Projeto de K Means Clustering - Projeto

September 13, 2025

1 Projeto de K Means Clustering

Para este projeto, tentaremos usar o KMeans Clustering para agrupar Universidades em dois grupos: Privadas e Públicas.

É muito importante observar, nós realmente temos os rótulos para este conjunto de dados, mas NÃO os usaremos para o algoritmo de agrupamento KMeans, pois esse é um algoritmo de aprendizado não supervisionado.

Ao usar o algoritmo Kmeans em situações reais, você não possuirá rótulos. Nesse caso, usaremos os rótulos para tentar ter uma idéia do quão bem o algoritmo foi executado, apenas. _____

1.1 Os dados

Usaremos um quadro de dados com 777 observações sobre as 18 variáveis a seguir. * Private: Um fator com níveis Não e Sim, indicando universidade privada ou pública. * Apps: Número de inscrições recebidas. * Accept: Quantidade de inscrições aceitas. * Enroll: Número de estudantes matriculados. * Top10perc: Percentual de novos estudantes vindo do grupo de 10% melhores do segundo grau. * Top25perc: Percentual de novos estudantes vindo do grupo de 25% melhores do segundo grau. * F.Undergrad: Número de alunos de graduação em tempo integral. * P.Undergrad Número de alunos de graduação em tempo parcial. * Outstate: Aulas fora do estado. * Room.Board: Custos da sala. * Books: Custos de livros estimados. * Personal: Estimativa de gastos por pessoa. * PhD: Percentual de PHD's na universidade. * Terminal: Percentual da faculdade com graduação. * S.F.Ratio: Taxa estudantes/faculdade. * perc.alumni: Percentual dos ex-alunos que doam. * Expend: Despesas da instituição por aluno. * Grad.Rate: Taxa de graduação

1.2 Importar bibliotecas

Importe as bibliotecas que você costuma usar para análise de dados.

[1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import pearsonr
from sklearn.linear_model import LinearRegression
```

1.3 Obtenha os dados

Leia no arquivo College_Data usando read_csv. Descubra como setar a primeira coluna como índice.

[2]:
```
dados = pd.read_csv('C:
↪\\Users\\bruno\\Documents\\MachineLearning\\2025\\atividades\\K-Means\\College_Data',␣
↪index_col=0)
```

Verifique o cabeçalho dos dados

[3]:
```
dados.head()
```

[3]:

	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD	Terminal	S.F.Ratio	perc.alumni	Expend	Grad.Rate
Abilene Christian University	Yes	1660	1232	721	23	52	2885	537	7440	3300	450	2200	70	78	18.1	12	7041	60
Adelphi University	Yes	2186	1924	512	16	29	2683	1227	12280	6450	750	1500	29	30	12.2	16	10527	56
Adrian College	Yes	1428	1097	336	22	50	1036	99	11250	3750	400	1165	53	66	12.9	30	8735	54
Agnes Scott College	Yes	417	349	137	60	89	510	63	12960	5450	450	875	92	97	7.7	37	19016	59
Alaska Pacific University	Yes	193	146	55	16	44	249	869	7560	4120	800	1500	76	72	11.9	2	10922	15

Verifique os métodos info() e describe() do DataFrame.

[4]:
```
dados.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 777 entries, Abilene Christian University to York College of Pennsylvania
Data columns (total 18 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
```

```
 0   Private      777 non-null    object
 1   Apps         777 non-null    int64
 2   Accept       777 non-null    int64
 3   Enroll       777 non-null    int64
 4   Top10perc    777 non-null    int64
 5   Top25perc    777 non-null    int64
 6   F.Undergrad  777 non-null    int64
 7   P.Undergrad  777 non-null    int64
 8   Outstate     777 non-null    int64
 9   Room.Board   777 non-null    int64
 10  Books        777 non-null    int64
 11  Personal     777 non-null    int64
 12  PhD          777 non-null    int64
 13  Terminal     777 non-null    int64
 14  S.F.Ratio    777 non-null    float64
 15  perc.alumni  777 non-null    int64
 16  Expend       777 non-null    int64
 17  Grad.Rate    777 non-null    int64
dtypes: float64(1), int64(16), object(1)
memory usage: 115.3+ KB
```

[5]:
```
dados.describe()
```

[5]:

	Apps	Accept	Enroll	Top10perc	Top25perc
count	777.000000	777.000000	777.000000	777.000000	777.000000
mean	3001.638353	2018.804376	779.972973	27.558559	55.796654
std	3870.201484	2451.113971	929.176190	17.640364	19.804778
min	81.000000	72.000000	35.000000	1.000000	9.000000
25%	776.000000	604.000000	242.000000	15.000000	41.000000
50%	1558.000000	1110.000000	434.000000	23.000000	54.000000
75%	3624.000000	2424.000000	902.000000	35.000000	69.000000
max	48094.000000	26330.000000	6392.000000	96.000000	100.000000

	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books
count	777.000000	777.000000	777.000000	777.000000	777.000000
mean	3699.907336	855.298584	10440.669241	4357.526384	549.380952
std	4850.420531	1522.431887	4023.016484	1096.696416	165.105360
min	139.000000	1.000000	2340.000000	1780.000000	96.000000
25%	992.000000	95.000000	7320.000000	3597.000000	470.000000
50%	1707.000000	353.000000	9990.000000	4200.000000	500.000000
75%	4005.000000	967.000000	12925.000000	5050.000000	600.000000
max	31643.000000	21836.000000	21700.000000	8124.000000	2340.000000

	Personal	PhD	Terminal	S.F.Ratio	perc.alumni
count	777.000000	777.000000	777.000000	777.000000	777.000000
mean	1340.642214	72.660232	79.702703	14.089704	22.743887
std	677.071454	16.328155	14.722359	3.958349	12.391801

```
min     250.000000    8.000000   24.000000    2.500000     0.000000
25%     850.000000   62.000000   71.000000   11.500000    13.000000
50%    1200.000000   75.000000   82.000000   13.600000    21.000000
75%    1700.000000   85.000000   92.000000   16.500000    31.000000
max    6800.000000  103.000000  100.000000   39.800000    64.000000

              Expend  Grad.Rate
count    777.000000  777.00000
mean    9660.171171   65.46332
std     5221.768440   17.17771
min     3186.000000   10.00000
25%     6751.000000   53.00000
50%     8377.000000   65.00000
75%    10830.000000   78.00000
max    56233.000000  118.00000
```

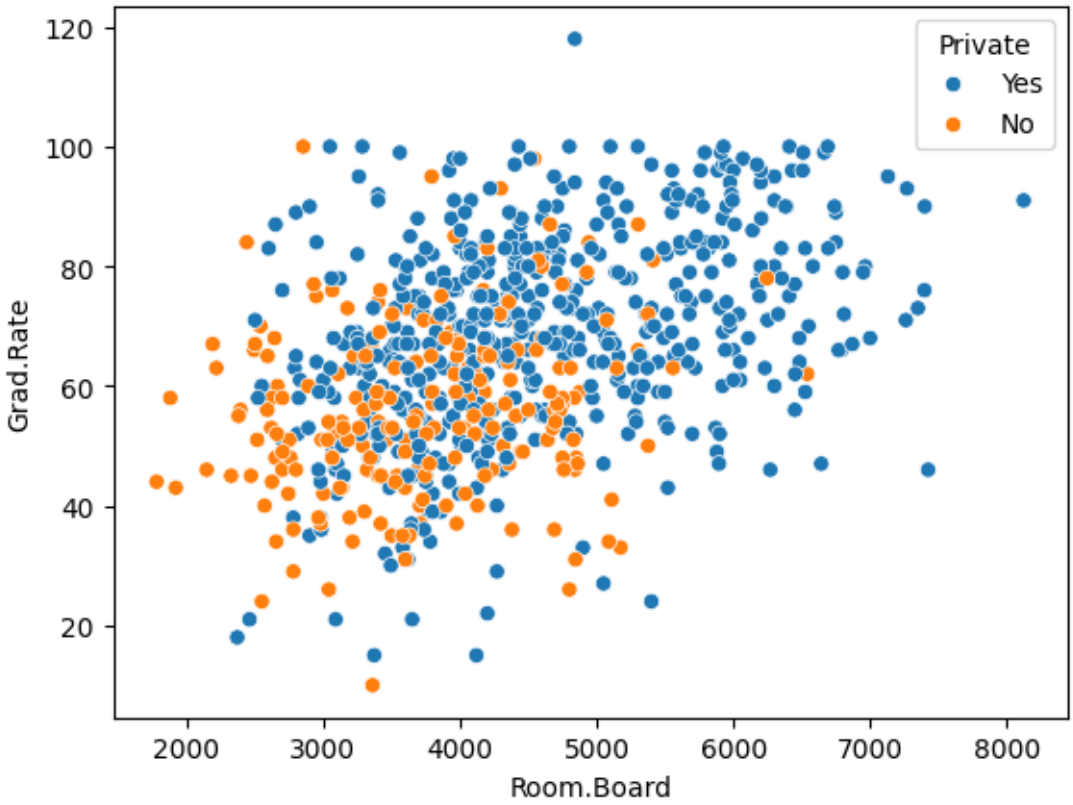
1.4 Análise exploratória de dados

É hora de criar algumas visualizações de dados.

Crie um scatterplot de Grad.Rate versus Room.Board onde os pontos são coloridos pela coluna "Private".

[6]:
```
sns.scatterplot(x='Room.Board', y='Grad.Rate', data=dados, hue='Private')
```

[6]:
```
<Axes: xlabel='Room.Board', ylabel='Grad.Rate'>
```



[7]:
```
publica = dados[dados['Private'] == 'No']
public_media_custo = publica['Room.Board'].mean()
publica_media_grad = publica['Grad.Rate'].mean()

print(f"Faculdades Públicas - Média do custo da sala: {public_media_custo:.2f}")
print(f"Faculdades Públicas - Média da taxa de graduação: {publica_media_grad:.
  ↪2f}%")
```

