Experiment 12

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

```
df = pd.read_csv('College_Data.csv')
```

```
df.head(3)
```

	Unnamed: 0	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board
0	Abilene Christian University	Yes	1660	1232	721	23	52	2885	537	7440	3300
1	Adelphi University	Yes	2186	1924	512	16	29	2683	1227	12280	6450
2	Adrian College	Yes	1428	1097	336	22	50	1036	99	11250	3750

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 777 entries, 0 to 776
Data columns (total 19 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Unnamed: 0   777 non-null    object
 1   Private      777 non-null    object
 2   Apps         777 non-null    int64
 3   Accept       777 non-null    int64
 4   Enroll       777 non-null    int64
 5   Top10perc    777 non-null    int64
 6   Top25perc    777 non-null    int64
 7   F.Undergrad  777 non-null    int64
 8   P.Undergrad  777 non-null    int64
 9   Outstate     777 non-null    int64
 10  Room.Board   777 non-null    int64
 11  Books        777 non-null    int64
 12  Personal     777 non-null    int64
 13  PhD          777 non-null    int64
 14  Terminal     777 non-null    int64
 15  S.F.Ratio    777 non-null    float64
 16  perc.alumni  777 non-null    int64
 17  Expend       777 non-null    int64
 18  Grad.Rate    777 non-null    int64
dtypes: float64(1), int64(16), object(2)
memory usage: 115.5+ KB
```

```
df.isnull().sum()
```

	0
Unnamed: 0	0
Private	0
Apps	0
Accept	0
Enroll	0
Top10perc	0
Top25perc	0
F.Undergrad	0
P.Undergrad	0
Outstate	0
Room.Board	0
Books	0
Personal	0
PhD	0
Terminal	0
S.F.Ratio	0
perc.alumni	0
Expend	0
Grad.Rate	0

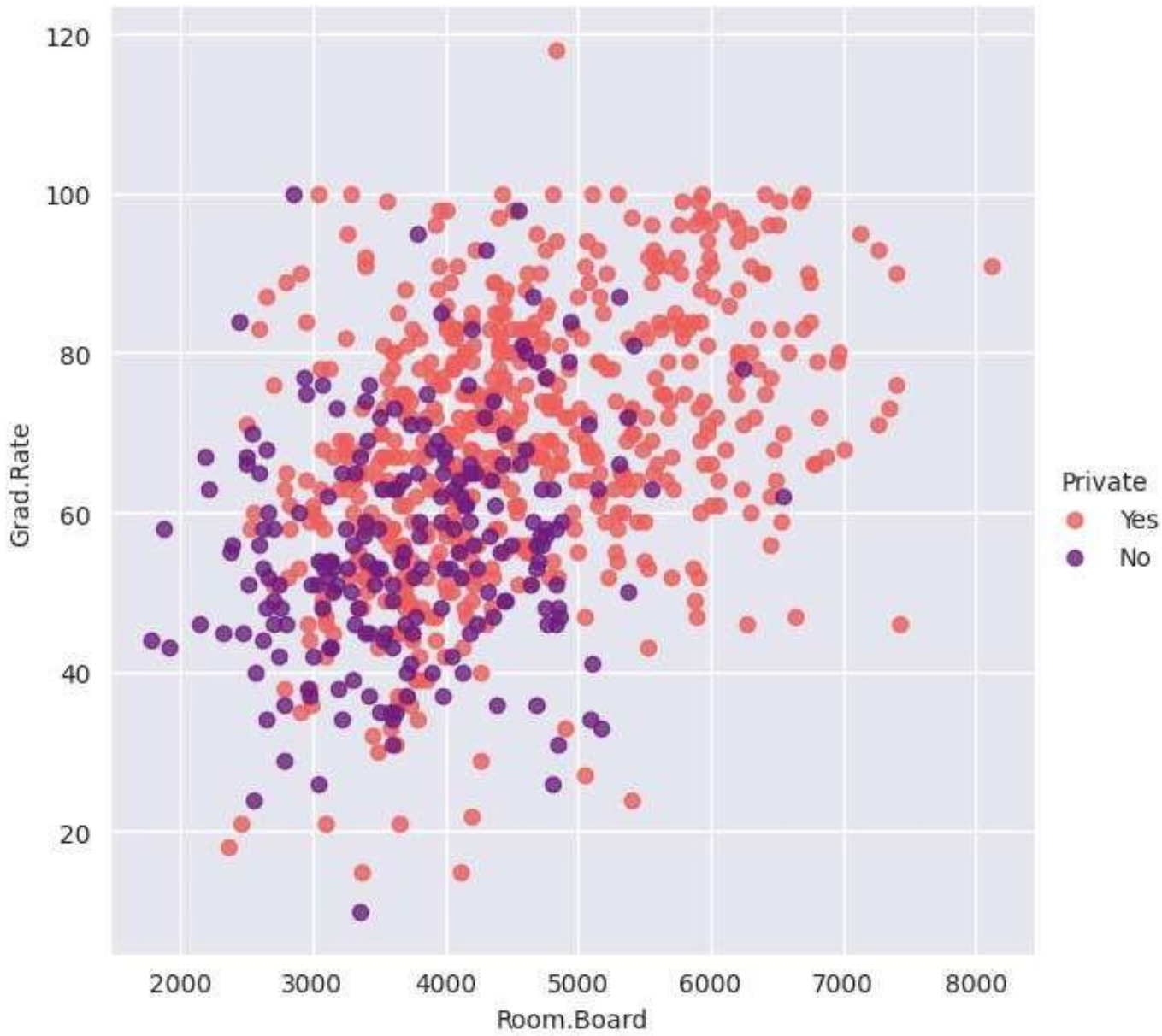
dtype: int64

