

✓ EXPLORANDO A DISTRIBUIÇÃO DOS DADOS COM GRÁFICOS

```
#aqui é só para importar as imagens e visualizar
import cv2
from google.colab.patches import cv2_imshow
#Biblioteca para montar o drible
from google.colab import drive
#importando o pandas
import pandas as pd
#Maiores informações: https://pandas.pydata.org/pandas-docs/version/0.23/
import matplotlib.pyplot as plt
#Maiores informações: https://matplotlib.org/3.1.1/gallery/
import seaborn as sns
#Maiores informações: http://seaborn.pydata.org/tutorial.html
import plotly.express as px
#https://plotly.com/python/
```

Quais gráficos utilizar na análise de dados?

- <https://operdata.com.br/blog/quais-graficos-usar-em-uma-analise-de-dados/>

```
imagem = cv2.imread('/content/drive/MyDrive/Colab Notebooks/Mineração de Dados/Gráficos/Gráficos.jpg')
cv2_imshow(imagem)
```



Saiba usar o gráfico certo em cada situação

Gráfico	Tipo de gráfico	Quando usar	Exemplo de uso
	Gráfico de barras	Comparar quantidades	Comparar as vendas de diferentes produtos.
	Gráfico de linhas	Mostrar tendências ao longo do tempo	Mostrar o número de visitas num site ao longo do ano.
	Gráfico de pizza	Destacar proporções e porcentagens	Proporção entre homens e mulheres em uma população.
	Dispersão	Representar relação entre variáveis	Correlação entre o tempo de estudo e as notas do aluno.
	Histograma	Visualizar distribuição de dados	Distribuição de idade dos participantes de uma pesquisa.
	Gráfico de radar	Comparar várias categorias em várias dimensões	Avaliar o desempenho de um produto em várias áreas.
	Gráfico de bolhas	Representar dados tridimensionais	Comparar receita, custo e lucro em três dimensões.
	Gráfico de donut	Enfatizar partes específicas dentro de um todo	Mostrar distribuição de gastos.

BOXPLOT:

- Gráfico apresentado por Tukey;
- Modo rápido de visualizar a distribuição dos dados;
- Em um mesmo gráfico você consegue visualizar a média, mediana, desvio padrão a amplitude;
- Interpretando o BOXPLOT: <https://icmcjunior.com.br/como-interpretar-um-grafico-boxplot/>.

```
from google.colab import drive
drive.mount('/content/drive')
```

 Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

```
#lendo arquivo csv
df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Mineração de Dados/Gráficos/weight-height.csv')
```

```
display(df)
```



	Gender	Height	Weight
0	Male	73.847017	241.893563
1	Male	68.781904	162.310473
2	Male	74.110105	212.740856
3	Male	71.730978	220.042470
4	Male	69.881796	206.349801
...
9995	Female	66.172652	136.777454
9996	Female	67.067155	170.867906
9997	Female	63.867992	128.475319
9998	Female	69.034243	163.852461
9999	Female	61.944246	113.649103

10000 rows x 3 columns

Comece a programar ou gere código com IA.

Comece a programar ou gere código com IA.

```
df.describe()
```



	Height	Weight
count	2.000000	2.000000
mean	68.758283	184.138927
std	0.100481	15.589038
min	68.687232	173.115813
25%	68.722758	178.627370
50%	68.758283	184.138927
75%	68.793808	189.650485
max	68.829334	195.162042

```
media = df.groupby(['Gender'])['Height'].mean()
media
```



```
Gender
Female    68.687232
Male      68.829334
Name: Height, dtype: float64
```

```
mediana = df.groupby(['Gender'])['Height'].median()
mediana
```



```
Gender
Female    68.687232
Male      68.829334
Name: Height, dtype: float64
```

```
minGender = df.groupby(['Gender'])['Height'].min()
minGender
```



```
Gender
Female    54.263133
Male      58.406905
Name: Height, dtype: float64
```

```
maxGender = df.groupby(['Gender'])['Height'].max()
maxGender
```



```
Gender
Female    73.389586
```

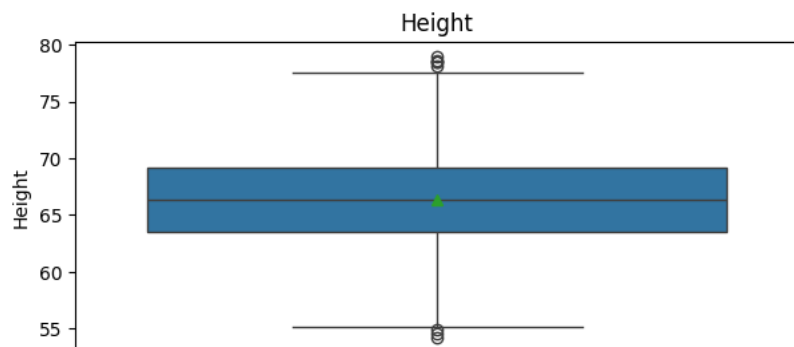
Male 78.998742
 Name: Height, dtype: float64

```
describe = df.groupby(['Gender'])['Height'].describe()
describe
```

	count	mean	std	min	25%	50%	75%	max
Gender								
Female	5000.0	63.708774	2.696284	54.263133	61.894441	63.730924	65.563565	73.389586
Male	5000.0	69.026346	2.863362	58.406905	67.174679	69.027709	70.988744	78.998742

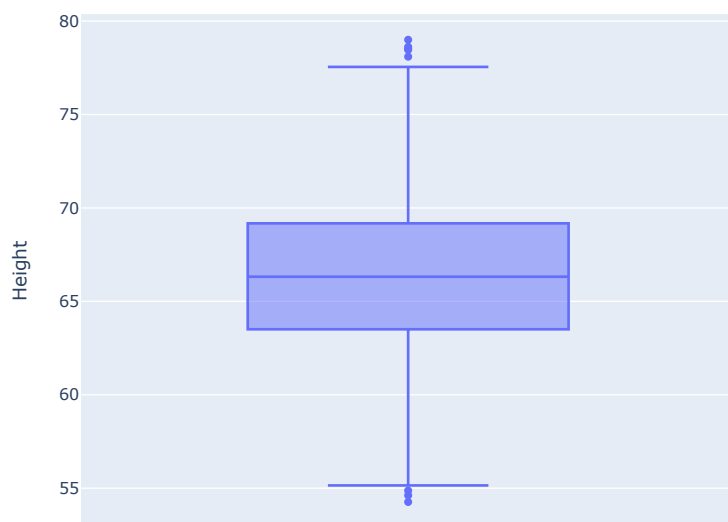
```
#Exemplo em python
plt.figure(figsize=(7,3),dpi=100)
sns.boxplot(df['Height'],showmeans=True).set_title('Height')
```

```
Text(0.5, 1.0, 'Height')
```



```
fig = px.box(df, y = "Height")
fig.show()
```

