Univerzita Pavla Jozefa Šafárika v Košiciach

Prírodovedecká fakulta

METÓDY ASOCIAČNÝCH PRAVIDIEL A ICH APLIKÁCIE

BAKALÁRSKA PRÁCA

Študijný odbor: **Informatika**
Školiace pracovisko: **Ústav informatiky**
Vedúci záverečnej práce: **doc. RNDr. Ľubomír Antoni, PhD.**
Konzultant: **prof. RNDr. Stanislav Krajči, PhD.**

Košice 2024

Vasyl Khorev

Poďakovanie

Rád by som poďakoval vedúcemu bakalárskej práce doc. RNDr. Ľubomírovi Antonimu, PhD. a konzultantovi práce prof. RNDr. Stanislavovi Krajčimu, PhD. za cenné pripomienky a za obetavosť počas tvorby mojej bakalárskej práce. V neposlednom rade veľká vďaka patrí mojej rodine a blízkym priateľom, ktorí ma neustále podporovali počas môjho štúdia.

Univerzita P. J. Šafárika v Košiciach
Prírodovedecká fakulta

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Vasyl Khorev
Študijný program: informatika (jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: Informatika
Typ záverečnej práce: Bakalárska práca
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Metódy asociačných pravidiel a ich aplikácie

Názov EN: Methods of association rules and their applications

Cieľ:
1. Popísať známe algoritmy pre generovanie asociačných pravidiel.
2. Implementovať známe algoritmy pre generovanie asociačných pravidiel a porovnať ich výhody a nevýhody.
3. Aplikovať implementované algoritmy na vybranú údajovú sadu.

Literatúra:
1. Agrawal, R., Imieliński, T., & Swami, A. (1993, June). Mining association rules between sets of items in large databases. In Proceedings of the 1993 ACM SIGMOD international conference on Management of data (pp. 207-216).
2. Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. In Proc. 20th int. conf. very large data bases, VLDB (Vol. 1215, pp. 487-499).
3. Lakhal, L., & Stumme, G. (2005). Efficient mining of association rules based on formal concept analysis. Formal concept analysis, 3626, 180-195.

Vedúci: doc. RNDr. Ľubomír Antoni, PhD.
Konzultant: prof. RNDr. Stanislav Krajči, PhD.
Oponent: prof. RNDr. Gabriel Semanišin, PhD.
Ústav : ÚINF - Ústav informatiky
Riaditeľ ústavu: doc. RNDr. Ondrej Krídlo, PhD.

Dátum schválenia: 14.05.2024

Abstrakt

Hľadanie asociačných pravidiel je dôležitou metódou v oblasti analýzy údajov. Cieľom tejto práce je prezentovať prehľad známych algoritmov v tejto oblasti, porovnať ich výhody a nevýhody a následne ich aplikovať na konkrétnu množinu údajov. V rámci práce sú identifikované a podrobne popísané existujúce algoritmy používané na generovanie asociačných pravidiel. Následne sme tieto algoritmy implementovali a aplikovali na vybraný dataset. Táto aplikácia ukazuje praktickú hodnotu týchto algoritmov v reálnych podmienkach a ich schopnosť odhaliť relevantné vzory v dátach. Vybrané metódy sme v závere porovnali z hľadiska času behu a prahovej hodnoty podpory.

Kľúčové slová: *asociačné pravidlá, analýza údajov, Apriori, FP-growth, GUHA.*

Abstract

Association rules mining is a important method in data mining. The aim of this paper is to provide an overview of the well-known algorithms in this area, compare their advantages and disadvantages, and then apply them to a specific dataset. Existing algorithms used to generate association rules are identified and described in detail within the thesis. We then implemented and applied these algorithms to a selected dataset. This application demonstrates the practical value of these algorithms in real-world settings and their ability to detect relevant patterns in the data. We compared selected methods by running time and values of support.

Keywords: *association rules, data analysis, Apriori, FP-growth, GUHA.*

Obsah

Úvod **7**

1 Asociačné pravidlá **9**

1.1 Základné pojmy 9

1.2 Prehľad súčasného stavu 10

2 Algoritmy na generovanie asociačných pravidiel **12**

2.1 Apriori 13

2.1.1 Princíp alebo myšlienka 13

2.1.2 Postup 13

2.1.3 Výhody a nevýhody 15

2.2 FP-growth 15

2.2.1 Konštrukcia stromu 15

2.2.2 Generovanie frekventovaných množín v strome 18

2.2.3 Výhody a nevýhody 19

3 GUHA asociačné pravidlá **21**

3.1 Reprezentácia dát 22

3.2 Testovanie 22

3.3 Výhody a nevýhody 24

4 Dostupné dáta a použité metódy **25**

4.1 Dostupné dáta 25

4.2 Apriori 26

4.3 FP-growth 27

5 Výsledky **28**

5.1 Efektívnosť 29

6 Diskusia **32**

Záver **34**

Úvod

Hľadanie asociačných pravidiel je dôležitou témou v oblasti dolovania údajov. K asociácii dochádza vtedy, keď niekoľko udalostí navzájom súvisí. Intuitívne by sme mohli asociácie popísať takým spôsobom, že určité údaje (alebo udalosti nájdené v údajoch) sú spojené s inými údajmi alebo udalosťami vyplývajúcimi z týchto údajov. Napríklad štúdia vykonaná v supermarkete môže ukázať, že 65% návštevníkov, ktorí si kúpili zemiakové lupienky, si kúpia aj Coca-Colu, a ak existuje zľava pre takýto súbor, kúpia si ju v 85% prípadov. Ak obchod má informácie o takejto asociácii, je pre manažérov ľahké posúdiť, nakoľko je zľava účinná.

Takéto vyhľadávanie odhaľuje skryté súvislosti v zdanlivo nesúvisiacich údajoch. Tieto súvislosti sa nazývajú **asociačné pravidlá**. Tie, ktoré presahujú určitú hranicu, sa považujú za zaujímavé. Takéto pravidlá dávajú možnosť prijať opatrenia na základe nájdených vzorov. Pomáhajú tiež pri rozhodovaní a vysvetľovaní. Podobne ako väčšina metód data miningu (hĺbková analýza dát), aj táto metóda umožňuje transformovať potenciálne obrovské množstvo informácií na malý a zrozumiteľný súbor štatistických ukazovateľov. Metódy asociačných pravidiel zaraďujeme z hľadiska typu úloh strojového učenia medzi učenie sa bez dozoru, samostatné alebo tiež nekontrolované učenie [1].

Jedným z najčastejšie uvádzaných príkladov hľadania asociačných pravidiel je problém hľadania vzťahov v nákupnom košíku. Cieľom pri hľadaní takýchto asociácií je identifikovať, ktoré produkty sa v košíku často vyskytujú spolu, aby marketingoví odborníci mohli tieto produkty vhodne umiestniť v obchode, prijať ďalšie rozhodnutia, a zvýšiť tak predaj.

Asociačné pravidlá možno použiť aj na analýzu zdravotných symptómov u pacientov v nemocniciach. McCormick a kol. [2] vo svojej štúdii skúmali asociácie medzi faktormi, ako sú etnický pôvod, vysoká hladina cholesterolu, hypertenzia, vek a podmienky liečby. Výsledky ukazujú sklony k infarktu myokardu medzi rôznymi skupinami pacientov. Použitý prístup je obzvlášť vhodný na predikciu ďalších symptómov.

Tieto informácie môžu pomôcť opatrovateľom alebo lekárom pri rozhodovaní v liečbe.

Iný príklad aplikácie dolovania dát v reálnom živote bol popísaný Flachom a kolektívom [3]. Cieľom daného štúdia bolo nájsť „zaujímavé“ asociácie (pravidlá) medzi triedou cesty, podmienkami (počasie, osvetlenie a pod.) a smrteľnosťou nehôd. Tento projekt by mohol pomôcť pochopiť a zlepšiť bezpečnosť na cestách, prijať vybrané opatrenia, ktoré by znížili výskyt a závažnosť nehôd.

V kapitole 1 prezentujeme základné pojmy a prehľad aktuálneho stavu v oblasti dolovania údajov. V kapitole 2 popisujeme známe algoritmy na generovanie asociačných pravidiel. V kapitole 3 prezentujeme teóriu a aplikácie GUHA metód v kontexte asociačných pravidiel. V kapitole 4 detailne popisujeme použitý dátový súbor a metódy analýzy. V kapitole 5 prezentujeme výsledky analýzy vybranej dátovej sady. V kapitole 6 hodnotíme naše výsledky a následne ich porovnávame s inými prácami.

Kapitola 1

Asociačné pravidlá

1.1 Základné pojmy

V tejto podkapitole sa zameriame na definíciu a vysvetlenie základných pojmov. Naším cieľom je poskytnúť čitateľovi jasný a komplexný pohľad na koncepty a termíny, ktoré budeme používať v ďalších častiach práce.

