STUDENTS – PCA ANALYSIS

Prepared by
Muralidhara N

PCA STUDENTS

The given dataset consists of data points of names of various university and college which has number of application received, accepted, and enrolled, percentage of new students from top 10% of higher secondary class, percentage of new students from top 25% of higher secondary class, Number of fulltime undergraduates, Number of part-time undergraduate students, Number of students for whom the particular college is out of state tuition, cost of room and board, estimated book costs for a student, estimated personal spending for a student, percentage of faculties with PHD, percentage of faculties with terminal degree, student/faculty ratio, percentage of alumni who donate, The instructional expenditure per student, Graduation Rate.

INFERENCE OF THE DATASET

The shape of the dataset seems to be with 777 rows and 18 columns.

All the columns seems to be integer or float values.

The Names column alone is a categorical value.

We also can see they are no duplicates in the dataset.

The entire dataset does not have missing values or null values.

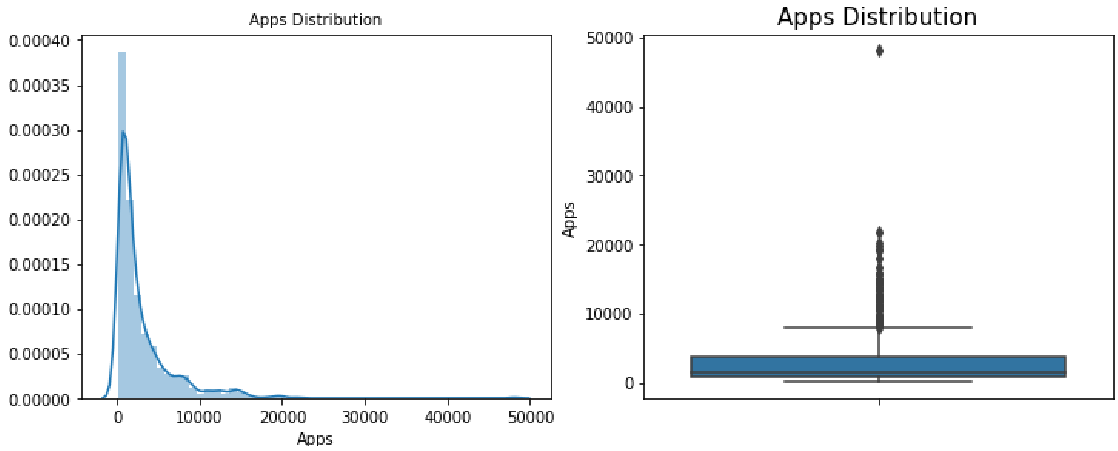
```
Names          0
Apps           0
Accept         0
Enroll         0
Top10perc      0
Top25perc      0
F.Undergrad    0
P.Undergrad    0
Outstate       0
Room.Board     0
Books          0
Personal       0
PhD            0
Terminal       0
S.F.Ratio      0
perc.alumni    0
Expend         0
Grad.Rate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

2.1) Perform Exploratory Data Analysis [both univariate and multivariate analysis to be performed]. The inferences drawn from this should be properly documented.

UNIVARIATE ANALYSIS

Helps us to understand the distribution of data in the dataset. With univariate analysis we can find patterns and we can summarize the data for

APPS

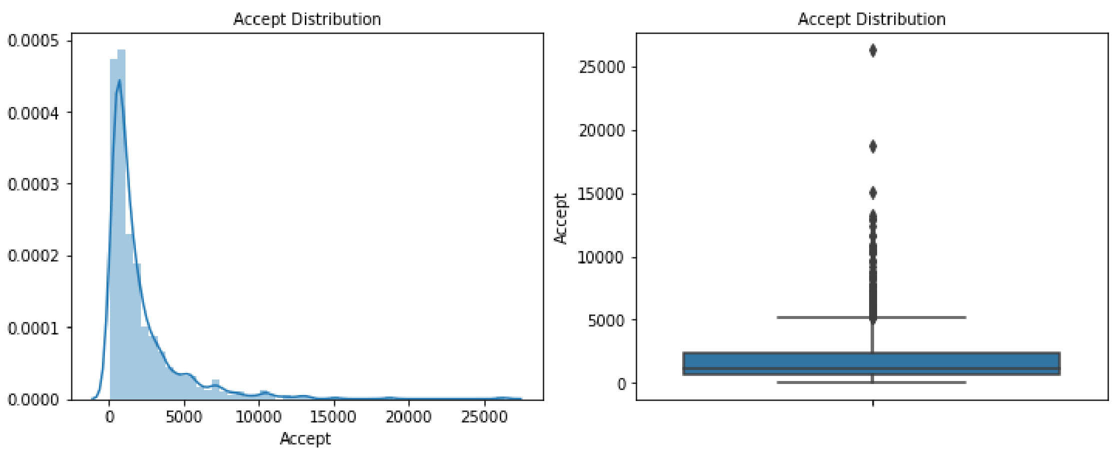


The Box plot of Apps variable seems to have outliers, the distribution of the data is skewed we could also understand that each college or university offers application in the range 3000 to 5000. The max applications seems to be around 50,000.

For univariate analysis of apps we are using box plot and dist plot to find information or patterns in the data.

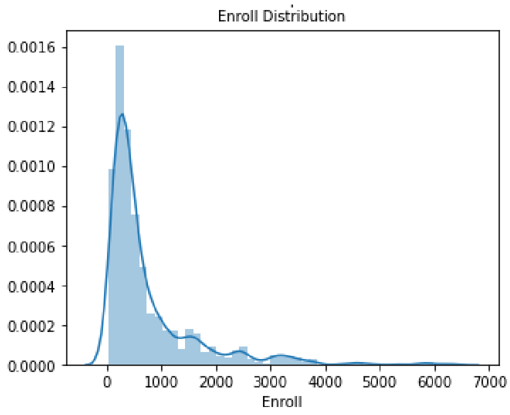
So we can clearly understand from the box plot we have outliers in the dataset.

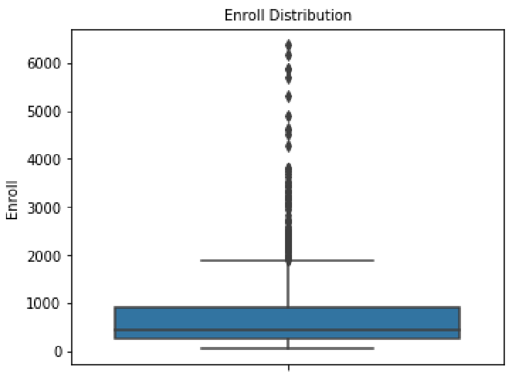
ACCEPT



The accept variable seems to have outliers. The dist plot shows us the majority of applications accepted from each university are in the range from 70 to 1500.The accept variable seems to be positively skewed.

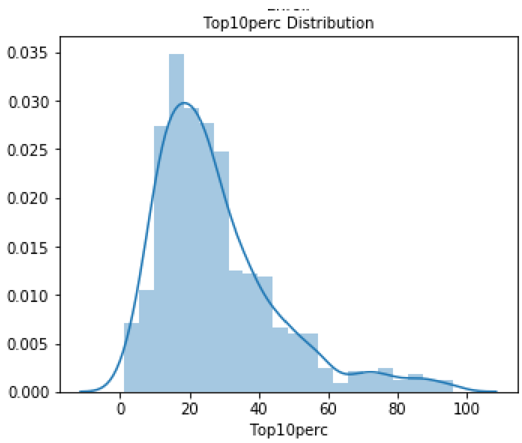
ENROLL

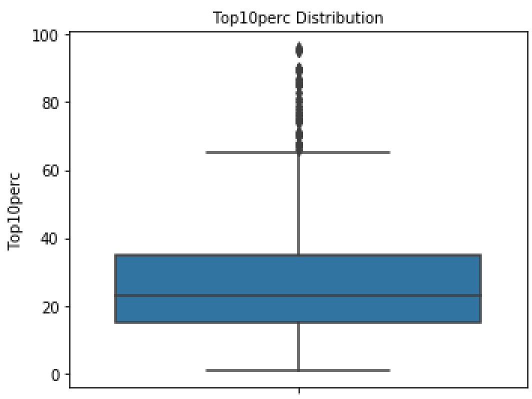




The box plot of the Enroll variable also has outliers. The distribution of the data is positively skewed. From the dist plot we can understand majority of the colleges have enrolled students in the range of 200 to 500 students.

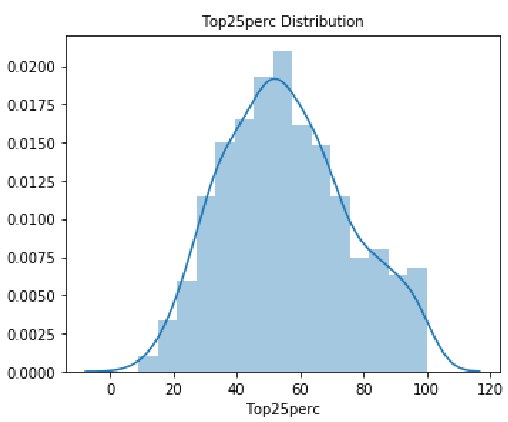
TOP 10 PERC

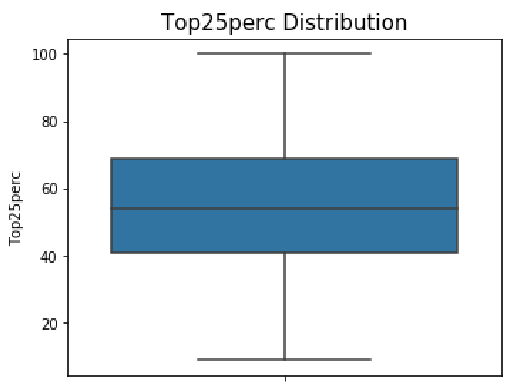




The box plot of the students from top 10 percentage of higher secondary class seems to have outliers. The distribution seems to be positively skewed. There is good amount of intake about 30 to 50 students from top 10 percentage of higher secondary class.

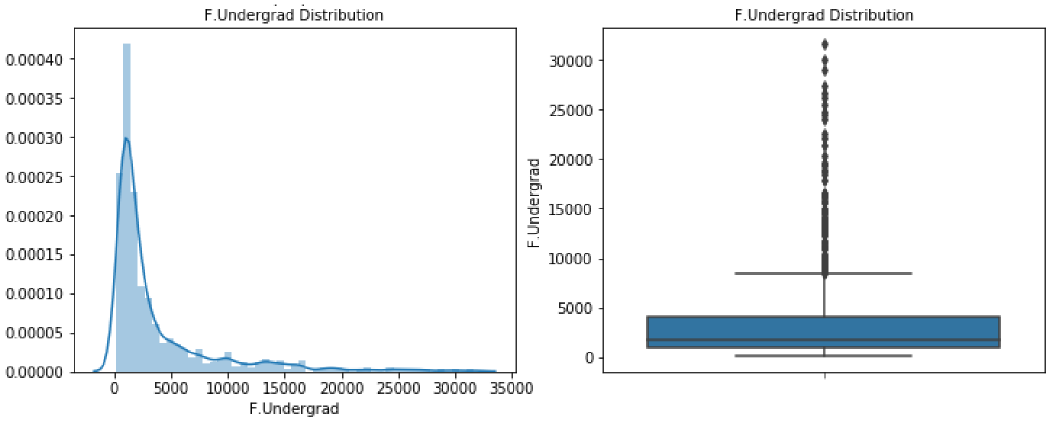
TOP 25 PERC





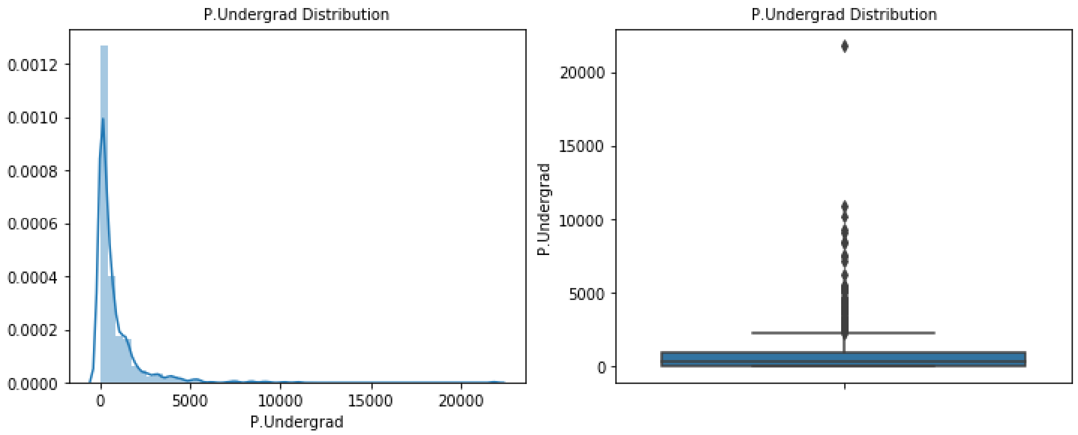
The box plot for the top 25% has no outliers. The distribution is almost normally distributed. Majority of the students are from top 25% of higher secondary class.

FULL TIME UNDERGRADUATE



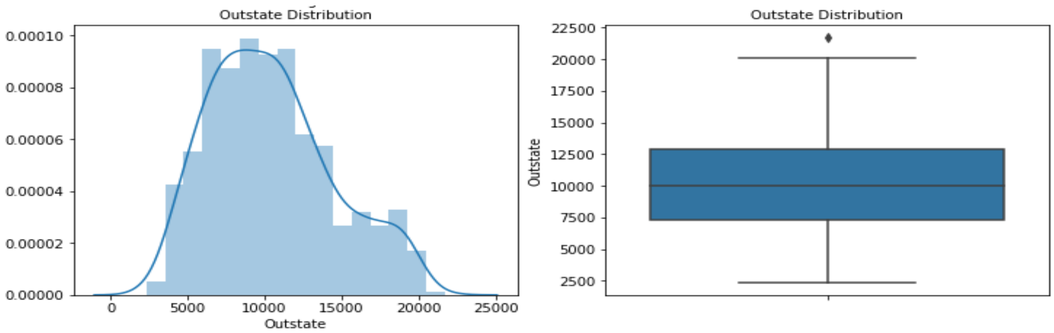
The box plot of the full time graduates has outliers. The distribution of the data is positively skewed. In the range about 3000 to 5000 they are full time graduates studying in all the university.

PART TIME UNDERGRADUATE



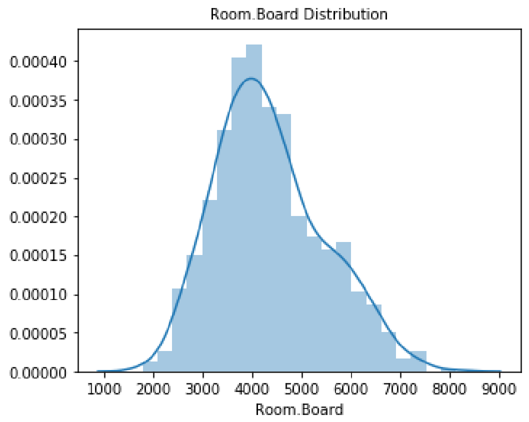
