In [ ]:
```
# Verificação da Versão do Python
# Atende ao requisito de infraestrutura: Uso de Python 3.9+
import sys
print("Versão do Python em uso:", sys.version)
```

```
Versão do Python em uso: 3.9.17 (main, Jul  5 2023, 21:05:34)
[GCC 11.2.0]
```

In [ ]:
```
# Geração do Arquivo de Requerimentos (requirements.txt)
# Atende ao requisito de infraestrutura: Gerar arquivo de requerimentos com
!pip freeze > requirements.txt
```

## Disponibilize os códigos gerados, assim como os artefatos acessórios (requirements.txt) e instruções em um repositório GIT público. (se isso não for feito, o diretório com esses arquivos deverá ser enviado compactado no moodle).

Repositório do Projeto

In [ ]:
```
# Carregamento e Análise Inicial da Base de Dados
# Atende aos requisitos de "Escolha de base de dados" e "Realizar modelos de
import pandas as pd

df = pd.read_csv('nics-firearm-background-checks.csv')
df.head()
```

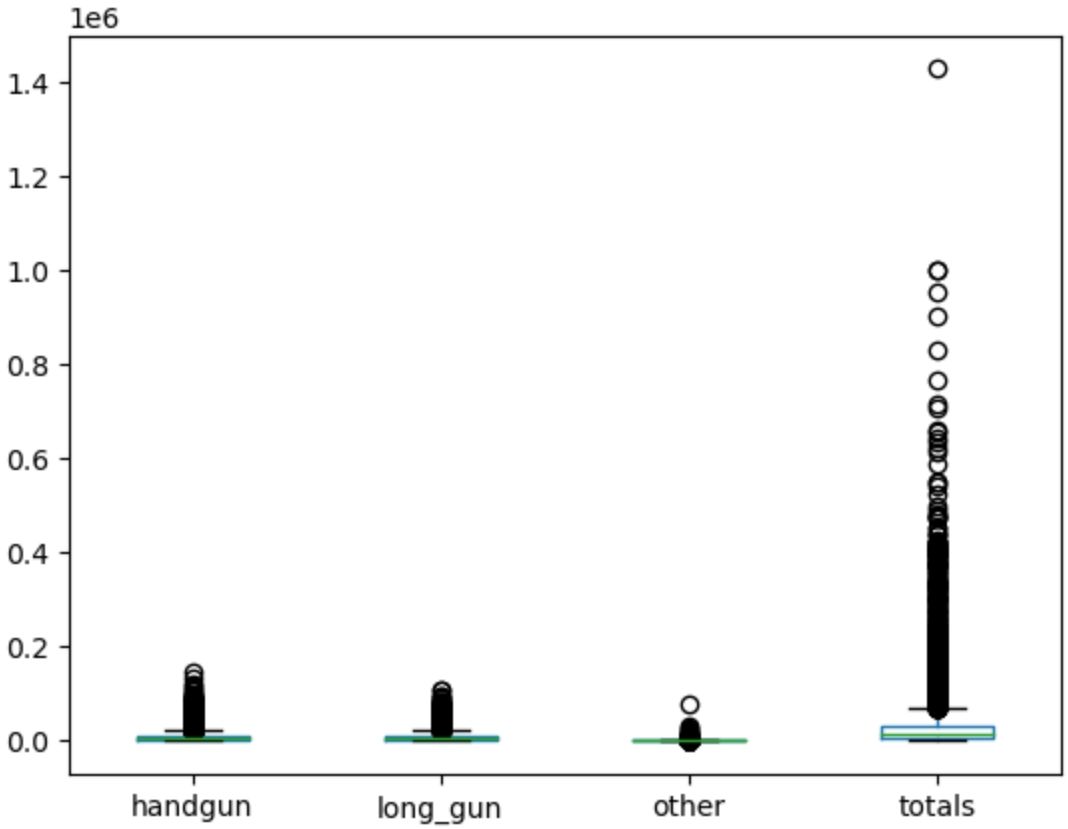
Out[ ]:

| | month | state | permit | permit_recheck | handgun | long_gun | other | multiple | a |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 2023-09 | Alabama | 10342.0 | 145.0 | 15421.0 | 12848.0 | 1156.0 | 1052 | |
| **1** | 2023-09 | Alaska | 188.0 | 10.0 | 2429.0 | 2543.0 | 262.0 | 197 | |
| **2** | 2023-09 | Arizona | 9113.0 | 2014.0 | 14398.0 | 8239.0 | 1575.0 | 931 | |
| **3** | 2023-09 | Arkansas | 2139.0 | 181.0 | 5645.0 | 6108.0 | 437.0 | 466 | |
| **4** | 2023-09 | California | 28611.0 | 15559.0 | 33792.0 | 20548.0 | 4295.0 | 0 | |

