

Задача

Для того чтобы отслеживать, как выглядит путь клиента от знакомства с продуктом до его получения, и какие в нем есть проблемные места, аналитики часто строят воронки. Этот инструмент позволяет понять, сколько клиентов "терпит" на каждом из этапов процесса и сколько в итоге совершает целевое действие.

Перед вами схема данных проекта Тинькофф-нефть. Постройте воронку, которая показывала бы в динамике по неделям и месяцам, сколько клиентов посещает наш сайт, и какая доля из них регистрируется на нем | подает заявку на игру | по факту приходит на нее. Количество клиентов и конверсии необходимо изобразить на одном графике.

Какие выводы можно сделать исходя из построенной воронки, какие точки роста продукта вы видите?

Задание нужно выполнить в JupyterNotebook | Google Colab.

Как выполнять задание:

- 1) подключиться к базе данных с помощью Python в JupyterNotebook \ Google Colab
- 2) собрать dataset с необходимыми полями,
- 3) посчитать конверсии и сделать визуализацию на Python,
- 4) прикрепить файл с выполненным заданием

Автор: Потапов Глеб

In [1]:

```
!pip3 install SQLAlchemy
Requirement already satisfied: SQLAlchemy in d:\anaconda\lib\site-packages (1.4.32)
Requirement already satisfied: greenlet==0.4.17 in d:\anaconda\lib\site-packages (from SQLAlchemy) (1.1.1)
```

In [2]:

```
import numpy as np
import pandas as pd
from sqlalchemy import create_engine
```

In [3]:

```
# Устанавливаем соединение с базой данных, используем SQLAlchemy
db_url = "postgresql://student:2M0u034m558.169.52.106@5432/postgres"
conn = create_engine(db_url)
```

In [4]:

```
# Сделаем SQL запросы, где получим количество клиентов за каждый месяц месяц в 2022 и 2023 годах

df1_c = pd.read_sql("""
    SELECT DATE_PART('month', visit_dttm) AS month, COUNT(client_rk) AS clients
    FROM msu_analytics.client
    WHERE DATE_PART('year', visit_dttm) = 2022
    GROUP BY DATE_PART('month', visit_dttm)
    ORDER BY DATE_PART('month', visit_dttm)
""", conn)

df2_c = pd.read_sql("""
    SELECT DATE_PART('month', visit_dttm) AS month, COUNT(client_rk) AS clients
    FROM msu_analytics.client
    WHERE DATE_PART('year', visit_dttm) = 2023
    GROUP BY DATE_PART('month', visit_dttm)
    ORDER BY DATE_PART('month', visit_dttm)
""", conn)

# Объединим получившиеся два датафрейма в один
df_c = pd.concat([df1_c, df2_c], ignore_index = True)
print(df_c)
```

month	clients
0	9.0
1	10.0
2	11.0
3	12.0
4	1.0
5	2.0

In [5]:

```
# Сделаем SQL запросы, где получим количество регистраций за каждый месяц месяц в 2022 и 2023 годах

df1_r = pd.read_sql("""
    SELECT COUNT(account_rk) AS accounts, DATE_PART('month', registration_dttm) AS month
    FROM msu_analytics.account
    WHERE DATE_PART('year', registration_dttm) = 2022
    GROUP BY DATE_PART('month', registration_dttm)
    ORDER BY DATE_PART('month', registration_dttm)
""", conn)

df2_r = pd.read_sql("""
    SELECT COUNT(account_rk) AS accounts, DATE_PART('month', registration_dttm) AS month
    FROM msu_analytics.account
    WHERE DATE_PART('year', registration_dttm) = 2023
    GROUP BY DATE_PART('month', registration_dttm)
    ORDER BY DATE_PART('month', registration_dttm)
""", conn)

# Объединим получившиеся два датафрейма в один
df_r = pd.concat([df1_r, df2_r], ignore_index = True)
print(df_r)
```

