Data Mining Project

KREETHI MISHRA
AP19110010424
CSE-C

INTRODUCTION

For this project I attempted to use KMeans Clustering to cluster Universities into two groups, Private and Public.

Note, I actually have the labels for this data set, but I did NOT use them for the KMeans clustering algorithm, since that is an unsupervised learning algorithm.

This project focuses on investigating the application of KDD to explore and discover patterns within the University dataset.

When using the K Means algorithm under normal circumstances, it is because you don't have labels. In this case I used the labels to try to get an idea of how well the algorithm performed, but you won't usually do this for Kmeans.

DATA SET USED(UNIVERSITY DATASET)

The DataSet used for the project:

I used 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
- Top 10 Perc Pct. new students from top 10% of H.S. class
- Top 25 Perc Pct. new students from top 25% of H.S. 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 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

MECHANISHM USED

K Means Clustering is an unsupervised learning algorithm that tries to cluster data based on their similarity. Unsupervised learning means that there is no outcome to be predicted, and the algorithm just tries to find patterns in the data. In k means clustering, we have to specify the number of clusters we want the data to be grouped into. The algorithm randomly assigns each observation to a cluster, and finds the centroid of each cluster. Then, the algorithm iterates through two steps: Reassign data points to the cluster whose centroid is closest. Calculate new centroid of each cluster. These two steps are repeated till the within cluster variation cannot be reduced any further. The within cluster variation is calculated as the sum of the euclidean distance between the data points and their respective cluster centroids.

EXPLANATION OF THE MECHANISM USED

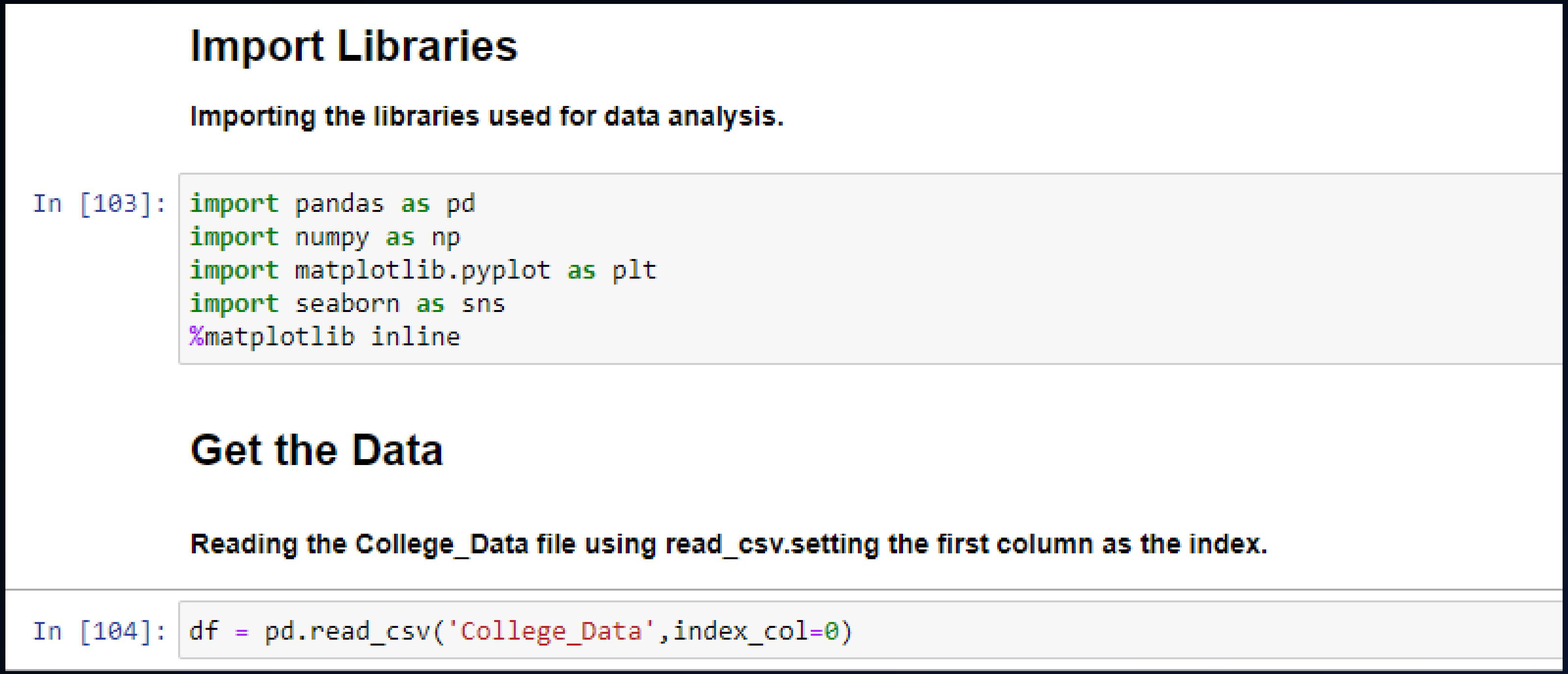
K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of predefined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

Hence each cluster has data points with some commonalities, and it is away from other clusters.

IMPLEMENTATION



Import Libraries

Importing the libraries used for data analysis.

