Name: Shubhangi Manik Mistari
Roll No. 2414

ASSIGNMENT NO.1 (ST-425)

This exercise relates to the College data set, which can be found in the file College.csv. It contains a number of variables for 777 different universities and colleges in the US.

```
import pandas as pd
```

1.Use the read.csv() function to read the data into python jupiter. Call the loaded data college.

```
college=pd.read_csv("college_data.csv")

college.isnull().sum()
```

```
Private        0
Apps           0
Accept         0
Enroll         0
Top10perc      0
Top25perc      0
F.Undergrad    0
P.Undergrad    0
Outstate       0
Room.Board     0
Books          0
Personal       0
PhD            0
Terminal       0
S.F.Ratio      0
perc.alumni    0
Expend         0
Grad.Rate      0
Elite          0
dtype: int64
```

```
college=college[:-1]

college
```

	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad
0	Yes	1660	1232	721	23	52	2885
1	Yes	2186	1924	512	16	29	2683
2	Yes	1428	1097	336	22	50	1036
3	Yes	417	349	137	60	89	510
4	Yes	193	146	55	16	44	249

```
..      ...    ...     ...     ...         ...         ...          ...
771     Yes   2768    2314     682          49          86         2802
772      No   2197    1515     543           4          26         3089
773     Yes   1959    1805     695          24          47         2849
774     Yes   2097    1915     695          34          61         2793
775     Yes  10705    2453    1317          95          99         5217


     P.Undergrad  Outstate  Room.Board  Books  Personal  PhD  Terminal
\
0            537      7440        3300    450      2200   70        78
1           1227     12280        6450    750      1500   29        30
2             99     11250        3750    400      1165   53        66
3             63     12960        5450    450       875   92        97
4            869      7560        4120    800      1500   76        72
..           ...       ...         ...    ...       ...  ...       ...
771           86     15884        5370    530       730   92        94
772         2029      6797        3900    500      1200   60        60
773         1107     11520        4960    600      1250   73        75
774          166      6900        4200    617       781   67        75
775           83     19840        6510    630      2115   96        96


     S.F.Ratio  perc.alumni  Expend  Grad.Rate
0         18.1           12    7041         60
1         12.2           16   10527         56
2         12.9           30    8735         54
3          7.7           37   19016         59
4         11.9            2   10922         15
..         ...          ...     ...        ...
771       15.2           34   10774         82
772       21.0           14    4469         40
773       13.3           31    9189         83
774       14.4           20    8323         49
775        5.8           49   40386         99
```

```
[776 rows x 18 columns]
```

i. Get numerical summary of the variables in the data set

```
print(college.describe())
```

```
                Apps        Accept       Enroll   Top10perc   Top25perc
\
count    776.000000    776.000000   776.000000  776.000000  776.000000

mean    3001.654639   2019.015464   780.087629   27.557990   55.787371

std     3872.697557   2452.687758   929.769965   17.651734   19.815859

min       81.000000     72.000000    35.000000    1.000000    9.000000

25%      776.000000    603.250000   242.000000   15.000000   41.000000

50%     1557.500000   1109.500000   434.000000   23.000000   54.000000

75%     3629.500000   2428.000000   902.250000   35.000000   69.000000

max    48094.000000  26330.000000  6392.000000   96.000000  100.000000


        F.Undergrad   P.Undergrad      Outstate   Room.Board
Books  \
count    776.000000    776.000000    776.000000   776.000000
776.000000
mean    3700.824742    854.176546  10447.693299  4358.554124
549.444588
std     4853.481370   1523.092275   4020.840763  1097.029256
165.202310
min      139.000000      1.000000   2340.000000  1780.000000
96.000000
25%      991.000000     95.000000   7338.000000  3597.750000
469.500000
50%     1707.000000    352.500000   9990.000000  4200.000000
500.000000
75%     4030.250000    964.000000  12931.250000  5050.000000
600.000000
max    31643.000000  21836.000000  21700.000000  8124.000000
2340.000000

          Personal         PhD    Terminal   S.F.Ratio  perc.alumni  \
count   776.000000  776.000000  776.000000  776.000000   776.000000
mean   1340.759021   72.657216   79.708763   14.084536    22.737113
std     677.500299   16.338469   14.730884    3.958278    12.398354
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
```