accounts	month
0	186
1	219
2	195
3	209
4	215
5	9

In [6]:

```
# Сделаем SQL запросы, где получим количество заявок от разных клиентов на игру за каждый месяц месяц в 2022 и 2023 годах
# Не учитываем повторные заявки клиентов. Один клиент - одна заявка

df1_a = pd.read_sql("""
    SELECT COUNT(DISTINCT account_rk) AS clients, DATE_PART('month', application_dttm) AS month
    FROM msu_analytics.application
    WHERE DATE_PART('year', application_dttm) = 2022
    GROUP BY DATE_PART('month', application_dttm)
    ORDER BY DATE_PART('month', application_dttm)
""", conn)

df2_a = pd.read_sql("""
    SELECT COUNT(DISTINCT account_rk) AS clients, DATE_PART('month', application_dttm) AS month
    FROM msu_analytics.application
    WHERE DATE_PART('year', application_dttm) = 2023
    GROUP BY DATE_PART('month', application_dttm)
    ORDER BY DATE_PART('month', application_dttm)
""", conn)

# Объединим получившиеся два датафрейма в один
df_a = pd.concat([df1_a, df2_a], ignore_index = True)
print(df_a)
```

clients	month
0	40
1	68
2	66
3	125
4	77
5	3

In [7]:

```
# Теперь посмотрим, сколько разных клиентов пришли на игры
# Ориентироваться будем на даты заявки

df1_g = pd.read_sql("""
    SELECT COUNT(DISTINCT account_rk) AS clients, DATE_PART('month', application_dttm) AS month
    FROM msu_analytics.application
    INNER JOIN msu_analytics.game
    ON msu_analytics.application.game_rk = msu_analytics.game.game_rk
    WHERE (DATE_PART('year', application_dttm) = 2022) AND (msu_analytics.game.game_fig = 1)
    GROUP BY DATE_PART('month', application_dttm)
    ORDER BY DATE_PART('month', application_dttm)
""", conn)

df2_g = pd.read_sql("""
    SELECT COUNT(DISTINCT account_rk) AS clients, DATE_PART('month', application_dttm) AS month
    FROM msu_analytics.application
    INNER JOIN msu_analytics.game
    ON msu_analytics.application.game_rk = msu_analytics.game.game_rk
    WHERE (DATE_PART('year', application_dttm) = 2023) AND (msu_analytics.game.game_fig = 1)
    GROUP BY DATE_PART('month', application_dttm)
    ORDER BY DATE_PART('month', application_dttm)
""", conn)

# Объединим получившиеся два датафрейма в один
df_g = pd.concat([df1_g, df2_g], ignore_index = True)
print(df_g)
```

Ответим что в базе данных последний дата это 2 февраля, поэтому за февраль всего 3 заявки, а игр пока не состоялось.

clients	month
0	29
1	59
2	54
3	63
4	24

In [8]:

```
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [9]:

```
month = ["Sep_2022", "Oct_2022", "Nov_2022", "Dec_2022", "Jan_2023", "Feb_2023"]
```

In [10]:

```
plt.bar(
    month,
    df_c["clients"]
)

plt.bar(
    month,
    df_r["accounts"]
)

plt.bar(
    month,
    df_a["clients"]
)

plt.bar(
    month,
    df_g["clients"]
)
```

Out[10]:

```
<BarContainer object of 6 artists>
```

In [11]:

```
cls = df_c["clients"].to_numpy()
acc = df_r["accounts"].to_numpy()
apl = df_a["clients"].to_numpy()
gms = df_g["clients"].to_numpy()
```

In [12]:

```
# Создадим df из клиентов на каждом этапе
array = ([ cls,
            acc,
            apl,
            gms
          ])

columns = month
index = ["Clients", "Accounts", "Applications", "Games"]

df = pd.DataFrame(array, index, columns)
df["Feb_2022"]["fillna"] = True
df
df.print(df)
```

Out[12]:

	Sep_2022	Oct_2022	Nov_2022	Dec_2022	Jan_2023	Feb_2023
Clients	778	843	786	805	852	32.0
Accounts	186	219	195	200	215	9.0
Applications	40	68	66	65	77	3.0
Games	29	59	54	45	24	0.0

In [13]:

```
# Вставим в таблицу значения конверсий
df.loc["Clients"]
df.loc["Conv_Acc"] = round((df.loc["Accounts"] / df.loc["Clients"]) * 100, 2)
df.loc["Conv_Apl"] = round((df.loc["Applications"] / df.loc["Accounts"]) * 100, 2)
df.loc["Conv_Apl_Gms"] = round((df.loc["Games"] / df.loc["Applications"]) * 100, 2)
df.loc["Conv_Clients_Games"] = round((df.loc["Games"] / df.loc["Clients"]) * 100, 2)
print(df)
```

	Sep_2022	Oct_2022	Nov_2022	Dec_2022	Jan_2023	Feb_2023
Clients	778.00	843.00	786.00	805.00	852.00	32.00
Accounts	186.00	219.00	195.00	200.00	215.00	9.00
Applications	40.00	68.00	66.00	65.00	77.00	3.00
Games	29.00	59.00	54.00	45.00	24.00	0.00
Conv_Cls_Acc	23.91	25.88	24.03	24.84	25.23	28.12
Conv_Acc_Apl	21.53	31.85	33.85	32.59	35.81	33.33
Conv_Apl_Gms	72.50	86.76	81.82	69.23	31.17	6.00
Conv_Clients_Games	3.73	7.00	6.87	5.59	2.82	6.00

In [14]:

```
# Теперь займемся визуализацией конверсий
# Начнем с конверсии из посетителей сайта в зарегистрированного пользователя
plt.plot(df.loc["Conv_Cls_Acc"])
```

Out[14]:

```
<matplotlib.lines.Line2D at 0x1acaf81a5c>
```

In [15]:

```
# Конверсия из зарегистрированного пользователя в заявку на игру
plt.plot(df.loc["Conv_Acc_Apl"])
```

Out[15]:

```
<matplotlib.lines.Line2D at 0x1lac02fe080>
```

In [16]:

```
# Конверсия из заявки на игру в посещение своей игры
plt.plot(df.loc["Conv_Apl_Gms"])
```

Out[16]:

```
<matplotlib.lines.Line2D at 0x1lac094c4c0>
```

In [17]:

```
# Общая конверсия
plt.plot(df.loc["Conv_Clients_Games"])
```

Out[17]:

```
<matplotlib.lines.Line2D at 0x1lac0f60a00>
```

Теперь сделаем тоже самое, только по неделям

In [18]:

```
# Сделаем SQL запросы, где получим количество клиентов за каждую неделю месяц в 2022 и 2023 годах

df1_c = pd.read_sql("""
    SELECT DATE_PART('week', visit_dttm) AS week, COUNT(client_rk) AS clients
    FROM msu_analytics.client
    WHERE DATE_PART('week', visit_dttm) > 30
    GROUP BY DATE_PART('week', visit_dttm)
    ORDER BY DATE_PART('week', visit_dttm)
""", conn)

df2_c = pd.read_sql("""
    SELECT DATE_PART('week', visit_dttm) AS week, COUNT(client_rk) AS clients
    FROM msu_analytics.client
    WHERE DATE_PART('week', visit_dttm) < 30
    GROUP BY DATE_PART('week', visit_dttm)
    ORDER BY DATE_PART('week', visit_dttm)
""", conn)

# Объединим получившиеся два датафрейма в один
df_c = pd.concat([df1_c, df2_c], ignore_index = True)
print(df_c)
```

week	clients
0	35.0
1	36.0
2	37.0
3	38.0
4	39.0
5	40.0
6	41.0
7	42.0
8	43.0
9	44.0
10	45.0
11	46.0
12	47.0
13	48.0
14	49.0
15	50.0
16	51.0
17	52.0
18	53.0
19	54.0
20	55.0
21	56.0
22	57.0

In [19]:

```
# Сделаем SQL запросы, где получим количество заявок на игру за каждую неделю месяц в 2022 и 2023 годах
# Не учитываем повторные заявки клиентов. Один клиент - одна заявка

df1_a = pd.read_sql("""
    SELECT COUNT(DISTINCT account_rk) AS clients, DATE_PART('week', application_dttm) AS week
    FROM msu_analytics.application
    WHERE DATE_PART('week', application_dttm) > 30
    GROUP BY DATE_PART('week', application_dttm)
    ORDER BY DATE_PART('week', application_dttm)
""", conn)

df2_a = pd.read_sql("""
    SELECT COUNT(DISTINCT account_rk) AS clients, DATE_PART('week', application_dttm) AS week
    FROM msu_analytics.application
    WHERE DATE_PART('week', application_dttm) < 30
    GROUP BY DATE_PART('week', application_dttm)
    ORDER BY DATE_PART('week', application_dttm)
""", conn)

# Объединим получившиеся два датафрейма в один
df_a = pd.concat([df1_a, df2_a], ignore_index = True)
print(df_a)
```

clients	week
0	7
1	12
2	21
3	16
4	23
5	24
6	24
7	24
8	24
9	24
10	24
11	24
12	24
13	24
14	24
15	24
16	24
17	24
18	24
19	24
20	24
21	24
22	24

In [20]:

```
# Сделаем SQL запросы, где получим количество заявок от разных клиентов на игру за каждую неделю в 2022 и 2023 годах
# Не учитываем повторные заявки клиентов. Один клиент - одна заявка

df1_g = pd.read_sql("""
    SELECT COUNT(DISTINCT account_rk) AS clients, DATE_PART('week', application_dttm) AS week
    FROM msu_analytics.application
    INNER JOIN msu_analytics.game
    ON msu_analytics.application.game_rk = msu_analytics.game.game_rk
    WHERE (DATE_PART('week', application_dttm) > 30) AND (msu_analytics.game.game_fig = 1)
    GROUP BY DATE_PART('week', application_dttm)
    ORDER BY DATE_PART('week', application_dttm)
""", conn)

df2_g = pd.read_sql("""
    SELECT COUNT(DISTINCT account_rk) AS clients, DATE_PART('week', application_dttm) AS week
    FROM msu_analytics.application
    INNER JOIN msu_analytics.game
    ON msu_analytics.application.game_rk = msu_analytics.game.game_rk
    WHERE (DATE_PART('week', application_dttm) < 30) AND (msu_analytics.game.game_fig = 1)
    GROUP BY DATE_PART('week', application_dttm)
    ORDER BY DATE_PART('week', application_dttm)
""", conn)

# Объединим получившиеся два датафрейма в один
df_g = pd.concat([df1_g, df2_g], ignore_index = True)
print(df_g)
```

clients	week
0	3
1	9
2	12
3	12
4	12
5	29
6	22
7	8
8	16
9	14
10	54
11	46
12	47
13	48
14	49
15	50
16	51
17	52
18	53
19	54
20	55
21	56
22	57

In [21]:

```
# Сделаем SQL запросы, где получим количество заявок от разных клиентов на игру за каждую неделю в 2022 и 2023 годах
# Не учитываем повторные заявки клиентов. Один клиент - одна заявка

df1_r = pd.read_sql("""
    SELECT DATE_PART('week', registration_dttm) AS week, COUNT(account_rk) AS clients
    FROM msu_analytics.account
    WHERE DATE_PART('week', registration_dttm) > 30
    GROUP BY DATE_PART('week', registration_dttm)
    ORDER BY DATE_PART('week', registration_dttm)
""", conn)

df2_r = pd.read_sql("""
    SELECT DATE_PART('week', registration_dttm) AS week, COUNT(account_rk) AS clients
    FROM msu_analytics.account
    WHERE DATE_PART('week', registration_dttm) < 30
    GROUP BY DATE_PART('week', registration_dttm)
    ORDER BY DATE_PART('week', registration_dttm)
""", conn)

