TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Analýza dat ze systémů pro chytrou domácnost

Diplomová práce

Studijní program: N2612 – Elektrotechnika a informatika

Studijní obor: 1802T007 – Informační technologie

Autor práce: **Bc. Vladimír Nevyhoštěný**

Vedoucí práce: doc. Ing. Otto Severýn, Ph.D.

Liberec 2020

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Zadání diplomové práce

Analýza dat ze systémů pro chytrou domácnost

Jméno a příjmení: **Bc. Vladimír Nevyhoštěný**
Osobní číslo: M17000134
Studijní program: N2612 Elektrotechnika a informatika
Studijní obor: Informační technologie
Zadávající katedra: Ústav mechatroniky a technické informatiky
Akademický rok: **2019/2020**

Zásady pro vypracování:

1. Seznamte se s architekturou a funkcí zabezpečovacích systémů a systémů pro chytrou domácnost firmy Jablotron a daty těmito systémy produkovanými.
2. Proveďte analýzu reálných anonymizovaných dat systémů firmy Jablotron nástroji data-miningu (např. hledání vzorů chování uživatelů, vyhodnocení falešných poplachů atp.).
3. Na základě výsledků analýz bodu 2. navrhněte možná zlepšení systémů (např. optimalizace vytápění, automatizace rutinních činností atp.)

Rozsah grafických prací: dle potřeby dokumentace
Rozsah pracovní zprávy: 40–50 stran
Forma zpracování práce: tištěná/elektronická
Jazyk práce: Čeština

Seznam odborné literatury:

[1] GOODFELLOW, Ian, Yoshua BENGIO a Aaron COURVILLE. *Deep learning*. Cambridge, Massachusetts: The MIT Press, [2016]. ISBN 978-026-2035-613.
[2] SILVER, Nate. The signal and the noise: why so many predictions fail–but some don't. New York: Penguin Press, 2012. ISBN 978-159-4204-111.
[3] Firemní dokumentace systémů firmy Jablotron a.s.

Vedoucí práce: doc. Ing. Otto Severýn, Ph.D.
Ústav mechatroniky a technické informatiky

Datum zadání práce: 10. října 2019
Předpokládaný termín odevzdání: 18. května 2020

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

doc. Ing. Milan Kolář, CSc.
vedoucí ústavu

V Liberci dne 10. října 2019

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu TUL.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Diplomovou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Současně čestně prohlašuji, že texty tištěné verze práce a elektronické verze práce vložené do IS STAG se shodují.

28. 5. 2020 Bc. Vladimír Nevyhoštěný

Analýza dat ze systémů pro chytrou domácnost

Abstrakt

Tato diplomová práce se zabývá analýzou dat produkovaných systémem pro zabezpečení a chytrou domácnosti společnosti Jablotron. V první části práce je popsána architektura systému a dat, které systém produkuje. Dále je popsán způsob anonymizace dat a jejich příprava pro následnou analýzu. V analytické části práce jsou navrženy a popsány metody pro detekci falešných poplachů, hledání vzorů v chování uživatele a následný návrh možné automatizace. U metod je popsán způsob jejich použití a vyhodnocení.

V rámci praktické části diplomové práce byl vytvořen R balíček **alarmanalysis**, kde jsou jednotlivé metody implementované.

Klíčová slova

Jazyk R, analýza dat, Jablotron, falešné poplachy, chování uživatele, návrh automatizace

Analysis of data produced by smart-home systems

Abstract

This diploma thesis deals with analysis of data produced by security and smart-home system of Jablotron company. The first part of this thesis is describes architecture of the system. Also there is description of produced data their anonymization. Then there is proposal of analysis method for detection of fake alarms, detection of pattern in user interactions and automatization suggestion. For each method is also described the usage and evaluation.

There is R package called **alarmanalysis** that has been created where are analysis methods implemented.

Keywords

Language R, data analysis, Jablotron, fake alarms, user interactions, automatization suggestion

Poděkování

Děkuji vedoucímu práce doc. Ing. Ottovi Severýnovi, Ph.D. za metodické vedení a jeho cenné rady během tvorby této práce. Rovněž děkuji Lukášovi Červenkovi ze společnosti Jablotron, který mi pomáhal s pochopením struktury celého systému.

Především bych chtěl poděkovat své přítelkyni a mamince, bez kterých bych tuto práci jen těžko zvládnul.

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

IoT	Internet of Things
CRAN	The Comprehensive R Archive Network
RDBS	Relační databázový systém
GDPR	General Data Protection Regulation
ONOOÚ	Obecné nařízení o ochraně osobních údajů
DPPC	Dohledové a poplachové přijímací centrum
IDE	Vývojové prostředí
CSV	Comma Separated Values
PDF	Portable Document Format

Úvod

V současnosti dochází k velkém rozmachu chytrých domácností a internetu věcí (IoT). Většina komponent těchto systémů, jako jsou například termostaty, čidla apod., generují data, která nesou zajímavé informace, avšak není pravidlem, že dochází k jejich využití. Přitom informace vydolované z těchto dat mohou vést k lepšímu pochopení, jak koncový uživatel systém používá, a výrobci tak pomoct kontinuálně systém optimalizovat.

Zdrojem dat analyzovaných v rámci této práce je systém **JABLOTRON 100**. Prvním cílem práce bylo seznámení se se systémem samotným a s daty, které produkuje. Systému **JABLOTRON 100** je věnována první kapitola, kde jsou popsané jeho základní vlastnosti a skupiny komponent, které jsou potenciálním zdrojem dat. Protože je nutné data před analýzou anonymizovat, bylo zapotřebí se seznámit s metodami anonymizace. Tomu je věnována druhá kapitola. Ve třetí kapitole jsou popsána data, která byla použita při analýze, a jakým způsobem byla anonymizována.

Dalším cílem bylo navrhnout a implementovat možné analýzy dat. První z nich je detekce falešných poplachů, tedy poplachů, které jsou způsobeny například špatným použitím systému či chybnou montáží. Další analýzou je hledání vzorů v chování uživatele. Velmi často se stává, že uživatel používá systém pro chytrou domácnost každý den velmi podobně. Například každé ráno před odchodem do práce uživatel vypne vytápění, stáhne rolety a zajistí dům. Analýza by měla tyto pravidelnosti odhalit a navrhnout příslušnou automatizaci. Poslední analýzou je dolování informací o spotřebě energií z dat naměřených pulzmetry. U každé metody je popsán způsob použití a její možný přínos. Implementace metod je zdokumentována v rámci páté kapitoly.

1 Systém pro chytrou domácnost Jablotron

Jablotron Group [1] je skupina technologických společností. Do této skupiny patří společnost **JABLOTRON ALARMS a.s.** [2], která se zabývá vývojem alarmů a systémů pro chytrou domácnost. Tyto systémy zahrnují řadu zařízení, která tvoří internet věcí (nebo také IoT). Vývoj a provoz celé IoT platformy je doménou společnosti **JABLOTRON CLOUD Services s.r.o.** [3]. V současnosti je v nabídce systém JABLOTRON 80 a JABLOTRON 100, kterému je věnována následující sekce.

1.1 JABLOTRON 100

V roce 2012 přinesla skupina Jablotron na trh systém JABLOTRON 100. Systém, který je evolucí staršího systému JABLOTRON 80, je revoluční v jednoduchosti svého ovládání a napojením na cloud. Právě díky interakci s cloudem je možné systém ovládat skrze mobilní či webovou aplikaci MyJABLOTRON. Neslouží pouze jako elektronický zabezpečovací systém, ale lze pomocí něj ovládat i další spotřebiče, jako třeba rolety či světla.

Základní komponentou celého systému je řídící jednotka, která se stará o chod systému a komunikaci s dalším příslušenstvím. Systém umožňuje připojit drátově i bezdrátově velkou škálu příslušenství (periferií), a proto budou dále popsána jen ta zařízení, která jsou schopna generovat data, jako například: detektory, kamery či termostaty.

Obrázek 1.1: Kompenenty systému JABLOTRON 100. [2]

1.1.1 PIR deketory pohybu

Pasivní infračervené čidlo (PIR čidlo) je běžně použito jako detektor pohybu a je základní komponentou zabezpečovacích systémů. Jablotron má v nabídce i detektory pohybu doplněné o další technologie, jako například kamerový modul pro verifikaci alarmu spuštěného pohybem. Pohybové detektory mohou zabírat celý prostor nebo mohou fungovat ve smyslu optické závory.

1.1.2 Plášťové detektory

Do této skupiny patří detektory, které detekují vniknutí do hlídaného objektu z venčí. Patří sem detektory rozbití oken, detektory pohybu s okny a dveřmi, ale i detektory pohybu a náklonu s vnějšími roletami.

1.1.3 Enviromentální detektory

Další skupinu tvoří detektory snímající různé veličiny prostředí, ve kterém se nacházejí. Mezi ně patří termostaty, detektory kouře a zvýšené teploty, detektory škodli-

vých plynů či záplavové čidlo.

Pulzmetry

Speciální skupinu tvoří pulzmetry umožňující napojení na různá další zařízení, která generují pulzy. Nejčastěji se pulzmetr používá pro sledování spotřeby elektrické energie po napojení na elektroměr.

1.1.4 Kamery

Poslední pomyslnou skupinou periferií jsou různé typy kamer. Veškerá doposud zmíněná zařízení produkují data charakteru časové řady. V určitý časový úsek se aktivuje detektor a vyvolá danou událost. Kamery navíc poskytují audiovizuální data, která lze právě v kombinaci s časovými řadami dále analyzovat a vyhodnocovat.

2 Anonymizace dat

Anonymizace dat je proces, při kterém dochází k nenávratnému odebrání informací, které by mohly vést k určení konkrétní osoby. Tento proces se používá, pokud subjekt, který shromažďuje data obsahující citlivé informace, potřebuje tyto data poskytnout třetí straně a nemá souhlas vlastníků citlivých informací. Příkladem je i tato diplomová práce, proto před použitím jakýchkoliv dat pro analýzu bylo nutné je nejprve anonymizovat.

2.1 Osobní údaj

V roce 2018, 25. května vešlo v platnost Obecné nařízení o ochraně osobních údajů [4] známé také pod zkratkou ONOOÚ či GDPR. Nařízení definuje pravidla pro uchovávání a zpracování osobních údajů. V českém znění článku 4, odstavci 1 je osobní údaj definován jako:

*„**Osobními údaji** veškeré informace o identifikované nebo identifikovatelné fyzické osobě (dále jen „subjekt údajů“); identifikovatelnou fyzickou osobou je fyzická osoba, kterou lze přímo či nepřímo identifikovat, zejména odkazem na určitý identifikátor, například jméno, identifikační číslo, lokační údaje, síťový identifikátor nebo na jeden či více zvláštních prvků fyzické, fyziologické, genetické, psychické, ekonomické, kulturní nebo společenské identity této fyzické osoby;“* [4] [čl. 4 odst. 1]

Osobní údaj lze tedy chápat jako informaci, která vede přímo, ale i nepřímo, k identifikování konkrétní fyzické osoby. Za osobní údaj lze považovat i zcela náhodné identifikační číslo, ke kterému je pořízení audiovizuální záznam, například při průchodu bránou s identifikační kartou.

Mezi osobní údaje patří například:

- jméno
- adresa
- věk
- pohlaví
- rodné číslo
- mzda
- IP adresa
- e-mailová adresa
- číslo občanského průkazu

Podmnožinu osobních údajů tvoří zvláštní nebo také citlivé osobní údaje. Jsou to údaje, které mohou subjekt poškodit ve společnosti či vést k projevům nenávisti. Až na výjimky je zakázáno tyto údaje zpracovávat.
Mezi citlivé osobní údaje patří například:

- rasový či etnický původ
- politický názor
- náboženské vyznání
- biometrické údaje
- genetické údaje

2.2 Pojmy správce, zpracovatel a příjemce

Je třeba definovat ještě další pojmy. **Správce** je v článku 4 odstavci 7 definován jako:

„fyzická nebo právnická osoba, orgán veřejné moci, agentura nebo jiný subjekt, který sám nebo společně s jinými určuje účely a prostředky zpracování osobních údajů.“ [4] [čl. 4 odst. 7]

Zpracovatel potom v témže článku odstavci 8 jako:

„fyzická nebo právnická osoba, orgán veřejné moci, agentura nebo jiný subjekt, který zpracovává osobní údaje pro správce.“ [4] [čl. 4 odst. 8]

A na závěr **příjmence** v odstavci 9 jako:

„fyzická nebo právnická osoba, orgán veřejné moci, agentura nebo jiný subjekt, kterým jsou osobní údaje poskytnuty, ať už se jedná o třetí stranu, či nikoli.“ [4] [čl. 4 odst. 9]

V případě této diplomové práce je správcem společnost **JABLOTRON ALARMS a.s.**, zpracovatelem **JABLOTRON CLOUD Services s.r.o.**. Moje osoba je příjemcem.

