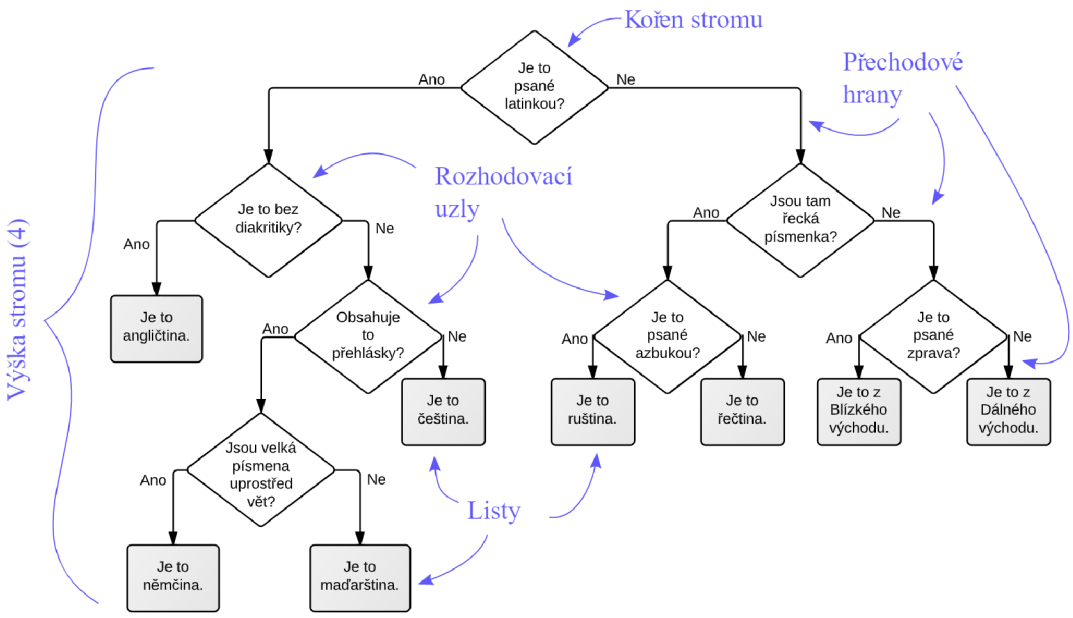
V této kapitole jsem čerpal ze zdrojů: [14] [38].

Jak už bylo zmíněno výše, Rozhodovací stromy patří k základním úlohám pro klasifikování dat. Jedná se o stromovou strukturu, která znázorňuje rozhodovací proces. Rozhodování o dané informaci začíná v kořeni a jde skrze jednotlivé rozhodovací uzly, které nám určují stále přesněji kam daná informace patří, dokud nedojdeme skrze přechodové hrany až k listu. Listy nám dají konečnou odpověď na námi položenou otázku, například o jaký jazyk se jedná (tento rozhodovací strom je znázorněn v obrázku 4.1). Princip odpovědí na otázky v daných uzlech funguje tím způsobem, že ke všem existujícím datovým záznamům zvolíme jeden cílový atribut. Tento atribut bývá zpravidla v binární podobě, což značí, že nabývá pouze dvou hodnot. Hodnoty mají zpravidla úplně opačný význam, aby se dalo bez problému rozhodnout, kterým směrem se v cestě rozhodovacím stromem dále vydat. Nejčastěji se používají hodnoty ANO a NE. Takto projdeme celý strom až do listů, čímž klasifikujeme vstupní data.

Rozhodovací strom je také možné postupně upravovat, a to s využitím strojového učení. Přesněji řečeno se sestavuje strom nový, a to na základě předchozích vyhodnocení. Strom se tedy naučí na datech, které již zpracoval, a tím získává lepší a přesnější výsledky. Přesněji se této problematice říká učení s učitelem. Tyto metody se často používají v internetových obchodech, kde se snaží obchodníci přijít na to, proč zákazníci opouští jejich web a nakupují stejné zboží u konkurence. Rozhodovací stromy však nemusí rozhodovat jen o binárních problémech, je tedy možné vytvořit strom, který odpověď na danou otázku klasifikuje do více jak dvou kategorií. Pokud chceme klasifikovat spojitou veličinu, pak pro tento problém existuje rozšíření zvané regresní strom.

Některé příklady využití rozhodovacího stromu v praxi:

- včasné odhalení odcházejícího zákazníka
- detekce spamu pro příchozí emaily
- nalezení podezřelých bankovních transakcí
- odhad konverzní míry nových produktů
- určení kombinace faktorů, které mají klíčový vliv na sportovní výkon

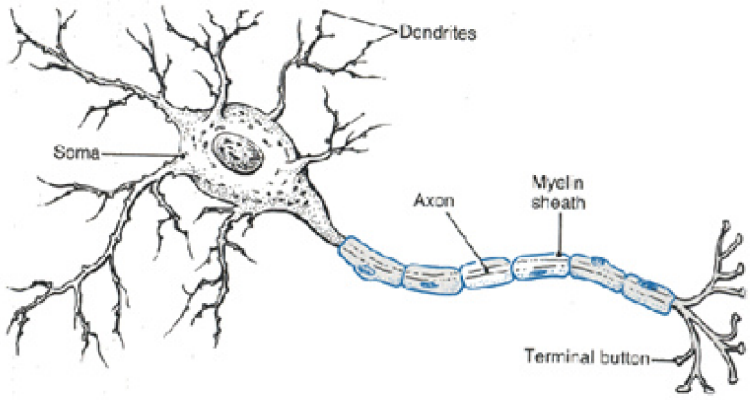


Obrázek 4.1: Rozhodovací strom, který zkoumá jakým jazykem je text napsaný [14]

4.1.2 Neuronové sítě

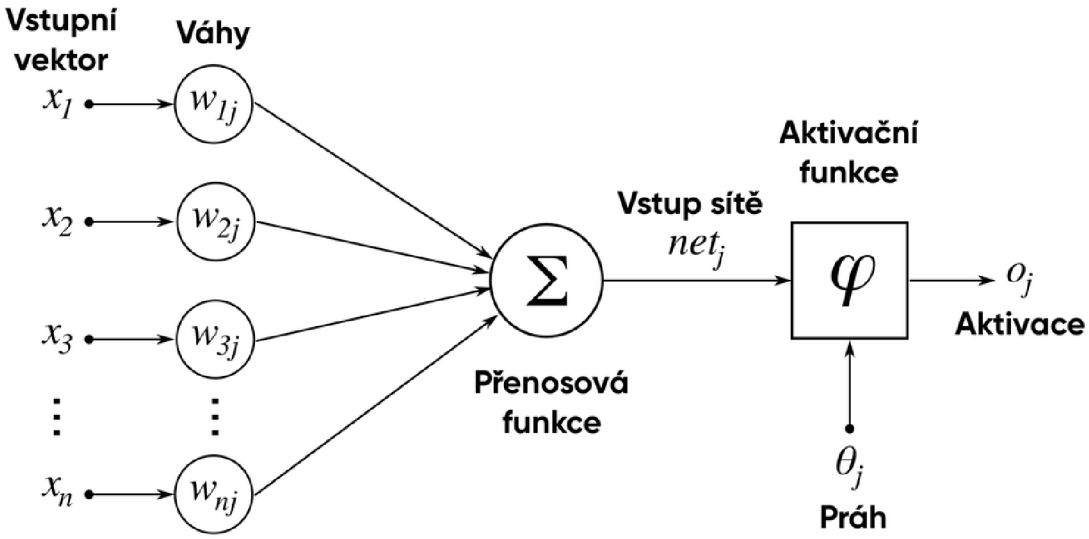
V této kapitole jsem čerpal ze zdrojů: [2] [3] [9] [11].

Na začátek této kapitoly se něco málo povíme o Neuronových sítích z pohledu biologie. Neuronové sítě se skládají, jak už název napovídá, z neuronů. Neuron je buňka, která v živých organizmech vede informační signály a umožňuje nám na ně reagovat. Neurony obsahují dendrity, které neuronům umožňují přijímat signály z okolního světa. Dále obsahují axon, díky němuž neurony signály vysílají. Výsledný signál, který neuron vysílá je většinou nějakým způsobem upravený o novou informaci. Samotný axon je na konci rozvětven do více výstupů, aby mohl svůj signál předat většímu množství příjemců. Nervová soustava lidského těla obsahuje něco mezi 10^{11} až 10^{12} neuronů. Počet neuronů každého lidského těla se s přibývajícím věkem snižuje. Neurony mohou mít stovky až tisíce dendritů, kterými jsou vzájemně propojeny. Po těchto informacích jsme tedy schopni zjednodušeně říci, jak z biologického hlediska fungují neurony. Pro úplnost to ještě shrneme do jedné věty. Neurony přijímají signály pomocí dendritů, tento signál se šíří až dovnitř buňky, kde vznikne potenciál, který při dosažení dostatečné hodnoty pošle neuron dál. Neuronové sítě mají široké využití a nedají se z hlediska dolování dat zařadit úplně pod přesnou kategorii, jelikož existují i některé varianty neuronových sítí pro detekci odchylek, a dala by se tak zařadit i pod Regresní analýzu.



Obrázek 4.2: Neuron z pohledu biologie [2]

Nyní se pojďme blíže seznámit s neuronem umělým. V některých starších článcích se můžeme setkat i s pojmem perceptron (označení matematického modelu biologického neuronu), jedná se však o jiný název pro neuron. Umělý neuron mívá z pravidla jeden, až několik vstupů a jeden výstup. Vstupy jsou chápány stejně jako v případě biologického neuronu, a to tedy jako podměty z vnějšího okolí nebo výstupy z jiných neuronů, pokud se jedná o neuronovou síť. Každý vstup je ohodnocen váhu daného vstupu. Součástí neuronu je jeho prahová hodnota neboli potenciál neuronu. Při překonání potenciálu je neuron nabuzen k vyslání vlastního signálu. Výstupní signál je ve formě přenosové funkce. Přenosová funkce bývá také nazývána funkcí přechodu nebo aktivační funkce.



Obrázek 4.3: Model umělého neuronu [12]

Nyní se podíváme na popis tohoto umělého neuronu z matematického hlediska. Vstupní signály si označíme jako x_1, x_2 až x_n a výstupní signál si označíme jako y. Neuron jsme matematicky schopni popsat následujícími rovnicemi:

1. Zachycení vstupních signálů, jejich součet s váhami pro daný vstup, čímž nám vznikne potenciál P:

- P = $w_1 * x_1 + w_2 * x_2 + ... + w_n * x_n$

2. Jakmile je potenciál P dostatečně velký neuron vyšle výstupní signál y:

   - y = 1 , jestliže P > w_0, jinak y = 0

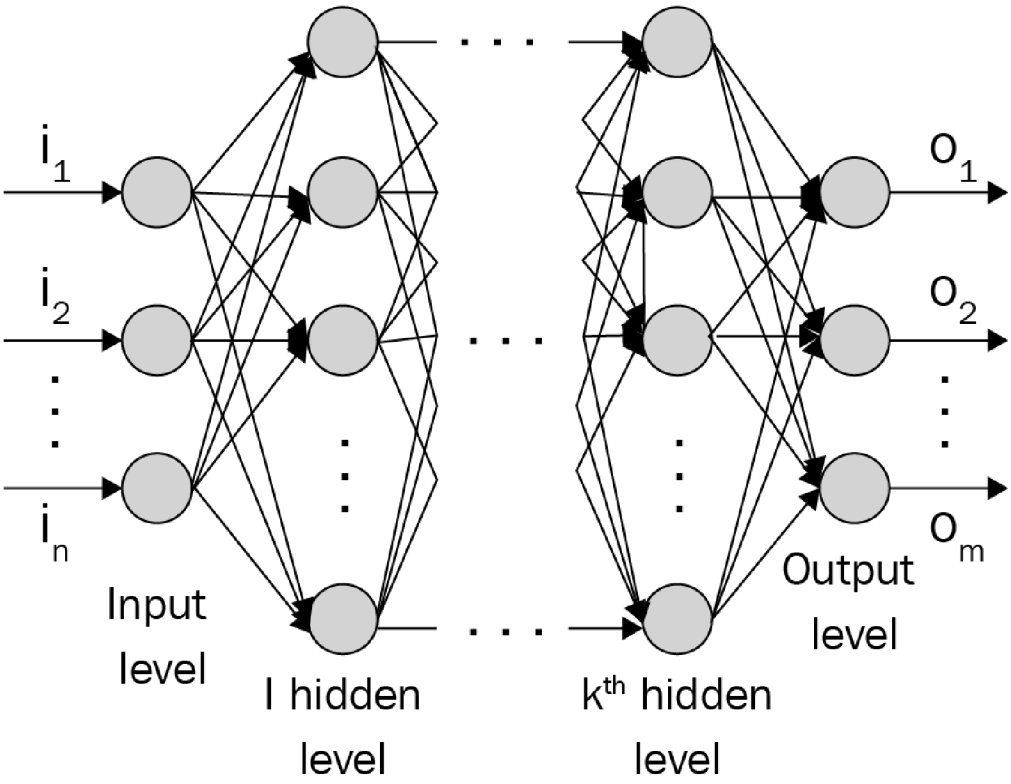
Podmínku P > w_0 můžeme přepsat za pomoci aktivační funkce f(P), kdy je výstupní signál neuronu roven jedné. V případě použití jiné funkce bude signál typicky hodnota z intervalu <0,1>. Funkce neuronu se dá také zapsat jednou matematickou rovnicí, kde w_0 bude obsahovat zápornou hodnotu, která představuje práh, který je potřeba překonat potenciálem P. Aktivační funkce bude poté vypadat následovně:

- y = f($w_0 + w_1 * x_1 + w_2 * x_2 + ... + w_n * x_n$)

Aktivační funkce by měla být nelineární. Můžeme proto využít širokou škálu funkcí, které mají nelineární průběh, mezi nejčastěji používanými aktivačními funkcemi je například funkce sigmoid nebo hyperbolický tangens.

Z předchozích odstavců je zřejmé, že neuron je schopen rozdělit prostor pouze na dva poloprostory. Ve složitějších úlohách je ale potřeba tento prostor rozdělit do více částí. Proto pro řešení složitějších problémů je zapotřebí sestavit neuronovou síť. Neuronová síť, jak už název napovídá, je zapojení více neuronů za sebou či pod sebou, čímž nám vznikne síť propojených neuronů. Jedním z nejpoužívanějších zapojení neuronů do sítě je feed-forward zapojení. Toto zapojení obsahuje několik vrstev neuronů, které jsou vzájemně propojeny každý s každým. Pojďme se tedy na toto zapojení podívat trošku blíže v dalším odstavci.

V zapojení neuronů do sítě feed-forward řeší každý neuron pouze část výsledného problému. Výsledek je tedy součet neboli sloučení všech dílčích výsledků menších podproblémů do jednoho řešení celé sítě. Toto řešení nalezneme i v reálném životě, a proto je velmi vhodné i pro řešení některých úloh. Přestane-li nějaký neuron v síti fungovat, výsledek sice ztratí na přesnosti, ale stále jsme schopni najít řešení zadaného problému. Na následujícím obrázku 4.4 si můžeme všimnout zapojení neuronů do tohoto typu sítě. Když se zaměříme na první vrstvu neuronů, tak si můžeme povšimnout, že tato vrstva pouze rozdistribuuje signál na veškeré neurony další vrstvy čili neprovádí se zde žádný výpočet. Další vrstvy už počítají svůj dílčí problém. Postup je v každém neuronu totožný a shoduje se s výpočtem, který byl zmíněn v předchozích odstavcích této kapitoly. Pro úplnost ho zde ale ještě shrneme. Každý neuron sečte své váhové vstupy a vzniká v něm potenciál, pokud je potenciál dostatečně velký vyšle výstupní signál. Tento signál se dále šíří sítí až do poslední vrstvy. Tuto vrstvu označujeme jako výstupní. Výstupní vrstva má z pravidla stejný počet neuronů jako je počet tříd do kterých je možné výsledek rozdělit. Dáme li příklad, že chceme určit z posloupnosti čísel medián a v posloupnosti jsou čísla od 1 do 10, bude mít výstupní vrstva 11 neuronů. Který z neuronů ve výstupní vrstvě bude mít po zpracování posloupnosti na konci hodnotu jedna nám řekne jaké číslo je medián. Je potřeba však předem určit jakou hodnotu z posloupnosti nám značí daný neuron ve výstupní vrstvě.



Obrázek 4.4: Neuronová síť v zapojení feed-forward [1]

4.2 Asociační analýza

V této kapitole jsem čerpal ze zdrojů: [36] [24] [5].

Z takřka všech programovacích jazyků známe syntaxi IF-THEN. Není tedy vůbec zvláštní, že tato konstrukce se dá využít i v případě dolování dat a patří společně s rozhodovacími stromy k těm nejčastěji využívaným prostředkům. Asociační analýza se snaží najít vzory, nebo lépe řečeno vztahy, ukryté v datovém souboru. Pomocí vzorů pak vytváří asociační pravidla, která popisují společně se vyskytující atributy. Asociační pravidla se dostala na výsluní počátkem 90. let, a to díky Rakeshi Agrawalovi, který začal asociační pravidla používat pro analýzu nákupního košíku. Při této analýze zkoumáme nákupní košík běžného člověka v supermarketu. Hlavním předmětem zkoumání je především souvislost mezi jednotlivými položkami v nákupním košíku. Jako příklad nám může posloužit párek, ke kterému nejčastěji koupíme pečivo, případně kečup nebo hořčici. Jde nám tedy o zkoumání asociací (vazeb) mezi zbožím, které se v supermarketu, či v online obchodě nabízí. Asociační pravidla se zapisují následujícím způsobem $A \Rightarrow B$, kde A i B jsou množiny hodnot (v případě nákupního košíku je to zboží).

