Exercises

February 21, 2025

[1]:
```
from notebookfuncs import *
```

[2]:
```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.pyplot import subplots
import pandas as pd
from ISLP import load_data
import seaborn as sns
from numpy import median
```

[3]:
```
College = pd.read_csv("College.csv")
College
```

[3]:
```
                         Unnamed: 0 Private   Apps  Accept  Enroll  Top10perc  \
0      Abilene Christian University     Yes   1660    1232     721         23
1                Adelphi University     Yes   2186    1924     512         16
2                    Adrian College     Yes   1428    1097     336         22
3               Agnes Scott College     Yes    417     349     137         60
4         Alaska Pacific University     Yes    193     146      55         16
..                              ...     ...    ...     ...     ...        ...
772         Worcester State College      No   2197    1515     543          4
773               Xavier University     Yes   1959    1805     695         24
774  Xavier University of Louisiana     Yes   2097    1915     695         34
775                 Yale University     Yes  10705    2453    1317         95
776    York College of Pennsylvania     Yes   2989    1855     691         28

     Top25perc  F.Undergrad  P.Undergrad  Outstate  Room.Board  Books  \
0           52         2885          537      7440        3300    450
1           29         2683         1227     12280        6450    750
2           50         1036           99     11250        3750    400
3           89          510           63     12960        5450    450
4           44          249          869      7560        4120    800
..         ...          ...          ...       ...         ...    ...
772         26         3089         2029      6797        3900    500
773         47         2849         1107     11520        4960    600
774         61         2793          166      6900        4200    617
775         99         5217           83     19840        6510    630
776         63         2988         1726      4990        3560    500
```

```
     Personal  PhD  Terminal  S.F.Ratio  perc.alumni  Expend  Grad.Rate
0        2200   70        78       18.1           12    7041         60
1        1500   29        30       12.2           16   10527         56
2        1165   53        66       12.9           30    8735         54
3         875   92        97        7.7           37   19016         59
4        1500   76        72       11.9            2   10922         15
..        …   …        …        …          …     …        …
772      1200   60        60       21.0           14    4469         40
773      1250   73        75       13.3           31    9189         83
774       781   67        75       14.4           20    8323         49
775      2115   96        96        5.8           49   40386         99
776      1250   75        75       18.1           28    4509         99

[777 rows x 19 columns]
```

[4]:
```
college2 = pd.read_csv("College.csv", index_col=0)
college2
```

[4]:
```
                               Private   Apps  Accept  Enroll  Top10perc  \
Abilene Christian University       Yes   1660    1232     721         23
Adelphi University                 Yes   2186    1924     512         16
Adrian College                     Yes   1428    1097     336         22
Agnes Scott College                Yes    417     349     137         60
Alaska Pacific University          Yes    193     146      55         16
…                                    …     …      …       …          …
Worcester State College             No   2197    1515     543          4
Xavier University                  Yes   1959    1805     695         24
Xavier University of Louisiana     Yes   2097    1915     695         34
Yale University                    Yes  10705    2453    1317         95
York College of Pennsylvania       Yes   2989    1855     691         28

                                Top25perc  F.Undergrad  P.Undergrad  Outstate  \
Abilene Christian University           52         2885          537      7440
Adelphi University                     29         2683         1227     12280
Adrian College                         50         1036           99     11250
Agnes Scott College                    89          510           63     12960
Alaska Pacific University              44          249          869      7560
…                                     …           …           …        …
Worcester State College                26         3089         2029      6797
Xavier University                      47         2849         1107     11520
Xavier University of Louisiana         61         2793          166      6900
Yale University                        99         5217           83     19840
York College of Pennsylvania           63         2988         1726      4990

                                Room.Board  Books  Personal  PhD  Terminal  \
Abilene Christian University          3300    450      2200   70        78
```

```
Adelphi University                             6450     750       1500    29          30
Adrian College                                 3750     400       1165    53          66
Agnes Scott College                            5450     450        875    92          97
Alaska Pacific University                      4120     800       1500    76          72
…                                             …     …          …   …          …
Worcester State College                        3900     500       1200    60          60
Xavier University                              4960     600       1250    73          75
Xavier University of Louisiana                 4200     617        781    67          75
Yale University                                6510     630       2115    96          96
York College of Pennsylvania                   3560     500       1250    75          75

                                S.F.Ratio  perc.alumni  Expend  Grad.Rate
Abilene Christian University         18.1           12    7041         60
Adelphi University                   12.2           16   10527         56
Adrian College                       12.9           30    8735         54
Agnes Scott College                   7.7           37   19016         59
Alaska Pacific University            11.9            2   10922         15
…                                    …          …    …       …
Worcester State College              21.0           14    4469         40
Xavier University                    13.3           31    9189         83
Xavier University of Louisiana       14.4           20    8323         49
Yale University                       5.8           49   40386         99
York College of Pennsylvania         18.1           28    4509         99

[777 rows x 18 columns]
```

[5]:
```python
College3 = College.rename({"Unnamed: 0": "College"}, axis=1)
College3.set_index("College")
College3
```

[5]:
```
                            College Private   Apps  Accept  Enroll  Top10perc  \
0      Abilene Christian University     Yes   1660    1232     721         23
1                Adelphi University     Yes   2186    1924     512         16
2                    Adrian College     Yes   1428    1097     336         22
3               Agnes Scott College     Yes    417     349     137         60
4         Alaska Pacific University     Yes    193     146      55         16
..                              …     …    …     …     …        …
772         Worcester State College      No   2197    1515     543          4
773               Xavier University     Yes   1959    1805     695         24
774  Xavier University of Louisiana     Yes   2097    1915     695         34
775                 Yale University     Yes  10705    2453    1317         95
776    York College of Pennsylvania     Yes   2989    1855     691         28

     Top25perc  F.Undergrad  P.Undergrad  Outstate  Room.Board  Books  \
0           52         2885          537      7440        3300    450
1           29         2683         1227     12280        6450    750
2           50         1036           99     11250        3750    400
```

```
3           89        510          63     12960        5450    450
4           44        249         869      7560        4120    800
..         …         …          …       …          …     …
772         26       3089        2029      6797        3900    500
773         47       2849        1107     11520        4960    600
774         61       2793         166      6900        4200    617
775         99       5217          83     19840        6510    630
776         63       2988        1726      4990        3560    500

     Personal  PhD  Terminal  S.F.Ratio  perc.alumni  Expend  Grad.Rate
0        2200   70        78       18.1           12    7041         60
1        1500   29        30       12.2           16   10527         56
2        1165   53        66       12.9           30    8735         54
3         875   92        97        7.7           37   19016         59
4        1500   76        72       11.9            2   10922         15
..        …  …        …         …          …     …      …
772      1200   60        60       21.0           14    4469         40
773      1250   73        75       13.3           31    9189         83
774       781   67        75       14.4           20    8323         49
775      2115   96        96        5.8           49   40386         99
776      1250   75        75       18.1           28    4509         99

[777 rows x 19 columns]
```

[6]:
```
College = College3
```

[6]:
```
                          College Private   Apps  Accept  Enroll  Top10perc  \
0    Abilene Christian University     Yes   1660    1232     721         23
1              Adelphi University     Yes   2186    1924     512         16
2                  Adrian College     Yes   1428    1097     336         22
3             Agnes Scott College     Yes    417     349     137         60
4       Alaska Pacific University     Yes    193     146      55         16
..                            …     …    …     …     …       …
772        Worcester State College      No   2197    1515     543          4
773             Xavier University     Yes   1959    1805     695         24
774  Xavier University of Louisiana     Yes   2097    1915     695         34
775               Yale University     Yes  10705    2453    1317         95
776    York College of Pennsylvania     Yes   2989    1855     691         28

     Top25perc  F.Undergrad  P.Undergrad  Outstate  Room.Board  Books  \
0           52         2885          537      7440        3300    450
1           29         2683         1227     12280        6450    750
2           50         1036           99     11250        3750    400
3           89          510           63     12960        5450    450
4           44          249          869      7560        4120    800
..         …         …          …       …          …     …
772         26         3089         2029      6797        3900    500
```

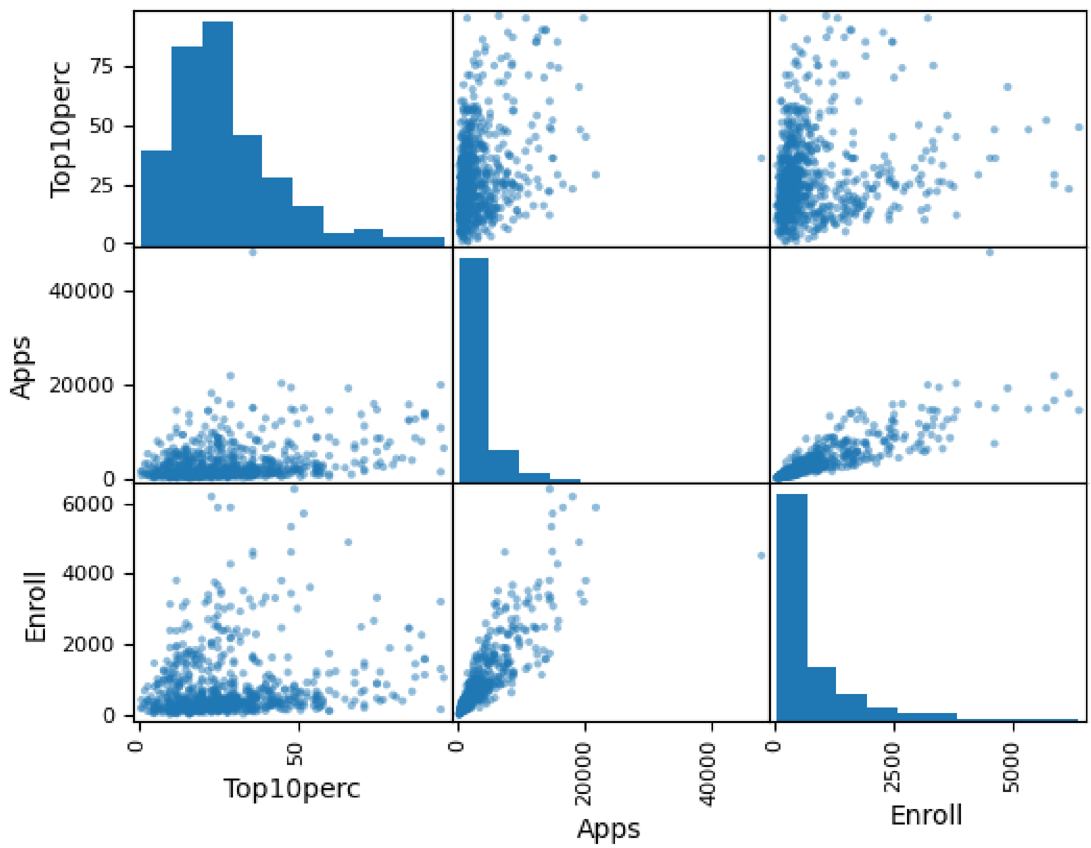
```
773          47         2849         1107      11520         4960     600
774          61         2793          166       6900         4200     617
775          99         5217           83      19840         6510     630
776          63         2988         1726       4990         3560     500

     Personal  PhD  Terminal  S.F.Ratio  perc.alumni  Expend  Grad.Rate
0        2200   70        78       18.1           12    7041         60
1        1500   29        30       12.2           16   10527         56
2        1165   53        66       12.9           30    8735         54
3         875   92        97        7.7           37   19016         59
4        1500   76        72       11.9            2   10922         15
..        …   …       …        …          …     …        …
772      1200   60        60       21.0           14    4469         40
773      1250   73        75       13.3           31    9189         83
774       781   67        75       14.4           20    8323         49
775      2115   96        96        5.8           49   40386         99
776      1250   75        75       18.1           28    4509         99

[777 rows x 19 columns]
```

[7]:
```python
College.describe()
```

[7]:
```
               Apps        Accept       Enroll    Top10perc    Top25perc  \
count    777.000000    777.000000   777.000000   777.000000   777.000000
mean    3001.638353   2018.804376   779.972973    27.558559    55.796654
std     3870.201484   2451.113971   929.176190    17.640364    19.804778
min       81.000000     72.000000    35.000000     1.000000     9.000000
25%      776.000000    604.000000   242.000000    15.000000    41.000000
50%     1558.000000   1110.000000   434.000000    23.000000    54.000000
75%     3624.000000   2424.000000   902.000000    35.000000    69.000000
max    48094.000000  26330.000000  6392.000000    96.000000   100.000000

        F.Undergrad   P.Undergrad      Outstate   Room.Board        Books  \
count    777.000000    777.000000    777.000000   777.000000   777.000000
mean    3699.907336    855.298584  10440.669241  4357.526384   549.380952
std     4850.420531   1522.431887   4023.016484  1096.696416   165.105360
min      139.000000      1.000000   2340.000000  1780.000000    96.000000
25%      992.000000     95.000000   7320.000000  3597.000000   470.000000
50%     1707.000000    353.000000   9990.000000  4200.000000   500.000000
75%     4005.000000    967.000000  12925.000000  5050.000000   600.000000
max    31643.000000  21836.000000  21700.000000  8124.000000  2340.000000

          Personal         PhD    Terminal   S.F.Ratio  perc.alumni  \
count   777.000000  777.000000  777.000000  777.000000   777.000000
mean   1340.642214   72.660232   79.702703   14.089704    22.743887
std     677.071454   16.328155   14.722359    3.958349    12.391801
min     250.000000    8.000000   24.000000    2.500000     0.000000
```

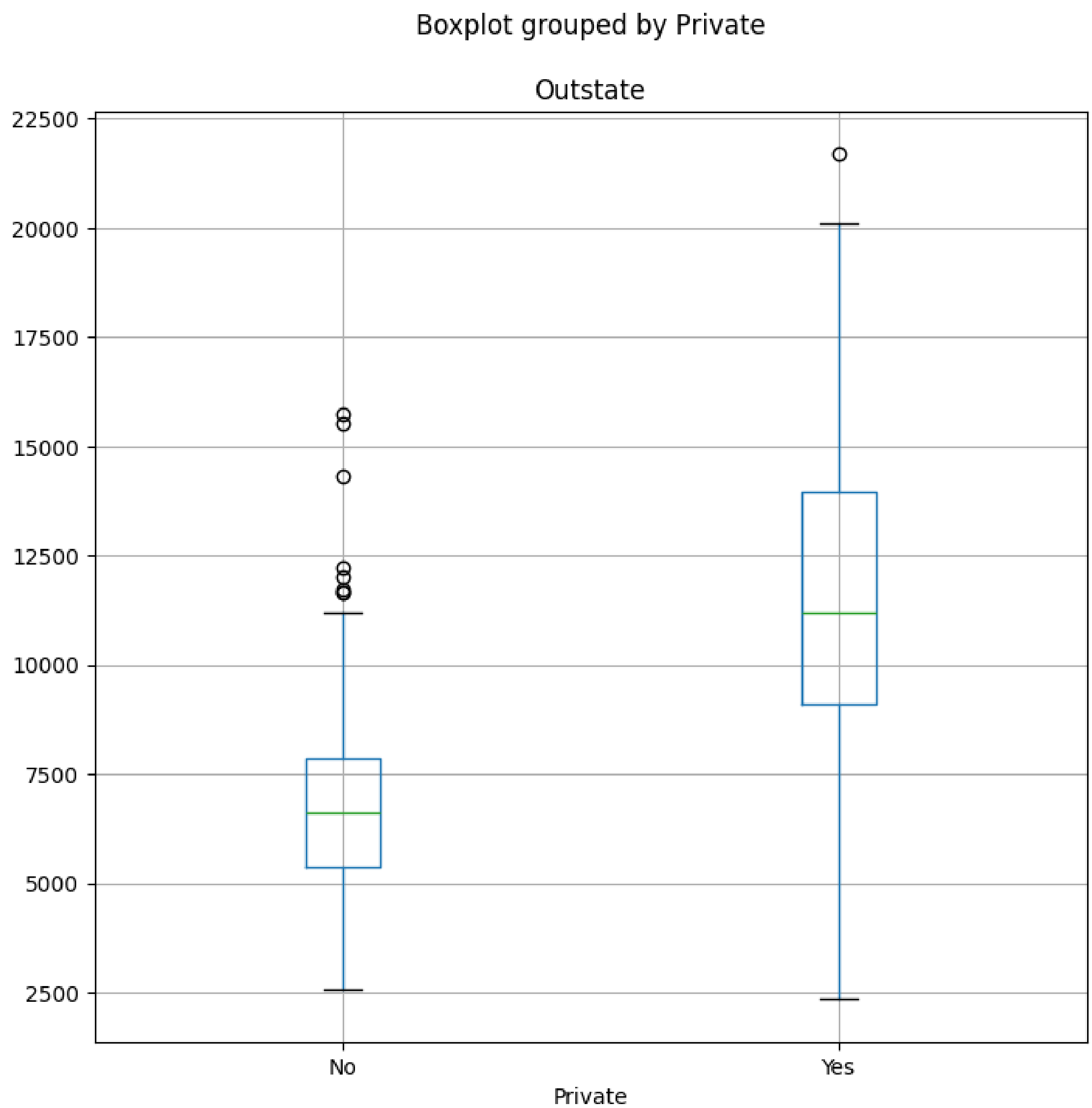
```
25%      850.000000    62.000000    71.000000    11.500000    13.000000
50%     1200.000000    75.000000    82.000000    13.600000    21.000000
75%     1700.000000    85.000000    92.000000    16.500000    31.000000
max     6800.000000   103.000000   100.000000    39.800000    64.000000

             Expend  Grad.Rate
count    777.000000  777.00000
mean    9660.171171   65.46332
std     5221.768440   17.17771
min     3186.000000   10.00000
25%     6751.000000   53.00000
50%     8377.000000   65.00000
75%    10830.000000   78.00000
max    56233.000000  118.00000
```

[8]:
```
pd.plotting.scatter_matrix(College[["Top10perc", "Apps", "Enroll"]]);
```



[9]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("Outstate", by="Private", ax=ax);
```

