VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

ROZŠÍŘENÍ UŽIVATELSKÝCH PROFILŮ PRO ÚČELY CÍLENÉ REKLAMY

EXTENSION OF USER PROFILES FOR TARGETED ADVERTISING PURPOSES

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. FILIP HADAČ
AUTHOR

VEDOUCÍ PRÁCE — Doc. Ing. JAROSLAV ZENDULKA, CSc.
SUPERVISOR

BRNO 2019

Ústav informačních systémů (UIFS) Akademický rok 2018/2019

Zadání diplomové práce

22119

Student: **Hadač Filip, Bc.**
Program: Informační technologie Obor: Inteligentní systémy
Název: **Rozšíření uživatelských profilů pro účely cílené reklamy**
Extension of User Profiles for Targeted Advertising Purposes
Kategorie: Data mining
Zadání:

1. Seznamte se s problematikou extrakce strukturovaných dat z webových stránek.
2. Seznamte se s problematikou pokročilých analýz a využití strojového učení při získávání znalostí z dat.
3. Seznamte se s vymezením uživatelských profilů a cílových skupin v zadavatelské firmě a s požadavky na obohacení profilů o nové informace extrahované z webových stránek.
4. Navrhněte a implementujte nástroj pro rozšíření profilů o požadované nové informace ze zadaných webových stránek.
5. Navrhněte analýzy a jim odpovídající experimenty pro vyhodnocení přínosu obohacení profilů pro účely cílené reklamy.
6. Analýzy a experimenty navržené v předchozím bodě proveďte.
7. Výsledky analýz a experimentů vyhodnoťte a diskutujte možná vylepšení.

Literatura:

- McKinney, W. Python for data analysis (2nd ed.). Sebastopol, USA: O'Reilly, Inc. (2017). ISBN 978-1-491-95766-0.
- Raschka, S. Python machine learning. Birmingham: Packt Publishing Ltd. (2015). ISBN 978-1-78355-513-0.
- Joshi, P. Artificial intelligence with Python. Birmingham, UK: Packt Publishing Ltd. (2017). ISBN 978-1-78646-439-2.
- What is web scraping - Part 1 - Beginner's guide. Dostupné na https://www.scrapehero.com/a-beginners-guide-to-web-scraping-part-1-the-basics/.

Při obhajobě semestrální části projektu je požadováno:

- Body 1 až 4.

Podrobné závazné pokyny pro vypracování práce viz http://www.fit.vutbr.cz/info/szz/

Vedoucí práce: **Zendulka Jaroslav, doc. Ing., CSc.**
Konzultant: Polcar Jiří, Mgr., Ph.D., GAUSSALGO
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 22. května 2019
Datum schválení: 29. října 2018

Abstrakt

Tato práce se zabývá návrhem a realizací obohacení uživatelských profilů pro účely vylepšení cílené reklamy. Pro získání nových informací je využita extrakce dat z webových stránek. Extrahovaná data pochází ze dvou serverů, ČSFD a Recepty. V případě ČSFD se jedná o filmové žánry, zatímco u Recepty se jedná o kategorie receptů. Pomocí streamovacích aplikací se tyto informace zpracují a uloží do databází uživatelských profilů. Nad profily spadajícími do určité reklamní kampaně se provádí předzpracování a následně klasifikační algoritmy strojového učení pro vyhodnocení přínosu nových informací. Vyhodnocením experimentů je poznatek, že nově obohacené informace mají mírný přínos pro vylepšení cílené reklamy.

Abstract

This thesis is devoted to designation and realisation of the extension of user profiles for improvement targeted advertising purposes. Web scraping is used for acquirement of new data information. Extracted data comes from two servers, ČSFD and Recepty. Data from ČSFD are film genres. Data from Recepty are categories of recepies. Streaming applications are used for processing of data and saving them to databases of user profiles. Preprocessing and machine learning classification algorithms are used for benefit evaluation of new informations for profiles in advertising campaigns. Evaluation of experiments shows that new informations have slight benefit in improvement advertising campaigns.

Klíčová slova

Dolování dat, extrakce dat z webových stránek, strojové učení, předzpracování, klasifikace, streamovací aplikace, uživatelské profily, cílená reklama.

Keywords

Data mining, web scraping, machine learning, preprocessing, classification, streaming applications, user profiles, targeted advertising.

Citace

HADAČ, Filip. *Rozšíření uživatelských profilů pro účely cílené reklamy.* Brno, 2019. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Doc. Ing. Jaroslav Zendulka, CSc.

Rozšíření uživatelských profilů pro účely cílené reklamy

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Doc. Ing. Jaroslav Zendulka, CSc. Další informace mi poskytl Mgr. Jiří Polcar, PhD. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

. . . . . .
Filip Hadač
15. května 2019

Poděkování

Chtěl bych poděkovat především svému vedoucímu diplomové práce Doc. Ing. Jaroslavu Zendulkovi, CSc. za veškerou pomoc při tvorbě a cenné rady. Dále bych rád poděkoval firmě Gauss algorithmic za poskytnutí diplomové práce a Mgr. Jiřímu Polcarovi, PhD. za externí konzultace.

Obsah

Kapitola 1

Úvod

Práce se zabývá obohacením uživatelských profilů pro vylepšení cílené internetové reklamy. Obohacením uživatelských profilů se myslí rozšíření jejich informací o nové informace z navštívených webových stránek. Pro obohacení je využita extrakce dat z webových stránek. Výběr cílových skupin probíhá za pomocí strojového učení a nové informace by tento proces měly vylepšit. Práce se dělí na 4 kapitoly, teorii, návrh, implementaci a analýzu.

První kapitola vysvětluje potřebnou teorii k pochopení problematiky a obsahuje 3 části. Extrakce dat z webových stránek popisuje vše od původu přes funkcionalitu až po možná využití. Uživatelské profily jsou využívány pro doporučování internetových reklam. Pro profily jsou popsány základní principy technologií, veškeré informace v nich obsažené a celý proces obohacení. Z profilů se vybírají cílové skupiny na základě reklamních kampaní a pro tyto účely je potřeba jejich obohacení o nové požadované informace. Strojové učení je odvětví umělé inteligence, jehož cílem je vytvořit počítačový program schopen samostatného učení a adekvátních reakcí.

Druhá kapitola se zabývá návrhem potřebným k implementaci a realizaci. Nejdříve je popsána analýza jednotlivých webových stránek, na kterých se nachází potřebné informace pro obohacení uživatelských profilů. Součástí analýzy je také ukázání struktury stránky a dat. Dále se kapitola zabývá návrhem získání těchto informací a jejich uložením do databází uživatelských profilů, kde jsou rozebrány jednotlivé komponenty k tomu potřebné. Pro manipulaci s informacemi se využívá mapování do přehlednějších forem.

Třetí kapitola navazuje na kapitolu druhou a obsahuje implementaci navržených částí v jazyce Python. První částí je mapování nových informací do různých forem. Dále je popsána streamovací knihovna, která se využívá pro streamovací aplikace. Další částí je získání nových informací z webových stránek a jejich uložení do URL databáze k dalšímu použití. Obohacení profilů je část popisující úpravu stávajících aplikací a tvorbu nových pro uložení nových informací do databází. Poslední částí je nasazení do produkce.

Čtvrtá kapitola obsahuje analýzy pro vyhodnocení přínosu nových informací v uživatelských profilech. Analýzy potřebují vstupní data, která je potřeba dále upravit pomocí technik předzpracování. Na předzpracování navazuje příprava dat, která se zabývá rozdělením na trénovací a testovací skupiny a vyřešením nepoměru klasifikačních tříd. Dále jsou popsány experimenty, kterým předchází výběr modelu. Na závěr je vyhodnocení experimentů a zhodnocení přínosu nových informací.

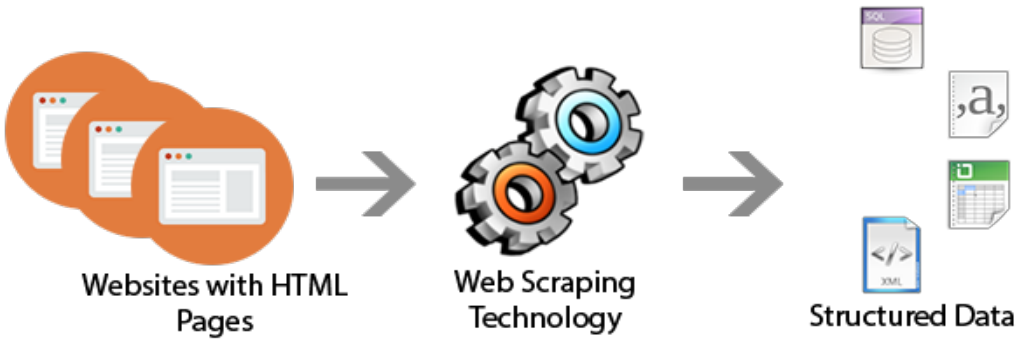
Kapitola 2

Související oblasti

Tato kapitola pojednává o potřebné teorii pro danou problematiku. Počínaje informacemi o extrakci dat z webových stránek neboli web scraping. Dále o popisu uživatelských profilů a jejich obohacování o různé informace, kde jsou popsány i veškeré potřebné technologie. Poslední částí je datová analýza, ve které jsou zmíněné nástroje pro jejich realizaci a strojové učení.

2.1 Extrakce dat z webových stránek

Extrakce dat z webových stránek je známa také pod pojmem web scraping. Základní princip fungování je znázorněn na obrázku 2.1. V dnešním světě je velice populární a využívá se hodně pro datovou analýzu, ale její využití je mnohem rozsáhlejší. Celá sekce čerpá informace ze zdrojů [14, 19, 11, 8].



Obrázek 2.1: Web scraping princip [13].

2.1.1 Původ a legitimnost

Slovo web scraping pochází původně ze slova screen scraping, které označuje získávání informací z terminálových aplikací. Získané informace z těchto sceen scraperů se vkládají dále do jiných aplikací, které je mohou jednoduše využít. S rozšířením internetu se tato technika musela začít přesouvat i do něj. [8, 11]

Původně byl internet pouze pár webových stránek, ale následně se začal značně rozšiřovat a v tu chvíli začalo být těžší a těžší nalézt určité stránky. Proto se velké firmy rozhodly vytvořit slovník pojmenován "Jerry and David's Guide to the World Wide Web", ve kterém byly jednotlivé stránky hierarchicky organizovány. Toto řešení však nebylo dlouhodobě udržitelné ručně, tudíž se musela vymyslet automatizace mapování, z čehož vznikl web scraping. [11]

Legitimnost získávání informací z webových stránek je poměrně nejednoznačná v detailech, avšak ve většině případů je legální. Pokud jsou data využívána pro osobní a neveřejné účely nebo pokud splňují autorská práva, tak to není problematické. Problém nastává až ve chvíli, kdy jsou data znovu publikována s porušením autorských práv nebo pokud je získávání informací natolik zatěžující, že omezí provoz webové stránky. V takové chvíli by se začalo řešit, zdali nedošlo k porušení zákona. [14, 8]

2.1.2 Programovací jazyk

Proces získávání informací z webových stránek je možné vytvořit téměř v jakémkoliv jazyce jako je C nebo Java. Vhodnější jsou jazyky skriptovací jako je Python nebo JavaScript. Nejčastěji se používá jazyk Python.

Python je velmi populární jazyk v dnešním světě, se kterým je možné zvládnout téměř cokoliv. Začal se hodně využívat pro big data a data science, kde spadá také získávání dat z webových stránek. Vzniklo mnoho knihoven pro tento účel jako jsou například Scrapy nebo BeautifulSoup. Díky těmto knihovnám se získávání informací stalo poměrně jednoduchým procesem. Pokud by však bylo potřeba, dá se sestavit celý proces bez využití knihoven třetích stran.

Hlavními důvody, proč je Python vhodným jazykem, jsou především [8]:

- **Jednoduchá syntaxe** - Python má jednu z nejpřehlednějších syntaxí co se týče programovacích jazyků. Programátor se může věnovat více principům vývoje a testování, než psaní samotného kódu, což ve výsledku způsobí kvalitnější práci.

- **Knihovny** - Mnoho užitečných knihoven je již zabudováno přímo v Pythonu, pokud ale není, stačí se podívat na knihovny třetích stran, kterých je nespočetné množství. Vše pro získávání informací z webových stránek se dá implementovat pouze pomocí Pythonu.

- **Open source** - Velikou výhodou je, že Python je open source jazyk. Má velikou podporu celé komunity a může běžet téměř na jakémkoliv zařízení a systému.

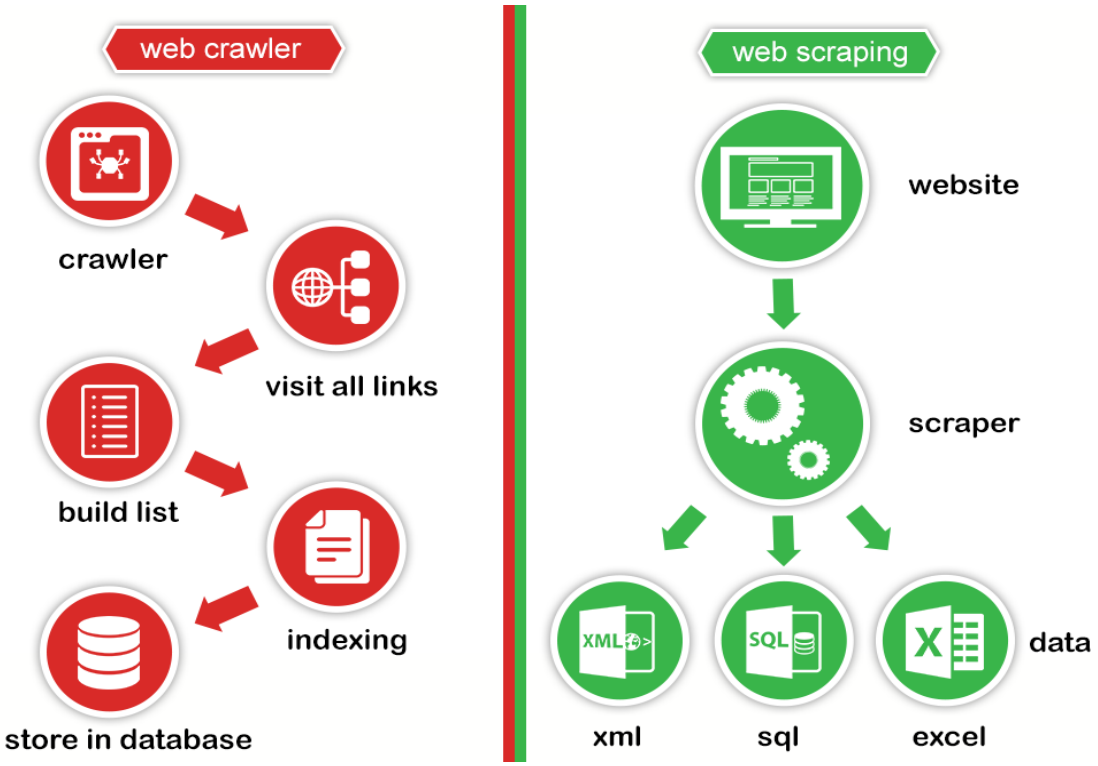
- **Široká škála aplikací** - Python může být použit pro projekty jakéhokoliv rozsahu, od malých skriptů přes různé webové aplikace až po desktopové aplikace velkého rozsahu.

2.1.3 Web scraping nebo web crawling

Web crawling spadá také do tématu extrakce dat z webových stránek. Je důležité znát rozdíl mezi těmito dvěma přístupy pro správné použití. Web crawling není to stejné jako web scraping, ačkoliv se to může plést. Na základě požadovaných informací a webového obsahu je nutné se rozhodnout, zdali je potřeba web scraper nebo web crawler. Jejich rozdíl je však poměrně velký a každý se využívá k něčemu jinému. Obecný rozdíl je možné vidět na obrázku 2.2.

Web scraper je specifický zaměřením pouze na určité webové stránky a málokdy se dá bez nějaké změny nasadit jinde. Získává určitou informaci z obsahu webu, která se může často měnit. V případě, že se změní obsah webové stránky, je potřeba upravit tento nástroj. Hledané informace můžou být například ceny produktů, jména nebo oblíbená jídla v restauraci. [14]

Web crawler se naopak nezaměřuje na specifickou informaci, ale na zmapování webu. Vyhledává ve staženém HTML obsahu relevantní odkazy a poté stahuje i jejich obsah. Výsledkem je tedy množina stažených HTML obsahů pro určité odkazy dané webové stránky. [19]

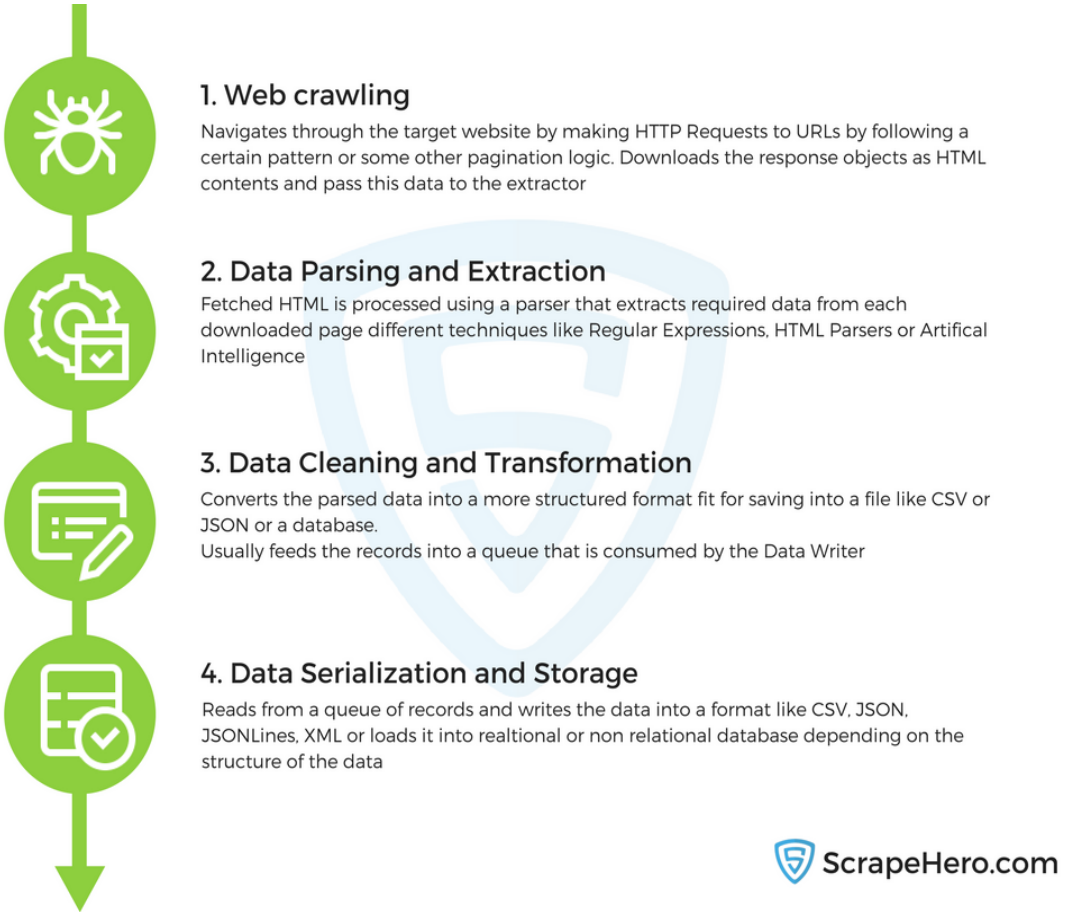


Obrázek 2.2: Web scraping a web crawling [9].

