KSI: ***Machine Learning & AI Dual certification***

Project build under the guidance of Professor **Nitesh Karmakar**

Project 2: **Public and Private University**

Machine Learning: KMeans Clustering

Name: Mohammed Faraz Hussain
College: Terna Engineering College, Nerul
Roll number: TE-B-24
ID number: TU3F1920077
Email: farazhussain503@gmail.com
Terna Mail: mohammedhussain@ternaengg.ac.in
Phone number: +91 9082905016 / +91 7303272780

Table of Contents

Introduction

K-Means Clustering is an unsupervised machine learning algorithm.

In contrast to traditional supervised machine learning algorithms, K-Means attempts to classify data without having first been trained with labeled data.

Once the algorithm has been run and the groups are defined, any new data can be easily assigned to the most relevant group.

Aim and Objective

We will attempt to use KMeans Clustering to cluster Universities into two groups, Private and Public.

Unlike supervised learning, clustering is considered an unsupervised learning method since we don’t have the ground truth to compare the output of the clustering algorithm to the true labels to evaluate its performance. We only want to try to investigate the structure of the data by grouping the data points into distinct subgroups.

In K-Means, each cluster is associated with a centroid. The main objective of the K-Means algorithm is **to minimize the sum of distances between the points and their respective cluster centroid.**

About Dataset

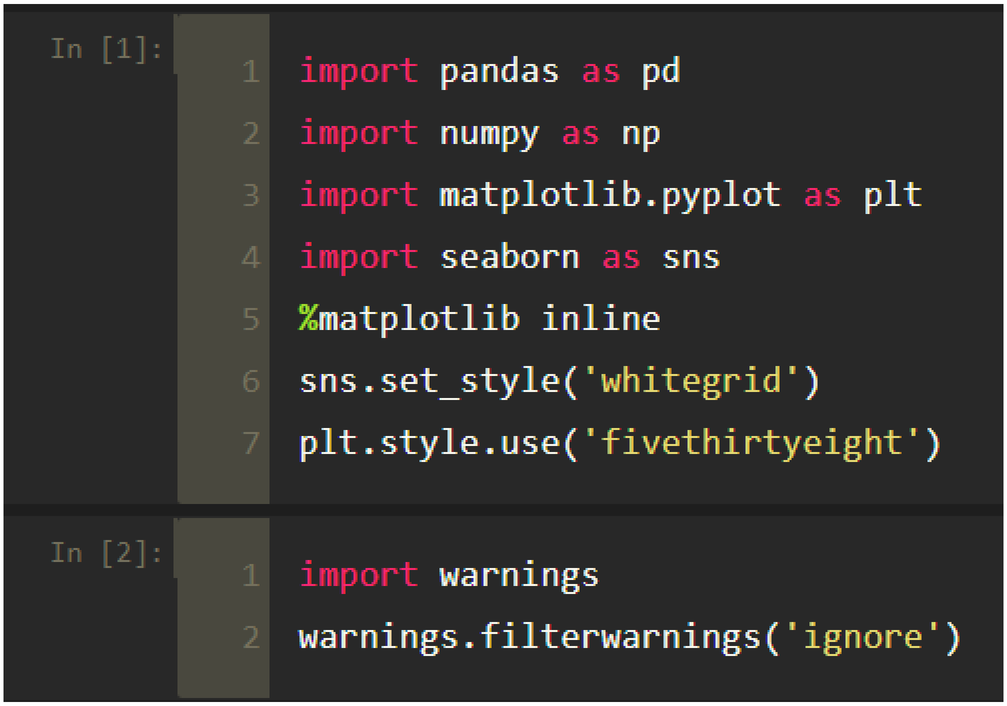
The dataset is a CSV(comma separated value) having 777 rows × 18 columns.

- **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:** Percentage new students from top 10% of high school class
- **Top25perc:** Percentage of new students from top 25% of their high school class
- **F.Undergrad:** Number of full-time undergraduates
- **P.Undergrad:** Number of part-time undergraduates
- **Outstate:** Out-of-state tuition
- **Room.Board:** Room and board costs
- **Books:** Estimated book costs
- **Personal:** Estimated personal spending
- **PhD:** Percentage of faculty with PhDs
- **Terminal:** Percentage of faculty with a terminal degree (PhD/JD/MD/MBA/etc)
- **S.F.Ratio:** Student/faculty ratio
- **perc.alumni:** Percentage alumni who donate
- **Expend:** Instructional expenditure per student
- **Grad.Rate:** Graduation rate

Implementation

1. Importing Libraries
2. Importing Dataset and Read the data (from csv)
3. Identify the dependent and independent variables.
4. Check if the data has missing values or the data is categorical or not.
5. Visualize the data.
6. Applying K means Clustering
7. Model Evaluation

