In [61]:
```
!jupyter nbconvert --to html /content/CS401Lab3b.ipynb
```

```
[NbConvertApp] Converting notebook /content/CS401Lab3a.ipynb to html
[NbConvertApp] Writing 1113260 bytes to /content/CS401Lab3a.html
```

In [56]:
```
from google.colab import drive
drive.mount('/content/drive')

#https://www.kaggle.com/datasets/ruiromanini/mtcars/ (where I got the dataset)
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call driv
e.mount("/content/drive", force_remount=True).
```

1. Data Exploration (10 points):

(a) Load the "mtcars" dataset and describe its structure, including the number of observations and variables.

(b) Explore the dataset by calculating summary statistics and visualizing the data. Create scatter plots to examine the relationships between the independent variables and the target variable (mpg).

In [55]:
```
#a) Load the "mtcars" dataset and describe its structure, including the number of o
import pandas as pd
from sklearn import datasets

file_path = '/content/drive/MyDrive/mtcars.csv'
df = pd.read_csv(file_path)

print(df)
print(df.shape[0])
print(df.shape[1])
```

```
                  model   mpg  cyl   disp   hp  drat     wt   qsec  vs  am  \
0             Mazda RX4  21.0    6  160.0  110  3.90  2.620  16.46   0   1
1         Mazda RX4 Wag  21.0    6  160.0  110  3.90  2.875  17.02   0   1
2            Datsun 710  22.8    4  108.0   93  3.85  2.320  18.61   1   1
3        Hornet 4 Drive  21.4    6  258.0  110  3.08  3.215  19.44   1   0
4     Hornet Sportabout  18.7    8  360.0  175  3.15  3.440  17.02   0   0
5               Valiant  18.1    6  225.0  105  2.76  3.460  20.22   1   0
6            Duster 360  14.3    8  360.0  245  3.21  3.570  15.84   0   0
7             Merc 240D  24.4    4  146.7   62  3.69  3.190  20.00   1   0
8              Merc 230  22.8    4  140.8   95  3.92  3.150  22.90   1   0
9              Merc 280  19.2    6  167.6  123  3.92  3.440  18.30   1   0
10            Merc 280C  17.8    6  167.6  123  3.92  3.440  18.90   1   0
11           Merc 450SE  16.4    8  275.8  180  3.07  4.070  17.40   0   0
12           Merc 450SL  17.3    8  275.8  180  3.07  3.730  17.60   0   0
13          Merc 450SLC  15.2    8  275.8  180  3.07  3.780  18.00   0   0
14   Cadillac Fleetwood  10.4    8  472.0  205  2.93  5.250  17.98   0   0
15  Lincoln Continental  10.4    8  460.0  215  3.00  5.424  17.82   0   0
16    Chrysler Imperial  14.7    8  440.0  230  3.23  5.345  17.42   0   0
17             Fiat 128  32.4    4   78.7   66  4.08  2.200  19.47   1   1
18          Honda Civic  30.4    4   75.7   52  4.93  1.615  18.52   1   1
19       Toyota Corolla  33.9    4   71.1   65  4.22  1.835  19.90   1   1
20        Toyota Corona  21.5    4  120.1   97  3.70  2.465  20.01   1   0
21     Dodge Challenger  15.5    8  318.0  150  2.76  3.520  16.87   0   0
22          AMC Javelin  15.2    8  304.0  150  3.15  3.435  17.30   0   0
23           Camaro Z28  13.3    8  350.0  245  3.73  3.840  15.41   0   0
24     Pontiac Firebird  19.2    8  400.0  175  3.08  3.845  17.05   0   0
25            Fiat X1-9  27.3    4   79.0   66  4.08  1.935  18.90   1   1
26        Porsche 914-2  26.0    4  120.3   91  4.43  2.140  16.70   0   1
27         Lotus Europa  30.4    4   95.1  113  3.77  1.513  16.90   1   1
28       Ford Pantera L  15.8    8  351.0  264  4.22  3.170  14.50   0   1
29         Ferrari Dino  19.7    6  145.0  175  3.62  2.770  15.50   0   1
30        Maserati Bora  15.0    8  301.0  335  3.54  3.570  14.60   0   1
31           Volvo 142E  21.4    4  121.0  109  4.11  2.780  18.60   1   1

    gear  carb
0      4     4
1      4     4
2      4     1
3      3     1
4      3     2
5      3     1
6      3     4
7      4     2
8      4     2
9      4     4
10     4     4
11     3     3
12     3     3
13     3     3
14     3     4
15     3     4
16     3     4
17     4     1
18     4     2
19     4     1
20     3     1
21     3     2
22     3     2
23     3     4
24     3     2
25     4     1
26     5     2
27     5     2
28     5     4
```

```
29      5      6
30      5      8
31      4      2
32
12
```

a) Load the "mtcars" dataset and describe its structure, including the number of observations and variables.

There are 32 rows in total with 11 columns made up of: model, mpg, cyl, disp, hp, drat, wt, qsec, vs, am, gear, carb.

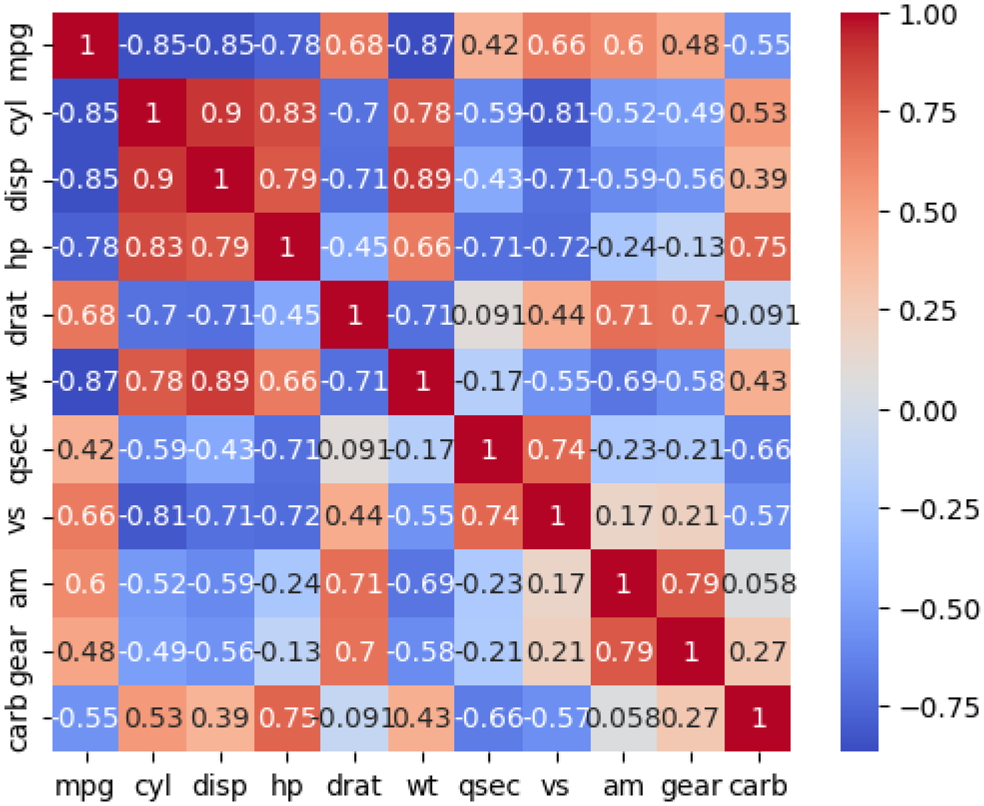
This is the model of the car, miles per gallon, number of cylinders in the engine, displacement (Engine's volume), horsepower, drat (affects car's acceleration and top speed), weight of the car, qsec is the amount of time it takes to go from 0mph to 60mph, vs is the engine type, am is the transmission type, gear is gears & carb is the number of carburetors that the engine has.

