K Means Clustering

For this project we will attempt to use KMeans Clustering to cluster Universities into to two groups, Private and Public. We will use a data frame with 777 observations on the following 18 variables.

- Private A factor with levels No and Yes indicating private or public university
- Apps Number of applications received
- Accept Number of applications accepted
- Enroll Number of new students enrolled
- Top10perc Pct. new students from top 10% of H.S. class
- Top25perc Pct. new students from top 25% of H.S. class
- F.Undergrad Number of fulltime undergraduates
- P.Undergrad Number of parttime undergraduates
- Outstate Out-of-state tuition
- Room.Board Room and board costs
- Books Estimated book costs
- Personal Estimated personal spending
- PhD Pct. of faculty with Ph.D.'s
- Terminal Pct. of faculty with terminal degree
- S.F.Ratio Student/faculty ratio
- perc.alumni Pct. alumni who donate
- Expend Instructional expenditure per student
- Grad.Rate Graduation rate

Import Libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

Get the Data

Read in the College_Data file using read_csv. Figure out how to set the first column as the index.

In [2]:
```
df = pd.read_csv('College_Data',index_col=0)
```

Check the head of the data

In [3]:
```
df.head()
```

Out[3]:

	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outs
Abilene Christian University	Yes	1660	1232	721	23	52	2885	537	
Adelphi University	Yes	2186	1924	512	16	29	2683	1227	1
Adrian College	Yes	1428	1097	336	22	50	1036	99	1
Agnes Scott College	Yes	417	349	137	60	89	510	63	1
Alaska Pacific University	Yes	193	146	55	16	44	249	869	

Check the info() and describe() methods on the data.

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 777 entries, Abilene Christian University to York College of Pennsylva
nia
Data columns (total 18 columns):
Private        777 non-null object
Apps           777 non-null int64
Accept         777 non-null int64
Enroll         777 non-null int64
Top10perc      777 non-null int64
Top25perc      777 non-null int64
F.Undergrad    777 non-null int64
P.Undergrad    777 non-null int64
Outstate       777 non-null int64
Room.Board     777 non-null int64
Books          777 non-null int64
Personal       777 non-null int64
PhD            777 non-null int64
Terminal       777 non-null int64
S.F.Ratio      777 non-null float64
perc.alumni    777 non-null int64
Expend         777 non-null int64
Grad.Rate      777 non-null int64
dtypes: float64(1), int64(16), object(1)
memory usage: 115.3+ KB
```

In [5]:
```
df.describe()
```

Out[5]:

	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Under
count	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.00
mean	3001.638353	2018.804376	779.972973	27.558559	55.796654	3699.907336	855.29
std	3870.201484	2451.113971	929.176190	17.640364	19.804778	4850.420531	1522.43
min	81.000000	72.000000	35.000000	1.000000	9.000000	139.000000	1.00
25%	776.000000	604.000000	242.000000	15.000000	41.000000	992.000000	95.00
50%	1558.000000	1110.000000	434.000000	23.000000	54.000000	1707.000000	353.00
75%	3624.000000	2424.000000	902.000000	35.000000	69.000000	4005.000000	967.00
max	48094.000000	26330.000000	6392.000000	96.000000	100.000000	31643.000000	21836.00

Exploratory Analysis

Create a scatterplot of Grad.Rate versus Room.Board (and their linear fit) where the points are colored by the Private column.

In [6]:
```
sns.set_style('whitegrid')
sns.lmplot('Room.Board','Grad.Rate',data=df, hue='Private',
           palette='coolwarm',size=6,aspect=1,fit_reg=True)
```

```
C:\Users\p2840013\AppData\Local\Continuum\anaconda3_school\lib\site-packages
\seaborn\regression.py:546: UserWarning: The `size` paramter has been renamed
to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```

Out[6]:
```
<seaborn.axisgrid.FacetGrid at 0x2953f10e438>
```

Create a scatterplot of F.Undergrad versus Outstate where the points are colored by the Private column.

