Анализ пользовательского взаимодействия с сервисом новостей

Содержание

Почти всё время аналитика в Яндекс.Дзен занимает анализ пользовательского взаимодействия с карточками статей.

Каждую карточку определяют её тема и источник (у него тоже есть тема). Примеры тем: «Красота и здоровье», «Россия», «Путешествия».

Пользователей системы характеризует возрастная категория: «26-30», «45+» и т.д.

Суествуют три способа взаимодействия пользователей с системой (события):

- Карточка отображена для пользователя (show);
- Пользователь кликнул на карточку (click);
- Пользователь просмотрел статью карточки (view).

Каждую неделю менеджеры по анализу контента задают одни и те же вопросы:

1. Сколько взаимодействий пользователей с карточками происходит в системе с разбивкой по темам карточек?
2. Как много карточек генерируют источники с разными темами?
3. Как соотносятся темы карточек и темы источников?

Требуется автоматизировать процесс сбора необходимой информаци и разработать дашборд для менеджеров, основанный на пайплайне, который будет брать данные из таблиц, в которых хранятся сырые данные, трансформировать данные и укладывать их в агрегирующую таблицу.

> **Замечание:** Пайплайн будет разработан дата-инженерами.

Для решения задачи необходимо:

- подробно обсудить с менеджерами состав дашборда, его внешний вид и набор отображаемых данных;
- пообщаться с администраторами баз данных (БД) и выяснить, куда и как собираются нужные данные;
- решить с администраторами БД, где хранить агрегирующие таблицы;
- по результатам - составить техническое задание (ТЗ);
- разработать пайплайн (задача дата-инженеров);
- разработать дашборд.

Для проверки построенного дашборда необходимо с его помощью ответитьна следующие вопросы менеджеров:

- Сколько взаимодействий пользователей с карточками происходит в системе с разбивкой по темам карточек?
- Как много карточек генерируют источники с разными темами?
- Как соотносятся темы карточек и темы источников?

Составление технического задания на разработку дашборда

По результатам общения с менеджерами и администраторами баз данных задача может быть формализована в виде следующего краткого ТЗ:

1. ***Бизнес-задача:*** анализ взаимодействия пользователей с карточками Яндекс.Дзен.

1. ***Насколько часто предполагается пользоваться дашбордом:*** не реже, чем раз в неделю.

1. ***Кто будет основным пользователем дашборда:*** менеджеры по анализу контента.

1. ***Состав данных для дашборда:***
   - история событий по темам карточек (два графика - абсолютные числа и процентное соотношение);
   - разбивка событий по темам источников;
   - таблица соответствия тем источников темам карточек.

1. ***По каким параметрам данные должны группироваться:***
   - дата и время;
   - тема карточки;
   - тема источника;
   - возрастная группа.

1. ***Характер данных:***
   - история событий по темам карточек — абсолютные величины с разбивкой по минутам;
   - разбивка событий по темам источников — относительные величины (% событий);
   - соответствия тем источников темам карточек - абсолютные величины.

1. ***Важность:*** все графики имеют равную важность.

1. ***Источники данных для дашборда:*** сырые данные о событиях взаимодействия пользователей с карточками (таблица `log_raw`).

1. ***База данных, в которой будут храниться агрегированные данные:*** дополнительная агрегированная таблица `dash_visits`.

1. ***Частота обновления данных:*** один раз в сутки, в полночь по UTC.

1. Информация о том, какие графики должны отображаться и в каком порядке, какие элементы управления должны быть на дашборде, приведена в *макете дашборда*.

Макет дашборда:  Макет

Импорт библиотек и определение функций проекта

Импорт библиотек

In [1]:
```
# импортируем необходимые библиотеки
import pandas as pd
from sqlalchemy import create_engine
```

Функция обзора данных

In [2]:
```
# определение функции обзора данных
# ================================================
# на вход подаётся датафрейм df
# на выходе:
#     - 10 случайных строк df
#     - информация df.info()
#     - количество явных дубликатов в строках df
#     - процент пропусков данных в столбцах df
# ================================================
def data_observe(df):
    row_num = 5    # количество отображаемых строк таблицы

    print('Произвольные строки таблицы:')
    print('============================')
    if len(df) >= row_num:
        display(df.sample(row_num))
    else:
        display(df)

    print('\nИнформация о таблице:')
    print('=====================')
    df.info()

    print('\nКоличество явных дубликатов в таблице:')
    print('=======================================')
    print(df.duplicated().sum())

    print('\nПроцент пропусков в столбцах:')
    print('=============================')
    display(pd.DataFrame(
                round((df.isna().mean()*100),2), columns=['NaNs, %'])
                .sort_values(by='NaNs, %', ascending=False
            )
        .style.format('{:.2f}')
        .background_gradient('coolwarm')
    )
```

Подключение к БД и выгрузка таблицы агрегированных данных

Дата-инженеры разработали pipeline, которфй один раз в сутки, в полночь по UTC, агрегирует сырые данные по событиям и записывает их в таблицу `dash_visits` в БД `zen`.

