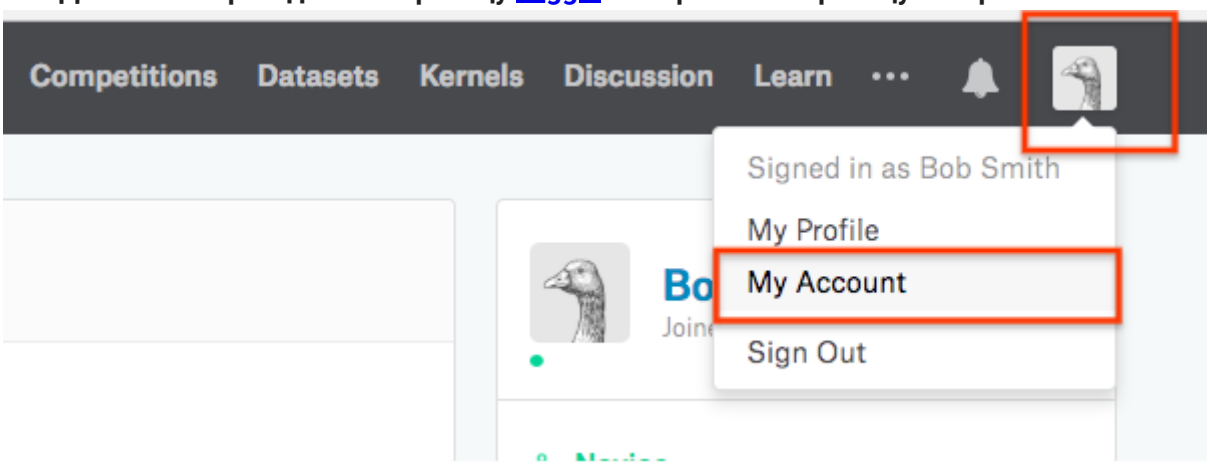
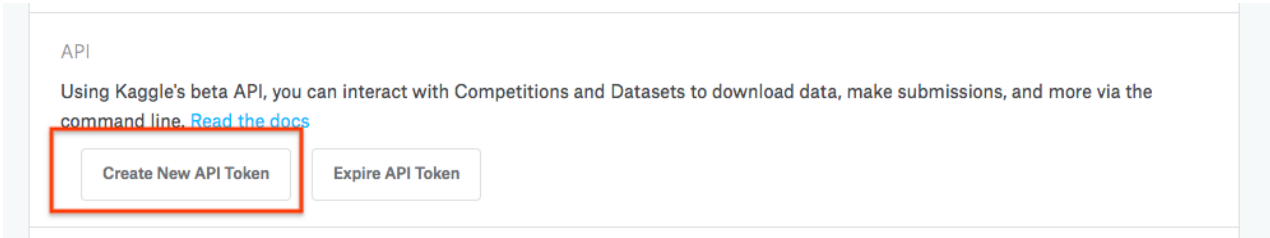
Установка окружения

1. Скачиваем keggle.json

1.1. Для этого переходим на страницу [kaggle] и открываем страницу настроек пользователя.



1.2. Затем прокручиваем вниз до раздела доступа к API и нажмаем «Создать», чтобы загрузить ключ API.



Загрузится файл kaggle.json на ваш компьютер. Мы будем использовать этот файл в Colab для доступа к Kaggle.

2. Устанавливаем API-интерфейс kaggle

2.1. Убедимся что файл kaggle.json присутствует.

Введем в cmd:

```
!ls -lha kaggle.json
```

Появится строка:

```
-rw-r--r--@ 1 mybook  staff    64B 21 мар 12:27 kaggle.json
```

2.2. Клиент API Kaggle ожидает, что этот файл будет в ~/.kaggle

Введем:

```
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
```

2.3. Это изменение разрешений позволяет избежать предупреждения при запуске инструмента Kaggle.

Введем:

```
!chmod 600 ~/.kaggle/kaggle.json
```

2.4. Теперь можем получить доступ к наборам данных.

Введем:

```
!kaggle datasets list
```

Появится набор данных.