1. Importing Libraries



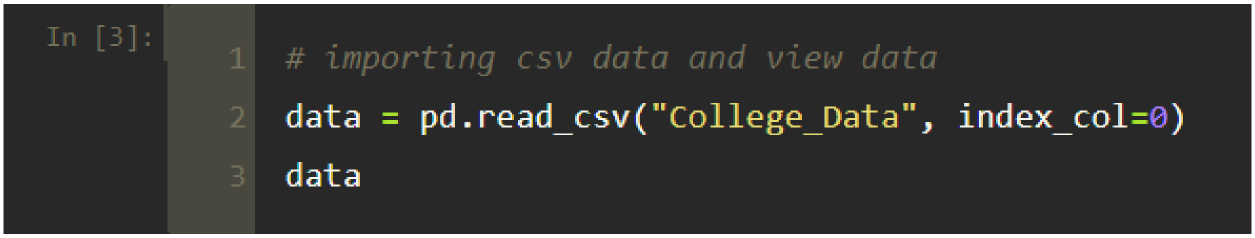
In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
sns.set_style('whitegrid')
plt.style.use('fivethirtyeight')
```

In [2]:

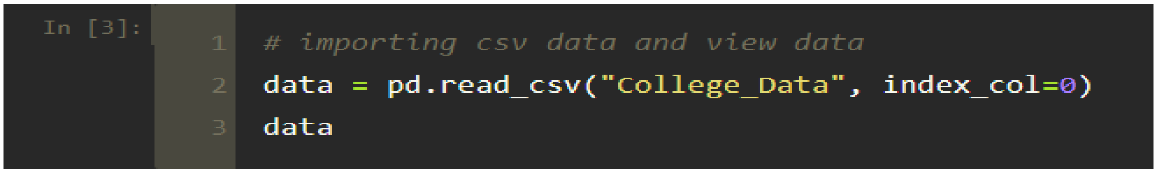
```
import warnings
warnings.filterwarnings('ignore')
```

2. Importing Dataset



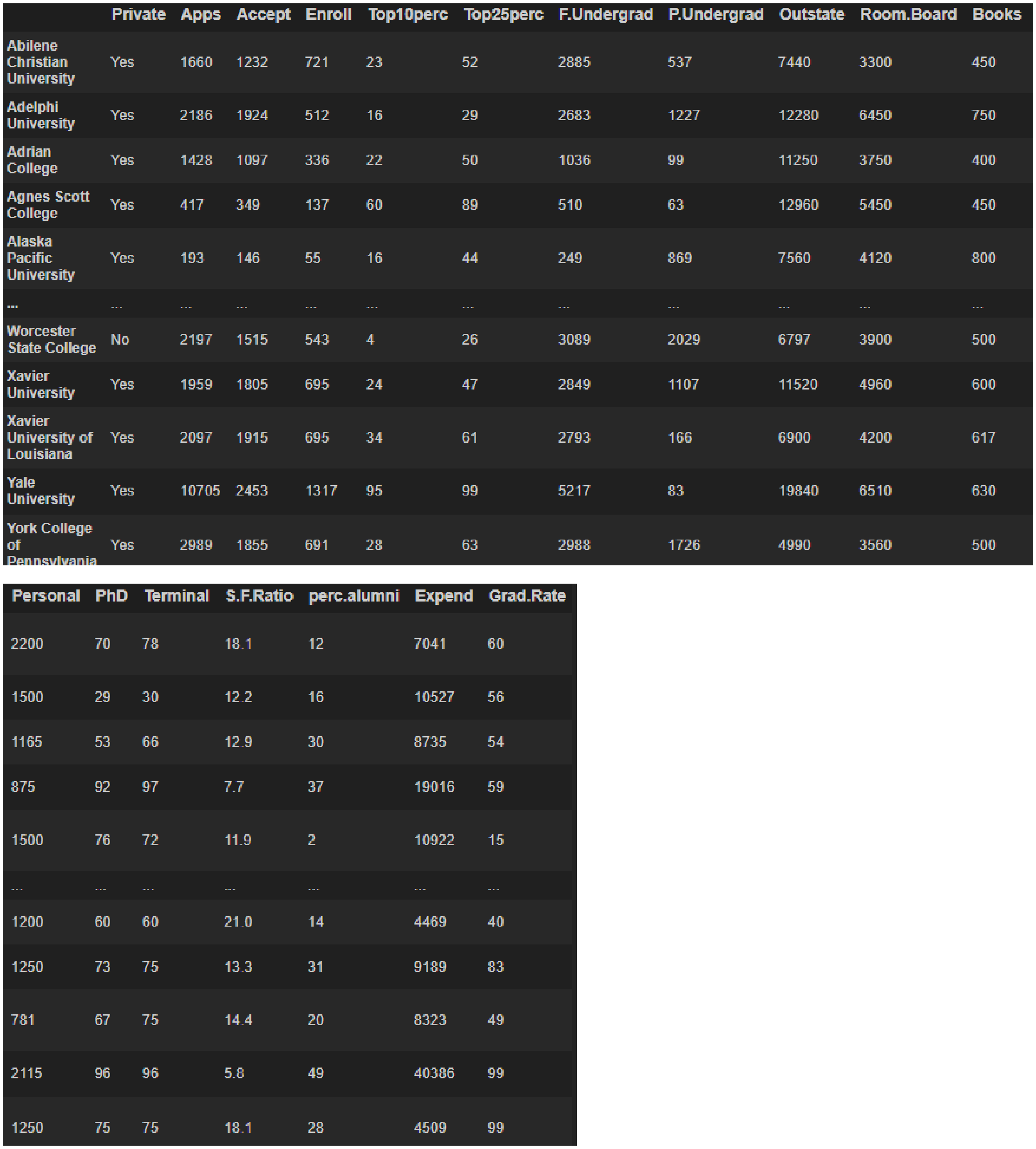
In [3]:

```
# importing csv data and view data
data = pd.read_csv("College_Data", index_col=0)
data
```



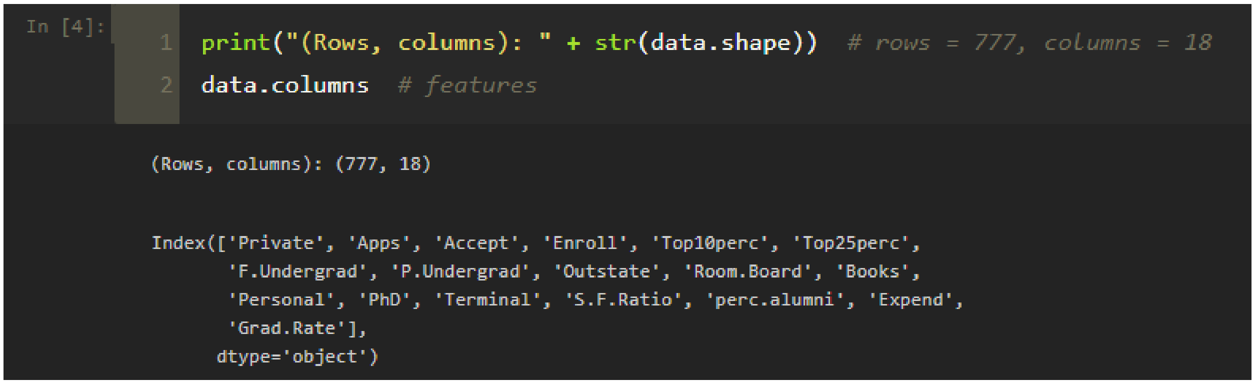
In [3]:

```
# importing csv data and view data
data = pd.read_csv("College_Data", index_col=0)
data
```



	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
Worcester State College	No	2197	1515	543	4	26	3089	2029	6797	3900	500	1200	60	60	21.0	14	4469	40
Xavier University	Yes	1959	1805	695	24	47	2849	1107	11520	4960	600	1250	73	75	13.3	31	9189	83
Xavier University of Louisiana	Yes	2097	1915	695	34	61	2793	166	6900	4200	617	781	67	75	14.4	20	8323	49
Yale University	Yes	10705	2453	1317	95	99	5217	83	19840	6510	630	2115	96	96	5.8	49	40386	99
York College of Pennsylvania	Yes	2989	1855	691	28	63	2988	1726	4990	3560	500	1250	75	75	18.1	28	4509	99

2.1 Checking rows and columns



```
In [4]:
print("(Rows, columns): " + str(data.shape))  # rows = 777, columns = 18
data.columns  # features
```

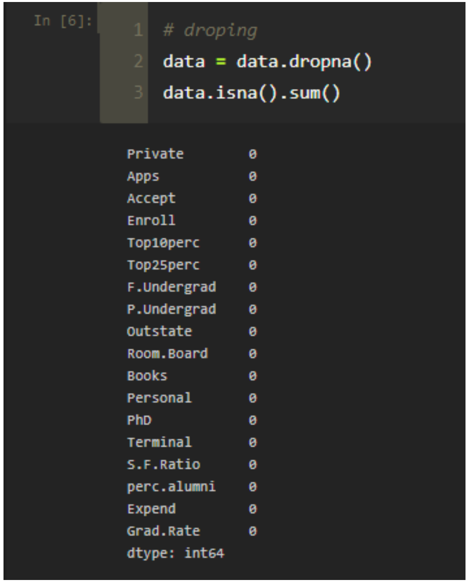
```
(Rows, columns): (777, 18)


Index(['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')
```

3. Identify the dependent and independent variables.

- We don't have any dependent variable, such problems fall into the category of unsupervised learning
- Since we don't have that frame of reference in unsupervised learning, thus the name
- No frame of reference means there is no dependent variable

4. Check if the data has missing values or the data is categorical or not.

- Dataset has no categorical values
- Check for missing/NaN/Null values and drop those columns

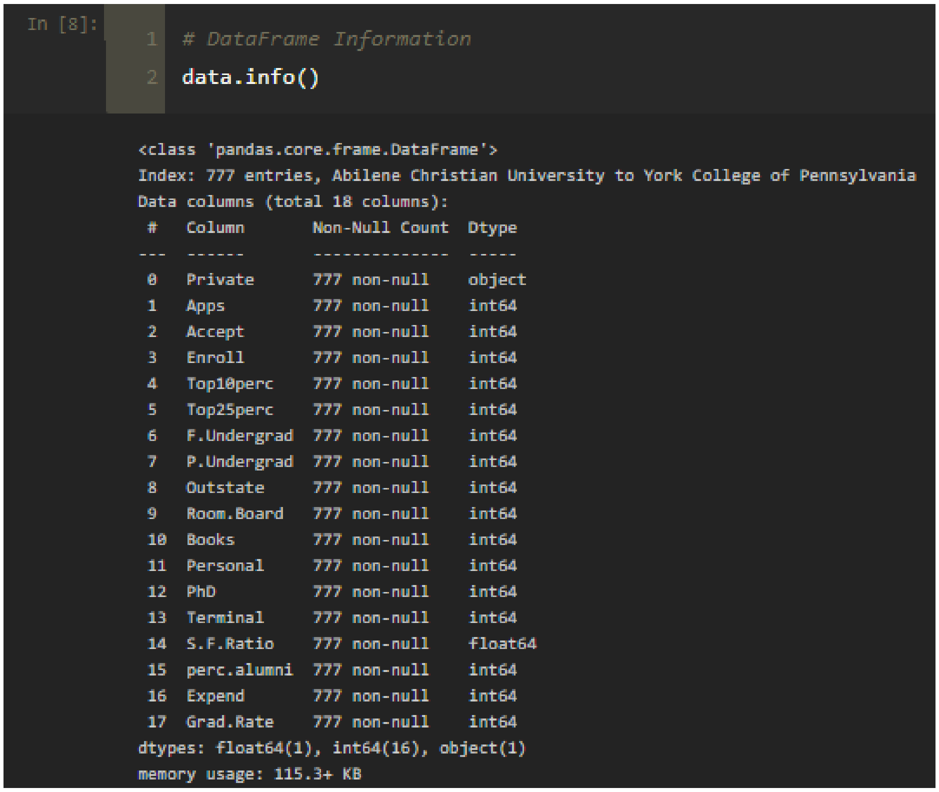


In [6]:

```
# droping
data = data.dropna()
data.isna().sum()
```

```
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
```

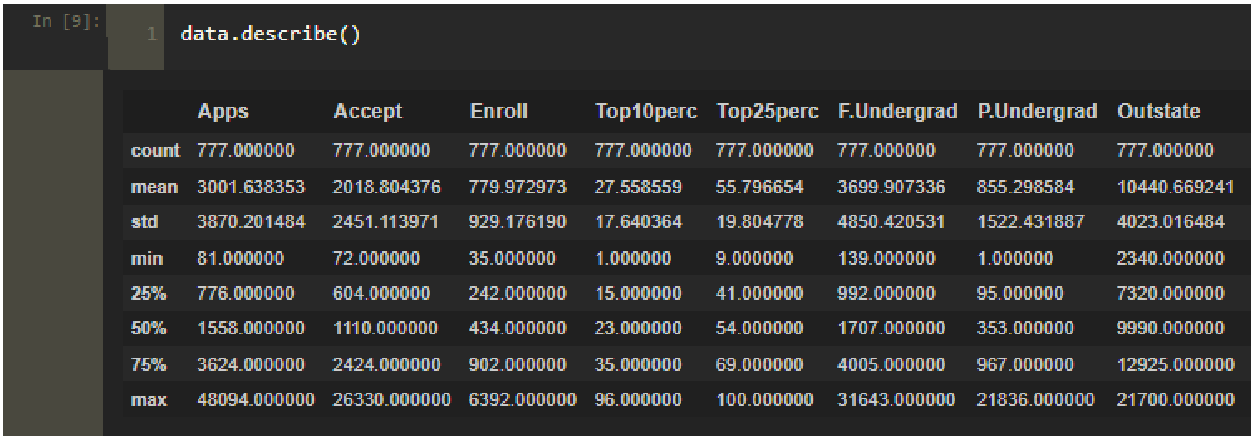
4.1: Concise summary of data



```
In [8]:
# DataFrame Information
data.info()
```

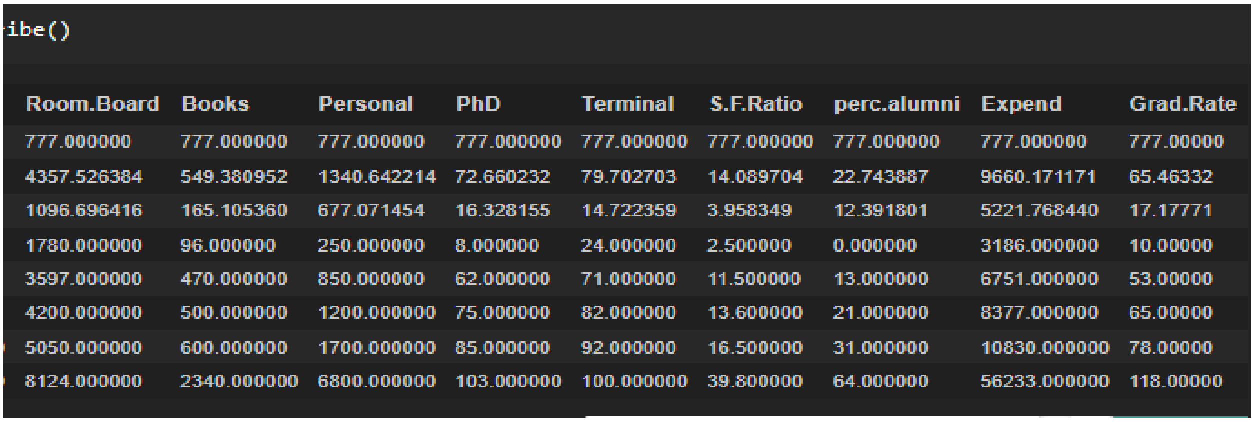
```
<class 'pandas.core.frame.DataFrame'>
Index: 777 entries, Abilene Christian University to York College of Pennsylvania
Data columns (total 18 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Private      777 non-null    object 
 1   Apps         777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

4.2: Statistical measures about the data



```
In [9]:

data.describe()
```

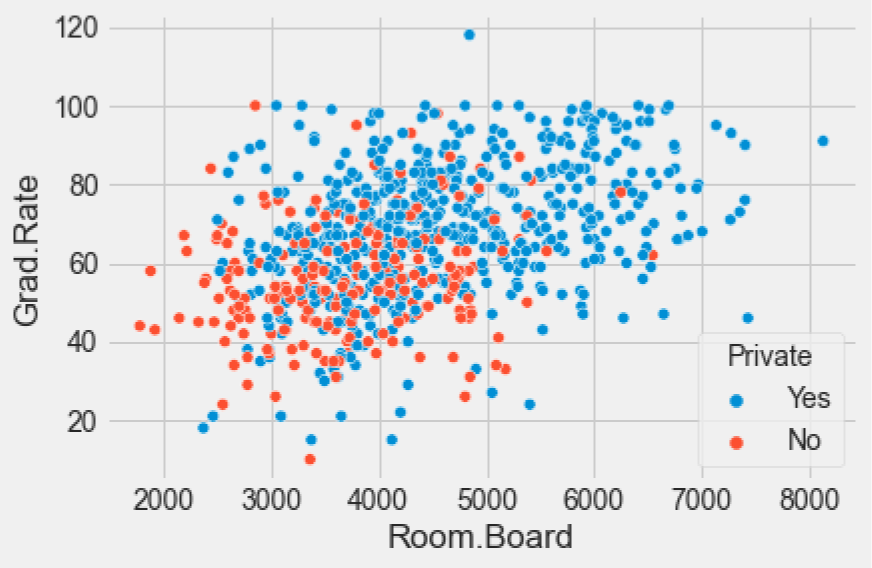
	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate
count	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000
mean	3001.638353	2018.804376	779.972973	27.558559	55.796654	3699.907336	855.298584	10440.669241
std	3870.201484	2451.113971	929.176190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