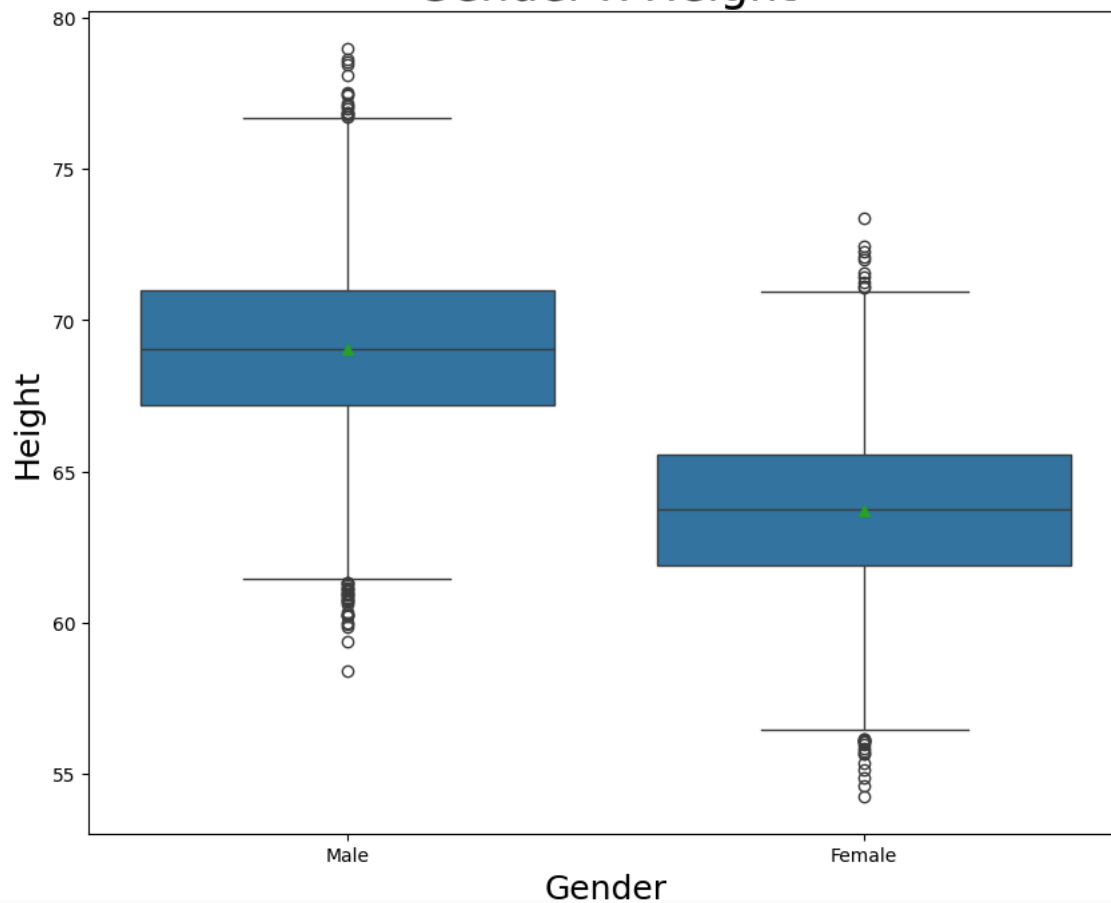
```
Text(0.5, 1.0, 'Height')
```



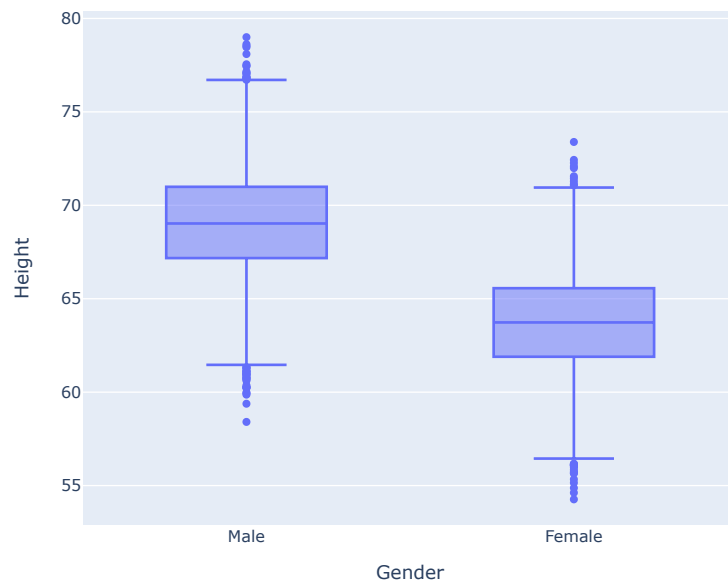
```
plt.figure(figsize=(10, 8))
ax = sns.boxplot(x='Gender',
                 y='Height',
                 data=df,
                 showmeans=True)
plt.ylabel("Height", size=18)
plt.xlabel("Gender", size=18)
plt.title("Gender x Height", size=25)
plt.show()
```



Gender x Height



```
fig = px.box(df, x="Gender", y="Height")
fig.show()
```



No boxplot temos:

- A caixa retangular dentro do gráfico que representa o intervalo entre o primeiro quartil(Q1, linha inferior) e o terceiro quartil (Q3, linha superior);
- Podemos dizer que Q1 corresponde à mediana dos valores compreendidos entre o menor valor e a mediana do conjunto de dados;
- Já Q3, corresponde à mediana dos valores compreendidos entre a mediana e o maior valor do conjunto de dados;
- A reta através da caixa mostra a mediana. A mediana também é chamada de Q2;
- O pequeno triângulo no meio da caixa é a média;

- As duas linhas dora da caixa, também chamadas de whiskes, determinam o menor e o maior valor. São utilizadas para indicar que os valores localizado fora delas podem ser considerados outliers;
- O boxplot representa 100% da base de dados;
- Do valor mínimo até o início da caixa, estão representados 25% dos dados, já dentro da caixa estão representados mais 50% dos dados, e por fim, a haste superior representa os outros 25% restantes;
- No histograma você consegue ver melhor a média e o desvio padrão. Já no boxplot se percebe um pouco melhor as medidas de quartis, mediana, amplitude, além de identificar muito bem os outliers.

HISTOGRAMAS:

- Divide valores de amostra para muitos intervalos;
- Utilizado para sumarizar a distribuição de um dado atributo numérico;
- Utilize um histograma para avaliar a forma e a dispersão dos dados, assim como os boxplot;
- Os histogramas são melhores quando o tamanho amostral for superior a 20.

Dados Assimétricos:

- Pode usar um histograma dos dados sobrepostos por uma curva normal para analisar a normalidade de seus dados;
- Uma distribuição normal é simétrica e em forma de sino, como indicada pela curva
- Quando a assimetria é à direita a mediana é inferior a média. Quando a assimetria é à esquerda a mediana é superior à média.

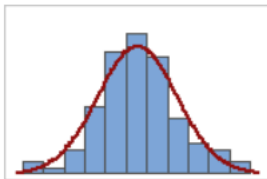
COMPLEMENTO - ASSIMETRIA

Distribuição assimétrica Negativa ou enviesada a esquerda (cauda).

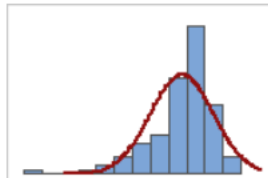
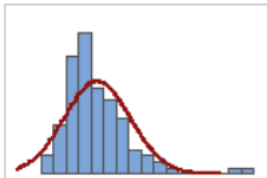
Distribuição assimétrica Positiva ou enviesada a direita (cauda).

#Exemplo

```
imagem = cv2.imread('/content/drive/MyDrive/Colab Notebooks/Mineração de Dados/Gráficos/histograma.png')
cv2_imshow(imagem)
```



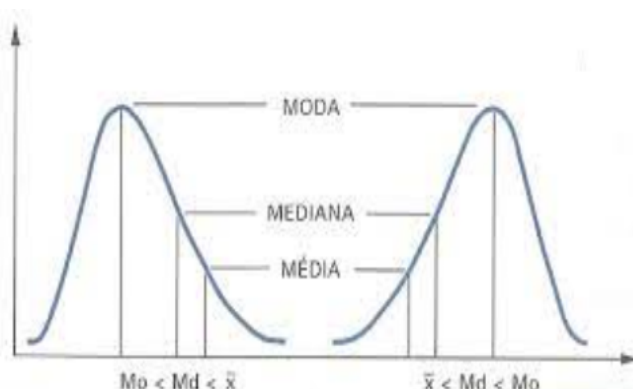
Bom ajuste



Ajuste ruim

#Exemplo assimetria

```
imagem = cv2.imread('/content/drive/MyDrive/Colab Notebooks/Mineração de Dados/Gráficos/assimetria.jpg')
bigger = cv2.resize(imagem, (500, 300))
cv2_imshow(bigger)
```



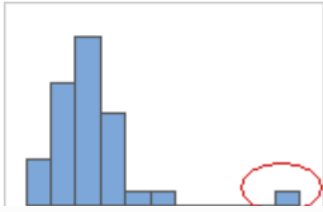
OUTLIERS:

- São valores de dados que estão distantes de outros valores de dados;
- Os outliers são mais fáceis de serem identificados em um boxplot.

Note: maiores detalhes consulte o arquivo histograma.pdf disponibilizado juntamente com o fonte.