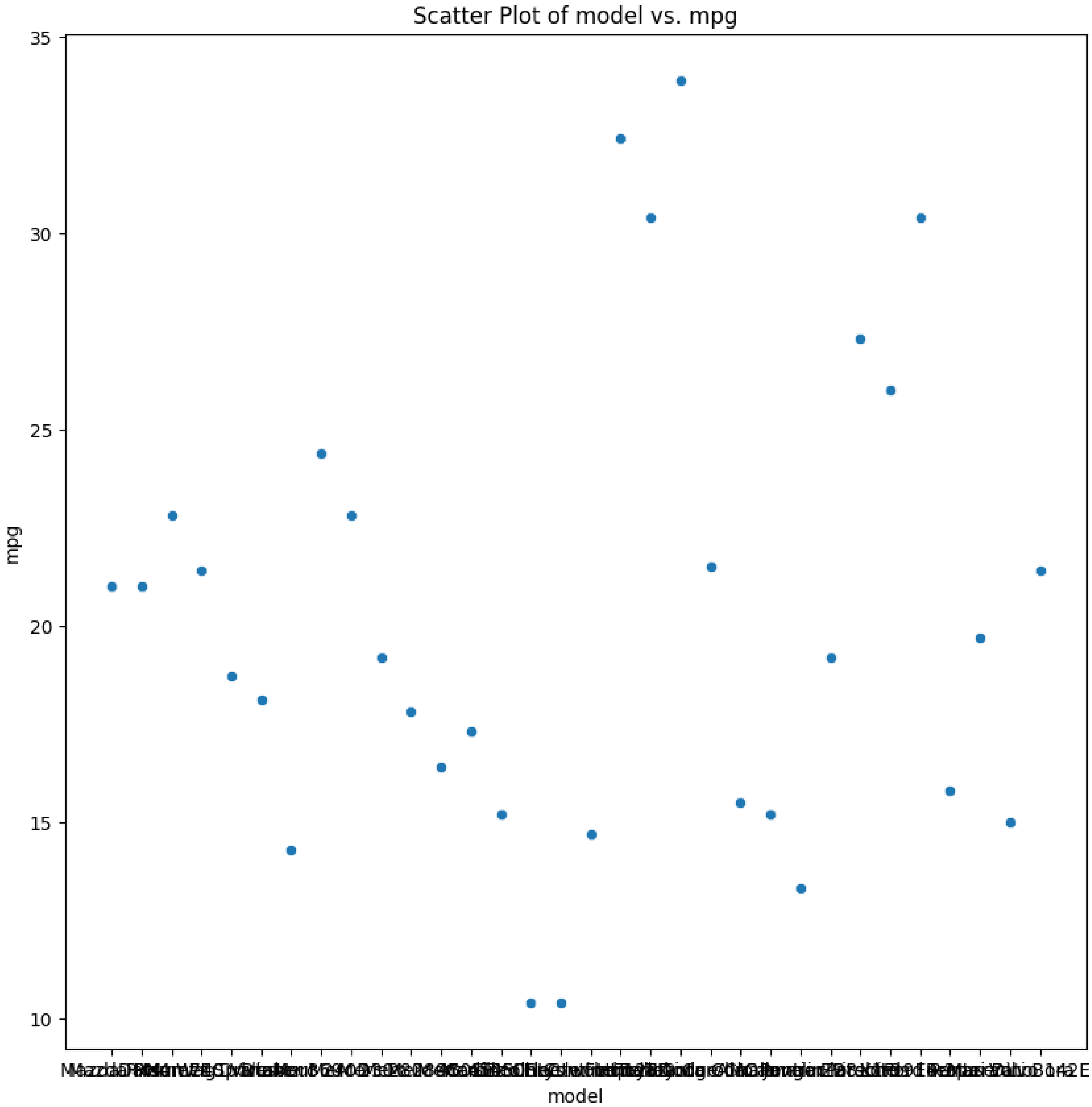
In [57]:

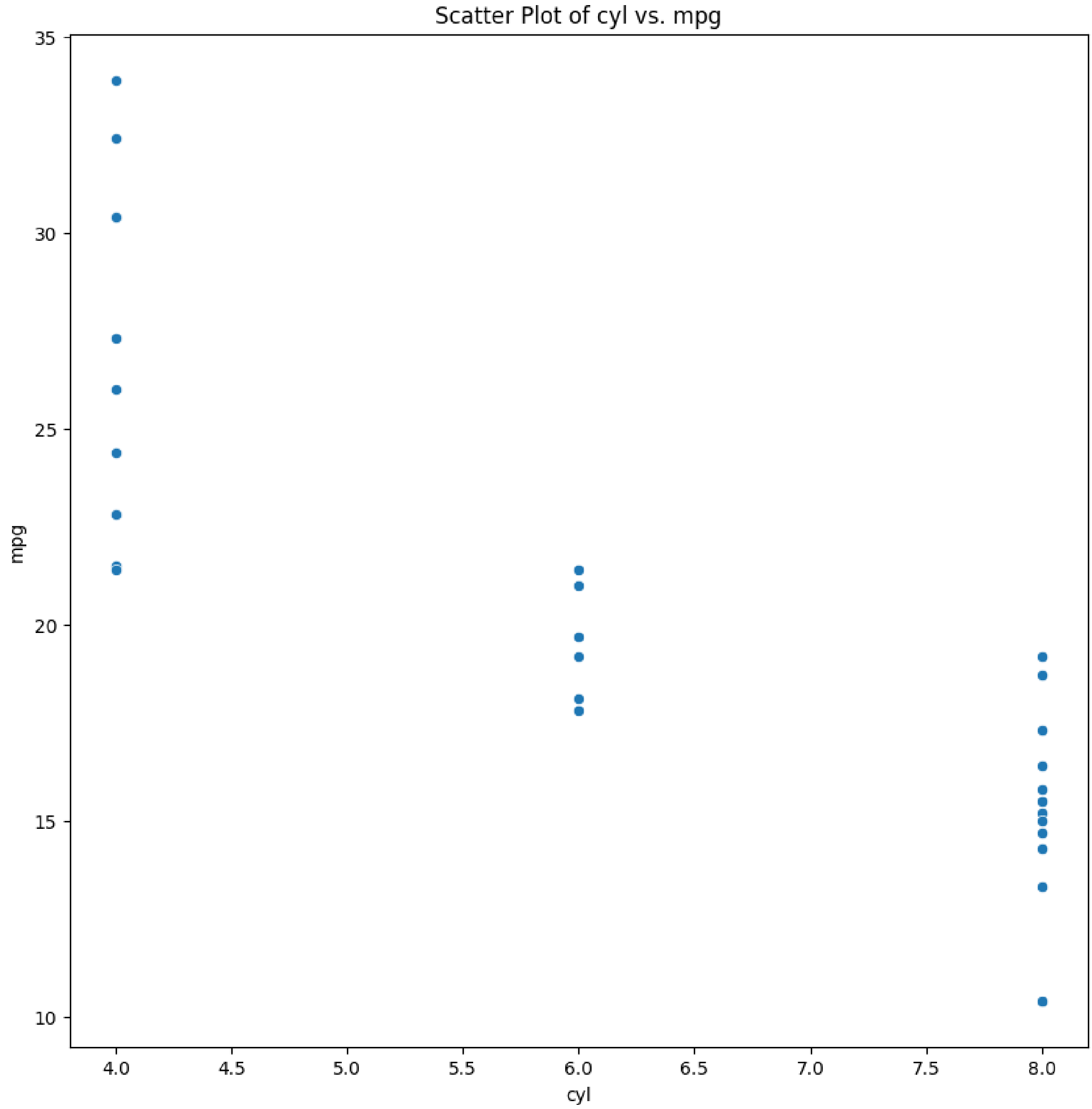
```python
#b) Explore the dataset by calculating summary statistics and visualizing the data.
#Create scatter plots to examine the relationships between the independent variable
import seaborn as sns
import matplotlib.pyplot as plt

summary_stats = df.describe()
print(summary_stats)

sns.heatmap(df.corr(), annot=True, cmap='coolwarm', square=True)
plt.show()

for column in df.columns:
    if column != 'mpg':  # Exclude 'mpg' from independent variables
        plt.figure(figsize=(10, 10))
        sns.scatterplot(x=column, y='mpg', data=df)
        plt.title(f'Scatter Plot of {column} vs. mpg')
        plt.xlabel(column)
        plt.ylabel('mpg')
        plt.show()
```

