Akhil Juneja 7015523 Aashita Balan 7012436

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Load the Auto dataset in a DataFrame using Pandas

```
data = pd.read_csv('data/auto-dataset.csv')
print(data)
```

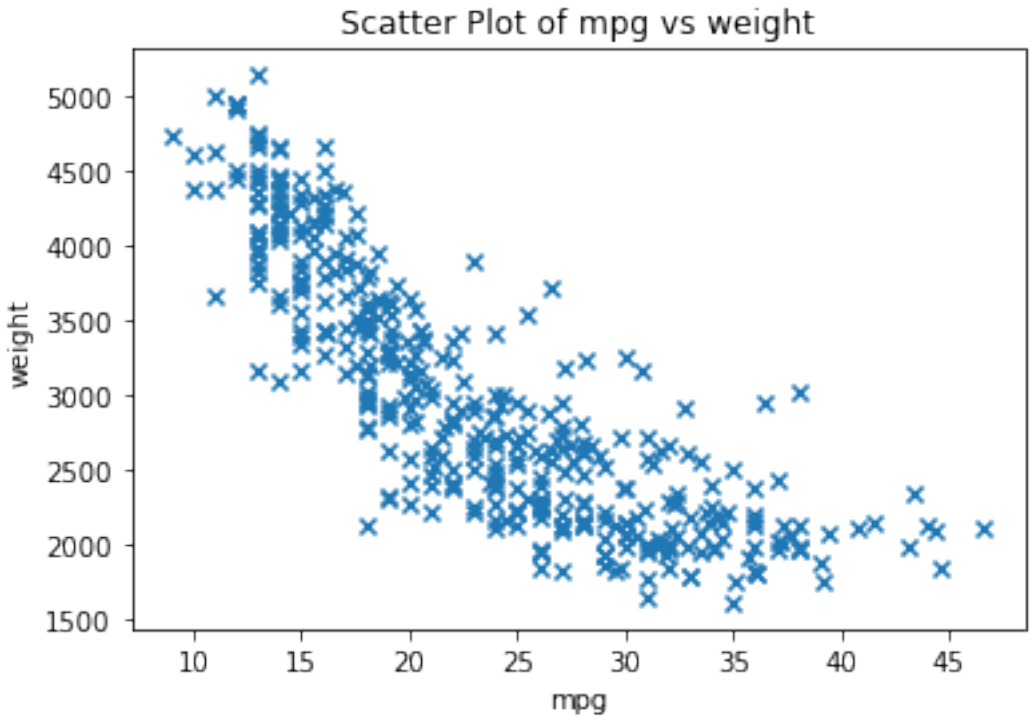
```
      mpg  cylinders  displacement  horsepower  weight  acceleration
year  \
0    18.0          8         307.0         130    3504          12.0
70
1    15.0          8         350.0         165    3693          11.5
70
2    18.0          8         318.0         150    3436          11.0
70
3    16.0          8         304.0         150    3433          12.0
70
4    17.0          8         302.0         140    3449          10.5
70
..    ...        ...           ...         ...     ...           ...
...
387  27.0          4         140.0          86    2790          15.6
82
388  44.0          4          97.0          52    2130          24.6
82
389  32.0          4         135.0          84    2295          11.6
82
390  28.0          4         120.0          79    2625          18.6
82
391  31.0          4         119.0          82    2720          19.4
82

     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
..      ...                        ...
387       1            ford mustang gl
388       2                  vw pickup
389       1              dodge rampage
390       1                ford ranger
391       1                 chevy s-10

[392 rows x 9 columns]
```

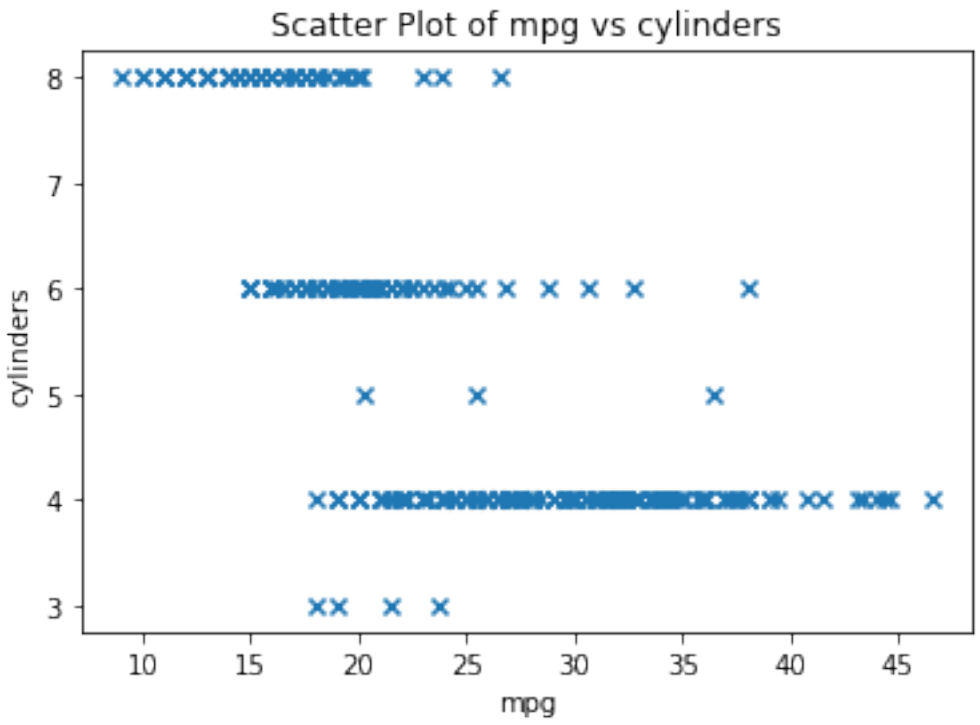
1. Scatterplots between features

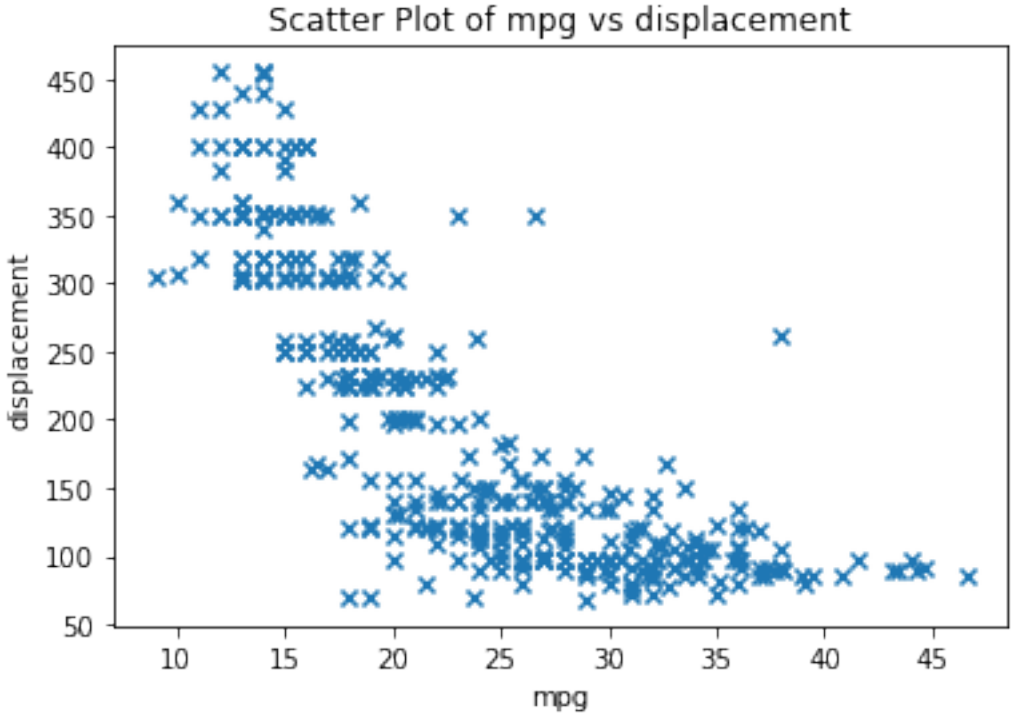
```
feat1 = data['mpg']
feat2 = data['weight']
feat3 = data['cylinders']
feat4 = data['displacement']
feat5 = data['horsepower']
feat6 = data['acceleration']
feat7 = data['year']
feat8 = data['origin']

plt.figure()
plt.scatter(feat1, feat2, marker='x')
plt.xlabel('mpg')
plt.ylabel('weight')
plt.title('Scatter Plot of mpg vs weight')

#Similarly add scatter plots for every pair of features

# x - mpg, y - rest of the features

plt.figure()
plt.scatter(feat1, feat3, marker='x')
plt.xlabel('mpg')
plt.ylabel('cylinders')
plt.title('Scatter Plot of mpg vs cylinders')

plt.figure()
plt.scatter(feat1, feat4, marker='x')
plt.xlabel('mpg')
plt.ylabel('displacement')
plt.title('Scatter Plot of mpg vs displacement')

plt.figure()
plt.scatter(feat1, feat5, marker='x')
plt.xlabel('mpg')
plt.ylabel('horsepower')
plt.title('Scatter Plot of mpg vs horsepower')

plt.figure()
plt.scatter(feat1, feat6, marker='x')
plt.xlabel('mpg')
plt.ylabel('acceleration')
plt.title('Scatter Plot of mpg vs acceleration')

plt.figure()
plt.scatter(feat1, feat7, marker='x')
plt.xlabel('mpg')
plt.ylabel('year')
plt.title('Scatter Plot of mpg vs year')
```

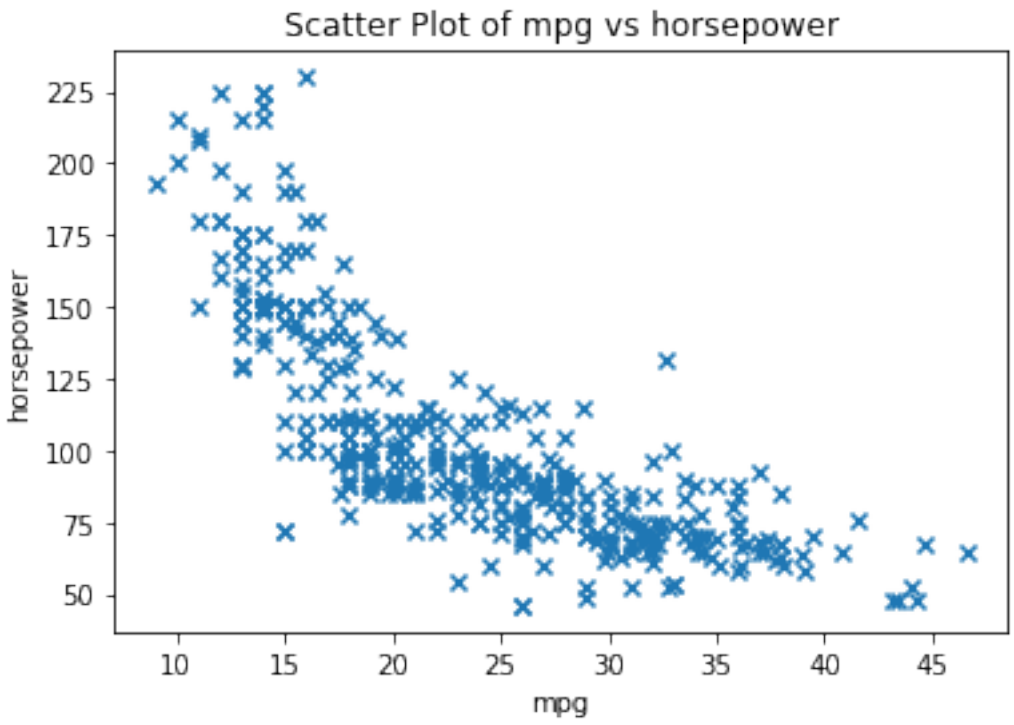
```
plt.figure()
plt.scatter(feat1, feat8, marker='x')
plt.xlabel('mpg')
plt.ylabel('origin')
plt.title('Scatter Plot of mpg vs origin')
```

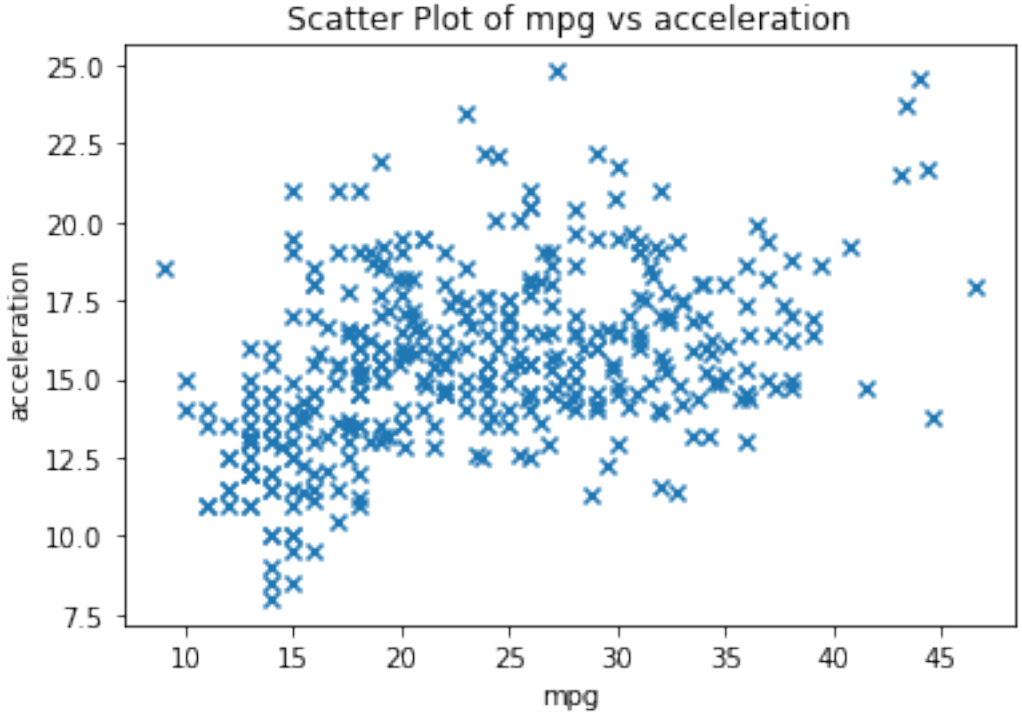
```
Text(0.5, 1.0, 'Scatter Plot of mpg vs origin')
```

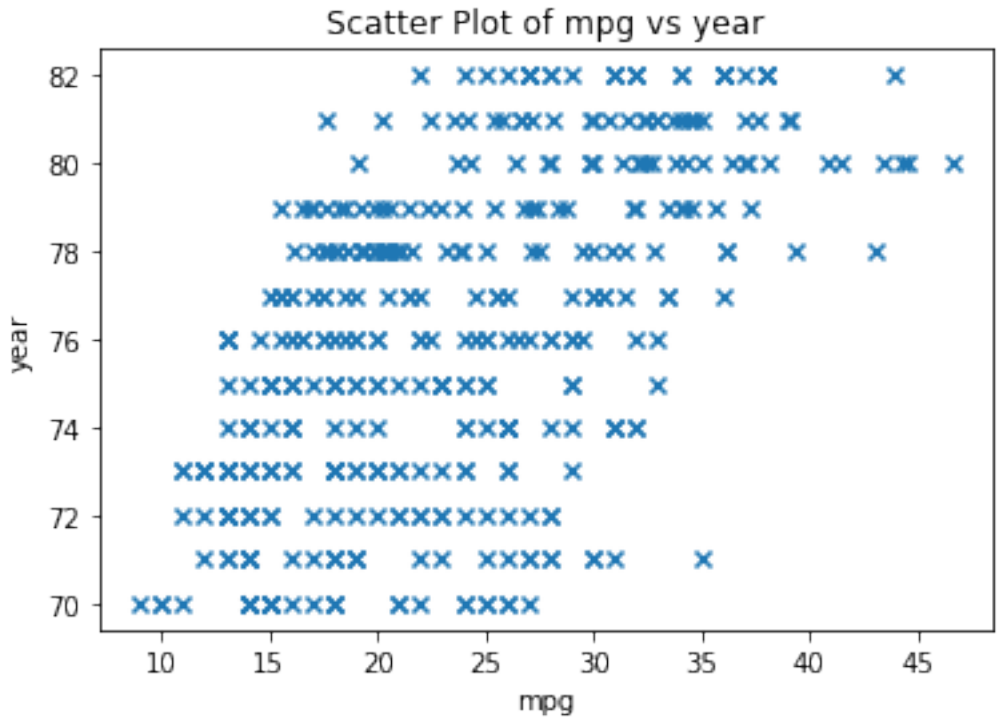


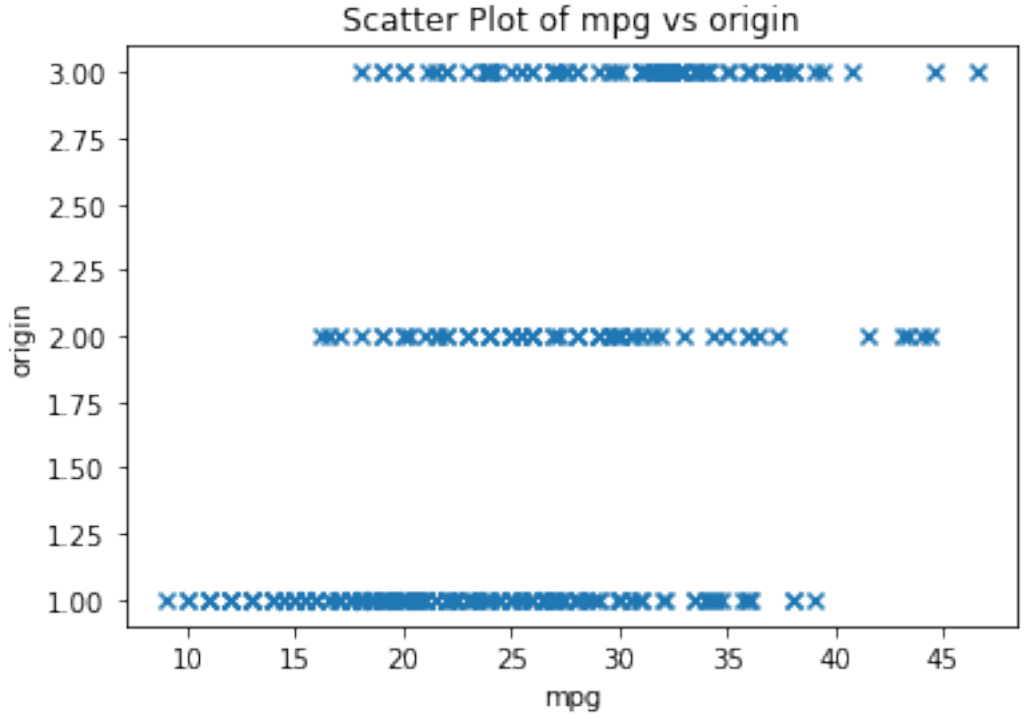

Scatter Plot of mpg vs cylinders
cylinders
mpg
8
7
6
5
4
3
10
15
20
25
30
35
40
45