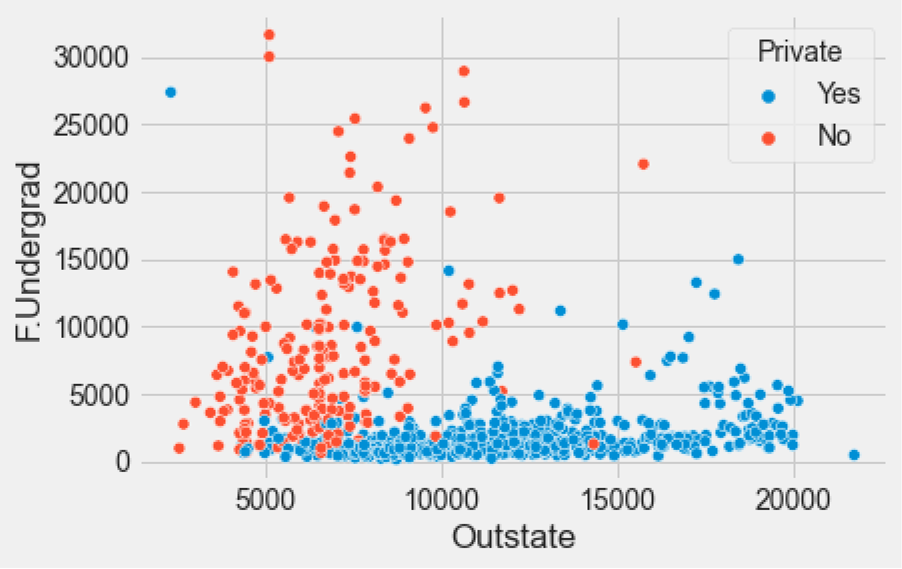


```
ibe()
```

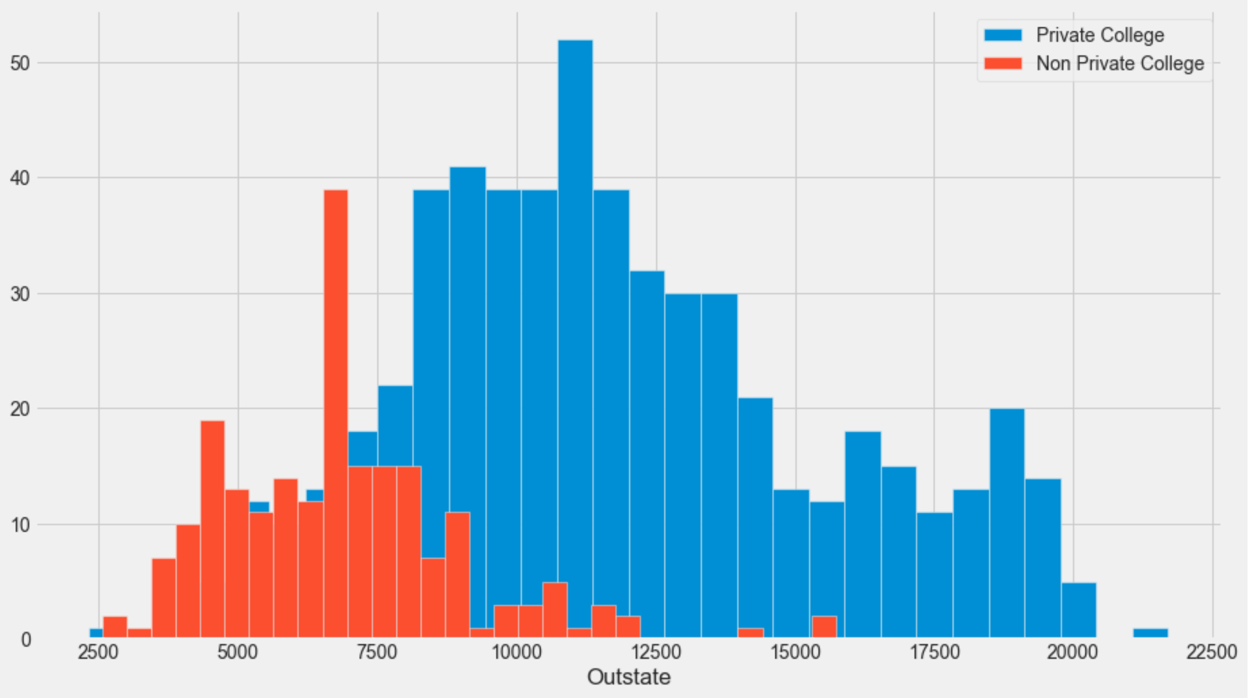
Room.Board	Books	Personal	PhD	Terminal	S.F.Ratio	perc.alumni	Expend	Grad.Rate
777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.00000
4357.526384	549.380952	1340.642214	72.660232	79.702703	14.089704	22.743887	9660.171171	65.46332
1096.696416	165.105360	677.071454	16.328155	14.722359	3.958349	12.391801	5221.768440	17.17771
1780.000000	96.000000	250.000000	8.000000	24.000000	2.500000	0.000000	3186.000000	10.00000
3597.000000	470.000000	850.000000	62.000000	71.000000	11.500000	13.000000	6751.000000	53.00000
4200.000000	500.000000	1200.000000	75.000000	82.000000	13.600000	21.000000	8377.000000	65.00000
5050.000000	600.000000	1700.000000	85.000000	92.000000	16.500000	31.000000	10830.000000	78.00000
8124.000000	2340.000000	6800.000000	103.000000	100.000000	39.800000	64.000000	56233.000000	118.00000