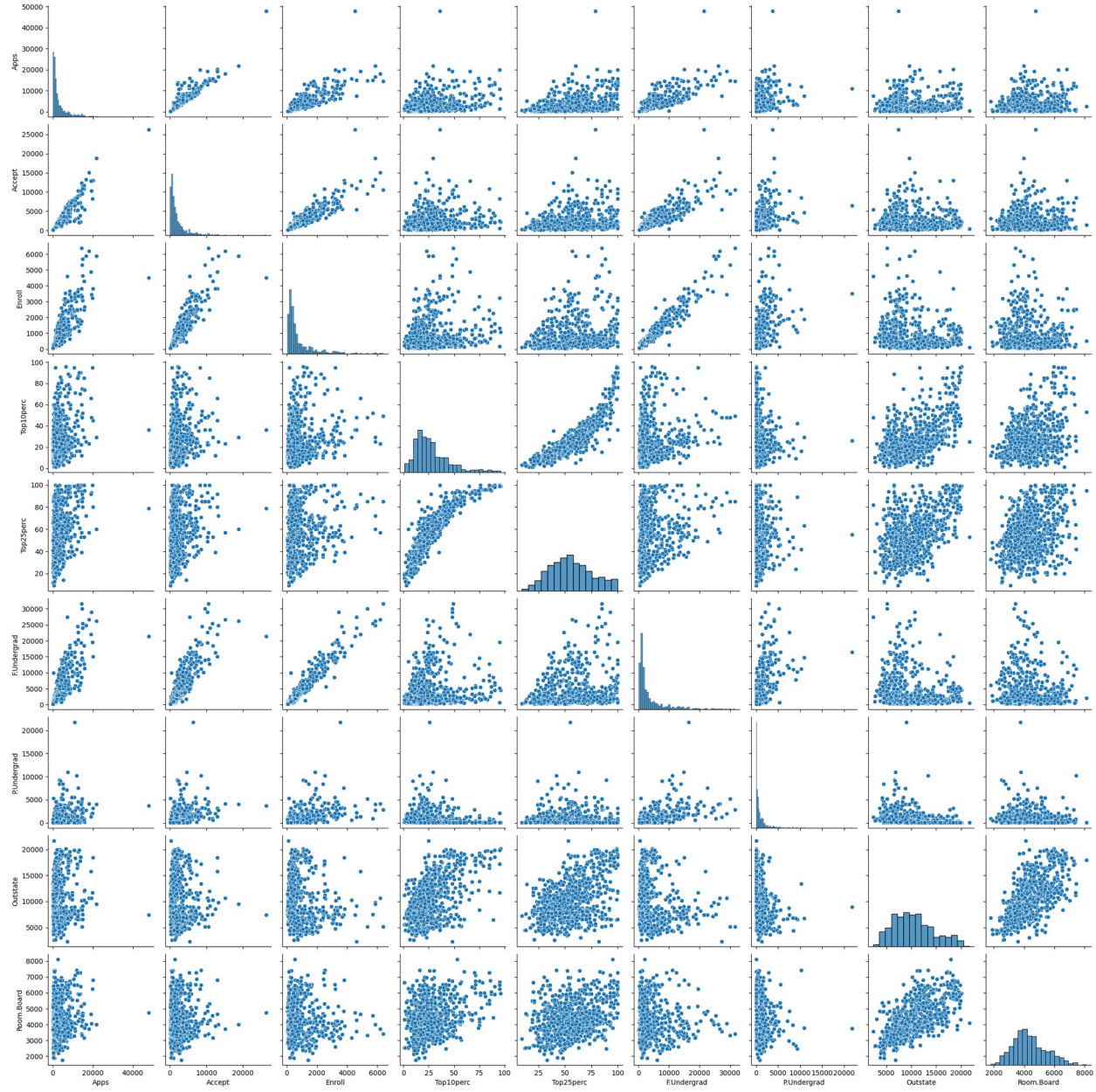
```
              Expend   Grad.Rate
count     776.000000  776.000000
mean     9666.809278   65.420103
std      5221.854692   17.146468
min      3186.000000   10.000000
25%      6755.500000   53.000000
50%      8392.500000   65.000000
75%     10838.500000   78.000000
max     56233.000000  118.000000
```

```
print(college.describe(include=['object'])) #summary of catagorical variable (private)
```

```
       Private
count      776
unique       2
top        Yes
freq       564
```

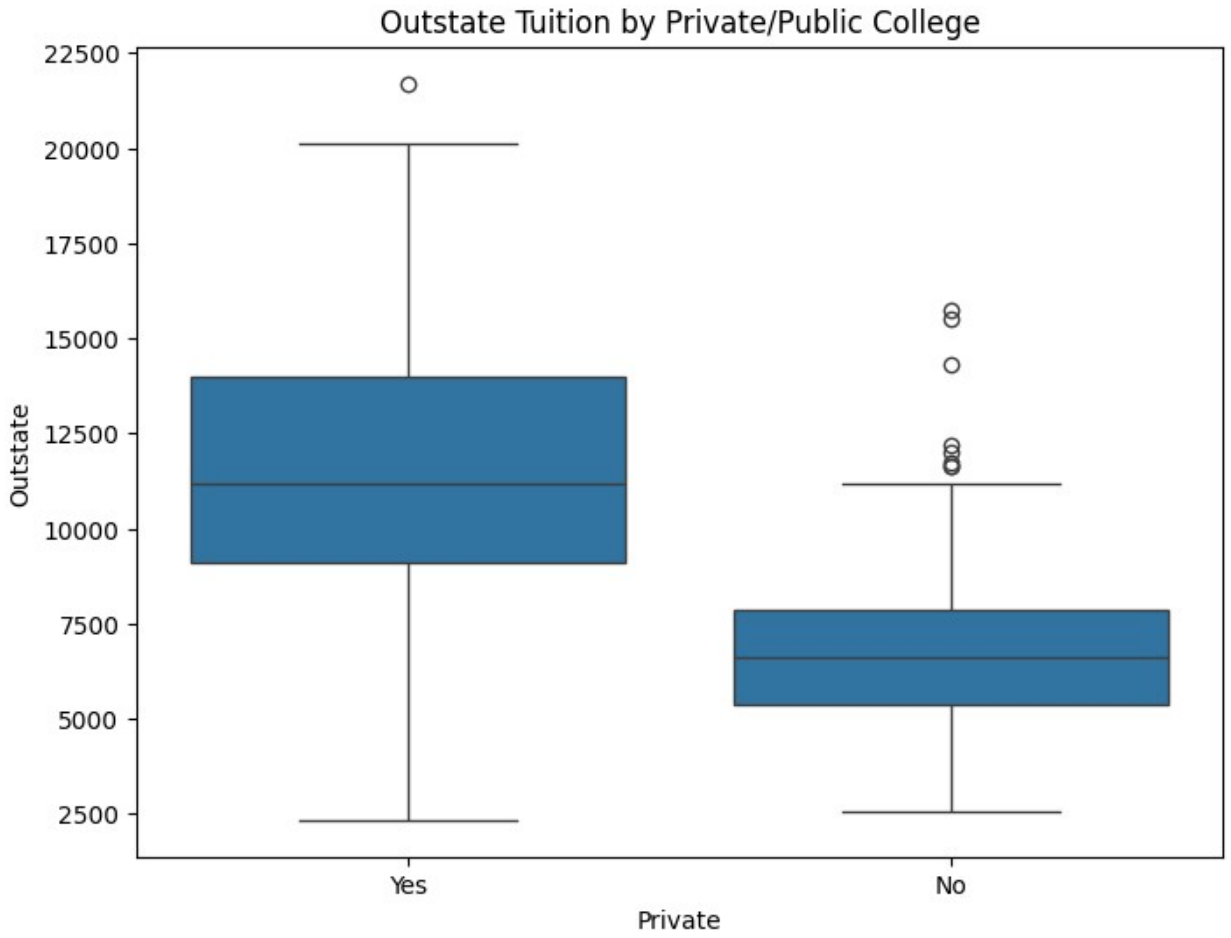
ii. Use the pairs() function to produce a scatterplot matrix of the first ten columns or variables of the data. Recall that you can reference the first ten columns of a matrix using A[,1:10].

```
import seaborn as sns
import matplotlib.pyplot as plt

college=pd.read_csv("college_data.csv")

college_subset=college.iloc[:,:10]

sns.pairplot(college_subset)
plt.show()
```