The box plot of the part time graduates has outliers. The distribution of the data is positively skewed. In the range about 1000 to 3000 they are part-time graduates studying in all the university.

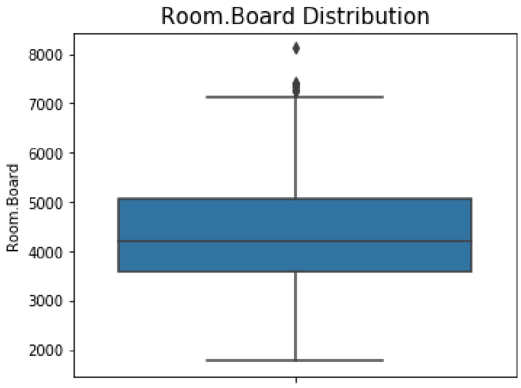
OUTSTATE



The box plot of outstate has only one outlier. The distribution is almost normally distributed.

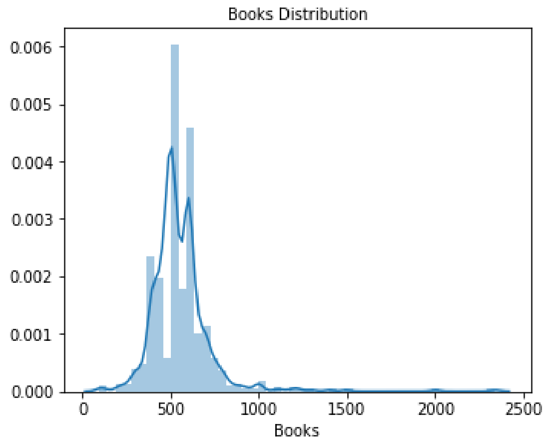
ROOM BOARD

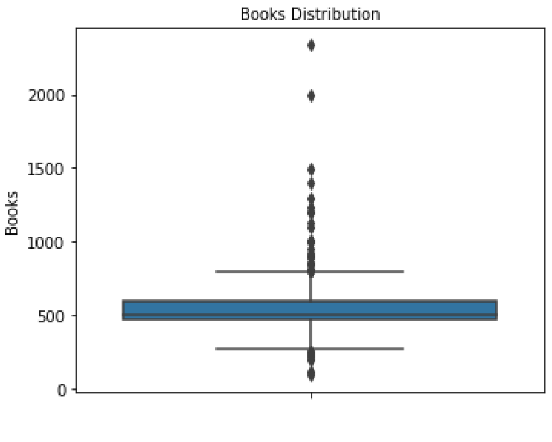




The Room board has few outliers. The distribution is normally distributed.

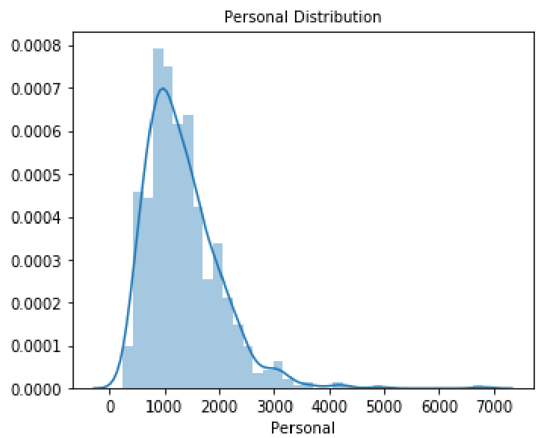
BOOKS

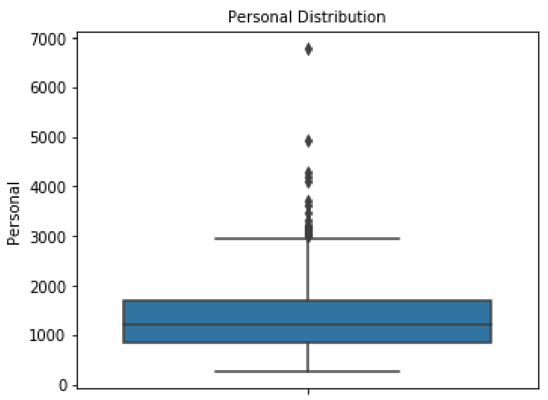




The box plot of books has outliers. The distribution seems to be bimodal. The cost of books per student seems to be in the range of 500 to 100.

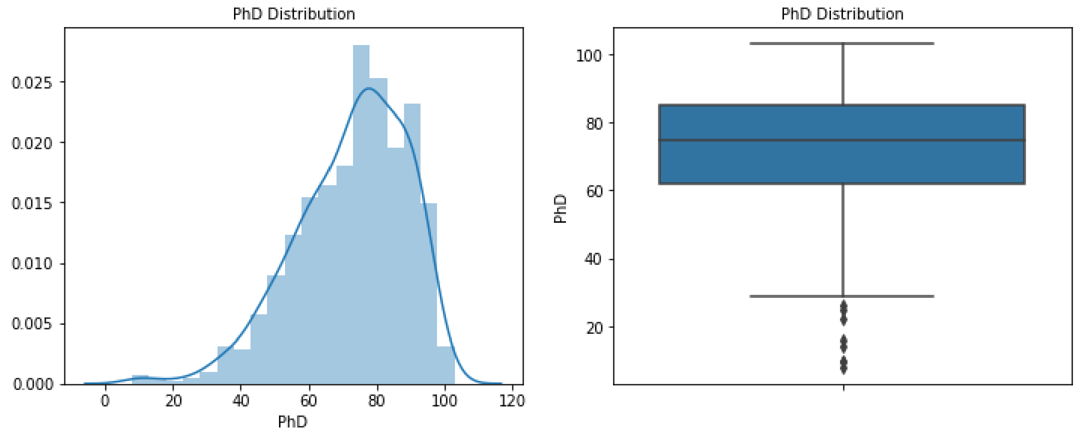
PERSONAL





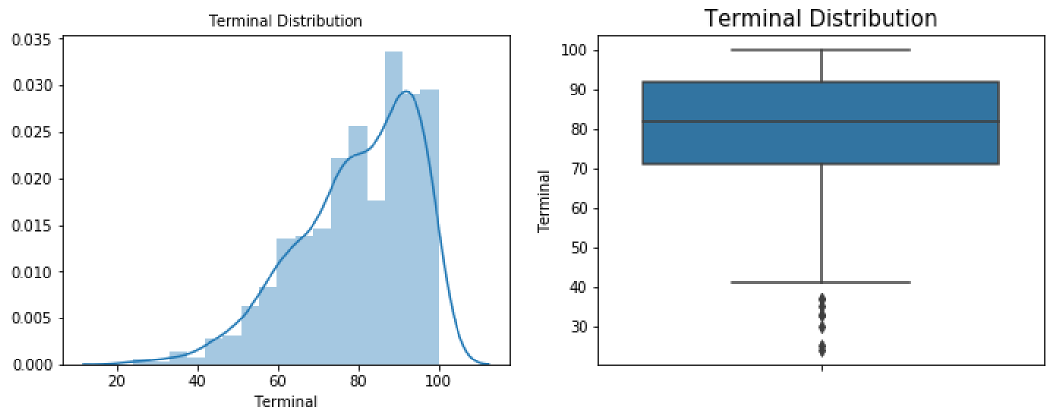
The box plot of personal expense has outliers. Some student's personal expense are way bigger than the rest of the students. The distribution seems to be positively skewed.

PHD



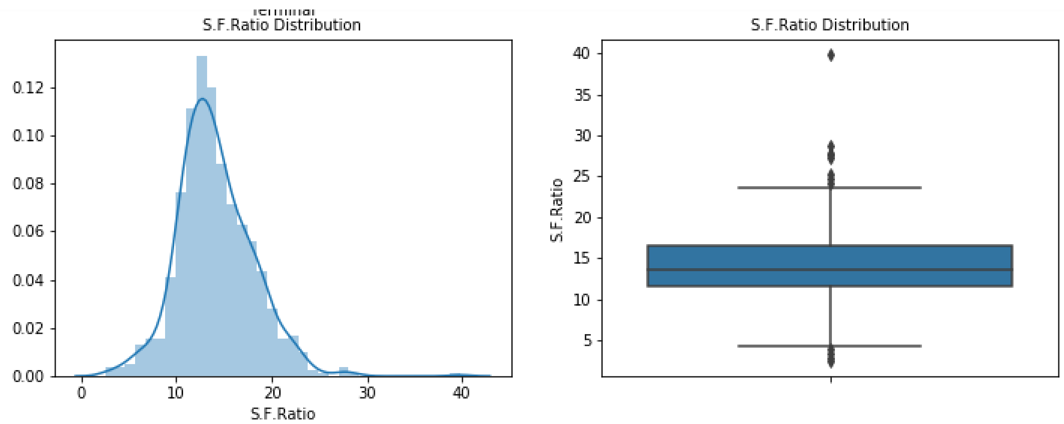
The box plot of PHD has outliers. The distribution seems to be negatively skewed.

TERMINAL



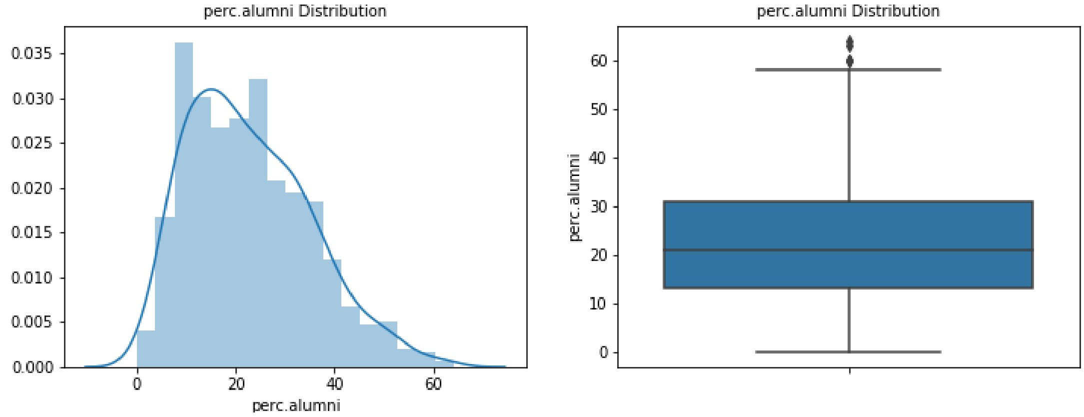
The box plot of terminal seems to have outliers in the dataset. The distribution for the terminal also seems to be negatively skewed.

SF RATIO



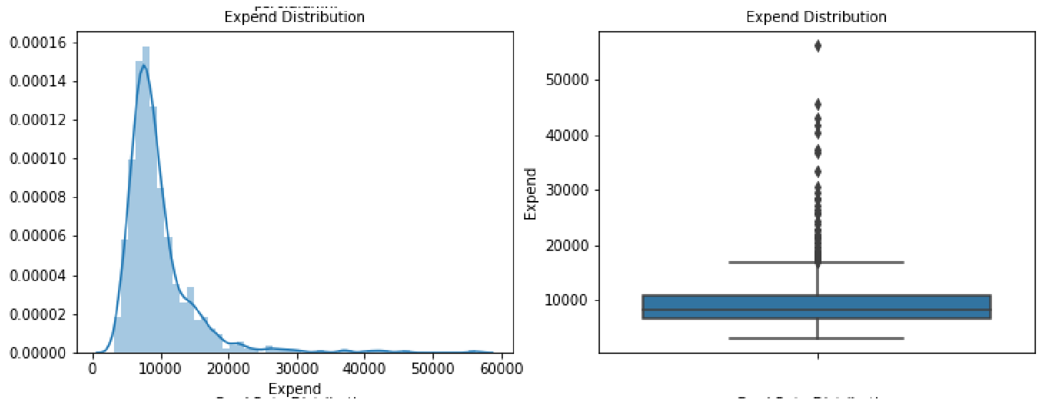
The SF ratio variable also has outliers in the dataset. The distribution is almost normally distributed. The student faculty ratio is almost same in all the university and colleges.

PERCI ALUMINI



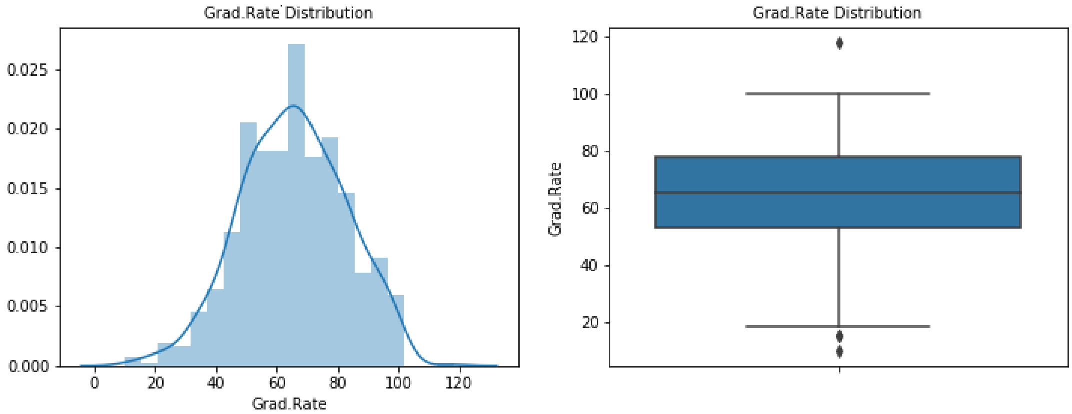
The percentage of alumni box plot seems to have outliers in the dataset. The distribution is almost normally distributed.

EXPENDITURE



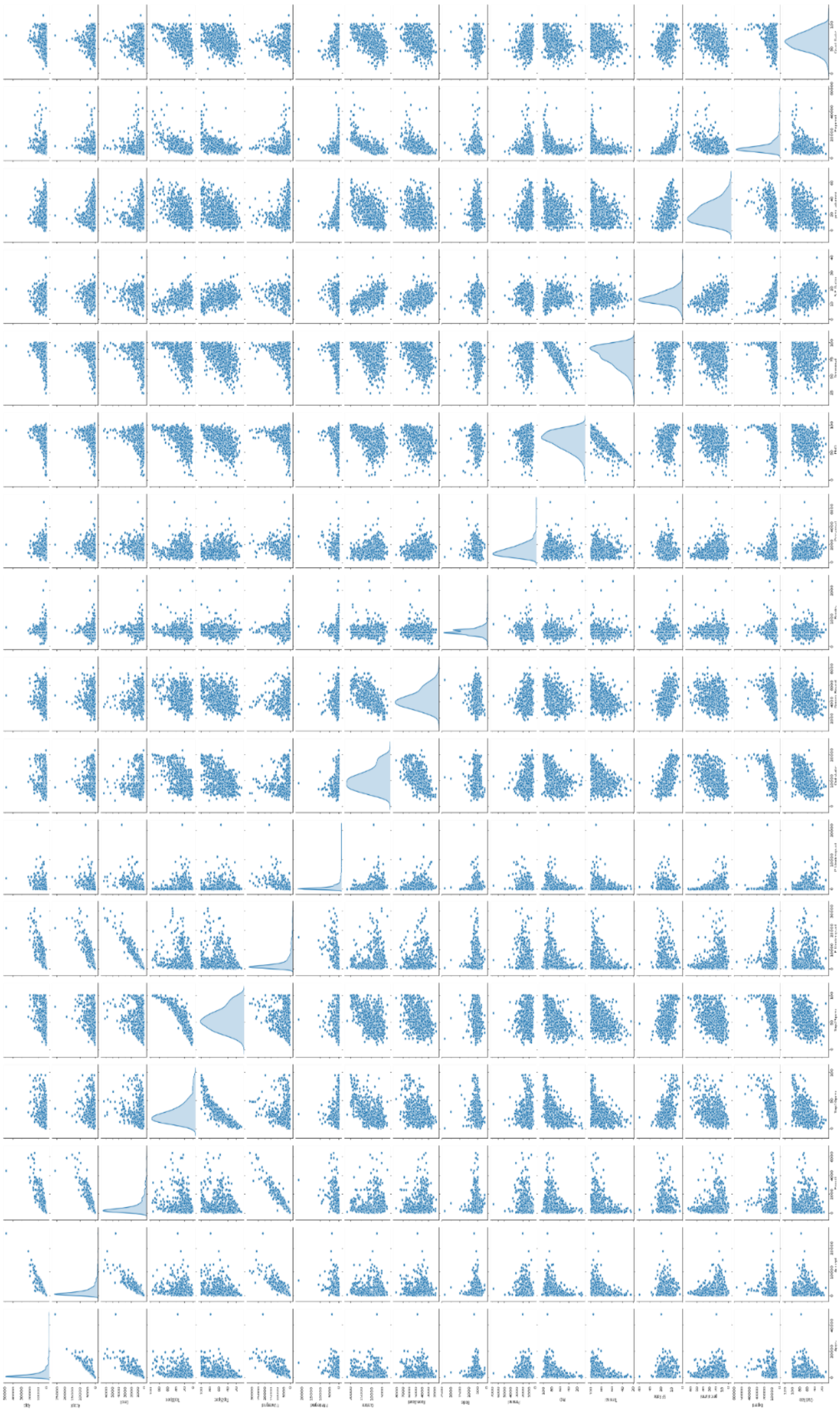
The expenditure variable also has outliers in the dataset. The distribution of the expenditure is positively skewed.

GRAD RATE



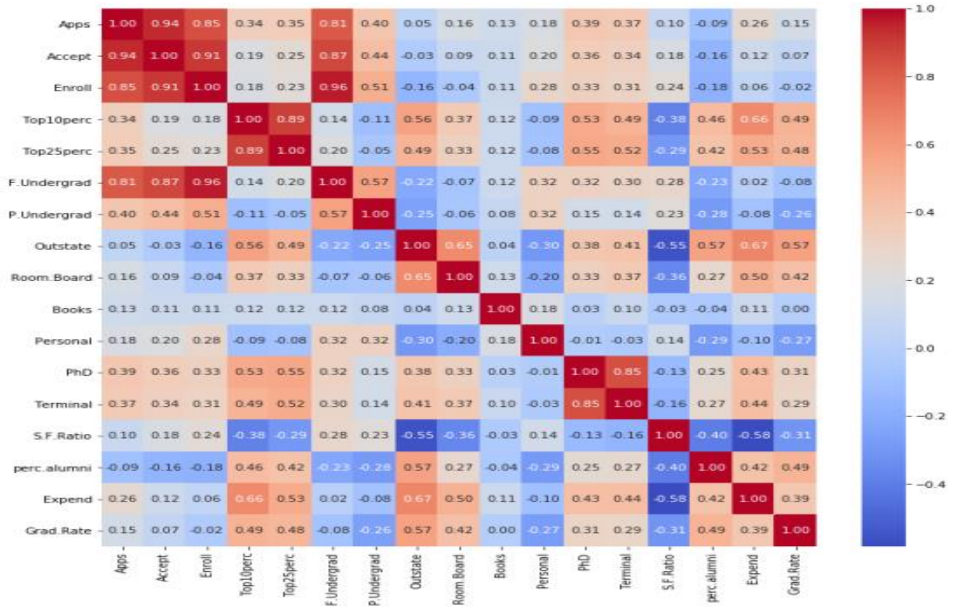
The graduation rate among the students in all the university above 60%. The box plot of the graduation rate has outliers in the dataset. The distribution is normally distributed.

MULTIVARIATE ANALYSIS



The pair plot helps us to understand the relationship between all the numerical values in the dataset. On comparing all the variables with each other we could understand the patterns or trends in the dataset.

HEATMAP



This Heat map gives us the correlation between two numerical values.

We could understand the application variable is highly positively correlated with application accepted, students enrolled and full time graduates. So this relationship gives the insights on when student submits the application it is accepted and the student is enrolled as fulltime graduate.

We can find negative correlation between application and percentage of alumni. This indicates us not all students are part of alumni of their college or university.

The application with top 10, 25 of higher secondary class, outstate, room board, books, personal, PhD, terminal, S.F ratio, expenditure and Graduation ratio are positively correlated.

2.2) Scale the variables and write the inference for using the type of scaling function for this case study.

Before scaling I have dropped the names variable which is categorical.

Now, the dataset consists of only numerical values, I have applied z-score method for this case study. We can also min max function to scale the variables.

Since, the dataset has 18 numerical columns with different scales.

For example the application, accepted application, enrolled fulltime graduates, part-time graduates, outstate are number of students. The top10 percent and top20 percent are students in which the values are given in percentage. Room board, books, and personal are values associated with money. The phd, sf ratio, percentage of alumni are percentage values of different combinations of students teachers alumini these are percentage values. The graduation rate is also percentage value of graduates who get graduated every year.