Z datového balíku jsme schopni vygenerovat obrovské množství pravidel. Všechna pravidla však nejsou pro daný problém, který chceme pomocí asociačních pravidel řešit, podstatná. Zajímá nás tedy, kolik příkladů splňuje předpoklady a kolik závěr daného pravidla. Kolik jich splňuje jak závěr, tak předpoklad nebo ani jedno. Tyto počty však není vůbec složité určit, a to díky kontingenční tabulce. Máme tedy pravidlo $Ant \Rightarrow Suc$, kde Ant (předpoklad, levá strana pravidla, antecedent) a Suc (závěr, pravá strana pravidla, sukcedent), kde pro n případů vypadá kontingenční tabulka 4.1 následovně:

- $n(Ant \wedge Suc) = a$ je počet objektů, které splňují současně předpoklad i závěr.
- $n(Ant \wedge \neg Suc) = b$ je počet objektů, které splňují předpoklad a nesplňují závěr.
- $n(\neg Ant \wedge Suc) = c$ je počet objektů, které nesplňují předpoklad, ale splňují závěr.
- $n(\neg Ant \wedge \neg Suc) = d$ je počet objektů, které nesplňují předpoklad ani závěr.
- Další hodnoty které naleznete v tabulce 4.1 se vypočítají následujícím způsobem:
  - $n(Ant) = a + b = r$
  - $n(\neg Ant) = c + d = s$
  - $n(Suc) = a + c = k$
  - $n(\neg Suc) = b + l = l$
  - $n = a + b + c + d$

	Suc	$\neg$ Suc	$\sum$
Ant	a	b	r
$\neg$ Ant	c	d	s
$\sum$	k	l	n

Tabulka 4.1: Kontingenční tabulka

Ze získaných hodnot, které vypočítáme z výše zmíněných rovnic sice nejsme schopni ještě nic pořádného využít, ale tyto výsledky nám pomohou k výpočtu různých charakteristik pravidel, díky nímž dosáhneme znalostí. Základní charakteristiky podle Rakesho

Agrawala jsou podpora (support) a spolehlivost (confidence). Dále se však mezi základní charakteristiky také řadí i pokrytí (coverage) a kvalita (quality). Pojďme si teď tyto základní charakteristiky popsat matematicky:

- Podpora
  - Vyjadřuje počet objektů splňujících jak předpoklad, tak i závěr. Tuto metriku je možné měřit jak v absolutní, tak i v relativní podobě.
  - A respektive $\mathrm{P}(\mathrm{Ant} \wedge \mathrm{Suc}) = \frac{a}{a+b+c+d}$
- Spolehlivost (nazývaná se též platnost – validity, či konsistence – consistency, nebo správnost – accuracy)
  - Jedná se o podmíněnou pravděpodobnost závěru, pokud platí předpoklad.
  - $\mathrm{P}(\mathrm{Ant}|\mathrm{Suc}) = \frac{a}{a+b}$
- Pokrytí
  - Vyjadřuje podmíněnou pravděpodobnost předpokladu, pokud platí závěr.
  - $\mathrm{P}(\mathrm{Ant}|\mathrm{Suc}) = \frac{a}{a+c}$
- Kvalita
  - Jedná se o vážený součet spolehlivosti a pokrytí.
  - Kvalita $= w_1 * \frac{a}{a+b} + w_2 * \frac{a}{a+c}$

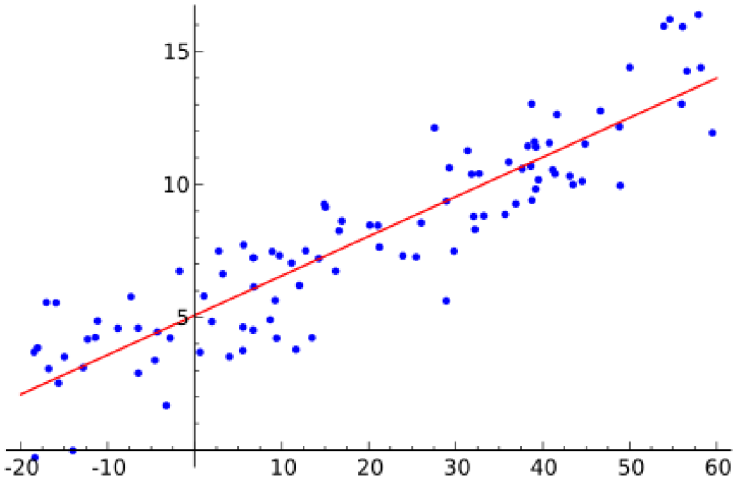
4.3 Regresní analýza

V této kapitole jsem čerpal ze zdrojů: [13] [10] [33].

Regresní analýza nám umožňuje zkoumat vztah mezi proměnnými. Prvním typem proměnné v regresní analýze je tzv. nezávislá proměnná X, kterou též nazýváme regresand nebo cílová proměnná. Druhým typem je tzv. závislá proměnná Y, kterou můžeme nazvat též regresor. Díky regresní analýze jsme schopni určit, jak se změní hodnota závislé proměnné v případech, kdy změníme hodnotu jedné z nezávislých proměnných. Ostatní nezávislé proměnné si však zachovávají konstantní neboli neměnnou hodnotu. Konečný výsledek pak závisí na regresní funkci. Regresní funkce nám vytváří křivku, kterou prokládáme vstupní data. Touto křivkou chceme co nejlépe vystihnout vztah mezi naměřenými hodnotami. V nejjednodušších případech se jedná o lineární regresi, kdy naměřené hodnoty vložíme do grafu a za pomoci regresní funkce do grafu vložíme přímku, z které lze vysledovat vztah mezi naměřenými hodnotami. Pro výběr regresní funkce se používá aproximační metoda nejmenších čtverců. U složitějších problémů nemusí být však funkce pouze lineární, můžeme se setkat s kvadratickou, kubickou či logaritmickou. Na následujícím obrázku 4.5 si můžeme prohlédnout lineární regresní analýzu vyobrazenou v grafu.

Pro složitější případy se využívá vícenásobná neboli mnohonásobná regrese. Tato regrese na rozdíl od jednoduché lineární regrese nám vyjadřuje vztah mezi jednou závislou proměnnou a více nezávislými proměnnými. U jednoduché se jedná pouze o jednu závislou a jednu nezávislou. Cílem vícenásobné regrese je tedy vysvětlit rozptyl závislé proměnné. Vypočítáme ho tak, že vypočítáme vliv každé z nezávislých proměnných na jednu proměnnou závislou. Vliv nezávislých proměnných je odhadován tak, že zkoumáme působení ostatních

nezávislých proměnných které vstupují do regresní analýzy.Tato analýza nám také pomáhá určit jaký relativní vliv na výsledek mají jednotlivé nezávislé proměnné. Opět se tedy jedná o rovnici, kde pravá strana rovnice obsahuje naše nezávislé proměnné a koeficienty, které mají vliv na tyto proměnné. Na Levé straně rovnice se nachází závislá proměnná.



Obrázek 4.5: Ukázka výstupu regersní analýzi [10]

4.4 Shluková analýza

V této kapitole jsem čerpal ze zdrojů: [29] [27].

Shluková analýza, jak už název napovídá, se zabývá metodami a algoritmy, pomocí kterých jsme schopni rozdělit vstupní data do shluku. Shluk je skupina (množina) dat s velice podobnými vlastnostmi. Shlukovou analýzu můžeme zařadit do skupiny nástrojů datové analýzy. Snažíme se co možná nejpřesněji vstupní data rozdělit do shluků objektů podobných vlastností tak, že v jednom shluku jsou data s co možná největší podobností. Data nebo objekt z jednoho shluku by měl mít co možná nejmenší shodné vlastnosti oproti objektům z jiných shluků. Díky shlukování jsme schopni určit vztahy mezi objekty bez další potřebné intepretace. Způsob určování shody vlastností daných objektů ve shluku je možné volit různě na základě daného problému, který chceme shlukovou analýzou řešit.

Existuje několik metod shlukové analýzy. U všech těchto metod je základním a prvním krokem určení souboru znaků. Tento soubor znaků nejčastěji vyjadřujeme ve formě vektoru číselných charakteristik. V některých, často složitějších metodách, se místo vektoru číselných charakteristik požívá matice, která vyjadřuje nepodobnost objektů. Na každém místě v této matici vyjadřuje číselnou hodnotou vzdálenosti (nepodobnost) mezi dvěma objekty. Jeden objekt je na svislé linii a druhý na vodorovné.

Shluková analýza se dělí do dvou základních skupin. Prvním je nehierarchická shluková analýza. Výsledkem tohoto druhu analýzy je zařazení objektů do předem zvoleného počtu shluků (skupin). Tyto shluky, jak už bylo řečeno výše, jsou navzájem odlišné a uvnitř shluku jsou si objekty co nejvíce podobné vlastnostmi, které nás zajímají. Mezi nejznámější nehierarchické metody můžeme zařadit algoritmus k-means, k-medoids a metody založené na hustotě. Druhým typem shlukové analýzy je analýza hierarchická. Výsledkem hierarchické shlukové analýzy je tak zvaný dendrogram. Dendrogram nám znázorňuje objekty ve

stromové struktuře, pomocí níž jsme schopni určit vzájemnou příbuznost či nepříbuznost vstupních objektů. U hierarchické shlukové analýzy známe dva druhy způsobu vytváření dendogramu, jedná se buď o aglomerativní shlukování nebo o divizivní shlukování.

4.5 Vyhledávání dokumentů (Information retrieval)

S tendencí rozšiřování internetu začalo čím dál tím více populace hledat informace na Internetu. Pole informací, které se daly a dají na internetu najít, začalo vědeckými publikacemi a knihovními záznamy, ale brzy se rozšířilo i na další formy obsahu. Zejména na ty, kteří pracují s informacemi, jako jsou novináři, právníci a lékaři. Jak se postupně celý web rozšiřoval bylo stále těžší a těžší najít potřebné informace. V dnešní době hledá každý z nás informace na internetu a neobejdeme se bez toho abychom v nich mohli vyhledávat. Právě k těmto účelů a mnoho dalším slouží metody vyhledávání dokumentů. Nás však spíše zajímá jak tyto dokumenty, či webové stránky, rozumným způsobem porovnávat a určovat jejich podobnost. V následujících odstavcích této kapitoly se proto zaměříme na popis metody TF-IDF, které k tomuto účelu slouží [25].

4.5.1 TF-IDF metoda

V této a následujících kapitolách jsem čerpal ze zdrojů: [16] [37] [25].

Tato metoda slouží k hodnocení relevance při vyhledávání dokumentů. Název teto metody je zkratka dvou metod, které se při výpočtu používají. První z nich je Term Frequency, neboli četnost slova v daném dokumentu a druhou je Inverse Document Frequency, která nám vyjadřuje převrácenou četnost daného slova ve všech dokumentech. My tuto metodu využijeme k tomu, abychom určili, jaká je relativní podobnost jednotlivých dvojic zařízení v datových sadách. Díky váze pro jednotlivé hodnoty jsme schopni určit, jak důležitá daná informace pro nás je a jak moc ji máme brát v potaz při hledání našeho zařízení v jiných datových sadách. Pojďme se tedy podívat postupně na obě metody a nakonec na to, jak se spojí v metodu TF-IDF, která nás zajímá. Dále se podíváme na to jak se z hodnot, které nám vrátí metoda TF-IDF získá jediná hodnota, která nám určí jak moc si jsou zařízení podobná.

Frekvence termů (Term Frequency)

Frekvence termů, jak už bylo zmíněno výše je počet výskytů daného slova v dokumentu. Jako nejjednodušší variantou získání této hodnoty je spočítat kolikrát se daná hodnota vyskytuje v dokumentu. Pro případy, kdy jsou oba dokumenty zhruba stejně velké, tento nejpřirozenější postup bohatě postačí. Máme však i další druhy výpočtu této hodnoty, které nám umožňují do výsledné hodnoty započíst délku textu daného dokumentu, či upravit frekvenci. Díky frekvenci termů jsme tedy schopni spočítat důležitost výskytu daného slova pro jeden dokument. Čím je hodnota nižší tím méně časté je toto slovo v daném dokumentu. Pojďme si shrnout základní varianty pro výpočet hodnoty frekvence termů:

- Binární - nabývá pouze hodnoty 1 nebo 0, v závislosti na tom jestli se daný term v dokumentu vyskytuje, či ne.
- Počet výskytů - výsledná hodnota se rovná počtu výskytů daného termu v dokumentu

- Frekvence termů - tuto variantu vypočítáme tak, že frekvenci termu t v dokumentu d vydělíme N (počet všech termů neboli slov v daném dokumentu d).

$$tf(t,d) = \frac{freg(t,d)}{N} \tag{4.1}$$

- Logaritmická normalizace frekvence termů - výpočet této varianty je zřejmý z vzorce 4.2.

$$tf(t,d) = log(1 + freg(t,d)) \tag{4.2}$$

- Dvojitá normalizace Frekvence termů hodnotou 0,5 - Z předchozích variant je zřejmý obsah rovnice až na jmenovatele. Ten je zde roven největšímu výsledku z varianty nazvané "Počet výskytů". Nejprve je tedy nutné spočítat variantu "Počet výskytů"pro celý dokument d a pak zjistit který term $t1$ má nejvyšší hodnotu.

$$tf(t,d) = 0,5 + 0,5 * \frac{freg(t,d)}{max(freg(t1,d))} \tag{4.3}$$

Inverzní frekvence dokumentů (Inverse Document Frequency)

Něž se dostaneme k tomu, jak se tato hodnota počítá, je důležité zmínit čeho vlastně dosáhneme jejím vypočítáním a co nám značí. Při hledání podobnosti v dokumentech, nebo jako v našem případě v zařízeních, je důležité najít schodné prvky, ale také zjistit jak jsou tyto prvky důležité. Právě onu důležitost neboli relevantnost získáme díky hodnotě IDF. Inverzní frekvence dokumentů je rozdílná od frekvence termů v tom, že nehledáme dané slovo v jednom dokumentu, ale ve všech dokumentech, které nás zajímají. Nepočítá však počet slov v daných dokumentech, ale počet dokumentů, které dané slovo obsahují. Pokud nás zajímá třeba slovo „term“ a hledáme v tisícovce dokumentů a zjistíme, že se vyskytuje v pěti z nich, tak nám tato informace bohatě stačí, čili není potřeba provádět součet výskytu slova „Term“ v pěti výše zmíněných dokumentech. Tuto metriku tedy lze vypočítat tak, že se vezme celkový počet dokumentů, vydělí se počtem dokumentů, které obsahují hledané slovo. Výsledek potom vložíme jako vstup funkce přirozeného algoritmu a máme námi hledanou hodnotu IFD. V našem případě by byl výsledek log (5/1000). Pokud se naše výsledná hodnota blíží 0, jsme schopni říci že hledané slovo se v dokumentech objevuje velice často. Naopak pokud se blíží hodnota číslu 1, víme že se hledané slovo vyskytuje v dokumentech jen velice zřídka.

$$idf(t,D) = log(\frac{N}{count(D)}) \tag{4.4}$$

TF-IDF

Konečně se dostáváme k tomu, jak získat hodnotu, která nás nejvíce zajímá. Jak už to tak bývá, tato hodnota se nakonec získá velice snadno, jedná se totiž pouze o součin dvou výše zmíněných hodnot, a to Frekvence termů a Inverzní frekvence dokumentů. Vysoké váhy v TF–IDF je dosaženo vysokou periodicitou (v daném dokumentu) a nízkou frekvencí dokumentů, které obsahují hledané slovo v celém sbírce dokumentů. Váhy proto mají tendenci odfiltrovat běžné výrazy. Pokud tyto informace shrneme, tak čím je vyšší hodnota TF-IDF, tím je dané slovo více zajímavé neboli relevantnější. Vyjádříme-li TF-IDF matematicky, pak skóre TF-IDF pro slovo t v dokumentu d ze sady dokumentů D, kde se nachází slovo t N-krát v dokumentech D, se vypočítá následovně:

$$tf - idf(t,d,D) = tf(t,d) * idf(t,D) \tag{4.5}$$

Výpočet podobnosti dvou zařízení

Nyní víme, jak pomocí metody TF-IDF získat hodnotu daného termu z dokumentu, ale tato hodnota nám o samotném dokumentu, či zařízení, nic neřekne. Je tedy zapotřebí s touto hodnotou nějak dále pracovat a zkusit ji porovnat s ostatními dokumenty, či zařízeními, abychom určili jejich podobnost. Jako nejjednodušší řešení se nabízí hodnoty spojit do vektoru, touto cestou se vydáme i my. Vytvoříme vektor hodnot pro každý dokument, který nás zajímá. Poté vezme vždy dva vektory dokumentů ($\vec{V}(d1)$, $\vec{V}(d2)$), které vložíme do vzorce 4.6. My si nyní vzorec rozdělíme na čitatele a jmenovatele. V čitateli se provádí skalární součin dvou vektorů (anglicky dot product), jehož výpočet naleznete ve vzorci 4.7. Výsledkem skalárního součinu je tedy hodnota, nikoliv vektor. Teď se podíváme na jmenovatele, kde nalezneme součin velikosti vektorů $\vec{V}(d1)$ a $\vec{V}(d2)$. Vzorec pro výpočet velikosti vektoru naleznete ve vzorci 4.8. Nakonec mezi sebou vydělíme čitatele a jmenovatele, výsledná hodnota nám značí jak jsou si mezi sebou dva dokumenty, či zařízení, podobné. Výsledná hodnota nabývá hodnoty z intervalu <0,1>, kde hodnota 1 značí sto procentní schodu a hodnota 0 žádný shodný term.

$$sim(d1, d2) = \frac{\vec{V}(d1) * \vec{V}(d2)}{|\vec{V}(d1)||\vec{V}(d2)|} \tag{4.6}$$

$$a * b = \sum_{i=1}^{N} a_i b_i = a_1 * b_1 + a_2 * b_2 + ... + a_n * b_n \; kde \; a = [a_1, a_2, ..., a_n], b = [b_1, b_2, .., b_n] \tag{4.7}$$

$$|\vec{v}| = \sqrt{v_1^2 + v_2^2 + ... + v_n^2} \; kde \; v = [v_1, v_2, ..., v_n] \tag{4.8}$$

Kapitola 5

Identifikace mobilních zařízení prostřednictvím dat zjištěných ze síťové komunikace

Identifikace mobilních zařízení se v dnešní době celkem často používá hlavně pro marketingové účely. Dá se však využit i k sledovaní uživatele na internetu. Mezi další cíle může například patřit i identifikace zařízení v neznámém síťovém provozu, kdy se může jednat i o kriminální činnost. Je proto nesmírné složíte určit, které části komunikace by měli být zaheslované a jak by k těmto soukromým datům měli programátoři přistupovat. V ideálním světě by byla komunikace šifrována, jakmile opustí zařízení. Tak tomu však v dnešní době zatím není. Některá zařízení v síti mají možnost ukládat data, ať už kvůli statistice, marketingu či z jiného důvodu. Zprávy se dají zachytávat bez menších problémů v obou typech komunikace, ať už se jedná o komunikaci drátovou anebo bezdrátovou. Na to abychom mohli zařízení identifikovat, potřebujeme ho nejprve nějakým způsobem popsat, k čemuž lze použít např. otisk zařízení, který bude popsaný v následující kapitole 5.1. Dále se podíváme na jednoduchou metodu pro porovnávání dvou otisků zařízení. Je totiž potřebné i tyto otisky nějakým způsobem porovnávat, aby jsme byli schopni určit, kdy se jedná o schodné otisky i když nejsou jejich otisky stoprocentně totožné.