Definícia 1.1 Uvažujme množinu $\mathcal{I} = \{i_1, i_2, \ldots, i_m\}$, pričom m je prirodzené číslo a prvky tejto množiny sú literály, ktoré budeme nazývať aj položky. Ľubovoľnú podmnožinu T množiny $\mathcal{I}$ budeme nazývať *množina položiek* alebo *transakcia*. Nech k je prirodzené číslo. Postupnosť transakcií $\mathcal{D} = (T_1, T_2, \ldots, T_k)$ budeme nazývať *databázou nad* $\mathcal{I}$, pričom $T_j \subseteq \mathcal{I}$ pre $j \in \{1, 2, \ldots, k\}$.

Príklad. Ak $\mathcal{I} = \{\text{mlieko}, \text{maslo}, \text{vajcia}, \text{chlieb}\}$, tak jednou možnou *databázou nad* $\mathcal{I}$ je postupnosť transakcií $(\{\text{chlieb}, \text{maslo}\}, \{\text{mlieko}\}, \{\text{vajcia}, \text{maslo}, \text{chlieb}\})$.

Poznámka. Ku každej transakcii je priradený jedinečný identifikátor, ktorý budeme označovať TID.

Definícia 1.2 Nech $\mathcal{I}$ je množina (všetkých) položiek a nech $X \subseteq \mathcal{I}$, $Y \subseteq \mathcal{I}$ a tiež $X \cap Y = \emptyset$. Dvojicu (X, Y) budeme nazývať *asociačné pravidlo nad* $\mathcal{I}$, pričom ho budeme označovať ako $X \Rightarrow Y$.

Príklad. Nech $\mathcal{I} = \{\text{mlieko, maslo, chlieb, zemiakové lupienky, Coca-Cola}\}$. Potom $\{\text{chlieb}, \text{maslo}\} \Rightarrow \{\text{mlieko}\}$ alebo $\{\text{zemiakové lupienky}\} \Rightarrow \{\text{Coca-Cola}\}$ sú dva príklady asociačných pravidiel nad $\mathcal{I}$.

Poznámka. Množina X sa nazýva *predpoklad* (antecedent) a Y *záver* (consequent) asociačného pravidla nad $\mathcal{I}$.

Definícia 1.3 Definujme funkciu support(X) pre množinu položiek X takto:

$$\text{support}(X) = \frac{|\{\ t \in \mathcal{D}\ :\ X \subseteq t\ \}|}{|\mathcal{D}|}$$

Príklad. Nech $\mathcal{D} = (\{\text{chlieb}, \text{maslo}\}, \{\text{mlieko}\}, \{\text{vajcia}, \text{maslo}, \text{chlieb}\})$.
Potom support$(\{\text{maslo}\}) = \frac{2}{3}$.

Poznámka. Hovoríme, že pravidlo $X \Rightarrow Y$ má *podporu* c, ak support$(X \cup Y) = c$.

Poznámka. V literatúre sa môžete stretnúť aj s pojmom absolútnej hodnoty podpory. Podpora sa zvykne v literatúre označovať aj ako nosič.

Definícia 1.4 Definujme funkciu confidence$(X \Rightarrow Y)$ pre pravidlo $X \Rightarrow Y$ takto:

$$\text{confidence}(X \Rightarrow Y) = \frac{\text{support}(X \cup Y)}{\text{support}(X)}$$

Príklad. Nech $\mathcal{D} = (\{\text{chlieb}, \text{maslo}\}, \{\text{mlieko}\}, \{\text{vajcia}, \text{maslo}, \text{chlieb}\})$.
Potom confidence$(\{\text{maslo}\} \Rightarrow \{\text{chlieb}\}) = \frac{\text{support}(\{\text{maslo,chlieb}\})}{\text{support}(\{\text{maslo}\})} = 1$.

Poznámka. Hovoríme, že pravidlo $X \Rightarrow Y$ má *spoľahlivosť* c, ak confidence$(X \Rightarrow Y) = c$.

Definícia 1.5 Nech min_sup je vopred stanovená prahová hodnota podpory. Množina položiek X je *frekventovaná*, ak platí support$(X) \geq$ min_sup.

1.2 Prehľad súčasného stavu

Vzhľadom na súčasné trendy v oblasti analýzy dát a rastúci objem údajov, metódy asociačných pravidiel sa stávajú čoraz dôležitejšími. Pri takomto objeme údajov je neustále ťažšie pochopiť proces ich analýzy. Samozrejme, všetky tieto faktory vyžadujú, aby algoritmy boli škálovateľné a efektívne. Spomenutý problém sa stal hnacou silou pre vývoj a zdokonalenie metód automatizovanej analýzy údajov.

Asociačné pravidlá získali popularitu najmä vďaka publikáciám Agrawala a kol. V prvej [4] sa zaoberali pravidlami, pričom pravá strana pravidla je jedna položka. Predstavili algoritmus **AIS**, ktorý generuje všetky významné pravidlá medzi položkami v transakciách. Účinnosť svojho algoritmu ukázali tým, že ho použili na údajoch o predaji, získaných od veľkej obchodnej spoločnosti. Autori demonštrovali, že algoritmus je pre danú údajovú sadu efektívny v porovnaní s inými prístupmi.

V druhom článku [5] došlo k vylepšeniu prístupu ku generovaniu kandidátnych množín. Na rozdiel od predchádzajúcej práce, kde boli kandidátne množiny generované priebežne počas skenovania, v novom algoritme sú teraz kandidátne množiny generované rozšírením frekventovaných množín o jeden prvok. Taktiež na rozdiel od predošlého článku dovoľujú, aby pravá strana pravidla mala viac ako jednu položku. Tento algoritmus dostal názov **Apriori**.

V skutočnosti, i keď sú tieto články jednými z najcitovanejších v oblasti dolovania dát, nemožno ich považovať za prvé v histórii. Dávno predtým, v roku 1966 Hájek a kolektív [6] preskúmavali podobné implikačné hypotézy, hoci samotný termín „asociačné pravidlá“ nepoužívali.

Vzhľadom na pokrok vo výkone a rastúci záujem o túto oblasť sa do skúmania tejto témy pustilo čoraz viac výskumníkov. Boli vyvinuté viaceré ďalšie algoritmy na hľadanie frekventovaných množín v dátach.

V roku 2000 Zaki predstavil iný prístup [7] – celý priestor vyhľadávania frekventovaných množín položiek sa rozdelí na malé, nezávislé podpriestory pomocou metód založených na prefixoch alebo maximálnych klikách. Každý podproblém potom možno riešiť pomocou postupu vyhľadávania zdola nahor, zhora nadol alebo hybridného postupu vyhľadávania a celý proces zvyčajne zaberá len niekoľko skenovaní databázy. Tento algoritmus sa nazýva **Eclat** (**E**quivalence **C**lass Clustering and bottom-up **La**ttice **T**raversal).

Ďalším populárnym algoritmom na dolovanie údajov je **FP-Growth** (Frequent Pattern Growth). Predstavili ho Han a kol. vo svojom článku [8]. Tento algoritmus využíva odlišný prístup v porovnaní s tradičnými algoritmami, ako sú *Apriori* alebo *Eclat*. Navrhli novú stromovú štruktúru frekventovaných množín a vyvinuli efektívnu metódu založenú na takomto strome. Veľká databáza je komprimovaná do menšej štruktúry, čím sa predišlo zbytočnému skenovaniu databázy. Boli taktiež implementované ďalšie sofistikovanejšie vylepšenia.

Delgado a kol. [9] zaviedli univerzálny model pre fuzzy transakcie. Demonštrovali, že ich model je jednoducho formulovateľný a použiteľný. Tieto poznatky sa môžu aplikovať pri dolovaní multimediálnych údajov a analýze obsahu webových stránok.

Nové, iné prístupy k eliminácii zbytočných pravidiel a generovaniu ich menšieho počtu boli prezentované v [10, 11].

Kapitola 2

Algoritmy na generovanie asociačných pravidiel

V tejto kapitole budeme analyzovať algoritmy, ktoré sa používajú pri generovaní asociačných pravidiel. Na týchto algoritmoch je založená analýza vzťahov v nami uvažovanom súbore údajov (kapitola 4). Úplné pochopenie týchto algoritmov je veľmi dôležité pre správnu analýzu.

Problém hľadania asociačných pravidiel môže byť rozdelený na dve úlohy [4]:

1. Nájsť všetky množiny položiek, ktorých podpora je väčšia ako prahová hodnota podpory. Množina, ktorá túto podmienku spĺňa, sa volá *frekventovaná*.

