



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ

DEPARTMENT OF INFORMATION SYSTEMS

ZÍSKÁVÁNÍ ZNALOSTÍ Z WEBOVÝCH LOGŮ

KNOWLEDGE DISCOVERY FROM WEB LOGS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

SAMUEL VALAŠTÍN

VEDOUCÍ PRÁCE

SUPERVISOR

Ing. VLADIMÍR BARTÍK, Ph.D.

BRNO 2022

Zadání bakalářské práce



Student: **Valaštin Samuel**
Program: Informační technologie
Název: **Získávání znalostí z webových logů**
Knowledge Discovery from Web Logs

Kategorie: Data mining

Zadání:

1. Prostudujte problematiku získávání znalostí z dat, poté se podrobněji zaměřte na získávání znalostí z webových logů.
2. Vyhledejte a seznamte se s několika dostupnými datasey vhodnými pro dolování webových logů.
3. Po konzultaci s vedoucím navrhnete aplikaci provádějící alespoň dvě dolovací úlohy nad zvolenými datasey.
4. Aplikaci implementujte a proveďte experimenty, které vyhodnotí úspěšnost úloh.
5. Zhodnoťte dosažené výsledky a další možnosti pokračování tohoto projektu.

Literatura:

- Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Third Edition. Morgan Kaufmann Publishers, 2012, 703 p., ISBN 978-0-12-381479-1.
- Lu Z., Yao Y., Zhong N. Web Log Mining. In: Zhong N., Liu J., Yao Y. (eds) Web Intelligence. Springer, Berlin. 2003.

Pro udělení zápočtu za první semestr je požadováno:

- Body 1-3.

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

Vedoucí práce: **Bartík Vladimír, Ing., Ph.D.**

Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.

Datum zadání: 1. listopadu 2021

Datum odevzdání: 11. května 2022

Datum schválení: 11. října 2021

Abstrakt

Táto bakalárska práca rieši problematiku získavania znalostí z webových logov. Zdroj dát v podobe webových prístupových logov umožňuje po vhodnom predspracovaní použitie mnohých techník, ktoré sú určené pre získavanie znalostí. Aplikáciou týchto techník na predspracované dáta je možné klasifikovať užívateľské správanie do skupín, vyhľadať zaujímavé asociácie v správaní užívateľov, či nájsť v bežnom užívateľskom správaní predom neznáme sekvencie.

Abstract

This bachelor thesis deals with the problem of knowledge discovery from web logs. The data source in the form of web access logs allows, after appropriate preprocessing, the use of a number of techniques that are designed to deal with knowledge discovery. By applying these techniques to preprocessed data, it is possible to classify user behavior into groups, to discover interesting associations in user behavior, or to discover previously unknown sequences in common user behavior.

Klíčové slová

získavanie znalostí, dolovanie znalostí, získavanie znalostí z webu, prístupové logy, zhľuková analýza, dolovanie frekventovaných vzorov, dolovanie asociačných pravidiel, dolovanie sekvenčných vzorov, Python, k-means, BIRCH, DBSCAN, Apriori, FP-Growth, PrefixSpan

Keywords

knowledge discovery, data mining, knowledge discovery from web, access logs, cluster analysis, mining frequent patterns, mining association rules, sequential pattern mining, Python, k-means, BIRCH, DBSCAN, Apriori, FP-Growth, PrefixSpan

Citácia

VALAŠTÍN, Samuel. *Získávaní znalostí z webových logů*. Brno, 2022. Bakalárska práca. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Vladimír Bartík, Ph.D.

Získávání znalostí z webových logů

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Ing. Vladimíra Bartíka Ph.D. Uviedol som všetky literárne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

.....
Samuel Valaštín
8. mája 2022

Podakovanie

Chcel by som sa poďakovať vedúcemu tejto práce Ing. Vladimírovi Bartíkovi Ph.D za jeho odbornú pomoc, konzultácie a cenné rady, ktoré mi poskytol pri vypracovaní tejto práce.

Obsah

1	Úvod	2
2	Získavanie znalostí z logov	3
2.1	Získavanie znalostí	3
2.2	Životný cyklus získavania znalostí	4
2.3	Získavanie znalostí z webu	6
2.4	Log – zdroj užitočných informácií	7
3	Predspracovanie dát	11
3.1	Extrakcia a čistenie dát	11
3.2	Identifikácia užívateľov	12
3.3	Rozdelenie záznamov do užívateľských sedení	14
3.4	Skompletizovanie ciest užívateľských sedení	15
3.5	Prieskumná analýza užívateľských sedení	16
4	Dolovanie znalostí	18
4.1	Zhluková analýza	18
4.2	Získavanie znalostí pomocou asociačných pravidiel	22
4.2.1	Dolovanie frekventovaných vzorov	22
4.2.2	Dolovanie asociačných pravidiel	26
4.3	Dolovanie sekvenčných vzorov	27
5	Implementácia aplikácie	28
5.1	Návrh a požiadavky na aplikáciu	28
5.2	Použité knižnice	29
5.3	Štruktúra aplikácie	32
5.4	Predspracovanie webových prístupových logov	34
5.4.1	Detekcia a spracovanie formátov webových prístupových logov . . .	34
5.4.2	Zbieranie a vizualizácia štatistík	35
5.4.3	Aplikácia prieskumnej analýzy	37
5.5	Ukladanie spracovaných dátových rámcov a štatistík	38
5.5.1	Spôsob ukladania a načítania predspracovaných rámcov a štatistík .	39
5.5.2	Administrácia správy ukladania	39
5.6	Scenáre pre získavanie znalostí z webových logov	41
6	Záver	49
	Literatúra	50

Kapitola 1

Úvod

Vzhľadom na pokračujúci rast využívania webových služieb je v súčasnej dobe stále viac vyžadované analyzovať a spoznávať správanie užívateľov webových služieb. Veľká dostupnosť týchto poskytovaných služieb a zvýšená frekvencia každodenného používania dostupných webových služieb ponúka skvelé príležitosti, ktoré umožňujú analýzu správania užívateľov týchto služieb. Manuálna analýza správania užívateľov webových služieb nie je v dnešnej dobe realizovateľná, a to z dôvodu obrovského objemu dát, ktoré webové servery ukladajú s každou jednou užívateľskou požiadavkou. Rovnako sú s každou relevantnou užívateľskou požiadavkou serverom ukladané aj požiadavky pre rôzne grafické či štýlovacie súbory, ktoré nie sú zaujímavé pre analýzu správania užívateľov. Získavanie znalostí o využívaní webových služieb umožňuje kategorizáciu správania užívateľov a tiež môže poskytnúť dôležité poznatky o využívaní webových služieb užívateľmi. Rovnako dôležitou činnosťou je hľadanie asociácií v užívateľskom správaní. Cieľom tejto práce je oboznámenie sa a následná aplikácia základných úloh určených pre riešenie tejto problematiky. Kapitola 2 rieši všeobecný úvod do tejto problematiky. V tejto kapitole je predstavený životný cyklus procesu získavania znalostí, na ktorý nadväzuje problematika získavania znalostí z webu. Nakoniec sú v tejto kapitole predstavené základné zdroje dát, ktoré sú dostupné z webu. Na toto základné rozdelenie nadväzuje predstavenie štandardizovaných logovacích formátov, ktoré sú využívané s cieľom uchovania užívateľského prehliadania webových stránok. Webové prístupové logy sú základným zdrojom dát pre túto prácu. Preto sú v podkapitole 2.4 detailne predstavené spoločne so všetkými poliami, ktoré sú pre logovacie záznamy daných formátov ukladané. Dáta v podobe v akej boli predstavené v kapitole 2 nie sú vo vhodnej podobe pre získavanie znalostí, a tak sú v kapitole 3 popísané základné techniky, ktoré umožňujú spracovanie a transformáciu dát do formy, ktorú vyžadujú jednotlivé algoritmy pre získavanie znalostí. Následne sú v kapitole 4 predstavené základné a často využívané modelovacie techniky a algoritmy, ktoré sú určené pre získavanie znalostí z webových prístupových logov. Niektoré z predstavených techník vyžadujú dodatočnú úpravu formátu vstupných dát pre danú techniku, avšak vzhľadom na fakt, že ide o úpravu vhodnú iba pre danú špecifickú techniku, tak je táto činnosť popísaná v tejto kapitole, a nie v kapitole zameranej na predspracovanie dát. Posledná kapitola tejto práce 5 popisuje výslednú aplikáciu¹ riešiacu problematiku, ktorá sa zaoberá získavaním znalostí z webových logov. Obsah tejto kapitoly je zameraný na návrh a implementáciu aplikácie. Okrem týchto bodov sa táto kapitola rovnako zaoberá dostupnými možnosťami, ktoré implementovaná aplikácia poskytuje a rovnako tak popisuje knižnice, ktoré vývoj a prácu na aplikácii tohto typu značne uľahčujú.

¹<https://logmine.herokuapp.com/>

Kapitola 2

Získavanie znalostí z logov

Predtým ako bude bližšie popísaná problematika, ktorou sa táto práca zaoberá je nutné priblížiť si základné pojmy, ktoré umožnia túto problematiku hlbšie pochopiť. Web je jedným z najčastejšie využívaných zdrojov pre získavanie znalostí z dát. Samotný web poskytuje množstvo dát, z ktorých aplikáciou vhodných postupov možno vydolovať nové znalosti.

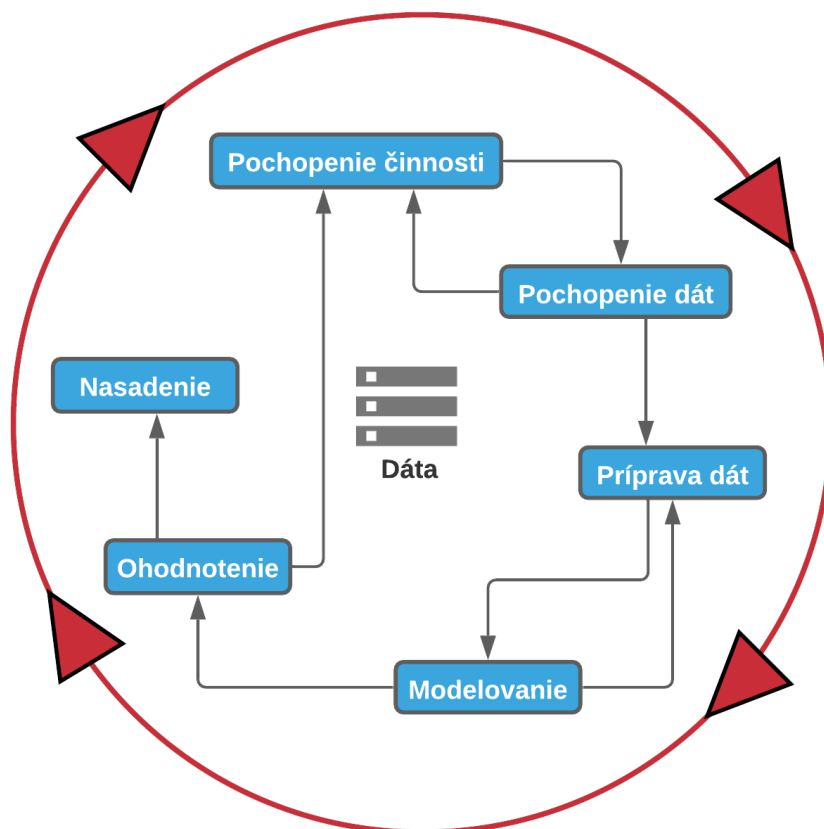
2.1 Získavanie znalostí

Získavanie znalostí, často označované ako dobývanie znalostí z databáz (KDD) je proces objavovania užitočných vzorov alebo informácií z dátových zdrojov [16]. Vzory musia byť platné, potenciálne užitočné a zrozumiteľné. Samotný proces začína pochopením aplikáčnej domény, pričom je nutné identifikovať vhodné zdroje dát a rovnako je potrebné správne zadefinovať aké dáta budú uchovávané. Aplikácia pre získavanie znalostí je obvykle vykonávaná v troch hlavných krokoch [1]:

- **Zber dát:** Prvou, a zároveň veľmi dôležitou fázou v procese získavania znalostí je zber vhodných dát pre získavanie znalostí. Ide o veľmi špecifickú fázu, pričom rozhodnutia v tejto fáze vedú výrazne ovplyvniť celý proces dolovania znalostí. Postupy pre zber dát sa môžu značne líšiť v závislosti od rôznych zdrojov a typov získavaných dát. Po dokončení tohto kroku sú získané dáta tradične ukladané do dátového skladu.
- **Extrakcia a čistenie dát:** Dáta získané v predchádzajúcom kroku zvyčajne nie sú vo vhodnej forme pre spracovanie algoritmi na dolovanie znalostí. Preto je v tomto kroku nutné získané dáta transformovať do prijateľnej podoby pre spracovanie týmito algoritmi. Pre dolovanie je dôležité extrahovať zaujímavé vlastnosti dát, a zároveň sa zbaviť vlastností, ktoré nie sú pre dolovanie znalostí ničím užitočné. Niektoré aplikácie využívajú dáta získané z mnohých zdrojov, ktoré je pred samotným spracovaním nutné vhodným spôsobom integrovať do jednotného formátu. Cieľom tohto kroku je dosiahnuť štrukturovaný súbor dát, ktorý možno využiť pre algoritmy dolovania znalostí.
- **Analytické spracovanie a dolovanie znalostí:** Záverečným krokom procesu dolovania znalostí je aplikácia účinných metód pre predspracované dáta. Tieto dáta sú privedené do algoritmu pre získavanie znalostí z dát, ktorý objavuje užitočné vzory alebo informácie z týchto dát. [16]

2.2 Životný cyklus získavania znalostí

Metodika *Cross-Industry Standard Process for Data Mining (CRISP-DM)* popisuje životný cyklus získavania znalostí [6]. Model tejto metodiky rozdeľuje proces získavania znalostí do šiestich fáz. Medzi jednotlivými fázami existujú vzájomné vzťahy. Postupnosť fáz nie je striktná a umožňuje pohyby medzi jednotlivými fázami smerom vpred aj naspäť. Vonkajší kruh diagramu predstavuje cyklickú povahu získavania dát. Fáza nasadenia nemusí nutne znamenať koniec získavania znalostí. Získané znalosti sa môžu využiť pre ďalšie dolovanie, ktoré môže ťažiť z predchádzajúcich skúseností. V tejto podkapitole sú predstavené jednotlivé fázy tejto metodiky, ktoré bližšie približujú ako vyzerá samotný životný cyklus projektu zameraného na získavanie znalostí z dát.



Obr. 2.1: Diagram modelu CRISP-DM.

Fáza pochopenia činnosti

Počiatková fáza životného cyklu je zameraná na určenie a pochopenie cieľov aj požiadaviek z obchodného hľadiska. Následne sú tieto ciele a požiadavky transformované do definície problému pre získavanie znalostí. V tejto fáze je rovnako potrebné zadefinovať ciele a kritériá úspechu činnosti projektu. Vzhľadom na možné riziká neúspechu je nutné vypracovať plán projektu, ktorý je rozdelený na jednotlivé fázy, ktoré počítajú s opatreniami, akým smerom bude pokračovať projekt v prípade výskytu problémov.

Fáza pochopenia dát

V tejto fáze je nutné získať sadu dát a bližšie sa s ňou zoznámiť. Získané dáta je potrebné preskúmať, charakterizovať, identifikovať a overiť kvalitu dát, či prípadne rozdeliť dáta na vhodné podmnožiny. Pri overení kvality dát je nutné zistiť, či sú dáta úplné (či poskytujú všetky potrebné údaje), prípadne je nutné overiť chybovosť jednotlivých záznamov a zároveň početnosť ich výskytu. V prípade využívania viacerých zdrojov dát, môže viesť integrácia týchto zdrojov k ďalším problémom, ktoré treba brať do úvahy pred posunom do ďalšej fázy životného cyklu projektu.

Fáza prípravy dát

Vo fáze prípravy dát sa vykonávajú všetky činnosti potrebné pre skonštruovanie konečnej množiny dát. Úlohy z predchádzajúcej fázy je možné vykonávať viacnásobne a zvyčajne nemajú striktné určené poradie vykonávania. Na základe cieľov a obmedzení projektu je potrebné vykonať správny výber dát, vyčistiť dáta a následne transformovať a integrovať dáta. Výstupom tejto fázy je konečná množina dát, ktorá bude privedená do nasledujúcej fázy modelovania.

Fáza modelovania

Prvým krokom vo fáze modelovania je výber vhodných modelovacích techník. Zvyčajne existuje viacero techník pre rovnaký problém získavania znalostí. Niektoré techniky môžu mať špecifické požiadavky na formu dát. Kalibráciou nastavenia modelu a vstupných parametrov je možné optimalizovať výsledky modelovania [19]. Rovnako treba zväžiť časovú a pamäťovú zložitosť jednotlivých modelovacích techník pri práci s veľkými súborami spracovaných dát.

Fáza ohodnotenia

Pred konečným nasadením modelu pre získanie znalostí je dôležité overiť a ohodnotiť kvalitu, či efektívnosť zvoleného modelu. Výsledky z predchádzajúcej fázy sú porovnávané s cieľmi a požiadavkami z fázy pochopenia činnosti. Na základe výsledkov ohodnotenia sa rozhoduje o pokračovaní projektu. Ak sú splnené všetky obchodné ciele a požiadavky, tak možno prejsť k záverečnej fáze nasadenia. Ak v použítom modeli nie je zahrnutý dôležitý aspekt skúmaného problému, tak je potrebné vrátiť sa k fáze pochopenia činnosti.

Fáza nasadenia

V záverečnej fáze sú získané výsledky spracované do podoby vhodnej pre užívateľa. Pre samotného užívateľa je dôležité, aby chápal aké činnosti je potrebné vykonať pre využitie získaných výsledkov. Výstupom tejto fázy sa môže líšiť v závislosti od požiadaviek, ktoré má zvolený model vykonávať. Výstupom použitých modelov môže byť napríklad vygenerovanie správy alebo prípadne graf, ktorých hodnoty reprezentujú výsledky získané použitím zvoleného modelu.

2.3 Získavanie znalostí z webu

Získavanie znalostí z webu aplikuje metódy, metodológie a techniky získavania znalostí na rôzne formy webových dát. Vzhľadom k rôznorodému, pološtrukturovanému, či prípadne neštrukturovanému charakteru je web bohatým zdrojom pre získavanie rôznych dát [1]. Úlohy, ktorými sa zaoberá táto oblasť sú ďalej rozdelené na základe rôznych kategórií zdrojov dát, ktoré sú využívané algoritmami pre získavanie znalostí. V tejto kapitole je predstavené základné rozdelenie kategórií získavania znalostí z webu. V literatúre sa často toto rozdelenie líši, pričom táto podkapitola zachováva rozdelenie kategórií publikované v [16], ktoré je primárne rozšírené o poznatky z [1], ktorá zahŕňa kategóriu získavania znalostí z webových štruktúr do kategórie získavania znalostí z webového obsahu.

Získavanie znalostí z webových štruktúr

Primárnym cieľom tejto podoblasti je proces odvodzovania znalostí medzi odkazmi a referenciami [24]. Toto odvetvie sa mimo iného zaoberá samotnou štruktúrou hypertextových odkazov. Hlavným zdrojom dát sú *údaje o prepojení* medzi webovými stránkami a odkazmi. Web je v tomto kontexte chápaný ako graf. Webové stránky reprezentujú uzly grafu. Prepojenia sú chápané ako hrany medzi uzlami. Tento kontext je využívaný pre vyhľadávanie na webe alebo pre hľadanie podobností medzi uzlami.

Získavanie znalostí z webového obsahu

Zdrojom informácií pre túto kategóriu je obsah webových dokumentov. Z obsahu webových dokumentov sa extrahujú a získavajú užitočné informácie či znalosti. Rozšírené sú najmä techniky získavania znalostí z textových dát. Dokumenty sú navzájom prepojené pomocou hypertextových odkazov. Tento fakt iba potvrdzuje úzke prepojenie medzi touto kategóriou a kategóriou získavania znalostí z webových štruktúr. Typickým príkladom získavania znalostí pre túto kategóriu je klasifikácia dokumentov na základe témy.

Získavanie znalostí z používania webu

Problematika **získavania znalostí z používania webu** sa pomocou aplikácie techník pre dolovanie znalostí snaží priradiť webového užívateľa k ostatným užívateľom, ktorých správanie a preferencie sa najviac zhodujú [19]. Na rozdiel od ostatných kategórií, ktoré využívajú ako primárne zdroje dát obsah dokumentov alebo odkazy sú znalosti pre túto kategóriu získavané zo vzorov aktivity užívateľov webových aplikácií, pričom medzi najčastejšie zdroje dát pre túto kategóriu patria:

- *Webové transakcie, hodnotenia a spätná väzba:* Návštevníci webových služieb vyhľadávajú alebo nakupujú rôzne typy položiek, prípadne prezentujú svoj názor formou recenzie či hodnotenia. V týchto prípadoch možno ich správanie využiť na získanie znalostí o preferenciách návštevníkov webových služieb.
- *Webové logy:* Správanie užívateľov pri prehliadaní webových stránok je zvyčajne zaznamenávané webovým serverom vo forme webových logov. Webové logy sú užitočným zdrojom dát pre určenie relevantných vzorov užívateľského prehliadania stránok. Táto práca sa zaoberá touto podoblasťou získavania znalostí a v nasledujúcich kapitolách bude táto problematika ďalej rozvedená.

2.4 Log – zdroj užitočných informácií

V predchádzajúcich podkapitolách bola všeobecne predstavená problematika, ktorou sa táto práca zaoberá. Rovnako bol predstavený životný cyklus procesu pre získavanie znalostí. Pre správne pochopenie a následné vyriešenie problému je dôležité zoznámiť sa s obsahom a formátom zdroja dát. Obsah tejto podkapitoly je zaradený v životnom cykle získavania znalostí do fázy pochopenia činnosti 2.2. Informácie podané v tejto podkapitole sú zamerané na kategorizáciu logov, ktoré predstavujú zdroj dát pre túto prácu. Následne sú v tejto podkapitole predstavené najznámejšie a najčastejšie využívané štandardizované formáty webových prístupových logov.

Definícia pojmu log

Log možno najpresnejšie charakterizovať ako chronologický a systematický zber udalostí [4]. Existuje množstvo rôznych typov webových logov, ktoré sú vzhľadom k obrovskej rozšírenosti webových služieb uchovávané. Rozšírené sú najmä užívateľské logy, systémové logy, serverové logy či dopytové logy. Medzi najčastejšie využívané typy webových logov pre získavanie znalostí z webových logov patria najmä posledné dve kategórie, ktoré sú v tejto podkapitole zbežne predstavené:

- **Serverové prístupové logy:** Pre každú klientskú požiadavku na webový server je automaticky vygenerovaná odpoveď. Táto odpoveď má formu jednoriadkového záznamu, ktorý je následne pripojený k textovému súboru na webovom serveri. Obsah týchto záznamov zodpovedá užívateľskej aktivite na webových stránkach. Zvyčajne sú ukladané v štandardizovaných formátoch ako napríklad *NCSA Common Log format*, *Combined log format* či *W3C Extended Log File Format*. Obsah a vlastnosti týchto štandardizovaných formátov budú v tejto podkapitole 2.4 ďalej predstavené.
- **Dopytové vyhľadávacie logy:** Obsah týchto logov zodpovedá dopytu užívateľov pri vyhľadávaní. Na rozdiel od serverových logov nemá tento typ logov jednotný štandard, v ktorom sú užívateľské vyhľadávania ukladané. Zvyčajne ide o rôzne komerčné riešenia využívajúce vlastný formát ukladania vyhľadávacích záznamov. Spoločnou charakteristikou obsahu týchto logov je ukladanie užívateľskej otázky a odkazu na stránku, ktorú užívateľ následne navštívil. Okrem uvedených vlastností sú tradične ukladané aj dátum a čas užívateľského vyhľadávania a tiež aj vertikálny index pozície výsledku, na ktorý užívateľ klikol zo zoznamu výsledkov poskytovaných vyhľadávačom po zadaní užívateľskej otázky.