The plot shows that these two feature dimensions separate out baed on the type of college

In [7]:
```
sns.set_style('whitegrid')
sns.lmplot('Outstate','F.Undergrad',data=df, hue='Private',
           palette='coolwarm',size=6,aspect=1,fit_reg=False)
```

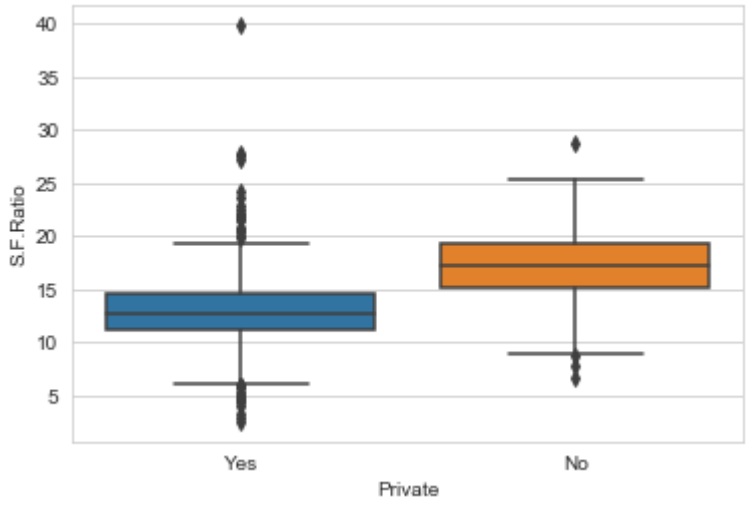
```
C:\Users\p2840013\AppData\Local\Continuum\anaconda3_school\lib\site-packages
\seaborn\regression.py:546: UserWarning: The `size` paramter has been renamed
to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```

Out[7]: <seaborn.axisgrid.FacetGrid at 0x2953f4312b0>

Create a boxplot of student-faculty ratio based on college type

In [8]:
```
sns.boxplot(x='Private',y='S.F.Ratio',data=df)
```

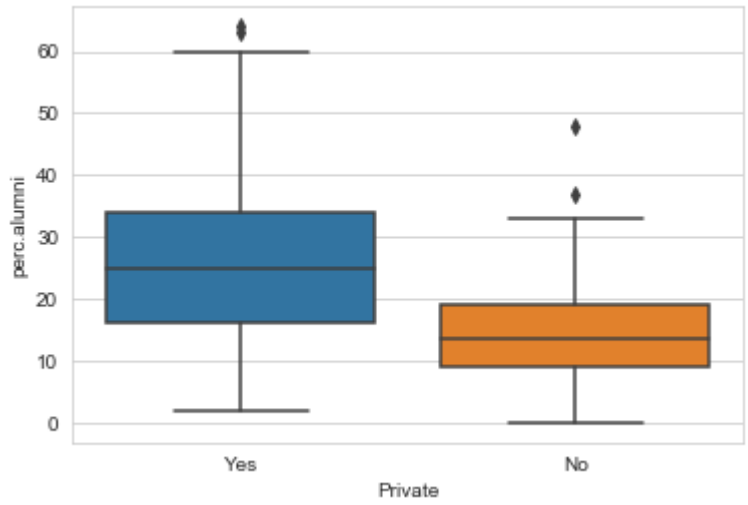
Out[8]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2953f4ecf28>
```



