FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání diplomové práce

Název:	Klasifikace síťového provozu pomocí strojového učení
Student:	Bc. Matej Hulák
Vedoucí:	Ing. Karel Hynek
Studijní program:	Informatika
Obor / specializace:	Počítačová bezpečnost
Katedra:	Katedra informační bezpečnosti
Platnost zadání:	do konce letního semestru 2022/2023

Pokyny pro vypracování

Seznamte se s principy monitorování síťového provozu pomocí rozšířených síťových toků a s metodami strojového učení používanými pro jejich analýzu.
Po konzultaci s vedoucím práce vyberte několik síťových protokolů a vytvořte datové sady.
Nad vytvořenými datovými sadami proveďte experimenty s různými modely strojového učení, kde úlohou bude rozpoznání síťového protokolu použitého v daném spojení, a to pouze na základě síťových statistik (tj. bez využití čísla portu a IP adres).

Porovnejte modely strojového učení pracující nad rozšířenými síťovými toky s modely využívajícími pouze základní toky. Dále porovnejte výsledky nových klasifikátorů s výsledky relevantní práce [1].

Vybranou variantu modelu implementujte jako modul pro systém NEMEA schopný klasifikovat provoz v reálném čase. Funkčnost implementovaného modulu otestujte na reálných síťových datech.

[1] Hulák, Matej. Klasifikace provozu a zařízení v počítačových sítích na základě toků. Bakalářská práce. Praha: ČVUT FIT, 2020

Elektronicky schválil/a prof. Ing. Róbert Lórencz, CSc. dne 13. listopadu 2021 v Praze.

Diplomová práca

Klasifikácia sieťovej premávky pomocou strojového učenia

Bc. Matej Hulák

Katedra informačnej bezpečnosti
Vedúci práce: Ing. Karel Hynek

1. mája 2022

Poďakovanie

V prvom rade by som chcel poďakovať svojmu vedúcemu práce Ing. Karlu Hynkovi za odborné vedenie, rady a ústretovosť. Ďalej by som chcel poďakovať svojej rodine, ktorá ma celú dobu podporovala a umožnila mi štúdium na vysokej škole.

Prehlásenie

Prehlasujem, že som predloženú prácu vypracoval(a) samostatne a že som uviedol(uviedla) všetky informačné zdroje v súlade s Metodickým pokynom o etickej príprave vysokoškolských záverečných prác.

Beriem na vedomie, že sa na moju prácu vzťahujú práva a povinnosti vyplývajúce zo zákona č. 121/2000 Sb., autorského zákona, v znení neskorších predpisov. V súlade s ustanovením § 46 odst. 6 tohoto zákona týmto udeľujem bezvýhradné oprávnenie (licenciu) k užívaniu tejto mojej práce, a to vrátane všetkých počítačových programov ktoré sú jej súčasťou alebo prílohou a tiež všetkej ich dokumentácie (ďalej len „Dielo“), a to všetkým osobám, ktoré si prajú Dielo užívať. Tieto osoby sú oprávnené Dielo používať akýmkoľvek spôsobom, ktorý neznižuje hodnotu Diela (vrátane komerčného využitia). Toto oprávnenie je časovo, územne a množstevne neobmedzené.

V Prahe 1. mája 2022 . . . . . .

Odkaz na túto prácu

Hulák, Matej. *Klasifikácia sieťovej premávky pomocou strojového učenia.* Diplomová práca. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2022.

Abstrakt

Práca sa zameriava na aspekty a činitele, ktoré ovplyvňujú úspešnosť klasifikácie sieťovej premávky pomocou strojového učenia. Prvá časť práce popisuje základy počítačových sietí a ich monitorovania, existujúce metódy klasifikácie a princípy strojového učenia. Praktická časť práce skúma možnosti klasifikácie sieťovej premávky pri použití rôzne obsiahlych dátových sád a rôznych metód strojového učenia. Posledná časť práce sa venuje návrhu a vytvoreniu klasifikačného modulu pre systém NEMEA, ktorý je schopný klasifikovať rozšírené sieťové toky v reálnom čase.

Výsledkom tejto práce je anotovaná dátová sada, obsahujúca rozšírené sieťové toky, sada experimentov skúmajúca možnosti klasifikácie sieťových tokov a klasifikačný modul pre systém NEMEA.

Klíčová slova klasifikácia, analýza sieťovej premávky, strojové učenie, NEMEA, sieťové toky

Abstract

This thesis focuses on the aspects and factors which affect the success of network traffic classification using machine learning. The first part of the thesis describes the basics of computer networks and their monitoring, existing classification methods and machine learning principles. The practical part of the thesis explores the possibilities of classifying network traffic using datasets of various features and different machine learning methods. The final part of the thesis deals with the design and development of a classification module for the NEMEA system, which is able to classify extended network flows in real time.

The outcome of this thesis includes an annotated dataset containing extended network flows, a set of experiments exploring the possibilities of classifying network flows, and a classification module for the NEMEA system.

Keywords classification, network traffic analysis, machine learning, NEMEA, network flows

Obsah

Zoznam obrázkov

Zoznam tabuliek

Úvod

Vo svete informačných technológií je prepojenie systémov jednou zo základných potrieb a skoro každé zariadenie pripojené na internet. Pripojenie na sieť internet umožňuje zariadeniam komunikovať s nespočetným množstvom zariadení po celom svete. Rozmach sociálnych sietí a inteligentných prístrojov v posledných rokoch značne zvýšilo počet zariadení, ktoré sa k sieti pripájajú a taktiež objem informácií, ktoré sa cez sieť prenášajú.

Aj vďaka týmto faktorom sa množstvo sieťovej premávky za posledných desať rokov zvýšilo viac ako 10-násobne [1]. Nárastom sieťovej premávky však nevzniká len potreba úpravy infraštruktúry sietí, ale aj potreba vývoja nových systémov, ktoré sú schopné túto premávku spracovávať. Medzi tieto systémy patria aj systémy, ktoré zabezpečujú bezpečnosť a správu týchto sietí. S veľkosťou siete súčasne narastá aj potreba efektívneho monitorovania premávky na sieti. Monitorovanie sieťovej premávky umožňuje bezpečnostným analytikom a správcom sietí odhaliť a agilne reagovať na incidenty a zmeny v sieti.

Táto práca sa zameriava na klasifikáciu sieťového protokolu bez využitia čísla portu. V rámci svojej bakalárskej práce som sa tejto problematike už venoval a vytvoril som klasifikačný modul, ktorý využíva štatistické informácie základných sieťových tokov, na klasifikáciu sieťového protokolu. Výsledky bakalárskej práce ukázali, že sieťový protokol je možno identifikovať aj bez použitia čísla portu. Touto problematikou som sa zaoberal ďalej a cieľom tejto práce je preskúmať možnosti použitia rozšírených sieťových tokov a metód strojového učenia na klasifikáciu sieťového protokolu.

V tejto práci sa zoznámim s novými formátmi sieťových tokov a preskúmam možnosti metód strojového učenia. Následne vytvorím sadu experimentov, v ktorých budem skúmať zmeny úspešnosti klasifikácie pri použití rozdielne obsiahnutých dátových sád a rôznych klasifikačných metód strojového učenia. Na základe výsledkov experimentov navrhnem a implementujem klasifikačný modul, ktorý bude schopný klasifikovať sieťové toky v reálnom čase.

Kapitola 1

Ciele práce

Táto práca sa zameriava na experimenty s metódami strojového učenia pri klasifikácií sieťových tokov. Hlavným cieľom je vyhodnotenie výsledkov jednotlivých klasifikačných metód strojového učenia pri klasifikácií sieťových tokov. Ďalším cieľom tejto práce je porovnanie úspešnosti klasifikácie pri použití dátových sád, ktoré obsahujú rozdielne formáty sieťových tokov.

K tomu je potrebné sa zoznámiť s princípmi monitorovania sieťovej premávky s pomocou rozšírených sieťových tokov, navrhnúť a vytvoriť vhodné dátové sady a preskúmať metódy strojového učenia.

Výsledkom tejto práce je anotovaná dátová sada obsahujúca rozšírené sieťové toky, sada experimentov skúmajúca možnosti klasifikácie sieťových tokov a klasifikačný modul pre systém NEMEA.

Výsledkom tejto práce bude anotovaná dátová sada, sada experimentov skúmajúca možnosti klasifikácie sieťových tokov a klasifikačný modul systému NEMEA, ktorý bude schopný klasifikovať premávku v reálnom čase.

Kapitola 2

Analýza a rešerš

V tejto práci som sa zoznámil s technológiami počítačových sietí, monitorovacími nástrojmi a klasifikačnými nástrojmi. Veľká časť analýzy a rešerše bola venovaná metódam strojového učenia s ktorými som nemal predchádzajúce skúsenosti. Nasledujúca kapitola popisuje potrebné znalosti a nástroje, s ktorými som pracoval.

2.1 Počítačové siete

Výmena dát medzi zariadeniami je jedna z najzákladnejších potrieb vo svete informačných technológií. Sprostredkovanie tejto potreby však nie je jednoduché. Je potrebné zabezpečiť nie len fyzické prepojenie zariadení, ale aj kompatibilitu komunikácie. S rastúcim množstvom zariadení bolo potrebné pravidlá komunikácie normalizovať a preto vznikli komunikačné štandardy, ktoré definujú pravidlá komunikácie. Z niekoľkých spojených zariadení sa stáva sieť. V súčasnej dobe je najväčšia a najrozšírenejšia sieť internet. Zariadenia z celého sveta môžu pomocou tejto siete komunikovať na enormné vzdialenosti. Sieť internet spája milióny zariadení a preto musia byť pravidlá komunikácie jasne stanovené štandardmi.

Sieťový protokol je konvencia alebo štandard, ktorý definuje syntax, sémantiku a synchronizáciu sieťovej komunikácie. Jedným z najrozšírenejších protokolov siete internet je protokol TCP/IP. Tento protokol obsahuje sadu protokolov pre komunikáciu v sieti. Vzhľadom na počet protokolov a zložitosť problému boli definované sieťové vrstvy [2]. Sieťové vrstvy znázorňujú hierarchiu činností a výmena informácií medzi vrstvami je presne definovaná. Sieťová vrstva využíva služby nižších vrstiev a zároveň poskytuje služby vyš-

ším vrstvám. Architektúra TCP/IP je členená do štyroch vrstiev :

- aplikačná vrstva
- transportná vrstva
- sieťová vrstva
- fyzická vrstva

Fyzická vrstva je hierarchicky najnižšia a definuje pravidlá pre prístup k fyzickému médiu. Sieťová vrstva obsahuje protokoly, ktoré zabezpečujú adresovanie a smerovanie dát v sieti. Transportná vrstva zabezpečuje dátový kanál medzi zariadeniami a zabezpečuje celistvosť prenášaných dát. Aplikačná vrstva je hierarchicky najvyššia a slúži k prenosu konkrétnych dát. Táto vrstva poskytuje aplikáciám prístup ku komunikačnej sieti a zabezpečuje všetku potrebnú komunikáciu pre aplikácie a užívateľov. Každý z protokolov na tejto vrstve je špecializovaný na konkrétne potreby. Medzi najpoužívanejšie protokoly tejto vrstvy patrí http, dhcp a dns. [3]

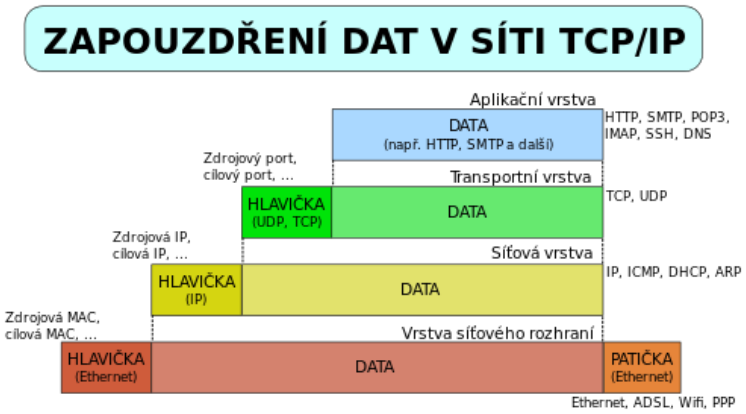
Kvôli jednoduchosti rozlíšenia jednotlivých protokolov na aplikačnej vrstve boli jednotlivým protokolom pridelené konkrétne čísla portov, ktoré by mali používať. Vďaka tomu je aj pri komunikácií s neznámym zariadením možné predpokladať, na akom porte pracujú konkrétne služby. O prideľovanie a spravovanie čísiel portov sa stará spoločnosť IANA[4]. Príklady protokolov aplikačnej vrstvy a ich čísel portu sú zobrazené v tabuľke 2.1:

Tabuľka 2.1: Protokoly aplikačnej vrstvy a ich čísla portov. [4]

Protokol	**Číslo portu**
http	80
https	443
dns	53
imap	143
ssh	22

Dáta nemôžu byť prenášané v jednom bloku, kvôli obmedzeniam sietí. Dáta sú v závislosti na použitom protokole transportnej vrstvy rozdelené do menších správ, ktoré nazývame pakety. Aby však tieto pakety mohli byť doručené adresátovi, je potrebné k nim pridať informácie o adresátovi samotnom a o použitom spôsobe prenosu. Tieto údaje sa ukladajú do hlavičiek paketu. Každý paket obsahuje niekoľko hlavičiek, pričom každá hlavička reprezentuje použitý protokol. Jednotlivé hlavičky obaľujú prenášané dáta a hlavičky vyšších vrstiev. Tento spôsob obalenia prenášaných dát sa nazýva zapuzdrenie paketu. Príklad zapuzdrenia paketu je zobrazený na obrázku 2.1. [5]

Maximálna veľkosť paketu je vo väčšine bežných sietí nastavená na 1,5 kB [6]. To však znamená, že aj na prenesenie jednoduchej správy, alebo obrázku je



Obr. 2.1: Zapuzdrenie paketu v sieti TCP/IP [5]

nutné preniesť postupnosť viacerých paketov. Bežný stolový počítač tak môže za hodinu prijať a odoslať stovky tisíc paketov. Reprezentácia prenášaných dát pomocou paketov je pre monitorovanie väčších sietí veľmi neefektívna. Z tohto dôvodu bol definovaný pojem sieťový tok.

Sieťový tok reprezentuje komunikáciu medzi dvoma zariadeniami, teda postupnosť paketov, ako je popísané napríklad v práci [7]. Sieťový tok je možné definovať podľa množiny informácií, ktoré majú všetky pakety príslušné k tomuto toku rovnaké a to odosielateľa, príjemcu, transportný protokol a použité čísla portov. Sieťový tok v základnej podobe obsahuje nasledujúce charakteristiky :

- IP adresa odosielateľa
- IP adresa príjemcu
- Číslo portu odosielateľa
- Číslo portu príjemcu
- Počet prenesených bajtov
- Časová známka
- Použitý protokol transportnej vrstvy

Sieťový tok môže obsahovať aj menej alebo viac informácií a to podľa použitého nastavenia, alebo verzie použitého protokolu. Jednou z výhod sieťového toku je, že neobsahuje prenášané dáta, vďaka čomu má niekoľkonásobne menšiu veľkosť ako postupnosť paketov, z ktorých je vytvorený. Sieťový tok teda agreguje niekoľko paketov do jediného záznamu a neuchováva prenášané dáta. Pomocou sieťových tokov je teda možné zaznamenávať a uchovávať sieťovú

premávku s použitím minimálneho výpočtového výkonu a pamäťovej náročnosti. Prvý štandard sieťového toku *NetFlow v1 [8]* bol definovaný spoločnosťou Cisco už v roku 1996. Tento štandard však nebol nikdy používaný vo väčšej miere. Prvá verzia sieťového toku, ktorá sa začala používať bola verzia *NetFlow v5 [9]* a táto verzia je používaná dodnes. Alternatívou tohto formátu je formát *IPFIX [10]*, ktorý je nezávislý a bol definovaný organizáciou IETF. Tento formát je spätne kompatibilný so všetkými štandardmi *NetFlow* a umožňuje väčšiu flexibilitu nastavenia. V súčasnej dobe predstavuje jeden z najpoužívanejších nezávislých formátov.

2.2 Monitorovanie sieťovej premávky

Monitorovanie a analýza sieťovej premávky je jeden z kľúčových faktorov bezpečnosti a stability siete. Monitorovanie siete značne zvyšuje prehľad o dianí v sieti(takzvaný *situational awareness* [11]) a umožňuje tak operátorovi rýchlo a agilne reagovať na zmeny v sieti. Vďaka monitorovacím nástrojom je možné identifikovať bezpečnostné incidenty a dokonca im prechádzať. Nárast sieťovej premávky si však vyžaduje nové prístupy k monitorovaniu sietí. V minulosti bola priepustnosť sietí obmedzená na 10 až 100 Mbps, vďaka čomu bolo možné využívať jednoduché metódy monitorovania a administrácie. Príchod vysokorýchlostných sietí, ktoré dosahujú rýchlosti viac ako 10 Gbps si však vyžaduje nové sofistikovanejšie nástroje a metódy analýzy [12].

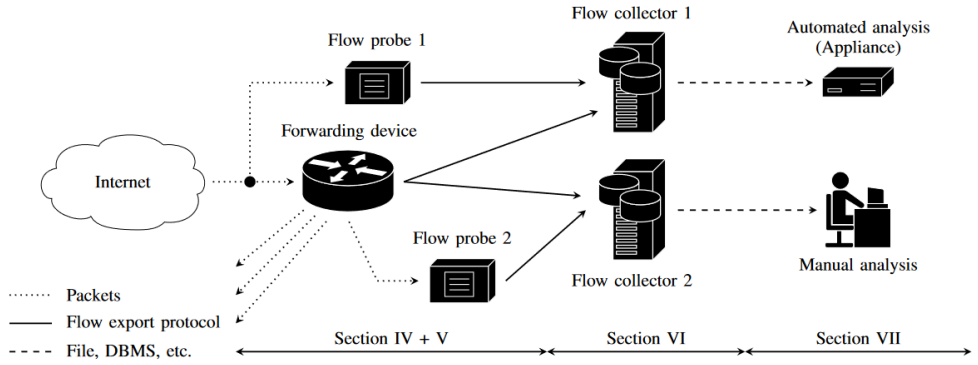
Metódy monitorovania sieťovej premávky delíme do dvoch skupín : aktívne a pasívne. Pasívne metódy zachytávajú a analyzujú premávku bez toho, aby ju upravovali. Naopak aktívne metódy sú schopné prenášané dáta modifikovať, alebo zahodiť. Tento prístup je veľmi výhodný v prípade, že monitorovací nástroj využíva automatické nástroje, ktoré sú schopné rozpoznať preťaženie siete, alebo útok na sieť. Pri použití aktívneho prístupu je totiž možné zamedziť zlyhaniu siete a škodlivú premávku upraviť, alebo zahodiť. Na tomto princípe pracujú napríklad DDoS detektory.

Jednou zo základných metód monitorovania sietí je zachytávanie paketov. Táto metóda poskytuje najviac informácií o prenášaných paketoch, včetne samotných prenášaných dát. Nevýhodou tejto metódy je slabá škálovateľnosť, ktorú spôsobuje potreba zachytávať, ukladať a analyzovať celú sieťovú premávku. V prípade vysokorýchlostných sietí s rýchlosťou väčšou ako 100 Gbps je tento spôsob vysoko neefektívny a vyžaduje silnú monitorovaciu infraštruktúru a veľké dátové úložisko [7].