Obsah štandardizovaných formátov serverových prístupových logov

Webové prístupové logy sú ukladané v rôznych formátoch, ktoré sa líšia v závislosti od konfigurácie webového serveru. Medzi jednotlivými štandardizovanými formátmi existuje veľká podobnosť a rozdiel predstavujú len rôzne polia, ktoré tieto formáty ukladajú alebo prípadne neukladajú. Väčšina formátov pracuje s pevným počtom polí, ktoré neumožňujú prispôbenie ukladaných informácií. Na druhej strane bude v tejto podkapitole rovnako predstavený aj štandardizovaný formát, ktorý umožňuje zdefinovať formát ukladaných informácií.

- **Common Log Format** – Štandardizovaný formát s pevným počtom polí, ktorý pre niektorých správcov webových služieb nemusí nutne ukladať dostatok informácií. Každý záznam tohto formátu je tvorený nasledujúcimi siedmimi poliami:

1. *Vzdialený hosťiteľ* – Tradične obsahuje IP adresu klienta, ktorý vytvoril požiadavku na webový server. IP adresa môže byť nahradená doménovým menom klienta v prípade, že je možné cez systém DNS (Domain Name System) získať doménové meno vzdialeného hosta [15].
2. *Identifikátor klienta* – Ak webový server vykonáva kontrolu identity, tak obsah tohto pola tvorí identifikátor klienta vo formáte *RFC 1413*. Vo väčšine prípadov toto pole obsahuje spojovník, ktorý reprezentuje prázdne pole záznamu.
3. *Užívateľské meno* – Tento identifikátor poskytuje klient v požiadavkách pre prístup k webovým dokumentom, pre ktoré je potrebná autentifikácia. V niektorých prípadoch je užívateľské meno nahradené identifikačným číslom užívateľa. Využívateľnosť tohto pola je rovnaká ako v prípade identifikátora klienta a tak je zvyčajne reprezentované spojovníkom, ktorý substituuje prázdne pole záznamu.
4. *Dátum a čas požiadavky* – Špecifikuje dátum a čas, kedy bolo dokončené spracovanie klientskej požiadavky na strane servera.
5. *HTTP požiadavka* – Obsahuje klientskú požiadavku na vyžiadaný dokument vo formáte "[metóda] [cesta k dokumentu] [protokol]/[verzia protokolu]".
6. *Stavový kód* – Predstavuje číselnú odpoveď zo strany servera na klientskú požiadavku [26].

Rozsah hodnôt	Význam	Popis
100-199	Informačný	Predbežná odpoveď zo strany servera
200-299	Úspešný	Požiadavka klienta bola úspešne spracovaná
300-399	Presmerovanie	Potreba vykonať ďalšie kroky na dokončenie žiadosti
400-499	Chyba klienta	Klient zaslal nesprávnu požiadavku
500-599	Chyba serveru	Server nie je schopný splniť požiadavku

Tabuľka 2.1: Rozdelenie stavových kódov na základe významu jednotlivých hodnôt. Znalosť tohto rozdelenia bude potrebná pre čistenie dát, ktoré rieši kapitola 3.1.

7. *Objem prenosu* – Udáva veľkosť súboru v bajtoch, ktorý bol zo strany servera v prípade úspešne spracovanej klientskej požiadavky zaslaný klientovi.

Vzdialený hosťiteľ	Identifikátor klienta	Užívateľské meno	Dátum a čas požiadavky	HTTP požiadavka	Stavový kód	Objem prenosu
109.230.38.133	-	-	[21/Feb/2022:08:38:51 +0100]	"GET / HTTP/1.0"	302	-
109.230.38.133	-	-	[21/Feb/2022:08:38:52 +0100]	"GET /news.php HTTP/1.0"	200	22527
109.230.38.133	-	-	[21/Feb/2022:05:38:52 +0100]	"GET /themes/styles.css HTTP/1.0"	200	10183

Obr. 2.2: Príklady záznamov vo formáte Common Log Format.

- **W3C Extended Log File Format** – Poskytuje príležitosť pre prispôsobenie ukladaného obsahu. Ide o formát s variabilným počtom polí, čím umožňuje väčšiu kontrolu nad ukladanými údajmi. Riadky tohto formátu sú rozdeľované na dva typy:

1. *Smernice* – Umožňujú bližšiu špecifikáciu a úpravu procesu logovania. Ukladané sú vo formáte: `#[názov smernice]: [špecifikácia smernice]`. Rozlišujú sa povinné a voliteľné smernice. Povinné smernice sa musia povinne nachádzať pred všetkými vstupmi logovacieho súboru tohto štandardizovaného formátu.

Smernica	Špecifikácia smernice	Typ
Version	Verzia formátu, ktorá je pre logovanie využívaná	povinná
Fields	Polia, ktoré sú pre logovací súbor ukladané	povinná
Software	Špecifikuje software, ktorý generuje logovací súbor	nepovinná
Start-Date	Dátum a čas, kedy bolo logovanie zahájené	nepovinná
End-Date	Dátum a čas, kedy bolo logovanie ukončené	nepovinná
Date	Dátum a čas, kedy bol záznam pridaný	nepovinná
Remark	Špecifikuje komentáre, ktoré sú ingorované	nepovinná

Tabuľka 2.2: Smernice pre logovacie súbory formátu W3C Extended Log File Format.

2. *Vstupy* – Pre špecifikáciu vstupov slúži smernica *Fields* (ukladané polia). Zvýšenie flexibility pri práci s ukladanými poliami je možné zabezpečiť použitím prefixov pred jednotlivými poliami [10]. Polia sa rozdeľujú do dvoch kategórií v závislosti od toho, či musia alebo nemusia využívať prefixy.

Prefix	Význam prefixu
cs	požiadavka klienta na server
rs	požiadavka vzdialeného serveru na server
sc	odpoveď od serveru na požiadavku klienta
sr	odpoveď serveru na vzdialený server
x	špecifický identifikátor pre aplikáciu

Tabuľka 2.3: Prefixy, ktoré možno použiť pre špecifikáciu polí. Okrem prefixov zobrazených v tabuľke možno využiť prefix, ktorý identifikuje klienta(c) alebo server(s).

- (a) *Polia s povinným prefixom* – medzi polia s nutnosťou špecifikácie patrí napríklad IP adresa, port, preložené DNS meno klienta, stavový kód, metóda či uri.
- (b) *Polia s nepovinným prefixom* – táto povinnosť neplatí napríklad pre dátum, čas, uplynulý čas alebo pre počet prenesených bajtov.

```
#Version: 1.0
#Date: 21-Feb-2022
#Fields: c-ip time cs-method cs-uri sc-status bytes
109.230.38.133 11:58:02 GET /profile.php?new_message=1 200 772
103.152.17.191 11:58:02 POST /register.php 200 19416
213.139.193.232 11:58:03 GET /forum/viewforum.php?forum_id=5 200 22169
```

Obr. 2.3: Príklad záznamov vo formáte W3C Extended Log File Format. Je možné si všimnúť, že správnym nastavením smernice *Fields* možno ukladať rovnaké informácie ako v prípade Common Log formátu 2.4.

- **Combined Log Format** – Ďalší logovací formát s pevným počtom ukladaných polí, ktorý rozširuje vyššie predstavený *Common Log Format 2.4* o ďalšie dve polia. Informácie uložené v týchto pridaných poliach predstavujú užitočný zdroj informácií, ktoré možno aplikovať vo fáze predspracovania dát, ktorej sa venuje nasledujúca kapitola 3.

8. *Odkazovateľ* – Identifikátor webovej stránky, ktorá klienta odkazuje na aktuálnu webovú stránku [7]. Informácie z tohto poľa možno okrem iného využiť pre marketingovú analýzu prístupov na stránku na základe informácie obsiahnutej v tomto poli.
9. *Užívateľský agent* – Obsah tohto poľa poskytuje informácie o operačnom systéme, prehliadači a v niektorých prípadoch možno detekovať aj zariadenie, ktoré využívajú užívatelia webových aplikácií. Logovanie tejto informácie ponúka skvelú príležitosť pre identifikáciu užívateľov a detekciu botov či takzvaných *crawlerov*, ktorej sa venuje ďalšia kapitola tejto práce.

Vzdialený hositeľ	Identifikátor klienta	Užívateľské meno	Dátum a čas požiadavky	HTTP požiadavka	Stavový kód	Objem prenosu	Odkazovateľ	Užívateľský agent
109.230.38.133	-	-	[21/Feb/2022:08:38:51+0100]	"GET / HTTP/1.0"	302	-	"https://www.google.com/"	"Mozilla/5.0 (Linux; Android 11; SM-A202F) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.101 Mobile Safari/537.36"
109.230.38.133	-	-	[21/Feb/2022:08:38:52+0100]	"GET /news.php HTTP/1.0"	200	22527	"https://www.google.com/"	"Mozilla/5.0 (Linux; Android 11; SM-A202F) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.101 Mobile Safari/537.36"
109.230.38.133	-	-	[21/Feb/2022:05:38:52+0100]	"GET /themes/styles.css HTTP/1.0"	200	10183	"https://www.priklad1.sk/news.php"	"Mozilla/5.0 (Linux; Android 11; SM-A202F) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.101 Mobile Safari/537.36"

Obr. 2.4: Príklady záznamov vo formáte Combined Log Format.

Zhodnotenie formátov prístupových logov

V tejto kapitole boli detailne predstavené najznámejšie a najvyužívanejšie formáty webových prístupových logov. Za zmienku stojí aj ďalší štandardizovaný formát s pevným počtom pätnástich polí *IIS Log File Format*, ktorý navyše ukladá napríklad informácie o serveri či o samotnej službe [19].

Kapitola 3

Predspracovanie dát

Dáta v podobe v akej boli predstavené v predchádzajúcej kapitole môžu poslúžiť ako vhodný zdroj pre rôzne štatistiky, ktoré môžu sledovať napríklad časové vyťaženie webových stránok, najnavštevovanejšie stránky či geolokalizáciu užívateľov. Avšak pre získanie znalostí o správaní užívateľov a využívaní webových služieb je potrebné vstupnú množinu dát zbaviť nepotrebných záznamov a zároveň vykonať transformáciu dát do podoby, ktorá je vhodná pre objavovanie vzorov užívateľských aktivít. V tejto kapitole sú predstavené základné techniky a činnosti zamerané na predspracovanie webových prístupových logov.

3.1 Extrakcia a čistenie dát

- *Odstránenie nevyužívaných polí a hodnôt* – Polia *identifikátor klienta* a *užívateľské meno* obsahujú vo väčšine prípadov prázdnu hodnotu, a tak nie je potrebné ich ďalej uchovávať. Vychádzajúc z faktov, že protokol HTTP je bezstavový, kedy každá požiadavka na webový server predstavuje izolovanú udalosť a vzhľadom k tomu, že prístup k webovým stránkam je vo väčšine prípadov anonymný vedie k potrebe využitia inej techniky určenej k identifikácii užívateľov [19].
- *Transformácia dátumu a času na časové razítko* – Dátumy a časy jednotlivých záznamov sú v logovacích súboroch ukladané v podobe reťazcov. Extrakcia a následná transformácia do podoby časového razítka je potrebná pre rozdelenie užívateľských vstupov do užívateľských sedení.
- *Rozdelenie HTTP požiadavky* – Pole HTTP požiadavky, ktorej formát bol špecifikovaný v predchádzajúcej podkapitole 2.4 je potrebné rozdeliť na časť obsahujúcu HTTP metódu a na časť obsahujúcu URL adresu obsahujúcu cestu k vyžiadanému dokumentu. Ostatná časť, špecifikujúca verziu HTTP protokolu nemá ďalšie využitie, a tak nie je potrebné túto časť ďalej uchovávať.
- *Odstránenie žiadostí s grafickými, štýlovacími či inými nepotrebnými súbormi* – Na základe predchádzajúceho bodu je z URL adresy obsahujúcej vyžiadaný dokument získaná koncovka tohto dokumentu, ktorá definuje typ súboru. Následne je potrebné zbaviť sa záznamov obsahujúcich grafické, štýlovacie či iné nepotrebné dokumenty, ktoré majú vysokú početnosť výskytu a zároveň neposkytujú žiadne informácie o správaní užívateľov webových aplikácií [27].

- *Odstránenie záznamov s chybovým stavovým kódom* – Na základe poľa stavového kódu sú odfiltrované všetky záznamy, ktorých hodnota je nižšia ako 200 a zároveň vyššia ako 299. Po vykonaní tohto kroku sú ďalej uchovávané len záznamy s úspešnou odpoveďou. Toto pole nie je ďalej potrebné uchovávať.
- *Odstránenie záznamov na základe HTTP metódy* – Pomocou obsahu poľa HTTP metódy sú ponechané záznamy s hodnotou tohto poľa GET a POST. Metóda GET informuje o získaní informácií o užívateľovi a metóda POST sa najčastejšie využíva na aktualizáciu webovej stránky [14].
- *Detekcia a odstránenie neľudského správania* – *Weboví roboti*, často prezývaní ako *weboví crawleri* sú automatizované programy, ktoré prechádzajú a extrahujú webové stránky a sú súčasťou mnohých webových vyhľadávačov. Keďže správanie webového robota sa nezhoduje s bežným užívateľským správaním, tak je potrebné nájsť spôsob ako detekovať takéto programy a odstrániť nimi vytvorené logovacie záznamy. Najznámejší weboví crawleri sú zvyčajne detekovaní vďaka porovnaniu užívateľského agenta so zoznamom známych crawlerov. Ďalším využívaným postupom je kontrola prvého vyžiadaného dokumentu užívateľských sedení. Ak je prvým vyžiadaným dokumentom URL adresa */robots.txt*, tak zvyčajne ide o podozrivé správanie a dané užívateľské sedenie je následne potrebné odstrániť [16].


3.2 Identifikácia užívateľov

Po zbavení sa chybových záznamov a pre ďalšie predspracovanie nepotrebných dát v predchádzajúcej podkapitole je ďalej potrebné rozlíšiť jednotlivé záznamy a identifikovať užívateľov. Získavanie znalostí z webových prístupových logov nevyžaduje identifikáciu identity užívateľov, avšak je potrebné využitím správnych techník rozlíšiť samotných užívateľov.

Identifikácia užívateľov pomocou IP Adresy a užívateľského agenta

Najčastejšie využívaná technika pre rozdelenie logovacích záznamov medzi unikátnych užívateľov sa vykonáva prostredníctvom kombinácie IP adresy obsiahnutej v poli vzdialeného hostiteľa a užívateľským agentom, ktorý špecifikuje zariadenie užívateľa. Pomocou tejto techniky je možné rozlíšiť viacerých užívateľov s rovnakou IP adresou. Toto rozdelenie ignoruje fakt, že užívateľ pripojený na rovnaké sieťové zariadenie môže prechádzať webové stránky z viacerých rozdielnych prehliadačov alebo zariadení.

Časové razítko	Vzdialený hostiteľ	URL	Odkazovateľ	Užívateľský agent
110	109.230.38.133	/index.html	-	A
113	109.230.38.133	/stranka1.html	-	B
116	109.230.38.133	/stranka3.html	/index.html	A
117	147.225.16.18	/stranka2.html	-	D
117	147.225.16.18	/stranka4.html	/stranka2.html	D
118	107.243.85.90	/stranka1.html	/stranka3.html	E
124	147.225.16.18	/stranka5.html	/stranka1.html	F
127	107.243.85.90	/stranka4.html	/stranka1.html	E



Užívateľ	Časové razítko	Vzdialený hostiteľ	URL	Odkazovateľ	Užívateľský agent
1	110	109.230.38.133	/index.html	-	A
2	113	109.230.38.133	/stranka1.html	-	B
1	116	109.230.38.133	/stranka3.html	/index.html	A
3	117	147.225.16.18	/stranka2.html	-	D
3	117	147.225.16.18	/stranka4.html	/stranka2.html	D
4	118	107.243.85.90	/stranka1.html	/stranka3.html	E
5	124	147.225.16.18	/stranka5.html	/stranka1.html	F
4	127	107.243.85.90	/stranka4.html	/stranka1.html	E

Obr. 3.1: Grafické znázornenie konceptu identifikácie užívateľov pomocou tejto techniky.


Identifikácia užívateľov pomocou Cookies

Cookie je malé množstvo dát, ktoré webový server posieľa klientovi pri prvej žiadosti o webovú stránku. Tieto dáta sú následne uložené prehliadačom na strane klienta. Informácie uložené v cookie môžu obsahovať informácie týkajúce sa užívateľa, takže je možné ich využitím jednoznačne a správne identifikovať užívateľov [27]. Na druhej strane nie všetky stránky využívajú súbory cookie, a to z dôvodu ochrany osobných údajov. Užívatelia majú navyše možnosť súbory cookie vymazať.

Identifikácia užívateľov pomocou IP Adresy

Identifikácia užívateľov pomocou IP adres sa využíva v prípade prístupových logov vo formáte *Common log formát 2.4* ak nie je možné využiť súbory *cookies* a zároveň chýbajú informácie o užívateľských agentoch. Využitie samotných IP adres nepostačuje pre spoľahlivé rozlíšenie užívateľov webových aplikácií, a to zapríčiňuje, že táto technika je značne nepresná. Nejednoznačnosť využitia IP adres pre identifikáciu užívateľov je spôsobená rozšírenosťou proxy serverov, ktoré vo veľkej miere využívajú poskytovatelia internetových služieb, ktorí dynamicky pridelujú obmedzený rozsah IP adres medzi svojich klientov [16].


Časové razítko	Vzdialený hosťiteľ	URL
110	109.230.38.133	/index.html
113	109.230.38.133	/stranka1.html
116	109.230.38.133	/stranka3.html
117	147.225.16.18	/stranka2.html
117	147.225.16.18	/stranka4.html
118	107.243.85.90	/stranka1.html
124	147.225.16.18	/stranka5.html
127	107.243.85.90	/stranka4.html



Užívateľ	Časové razítko	Vzdialený hosťiteľ	URL
1	110	109.230.38.133	/index.html
1	113	109.230.38.133	/stranka1.html
1	116	109.230.38.133	/stranka3.html
2	117	147.225.16.18	/stranka2.html
2	117	147.225.16.18	/stranka4.html
3	118	107.243.85.90	/stranka1.html
2	124	147.225.16.18	/stranka5.html
3	127	107.243.85.90	/stranka4.html

Obr. 3.2: Grafické znázornenie identifikácie užívateľov pomocou IP adresy.

Časové razítko	Vzdialený hosťiteľ	URL	Odkazovateľ	Užívateľský agent
110	109.230.38.133	/index.html	-	A
113	109.230.38.133	/stranka1.html	-	B
116	109.230.38.133	/stranka3.html	/index.html	A
117	147.225.16.18	/stranka2.html	-	D
117	147.225.16.18	/stranka4.html	/stranka2.html	D
118	107.243.85.90	/stranka1.html	/stranka3.html	E
124	147.225.16.18	/stranka5.html	/stranka1.html	F
127	107.243.85.90	/stranka4.html	/stranka1.html	E



Užívateľ	Časové razítko	Vzdialený hosťiteľ	URL	Odkazovateľ	Užívateľský agent
1	110	109.230.38.133	/index.html	-	A
2	113	109.230.38.133	/stranka1.html	-	B
1	116	109.230.38.133	/stranka3.html	/index.html	A
3	117	147.225.16.18	/stranka2.html	-	D
3	117	147.225.16.18	/stranka4.html	/stranka2.html	D
4	118	107.243.85.90	/stranka1.html	/stranka3.html	E
5	124	147.225.16.18	/stranka5.html	/stranka1.html	F
4	127	107.243.85.90	/stranka4.html	/stranka1.html	E

IP adresa

Užívateľ	Časové razítko	Vzdialený hosťiteľ	URL
1	110	109.230.38.133	/index.html
1	113	109.230.38.133	/stranka1.html
1	116	109.230.38.133	/stranka3.html
2	117	147.225.16.18	/stranka2.html
2	117	147.225.16.18	/stranka4.html
3	118	107.243.85.90	/stranka1.html
2	124	147.225.16.18	/stranka5.html
3	127	107.243.85.90	/stranka4.html

Obr. 3.3: Porovnanie identifikácie užívateľov pomocou predstavených techník.

3.3 Rozdelenie záznamov do užívateľských sedení

Po identifikovaní jednotlivých užívateľov je ďalším krokom predspracovania rozdelenie užívateľských prístupov do sedení. *Užívateľské sedenie* možno chápať ako množinu webových stránok navštívených určitým užívateľom s určitým cieľom [19]. Rovnako ako pri identifikácii užívateľov aj tu možno využiť mechanizmus cookies. Avšak tento mechanizmus má vyššie ozrejmene obmedzenia čo spôsobuje, že táto technika býva často v praxi nahradená inými heuristickými technikami.

Rozdelenie na základe časového prahu

Technika rozdelenia užívateľských záznamov na základe časového prahu využíva časový rozdiel medzi každými dvomi po sebe nasledujúcimi užívateľskými požiadavkami. Ak časový odstup medzi požiadavkami presiahne zvolený prah, tak je vytvorené nové užívateľské sedenie. V opačnom prípade je záznam priradený k aktuálnemu sedeniu. Hodnota prahu sa v praxi využíva v rozmedzí 25 minút až 24 hodín.

Užívateľ	Časové razítko	Vzdialený hositeľ	URL	Odkazovateľ	Užívateľský agent
1	00:01:10	109.230.38.133	/index.html	-	A
1	00:01:23	109.230.38.133	/stranka1.html	/index.html	A
1	00:02:07	109.230.38.133	/stranka3.html	/stranka1.html	A
1	00:12:37	109.230.38.133	/stranka2.html	/stranka3.html	A
1	00:13:01	109.230.38.133	/stranka4.html	/stranka2.html	A
1	00:55:12	109.230.38.133	/index.html	/stranka4.html	A
1	00:55:37	109.230.38.133	/stranka5.html	/index.html	A
1	00:56:12	109.230.38.133	/stranka4.html	/stranka5.html	A
1	01:38:14	109.230.38.133	/stranka5.html	/stranka4.html	A

Užívateľské sedenie	Užívateľ	Časové razítko	Vzdialený hositeľ	URL	Odkazovateľ	Užívateľský agent
1	1	00:01:10	109.230.38.133	/index.html	-	A
1	1	00:01:23	109.230.38.133	/stranka1.html	/index.html	A
1	1	00:02:07	109.230.38.133	/stranka3.html	/stranka1.html	A
1	1	00:12:37	109.230.38.133	/stranka2.html	/stranka3.html	A
1	1	00:13:01	109.230.38.133	/stranka4.html	/stranka2.html	A
2	1	00:55:12	109.230.38.133	/index.html	/stranka4.html	A
2	1	00:55:37	109.230.38.133	/stranka5.html	/index.html	A
2	1	00:56:12	109.230.38.133	/stranka4.html	/stranka5.html	A
3	1	01:38:14	109.230.38.133	/stranka5.html	/stranka4.html	A

Obr. 3.4: Rozdelenie užívateľských záznamov do sedení. Zvolený prah 30 minút. Formát časového razítka je pre lepšie znázornenie ukážky *hh:mm:ss*. Hodnota užívateľského agenta je nahradená hodnotou *A* pre lepšie grafické zobrazenie.