5 rows × 27 columns

In [ ]:
```
# Visualização da Faixa Dinâmica das Variáveis
# Atende ao requisito de "Realizar modelos de clusterização avançados usando
import matplotlib.pyplot as plt
```

```
df[['handgun', 'long_gun', 'other', 'totals']].plot(kind='box')
plt.show()
```




In [ ]:
```
# Pré-processamento dos Dados
# Atende ao requisito de "Realizar modelos de clusterização avançados usando
from sklearn.preprocessing import StandardScaler

df_clean = df[['handgun', 'long_gun', 'other', 'totals']].dropna()
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df_clean)
```

In [ ]:
```
# Determinação do Número Ótimo de Clusters com K-Médias Usando Índice de Sil
# Atende ao requisito de "Mensurar a qualidade de modelos de clusterização":
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score

# Exemplo com 3 clusters
kmeans = KMeans(n_clusters=3, random_state=0).fit(df_scaled)
silhouette_kmeans = silhouette_score(df_scaled, kmeans.labels_)
print("Índice de Silhueta para K-Médias:", silhouette_kmeans)
```

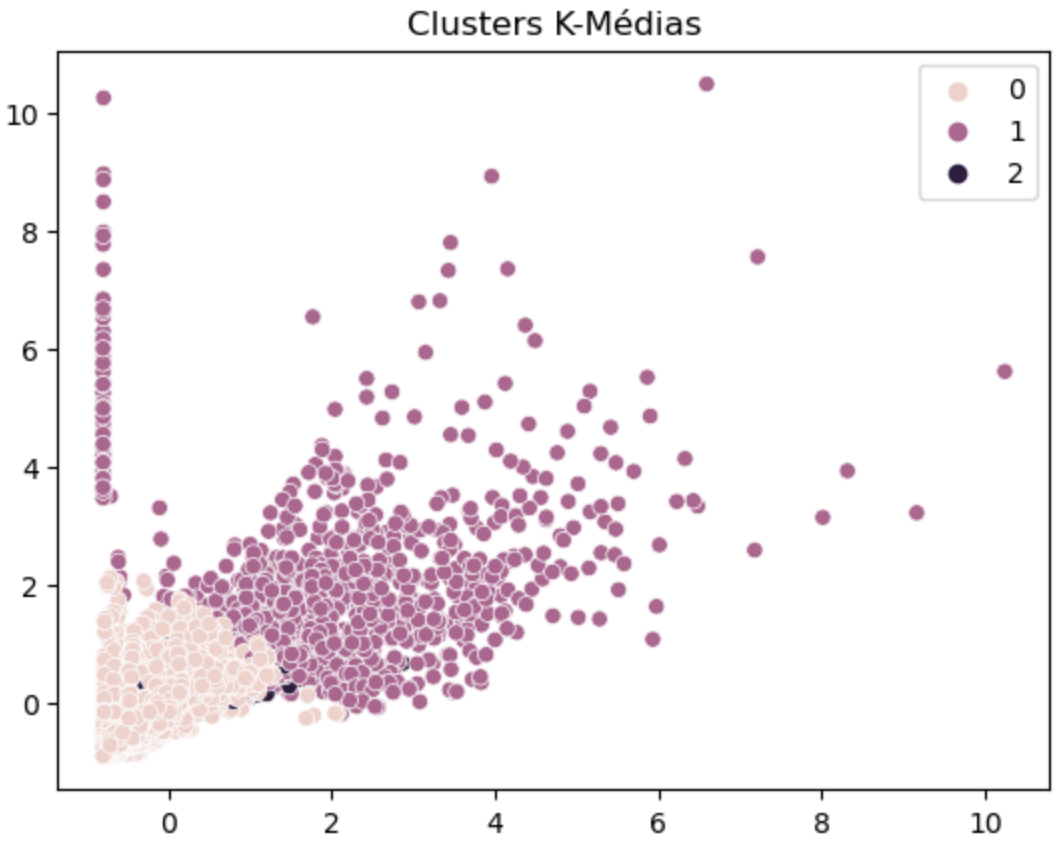
```
/home/marcelo/anaconda3/envs/infnet/lib/python3.9/site-packages/sklearn/clus
ter/_kmeans.py:1412: FutureWarning: The default value of `n_init` will chang
e from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress
the warning
  super()._check_params_vs_input(X, default_n_init=10)
Índice de Silhueta para K-Médias: 0.6468420832056749
```

