### **PROJECT-4** : создание рекомендательной системы для онлайн-школы MasterMind

Задача прямая**:** Анализ интереса пользователей. Создание таблицы рекомендаций, т.е. к каждому курсу в базе будут рекомендованы еще два

Задача косвенная: Увеличение среднего чека покупки Курсов

### [// ссылка на проект в Google Drive](https://drive.google.com/drive/folders/1yJAzW_o-GJFb-g-Ury1YHdx_w4DmrdLT?usp=share_link)

[// ссылка на Google Colab](https://colab.research.google.com/drive/1Og2gSE4ov6sR5HUSOchLV-QxOs72kuLO?usp=sharing)

### На этапе анализа на базе sql необходимо : изучить исходные данные, подготовить их для дальнейшего анализа.

1. Выгрузить из БД: таблицу всех пользователей, купивших курс и их покупки, те курсы (Таблица Couple)
2. Обработать данные (Таблица Couple) средствами Python и создать таблицу с рекомендациями. Каждому курсу в базе будут рекомендованы еще два.
3. Рекомендательную систему будем строить на основании тех курсов которые человек уже купил. Посмотрим какие курсы чаще всего встречаются в паре с купленным. Выберем 2 с самой высокой частотой повторений.
4. Установим минимальную границу (число повторений) значимости на квантиль = 0.25 (в нашем датасете это 1, те отсутствие повторений вовсе). Вместо этих курсов будет рекомендовать альтернативный вариант.
5. Для запуска A/B-теста рассчитан минимальный размер выборки для получения статистически значимого результата. (7 866 человек)

### **ТАБЛИЦА РЕКОМЕНДАЦИЙ**

### 

| **courses\_ids** | **recommend\_1** | **recommend\_2** |
| --- | --- | --- |
| 356 | 1144 | 1125 |
| 515 | 1125 | 912 |
| 566 | 752 | 570 |
| 679 | 570 | 551 |
| 504 | 745 | 568 |
| 551 | 1138 | 1125 |
| 570 | 566 | 551 |
| 514 | 777 | 566 |
| 571 | 551 | 551 |
| 1129 | 566 | 566 |
| 503 | 1103 | 777 |
| 523 | 745 | 566 |
| 568 | 840 | 572 |
| 1138 | 1152 | 1128 |
| 511 | 1152 | 1125 |
| 745 | 1125 | 1115 |
| 516 | 1185 | 1140 |
| 741 | 1161 | 1145 |
| 1181 | 1184 | 1115 |
| 569 | 1115 | 809 |
| 764 | 1139 | 1099 |
| 572 | 777 | 566 |
| 794 | 1188 | 1141 |
| 490 | 1128 | 1115 |
| 489 | 1152 | 1128 |
| 564 | 1187 | 1146 |
| 502 | 777 | 753 |
| 753 | 809 | 566 |
| 750 | 1138 | 1125 |
| 829 | 1184 | 1161 |
| 791 | 1185 | 1152 |
| 752 | 1144 | 1125 |
| 776 | 1187 | 1147 |
| 563 | 1103 | 1100 |
| 908 | 566 | 566 |
| 1144 | 1116 | 1103 |
| 357 | 1185 | 1152 |
| 1128 | 1187 | 1139 |
| 1141 | 1161 | 1145 |
| 553 | 1184 | 1181 |
| 519 | 1186 | 1144 |
| 365 | 1161 | 1145 |
| 659 | 794 | 753 |
| 368 | 1185 | 1140 |
| 1125 | 566 | 566 |
| 840 | 753 | 750 |
| 863 | 1138 | 1100 |
| 358 | 1129 | 1116 |
| 809 | 777 | 764 |
| 1103 | 1141 | 664 |
| 363 | 809 | 794 |
| 1184 | 741 | 566 |
| 756 | 1187 | 1147 |
| 360 | 1103 | 794 |
| 507 | 1125 | 1103 |
| 1100 | 1184 | 1181 |
| 552 | 750 | 671 |
| 664 | 1115 | 809 |
| 518 | 1100 | 794 |
| 359 | 1103 | 794 |
| 757 | 1115 | 1103 |
| 672 | 1125 | 912 |
| 1156 | 1103 | 794 |
| 765 | 1115 | 1103 |
| 1140 | 794 | 745 |
| 562 | 1125 | 1103 |
| 1188 | 566 | 566 |
| 509 | 566 | 566 |
| 749 | 1184 | 1181 |
| 366 | 566 | 566 |
| 1102 | 672 | 566 |
| 907 | 840 | 809 |
| 1152 | 1103 | 1100 |
| 1161 | 566 | 566 |
| 517 | 566 | 566 |
| 777 | 566 | 566 |
| 1139 | 566 | 566 |
| 670 | 566 | 566 |
| 1145 | 566 | 566 |
| 671 | 1161 | 1145 |
| 1101 | 566 | 566 |
| 361 | 908 | 566 |
| 1099 | 566 | 566 |
| 513 | 566 | 566 |
| 902 | 566 | 566 |
| 743 | 566 | 566 |
| 1115 | 1103 | 908 |
| 1116 | 1125 | 1124 |
| 367 | 908 | 829 |
| 1187 | 566 | 566 |
| 912 | 1186 | 1125 |
| 814 | 1187 | 1147 |
| 1124 | 1129 | 1125 |
| 1186 | 1103 | 1102 |
| 742 | 1103 | 1101 |
| 909 | 1125 | 1102 |
| 1146 | 1115 | 809 |
| 744 | 1184 | 1181 |
| 508 | 1100 | 908 |
| 835 | 1128 | 908 |
| 837 | 1186 | 1144 |
| 364 | 1124 | 566 |
| 362 | 1100 | 679 |
| 803 | 1125 | 672 |
| 810 | 1187 | 1146 |
| 813 | 1185 | 1141 |
| 862 | 1188 | 1140 |
| 1160 | 1125 | 571 |
| 800 | 1161 | 840 |
| 1104 | 1139 | 745 |
| 755 | 1099 | 745 |
| 864 | 809 | 566 |
| 1147 | 566 | 566 |
| 834 | 566 | 566 |
| 1185 | 1145 | 840 |
| 836 | 1184 | 1115 |
| 1198 | 566 | 566 |
| 1199 | 1181 | 1115 |
| 911 | 1188 | 1140 |
| 830 | 1125 | 912 |
| 833 | 1139 | 1099 |
| 1182 | 1185 | 1141 |
| 865 | 566 | 566 |
| 866 | 566 | 566 |
| 1200 | 566 | 566 |
| 1201 | 566 | 566 |