3. Загружаем dataset в colab

3.1. Заходим в colab

3.2. Скачиваем любой dataset из kaggle

3.3. Прописываем данный код:

```
from google.colab import files
files.upload()

#Выбираем скаченный dataset:
```

Выбрать файлы | Файл не выбран — Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving BlackFriday.csv to BlackFriday (1).csv
{'BlackFriday.csv': b'User_ID,Product_ID,Gender,Age,Occupation,City_Categor
```

Ход выполнения задания

Подключение библиотек

```
from datetime import datetime
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
```

```
#Настроим отображение графиков [3,4]:
%matplotlib inline
```

```
#Задаем стиль
sns.set(style="ticks")
```

```
#Установка форматов участков для сохранения PNG с высоким разрешением
from IPython.display import set_matplotlib_formats
set_matplotlib_formats("retina")
```

```
#Зададим ширину текстового представления данных, чтобы в дальнейшем текст в отчёте влезал на
pd.set_option("display.width", 70)
```

```
#Загрузим непосредственно данные:
data = pd.read_csv("BlackFriday.csv")
```

Текстовое описание выбранного набора данных.

В качестве набора данных мы будем использовать набор данных "Черной пятницы".

Данный датасет содержит следующие колонки:

- User_ID - уникальный идентификатор пользователя.
- Product_ID - уникальный идентификатор продукта.
- Gender - пол лица.
- Age - возрастная группа лица.
- Occupation - должность лица.
- City_Category - категория города, где проживает лицо.
- Stay_In_Current_City_Years - срок проживания лица в городе.
- Marital_Status - 0, не женат и 1 в противном случае.
- Product_Category_1 - первая категория продукта.
- Product_Category_2 - вторая категория продукта.
- Product_Category_3 - третья категория продукта.
- Purchase - сумма покупки.

Основные характеристики датасета.

1. Выведем данные

```
#Проверим типы данных:
data.dtypes
```

```
User_ID                      int64
Product_ID                  object
Gender                      object
```

```
#Посмотрим на данные в данном наборе данных:
data.head()
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Curr
0	1000001	P00069042	F	0-17	10	A	
1	1000001	P00248942	F	0-17	10	A	
2	1000001	P00087842	F	0-17	10	A	
3	1000001	P00085442	F	0-17	10	A	
4	1000002	P00285442	M	55+	16	C	

```
#Проверим основные статистические характеристики набора данных:
describe = data.describe()

#Добавим параметр "уникальность"
describe.loc['#unique'] = data.nunique()
display(describe)
```

	User_ID	Occupation	Marital_Status	Product_Category_1	Produc
count	5.375770e+05	537577.00000	537577.000000	537577.000000	
mean	1.002992e+06	8.08271	0.408797	5.295546	
std	1.714393e+03	6.52412	0.491612	3.750701	
min	1.000001e+06	0.00000	0.000000	1.000000	
25%	1.001495e+06	2.00000	0.000000	1.000000	
50%	1.003031e+06	7.00000	0.000000	5.000000	
75%	1.004417e+06	14.00000	1.000000	8.000000	
max	1.006040e+06	20.00000	1.000000	18.000000	
#unique	5.891000e+03	21.00000	2.000000	18.000000	

---

2. Разберемся в наборе данных датасета:

```
purchase_desc = data['Purchase'].describe()
purchase_desc.drop(['count', 'std'], inplace=True)
purchase_desc.loc['sum'] = data['Purchase'].sum()
purchase_desc.loc['mean_by_user'] = data['Purchase'].sum() / data['User_ID'].r
display(pd.DataFrame(purchase_desc).T)
```

	mean	min	25%	50%	75%	max	sum	mean_b

Из результатов можем сказать, что средняя сумма покупки составляет 9333, а средняя сумма всех лиц, около 851751

---

3. Найдем популярный продукт среди пола и возраста:

```
cat_describe = data[['Product_ID', 'Gender', 'Age', 'Occupation', 'City_Categ

#Добавим переменную процентного соотношения
cat_describe.loc['percent'] = 100*cat_describe.loc['freq'] / cat_describe.loc[
display(cat_describe)
```

	Product_ID	Gender	Age	Occupation	City_Category	Marital_Stat
count	537577	537577	537577	537577.00000	537577	537577.0000
unique	3623	2	7	21.00000	3	2.0000
top	P00265242	M	26-35	4.00000	B	0.0000
freq	1858	405380	214690	70862.00000	226493	317817.0000
percent	0.345625	75.4087	39.9366	13.18174	42.1322	59.1202

Из полученных данных видим, что:

- Продукт P00265242 является самым популярным продуктом.
- Большинство сделок совершались мужчинами.
- Возрастная группа с большинством сделок составила 26-35 лет.