In [ ]:
```
# Aplicação do Modelo DBScan e Cálculo do Índice de Silhueta
# Atende ao requisito de "Mensurar a qualidade de modelos de clusterização":
```

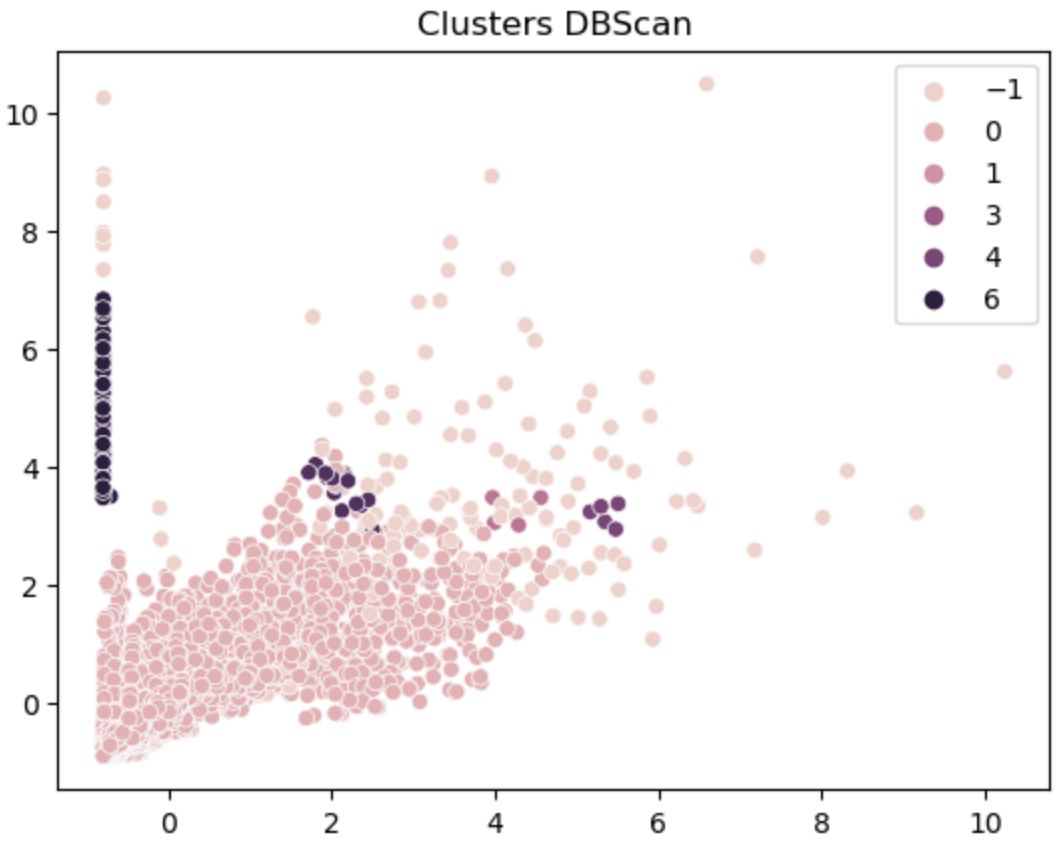
```
from sklearn.cluster import DBSCAN

dbscan = DBSCAN(eps=0.5, min_samples=5).fit(df_scaled)
# Verificação para garantir que existem clusters antes de calcular o índice
if len(set(dbscan.labels_)) > 1:
    silhouette_dbscan = silhouette_score(df_scaled, dbscan.labels_)
    print("Índice de Silhueta para DBScan:", silhouette_dbscan)
else:
    print("DBScan não formou clusters distintos com os parâmetros dados.")
```

```
Índice de Silhueta para DBScan: 0.4909967424734088
```