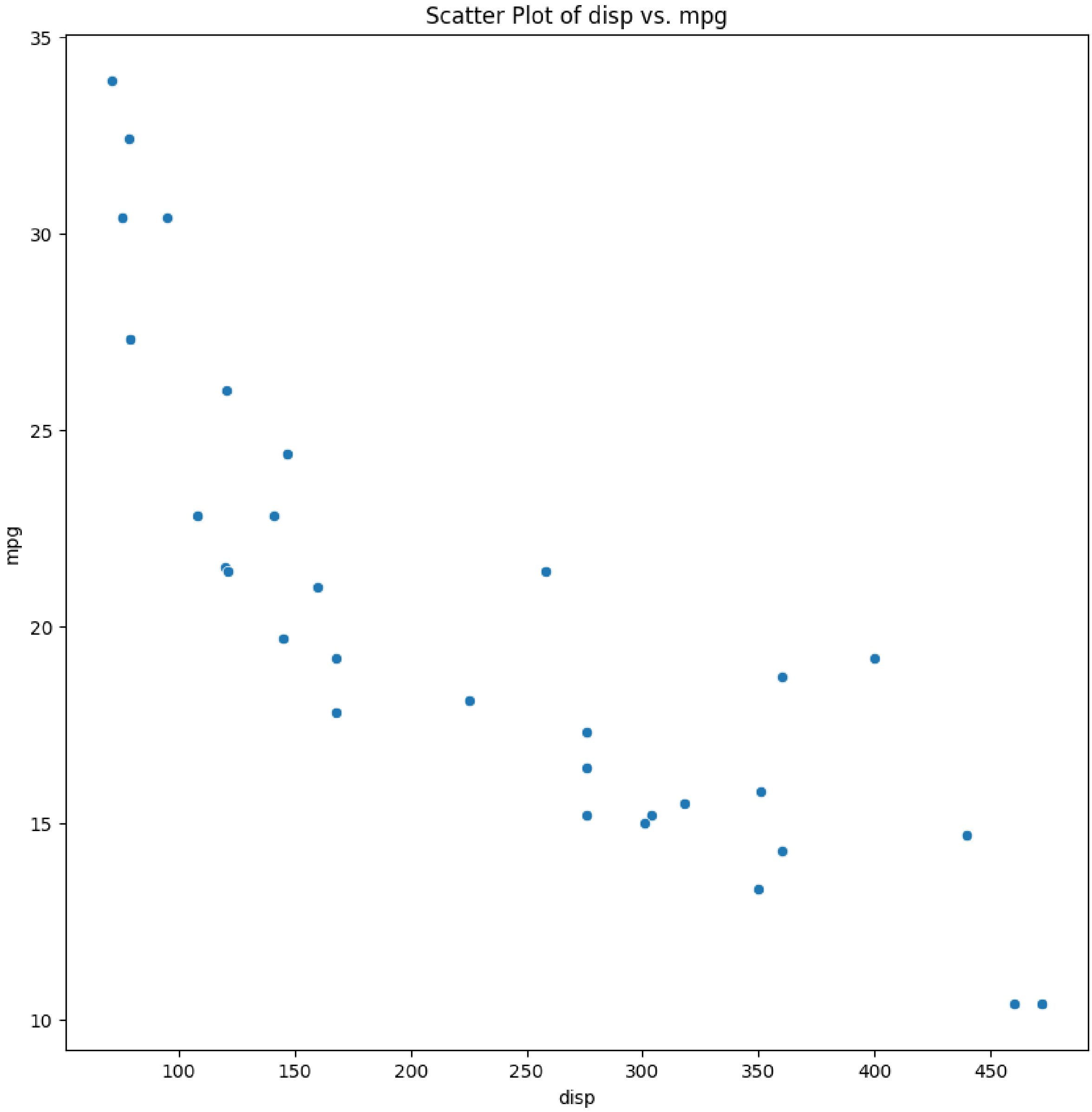
```
             mpg        cyl        disp          hp       drat         wt  \
count  32.000000  32.000000   32.000000   32.000000  32.000000  32.000000
mean   20.090625   6.187500  230.721875  146.687500   3.596563   3.217250
std     6.026948   1.785922  123.938694   68.562868   0.534679   0.978457
min    10.400000   4.000000   71.100000   52.000000   2.760000   1.513000
25%    15.425000   4.000000  120.825000   96.500000   3.080000   2.581250
50%    19.200000   6.000000  196.300000  123.000000   3.695000   3.325000
75%    22.800000   8.000000  326.000000  180.000000   3.920000   3.610000
max    33.900000   8.000000  472.000000  335.000000   4.930000   5.424000

            qsec         vs         am       gear     carb
count  32.000000  32.000000  32.000000  32.000000  32.0000
mean   17.848750   0.437500   0.406250   3.687500   2.8125
std     1.786943   0.504016   0.498991   0.737804   1.6152
min    14.500000   0.000000   0.000000   3.000000   1.0000
25%    16.892500   0.000000   0.000000   3.000000   2.0000
50%    17.710000   0.000000   0.000000   4.000000   2.0000
75%    18.900000   1.000000   1.000000   4.000000   4.0000
max    22.900000   1.000000   1.000000   5.000000   8.0000
```

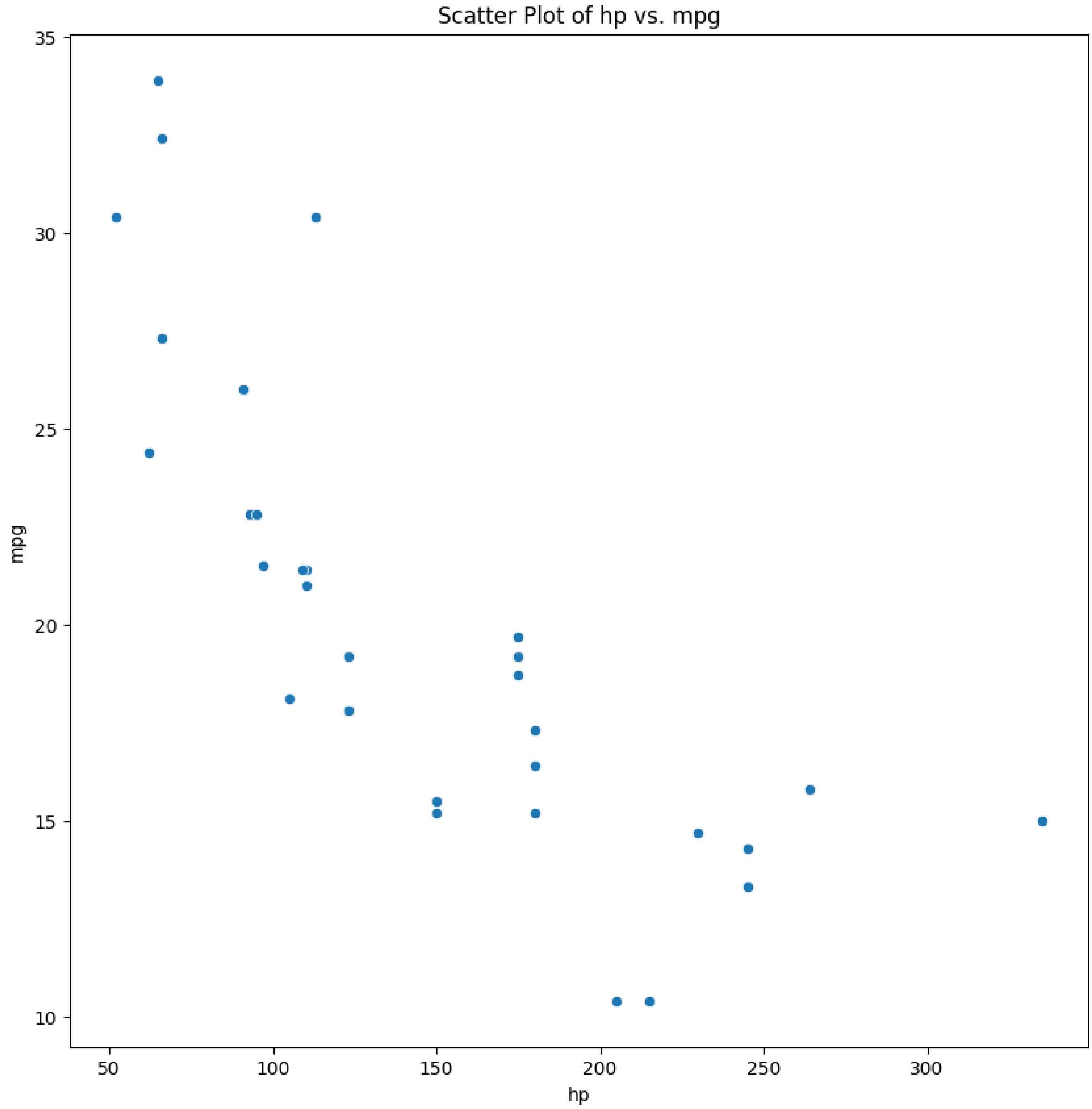
```
<ipython-input-57-53a29aa8611e>:9: FutureWarning: The default value of numeric_onl
y in DataFrame.corr is deprecated. In a future version, it will default to False.
Select only valid columns or specify the value of numeric_only to silence this war
ning.
  sns.heatmap(df.corr(), annot=True, cmap='coolwarm', square=True)
```