```
from scipy.stats import zscore
stud_z=stud_1.apply(zscore)
stud_z.head()
```

7]:

	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	
0	-0.346882	-0.321205	-0.063509	-0.258583	-0.191827	-0.168116	-0.209207	-0.746356	-0.964905	-0.6
1	-0.210884	-0.038703	-0.288584	-0.655656	-1.353911	-0.209788	0.244307	0.457496	1.909208	1.2
2	-0.406866	-0.376318	-0.478121	-0.315307	-0.292878	-0.549565	-0.497090	0.201305	-0.554317	-0.9
3	-0.668261	-0.681682	-0.692427	1.840231	1.677612	-0.658079	-0.520752	0.626633	0.996791	-0.6
4	-0.726176	-0.764555	-0.780735	-0.655656	-0.596031	-0.711924	0.009005	-0.716508	-0.216723	1.5

$$Z = \frac{x - \mu}{\sigma}$$

Z score tells us how many standard deviation is the point away from the mean and also the direction. Now, we can understand that all the variables are scaled by using z score function. Scaling is one of the most important method to follow before implementing models.

2.3) Comment on the comparison between covariance and the correlation matrix after scaling.

The comparison between the covariance and correlation matrix is that both of the terms measures the relationship and the dependency between two variables.

Scaling in general means representation of the dataset. The numbers will not change. We are bringing the dataset into one unit.

Covariance indicates the direction of the linear relationship between the variables whether it is positive or negative. By direction means it is directly proportional or inversely proportional.

$$Cov(x,y) = \frac{\sum (x_i - \bar{x}) * (y_i - \bar{y})}{N}$$

This below snippet is the covariance matrix on scaled dataset. We can clearly understand covariance matrix indicates direction of the linear relationship between the variables. By direction means it is directly proportional or inversely proportional

Covariance Matrix