5. Visualizing the Data

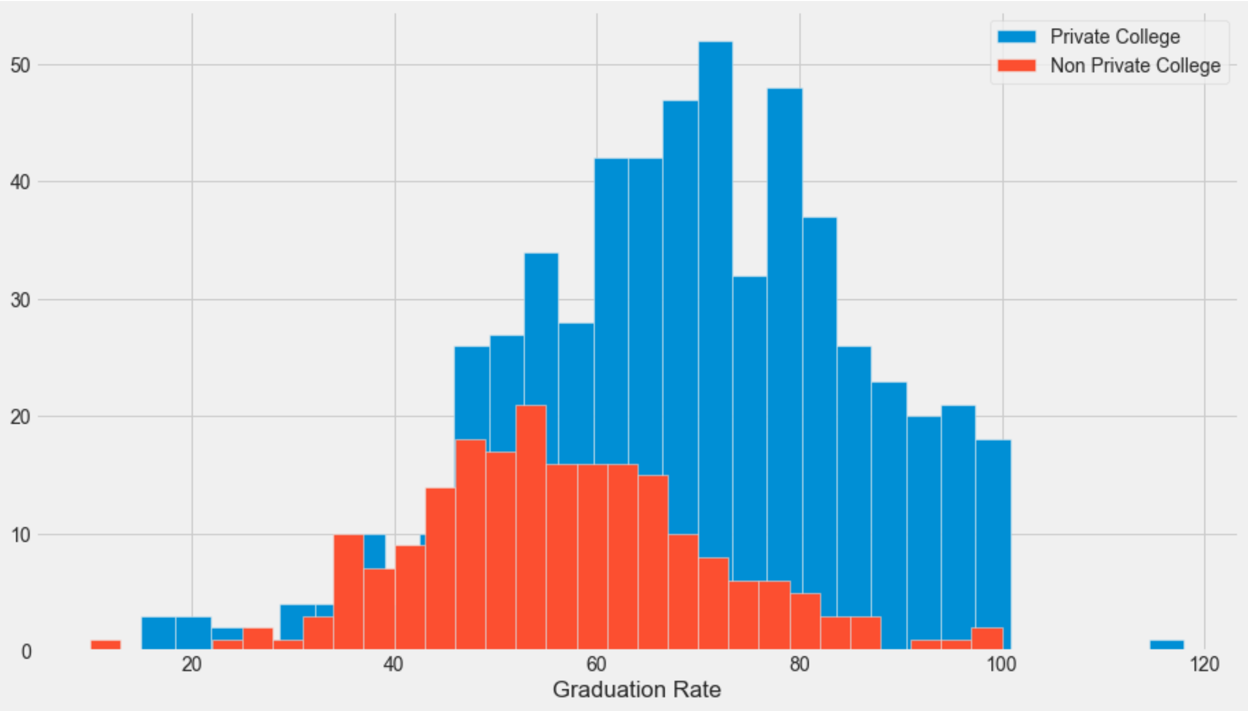




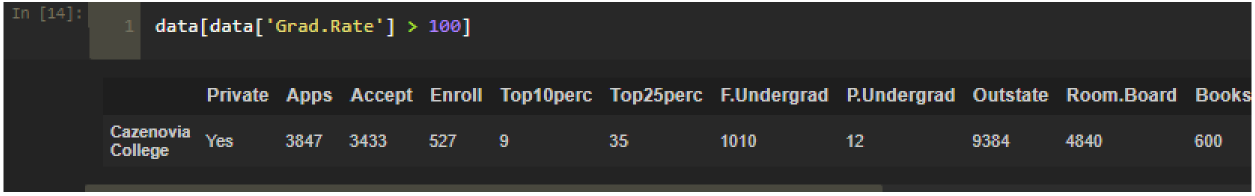
Histogram of Outstate Tuition based on the Private:



Histogram of Grad.Rate based on the Private:



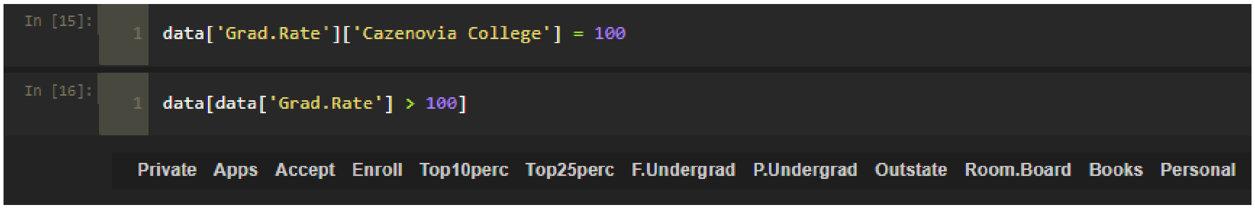
- School with a graduation rate of higher than 100%
- Notice how there seems to be a private school with a graduation rate of higher than 100%



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

	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books
Cazenovia College	Yes	3847	3433	527	9	35	1010	12	9384	4840	600

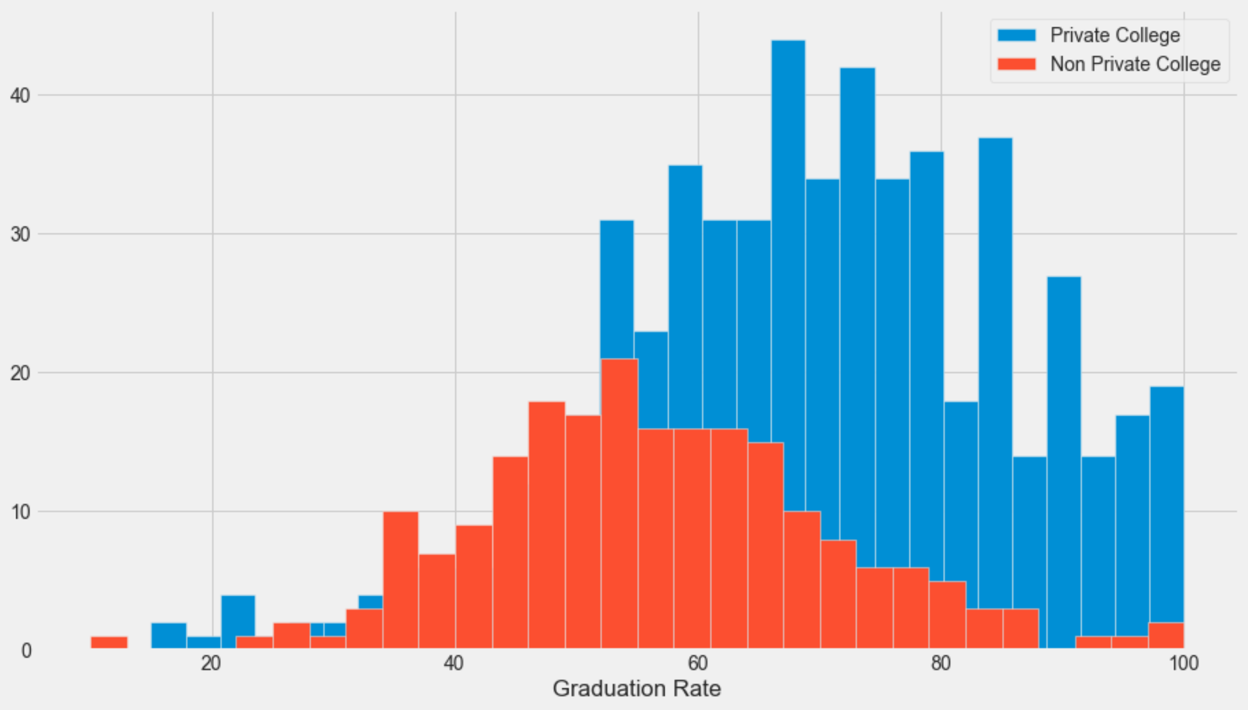
We need to set that school's graduation rate to 100% so it makes sense.



```
In [15]:  data['Grad.Rate']['Cazenovia College'] = 100

In [16]:  data[data['Grad.Rate'] > 100]
```