Визуальное исследование датасета.

Оценим распределение целевого признака — суммы покупки

```
sns.distplot(data["Purchase"])
```

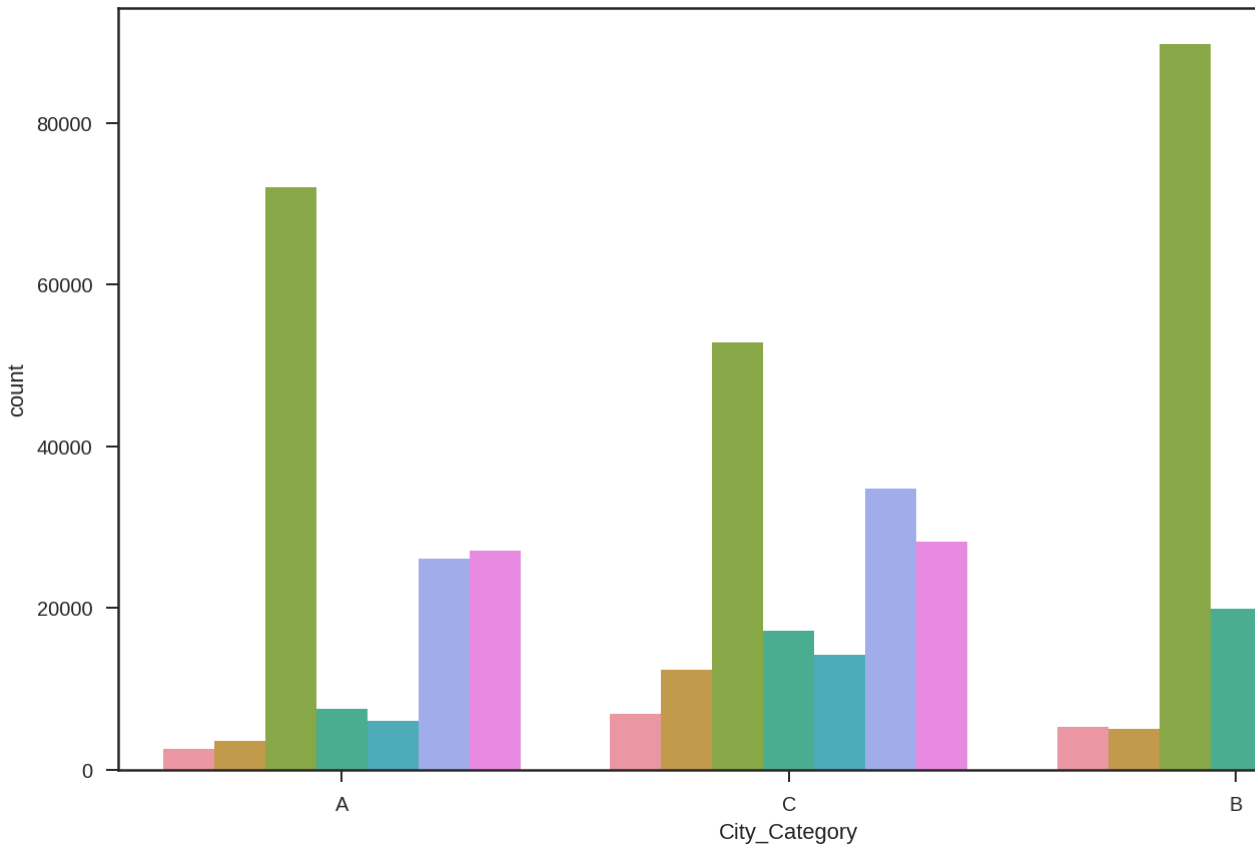
```
/usr/local/lib/python3.6/dist-packages/matplotlib/axes/_axes.py:6521: Matpl
The 'normed' kwarg was deprecated in Matplotlib 2.1 and will be removed in
  alternative="'density'", removal="3.1")
<matplotlib.axes._subplots.AxesSubplot at 0x7fdc8be9d0f0>
```