2.3 Metody anonymizace

Je definováno několik metod, pomocí kterých se anonymizují data. Každá metoda má svá specifika, a tak je nutné brát v úvahu jejich vlastnosti. Je vhodné uvážit i následné použití anonymizovaných dat, aby nedošlo k tomu, že po anonymizaci ztratí data potřebný význam. Tyto metody budou dále popsány a u každé metody bude názorná ukázka. Výchozí data pro ukázky mají strukturu tubulky, nicméně metody lze aplikovat i na další struktury (např.: grafy). Detailně jsou techniky anonymizace popsány ve **stanovisku č. 5/2014 k technikám anonymizace** [5].

Tyto anonymizační metody lze implementovat pomocí běžných programovacích jazyků. Existují i další možnosti, jak provést anonymizaci. Pokud uvažujeme, že data, která chceme anonymizovat jsou uložena v databázi, tak je možné provádět

anonymizaci už na úrovni dotazů RDBS. Některé databázové systémy mohou disponovat možností vytvářet pohledy, a tak lze vytvořit pohled, který bude obsahovat anonymizovaná data. Tyto anonymizované dotazy mohou mít však negativní dopad, z důvodu své složitosti na vytížení databázového systému.

Dále také existují softwarová řešení zaměřená na anonymizaci dat. Jedním z nich je **ARX Data Anonymization Tool** [6] v podobě aplikace s grafickým rozhraním či Java knihovnou. Jedná se o open source software a podporuje importování dat z *CSV* souborů, ale i relačních databází jako *MySQL*, *MS SQL* a *PostgreSQL*. Dalším nástrojem může být **sdcMicro** [7], což je R balíček určený pro anonymizaci dat. Balíček disponuje grafickým rozhraním vytvořeným pomocí technologie *Shiny*, které usnadňuje práci s tímto balíčkem.

2.3.1 Maskování dat

Maskování je jednoduchá a účinná metoda, jak anonymizovat data. Jedná se o nahrazení hodnoty univerzální maskovací hodnotou. V případě dat ve formě řetězce lze měnit hodnoty, které chceme skrýt maskovacím znakem, třeba # či *. Metodu však lze aplikovat i na vizuální data, kdy můžeme například rozmazat obraz v místě registrační značky snímaného automobilu. Rozsah maskování je třeba zvážit vzhledem k použití dat.

Původní data

Subjekt	**Datum narození**
Subjekt 1	09/02/1990
Subjekt 2	10/05/1995
Subjekt 3	24/12/1996

Anonymizovaná data

Subjekt	**Datum narození**
Subjekt 1	**/**/1990
Subjekt 2	**/**/1995
Subjekt 3	**/**/1996

Tabulka 2.1: V tomto případě dojde k zamaskování dne a měsíce v datumu, protože pro výsledné zpracovaní není tato informace podstatná. Je zde patrné, že je nutné znát kontext použití anonymizovaných dat.

2.3.2 Pseudoanonymizace dat

Jedná se o proces, kdy dojde k zaměnění jednoho údaje či skupiny údajů (jméno, příjmení, aj.), které by mohly vést k identifikaci subjektu. Bývají zaměněny zpravidla za hodnotu (pseudonym) bez přesného významu, jako například náhodné číslo nebo hash. Jedná se o velmi často využívanou metodu správci dat, protože je možné poskytnout zpracovatelům osobní údaje bez nutnosti další modifikace.

Pseudoanonymizace je narozdíl od běžné anonymizace vratný proces v momentě, kdy je znám způsob mapování identifikačních údajů na pseudonym. Mnohdy se totiž pseudoanonymizace provádí tak, že dojde k rozdělení identifikačních údajů a dalších citlivých údajů a identifikační údaje jsou nahrazeny pseudonymy. Pokud jsou známy oba soubory, je možné je spojit zpět a identifikovat tak subjekt. Pokud pro vytvoření pseudonymu byla použita některá hashovací funkce (např.: *md5*) a je opět známý soubor identifikačních údajů, je možné opět zpětně spojit citlivé a identifikační údaje. Tomu lze částečně zamezit použitím dostatečné soli na vstupu hashovací funkce.

Původní data

Jméno	**Příjmení**	**Rok narození**	**Váha**	**Výška**
Jan	Novák	1990	102	201
Adam	Krátký	1995	72	180
Jaroslav	Mikeš	1996	81	176

Anonymizovaná data

Subjekt	**Rok narození**	**Váha**	**Výška**
35663	1990	102	201
78593	1995	72	180
99905	1996	81	176

Tabulka 2.2: Ukázka pseudoanonymizace, kdy identifikační údaje *Jméno* a *Příjmení* byly nahrazeny za pseudonym náhodného čísla.

2.3.3 Generalizace dat

Generalizace funguje na principu odstranění konkrétních hodnot a nahrazením hodnot více obecných. Příkladem může být věk, kdy konkrétní rok je nahrazen rozmezím. Konkrétně může být hodnota věku *18 let* nahrazena za *10 až 20 let*. Generalizace může mít několik forem a úrovní, a tak můžeme věk *18 let* dále převést i na nečíselnou hodnotu *dospělost*.

18	15-20	10-30 /	10-60
23	20-25	dospělost	
43	40-45	40-60 /	
52	50-55	střední věk	

Tabulka 2.3: Ukázka generalizace věku.

Liberec	Okres Liberec	Liberecký kraj	ČR
Frýdlant			
Semily	Okres Semily		
Ústí nad Labem	Okres Ústí nad Labem	Ústecký kraj	

Tabulka 2.4: Ukázka geografické generalizace.

Anonymizovat data lze například i zaokrouhlením, což je vhodné například pro časové záznamy.

Původní data

Jméno	**Příjmení**	**Čas průchodu**
Jan	Novák	15:34:12
Adam	Krátký	15:45:01
Jaroslav	Mikeš	15:57:22

Anonymizovaná data

Jméno	**Příjmení**	**Čas průchodu**
Jan	Novák	15:30:00
Adam	Krátký	16:00:00
Jaroslav	Mikeš	16:00:00

Tabulka 2.5: Ukázka generalizace času zaokrouhlením. Časový záznam je zaokrouhlen na nejbližší celou hodinu nebo půlhodinu.

2.3.4 Anonymizace pomocí náhodných hodnot

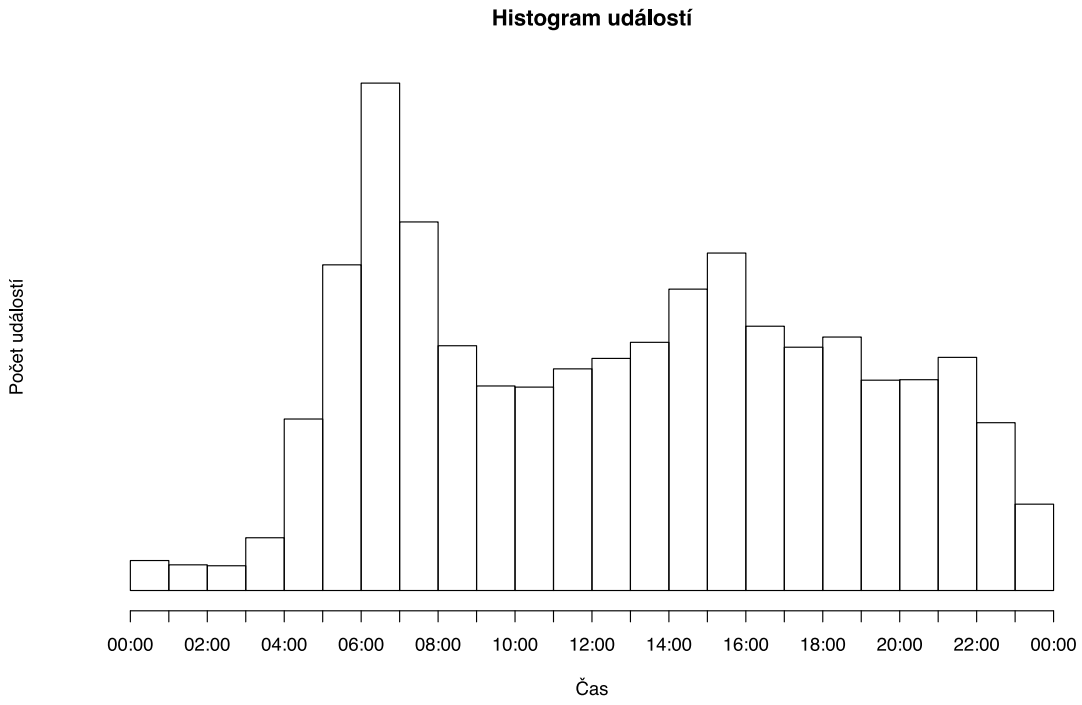
Další možností, jak změnit hodnotu citlivého osobního údaje je přiřadit mu jinou náhodnou hodnotu. Toho lze docílit několika způsoby. Například permutací jednotlivých znaků v řetězci, nebo dokonce permutací atributů v souboru dat. Tyto operace mají však za následek snížení srozumitelnosti a autenticity anonymizovaného souboru dat. Řešením toho problému může být ve využití některé z mnoha dostupných databází náhodných, avšak realisticky vypadajících dat. Jednou z nich je například **Mockaroo** [8], která disponuje náhodnými údaji o osobách, dopravě aj. V jazyce R lze využít balíčků jako **generator** [9] nebo **randNames** [10], které umí generovat sady náhodných dat.

3 Popis dat

V sekci JABLOTRON 100 je systém pro chytrou domácnost a jednotlivé komponenty tohoto systému. Protože je celý systém *IoT*, jsou komponenty zdrojem velkého množství dat. V této kapitole budou popsána data, která byla použita při řešení této práce. Data jsou uložena v relační databázi, nicméně pro tuto diplomovou práci jsou zdrojem dat pohledy nad tabulkami.

Celkových počet všech aktivních instalací systému, které produkují data do cloudu, se pohybuje okolo 230 tisíc instalací umístěných v desítkách zemí. Data nebyla nikterak časově omezená, nicméně byla použita data z let 2020, 2019 a 2018.

Zásadním, a z pohledu analýz, velmi zajímavým souborem dat je časová řada událostí. Tato časová řada zachycuje informace o událostech v rámci chytré domácnosti, jako například aktivace (zajištění) a deaktivace (odjištění) zabezpečení objektu či vyvolání poplachu. V čase vytváření této práce se počet záznamů událostí pohybuje v řádech miliard. Vzhledem k velkému objemu jsou data časové řady analyzována v částech (dny, týdny apod.). Případně se filtrují záznamy konkrétní lokace.



Obrázek 3.1: Histogram popisující frekvenci událostí v průběhu dne. Záměrně nejsou v histogramu uvedené konkrétní počty, ale je z něj zřejmé, že nejvíce událostí se odehrává mezi 6. a 8. hodinou ranní, kdy dochází pravidelně k zajištění či odjištění objektů. Naopak nejmenší aktivita je v nočních hodinách.

3.1 Pojmy v popisu dat

Při popisu dat jsou použity některé pojmy, které je třeba definovat.

- **Lokace** je konkrétní instalace systému JABLOTRON 100. Jedná se tedy například o konkrétní instanci chytré domácnosti.
- **Sekce** je konkrétní část lokace. Lokaci lze rozpadnout na více sekcí. Sekce může například představovat jedno patro.

