

**UNIVERSIDADE PRESBITERIANA MACKENZIE**  
**TECNÓLOGO EM CIÊNCIAS DE DADOS**

**PARTICIPANTES DO GRUPO**

Adrieli Machado Zaluski

Caroline Ribeiro Ferreira

Lais César Fonseca

Liliane Gonçalves de Brito Ferraz

Mucio Emanuel Feitosa Ferraz Filho

Otávio Bernardo Scandiuzzi

**TENDÊNCIAS DE COMPRAS EM SHOPPINGS DE ISTAMBUL**

**SÃO PAULO**

**2023**

## Sumário

1.	GLOSSÁRIO .....	3
2.	INTRODUÇÃO .....	5
3.	COMPOSIÇÃO DO GRUPO .....	6
4.	CRONOGRAMA DE DESENVOLVIMENTO DO PROJETO .....	6
5.	INFORMAÇÕES SOBRE O DIRETÓRIO .....	7
6.	APRESENTAÇÃO DO PROJETO .....	9
7.	APRESENTAÇÃO DA EMPRESA .....	10
8.	PROBLEMA DO ESTUDO .....	14
9.	METADADOS .....	15
10.	TRATAMENTO DOS DADOS .....	19
11.	ANÁLISE EXPLORÁTÓRIA DE DADOS .....	20

## 1. GLOSSÁRIO

- **Colaboratory:** Conhecido também como “Colab”, é um produto do Google Research, área de pesquisas científicas do Google. O Colab permite que qualquer pessoa escreva e execute código Python arbitrário pelo navegador e é especialmente adequado para aprendizado de máquina, análise de dados e educação.
- **DataFrame:** É uma estrutura de dados bidimensional com os dados alinhados de forma tabular em linhas e colunas.
- **Datasets:** conjuntos de dados organizados em um formato similar ao das tabelas, com linhas e colunas que contém informações sobre determinado tema.
- **GitHub:** GitHub é uma plataforma de hospedagem de código-fonte e arquivos com controle de versão usando o Git. Ele permite que programadores, utilitários ou qualquer usuário cadastrado na plataforma contribuam em projetos privados e/ou Open Source de qualquer lugar do mundo.
- **Kaggle:** É uma plataforma para aprendizado de ciência de dados. É também uma comunidade, a maior da internet, para assuntos relacionados com Data Science.
- **NumPy:** É uma biblioteca para a linguagem de programação Python, que suporta o processamento de grandes, multi-dimensionais arranjos e matrizes, juntamente com uma grande coleção de funções matemáticas de alto nível para operar sobre estas matrizes
- **Matplotlib:** É uma biblioteca de softwares para criação de gráficos e visualizações de dados em geral, feita para e da linguagem de programação Python e sua extensão de matemática NumPy.
- **Pandas:** É uma biblioteca de software criada para a linguagem Python para manipulação e análise de dados. Em particular, oferece estruturas e operações para manipular tabelas numéricas e séries temporais.
- **Python:** É uma linguagem de programação de alto nível, interpretada de script, imperativa, orientada a objetos, funcional, de tipagem dinâmica e forte. Foi lançada por Guido van Rossum em 1991.
- **Readme:** É um arquivo com extensão .md, ou seja, ele é escrito em Markdown que é uma linguagem de marcação utilizada para converter o texto em um HTML válido.

- **Seaborn:** É uma biblioteca Python de visualização de dados amplamente popular, comumente usada para tarefas de ciência de dados e aprendizado de máquina.
- **String:** Sequências de caracteres alfanuméricos (letras, números e/ou símbolos) amplamente usadas em programação.

## 2. INTRODUÇÃO

O propósito deste projeto é realizar um estudo de caso prático em uma base de dados pública, disponibilizada no site Kaggle<sup>1</sup>. Os dados escolhidos para este estudo referem-se a compras em shoppings na cidade de Istambul, na Turquia.

Esses dados serão utilizados para identificar padrões no comportamento de compras e prever tendências futuras de consumo. Para este estudo de ciências de dados será utilizado técnicas adquiridas nos componentes curriculares de introdução a ciência de dados, pensamento computacional e análise exploratória de dados, com o objetivo principal de analisar e interpretar o que esses dados podem mostrar através de aplicação de medidas estatísticas, no qual discutiremos e responderemos algumas perguntas ao longo do projeto, permitindo uma compreensão mais profunda do comportamento dos consumidores em Istambul.

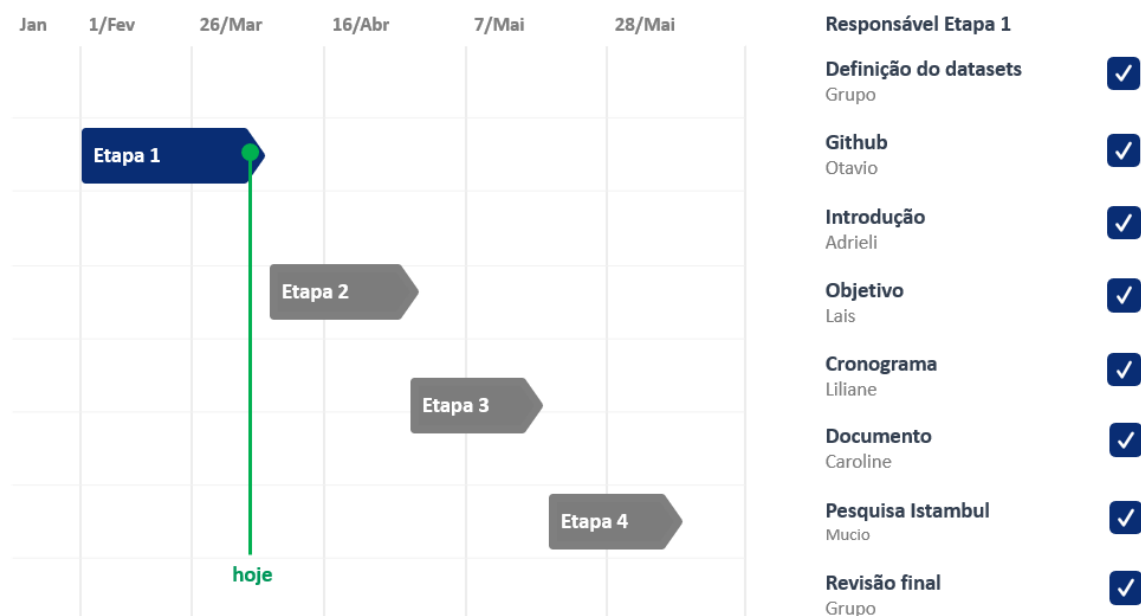
Neste projeto apresentaremos os principais comandos utilizados para realizar as análises dos dados através da aplicação da linguagem Python, com o uso do Colab. Como acessar o metadado, consultar quantidade de atributos, linhas, resumo do dataset, verificar os tipos dos atributos e como importar as bibliotecas necessárias para realizar a exploração de dados de forma mais simplificada.

