Project 2

In this project, you will implement the exploratory analysis plan developed in Project 1. This will lay the groundwork for our our first modeling exercise in Project 3.

Step 1: Load the python libraries you will need for this project

In [3]:
```
#imports
from __future__ import division
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sm
import pylab as pl
import numpy as np
%matplotlib inline
```

```
C:\Users\elizabeth.xu\AppData\Local\Continuum\Anaconda2\lib\site-packages\sta
tsmodels\compat\pandas.py:56: FutureWarning: The pandas.core.datetools module
is deprecated and will be removed in a future version. Please use the pandas.
tseries module instead.
  from pandas.core import datetools
```

Step 2: Read in your data set

In [4]:
```
#Read in data from source
df_raw = pd.read_csv("../assets/admissions.csv")
print df_raw.head()
```

```
   admit    gre   gpa  prestige
0      0  380.0  3.61       3.0
1      1  660.0  3.67       3.0
2      1  800.0  4.00       1.0
3      1  640.0  3.19       4.0
4      0  520.0  2.93       4.0
```

Questions

Question 1. How many observations are in our dataset?

In [5]:
```
df_raw.count()
```

Out[5]:
```
admit       400
gre         398
gpa         398
prestige    399
dtype: int64
```

In [6]:
```
df_raw.tail()
```

Out[6]:

	admit	gre	gpa	prestige
395	0	620.0	4.00	2.0
396	0	560.0	3.04	3.0
397	0	460.0	2.63	2.0
398	0	700.0	3.65	2.0
399	0	600.0	3.89	3.0

Answer: 400 Observations, because these tables start counting at 0 (looking at df_raw.head)

Question 2. Create a summary table

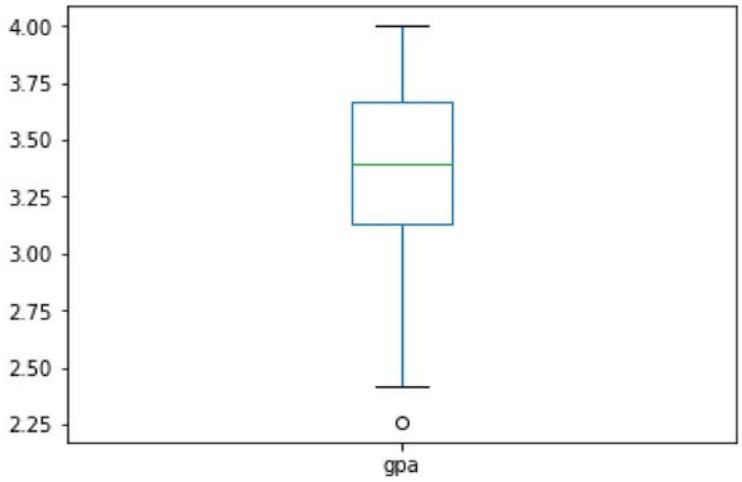
In [7]:
```
#function
df_raw.describe()
```

Out[7]:

	admit	gre	gpa	prestige
count	400.000000	398.000000	398.00000	399.000000
mean	0.317500	588.040201	3.39093	2.486216
std	0.466087	115.628513	0.38063	0.945333
min	0.000000	220.000000	2.26000	1.000000
25%	0.000000	520.000000	3.13000	2.000000
50%	0.000000	580.000000	3.39500	2.000000
75%	1.000000	660.000000	3.67000	3.000000
max	1.000000	800.000000	4.00000	4.000000

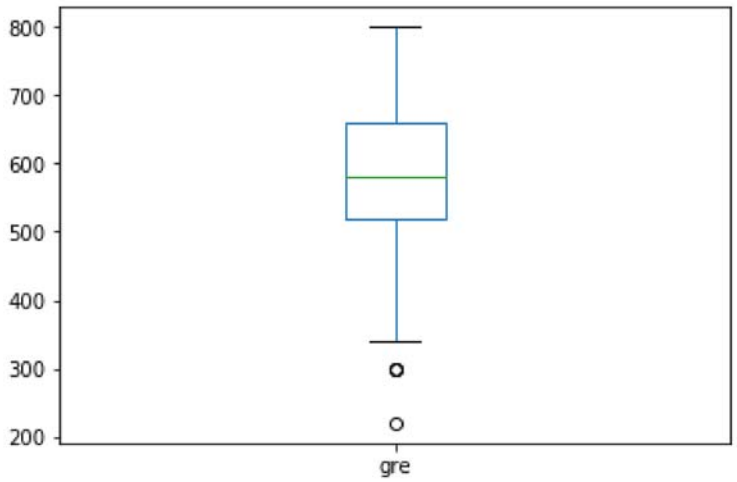
In [8]:
```
df_raw['gpa'].plot(kind='box')
```

Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0xc95d320>



In [9]:
```
df_raw['gre'].plot(kind='box')
```

Out[9]: <matplotlib.axes._subplots.AxesSubplot at 0xd4fd208>



Question 3. Why would GRE have a larger STD than GPA?