Rozdelenie na základe času stráveného prehliadaním stránky

Táto technika rozdeľuje webové stránky do dvoch skupín na základe času strávenom na jednotlivých stránkach:

- *Navigačné stránky* – slúžia výhradne len pre navigačné účely a pomáhajú užívateľom dostať sa na informačné stránky.
- *Informačné stránky* – táto kategória stránok reprezentuje stránky, o ktoré majú užívatelia reálny záujem a trávajú na nich viac času ako v prípade navigačných stránok.

Využitie tejto techniky vyžaduje poznatky o analyzovanej webovej stránke pre kategorizáciu jednotlivých stránok. Ak je známy podiel zastúpenia navigačných a informačných stránok v predspracovanom dátovom rámci, tak prah definujúci maximálny čas strávený na navigačnej stránke možno vypočítať ako [27]:

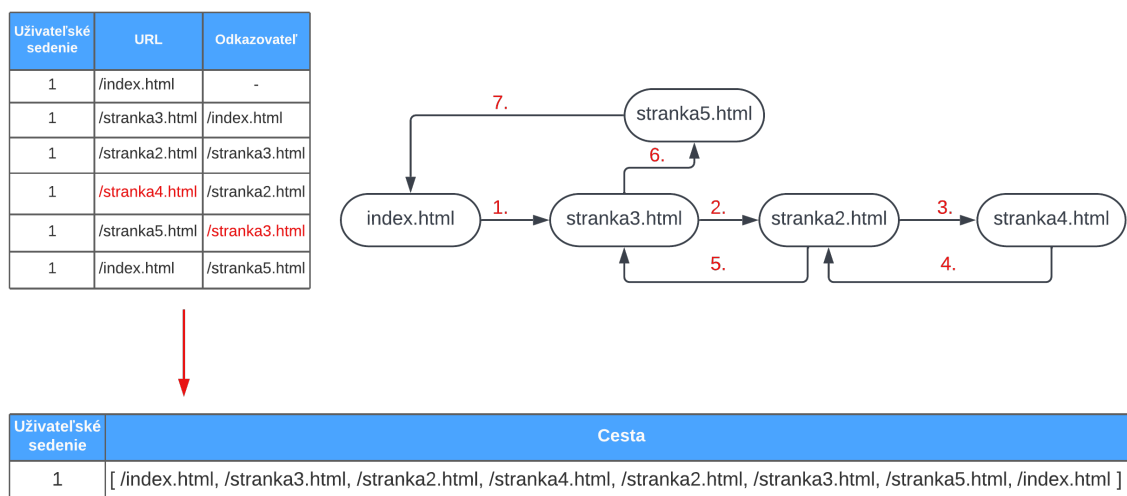
$$Prah = - \frac{\ln \left(1 - \frac{\text{počet navigačných stránok}}{\text{celkový počet stránok}} \right)}{\text{stredná hodnota doby strávenej prezeraním stránok}} \quad (3.1)$$

3.4 Skompletizovanie ciest užívateľských sedení

Po rozdelení užívateľských záznamov do užívateľských sedení je poslednou štandardne vykonávanou činnosťou predspracovania dát *skompletizovanie ciest užívateľských sedení*. Hlavným cieľom tejto finálnej fázy predspracovania dát je identifikácia a doplnenie chýbajúcich prístupov na stránky. Ide o veľmi významnú úlohu v predspracovaní dát vzhľadom na fakt, že chýbajúce záznamy menia užívateľské správanie, a tak môžu významne ovplyvniť získané výsledky. Niektoré užívateľské prístupy nie sú zaznamenané v dôsledku využívania vyrovnávajúcej pamäte. Tento problém sa vyskytuje v prípadoch prístupu na stránku s použitím tlačidla späť, kedy webový prehliadač namiesto zaslania požiadavky na server vráti klientovi kópiu vyžiadanej stránky, ktorá je už uložená vo vyrovnávajúcej pamäti po predchádzajúcej požiadavke [27].

Identifikácia a doplnenie chýbajúcich prístupov na stránky

Pre identifikáciu chýbajúcich prístupov je potrebné mať k dispozícii pole *odkazovateľ* 2.4. Toto pole možno využiť v kombinácii so znalosťou topológie webovej stránky pre presnejšiu detekciu chýbajúcich odkazov na základe znalosti odkazov vedúcich z danej stránky. Identifikačná činnosť prebieha na základe porovnávania hodnôt medzi URL adresou predchádzajúceho záznamu a hodnoty poľa odkazovateľ aktuálneho záznamu. V prípade nerovnosti medzi týmito dvoma hodnotami sa vykoná spätná kontrola predchádzajúcich navštívených stránok užívateľského sedenia. Ak sa aktuálna hodnota poľa odkazovateľ nachádza v histórii cesty aktuálneho užívateľského sedenia, tak sa predpokladá, že užívateľ využil tlačidlo späť a je potrebné doplniť cestu medzi aktuálnou URL adresou a odkazovateľom [25].



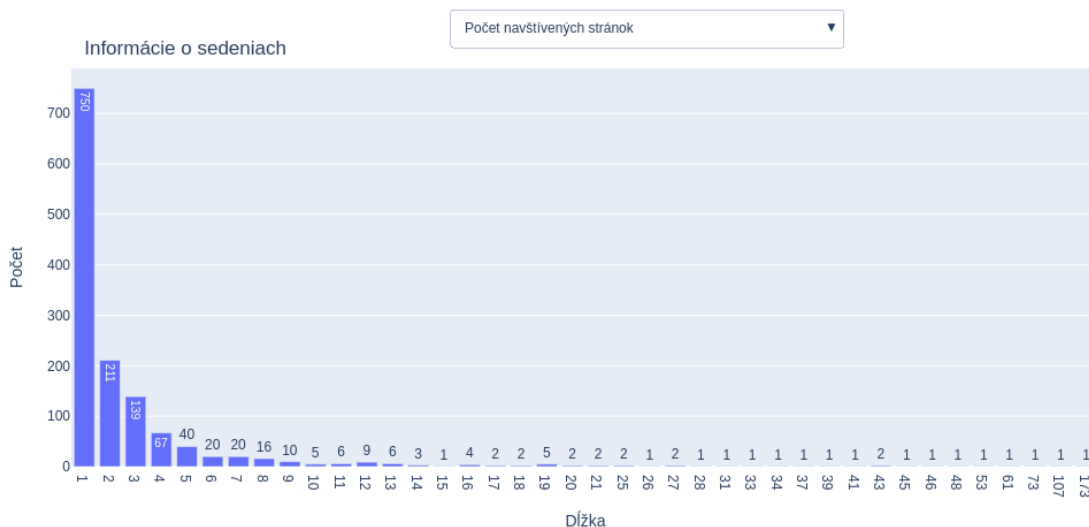
Obr. 3.5: Príklad identifikácie a doplnenia chýbajúcich záznamov do cesty užívateľského sedenia. Po získaní užívateľskej cesty je vykonávaná transformácia formátu dátového rámca. V ukážke chýbajú niektoré atribúty, ktoré sú ukladané pre transformovaný dátový rámec. Chýbajúce atribúty sú bližšie popísané v nasledujúcej sekcii 3.5.

3.5 Prieskumná analýza užívateľských sedení

V predchádzajúcich podkapitolách boli predstavené základné techniky a činnosti vykonávané pre predspracovanie webových prístupových logov. Úlohou prieskumnej analýzy užívateľských sedení je získanie prieskumných informácií o používaní webu užívateľmi. Tieto informácie sú následne využívané k identifikácii zaujímavých podmnožín užívateľských sedení 4.1. Rovnako poskytujú možnosť sledovania vzájomných vzťahov medzi sledovanými premennými. Informácie získané v tejto fáze môžu poslúžiť ako vhodná forma spätnej väzby pre vývojárov webových aplikácií. V praxi sa táto časť predspracovania vykonáva súbežne so skompletizovaním ciest užívateľských sedení. Poznatky prezentované v tejto kapitole vychádzajú z [19] a sú doplnené o grafické ukážky z implementačnej časti tejto práce.

Počet navštívených stránok

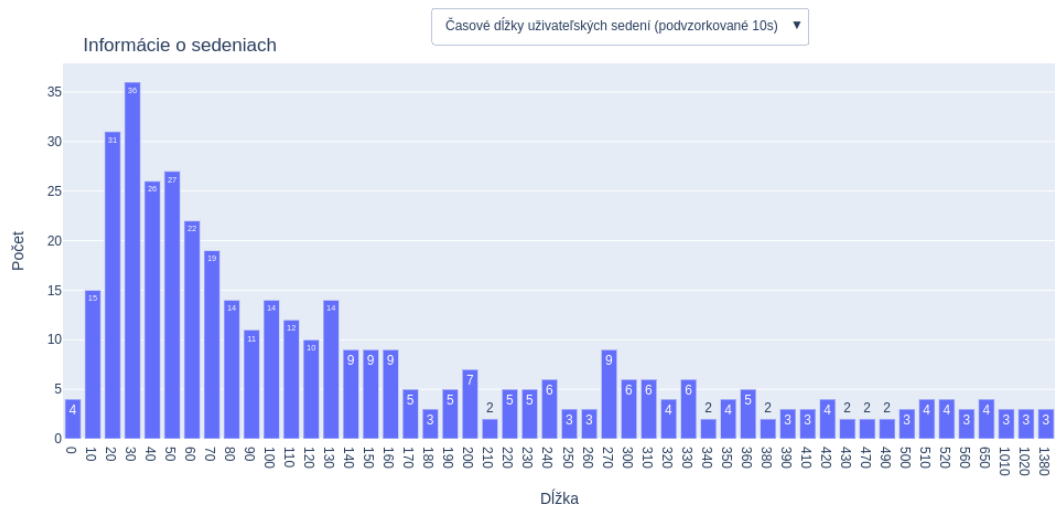
Prvým faktorom, ktorý môže poskytnúť vývojárom webových služieb spätnú väzbu o kvalite navrhnutých webových stránok je sledovanie počtu navštívených stránok počas užívateľských sedení. Z globálneho hľadiska dĺžka väčšiny užívateľských sedení nepresahuje viac ako 10 užívateľských akcií. Hodnotu tejto premennej ovplyvňuje stratégia predspracovania pri detekcii botov, ktorí systematicky prechádzajú celú webovú stránku, čo vedie k zvýšeniu priemernej dĺžky navštívených stránok počas užívateľského sedenia.



Obr. 3.6: Príklad agregovanej štatistiky počtu navštívených akcií užívateľských sedení.

Dĺžka užívateľských sedení

Ďalšou sledovanou premennou je čas, ktorý užívatelia strávili na webovej stránke počas užívateľských sedení. Pri výpočte tejto premennej sú do výpočtu zahrnuté iba sedenia s minimálnym počtom navštívených akcií 2, a to vzhľadom na fakt, že nie je možné presne určiť čas strávený prezeraním poslednej stránky sedenia. Hodnota dĺžky užívateľského sedenia je získaná odčítaním časového razítka prvej akcie sedenia od časového razítka poslednej akcie. Táto metóda výpočtu trvania užívateľských sedení vedie k podhodnoteniu celkového trvania z dôvodu nezahrnutia času stráveného na poslednej stránke užívateľského sedenia.



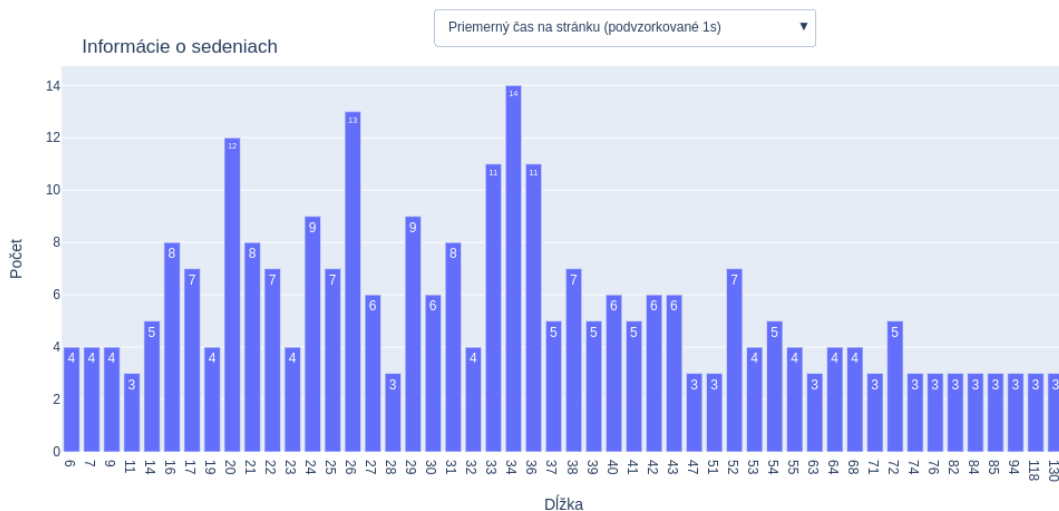
Obr. 3.7: Príklad agregovanej štatistiky dĺžky užívateľských sedení. Pre lepšie grafické znázornenie sú hodnoty podvzorkované na desiatky sekúnd.

Priemerný čas strávený na stránke

Po získaní informácií o počte navštívených stránok a dĺžkach užívateľských sedení je možné odvodiť priemerný čas strávený na stránke pomocou vzorca:

$$\text{priemerný čas} = \frac{\text{dĺžka užívateľského sedenia}}{\text{počet navštívených akcií počas sedenia} - 1} \quad (3.2)$$

Rovnako ako v prípade výpočtu dĺžky užívateľských sedení sú aj pre tento výpočet zahrnuté iba užívateľské sedenia s viac ako jednou navštívenou akciou. Užívateľské sedenia s nízkym priemerným časom na stránku a zároveň s vysokým počtom navštívených akcií pôsobia podozrivo a môže ísť o botov, ktorých sa nepodarilo detekovať vo fáze čistenia dát 3.1.



Obr. 3.8: Príklad agregovanej štatistiky priemerného času stráveného na stránkach. Pre lepšie grafické znázornenie sú hodnoty podvzorkované na celé sekundy.

Kapitola 4

Dolovanie znalostí

Po úprave a transformácii získaných dát do podoby predstavenej v kapitole 3 sú v tejto kapitole detailnejšie rozobrané základné techniky pre získavanie znalostí z webových logov. Niektoré analytické techniky pre získavanie znalostí vyžadujú dodatočnú úpravu či transformáciu formátu dát. Podkapitola 4.1 sa zaoberá rozdelením množiny užívateľských sedení do podmnožín. Rovnako sú v tejto podkapitole predstavené základné kategórie metód pre zhľukovanie. V podkapitole 4.2.1 je popísaná transformácia ciest užívateľských sedení na transakcie a zároveň je predstavená činnosť známych algoritmov pre hľadanie frekventovaných položiek. Následne je v podkapitole 4.2.2 vysvetlená problematika dolovania asociačných pravidiel. Posledná podkapitola 4.3 rieši problematiku dolovania sekvenčných vzorov.

4.1 Zhľuková analýza

Zhľuková analýza sa zaoberá procesom rozdelenia množiny dátových objektov do vhodných podmnožín [11]. Každá výsledná podmnožina predstavuje zhľuk. Objekty zaradené zhľukovaním do rovnakého zhľuku sú si navzájom podobné, a zároveň sú nepodobné objektom, ktoré náležia iným zhľukom. Rôzne algoritmy pre zhľukovanie môžu vytvárať rôzne zhľuky na rovnakej množine dátových objektov. Zhľukovanie je mimo iného vhodné k objaveniu doposiaľ neznámych skupín dát. Tento proces je navyše možné využiť ako nástroj pre získanie charakteristík jednotlivých zhľukov. Zhľukovanie je rovnako možné využiť pred aplikáciou iných dolovacích algoritmov pre získavanie znalostí, kedy je možné identifikované zaujímavé zhľuky využiť ako vstup pre tieto algoritmy.

Normalizácia hodnôt

Pre zlepšenie výkonnosti je často vykonávaním krokom normalizácia číselných premenných pre zhľukovanie [19]. Táto činnosť pomáha zabrániť problému, ktorý zapríčiňuje, že atribúty s veľkým rozsahom hodnôt prevážia nad atribútmi s menším rozsahom hodnôt [11]. Pre normalizáciu možno použiť napríklad *Min-max* normalizáciu, ktorá transformuje hodnoty zvolených atribútov číselných premenných do rozsahu $\langle 0, 1 \rangle$ pomocou vzorca:

$$v' = \frac{v - \min A}{\max A - \min A} \quad (4.1)$$

Vo vyššie uvedenej rovnici reprezentuje v číselnú premennú atribútu A určeného pre zhľukovanie, $\max A$ predstavuje maximálnu hodnotu tohto atribútu, $\min A$ predstavuje najmenšiu hodnotu tohto atribútu a v' predstavuje normalizovanú číselnú premennú atribútu A .

Zhlukovanie založené na rozdelení

K-Means predstavuje najznámejší a najvyužívanejší algoritmus pre zhlukovanie. Tento algoritmus patrí do kategórie zhlukovacích metód založených na rozdelení. Metódy zhlukovania založené na rozdelení premiestňujú jednotlivé inštanacie objektov takým spôsobom, že ich presúvajú medzi jednotlivými zhlukmi [18]. Táto kategória metód zhlukovania zvyčajne vyžaduje, aby bol počet zhlukov dopredu určený užívateľom. Algoritmus k-means rozdeľuje vstupnú množinu dát do K zhlukov (C_1, C_2, \dots, C_k), pričom jednotlivé zhluky sú reprezentované stredmi samotných zhlukov, alebo prípadne strednými hodnotami. Počiatočná množina stredov zhlukov býva zvolená náhodne, prípadne sa využívajú rôzne heuristické metódy. Objekty vstupnej množiny dát sú každou iteráciou tohto algoritmu priradené k najbližšiemu stredu zhluku na základe euklidovskej vzdialenosti. Po rozdelení objektov sú prepočítané nové stredy zhlukov μ_k ako stredná hodnota všetkých objektov patriacich do totožného zhluku pomocou vzorca:

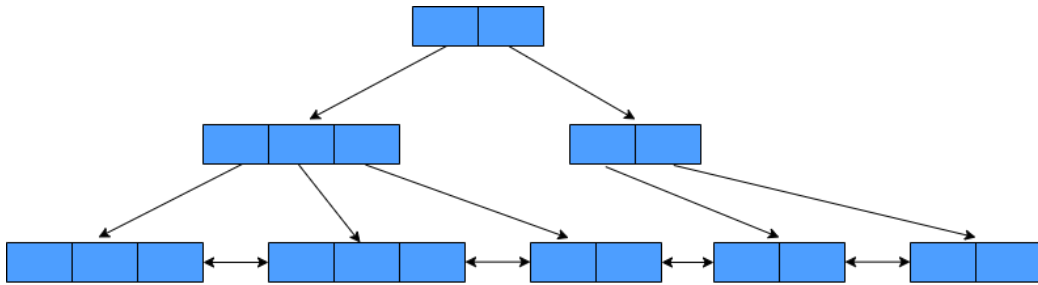
$$\mu_k = \frac{1}{N_k} \sum_{q=1}^{N_k} x_q \quad (4.2)$$

Vo vyššie uvedenom vzorci N_k predstavuje počet inštancií objektov patriacich do zhluku k . Iterácie tohto algoritmu, popísané vyššie sú vykonávané až pokiaľ nie je uspokojená ukončujúca podmienka. Hľadanie optimálneho rozdelenia objektov do zhluku sa môže zastaviť napríklad v prípade prekročenia užívateľom špecifikovaného počtu iterácií, alebo v prípade ak sa chyba rozdelenia nezníži zmenou stredov zhlukov. Hlavnou prednosťou tohto algoritmu je jeho lineárna zložitosť. Pri I iteráciách, na vzorke m objektov s N atribútmi dosahuje zložitosť pri rozdelení vstupnej množiny na k zhlukov $O(I \times k \times m \times N)$. Medzi problémy tohto algoritmu patrí citlivosť na zašumené dáta a odľahlé hodnoty. Ďalším problémom je výber počiatočného rozdelenia vzhľadom na citlivosť tohto algoritmu. Algoritmus rovnako vyžaduje dopredu špecifikovaný počet vyžiadaných zhlukov.

Hierarchické zhlukovanie

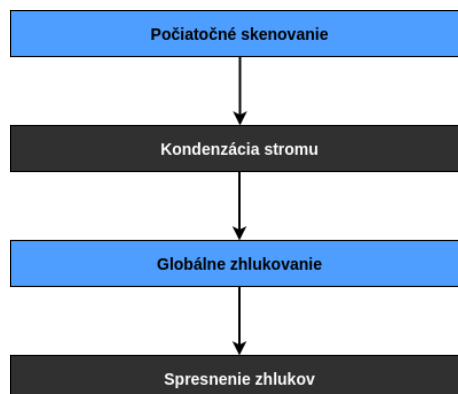
Hierarchické zhlukovacie metódy okrem rozdelenia vstupnej množiny dátových objektov na podmnožiny umožňujú rozdelenie vstupnej množiny do rôznych hierarchických úrovní [11]. Jedným z najpopulárnejších algoritmov zhlukovacích metód založených na hierarchickom zhlukovaní je *Balanced Iterative Reducing and Clustering* (BIRCH), ktorý zovšeobecňuje prístup algoritmu *k-means* k procesu zhlukovania [3]. Hlavnou výhodou využitia tohto algoritmu je jeho vstupno-výstupná efektivita vzhľadom na obmedzené množstvo dostupnej pamäte. Algoritmus využíva dátovú štruktúru *clustering feature* (*CF*), ďalej len *CF*, v kombinácii s *CF-stromom*. *CF* predstavuje stručný prehľad každého zhluku, pričom tento koncept využíva fakt, že nie každý dátový bod je rovnako dôležitý pre zhlukovanie a tak nie je nutné uchovávať všetky dátové body. Táto činnosť vedie k zvýšeniu pamätovej efektivity. *CF* je trojica $\langle N, LS, SS \rangle$, kde N predstavuje počet dátových bodov v zhluku, LS možno chápať ako lineárny súčet dátových bodov v zhluku a SS predstavuje štvorcový súčet dátových bodov v zhluku. Zlúčenie zhlukov prebieha súčtom *CF* dvoch pôvodných zhlukov. Pri zlúčení nie je nutné pristúpiť k jednotlivým objektom zhlukov. *CF-strom* je výškovo vyvážený strom, pričom každý uzol v strome predstavuje zhluk, ktorý sa skladá z najviac B podzhlukov. Parameter B označuje vyvažovací faktor. Okrem tohto parametru algoritmus využíva prah T . Priemery všetkých podzhlukov sú v listových uzloch horne ohraničené

prahom T . Hodnota prahu ovplyvňuje výšku stromu, pričom čím vyššia je hodnota tohto parametru, tým nižšia je výška CF-stromu. Listové uzly stromu sú vzájomne zretazené.



Obr. 4.1: Grafická ukážka *CF-stromu*. Jednotlivé uzly stromu predstavujú zhluky, ktoré pozostávajú z B podzhlukov.