```
%s [[ 1.00128866  0.94466636  0.84791332  0.33927032  0.35209304  0.81554018
   0.3987775   0.05022367  0.16515151  0.13272942  0.17896117  0.39120081
   0.36996762  0.09575627 -0.09034216  0.2599265   0.14694372]
 [ 0.94466636  1.00128866  0.91281145  0.19269493  0.24779465  0.87534985
   0.44183938 -0.02578774  0.09101577  0.11367165  0.20124767  0.35621633
   0.3380184   0.17645611 -0.16019604  0.12487773  0.06739929]
 [ 0.84791332  0.91281145  1.00128866  0.18152715  0.2270373   0.96588274
   0.51372977 -0.1556777  -0.04028353  0.11285614  0.28129148  0.33189629
   0.30867133  0.23757707 -0.18102711  0.06425192 -0.02236983]
 [ 0.33927032  0.19269493  0.18152715  1.00128866  0.89314445  0.1414708
  -0.10549205  0.5630552   0.37195909  0.1190116  -0.09343665  0.53251337
   0.49176793 -0.38537048  0.45607223  0.6617651   0.49562711]
 [ 0.35209304  0.24779465  0.2270373   0.89314445  1.00128866  0.19970167
  -0.05364569  0.49002449  0.33191707  0.115676   -0.08091441  0.54656564
   0.52542506 -0.29500852  0.41840277  0.52812713  0.47789622]
 [ 0.81554018  0.87534985  0.96588274  0.1414708   0.19970167  1.00128866
   0.57124738 -0.21602002 -0.06897917  0.11569867  0.31760831  0.3187472
   0.30040557  0.28006379 -0.22975792  0.01867565 -0.07887464]
 [ 0.3987775   0.44183938  0.51372977 -0.10549205 -0.05364569  0.57124738
   1.00128866 -0.25383901 -0.06140453  0.08130416  0.32029384  0.14930637
   0.14208644  0.23283016 -0.28115421 -0.08367612 -0.25733218]
 [ 0.05022367 -0.02578774 -0.1556777   0.5630552   0.49002449 -0.21602002
  -0.25383901  1.00128866  0.65509951  0.03890494 -0.29947232  0.38347594
   0.40850895 -0.55553625  0.56699214  0.6736456   0.57202613]
 [ 0.16515151  0.09101577 -0.04028353  0.37195909  0.33191707 -0.06897917
  -0.06140453  0.65509951  1.00128866  0.12812787 -0.19968518  0.32962651
   0.3750222  -0.36309504  0.27271444  0.50238599  0.42548915]
 [ 0.13272942  0.11367165  0.11285614  0.1190116   0.115676    0.11569867
   0.08130416  0.03890494  0.12812787  1.00128866  0.17952581  0.0269404
   0.10008351 -0.03197042 -0.04025955  0.11255393  0.00106226]
 [ 0.17896117  0.20124767  0.28129148 -0.09343665 -0.08091441  0.31760831
   0.32029384 -0.29947232 -0.19968518  0.17952581  1.00128866 -0.01094989
  -0.03065256  0.13652054 -0.2863366  -0.09801804 -0.26969106]
 [ 0.39120081  0.35621633  0.33189629  0.53251337  0.54656564  0.3187472
   0.14930637  0.38347594  0.32962651  0.0269404  -0.01094989  1.00128866
   0.85068186 -0.13069832  0.24932955  0.43331936  0.30543094]
 [ 0.36996762  0.3380184   0.30867133  0.49176793  0.52542506  0.30040557
   0.14208644  0.40850895  0.3750222   0.10008351 -0.03065256  0.85068186
   1.00128866 -0.16031027  0.26747453  0.43936469  0.28990033]
 [ 0.09575627  0.17645611  0.23757707 -0.38537048 -0.29500852  0.28006379
```