iii. Use the plot() function to produce side-by-side boxplots of Outstate versus Private.

```
plt.figure(figsize=(8,6))
sns.boxplot(x="Private",y="Outstate",data=college)
plt.title("Outstate Tuition by Private/Public College")
plt.show()
```



iv. Create a new qualitative variable, called Elite, by binning the Top10perc variable. We are going to divide universities into two groups based on whether or not the proportion of students coming from the top 10% of their high schools classes exceeds 50%.

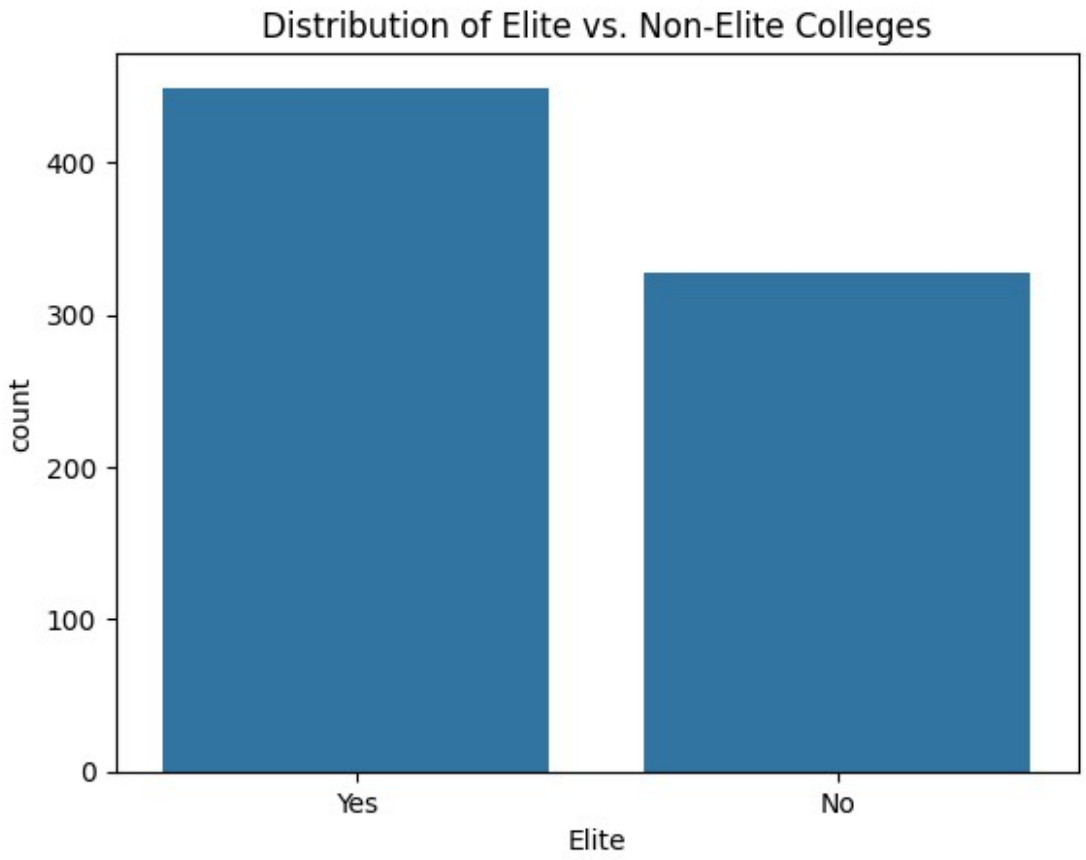
```
college['Elite']=college['Top25perc'].apply(lambda x: 'Yes' if x>50
else 'No')
```

Cheacking the Distribution of Elite Schools

```
print(college['Elite'].value_counts())
```