Činnosť algoritmu **BIRCH** pozostáva zo štyroch krokov, pričom dva z nich sú voliteľné. V prvom kroku sa pri skenovaní vstupných dátových bodov vytvorí *CF-strom*. Pri vytváraní stromu sú vstupné dátové body vkladané s hodnotou prahu nastavenou na nulu. Takto zvolená hodnota prahu spôsobuje, že každý dátový bod je chápaný ako samostatný zhluk v listovom uzle a zároveň tento fakt môže spôsobiť problém s nedostatkom pamäte. V takom prípade je potrebné zvýšiť hodnotu prahu a strom prestavať. Zoskúpením blízkych dátových bodov sa vytvorí počiatočná hierarchia vstupného súboru údajov, avšak počiatočná štruktúra zhlučovania nemusí byť dostatočne presná. Po tomto kroku nasleduje voliteľný krok kondenzácie stromu, ktorý slúži k spresneniu pôvodne v prvom kroku vytvoreného *CF-stromu*. V tomto kroku sa kondenzuje strom opakovaným vložením listových uzlov pôvodného stromu. Následne v kroku globálneho zhlučovania sa algoritmus pokúša zhlučovať všetky podzhluky v listových uzloch. Táto činnosť je vykonávaná spôsobom, kedy sa podzhluk s n dátovými bodmi n -krát prevedie na stred a potom sa využije aglomeratívny hierarchický zhlučovací algoritmus alebo prípadne iný existujúci zhlučovací algoritmus. Posledným a to voliteľným krokom algoritmu je *spresnenie zhlučkov*. V tomto kroku algoritmus opäť priradí všetky dátové body na základe stredových bodov zhlučkov, ktoré boli vytvorené v predchádzajúcom kroku tohto algoritmu.



Obr. 4.2: Kroky z ktorých pozostáva činnosť zhlučovacieho algoritmu BIRCH. Voliteľné kroky tohto hierarchického zhlučovacieho algoritmu sú znázornené s tmavým pozadím.

Rozšírenosť využívania tohto hierarchického zhlukovacieho algoritmu je zapríčinená efektívnou časovou zložitostou, dobrou škálovateľnosťou a výbornou kvalitou zhlukovania [11]. Pre N objektov môže *BIRCH* dosiahnuť až lineárnu časovú zložitosť $O(N)$.

Zhlukovanie založené na hustote

Doposiaľ predstavené zhlukovacie metódy dosahujú kvalitné výsledky pri hľadaní zhlukov guľovitého tvaru. Avšak tieto metódy môžu mať problém s identifikáciou zhlukov ľubovoľných tvarov. Príkladom takéhoto tvaru zhluku je oválny tvar, kedy tieto techniky nepresne identifikujú konvexné oblasti. Dôsledkom tohto problému je, že do jednotlivých zhlukov sú zahrnuté šumy a odľahlé hodnoty. Metódy zhlukovania založené na hustote modelujú zhluky ako husté oblasti v dátovom priestore, ktoré sú oddelené riedkymi oblasťami.

Prvým a zároveň najstarším predstaviteľom zhlukovacích metód založených na hustote je algoritmus **DBSCAN** [13]. Tento algoritmus vyžaduje dva vstupné parametre. Prvým z parametrov je ϵ , ktorý reprezentuje polomer okolia. Druhým parametrom je minimálny počet bodov v okolí jadrového bodu. *DBSCAN* bol navrhnutý s cieľom umožniť zhlukovanie dát ľubovoľných tvarov, pričom samotný algoritmus počíta s prítomnosťou šumu vo vstupnej množine dát. Pre každý objekt zhluku s polomerom ϵ musí kardinalita okolia tohto objektu prekročiť prah, ktorý predstavuje minimálny počet objektov v okolí daného objektu. Pre databázu objektov D je možné za predpokladu ak body $p, q \in D$ vyjadriť ϵ -okolie ľubovoľného bodu p ako:

$$N_{\epsilon p} = \{dist(p, q) < \epsilon\} \quad (4.3)$$

Ak okolie bodu spĺňa podmienku minimálneho počtu bodov, tak tento bod je nazývaný ako takzvaný *jadrový bod*. Okrem tohto typu bodov rozlišujú zhlukovacie metódy založené na hustote body na hraničné body a šumové body [3]. Hraničné body patria do zhluku, avšak nespĺňajú podmienku minimálneho počtu bodov v okolí. Šumové body naopak nepatria do žiadneho zhluku. Činnosť algoritmu *DBSCAN* pozostáva vo vyhľadávaní zhlukov pomocou kontroly okolia každého objektu vo vstupnej množine dát. Algoritmus vytvorí nový zhluk s objektom reprezentujúcim jadro zhluku v prípade, ak kardinalita okolia prekračuje prah minimálneho počtu bodov v okolí. Následne sa iteratívne zhromažďujú priamo hustotovo dosiahnuteľné objekty z jadra objektov. Táto činnosť môže zahŕňať proces zlúčenia nového hustotne dosiahnuteľného zhluku. Činnosť algoritmu je ukončená v prípade, ak nie je možné pridať nový objekt do žiadneho zhluku.

Výpočtová zložitosť algoritmu *DBSCAN* pre N objektov dosahuje v najhoršom prípade $O(N^2)$. Tento zhlukovací algoritmus založený na hustote je pri vhodnom nastavení parametrov účinný pri hľadaní zhlukov ľubovoľných tvarov.

4.2 Získavanie znalostí pomocou asociačných pravidiel

Problém dolovania asociačných pravidiel bol pôvodne definovaný v kontexte údajov o spotrebnom koši [1]. Primárnym cieľom tejto problematiky je určenie asociácií medzi rôznymi skupinami položiek. Najčastejšie používaný model pre dolovanie asociačných pravidiel využíva frekvenciu výskytov jednotlivých množín položiek pre kvantitatívne vyjadrenie úrovni asociácií. Aplikovateľnosť tejto problematiky má širokú škálu pôsobnosti, pričom dolovaním asociačných pravidiel pre získavanie znalostí z webových logov možno nájsť skryté závislosti pre cesty užívateľských sedení.

Transformácia ciest užívateľských sedení na transakcie

Výsledkom fázy predspracovania webových prístupových logov predstavených v kapitole 3 je množina n zobrazení stránok $P = \{p_1, p_2, \dots, p_n\}$ a rovnako množina m užívateľských transakcií $T = \{t_1, t_2, \dots, t_m\}$ [16]. Každá užívateľská transakcia $t_i \in T$ je podmnožinou množiny zobrazení stránok P . Zobrazenia stránok predstavujú sémanticky významné entity. Problém dolovania asociačných vzorov je vo svojej najprimitívnejšej forme definovaný v kontexte riedkych binárnych databáz [1]. V tomto prípade dátová matica obsahuje iba položky s binárnymi hodnotami 0/1.

Užívateľ	Užívateľské sedenie	Cesta
1	1	[/stranka1.html, /stranka2.html, /stranka4.html]
1	2	[/index.html, /stranka3.html, /stranka4.html]
2	3	[/index.html, /stranka1.html]
3	4	[/stranka1.html, /stranka5.html, /stranka3.html, /stranka1.html]

↓

/index.html	/stranka1.html	/stranka2.html	/stranka3.html	/stranka4.html	/stranka5.html
0	1	1	0	1	0
1	0	0	1	1	0
1	1	0	0	0	0
0	1	0	1	0	1

Obr. 4.3: Grafická ukážka transformácie ciest užívateľských sedení do transakčnej matice.

4.2.1 Dolovanie frekventovaných vzorov

Primárnou činnosťou, ktorou sa zaoberá problematika dolovania frekventovaných vzorov je hľadanie vzťahov medzi jednotlivými prvkami zo vstupnej databázy [2]. Táto problematika si prešla svojim vývojom a pre samotné dolovanie frekventovaných vzorov boli zadefinované rôzne rámcové riešenia. Najpoužívanejší je rámec založený na podpore. V tomto prípade sa hľadajú množiny položiek, ktorých frekvencia sa nachádza nad zvoleným prahom. Na dolovanie frekventovaných vzorov existuje niekoľko tried algoritmov. Veľa algoritmov využívaných na riešenie tejto problematiky sa líši iba v poradí, v akom sú frekventované vzory

skúmané. Metódy určené na dolovanie frekventovaných vzorov využívajú transakčnú databázu $\mathcal{T} = \{T_1, T_2, \dots, T_n\}$. Každá položka transakčnej databázy $T_i \in \mathcal{T}, \forall i = \{1, \dots, n\}$ pozostáva z množiny prvkov a možno ju vyjadriť ako $T_i = \{x_1, x_2, \dots, x_l\}$. Množina $P \subseteq T_i$ sa nazýva *súborom položiek* a zároveň platí, že veľkosť súboru položiek je definovaná ako počet prvkov, z ktorej súbor položiek pozostáva. Počet transakcií, ktoré obsahujú vzor P sa v problematike dolovania frekventovaných vzorov nazýva podpora vzoru P . Vzor P je považovaný za frekventovaný vzor v prípade, ak je jeho podpora väčšia alebo rovná minimálnej hodnote prahu, ktorý ohraničuje minimálnu hodnotu podpory. Získané frekventované vzory sú často využívané na generovanie asociačných pravidiel, ktorými sa bližšie zaoberá nasledujúca podkapitola 4.2.2.

Algoritmus Apriori

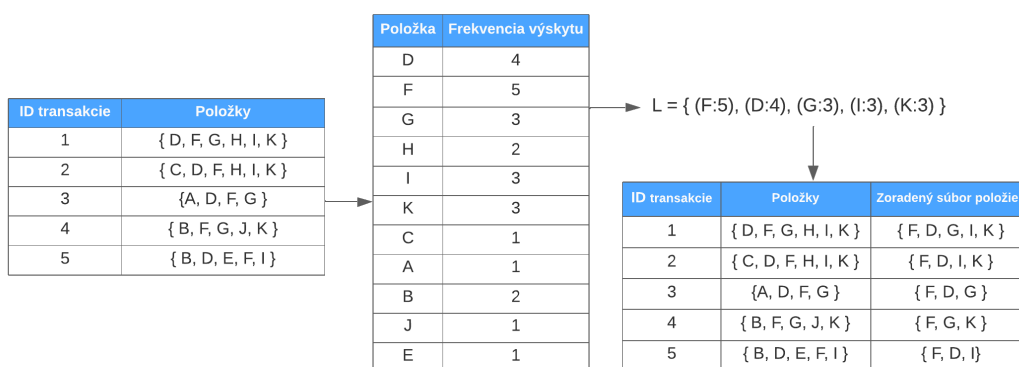
Apriori patrí do skupiny metód založených na spájaní. Takéto metódy generujú $(k + 1)$ kandidátov z k frekventovaných vzorov pomocou spájania. Následne sú vygenerovaní kandidáti overení v transakčnej databáze. Apriori využíva prístup založený na postupnom postupe po úrovniach. Pri tomto postupe sú najskôr vygenerované všetky súbory položiek dĺžky k , až následne sú vygenerované súbory položiek s dĺžkou $(k + 1)$. Algoritmus využíva princíp, kedy je každá podmnožina frekventovaného vzoru tiež považovaná za frekventovanú. Tento fakt vedie k tomu, že Apriori umožňuje generovanie kandidátov dĺžky $(k + 1)$ z už generovaných frekventovaných vzorov dĺžky k pomocou spájania. Princíp spojenia využívaný pre tento algoritmus možno definovať všetkými dvojicami frekventovaných k -vzorov, ktoré majú minimálne $(k - 1)$ spoločných položiek. Po vykonaní spojenia a získaní nového vzoru sa získaný vzor označuje ako kandidátsky vzor, a to z dôvodu, že nie je možné určiť, či daný vzor spĺňa minimálnu podporu. Preto je následne potrebné spočítať podporu kandidátskeho vzoru v transakčnej databáze. Pre zabránenie duplicit pri vytváraní kandidátskych vzorov je potrebné spájať len množiny položiek, ktoré majú prvých $(k - 1)$ položiek rovnakých na základe lexikografického usporiadania položiek. Tiež je veľmi dôležité, aby metóda počítania podpory bola maximálne efektívna a účinná. Niektoré kandidátske vzory je možné efektívne vyradiť bez overenia voči transakčnej databáze. Táto činnosť je vykonávaná overovaním všetkých podmnožín kandidátskych vzorov. Je známe, že dve z podmnožín, ktorých spojením bol vygenerovaný kandidátsky vzor sú frekventované, avšak je potrebné, aby všetky podmnožiny daného kandidátskeho vzoru boli frekventované. Ak nie sú všetky podmnožiny daného kandidátskeho vzoru frekventované, tak je možné vyradiť takýto kandidátsky vzor. Činnosť vyradenia kandidátskych vzorov bez overenia voči transakčnej databáze výrazne urýchluje tento algoritmus. Počiatočné generovanie vzorov dĺžky 1–2 je zvyčajne možné vykonávať pomocou špecializovaných metód. Algoritmus Apriori ako celok pozostáva z troch hlavných krokov, ktoré sa opakujú. Jedinou zmenou pre jednotlivé iterácie algoritmu je zvyšovanie hodnoty k , ktorá predstavuje dĺžku aktuálne generovaného vzoru. Činnosti vykonávané v jednotlivých krokoch algoritmu pozostávajú z:

1. Generovania kandidátskych vzorov pomocou spájania.
2. Vyradenia kandidátskych vzorov, ktorých podmnožiny nie sú frekventovanými vzormi.
3. Overenia kandidátskych vzorov voči transakčnej databáze.

Ukončujúcou podmienkou algoritmu je prázdna množina frekventovaných vzorov dĺžky k pre danú iteráciu algoritmu.

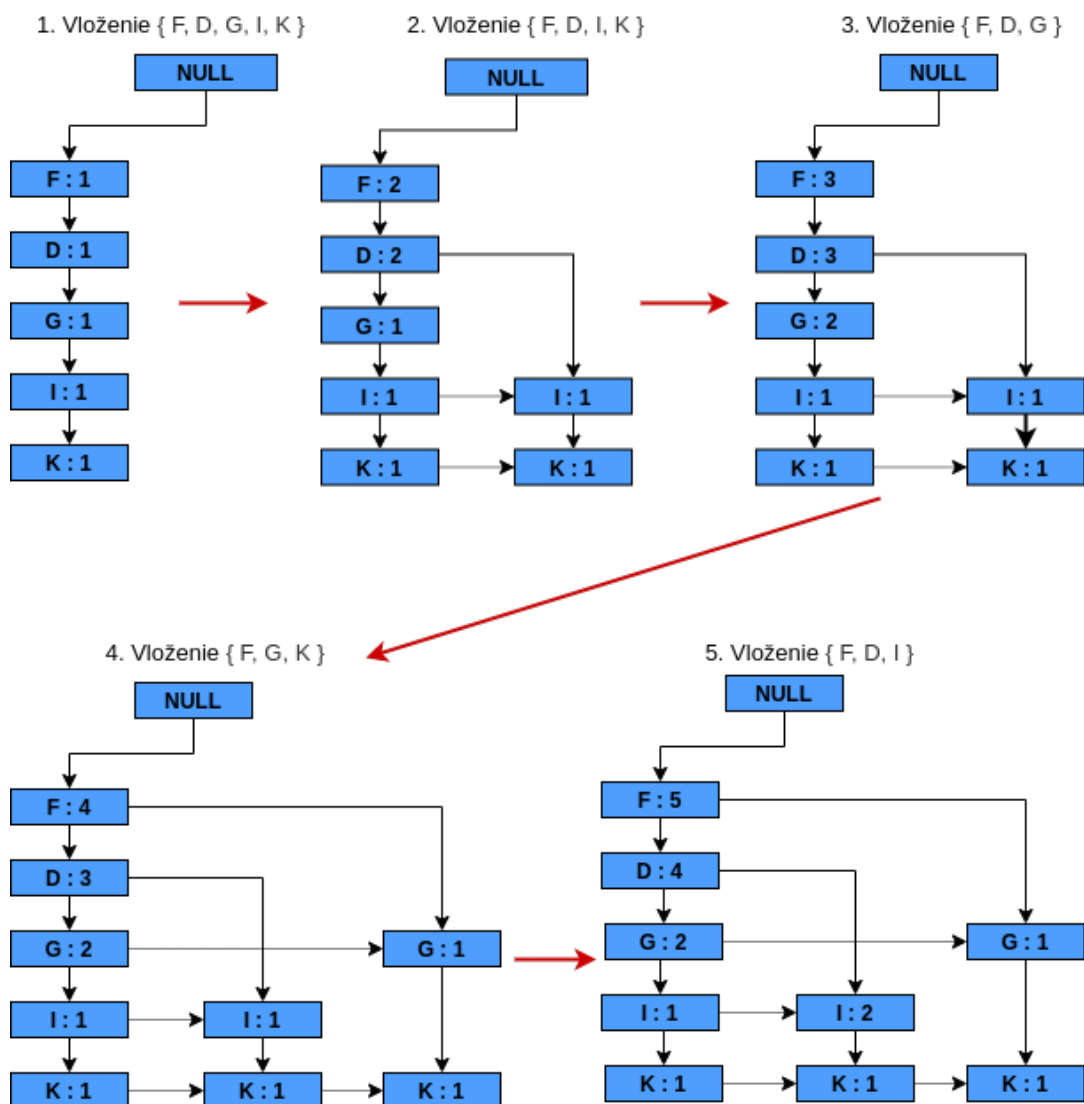
Algoritmus FP-Growth

Frequent Pattern Growth (FP-Growth) predstavuje ďalší algoritmus, ktorý je využívaný pre dolovanie frekventovaných vzorov, z ktorých možno následne získať asociačné pravidlá. Na rozdiel od vyššie predstaveného algoritmu *Apriori* nevyžaduje generovanie kandidátskych vzorov [17]. *FP-Growth* využíva špeciálnu dátovú štruktúru *FP-strom*, ktorý je vysoko komprimovaný, a tak znižuje pôvodnú transakčnú databázu. Výhodou tohto algoritmu je fakt, že *FP-Growth* prehladáva databázu iba dvakrát. Pri prvom priechode transakčnou databázou algoritmus vyhľadáva množinu frekventovaných položiek s dĺžkou 1, ktorých frekvencia výskytu presahuje prah určený minimálnou hodnotou podpory. Následne pri druhom priechode transakčnou databázou sú odfiltrované všetky položky, ktoré nespĺňajú podmienku pre frekventované položky.



Obr. 4.4: Príklad získania zoradeného súboru frekventovaných položiek. Hodnota prahu, ktorá označuje minimálnu hodnotu podpory pre týchto 5 transakcií predstavuje hodnotu 3. Výsledkom priechodov transakčnej databázy je zoradená množina frekventovaných položiek, z ktorej sú navyše odfiltrované všetky nefrekventované položky. Pre zjednodušenie grafickej ukážky sú stránky, z ktorých sa skladajú cesty užívateľských sedení nahradené písmenami.

Množina, ktorá je získaná po druhom priechode transakčnou databázou je zoradená vzostupne podľa frekvencie, a to z dôvodu, že každá cesta FP-stromom sa následne riadi poradím, v ktorom sú frekventované položky v tejto množine zoradené. Súčasne s druhým priechodom je vytvorený FP-strom, ktorého koreň sa označí ako *null*. Pri každom vložení sú súčty uzlov, ktoré sú ovplyvnené vložením inkrementované [2]. Ak aktuálne vkladaná transakcia zdieľa prefix s predtým vloženou transakciou, tak táto transakcia zdieľa cestu v strome až do konca spoločného prefixu. Pri zdieľaní cesty v strome je podpora zdieľaných uzlov inkrementovaná. V ceste za spoločným prefixom sú do stromu vložené nové uzly pre zostávajúce prvky vkladanej transakcie. Táto činnosť je ukončená po vložení všetkých zoradených súborov položiek. Následne sú vytvorené *bázy podmienených vzorov*. Táto činnosť sa vykonáva pomocou prehládavania FP-stromu. Do bázy podmienených vzorov sú uložené všetky prefixové cesty, ktoré vedú k danej frekventovanej položke. Z bázy podmienených vzorov sú prechádzané všetky položky vrátane samotnej položky a možných kombinácií. V tomto kroku je spočítaná podpora prechádzaných položiek a tiež je vygenerovaný podmienený FP-strom. Posledným krokom tohto algoritmu je spojenie podmieneného FP-stromu s danou frekventovanou položkou, čím sú získané množiny frekventovaných vzorov.



Položky	Báza podmienených vzorov	Podmienený FP-strom	Frekventované vzory
K	{{(FDGI:1), (FDI:1), (FG:1)}	{{(F:3)} K	{{(FK:3)}
I	{{(FDG:1), (FD:2)}	{{(F:3), (D:3), (FD:3)} I	{{(FI:3), (DI:3), (FDI:3)}
G	{{(FD:2), (F:1)}	{{(F:3)} G	{{(FG:3)}
D	{{(F:4)}	{(F:4)} D	{(FD:4)}
F	{}	{}	{}

Obr. 4.5: Pokračovanie príkladu, ktorý nadväzuje na ukážku 4.4. Na obrázku je ukázaný princíp vkladania do *FP-stromu* a následný postup pre získanie frekventovaných vzorov.

4.2.2 Dolovanie asociačných pravidiel

Po získaní súboru frekventovaných vzorov existuje možnosť využiť získané frekventované vzory na generovanie asociačných pravidiel [1]. Získané asociačné pravidlá umožňujú pochopiť reprezentáciu vzájomných závislostí medzi jednotlivými transakciami [14]. V problematike získavania znalostí z webových logov je možné chápať transakcie ako samotné cesty užívateľských sedení. Asociačné pravidlá sú zapisované vo formáte $X \Rightarrow Y$. Pre generovanie asociačných pravidiel z množiny frekventovaných vzorov je využívaná miera, ktorá sa označuje ako spoľahlivosť. Spoľahlivosť predstavuje podmienenú pravdepodobnosť, že transakcia obsahuje množinu položiek Y a to za predpokladu, že obsahuje množinu položiek X . Túto mieru možno matematicky definovať ako podiel podpory zjednotenia frekventovaných množín X, Y ku podpore frekventovanej množiny X . Ďalším asociačným pravidlom, ktoré bolo predstavené v predchádzajúcej kapitole je pravidlo podpory. Asociačné pravidlo podpory je matematicky definované ako pravdepodobnosť zjednotenia množín X, Y [11]. Za silné pravidlá sú považované tie pravidlá, ktoré spĺňajú predom definovaný minimálny prah podpory a spoľahlivosti. Problematiku silných asociačných pravidiel je možné vysvetliť na jednoduchom hypotetickom príklade. Uvažujme obchod so športovým tovarom. K dispozícii je 1000 transakcií, ktoré reprezentujú užívateľské nákupy v obchode so športovým tovarom. V tomto príklade nás zaujíma identifikácia pravidiel, ktoré dosahujú minimálnu podporu 15%. To znamená, že nás zaujíma aké tovarové položky si spoločne užívatelia zakúpili aspoň v 150 transakciách. Rovnako nás v tomto príklade zaujíma identifikácia pravidiel, ktoré dosahujú mieru spoľahlivosti vo výške minimálne 30%. To znamená, že nás zaujíma, aký tovar si užívatelia nakúpili s tovarom, ktorého minimálna podpora presahuje prah minimálnej podpory. Uvažujme nasledujúce získané pravidlo:

$$\text{nakupuje}(X, \text{rakety}) \Rightarrow \text{nakupuje}(X, \text{lopty}) \text{ [podpora} = 20\%, \text{ spoľahlivosť} = 55\%] \quad (4.4)$$

V prípade vyššie uvedeného pravidla ide o silné asociačné pravidlo a to z dôvodu, že hodnoty podpory a spoľahlivosti pre toto pravidlo presahujú prahy určujúce spodné ohraničenia uvedených mier. Podpora vo výške 20% značí, že v prípade 200 transakcií nakúpili zákazníci obchodu so športovým tovarom rakety a lopty spoločne. Miera spoľahlivosti vo výške 55% vyjadruje fakt, že v prípade ak si zákazníci kúpili rakety, tak si v 55% transakcií kúpili aj lopty. Doteraz predstavené miery podpory a spoľahlivosti nemusia byť dostatočné pre odfiltrovanie nezaujímavých asociačných pravidiel. Na odstránenie tohto nedostatku je možné využiť jednoduchú korelačnú mieru *lift*, ktorá skúma závislosti medzi množinami položiek. Výskyt množiny položiek X je nezávislý od výskytu množiny položiek Y , ak pravdepodobnosť zjednotenia týchto položiek sa rovná súčinu pravdepodobností týchto množín. V opačnom prípade sú množiny položiek X a Y závislé a korelované.

