Question (a)

Quantitative Predictors:

1. mpg
2. cylinders
3. displacement
4. horsepower
5. weight
6. acceleration

Qualitative Predictor:

1. name
2. year
3. origin

In [1]:

```
import numpy as np
import pandas as pd
auto_data = pd.read_csv('auto.csv')
auto_data
```

Out[1]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	year	origin	name
0	18.0	8	307.0	130	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150	3433	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140	3449	10.5	70	1	ford torino
...	...	...	...	...	...	...	...	...	...
392	27.0	4	140.0	86	2790	15.6	82	1	ford mustang gl
393	44.0	4	97.0	52	2130	24.6	82	2	vw pickup
394	32.0	4	135.0	84	2295	11.6	82	1	dodge rampage
395	28.0	4	120.0	79	2625	18.6	82	1	ford ranger
396	31.0	4	119.0	82	2720	19.4	82	1	chevy s-10

397 rows × 9 columns

In [2]:

```
auto_data['horsepower'] = pd.to_numeric(auto_data['horsepower'], errors='coerce')
auto_data_cleaned = auto_data.dropna()
auto_data_cleaned
```

Out[2]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	year	origin	name
0	18.0	8	307.0	130.0	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150.0	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150.0	3433	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140.0	3449	10.5	70	1	ford torino
...	...	...	...	...	...	...	...	...	...
392	27.0	4	140.0	86.0	2790	15.6	82	1	ford mustang gl
393	44.0	4	97.0	52.0	2130	24.6	82	2	vw pickup
394	32.0	4	135.0	84.0	2295	11.6	82	1	dodge rampage
395	28.0	4	120.0	79.0	2625	18.6	82	1	ford ranger
396	31.0	4	119.0	82.0	2720	19.4	82	1	chevy s-10

392 rows × 9 columns

In [3]:

```
numpy_array=auto_data_cleaned[['mpg','cylinders','displacement','horsepower','weight','acceleration']].values
numpy_array
```

Out[3]:

```
array([[  18. ,    8. ,  307. ,  130. , 3504. ,   12. ],
       [  15. ,    8. ,  350. ,  165. , 3693. ,   11.5],
       [  18. ,    8. ,  318. ,  150. , 3436. ,   11. ],
       ...,
       [  32. ,    4. ,  135. ,   84. , 2295. ,   11.6],
       [  28. ,    4. ,  120. ,   79. , 2625. ,   18.6],
       [  31. ,    4. ,  119. ,   82. , 2720. ,   19.4]])
```

Question (b)

In [4]:

```
max_values_columnwise = np.max(numpy_array, axis=0)
```

```
min_values_columnwise = np.min(numpy_array, axis=0)
```

In [5]:
```
range=pd.DataFrame({'Minimum Value':min_values_columnwise,'Maximum value':max_values_columnwise,'Range':max_values_co
                   index=['mpg','cylinders','displacement','horsepower','weight','acceleration'])
range
```

Out[5]:

	Minimum Value	Maximum value	Range
mpg	9.0	46.6	37.6
cylinders	3.0	8.0	5.0
displacement	68.0	455.0	387.0
horsepower	46.0	230.0	184.0
weight	1613.0	5140.0	3527.0
acceleration	8.0	24.8	16.8

Question (c)

In [6]:
```
mean=np.mean(numpy_array, axis=0)
std_dev=np.std(numpy_array, axis=0)
```

In [7]:
```
Qc=pd.DataFrame({'Mean':mean, 'Standard Deviation':std_dev}, index=['mpg','cylinders','displacement','horsepower','we
Qc
```

Out[7]:

	Mean	Standard Deviation
mpg	23.445918	7.795046
cylinders	5.471939	1.703606
displacement	194.411990	104.510444
horsepower	104.469388	38.442033
weight	2977.584184	848.318447
acceleration	15.541327	2.755343

Question (d)

In [8]:
```
numpy_array_rem=np.delete(numpy_array, np.s_[9:85], axis=0)
numpy_array_rem
```

Out[8]:
```
array([[  18. ,    8. ,  307. ,  130. , 3504. ,   12. ],
       [  15. ,    8. ,  350. ,  165. , 3693. ,   11.5],
       [  18. ,    8. ,  318. ,  150. , 3436. ,   11. ],
       ...,
       [  32. ,    4. ,  135. ,   84. , 2295. ,   11.6],
       [  28. ,    4. ,  120. ,   79. , 2625. ,   18.6],
       [  31. ,    4. ,  119. ,   82. , 2720. ,   19.4]])
```

In [9]:
```
max_values_columnwise_rem = np.max(numpy_array_rem, axis=0)
min_values_columnwise_rem = np.min(numpy_array_rem, axis=0)
```

In [10]:
```
range=pd.DataFrame({'Minimum Value':min_values_columnwise_rem,'Maximum value':max_values_columnwise_rem,'Range':max_v
range
```