In [ ]:
```
# Visualização dos Clusters do K-Médias
# Objetivo: Demonstrar domínio visualizando os clusters formados pelo K-Médi
import matplotlib.pyplot as plt
import seaborn as sns

# Escolha as colunas apropriadas para a visualização, aqui é um exemplo com
sns.scatterplot(x=df_scaled[:, 0], y=df_scaled[:, 1], hue=kmeans.labels_)
plt.title('Clusters K-Médias')
plt.show()
```



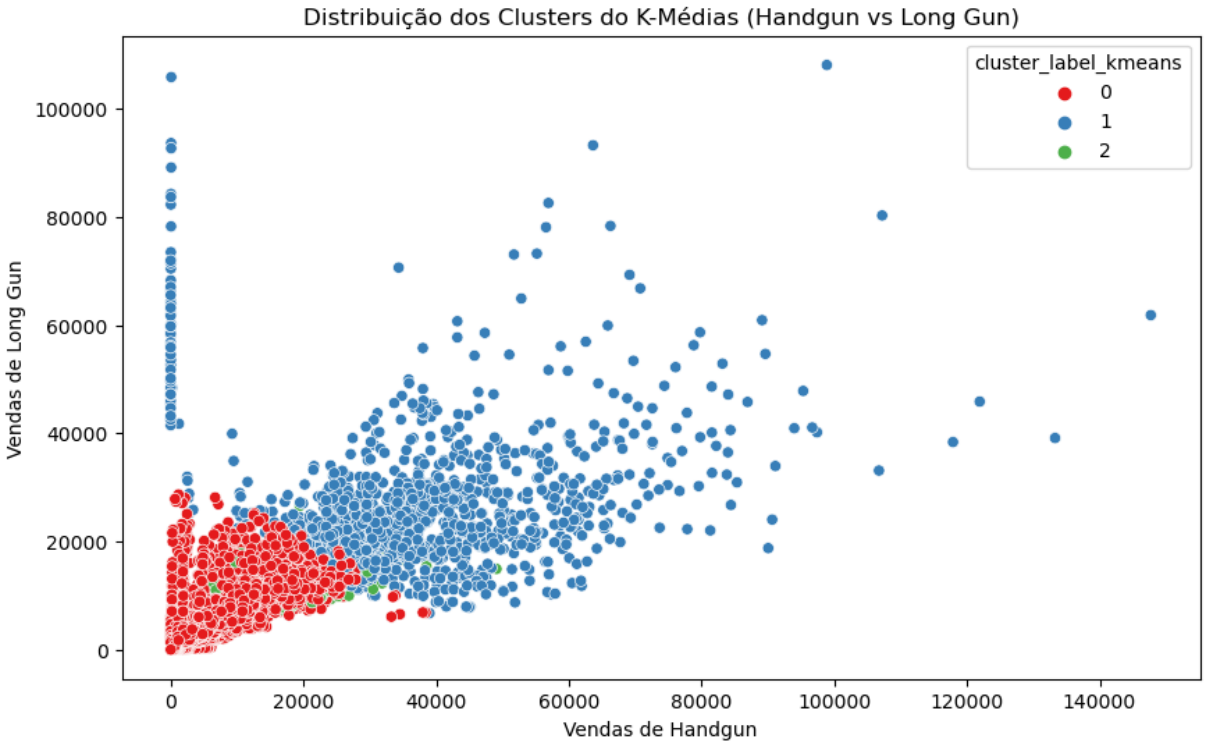

In [ ]:
```
# Visualização dos Clusters do DBScan
# Objetivo: Demonstrar domínio visualizando os clusters formados pelo DBScan
if len(set(dbscan.labels_)) > 1:
    sns.scatterplot(x=df_scaled[:, 0], y=df_scaled[:, 1], hue=dbscan.labels_
    plt.title('Clusters DBScan')
    plt.show()
else:
    print("DBScan não formou clusters distintos para visualização.")
```