Ďalšou metódou monitorovania sietí je metóda analýzy sieťových tokov. Táto metóda je veľmi účinná na vysokorýchlostných sieťach, pretože pakety agreguje a exportuje z nich len sieťové toky, ktoré majú menšie pamäťové nároky. Táto metóda je v súčasnej dobe veľmi rozšírená, pretože umožňuje efektívne monitorovanie vysokorýchlostných sietí. [7]

Podľa Hofstede et al. [7] sa bežná monitorovacia architektúra, využíva-

júca sieťové toky, delí na štyri časti. Prvá časť slúži na zachytávanie paketov. Zachytávanie paketov prebieha na sieťových sondách, ktoré sú rozmiestnené v sledovanej sieti. Druhá časť sa stará o konvertovanie paketov na sieťové toky. Konvertovanie je väčšinou zabezpečené priamo na sieťovej sonde, pomocou vstavaného exportéru sieťových tokov. Treťou časťou je kolektor, ktorý prijíma toky zo sieťových sond a ukladá ich. Posledným časť architektúry sa zaoberá analýzou sieťových tokov. Analýza môže prebiehať s pomocou operátora, alebo pomocou automatických nástrojov. Jedna z možných architektúr nasadenia monitorovacieho systému je zobrazená na obrázku 2.2.

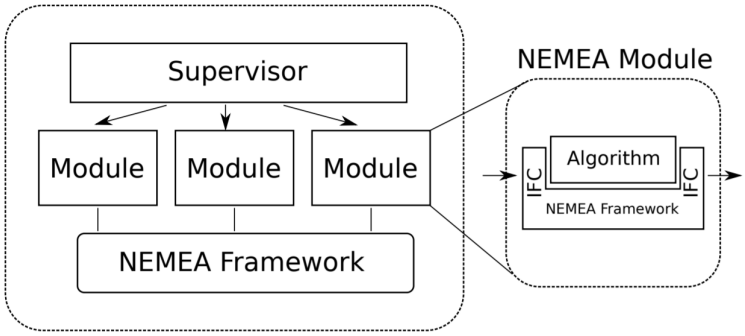


Obr. 2.2: Ukážka možnej inštalácie monitorovacieho systému [7]

2.2.1 NEMEA systém

Monitorovací systém NEMEA [13] sa zameriava na automatickú analýzu a detekciu hrozieb. Tento nástroj bol vyvinutý výskumnou skupinou Liberouter združenia CESNET v spolupráci s českými univerzitami. Systém bol navrhnutý tak, aby bol schopný monitorovať siete v reálnom čase a na základe získaných informácií identifikoval bezpečnostné incidenty. Celý systém sa skladá z nezávislých modulov, ktoré sú prepojené pomocou špeciálneho komunikačného rozhrania TRAP. Toto rozhranie umožňuje stabilnú a rýchlu komunikáciu medzi modulmi. Systém pre analýzu sieťovej premávky využíva sieťové toky, vďaka čomu je schopný efektívnejšie a rýchlejšie spracovávať prichádzajúcu komunikáciu. Systém je distribuovaný pod licenciou GPL v2, vďaka čomu je verejne dostupný a všetky zdrojové súbory sú dostupné na portáli GitHub [14].

Základná štruktúra systému NEMEA je zobrazená na obrázku 2.3. Systém



Obr. 2.3: Štruktúra NEMEA systému [13]

sa skladá z troch základných častí :

- NEMEA Framework
- NEMEA Supervisor
- moduly a detektory

Časť *NEMEA Framework* obsahuje všetky knižnice potrebné pre chod systému a jeho modulov. Medzi najdôležitejšie knižnice patrí knižnica TRAP, Unirec a Commom. Časť *NEMEA Supervisor* sa stará o riadenie a kontrolu spustených modulov v reálnom čase. *Supervisor* obsahuje zdieľané úložisko záznamov a výstupov z modulov, zdieľané úložisko konfiguračných súborov jednotlivých modulov a taktiež užívateľské rozhranie, v ktorom je možné sledovať, v akom stave sa nachádzajú jednotlivé moduly. Poslednou časťou systému sú moduly a detektory. Detektory slúžia k detekcií škodlivej premávky a bezpečnostných hrozieb. Systém aktuálne poskytuje niekoľko detektorov DoS, DNS tunelov a skenovania portov. Moduly systému NEMEA slúžia zvyčajne na spracovanie a agregáciu dát.

Všetky časti systému sú prepojené pomocou rozhrania TRAP, ktoré je implementované v knižnici libtrap. Táto knižnica je jednou z najdôležitejších knižníc v systéme, pretože zabezpečuje vysokorýchlostné prepojenie všetkých súčastí systému. Každá časť systému môže obsahovať ľubovolný počet vstupných a výstupných rozhraní, ktoré sa nazývajú IFC. Maximálna veľkosť jednej správy je 65 kB. Rozhranie TRAP poskytuje štyri základné typy IFC :

- UNIX rozhranie
- TCP rozhranie
- súbor
- čierna diera (zahadzuje všetky správy)

Všetky tieto typy sú si ekvivalentné a pri vývoji modulov nie je nutné ich

rozlišovať, čo umožňuje vysokú abstrakciu a užívateľ tak môže pracovať s rozhraním nezávisle na tom, či ide o súbor, alebo UNIX rozhranie.

Rozhranie TRAP umožňuje prenášať tri dátové formáty a to neštrukturované dáta, JSON a Unirec. Formát Unirec [15] je binárny formát, vyvinutý pre potreby systému NEMEA, ktorý sa vyznačuje veľkou flexibilitou a bol navrhnutý špeciálne na prenos sieťových tokov. Formát umožňuje definovať premenné variabilnej dĺžky a štruktúry za behu. Štruktúra dát je prenášaná spolu so záznamom a nazýva sa vzor. Vzor neobsahuje žiadne dáta, ale len informácie o štruktúre dát, a preto je veľmi malý a ľahko prenositeľný.

Systém NEMEA umožňuje využitie viacerých sieťových sond. Jedinou požiadavkou na sondy je schopnosť exportovať sieťové toky v niektorom z podporovaných formátov kolektoru. Systém k spracovaniu tokov využíva kolektor IPFIXcol [16], alebo jeho novšiu verziu IPFIXcol2 [17]. Tento kolektor je schopný spracovať toky vo formáte *IPFIX*, *NetFlow v5* až *NetFlow v9* a dokáže exportovať toky v rôznych formátoch ako napríklad Unirec, JSON alebo FDS. Pre účely vývoja alebo monitorovania domácej siete je možné využiť aj nástroj ipfixprobe [18], ktorý slúži ako sieťová sonda aj kolektor a dokáže zachytávať pakety priamo na sieťovom rozhraní zariadenia a exportovať sieťové toky vo formáte Unirec.

Systém spočiatku podporoval len moduly a detektory v programovacom jazyku C. V dobe písania tejto práce už obsahuje rozšírenie pytrap [19], ktoré umožňuje implementáciu modulov v jazyku Python.

2.3 Klasifikačné nástroje

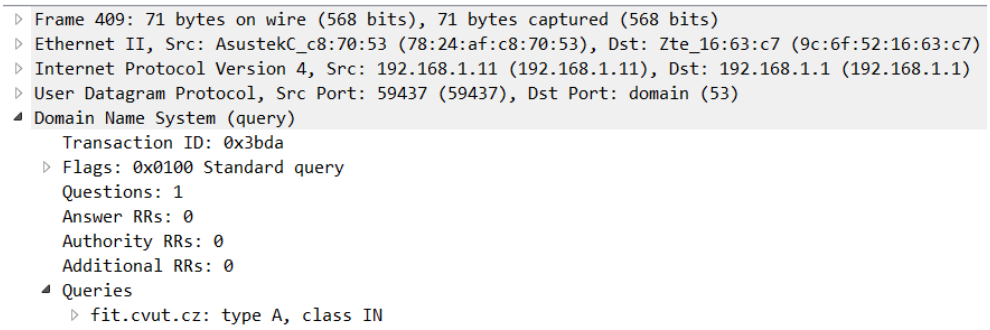
V tejto sekcií sa budem venovať existujúcim klasifikačným metódam, ktoré sa zameriavajú na klasifikáciu sieťových protokolov.

2.3.1 Wireshark

Jeden z najpoužívanejších nástrojov pre monitorovanie a analýzu sieťovej premávky je nástroj Wireshark [20]. Tento nástroj je distribuovaný pod licenciou GPL v2 a je teda verejne dostupný. Nástroj pracuje na úrovni paketov a k analýze využíva všetky obsiahnuté informácie v pakete a poskytuje tak užívateľovi všetky dostupné informácie o prenášaných paketoch, včetne samotných prenášaných dát. Nástroj obsahuje množstvo nástrojov a rozšírení na analýzu dát a obsahuje aj grafické prostredie.

Wireshark dokáže analyzovať zachytené dáta, ale umožňuje taktiež analýzu dát v reálnom čase. K tomu využíva knižnicu pcap, ktorá je schopná zachytávať pakety na vybranom sieťovom rozhraní. Pomocou nástroja Wireshark je možné vďaka dostupným nástrojom analyzovať obsah paketu. Nástroj je schopný identifikovať a načítať hlavičky jednotlivých protokolov. Vďaka tomu je možné pomocou tohto nástroja klasifikovať všetky použité protokoly

s vysokou presnosťou. Na obrázku 2.4 je zobrazená hĺbková analýza paketu, ktorý využíva DNS protokol.



Obr. 2.4: Wireshark hĺbková analýza DNS paketu [21]

Nástroj má aj distribúciu, ktorá nepodporuje grafické rozhranie čo umožňuje jeho nasadenie aj na serveroch, ktoré nepodporujú grafické rozhranie. Táto distribúcia je zároveň vhodnejšia na nasadenia v prípadoch, kedy sa nepredpokladá užívateľská obsluha a nástroj sa používa na automatické spracovanie sieťovej premávky. Príkladom využitia tohto nástroja je záverečná práca [22], v ktorej bol nástroj použitý na detekciu útokov na sieť.

2.3.2 libprotoident

Knižnica libprotoident [23] sa zameriava na klasifikáciu protokolov aplikačnej vrstvy. Bola vyvinutá výskumnou skupinou WAND na univerzite Waikato a ide o open-source knižnicu, ktorá je distribuovaná pod licenciou LGPL v3. Knižnica využíva na klasifikáciu hĺbkovú analýzu paketov v obmedzenej podobe, v ktorej využíva len niekoľko prvých bajtov paketu.

Možnosti tejto knižnice som skúmal vo svojej bakalárskej práci [2], v ktorej knižnica dosahovala vysokú presnosť, ale veľmi nízku úspešnosť (pojmy definované v sekcií 2.4.2). Nástroj nepodporuje grafické rozhranie a ukážka výstupu je zobrazená na obrázku 2.5.

```
protident label, source IP, dest IP, src port, dest port, transport protocol, start timestamp,
end timestamp, src bytes, dst bytes, first four bytes of payload(hex)

DNS_TCP 192.58.128.30 193.87.0.34 53 55695 6 1584016481.075 1584016481.081 1171 43 0491545c
Unknown_TCP 192.203.230.10 158.197.8.8 53 37082 6 1584016502.141 1584016546.400 1039 51 040d19a8
DNS_TCP 192.58.128.30 193.87.0.34 53 41585 6 1584016481.075 1584016481.081 1171 43 04916e1a
DNS_TCP 192.58.128.30 193.87.0.34 53 41845 6 1584016481.075 1584016481.081 1099 30 0449f7c9
```

Obr. 2.5: Ukážka výstupu knižnice libprotoident

2.3.3 NEMEA klasifikátor

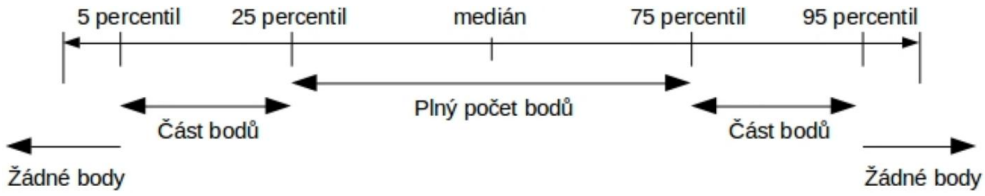
V rámci mojej bakalárskej práce som vytvoril klasifikačný modul, ktorý klasifikuje sieťovú premávku na základe sieťových tokov. Modul podporuje základné jednosmerné sieťové toky, ktoré prijíma prostredníctvom vstupného rozhrania IFC vo formáte Unirec. Výstupom klasifikačného modulu je prijatý sieťový tok, ku ktorému je pridaná charakteristika, obsahujúca kategóriu sieťovej premávky. Výstup je sprostredkovaný pomocou výstupného rozhrania IFC vo formáte Unirec. Zoznam klasifikačných kategórií je zobrazený v nasledujúcom zozname [2] :

- Web
  - http
  - https
- Mail
  - imap
  - pop3
  - smtp
- Vzdialený prístup
  - ssh
  - telnet
  - ftp
  - rdp
- Sieťová obsluha
  - ntp
  - dns
  - dhcp
  - rip

Modul ku klasifikácií využíva intervaly spoľahlivosti. Pre každú charakteristiku sieťového toku je vytvorený jeden interval spoľahlivosti, ktorý reprezentuje bližšie a širšie okolie mediánu pozorovaných hodnôt. Každá klasifikačná kategória obsahuje niekoľko intervalov spoľahlivosti. Pri klasifikácií modul pre každú charakteristiku overuje, či patrí do jedného z týchto okolí. Po vyhodnotení všetkých charakteristík toku je percentuálna pravdepodobnosť príslušnosti sieťového toku k testovanej klasifikačnej kategórií. Ukážka intervalu spoľahlivosti je zobrazená na obrázku 2.6.

Na vyhodnotenie úspešnosti a trénovanie klasifikačného modulu bola použitá dátová sada *UniFlow*(viac v sekcií 3.4). Klasifikačný modul vo vyhodnotení dosiahol priemernú úspešnosť 92,82 %, zatiaľ čo knižnica libprotoident na tejto dátovej sade dosiahla úspešnosť 73,68 % [2].

Tento klasifikačný modul je použitý ako jeden zo vstupných modulov v projekte ADiCT [24].



Obr. 2.6: Intervaly spoľahlivosti klasifikačného modulu z bakalárskej práce [2]

2.4 Klasifikačné metódy strojového učenia

Strojové učenie je podoblasť umelej inteligencie, ktorá sa zaoberá metódami a algoritmami, ktoré umožňujú programu učiť sa a následne reagovať na vstupné hodnoty bez toho, aby bol na ne explicitne naprogramovaný. Tieto algoritmy a metódy využívajú matematickú štatistiku, štatistickú analýzu a hĺbkovú analýzu dát (data mining). [25]

Metódy strojového učenia umožňujú predpovedanie budúcich javov, alebo budúceho správania s pomocou samoučiacich algoritmov. Táto schopnosť je nutná hlavne v prípadoch, kedy je nie je možné program explicitne naprogramovať na vyriešenie daného problému, a nie je možné vopred určiť presnú množinu vstupov programu. V týchto prípadoch je použitie samoučiacich algoritmov, ktoré sú schopné identifikovať dôležité charakteristiky bez ľudskej pomoci veľmi výhodné. Strojové učenie je taktiež veľmi užitočné v prípadoch, kedy je potrebné nájsť a extrahovať vzťahy a závislosti v dátovej sade, ktoré nie sú na prvý pohľad, bez dôkladnej analýzy viditeľné. [26]

Algoritmy strojového učenia dokážu vykonávať tri základne typy úloh : klasifikáciu, regresiu, zhlukovanie [26]. Klasifikácia môže byť binárna, alebo viactriedna a jej úlohou je rozoznať a určiť triedu objektu. Výstupom klasifikácie sú diskrétne hodnoty. Príkladom môže byť klasifikácia emailu na spam, alebo ham. Regresia sa snaží na základe jedného atribútu odvodiť, alebo odhadnúť hodnotu iného atribútu. Príkladom môže byť zistenie hmotnosti človeka na základe jeho výšky. Posledný typ úlohy je zhlukovanie, ktoré využíva učenie bez učiteľa a snaží sa vstupné dáta na základe ich vlastností zlúčiť do niekoľkých kategórií.

Pri vytváraní modelu strojového učenia sa využívajú tri metódy učenia a

to :

- **Učenie bez učiteľa** : sa využíva v prípadoch, kedy je potrebné hľadať vo vstupných dátach určité vzory, alebo charakteristiky. Metóda spracováva neoznačené údaje, alebo údaje neznámej štruktúry. Na získanie informácií z takýchto dát sa využíva metóda klastrovania, alebo metóda redukcie dimenzionality.
- **Učenie s učiteľom** : sa využíva v prípadoch, kedy je potrebné predpovedať výstupné hodnoty na základe vstupných hodnôt. Metóda k tréningu používa anotované dáta na základe ktorých sa snaží vytvoriť model. Pokiaľ sú výstupné hodnoty diskrétne je to úloha klasifikácie, v prípade že sú výstupné hodnoty spojité ide o úlohu regresie.
- **Učenie s posilňovaním** : typ učenia, v ktorom metóda dostáva v každom kroku spätnú väzbu v podobe odmeny. Výška odmeny je stanovená na základe vzdialenosti vykonanej akcie od požadovanej akcie. Metóda na základe spätnej väzby vytvára hodnotovú funkciu, s pomocou ktorej sa následne snaží odhadnúť a určiť akciu, ktorá povedie v najvyššej odmene.

2.4.1 Postup klasifikácie

Na vytvorenie klasifikátora strojového učenia je nutné splniť niekoľko pre prerekvizít. Prvou je správne definovanie problému, ktorý má metóda riešiť. Ďalej je potrebné nazbierať vhodné dáta. Tieto dáta je nutné následne analyzovať a prípadne upraviť a extrahovať relevantné charakteristiky. Úpravou dátovej sady sa rozumie nahradenie chýbajúcich, alebo nevyhovujúcich hodnôt, normalizácia vstupov, alebo nelineárne transformácie. Ďalším krokom je vytvorenie, natrénovanie a otestovanie klasifikátoru pomocou vytvorených dátových sád. Posledným krokom je použitie vytvoreného klasifikátoru na klasifikáciu neznámych dát. Tento postup je možné zhrnúť do nasledujúcich krokov:

1. Definícia problému
2. Vytvorenie a úprava dátovej sady
3. Trénovanie a testovanie klasifikátora
4. Nasadenie klasifikátora

2.4.2 Prehľad klasifikačných metrík

Interpretácia a vyhodnotenie výsledkov klasifikátorov je veľmi dôležitým krokom. V rámci metód strojového učenia boli zavedené unifikované pojmy a metriky naprieč všetkými metódami.

Rozoznávame štyri kategórie výsledkov. Každá z týchto kategórií reprezentuje výsledky z pohľadu jednej klasifikačnej triedy. Pozitívna trieda je trieda, ktorej výsledky skúmame a všetky ostatné triedy sú negatívne. Jednotlivé kategórie s vysvetlením sú zobrazené v nasledujúcom zozname :

- **True positive (TP)** : Počet správne klasifikovaných pozitívnych záznamov
- **False positive (FP)** : Počet nesprávne klasifikovaných záznamov do pozitívnej triedy
- **True negative (TN)** : Počet správne klasifikovaných negatívnych záznamov
- **False negative (FN)** : Počet nesprávne klasifikovaných záznamov do negatívnej triedy

Tieto kategórie je možné taktiež reprezentovať pomocou matice zámeny, zobrazenej v tabuľke 2.2, ktorá sa používa na prezentáciu výsledkov klasifikácie.

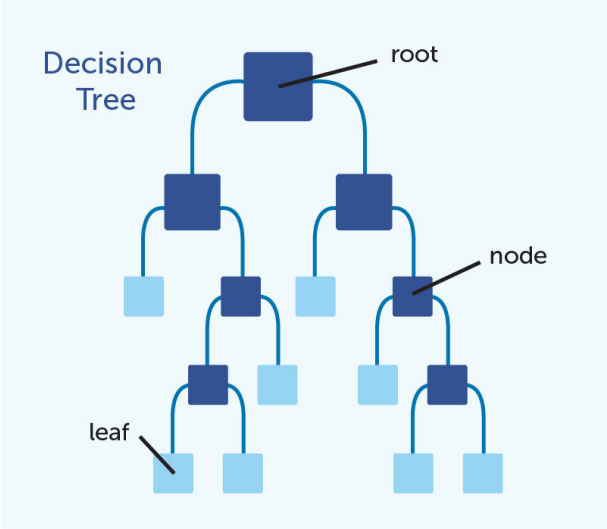
Tabuľka 2.2: Ukážková matica zámeny, stĺpce matice reprezentujú predikované hodnoty a riadky pravdivé hodnoty.