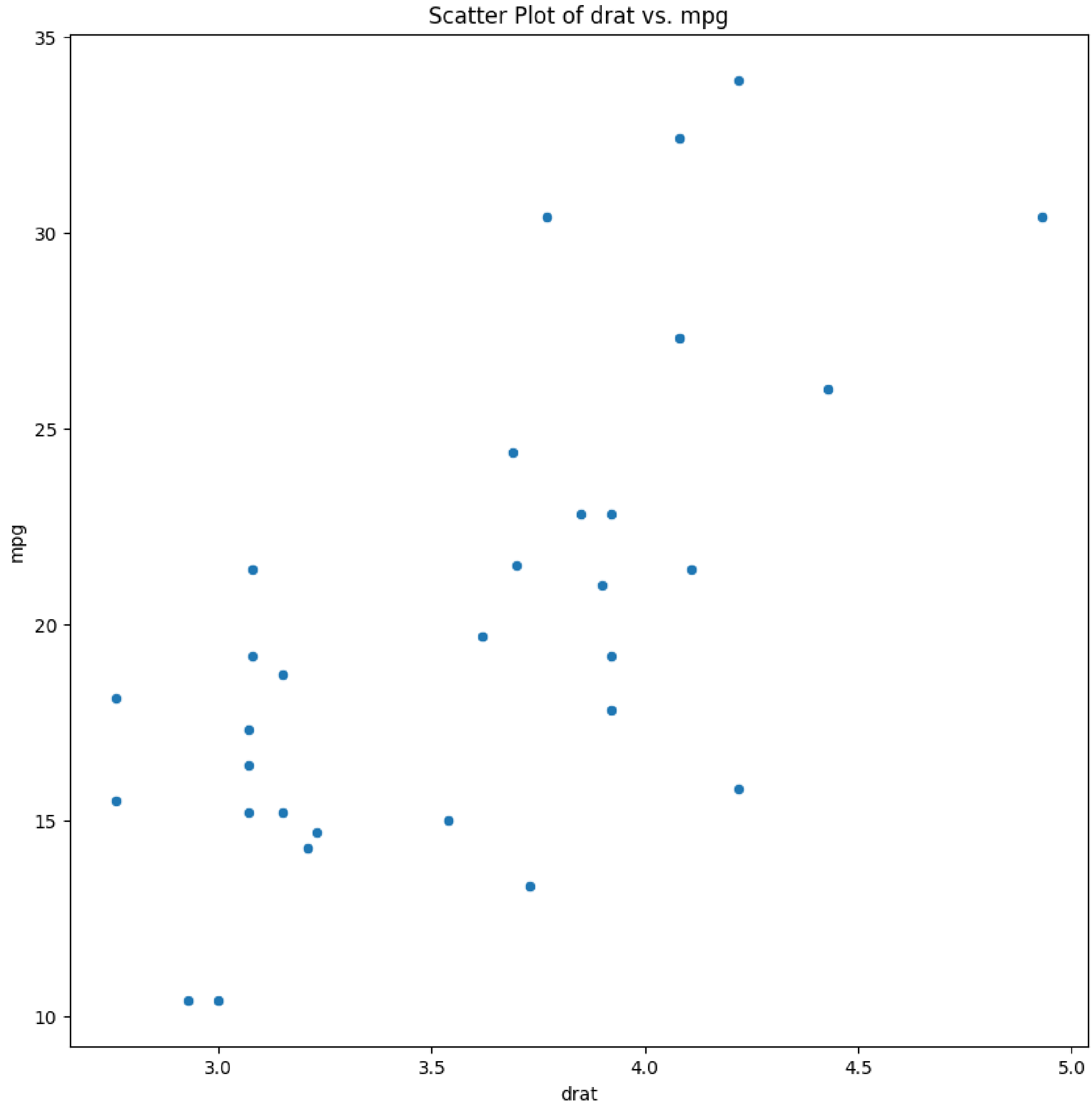


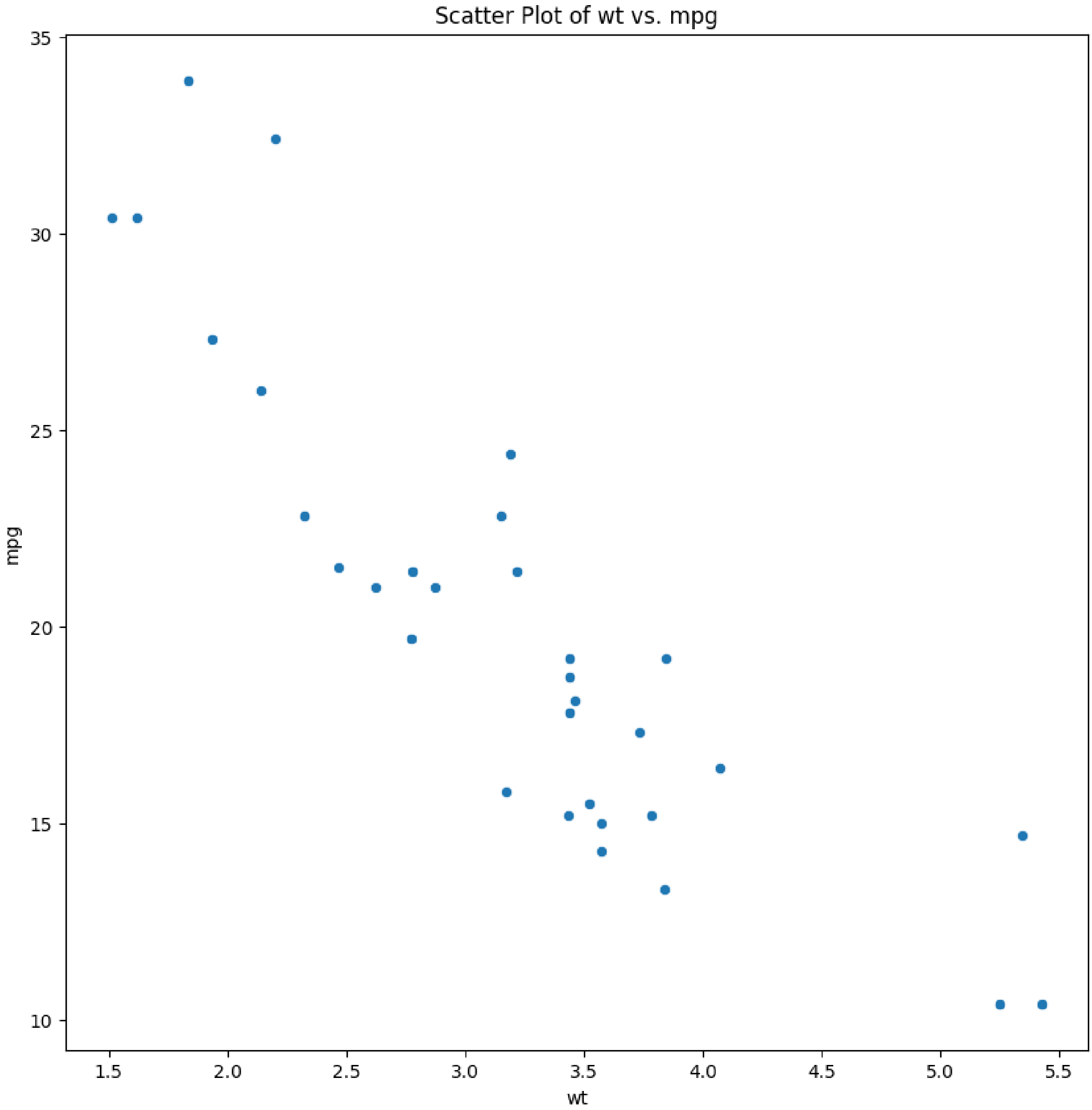

Scatter Plot of model vs. mpg
mpg
35
30
25
20
15
10
model

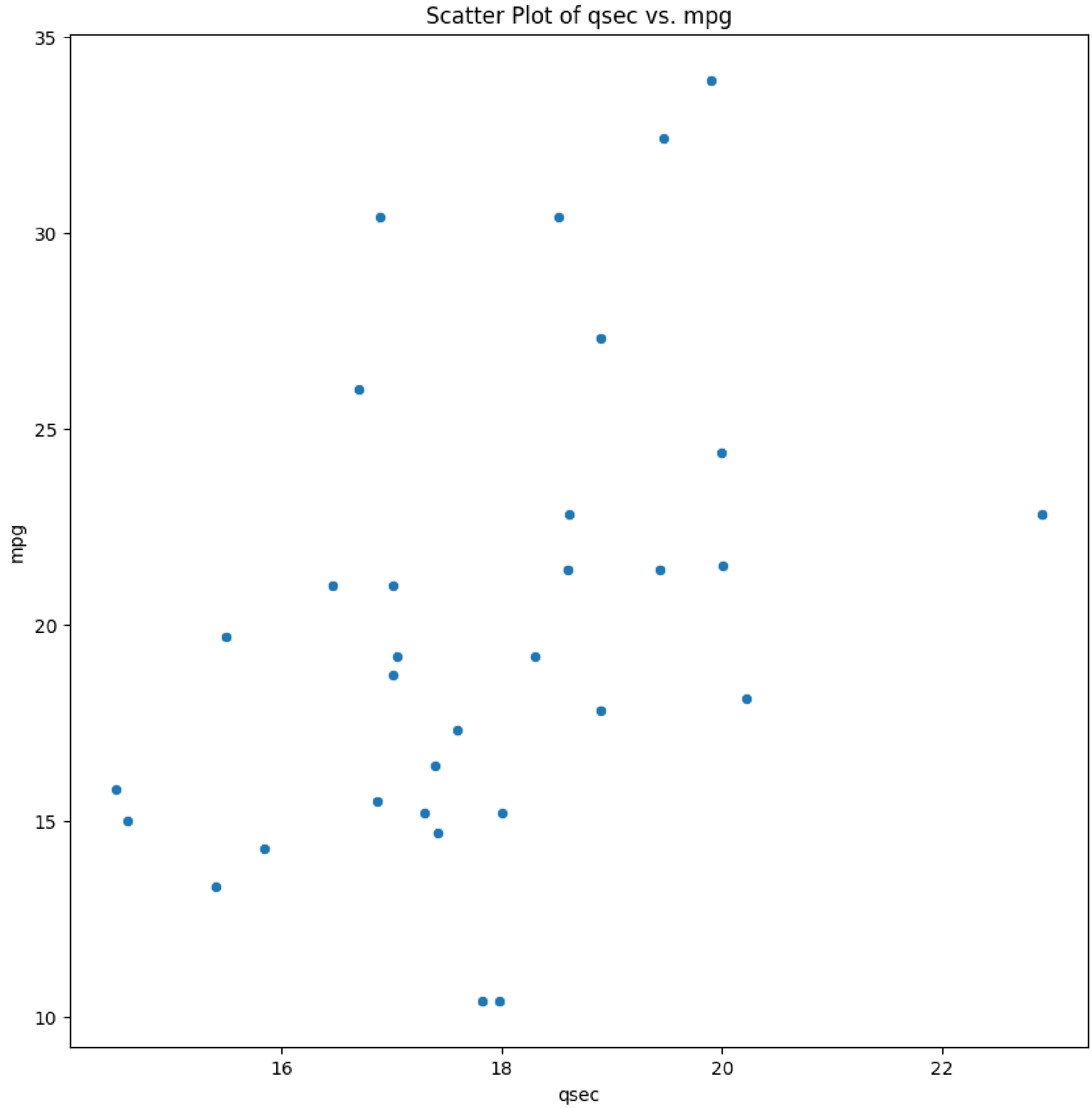

Scatter Plot of cyl vs. mpg