```
df.describe()
```

	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Rc
count	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	7
mean	3001.638353	2018.804376	779.972973	27.558559	55.796654	3699.907336	855.298584	10440.669241	43
std	3870.201484	2451.113971	929.176190	17.640364	19.804778	4850.420531	1522.431887	4023.016484	10
min	81.000000	72.000000	35.000000	1.000000	9.000000	139.000000	1.000000	2340.000000	17
25%	776.000000	604.000000	242.000000	15.000000	41.000000	992.000000	95.000000	7320.000000	35
50%	1558.000000	1110.000000	434.000000	23.000000	54.000000	1707.000000	353.000000	9990.000000	42
75%	3624.000000	2424.000000	902.000000	35.000000	69.000000	4005.000000	967.000000	12925.000000	50
max	48094.000000	26330.000000	6392.000000	96.000000	100.000000	31643.000000	21836.000000	21700.000000	81

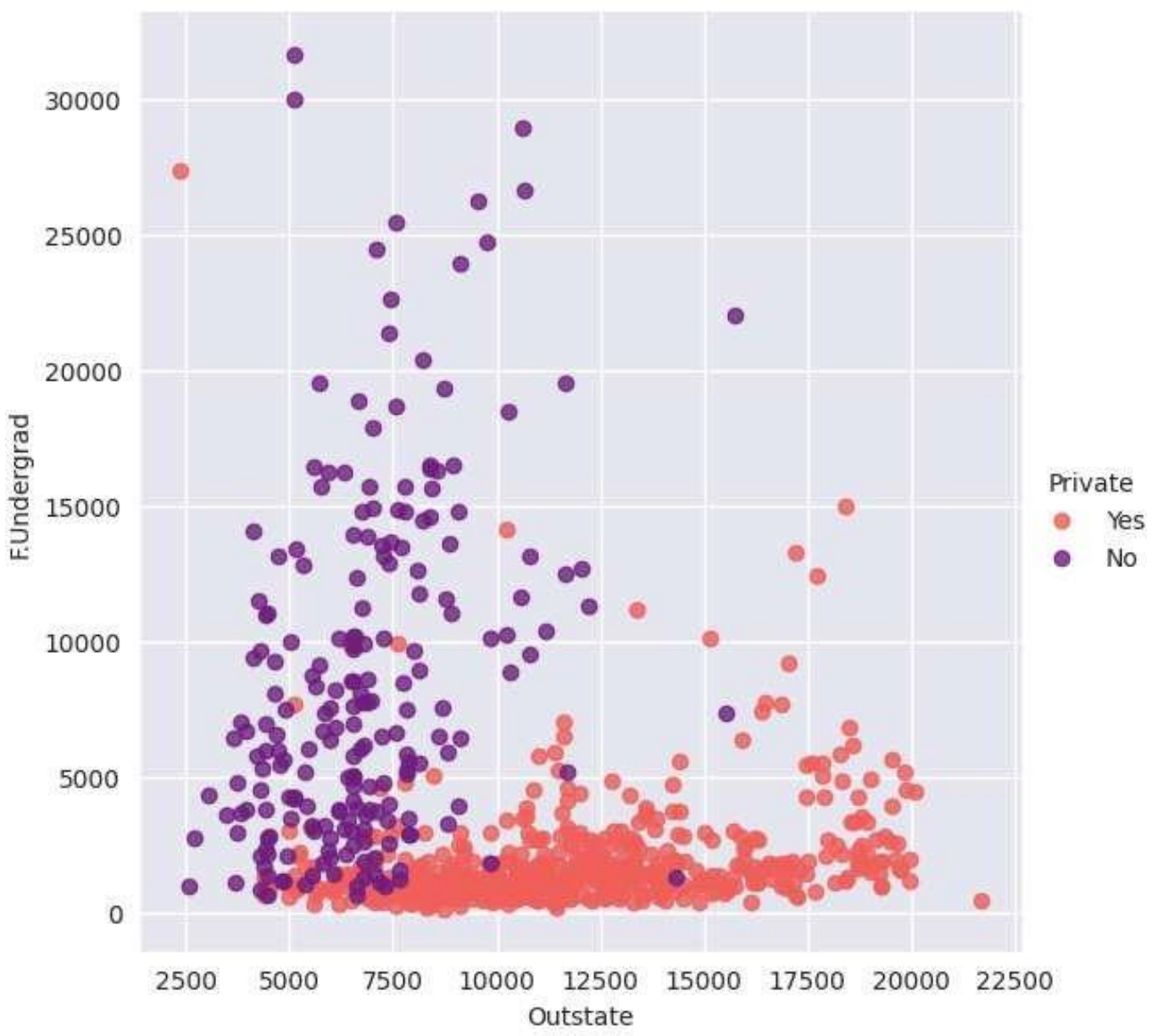
```
sns.set_style('darkgrid')
sns.lmplot(x='Room.Board', y='Grad.Rate', data=df,
           hue='Private', palette='magma_r', height=6, aspect=1, fit_reg=False)
```