5.1 Otisk zařízení (Device Fingerprint)

Jedním z dnes nejpoužívanějších metod pro identifikování zařízení v síti je takzvaný otisk zařízení, anglicky device fingerprint. Otisk zařízení je soubor informací, lépe řečeno kolekce, které o daném zařízení víme. Občas se část těchto dat transformuje do, dalo by se říci, hutnější verze pomocí hashovacích algoritmů, které my však využívat nebudeme. Dnes nejdebatovanější a nejčastěji řešení informace o zařízení se nacházejí v souborech cookies. O těchto souborech naleznete základní informace v kapitole 5.1.1. Informace o daném zařízení je možné sbírat dvěma přístupy a to pasivním a aktivním. Oba tyto přístupy jsou popsány v následujících kapitolách.

5.1.1 Soubory cookies

Mechanismus souborů cookie umožňuje webovému serveru ukládat malé množství dat do počítačů uživatelů, navštěvujících webové servery. Tyto data se poté odesílají zpět na webový

server na základě následných požadavků ze strany serveru. Dnes jsou soubory cookie jednou z klíčových technologií, na nichž jsou postaveny komplexní stavové webové aplikace. Krátce po zavedení souborů cookie bylo však zjištěno, že se dá zneužit jejich stavové povahy. Webové stránky se obvykle skládají z mnoha různých zdrojů, například HTML, obrázků, JavaScriptu a CSS, které mohou být umístěny jak na webovém serveru hostujícím hlavní stránku, tak na jiných webových serverech třetích stran. Při každém požadavku na webový server třetích stran, má tento server možnost nastavit a číst dříve nastavené soubory cookie v prohlížeči uživatele. Webový server třetích stran, tedy může zjistit veškeré informace o historii webové komunikace tohoto zařízení. Z těch to informací jsme schopni říci, že z načtení souborů cookies, jsme schopni vytvořit soubor informací, které ho charakterizují, což je jedna z možností jak se dá vytvořit otisk zařízení [31].

5.1.2 Aktivní metoda získávání otisku zařízení

Aktivní metoda získávání otisku zařízení, používá veřejně dostupné parametry, jako je název zařízení, typ, verze operačního systému, IP, operátor atd. Které využívá k identifikaci a vytvoření digitálního identifikátoru pro dané zařízení. Jedná se vlastně o aplikaci, která výše zmíněná data sbírá a uchovává v lepším případě pro marketingové účely. Otisk mobilních zařízení se v systému iOS používají převážně kvůli nedostatku dostačujících údajů z App Store. V některých případech se také používá v systému Android, pokud nejsou k dispozici data Google Play. Chod aplikace pro získaní informací o zařízení začíná od okamžiku, kdy uživatel klikne na adresu URL. JavaScript, který aplikace umísťuje jako sledovač, který získává data, začne shromažďovat všechna data kolem uživatele, například:

- IP adresa
- Platforma zařízení
- Značka zařízení
- Model zařízení
- Operační systém a jeho verze
- A spousty dalších

Aplikace nasbíraná data ještě kombinuje se soubory cookies, které vytěží z prohlížeče, aby získala co možná nejlepší otisk mobilního zařízení. Tato data pak mohou být dále obohacena o jejich korelaci s dalšími otisky zařízení a identifikací podobností mezi dalšími uživateli. Mezi nejznámější aplikace na vytvoření otisku zařízení patří Google Analytics. Google Analytics je v dnešní době nainstalován na velké množství webových serverů, a tak si občas ani provozovatelé webových stránek neuvědomují, že vytváří otisk zařízení [32].

5.1.3 Pasivní metoda získávání otisku zařízení

Pasivní metoda se od aktivní, která byla zmíněna v předešlé kapitole liší hlavně v tom jak data získáváme. V aktivní metodě se používají aplikace, které tyto informace sbírají a ukládají. Zatím co při pasivní sledujeme komunikaci těchto zařízení a data sbíráme z jejich komunikace. Takto je i v námi řešením případě, kdy odchytáváme pakety, které toto zařízení do sítě posílá a filtrujeme z nich pouze ty informace, které nás zajímají. z těchto informací pak jsme schopni vytvořit otisk zařízení, který nese informace o tom se kterým serverem a jak často dané zařízení komunikovalo.

5.1.4 Přesnost otisků zařízení

Otisky mobilních zařízení nemohou nejsou nikdy stoprocentně přesné. K nepřesnosti dochází v důsledku několika identifikátorů. Informace pocházející z mobilního procházení internetu mají pouze veřejně dostupná data z webového prohlížeče a adresy IP. Ukažme si příklad, kdy dochází k nepřesnosti. Předpokládejme, že jsou dva uživatelé připojeni ke stejné veřejné síti wifi. Mají tedy stejnou veřejnou IP adresu. Navíc mají stejný typ zařízení, model, operační systém a verzi operačního systému. Otisk mobilního zařízení mají tito dva uživatelé takřka identický.

5.1.5 Formát otisku zařízení

Ještě jsme se nedostali k tomu jak vlastně takový otisk zařízeni vypadá. Není zde stanoven přesný způsob jak si ho vytvořit nebo uložit. Mezi základní formáty otisků zařízení se dá počítat seznam informací, kde se budou postupně ukládat zjištěné hodnoty. Tyto hodnoty by se ukládali jako celý text. Další možností je ukládat pouze hodnoty 0 a nebo 1 do seznamu či vektoru, kde každá z hodnot odpovídá dané vlastnosti zařízení, například se může jednat o značku a nebo operační systém. Tyto seznamy a nebo vektory jsou jakési kolekce všech možných hodnot jednotlivých vlastností.

5.2 Porovnání dvou otisků zařízení pomocí Hammingovy vzdálenosti

Z předchozí kapitoly víme co to je otisk zařízení, teď se však zaměříme na to jak zjistit jestli jsou si dva otisky podobné. Podobnost dvou zařízení je totiž jedna z nejčastějších věcí, které nás zajímají a zaobírá se jím i tato práce. Ukažme si tedy jednu z jednodušších variant určení podobnosti dvou zařízení a to za pomoci Hammingovi vzdálenosti.

Pro začátek je zapotřebí si ujasnit jak vypadají naše otisky zařízení. Pro Hammingovu vzdálenost je dobrá, aby jsme měli hodnoty uložené do vektoru. Jednotlivé elementy vektoru budou nabývá hodnoty od 0 do 1, kde každá z hodnot odpovídá předem definované vlastnosti zařízení. Každé zařízení má svůj vektor, který značí otisk tohoto zařízení, seřazený tak aby každý element na dané pozici ve vektoru značil stejnou informaci ve všech vektorech. Ukažme si to na příkladu, máme vektor pro zařízení, které nás zajímá a na pozici nula ve vektoru je hodnota značící operační systém, který běží na daném zařízení, pak i ve všech ostatních vektorech, které značí další zařízení, musí být na pozici nula hodnota odpovídající operačnímu systému.

Máme-li takto upravené vektory pro zařízení u kterých chceme zjistit podobnost, můžeme přistoupit k tomu že zjistíme Hammingovu vzdálenost vždy pro dva vektory (zařízení). Nyní je třeba říci, co je to Hammingova vzdálenost a jak ji vypočítáme. Hammingovu vzdálenost nejčastěji používáme pro určení podobnosti dvou řetězců, kde je definována následovně:

- Hammingova vzdálenost $d_H(v, w)$ mezi dvěma řetězci $v, w \in \Sigma^*, |v| = |w|$ je minimální počet editačních operací výměna (replace) ke konverzi v na w [30].

Pro náš výpočet je, ale zapotřebí definici trošku poupravit a to tak, že místo porovnávání dvou řetězcům a jejich znaků dáme dva vektory a jejich elementy. Po této úpravě jsme, schopni určit Hammingovu vzdálenost mezi dvěma vektory a tedy i mezi dvěma otisky zařízení. Výsledná hodnota Hammingovi vzdálenosti nám značí, jak moc se od sebe dva

otisky (vektory) liší. Tedy pokud je hodnota rovna nule jsou shodné, ale pokud je rovna velikosti vektoru, tak se liší v každé informaci, kterou o zařízeních víme. Můžeme tedy říci, že čím je Hammingova vzdálenost mezi dvěma vektory nižší, tím jsou si podobnější.

Kapitola 6

Datové sady

Jak už bylo zmíněno v úvodu této práce, jako vstupní data pro analýzu dat síťové komunikace mobilních zařízení nám poslouží datové sady, které byly získané zachytáváním síťového provozu na fakultě informačních technologií pro výzkumné účely. Těchto pět datových sad obsahuje informace o komunikaci mobilních zařízení pomocí protokolu DNS a SSL/TLS. Navíc pro ověření, že se nám skutečně podařilo identifikovat zařízení máme k dispozici takzvanou ARP tabulku, kde se nachází mapování IP adresy na MAC adresy. Tato tabulka je důležitá, jelikož v ethernetových sítích a na internetu probíhá komunikace pomocí IP adresy a ta ve většině případů, minimálně co se týká IPv4, bývá přidělovaná DHCP serverem, takže námi hledaní zařízení jednou za čas dostane jinou IP adresu, a to jen v jedné síti. Pokud by se brala komunikace z více sítí, tak v každé dostane jinou IP adresu. Z těchto důvodu je zřejmé že IP adresa nám jako informace pro rozpoznání zařízení nestačí.

Za pomoci IP adresy jsme však schopni určit v jednotlivých datových sadách komunikaci jednoho zařízení a odfiltrovat případné zašumění. Mezi zašumění se můžou počítat zprávy typu multicast nebo broadcast. V našich datových sadách se však objevují pouze multicasty při vysílání, které se můžou hodit k lepší identifikaci zařízení, jelikož se objevují jen velmi zřídka. Pojďme si projít jednotlivé informace ve dvou hlavních typech datových sad. Pak se podíváme na předzpracovaní těchto sad.

6.1 DNS datové sady

DNS datové sady obsahují upravené informace z dotazů mobilních zařízení na DNS server. Obsahují tedy pouze informace, které by nám mohli pomoci k identifikaci zařízení v dalších datových sadách. V datových sadách se ale objevují i zprávy tipu multicast. V testování vyzkoušíme, jestli nám tyto zprávy budou při analýze k užitku, či ne. Pojďme se tedy blíže podívat z čeho se skládají DNS datové sady. Každý řádek nám značí jeden DNS dotaz na server, nebo multicastovou zprávu. Jednotlivé hodnoty v každém řádku jsou odděleny středníkem. Význam jednotlivých hodnot je následující:

- IP adresa odesílatele dotazu.
- IP adresa příjemce dotazu.
- Flagy posílané v DNS paketu.
- DNS jméno dotazovaného serveru.

6.2 SSL/TLS datové sady

Datové sady, které obsahují informace z komunikace mezi zařízením a (aplikačním) serverem, pomocí protokolů SSL a TLS, nesou více informací oproti datovým sadám DNS. Jsou rovněž upraveny, aby obsahovali jen informace, které jsou užitečné pro analýzu a identifikaci mobilních zařízení. Přesněji se jedná o vybrané informace z paketů "ClienHello", které klient posílá jako první při navázání spojení se serverem. Oproti DNS zprávám se zde nevykytují žádné multicasty. Stejně jako u DNS jdou jednotlivé hodnoty na řádku odděleny středníkem. Jednotlivé řádky znamenají jednotlivé zachycené zprávy. Pojďme se tedy podívat z čeho se jednotlivé hodnoty v řádcích skládají:

- IP adresa odesílatele paketu
- IP adresa příjemce paketu
- Číslo portu
- Podporovaná verze protokolu
- Seznam podporovaných šifrovacích algoritmů
- Seznam rozšíření (extensions), které může paket obsahovat (v našem případě ho obsahují všechny zprávy)

6.3 Příprava dat

Datové sady potřebujeme před samotnou analýzou zpracovat. Kdybychom nechaly data ve formě textovém souboru, bylo by nezbytné po každém kroku prohledávat datové sady řádek po řádku a psát složité funkce pro vyhledávání potřebných informací. Pro tuto analýzu se nabízejí dvě relativně stejně dobrá řešení. Pojďme se na každé z nich podívat a shrnout si výhody, či nevýhody těchto řešení.

1. První variantou je vytvoření takzvaných slovníků (directories), které by měly obsahovaly pro každou IP adresu seznam parametrů a počet jejich výskytu v daném data setu. S těmito slovníky se dá relativně dobře pracovat a jsou přehledné. Ovšem jejich vytvoření je časově i programově náročné. Samotné dotazování do slovníků na určité parametry je vcelku jednoduché a to pomocí klíče. Klíč by v tomto případě byla IP adresa daného zařízení. Slovníky by navíc měli po jejich vytvoření, už některé dotazy vypočítané jako například počet výskytů pro každou položku ve slovníku. Další nemalou výhodou je to, že není potřeba využívat jiné pomocné programy ani aplikace, vše je řešeno čistě programově. Shrňme si tedy hlavní výhody a nevýhody této varianty v bodech:

   + Čistě programové řešení bez využití jiného programu nebo jazyka.
   + Výpočet některých informací probíhá už při vytváření slovníků.
   – Složité vytváření slovníků.
   – Složitější získávání jen některých dat.

2. Druhá varianta je uložení našich dat z datových sad do databáze. Konkrétněji se jedná o MySQL databázi, která je volně dostupná a nabízí spoustu skvělých vlastností, které se při dalších operacích mohou hodit. Hlavní výhoda MySQL databáze je v dotazování do tabulek a jejich spojování, kdy můžeme na základě zvoleného sloupce spojit různé tabulky. Samozřejmě je nutné při tomto spojování dodržovat určitá pravidla. My se budeme hlavně zabývat spojením informací nasbíraných ze síťové komunikace protokolů DNS a SSL/TLS, tak se nám tato výhoda může velice hodit. Další nesrovnatelnou výhodou je jednoduché načtení dat ze souboru do databáze. Dotazování však zde bude o trochu pomalejší, jelikož je potřeba vytvořit dotaz a ten odeslat na databázi, která nám pak vrátí požadovaný výsledek. Nevýhodou je však to, že se zde využívá MySQL server, který musí být nainstalovaný a zapnutý. Pojďme se tedy ještě podívat na výhody a nevýhody tohoto řešení:

   + Jednoduché načtení dat do databáze.
   + Takřka neomezené možnosti získání a seřazení dat díky MySQL dotazům.
   − Potřeba dalšího programu(MySQL server).
   − Pomalejší získávání dat z databáze oproti slovníkům.

Nyní už známe klady a zápory dvou hlavních variant. Teď je zapotřebí, se zamyslet nad tím, která varianta nám přináší více užitku a bude se s ní lépe pracovat. Já osobně se přikláním více k druhé variantě, která nabízí jednodušší načtení dat do databáze a hlavně daleko větší možnosti při vybírání dat pro další zpracování. Čas o který se nám zvýší doba výpočtu je zanedbatelný, jelikož tato varianta je daleko rychlejší při vytváření tabulek v databázi oproti vytváření složitých slovníků. Tím pádem se nám výsledný čas výpočtu takřka nezmění v obou variantách.