Scatter Plot of mpg vs displacement
displacement
mpg
450
400
350
300
250
200
150
100
50
10
15
20
25
30
35
40
45


Scatter Plot of mpg vs horsepower
horsepower
mpg
225
200
175
150
125
100
75
50
10
15
20
25
30
35
40
45


Scatter Plot of mpg vs acceleration
acceleration
mpg
25.0
22.5
20.0
17.5
15.0
12.5
10.0
7.5
10
15
20
25
30
35
40
45






```
# Plot the feature plots to observe the relationships between all the
predictors

# x - weight, y - rest of the features
```

```
plt.figure()
plt.scatter(feat2, feat1, marker='x')
plt.xlabel('weight')
plt.ylabel('mpg')
plt.title('Scatter Plot of weight vs mpg')

plt.figure()
plt.scatter(feat2, feat3, marker='x')
plt.xlabel('weight')
plt.ylabel('cylinders')
plt.title('Scatter Plot of weight vs cylinders')

plt.figure()
plt.scatter(feat2, feat4, marker='x')
plt.xlabel('weight')
plt.ylabel('displacement')
plt.title('Scatter Plot of weight vs displacement')

plt.figure()
plt.scatter(feat2, feat5, marker='x')
plt.xlabel('weight')
plt.ylabel('horsepower')
plt.title('Scatter Plot of weight vs horsepower')

plt.figure()
plt.scatter(feat2, feat6, marker='x')
plt.xlabel('weight')
plt.ylabel('acceleration')
plt.title('Scatter Plot of weight vs acceleration')

plt.figure()
plt.scatter(feat2, feat7, marker='x')
plt.xlabel('weight')
plt.ylabel('year')
plt.title('Scatter Plot of weight vs year')

plt.figure()
plt.scatter(feat2, feat8, marker='x')
plt.xlabel('weight')
plt.ylabel('origin')
plt.title('Scatter Plot of weight vs origin')

# x - cylinders, y - rest of the features

plt.figure()
plt.scatter(feat3, feat1, marker='x')
plt.xlabel('cylinders')
plt.ylabel('mpg')
plt.title('Scatter Plot of cylinders vs mpg')
```

```
plt.figure()
plt.scatter(feat3, feat2, marker='x')
plt.xlabel('cylinders')
plt.ylabel('weight')
plt.title('Scatter Plot of cylinders vs weight')

plt.figure()
plt.scatter(feat3, feat4, marker='x')
plt.xlabel('cylinders')
plt.ylabel('displacement')
plt.title('Scatter Plot of cylinders vs displacement')

plt.figure()
plt.scatter(feat3, feat5, marker='x')
plt.xlabel('cylinders')
plt.ylabel('horsepower')
plt.title('Scatter Plot of cylinders vs horsepower')

plt.figure()
plt.scatter(feat3, feat6, marker='x')
plt.xlabel('cylinders')
plt.ylabel('acceleration')
plt.title('Scatter Plot of cylinders vs acceleration')

plt.figure()
plt.scatter(feat3, feat7, marker='x')
plt.xlabel('cylinders')
plt.ylabel('year')
plt.title('Scatter Plot of cylinders vs year')

plt.figure()
plt.scatter(feat3, feat8, marker='x')
plt.xlabel('cylinders')
plt.ylabel('origin')
plt.title('Scatter Plot of cylinders vs origin')

# x - displacement, y - rest of the features

plt.figure()
plt.scatter(feat4, feat1, marker='x')
plt.xlabel('displacement')
plt.ylabel('mpg')
plt.title('Scatter Plot of displacement vs mpg')

plt.figure()
plt.scatter(feat4, feat2, marker='x')
plt.xlabel('displacement')
plt.ylabel('weight')
```

```
plt.title('Scatter Plot of displacement vs weight')

plt.figure()
plt.scatter(feat4, feat3, marker='x')
plt.xlabel('displacement')
plt.ylabel('cylinders')
plt.title('Scatter Plot of displacement vs cylinders')

plt.figure()
plt.scatter(feat4, feat5, marker='x')
plt.xlabel('displacement')
plt.ylabel('horsepower')
plt.title('Scatter Plot of displacement vs horsepower')

plt.figure()
plt.scatter(feat4, feat6, marker='x')
plt.xlabel('displacement')
plt.ylabel('acceleration')
plt.title('Scatter Plot of displacement vs acceleration')

plt.figure()
plt.scatter(feat4, feat7, marker='x')
plt.xlabel('displacement')
plt.ylabel('year')
plt.title('Scatter Plot of displacement vs year')

plt.figure()
plt.scatter(feat4, feat8, marker='x')
plt.xlabel('displacement')
plt.ylabel('origin')
plt.title('Scatter Plot of displacement vs origin')

# x - horsepower, y - rest of the features

plt.figure()
plt.scatter(feat5, feat1, marker='x')
plt.xlabel('horsepower')
plt.ylabel('mpg')
plt.title('Scatter Plot of horsepower vs mpg')

plt.figure()
plt.scatter(feat5, feat2, marker='x')
plt.xlabel('horsepower')
plt.ylabel('weight')
plt.title('Scatter Plot of horsepower vs weight')

plt.figure()
plt.scatter(feat5, feat3, marker='x')
plt.xlabel('horsepower')
```

```
plt.ylabel('cylinders')
plt.title('Scatter Plot of horsepower vs cylinders')

plt.figure()
plt.scatter(feat5, feat4, marker='x')
plt.xlabel('horsepower')
plt.ylabel('displacement')
plt.title('Scatter Plot of horsepower vs displacement')

plt.figure()
plt.scatter(feat5, feat6, marker='x')
plt.xlabel('horsepower')
plt.ylabel('acceleration')
plt.title('Scatter Plot of horsepower vs acceleration')

plt.figure()
plt.scatter(feat5, feat7, marker='x')
plt.xlabel('horsepower')
plt.ylabel('year')
plt.title('Scatter Plot of horsepower vs year')

plt.figure()
plt.scatter(feat5, feat8, marker='x')
plt.xlabel('horsepower')
plt.ylabel('origin')
plt.title('Scatter Plot of horsepower vs origin')

# x - acceleration, y - rest of the features

plt.figure()
plt.scatter(feat6, feat1, marker='x')
plt.xlabel('acceleration')
plt.ylabel('mpg')
plt.title('Scatter Plot of acceleration vs mpg')

plt.figure()
plt.scatter(feat6, feat2, marker='x')
plt.xlabel('acceleration')
plt.ylabel('weight')
plt.title('Scatter Plot of acceleration vs weight')

plt.figure()
plt.scatter(feat6, feat3, marker='x')
plt.xlabel('acceleration')
plt.ylabel('cylinders')
plt.title('Scatter Plot of acceleration vs cylinders')

plt.figure()
plt.scatter(feat6, feat4, marker='x')
```

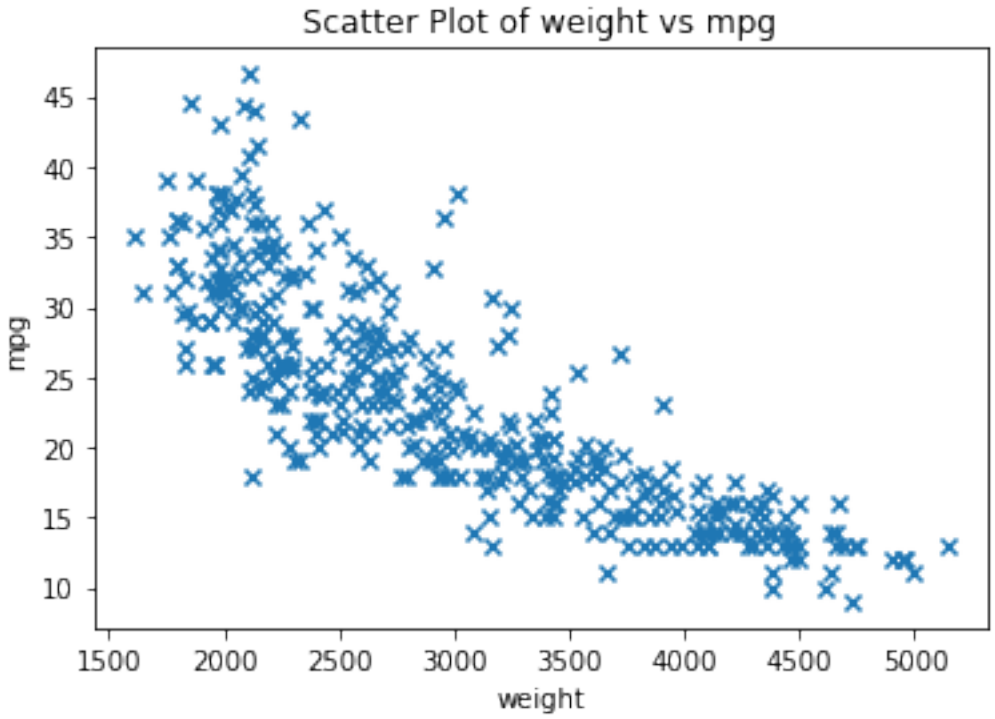
```
plt.xlabel('acceleration')
plt.ylabel('displacement')
plt.title('Scatter Plot of acceleration vs displacement')

plt.figure()
plt.scatter(feat6, feat5, marker='x')
plt.xlabel('acceleration')
plt.ylabel('horsepower')
plt.title('Scatter Plot of acceleration vs horsepower')

plt.figure()
plt.scatter(feat6, feat7, marker='x')
plt.xlabel('acceleration')
plt.ylabel('year')
plt.title('Scatter Plot of acceleration vs year')

plt.figure()
plt.scatter(feat6, feat8, marker='x')
plt.xlabel('acceleration')
plt.ylabel('origin')
plt.title('Scatter Plot of acceleration vs origin')

# x - year, y - rest of the features

plt.figure()
plt.scatter(feat7, feat1, marker='x')
plt.xlabel('year')
plt.ylabel('mpg')
plt.title('Scatter Plot of year vs mpg')

plt.figure()
plt.scatter(feat7, feat2, marker='x')
plt.xlabel('year')
plt.ylabel('weight')
plt.title('Scatter Plot of year vs weight')

plt.figure()
plt.scatter(feat7, feat3, marker='x')
plt.xlabel('year')
plt.ylabel('cylinders')
plt.title('Scatter Plot of year vs cylinders')

plt.figure()
plt.scatter(feat7, feat4, marker='x')
plt.xlabel('year')
plt.ylabel('displacement')
plt.title('Scatter Plot of year vs displacement')

plt.figure()
```

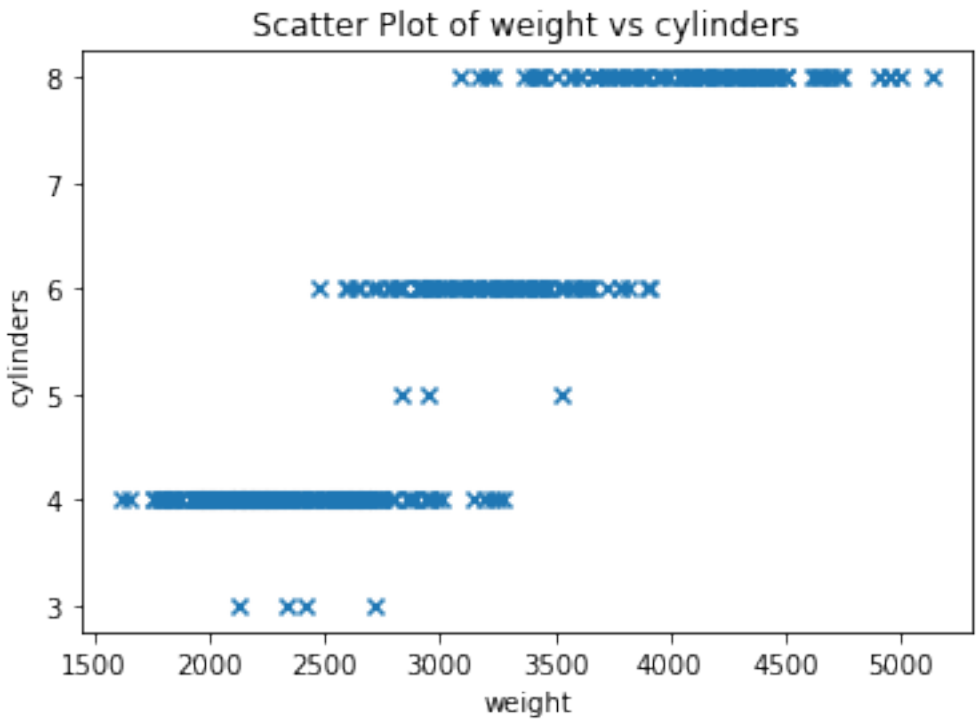
```
plt.scatter(feat7, feat5, marker='x')
plt.xlabel('year')
plt.ylabel('horsepower')
plt.title('Scatter Plot of year vs horsepower')

plt.figure()
plt.scatter(feat7, feat6, marker='x')
plt.xlabel('year')
plt.ylabel('acceleration')
plt.title('Scatter Plot of year vs acceleration')

plt.figure()
plt.scatter(feat7, feat8, marker='x')
plt.xlabel('year')
plt.ylabel('origin')
plt.title('Scatter Plot of year vs origin')

# x - origin, y - rest of the features

plt.figure()
plt.scatter(feat8, feat1, marker='x')
plt.xlabel('origin')
plt.ylabel('mpg')
plt.title('Scatter Plot of origin vs mpg')

plt.figure()
plt.scatter(feat8, feat2, marker='x')
plt.xlabel('origin')
plt.ylabel('weight')
plt.title('Scatter Plot of origin vs weight')

plt.figure()
plt.scatter(feat8, feat3, marker='x')
plt.xlabel('origin')
plt.ylabel('cylinders')
plt.title('Scatter Plot of origin vs cylinders')

plt.figure()
plt.scatter(feat8, feat4, marker='x')
plt.xlabel('origin')
plt.ylabel('displacement')
plt.title('Scatter Plot of origin vs displacement')

plt.figure()
plt.scatter(feat8, feat5, marker='x')
plt.xlabel('origin')
plt.ylabel('horsepower')
plt.title('Scatter Plot of origin vs horsepower')
```

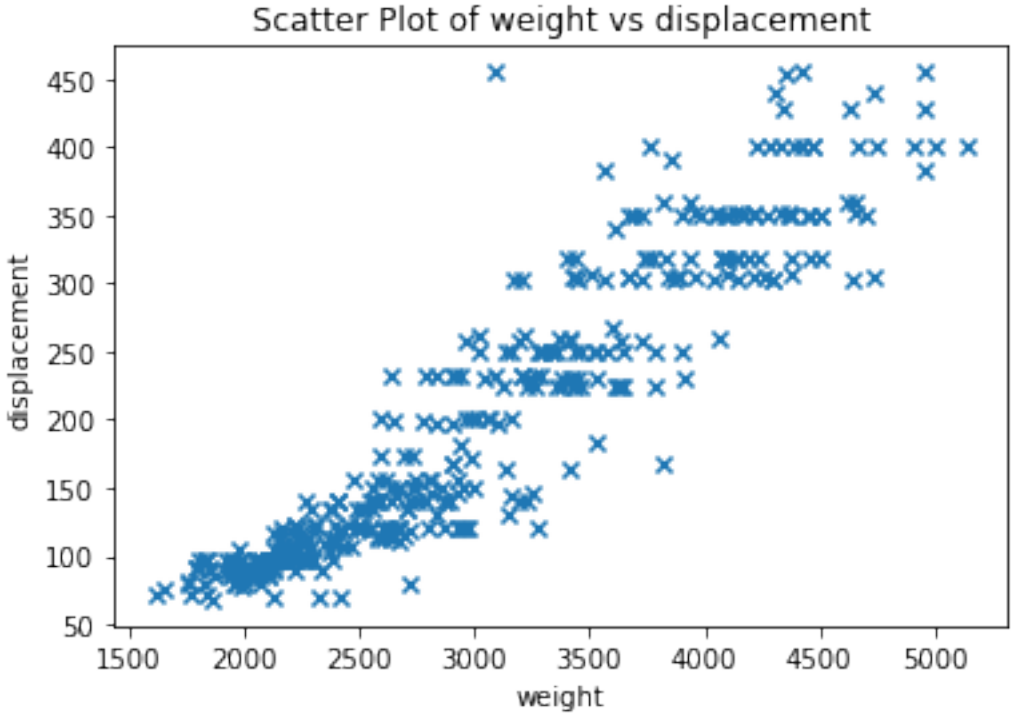
```
plt.figure()
plt.scatter(feat8, feat6, marker='x')
plt.xlabel('origin')
plt.ylabel('acceleration')
plt.title('Scatter Plot of origin vs acceleration')

plt.figure()
plt.scatter(feat8, feat7, marker='x')
plt.xlabel('origin')
plt.ylabel('year')
plt.title('Scatter Plot of origin vs year')
```

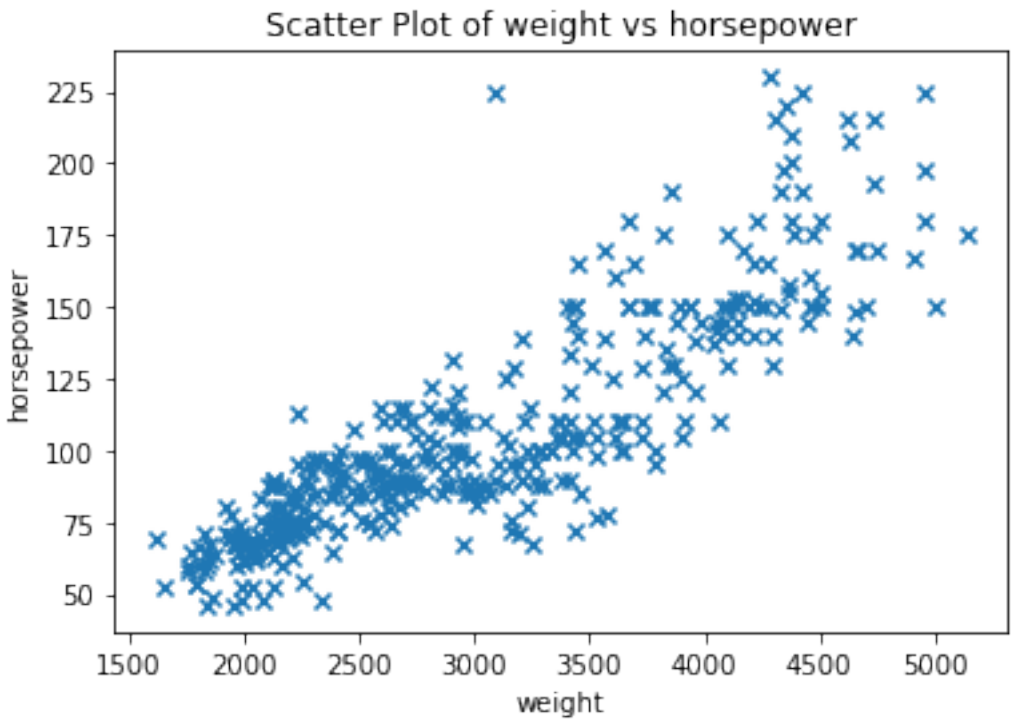
```
C:\Users\techn\AppData\Local\Temp/ipykernel_18484/502524609.py:129:
RuntimeWarning: More than 20 figures have been opened. Figures created
through the pyplot interface (`matplotlib.pyplot.figure`) are retained
until explicitly closed and may consume too much memory. (To control
this warning, see the rcParam `figure.max_open_warning`).
  plt.figure()
```