2. Použitím nájdených frekventovaných množín vygenerovať všetky pravidlá. Pre danú úlohu máme priamočiary algoritmus. Pre každú frekventovanú množinu Y nájdeme všetky jej neprázdne podmnožiny a pre každú takú podmnožinu X vrátime na výstup pravidlo v tvare

$$X \Rightarrow Y \setminus X$$

vtedy, ak dané pravidlo spĺňa podmienku minimálnej hodnoty spoľahlivosti.

2.1 Apriori

V tejto podkapitole sa zameriame na algoritmus *Apriori*, ktorý je jedným z najpopulárnejších algoritmov, ktoré dokážu identifikovať asociačné pravidlá. Podrobne sa budeme venovať jeho základnému princípu a tiež tomu, ako hľadá *frekventované* množiny položiek v údajoch.

2.1.1 Princíp alebo myšlienka

Tento algoritmus bol navrhnutý Agrawalom a Srikanthom v roku 1994 [5]. Svoj názov dostal vďaka predpokladu, ktorý nazývame „a priori princíp“. Ten hovorí, že *každá podmnožina frekventovanej množiny položiek je tiež frekventovaná*. Ekvivalentne, *ak je množina nefrekventovaná, tak každá jej nadmnožina je tiež nefrekventovaná*[1]. Vďaka tomuto faktu možno efektívne zúžiť oblasť hľadania. Daný princíp umožňuje postupne zväčšovať veľkosť množín, a takto nájsť všetky frekventované množiny.

2.1.2 Postup

Algoritmus vykoná niekoľko prechodov databázou. Pri prvom prechode sa jednoducho počítajú hodnoty *podpory* jednotlivých položiek – tie, ktoré spĺňajú určenú prahovú hodnotu, sú považované za frekventované jednoprvkové množiny. V k-tom kroku na základe frekventovaných množín nájdených v $(k-1)$. kroku a pomocou funkcií *apriori-gen* budu vygenerované k-prvkové kandidátne množiny. V tomto kroku následne prejdeme databázou a spočítame *podporu* kandidátnych množín. Na konci prechodu sa vyberú iba tie kandidátne množiny, ktoré sú frekventované, a tie sa použijú v ďalšom kroku. Algoritmus sa vykonáva, kým všetky kandidátne množiny z predchádzajúceho prechodu nemajú podporu menšiu ako zadaná prahová hodnota.

Poznámka. Predpokladáme, že položky v každej transakcii sú zoradené v lexikografickom poradí.

Nižšie je uvedený podrobný pseudokód Agrawalovho algoritmu (algoritmus 1).

[1] V literatúre sa môžete stretnúť aj s pojmom *antimonotónnosť* alebo *downward closure*.

Algoritmus 1 Apriori

$L \leftarrow \{$frekventované jednoprvkové množiny$\}$;
for $(k = 2;\ L_{k-1} \neq \emptyset;\ k++)$ **do**
  $C_k \leftarrow$ apriori-gen(L_{k-1}); ▷ kandidátne množiny
  for all $t \in \mathcal{D}$ **do**
    $C_t \leftarrow \{c \in C_k\ :\ c \subseteq t\}$; ▷ kandidáti obsiahnutí v t
    for all $c \in C_t$ **do**
      $c.\text{count} \leftarrow c.\text{count} + 1$;
  $L_k \leftarrow \{c \in C_k\ :\ c.\text{count} \geq \text{min\_sup}\}$;
return $\bigcup_{i=2}^{k} L_i$

Funkcia apriori-gen, ktorá berie na vstup všetky $(k-1)$-prvkové frekventované množiny a vracia všetky k-prvkové frekventované množiny, pozostáva z dvoch krokov:

1. **JOIN**: spájame L_{k-1} s L_{k-1}:

   for all $p \in L_{k-1}$ **do**
     for all $q \in L_{k-1}$ **do**
       /* rovnosť prvých $k-2$ položiek */
       if $p.\text{item}_i = q.\text{item}_i$ pre $i \in \{1, \ldots, k-2\}$ **then**
         if $p.\text{item}_{k-1} < q.\text{item}_{k-1}$ **then**
           $c \leftarrow \{\ p.\text{item}_1,\ p.\text{item}_2,\ \ldots,\ p.\text{item}_{k-1},\ q.\text{item}_{k-1}\ \}$;
           $C_k \leftarrow C_k \cup c$;

2. **PRUNE**: mažeme všetky položkové množiny c také, že existuje nejaká $(k-1)$-prvková podmnožina c, ktorá nie je v L_{k-1}:

   for all $c \in C_k$ **do**
     for all $s \subseteq c$ **do**
       if $|s| = k-1$ **and** $s \notin L_{k-1}$ **then** ▷ $(k-1)$-prvková množina
         $C_k \leftarrow C_k \setminus c$;

Príklad. Nech L_3 je $\{\{1,2,3\},\{1,2,4\},\{1,3,4\},\{1,3,5\},\{2,3,4\}\}$. Po **JOIN** kroku máme $C_4 = \{\{1,2,3,4\},\{1,3,4,5\}\}$. Aplikovaním **PRUNE** kroku sa vymaže množina $\{1,3,4,5\}$, lebo jej podmnožina $\{1,4,5\}$ nepatrí L_3. Potom $C_4 = \{\{1,2,3,4\}\}$.

Kandidátne k-prvkové množiny položiek sú generované spojením frekventovaných $(k-1)$-prvkových množín na základe rovnosti prvých $(k-2)$ položiek a vymaza-

ním tých, ktoré obsahujú akúkoľvek podmnožinu, ktorá nie je frekventovaná. Vďaka tomuto postupu sa vytvorí výrazne menej kandidátnych množín.

2.1.3 Výhody a nevýhody

Hlavnou nevýhodou tohto algoritmu je jeho výpočtová náročnosť. Spracovanie veľkého počtu kandidátnych množín je nákladné. Napríklad, ak existuje 10^4 frekventovaných jednoprvkových množín, tak Apriori potrebuje vygenerovať viac ako 10^7 dvojprvkových množín. Okrem toho, aby sme našli nejakú frekventovanú množinu mohutnosti 100, ako napr. $\{a_1, \ldots, a_{100}\}$, musí algoritmus vygenerovať viac ako $2^{100} \approx 10^{30}$ kandidátov [8].

Na druhej strane, algoritmus je jednoduchý na pochopenie a implementáciu.

2.2 FP-growth

Algoritmus bol navrhnutý v roku 2000 Hanom, Peiom a Yinom [8]. Predstavuje značné vylepšenie oproti predchádzajúcemu – nevyžaduje generovanie kandidátnych množín, čo vedie k podstatnému zvýšeniu výkonu.

FP-growth algoritmus sa skladá z dvoch fáz:

1. v prvej fáze sa **skonštruuje** FP-strom, ktorý je kompaktnou reprezentáciou dát.
2. z tohto stromu algoritmus **vygeneruje** frekventované množiny.

2.2.1 Konštrukcia stromu

Algoritmus reprezentuje databázu pomocou stromovej štruktúry nazývanej **strom frekventovaných vzorov** (angl. frequent pattern tree), skrátene **FP-strom**. Táto štruktúra umožňuje efektívne dopytovanie na frekvencie (podporu) ľubovoľnej množiny a je navrhnutá tak, aby bola kompaktná a zároveň nevyžadovala viacnásobné prechody databázou pri jej konštrukcii.

Logicky vzaté, takýto strom nemusí obsahovať nefrekventované položky[2]. Každý uzol stromu okrem koreňa má nasledovné atribúty:

- **item-name:** zaznamenáva, ktorú položku tento uzol predstavuje.
- **count:** počet transakcií, ktoré obsahujú položky na ceste k danému uzlu.

[2]Každá množina, ktorá obsahuje nejakú nefrekventovanú položku, frekventovanou byť nemôže.

- **node-link:** odkazy na ďalšie uzly v strome, ktoré majú rovnaký názov.

Pre jednoduchšie vyhľadávanie a prístup k vrcholom je vytvorená hlavičková tabuľka. Každý záznam v tabuľke reprezentuje jednu položku a obsahuje odkaz na uzol v strome, kde sa daná položka nachádza.

Každá transakcia z databázy sa mapuje na cestu v strome. Celá konštrukcia je popísaná v algoritme 2.