Vzorce asociačných pravidiel

V tejto podkapitole sú zhrnuté vzorce definujúce asociačné pravidlá vo formáte $X \Rightarrow Y$.

$$\text{Podpora}(X \Rightarrow Y) = P(X \cup Y) \quad (4.5)$$

$$\text{Spoľahlivosť}(X \Rightarrow Y) = P(X|Y) = \frac{\text{Podpora}(X \cup Y)}{\text{Podpora}(X)} \quad (4.6)$$

$$\text{Lift}(X \Rightarrow Y) = \frac{P(X \cup Y)}{P(X) \times P(Y)} = \frac{\text{Spoľahlivosť}(X \Rightarrow Y)}{\text{Podpora}(Y)} \quad (4.7)$$

4.3 Dolovanie sekvenčných vzorov

Cieľom problematiky dolovania sekvenčných vzorov je objavovanie zaujímavých podsekvencií, ktoré sa rovnako ako v prípade dolovania frekventovaných vzorov vyskytujú v databáze s frekvenciou vyššou alebo rovnou ako užívateľom zadaná hodnotou prahu [2]. Sekvenčná databáza zvyčajne obsahuje množstvo záznamov. Samotné záznamy sú usporiadané sekvenčne udalostí, ktoré môžu, ale zároveň nemusia obsahovať dodatočné informácie o konkrétnych časových údajoch pre jednotlivé záznamy. Problematika dolovania sekvenčných vzorov má široké aplikačné využitie. Medzi typické príklady využitia sekvenčných vzorov patria transakcie zákazníkov, sekvencie DNA alebo webové logy.

Definícia 1 „Nech $I = \{i_1, i_2, \dots, i_m\}$ je množina literálov označovaných ako položky, ktoré tvoria abecedu. Udalosť je neprázdna neusporiadaná kolekcia položiek. Bez straty všeobecnosti sa predpokladá, že položky udalosti sú zoradené v lexikografickom poradí. Sekvencia je usporiadaný zoznam udalostí. Udalosť sa označuje ako (i_1, i_2, \dots, i_k) , kde i_j je položka. Sekvencia α je označená ako $\langle \alpha_1 \rightarrow \alpha_2 \rightarrow \dots \rightarrow \alpha_q \rangle$, kde α_i je udalosť. Sekvencia s k položkami, kde $k = \sum_j |\alpha_j|$ sa nazýva k -sekvencia. Máme k dispozíciu databázu vstupných sekvencií \mathcal{D} , kde každá vstupná sekvenca v databáze má nasledujúce polia: identifikačné číslo sekvencie, čas udalosti a položky prítomné v udalosti. Predpokladá sa, že žiadna sekvenca nemá viac ako jednu udalosť s rovnakým časovým razítkom, takže časové razítko sa môže využiť ako identifikátor udalosti. Vo všeobecnosti je podpora alebo frekvencia sekvencie označená $\sigma(\alpha, \mathcal{D})$, definovaná ako počet (alebo podiel) vstupných sekvencií v databáze \mathcal{D} , ktoré obsahujú α “ [22].

Algoritmus PrefixSpan

Algoritmus *PrefixSpan* patrí do kategórie algoritmov pre dolovanie sekvenčných vzorov založených na raste vzorov [2]. Kľúčovou myšlienkou algoritmov tejto kategórie je opakované zmenšovanie prehľadávacieho priestoru rozdelením databázy na súbor menších databáz, ktoré umožňujú samostatné dolovanie sekvenčných vzorov. *PrefixSpan* skúma iba prefixové podsekvencie a následne premieta zodpovedajúce sufixové podsekvencie do podmnožín databázy, ktoré obsahujú dané frekventované položky. V prvom kroku činnosti tohto algoritmu je prehľadávaná sekvenčná databáza s cieľom identifikácie všetkých frekventovaných položiek, ktoré sa nachádzajú v sekvenciách [5]. Tieto identifikované frekventované položky predstavujú sekvenčné vzory s jednotkovou dĺžkou. Následne v ďalšom kroku algoritmu je získaná množina sekvenčných vzorov rozdelená do podmnožín a to na základe prefixov, ktoré predstavujú frekventované položky získané v predchádzajúcom kroku algoritmu. Každá podmnožina získaná rozdelením množiny sekvenčných vzorov je projekciou databázy sekvencií, a to vzhľadom na frekventované položky s jednotkovou dĺžkou. Vytvorenie súboru podmnožín databáz, ktoré obsahujú dané frekventované položky umožňuje dolovanie podmnožín sekvenčných vzorov. Z týchto databáz následne pokračuje činnosť algoritmu vytvorením frekventovaných sekvencií s dĺžkou dva s rovnakým prefixom pomocou frekventovaných sekvencií s jednotkovou dĺžkou. Vo všeobecnosti *PrefixSpan* pre každú frekventovanú sekvenciu s dĺžkou k rekurzívne vytvára podmnožinu databázy obsahujúcu danú frekventovanú položku určenú pre dolovanie frekventovaných sekvencií s dĺžkou $(k + 1)$.

Kapitola 5

Implementácia aplikácie

Informácie predstavené v tejto kapitole sa zaoberajú návrhom a implementáciou aplikácie, ktorá rieši problematiku získavania znalosti z webových logov. Pre implementáciu aplikácie bol zvolený interpretovaný programovací jazyk *Python*. Vzhľadom k veľkej popularite webových služieb a tiež s ohľadom na tému tejto bakalárskej práce je vytvorená aplikácia implementovaná ako webová aplikácia s použitím rámcového riešenia, ktoré poskytuje *Flask*. V prvej podkapitole tejto kapitoly 5.1 sú predstavené základné požiadavky, ktoré sú kľúčové pre implementáciu aplikácie tohto typu. Následne sú v podkapitole 5.2 zhrnuté knižnice, ktoré sú aplikáciou využívané pre rôzne oblasti jej činnosti. V podkapitole 5.3 je popísaná štruktúra implementovanej aplikácie. Zvyšné podkapitoly tejto kapitoly popisujú a bližšie špecifikujú jednotlivé čiastkové kroky, ktoré implementovaná aplikácia poskytuje pre získavanie znalostí z webových logov.

5.1 Návrh a požiadavky na aplikáciu

V tejto sekcii sú predstavené základné požiadavky na implementovanú aplikáciu, ktoré boli pre väčšiu prívetivosť a efektivitu aplikácie zvolené vo fáze návrhu aplikácie a následne boli prekonzultované. Detailnejšie implementačné informácie o jednotlivých položkách zo zoznamu požiadaviek na aplikáciu budú ďalej predstavené v ďalších sekciách tejto kapitoly. Medzi základné požiadavky na aplikáciu pre získavanie znalostí z webových logov patrí:

- Podpora spracovania veľkých logovacích súborov.
- Rýchlosť a efektivita predspracovania, zozbierania a vizualizácie štatistík.
- Poskytnutie rozšírených užívateľských možností pre predspracovanie.
- Zozbieranie zaujímavých štatistík počas predspracovania prístupových logov.
- Jednoduché a intuitívne grafické užívateľské rozhranie aplikácie.
- Dostupnosť a možnosť zobrazenia predspracovaných dátových rámcov.
- Dostupnosť a možnosť opätovného zobrazenia štatistík po predspracovaní.
- Možnosť opätovného využívania dostupných metód pre dolovanie znalostí s rôznymi vstupnými parametrami bez nutnosti opakovaného predspracovania dátových rámcov.

5.2 Použité knižnice

Python patrí medzi najrozšírenejšie programovacie jazyky. Rozšírenosť tohto programovacieho jazyka je zapríčinená jednoduchou syntaxou, a zároveň veľkou rozšírenosťou knižníc rôzneho zamerania. Pri implementácii aplikácie je vhodné využiť širokú podporu knižníc, ktoré sú pre tento jazyk dostupné, a to s ohľadom na fakt, že Python je interpretovaný jazyk, čo so sebou prináša určité rýchlostné obmedzenia. V tejto sekcii sú predstavené knižnice, ktoré tvoria základný stavebný kameň implementovanej aplikácie.

Knižnice pre predspracovanie a spracovanie dát

Po detekcii webových logovacích formátov (touto prvotnou fázou predspracovania dát sa bližšie zaoberá nasledujúca podkapitola 5.4.1) sú logovacie záznamy načítané do dátového rámca, ktorý poskytuje knižnica *pandas*¹. Táto knižnica určená pre jazyk Python poskytuje rýchle a flexibilné dátové štruktúry, ktoré boli navrhnuté a implementované s cieľom poskytnúť jednoduché a intuitívne operácie pre prácu s dátami [21]. Knižnica *pandas* je vhodná na spracovanie mnohých rôznych druhov dát, a primárne poskytuje dve dátové štruktúry:

- *Series*: jednorozmerná dátová štruktúra, vhodná k reprezentácii stĺpca tabuľky.
- *DataFrame*: dvojrozmerná dátová štruktúra, vhodná k reprezentácii tabuľky.

Pandas okrem iného umožňuje jednoduché spracovanie chýbajúcich záznamov, čo je možné v problematike predspracovania webových prístupových logov aplikovať na jednotlivé polia logovacích formátov s účelom odstránenia neúplných záznamov. Rovnako táto knižnica poskytuje možnosti vkladať alebo odstraňovať stĺpce z dátových rámcov. Odstránenie nepotrebných stĺpcov, ktoré reprezentujú polia logovacích záznamov vedú k nižšej spotrebe využívanej pamäte. Naopak možnosť vkladania nových stĺpcov je vhodná pre vytvorenie rôznych agregovaných štatistík. Okrem uvedených základných možností poskytuje *pandas* možnosť rozdelenia dátového rámca do skupín, a to pomocou aplikácie funkcie *groupby()*. Ďalšou pokročilou možnosťou, ktorú poskytuje knižnica je možnosť aplikovať vlastnú funkciu alebo prípadne funkciu implementovanú v inej knižnici na každú hodnotu v stĺpci dátového rámca. Túto možnosť poskytuje *pandas* pomocou dostupnej funkcie *apply()*. V prípade aplikácie tejto funkcie na reťazcové pole s veľkým množstvom záznamov môže táto aplikácia spotrebovať veľa výpočetného času, a tak implementovaná aplikácia využíva funkciu *apply()* v kombinácii s knižnicou *swifter*². Knižnica *swifter* poskytuje možnosť vektorizácie funkcie, ktorá je privedená do *pandas* funkcie *apply()*.

Knižnice pre vizualizáciu dát

Vzhľadom na povahu aplikácie pracujúcej s dátami a tiež s ohľadom na fakt, že aplikácia je implementovaná ako webová aplikácia bolo potrebné nájsť vhodný spôsob, akým spracované dáta rozumným spôsobom vizualizovať. Rámcové riešenie *Flasku* poskytuje šablónovací nástroj *Jinja2*³, ktorý umožňuje zdefinovanie vlastných HTML šablón. Implementovaná aplikácia pre korektný užívateľský vizuálny zážitok využíva okrem vlastných *CSS selektorov* vo veľkej miere štýlovacie rámcové riešenie, ktoré poskytuje *Bootstrap*⁴. Toto rámcové riešenie poskytuje jednoduché ovládanie štylizácie HTML tagov pomocou špecifikácie hodnôt

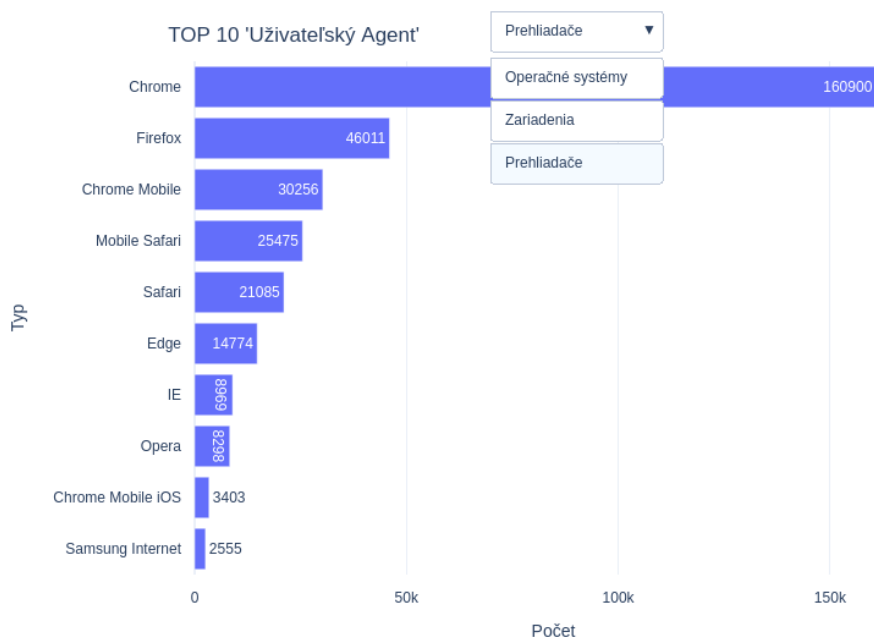
¹<https://github.com/pandas-dev/pandas>

²<https://github.com/jmcarpenter2/swifter>

³<https://flask.palletsprojects.com/en/2.1.x/templating/>

⁴<https://github.com/twbs/bootstrap>

triednych atribútov HTML tagov. Využitie *Bootstrapu* umožňuje jednoduchší a rýchlejší vývoj webových aplikácií, a to v kombinácii s vlastnými selektormi dotvára jednoduché grafické užívateľské rozhranie aplikácie. Okrem vizualizácie jednotlivých HTML šablón je tiež potrebné vizualizovať spracovávané dáta. Vzhľadom na širokú paletu dostupných grafov využíva implementovaná aplikácia pre vizualizáciu rôznych štatistik ale aj pre výstup do-
 lovacích algoritmov knižnicu *plotly*⁵. Táto knižnica poskytuje možnosť vytvoriť rôzne typy interaktívnych grafov či tabuliek, pričom zároveň umožňuje vhodným spôsobom upravovať rozloženie jednotlivých komponentov tvoriacich výsledný graf. Ďalšou výhodou pri využívaní knižnice *plotly* je možnosť zakomponovania rozbalovacích tlačidiel, ktoré aktualizujú atribúty grafov a prepínajú medzi jednotlivými vrstvami stôp. Táto možnosť výrazne šetrí nároky na miesto, ktoré jednotlivé grafy zaberajú. Plotly umožňuje voľbu z niekoľkých dostupných štýlovacích šablón, čím umožňuje priamočiarejšiu implementáciu. Rozhranie knižnice *plotly* ponúka dva rôzne moduly určené pre vytváranie grafických objektov [23]. Modul *plotly.graph_objects* umožňuje vytvárať grafické objekty, ktoré sú reprezentované stromovou dátovou štruktúrou. Túto štruktúru je možné pre vykreslenie automaticky serializovať do formátu *JSON*. Druhý vysokoúrovňový modul *plotly.express* poskytuje príležitosti pre jednoduché vytváranie grafických objektov. Nevýhodou tohto vysokoúrovňového modulu je fakt, že nie všetky typy grafov sú pre tento modul podporované. Všetky funkcie vysokoúrovňového modulu *plotly.express* sú postavené na grafických objektoch modulu *plotly.graph_objects* a rovnako poskytujú možnosť serializácie do formátu *JSON*.



Obr. 5.1: Ukážka horizontálneho stĺpcového grafu, ktorý bol vykreslený pomocou knižnice *plotly*. Graf v tejto ukážke využíva rozbalovacie tlačidlo, ktoré šetrí priestorové nároky na vizualizáciu spracovaných dát. Ukážka poskytuje informácie o najčastejšie využívaných prehliadačoch. Dáta, ktoré aplikácia spracúva za účelom poskytnutia tejto štatistiky boli získané z dátovej sady [20]. Ukážka pracuje s podmnožinou tejto dostupnej dátovej sady.

⁵<https://github.com/plotly/plotly.py>

Ďalšou knižnicou, ktorá je využívaná pre vizualizáciu spracovaných dát je *DataTables*⁶. Ako už aj názov tejto knižnice hovorí, tak táto knižnica slúži na jednoduchú vizualizáciu tabuľkových dát. Použitie tejto knižnice vyžaduje prilinkovanie ďalšej Javascriptovej knižnice *jQuery*⁷. Medzi výhody *DataTables* patria možnosti filtrácie riadkov podľa hľadaného vzoru, zoradenie jednotlivých stĺpcov tabuľky podľa hodnôt, ktoré danému stĺpcu náležia, a to zostupne, ale aj vzostupne. Rovnako ďalšou výhodou pre využitie tejto knižnice je jednoduché odoslanie dát pre následné zobrazenie dát, a tiež je dôležité vyzdvihnúť fakt, že táto knižnica ponúka jednoduché rozhranie k nastaveniu výsledného rozloženia tabuľky.

Čas uloženia	Nahrat	Formát	Spracované záznamy	Počet užívateľov	Počet sedení	Počet sedení (>1)				
21 Mar 2022 00:39:58	samuel.valastinn@gmail.com	Combined	261164	34039	62711	25181	Zobrazit štatistiky	Zobrazit dataset	Modelovať	Zmazať
16 Apr 2022 12:08:08	samuel.valastinn@gmail.com	Combined	18132	3794	4944	1914	Zobrazit štatistiky	Zobrazit dataset	Modelovať	Zmazať
16 Apr 2022 20:16:17	samuel.valastinn@gmail.com	Combined	5904	1510	1703	631	Zobrazit štatistiky	Zobrazit dataset	Modelovať	Zmazať
23 Mar 2022 21:35:28	samuel.valastinn@gmail.com	Combined	2046	413	537	206	Zobrazit štatistiky	Zobrazit dataset	Modelovať	Zmazať
23 Mar 2022 21:36:30	samuel.valastinn@gmail.com	Combined	2046	413	537	206	Zobrazit štatistiky	Zobrazit dataset	Modelovať	Zmazať
21 Mar 2022 14:58:14	samuel.valastinn@gmail.com	Combined	1962	695	721	316	Zobrazit štatistiky	Zobrazit dataset	Modelovať	Zmazať
27 Mar 2022 19:37:30	samuel.valastinn@gmail.com	Combined	969	401	449	255	Zobrazit štatistiky	Zobrazit dataset	Modelovať	Zmazať
21 Mar 2022 22:09:16	samuel.valastinn@gmail.com	Combined	419	300	331	40	Zobrazit štatistiky	Zobrazit dataset	Modelovať	Zmazať
21 Mar 2022 22:24:52	samuel.valastinn@gmail.com	Combined	419	300	331	40	Zobrazit štatistiky	Zobrazit dataset	Modelovať	Zmazať
27 Mar 2022 20:27:09	samuel.valastinn@gmail.com	Combined	419	300	331	40	Zobrazit štatistiky	Zobrazit dataset	Modelovať	Zmazať

Spracované logy

Záznamy 1 až 10 z celkom 16 (vyfiltrované spomedzi 17 záznamov)

Predchádzajúca 1 2 Nasledujúca

Obr. 5.2: Ukážka tabuľky obsahujúcej zoznam spracovaných logov. Pre zobrazenie tejto tabuľky je využívaná knižnica *DataTables*. Ako si možno všimnúť na ukážke sú zobrazené záznamy vyfiltrované na logovacie záznamy typu Combined a riadky tabuľky sú zoradené zostupne podľa počtu spracovaných záznamov.

Ostatné využívané knižnice

Okrem knižníc predstavených v tejto podkapitole využíva implementovaná aplikácia ešte niekoľko ďalších knižníc. Na rozdiel od doteraz predstavených knižníc je však ich činnosť špecifická iba pre určitú časť aplikácie, a tak budú tieto knižnice a ich činnosť ďalej predstavené až v nasledujúcich podkapitolách (5.4, 5.5, 5.6) tejto implementačnej kapitoly.

⁶<https://github.com/DataTables/DataTables>

⁷<https://github.com/jquery/jquery>

5.3 Štruktúra aplikácie

Informácie podané v tejto sekcii slúžia k odprezentovaniu základnej štruktúry implementovanej aplikácie. Implementovaná aplikácia má nasledujúcu štruktúru:

```
/
├── datasetsy
├── tmp
├── src
│   ├── static
│   │   ├── css
│   │   │   └── custom.css
│   │   └── favicon.ico
│   └── templates
│       ├── about.html
│       ├── change_folder_token.html
│       ├── display_dataset.html
│       ├── home.html
│       ├── preprocessing.html
│       ├── processed_logs.html
│       ├── setup_mining.html
│       ├── stats.html
│       └── template.html
├── analyze.py
├── detect.py
├── display.py
├── __init__.py
├── mine.py
├── save.py
├── stats.py
├── main.py
├── client_secrets.json
├── folder_token.txt
├── Procfile
├── README.md
├── requirements.txt
├── run.sh
└── token.json
```

Koreňový adresár aplikácie obsahuje tri priečinky, ktorých obsah ďalej tvorí:

- Priečinko *datasetsy* obsahuje niekoľko predom pripravených podmnožín voľne dostupných dátových sád. Tieto dátové sady sú určené pre stiahnutie a následné využitie, kedy je následne možné pomocou priloženia týchto dátových sád demonštrovať činnosť aplikácie vrátane získavania znalostí z týchto dátových sád.
- Priečinko *tmp* obsahuje lokálne dočasne uložené spracované dátové rámce a štatistiky získané po predspracovaní. Detailnejšie informácie a zmysel k čomu slúži obsah tohto priečinku bude predstavený v nasledujúcich podkapitolách.

- Priechinok *src* obsahuje zdrojové kódy implementovanej aplikácie a okrem zdrojových súborov implementovaných v jazyku Python, ktoré sú rozdelené na niekoľko modulov obsahuje ďalšie podpriechinky, ktorých obsah tvorí:
 - Podpriechinok *static* obsahuje vytvorenú ikonu aplikácie a okrem tejto ikony obsahuje podpriechinok *css*, ktorý obsahuje zdrojový súbor obsahujúci vlastné css selektory určené pre grafické užívateľské rozhranie aplikácie.
 - Podpriechinok *templates* tvoria vytvorené HTML šablóny. Všetky šablóny zo zoznamu šablón dedia a rozširujú šablónu zadanú v súbore *template.html*, ktorá obsahuje definície pre jednotlivé komponenty HTML a Javascriptu.