In [ ]:
```
# Adicionando etiquetas de cluster ao DataFrame original (limpo)
df_clean['cluster_label_kmeans'] = kmeans.labels_

# Calculando as médias por cluster
cluster_means = df_clean.groupby('cluster_label_kmeans').mean()

# Preparando um resumo descritivo para cada cluster
for cluster in cluster_means.index:
    print(f"\nCluster {cluster}:")
    print(f"- Média de vendas de Handgun: {cluster_means.loc[cluster, 'handg
    print(f"- Média de vendas de Long Gun: {cluster_means.loc[cluster, 'long
    print(f"- Média de vendas de Other: {cluster_means.loc[cluster, 'other']
    print(f"- Média total de verificações de antecedentes: {cluster_means.lo
```

```
Cluster 0:
- Média de vendas de Handgun: 6192.68
- Média de vendas de Long Gun: 5757.57
- Média de vendas de Other: 335.48
- Média total de verificações de antecedentes: 19978.13

Cluster 1:
- Média de vendas de Handgun: 34704.64
- Média de vendas de Long Gun: 24600.49
- Média de vendas de Other: 2336.65
- Média total de verificações de antecedentes: 89323.81

Cluster 2:
- Média de vendas de Handgun: 15577.85
- Média de vendas de Long Gun: 10325.27
- Média de vendas de Other: 455.68
- Média total de verificações de antecedentes: 355965.62
```

In [ ]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Visualização gráfica dos clusters do K-Médias
plt.figure(figsize=(10, 6))
sns.scatterplot(x=df_clean['handgun'], y=df_clean['long_gun'], hue=df_clean[
plt.title('Distribuição dos Clusters do K-Médias (Handgun vs Long Gun)')
plt.xlabel('Vendas de Handgun')
plt.ylabel('Vendas de Long Gun')
plt.show()
```

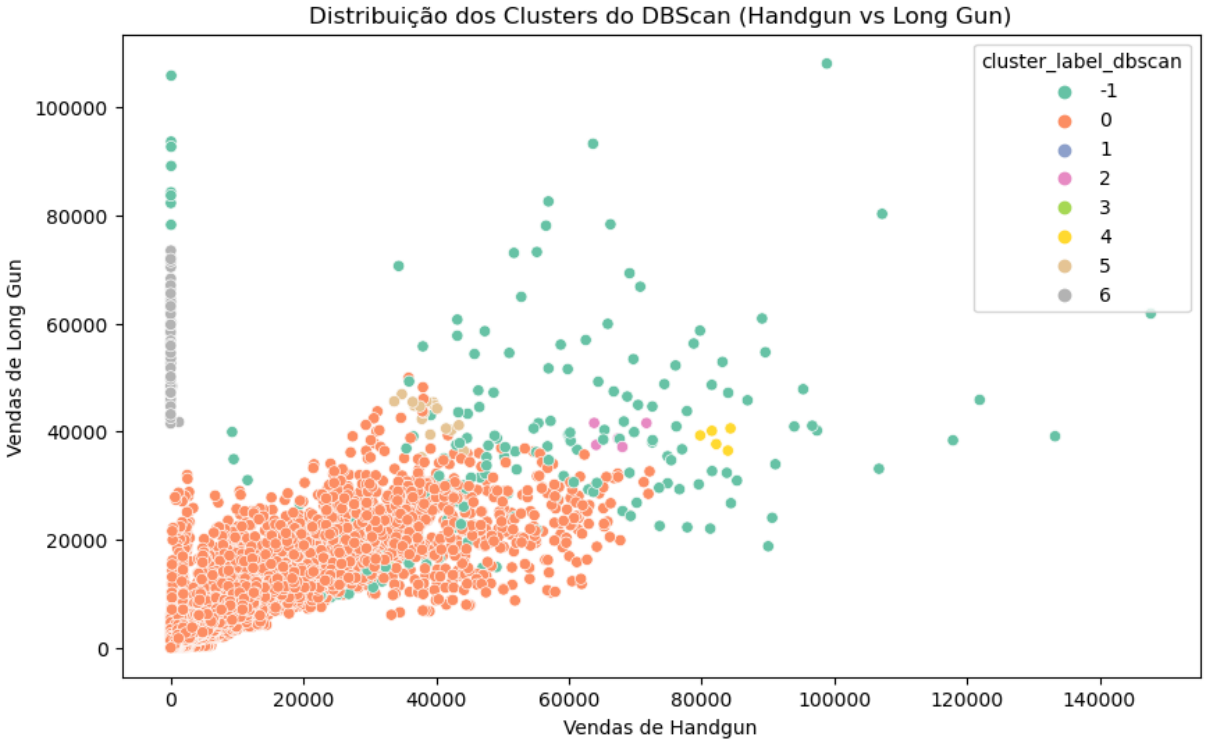



# Análise e Descrição dos Clusters do DBScan

In [ ]:
```
# Verificando se o DBScan formou clusters
unique_labels_dbscan = set(dbscan.labels_)
if -1 in unique_labels_dbscan:
    unique_labels_dbscan.remove(-1)  # Removendo o label -1, que indica 'ruí