3.2 Časová řada událostí

Časová řada událostí, které systém zaznamenává je základní sadou dat pro další výpočty. Každou událost lze rozlišit podle typu události. Dále má každá událost časovou známku, ve které se udála s rozlišením jedné sekundy. Data mohou být popsána následující tabulkou:

Název	**Typ**	**Popis**
Location	String	Identifikátor lokace
Section	String	Identifikátor sekce
Time	DateTime	Čas události
LocationTime	DateTime	Čas události lokace
Action	String	Typ události
Sender	String	Identifikátor entity, která událost vyvolala
SenderRole	String	Role entity

Tabulka 3.1: Datový model události zabezpečení

3.2.1 Události zabezpečení

Události týkající se zabezpečení mohou být následujícího typu (pole **Action**):

- **AlarmCanceled** - odvolání alarmu
- **Armed** - zabezpečeno
- **ArmedPartially** - částečně zabezpečeno (např.: pouze jedna sekce)
- **ArmedRemotely** - zabezpečeno vzdáleně (např.: skrze mobilní aplikaci)
- **ArmedPartiallyRemotely** - částečně zabezpečeno vzdáleně
- **Disarmed** - zabezpečení deaktivováno
- **DisarmedRemotely** - zabezpečení deaktivováno vzdáleně
- **InstantAlarm** - vyvolaný poplach

3.2.2 Události programovatelných výstupních obvodů

Centrální jednotka chytré domácnosti umožňuje připojení takzvaných programovatelných výstupních obvodů. Může se jednat o fyzické zařízení, zpravidla relé, které je spínáno dle nastavené logiky, jako například: jednotkový impuls, jednotkový skok, nastavení hodnoty po určitou dobu aj. Programovatelný výstup může být také pouze logický (virtuální) a na základě stavu systému vyvolat další akce.

Data mají stejnou strukturu jako 3.1. Události v časové řadě popisující programovatelné výstupní obvody jsou dvojího typu (pole **Action**): **PgOn** a **PgOff**. Konkrétní reakce na danou událost je otázkou implementace konkrétní lokace a není součástí dat. Díky tomu je systém poměrně variabilní.

Událost programovatelného výstupního obvodu je vždy vztažená ke konkrétní sekci. Z pohledu analýzy mají tedy význam kombinace sekcí a typ (PgOn, PgOff) v čase.

3.2.3 Anonymizace časové řady

Data časové řady byla anonymizována následujícím způsobem:

- **Location** - ačkoliv nemá z pohledu analýz hodnota příliš význam, protože je použita pouze pro rozlišení lokací, byla použita pseudoanonymizace přidělením náhodné adresy.
- **Section** - rovněž není hodnota důležitá z pohledu analýz, a tak byla anonymizována hashovací funkcí.
- **Time** - časová známka událostí je pro analýzu velmi důležitá. Jelikož má význam, jak přesný čas v rámci dne, ale i datum, nebylo možné použít například posunutí v čase a hodnota tedy nebyla anonymizována.
- **LocationTime** - Stejné jako u pole **Time**.
- **Action** - typ události nelze anonymizovat, aby nedošlo ke ztrátě informace.

- **Sender** - i zde je účel pole pouze rozlišení uživatelů. Pro ilustraci však byla použita anonymizaci předělením náhodného jména.
- **Section** - role uživatele nelze anonymizovat, aby nedošlo ke ztrátě informace.

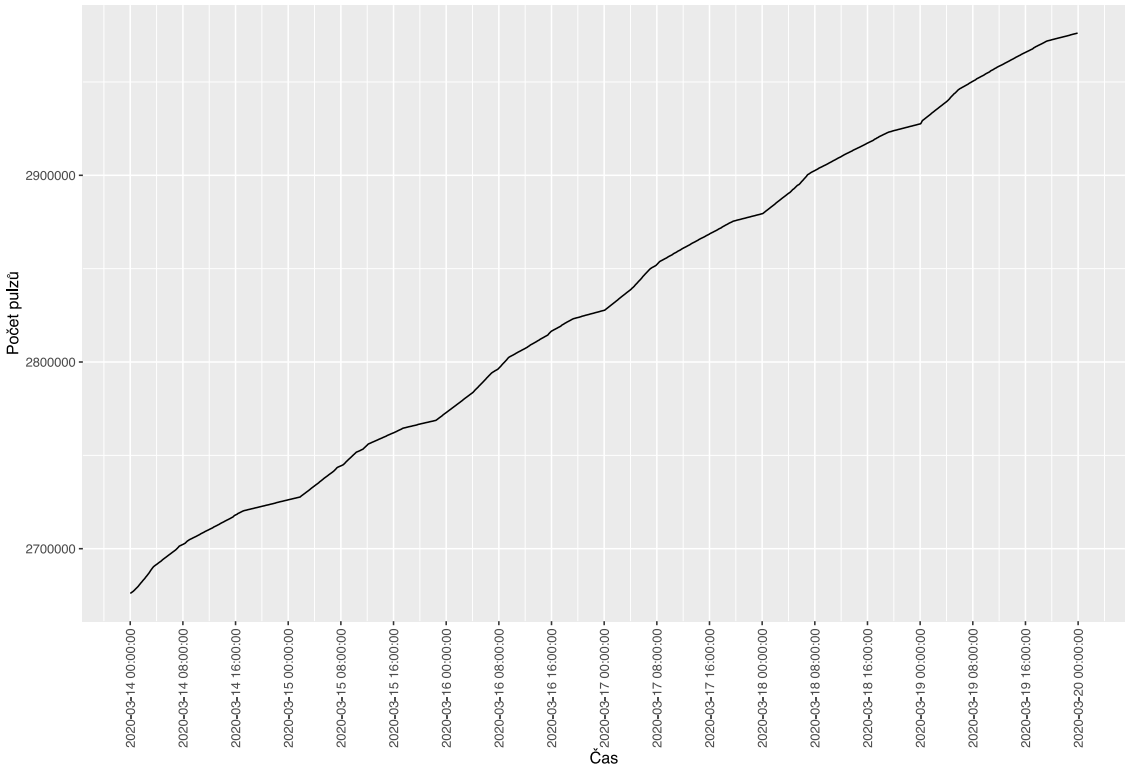
3.3 Časová řada hodnot pulzmetrů

Dalším souborem dat je časová řada hodnot naměřených pulzmetry. Data popisují chování systému, z pohledu spotřeby energií, jako například elektrická energie či vody. Pulzmetry fungují tak, že sčítají pulzy a v určitých intervalech součet odesílají do cloudu. Interval bývá zpravidla v řádech minut, nicméně nemusí být vždy konzistentní. Časovou řadu lze popsat následující tabulkou:

Název	**Typ**	**Popis**
Location	String	Identifikátor lokace
Time	DateTime	Čas události
ComponentId	Integer	Identifikátor pulzmetru
Input	Integer	Slouží pro rozlišení vysokého a nízkého tarifu
Value	Integer	Hodnota

Tabulka 3.2: Datový model události zabezpečení

Cloud součet pulzů z pulzmetrů zpracovává tak, že příchozí součet z pulzmetru přičte k poslednímu součtu. Hodnota pole **Value** má tak podobu rostoucí posloupnosti. Pole **Input** rozlišuje vysoký a nízký tarif, přičemž pulzmetr sčítá hodnoty pro oba tarify.



Obrázek 3.2: Vývoj pulzů v průběhu týdne. Na ose **x** je čas a osa **y** značí bezrozměrný součet pulzů. Na grafu je možné pozorovat zpomalení růstu v průběhu noci.

4 Analýza dat

V této kapitole budou popsány provedené analýzy na datech a jejich výsledky.

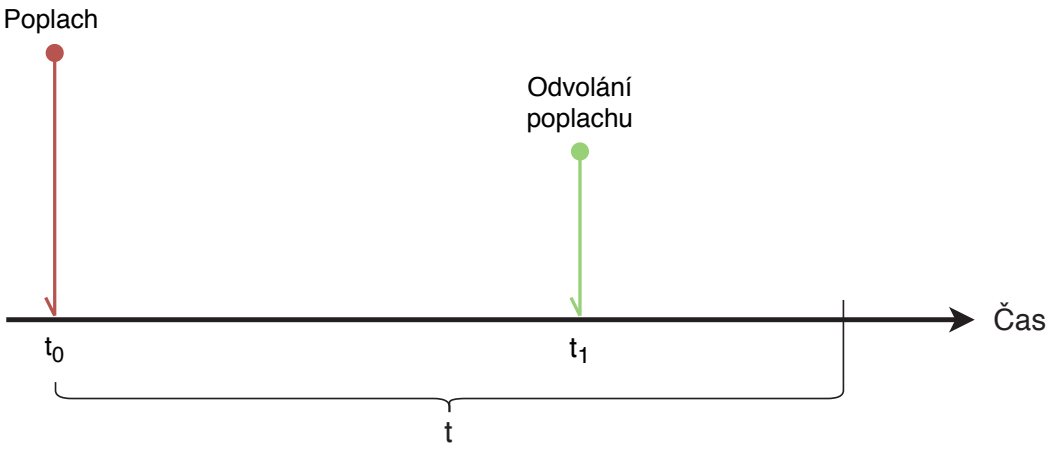
4.1 Detekce falešných poplachů

4.1.1 Popis problému

Základní a přirozenou funkcí každého zabezpečovacího systému je vyvolání poplachu v momentě, kdy dojde k narušení zabezpečeného objektu. Poplach může informovat uživatele či přímo kontaktovat pult centrální ochrany. Mnohdy se stává, že vyvolaný poplach je falešný, tedy takový poplach, při kterém reálně nehrozí žádné nebezpečí. Příčin vzniku falešného poplachu je několik. Může jej vyvolat samotný uživatel (vlastník objektu) špatnou manipulací, nebo poplach může způsobit špatná instalace systému, respektive komponent. Falešné poplachy jsou nepříjemné pro samotného uživatele, ale mohou způsobit i zbytečný výjezd zásahové jednotky pultu centrální ochrany.

4.1.2 Navržené řešení

V datech (viz 3.2) není ani částečně informace o tom, zda je poplach falešný či nikoliv, a tak úplně chybí data, na kterých lze učit nějaký algoritmus. Falešný poplach má tu vlastnost, že v brzké době po jeho vyvolání dojde uživatelem k jeho zrušení. Detekce je založena na tomto předpokladu.