35
30
25
20
15
10
mpg
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
cyl

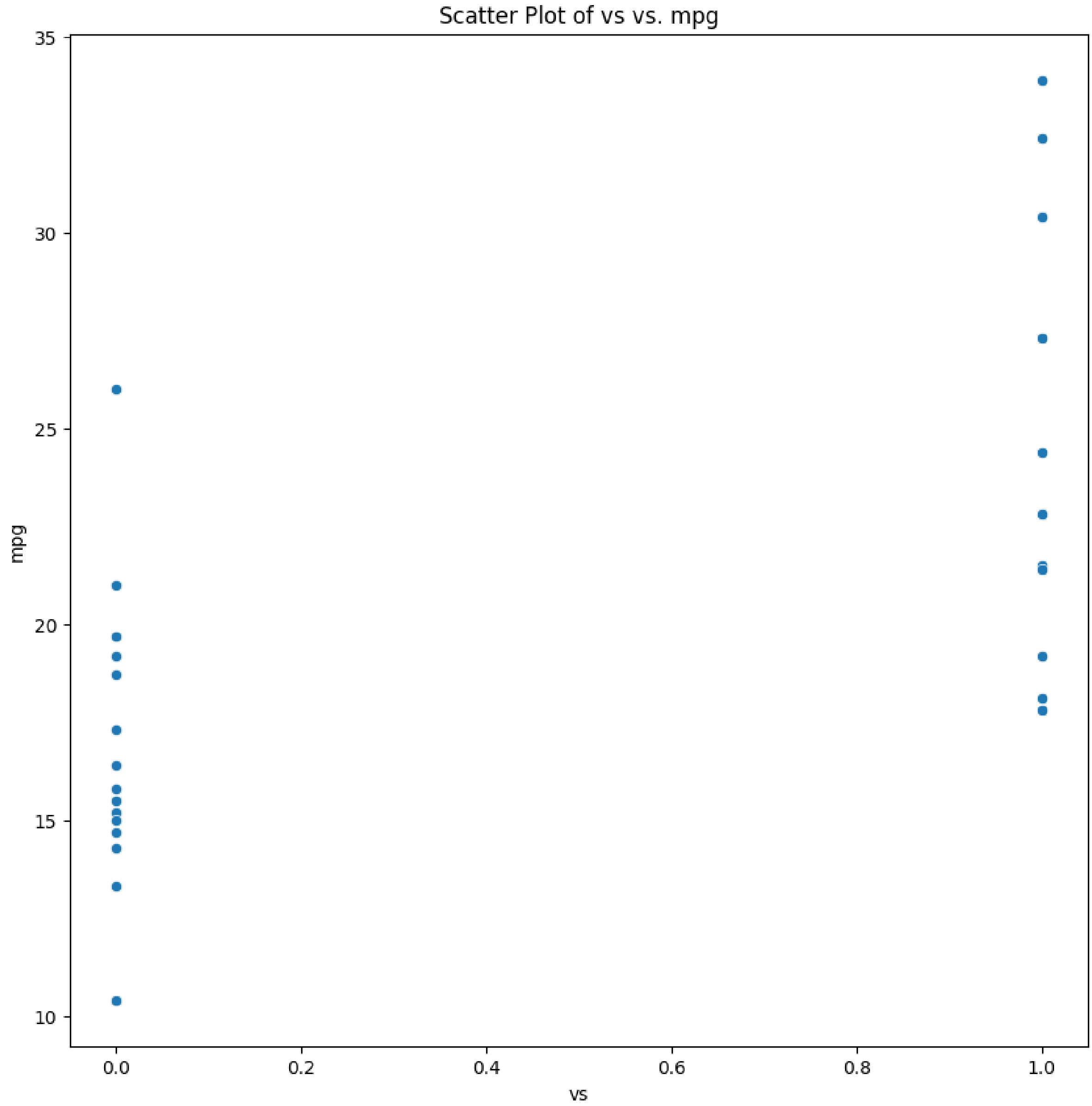

Scatter Plot of disp vs. mpg
35
30
25
20
15
10
mpg
100
150
200
250
300
350
400
450
disp