```
<seaborn.axisgrid.FacetGrid at 0x79e78de4fe10>
```

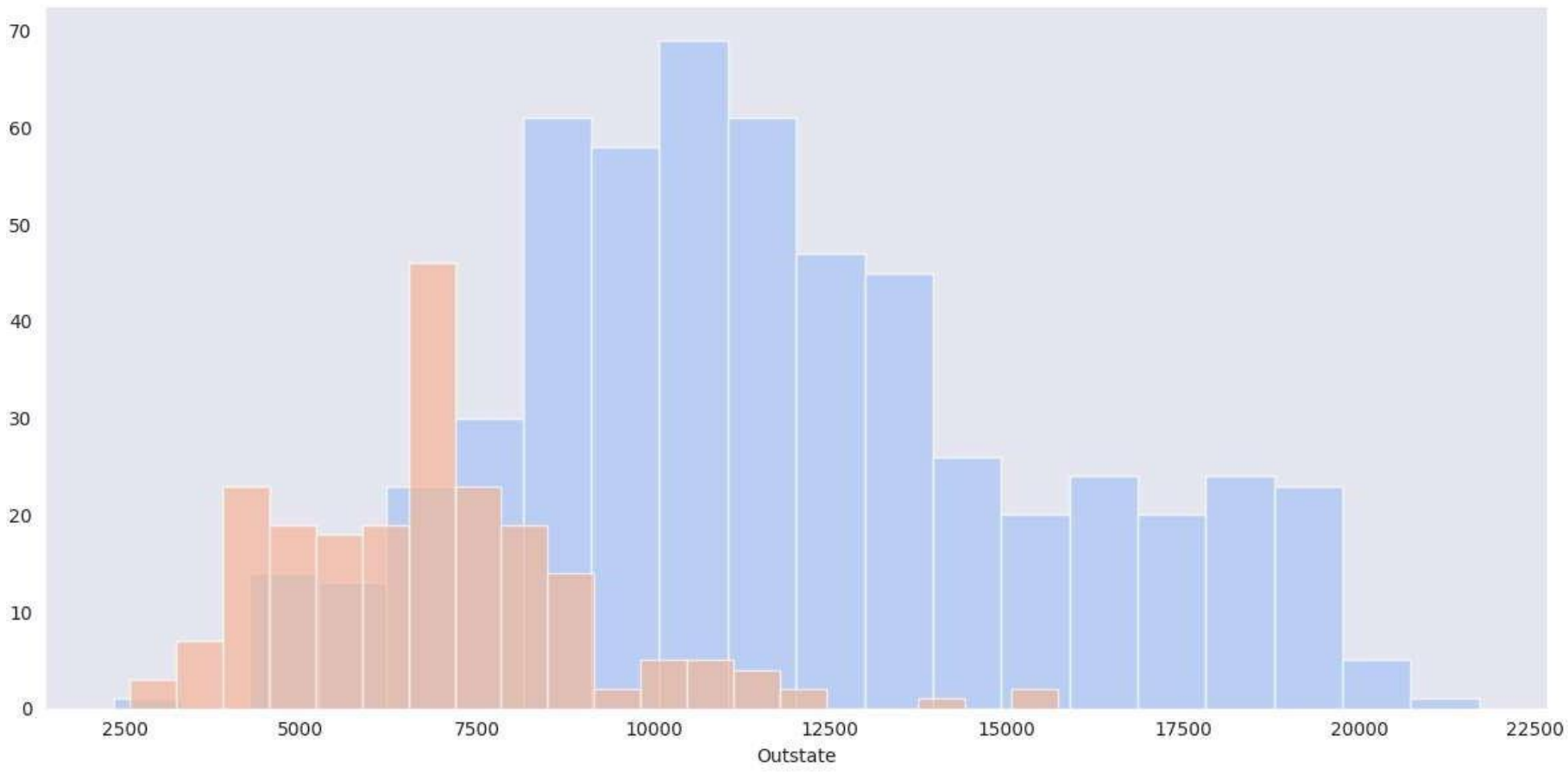



