Relatório: Análise do Comportamento de Compras dos Clientes

1. Visão Geral do Projeto

Este projeto tem como objetivo analisar o comportamento de compras dos clientes a partir de dados transacionais, integrando **Python, PostgreSQL e Power BI**, com foco em **gerar insights de negócio e apoiar a tomada de decisão**.

A análise busca responder perguntas estratégicas relacionadas a:

- Desempenho de vendas,
- Perfil dos clientes,
- Impacto de descontos,
- Comportamento de clientes recorrentes,
- potencial de subscrição.

📌 **Ferramentas utilizadas:**
Python (Pandas), PostgreSQL, Power BI

2. Descrição do Conjunto de Dados

O conjunto de dados contém informações de **3.900 compras**, distribuídas em **18 variáveis**, incluindo:

- **Dados demográficos:** idade, género, localização, status de subscrição
- **Dados de compra:** produto, categoria, valor da compra, estação, tamanho, cor
- **Comportamento do cliente:** compras anteriores, frequência, uso de desconto, avaliações e tipo de envio

Foram identificados **37 valores ausentes** na coluna de avaliação (*Review Rating*), posteriormente tratados.

3. Análise Exploratória de Dados (Python)

3.1.Preparação e Tratamento dos Dados (Python)

Nesta etapa, os dados foram preparados para garantir **qualidade, consistência e confiabilidade** das análises. Principais ações realizadas:

- Limpeza e tratamento de valores nulos na coluna de avaliação, utilizando a **mediana por categoria**
- Padronização dos nomes das colunas para o formato **snake_case**
- Tradução e padronização dos nomes das colunas para português
- Criação de variáveis derivadas, como:
  - **Faixa etária**
  - **Nível de recorrência de compra**
- Integração do DataFrame tratado com o banco de dados **PostgreSQL**

📌. Esta *etapa garante que todas as análises posteriores sejam feitas sobre uma base confiável.*

3.2.Verificação da estrutura do dataset Nesta etapa utilizamos o comando `df.info()` para inspecionar a base de dados após o carregamento inicial. O objetivo foi confirmar o número de registros, os tipos de variáveis e a presença de valores nulos em cada coluna. Essa checagem é fundamental para garantir a qualidade da análise, pois permite identificar inconsistências logo no início do processo e direcionar as etapas de limpeza e padronização.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3900 entries, 0 to 3899
Data columns (total 18 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Customer ID             3900 non-null   int64  
 1   Age                     3900 non-null   int64  
 2   Gender                  3900 non-null   object 
 3   Item Purchased          3900 non-null   object 
 4   Category                3900 non-null   object 
 5   Purchase Amount (USD)   3900 non-null   int64  
 6   Location                3900 non-null   object 
 7   Size                    3900 non-null   object 
 8   Color                   3900 non-null   object 
 9   Season                  3900 non-null   object 
 10  Review Rating           3863 non-null   float64
 11  Subscription Status     3900 non-null   object 
 12  Shipping Type           3900 non-null   object 
 13  Discount Applied        3900 non-null   object 
 14  Promo Code Used         3900 non-null   object 
 15  Previous Purchases      3900 non-null   int64  
 16  Payment Method          3900 non-null   object 
 17  Frequency of Purchases  3900 non-null   object 
dtypes: float64(1), int64(4), object(13)
memory usage: 548.6+ KB
```

3.3. Para o print das Estatísticas (`df. describe()`)

Exploração Estatística e Sumário de Dados > Através do método `df.describe()`, realizei um levantamento estatístico das variáveis numéricas e categóricas. Foi possível identificar que a média de gastos por cliente é de aproximadamente 59,76 USD, com idades variando entre 18 e 70 anos, fornecendo uma visão clara do perfil demográfico inicial.

