РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра информационных технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

Дисциплина: Интеллектуальный анализ данных

Студент: Николаев Александр Викторович

Группа: НФИбд-01-17

Москва 2020

Вариант №18

1. При помощи модуля sqlite3 откроем базу данных Instacart в файле instacart.db.

In [1]:

```
import sqlite3

conn = sqlite3.connect('instacart.db')
```

1. При помощи запроса SELECT извлекаем из таблицы order_products__train записи, с order_dow = 2 и department_id = 8. Определяем количество записей в полученном наборе и количество товаров в транзакциях набора.

In [2]:

```
import pandas as pd

query =\
'''
select distinct
    train.*, products.product_name
from
    order_products__train as train
join
    orders
on
    train.order_id = orders.order_id
join
    products
on
    train.product_id = products.product_id
where
    orders.order_dow = 2
and
    products.department_id = 8
'''
data = pd.read_sql(query, conn)
```

In [3]:

```
data.head()
```

Out[3]:

	order_id	product_id	add_to_cart_order	reordered	product_name
0	3088145	377	4	1	Chunky Beef with Vegetables & Brown Rice Dog Food
1	3174585	49193	1	1	SmartBlend Chicken & Rice Formula Adult Dry Do...
2	2689040	30464	5	0	Original Beef Flavor Dog Snacks
3	1995260	17460	1	1	Purina Classic Pate Turkey & Giblets Dinner Ca...
4	1995260	18489	14	1	Purina Mixed Grill Classic Pate Cat Food

In [4]:

```
print(f'Количество записей = {len(data)}')
print(f'Количество товаров в транзакциях набора = {data.order_id.nunique()}')
```

```
Количество записей = 459
Количество товаров в транзакциях набора = 310
```

1. Определяем количество покупок (транзакций) для пяти наиболее популярных товаров в наборе.

Посмотрим на 10 самых популярных товаров

In [5]:

```
data.groupby('product_id')['order_id'].count().nlargest(10)
```

Out[5]:

```
product_id
49355    12
15175     7
16647     6
37520     6
12905     5
16997     5
30464     5
35412     5
48505     5
54        5
Name: order_id, dtype: int64
```

Поскольку 5-й по популярности выбрать невозможно, возьмем любой, например 12905

Таким образом, 5 самых популярных товаров:

In [6]:

```
popular_products = (data.groupby('product_id')['order_id'].count().nlargest(5).rename(
'order_count').
                    to_frame().join(data.set_index('product_id')['product_name'].drop_d
uplicates())
                   )
print('Топ 5 популярных товаров:')
popular_products
```

```
Топ 5 популярных товаров:
```

Out[6]:

product_id	order_count	product_name
49355	12	Instant Action Cat Litter
15175	7	24/7 Performance Light Weight Cat Litter
16647	6	Double Duty Advanced Odor Control Clumping Cat...
37520	6	Indoor Cat Food
12905	5	Adult Indoor Advantage Cat Food

1. Построем транзакционную базу данных для поиск ассоциативных правил из полученного набора записей таблицы order_products__train, используя в качестве идентификатора транзакции поле order_id, а в качестве названий товаров - поле product_name из таблицы products, соответствующее полю product_id.

In [7]:

```
data
```

Out[7]:

	order_id	product_id	add_to_cart_order	reordered	product_name
0	3088145	377	4	1	Chunky Beef with Vegetables & Brown Rice Dog Food
1	3174585	49193	1	1	SmartBlend Chicken & Rice Formula Adult Dry Do...
2	2689040	30464	5	0	Original Beef Flavor Dog Snacks
3	1995260	17460	1	1	Purina Classic Pate Turkey & Giblets Dinner Ca...
4	1995260	18489	14	1	Purina Mixed Grill Classic Pate Cat Food
...	...	...	...	...	...
454	1580174	7351	12	0	Multi-Cat Extra Strength Clumping Unscented Ca...
455	2316225	37068	9	1	24/7 Performance Clumping Litter
456	2286887	19368	1	1	With Glade Tough Odor Solutions Cat Litter
457	390836	32635	16	0	Standard Size Pet Waste bags
458	1345413	45751	4	0	Light Weight Instant Action Litter