Видим что наибольшее количество покупок происходит в районе 7000

Сравним покупательную способность по областям:


```
fig1, ax1 = plt.subplots(figsize=(12,7))
sns.countplot(data['City_Category'],hue=data['Age'])
```

```
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:1468: FutureW
  stat_data = remove_na(group_data[hue_mask])
<matplotlib.axes._subplots.AxesSubplot at 0x7fdc8b024128>
```



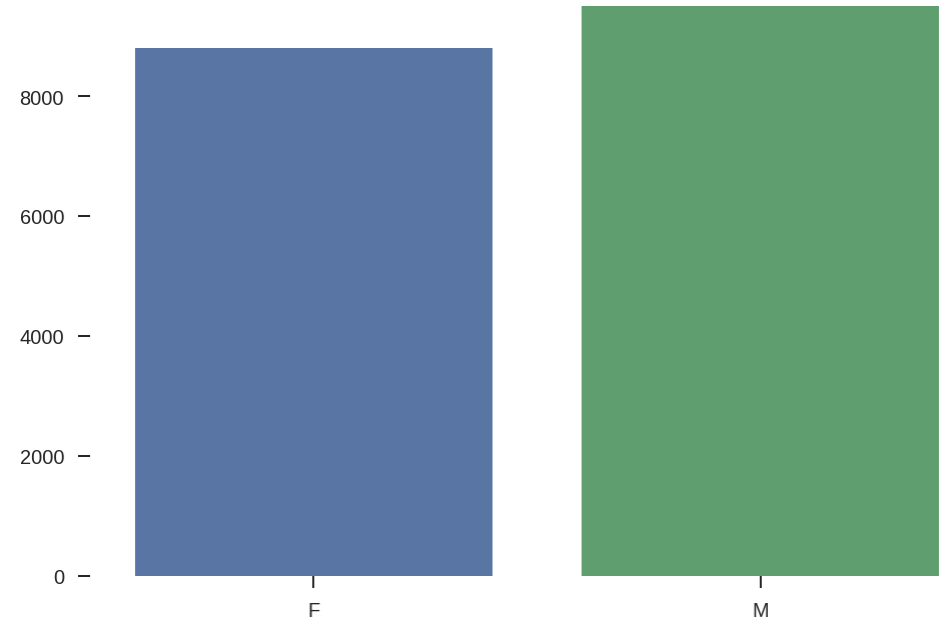
Как видим, люди из области B имеют большую покупательную способность, чем другие.

Сравним среднюю сумму покупки по полу:

```
gender_gb = data[['Gender', 'Purchase']].groupby('Gender', as_index=False).agg
sns.barplot(x='Gender', y='Purchase', data=gender_gb)
plt.ylabel('')
plt.xlabel('')
for spine in plt.gca().spines.values():
```

```
    spine.set_visible(False)
plt.title('Средняя сумма покупки по полу', size=14)
plt.show()
```

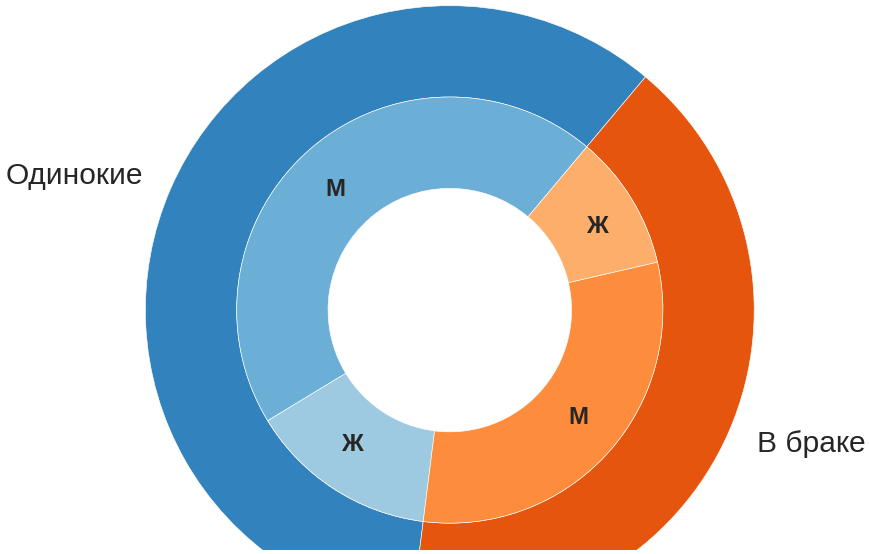
```
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:1428: FutureW
  stat_data = remove_na(group_data)
```