2.1.4 Web scraping a jeho funkcionalita

Funkcionalitu zajišťuje počítačový program nebo skript, pomocí kterého se extrahují informace z webové stránky. Nedílnou součástí je stažení stránky, její zpracování a uložení výsledků. Tento proces je velice podobný ETL (Extract Transform Load) procesu u datových skladů. Extract je stažení obsahu webové stránky, Transform je její zpracování a změna do žádané podoby a Load je závěrečné uložení. [11]

Typické pro extrakci informací z webových stránek jsou čtyři komponenty znázorněny na obrázku 2.3, které na sebe postupně navazují a předávají si mezivýsledky. Tyto komponenty jsou získání webového obsahu, extrakce a zpracování, čištění a transformace a serializace a uložení.



Obrázek 2.3: Jednotlivé části [11].

Krok 1 - získání webového obsahu

Web crawling je systematické stahování obsahů z různých webových stránek, nikoliv jejich zpracování. Modul vytváří HTTP dotazy na určité URL odkazy, které splňují nějaký vzor nebo stránkování. Stažené HTML obsahy dále předává procesu na extrakci. [19]

Příkladem takového modulu může být začátek na domovské stránce nějaké webové stránky a následné stažení HTML obsahů všech možných odkazů v menu nabídce. Tímto se docílí stažení všech hlavních sekcí dané webové domény.

Krok 2 - extrakce a zpracování

Extrakce a zpracování probíhá na dodaném HTML obsahu, ze kterého jsou získány určité části obsahující požadované informace a zpracovány do strukturované podoby.

Aby mohla být tato část úspěšná, je potřeba nejprve provést analýzu webové stránky, ze které se budou informace získávat. V případě složitých webových stránek nemusí být analýza vůbec jednoduchá a navržení extrakce dat společně s realizací je o to složitější. Analýza spočívá v nalezení požadované informace na dané webové stránce a následně zjištění, jakým způsobem se k ní nejlépe dostat. Nalezení struktury HTML obsahu je možné v každém webovém prohlížeči pomocí pomocných nástrojů k tomu určených, jako je například "prozkoumat prvek". Tento nástroj zobrazí HTML kód, kterým se dá přehledně proklikávat a prozkoumat, co je potřeba. Na základě této analýzy se vybere nejvhodnější metoda, která se následně může naimplementovat a začít pro extrakci využívat.

Existují různé typy přístupu extrakce, hlavní jsou tyto 2 [14, 19]:

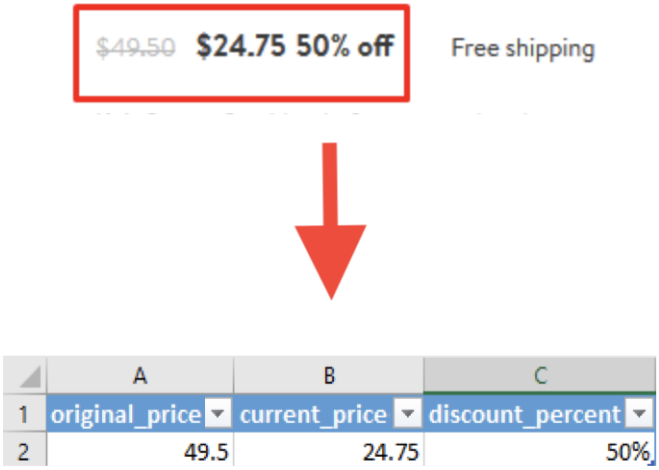
- **Regulární výrazy** - Regulární výrazy jsou nejjednodušším přístupem pro získání určitých informací při extrakci. Pro každý text existuje regulární výraz, který jej dokáže zpracovat dle požadavků. Stačí pouze dobře ovládat syntaxi a sémantiku těchto výrazů, protože nějaká složitější extrakce může vést k složitému a téměř nečitelnému regulárnímu výrazu. Velikou nevýhodou je, že se dá snadno drobnou změnou na webové stránce porušit a v tu chvíli bude potřeba vytvořit nový výraz nebo daný výraz upravit. Výhoda je v rychlosti. Regulární výrazy jsou tedy vhodné pouze pro jednoduché webové stránky, které často nemění obsah, na složitější webové stránky je lepší použít jiný přístup extrakce dat.

- **HTML zpracování** - HTML zpracování je nejrozšířenější přístup pro získání informací z webových stránek. Tento přístup spoléhá pouze na stažený HTML obsah, který si převede do stromové podoby. V případě, že je HTML kód neúplný, je možné jej pomocí různých nástrojů opravit a následně převést. Stromová struktura napomáhá k vyhledávání a navigaci mezi jednotlivými elementy stromu. Technologie pro tuto navigaci jsou především CSS a XPath selektory.

- **Real time zpracování** - Některé webové stránky nezobrazují potřebné informace hned, ale až na základě prokliků pomocí AJAX technologie. V takové chvíli je zpracování HTML obsahu zbytečné, protože tento obsah neobsahuje hledanou informaci, ale pouze strukturu webové stránky. K tomuto se využívají tzv. headless prohlížeče, ve kterých se dá pomocí kódu navigovat a provádět operace. Nejznámějším nástrojem je Selenium, který se primárně obsluhuje za pomocí DOM (Document Object Model) selektorů, jako jsou CSS a XPath.

Krok 3 - čištění a transformace

Proces čištění a transformace probíhá buď zároveň s extrakcí nebo dostává již extrahovaná data z HTML obsahu, která však nemusí být vždy v požadovaném formátu pro následné použití. Proto je potřeba provést různé metody čištění a transformace, do kterých spadá manipulace s textem, úprava do správných jednotek, rozdělení a vytáhnutí určitých částí informace a mnoho dalších. K tomu pomáhají především regulární výrazy, vyhledávací metody a úprava textu. [19]



Obrázek 2.4: Transformace a čištění [11].

Na obrázku 2.4 je znázorněno čištění a transformace pro určitý případ. Extrahovaná část je celá sekce v červeném rámečku. Z této informace je dále potřeba získat 3 různé nové informace, kterými jsou aktuální cena jako `current_price`, původní cena jako `original_price` a sleva v procentech jako `discount_percent`. Nejprve je potřeba pomocí transformace rozdělit tuto informaci na 3 části. Každou z těchto částí je následně potřeba pomocí například regulárních výrazů zpracovat a následně očistit z formátování.

Krok 4 - serializace a uložení

Závěrečná část celého procesu slouží hlavně pro uchování výsledných dat. Její součástí je převedení získaných očištěných dat do požadované podoby, kterou může být například formát JSON, CSV a další. Jakmile jsou data serializovaná, lze s nimi dále nakládat. Nejčastěji se tato data ukládají do souborů na disk nebo do různých databází od MongoDB přes MySQL až po Elasticsearch a Aerospike. Uložením dat se ukončí celý proces a následně se může s těmito daty pracovat dále, ať už pouze informativně nebo pro různorodé analýzy.

2.1.5 Možné využití

Možných využití je celá řada, stačí si vzpomenout na cokoliv, k čemu je potřeba nějaká informace z webových stránek. Jakmile se tato akce zautomatizuje, tak vznikne web scraper. Jakoukoliv informaci je z webových stránek schopen získat člověk, tak stejnou informaci zvládne získat i počítač. Dá se tedy zautomatizovat například hlídání slev pro určitý produkt v obchodě, stahování relevantních novinek z více serverů na základě preferencí nebo třeba periodické stahování dat pro datovou analýzu, která se mohou neustále měnit.

Níže jsou uvedeny některé příklady využití, existuje jich mnohonásobně více [11, 8]:

- **Vyhledávače** - Celý systém vyhledávání na internetu je postavený na získávání informací z webových stránek, bez těchto informací by vyhledávače nebyly schopné zobrazit relevantní výsledky. Na tomto řešení jsou založeny jedny z největších firem jako například Google.
- **Hlídání cen** - Získávání informací o cenách určitých produktů z různých webových obchodů a následné porovnávání, kde jsou produkty nejvýhodnější v danou chvíli nakoupit nebo jak se jejich cena na trhu vyvíjí.
- **Marketing** - Pro marketing je výhodné získávání různých kontaktů, emailových adres, telefonních čísel a profilů ze sociálních sítí. Tato data jsou potom využita pro propagování určitých informací.
- **Agregátory obsahů** - Agregátory projíždějí různé stránky stejného typu a agregují jejich informace dohromady. Příklady takových agregátorů jsou agregátory novinek nebo pracovních nabídek.
- **Data pro výzkum** - Výzkumníci potřebují dostatek dat pro jejich výzkum, který si museli získávat a čistit ručně. Nyní již využívají technik získávání informací z webových stránek pro usnadnění práce a možnost soustředění se na výzkum.
- **Trénovací data pro strojové učení** - Obdobně jako data pro výzkum je potřeba i trénovací data pro učení modelů strojového učení. Ne všechny weby poskytují kvalitní data, proto je potřeba je získat a upravit do potřebné podoby.
- **SEO (Search Engine Optimization)** - SEO nástroje pravidelně prohledávají vyhledávače, aby zjistily co nevíce informací pro určitá klíčová slova a mohly dát relevantní informace o rankingu pro vyhledávání.

2.2 Uživatelské profily

Uživatelské profily jsou využívány pro doporučování internetových reklam ve firmě, ve spolupráci s níž zadání diplomové práce vzniklo (dále jen firma). V databázích této firmy je uloženo za měsíc celkem přes 30 milionů unikátních záznamů uživatelských profilů. Každý takový profil obsahuje velké množství informací nasbíraných o pohybu daného uživatele na internetu. Sbírají se informace hlavně o navštívených stránkách a prohlížených nebo zakoupených produktech v online obchodech. Se všemi těmito daty se dále pracuje a získává se co nejvíce možných informací pro efektivní doporučování reklam.

2.2.1 Technologie

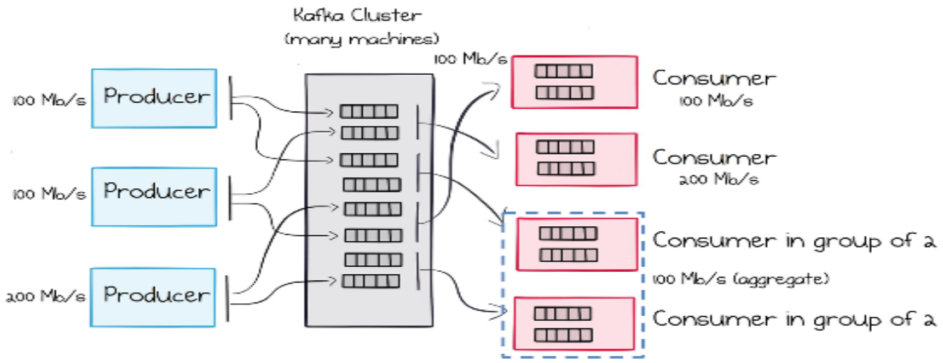
Pro obohacení a uložení potřebných informací o uživatelských profilech je využito několik technologií. Všechny programy zajištující celý proces jsou vytvořeny jako streamovací služby, které jsou postaveny na technologii Kafka. Zakončující programy ukládají vždy výsledná data do dvou databází pro big data, kterými jsou Elasticsearch a Aerospike.

Apache Kafka

Technologie Apache Kafka je streamovací platforma, která se využívá pro předávání informací mezi aplikacemi, jak je vidět na obrázku 2.5. Data jsou ukládána za pomocí technologie Zookeeper pro uchovávání dat v podobě klíč-hodnota, která podporuje sledování a upozornění na změny. Kafka se skládá ze 3 hlavních částí:

- **Producer (producent)** - posílá zprávy
- **Broker (zprostředkovatel)** - uchovává zprávy
- **Consumer (konzument)** - čte zprávy

Pro fungování těchto 3 částí je zapotřebí systém pro zprávy, který je založen na principu publish-subscribe. Tento princip fungování je postaven na přidání se do seznamu příjemců zpráv, kteří dostanou zapsanou zprávu jakmile se objeví. Jádrem je skupina zprostředkovatelů rozprostřená na více serverech, se kterými se komunikuje pomocí API v mnoha jazycích. [3]



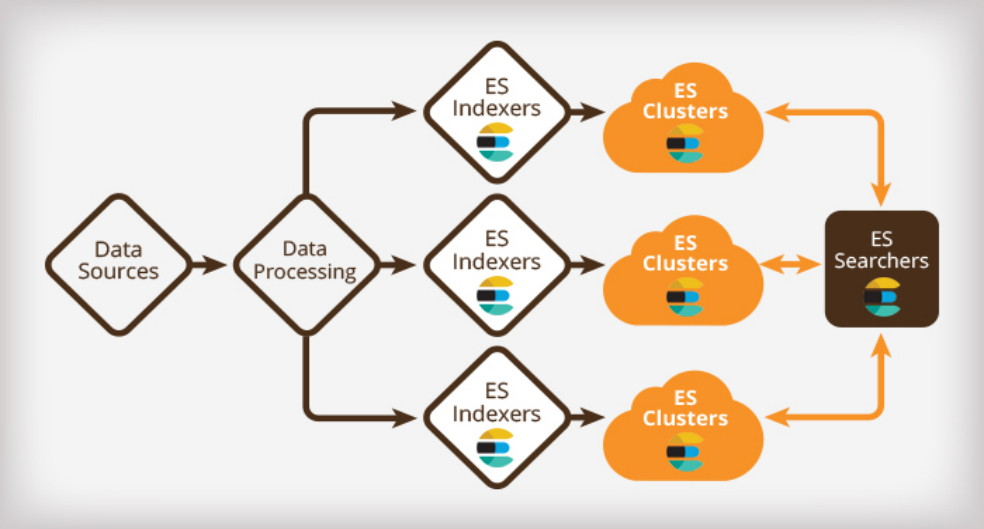
Obrázek 2.5: Streamovací platforma Apache Kafka [22].

Existují zde témata a skupiny konzumentů, které napomáhají lepší škálovatelnosti. Jednotlivá témata jsou unikátní identifikátory pro rozdílné sekvence zpráv. Skupiny určují, zdali chce aplikace dostávat data sama nebo patří do skupiny, která se o data dělí rovným dílem. Každá skupina má pro každé téma vlastní offset na data, která jsou ukládána sekvenčně. Čtení probíhá z daného místa offsetu, zatímco zápis je vždy na konec sekvence. [2, 22]

Všechna data jsou uložena na serverech společně s offsetem pro každého konzumenta a můžou se kdykoliv znovu přečíst. To umožňuje v případě problémů spustit streamovací aplikace znovu a o nic nepřijít, popřípadě restartovat aplikaci a navázat na přerušeném místě, což je velká výhoda této technologie. Nový konzument se může přihlásit k odběru kdykoliv během provozu a zvolit si místo, odkud chce začít číst data. Nejčastěji se nový konzument připojuje na konec sekvence dat, může se však připojit i na začátek nebo na specifickou pozici. V případě přidání konzumenta do již existující skupiny se konzument naváže na offset dané skupiny a začne se dělit o data. [2, 22]

Elasticsearch

Elasticsearch je NoSQL databáze založena na dokumentech, které jsou ukládány ve formátu JSON. Hlavní výhodou Elasticsearch je distribuováné real-time fulltextové vyhledávání, které zvládá zpracovávat obrovské množství dat díky indexaci a rozložení do clusterů (obrázek 2.6). Přistupuje se k němu pomocí RESTful API. Díky vybudování v jazyce Java může být použit na rozdílných distribucích. [5]



Obrázek 2.6: Platforma Elasticsearch [10].

Pro vizualizaci se využívá Kibana, open source platforma pro analýzu a zobrazení dat z Elasticsearch. Kibana komunikuje s Elasticsearch pomocí API stejně jako klienti, vytváří dotazy a dostává odpovědi. Výhodou jsou kromě přehledného zobrazení dat dále grafy, tabulky nebo mapy.

Elasticsearch se skládá z několika částí [4]:

- **Cluster** - Cluster je kolekce jednoho a více serverů (nodů), které dohromady uchovávají data. To umožňuje indexaci a vyhledávání napříč jednotlivými servery. Každý cluster je pojmenován unikátním jménem, aby se jednotlivé servery mohly k danému clusteru připojit.

- **Node** - Node je samostatný server který je součástí nějakého clusteru. Jednotlivé nody uchovávají data samostatně a napomáhají svému clusteru v indexaci a vyhledávání. Každý node má unikátní jméno, které je defaultně vygenerováno jako UUID (Universally Unique IDentifier).

- **Index** - Index je kolekce dokumentů se společnou charakteristikou. Cluster může obsahovat neomezené množství indexů. Každý index nese jméno, které je využito pro indexaci, vyhledávání, aktualizaci a mazání dokumentů. Příklady indexů můžou být produkty a uživatelské profily.

- **Mapping** - Mapping je mapování využité pro indexy. Pro každý index se v mappingu specifikují jednotlivé položky a jejich datový typ pro usnadnění indexace a vyhledávání.

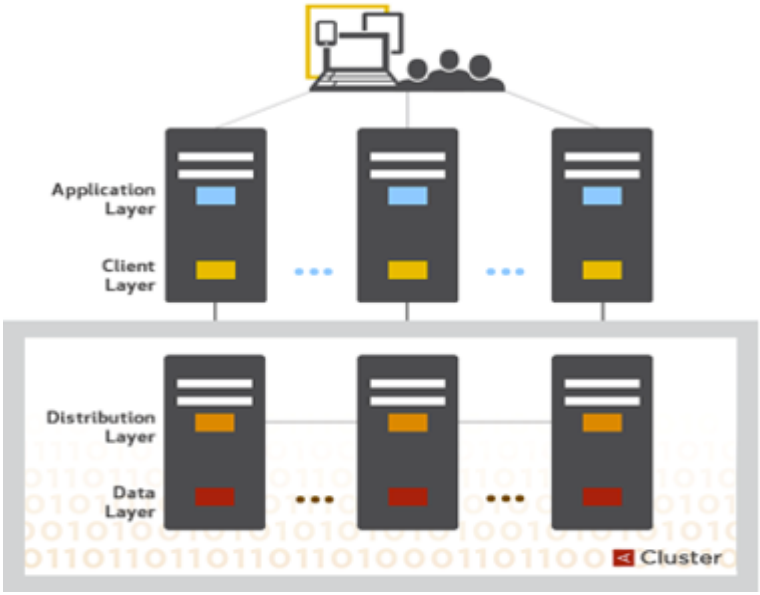
- **Document** - Dokument je kolekce položek ve formátu JSON. Dokumenty jsou jednotlivé záznamy specifikované unikátním identifikátorem UID, které náleží přiřazeným indexům. V jednom indexu může být neomezené množství dokumentů.

- **Shard** - Shard je rozdělení indexů do částí za účelem rozložení zabraného místa na disku. Rozdělení indexů do shardů napomáhá paralelizaci a lepší škálovatelnosti. Jednotlivé shardy pro společný index můžou být uloženy na různých serverech.