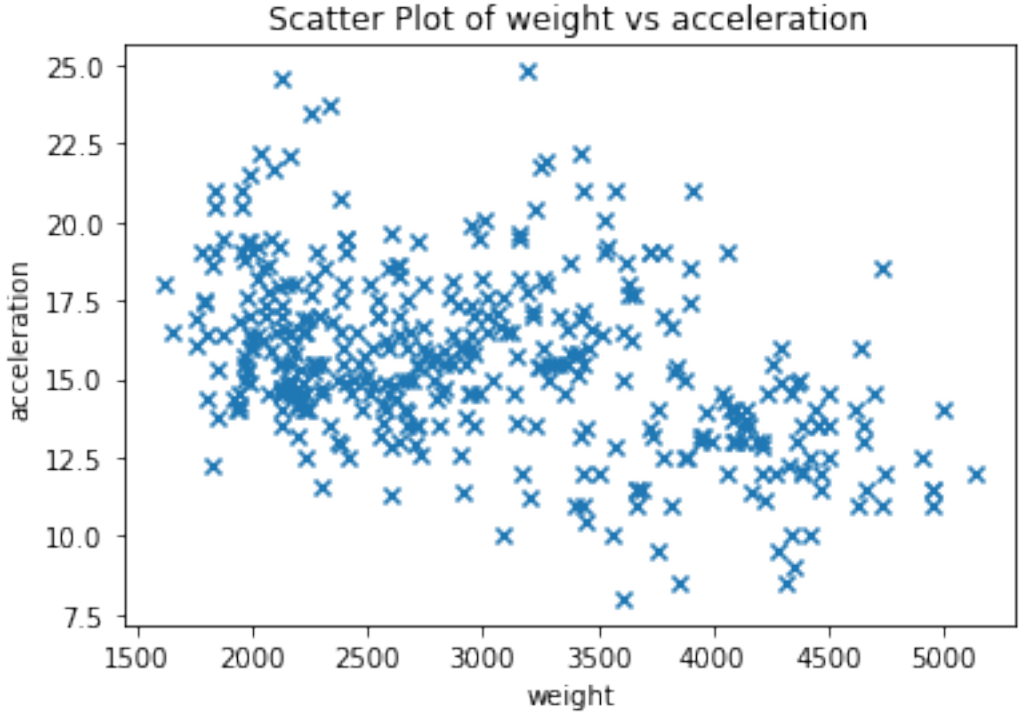
```
Text(0.5, 1.0, 'Scatter Plot of origin vs year')
```

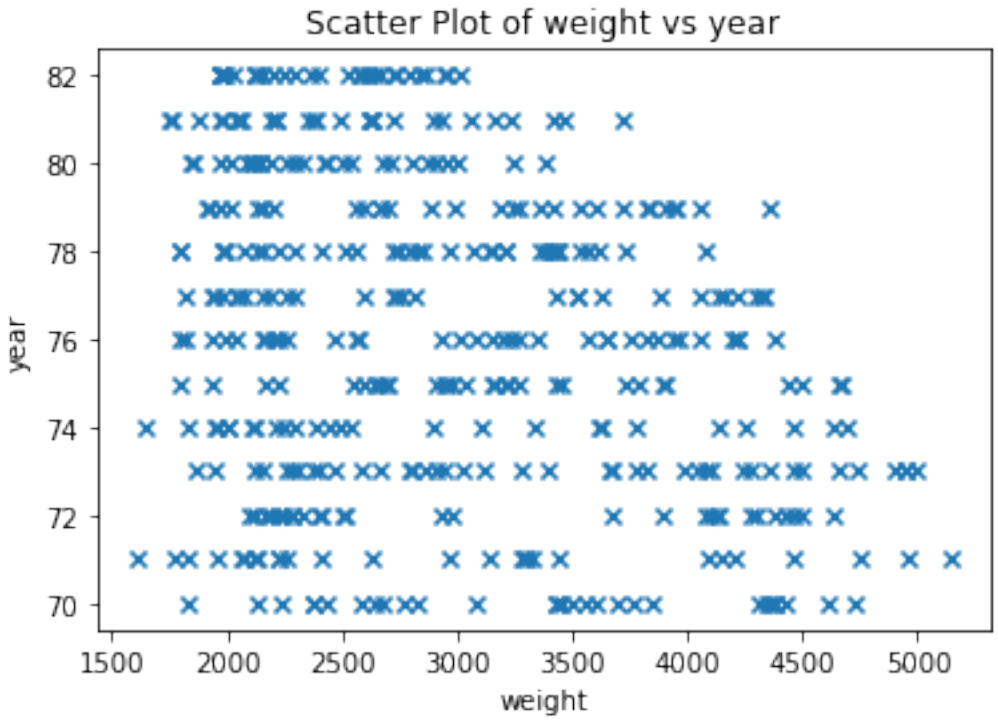


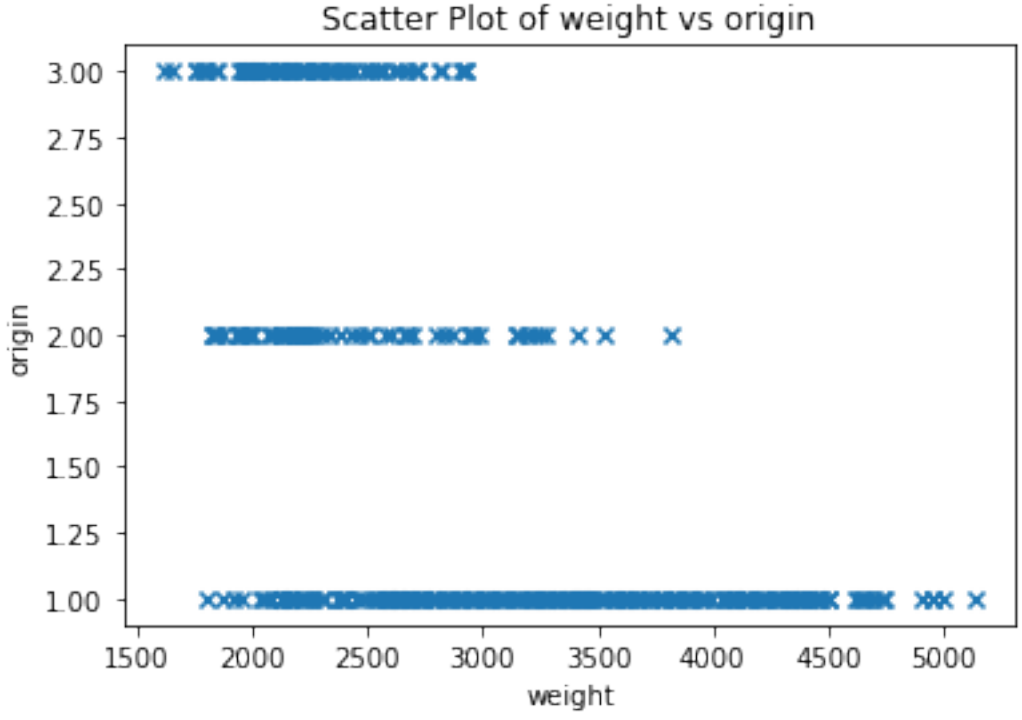

Scatter Plot of weight vs cylinders
cylinders
weight
8
7
6
5
4
3
1500
2000
2500
3000
3500
4000
4500
5000

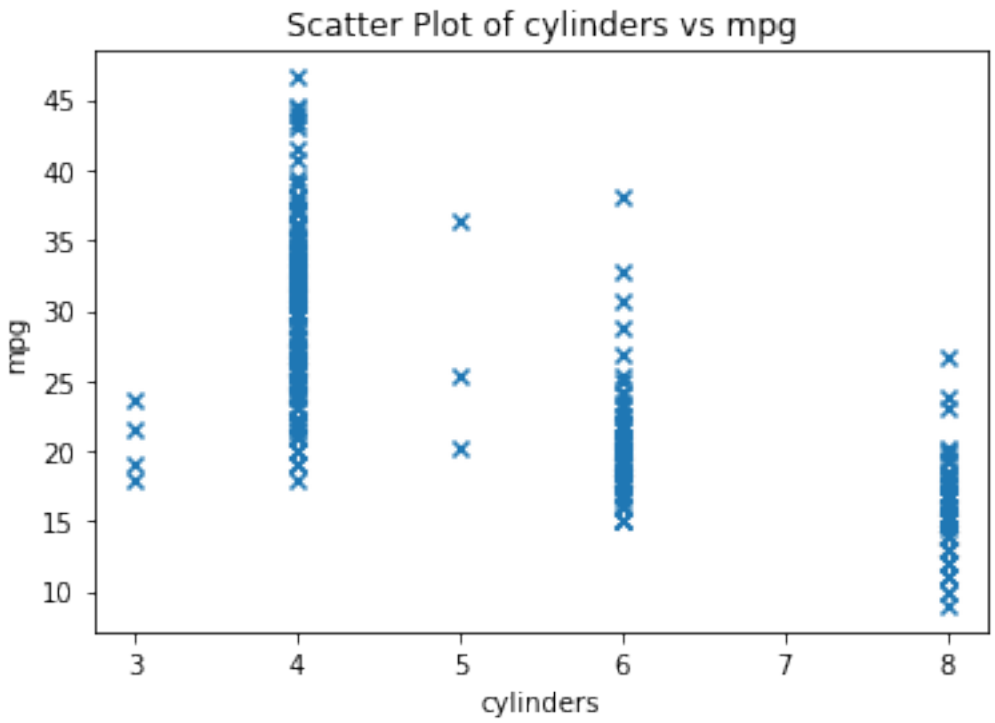

Scatter Plot of weight vs displacement
displacement
weight
450
400
350
300
250
200
150
100
50
1500
2000
2500
3000
3500
4000
4500
5000

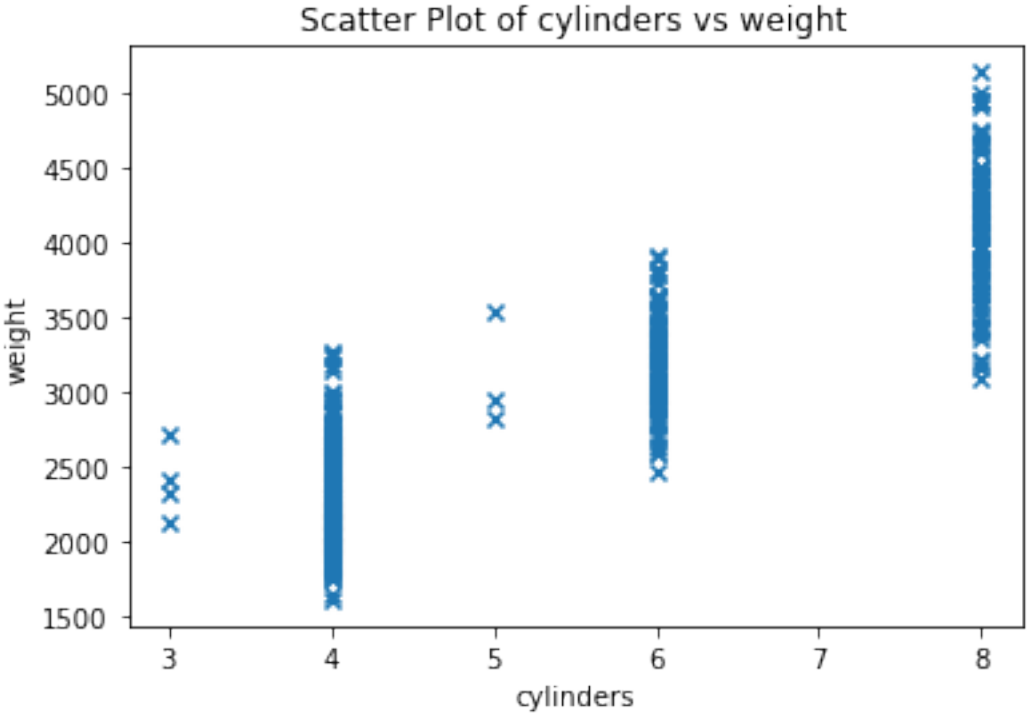

Scatter Plot of weight vs horsepower
horsepower
225
200
175
150
125
100
75
50
1500
2000
2500
3000
3500
4000
4500
5000
weight

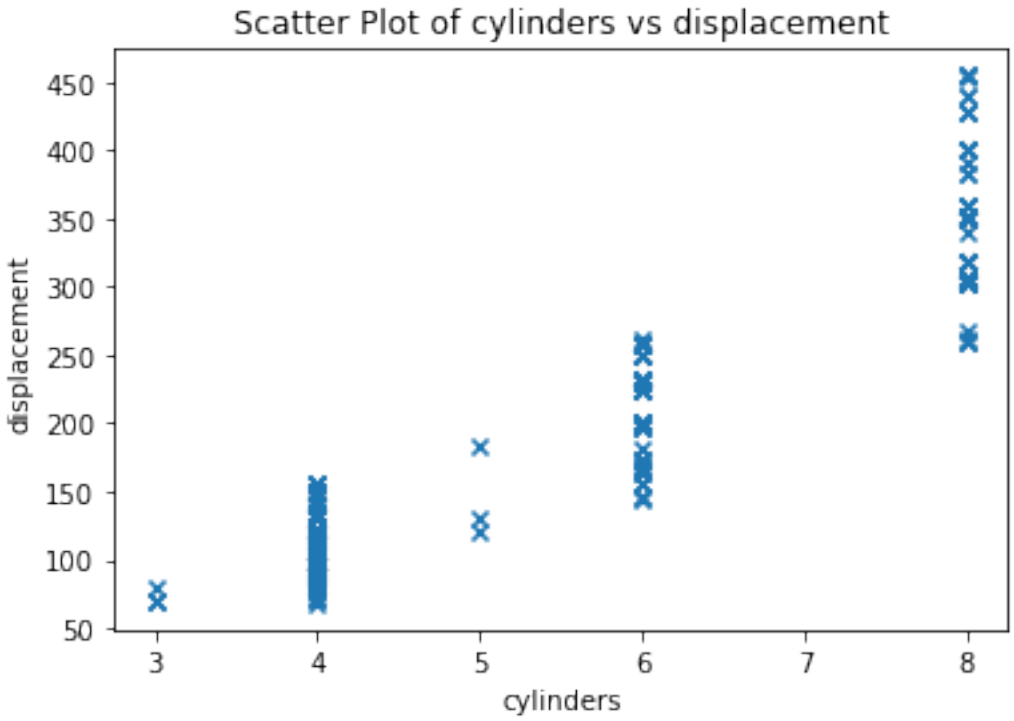

Scatter Plot of weight vs acceleration
acceleration
25.0
22.5
20.0
17.5
15.0
12.5
10.0
7.5
1500
2000
2500
3000
3500
4000
4500
5000
weight

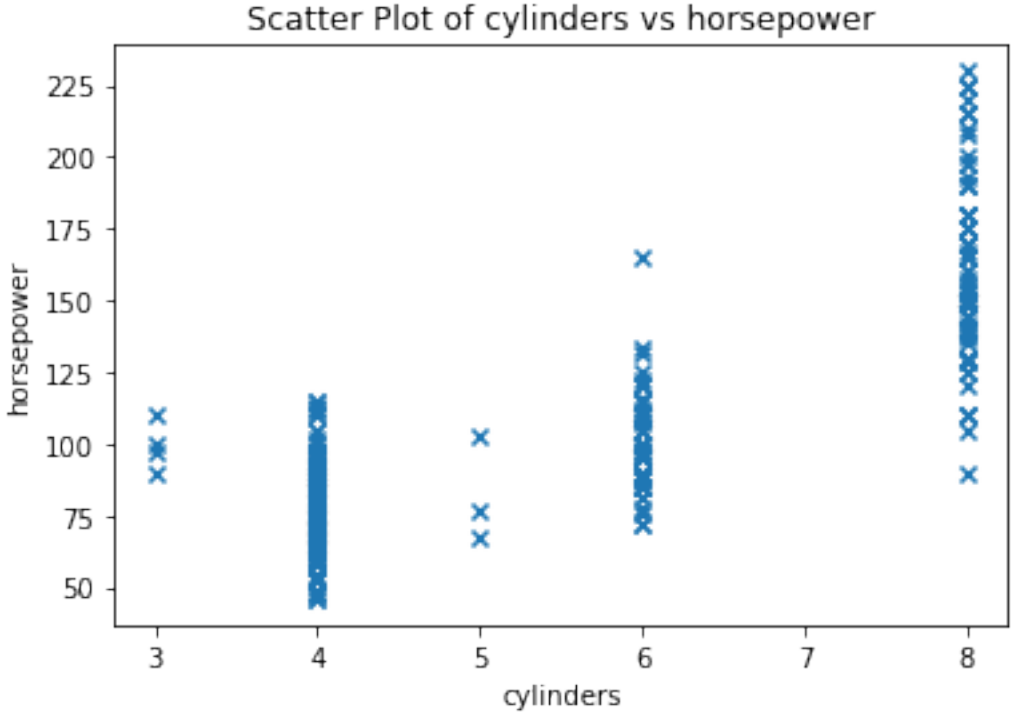

Scatter Plot of weight vs year
year
weight
82
80
78
76
74
72
70
1500
2000
2500
3000
3500
4000
4500
5000