```
Executing &lt;Handle IOLoop._run_callback(functools.par…7dd256f2f600>)) created
at /home/linus/ISLP/islpenv/lib/python3.12/site-
packages/tornado/platform/asyncio.py:235> took 0.122 seconds
IOStream.flush timed out
Executing <Handle BaseAsyncIOLoop._handle_events(28, 1) created at
/usr/lib/python3.12/asyncio/selector_events.py:280> took 0.134 seconds
```



[10]:
```python
College["Top10perc"]
```

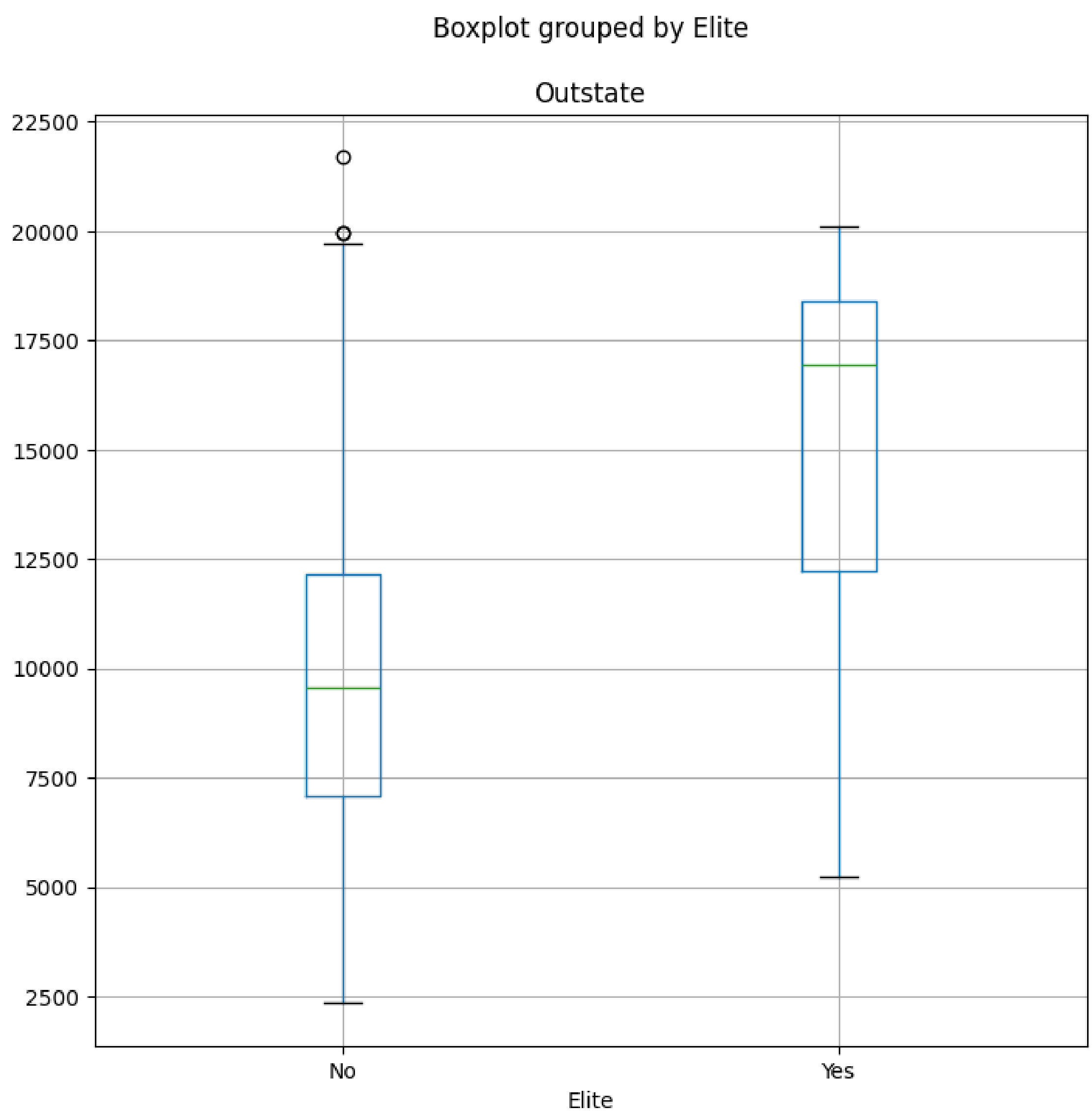
[10]:
```
0      23
1      16
2      22
```

```
3      60
4      16
       ..
772     4
773    24
774    34
775    95
776    28
Name: Top10perc, Length: 777, dtype: int64
```

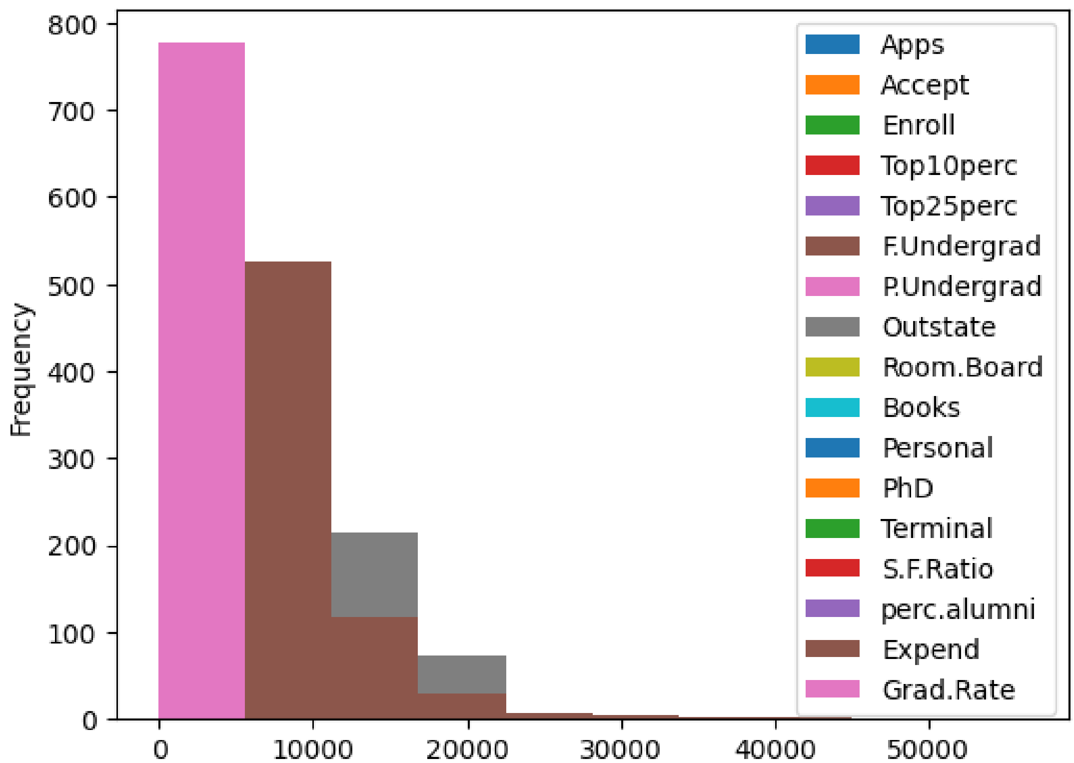
[11]:
```
College["Elite"] = pd.cut(College["Top10perc"], [0, 50, 100], labels=["No",␣
↪"Yes"])
College["Elite"].value_counts()
```

[11]:
```
Elite
No     699
Yes     78
Name: count, dtype: int64
```

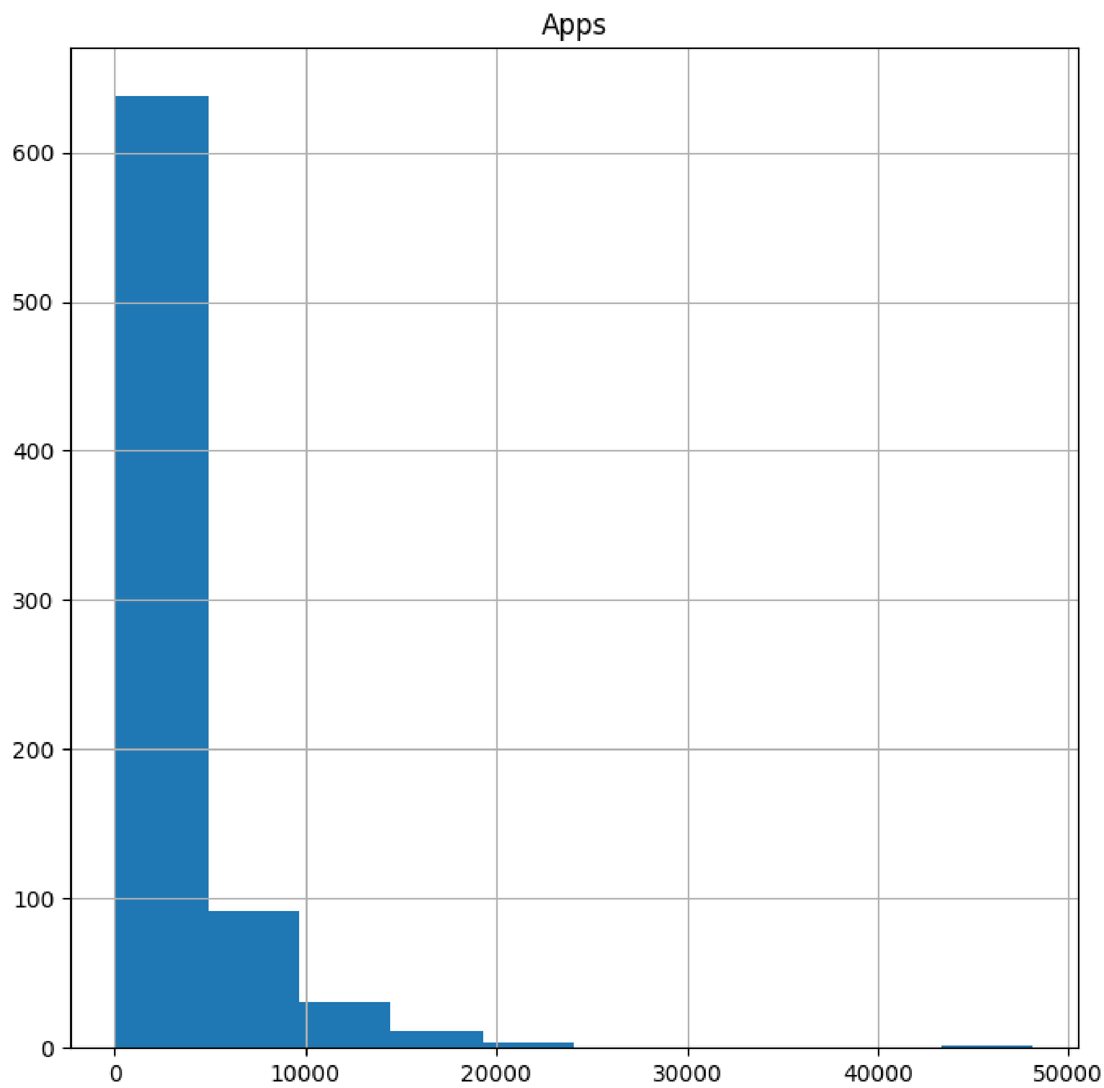
[12]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("Outstate", by="Elite", ax=ax);
```



[13]:
```
College.plot.hist();
```



[14]:
```
fig, ax = subplots(figsize=(8, 8))
College.hist("Apps", ax=ax);
```

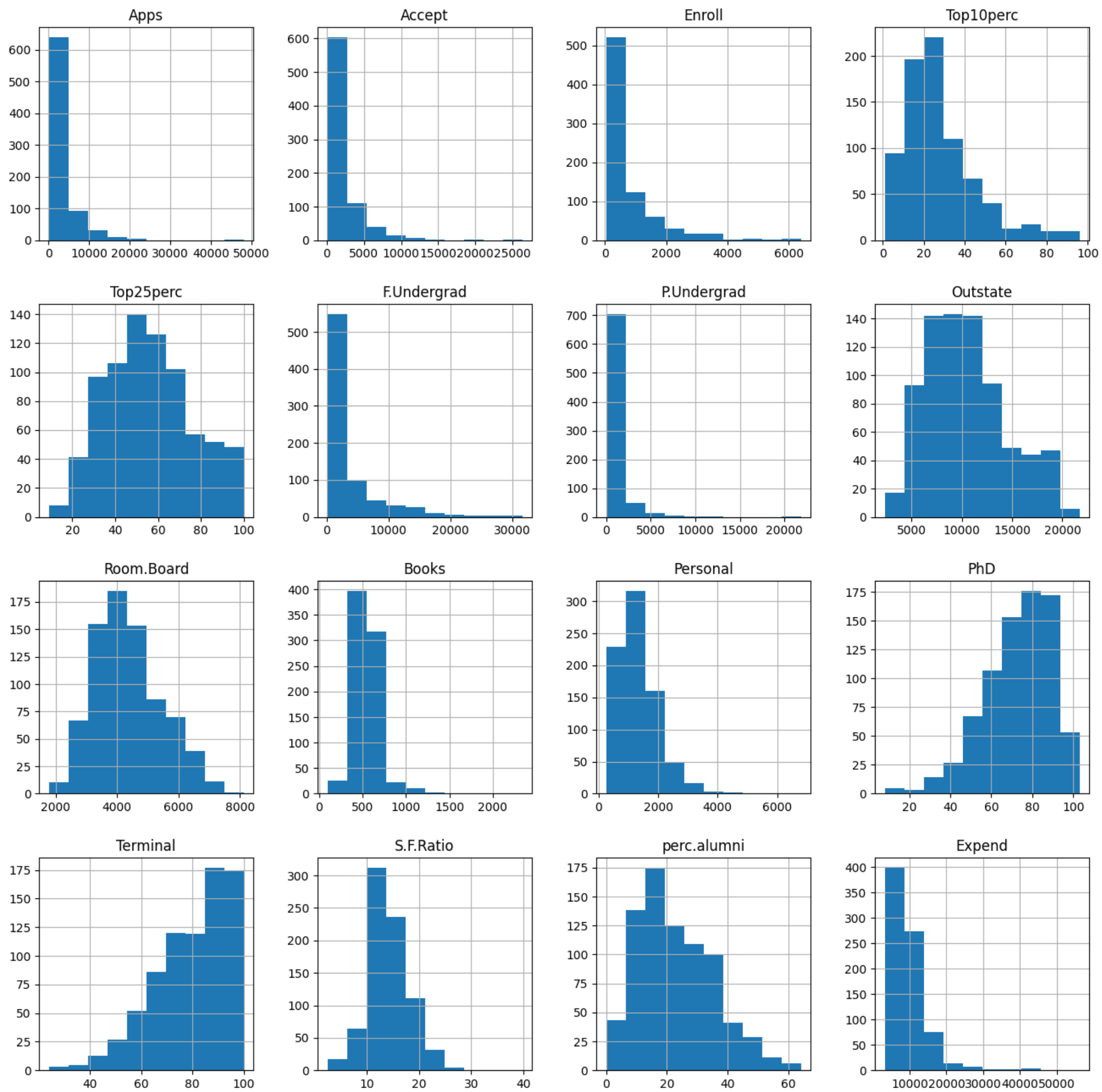


[15]:
```
numeric_columns = College.select_dtypes(include="number").columns.tolist()
numeric_columns
```

[15]:
```
['Apps',
 'Accept',
 'Enroll',
 'Top10perc',
 'Top25perc',
 'F.Undergrad',
 'P.Undergrad',
 'Outstate',
 'Room.Board',
 'Books',
```

```
'Personal',
'PhD',
'Terminal',
'S.F.Ratio',
'perc.alumni',
'Expend',
'Grad.Rate']
```

[16]:
```
fig, axs = subplots(4, 4, figsize=(16, 16))
for row in range(0, 4):
    for column in range(0, 4):
        College.hist(numeric_columns[row * 4 + column], ax=axs[row, column])
```



0.0.1 Count of private and public colleges

[17]:
```python
College["Private"].value_counts()
```

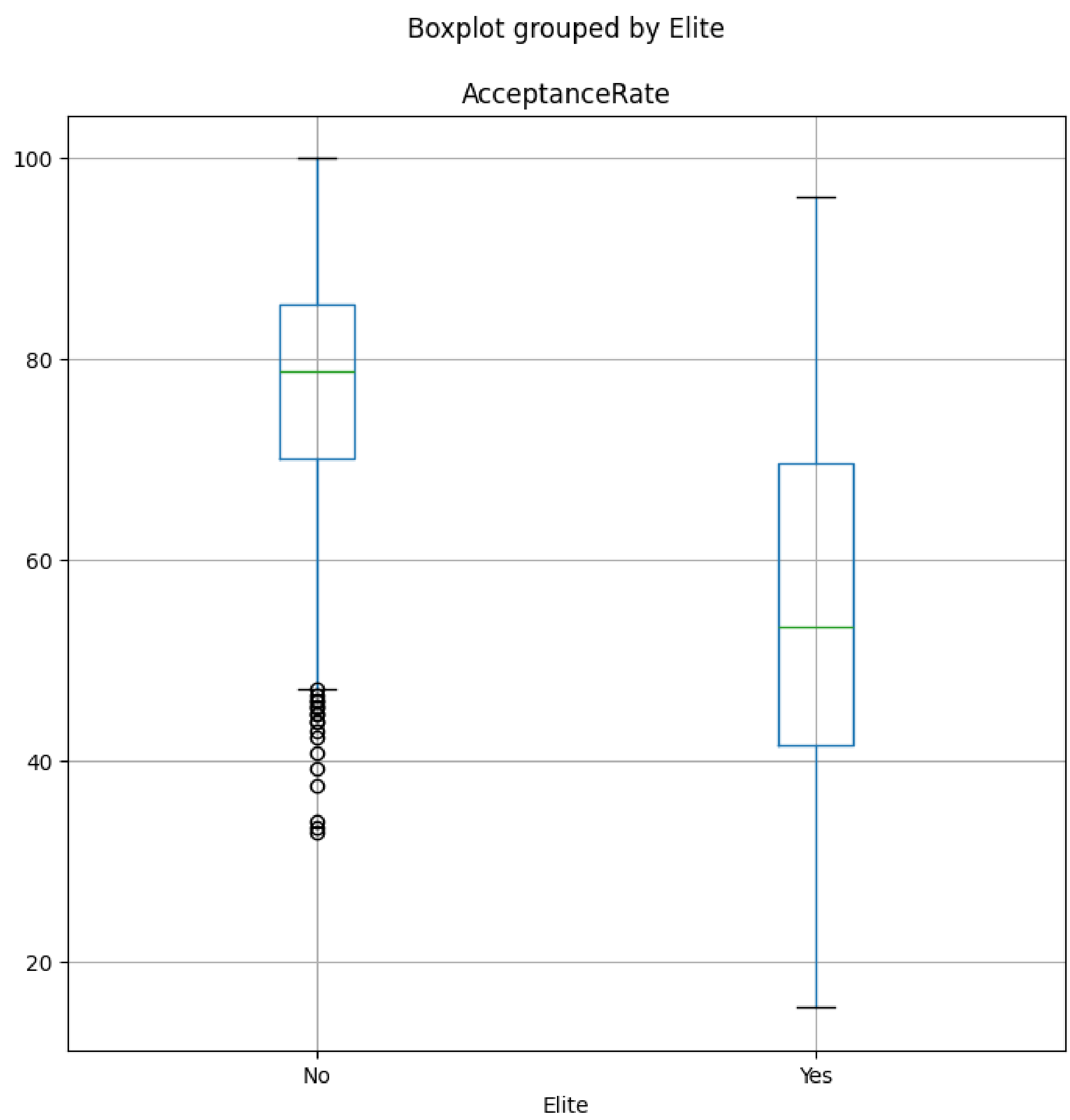
[17]:
```
Private
Yes    565
No     212
Name: count, dtype: int64
```

[18]:
```python
College["AcceptanceRate"] = round(College["Accept"] / College["Apps"] * 100, 2)
College["AcceptanceRate"]
```

[18]:
```
0      74.22
1      88.01
2      76.82
3      83.69
4      75.65
       …
772    68.96
773    92.14
774    91.32
775    22.91
776    62.06
Name: AcceptanceRate, Length: 777, dtype: float64
```