- **Replica** - Repliky jsou kopie indexů a shardů, které jsou rozmístěny po rozdílných serverech. Hlavním přínosem replik je přístupnost dat při výpadku některého ze serverů a možnost paralelizace vyhledávání.

Aerospike

Aerospike je databáze typu NoSQL, která ukládá data v podobě klíč-hodnota. Jeho předností je využití DRAM a SSD pro rychlý přístup k datům a jejich distribuované ukládání mezi servery. Pro efektivní rozprostření dat mezi servery jsou využity různé algoritmy, které se o vše postarají bez zásahu uživatele. Aerospike se skládá ze tří vrstev, které mezi sebou komunikují a zařizují funkcionalitu. Architektura je znázorněna na obrázku 2.7. [1]



Obrázek 2.7: Architektura databáze Aerospike [1].

Architektura se skládá ze 3 vrstev [1]:

- **Client Layer** - Klientská vrstva zajišťuje komunikaci s clustery, o kterých má přehled. Sleduje jednotlivé nody a ví, kde jsou data uložena. O změně v konfiguraci clusteru nebo pádu nodu je ihned informována. Klientská vrstva od Aerospike obsahuje API v mnoha různých programovacích jazycích a klient-server protokol pro jednoduchou komunikaci. Tato architektura snižuje transakční latenci, odlehčuje práci pro clustery a zajišťuje, že se aplikace nemusí restartovat v případě problémů s pádem některého nodu.

- **Distribution Layer** - Tato vrstva obsahuje 3 moduly, které se starají o funkcionalitu:

  – **Cluster Management Module** - Pomocí algoritmu zjistí, které nody patří do kterých clusterů a následně je sleduje. Ke sledování napomáhá speciální kontrola stavu.

  – **Data Migration Module** - Modul zajišťuje rovnovážné rozdělení dat mezi všemi nody a replikaci napříč clustery. Automaticky vše upraví v případě přidání nebo odebrání některého nodu.

  – **Transaction Processing Module** - Modul provádí čtení a zápis dat na požadavek, přičemž zaručuje konzistenci a izolovanost. Je odpovědný za synchronizovanou replikaci a odstranění duplicitních dat.

- **Data Layer** - Datová vrstva se stará o uložená data a jejich manipulaci. Data jsou uložena jako klíč-hodnota bez schématu v kontejnerech. Každý záznam má indexovaný unikátní klíč a hodnoty spojené s tímto záznamem. Hodnoty uložené v záznamu jsou silně typované a mohou mít nastaven TTL (Time To Live), který určuje životnost dat. Indexy (primární a vyhledávací klíče) jsou uloženy v DRAM pro rychlý přístup a hodnoty v DRAM nebo na SSD discích.

2.2.2 Informace v profilech

Uživatelské profily obsahují velké množství nasbíraných informací během pohybu uživatele na internetu. Tyto informace jsou různorodé a pro lepší přehlednost jsou níže rozděleny do podobných skupin. Jednotlivé informace mají rozdílný maximální počet uchovaných záznamů. Doba uchování každého záznamu je 30 dní, po uplynutí této doby je záznam smazán.

Webové prohlížeče a zařízení

Zde spadají informace o webových prohlížečích a zařízeních.

- `typ` - Typ zařízení (maximální počet - 2)
- `dev` - Jméno zařízení (maximální počet - 2)
- `OS` - Název operačního systému (maximální počet - 2)
- `br` - Název webového prohlížeče (maximální počet - 2)
- `blng` - Nastavený jazyk v prohlížeči (maximální počet - 10)

Lokality

Zde spadají informace o lokalitách, ze kterých daný uživatel navštívil internet.

- `cr` - ISO kód státu (maximální počet - 30)
- `lId` - Město lokality (maximální počet - 30)
- `lRe` - Region lokality (maximální počet - 30)
- `lCat` - Velikostní kategorie lokality (maximální počet - 30)

Domény

Zde spadají informace o navštívených doménách.

- `dom` - Domény druhého řádu (maximální počet - 50)
- `sub` - Domény třetího řádu (maximální počet - 50)
- `topD` - Nejvíce navštěvované domény (maximální počet - 50)

Extrahované informace

Zde spadají informace získané z obsahů webových stránek nebo jejich URL adres.

- `WebC` - Kategorie získané na základě URL (maximální počet - 50)
- `KW` - Klíčová slova získaná z URL (maximální počet - 50)
- `sex` - Pohlaví na základě chování (maximální počet - 50)
- `brnd` - Oblíbené značky (maximální počet - 50)

- `WebCtg` - Kategorie na základě obsahu webové stránky (maximální počet - 50)
- `WebKw` - Klíčová slova z obsahu webové stránky (maximální počet - 50)
- `WebNe` - Pojmenované entity z obsahu webové stránky (maximální počet - 50)
- `WebPh` - Fráze z obsahu webové stránky (maximální počet - 50)

Produkty

Zde spadají informace získané z internetových obchodů o produktech.

- `Id` - ID produktu (maximální počet - 30)
- `Name` - Název produktu (maximální počet - 30)
- `Cat` - Kategorie produktu (maximální počet - 30)
- `Pr` - Cena produktu (maximální počet - 30)
- `PC` - Cenová kategorie produktu (maximální počet - 30)
- `Tag` - Tag produktu (maximální počet - 30)
- `ShopId` - ID obchodu (maximální počet - 30)

Informace o produktech se dělí na 3 kategorie:

- **Navštívené produkty** - prefix `"v"`
- **Produkty vložené do košíku** - prefix `"c"`
- **Zakoupené produkty** - prefix `"o"`

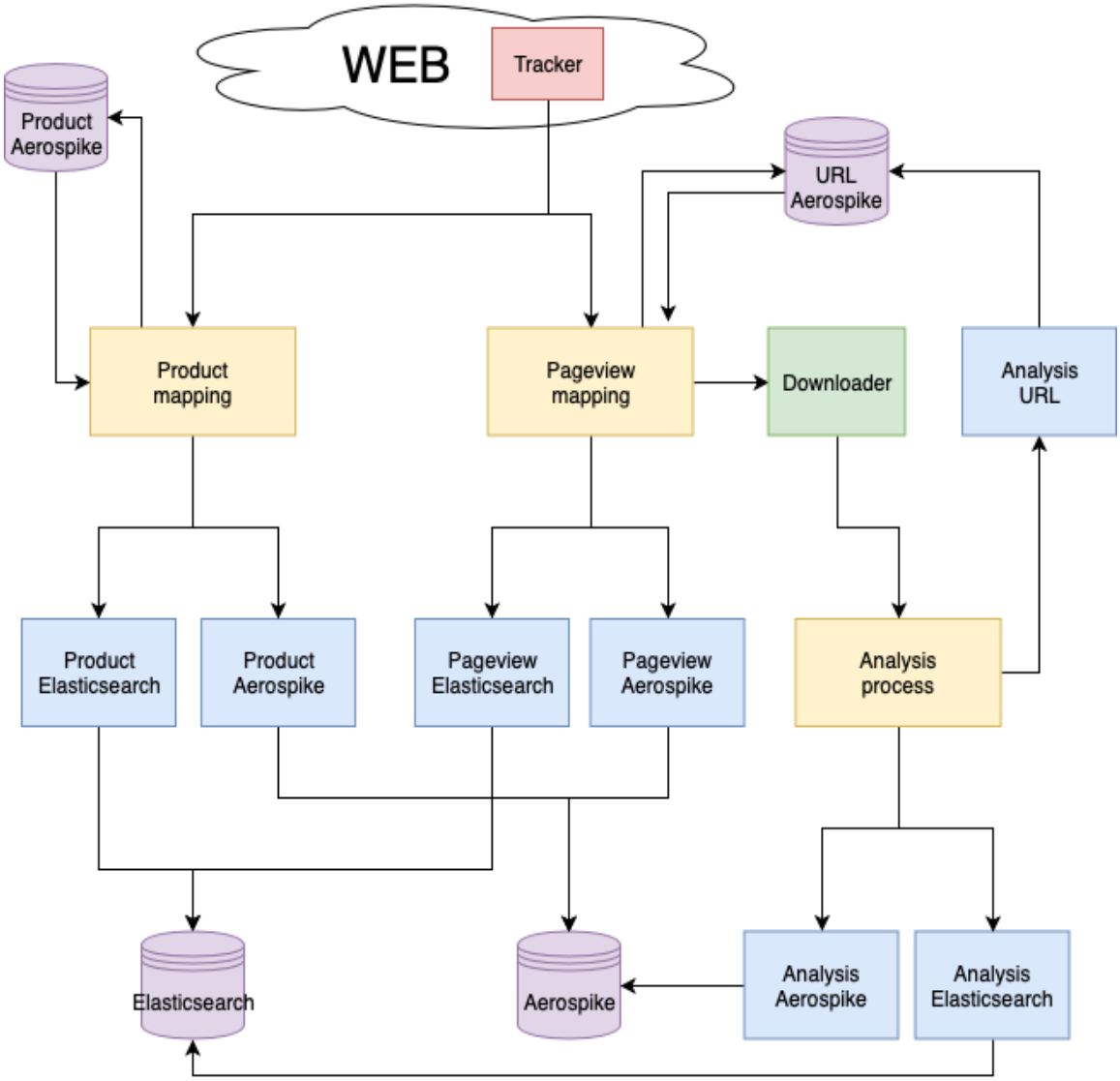
Dodatečné informace

Zde spadají ostatní informace.

- `tsg` - Seznam zacílených kampaní (maximální počet 100)
- `is_abroad` - Profil je aktuálně v zahraničí
- `is_traveler` - Profil cestuje ve své zemi
- `is_international` - Profil cestuje do zahraničí
- `is_bot` - Profil je robot
- `first_seen` - První objevení profilu
- `last_seen` - Poslední objevení profilu
- `viewed_campaigns` - Zobrazené kampaně
- `imps` - Počet zobrazených impresí pro jednotlivé kampaně
- `clicks` - Počet kliknutí na jednotlivé kampaně

2.2.3 Proces obohacování profilů

Proces obohacování začíná sledovacím skriptem na webových stránkách, který se chová jako web crawler. Tento skript posílá informace o navštívení stránky a produktu streamovacím aplikacím, které jsou postaveny na technologii Apache Kafka. Produktový stream spolupracuje s produktovou databází Aerospike a obohacuje přijatá data informacemi o produktech. Stream pro zhlédnuté stránky se označuje jako `pageview` a spolupracuje s URL databází Aerospike, která funguje jako cache extrakce již zpracovaných stránek. Pokud se zde informace nenachází, posílá se webová stránka na stáhnutí a zpracování. Všechny tyto části jsou zakončeny dvěma streamy na uložení dat do profilových databází Elasticsearch a Aerospike. Celý proces je znázorněn na 2.8.



Obrázek 2.8: Proces obohacování uživatelských profilů.

2.2.4 Požadované informace

Požadované nové informace pro další práci jsou obsaženy v obsahu určitých webových stránek. Informace získané z těchto webových stránek jsou potřeba uložit k uživatelským profilům. V této diplomové práci se jedná o 2 webové servery, které byly v rámci zadání zvoleny firmou:

- **ČSFD (csfd.cz)** - ČSFD je filmová databáze, kde lidé hledají informace o filmech. Filmy obsahují základní informace, ze kterých je potřeba vytáhnout žánry.
- **Recepty (recepty.cz)** - Recepty jsou databáze receptů, kde lidé hledají různá jídla k uvaření. Recepty spadají do mnoha kategorií, které je potřeba získat.

Požadované informace spadají do kategorie extrahované informace a jejich název bude `WebExt` neboli web extras. Budou zde obsaženy 2 různé věci:

- Žánry filmů
- Kategorie receptů

2.2.5 Cílové skupiny

Cílovými skupinami se rozumí skupiny uživatelů pro doporučování reklam. Na základě požadované reklamní kampaně se z databáze uživatelských profilů vybere určité množství nejvíce relevantních profilů, kterým se cílená reklama nabízí. Tento proces je napůl automatizovaný a využívají se pro něj metody strojového učení. Pro lepší analýzu a výběr profilů by měly napomoci nové informace popsané v 2.2.4.

Aktuální proces vybírání uživatelských profilů pro cílenou reklamu se dělí na 2 části. První částí je vybrání prvotního vzorku uživatelů vyhovující zadání cílené reklamy od různých firem a rozhodnutí, které informace jsou relevantní a které zbytečné v kontextu kampaně. Jsou to například uživatelé, kteří si zakoupili určitý produkt. Druhou částí je vymyšlení principů datové analýzy, které budou vybranému vzorku vyhovovat a zaručí tak nejlepší možnou cílovou skupinu. Následně provedení celé analýzy, kde výstupem je systematický váhový dotaz do Elasticsearch databáze, který vrátí uživatelé spadající do cílové skupiny dané reklamní kampaně.

Hlavní 2 přístupy ke tvorbě cílových skupin:

- **Průměrný uživatel** - Z vybraného prvotního vzorku uživatelů se na základě jejich profilů a chování vytvoří průměrný uživatel. Ostatní uživatelské profily se s tímto průměrným uživatelem porovnávají v oblasti jednotlivých informací a poměrově se určuje jejich podobnost.
- **Shluky uživatelů** - Pro prvotní vzorek uživatelů se vytvářejí jednotlivé shluky pomocí strojového učení. Ke každému shluku se vyhledávají z databáze uživatelské profily, které spadají do blízkosti těchto shluků.

Tvorba průměrného uživatele a shluky uživatelů jsou poměrně náročné na výpočty, protože musí zpracovat obrovské množství dat a využít při tom algoritmy strojového učení. Proto se provádí pouze jednou za čas a ne každý den. Výsledné cílové skupiny to výrazně neovlivňuje, protože průměrný uživatel ani shluky uživatelů by se neměly pro stejné zadání časem příliš lišit.

2.3 Strojové učení

Strojové učení je odvětví umělé inteligence, které se velice rozšířilo. Cílem je vytvořit počítačový program, který se dokáže sám učit z dodaných dat a následně adekvátně reagovat na požadavky. K tomu je využito trénovaní vybraného modelu na předzpracovaných datech. Problémem je malá množina trénovacích dat, protože model má v takovém případě omezené možnosti učení. Mezi hlavní podkategorie strojového učení patří učení s učitelem a učení bez učitele. Těmto kategoriím předchází předzpracování, které je velice důležité pro zlepšení učení. Celá sekce čerpá ze především ze 3 zdrojů [21, 15, 24].

2.3.1 Knihovny

Pro usnadnění celého procesu datové analýzy v této práci se využívá několik potřebných knihoven v jazyce Python. Každá z těchto knihoven je využita pro rozdílné potřeby. Pandas se využívá pro reprezentaci dat ve strukturách a jejich manipulaci. Scikit-learn obsahuje mnoho algoritmů pro strojové učení. Imbalanced-learn je zaměřen na vyrovnání tříd v datech. Catboost obsahuje specifické klasifikační modely. Matplotlib je knihovna pro vizualizaci.

Pandas

Pandas je knihovna vytvořena pro rychlou, jednoduchou a expresivní práci s daty. Data jsou uchovávána ve strukturovaném tabulkovém formátu, který je striktně typovaný. Základní Pandas objekty, které se nejčastěji využívají jsou [17]:

- **DataFrame** - Tabulka, která obsahuje označené sloupce a řádky.
- **Series** - Jednodimenzionální datové pole s označením.

Pandas obsahuje množství funkcí, které napomáhají v importování dat z různých datových zdrojů, jako jsou například CSV soubory. Použitím těchto funkcí se naplní DataFrame potřebnými daty, která mají pojmenované jednotlivé řádky a sloupce.

Základem funkčnosti je knihovna NumPy, která umožňuje vysoký výkon při práci s poli. NumPy využívá indexování pro jednoduchou manipulaci s daty, jako je ořezání, přetvoření, agregace nebo výběr části dat. [17]

Scikit-learn

Scikit-learn je knihovna v oblasti strojového učení včetně předzpracování pro Python programátory, která je velice populární. Tato knihovna obsahuje mnoho implementací algoritmů ve všech kategoriích. Scikit-learn má 6 částí, kde každá pokrývá algoritmy určité oblasti strojového učení: **Klasifikace**, **Regrese**, **Shlukování**, **Redukce dimenzionality**, **Selekce modelu**, **Předzpracování**. [17]

Imbalanced-learn

Imbalanced-learn je knihovna v oblasti strojového učení pro vypořádání se s nepoměrem klasifikačních tříd v datech. Knihovna se dělí na 2 hlavní části, kterými jsou `over-sampling` a `under-sampling`. Každá z těchto částí obsahuje mnoho algoritmů. `Over-sampling` se stará o rozmnožení méně početné třídy. `Under-sampling` se stará o redukci více početné třídy. [6]

Catboost

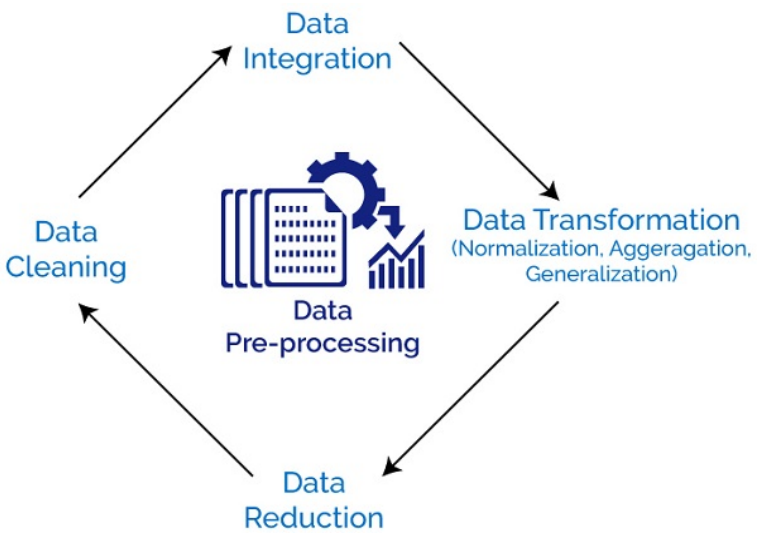
Catboost je knihovna v oblasti strojového učení obsahující modely posílených rozhodovacích stromů. Model pro klasifikaci je `CatBoostClassifier` a pro regresi `CatBoostRegressor`. Výhodou této knihovny je především výkonnost učení a kvalita predikce bez potřeby hledat nejlepší možné parametry. [7]

Matplotlib

Matplotlib je knihovna pro vizualizaci dat, která byla vytvořena primárně pro tvorbu grafů do publikací. Postupem času se začala využívat pro zobrazení dvoudimenzionálních dat všeho druhu. Knihoven pro vizualizaci existuje celá řada, matplotlib vítězí díky výborné integraci do celého ekosystému. Proto je to bezpečná volba při výběru nástroje pro vizualizaci. [17]

2.3.2 Předzpracování

Proces předzpracování je velice důležitý pro strojové učení. Pokud jsou data nekvalitní, bude nekvalitní i natrénovaný model na takových datech. Pro zkvalitnění dat napomáhá právě předzpracování, ve kterém probíhají 4 etapy. První etapa je čištění, druhá integrace, třetí redukce a čtvrtá transformace. Tyto etapy mohou probíhat v různém pořadí a v několika cyklech, jak je vidět na obrázku 2.9. Po provedení těchto etap by měla být data o poznání kvalitnější a modely strojového učení by je měly lépe zpracovat.