	Pozitívne	Negatívne
Pozitívne	TP	FN
Negatívne	FP	TN

Na vyhodnotenie výsledkov experimentov boli definované metriky. Existuje niekoľko vyhodnocovacích metrík [27]. Medzi najbežnejšie používané metriky patria:

- **Úspešnosť (ACC)** : $\frac{FP + FN}{FP + FN + TP + TN}$
- **Presnosť (PRE)** : $\frac{TP}{TP + FP}$
- ***Recall* (REC)** : $\frac{TP}{FN + TP}$
- **f1-skóre (F1)** : $\frac{PRE * REC}{PRE + REC}$

Pomocou týchto metrík je možné presne reprezentovať výsledky experimentov a sledovať rozdielne správanie klasifikátorov.



Obr. 2.7: Zjednodušená štruktúra rozhodovacieho stromu [28]

2.4.3 Metóda rozhodovacích stromov

Rozhodovací strom patrí k najjednoduchším algoritmom strojového učenia. Metóda využíva sadu hierarchicky usporiadaných rozhodovacích pravidiel, ktoré tvoria štruktúru stromu. Štruktúra je tvorená uzlami a listami. List reprezentuje výstupnú hodnotu a je hierarchicky najnižšie. Uzly obsahujú rozhodovaciu logiku, ktorá na základe vstupnej hodnoty rozhodne o prechode do ďalšieho uzlu alebo listu. Zjednodušená štruktúra rozhodovacieho stromu je zobrazená na obrázku 2.8.

Proces trénovania spočíva v postupnom vetvení stromu od prvého uzlu, teda koreňa. V každom uzle sa identifikuje rozdelenie množiny, ktoré najlepšie separuje dáta podľa zvolených kritérií. Následne je vytvorené pravidlo, ktoré rozdelí množinu na podmnožiny. Vo vzniknutých podmnožinách sa následne opakuje postup hľadania najlepšieho rozdelenia a vytvorenie pravidla. Týmto spôsobom sa vytvárajú uzly stromu až do doby, kedy je dosiahnutá minimálna povolená veľkosť množiny (väčšinou 2). Najpoužívanejšími kritériami rozdelenia množín sú:

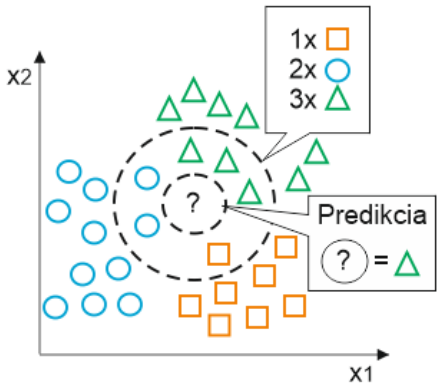
- **Entropia** : meria podiel jednotlivých klasifikačných tried v podmnožinách. Klasifikátor sa snaží rozdeliť množinu na podmnožiny tak, aby mali čo najnižšiu entropiu. [29]
- **Gini index** : meria ako často by sa klasifikátor zmýlil, ak by sme podmnožinu klasifikovali ako jednu triedu. Klasifikátor sa snaží množinu rozdeliť tak, aby mali obe podmnožiny čo najnižší index. [29]

Proces klasifikácie spočíva vo vyhodnotí vstupných charakteristík pomocou

vytvorených pravidiel. Vyhodnotenie prebieha hierarchicky od uzlu smerom k listom. Jednou z výhod tejto metódy je prehľadnosť a ľahká interpretovateľnosť, vďaka ktorej je možné pochopiť výsledky a identifikovať kľúčové charakteristiky vstupných dát. Rozhodovacie stromy taktiež dokážu rozpoznať dôležité charakteristiky, čo je veľmi prínosné v prípade, že metóda pracuje s dátovou sadou, ktorá obsahuje veľké množstvo charakteristík.

2.4.4 Metóda k-najbližších susedov

Metóda ku klasifikácií využíva využíva meranie vzdialenosti medzi záznamami. Metóda si pri trénovaní uloží charakteristiky všetkých záznamov a pri klasifikácií meria vzdialenosť charakteristík nového toku s uloženými hodnotami. Týmto procesom merania následne určí k najbližších susedov a na základe triedy týchto susedov určí triedu nového toku. Na meranie vzdialenosti sa môže použiť ľubovolná miera vzdialenosti, napríklad Manhattanská alebo Euklidovská vzdialenosť.



Obr. 2.8: Ukážka rozhodovacieho algoritmu metódy k-najbližších susedov [30]

Túto metódu je možné použiť na učenie s učiteľom, alebo bez učiteľa. Jediným rozdielom je, že v prípade učenia s učiteľom je uloženým záznamom pridelený názov.

2.4.5 Metóda Extra-tree

Táto metóda využíva ku klasifikácií rozhodovacie stromy. Základná metóda rozhodovacích stromov využíva jeden strom. Z tejto metódy bola odvodená metóda náhodných lesov, ktorá namiesto jedného stromu využíva niekoľko stromov. Metóda Extra-tree je variáciou tejto metódy a je často označovaná ako metóda extrémne náhodných lesov a líši sa v spôsobe konštrukcie jednotlivých rozhodovacích stromov.

Namiesto použitia jedného rozhodovacieho stromu metóda vytvorí niekoľko stromov, ktoré nazývame les. Každý strom je následne natrénovaný po-

mocou náhodnej podmnožiny charakteristík dátovej sady. Veľkosť každej podmnožiny je definovaná ako odmocnina z celkového počtu charakteristík [31]. Výsledkom tohto procesu je niekoľko stromov, pričom každý z týchto stromov využíva náhodnú množinu charakteristík.

Pri klasifikácií je záznam skúmaný všetkými stromami. Výsledky jednotlivých stromov sú následne vyhodnotené, a to tak, že metóda hľadá triedu ktorá je vo výsledkoch najviac zastúpená. [31]

Kapitola 3

Dátové sady

Dátové sady sú jedným z najdôležitejších elementov, ktoré zásadne ovplyvňujú výsledky a vierohodnosť experimentov strojového učenia [32]. Je preto nutné dbať na použitie vhodných dátových sád a vlastnosti dátových sád vždy preveriť. Dátové sady musia obsahovať istú mieru neurčitosti a variácie, v opačnom prípade môže nastať bez ohľadu na klasifikátor *overfitting* [33]. *Overfitting* je jav, ktorý nastáva v prípade, že klasifikátor dosahuje vysokú úspešnosť klasifikácie na jednej dátovej sade ale jeho úspešnosť sa dramaticky zhorší pri testovaní na inej dátovej sade. Príkladom dátovej sady, ktorá spôsobuje tento jav, je dátová sada zachytená len na jednom zariadení, napríklad osobnom počítači. Úspešnosť klasifikácie na tejto dátovej sade môže byť vysoká, ale v prípade použitia dátovej sady z iného zariadenia, ktoré má napríklad iný operačný systém alebo sa jedná o rozdielny typ zariadenia, výsledky klasifikácie sa dramaticky zhoršia, pretože klasifikátor nevie ako správne reagovať na odlišné dáta a správanie siete alebo užívateľa. V rámci tejto práce sa snažím tento jav, čo najviac potlačiť, pretože cieľom je vyvinúť univerzálny klasifikátor, ktorý nie je závislý na zariadení alebo sieti, v ktorej sa nachádza. Opakom *overfitting* je *underfitting*, ktorý nastáva vtedy, keď metóda nie je schopná identifikovať dostatočné množstvo rozdielov medzi klasifikačnými kategóriami. V tomto prípade metóda dosahuje nízku presnosť klasifikácie na tréningovej aj testovacej dátovej sade.

V rámci bakalárskej práce [2] som vytvoril dátovú sadu, ktorá obsahovala základné sieťové toky a obsahovala celkom 9 protokolov. V rámci tejto práce som využil túto dátovú sadu na vyhodnotenie a porovnanie výsledkov klasifikácie tohto experimentu s výsledkami dosiahnutými v bakalárskej práci. Táto dátová sada však neobsahuje rozšírené údaje o sieťových tokoch, ktoré sú pre ďalšie experimenty potrebné. V rámci rešerše som nenašiel žiadne vhodné dátové sady a preto bolo nutné vytvoriť nové.

Po konzultácií s vedúcim práce som sa rozhodol vytvoriť dátovú sadu, ktorá bude obsahovať nasledujúcich 9 protokolov :

- http
- https
- imap
- pop3
- smtp
- ssh
- telnet
- dns
- ntp

Tieto protokoly sú zhodné s protokolmi použitými v rámci mojej bakalárskej práce, čo mi umožní porovnať výsledky týchto prác. Ďalším dôvodom voľby týchto protokolov je fakt, že sa jedná o najčastejšie sa vyskytujúce protokoly na sieti CESNET2 a tieto protokoly pokrývajú veľkú časť premávky, ktorú chcem klasifikovať. Pri voľbe počtu sieťových tokov som sa na základe výsledkov a poznatkov z bakalárskej práce a po konzultácií s vedúcim práce, rozhodol vytvoriť dátovú sadu, ktorá bude obsahovať minimálne 30-tisíc tokov z každého protokolu.

Zachytávanie, konvertovanie a anotácia dátovej sady prebiehali v kontrolovanom prostredí združenia CESNET s pomocou vedúceho práce.

3.1 Zachytávanie dátových sad

Na základe prieskumu a výsledkov minulých experimentov bolo zrejmé, že dátové sady nemožno zachytávať len na jednom zariadení alebo na malej skupine zariadení. Tento postup totiž produkuje značné množstvo monotónnej premávky a nedostatok premávky určitých protokolov, ktoré koncové zariadenia bežne nepoužívajú. Taktiež dáta z takto zachytených sietí nie sú dostatočne rôznorodé, keďže samotné zariadenia a aj sieťové prvky v bezprostrednom okolí majú svoje špecifiká a limity.

Vďaka ústretovosti a ochote združenia CESNET som mohol využiť na zachytenie dátových sád sieťové sondy, s pomocou ktorých som získal dáta nie len z lokálnych sietí, ale aj zo sieťových liniek prenášajúcich veľké množstvo dát od rôznych subjektov, čo značne zvýši rôznorodosť a univerzálnosť dátových sád. Na zachytávanie dát sme taktiež použili viaceré sieťové sondy z rôznych sieťových liniek združenia aby sme ešte viac zvýšili rôznorodosť a čo najviac sa priblížili reálnej sieťovej premávke.

Dáta sme zachytávali v bežne používanom formáte pcap s použitím nástroja, určeného na zachytávanie vysokorýchlostnej premávky v združení CESNET. Kvôli pamäťovej náročnosti zachytávania sme aplikovali dostupné filtre nástroja a zachytávali sme len premávku na požadovaných portoch. Zachytené

pakety vo formáte pcap boli konvertované na rozšírené sieťové toky pomocou nástroja *ipfixprobe [18]*.

Celkom bolo zachytených 132 GB dát vo formáte pcap. Po prekonvertovaní paketov vzniklo celkom 680-tisíc sieťových tokov.

Kvôli zachovaniu anonymity boli zo zachytených dát odstránené MAC adresy a všetky IP adresy boli anonymizované pomocou nástroja anonymizer [34].

3.2 Anotácia a úpravy dátovej sady

Anotácia dát je jeden z kľúčových faktorov, ktorý ovplyvňuje úspešnosť klasifikácie. Nesprávne anotované dáta môžu zapríčiniť nesprávne trénovanie klasifikátorov a tak zásadne ovplyvniť výsledky experimentu. Zachytené dáta by mali obsahovať prevažne požadovanú sieťovú premávku. Avšak filtrovanie prebiehalo len s pomocou čísla sieťového portu a preto nie je možné sa plne spoľahnúť na ich pravdivosť a je nutné ubezpečiť sa, že zachytené dáta naozaj obsahujú požadované protokoly.

Jedným z najpresnejších a najdostupnejších riešení bolo využitie hĺbkovej analýzy paketov a overenie obsahu paketov. Na tento účel som využil nástroj TShark. Nástroj TShark identifikuje protokoly na základe hlavičiek paketov a na základe špecifických vlastností určitých protokolov.

Nástroj TShark však pracuje s paketmi a nebolo teda možné anotovať sieťové toky priamo. Preto sme nástroj TShark použili na vygenerovanie zoznamu IP adries, ktoré spolu komunikovali požadovaným protokolom. Každý záznam v zozname obsahoval zdrojovú a cieľovú IP adresu, zdrojový a cieľový port a použitý protokol. Tento zoznam už bolo možné spárovať so sieťovými tokmi. Na tento účel som vytvoril program, ktorý prefiltruje sieťové toky na základe zoznamu z nástroja TShark. Vytvorený program starostlivo kontroluje krajné prípady a v prípade neistoty sieťový tok zahodí. Príkladom krajného prípadu môže byť komunikácia dvoch IP adries na rovnakých portoch viacerými sieťovými protokolmi. Táto situácia môže teoreticky vzniknúť na sieti alebo chybou anotácie nástrojom TShark.

Výstupom programu sú anotované dátové sady obsahujúce rozšírené obojsmerné sieťové toky. Tieto dátové sady neobsahujú žiadne prenášané dáta z paketov. Z celkového počtu 680-tisíc tokov bolo úspešne anotovaných približne 470-tisíc tokov.

Pri kontrole dátových sád som zistil, že niektoré protokoly majú oveľa väčšie zastúpenie. To by mohlo viesť k skresleniu výsledkov, kedy by úspešnosť menej zastúpených protokolov nemala v celkovom hodnotení rovnaký podiel ako úspešnosť viac zastúpených protokolov. Samozrejme túto skutočnosť by bolo možné odhaliť preskúmaním iných klasifikačných metrík. Pre lepšiu transparentnosť som sa rozhodol nastaviť jednotný počet sieťových tokov pre

každý protokol, vďaka čomu je rovnomerné rozdelenie jednotlivých protokolov vo výsledku zaručené. Výsledná dátová sada teda obsahuje 360-tisíc tokov.

V rámci finálnych úprav som do dátovej sady pridal charakteristiku *BPP*, ktorá obsahuje priemernú veľkosť jedného paketu. Túto charakteristiku som vypočítal vydelením celkového počtu prenesených bajtov počtom paketov sieťového toku. Táto charakteristika nevyužíva PSTATS, ale základné charakteristiky toku.

3.3 Analýza pomocou nástroja *FET*

Vlastnosti rozšírených dátových sád, som preskúmal pomocou knižnice *FET*, vytvorenej v rámci záverečnej práce [35]. Knižnica *FET* sa zameriava na analýzu charakteristík PSTATS. K extrakcií informácií z PSTATS využíva knižnica štatistické vlastnosti charakteristík ako je priemer, stredná hodnota, maximum alebo minimum. Knižnica celkom extrahuje 33 charakteristík. Ukážka extrahovaných charakteristík je zobrazená v nasledujúcom zozname a kompletný zoznam všetkých extrahovaných charakteristík je dostupný v dokumentácií nástroja CICFlowMeter[36].

- **Flow duration** - Trvanie toku v nanosekundách
- **Fwd Packet Length Min** - Minimálna veľkosť paketu smerujúceho dopredu
- **Fwd Packet Length Max** - Maximálna veľkosť paketu smerujúceho dopredu
- **Fwd Packet Length Mean** - Stredná hodnota veľkosti paketu smerujúceho dopredu
- **Fwd Packet Length Std** - Smerodajná odchýlka veľkosti paketu smerujúceho dopredu

Vlastnosti týchto atribútov sú následne skúmané pomocou rôznych knižníc jazyka Python. Výsledok analýzy je v súbore *FET.ipynb*. Nástroj vykresľuje väčšinu výsledkov do grafu a preto som kvôli prehľadnosti a časovej náročnosti z dátových sád vybral pre každý protokol 500 sieťových tokov, ktorých vlastnosti som skúmal pomocou tohto nástroja. Nástroj som spustil niekoľko krát na náhodne vybraných tokoch a nepozoroval som žiadne významné odlišnosti vo výsledkoch.

Prvým výstupom nástroja je korelačná matica, ktorá zobrazuje koreláciu medzi jednotlivými extrahovanými charakteristikami. Korelačná matica je zobrazená na obrázku 3.1. V matici môžeme vidieť silnú koreláciu medzi charakteristikami *pkt_iat*, ktoré reprezentujú rozdiel času medzi dvoma po

sebe idúcimi paketmi. Knižnica sa ďalej zameriavala na reprezentáciu rozdielov medzi charakteristikami jednotlivých protokolov a využila niekoľko metód identifikácie najdôležitejšej charakteristiky. Ukážka výstupu funkcie *feature_importances* je zobrazená na obrázku 3.2. Výsledky rozdielnych použitých metód sa značne líšili, ale bol som schopný identifikovať niekoľko charakteristík, ktoré dosahovali nadpriemerné výsledky :

- **ack_ratio** - podiel TCP príznakov ACK
- **syn_ratio** - podiel TCP príznakov SYN
- **psh_ratio** - podiel TCP príznakov PSH
- **lengths_max** - maximálna veľkosť paketu

Výsledky analýzy dátovej sady mi poskytli prehľad o extrahovaných charakteristikách a ich vlastnostiach.

3.4 Prehľad získaných dátových sád

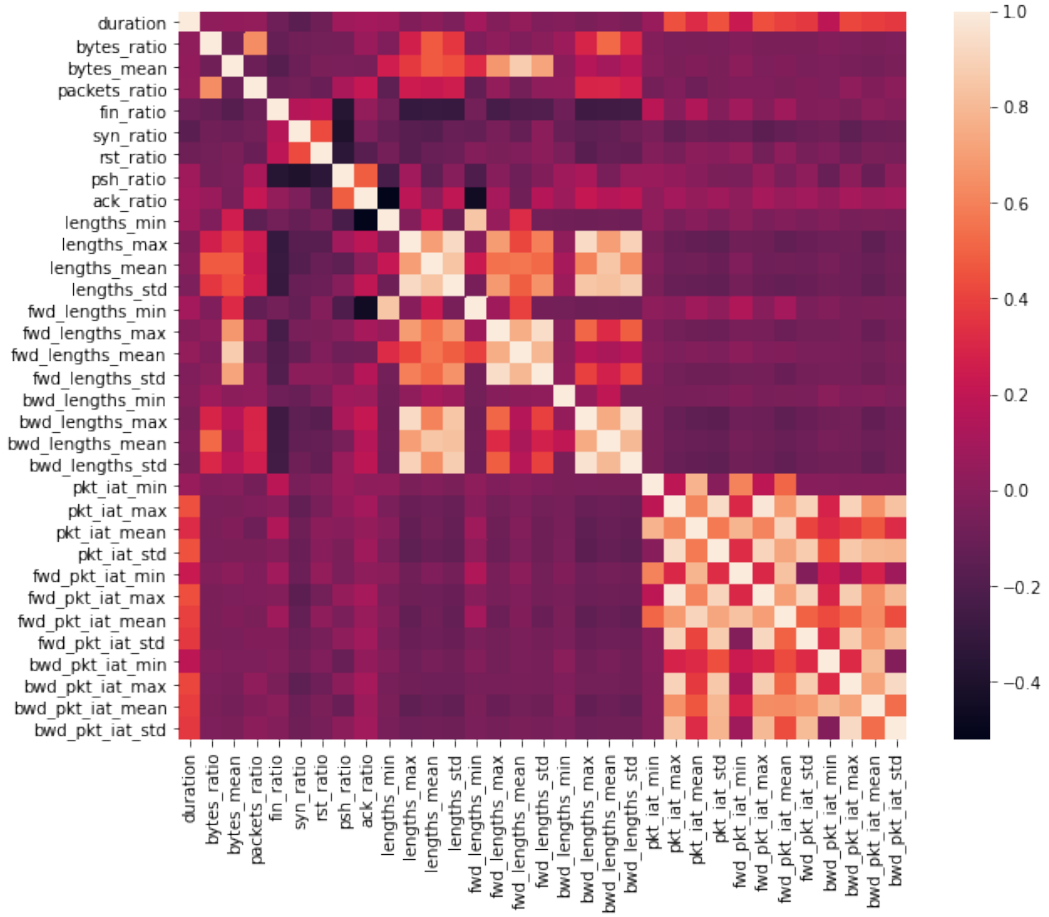
V rámci práce s dátovými sadami sa mi mi podarilo získať a vytvoriť tri dátové sady. Všetky dátové sady obsahujú rovnaké protokoly, a pokiaľ je to možné, tak aj rovnaké množstvo sieťových tokov. Prehľad vytvorených dátových sád a ich vlastností je zobrazený v tabuľke 3.1

Tabuľka 3.1: Prehľad vlastností jednotlivých dátových sád.