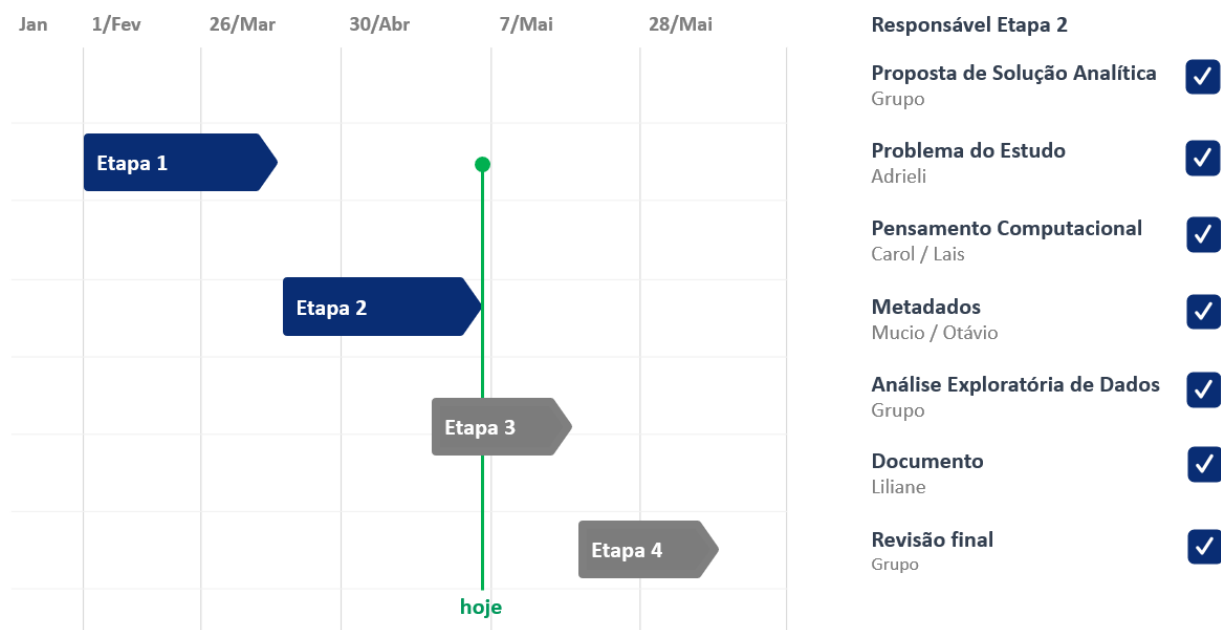
### 3. COMPOSIÇÃO DO GRUPO

Integrantes	Nº de matrícula
Adrieli Machado Zaluski	22503668
Caroline Ribeiro Ferreira	22514635
Lais César Fonseca	22500790
Liliane Gonçalves de Brito Ferraz	22501142
Mucio Emanuel Feitosa Ferraz Filho	22515925
Otávio Bernardo Scanduzzi	22511921

### 4. CRONOGRAMA DE DESENVOLVIMENTO DO PROJETO

Reportar-se o percentual de evolução de entregas referente as ações propostas pelo componente curricular de Projeto Aplicado I do curso de Tecnologia em Ciências de Dados.





## 5. INFORMAÇÕES SOBRE O DIRETÓRIO

Todo o conteúdo do projeto estará disponível no site da GitHub, que poderá ser acessado pelo link:

<https://github.com/OtavioBer/ProjetoAplicadoI>

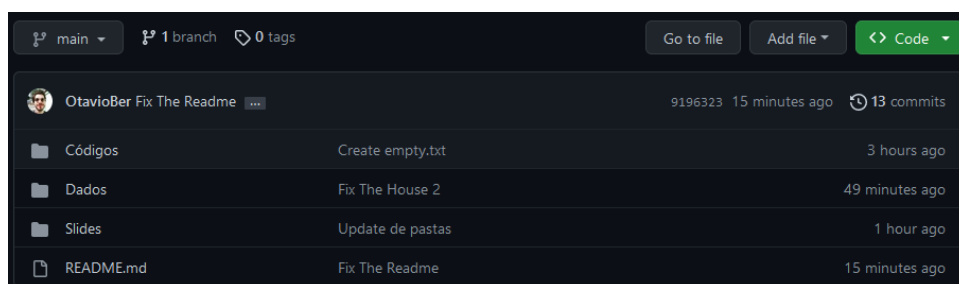
O diretório está organizado por pastas e subpastas.

Na pasta “Códigos” será disponibilizado os códigos em Python de cada resposta.

Na pasta “Slide” temos o cronograma de atividades.

Na pasta “Dados” temos os arquivos utilizados para análise e a pasta de “Backup”, contendo um arquivo original de todos os dados.

Temos também o arquivo README.md com algumas informações do projeto, que ao decorrer do projeto, poderá ocorrer alterações nas estruturas de pastas, arquivos e no ReadMe.



## 🔗 Link dos Datasets analisado

### Dataset público Kaggle

<https://www.kaggle.com/datasets/mehmettahiraslan/customer-shopping-dataset>

### Dataset criado a partir de busca no Google

[https://github.com/OtavioBer/ProjetoAplicadoI/blob/main/Dados/Backup/Base\\_Google\\_Endere%C3%A7o\\_Shoppin](https://github.com/OtavioBer/ProjetoAplicadoI/blob/main/Dados/Backup/Base_Google_Endere%C3%A7o_Shoppin)

☰ README.md

# Projeto Aplicado I - Tendência de compras em shoppings de Istambul

## Sumário

### Apresentação

- Integrantes
- Introdução

### Link dos Datasets analisado

- Dataset público Kaggle
- Dataset criado a partir de busca no Google

### Perguntas a serem respondidas

- O público feminino consome mais roupas que o masculino?
- Quais produtos são mais consumidos por gênero?
- Qual a categoria de compras mais consumida por faixa etária?
- Qual o produto com maior consumo no público em geral?
- Qual a forma de pagamento mais utilizada?
- Qual a sazonalidade (dia da semana) de consumo com maior volume de compras?
- Qual a média de consumo por gênero anual?
- Qual shopping possui o maior ticket médio de compras?
- Qual a linha de tendência de gastos do gênero e por faixa etária?
- Qual categoria tem a maior quantidade consumida?
- Qual a localidade de shopping tem o maior consumo por categoria?



## 6. APRESENTAÇÃO DO PROJETO

Para o desenvolvimento do projeto teremos como base de estudo os dados públicos disponibilizados na plataforma de datasets Kaggle.

O dataset escolhido nos fornece conjunto de dados que contém informações de compras de 10 shoppings diferentes no período de janeiro/2021 a março/2023 na cidade de Istambul, na Turquia. O conjunto de dados inclui informações essenciais para a realização das análises propostas, como números de faturas, IDs de clientes, faixa etária, gênero, métodos de pagamento, categorias de produtos, quantidade, preço, datas de pedidos e o nome dos shoppings centers.

Com posse dos dados, nosso objetivo é, por meio da exploração dos dados, conseguir identificar tendências de compras, padrões, e entender melhor o comportamento de consumo do varejo em Istambul.

Com o objetivo de explorar os dados de forma a identificar a regionalização dos shoppings da cidade de Istambul, realizamos uma busca sobre a localização de cada shopping listado na base de dados e criamos um novo dataset em xls com o endereço, distrito e a região. Com este novo dataset poderemos analisar se a localização do shopping interfere nas demais análises que serão avaliadas.

### 6.1. Origem Dataset

O dataset adquirido no site da Kaggle, contém 99.457 linhas e 10 colunas, no link de acesso:

<https://www.kaggle.com/datasets/mehmettahiraslan/customer-shopping-dataset>

```
In [1]: import pandas as pd

df = pd.read_csv(
    r"C:\Users\otavi\Documents\GitHub\ProjetoAplicadoI\Dados\Backup\Base_Keaggle_Original.csv")

In [2]: df
```

Out[2]:

	invoice_no	customer_id	gender	age	category	quantity	price	payment_method	invoice_date	shopping_mall
0	I138884	C241288	Female	28	Clothing	5	1500.40	Credit Card	5/8/2022	Kanyon
1	I317333	C111565	Male	21	Shoes	3	1800.51	Debit Card	12/12/2021	Forum Istanbul
2	I127801	C266599	Male	20	Clothing	1	300.08	Cash	9/11/2021	Metrocity
3	I173702	C988172	Female	66	Shoes	5	3000.85	Credit Card	16/05/2021	Metropol AVM
4	I337046	C189076	Female	53	Books	4	60.60	Cash	24/10/2021	Kanyon
...	...	...	...	...	...	...	...	...	...	...
99452	I219422	C441542	Female	45	Souvenir	5	58.65	Credit Card	21/09/2022	Kanyon
99453	I325143	C569580	Male	27	Food & Beverage	2	10.46	Cash	22/09/2021	Forum Istanbul
99454	I824010	C103292	Male	63	Food & Beverage	2	10.46	Debit Card	28/03/2021	Metrocity
99455	I702964	C800631	Male	56	Technology	4	4200.00	Cash	16/03/2021	Istinye Park
99456	I232867	C273973	Female	36	Souvenir	3	35.19	Credit Card	15/10/2022	Mall of Istanbul

99457 rows × 10 columns

## 6.2. Dados de Apoio

A base de dados de endereço que foi desenvolvida a partir de uma pesquisa da localização de cada shopping dispõe de 10 linhas e 4 colunas.