Obrázek 2.9: Proces předzpracování [18].

Čištění

Čištění napomáhá zkvalitnění dat. Důvody pro nekvalitní data můžou být různé, nejčastější jsou však následující 3 [24]:

- **Chybějící hodnoty** - U některých záznamů mohou chybět hodnoty atributů. Ošetření má 3 možnosti, kterými jsou ignorování záznamu, ruční doplnění a automatické doplnění.

- **Zašuměná data** - Za šum se považuje náhodná chyba nebo odchylka v hodnotách. Vyhlazení šumu se může provést na základě okolních hodnot, regresní funkcí a detekcí odlehlých hodnot.
- **Nekonzistentní a redundandní data** - Integrací dat z různých zdrojů může vzniknout nekonzistence a redundance. Pro detekci takových dat se využívá například korelační analýza.

Integrace

Při získávání datové sady probíhá integrace z více uložišť. Jedná se o kombinaci dat z několika zdrojů do jednoho konečného. Problémy nastávající při integraci jsou nekonzistence a redundance zmíněné v čištění a dále [24]:

- **Identifikace entit** - Pro entity mohou nastat problémy v identifikaci z více zdrojů. Je tedy potřeba tyto identifikátory sjednotit.
- **Konflikt hodnot** - Pro stejné entity mohou být rozdílné hodnoty atributů z různých zdrojů. Takové hodnoty je nutné vyřešit a vybrat správné.

Redukce

Redukce se provádí pro zmenšení datové sady, aby se zabránilo časové náročnosti výpočtů. Cílem je získat menší počet dat a jejich atributů při zachování téměř stejných výsledků při analýze. Redukce se dělí na 2 kategorie podle přístupu [24]:

- **Redukce dimenzionality** - Redukcí dimenzionality se myslí snížení počtu atributů bez znehodnocení výsledků. Využívá se zde konstrukce (tvorba nových atributů), extrakce (transformace atributů do nového prostoru) a výběr atributů (hledání nejlepší podmnožiny atributů).
- **Redukce počtu záznamů** - Redukce počtu objektů se využívá pro zmenšení datové sady v počtu prvků. Hlavním přístupem při výběru objektů je vzorkování, které má mnoho podob.

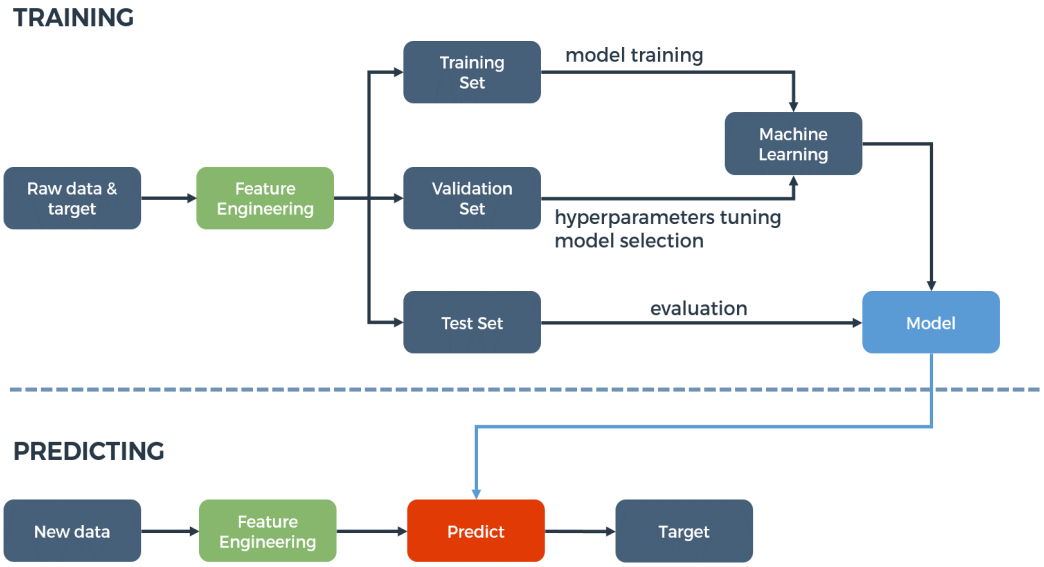
Transformace

Transformace zajišťuje převedení dat do vhodné podoby pro analýzy a učení modelů. Transformace má mnoho možností úpravy dat, příkladem jsou [24]:

- **Normalizace** - Normalizace je převedení kvantitativních hodnot do potřebného intervalu. Využívá se především min-max normalizace, z-score normalizace a normalizace změnou dekadického měřítka.
- **Diskretizace** - Diskretizace je proces převedení kvantitativních atributů na intervaly. Metody pro diskretizaci jsou například plnění nebo shlukování.

2.3.3 Učení s učitelem

Učení s učitelem je proces vytváření modelu strojového učení, který je založen na označkovaných datech. Označkovanými daty se myslí množina dat, u kterých je znám výsledek. Taková data jsou v tomto procesu považována za zmíněného učitele. Model se pomocí informací obsažených v těchto datech naučí predikovat. Princip učení s učitelem je znázorněn na obrázku 2.10. Nejdříve se zpracují data, která se rozdělí do 2 nebo 3 skupin, podle toho zdali se využívají validační data pro zlepšení algoritmu. Na trénovacích, respektive i validačních datech se model naučí predikovat a na testovacích datech se predikce otestuje. Následně může být model využit pro predikci dat. Učení s učitelem se dělí na 2 skupiny, kterými jsou klasifikace a regrese. [21, 15]



Obrázek 2.10: Princip učení s učitelem [12].

Klasifikace

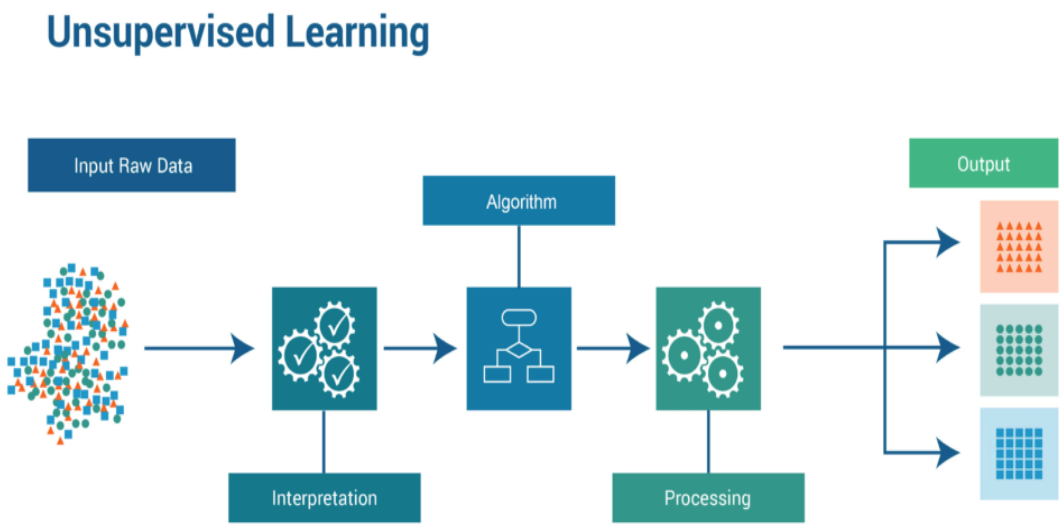
Klasifikace je typ strojového učení s učitelem, který rozděluje data do několika tříd. Vstupní množina dat obsahuje označkovaná data výslednými třídami. Na základě jednotlivých informací obsažených v datech se modely snaží naučit predikovat výslednou třídu. Klasifikace je nejrozšířenější typ strojového učení, tudíž existuje mnoho principů a algoritmů. Mezi nejznámější principy patří rozhodovací stromy, neuronové sítě, bayesovské sítě a SVM (Support Vector Machine). [21, 24]

Regrese

Regrese neboli regresní analýza je typ strojového učení s učitelem, který predikuje hodnotu kvantitativního atributu. Regresní analýza se zabývá nalezením regresní funkce pro vztah mezi vstupními hodnotami a výstupní hodnotou. Základním přístupem je lineární regrese, která slouží k proložení množiny bodů v grafu přímkou. Dalšími přístupy jsou vícenásobná lineární regrese a nelineární regrese, která je poměrně složitá. [21, 15]

2.3.4 Učení bez učitele

Učení bez učitele je proces vytváření modelu strojového učení, který na rozdíl od učení s učitelem nepožaduje označkovaná data. Model dostává pro naučení data, která nikdo neohodnotil a neví se u nich, jaký je výsledek. Výsledek učení pomůže získat významné informace bez pomoci neboli bez učitele. Princip učení bez učitele je znázorněn na obrázku 2.11. Vstupní data nejsou označena a tudíž není znám jejich výsledek, proto se zde nepoužívají ani testovací data. Model vstupní data rozdělí do tříd na základě zvoleného algoritmu. Do učení bez učitele patří především shlukování. [21, 15]



Obrázek 2.11: Princip učení bez učitele [20].

Shlukování

Shlukování je typ strojového učení bez učitele, který rozděluje data do tříd. Jednotlivé třídy jsou tvořeny na základě podobnosti mezi objekty. Mezi hlavní přístupy shlukování patří [21, 24]:

- **Metody založené na rozdělování** - Metody rozdělují objekty do určitého počtu tříd. Celý proces funguje iterativně, kde se nejdříve vyberou objekty jako středy shluků a následně se k nim přiřadí zbývající objekty na základě podobnosti. V dalších krocích se vylepšují středy shluků a přesouvají objekty. Existují metody založené na centrálním bodu a na reprezentujícím objektu.

- **Hierarchické metody** - Metody vytvářejí hierarchický rozklad objektů na základě vzdálenostní funkce. Existují metody shlukující a rozdělující, které se liší ve směru hierarchie. Shlukující metody nejdříve udělají z každého objektu samostatnou třídu a tyto třídy následně shlukují. Rozdělující metody začínají od jedné třídy všech objektů a postupně ji rozdělují.

- **Metody založené na hustotě** - Metody vytvářejí třídy na základě hustoty mezi objekty. Objekty, které jsou nahuštěné na sobě patří do stejné třídy. Jednotlivé třídy jsou rozděleny oblastmi s nízkou hustotou objektů. Objekty v takové oblasti jsou hodnoceny jako šum. Narozdíl od ostatních metod si tyto metody dokážou poradit se šumem a odlehlými hodnotami.

2.3.5 Klasifikační metriky

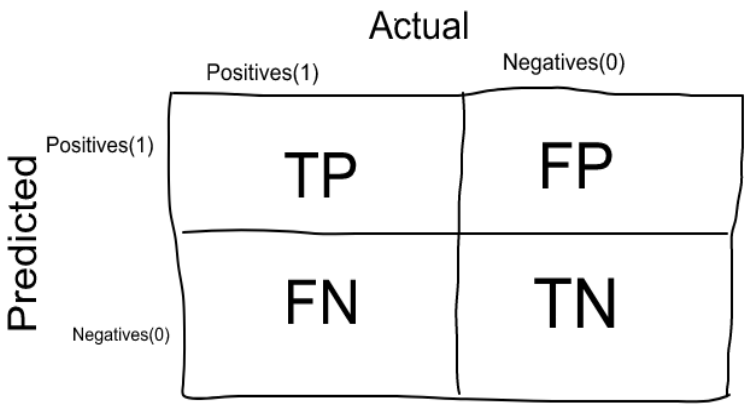
Klasifikační metriky jsou využity pro vyhodnocení kvality naučeného klasifikačního modelu na testovacích datech. Volba klasifikační metriky je velice důležitá, protože každá se zaměřuje na trochu jinou oblast měření. Většina klasifikačních metrik je založených na klasifikační matici, která přehledně zobrazuje třídy ohodnocení mezi reálnými a predikovanými daty. [23]

Klasifikační matice

Klasifikační matice je tabulka se dvěma dimenzemi (Aktuální a Predikovaná). V každé z dimenzí jsou všechny třídy, které je možné v datech nalézt. Na obrázku 2.12 je vidět klasifikační matice, kde řádky jsou predikce a sloupce aktuální data.

Hodnoty z klasifikační matice využívá většina klasifikačních metrik pro své výpočty. Jednotlivé položky klasifikační matice s binárními třídami, označenými jako `0` a `1`, jsou [23]:

- **True pozitivní (TP)** - True pozitivní jsou ta data, která jsou ve třídě `1` a jejich predikce je také `1`. Znamená to správnou predikci pozitivní třídy.
- **True negativní (TN)** - True negativní jsou ta data, která jsou ve třídě `0` a jejich predikce je také `0`. Znamená to správnou predikci negativní třídy.
- **False pozitivní (FP)** - False pozitivní jsou ta data, která jsou ve třídě `0` a jejich predikce je `1`. Znamená to špatnou predikci pozitivní třídy.
- **False negativní (FN)** - False negativní jsou ta data, která jsou ve třídě `1` a jejich predikce je `0`. Znamená to špatnou predikci negativní třídy.



Obrázek 2.12: Klasifikační matice [23].

Metriky

Metrik založených na klasifikační matici je mnoho, zde jsou některé z nich [23]:

- **Accuracy (správnost)** - Správnost je založena na celé klasifikační matici. Jejím výsledkem je přesnost správné predikce, která se vypočítá jako: $Accuracy = (TP + TN)/(TP + FP + FN + TN)$. Metrika je vhodná k použití v případě dobrého poměru tříd v datech.

- **Precision (přesnost)** - Přesnost je založena na `TP` a `FP`. Jejím výsledkem je preciznost predikce pro pozitivní třídu. Vypočítá se jako: $Precision = TP/(TP + FP)$. Metrika je vhodná k použití v případě důrazu na co nejpřesnější predikci pro pozitivní třídu.

- **Recall (pokrytí)** - Pokrytí je založen na `TP` a `FN`. Jejím výsledkem je míra pokrytí predikce pro pozitivní třídu. Vypočítá se jako: $Recall = TP/(TP + FN)$. Metrika je vhodná k použití v případě důrazu na co největší počet správně predikovaných pozitivních tříd.

- **F1 (míra F1)** - Míra F1 je založena na `Precision` a `Recall`. Jejím výsledkem je poměr mezi `Precision` a `Recall` vzhledem k jejich hodnotám. Vypočítá se jako: $F1 = (2 * Precision * Recall)/(Precision + Recall)$. Metrika je vhodná k použití v případě potřeby obou předchozích metrik zároveň.

Kapitola 3

Návrh řešení

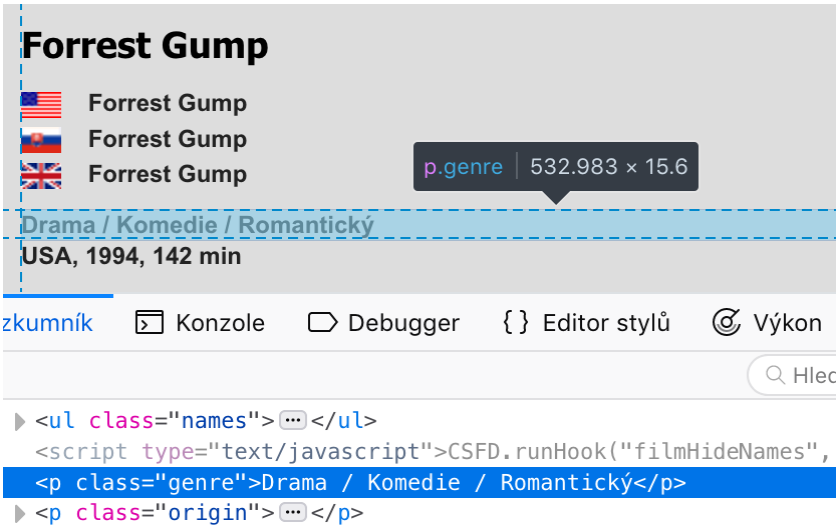
V této kapitole jsou popsány návrhy k realizaci obohacování uživatelských profilů. Nejprve je ukázána analýza webových serverů obsahujících požadované informace. Následně je navrhnut proces obohacení uživatelských profilů s extrakcí informací a jejich zpracováním.

3.1 Analýza webu a požadovaných informací

Pro kvalitní extrakci informací z webové stránky je nejprve potřeba ji pořádně prozkoumat a zjistit, kde a v jakém formátu se požadovaná data nachází. Tato analýza se dá provést pomocí kteréhokoliv webového prohlížeče s funkcí pro prozkoumání HTML obsahu. Níže jsou uvedeny analýzy pro 2 webové servery, na kterých se nachází nové požadované informace specifikované v 2.2.4.

3.1.1 ČSFD

Pro server ČSFD s filmovou databází je nutné získat žánry pro jednotlivé filmy. Žánry se nachází na stránkách určených každému filmu zvlášť. Po načtení jedné z těchto stránek jsou přehledně vidět v levém horním panelu, ze kterého bude potřeba je extrahovat.

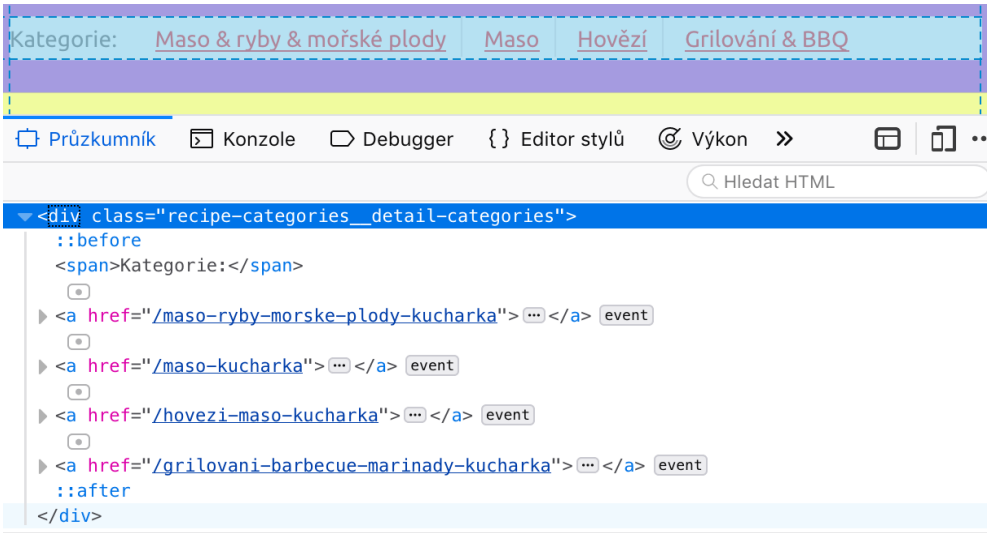


Obrázek 3.1: Analýza filmových žánrů webového serveru ČSFD.

Na obrázku 3.1 je zobrazen výřez stránky filmu s HTML obsahem a zaznačeným výběrem požadované části. Díky zaznačení a nahlédnutí do HTML kódu je možné vidět, jakým způsobem jsou žánry zobrazeny a co bude dále potřeba udělat. Žánry jsou v HTML značce pro paragraf `p` se třídou `genre`, což zajišťuje unikátní vybrání této části. Není zde žádná další HTML značka a zbylý text se nachází v jednoduché podobě.