459 rows × 5 columns

In [8]:

```
# идём по уникальным заказам и создаём множество из продуктов, содержащихся в нём
# полученную пару заказ - множество добавляем в транзакционную базу данных
transactions = []
for order in data['order_id'].unique():
    products = set(data[data['order_id'] == order]['product_name'])
    pair = list((order, products))
    transactions.append(pair)
```

In [9]:

```
# выведем первые пять тразакций
for i in range(5):
    print(transactions[i])
```

```
['3088145', {'Chunky Beef with Vegetables & Brown Rice Dog Food'}]
['3174585', {'SmartBlend Chicken & Rice Formula Adult Dry Dog Food'}]
['2689040', {'Original Beef Flavor Dog Snacks'}]
['1995260', {'Chicken & Salmon Dinner in Gravy Savory Shreds Cat Food', 'S
avory Shreds Turkey & Cheese Dinner in Gravy Cat Food', 'Tasty Treasures C
hicken, Tuna & Cheese In Gravy Cat Food', "Classic Pate Mariner's Catch Ca
t Food", 'Indoor Saucy Seafood Bake in Sauce Cat Food', 'Purina Mixed Gril
l Classic Pate Cat Food', 'Classic Pate Ocean Whitefish & Tuna Dinner Cat
Food', 'Classic Pate Poultry Platter Cat Food', 'Proactive Health Indoor W
eight & Hairball Care with Chicken Cat Food', 'Purina Classic Pate Turkey
& Giblets Dinner Cat Food'}]
['632142', {'SmartBlend Lamb & Rice Formula Adult Dog Food'}]
```

1. Реализуйте указанный в индивидуальном задании метод построения популярных наборов предметов (Apriori/Eclat/Declat) (3 балла) или используйте метод BruteForce (0 баллов). Протестируйте корректность реализации алгоритма на учебном наборе данных из материалов лекции.

Реализуем Eclat

Основная идея алгоритма - поиск в глубину для эффективного подсчета по памяти и скорости

In [10]:

```
def prepare_for_eclat(database):
    all_items = set()
    for pair in database:
        all_items = all_items.union(pair[1])
    all_items

    res = dict()
    for item in all_items:
        res[item] = set()
        for pair in database:
            if item in pair[1]:
                res[item].add(pair[0])

    res = list(res.items())
    return res

def eclat(prefix, items):
    while items:
        i, itids = items.pop()
        isupp = len(itids)
        if isupp >= minsup:
            items_freq[frozenset(prefix + [i])] = isupp
            suffix = []
            for j, ojtids in items:
                jtids = itids & ojtids
                if len(jtids) >= minsup:
                    suffix.append((j,jtids))
            eclat(prefix+[i], sorted(suffix, key=lambda item: len(item[1]), reverse=Tru
e))
```

Проверим реализацию на примере с семинара.

In [11]:

```
D_train = [
    [ 1, {"A","B","D","E"} ],
    [ 2, {"B","C","E"} ],
    [ 3, {"A","B","D","E"} ],
    [ 4, {"A","B","C","E"} ],
    [ 5, {"A","B","C","D","E"} ],
    [ 6, {"B","C","D"} ],
]

minsup = 3
items_freq = dict()
eclat([], prepare_for_eclat(D_train))

res_lst = []
for key, value in items_freq.items():
    tmp = []
    tmp.append(tuple(key))
    tmp.append(value)
    res_lst.append(tmp)
res_lst = sorted(res_lst, key=lambda x: len(x[0]))

for result in res_lst:
    print(result)
```

```
[('C',), 4]
[('D',), 4]
[('A',), 4]
[('B',), 6]
[('E',), 5]
[('E', 'C'), 3]
[('B', 'C'), 4]
[('D', 'A'), 3]
[('E', 'D'), 3]
[('B', 'D'), 4]
[('B', 'A'), 4]
[('E', 'A'), 4]
[('B', 'E'), 5]
[('B', 'E', 'C'), 3]
[('E', 'D', 'A'), 3]
[('B', 'D', 'A'), 3]
[('B', 'E', 'D'), 3]
[('B', 'E', 'A'), 4]
[('B', 'E', 'D', 'A'), 3]
```

Всё совпадает

1. При помощи Elcat построим популярные наборы товаров с минимальной поддержкой, равной половине среднего количества покупок пяти наиболее популярных товаров.