```
Faculdades Públicas - Média do custo da sala: 3748.24
Faculdades Públicas - Média da taxa de graduação: 56.04%
```

[8]:
```
privada = dados[dados['Private'] == 'Yes']
privada_media_custo = privada['Room.Board'].mean()
privada_media_grad = privada['Grad.Rate'].mean()

print(f"Faculdades Privadas - Média do custo da sala: {privada_media_custo:.
  ↪2f}")
print(f"Faculdades Privadas - Média da taxa de graduação: {privada_media_grad:.
  ↪2f}%")
```

```
Faculdades Privadas - Média do custo da sala: 4586.14
Faculdades Privadas - Média da taxa de graduação: 69.00%
```

[9]:
```
correlacao, p_valor = pearsonr(dados['Room.Board'], dados['Grad.Rate'])
print(f"Correlação de Pearson: {correlacao:.4f}, Valor-p: {p_valor:.4f}")
```

```
Correlação de Pearson: 0.4249, Valor-p: 0.0000
```

[10]:
```
X = dados[['Room.Board']]
y = dados['Grad.Rate']

model = LinearRegression()
model.fit(X, y)

slope = model.coef_[0]
intercept = model.intercept_
print(f"Linear equation: Grad_Rate = {slope:.2f} * Room_Board + {intercept:.
↪2f}")