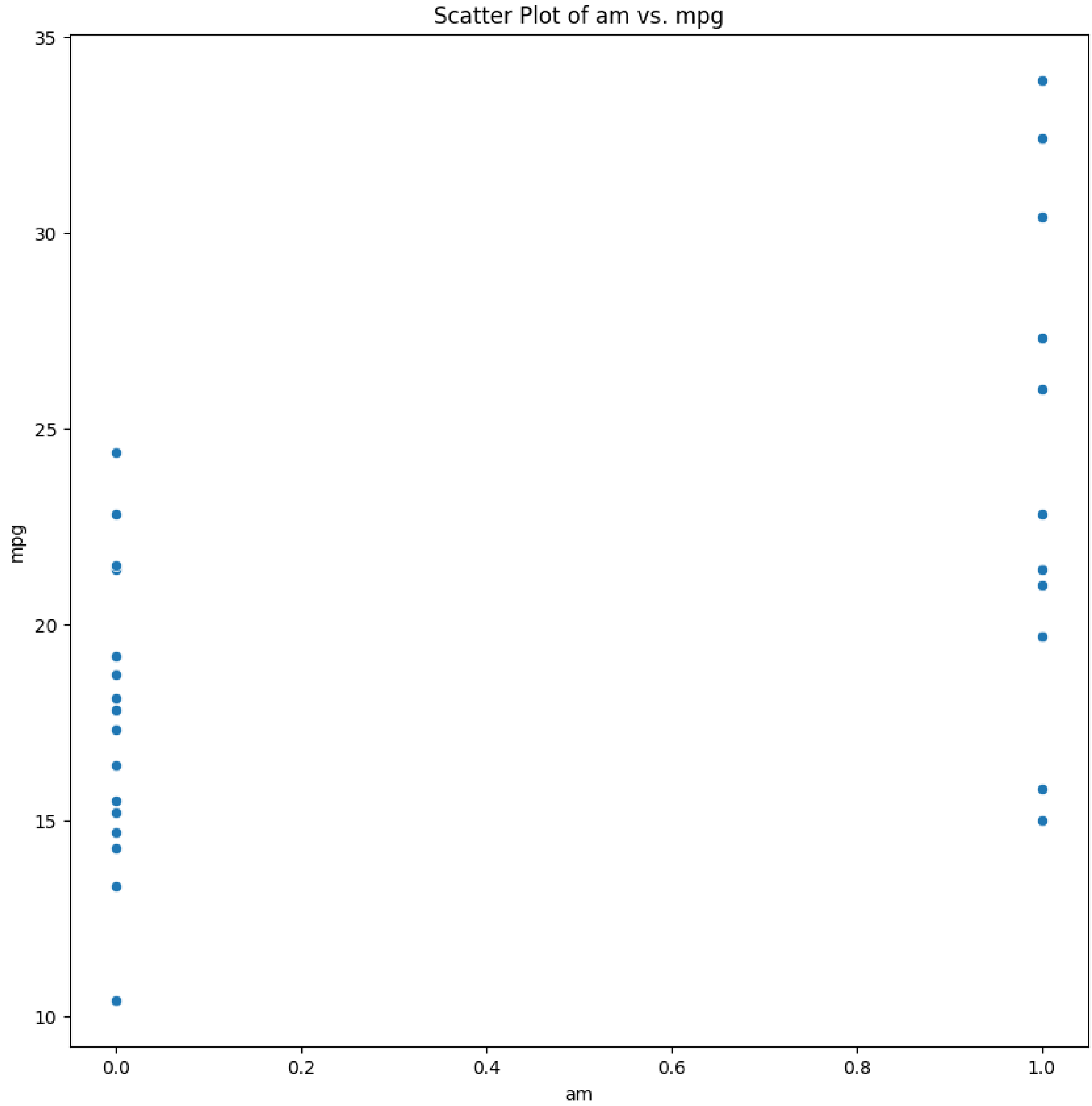

Scatter Plot of hp vs. mpg
mpg
hp
35
30
25
20
15
10
50
100
150
200
250
300

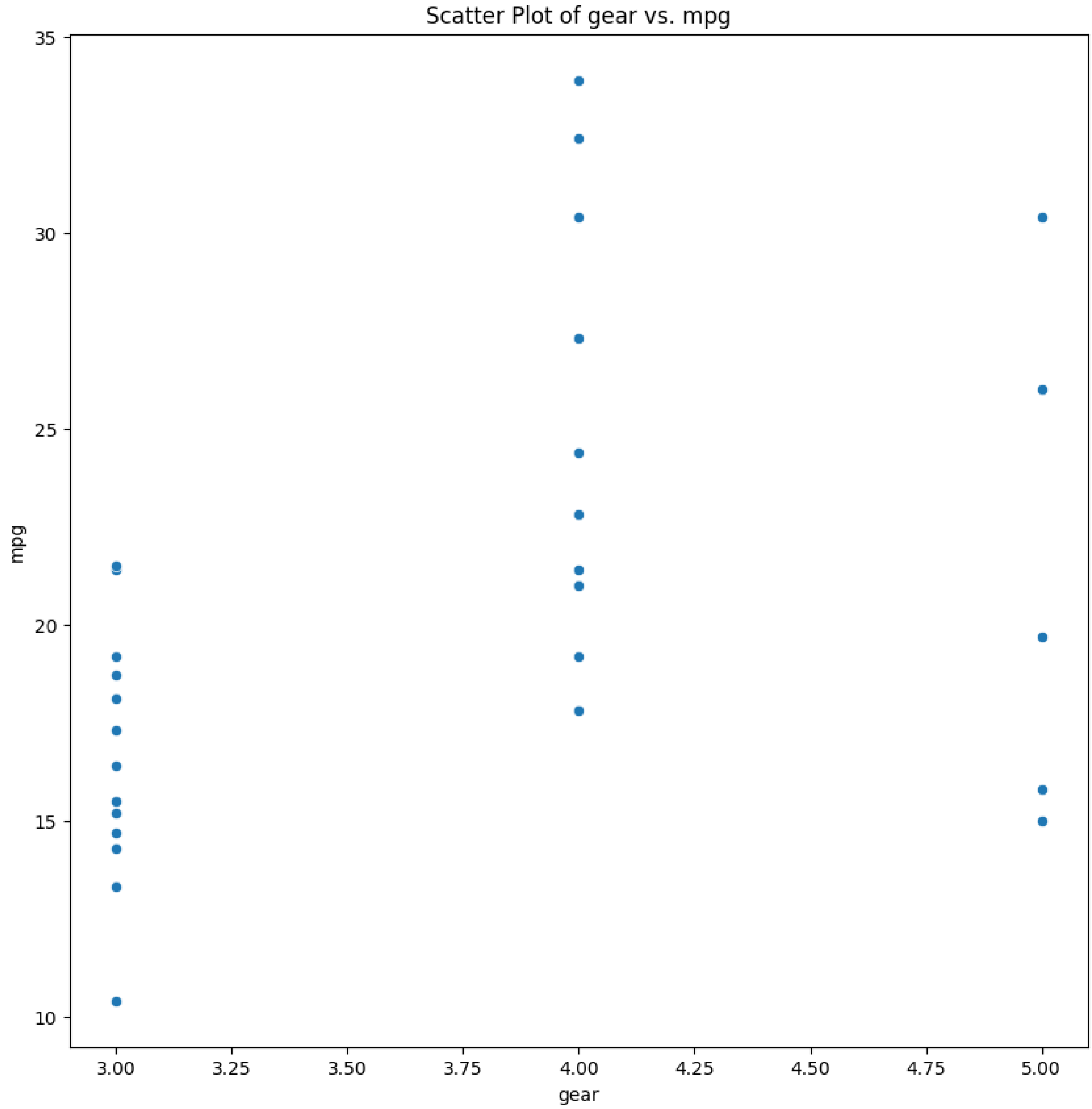

Scatter Plot of drat vs. mpg
mpg
drat
35
30
25
20
15
10
3.0
3.5
4.0
4.5
5.0

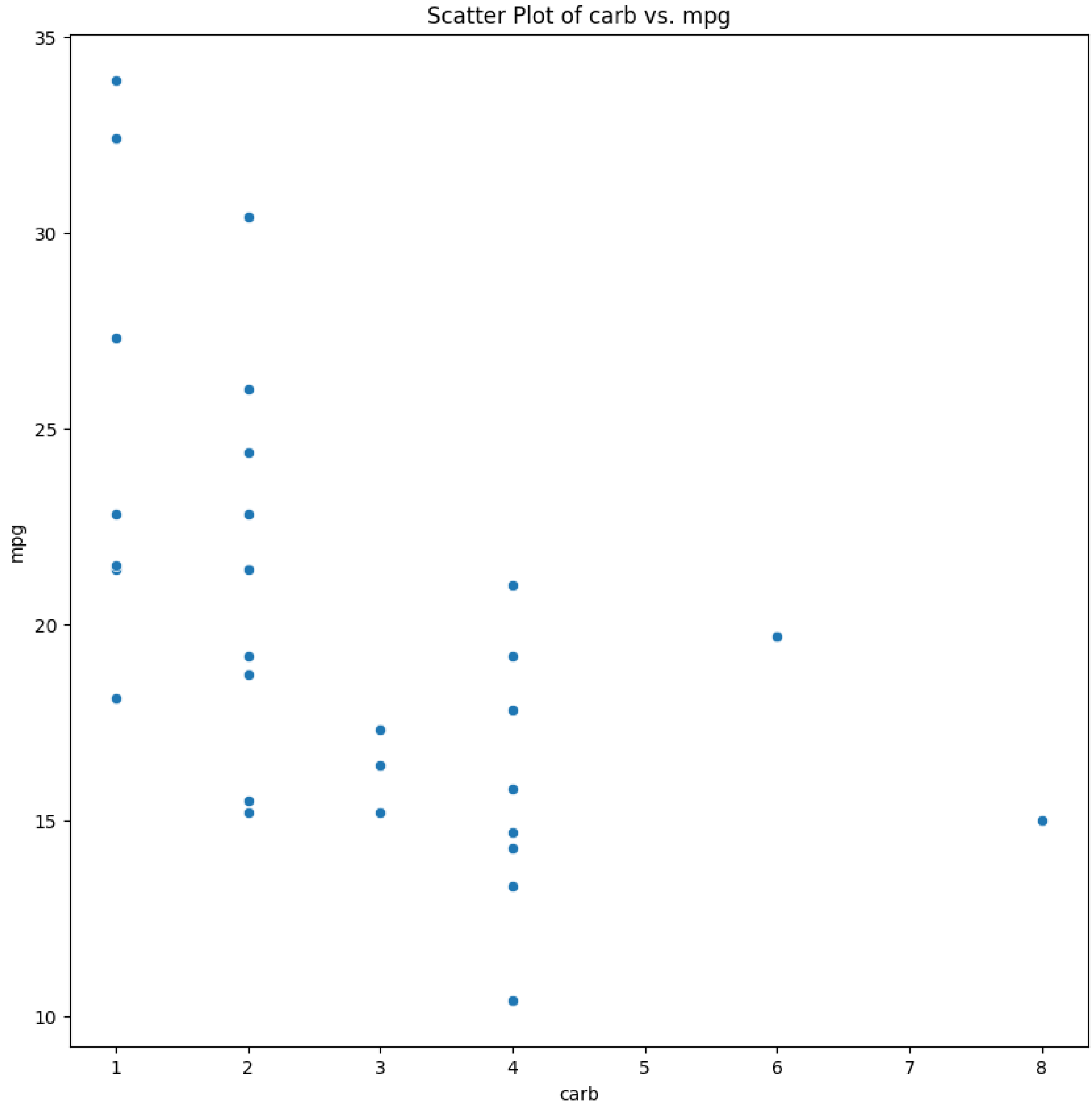

Scatter Plot of wt vs. mpg
mpg
wt
35
30
25
20
15
10
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5


Scatter Plot of qsec vs. mpg
35
30
25
20
15
10
mpg
16
18
20
22
qsec


Scatter Plot of vs vs. mpg
mpg
vs
35
30
25
20
15
10
0.0
0.2
0.4
0.6
0.8
1.0


Scatter Plot of am vs. mpg
mpg
am
35
30
25
20
15
10
0.0
0.2
0.4
0.6
0.8
1.0


Scatter Plot of gear vs. mpg
mpg
gear
35
30
25
20
15
10
3.00
3.25
3.50
3.75
4.00
4.25
4.50
4.75
5.00



b) Explore the dataset by calculating summary statistics and visualizing the data. Create scatter plots to examine the relationships between the independent variables and the target variable (mpg).

The summary statistics tell us information about each of the 12 columns. This includes, the count, mean, std, min, max & number at each quartile.

We also have a heatmap which shows which variables have the most correlation. The darker red it is, the higher the correlation. The darker the blue, the lower amount of correlation.

This is important to do as it allows us to see the relationship between the other independent variables and MPG before the scatterplots are made. Drat has the highest amount of correlation with mpg and cyl has the least amount of correlation of