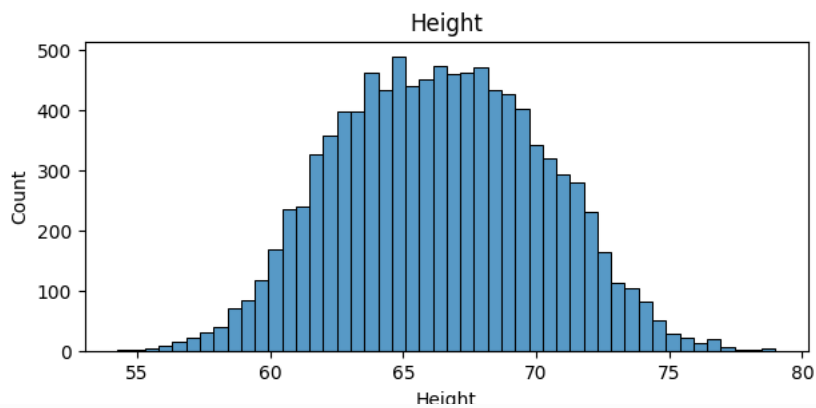
#Exemplo

```
imagem2 = cv2.imread('/content/drive/MyDrive/Colab Notebooks/Mineração de Dados/Gráficos/histogram_outlier.png')
cv2_imshow(imagem2)
```

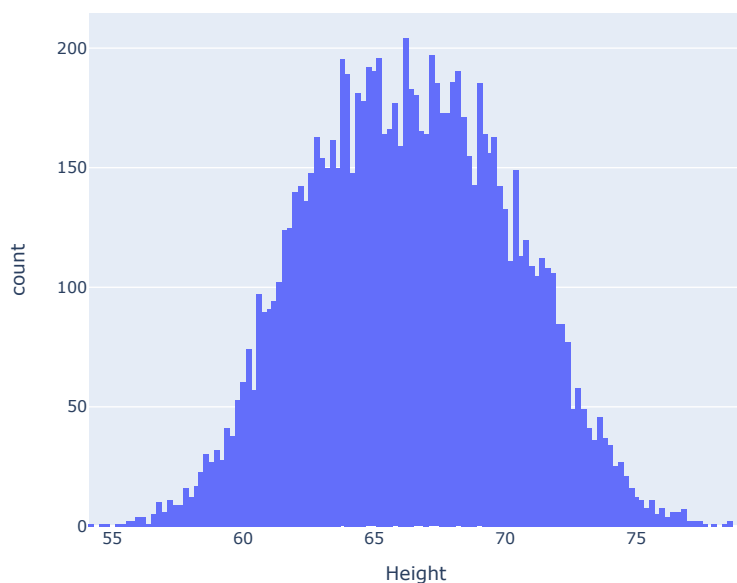


#Exemplo em python

```
plt.figure(figsize=(7,3),dpi=100)
ax = sns.histplot(df['Height']).set_title('Height')
```




```
fig = px.histogram(df, x="Height")
fig.show()
```

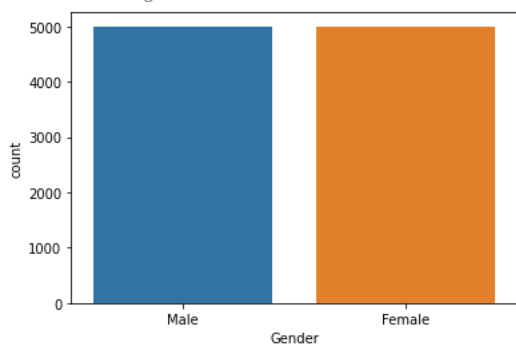
**COUNTPLOT:**

- Um gráfico simples e muito útil;
- É um gráfico de barras mostrando a contagem das variáveis;

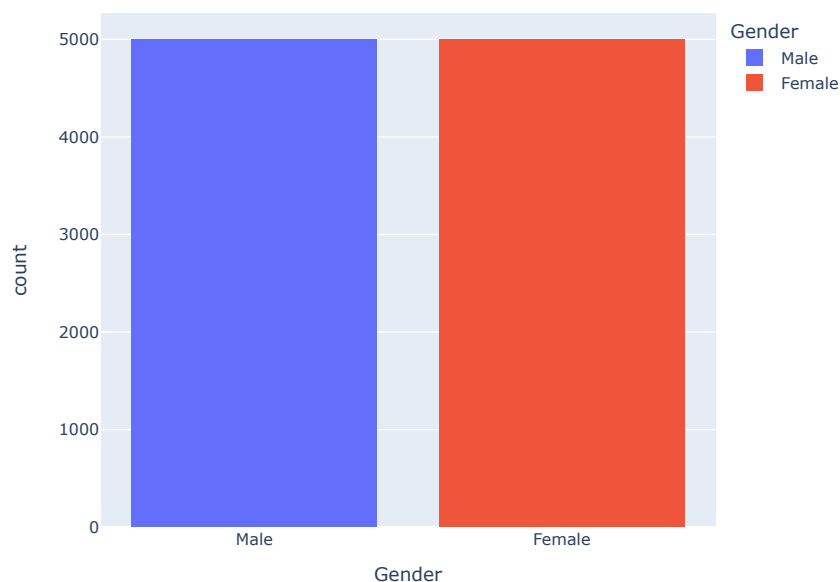
- Muito utilizado para verificar o balanceamento das variáveis alvo.

```
#Criando um countplot
ax = sns.countplot(df['Gender'])
```

 /usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. Fr
FutureWarning




```
fig = px.histogram(df, x='Gender', color='Gender')
# Exibindo o plot
fig.show()
```



Note: Ao observar o gráfico vemos que as classes estão balanceadas.

```
df.groupby('Gender').size()
```

```
 Gender
Female    5000
Male      5000
dtype: int64
```

```
#Importando os dados
df1 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Mineração de Dados/Gráficos/dados.csv', sep=';')
```

```
#Visualizar os dados
df1.head()
```


	CODIGO	MUNICIPIO	PIB	VALOREMPENHO
0	106	SANTANA DO LIVRAMENTO	12240.76	1088666.10
1	113	SANTO ANGELO	16575.82	800669.92
2	118	SAO FRANCISCO DE ASSIS	12037.61	466122.80
3	13	CACAPAVA DO SUL	13674.54	485535.86
4	120	SAO GABRIEL	19912.38	533719.86

```
display(df1)
```

	CODIGO	MUNICIPIO	PIB	VALOREMPENHO
0	106	SANTANA DO LIVRAMENTO	12240.76	1088666.10
1	113	SANTO ANGELO	16575.82	800669.92
2	118	SAO FRANCISCO DE ASSIS	12037.61	466122.80
3	13	CACAPAVA DO SUL	13674.54	485535.86
4	120	SAO GABRIEL	19912.38	533719.86
...
182	372	SANTA VITORIA DO PALMAR	27170.89	760.00
183	107	SANTA BARBARA DO SUL	29654.02	365.00
184	54	GETULIO VARGAS	16876.33	233.01
185	382	NOVA SANTA RITA	25938.38	310.58
186	60	GUARANI DAS MISSOES	24363.31	172.13

187 rows x 4 columns

```
df1.shape
```

(187, 4)

GRÁFICOS DE COLUNAS:

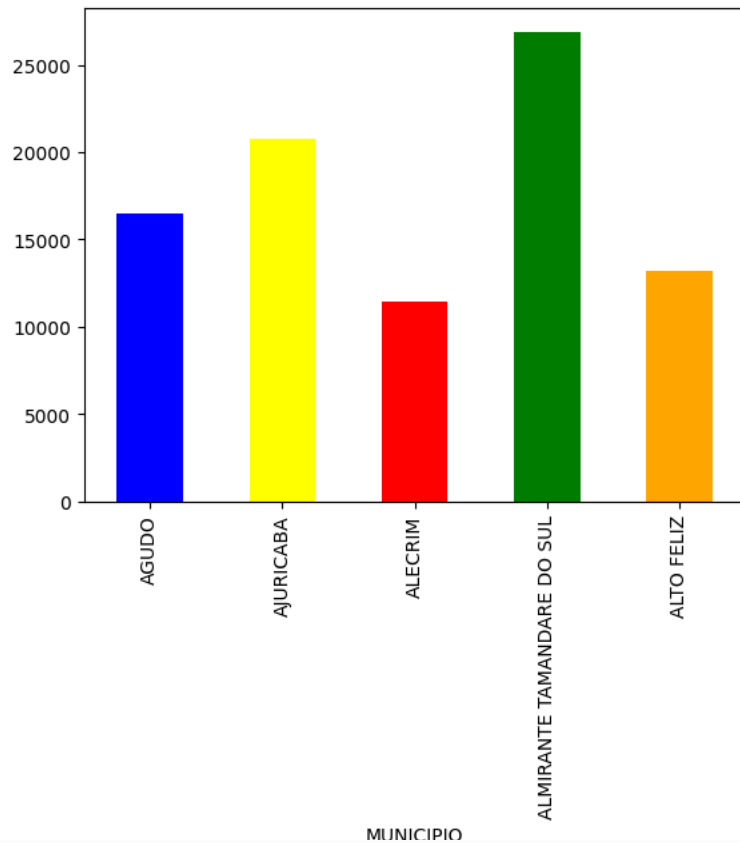
- Juntamente aos gráficos em barra, são os mais utilizados;
- Indicam, geralmente, um dado quantitativo sobre diferentes variáveis, lugares ou setores e não dependem de proporções;
- Os dados são indicados na posição vertical, enquanto as divisões qualitativas apresentam-se na posição horizontal.

```
#Vamos agrupar os dados o municio por pip
agrupado = df1.groupby(['MUNICIPIO'])['PIB'].sum()
#vamos pegar apenas o 5 primeiros registros
agrupado = agrupado.head(5)
print(agrupado)
```