Okrem priechinok obsahuje koreňový adresár aj niekoľko súborov, ktoré sa líšia významom, ktorý pre implementovanú aplikáciu prinášajú.

- Zdrojový súbor *main.py* inicializuje balíček aplikácie z priechinku *src* a následne iniciuje štart aplikácie.
- Súbory *client_secrets.json*, *folder_token.txt* a *token.json* slúžia k správe ukladania spracovaných dátových rámcov a štatistík. Bližšie informácie k využitiu týchto súborov sú k dispozícii v podkapitole 5.5, ktorá sa detailnejšie zaoberá správou ukladania.
- Súbor *Procfile* obsahuje príkazy, ktoré sa vykonávajú pri spustení aplikácie na webovom serveri [12]. Existencia tohto súboru nie je nutná pre nasadenie aplikácie na webový hosting, avšak využitie tohto súboru poskytuje väčšiu kontrolu a flexibilitu pre výslednú aplikáciu.
- Textový súbor vo formáte markdown *README.md* poskytuje jednoduchý manuál pre lokálne spustenie výslednej aplikácie.
- Skript *run.sh* obsahuje súbor príkazov pre lokálne naštartovanie implementovanej aplikácie.
- Textový súbor *requirements.txt* obsahuje zoznam externých *Python* balíčkov, ktoré je potrebné nainštalovať pred spustením aplikácie.

5.4 Predspracovanie webových prístupových logov

V tejto sekcii je bližšie predstavená implementovaná problematika predspracovania webových prístupových logov vrátane viacerých podporných činností, ktoré sú v tejto fáze vykonávané. V podkapitole 5.4.1 je vysvetlený spôsob, akým implementovaná aplikácia detekuje podporované formáty webových prístupových logov a zároveň možnosti, ktoré umožňuje pre jednotlivé súbory s logovacími záznamami. Ďalšia podkapitola 5.4.2 sa zaoberá zberom rôznych štatistík, ktoré sú počas predspracovania logov získavané a vizualizované. V poslednej podkapitole 5.4.3 je popísaná prieskumná analýza užívateľských sedení vrátane vizualizácie. Pred samotným predspracovaním priloženého súboru, ktorý obsahuje prístupové záznamy je užívateľom aplikácie umožnené špecifikovať koncovky súborov, ktoré budú v rámci čistenia dát ponechané, a ktoré zároveň budú následne súčasťou ciest užívateľských sedení. Ďalšou užívateľskou možnosťou je voľba prahu, ktorý rozdeľuje užívateľské záznamy do užívateľských sedení. Pre čistenie a extrakciu vstupnej množiny dát vykonáva aplikácia kroky, ktoré už boli detailne popísané v kapitole 3.1.

Pedspracovanie a čistenie logov. Podporované logovacie formáty COMBINED, COMMON.

Pretiahnite logovací súbor/archív alebo kliknite pre výber súboru.
Podporované súbory s koncovkou .txt, .log a archívy ZIP a GZIP.

Analýza a štatistika logovacích súborov

+ Rozšírené možnosti

Pridanie podporovaných koncoviek súborov

.php, .html, .htm

Prah pre rozdelenie záznamov do užívateľských sedení

Obr. 5.3: Ukážka formuláru, ktorý spracováva užívateľské možnosti pre pedspracovanie prístupových logov.

5.4.1 Detekcia a spracovanie formátov webových prístupových logov

Implementovaná aplikácia v aktuálnej podobe podporuje spracovanie prístupových logov v štandardizovaných formátoch *Combined* a *Common*. Obsah týchto formátov bol predstavený v kapitole 2.4. Detekcia formátu prebieha načítaním prvého záznamu z priloženého súboru, ktorý je následne parsovaný. Aplikácia umožňuje spracovanie komprimovaných archívov *ZIP* a *GZIP*, a to s ohľadom na obrovskú veľkosť, ktorú môžu logovacie súbory zaberáť. V prípade spracovania *ZIP* archívu umožňuje aplikácia integráciu viacerých súborov s logovacími záznamami, ktoré majú jednotný formát. Po detekcii logovacieho formátu

je obsah priloženého súboru načítaný do dátového rámca. Následne sú odstránené záznamy, ktoré obsahujú prázdnu či chýbajúcu hodnotu v niektorom z polí, čím je docielené, že v načítanom dátovom rámci ostávajú iba plnohodnotné záznamy. Následne je vykonaná transformácia poľa určujúceho dátum a čas užívateľských žiadostí na časové razítko, a rovnako sú pre nižšiu spotrebu využívanej pamäte odstránené nevyužívané a nepotrebné polia.

5.4.2 Zbieranie a vizualizácia štatistík

Fáza predspracovania dát ponúka ideálne príležitosti pre zber rôznych štatistík počas čistenia priložených logovacích súborov. Zozbierané štatistiky možno rozdeliť na dve kategórie a to podľa doby kedy sú štatistiky zozbierané. Ešte pred očistením a odstránením nepotrebných logovacích záznamov sú aplikáciou získavané nasledujúce štatistiky:

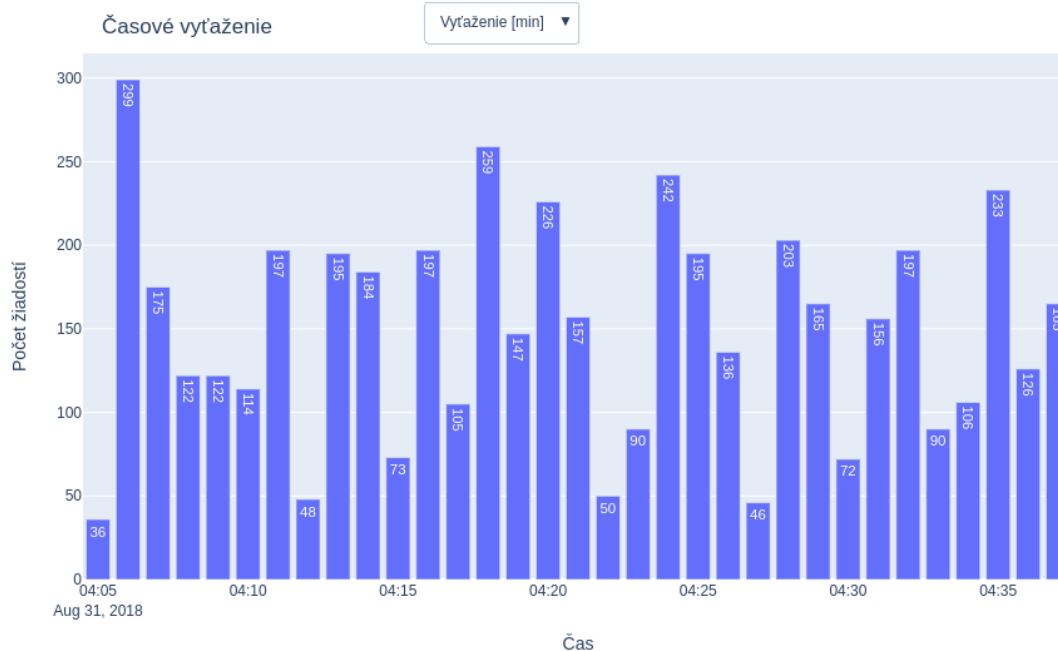
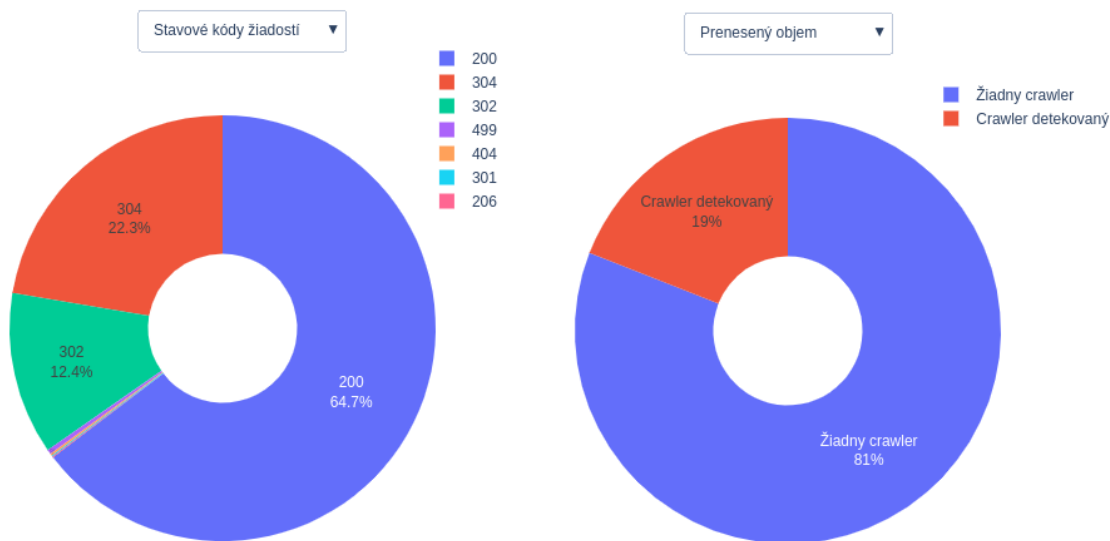
- Tabuľka agregovaných štatistík na základe *mime typov* vyžiadaných súborov, ktorá zahŕňa počet žiadostí o súbory daného typu, objem prenosu a počet žiadostí, pre ktoré bol záznam daného typu vytvorený detekovanými botmi.
- Grafy vyjadrujúce zastúpenie žiadostí, ktoré boli vykonané webovými botmi a reálnymi užívateľmi (prípadne botmi, ktorých sa nepodarilo detekovať), prenesený objem dát týmito subjektami, početnosť stavových kódov a početnosť zastúpenia jednotlivých HTTP metód.
- Stĺpcový graf zachytávajúci časové vyťaženie pre rôzne časové podvzorkovania.
- Pre dátové rámce štandardizovaného formátu *Combined* je získavaná početnosť zastúpenia najčastejšie sa vyskytujúcich operačných systémov, zariadení a prehliadačov.
- Pre dátové rámce štandardizovaného formátu *Common* je vyššie spomenutá štatistika vzhľadom na absenciu poľa *užívateľský agent* nahradená štatistikou najčastejšie vyžiadaných súborov.

Zozbieranie vyššie popísaných štatistík je voliteľné, čo znamená, že užívateľ v rámci rozšírených užívateľských možností môže zber týchto štatistík vypnúť. V prípade logovacích súborov s miliónmi záznamov je vykonávanie týchto prídavných činností, ktoré dokresľujú a dodávajú prvú predstavu o obsahu priloženého logovacieho súboru časovo náročnejšou operáciou.

Druhú kategóriu zbieraných štatistík (povinnú, na rozdiel od vyššie spomenutej kategórie) predstavujú štatistiky, ktoré sú zbierané počas alebo po predspracovaní webových prístupových logov. Túto kategóriu tvoria nasledujúce štatistiky:

- Tabuľka uchováajúca informácie počas predspracovania priložených logovacích súborov. Obsahuje detekovaný formát, počet načítaných záznamov, počet záznamov po vykonaní jednotlivých čistiacich krokov, následne počet identifikovaných užívateľov a informácie o počte užívateľských sedení.
- Graf znázorňujúci najčastejšie sa vyskytujúce prvé navštívené stránky užívateľských sedení.
- Graf znázorňujúci najčastejšie sa vyskytujúce posledné navštívené stránky pre jednotlivé užívateľské sedenia.

MIME typ	Koncovka	Objem prenosu (B)	Detekovaný crawler	Počet žiadostí
image/gif	.gif	1124114	10	1634
image/png	.png	219064	4	277
application/x-httpd-php	.php	1415407	47	153
text/css	.css	36054	2	49
font/woff	.woff	644924	0	14
image/jpeg	.jpg	265527	0	11
font/ttf	.ttf	158596	1	6
image/vnd.microsoft.icon	.ico	559146	0	6
image/svg+xml	.svg	72582	0	1



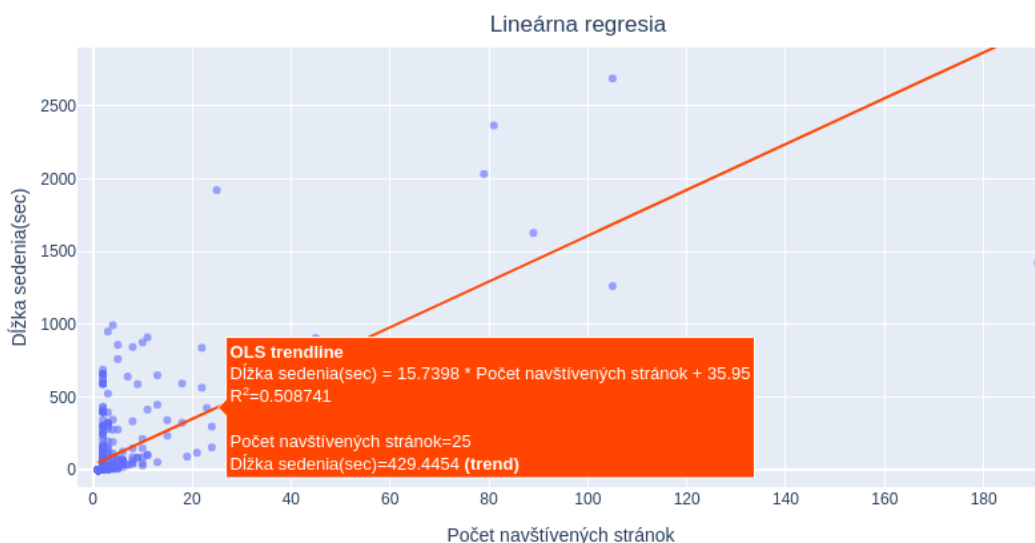
Obr. 5.4: Ukážka niektorých zozbieraných štatistík a ich grafická vizualizácia. Zdrojové dáta, z ktorých aplikácia spracováva a vizualizuje štatistiky sú získané z dátovej sady [20].

5.4.3 Aplikácia prieskumnej analýzy

Po extrakcii a vyčistení načítaného dátového rámca je vykonávaná identifikácia užívateľov. Implementovaná aplikácia poskytuje možnosť identifikácie užívateľov pomocou IP adresy, a rovnako pomocou kombinácie IP adresy a užívateľského agenta. Následne sú užívateľské záznamy rozdelené do užívateľských sedení, a to na základe užívateľom špecifikovaného prahu, ktorý rozdeľuje záznamy do sedení. Po vykonaní týchto činností prichádza na rad prieskumná analýza užívateľských sedení. Obsah tejto vykonávanej činnosti zahŕňa nasledujúce činnosti:

- Identifikáciu a vizualizáciu počtu navštívených stránok pre užívateľské sedenia.
- Identifikáciu a vizualizáciu dĺžky jednotlivých užívateľských sedení.
- Identifikáciu a vizualizáciu priemerného času na stránku počas užívateľských sedení.

Po vykonaní týchto činností sú v dátovom rámci ponechané iba užívateľské sedenia, ktorých dĺžka cesty je väčšia alebo rovná dvom. Následne je pozornosť zameraná na identifikáciu vzťahu medzi dĺžkou užívateľského sedenia a počtom navštívených stránok pre dané sedenie. Využívaným nástrojom na vyjadrenie vzťahu medzi týmito identifikovanými premennými je *regresná analýza* [19]. Jednoduchá lineárna regresia patrí medzi najvyužívanejšie techniky slúžiace pre určenie numerických predpovedí. Primárnym účelom tejto problematiky je vyhodnotiť relatívny vplyv prediktora⁸ na výsledok [28]. V tomto konkrétnom prípade je prediktorom premenná, ktorej hodnota reprezentuje počet navštívených stránok počas jednotlivých užívateľských sedení a je skúmané, aký vplyv má na dĺžku užívateľských sedení.

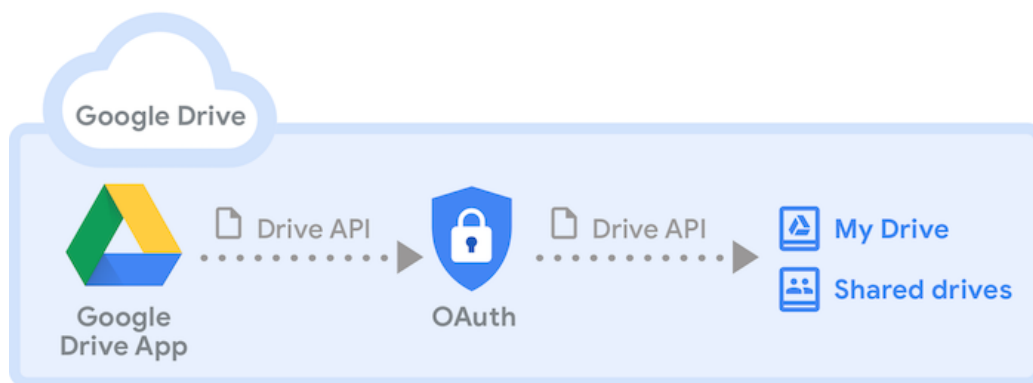


Obr. 5.5: Príklad grafickej vizualizácie jednoduchej lineárnej regresie pre menší dátový rámec. Odhadovaná dĺžka užívateľského sedenia sa pre tento predspracovaný dátový rámec rovná $15.7398 \times$ počet navštívených stránok + 35.95. To znamená, že odhadovaná dĺžka sedenia, pri ktorom užívateľ navštíví 25 stránok trvá približne 430 sekúnd. Zdroj dát, ktoré sú spracované aplikáciou pre túto ukážku pochádzajú z dátovej sady [20]. Ukážka využíva z dôvodu prehľadnejšej vizualizácie iba menšiu vhodne zvolenú podmnožinu dát.

⁸Prediktor možno chápať ako premennú, na základe ktorej možno predikovať hodnotu inej premennej.

5.5 Ukladanie spracovaných dátových rámcov a štatistík

Jednou zo základných požiadaviek z fázy návrhu aplikácie je možnosť zobrazenia dostupných štatistík a obsahu dátových rámcov po predspracovaní dát. Aplikácia tiež vyžaduje ukladanie spracovaných dát pre umožnenie následného spracovania algoritmami pre získavanie znalostí z webových logov. Takto zadané požiadavky vyžadujú ukladanie spracovaných dát. Štandardným riešením je využitie relačnej či prípadne nerelačnej databázy. Avšak v tomto prípade implementovaná aplikácia využíva alternatívne riešenie, ktoré pre tento problém ponúka *Google Drive API*. Toto rozhranie pre programovanie aplikácií umožňuje vytvárať aplikácie, ktoré využívajú cloudové úložisko Google Disk [8].



Obr. 5.6: Diagram znázorňujúci vzťahy medzi aplikáciou využívajúcou Google Drive, službou Google Drive a rozhraním pre programovanie aplikácií Google Drive API. Obrázok prevzatý z [8].

Ako je možné si všimnúť z vyššie ukázaného diagramu, tak *Google Drive API* využíva autorizačný protokol *OAuth 2.0*, ktorý slúži pre overenie používateľov aplikácie. V prípade, ak aplikácia využíva poskytovanú službu Google prihlásenia, tak tento autorizačný protokol slúži pre spracovanie prístupových tokenov aplikácie. Pre zjednodušenie komunikácie s rozhraním *Google Drive API* využíva implementovaná aplikácia knižnicu *PyDrive2*⁹, ktorá okrem zjednodušenia práce s rozhraním poskytuje tiež flexibilitu pre správu autorizácie. Táto knižnica pre svoju činnosť vyžaduje lokálnu kópiu súboru *client_secrets.json*, ktorý obsahuje všetky autentifikačné informácie o aplikácii [9]. To znamená, že využitie tejto knižnice pre ukladanie štatistík a dátových rámcov na cloudové úložisko Google Disku vyžaduje vytvorenie projektu, ktorý musí následne povoliť využívanie rozhrania *Google Drive API*. Následne je potrebné vytvoriť a nastaviť poverenia. Knižnica *PyDrive2* umožňuje správu autentifikácie prostredníctvom vytvorenia lokálneho webového servera, ktorý spracováva autentifikáciu. Rovnako je tiež dostupná možnosť autentifikácie z terminálu. Po úspešnom prihlásení umožňuje knižnica uložiť poverenia obsahujúce prístupový token do súboru, čím nie je nutné vyžadovať opakovanú autentifikáciu prihláseného užívateľa. Token je ukladaný do *JSON* súboru *token.json*, a zároveň je využívaný počas celej doby existencie prihlásenia pre aplikačný prístup k Google disku.

⁹<https://github.com/iterative/PyDrive2>

5.5.1 Spôsob ukladania a načítania predspracovaných rámcov a štatistík

Po dokončení všetkých činností vykonávaných pre predspracovanie dát a po zozbieraní všetkých štatistík je ešte pred zobrazením informácií a štatistík o predspracovanom rámci lokálne uložený dátový rámec aj štatistiky. Dátový rámec je lokálne uložený do súboru v serializovanom formáte *pickle*. Grafické objekty, slúžiace na vizualizáciu získaných štatistík sú počas predspracovania ukladané do slovníku, ktorý je následne serializovaný do *JSON* súboru. Následne sú užívateľovi zobrazené štatistiky a informácie získané z fázy predspracovania. V prípade ak sa užívateľ rozhodne pre uloženie spracovaných dát, tak sú tieto súbory presunuté na Google Disk. Pre úspešne vykonanie tohto presunu súborov je potrebné prihlásenie *Google* účtom do aplikácie a rovnako je potrebné zvoliť aktívny priečinok, do ktorého sú súbory ukladané (informácie o možnostiach, ktoré poskytuje aplikácia pre správu ukladania sú k dispozícii v sekcii 5.5.2). Lokálne uložené serializované súbory sú k dispozícii iba obmedzený čas, pričom zvolený prah, počas ktorého sú tieto súbory lokálne uložené je 15 minút. To znamená, že pri ďalšom predspracovaní iného alebo totožného dátového rámca je kontrolované lokálne úložisko a dané súbory sú vymazané v prípade, ak dátové rámce a štatistiky boli uložené pred dlhším časovým úsekom ako je zvolený prah. Užívateľ má rovnako možnosť po vzhliadnutí informácií o spracovanom rámci rozhodnúť, že nechce ukladať daný dátový rámec. V tomto prípade sú dočasne uložené lokálne súbory zmazané. Knižnica *PyDrive2* umožňuje jednoduchú manipuláciu so súbormi uloženými na Google Disku, a tak umožňuje jednoduchý prístup k uloženým súborom. Tento spôsob ukladania bol zvolený pre jednoduchosť načítania uložených dátových rámcov, ktoré z formátu *pickle* umožňuje knižnica *pandas*. Ďalšou výhodou využívania tohto konceptu je fakt, že tento spôsob ukladania podporuje prístup k spracovaným dátám bez ohľadu na to, či aplikácia beží lokálne alebo na webovom serveri.