```
df.describe(include='all')
```

	Customer ID	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	Location	Size	Color	Season	Review Rating
count	3900.000000	3900.000000	3900	3900	3900	3900.000000	3900	3900	3900	3900	3863.000000
unique	NaN	NaN	2	25	4	NaN	50	4	25	4	NaN
top	NaN	NaN	Male	Blouse	Clothing	NaN	Montana	M	Olive	Spring	NaN
freq	NaN	NaN	2652	171	1737	NaN	96	1755	177	999	NaN
mean	1950.500000	44.068462	NaN	NaN	NaN	59.764359	NaN	NaN	NaN	NaN	3.750065
std	1125.977353	15.207589	NaN	NaN	NaN	23.685392	NaN	NaN	NaN	NaN	0.716983
min	1.000000	18.000000	NaN	NaN	NaN	20.000000	NaN	NaN	NaN	NaN	2.500000
25%	975.750000	31.000000	NaN	NaN	NaN	39.000000	NaN	NaN	NaN	NaN	3.100000
50%	1950.500000	44.000000	NaN	NaN	NaN	60.000000	NaN	NaN	NaN	NaN	3.800000
75%	2925.250000	57.000000	NaN	NaN	NaN	81.000000	NaN	NaN	NaN	NaN	4.400000
max	3900.000000	70.000000	NaN	NaN	NaN	100.000000	NaN	NaN	NaN	NaN	5.000000

3.4. Para o print de Valores Nulos (`df.isna().sum()`)

Identificação de Dados Ausentes e Qualidade da Base > Executei uma auditoria de qualidade para identificar lacunas nos dados. O comando `df.isna().sum()` revelou **37 valores ausentes** especificamente na coluna de "Review Rating", enquanto as demais colunas apresentaram preenchimento total. Estes valores foram posteriormente tratados para garantir que as médias de avaliação no painel final fossem precisas.

```
df.isna().sum()
```

```
Customer ID                0
Age                        0
Gender                     0
Item Purchased             0
Category                   0
Purchase Amount (USD)      0
Location                   0
Size                       0
Color                      0
Season                     0
Review Rating             37
Subscription Status        0
Shipping Type              0
Discount Applied           0
Promo Code Used            0
Previous Purchases         0
Payment Method             0
Frequency of Purchases     0
dtype: int64
```

3.5.Tratamento de valores ausentes na coluna de avaliação Nesta etapa utilizamos o `pandas` para garantir a consistência dos dados. Os valores nulos da coluna *Review Rating* foram preenchidos com a mediana de cada categoria de produto, assegurando que não houvesse distorções nas análises. Em seguida, verificamos novamente os dados com `df.isna().sum()`, confirmando que não restaram valores ausentes em nenhuma coluna do dataset.

```
df['Review Rating']=df.groupby('Category')['Review Rating'].transform(lambda x:x.fillna(x.median()))
```

```
df.isna().sum()
```

```
Customer ID               0
Age                       0
Gender                    0
Item Purchased            0
Category                  0
Purchase Amount (USD)     0
Location                  0
Size                      0
Color                     0
Season                    0
Review Rating             0
Subscription Status       0
Shipping Type             0
Discount Applied          0
Promo Code Used           0
Previous Purchases        0
Payment Method            0
Frequency of Purchases    0
dtype: int64
```

3.6. Padronização dos nomes das colunas nesta etapa realizamos a limpeza e padronização dos nomes das colunas do dataset. Primeiro, todos os nomes foram convertidos para letras minúsculas, garantindo consistência na manipulação. Em seguida, os espaços foram substituídos por underlines (_), facilitando a leitura e evitando erros em consultas e scripts. Essa prática é essencial para manter o código organizado e reduzir problemas em análises futuras.

```
# Converte nome das colunas para minuscula
df.columns = df.columns.str.lower()
```

```
# Substitui os espaços por underline
df.columns=df.columns.str.replace(' ','_')
```

3.7. Tradução e padronização dos nomes das colunas Nesta etapa criamos um dicionário de tradução para converter os nomes originais das colunas do dataset (em inglês) para português. Essa prática facilita a leitura, torna o relatório mais acessível para públicos locais e mantém a consistência entre código e apresentação. Em seguida, aplicamos o `df.rename()` com o dicionário, garantindo que todas as colunas fiquem documentadas de forma clara e padronizada.