# Объединим получившиеся два датафрейма в один
df_r = pd.concat([df1_r, df2_r], ignore_index = True)
print(df_r)
```

week	clients
0	35.0
1	36.0
2	37.0
3	38.0
4	39.0
5	40.0
6	41.0
7	42.0
8	43.0
9	44.0
10	45.0
11	46.0
12	47.0
13	48.0
14	49.0
15	50.0
16	51.0
17	52.0
18	53.0
19	54.0
20	55.0
21	56.0
22	57.0

In [22]:

```
# Сделаем SQL запросы, где получим количество заявок от разных клиентов на игру за каждую неделю в 2022 и 2023 годах
# Не учитываем повторные заявки клиентов. Один клиент - одна заявка

df1_a = pd.read_sql("""
    SELECT COUNT(DISTINCT account_rk) AS clients, DATE_PART('week', application_dttm) AS week
    FROM msu_analytics.application
    WHERE DATE_PART('week', application_dttm) > 30
    GROUP BY DATE_PART('week', application_dttm)
    ORDER BY DATE_PART('week', application_dttm)
""", conn)

df2_a = pd.read_sql("""
    SELECT COUNT(DISTINCT account_rk) AS clients, DATE_PART('week', application_dttm) AS week
    FROM msu_analytics.application
    WHERE DATE_PART('week', application_dttm) < 30
    GROUP BY DATE_PART('week', application_dttm)
    ORDER BY DATE_PART('week', application_dttm)
""", conn)

# Объединим получившиеся два датафрейма в один
df_a = pd.concat([df1_a, df2_a], ignore_index = True)
print(df_a)
```

clients	week
0	7
1	12
2	21
3	16
4	23
5	24
6	24
7	24
8	24
9	24
10	24
11	24
12	24
13	24
14	24
15	24
16	24
17	24
18	24
19	24
20	24
21	24
22	24

In [23]:

```
# Сделаем SQL запросы, где получим количество заявок от разных клиентов на игру за каждую неделю в 2022 и 2023 годах
# Не учитываем повторные заявки клиентов. Один клиент - одна заявка

df1_g = pd.read_sql("""
    SELECT COUNT(DISTINCT account_rk) AS clients, DATE_PART('week', application_dttm) AS week
    FROM msu_analytics.application
    INNER JOIN msu_analytics.game
    ON msu_analytics.application.game_rk = msu_analytics.game.game_rk
    WHERE (DATE_PART('week', application_dttm) > 30) AND (msu_analytics.game.game_fig = 1)
    GROUP BY DATE_PART('week', application_dttm)
    ORDER BY DATE_PART('week', application_dttm)
""", conn)

df2_g = pd.read_sql("""
    SELECT COUNT(DISTINCT account_rk) AS clients, DATE_PART('week', application_dttm) AS week
    FROM msu_analytics.application
    INNER JOIN msu_analytics.game
    ON msu_analytics.application.game_rk = msu_analytics.game.game_rk
    WHERE (DATE_PART('week', application_dttm) < 30) AND (msu_analytics.game.game_fig = 1)
    GROUP BY DATE_PART('week', application_dttm)
    ORDER BY DATE_PART('week', application_dttm)
""", conn)

# Объединим получившиеся два датафрейма в один
df_g = pd.concat([df1_g, df2_g], ignore_index = True)
print(df_g)
```

clients	week
0	3
1	9
2	12
3	12
4	12
5	29
6	22
7	8
8	16
9	14
10	54
11	46
12	47
13	48
14	49
15	50
16	51
17	52
18	53
19	54
20	55
21	56
22	57

Out[24]:

```
array([[35., 36., 37., 38., 39., 40., 41., 42., 43., 44., 45., 46., 47.,
        48., 49., 50., 51., 52., 53., 54., 55., 56., 57.]])
```

In [25]:

```
cls = df_c["clients"].to_numpy()
acc = df_r["clients"].to_numpy()
apl = df_a["clients"].to_numpy()
gms = df_g["clients"].to_numpy()
```

In [26]:

```
# Создадим df из клиентов на каждом этапе
array = ([ cls,
            acc,
            apl,
            gms
          ])

index = ["Clients", "Accounts", "Applications", "Games"]

df = pd.DataFrame(array, index, weeks)
df.fillna(0, inplace = True)
df
```

Out[24]:

	35.0	36.0	37.0	38.0	40.0	41.0	42.0	43.0	44.0	...	48.0	49.0	50.0	51.0	52.0	53.0	54.0	55.0	56.0	57.0	
Clients	111	114	107	181	195	204	190	171	189	197	...	177	168	196	192	192	201	205	181	182	0
Accounts	22	38	42	46	56	59	46	42	48	46	...	45	44	52	46	44	55	54	42	40	26.0
Applications	7	12	21	16	23	24	13	22	20	...	17	16	22	19	20	21	28	17	7.0	0.0	
Games	3	9	12	12	17	20	22	8	16	14	...	12	10	11	9	3	8	7	5	0.0	0.0

4 rows x 23 columns

In [25]:

```
# Вставим в таблицу значения конверсий
df.loc["Clients"]
df.loc["Conv_Acc"] = round((df.loc["Accounts"] / df.loc["Clients"]) * 100, 2)
df.loc["Conv_Apl"] = round((df.loc["Applications"] / df.loc["Accounts"]) * 100, 2)
df.loc["Conv_Apl_Gms"] = round((df.loc["Games"] / df.loc["Applications"]) * 100, 2)
df.loc["Conv_Clients_Games"] = round((df.loc["Games"] / df.loc["Clients"]) * 100, 2)
print(df)
```

Out[25]:

	35.0	36.0	37.0	38.0	40.0	41.0	42.0	43.0	44.0	...	48.0	49.0	50.0	51.0	52.0	53.0	54.0	55.0	56.0	57.0	
Clients	111.00	114.00	107.00	181.00	195.00	204.00	190.00	171.00	189.00	197.00	...	177.00	168.00	196.00	192.00	192.00	201.00	205.00	181.00	182.00	0.0
Accounts	22.00	38.00	42.00	46.00	56.00	59.00	46.00	42.00	48.00	46.00	...	45.00	44.00	52.00	46.00	44.00	55.00	54.00	42.00	40.00	26.0
Applications	7.00	12.00	21.00	16.00	23.00	24.00	13.00	22.00	20.00	...	17.00	16.00	22.00	19.00	20.00	21.00	28.00	17.00	7.00	0.0	
Games	3.00	9.00	12.00	12.00	17.00	20.00	22.00	8.00	16.00	14.00	...	12.00	10.00	11.00	9.00	3.00	8.00	7.00	5.00	0.00	0.00
Conv_Cls_Acc	19.82	33.00	26.65	25.15	25.41	28.72	26.92	24.21	24.96	25.40	23.95	...	25.42	26.19	26.53	23.90	22.92	27.36	26.34	23.90	32.1
Conv_Acc_Apl	53.63	71.50	50.00	74.78	41.07	40.68	52.17	30.95	45.83	43.48	...	37.78	36.36	42.31	43.30	45.45	38.18	51.85	40.81	17.50	0.0
Conv_Apl_Gms	42.86	75.00	57.14	75.00	37.81	63.33	91.67	61.54	72.73	70.00	...	70.59	62.50	60.00	47.37	15.00	38.18	25.00	29.41	0.00	NaN
Conv_Clients_Games	2.70	4.89	7.19	6.63	8.72	9.00	11.58	4.68	8.47	7.11	...	6.78	5.95	5.61	4.69	1.56	3.98	3.41	2.78	0.00	0.0

8 rows x 23 columns