[https://github.com/OtavioBer/ProjetoAplicadoI/blob/main/Dados/Backup/Base\\_Google\\_Endere%C3%A7o\\_Shoppings\\_Istambul.xlsx](https://github.com/OtavioBer/ProjetoAplicadoI/blob/main/Dados/Backup/Base_Google_Endere%C3%A7o_Shoppings_Istambul.xlsx)

```
In [4]: df2 = pd.read_excel(r"C:\Users\otavi\Documents\GitHub\ProjetoAplicadoI\Dados\Backup\Base_Google_Endereço_Shoppings_Istambul.xlsx")
```

```
In [5]: df2
```

```
Out[5]:
```

	Nome do Shopping	Endereço	Distrito	Regiões
0	Cevahir AVM	19 Mayıs, Büyükdere Cd. No:22, 34360 Şişli/İst...	Sisli	Região Européia
1	Emaar Square Mall	Ünalan, Libadiye Cd. No:88, 34700 Üsküdar/İsta...	Uskudar	Região Asiática
2	Forum Istanbul	Kocatepe, Paşa Cd, 34045 Bayrampaşa/Istanbul, ...	Bayrampasa	Região Européia
3	Istinye Park	Pınar, 34460 Sarıyer/Istanbul, Turquia	Sarıyer	Região Européia
4	Kanyon	Levent Mah, Büyükdere Cd. No:185, 34394 Şişli/...	Sisli	Região Européia
5	Mall of Istanbul	Ziya Gökalp, Süleyman Demirel Blv No:7, 34490...	İkitelli	Região Européia
6	Metrocity	Levent, Büyükdere Cd. No:171, 34330 Beşiktaş/İ...	Besiktas	Região Européia
7	Metropol AVM	Ertuğrul, Atatürk Mahallesi Ataşehir Bulvarı, ...	Atasehir	Região Asiática
8	Viaport Outlet	Yenişehir, Dedepaşa Cd No:19, 34912 Pendik/İst...	Pendik	Região Asiática
9	Zorlu Center	Levazım, Kuru Sok. No:2, 34340 Beşiktaş/Istanb...	Besiktas	Região Européia

## 7. APRESENTAÇÃO DA EMPRESA

Os dados que serão analisados, são referentes a 10 shopping da cidade em Istambul, na Turquia, onde descreveremos a seguir um breve resumo sobre cada um deles.

Para esta classe de segmento existem semelhanças em seus propósitos, sendo eles:

**Missão:** Proporcionar aos clientes experiências incríveis e momentos mágicos sendo referência em entretenimento, moda, lazer e gastronomia.

**Visão:** Ser referência no mercado de Shoppings Centers, oferecendo marcas de destaque no cenário nacional e internacional, promovendo eventos exclusivos para fomentar a cultura, entretenimento e lazer.

**Valores:** Ética, transparência, compromisso, inovação, eficiência e gestão ambiental.

### 7.1. Conhecendo os Shoppings

- **Kanyon**

Este é um shopping notável dentro de Istambul, isso por que possui uma arquitetura luxuosa, abrigando diversas marcas de grife, além de fitness center, e uma torre de escritórios com 179

residências de luxo. É mais um que se localiza na parte europeia da capital, situando-se no distrito comercial de Levent.

Figura 1 - Cevahir AVM



Fonte: <https://www.nenerede.com.tr/ilan/istanbul-cevahir-avm/>

- **Cevahir AVM**

O Istanbul Cevahir Shopping foi aberto no ano de 2005 e desde sua abertura até o ano de 2011 ele foi considerado o maior shopping em área bruta da Europa, mas ainda hoje permanece na lista dos maiores do mundo. Este moderno centro comercial turco fica localizado no distrito de Sisli, situado na parte europeia de Istambul.

- **Emaar Square Mall**

O Emaar Square Mall é o maior shopping center da Turquia, possuindo 6 andares em sua estrutura. Foi aberto para o público no ano de 2017 e possui diversas atrações, como um aquário e zoológico subaquático, vista panorâmica e um museu de ilusões. É localizado na parte asiática da cidade, dentro do distrito de Uscudar.

- **Istinye Park**

Este shopping é um dos mais populares de toda a Turquia, sendo conhecido por possuir diversas marcas de luxo ao passo que possui diversas iguarias locais, além de várias opções para lazer e alimentação. Este centro comercial se localiza na parte europeia de Istambul, no distrito de Sariyer.



Figura 2 - Forum Istanbul



Fonte: [https://www.tripadvisor.com.br/Attraction\\_Review-g293974-d3386223-Reviews-Forum\\_Istanbul\\_Alisveris\\_Merkezi-Istanbul.html](https://www.tripadvisor.com.br/Attraction_Review-g293974-d3386223-Reviews-Forum_Istanbul_Alisveris_Merkezi-Istanbul.html)

- **Forum Istanbul**

Este é mais um grande shopping da capital turca, empregando mais de 5000 funcionários e abrigando mais de 280 marcas, nacionais e internacionais, algumas das mais conhecidas são: Adidas, Carrefour, Ecco, Lego e Levi's. Este centro se situa no distrito de Bayrampasa, parte europeia da capital.

- **Mall of Istanbul**

Este é mais um grande e luxuoso shopping da capital turca, possuindo 350 lojas de marcas de luxo, parque de diversões cobertos e cinema. Sua arquitetura é luxuosa, com características que lembram a de um enorme teatro. Sua localização é no distrito de Basaksehir, parte europeia da cidade.

- **Metrocity**

O shopping Metrocity conta com 4 andares em sua estrutura, abrigando uma grande variedade de lanchonetes, restaurantes, cafeterias, além de diversas lojas de marcas nacionais e internacionais. Assim como o Kanyon, fica localizado no distrito comercial de Levent.

- **Zorlu Center**

Este é um centro comercial que apresenta uma ampla variedade de restaurantes e cafés, além de diversas lojas de marcas conhecidas mundialmente. O Zorlu Center possui um ambiente que combina espaço interno e externo, possuindo áreas verdes em sua estrutura. Sua localização é no distrito de Sisli, assim como o Cevahir AVM.

Figura 3 - Metropol AVM



Fonte: <https://www.endeksa.com/en/analiz/istanbul/atasehir/ataturk/malls>

- **Metropol AVM**

O Metropol Istanbul Shopping Center possui 5000 metros quadrados de área de entretenimento, com playgrounds para as crianças, pátios esportivos e campos abertos, além disso possui cerca de 250 lojas em sua estrutura. Este estabelecimento se localiza no distrito de Atasehir, parte asiática da cidade.

- **Viaport Outlet**

Este é o maior shopping do segmento outlet de toda a Turquia, em sua estrutura estão incluídas áreas verdes, ruas, praças e até um lago, apresentando, assim, uma arquitetura diferente dos shoppings convencionais. Além disso possui cinema, boliche, ampla variedade culinária e diversas lojas. Se situa na parte asiática da capital turca, no distrito de Pendik.

## 8. PROBLEMA DO ESTUDO

O objetivo será gerar alguns insights sobre padrões, tendências de perfil de compras e responder algumas perguntas como:

1. O público feminino consome mais roupas que o masculino?
2. Quais produtos são mais consumidos por gênero?
3. Qual o ranking de produtos mais consumidos pelo público em geral?
4. Qual a forma de pagamento mais utilizada por faixa etária e por gênero?
5. Qual a sazonalidade (dia da semana) de consumo com maior volume de compras?
6. Qual a média de consumo anual por gênero e faixa etária?
7. Qual shopping possui o maior ticket médio de compras?
8. Como se dá a comparação de gastos totais por gênero e por faixa etária? E dos gastos médios?
9. Qual faixa etária mais consome artigos de leitura?
10. Qual a localidade de shopping tem o maior consumo por categoria?
11. Qual produto tem o maior e menor ticket médio por região?

### 8.1. Pensamento Computacional

Para iniciar um projeto ou a resolução de um problema, se faz necessário uma avaliação do contexto geral no qual será trabalhado, aplicando métodos importantes para alcançar os objetivos propostos, ter assertividade na entrega, minimizar impactos de erros, realizar a entrega dentro do cronograma definido e na qualidade e custo esperado.

E neste processo existem 4 etapas que subsidiam este momento inicial de um projeto, que são as dimensões do pensamento computacional, sendo elas:

1. **Decomposição:** Separar problemas complexos em problemas menores;
2. **Reconhecimento de padrões:** Analisando problemas menores torna mais simples o processo de identificação de problema semelhantes já resolvidos em outros momentos e/ou projetos;
3. **Abstração:** É um método essencial de ser aplicado, para focar no que é relevante.
4. **Algoritmo:** É o conjunto de instruções bem estruturada dadas ao sistema para que seja alcançando os resultados esperados.