```python
# Dicionário de tradução das colunas
colunas_pt = {
    'customer_id': 'id_cliente',
    'age': 'idade',
    'gender': 'genero',
    'item_purchased': 'item_comprado',
    'category': 'categoria',
    'purchase_amount_(usd)': 'valor_compra_usd',
    'location': 'localizacao',
    'size': 'tamanho',
    'color': 'cor',
    'season': 'estacao',
    'review_rating': 'avaliacao',
    'subscription_status': 'status_assinatura',
    'shipping_type': 'tipo_envio',
    'discount_applied': 'desconto_aplicado',
    'promo_code_used': 'codigo_promocional_usado',
    'previous_purchases': 'compras_anteriores',
    'payment_method': 'metodo_pagamento',
    'frequency_of_purchases': 'frequencia_compras'
}
```

```python
# Renomear colunas
df.rename(columns=colunas_pt,inplace=True)
```

3.8. Integração com banco de dados PostgreSQL

Após o pré-processamento dos dados em Python, realizamos a integração com o banco de dados PostgreSQL. Para isso, configuramos a conexão utilizando a biblioteca `SQLAlchemy` e carregamos o DataFrame diretamente para a tabela *customer*. Esse processo garante que os dados tratados em pandas fiquem disponíveis para consultas estruturadas em SQL, permitindo análises mais robustas e integradas ao ambiente corporativo. A mensagem de sucesso confirma que os dados foram carregados corretamente no banco.

```python
from sqlalchemy import create_engine

# Configurações atualizadas
username = "postgres"
password = "12345"  # Senha corrigida
host = "localhost"
port = "5432"
database = "customer_behavior"

# Passo 1: Criar a conexão
engine = create_engine(f"postgresql+psycopg2://{username}:{password}@{host}:{port}/{database}")

# Passo 2: Carregar o seu DataFrame para o banco
# Certifique-se de que a variável 'df' existe (ex: df = pd.read_csv('seu_arquivo.csv'))
try:
    table_name = "customer"
    df.to_sql(table_name, engine, if_exists="replace", index=False)
    print(f"✅ Sucesso! Dados carregados na tabela '{table_name}' do banco '{database}'.")
except NameError:
    print("❌ Erro: O DataFrame 'df' não foi definido. Você precisa carregar seus dados primeiro!")
except Exception as e:
    print(f"❌ Ocorreu um erro inesperado: {e}")
```