r_squared = model.score(X, y)
print(f"R-squared: {r_squared:.2f}")
```

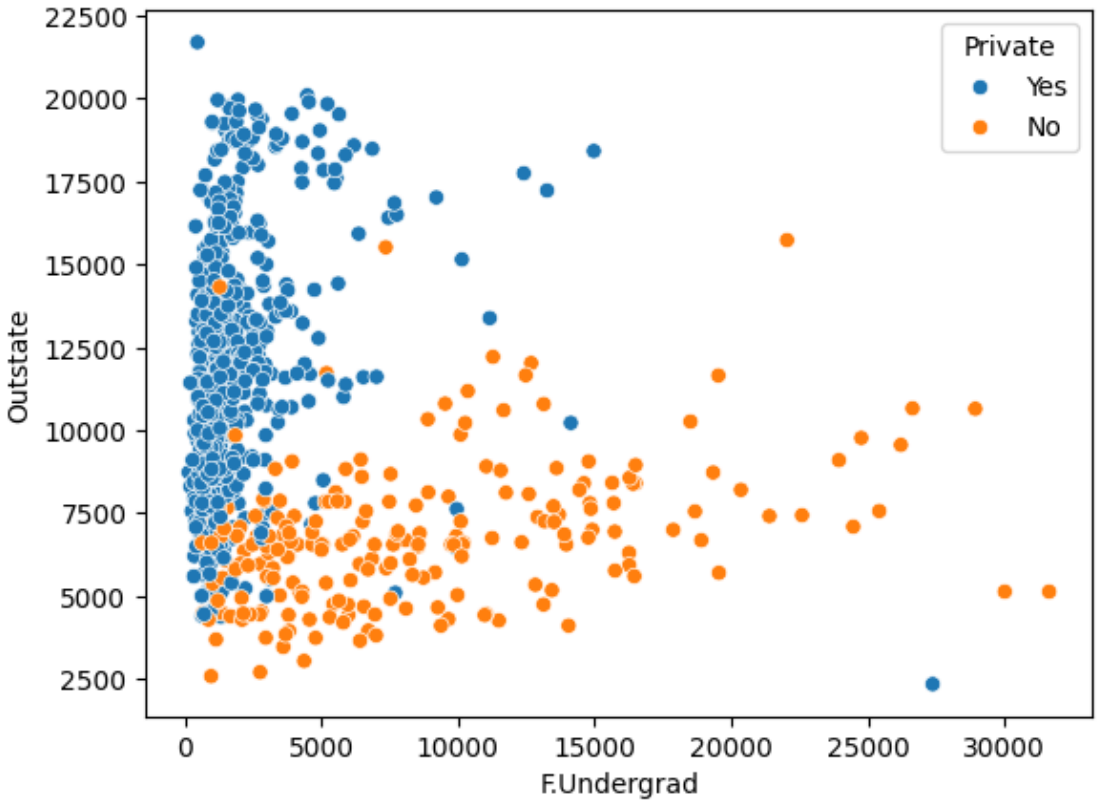
```
Linear equation: Grad_Rate = 0.01 * Room_Board + 36.46
R-squared: 0.18
```

Ou seja, para cada aumento na unidade de custo, a taxa de graduação aumenta em 0.01%. Já quando o custo do quarto é zero, a taxa de graduação 36.46%.

Crie um scatterplot de F.Undergrad versus Outstate onde os pontos são coloridos pela coluna Private.

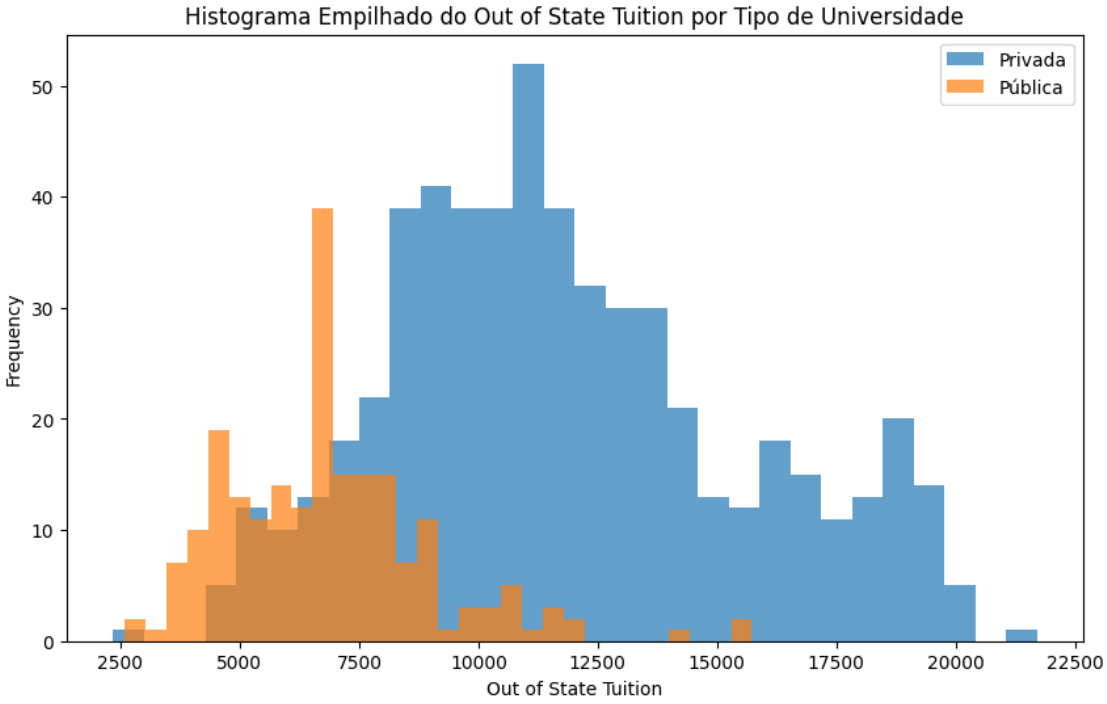
[11]:
```
sns.scatterplot(x='F.Undergrad', y='Outstate', data=dados, hue='Private')
```

[11]:
```
<Axes: xlabel='F.Undergrad', ylabel='Outstate'>
```



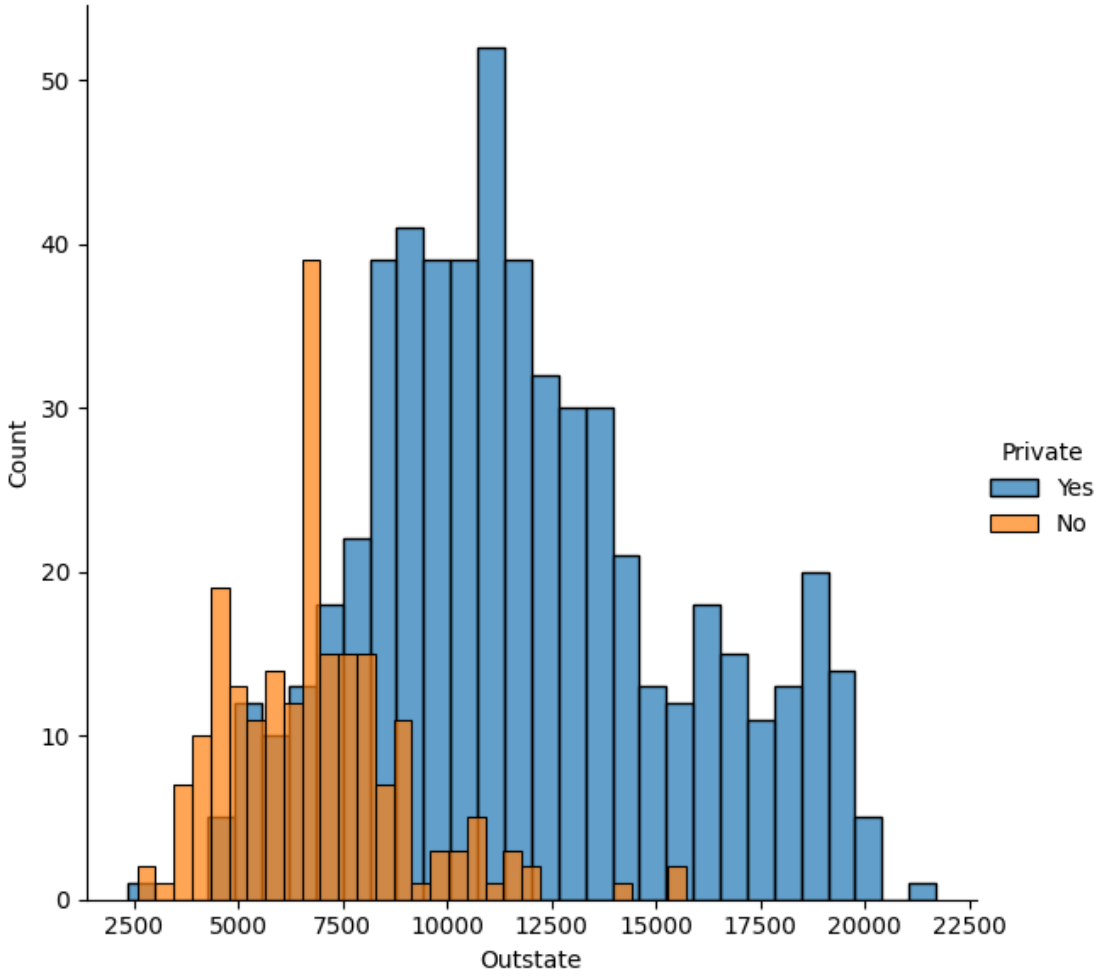
Crie um histograma empilhado que mostra o "Out of State Tuition" com base na coluna Private. Tente fazer isso usando sns.FacetGrid. Se isso for muito complicado, veja se você pode fazê-lo apenas usando duas instâncias de pandas.plot(kind='hist').

[12]:
```
plt.figure(figsize=(10,6))
dados[dados['Private'] == 'Yes']['Outstate'].plot(kind='hist', bins=30, alpha=0.
 ↪7, label='Privada')
dados[dados['Private'] == 'No']['Outstate'].plot(kind='hist', bins=30, alpha=0.
 ↪7, label='Pública')
plt.legend()
plt.xlabel('Out of State Tuition')
plt.title('Histograma Empilhado do Out of State Tuition por Tipo de␣
 ↪Universidade')
plt.show()
```



[13]:
```
sns.FacetGrid(dados, hue='Private', height=6).map(sns.histplot, 'Outstate',␣
↪bins=30, alpha=0.7).add_legend()
```

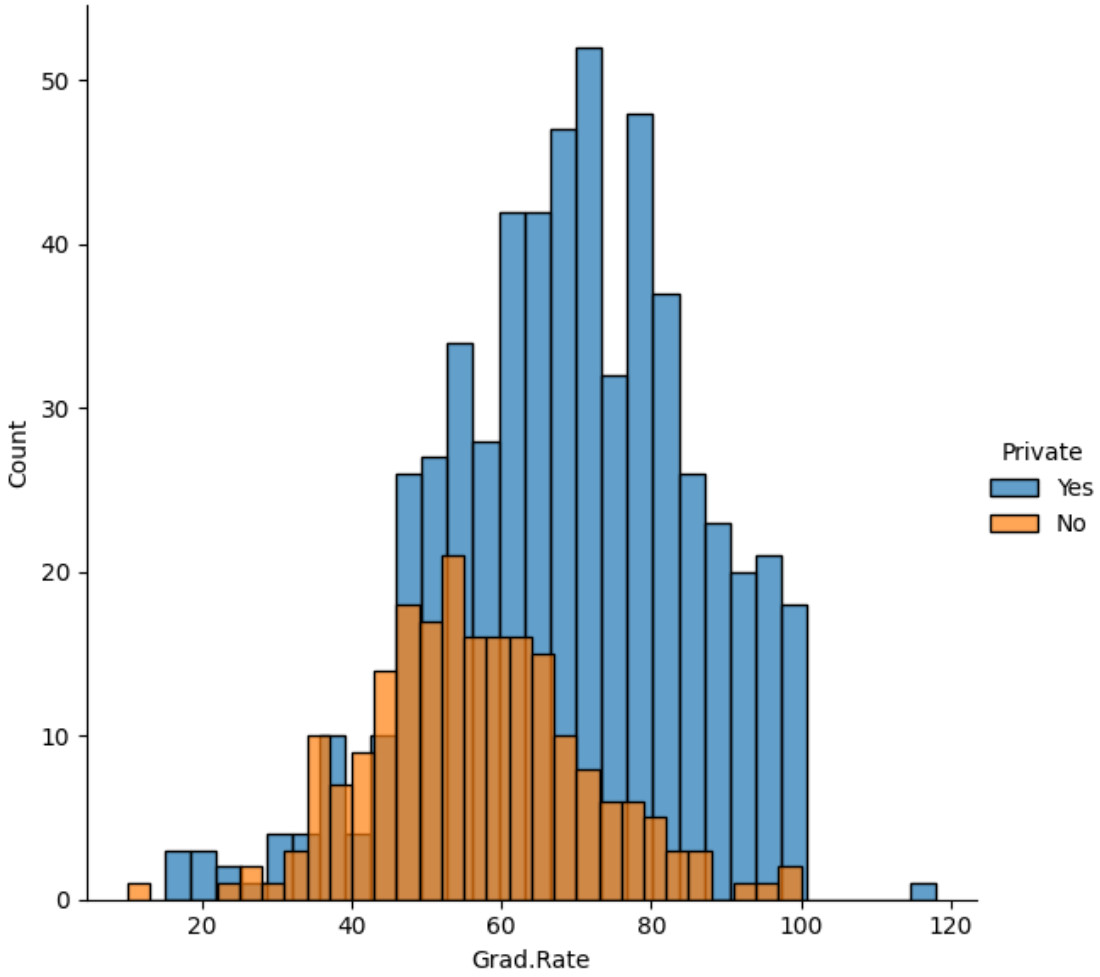
[13]: <seaborn.axisgrid.FacetGrid at 0x239fe1d9550>



Crie um histograma semelhante para a coluna Grad.Rate.

[14]:
```
sns.FacetGrid(dados, hue='Private', height=6).map(sns.histplot, 'Grad.Rate',␣
↪bins=30, alpha=0.7).add_legend()
```

[14]:
```
<seaborn.axisgrid.FacetGrid at 0x239fe329590>
```



Observe que parece haver uma escola particular com uma taxa de graduação superior a 100%. Qual é o nome dessa escola?

[15]:
```
anomalia = dados[(dados['Grad.Rate'] > 100) & (dados['Private'] == 'Yes')]
anomalia.index.tolist()
```

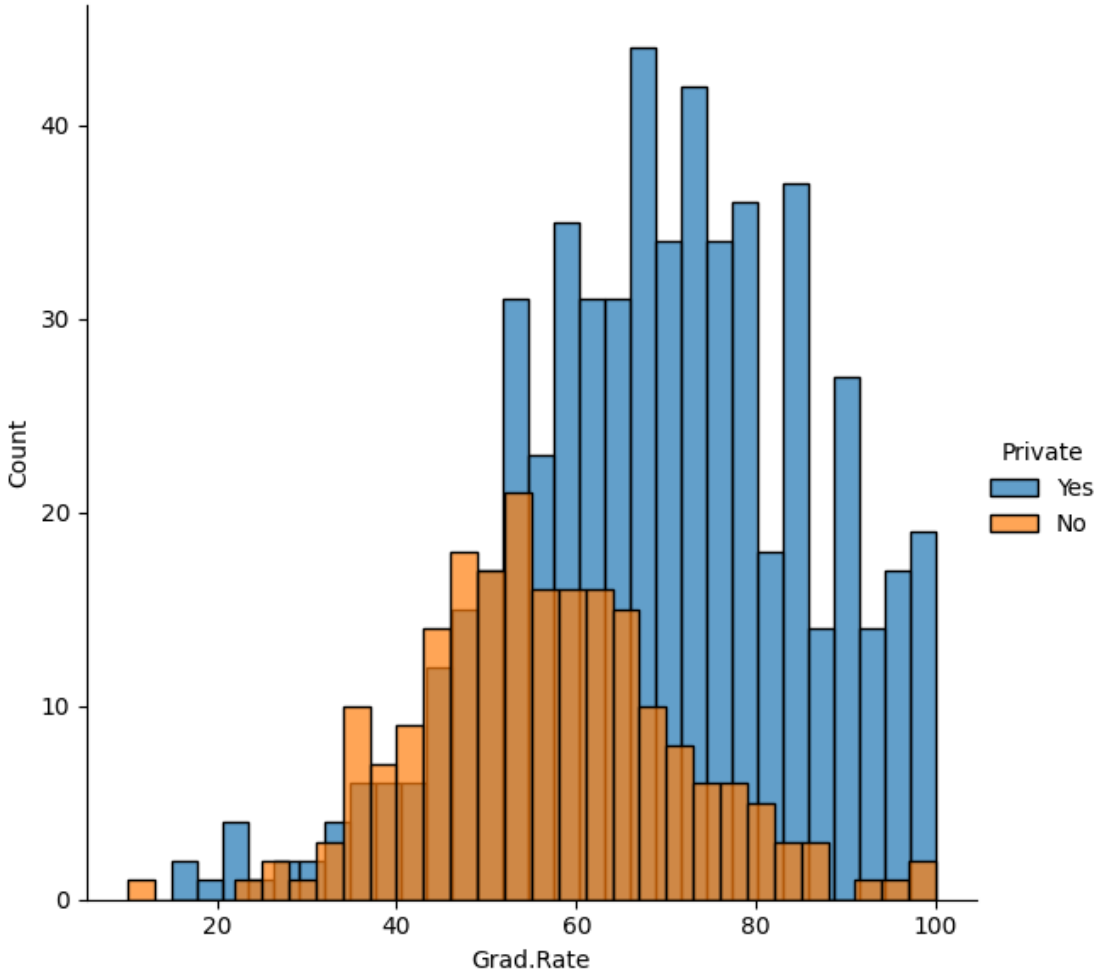
[15]:
```
['Cazenovia College']
```

Defina a taxa de graduação dessa escola para 100 para que isso faça sentido. Você pode obter um aviso (e não um erro) ao fazer esta operação basta usar operações de dataframe ou simplesmente re-fazer a visualização do histograma para garantir que ela realmente foi alterado.

[16]:
```
dados.loc[anomalia.index, 'Grad.Rate'] = 100
```

[17]:
```
sns.FacetGrid(dados, hue='Private', height=6).map(sns.histplot, 'Grad.Rate',␣
↪bins=30, alpha=0.7).add_legend()
```

[17]: <seaborn.axisgrid.FacetGrid at 0x239fe923ed0>



1.5 Criação de clusters "K Means"

Agora é hora de criar os rótulos de Cluster!

Importe KMeans da SciKit Learn.

[18]:
```
from sklearn.cluster import KMeans
```

Crie uma instância do modelo K Means com 2 clusters.

[19]:
```
kmeans = KMeans(n_clusters=2)
```

Fite o modelo para todos os dados, exceto para o rótulo privado.

[20]:
```
kmeans.fit(dados.drop('Private', axis=1))
```

[20]:
```
KMeans(n_clusters=2)
```

[21]:
```
centroids = kmeans.cluster_centers_
print("Centroids:\n", centroids)
```