Create a boxplot of percent of alumni who donate based on college type

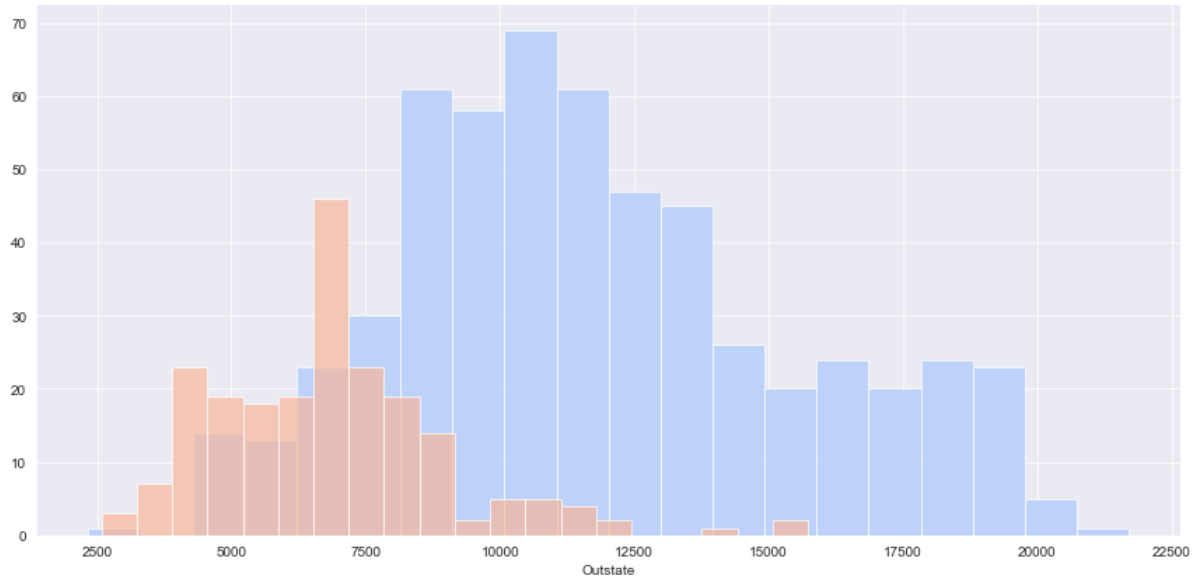
In [9]:
```
sns.boxplot(x='Private',y='perc.alumni',data=df)
```