Algoritmus 2 Konštrukcia FP-stromu

Require: databáza $\mathcal{D}$

Ensure: FP-strom T

1: zostupne usporiadať položky $i \in \mathcal{I}$ podľa hodnoty support(i) ▷ Definované v kapitole 1;
2: **for all** transakcia $\in \mathcal{D}$ **do** ▷ Definované v kapitole 1
3: $T \leftarrow$ koreň stromu
4: **for all** $i \in$ transakcia **do** ▷ Definované v kapitole 1
5: **if** T má dieťa N **AND** $N.$item-name $= p.$item-name **then**
6: N.count $++$;
7: **else**
8: $N \leftarrow$ nový uzol, ktorý bude potomkom uzla T;
9: N.count $\leftarrow 1$;
10: $N.$ node-link $\leftarrow$ iný uzol, ktorý má rovnaký názov;
11: $T \leftarrow N$;

Na riadku 1 bude vytvorená hlavičková tabuľka a definované poradie pre položky v transakciách. Tým, že položky usporiadame, budú časté položky uprednostnené na začiatku transakcie a pri vkladaní do stromu sa frekventované položky budú vkladať bližšie ku koreňu stromu, čím sa zvyšuje šanca, že sa táto položka bude zdieľať medzi viacerými transakciami.

Riadok 6 zodpovedá prechodu po už existujúcej ceste. V takomto prípade sa jednoducho zvýši počítadlo daného uzlu, čím sa šetrí čas a zdroje.

Riadky 8–10 riešia prípad, kedy v strome neexistuje príslušný vrchol, do ktorého by sa dalo vstúpiť. V takom prípade taký vrchol vytvoríme a inicializujeme mu všetky potrebné hodnoty.

Pre takto zostrojený strom platí, že obsahuje kompletnú informáciu o databáze z pohľadu hľadania frekventovaných množín. Totiž každej transakcii zodpovedá jedna

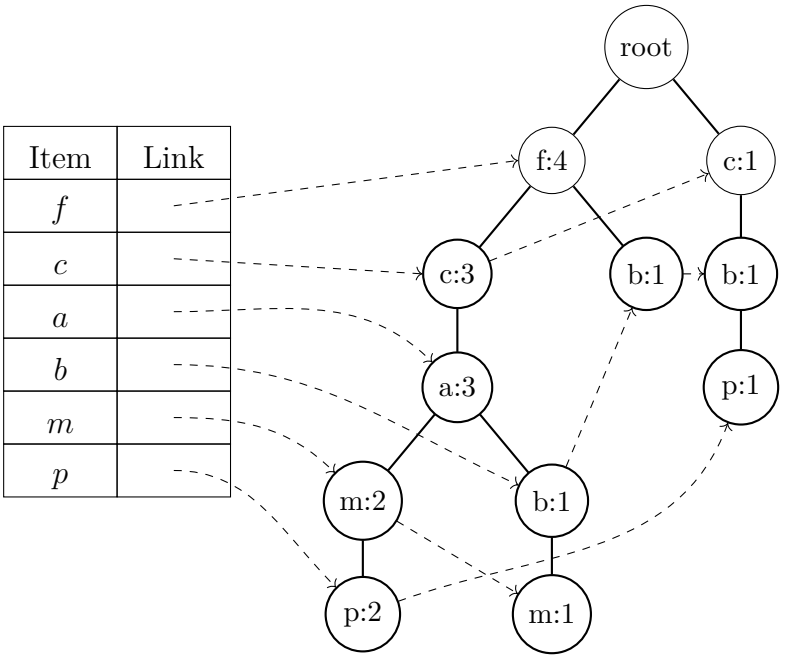
cesta a všetky potrebné informácie, ako napr. hodnota *podpory*, sú v strome uložené.

Je povšimnutiahodné, že rozmer stromu je ohraničený celkovým výskytom frekventovaných množín v databáze a výška je ohraničená veľkosťou najväčšej frekventovanej množiny v akejkoľvek transakcii [8].

Príklad. Uvažujme databázu $\mathcal{D}$ na Obr. 2.1 a minimálnu prahovú hodnotu podpory min_sup = $\frac{3}{5}$. FP-strom zostrojený na tejto databáze je zobrazený na Obr. 2.2.

Databáza $\mathcal{D}$	Usporiadané
f, a, c, d, g, i, m, p	f, c, a, m, p
a, b, c, f, l, m, o	f, c, a, b, m
b, f, h, j,o	f, b
b, c, k, s, p	c, b, p
a, f, c, e, l, p, m, n	f, c, a, m, p

Obr. 2.1: Príklad databázy $\mathcal{D}$ [8].



Obr. 2.2: Príklad FP-stromu a hlavičkovej tabuľky [8]

2.2.2 Generovanie frekventovaných množín v strome

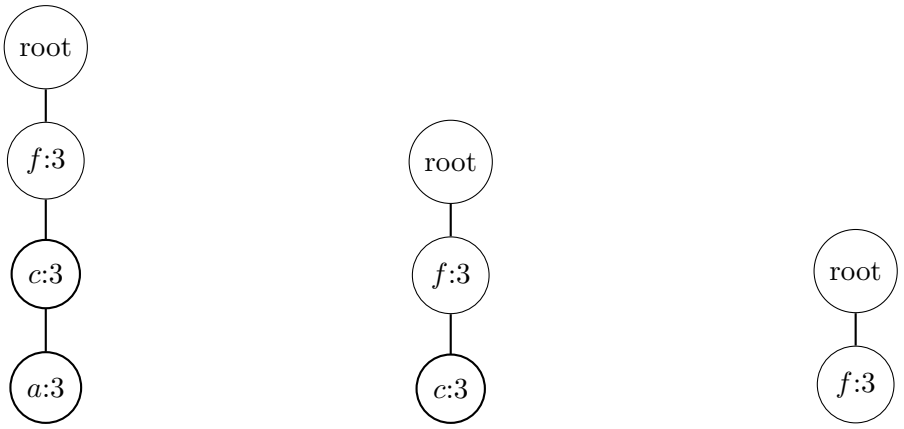
Aby sme komplexne pochopili myšlienku generovania frekventovaných množín v strome, je nutné na úvod predstaviť kľúčové vlastnosti navrhovanej štruktúry.

Uvažujme iba tie transakcie, kde sa vyskytuje konkrétna položka a_i. Takéto transakcie budú v strome vyzerať ako prefixové cesty. Všetky ich spolu nazývame *podmienený základ vzoru* a_i.

Príklad. Zoberme si FP-strom zobrazený na Obr. 2.2. Ak vezmeme do úvahy len transakcie, ktoré obsahujú položku m, dostaneme v strome tieto dve prefixové cesty: $(f{:}2, c{:}2, a{:}2)$ a $(f{:}1, c{:}1, a{:}1, b{:}1)$. Spolu tvoria *podmienený základ vzoru* m.

Ak máme podmienený základ vzoru a_i, možno pomocou neho zostrojiť FP-strom. Tento strom budeme volať *podmienený FP-strom* a_i a označovať $T' = T|a_i$.

Príklad. Uvažujme *podmienený základ vzoru* m s predchádzajúceho príkladu. Na Obr. 2.3a je zobrazený podmienený strom $T|m$.



(a) Podmienený FP-strom $T' = T|m$.

(b) Podmienený FP-strom $T'' = T'|a = T|am$.

(c) Podmienený FP-strom $T''' = T''|c = T|cam$.

Obr. 2.3: Hľadanie frekventovaných mnozin obsahujúcich m pomocou T'.

Prezentované nižšie vety tvoria základ algoritmu na hľadanie frekventovaných množín v databázach transakcií. Veta 2.1 umožňuje rekurzívne prehľadávanie podmienených FP-stromov, veta 2.2 definuje podmienky pre ukončenie rekurzie. Dôkazy vynecháme (tie je možné nájsť v [8]).

Veta 2.1 *Nech α je frekventovaná položková množina v $\mathcal{D}$, B je podmienený základ položky α a β je množina položiek v B. Potom $\alpha \cup \beta$ je frekventovaná v $\mathcal{D}$ práve vtedy, keď je β frekventovaná v B.*

Veta 2.2 *Nech FP-strom má iba jednu cestu* $(a_1{:}s_1, a_2{:}s_2, \ldots, a_k{:}s_k)$. *Potom ľubovoľná podmnožina* $A \subseteq \{a_1, \ldots, a_k\}$ *je frekventovaná, pričom platí*

$$support(A) = \min(\{s_i : a_i \in a\ \}).$$

Z toho vyplýva, že na proces generovania frekventovaných položkových množín sa dá pozerať ako na generovanie frekventovaných položiek a potom na postupné zväčšovanie (growing) každej takejto množiny položiek dolovaním jej podmieneného základu.

Podrobný pseudokód je popísaný v algoritme 3. Pozorný čitateľ si môže všimnúť podobnosť daného prístupu a prístupu „rozdeľuj-a-panuj“.