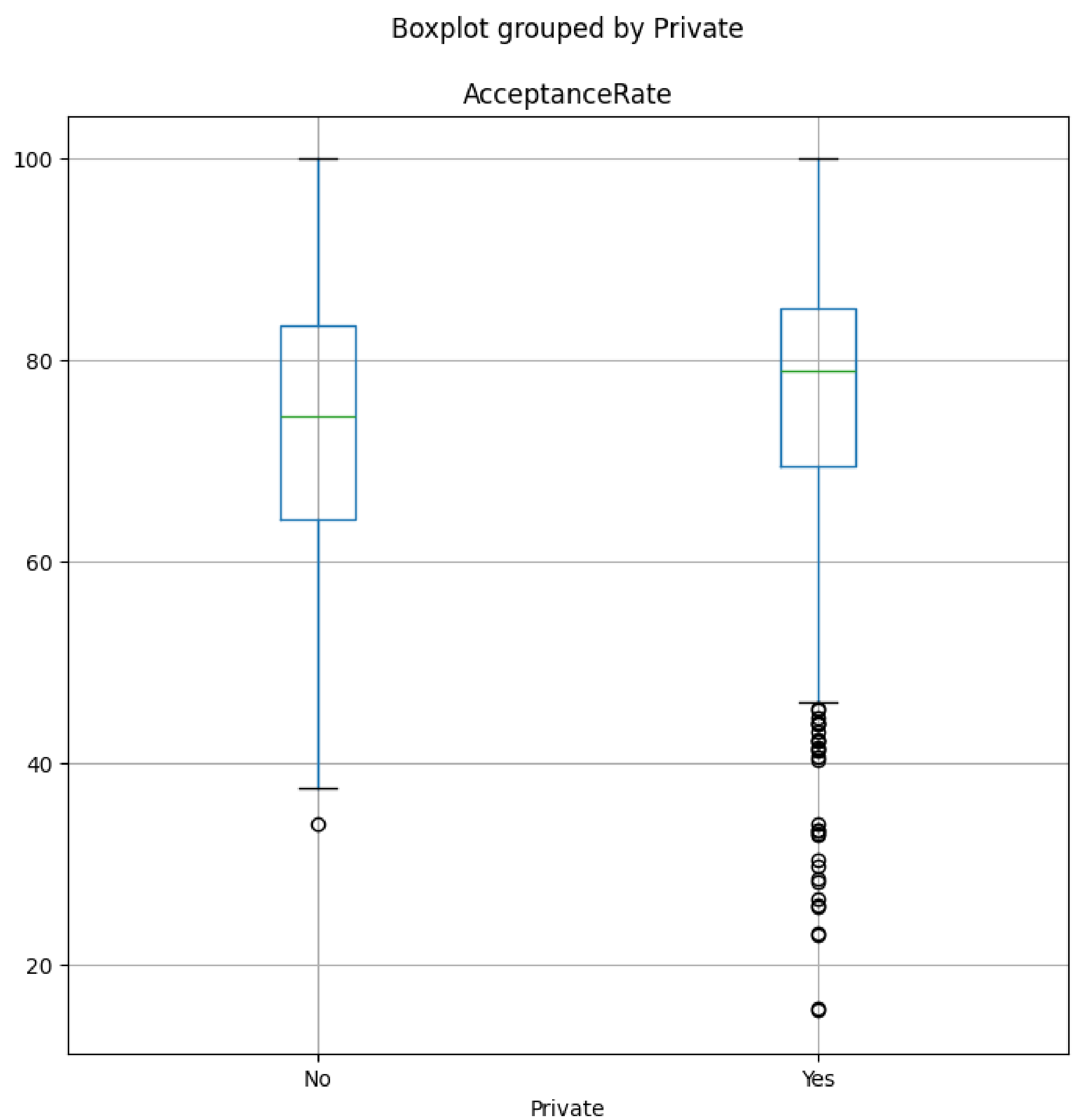
[19]:
```python
### Plot boxplot for acceptance rate by College Type : Elite or not
```

[20]:
```python
fig, ax = subplots(figsize=(8, 8))
College.boxplot("AcceptanceRate", by="Elite", ax=ax);
```




```
[21]: ### Plot boxplot for acceptance rate for Private colleges or not
```

```
[22]: fig, ax = subplots(figsize=(8, 8))
      College.boxplot("AcceptanceRate", by="Private", ax=ax);
```

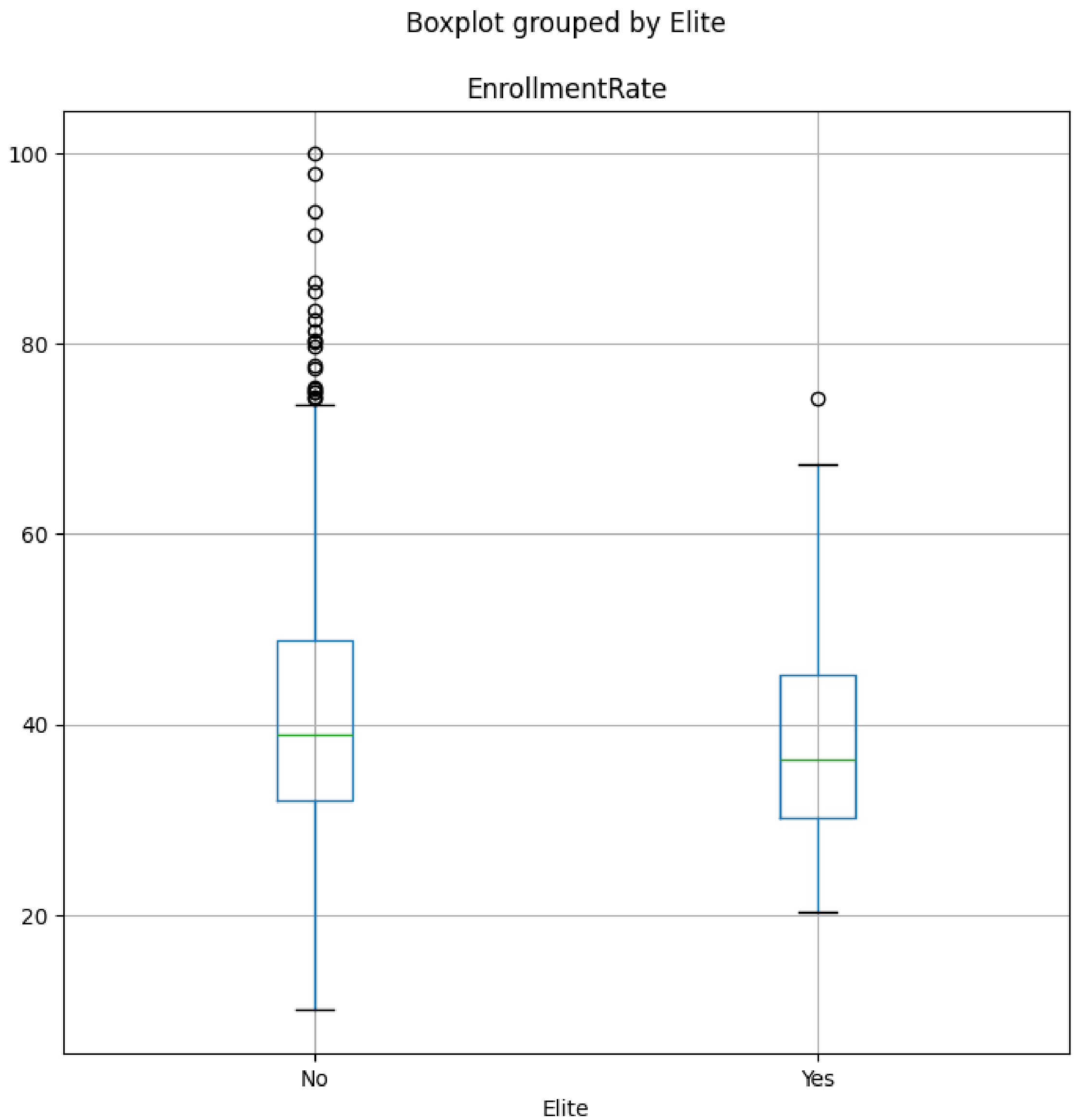


[23]:
```
College["EnrollmentRate"] = round(College["Enroll"] / College["Accept"] * 100,␣
↪2)
College["EnrollmentRate"]
```

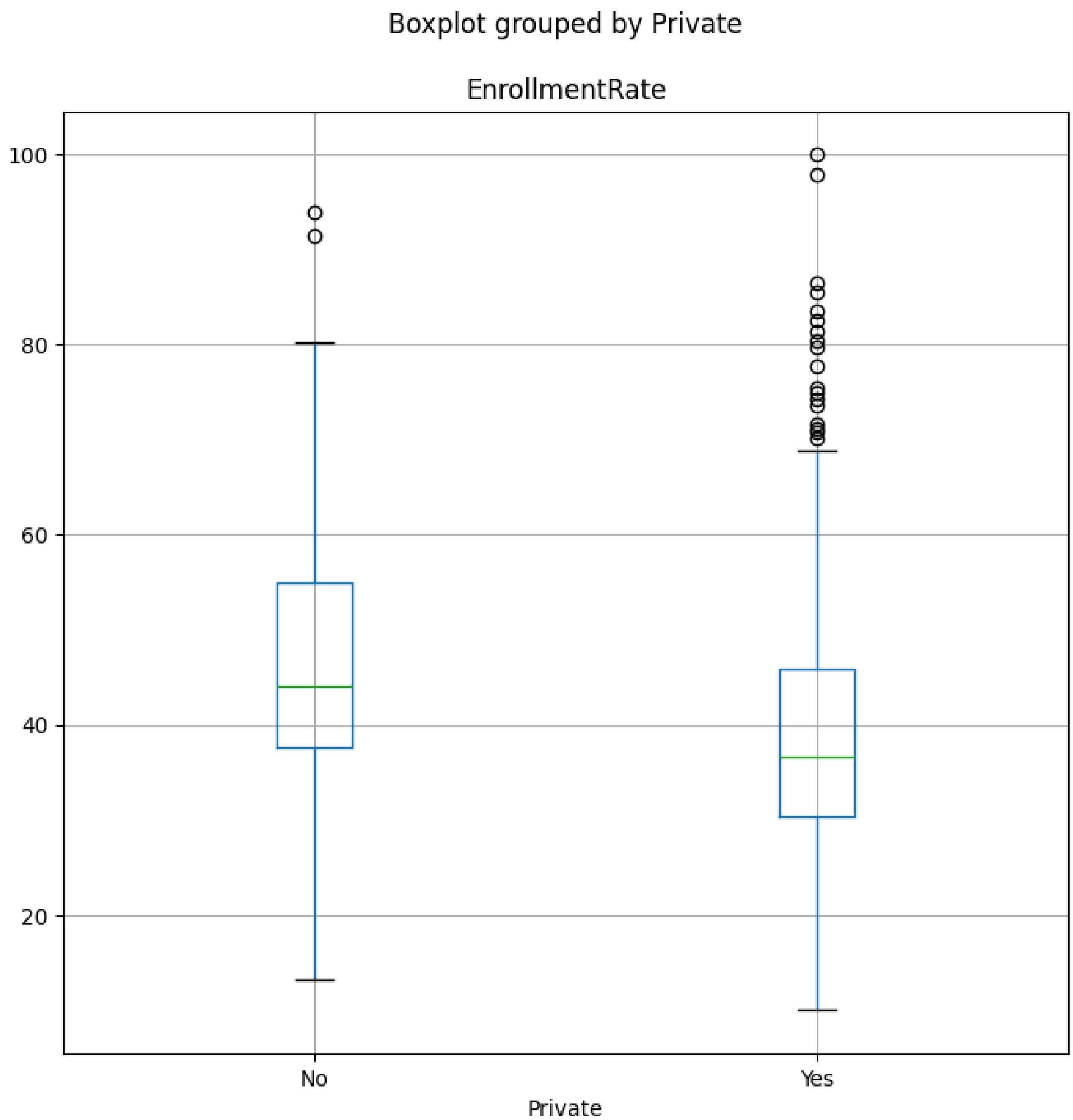
[23]:
```
0      58.52
1      26.61
2      30.63
3      39.26
4      37.67
       …
772    35.84
```

```
773    38.50
774    36.29
775    53.69
776    37.25
Name: EnrollmentRate, Length: 777, dtype: float64
```

[24]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("EnrollmentRate", by="Elite", ax=ax);
```

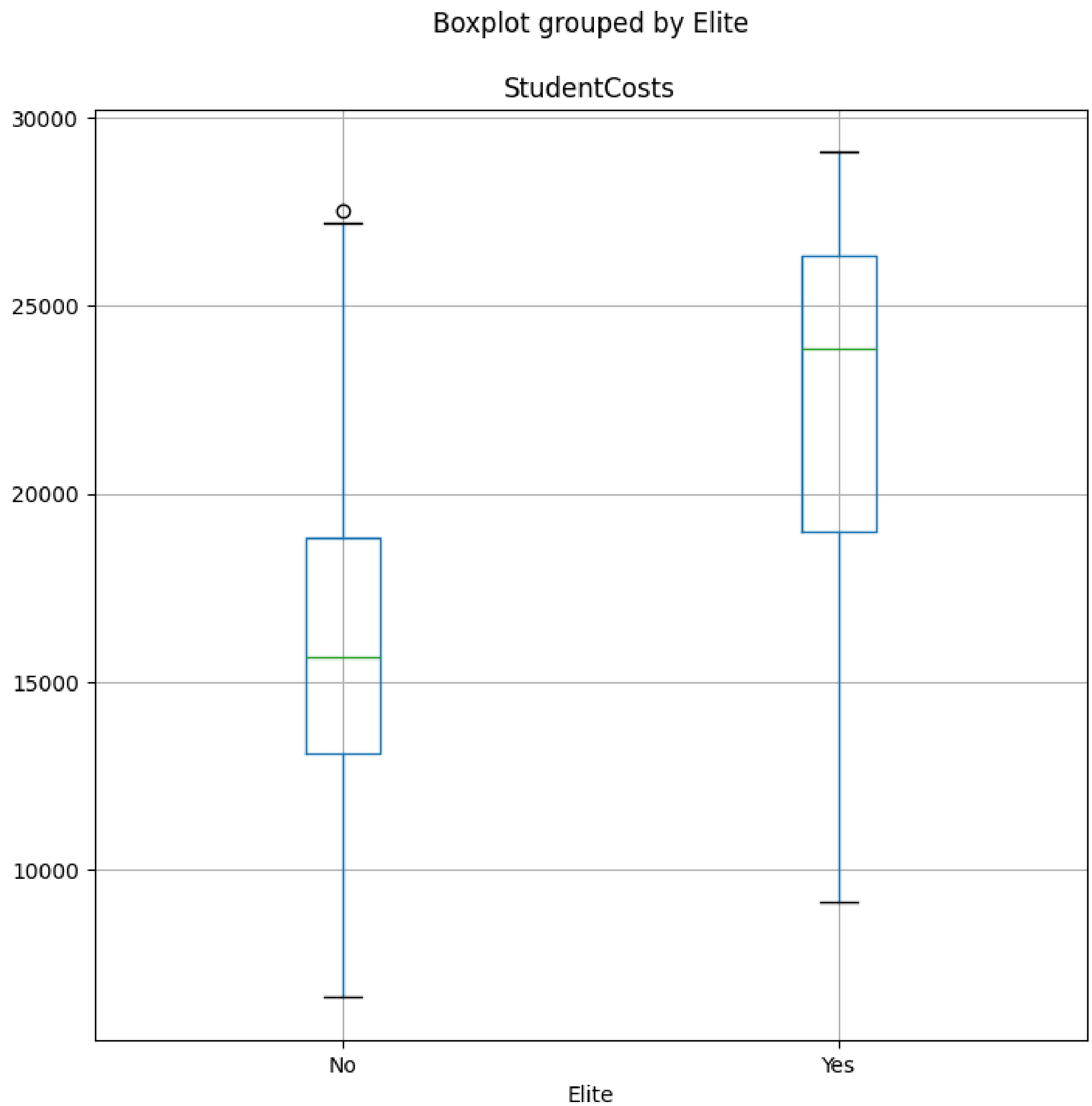


[25]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("EnrollmentRate", by="Private", ax=ax);
```

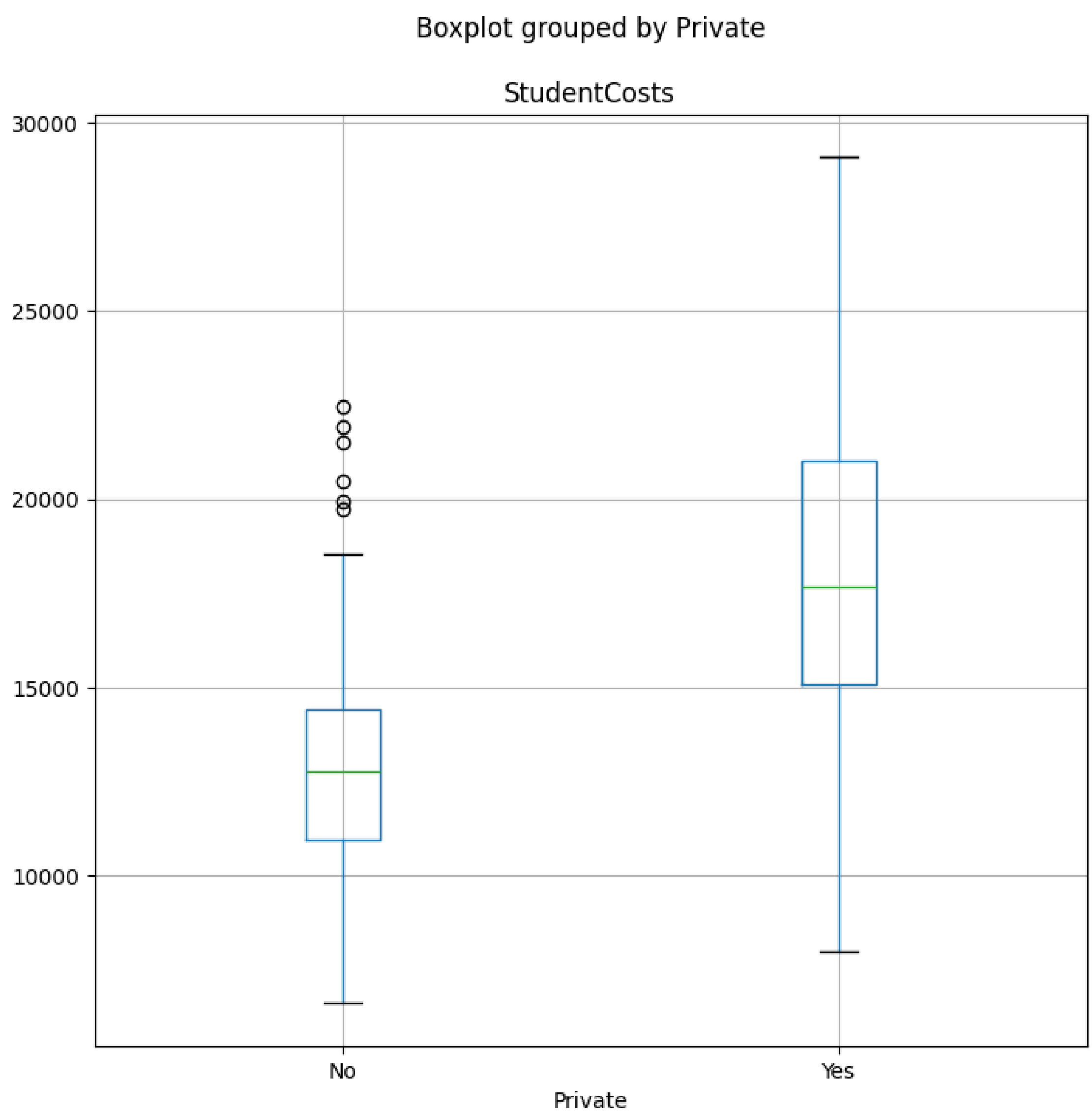


[26]:
```
College["StudentCosts"] = (
    College["Outstate"] + College["Room.Board"] + College["Books"] +␣
 ↪College["Personal"]
)
```

