

```
In [1]: pip install pandas numpy matplotlib scipy scikit-learn
```

```
Requirement already satisfied: pandas in c:\users\ssd256\anaconda3\lib\site-packages (2.2.3)
Requirement already satisfied: numpy in c:\users\ssd256\anaconda3\lib\site-packages (2.1.3)
Requirement already satisfied: matplotlib in c:\users\ssd256\anaconda3\lib\site-packages (3.10.0)
Requirement already satisfied: scipy in c:\users\ssd256\anaconda3\lib\site-packages (1.15.3)
Requirement already satisfied: scikit-learn in c:\users\ssd256\anaconda3\lib\site-packages (1.6.1)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\ssd256\anaconda3\lib\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\ssd256\anaconda3\lib\site-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\ssd256\anaconda3\lib\site-packages (from pandas) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\ssd256\anaconda3\lib\site-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\users\ssd256\anaconda3\lib\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\ssd256\anaconda3\lib\site-packages (from matplotlib) (4.55.3)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\ssd256\anaconda3\lib\site-packages (from matplotlib) (1.4.8)
Requirement already satisfied: packaging>=20.0 in c:\users\ssd256\anaconda3\lib\site-packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in c:\users\ssd256\anaconda3\lib\site-packages (from matplotlib) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\ssd256\anaconda3\lib\site-packages (from matplotlib) (3.2.0)
Requirement already satisfied: joblib>=1.2.0 in c:\users\ssd256\anaconda3\lib\site-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in c:\users\ssd256\anaconda3\lib\site-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: six>=1.5 in c:\users\ssd256\anaconda3\lib\site-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
Note: you may need to restart the kernel to use updated packages.
```

Trabalho de Estatística – NHANES

Pós-graduação: MBA em Data Science & Advanced Analytics

Disciplina: Applied Statistics

Aluno: Douglas Wesley Moraes de Souza

RA: 2503886

```
In [ ]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
from sklearn.preprocessing import MinMaxScaler
```

```
# Lendo o arquivo CSV
df = pd.read_csv("nhanes_2015_2016.csv")

df.head()
```

Out[]:

	ID_Participante	ConsumiuAlcool12Meses	FrequenciaConsumoAlcool	QuantidadeBebida
0	83732	1.0	NaN	1
1	83733	1.0	NaN	6
2	83734	1.0	NaN	NaN
3	83735	2.0	1.0	1
4	83736	2.0	1.0	1

5 rows × 28 columns

1. Variáveis categóricas e níveis de medida

Para este item, foram escolhidas as seguintes variáveis do dataset NHANES:

- **Sexo:** variável categórica do tipo **nominal**, pois suas categorias não possuem ordenação natural.
- **Escolaridade:** variável categórica do tipo **ordinal**, pois representa níveis crescentes de formação educacional.

Em seguida, foi construída uma tabela de contingência entre essas variáveis e um gráfico de barras para representar a relação entre elas.

In []:

```
# Seleção das variáveis categóricas
var_nominal = "Sexo"
var_ordinal = "Escolaridade"

# Remove valores ausentes
df_cat = df[[var_nominal, var_ordinal]].dropna()

# Tabela de contingência
tabela_contingencia = pd.crosstab(df_cat[var_nominal], df_cat[var_ordinal])
tabela_contingencia
```

Out[]:

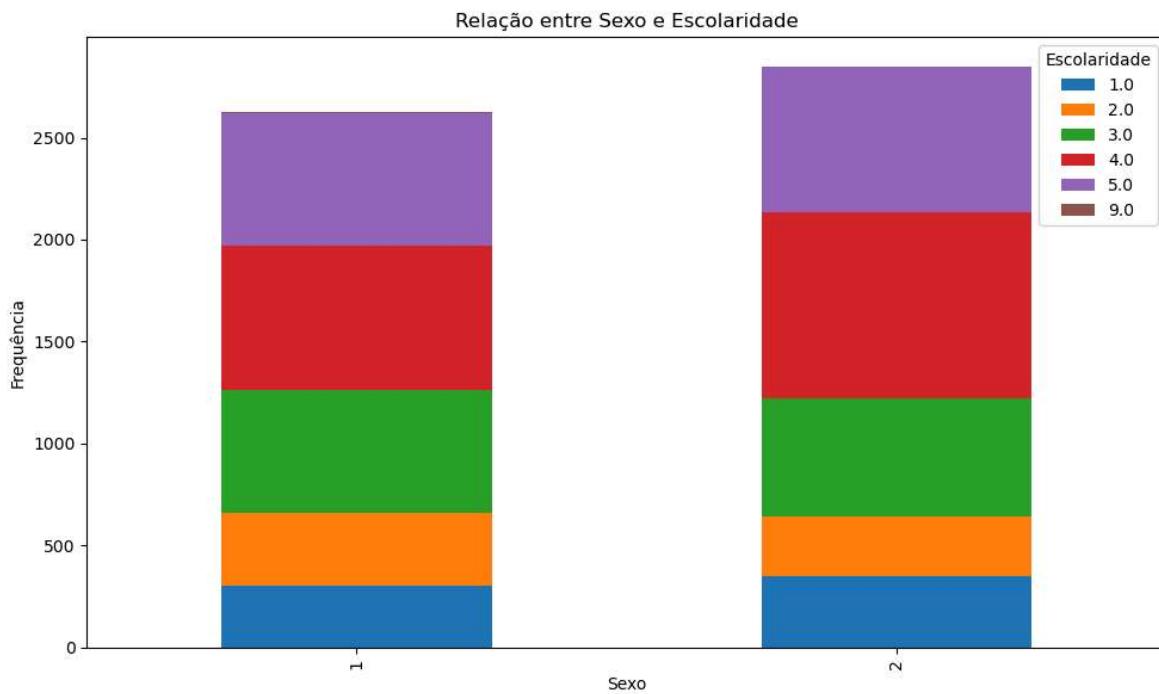
Escolaridade	1.0	2.0	3.0	4.0	5.0	9.0
Sexo						
1	306	352	603	712	649	2
2	349	291	583	909	717	1

In []:

```
# Gráfico de barras da tabela de contingência
tabela_contingencia.plot(kind="bar", stacked=True, figsize=(10, 6))

plt.title("Relação entre Sexo e Escolaridade")
```

```
plt.xlabel("Sexo")
plt.ylabel("Frequência")
plt.legend(title="Escolaridade")
plt.tight_layout()
plt.show()
```



Interpretação

A tabela de contingência e o gráfico de barras mostram como os níveis de escolaridade estão distribuídos entre os diferentes sexos. É possível observar diferenças no perfil educacional entre os grupos, o que reforça a importância de analisar variáveis categóricas em conjunto para identificar padrões sociais e demográficos.

2. Tendência central em variáveis assimétricas

Neste item, foi selecionada a variável contínua **IMC (Índice de Massa Corporal)**, que apresenta assimetria à direita. Foram calculadas as medidas de tendência central (média, mediana e moda) e construído o histograma da distribuição para análise visual.

```
In [10]: # Selecionando a variável IMC
imc = df["IMC"].dropna()

# Medidas de tendência central
media_imc = imc.mean()
mediana_imc = imc.median()
moda_imc = imc.mode().iloc[0]

media_imc, mediana_imc, moda_imc
```

```
Out[10]: (np.float64(29.382197103497), 28.3, np.float64(29.1))
```