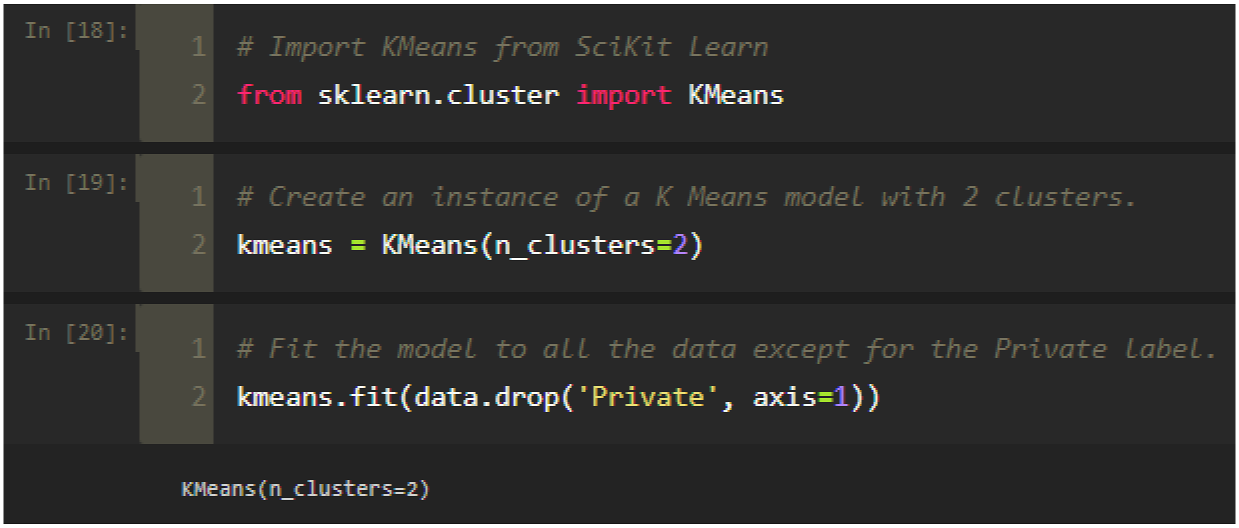
Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal

Now there is no school with graduation rate higher than 100%



- We can see there are no data points that fall outside 100.
- No school with graduation rate higher than 100%

6. Applying K means Clustering

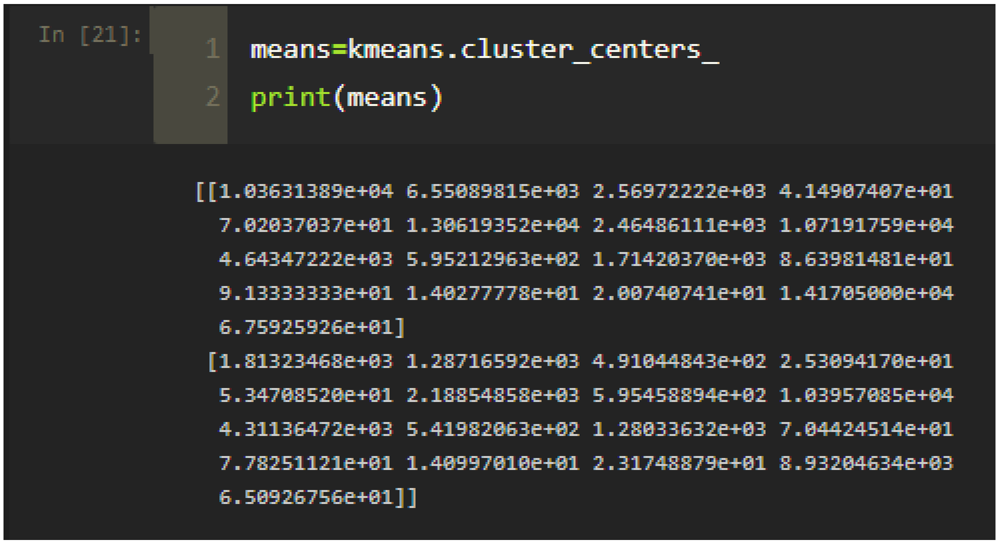


```
In [18]:
# Import KMeans from SciKit Learn
from sklearn.cluster import KMeans
```

```
In [19]:
# Create an instance of a K Means model with 2 clusters.
kmeans = KMeans(n_clusters=2)
```

```
In [20]:
# Fit the model to all the data except for the Private label.
kmeans.fit(data.drop('Private', axis=1))
```