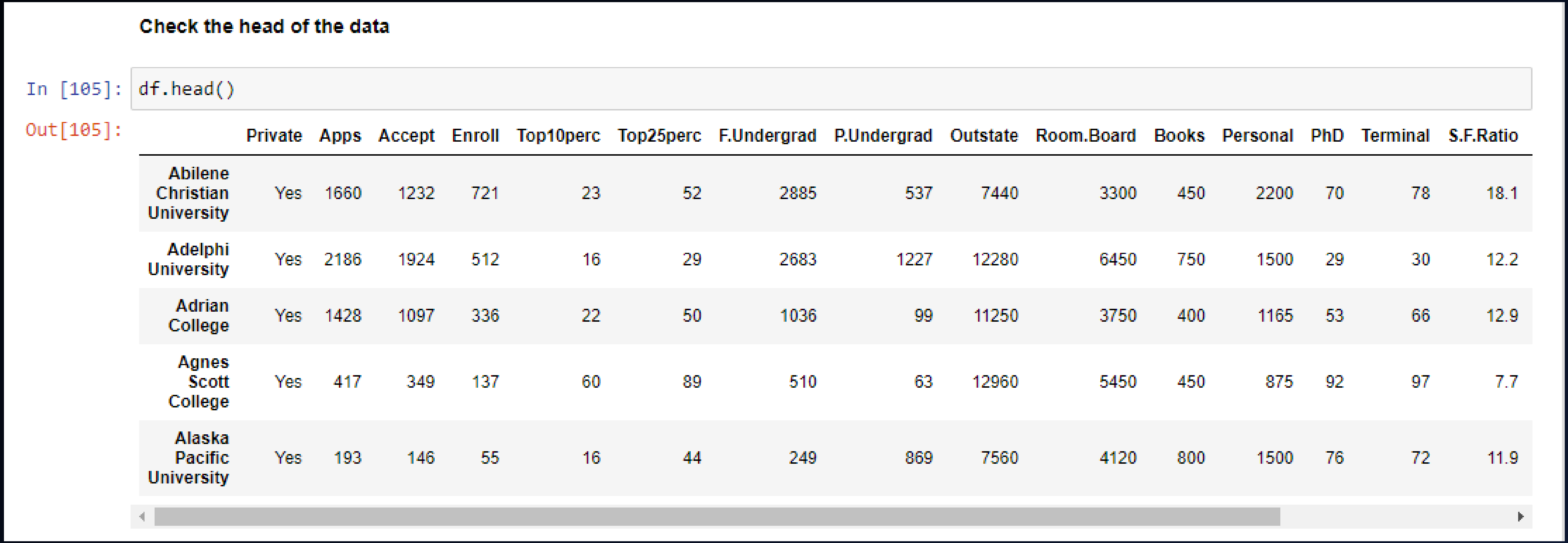
```
In [103]: import pandas as pd
          import numpy as np
          import matplotlib.pyplot as plt
          import seaborn as sns
          %matplotlib inline
```

Get the Data

Reading the College_Data file using read_csv.setting the first column as the index.

```
In [104]: df = pd.read_csv('College_Data',index_col=0)
```

IMPLEMENTATION



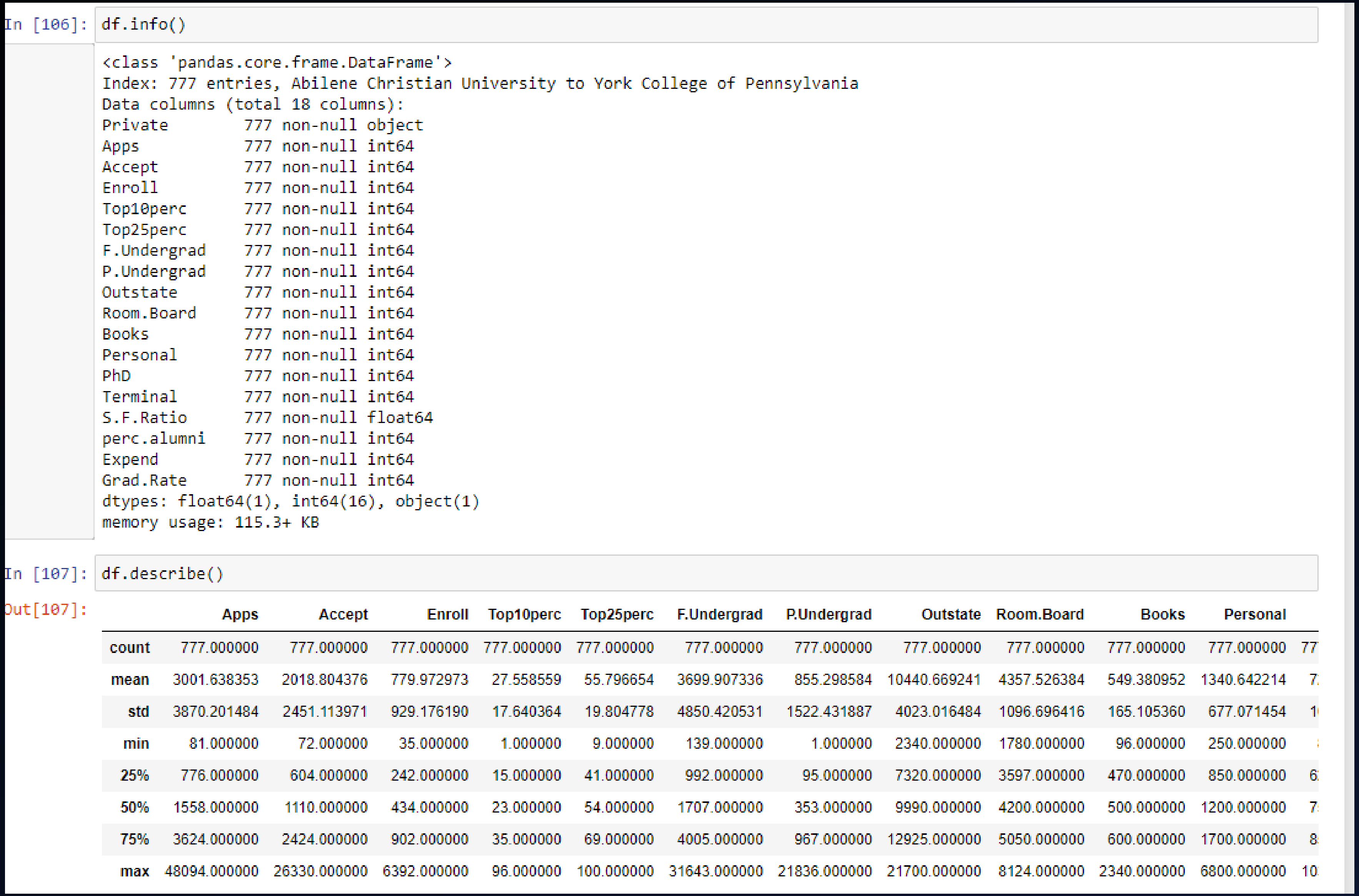
Check the head of the data

```
In [105]: df.head()
```

Out[105]:

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

IMPLEMENTATION



In [106]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 777 entries, Abilene Christian University to York College of Pennsylvania
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
```

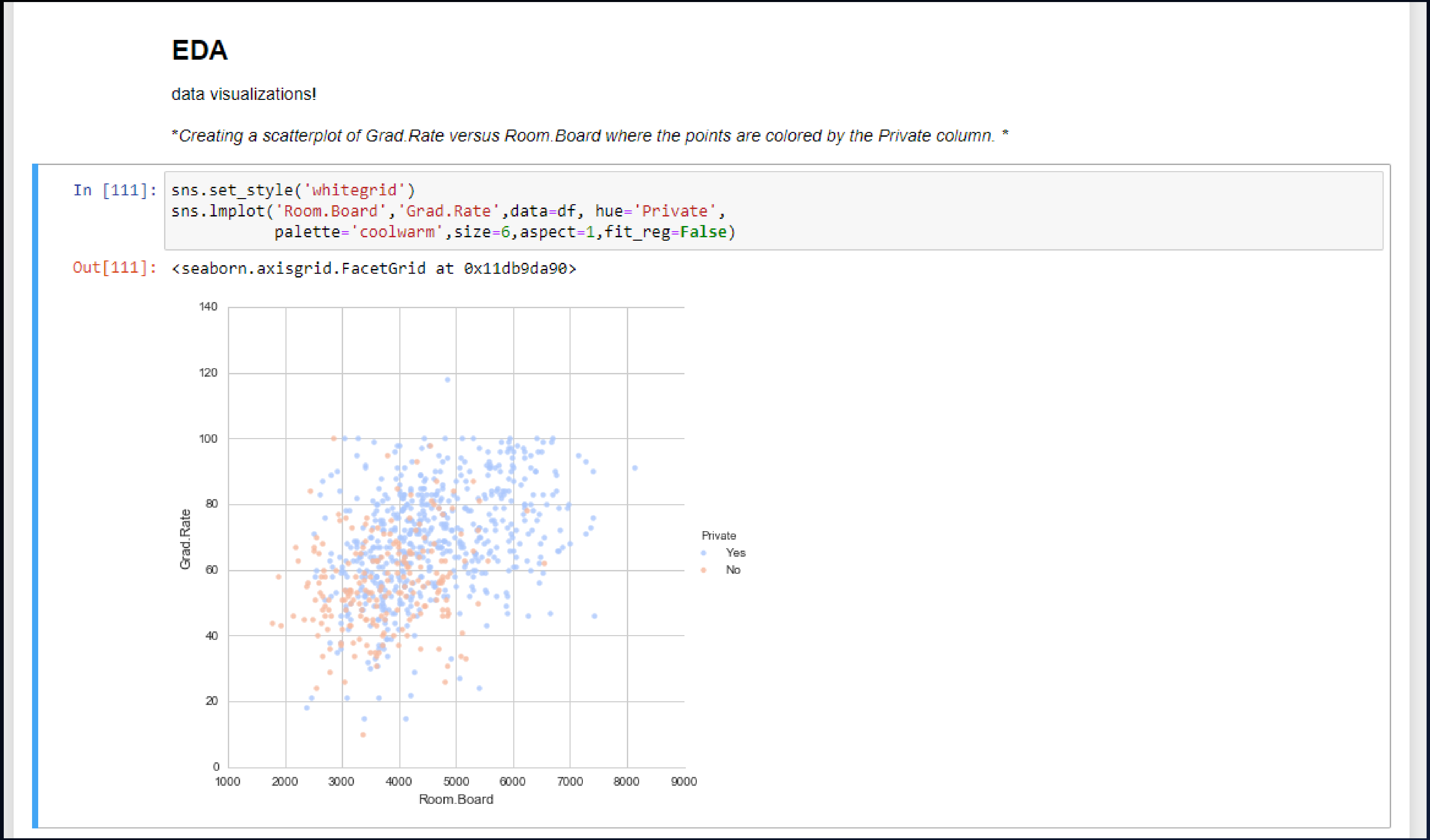
In [107]:

```
df.describe()
```

Out[107]:

	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	
count	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	77
mean	3001.638353	2018.804376	779.972973	27.558559	55.796654	3699.907336	855.298584	10440.669241	4357.526384	549.380952	1340.642214	7
std	3870.201484	2451.113971	929.176190	17.640364	19.804778	4850.420531	1522.431887	4023.016484	1096.696416	165.105360	677.071454	1
min	81.000000	72.000000	35.000000	1.000000	9.000000	139.000000	1.000000	2340.000000	1780.000000	96.000000	250.000000	
25%	776.000000	604.000000	242.000000	15.000000	41.000000	992.000000	95.000000	7320.000000	3597.000000	470.000000	850.000000	6
50%	1558.000000	1110.000000	434.000000	23.000000	54.000000	1707.000000	353.000000	9990.000000	4200.000000	500.000000	1200.000000	7
75%	3624.000000	2424.000000	902.000000	35.000000	69.000000	4005.000000	967.000000	12925.000000	5050.000000	600.000000	1700.000000	8
max	48094.000000	26330.000000	6392.000000	96.000000	100.000000	31643.000000	21836.000000	21700.000000	8124.000000	2340.000000	6800.000000	10

IMPLEMENTATION



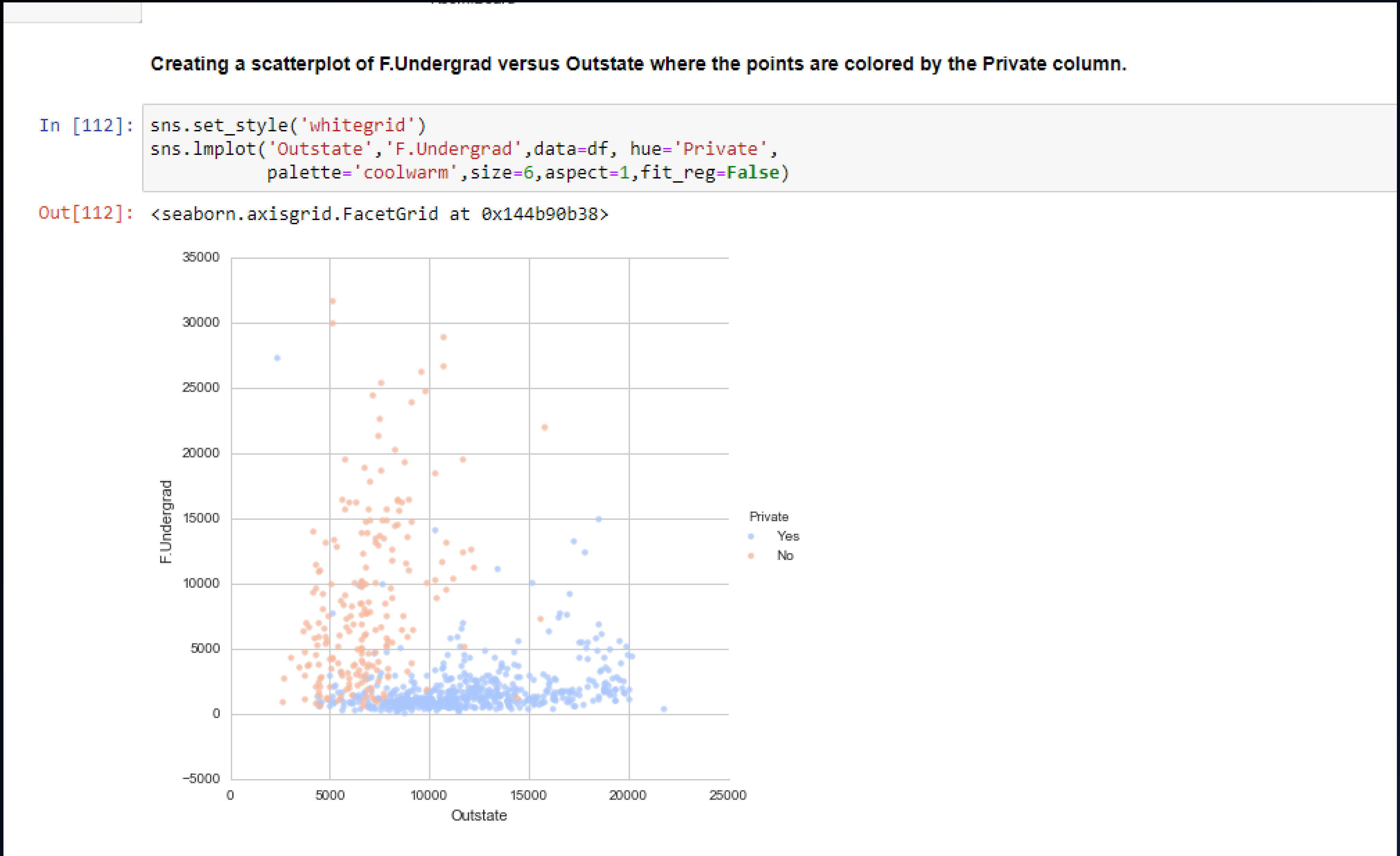
EDA

data visualizations!

*Creating a scatterplot of Grad.Rate versus Room.Board where the points are colored by the Private column. *

```
In [111]: sns.set_style('whitegrid')
          sns.lmplot('Room.Board','Grad.Rate',data=df, hue='Private',
                     palette='coolwarm',size=6,aspect=1,fit_reg=False)

Out[111]: <seaborn.axisgrid.FacetGrid at 0x11db9da90>
```

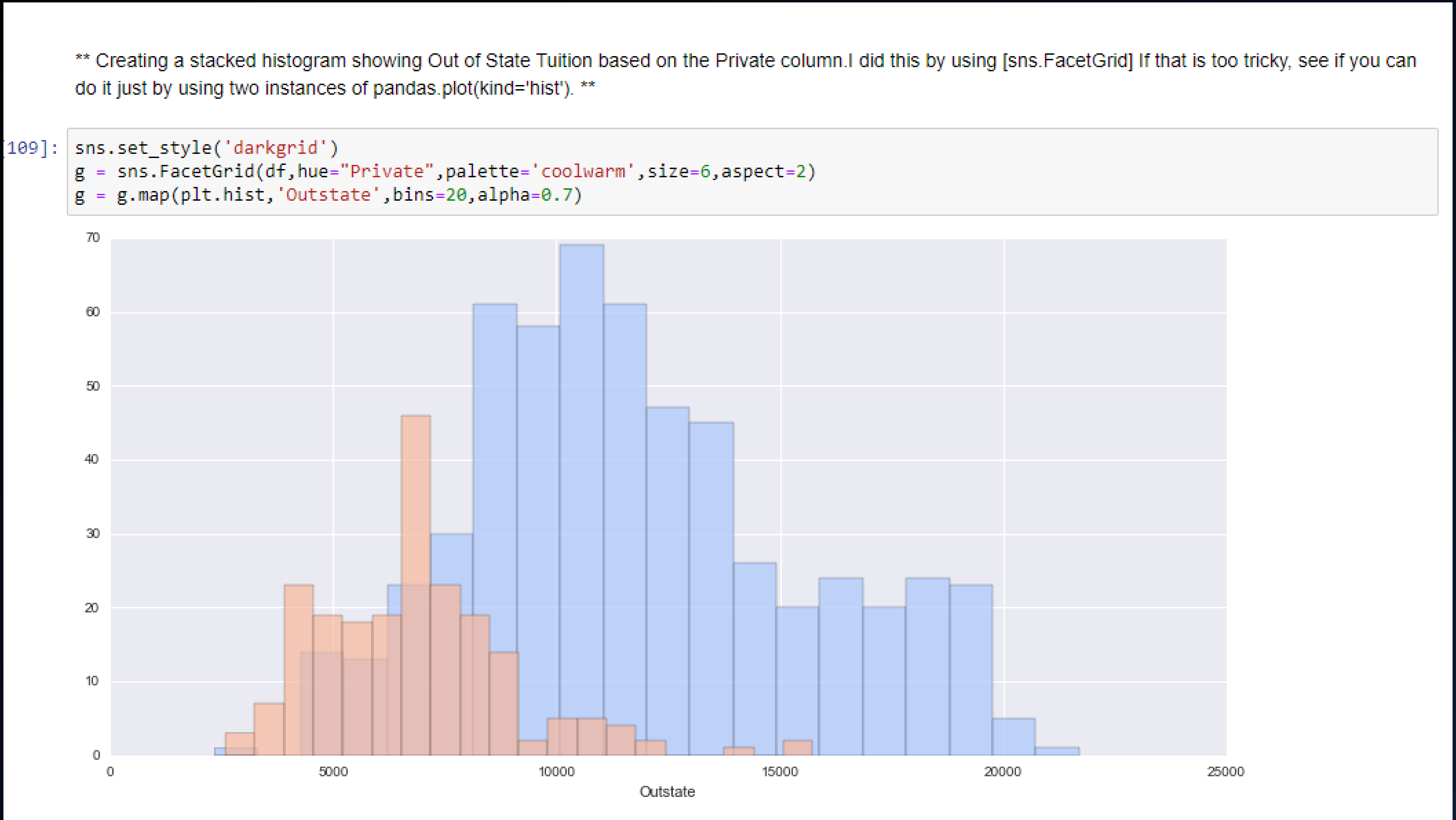
IMPLEMENTATION



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