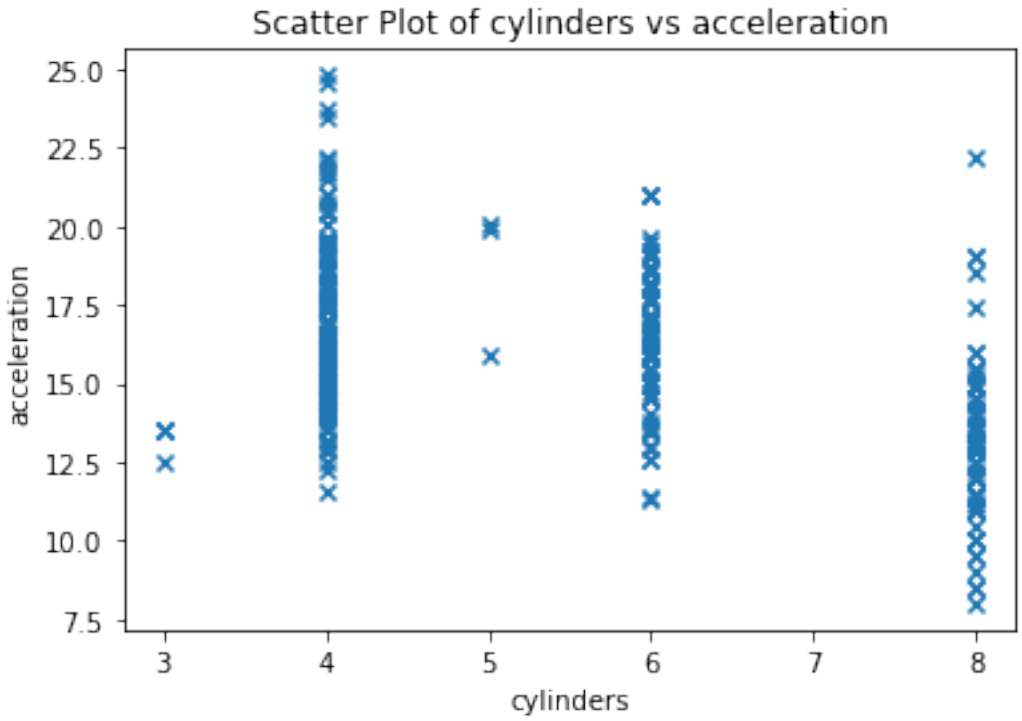

Scatter Plot of weight vs origin
origin
weight
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1.00
1500
2000
2500
3000
3500
4000
4500
5000

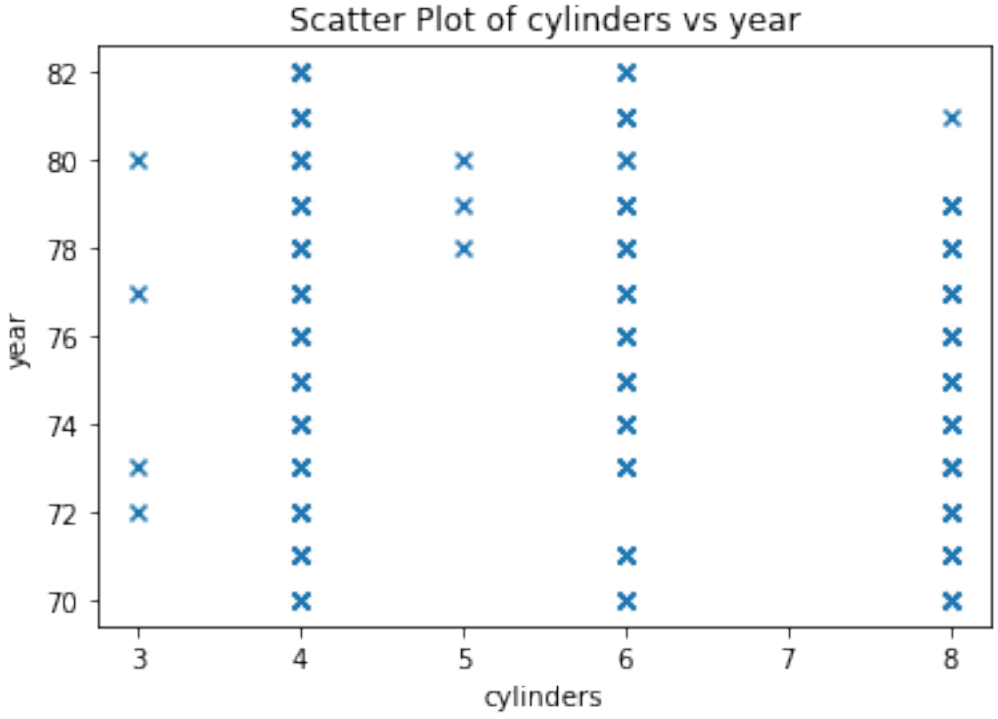

Scatter Plot of cylinders vs mpg
mpg
cylinders
45
40
35
30
25
20
15
10
3
4
5
6
7
8

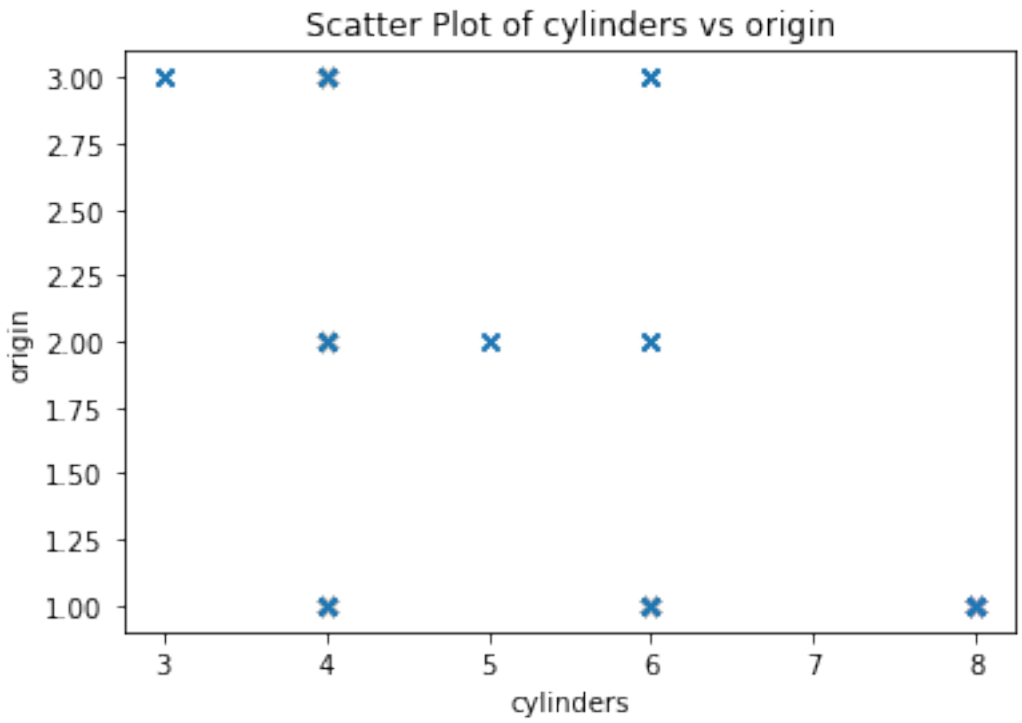

Scatter Plot of cylinders vs weight
weight
cylinders
5000
4500
4000
3500
3000
2500
2000
1500
3
4
5
6
7
8

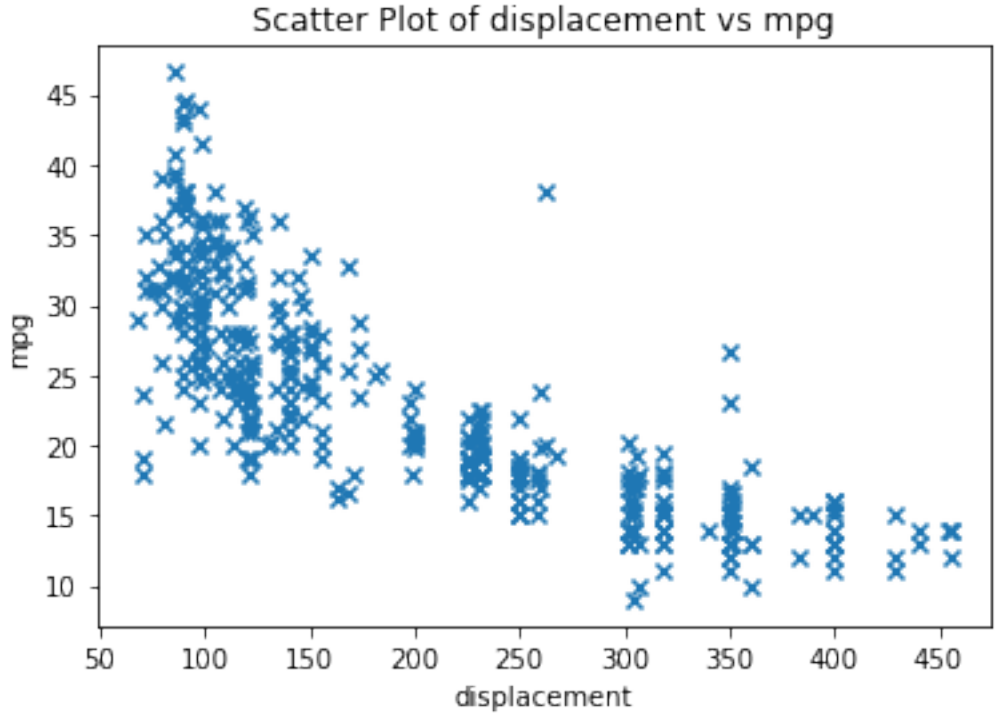

Scatter Plot of cylinders vs displacement
displacement
cylinders
50
100
150
200
250
300
350
400
450
3
4
5
6
7
8

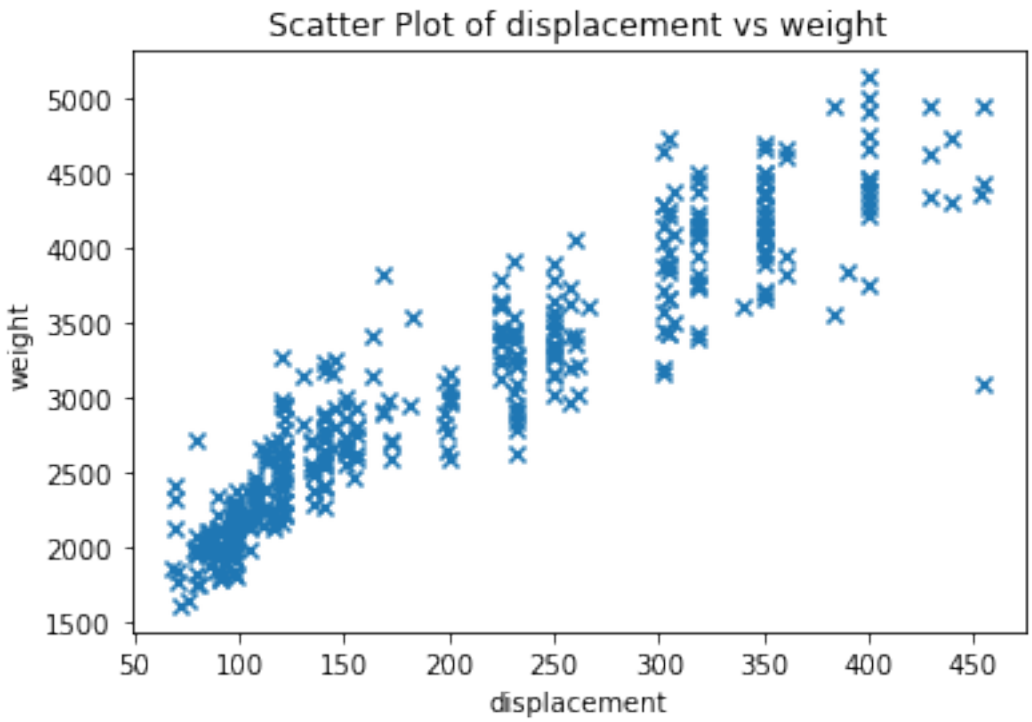

Scatter Plot of cylinders vs horsepower
horsepower
cylinders
50
75
100
125
150
175
200
225
3
4
5
6
7
8

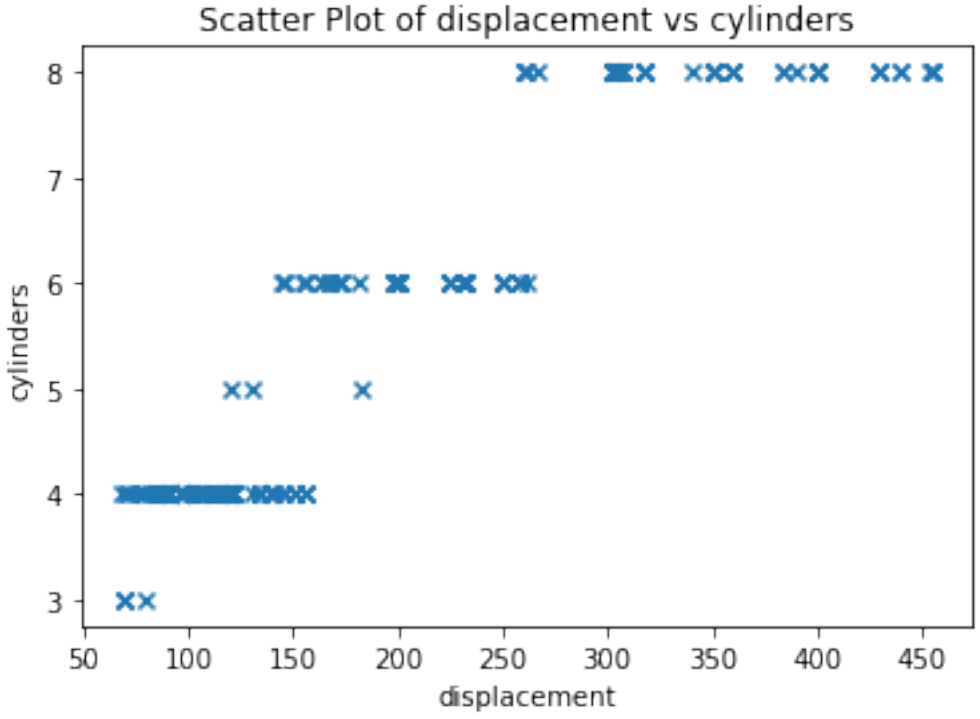

Scatter Plot of cylinders vs acceleration
acceleration
cylinders
25.0
22.5
20.0
17.5
15.0
12.5
10.0
7.5
3
4
5
6
7
8

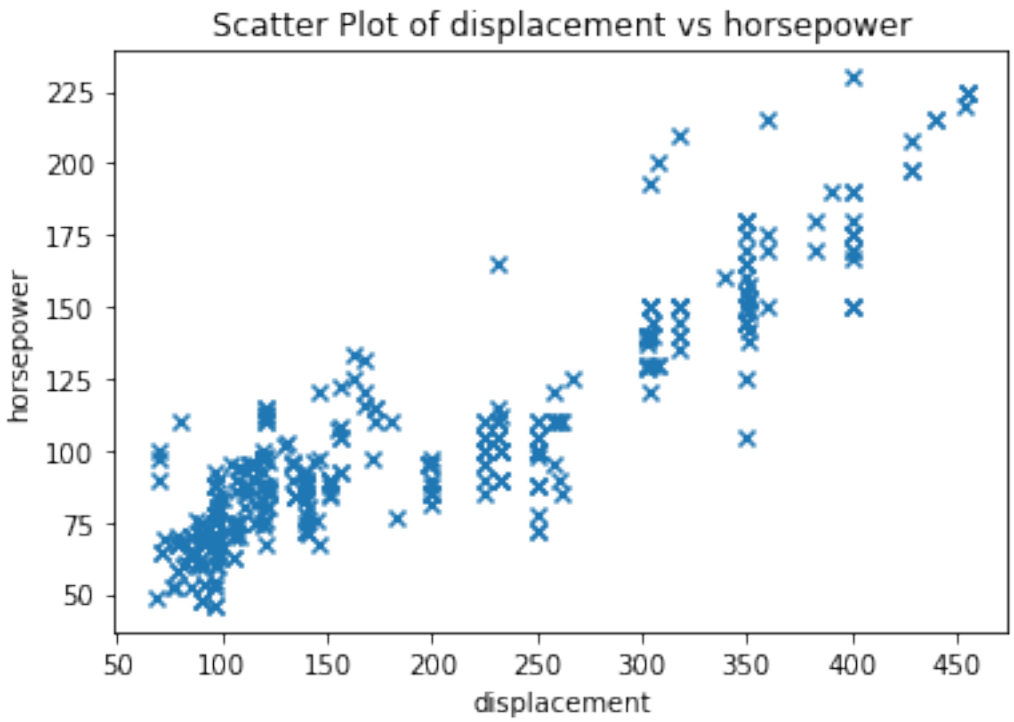

Scatter Plot of cylinders vs year
year
cylinders
82
80
78
76
74
72
70
3
4
5
6
7
8

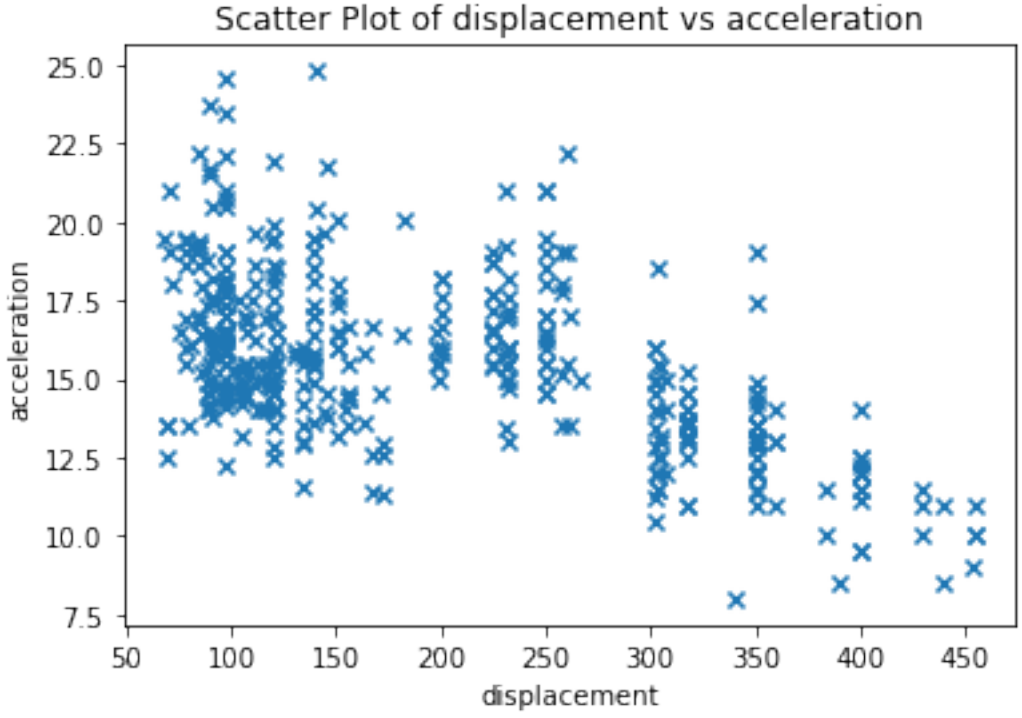

Scatter Plot of cylinders vs origin
origin
cylinders
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1.00
3
4
5
6
7
8