Out[10]:

	Minimum Value	Maximum value	Range
mpg	11.0	46.6	35.6
cylinders	3.0	8.0	5.0
displacement	68.0	455.0	387.0
horsepower	46.0	230.0	184.0
weight	1649.0	4997.0	3348.0
acceleration	8.5	24.8	16.3

In [11]:

```
mean_rem=np.mean(numpy_array_rem, axis=0)
std_dev_rem=np.std(numpy_array_rem, axis=0)
```

In [12]:

```
Qd=pd.DataFrame({'Mean':mean_rem, 'Standard Deviation':std_dev_rem}, index=['mpg','cylinders','displacement','horsepo
Qd
```

Out[12]:

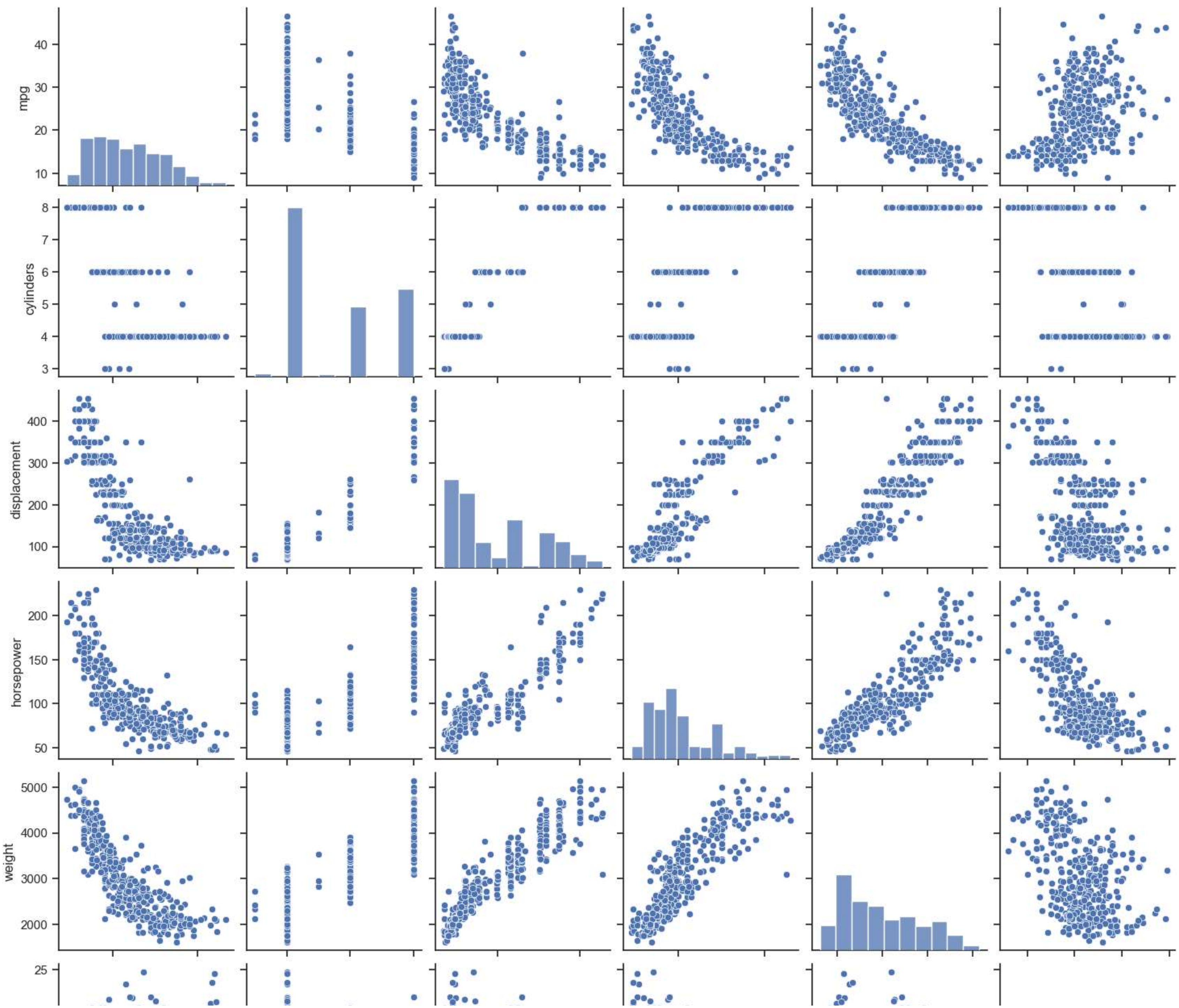
	Mean	Standard Deviation
mpg	24.404430	7.854825
cylinders	5.373418	1.651559
displacement	187.240506	99.520523
horsepower	100.721519	35.652307
weight	2935.971519	810.015488
acceleration	15.726899	2.689455