Algoritmus 3 Procedúra FP-growth(T, α)

```
if T obsahuje iba jednu cestu P then
    for all množinu β : β ⊆ P do
        if minimálna podpora položiek v β > min_supp then
            generuj množinu α ∪ β;
else
    for all a_i v hlavičkovej tabuľke T začínajúc najmenej častou položkou do
        if support(a_i) > min_sup then
            generuj množinu β = α ∪ a_i;
            T_β ← podmienený FP-strom T|β;
            if T_β ≠ ∅ then
                FP-growth(T_β, β)                          ▷ rekurzia
```

Na Obr. 2.3 sú zobrazené stromy pri volaní vetvy výpočtu FP-growth$(T', m) \rightarrow$ FP-growth$(T'', a) \rightarrow$ FP-growth(T''', c). Týmto výpočtom sa nájdu všetky frekventované množiny, ktoré obsahujú $\{a\}$, $\{a, m\}$, $\{c, a, m\}$ a to sú: $\{a, m\}$, $\{c, a, m\}$, $\{f, a, m\}$ a $\{f, c, a, m\}$. Algoritmus systematicky prejde všetky možnosti a identifikuje všetky frekventované množiny.

2.2.3 Výhody a nevýhody

Autori tohto prístupu uvádzajú, že rozmer FP-stromu je zvyčajne oveľa menší ako rozmer databázy. Podobne, podmienený FP-strom je obvykle oveľa menší ako pôvodný strom (podmienený FP-strom je podstromom pôvodného stromu). Vďaka tomuto princípu každá nasledujúca iterácia pracuje so stále redukovaným množstvom

dát. Z hľadiska efektivity je tak algoritmus FP-growth **rádovo efektívnejší** v porovnaní s algoritmom Apriori.

Hoci algoritmus FP-growth je rýchlejší, jeho implementácia je komplexnejšia v porovnaní s algoritmom Apriori.

Kapitola 3

GUHA asociačné pravidlá

V tejto kapitole sa zameriame na teoretické základy a aplikácie metódy nazývanej Generalized Universal Hypothesis Automaton (GUHA) v kontexte asociačných pravidiel.

Metóda GUHA sa opiera o jednoduchú, ale zároveň presnú matematickú a štatistickú analýzu. To umožňuje vygenerovať všetky možné tvrdenia (pravidlá), následne ich pomocou počítača automaticky overiť [6]. Človek si potom môže zo všetkých platných hypotéz oddeliť iba dôležité tvrdenia, ohodnotiť ich a na ich základe zvoliť ďalší smer/postup výskumu.

Podstatou metódy je teda (tieto funkcie má vykonávať počítač):

1. optimálnym spôsobom na základe súboru dát vygenerovať čím viac hypotéz,
2. overiť každú takú hypotézu,
3. vypísať iba zaujímavé tvrdenia.

Tato metóda umožňuje hľadať nielen pravidlá, ktoré platia pre všetky (alebo takmer všetky) objekty v databáze, ale aj také, ktoré neplatia pre všetky (alebo takmer všetky) objekty. Takýto prístup má potenciálne využitie v rôznych oblastiach, napríklad v analýze kybernetickej bezpečnosti [12].

GUHA metóda má význam aj pri generovaní tzv. *negatívnych*[1] pravidiel, ktoré zachytávajú vzťahy medzi prítomnými a neprítomnými položkami v dátach, teda sú tvaru $A \Rightarrow \neg B$. Negatívne pravidlá dôkazu odhaliť nezreteľné vzťahy v dátach, ktoré by klasické pravidlá nezachytili. To umožňuje hlbšie pochopenie údajovej sady a otvára nové možnosti analýzy.

[1] Opačne, klasické pravidlá sa v podobnej literatúre označujú ako *pozitívne*.

Práve takéto pravidlá možno použiť na detekciu potenciálnych chýb v kóde [13]. Na základe datasetov troch prieskumov cestovania obyvateľov Hongkongu boli vygenerované pravidlá, ktoré pomôžu odborníkom a politikom prijímať vhodné rozhodnutia a rozvíjať oblasť cestovného ruchu [14].

3.1 Reprezentácia dát

GUHA procedúry pracujú s dátami vo forme obdĺžnikovej matice, ktorej riadky zodpovedajú objektom a stĺpce vlastnostiam, teda prvok v i-tom riadku a j-tom stĺpci je 0 a 1 podľa toho, či i-tý objekt má j-tú vlastnosť. V prípade analýzy nákupného koša, objektami môžu byť transakcie a vlastnosťami položky. Hodnota v príslušnom riadku a stĺpci v tomto prípade indikuje, či bola táto položka nakúpená v danej transakcii.

Neskoršie implementácie fungujú s údajmi reprezentovanými pomocou bitových reťazcov.

V GUHA terminológii na vyjadrenie vzťahu medzi dvoma množinami atribútov X a Y sa namiesto klasického implikačného pravidla $X \Rightarrow Y$, používa označenie $X \approx Y$, kde symbol $\approx$ reprezentuje jeden z rôznych štatistických testov (ukazovateľov, kvantifikátorov), ktoré sa používajú v GUHA procedúrach.

3.2 Testovanie

V tejto časti popíšeme ako otestovať platnosť GUHA asociačného pravidla v dátach. Predstavíme si rôzne štatistické testy, ktoré sa na tento účel bežne používajú.

Pre dva binárne atribúty φ, ψ asociácie $\varphi \approx \psi$ budeme počítať:

- a – počet objektov v $\mathcal{D}$ spĺňajúcich φ a ψ,
- b – počet objektov v $\mathcal{D}$ spĺňajúcich φ a $\neg\psi$,
- c – počet objektov v $\mathcal{D}$ spĺňajúcich $\neg\varphi$ a ψ,
- d – počet objektov v $\mathcal{D}$ spĺňajúcich $\neg\varphi$ a $\neg\psi$.

Tieto štyri hodnoty spolu tvoria kontingenčnú tabuľku znázornenú v Tab. 3.1.

$\mathcal{D}$	ψ	$\neg\psi$
φ	a	b
$\neg\varphi$	c	d

Tabuľka 3.1: 4ft kontingenčná tabuľka

Kvantifikátor je funkcia, ktorá každej štvorice čísel a, b, c, d priradí pravdivostnú hodnotu 0 alebo 1. Hodnota 1 indikuje, že medzi atribútmi asociácia platí, zatiaľ čo hodnota 0 indikuje, že neexistuje žiadna relevantná asociácia. V kontexte GUHA pravidiel kvantifikátor nám odpovedá na otázku: „Platí skúmaná asociácia v tomto súbore údajov?“

Nižšie uvedieme zopár príkladov možných kvantifikátorov (funkcií):

1. p-implikácia [15]. Nech p je parameter a platí $p \in (0, 1]$, potom:

$$\mathrm{q}(a, b, c, d) = \begin{cases} 1, & \text{ak } \frac{a}{a+b} \geq p \\ 0, & \text{inak} \end{cases}$$

2. FIMPL (**F**ounded **impl**ication). Nech p a s sú parametre a platí $s \geq 1$, $p \in (0, 1]$, potom:

$$\mathrm{q}(a, b, c, d) = \begin{cases} 1, & \text{ak } a \ \geq s \ \wedge \ \frac{a}{a+b} > p \\ 0, & \text{inak} \end{cases}$$

   Za použitia daného kvantifikátora, asociácia $\varphi \approx_{s,p} \psi$ a predtým uvažované „klasické“ asociačné pravidlo $\varphi \Rightarrow \psi$ s podporou $\frac{s}{a+b+c+d}$ a spoľahlivosťou p sú **ekvivalentné**[2].

   Príklad. Majme databázu $(\{\text{chlieb}, \text{maslo}\}, \{\text{mlieko}\}, \{\text{vajcia}, \text{maslo}, \text{chlieb}\})$. Uvažujme asociáciu $\{\text{maslo}\} \approx \{\text{chlieb}\}$. Kontingenčná tabuľka pre tieto atribúty je znázornená v Tab. 3.2.

$\mathcal{D}$	chlieb	¬chlieb
maslo	2	0
¬maslo	0	1

Tabuľka 3.2: $4ft$ kontingenčná tabuľka

V tomto prípade má daná asociácia absolútnu podporu 2 a spoľahlivosť $\frac{2}{2+1} = \frac{2}{3}$.

[2]V GUHA terminológii sa zvykne používať pojem absolútnej podpory.

3. SIMPLE (po angl. jednoduchý). Nech s je parameter, potom:

$$\mathrm{q}(a,b,c,d) = \begin{cases} 1, & \text{ak } ad > bc \text{ a } a \ \geq s \\ 0, & \text{inak} \end{cases}$$

Takáto asociácia $\varphi \approx \psi$ hovorí, že „ψ sa vyskytuje častejšie v φ ako v $\neg\varphi$“.