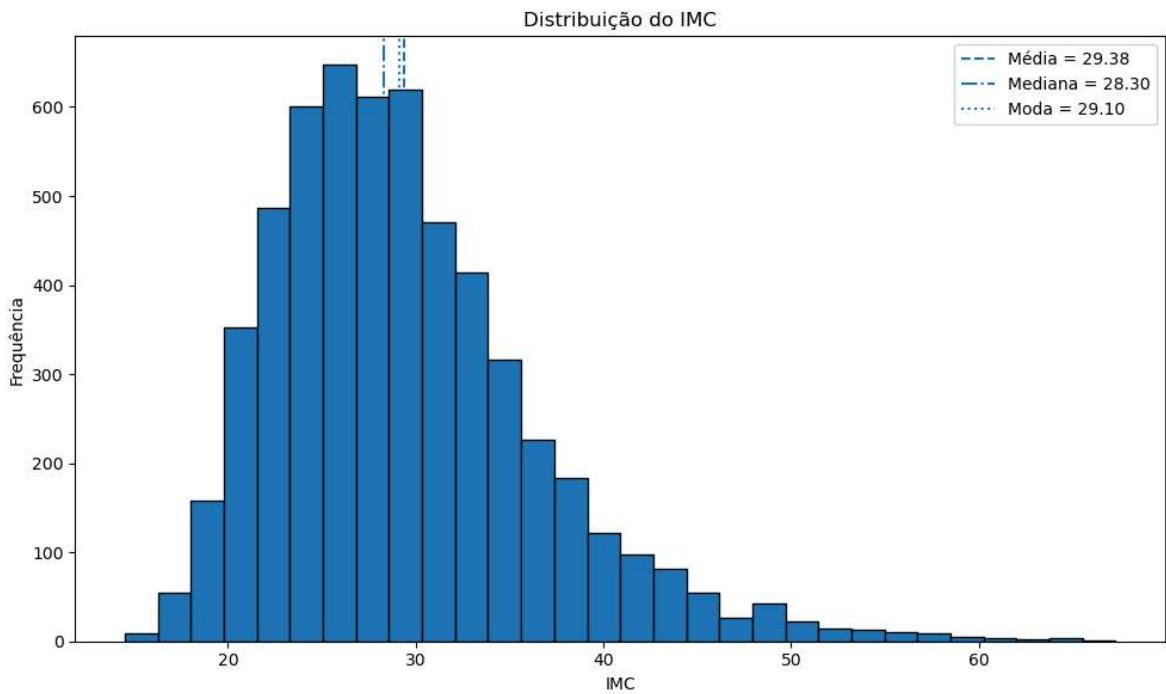
```
In [11]: plt.figure(figsize=(10, 6))
plt.hist(imc, bins=30, edgecolor="black")
```

```

plt.axvline(media_imc, linestyle="--", label=f"Média = {media_imc:.2f}")
plt.axvline(mediana_imc, linestyle="-.-", label=f"Mediana = {mediana_imc:.2f}")
plt.axvline(modas_imc, linestyle=":", label=f"Moda = {modas_imc:.2f}")

plt.title("Distribuição do IMC")
plt.xlabel("IMC")
plt.ylabel("Frequência")
plt.legend()
plt.tight_layout()
plt.show()

```



Interpretação

A distribuição do IMC apresenta assimetria à direita, com uma cauda longa em direção aos valores mais elevados. Observa-se que a **média é maior que a mediana**, o que é característico de distribuições assimétricas à direita. Nesses casos, a **mediana representa melhor o valor típico dos dados**, pois sofre menor influência de valores extremos (outliers).

3. Aplicação de Min-Max e Z-score

Neste item, foi aplicada a normalização **Min-Max** e a padronização **Z-score** sobre a variável **IMC**. Em seguida, foram plotados os histogramas das distribuições transformadas para avaliar o comportamento dos dados após as transformações de escala.

```

In [12]: # --- Min-Max Scaling ---
imc_array = imc.values.reshape(-1, 1)
scaler = MinMaxScaler()
imc_minmax = scaler.fit_transform(imc_array).flatten()

# --- Z-score ---
imc_zscore = (imc - imc.mean()) / imc.std()

```

```
imc_minmax[:5], imc_zscore[:5]
```

```
Out[12]: array([0.25189394, 0.30871212, 0.27083333, 0.52840909, 0.10984848]),
0    -0.222973
1     0.199805
2    -0.082047
3     1.834547
4    -1.279918
Name: IMC, dtype: float64)
```

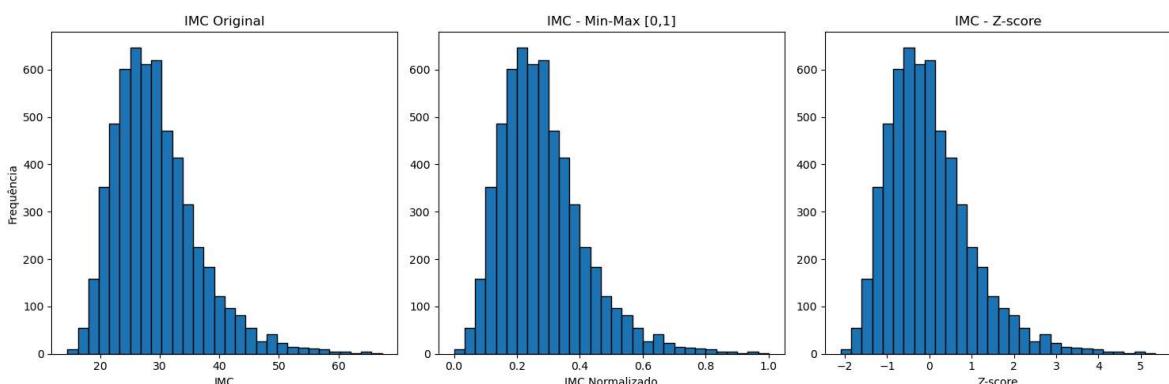
```
In [13]: plt.figure(figsize=(15, 5))
```

```
# IMC Original
plt.subplot(1, 3, 1)
plt.hist(imc, bins=30, edgecolor="black")
plt.title("IMC Original")
plt.xlabel("IMC")
plt.ylabel("Frequência")

# Min-Max
plt.subplot(1, 3, 2)
plt.hist(imc_minmax, bins=30, edgecolor="black")
plt.title("IMC - Min-Max [0,1]")
plt.xlabel("IMC Normalizado")

# Z-score
plt.subplot(1, 3, 3)
plt.hist(imc_zscore, bins=30, edgecolor="black")
plt.title("IMC - Z-score")
plt.xlabel("Z-score")

plt.tight_layout()
plt.show()
```