```
Centroids:
 [[1.99097222e+03 1.34700585e+03 5.01001462e+02 2.66637427e+01
  5.46023392e+01 2.19326316e+03 5.53080409e+02 1.06887091e+04
  4.37517398e+03 5.44059942e+02 1.26739474e+03 7.10745614e+01
  7.83391813e+01 1.38330409e+01 2.35716374e+01 9.58258772e+03
  6.58815789e+01]
 [1.04349247e+04 6.95977419e+03 2.83176344e+03 3.41397849e+01
  6.45806452e+01 1.47810323e+04 3.07806452e+03 8.61637634e+03
  4.22773118e+03 5.88516129e+02 1.87936559e+03 8.43225806e+01
  8.97311828e+01 1.59774194e+01 1.66559140e+01 1.02307849e+04
  6.21935484e+01]]
```

Quais são os vetores centrais do cluster?

[22]:
```
vetores_centrais = kmeans.cluster_centers_
rotulos = kmeans.labels_
print(vetores_centrais)
print(rotulos)
```

```
[[1.99097222e+03 1.34700585e+03 5.01001462e+02 2.66637427e+01
  5.46023392e+01 2.19326316e+03 5.53080409e+02 1.06887091e+04
  4.37517398e+03 5.44059942e+02 1.26739474e+03 7.10745614e+01
  7.83391813e+01 1.38330409e+01 2.35716374e+01 9.58258772e+03
  6.58815789e+01]
 [1.04349247e+04 6.95977419e+03 2.83176344e+03 3.41397849e+01
  6.45806452e+01 1.47810323e+04 3.07806452e+03 8.61637634e+03
  4.22773118e+03 5.88516129e+02 1.87936559e+03 8.43225806e+01
  8.97311828e+01 1.59774194e+01 1.66559140e+01 1.02307849e+04
  6.21935484e+01]]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0
 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 1 0 0 0 0
 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 1 0 0 0 1 1 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0
 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 1 0 0 1 0 0 0 0 1 1 0 1 1 1 0 1 0 1
0 0 0 0 1 1 0 0 1 0 0 1 0 1 0 0 0 0 1 0 1 0 1 1 0 0 0 0 1 0 1 0 1 1 1 1 0
0 1 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 1 1 1 0 0 0 0 1 0 1 1 0 0 0 0 0 1 1 0
0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
```

1.6 Avaliação

Não há uma maneira perfeita de avaliar o agrupamento se você não tiver os rótulos, no entanto, como isso é apenas um exercício, temos os rótulos então aproveitamos isso para avaliar nossos clusters. Tenha em mente que não terá esse luxo no mundo real.

Crie uma nova coluna para df chamado 'Cluster', que é 1 para escola particular e 0 para uma escola pública.

[23]:
```
dados['Cluster'] = np.where(dados['Private'] == 'Yes', 1, 0)
```

[24]:
```
dados.head()
```

[24]:

	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD	Terminal	S.F.Ratio	perc.alumni	Expend	Grad.Rate
Abilene Christian University	Yes	1660	1232	721	23	52	2885	537	7440	3300	450	2200	70	78	18.1	12	7041	60
Adelphi University	Yes	2186	1924	512	16	29	2683	1227	12280	6450	750	1500	29	30	12.2	16	10527	56
Adrian College	Yes	1428	1097	336	22	50	1036	99	11250	3750	400	1165	53	66	12.9	30	8735	54
Agnes Scott College	Yes	417	349	137	60	89	510	63	12960	5450	450	875	92	97	7.7	37	19016	59
Alaska Pacific University	Yes	193	146	55	16	44	249	869	7560	4120	800	1500	76	72	11.9	2	10922	15

```
                              Cluster
Abilene Christian University        1
Adelphi University                  1
Adrian College                      1
Agnes Scott College                 1
Alaska Pacific University           1
```

Crie uma matriz de confusão e um relatório de classificação para ver o quão bem o clustering K Means funcionou sem ter nenhum rótulo.

[26]:
```
from sklearn.metrics import confusion_matrix, classification_report

matriz_confusão = confusion_matrix(dados['Cluster'], kmeans.labels_)
print("Matriz de Confusão:\n", matriz_confusão)

print("\nRelatório de Classificação:\n")
print(classification_report(dados['Cluster'], kmeans.labels_))
```

```
Matriz de Confusão:
 [[131  81]
 [553  12]]

Relatório de Classificação:

              precision    recall  f1-score   support

           0       0.19      0.62      0.29       212
           1       0.13      0.02      0.04       565

    accuracy                           0.18       777
   macro avg       0.16      0.32      0.16       777
weighted avg       0.15      0.18      0.11       777
```

Há 212 universidades públicas (rótulo verdadeiro 0) e 565 privadas (rótulo verdadeiro 1), totalizando 777 observações. O cluster 0 capturou a maioria das privadas (553 de 565, ou ~98%), mas também incluiu 131 públicas. O cluster 1 capturou poucas privadas (12) e 81 públicas. Então, cluster 0 está fortemente associado às universidades privadas enquanto o cluster 1 está mais associado às públicas. Os centroids confirmam isso, já que: - Centroid 0: Valores menores (ex.: ~1.991 aplicações, ~501 matrículas), típico de instituições privadas menores. - Centroid 1: Valores maiores (ex.: ~10.435 aplicações, ~2.832 matrículas), típico de instituições públicas maiores.

A baixa precisão surge porque os rótulos estão "invertidos" em relação à sua definição de 'Cluster'. O K-Means agrupou razoavelmente bem, mas a comparação direta ignora essa arbitrariedade.

Vamos tentar, então aumentar essa precisão invertendo os labels.

[27]:
```
rotulos2 = kmeans.labels_
```

[ ]:
```
# Verifique qual cluster corresponde à maioria das privadas (1)
# Calcule a proporção de privadas em cada cluster
cluster_private_prop = [
  np.mean(dados['Cluster'][rotulos2 == 0]),  # Proporção de 1's no cluster 0
  np.mean(dados['Cluster'][rotulos2 == 1])   # Proporção de 1's no cluster 1
]