```
sns.set_style('darkgrid')
sns.lmplot(x='Outstate', y='F.Undergrad', data=df, hue='Private', palette='magma_r', height=6, aspect=1, fit_reg=Fals
```

```
<seaborn.axisgrid.FacetGrid at 0x79e78dad3b90>
```




```
sns.set_style('dark')
# Replaced 'size' with 'height' as 'size' is deprecated in newer seaborn versions.
h = sns.FacetGrid(df,hue="Private",palette='coolwarm',height=6,aspect=2)
h = h.map(plt.hist,'Outstate',bins=20,alpha=0.7)
```

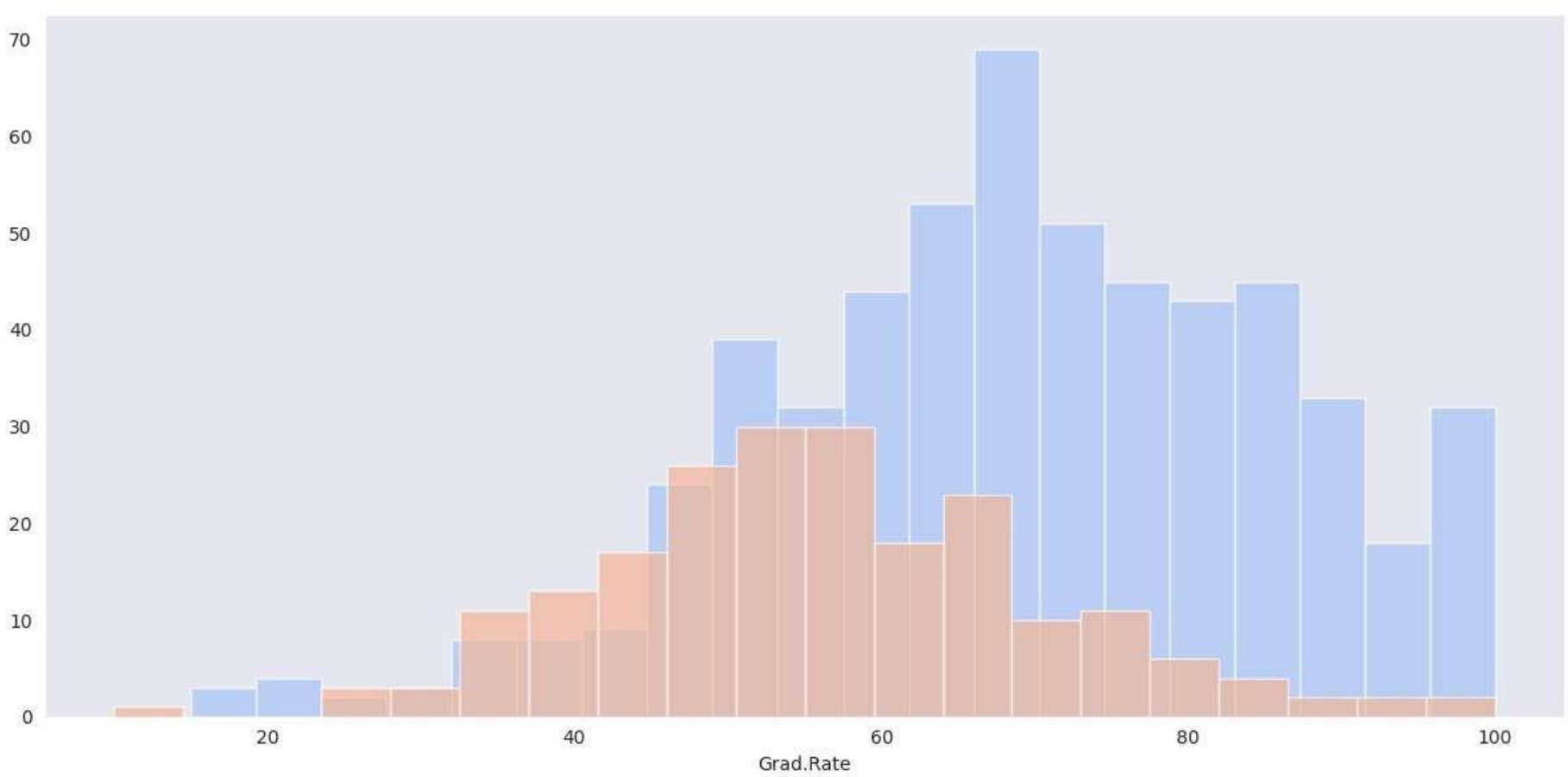