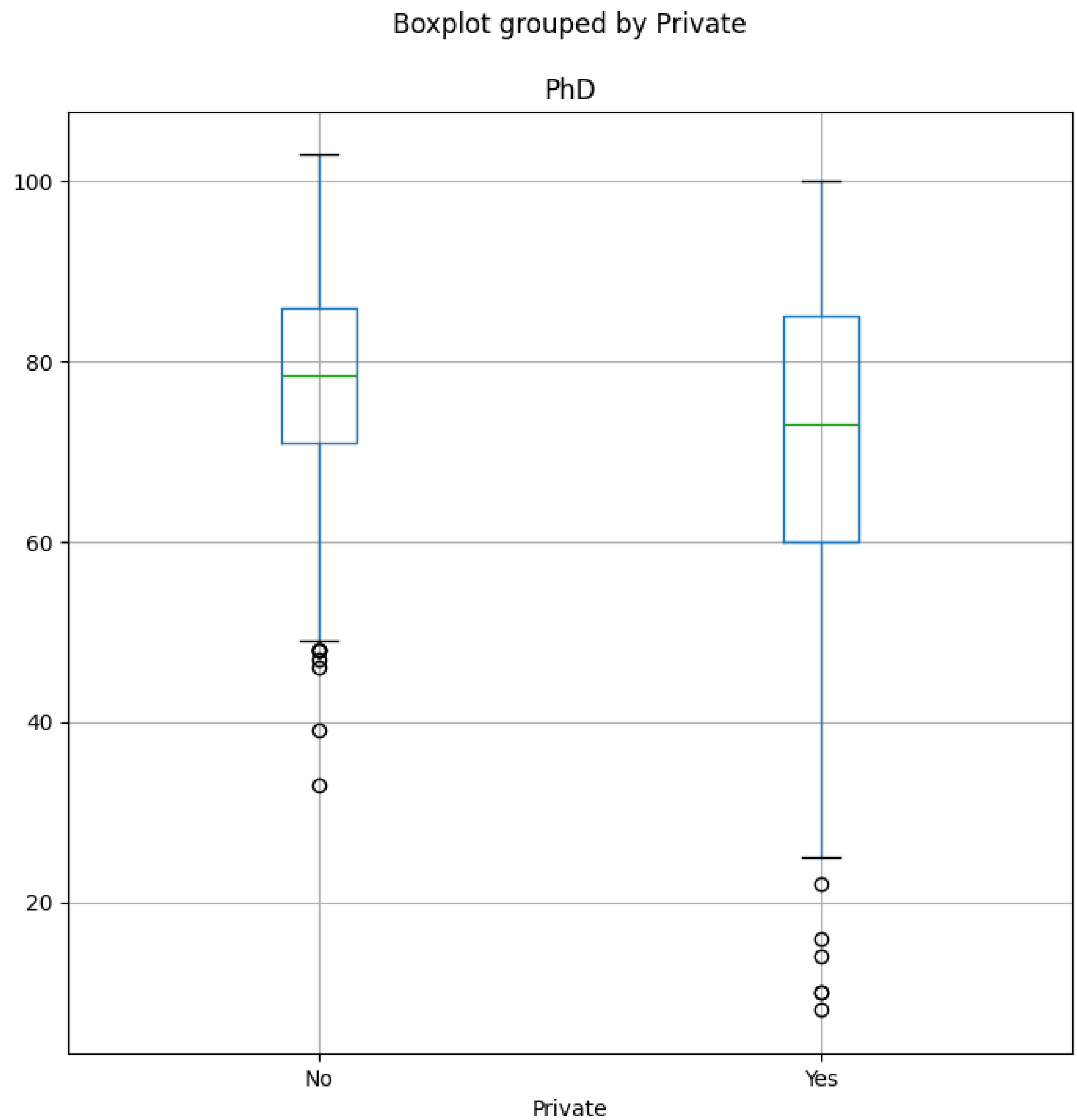
[27]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("StudentCosts", by="Elite", ax=ax);
```



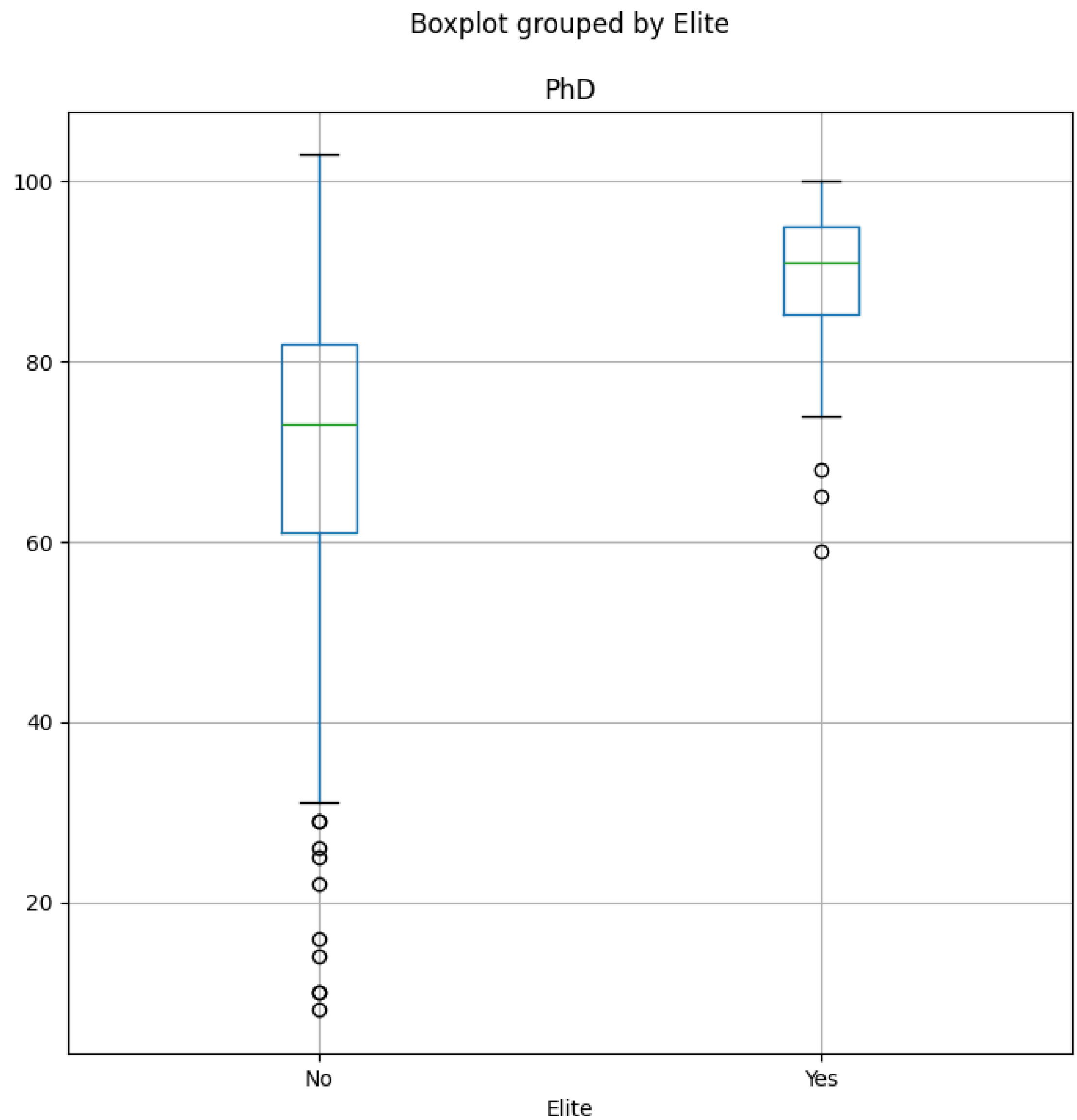
[28]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("StudentCosts", by="Private", ax=ax);
```



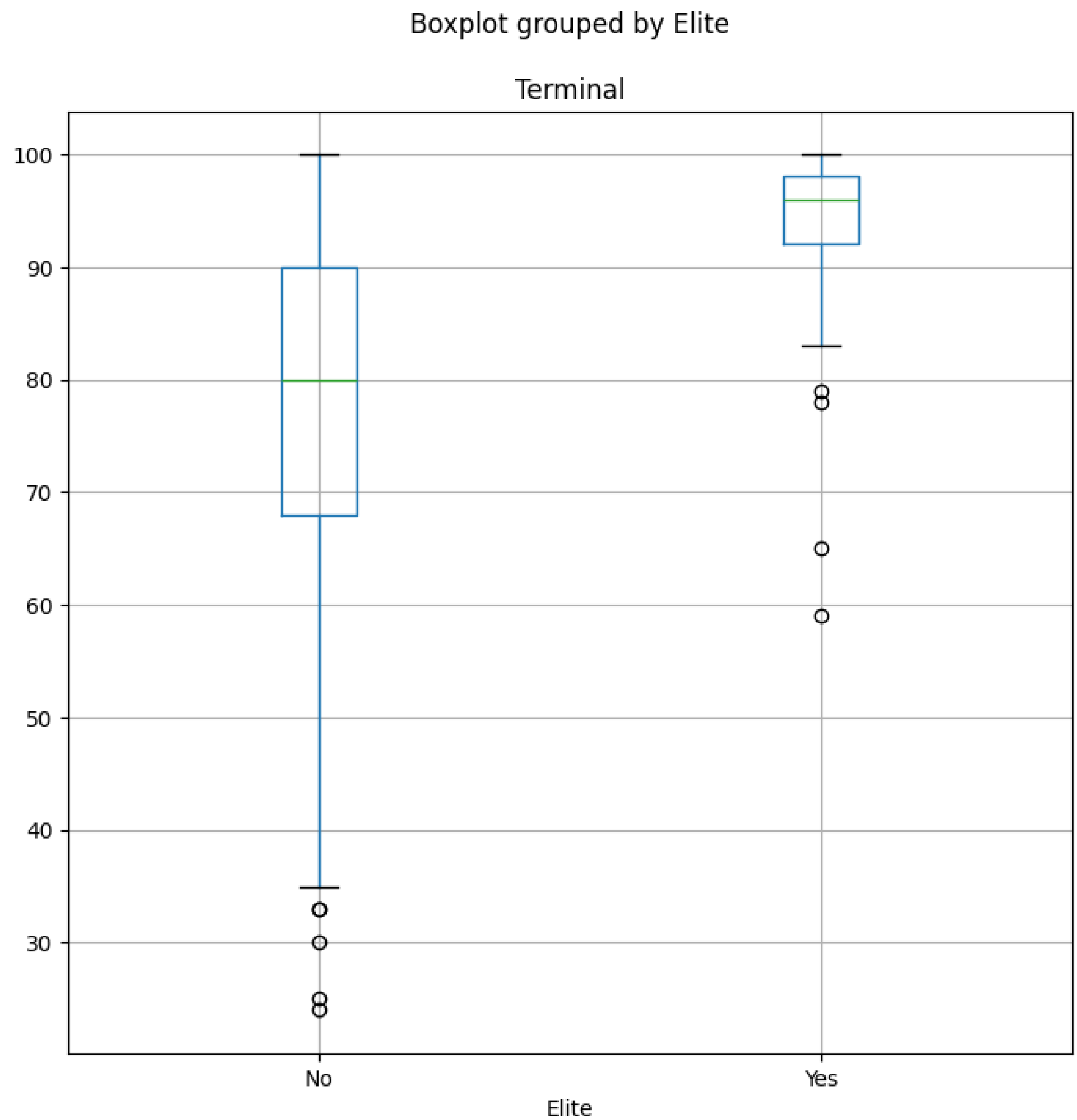
[29]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("PhD", by="Private", ax=ax);
```



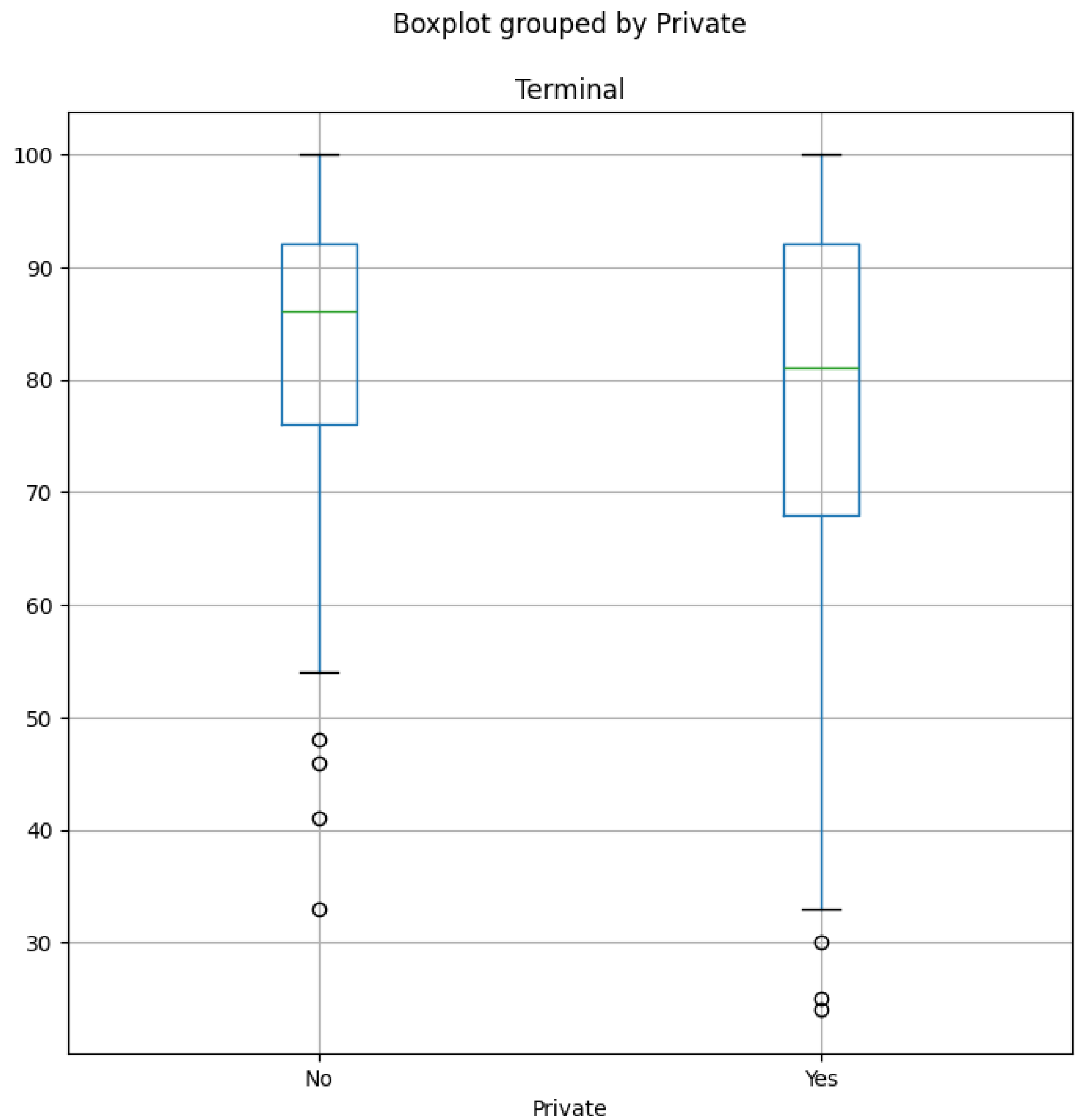
[30]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("PhD", by="Elite", ax=ax);
```