Out[9]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2953f569ac8>
```



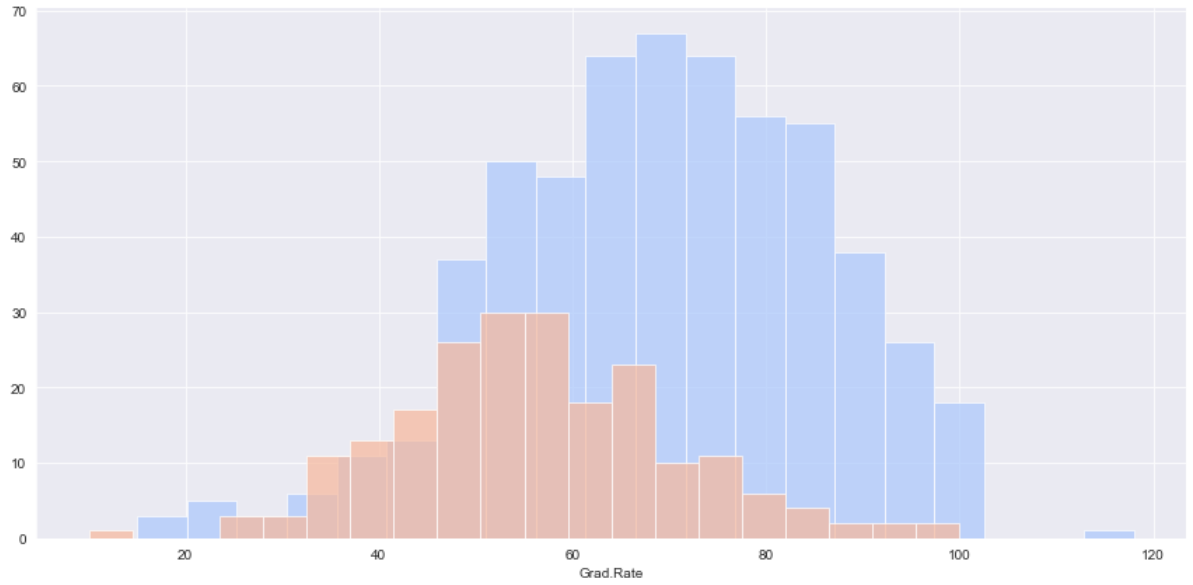
Create a stacked histogram showing Out of State Tuition based on the Private column.

In [10]:
```
sns.set_style('darkgrid')
g = sns.FacetGrid(df,hue="Private",palette='coolwarm',height=6,aspect=2)
g = g.map(plt.hist,'Outstate',bins=20,alpha=0.7)
```



Create a similar histogram for the Grad.Rate column.

In [11]:
```
sns.set_style('darkgrid')
g = sns.FacetGrid(df,hue="Private",palette='coolwarm',height=6,aspect=2)
g = g.map(plt.hist,'Grad.Rate',bins=20,alpha=0.7)
```



There seems to be a private school with a graduation rate of higher than 100%

In [12]:
```
df[df['Grad.Rate'] > 100]
```

Out[12]:

	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Out
Cazenovia College	Yes	3847	3433	527	9	35	1010	12	

Set that school's graduation rate to 100 so it makes sense. You may get a warning not an error) when doing this operation, so use dataframe operations or just re-do the histogram visualization to make sure it actually went through.

In [13]:
```
df['Grad.Rate']['Cazenovia College'] = 100
```

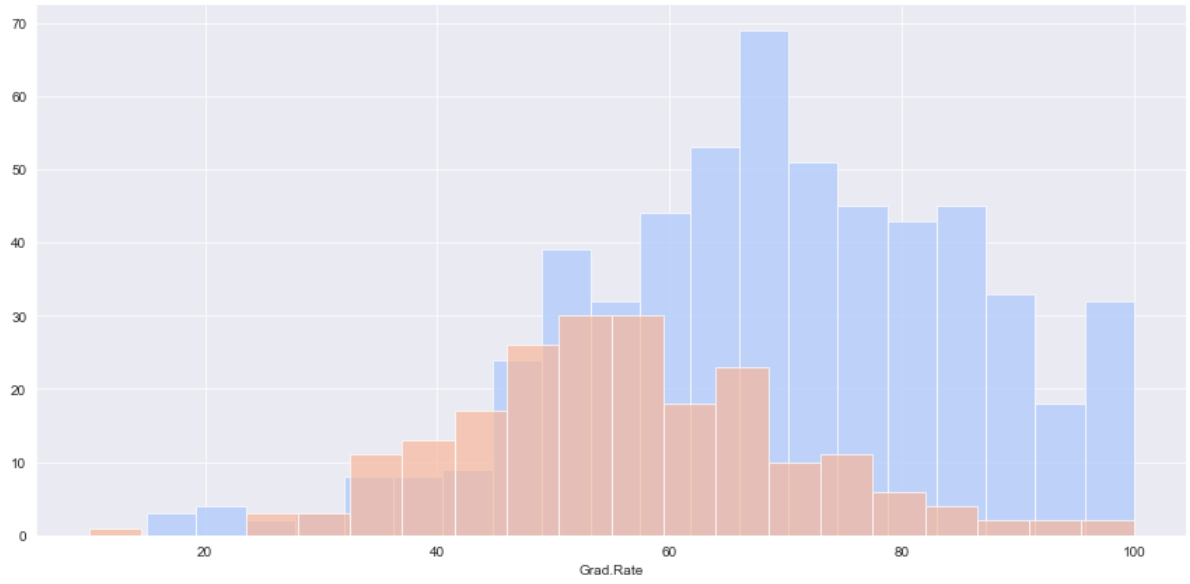
```
C:\Users\p2840013\AppData\Local\Continuum\anaconda3_school\lib\site-packages
\ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy
  """Entry point for launching an IPython kernel.
```

In [14]:
```
df[df['Grad.Rate'] > 100]
```

Out[14]:

	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Roc

In [15]:
```
sns.set_style('darkgrid')
g = sns.FacetGrid(df,hue="Private",palette='coolwarm',height=6,aspect=2)
g = g.map(plt.hist,'Grad.Rate',bins=20,alpha=0.7)
```



K Means Cluster Creation

Import KMeans from SciKit Learn.

In [16]:
```
from sklearn.cluster import KMeans
```

Create an instance of a K Means model with 2 clusters.

In [17]:
```
kmeans = KMeans(n_clusters=2,verbose=0,tol=1e-3,max_iter=300,n_init=20)
```

Fit the model to all the data except for the Private label.

In [18]:
```
kmeans.fit(df.drop('Private',axis=1))
```

Out[18]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
    n_clusters=2, n_init=20, n_jobs=None, precompute_distances='auto',
    random_state=None, tol=0.001, verbose=0)
```

What are the cluster center vectors?