	UniFlow	BiFlow	Rozšírené BiFlow
Počet tokov	90 000	360 000	360 000
Počet protokolov	9	9	9
Min. počet tokov protokolu	10 000	40 000	40 000
Typ sieťových tokov	jednosmerné	obojsmerné	obojsmerné
Použité rozšírenie	žiadne	žiadne	PSTATS

Prvá dátová sada, nazvaná *UniFlow*, obsahuje jednosmerné sieťové toky. Túto sadu som vytvoril a použil v rámci svojej bakalárskej práce. Táto dátová sada má však nerovnomerne zastúpené rozdelenie protokolov ktoré, je zobrazené na obrázku 3.3. Pre zachovanie rovnomerného rozdelenia jednotlivých protokolov som dátovú sadu zmenšil tak, aby mal každý protokol rovnaké zastúpenie. Ukážka dátovej sady je zobrazená na obrázku 3.4.

Druhá dátová sada, nazvaná *BiFlow*, obsahuje obojsmerné sieťové toky. Táto dátová sada obsahuje v porovnaní s dátovou sadou *UniFlow* ďalšie dve charakteristiky a to *PACKETS_REV* a *BYTES_REV*. Táto dátová sada bola vytvorená odstránením rozšírenia PSTATS z dátovej sady vytvorenej v tejto práci. Ukážka tejto dátovej sady je zobrazená na obrázku 3.5.



Obr. 3.1: Korelačná matica vytvorená knižnicou *FET*. Matica zobrazuje korelácie medzi charakteristikami extrahovanými z PSTATS.

Posledná dátová sada, pomenovaná *Rozšírené BiFlow*, obsahuje obojsmerné sieťové toky rozšírené o PSTATS. Túto dátovú prácu som vytvoril v rámci tejto práce a jej vlastnosti sú uvedené v tejto kapitole. Ukážka charakteristík PSTATS je zobrazená na obrázku 3.6.

Všetky dátové sady obsahujú charakteristiku *TSHARK_label*, ktorá bola vytvorená pri procese anotácie a slúži ako referenčná hodnota pre vyhodnocovanie výsledkov.

```
[('bwd_lengths_max', 0.0710660687982738),
 ('bwd_lengths_std', 0.0680753974053012),
 ('lengths_max', 0.05962476253945628),
 ('syn_ratio', 0.056791399153639864),
 ('ack_ratio', 0.055565224477266764),
 ('psh_ratio', 0.05361573185371712),
 ('fwd_lengths_max', 0.048238721729710624),
 ('lengths_std', 0.04295244015566358),
 ('fwd_lengths_std', 0.04276574188561397),
 ('bwd_lengths_mean', 0.042083077543660974),
 ('lengths_mean', 0.035608021544543345),
 ('bytes_mean', 0.035348384189425444),
```

Obr. 3.2: Výstup funkcie *feature_importances* knižnice *FET*. Výstup zobrazuje najdôležitejšie charakteristiky extrahované z PSTATS.

Datová sada	Celkem
http	16 131
https	14 697
imap	21 725
pop3	20 121
smtp	33 020
ssh	33 281
telnet	39 471
dns	61 772
ntp	2 750 865

Obr. 3.3: Obsah dátovej sady použitej v mojej bakalárskej práci [2].

	uint64 BYTES	uint32 PACKETS	uint8 PROTOCOL	string TSHARK_LABEL	duration	BPP
0	1108	18	6	http	153915.0	61.555556
1	774	8	6	http	840.0	96.750000
2	33662	643	6	http	255.0	52.351477
3	877	5	6	http	78.0	175.400000
4	546	5	6	http	25.0	109.200000
...	...	...	...	...	...	...

Obr. 3.4: Ukážka obsahu dátovej sady *UniFlow*.

	uint64 BYTES	uint64 BYTES_REV	uint32 PACKETS	uint32 PACKETS_REV	uint8 PROTOCOL	string TSHARK_LABEL	duration	BPP
0	122	0	1	0	17	dns	0.0	122.0
1	137	190	1	1	17	dns	29.0	137.0
2	63	79	1	1	17	dns	6.0	63.0
3	82	143	1	1	17	dns	8.0	82.0
4	84	140	1	1	17	dns	1.0	84.0

Obr. 3.5: Ukážka obsahu dátovej sady *BiFlow*.

int8* PPI_PKT_DIRECTIONS	uint8* PPI_PKT_FLAGS	uint16* PPI_PKT_LENGTHS	time* PPI_PKT_TIMES
[1]	[0]	[122]	[2020-10-28T09:30:31.293]
[1|-1]	[0|0]	[137|190]	[2020-10-28T09:30:30.923|2020-10-28T09:30:30.952]
[1|-1]	[0|0]	[63|79]	[2020-10-28T09:30:31.456|2020-10-28T09:30:31.462]
[1|-1]	[0|0]	[82|143]	[2020-10-28T09:30:32.369|2020-10-28T09:30:32.377]
[1|-1]	[0|0]	[84|140]	[2020-10-28T09:30:20.639|2020-10-28T09:30:20.640]

Obr. 3.6: Ukážka obsahu charakteristík PSTATS. Každá zo zobrazených charakteristík PSTATS môže obsahovať jednu až tridsať charakteristík.

Kapitola 4

Experimenty

V tejto kapitole som popísal experimenty, ktoré som uskutočnil v rámci skúmania možností klasifikácie sieťových protokolov s pomocou strojového učenia.

4.1 Výskumné ciele a zavedené pojmy

Po vytvorení dátových sád a preskúmaní dostupných nástrojov a metód som po konzultácií s vedúcim práce definoval nasledujúce výskumné ciele :

1. Vytvorenie a vyhodnotenie klasifikátora využívajúceho dátovú sadu *UniFlow* a porovnanie úspešnosti s modulom z bakalárskej práce. (sekcia 4.4.1)

2. Vytvorenie a vyhodnotenie klasifikátora využívajúceho dátovú sadu *BiFlow*. (sekcia 4.4.2)

3. Vytvorenie a vyhodnotenie klasifikátora využívajúceho dátovú sadu *Rozšírené BiFlow*. (sekcia 4.4.3)

4. Porovnanie úspešnosti klasifikátorov využívajúcich *UniFlow*, *BiFlow*, *Rozšírené BiFlow*. (sekcia 4.4)

5. Porovnanie úspešnosti jednotlivých klasifikačných metód na rovnakej dátovej sade. (sekcia 4.5)

6. Preskúmanie možností metód hľadania najdôležitejších charakteristík a metód hľadania hyper parametrov. (sekcia 4.4.3 a 4.5.1)

Získané výsledky znazorňujú a dokazujú možnosti použitia metód strojového učenia na klasifikáciu sieťových tokov za rôznych podmienok. Získané poznatky mi umožnia navrhnúť a implementovať klasifikačný modul, ktorý preukázateľne využíva najvhodnejšie metódy.

Kvôli nejasnosti, ktorá je spôsobená prekladom cudzích pojmov a rozdielnou interpretáciou v literatúre a v dokumentácií som sa rozhodol definovať pojmy používané v tejto práci :

- **charakteristika** : doslovný preklad anglického slova *feature*. Tento pojem používam v kontexte strojového učenia a reprezentuje individuálnu merateľnú vlastnosť. Pri použití pojmu v kontexte dátových sád, tento pojem reprezentuje jeden stĺpec dátovej sady vo formáte CSV a v kontexte sieťových tokov pod charakteristikou rozumieme vlastnosť alebo atribút sieťového toku.
- **metóda strojového učenia** : metóda používaná v rámci strojového učenia na vytvorenie modelov a klasifikáciu. Príkladom metód je napríklad rozhodovací strom a k-najbližších susedov .
- **klasifikátor** : inštancia metódy strojového učenia, ktorá obsahuje natrénovaný model a dokáže klasifikovať dáta
- **DT** : skratka metódy rozhodovacích stromov
- **KNN** : skratka metódy k-najbližších susedov
- **hyper parametre** : parametre funkcií strojového učenia ktoré ovplyvňujú proces tréningu

4.2 Návrh experimentov

Experimenty vyplývajúce z výskumných cieľov je možné rozdeliť do dvoch hlavných častí :

1. Experimenty s rozdielne obsiahnutými dátovými sadami
2. Experimenty s metódami strojového učenia

V prvej časti experimentov budem skúmať zmeny v úspešnosti klasifikácie pri použití rozdielne obsiahnutých dátových sád. Pre tieto experimenty používam 3 dátové sady a to dátovú sadu *UniFlow*, *BiFlow* a *Rozšírené BiFlow*. Vlastnosti týchto dátových sád sú uvedené v sekcií 3.4. Na vyhodnotenie úspešnosti klasifikácie používam metódy strojového učenia. Pri výbere konkrétnych metód som sa rozhodol použiť dve metódy a to metódu rozhodovacích stromov a metódu k-najbližších susedov. Tieto metódy najlepšie vyhovujú potrebám experimentu a často sa využívajú na sieťovú klasifikáciu [37]. Obe metódy sú schopné pracovať priamo so základnými charakteristikami sieťových tokov. Tieto metódy sú taktiež založené na jednoduchom princípe a ich logika rozhodovania je ľahko vysvetliteľná [38]. Natrénovaný rozhodovací strom je možné vykresliť a preskúmať, na základe akých charakteristík a podmienok sa

sieťové toky klasifikujú a v prípade nejasností je možné pre vybraný sieťový tok ručne prehliadnuť cesty a vetvy, ktoré boli pri klasifikácií použité [39]. To je pre tieto experimenty kľúčové, pretože cieľom je pochopiť a preskúmať, na základe akých informácií a znakov sa metóda rozhodla. Presným opakom týchto metód sú metódy hĺbkového učenia, ktoré nie sú vysvetliteľné [38] a nie je možné preskúmať ich rozhodovaciu logiku.

Druhá časť experimentov sa zameriava na rozdielne klasifikačné metódy strojového učenia. Cieľom tejto časti je identifikácia klasifikačných metód, ktoré dosahujú najvyššiu úspešnosť a vyhodnotiť ich použiteľnosť v reálnej premávke. V rámci tejto časti experimentov vyberiem jednu dátovú sadu na ktorej budem rozdielne metódy testovať. Na základe prieskumu a na základe dokumentácie knižnice *scikit-learn* som vybral niekoľko metód, ktoré sú vhodné na klasifikáciu sieťových tokov. Odporučené použitie klasifikačných metód je zobrazené na obrázku v prílohe A.5. Vybrané klasifikačné metódy sú uvedené v nasledujúcom zozname :

- Decision tree
- K-nearest neighbors
- Extra-tree
- Random forest
- Ada Boost
- Gradient
- Naive Bayes
- MLP
- Linear SVM

V tejto časti experimentov budem taktiež skúmať možnosti pred spracovania dátovej sady. V rámci metód pred spracovania overím vplyv použitia škálovania a normalizácie na úspešnosť klasifikácie. V poslednom experimente sa pokúsim využiť metódy hľadania najlepších hyper parametrov k vylepšeniu úspešnosti niektorých klasifikačných metód. Cieľom je zmerať rozdiel v úspešnosti klasifikačných metód pri využití východzieho nastavenia, kedy sa metóda riadi predom nastavenými pravidlami.

Súhrn týchto experimentov preskúma dostupné možnosti klasifikácie a na základe získaných výsledkov bude navrhnutý klasifikačný modul, ktorý bude klasifikovať sieťové toky v reálnom čase.

4.3 Implementácia experimentov

Na základe prieskumu programovacích jazykov a knižníc pre strojové učenie som pre tieto experimenty zvolil programovací jazyk Python, ktorý obsahuje knižnicu *scikit-learn*. Jazyk Python poskytuje množstvo výhod pri výskume a práci s dátovými sadami. Medzi tieto výhody patrí knižnica Pandas, ktorá sa zameriava na prácu s dátovými sadami. Knižnica *scikit-learn* zase poskytuje množstvo predpripravených funkcií a modelov strojového učenia. Pri voľbe vývojového prostredia som sa rozhodol využívať rozhranie Jupyter Notebook, ktoré poskytuje vynikajúce možnosti interpretácie dát v podobe grafov, tabuliek a taktiež interaktívne prostredie, ktoré umožňuje rozdelenie kódu do blokov ktoré je možné spustiť nezávisle. Jupyter Hub je webové rozhranie, ktoré umožňuje tvorbu a spustenie Jupyter Notebook súborov.

Hlavná časť experimentov sa vykonávala na serveri *netmonlab.fit.cvut.cz*, ktorý spravuje laboratórium monitorovania sieťovej premávky na FIT ČVUT. Tento server už mal nainštalovaný Jupyter Hub a väčšinu potrebných knižníc. Taktiež mi to umožnilo vyhodnocovať časovo náročnejšie experimenty aj niekoľko dní bez potreby prerušenia. Niektoré použité dátové sady obsahovali čísla použitých portov. Čísla portov boli v každom experimente z dátovej sady odstránené.

Aby bolo možné výsledky jednotlivých výsledkov porovnávať, bolo nutné zaviesť unifikovaný spôsob vyhodnotenia a interpretácie výsledkov. Na interpretáciu a vyhodnotenie úspešnosti klasifikačných metód sa v prípade strojového učenia používajú existujúce funkcie knižnice. To umožňuje jednoduchý a rýchly výpočet vyhodnocovacích metrík. Medzi najdôležitejšie metriky úspešnosti klasifikácie patrí celková úspešnosť, precíznosť a f1-skóre. Preto pri všetkých experimentoch budem zaznamenávať výsledky týchto metrík. Vo výsledkoch týchto experimentov by však nemali byť zásadné rozdiely v týchto metrikách, pretože budeme používať z pohľadu klasifikačných kategórií len rovnomerne rozdelené dátové sady.

Medzi odporučené [40] a najčastejšie používané modely vyhodnotenia patrí k-násobná krížová validácia [41] a preto bude v rámci týchto experimentov použitá. Vzhľadom na veľkosť dátových sád a na odporučeniach v manuálových stránkach budem používať 10-násobnú verziu. Vo všeobecnosti tento model nešpecifikuje metódu použitú na rozdelenie dátovej sady na k menších dátových sád, ale funkcia použitej knižnice v základnom nastavení používa *stratified k-fold*, ktorý zaistí rovnomerné rozdelenie klasifikačných kategórií v každej dátovej sade.

V rámci týchto experimentov som nenastavoval žiadne hyper parametre a metódy som nechal pracovať podľa východzích nastavení. Problematike voľby hyper parametrov som sa však venoval v poslednej časti experimentov v sekcií 4.5.1.

4.4 Experimenty s rozdielne obsiahnutými dátovými sadami

4.4.1 *UniFlow*

Pre tento experiment bola použitá dátová sada *UniFlow*. Hlavným účelom tohto experimentu bolo porovnať úspešnosť klasifikácie pôvodného modulu, ktorý využíva základné štatistické metódy s úspešnosťou základných metód strojového učenia na rovnakej dátovej sade. Výsledky klasifikačného modulu z bakalárskej práce však využívajú pôvodnú dátovú sadu, ktorá nemá rovnomerné rozdelené klasifikačné kategórie a preto budeme musieť pri interpretácii výsledkov prihliadnuť na rozdielnosť dátovej sady. Rozdelenie klasifikačných kategórií z pôvodnej dátovej sady je zobrazené na obrázku 3.3. Ďalším zásadným rozdielom je, že klasifikačný modul z práce [2] do určitej miery využíva aj čísla portov pre zvýšenie rýchlosti klasifikácie. V týchto experimentoch je však číslo portu z dátovej sady úplne odstránené. Náhľad použitej dátovej sady je zobrazený na obrázku 3.4.

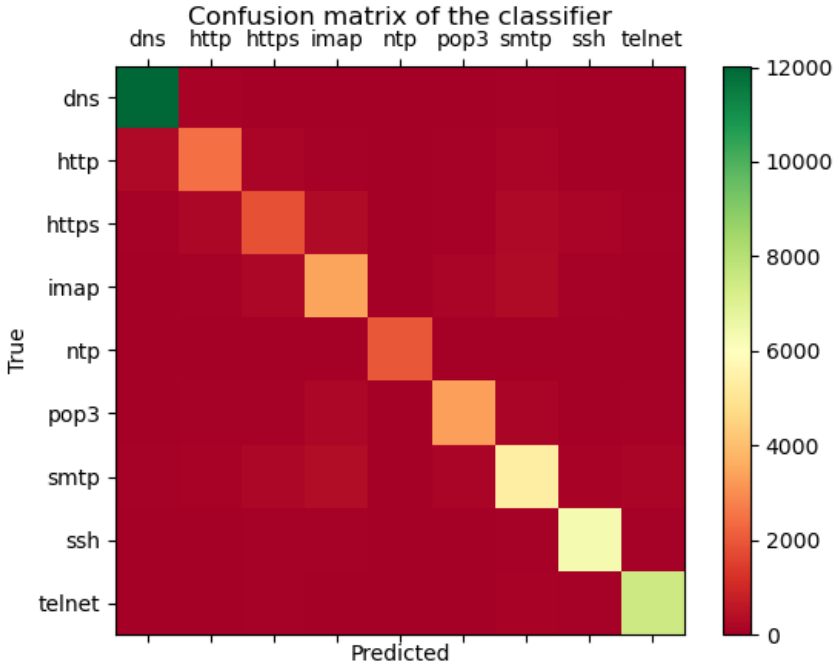
Pri týchto experimentoch som na vyhodnotenie používal len jednu metriku a to celkovú úspešnosť(ACC). Táto metrika bola použitá v bakalárskej práci. Výsledky experimentu sú uvedené v tabuľke 4.1.

Tabuľka 4.1: Výsledky experimentov s dátovou sadou *UniFlow*.

Klasifikátor	**Počet tokov**	**Úspešnosť**	**Presnosť**	**f1-skóre**
DT	90 000	84,7 %	84,7 %	84,7 %
KNN	90 000	78,2 %	78,1 %	78,1 %
Štatistická	2 991 083	92,8 %	neuvedené	neuvedené

Výsledky experimentu som porovnal s výsledkami dosiahnutými v bakalárskej práci, porovnanie výsledkov je uvedené v tabuľke 4.1. Výsledky ukazujú, že štatistický model má o 4 až 5 percent vyššiu úspešnosť. Táto skutočnosť je veľmi prekvapujúca, pretože metódy strojového učenia využívajú sofistikovanejšiu logiku rozhodovania. Dôvodom vysokej úspešnosti klasifikátoru z práce [2] môže byť fakt, že pri klasifikácií využíva aj číslo portu, zatiaľ čo v experimentoch strojového učenia je číslo portu úplne odstránené. Druhým faktorom to že na zníženej úspešnosti metód strojového učenia môže byť použitie neoptimálnych hyper parametrov.

Jedna z výhod použitia knižnice *scikit-learn* je možnosť vykreslenia matice zámeny [27]. Matica zámeny poskytuje potrebné informácie o úspešnosti jednotlivých kategórií. Matica zámeny klasifikátoru DT je zobrazená na obrázku 4.1.



Obr. 4.1: Matica zámeny klasifikátoru DT, pri použití dátovej sady *UniFlow*.