5.5.2 Administrácia správy ukladania

Ako už bolo v predchádzajúcich sekciách tejto kapitoly spomenuté, tak aplikácia využívajúca tento spôsob ukladania vyžaduje spracovanie potrieb, ktoré zjednodušujú správu ukladania. Implementovaná aplikácia na tieto potreby reaguje a snaží sa vhodným spôsobom automatizovať úkony, ktoré umožňujú jednoduchšiu správu ukladania. Každý priečinok existujúci na Google Disku má svoj unikátny identifikátor, ktorý je využívaný pre možnosť ukladania súborov obsahujúcich spracované rámce a vizualizované grafické objekty do tohto priečinku. Rovnako majú svoje jednoznačné identifikátory aj súbory všetkých typov. Problémom je, že pri manuálnom prehliadaní disku nie sú tieto identifikátory dostupné. Tento a aj ďalšie problémy, ktoré sa vyskytujú so správou ukladania tohto typu rieši implementovaná aplikácia. Medzi možnosti, ktoré ponúka patria:

- Vytvorenie priečinku na Google Disku, pričom pre samotné vytvorenie stačí špecifikovať názov priečinku. Pred vytvorením je kontrolovaná existencia priečinku rovnakého mena, a tak sa nemôže stať, že budú vytvorené priečinky ktoré zdieľajú rovnaké meno.
- Prelistovanie obsahu aktívneho priečinku, ktorý zobrazí zoznam súborov uložených v danom priečinku vrátane ich *mime typov*.
- Vyhľadanie zoznamu existujúcich priečinkov, ktoré sú k dispozícii pre aktuálne prihláseného užívateľa. V tomto zozname sú obsiahnuté aj identifikátory priečinku, ktoré možno využiť pre nasledujúci bod tohto zoznamu možností.

- Zmena aktívneho priečinku. Táto činnosť vyžaduje dodanie identifikátoru zvoleného priečinku. Identifikátor aktívneho priečinku je ukladaný do súboru *folder_token.txt*.
- Zmena prihláseného užívateľa. Pri tejto možnosti sú zmazané podporné súbory. Konkrétne ide o súbory *token.json* a *folder_token.txt*. Po zmazaní týchto súborov je pomocou knižnice *PyDrive2* vytvorený webový server, ktorý spracováva proces autentifikácie.
- Zmazanie spracovaného dátového rámca spoločne s grafickými objektami zo zoznamu spracovaných logovacích dátových rámcov.

Zmena priečinku pre ukladanie spracovaných rámcov/štatistik

Aktuálny identifikátor priečinku
1pH7QdEKdAg4E3CBEUSRbAODysRlnk_qh

Nastaviť nové ID priečinku na G drive

Zmeniť aktívny priečinok

Rozšírené možnosti

Prelisťovať obsah aktívneho priečinku na google drive

Vytvoriť priečinok na google drive

Vyhľadať dostupné priečinky

Zmeniť prihláseného užívateľa

Názov priečinku	Dátum a čas vytvorenia	Identifikátor priečinku
backup_logs	2022-04-16T21:19:26.272Z	1pH7QdEKdAg4E3CBEUSRbAODysRlnk_qh
processed_logs	2022-02-28T12:24:43.629Z	1NSQvicFgYUoZVytKxEkZHw8QclRijyth

Zoznam dostupných priečinkov

Obr. 5.7: Grafická ukážka administrácie správy uloženia. Na ukážke je zobrazená tabuľka dostupných priečinkov. Na základe identifikátoru priečinku možno nastaviť aktívny priečinok. Do aktívneho priečinku sú ukladané spracované dátové rámce a grafické objekty.

5.6 Scenáre pre získavanie znalostí z webových logov

V predchádzajúcich sekciách tejto implementačnej kapitoly boli popísané činnosti, ktoré sú aplikáciou vykonávané so zámerom efektívne a čo najpresnejšie pripraviť a spracovať dáta pre následné získavanie znalostí z týchto dát. Jednou z najdôležitejších požiadaviek na implementovanú aplikáciu je možnosť viacnásobnej úpravy vstupných parametrov pre techniky a algoritmy určené na získavanie znalostí z dát. Splnenie tejto kľúčovej požiadavky a teda možnosti viacnásobnej úpravy vstupných parametrov poskytuje lepšie podmienky pre identifikáciu doposiaľ neznámych znalostí. Implementovaná aplikácia poskytuje možnosti pre získavanie znalostí prostredníctvom techník a algoritmov, ktoré boli detailne popísané a špecifikované v kapitole 4. Celkovo je k dispozícii 5 rôznych dolovacích scenárov. Pre riešenie tejto problematiky prostredníctvom dolovania sekvenčných vzorov alebo asociačných pravidiel je poskytovaná možnosť vykonania zhlukovej analýzy pred samotným aplikovaním týchto dolovacích metód. Grafické ukážky, ktoré v tejto podkapitole demonštrujú výstup dolovacích algoritmov v podobe získaných znalostí využívajú vhodne zvolené podmnožiny dát z rovnakej dátovej sady, ktorá bola získaná z dostupného zdroja [20].

Zhlukovanie

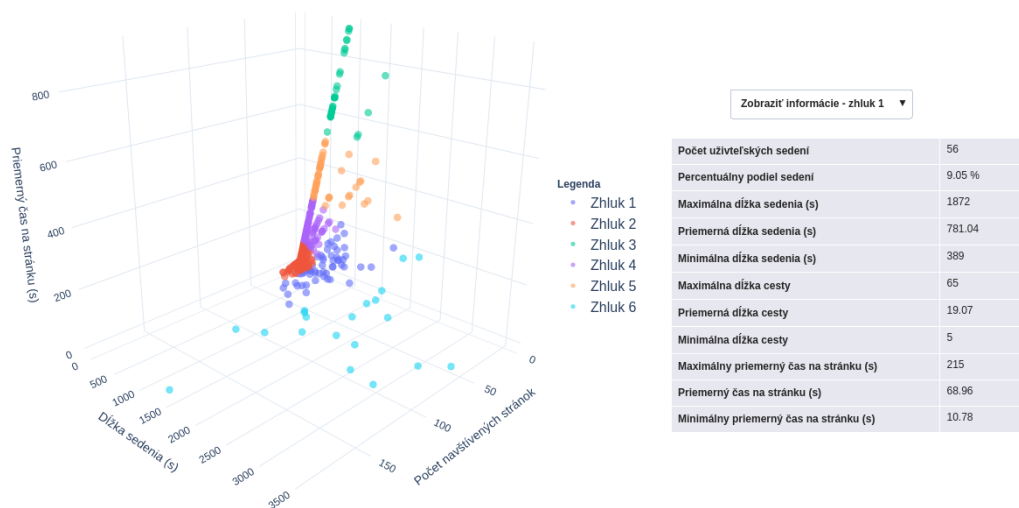
Pre rozdelenie záznamov užívateľských sedení do zhlukov sú pre aktuálnu implementáciu aplikácie podporované nasledujúce zhlukovacie algoritmy:

- *K-Means*: Pre možnosť využitia tohto zhlukovacieho algoritmu založeného na rozdelení je pre užívateľov dopredu vyžadované špecifikovať vyžadovaný počet zhlukov. Platí podmienka, že špecifikovaný počet zhlukov musí byť menší ako počet záznamov užívateľských sedení, ktorých cesta obsahuje minimálne dve navštívené stránky.
- *BIRCH*: Možnosť využitia tohto hierarchického zhlukovacieho algoritmu vyžaduje špecifikáciu vstupných parametrov určujúcich prah a vyvažovací faktor. Aplikácia zároveň umožňuje voľbu špecifikácie vyžadovaného počtu zhlukov, pričom voľba tohto parametra je pre zhlukovací algoritmus *BIRCH* voliteľná.
- *DBSCAN*: Tento zhlukovací algoritmus založený na hustote vyžaduje pre svoju činnosť dopredu špecifikovať vstupný parameter ϵ , ktorý reprezentuje polomer okolia pre zhlukovanie a rovnako je vyžadovaná špecifikácia vstupného parametra určujúceho minimálny počet bodov v okolí jadrového bodu.

Po načítaní zvoleného dátového rámca zo zoznamu predspracovaných dátových rámcov je vykonávaná *Min-Max* normalizácia. Pri normalizácii hodnôt sú normalizované číselné atribúty, ktoré sú získané z prieskumnej analýzy užívateľských sedení. Pri normalizácii sú hodnoty týchto atribútov transformované do rozsahu $(0, 1)$. Konkrétne ide o tieto atribúty:

- Počet navštívených stránok
- Dĺžka užívateľských sedení
- Priemerný čas strávený na stránke počas užívateľského sedenia

Následne je podľa týchto normalizovaných atribútov vykonané zhlukovanie, a to pomocou užívateľom zvoleného algoritmu. Po vykonaní zhlukovania sú vizualizované užívateľské sedenia zaradené do zhlukov prostredníctvom grafu, pričom v tomto prípade sú pre jednotlivé osi využívané nenormalizované hodnoty. Rovnako je pre každý zhluk vytvorená prehľadná tabuľka, ktorá poskytuje zaujímavé charakteristiky a znalosti o jednotlivých zhlukoch.



Obr. 5.8: Grafická ukážka vizualizácie zhlukovania.

Interpretácia ukážky a výsledkov získaných zhlukovaním

V ukážke bol pre rozdelenie vstupnej množiny 619 užívateľských sedení do zhlukov využitý algoritmus *k-Means* s vyžiadaným počtom šiestich zhlukov. Tento algoritmus zaradil približne polovicu užívateľských sedení do *zhluku 2*, ktorého užívateľské sedenia sú charakteristické malým počtom navštívených stránok v rozsahu 2–32 stránok s krátkou dĺžkou sedení, ktorá v priemere trvá niečo viac ako jednu minútu. Ako je možné si všimnúť aj z ukážky, tak *zhluk 1* pozostáva z o niečo dlhších užívateľských sedení, ktoré pozostávajú z navštívených stránok v rozsahu 5–65 stránok. Medzi jednotlivými užívateľskými žiadosťami o ďalšiu stránku je dlhší časový rozstup ako v prípade zhluku 2. *Zhluk 3* rovnako pozostáva z kratších užívateľských sedení, avšak v prípade objektov zaradených do tohto zhluku ide zrejme o sedenia, ktoré boli spracované zo začiatku, či prípadne konca priloženého logovaciego súboru. Pre užívateľské sedenia s dlhým priemerným časom na stránku má obrovský vplyv voľba prahu, ktorá je užívateľmi špecifikovaná vo fáze predspracovania. V tomto prípade bol zvolený prah pre rozdelenie užívateľských záznamov do užívateľských sedení s hodnotou šesťnástich minút. Ďalším zaujímavým zhlukom je *zhluk 6*, ktorý pozostáva z dlhých užívateľských sedení, pri ktorých užívatelia pracujú s informačným systémom dlhšie ako bežný užívatelia informačného systému. Dlhé užívateľské sedenia, počas ktorých užívatelia navštívia v priemere 72 stránok a strávia na stránke približne tridsať minút sú v tomto predspracovanom dátovom rámci zastúpené s podielom dosahujúcim takmer 16%.

Zhodnotenie výsledkov získavania znalostí pomocou zhlukovania

Implementovaná aplikácia bola navrhnutá s cieľom, aby poskytla čo najvšeobecnejšie možnosti, ktoré sú určené pre vykonanie zhlukovej analýzy. Tento cieľ bol určený z dôvodu, že dáta obsiahnuté v prístupových logoch, ktoré sú počas predspracovania zaradené do užívateľských sedení môžu byť rôzneho charakteru. Z dostupných algoritmov je pre následné dolovanie znalostí pomocou sekvenčných vzorov alebo asociačných pravidiel najjednoduchšie zvoliť možnosť zhlukovania pomocou algoritmu *k-Means*. Menším problémom pre využitie tohto algoritmu je jeho aplikácia na väčšie spracované dátové rámce, kde využitie algoritmu *BIRCH* zaberá rádovo menej výpočetného času. V prípade využitia algoritmu

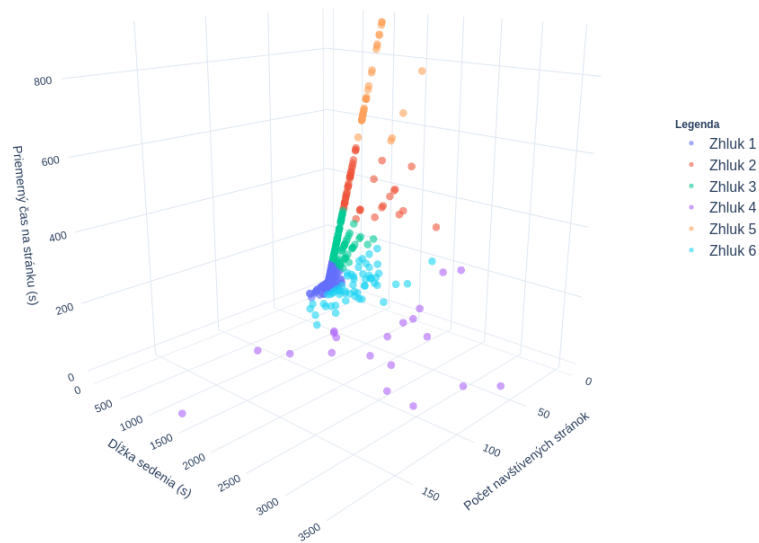
BIRCH má vplyv na získané výsledky zhlukovania najmä voľba parametra, ktorý špecifikuje prah pre rozdelenie do zhlukov. Zmena vyvažovacieho faktoru má oveľa menší vplyv na získané výsledky pri použití tohto algoritmu. Algoritmus *DBSCAN* dosahuje najhoršie výsledky pre možnosti následného dolovania znalostí. Aplikácia tohto algoritmu na predspracovaný dátový rámec obsahujúci užívateľské sedenia zvyčajne prináša totožný výsledok, kedy tento algoritmus radí viac ako 70–80% užívateľských sedení do rovnakého zhluku a ostatné zhluky sú reprezentované nízkym percentuálnym zastúpením užívateľských sedení. Vhodnou voľbou vstupných parametrov je možné využiť tento algoritmus pre následné dolovanie znalostí, avšak voľba vstupných parametrov, ktoré umožňujú rozdelenie do zhlukov vhodnej veľkosti má malé pracovné okno, v ktorom spoľahlivo pracuje. V prípade niektorých dátových sád, vzhľadom k faktu, že dáta s ktorými aplikácia pracuje nemožno ovplyvniť je využitie tohto algoritmu pre následné dolovanie sekvenčných vzorov alebo asociačných pravidiel ťažko využiteľné. Avšak na druhej strane *DBSCAN* umožňuje skúmať časté užívateľské správanie, a to s ohľadom na veľmi vysoké percentuálne zastúpenie užívateľských sedení, ktoré sú zaradené do rovnakého zhluku.

Získavanie znalostí pomocou asociačných pravidiel

Pre získavanie znalostí prostredníctvom *dolovania asociačných pravidiel* umožňuje implementovaná aplikácia pred touto dolovaciou úlohou vykonať *zhlukovú analýzu*. V prípade užívateľskej voľby tohto dolovacieho scenára sú následne dolované asociačné pravidlá len pre tie zhluky, ktoré obsahujú minimálne desať objektov reprezentujúcich užívateľské sedenia. Okrem tejto možnosti podporuje aplikácia užívateľskú voľbu druhého dostupného dolovacieho scenára, ktorý poskytuje možnosť dolovania asociačných pravidiel so zdrojom dát v podobe všetkých ciest užívateľských sedení. Činnosť dolovania asociačných pravidiel je v prípade implementovanej aplikácie vykonávaná v troch krokoch:

- V prvom kroku sú cesty užívateľských sedení transformované do dátovej matice, ktorá reprezentuje riedku binárnu databázu. Dáta sú tak pripravené na dolovanie frekventovaných vzorov a na následné dolovanie asociačných pravidiel. Formát v akom sa v tomto kroku dáta nachádzajú bol bližšie špecifikovaný a znázornený v kapitole 4.2.
- V ďalšom kroku je vykonávané dolovanie frekventovaných vzorov. Pre možnosť dolovania frekventovaných vzorov vyžaduje aplikácia špecifikáciu minimálnej podpory v rozsahu $\langle 0, 1 \rangle$. Pre riešenie problematiky dolovania frekventovaných vzorov sú k dispozícii algoritmy *FP-Growth* a *Apriori*, ktorých činnosť bola detailne popísaná v kapitole 4.2.1. Využitie oboch algoritmov vedie k rovnakým výsledkom, čím sa tieto algoritmy odlišujú od zhlukovacích algoritmov.
- Posledným krokom, ktorý je vykonávaný je samotné dolovanie asociačných pravidiel. Asociačné pravidlá sú získavané z množiny frekventovaných vzorov. Pre možnosť dolovania silných asociačných pravidiel je nutné dopredu špecifikovať mieru spoľahlivosti. Táto miera opäť očakáva a vyžaduje užívateľskú špecifikáciu v rozsahu $\langle 0, 1 \rangle$.

Po vykonaní týchto krokov sú následne získané silné asociačné pravidlá vizualizované prostredníctvom tabuľky. Na nasledujúcej stránke tejto práce je k dispozícii ukážka výstupu aplikácie v podobe získaných silných asociačných pravidiel 5.9.



Zobrazit asocičné pravidlá - zhluk 1 ▾

Asociačné pravidlo	Podpora	Spoľahlivosť	Lift
/studium/lib_class/sess_ext.php => /studium/index.php	27.09 %	52.43 %	0.8
/studium/login.php => /studium/index.php	22.63 %	95.29 %	1.45

Zobrazit asocičné pravidlá - zhluk 4 ▾

Asociačné pravidlo	Podpora	Spoľahlivosť	Lift
/studium/index.php => /studium/predmety/index.php	60.0 %	63.16 %	1.05
/studium/predmety/index.php, /studium/lib_class/sess_ext.php => /studium/index.php	60.0 %	100.0 %	1.05
/studium/predmety/index.php, /studium/index.php => /studium/lib_class/sess_ext.php	60.0 %	100.0 %	1.18
/studium/lib_class/sess_ext.php, /studium/index.php => /studium/predmety/index.php	60.0 %	70.59 %	1.18
/studium/predmety/index.php => /studium/lib_class/sess_ext.php, /studium/index.php	60.0 %	100.0 %	1.18
/studium/lib_class/sess_ext.php => /studium/predmety/index.php, /studium/index.php	60.0 %	70.59 %	1.18
/studium/index.php => /studium/predmety/index.php, /studium/lib_class/sess_ext.php	60.0 %	63.16 %	1.05
/studium/lib_class/sess_ext.php => /studium/rozhng/roz_predmet_gl.php	50.0 %	58.82 %	1.18
/studium/rozhng/roz_predmet_gl.php => /studium/lib_class/sess_ext.php	50.0 %	100.0 %	1.18
/studium/index.php => /studium/rozhng/roz_predmet_gl.php	50.0 %	52.63 %	1.05
/studium/rozhng/roz_predmet_gl.php => /studium/index.php	50.0 %	100.0 %	1.05
/studium/predmety/index.php => /studium/rozhng/roz_predmet_gl.php	45.0 %	75.0 %	1.5
/studium/rozhng/roz_predmet_gl.php => /studium/predmety/index.php	45.0 %	90.0 %	1.5

Obr. 5.9: Grafická ukážka získaných silných asociačných pravidiel. Pred dolovaním asociačných pravidiel bolo v tomto prípade vykonané zhlukovanie algoritmom *k-Means*. Minimálna miera podpory pre tento príklad dosahuje 20% a minimálna miera spoľahlivosti 50%.

Interpretácia ukážky a výsledkov získaných dolovaním asociačných pravidiel

Táto ukážka pracuje s rovnakým predspracovaným dátovým rámcom, ktorý obsahuje 619 užívateľských sedení. V prípade *zhluku 1*, ktorého užívateľské sedenia sú považované za krátke užívateľské sedenia s priemernou dĺžkou sedenia viac ako 1 minútu a s počtom navštívených stránok v rozsahu 2–32 navštívených stránok sa podarilo vydolovať dve silné asociačné pravidlá. Prvé vydolované pravidlo, pri ktorom informačný systém nevyžaduje opätovnú autentifikáciu užívateľa sa vyskytuje vo viac ako 27% percentách sedení tohto

zhluku. Spôľahlivosť tohto pravidla a teda podmienená pravdepodobnosť návštevy hlavnej stránky informačného systému dosahuje nižšiu hodnotu, ako v prípade druhého vydolovaného pravidla, kedy v 22% percentách užívateľských sedení zaradených do tohto zhluku užívateľ navštívi hlavnú stránku a stránku pre prihlásenie do informačného systému zároveň. Vo všeobecnosti je mnoho asociačných pravidiel vydolovaných najmä pre zhluky, ktoré pozostávajú z dlhších užívateľských sedení, pri ktorých užívatelia navštívia mnoho stránok. V tomto konkrétnom prípade sa podarilo vydolovať najviac silných asociačných pravidiel pre *zhluku 4*. Veľké množstvo silných asociačných pravidiel bolo rovnako vydolované pre sedenia zaradené do *zhluku 6*, ktorého cesty užívateľských sedení sa skladajú z 5–65 navštívených stránok. Žiadne silné asociačné pravidlá sa nepodarilo vydolovať v prípade *zhluku 5* a *zhluku 2*, ktoré sú špecifické veľkým časovým rozstupom medzi jednotlivými navštívenými stránkami počas užívateľských sedení. To znamená, že podobnosť správania užívateľov pre tieto kategórie užívateľských sedení je nižšia ako v prípade sedení, ktoré majú v priemere nižšie časové rozstupy medzi návštevami jednotlivých stránok.

Asociačné pravidlo	Podpora	Spoľahlivosť	Lift
/studium/lib_class/sess_ext.php => /studium/index.php	26.62 %	56.76 %	0.82
/studium/index.php => /studium/lib_class/sess_ext.php	26.62 %	38.53 %	0.82
/studium/login.php => /studium/index.php	19.65 %	89.86 %	1.3
/studium/term_st2/index.php => /studium/lib_class/sess_ext.php	11.89 %	78.12 %	1.67
/studium/term_st2/index.php => /studium/index.php	10.14 %	66.67 %	0.96
/studium/zkous_st/index.php => /studium/lib_class/sess_ext.php	10.78 %	85.0 %	1.81
/studium/zkous_st/index.php => /studium/index.php	9.83 %	77.5 %	1.12

Obr. 5.10: Grafická ukážka vydolovaných asociačných pravidiel bez vykonania zhlukovej analýzy. Ako je možné vidieť z ukážky, tak aplikácia zhlukovej analýzy pred dolovaním umožňuje dolovanie s vyššie špecifikovanými prahmi pre miery podpory a spoľahlivosti.