MUNICIPIO	
AGUDO	16444.80
AJURICABA	20784.67
ALECRIM	11431.18
ALMIRANTE TAMANDARE DO SUL	26895.37
ALTO FELIZ	13209.54
Name: PIB, dtype: float64	

```
agrupado.plot.bar(color = ['blue','yellow','red','green','orange'])
```

<Axes: xlabel='MUNICIPIO'>

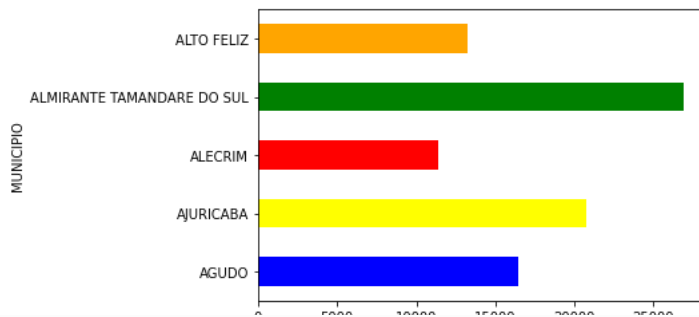


GRÁFICOS DE BARRAS:

- Possuem basicamente a mesma função dos gráficos em colunas;
- Contudo, os dados são dispostos na posição horizontal e as informações e divisões na posição vertical.

```
agrupado.plot.barh(color = ['blue','yellow','red','green','orange'])
```

<matplotlib.axes._subplots.AxesSubplot at 0x7fb534cf0050>

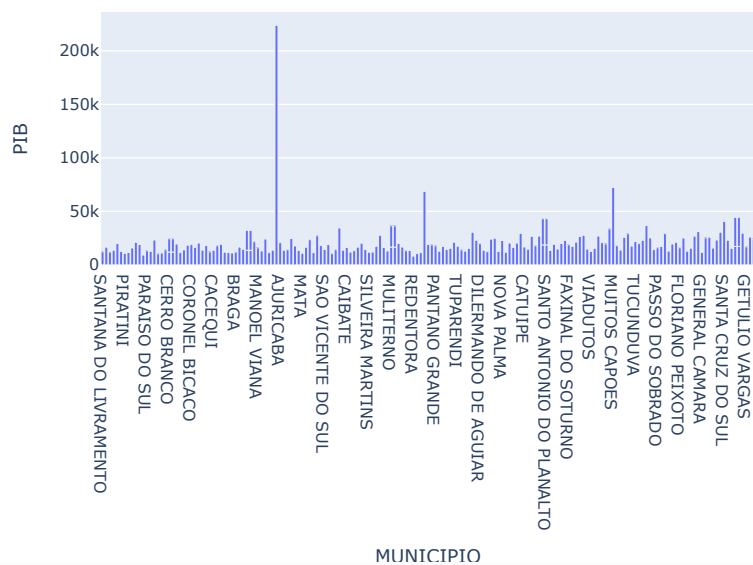


```
fig = px.bar(df1, x='MUNICIPIO', y='PIB',
             title='PIB por Município' )
```

```
# Exibindo o gráfico
fig.show()
```



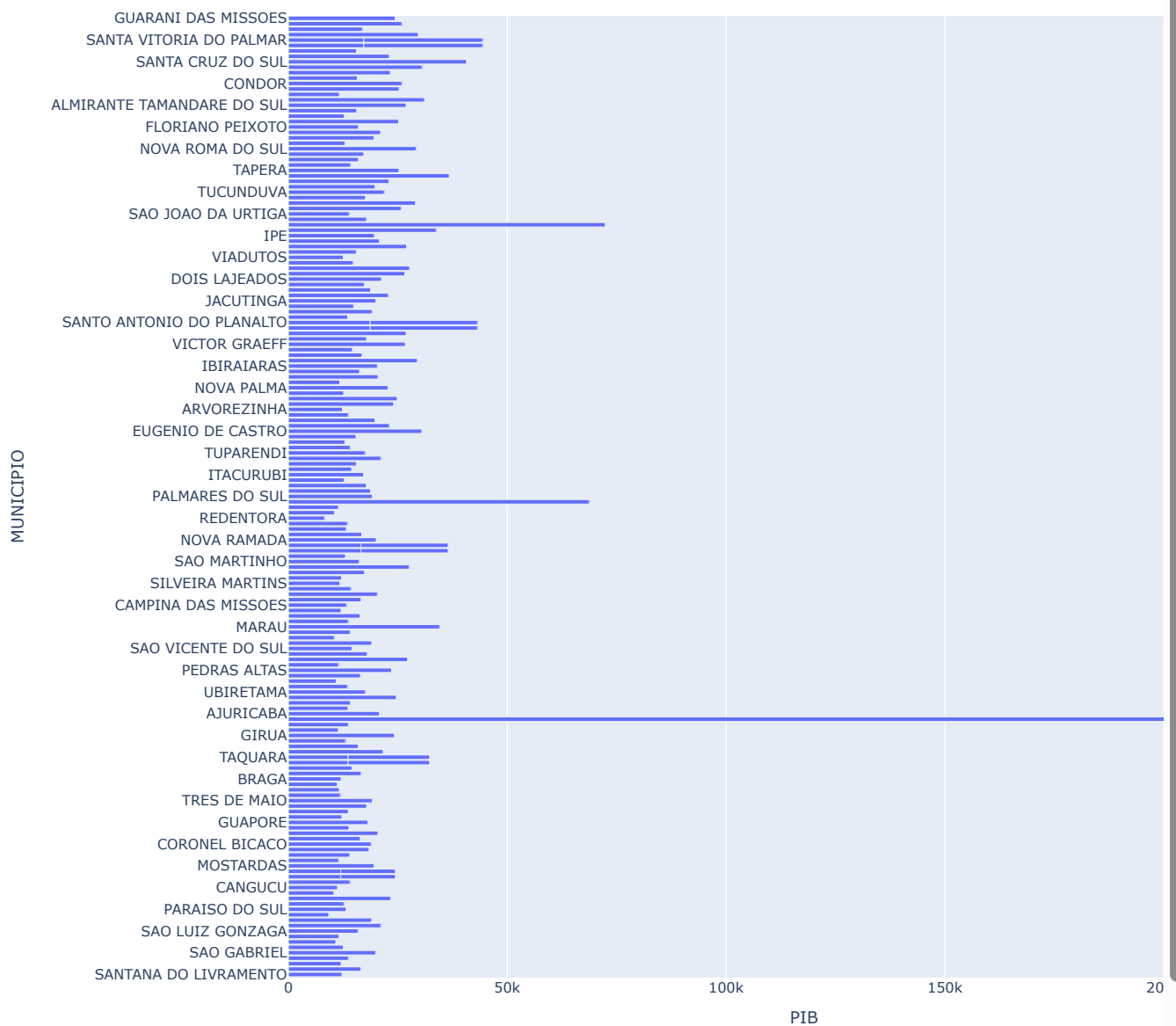
PIB por Município



```
fig = px.bar(df1, x='PIB', y='MUNICIPIO', title='PIB por Município', orientation='h')
#fig = px.bar(df1, x='PIB', y='MUNICIPIO', title='PIB por Município', orientation='h', barmode='group')
fig.update_layout(width=1200, height=1000)
# Show the plot
fig.show()
```