Kapitola 7

Návrh systému

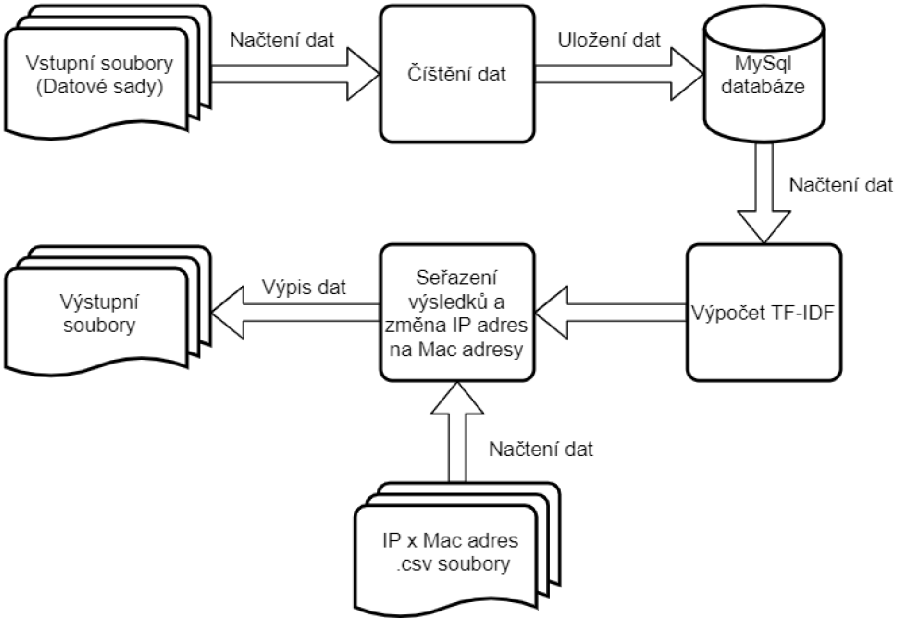
V této části práce, jak už název napovídá, se bude zabývat návrhem systému pro analýzu dat síťové komunikace mobilních zařízení. Systém bude sloužit k ověření toho, do jaké míry lze využít data z DNS a SSL/TLS komunikace pro profilování mobilních zařízení a určení shody těchto zařízení. Jedná se o analýzu dat ze síťové komunikace protokolů DNS a SSL/TLS. Pokusíme se navrhnout systém, který bude zpracovávat tuto komunikace jež je obsažena v pěti datových sadách, které jsme získali ze síťové komunikace na Fakultě Informačních Technologií v Brně. Systém se pokusí najít shodná zařízení mezi těmito jednotlivými datovými sadami. Nejprve se zamyslíme nad tím jak tyto data předzpracovat a poté navrhneme jak by měl celý systém vypadat. Vytvoříme také detail případu užití, který nám nastíní hlavní průchod tohoto systému.

Hlavní myšlenkou výsledného systému bude z těchto dat vytvořit otisky zařízení na základě IP adresy. Tyto otisky se budou vytvářet pro jednotlivá zařízení z datových sad a systém je bude porovnávat mezi sebou navzájem vždy mezi dvěma datovými sadami, za pomoci metody TF-IDF, které je popsána v kapitole 4.5.1. Výstup by měl obsahovat výsledek porovnání těchto otisků zařízení a to tím způsobem, že vypíše bodové ohodnocení neboli skóre na základě podobnosti síťové komunikace těchto zařízení. Měl by být schopný spojovat informace z komunikace protokolu DNS a SSL/TLS, ale zároveň by měl nabídnout možnost porovnávat pouze komunikaci jen jednoho z výše zmíněných síťových protokolů. Na následujícím obrázku 7.1 si můžete prohlédnout návrh celého systému.

Na návrhu systému je jasně patrné jakým způsobem budou procházet vstupní data systémem. Důležité je si uvědomit, že některé informace zadané od uživatele budeme potřebovat, až skoro úplně na konci. Jedná se o tabulky, které obsahují dva sloupce. V jednom sloupci se nachází MAC adresa a ve druhém odpovídající IP adresa. Tuto tabulku má každá datová sada, abychom byli schopni na konci výpočtu určit, jestli nejvyšší skóre opravdu náleží schodným zařízením. Jedním z další požadavků na systém, které zde zatím nebyli zmíněny je i seřazení dvojic porovnávaných zařízení podle výsledného skóre, a to od nejhoršího po nejlepší.

7.1 Návrh pro předzpracování dat a jejich uložení

Nejprve je zapotřebí rozmyslet si, jak budeme načítat vstupní data a jestli je potřeba vstupní data předzpracovat, či pročistit. Techniky pro čištění dat a jejich předzpracování naleznete v kapitole 3. Po nastudování technik pro redukci dat a zkoumání datových sad je zřejmé, že žádná z technik není potřeba. Data která jsme obdrželi od Fakulty Informačních

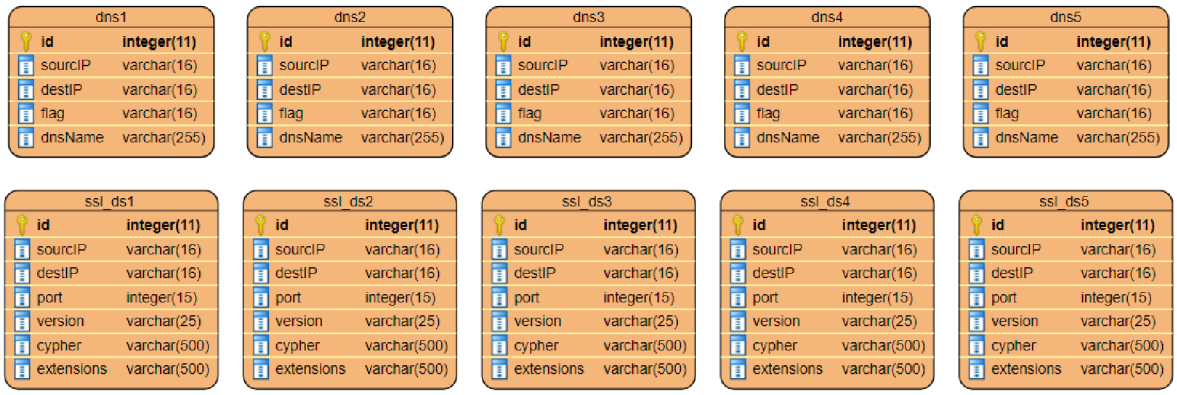


Obrázek 7.1: Návrh systému pro analýzu dat síťové komunikace mobilních zařízení

Technologií, jsou ve stavu, kdy tyto techniky, pro účel našeho výzkumu, nejsou potřeba. Když se však podíváme na druhou část výše zmíněné kapitoly a přidáme k ní informace z kapitoly 6, tak zjistíme, že datové sady obsahují i zprávy typu multicast. Jedná se o pár dotazů, které se v některých datových sadách vyskytují, které po patřičném prozkoumání pro nás mají jen nepatrný význam a při vytváření otisku zařízení, či jejich porovnávání by nám k užitku nebyly. Proto bude potřeba tyto dotazy odstranit.

Dále jsme v kapitole 6.3 rozvedly dvě myšlenky pro uložení dat po jejich pročištění. Přesněji se jedná o dvě možnosti, jimiž jsou slovníky (directories), a nebo databáze (MySQL). V kapitole 6.3 byly popsány jak výhody, tak nevýhody obou těchto variant, proto se k nim zde již nebudeme vracet. Co je však důležité zmínit, je to jakou cestou se nyní vydáme, v případě uložení pročištěných dat. Po patřičném uvážení a zhodnocení obou variant, jsme se rozhodli pro variantu dvě, tedy uložit data do databáze. Přesněji se bude jednat o databázi typu MySQL, která je volně k dispozici a není tedy problém s jejím obstaráním. Dále nám nabízí širokou možnost dotazování a lepší získávání dat pro vytvoření otisků zařízení. Pro účel vytvoření databáze z našich datových sad jsme vytvořili následující ER diagram 7.2.

Tento diagram obsahuje návrh naší databáze. Můžeme si povšimnout, že se nejedná o nějak složitou databázi. Nejedná se tedy o typické řešení, ale k řešení nebylo zapotřebí využití cizích klíčů, které tabulky typicky obsahují. Obsahuje každou tabulku pro danou datovou sadu a protokol, jehož komunikaci obsahují data v dané datové sadě. Takže nám pro každou datovou sadu vzniknou dvě tabulky, které nesou název stejný jako je název vstupního souboru. Z toho názvu je patrné o jakou datovou sadu i o jaký protokol se jedná. Jednotlivé sloupce nesou názvy informací, které obsahují. Každý řádek v tabulce nám pak bude značit informace, které jsme zjistili z jednoho odeslaného paketu daného zařízení. Přičemž, aby byl měl každý řádek svůj unikátní klíč je zde vložen sloupec Id, který obsahuje pouze číslo od 1 do maximálního počtu záznamů v tabulce a automaticky se inkrementuje s každým přidáním záznamu.



Obrázek 7.2: ER diagram navrhovaného systému

7.2 Návrh hlavní části systému a ošetření chyb

Nyní se dostáváme k návrhu hlavní částí systému, který bude potřebná data získávat již z databáze, či z csv souborů, které obsahují mapování z IP adres na Mac adresy. Tato část systému, by měla provádět výpočty, které jsou potřebné pro to, aby bylo možné jednoduše vypočítat TF-IDF hodnoty a následné skóre pro dvě zařízení. Je potřeba tedy promyslet celkové chování systému, které si ukážeme v tabulce 7.1 detailu případu užití. Typicky by profilování (nebo využití TF-IDF) fungovalo tak, že vezmeme jeden profil a hledáme k němu nejpodobnější zařízení v nějaké množině(zvolené datové sadě). Protože, ale chceme ověřit, zda otisk sestavený z vybraných dotazů je dostatečně dobrý pro profilování, nestačí nám profilovat jedno zařízení. Potřebujeme hodnoty ověřit pro různá zařízení, proto vezmeme vše co víme o daném zařízení a postupně pro všechna zařízení z jiné datové sady hledáme nejpodobnější zařízení.

Z tabulky 7.1 případů užití vypadá, že systém není nikterak složitý. Nejzapeklitější část se však skrývá ve výpočtu metody TF-IDF a výsledného skóre. Tato část systému, však bude dále popsán v kapitole o implementaci systému, a tak se jím momentálně nebudeme zaobírat. Systém bude také pouze konzolová aplikace, čili nebude mít žádné uživatelské prostředí. Jelikož se jedná o výzkumný systém je to rozumné rozhodnutí. Pokud se systém ukáže jako funkční nebylo by na škodu uživatelské prostředí do další verze přidat. Můžeme si však povšimnout, že systém může skončit i chybou a tak se nyní pojďme podívat na chyby, které by mohly v systému nastat, a které bude potřeba odchytit a dále je uživateli rozumným způsobem prezentovat, aby věděl proč systém neskončil úspěchem.

Když to vezmeme postupně tak první z chyb, které mohou nastat je špatně zadaný vstup uživatele. Jedna z možností je, že uživatel zadal špatnou cestu k souborům. Další by byla špatně zadané parametry pro výpočet a běh celého programu. Toto jsou asi nejdůležitější chyby ze strany uživatele, které je potřeba odchytit. Další problém může nastat při propojení našeho systému s databází. Problém ze strany síťového spojení by snad neměl nastat jelikož MySQL server bude nejčastěji spouštěn lokálně u uživatele a tak by se nemělo stávat, že nebude dostupný. Uživatele však může zadat špatné parametry pro přihlášení do databáze, a tak je potřeba i tuto uživatelskou chybu ošetřit. V neposlední řadě se může stát že se systém ocitne v neočekávaném stavu, tomu se samozřejmě pokusíme zabránit, a tak je zapotřebí části kódu, kde by tato situace mohla nastat, ošetřit takzvanými "try catch"bloky pro odchycení výjimky.

ID:	I1UC1
Název:	Spočítání skóre pro dvě datové sady
Aktéři:	Uživatel U, systém S
Funkcionalita:	Načíst vstupní data do systému, pročistit je a následně spočítat skóre pro jednotlivé dvojice zařízení
Vstupní požadavky:	Spuštěná databáze MySQL
Primární scénář:	1. U nastaví vstupní parametry 2. U spustí systém pro výpočet skóre 3. S vytvoří výstupní soubory a zapíše do nich výsledek předchozího bodu
Alternativní scénář:	1. S skončí chybou a vypíše se chybové hlášení
Frekvence:	Pokaždé
Výstupní podmínky:	Výstup je úspěšně vypsán do výstupních souborů

Tabulka 7.1: Detail případu užití systému

7.3 Parametry zadané uživatelem

Jelikož náš systém je v podstatě konzolová aplikace, tak grafické prostředí pro zadávání vstupních parametrů nepřichází v úvahu a bylo by zde možná i na škodu. Proto jsme se vydali cestou souboru, který bude obsahovat všechny potřebné informace, které pak systém načte a podle parametrů, které v něm budou vypočítá a vypíše výsledek. Soubor ve kterém uživatel zadává své vstupní parametry nese název **option.txt**. Soubor je textového formátu a jednotlivé parametry se zadávají na jednotlivé řádky a jsou podobné běžnému vkládání parametrů do konzole. Ukázku tohoto souboru si můžete prohlédnout na následujícím obrázku 7.3. Každý parametr v souboru začíná symbolem zavináče (@), ostatní řádky jsou ignorovány a dají se tak použít k popiskům, či komentářům tohoto souboru. Zadané parametry se dají logicky rozdělit do čtyř kategorií.

První z nich je nastaven databáze, kde uživatel musí zadat čtyři důležité parametry, které jsou zapotřebí pro projení systému s databází. Jedná se o login uživatele, jeho heslo, adresa databázového serveru a jméno databáze, kterou chce uživatel vytvořit, či k ní přistoupit. Způsob jakým se zadávají je patrný z obrázku 7.3.

Další kategorií jsou vstupní soubory, s daty o síťové komunikaci. Tyto soubory se zadávají jako parametr "I", jdoucí po znaku zavináče. Za tyto dva symboly je zapotřebí vložit symbol rovnítka (=) a pak stačí jen zadat cestu k souboru. Je velice důležité aby soubory, které obsahují ssl soubory byly ve složce, která se jmenuje "SSL" a ty které obsahují dns záznamy byli ve složce "DNS". Program totiž rozděluje vstupní soubory na dns a ssl právě díky tomu, že jsou obsažené v jiných složkách. Je totiž logické, že tyto soubory chceme uchovávat odděleně. Dále je velice důležité, aby název souboru obsahoval číslo datové sady, a to kvůli určení o kterou datovou sadu se jedná, aby bylo možné namapovat IP adresy na správné MAC adresy.

```
# DB settings

@DBLogin=root
@DBPassword=
@DBName=DP2
@DBHost=localhost

# Input settings

@I=C:\škola\FIT_VUT\DP\DP-git\DataSety\SSL\ssl_ds1.txt
@I=C:\škola\FIT_VUT\DP\DP-git\DataSety\SSL\ssl_ds2.txt
@I=C:\škola\FIT_VUT\DP\DP-git\DataSety\SSL\ssl_ds3.txt
@I=C:\škola\FIT_VUT\DP\DP-git\DataSety\SSL\ssl_ds4.txt
@I=C:\škola\FIT_VUT\DP\DP-git\DataSety\SSL\ssl_ds5.txt
@I=C:\škola\FIT_VUT\DP\DP-git\DataSety\DNS\dns1.txt
@I=C:\škola\FIT_VUT\DP\DP-git\DataSety\DNS\dns2.txt
@I=C:\škola\FIT_VUT\DP\DP-git\DataSety\DNS\dns3.txt
@I=C:\škola\FIT_VUT\DP\DP-git\DataSety\DNS\dns4.txt
@I=C:\škola\FIT_VUT\DP\DP-git\DataSety\DNS\dns5.txt

# MACK - IP input

@MACIP=C:\škola\FIT_VUT\DP\DP-git\DataSety\mac-ip\mac-ip_ds1.csv
@MACIP=C:\škola\FIT_VUT\DP\DP-git\DataSety\mac-ip\mac-ip_ds2.csv
@MACIP=C:\škola\FIT_VUT\DP\DP-git\DataSety\mac-ip\mac-ip_ds3.csv
@MACIP=C:\škola\FIT_VUT\DP\DP-git\DataSety\mac-ip\mac-ip_ds4.csv
@MACIP=C:\škola\FIT_VUT\DP\DP-git\DataSety\mac-ip\mac-ip_ds5.csv

# Output file settings

@O=1:2:DNS=dnsName,flag:SSL=cypher,extensions
@O=1:2:DNS=dnsName:SSL=cypher,extensions
@O=1:2:DNS=dnsName,flag:SSL=cypher
@O=1:2:DNS=dnsName,flag:SSL=extensions
@O=1:2:DNS=dnsName:SSL=
@O=1:2:DNS=:SSL=extensions,cypher
@O=1:2:DNS=:SSL=extensions
@O=1:2:DNS=:SSL=cypher
```

Obrázek 7.3: Ukázka souboru option.txt do kterého se zadávají vstupní parametry

Předposlední kategorií, je kategorie o tom, kde se nacházejí tabulkové soubory, které obsahují mapování IP adres na MAC adresy. Zadání těchto parametrů je velice jednoduché stačí za následující znaky "@MACIP=" přidat cestu k souboru. Je zapotřebí, aby název souboru obsahoval číslo datové sady. Systém jí totiž potřebuje k správné identifikaci datové sady.

Poslední kategorií jsou výstupní soubory a zároveň se touto metodou zadávají systému i informace, jaké datové sady má porovnávat a s jakými informacemi. Pro tyto informace slouží parametr "O", před nímž je znak zavináče. Dále následují informace o tom jaké datové sady by se měly porovnávat, mezi datové sady se vloží oddělovač symbolem dvojtečky. Za dalším symbolem dvojtečky jsou očekávány informace o tom jaké sloupce se mají vytáhnout z databáze k dns záznamům. Tyto informace je nutné zadat za symboly "DNS=", které označují, že se jedná o dns sloupce. Úplně stejným způsobem uživatel zadá informace o tom jaké parametry, chce získat z databáze o zařízení z komunikace nad protokolem ssl. Jen je zapotřebí podobně jak u dns zadat potřebné sloupce za těmito symboly "SSL=". Jednotlivé

sloupce, které chce uživatel filtrovat je zapotřebí oddělit čárkou a to jak u dns tak u ssl sloupců.