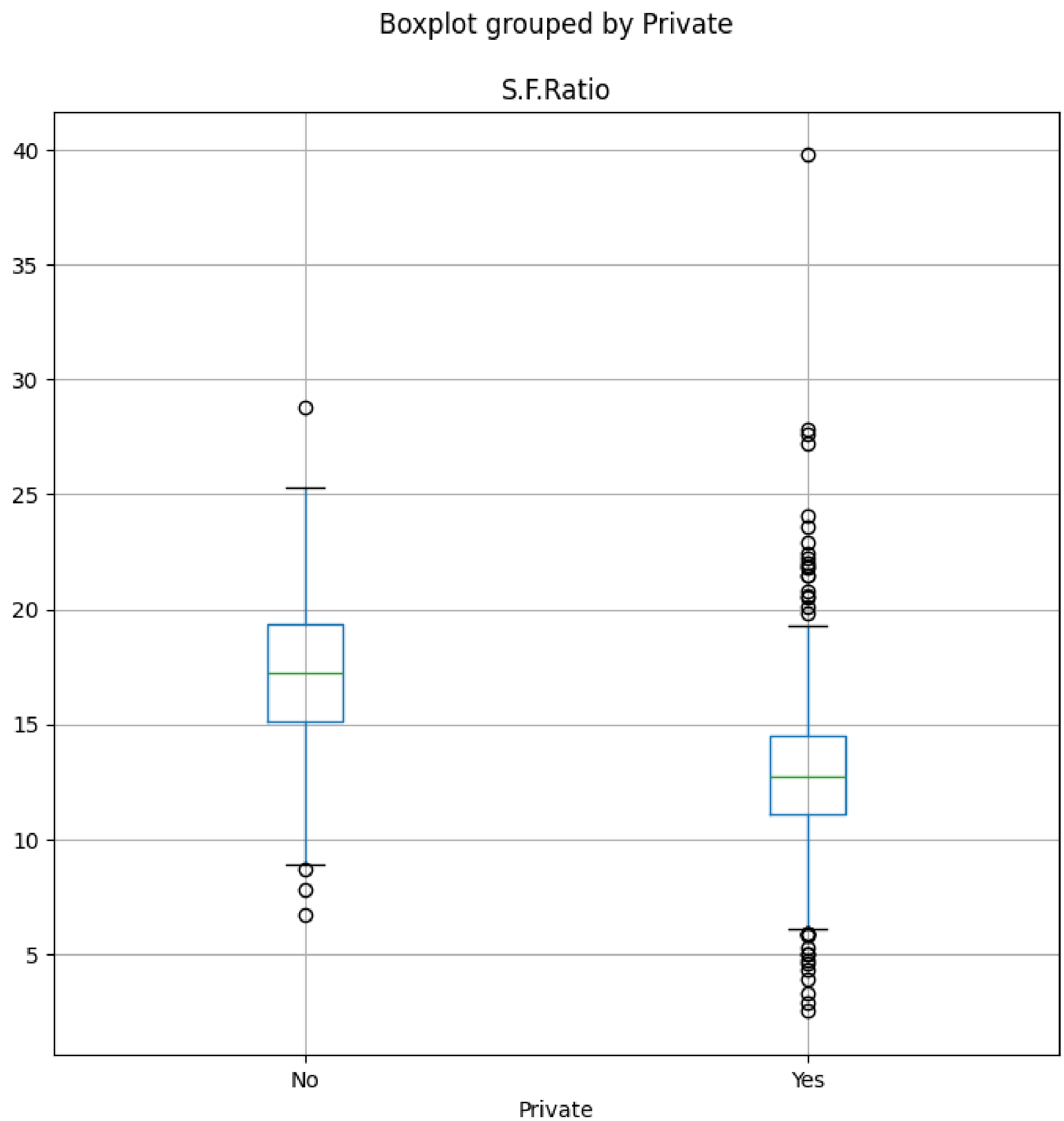
[31]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("Terminal", by="Elite", ax=ax);
```



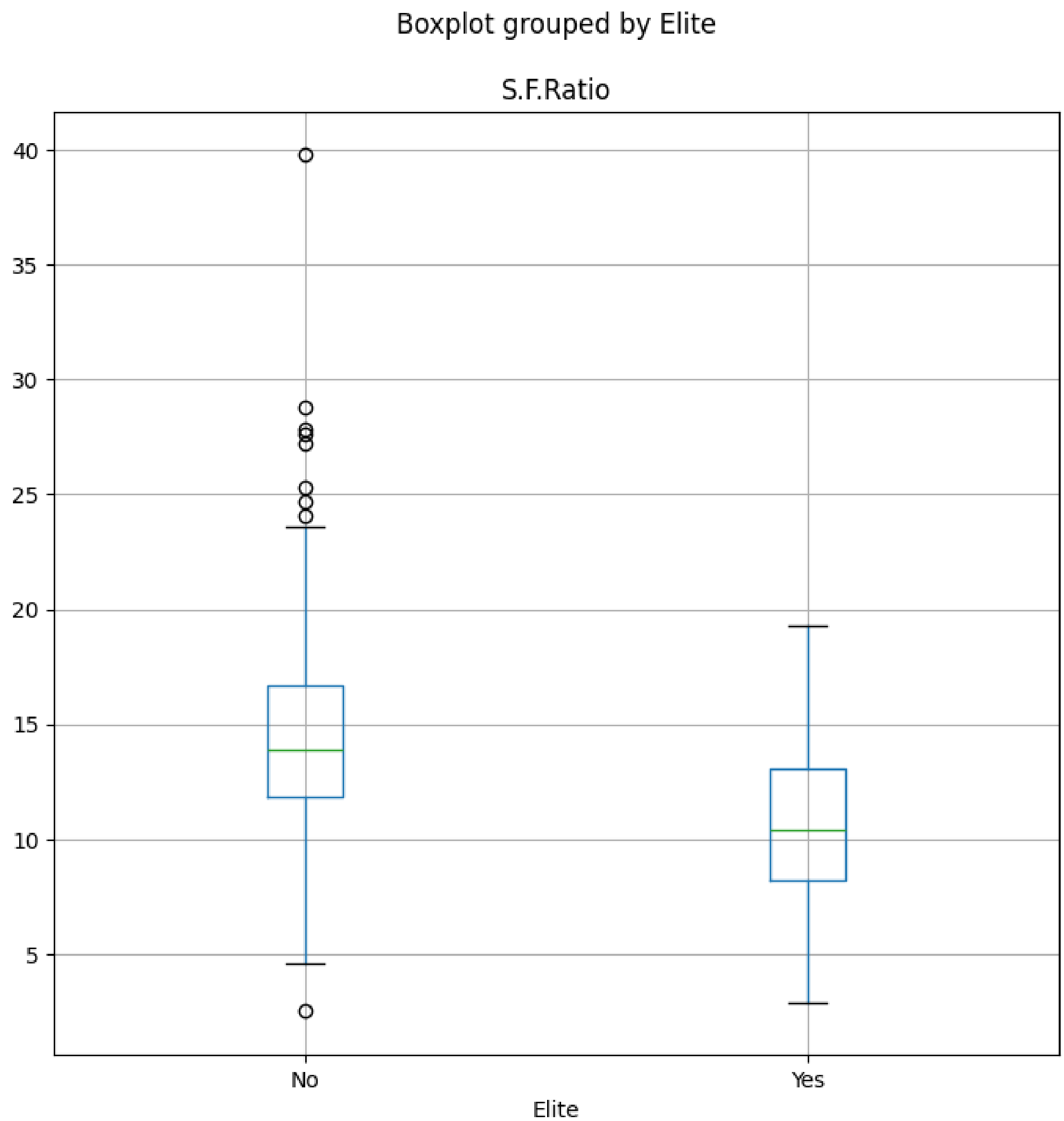
[32]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("Terminal", by="Private", ax=ax);
```



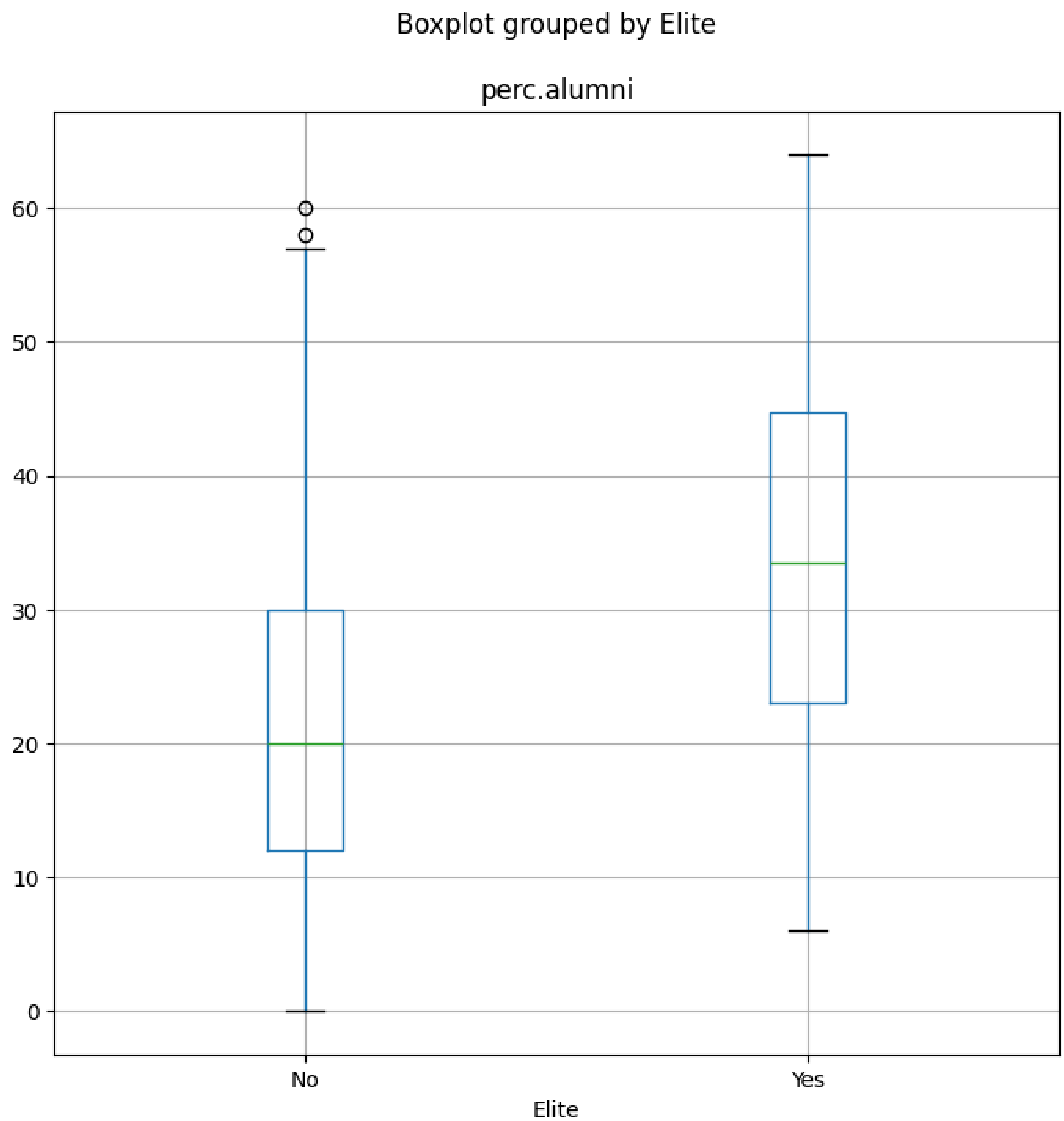
[33]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("S.F.Ratio", by="Private", ax=ax);
```



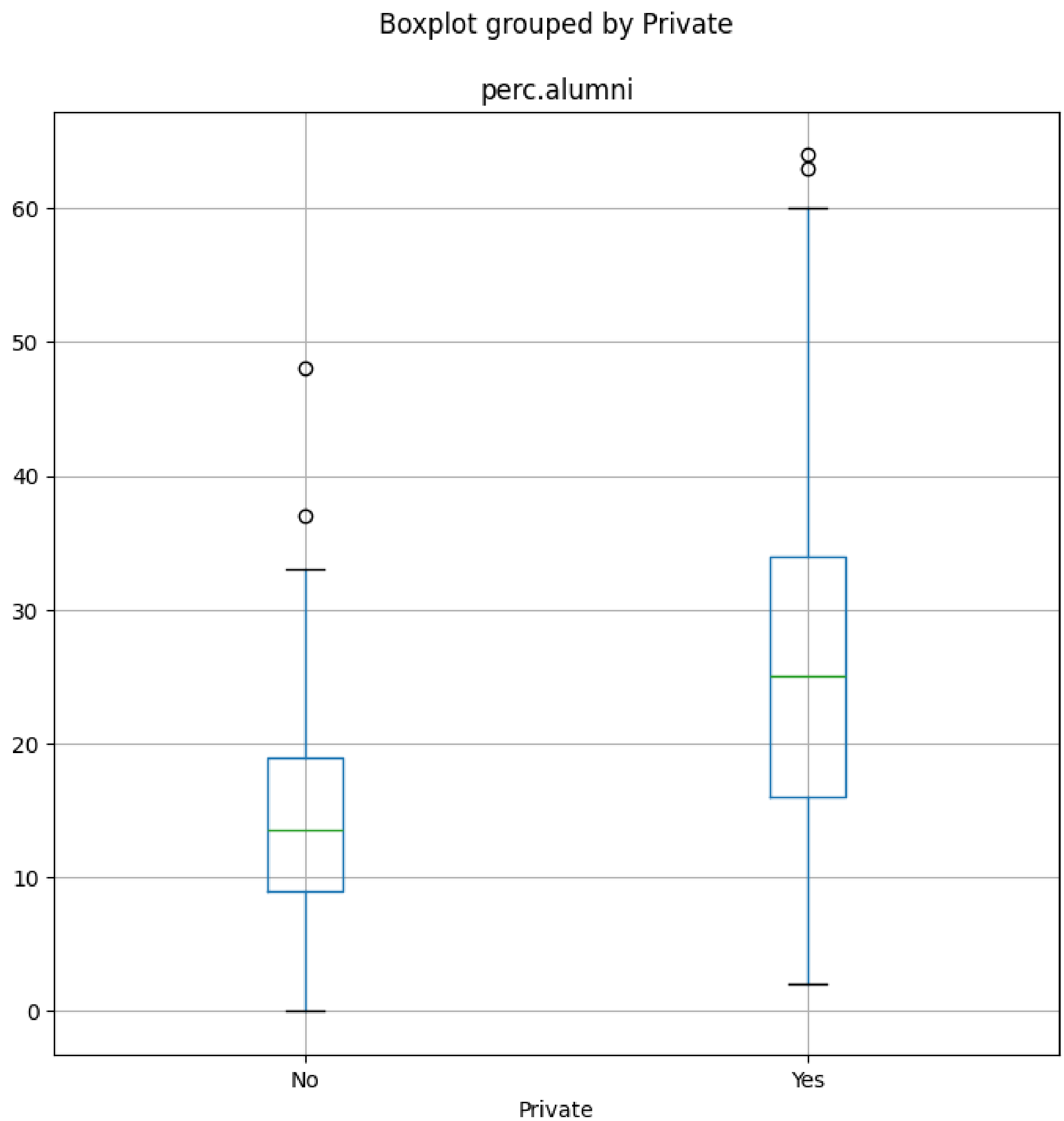
[34]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("S.F.Ratio", by="Elite", ax=ax);
```



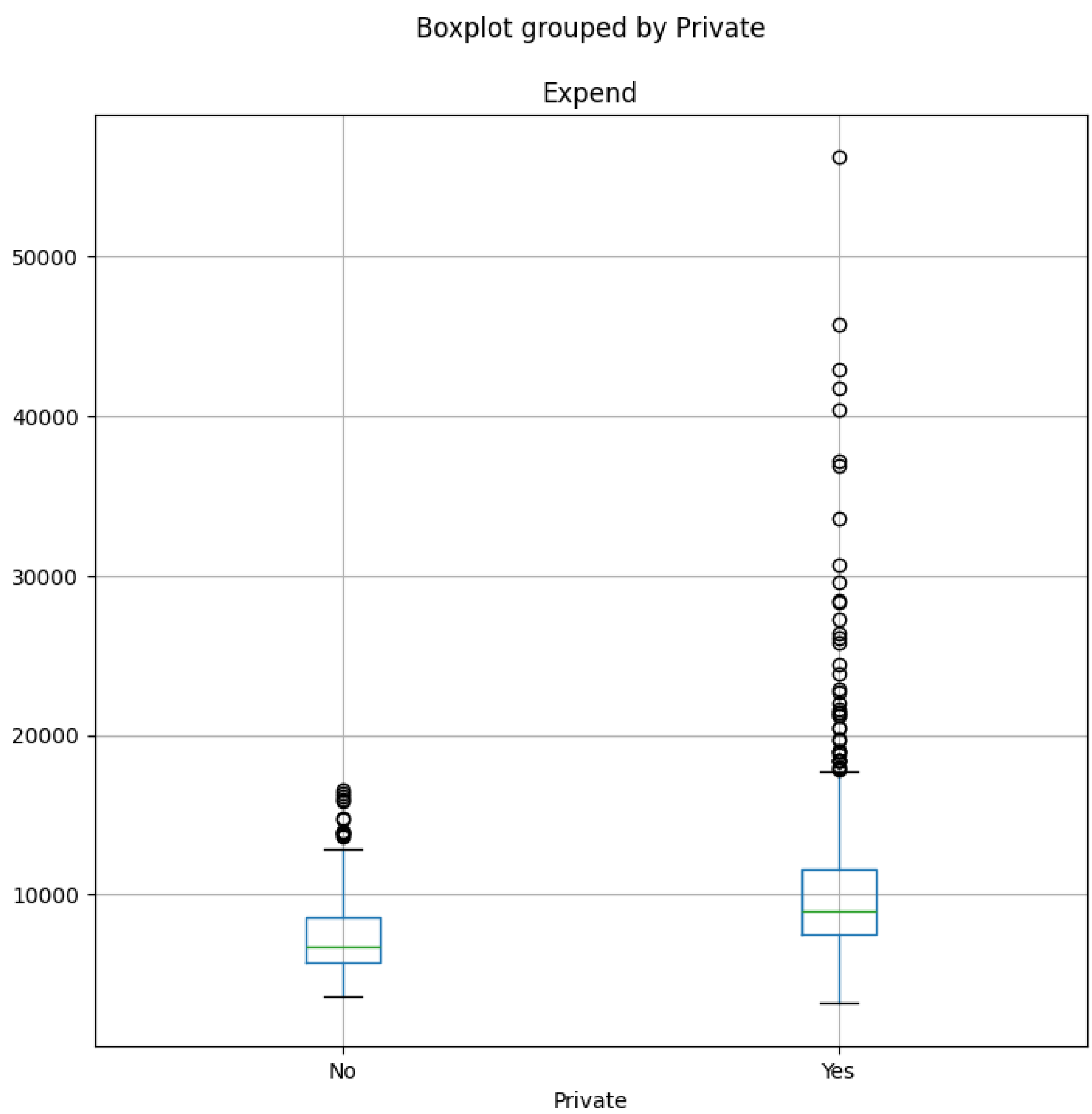
[35]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("perc.alumni", by="Elite", ax=ax);
```



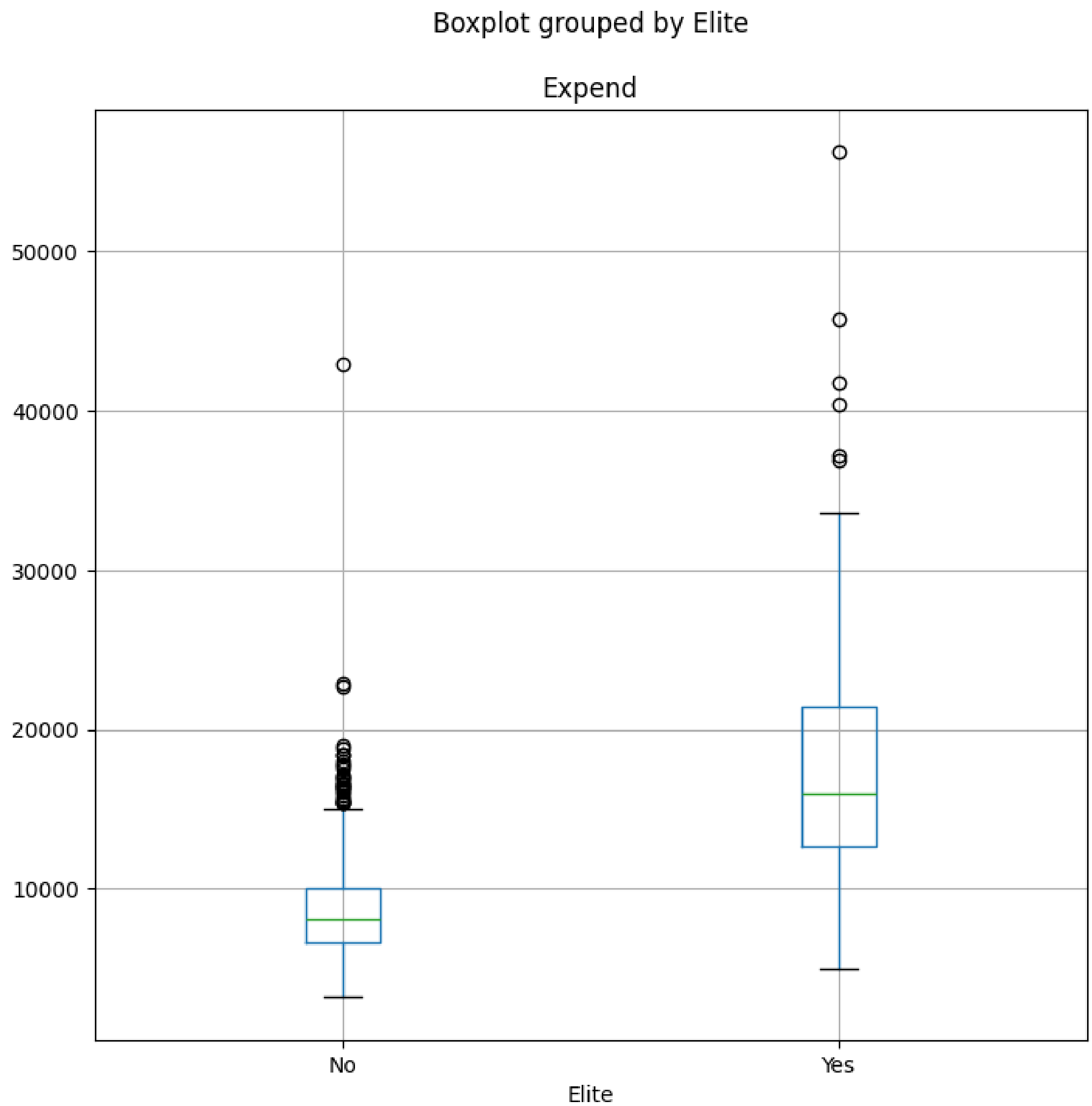
[36]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("perc.alumni", by="Private", ax=ax);
```



