Катедра „Компютърни науки и технологии“

КУРСОВА РАБОТА

Тема:

Априори алгоритъм

Изготвил: Татяна Христова

Специалност: Софтуерни и интернет технологии

Дисциплина: Изкуствен интелект

Факултетен номер: 19621602

ТУ-Варна, 2022 г.

Ръководител: ас. М. Марков

Съдържание

1. Увод

Често срещана задача в машинното обучение е търсенето на зависимости между множества от елементи. Елементите може да са обекти или събития. След като се намери зависимост между тях, се извеждат правила, които се използват за различни цели, например за изследване на потребителско поведение, пазарни анализи, финансови анализи или в здравеопазването за намиране на сходни лекарствени реакции при пациенти. Пазарният анализ позволява на търговците на дребно да идентифицират връзките между артикулите, които хората купуват. По този начин мениджърите могат да обобщят в определени групи артикулите, закупени заедно и да използват данните за коригиране оформлението на магазините, за промоции или кръстосани продажби. Финансовият анализ помага да се покаже как различни акции са в тенденция заедно.

За решаване на подобни задачи се използват алгоритми за извличане на асоциативни правила - Apriori алгоритъм, FP-Growth алгоритъм, Eclat алгоритъм и други.

2. Асоциативни правила

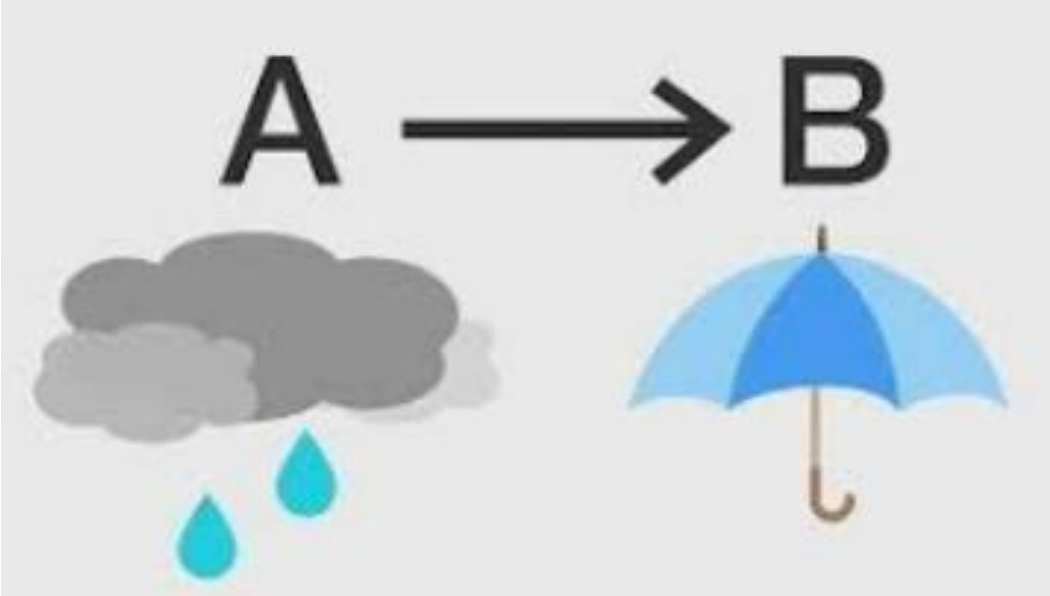
Асоциативните правила са един от основните подходи в извличането на знания от данни. Обучението с тях представлява метод за машинно самообучение, базирано на правила. Използва се откриване на връзки между променливи в големи бази от данни. Предназначено е да открива закономерности с помощна на някаква мярка за интересност. Анализът на допълнителните данни води до генериране на нови правила. Крайната цел е да се помогне на машината да наподоби способността на човешкия мозък за извличане на признаци и абстрактни асоциации.

Асоциативните правила водят началото си от анализите на „пазарската кошница“. Складираните данни се претърсват за незабележими на пръв поглед

зависимости в множества от елементи, представени под формата на транзакции, които най-често се съхраняват в релационни бази данни.