7.4 Forma výstupu

V předchozích kapitolách jsme řešili návrh zpracování vstupu a návrh hlavní části systému, a to včetně zobrazování chyb. Nyní se však zamyslíme nad tím jak by měl vypadat výstup z našeho systému. Jako první možnost se samozřejmě nabízí pouhý výpis do konzole (příkazové řádky). Tato varianta je však v případě vícero zařízení v datových sadách, kde se vyskytuje zhruba 5 až 8 zařízení, dosti nepřehledná. Chceme také uživateli dovolit porovnávat naráz vícero sad mezi sebou s jinými informacemi, které byly zachyceny v síťové komunikaci DNS, či SLL/TLS. Proto pro větší přehlednost a také možnost archivovaní výsledků, je nejlepší variantou ukládání výsledků do souborů. Je zapotřebí v těchto souborech jednotlivá zařízení rozumný způsobem oddělit a také seřadit. Uživatel má také možnost z výsledku smazat části, které ho nezajímají, kterými můžou být například zařízení, pro které vyšlo skóre podobnosti rovno nulové hodnotě.

Kapitola 8

Implementace systému

V předchozí kapitole 7, jsme se zabývali návrhem systému pro analýzu dat síťové komunikace mobilních zařízení a nyní se podíváme na samotnou implementaci toho systému. Podíváme se blíže na to jaký jazyk a hlavně proč, byl pro implementaci použit. Shrneme si využité knihovny, které byli pro implementaci nezbytné. Uvedeme taky popis jednotlivých funkcí, či metod, které byli z těchto knihoven použity. Následně se podíváme na jednotlivé třídy systému a popíšeme si celkovou implementaci. Nesmíme také zapomenout na jejich souhrn v digramu tříd, který tato kapitola obsahuje.

8.1 Programovací jazyk Python a použité knihovny

Nyní se podíváme na to jaké technologie jsou použity k implementaci systému. Jako programovací jazyk jsme po delším přemýšlení použili Python. Oproti ostatním jazykům nám nabízí jistou výhodu. Především se jedná o to, že umí bezproblémově pracovat s databází MySQL, ale také hlavně proto, že nabízí dynamickou kontrolu datových typů, které se pro řešení daného problému velice hodí. Dalším důvodem pro volbu Pythonu, byla široká škála knihoven, jejichž využití se pro navrhovaný systém hodí a ušetří nám spoustu práce. Mezi tyto knihovny se řadí i knihovna sklearn, která obsahuje implementaci metody TF-IDF a tudíž není potřeba ji ručně programovat. Navíc má pro tuto metodu i jistá vylepšení při výpočtu, která umožňuji lepší a hlavně rychlejší výsledky. V následujícím seznamu naleznete seznam všech knihoven, které jsme při implementaci systému využili. Postupně si projdeme funkce, které jsme z nich využili a popíšeme si jak fungují.

Seznam použitých knihoven:

- sklearn
- pandas
- numpy
- collections
- csv
- os
- pymysql
- sys

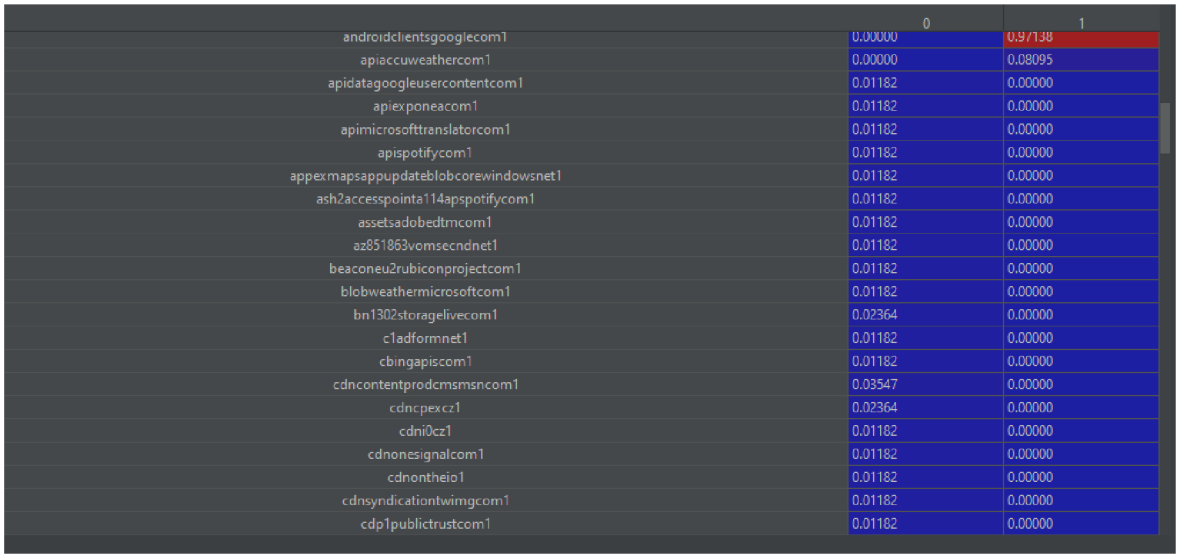
Knihovna sklearn

Tato knihovna obsahuje velké množství algoritmů, které se dají použít pro různé úlohy pro dolování dat. Například je zde podpora pro Rozhodovací stromy a klasifikaci. Tyto úlohy nás však až tolik nezajímají, ale mimo jiné obsahuje i třídu **TfidfVectorizer**. Tuto třídu jsem využili k výpočtu vektorů s hodnotami TF-IDF. Stačí prvně vytvořit objekt, z této třídy a pak nad ním zavolat metodu **fit_transform()**, které jako parametr pošleme seznam, který obsahuje termy ze síťové komunikace. Termy je však zapotřebí upravit, jelikož tato metoda bere jako oddělovač jednoho termu jakékoliv znaky, které nejsou písmena. Proto je zapotřebí u jednotlivých termů tyto znaky odstranit a vytvořit tak term, který se skládá jen z písmen. Tato úprava však není nikterak náročná a pokud všechny termy upravíme správným způsobem a to tak, že nebude prohazovat žádné znaky nezmění se jejich informační hodnota. Termy z každého zařízení je pak zapotřebí spojit do jednoho řetězce a oddělit mezerou. Máme li takto upravené termy pro dvě zařízení stačí z nich udělat seznam, který obsahuje dvě položky a to právě tyto dva řetězce. Nejlépe si to asi ukážeme na následujícím příkladu. Máme-li první zařízení jehož upravené termy jsou v řetězci a vypadají následovně: "aaa bbb ccc ddd". Termy druhého zařízení vypadají následovně: "ccc eee iii". A výsledný seznam, který pošleme jako parametr metodě **fit_transform()**, bude vypadat následovně: ["aaa bbb ccc ddd", "ccc eee iii"].

Další a zároveň poslední metodu, kterou jsme z této knihovny využili je metoda **get_feature_names()**. Tato metoda není nikterak složitá a nemá žádné parametry. Pro nás je však velice důležitá z jednoho prostého důvodu. Tímto důvodem je ten, že nám vrací seznam termů, tak jak jim odpovídají hodnoty ve vektoru hodnot TF-IDF, který jsme získali v předchozím odstavci. Hodí se nám totiž pro další výpočty, abychom vybírali z vektorů odpovídající hodnoty pro stejný term a dávali je do výsledného vektoru otisku zařízení správně seřazené.

Knihovna pandas

Tato knihovna slouží k rychlé, výkonné, flexibilní a snadno použitelné analýze a manipulaci s daty. My z ní využijeme datovou strukturu, která se jmenuje **DataFrame**. Tato struktura se nám hodí pro to, abychom s její pomocí vytvořili něco velice podobného databázi, z této datové struktury jednoduše vytáhneme potřebné hodnoty TF-IDF pro daný term. Vytvoříme ho za pomoci dvou metod, které jsou zmíněné v sekci 8.1 o knihově sklearn. Pro vytvoření této datové struktury tedy potřebujeme, seznam termů a hodnoty TF-IDF pro dané termy ve vektorech, které už víme jak vytvořit z předchozích odstavců. Výsledkem je pro nás datová struktura podobná databázi, kterou si můžeme prohlédnout na následujícím obrázku 8.1. Tuto strukturu použijeme pro vytvoření vektorů, otisků zařízení tak, aby z nich bylo možné vypočítat skóre pro míru jejich shody, která je popsána v kapitole 4.5.1. Pro vytvoření výše zmíněných vektorů totiž potřebujeme mít jistotu, že se nám na stejných pozicích v obou vektorech vyskytují správné hodnoty TF-IDF. Tyto hodnoty totiž musí být pro daný term i zařízení na stejném místě v obou vektorech. To zajistíme tak, že si z datové struktury DataFrame vytáhneme vždy hodnotu TF-IDF daného termu pro dané zařízení pomocí klíče, kterým je právě název daného Termu. Na následující ukázce z kódu snad bude patrné jak výše zmíněnou hodnotu získáme: daná hodnota TF-IDF pro term "Term"= pandas.loc["číslo zařízení"].at["Term"].



	0	1
androidclientsgooglecom1	0.00000	0.97138
apiaccuweathercom1	0.00000	0.08095
apidatagoogleusercontentcom1	0.01182	0.00000
apiexponeacom1	0.01182	0.00000
apimicrosofttranslatorcom1	0.01182	0.00000
apispotifycom1	0.01182	0.00000
appexmapsappupdateblobcorewindowsnet1	0.01182	0.00000
ash2accesspointa114apspotifycom1	0.01182	0.00000
assetsadobedtmcom1	0.01182	0.00000
az851863vomsecndnet1	0.01182	0.00000
beaconeu2rubiconprojectcom1	0.01182	0.00000
blobweathermicrosoftcom1	0.01182	0.00000
bn1302storagelivecom1	0.02364	0.00000
c1adformnet1	0.01182	0.00000
cbingapiscom1	0.01182	0.00000
cdncontentprodcmsmsncom1	0.03547	0.00000
cdncpexcz1	0.02364	0.00000
cdni0cz1	0.01182	0.00000
cdnonesignalcom1	0.01182	0.00000
cdnontheio1	0.01182	0.00000
cdnsyndicationtwimgcom1	0.01182	0.00000
cdp1publictrustcom1	0.01182	0.00000

Obrázek 8.1: Ukázka datové struktury DataFrame

Knihovna numpy

Numpy je knihovna, která obsahuje spoustu funkcí, které pokrývají většinu numerických výpočtů. Právě i z tohoto důvodu jsme ji použili i my. Je totiž potřeba z vektorů, pro jednotlivá zařízení vypočítat výsledné skóre podobnosti. Tento postup je popsán v kapitole 4.5.1, potřebujeme tedy vypočítat skalární součin dvou vektorů a velikost jednotlivých vektorů. Pro výpočet skalárního součinu vektorů slouží metoda **dot()**, která má jako vstupní parametry dva vektory, u nás to jsou dva vektory, které vytvoříme s pomocí metod, které jsou zmíněné v předchozích kapitolách. Pro výpočet velikosti vektoru využijeme metodu **linalg.norm()**, která přijímá pouze jeden parametr a to je vektor jednoho zařízení. Proto tuto metodu budeme muset zavolat nad každým vektorem zvlášť.

Knihovna collections

Tato knihovna, jak už název napovídá, slouží k vytváření různých datových kolekcí. My jsme z této knihovny využili pouze třídu **defaultdict**. Tato třída slouží k vytváření slovníků. Bylo totiž nutné vytvořit slovník. Bylo zapotřebí nějakým způsobem mapovat IP adresy na MAC adresy. Tato třída vytváří slovník, který našim účelům bohatě vyhovuje. Podívejme se teď podívat na to jakým způsobem se tyto slovníky, které k daným IP addresám uchovávávají hodnotu odpovídající MAC adresy, vytvářejí. Pro každé mapování IP adresy a MAC adresy vložíme do slovníku jednu hodnotu, kde klíčem pro získání správné MAC adresy bude daná IP adresa.

Knihovna csv

Knihovna csv slouží k práci s csv soubory. My jsme ji využili k práci právě s těmito soubory. Vstupní soubory s mapování IP adres na MAC adresy je totiž obsažený právě v csv souborech. Proto jsme pro jejich zpracování použili tuto knihovnu.

Knihovna os

Tato knihovna slouží k práci s rozhraním operačního systému. Pro nás má využití při načítáni vstupu. První metoda, kterou jsme z této knihovny použili slouží k tomu, abychom byli schopni určit, jestli uživatel správně zadal cestu k vstupním souborům. Ověříme jestli cesta k souboru existuje a pokud ano tak ověříme, že se opravdu jedná o soubor. Použili jsme ji také pro práci s cestami k souborům. Nejlepší způsob jak vytvořit názvy tabulek v databázi, je využít jména vstupních souborů. Proto potřebujeme tato jména dostat z cesty k souboru, kterou nám zadá uživatel a k těmto účelům má knihovna os své funkce.

Knihovna pymysql

Pymysql knihovna slouží pro práci s databází MySQL. Potřebujeme vytvářet tabulky a provádět nad nimi dotazy. Postup, který jsme využili pro vytváření tabulek, jejich naplnění a následné dotazování je už velice jednoduché. Stačí nám k tomu pouze vytvořit spojení (connection). Pro tyto účeli jsme využili metodu z knihovny, která nese přínosný název, **connect()**. Metoda **connect()** potřebuje k připojení do databáze tři parametry. Parametry se rovnou přiřazují správným proměnným při volání výše zmíněné metody. Parametry jsou následující:

- host - neboli jméno serveru, či jeho adresa
- user - jméno uživatele, který má do databáze přístup
- password - heslo daného uživatele

Knihovna sys

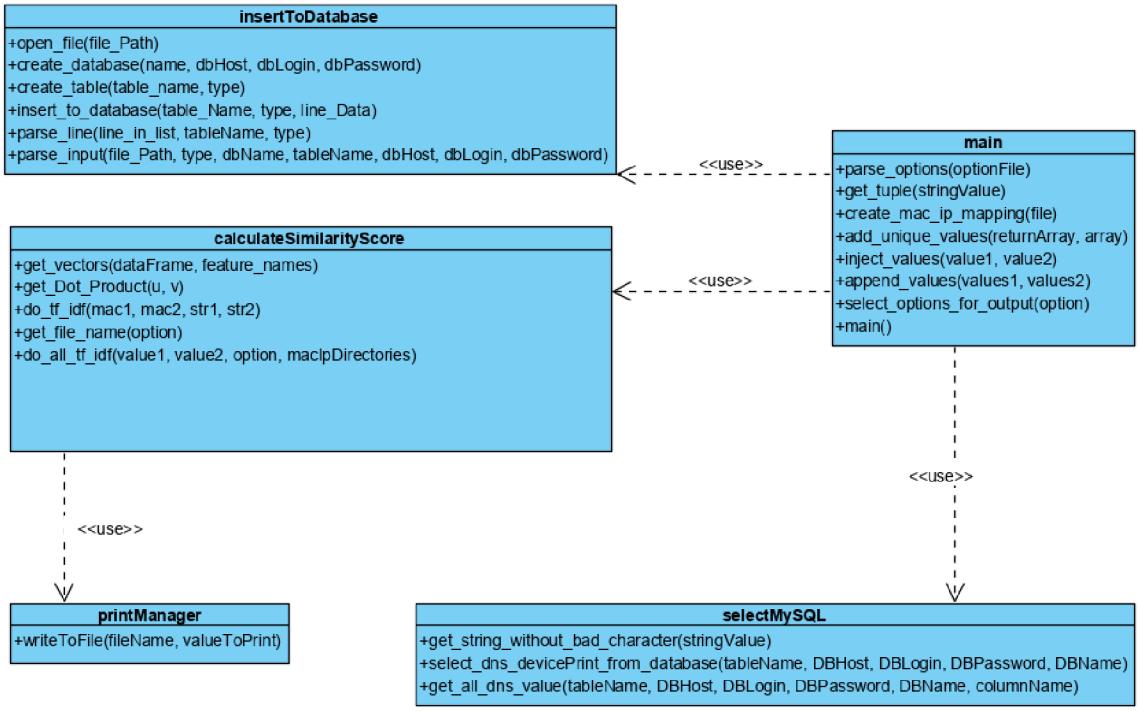
Knihovna sys nám umožňuje přístup k proměnným které používá interpret nebo s ním silně interagují. Pro nás má využití při odchytávání výjimek v takzvaných "try catch"blocích, které využíváme v částech kódu kde je možné, že by mohl selhat. Systém nám díky ním neselže jen uživateli do konzole vypíše výjimku, kterou odchytil.

8.2 Popis systému

Nyní se podíváme na celkovou implementaci systému pro analýzu dat síťové komunikace mobilních zařízení. Ukážeme si jednotlivé třídy v digramu tříd 8.2 a popíšeme si jak byl celý systém implementován, jak se data ukládají do databáze a která z nich jsou vynechána. Dále si vysvětlíme, jak se z databáze data přesně získávají a jak se z nich vypočítává výsledné skóre podobnosti dvou zařízení.

Systém nejprve načte vstupní soubor "option.txt", ze kterého získá potřebné informace jeho rozpársováním. Tento soubor je procházen řádek po řádku a systém v něm hledá všechny potřebné informace. Data získaná z tohoto souboru si uloží do potřebných globálních proměnných, aby k nim měly přístup veškeré další funkce a mohly z nich kdykoliv získat potřebné informace. Cesty k vstupním souborům si systém uloží do polí, aby se přes ně dalo procházet a postupně je nahrávat do databáze. Úplně stejně tomu je u výstupních souborů a souborů obsahujících mapování IP adres na MAC adresy.