Obrázek 4.1: Detekce falešného poplachu

Uvažujeme, že poplach nastane v čase t_0, odvolání poplachu pak v čase t_1, vstupní parametr t platí, že:

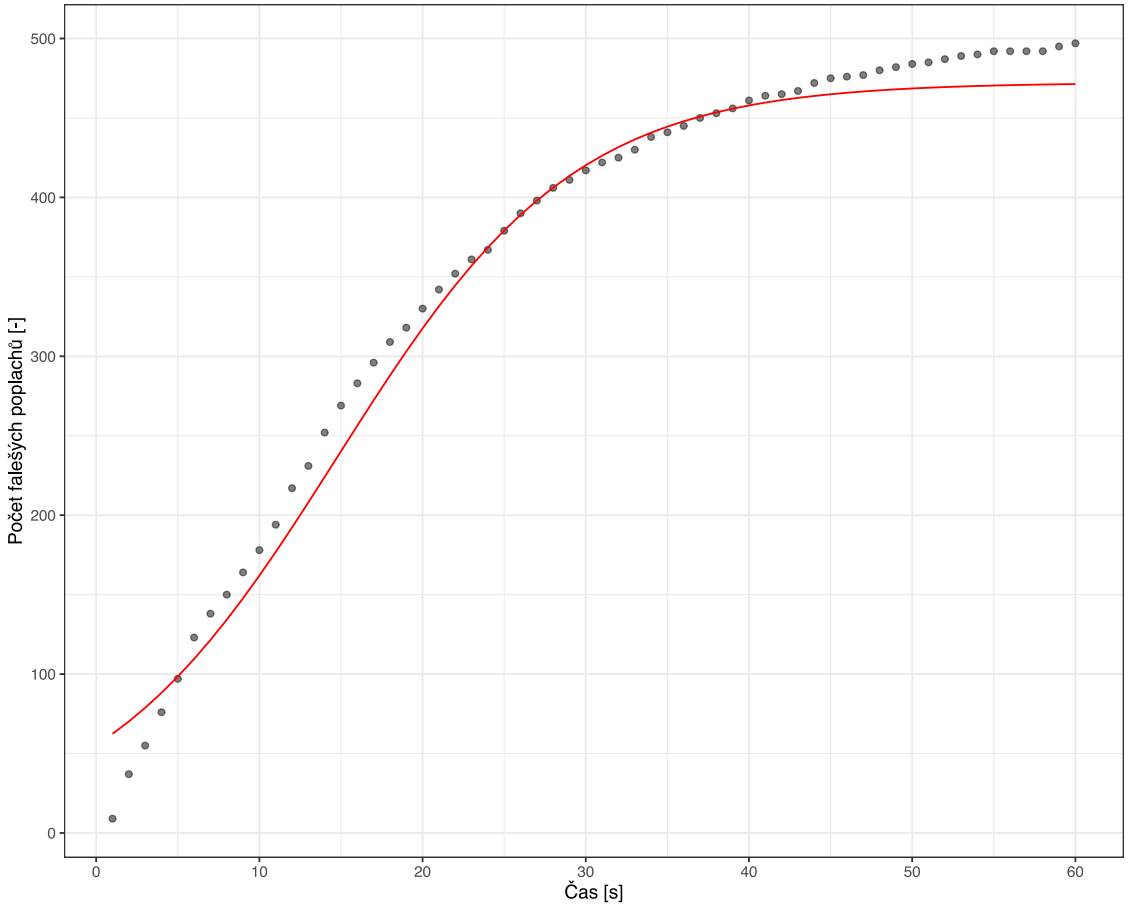
$$t, t_0, t_1 > 0 \tag{4.1}$$

Pak pokud

$$t \geq t_1 - t_0 \tag{4.2}$$

je poplach v čase t_0 považován za falešný.

Kromě samotných dat je jediným vstupním parametrem funkce parametr t. Dohledová a poplachová přijímací centra (dále jen DPPC) mají na základě normy **ČSN EN 50518-2** [11] povinnost zareagovat na tísňový poplach do *třiceti* sekund. Pokud je předpoklad, že falešný poplach je ten, kterým se ještě nezabývá DPPC, pak *třiceti* sekund může být výchozí hodnota parametru t. Toto tvrzení je možné také podložit grafem 4.2, kde je možné pozorovat, že při hodnotě t *třiceti* sekund dochází k zpomalení růstu počtu detekovaných poplachů. Při hodnotě t *šedesáti* sekund, už pak počet detekovaných falešných poplachů téměř neroste.



Obrázek 4.2: Vývoj počtu detekovaných falešných poplachů v závislosti na parametru t. Na datech byla provedena detekce falešných poplachů, přičemž parametr t postupně nabýval hodnot $\{1, 2, ..., 60\}$ (sekundy). Body znamenají počet falešných poplachů na hodnotě t a červená křivka představuje křivku růstu.

4.1.3 Vyhodnocení

Výsledkem výše popsané metody je tedy seznam poplachů, které jsou vyhodnocené jako falešné a seznam zbylých poplachů. Takto detekované falešné poplachy lze například použít pro servis. Detekce může identifikovat komponentu zabezpečovacího systému, která vyvolává falešné poplachy. Zvýšené množství falešných poplachů může znamenat závadu komponenty, ale také její špatnou instalaci. Případ užití může být teda takový, že montážní firma využije metodu, jako zpětnou analýzu systému a detekuje možné problémy. Vzhledem k tomu, že metoda po vyvolání poplachu če-

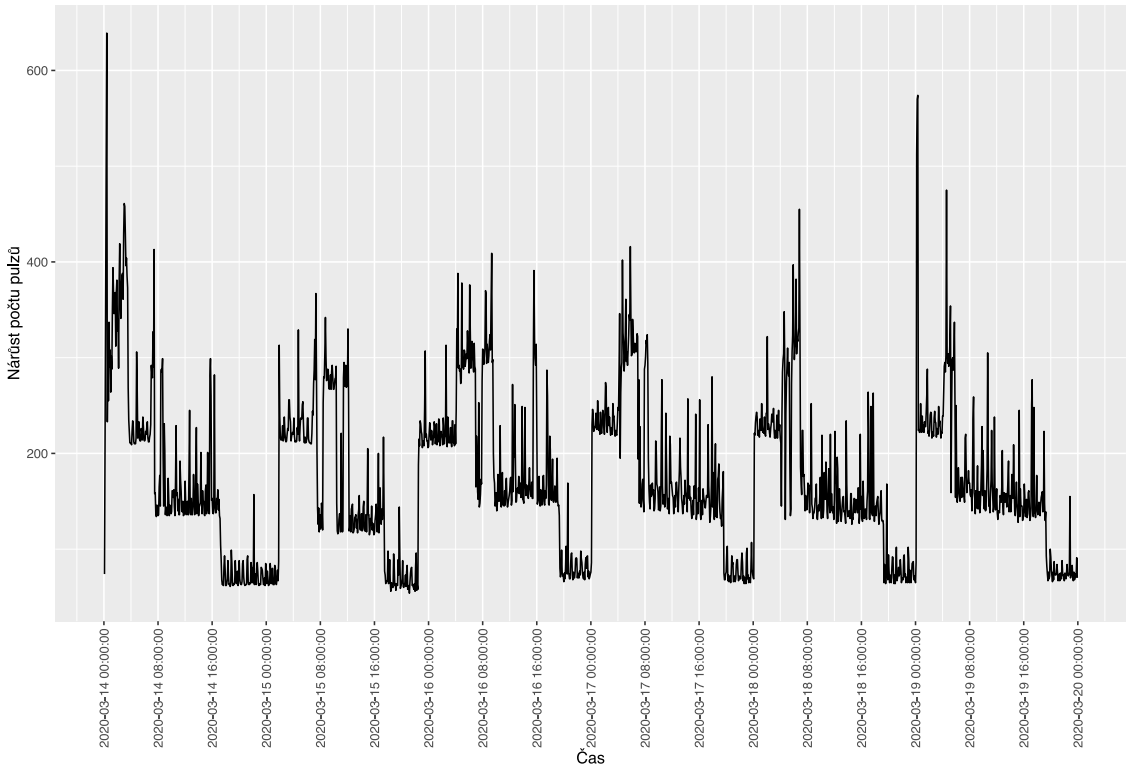
ká na jeho odvolání, nelze metodu použít pro detekci v reálném čase. Zároveň nelze u této metody zaručit 100% úspěšnost, a tak není vhodné, z podstaty problematiky zabezpečení objektů, metodu použít pro predikci.

Úspěšnost této metody lze určit tak, že se aplikuje na data, která obsahují poplachy, které jsou falešné. Úspěšnost pak bude rovna poměru celkovému počtu falešných poplachů k počtu těch detekovaných. Možným řešením je, vytvořit mechanismus, pomocí kterého by uživatel mohl zanést informaci, že právě vyvolaný poplach byl falešný. Pokud by tímto způsobem, alespoň část uživatelů spolupracovala, bylo by možné získat učící data, která by vedla k optimalizaci detekce.

4.1.4 Použití dat z pulzmetrů při detekci

Data z pulzmetrů mohou být použita pro zvýšení přesnosti detekce. Úvaha je taková, že pokud je detekován falešných poplach, mohla by být využita informace o spotřebě energií pro danou lokaci. A to tak, že nízká spotřeba může indikovat, že se nejedná o falešný poplach, jelikož v domě či bytě právě nikdo není. Opačně tomu bude v případě vysoké spotřeby.

Data z pulzmetrů poskytují informaci o součtech pulzů v čase, přičemž čím více pulzů, tím více bylo spotřebováno energie. Ačkoliv nelze exaktně určit aktuální spotřebu například v kWh, je možné vypočítat růst součtu pulzů a to následujícím vztahem: $r_1 = s_1 - s_0$, kde s_1 je součet pulzů v bodě 1 a s_0 součet v bodě předchozím, přičemž platí $s_1 \geq s_0$. Pak r_1 je nárůst v bodě 1.



Obrázek 4.3: Vývoj růstu součtů pulzů stejných dat jako v grafu 3.2.

4.2 Hledání vzorů v chování uživatele

4.2.1 Popis problému

U systémů pro chytrou domácnost je možné velmi často pozorovat, že uživatel systém ovládá pravidelně a podobně každý den. Může se například jednat o rodinný dům, kdy uživatel pravidelně odjíždí před 8. hodinou ranní do práce. Takový uživatel kromě zajištění objektu může například stahovat rolety a otevírat garážová vrata. Uživatelské návyky se mohou zároveň lišit i v závislosti na ročním období, kdy například v zimě spolu se zajištěním objektu dojde i k vypnutí vytápění. Pokud je možné najít v chování uživatele takový vzor, může mu být nabídnuta automatizace, která zefektivní jeho práci se systémem.

V rámci této analýzy je zapotřebí odhalit pravidelnost v chování uživatele a identifikovat posloupnost jednotlivých úkonů uživatele. Analýza musí zohlednit i drobné

odchylky v posloupnostech úkonů. Zároveň, vzhledem k velkému objemu dat je zapotřebí, aby bylo možné efektivně odlišit data, která jsou pro analýzu na první pohled nevhodná – nejsou pravidelná.

4.2.2 Navržené řešení

Navržené řešení spočívá v převedení jednotlivých událostí na dvourozměrné vektory, rozdělení událostí do sekvencí a určení podobnosti sekvencí. V této sekci je celý proces popsán.

Příprava dat

V první fázi analýzy je zapotřebí připravit vstupní data pro pozdější výpočty. Vstupní data jsou popsána v sekci 3.2.2. Jednotlivé kroky přípravy lze popsat následujícím diagramem:

Obrázek 4.4: Diagram procesu přípravy dat pro analýzu chování uživatele

Výběr informací - Ze vstupních dat jsou pro analýzu zapotřebí zejména následující informace (ostatní je možné pro analýzu zanedbat):

- **Time**
- **Section**
- **Action**
- **Sender**

Sjednocení událostí typu zajištění - Ačkoliv se v datech vyskytuje více typů událostí představujících zabezpečení (popsáno v sekci 3.2), z pohledu analýzy má význam rozlišovat pouze jeden typ zajištění/odjištění, a tak dojde v tomto kroku k jejich sloučení na jednotný typ **Armed**/**Disarmed**.

Vytvoření číselníku typu události a sekce - Pro další výpočty je potřeba vytvořit číselník kombinace hodnot **Section** a **Action**. Číselné hodnoty se přidělují tak, že pokud nabývá atribut **Action** hodnoty **Armed**, pak má rezervovanou hodnotu 1, ostatní hodnoty i, kde $i \in \mathbb{N} - \{1\}$.