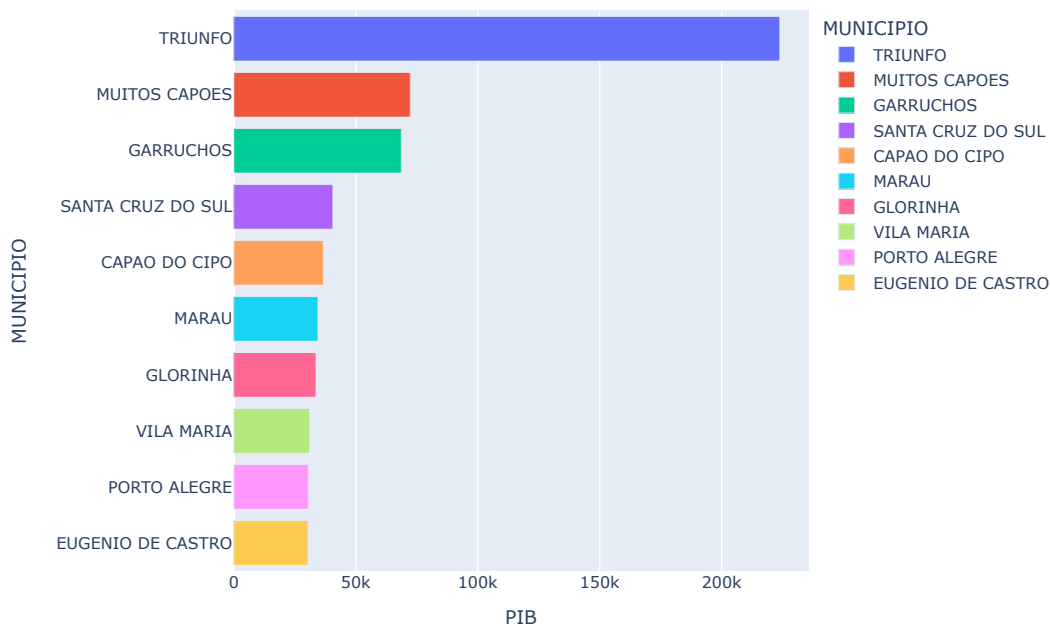
PIB por Município



```
#Os 10 melhores PIBS
maiores = df1.nlargest(10, 'PIB')
maiores
#Vou criar o gráfico
#Aqui deu certo usar esta técnica, pois eu não agrupei
fig = px.bar(maiores, x='PIB', y='MUNICIPIO', title='PIB por Município', orientation='h', color='MUNICIPIO')
fig.update_layout(width=800, height=600)
fig.show()
```



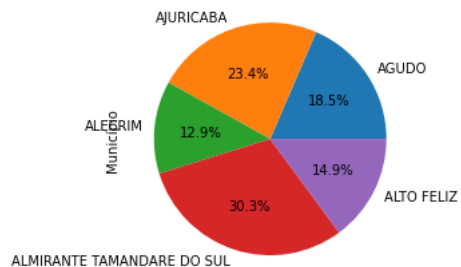
PIB por Município



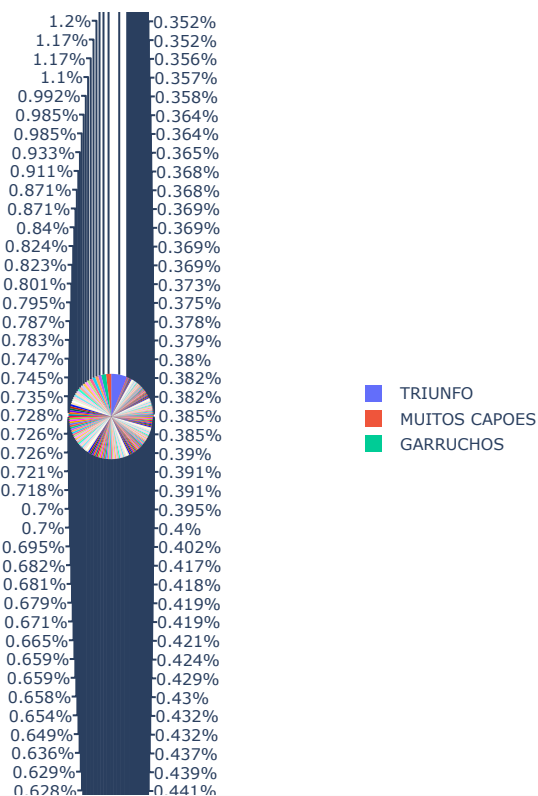
GRÁFICOS EM PIZZA:

- É um tipo de gráfico, também muito utilizado, indicado para expressar uma relação de proporcionalidade;
- Nesse todos os dados somados compõem o todo de um dado aspecto da realidade.

```
agrupado.plot.pie(autopct="%.1f%%", label = 'Município');
```

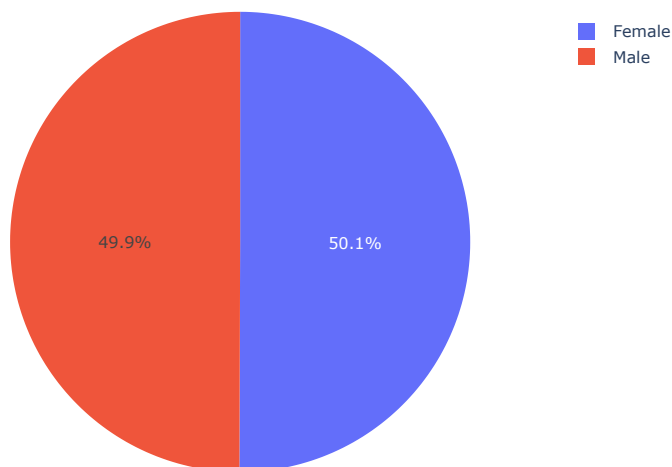


```
fig = px.pie(df1, values='PIB', names='MUNICIPIO')
fig.update_layout(width=800, height=600)
fig.show()
```

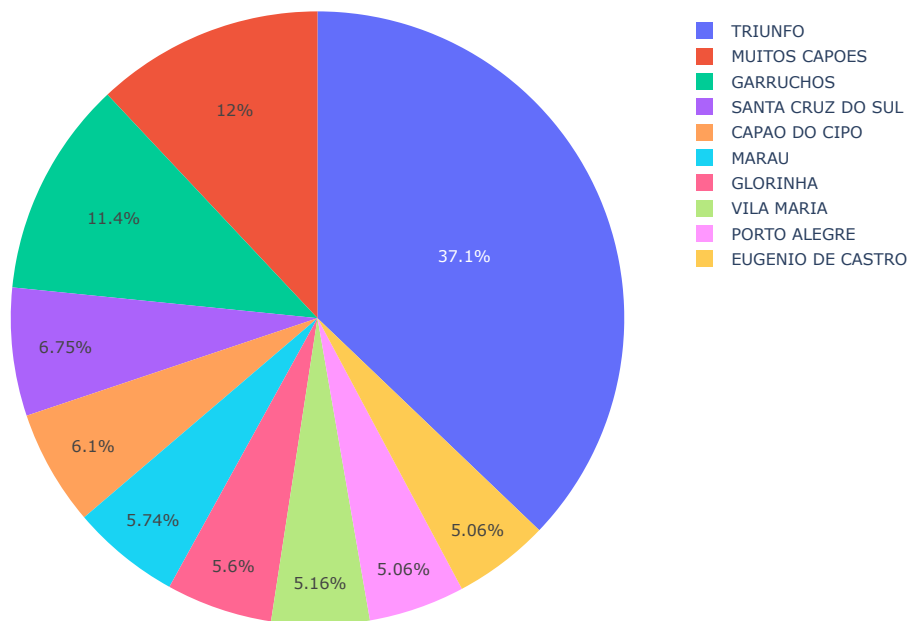


```
import plotly.graph_objects as go
#fig = go.Figure(media=[go.Pie(labels='Gender', values='Height')])
#fig.show()

fig = go.Figure(data=[go.Pie(labels=["Male", "Female"], values=media)])
fig.show()
```



```
#Os 10 maiores PIBS
maiores = df1.nlargest(10, 'PIB')
maiores
#Vou criar o gráfico
#Aqui deu certo usar esta técnica, pois eu não agrupei
fig = px.pie(maiores, values='PIB', names='MUNICIPIO')
fig.update_layout(width=800, height=600)
fig.show()
```



SCATTER PLOT - DISPERSÃO (Determina se tem uma correlação linear)

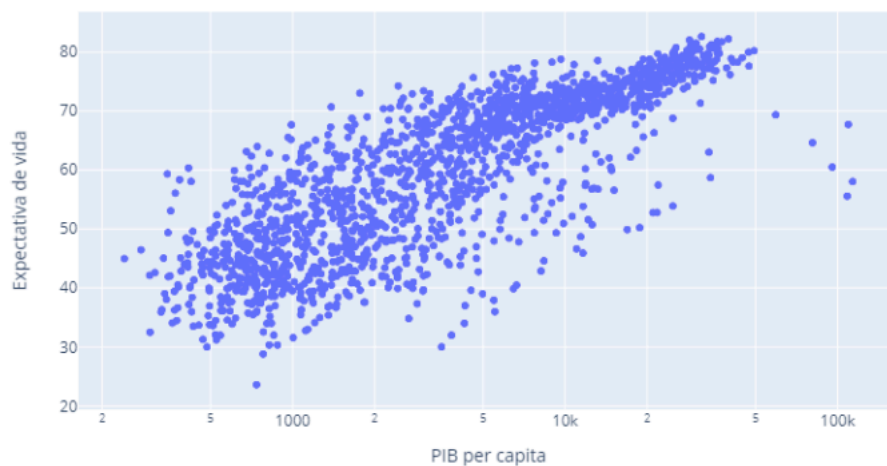
- Os gráficos de dispersão ou Scatter plot são representações gráficas do relacionamento entre duas variáveis numéricas;
- O Scatter plot utiliza pontos para representar essa relação;
- Cada ponto representa o valor de uma variável no eixo horizontal e o valor de outra variável no eixo vertical.