Neste contexto, é importante ressaltar que o pensamento computacional é um processo de resolução de problemas que se faz necessário seguir alguns passos, sendo eles:

1. Formulação de problemas de forma que nos permita usar um computador e outras ferramentas para nos ajudar a resolvê-los.
2. Organização e análise lógica de dados.

3. Representação de dados por meio de abstrações, como modelos e simulações.
4. Automatização de soluções por meio do pensamento algorítmico (uma série de etapas ordenadas).
5. Identificação, análise e implementação de possíveis soluções com o objetivo de alcançar a combinação mais eficiente e efetiva de etapas e recursos.
6. Generalização e transferência deste processo de resolução de problemas para uma grande variedade de problemas.

## 9. METADADOS

### 9.1. Tipo de arquivo

A base de dados adquirida é de extensão csv.

### 9.2. Origem dos dados

Os dados são de domínio público/aberto, do site da Kaggle.

### 9.3. Sensibilidade / LGPD

Ao criar o conjunto de dados, as informações pessoais dos clientes foram anonimizadas para proteger a privacidade, no entanto, esta base de dados não possui dados sensíveis e está de acordo com a Lei Geral de Proteção de Dados Pessoais – LGPD.

### 9.4. Validade

Os dados foram disponibilizados recentemente, março/2023, e está dentro da validade para análise exploratória dos dados e avaliação de comportamento atual de compras.

### 9.5. Proprietário do dado

Mehmet Tahir Aslam, Analista de dados na Crystal System, Istambul, Turquia.

### 9.6. Descrição dos atributos dos dados

- **invoice\_no:** Número da fatura. Nominal. Uma combinação da letra 'I' e um número inteiro de 6 dígitos atribuído exclusivamente a cada operação.
- **customer\_id:** Número do cliente. Nominal. Uma combinação da letra 'C' e um número inteiro de 6 dígitos atribuído exclusivamente a cada operação.
- **gender:** Variável string do sexo do cliente.
- **age:** Variável inteiro positivo da idade do cliente.
- **categoria:** Variável string da categoria do produto adquirido.

- **quantidade:** As quantidades de cada produto (item) por transação. Numérico.
- **preço:** preço unitário. Numérico. Preço do produto por unidade em liras turcas (TL).
- **Payment\_method:** Variável string da forma de pagamento (dinheiro, cartão de crédito ou débito) utilizada na transação.
- **invoice\_date:** Data da fatura. O dia em que uma transação foi gerada.
- **shopping\_mall:** Variável string do nome do shopping onde foi feita a transação.

## 9.7. Acessando o metadados

Para realizar a operação de acessar os metadados, se faz necessário a importação da biblioteca “Pandas”, que tem por funcionalidade adquirir dados, selecionar dados de interesse e fazer algumas transformações simples. E para realizar esta operação inserir o comando: “import pandas as pd” e em seguida inserir o comando para buscar o arquivo em seu diretório de origem, “df = pd.read\_csv(r'https://raw.githubusercontent.com/OtavioBer/ProjetoAplicadoI/main/Dados/Backup/Base\_Keaggle\_Original.csv')”.

```
import pandas as pd
df = pd.read_csv(r'https://raw.githubusercontent.com/OtavioBer/ProjetoAplicadoI/main/Dados/Backup/Base_Keaggle_Original.csv')
```

## 9.8. Conhecendo o metadados

Para se iniciar uma análise com os dados, primeiramente é importante conhecer o metadados que será explorado, para isto é possível utilizar o comando “df.head()”, que é um comando que permite exibir o aspecto inicial dos dados a partir das primeiras linhas, identificando quais os tipos de dados que será analisado e quais são seus atributos.

```
df.head()
```

	invoice_no	customer_id	gender	age	category	quantity	price	payment_method	invoice_date	shopping_mall
0	I138884	C241288	Female	28	Clothing	5	1500.40	Credit Card	5/8/2022	Kanyon
1	I317333	C111565	Male	21	Shoes	3	1800.51	Debit Card	12/12/2021	Forum Istanbul
2	I127801	C266599	Male	20	Clothing	1	300.08	Cash	9/11/2021	Metrocity
3	I173702	C988172	Female	66	Shoes	5	3000.85	Credit Card	16/05/2021	Metropol AVM
4	I337046	C189076	Female	53	Books	4	60.60	Cash	24/10/2021	Kanyon

Identificando a quantidades de linhas e colunas do dataset, podemos utilizar o comando “df.shape”:



```
df.shape
```

(99457, 10)

Identificando quais os nomes dos atributos, colunas do dataset, podemos utilizar o comando “df.columns.to\_list()”.

```
df.columns.to_list()
```

['invoice\_no',  
'customer\_id',  
'gender',  
'age',  
'category',  
'quantity',  
'price',  
'payment\_method',  
'invoice\_date',  
'shopping\_mall']

Identificando quais os tipos dos atributos, podemos utilizar o comando “df.dtypes”.

```
df.dtypes
```

invoice_no	object
customer_id	object
gender	object
age	int64
category	object
quantity	int64
price	float64
payment_method	object
invoice_date	object
shopping_mall	object
dtype:	object

Identificando informações estatísticas sumarizadas dos dados, podemos utilizar o comando “df.describe(include="all")”.

```
[9] df.describe(include="all")
```

	invoice_no	customer_id	gender	age	category	quantity	price	payment_method	invoice_date	shopping_mall
count	99457	99457	99457	99457.000000	99457	99457.000000	99457.000000	99457	99457	99457
unique	99457	99457	2	NaN	8	NaN	NaN	3	797	10
top	1138884	C241288	Female	NaN	Clothing	NaN	NaN	Cash	24/11/2021	Mall of Istanbul
freq	1	1	59482	NaN	34487	NaN	NaN	44447	159	19943
mean	NaN	NaN	NaN	43.427089	NaN	3.003429	689.256321	NaN	NaN	NaN
std	NaN	NaN	NaN	14.990054	NaN	1.413025	941.184567	NaN	NaN	NaN
min	NaN	NaN	NaN	18.000000	NaN	1.000000	5.230000	NaN	NaN	NaN
25%	NaN	NaN	NaN	30.000000	NaN	2.000000	45.450000	NaN	NaN	NaN
50%	NaN	NaN	NaN	43.000000	NaN	3.000000	203.300000	NaN	NaN	NaN
75%	NaN	NaN	NaN	56.000000	NaN	4.000000	1200.320000	NaN	NaN	NaN
max	NaN	NaN	NaN	69.000000	NaN	5.000000	5250.000000	NaN	NaN	NaN

Identificando informações de valores/quantidade únicos em um determinado atributo, para entender com a variedades de informações que será possível desdobrar as análises, podemos utilizar o comando “df['category'].nunique()”.

```
df['category'].nunique()
8
```

## 9.9. Bibliotecas de exploração de dados

Na exploração do dataframe é importante identificar quais bibliotecas Python serão necessárias para realizar a exploração dos dados de forma que os códigos aplicados sejam performados adequadamente e de forma eficaz e simplificada. Neste projeto identificamos a necessidade de utilização das seguintes bibliotecas:

- Pandas
- Numpy
- Seborn
- Matplotlib

Utilizando os comandos:

```
import pandas as pd
```

```
import numpy as np
```

```
import seaborn as sns
```

```
import matplotlib.pyplot as plt
```

## 10. TRATAMENTO DOS DADOS

Para melhor desempenhar a exploração do dataset escolhido, identificamos a necessidades de criar algumas funções para realizar análises de forma mais resumida e agrupada.