### 

### **ОПИСАНИЕ ТАБЛИЦ ДАННЫХ**

| пользовательские корзины | **carts** |  | **cart\_items** | курсы в корзине пользователей |
| --- | --- | --- | --- | --- |
| *ID промокода, если он есть* | promo\_code\_id |  | id | *ID операции* |
| *дата оплаты* | purchased\_at |  | resource\_type | *тип продукта* |
| *ID пользователя* | **user\_id** |  | **resource\_id** | *ID курса* |
| *дата создания корзины* | created\_at |  | created\_at | *дата создания события* |
| *дата последнего обновления информации* | updated\_at |  | updated\_at | *дата последнего обновления информации* |
| *ID корзины* | **id** |  | **cart\_id** | *ID корзины* |
| *состояние оплаты* | state |  |  |  |

Данные из SQL будут выгружаться в таблице **data\_sql** : user\_id, resource\_id,

где user\_id - перечень всех id пользователей, купивших курс, resource\_id - соответствующий курс.

### **SQL - СКРИПТЫ**

-- В таблицах содержатся данные по продажам курсов за 2017-2018 гг

-- выборка минимальной и максимальной дат из объединенных таблиц

SELECT *MAX(purchased\_at),*

*MIN(purchased\_at)*

FROM final.carts c JOIN final.cart\_items it

ON c.id = it.cart\_id

WHERE it.resource\_type = 'Course'

-- Всего 49 006 клиентов были замечены за покупкой курсов

-- CTE x: группировка пользователей с вычислением числа купленных курсов >0

**With x AS**(

SELECT *c.user\_id as users,*

*it.resource\_type,*

*count(c.purchased\_at)*

FROM final.carts c JOIN final.cart\_items it ON c.id = it.cart\_id

WHERE it.resource\_type = 'Course'

GROUP BY c.user\_id, it.resource\_type

HAVING count(c.purchased\_at) > 0

)

SELECT**count(users)**

FROM x

-- В данных содержится 127 позиций с различными курсами [тип 'Course']

-- CTE x: выборка позиций с курсами

**With x AS**(

SELECT *resource\_id,*

*resource\_type*

FROM final.cart\_items

GROUP BY resource\_id, resource\_type

HAVING resource\_type = 'Course'