Question (e)

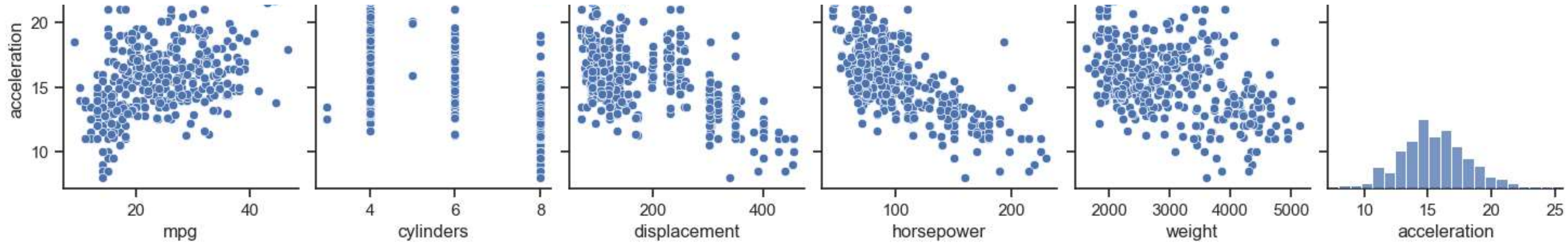
In [13]:

```
import seaborn as sns
import matplotlib.pyplot as plt
#Plotting pairplot to analyze relation between two quantitative predictors
quantitative_predictors = ['mpg', 'cylinders', 'displacement', 'horsepower', 'weight', 'acceleration']
```

```
sns.set(style='ticks')
sns.pairplot(auto_data_cleaned[quantitative_predictors])
plt.show()
```