Можно по разному округлять minsup, давайте возьмем пополам среднее и математическое округление (можно целочисленно поделить на два, можно брать округление вверх/вниз)

In [12]:

```
minsup = round(popular_products.order_count.mean() / 2)
print(f'Среднее количество покупок популярных товаров = {popular_products.order_count.mean()}')
print(f'minsup = {minsup}')
```

```
Среднее количество покупок популярных товаров = 7.2
minsup = 4
```

Получается minsup 3 или 4 в зависимости от стратегии округления и деления.

Но поскольку при таких значениях minsup всё множество наборов состоит из одного продукта ассоциативные правила получаются неинтересными, поэтому возьмем minsup = 2, чтобы в 7 пункте посчитать правила поинтереснее, чем просто []

In [13]:

```
minsup=2
items_freq = dict()
eclat([], prepare_for_eclat(transactions))

res_lst = []
for key, value in items_freq.items():
    tmp = []
    tmp.append(tuple(key))
    tmp.append(value)
    res_lst.append(tmp)
res_lst = sorted(res_lst, key=lambda x: len(x[0]))

print('Полученные популярные наборы:\n')
for result in res_lst:
    print(result)
```

Полученные популярные наборы:

```
[('Grilled Chicken Feast in Gravy Cat Food',), 4]
[('Ocean Whitefish & Tuna Feast in Sauteed Seafood Flavor Gravy Cat Foo
d',), 2]
[('Adult Indoor Advantage Cat Food',), 5]
[('Jumbone Mini Toy/Small (PS #5129732) Dog Care & Treats',), 2]
[('Brushless Dog Toothpaste',), 2]
[('Flaked Tuna Feast Cat Food',), 2]
[('Multi Cat Scented Scoopable Cat Litter',), 3]
[('Original Beef Flavor Dog Snacks',), 5]
[('Organic Dog Cookies Peanut Butter',), 2]
[('Real Chicken & Veggies Recipe Dog Food',), 2]
[('Chicken Jerky Strips Dog Treats',), 2]
[('Beyond Grain Free Wild Salmon Recipe Wet Cat Food',), 2]
[('Super Scoop Clumping with Odor Eliminating Baking Soda Cat Litter',),
2]
[('Beef Feast in Roasted Beef Flavor Gravy Cat Food',), 2]
[('Romana Style Medley Dog Food',), 2]
[('Multi-Cat Extra Strength Clumping Cat Litter',), 2]
[('Grain Free Turkey Formula Cat Food',), 2]
[("Chicken Lickin' Chicken Pate with Vegetables Cat Food",), 2]
[('Grain Free Turkey & Salmon Formula Cat Food',), 4]
[('Grain Free Turkey Canned Cat Food',), 2]
[('Pet Fresh Carpet Odor Eliminator + Oxi Clean Dirt Fighters',), 2]
[('Organix Butcher & Bushel Chicken Wing & Thigh Grain Free Adult Dog Foo
d',), 3]
[('Beef Jerky',), 2]
[('Classic Seafood Feast Cat Food',), 2]
[('Poultry & Beef Feast Variety Grilled Wet Cat Food',), 5]
[('Heavy Duty Jumbo Litter Box Liners',), 2]
[('Florentine Collection Cat Food',), 2]
[('Original Maximum Odor Control Cat Litter',), 2]
[('Adult Health Small Breed Formula Dry Dog Food',), 2]
[("Classic Pate Mariner's Catch Cat Food",), 2]
[('Chicken & Brown Rice Premium Dog Food',), 2]
[('Proactive Health Indoor Weight & Hairball Care with Chicken Cat Foo
d',), 2]
[('Original Cat Chow Naturals Plus Vitamins & Minerals Cat Food',), 2]
[('Fancy Feast Seafood Feast Variety Classic Wet Cat Food',), 4]
[('Non Clumping 24/7 Performance Cat Litter',), 2]
[('Double Duty Advanced Odor Control Clumping Cat Litter',), 6]
[('With Glade Tough Odor Solutions Cat Litter',), 4]
[('Chicken and Liver Canine Cuisine Wet Dog Food',), 2]
[('Cat Treats Temptations Seafood Medley',), 2]
[('Savory Shreds Turkey & Cheese Dinner in Gravy Cat Food',), 2]
[('Indoor Delights Cat Food',), 3]
[('Fancy Feast Wet Classic Chicken Feast Cat Food',), 2]
[('Super Scoop Instant Clumping Unscented Cat Litter',), 3]
[('Purina Classic Pate Turkey & Giblets Dinner Cat Food',), 2]
[('Fancy Feast Wet Classic Turkey & Giblets Feast Cat Food',), 2]
[('Classic Tender Beef Feast Cat Food',), 2]