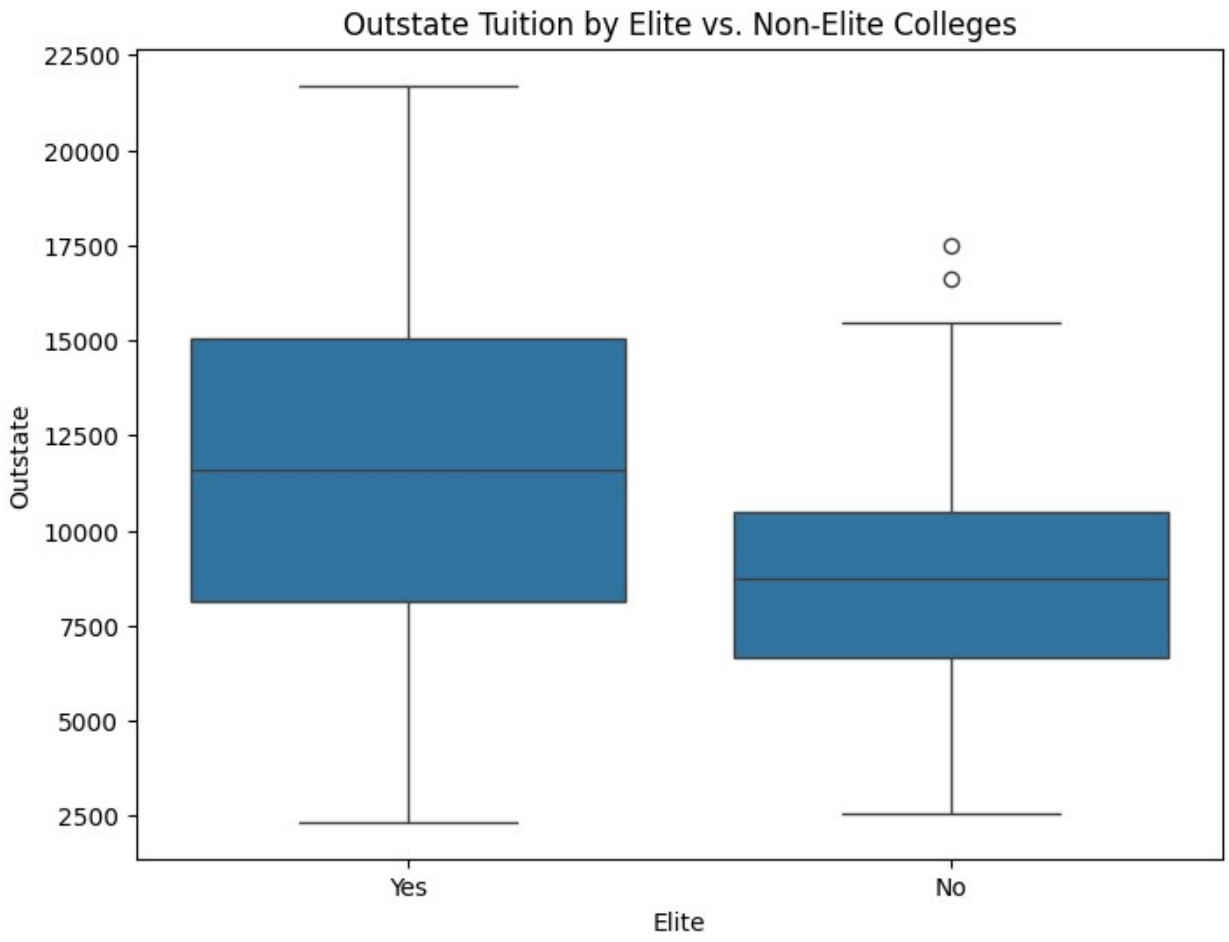
```
Elite
Yes    449
No     328
Name: count, dtype: int64
```

```
sns.countplot(x='Elite',data=college)
plt.title("Distribution of Elite vs. Non-Elite Colleges")
plt.show()
```