Foi desenvolvido função para tratamento de data, separando da data o ano/mês, o ano, o mês e o dia da semana, aplicando o comando:

```
# converter tipo data da fatura para data
df['invoice_date'] = pd.to_datetime(df['invoice_date'], dayfirst=True)
# criar novas colunas para ano_mês, ano, mês, dia da semana
df['year_month'] = df['invoice_date'].dt.to_period('M')
df['year'] = df['invoice_date'].dt.strftime('%Y')
df['month'] = df['invoice_date'].dt.strftime('%b')
df['day_of_week'] = df['invoice_date'].dt.strftime('%a')
```

Ordenar os meses e os dias da semana, aplicando o comando:

```
# definir a ordem do mês
month_order = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']
df['month'] = df['month'].astype(pd.CategoricalDtype(categories=month_order, ordered=True))

# definir a ordem do dia da semana
weekday_order = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun']
df['day_of_week'] = df['day_of_week'].astype(pd.CategoricalDtype(categories=weekday_order, ordered=True))
```

Criar cálculo de vendas, onde será realizado a operação de multiplicação entre o atributo de quantity e price, apresentando o resultado em vendas, aplicando o comando:

```
# criar novas colunas para o cálculo de vendas
df['vendas'] = df['quantity'] * df['price']
```

Criar funcionalidade para range de faixa etária a partir do atributo age, aplicando o comando:

```
# Criar função para range de idade
def age_range(age):
    if age <= 19:
        age = '18-19'
    elif 20 <= age <= 29:
        age = '20-29'
    elif 30 <= age <= 39:
        age = '30-39'
    elif 40 <= age <= 49:
        age = '40-49'
```

```

elif 50 <= age <= 59:
    age = '50-59'
else:
    age = '60-69'
return age

# criar novas colunas para age_range
df['age_range'] = df['age'].apply(age_range)

```

## 11. ANÁLISE EXPLORÁTÓRIA DE DADOS

Realizar as análises dos dados para resolver as questões apresentadas na proposta de análise, e identificar problemas, tendências e comportamentos, aplicando os métodos estatísticos de análise exploratória de dados, através do uso da linguagem Python, utilizando a ferramenta Colaboratory.

### 11.1. O público feminino consome mais roupas que o masculino?

Os códigos abaixo classificam o grupo feminino e masculino que comprou na categoria de roupas (clothing) em quantidade de pessoas, neste caso podemos analisar que o público feminino tende a comprar mais roupas que o masculino.

```
df.loc[(df['gender'] == 'Female') & (df['category'] == 'Clothing'), 'gender'].value_counts()
```

```

#Número de mulheres que compram roupa
df.loc[(df['gender'] == 'Female') & (df['category'] == 'Clothing'), 'gender'].value_counts()

Female    20652
Name: gender, dtype: int64

```

```
df.loc[(df['gender'] == 'Male') & (df['category'] == 'Clothing'), 'gender'].value_counts()
```

```

#Numero de homens que compram roupa
df.loc[(df['gender'] == 'Male') & (df['category'] == 'Clothing'), 'gender'].value_counts()

Male      13835
Name: gender, dtype: int64

```

Também é possível analisar em quantidade de roupas (clothing), somando a quantidade que cada gênero compra.

```
df[df['category'] == 'Clothing'].groupby(['gender', 'category'])['quantity'].agg(['sum'])
```



#Quantidade de clothing por gênero

```
df[df['category'] == 'Clothing'].groupby(['gender', 'category'])['quantity'].agg(['sum'])
```

sum



gender category

Female Clothing 62039

Male Clothing 41519

## 11.2. Quais produtos são mais consumidos por gênero?

O seguinte código utilizado para analisar, foi agrupado por gênero (gender) e categoria (category), e no final somando a quantidade, gerando uma lista com a quantidade de produtos vendidos por gênero. Identifica o produto “roupas” como o mais consumido por mulheres e homens.

```
df.groupby(['gender', 'category'])['quantity'].agg(['sum'])
```

[11]

#Quantidade de produtos por categoria e gênero

```
df.groupby(['gender', 'category'])['quantity'].agg(['sum'])
```

sum



gender category

Female Books 8776

Clothing 62039

Cosmetics 27261

Food & Beverage 26362

Shoes 17906

Souvenir 8976

Technology 8977

Toys 18362

Male Books 6206

Clothing 41519

Cosmetics 18204

Food & Beverage 17915

Shoes 12311

Souvenir 5895

Technology 6044

Toys 11959

### 11.3. Qual o ranking de produtos mais consumidos pelo público em geral?

Utilizamos o código abaixo para identificar a quantidade de vezes que cada categoria de produtos mais aparecia no DataFrame, para assim identificar o ranking de produtos mais consumidos pelo público em geral em ordem decrescente.

```
df.category.value_counts()
```

```
✓ [32] #ordenando os produtos mais consumidos pelo público
Ds df.category.value_counts()
```

```
Clothing      34487
Cosmetics     15097
Food & Beverage 14776
Toys          10087
Shoes         10034
Souvenir      4999
Technology    4996
Books         4981
Name: category, dtype: int64
```

### 11.4. Qual a forma de pagamento mais utilizada por faixa etária e por gênero?

Nessa questão, foi utilizado o método para criar uma faixa etária no DataFrame e um group-by, para que possamos verificar qual a forma de pagamento mais utilizada. No qual foi identificado que o método mais utilizado para homens e mulheres em todas as faixas etárias é o dinheiro (cash).

```
grouped = df.groupby(['gender', pd.cut(df['age'], bins=[0, 20, 30, 40, 50, 60, 100],
labels=['0-20', '21-30', '31-40', '41-50', '51-60', '60+'])])
result = grouped['payment_method'].agg(pd.Series.mode)
print(result)
```

```
[40]
# agrupar por faixa etária e gênero
grouped = BD.groupby(['gender', pd.cut(BD['age'], bins=[0, 20, 30, 40, 50, 60, 100], labels=['0-20', '21-30', '31-40', '41-50', '51-60', '60+'])])

# calcular a forma de pagamento mais comum em cada grupo
result = grouped['payment_method'].agg(pd.Series.mode)

# imprimir o resultado
print(result)
```

```
gender  age  payment_method
Female  0-20  Cash
        21-30  Cash
        31-40  Cash
        41-50  Cash
        51-60  Cash
        60+   Cash
Male    0-20  Cash
        21-30  Cash
        31-40  Cash
        41-50  Cash
        51-60  Cash
        60+   Cash
Name: payment_method, dtype: object
```

```
[ ] # Nessa análise vimos que o método de pagamento mais utilizado tanto para homens quanto para mulheres é o CASH.
```

### 11.5. Qual a sazonalidade (dia da semana) de consumo com maior volume de compras?

Nessa questão, na primeira parte tivemos que incluir uma nova coluna com o nome dos dias da semana, chamada de “day\_of\_week” para ser possível fazer a análise. Na segunda parte, foi feito um group-by com os dias da semana e a soma das quantidades de compras feitas em ordem decrescentes para que possamos concluir a análise.

```
df['invoice_date'] = pd.to_datetime(df['invoice_date'], format='%d/%m/%Y')
df['day_of_week'] = df['invoice_date'].dt.day_name()
day_of_week_sales = df.groupby('day_of_week')['quantity'].sum()
day_of_week_sales_sorted = day_of_week_sales.sort_values(ascending=False)
print(day_of_week_sales_sorted.head())
```

[41]

```
# transformar a coluna invoice_date em um objeto datetime
BD['invoice_date'] = pd.to_datetime(BD['invoice_date'], format='%d/%m/%Y')

# criar uma nova coluna para o dia da semana correspondente a cada data
BD['day_of_week'] = BD['invoice_date'].dt.day_name()

# visualizar as primeiras linhas da base de dados com a nova coluna
print(BD.head())
```

	invoice_no	customer_id	gender	age	category	quantity	price \
0	I138884	C241288	Female	28	Clothing	5	1500.40
1	I317333	C111565	Male	21	Shoes	3	1800.51
2	I127801	C266599	Male	20	Clothing	1	300.08
3	I173702	C988172	Female	66	Shoes	5	3000.85
4	I337046	C189076	Female	53	Books	4	60.60

	payment_method	invoice_date	shopping_mall	day_of_week
0	Credit Card	2022-08-05	Kanyon	Friday
1	Debit Card	2021-12-12	Forum Istanbul	Sunday
2	Cash	2021-11-09	Metrocity	Tuesday
3	Credit Card	2021-05-16	Metropol AVM	Sunday
4	Cash	2021-10-24	Kanyon	Sunday

```
[42] # agrupar os dados por dia da semana e somar as quantidades compradas para cada dia
day_of_week_sales = BD.groupby('day_of_week')['quantity'].sum()

# classificar os dados em ordem decrescente de quantidade
day_of_week_sales_sorted = day_of_week_sales.sort_values(ascending=False)

# visualizar as cinco primeiras linhas da série com as vendas por dia da semana
print(day_of_week_sales_sorted.head())
```

```
day_of_week
Monday      43568
Friday      43075
Tuesday     42822
Thursday    42469
Wednesday   42443
Name: quantity, dtype: int64
```