4. LIMPL (**L**ower critical **impl**ication) [16]. Ak s, p a α sú parametre, potom:

$$\mathrm{q}(a,b,c,d) = \begin{cases} 1, & \text{ak } a \ \geq s \ \text{ a } \ \sum_{i=a}^{a+b} \binom{a+b}{i} \cdot p^i \cdot (1-p)^{a+b-i} \leq \alpha \\ 0, & \text{inak} \end{cases}$$

Ďalšie kvantifikátory, ktoré presahujúce rámec tejto práce, je možné nájsť v [17, 18].

3.3 Výhody a nevýhody

Vďaka veľkému množstvu kvantifikátorov majú analytici široký priestor pri analýze[3]. To umožňuje odhaliť neočakávané vzťahy v dátach, ktoré by pri použití klasických metód „data miningu“ mohli byť prehliadnuté.

Táto metóda môže byť použitá na rôzne typy dát bez nutnosti špecifických predpokladov o ich distribúcii.

Pri analýze reálnych údajov (v závislosti od veľkosti dát) môže GUHA metóda vygenerovať enormné množstvo hypotéz, drvivá väčšina ktorých môže byť z pohľadu analytika úplne nezmyselná a irelevantná. Pri veľkom počte hypotéz sa samozrejme zvyšujú aj výpočtové a časové nároky.

[3]Zdatný štatistik si dokonca môže definovať aj nový kvantifikátor a analyzovať jeho vlastnosti.

Kapitola 4

Dostupné dáta a použité metódy

V tejto kapitole popíšeme dáta, s ktorými sme pracovali v rámci bakalárskej práce. Tiež popíšeme spôsob použitia metód z prvých troch kapitol, ktoré sme na nich aplikovali. Konkrétne výsledky budú prezentované v piatej kapitole.

4.1 Dostupné dáta

Na analýzu pomocou asociačných pravidiel sme použili dátový súbor „The Bread Basket“ dostupný na platforme Kaggle [19]. Tento súbor dát obsahuje informácie o transakciách v pekárni „The Bread Basket“ v Edinburghu v Škótsku.

Dataset je verejne dostupný pre stiahnutie vo formáte `.csv`. Obsahuje **20507** záznamov, viac ako **9000** transakcií a **5** atribútov:

- **No:** jedinečný identifikátor transakcie (číslo z rozsahu 1 až 9684).
- **Item:** názov zakúpeného produktu (napr. bread, coffee, tea).
- **DateTime:** dátum a čas transakcie (napr. 30.10.2016 9:58).
- **Daypart:** časť dňa, kedy bola transakcia uskutočnená (napr. morning).
- **DayType:** označuje, či bola transakcia vykonaná počas víkendu alebo pracovných dní.

Na ilustráciu štruktúry a obsahu dátového súboru uvádzame Tab. 4.1 ako reprezentatívny príklad.

No.	Items	DateTime	Daypart	DayType
1	Bread	30. 10. 2016 9:58	Morning	Weekend
2	Scandinavian	30. 10. 2016 10:05	Morning	Weekend
2	Scandinavian	30. 10. 2016 10:05	Morning	Weekend
⋮	⋮	⋮	⋮	⋮
9683	Coffee	04. 09. 2017 14:57	Afternoon	Weekend
9683	Pastry	04. 09. 2017 14:57	Afternoon	Weekend
9684	Smoothies	04. 09. 2017 15:04	Afternoon	Weekend

Tabuľka 4.1: Ukážka použitého súboru dát

Z Tab. 4.1 vyplýva, že v rámci transakcie pod číslom 2 dňa 30. 10. 2016 (počas víkendu) ráno boli nakúpené dva kusy položky „Scandinavian“.

4.2 Apriori

V rámci tejto práce sme v programovacom jazyku Python implementovali algoritmus Apriori. Na uľahčenie práce s dátovou sadou a matematickými operáciami sme využili knižnice *pandas* a *numpy*. Vlastná implementácia metódy Apriori prijíma na vstupe údaje v tvare, kde v riadkoch máme jednotlivé transakcie a v stĺpcoch položky. Hodnota 1 alebo 0 v i-tom riadku a j-tom stĺpci tabuľky určuje, či bola položka j v rámci transakcie i nakúpená[1].

Pri analýze zohľadníme aj časový aspekt. Do výslednej tabuľky doplníme štyri stĺpce: Afternoon, Evening, Morning, Night. Výsledný dátový rámec, ktorý bude slúžiť ako vstup pre Apriori metódu, je uvedený v Tab. 4.2.

No.	Položka 1	...	Položka 94	Afternoon	Evening	Morning	Night
1	0	...	0	0	0	1	0
2	0	...	0	0	0	1	0
⋮	⋮	...	⋮	⋮	⋮	⋮	⋮
9683	0	...	0	0	1	0	0
9684	0	...	0	0	1	0	0

Tabuľka 4.2: Výsledná tabuľka po transformácii

[1]Pozorný čitateľ si všimne, že v niektorých transakciách boli niektoré položky zakúpené viackrát. Tento fakt pri analýze ignorujeme.

Experimentálne boli zvolené nasledovné parametre pre asociačné pravidlá:

- prahová hodnota podpory pravidla: min_sup = 0,03.
- prahová hodnota spoľahlivosti: min_conf = 0,5.

Totiž od pravidla $A \Rightarrow B$ požadujeme, aby sa $A \cup B$ vyskytlo v aspoň 3% transakcií a v každej transakcii s A sa B vyskytne s pravdepodobnosťou aspoň 50%.

Z dostupných online implementácií Apriori algoritmu, ako sú napríklad [20, 21] a [22], sme v rámci tejto práce zvolili realizáciu metódy z knižnice *mlxtend* [23].

4.3 FP-growth

Vzhľadom na rozsiahlosť a komplexnosť algoritmu sme sa v rámci tejto práce rozhodli pre jeho implementáciu pomocou knižnice *mlxtend* v programovacom jazyku Python [24].

Hodnoty parametrov prenecháme rovnaké ako pri predchádzajúcej metóde.

Kapitola 5

Výsledky

Táto kapitola prezentuje výsledky analýzy košov nákupu z kaviarne. Cieľom analýzy bolo zistiť, aké sú najčastejšie kupované položky a aké asociačné pravidlá medzi nimi existujú.

V Tab. 5.1 uvádzame zoznam frekventovaných množín nájdených realizáciou metódy Apriori [23] a FP-growth [24], ako aj našou vlastnou implementáciou Apriori. Zhodnosť výsledkov troch implementovaných metód indikuje korektnosť týchto riešení.

No.	**Itemsets**	**Support**
1	{Afternoon}	0, 537665
2	{Coffee}	0, 478394
3	{Morning}	0, 433492
4	{Bread}	0, 327205
⋮	⋮	⋮
40	{Hot chocolate, Afternoon}	0, 033069
41	{Afternoon, Soup}	0, 032647
42	{Afternoon, Cookies}	0, 031696
43	{Coffee, Afternoon, Tea}	0, 030745

Tabuľka 5.1: Obdržané frekventované množiny položiek z údajového rámca.

V ďalšom kroku sme pomocou nájdených frekventovaných množín vygenerovali asociačné pravidlá. Iteráciou cez všetky frekventované položky skúmame možné kombinácie pravidiel. Pravidlá, ktoré spĺňajú prahové hodnoty spoľahlivosti, boli pridané do výslednej tabuľky a následne usporiadané podľa hodnoty spoľahlivosti. V Tab. 5.2

uvádzame časť vygenerovaných asociačných pravidiel.

No.	Antecedents	Consequents	Support	Confidence
1	{Soup}	{Afternoon}	0,032647	0,947853
2	{Sandwich, Coffee}	{Afternoon}	0,033492	0,875691
⋮	⋮	⋮	⋮	⋮
12	{Medialuna}	{Coffee}	0,035182	0,569231
13	{Hot chocolate}	{Afternoon}	0,033069	0,567029
14	{Morning, Pastry}	{Coffee}	0,033492	0,554196
⋮	⋮	⋮	⋮	⋮
18	{Sandwich}	{Coffee}	0,038246	0,532352
19	{Cake}	{Coffee}	0,054727	0,526958
⋮	⋮	⋮	⋮	⋮
22	{Morning}	{Coffee}	0,223244	0,514989
23	{Bread}	{Afternoon}	0,164395	0,502422

Tabuľka 5.2: Získané asociačné pravidla so spoľahlivosťou viac ako 50 %.

Tieto pravidlá môže analytik interpretovať napríklad takto:

- Pravidlá 1, 2, 13: Počas popoludňajších hodín sa v kaviarni pozoruje zvýšený predaj polievok, koláčov, sendvičov a čaju. V reakcii na túto situáciu by majiteľ kaviarne mohol zvážiť zavedenie špeciálnych ponúk, napríklad kombinovanie čaju s koláčom alebo sendvičom za zvýhodnenú cenu.
- Pravidlá 14, 18, 19: Káva slúži ako sprievodný prostriedok pre rôzne jedlá. Túto silnú asociáciu môže majiteľ kaviarne využiť na stimuláciu predaja oboch produktov súčasne, napríklad formou zľavy na kávu pri nákupe jedla.