3.1.2 Recepty

Pro server Recepty s databází receptů je nutné získat kategorie jednotlivých receptů. Kategorie se nachází na stránkách určených každému receptu zvlášť. Po zobrazení stránky jsou ve spodní části pod receptem ve vlastním panelu obsahujícím nadpis "Kategorie".



Obrázek 3.2: Analýza filmových žánrů webového serveru Recepty.

Na obrázku 3.2 je zobrazen výřez stránky receptu s HTML obsahem a zaznačeným výběrem požadované části. Díky zaznačení a nahlédnutí do HTML kódu je možné vidět, jakým způsobem jsou kategorie zobrazeny a co bude dále potřeba udělat. Kategorie jsou v HTML značce `div` se třídou `recipe-categories__detail-categories`, což v tomto případě nezajišťuje unikátní vybrání této části. Na stránce se nachází ještě další sekce se stejným označením, rozdíl je pouze v nadpise "Zařazení". Tento problém se dá vyřešit načtením obou částí a vybrat tu, která nese nadpis "Kategorie". Text kategorií se nachází v jednotlivých HTML značkách `a`, ze kterých se dá text jednoduše vytáhnout.

3.2 Obohacení uživatelských profilů

Obohacení uživatelských profilů se v rámci návrhu dělí na několik částí, kde každá je popsána samostatně. První částí je mapování informací do obecnějšího formátu. Druhou částí je přidání položky do uložiště webových stránek. Třetí částí je navržení celého procesu získání informací z webové stránky a poslední čtvrtou částí je výsledné obohacení profilů.

3.2.1 Mapování informací

Mapování informací napomáhá ukládání přehlednějšího formátu informace a zároveň s menším objemem. Je potřeba vytvořit nástroj, který bude toto mapování zprostředkovávat a bude jej možné využít všemi směry transformace. Níže je uveden stručný přehled některých dostupných informací pro extrakci:

- **ČSFD** - Akční, Animovaný, Dobrodružný, Dokumentární, Drama, Komedie, Muzikál, Pohádka, Rodinný, Romantický, Sci-Fi, Sportovní, Thriller, Životopisný
- **Recepty** - Brambory, Dezerty, Maso, Polévky, Předkrmy, Rizota, Ryby, Rýže, Saláty, Shake & smoothies, Svíčková, Špagety, Zákusky

Každá informace bude ze standardní podoby mapovaná do dvou formátů, kterými jsou **slug** a **abbreviation**. Tyto formáty se používají pro některé dlouhé názvy v celém systému. Slug obsahuje pouze malá písmena, čísla a podtržítko. Abbreviation je nejkratší možná reprezentace. K extrahované informaci se přidá název webového serveru, ze kterého byla získána. Mapování bude vypadat následovně:

- **Standardní** - `CSFD: Akční`, `Recepty: Brambory`
- **Slug** - `csfd_akcni`, `recepty_brambory`
- **Abbreviation** - `csfd:akc`, `rcp:bram`

3.2.2 Získání informací

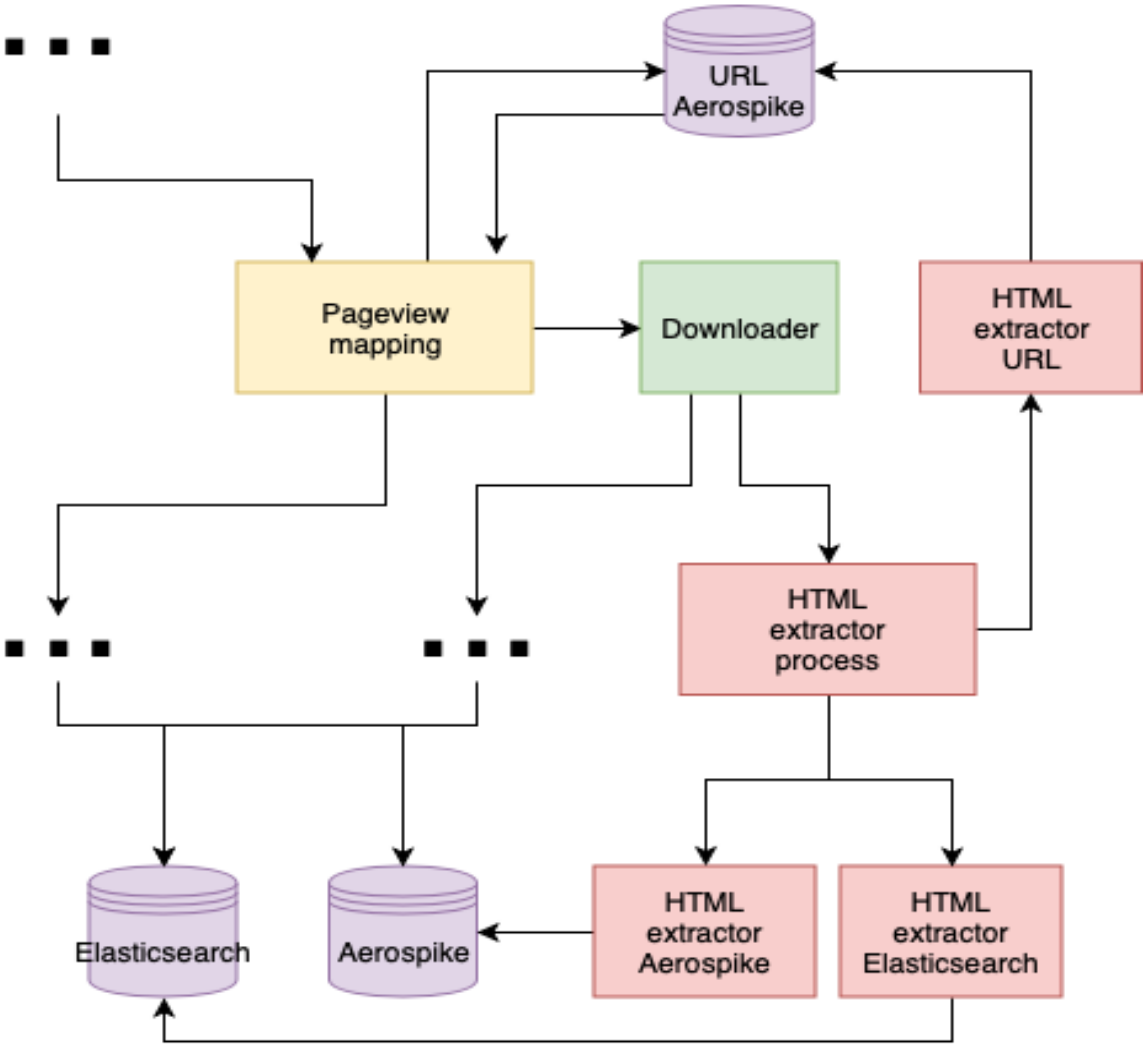
Pro získání informací z webových stránek je potřeba vytvořit dvě streamovací aplikace. Jedna aplikace bude dostávat stažený HTML obsah a z něj extrahovat potřebné informace. Druhá aplikace bude ukládat tato data do URL databáze pro následné využití. Obě aplikace jsou znázorněny na obrázku 3.3.

HTML extrakce

Stream pro extrakci dat z HTML obsahu bude dostávat stažená data ze streamu na stahování. Tato data je potřeba zpracovat pomocí některého principu extrakce a vytáhnout potřebné informace. Vytažené informace je dále potřeba očistit a dostat do strukturované podoby pro následné uložení. V rámci očištění probíhá také mapování informací z 3.2.1. Očištěná data se posílají dalším streamům pro jejich uložení.

Uložení do URL databáze

Stream pro uložení dat do URL Aerospike databáze bude dostávat extrahovaná a očištěná data ze streamu na HTML extrakci. Tato data uloží do URL databáze, která stojí na principu klíč-hodnota. Klíčem v databázi je URL adresa a hodnotou extrahovaná data. Tato databáze slouží jako cache pro data, aby se nemusela pořád dokola extrahovat.



Obrázek 3.3: Nové streamovací aplikace v procesu obohacování uživatelských profilů.

3.2.3 Obohacení profilů

Pro obohacení profilů je potřeba vytvořit 2 streamovací aplikace a další 3 upravit. Dále je potřeba upravit Elasticsearch a Aerospike nastavení, aby očekávaly tato data a uměly je zpracovat. Nové 2 streamy budou ukládat extrahovaná data z HTML obsahu do Elasticsearch a Aerospike profilové databáze. Upravit se musí mapovací stream pro zhlédnuté stránky, který bude brát data z URL databáze. Dále se musí upravit streamy pro ukládání zhlédnutých stránek do databází. Vše je možno vidět na obrázcích 2.8 a 3.3.

Úprava databází

Pro efektivní ukládání do databází Elasticsearch je nutné mít správně nastavené schéma a pro Aerospike soubor s metadaty. Pro Elasticsearch je schéma uloženo přímo na serveru, se kterým se komunikuje pomocí API. Pro Aerospike jsou metadata uložena v souboru `yaml` a využívají se až při klientské aplikaci. Ve schématu Elasticsearch stačí přidat název pole s datovým typem. Metadata pro Aerospike by měla být obohacena o počet a čas udržování jednotlivých dat.

Mapování zhlédnutých stránek

Stream na mapování zhlédnutých stránek je potřeba upravit, aby byl schopen obohatit informace o extrahovaná data z HTML obsahu. Hlavní úpravou zde bude vytažení těchto dat z URL databáze, ve které se mohou nacházet. V případě, že se zde data nenachází, je potřeba poslat danou webovou stránku na zpracování.

Uložení zhlédnutých stránek

Streamy na ukládání zhlédnutých stránek, které ukládají informace do databází Elasticsearch a Aerospike, je potřeba upravit. Přidáním funkcí na uložení nových informací do těchto streamů se umožní jejich uložení.

Uložení extrahovaných stránek

Pro extrahovaná data z webových stránek je potřeba vytvořit 2 nové streamy, které tato data uloží do databází Elasticsearch a Aerospike. Tyto 2 streamy budou obdobné streamům na uložení zhlédnutých stránek, pouze budou ukládat jiné informace a získávat data ze streamu na extrakci dat z HTML obsahu.

Kapitola 4

Implementace

Tato kapitola popisuje implementaci, která navazuje na návrh v kapitole 3. Nejdříve je popsána implementace mapování informací do různých podob. Dále je popsána streamovací knihovna vytvořena firmou. Navazuje na ni proces získávání informací z obou webových serverů a uložení do pomocné databáze. Nakonec je popsáno obohacení uživatelských profilů o zmíněné informace a uložení do databází. Celá implementace je vytvořena v jazyce Python.

4.1 Mapování informací

Mapování informací navazuje na návrh z 3.2.1. Pro tyto účely je vytvořena základní třída `WebExtrasApi`, která slouží jako API pro mapování informací mezi různými formáty. Tato třída je uložena v souboru `web_extras_api.py` a je vložena do knihovny firmy, která nese název `ga-2502-constants`. Třída může dostat při vytvoření jeden argument `filename`, který obsahuje cestu k CSV souboru. Pokud není `filename` zadán, tak si třída načte defaultní seznam mapování ze souboru `web_extras_all.csv`. Pokud je `filename` zadán, tak se využije třída `FileWatcher` z knihovny `ga-2502-watcher` (vytvořena firmou), která hlídá soubor na změny a v případě změn se zavolá metoda `_reload_web_extras` pro aktualizaci mapování. Argument `filename` je využit především pro případy, že se jednotlivé mapování může měnit, rozšiřovat nebo zmenšovat oproti defaultnímu CSV souboru.

Každé mapování je uloženo v Python `dictionary` typu, který drží informace jako klíč-hodnota. Soubor ve formátu CSV se otevře a jednotlivé informace z něj se do těchto `dictionary` uloží. Pro práci nad těmito informacemi existují 2 typy metod, které se dělí na:

- **Metody pro seznamy názvů**
  - `get_all_name` - vrátí všechny názvy pro standardní formát
  - `get_all_slug` - vrátí všechny názvy pro slug formát
  - `get_all_abbr` - vrátí všechny názvy pro abbreviation formát
- **Metody pro mapování**
  - `name_to_abbr` - mapuje ze standardního formátu do abbreviation formátu
  - `name_to_slug` - mapuje ze standardního formátu do slug formátu
  - `abbr_to_name` - mapuje z abbreviation formátu do standardního formátu

- `abbr_to_slug` - mapuje z abbreviation formátu do slug formátu
- `slug_to_name` - mapuje ze slug formátu do standardního formátu
- `slug_to_abbr` - mapuje ze slug formátu do abbreviation formátu

Ukázku očekávaného CSV souboru je možné vidět na obrázku 4.1. První sloupec značí webový server, pod který tato informace spadá. Druhý sloupec obsahuje získaný úplný název (standardní formát), třetí sloupec obsahuje slug formát a čtvrtý abbreviation formát.

```
csfd.cz,CSFD: Akční,csfd_akcni,csfd:akc
csfd.cz,CSFD: Drama,csfd_drama,csfd:dra
csfd.cz,CSFD: Komedie,csfd_komedie,csfd:kom
recepty.cz,RECEPTY: Polévky,recepty_polevky,rcp:pole
recepty.cz,RECEPTY: Brambory,recepty_brambory,rcp:bram
recepty.cz,RECEPTY: Maso,rcp_maso,rcp:maso
```

Obrázek 4.1: Očekávaný CSV formát pro mapování informací.

V tabulce 4.1 jsou vypsány všechny nové informace z obou webových serverů, které se budou využívat a jsou obsaženy v defaultním CSV souboru pro mapování informací. Pro webový server `csfd.cz` jsou obsaženy všechny existující filmové žánry. Pro webový server `recepty.cz` bylo kategorií příliš mnoho a tudíž se využívají pouze hlavní sekce. Pro každou informaci je vypsán standardní, slug i abbreviation formát.

Tabulka 4.1: Přehled všech mapování informací.

Standardí formát	Slug	Abbreviation
CSFD: Akční	csfd_akcni	csfd:akc
CSFD: Animovaný	csfd_animovany	csfd:ani
CSFD: Dobrodružný	csfd_dobrodruzny	csfd:dob
CSFD: Dokumentární	csfd_dokumentarni	csfd:dok
CSFD: Drama	csfd_drama	csfd:dra
CSFD: Erotický	csfd_eroticky	csfd:ero
CSFD: Experimentální	csfd_experimentalni	csfd:exp
CSFD: Fantasy	csfd_fantasy	csfd:fan
CSFD: Film-Noir	csfd_film_noir	csfd:noi
CSFD: Historický	csfd_historicky	csfd:his
CSFD: Horor	csfd_horor	csfd:hor
CSFD: Hudební	csfd_hudebni	csfd:hud
CSFD: IMAX	csfd_imax	csfd:ima
CSFD: Katastrofický	csfd_katastroficky	csfd:kat
CSFD: Komedie	csfd_komedie	csfd:kom
CSFD: Krátkometrážní	csfd_kratkometrazni	csfd:kra
CSFD: Krimi	csfd_krimi	csfd:kri
CSFD: Loutkový	csfd_loutkovy	csfd:lou
CSFD: Muzikál	csfd_muzikal	csfd:muz

pokračování na další straně

Tabulka 4.1 – pokračování z předchozí strany

Standardí formát	Slug	Abbreviation
CSFD: Mysteriózní	csfd_mysteriozni	csfd:mys
CSFD: Podobenství	csfd_podobenstvi	csfd:pod
CSFD: Poetický	csfd_poeticky	csfd:poe
CSFD: Pohádka	csfd_pohadka	csfd:poh
CSFD: Povídkový	csfd_povidkovy	csfd:pov
CSFD: Psychologický	csfd_psychologicky	csfd:psy
CSFD: Publicistický	csfd_publicisticky	csfd:pub
CSFD: Reality-TV	csfd_reality_tv	csfd:rea
CSFD: Road movie	csfd_road_movie	csfd:roa
CSFD: Rodinný	csfd_rodinny	csfd:rod
CSFD: Romantický	csfd_romanticky	csfd:rom
CSFD: Sci-Fi	csfd_scifi	csfd:sci
CSFD: Soutěžní	csfd_soutezni	csfd:sou
CSFD: Sportovní	csfd_sportovni	csfd:spo
CSFD: Talk-show	csfd_talk_show	csfd:tal
CSFD: Taneční	csfd_tanecni	csfd:tan
CSFD: Telenovela	csfd_telenovela	csfd:tel
CSFD: Thriller	csfd_thriller	csfd:thr
CSFD: Válečný	csfd_valecny	csfd:val
CSFD: Western	csfd_western	csfd:wes
CSFD: Životopisný	csfd_zivotopisny	csfd:ziv
RECEPTY: Saláty	recepty_salaty	rcp:sala
RECEPTY: Předkrmy, chuťovky & svačiny	recepty_predkrmy	rcp:pred
RECEPTY: Dezerty	recepty_dezerty	rcp:deze
RECEPTY: Polévky	recepty_polevky	rcp:pole
RECEPTY: Omáčky & guláše	recepty_omacky	rcp:omac
RECEPTY: Brambory	recepty_brambory	rcp:bram
RECEPTY: Knedlíky	recepty_knedliky	rcp:kned
RECEPTY: Rýže	recepty_ryze	rcp:ryze
RECEPTY: Luštěniny	recepty_lusteniny	rcp:lust
RECEPTY: Noky	recepty_noky	rcp:noky
RECEPTY: Kuskus	recepty_kuskus	rcp:kusk
RECEPTY: Bulgur	recepty_bulgur	rcp:bulg
RECEPTY: Quinoa	recepty_quinoa	rcp:quin
RECEPTY: Housky a rohlíky	recepty_housky	rcp:hous
RECEPTY: Chléb & placky	recepty_chleb	rcp:chle
RECEPTY: Těstoviny & rizota	recepty_testoviny	rcp:test
RECEPTY: Maso	recepty_maso	rcp:maso
RECEPTY: Ryby	recepty_ryby	rcp:ryby
RECEPTY: Uzeniny & mleté maso	recepty_uzeniny	rcp:uzen
RECEPTY: Mořské plody (Dary moře)	recepty_more	rcp:more
RECEPTY: Nápoje	recepty_napoje	rcp:napo
RECEPTY: Zavařování & nakládání	recepty_zavarovani	rcp:zava

4.2 Streamovací knihovna

Pro všechny streamovací aplikace v této práci je využita streamovací knihovna vytvořena firmou, která nese název `ga-streaming-utils`. Knihovna je navržena pro bezproblémovou a jednoduchou práci s technologií Apache Kafka.

Tato knihovna obsahuje základní třídu `BaseKafkaMinibatchStream`, ze které všechny streamovací aplikace dědí. Inicializace této třídy očekává vstupní a výstupní (volitelné) témata společně se skupinou konzumentů popsané v 2.2.1. Na základě argumentů se vytvoří čtecí a zapisovací třídy, díky kterým je umožněna komunikace s Kafka serverem. Základní třída obsahuje 2 důležité metody, kterými se ovládá komunikace. Těmito metodami jsou `process_batch` pro zpracování příchozích zpráv a `send_to_kafka` pro poslání zprávy. Metoda na zpracování příchozích zpráv zůstává prázdná a každá nová aplikace si ji musí upravit. Celý proces se spustí metodou `start_streaming`.

4.3 Získání informací

Získání informací navazuje na návrh z 3.2.2. Pro tyto účely jsou vytvořeny 2 streamovací aplikace, které je možné vidět na obrázku 3.3. Těmito aplikacemi jsou `HTML extractor process` a `HTML extractor URL` popsané níže.