4.4.2 *BiFlow*

V tomto experimente som skúmam rozdiely úspešnosti medzi metódami, ktoré využívali jednosmerné a obojsmerné dátové toky a zároveň budem skúmať, ako zväčšenie dátovej sady ovplyvní úspešnosť klasifikácie. V tomto experimente bola použitá dátová sada *BiFlow* a experiment bol vyhotovený v dvoch variantách. V prvej variante dátová sada obsahovala 10-tisíc tokov z každej kategórie, vďaka čomu je možné porovnať tieto výsledky s výsledkami experimentov, ktoré využívali len jednosmerné toky. Druhá varianta experimentov bude využívať dátovú sadu, ktorá bude obsahovať 40-tisíc tokov z každej kategórie a výsledky tejto varianty následne porovnám s výsledkami prvej varianty.

Výsledky experimentov sú uvedené v tabuľke 4.2 a matice zámeny oboch klasifikátorov sú zobrazené na obrázkoch A.1 a A.2. Náhľad použitých charakteristík dátovej sady s obojsmernými sieťovými tokmi je zobrazený na obrázku 3.5.

Výsledky tohto experimentu využívajúceho 10-tisíc tokov z každej kategórie ukazujú, že použitie obojsmerných tokov zvyšuje úspešnosť klasifikátorov v priemere o 6 %, čo je zásadný rozdiel, keďže dátové sady sa líšia len v dvoch charakteristikách. Použitie obojsmerných tokov v reálnej premávke taktiež môže znížiť počet exportovaných tokov, pretože obojsmerný tok dokáže repre-

Tabuľka 4.2: Výsledky experimentov s dátovou sadou *BiFlow*.

Klasifikátor	Počet tokov	Úspešnosť	Presnosť	f1-skóre
DT	90 000	90,6 %	90,7 %	90,7 %
KNN	90 000	84,7 %	84,7 %	84,7 %
DT	360 000	92,7 %	92,7 %	92,7 %
KNN	360 000	87,3 %	87,4 %	87,3 %

zentovať dva jednosmerné toky. Toto zistenie je teda zásadné a po uvážení všetkých faktov usudzujem, že použitie obojsmerných tokov je značne zvyšuje úspešnosť a efektivitu klasifikácie.

Vo výsledkoch je taktiež vidieť, že zvýšenie počtu tokov zvýšilo úspešnosť. Toto zvýšenie síce nezodpovedá zväčšeniu dátovej sady, ale klasifikátor sa trénuje a testuje na väčšom množstve sieťových tokov, čo by malo zvýšiť kvalitu trénovaných modelov a znížiť riziko pretrénovania klasifikátoru.

4.4.3 *Rozšírené BiFlow*

V týchto experimentoch som využil dátovú sadu *Rozšírené BiFlow*. Rozšírenie PSTATS pridáva k základnému obojsmernému toku celkom štyri charakteristiky. Každá z týchto charakteristík obsahuje informácie o prvých tridsiatich paketoch sieťového toku. Tieto charakteristiky je potrebné predspracovať a extrahovať z nich charakteristiky použiteľné pre klasifikáciu. Metódy strojového učenia totiž nedokážu spracovať charakteristiku, ktorá obsahuje pole hodnôt. Taktiež nie je vhodné tieto hodnoty interpretovať ako samostatné charakteristiky toku. Tieto hodnoty totiž nepopisujú tok samotný, ale len prvé pakety jeho spojenia a nemali by mať rovnakú významnosť, ako charakteristiky sieťového toku.

V rámci rešerše som preskúmal nástroj *FET*, ktorý obsahuje funkciu na extrakciu štatistických vlastností charakteristík PSTATS. Funkcia knižnice *FET* extrahuje mnoho používaných štatistických vlastností a preto som sa rozhodol ju použiť. Náhľad štatistických vlastností extrahovaný knižnicou je zobrazený v sekcií 3.3. V rámci rešerše som taktiež našiel metódu knižnice *scikit-learn*, ktorá implementuje analýzu hlavných komponentov a preto som sa rozhodol v rámci experimentov preskúmať jej možnosti a porovnať jej výsledky oproti extrakcií štatistických vlastností.

Použitie nástroja *FET* vytvorilo a pridalo každému toku 33 charakteristík, celkom každý tok obsahuje 58 charakteristík. V rámci experimentov som vytvoril klasifikátor, ktorý spracoval všetky pridané charakteristiky a otestoval jeho úspešnosť klasifikácie. Tento počet charakteristík je však pre použitie v reálnom čase príliš vysoký a neúmerne zvyšuje výpočtovú náročnosť. Aby som tento nedostatok odstránil, potreboval som identifikovať charakteristiky, ktoré najviac ovplyvňujú úspešnosť klasifikácie. Najdôležitejšie charakteristiky, ktoré identifikoval nástroj *FET* sú k nahliadnutiu v sekcií 3.3. Tieto vý-

sledky sa môžu líšiť v závislosti od použitej klasifikačnej metódy. Pre dosiahnutie ideálnych výsledkov je vhodné vyhodnotiť najdôležitejšie charakteristiky pre každú metódu zvlášť. Na tento účel som použil funkciu *feature_selection* z knižnice *scikit-learn*. Táto funkcia určuje najdôležitejšie charakteristiky pre danú metódu. Následne som podľa výsledkov tejto funkcie upravil dátovú sadu a vytvoril nový klasifikátor, s ktorým som experimenty opakoval. Výsledky tohto experimentu som následne porovnal s výsledkami klasifikátorov, ktoré použili všetky dostupné charakteristiky. Zároveň som porovnal charakteristiky, ktoré vybrala funkcia *feature_selection* charakteristikami vybranými knižnicou *FET*.

Funkcia *feature_selection* umožňuje obmedziť maximálny počet charakteristík. V rámci týchto experimentov som však nechal funkciu vo východzom nastavení. To bol aj dôvod, prečo *feature_selection* rozhodovacieho stromu vybralo celkom 13 charakteristík a k-najbližších susedov len 10 charakteristík. V rámci menších experimentov som zistil, že funkcia vyberá veľmi optimálne riešenia. Manuálnym zvýšením počtu charakteristík o 3 až 5 sa úspešnosť klasifikátorov nezvýšila viac ako o 0,3 %. Pri manuálnom znížení počtu charakteristík o 3 sa úspešnosť znížila mierne, pri znížení počtu o viac ako 3 charakteristiky sa úspešnosť znížila o viac ako 1 %.

Výsledky klasifikátoru využívajúceho všetky charakteristiky a klasifikátoru využívajúceho len vybrané charakteristiky sú uvedené v tabuľke 4.3. Matica zámeny klasifikátorov DT a KNN s 58 charakteristikami sú zobrazené v prílohe A.3 a v prílohe A.4.

Tabuľka 4.3: Výsledky experimentov s dátovou sadou *Rozšírené BiFlow*.

Klasifikátor	**Úspešnosť**	**Presnosť**	**f1-skóre**
DT - 58 charakteristík	96,2 %	96,2 %	96,2 %
DT - 13 charakteristík	95,8 %	95,9 %	95,8 %
KNN - 58 charakteristík	88,4 %	88,3 %	88,4 %
KNN - 10 charakteristík	84,9 %	85,1 %	84,9 %

Tabuľka 4.4: Tabuľka časovej náročnosti experimentov s dátovou sadou *Rozšírené BiFlow*. Stĺpec *Celkový čas* reprezentuje celkový čas vyhodnotenia 10-násobnej krížovej validácie. Stĺpec *Čas 10-tisíc tokov* reprezentuje čas potrebný na spracovanie 10-tisíc tokov, pomocou 10-násobnej krížovej validácie.

Klasifikátor	**Celkový čas**	**Čas 10-tisíc tokov**
DT - 58 charakteristík	305,97 s	8,50 s
DT - 13 charakteristík	79,98 s	2,22 s
KNN - 58 charakteristík	7141,44 s	198,37 s
KNN - 10 charakteristík	228,14 s	6,34 s

Výsledky experimentu jasne ukazujú, že charakteristiky extrahované z PS-

TATS značne zlepšili úspešnosť oboch klasifikátorov v porovnaní s výsledkami *BiFlow*. V tabuľke je taktiež vidieť, že oba klasifikátory dosiahli vyššiu úspešnosť pri využití všetkých charakteristík. Rozdiel v časovej náročnosti je zobrazený v tabuľke 4.4. Výsledky ukazujú, že klasifikácia toku, ktorý obsahuje 58 charakteristík je časovo náročnejšia. V závislosti od požiadaviek na klasifikátor je teda nutné určiť, či zvýšenie časovej náročnosti stojí za zlepšenie klasifikačnej úspešnosti.

```
Index(['protocol ', 'duration', 'bytes_rate', 'packets_total_rate',
       'rst_count', 'urg_ratio', 'lengths_min', 'lengths_mean',
       'fwd_lengths_min', 'fwd_lengths_max', 'fwd_lengths_mean',
       'bwd_lengths_min', 'bwd_lengths_max'],
      dtype='object')
```

Obr. 4.2: Charakteristiky vybrané pomocou funkcie *feature_selection* pre klasifikátor DT.

Výsledky funkcie zobrazené na 4.2, taktiež ukazujú že funkcia *feature_selection*, ktorá vybrala celkom 13 charakteristík, vybrala iba 2 charakteristiky základného obojsmerného toku a 11 charakteristík extrahovaných z PSTATS. Tento fakt dokazuje, že charakteristiky PSTATS obsahujú validné informácie na klasifikáciu kategórie sieťovej premávky. Funkcia taktiež vybrala iba dve charakteristiky, ktoré sa zhodovala s vybranými charakteristikami knižnice *FET*.

4.4.4 Analýza hlavných komponentov

Charakteristiky PSTATS som spracoval aj pomocou analýzy hlavných komponentov(PCA). Cieľom použitia PCA bolo zníženie dimenzionality charakteristík PSTATS. V rámci tohto experimentu som zistil, že nastavenia funkcie knižnice obsahujú veľké množstvo parametrov, ktoré zásadne ovplyvňujú výsledok extrakcie. V rámci experimentov s PCA som sa zameral na charakteristiku *PPI_PKT_LENGTHS*, ktorá obsahuje informácie o veľkosti prvých paketov. Z tejto charakteristiky som extrahoval 1 až 3 charakteristiky a porovnal som úspešnosť klasifikácie. Klasifikátor dosiahol najvyššiu úspešnosť pri použití dvoch komponentov. Rozdiel medzi použitým jedného komponentu bol minimálny a preto som pri spracovávaní ďalších atribútov PSTATS využíval iba jeden komponent. PCA som následne aplikoval aj na atribúty ktoré obsahujú informácie o smere prúdenia prvých paketov a ich TCP príznakoch.

Charakteristika *PPI_PKT_TIMES* obsahuje čas zachytenia prvých paketov v ISO formáte. Metóda analýzy hlavných komponentov však tieto hodnoty nevie priamo spracovať a preto som čas konvertoval do UNIX formátu, ktorý reprezentuje počet sekúnd od 1.januára 1970. Tieto hodnoty však stále nepopisujú kategóriu sieťového toku ale čas, v ktorom bol tok zachytený. Preto som z týchto hodnôt extrahoval čas, ktorý uplynul od zachytenia prvého paketu.

Obsah vytvorenej charakteristiky teda obsahuje pole 30-tích hodnôt a každá hodnota reprezentuje počet sekúnd, ktorý ubehol od prvého paketu toku.

Všetky štyri charakteristiky PSTATS (veľkosť paketov, časové značky, tcp príznaky a smery prúdenia tokov) som spracoval pomocou PCA. To bolo nastavené tak, aby redukovalo dimenzionalitu z pôvodných 30 na 1, čím som získal 1 charakteristiku z každej charakteristiky PSTATS. Experimenty prebiehali na dátovej sade *Rozšírené BiFlow*. Výsledky sú zobrazené v tabuľke 4.5.

Tabuľka 4.5: Výsledky experimentov s dátovou sadou *Rozšírené BiFlow* s využitím PCA.

Klasifikátor	**Úspešnosť**	**Presnosť**	**f1-skóre**
DT	93,6 %	93,5 %	93,6 %

Výsledky ukazujú že extrakcia charakteristík bola úspešná. Úspešnosť klasifikácie sa zvýšila o 1 % v porovnaní s výsledkami *BiFlow*. Zvýšenie úspešnosti je však nižšie, než v prípade použitia extrakcie štatistických vlastností. Metóda extrakcie štatistických vlastností bola úspešnejšia o 2 %.

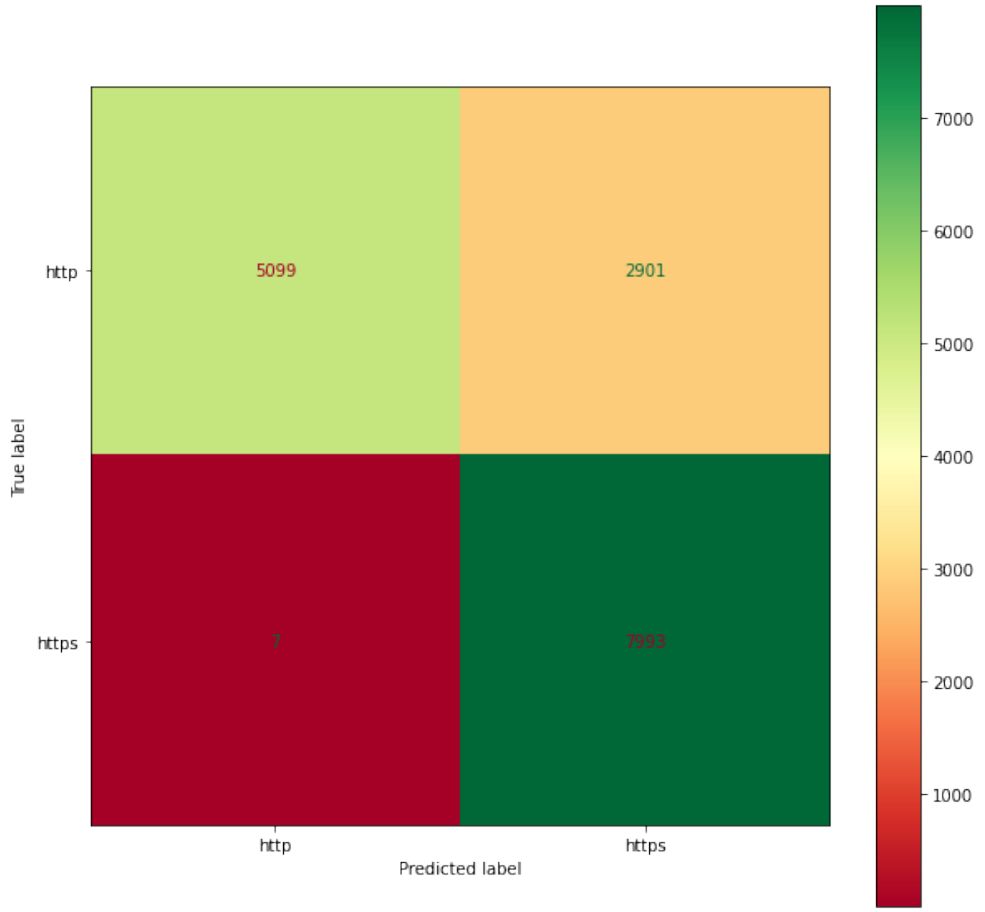
4.4.5 Analýza výsledkov protokolov http/s

V rámci analýzy výsledkov som zistil, že klasifikátor strojového učenia rozoznáva protokoly http a https veľmi úspešne, čo je možné pozorovať napríklad v matici zámeny experimentov s obojsmernými tokmi, zobrazenej v prílohe A.1. Tento fakt bol interesantný, pretože z hľadiska použitia ide o veľmi podobné protokoly. Jedným z hlavných rozdielov týchto protokolov je šifrovanie, ktoré by nemalo zásadne ovplyvniť vlastnosti tokov. Preto som sa rozhodol preskúmať, ako dokáže klasifikátor odlíšiť tieto protokoly s takou vysokou úspešnosťou. Táto analýza prebiehala v súbore *httpVShttps.ipynb*.

Pre túto analýzu som zvolil dátovú sadu *BiFlow*. Túto dátovú sadu som prefiltroval a odstránil som sieťové toky, ktorých kategória nebola http alebo https. Výsledkom je bola dátová sada veľkosti 80-tisíc tokov, 40-tisíc tokov z každej kategórie. V rámci analýzy som sa zameral na metódu rozhodovacích stromov.

Prvým krokom bolo vykreslenie rozhodovacieho stromu. Vykreslenie však nebolo možné, pretože v základnom nastavení mal model klasifikačného stromu celkom 10 úrovní. Vykreslenie a analýza grafu tejto veľkosti by bola veľmi náročná a preto som obmedzil maximálnu hĺbku stromu na 4 a maximálny počet listov na 3. To malo za následok pokles klasifikačnej úspešnosti. Tento fakt však v tomto prípade nie je dôležitý, nakoľko úspešnosť klasifikácie bola stále abnormálne vysoká. V matici zámeny zobrazenej na obrázku 4.3 je vidieť, že aj v tomto obmedzenom nastavení dokázal klasifikátor správne klasifikovať skoro všetky kategórie https a celková úspešnosť klasifikácie bola 82,5 %.

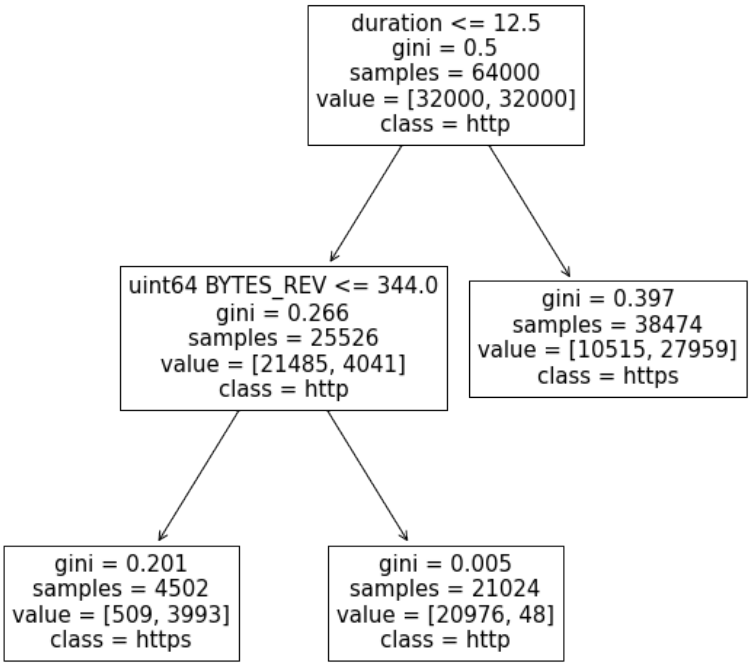
Na vykreslenie rozhodovacieho stromu bola použitá funkcia knižnice *scikit-learn plot_tree*. Vykreslený rozhodovací strom je zobrazený na obrázku 4.4.



Obr. 4.3: Matica zámeny klasifikátoru DT, pri použití dátovej sady *BiFlow*, ktorá obsahuje len protokoly http a https.

Na vykreslenom rozhodovacom strome je vidieť, že klasifikátor na klasifikáciu využíva dve charakteristiky a to trvanie toku a počet bajtov spiatočného toku. Prvý uzol sleduje trvanie toku, v prípade že trvanie je väčšie alebo rovné 12,5 sekúnd, tak je tok klasifikovaný ako https. V prípade, že tok trvá kratšie, klasifikátor v ďalšom uzly sleduje počet bajtov spiatočného toku. V prípade, že počet spiatočných bajtov je menší, alebo rovný 344, tak klasifikátor označí tok ako https, v opačnom prípade ako http. Na základe týchto dvoch hodnôt a dvoch uzlov je teda klasifikátor schopný klasifikovať s nečakane vysokou presnosťou. Z tohto dôvodu som sa rozhodol preskúmať použité dátové sady a preskúmať hodnoty týchto dvoch charakteristík.