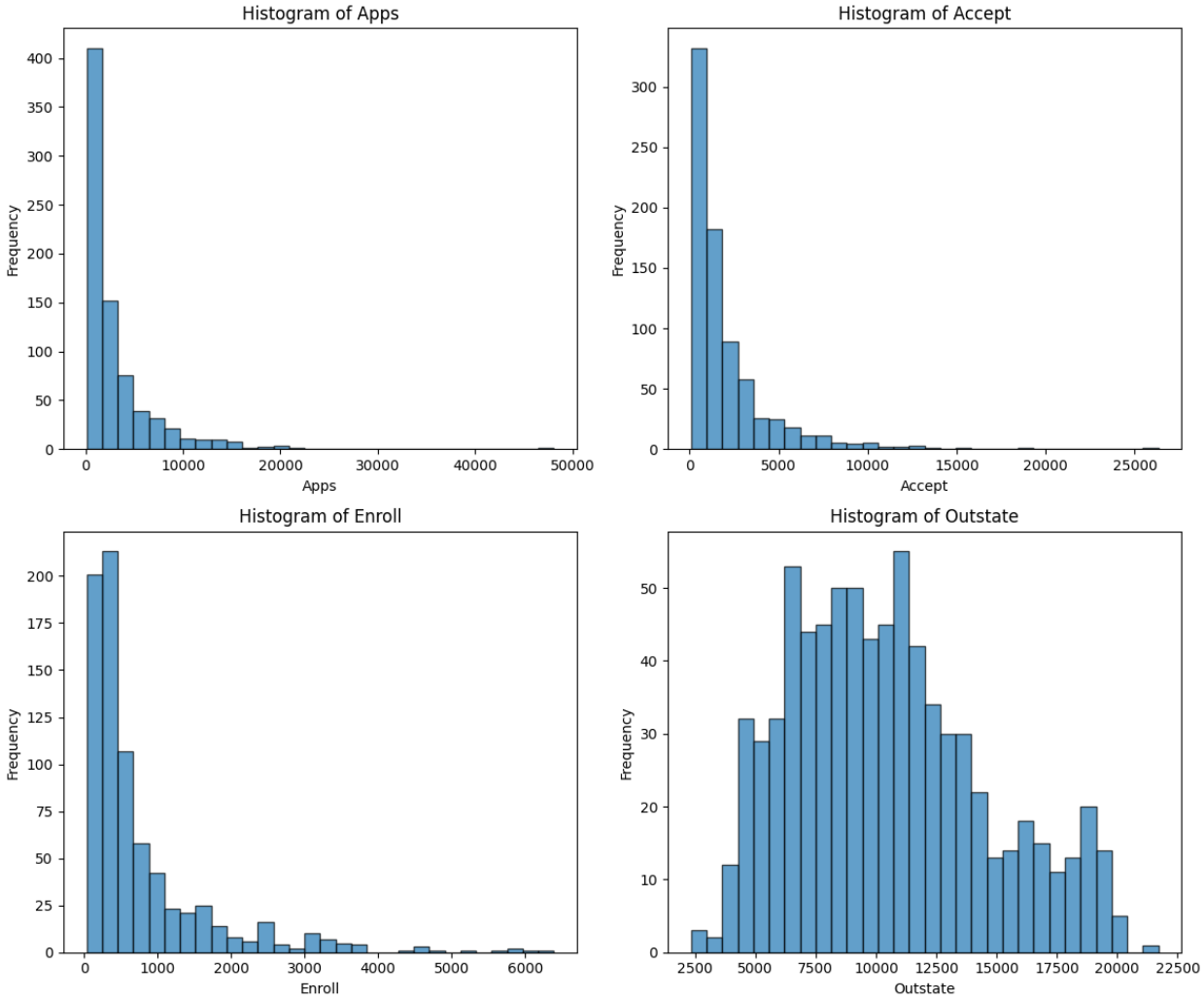


Now use the plot() function to produce side-by-side boxplots of Outstate versus Elite.

```
plt.figure(figsize=(8,6))
sns.boxplot(x="Elite",y="Outstate",data=college)
plt.title("Outstate Tuition by Elite vs. Non-Elite Colleges")
plt.show()
```




```
variables=["Apps","Accept","Enroll","Outstate"]

fig,axes=plt.subplots(2,2,figsize=(12,10))
for ax, var in zip (axes.flatten(),variables):
    ax.hist(college[var],bins=30,edgecolor="black",alpha=0.7)
    ax.set_title(f"Histogram of {var}")
    ax.set_xlabel(var)
    ax.set_ylabel("Frequency")
plt.tight_layout()
plt.show()
```




```
# Select only numerical variables for correlation analysis
num_vars = ["Apps", "Accept", "Enroll", "Outstate", "Expend",
"Room.Board",
 "Grad.Rate", "Top10perc", "S.F.Ratio", "PhD", "perc.alumni"]

# Compute correlation matrix
corr_matrix = college[num_vars].corr()

# Print correlation values
print("Top Correlations:\n", corr_matrix)

# Display highly correlated pairs (absolute correlation > 0.5)
strong_corrs = corr_matrix[(corr_matrix > 0.5) | (corr_matrix < -0.5)]
print("\nHighly Correlated Pairs (|r| > 0.5):\n",strong_corrs)
```