4.3.1 HTML extrakce

Streamovací aplikace pro HTML extrakci se dělí na HTML extraktor a samotný stream využívající jeho funkcionalit.

HTML extraktor

Pro účely extrakce informací z HTML je vytvořena třída `WebExtrasExtractor` uložená v souboru `web_extras_extractor.py`. Tato třída využívá `WebExtrasApi` pro mapování informací popsané v sekci 4.1. Dále využívá `TLDExtract` pro získání domény z URL adresy a `BeautifulSoup` pro samotnou extrakci informací z HTML kódu.

Vytvořená třída má jednu veřejnou metodu `extract`, která očekává dva argumenty `url` (URL adresa) a `html` (HTML kód). Zadaná URL adresa se pomocí `TLDExtract` zpracuje a získá se doména. Doména se společně s HTML kódem předá metodě `_get_web_extras`, která na základě domény určí další postup. Jejím výstupem jsou extrahované informace z webové stránky a nebo nic. Jednotlivé postupy jsou uloženy ve vnitřní proměnné typu `dictionary`, kde klíčem je vždy doména a hodnotou zvolená metoda pro vlastní extrakci. Na závěr se každá extrahovaná informace převede do slug formátu pomocí instance třídy `WebExtrasApi`. Informace se zahazuje v případě, že třída nezná mapování pro tuto informace. Znamená to tedy, že se extrahují pouze informace, které mají mapování v CSV souboru pro `WebExtrasApi`.

Metody pro jednotlivé domény využívají knihovnu `BeautifulSoup`, která slouží pro extrakci informací z webových stránek. Třídě `BeautifulSoup` se předá HTML kód, který je uchován pro následné vyhledávání. Tyto privátní metody jsou:

- **csfd.cz** (`_get_csfd_genres`) - Na základě analýzy z 3.1.1 je možné vidět, že filmové žánry jsou uloženy v HTML tagu `p` se třídou `genre` a jsou odděleny znakem `/`. Třída `BeautifulSoup` má metodu `find`, které se předá tag se třídou a výsledkem je první výskyt tohoto případu v HTML kódu. Pomocí funkcí `split` a `strip` se dále získaný text převede na pole jednotlivých žánrů a ořežou se bílé znaky okolo.

- **recepty.cz** (`_get_recepty_categories`) - Na základě analýzy z 3.1.2 je možné vidět, že kategorie receptů jsou uloženy v HTML tagu `div` s určitou třídou. Z analýzy je také vidět, že tento HTML tag a třída nejsou využity pouze pro kategorie receptů, ale také pro další vlastnosti. Je tedy nutné vyhledat správnou oblast k extrakci. Třída `BeautifulSoup` má metodu `find_all`, která funguje stejně jako `find`, pouze vrátí pole všech výskytů. Pro každý vrácený výskyt se zkontroluje, zdali je obsažen text *Kategorie*. Pokud ano, tak se extrahují jednotlivé kategorie receptů ze zanořených HTML tagů `a` a pomocí funkce `strip` se odstraní bílé znaky okolo.

Streamovací aplikace

Streamovací aplikace je postavena na streamovací knihovně popsané v 4.2 a je uložena v souboru `html_extractor_map.py`. Hlavní třídu tvoří `HtmlExtractor`, který dostane jako parametr objekt třídy `WebExtrasExtractor` popsaný výše. Třída může dostat parametry pro statistiky `stats_server` a zároveň `stats_prefix`, díky kterým může posílat statistiky na server.

Metoda `process_batch` je vytvořena tak, aby zároveň zpracovala příchozí zprávu a poslala ji dále. Metoda `parse_batch` zpracovává jednotlivé příchozí zprávy, které by měly obsahovat HTML kód a URL adresu. Pokud něco z toho neobsahuje, tak se daná zpráva ignoruje. Pro dvojici HTML kód a URL adresa se volá metoda `extract` (popsána výše) ze třídy `WebExtrasExtractor`. Pokud jsou extrahovány nové informace, tak se uloží do původní zprávy, ze které byl smazán HTML kód a nová zpráva se vrátí k poslání dále.

4.3.2 Uložení do URL databáze

Pro uložení extrahovaných informací do Aerospike URL databáze je využita knihovna `ga-2502-url-storage` implementovaná firmou. Tato knihovna obsahuje třídu pro manipulaci s databází `AerospikeUrlStorage`, která byla obohacena o metodu `put_web_extras`. Tato metoda uloží extrahované informace pod klíčem zadané URL adresy.

Streamovací aplikace je postavena na streamovací knihovně popsané v 4.2 a je uložena v souboru `html_extractor_url_storage.py`. Hlavní třídu tvoří `HtmlURLStorageWriter`, který očekává parametr pro URL databázi. Třída může dostat parametry pro statistiky `stats_server` a zároveň `stats_prefix`, díky kterým může posílat statistiky na server. Komunikaci s URL databází zajišťuje třída `AerospikeUrlStorage`. Tato streamovací aplikace je koncová a tudíž neposílá žádné zprávy dále, pouze čte a zpracovává příchozí.

Metoda `process_batch` je tedy vytvořena tak, aby zprávy pouze četla. Jednotlivé zprávy zpracovává metoda `insert_to_url_storage`, která uloží extrahované informace na základě obsažené URL adresy. Uložení probíhá zavoláním metody `put_web_extras` pro třídu `AerospikeUrlStorage`, kde argumenty jsou extrahované informace a URL adresa.

4.4 Obohacení profilů

Obohacení profilů navazuje na návrh v sekci 3.2.3. Popsané části jsou úprava databází, ve které je popsána potřebná procedura pro správné fungování databází, mapování a uložení zhlédnutých stránek, které se zabývá úpravou již existujících streamovacích aplikací a uložení extrahovaných stránek, které navazuje na předchozí sekci HTML extrakce a obsahuje nové 2 streamovací aplikace pro uložení do Elasticsearch a Aerospike databází.

4.4.1 Úprava databází

Úpravou databází se myslí úprava schématu Elasticsearch databáze a metadat pro Aerospike databázi. Úpravy pro jednotlivé databáze jsou:

- **Elasticsearch** - Pro Elasticsearch je schéma uloženo přímo na serveru. Schémata jsou uloženy pro šablonu indexu a existující index. V obou případech je potřeba přidat `{"WebExt": {"type": "keyword"}}` do mapování schématu a schéma aktualizovat na serveru. Aktualizace probíhá zavoláním metody `PUT` pro specifickou URL adresu s vloženým JSON schématem. Elasticsearch databáze ví, jak se chovat k danému atributu díky zadanému typu a klíčovému slovu ve schématu.

- **Aerospike** - Pro Aerospike jsou metadata uložena v souboru `ups-2502.yml` na firemním serveru. Aerospike klient si tato metadata načítá a pro každé klíčové slovo se podle nich zachová a přidá je k datům. Klíčové slovo je `WebExt`, které má v metadatech nastavené 3 parametry. Parametr pro unikátnost `unique: True`, parametr pro počet dní držení uložené hodnoty `keep_days: 30` a parametr pro počet uložených hodnot `keep_count: 20`.

4.4.2 Uložení extrahovaných stránek

Pro uložení extrahovaných stránek do databází jsou vytvořeny 2 streamovací aplikace. Jedná se o aplikace `HTML extractor Aerospike` a `HTML extractor Elasticsearch` z návrhu 3.3 popsané níže.

Uložení do Aerospike databáze

Pro uložení extrahovaných informací do Aerospike databáze uživatelských profilů je využita knihovna `ga-user-profile-storage` implementovaná firmou. Knihovna obsahuje třídu `AerospikeUserProfile` s metodou `save_many`, která umí uložit pole záznamů do databáze. Jednotlivé záznamy musí být třídy `Event` z této knihovny, která obsahuje ID uživatele, název uložené hodnoty, časové razítko a hodnotu pro uložení.

Streamovací aplikace pro uložení do Aerospike databáze je postavena na streamovací knihovně popsané v 4.2 a je uložena v souboru `html_extractor_ups.py`. Hlavní třídu tvoří `HtmlUPSWriter`, který očekává parametr pro Aerospike databázi profilů a třídu pro mapování `WebExtrasApi`. Třída může dostat parametry pro statistiky `stats_server` a zároveň `stats_prefix`, díky kterým může posílat statistiky na server. Komunikaci s databází profilů zajišťuje třída `AerospikeUserProfile`. Tato streamovací aplikace je koncová a tudíž neposílá žádné zprávy dále, pouze čte a zpracovává příchozí.

Metoda `process_batch` je tedy vytvořena tak, aby zprávy pouze četla. Jednotlivé zprávy zpracovává metoda `map_to_events`, která je předělává do třídy `Event`. Každá zpráva

obsahuje ID uživatele, časové razítko v podobě textu a extrahované informace. Název uložené hodnoty je zadán v globální proměnné `WEB_EXTRAS_EVENT_NAME`. Časové razítko je převedeno do potřebného formátu a každá extrahovaná informace přemapována do abbreviation formátu. Vše je vloženo do třídy `Event`, která je vrácena a následně uložena pomocí metody `save_many` z `AerospikeUserProfile`.

Uložení do Elasticsearch databáze

Pro uložení extrahovaných informací do Elasticsearch databáze uživatelských profilů je využita knihovna `Elasticsearch`. Tato knihovna obsahuje třídu `Elasticsearch` s metodou `update_by_query`, která umí uložit informace na základě strukturovaného dotazu.

Streamovací aplikace pro uložení do Elasticsearch databáze je postavena na streamovací knihovně popsané v 4.2 a je uložena v souboru `html_extractor_es.py`. Hlavní třídu tvoří `HtmlESWriter`, který očekává parametr pro Elasticsearch databázi. Třída může dostat parametry pro statistiky `stats_server` a zároveň `stats_prefix`, díky kterým může posílat statistiky na server. Komunikaci s databází zajišťuje třída `Elasticsearch`. Tato streamovací aplikace je koncová a tudíž neposílá žádné zprávy dále, pouze čte a zpracovává příchozí.

Metoda `process_batch` je tedy vytvořena tak, aby zprávy pouze četla. Jednotlivé zprávy zpracovává metoda `_update_url_keywords`, která pro ně vytváří dotaz do Elasticsearch databáze. Uložení probíhá zavoláním metody `update_by_query` z `Elasticsearch`.

4.4.3 Mapování a uložení zhlédnutých stránek

V rámci Mapování a uložení zhlédnutých stránek je nutné upravit stávající 3 aplikace vytvořené firmou. Tyto aplikace pracují se zhlédnutými stránkami uživatelů. Těmito aplikacemi jsou samotné mapování informací a uložení do Aerospike a Elasticsearch databází uživatelských profilů.

- **Mapování** - Streamovací aplikace na mapování webových stránek využívá Aerospike URL databází zmíněnou v 4.3.2. Pokud se data nenachází v URL databází, tak je zhlédnutá stránka poslána na stažení HTML kódu, který je následně zpracován pomocí aplikací popsaných v 4.3. Pro detekci, zdali se extrahované informace nachází v URL databázi, je použita metoda `_try_set_pageview_web_extras`. Tato metoda se pokusí z databáze vytáhnout nové informace a vrátit je, pokud existují. Vrácené informace se přidají k informacím o zhlédnuté webové stránce.

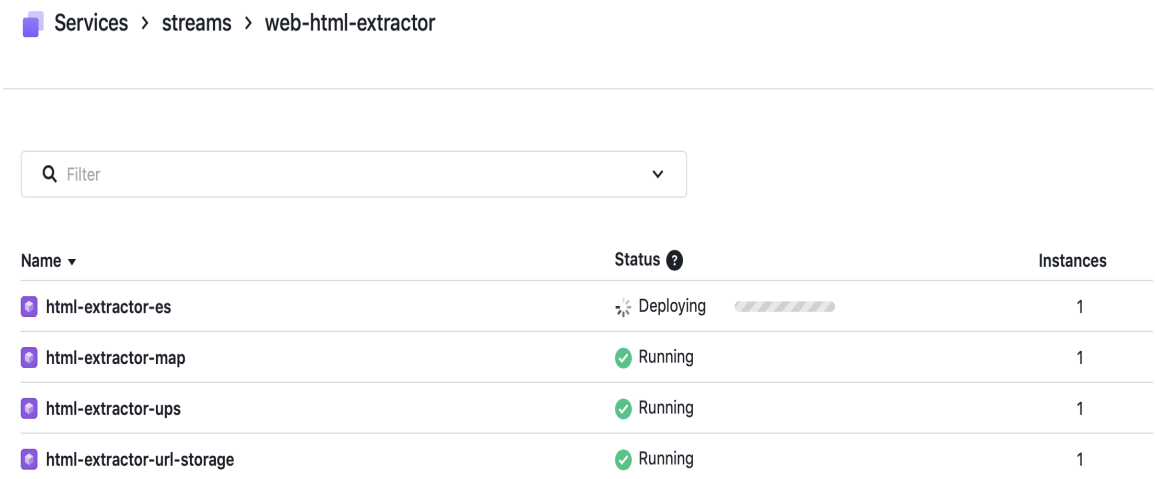
- **Uložení do Aerospike databáze** - Streamovací aplikace pro uložení do Aerospike databáze využívá knihovnu pro práci s touto databází popsanou v 4.4.2. Do ukládací metody této aplikace je přidán proces na převedení nových informací do abbreviation formátu a následně jejich modifikace na třídu `Event`. Třídu event umí zmíněná knihovna ukládat do databáze a stará se o to zbytek aplikace.

- **Uložení do Elasticsearch databáze** - Streamovací aplikace pro uložení do Elasticsearch databáze využívá knihovnu popsanou v 4.4.2. V tomto případě je přítomna vyšší míra abstrakce a stačí přidat nový název atributu `WebExt` společně s datových typem do třídy obsahující tyto informace. O zpracování a uložení se postará zbytek aplikace.

4.5 Nasazení do produkce

Všechny 4 nové streamovací aplikace popsané v této kapitole jsou uloženy v Python balíčku `ga-2502-web-html-extractor`, který je vytvořen pomocí šablony na tvorbu balíčku od firmy. Vše je uloženo v adresáři `ga/p2502_web_html_extractor` tohoto balíčku. Pro streamovací aplikace jsou napsány i testy pro otestování funkčnosti. Streamovací aplikace jsou uloženy v podsložce `streams`, HTML extraktor v podsložce `extractors` a testy v podsložce `test`. Balíček obsahuje příkazy v souboru `commands.py`, pomocí kterých se dají spustit jednotlivé aplikace.

V produkci využívá firma službu `DC/OS`, která slouží pro nasazení a správu jednotlivých aplikací. Na obrázku 4.2 je možné vidět všechny 4 aplikace nového balíčku nasazeny v tomto prostředí.



Obrázek 4.2: Ukázka DC/OS s nasazenými streamovacími aplikacemi.

Kapitola 5

Analýza vlivu rozšíření profilů

V této kapitole je popsáno vše potřebné k analýze a vyhodnocení přínosu obohacených informací v uživatelských profilech. Jsou zde popsána vstupní data, na kterých se analýza provádí. Na těchto vstupních datech je potřeba provést předzpracování a převedení do formátu, se kterým se dá dále lépe pracovat. Dalším problémem pro analýzu je nepoměr počtu záznamů pro jednotlivé třídy v datech. Na závěr jsou popsány jednotlivé experimenty a jejich provedení, včetně vyhodnocení. Celá kapitola využívá informace popsané v sekci o strojovém učení 2.3.

5.1 Vstupní data

Vstupní data byla získána z databází od firmy a jsou jimi uživatelské profily popsané v sekci 2.2. Byly vybrány pouze profily, kterým byly zobrazeny produkty z reklamní kampaně na výživové doplňky nejmenované společnosti. Veškeré vybrané profily obsahují alespoň jednu novou informaci z webových stránek, které byly předmětem zadání této diplomové práce. Tyto nové informace jsou uloženy pod klíčovým slovem `WebExt`. Důležitým prvkem těchto profilů je atribut `clicks`, ve kterém je zaznamenána informace, jestli uživatel kliknul na zobrazenou reklamu nebo ne.

Data byla vytažena z databáze Elasticsearch, která je přímo přizpůsobena k vyhledávání dat ve velkém množství záznamů na základě specifického dotazu. Pro data bylo důležité, aby vždy obsahovala nově extrahované informace z webových stránek a zobrazení nebo kliknutí na reklamu. Dotaz do Elasticsearch databáze: `(_exists_: clicks.XXXXXXX OR _exists_: imps.XXXXXXX) AND _exists_: WebExt`, kde `XXXXXXX` je ID reklamní kampaně.

Jednotlivé profily jsou uloženy ve formátu JSON v souboru `user_profiles.json`. Typ informací se liší pro různá klíčová slova, mohou to být textová pole, pole dalších informací a nebo vnořené JSON objekty. Na obrázku 5.1 je malá ukázka dat pro představu pro další zpracování. Celkový počet profilů ve vstupních datech je **71249**.

```
[
   {
      "sex": "M",
      "WebExt": [
         "csfd_drama",
         "csfd_komedie",
         "recepty_predkrmy"
      ],
      "clicks": {
         "1000138": 1,
         "2000386": 1,
         "total": 1
      },
      ...
   },
   ...
]
```

Obrázek 5.1: Ukázka části vstupních dat.

5.2 Předzpracování

Vstupní data nejsou vhodná pro analýzy ani pro modely strojového učení, z tohoto důvodu musí být upravena. Sekce předzpracování se zabývá touto úpravou dat do formátu, který je vhodnější pro analýzu. V rámci této úpravy se provádí převedení dat do vektoru `0` a `1` a následně redukce dimenzionality vlastností pomocí výběru atributů a korelační analýzy. Celé předzpracování se nachází v souboru Python Jupyter notebook `preprocessing.ipynb` a jeho výstup je uložen v souboru `final_dataframe.csv`. Sekce čerpá z 2.3.2.

5.2.1 Transformace na vektor

Nejdůležitější částí předzpracování v této diplomové práci je transformace vstupních dat uživatelských profilů z JSON formátu na vektor `0` a `1`, který obsahuje veškeré možnosti všech atributů, které se v profilech nachází.