# Se o cluster 0 tiver mais privadas, inverta os labels
if cluster_private_prop[0] > cluster_private_prop[1]:
  mapped_labels = 1 - rotulos2  # Inverte 0<->1
else:
  mapped_labels = rotulos2

# Agora avalie com os labels mapeados
print("Matriz de Confusão:\n", confusion_matrix(dados['Cluster'],␣
 ↪mapped_labels))
print("\nRelatório de Classificação:\n",␣
 ↪classification_report(dados['Cluster'], mapped_labels))
```

```
Matriz de Confusão:
 [[ 81 131]
 [ 12 553]]

Relatório de Classificação:
              precision    recall  f1-score   support

           0       0.87      0.38      0.53       212
           1       0.81      0.98      0.89       565

    accuracy                           0.82       777
   macro avg       0.84      0.68      0.71       777
weighted avg       0.83      0.82      0.79       777
```

Precisão bem maior, mostrando que os rótulos estão melhores escolhidos para cada cluster

[31]:
```
from sklearn.metrics import silhouette_score
print("Pontuação da silhueta do cluster:", silhouette_score(dados.
 ↪drop('Private', axis=1), kmeans.labels_))
```

```
Pontuação da silhueta do cluster: 0.5365983302413596
```

O teste foi feito sem normalização dos dados e com dois cluster. Mas, podemos normalizar e descobrir quantos clusters seria o ideal.

[32]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
dados_scaled = scaler.fit_transform(dados.drop(['Private', 'Cluster'], axis=1))
kmeans.fit(dados_scaled)
```

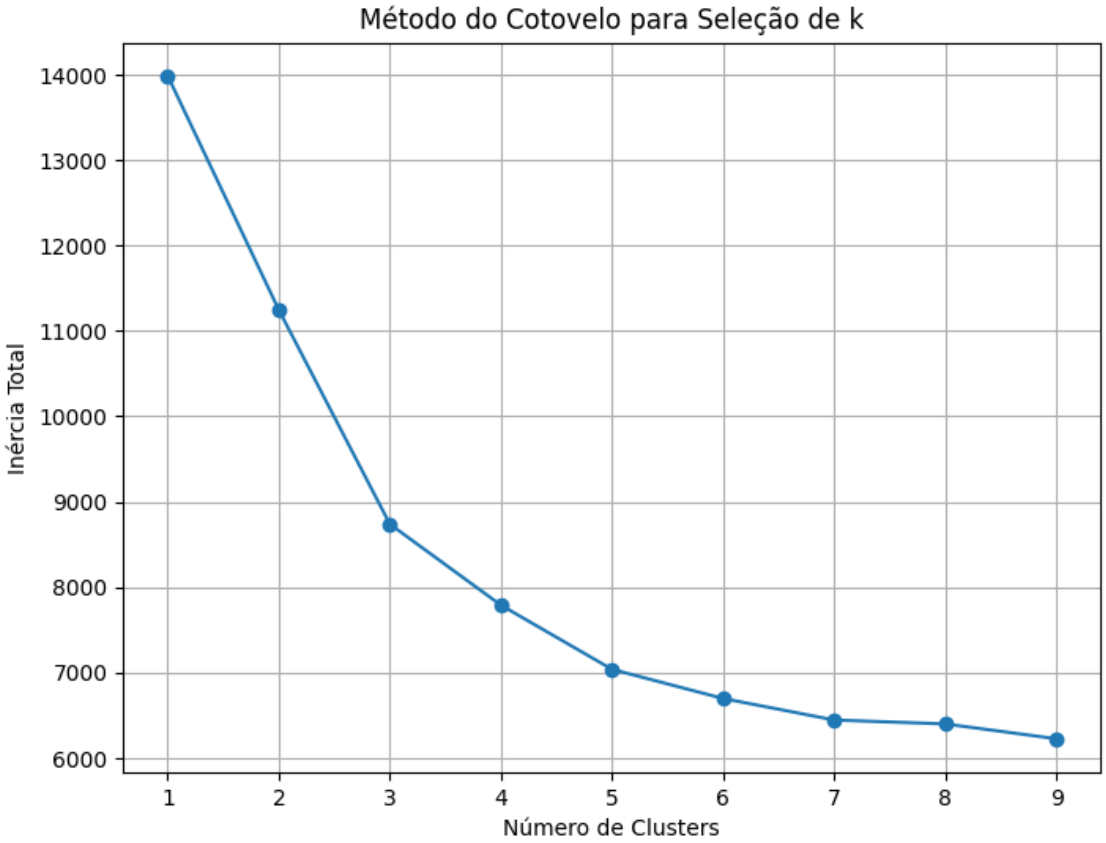
[32]: KMeans(n_clusters=2)

[ ]:
```
# Remover a coluna 'Private' e normalizar os dados
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Lista para armazenar inércias totais e por cluster
inertias_totais = []
inertias_por_cluster = []
num_clusters_range = range(1, 10)  # Testar de 1 a 9 clusters

# Calcular inércias para diferentes números de clusters
for k in num_clusters_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(X_scaled)

    # Inércia total
    inertias_totais.append(kmeans.inertia_)

    # Calcular inércia por cluster
    labels = kmeans.labels_
    centroids = kmeans.cluster_centers_
    inercia_cluster = []

    for cluster in range(k):
        # Selecionar pontos pertencentes ao cluster
        pontos_cluster = X_scaled[labels == cluster]
        if len(pontos_cluster) > 0:  # Verificar se o cluster não está vazio
            # Calcular a soma das distâncias quadradas ao centroide
            distancias = np.sum((pontos_cluster - centroids[cluster]) ** 2)
            inercia_cluster.append(distancias)
        else:
            inercia_cluster.append(0)  # Cluster vazio (raro, mas para␣
 ↪segurança)

    inertias_por_cluster.append(inercia_cluster)

# Calcular variação percentual da inércia total entre k e k-1
variacoes_percentuais = []
for i in range(1, len(inertias_totais)):
    variacao = ((inertias_totais[i-1] - inertias_totais[i]) /␣
 ↪inertias_totais[i-1]) * 100
    variacoes_percentuais.append(variacao)

print("Resultados da Inércia por Cluster e Variação Percentual:\n")
for k, inercia_total, inercia_cluster in zip(num_clusters_range,␣
 ↪inertias_totais, inertias_por_cluster):
```

```
    print(f"Número de Clusters: {k}")
    print(f"Inércia Total: {inercia_total:.2f}")
    print(f"Inércias por Cluster: {[round(x, 2) for x in inercia_cluster]}")
    if k > 1:
        print(f"Variação Percentual (de {k-1} para {k} clusters):␣
↪{variacoes_percentuais[k-2]:.2f}%")
    print("-" * 50)


plt.figure(figsize=(8, 6))
plt.plot(num_clusters_range, inertias_totais, marker='o')
plt.xlabel('Número de Clusters')
plt.ylabel('Inércia Total')
plt.title('Método do Cotovelo para Seleção de k')
plt.grid(True)
plt.show()
```

```
Resultados da Inércia por Cluster e Variação Percentual:

Número de Clusters: 1
Inércia Total: 13986.00
Inércias por Cluster: [np.float64(13986.0)]
--------------------------------------------------
Número de Clusters: 2
Inércia Total: 11246.80
Inércias por Cluster: [np.float64(6103.58), np.float64(5143.22)]
Variação Percentual (de 1 para 2 clusters): 19.59%
--------------------------------------------------
Número de Clusters: 3
Inércia Total: 8737.91
Inércias por Cluster: [np.float64(2763.34), np.float64(3780.45),
np.float64(2194.13)]
Variação Percentual (de 2 para 3 clusters): 22.31%
--------------------------------------------------
Número de Clusters: 4
Inércia Total: 7788.90
Inércias por Cluster: [np.float64(2736.88), np.float64(1426.69),
np.float64(1244.06), np.float64(2381.28)]
Variação Percentual (de 3 para 4 clusters): 10.86%
--------------------------------------------------
Número de Clusters: 5
Inércia Total: 7038.02
Inércias por Cluster: [np.float64(2012.99), np.float64(1295.59),
np.float64(1233.08), np.float64(1715.6), np.float64(780.75)]
Variação Percentual (de 4 para 5 clusters): 9.64%
--------------------------------------------------
Número de Clusters: 6
Inércia Total: 6696.43
```

```
Inércias por Cluster: [np.float64(1924.22), np.float64(822.48),
np.float64(711.32), np.float64(1612.22), np.float64(780.75), np.float64(845.45)]
Variação Percentual (de 5 para 6 clusters): 4.85%
--------------------------------------------------
Número de Clusters: 7
Inércia Total: 6444.54
Inércias por Cluster: [np.float64(1138.64), np.float64(799.63),
np.float64(711.32), np.float64(1310.27), np.float64(619.02),
np.float64(1043.19), np.float64(822.48)]
Variação Percentual (de 6 para 7 clusters): 3.76%
--------------------------------------------------
Número de Clusters: 8
Inércia Total: 6398.61
Inércias por Cluster: [np.float64(1924.22), np.float64(845.45),
np.float64(1555.17), np.float64(125.75), np.float64(785.02), np.float64(345.91),
np.float64(817.1), np.float64(0.0)]
Variação Percentual (de 7 para 8 clusters): 0.71%
--------------------------------------------------
Número de Clusters: 9
Inércia Total: 6225.02
Inércias por Cluster: [np.float64(1843.89), np.float64(614.25),
np.float64(1402.61), np.float64(165.55), np.float64(754.51), np.float64(313.32),
np.float64(640.48), np.float64(0.0), np.float64(490.42)]
Variação Percentual (de 8 para 9 clusters): 2.71%
--------------------------------------------------
```



[ ]:
```python
for k in num_clusters_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(X_scaled)
    if k > 1:  # Silhouette requer pelo menos 2 clusters
        score = silhouette_score(X_scaled, kmeans.labels_)
        print(f"Silhouette Score para k={k}: {score:.3f}")
```

```
Silhouette Score para k=2: 0.221
Silhouette Score para k=3: 0.227
Silhouette Score para k=4: 0.211
Silhouette Score para k=5: 0.189
Silhouette Score para k=6: 0.182
Silhouette Score para k=7: 0.133
Silhouette Score para k=8: 0.181
Silhouette Score para k=9: 0.166
```

Vemos que 3 cluster conseguem representar bem os resultados, mesmo que só haja duas classes (privada ou não privada). Logo, podemos aplicar o kmeans novamente e comparar com os resultados originais.