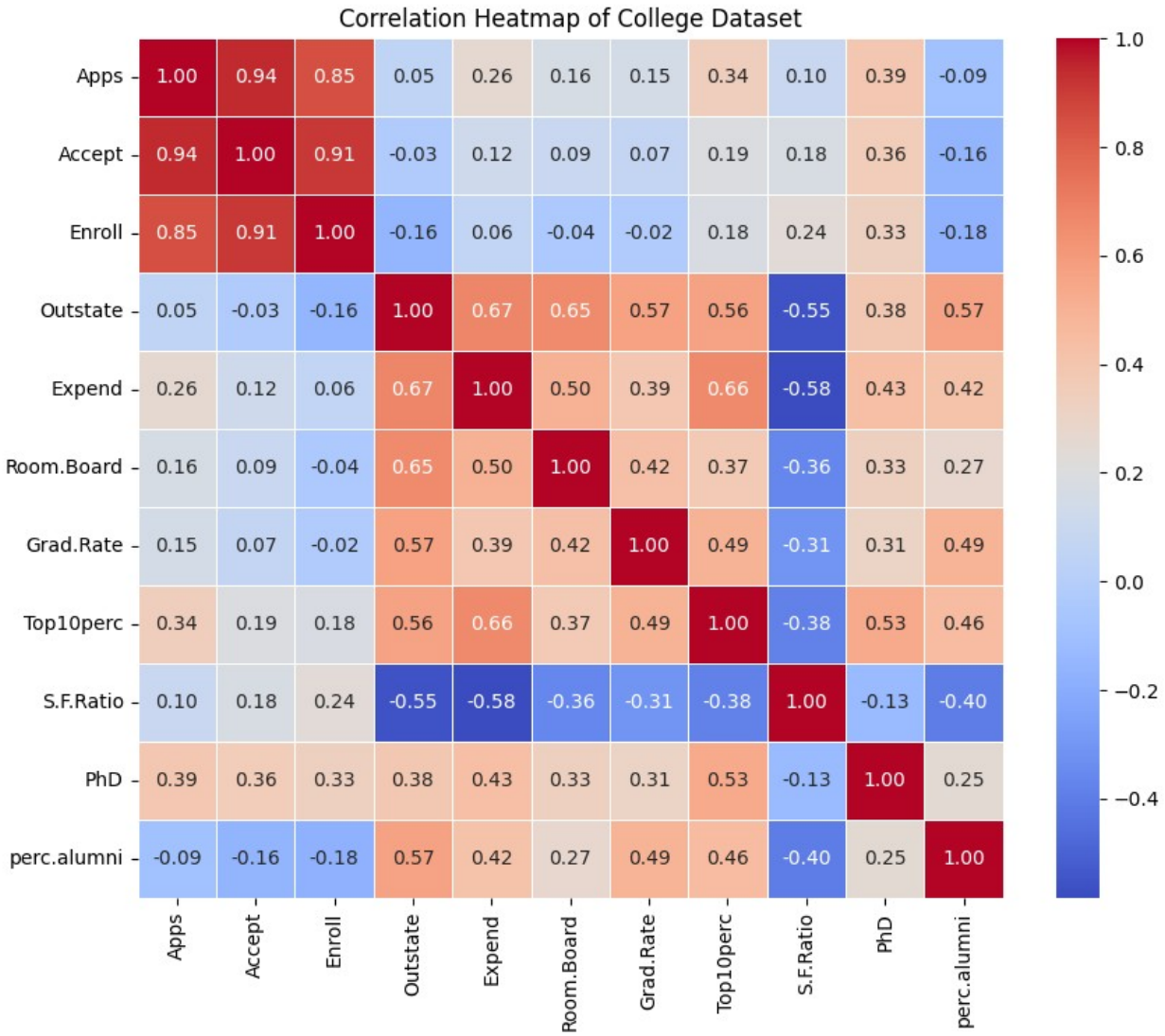
```
Top Correlations:
                 Apps    Accept    Enroll  Outstate    Expend
Room.Board  \
Apps         1.000000  0.943451  0.846822  0.050159  0.259592
```

```
0.164939
Accept       0.943451  1.000000  0.911637 -0.025755  0.124717
0.090899
Enroll       0.846822  0.911637  1.000000 -0.155477  0.064169   -
0.040232
Outstate     0.050159 -0.025755 -0.155477  1.000000  0.672779
0.654256
Expend       0.259592  0.124717  0.064169  0.672779  1.000000
0.501739
Room.Board   0.164939  0.090899 -0.040232  0.654256  0.501739
1.000000
Grad.Rate    0.146755  0.067313 -0.022341  0.571290  0.390343
0.424942
Top10perc    0.338834  0.192447  0.181294  0.562331  0.660913
0.371480
S.F.Ratio    0.095633  0.176229  0.237271 -0.554821 -0.583832   -
0.362628
PhD          0.390697  0.355758  0.331469  0.382982  0.432762
0.329202
perc.alumni -0.090226 -0.159990 -0.180794  0.566262  0.417712
0.272363

             Grad.Rate  Top10perc  S.F.Ratio       PhD  perc.alumni
Apps          0.146755   0.338834   0.095633  0.390697    -0.090226
Accept        0.067313   0.192447   0.176229  0.355758    -0.159990
Enroll       -0.022341   0.181294   0.237271  0.331469    -0.180794
Outstate      0.571290   0.562331  -0.554821  0.382982     0.566262
Expend        0.390343   0.660913  -0.583832  0.432762     0.417712
Room.Board    0.424942   0.371480  -0.362628  0.329202     0.272363
Grad.Rate     1.000000   0.494989  -0.306710  0.305038     0.490898
Top10perc     0.494989   1.000000  -0.384875  0.531828     0.455485
S.F.Ratio    -0.306710  -0.384875   1.000000 -0.130530    -0.402929
PhD           0.305038   0.531828  -0.130530  1.000000     0.249009
perc.alumni   0.490898   0.455485  -0.402929  0.249009     1.000000

Highly Correlated Pairs (|r| > 0.5):
                 Apps    Accept    Enroll  Outstate    Expend
Room.Board  \
Apps         1.000000  0.943451  0.846822       NaN       NaN
NaN
Accept       0.943451  1.000000  0.911637       NaN       NaN
NaN
Enroll       0.846822  0.911637  1.000000       NaN       NaN
NaN
Outstate          NaN       NaN       NaN  1.000000  0.672779
0.654256
Expend            NaN       NaN       NaN  0.672779  1.000000
0.501739
Room.Board        NaN       NaN       NaN  0.654256  0.501739
```

```
1.000000
Grad.Rate          NaN       NaN       NaN  0.571290       NaN
NaN
Top10perc          NaN       NaN       NaN  0.562331  0.660913
NaN
S.F.Ratio          NaN       NaN       NaN -0.554821 -0.583832
NaN
PhD                NaN       NaN       NaN       NaN       NaN
NaN
perc.alumni        NaN       NaN       NaN  0.566262       NaN
NaN

             Grad.Rate  Top10perc  S.F.Ratio       PhD  perc.alumni
Apps               NaN        NaN        NaN       NaN          NaN
Accept             NaN        NaN        NaN       NaN          NaN
Enroll             NaN        NaN        NaN       NaN          NaN
Outstate       0.57129   0.562331  -0.554821       NaN     0.566262
Expend             NaN   0.660913  -0.583832       NaN          NaN
Room.Board         NaN        NaN        NaN       NaN          NaN
Grad.Rate      1.00000        NaN        NaN       NaN          NaN
Top10perc          NaN   1.000000        NaN  0.531828          NaN
S.F.Ratio          NaN        NaN   1.000000       NaN          NaN
PhD                NaN   0.531828        NaN  1.000000          NaN
perc.alumni        NaN        NaN        NaN       NaN     1.000000
```

Heatmap of correlation matrix

```
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap="coolwarm", fmt=".2f",
linewidths=0.5)
plt.title("Correlation Heatmap of College Dataset")
plt.show()
```



This exercise involves the Auto data set studied in the lab. Make sure that the missing values have been removed from the data.

1) Which of the predictors are quantitative, and which are qualitative?

```
import pandas as pd

Auto=pd.read_csv("Auto_data.csv")
print(Auto.head())
```

```
    mpg  cylinders  displacement  horsepower  weight  acceleration
year  \
0  18.0          8         307.0         130    3504          12.0
70
1  15.0          8         350.0         165    3693          11.5
70
2  18.0          8         318.0         150    3436          11.0
```

```
70
3  16.0          8         304.0          150    3433           12.0
70
4  17.0          8         302.0          140    3449           10.5
70

   origin                       name
0       1  chevrolet chevelle malibu
1       1          buick skylark 320
2       1         plymouth satellite
3       1              amc rebel sst
4       1                ford torino
```

```
# Load the Auto dataset (replace 'Auto.csv' with the correct file
path)
df = pd.read_csv("Auto_data.csv")
df=pd.DataFrame(df)

df.isnull().sum()
```

```
mpg             0
cylinders       0
displacement    0
horsepower      0
weight          0
acceleration    0
year            0
origin          0
name            0
dtype: int64
```