[('Grain Free Turkey, Sweet Potato & Spinach Recipe in Gravy Cat Food',),
3]
[('Temptations Tasty Chicken Flavor Cat Treats',), 3]
[('24/7 Performance Light Weight Cat Litter',), 7]
[('Grain Free Chicken Formula Cat Food',), 5]
[('Classic Poultry & Beef Feast Variety Cat Food',), 2]
[('Instant Action Cat Litter',), 12]
[('Poultry & Beef Feast Variety  Cat Food',), 2]
[('Biscuits for Dogs less than 20 lbs',), 4]
```

```
[('24/7 Performance Cat Litter',), 5]
[('Tasty Chicken Flavor Treats for Cat',), 2]
[('Organix Butcher & Bushel Grain-Free Turkey Dinner Canned Dog Food',),
2]
[('Grilled Seafood Feast Variety Cat Food',), 3]
[('Organic Turn Up Da Turkey Recipe',), 2]
[('Classic Chopped Grill Feast Cat Food',), 2]
[('Flaked Skipjack Tuna Cat Food',), 2]
[('Complete Health Deboned Chicken, Chicken Meal and Rice Cat Food',), 2]
[('Elegant Medleys Tuna Recipe Variety Wet Cat Food',), 2]
[('Indoor Cat Food',), 6]
[('Organix Cat Treats',), 3]
[('Prime Filets Meaty Favorites Variety Pack Cat Food',), 2]
[('Busy Bone Dog Treats Real Meat Sm & Med 6',), 3]
[('Multiple Cat Clumping Formula Cat Litter',), 3]
[('Cat Chow Complete Formula',), 3]
[('Salmon Feast in Seared Salmon Flavor Gravy Cat Food',), 2]
[('Smart Blend Lamb & Rice Dry Dog Food',), 2]
[('Organix Chicken & Brown Rice Recipe',), 4]
[('Scented Multi-Cat Scoopable Cat Litter',), 2]
[('Prepared Meals Simmered Chicken Medley Wet Dog Food',), 2]
[('Simply 9 White Meat Chicken & Whole Barley Recipe Dog Food',), 2]
[('Friskies Dry Signature Blend Tender & Crunchy Combo Cat Food',), 2]
[('24/7 Performance Clumping Litter',), 2]
[('SauceSations Chicken Dinner in Garden Sauce Cat Food',), 2]
[('Standard Size Pet Waste bags',), 3]
[("Surfin' & Turfin' Favorites Cat Food",), 5]
[('ProActive Health Adult MiniChunks Premium Dry Dog Food',), 2]
[('Small Dog Biscuits',), 2]
[('Purina Mixed Grill Classic Pate Cat Food',), 2]
[('Turkey Stew With Barley & Carrots Natural Dog Food',), 2]
[('Classic Tender Liver & Chicken Feast Cat Food',), 2]
[('Wild Mackerel & Whitefish Cat Food',), 2]
[('Original Choice Dry Cat Food',), 2]
[('Beef Stew Canned Dog Food',), 2]
[('Premium Dog Treats Peanut Butter Flavor, Medium Size',), 2]
[('Core Original Grain Free Dry Dog Food',), 2]
[('Gravy Lovers Chicken Feast Can Cat Food',), 3]
[('Canine Cuisine Gourmet Filet Mignon New York Strip Wet Dog Food',), 2]
[('Natural Dog Food Turkey & Sweet Potato Formula',), 2]
[('Dog Snacks, Medium',), 4]
[('Purina Beyond Natural Cat Food Grain Free Ocean Whitefish & Egg Recip
e',), 2]
[('Organix Chicken & Rice Dry Cat Food',), 2]
[('Classic Pate Poultry Platter Cat Food',), 2]
[('Good Buddy Beef Jerky Dog Treat',), 2]
[('Select Tender Chicken with Vegetables & Brown Rice Dog Food Recipe',),
5]
[('Liv A Littles Chicken Cat & Dog Treats',), 2]
[('Salmon Feast in Seared Salmon Flavor Gravy Cat Food', 'Ocean Whitefish
& Tuna Feast in Sauteed Seafood Flavor Gravy Cat Food'), 2]
[('Beef Feast in Roasted Beef Flavor Gravy Cat Food', 'Gravy Lovers Chicke
n Feast Can Cat Food'), 2]
[('Organix Butcher & Bushel Grain-Free Turkey Dinner Canned Dog Food', 'Or
ganix Butcher & Bushel Chicken Wing & Thigh Grain Free Adult Dog Food'),
2]
[('Purina Mixed Grill Classic Pate Cat Food', "Classic Pate Mariner's Catc
h Cat Food"), 2]
[('Classic Pate Poultry Platter Cat Food', "Classic Pate Mariner's Catch C
at Food"), 2]
[('Classic Pate Poultry Platter Cat Food', 'Purina Mixed Grill Classic Pat
```

```
e Cat Food'), 2]
[('Classic Pate Poultry Platter Cat Food', 'Purina Mixed Grill Classic Pat
e Cat Food', "Classic Pate Mariner's Catch Cat Food"), 2]
```