[37]:
```
# Aplicar K-Means com k=3
kmeans = KMeans(n_clusters=3, random_state=42)
kmeans.fit(X_scaled)
dados['Cluster'] = kmeans.labels_

# Analisar características dos clusters
cluster_summary = dados.groupby('Cluster').mean(numeric_only=True)
cluster_summary['Private_Prop'] = dados.groupby('Cluster')['Private'].
 ↪apply(lambda x: (x == 'Yes').mean())
print("Resumo das Características por Cluster (k=3):")
print(cluster_summary)
```

```
Resumo das Características por Cluster (k=3):
                Apps       Accept       Enroll  Top10perc  Top25perc  \
Cluster
0        2853.912214  1743.969466   570.366412  41.442748  71.053435
1        1340.626263   984.113636   405.444444  18.047980  44.116162
2        8854.285714  6067.075630  2487.789916  28.638655  61.075630

           F.Undergrad  P.Undergrad      Outstate   Room.Board       Books  \
Cluster
0          2285.793893   317.973282  14528.393130  5153.580153  560.534351
1          1844.398990   578.522727   8539.684343  3930.295455  531.328283
2         12987.966387  2959.352941   7766.773109  4026.579832  584.899160

             Personal        PhD   Terminal  S.F.Ratio  perc.alumni  \
Cluster
0         1105.893130  84.328244  90.316794  11.672137    32.412214
1         1344.300505  61.886364  70.068182  14.840404    18.689394
2         1845.310924  82.823529  88.394958  16.914286    14.949580

                Expend  Grad.Rate  True_Label  Private_Prop
Cluster
0         13630.553435  78.156489    0.977099      0.977099
1          7340.419192  59.181818    0.760101      0.760101
2          8638.168067  58.268908    0.067227      0.067227
```

Temos então o seguinte: - Cluster 0 = representa universidades privadas de elite, com alta seletividade, turmas menores, altas mensalidades e despesas por aluno, e fortes taxas de graduação. - Cluster 1 = inclui principalmente universidades privadas menores e menos seletivas, mas também algumas públicas menores (menos recursos, já que tem o menos custo por aluno e custo por sala) - Cluster 2 = representa principalmente universidades públicas grandes, com grande número de aplicações, matrículas e alunos de meio período, mensalidades mais baixas e turmas maiores.

[38]:
```
# Comparar com rótulos verdadeiros
dados['True_Label'] = np.where(dados['Private'] == 'Yes', 1, 0)
confusion_matrix = pd.crosstab(dados['True_Label'], dados['Cluster'],␣
 ↪rownames=['True_Label'], colnames=['Cluster'])
```

```
print("\nMatriz de Confusão (k=3):")
print(confusion_matrix)
```

```
Matriz de Confusão (k=3):
Cluster       0    1    2
True_Label
0             6   95  111
1           256  301    8
```

Se colocarmos o cluster 0 e 1 como privadas, mesmo a 1 incluindo algumas faculdades públicas, a acurácia é de 86%, levemente melhor do que a anterior, que deu 83%. Mas, há limitações já que a baixa Silhouette Score (0.227) indica que os clusters não são altamente distintos, provavelmente devido a sobreposições nas features (ex.: Enroll' não separa perfeitamente públicas de privadas). O Cluster 1 é misto, o que reduz a pureza dos clusters em relação aos rótulos verdadeiros.

Cluster 1 contém uma mistura de privadas (301) e públicas (95). Para entender melhor, vamos analisar esse cluster

[39]:
```
print("Universidades no Cluster 1:")
print(dados[dados['Cluster'] == 1][['Private'] + list(X.columns)].head(10))
```

```
Universidades no Cluster 1:
                                           Private  Apps  Accept  Enroll  \
Abilene Christian University                   Yes  1660    1232     721
Adelphi University                             Yes  2186    1924     512
Adrian College                                 Yes  1428    1097     336
Alaska Pacific University                      Yes   193     146      55
Albertson College                              Yes   587     479     158
Alderson-Broaddus College                      Yes   582     498     172
Allentown Coll. of St. Francis de Sales        Yes  1179     780     290
Alverno College                                Yes   494     313     157
American International College                 Yes  1420    1093     220
Anderson University                            Yes  1216     908     423

                                           Top10perc  Top25perc  F.Undergrad  \
Abilene Christian University                      23         52         2885
Adelphi University                                16         29         2683
Adrian College                                    22         50         1036
Alaska Pacific University                         16         44          249
Albertson College                                 38         62          678
Alderson-Broaddus College                         21         44          799
Allentown Coll. of St. Francis de Sales           38         64         1130
Alverno College                                   23         46         1317
American International College                     9         22         1018
Anderson University                               19         40         1819

                                           P.Undergrad  Outstate  Room.Board  \
Abilene Christian University                       537      7440        3300
```

Adelphi University	1227	12280	6450
Adrian College	99	11250	3750
Alaska Pacific University	869	7560	4120
Albertson College	41	13500	3335
Alderson-Broaddus College	78	10468	3380
Allentown Coll. of St. Francis de Sales	638	9690	4785
Alverno College	1235	8352	3640
American International College	287	8700	4780
Anderson University	281	10100	3520

	Books	Personal	PhD	Terminal	\
Abilene Christian University	450	2200	70	78	
Adelphi University	750	1500	29	30	
Adrian College	400	1165	53	66	
Alaska Pacific University	800	1500	76	72	
Albertson College	500	675	67	73	
Alderson-Broaddus College	660	1800	40	41	
Allentown Coll. of St. Francis de Sales	600	1000	60	84	
Alverno College	650	2449	36	69	
American International College	450	1400	78	84	
Anderson University	550	1100	48	61	

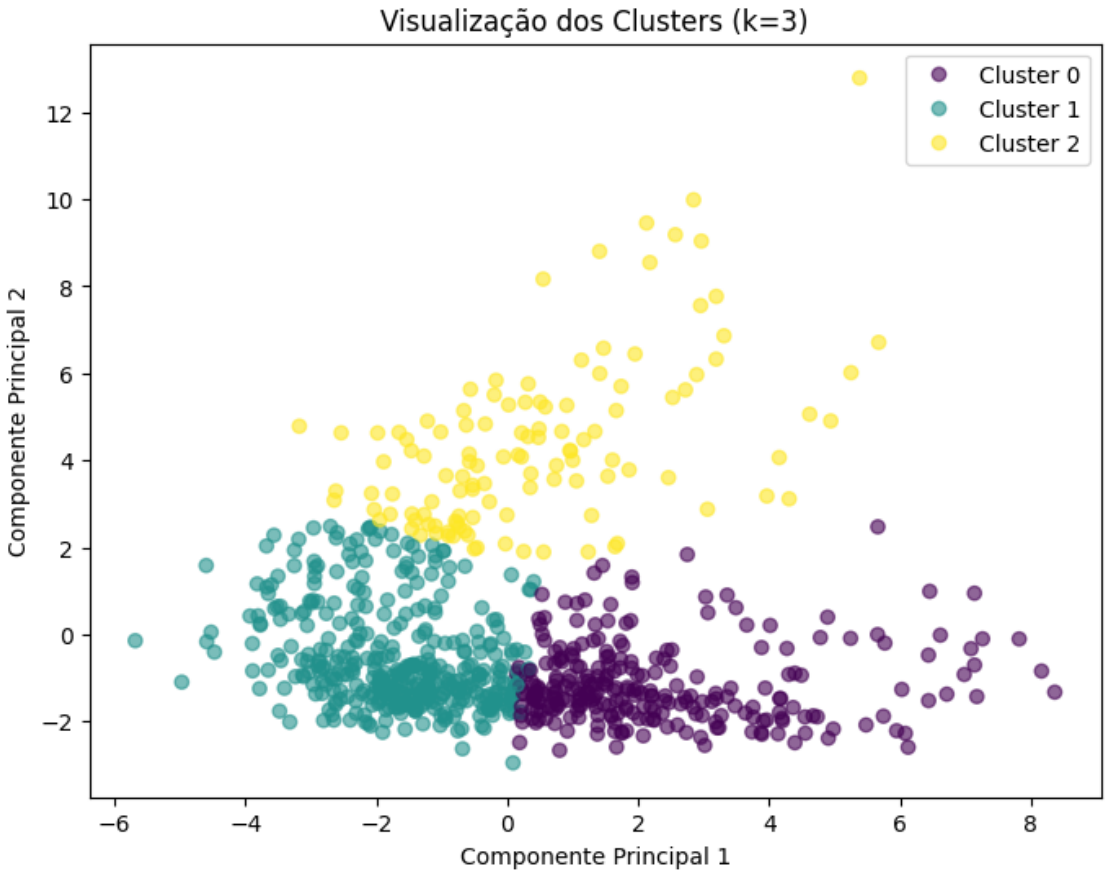
	S.F.Ratio	perc.alumni	Expend	\
Abilene Christian University	18.1	12	7041	
Adelphi University	12.2	16	10527	
Adrian College	12.9	30	8735	
Alaska Pacific University	11.9	2	10922	
Albertson College	9.4	11	9727	
Alderson-Broaddus College	11.5	15	8991	
Allentown Coll. of St. Francis de Sales	13.3	21	7940	
Alverno College	11.1	26	8127	
American International College	14.7	19	7355	
Anderson University	12.1	14	7994	