```
✅ Sucesso! Dados carregados na tabela 'customer' do banco 'customer_behavior'.
```

3.9. Análise dentro do banco de dados PostgreSQL

Após a integração bem-sucedida do DataFrame no PostgreSQL, iniciamos as consultas estruturadas diretamente no banco. Essa etapa é fundamental porque permite explorar os dados com maior flexibilidade, utilizando SQL para responder perguntas de negócio. A partir daqui, realizamos análises como receita total por gênero, categorias mais lucrativas, impacto dos descontos e segmentação de clientes. O uso do banco garante escalabilidade, consistência e facilita a integração com outras ferramentas corporativas.

3.10. **Qual é a diferença de receita total entre clientes masculinos e femininos?"**
Explicação: Homens geraram 157.890 em receita, enquanto mulheres contribuíram com 75.191.

Data Output | Messages | Notifications

SQL

	genero text	Receita total text
1	Male	157.890
2	Female	75.191

3.11. Quais clientes usaram um desconto, mas ainda assim gastaram mais do que o valor médio de compras?"

	id_cliente bigint	valor_compra_usd bigint
1	2	64
2	3	73
3	4	90
4	7	85
5	9	97
6	12	68
7	13	72
8	16	81

Total rows: 839 Query complete 00:0

3.12. Quais são os 5 produtos com a maior média de avaliação

	item_comprado text	média produto Raking numeric
1	Gloves	3.86
2	Sandals	3.84
3	Boots	3.82
4	Hat	3.80
5	Skirt	3.78

3.13. Compare os valores médios de compra entre o envio padrão e o envio expresso

	tipo_envio text	comparaçao numeric
1	Express	60.48
2	Next Day Air	58.63
3	Standard	58.46

3.14. Os clientes assinantes gastam mais? Compare o gasto médio e a receita total entre assinantes e não assinantes

	status_assinatura text	total_cliente bigint	média_gastos numeric	total_receita text
	No	2847	59.87	170.436
2	Yes	1053	59.49	62.645

3.15. Quais 5 produtos têm a maior porcentagem de compras com descontos aplicados

	item_comprado text	desconto_apli numeric
1	Hat	50.00
2	Sneakers	49.66
3	Coat	49.07
4	Sweater	48.17
5	Pants	47.37

3.16. Segmente os clientes em Novos, Retornando e Fiéis com base no número total de compras anteriores, e mostre a distribuição.

	Segmento Cliente text	Número de Clientes bigint
	Fiel	3116
	Retornando	701
	Novo	83

3.17. Quais são os 3 produtos mais comprados dentro de cada categoria

	categoria text	item_comprado text	total_pedidos bigint	posicao bigint
	Accessori...	Jewelry	171	1
	Accessori...	Sunglasses	161	2
	Accessori...	Belt	161	3
	Clothing	Blouse	171	1
	Clothing	Pants	171	2
	Clothing	Shirt	169	3
	Footwear	Sandals	160	1
	Footwear	Shoes	150	2

otal rows: 11 Query complete 00:00:00.616

3.18. Clientes recorrentes (com mais de 5 compras anteriores) têm maior probabilidade de subscrever o serviço

Data Output Messages Notifications

	status_assinatura text	clientes recorrente bigint
1	No	2518
2	Yes	958

3.11. Qual é a contribuição da receita de cada faixa etária

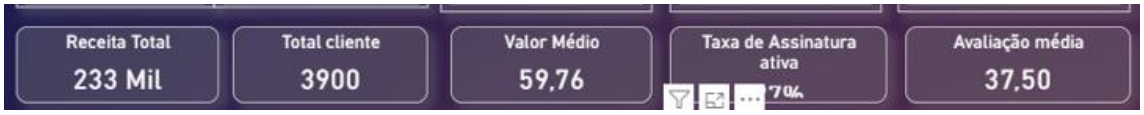