У мужчин транзакции были примерно в 3 раза выше, чем у женщин в black friday. У них также была пропорционально более высокая сумма покупки, что приводит к предположению, что нет существенной разницы между средними суммами покупки мужчин и женщин.

Определим, сотношение покупок одиноких и в браке мужчин/женщин:

```
out_vals = data.Marital_Status.value_counts()
in_vals = np.array([data[data.Marital_Status==x]['Gender'].value_counts() for

fig, ax = plt.subplots(figsize=(7, 7))

size = 0.3
cmap = plt.get_cmap("tab20c")
outer_colors = cmap(np.arange(2)*4)
inner_colors = cmap(np.array([1, 2, 5, 6]))

ax.pie(out_vals, radius=1, colors=outer_colors,
       wedgeprops=dict(width=size, edgecolor='w'), labels=['Одинокие', 'В браке'
       textprops={'fontsize': 15}, startangle=50)

ax.pie(in_vals, radius=1-size, colors=inner_colors,
       wedgeprops=dict(width=size, edgecolor='w'), labels=['М', 'Ж', 'М', 'Ж
       labeldistance=0.75, textprops={'fontsize': 12, 'weight': 'bold'}, start

ax.set(aspect="equal")
plt.title('Соотношение покупок по семейному положению и полу', fontsize=16)
plt.show()

```

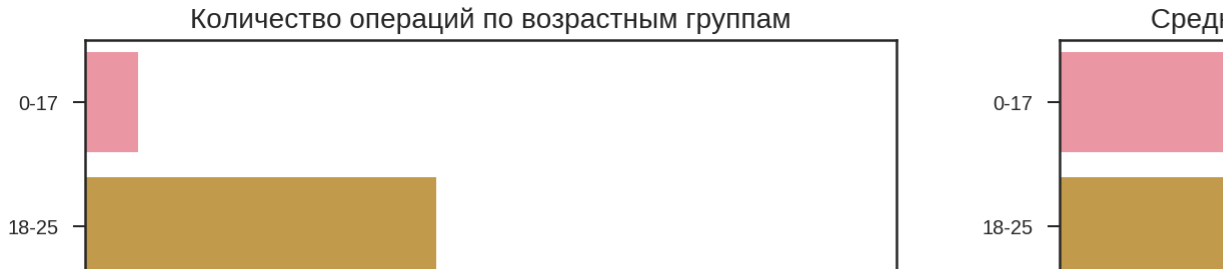


Одинокие люди приобрели больше товаров, чем люди в браке, и в обеих категориях мужчины, следуя общей схеме набора данных, приобрели больше, чем женщины.

Посмотрим, что мы можем наблюдать из возрастных групп

```
plt.figure(figsize=(16, 8))
plt.subplot(121)
sns.countplot(y='Age', data=data, order=sorted(data.Age.unique()))
plt.title('Количество операций по возрастным группам', size=14)
plt.xlabel('')
plt.ylabel('Возрастная группа', size=13)
plt.subplot(122)
age_gb = data[['Age', 'Purchase']].groupby('Age', as_index=False).agg('mean')
sns.barplot(y='Age', x='Purchase', data=age_gb, order=sorted(data.Age.unique()
plt.title('Средняя сумма покупки по возрастным группам', size=14)
plt.xlabel('')
plt.ylabel('')
plt.show()
```

```
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:1428: FutureW
  stat_data = remove_na(group_data)
```



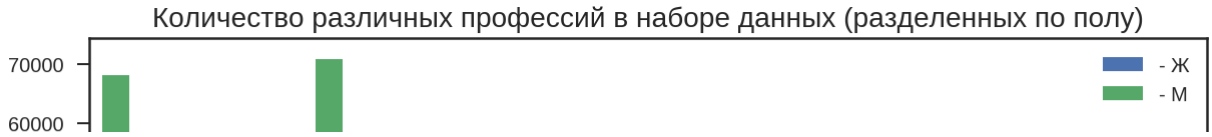
Люди в возрасте от 26 до 35 лет совершили больше всего покупок, и, как мы видели в отношении Пола, люди в разных возрастах также имеют почти одинаковую среднюю сумму покупки.