V předchozí sekci bylo řečeno, že jednotlivé atributy profilů mohou být ve 3 formátech: textové pole, pole dalších informací a vnořené JSON objekty. Z tohoto důvodu je potřeba si z dat vytáhnout informaci, které atributy mají který formát. Už v této části se zasáhne do další sekce na výběr atributů 5.2.2, protože do jednotlivých kategorií atributů se zahrnou pouze ty atributy, které budou dále použity. Znamená to tedy, že již v této fázi se zahodí atributy uživatelských profilů, které nejsou důležité. Kategorie vybraných atributů ze sekce 2.2.2 jsou:

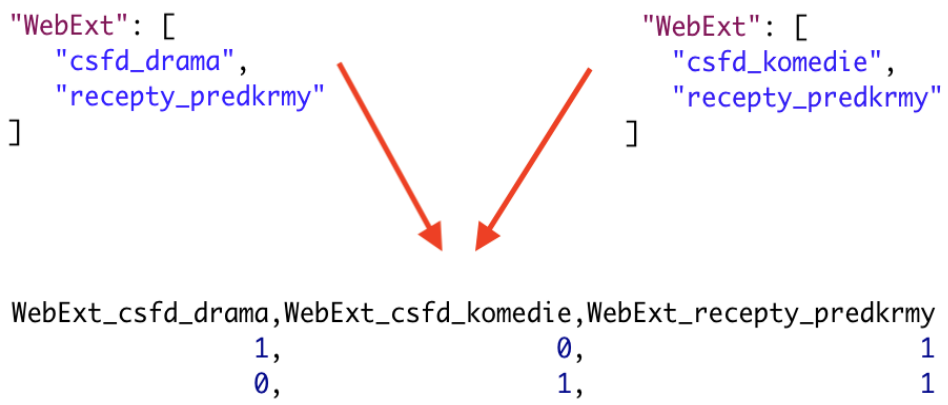
- **Textové pole** - `sex`, `is_traveler`, `is_international`, `is_abroad`
- **Pole dalších informací** - `typ`, `dev`, `OS`, `br`, `blng`, `cr`, `lId`, `lRe`, `lCat`, `dom`, `sub`, `TopD`, `WebC`, `KW`, `brnd`, `WebCtg`, `WebKw`, `WebNe`, `WebPh`, `WebExt`, `vPC`, `vTag`, `vSId`, `oPC`, `oTag`, `oSId`, `cPC`, `cTag`, `cSId`
- **Vnořené JSON objekty** - `clicks`

Samotná transformace na vektor probíhá ve dvou fázích:

- **První fáze** - Iterace přes všechny profily a transformace dat každého atributu na nový příslušný atribut s hodnotou `1`. Pro hodnoty atributu s polem dalších informací se provádí transformace každého prvku pole. Příkladem je transformace `"WebExt": ["csfd_drama"]` na `"WebExt_csfd_drama": 1`. Pro textové pole se provádí transformace jako pro pole o 1 prvku. Vnořený JSON objekt obsahuje pouze atribut `clicks`, ze kterého se vytváří nový atribut `label`. Nový atribut `"label": 1` se vytvoří v případě, že původní atribut `clicks` obsahuje počet kliknutí pro danou kampaň větší než `1`. Uživatelské profily obsahují jeden atribut `is_bot`, který označuje daný profil jako robota. V této části transformace se ignorují záznamy, které mají tento atribut jako `True`. Zahazování robotických profilů je prospěšné pro následnou analýzu, aby neovlivňovaly správnost výsledků.

- **Druhá fáze** - Transformovaná data z první fáze nejsou vektory, ale JSON objekty s atributy ohodnocenými hodnotou `1`. Tato data je tedy nutné převést na vektory. Pro každý profil z transformovaných dat se uloží všechny jeho existující atributy do pole. Pro každý atribut v poli všech existujících atributů se pro každý profil z transformovaných dat poznamená `1`, pokud profil atribut s hodnotou `1` obsahuje, jinak `0`. Vznikne pole vektorů, kde každý záznam představuje jeden profil a každý prvek vektoru hodnotu jednoho existujícího atributu.

Transformovaná data obsahují **589818** atributů a jsou uložena jako vektor. Názorná ukázka převedení dat na vektor je zobrazena na obrázku 5.2.



Obrázek 5.2: Ukázka transformace dat na vektor.

5.2.2 Výběr atributů

Vzhledem k velkému množství nově vzniklých atributů z transformace dat v předchozí sekci 5.2.1 je potřeba tento počet snížit pro možnost výpočtů v reálném čase bez ztráty kvality. K tomuto účelu slouží redukce dimenzionality. Výběr atributů je jedna z metod využívaných při redukci dimenzionality. Dále se využívá konstrukce a extrakce atributů, v tomto případě by data ztratila interpretovatelnost pro následné analýzy a tudíž je použit pouze výběr atributů. Výběr atributů je aplikován na základě dvou různých přístupů, kterými jsou počet a korelace.

Výběr na základě počtu

Výběr atributů na základě počtu je jednodušší z použitých přístupů. V tomto případě se jedná o snížení počtu atributů razantním způsobem za pomocí součtu výskytů jednotlivých atributů napříč profily (součtu `1` mezi vektory). Pro každý existující atribut se spočítá součet jeho výskytů a poznamenají se ty atributy, které se nachází v profilech označených kliknutím na reklamu. Vzhledem ke znalosti, že data obsahují nízký počet takto označených profilů, je potřeba se zachovat ke každé třídě (kliknutí a nekliknutí na reklamu) rozdílně, aby nebylo ztraceno příliš informací z méně početné skupiny.

Pro obě skupiny profilů jsou určeny spodní a vrchní hranice počtu výskytů. Méně početná skupina profilů, které klikly na reklamu, mají spodní hranici určenou alespoň na `300` výskytů atributu a horní hranici na `65000`. Početnější skupina profilů, které na reklamu neklikly, mají spodní hranici stanovenou na `1000` výskytů atributu a horní hranici na `60000`. Důležitější je spodní hranice, protože hranice horní nemá téměř žádný vliv. Spodní hranice zajišťuje, že se neobjevují v redukovaných datech atributy, které by obsahovalo příliš malé množství profilů.

Nad seznamem atributů je provedena funkce `filter`, která na základě určených hranic a ignorováním počtu výskytů pro atributy `WebExt` a `label` odstraní nepotřebné atributy. Výsledkem je zredukování počtu atributů z **589818** na **12151**, které jsou dále vhodnější pro korelační analýzu.

Výběr na základě korelace

Díky redukci na základě počtu atributů je možnost provést korelační analýzu v reálném čase. Pro výpočet korelací mezi atributy je použita metoda `corr` pro `DataFrame` z Pandas knihovny. Tato metoda vypočítá matici korelací formou každý s každým, ve které jsou sloupce i řádky jednotlivé atributy. Hodnoty matice korelace jsou čísla mezi `-1` a `1` podle míry korelace. Hodnoty blížící se `-1` a `1` mají silnou kladnou nebo zápornou korelaci, zatímco hodnoty blízko `0` nekorelují skoro vůbec.

Procházet, vybírat a redukovat korelující data v korelační matici ručně není téměř možné. Tento proces je zautomatizován, ovšem za cenu ztráty možnosti rozhodnutí, který z korelujících atributů je ten důležitější pro ponechání.

Proces automatizace se provádí ve dvou krocích:

- **První krok** - První krok slouží k tomu, aby nedošlo ke smazání všech korelujících atributů, ale zůstal zachován alespoň jeden z nich. Vytvoření pomocné matice s hodnotami `True` a `False` označující pozice z korelační matice, které je potřeba zpracovat. Tato pomocná matice obsahuje `True` v pravé hodní části matice bez hlavní diagonály a `False` v levé dolní včetně hlavní diagonály. Tato matice zaručí, že atributy ve sloupci více vlevo zůstanou v případě korelace zachovány, zatímco všechny ostatní se odstraní. Vytvoření této matice je vynuceno použitím funkcí z `Pandas` knihovny.
- **Druhý krok** - Ve druhém kroku se vyberou hodnoty z korelační matice na základě nově vytvořené pomocné matice. Vznikne nová korelační matice obsahující data pouze v pravé horní části od hlavní diagonály, což označují `True` v pomocné matici. Nad touto novou korelační maticí se provede pro každý atribut ve sloupci podmínka, která ověřuje, zdali je alespoň jedna hodnota ve sloupci menší než `-0.8` nebo větší než `0.8`. Z aplikování podmínky na každý sloupec vznikne pole atributů, které silně korelují a je potřeba je odstranit.

Po provedení celého procesu se z **12151** atributů odstraní **2198** korelujících a zůstane **9953** atributů pro následnou analýzu. Znázornění jednotlivých matic je možné vidět na obrázku 5.3.

	KW_eroticky	KW_erotika	KW_auto	dom_erotika	sex_M
KW_eroticky	1.0	0.9	0.1	0.8	0.5
KW_erotika	0.9	1.0	0.1	0.9	0.6
KW_auto	0.1	0.1	1.0	0.1	0.7
dom_erotika	0.8	0.9	0.1	1.0	0.5
sex_M	0.5	0.6	0.7	0.5	1.0

	KW_eroticky	KW_erotika	KW_auto	dom_erotika	sex_M
KW_eroticky	False	True	True	True	True
KW_erotika	False	False	True	True	True
KW_auto	False	False	False	True	True
dom_erotika	False	False	False	False	True
sex_M	False	False	False	False	False

	KW_eroticky	KW_erotika	KW_auto	dom_erotika	sex_M
KW_eroticky	NaN	0.9	0.1	0.8	0.5
KW_erotika	NaN	NaN	0.1	0.9	0.6
KW_auto	NaN	NaN	NaN	0.1	0.7
dom_erotika	NaN	NaN	NaN	NaN	0.5
sex_M	NaN	NaN	NaN	NaN	NaN

Obrázek 5.3: Znázornění matic pro korelaci.

5.3 Příprava dat

Předzpracovaná data je potřeba připravit pro modely strojového učení, aby se na nich byly schopny kvalitně učit. Nejprve je potřeba data rozdělit do trénovacích a testovacích skupin. Trénovací skupinu je dále potřeba upravit kvůli nepoměru dat jednotlivých tříd.

5.3.1 Rozdělení dat

Pro efektivní analýzu je potřeba data rozdělit do několika skupin, se kterými se pracuje rozdílně. Většinou se data rozdělují do dvou skupin, kterými jsou **trénovací** a **testovací**. V tomto případě se data dělí do 3 skupin, protože model využitý dále je možné ladit pomocí validačních dat. Jednotlivé skupiny jsou:

- **Trénovací** - Trénovací sada je využita při učení modelu, který se postupně na každém záznamu snaží co nejlépe naučit.
- **Validační** - Validační sada je také využita při učení modelu, kde po každé iteraci učení se model otestuje na těchto datech. Na základě výsledků validace se určí a vybere přínos dané iterace.
- **Testovací** - Testovací sada je využita až po naučení modelu. Na tuto sadu se aplikuje naučený model a výsledky se porovnají s těmi očekávanými.

Nejdříve je potřeba data rozdělit na data a ohodnocení, čehož lze docílit přesunutím sloupce `label` do vlastní proměnné. Následně je možné využít funkci `train_test_split` z knihovny Scikit-learn, která rozdělí data na základě zadané hodnoty v procentech pro testovací sadu. Hodnota procent byla zvolena na `15%`, což znamená, že trénovací sada má kolem `70%` záznamů a testovací sada s validační mají každá kolem `15%` záznamů. Počty záznamů v rozdělených datech pro efektivní analýzu je možné vidět v tabulce 5.1.

Část dat	Počet profilů s kliknutím	Počet profilů bez kliknutí
Trénovací	339	50689
Validační	59	8947
Testovací	72	10523

Tabulka 5.1: Rozdělení dat pro efektivní analýzu.

5.3.2 Nepoměr dat

Nepoměr tříd v datech může způsobit špatné naučení modelu strojového učení. Je tedy nutné data upravit, aby se toto nestávalo. K tomuto účelu slouží 3 přístupy, kterými jsou redukce početnější třídy, rozšíření méně početné třídy a využití vah pro učení. Počty záznamů jednotlivých vytvořených datových sad pro vypořádání se s nepoměrem dat je možné vidět v tabulce 5.2 a dělí se na:

- **Původní data** - Původní datová sada se bere z předchozího předzpracování. Nebyly s ní prováděny žádné další změny a nachází se mezi datovými sadami, aby byl vidět pokrok u upravených datových sad.

- **Vážená data** - Vážená datová sada se nijak neliší od původní datové sady, pouze obsahuje navíc váhy jednotlivých tříd. Tyto váhy tříd se vytvoří na základě původních dat. V datech je obrovský nepoměr mezi třídami a tudíž vyvážené váhy by daly příliš velký důraz na méně početnou třídu. Z tohoto důvodu je zvolen poměr vah `1:5` pro více početnou třídu ku méně početné třídě. Znamená to, že pro každý profil z méně početné třídy se dá `5x` větší důraz na jeho atributy. Pro výpočet tohoto poměru nad daty je možné využít modul `class_weight` z knihovny Scikit-learn.

- **Redukovaná data** - Redukovaná datová sada využívá metodu `RandomUnderSampler` z knihovny imbalanced-learn, která je určena pro náhodné mazání vzorků při zachování určitého poměru. V případě nezadání žádného poměru se vezme počet méně početné třídy a více početná třída se náhodným způsobem zmenší na tento počet. Výsledkem jsou stejně početné třídy, ovšem za cenu ztráty obrovského množství dat.

- **SMOTE data** - SMOTE datová sada využívá metodu `SMOTE` (Synthetic Minority Over-sampling Technique) z knihovny imbalanced-learn, jak již napovídá název. Tato metoda je určena pro rozšíření záznamů z méně početné třídy pomocí tvoření úplně nových záznamů. Pro tvorbu nových záznamů se vezme vždy vzorek dat, pro který se najde `k` nejbližších sousedů. Vektor mezi aktuálním záznamem a jedním z nejbližších sousedů se vynásobí náhodným číslem mezi `0` a `1`. Výsledný vektor se přičte k aktuálnímu záznamu a vznikne nový záznam. Tento proces se opakuje, dokud není naplněn očekávaný počet dat. V případě této datové sady se poměr mezi třídami bere jako `balanced`, což značí vyrovnaný počet dat. Méně početná třída se rozšíří na počet více početné třídy. [16]

- **SMOTE redukovaná data** - SMOTE redukovaná datová sada kombinuje předchozí 2 přístupy, kterými jsou redukce pomocí `RandomUnderSampler` a rozšíření pomocí `SMOTE`. Pro každou třídu se použije jedna ze zmíněných metod. Nejdříve je více početná třída zredukována na menší počet záznamů, poté se méně početná třída rozšíří. Redukce a rozšíření jsou v této datové sadě zvoleny tak, aby nebyly obě třídy vyrovnané, ale aby zůstaly v nepoměru `3:7`.

Typ dat	Počet profilů s kliknutím	Počet profilů bez kliknutí
Původní	339	50689
Redukovaná	339	339
Vážená	339	50689
SMOTE	50689	50689
SMOTE redukovaná	3000	7000

Tabulka 5.2: Datové sady po vyřešení nepoměru dat.

5.4 Model

Zhodnocení přínosu nově obohacených informací je na základě predikce kliknutí. Model pro predikci kliknutí doporučený firmou je `CatBoostClassifier` z knihovny Catboost, který slouží jako gradientní boosting pro rozhodovací stromy. Nejdříve je nutné model vytvořit a naučit na datech, kde se mohou využívat pro efektivnější učení navíc validační data. Poté je provedena predikce na testovacích datech, díky které se dají přehledně zobrazit metriky predikce a výsledky ohodnocení. Zajímavým prvkem modelu je možnost vypsání použitých atributů s váhou důležitosti.

Proces využití modelu se dělí na:

- **Vytvoření** - Vytvoření modelu je poměrně snadné, důležité je zadání všech potřebných parametrů, které se mají změnit oproti defaulním. `CatBoostClassifier` je připraven fungovat na vysoké úrovni s defaultními parametry, proto jsou upraveny jenom některé parametry. Nejdůležitějším parametrem je evaluační metrika, která se využívá pro zhodnocení přínosu po každé iteraci. Evaluační metrikou je zvolena `F1`. Další parametry jsou zvoleny spíše pro efektivní využití procesoru, paměti a času.

- **Naučení** - Naučení modelu zajišťuje metoda `fit`, která očekává vstupní data rozdělena na data a ohodnocení. Tato metoda může navíc dostat parametr `cat_features`, který označuje kategorické atributy. Pokud jsou vložena i validační data, tak se provede po každé iteraci zhodnocení přínosu na základě zvolené evaluační metriky. Výstupem je naučený model.

- **Predikce** - Vytvořený a naučený model může predikovat výsledky pro vstupní data, k čemu je určena metoda `predict`. Metoda očekává vstupní data ve stejném formátu jako ta trénovací. Výsledkem je seznam predikovaného ohodnocení pro jednotlivá vstupní data.

- **Důležitost atributů** - Model umí přehledně vypsat důležitost jednotlivých atributů díky využitým rozhodovacím stromům. Slouží k tomu `get_feature_importance` metoda, které se dodají data a výsledkem je seznam důležitosti atributů.

5.5 Experimenty

Experimenty v této sekci mají za úkol zhodnotit přínos nově obohacených informací v uživatelských profilech. Model ze sekce 5.4 je naučen, zvalidován a otestován na datových sadách získaných z předchozích zpracovaní. Výsledky pro zhodnocení přínosu jsou získány pomocí funkce `classification_report` z knihovny Scikit-learn, které obsahují informace o metrikách `precision`, `recall` a `F1`. Následující 3 experimenty se snaží o naučení modelu se všemi atributy, s atributy bez nových informací a s atributy obsahující pouze nové informace.

5.5.1 Všechny atributy

Experiment se všemi atributy se nachází v souboru Jupyter Python notebook pojmenovaným `experiment_all.ipynb`. Datová sada pro naučení obsahuje **9953** atributů.

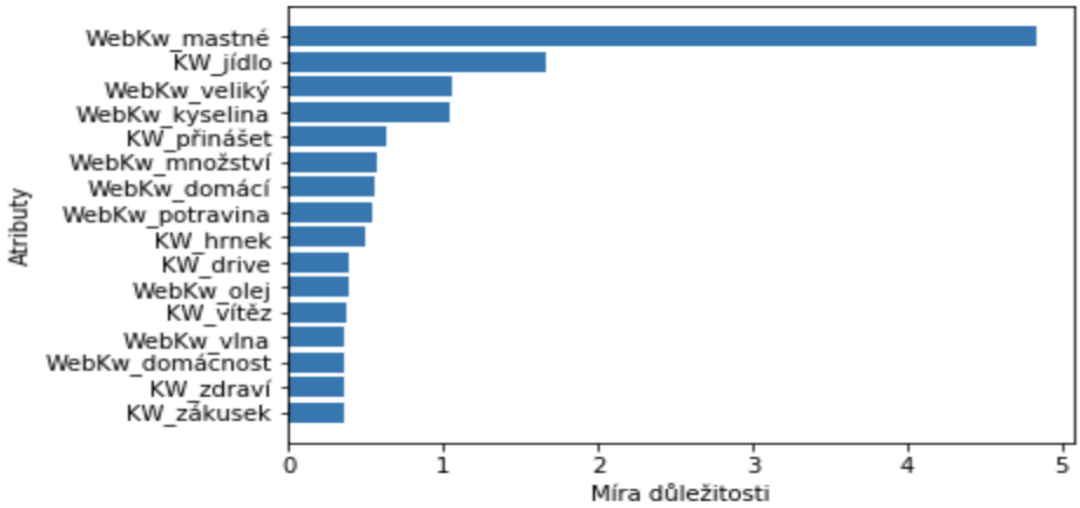
Typ dat	Precision	Recall	F1	Označeno
Původní	0.00	0.00	0.00	0
Redukovaná	0.02	0.49	0.04	1571
Vážená	0.62	0.07	0.12	8
SMOTE	0.00	0.00	0.00	29
SMOTE redukovaná	0.11	0.15	0.13	98

Tabulka 5.3: Vyhodnocení naučených modelů pro všechny atributy.