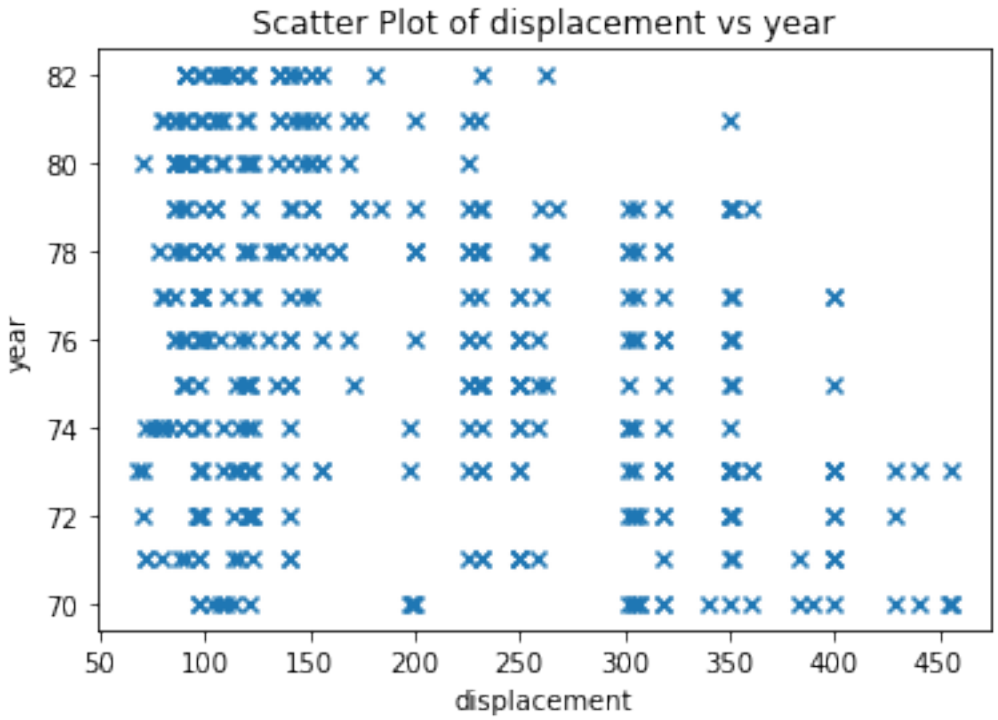

Scatter Plot of displacement vs mpg
mpg
displacement
45
40
35
30
25
20
15
10
50
100
150
200
250
300
350
400
450

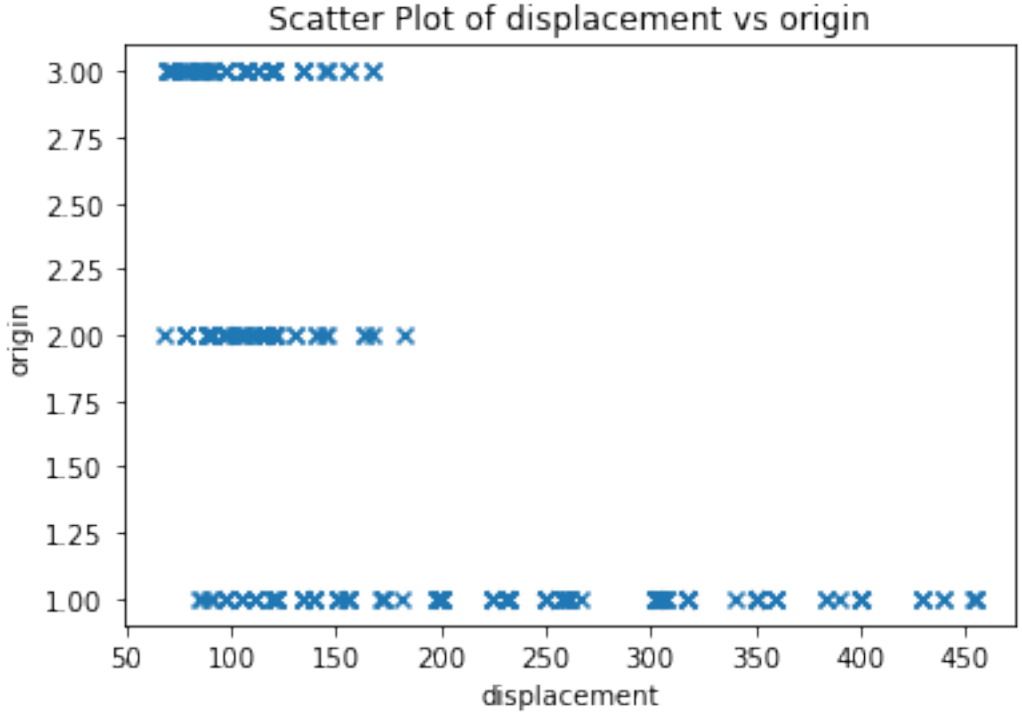

Scatter Plot of displacement vs weight
weight
displacement
5000
4500
4000
3500
3000
2500
2000
1500
50
100
150
200
250
300
350
400
450

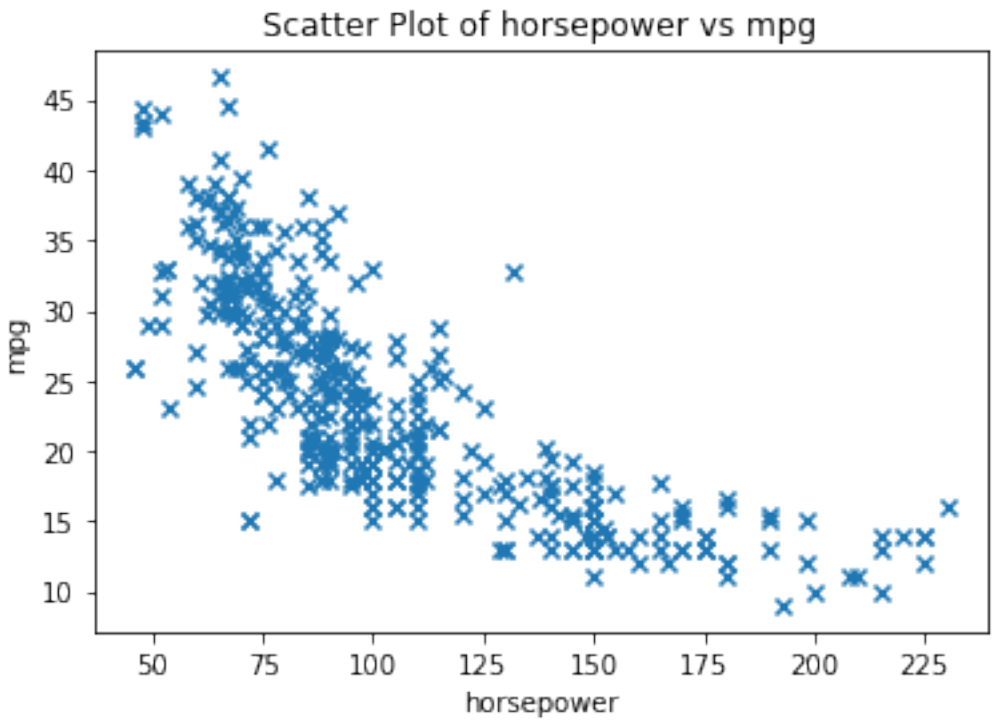

Scatter Plot of displacement vs cylinders
cylinders
displacement
8
7
6
5
4
3
50
100
150
200
250
300
350
400
450

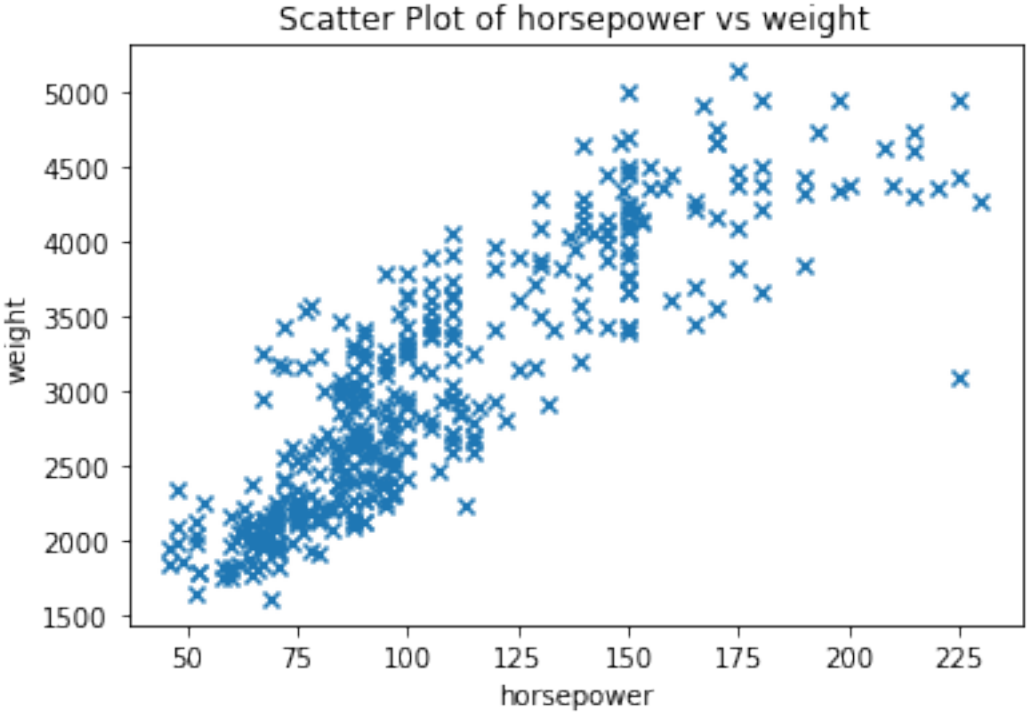

Scatter Plot of displacement vs horsepower
horsepower
displacement
225
200
175
150
125
100
75
50
50
100
150
200
250
300
350
400
450

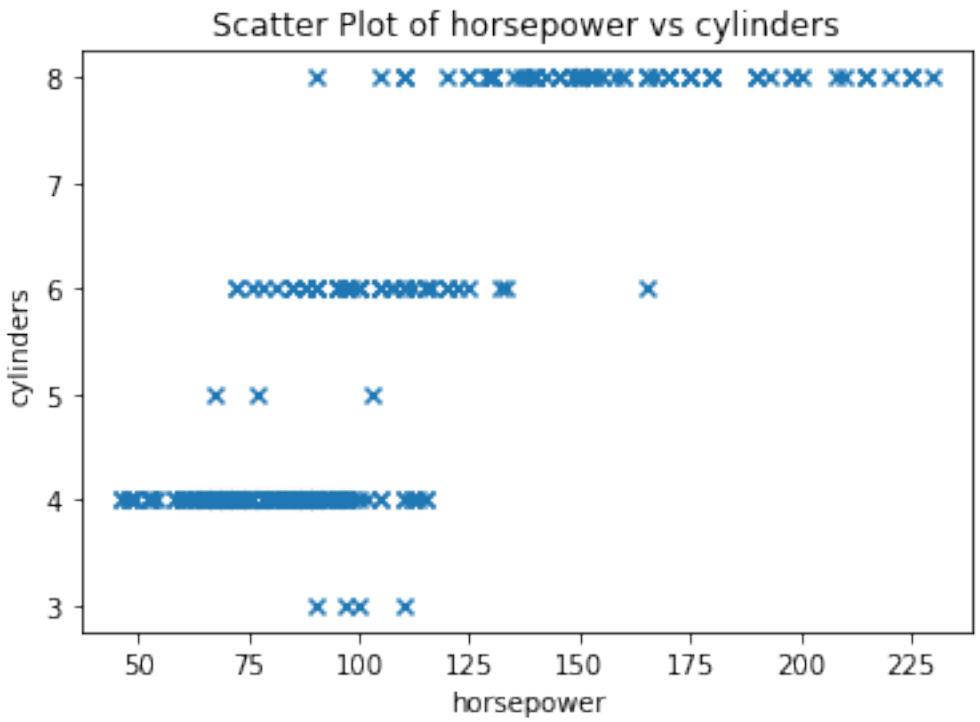

Scatter Plot of displacement vs acceleration
acceleration
displacement
25.0
22.5
20.0
17.5
15.0
12.5
10.0
7.5
50
100
150
200
250
300
350
400
450

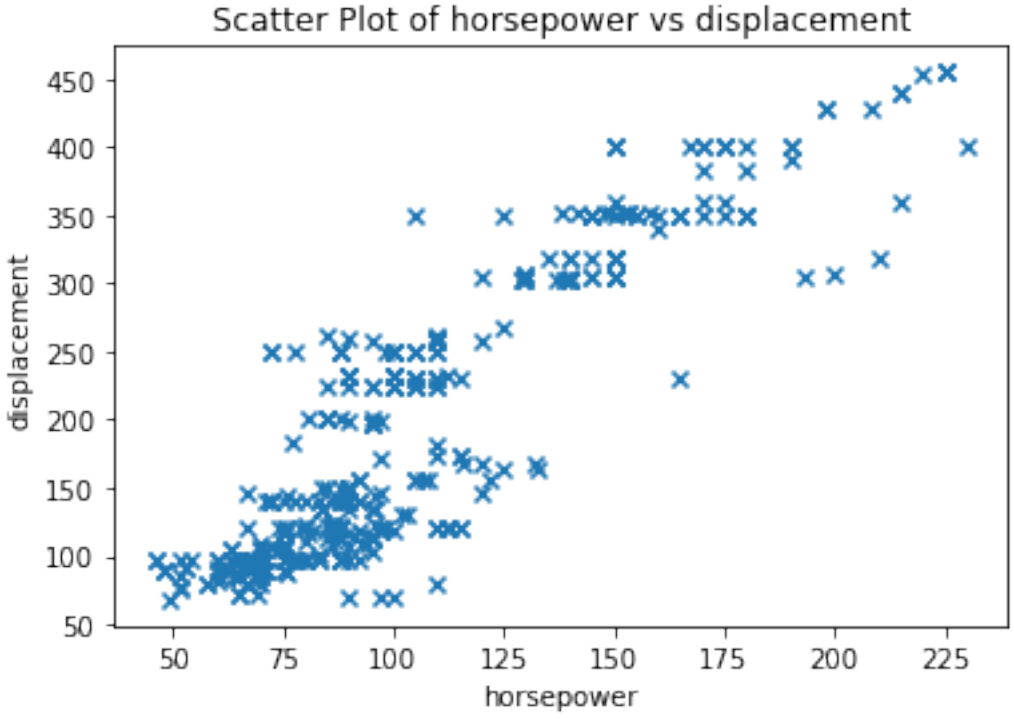

Scatter Plot of displacement vs year
year
displacement
82
80
78
76
74
72
70
50
100
150
200
250
300
350
400
450

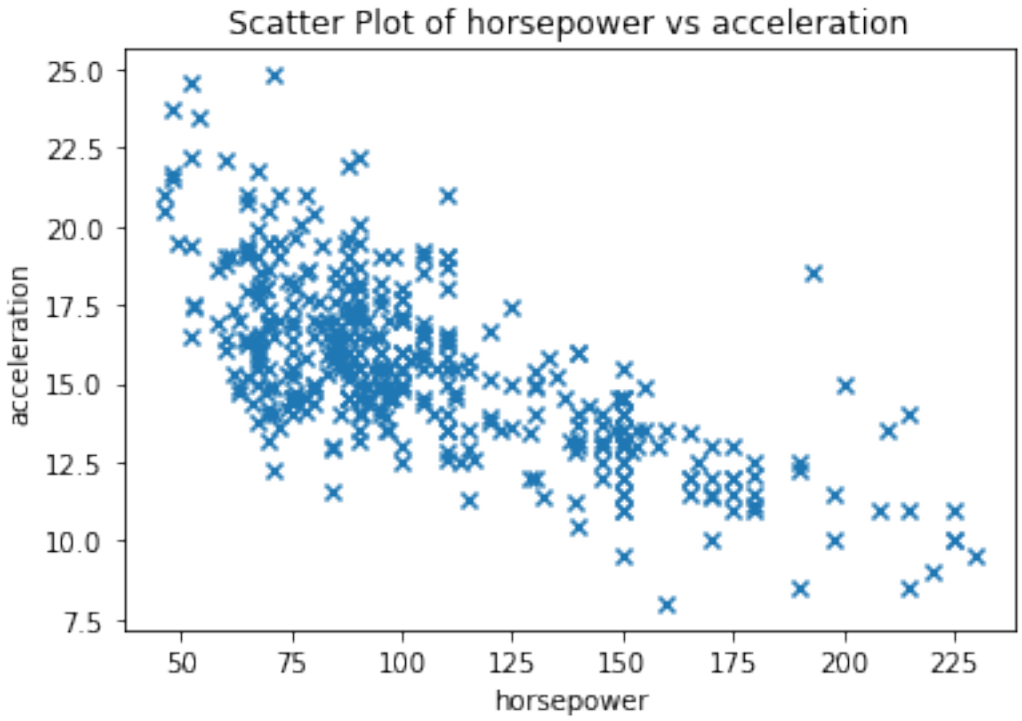

Scatter Plot of displacement vs origin
origin
displacement
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1.00
50
100
150
200
250
300
350
400
450