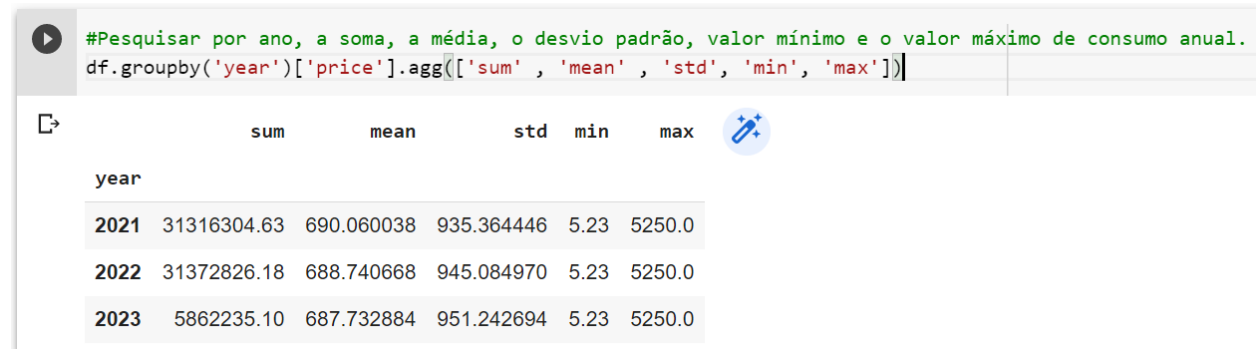


# Nessa análise vimos que o dia da semana que é realizado mais compras é na Segunda - feira (Monday).

## 11.6. Qual a média de consumo anual por gênero e faixa etária?

Foi analisado primeiramente os atributos separadamente, por ano, gênero e faixa etária, utilizando a medida de preço e posteriormente os três juntamente. Pesquisar por ano (year) aplicando as medidas de soma, média, desvio padrão, valor mínimo e o valor máximo de consumo anual, aplicou-se o seguinte código:

```
df.groupby('year')['price'].agg(['sum', 'mean', 'std', 'min', 'max'])
```

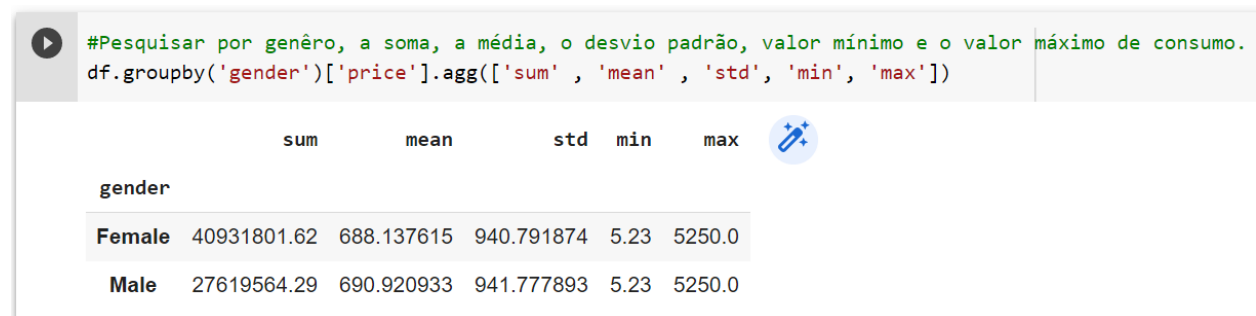


The screenshot shows a Jupyter Notebook cell with a play button icon. The code in the cell is: `#Pesquisar por ano, a soma, a média, o desvio padrão, valor mínimo e o valor máximo de consumo anual. df.groupby('year')['price'].agg(['sum', 'mean', 'std', 'min', 'max'])`. Below the code, the output is displayed as a table with columns: year, sum, mean, std, min, max. The table has three rows corresponding to the years 2021, 2022, and 2023.

	sum	mean	std	min	max
year					
2021	31316304.63	690.060038	935.364446	5.23	5250.0
2022	31372826.18	688.740668	945.084970	5.23	5250.0
2023	5862235.10	687.732884	951.242694	5.23	5250.0

Pesquisar por gênero (gender) aplicando as medidas de soma, média, desvio padrão, valor mínimo e o valor máximo de consumo por gênero, aplicou-se o seguinte código:

```
df.groupby('gender')['price'].agg(['sum', 'mean', 'std', 'min', 'max'])
```



The screenshot shows a Jupyter Notebook cell with a play button icon. The code in the cell is: `#Pesquisar por gênero, a soma, a média, o desvio padrão, valor mínimo e o valor máximo de consumo. df.groupby('gender')['price'].agg(['sum', 'mean', 'std', 'min', 'max'])`. Below the code, the output is displayed as a table with columns: gender, sum, mean, std, min, max. The table has two rows corresponding to the genders Female and Male.

	sum	mean	std	min	max
gender					
Female	40931801.62	688.137615	940.791874	5.23	5250.0
Male	27619564.29	690.920933	941.777893	5.23	5250.0

Pesquisar por faixa etária (age\_group) aplicando as medidas de soma, média, desvio padrão, valor mínimo e o valor máximo de consumo por faixa etária, aplicou-se o seguinte código:

```
df.groupby('age_group')['price'].agg(['sum', 'mean', 'std', 'min', 'max'])
```



#Pesquisar por faixa etária, a soma, a média, o desvio padrão, valor mínimo e o valor máximo de consumo.

```
df.groupby('age_group')['price'].agg(['sum', 'mean', 'std', 'min', 'max'])
```

	sum	mean	std	min	max
age_group					
18-19	2506620.09	663.127008	891.567001	5.23	5250.0
20-29	13324658.44	691.722911	940.368102	5.23	5250.0
30-39	13245827.96	686.774924	945.422467	5.23	5250.0
40-49	13376514.35	698.403088	950.336716	5.23	5250.0
50-59	12937020.02	683.377530	934.663210	5.23	5250.0
60-69	13160725.05	691.105658	944.480796	5.23	5250.0

Pesquisar os três atributos juntos atendendo a questão proposta, sendo ano, gênero e faixa etária aplicando as medidas de soma, média, desvio padrão, valor mínimo e o valor máximo de consumo, aplicou-se o seguinte código:

```
display(df.groupby(['year', 'gender', 'age_group'])['price'].agg(['sum', 'mean', 'std', 'min', 'max']))
```

#Pesquisar por ano, gênero e faixa etária, a soma, a média, o desvio padrão, valor mínimo e o valor máximo de consumo.

```
display(df.groupby(['year', 'gender', 'age_group'])['price'].agg(['sum', 'mean', 'std', 'min', 'max']))
```

			sum	mean	std	min	max
year	gender	age_group					
2021	Female	18-19	646551.95	653.082778	859.071209	5.23	5250.0
		20-29	3696981.12	706.339534	944.666687	5.23	5250.0
		30-39	3702041.49	693.396046	941.932613	5.23	5250.0
		40-49	3588539.32	687.723135	918.290513	5.23	5250.0
		50-59	3458207.76	677.947022	926.756691	5.23	5250.0
		60-69	3640398.86	690.253860	944.255374	5.23	5250.0
	Male	18-19	467429.64	671.594310	914.026383	5.23	5250.0
		20-29	2475394.75	698.474817	950.153993	5.23	5250.0
		30-39	2401661.74	695.126408	946.665292	5.23	5250.0
		40-49	2412462.08	686.528765	939.433797	5.23	5250.0
		50-59	2428602.52	682.383400	930.353885	5.23	5250.0
		60-69	2398033.40	693.474089	935.921083	5.23	5250.0