faixa_etaria text	receita_total text
Jovem Adulto	62.143
Meia-idade	59.197
Adulto	55.978
Idoso	55.763

4.Indicadores-Chave do Negócio (KPIs)

Os **KPIs representam a visão executiva do negócio** e são apresentados no painel em formato de **cartões (cards)**. Principais KPIs monitorados:

- **Receita Total**
- **Número de Clientes**
- **Quantidade Total de Vendas**
- **Ticket Médio**
- **Taxa de Subscrição**
- **Avaliação Média dos Produtos**

📌. Esses indicadores permitem uma **leitura rápida do desempenho geral** da empresa.



5.Perguntas de Negócio Analisadas (Gráficos)

Após a visão geral dos KPIs, a análise aprofunda-se em **perguntas de negócio**, respondidas através de **gráficos e tabelas**, permitindo entender **o porquê dos números**.

Principais perguntas respondidas:

- Qual é a **receita total por género**?
- Como a **receita se distribui por faixa etária**?
- Clientes assinantes gastam mais do que não assinantes?
- Qual o impacto do **uso de descontos** no valor das compras?
- Quais categorias e produtos concentram maior volume de vendas?
- Clientes recorrentes apresentam maior probabilidade de subscrição?

📌. Essas análises permitem identificar **padrões de comportamento e segmentos estratégicos**.

6.Insights Principais

A análise revelou insights relevantes para o negócio, entre eles:

- Clientes recorrentes tendem a apresentar **maior probabilidade de subscrição**, indicando potencial para estratégias de fidelização
- Determinados grupos etários concentram maior contribuição de receita
- O uso de descontos aumenta o volume de vendas, mas requer atenção para não comprometer a margem
- Produtos bem avaliados tendem a apresentar maior consistência de vendas

📌 Esses insights ajudam a transformar dados em **informação acionável**.

7. **Recomendações** de **Negócio**

Com base nos resultados obtidos, recomenda-se:

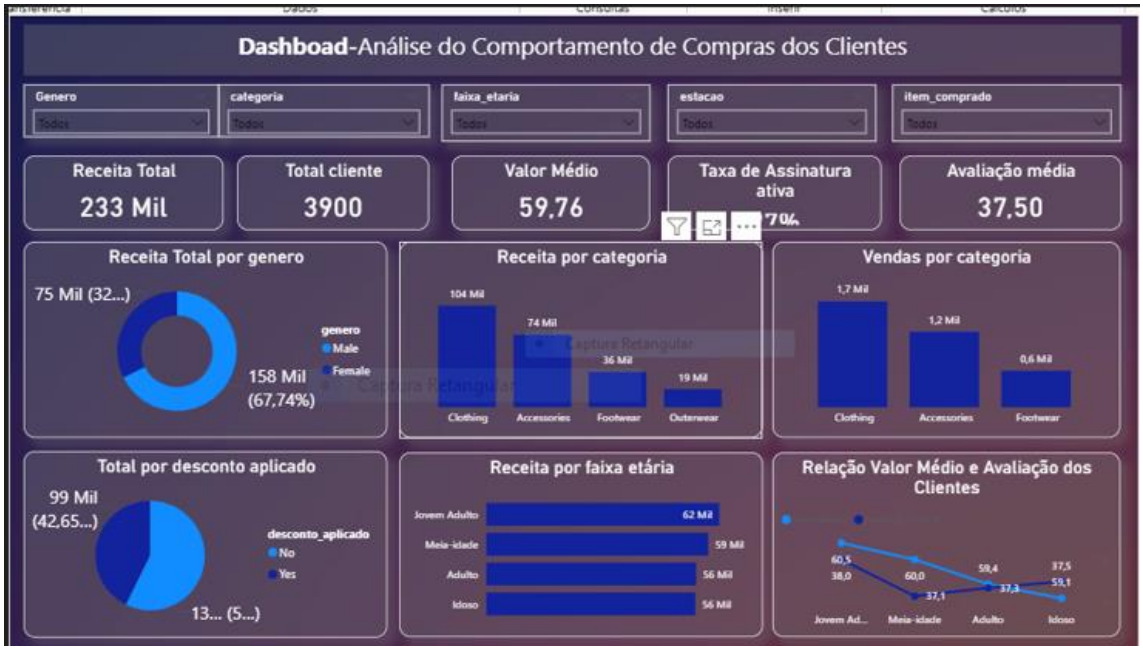
- Implementar **programas de fidelização** para clientes recorrentes
- Criar campanhas específicas para **aumentar a taxa de subscrição**
- Ajustar políticas de desconto com foco em **produtos estratégicos**
- Direcionar campanhas de marketing para **segmentos mais rentáveis**
- Destacar produtos com **alta avaliação** para reforçar confiança do cliente

8. Dashboard Final (Power BI)

O dashboard final consolida todos os resultados da análise, combinando:

- KPIs executivos
- Gráficos analíticos
- Segmentações por perfil de cliente

📌 O painel permite uma **exploração interativa**, facilitando a tomada de decisão por gestores e equipes de negócio.



9.**Conclusão**

Este projeto demonstra a aplicação prática de **análise de dados ponta a ponta**, desde o tratamento dos dados até a entrega de insights visuais e estratégicos. A integração entre **Python, SQL e Power BI** reforça a capacidade de transformar dados brutos em **valor real para o negócio**.