Выгрузим эту таблицу и изучим собранные данные.

Загрузка данных из БД

Для получения данных подключимся к PostgreSQL базе данных:

In [3]:
```
# зададим параметры подключения
db_config = {'user': 'praktikum_student', # имя пользователя
             'pwd': 'Sdf4$2;d-d30pp', # пароль
             'host': 'rc1b-wcoijxj3yxfsf3fs.mdb.yandexcloud.net',
             'port': 6432, # порт подключения
             'db': 'data-analyst-zen-project-db'} # название базы данных

# сформируем строку подключения
connection_string = 'postgresql://{}:{}@{}:{}/{}'.format(db_config['user'],
                                                         db_config['pwd'],
                                                         db_config['host'],
                                                         db_config['port'],
                                                         db_config['db'])

# выполним подключение
engine = create_engine(connection_string)
```

Зададим и выполним SQL-запрос:

In [4]:
```
query = '''SELECT * FROM dash_visits'''

dash_visits = pd.io.sql.read_sql(query, con = engine)
```

Обзор данных

Данные загружены, изучим их:

In [5]:
```
data_observe(dash_visits)
```

```
Произвольные строки таблицы:
============================
```

	record_id	item_topic	source_topic	age_segment	dt	visits
10488	1051085	История	Сад и дача	41-45	2019-09-24 19:00:00	1
15463	1056060	Общество	Психология	31-35	2019-09-24 18:57:00	37
26781	1067378	Скандалы	Знаменитости	36-40	2019-09-24 18:58:00	62
28175	1068772	Туризм	Полезные советы	31-35	2019-09-24 18:55:00	8
6906	1047503	Интересные факты	Дети	18-25	2019-09-24 18:54:00	4

```
Информация о таблице:
=====================
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30745 entries, 0 to 30744
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   record_id     30745 non-null  int64
 1   item_topic    30745 non-null  object
 2   source_topic  30745 non-null  object
 3   age_segment   30745 non-null  object
 4   dt            30745 non-null  datetime64[ns]
 5   visits        30745 non-null  int64
dtypes: datetime64[ns](1), int64(2), object(3)
memory usage: 1.4+ MB

Количество явных дубликатов в таблице:
======================================
0

Процент пропусков в столбцах:
=============================
```

	NaNs, %
record_id	0.00
item_topic	0.00
source_topic	0.00
age_segment	0.00
dt	0.00
visits	0.00

Таблица `dash_visits` состоит из 30745 строк, 6 столбцов. Типы данных в столбцах - object, int64, datetime64[ns].

Столбцы хранят следующую информацию:

- `record_id` — номер агрегированной записи;
- `item_topic` — тема карточки;
- `source_topic` — тема источника;
- `age_segment` — возрастная группа;
- `dt` — дата и время события;
- `visits` — количество событий на указанные дату и время.

Столбцы поименованы в хорошем стиле snake_case.

В таблице отсутствуют явные дубликаты и пропуски данных.

Изучим состав полей `item_topic`, `source_topic` и `age_segment`, а также охарактеризуем столбец `visits`:

In [6]:
```
print('min event date:', dash_visits.dt.min())
print('max event date:', dash_visits.dt.max())

print('\nitem_topic:', dash_visits.item_topic.sort_values().unique(),
      len(dash_visits.item_topic.sort_values().unique()))

print('\nsource_topic:', dash_visits.source_topic.sort_values().unique(),
      len(dash_visits.source_topic.sort_values().unique()))

print('\nage_segment:', dash_visits.age_segment.sort_values().unique(),
      len(dash_visits.age_segment.sort_values().unique()))

print('\nvisits:')
dash_visits.visits.describe()
```

```
min event date: 2019-09-24 18:28:00
max event date: 2019-09-24 19:00:00

item_topic: ['Деньги' 'Дети' 'Женская психология' 'Женщины' 'Здоровье' 'Знаменитости'
 'Интересные факты' 'Искусство' 'История' 'Красота' 'Культура' 'Наука'
 'Общество' 'Отношения' 'Подборки' 'Полезные советы' 'Психология'
 'Путешествия' 'Рассказы' 'Россия' 'Семья' 'Скандалы' 'Туризм' 'Шоу'
 'Юмор'] 25

source_topic: ['Авто' 'Деньги' 'Дети' 'Еда' 'Здоровье' 'Знаменитости' 'Интерьеры'
 'Искусство' 'История' 'Кино' 'Музыка' 'Одежда' 'Полезные советы'
 'Политика' 'Психология' 'Путешествия' 'Ремонт' 'Россия' 'Сад и дача'
 'Сделай сам' 'Семейные отношения' 'Семья' 'Спорт' 'Строительство'
 'Технологии' 'Финансы'] 26

age_segment: ['18-25' '26-30' '31-35' '36-40' '41-45' '45+'] 6

visits:
```

Out[6]:
```
count    30745.000000
mean        10.089673
std         19.727601
min          1.000000
25%          1.000000
50%          3.000000
75%         10.000000
max        371.000000
Name: visits, dtype: float64
```

В таблице представлены события за период 2019-09-24 18:28:00 - 2019-09-24 19:00:00.