Na vykreslenie dátových sád som použil knižnicu *pyplot*. Hodnoty charak-



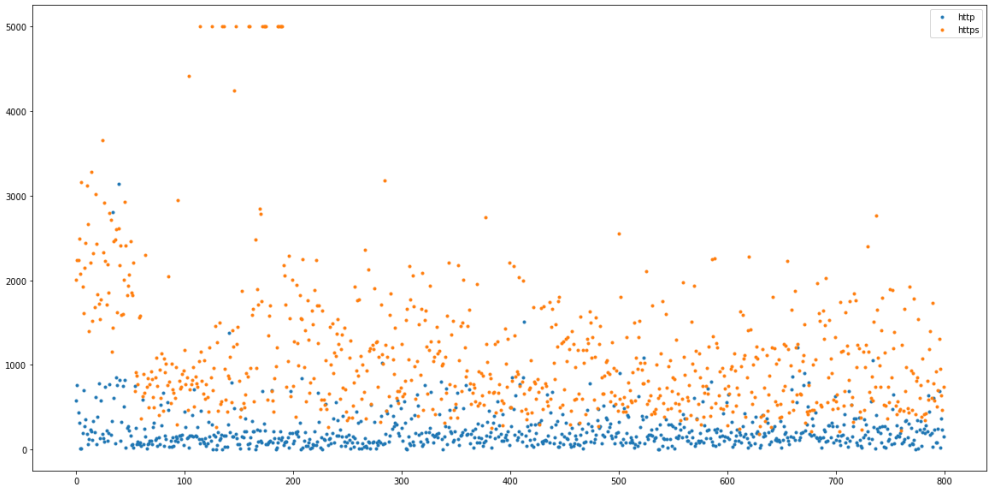
Obr. 4.4: Rozhodovací strom vykreslený pomocou funkcie *plot_tree*.

teristík nemohli byť vzhľadom na veľkosť dátovej sady vykreslené priamo do jedného grafu, preto som musel počet tokov znížiť, alebo dáta nejak predspracovať. Na predspracovanie som vytvoril funkciu, ktorá z každých 100 po sebe idúcich tokov vypočíta strednú hodnotu. V rámci tejto funkcie som taktiež implementoval obmedzenie maximálnej veľkosti hodnoty a v prípade že hodnota presahuje maximálnu hodnotu je znížená na nastavenú hodnotu. Po niekoľkých iteráciách a testovaní som našiel optimálne nastavenie tejto funkcie, ktoré najlepšie vystihovalo pozorované javy.

Ako prvý som vykreslil graf trvania toku, ktorý je zobrazený na obrázku 4.5. V grafe môžeme vidieť, že rozdiel v trvaní tokov http a https je zásadný a toky http premávku trvajú kratšie. Tento fakt bol prekvapujúci, keďže protokoly http a https sa používajú na rovnaký účel. Dôvodom tohto rozdielu však môže byť fakt, že väčšina internetových stránok na sieti CESNET2 už používa protokol https. Maopak protokol http sa využíva väčšinou na presmerovanie prehliadania na https verziu stránky alebo na automatizované dotazy.

Druhá najdôležitejšia charakteristika klasifikácie bola charakteristika počtu spiatočných bajtov. Graf tejto charakteristiky je vykreslený na obrázku

4.6. Na tomto grafe však nie sú vidieť žiadne zásadné rozdiely, čo je veľmi prekvapujúce. Dôvodom je, že zobrazený graf zobrazuje všetky toky z dátovej sady, avšak klasifikátor túto charakteristiku používal len pri tokoch ktoré trvali kratšie ako 12,5 sekundy. Preto som vytvoril ďalší graf v ktorom boli filtrované len toky, ktoré trvali maximálne 12,5 sekundy. Tento graf je zobraný na obrázku 4.7. V tomto grafe už sú vidieť značné rozdiely, vďaka čomu môže klasifikátor jasne odlíšiť jednotlivé protokoly. Skutočnosť, že toky ktoré trvajú kratšie ako 12,5 sekundy je možné odlíšiť pomocou počtu spiatočných bajtov, zatiaľ čo pri tokoch, ktoré trvajú dlhšie ako 12,5 sekundy je odlíšenie nemožné je ukážkovým príkladom, ako metódy strojového učenia dokážu nájsť a extrahovať vzťahy a závislosti v dátovej sade, ktoré nie je možné identifikovať manuálne.

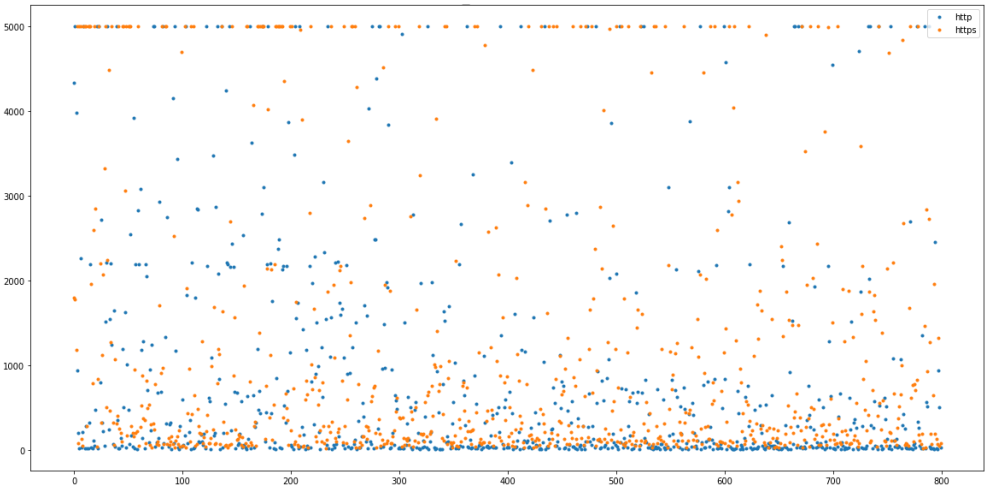


Obr. 4.5: Graf charakteristiky *duration*.

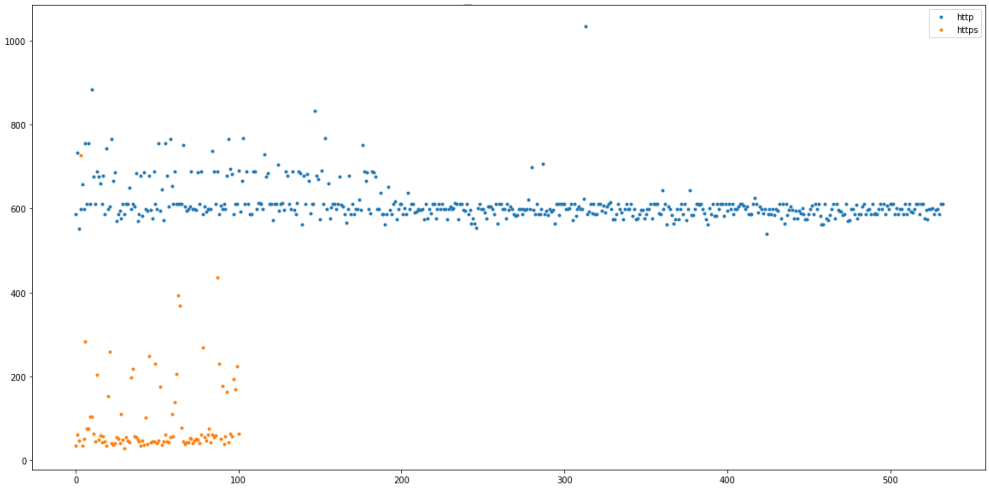
Výsledok tejto analýzy teda ukazuje, že protokoly http a https nemajú v dátovej sade *BiFlow* podobné vlastnosti a použité metódy strojového učenia dokážu rozdiely veľmi jednoducho rozpoznať.

4.5 Experimenty s metódami strojového učenia

Po otestovaní a porovnaní úspešnosti klasifikácie nad rozdielne obsiahnutými dátovými sadami bolo ďalším krokom porovnanie rozdielnych klasifikačných metód. Po preštudovaní dokumentácie knižnice *scikit-learn* som vybral metódy strojového učenia, ktoré spĺňali požiadavky týchto experimentov. Jednoduché roztriedenie jednotlivých metód knižnice je zobrazené na obrázku v prílohe A.5. Vybrané metódy strojového učenia ktoré som testoval sú zobrazené v zozname, v sekcií 4.2. Medzi testované metódy som zaradil aj metódy roz-



Obr. 4.6: Graf charakteristiky *bytes*.



Obr. 4.7: Prefiltrovaný graf charakteristiky *bytes*.

hodovacích stromov a k-najbližších susedov, čo mi umožní porovnanie týchto metód v rovnakých podmienkach.

Pri voľbe dátovej sady som sa rozhodol použiť najúspešnejšiu dátovú sadu z predchádzajúcich experimentov a to dátovú sadu *Rozšírené BiFlow*. V rámci týchto experimentov som dátovú sadu obmedzil na 9 tisíc tokov celkom, pričom každá klasifikačná kategória mala rovnaké zastúpenie. Na tento účel som použil funkciu *stratified k-fold*. Taktiež som využíval len 3-násobnú krížovú validáciu. Tieto obmedzenia som bol nútený urobiť kvôli časovej náročnosti. Na extrakciu znakov PSTATS som použil knižnicu *FET*, ktorá extrahuje štatistické vlastnosti týchto charakteristík. Tento postup sa v predchádzajúcich

experimentoch preukázal ako veľmi spoľahlivý a dosiahol vynikajúce výsledky.

Tieto experimenty prebiehali v súbore s názvom *ClassifiersComparison.ipynb*. Výsledky experimentov sú uvedené v tabuľke 4.6. Ako hlavnú klasifikačnú metriku som zvolil celkovú úspešnosť. Vo výsledkoch predchádzajúcich experimentov je vidieť, že výsledky tejto metriky sa v používanom systéme vyhodnotenia nijak zásadne nelíšia od iných zložitejších metrík. V tabuľke je taktiež uvedený celkový čas vyhodnotenia krížovej validácie. Tento údaj môže byť zavádzajúci, avšak môže slúžiť ako hrubý odhad časovej náročnosti jednotlivých klasifikačných metód.

Tabuľka 4.6: Výsledky experimentov s rozdielnymi metódami strojového učenia s použitím dátovej sady *Rozšírené BiFlow*. Stĺpec *Celkový čas* reprezentuje celkový čas vyhodnotenia 3-násobnej krížovej validácie.

Klasifikátor	**Úspešnosť**	**Celkový čas**
DT	91,7 %	9 s
KNN	80,8 %	154 s
Extra-tree	93,9 %	28 s
Random forest	93,8 %	68 s
Ada Boost(DT)	93,8 %	373 s
Gradient	91,5 %	1905 s
Naive Bayes	22,0 %	2 s
MLP	67,1 %	250 s
Linear SVM	60,9 %	3555 s

V rámci tohto experimentu sa ukázalo, že metódy využívajúce klasifikačné stromy dosahujú v priemere lepšie výsledky a majú nižšiu časovú náročnosť než ostatné metódy.

Ďalším krokom bolo preskúmať ako normalizácia škálovanie dátovej sady zmení klasifikačnú úspešnosť jednotlivých klasifikátorov. Experimenty so škálovaním a normalizáciou prebiehali v súbore s názvom *ClassifierComparison.ipynb*. V rámci tohto experimentu som vytvoril celkom 9 upravených dátových sád, každú o veľkosti 9-tisíc tokov. Tieto dátové sady vznikli použitým funkcií *preprocessing.scale* a *preprocessing.normalize* na dátovú sadu použitú v predchádzajúcom experimente. Upravené dátové sady som následne vyhodnotil rovnakým spôsobom, ako v predchádzajúcom experimente. Z výsledkov som zistil, že každý klasifikátor vyžaduje inak upravené dátové sady a nie je možné definovať najlepšiu úpravu dátovej sady nezávisle na klasifikátore.

Posledným krokom tohto experimentu bol výber najlepšej úpravy pre daný klasifikátor a vyhodnotenie klasifkátora na plnohodnotnej dátovej sade, ktorá obsahuje 360 tisíc tokov a taktiež použitie 5-násobnej krížovej validácie. Výsledok, čas a použitá úprava dátovej sady je uvedená v tabuľke 4.7.

Tabuľka 4.7: Najlepšie dosiahnuté výsledky, jednotlivých metód strojového učenia s použitím dátovej sady *Rozšírené BiFlow*. Stĺpec *Úprava* reprezentuje použitú metódu predspracovania.

Klasifikátor	**Úspešnosť**	**Úprava**
DT	93,9 %	normalizácia
KNN	93,7 %	normalizácia
Extra-tree	95,1 %	škálovanie
Random forest	95,1 %	škálovanie
Ada Boost(DT)	94,3 %	škálovanie
Gradient	94,1 %	škálovanie
Naive Bayes	59,8 %	normalizácia
MLP	96,1 %	škálovanie
Linear SVM	82,3 %	škálovanie

4.5.1 Metódy hľadania hyper parametrov

V porovnaní klasifikačných metód dosiahla najlepšie výsledky metóda Extra-tree. V tomto porovnaní som však neskúmal možnosti nastavenia hyper parametrov a využíval som východzie nastavenia. V tomto nastavení sa metóda riadi dopredu nastavenými pravidlami. Metóda však poskytuje možnosť nastavenia určitých parametrov a preto som sa rozhodol preskúmať, ako zmena týchto nastavení ovplyvní klasifikátor. Metóda Extra-tree umožňuje nastavenie nasledujúcich hyper parametrov :

- ***n_estimators*** : Počet stromov v lese. Východzie : 100
- ***criterion*** : Funkcia ktorá merá kvalitu rozdelenia. Východzie : *gini*
- ***max_depth*** : Maximálna hĺbka stromov. Východzie : ľubovolná
- ***max_features*** : Maximálni počet charakteristík využívaných na hľadanie najlepšieho rozdelenia. Východzie : $\sqrt{n\_features}$
- ***min_samples_split*** : Minimálny počet prvkov potrebných na rozdelenie uzlu. Východzie : 2

Ručné natavenie rôznych hodnôt hyper parametrov a následné vyhodnotenie je však veľmi časovo náročné. Knižnica *scikit-learn* obsahuje metódu *GridSearchCV* a *RandomizedSearchCV*. Tieto metódy prijímajú na vstup klasifikátor, dátovú sadu a tabuľu hyper parametrov. Tabuľka hyper parametrov obsahuje rôzne nastavenie jednotlivých parametrov, ktoré metóda využije k tréningu klasifikátora. Metóda *GridSearchCV* otestuje všetky dostupné kombinácie parametrov v tabuľke, zatiaľ čo metóda *RandomizedSearchCV* testuje rôzne kombinácie náhodne, až dokým nedosiahne určený počet iterácií.

Tabuľka parametrov použitá v tomto experimente je zobrazená na obrázku 4.8. Pre tento experiment som sa rozhodol využiť metódu *GridSearchCV* a dátovú sadu *Rozšírené BiFlow*. Kvôli časovej náročnosti som na dátovú sadu aplikoval *stratified k-fold* a dátovú sadu zmenšil tak, aby obsahovala len 10-tisíc tokov každého protokolu.

```
{'n_estimators': [50, 100, 150, 200],
 'criterion': ['gini', 'entropy'],
 'max_depth': [10, 20, 30, 40, 50],
 'max_features': ['sqrt', 'log2'],
 'min_samples_split': [2, 7, 12]}
```

Obr. 4.8: Tabuľka parametrov pre funkciu *GridSearchCV*.

Hyper parametre, ktoré boli vybrané funkciou ako najlepšie sú zobrazené na obrázku 4.9. Kvalitu vybraných hyper parametrov som následne otestoval na plnohodnotnej dátovej sade *Rozšírené BiFlow*. V rámci experimentu som vytvoril dva klasifikátory ktoré využívali metódu Extra-tree. Prvý klasifikátor nemal nastavené žiadne hyper paramtre a bol vo východziom nastavení, zatiaľ čo druhý klasifikátor mal nastavený hyper parametre podľa výstupu metódy *GridSearchCV* 4.9. Následne som oba klasifkátory vyhodnotil 10-násobnou krížovou validáciou. Výsledky experimentu sú zobrazené v tabuľke 4.8. Výsledky ukazujú, že nastavenie hyper parametrov nijak merateľne nezvýšilo úspešnosť klasifikácie a ich nastavenie v tomto prípade nie je nutné.

Tabuľka 4.8: Výsledku experimentu hľadania najlepších hyper parametrov klasifátoru Extra-tree.

Klasifikátor	**Úspešnosť**	**Presnosť**	**f1-skóre**
Extra-tree bez hyper parametrov	95,2 %	95,7 %	95,2 %
Extra-tree s hyper parametrami	95,2 %	95,8 %	95,2 %

```
{'criterion': 'gini',
 'max_depth': 40,
 'max_features': 'sqrt',
 'min_samples_split': 2,
 'n_estimators': 200}
```

Obr. 4.9: Výstup funkcie *GridSearchCV*. Výstup zobrazuje hyper parametre, ktoré dosiahli najvyššiu úspešnosť.

4.6 Zhrnutie výsledkov experimentov

V tejto kapitole bola vytvorená a vyhodnotená séria experimentov. Prvá časť experimentov sa venovala skúmaniu dátových sád a vplyvom rozšírených sieťových tokov na úspešnosť klasifikácie. V rámci tejto časti som tiež skúmal možnosti extrakcie charakteristík PSTATS. V druhej časti experimentov som sa venoval rozdielnym metódam klasifikácie a vplyvom normalizácie a škálovania na jednotlivé klasifikačné metódy.

Prvé experimenty skúmali možnosti klasifikácie jednosmerných sieťových tokov pomocou metód strojového učenia. Výsledky týchto experimentov jasne ukázali, že metódy strojového učenia dokážu rozoznávať rozdielne kategórie sieťovej premávky a to bez použitia čísla portu. Zároveň som výsledky týchto experimentov porovnal s výsledkami dosiahnutými v bakalárskej práci, ktorá na klasifikáciu sieťových tokov využívala intervaly spoľahlivosti. V tomto porovnaní použité metódy strojového učenia dosahovali nižšie výsledky, než klasifikátor z relevantnej práce. Táto skutočnosť môže byť spôsobená tým, že metódy strojového učenia nepoužívali číslo sieťového portu zatiaľ čo klasifikátor z mojej bakalárskej práce v obmedzenej miere využíval aj číslo portu a to na zrýchlenie procesu klasifikácie.

Ďalej som skúmal ako použitie obojsmerných tokov a rozšírenie PSTATS ovplyvní úspešnosť klasifikácie. V rámci týchto experimentov som zistil, že použitie obojsmerných tokov zvyšuje úspešnosť klasifikácie v priemere o 6 %. Pri skúmaní možností extrakcie atribútov PSTATS som preskúmal možnosti knižnice *FET* a vyhodnotil som využitie štatistických vlastností charakteristík PSTATS. Pri použití extrakcie štatistických vlastností sa zvýšila úspešnosť klasifikácie v priemere o 2 %. Časová náročnosť na výpočet všetkých 58 štatistických vlastností je však privysoká a preto som na tieto dáta aplikoval metódu *feature_selection*. Výsledkom bolo zníženie počtu charakteristík z 58 na 10 až 13 a zníženie úspešnosti klasifikácie o 0 až 3 %. Na extrakciu atribútov PSTATS som taktiež využil metódu analýzy hlavných komponentov. Využitím tejto metódy som bol schopný extrahovať charakteristiky PSTATS a vytvoriť niekoľko komponentov ktoré reprezentovali nové charakteristiky. Dosiahnuté výsledky zobrazené v tabuľke 4.5 však dokazujú, že použitie tejto metódy extrakcie je možné a extrahované komponenty zvýšili úspešnosť klasifikácie v porovnaní so základnými obojsmernými tokmi. Avšak v rámci porovnania metód extrakcie, extrakcia pomocou výpočtu štatistických vlastností charakteristík PSTATS dosahovala lepšie výsledky a taktiež nároky na výpočet charakteristík boli menšie.