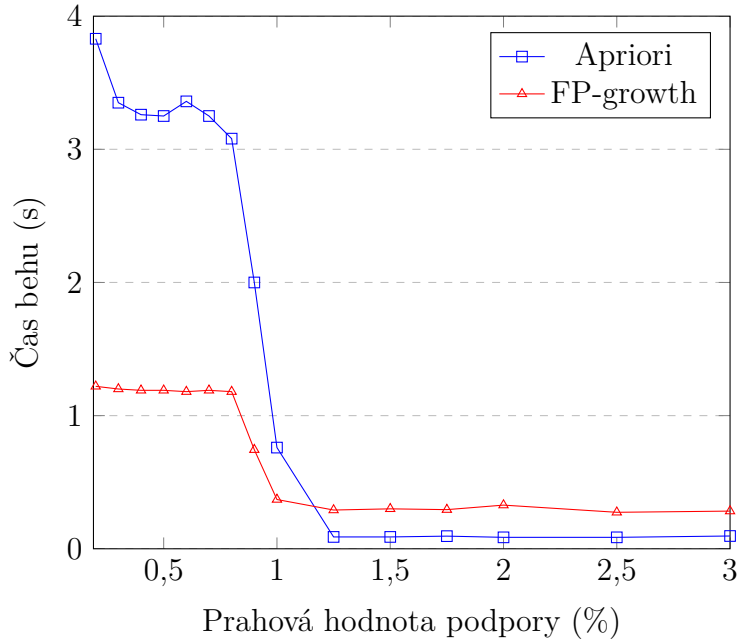
5.1 Efektívnosť

V tejto časti kapitoly sa zameriavame na analýzu výkonnosti dvoch implementovaných algoritmov. Cieľom je porovnať ich výkonnosť v závislosti od prahovej hodnoty podpory a veľkosti databázy. Získané výsledky nám pomôžu vybrať optimálny algoritmus pre konkrétnu úlohu.

Analýza sa bude opierať na experiment využívajúci syntetické údaje. Dáta boli generované tak, aby boli rovnomerne rozdelené a homogénne. To znamená, že frek-

vencia výskytu jednotlivých položiek a ich početnosť v transakciách je rovnaká bez ohľadu na rozmer databázy. Zistenia budú zobrazené v grafoch.

Obr. 5.4 ukazuje dĺžku behu algoritmov Apriori a FP-growth pri hľadaní asociačných pravidiel v závislosti od prahovej hodnoty podpory. Ako je vidno, dĺžka behu Apriori rastie exponenciálne s klesajúcou prahovou hodnotou podpory, zatiaľ čo dĺžka behu algoritmu FP-growth rastie pomalšie. Dôvodom je to, že s klesajúcou podporou dochádza k dramatickému nárastu počtu a dĺžky generovaných množín. To vedie k prudkému zväčšeniu množín kandidátov, ktoré Apriori musí spracovať.

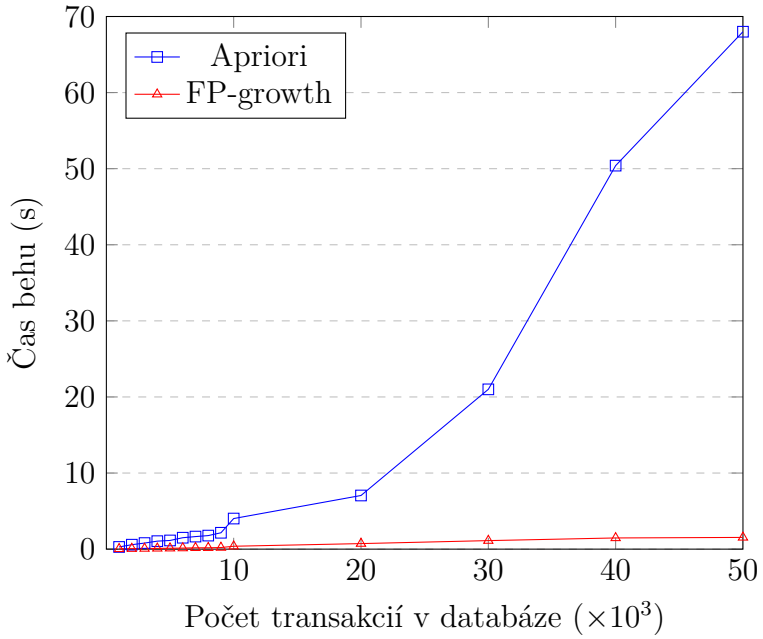


Obr. 5.4: Vplyv prahovej hodnoty podpory na čas behu Apriori a FP-Growth

Nesprávna voľba parametrov pri hľadaní asociačných pravidiel teda môže mať dva závažné dopady:

1. zvýšený počet pravidiel,
2. predĺžený výpočtový čas.

Obr. 5.5 znázorňuje, ako dĺžka behu algoritmu Apriori rastie exponenciálne s počtom transakcií v databáze, zatiaľ čo dĺžka behu algoritmu FP-growth rastie lineárne. To potvrdzuje, že pri spracovaní rozsiahlych, veľkých datasetov je FP-growth radovo efektívnejší v porovnaní s Apriori.



Obr. 5.5: Porovnanie času behu Apriori a FP-Growth.

Kapitola 6

Diskusia

V tejto kapitole zhodnotíme výsledky nášho výskumu a porovnáme ich s inými prácami v oblasti data miningu.

Pravdou je, že výsledky analýzy väčšinou obsahujú pravidlá, ktoré spájajú buď „coffee“ alebo „morning“. Tento výskyt možno pripísať významnosti týchto atribútov v rámci súboru údajov.

Potenciálne je možné skúsiť tieto časté atribúty zo súboru údajov vymazať a následne sa pokúsiť v týchto dátach hľadať asociácie. Týmto spôsobom sa zbavíme najčastejších atribútov.

Vykonaná analýza odhalila niekoľko možností pre ďalší výskum. Jednou z nich je skúmanie sezónnych faktorov a ich vplyvu na nákupné správanie spotrebiteľov. Je možné, že nákupné správanie zákazníkov sa v priebehu roka menia. Takto by mohli byť identifikované sezónne vzorce a podľa nich by mohla byť prispôsobená ponuka produktov kaviarne.

Ďalšou oblasťou záujmu je analýza účinnosti marketingových kampaní. Vplyv týchto stratégií na nákupné správanie spotrebiteľov môžeme posúdiť porovnaním údajov pred ich nasadení a po ňom. To umožní analytikom určiť, ktoré kampane boli najúčinnejšie a z akých dôvodov.

Podobné výsledky týkajúce sa časovej zložitosti algoritmov boli popísané v [25]. Autori došli ku analogickému záveru a pozorovali zhodný rast zložitosti algoritmov v závislosti od hodnoty podpory.

V [26] autori úspešne aplikovali metódu Apriori na analýzu nákupného koša maloobchodnej predajni s cieľom analyzovať správanie zákazníkov. Dátový súbor sa z hľadiska veľkosti a obsahu podobal na ten, ktorý sme použili v kapitole 4.

Rovnako ako v našej bakalárskej práci, v [27] bola pri analýze použitá metóda

FP-growth, ktorá preukázala svoju účinnosť. Úspešne našli cenné asociácie medzi produktmi. Tieto zistenia sú nápomocné pre obchody, ktoré sa snažia zlepšiť efektívnosť a zvýšiť ziskovosť.

Dôležite je zdôrazniť, že v prípade databáz s veľkým počtom atribútov má zmysel vykonať **kategorizáciu** alebo **hierarchiu** atribútov. V praxi sa stretávame s obrovským množstvom produktov, ktoré sa dajú zoskupiť do kategórií. V takomto prípade postačí uvažovať všeobecnejšie pravidlá, ako napríklad pečivo $\Rightarrow$ limonády. Pod pojmom *pečivo* by sme v tomto kontexte rozumeli viacero druhov pečiva.

Takéto vylepšenia by mohli zvýšiť výkon a súčasne objaviť všeobecnejšie asociácie. Ich implementácia však presahuje rámec tejto práce.

Záver

Úloha hľadania asociačných pravidiel sa často rieši pomocou dvoch populárnych algoritmov, napríklad Apriori a FP-growth. Algoritmus Apriori je pri spracovaní rozsiahlych databáz neefektívny, pretože vyžaduje viacnásobný prechod databázou, pričom algoritmus FP-growth tento nedostatok odstraňuje. V našej práci sme analyzovali aj ďalšie metódy asociačných pravidiel, ktoré môžu mať oproti týmto populárnym metódam ďalšie výhody. Príkladom takejto metódy sú GUHA asociačné pravidlá, ktoré majú v porovnaní s klasickými metódami širšie možnosti: nie sú obmedzené iba na hľadanie asociačných pravidiel v tvare implikácií, ale dokážu odhaliť aj širšie súvislosti v dátach, napríklad pomocou ekvivalencie alebo iných štatistických ukazovateľov, ktoré popisujeme v tejto práci.