4. Distribuição Binomial no NHANES

Neste item, foi utilizada a variável binária **JaFumou100Cigarros**, que indica se o indivíduo já fumou pelo menos 100 cigarros na vida. A partir dessa variável, foi calculada a probabilidade empírica \hat{p} . Em seguida, foram geradas e comparadas graficamente as distribuições **Binomial($n = 100$, $p = \hat{p}$)** e **Poisson($\lambda = n \cdot \hat{p}$)**.

```
In [14]: # Selecionando a variável binária
fumou = df["JaFumou100Cigarros"].dropna()
```

```
# Mantendo apenas valores válidos: 1 = Sim, 2 = Não
fumou = fumou[fumou.isin([1, 2])]

# Convertendo para binário: 1 = Sim, 0 = Não
fumou_bin = (fumou == 1).astype(int)

# Probabilidade empírica p-hat
p_hat = fumou_bin.mean()
p_hat
```

Out[14]: np.float64(0.4050655021834061)

In [15]:

```
from scipy.stats import binom, poisson
```

```
n = 100
lam = n * p_hat

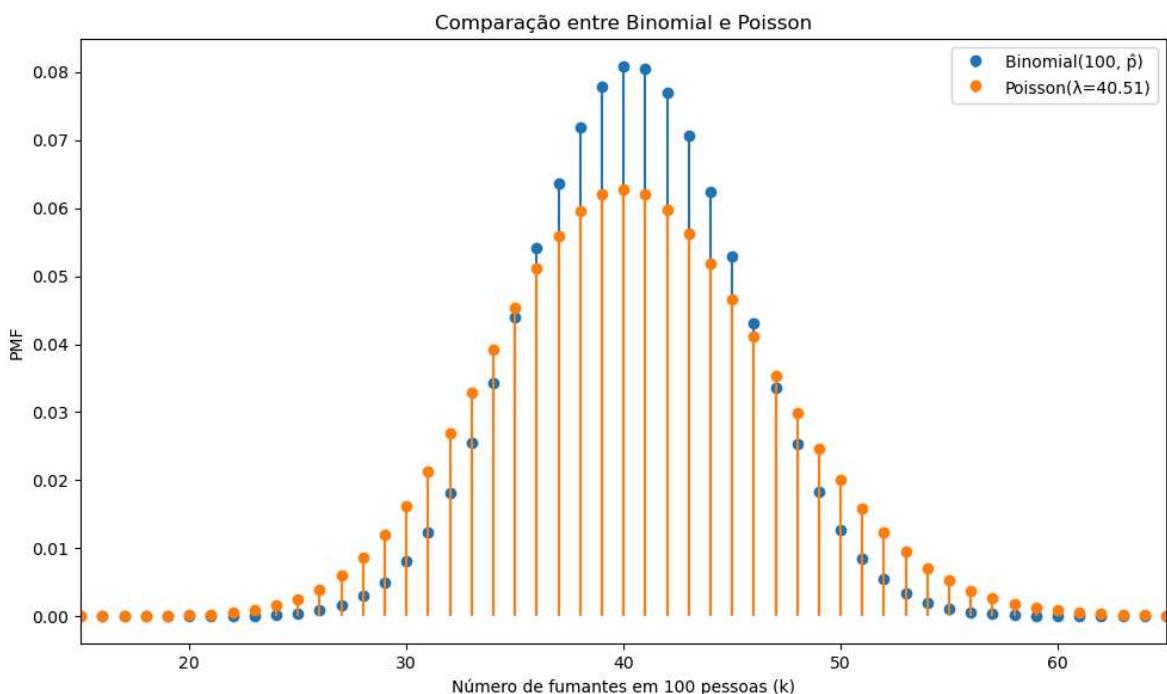
k = np.arange(0, n + 1)

pmf_binom = binom.pmf(k, n, p_hat)
pmf_pois = poisson.pmf(k, lam)

plt.figure(figsize=(10, 6))

plt.stem(k, pmf_binom, basefmt=" ", label="Binomial(100, p-hat)")
plt.stem(k, pmf_pois, basefmt=" ", markerfmt="C1o", linefmt="C1-", label="Poisson(λ=40.51)")

plt.xlim(max(0, int(lam - 4*np.sqrt(lam))), int(lam + 4*np.sqrt(lam)))
plt.xlabel("Número de fumantes em 100 pessoas (k)")
plt.ylabel("PMF")
plt.title("Comparação entre Binomial e Poisson")
plt.legend()
plt.tight_layout()
plt.show()
```