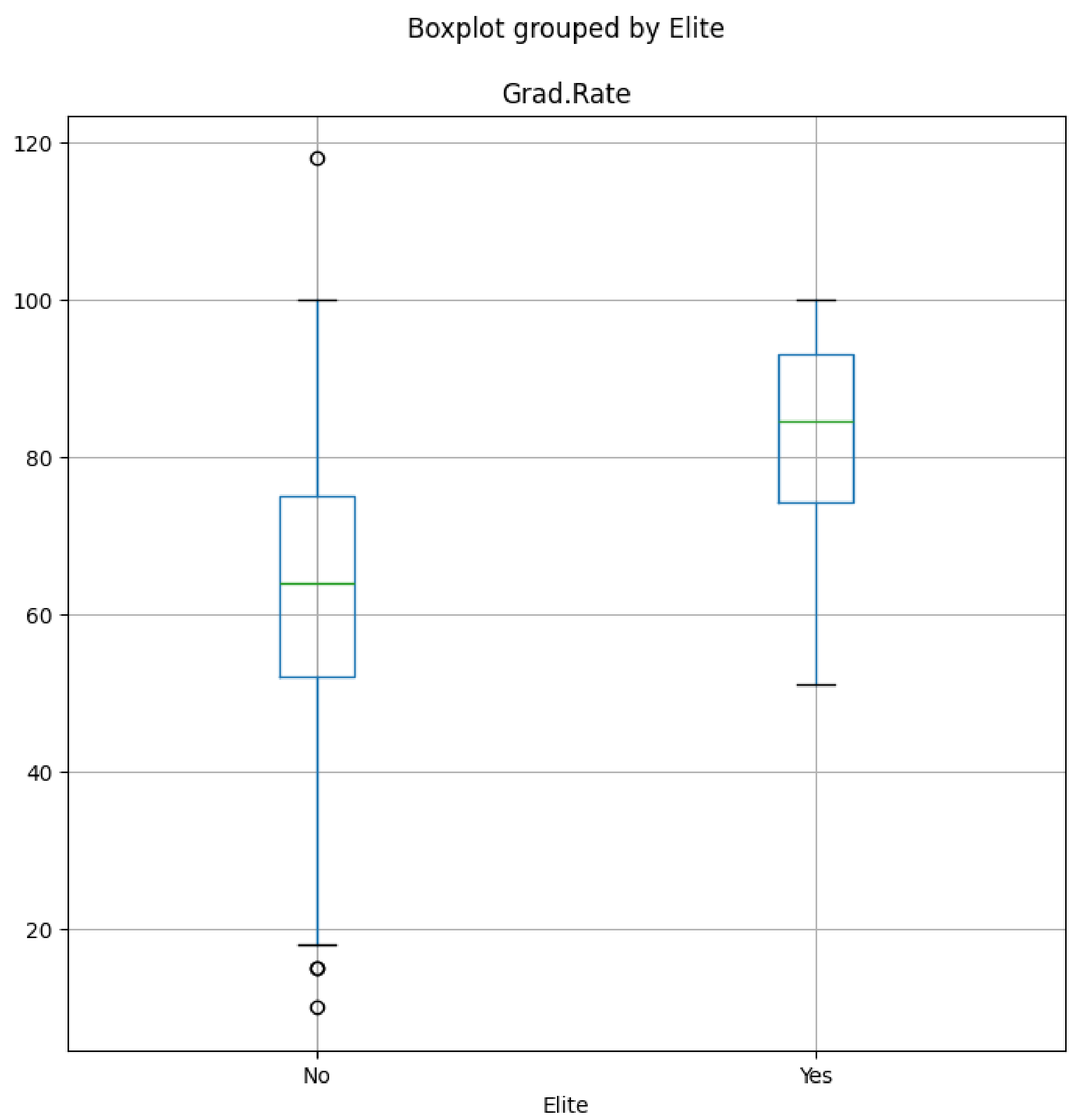
[37]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("Expend", by="Private", ax=ax);
```



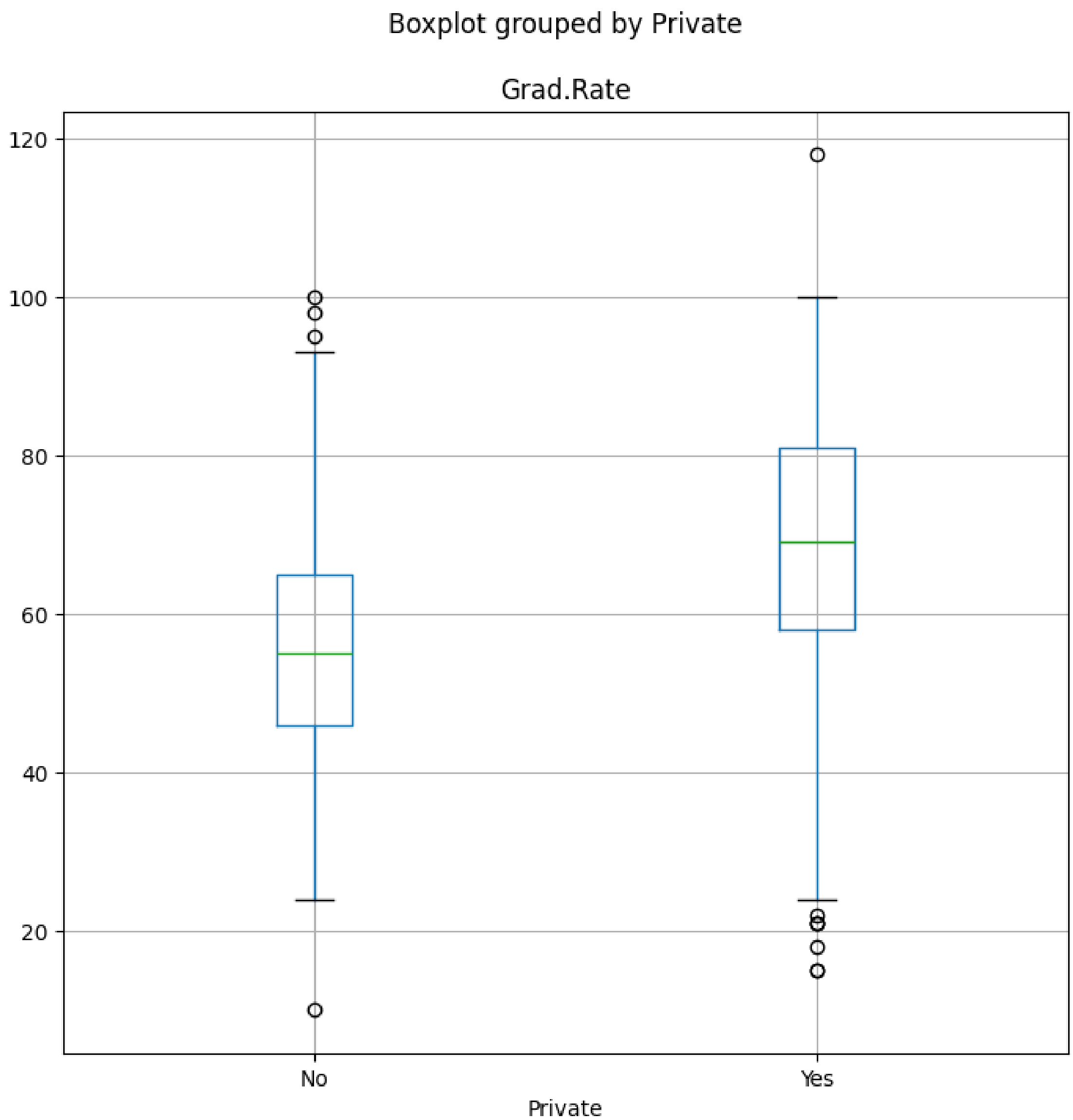
[38]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("Expend", by="Elite", ax=ax);
```



[39]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("Grad.Rate", by="Elite", ax=ax);
```



[40]:
```
fig, ax = subplots(figsize=(8, 8))
College.boxplot("Grad.Rate", by="Private", ax=ax);
```

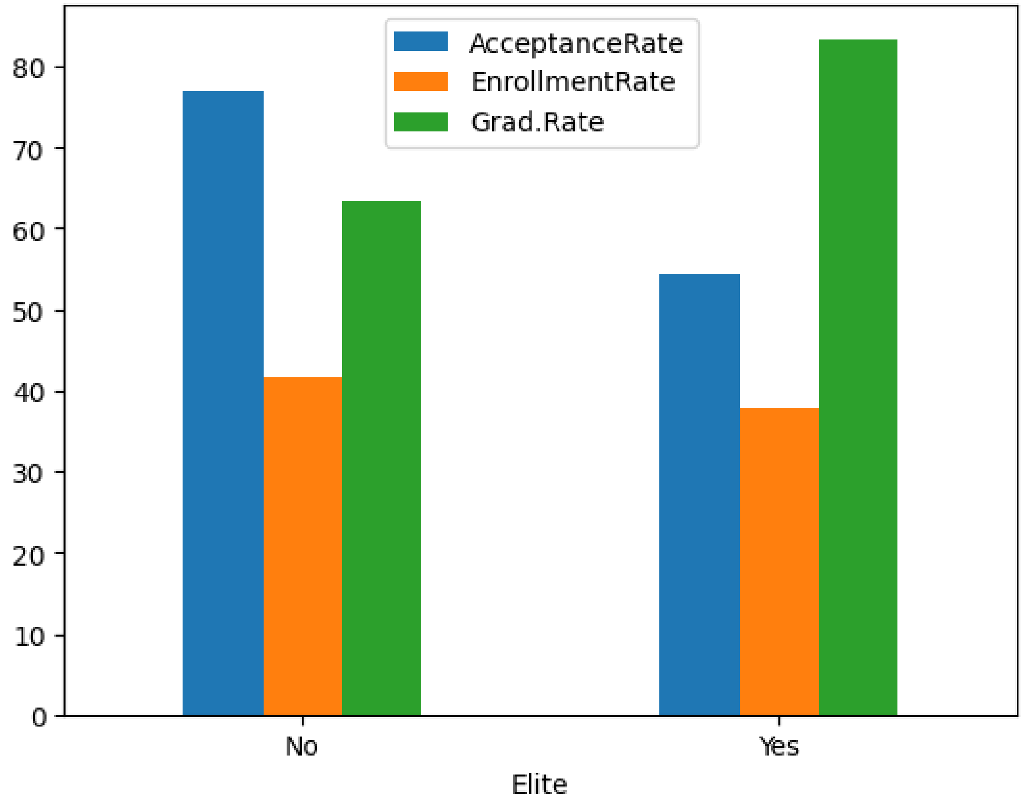


[41]:
```
mean_grad_rate = College.groupby("Elite", observed=True)[
    ["AcceptanceRate", "EnrollmentRate", "Grad.Rate"]
].mean()
mean_grad_rate
```

[41]:

Elite	AcceptanceRate	EnrollmentRate	Grad.Rate
No	76.963834	41.586452	63.463519
Yes	54.340128	37.748205	83.384615

[42]:
```
mean_grad_rate.plot(
    y=["AcceptanceRate", "EnrollmentRate", "Grad.Rate"], kind="bar", rot=0
);
```



[43]:
```
Auto = pd.read_csv("Auto.csv", na_values={"?"})
print(Auto.shape)
np.unique(Auto["horsepower"])
```

```
(397, 9)
```

[43]:
```
array([ 46.,  48.,  49.,  52.,  53.,  54.,  58.,  60.,  61.,  62.,  63.,
        64.,  65.,  66.,  67.,  68.,  69.,  70.,  71.,  72.,  74.,  75.,
        76.,  77.,  78.,  79.,  80.,  81.,  82.,  83.,  84.,  85.,  86.,
        87.,  88.,  89.,  90.,  91.,  92.,  93.,  94.,  95.,  96.,  97.,
        98., 100., 102., 103., 105., 107., 108., 110., 112., 113., 115.,
       116., 120., 122., 125., 129., 130., 132., 133., 135., 137., 138.,
       139., 140., 142., 145., 148., 149., 150., 152., 153., 155., 158.,
       160., 165., 167., 170., 175., 180., 190., 193., 198., 200., 208.,
       210., 215., 220., 225., 230.,  nan])
```

0.0.2 Which predictors are quantitative and which are qualitative?

Rename the misleading column name acceleration to timetoacceleration since it's a tad misleading.

[44]:
```
Auto["timetoacceleration"] = Auto["acceleration"]
Auto = Auto.drop("acceleration", axis=1)
```

[44]:
```
      mpg  cylinders  displacement  horsepower  weight  year  origin  \
0    18.0          8         307.0       130.0    3504    70       1
1    15.0          8         350.0       165.0    3693    70       1
2    18.0          8         318.0       150.0    3436    70       1
3    16.0          8         304.0       150.0    3433    70       1
4    17.0          8         302.0       140.0    3449    70       1
..    …        …           …       …      …      …
392  27.0          4         140.0        86.0    2790    82       1
393  44.0          4          97.0        52.0    2130    82       2
394  32.0          4         135.0        84.0    2295    82       1
395  28.0          4         120.0        79.0    2625    82       1
396  31.0          4         119.0        82.0    2720    82       1

                          name  timetoacceleration
0    chevrolet chevelle malibu                12.0
1            buick skylark 320                11.5
2           plymouth satellite                11.0
3                amc rebel sst                12.0
4                  ford torino                10.5
..                         …                 …
392            ford mustang gl                15.6
393                  vw pickup                24.6
394              dodge rampage                11.6
395                ford ranger                18.6
396                 chevy s-10                19.4

[397 rows x 9 columns]
```

[45]:
```
Auto = Auto.dropna()
Auto.shape
```

[45]:
```
(392, 9)
```

[46]:
```
Auto.describe()
```

[46]:
```
              mpg   cylinders  displacement  horsepower       weight  \
count  392.000000  392.000000    392.000000  392.000000   392.000000
mean    23.445918    5.471939    194.411990  104.469388  2977.584184
std      7.805007    1.705783    104.644004   38.491160   849.402560
min      9.000000    3.000000     68.000000   46.000000  1613.000000
25%     17.000000    4.000000    105.000000   75.000000  2225.250000
```

```
50%      22.750000    4.000000    151.000000   93.500000  2803.500000
75%      29.000000    8.000000    275.750000  126.000000  3614.750000
max      46.600000    8.000000    455.000000  230.000000  5140.000000

             year      origin  timetoacceleration
count  392.000000  392.000000          392.000000
mean    75.979592    1.576531           15.541327
std      3.683737    0.805518            2.758864
min     70.000000    1.000000            8.000000
25%     73.000000    1.000000           13.775000
50%     76.000000    1.000000           15.500000
75%     79.000000    2.000000           17.025000
max     82.000000    3.000000           24.800000
```

[47]:

```python
Auto["origin"] = Auto.origin.astype("category")
Auto["year"] = Auto.year.astype("category")
Auto["cylinders"] = Auto.cylinders.astype("category")
print(np.unique(Auto["year"]))
print(np.unique(Auto["cylinders"]))
```

```
[70 71 72 73 74 75 76 77 78 79 80 81 82]
[3 4 5 6 8]

/tmp/ipykernel_14396/803089839.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  Auto["origin"] = Auto.origin.astype("category")
/tmp/ipykernel_14396/803089839.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  Auto["year"] = Auto.year.astype("category")
/tmp/ipykernel_14396/803089839.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  Auto["cylinders"] = Auto.cylinders.astype("category")
```

[48]:

```python
Auto["origin"] = Auto["origin"].cat.rename_categories(
    {1: "American", 2: "European", 3: "Japanese"}
)
```

```
np.unique(Auto["origin"])
```

```
/tmp/ipykernel_14396/29212070.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  Auto["origin"] = Auto["origin"].cat.rename_categories(
```

[48]:
```
array(['American', 'European', 'Japanese'], dtype=object)
```

[49]:
```
Auto.head()
```

[49]:
```
    mpg cylinders  displacement  horsepower  weight year   origin  \
0  18.0         8         307.0       130.0    3504   70  American
1  15.0         8         350.0       165.0    3693   70  American
2  18.0         8         318.0       150.0    3436   70  American
3  16.0         8         304.0       150.0    3433   70  American
4  17.0         8         302.0       140.0    3449   70  American

                        name  timetoacceleration
0  chevrolet chevelle malibu                12.0
1          buick skylark 320                11.5
2         plymouth satellite                11.0
3              amc rebel sst                12.0
4                ford torino                10.5
```

[50]:
```
Auto = Auto.set_index("name")
```

[50]:
```
                            mpg cylinders  displacement  horsepower  weight  \
name
chevrolet chevelle malibu  18.0         8         307.0       130.0    3504
buick skylark 320          15.0         8         350.0       165.0    3693
plymouth satellite         18.0         8         318.0       150.0    3436
amc rebel sst              16.0         8         304.0       150.0    3433
ford torino                17.0         8         302.0       140.0    3449
…                          …        …             …          …       …
ford mustang gl            27.0         4         140.0        86.0    2790
vw pickup                  44.0         4          97.0        52.0    2130
dodge rampage              32.0         4         135.0        84.0    2295
ford ranger                28.0         4         120.0        79.0    2625
chevy s-10                 31.0         4         119.0        82.0    2720

                          year    origin  timetoacceleration
name
chevrolet chevelle malibu   70  American                12.0
buick skylark 320           70  American                11.5
```

```
plymouth satellite          70  American                  11.0
amc rebel sst               70  American                  12.0
ford torino                 70  American                  10.5
…                          …     …                  …
ford mustang gl             82  American                  15.6
vw pickup                   82  European                  24.6
dodge rampage               82  American                  11.6
ford ranger                 82  American                  18.6
chevy s-10                  82  American                  19.4

[392 rows x 8 columns]
```

[51]:
```
Auto
```

[51]:
```
                            mpg  cylinders  displacement  horsepower  weight  \
name
chevrolet chevelle malibu  18.0          8         307.0       130.0    3504
buick skylark 320          15.0          8         350.0       165.0    3693
plymouth satellite         18.0          8         318.0       150.0    3436
amc rebel sst              16.0          8         304.0       150.0    3433
ford torino                17.0          8         302.0       140.0    3449
…                           …        …            …          …      …
ford mustang gl            27.0          4         140.0        86.0    2790
vw pickup                  44.0          4          97.0        52.0    2130
dodge rampage              32.0          4         135.0        84.0    2295
ford ranger                28.0          4         120.0        79.0    2625
chevy s-10                 31.0          4         119.0        82.0    2720

                          year    origin  timetoacceleration
name
chevrolet chevelle malibu   70  American                  12.0
buick skylark 320           70  American                  11.5
plymouth satellite          70  American                  11.0
amc rebel sst               70  American                  12.0
ford torino                 70  American                  10.5
…                          …     …                  …
ford mustang gl             82  American                  15.6
vw pickup                   82  European                  24.6
dodge rampage               82  American                  11.6
ford ranger                 82  American                  18.6
chevy s-10                  82  American                  19.4

[392 rows x 8 columns]
```

[52]:
```
Auto_new = Auto.drop(Auto.index[10:86])
Auto_new.describe()
```

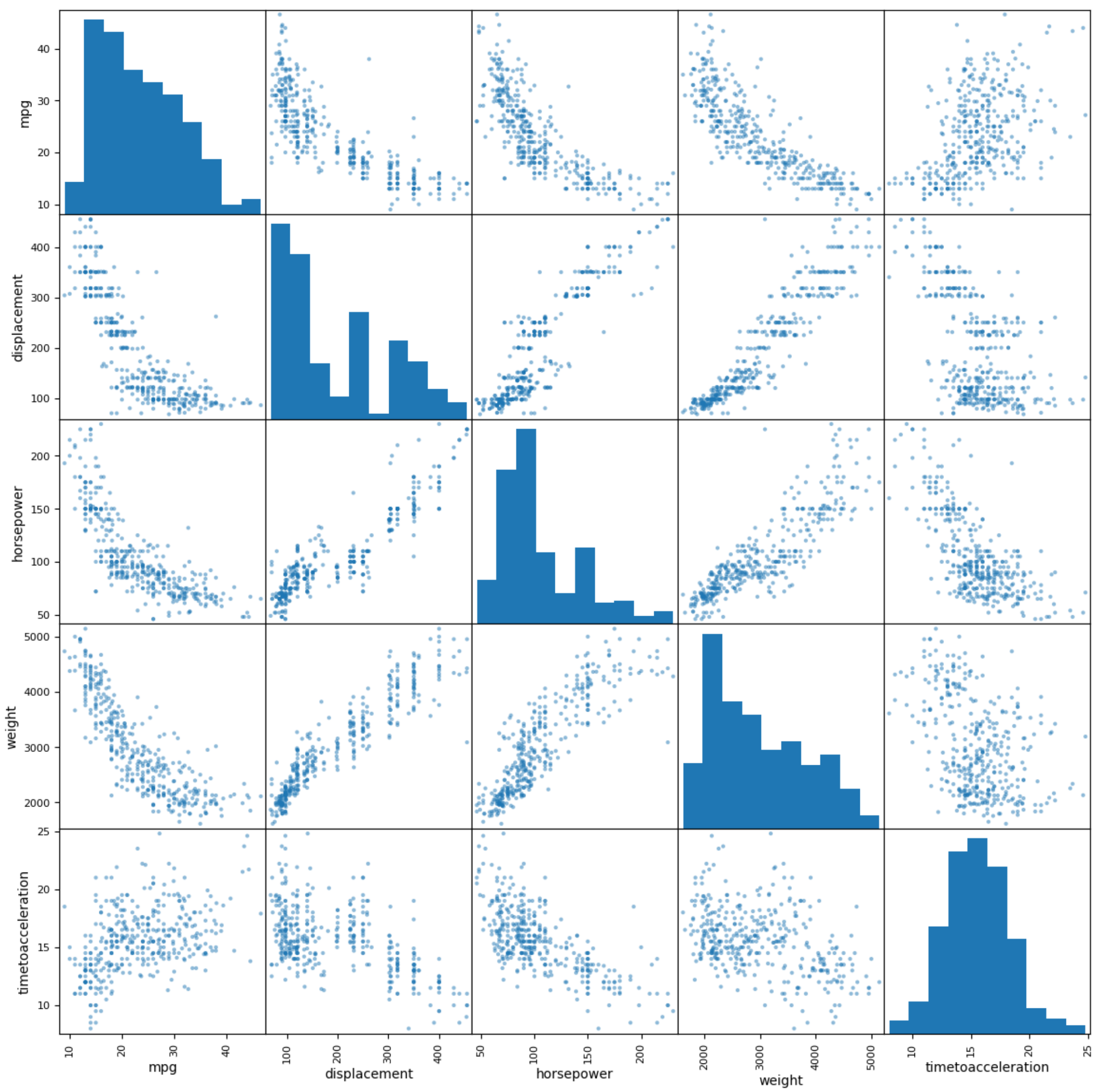
[52]:
```
              mpg  displacement  horsepower       weight  timetoacceleration
count  282.000000    282.000000  282.000000   282.000000          282.000000
mean    25.006028    180.120567   99.039007  2884.939716           15.713121
std      7.921384     96.164263   34.197280   793.236373            2.601575
min     11.000000     68.000000   46.000000  1755.000000            8.500000
25%     18.125000     98.000000   74.250000  2188.500000           14.000000
50%     24.500000    140.000000   90.000000  2715.500000           15.500000
75%     31.000000    250.000000  112.000000  3435.250000           17.275000
max     46.600000    455.000000  230.000000  4952.000000           24.600000
```

[53]:
```
Auto_new
```

[53]:
```
                      mpg  cylinders  displacement  horsepower  weight  year  \
name
buick skylark 320    15.0          8         350.0       165.0    3693    70
plymouth satellite   18.0          8         318.0       150.0    3436    70
amc rebel sst        16.0          8         304.0       150.0    3433    70
ford torino          17.0          8         302.0       140.0    3449    70
amc ambassador dpl   15.0          8         390.0       190.0    3850    70
…                     …          …             …           …      …   …
ford mustang gl      27.0          4         140.0        86.0    2790    82
vw pickup            44.0          4          97.0        52.0    2130    82
dodge rampage        32.0          4         135.0        84.0    2295    82
ford ranger          28.0          4         120.0        79.0    2625    82
chevy s-10           31.0          4         119.0        82.0    2720    82

                      origin  timetoacceleration
name
buick skylark 320    American                 11.5
plymouth satellite   American                 11.0
amc rebel sst        American                 12.0
ford torino          American                 10.5
amc ambassador dpl   American                  8.5
…                       …                  …
ford mustang gl      American                 15.6
vw pickup            European                 24.6
dodge rampage        American                 11.6
ford ranger          American                 18.6
chevy s-10           American                 19.4

[282 rows x 8 columns]
```

Using the full data set, investigate the predictors graphically, using scatter plots or other tools of your choice. Create some plots highlighting the relationships among the predictors. Comment on your findings.

[54]:
```
pd.plotting.scatter_matrix(Auto, figsize=(14, 14));
```



0.0.3 Findings:

1. Weight and displacement seem to be negatively correlated with MPG.
2. timetoacceleration (0–60 mph in seconds) seems to be positively correlated with MPG. As time to acceleration increases, MPG also increases. The longer the time to acceleration, the better the fuel efficiency.
3. weight is also positively correlated with displacement. As weight increases, so does displacement, i.e., as the body weight increases, so does displacement need to increase.
4. Displacement is seen to increase as the number of cylinders increase. This is expected since displacement is a function of the number of cylinders, amongst other components.

We can conclude that MPG can be predicted using the variables weight, displacement and timetoacceleration.

[55]:
```python
mean_mpg_origin = Auto.groupby(["origin"], observed=True)[["mpg"]].mean()
mean_mpg_origin
```

[55]:
```
                mpg
origin
American  20.033469
European  27.602941
Japanese  30.450633
```

[56]:
```python
mean_mpg_year = Auto.groupby(["year"], observed=True)[["mpg"]].mean()
mean_mpg_year
```

[56]:
```
            mpg
year
70    17.689655
71    21.111111
72    18.714286
73    17.100000
74    22.769231
75    20.266667
76    21.573529
77    23.375000
78    24.061111
79    25.093103
80    33.803704
81    30.185714
82    32.000000
```

[57]:
```python
mean_mpg_cylinders = Auto.groupby(["cylinders"], observed=True)[["mpg"]].mean()
mean_mpg_cylinders
```

[57]:
```
                 mpg
cylinders
3          20.550000
4          29.283920
5          27.366667
6          19.973494
8          14.963107
```

We can also observe that fuel efficiency is affected by the make of the car. Japanese > European > American The year also plays a significant role. Later model cars are more fuel efficient than the earlier models. Cars are also more fuel efficient with lesser number of cylinders. These can also be used as predictors to deduce the MPG.

[58]:
```python
Boston = load_data("Boston")
Boston.columns
```

[58]:
```
Index(['crim', 'zn', 'indus', 'chas', 'nox', 'rm', 'age', 'dis', 'rad', 'tax',
       'ptratio', 'lstat', 'medv'],
      dtype='object')
```

[59]:
```
Boston.shape
```

[59]:
```
(506, 13)
```

The rows represent data for 506 suburbs in Boston. The columns represent housing values and variables of interest that may predict housing values in each suburb.

[60]:
```
Boston.describe()
```

[60]:
```
             crim          zn       indus        chas         nox          rm  \
count  506.000000  506.000000  506.000000  506.000000  506.000000  506.000000
mean     3.613524   11.363636   11.136779    0.069170    0.554695    6.284634
std      8.601545   23.322453    6.860353    0.253994    0.115878    0.702617
min      0.006320    0.000000    0.460000    0.000000    0.385000    3.561000
25%      0.082045    0.000000    5.190000    0.000000    0.449000    5.885500
50%      0.256510    0.000000    9.690000    0.000000    0.538000    6.208500
75%      3.677083   12.500000   18.100000    0.000000    0.624000    6.623500
max     88.976200  100.000000   27.740000    1.000000    0.871000    8.780000

              age         dis         rad         tax     ptratio       lstat  \
count  506.000000  506.000000  506.000000  506.000000  506.000000  506.000000
mean    68.574901    3.795043    9.549407  408.237154   18.455534   12.653063
std     28.148861    2.105710    8.707259  168.537116    2.164946    7.141062
min      2.900000    1.129600    1.000000  187.000000   12.600000    1.730000
25%     45.025000    2.100175    4.000000  279.000000   17.400000    6.950000
50%     77.500000    3.207450    5.000000  330.000000   19.050000   11.360000
75%     94.075000    5.188425   24.000000  666.000000   20.200000   16.955000
max    100.000000   12.126500   24.000000  711.000000   22.000000   37.970000

             medv
count  506.000000
mean    22.532806
std      9.197104
min      5.000000
25%     17.025000
50%     21.200000
75%     25.000000
max     50.000000
```

[61]:
```
Boston_quant = Boston.drop("chas", axis=1)
```

[61]:
```
      crim    zn  indus    nox     rm   age     dis  rad  tax  ptratio  \
0  0.00632  18.0   2.31  0.538  6.575  65.2  4.0900    1  296     15.3
1  0.02731   0.0   7.07  0.469  6.421  78.9  4.9671    2  242     17.8
```

```
2    0.02729   0.0    7.07  0.469  7.185  61.1  4.9671    2  242      17.8
3    0.03237   0.0    2.18  0.458  6.998  45.8  6.0622    3  222      18.7
4    0.06905   0.0    2.18  0.458  7.147  54.2  6.0622    3  222      18.7
..        …     …      …     …      …    …     …   …  …        …
501  0.06263   0.0   11.93  0.573  6.593  69.1  2.4786    1  273      21.0
502  0.04527   0.0   11.93  0.573  6.120  76.7  2.2875    1  273      21.0
503  0.06076   0.0   11.93  0.573  6.976  91.0  2.1675    1  273      21.0
504  0.10959   0.0   11.93  0.573  6.794  89.3  2.3889    1  273      21.0
505  0.04741   0.0   11.93  0.573  6.030  80.8  2.5050    1  273      21.0

     lstat  medv
0     4.98  24.0
1     9.14  21.6
2     4.03  34.7
3     2.94  33.4
4     5.33  36.2
..     …     …
501   9.67  22.4
502   9.08  20.6
503   5.64  23.9
504   6.48  22.0
505   7.88  11.9

[506 rows x 12 columns]
```

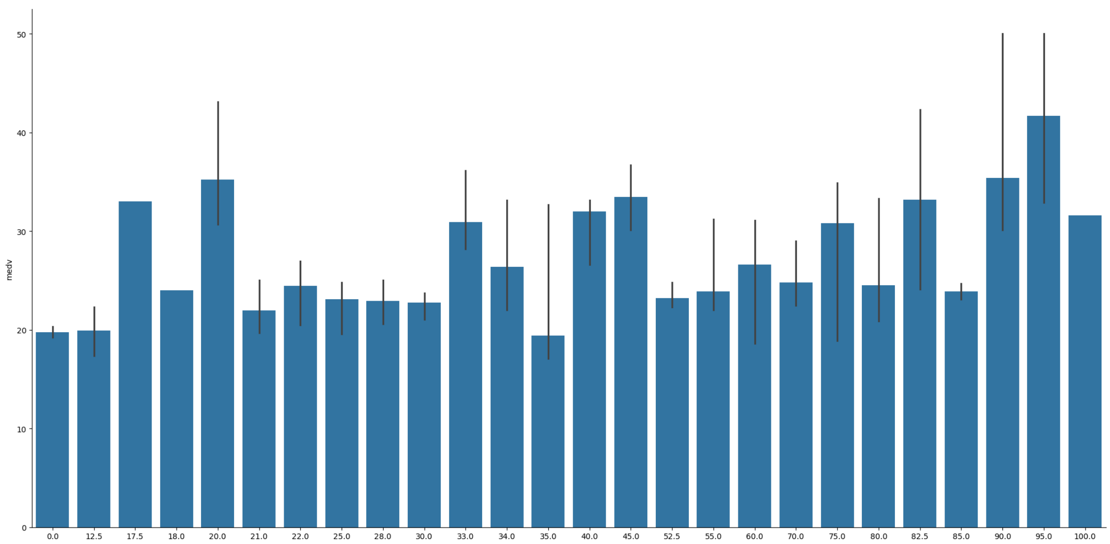
[62]:
```python
print(np.unique(Boston_quant["zn"]))
median_medv = Boston_quant.groupby(["zn"], observed=True)[["medv"]].median()
median_medv
```

```
[  0.   12.5  17.5  18.   20.   21.   22.   25.   28.   30.   33.   34.
  35.   40.   45.   52.5  55.   60.   70.   75.   80.   82.5  85.   90.
  95.  100. ]
```

[62]:
```
        medv
zn
0.0    19.75
12.5   19.90
17.5   33.00
18.0   24.00
20.0   35.20
21.0   21.95
22.0   24.45
25.0   23.10
28.0   22.90
30.0   22.75
33.0   30.90
34.0   26.40
35.0   19.40
```

40.0	32.00
45.0	33.45
52.5	23.20
55.0	23.90
60.0	26.60
70.0	24.80
75.0	30.80
80.0	24.50
82.5	33.20
85.0	23.90
90.0	35.40
95.0	41.70
100.0	31.60

[63]:
```
sns.catplot(
    data=Boston_quant,
    x="zn",
    y="medv",
    kind="bar",
    height=10,
    aspect=2,
    estimator=median,
);
```



[64]:
```
print(np.unique(Boston_quant["rad"]))
mean_rad = Boston_quant.groupby(["rad"], observed=True)[["medv"]].mean()
mean_rad
```

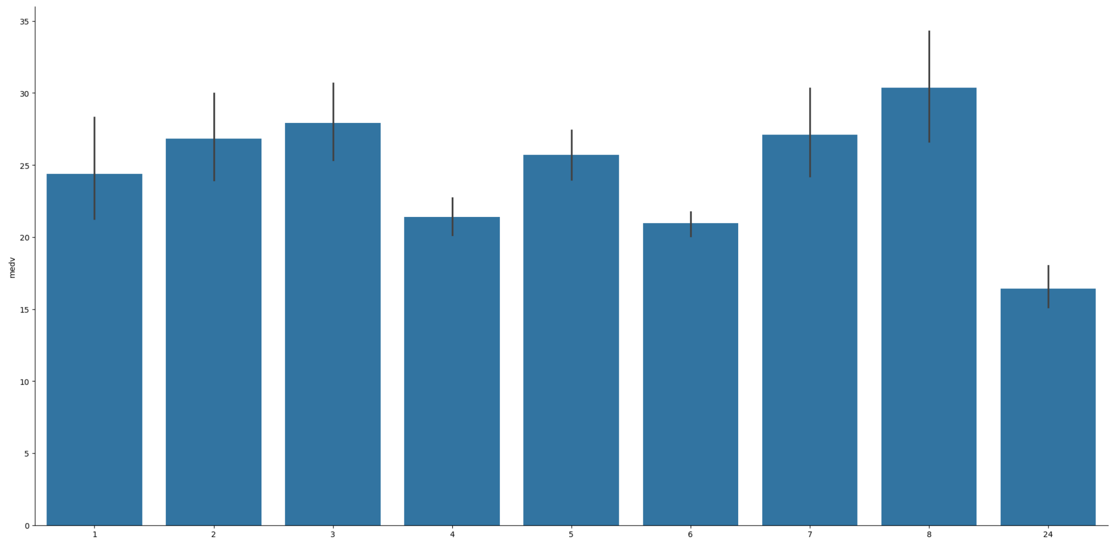
```
[ 1  2  3  4  5  6  7  8 24]
```

[64]:

	medv
rad	
1	24.365000
2	26.833333
3	27.928947
4	21.387273
5	25.706957
6	20.976923
7	27.105882
8	30.358333
24	16.403788

[65]:

```python
sns.catplot(data=Boston_quant, x="rad", y="medv", kind="bar", height=10, aspect=2);
```

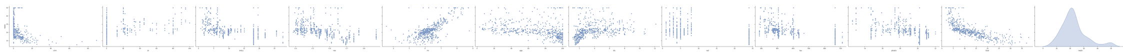


[66]:

```python
sns.set_theme(style="ticks")
g = sns.pairplot(Boston_quant, height=5, aspect=2, diag_kind="kde", y_vars=["medv"]);
```

```
Executing <Handle BaseSelectorEventLoop._read_from_self() created at
/usr/lib/python3.12/asyncio/selector_events.py:280> took 0.289 seconds
Executing <Handle IOLoop._run_callback(functools.par…7dd251c1b880>)) created
at /home/linus/ISLP/islpenv/lib/python3.12/site-
packages/tornado/platform/asyncio.py:235> took 0.111 seconds
Executing <Task pending name='Task-2' coro=<Kernel.poll_control_queue() running
at /home/linus/ISLP/islpenv/lib/python3.12/site-
packages/ipykernel/kernelbase.py:304> wait_for=<Future pending
cb=[Task.task_wakeup()] created at /home/linus/ISLP/islpenv/lib/python3.12/site-
packages/tornado/queues.py:248> cb=[_chain_future.<locals>._call_set_state() at
```