Правилото за асоциативност се състои от две части – антецедент и консеквент. Антецедентът (if) е елемент или множество от елементи, намерени в базата данни. Консеквентът (then) е елемент или множество от елементи, които са намерени в комбинация с антецетента.

if A *then* B (A ->B)



Изображение: Асоциативност A -> B

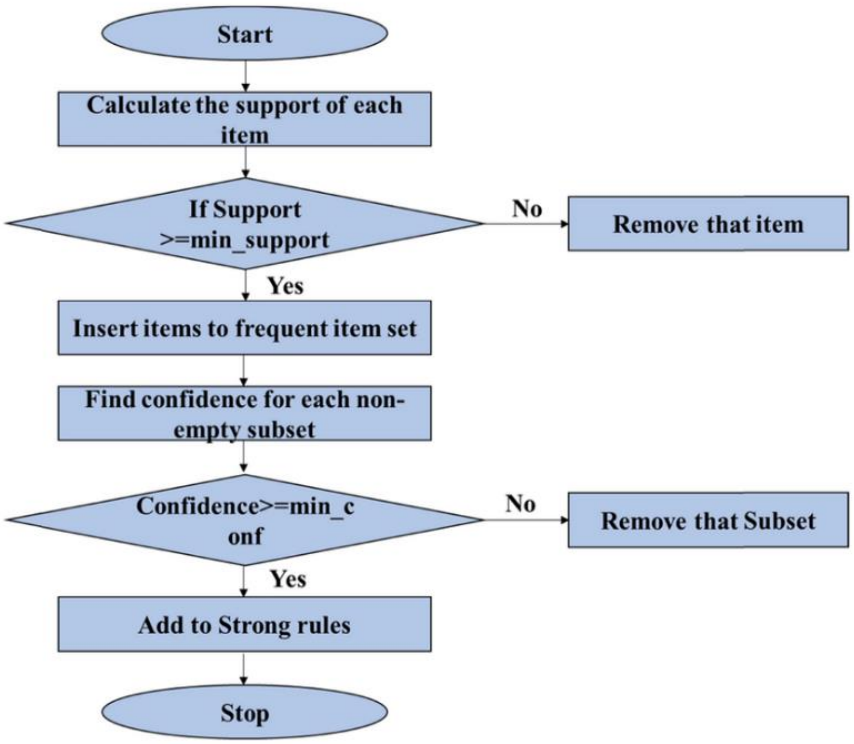
3. Априори алгоритъм – Същност

Основен алгоритъм за генериране на асоциативни правила и извличане на чести елементи е Априори алгоритъмът. При него се приема, че ако едно множество е често, то и неговите подмножества ще бъдат чести. Благодарение на това си свойство, се редуцира обемът от транзакции за претърсване.

Априори алгоритъмът е предложен от R. Agrawal и R. Srikant през 1994 година, с цел намиране на чести набори от елементи в набор от данни за правилото на булевата асоциация. Името на алгоритъма е избрано, защото използва предварителни познание за честите срещания на набор от елементи.

Apriori е алгоритъм, позволяващ откриване на обекти, които често се срещат заедно. Това става на базата на предварително зададен праг (threshold) за честота на срещане. Apriori генерира подмножества, които могат да бъдат огромен брой дори ако продуктите в извадката са малко. Първо алгоритъмът изчислява честотата на срещане на индивидуалните продукти и премахва тези, при които тя е под зададения праг, след това продължава с комбинации от 2 продукта, после с 3 и т.н. докато не се отсеят всички излишни подмножества.

Apriori е проектиран да работи с бази данни, съдържащи транзакции. Всяка транзакция се разглежда като набор от елементи (*itemset)*. Apriori използва подход "отдолу нагоре", при който честите подмножества се разширяват с един елемент в даден момент (стъпка, известна като генериране на кандидати), а групи от кандидати се тестват спрямо данните. Алгоритъмът се прекратява, когато не бъдат намерени други успешни разширения.



Изображение: Блок-схема на Априори алгоритъм

3.1 Често срещани набори от артикули (Frequent Itemset)

Честите набори от артикули са тези елементи, чиято поддръжка е по-голяма от праговата стойност или зададената от потребителя минимална поддръжка. Това означава, че ако A & B са честите набори от елементи заедно, тогава поотделно A и B също трябва да бъдат честите елементи.

Да предположим, че има две транзакции: A = {1,2,3,4,5} и B = {2,3,7}, в тези две транзакции, 2 и 3 са честите елементи.

3.2 Стъпки за извличане на данни

В таблица 1 са показани 5 транзакции, в които участват различни хранителни продукти:

Показване на 10 резултата Търсене във всички колони:

ТР	хляб (Х)	месо (М)	бира (Б)	вода (В)	яйца (Я)
ТР1	1	1	1	0	0
ТР2	1	0	1	0	1
ТР3	0	1	1	1	0
ТР4	1	1	1	0	0
ТР5	1	1	0	1	0

Изображение: Таблица 1

Ако приемем, че сме задали като минимален праг 0.3, алгоритъмът ще премахне тези продукти, които се срещат в по-малко от 30% от всички транзакции.

Стъпка 1 – преглед на честотата на срещане на индивидуалните продукти.

Показване на 10 резултата Търсене във всички колони:

Х	М	Б	В	*Я*
0.8	0.8	0.8	0.4	0.2

Изображение: Таблица 2

Яйцата попадат под праговата стойност и съответно се премахват от по-нататъшните разглеждания. Apriori работи на принципа, че ако един продукт е рядко срещан, то комбинациите с него също са.

Стъпка 2 – преглед на комбинациите от 2 продукта

Показване на 10 резултата Търсене във всички колони:

Х-М	Х-Б	*Х-В*	М-Б	М-В	*Б-В*
0.6	0.6	0.2	0.6	0.4	0.2

Изображение: Таблица 3

Две от комбинациите (хляб-вода и бира-вода) са под зададения праг и се премахват.

Стъпка 3 – преглед на комбинации от 3 продукта

Показване на 10 резултата Търсене във всички колони:

Х-М-Б
0.4

Изображение: Таблица 4

Остава само една възможна комбинация, която е над зададения праг, след което алгоритъмът приключва работа.

3.3 Предимства на Априори алгоритъм

1. Лесен за разбиране алгоритъм.
2. Стъпките за реализирането му са логични и лесни за изпълнение.

3.4 Недостатъци на Априори алгоритъм

1. Работи бавно в сравнение с други алгоритми.
2. Сканира цялата база данни многократно, което намалява производителността.

3.5 Критерии за оценка

Чрез различни метрики за оценка, можем да определим доколко са адекватни направените асоциации.

- Поддръжка (support, coverage) – какъв процент от анализираните случаи съдържат определено множество от елементи

$$supp(A \to T) = P(A \cup T)$$

Изображение: Формула за поддръжка

- Достоверност (confidence, accuracy) – вероятността от частта *списък от елементи* да следва частта *следствие*.

$$confidence(A \to T) = \frac{P(A \cup T)}{P(A)} = \frac{supp(A \to T)}{supp(A)}$$

Изображение: Формула за достоверност

- Подемна сила (lift) – показва дали има зависимост между определени обекти или те присъстват заедно по случайност

$$lift(A \to T) = \frac{conf(A \to T)}{supp(T)} = \frac{supp(A \to T)}{supp(A) \times supp(T)}$$

Изображение: Формула за подемна сила

- Ако **lift > 1** има положителна зависимост, шансът обектите да се срещнат заедно, е по-висок.
- Ако **lift < 1** има отрицателна зависимост, шансът обектите да се срещнат отделно е по-висок.
- Ако **lift = 1** наличният списъка от елементи няма влияние върху следствието.

- Влияние, натиск (leverage) – колко по-вероятно е определени обекти да се срещат заедно отколкото поотделно.

$$leverage(A \to C) = P(X \cap Y) - P(X) \times P(Y) = support(A \to C) - support(A) \times support(C)$$

Изображение: Формула за влиянието

Например, ако имаме продажби на хранителни стоки, целта е да разберем с колко повече конкретни продукти се закупуват заедно отколкото индивидуално.

- Убеденост (conviction) – до каква степен асоциацията между списъка от елементи и следствието е неправилно

$$conviction(A \to C) = \frac{1 - support(C)}{1 - confidence(A \to C)} = \frac{P(X) \times P(\overline{Y})}{P(X \cap \overline{Y})}$$

Изображение: Формула за убеденост

Изчислява се съотношението между пропорцията от транзакции, в които не присъства следствието и вероятността асоциацията да е неправилна.

Например ако получим като стойност за тази метрика 1.4, тогава асоциацията ще е неправилна 40% по-често отколкото ако тя се дължи на случайност (когато метриката е равна на 1).

3.6 Област на приложение

Освен в търговията, Априори алгоритъмът се прилага и:

1. **В областта на образованието:** Извличане на правила за асоцииране при извличане на данни от приети студенти чрез характеристики и специалности.
2. **В областта на медицината:** Например Анализ на базата данни на пациента.
3. **В горското стопанство:** Анализ на вероятността и интензивността на горските пожари с данните за горските пожари.

Apriori се използва от много компании като Amazon в системата за препоръки и от Google за функцията за автоматично довършване.

4. Практическа задача

Задачата съдържа данни за продажби на хранителни стоки (9835 транзакции за 30 дневен период). Трябва да намерим кои продукти с кои други най-често се купуват. За решението ще се използва библиотека Mlxtend на Python. Тя съдържа функция apriori (), с която ще генерираме често срещани подмножества от елементи и функция association_rules (), която използва получените данни и генерира асоциативни правила.

Пет случайни реда от извадката изглеждат по следния начин:

Показване на 10 резултата Търсене във всички колони:

	Items
3859	pip fruit,canned beer
9774	white bread
3725	chicken,beef,yogurt,rolls/buns,newspapers
9747	rolls/buns,pastry,soda
2191	chicken,hamburger meat,other vegetables

Изображение: Пет случайни реда от извадката на задачата

- **Откриване на броя и честотата на закупуване на всеки продукт:**

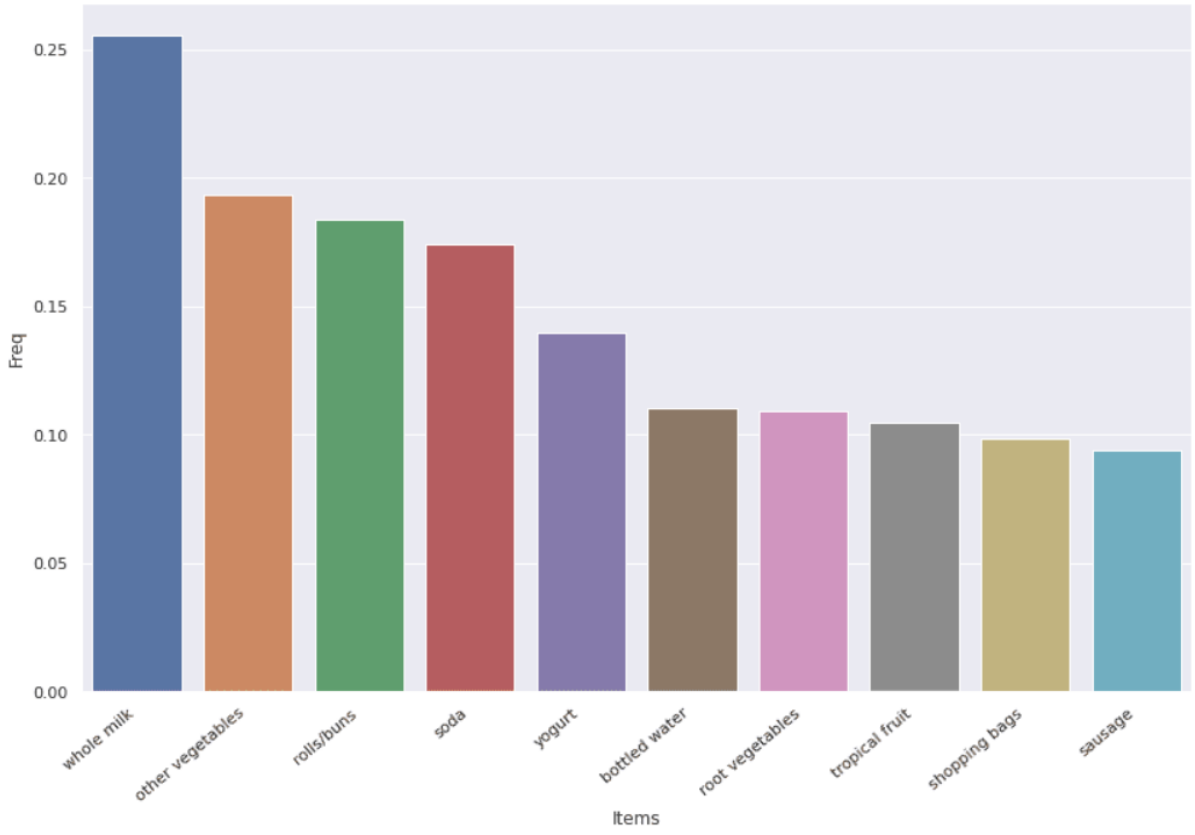
```
# Разделяне на продуктите поотделно
items = df['Items'].str.findall('[^,]+').sum()

# Запазване на продуктите в нов DataFrame
df_items = pd.DataFrame(items, columns=['Items'])

# Откриване на броя закупени продукти
df_items = df_items.value_counts().to_frame('Count')

# Нулиране на индекса
df_items.reset_index(inplace=True)

# Откриване на честотата на закупуване на продуктите
df_items['Freq'] = df_items['Count']/df.shape[0]
```

Изображение: Код за откриване на броя и честотата на закупените продукти

Показване на 10 резултата Търсене във всички колони:

	Items	Count	Freq
0	whole milk	2513	0.255516
1	other vegetables	1903	0.193493
2	rolls/buns	1809	0.183935
3	soda	1715	0.174377
4	yogurt	1372	0.139502

Изображение: Петте продукта, закупувани най-често



Изображение: Стълбовидна диаграма с десетте най-често срещани продукти

На сълбовидната диаграма са представени 10-те най-често купувани продукти, като те са подредени в низходящ ред според честотата на закупуването им. На първо място е пълномасленото мляко, което присъства в 26% от всички транзакции, а на десето е наденицата, закупувана в 9% от всички случаи.

- **Използване на алгоритъма Apriori и генериране на асоциативни правила**

Необходимо е първо да обработим данните, за да можем след това да ги подадем като параметър на функцията apriori ().

```
# Разделяне на продуктите в отделни колони
splitted = df['Items'].str.split(',', expand=True)

# Създаване на списък с транзакциите
records = []
for i in range(splitted.shape[0]):
    records.append([str(splitted.values[i,j]) for j in range(splitted.shape[1])])

# Използване на TransactionEncoder() и създаване на подходящият DataFrame за функцията
te = TransactionEncoder()
te_ary = te.fit(records).transform(records)
sparse_df = pd.DataFrame(te_ary, columns=te.columns_)
sparse_df.drop('None', axis=1, inplace=True)
```

Изображение: Код за разделяне на продуктите в колони, създаване на списък с транзакции и използване на фукцията TransactionEncoder()

Нужно е да създадем отделна колона за всеки продукт. В редовете, където присъства продукт, се поставя стойност 1, а там където не присъства – 0.

На изображение *Таблица_Продукти5* се вижда нагледно как изглеждат 5 случайни реда за двата най-продавани продукта – пълномаслено мляко и други зеленчуци.

Показване на 10 резултата Търсене във всички колони:

	whole milk	other vegetables
6704	1	0
3042	0	0
8862	1	1
902	1	0
6773	0	1

Изображение: Таблица_Продукти5

Цялата таблица след обработката е с 9835 реда и 169 колони. Преди да я подадем на apriori (), трябва да изчислим минимален праг.

Нека да приемем, че ни интересуват само онези продукти, които се продават по 5 пъти на ден и тъй като данните ни са за 30 дни, от тук можем да изчислим като праг 5 * 30 / 9835 = 0.015.

```
# Прилагане на алгоритъма Apriori
df_apriori = apriori(sparse_df, min_support=0.015, use_colnames=True, verbose=1)

# Откриване на асоциативни правила чрез функцията association_rules()
rules = association_rules(df_apriori, metric=support, min_threshold=0.015)
```

Изображение: Прилагане на Априори Алгоритъм

Алгоритъмът е генерирал 238 на брой правила.

Показване на 10 резултата Търсене във всички колони:

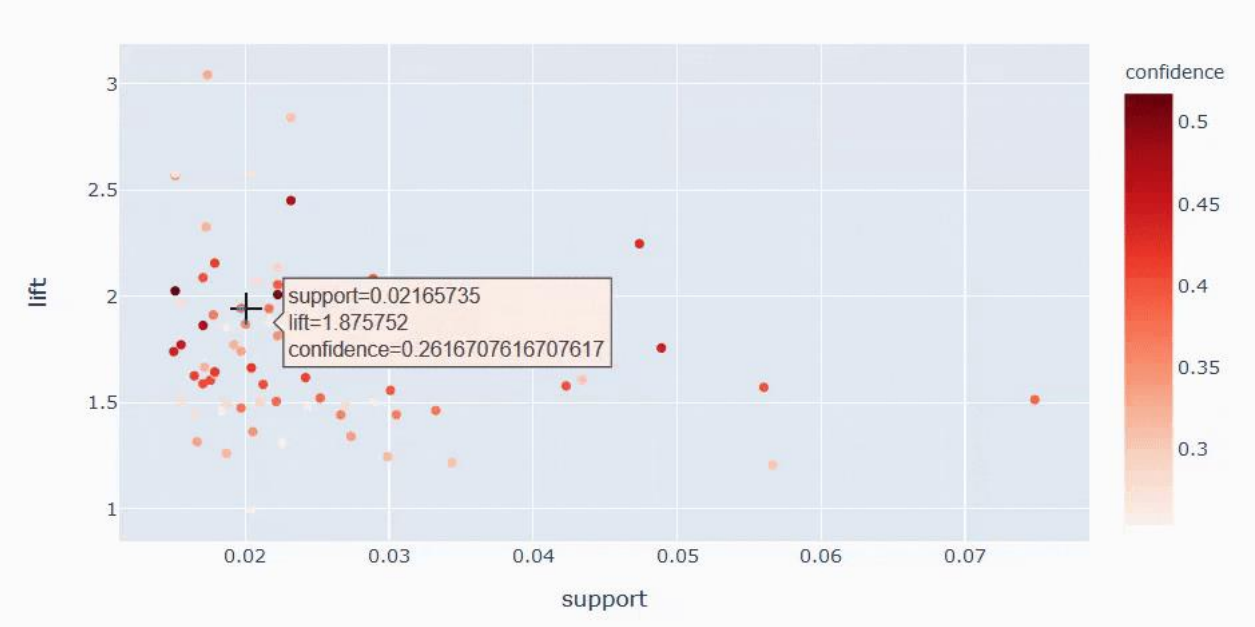
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conv
180	yogurt	shopping bags	0.139502	0.0985257	0.0152517	0.109329	1.10965	0.00150715	1.012
27	yogurt	bottled water	0.139502	0.110524	0.0229792	0.164723	1.49039	0.00756091	1.064
107	pork	other vegetables	0.0576512	0.193493	0.0216573	0.375661	1.94148	0.0105023	1.291
203	whole milk, rolls/buns	other vegetables	0.0566345	0.193493	0.0178953	0.315978	1.63303	0.00693692	1.179
127	pastry	rolls/buns	0.088968	0.183935	0.0209456	0.235429	1.27996	0.00458129	1.067

Изображение: Таблица след прилагане на Априори алгоритъм

Таблицата представя 5 случайно избрани правила, както и различните метрики за оценка. Например на ред 203 в 18% от случаите пълномаслено мляко и хлебчета са закупувани заедно, а в 32% от всички транзакции това е довело до закупуване на други зеленчуци. Стойността на подемната сила е 1.63, т.е. има положителна зависимост между продуктите за това правило и шанса те да се срещат заедно е по-висок.

- **Анализ на получените резултати**

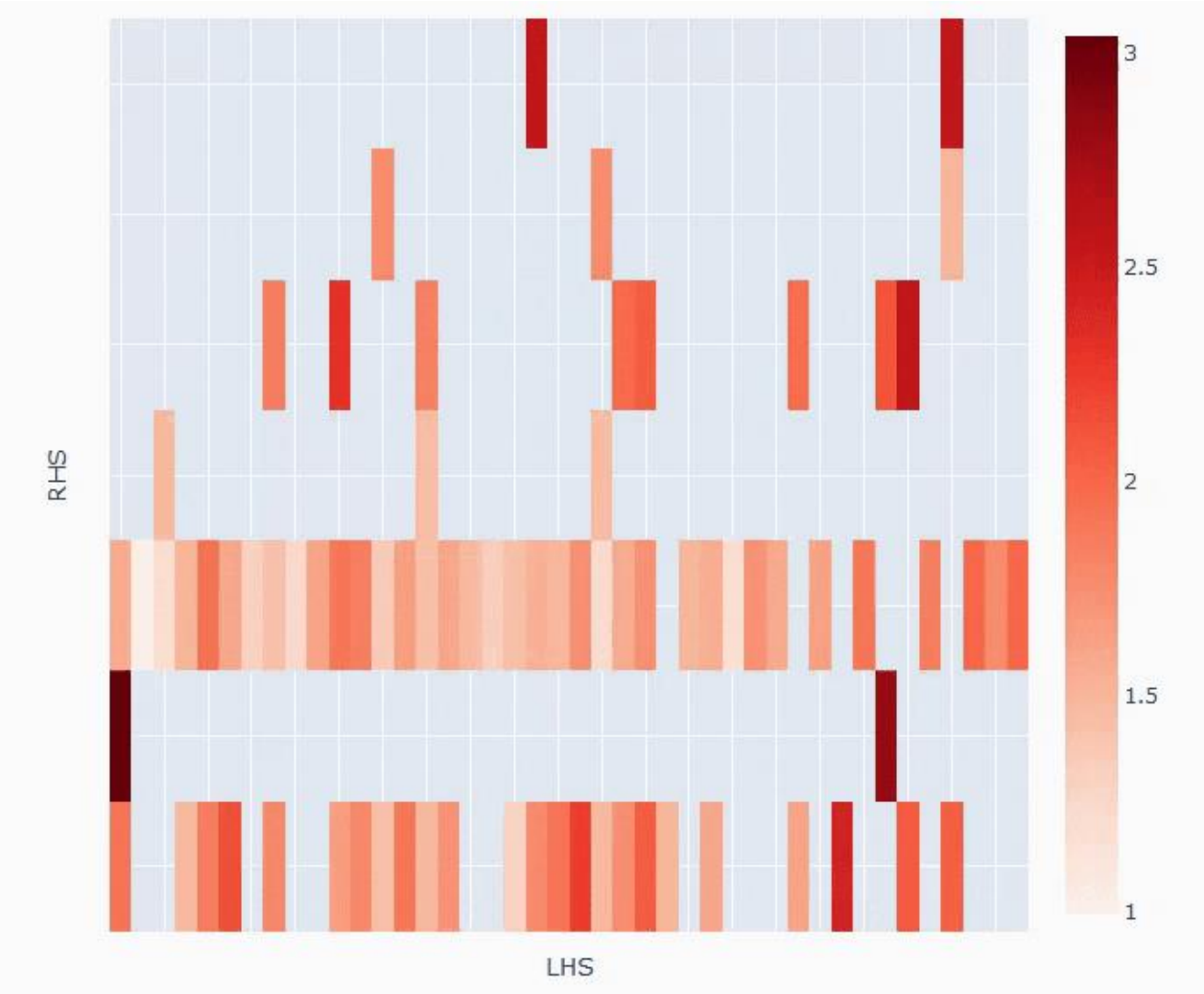
След като имаме генерирани асоциативни правила и стойности за отделните метрики за оценка, можем с помощта на различни визуализации да представим нагледно тези, които са по-важни.



Диаграма на разсейването

На визуализацията се вижда връзката между метриките поддръжка и подемна сила за данни, при които достоверността >= 0.25, като цветовете на точките се определят в зависимост от стойността на достоверността. Колкото по-висока е тя, толкова по-червен е цветът.

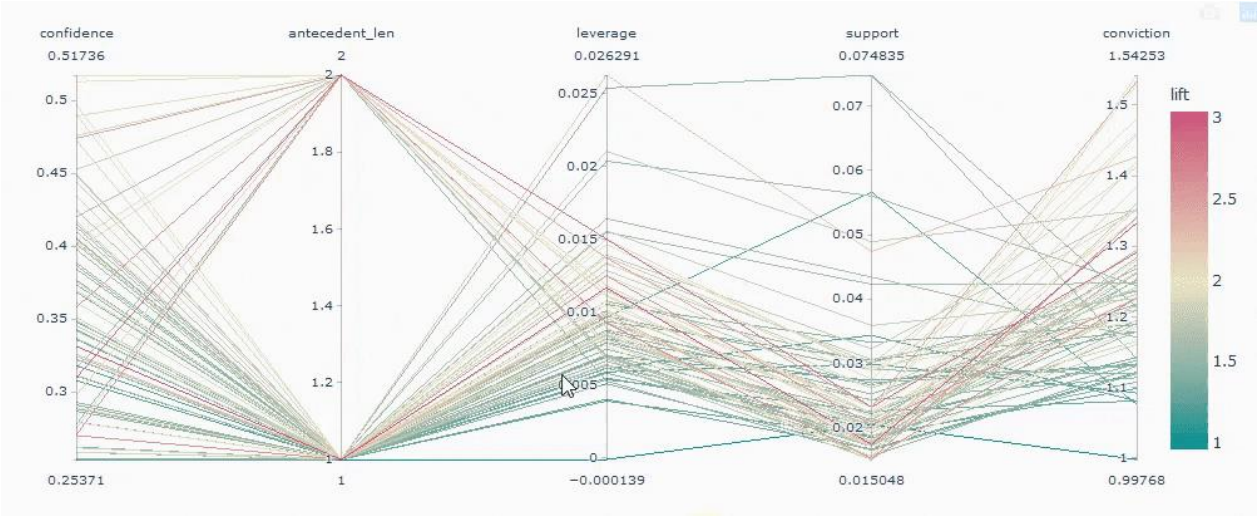
На следващата графика е връзката между продуктите в колоната със списъка от елементи и тази със следствията за филтрираните данни. Силата на връзката ще се определя от метриката подемна сила.



Визуализация с приложени правилата за подемна сила

На визуализацията се вижда, че някои от правилата имат подемна сила със стойности над 2.5. Такива са например {whole milk, other vegetables} -> {root vegetables} и {beef} -> {root vegetables}, което означава, че тези продукти много често се закупуват заедно.