V praktickej časti práce sme porovnávali časovú náročnosť metódy Apriori a FP-growth v závislosti od veľkosti databázy. Ako sme aj očakávali z teoretických predpokladov, metóda FP-growth bola značne rýchlejšia ako Apriori. Ukázali sme aj vplyv prahovej hodnoty podpory na čas behu týchto algoritmov. Ak tento parameter je príliš malý, prístup generovania a testovania kandidátnych množín je neefektívny – FP-growth je potom efektívnejší ako Apriori algoritmus. Napriek tomu oba prístupy majú rozsiahle praktické uplatnenie.

Spomenuté metódy boli aplikované na údaje z nákupov pekárne. Tato aplikácia na reálnych údajoch ukázala ich praktickú hodnotu v oblasti analýzy dát. Vygenerovali sme 23 asociačných pravidiel spolu s hodnotami, ktoré popisujú ich silu a význam. Následne sme tieto pravidlá interpretovali a ukázali, ako môžu byť použité v praxi.

Získané výsledky demonštrujú, že metódy asociačných pravidiel môžu byť užitočné pri analýze nákupného košíka, ako aj v iných oblastiach.

Zoznam použitej literatúry

[1] Antoni, Ľubomír et al, 2020. *Dátová veda a jej aplikácie*. Košice: Univerzita Pavla Jozefa Šafárika v Košiciach. ISBN 978-80-8152-917-7.

[2] McCormick, Tyler, Cynthia Rudin a David Madigan, 2011. A Hierarchical Model for Association Rule Mining of Sequential Events: An Approach to Automated Medical Symptom Prediction. In: *Annals of Applied Statistics*. doi: 10.2139/ssrn.1736062.

[3] Flach, Peter et al, 2003. On the Road to Knowledge. In: *Data Mining and Decision Support: Integration and Collaboration*. Boston: Springer US, s. 143–155. ISBN 978-1-4615-0286-9. doi: 10.1007/978-1-4615-0286-9_12.

[4] Agrawal, Rakesh, Tomasz Imieliński a Arun Swami, 1993. Mining Association Rules between Sets of Items in Large Databases. In: *SIGMOD Rec.*. Vol. 22, no. 2, s. 207–216. ISSN 0163-5808. doi: 10.1145/170036.170072.

[5] Agrawal, Rakesh a Ramakrishnan Srikant, 1994. Fast Algorithms for Mining Association Rules in Large Databases. In: *Proceedings of the 20th International Conference on Very Large Data Bases*. San Francisco: Morgan Kaufmann Publishers Inc., s. 487–499. ISBN 1558601538.

[6] Hájek, Petr, Ivan M. Havel a Metodej K. Chytil, 1966. The GUHA method of automatic hypotheses determination. In: *Computing*. Vol. 1, s. 293–308. Dostupné na: https://api.semanticscholar.org/CorpusID:10511114.

[7] Zaki, M.J., 2000. Scalable algorithms for association mining. In: *IEEE Transactions on Knowledge and Data Engineering*. Vol. 12, no. 3, s. 372–390. doi: 10.1109/69.846291.

[8] Han, Jiawei et al, 2004. Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. In: *Data Min. Knowl. Discov.*. Vol. 8, s. 53–87. doi: 10.1023/B:DAMI.0000005258.31418.83.

[9] Delgado, M. et al, 2005. Mining Fuzzy Association Rules: An Overview. In: *Soft Computing for Information Processing and Analysis*. Berlin: Springer Berlin Heidelberg, s. 351–373. ISBN 978-3-540-32365-5. doi: 10.1007/3-540-32365-1_15.

[10] Ashrafi, Mafruz, David Taniar a Kate Smith-Miles, 2004. A New Approach of Eliminating Redundant Association Rules. In: *15th International Conference, Springer, LNCS*. S. 465–474. ISBN 978-3-540-22936-0. doi: 10.1007/978-3-540-30075-5_45.

[11] Ashrafi, Mafruz Zaman, David Taniar a Kate Smith, 2007. Redundant association rules reduction techniques. In: *International Journal of Business Intelligence and Data Mining*. Vol. 2, no. 1, s. 29–63. doi: 10.1504/IJBIDM.2007.012945. Dostupné na: https://www.inderscienceonline.com/doi/abs/10.1504/IJBIDM.2007.012945.

[12] Sokol, Pavol et al, 2023. Formal concept analysis approach to understand digital evidence relationships. In: *International Journal of Approximate Reasoning*. Vol. 159. ISSN 0888-613X. doi: https://doi.org/10.1016/j.ijar.2023.108940.

[13] Bian, Pan et al, 2018. NAR-miner: discovering negative association rules from code for bug detection. In: *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. New York: Association for Computing Machinery, s. 411–422. ISBN 9781450355735. doi: 10.1145/3236024.3236032.

[14] Gang Li, Jia Rong a Huy Quan Vu, 2010. Incorporating Both Positive and Negative Association Rules into the Analysis of Outbound Tourism in Hong Kong. In: *Journal of Travel & Tourism Marketing*. Vol. 27, no. 8, s. 812–828. doi: 10.1080/10548408.2010.527248.

[15] Hájek, Petr, Ivan Havel a Metoděj Chytil, 1967. GUHA - metoda systematického vyhledávání hypotéz. II. In: *Kybernetika*. Vol. 03, no. 5, s. 430–437. Dostupné na: http://eudml.org/doc/27520.

[16] Havránek, Tomáš, 1971. The statistical interpretation and modification of GUHA method. In: *Kybernetika*. Vol. 07, no. 1, s. 13–21. Dostupné na: http://eudml.org/doc/27910.

[17] HÁJEK, Petr, 1968. Problém obecného pojetí metody GUHA. In: *Kybernetika*. Vol. 04, no. 6, s. 505–515. Dostupné na: http://eudml.org/doc/28614.

[18] HAVRÁNEK, Tomáš, 1975. Statistical quantifiers in observational calculi: An application in GUHA-methods. In: *Theory and Decision*. Vol. 6, no. 2, s. 213–230. ISSN 1573-7187. doi: 10.1007/BF00169108.

[19] KUILA, Akashdeep, 2021. *Bakery Sales Dataset* [online]. Dáta vo formáte CSV. Kaggle. Dostupné na: https://www.kaggle.com/datasets/akashdeepkuila/bakery. [cit. 2024. 4. 28].

[20] SAINI, Abhinav, 2013. *Apriori* [online]. GitHub. Dostupné na: https://github.com/asaini/Apriori. [cit. 2024. 4. 25].

[21] MOCHIZUKI, Yu, 2016. *Apyori* [online]. GitHub. Dostupné na: https://github.com/ymoch/apyori. [cit. 2024. 4. 25].

[22] ANALYST, A Data, 2021. *Apriori Algorithm (python 3.0)* [online]. Dostupné na: https://adataanalyst.com/machine-learning/apriori-algorithm-python-3-0/. [cit. 2024. 4. 25].

[23] RASCHKA, Sebastian. *apriori: Frequent itemsets via the Apriori algorithm* [online]. Dostupné na: https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/apriori/. [cit. 2024. 4. 25].

[24] RASCHKA, Sebastian. *fpgrowth: Frequent itemsets via the FP-growth algorithm* [online]. Dostupné na: https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/fpgrowth/. [cit. 2024. 4. 25].

[25] FAGEERI, Sallam Osman, Mohammad Abu KAUSAR a Arockiasamy SOOSAIMANICKAM, 2023. MBA: Market Basket Analysis Using Frequent Pattern Mining Techniques. In: *International Journal on Recent and Innovation Trends in Computing and Communication*. Vol. 11, s. 15–21. ISSN 23218169. doi: 10.17762/ijritcc.v11i5s.6591.

[26] LEKIREDDY, Bharadhwaj Reddy et al, 2023. Market-Based Analysis: Apriori approach to analyze purchase patterns. In: *EAI Endorsed Transactions on Scalable Information Systems*. Vol. 10, no. 5, s. 1–5. doi: 10.4108/eetsis.3355.

[27] PING, Haoyang et al, 2024. Optimization of Vegetable Restocking and Pricing Strategies for Innovating Supermarket Operations Utilizing a Combination of ARIMA, LSTM, and FP-Growth Algorithms. In: *Mathematics*. Vol. 12, no. 7. doi: 10.3390/math12071054.