Proces přípravy dat lze demonstrovat následujícím příkladem:

Location	**Action**	**Time**	**Sender**	**SenderRole**	**Section**
Location1	PgOff	2019-03-18 07:32:01	User1	User	Vytápění
Location1	PgOff	2019-03-18 07:32:03	User1	User	Vytápění
Location1	PgOn	2019-03-18 07:32:05	User1	User	Rolety venku
Location1	ArmedRemotelly	2019-03-18 07:38:05	User1	User	Dům
Location1	PgOn	2019-03-18 07:38:07	User1	User	Vrata

Tabulka 4.1: Názorný vzorek vstupních dat pro analýzu chování uživatele

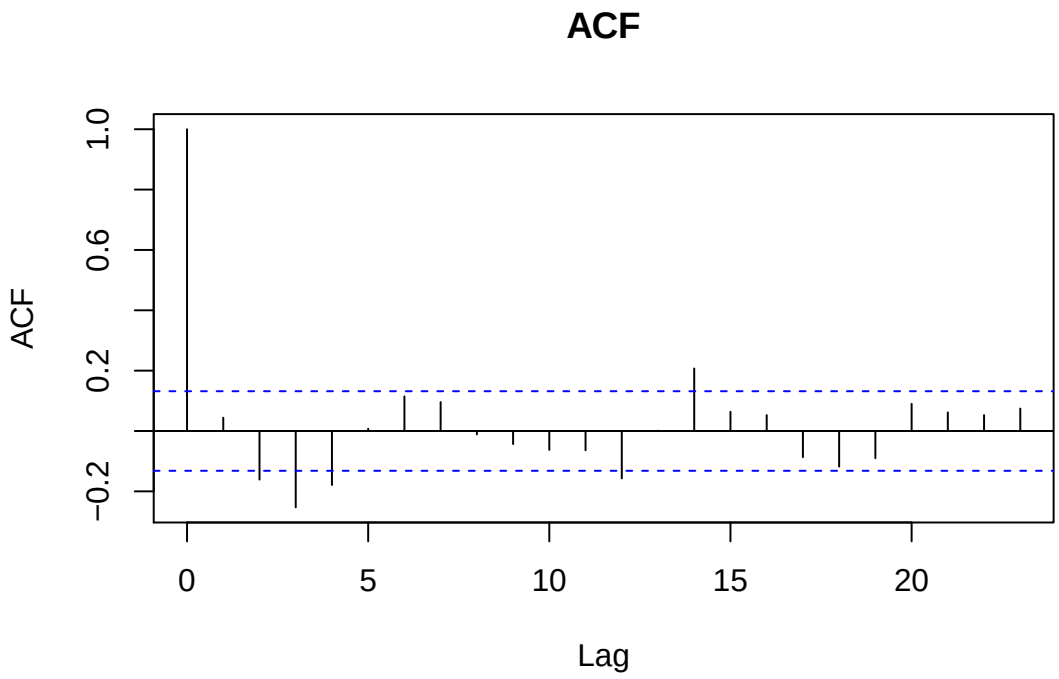
Data, která prošla popsanou přípravou:

Time	**Action**	**Section**	**Sender**	**action_section_number**
2019-03-18 07:32:01	PgOff	Vytápění	User1	2
2019-03-18 07:32:03	PgOff	Vytápění	User1	2
2019-03-18 07:32:05	PgOn	Rolety venku	User1	3
2019-03-18 07:38:05	Armed	Dům	User1	1
2019-03-18 07:38:07	PgOn	Vrata	User1	4

Tabulka 4.2: Připravený vzorek dat pro analýzu chování uživatele

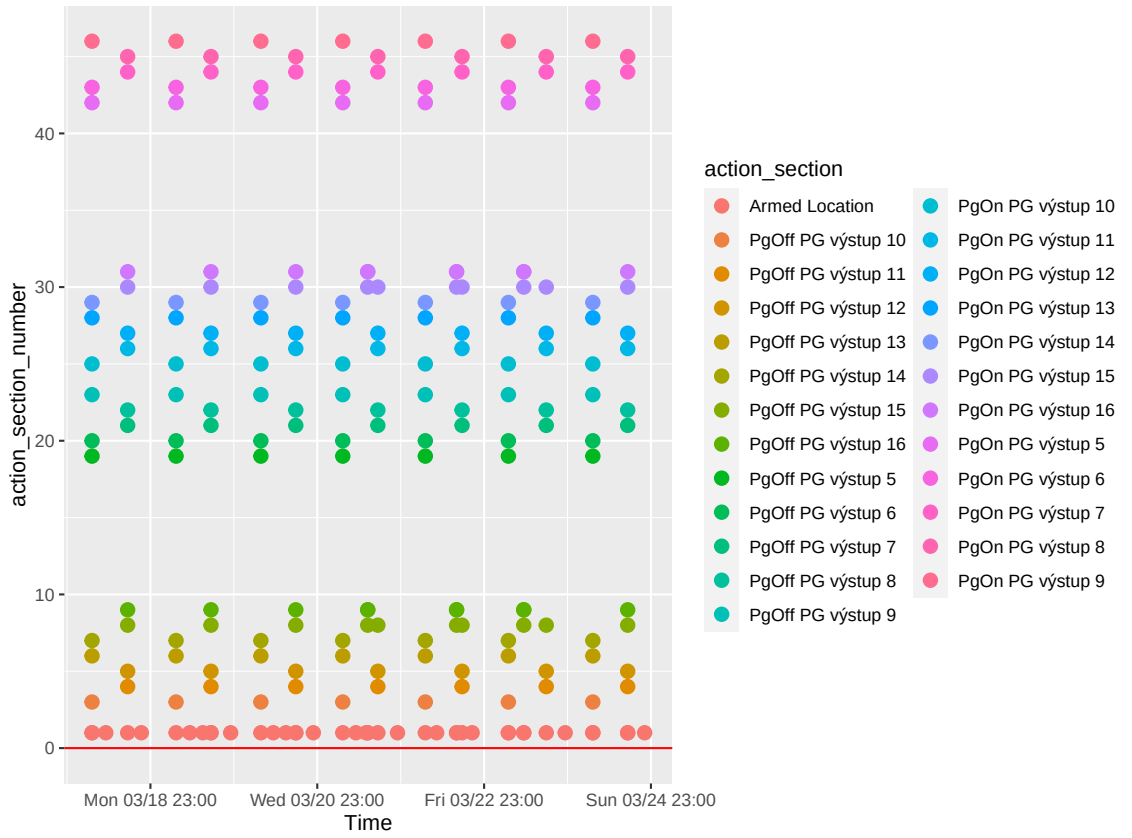
Hledání sekvencí událostí

Pokud jsou vstupní data připravená, je třeba ještě před samotnou analýzou efektivně rozhodnout, zda jsou data pro analýzu vhodná. K tomu je použita autokorelační funkce. Vstupem je dopočtený atribut *action_section_number*. Touto metodou lze zjistit, zdali se nachází v datech pravidelnost, nicméně není zohledněný čas, ve kterém se události udály, ale pouze jejich posloupnost.



Obrázek 4.5: Autokorelační funkce atributu *action_section_number*. Jedná se o úsek dat jednoho týdne v březnu 2019 konkrétní lokace.

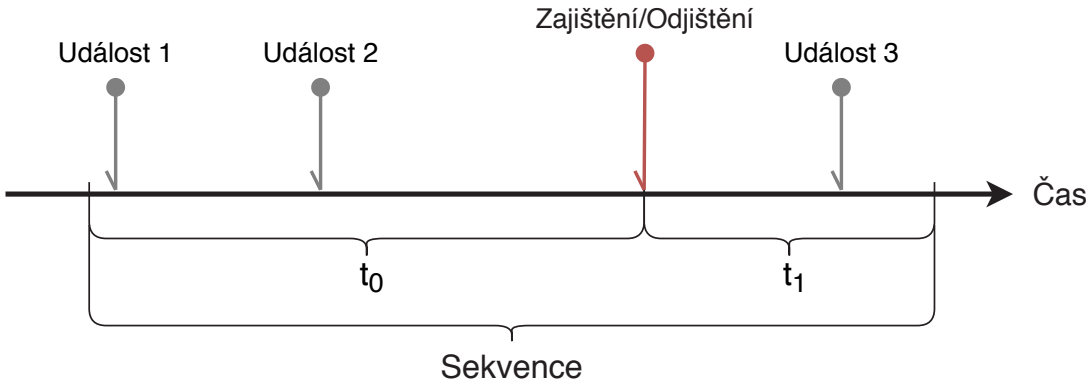
Dále je možné vhodným způsobem vizualizovat data a určit pravidelnost událostí. K tomu lze vyžít metodu `plot_events_time_line_around_armed`, která je součástí balíčku **alarma=nalysis**.



Obrázek 4.6: Vizualizace událostí metodou *plot_events_time_line_around_armed*. Použita byla stejná data jako v případě autokorelační funkce (viz obrázek 4.5). Na ose x se nachází čas a na ose y hodnota *action_section_number*. Díky tomuto grafu je možné pozorovat určitou pravidelnost událostí konaných v čase pro lokaci.

Z grafu je patrné, že jednotlivé události tvoří skupiny. Skupiny jsou tvořeny sekvencí událostí a velmi často obsahují událost zajištění lokace. Jedná se o vlastnost dat, která není specifická pouze pro tuto lokaci, ale je možné ji pozorovat u mnoho lokací. Vlastnost vyplývá z velmi častého případu užití systému pro chytrou domácnost, kdy uživatel odchází z domu, který zajistí a provede další navázané úkony, jako například vypnutí vytápění, stažení rolet apod. Analogicky to lze pozorovat i naopak při odjištění objektu.

Této vlastnosti bylo využito při analýze chování uživatele, a to tak, že jsou v čase nalezeny události zajištění (odjištění), pak události, které jsou v intervalu před respektive za, tvoří sekvenci událostí.



Obrázek 4.7: Rozdělení událostí do sekvencí. Vstupními parametry jsou t_0 a t_1.

Podobnost sekvencí

Na sekvence událostí lze nahlížet jako na n-rozměrné vektory. K výpočtu jejich podobnosti, respektive vzdáleností je použita implementace **Manhattanské metriky**. Protože v některých případech nehraje pořadí jednotlivých událostí roli, např. v momentě kdy uživatel před zajištěním objektu vypne vytápění a stáhne rolety, lze výpočet vzdáleností ovlivnit tak, že se ignoruje pořadí jednotlivých událostí.

Nelze očekávat, že uživatel bude ovládat systém vždy naprosto stejně, ale přesto ho může používat v rutinách. Analýza musí tedy zohlednit danou toleranci v odchylkách sekvencí a nahlížet na sekvence jako sobě podobné. Vzdálenosti sekvencí jsou popsány maticí $\mathrm{A} = (a_{i,j})$, kde i a j jsou sekvence. Dvě sekvence m a n jsou považovány za podobné pokud platí:

$$max(\mathrm{A}) \cdot \alpha \geq a_{m,n} \tag{4.3}$$

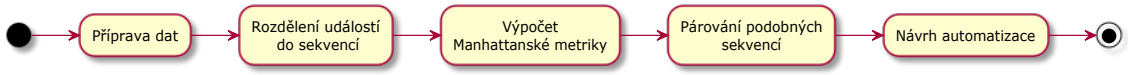
kde α je tolerance v procentech. Následně dojde k sloučení podobných sekvencí a výpočet četností výskytu, která je použita pro návrh automatizace.

Sekvence	1	2	3	5	6	8	9	11	12	13	15	16	18	20	21	23	25
1	0																
2	32	0															
3	1	31	0														
5	31	1	32	0													
6	1	31	0	32	0												
8	31	1	32	0	32	0											
9	1	31	0	32	0	32	0										
11	173	153	172	154	172	154	172	0									
12	62	40	61	41	61	41	61	113	0								
13	1	31	0	32	0	32	0	172	61	0							
15	173	153	172	154	172	154	172	0	113	172	0						
16	62	40	61	41	61	41	61	113	0	61	113	0					
18	0	32	1	31	1	31	1	173	62	1	173	62	0				
20	173	153	172	154	172	154	172	0	113	172	0	113	173	0			
21	62	40	61	41	61	41	61	113	0	61	113	0	62	113	0		
23	0	32	1	31	1	31	1	173	62	1	173	62	0	173	62	0	
25	31	1	32	0	32	0	32	154	41	32	154	41	31	154	41	31	0

Tabulka 4.3: Matice vzdáleností Manhattanské metriky, kde hodnota 0 značí absolutní shodu sekvencí a čím větší číslo, tím méně jsou si sekvence podobné.

4.2.3 Vyhodnocení

Navržený algoritmus pro hledání vzorů v chování uživatele lze popsat následujícím diagramem:



Obrázek 4.8: Diagram procesu navrženého algoritmu pro hledání vzorů v chování uživatele.

Výstupem tohoto algoritmu je seznam navržených automatizací. Například pro vstupní data popsaná grafem 4.6 a vstupními parametry $t_0 = 20min$, $t_1 = 5min$, $\alpha = 0.15$ a zanedbáním pořadí událostí byla navržena následující automatizace:

Události	**Počet výskytů**	**Poměr výskytů**
PgOn PG výstup 10,PgOn PG výstup 14,Armed Location, PgOn PG výstup 6,PgOff PG výstup 10,PgOff PG výstup 14, PgOff PG výstup 6,PgOff PG výstup 13,PgOff PG výstup 9, PgOff PG výstup 5,PgOn PG výstup 13,PgOn PG výstup 9, PgOn PG výstup 5	10	0.5882353

Tabulka 4.4: Navržená automatizace pro data z grafu 4.6.