	Grad.Rate	Cluster
Abilene Christian University	60	1
Adelphi University	56	1
Adrian College	54	1
Alaska Pacific University	15	1
Albertson College	55	1
Alderson-Broaddus College	52	1
Allentown Coll. of St. Francis de Sales	74	1
Alverno College	55	1
American International College	69	1
Anderson University	59	1

O Cluster 1 agrupa principalmente privadas menores e menos seletivas, mas inclui públicas pequenas que compartilham características como baixo número de aplicações e matrículas. Isso explica a

sobreposição com Cluster 0 (privadas de elite) e a dificuldade em separação pura.

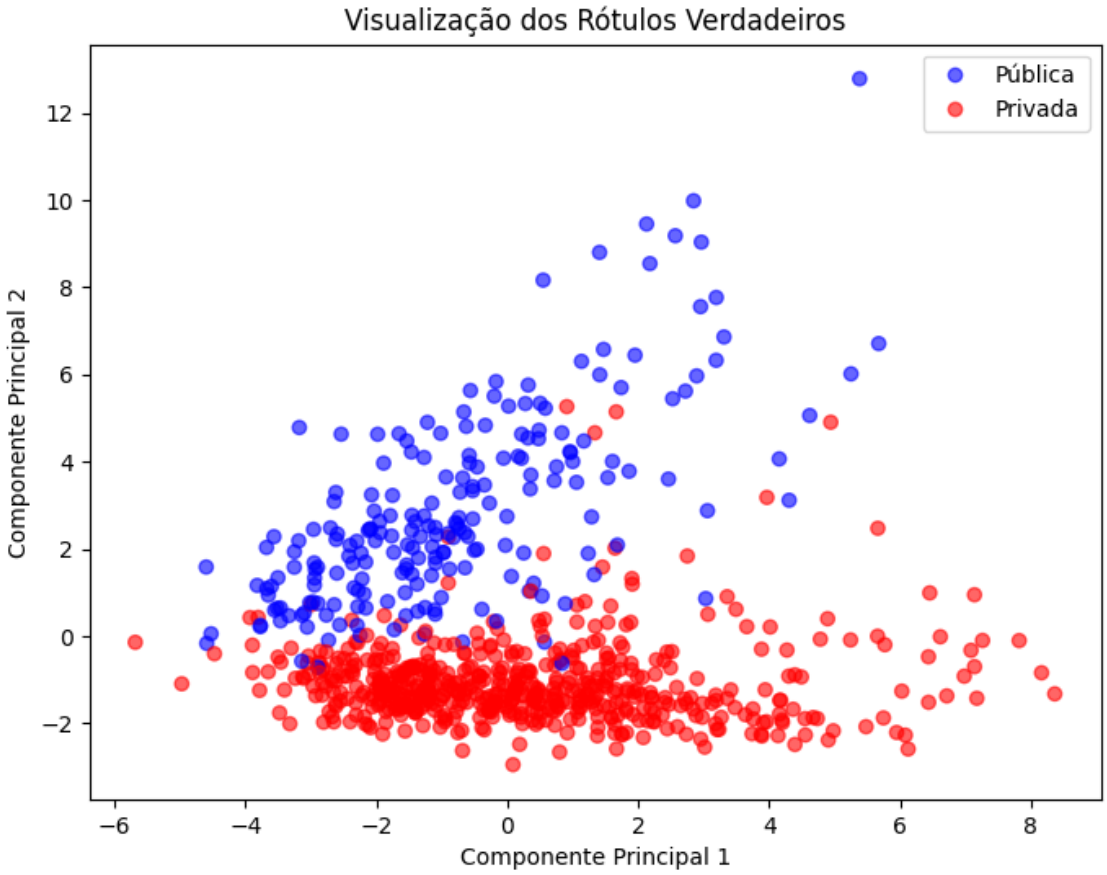
[40]:
```
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)
plt.figure(figsize=(8, 6))
scatter = plt.scatter(X_pca[:, 0], X_pca[:, 1], c=dados['Cluster'],␣
 ↪cmap='viridis', alpha=0.6)
plt.legend(handles=scatter.legend_elements()[0], labels=['Cluster 0', 'Cluster␣
 ↪1', 'Cluster 2'])
plt.xlabel('Componente Principal 1')
plt.ylabel('Componente Principal 2')
plt.title('Visualização dos Clusters (k=3)')
plt.show()
```



A separação não é perfeita, com sobreposição entre Cluster 0 e Cluster 1, o que é consistente com o Silhouette Score baixo (0.227) e a mistura de públicas e privadas no Cluster 1 (conforme a matriz de confusão anterior). Cluster 2 (amarelo) é o mais distinto, se alinhando com as públicas grandes,

que têm características mais homogêneas (ex.: alto número de aplicações e matrículas).

[41]:
```
plt.figure(figsize=(8, 6))
scatter = plt.scatter(X_pca[:, 0], X_pca[:, 1], c=dados['True_Label'],␣
 ↪cmap='bwr', alpha=0.6)
plt.legend(handles=scatter.legend_elements()[0], labels=['Pública', 'Privada'])
plt.xlabel('Componente Principal 1')
plt.ylabel('Componente Principal 2')
plt.title('Visualização dos Rótulos Verdadeiros')
plt.show()
```



É possível ver uma sobreposição significativa entre públicas e privadas, especialmente na região central (valores próximos de 0 em ambas as componentes principais). A sobreposição confirma que as features originais não separam perfeitamente públicas de privadas, o que explica a dificuldade do K-Means em alcançar clusters puros com $ k=2 $ ou $ k=3 $.

[44]:
```
from sklearn.cluster import DBSCAN
dbscan = DBSCAN(eps=1.0, min_samples=3)
labels_dbscan = dbscan.fit_predict(X_scaled)
```

```
print("Número de clusters (DBSCAN):", len(np.unique(labels_dbscan)) - (1 if -1␣
↪in labels_dbscan else 0))
```

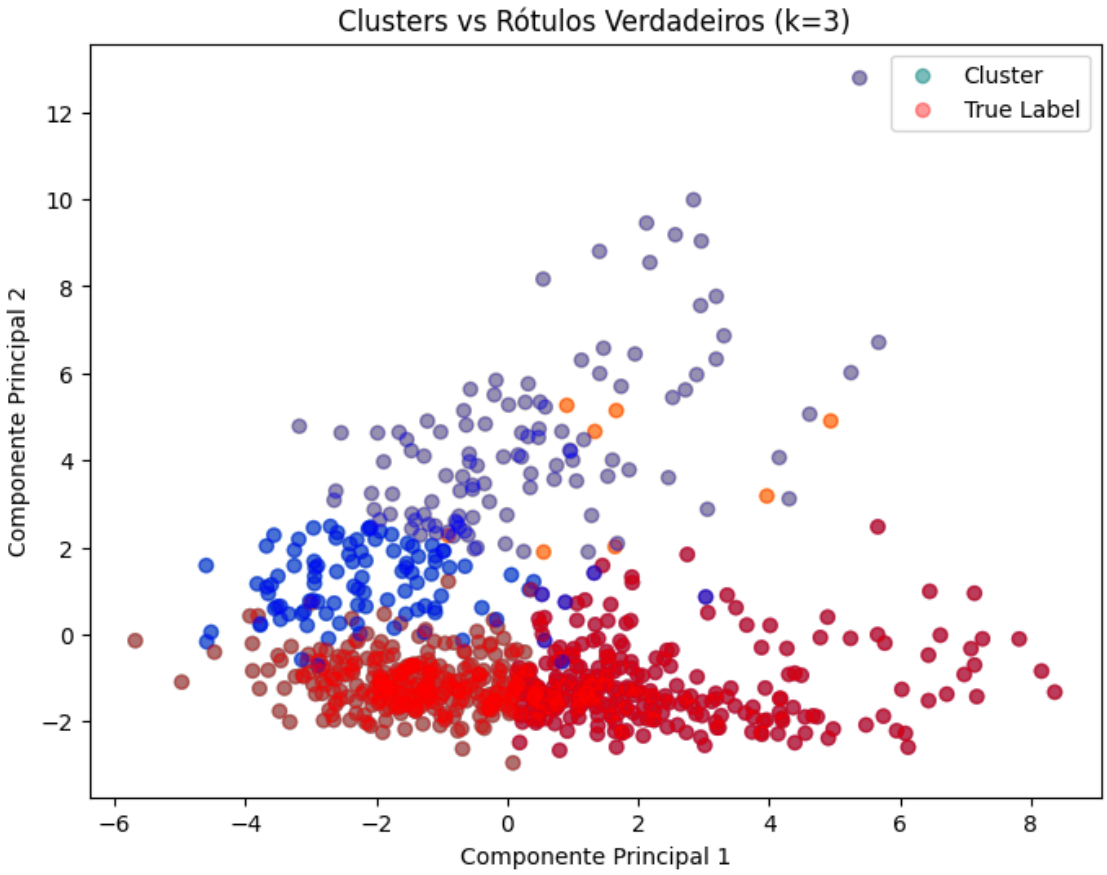
```
Número de clusters (DBSCAN): 0
```

[43]:
```
from sklearn.metrics import adjusted_rand_score
print("Adjusted Rand Index (k=3):", adjusted_rand_score(dados['True_Label'],␣
↪dados['Cluster']))
```

```
Adjusted Rand Index (k=3): 0.2090580468768935
```