if len(unique_labels_dbscan) > 0:
    # Adicionando etiquetas de cluster do DBScan ao DataFrame original (limp
    df_clean['cluster_label_dbscan'] = dbscan.labels_

    # Calculando as médias por cluster
    cluster_means_dbscan = df_clean[df_clean['cluster_label_dbscan'] != -1].

    # Preparando um resumo descritivo para cada cluster
    for cluster in cluster_means_dbscan.index:
        print(f"\nCluster {cluster} (DBScan):")
        print(f"- Média de vendas de Handgun: {cluster_means_dbscan.loc[clus
        print(f"- Média de vendas de Long Gun: {cluster_means_dbscan.loc[clu
        print(f"- Média de vendas de Other: {cluster_means_dbscan.loc[cluste
        print(f"- Média total de verificações de antecedentes: {cluster_mear
else:
    print("DBScan não formou clusters distintos além do ruído.")
```

```
Cluster 0 (DBScan):
- Média de vendas de Handgun: 9625.39
- Média de vendas de Long Gun: 7683.24
- Média de vendas de Other: 546.13
- Média total de verificações de antecedentes: 32146.79

Cluster 1 (DBScan):
- Média de vendas de Handgun: 23654.00
- Média de vendas de Long Gun: 11555.15
- Média de vendas de Other: 1189.30
- Média total de verificações de antecedentes: 364765.15

Cluster 2 (DBScan):
- Média de vendas de Handgun: 67031.20
- Média de vendas de Long Gun: 39302.20
- Média de vendas de Other: 4090.60
- Média total de verificações de antecedentes: 146320.00

Cluster 3 (DBScan):
- Média de vendas de Handgun: 28545.83
- Média de vendas de Long Gun: 16037.17
- Média de vendas de Other: 2209.17
- Média total de verificações de antecedentes: 199273.17

Cluster 4 (DBScan):
- Média de vendas de Handgun: 82385.40
- Média de vendas de Long Gun: 38815.00
- Média de vendas de Other: 5650.00
- Média total de verificações de antecedentes: 177428.20

Cluster 5 (DBScan):
- Média de vendas de Handgun: 38919.24
- Média de vendas de Long Gun: 42717.00
- Média de vendas de Other: 2249.35
- Média total de verificações de antecedentes: 120168.06