Hlavním z případů užití algoritmu je, kdy jej budou využívat samotní uživatelé, kterým bude algoritmus navrhovat možné automatizace na základě používání, a tak jim ulehčit práci se systémem. Dále může výsledky použít přímo Jablotron, který pomocí analýzy toho, jak uživatel systém používá, může navrhovat zlepšení systému.

Jelikož výsledkem algoritmu je pouze návrh automatizace, je potřeba pro určení efektivity nasbírat nejprve informace o tom, kolik z navržených automatizacích bylo skutečně použito uživatelem.

5 R balíček alarmanalysis

Pro implementaci analýz popsaných výše byl použit jazyk R [12] a to především z důvodu jeho síly v oblasti statistických výpočtů. Součást praktické části této diplomové práce je R balíček, jenž je pojmenován **alarmanalysis** a obsahuje kromě implementace analýz i nástroje pro anonymizaci a přípravu dat. V následujících sekcích bude popsána standardizovaná struktura balíčku a zdokumentovány jednotlivé funkcionality balíčku.

5.1 Instalace

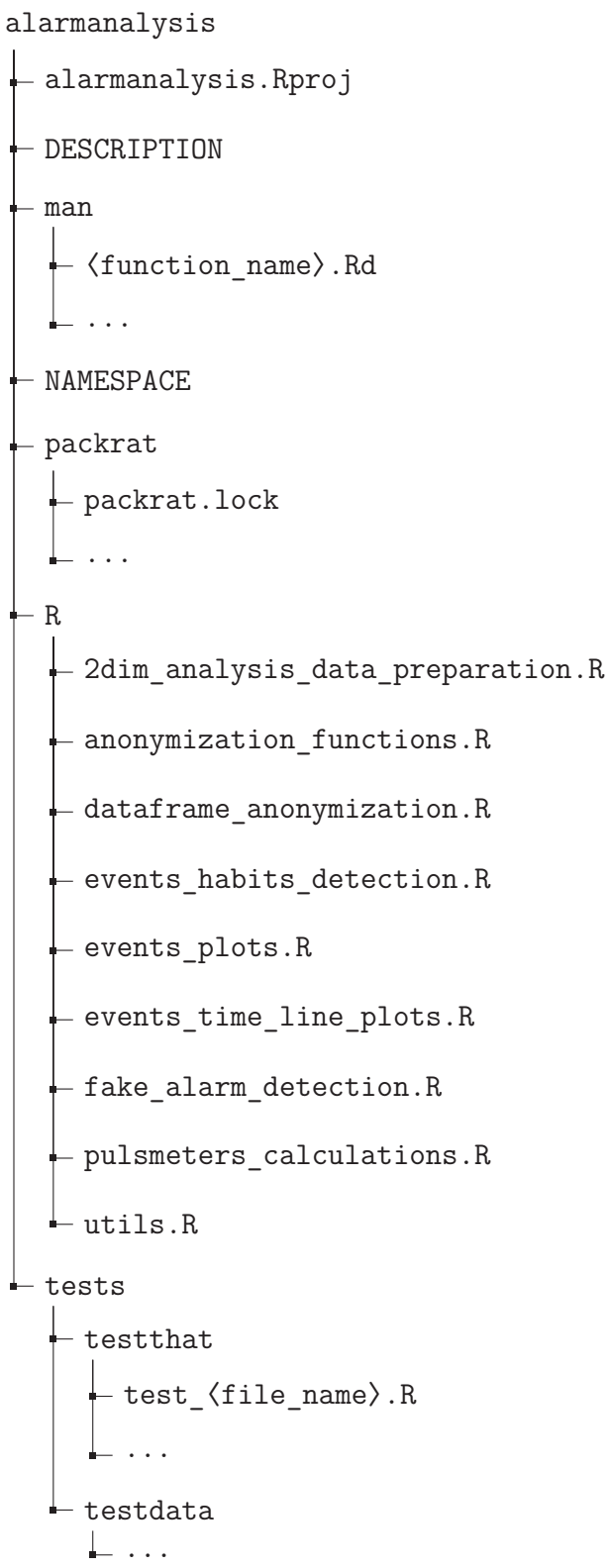
Pro instalaci a použití balíčku je zapotřebí instalace jazyka R. Balíček se nachází v **GitHub repositáři** [13] a klonovat jej lze příkazem:

```
$ git clone git@github.com:vlnevyhosteny/alarmanalysis.git
$ cd alarmanalysis/alarmanalysis
```

Nejjednodušší je instalace otevřením souboru `alarmanalysis.Rproj` v RStudiu [14], což je IDE pro vývoj v jazyce R. RStudio automaticky instaluje všechny potřebné závislosti. Alternativní postup instalace balíčku bez RStudia je popsán v souboru `README.md`.

5.2 Struktura balíčku

Balíček **AlarmAnalysis** má následující strukturu:


alarmanalysis
alarmanalysis.Rproj
DESCRIPTION
man
⟨function_name⟩.Rd
...
NAMESPACE
packrat
packrat.lock
...
R
2dim_analysis_data_preparation.R
anonymization_functions.R
dataframe_anonymization.R
events_habits_detection.R
events_plots.R
events_time_line_plots.R
fake_alarm_detection.R
pulsmeters_calculations.R
utils.R
tests
testthat
test_⟨file_name⟩.R
...
testdata
...

5.3 Metadata balíčku

Soubor `DESCRIPTION` udržuje důležité informace o balíčku jako takovém. Je zde například uvedena verze balíčku, informace o autorovi nebo definice závislostí, tedy další R balíčky. Pro správu závislostí balíčku je použit nástroj **Packrat** [15]. Izolované závislosti jsou umístěny v adresáři `packrat`. Důležitý je pak především soubor `packrat.lock`, který uchovává obraz závislostí a jejich verzí.

5.4 Dokumentace balíčku

Jelikož jazyk R nemá typovou kontrolu, je dokumentace nezbytně nutná pro vytváření kvalitního kódu. Pro vytvoření dokumentace balíčku byl použit nástroj **Roxygen2** [16]. Dokumentace je tvořena anotací nad každou funkcí. Kromě popisu vstupů a výstupů funkce se dále dokumentují i závislosti nutné pro chod funkce, popis funkce a příklady použití. Z anotace se generují **Rd** soubory, které jsou umístěny v adresáři `man`. Pro každou funkci existuje jeden soubor. Prohlížet dokumentaci konkrétní funkce lze následovně:

```
?prepare_data_for_2dim_analysis_armed
```

Z **Rd** souborů je možné generovat dokumentaci do dalších formátů, jako například **PDF** dokumentaci celého balíčku.

5.5 Testování balíčku

Balíček obsahuje soubor unit testů pro vybrané funkce balíčku. Pro testování byl použit nástroj **testthat** [17]. Testy jsou umístěny v adresáři `tests/testthat` a je možné je spustit interaktivně v RStudio pomocí `Cmd/Ctrl + Shift + T`, případně příkazem:

```
devtools::test()
```

5.6 Funkční prvky balíčku

Zdrojový kód balíčku je umístěn dle konvencí ve složce `R`. Jednotlivé funkce balíčku jsou rozděleny do souborů tak, jak spolu souvisí. Podrobná technická specifikace funkcí včetně příkladů použití se nachází v příloha A. Zde jsou uvedeny pouze základní vlastnosti funkcí.

5.6.1 Anonymizace dat

Balíček obsahuje několik funkcí pro anonymizaci vstupních dat. Do první pomyslné skupiny patří funkce, nacházející se v souboru `R/anonymization_functions.R`, které anonymizují vektor:

`hash_anonymization` - převádí hodnoty na hash pomocí zvoleného algoritmu

`random_integer_anonymization` - převádí hodnoty na náhodné hodnoty typu **integer**

`random_person_anonymization` - převádí hodnoty na náhodná lidská jména. Využívá databáze balíčku **generator** [9].

`random_location_anonymization` - převádí hodnoty na náhodné adresy. Využívá databáze balíčku **randNames** [10].

Pro anonymizaci celého **data.framu**[1] slouží funkce `dataframe_anonymization`, která je umístěna v souboru `R/dataframe_anonymization.R`. Kromě dat má funkce na vstupu anonymizační parametry, které jsou tvořeny funkcemi pro anonymizaci vektorů popsané výše. Pomocí funkce zároveň lze přejmenovat sloupce.

5.6.2 Detekce falešných poplachů

Funkce pro detekci falešných poplachů (viz sekce 4.1) se nachází v souboru `R/fake_alarm_detection.R`. Vstupem jsou data ve struktuře **data.frame**,

[1] **data.frame** je struktura v jazyce R, která má podobu tabulky. Hodí se pro data z relačních databází, **CSV** souborů apod. Sloupce jsou tvořeny vektory.

jenž musí obsahovat sloupce **Action**, **Location** a **Time**. Druhým vstupem je `time_span_in_seconds` s výchozí hodnotou **30**. Výstupem funkce je jmenný seznam, který obsahuje vektory `fake_alerts` a `true_alerts`, přičemž `fake_alerts` obsahuje ukazatele na události, které jsou detekovány jako falešný poplach. Analogicky `true_alerts` jako poplachy, které falešné nejsou. Ukazatelem je index řádku příslušné události ve vstupních datech. Příklad výstupu:

```
> alarmanalysis::detect_fake_alarm(data)
$fake_alerts
 [1]    83   610  1229  1645

$true_alerts
 [1]   102   618   619   747
```

5.6.3 Hledání vzorů v chování uživatele

Funkce pro hledání vzorů v chování uživatele (viz sekce 4.2) jsou rozděleny do několika souborů. V souboru `R/2dim_analysis_data_preparation.R` jsou funkce pro přípravu dat. Protože analýza hledá návyky uživatele navázané na události zajištění či odjištění lokace (v kódu označován jako **base** - nultý bod) je příprava dat rozdělena na následující dvě funkce:

`prepare_data_for_2dim_analysis_armed` - připravuje data pro následnou analýzu událostí navázaných na událost zajištění. Vstupní data mají strukturu **data.frame** a musí mít následující sloupce: **Action**, **Location**, **Time** a **Section**. Výstupem funkce je **data.frame** upravených dat. Funkce využívá balíčky **scales** [18] a **dplyr** [19].

`prepare_data_for_2dim_analysis_disarmed` - stejně připravuje data pro analýzu událostí navázaných na událost odjištění.

Soubor `R/events_habits_detection.R` obsahuje několik funkcí používaných pro analýzu:

`find_seq_around_base` - funkce, která rozděluje události v časové řadě do sekvencí. Na vstupu jsou kromě dat také parametry `t_before_minutes` a `t_after_minutes`, kterými se definuje interval od nultého bodu. Výstupem funkce je **data.frame** doplněný o číselný sloupec `seq`. Funkce používá balíček **flifo** [20].

`get_similarity_matrix` - funkce pro výpočet matice vzdáleností. Vstupem jsou data rozdělená do sekvencí a výstupem je matice o rozměrech $n \times n$, kde n je počet sekvencí.

`get_sequence_occurrence` - určuje počet výskytů sekvencí. Vstupem je matice vzdáleností a procentuální tolerance odlišnosti sekvencí `toleration`. Výstupem je **data.frame** sekvencí doplněný o počet výskytů.

`get_habits` - využívá všech tří předešlých funkcí a navrhuje automatizace pro uživatele. Vstupem jsou připravená data a všechny parametry výše popsaných funkcí. Výstupem je **data.frame** s návrhy automatizací.

6 Závěr

Práce byla zaměřena na analýzu dat generovaných systémy pro zabezpečení a chytrou domácnost společnosti Jablotron. Měla zmapovat, jaká data systém poskytuje, a jaké informace lze z nich pomocí analýz získat. Takto získané informace mají vést k lepšímu pochopení, jak uživatel systém používá, a také mají pomoct samotnému uživateli.