▶	2022	Female	18-19	729649.69	702.938044	941.359340	5.23	5250.0
			20-29	3648086.04	687.539774	942.534560	5.23	5250.0
			30-39	3632355.52	693.859698	961.118195	5.23	5250.0
			40-49	3662499.85	695.499402	960.911100	5.23	5250.0
			50-59	3577641.64	685.766080	942.104962	5.23	5250.0
			60-69	3480907.51	678.539476	928.727259	5.23	5250.0
	Male		18-19	463929.28	633.783169	857.348020	5.23	5250.0
			20-29	2358263.58	673.020428	922.589070	5.23	5250.0
			30-39	2449938.34	673.615161	936.224255	5.23	5250.0
			40-49	2513119.25	722.991729	974.830413	5.23	5250.0
			50-59	2375256.33	685.301884	926.410175	5.23	5250.0
			60-69	2481179.15	700.106984	966.756313	5.23	5250.0
	2023	Female	18-19	122504.97	665.787880	911.541307	5.23	5250.0
			20-29	656033.55	672.854923	970.433375	5.23	5250.0
			30-39	601694.77	634.699124	893.416867	5.23	5250.0
			40-49	689904.55	667.219101	947.673380	5.23	5250.0
			50-59	684122.62	688.945237	977.305694	5.23	5250.0
			60-69	713680.41	713.680410	969.066499	5.23	5250.0
		Male	18-19	76554.56	546.818286	762.128241	5.23	4200.0
			20-29	489899.40	699.856286	887.062880	5.23	5250.0
			30-39	458136.10	680.737147	964.442376	5.23	5250.0
			40-49	509989.30	790.681085	1039.697129	5.23	5250.0
			50-59	413189.15	694.435546	941.138342	5.23	5250.0
			60-69	446525.72	700.982292	956.205364	5.23	5250.0

### 11.7. Qual shopping possui o maior ticket médio de compras?

Foi analisado a média geral e a de todos os shoppings, para identificar aquele que obteve o maior ticket médio, aplicando a consulta ao atributo de shopping\_mall, utilizando as medidas de soma, média, desvio padrão, valor mínimo e o valor máximo de consumo, aplicou-se o seguinte código:

```
display(df.groupby(['shopping_mall'])['price'].agg(['sum' , 'mean' , 'std', 'min', 'max']))
```

```
#Média de consumo geral
df.price.mean()
```

689.2563209226097

```
#Pesquisar por shopping a soma, a média, o desvio padrão, valor mínimo e o valor máximo de consumo.
display(df.groupby(['shopping_mall'])['price'].agg(['sum', 'mean', 'std', 'min', 'max']))
```

	sum	mean	std	min	max
shopping_mall					
<b>Cevahir AVM</b>	3433671.84	687.972719	952.002169	5.23	5250.0
<b>Emaar Square Mall</b>	3390408.31	704.720081	957.855695	5.23	5250.0
<b>Forum Istanbul</b>	3336073.82	674.363012	916.138978	5.23	5250.0
<b>Istinye Park</b>	6717077.54	686.747525	932.392502	5.23	5250.0
<b>Kanyon</b>	13710755.24	691.658944	951.995108	5.23	5250.0
<b>Mall of Istanbul</b>	13851737.62	694.566395	946.272210	5.23	5250.0
<b>Metrocity</b>	10249980.07	682.831262	930.235014	5.23	5250.0
<b>Metropol AVM</b>	6937992.99	682.806120	923.461864	5.23	5250.0
<b>Viaport Outlet</b>	3414019.46	694.753655	953.746443	5.23	5250.0
<b>Zorlu Center</b>	3509649.02	691.556457	948.450171	5.23	5250.0

Para realizar a análise de identificação de qual shopping possui o maior ticket médio de compras, foi necessário realizar o agrupamento e realizar cálculos de comparação, no qual foi identificado o shopping “Emaar Square Mall” atendendo esta análise, aplicou-se o seguinte código:

```
grouped_df = df.groupby('shopping_mall')['price'].mean().reset_index()
media_geral = df['price'].mean()
diferenca_media = grouped_df['price'] - media_geral
max_categoria = grouped_df.loc[diferenca_media.idxmax(), 'shopping_mall']
print("O shopping que possui o maior ticket médio de compras é:", max_categoria)
```

```
[126] #Agrupar os dados por categoria
grouped_df = df.groupby('shopping_mall')['price'].mean().reset_index()
```

```
[127] #Média geral
media_geral = df['price'].mean()
```

```
[129] #Calcular a diferença da média geral em relação a média de cada shopping
diferenca_media = grouped_df['price'] - media_geral
```

```
[133] #Verificar qual shopping tem a média superior a média geral.
max_categoria = grouped_df.loc[diferenca_media.idxmax(), 'shopping_mall']
```

```
O shopping que possui o maior ticket médio de compras é: Emaar Square Mall
```

### 11.8. Como se dá a comparação de gastos totais por gênero e por faixa etária? E dos gastos médios?

```
#criando uma nova coluna na tabela
df2 = df
df2['consumption'] = df2['price']*df2['quantity']

df3 = df2
df3['age_group'] = 0

def age_group(age):

    if age <= 19:
        age = '18-19'
    elif 20 <= age <= 29:
        age = '20-29'
    elif 30 <= age <= 39:
        age = '30-39'
    elif 40 <= age <= 49:
        age = '40-49'
    elif 50 <= age <= 59:
        age = '50-59'
    else:
        age = '60-69'
    return age

# criar novas colunas para age_group
df3['age_group'] = df3['age'].apply(age_group)
df3['age_group'].value_counts()

#agrupando e verificando a soma e a média
df4 = df3.groupby(['gender', 'age_group'])['consumption'].agg(['sum', 'mean'])
df4 = df4.sort_values(['gender'], ascending=True)
df4 = df4.reset_index()
df4

#montagem de filtros
filt = (df4['gender'] == 'Female')
filtro = (df4['gender'] == 'Male')

df5 = df4[filt]
df6 = df4[filtro]

#criação da primeira tabela
n1 = list(df5['sum'])
n2 = list(df6['sum'])
```

```

barWidth = 0.34

r1 = np.arange(len(n1))
r2 = [x + barWidth for x in r1]

plt.bar(r1, n1, color='#6A5ACD', width = barWidth, label='Female')
plt.bar(r2, n2, color='#6495ED', width = barWidth, label='Male')

plt.xlabel('Idade')
plt.xticks([r + barWidth for r in range(len(n1))], list(df5['age_group']))
plt.ylabel('Total Gastos')
plt.title('Gasto total por faixa etária')

plt.legend(loc='lower right')
plt.show()

#criação da segunda tabela
m1 = list(df5['mean'])
m2 = list(df6['mean'])
barWidth = 0.34

s1 = np.arange(len(m1))
s2 = [x + barWidth for x in r1]

#criação da segunda tabela
plt.bar(s1, m1, color='#6A5ACD', width = barWidth, label='Female')
plt.bar(s2, m2, color='#6495ED', width = barWidth, label='Male')

plt.xlabel('Idade')
plt.xticks([r + barWidth for r in range(len(n1))], list(df5['age_group']))
plt.ylabel('Total Gastos')
plt.title('Gasto médio por faixa etária')
plt.ylim([2000, 2700])

plt.legend(loc='lower right')
plt.show()

```

```

In [3]: #criando uma nova coluna na tabela
df2 = df
df2['consumption'] = df2['price']*df2['quantity']

```

```
In [11]: #agrupando e verificando a soma e a média
df4 = df3.groupby(['gender', 'age_group'])['consumption'].agg(['sum', 'mean'])
df4 = df4.sort_values(['gender'], ascending=True)
df4 = df4.reset_index()
df4
```

Out[11]:

	gender	age_group	sum	mean
0	Female	18-19	5430011.39	2454.797193
1	Female	20-29	29580549.21	2568.870969
2	Female	30-39	28974202.84	2514.685197
3	Female	40-49	29077453.38	2524.522780
4	Female	50-59	28376447.98	2508.747943
5	Female	60-69	28768471.22	2522.664961
6	Male	18-19	3621047.50	2309.341518
7	Male	20-29	19463841.37	2512.111689
8	Male	30-39	19313955.98	2487.309205
9	Male	40-49	20156914.71	2640.067415
10	Male	50-59	19217186.16	2521.940441
11	Male	60-69	19525712.51	2556.056095

```
In [12]: #montagem de filtros
filt = (df4['gender'] == 'Female')
filtro = (df4['gender'] == 'Male')
```

```
In [13]: df5 = df4[filt]
df6 = df4[filtro]
```