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

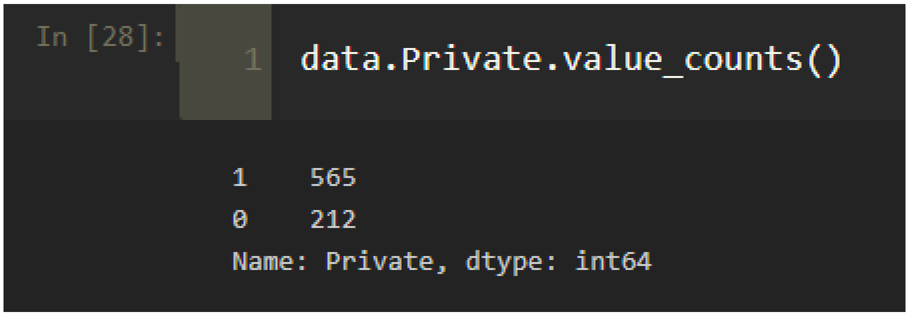


```
In [21]:
means=kmeans.cluster_centers_
print(means)
```

```
[[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]
 [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]]
```

7. Model Evaluation

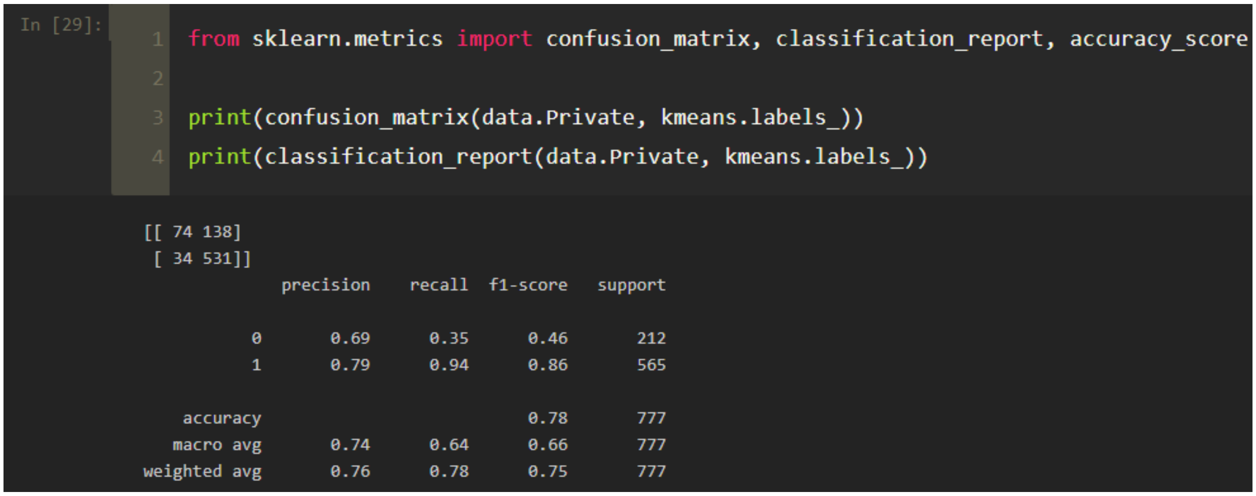
- There is no perfect way to evaluate clustering if we don’t have the labels, however, we do have the labels, so we take advantage of this to evaluate our clusters.
- Create a new column for df called ‘Cluster’, which is a 1 for a Private school, and a 0 for a public school.



```
In [28]:
data.Private.value_counts()
```

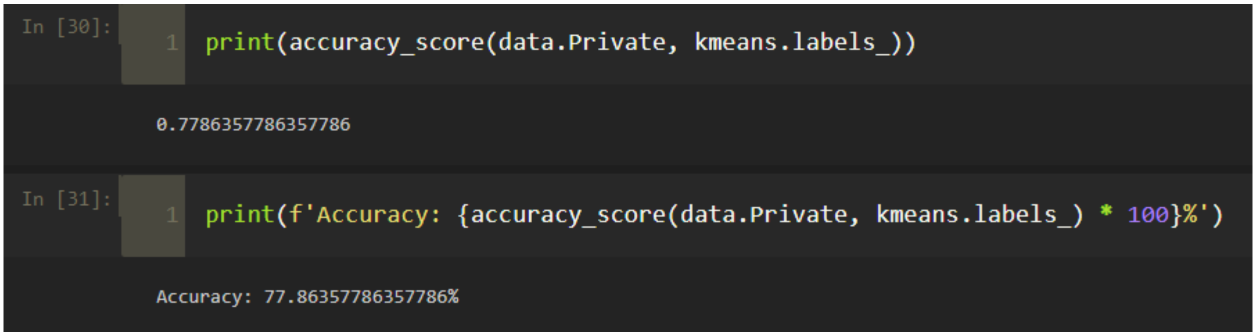
```
1    565
0    212
Name: Private, dtype: int64
```

7.1: Creating a confusion matrix and classification report to see how well the K means clustering worked without being given any labels.



```
In [29]:
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score

print(confusion_matrix(data.Private, kmeans.labels_))
print(classification_report(data.Private, kmeans.labels_))
```

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

           0       0.69      0.35      0.46       212
           1       0.79      0.94      0.86       565

    accuracy                           0.78       777
   macro avg       0.74      0.64      0.66       777
weighted avg       0.76      0.78      0.75       777
```



```
In [30]:
print(accuracy_score(data.Private, kmeans.labels_))
```

```
0.7786357786357786
```

```
In [31]:
print(f'Accuracy: {accuracy_score(data.Private, kmeans.labels_) * 100}%')
```

```
Accuracy: 77.86357786357786%
```

Accuracy: 77.86%

Conclusion

Thus, we got the accuracy to be 77.86% which is not so bad considering the algorithm is purely using the features to cluster the universities into 2 distinct groups.

References

- https://towardsdatascience.com/k-means-clustering-of-university-data-9e8491068778
- https://medium.com/analytics-vidhya/k-means-clustering-43d0136bf005
- https://www.kaggle.com/faressayah/k-means-clustering-private-vs-public-universities