V druhej časti experimentov som sa sústredil na porovnanie úspešnosti rôznych klasifikačných metód strojového učenia. Metódy som testoval na veľkostne obmedzenej dátovej sade, ktorá obsahovala extrahované štatistické vlastnosti PSTATS. Výsledky týchto experimentov sú zobrazené v tabuľke 4.6. Tieto výsledky ukázali, že metódy využívajúce nejakú variantu rozhodovacích stromov dosahovali v priemere lepšiu úspešnosť. Výsledky tohto expe-

rimentu budú použité pri voľbe klasifikačnej metódy klasifikačného modulu. V rámci týchto experimentov som taktiež skúmal vplyv škálovania a normalizácie dátovej sady na úspešnosť klasifikácie. V rámci týchto experimentov som nedokázal určiť špecifickú úpravu dátovej sady, ktorá by pomohla zvýšiť úspešnosť klasifikácie všetkých metód. Výsledky ukazujú že každá klasifikačná metóda vyžaduje rozdielnu úpravu dátovej sady. Z pozorovaní som taktiež zistil, že metódy, ktoré využívajú rozhodovacie stromy sú na úpravy dátovej sady menej citlivé, než iné metódy. V poslednom experimente som preskúmal možnosti nastavenia hyper parametrov metódy Extra-tree. Pomocou funkcie *GridSearchCV* som našiel hyper parametre, s ktorými metóda dosahovala najlepšie výsledky. Nastavením týchto hyper parametrov sa úspešnosť klasifikácie nijako nezmenila.

Výsledky týchto experimentov jasne ukazujú že metódy strojového učenia je možné použiť na klasifikáciu sieťového protokolu bez použitia čísla portu. Tieto metódy sú totiž schopné nájsť závislosti v dátovej sada a tak kvantifikovať rozdiely medzi jednotlivými kategóriami sieťovej premávky. Výsledky týchto experimentov budú využité pri vytváraní nového klasifikačného modulu určeného na klasifikáciu v reálnom čase.

Kapitola 5

Klasifikačný modul

Táto kapitola popisuje návrh, implementáciu a testovanie klasifikačného modulu. Na základe výsledkov experimentov z predchádzajúcej kapitoly som vybral a implementoval variantu, ktorá dosahovala najlepšie výsledky.

5.1 Návrh a implementácia

Klasifikačný modul sa skladá z dvoch hlavných častí, prvá časť je určená na trénovanie klasifikačných modelov a druhá na klasifikáciu tokov. Prvá časť bude prijímať anotované dátové sady a vytvárať z nich klasifikačné modely, ktoré budú následne ukladané do súboru. Klasifikačná časť modulu bude pri spustení načítavať natrénované klasifikačné modely z tohto súboru a následne bude prijímať prichádzajúce sieťový toky cez rozhranie TRAP. Prichádzajúce toky vyhodnotí, a k existujúcim atribútom sieťových tokov pridá ďalší atribút, v ktorom bude uložený výsledok klasifikácie. Následne spracované sieťové toky odošle cez výstupné rozhranie.

Rozdelenie tréningovej a klasifikačnej časti implikuje niekoľko výhod. Prvou výhodou je možnosť použitia rozdielneho užívateľského rozhrania. Vďaka tomu je možné použiť pre tréningovú časť interaktívne a viac graficky orientované rozhranie, vďaka čomu je tvorba klasifikačných modelov jednoduchšia a prehľadnejšia. Naopak používateľské rozhranie klasifikačnej časti môže byť strohé, pretože táto časť bude bežať v reálnom čase na serveri a nebude obsluhovaná užívateľom. Taktiež nároky na časovú a pamäťovú náročnosť tréningovej časti sú menšie než u klasifikačnej časti. Trénovanie totiž nebude prebiehať tak často a preto tak nezáleží na jeho rýchlosti. Naopak v prípade klasifikačnej časti je rýchlosť jedna z kľúčových požiadavok.

Prepojenie týchto častí bude sprostredkované pomocou súboru obsahujúceho klasifikačné modely. Výhodou tohto prepojenia je, že tréning nemusí prebiehať na zariadení, kde sa nachádza klasifikátor. Tréning môže prebiehať na inom, menej vyťaženom zariadení a klasifikačné modely môžu byť ná-

sledne poslané na požadované zariadenia, alebo dokonca viaceré zariadenia. Toto rozdelenie taktiež umožní tréning klasifikačných modelov aj v súkromných sieťach. Partneri tak môžu natrénovať a zdielať klasifikačné modely bez vystavenia a ohrozenia dát v ich sieti.

Implementácia klasifikačného modulu prebiehala v jazyku Python. Tréningová časť využíva užívateľské rozhranie Jupyter notebook, zatiaľ čo klasifikačná časť je implementovaná bez užívateľského rozhrania. Monitorovací systém NEMEA využíva na komunikáciu rozhranie TRAP. Toto rozhranie umožňuje komunikáciu medzi jednotlivými modulmi alebo prenos dát. Pre prenos sieťových tokov sa v tomto systéme najčastejšie používa formát Unirec. Klasifikačný modul obsahuje jedno výstupné a jedno vstupné TRAP rozhranie. Obe rozhrania budú podporovať formát Unirec.

Finálna verzia klasifikačného modulu by mala byť schopná prispôsobiť sa formátu prichádzajúcich tokov. V prípade že na vstupe budú len jednosmerné sieťové toky, modul by mal byť schopný prispôsobiť sa a klasifikovať aj tento typ tokov. Táto požiadavka si vyžaduje vytvorenie niekoľkých klasifikačných modelov pre každú klasifikačnú kategóriu. Nie je totiž možné klasifikovať jednosmerné toky s využitím modelov pre obojsmerné toky. Táto funkcionalita taktiež vyžaduje prispôsobenie vstupného a výstupného rozhrania. Po dohode s vedúcim práce bolo rozhodnuté obmedziť sa len na jeden typ sieťových tokov. Na základe výsledkov experimentov som sa rozhodol použiť typ sieťových tokov, ktorý dosahoval najlepšie výsledky. Táto verzia klasifikačného modulu bude teda podporovať len obojsmerné sieťové toky, ktoré obsahujú rozšírenie PSTATS. Na extrakciu charakteristík PSTATS som sa rozhodol využiť knižnicu *FET*. Metóda extrakcie štatistických vlastností dosiahla vynikajúce výsledky a v prípade potreby, bude možné výpočet štatistických vlastností implementovať špecializovanými funkciami a zvýšiť tak rýchlosť klasifikácie.

V rámci experimentov v predchádzajúcej kapitole som identifikoval najúspešnejšie metódy strojového učenia 4.6. Najúspešnejšia klasifikačná metóda bola metóda Extra-tree. Táto metóda dosahovala na všetkých dátových sadách najlepšie výsledky z testovaných metód a jej nároky na časovú a pamäťovú náročnosť boli akceptovateľné. Preto som sa rozhodol použiť práve túto metódu.

Pri klasifikácií sieťových tokov v reálnom čase je predspracovanie prichádzajúceho toku veľmi neefektívne. Pred úpravou atribútov toku je totiž nutné vytvoriť kópiu, pôvodných hodnôt. Dôvodom je, že klasifikačný modul preposiela atribúty prichádzajúceho toku v nezmenenej podobe na výstup. Škálovanie a normalizácia dátovej sady pri použití metódy Extra-tree znateľne nezlepšila úspešnosť klasifikácie a preto som sa rozhodol ponechať charakteristiky toku v základnej podobe.

5.2 Testy a nasadenie

Vytvorený klasifikačný modul som pred nasadením testoval na zhromaždených dátových sadách. Klasifikačný modul však nepodporuje žiadne vyhodnocovacie metódy. Preto bolo nutné aplikovať vyhodnocovaciu metódu manuálne. Pre vyhodnotenie som sa rozhodol použiť rovnakú metódu ako v experimentoch a to k-násobnú krížovú validáciu. Keďže aplikácia metódy prebiehala manuálne rozhodol som sa použiť 5-násobnú validáciu miesto 10-násobnej. Dátovú sadu, ktorá obsahovala 360-tisíc tokov celkom a 40-tisíc tokov z každej klasifikačnej kategórie som rozdelil na 5 menších dátových sád, ktoré obsahovali 72-tisíc tokov celkom. Aby som zamedzil nerovnomernému rozdeleniu klasifikačných kategórie medzi jednotlivými dátovými sadami použil som *stratified k-fold* metódu, ktorá zaistí rovnomerné rozdelenie klasifikačných kategórií v jednotlivých dátových sadách. Následne som aplikoval metódu 5-násobnej krížovej validácie. V každej z piatich iterácií som vybral jednu dátovú sadu ako testovaciu a zvyšné štyri dátové sady som použil na tréning modelu. Testovacia dátová sada nebola v žiadnej iterácií rovnaká. Natrénovaný model som následne použil na vyhodnotenie testovacej dátovej sady. Výsledky každej iterácie som si uložil. Po ukončení testovania som výsledky všetkých iterácií spriemeroval.

Výsledné hodnoty úspešnosti, presnosti a f1-skóre je uvedené v tabuľke 5.1. Čas potrebný na klasifikáciu 72-tisíc tokov bol v priemere 1,3 sekundy, z čoho plynie že modul je schopný klasifikovať 55-tisíc tokov za sekundu.

Tabuľka 5.1: Výsledky klasifikačného modulu.

Klasifikátor	**Úspešnosť**	**Presnosť**	**f1-skóre**
Klasifikačný modul (Extra-tree)	94,8 %	95,2 %	94,8 %

Dosiahnuté výsledky klasifikačného modulu zodpovedajú výsledkom experimentu, v ktorom boli použité rozšírené dátové sady bez použitia škálovania alebo normalizácie. Modul teda funguje správne a podľa očakávaní. Po úspešnom testovaní bol modul nasadený do testovacej prevádzky na sieť „netmonlab“. Táto sieť je využívaná výskumným tímom laboratória monitorovania sieťovej premávky a nachádza sa na FIT ČVUT v Prahe. Na tejto sieti je nasadení monitorovací systém NEMEA, ktorý monitoruje všetku premávku na sieti. V čase písania tejto práce sa klasifikačný modul testuje na reálnych dátach a zatiaľ nedošlo k žiadnym zlyhaniam modulu a modul pracuje podľa očakávaní.

Záver

V rámci tejto práce som preskúmal možnosti využitia rozšírených sieťových tokov a metód strojového učenia ku klasifikácií sieťového protokolu bez využitia čísla portu. V prvej časti práce som sa zaoberal prieskumom dostupných technológií a nástrojov. Zoznámil som sa s metódami monitorovania sieťovej premávky s pomocou sieťových tokov, pričom som venoval zvýšenú pozornosť rozšíreniu PSTATS. V rámci prieskumu rozšírených charakteristík PSTATS som popísal metódy používané na extrakciu dát a preskúmal som knižnicu *FET*. Túto knižnicu som taktiež využil na reprezentáciu a prieskum štatistických vlastností atribútov PSTATS. Počas prieskumu aktuálne dostupných klasifikačných nástrojov som sa sústredil na prieskum metód strojového učenia, s ktorými som nemal predchádzajúce skúsenosti. V rámci prieskumu som sa zoznámil s jednotlivými metódami, používanými na klasifikáciu, ale aj s postupom a metódami používanými pri práci s metódami strojového učenia. V rámci prieskumu som taktiež preskúmal rozsiahle možnosti a metódy využívané na spracovanie dátových sád a na vyhodnotenie úspešnosti. Medzi tieto metódy patrí metóda extrakcie hlavných komponentov, metóda výberu najdôležitejších charakteristík, alebo metóda *GridSearchCV*, určená na hľadanie optimálnych hyper parametrov. Na vyhodnotenie úspešnosti klasifikácie som preskúmal možnosti vyhodnotenia pomocou k-násobnej krížovej validácie a zameral som sa taktiež na metriky používané na vyhodnotenie úspešnosti klasifikácie, ako napríklad presnosť a f1-skóre.

Kvalita dátových sád je kľúčovým elementom pri experimentoch so strojovým učením. Poznatky z rešerše a z predchádzajúcej práce som aplikoval pri tvorbe nových dátových sád. S pomocou vedúceho práce som zachytil a anotoval celkom 360-tisíc sieťových tokov. Dátová sada je tvorená 9 sieťovými protokolmi, pričom každý protokol je v sade zastúpený 40-tisíc tokmi. Táto dátová sada bola použitá na väčšinu experimentov v tejto práci a taktiež na vytvorenie modelov pre klasifikačný modul. Dátová sada sa môže využiť v rámci budúcej práce, alebo na relevantný výskum v rámci oddelenia monitorovania sieťovej premávky.

Prostredníctvom tejto práce som vytvoril sadu experimentov, ktoré mali za cieľ preskúmať a ohodnotiť rôzne spôsoby klasifikácie sieťových tokov. Prvá časť experimentov skúmala vplyvy rozdielne obsiahnutých dátových sád na úspešnosť klasifikácie. V týchto experimentoch som sledoval zmenu úspešnosti a časovej náročnosti pri použití jednosmerných, obojsmerných a obojsmerných rozšírených tokov. V tejto časti som taktiež skúmal metódy extrakcie charakteristík PSTATS. Po vyhodnotení výsledkov som sa zameral na použité klasifikačné metódy a vytvoril som sadu experimentov, ktoré porovnávali úspešnosť jednotlivých klasifikačných metód na rovnakej dátovej sade. V poslednej časti experimentov som sa zameral na úpravy dátových sád a na výber najvhodnejších klasifikačných charakteristík. Výsledky týchto experimentov sú k nahliadnutiu v kapitole 4.5. Výsledky všetkých experimentov som preskúmal, porovnal a vyhodnotil som najvhodnejšiu metódu klasifikácie sieťových tokov s prihliadnutím na časovú a výpočtovú náročnosť.

Na základe výsledkov experimentov som vytvoril klasifikačný modul, ktorý je schopný klasifikovať sieťové toky na sieti v reálnom čase. Klasifikačný modul som navrhol a implementoval tak, aby bolo trénovanie klasifikačných modelov nezávislé na samotnom klasifikačnom module. Jediné prepojenie tréningovej a klasifikačnej časti modulu je pomocou súboru klasifikačných modelov. Vďaka tomu je možné trénovať modely v kontrolovanom prostredí na uzavretej sieti a v prípade potreby natrénované modely rozposlať na viacero klasifikačných modulov. Tento modul dosiahol počas testov výborné výsledky a aktuálne je nasadení na sieti *netmonlab*, kde nepretržite klasifikuje sieťovú premávku a vyhodnocuje sa jeho dlhodobý prínos.

V rámci tejto práce boli splnené všetky plánované ciele. Dôkladne som preskúmal možnosti klasifikácie sieťových protokolov bez použitia čísla portu. V rámci budúcej práce sa chcem venovať rozšíreniu dátových sád o ďalšie protokoly a implementácií efektívnejšej extrakcie štatistických vlastností PSTATS. Ďalej by som chcel overiť možnosti klasifikácie všeobecných kategórií sieťovej premávky ako napríklad prehrávanie videa, sťahovanie a automatická komunikácia. V rámci budúcej práce by som chcel taktiež preskúmať možnosti klasifikácie obsahu šifrovanej premávky.

Literatúra

[1] Labovitz, C.: Internet Traffic 2009-2019. *Presentation at NANOG*, ročník 76, 2019: s. 4–6.

[2] Hulák, M.: *Klasifikace provozu a zařízení v počítačových sítích na základě tok*. B.S. thesis, České vysoké učení technické v Praze. Vypočetní a informační centrum., 2020.

[3] Alani, M. M.: Tcp/ip model. In *Guide to OSI and TCP/IP models*, Springer, 2014.

[4] Cotton, M.; Eggert, L.; Touch, D. J. D.; aj.: Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry. RFC 6335, Srpen 2011, doi:10.17487/RFC6335. Dostupné z: `https://rfc-editor.org/rfc/rfc6335.txt`

[5] Mudrák, D.: Schéma zapouzdření aplikačních dat na jednotlivých vrstvách rodiny protokolů TCP/IP [online]. Wikimedia Commons [vid. 2021-03-04]. Dostupné z: `https://commons.wikimedia.org/wiki/File:Tcpip_zapouzdreni.svg`

[6] Sinha, R.; Papadopoulos, C.; Heidemann, J.: Internet packet size distributions: Some observations. *USC/Information Sciences Institute, Tech. Rep. ISI-TR-2007-643*, 2007.

[7] Hofstede, R.; Čeleda, P.; Trammell, B.; aj.: Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX. *IEEE Communications Surveys Tutorials*, ročník 16, č. 4, 2014: s. 2037–2064.

[8] Cisco Systems, Inc.: NetFlow gives network managers a detailed view of application flows on the network [online]. Cisco Systems, Inc. [vid. 2021-03-24]. Dostupné z: `https://www.cisco.com/c/dam/en/`

us/products/collateral/ios-nx-os-software/ios-netflow/prod_case_study0900aecd80311fc2.pdf

[9] Cisco Systems, Inc.: NetFlow Export Datagram Format [online]. Cisco Systems, Inc. [vid. 2021-03-24]. Dostupné z: https://www.cisco.com/c/en/us/td/docs/net_mgmt/netflow_collection_engine/3-6/user/guide/format.html

[10] Claise, B.: Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information. RFC 5101, Leden 2008, doi:10.17487/RFC5101. Dostupné z: https://rfc-editor.org/rfc/rfc5101.txt

[11] Pahi, T.; Leitner, M.; Skopik, F.: Analysis and Assessment of Situational Awareness Models for National Cyber Security Centers. In *ICISSP*, 2017, s. 334–345.

[12] Svoboda, J.; Ghafir, I.; Prenosil, V.; aj.: Network monitoring approaches: An overview. *Int J Adv Comput Netw Secur*, ročník 5, č. 2, 2015: s. 88–93.

[13] Cejka, T.; Bartos, V.; Svepes, M.; aj.: NEMEA: a framework for network traffic analysis. In *2016 12th International Conference on Network and Service Management (CNSM)*, IEEE, 2016, s. 195–201.

[14] CESNET: Nemea [online]. GitHub. [vid. 2021-02-09]. Dostupné z: https://github.com/CESNET/Nemea

[15] CESNET: UniRec [online]. GitHub. [vid. 2021-05-05]. Dostupné z: https://github.com/CESNET/Nemea-Framework/tree/master/unirec

[16] Velan, P.; Krejčí, R.: Flow information storage assessment using IPFIXcol. In *IFIP International Conference on Autonomous Infrastructure, Management and Security*, Springer, 2012, s. 155–158.

[17] Hutak, L.: A New Generation of an IPFIX Collector. In *7 th Prague Embedded Systems Workshop*, 2019, str. 33.

[18] CESNET: IPFIX flow exporter [online]. GitHub. [vid. 2021-08-09]. Dostupné z: https://github.com/CESNET/ipfixprobe

[19] CESNET: pytrap [online]. GitHub. [vid. 2022-03-04]. Dostupné z: https://github.com/CESNET/Nemea-Framework/tree/master/pytrap

[20] Lamping, U.; Warnicke, E.: Wireshark user's guide. *Interface*, ročník 4, č. 6, 2004: str. 1.

[21] Wireshark User's Guide [online]. Wireshark. [vid. 2022-02-02]. Dostupné z: https://www.wireshark.org/docs/wsug_html/wsug_graphics/ws-main.png

[22] DUDEK, P. B. J.; HOLKOVIČ, M.: DETEKCE SÍŤOVÝCH ÚTOK POMOCÍ NÁSTROJE TSHARK.

[23] Alcock, S.; Nelson, R.: Libprotoident: traffic classification using lightweight packet inspection. Technická zpráva, Technical report, University of Waikato, 2012.

[24] Matěj, B.: *Časové agregace a extrapolace dat profil síť ových zařízení*. Diplomová práce, České vysoké učení technické v Praze. Vypočetní a informační centrum., 2022.

[25] Honzík, P.: Strojové učení. *Elektronická skripta VUT Brno*, 2006.

[26] Poznávanie, umelá inteligencia a strojové učenie [online]. Efocus [vid. 2022-02-03]. Dostupné z: `https://www.efocus.sk/images/uploads/strojove_ucenie.pdf`

[27] Raschka, S.: An overview of general performance metrics of binary classifier systems. arXiv 2014. *arXiv preprint arXiv:1410.5330*.