```
/usr/lib/python3.12/asyncio/futures.py:394] created at
/usr/lib/python3.12/asyncio/tasks.py:695> took 0.159 seconds
```

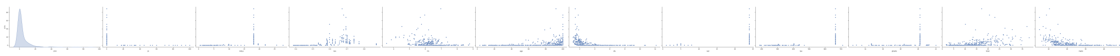


Plotting the other quantitative columns against medv (Median value of owner-occupied homes), we can see that: 1. crim is negatively correlated with medv. i.e., as crime rate increases, median value of homes decrease. 2. indus is negatively correlated with medv which is expected as industrialisation of a town increases, the house prices decrease. 3. nox is negatively correlated with medv which is also expected. 4. as the number of rooms (rm) increase, so does the value of the home. 5. as the proportion of homes built prior to 1940 increase, the value of homes in that area decrease. There are some notable outliers, but that appears to be the general trend. 6. There is a clear relationship in the lsat (lower status of population percent) versus medv where medv decreases with the increase in lstat on the x-axis.

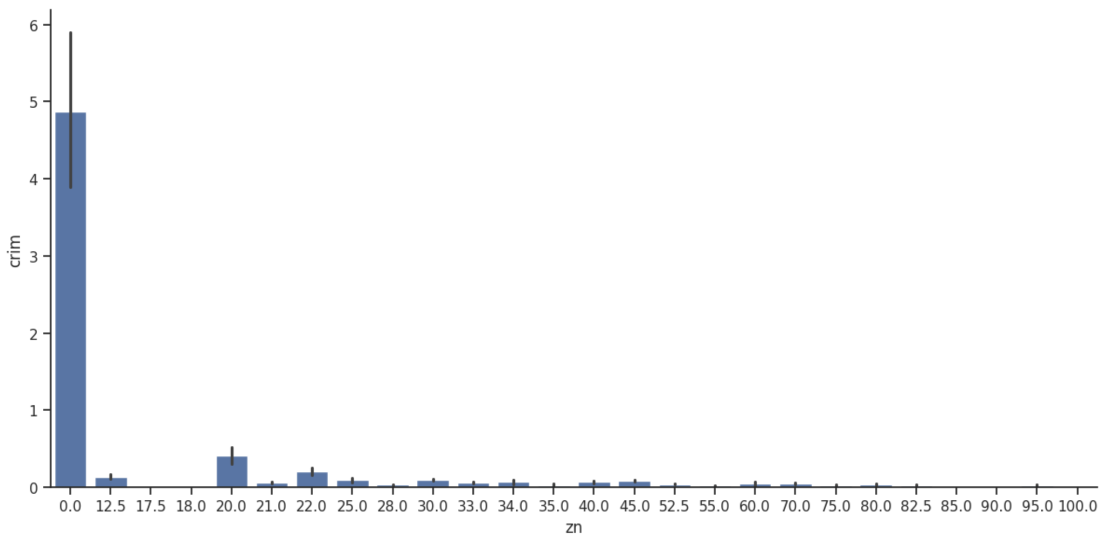
[67]:
```python
Boston_quant["zn"].value_counts()
```

[67]:
```
zn
0.0      372
20.0      21
80.0      15
22.0      10
12.5      10
25.0      10
40.0       7
45.0       6
30.0       6
90.0       5
95.0       4
60.0       4
21.0       4
33.0       4
55.0       3
70.0       3
34.0       3
52.5       3
35.0       3
28.0       3
75.0       3
82.5       2
85.0       2
17.5       1
100.0      1
18.0       1
Name: count, dtype: int64
```

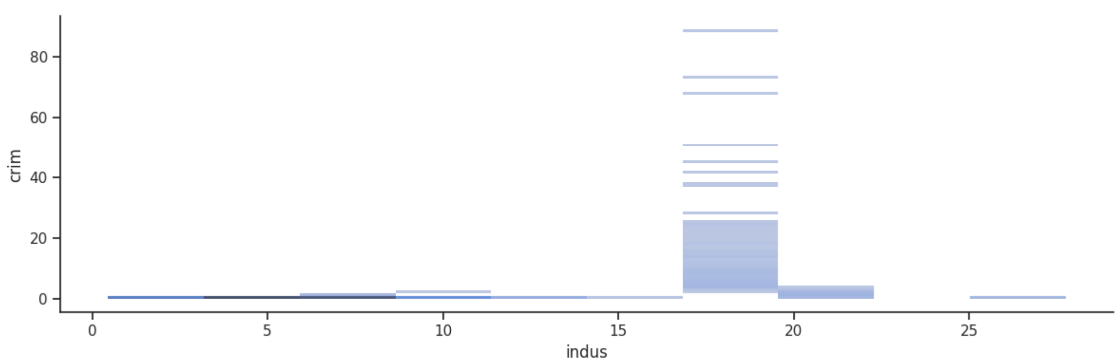
[68]:
```python
g = sns.pairplot(Boston_quant, height=5, aspect=2, diag_kind="kde", y_vars=["crim"]);
```



[69]:
```python
sns.catplot(data=Boston_quant, x="zn", y="crim", kind="bar", height=6, aspect=2);
```



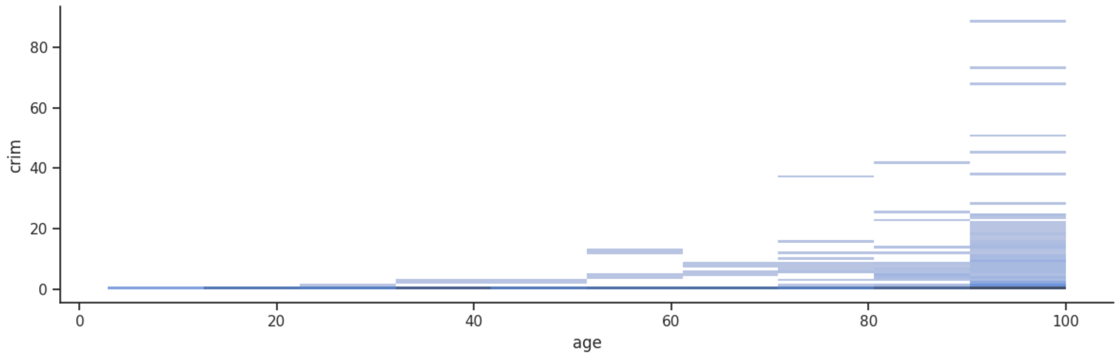
[70]:
```python
sns.displot(data=Boston_quant, x="indus", y="crim", height=4, aspect=3);
```



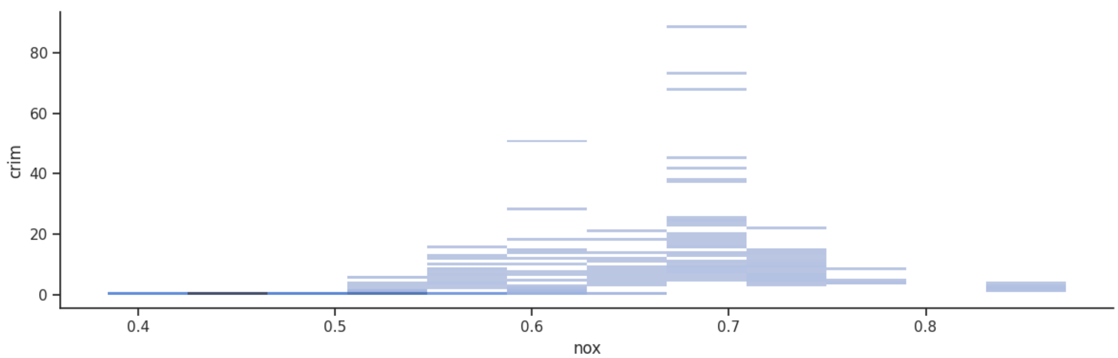
[71]:
```python
sns.displot(data=Boston_quant, x="age", y="crim", height=4, aspect=3);
```

```
Executing <Handle BaseAsyncIOLoop._handle_events(28, 1) created at
```

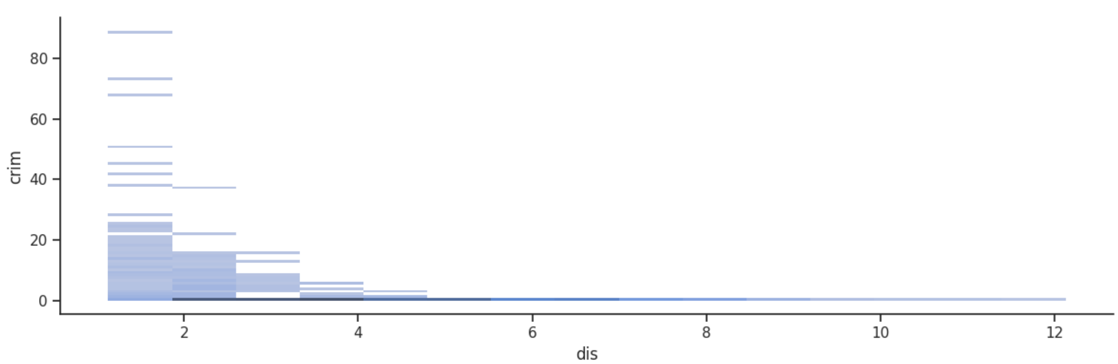
```
/usr/lib/python3.12/asyncio/selector_events.py:280> took 0.290 seconds
Executing <Handle IOLoop._run_callback(functools.par…7dd2511184a0>)) created
at /home/linus/ISLP/islpenv/lib/python3.12/site-
packages/tornado/platform/asyncio.py:235> took 0.299 seconds
```



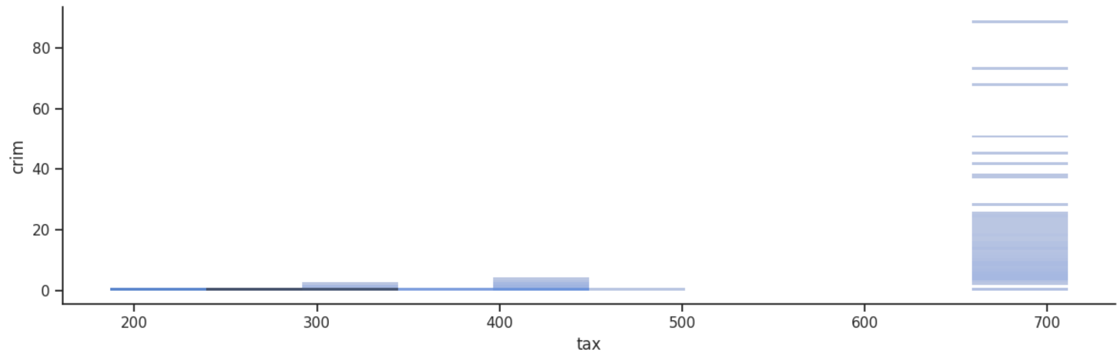
[72]:
```python
sns.displot(data=Boston_quant, x="nox", y="crim", height=4, aspect=3);
```



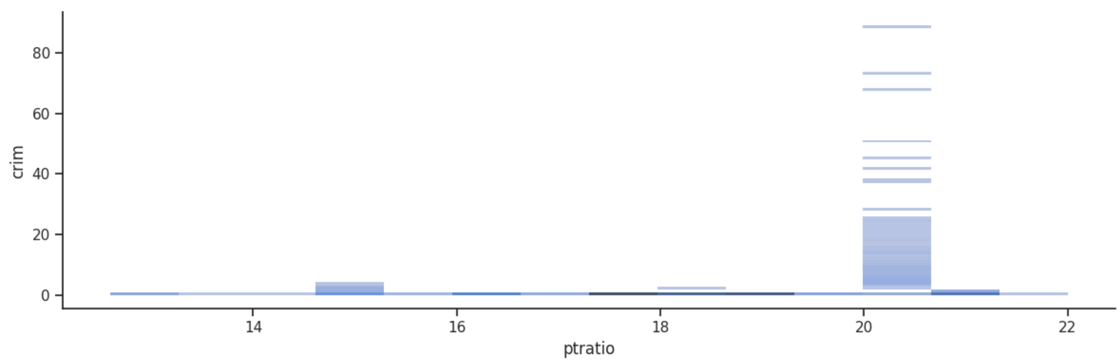
[73]:
```python
sns.displot(data=Boston_quant, x="dis", y="crim", height=4, aspect=3);
```



[74]:
```
sns.displot(data=Boston_quant, x="tax", y="crim", height=4, aspect=3);
```



[75]:
```
sns.displot(data=Boston_quant, x="ptratio", y="crim", height=4, aspect=3);
```



We've already seen that there appears to be a relationship b/w crime rate and medv where a higher crime rate is associated with lower property prices. Additionally, plotting the other quantitative variables against crime rate (crim), we can perceive the following: 1. No-zoned areas or towns are associated with higher crime rate compared to all other zoning percentages. 2. For some reason, industrialization of around 18% displays a spike in the crime rate compared to the other suburbs. This might be worth investigating further. 3. Suburbs with nox > 0.55 or so have an elevated crime rate. That could be because lower strata income people live in those areas, and they are more inclined to criminal activities. 4. There also seems to be an increasing relationship b/w crime rate and percentage of homes built prior to 1940. Once that percentage crosses 40%, there is an increasing number of suburbs that exhibit elevated crime rates. 5. Suburbs within a distance to Boston employment centres that range from 1 to 4.5 show an elevated crime rate. This needs to be investigated further. Where are these employment centres located? 6. There seems to be a higher incidence of crimes for areas with tax rate around 670. Why? 7. The crime rate does not seem to have a strong relationship with ptratio, but for around point 20.1 where the crime rate spikes compared to the other areas. 8. Crime rate decreases as the median value of properties rise across suburbs as a whole.

Do any of the suburbs of Boston appear to have particularly high crime rates? Tax rates? Pupil-teacher ratios? Comment on the range of each predictor.

[76]:
```
Boston_crim = Boston_quant.sort_values(
    by="crim", axis=0, ascending=False, inplace=False
)
top_crim = Boston_crim.head()
print(top_crim)
df = pd.DataFrame((Boston_quant.min(), Boston_quant.max()), index=["Min",␣
 ↪"Max"])
df
```

```
        crim   zn  indus    nox     rm    age     dis  rad  tax  ptratio  \
380  88.9762  0.0   18.1  0.671  6.968   91.9  1.4165   24  666     20.2
418  73.5341  0.0   18.1  0.679  5.957  100.0  1.8026   24  666     20.2
405  67.9208  0.0   18.1  0.693  5.683  100.0  1.4254   24  666     20.2
410  51.1358  0.0   18.1  0.597  5.757  100.0  1.4130   24  666     20.2
414  45.7461  0.0   18.1  0.693  4.519  100.0  1.6582   24  666     20.2

     lstat  medv
380  17.21  10.4
418  20.62   8.8
405  22.98   5.0
410  10.11  15.0
414  36.98   7.0
```

[76]:
```
         crim     zn  indus    nox     rm    age      dis   rad    tax  \
Min   0.00632    0.0   0.46  0.385  3.561    2.9   1.1296   1.0  187.0
Max  88.97620  100.0  27.74  0.871  8.780  100.0  12.1265  24.0  711.0

     ptratio  lstat  medv
Min     12.6   1.73   5.0
Max     22.0  37.97  50.0
```

As we can see from the dataset above, the top five crime rate suburbs are unzoned, have an industrialization rate of 18.1%, nox of 0.671, rooms ranging from 4.5 to 7, percentage of houses built prior to 1940 ranging from 92 to 100%, high tax rate of $666per10,000$ property tax , ptratio of 20.2. The lstat varies from 10.11 rto 36.98 and the median house values from 5.0 to 15.0 which are among the lowest. The index of accessibility to radial highways is 24 which is the best rank amongst all the suburbs.

How many of the suburbs in this data set bound the Charles River?

[77]:
```
len(Boston.query("chas == 1"))
```

[77]: 35

What is the median pupil-teacher ratio among the towns in this data set?

[78]:
```
median(Boston_quant["ptratio"])
```

[78]: 19.05

Which suburb of Boston has lowest median value of owner-occupied homes? What are the values of the other predictors for that suburb, and how do those values compare to the overall ranges for those predictors? Comment on your findings.

[79]:
```
lowest_medv = Boston_quant[Boston_quant["medv"] == Boston_quant["medv"].min()]
lowest_medv
```

[79]:

	crim	zn	indus	nox	rm	age	dis	rad	tax	ptratio	\
398	38.3518	0.0	18.1	0.693	5.453	100.0	1.4896	24	666	20.2	
405	67.9208	0.0	18.1	0.693	5.683	100.0	1.4254	24	666	20.2	

	lstat	medv
398	30.59	5.0
405	22.98	5.0

From the above two data points with the lowest median value for owner-occupied homes, it's evident that crime rate by itself does not determine the median value of homes for those regions. Except for lstat and crim, the other predictors match exactly the two data points. lstat for these two data points are high at 22.98 and 30.59 respectively. In these two cases, the other predictors do a better job of explaining the median value for the homes in these suburbs or these suburbs are neighbouring each other.

In this data set, how many of the suburbs average more than seven rooms per dwelling? More than eight rooms per dwelling? Comment on the suburbs that average more than eight rooms per dwelling.

[80]:
```
len(Boston_quant[Boston_quant["rm"] > 7])
```

[80]: 64

[81]:
```
eight_rooms = Boston_quant[Boston_quant["rm"] > 8]
print(len(eight_rooms))
eight_rooms
```

13

[81]:

	crim	zn	indus	nox	rm	age	dis	rad	tax	ptratio	\
97	0.12083	0.0	2.89	0.4450	8.069	76.0	3.4952	2	276	18.0	
163	1.51902	0.0	19.58	0.6050	8.375	93.9	2.1620	5	403	14.7	
204	0.02009	95.0	2.68	0.4161	8.034	31.9	5.1180	4	224	14.7	
224	0.31533	0.0	6.20	0.5040	8.266	78.3	2.8944	8	307	17.4	
225	0.52693	0.0	6.20	0.5040	8.725	83.0	2.8944	8	307	17.4	
226	0.38214	0.0	6.20	0.5040	8.040	86.5	3.2157	8	307	17.4	
232	0.57529	0.0	6.20	0.5070	8.337	73.3	3.8384	8	307	17.4	
233	0.33147	0.0	6.20	0.5070	8.247	70.4	3.6519	8	307	17.4	
253	0.36894	22.0	5.86	0.4310	8.259	8.4	8.9067	7	330	19.1	

```
257  0.61154  20.0   3.97  0.6470  8.704  86.9  1.8010    5  264     13.0
262  0.52014  20.0   3.97  0.6470  8.398  91.5  2.2885    5  264     13.0
267  0.57834  20.0   3.97  0.5750  8.297  67.0  2.4216    5  264     13.0
364  3.47428   0.0  18.10  0.7180  8.780  82.9  1.9047   24  666     20.2

     lstat  medv
97    4.21  38.7
163   3.32  50.0
204   2.88  50.0
224   4.14  44.8
225   4.63  50.0
226   3.13  37.6
232   2.47  41.7
233   3.95  48.3
253   3.54  42.8
257   5.12  50.0
262   5.91  48.8
267   7.44  50.0
364   5.29  21.9
```

There are 13 suburbs that average more that 8 rooms per dwelling.

The median value for these dwellings range from 21.9 to 50.0 which is the priciest.

The crime rate in these suburbs is extremely low with the highest at around 3.5%.

Industrialization of these suburbs is also low with 19.58 the maximum.

The percentage of people from the lower income strata tops out at 7.44%

A substantial percentage of dwellings are built prior to 1940 which could explain the higher number of rooms with only one outlier at 8.4%.

[82]:
```
allDone()
```

```
<IPython.lib.display.Audio object>
```