Answer: Because the range for GRE is much larger than GPA.

Question 4. Drop data points with missing data

In [10]:
```
df_clean = df_raw.dropna()
df_clean
```

Out[10]:

	admit	gre	gpa	prestige
0	0	380.0	3.61	3.0
1	1	660.0	3.67	3.0
2	1	800.0	4.00	1.0
3	1	640.0	3.19	4.0
4	0	520.0	2.93	4.0
5	1	760.0	3.00	2.0
6	1	560.0	2.98	1.0
7	0	400.0	3.08	2.0
8	1	540.0	3.39	3.0
9	0	700.0	3.92	2.0
10	0	800.0	4.00	4.0
11	0	440.0	3.22	1.0
12	1	760.0	4.00	1.0
13	0	700.0	3.08	2.0
14	1	700.0	4.00	1.0
15	0	480.0	3.44	3.0
16	0	780.0	3.87	4.0
17	0	360.0	2.56	3.0
18	0	800.0	3.75	2.0
19	1	540.0	3.81	1.0
20	0	500.0	3.17	3.0
21	1	660.0	3.63	2.0
22	0	600.0	2.82	4.0
23	0	680.0	3.19	4.0
24	1	760.0	3.35	2.0
25	1	800.0	3.66	1.0
26	1	620.0	3.61	1.0
27	1	520.0	3.74	4.0
28	1	780.0	3.22	2.0
29	0	520.0	3.29	1.0
...	...	...	...	...
370	1	540.0	3.77	2.0

	admit	gre	gpa	prestige
371	1	680.0	3.76	3.0
372	1	680.0	2.42	1.0
373	1	620.0	3.37	1.0
374	0	560.0	3.78	2.0
375	0	560.0	3.49	4.0
376	0	620.0	3.63	2.0
377	1	800.0	4.00	2.0
378	0	640.0	3.12	3.0
379	0	540.0	2.70	2.0
380	0	700.0	3.65	2.0
381	1	540.0	3.49	2.0
382	0	540.0	3.51	2.0
383	0	660.0	4.00	1.0
384	1	480.0	2.62	2.0
385	0	420.0	3.02	1.0
386	1	740.0	3.86	2.0
387	0	580.0	3.36	2.0
388	0	640.0	3.17	2.0
389	0	640.0	3.51	2.0
390	1	800.0	3.05	2.0
391	1	660.0	3.88	2.0
392	1	600.0	3.38	3.0
393	1	620.0	3.75	2.0
394	1	460.0	3.99	3.0
395	0	620.0	4.00	2.0
396	0	560.0	3.04	3.0
397	0	460.0	2.63	2.0
398	0	700.0	3.65	2.0
399	0	600.0	3.89	3.0

397 rows × 4 columns

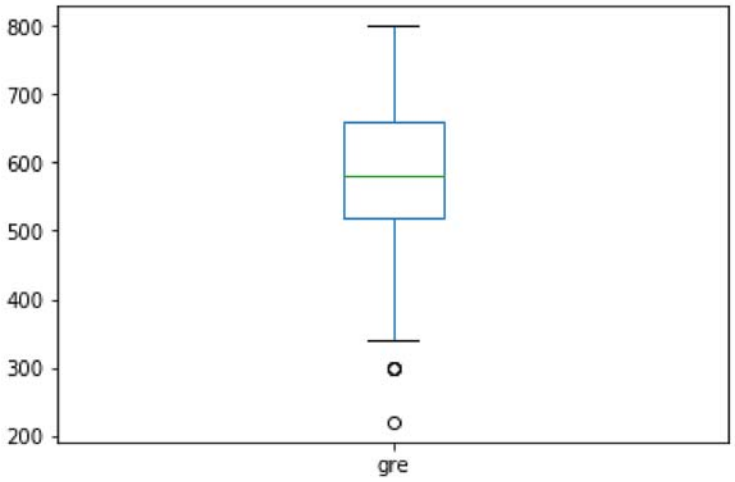
Question 5. Confirm that you dropped the correct data. How can you tell?