В категориях `item_topic`, `source_topic` и `age_segment` неявные дубликаты не наблюдаются. Имеется:

- 25 различных тем карточек;
- 26 различных тем источников;
- 6 возрастных категорий.

Количество посещений:

- варьируется от 1 до 371,
- среднее - 10,
- медиана - 3.

Статистические характеристики числа посещений могут свидетельствовать о том, что для большинства сочетаний тем карточек и источников в большей степени характерна слабая посещаемость, но есть определённая доля тем с высокой популярностью.

Вывод: Данные собраны качественно и могут быть использованы для построения дашборда.

Сохранение данных в CSV-файл

Сохраним данные в файл `dash_visits.csv`, который затем будем использовать для построения дашборда в Tableau:

In [7]:
```
dash_visits.to_csv('dash_visits.csv')
```

Данные успешно выгружены. Можно переходить к построению дашборда в Tableau в соответствии с макетом.

Построение Дашборда в Tableau

В соответствии с макетом дашборда нам необходимо подготовить:

1. Три графика:
   - график истории взаимодействия "События по темам карточек" (абсолютные значения, stacked area chart);
   - график истории взаимодействия "% событий по темам карточек" (% от общего, stacked area chart);
   - график разбивки событий по темам источников "События по темам источников" (относительные значения, pie chart).
1. Таблицу соответствия тем карточек темам источников "Темы источников - темы карточек" (колонки - темы источников, строки - темы карточек, пересечение - абсолютное количество событий). Ячейки таблицы должны иметь окраску в зависимости от суммарного количества взаимодействий (highlight table).

Дашборд должен быть оснащён следующими фильтрами:

- по дате и времени;
- по темам карточек;
- по возррастным категориям.

Итоговая версия дашборда представлена на сайте Tableau Public.

Общий вывод исследования

В ходе исследования мы:

- по результатам бесед с контент-менеджерами и администраторами БД составили краткое ТЗ на разработку дашборда для анализа взаимодействияпользователей с карточками Яндекс.Дзен;
- выгрузили из БД агрегированные данные для построения дашборда и кратко изучили их;
- с использованием Tableau разработали в соответствии с макетом и опубликовали дашборд.

Итоговая версия дашборда представлена на сайте Tableau Public.

Проверим работу дашборда и ответим на вопросы менеджеров, сформулированные в контексте исследования:

1. *Сколько взаимодействий пользователей с карточками происходит в системе с разбивкой по темам карточек?*

На основании имеющихся агрегированных данных можно утверждать, что в наиболее полный период наблюдений (с 18:54 по 18:59 24 сентября 2019 года) общее количество событий превысило 60 тыс.

- При этом наибольшее количество событий (от 3897 до 4372) произошло с карточками по тематикам Наука, Отношения, Интересные факты и Общество (в порядке убывания популярности).
- Наименее популярными рубриками (менее 2000 событий) являются Красота, Туризм, Юмор, Путешествия, Психология, Женская психология, Шоу и Знаменитости.
- около 51% от общего количества событий в полный период наблюдений генерируют следующие темы: Наука, Отношения, Интересные факты, Общество, Подборки, Россия, Полезные советы, История и Семья (36% от имеющихся 25 рубрик).

1. *Сколько событий генерируют источники с разными темами?*

Наиболее популярными являются следующие 6 источников (около 23% от 26 источников), в совокупности обеспечивающие свыше 50% от общего количества событий:

- Семейные отношения (10,73%);
- Россия (9,6%);
- Полезные советы (8,84%);
- Путешествия (7,76%);

- Знаменитости (7,74%);
- Кино (6,49%).

Наименее популярны источники: Финансы, Музыка, Строительство, Технологии - менее 1% событий каждый.

1. *Как соотносятся темы карточек и темы источников?*

- Для источников-лидеров (Семейные отношения, Россия, Полезные советы, Путешествия, Знаменитости, Кино) генерируемый контент в большей степени релевантен теме источника.
- Для аутсайдеров (Финансы, Музыка, Строительство, Технологии) генерируемый контент в большинстве случаев не релевантен теме источника.

Рекомендации: Обратить внимание на корректность работы системы оценки релевантности контента.

Презентация к отчёту в формате PDF.