V dalším kroku systém vytvoří databázi. Při vytváření databáze prvně systém ověří jestli bylo zadáno jméno databáze, bez této informace totiž není schopen, žádné další akce a tak se ukončí s tím, že upozorní uživatele na tuto skutečnost. Pokud však bylo jméno



Obrázek 8.2: Diagram tříd systému pro analýzu dat síťové komunikace mobilních zařízení

databáze vyplněno systém pokračuje dále a pokusí se vytvořit databázi s názvem, které zvolil uživatel. K této akci je zapotřebí, aby uživatel také zadal jméno uživatele, který má k databázi přístup a také heslo pro tohoto uživatele. Musí být i vyplněna adresa serveru, v případě kdy je MySQL server spuštěn lokálně je název serveru localhost. Před samotným vytvářením databáze systém ověří jestli daná databáze již není vytvořena, pokud tomu tak je tak ji znovu nevytváří.

Po úspěšném vytvoření databáze, pokud tedy databáze již neexistovala, systém projde všechny vstupní soubory, které byly zadány uživatelem. Prvně zkontroluje jestli uživatel nějaké takové soubory zadal. Pokud ano postupně se soubory kontrolují. Prvně se pomocí cesty k souboru určí jaká data soubor obsahuje a také z ní získáme název tabulky, protože systém použije jako jméno tabulky název souboru, z kterého data ukládá do databáze. Před samotným vkládáním ještě systém ověří jestli soubor, který má do databáze vkládat, opravdu existuje. Jestli je vše v pořádku systém vytvoří tabulku v databázi a začne postupně procházet jednotlivé řádky vstupního souboru a vkládat je do databáze. Před samotným vložením provede systém čištění dat. Co se týká dns záznamů tak zde není potřeba data čistit ani nijak upravovat, takže tyto data jsou rovnou vložena do databáze. U ssl záznamů, je však čištění provedeno, jelikož zde se vyskytují určité záznamy, které je zapotřebí odstranit. Kontroluje se číslo portu, které by nemělo být větší jak 49151, dále se kontroluje jestli některé z informací o šifrovacích metodách nebo rozšíření nejsou prázdné. Tyto data pro nás totiž nemají žádný význam a tak je dobré je odstranit v tomto kroku, abychom je nemuseli kontrolovat dále a nezabírala nám místo v databázi.

Nyní se dostáváme ke kroku, kdy systém vytváří slovníky, které mapují IP adresy na MAC adresy. Lepší je vytvořit je předem, aby jsme se s nimi nezdržovali v dalších krocích, kdy chceme vypsat informace o tom jaká zařízení se porovnávala. Prvně systém provede kontrolu, jestli uživatel zadal cestu k souborů jež obsahují potřebné informace. Pokud uži-

vatel zadal správně cestu k souborům tak se prochází postupně jeden za druhým. Každý soubor se prochází řádek po řádku a systém si informace z nich ukládá do slovníků. Slovníky vypadají tak, že klíčem k získání IP adresy je MAC adresa daného zařízení. Takže pak systému stačí jen vytaženou MAC adresu z databáze použít jako klíč, či index do tohoto slovníku a ten nám vrátí správnou IP adresu. Systém při vytváření slovníků bere v potaz to jak jsou zandány uživatelem a předpokládá, že první soubor, který byl zadán náleží k prvnímu datové sadě.

Teď se zaměříme na nejzajímavější a zároveň hlavní část celého systému a tou je vypočítání skóre shody pro dvojici zařízení. Nejprve systém zjistí jaké dvě datové sady chce uživatel porovnávat a které informace ze síťové komunikace chce použít. Všechny tyto informace zjistí ze zadaného parametru o výstupním souboru. Ten je však potřeba prvně správně rozparsovat. Po té co systém zjistí jaká data je potřeba získat z databáze provede dotazy do databáze. Dotazy probíhají tak, že prvně na správné tabulce podle datové sady, kterou zadal uživatel, získá unikátní IP adresy, které se v datové sadě vyskytují. Po té pro jednotlivé IP adresy vytáhne z databáze potřebné sloupce. Toto vše se provádí v jednom for cyklu. Data vrácená z databáze jsou ve formě pole a je potřeba z nich udělat jeden řetězec. Zároveň pro každou položku z pole, co systému vrátí databáze, systém provede odstranění nepotřebných znaků, které by komplikovali další výpočet. Symboly, které je zapotřebí odstranit jsou v systému v seznamu, který nese název "bad_chars". Tento seznam znaků je vyobrazen na obrázku 8.3. Systém také umožňuje spojit informace z více sloupců v dané tabulce. V tomto případě jsou informace z daného řádku tabulky spojeny do jednoho řetězce. Výsledné informace o daném zařízení má tedy systém upravené v jednom řetězci, kde hodnoty získané z databáze jsou oddělené mezerou. Pokud uživatel zvolil kombinaci informací z dns a ssl datových sad, je zapotřebí tato data z databáze spojit do jednoho řetězce, k tomuto účelu systém použije konkatenaci dvou řetězců. K tomuto výslednému řetězci si také ukládáme i IP adresu, abychom věděli o jakém zařízení tento řetězec obsahuje informace.

```
bad_chars = [';', ':', '!', "-", ",", " ", "_", "-", "/", "\'", "\n"]
```

Obrázek 8.3: Seznam znaků z proměnné bad_chars

Po té co systém získá potřebná data z databáze máme k dispozici seznam řetězců (otisků zařízení), a k nim odpovídající IP adresy. Nyní je zapotřebí tyto získané informace ohodnotit s pomocí knihovny **sklearn**, která je popsána v kapitole 8.1. Za pomocí metod z této knihovny systém spočítá TF-IDF hodnotu pro jednotlivé termy, což jsou v daném řetězci jednotlivé položky oddělené mezerou. Tyto hodnoty pak použije pro výpočet skóre podobnosti, který je popsán v kapitole 4.5.1. K výpočtům skalárního součinu dvou vektorů a velikosti vektorů se použijí metody z knihovny **numpy**, které jsou popsané v kapitole 8.1. Tento výpočet se provádí ve for cyklu pro všechny kombinace zařízení ze dvou datových sad, které zadal uživatel.

Nyní systém výsledky získané ze výpočtu skóre pro dvě zařízení zapíše do výsledného souboru. Tento výpis se provádí vždy po porovnání jednoho zařízení z první datové sady se všemi ostatními zařízeními z druhé datové sady. Pro přehlednost jsou před výpisem data seřazena od nejnižšího skóre po nejvyšší skóre. Zároveň jsou IP adresy nahrazeny MAC adresami, aby uživatel z výstupu poznal na první pohled jestli nejvyšší skóre je pro stejné zařízení, či nikoliv. Zároveň může z výsledku zjistit jestli je shodné zařízení aspoň v top 3, či někde na konci. Název výstupního souboru systém generuje tím způsobem, že spojí datové

sady, které se porovnávali s názvy sloupců, které z dané datové sady získal z databáze. Tímto způsobem uživatel na první pohled pozná jaké informace se použili na vytváření otisků a výsledné porovnávání.

8.3 Ošetření chyb

V systému bylo také zapotřebí ošetřit chybové stavy do kterých by se systém mohl dostat, ať už při zadání špatných dat, či vstupních parametrů. Chyby, které mohou nastat při načítání souboru s parametry jsou ošetřené tak, že pokud je v nich nějaká chyba nenačtou se a ani se neprovede žádný výpočet. Čili pokud uživatel zadá systému špatné parametry systém se ve své podstatě nespustí protože k tomu nemá důvod. Veškeré chyby, které by mohli nastat při komunikaci s databází za nás řeší knihovna **pymysql**, které chyby vypíše do příkazové řádky, ale chod programu se neukončí pokud nenastane nějaká kritická chyby, v tom případě program skončí v chybovém stavu. Dále veškeré záznamy z datových sad, které se neukládají do databáze z důvodu čištění, či špatně zadaných dat jsou vypsány do konzole a ohraničeny zprávou, která uživateli oznamuje, že tato informace se do databáze nevložila. Poslední chybou, které může v systému nastat je chyba při výpočtu výsledného skóre shody pro dvě zařízení a nebo při výpisu do soubory. Tyto dvě operace má na starost jediná funkce, která je ošetřena, už párkrát zmíněným, "try catch" blokem, který odchytí výjimku a tu vypíše uživateli do konzole, aby měl stále přehled o tom co se děje.

Kapitola 9

Testování a Experimenty

Nyní se podíváme na to jakým způsobem byl systém testován, a jak je ověřeno, že použité knihovny správně počítají výsledky, se kterými se dále pracuje. Pro ověření správné činnosti systému je totiž dobré otestovat výstup z volně dostupných knihovních funkcí. Dále se zaměříme na experimenty, které na systému byly provedeny. Jejich výsledky se pokusíme rozumným způsobem zobrazit za použití tabulek a grafů, aby bylo na první pohled patrné jak experiment dopadl.

9.1 Testování systému

Testování systému bylo prováděno po částech, podle toho jak byl systém postupně implementován. Prvně bylo důležité otestovat jestli se data propisují do databáze a jestli obsahuje všechna nahraná data. Toto testování bylo prováděno ručně, po připojení do databáze skrze uživatelské prostředí, kde byla zobrazena veškerá data, která byla nahrána do databáze. Podle počtu řádků v souborech s datovými sadami a záznamech v databázi jsme zjistili, že se nám při nahrávání do databáze, žádná data neztrácejí. Dále byla provedena kontrola několika náhodných záznamů v databázi, které byli srovnány se vstupními daty, jestli jsme nepřišli o žádnou potřebnou informaci.

Dále bylo potřeba otestovat funkcionalitu metody na výpočet TF-IDF. Za tímto účelem byli provedeny dva testy. První z nich, byl takový, že jsme systému na vstup vložili testovací data, na kterých jsme prvně ručně vypočítali jejich hodnotu a poté jsme ji porovnali s výsledky metody. Jelikož tato metoda používá složitější a rychlejší algoritmy než ruční výpočet, byli výsledky jen maličko odlišné a to až v řádech tisícin, což mohlo být i způsobeno zaokrouhlováním. Pro účely toho systému však tento drobný rozdíl nehraje žádnou roli.

Druhý test metody TF-IDF se prováděl, až po vypočtení výsledného skóre shody dvou zařízení, kdy jsme porovnávali výsledky skóre a datové sady. Testování tedy bylo v tomto případě manuální, zkoumali jsme množství shodných termů mezi různými dvojicemi zařízení a posoudili jsme zda hodnota skóre podobnosti intuitivně odpovídá dosažených výsledků.

Poslední testování, které jsme vykonali bylo při počítání skalárního součinu dvou vektorů a velikosti vektoru. Testování vypadalo následujícím způsobem. Prvně jsme si připravili uměle vytvořené vektory u kterých jsme znaly výsledek, který jsme ručně předem vypočítali. Po té jsme uměle vytvořené vektory poslali jako vstupní parametr do funkcí a porovnali jsme výsledky. Zde jsme dosáhli perfektního výsledku jelikož všechny předem spočítané výsledky a výsledky z funkcí se rovnaly ve všech případech.

9.2 Experimenty

Nyní se podíváme na experimenty, které byly s pomocí systému pro analýzu dat síťové komunikace mobilních zařízení provedeny a ohodnotíme výsledky, kterých jsme v těchto experimentech dosáhly. Zaměříme se v experimentech jen na první dvě datové sady, jelikož se v nich nachází nejvíce shodných zařízení, které komunikovaly. Také mají z datových sad, které byli pro testování a experimenty získány, nejvíce záznamů a proto se na experimenty hodí nejvíce. Prvně se blíže seznámíme s tím jaká zařízení se shodují v těchto datových sadách a kolik zařízení naopak vysílalo jen v jedné, či druhé datové sadě. Dále se podíváme jak se výsledné skóre shody mezi jednotlivými zařízení v prvních dvou datových sadách měnilo za použití různých parametrů, které jsme systému zadávali.

Shodná zařízení v první a druhé datové sadě

Před samotnými experimenty je dobré si projít první dvě datové sady a zjistit, která zařízení jsou shodná a kolik záznamů mají zařízení v jednotlivých datových sadách. Bude nás totiž v dalších experimentech zajímat jaký vliv má počet záznamů na výsledné skóre shody dvou zařízení.

Prvně se podíváme na obecné informace o prvních dvou datových sadách. Je dobré vědět jak jsou datové sady obsáhlé, kolik obsahují záznamů a kolik Zařízení v nich komunikovalo. V tabulce 9.1, jsou přehledné informace o záznamech zachycených ze síťové komunikace pomocí DNS paketů. V tabulce 9.2 jsou stejné hodnoty, ale z datových sad zachycených z komunikace pomocí SSL/TLS paketů. Můžeme si z těchto tabulek odnést to, že ne všechna zařízení, která komunikovala za pomocí DNS také komunikovala pomocí SSL/TLS. Je také jasně patrné, že SSL/TLS záznamů je daleko méně.

Číslo datové sady:	Počet zařízení:	Počet záznamů:
1	13	3050
2	8	507

Tabulka 9.1: Metriky prvních dvou datových sad s DNS záznamy

Číslo datové sady:	Počet zařízení:	Počet záznamů:
1	12	1318
2	5	273

Tabulka 9.2: Metriky prvních dvou datových sad se SSL/TLS záznamy

Nyní se zaměříme na to kolik zařízení se nám shoduje, mezi těmito datovými sadami. Hodnoty v následující tabulce 9.3 znázorňují kolik zařízení se shoduje pro jednotlivé typy záznamů v datových sadách. Z této tabulky 9.3 je patrné, že zařízení, která se vyskytují v druhé datové sadě, až na jedno, se vyskytují také v první datové sadě.

Typ záznamu:	Počet shodných zařízení mezi datovými sadami:
DNS	6
SSL/TLS	5

Tabulka 9.3: Počet shodných zařízení v prvních dvou datových sadách

Teď se podíváme samostatně na jednotlivá zařízení prvních dvou datových sad. Podíváme se kolik mají jednotlivá zařízení záznamů, abychom mohli tuto informaci použít v následujících experimentech. V tabulce 9.4 a 9.5 si můžeme všimnout kolik DNS záznamů mají jednotlivá zařízení z prvních dvou datových sad. V tabulce 9.6 a 9.7 si můžeme všimnout kolik SSL/TLS záznamů mají jednotlivá zařízení z prvních dvou datových sad. Dále jsme kvůli jednoduššímu rozpoznání dvou shodných zařízení, a také kvůli citlivosti údajů, nahradili MAC adresy z tabulek za identifikátory zařízení. Tento identifikátor nám čísluje všechna zařízení v obou datových sadách, takže si můžete povšimnout, že některé identifikátory jsou shodné pro obě datové sady, jelikož se jedná o stejná zařízení.

IP adresa:	Identifikátor zařízení:	Počet záznamů:
10.42.0.100	1	191
10.42.0.134	2	54
10.42.0.156	3	34
10.42.0.161	4	4
10.42.0.162	5	75
10.42.0.171	6	31
10.42.0.199	7	16
10.42.0.205	8	98
10.42.0.232	9	807
10.42.0.253	10	784
10.42.0.76	11	382
10.42.0.85	12	485
10.42.0.97	13	89

Tabulka 9.4: Počet záznamů pro jednotlivá zařízení v DNS datové sadě číslo 1.

IP adresa:	Identifikátor zařízení:	Počet záznamů:
192.168.42.151	13	61
192.168.42.157	14	42
192.168.42.16	11	203
192.168.42.1	15	2
192.168.42.41	1	24
192.168.42.76	8	91
192.168.42.77	2	3
192.168.42.82	3	81

Tabulka 9.5: Počet záznamů pro jednotlivá zařízení v DNS datové sadě číslo 2.

IP adresa:	Identifikátor zařízení:	Počet záznamů:
10.42.0.100	1	105
10.42.0.156	3	14
10.42.0.161	4	2
10.42.0.162	5	36
10.42.0.171	6	13
10.42.0.199	7	1
10.42.0.205	8	44
10.42.0.232	9	550
10.42.0.253	10	150
10.42.0.76	11	225
10.42.0.85	12	82
10.42.0.97	13	96

Tabulka 9.6: Počet záznamů pro jednotlivá zařízení v SSL/TLS datové sadě číslo 1.

IP adresa:	Identifikátor zařízení:	Počet záznamů:
192.168.42.151	13	15
192.168.42.16	11	120
192.168.42.41	1	8
192.168.42.76	8	70
192.168.42.82	3	60

Tabulka 9.7: Počet záznamů pro jednotlivá zařízení v SSL/TLS datové sadě číslo 2.

Vyhledávání stejných zařízení v datových sadách za pomoci jména dotazovaného serveru

První experiment, který jsme za pomoci systému provedly, byl o tom, pokusit se ověřit jestli nám pro určení dvou shodných zařízení stačí využít jen DNS data. Konkrétně se jedná o data obsahující pouze jméno dotazovaného DNS serveru. Experiment bude probíhat pouze za využití prvních dvou datových sad. Budou se vždy porovnávat všechna zařízení z první datové sady se všemi zařízeními z druhé datové sady. Čili se porovnávají zařízení z první datové sady se zařízeními v druhé datové sadě. Výsledky toho experimentu naleznete v tabulce 9.8, kde Id zařízení ve sloupci obsahuje zařízení z první datové sady a v řádcích z druhé. Vstupní hodnoty jsou tedy v první sloupci a řádku, ve zbylých polích se nachází výstupní hodnoty, které nám vypočítal systém. Tento systém pro zobrazení vstupních a výstupních hodnot je použit i ve všech následujících tabulkách v dalších experimentech. Je zde patrné, že zařízení 14 a 15, rozhodně není v obou datových sadách stejně tak zařízení 4 a 7. Celkově zde máme šest zařízení pro které bychom měli být schopni nalézt shodu. Podařilo se nám však identifikovat pouze 4 zařízení, z toho pouze u dvou byla hodnota skóre vyšší jak 0,5. Také si, ale můžeme všimnout, že se nám nenulové hodnoty objevují i u zařízení, které se vyskytuje pouze v jedné datové sadě, což bychom nechtěli. Je zřejmé z toho jak pracuje metoda TF-IDF, že pro některá zařízení, která nejsou stejná, nebudou vždy hodnoty nulové, protože nám stačí jeden shodný term, a výsledek nebude nulový. Tuto vlastnost by však měli i další metody, které by se pro tento účel daly použít. Zařízení, která mají všude nulovou hodnotu nemají žádné shodné termy s ostatními zařízením, což je nejspíše způsobeno malou síťovou aktivitou těchto zařízení, které si můžeme všimnout v tabulkách 9.4 a 9.5. Pokud shrneme tento experiment, tak informace pouze z jména dotazovaného DNS serveru, nám nestačí k tomu, abychom byli schopní rozpoznávat zařízení v síťové komunikaci.