```
In [112]: sns.set_style('whitegrid')
          sns.lmplot('Outstate','F.Undergrad',data=df, hue='Private',
                     palette='coolwarm',size=6,aspect=1,fit_reg=False)

Out[112]: <seaborn.axisgrid.FacetGrid at 0x144b90b38>
```

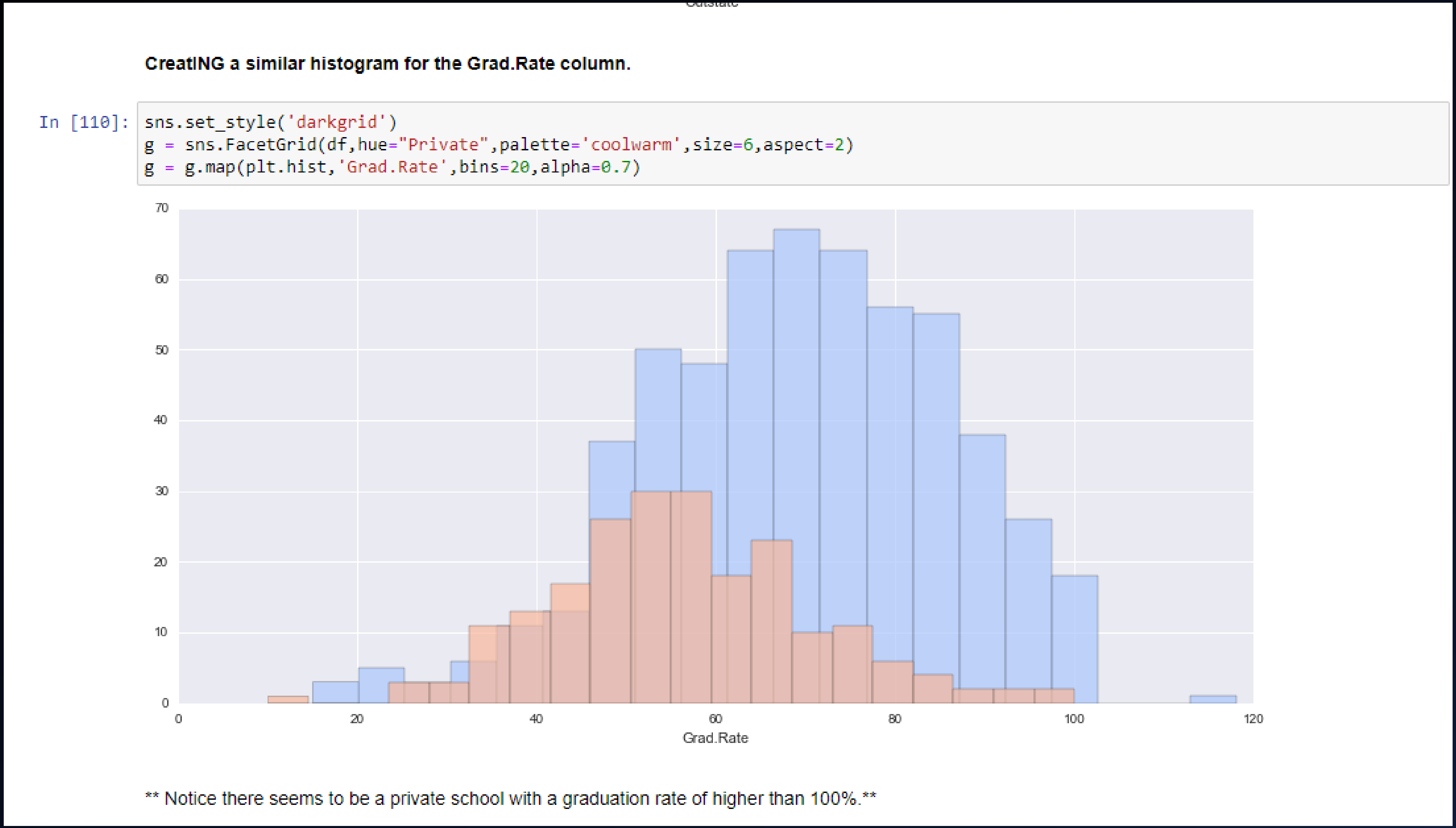
IMPLEMENTATION



** Creating a stacked histogram showing Out of State Tuition based on the Private column.I did this by using [sns.FacetGrid] If that is too tricky, see if you can do it just by using two instances of pandas.plot(kind='hist'). **

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

IMPLEMENTATION

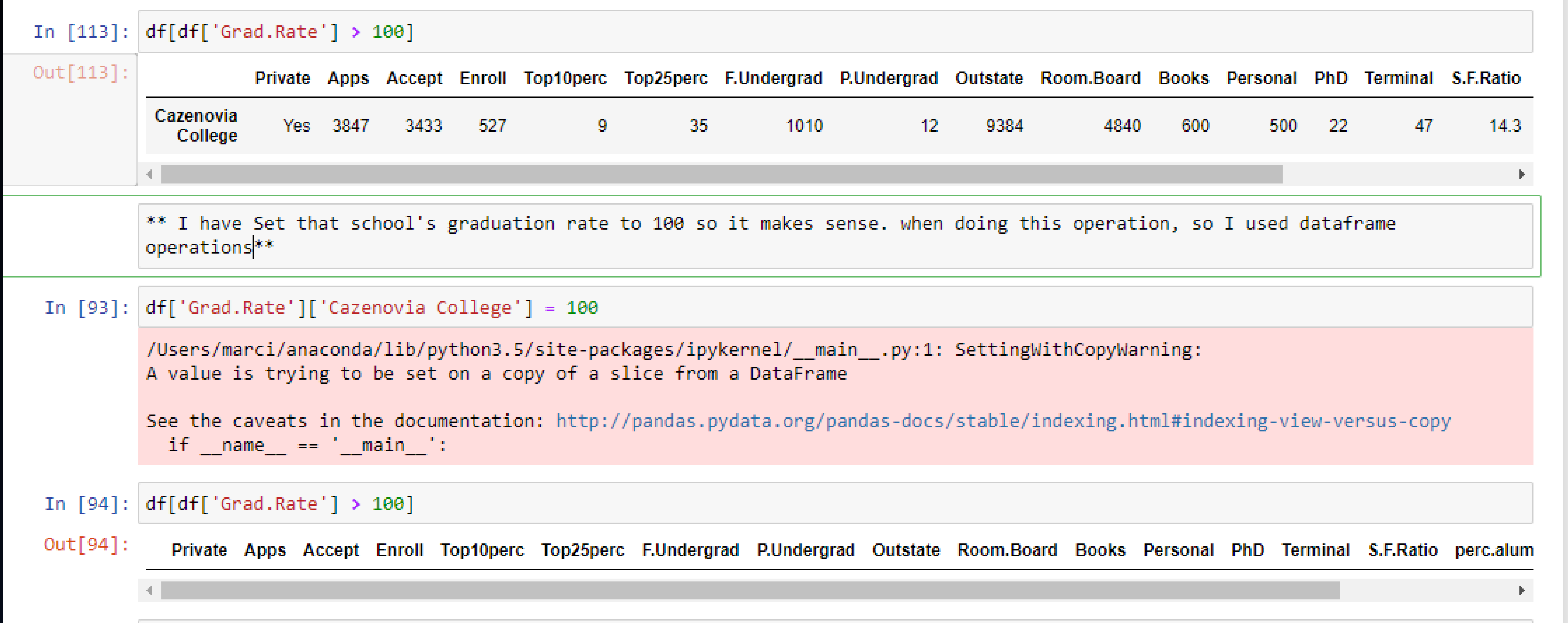


CreatING a similar histogram for the Grad.Rate column.