Проверим, какие продукты были наиболее популярны в каждой возрастной группе:

```
age_product_gb = data[['Age', 'Product_ID', 'Purchase']].groupby(['Age', 'Proc
age_product_gb.sort_values('count', inplace=True, ascending=False)
ages = sorted(data.Age.unique())
result = pd.DataFrame({
    x: list(age_product_gb.loc[x].index)[:5] for x in ages
}, index=['#{}'.format(x) for x in range(1,6)])
display(result)
```

	0-17	18-25	26-35	36-45	46-50	51-55	55+
#1	P00255842	P00265242	P00265242	P00025442	P00265242	P00265242	P00265242
#2	P00145042	P00112142	P00110742	P00110742	P00046742	P00025442	P00080342
#3	P00112142	P00110742	P00112142	P00265242	P00025442	P00110742	P00051442
#4	P00242742	P00237542	P00025442	P00112142	P00051442	P00059442	P00184942
#5	P00000142	P00046742	P00058042	P00057642	P00184942	P00010742	P00025442

```
men = data[data.Gender == 'M']['Occupation'].value_counts(sort=False)
women = data[data.Gender == 'F']['Occupation'].value_counts(sort=False)
pd.DataFrame({'- М': men, '- Ж': women}, index=range(0,21)).plot.bar(stacked=
plt.gcf().set_size_inches(10, 4)
plt.title("Количество различных профессий в наборе данных (разделенных по полу)", size=
plt.legend(loc="upper right")
plt.xlabel('Должности', size=13)
plt.ylabel('Колличество', size=13)
plt.show()
```



Из графика можно отметить то, что люди, занимающие 0, 4 и 7 должность, приобрели больше всего товаров в Черную пятницу.

Проверим, какие продукты больше всего интересовали людей разных профессий:

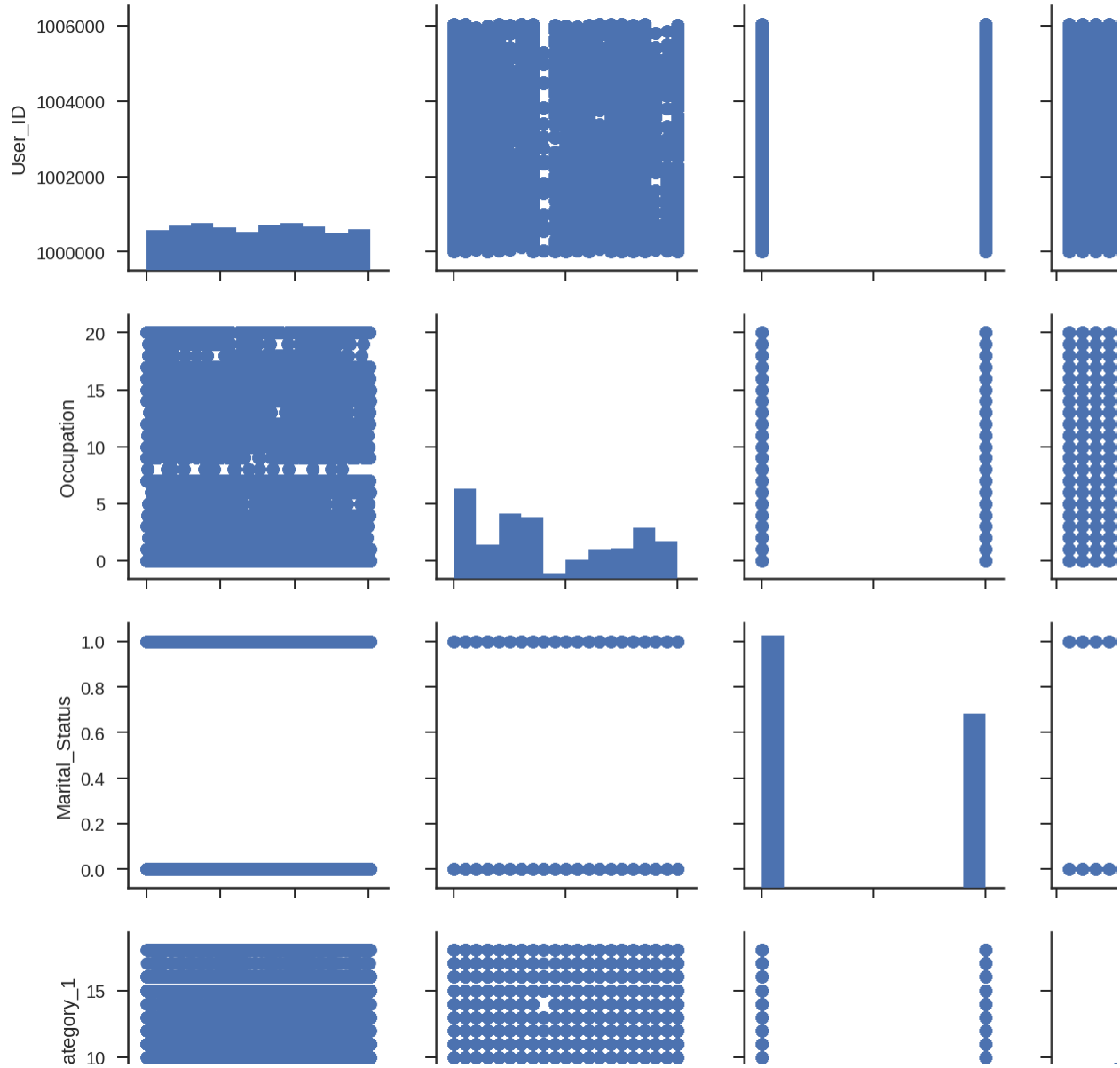
```
import random
color_mapping = {}
def random_color(val):
    if val in color_mapping.keys():
        color = color_mapping[val]
    else:
        r = lambda: random.randint(0,255)
        color = 'rgba({}, {}, {}, 0.4)'.format(r(), r(), r())
        color_mapping[val] = color
    return 'background-color: %s' % color

occ_product_gb = data[['Occupation', 'Product_ID', 'Purchase']].groupby(['Occu
occ_product_gb.sort_values('count', inplace=True, ascending=False)
result = pd.DataFrame({
    x: list(occ_product_gb.loc[x].index)[:5] for x in range(21)
}, index=['#{}'.format(x) for x in range(1,6)])
display(result.style.applymap(random_color))
```

	0	1	2	3	4	5	6	7
#1	P00265242	P00265242	P00265242	P00265242	P00265242	P00265242	P00265242	P00265242
#2	P00110742	P00220442	P00025442	P00117942	P00110742	P00114942	P00058042	P00110742
#3	P00025442	P00110742	P00058042	P00025442	P00112142	P00251242	P00110742	P00025442
#4	P00057642	P00025442	P00110842	P00110842	P00237542	P00110742	P00031042	P00112142
#5	P00112142	P00059442	P00059442	P00110742	P00025442	P00057642	P00255842	P00184942

Построим парные диаграммы по всем показателям по исходному набору данных:

```
sns.pairplot(data, plot_kws=dict(linewidth=0))
```

```
/usr/local/lib/python3.6/dist-packages/numpy/lib/function_base.py:780: Runt
  keep = (tmp_a >= first_edge)
/usr/local/lib/python3.6/dist-packages/numpy/lib/function_base.py:781: Runt
  keep &= (tmp_a <= last_edge)
<seaborn.axisgrid.PairGrid at 0x7fdc91018780>
```



Информация о корреляции признаков.

Построим корреляционную матрицу по всему набору данных:

```
1 data.corr()
```

	User_ID	Occupation	Marital_Status	Product_Category_1
User_ID	1.000000	-0.023024	0.018732	0.003687
Occupation	-0.023024	1.000000	0.024691	-0.008114

Визуализируем корреляционную матрицу с помощью тепловой карты:

```
1 sns.heatmap(data.corr(), annot=True, fmt=".4f")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdc8b466940>
```