In [19]:
```
clus_cent=kmeans.cluster_centers_
clus_cent
```

Out[19]:
```
array([[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]])
```

Now compare these cluster centers (for all dimensions/features) to the known means of labeled data

In [20]:
```
df[df['Private']=='Yes'].describe() # Statistics for private colleges only
```

Out[20]:

	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Under
count	565.000000	565.000000	565.000000	565.000000	565.000000	565.000000	565.00
mean	1977.929204	1305.702655	456.945133	29.330973	56.957522	1872.168142	433.96
std	2443.341319	1369.549478	457.529136	17.851391	19.588360	2110.661773	722.37
min	81.000000	72.000000	35.000000	1.000000	9.000000	139.000000	1.00
25%	619.000000	501.000000	206.000000	17.000000	42.000000	840.000000	63.00
50%	1133.000000	859.000000	328.000000	25.000000	55.000000	1274.000000	207.00
75%	2186.000000	1580.000000	520.000000	36.000000	70.000000	2018.000000	541.00
max	20192.000000	13007.000000	4615.000000	96.000000	100.000000	27378.000000	10221.00

In [21]:
```
df[df['Private']=='No'].describe() # Statistics for public colleges only
```

Out[21]:

	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Under
count	212.000000	212.000000	212.000000	212.000000	212.000000	212.000000	212.00
mean	5729.919811	3919.287736	1640.872642	22.834906	52.702830	8571.004717	1978.18
std	5370.675335	3477.266276	1261.592009	16.180443	20.091058	6467.696087	2321.03
min	233.000000	233.000000	153.000000	1.000000	12.000000	633.000000	9.00
25%	2190.750000	1563.250000	701.750000	12.000000	37.000000	3601.000000	600.00
50%	4307.000000	2929.500000	1337.500000	19.000000	51.000000	6785.500000	1375.00
75%	7722.500000	5264.000000	2243.750000	27.500000	65.000000	12507.000000	2495.25
max	48094.000000	26330.000000	6392.000000	95.000000	100.000000	31643.000000	21836.00

Create a data frame with cluster centers and with column names borrowed from the original data frame

Is it clear from this data frame which label corresponds to private college (0 or 1)?

In [22]:
```
df_desc=pd.DataFrame(df.describe())
feat = list(df_desc.columns)
kmclus = pd.DataFrame(clus_cent,columns=feat)
kmclus
```

Out[22]:

	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	
0	1813.234679	1287.165919	491.044843	25.309417	53.470852	2188.548580	595.458894	1(
1	10363.138889	6550.898148	2569.722222	41.490741	70.203704	13061.935185	2464.861111	1(

What are the cluster labels?

In [23]:
```
kmeans.labels_
```

Out[23]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
       0, 1, 0, 0, 0, 1, 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, 1, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 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, 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, 1, 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, 1, 0,
       1, 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, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       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, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 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, 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, 1, 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, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 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, 1, 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, 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, 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, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 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, 1, 1,
       0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0,
       1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0,
       0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0,
       0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 1, 1, 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])
```

Evaluation

There is no perfect way to evaluate clustering if you don't have the labels, however since this is just an exercise, we do have the labels, so we take advantage of this to evaluate our clusters, keep in mind, you usually won't have this luxury in the real world.

Create a new column for df called 'Cluster', which is a 1 for a Private school, and a 0 for a public school.

In [24]:
```
def converter(cluster):
    if cluster=='Yes':
        return 1
    else:
        return 0
```

In [25]:
```
df1=df # Create a copy of data frame so that original data frame does not get
 'corrupted' with the cluster index
df1['Cluster'] = df['Private'].apply(converter)
```

In [26]:
```
df1.head()
```

Out[26]:

	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outs
Abilene Christian University	Yes	1660	1232	721	23	52	2885	537	
Adelphi University	Yes	2186	1924	512	16	29	2683	1227	1
Adrian College	Yes	1428	1097	336	22	50	1036	99	1
Agnes Scott College	Yes	417	349	137	60	89	510	63	1
Alaska Pacific University	Yes	193	146	55	16	44	249	869	

Create a confusion matrix and classification report to see how well the Kmeans clustering worked without being given any labels.

In [27]:
```
from sklearn.metrics import confusion_matrix,classification_report
print(confusion_matrix(df1['Cluster'],kmeans.labels_))
print(classification_report(df1['Cluster'],kmeans.labels_))
```