Positive/Negative Linear Relationship No Linear Relationship Outliers Clusters or Groups Correlation Strength Non-Linear Relationships D

Cyl, vs, am, gear & cabs are split up into groups of whole numbers. Carb is mainly split up into 1, 2, 3 & 4 but there is an outlier in both 6 & 8. There's a strong negative linear relationship in (cyl vs. mpg), (disp vs. mpg), (hp vs. mpg), (wt vs. mpg) & (carb vs. mpg) (qsec vs. mpg) & (drat vs. mpg) are the only ones with a notable with strong positive correlation while (am vs. mpg) & (vs vs. mpg) does have some correlation.

(model vs. mpg) has no apparent linear relationship.

1. Simple Linear Regression (30 points):

a) Select one independent variable from the "mtcars" dataset that you believe may have a strong linear relationship with the target variable (mpg).

b) Implement a simple linear regression model to predict mpg using the selected independent variable.

c) Calculate the model's coefficients (slope and intercept) and evaluate its performance using appropriate regression evaluation metrics (on testing dataset).

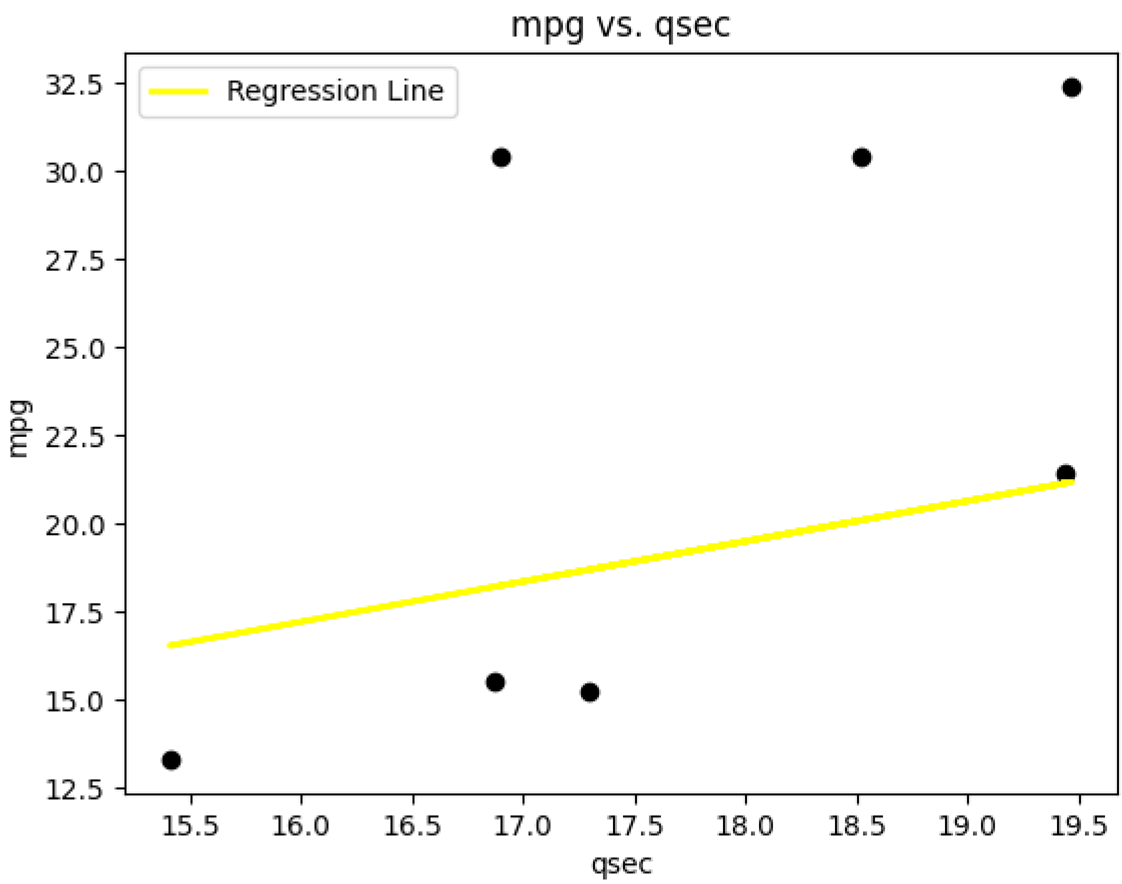
In [ ]:

```
#a) Select one independent variable from the "mtcars" dataset that you believe may
#I believe qsec vs. mpg has a strong linear relationship with mpg so I will choose
```