Interpretação

A probabilidade empírica \hat{p} representa a proporção de indivíduos que já fumaram pelo menos 100 cigarros na amostra. A distribuição Binomial modela o número esperado de fumantes em 100 pessoas. A distribuição de Poisson, com $\lambda = n \cdot \hat{p}$, fornece uma boa aproximação da Binomial para esse cenário, o que pode ser visualizado pela forte semelhança entre as duas curvas.

5. Distribuição Normal vs. Log-Normal

Neste item, foi selecionada a variável contínua positiva **CircunferenciaCintura**, que apresenta cauda longa à direita. Foram construídos dois histogramas: um com os valores originais e outro após a aplicação da transformação logarítmica. O objetivo é avaliar como a transformação aproxima a distribuição de uma normal.

```
In [16]: # Selecionando a variável
cintura = df["CircunferenciaCintura"].dropna()

# Medidas de tendência central
media_cintura = cintura.mean()
mediana_cintura = cintura.median()
moda_cintura = cintura.mode().iloc[0]

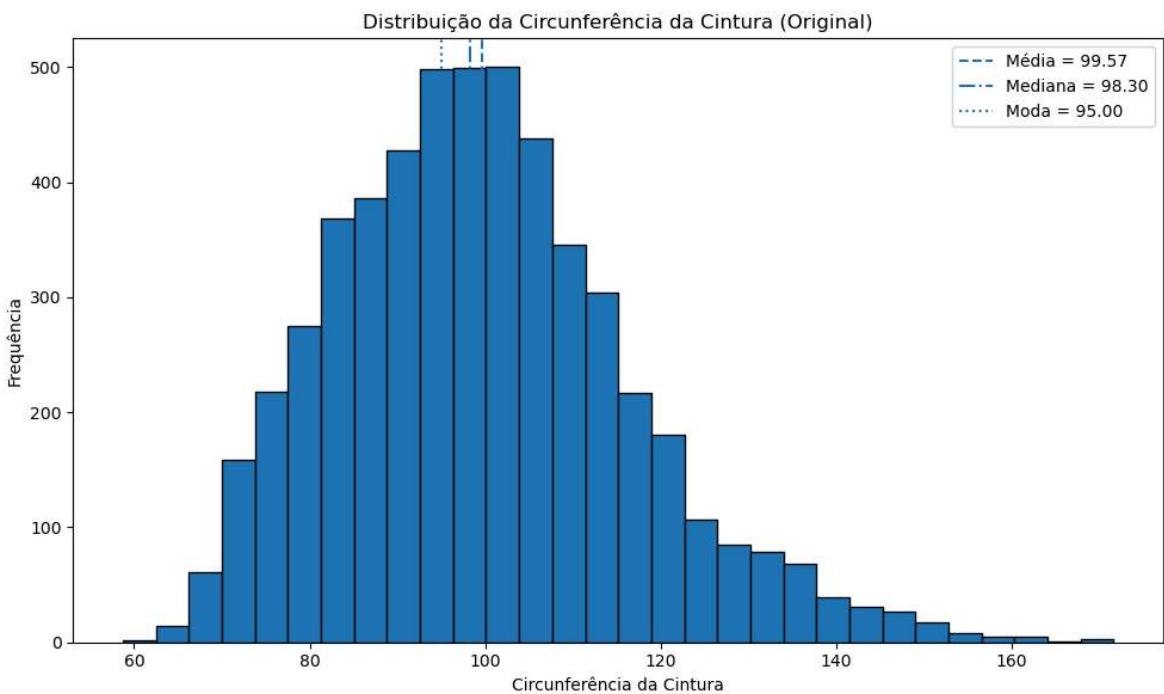
media_cintura, mediana_cintura, moda_cintura
```

```
Out[16]: (np.float64(99.56721311475411), 98.3, np.float64(95.0))
```

```
In [17]: plt.figure(figsize=(10, 6))
plt.hist(cintura, bins=30, edgecolor="black")