(b) What is the range of each quantitative predictor?

```
# Separate quantitative and qualitative variables
quanti_vars = df.select_dtypes(include=['int64',
'float64']).columns.tolist()
quali_vars = df.select_dtypes(include=['object']).columns.tolist()

print("Quantitative Variables:", quanti_vars)
print("Qualitative Variables:", quali_vars)
```

```
Quantitative Variables: ['mpg', 'cylinders', 'displacement',
'horsepower', 'weight', 'acceleration', 'year', 'origin']
Qualitative Variables: ['name']
```

Summary of Quntitative variables

```
print(df.describe())
```

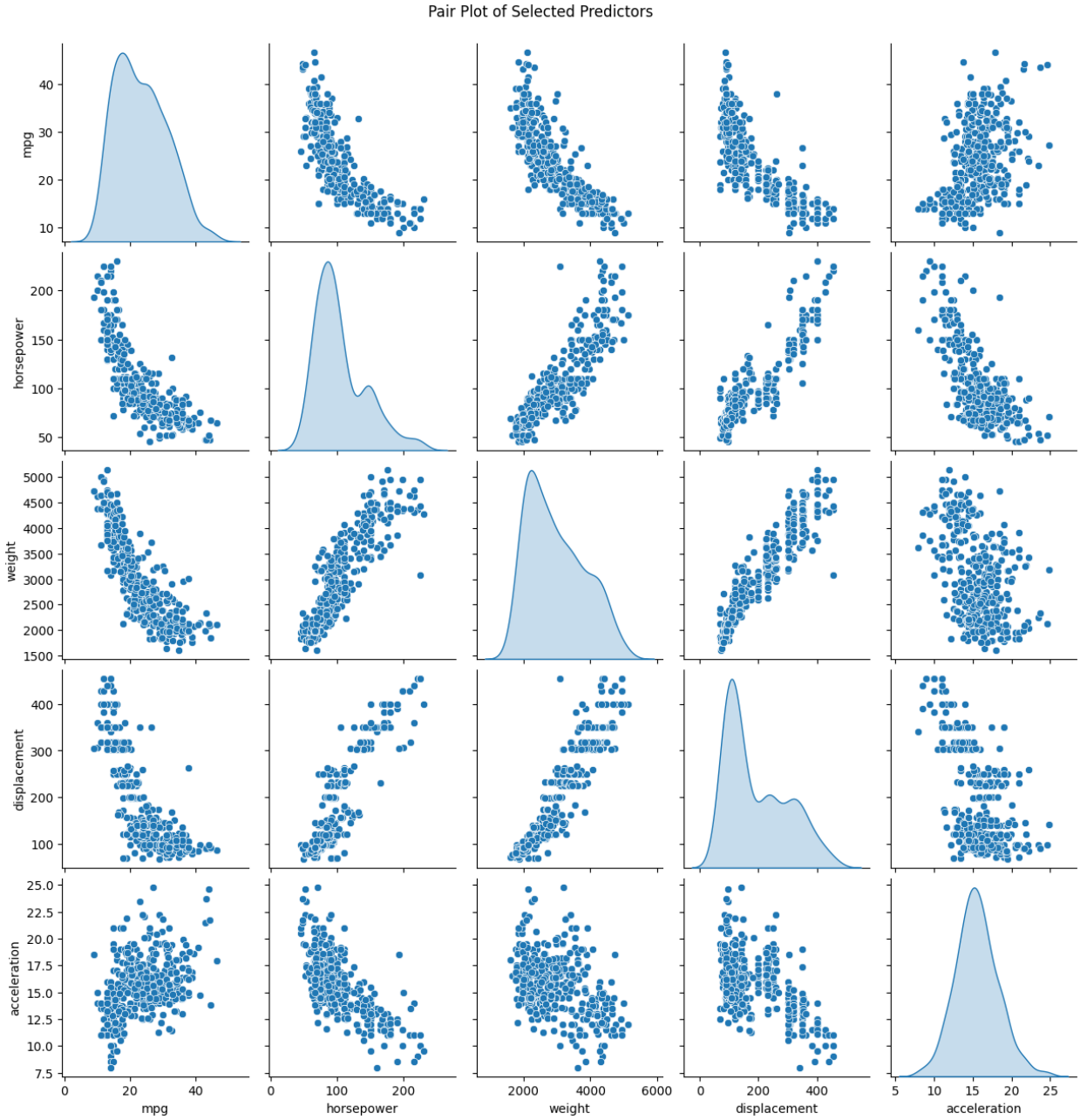
```
              mpg   cylinders  displacement  horsepower
weight  \
count  392.000000  392.000000    392.000000  392.000000   392.000000

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


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

```
quanti_vars = df.select_dtypes(include=['int64', 'float64'])

# Remove 10th through 85th observations (Python uses zero-based
indexing)
df_subset = quanti_vars.drop(index=range(10, 86))

range_values = df_subset.max() - df_subset.min() #range (Max - Min)
mean_values = df_subset.mean() #mean
std_values = df_subset.std() #standard deviation