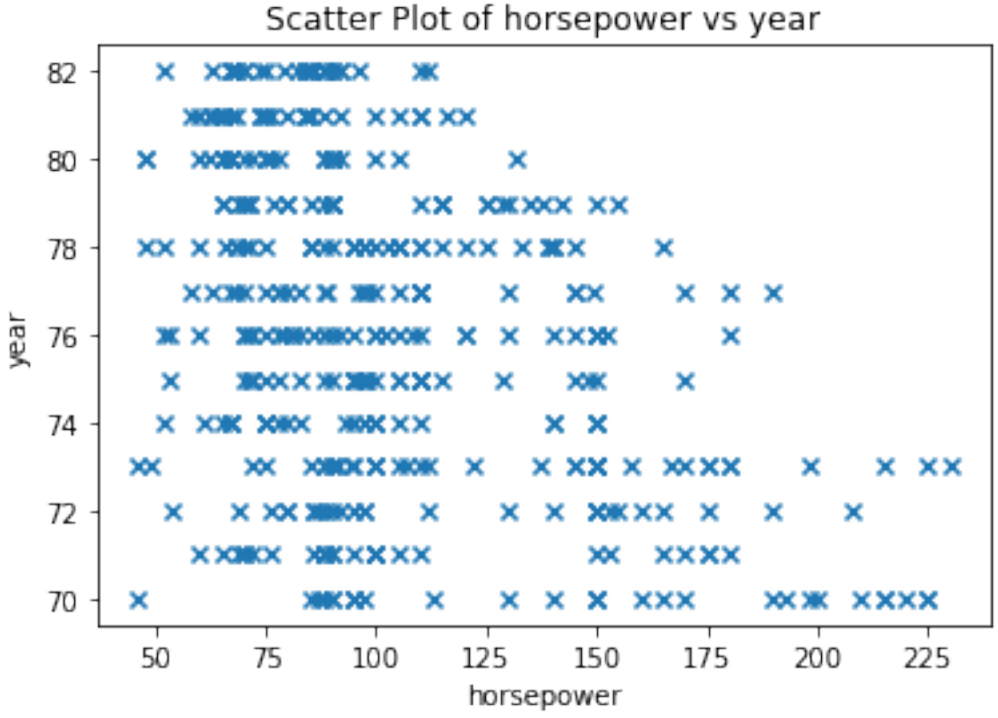

Scatter Plot of horsepower vs mpg
mpg
horsepower
45
40
35
30
25
20
15
10
50
75
100
125
150
175
200
225

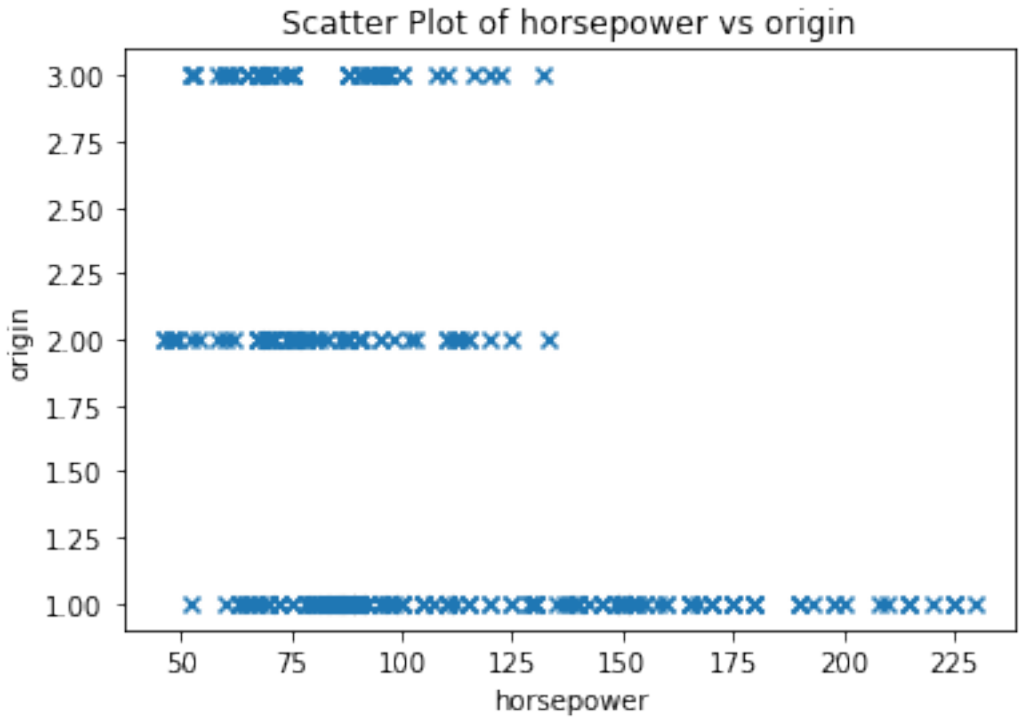

Scatter Plot of horsepower vs weight
weight
horsepower
5000
4500
4000
3500
3000
2500
2000
1500
50
75
100
125
150
175
200
225

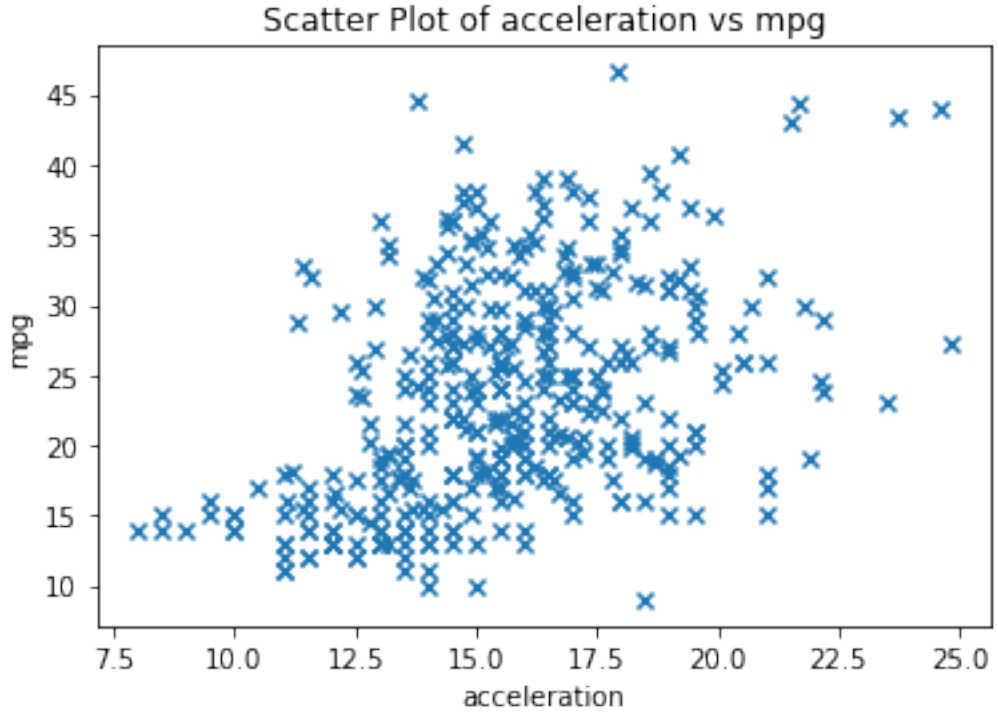

Scatter Plot of horsepower vs cylinders
cylinders
horsepower
8
7
6
5
4
3
50
75
100
125
150
175
200
225

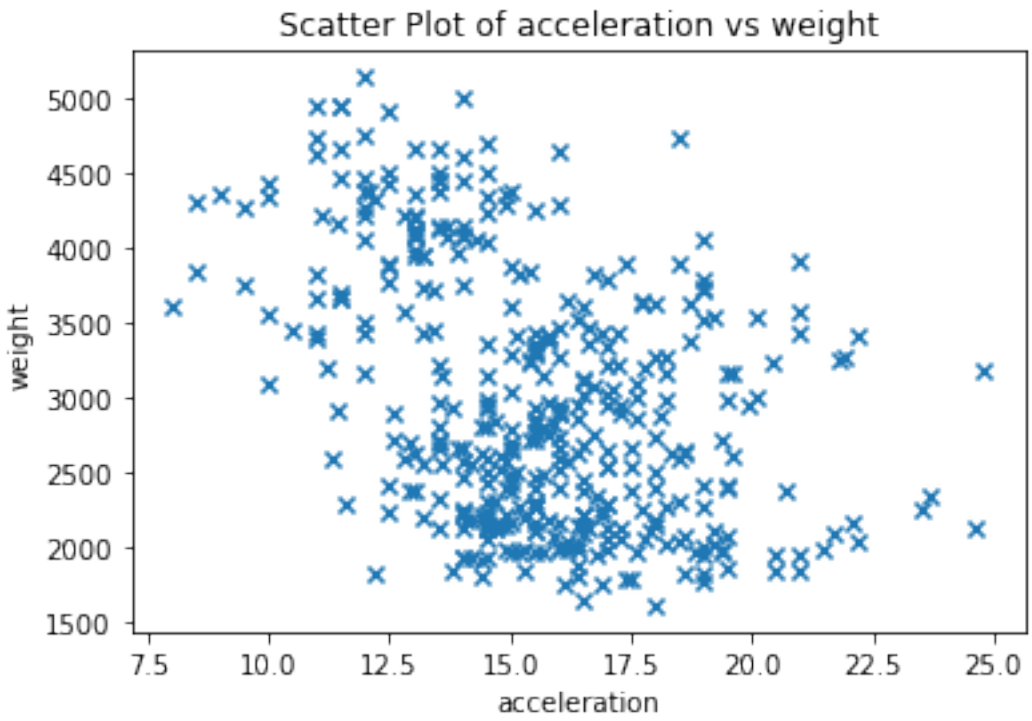

Scatter Plot of horsepower vs displacement
displacement
horsepower
450
400
350
300
250
200
150
100
50
50
75
100
125
150
175
200
225

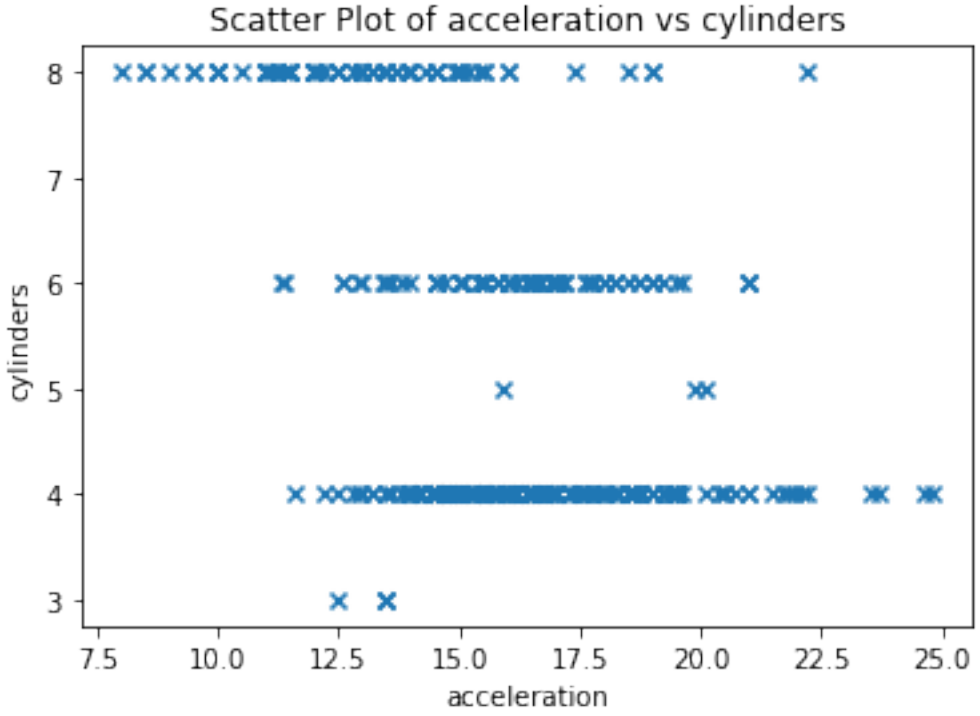

Scatter Plot of horsepower vs acceleration
acceleration
horsepower
25.0
22.5
20.0
17.5
15.0
12.5
10.0
7.5
50
75
100
125
150
175
200
225

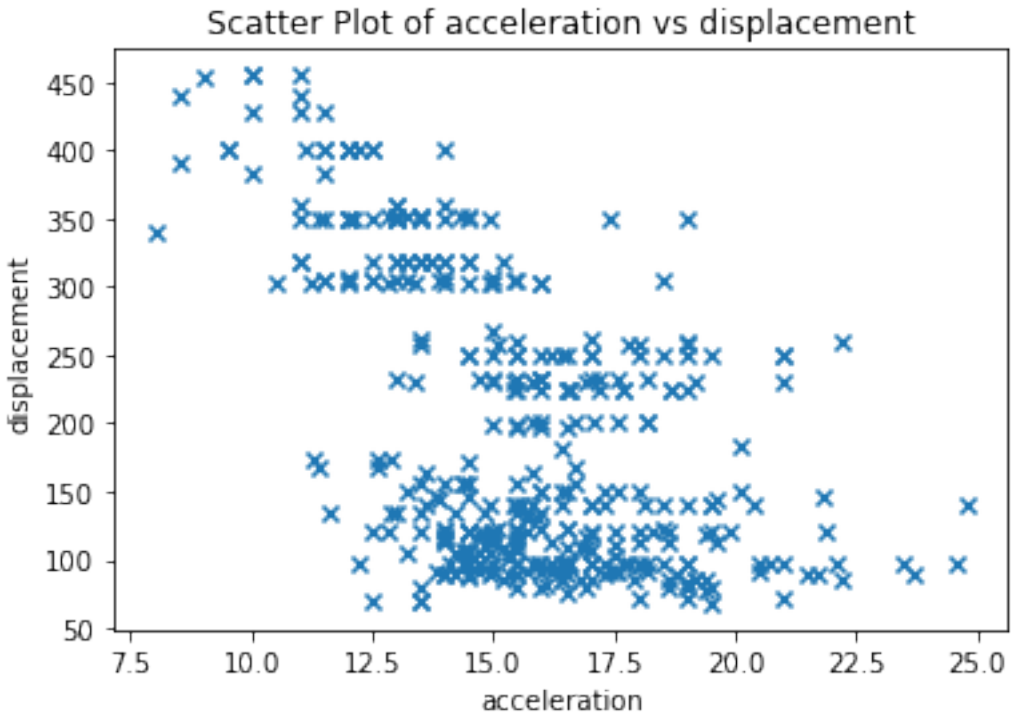

Scatter Plot of horsepower vs year
year
horsepower
82
80
78
76
74
72
70
50
75
100
125
150
175
200
225

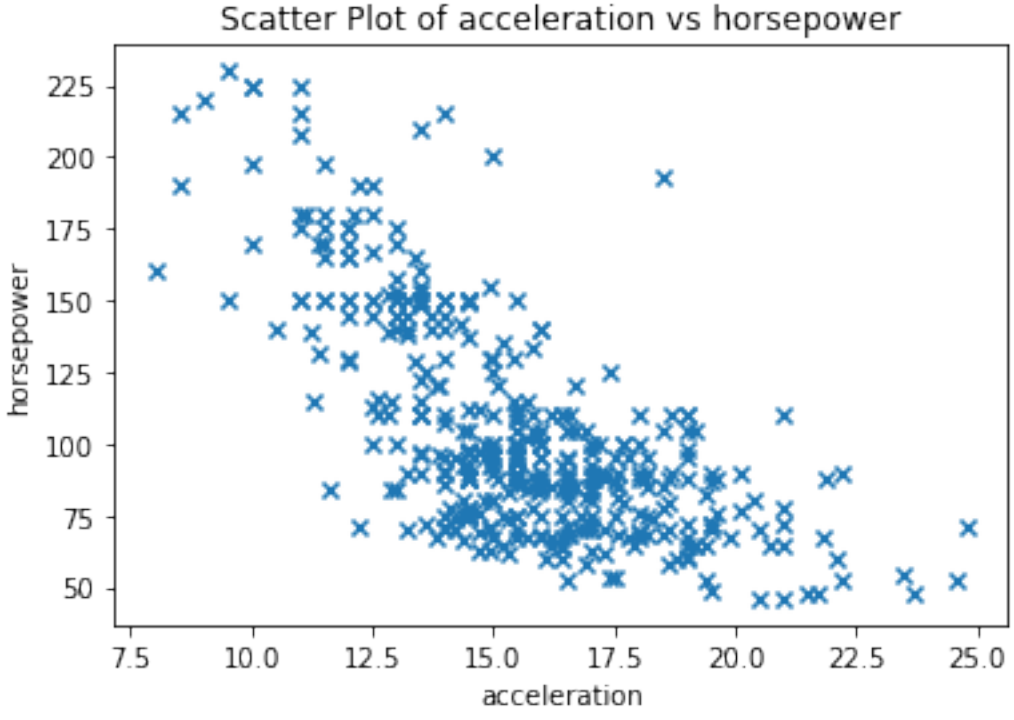

Scatter Plot of horsepower vs origin
origin
horsepower
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1.00
50
75
100
125
150
175
200
225

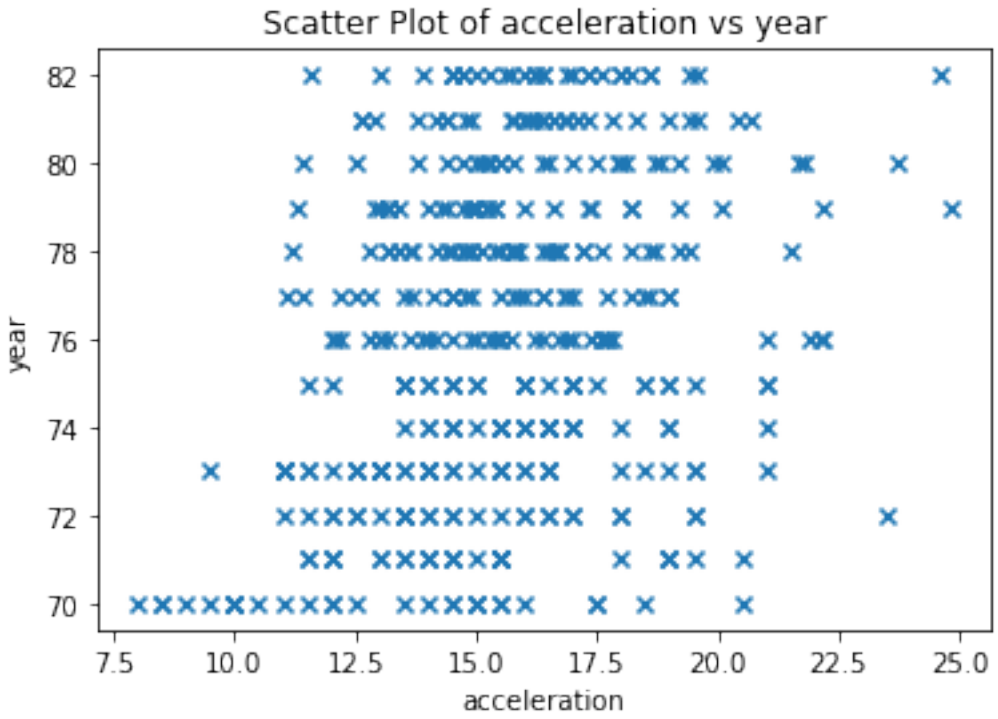

Scatter Plot of acceleration vs mpg
mpg
acceleration
45
40
35
30
25
20
15
10
7.5
10.0
12.5
15.0
17.5
20.0
22.5
25.0

