|  |
| --- |
| ­  МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА − Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИИТ)**

**Кафедра Вычислительной техники (ВТ)**

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 1**

по дисциплине «Проектирование систем поддержки принятия решений»

на тему: «Ассоциотивные правила»

|  |  |  |  |
| --- | --- | --- | --- |
| Студент группы ИНБО-01-17 | *ИКБО-14-20, Вежновец Ф.Ю .* | (подпись) | |
| Преподаватель | *Железняк Л.М.* | (подпись) | |
| Отчет представлен | «\_\_\_»\_\_\_\_\_\_\_\_2023 г. | |  | |

Москва 2023 г.

Содержание

[Содержание 5](#_Toc131041709)

[ВВЕДЕНИЕ 6](#_Toc131041710)

[1. Практическая часть 7](#_Toc131041711)

[1.1 Данные для анализа 9](#_Toc131041712)

[1.2 Ассоциотивные правила в программе deductor 9](#_Toc131041713)

[1.3 Расчёт поддержки 10](#_Toc131041714)

[1.4 Расчёт достоверности 11](#_Toc131041715)

[1.5 Расчёт лифта 11](#_Toc131041716)

[1.6 Программная реализация 11](#_Toc131041717)

[ЗАключение 13](#_Toc131041718)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 14](#_Toc131041719)

[ПРИЛОЖЕНИЯ 15](#_Toc131041720)

ВВЕДЕНИЕ

**Цель работы:** найти закономерность в наборе данных при помощи ассоциативных правил.

**Задачи:** Задача данной практической работы: найти наиболее часто встречающиеся наборы товаров и набор ассоциативных правил с определенными границами значений поддержки и доверия.

В качестве основы для его построения обучающего множества объектов БД выбираем группы товаров магазина «Магазин автозапчастей».

1. Практическая часть

Аффинитивный анализ (affinity analysis) – один из распространенных методов Data Mining. Его название происходит от английского слова affinity, которое в переводе означает «близость», «сходство». Цель дангоro метода исследование взаимной связи между событиями, которые происходят совместно. Разновидностью аффинитивного анализа является анализ рыночной корзины, цель которого обнаружить ассоциации между различными событиями, то есть найти правила для количественного описания взаимной связи между двумя или более событиями. Такие правила называются ассоциативными правилами.

Примерами приложения ассоциативных правил могут быть следующие задачи:

* + - * выявление наборов товаров, которые в супермаркетах часто покупаются вместе или никогда не покупаются вместе;
      * определение доли клиентов, положительно относящихся к нововведениям в их обслуживании;
      * определение профиля посетителей веб-ресурса;
* определение доли случаев, в которых новое лекарство оказывает опасный побочный эффект.

Следующее важное понятие – предметный набор. Это непустое множество предметов, появившихся в одной транзакции.

Анализ рыночной корзины – это анализ наборов данных для определения комбинаций товаров, связанных между собой, иными словами, производится поиск товаров, присутствие которых в транзакции влияет на вероятность наличия других товаров или комбинаций товаров.

Современные кассовые аппараты в супермаркетах позволяют собирать информацию о покупках, которая может храниться в базе данных. Затем накопленные данные могут использоваться для построения систем поиска ассоциативных правил.

Поддержка ассоциативною правила – это число транзакций, которые содержат каr условие, так и следствие.

Например, для ассоциации 𝐴→𝐵 можно записать:

Достоверность ассоциативного правила 𝐴→𝐵 представляет собой меру точности правила и определяется как отношение количества транзакций, содержащих и условие, и следствие, к количеству транзакций, содержащих только условие:

Если поддержка и достоверность достаточно высоки, можно с большой вероятностью утверждать, что любая будущая транзакция, которая включает условие, будет также содержать и следствие.

Лифт – отношение частоты появления условия в транзакциях, которые также содержат и следствие к частоте появления следствия в целом. Значения лифта большие 1 показывают, что условие чаще появляется в транзакциях, содержащих следствие, чем в остальных. Можно утверждать, что лифт является обобщенной мерой связи двух предметных наборов: при значениях лифта больше 1 связь положительная, при 1 она отсутствует, а при значениях меньше 1 – отрицательная.

Лифт (оригинальное название — интерес) вычисляется следующим образом:

* 1. Данные для анализа

Исходные данные из магазина автозапчастей представленные (Рисунок 1).



Рисунок 1 – Исходные данные

Где колонка receipt\_number» это номер транзакции, а «product» название продукта.

* 1. Ассоциотивные правила в программе deductor

В программе Deductor Визуализатор "Правила" отображает ассоциативные правила в виде списка правил. Этот список представлен таблицей со столбцами: "Номер правила", "Условие", "Следствие", "Поддержка, %", "Поддержка, Количество", "Достоверность", "Лифт" (Рисунок 2).