```
In [110]: sns.set_style('darkgrid')
          g = sns.FacetGrid(df,hue="Private",palette='coolwarm',size=6,aspect=2)
          g = g.map(plt.hist,'Grad.Rate',bins=20,alpha=0.7)
```

** Notice there seems to be a private school with a graduation rate of higher than 100%.**

IMPLEMENTATION



In [113]:
```
df[df['Grad.Rate'] > 100]
```

Out[113]:

	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD	Terminal	S.F.Ratio
Cazenovia College	Yes	3847	3433	527	9	35	1010	12	9384	4840	600	500	22	47	14.3

```
** I have Set that school's graduation rate to 100 so it makes sense. when doing this operation, so I used dataframe
operations**
```

In [93]:
```
df['Grad.Rate']['Cazenovia College'] = 100
```

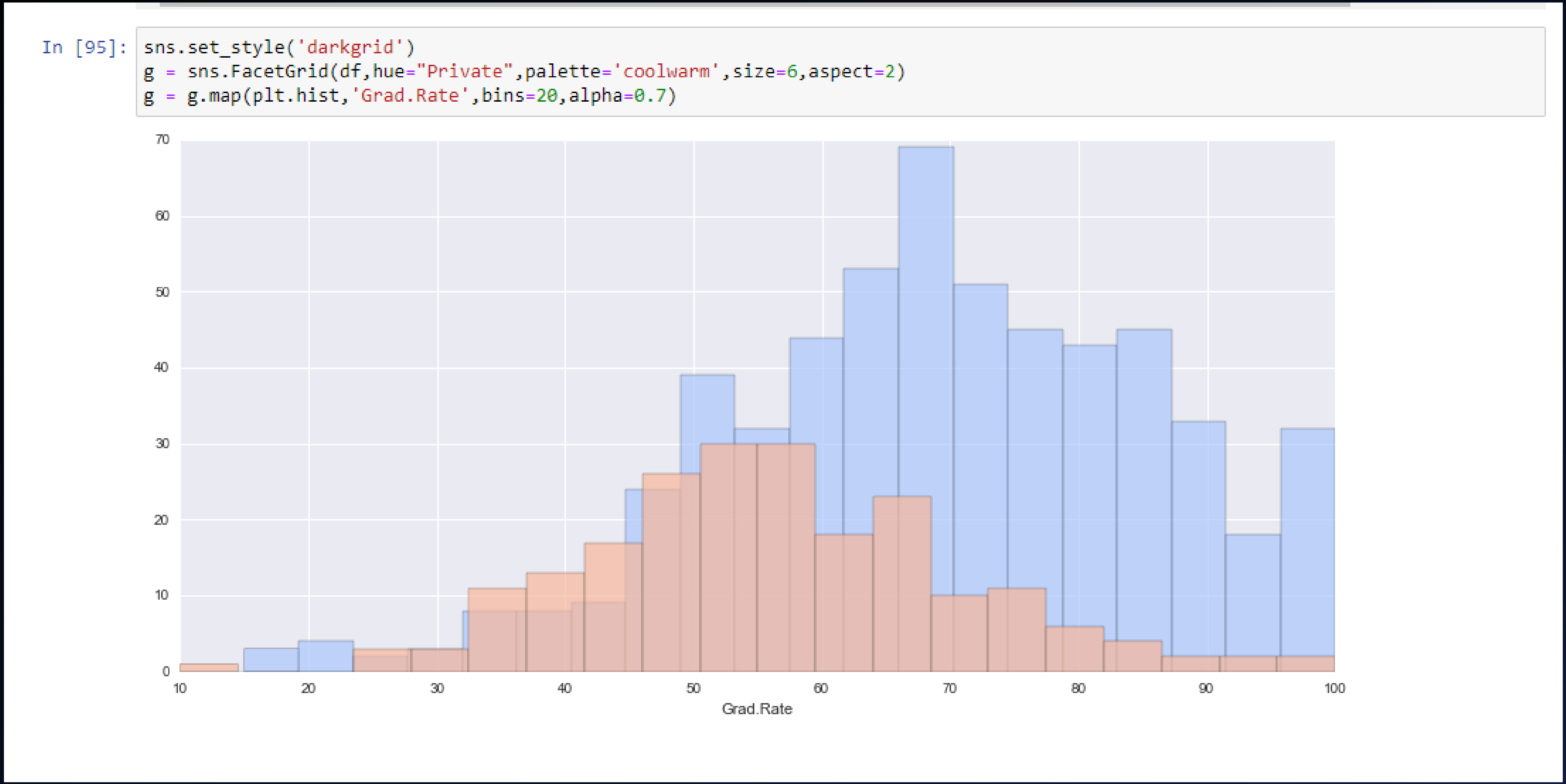
```
/Users/marci/anaconda/lib/python3.5/site-packages/ipykernel/__main__.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  if __name__ == '__main__':
```

In [94]:
```
df[df['Grad.Rate'] > 100]
```

Out[94]:

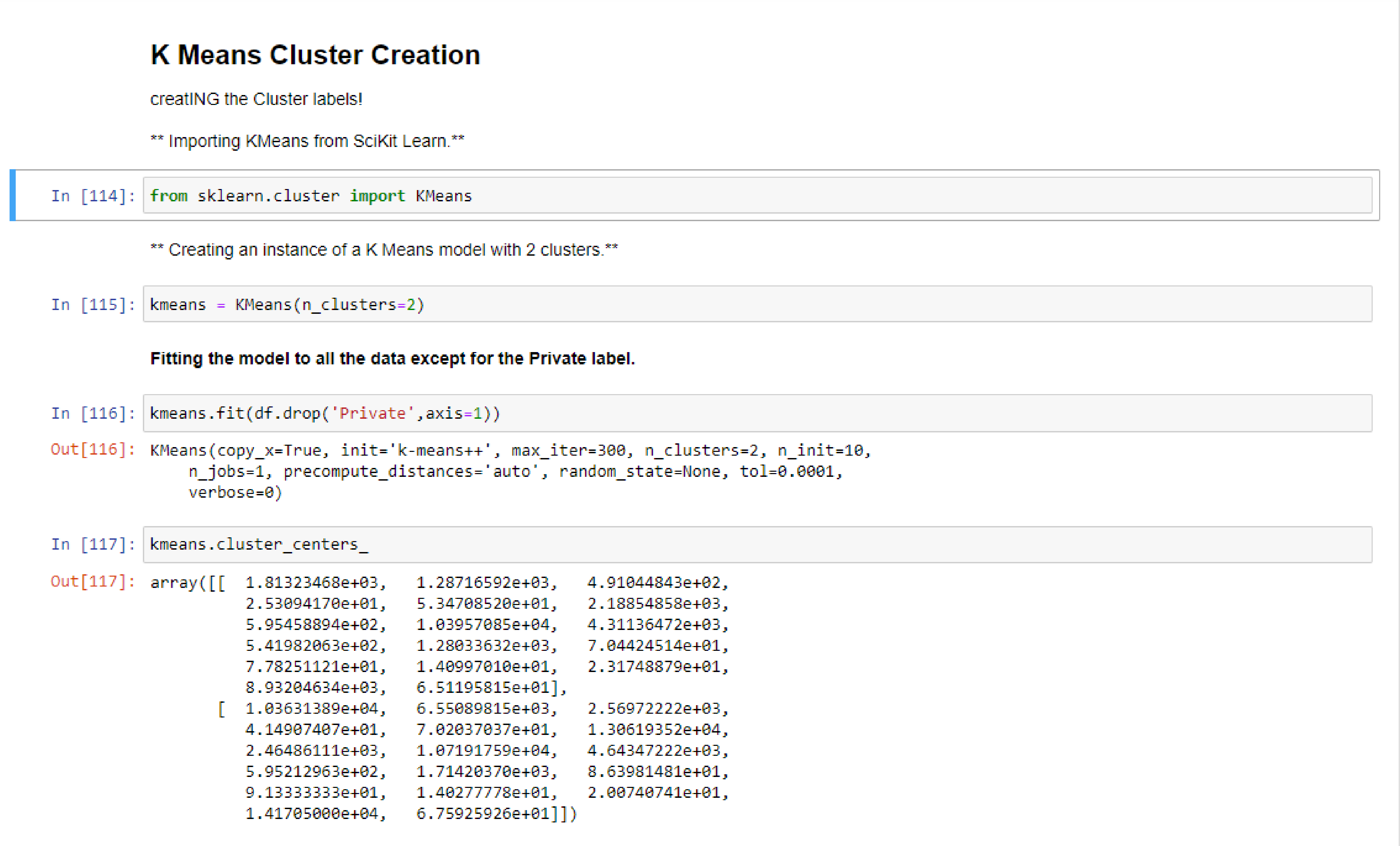
Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD	Terminal	S.F.Ratio	perc.alum

IMPLEMENTATION