1. Для какого-либо из полученных популярных наборов товаров постройте набор ассоциативных правил.

In [14]:

```
sample, _ = res_lst[-1]
print(sample, len(sample))
```

```
('Classic Pate Poultry Platter Cat Food', 'Purina Mixed Grill Classic Pate
Cat Food', "Classic Pate Mariner's Catch Cat Food") 3
```

In [15]:

```
from itertools import chain, combinations

def ComputeSupport(X, D):
    supX = 0
    for _, itemset in D:
        if X.issubset(itemset):
            supX += 1
    return supX

def powersetk(iterable, k):
    xs = list(iterable)
    return list(chain.from_iterable(combinations(xs, n) for n in range(k, len(xs) + 1
)))[:-1]

def AssociationRules(D, Z_set, minconf):
    A_rules = []
    supZ = ComputeSupport(set(Z_set), D)
    A_set = powersetk(Z_set, 1)
    while len(A_set)>0:
        X_set = A_set[-1]
        A_set.pop()
        conf = supZ / ComputeSupport(set(X_set), D)
        if conf >= minconf:
            Y_set = sorted(list(set(Z_set) - set(X_set)))
            A_rules.append([X_set, Y_set])
        else:
            for W_set in powersetk(X_set, 1):
                if W_set in A_set:
                    A_set.remove(W_set)
    return A_rules

rules = AssociationRules(transactions, sample, 0.9)
print('Полученные ассоциативные правила:\n')
for rule in rules:
    print('rule:')
    for x in rule:
        print(x)
    print()
```

```
Полученные ассоциативные правила:

rule:
('Purina Mixed Grill Classic Pate Cat Food', "Classic Pate Mariner's Catch
Cat Food")
['Classic Pate Poultry Platter Cat Food']

rule:
('Classic Pate Poultry Platter Cat Food', "Classic Pate Mariner's Catch Ca
t Food")
['Purina Mixed Grill Classic Pate Cat Food']

rule:
('Classic Pate Poultry Platter Cat Food', 'Purina Mixed Grill Classic Pate
Cat Food')
["Classic Pate Mariner's Catch Cat Food"]

rule:
("Classic Pate Mariner's Catch Cat Food",)
['Classic Pate Poultry Platter Cat Food', 'Purina Mixed Grill Classic Pate
Cat Food']

rule:
('Purina Mixed Grill Classic Pate Cat Food',)
["Classic Pate Mariner's Catch Cat Food", 'Classic Pate Poultry Platter Ca
t Food']

rule:
('Classic Pate Poultry Platter Cat Food',)
["Classic Pate Mariner's Catch Cat Food", 'Purina Mixed Grill Classic Pate
Cat Food']
```