plt.axvline(media_cintura, linestyle="--", label=f"Média = {media_cintura:.2f}")
plt.axvline(mediana_cintura, linestyle="-.", label=f"Mediana = {mediana_cintura:.2f}")
plt.axvline(modal_cintura, linestyle=":", label=f"Moda = {moda_cintura:.2f}")

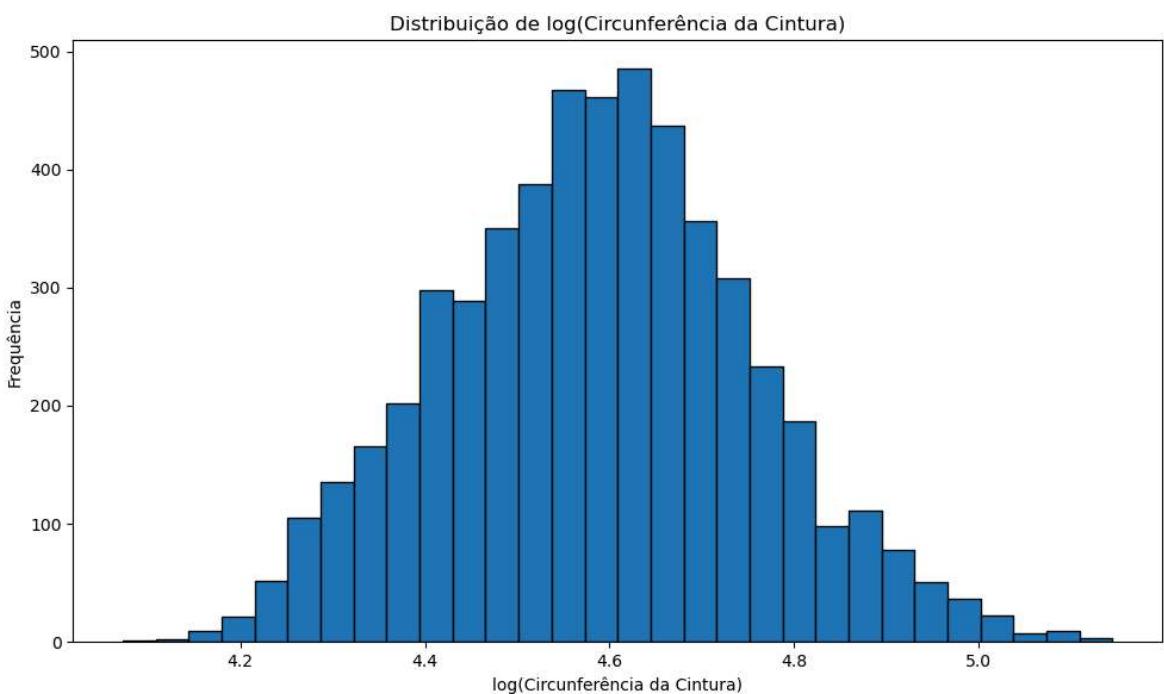
plt.title("Distribuição da Circunferência da Cintura (Original)")
plt.xlabel("Circunferência da Cintura")
plt.ylabel("Frequência")
plt.legend()
plt.tight_layout()
plt.show()
```



```
In [18]: # Removendo valores zero ou negativos (log não existe para <= 0)
cintura_pos = cintura[cintura > 0]

log_cintura = np.log(cintura_pos)

plt.figure(figsize=(10, 6))
plt.hist(log_cintura, bins=30, edgecolor="black")
plt.title("Distribuição de log(Circunferência da Cintura)")
plt.xlabel("log(Circunferência da Cintura)")
plt.ylabel("Frequência")
plt.tight_layout()
plt.show()
```



Interpretação

A distribuição original da circunferência da cintura apresenta forte assimetria à direita, com cauda longa para valores elevados. Após a aplicação da transformação logarítmica, observa-se que a distribuição se torna mais simétrica e próxima de uma normal. Isso confirma que variáveis positivas com cauda longa podem ser bem modeladas por uma distribuição log-normal, conforme discutido nos slides da disciplina.