Answer: There are now less rows 397 vs. 400 before

Question 6. Create box plots for GRE and GPA

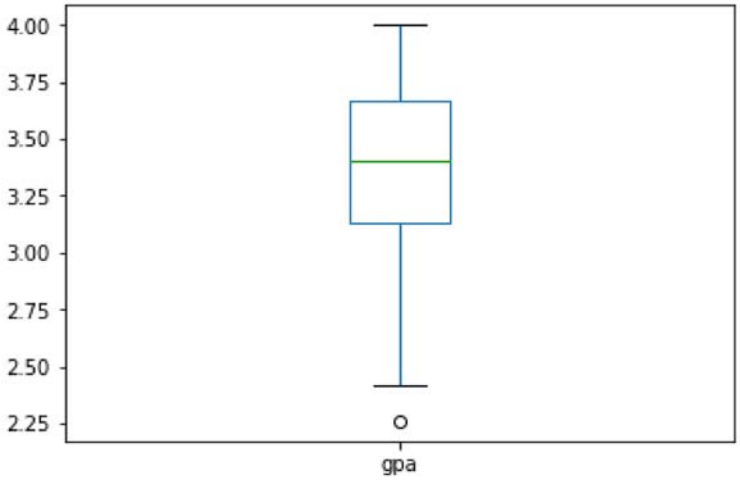
In [11]:
```
#boxplot 1
df_clean['gre'].plot(kind='box')
```

Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0xd6e5f98>



In [12]:
```
#boxplot 2
df_clean['gpa'].plot(kind='box')
```

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0xd92d1d0>



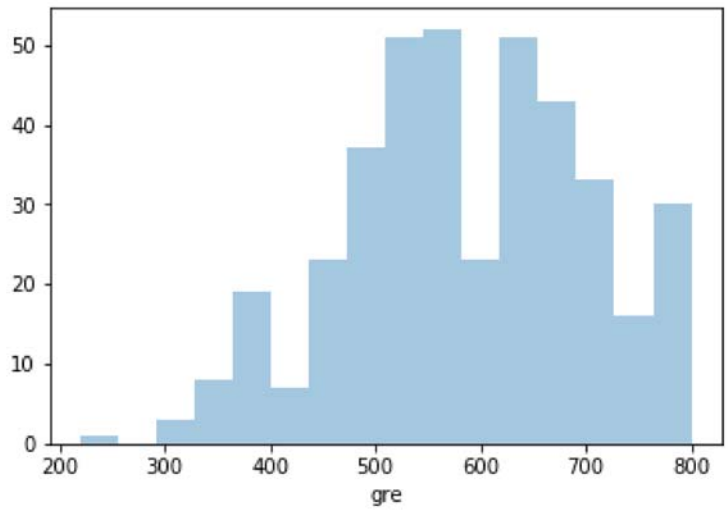
Question 7. What do this plots show?

Answer: The plot shows that GRE has 2 outliers on the lower end with the circles. The plot shows GPA has 1 outlier on the lower end, and there is skewness towards higher GPA values.

Question 8. Describe each distribution

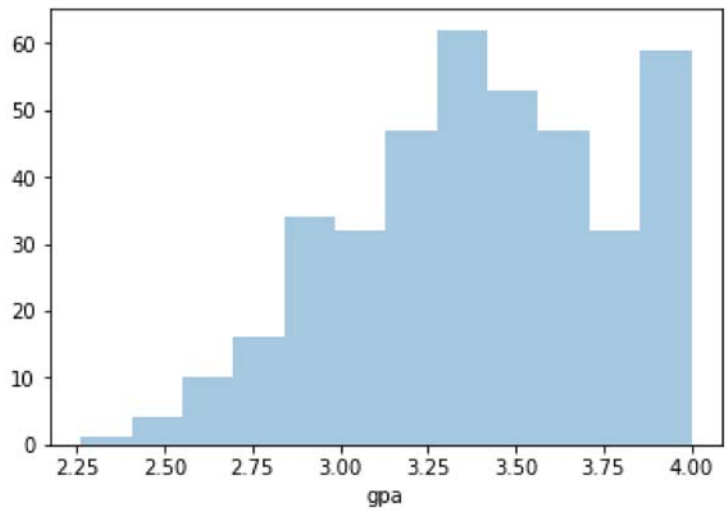
In [13]:
```
# plot the distribution of each variable

import seaborn as sns
sns.distplot(df_clean['gre'], kde=False)
```

Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0xde5b7f0>



In [14]:
```
sns.distplot(df_clean['gpa'], kde=False)
```

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0xdf7a940>



Question 9. If our model had an assumption of a normal distribution would we meet that requirement?

Answer: Looking at the histograms, both don't look exactly normal, however GRE seems more symmetric compared to GPA, which skews left.

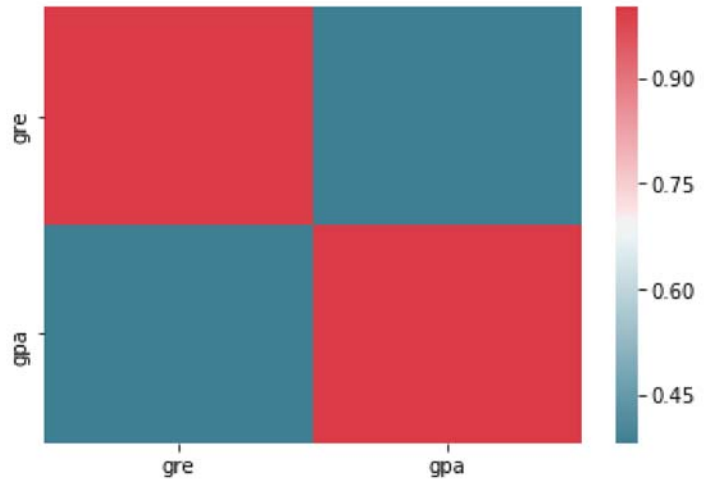
Question 10. Does this distribution need correction? If so, why? How?

Answer: To normalize the distribution, we can take the log.

Question 11. Which of our variables are potentially colinear?

In [15]:
```
# create a correlation matrix for the data

cmap = sns.diverging_palette(220, 10, as_cmap =True)

correlations = df_clean[['gre', 'gpa']].corr()
print correlations
print sns.heatmap(correlations, cmap=cmap)
```

```
          gre       gpa
gre  1.000000  0.382408
gpa  0.382408  1.000000
Axes(0.125,0.125;0.62x0.755)
```



Question 12. What did you find?

Answer: The correlation for the two variables is 0.382408

Question 13. Write an analysis plan for exploring the association between grad school admissions rates and prestige of undergraduate schools.

Answer: To explore the association between grad school admissions rate and prestige of undergraduate schools, we will look at the correlation between grad school admissions acceptance rate and a undergraduate school rankings (prestige).

Question 14. What is your hypothesis?

Answer: Hypothesis is that the two are positively correlated: The higher the prestige of undergraduate schools, the higher the admissions rate for grad school.

Bonus/Advanced

1. Bonus: Explore alternatives to dropping obervations with missing data

```
In [16]: df_raw.describe()
```

Out[16]:

	admit	gre	gpa	prestige
count	400.000000	398.000000	398.00000	399.000000
mean	0.317500	588.040201	3.39093	2.486216
std	0.466087	115.628513	0.38063	0.945333
min	0.000000	220.000000	2.26000	1.000000
25%	0.000000	520.000000	3.13000	2.000000
50%	0.000000	580.000000	3.39500	2.000000
75%	1.000000	660.000000	3.67000	3.000000
max	1.000000	800.000000	4.00000	4.000000

In [17]:
```
df_raw.head(100)
```

Out[17]:

	admit	gre	gpa	prestige
0	0	380.0	3.61	3.0
1	1	660.0	3.67	3.0
2	1	800.0	4.00	1.0
3	1	640.0	3.19	4.0
4	0	520.0	2.93	4.0
5	1	760.0	3.00	2.0
6	1	560.0	2.98	1.0
7	0	400.0	3.08	2.0
8	1	540.0	3.39	3.0
9	0	700.0	3.92	2.0
10	0	800.0	4.00	4.0
11	0	440.0	3.22	1.0
12	1	760.0	4.00	1.0
13	0	700.0	3.08	2.0
14	1	700.0	4.00	1.0
15	0	480.0	3.44	3.0
16	0	780.0	3.87	4.0
17	0	360.0	2.56	3.0
18	0	800.0	3.75	2.0
19	1	540.0	3.81	1.0
20	0	500.0	3.17	3.0
21	1	660.0	3.63	2.0
22	0	600.0	2.82	4.0
23	0	680.0	3.19	4.0
24	1	760.0	3.35	2.0
25	1	800.0	3.66	1.0
26	1	620.0	3.61	1.0
27	1	520.0	3.74	4.0
28	1	780.0	3.22	2.0
29	0	520.0	3.29	1.0
...	...	...	...	...
70	0	640.0	4.00	3.0

	admit	gre	gpa	prestige
71	0	300.0	2.92	4.0
72	0	480.0	3.39	4.0
73	0	580.0	4.00	2.0
74	0	720.0	3.45	4.0
75	0	720.0	4.00	3.0
76	0	560.0	3.36	3.0
77	1	800.0	4.00	3.0
78	0	540.0	3.12	1.0
79	1	620.0	4.00	1.0
80	0	700.0	2.90	4.0
81	0	620.0	3.07	2.0
82	0	500.0	2.71	2.0
83	0	380.0	2.91	4.0
84	1	500.0	3.60	3.0
85	0	520.0	2.98	2.0
86	0	600.0	3.32	2.0
87	0	600.0	3.48	2.0
88	0	700.0	3.28	1.0
89	1	660.0	4.00	2.0
90	0	700.0	3.83	2.0
91	1	720.0	3.64	1.0
92	0	800.0	3.90	2.0
93	0	580.0	2.93	2.0
94	1	660.0	3.44	2.0
95	0	660.0	3.33	2.0
96	0	640.0	3.52	4.0
97	0	480.0	3.57	2.0
98	0	700.0	2.88	2.0
99	0	400.0	3.31	3.0

100 rows × 4 columns

In [18]:
```
#Will fill in missing data with zeros instead of dropping observation.
#Know that this worked because there are still  400 rows left.
df_raw.fillna(0)
print df_raw
```

	admit	gre	gpa	prestige
0	0	380.0	3.61	3.0
1	1	660.0	3.67	3.0
2	1	800.0	4.00	1.0
3	1	640.0	3.19	4.0
4	0	520.0	2.93	4.0
5	1	760.0	3.00	2.0
6	1	560.0	2.98	1.0
7	0	400.0	3.08	2.0
8	1	540.0	3.39	3.0
9	0	700.0	3.92	2.0
10	0	800.0	4.00	4.0
11	0	440.0	3.22	1.0
12	1	760.0	4.00	1.0
13	0	700.0	3.08	2.0
14	1	700.0	4.00	1.0
15	0	480.0	3.44	3.0
16	0	780.0	3.87	4.0
17	0	360.0	2.56	3.0
18	0	800.0	3.75	2.0
19	1	540.0	3.81	1.0
20	0	500.0	3.17	3.0
21	1	660.0	3.63	2.0
22	0	600.0	2.82	4.0
23	0	680.0	3.19	4.0
24	1	760.0	3.35	2.0
25	1	800.0	3.66	1.0
26	1	620.0	3.61	1.0
27	1	520.0	3.74	4.0
28	1	780.0	3.22	2.0
29	0	520.0	3.29	1.0
..	...	...	...	...
370	1	540.0	3.77	2.0
371	1	680.0	3.76	3.0
372	1	680.0	2.42	1.0
373	1	620.0	3.37	1.0
374	0	560.0	3.78	2.0
375	0	560.0	3.49	4.0
376	0	620.0	3.63	2.0
377	1	800.0	4.00	2.0
378	0	640.0	3.12	3.0
379	0	540.0	2.70	2.0
380	0	700.0	3.65	2.0
381	1	540.0	3.49	2.0
382	0	540.0	3.51	2.0
383	0	660.0	4.00	1.0
384	1	480.0	2.62	2.0
385	0	420.0	3.02	1.0
386	1	740.0	3.86	2.0
387	0	580.0	3.36	2.0
388	0	640.0	3.17	2.0
389	0	640.0	3.51	2.0
390	1	800.0	3.05	2.0
391	1	660.0	3.88	2.0
392	1	600.0	3.38	3.0
393	1	620.0	3.75	2.0
394	1	460.0	3.99	3.0

```
395      0  620.0  4.00       2.0
396      0  560.0  3.04       3.0
397      0  460.0  2.63       2.0
398      0  700.0  3.65       2.0
399      0  600.0  3.89       3.0

[400 rows x 4 columns]
```

2. Bonus: Log transform the skewed data

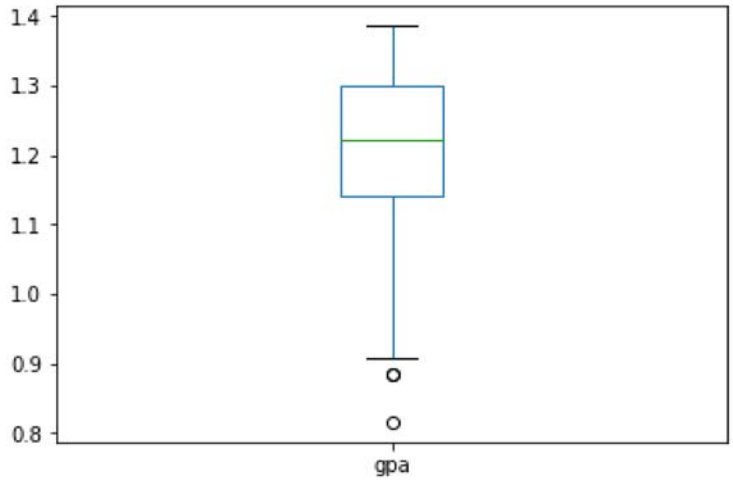
In [32]:
```
#GPA is the column that is clearly skewed left
df_clean.head()
```

Out[32]:

	admit	gre	gpa	prestige
0	0	380.0	3.61	3.0
1	1	660.0	3.67	3.0
2	1	800.0	4.00	1.0
3	1	640.0	3.19	4.0
4	0	520.0	2.93	4.0

In [33]:
```
np.log(df_clean['gpa']).plot(kind='box')
```

Out[33]: `<matplotlib.axes._subplots.AxesSubplot at 0xe950828>`



3. Advanced: Impute missing data

In [31]:
```
#As we did in problem #1, filling in missing data as 0
df_raw.fillna(0)
```

Out[31]:

	admit	gre	gpa	prestige
0	0	380.0	3.61	3.0
1	1	660.0	3.67	3.0
2	1	800.0	4.00	1.0
3	1	640.0	3.19	4.0
4	0	520.0	2.93	4.0
5	1	760.0	3.00	2.0
6	1	560.0	2.98	1.0
7	0	400.0	3.08	2.0
8	1	540.0	3.39	3.0
9	0	700.0	3.92	2.0
10	0	800.0	4.00	4.0
11	0	440.0	3.22	1.0
12	1	760.0	4.00	1.0
13	0	700.0	3.08	2.0
14	1	700.0	4.00	1.0
15	0	480.0	3.44	3.0
16	0	780.0	3.87	4.0
17	0	360.0	2.56	3.0
18	0	800.0	3.75	2.0
19	1	540.0	3.81	1.0
20	0	500.0	3.17	3.0
21	1	660.0	3.63	2.0
22	0	600.0	2.82	4.0
23	0	680.0	3.19	4.0
24	1	760.0	3.35	2.0
25	1	800.0	3.66	1.0
26	1	620.0	3.61	1.0
27	1	520.0	3.74	4.0
28	1	780.0	3.22	2.0
29	0	520.0	3.29	1.0
...	...	...	...	...
370	1	540.0	3.77	2.0

	admit	gre	gpa	prestige
371	1	680.0	3.76	3.0
372	1	680.0	2.42	1.0
373	1	620.0	3.37	1.0
374	0	560.0	3.78	2.0
375	0	560.0	3.49	4.0
376	0	620.0	3.63	2.0
377	1	800.0	4.00	2.0
378	0	640.0	3.12	3.0
379	0	540.0	2.70	2.0
380	0	700.0	3.65	2.0
381	1	540.0	3.49	2.0
382	0	540.0	3.51	2.0
383	0	660.0	4.00	1.0
384	1	480.0	2.62	2.0
385	0	420.0	3.02	1.0
386	1	740.0	3.86	2.0
387	0	580.0	3.36	2.0
388	0	640.0	3.17	2.0
389	0	640.0	3.51	2.0
390	1	800.0	3.05	2.0
391	1	660.0	3.88	2.0
392	1	600.0	3.38	3.0
393	1	620.0	3.75	2.0
394	1	460.0	3.99	3.0
395	0	620.0	4.00	2.0
396	0	560.0	3.04	3.0
397	0	460.0	2.63	2.0
398	0	700.0	3.65	2.0
399	0	600.0	3.89	3.0

400 rows × 4 columns