Zhodnocení jednotlivých datových sad a výsledků z tabulky 5.3:

- **Původní data** - Pro původní datovou sadu byly hodnoty všech metrik `0.00` a označené množství jako profil s kliknutím `0`. To znamená, že se model na datové sadě bez úpravy nepoměru dat nic nenaučí.
- **Redukovaná data** - Redukovaná datová sada ohodnotila příliš mnoho profilů pozitivně (téměř `22x více`). `Precision` je velice nízký, což způsobuje mnoho profilů označených špatně. Přívětivý je `recall`, který říká, že byla označena téměř polovina profilů správně. Důležitý je `F1`, který je kvůli `precision` také velice nízký a tudíž lze označit redukovanou datovou sadu za špatnou.
- **Vážená data** - Vážená datová sada vykazuje nejlepších výsledků. Ohodnotí pozitivně pouze malý počet dat, ale za to s vysokou přesností, kde `5` z `8` je správně. Z důvodů ohodnocení malého počtu je `recall` poměrně nízký, vyrovnává to však zmíněný `precision`.
- **SMOTE data** - Pro SMOTE datovou sadu byly hodnoty všech metrik `0.00` a označené množství jako profil s kliknutím malé, ale bezvýznamné. To znamená, že se model na datové sadě s úpravou dat pomocí algoritmu `SMOTE` naučí úplně špatně.
- **SMOTE redukovaná data** - SMOTE redukovaná datová sada vykazuje rovnoměrných výsledků, ale ohodnocuje pozitivně stále velké množství profilů. Kvůli tomu jsou metriky poměrně nízké, neznamená to však, že jsou špatné. `F1` je sice trochu lepší než pro váženou datovou sadu, ale `precision` zase o dost horší. Vše je způsobeno vyšším počtem pozitivně ohodnocených profilů.

Vybrané atributy a jejich důležitost z výsledků pro váženou datovou sadu je možné vidět na obrázku 5.4. Všechny tyto atributy mají něco podobného s tématem zvolené reklamní kampaně. Největší význam má slovo `mastné`, které je ve spojení s `kyselina` důležité pro výživové doplňky. Dalšími zajímavými slovy jsou `jídlo` nebo `potravina`. Mezi hlavními atributy nejsou využity žádné nově obohacené informace.



Obrázek 5.4: Graf míry důležitosti atributů pro váženou datovou sadu.

5.5.2 Atributy bez nových informací

Experiment s atributy bez nových informací se nachází v souboru Jupyter Python notebook pojmenovaným `experiment_without.ipynb`. Experiment potřebuje každou datovou sadu upravenou, aby neobsahovala nově obohacené informace. Seznam nově obohacených informací se dá získat kontrolou každého atributu, zdali nezačíná textem `WebExt`. Všechny tyto získané atributy se z datových sad odstraní. Datová sada pro naučení obsahuje tedy **9890** atributů.

Typ dat	Precision	Recall	F1	Označeno
Původní	0.00	0.00	0.00	0
Redukovaná	0.04	0.38	0.08	622
Vážená	0.54	0.10	0.16	13
SMOTE	0.00	0.00	0.00	25
SMOTE redukovaná	0.08	0.12	0.09	118

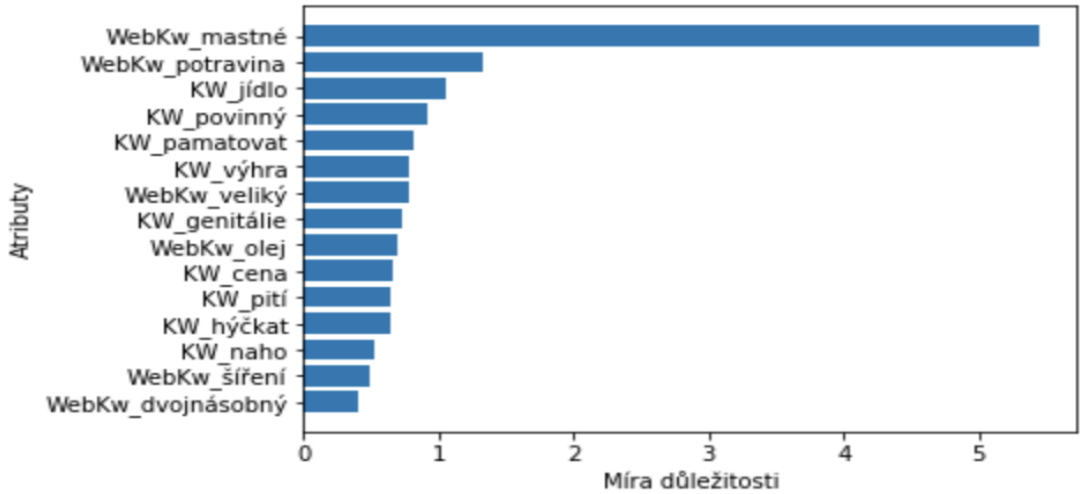
Tabulka 5.4: Vyhodnocení naučených modelů pro atributy bez nových informací.

Zhodnocení jednotlivých datových sad a výsledků z tabulky 5.4:

- **Původní data** - Stejné jako v experimentu 5.5.1.
- **Redukovaná data** - Obdobné jako v experimentu 5.5.1, ale méně označených dat pozitivně a tudíž lepší `precision` i `F1`.
- **Vážená data** - Obdobné jako v experimentu 5.5.1, ale mírně více označených dat pozitivně. Následkem toho je lepší `recall` a `F1`, ale horší `precision`.

- **SMOTE data** - Stejné jako v experimentu 5.5.1, pouze se mírně liší počet označených dat pozitivně.
- **SMOTE redukovaná data** - Obdobné jako v experimentu 5.5.1, ale mírně více označených dat pozitivně. Následkem toho jsou horší všechny metriky.

Vybrané atributy a jejich důležitost z výsledků pro váženou datovou sadu je možné vidět na obrázku 5.5. Všechny atributy jsou velice podobné s experimentem 5.5.1, ve kterém se nově obohacené informacím nedala vysoká míra důležitosti.



Obrázek 5.5: Graf míry důležitosti atributů pro váženou datovou sadu.

5.5.3 Pouze nové informace

Experiment s atributy pouze nových informací se nachází v souboru Jupyter Python notebook pojmenovaným `experiment_webext.ipynb`. Experiment potřebuje každou datovou sadu upravenou, aby obsahovala pouze nově obohacené informace. Seznam nově obohacených informací se dá získat kontrolou každého atributu, zdali nezačíná textem `WebExt`. Všechny ostatní atributy kromě těch získaných se z datových sad odstraní. Datová sada pro naučení obsahuje tedy **63** atributů.

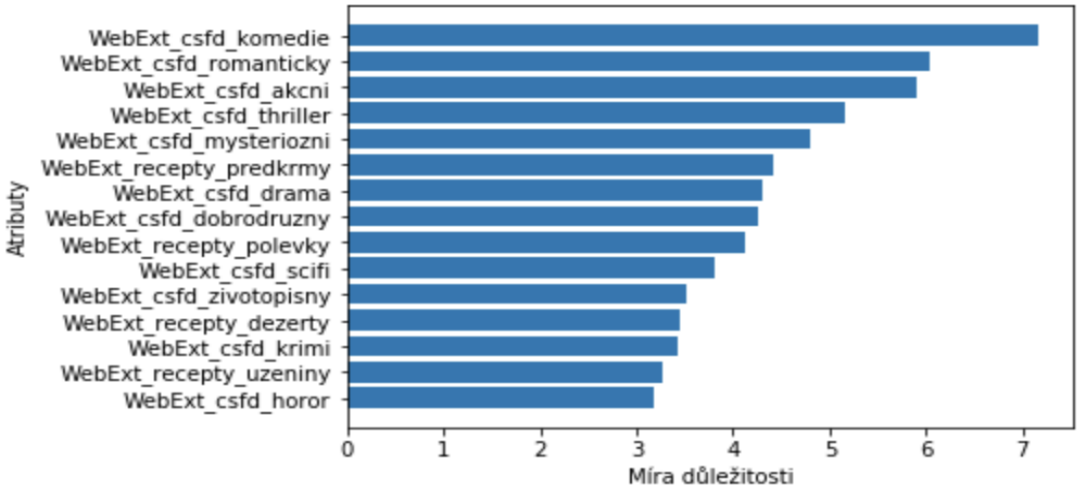
Typ dat	Precision	Recall	F1	Označeno
Původní	0.00	0.00	0.00	0
Redukovaná	0.01	0.51	0.01	4887
Vážená	0.08	0.01	0.02	12
SMOTE	0.01	0.49	0.01	4824
SMOTE redukovaná	0.01	0.08	0.02	611

Tabulka 5.5: Vyhodnocení naučených modelů pro pouze nové informace.

Zhodnocení jednotlivých datových sad a výsledků z tabulky 5.5:

- **Původní data** - Stejné jako v experimentu 5.5.1.
- **Redukovaná data** - Obdobné jako v experimentu 5.5.1, ale mnohem více označených dat pozitivně (téměř polovina) a tudíž horší metriky. Model se naučil velice špatně (skoro jako náhoda), `50%` dat ohodnotí pozitivně a pouze pro `50%` očekávaných ohodnocení se trefí.
- **Vážená data** - Vážená datová sada je stejně jako v předchozích experimentech nejlepší, ale ani zdaleka nedosahuje takových hodnot. Ohodnotí pozitivně zhruba stejné množství profilů, ale s mnohem nižším `precision` i `recall`.
- **SMOTE data** - Obdobné jako pro redukovanou datovou sadu.
- **SMOTE redukovaná data** - SMOTE redukovaná datová sada dosahuje podobných výsledků jako ta vážená, pouze má prohozené `precision` a `recall`. Je to způsobené ohodnocením pozitivně mnohem více profilů.

Vybrané atributy a jejich důležitost z výsledků pro váženou datovou sadu je možné vidět na obrázku 5.6. Mezi hlavními atributy jsou 4 nejznámější filmové žánry, kterými jsou `komedie`, `romantický`, `akční` a `drama`. Tyto filmové žánry zřejmě nemají nic společného s produkty zvolené reklamní kampaně a tudíž lze usoudit, že naučení modelu je spíše náhodné. Zajímavým atributem jsou `polévky` nebo `předkrmy` z receptů, které by mohly mít něco společného s výživovými produkty.



Obrázek 5.6: Graf míry důležitosti atributů pro váženou datovou sadu.

5.6 Vyhodnocení experimentů

Na základě 3 experimentů ze sekce 5.5 je možné vidět, že vždy byla nejlepší vážená datová sada pro naučení klasifikátoru. Při porovnání experimentu pro všechny atributy (5.5.1) a experimentu pro atributy bez nových informací (5.5.2) je zřejmé, že se naučily na velice podobných datech a dosáhly podobných výsledků. Experiment s pouze novými informacemi (5.5.3) se nenaučil téměř nic, ale mírná snaha tam je. Domněnka, že by mohly informace z webového serveru `recepty.cz` pomoci ve zvolené reklamní kampani, byla nejspíše mylná. Pokud mají tyto informace nějaký vliv, tak poměrně malý. Co se týče informací ze serveru `csfd.cz`, tak nemají vliv téměř žádný. Z těchto závěrů lze usoudit, že nově obohacené informace mají mírný přínos pro vyhodnocení profilů, které kliknou na zvolenou reklamní kampaň. Tento přínos je ale téměř zanedbatelný.

Firma se shodla s řešením, že uživatelské profily v případě zvolené reklamní kampaně nejsou příliš specifické. Jejich produkty nemají přesnou skupinu cílení a může si je koupit téměř kdokoliv. V reálném světě existuje mnoho téměř totožných lidí, kde jeden bude jejich produkty opěvovat chválou a druhý přesně naopak.

Nově obohacené informace by mohly mít vysoký přínos v jiných typech reklamních kampaní. Pokud by se jednalo o doporučení knih, filmů nebo her, tak by mohly mít informace z webového serveru `csfd.cz` vysokou prioritu. Naopak pokud by se jednalo o doporučování nějakých jídel nebo restaurací, tak by mohly mít vysokou prioritu informace z webového server `recepty.cz`.

Jedním z vylepšení je tedy výše zmíněné použití jiné reklamní kampaně. Dalším vylepšením by mohlo být získání různorodých informací z více webových serverů. Takovými informacemi by mohly být například sportovní zaměření (běh, basketbal, fotbal, atd.) nebo hudební žánry (rock, pop, metal, atd.). Naučené modely pro klasifikaci by mohly být využity pro vybrání skupiny uživatelských profilů, kterým by se doporučovala reklama s vyšší prioritou.

Kapitola 6

Závěr

Cílem práce bylo obohacení uživatelských profilů o extrahované informace z webových stránek pro účely cílené internetové reklamy a zhodnocení přínosu těchto informací.

Popsána je extrakce dat z webových stránek, která na základě staženého HTML obsahu získá potřebná data a následně je zpracuje a uloží. Dále jsou popsány uživatelské profily a data v nich obsažená. Je znázorněn celý proces obohacování těchto uživatelských profilů o různé informace a požadavky na nové informace. Nezbytnou součástí je princip vytváření cílových skupin z těchto profilů. Na závěr jsou popsány přístupy strojového učení včetně předzpracování a klasifikačních metrik.

Na základě studijní etapy bylo možné navrhnout a implementovat streamovací aplikace pro obohacení uživatelských profilů o nové informace. Nejdříve bylo navrženo a implementováno mapování informací do dvou přehlednějších formátů. Poté byly navrženy a implementovány aplikace schopné zpracovávat stažená data z webových stránek a uložit je do databáze pro URL adresy. Na závěr byly navrženy a implementovány úpravy stávajících aplikací a nové aplikace pro úspěšné uložení informací do databází uživatelských profilů.

Obohacené uživatelské profily byly využity pro následující analýzy přínosu nových informací pro účely cílené reklamy. Vstupní data byla zvolena pro jednu reklamní kampaň, kde bylo možné vybrat profily, které klikly na zobrazenou reklamu. Tato vstupní data byla pomocí technik předzpracování upravena do potřebných podob. Na předzpracování navazovala příprava dat, která rozdělila data na trénovací a testovací a vyřešila nepoměr klasifikačních tříd v datech vytvořením více různých trénovacích sad. Pomocí zvoleného modelu strojového učení byly provedeny experimenty pro zjištění přínosu nových informací v uživatelských profilech.

Vyhodnocením experimentů se došlo k závěru, že nově získané informace jsou mírně prospěšné ve zvolené reklamní kampani. Tyto informace by mohly mít mnohem vyšší přínos ve specificky zaměřených kampaních. Jedním z vylepšení by mohla být extrakce dalších různorodých informací z různých webových serverů. Naučené modely pro klasifikaci by mohly být využity pro vybrání skupiny uživatelských profilů, kterým by se doporučovala reklama s vyšší prioritou.

Literatura

[1] *Aerospike Documentation*. [Online; navštíveno 7.1.2019].
URL https://www.aerospike.com/docs

[2] *Apache Kafka Introduction*. [Online; navštíveno 7.1.2019].
URL https://kafka.apache.org/intro

[3] *Apache Kafka Tutorial*. [Online; navštíveno 7.1.2019].
URL https://www.tutorialspoint.com/apache_kafka

[4] *Elasticsearch Basic Concepts*. [Online; navštíveno 7.1.2019].
URL https://www.elastic.co/guide/en/elasticsearch/reference/current/getting-started-concepts.html

[5] *Elasticsearch Tutorial*. [Online; navštíveno 7.1.2019].
URL https://www.tutorialspoint.com/elasticsearch

[6] *Imbalanced-learn Documentation*. [Online; navštíveno 30.4.2019].
URL https://imbalanced-learn.readthedocs.io/en/stable/

[7] *Overview of CatBoost*. [Online; navštíveno 30.4.2019].
URL https://catboost.ai/docs/

[8] *Python Web Scraping Tutorial*. [Online; navštíveno 7.1.2019].
URL https://www.tutorialspoint.com/python_web_scraping

[9] *Web Scraping Vs Web Crawling*. [Online; navštíveno 7.1.2019].
URL http://prowebscraping.com/web-scraping-vs-web-crawling

[10] *Elasticsearch with Philipp Krenn*. Duben 2017, [Online; navštíveno 7.1.2019].
URL https://softwareengineeringdaily.com/2017/04/12/elasticsearch-with-philipp-krenn

[11] *What is web scraping – Part 1 – Beginner's guide*. Leden 2018, [Online; navštíveno 7.1.2019].
URL https://www.scrapehero.com/a-beginners-guide-to-web-scraping-part-1-the-basics

[12] Bellec, A.: *A brief overview of Automatic Machine Learning solutions (AutoML)*. Květen 2018, [Online; navštíveno 13.5.2019].
URL https://techblog.cdiscount.com/a-brief-overview-of-automatic-machine-learning-solutions-automl/

[13] Gardideh, N.: *Microservice Series: Scraper*. Leden 2017, [Online; navštíveno 7.1.2019].
URL https://hackernoon.com/microservice-series-scraper-ee970df3e81f

[14] Jarmul, K.; Lawson, R.: *Python Web Scraping*. Packt Publishing, Second Edition, 2017, ISBN 978-1-78646-258-9.

[15] Joshi, P.: *Artificial Intelligence with Python*. Packt Publishing, 2017, ISBN 978-1-78646-439-2.

[16] Ludert, E. E.: *SMOTE — Synthetic Minority Over-sampling Technique*. Červen 2017, [Online; navštíveno 30.4.2019].
URL https://medium.com/erinludertblog/smote-synthetic-minority-over-sampling-technique-caada3df2c0a

[17] McKinney, W.: *Python for Data Analysis*. O'Reilly Media, Second Edition, 2017, ISBN 978-1-491-95766-0.

[18] Media, E.: *What do you mean by Data preprocessing and why it is needed?* Prosinec 2017, [Online; navštíveno 7.1.2019].
URL
https://www.electronicsmedia.info/2017/12/20/what-is-data-preprocessing

[19] Mitchell, R.: *Web Scraping with Python*. O'Reilly Media, Second Edition, 2018, ISBN 978-1-491-985557-1.

[20] Paul, C.: *Discovering the Children of AI: Machine Learning & Deep Learning*. Květen 2018, [Online; navštíveno 13.5.2019].
URL https://www.functionize.com/blog/discovering-the-children-of-ai-machine-learning-deep-learning/

[21] Raschka, S.: *Python Machine Learning*. Packt Publishing, 2015, ISBN 978-1-78355-513-0.

[22] Stopford, B.: *Designing Event-Driven Systems*. O'Reilly Media, First Edition, 2018, ISBN 978-1-492-03822-1.

[23] Sunasra, M.: *Performance Metrics for Classification problems in Machine Learning*. Listopad 2017, [Online; navštíveno 30.4.2019].
URL https://medium.com/thalus-ai/performance-metrics-for-classification-problems-in-machine-learning-part-i-b085d432082b

[24] Zendulka, J.; Bartík, V.; Lukáš, R.; aj.: *Získávání znalostí z databází. Studijní opora*. FIT VUT v Brně, Září 2010.

Příloha A

Obash CD

- `/implementation/` - Zdrojové soubory pro obohacení uživatelských profilů o nové informace
- `/analysis/` - Zdrojové soubory a vstupní data pro analýzu vlivu rozšíření profilů
- `/thesis/` - Technická zpráva ve formátu PDF
- `/thesis_latex/` - Zdrojové soubory technické zprávy

Pro implementaci obohacení uživatelských profilů o nové informace nelze exportovat knihovny firmy a tudíž jsou obsaženy pouze nově naprogramované části.