```
[[138  74]
 [531  34]]
              precision    recall  f1-score   support

           0       0.21      0.65      0.31       212
           1       0.31      0.06      0.10       565

   micro avg       0.22      0.22      0.22       777
   macro avg       0.26      0.36      0.21       777
weighted avg       0.29      0.22      0.16       777
```

Clustering performance (e.g. distance between centroids)

Create two data frames consisting of only private or public university data

In [28]:
```
df_pvt=df[df['Private']=='Yes']
df_pub=df[df['Private']=='No']
```

Play with parameters such as max_iter and n_init and calculate cluster centroid distances

In [29]:
```
kmeans = KMeans(n_clusters=2,verbose=0,tol=1e-3,max_iter=50,n_init=10)
kmeans.fit(df.drop('Private',axis=1))
clus_cent=kmeans.cluster_centers_
df_desc=pd.DataFrame(df.describe())
feat = list(df_desc.columns)
kmclus = pd.DataFrame(clus_cent,columns=feat)
a=np.array(kmclus.diff().iloc[1])
centroid_diff = pd.DataFrame(a,columns=['K-means cluster centroid-distance'],i
ndex=df_desc.columns)
centroid_diff['Mean of corresponding entity (private)']=np.array(df_pvt.mean
())
centroid_diff['Mean of corresponding entity (public)']=np.array(df_pub.mean())
centroid_diff
```

Out[29]:

	K-means cluster centroid-distance	Mean of corresponding entity (private)	Mean of corresponding entity (public)
Apps	8549.904210	1977.929204	5729.919811
Accept	5263.732229	1305.702655	3919.287736
Enroll	2078.677379	456.945133	1640.872642
Top10perc	16.181324	29.330973	22.834906
Top25perc	16.732852	56.957522	52.702830
F.Undergrad	10873.386605	1872.168142	8571.004717
P.Undergrad	1869.402217	433.966372	1978.188679
Outstate	323.467406	11801.693805	6813.410377
Room.Board	332.107499	4586.143363	3748.240566
Books	53.230900	547.506195	554.377358
Personal	433.867381	1214.440708	1676.981132
PhD	15.955697	71.093805	76.834906
Terminal	13.508221	78.534513	82.816038
S.F.Ratio	-0.071923	12.945487	17.139151
perc.alumni	-3.100814	25.890265	14.358491
Expend	5238.453662	10486.353982	7458.316038
Grad.Rate	2.499917	68.966372	56.042453
Cluster	-0.478907	1.000000	0.000000

QUESTIONS

1. Explain the k-means algorithm in your own words.
2. The results for the original data show means for attributes for the "private" cluster and the "public" cluster - describe the two clusters in your own words. What is similar? What is different?
3. Based on the confusion matrix, how well did the k-means clustering work without being given labels?

1. In my own owrds I use k-means by first guessing where the centroid should be and assign data points to the clusters in my data set. I then calculate the distance for each data point to the closests centroid and repeat until i get the same values as a previous calculation so I know I am done.

1. The Private cluster is attempting to group all the private schools into a cluster based on the data and the public cluster is doing the same with Public schools. The Similarities are in Top10perc, top25perc,books, wheras there are major differences in number of applications received, number of applications accepted, number of new students enrolled, number of fulltime undergraduates, and the number of parttime undergraduates.

1. Since (172/777=.22) and (305/777=.40) this shows a 22% Overall Error Rate and a 40% Accuracy which is decent but not amazing.