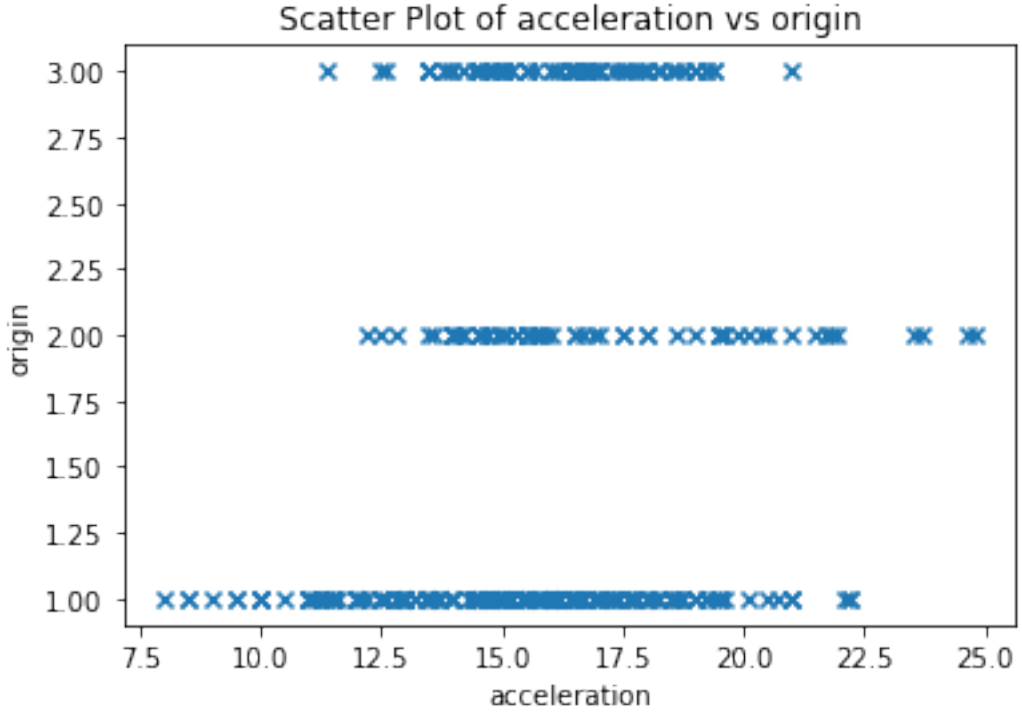

Scatter Plot of acceleration vs weight
weight
5000
4500
4000
3500
3000
2500
2000
1500
7.5
10.0
12.5
15.0
17.5
20.0
22.5
25.0
acceleration

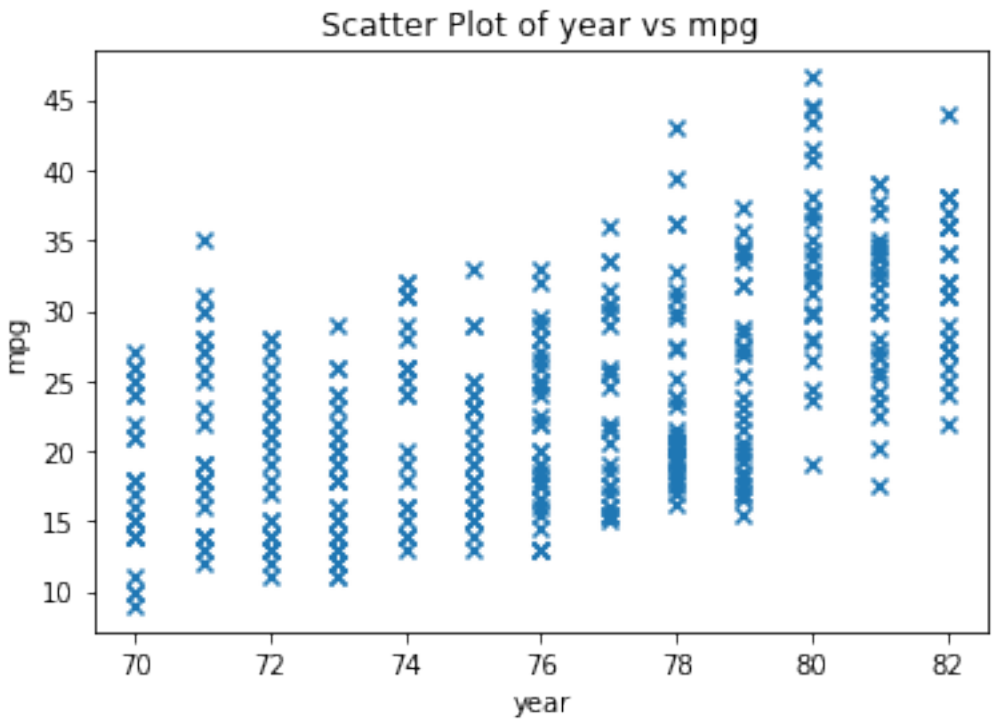

Scatter Plot of acceleration vs cylinders
cylinders
8
7
6
5
4
3
7.5
10.0
12.5
15.0
17.5
20.0
22.5
25.0
acceleration

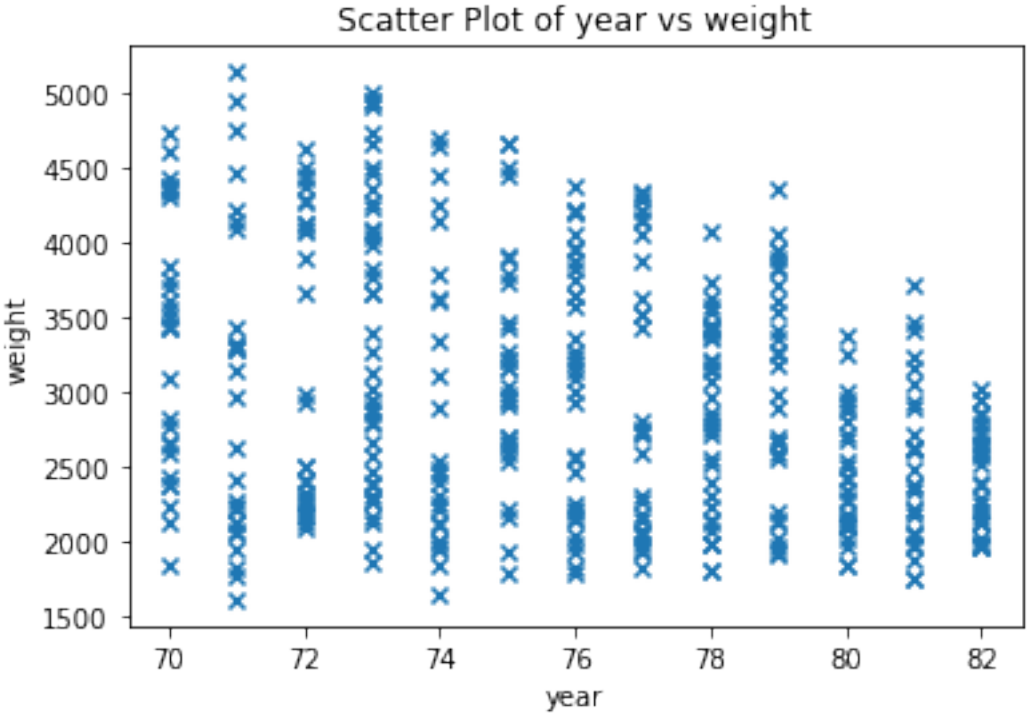

Scatter Plot of acceleration vs displacement
displacement
acceleration
450
400
350
300
250
200
150
100
50
7.5
10.0
12.5
15.0
17.5
20.0
22.5
25.0

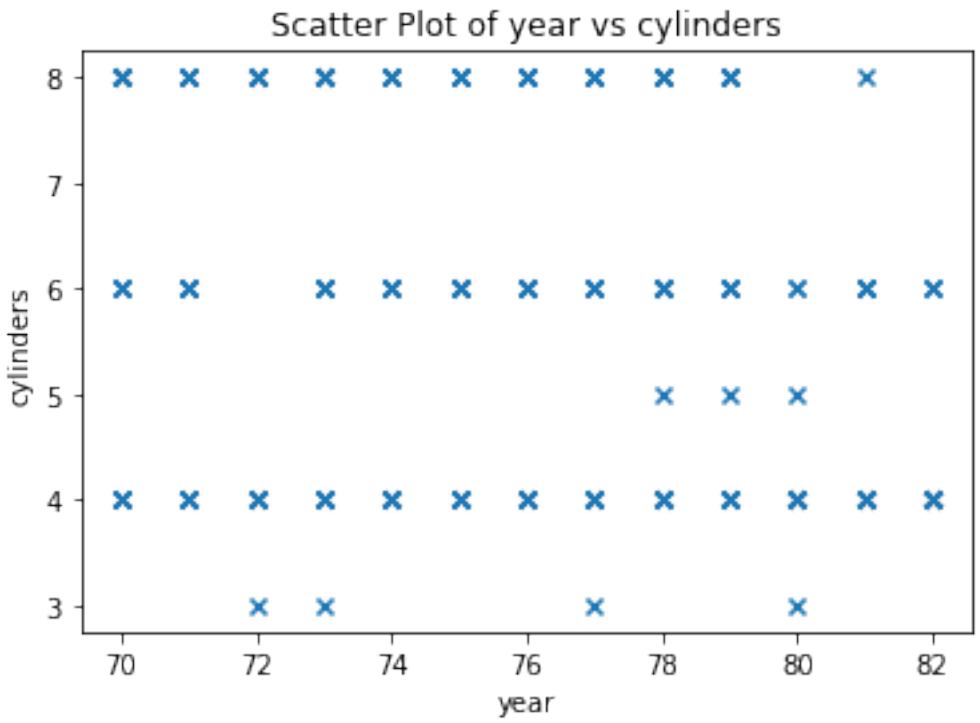

Scatter Plot of acceleration vs horsepower
horsepower
acceleration
225
200
175
150
125
100
75
50
7.5
10.0
12.5
15.0
17.5
20.0
22.5
25.0

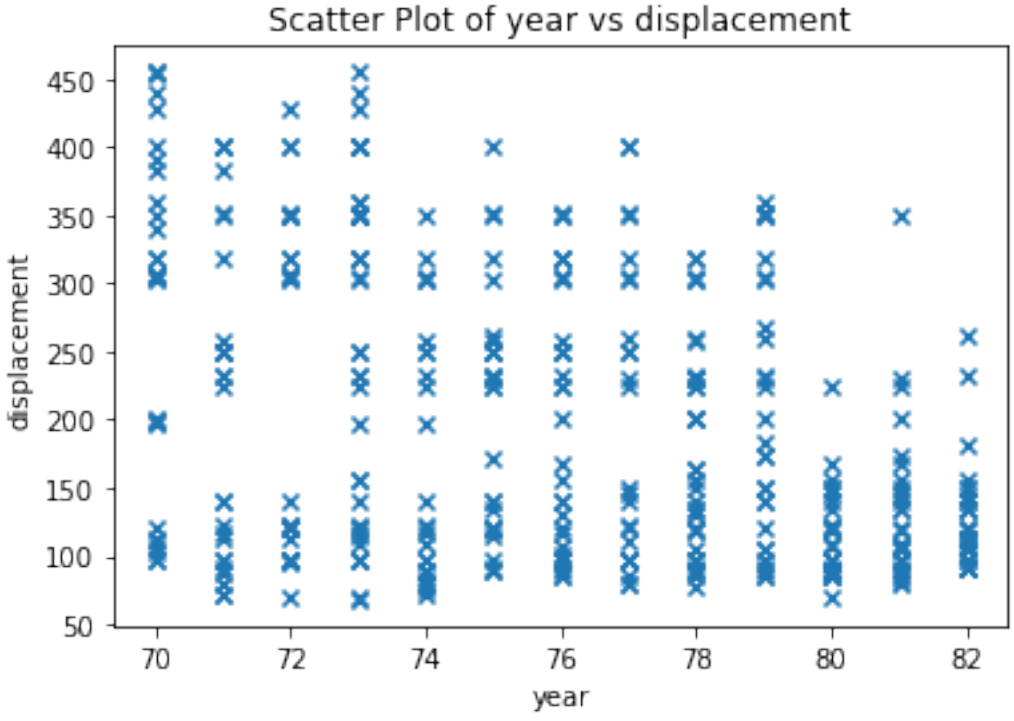

Scatter Plot of acceleration vs year
year
acceleration
82
80
78
76
74
72
70
7.5
10.0
12.5
15.0
17.5
20.0
22.5
25.0

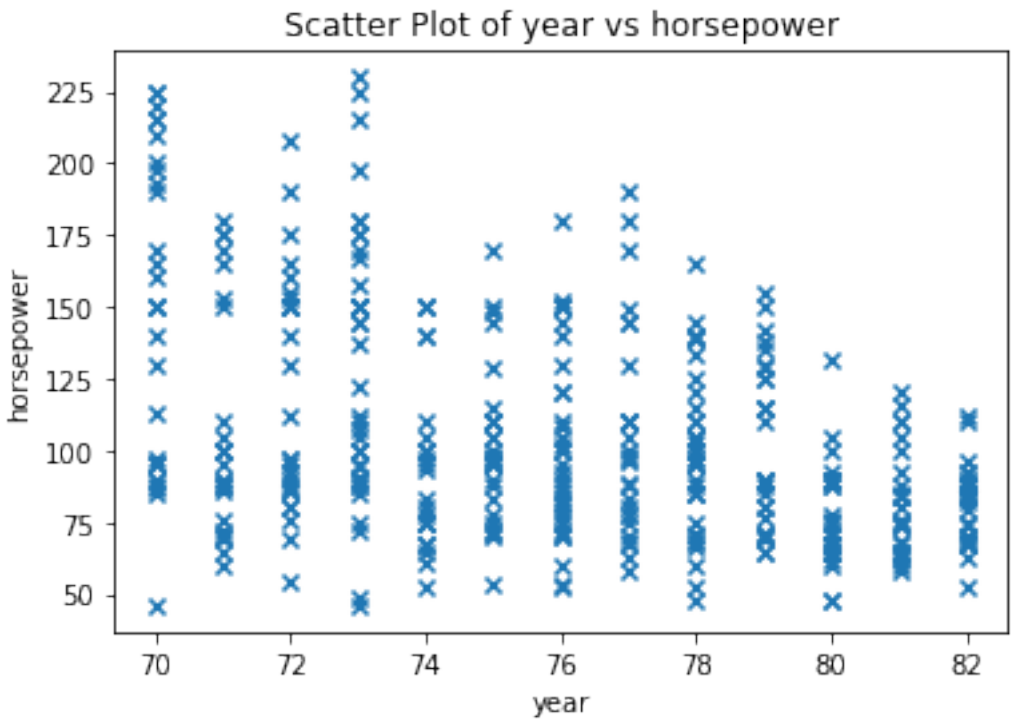

Scatter Plot of acceleration vs origin
origin
acceleration
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1.00
7.5
10.0
12.5
15.0
17.5
20.0
22.5
25.0

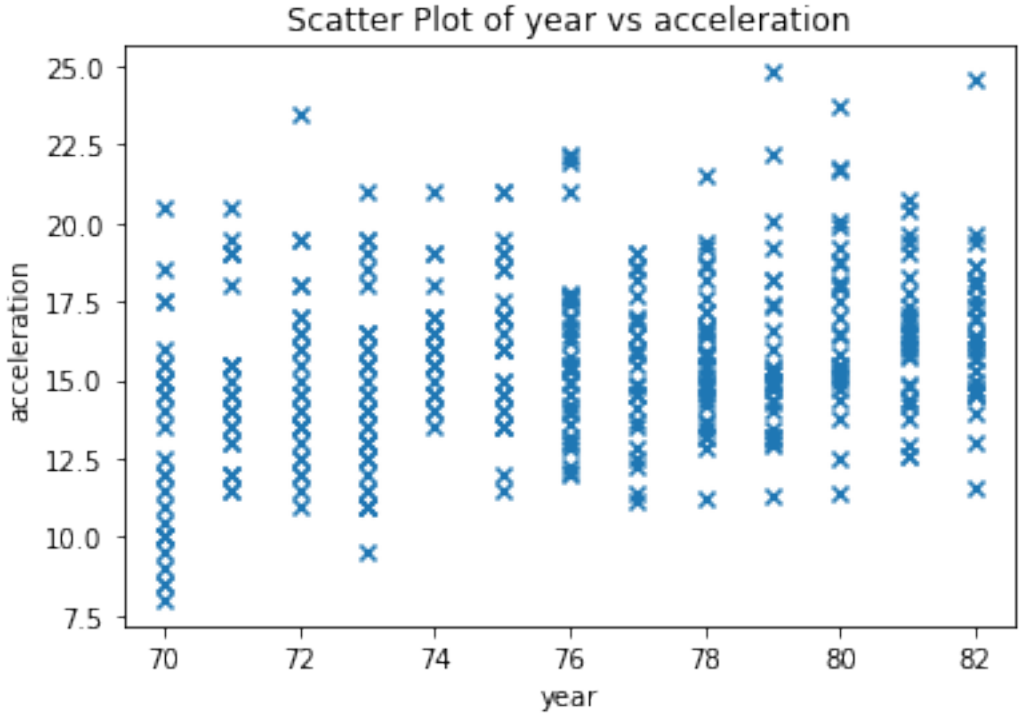

Scatter Plot of year vs mpg
mpg
year
10
15
20
25
30
35
40
45
70
72
74
76
78
80
82

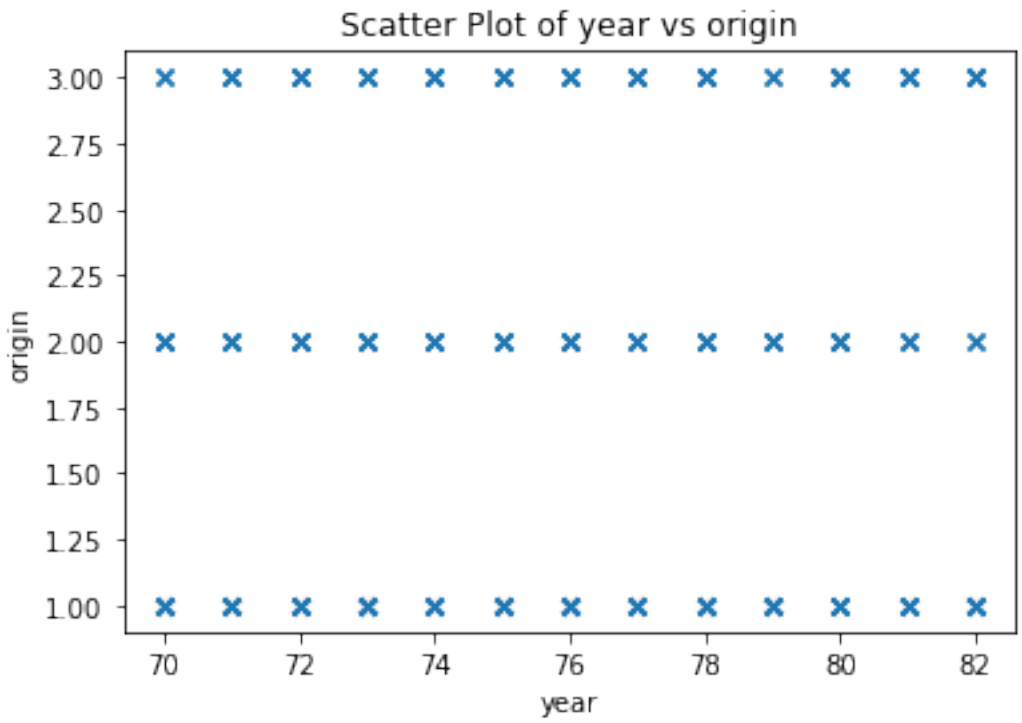

Scatter Plot of year vs weight
weight
year
1500
2000
2500
3000
3500
4000
4500
5000
70
72
74
76
78
80
82