In [ ]:

```
#b) Implement a simple linear regression model to predict mpg using the selected ir
#c) Calculate the model's coefficients (slope and intercept) and evaluate its perfc

from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score

x = df[['qsec']]
y = df['mpg']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_sta

model = LinearRegression().fit(x_train, y_train)
Prediction = model.predict(x_test)
slope = model.coef_[0]
intercept = model.intercept_
mse = mean_squared_error(y_test, Prediction)
r2 =  r2_score(y_test, Prediction)


print(f"Slope: {slope:.2f}")
print(f"Intercept: {intercept:.2f}")
print(f"Mean Squared Sum of Errors: {mse:.2f}")
print(f"R2: {r2:.2f}")

plt.scatter(x_test, y_test, color='Black')
plt.plot(x_test, Prediction, color='Yellow', linewidth=2, label='Regression Line')
plt.xlabel('qsec')
plt.ylabel('mpg')
plt.legend()
plt.title('mpg vs. qsec')
plt.show()
```

```
Slope: 1.15
Intercept: -1.12
Mean Squared Sum of Errors: 58.58
R2: 0.00
```



b) Implement a simple linear regression model to predict mpg using the selected independent variable. c) Calculate the model's coefficients (slope and intercept) and evaluate its performance using appropriate regression evaluation metrics (on testing dataset).

This is done! We can see that the slipe is 1.15, the intercept is -1.12 and the MSE is 58.58.

This is similar to what was asked of us last week to it made doing this question a lot easier.

1. Multiple Linear Regression (40 points):

(a)Implement a multiple linear regression model using a combination of independent variables from the "mtcars" dataset.

(b) Train the model to predict mpg using multiple features.

(c) Evaluate the model's performance using appropriate regression evaluation metrics (on testing dataset).

In [43]:
```
#(a)Implement a multiple linear regression model using a combination of independent
#(b) Train the model to predict mpg using multiple features.
#On a side note, it's when we are doing stuff like this that I just wish that we we
#a repeat of what was done previously.

X = df[['qsec', 'wt', 'hp']]
Y = df['mpg']
model = LinearRegression()
model.fit(X,Y)

print(f"Slope: {model.coef_}")
print(f"Intercept: {model.intercept_}")
```

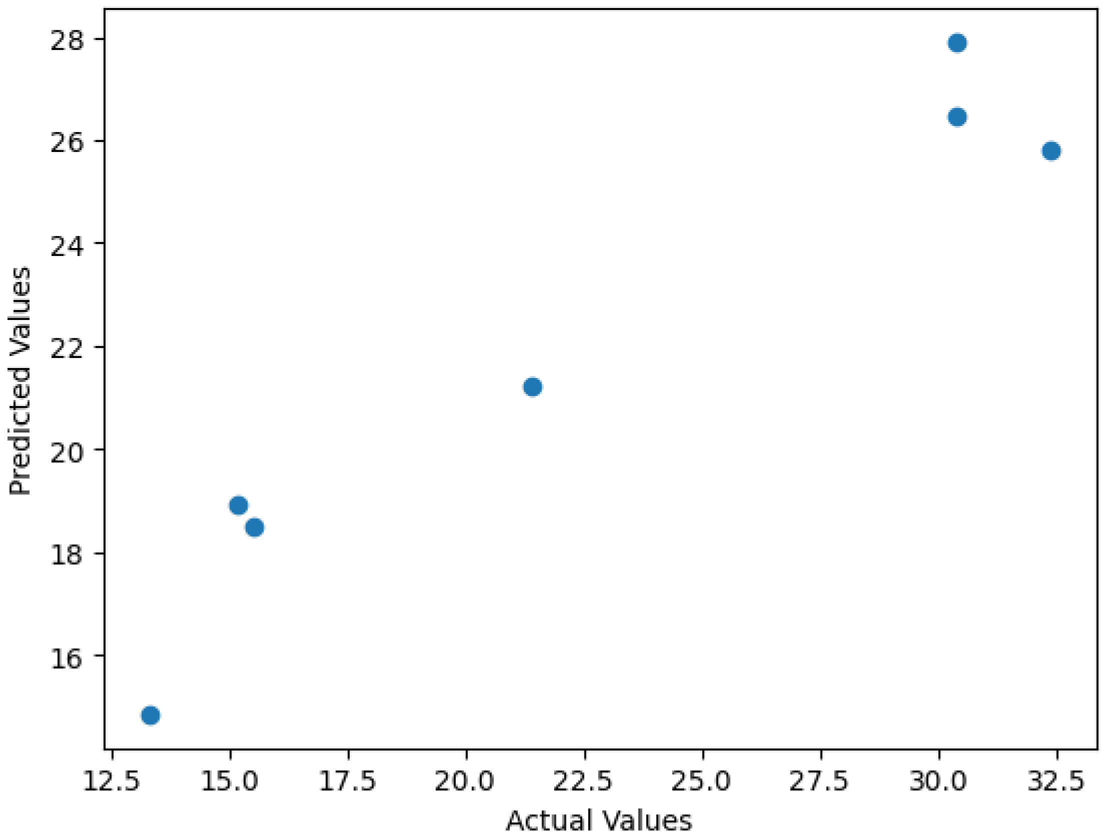
```
Slope: [ 0.51083369 -4.3587972  -0.01782227]
Intercept: 27.610526858205063
```

Something of note is X = df[['qsec', 'wt', 'hp']] is made up of the dataframe and array. The two [] need to be used here or it simply doesn't work.

Apart from that, it's simple to do the Multiple Linear Regression model.

In [52]:
```
#(c) Evaluate the model's performance using appropriate regression evaluation metri
from math import sqrt

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta
model = LinearRegression()
model.fit(X_train, y_train)
PredictionB = model.predict(X_test)
RMSE = sqrt(mean_squared_error(y_test, PredictionB))

print(f"Mean Squared Error (MSE): {mean_squared_error(y_test, PredictionB):.2f}")
print(f"R-squared (R2): {r2_score(y_test, PredictionB):.2f}")
print(f"Root Mean Squared Error (RMSE): {RMSE:.2f}")

plt.scatter(y_test, PredictionB)
plt.xlabel("Actual Values")
plt.ylabel("Predicted Values")
#plt.legend()
plt.show()
```

```
Mean Squared Error (MSE): 12.89
R-squared (R2): 0.78
Root Mean Squared Error (RMSE): 3.59
```



Mean Squared Error (MSE): 12.89 R-squared (R2): 0.78 Root Mean Squared Error (RMSE): 3.59

We get the expected result as seen in the scatterplot. The correlation is positive. There is a small cluster at the top right of the model that should be made note of but our actual results are not unexpected.

1. Discussion and Conclusion 20 points):

(a) Compare the performance and interpretability of the simple linear regression model with the multiple linear regression model. Discuss the trade-offs between simplicity and complexity.

The simple linear regression model shows the effect 2 variables have on each other. It shows us the relationship between the 2 variables. it's easy to produce and easy to interpret.

Compared to Simple Linear Regression, Multiple Linear Regression shows the complexities of the linear relationships by comparing to more variables. While this can lead to a better model performance, too many variables could make the model moot.

Simply, the Simple Linear Regression model is a lot easier to understand. We are comparing 2 variables to each other. Multilinear Regresson Model has more factors to influence the model, thus makes it more complicated.

For the most accurate answer, the multi-linear regression model is better for predicting how the relationship is but if there are variables that do not matter, it can skew the model.

Choosing between the two depends on the situation. Knowing the best one to choose depending on the situation is key to a successful model.

(b) Reflect on the insights gained from the assignment and the implications for predicting fuel efficiency in car models.

Simple & Multiple Linear Regression models can be used for predicting fuel efficiency in car models in the future. However, the variables being compared must be important. By developing regression models to predict fuel efficiency, car makers/manufactuers can make a more informed decision when makeing the car. Seeing how different fuel types react affect sales of cars could be very important. E.G: Electricity vs. Gas.

This assignments shows the importance of fuel efficiency, linear regression models and why you should use both simple & multiple linear regression models.