Id zařízení 1:2	1	2	3	8	11	13	14	15
1	**0.10519**	0.0	0.00512	0.09372	0.02494	0.00090	0.0	0.0
2	0.04864	**0.70929**	0.03260	0.08503	0.03332	0.05110	0.0	0.0
3	0.04830	0.13098	0.06370	0.05513	0.05498	**0.23638**	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	0.00723	0.0	0.01139	0.01656	0.02357	**0.13410**	0.0	0.0
6	0.0	0.17924	0.03654	0.02075	0.03944	**0.21595**	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8	0.02255	0.03955	0.03655	0.07264	0.04486	**0.32819**	0.0	0.0
9	**0.01069**	0.0	0.00717	0.00756	0.00681	0.00379	0.0	0.0
10	0.00514	0.00328	0.00519	**0.54233**	0.00563	0.00870	0.0	0.0
11	0.00782	0.02218	0.01649	0.14238	**0.64956**	0.06516	0.0	0.0
12	0.0178	0.06432	**0.37700**	0.01519	0.00849	0.02180	0.0	0.0
13	0.02573	0.0	0.05008	0.07403	0.02624	**0.19205**	0.0	0.0

Tabulka 9.8: Tabulka skóre porovnávání zařízení mezi datovými sadami 1 a 2, termy jsou pouze jména dotazovacích DNS serverů. Tučným písmem jsou v tabulce znázorněny nejvyšší hodnoty v řádcích, tedy nejlepší skóre pro zařízeni z první datové sady. Žlutě vybarvená pole vyznačují shodná zařízení mezi prvními dvěma datovými sadami.

Vyhledávání stejných zařízení v datových sadách za pomoci jména dotazovaného serveru a flagu

V tomto experimentu zkusíme data sloužící pro otisk zařízení trošku rozšířit a to tím způsobem, že přidáme k jménu DNS serveru i flag(Jedná se o typ záznamu, který je uložen na doménovém serveru nejčastěji se jedná o záznam hodnoty 1 a nebo 28, kde se jedná o překlad doménové jména na IPV4 a IPV6), který je v paketech obsažen. Výsledky, které nám vypočítal systém jsou znázorněné v tabulce 9.9. Když si data pozorně prohlédneme zjistíme, že nedošlo k žádným výrazným změnám. Hodnoty skóre se u některých kombinací dvou zařízení buď o maličko zvětšili nebo zmenšili. Tato změna je však úplně nepatrná a ani se nám nepodařilo identifikovat žádné nové shodné dvojice zařízení. Na druhou stranu jsme o žádné také nepřišli. Můžeme tedy z tohoto experimentu vyvodit, že na výsledné skóre shody nemá přidáni informace o flagu takřka žádný význam.

Id zařízení 1:2	1	2	3	8	11	13	14	15
1	**0.10519**	0.0	0.00512	0.09372	0.02495	0.00090	0.0	0.0
2	0.06285	**0.58996**	0.04213	0.07337	0.03236	0.01064	0.0	0.0
3	0.04830	0.10716	0.06370	0.05513	0.05498	**0.23638**	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	0.00723	0.0	0.01139	0.01656	0.02358	**0.13410**	0.0	0.0
6	0.0	0.12322	0.03811	0.01696	0.03920	**0.22416**	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8	0.02255	0.03236	0.03655	0.07264	0.04488	**0.32819**	0.0	0.0
9	**0.01236**	0.0	0.00829	0.00874	0.00789	0.00439	0.0	0.0
10	0.00530	0.00277	0.00536	**0.57062**	0.00581	0.00675	0.0	0.0
11	0.00782	0.01816	0.01651	0.14252	**0.65093**	0.06523	0.0	0.0
12	0.0178	0.05263	**0.37702**	0.01506	0.00841	0.02180	0.0	0.0
13	0.02592	0.0	0.05052	0.07469	0.02649	**0.19382**	0.0	0.0

Tabulka 9.9: Tabulka skóre porovnávání zařízení mezi datovými sadami 1 a 2, termy jsou jména dotazovacích DNS serverů a hodnoty flagu. Tučným písmem jsou v tabulce znázorněny nejvyšší hodnoty v řádcích, tedy nejlepší skóre pro zařízeni z první datové sady. Žlutě vybarvená pole vyznačují shodná zařízení mezi prvními dvěma datovými sadami.

Vyhledávání stejných zařízení v datových sadách za pomoci informace o šifrování z komunikace SSL/TLS

V tomto experimentu se podíváme na data ze sítové komunikace SSL/TLS, konkrétně se jedná data, která obsahují informace o tom jaké šifrovací algoritmy zařízení podporují. Výsledky porovnávání zařízení mezi první a druhou datovou sadou jsou vyobrazeny v následující tabulce 9.10. Oproti výsledkům porovnávání zařízení na základě DNS záznamům je zde patrný rozdíl výsledků skóre hned na první pohled. vyskytuje se zde daleko více nulových hodnot. Tento efekt je způsoben menším počtem záznamů v datové sadě se SSL/TLS záznamy oproti DNS záznamům. Počet záznamů SSL/TLS má totiž přibližně jen šestinu velikosti DNS záznamů. Když se podíváme jak se nám dařilo, při nacházení dvou shodných zařízení mezi datovými sadami, tak z výsledků vidíme, že se nám podařilo určit pouze tři zařízení z 5. Z těchto pěti zařízení však jen 2 mají dostatečně velkou hodnotu na to, aby jsme mohli i bez tabulky mapující IP adresy na MAC adresy říct, že jsou shodné.

Id zařízení 1:2	1	3	8	11	13
1	**0.69859**	0.0	0.0	0.0	0.0
3	0.0	0.05016	0.02721	**0.05604**	0.0
4	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0	**0.69002**
6	0.0	0.0	0.0	0.0	**0.17801**
7	0.0	0.0	0.0	0.0	0.0
8	0.0	0.04304	0.03280	**0.05145**	0.0
9	**0.14433**	0.0	0.0	0.02244	0.0
10	0.0	**0.01865**	0.01652	0.01372	0.0
11	0.0	0.02504	0.02539	**0.07204**	0.0
12	0.0	**0.03344**	0.02976	0.02847	0.0
13	0.0	0.0	0.0	0.00286	**0.55328**

Tabulka 9.10: Tabulka skóre porovnávání zařízení mezi datovými sadami 1 a 2, termy jsou hodnoty pro podporované šifrování z komunikace SSL/TLS. Tučným písmem jsou v tabulce znázorněny nejvyšší hodnoty v řádcích, tedy nejlepší skóre pro zařízeni z první datové sady. Žlutě vybarvená pole vyznačují shodná zařízení mezi prvními dvěma datovými sadami.

Vyhledávání stejných zařízení v datových sadách za pomoci informace o rozšíření z komunikace SSL/TLS

Nyní se podíváme na experiment, kde jsme k nalezení shodných zařízení použili pouze informaci o tom jaké rozšíření obsahoval Client Hello paket v síťové komunikaci pod protokolem SSL/TLS. Výsledky tohoto experimentu jsou vyobrazeny v tabulce 9.11. Z výsledků je jasně patrné, že tato informace nám k porovnávání zařízení moc nepomohla, jelikož se nám povedlo určit pouze jedno shodné zařízení, které má hodnotu větší než 0.5, zbylá dvě zařízení, která jsou ve výsledcích označena také jako shodná s nejvyšším dosaženým skóre pro dané zařízení, však mají hodnotu okolo 0.1, což nejde brát jako dobrý výsledek. Když si shrneme celkové výsledky tohoto experimentu tak vidíme, že ze všech experimentů nám nabízí nejhorší možný výsledek, podobně špatných výsledků dosáhl i experiment 9.2.

Id zařízení 1:2	1	3	8	11	13
1	**0.12474**	0.0	0.0	0.0	0.0
3	0.0	0.02641	0.02368	**0.11407**	0.01167
4	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	0.00071	0.00819	**0.60207**
6	0.0	0.0	0.00594	0.07923	**0.08879**
7	0.0	0.0	0.0	0.0	0.0
8	0.0	**0.07454**	0.02632	0.07251	0.02934
9	**0.39071**	0.0	0.0	0.00082	0.00188
10	0.0	0.02840	0.01614	0.02919	**0.03135**
11	0.0	0.07175	0.02288	**0.07473**	0.00905
12	0.0	**0.04818**	0.02841	0.03231	0.02586
13	0.0	0.0	0.0	0.00233	**0.51647**

Tabulka 9.11: Tabulka skóre porovnávání zařízení mezi datovými sadami 1 a 2, termy jsou hodnoty rozšíření z komunikace SSL/TLS. Tučným písmem jsou v tabulce znázorněny nejvyšší hodnoty v řádcích, tedy nejlepší skóre pro zařízeni z první datové sady. Žlutě vybarvená pole vyznačují shodná zařízení mezi prvními dvěma datovými sadami.

Vyhledávání stejných zařízení v datových sadách za pomoci informace o šifrování i rozšíření z komunikace SSL/TLS

Tento experiment spočívá v tom, že se pokusíme najít shodná zařízení z prvních dvou datových sad za pomoci všech informací, které by mohli být užiteční ze síťové komunikace protokolu SSL/TLS. Jedná se konkrétně o data obsahují informace o šifrovacích algoritmech, které zařízení podporují a rozšířeních, které byli součástí Client Hello paketu. Výsledky toho experimentu jsou vyobrazeny v tabulce 9.12. Je zde patrné, že tento experiment nám nepřinesl tížené ovoce a výsledek patří opět k těm nejhorším, které v experimentech naleznete. Opět se nám podařilo identifikovat pouze tři zařízení z pěti možných, při čemž v tomto experimentu jen jedno skóre se blíží hodnotě 0.5. Ostatní zařízení, která se nám povedlo identifikovat mají výsledné skóre okolo hodnoty 0.1, což není potřebný výsledek pro to, aby jsme byli schopni říci, že jsou opravdu tyto dvě zařízení shodná bez informace o tom, jakou mají MAC adresu.

Id zařízení 1:2	1	3	8	11	13
1	**0.11145**	0.0	0.0	0.0	0.0
3	0.0	0.02854	0.00985	**0.07917**	0.0
4	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0	**0.48168**
6	0.0	0.0	0.0	0.0	**0.05802**
7	0.0	0.0	0.0	0.0	0.0
8	0.0	0.07144	0.01905	**0.08525**	0.0
9	**0.44584**	0.0	0.0	0.0	0.0
10	0.0	**0.03190**	0.01003	0.01967	0.0
11	0.0	0.06045	0.01850	**0.08502**	0.0
12	0.0	**0.04987**	0.01957	0.02542	0.0
13	0.0	0.0	0.0	0.0	**0.49622**

Tabulka 9.12: Tabulka skóre porovnávání zařízení mezi datovými sadami 1 a 2, termy jsou hodnoty pro podporované šifrování a rozšíření z komunikace SSL/TLS. Tučným písmem jsou v tabulce znázorněny nejvyšší hodnoty v řádcích, tedy nejlepší skóre pro zařízeni z první datové sady. Žlutě vybarvená pole vyznačují shodná zařízení mezi prvními dvěma datovými sadami.

Vyhledávání stejných zařízení v datových sadách za pomoci všech dostupných informací z DNS a šifrování ze SSL/TLS

Tento experiment kombinuje jak DNS tak SSL/TLS informace k vytváření termů. Z DNS záznamů si bere jak jméno DNS jmenného serveru, tak hodnotu flagu. Ze SSL/TLS získává pouze informaci o podporovaných šifrovacích algoritmech. Tato kombinace je zajímavá tím, že výsledky, které jsme z ní získaly jsou nejlepší jakých se nám povedlo docílit. Podařilo se nám sice stále určit jen 4 zařízení shodná, avšak žádná neshodná zařízení nemají hodnotu vyšší než 0,39. Když se však podíváme na tabulky 9.4 a 9.5, či 9.6 a 9.7 a blíže se podíváme na rozdíly mezi počty záznamů v první a druhé datové sadě zjistíme, že všechna zařízení, která se nám podařilo identifikovat mají v prvním datové sadě více záznamů než v druhém. Pro zbylé dvě zařízení, které jsme neidentifikovali je tomu přesně naopak. Tento experiment byl nejúspěšnější ze všech, které byli provedeny a poukazuje na fakt, že pokud bychom měli potřebně velkou datovou sadu pro jednotlivá zařízení a pokusili se je identifikovat v menších datových sadách měli bychom zřejmě větší šanci na úspěch. Výstup systému pro tento experiment je možné nalézt v tabulce 9.13.

Id zařízení 1:2	1	2	3	8	11	13	14	15
1	**0.44671**	0.0	0.00079	0.00334	0.00347	0.00014	0.0	0.0
2	0.04423	**0.58996**	0.03953	0.01737	0.02859	0.01039	0.0	0.0
3	0.02791	0.08928	0.06061	0.02454	0.05440	**0.18267**	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	0.00456	0.0	0.00957	0.00349	0.01730	**0.18544**	0.0	0.0
6	0.0	0.09593	0.02626	0.00311	0.02476	**0.19716**	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8	0.01106	0.02270	0.03589	0.03310	0.04393	**0.20902**	0.0	0.0
9	**0.06875**	0.0	0.00538	0.00143	0.01233	0.00296	0.0	0.0
10	0.00346	0.00258	0.00691	**0.10905**	0.00708	0.00603	0.0	0.0
11	0.00405	0.01339	0.01747	0.03651	**0.39639**	0.04580	0.0	0.0
12	0.0124	0.05224	**0.33600**	0.00633	0.00899	0.02069	0.0	0.0
13	0.00780	0.0	0.01892	0.00690	0.01047	**0.19150**	0.0	0.0

Tabulka 9.13: Tabulka skóre porovnávání zařízení mezi datovými sadami 1 a 2, termy jsou jména dotazovacích DNS serverů, hodnoty flagu a možnosti šifrování protokolu SSL/TLS. Tučným písmem jsou v tabulce znázorněny nejvyšší hodnoty v řádcích, tedy nejlepší skóre pro zařízeni z první datové sady. Žlutě vybarvená pole vyznačují shodná zařízení mezi prvními dvěma datovými sadami.

Vyhledávání stejných zařízení v datových sadách za pomoci všech dostupných informací z DNS a šifrování i rozšíření z SSL/TLS komunikace

Tento poslední experiment využívá pro identifikaci mobilních zařízeních v síťové komunikaci veškeré informace, které mám z prvních dvou datových sad a jsou dostatečně využitelné pro tuto identifikaci. K vytvoření otisku zařízení je v tomto experimentu tedy využito jako jméno DNS serveru tak flagy, také jsme však použili možnosti šifrovacích zařízení a rozšíření, které nám nabízí SSL/TLS část datových sad. Používáme tedy maximální kombinaci informací, kterou můžeme využít. Výsledky, které nám systém po zadání potřebných parametrů vypočítal jsou znázorněny v tabulce 9.14. Z výsledků je patrně, že né vždy je dobré využít všechny data, jelikož výsledky toho experimentu jsou o malinko horší než v experimentu 9.2, a to hlavně proto, že se nám skóre podobnosti rapidně zmenšilo pro zařízení číslo 1. Z tabulky 9.10 je patrné proč tomu tak je. První zařízení se nám totiž velice úspěšně podařilo identifikovat při použití pouze šifrovacího algoritmu, kdežto když jsme tuto informaci spojili s rozšířením v experimentu 9.2 tak se skóre podobnosti zmenšilo zhruba o 0.5, a proto je v tomto experimentu skóre podobnosti daleko menší. Kromě této změny u prvního zařízení je výsledek celkem obstojný, podařilo se nám určit čtyři zařízení ze šesti, i když pouze dvě z těchto zařízení mají hodnotu vyšší jak 0,5.