```
In [95]: sns.set_style('darkgrid')
         g = sns.FacetGrid(df,hue="Private",palette='coolwarm',size=6,aspect=2)
         g = g.map(plt.hist,'Grad.Rate',bins=20,alpha=0.7)
```

IMPLEMENTATION



K Means Cluster Creation

creatING the Cluster labels!

** Importing KMeans from SciKit Learn.**

In [114]:
```
from sklearn.cluster import KMeans
```

** Creating an instance of a K Means model with 2 clusters.**

In [115]:
```
kmeans = KMeans(n_clusters=2)
```

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

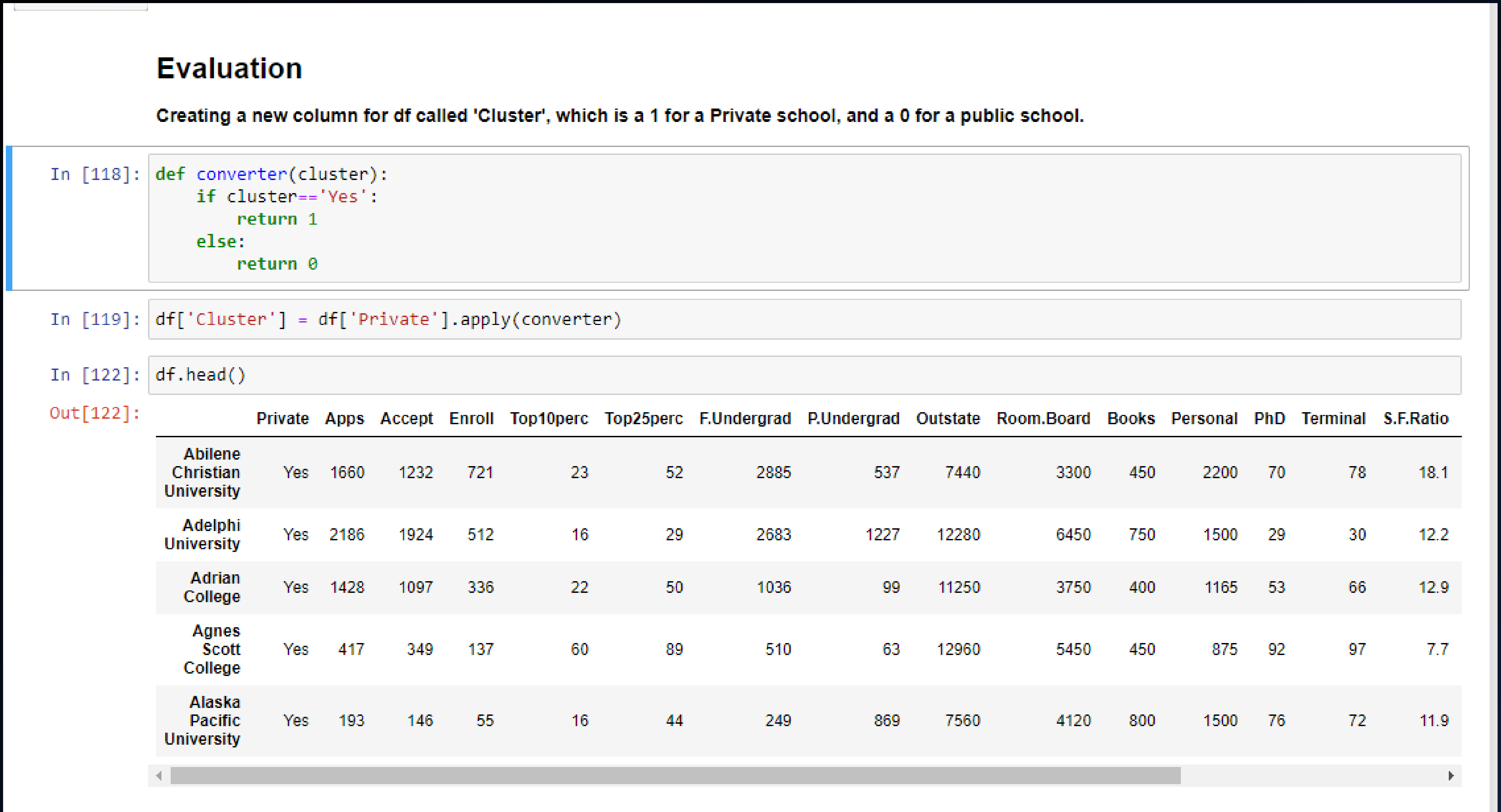
In [116]:
```
kmeans.fit(df.drop('Private',axis=1))
```

Out[116]:
```
KMeans(copy_x=True, init='k-means++', max_iter=300, n_clusters=2, n_init=10,
    n_jobs=1, precompute_distances='auto', random_state=None, tol=0.0001,
    verbose=0)
```

In [117]:
```
kmeans.cluster_centers_
```

Out[117]:
```
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,   6.51195815e+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]])
```

IMPLEMENTATION



Evaluation

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

```
In [118]: def converter(cluster):
              if cluster=='Yes':
                  return 1
              else:
                  return 0
```

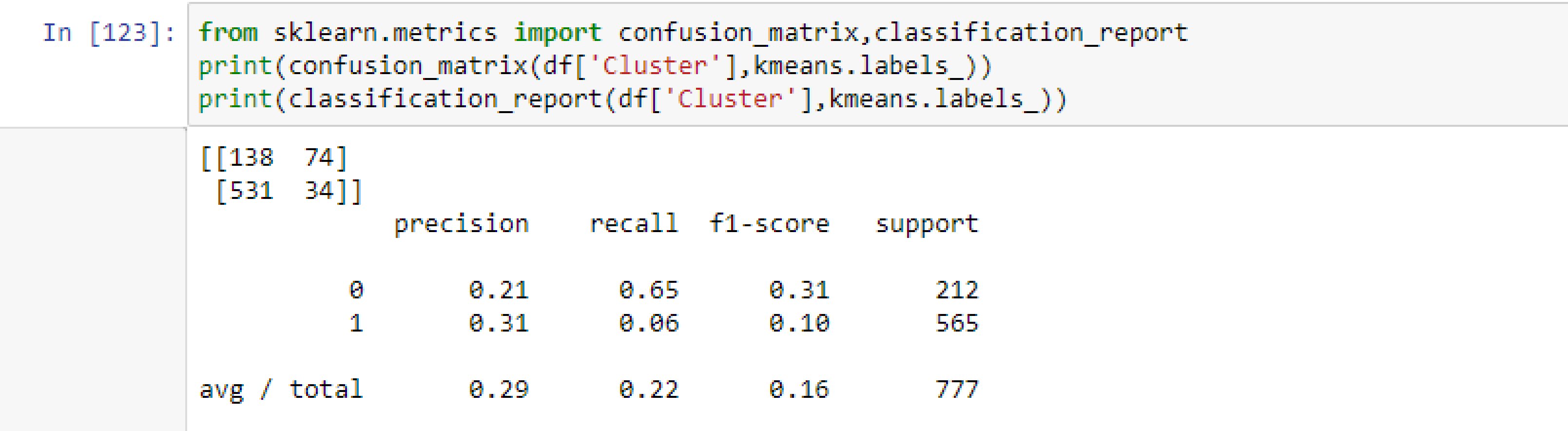
```
In [119]: df['Cluster'] = df['Private'].apply(converter)
```

```
In [122]: df.head()
```

Out[122]:

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

RESULT



```
In [123]: from sklearn.metrics import confusion_matrix,classification_report
          print(confusion_matrix(df['Cluster'],kmeans.labels_))
          print(classification_report(df['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

avg / total       0.29      0.22      0.16       777
```

CONCLUSION

I have created a confusion matrix and classification report to see how well the Kmeans clustering worked by the end I can conclude that it was better considering the algorithm is purely using the features to cluster the universities into 2 distinct groups which happens in our real life

THANK YOU