)

SELECT **COUNT(resource\_id)**

FROM x

-- Среднее число купленных курсов на одного клиента = 1.44

-- CTE x: группировка пользователей с вычислением числа купленных курсов >0

**With x AS**(

SELECT *c.user\_id as users,*

*count(c.purchased\_at) as purch,*

*it.resource\_type*

FROM final.carts c JOIN final.cart\_items it ON c.id = it.cart\_id

WHERE it.resource\_type = 'Course'

GROUP BY c.user\_id, it.resource\_type

HAVING count(c.purchased\_at) > 0

)

SELECT **AVG(purch)**

FROM x

-- 12 656 клиентов купили больше 1 курса

-- CTE x: группировка клиентов, по типу продукта и успешным оплатам, подсчет количества купленных курсов.

**With x AS**(

SELECT c.user\_id,

c.state,

it.resource\_type,

count(it.resource\_id)

FROM final.carts c JOIN final.cart\_items it ON c.id = it.cart\_id

WHERE it.resource\_type = 'Course' AND state = 'successful'

GROUP BY 1,2,3

HAVING count(distinct it.resource\_id) > 1

)

SELECT count(user\_id)

FROM x

### **ТАБЛИЦА К ВЫГРУЗКЕ В PYTHON**

-- CTE x: таблица с успешными покупками курсов

**With x AS**(

SELECT c.user\_id,

c.state,

it.resource\_type,

it.resource\_id

FROM final.carts c JOIN final.cart\_items it ON c.id = it.cart\_id

WHERE it.resource\_type = 'Course'

AND state = 'successful'

),

-- CTE y: таблица клиентов, купивших более одного курса и количеством покупок

**y AS**(

SELECT x.user\_id, count(x.resource\_id)

FROM x

GROUP BY 1

HAVING count(distinct x.resource\_id) > 1

),

-- CTE u: таблица id клиентов для дальнейшего сравнения и выборки

**u AS**(

SELECT user\_id FROM y

)

-- таблица клиентов, которым соответствует купленный курс (34074 строк)

SELECT distinct(u.user\_id, x.resource\_id)

FROM x

RIGHT JOIN u ON x.user\_id=u.user\_id

### **РЕЗЮМИРУЕМ**

-- В таблицах содержатся данные по продажам курсов за 2017-2018 гг

-- 127 различных курса у школы

-- 49 006 клиентов покупали курсы

-- 12 656 клиентов купили больше 1го курса (это 25 % всех покупателей курсов)

-- 1.44 - среднее число купленных курсов на одного клиента

### **PYTHON NOTEBOOK**

from openpyxl import Workbook

import pandas as pd

from collections import Counter

import itertools

# 1. Выгрузка данных

# df, [34074 rows × 2 cols] - таблица пользоввтелей и купленных ими курсов

df = pd.read\_csv('data/data\_sql.csv')

# 2. Подготовка словаря пар курсов, которые встречаются в опкупках у клиентов, сортировка

# Для каждого клиента в df собраны все купленные им курсы

# Зеркальные пары убираем сортировкой

user\_groups = df.groupby('user\_id')['resource\_id'].apply(list).reset\_index()

user\_groups['sorted\_resource'] = user\_groups['resource\_id'].apply(lambda x: sorted(x)) # display(user\_groups)

# Узнаем сколько различных пар курсов встречаются вместе в покупках клиентов

# используем вспомогательный list(), itertools.combinations() и Counter()

courses = []

for i in user\_groups['sorted\_resource']:

for j in itertools.combinations(i, 2):

courses.append(j) # display(courses)

# Количество уникальных пар в списке - 3989 уникальные пары

couplesCount = Counter(courses)

couplesUnique = list(couplesCount) # display(len(c))

# Cловарь пар курсов: 3989 (key - пара, value - количество повторений)

couples\_most\_common = couplesCount.most\_common()

couples\_dict = dict(couples\_most\_common)

couples\_dictC = Counter(couples\_dict) # display(couples\_dictC)

# Таблица уникальных id курсов, за исключением курса, который ни разу не продавался

cources\_and\_purchase = df.groupby('resource\_id')['user\_id'].count().reset\_index()

all\_cources = cources\_and\_purchase['resource\_id']

# Разделим на квантили количества повторений

# Пары с q=0.25 (повторений нет) и q=0.5 (3 повторения) рекоментовать не будем

# Для рекомендации оставим курсы с q=0.75 (9 повторений)

# Для курсов, не прошедших нимальгую границу: будем рекомендовать самую популярную пару кусов (551, 566): 797 повторений пары