V první fázi práce bylo mým úkolem seznámit se s architekturou systému **JABLOTRON 100** a s daty, které poskytuje. To byl zároveň první cíl práce. Bylo nutné zjistit, jaké komponenty systém nabízí, a jaké jsou jeho vnitřní procesy. Velkou výzvou bylo zorientovat se ve struktuře dat samotných a zároveň odhadnout, jaké informace mohou poskytovat, při tak velkém objemu.

Před samotným analyzováním dat bylo také zapotřebí data anonymizovat. Musel jsem se tedy seznámit s metodami anonymizace dat a také nastudovat problematiku osobních údajů. Musel jsem prohloubit svoje znalosti nařízení GDPR, v čemž vidím pro mě velký přínos. Po výběru vhodných metod jsem vytvořil vlastní implementaci.

Druhým cílem práce bylo navrhnout možné analýzy dat. Tou první je detekce falešných poplachů. Určitou překážkou byla absence učících dat, tedy takových dat, ve kterých by byly falešné poplachy identifikované. Navržená detekce je tedy založena na vypozorovaných vlastnostech dat a také normě **ČSN EN 50518-2** [11] pro dohledová a poplachová přijímací centra. Detekce může pomoct především s odhalením komponent, které často vyvolávají falešné poplachy, což může být důsledkem například špatné montáže komponenty. Dále je také navržená interakce systému s uživatelem, kdy by po odvolání poplachu uživatel určil, zdali se jednalo o falešný poplach. Na základě těchto informací by bylo možné vyhodnotit přesnost metody

a její další optimalizaci. Zároveň by bylo možné využít některých z metod strojového učení, a tak detekci ještě více zpřesnit.

Další navrženou metodou je hledání vzorů v chování uživatele a následný návrh automatizace. Metoda rozděluje jednotlivé činnosti uživatele do sekvencí a následně porovnává podobnosti jednotlivých sekvencí. Podobné sekvence s častým výskytem, jsou navržené jako případná automatizace. Metoda může pomoct k lepšímu pochopení, jak uživatel systém používá, ale především pomůže samotnému uživateli.

Dále byla také navržena metoda, jak získat z dat naměřených pomocí pulzmetrů, informace o aktuální spotřebě energií (elektrická energie aj.) v systému. Informace o spotřebě může pomoct například při detekci falešných poplachů.

Popsané metody jsou implementovány v jazyce R a jsou součástí vytvořeného balíčku **alarmanalysis**. Rovněž byla vytvořena technická specifikace balíčku, která se nachází v příloze práce a uvnitř balíčku. Na přiloženém CD-ROM se nachází zdrojový kód balíčku, a také sada testovacích dat.

Logický krok, jak navázat na práci je integrovat navrhované metody do systému pro chytrou domácnost. Výpočty samotné by bylo možné integrovat do cloudu. Ovládat výpočty by bylo možné realizovat mobilní aplikací. Balíček tak může sloužit, jako předloha k integraci metod do systému.

Literatura

[1] *Jablotron Group* [online]. 2020 [cit. 2020-02-24]. Dostupné z: https://jablotrongroup.com/index

[2] *JABLOTRON ALARMS a.s.* [online]. 2020 [cit. 2020-02-24]. Dostupné z: https://www.jablotron.com/cz/

[3] *JABLOTRON CLOUD Services s.r.o.* [online]. 2020 [cit. 2020-02-24]. Dostupné z: https://www.jablotron.cloud

[4] *NAŘÍZENÍ EVROPSKÉHO PARLAMENTU A RADY (EU) 2016/679: o ochraně fyzických osob v souvislosti se zpracováním osobních údajů a o volném pohybu těchto údajů a o zrušení směrnice 95/46/ES (obecné nařízení o ochraně osobních údajů)*. In: . 2016, ročník 2016, 2016/679. Dostupné také z: https://eur-lex.europa.eu/legal-content/CS/TXT/HTML/?uri=CELEX:32016R0679&from=EN#d1e2233-1-1

[5] Stanovisko č. 5/2014 k technikám anonymizace. *European Commission: Data protection* [online]. 2014, 10. dubna 2014 [cit. 2020-02-24]. Dostupné z: https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_cs.pdf

[6] *ARX Data Anonymization Tool* [online]. 2020 [cit. 2020-02-24]. Dostupné z: https://arx.deidentifier.org

[7] *SdcMicro* [online]. 2020 [cit. 2020-02-24]. Dostupné z: http://sdctools.github.io/sdcMicro/articles/sdcMicro.html

[8] *Mockaroo: Random Data Generator and API* [online]. 2020 [cit. 2020-02-25]. Dostupné z: https://mockaroo.com

[9] Package 'randNames'. *CRAN* [online]. 2016 [cit. 2020-02-25]. Dostupné z: https://cran.r-project.org/web/packages/randNames/index.html

[10] Package 'generator'. *CRAN* [online]. 2015 [cit. 2020-02-25]. Dostupné z: https://cran.r-project.org/web/packages/generator/index.html

[11] ČSN EN 50518-2. *Dohledová a poplachová přijímací centra: Část 2: Technické požadavky*. 2. Česká republika: Asociace technických bezpečnostních služeb Grémium Alarm, o. s., 2014.

[12] R-Project. *The R Project for Statistical Computing* [online]. Auckland: The R Foundation, 1997 [cit. 2020-05-02]. Dostupné z: https://www.r-project.org

[13] NEVYHOŠTĚNÝ, Vladimír. *Alarmanalysis* [online]. GitHub, 2020 [cit. 2020-05-17]. Dostupné z: https://github.com/vlnevyhosteny/alarmanalysis

[14] *RStudio* [online]. 2016 [cit. 2016-05-07]. Dostupné z: https://www.rstudio.com

[15] *Packrat: Packrat is a dependency management system for R* [online]. 2018 [cit. 2020-05-17]. Dostupné z: https://github.com/rstudio/packrat/

[16] *Roxygen2: Generate your Rd documentation* [online]. 2020 [cit. 2020-05-17]. Dostupné z: https://github.com/r-lib/roxygen2

[17] *Testthat* [online]. 2020 [cit. 2020-05-17]. Dostupné z: https://github.com/r-lib/testthat/

[18] *Package 'scales'* [online]. CRAN, 2020 [cit. 2020-05-18]. Dostupné z: https://cran.r-project.org/package=scales

[19] *Package 'dplyr'* [online]. CRAN, 2020 [cit. 2020-05-18]. Dostupné z: https://cran.r-project.org/package=dplyr

[20] *Package 'flifo'* [online]. CRAN, 2018 [cit. 2020-05-18]. Dostupné z: https://cran.r-project.org/package=flifo

[21] SILVER, Nate. *The signal and the noise: why so many predictions fail--but some don't*. New York: Penguin Press, 2012. ISBN 978-159-4204-111.

[22] GOODFELLOW, Ian, Yoshua BENGIO a Aaron COURVILLE. *Deep learning*. 1. Cambridge, Massachusetts: The MIT Press, [2016]. ISBN 978-026-2035-613.

Příloha A - Technická specifikace balíčku alarmanalysis

Package 'alarmanalysis'

May 20, 2020

Type Package

Title Analysing data produced by physical alarms

Version 0.0.1

Description Analysing data produced by physical alarms.

License MIT

Encoding UTF-8

Imports digest,
generator,
randNames,
graphics,
utils,
lubridate,
dplyr,
ggplot2,
scales,
flifo,
matlib

Suggests knitr,
roxygen2,
testthat,
devtools,

LazyData true

RoxygenNote 7.1.0

R topics documented:

`add_to_date` *Add given values to POSIXct date.*

Description

Add given values to POSIXct date.

Usage

```
add_to_date(date, days = 0, hours = 0, minutes = 0)
```

Arguments

`date`	given date
`days`	to be added
`hours`	to be added
`minutes`	to be added

Value

date with added values

Examples

```
## Not run:
alarmanalysis::add_to_date(date, days = 2, hours = 3, minutes = 1)

## End(Not run)
```

`calculate_grow_of_point` *Calculates value of grow between two measured values.*

Description

Calculates value of grow between two measured values.

Usage

```
calculate_grow_of_point(data_with_grow, time_of_point)
```

Arguments

`data_with_grow`	data.frame with _grow_ column
`time_of_point`	POSIXct datetime of value that will be calculated

Value

numeric calculated value

Examples

```
## Not run:
alarmanalysis::calculate_grow_of_point(data)

## End(Not run)
```

`calculate_point_on_line_given_by_points` *Calculates *y* value of point on defined line.*

Description

Calculates *y* value of point on defined line.

Usage

```
calculate_point_on_line_given_by_points(x1, y1, x2, y2, x)
```

Arguments

`x1`	of first point defining line (numeric)
`y1`	of first point defining line (numeric)
`x2`	of second point defining line (numeric)
`y2`	of second point defining line (numeric)
`x`	numeriv value of requested point

Value

numeric value of *y*

Examples

```
alarmanalysis::calculate_point_on_line_given_by_points(0, 3, 2, 7, 1)
```

---

calculate_pulses_grow *Calculates grow of pulses count for each given inputs.*

Description

Calculates grow of pulses count for each given inputs.

Usage

```
calculate_pulses_grow(data, inputs = c(1, 0))
```

Arguments

data	data.frame that have _Input_ and _Value_ columns
inputs	list of available inputs (default: c(1, 0))

Value

given data.frame with calculated column _grow_

Examples

```
## Not run:
alarmanalysis::calculate_pulses_grow(data)

## End(Not run)
```

---

convert_column_to_POSIXct *Converts given character into POSIXct object*

Description

Converts given character into POSIXct object

Usage

```
convert_column_to_POSIXct(
  data,
  column = "Time",
  date_format = "%Y-%m-%d %H:%M:%S"
)
```

Arguments

data	input data.frame
column	column of character that will be converted
date_format	character format

Value

data.frame with converted data

Examples

```
## Not run:
alarmanalysis::convert_column_to_POSIXct(input_data)

## End(Not run)
```

---

dataframe_anonymization

Function that anonymize data in given data.frame

Description

Function that anonymize data in given data.frame

Usage

```
dataframe_anonymization(data, anonymize_params, trim_unknown_variables = FALSE)
```

Arguments

data	data.frame of data that should be anonymized
anonymize_params	list of `dataframe_anonymize_param` function results
trim_unknown_variables	if TRUE than removes columns that are not listed in `anonymize_params`. Default is FALSE

Value

data.frame of anonymized data

Examples

```
## Not run:
dataframe_anonymization(
data = iris,
anonymize_params = list(dataframe_anonymize_param(column = 'Sepal.Length',
                                                  new_name = 'length',
                                                  hash_anonymization,
                                                  args = list(alg = 'md5'))))

## End(Not run)
```

dataframe_anonymize_param

Function that return structure of anonymize_param used in `dataframe_anonymization`

Description

Function that return structure of anonymize_param used in `dataframe_anonymization`

Usage

```
dataframe_anonymize_param(
  column,
  new_name = NULL,
  func = hash_anonymization,
  args = NULL
)
```

Arguments

`column`	name of colomn that is going to be anonymized
`new_name`	in case this arg is set column will be renamed. Default is NULL
`func`	function that will be applied on column. Default is `hash`
`args`	list of arguments for anonymization function.

Value

list of given arguments with naming

Examples

```
alarmanalysis::dataframe_anonymize_param(column = 'foo',
                                         new_name = 'bar',
                                         func = hash_anonymization,
                                         args = list(alg = 'md5'))
```

detect_fake_alarm *Method*

Description

Method

Usage

```
detect_fake_alarm(data, time_span_in_seconds = 30)
```

Arguments

`data`	of events data.frame. Data.frame should contain columns: Action with values: InstantAlarms, AlarmCanceled, Disarmed. Location: character. Time: datetime in format `Y-m-d H:M:S`
`time_span_in_seconds`	delay between alarm and cancelation in seconds.

Value

list of `fake_alerts`, `true_alerts` list with indexes of alerts in data.