#Exemplo

```
imagem = cv2.imread('/content/drive/MyDrive/Colab Notebooks/Mineração de Dados/Gráficos/scat-768x432.png')
cv2_imshow(imagem)
```



PIB per capita X Expectativa de vida



Quando usar:

- Pode ser usado para verificar se existe uma relação entre causa e efeito entre duas variáveis numéricas;
- Contudo, não significa que uma variável causa efeito na outra, mas apenas se existe uma relação e qual intensidade entre essa relação;
- A relação entre duas variáveis pode ser positiva, negativa ou neutra, linear ou não linear.

#Exemplo

```
imagem = cv2.imread('/content/drive/MyDrive/Colab Notebooks/Mineração de Dados/Gráficos/cor.png')
cv2_imshow(imagem)
```

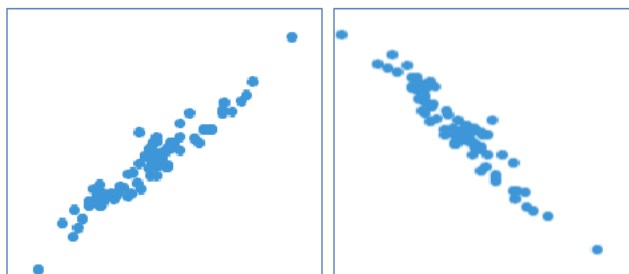


Figura: Relação linear positiva (esquerda) e negativa (direita)

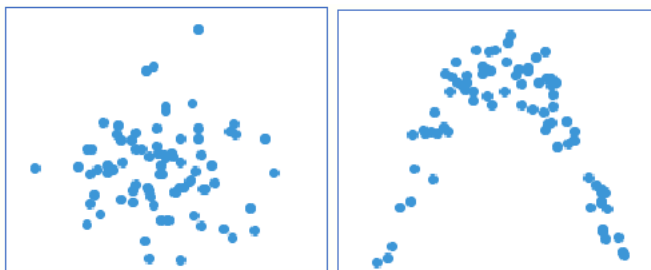


Figura: Relação neutra (esquerda) e não linear (direita)

Outras Informações:

- O gráfico de dispersão não mostra apenas o valor individualmente, mas mostra os dados como um todo;
- É útil para identificar outros padrões nos dados, como outlier (pontos extremos) ou possíveis grupos entre os dados.

#Exemplo

```
imagem = cv2.imread('/content/drive/MyDrive/Colab Notebooks/Mineração de Dados/Gráficos/cor1.png')
cv2.imshow(imagem)
```

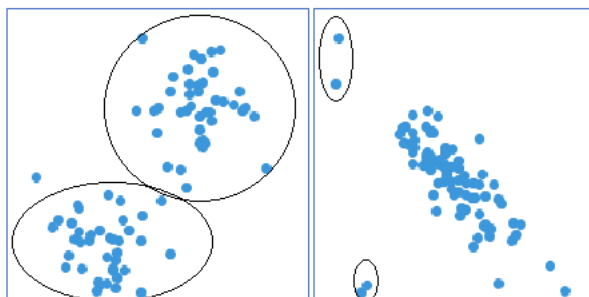
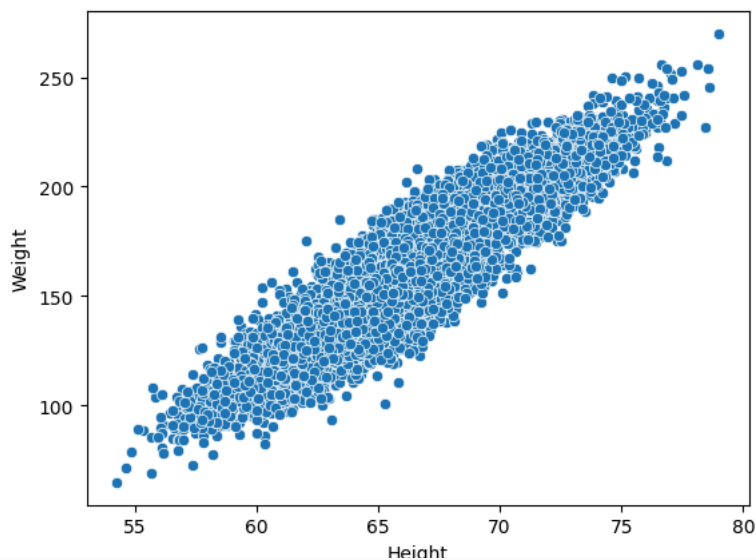


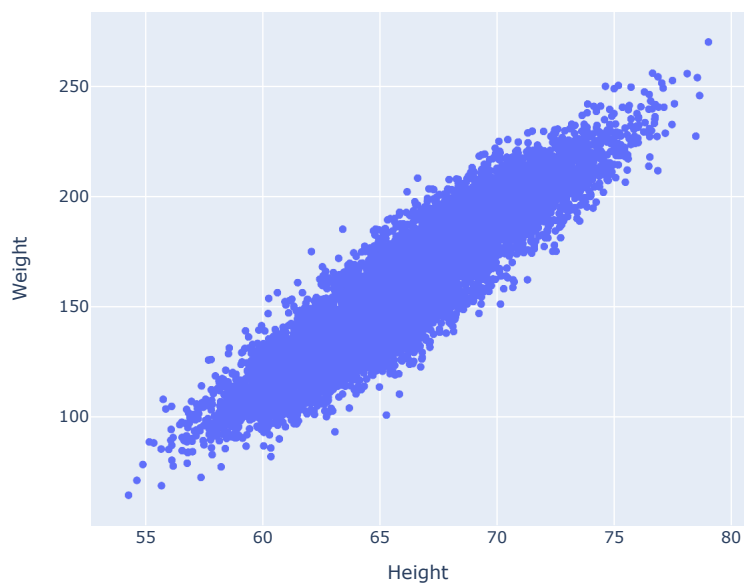
Figura: Dados com dois grupos (esquerda) e gráfico com outliers (direita)

#Implementação em python

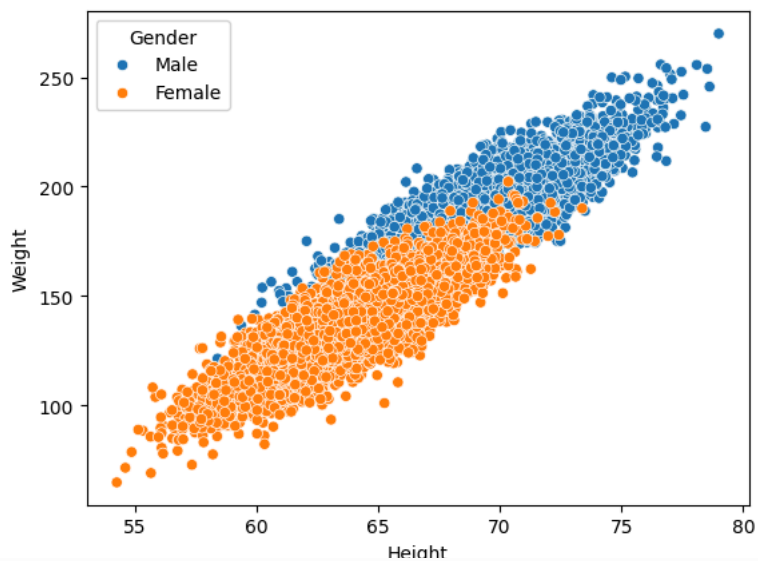
```
ax = sns.scatterplot(x='Height', y='Weight', data=df)
```