Id zařízení 1:2	1	2	3	8	11	13	14	15
1	**0.08364**	0.0	0.00091	0.00438	0.00419	0.00016	0.0	0.0
2	0.04423	**0.58996**	0.04153	0.02085	0.03165	0.01052	0.0	0.0
3	0.02791	0.08928	0.05399	0.01827	0.05242	**0.18688**	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	0.00473	0.0	0.01042	0.00434	0.02029	**0.15160**	0.0	0.0
6	0.0	0.10353	0.03019	0.00404	0.03018	**0.18799**	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8	0.01290	0.02641	0.03674	0.02654	0.04493	**0.25319**	0.0	0.0
9	**0.07872**	0.0	0.00784	0.00237	0.00741	0.00416	0.0	0.0
10	0.00354	0.00264	0.00658	**0.13048**	0.00660	0.00631	0.0	0.0
11	0.00484	0.01597	0.01907	0.03689	**0.54033**	0.05601	0.0	0.0
12	0.01249	0.05246	**0.36685**	0.00541	0.00862	0.02126	0.0	0.0
13	0.01150	0.0	0.02979	0.01246	0.01534	**0.18788**	0.0	0.0

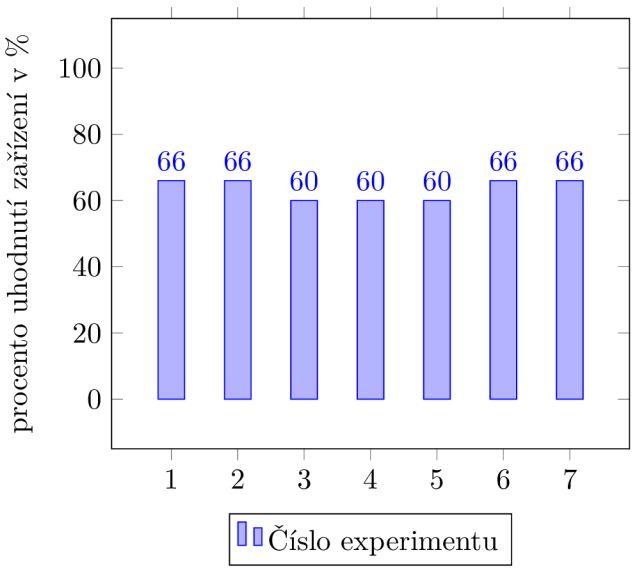
Tabulka 9.14: Tabulka skóre porovnávání zařízení mezi datovými sadami 1 a 2, termy jsou jména dotazovacích DNS serverů, hodnoty flagu a možnosti šifrování i rozšíření protokolu SSL/TLS. Tučným písmem jsou v tabulce znázorněny nejvyšší hodnoty v řádcích, tedy nejlepší skóre pro zařízeni z první datové sady. Žlutě vybarvená pole vyznačují shodná zařízení mezi prvními dvěma datovými sadami.

Grafové zobrazení uhodnutých zařízení

Nyní se podíváme na to jak jsme byli úspěšný v jednotlivých experimentech v hledání shodných zařízení. V grafu 9.1 je znázorněna procentuální úspěšnost jednotlivých experimentů při čemž, čísla odkazují na jednotlivé experimenty, jejich křížový odkaz je znázorněn v tabulce 9.15 . Můžeme si povšimnout, že se nám v nejhorším případě povedlo uhodnout 60% zařízení. Ovšem zde není znázorněno to jak byla hodnota skóre podobnosti vysoká, nýbrž to kdy nejvyšší hodnota byla pro dané zařízení z první datové sady náležela stejnému zařízení v druhé datové sadě.

Číslo experimentu v grafu	Křížový odkaz na popis experimentu:
1	Experiment 9.2
2	Experiment 9.2
3	Experiment 9.2
4	Experiment 9.2
5	Experiment 9.2
6	Experiment 9.2
7	Experiment 9.2

Tabulka 9.15: Tabulka křížových odkazů na jednotlivé experimenty



Obrázek 9.1: Graf zobrazující procentuální úspěšnost při hledání zařízení

Kapitola 10

Závěr

Cílem této práce je vytvořit systém, který bude sloužit k ověření toho, do jaké míry lze využít data z DNS a SSL/TLS komunikace pro profilování mobilních zařízení a určení shody těchto zařízení. Na této práci je velice zajímavé především to, že se snažíme určit zařízení na základě běžné komunikace. K identifikaci zařízení se totiž používá nezašifrovaná komunikace v síti, kterou vytváří v dnešní době každý člověk s přístupem na internet. Další neméně zajímavou částí je metoda, kterou systém pro určení zařízení používá a hlavně jakým způsobem.

Na začátku práce se nachází stručný popis protokolů, z jejichž komunikace vychází datové sady, se kterými se bude dále pracovat. Jedná se o protokoly DNS, SSL a TLS. U každého protokolu se dá nalézt základní popis a funkcionalita. Dále se zde dá dočíst o tom, jak probíhá komunikace mezi zařízeními pomocí těchto protokolů. Následující část práce se věnuje předzpracování dat před použitím metod pro dolování dat. Pojednává o tom, jak se data čistí a upravují, aby pak metody dosáhly co možná nejlepších výsledků. Na toto téma navazuje další část práce, která shrnuje základní vlastnosti dolování dat. Věnuje se hlavně nejznámějším a nejdůležitějším základním metodám určeným k dolování dat. Následující část práce je zaměřená na získávání dat z komunikace mezi mobilními zařízeními. Pojednává o identifikaci těchto zařízení díky komunikaci s okolním světem za pomoci internetu a jak z dat, které po sobě zařízení nechávají, či jiné aplikace sbírají vytvářet otisk těchto zařízení.

V další části se práce zabývá hlavně datovými sadami. Jejich obsahem a úpravou, která by měla usnadnit vývoj aplikace na identifikaci mobilních zařízení a zajistit lepší práci s daty. Důležitá je především informace o tom, jaké možnosti zpracování těchto vstupních dat jsou na výběr a jaké jsou jejich výhody, či nevýhody.

Dále se práce shrnuje postup pro návrh systému, který analyzuje data ze síťové komunikace mobilních zařízení a také implementace toho systému. Jsou zde popsány veškeré informace i tom, jak se při návrhu postupovalo, a vyobrazeny veškeré diagramy, které při návrhu i implementaci pomohli pro názornější pochopení daného problému. Jsou zde také zmíněny užitečné informace a poznatky při implementaci. Nechybí ani zajímavé části systému, mezi které určitě patří implementace metody TF-IDF a následná implementace funkcí pro vypočítání výsledného skóre podobnosti.

Tento systém se nám tedy povedlo implementovat a následně na něm bylo provedena série testů a experimentů. Mezi nejzajímavější experimenty patří kombinace informací z dvou datových sad, kdy jedna obsahuje informace z DNS komunikace a druhá z komunikace pod protokolem SSL/TLS. Bohužel je však jasně patrné, že tato metoda pro určování shodných zařízení není stoprocentní a ani kombinace záznamů DNS a SSL/TLS nám k tomu

nepomohla. Nicméně pár zařízení systém dokázal určit s uspokojivým výsledkem. Kdyby se podařilo systém vylepšit v přesnosti, mohly být výsledky nejspíše lepší.

Závěrem bych rád shrnul dosažené výsledky této práce. Systém pro analýzu síťových dat se podařilo úspěšně implementovat i otestovat. Bohužel však výsledky při porovnávání zařízení na základě otisků, vytvořených z analýzy síťové komunikace protokolů DNS a SSL/TLS, nebyly až tolik uspokojivé, ale ani úplně špatné. V některých případech se nám zařízení povedlo identifikovat velice úspěšně. Jednalo se především o ty zařízení, která měla v sadě, kde se vytvářel otisk zařízení, více záznamů než v datové sadě, kde se dané zařízení vyhledávalo. Z čehož vyplývá, že pro zlepšení funkcionality celého systému by bylo dobré doplnit nějaké rozšíření pro lepší dosažení výsledků ať už by se jednalo o kombinování informací z více datových sad do jednoho velkého datového balíčku nebo lepší ohodnocení termů v zařízení.

Literatura

[1] *Architecture of a simple neural network* [Packt]. [Online; navštíveno 10.5.2020]. Dostupné z: https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781789346640/1/ch01lvl1sec12/architecture-of-a-simple-neural-network.

[2] *Biologické algoritmy (4) - Neuronové sítě* [Root.cz]. [Online; navštíveno 2.1.2020]. Dostupné z: https://www.root.cz/clanky/biologicke-algoritmy-4-neuronove-site/.

[3] *Biologické algoritmy (5) - Neuronové sítě* [Root.cz]. [Online; navštíveno 2.1.2020]. Dostupné z: https://www.root.cz/clanky/biologicke-algoritmy-5-neuronove-site/.

[4] *Data Mining* [SAS Insights Inc.]. [Online; navštíveno 28.12.2019]. Dostupné z: https://www.sas.com/cs_cz/insights/analytics/data-mining.html#dmworld.

[5] *DATAMININGOVÉ MODELY: Asociační pravidla a analýza sekvencí* [ACREA CR, spol. s r.o.]. [Online; navštíveno 1.1.2020]. Dostupné z: https://kurzystatistiky.cz/kurzy/data-mining/dataminingove-modely-asociacni-pravidla-a-analyza-sekvenci/.

[6] *Domain Name System* [Wikipedia: the free encyclopedia]. [Online; navštíveno 23.12.2019]. Dostupné z: https://cs.wikipedia.org/wiki/Domain_Name_System.

[7] *Domain Name System* [Wikipedia: the free encyclopedia]. [Online; navštíveno 23.12.2019]. Dostupné z: https://en.wikipedia.org/wiki/Domain_Name_System.

[8] *Historie SSL/TLS certifikátů* [SSLmentor blog]. [Online; navštíveno 25.12.2019]. Dostupné z: https://blog.sslmentor.cz/clanky/historie-ssl-tls-certifikatu/.

[9] *Koncept umělé neuronové sítě* [Institut biostatistiky a analýz Lékařské fakulty Masarykovy univerzity]. [Online; navštíveno 3.1.2020]. Dostupné z: https://portal.matematickabiologie.cz/index.php?pg=analyza-a-hodnoceni-biologickych-dat--umela-inteligence--neuronove-site-jednotlivy-neuron--uvod-do-neuronovych-siti--koncept-umele-neuronove-site.

[10] *Korelační a regresní analýza* [wikisofia]. [Online; navštíveno 11.1.2020]. Dostupné z: https://wikisofia.cz/wiki/Korela%C4%8Dn%C3%AD_a_regresn%C3%AD_anal%C3%BDza.

[11] *Neuronové sítě* [Mendelova univerzita v Brně]. [Online; navštíveno 8.1.2020]. Dostupné z: https://is.mendelu.cz/eknihovna/opory/zobraz_cast.pl?cast=21471.

[12] *Odemykáme potenciál AI Machine Learning v e-commerce - otevíráme černou skříňku (část 5)* [Ecommerce-Academy.cz]. [Online; navštíveno 10.5.2020]. Dostupné z: https://www.ecommerce-academy.cz/post/serial-uvod-do-ai-5.

[13] *Regresní a korelační analýza* [Lean Six Sigma]. [Online; navštíveno 10.1.2020]. Dostupné z: https://lean6sigma.cz/regresni-a-korelacni-analyza/.

[14] *Rozhodovací stromy a chytré otázky* [imysleni.cz]. [Online; navštíveno 28.12.2019]. Dostupné z: https://popelka.ms.mff.cuni.cz/~lessner/mw/index.php/U%C4%8Debnice/Informace/Rozhodovac%C3%AD_stromy_a_chytr%C3%A9_ot%C3%A1zky.

[15] *STD 13 RFC 1035* [RFC Editor]. [Online; navštíveno 25.12.2019]. Dostupné z: https://www.rfc-editor.org/info/rfc1035.

[16] *tf–idf* [Wikipedia: the free encyclopedia]. [Online; navštíveno 15.1.2020]. Dostupné z: https://en.wikipedia.org/wiki/Tf-idf.

[17] *The Illustrated TLS Connection* [XargsNotBombs]. [Online; navštíveno 16.5.2020]. Dostupné z: https://tls.ulfheim.net/.

[18] *TLS* [SSLS.CZ]. [Online; navštíveno 25.12.2019]. Dostupné z: https://www.ssls.cz/slovnik/tls.html#ssl.

[19] *TLS Security 5: Establishing a TLS Connection* [acunetix]. [Online; navštíveno 26.12.2019]. Dostupné z: https://www.acunetix.com/blog/articles/establishing-tls-ssl-connection-part-5/.

[20] *Transport Layer Security* [Wikipedia: the free encyclopedia]. [Online; navštíveno 25.12.2019]. Dostupné z: https://cs.wikipedia.org/wiki/Transport_Layer_Security.

[21] *Transport Layer Security* [Wikipedia: the free encyclopedia]. [Online; navštíveno 25.12.2019]. Dostupné z: https://en.wikipedia.org/wiki/Transport_Layer_Security.

[22] *Úvod do data miningu* [StatSoft CR s.r.o.]. [Online; navštíveno 28.12.2019]. Dostupné z: http://www.statsoft.cz/file1/PDF/newsletter2014_02_26_StatSoft_Uvod_do_data_miningu.pdf.

[23] *Čištění dat (Data cleansing)* [ManagementMania.com]. [Online; navštíveno 11.1.2020]. Dostupné z: https://managementmania.com/cs/cisteni-dat-data-cleansing.

[24] Berka, P. *Dobývání znalostí z databází*. Academia, 2003. 366 s. ISBN 80-200-1062-9.

[25] Christopher D. Manning, H. S. *An Introduction to Information Retrieval*. Cambridge University Press, 2008. 482 s. ISBN 05-218-6571-9.

[26] Holčík, J. *Analýza a klasifikace dat* [Institut biostatistiky a analýz, Lékařská fakulta Masarykovy univerzity]. [Online; navštíveno 22.5.2020]. Dostupné z: https://www.iba.muni.cz/res/file/ucebnice/holcik-analyza-klasifikace-dat.pdf.

[27] Kučera, J. *Shluková analýza* [Masarykova Univerzita v Brně]. [Online; navštíveno 13.1.2020]. Dostupné z: https://is.muni.cz/th/172767/fi_b/5739129/web/web/main.html.

[28] Matoušek, P. *Síťové služby a jejich architektura*. Publishing house of Brno University of Technology VUTIUM, 2014. 396 s. ISBN 978-80-214-3766-1.

[29] Michaela Koščová, J. M. *SHLUKOVÁ ANALÝZA)* [CzechEncy - Nový encyklopedický slovník češtiny]. [Online; navštíveno 13.1.2020]. Dostupné z: https://www.czechency.org/slovnik/SHLUKOV%C3%81%20ANAL%C3%9DZA.

[30] Michal Krátký, V. S. *Vícerozměrný přístup pro netriviální vyhledávání termů* [Katedra informatiky, FEI, VŠB–Technická univerzita Ostrava]. [Online; navštíveno 12.5.2020]. Dostupné z: http://siret.ms.mff.cuni.cz/skopal/pub/znalTerms.pdf.

[31] Nick Nikiforakis, W. J. C. K. F. P. G. V. *Cookieless Monster: Exploring the Ecosystem of Web-Based Device Fingerprinting* [IEEE]. [Online; navštíveno 21.4.2020]. Dostupné z: https://ieeexplore.ieee.org/abstract/document/6547132.

[32] Patel, V. *Device Fingerprinting for Mobile Attribution* [GetSocial a keywords studio]. [Online; navštíveno 15.1.2020]. Dostupné z: https://blog.getsocial.im/device-fingerprinting-for-mobile-attribution/.

[33] Rabušic, L. *Mnohonásobná lineární regrese* [Informační systém Masarykovy univerzity]. [Online; navštíveno 18.5.2020]. Dostupné z: https://is.muni.cz/el/1423/podzim2004/SOC418/multipl_regres_1.pdf.

[34] Rychlý, M. *Klasifikace a predikce* [Ústav informačních systémů, Vysoké učení technické v Brně]. [Online; navštíveno 22.5.2020]. Dostupné z: http://www.fit.vutbr.cz/~rychly/public/docs/classification-and-prediction/classification-and-prediction.pdf.

[35] Salašová, M. P. *SSL a TLS, rozdíly, o kterých jste možná nevěděli!* [SSL market od ZONER software]. [Online; navštíveno 25.12.2019]. Dostupné z: https://blog.sslmarket.cz/inpage/ssl-tls-rozdily-o-kterych-jste-mozna-nevedeli/.

[36] Seiner, M. *Asociační analýza - příčiny a následky dopravních nehod*. 2014. 77 s.

[37] Stecanella, B. *What is TF-IDF?* [MonkeyLearn]. [Online; navštíveno 15.1.2020]. Dostupné z: https://monkeylearn.com/blog/what-is-tf-idf/.

[38] Trejbal, P. *Jak na rozhodovací stromy* [optimics]. [Online; navštíveno 31.12.2019]. Dostupné z: https://www.optimics.cz/jak-na-rozhodovaci-stromy/.

Příloha A

Obsah CD

Na přiloženém CD jsou uloženy veškeré zdrojové soubory, dokumentace k použitým knihovnám a systému. Také zde manuál pro úspěšné spuštění systému a datové sady se, kterými byli prováděny experimenty. Zde naleznete přesný popis adresářové struktury, která je uložena na CD disku:

- **tex** zdrojové LATEX soubory pro sestavení PDF souboru této práce
- **tex\obrazky** obrázky použité v této práci
- **pdf** tento PDF soubor
- **manual** manuál pro spuštění systému
- **DataSety\DNS** datové sady s DNS záznamy
- **DataSety\SSL** datové sady se SSL/TLS záznamy
- **DataSety\mac-ip** tabulky pro mapování IP adres na MAC adresy
- **src** zdrojové soubory pro systém pro analýzu dat síťové komunikace mobilních zařízení

Příloha B

Manuál pro spuštění systému

Následují text slouží jako jednoduchý návod ke spuštění systému.

1. Nainstalovat lokální MySQL server, který se dá stáhnout z https://www.apachefriends.org/index.htm
2. Po nainstalování serveru ho spustíme. Stačí nám spustit pouze Apache a MySQL server.
3. Nainstalujeme Python (pro vývoj byla použita verze 3.7.2), který je dostupný z https://www.python.org/downloads/release/python-372/
4. Dále je zapotřebí doinstalovat následující knihovny:
   - sklearn
   - pandas
   - numpy
   - collections
   - csv
   - os
   - pymysql
   - sys
5. Nyní už stačí jen nastavit parametry pro systém v souboru options.txt.
6. Spustit hlavní třídu systému main.py.