Um ARI de $0.209 $ é moderado, sugerindo que o clustering com $ k=3 $ tem alguma correspondência com os rótulos verdadeiros (públicas vs. privadas), mas não é forte. Comparado à acurácia ajustada (86% com mapeamento), o ARI baixo reflete a sobreposição e a natureza não supervisionada do K-Means, que não foi otimizado para os rótulos verdadeiros.

[45]:
```
plt.figure(figsize=(8, 6))
scatter = plt.scatter(X_pca[:, 0], X_pca[:, 1], c=dados['Cluster'],␣
↪cmap='viridis', alpha=0.6, label='Cluster')
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=dados['True_Label'], cmap='bwr',␣
↪alpha=0.4, label='True Label')
plt.legend()
plt.xlabel('Componente Principal 1')
plt.ylabel('Componente Principal 2')
plt.title('Clusters vs Rótulos Verdadeiros (k=3)')
plt.show()
```



[46]:
```
for eps in [0.8, 1.0, 1.5]:
    for ms in [3, 5, 10]:
        dbscan = DBSCAN(eps=eps, min_samples=ms)
        labels_dbscan = dbscan.fit_predict(X_scaled)
        n_clusters = len(np.unique(labels_dbscan)) - (1 if -1 in labels_dbscan␣
 ↪else 0)
        print(f"eps={eps}, min_samples={ms}, Número de clusters: {n_clusters}")
```

```
eps=0.8, min_samples=3, Número de clusters: 0
eps=0.8, min_samples=5, Número de clusters: 0
eps=0.8, min_samples=10, Número de clusters: 0
eps=1.0, min_samples=3, Número de clusters: 0
eps=1.0, min_samples=5, Número de clusters: 0
eps=1.0, min_samples=10, Número de clusters: 0
eps=1.5, min_samples=3, Número de clusters: 12
eps=1.5, min_samples=5, Número de clusters: 1
eps=1.5, min_samples=10, Número de clusters: 4
```

[47]:
```
dados['Acceptance_Rate'] = dados['Accept'] / dados['Apps']
X_new = dados.drop(['Private', 'Cluster'], axis=1)
X_new_scaled = scaler.fit_transform(X_new)
kmeans_new = KMeans(n_clusters=3, random_state=42)
kmeans_new.fit(X_new_scaled)
dados['Cluster_New'] = kmeans_new.labels_
print(dados.groupby('Cluster_New').mean(numeric_only=True))
```

```
                    Apps       Accept       Enroll  Top10perc  Top25perc  \
Cluster_New
0            1176.830882   904.294118   371.950980  18.941176  45.004902
1            8096.625000  5491.673611  2234.909722  26.729167  59.159722
2            3049.831111  1817.146667   588.693333  43.715556  73.213333

              F.Undergrad  P.Undergrad      Outstate   Room.Board       Books  \
Cluster_New
0             1607.210784   510.073529   8936.480392  4000.816176  534.068627
1            11746.256944  2688.298611   7536.993056  3988.395833  581.020833
2             2345.000000   308.186667  15026.617778  5240.604444  556.897778

                Personal        PhD   Terminal  S.F.Ratio  perc.alumni  \
Cluster_New
0            1310.696078  62.328431  70.475490  14.547794    19.943627
1            1806.791667  81.708333  87.486111  16.955556    14.541667
2            1096.608889  85.604444  91.453333  11.424889    33.071111

                  Expend  Grad.Rate   Cluster  True_Label  Acceptance_Rate
Cluster_New
0             7525.767157  60.517157  0.914216    0.821078         0.794834
1             8328.229167  57.979167  1.812500    0.062500         0.695215
2            14383.000000  79.142222  0.000000    0.982222         0.693154
```

[48]:
```
def purity_score(y_true, y_pred):
    contingency_matrix = pd.crosstab(y_true, y_pred)
    return np.sum(np.max(contingency_matrix.values, axis=0)) / np.
 ↪sum(contingency_matrix.values)
print("Pureza dos Clusters (k=3):", purity_score(dados['True_Label'],␣
 ↪dados['Cluster']))
```

```
Pureza dos Clusters (k=3): 0.8597168597168597
```

[49]:
```
public_in_cluster1 = dados[(dados['Cluster'] == 1) & (dados['True_Label'] == 0)]
print("Públicas no Cluster 1:")
print(public_in_cluster1[['Private'] + list(X.columns)].head(10))
```

```
Públicas no Cluster 1:
                                        Private  Apps  Accept  Enroll  \
Angelo State University                      No  3540    2001    1016
Arkansas Tech University                     No  1734    1729     951
```

```
                                        Private Apps Accept Enroll  \
Bemidji State University                      No  1208    877    546
Bloomsburg Univ. of Pennsylvania              No  6773   3028   1025
Castleton State College                       No  1257    940    363
Central Connecticut State University          No  4158   2532    902
Central Washington University                 No  2785   2011   1007
Christopher Newport University                No   883    766    428
Clinch Valley Coll. of  the Univ. of Virginia No   689    561    250
Delta State University                        No   967    945    459

                                        Top10perc  Top25perc  \
Angelo State University                        24         54
Arkansas Tech University                       12         52
Bemidji State University                       12         36
Bloomsburg Univ. of Pennsylvania               15         55
Castleton State College                         9         22
Central Connecticut State University            6         24
Central Washington University                   8         65
Christopher Newport University                  3         37
Clinch Valley Coll. of  the Univ. of Virginia  15         30
Delta State University                         15         48

                                        F.Undergrad  P.Undergrad  \
Angelo State University                        4190         1512
Arkansas Tech University                       3602          939
Bemidji State University                       3796          824
Bloomsburg Univ. of Pennsylvania               5847          946
Castleton State College                        1547          294
Central Connecticut State University           6394         3881
Central Washington University                  6507          898
Christopher Newport University                 2910         1749
Clinch Valley Coll. of  the Univ. of Virginia  1125          422
Delta State University                         2806          538

                                        Outstate  Room.Board  Books  \
Angelo State University                     5130        3592    500
Arkansas Tech University                    3460        2650    450
Bemidji State University                    4425        2700    660
Bloomsburg Univ. of Pennsylvania            7844        2948    500
Castleton State College                     7656        4690    400
Central Connecticut State University        5962        4444    500
Central Washington University               7242        3603    654
Christopher Newport University              7860        4750    525
Clinch Valley Coll. of  the Univ. of Virginia 7168      3689    600
Delta State University                      4528        1880    500

                                        Personal  PhD  Terminal  \
Angelo State University                     2000   60        62
Arkansas Tech University                    1000   57        60
```

Bemidji State University	1800	57	62
Bloomsburg Univ. of Pennsylvania	1680	66	68
Castleton State College	700	89	91
Central Connecticut State University	985	69	73
Central Washington University	1416	67	89
Christopher Newport University	1889	80	82
Clinch Valley Coll. of the Univ. of Virginia	1900	67	67
Delta State University	1200	49	63

	S.F.Ratio	perc.alumni	Expend	\
Angelo State University	23.1	5	4010	
Arkansas Tech University	19.6	5	4739	
Bemidji State University	19.6	16	3752	
Bloomsburg Univ. of Pennsylvania	18.0	19	7041	
Castleton State College	14.7	8	6318	
Central Connecticut State University	16.7	4	4900	
Central Washington University	18.1	0	6413	
Christopher Newport University	21.2	16	4639	
Clinch Valley Coll. of the Univ. of Virginia	18.1	9	4417	
Delta State University	17.1	16	5113	

	Grad.Rate	Cluster
Angelo State University	34	1
Arkansas Tech University	48	1
Bemidji State University	46	1
Bloomsburg Univ. of Pennsylvania	75	1
Castleton State College	79	1
Central Connecticut State University	49	1
Central Washington University	51	1
Christopher Newport University	48	1
Clinch Valley Coll. of the Univ. of Virginia	46	1
Delta State University	58	1