Zhodnotenie výsledkov získavania znalostí pomocou asociačných pravidiel

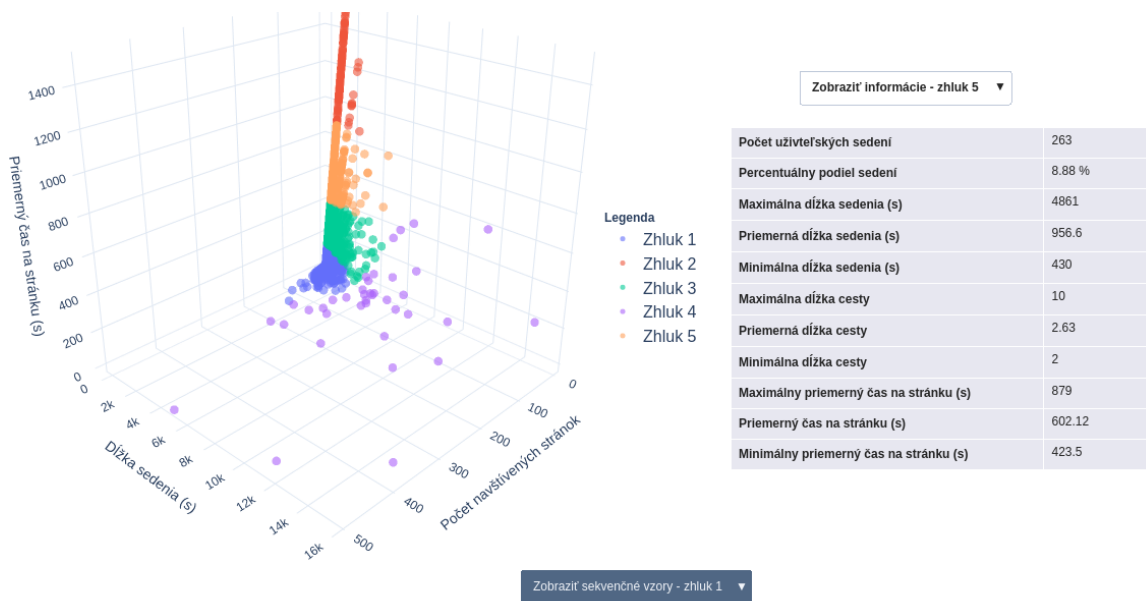
V počiatkovej implementácii aplikácie bolo pri voľbe zhlukovania pred získavaním asociačných pravidiel umožnené dolovať asociačné pravidlá zo všetkých zhlukov, ktoré obsahovali päť a viac užívateľských sedení. Táto podmienka pre následné dolovanie frekventovaných vzorov, z ktorých sú následne získavané asociačné pravidlá bola zmenená do súčasnej podoby, kedy je umožnené dolovať asociačné pravidlá zo zhlukov, ktoré obsahujú minimálne desať užívateľských sedení z dôvodu, že aplikácia algoritmov dolovacích frekventované vzory vyžadovala množstvo pamäte a znižovala efektivitu týchto algoritmov čo zapríčiňovalo, že dolovanie frekventovaných vzorov trvalo neúmerne dlhý časový úsek. Tento problém sa vyskytoval a prejavoval najmä v prípadoch existencie zhluku obsahujúceho malý počet užívateľských sedení, ktorých cesty pozostávali z mnohých navštívených stránok. Pre aktuálnu implementáciu sú asociačné pravidlá s vhodne zvolenými prahmi mier podpory a spoľahlivosti dolované za primeraný časový úsek. Využitím zhlukovej analýzy je umožnené špecifikovať a následne vydolovať asociačné pravidlá, ktoré po zaradení užívateľských sedení do zhlukov umožňujú dolovanie s vyššími prahmi špecifikujúcimi minimálne miery podpory a spoľahlivosti. Ako už bolo spomenuté, tak využitie oboch algoritmov pre dolovanie frekventovaných vzorov umožňuje získanie rovnakých výsledkov, pričom mierne rýchlejšim riešením sa vo väčšine prípadov prezentuje algoritmus *FP-Growth*. V budúcnosti je možné implementovať rozšírenie, ktoré umožní filtrovať získané silné asociačné pravidlá pomocou špecifikácie vyžadovanej hodnoty pre mieru *lift*, ktorá už v tejto práci bola predstavená v kapitole 4.2.2.

Získavanie znalostí pomocou sekvenčných vzorov

Poslednou dostupnou dolovacíou úlohou určenou pre získavanie znalostí z webových logov je získavanie znalostí pomocou sekvenčných vzorov. Implementovaná aplikácia pre vyriešenie tejto problematiky využíva algoritmus *PrefixSpan*, ktorého činnosť bola predstavená v sekcii 4.3. Totožne ako v prípade asociačných pravidiel sú pre riešenie tejto problematiky poskytované dva rôzne dolovacie scenáre. Pre prvý scénar je pred samotným dolovaním sekvenčných vzorov vykonaná *zhluková analýza*, pričom následné dolovanie sekvenčných vzorov prebieha len z tých zhlukov, ktoré obsahujú minimálne desať záznamov užívateľských sedení. Druhý dostupný scénar doluje sekvenčné vzory zo všetkých ciest užívateľských sedení, ktoré sú k dispozícii pre užívateľom zvolený spracovaný dátový rámec. Oba scenáre vyžadujú špecifikáciu dvoch vstupných parametrov. Prvým vstupným parametrom je miera podpory vzorov pričom aplikácia vyžaduje špecifikáciu tejto miery v rozsahu $(0, 1)$. Druhým vstupným parametrom je minimálna dĺžka sekvencie. Po získaní sekvenčných vzorov, ktorých podpora je vyššia alebo rovná hodnote prahu minimálnej podpory sú získané vzory vizualizované prostredníctvom tabuľky, ktorá uchováva okrem sekvenčného vzoru aj percentuálnu početnosť výskytu daného sekvenčného vzoru, a tiež aj podporu vydolovaného sekvenčného vzoru vo formáte “podpora sekvencie/dĺžka dátového rámca”. Na nasledujúcej strane tejto práce je k dispozícii ukážka výstupu aplikácie spoločne s interpretáciou týchto výsledkov, ktoré boli získané prostredníctvom dolovania sekvenčných vzorov 5.11.

Zhodnotenie výsledkov získavania znalostí pomocou sekvenčných vzorov

Rovnako ako v prípade dolovania asociačných pravidiel tak aj pre riešenie problematiky dolovania sekvenčných vzorov po vykonaní zhlukovania bolo pôvodne umožnené dolovať sekvenčné vzory zo zhlukov, ktorým bolo priradených minimálne päť užívateľských sedení. Pre aktuálnu implementáciu bol tento počet navýšený na minimálne desať užívateľských sedení pre zachovanie jednotného štandardu aj keď sa ukázalo, že v prípade dolovania sekvenčných vzorov pomocou algoritmu *PrefixSpan* sa tento problém neprejavoval až tak výrazne ako v prípade dolovania frekventovaných vzorov. Zefektívnenie a zníženie vyžadovaného výpočetného času pre riešenie tejto problematiky umožnila najmä transformácia reťazcového formátu jednotlivých stránok, z ktorých sa skladajú cesty užívateľských sedení na celočíselný formát. Táto činnosť je vykonávaná po načítaní dátového rámca a ešte pred samotným dolovaním sekvenčných vzorov. Všetky unikátne stránky, ktoré sú získané zo všetkých ciest užívateľských sedení sú namapované na celočíselné indexy. Následne sú jednotlivé stránky, ktoré tvoria cesty užívateľských sedení nahradené týmito celočíselnými indexami. Počas dolovania sekvenčných vzorov sú využívané tieto celočíselné indexy, čo znižuje pamäťové nároky a tiež značne zrýchľuje rýchlosť dolovania. Po vydolovaní sú následne indexy stránok, z ktorých sa skladajú získané sekvenčné vzory spätne namapované na stránky, ktoré sú na konci vizualizované. Vykonaná úprava a optimalizácia umožňuje vyriešenie tejto problematiky s využitím zhlukovej analýzy, ale aj bez, a to s rozumnou spotrebou výpočetného času. Využitie zhlukovania pred dolovaním sekvenčných vzorov umožňuje zvýšenie prahu minimálnej podpory pre získanie sekvenčných vzorov z užívateľských sedení zaradených do zhlukov, alebo prípadne umožňuje dolovanie sekvencií s vyššou minimálnou dĺžkou pri zachovaní totožnej hodnoty prahu reprezentujúcej minimálnu hodnotu podpory pre dolovanie sekvenčných vzorov.



Podpora	Sekvenčný vzor	Početnosť [%]
675/2059	/studium/index.php -> /studium/index.php	32.78
583/2059	/studium/lib_class/sess_ext.php -> /studium/index.php	28.31
401/2059	/studium/login.php -> /studium/index.php	19.48
389/2059	/studium/lib_class/sess_ext.php -> /studium/term_st2/index.php	18.89
352/2059	/studium/term_st2/index.php -> /studium/term_st2/index.php	17.1
289/2059	/studium/lib_class/sess_ext.php -> /studium/term_st2/index.php -> /studium/term_st2/index.php	14.04
286/2059	/studium/lib_class/sess_ext.php -> /studium/zkous_st/index.php	13.89
242/2059	/studium/index.php -> /studium/term_st2/index.php	11.75
239/2059	/studium/lib_class/sess_ext.php -> /studium/index.php -> /studium/index.php	11.61
238/2059	/studium/index.php -> /studium/zkous_st/index.php	11.56
233/2059	/studium/lib_class/sess_ext.php -> /studium/predmety/index.php	11.32
228/2059	/studium/term_st2/index.php -> /studium/term_st2/index.php -> /studium/term_st2/index.php	11.07

Zobrazit sekvenčné vzory - zhluk 3

Podpora	Sekvenčný vzor	Početnosť [%]
244/496	/studium/index.php -> /studium/index.php	49.19
130/496	/studium/login.php -> /studium/index.php	26.21
85/496	/studium/index.php -> /studium/index.php -> /studium/index.php	17.14
70/496	/studium/lib_class/sess_ext.php -> /studium/index.php	14.11
59/496	/studium/login.php -> /studium/login.php	11.9
57/496	/studium/index.php -> /studium/login.php	11.49

Obr. 5.11: Grafická ukážka získaných sekvenčných vzorov.

Interpretácia ukážky a výsledkov získaných dolovaním sekvenčných vzorov

Experiment, z ktorého bola vyhotovená vyššie vizualizovaná ukážka prebiehal na väčšej dátovej sade 2963 užívateľských sedení, ktoré vznikli predspracovaním 1 700 000 uchovaných logovacích záznamov. Tento počet záznamov bol zozbieraný webovým serverom v časovom intervale dvanástich hodín. Pred dolovaním sekvenčných vzorov bolo vykonané zhľukovanie hierarchickým zhľukovacím algoritmom *BIRCH*, ktorý klasifikoval užívateľské sedenia do

piatich rozdielnych zhlukov. Najviac sekvenčných vzorov je opätovne vydolovaných z dlhých užívateľských sedení, ktoré sú v tomto prípade zaradené do *zhluku 4*. Rovnako ako v prípade vyššie spracovaného experimentu s asociačnými pravidlami, tak aj v prípade dolovania sekvenčných vzorov bolo najmenej sekvenčných vzorov získaných zo zhlukov, ktoré majú väčšie priemerné časové rozostupy medzi žiadosťami o stránku a teda vyšší priemerný strávený čas na stránku. Kvalitné výsledky a množstvo vydolovaných vzorov poskytujú aj *zhluk 1* a *zhluk 3*, pričom vydolované sekvenčné vzory s minimálnou dĺžkou 2 a s prahom miery podpory vo výške 10% sú v ukážke zobrazené a vizualizované. Jednotlivé stránky, z ktorých sa vydolované sekvenčné vzory skladajú sa pre niektoré vzory opakujú, čo značí, že používatelia informačného systému vyžadujú počas sedení niektoré unikátne stránky opakovane. Obzvlášť zaujímavé vzory poskytuje *zhluk 3*, ktorý pozostáva z kratších užívateľských sedení v rozmedzí 2–45 navštívených stránok. Pre viac ako 26% sedení zaradených do tohto zhľuku navštívia užívatelia sekvenciu pozostávajúcu z návštevy stránky pre prihlásenie a následne rovnako navštívia hlavnú stránku informačného systému. Pre takmer 12% sedení zaradených do tohto zhľuku sa návšteva stránky pre prihlásenie opakuje. Tento fakt možno interpretovať a chápať viacerými spôsobmi. Užívatelia mohli v týchto prípadoch buď zadať chybné prihlasovacie údaje či prípadne mohli počas jedného užívateľského sedenia využívať viacero rôznych účtov. Na poslednom vydolovanom sekvenčnom vzore *zhľuku 3* je rovnako možné vidieť úzke prepojenie medzi hlavnou stránkou informačného systému a stránkou určenou pre prihlásenie užívateľov do tohto informačného systému.

Sekvenčné vzory

Podpora	Sekvenčný vzor	Početnosť [%]
1129/2963	/studium/index.php -> /studium/index.php	38.1
700/2963	/studium/lib_class/sess_ext.php -> /studium/index.php	23.62
613/2963	/studium/login.php -> /studium/index.php	20.69
431/2963	/studium/lib_class/sess_ext.php -> /studium/term_st2/index.php	14.55
395/2963	/studium/index.php -> /studium/index.php -> /studium/index.php	13.33
387/2963	/studium/term_st2/index.php -> /studium/term_st2/index.php	13.06
323/2963	/studium/lib_class/sess_ext.php -> /studium/term_st2/index.php -> /studium/term_st2/index.php	10.9
321/2963	/studium/lib_class/sess_ext.php -> /studium/zkous_st/index.php	10.83

Obr. 5.12: Grafická ukážka získaných sekvenčných vzorov pre rovnaký predspracovaný dátový rámec bez využitia klasifikácie užívateľských sedení, ktorá je k dispozícii prostredníctvom vykonania zhľukovej analýzy. Ako je možné vidieť na základe porovnania s ukážkou 5.11, tak klasifikácia užívateľských sedení umožňuje vydolovať viac sekvenčných vzorov, či prípadne umožňuje dolovanie s vyššie špecifikovanou mierou minimálnej podpory a minimálnej dĺžky sekvencie. Pre obe ukážky v tejto podkapitole boli sekvenčné vzory dolované s minimálnou mierou podpory v hodnote 10% a zároveň s minimálnou dĺžkou sekvencie 2.

Kapitola 6

Záver

Cieľom tejto bakalárskej práce bolo preštudovanie problematiky získavania znalostí z webových logov a následná implementácia aplikácie, ktorá vykonáva niekoľko dolovacích úloh. Aplikácia¹ v súčasnosti podporuje spracovanie dvoch webových prístupových logovacích formátov s pevným počtom polí. Aplikácia poskytuje päť rôznych dolovacích scenárov. Pre dolovanie asociačných pravidiel a sekvenčných vzorov je poskytovaná možnosť zhlukovania pred dolovaním znalostí pomocou týchto metód. Pre rozdelenie množiny užívateľských sedení do zhlukov sú k dispozícii tri rôzne zhlukovacie algoritmy. V prípade algoritmu *DB-SCAN* sa ukázalo, že tento algoritmus umožňuje detekciu odlahlých užívateľských sedení, avšak využitie tohto algoritmu nie je vhodné pre následné dolovanie sekvenčných vzorov alebo asociačných pravidiel. Ostatné algoritmy určené pre zhlukovanie umožňujú s vhodne zvolenými vstupnými parametrami rozdelenie a kategorizáciu užívateľských sedení. Pre dolovanie frekventovaných vzorov, z ktorých možno následne získať asociačné pravidlá poskytuje aplikácia možnosť voľby medzi algoritmami *FP-Growth* a *Apriori*. Aplikovanie týchto algoritmov po zhlukovaní môže v prípade výskytu zhľuku, ktorého objekty predstavujúce užívateľské sedenia obsahujú dlhé cesty spotrebovať množstvo dostupnej pamäte a výpočetného času. Na dolovanie sekvenčných vzorov je umožnené získavať znalosti prostredníctvom algoritmu *PrefixSpan*. Aplikácia tohto algoritmu po zhlukovaní je časovo náročná v prípade existencie zhľuku, ktorého cesty sú zložené z mnohých položiek. Tento problém sa podarilo eliminovať transformáciou reťazcového formátu stránok do celočíselného formátu pred dolovaním znalostí. Po dolovaní sú formáty stránok spätne transformované do reťazcového formátu. V budúcnosti je možné rozšíriť aplikáciu pridaním ďalšieho algoritmu pre dolovanie sekvenčných vzorov, pričom zaujímavým riešením je implementácia algoritmu pre dolovanie uzavretých sekvenčných vzorov. Najväčším problémom pri riešení tejto práce bolo získanie dostupných dátových sád, a to vzhľadom na citlivosť dát obsiahnutých v prístupových logoch. Ďalším možným rozšírením je možnosť získavania znalostí z dopytových vyhľadávacích logov. Aplikácia v súčasnej podobe umožňuje načítanie jedného konkrétneho dopytového vyhľadávacieho logu, ktorého predspracovanie a následné získavanie znalostí nebolo kvôli nedostatku času implementované. Ďalším problémom tohto možného rozšírenia je fakt, že tento druh webových logov nemá jednotný štandard, čo neumožňuje všeobecné riešenie tohto problému. V budúcnosti je rovnako možné rozšírenie aplikácie a umožnenie spracovania ďalších webových štandardizovaných formátov. Pre aktuálnu implementáciu aplikácie nie sú tieto formáty podporované, a to kvôli absencii dostupných dátových sád.

¹<https://logmine.herokuapp.com/>

Literatúra

- [1] AGGARWAL, C. C. *Data Mining: The Textbook*. Cham: Springer International Publishing, 2015. ISBN 978-3-319-14141-1.
- [2] AGGARWAL, C. C. a HAN, J. *Frequent Pattern Mining*. Cham: Springer International Publishing AG, 2014. ISBN 978-3-319-07820-5.
- [3] AGGARWAL, C. C. a REDDY, C. K. *Data clustering: algorithms and applications*. Boca Raton, FL: CRC Press, 2014. Chapman & Hall/CRC data mining and knowledge discovery series. ISBN 978-1-4665-5821-2.
- [4] AGOSTI, M. a DI NUNZIO, G. M. Gathering and Mining Information from Web Log Files. In: *Digital Libraries: Research and Development*. Springer Berlin Heidelberg, 2007, s. 104–113. Lecture Notes in Computer Science. ISBN 3540770879.
- [5] ALOYSIUS, G. a BINU, D. An approach to products placement in supermarkets using PrefixSpan algorithm. *Journal of King Saud University - Computer and Information Sciences*. 2013, zv. 25, č. 1, s. 77–87. DOI: 10.1016/j.jksuci.2012.07.001. ISSN 1319-1578. Dostupné z: <https://www.sciencedirect.com/science/article/pii/S1319157812000353>.
- [6] CHAPMAN, P., CLINTON, J., KERBER, R. et al. *CRISP-DM 1.0* [online]. the-modeling-agency.com, 2000 [cit. 2021-11-12]. Dostupné z: <https://www.the-modeling-agency.com/crisp-dm.pdf>.
- [7] CHODAK, G., SUCHACKA, G. a CHAWLA, Y. HTTP-level e-commerce data based on server access logs for an online store. *Computer networks*. 2020, zv. 183, s. 107589–107589. DOI: <https://doi.org/10.1016/j.comnet.2020.107589>. ISSN 1389-1286. Dostupné z: <https://www.sciencedirect.com/science/article/pii/S1389128620312263>.
- [8] GOOGLE. *Introduction to Google Drive Api* [online]. Apríl 2022 [cit. 2022-04-21]. Dostupné z: <https://developers.google.com/drive/api/guides/about-sdk>.
- [9] GWAK, J., BLEVINS, S., NABEL, R. et al. *Quickstart* [online]. 2020 [cit. 2022-04-21]. Dostupné z: <https://docs.iterative.ai/PyDrive2/quickstart/>.
- [10] HALLAM BAKER, P. M. a BEHLENDORF, B. *Extended Log File Format* [online]. 1996 [cit. 2021-12-17]. Dostupné z: <https://www.w3.org/TR/WD-logfile.html>.
- [11] HAN, J., KAMBER, M. a PEI, J. *Data mining: Concepts and Techniques*. 3. vyd. Waltham, Mass: Morgan Kaufmann Publishers, 2011. ISBN 978-0-12-381479-1.

- [12] HEROKU DEV CENTER. *The Procfile* [online]. Február 2022 [cit. 2022-04-18]. Dostupné z: <https://devcenter.heroku.com/articles/procfile>.
- [13] KHAN, K., REHMAN, S. U., AZIZ, K. et al. DBSCAN: Past, present and future. In: *The Fifth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT 2014)*. 2014, s. 232–238. DOI: 10.1109/ICADIWT.2014.6814687.
- [14] LI, C., LI, Y., YANG, Y. a DENG, Y. Analysis of Web Log Mining Based on Association Rule. In: *Innovative Computing*. Singapore: Springer Singapore, 2020, sv. 675, s. 665–674. Lecture Notes in Electrical Engineering. ISBN 9789811559587.
- [15] LINGRAS, P. a AKERKAR, R. *Building An Intelligent Web: Theory And Practice*. 1. vyd. Jones & Bartlett Learning, Llc, 2007. ISBN 978-0-7637-4137-2.
- [16] LIU, B. *Web data mining : Exploring Hyperlinks, Contents, and Usage Data*. 2. vyd. Berlin : Springer, 2011. ISBN 978-3-642-19459-7.
- [17] LIU, Y. a GUAN, Y. FP-Growth algorithm for application in research of market basket analysis. In: IEEE. *2008 IEEE International Conference on Computational Cybernetics*. 2008, s. 269–272. ISBN 978-1-4244-2874-8.
- [18] MAIMON, O. a ROKACH, L. *Data Mining and Knowledge Discovery Handbook*. New York, NY: Springer, 2005. ISBN 978-0-387-24435-8.
- [19] MARKOV, Z. a LAROSE, D. T. *Data mining the web: Uncovering Patterns in Web Content, Structure, and Usage*. Hoboken : John Wiley & Sons, 2007. ISBN 978-0-471-66655-4.
- [20] MAŇÁSEK, M. a TŮMA, P. *Charles University SIS Access Log Dataset* [online]. Zenodo, jún 2019 [cit. 2022-04-26]. DOI: 10.5281/zenodo.3241445. Dostupné z: https://zenodo.org/record/3241445/export/hx#.Ymkv_3tBxhE.
- [21] MCKINNEY, W. a PANDAS DEVELOPMENT TEAM. *Pandas: powerful Python data analysis toolkit* [online]. Apríl 2022 [cit. 2022-04-16]. Dostupné z: <https://pandas.pydata.org/docs/pandas.pdf>.
- [22] MOONEY, C. H. a RODDICK, J. F. Sequential pattern mining – Approaches and Algorithms. *ACM computing surveys*. NEW YORK: ACM. 2013, zv. 45, č. 2, s. 1–39. DOI: 10.1145/2431211.2431218. ISSN 0360-0300.
- [23] PLOTLY. *Graph Objects in Python* [online]. Apríl 2022 [cit. 2022-04-17]. Dostupné z: <https://plotly.com/python/graph-objects/>.
- [24] PRABA, V. L. a VASANTHA, T. WEB STRUCTURE MINING USING PAGERANK, IMPROVED PAGERANK – AN OVERVIEW. *ICTACT journal on communication technology*. ICT Academy of Tamil Nadu. 2011, zv. 2, č. 1, s. 270–276. ISSN 0976-0091.
- [25] REDDY, K. S., REDDY, M. K. a SITARAMULU, V. An effective data preprocessing method for Web Usage Mining. In: *2013 International Conference on Information Communication and Embedded Systems (ICICES)*. 2013, s. 7–10. DOI: 10.1109/ICICES.2013.6508197.

- [26] SRIVASTAVA, M., GARG, R. a KAUSHALA, P. K. Analysis of Data Extraction and Data Cleaning in Web Usage Mining. In: *Proceedings of the 2015 International Conference on Advanced Research in Computer Science Engineering & Technology (ICARCSET 2015)*. New York, NY, USA: Association for Computing Machinery, Marec 2015. DOI: 10.1145/2743065.2743078. ISBN 9781450334419.
- [27] SRIVASTAVA, M., GARG, R. a MISHRA, K. Preprocessing Techniques in Web Usage Mining: A Survey. *International Journal of Computer Applications*. Júl 2014, zv. 97, s. 18. DOI: 10.5120/17104-7737.
- [28] ZOU, K. H., TUNCALI, K. a SILVERMAN, S. G. Correlation and simple linear regression. *Radiology*. Radiological Society of North America. 2003, zv. 227, č. 3, s. 617–628.