```
  0.23283016 -0.55553625 -0.36309504 -0.03197042  0.13652054 -0.13069832
 -0.16031027  1.00128866 -0.4034484  -0.5845844  -0.30710565]
 [-0.09034216 -0.16019604 -0.18102711  0.45607223  0.41840277 -0.22975792
 -0.28115421  0.56699214  0.27271444 -0.04025955 -0.2863366   0.24932955
  0.26747453 -0.4034484   1.00128866  0.41825001  0.49153016]
 [ 0.2599265   0.12487773  0.06425192  0.6617651   0.52812713  0.01867565
 -0.08367612  0.6736456   0.50238599  0.11255393 -0.09801804  0.43331936
  0.43936469 -0.5845844   0.41825001  1.00128866  0.39084571]
 [ 0.14694372  0.06739929 -0.02236983  0.49562711  0.47789622 -0.07887464
 -0.25733218  0.57202613  0.42548915  0.00106226 -0.26969106  0.30543094
  0.28990033 -0.30710565  0.49153016  0.39084571  1.00128866]]
```

Correlation measures the strength (how much?) and the direction of the linear relationship between two variables. Strength is that is that positively correlated or negatively correlated.

$$Correlation = \frac{Cov\ (x, y)}{\sigma x * \sigma y}$$

where:

- cov is the covariance
- σ_X is the standard deviation of X
- σ_Y is the standard deviation of Y

This below snippet is the correlation matrix. We can clearly understand the correlation matrix which gives the strength and the relationship between the variables.

The correlation matrix before scaling and after scaling will remain the same.

From this snippet we can understand variables which are highly positively correlated and the variables which are highly negatively correlated. We can also understand the variables which are moderately correlated with each other.

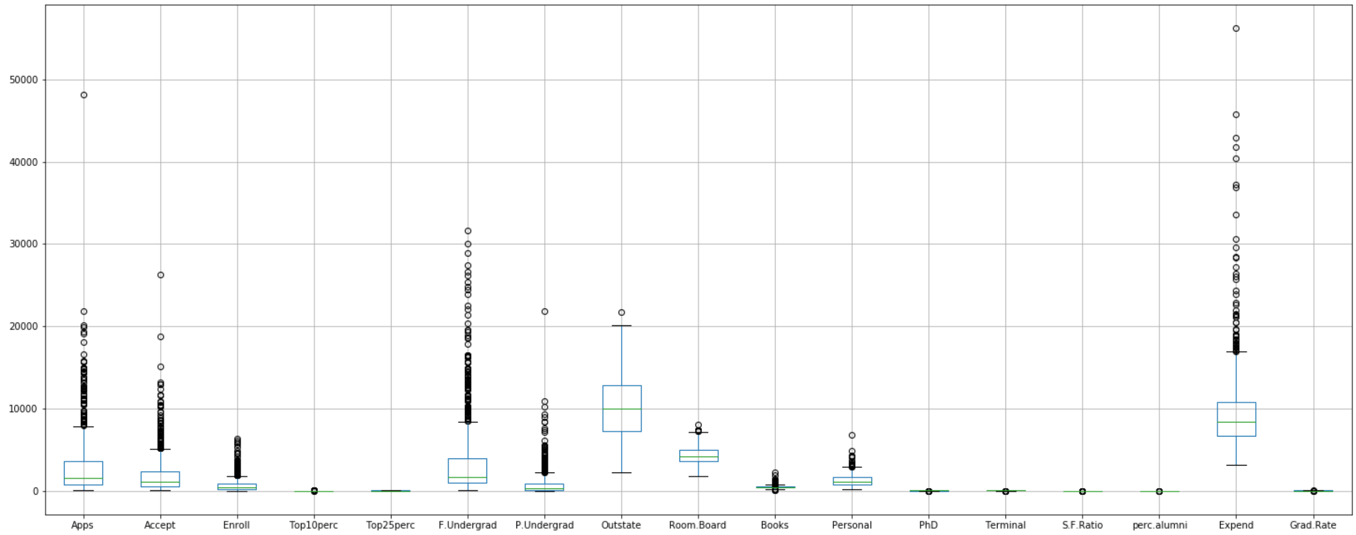
We can see that application, acceptance, enrollment and fulltime graduates are highly positively correlated

Also the top 10 percentage and top 25 percentage are highly positively correlated

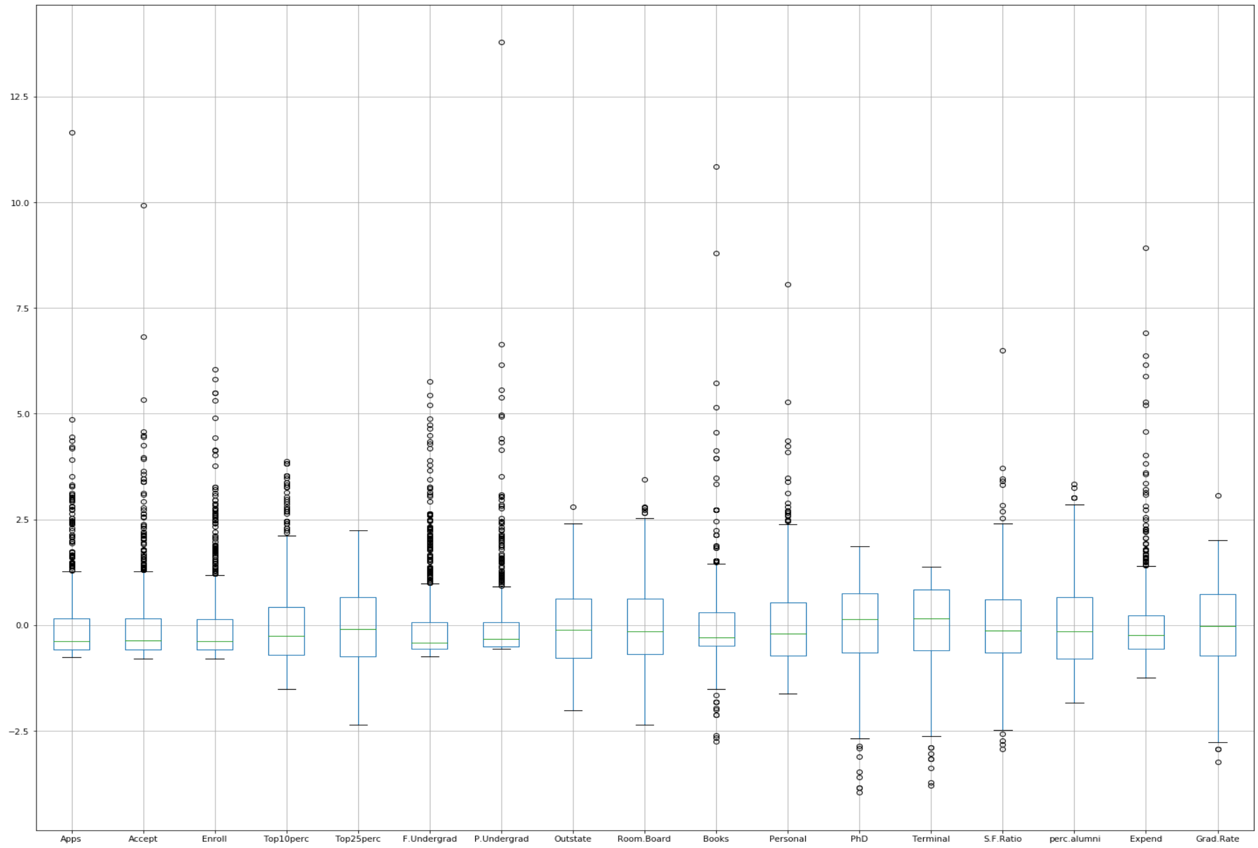
Apps	1.000000	0.943451	0.846822	0.338834	0.351640	0.814491	0.398264	0.050159	0.164939	0.132559	0.178731	0.390697
Accept	0.943451	1.000000	0.911637	0.192447	0.247476	0.874223	0.441271	-0.025755	0.090899	0.113525	0.200989	0.355758
Enroll	0.846822	0.911637	1.000000	0.181294	0.226745	0.964640	0.513069	-0.155477	-0.040232	0.112711	0.280929	0.331469
Top10perc	0.338834	0.192447	0.181294	1.000000	0.891995	0.141289	-0.105356	0.562331	0.371480	0.118858	-0.093316	0.531828
Top25perc	0.351640	0.247476	0.226745	0.891995	1.000000	0.199445	-0.053577	0.489394	0.331490	0.115527	-0.080810	0.545862
F.Undergrad	0.814491	0.874223	0.964640	0.141289	0.199445	1.000000	0.570512	-0.215742	-0.068890	0.115550	0.317200	0.318337
P.Undergrad	0.398264	0.441271	0.513069	-0.105356	-0.053577	0.570512	1.000000	-0.253512	-0.061326	0.081200	0.319882	0.149114
Outstate	0.050159	-0.025755	-0.155477	0.562331	0.489394	-0.215742	-0.253512	1.000000	0.654256	0.038855	-0.299087	0.382982
Room.Board	0.164939	0.090899	-0.040232	0.371480	0.331490	-0.068890	-0.061326	0.654256	1.000000	0.127963	-0.199428	0.329202
Books	0.132559	0.113525	0.112711	0.118858	0.115527	0.115550	0.081200	0.038855	0.127963	1.000000	0.179295	0.026906
Personal	0.178731	0.200989	0.280929	-0.093316	-0.080810	0.317200	0.319882	-0.299087	-0.199428	0.179295	1.000000	-0.010936
PhD	0.390697	0.355758	0.331469	0.531828	0.545862	0.318337	0.149114	0.382982	0.329202	0.026906	-0.010936	1.000000
Terminal	0.369491	0.337583	0.308274	0.491135	0.524749	0.300019	0.141904	0.407983	0.374540	0.099955	-0.030613	0.849587
S.F.Ratio	0.095633	0.176229	0.237271	-0.384875	-0.294629	0.279703	0.232531	-0.554821	-0.362628	-0.031929	0.136345	-0.130530
perc.alumni	-0.090226	-0.159990	-0.180794	0.455485	0.417864	-0.229462	-0.280792	0.566262	0.272363	-0.040208	-0.285968	0.249009
Expend	0.259592	0.124717	0.064169	0.660913	0.527447	0.018652	-0.083568	0.672779	0.501739	0.112409	-0.097892	0.432762
Grad.Rate	0.146755	0.067313	-0.022341	0.494989	0.477281	-0.078773	-0.257001	0.571290	0.424942	0.001061	-0.269344	0.305038

2.4) check the dataset for outliers before and after scaling. Draw your inferences from this exercise.

Checking the data before scaling



Checking the dataset after scaling



Inference

The outliers are still present in dataset.

Reason: scaling does not remove outliers scaling scales the values on a Z score distribution. We can use any one method to remove outliers for further processes.

For example if we wish to remove outliers we can consider taking 3 standard deviations as outliers or either we can remove them or impute them with IQR values.

2.5) Build the covariance matrix and calculate the eigenvalues and the eigenvector.

Covariance Matrix