1. Для построенного набора ассоциативных правил вычислим показатели: support, confidence, lift, leverage, conviction и выведите на экран.

In [16]:

```
import numpy as np

N = len(transactions)
supps = []
confs = []
lifts = []
leverages = []
convictions = []
for rule in rules:
    x1 = set(rule[0])
    x2 = set(rule[1])
    supp_x1_x2 = ComputeSupport(x1.union(x2), transactions) / N # support
    supp_x1 = ComputeSupport(x1, transactions) / N
    supp_x2 = ComputeSupport(x2, transactions) / N
    conf = supp_x1_x2 / supp_x1 # confidence
    lift = supp_x1_x2 / (supp_x1 * supp_x2) # lift
    leverage = supp_x1_x2 - supp_x1 * supp_x2 # leverage
    if conf == 1:
        conv = np.inf
    else:
        conv = (1 - supp_x2) / (1 - conf) # conviction
    supps.append(supp_x1_x2)
    confs.append(conf)
    lifts.append(lift)
    leverages.append(leverage)
    convictions.append(conv)
```

Полученные правила и показатели:

In [17]:

```
for i in range(len(rules)):
    print('rule:')
    for x in rules[i]:
        print(x)
    print(f'Support = {supps[i]}')
    print(f'Confidence = {confs[i]}')
    print(f'Lift = {lifts[i]}')
    print(f'Leverage = {leverages[i]}')
    print(f'Conviction = {convictions[i]}')
    print()
```

```
rule:
('Purina Mixed Grill Classic Pate Cat Food', "Classic Pate Mariner's Catch
Cat Food")
['Classic Pate Poultry Platter Cat Food']
Support = 0.0064516129032258064
Confidence = 1.0
Lift = 155.0
Leverage = 0.006409989594172737
Conviction = inf

rule:
('Classic Pate Poultry Platter Cat Food', "Classic Pate Mariner's Catch Ca
t Food")
['Purina Mixed Grill Classic Pate Cat Food']
Support = 0.0064516129032258064
Confidence = 1.0
Lift = 155.0
Leverage = 0.006409989594172737
Conviction = inf

rule:
('Classic Pate Poultry Platter Cat Food', 'Purina Mixed Grill Classic Pate
Cat Food')
["Classic Pate Mariner's Catch Cat Food"]
Support = 0.0064516129032258064
Confidence = 1.0
Lift = 155.0
Leverage = 0.006409989594172737
Conviction = inf

rule:
("Classic Pate Mariner's Catch Cat Food",)
['Classic Pate Poultry Platter Cat Food', 'Purina Mixed Grill Classic Pate
Cat Food']
Support = 0.0064516129032258064
Confidence = 1.0
Lift = 155.0
Leverage = 0.006409989594172737
Conviction = inf

rule:
('Purina Mixed Grill Classic Pate Cat Food',)
["Classic Pate Mariner's Catch Cat Food", 'Classic Pate Poultry Platter Ca
t Food']
Support = 0.0064516129032258064
Confidence = 1.0
Lift = 155.0
Leverage = 0.006409989594172737
Conviction = inf

rule:
('Classic Pate Poultry Platter Cat Food',)
["Classic Pate Mariner's Catch Cat Food", 'Purina Mixed Grill Classic Pate
Cat Food']
Support = 0.0064516129032258064
Confidence = 1.0
Lift = 155.0
Leverage = 0.006409989594172737
Conviction = inf
```

In [ ]:

```
```