```
# df.set_value(95, 'Grad.Rate', 100) # Deprecated method
df.at[95, 'Grad.Rate'] = 100 # Use .at accessor for label-based indexing

df[df['Grad.Rate'] > 100]
```

Unnamed: 0	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	B

```
# Replaced 'size' with 'height' as 'size' is deprecated in newer seaborn versions.
g = sns.FacetGrid(df,hue="Private",palette='coolwarm',height=6,aspect=2)
g = g.map(plt.hist,'Grad.Rate',bins=20,alpha=0.7)
```




```
print(df.columns)
```

```
Index(['Unnamed: 0', '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'],
      dtype='object')
```

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=2)
```

```
kmeans.fit(df.drop(['Private', 'Unnamed: 0'], axis=1))
```

```
KMeans
KMeans(n_clusters=2)
```

```
means=kmeans.cluster_centers_
print(means)
```

```
[[1.81323468e+03 1.28716592e+03 4.91044843e+02 2.53094170e+01
  5.34708520e+01 2.18854858e+03 5.95458894e+02 1.03957085e+04
  4.31136472e+03 5.41982063e+02 1.28033632e+03 7.04424514e+01
  7.78251121e+01 1.40997010e+01 2.31748879e+01 8.93204634e+03
  6.50926756e+01]
 [1.03631389e+04 6.55089815e+03 2.56972222e+03 4.14907407e+01
  7.02037037e+01 1.30619352e+04 2.46486111e+03 1.07191759e+04
  4.64347222e+03 5.95212963e+02 1.71420370e+03 8.63981481e+01
  9.13333333e+01 1.40277778e+01 2.00740741e+01 1.41705000e+04
  6.75925926e+01]]
```

```
def converter(cluster):
    if cluster=='Yes':
        return 1
    else:
        return 0
df['Cluster'] = df['Private'].apply(converter)
df.head(3)
```

Unnamed: 0	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board

	Unnamed: 0	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board
0	Abilene Christian University	Yes	1660	1232	721	23	52	2885	537	7440	3300
1	Adelphi University	Yes	2186	1924	512	16	29	2683	1227	12280	6450
2	Adrian College	Yes	1428	1097	336	22	50	1036	99	11250	3750

Conclusion:-

In this experiment, we understand how to evaluate the performance of the K-Means Clustering algorithm using a confusion matrix. It demonstrated how clustering results can be compared with actual labels to assess accuracy and cluster quality. In this experiment, the theoretical approach and the practical approach are the same.