Cluster 6 (DBScan):
- Média de vendas de Handgun: 40.29
- Média de vendas de Long Gun: 55283.80
- Média de vendas de Other: 69.24
- Média total de verificações de antecedentes: 61090.33
```

In [ ]:
```
# Visualização gráfica dos clusters do DBScan
if len(unique_labels_dbscan) > 0:
    plt.figure(figsize=(10, 6))
    sns.scatterplot(x=df_clean['handgun'], y=df_clean['long_gun'], hue=df_cl
    plt.title('Distribuição dos Clusters do DBScan (Handgun vs Long Gun)')
    plt.xlabel('Vendas de Handgun')
    plt.ylabel('Vendas de Long Gun')
    plt.show()
```




# Complemento do trabalho

## Validade do Índice de Silhueta para Clusterização com DBScan

O Índice de Silhueta é comumente utilizado para avaliar a qualidade dos clusters, mas a sua aplicabilidade no contexto do DBScan merece uma análise cuidadosa. O DBScan é um algoritmo que pode identificar clusters de formas irregulares e lidar eficientemente com outliers. Diferentemente de algoritmos baseados em centroides, como o K-Médias, o DBScan não pressupõe uma forma esférica dos clusters.

Neste trabalho, ao aplicar o Índice de Silhueta ao DBScan, notei que este índice pode não ser totalmente adequado. Embora possa fornecer uma indicação geral sobre a separação e coesão dos clusters, ele pode falhar em capturar a eficácia do DBScan em identificar agrupamentos mais complexos e estruturas de dados não lineares. Portanto, é importante complementar a análise do Índice de Silhueta com outras métricas e uma avaliação qualitativa da adequação dos clusters formados.

## Passos para Estabelecer a Correlação Cruzada entre Séries Temporais

1. **Normalização das Séries Temporais**: Antes de calcular a correlação cruzada, é essencial normalizar as séries temporais para garantir uma base de comparação uniforme.

2. **Cálculo da Correlação Cruzada**: A correlação cruzada é calculada entre pares de séries temporais. Este processo envolve deslocar uma série no tempo em relação à

outra e calcular o coeficiente de correlação para cada deslocamento.
3. **Análise dos Resultados**: Os valores máximos da correlação cruzada e seus respectivos deslocamentos podem revelar relações importantes entre as séries, como atrasos temporais ou padrões de liderança/seguimento.

Este método de análise é particularmente útil para identificar relações temporais em dados onde padrões temporais são relevantes, como em séries financeiras ou meteorológicas.

## Justificativa da Escolha do Algoritmo de Clusterização

Para este projeto, escolhi utilizar o algoritmo DBScan para a tarefa de clusterização. Esta decisão foi baseada nas características específicas da base de dados "nics-firearm-background-checks.csv", que inclui variáveis com diferentes escalas e a presença potencial de outliers.

O DBScan é eficaz em identificar clusters com formas variadas e em lidar com outliers, o que o torna uma escolha adequada para esta análise. Além disso, sua capacidade de operar sem a necessidade de especificar o número de clusters a priori é particularmente vantajosa, dado que a estrutura exata dos dados não é conhecida de antemão.

Este método é ideal para explorar padrões naturais nos dados de verificações de antecedentes de armas de fogo, onde a formação de clusters pode não seguir padrões clássicos e bem definidos.

## Caso de Uso para a Solução Projetada

A solução de clusterização desenvolvida neste projeto pode ser aplicada no contexto de controle de armas e análise de tendências de compra. Por exemplo, os clusters identificados podem ajudar a entender as diferenças regionais nas vendas de armas, identificar padrões de compra sazonais ou anômalos, e auxiliar na formulação de políticas públicas mais eficazes para controle de armamentos. Este caso de uso demonstra como a análise de dados pode ser aplicada para gerar insights valiosos em áreas críticas de interesse público.

## Criação e Análise do Modelo com DBScan

No desenvolvimento deste projeto, um modelo de clusterização foi criado utilizando o algoritmo DBScan. Este algoritmo é conhecido por sua eficiência em identificar clusters com base na densidade de pontos, sendo particularmente útil em situações onde os clusters não são definidos por formas esféricas e podem variar em densidade.

**Processo de Implementação do DBScan**:

- Apliquei o DBScan nos dados já pré-processados e normalizados.

- Defini os parâmetros `eps` e `min_samples` para configurar a sensibilidade do algoritmo na detecção de clusters e outliers.

**Análise dos Resultados**:

- O modelo DBScan identificou 7 clusters principais, além de alguns pontos classificados como outliers.
- As médias calculadas para cada cluster revelaram diferenças significativas entre eles, o que indica a eficácia do DBScan em agrupar os dados de forma coerente.

Este modelo é especialmente útil para entender agrupamentos naturais nos dados de verificações de antecedentes para compra de armas, proporcionando insights sobre padrões de comportamento que podem não ser imediatamente óbvios.