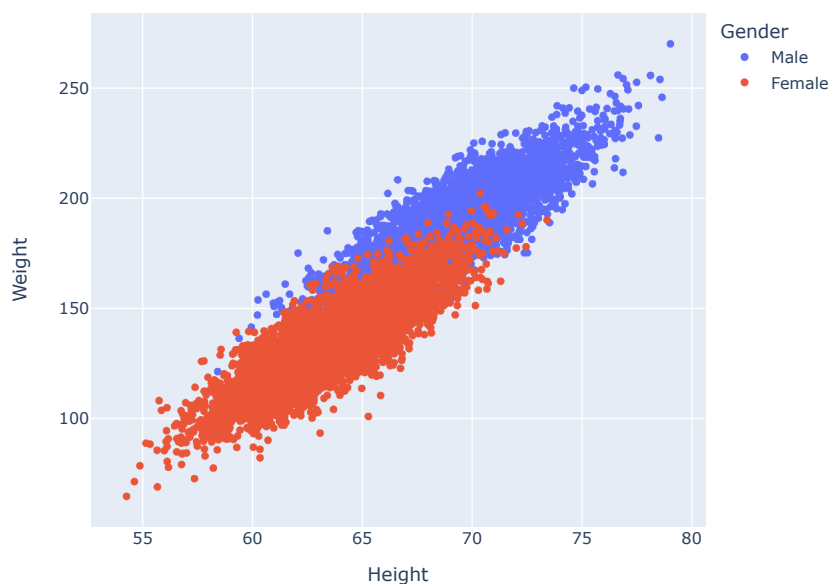

```
fig = px.scatter(df,x='Height', y='Weight')  
fig.show()
```



```
#Implementação em python colorido  
ax = sns.scatterplot(x='Height', y='Weight', hue = 'Gender', data=df)
```



```
fig = px.scatter(df, x="Height", y="Weight", color="Gender")  
fig.show()
```



HEATMAP COM CORRELAÇÃO

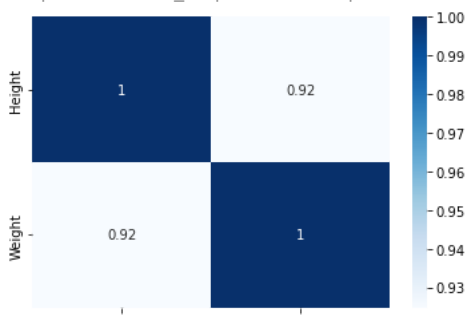
- O heatmap é um gráfico muito útil para identificar padrões;
- É utilizado principalmente quando temos muitas variáveis no gráfico;
- Iremos utilizar para estabelecer a correção entre as variáveis.

#Exemplo em python

```
sns.heatmap(df.corr(method = 'pearson'), cmap= 'Blues', annot = True)
```



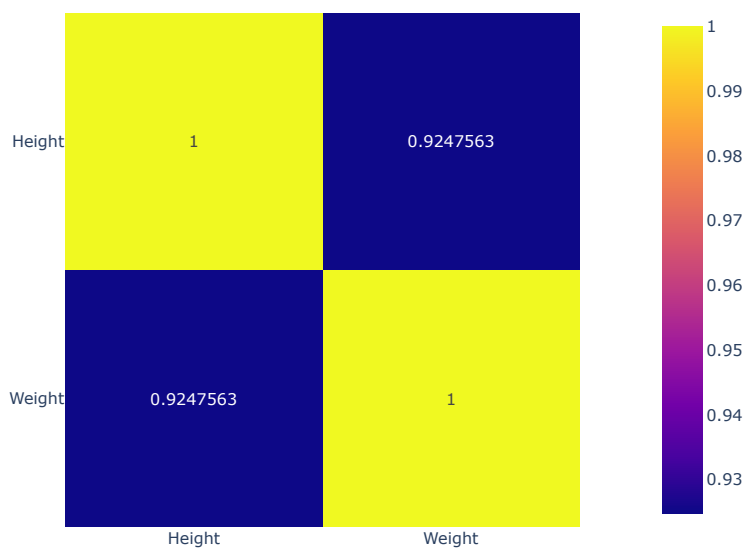
<matplotlib.axes._subplots.AxesSubplot at 0x7fbccfb6a790>



```
fig = px.imshow(df.corr(method = 'pearson'), text_auto=True)  
fig.show()
```

<ipython-input-55-4c01b843d546>:1: FutureWarning:

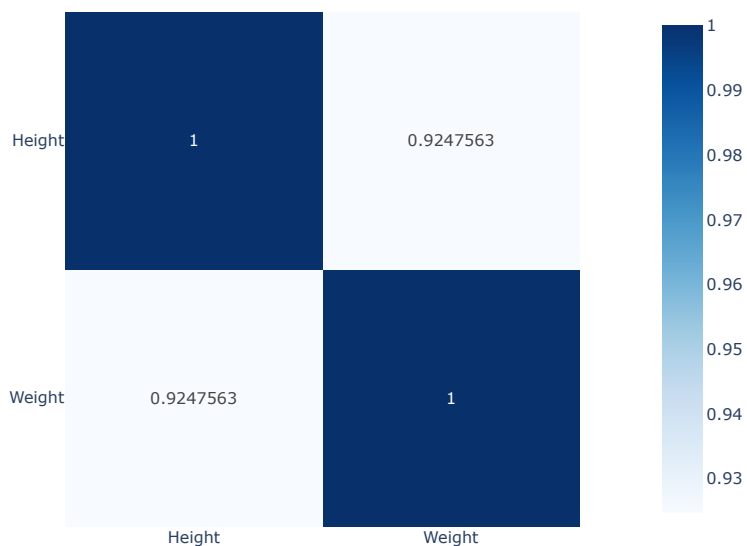
The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid



```
fig = px.imshow(df.corr(method = 'pearson'), text_auto=True,color_continuous_scale='Blues')  
fig.show()  
#Bluered_r
```

<ipython-input-65-f32bf354c226>:1: FutureWarning:

The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid

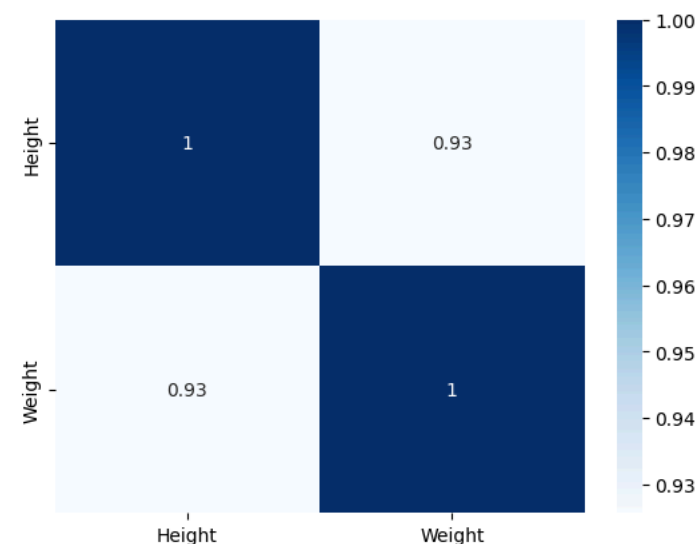


```
#Exemplo em python  
sns.heatmap(df.corr(method = 'spearman'), cmap= 'Blues', annot = True)
```

 <ipython-input-67-32dd81a0b3ad>:2: FutureWarning:

The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid

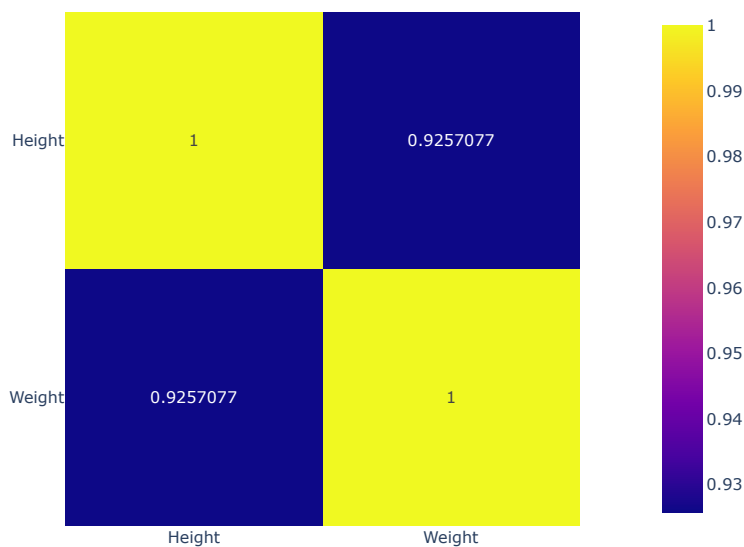
<Axes: >



```
fig = px.imshow(df.corr(method = 'spearman'), text_auto=True)
fig.show()
```

 <ipython-input-66-cff58a1937b3>:1: FutureWarning:

The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid



```
#outro exemplo
#Importando os dados
df2 = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Mineração de Dados/Gráficos/Credit.csv', sep=',')

df2.head()
```

	checking_status	duration	credit_history	purpose	credit_amount	savings_status	employment	installment_commitment	pr
0	<0	6	'critical/other existing credit'	radio/tv	1169	'no known savings'	>=7		4
1	0<=X<200	48	'existing paid'	radio/tv	5951	<100	1<=X<4		2
2	'no checking'	12	'critical/other existing credit'	education	2096	<100	4<=X<7		2
3	<0	42	'existing paid'	furniture/equipment	7882	<100	4<=X<7		2
4	<0	24	'delayed previously'	'new car'	4870	<100	1<=X<4		3

5 rows × 21 columns

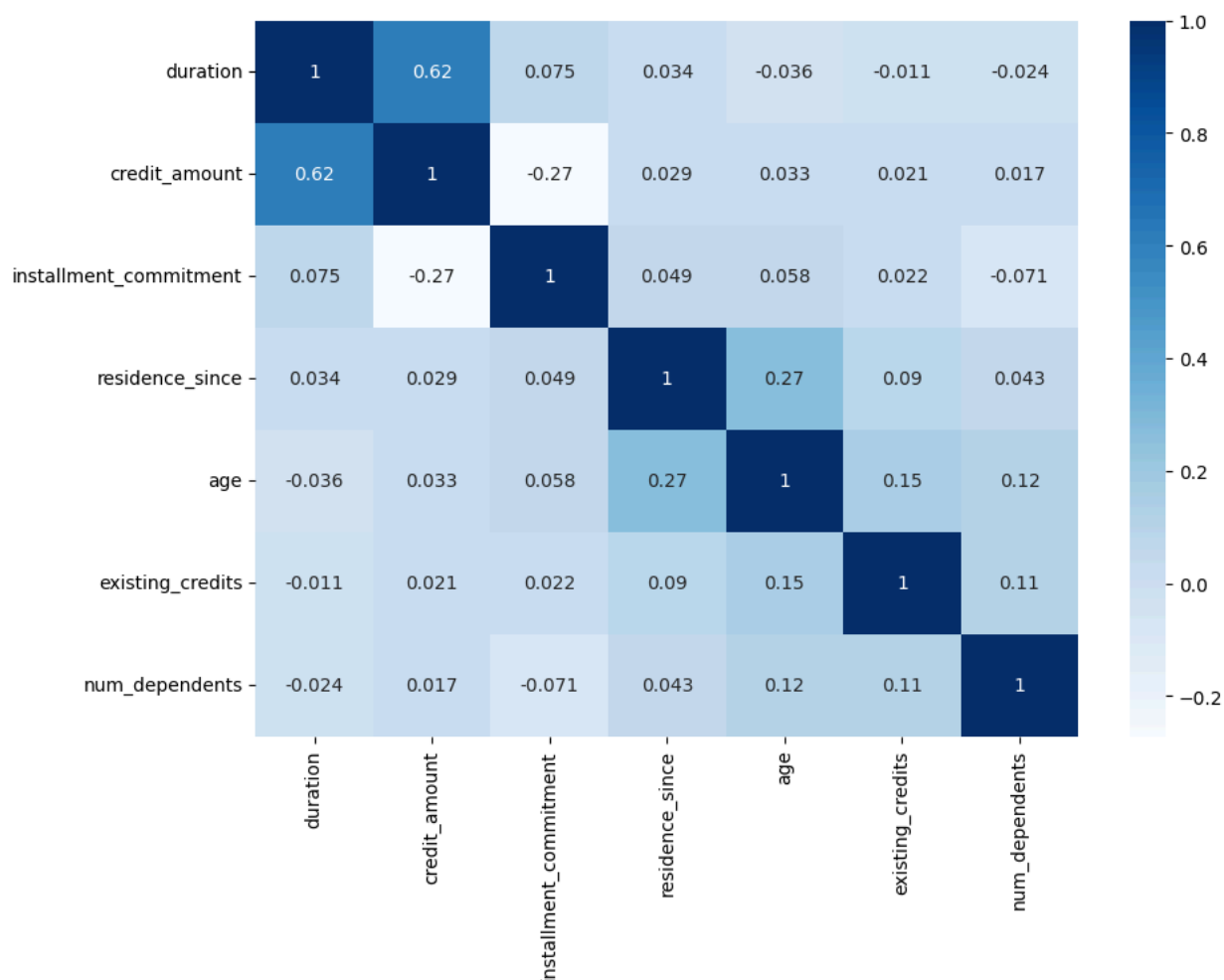
df2.shape

(1000, 21)

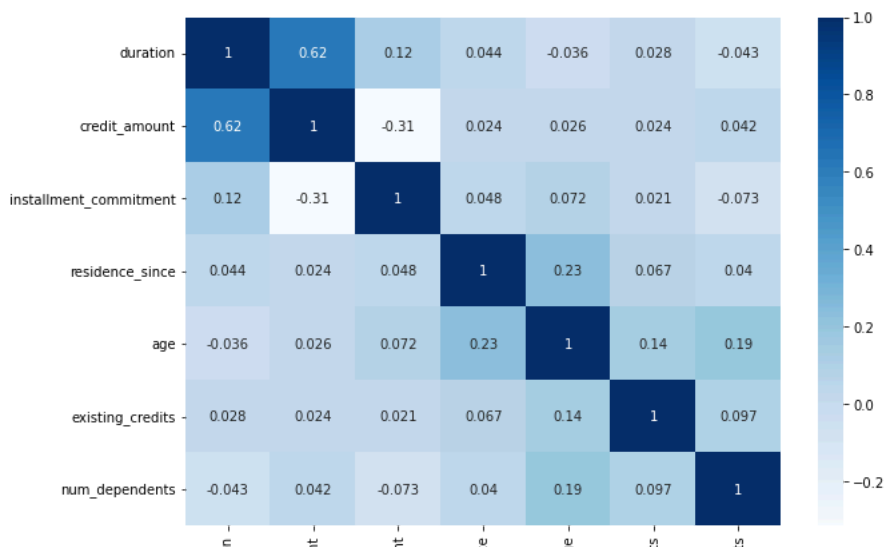
```
#Exemplo em python
plt.figure(figsize=(10,7))
g = sns.heatmap(df2.corr(method='pearson'), cmap= 'Blues', annot = True)
```

```
<ipython-input-71-aa06bad1e3b0>:3: FutureWarning:
```

The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid



```
plt.figure(figsize=(10,7))
g = sns.heatmap(df2.corr(method='spearman'), cmap= 'Blues', annot = True)
```

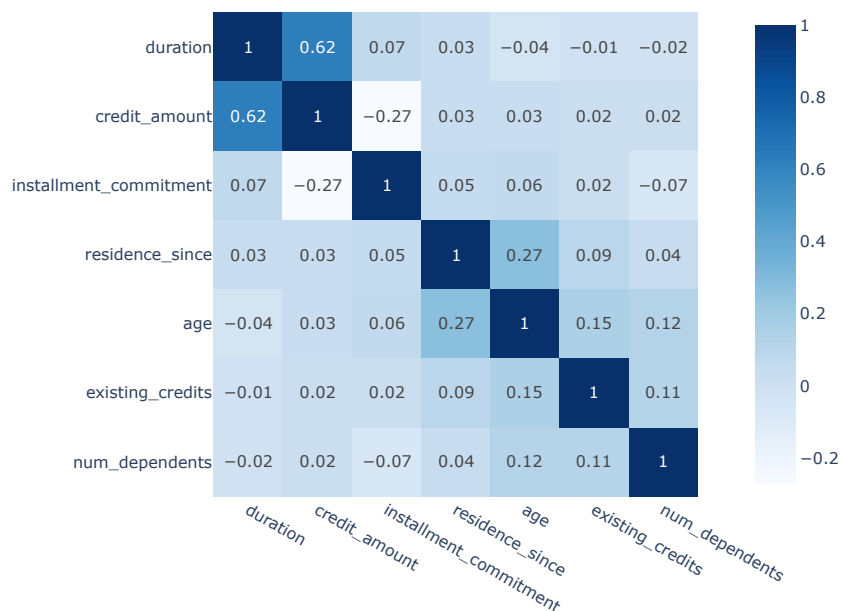


```
fig = px.imshow(round(df2.corr(method = 'pearson'),2), text_auto=True,color_continuous_scale='Blues')
fig.show()
```



<ipython-input-76-d870c030dcc0>:1: FutureWarning:

The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid



```
fig = px.imshow(round(df2.corr(method = 'kendall'),2), text_auto=True,color_continuous_scale='Blues')
fig.show()
```



<ipython-input-77-c34b4aac2cbc>:1: FutureWarning:

The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid