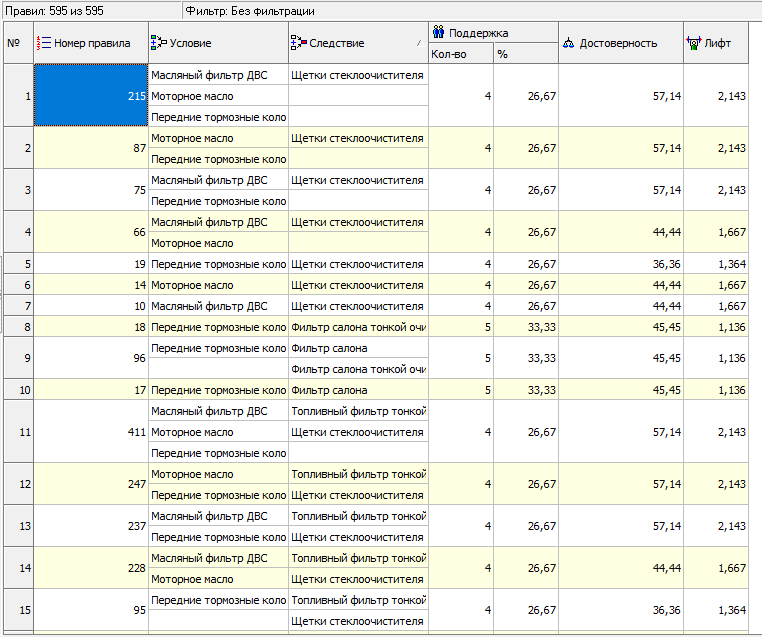


Рисунок 2 – Анализ данных в программе Deductor

* 1. Расчёт поддержки

Возьмем ассоциацию передние тормозные колодки и моторное масло. Поскольку количество транзакций, содержащих как передние тормозные колодки, так и моторное масло, равно 7, а общее число транзакций 15, то поддержка данной ассоциации будет:

* 1. Расчёт достоверности

Возьмем ассоциацию передние тормозные колодки и моторное масло.

Поскольку количество транзакций, содержащих только моторное масло (условие), равно 4, то достоверность данной ассоциации будет:

* 1. Расчёт лифта

*Лифт* – отношение частоты появления условия в транзакциях, которые также содержат и следствие к частоте появления следствия в целом.

* 1. Программная реализация

Создадим ассоциативные правила в программной реализации на языке высокого уровня Python [2].

Полученный результат работы программы ассоциативных правил (Рисунок 3).



Рисунок 3 – Результат работы программы

ЗАключение

В ходе выполнения практической работы были получены навыки расчёта ассоциативных правил в программе Deductor и последующая ее реализация на языке Python.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Сорокин А. Б. Лекции и методические материалы — РТУ МИРЭА, 2023.
2. Python documentation — URL: https://www.python.org/ (Дата обращения: 20.12.2022)
3. Matplotlib — Visualization with Python — URL: https://matplotlib.org/ (Дата обращения: 21.12.2022)
4. Информационные технологии, № 7, 2012. Теоретический и прикладной научно-технический журнал. — М.: Новые технологии, 2012. — 80 с.
5. Настройка весовых матриц ЗСУР регулятора с помощью биоинспирированных алгоритмов оптимизации. Вестник РГРТУ. 2016. № 55. — Рязань: РГРТУ. — С. 131–139.

ПРИЛОЖЕНИЯ

Приложение А – Листинг кода для ассоциативных правил

Приложение А

Листинг кода для ассоциативных правил

Листинг А.1 – Используемые библиотеки

import pandas as pd

import numpy as np

import seaborn as sn

Листинг А.2 – Файл txt с данными для анализа

Номер\_чека,Товар

15,Передние тормозные колодки

15,Моторное масло

15,Масляный фильтр ДВС

20,Фильтр салона

20,Фильтр салона тонкой очистки

30,Свечи зажигания на моделях с бензиновым ДВС

30,Передние тормозные колодки

30,Задние тормозные колодки

30,Моторное масло

30,Воздушный фильтр ДВС

30,Масляный фильтр ДВС

30,Топливный фильтр тонкой очистки

30,Щетки стеклоочистителя

40,Передние тормозные колодки

40,Фильтр салона

40,Фильтр салона тонкой очистки

45,Моторное масло

45,Масляный фильтр ДВС

50,Передние тормозные колодки

50,Масло в коробке передач

50,Жидкость ГУР

50,Масляный фильтр коробки передач

55,Тормозная жидкость

60,Свечи зажигания на моделях с бензиновым ДВС

60,Передние тормозные колодки

60,Задние тормозные колодки

60,Моторное масло

60,Воздушный фильтр ДВС

60,Масляный фильтр ДВС

60,Топливный фильтр тонкой очистки

60,Фильтр салона

60,Фильтр салона тонкой очистки

60,Щетки стеклоочистителя

70,Приводной ремень

70,Передние тормозные колодки

70,Основной аккумулятор

75,Моторное масло

75,Масляный фильтр ДВС

80,Рычаги подвески

80,Передние тормозные колодки

80,Фильтр салона

80,Фильтр салона тонкой очистки

90,Свечи зажигания на моделях с бензиновым ДВС

90,Шрусы или их составные части

90,Передние тормозные колодки

90,Задние тормозные колодки

90,Моторное масло

90,Воздушный фильтр ДВС

90,Масляный фильтр ДВС

Продолжение Листинга А.2

90,Топливный фильтр тонкой очистки

90,Щетки стеклоочистителя

100,Свечи накаливания на моделях с дизельным ДВС

100,Стойки (тяги) стабилизатора

100,Тормозные диски

100,Ступичный подшипник на передней и задней паре колёс

100,Сальники ступицы на передней и задней паре колёс

100,Передние тормозные колодки

100,Зимний и летний комплект шин

100,Топливный насос

100,Моторное масло

100,Масло в коробке передач

100,Жидкость ГУР

100,Масляный фильтр ДВС

100,Масляный фильтр коробки передач

100,Фильтр салона

100,Фильтр салона тонкой очистки

110,Передние тормозные колодки

110,Моторное масло

110,Тормозная жидкость

110,Масляный фильтр ДВС

120,Свечи зажигания на моделях с бензиновым ДВС

120,Термостат системы охлаждения ДВС

120,Термостат AGR/EGR на моделях с охлаждением рециркулируемых ОГ

120,Передние тормозные колодки

120,Задние тормозные колодки

120,Моторное масло

120,Смазка в шрусах, пыльники и хомуты

120,Воздушный фильтр ДВС

120,Масляный фильтр ДВС

120,Топливный фильтр тонкой очистки

120,Фильтр салона

120,Фильтр салона тонкой очистки

120,Щетки стеклоочистителя

Листинг А.3 – Функция main

data = pd.read\_csv('stasic//data.csv')   
data.hist()  
unique\_receipts = data.receipt\_number.unique()

print(len(unique\_receipts))

unique\_receipts  
count\_receipt = len(unique\_receipts)  
unique\_products = data['product'].unique()

print(len(unique\_products))

unique\_products  
data\_np = data.to\_numpy()  
group\_products\_receipts = []

for i in unique\_receipts:

micro\_data = []

for j in data\_np:

if j[0] == i:

micro\_data.append(j[1])

group\_products\_receipts.append(micro\_data)

Листинг А.4 – Функция создание всех сочетаний

mass\_group\_products = []

for i in range(len(unique\_products)-1):

for j in range(i + 1, len(unique\_products)):

mass\_group\_products.append([[unique\_products[i]],[unique\_products[j]]])

mass\_for\_two = [unique\_products[i], unique\_products[j]]

for two in unique\_products:

if two not in mass\_for\_two:

mass\_group\_products.append([mass\_for\_two, [two]])

mass\_for\_three = [mass\_for\_two[0], mass\_for\_two[1], two]

for three in unique\_products:

if three not in mass\_for\_three:

mass\_group\_products.append([mass\_for\_three, [three]])

mass\_for\_four = [mass\_for\_three[0], mass\_for\_three[1], mass\_for\_three[2], three]

for four in unique\_products:

if four not in mass\_for\_four:

mass\_group\_products.append([mass\_for\_four, [four]])

Листинг А.5 – Функция расчёта поддержки

for i in range(len(mass\_group\_products)):

count = 0

for j in group\_products\_receipts:

if set(mass\_group\_products[i][0]).issubset(j) and set(mass\_group\_products[i][1]).issubset(j):

count += 1

if i == 1:

print(f'{count\*100}/{count\_receipt} = {round((count\*100/count\_receipt),2)}')

mass\_group\_products[i].append((round((count\*100/count\_receipt),2)))

Листинг А.6 – Функция расчёта достоверности

for i in range(len(mass\_group\_products)):

count\_one = 0

count\_two = 0

for j in group\_products\_receipts:

if set(mass\_group\_products[i][0]).issubset(j) and set(mass\_group\_products[i][1]).issubset(j):

count\_one += 1

for j in group\_products\_receipts:

if set(mass\_group\_products[i][0]).issubset(j):

count\_two += 1

if i == 1:

print(f'{count\_one}/{count\_two} = {round((count\_one\*100/count\_two),2)}')

if count\_two != 0:

mass\_group\_products[i].append((round((count\_one\*100/count\_two),2)))

else:

mass\_group\_products[i].append(0)

Листинг А.7 – Функция расчёта лифта

for i in range(len(mass\_group\_products)):

count\_one = 0

for j in group\_products\_receipts:

if set(mass\_group\_products[i][1]).issubset(j):

count\_one += 1

if i == 1:

print(f'{count\_one}/{count\_receipt} = {round((count\_one\*100/count\_receipt),2)}')

if (count\_one\*100/count\_receipt) != 0:

mass\_group\_products[i].append((round(mass\_group\_products[i][-1]/(count\_one\*100/count\_receipt),2)))

else:

mass\_group\_products[i].append(0)

Листинг А.8 – Функция получения финальных данных

Dataframe = pd.DataFrame(dataframe, columns =['Suported', 'Reliability', 'Lift'], index=name\_for\_index)

Dataframe = (Dataframe.loc[Dataframe.Suported < 89])

Dataframe = (Dataframe.loc[20 < Dataframe.Suported])

Dataframe = (Dataframe.loc[30 < Dataframe.Reliability])

Dataframe = (Dataframe.loc[Dataframe.Reliability < 70])  
Dataframe.sort\_values('Lift', ascending=False)