Examples

```
## Not run:
alarmanalysis::detect_fake_alarm(data)

## End(Not run)
```

`find_seq_around_base` *Method for finding sequencies around base action (_action_section_number_ == 1)*

Description

Method for finding sequencies around base action (_action_section_number_ == 1)

Usage

```
find_seq_around_base(
  prepared_data,
  t_before_minutes = 20,
  t_after_minutes = 5,
  remove_single_event_seqs = TRUE
)
```

Arguments

`prepared_data`	by prepare_data_for_2dim_analysis methods
`t_before_minutes`	time interval before base event in minutes
`t_after_minutes`	time interval after base event in minutes
`remove_single_event_seqs`	if TRUE it will omit sequencies with just single event (default: TRUE)

Value

data.frame of given data with _seq_ column

Examples

```
## Not run:
alarmanalysis::find_seq_around_base(prepared, 20, 5)

## End(Not run)
```

`get_each_day_between_hours` *Method that filter rows that are between given hours. (ignore date)*

Description

Method that filter rows that are between given hours. (ignore date)

Usage

```
get_each_day_between_hours(data, posixsct_column_name, from, to)
```

Arguments

`data`	frame of data
`posixsct_column_name`	name of the column that have to be in POSIXct format
`from`	hour in range of c(0:24)
`to`	hour in range of c(0:24)

Value

filtered data

Examples

```
## Not run:
alarmanalysis::get_each_day_between_hours(data_frame, 'Time', 6, 8)

## End(Not run)
```

`get_habits` *Suggests automatization (which is sequence of events) based on given data around base action (_action_section_number_ == 1). It uses these methods for suggesting: _find_seq_around_base_, _get_similarity_matrix_ and _get_sequence_occurrence_*

Description

Suggests automatization (which is sequence of events) based on given data around base action (_action_section_number_ == 1). It uses these methods for suggesting: _find_seq_around_base_, _get_similarity_matrix_ and _get_sequence_occurrence_

Usage

```
get_habits(
  prepared_data,
  min_occurrences_percentage = 0.3,
  t_before_minutes = 20,
  t_after_minutes = 5,
  remove_single_event_seqs = TRUE,
  ignore_event_order = FALSE,
  toleration = 0.2
)
```

Arguments

prepared_data	by prepare_data_for_2dim_analysis methods
min_occurrences_percentage	set minimal percentage of occurrences of sequence that will be included in suggestion. (default: 0.30)
t_before_minutes	time interval before base event in minutes
t_after_minutes	time interval after base event in minutes
remove_single_event_seqs	if TRUE it will omit sequencies with just single event (default: TRUE)
ignore_event_order	if TRUE it will ignore of events in sequence. (default: FALSE)
toleration	of difference between matched sequencies in percent (default: 0.20)

Value

data.frame with columns _actions_, _action_section_numbers_, _occurrences_ and _occurrences_percent_

Examples

```
## Not run:
alarmanalysis::get_habits(prepared)

## End(Not run)
```

---

get_sequence_occurrence

Look up over similarity matrix and select similar sequencies (with given toleration). Than it compute occurrencies of these sequencies.

Description

Look up over similarity matrix and select similar sequencies (with given toleration). Than it compute occurrencies of these sequencies.

Usage

```
get_sequence_occurrence(similarity_matrix, toleration = 0.2)
```

Arguments

similarity_matrix	computed by _get_similarity_matrix_
toleration	of difference between matched sequencies in percent (default: 0.20)

Value

data.frame with columns: seq, occurrences, occurences_percent and matched_sequencies (which is list of sequencies that are similar to given sequence).

Examples

```
## Not run:
alarmanalysis::get_sequence_occurrence(similarity_matrix)

## End(Not run)
```

---

get_similarity_matrix *For computing similarity matrix of given sequencies using Manhattan metric.*

Description

For computing similarity matrix of given sequencies using Manhattan metric.

Usage

```
get_similarity_matrix(data, ignore_order = FALSE)
```

Arguments

data	data.frame of events with computed _seq_ column. (see. _find_seq_around_base_)
ignore_order	if TRUE it will ignore of events in sequence. (default: FALSE)

Value

n x n matrix where n is count of sequencies. Each value indicates similarity of two sequencies (row and column). 0 indicates absolute similarity and bigger value distinction.

Examples

```
## Not run:
alarmanalysis::get_similarity_matrix(prepared_with_seqs)

## End(Not run)
```

`hash_anonymization` *Function that hash value to be anonymized*

Description

Function that hash value to be anonymized

Usage

```
hash_anonymization(data, alg = "md5", ...)
```

Arguments

`data`	to be anonymized. In case it is list all values will be hashed. Other structures will be serialized.
`alg`	hashing algorithms. One of these "md5", "sha1", "crc32", "sha256", "sha512", "xxhash32", "xxhash64", "murmur32", "spookyhash". Default is `"md5"`
`...`	rest of arguments. see `digest()`

Value

hash of data

Examples

```
## Not run:
alarmanalysis::hash_anonymization('foo')

## End(Not run)
```

`merge_csv_files_in_folder` *Method that merge all csv files in folder.*

Description

Method that merge all csv files in folder.

Usage

```
merge_csv_files_in_folder(folder_path)
```

Arguments

`folder_path`	path to the folder which contains only csv file that should be imported.

Value

data.frame of merged csv files

Examples

```
## Not run:
alarmanalysis::merge_csv_files_in_folder('some/folder')

## End(Not run)
```

`modus` *Get modus of vector*

Description

Get modus of vector

Usage

```
modus(vector)
```

Arguments

`vector`	of data

Value

modus

Examples

```
## Not run:
alarmanalysis::modus(c(1, 2, 2, 2, 3))

## End(Not run)
```

`nop_anonymization` *Function that only returns same data as given. Used for skipping columns during data.frame anonymization.*

Description

Function that only returns same data as given. Used for skipping columns during data.frame anonymization.

Usage

```
nop_anonymization(data)
```

Arguments

`data`	to be anonymized

Value

anonymized data

Examples

```
## Not run:
alarmanalysis::nop_anonymization('foo')

## End(Not run)
```

`order_data_fram_by_column` *Method for simple ordering data frame by column.*

Description

Method for simple ordering data frame by column.

Usage

```
order_data_fram_by_column(data, column, order = "ASC")
```

Arguments

`data`	frame to be ordered
`column`	name by which should be data frame ordered
`order`	default is 'ASC'. Can be values of 'ASC' and 'DESC'.,

Value

ordered data frame

Examples

```
## Not run:
alarmanalysis::order_data_fram_by_column(data_frame, 'column_name', 'DESC')

## End(Not run)
```

`plot_events_freq_histogram_by_hour`
Method that plot histogram of events frequency by hours.

Description

Method that plot histogram of events frequency by hours.

Usage

```
plot_events_freq_histogram_by_hour(data)
```

Arguments

`data`	data.frame of events that must include Time column

Value

void

Examples

```
## Not run:
alarmanalysis::plot_events_freq_histogram_by_hour(data)

## End(Not run)
```

`plot_events_time_line_around_base`
Method that visualize events in time.

Description

Method that visualize events in time.

Usage

```
plot_events_time_line_around_base(plot_data, plot_style = "daily")
```

Arguments

`plot_data`	data prepared by _alarmanalysis::prepare_data_for_2dim_analysis_
`plot_style`	character that specify analysis in terms of time. Supported values: 'daily'

Value

ggplot2 plot structure

Examples

```
## Not run:
alarmanalysis::plot_events_time_line_around_base(data)

## End(Not run)
```

`prepare_data_for_2dim_analysis_armed`
Method for preparing data for 2dim analysis around armed.

Description

In first part this method normalize data: - if requested it merge all armed (remote, partial etc.) actions in to single action _armed_ - remove NA values that can break the final analysis - order by Time ASC

Then it converts all combinations of action and section into dial and assings new dial values. There is reserved values for _armed_ - **1** and _no-action_ - **0**. All other values are greater than **1**.

Eventually series are resampled to **Fs = 1**.

Usage

```
prepare_data_for_2dim_analysis_armed(
  data,
  armed_action_names = c("Armed", "ArmedPartially", "ArmedPartiallyRemotely",
    "ArmedRemotely"),
  programmable_port_action_names = c("PgOn", "PgOff"),
  merge_armed_action = TRUE,
  remove_duplicates = TRUE,
  resampling = FALSE
)
```

Arguments

`data`	frame that have following columns **Action** - character describing action in time **Location** - character, events location. Should be same for all rows. Otherwise the first location will be taken for analysis. **Time** - POSIXct datetime of event **Section** - character
`armed_action_names`	names of actions that indicates arming location
`programmable_port_action_names`	names of actions that indicates programmable ports actions
`merge_armed_action`	if TRUE all action that are generally armed type will be merged into single Action type
`remove_duplicates`	if TRUE it will remove duplicates (default: TRUE)
`resampling`	if TRUE it will resample data (default: TRUE)

Value

data prepared for 2dim analysis

Examples

```
## Not run:
alarmanalysis::prepare_data_for_2dim_analysis_armed(data)

## End(Not run)
```

`prepare_data_for_2dim_analysis_disarmed`
Method for preparing data for 2dim analysis around disarmed.

Description

In first part this method normalize data: - if requested it merge all disarmed (remote, partial etc.) actions in to single action _disarmed_ - remove NA values that can break the final analysis - order by Time ASC

Then it converts all combinations of action and section into dial and assings new dial values. There is reserved values for _disarmed_ - **1** and _no-action_ - **0**. All other values are greater than **1**.

Eventually series are resampled to **Fs = 1**.

Usage

```
prepare_data_for_2dim_analysis_disarmed(
  data,
  disarmed_action_names = c("Disarmed", "DisarmedPartially",
    "DisarmedPartiallyRemotely", "DisarmedRemotely"),
  programmable_port_action_names = c("PgOn", "PgOff"),
  merge_disarmed_action = TRUE,
  remove_duplicates = TRUE,
  resampling = FALSE
)
```

Arguments

`data`	frame that have following columns **Action** - character describing action in time **Location** - character, events location. Should be same for all rows. Otherwise the first location will be taken for analysis. **Time** - POSIXct datetime of event **Section** - character
`disarmed_action_names`	names of actions that indicates disarming location
`programmable_port_action_names`	names of actions that indicates programmable ports actions
`merge_disarmed_action`	if TRUE all action that are generally disarmed type will be merged into single Action type
`remove_duplicates`	if TRUE it will remove duplicates (default: TRUE)
`resampling`	if TRUE it will resample data (default: TRUE)

Value

data prepared for 2dim analysis

Examples

```
## Not run:
alarmanalysis::prepare_data_for_2dim_analysis_disarmed(data)

## End(Not run)
```

`random_integer_anonymization`
Function that generate random integer to anonymize data

Description

Function that generate random integer to anonymize data

Usage

```
random_integer_anonymization(data, factorize = TRUE)
```

Arguments

`data`	to be anonymized
`factorize`	if FALSE all values will have unique values. Default is TRUE that will anonymize factors.

Value

anonymized data

Examples

```
## Not run:
alarmanalysis::random_person_anonymization('foo')

## End(Not run)
```

`random_location_anonymization` *Function that generate random address to anonymize data*

Description

Function that generate random address to anonymize data

Usage

```
random_location_anonymization(data, factorize = TRUE)
```

Arguments

`data`	to be anonymized
`factorize`	if FALSE all values will have unique values. Default is TRUE that will anonymize factors.

Value

anonymized data

Examples

```
## Not run:
alarmanalysis::random_location_anonymization('foo')

## End(Not run)
```

`random_person_anonymization` *Function that generate random person name to anonymize data*

Description

Function that generate random person name to anonymize data

Usage

```
random_person_anonymization(data, factorize = TRUE)
```

Arguments

`data`	to be anonymized
`factorize`	if FALSE all values will have unique values. Default is TRUE that will anonymize factors.

Value

anonymized data

Examples

```
## Not run:
alarmanalysis::random_person_anonymization('foo')

## End(Not run)
```

`read_data` *Reads csv file with _stringsAsFactors_ set to _FALSE_.*

Description

Reads csv file with _stringsAsFactors_ set to _FALSE_.

Usage

```
read_data(file_path)
```

Arguments

`file_path`	path to csv file

Value

readed data.frame

Examples

```
## Not run:
alarmanalysis::read_data('../data_file.csv')

## End(Not run)
```

Index