```
[[ 1.00128866  0.94466636  0.84791332  0.33927032  0.35209304  0.81554018
   0.3987775   0.05022367  0.16515151  0.13272942  0.17896117  0.39120081
   0.36996762  0.09575627 -0.09034216  0.2599265   0.14694372]
 [ 0.94466636  1.00128866  0.91281145  0.19269493  0.24779465  0.87534985
   0.44183938 -0.02578774  0.09101577  0.11367165  0.20124767  0.35621633
   0.3380184   0.17645611 -0.16019604  0.12487773  0.06739929]
 [ 0.84791332  0.91281145  1.00128866  0.18152715  0.2270373   0.96588274
   0.51372977 -0.1556777  -0.04028353  0.11285614  0.28129148  0.33189629
   0.30867133  0.23757707 -0.18102711  0.06425192 -0.02236983]
 [ 0.33927032  0.19269493  0.18152715  1.00128866  0.89314445  0.1414708
  -0.10549205  0.5630552   0.37195909  0.1190116  -0.09343665  0.53251337
   0.49176793 -0.38537048  0.45607223  0.6617651   0.49562711]
 [ 0.35209304  0.24779465  0.2270373   0.89314445  1.00128866  0.19970167
  -0.05364569  0.49002449  0.33191707  0.115676   -0.08091441  0.54656564
   0.52542506 -0.29500852  0.41840277  0.52812713  0.47789622]
 [ 0.81554018  0.87534985  0.96588274  0.1414708   0.19970167  1.00128866
   0.57124738 -0.21602002 -0.06897917  0.11569867  0.31760831  0.3187472
   0.30040557  0.28006379 -0.22975792  0.01867565 -0.07887464]
 [ 0.3987775   0.44183938  0.51372977 -0.10549205 -0.05364569  0.57124738
   1.00128866 -0.25383901 -0.06140453  0.08130416  0.32029384  0.14930637
   0.14208644  0.23283016 -0.28115421 -0.08367612 -0.25733218]
 [ 0.05022367 -0.02578774 -0.1556777   0.5630552   0.49002449 -0.21602002
  -0.25383901  1.00128866  0.65509951  0.03890494 -0.29947232  0.38347594
   0.40850895 -0.55553625  0.56699214  0.6736456   0.57202613]
 [ 0.16515151  0.09101577 -0.04028353  0.37195909  0.33191707 -0.06897917
  -0.06140453  0.65509951  1.00128866  0.12812787 -0.19968518  0.32962651
   0.3750222  -0.36309504  0.27271444  0.50238599  0.42548915]
 [ 0.13272942  0.11367165  0.11285614  0.1190116   0.115676    0.11569867
   0.08130416  0.03890494  0.12812787  1.00128866  0.17952581  0.0269404
   0.10008351 -0.03197042 -0.04025955  0.11255393  0.00106226]
 [ 0.17896117  0.20124767  0.28129148 -0.09343665 -0.08091441  0.31760831
   0.32029384 -0.29947232 -0.19968518  0.17952581  1.00128866 -0.01094989
  -0.03065256  0.13652054 -0.2863366  -0.09801804 -0.26969106]
 [ 0.39120081  0.35621633  0.33189629  0.53251337  0.54656564  0.3187472
   0.14930637  0.38347594  0.32962651  0.0269404  -0.01094989  1.00128866
   0.85068186 -0.13069832  0.24932955  0.43331936  0.30543094]
 [ 0.36996762  0.3380184   0.30867133  0.49176793  0.52542506  0.30040557
```

0.14208644 0.40850895 0.3750222 0.10008351 -0.03065256 0.85068186
1.00128866 -0.16031027 0.26747453 0.43936469 0.28990033]
[0.09575627 0.17645611 0.23757707 -0.38537048 -0.29500852 0.28006379
0.23283016 -0.55553625 -0.36309504 -0.03197042 0.13652054 -0.13069832
-0.16031027 1.00128866 -0.4034484 -0.5845844 -0.30710565]
[-0.09034216 -0.16019604 -0.18102711 0.45607223 0.41840277 -0.22975792
-0.28115421 0.56699214 0.27271444 -0.04025955 -0.2863366 0.24932955
0.26747453 -0.4034484 1.00128866 0.41825001 0.49153016]
[0.2599265 0.12487773 0.06425192 0.6617651 0.52812713 0.01867565
-0.08367612 0.6736456 0.50238599 0.11255393 -0.09801804 0.43331936
0.43936469 -0.5845844 0.41825001 1.00128866 0.39084571]
[0.14694372 0.06739929 -0.02236983 0.49562711 0.47789622 -0.07887464
-0.25733218 0.57202613 0.42548915 0.00106226 -0.26969106 0.30543094
0.28990033 -0.30710565 0.49153016 0.39084571 1.00128866]]

Eigen Vectors

[[-2.48765602e-01 3.31598227e-01 6.30921033e-02 -2.81310530e-01
5.74140964e-03 1.62374420e-02 4.24863486e-02 1.03090398e-01
9.02270802e-02 -5.25098025e-02 3.58970400e-01 -4.59139498e-01
4.30462074e-02 -1.33405806e-01 8.06328039e-02 -5.95830975e-01
2.40709086e-02]
[-2.07601502e-01 3.72116750e-01 1.01249056e-01 -2.67817346e-01
5.57860920e-02 -7.53468452e-03 1.29497196e-02 5.62709623e-02
1.77864814e-01 -4.11400844e-02 -5.43427250e-01 5.18568789e-01
-5.84055850e-02 1.45497511e-01 3.34674281e-02 -2.92642398e-01
-1.45102446e-01]
[-1.76303592e-01 4.03724252e-01 8.29855709e-02 -1.61826771e-01
-5.56936353e-02 4.25579803e-02 2.76928937e-02 -5.86623552e-02
1.28560713e-01 -3.44879147e-02 6.09651110e-01 4.04318439e-01
-6.93988831e-02 -2.95896092e-02 -8.56967180e-02 4.44638207e-01
1.11431545e-02]
[-3.54273947e-01 -8.24118211e-02 -3.50555339e-02 5.15472524e-02
-3.95434345e-01 5.26927980e-02 1.61332069e-01 1.22678028e-01
-3.41099863e-01 -6.40257785e-02 -1.44986329e-01 1.48738723e-01
-8.10481404e-03 -6.97722522e-01 -1.07828189e-01 -1.02303616e-03
3.85543001e-02]
[-3.44001279e-01 -4.47786551e-02 2.41479376e-02 1.09766541e-01
-4.26533594e-01 -3.30915896e-02 1.18485556e-01 1.02491967e-01
-4.03711989e-01 -1.45492289e-02 8.03478445e-02 -5.18683400e-02
-2.73128469e-01 6.17274818e-01 1.51742110e-01 -2.18838802e-02
-8.93515563e-02]
[-1.54640962e-01 4.17673774e-01 6.13929764e-02 -1.00412335e-01
-4.34543659e-02 4.34542349e-02 2.50763629e-02 -7.88896442e-02
5.94419181e-02 -2.08471834e-02 -4.14705279e-01 -5.60363054e-01
-8.11578181e-02 -9.91640992e-03 -5.63728817e-02 5.23622267e-01
5.61767721e-02]
[-2.64425045e-02 3.15087830e-01 -1.39681716e-01 1.58558487e-01
3.02385408e-01 1.91198583e-01 -6.10423460e-02 -5.70783816e-01
-5.60672902e-01 2.23105808e-01 9.01788964e-03 5.27313042e-02