mpg
cylinders
displacement
horsepower
weight

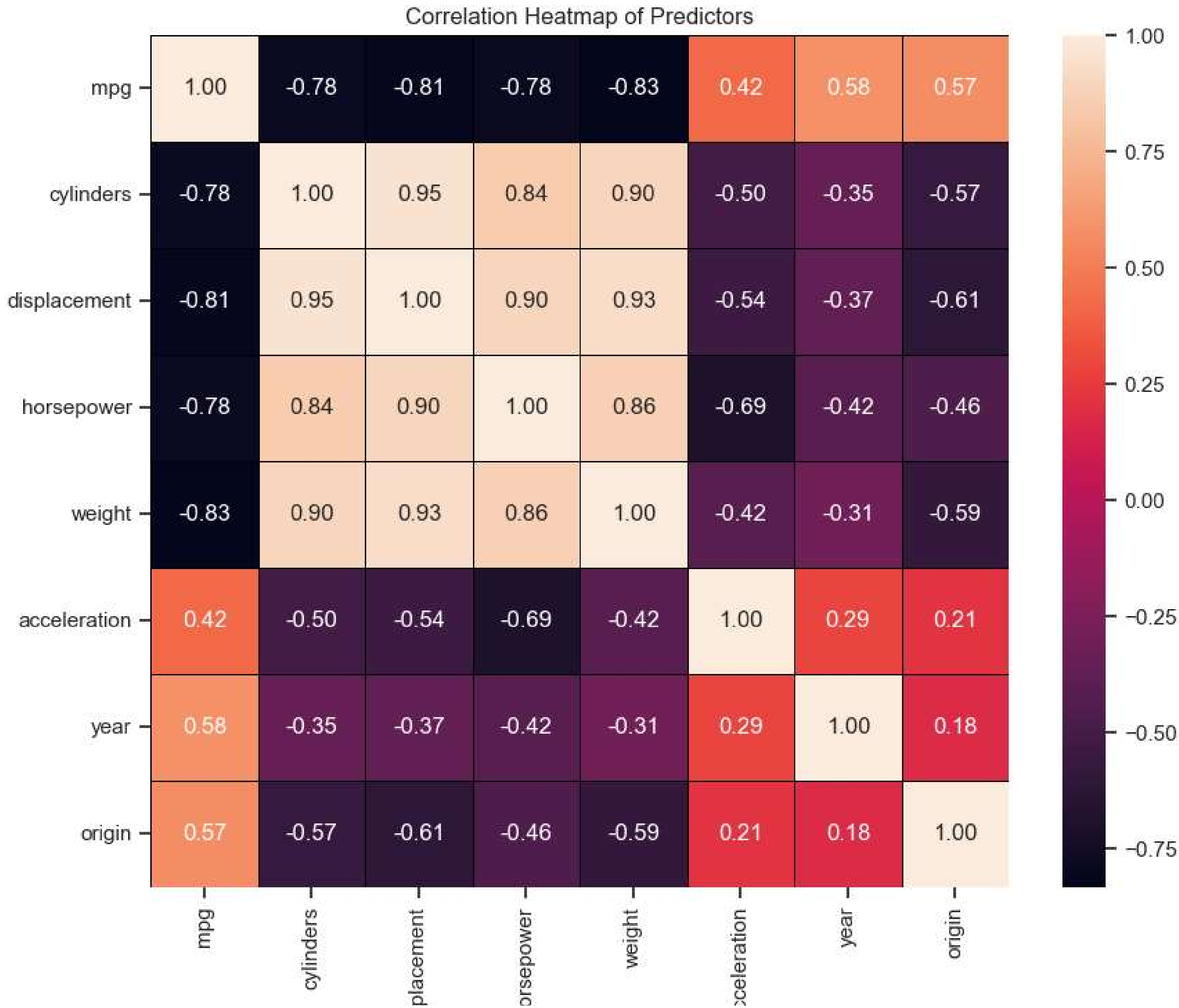


In [14]:

```
predictors=auto_data_cleaned[['mpg', 'cylinders', 'displacement', 'horsepower', 'weight', 'acceleration','year','orig

correlation_matrix =(predictors.corr())

plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True,  fmt='.2f', linecolor='black', linewidths=.5)
plt.title('Correlation Heatmap of Predictors')
plt.show()
```


Correlation Heatmap of Predictors
mpg
cylinders
displacement
horsepower
weight
acceleration
year
origin
1.00 -0.78 -0.81 -0.78 -0.83 0.42 0.58 0.57
-0.78 1.00 0.95 0.84 0.90 -0.50 -0.35 -0.57
-0.81 0.95 1.00 0.90 0.93 -0.54 -0.37 -0.61
-0.78 0.84 0.90 1.00 0.86 -0.69 -0.42 -0.46
-0.83 0.90 0.93 0.86 1.00 -0.42 -0.31 -0.59
0.42 -0.50 -0.54 -0.69 -0.42 1.00 0.29 0.21
0.58 -0.35 -0.37 -0.42 -0.31 0.29 1.00 0.18
0.57 -0.57 -0.61 -0.46 -0.59 0.21 0.18 1.00
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

Positive Correlations

Some pairs like (cylinders, displacement), (cylinders,horsepower), (cylinders,weight), (displacement, weight) etc. shows positive correlation. This means that as one variable increases, the other variable also increases and vice versa.

Negative Correlations:

Some pairs like (mpg, cylinders), (mpg, displacement), (mpg, horsepower),(mpg, weight) etc. shows strong negative correlations. This mean that as one variable increases, the other tends to decrease and vice versa.

Question (f)

According to the correlation heatmap we can see that cylinders, displacement, horsepower and weight have relatively high correlation value, so these values can be used to predict mpg.

In [15]:
```
from sklearn.linear_model import LinearRegression
```

In [16]:
```
import statsmodels.api as sm
X = sm.add_constant(auto_data_cleaned[[ 'cylinders', 'displacement', 'horsepower', 'weight', 'acceleration','year','c
X['horsepower']=(1/X['horsepower'])
#X['displacement']=(1/X['displacement'])
X['weight']=(1/X['weight'])
y = auto_data_cleaned['mpg']
model = sm.OLS(y, X).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    mpg   R-squared:                       0.861
Model:                            OLS   Adj. R-squared:                  0.858
Method:                 Least Squares   F-statistic:                     339.4
Date:                Sun, 21 Jan 2024   Prob (F-statistic):         3.86e-160
Time:                        01:56:16   Log-Likelihood:                -974.65
No. Observations:                 392   AIC:                             1965.
Df Residuals:                     384   BIC:                             1997.
Df Model:                           7
Covariance Type:            nonrobust
================================================================================
                   coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
const          -51.3729      4.810    -10.681      0.000     -60.829     -41.917
cylinders       -0.4980      0.283     -1.761      0.079      -1.054       0.058
displacement     0.0076      0.006      1.306      0.192      -0.004       0.019
horsepower     665.2107    130.901      5.082      0.000     407.839     922.583
weight        3.664e+04   5136.204      7.134      0.000    2.65e+04    4.67e+04
acceleration    -0.2306      0.098     -2.348      0.019      -0.424      -0.038
year             0.7626      0.045     16.884      0.000       0.674       0.851
origin           0.7889      0.244      3.238      0.001       0.310       1.268
==============================================================================
Omnibus:                       37.111   Durbin-Watson:                   1.543
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               75.532
Skew:                           0.532   Prob(JB):                     3.97e-17
Kurtosis:                       4.868   Cond. No.                     8.00e+06
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large,  8e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
```

The OLS(Ordinary Least Squares) regression model predicts MPG based on seven variables. The model has an R-squared of 0.861 i.e. 86.1%, indicating a strong fit. Notable predictors include horsepower, weight, acceleration, year