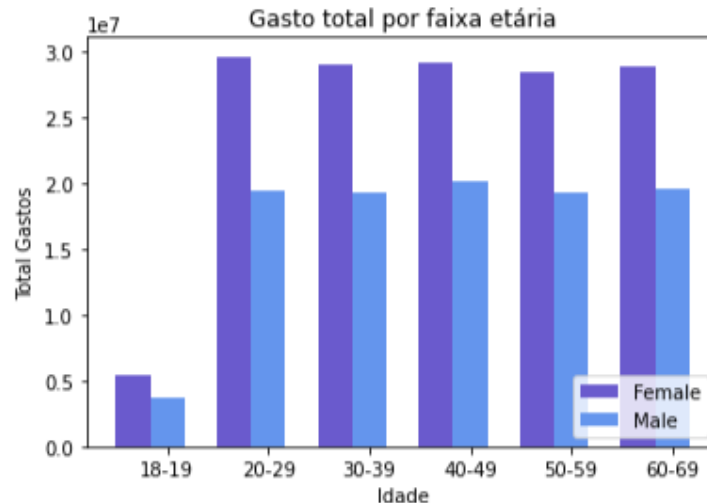
```
In [14]: #criação da primeira tabela
n1 = list(df5['sum'])
n2 = list(df6['sum'])
barWidth = 0.34
```

```
In [15]: r1 = np.arange(len(n1))
r2 = [x + barWidth for x in r1]
```

```
In [20]: plt.bar(r1, n1, color='#6A5ACD', width = barWidth, label='Female')
plt.bar(r2, n2, color='#6495ED', width = barWidth, label='Male')

plt.xlabel('Idade')
plt.xticks([r + barWidth for r in range(len(n1))], list(df5['age_group']))
plt.ylabel('Total Gastos')
plt.title('Gasto total por faixa etária')

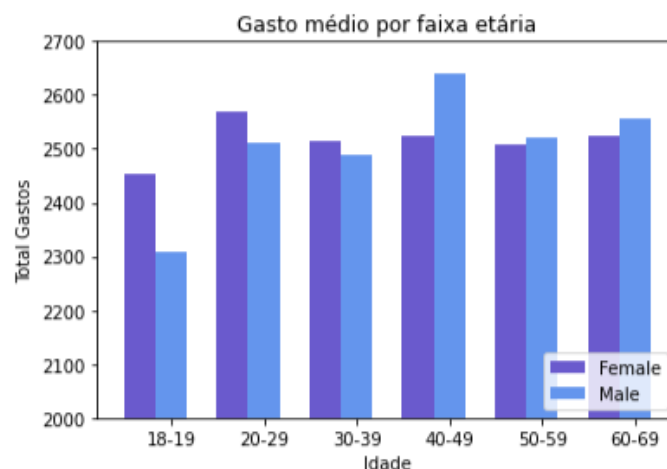
plt.legend(loc='lower right')
plt.show()
```



```
In [21]: #criação da segunda tabela
plt.bar(s1, m1, color='#6A5ACD', width = barWidth, label='Female')
plt.bar(s2, m2, color='#6495ED', width = barWidth, label='Male')

plt.xlabel('Idade')
plt.xticks([r + barWidth for r in range(len(n1))], list(df5['age_group']))
plt.ylabel('Total Gastos')
plt.title('Gasto médio por faixa etária')
plt.ylim([2000, 2700])

plt.legend(loc='lower right')
plt.show()
```



## 11.9. Qual faixa etária mais consome artigos de leitura?

```
df2 = df.groupby('category')['quantity'].agg([sum])
df2 = df2.sort_values(['sum'], ascending=False)
df2

df3 = df
df3['age_group'] = 0
df3

def age_group(age):

    if age <= 19:
        age = '18-19'
    elif 20 <= age <= 29:
        age = '20-29'
    elif 30 <= age <= 39:
        age = '30-39'
    elif 40 <= age <= 49:
        age = '40-49'
    elif 50 <= age <= 59:
        age = '50-59'
    else:
        age = '60-69'
    return age

# criar novas colunas para age_group
df3['age_group'] = df3['age'].apply(age_group)

df3['age_group'].value_counts()

#Agrupando por categoria, faixa etária e quantidade consumida
df2 = df3.groupby(['category', 'age_group'])['quantity'].agg([sum])
df2 = df2.sort_values(['category'], ascending=True)
df2 = df2.reset_index()
df2

#filtrando apenas artigos de leitura
filt = (df2['category'] == 'Books')
df4 = df2[filt]
df4 = df4.sort_values(['sum'], ascending=False)
df4

mv = (df4['sum'] == df4['sum'].max())
df5 = df4[mv]
mv2 = list(df5['age_group'])
cat_max = mv2[0]

#Obtendo a resposta:
print('A faixa etária que mais consome artigos de leitura é:', cat_max)
```



```
In [20]: #Agrupando por categoria, faixa etária e quantidade consumida
df2 = df3.groupby(['category', 'age_group'])['quantity'].agg([sum])
df2 = df2.sort_values(['category'], ascending=True)
df2 = df2.reset_index()
df2
```

Out[20]:

	category	age_group	sum
0	Books	18-19	617
1	Books	20-29	2942
2	Books	30-39	2871
3	Books	40-49	2789
4	Books	50-59	2901
5	Books	60-69	2862
6	Clothing	60-69	19852
7	Clothing	50-59	19811
8	Clothing	40-49	20146
9	Clothing	20-29	20190
10	Clothing	18-19	3883
11	Clothing	30-39	19676
12	Cosmetics	18-19	1724
13	Cosmetics	20-29	8525
14	Cosmetics	30-39	9040

```
In [23]: #filtrando apenas artigos de leitura
filt = (df2['category'] == 'Books')
df4 = df2[filt]
df4 = df4.sort_values(['sum'], ascending=False)
df4
```

Out[23]:

	category	age_group	sum
1	Books	20-29	2942
4	Books	50-59	2901
2	Books	30-39	2871
5	Books	60-69	2862
3	Books	40-49	2789
0	Books	18-19	617

```
In [40]: mv = (df4['sum'] == df4['sum'].max())
df5 = df4[mv]
mv2 = list(df5['age_group'])
cat_max = mv2[0]
```

```
In [42]: #Obtendo a resposta:
print('A faixa etária que mais consome artigos de leitura é:', cat_max)
```

A faixa etária que mais consome artigos de leitura é: 20-29

### 11.10. Qual a localidade de shopping tem o maior consumo por categoria?

```
# Agrupar o DataFrame por "region" e "category"
grupo = df.groupby(['region', 'category'])

# Calcular o total de vendas para cada categoria e shopping_mall
total_vendas = grupo['price'].sum()

# Transformar a série resultante em um DataFrame e ordenar em ordem
# decrescente de vendas
total_vendas_df =
total_vendas.reset_index().sort_values(by=['category', 'price'],
ascending=[True, False])

# Selecionar a primeira linha do DataFrame resultante para cada categoria,
# que terá a localidade de shopping com o maior consumo por categoria
maior_consumo = total_vendas_df.drop_duplicates(subset='category',
keep='first')

# Imprimir o resultado
print("Localidade de shopping com o maior consumo por categoria:")
print(maior_consumo[['category', 'region', 'price']])
```

	category	region	price
8	Books	Região Européia	120078.90
9	Clothing	Região Européia	16357060.72
10	Cosmetics	Região Européia	977303.76
11	Food & Beverage	Região Européia	122612.12
12	Shoes	Região Européia	9490488.21
13	Souvenir	Região Européia	92256.45
14	Technology	Região Européia	8305500.00
15	Toys	Região Européia	568709.12

### 11.11. Qual produto tem o maior e menor ticket médio por região?

```
# Agrupamos o DataFrame por "region" e "category"
grupo = df.groupby(['region', 'category'])
```

```

# Calcular o ticket médio para cada categoria e região
ticket_medio = grupo['price'].mean()

# Transformar a série resultante em um DataFrame e ordenar em ordem
# decrescente de ticket médio
ticket_medio_df = ticket_medio.reset_index().sort_values(by=['region',
'price'], ascending=[True, False])

# Selecionar a primeira e a última linha do DataFrame resultante para cada
# região, que terá o produto com o maior e menor ticket médio por região,
# respectivamente
maior_ticket_medio = ticket_medio_df.drop_duplicates(subset='region',
keep='first')
menor_ticket_medio = ticket_medio_df.drop_duplicates(subset='region',
keep='last')

# Imprimir o resultado
print("Produto com o maior ticket médio por região:")
print(maior_ticket_medio[['region', 'category', 'price']])
print()
print("Produto com o menor ticket médio por região:")
print(menor_ticket_medio[['region', 'category', 'price']])

```

```

Produto com o maior ticket médio por região:
      region  category  price
6  Região Asiática  Technology  3166.178737
14 Região Européia  Technology  3136.518127

```

```

Produto com o menor ticket médio por região:
      region  category  price
3  Região Asiática  Food & Beverage  15.697936
11 Região Européia  Food & Beverage  15.699375

```