# курс с id 551 куплен 2935 раз, 566 - 2342 раза

# display(cources\_and\_purchase.sort\_values(by='user\_id', ascending = False))

# Запишем перечень пар и количество их повторений во вспомогательный df

df1 = pd.DataFrame({'couples': couples\_most\_common}, index=None)

df1['count\_repeat'] = df1['couples'].apply(lambda x: x[1]) # количество

quantiles = df1['count\_repeat'].quantile(q=[0.25, 0.5, 0.75]) # display('quantiles', quantiles)

# 3. Подготовка Таблицы рекомендаций:

# Функция рекомендаций на вход принимает номер курса, возврвщвет 2 курса, с max числом повторенй

# Цикл по всем уникальным курсам и запись результатов в датафрейм.

def Recommend(x):

cource\_list = []

for i in couples\_dict.keys():

if (i[0] == x and couples\_dictC[i] >= 9): # условие: 1е знач. ключа соответствует номеру курса, повторений >=9

cource\_list.append((i[1], couples\_dictC[i])) # ключ и значение в список : курс-рекомндация

if i[1] == x and couples\_dictC[i] >= 9: # условие: 2е знач. ключа (рекомндация) соответствует номеру курса, повторений >=9

cource\_list.append((i[0], couples\_dictC[i])) # ключ и значение в список :

if i[0] == x and couples\_dictC[i] < 9:

cource\_list.append((551,0)) # (551, 566): 797 - самая популярная пара кусов, рекомендуем их альтернативно

if i[1] == x and couples\_dictC[i] <9:

cource\_list.append((566,1))

sorted\_list = sorted(cource\_list, key=lambda x:x[0], reverse = True) # сортировка от большего к меньшему числу повторений

return sorted\_list[:2]

# display(all\_cources.apply(lambda x: Recommend(x))) # исходный вид выгрузки

# пустой df для таблицы

recommend\_df = pd.DataFrame(columns = ['courses\_ids', 'recommend\_1', 'recommend\_2'])

# Заполнение df рекомендаций

recommend\_df['courses\_ids'] = df['resource\_id'].unique()

recommend\_df['recommend\_1']= all\_cources.apply(lambda x: Recommend(x)[0][0])

recommend\_df['recommend\_2'] = all\_cources.apply(lambda x: Recommend(x)[1][0])

display(recommend\_df)

# Выгрузка таблицы рекомендаций в excel

recommend\_df.to\_excel('data/recommend\_df.xlsx', sheet\_name='table', index=False)

### **ЗАПУСК А/В ТЕСТА**

| Рассчитать **минимальный размер выборки** для получения статистически значимого результата | | | | | |
| --- | --- | --- | --- | --- | --- |
| Все клиенты случайным образом делятся на контрольную и тестовую группы:  тестовой группе показываются рекомендации, а контрольной — нет. | | | | | |
| Ранее средняя CR в покупку второго курса была 3,2%.  ожидаемое увеличение до 4%.  при уровне достоверности 95%  статистической мощности 80%.  для расчёта используем Калькулятор Эвана-Миллера  <https://www.evanmiller.org/ab-testing/sample-size.html> | | | | | |
| 3,2% | **текущая CR** | | | | |
| 4,0% | **ожидаемая CR** | | | | |
| 0,80% | **мин.увеличение СR** | | | | |
| 80% | **статистическая мощность 1−β:** | | | | |
| 5% | **ур. значимости α** | | | | |
| **7 866** | **размер абсолютной выборки** | | | | |

| **Оценка результатов** | | | | | |
| --- | --- | --- | --- | --- | --- |
| В контрольной группе оказалось 8732 клиента,  из них 293 купили больше одного курса.  В тестовой — 8847 клиентов,  из них 347 купили больше одного курса | | | | | |
| Рассчитаем p-value в калькуляторе  <https://abtestguide.com/calc/> | | | | | |
| 8 732 | **клиенты тестовой группы** | | | | |
| 293 | **купили больше одного курса** | | | | |
| 8 847 | **клиенты контрольной группы** | | | | |
| 347 | **купили больше одного курса.** | | | | |
| 0,0224 | **p-value** | | | | |
| Вариант с рекомендациями показал статистически значимо лучший результат, нужно реализовывать его для всех клиентов. | | | | | |