```
 1.00693324e-01 -2.09515982e-02  1.92857500e-02 -1.25997650e-01
 -6.35360730e-02]
[-2.94736419e-01 -2.49643522e-01 -4.65988731e-02 -1.31291364e-01
  2.22532003e-01  3.00003910e-02 -1.08528966e-01 -9.84599754e-03
  4.57332880e-03 -1.86675363e-01  5.08995918e-02 -1.01594830e-01
  1.43220673e-01 -3.83544794e-02 -3.40115407e-02  1.41856014e-01
 -8.23443779e-01]
[-2.49030449e-01 -1.37808883e-01 -1.48967389e-01 -1.84995991e-01
  5.60919470e-01 -1.62755446e-01 -2.09744235e-01  2.21453442e-01
 -2.75022548e-01 -2.98324237e-01  1.14639620e-03  2.59293381e-02
 -3.59321731e-01 -3.40197083e-03 -5.84289756e-02  6.97485854e-02
  3.54559731e-01]
[-6.47575181e-02  5.63418434e-02 -6.77411649e-01 -8.70892205e-02
 -1.27288825e-01 -6.41054950e-01  1.49692034e-01 -2.13293009e-01
  1.33663353e-01  8.20292186e-02  7.72631963e-04 -2.88282896e-03
  3.19400370e-02  9.43887925e-03 -6.68494643e-02 -1.14379958e-02
 -2.81593679e-02]
[ 4.25285386e-02  2.19929218e-01 -4.99721120e-01  2.30710568e-01
 -2.22311021e-01  3.31398003e-01 -6.33790064e-01  2.32660840e-01
  9.44688900e-02 -1.36027616e-01 -1.11433396e-03  1.28904022e-02
 -1.85784733e-02  3.09001353e-03  2.75286207e-02 -3.94547417e-02
 -3.92640266e-02]
[-3.18312875e-01  5.83113174e-02  1.27028371e-01  5.34724832e-01
  1.40166326e-01 -9.12555212e-02  1.09641298e-03  7.70400002e-02
  1.85181525e-01  1.23452200e-01  1.38133366e-02 -2.98075465e-02
  4.03723253e-02  1.12055599e-01 -6.91126145e-01 -1.27696382e-01
  2.32224316e-02]
[-3.17056016e-01  4.64294477e-02  6.60375454e-02  5.19443019e-01
  2.04719730e-01 -1.54927646e-01  2.84770105e-02  1.21613297e-02
  2.54938198e-01  8.85784627e-02  6.20932749e-03  2.70759809e-02
 -5.89734026e-02 -1.58909651e-01  6.71008607e-01  5.83134662e-02
  1.64850420e-02]
[ 1.76957895e-01  2.46665277e-01  2.89848401e-01  1.61189487e-01
 -7.93882496e-02 -4.87045875e-01 -2.19259358e-01  8.36048735e-02
 -2.74544380e-01 -4.72045249e-01 -2.22215182e-03  2.12476294e-02
  4.45000727e-01  2.08991284e-02  4.13740967e-02  1.77152700e-02
 -1.10262122e-02]
[-2.05082369e-01 -2.46595274e-01  1.46989274e-01 -1.73142230e-02
 -2.16297411e-01  4.73400144e-02 -2.43321156e-01 -6.78523654e-01
  2.55334907e-01 -4.22999706e-01 -1.91869743e-02 -3.33406243e-03
 -1.30727978e-01  8.41789410e-03 -2.71542091e-02 -1.04088088e-01
  1.82660654e-01]
[-3.18908750e-01 -1.31689865e-01 -2.26743985e-01 -7.92734946e-02
  7.59581203e-02  2.98118619e-01  2.26584481e-01  5.41593771e-02
  4.91388809e-02 -1.32286331e-01 -3.53098218e-02  4.38803230e-02
  6.92088870e-01  2.27742017e-01  7.31225166e-02  9.37464497e-02
  3.25982295e-01]
[-2.52315654e-01 -1.69240532e-01  2.08064649e-01 -2.69129066e-01
 -1.09267913e-01 -2.16163313e-01 -5.59943937e-01  5.33553891e-03
 -4.19043052e-02  5.90271067e-01 -1.30710024e-02  5.00844705e-03
```

2.19839000e-01 3.39433604e-03 3.64767385e-02 6.91969778e-02
1.22106697e-01]]

Eigen Values
[5.45052162 4.48360686 1.17466761 1.00820573 0.93423123 0.84849117
0.6057878 0.58787222 0.53061262 0.4043029 0.02302787 0.03672545
0.31344588 0.08802464 0.1439785 0.16779415 0.22061096]

2.6) write the explicit form of the first PC (in terms of Eigen Vectors).

