Amazon: EDA & Recommendation System

В данном проекте мы будем производить исследовательский анализ данных о продажах Amazon.

Amazon - американская компания, крупнейшая в мире на рынках платформ электронной коммерции и публично-облачных вычислений по выручке и рыночной капитализации.

Источник данных

Датасет доступен на сайте Kaggle по этой ссылке.

Работа с данными

Первичная оценка данных

Прежде всего, проанализируем данные на предмет пустых/некорректных значений, очистим датасет от ненужных записей и отформатируем некоторые поля.

В частности, отформатируем **discounted_price**, **actual_price**, **discount percentage** - здесь нужно убрать лишние символы и перевести значения в тип данных float.

▼ Код

```
df['discounted_price'] = df['discounted_price'].str.replace('₹', '')
df['discounted_price'] = df['discounted_price'].str.replace(',', '')
df['discounted_price'] = df['discounted_price'].astype('float64')
```

```
df['actual_price'] = df['actual_price'].str.replace('₹', '')
df['actual_price'] = df['actual_price'].str.replace(',', '')
df['actual_price'] = df['actual_price'].astype('float64')
```

```
df['discount_percentage'] = df['discount_percentage'].str.replace('%', '')
df['discount_percentage'] = df['discount_percentage'].astype('float64')
df['discount_percentage'] = df['discount_percentage']/100
```

Исследование категорий

Наименования категорий имеют вид, который усложняет восприятие, поэтому мы выделим категорию и подкатегорию в отдельные колонки.

▼ Код

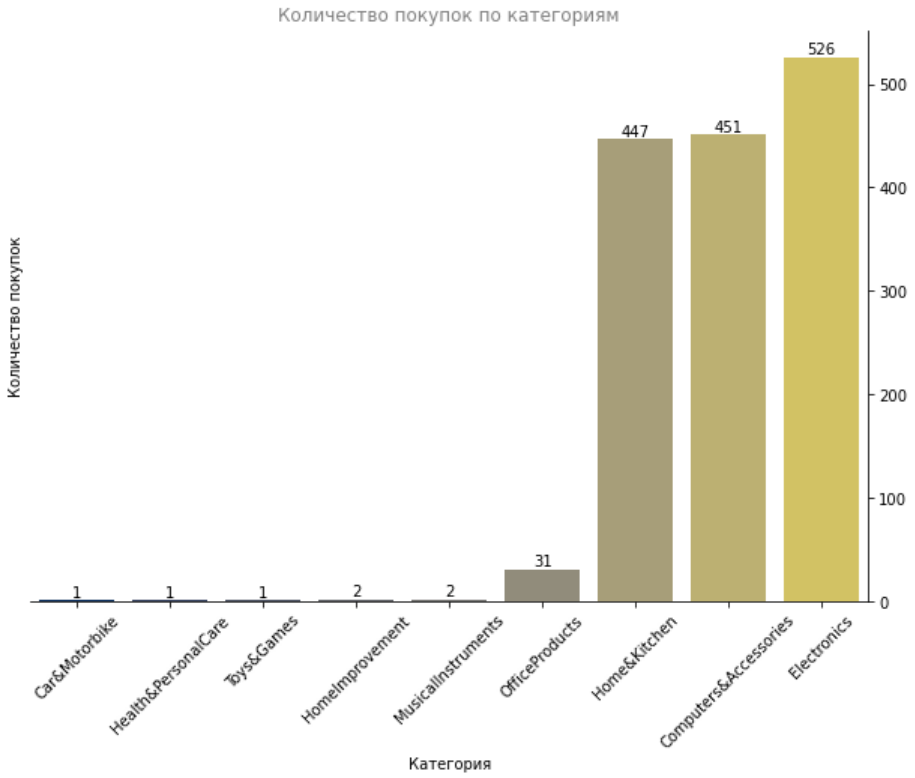
```
df['product_category'] = df['category'].str.split('|').str.get(0)
df['product_category'].unique()
```

```
df['product_subcategory'] = df['category'].str.split('|').str.get(1)
df['product_subcategory'].unique()
```

Наиболее популярная (по количеству покупок) категория - электроника:

▼ Код

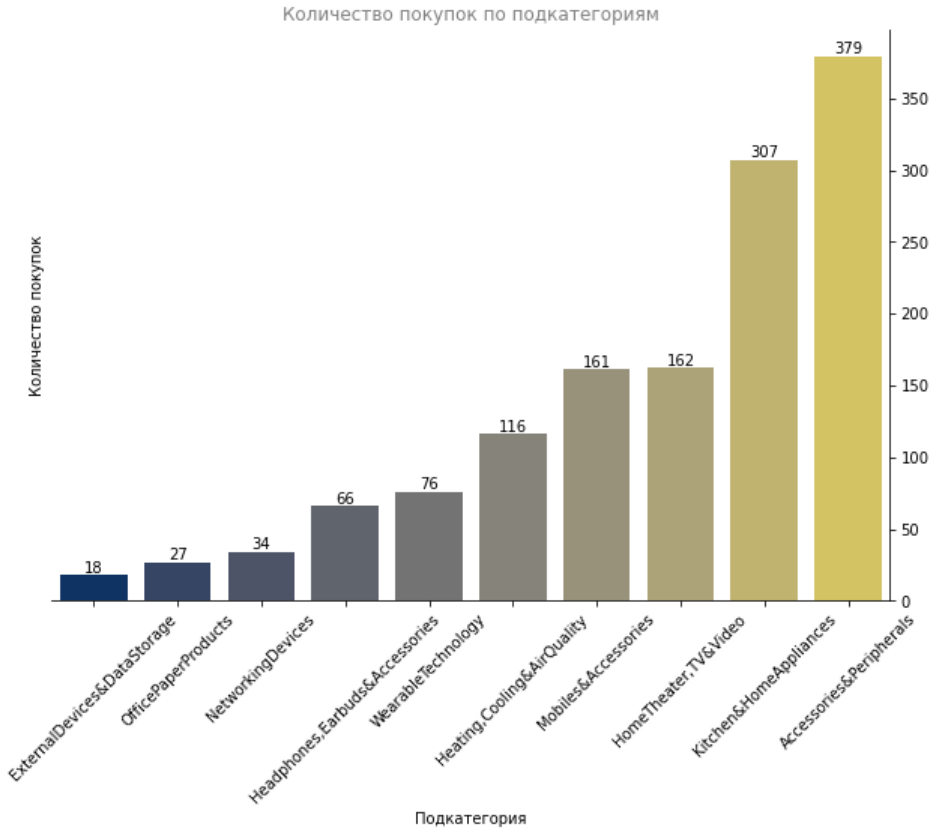
```
cat_count = df.groupby('product_category', as_index=False)['product_id'].coun
t().reset_index()

fig = plt.figure(figsize=(10,7))

ax = sns.barplot(data = cat_count, x = 'product_category', y = 'product_id',
palette='cividis', order=cat_count.sort_values('product_id').product_categor
y)

plt.xticks(rotation=45)

ax.set_xlabel('Категория')
ax.set_ylabel('Количество покупок')

sns.despine(left=True, right=False)

for i in ax.containers:
    ax.bar_label(i,)

plt.title('Количество покупок по категориям', color = 'gray')
```



Наиболее популярная подкатегория - это аксессуары и внешние устройства.

▼ Код

```
subcat_count = df.groupby('product_subcategory')['product_id'].count().sort_v
alues(ascending=False).reset_index().head(10)
fig = plt.figure(figsize=(10,7))

ax = sns.barplot(data = subcat_count, x = 'product_subcategory', y = 'product
_id', palette='cividis', order=subcat_count.sort_values('product_id').product
_subcategory)

plt.xticks(rotation=45)

ax.set_xlabel('Подкатегория')
ax.set_ylabel('Количество покупок')

sns.despine(left=True, right=False)

for i in ax.containers:
    ax.bar_label(i,)

plt.title('Количество покупок по подкатегориям', color = 'gray')
```



Создание системы рекомендаций

Для ускорения процесса обработки переводим user id в тип данных category

▼ Код

```
df['user_id_category'] = df['user_id'].astype('category')
```

Теперь отбираем 1000 клиентов и создаем словарь, по которому будем заменять длинные id на числа 0-1000

▼ Код

```
target_cust = df['user_id_category'].unique()[:1000]
cust_dict = {id: i for i, id in enumerate(target_cust)}
```

Создаем датафрейм с нашей 1000 клиентов, которые будем исследовать в рамках создания системы рекомендаций

▼ Код

```
rec = df[df['user_id_category'].isin(target_cust)].copy()
```

Заменяем оригинальный user id по словарю

▼ Код

```
def replace(x):
    n_id = 0
    try:
      n_id = cust_dict[x.user_id_category]
    except Exception as e:
        print(e)
        res = -1
    return n_id

rec['user_id'] = rec.apply(replace, axis=1)
rec.head()
```

Группируем данные для последующего создания матрицы

▼ Код

```
gr = rec.groupby(['user_id', 'product_name'])['product_id'].count()
```

Создаем матрицу, где индекс - это user id, а колонки - это наименования товаров

▼ Код

```
matrix_cust = gr.unstack('product_name')
```

Применим лямбда функцию, чтобы заменить значения на численные

▼ Код

```
matrix_cust = matrix_cust.applymap(lambda x: 1 if x > 0 else 0)
```

Мы будем использовать косинусное сходство для идентификации клиентов с похожими покупками

▼ Код

```
cos_sim = cosine_similarity(matrix_cust, matrix_cust)
cos_sim_df = pd.DataFrame(cos_sim)
cos_sim_df.columns = matrix_cust.index
cos_sim_df['user_id'] = matrix_cust.index
cos_sim_df = cos_sim_df.set_index('user_id')
```

Теперь найдем user id с высоким значением косинусного сходства

▼ Код

```
col_list = list(cos_sim_df)
cos_sim_df['sum'] = cos_sim_df[col_list].sum(axis=1).round(2)
cos_sim_df[cos_sim_df['sum']>1]
```

Найдем рекомендации для клиента с user_id 296.

Клиент с user id 5 купил следующие товары:

▼ Код

```
cust_a = matrix_cust.loc[5]

cust_a_purchases = cust_a[cust_a > 0].index.tolist()
cust_a_purchases
```

Клиент с user id 296 купил следующие товары:

▼ Код

```
cust_b = matrix_cust.loc[296]

cust_b_purchases = cust_b[cust_b > 0].index.tolist()
cust_b_purchases
```

Вот эти товары мы можем порекомендовать клиенту с user id 296:

▼ Код

```
b_recs = set(cust_a_purchases) - set(cust_b_purchases)
b_recs
```

Вывод

В рамках данного мини-проекта мы произвели исследовательский анализ данных по продажам компании Amazon и выявили самые популярные категории и подкатегории товаров. Мы также создали простую систему рекомендаций, которую можно использовать для продвижения определенных товаров.

Ссылки

Ссылка на полный блокнот Jupyter (RU)

Ссылка на полный блокнот Jupyter (EN)

Контакты

lianazaripovar@gmail.com

tg: @zaripova_liana