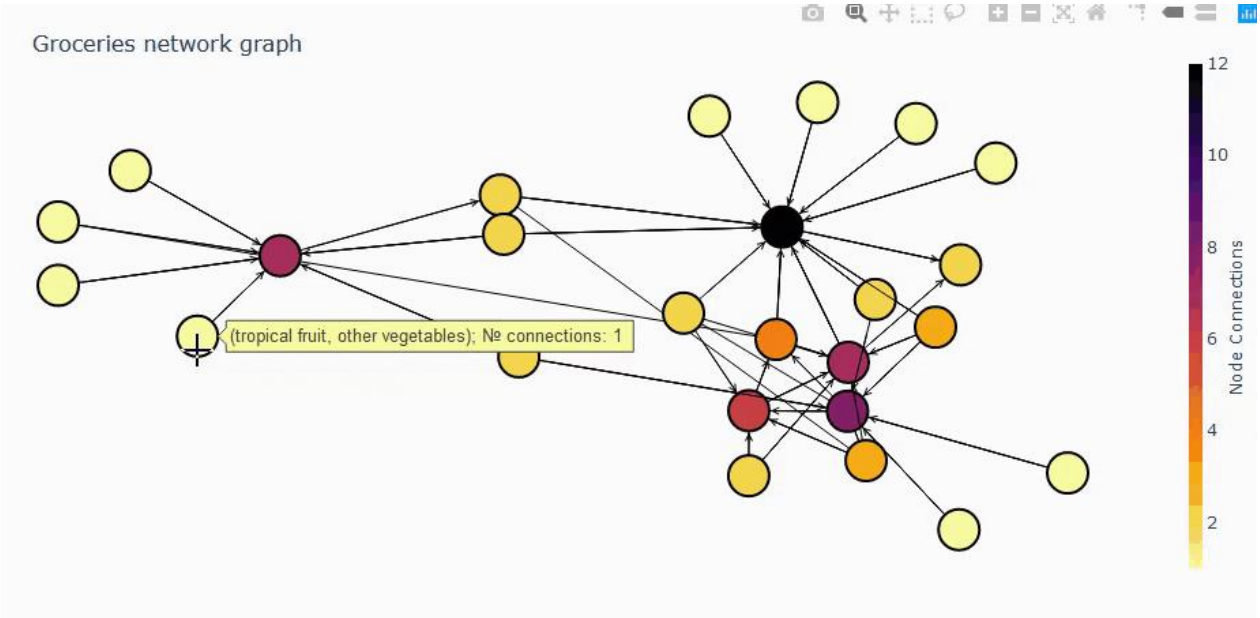
Можем да видим разликите в стойностите за отделните характеристики със следващата визуализация – паралелна диаграма.



Паралелна диаграма

При паралелната диаграма виждаме за всеки един ред какви са стойностите на конкретни характеристики. Можем да проследим как се изменят те и да правим сравнения между тях. В примера отново използваме данните с достоверност >= 0.25, като прави впечатление това, че правилата, които имат висока подемна сила, предимно съдържат 2 продукта в колоната със списъка от продукти.

За следващата визуализация ще използваме само данни, където стойностите в колоната подемна сила са >= 1.8.



Визуализация на данните с подемна сила >=18

На графиката се виждат различните продукти и броят връзки между тях. С най-голям брой връзки са други зеленчуци – 12. Също с голям брой връзки са пълномасленото мляко и йогуртът – 7 и 8.

- **Извод от практическата задача:**

Можем да кажем, че най-често след като се закупува пълномаслено мляко и кореноплодни зеленчуци, се купуват и други зеленчуци. Също покупките на йогурт, масло или извара най-често ще доведе до покупка на пълномаслено мляко.

На базата на получените резултати можем да определим кои продукти да сложим близо едни до други или ако имаме система за препоръки, да се правят предложения за продукти, които клиентите биха харесали.

5. Заключение

Априори алгоритъмът е един от редицата алгоритми, използващи подход „отдолу-нагоре“ за постепенно контрастиране на сложни записи. Той е полезен в днешните проекти за сложно машинно обучение и изкуствен интелект.

Алгоритъмът Априори може да се използва заедно с други алгоритми за ефективно сортиране и контрастиране на данни, за да покаже много по-добра картина за това как сложните системи отразяват моделите и тенденциите.

6. Източници

Apriori Algorithm in Machine Learning - Javatpoint

Apriori algorithm - Wikipedia

Apriori Algorithm - GeeksforGeeks

Apriori Algorithm in Data Mining: Implementation with Examples (softwaretestinghelp.com)

How to solve the Apriori algorithm in a simple way from scratch? | by Melanie Group | Level Up Coding (gitconnected.com)

Machine Learning: Кой Melanie с Melanie други се Melanie? (expert-bg.org)

What is the apriorism algorithm? - definition from tehomedia - development 2022 (theastrologypage.com)

НАУЧНО-ПРИЛОЖНА КОНФЕРЕНЦИЯ (conference-burgas.com)

Top 20 AI and Machine Learning Algorithms, Methods and Techniques (ciksiti.com)

Flowchart of Apriori algorithm. | Download Scientific Diagram (researchgate.net)