```
pca.components_
```

```
]: array([[ 0.2487656 ,  0.2076015 ,  0.17630359,  0.35427395,  0.34400128,
           0.15464096,  0.0264425 ,  0.29473642,  0.24903045,  0.06475752,
          -0.04252854,  0.31831287,  0.31705602, -0.17695789,  0.20508237,
           0.31890875,  0.25231565],
```

```
print('The Linear eq of 1st component: ')
for i in range(0,stud_z.shape[1]):
    print('{} * {}'.format(np.round(pca.components_[0][i],3),stud_z.columns[i]),end=' + ')
```

The Linear equation of 1st component:
0.249 * Apps + 0.208 * Accept + 0.176 * Enrol + 0.354 * Top10perc + 0.344 * Top25perc + 0.155 * F.Undergrad + 0.026 * P.Undergrad + 0.295 * Outstate + 0.249 * Room. Board + 0.065 * Books + -0.043 * Personal + 0.318 * PhD + 0.317 * Terminal + -0.177 * S.F.Rat io + 0.205 * perc.alumni + 0.319 * Expend + 0.252 * Grad. Rate

2.7) discuss the cumulative values of the eigenvalues. How does it help you to decide on the optimum number of principal components? What do the eigenvectors indicate? Perform PCA and export the data of the Principal Component scores into a data frame.

```
tot = sum(eig_vals)
var_exp = [( i /tot ) * 100 for i in sorted(eig_vals, reverse=True)]
cum_var_exp = np.cumsum(var_exp)
cum_var_exp
```

```
array([ 32.0206282 ,  58.36084263,  65.26175919,  71.18474841,
        76.67315352,  81.65785448,  85.21672597,  88.67034731,
        91.78758099,  94.16277251,  96.00419883,  97.30024023,
        98.28599436,  99.13183669,  99.64896227,  99.86471628,
       100.        ])
```

Adding the Eigen values we will get sum of 100.

To decide the optimum number of principal components

1. Check for cumulative variance up to 90%, check the corresponding associated with 90%
2. The incremental value between the components should not be less than five percent.

So basis on this we can decide the optimum number of principal components as 6, because after this the incremental value between the is less than 5%.

So, we select 5 principal components for this case study.

```
pca = PCA(n_components=5)
X_pca= pca.fit_transform(stud_z)
```

```
pca.components_
```

```
]: array([[ 0.2487656 ,  0.2076015 ,  0.17630359,  0.35427395,  0.34400128,
          0.15464096,  0.0264425 ,  0.29473642,  0.24903045,  0.06475752,
         -0.04252854,  0.31831287,  0.31705602, -0.17695789,  0.20508237,
          0.31890875,  0.25231565],
        [ 0.33159823,  0.37211675,  0.40372425, -0.08241182, -0.04477866,
          0.41767377,  0.31508783, -0.24964352, -0.13780888,  0.05634184,
          0.21992922,  0.05831132,  0.04642945,  0.24666528, -0.24659527,
         -0.13168986, -0.16924053],
        [-0.06309211, -0.10124904, -0.08298556,  0.03505554, -0.02414794,
         -0.06139299,  0.13968172,  0.04659887,  0.14896739,  0.67741165,
          0.49972112, -0.12702837, -0.06603754, -0.2898484 , -0.14698927,
          0.22674399, -0.20806465],
        [ 0.28131053,  0.26781734,  0.16182677, -0.05154725, -0.10976654,
          0.10041234, -0.15855849,  0.13129136,  0.18499599,  0.08708922,
         -0.23071057, -0.53472483, -0.51944302, -0.16118949,  0.01731422,
          0.07927349,  0.26912907],
        [ 0.00574141,  0.05578609, -0.05569366, -0.39543434, -0.4265336 ,
         -0.04345435,  0.30238541,  0.222532  ,  0.56091947, -0.12728883,
         -0.22231102,  0.14016633,  0.20471973, -0.07938825, -0.21629741,
          0.07595812, -0.10926791]])
```

The first components explain 32.02% variance in data

The first two components explains 58.36% variance in data

The first three components explains 65.26% variance in data

The first four components explains 71.18% variance in data

The first five components explains 76.67% variance in data

```
tot = sum(eig_vals)
var_exp = [( i /tot ) * 100 for i in sorted(eig_vals, reverse=True)]
cum_var_exp = np.cumsum(var_exp)
cum_var_exp
```

```
]: array([ 32.0206282 ,  58.36084263,  65.26175919,  71.18474841,
         76.67315352,  81.65785448,  85.21672597,  88.67034731,
         91.78758099,  94.16277251,  96.00419883,  97.30024023,
         98.28599436,  99.13183669,  99.64896227,  99.86471628,
        100.        ])
```

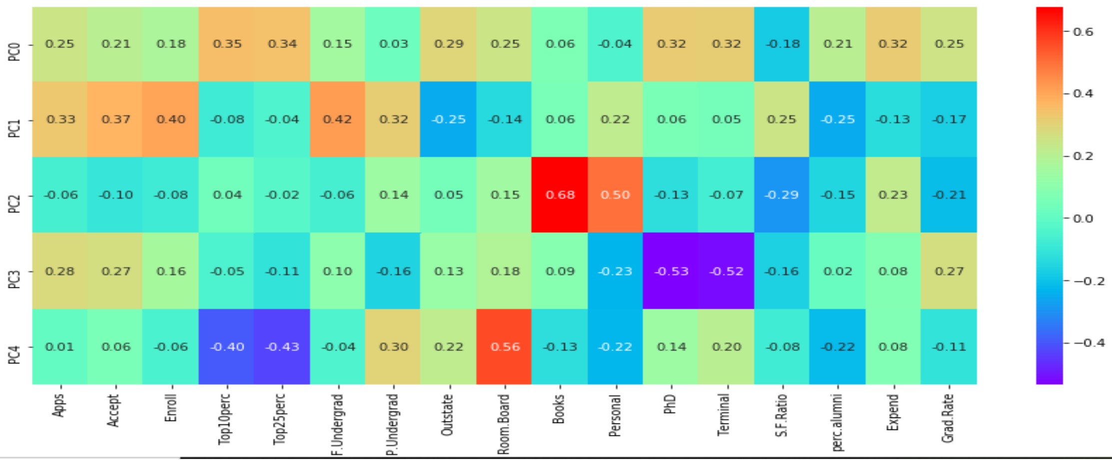
The Eigen vectors or PC for this case study is five, we can understand how much each variable contributes to the principal components. In other words we can also say weights attached to each variable. With this Eigen vectors we can understand which variable has more weightage and influences the dataset in the principal components. The PCA reduces the multi collinearity and with this reduced collineraity we can runs models and improved efficiency scores.

PCA is performed and it is exported into a data frame. After PCA the multi collinearity is highly reduced.

```
df_comp = pd.DataFrame(pca.components_,columns=list(stud_z))
df_comp
```

]:

	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD	Terminal
0	0.248766	0.207602	0.176304	0.354274	0.344001	0.154641	0.026443	0.294736	0.249030	0.064758	-0.042529	0.318313	0.317056
1	0.331598	0.372117	0.403724	-0.082412	-0.044779	0.417674	0.315088	-0.249644	-0.137809	0.056342	0.219929	0.058311	0.046429
2	-0.063092	-0.101249	-0.082986	0.035056	-0.024148	-0.061393	0.139682	0.046599	0.148967	0.677412	0.499721	-0.127028	-0.066038
3	0.281311	0.267817	0.161827	-0.051547	-0.109767	0.100412	-0.158558	0.131291	0.184996	0.087089	-0.230711	-0.534725	-0.519443
4	0.005741	0.055786	-0.055694	-0.395434	-0.426534	-0.043454	0.302385	0.222532	0.560919	-0.127289	-0.222311	0.140166	0.204720



2.8) mention the business implication of using the Principal Component Analysis for this case study.

This business case study is about education dataset which contain the names of various colleges, which has various details of colleges and university. To understand more about the dataset we perform univariate analysis and multivariate analysis which gives us the understanding about the variables. From analysis we can understand the distribution of the dataset, skew, and patterns in the dataset. From multivariate analysis we can understand the correlation of variables. Inference of multivariate analysis shows we can understand multiple variables highly correlated with each other. The scaling helps the dataset to standardize the variable in one scale. Outliers are imputed using IQR values once the values are imputed we can perform PCA. The principal component analysis is used reduce the multicollinearity between the variables. Depending on the variance of the dataset we can reduce the PCA components. The PCA components for this business case is 5 where we could understand the maximum variance of the dataset. Using the components we can now understand the reduced multicollinearity in the dataset. with this analysis we can perform further analysis and model building PCA will improve the efficiency of machine learning models.