[28] Brom, O.: Rozhodovací stromy – pomocník při hledání předpovědi [online]. Acrea [vid. 2022-04-03]. Dostupné z: `https://acrea.cz/rozhodovaci-stromy/`

[29] Tangirala, S.: Evaluating the impact of GINI index and information gain on classification using decision tree classifier algorithm. *International Journal of Advanced Computer Science and Applications*, ročník 11, č. 2, 2020: s. 612–619.

[30] Eštok, P.: Dolovanie znalostí a strojové učenie [online]. FEI TUKE [vid. 2022-04-03]. Dostupné z: `https://magazin.kpi.fei.tuke.sk/2018/05/dolovanie-znalosti-a-strojove-ucenie/`

[31] Geurts, P.; Ernst, D.; Wehenkel, L.: Extremely randomized trees. *Machine learning*, ročník 63, č. 1, 2006: s. 3–42.

[32] Soukup, D.; Tisovčík, P.; Hynek, K.; aj.: Towards Evaluating Quality of Datasets for Network Traffic Domain. In *2021 17th International Conference on Network and Service Management (CNSM)*, 2021, s. 264–268, doi:10.23919/CNSM52442.2021.9615601.

[33] Ying, X.: An Overview of Overfitting and its Solutions. *Journal of Physics: Conference Series*, ročník 1168, feb 2019: str. 022022, doi:10.1088/1742-6596/1168/2/022022. Dostupné z: `https://doi.org/10.1088/1742-6596/1168/2/022022`

[34] CESNET: Anonymizer [online]. GitHub. [vid. 2021-08-09]. Dostupné z: `https://github.com/CESNET/Nemea-Modules/tree/c087d9a63f8feb0023e9f6400400450d2474f725/anonymizer`

[35] Daniel, U.: *Detekce IoT malware v počítačových sítích*. Diplomová práce, České vysoké učení technické v Praze. Vypočetní a informační centrum., 2021.

[36] Lashkari, A. H.: CICFlowMeter [online]. GitHub. [vid. 2021-08-10]. Dostupné z: https://github.com/ahlashkari/CICFlowMeter/blob/master/ReadMe.txt

[37] Boutaba, R.; Salahuddin, M. A.; Limam, N.; aj.: A comprehensive survey on machine learning for networking: evolution, applications and research opportunities. *Journal of Internet Services and Applications*, ročník 9, č. 1, 2018: s. 1–99.

[38] Barredo Arrieta, A.; DĂaz-RodrĂguez, N.; Del Ser, J.; aj.: Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, ročník 58, 2020: s. 82–115, ISSN 1566-2535, doi:https://doi.org/10.1016/j.inffus.2019.12.012. Dostupné z: https://www.sciencedirect.com/science/article/pii/S1566253519308103

[39] Cohen, I.: Explainable AI (XAI) with a Decision Tree [online]. Towardsdatascience [vid. 2022-04-03]. Dostupné z: https://towardsdatascience.com/explainable-ai-xai-with-a-decision-tree-960d60b240bd

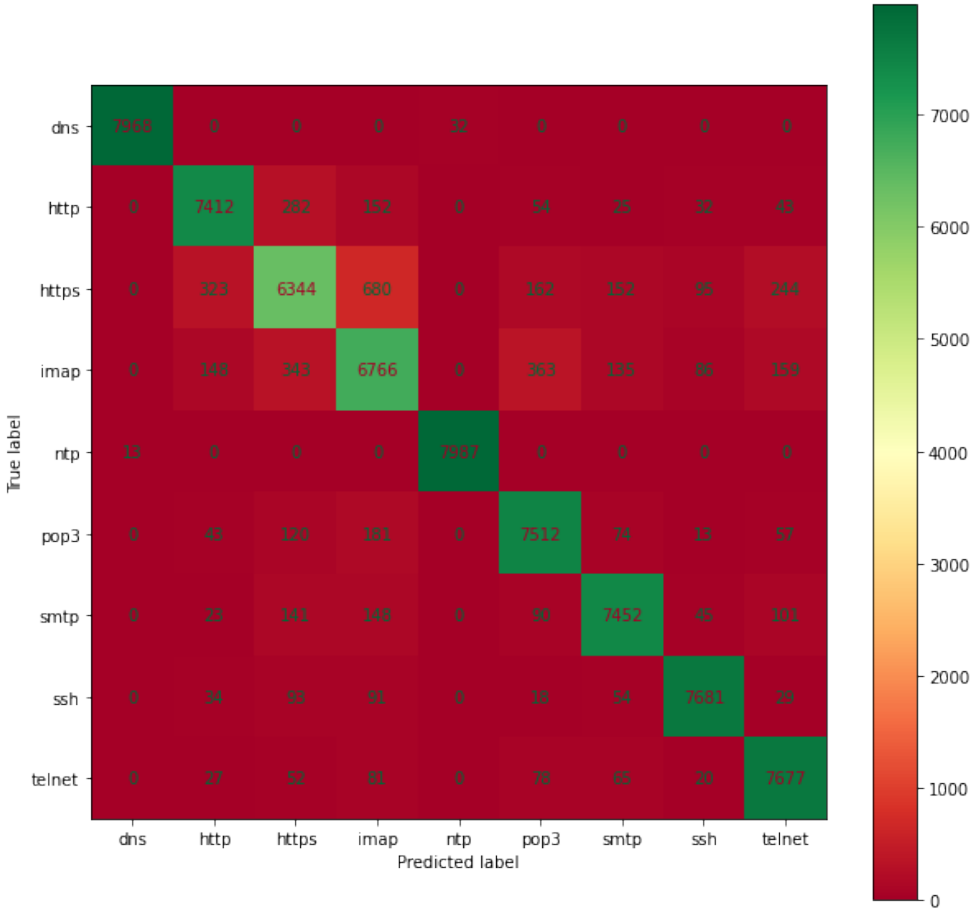
[40] Machine Learning in Python [online]. Scikit-learn [vid. 2022-02-03]. Dostupné z: https://scikit-learn.org/stable/index.html

[41] Nature Inspired Technologies Group: Testování modelů a jejich výsledků [online]. ČVUT. [vid. 2020-05-01]. Dostupné z: https://cw.fel.cvut.cz/old/_media/courses/a6m33dvz/03-testovanimodelu.pdf

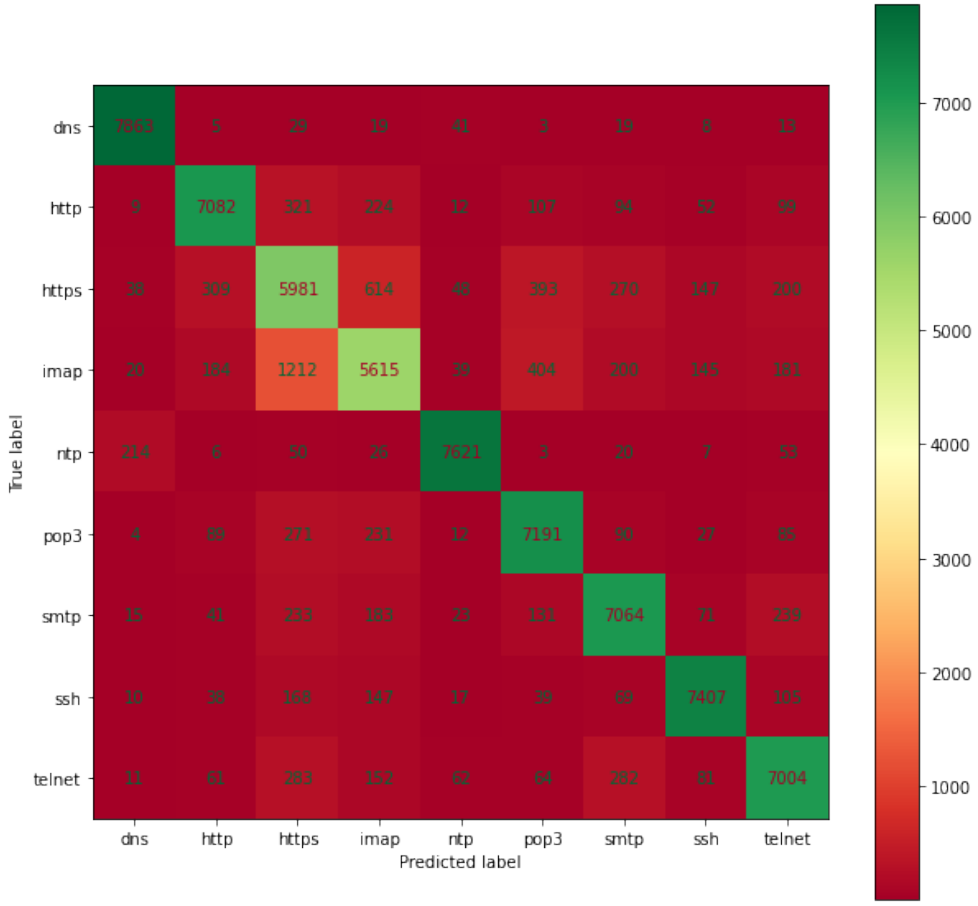
[42] Choosing the right estimator [online]. Scikit-learn [vid. 2022-02-03]. Dostupné z: https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

Dodatok A

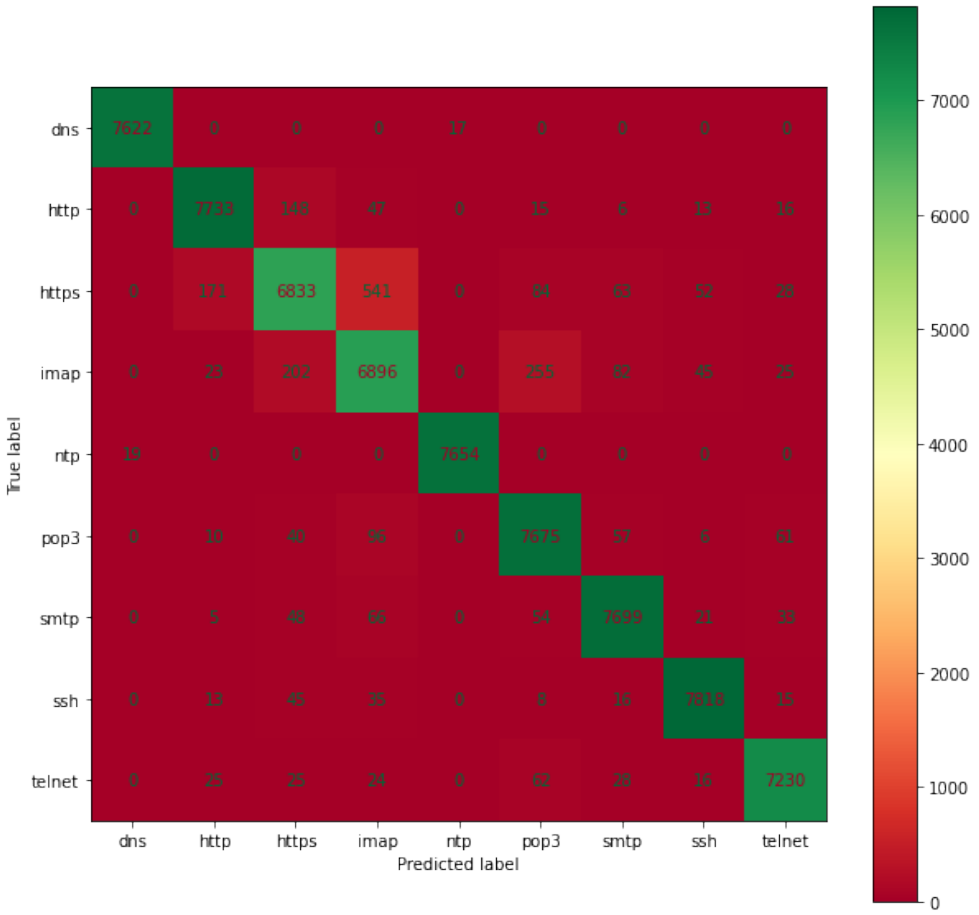
Obrázky



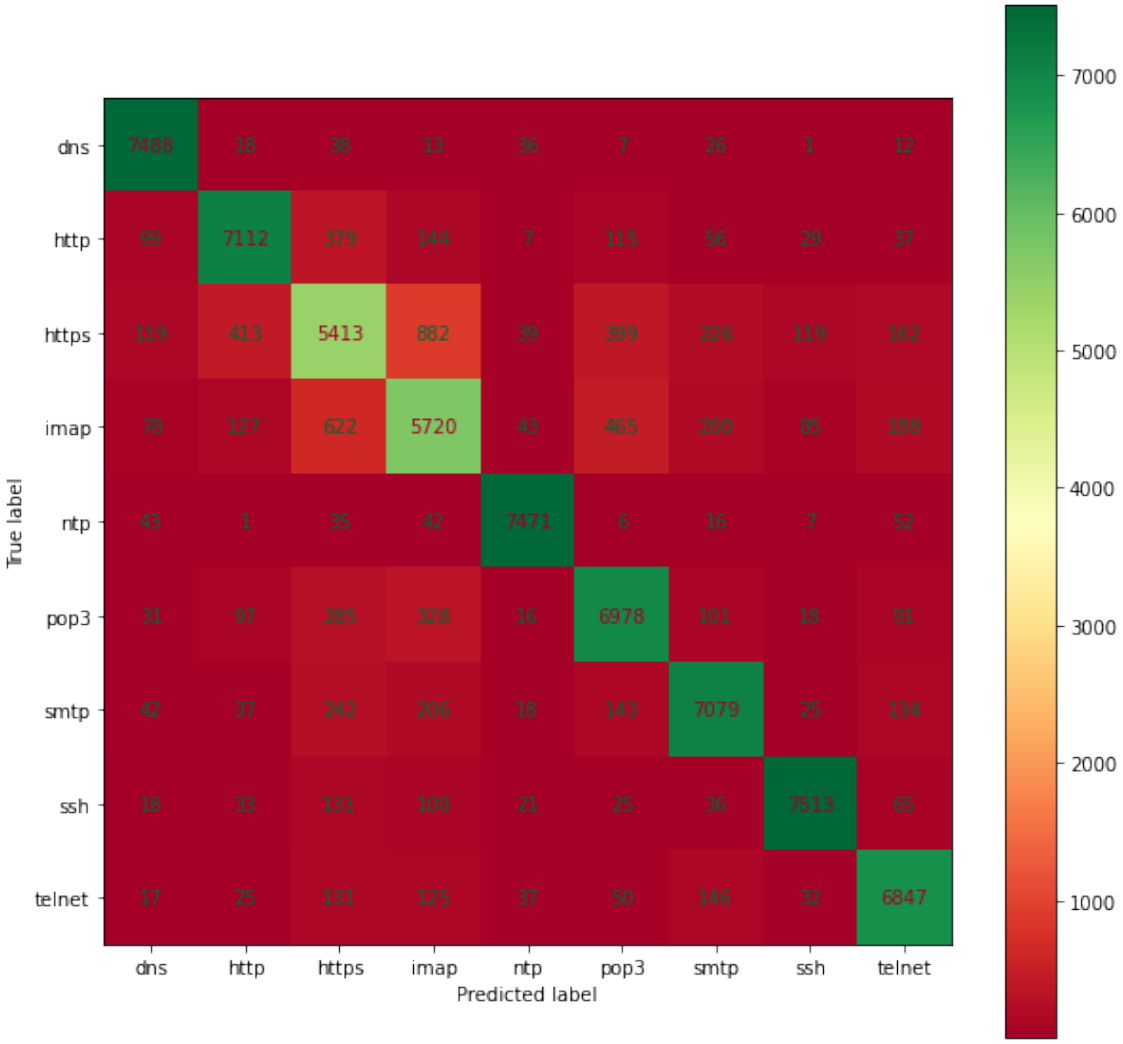
Obr. A.1: Matica zámeny klasifikátoru DT pri použití dátovej sady *BiFlow*



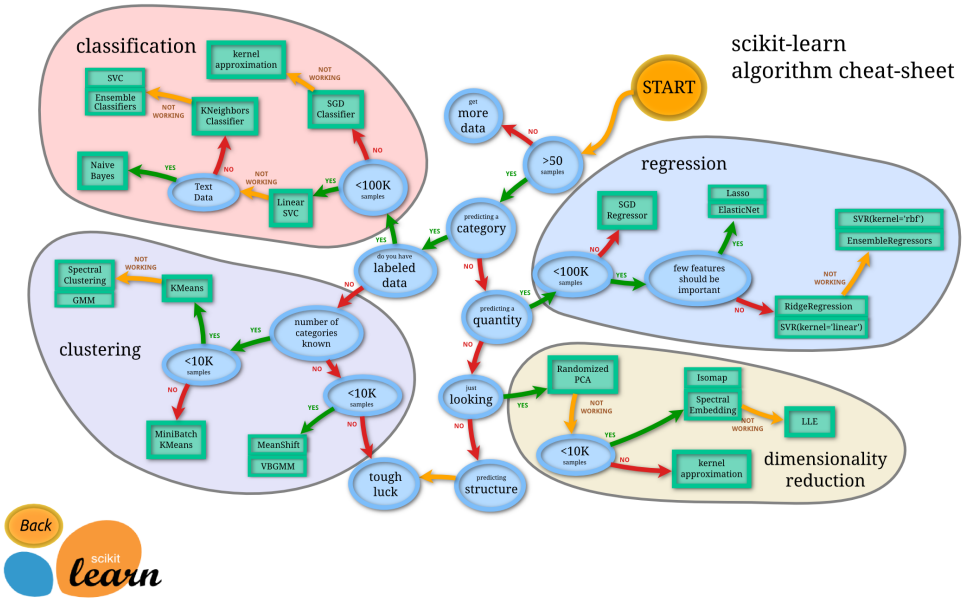
Obr. A.2: Matica zámeny klasifikátoru KNN pri použití dátovej sady *BiFlow*



Obr. A.3: Matica zámeny klasifikátoru DT pri použití dátovej sady *Rozšírené BiFlow*



Obr. A.4: Matica zámeny klasifikátoru KNN pri použití dátovej sady *Rozšírené BiFlow*



Obr. A.5: Mapa použitia metód strojového učenia [42]

Dodatok B

Zoznam použitých skratiek

ACC Accuracy

ACK Acknowledgement

ADiCT Asset Discovery, Classification and Tagging

BPP Bytes per packet

CESNET Czech Educational and Scientific NETwork

CSV Comma-separated values

ČVUT České vysoké učení technické v Praze

DNS Domain Name System

DoS Denial of Service

DDoS Distributed Denial of Service

DT Decision tree

FDS FDS file format

FET Feature Exploration Toolkit

FIT Fakulta informačních technologií

Gbps Gigabit per second

GB Gigabajt

GPL General Public License

GUI Graphical user interface

HTTP Hypertext Transfer Protocol

HTTPS Secure Hypertext Transfer Protocol

IANA Internet Assigned Numbers Authority

IETF Internet Engineering Task Force

IFC Interface

IMAP Internet Message Access Protocol

IP Internet protocol

IPFIX IP Flow Information Export

JSON JavaScript Object Notation

kB kilobajt

KNN k-nearest neighbors

LGPL Lesser General Public License

MAC Media Access Control

Mbps Megabit per second

MLP Multilayer perceptron

NEMEA Network Measurements Analysis

NTP Network Time Protocol

PCA Principal component analysis

PRE Presision

POP3 Post Office Protocol 3

PSH Push

RAM Random Access Memory

REC Recall

RFC Request For Comments

SMTP Simple Mail Transfer Protocol

SSH Secure Shell

SVM Support vector machine

SYN Synchronization

TCP Transmission Control Protocol

TELNET Network Virtual Terminal Protocol

TRAP Traffic Analysis Platform

UniRec Unified Record

UNIX Uniplexed Information and Computer Systems

Dodatok C

Obsah priloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `src`
  - `classifier` ..... zdrojové kódy klasifikačného modulu
  - `research` ..... Jupyter notebooky použité na experimenty
  - `thesis` ..... zdrojová forma práce vo formáte LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce vo formáte PDF