# Display the results
print("Range of Each Quantitative Predictor (After Removing 10th-85th
Observations):")
print(range_values)
print("\nMean of Each Quantitative Predictor:")
print(mean_values)
print("\nStandard Deviation of Each Quantitative Predictor:")
print(std_values)
```

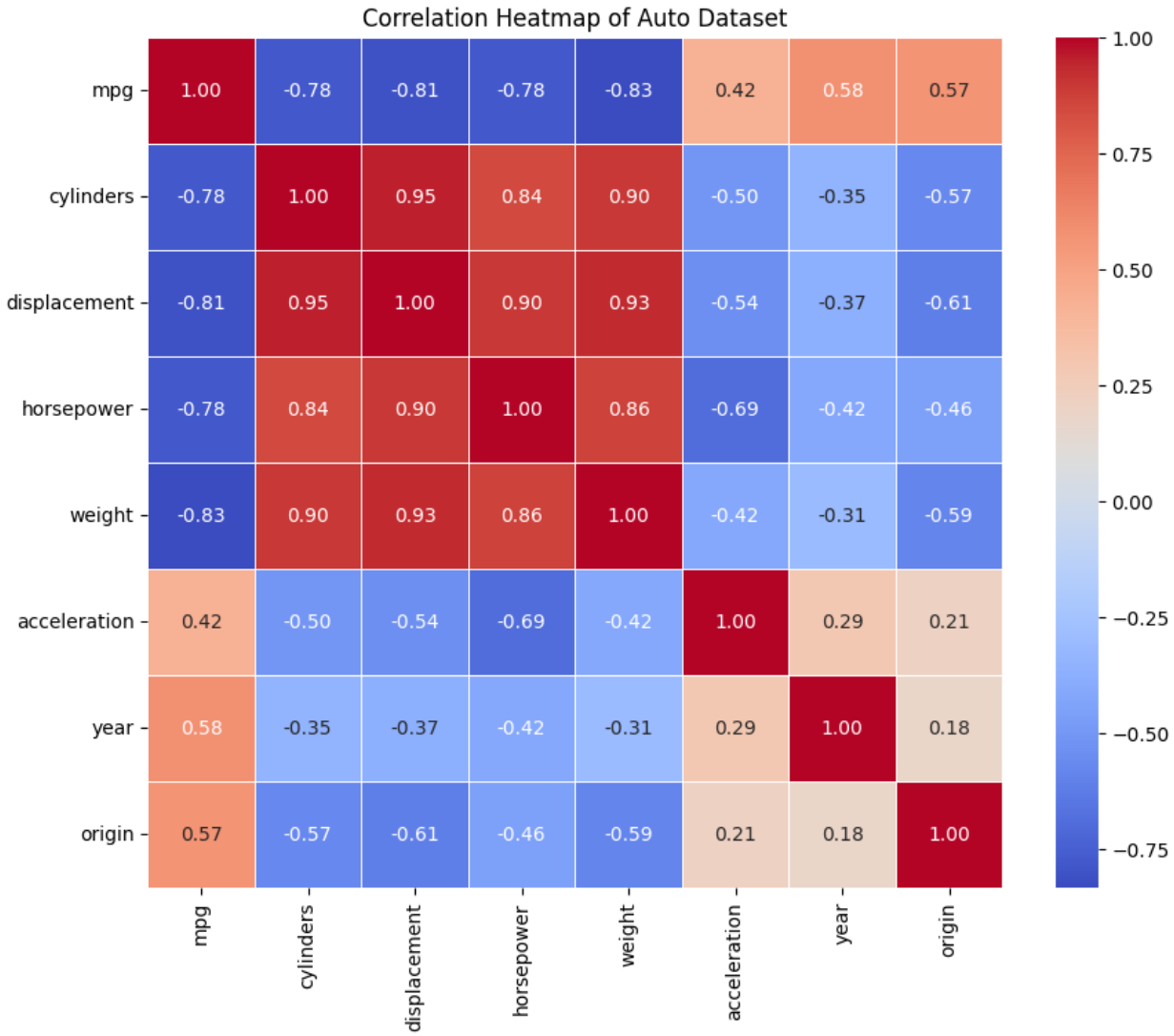
```
Range of Each Quantitative Predictor (After Removing 10th-85th
Observations):
mpg               35.6
```

```
cylinders          5.0
displacement     387.0
horsepower       184.0
weight          3348.0
acceleration      16.3
year              12.0
origin             2.0
dtype: float64

Mean of Each Quantitative Predictor:
mpg               24.407595
cylinders          5.373418
displacement     187.512658
horsepower       100.848101
weight          2936.534810
acceleration      15.717405
year              77.136076
origin             1.601266
dtype: float64

Standard Deviation of Each Quantitative Predictor:
mpg               7.863285
cylinders         1.654179
displacement    100.114616
horsepower       35.954147
weight          811.874450
acceleration      2.713876
year              3.123464
origin            0.819910
dtype: float64
```

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv("Auto_data.csv")

# Drop non-numeric columns (like 'name')
df_numeric = df.select_dtypes(include=['int64', 'float64'])

# □ Scatter Plot Matrix (Pair Plot) for Key Predictors
sns.pairplot(df_numeric[['mpg', 'horsepower', 'weight',
'displacement', 'acceleration']], diag_kind='kde')
plt.suptitle("Pair Plot of Selected Predictors", y=1.02)
plt.show()
```




```
#Correlation Heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(df_numeric.corr(), annot=True, cmap="coolwarm", fmt=".2f",
linewidths=0.5)
plt.title("Correlation Heatmap of Auto Dataset")
plt.show()
```



d) Suppose that we wish to predict gas mileage (mpg) on the basis of the other variables. Do your plots suggest that any of the other variables might be useful in predicting mpg? Justify your answer.

#To determine which variables are useful for predicting mpg, we analyzed:

a) Scatter plots to visualize relationships.

b) Correlation heatmap to check linear dependencies.

1. Strong Negative Correlations with mpg weight have Strongest negative correlation (-0.83). Heavier cars tend to have lower mpg.

horsepower have Strong negative correlation (-0.78). More powerful engines reduce mpg.

displacement Highly negative correlation (-0.79). Larger engines generally consume more fuel.

Weight, Horsepower, and Displacement are the best predictors for mpg, as they show strong negative relationships.

Year is also useful since newer cars tend to have better mpg.

Acceleration is a weaker predictor and might not significantly impact mpg.