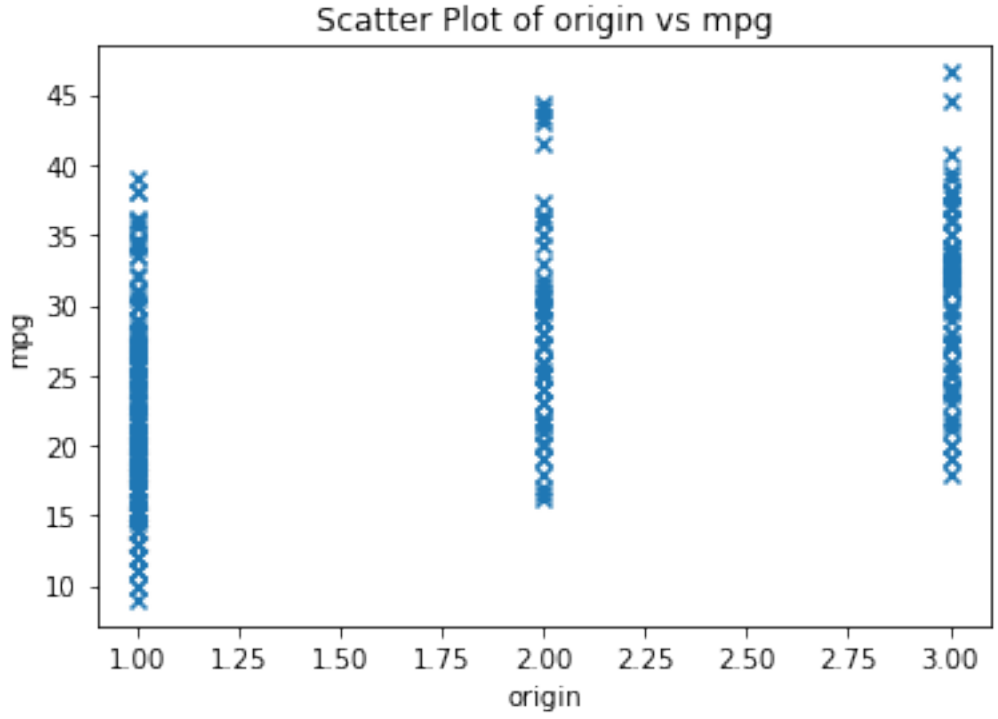

Scatter Plot of year vs cylinders
cylinders
year
8
7
6
5
4
3
70
72
74
76
78
80
82

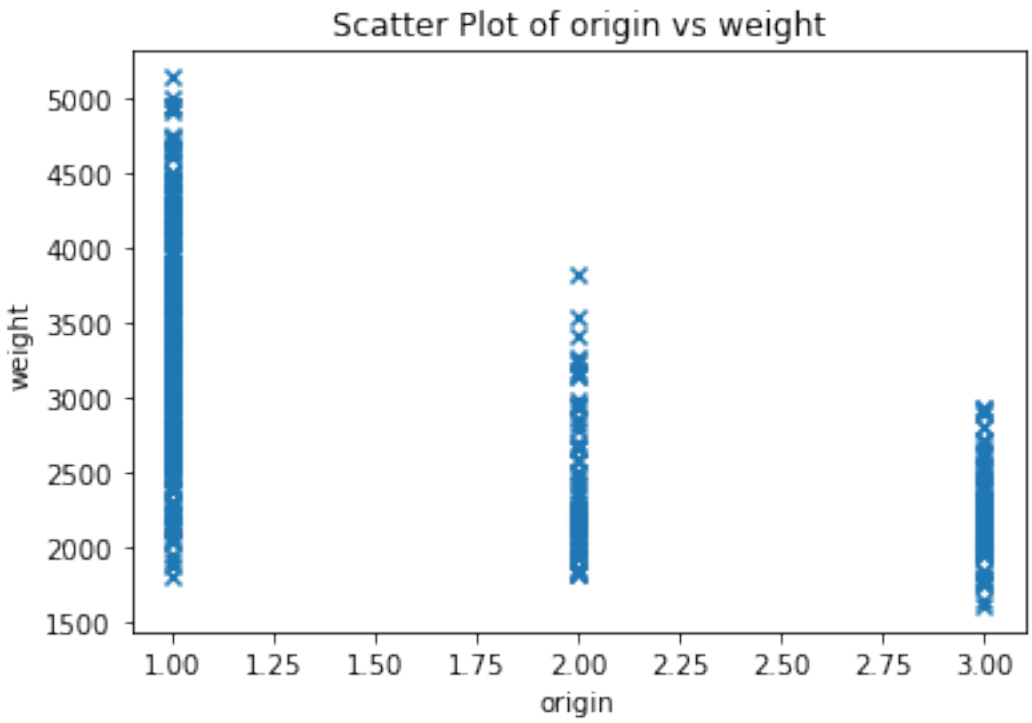

Scatter Plot of year vs displacement
displacement
year
450
400
350
300
250
200
150
100
50
70
72
74
76
78
80
82

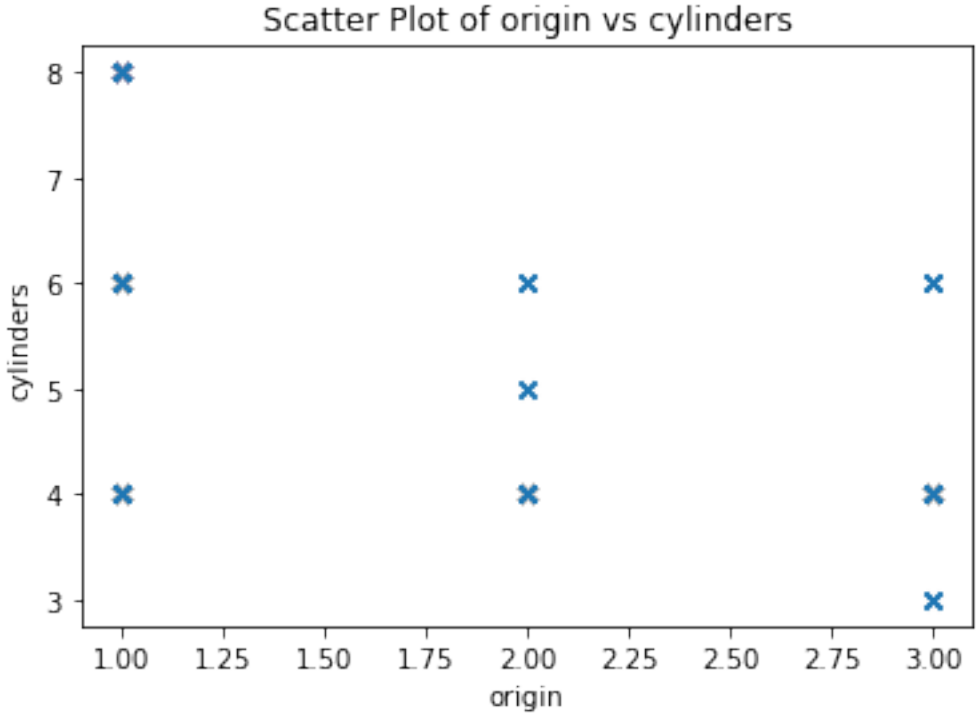

Scatter Plot of year vs horsepower
horsepower
225
200
175
150
125
100
75
50
70
72
74
76
78
80
82
year

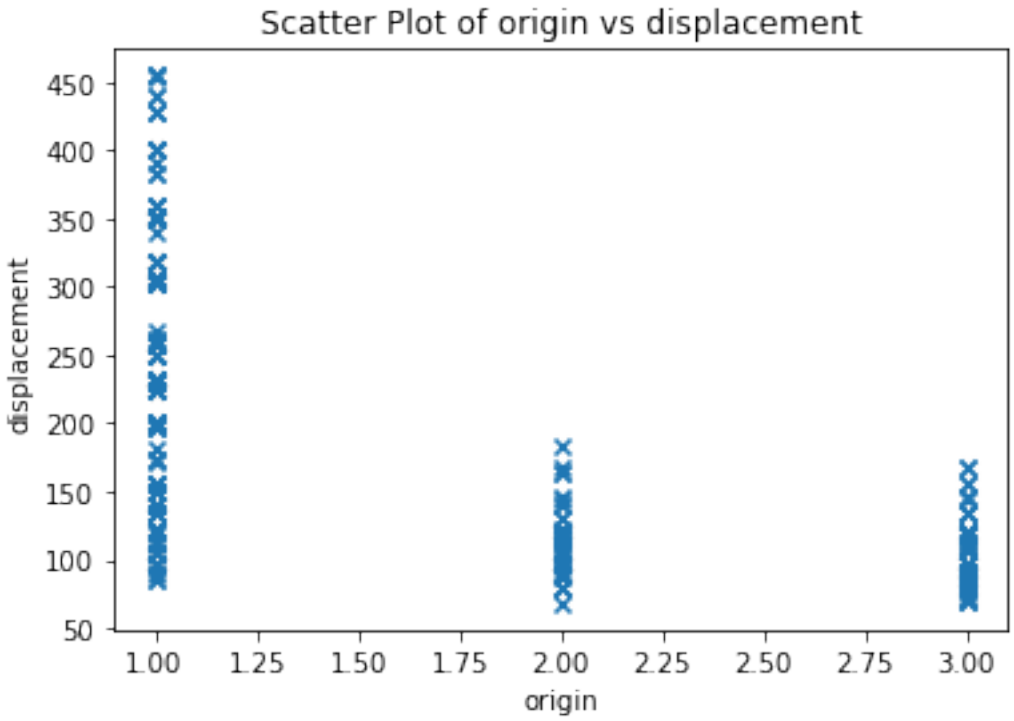

Scatter Plot of year vs acceleration
acceleration
25.0
22.5
20.0
17.5
15.0
12.5
10.0
7.5
70
72
74
76
78
80
82
year

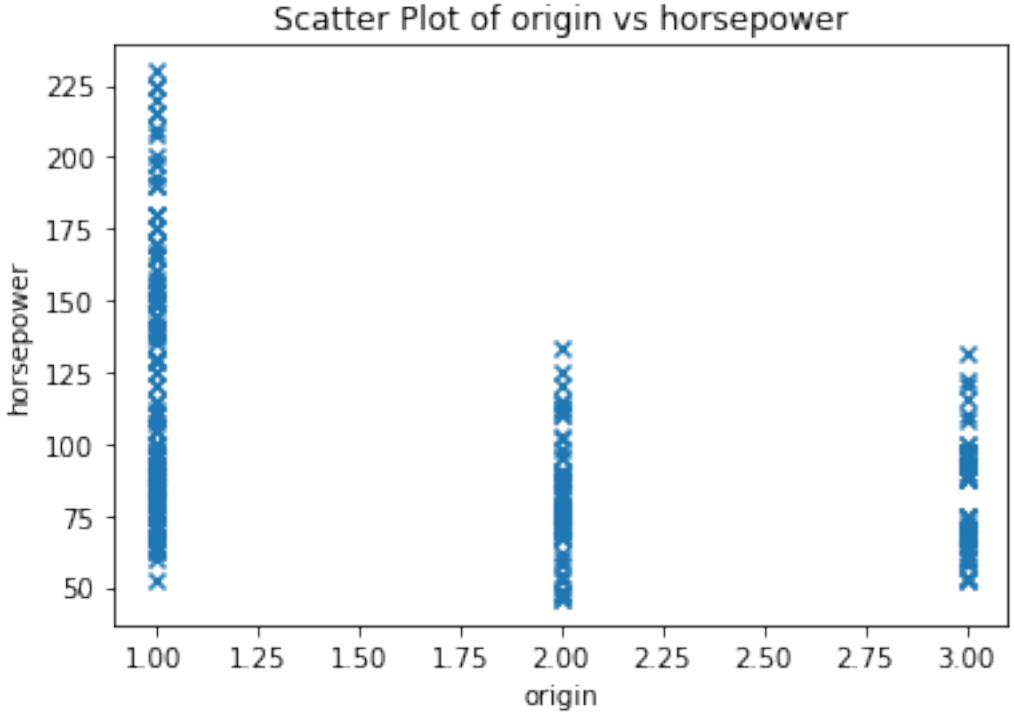

Scatter Plot of year vs origin
origin
year
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1.00
70
72
74
76
78
80
82

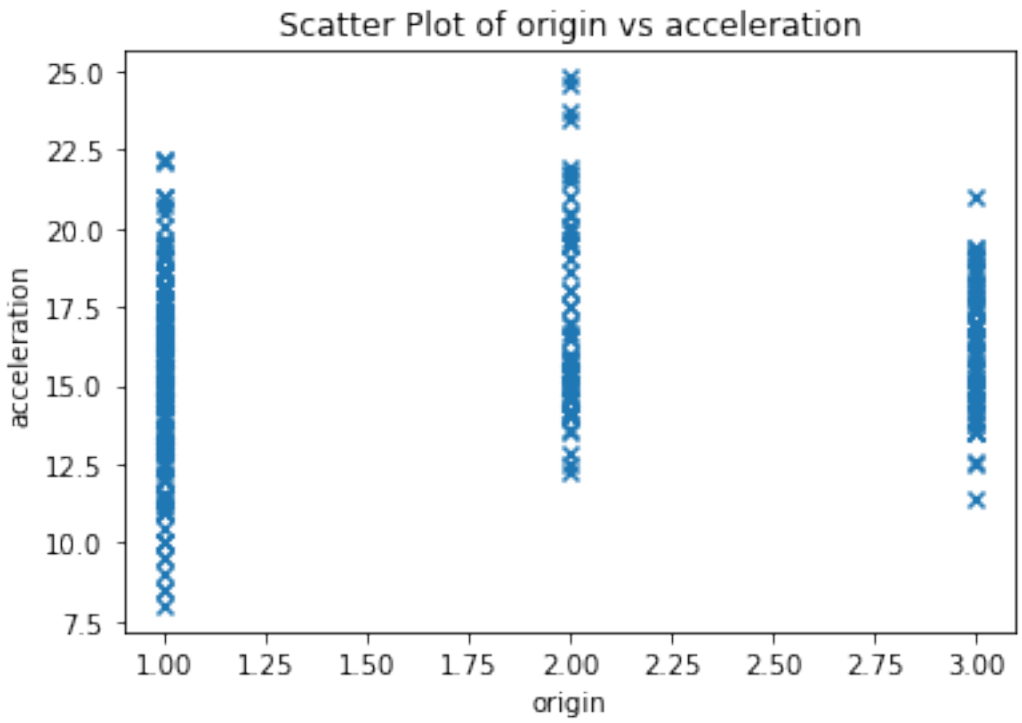

Scatter Plot of origin vs mpg
mpg
origin
45
40
35
30
25
20
15
10
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00

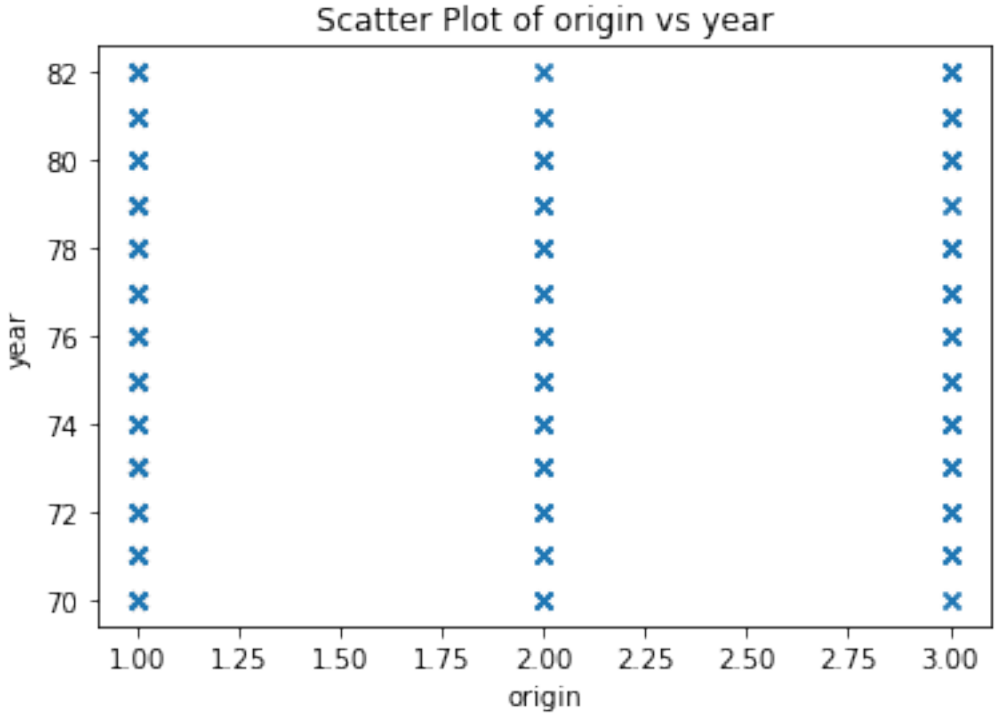

Scatter Plot of origin vs weight
weight
origin
5000
4500
4000
3500
3000
2500
2000
1500
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00


Scatter Plot of origin vs cylinders
cylinders
origin
8
7
6
5
4
3
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00


Scatter Plot of origin vs displacement
displacement
origin
450
400
350
300
250
200
150
100
50
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00


Scatter Plot of origin vs horsepower
horsepower
origin
225
200
175
150
125
100
75
50
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00





We can observe the linear relationship between the variables - displacement v/s weight, displacement v/s horsepower, acceleration v/s horsepower, horsepower v/s weight, mpg v/s weight. (Displacement has positive correlation with horsepower and weight. Acceleration has negative correlation with horsepower and no correlation with mpg. mpg

has negative correlation with weight.) cylinders has non-linear relationship with mpg, displacement, origin, weight.

2. Correlation

```
# Use np corrcoef to observe the correlation between  predictors

print(np.corrcoef(feat4, feat2)) # High correlation
feat4(displacement) & feat2(weight)

print(np.corrcoef(feat7, feat8)) # High anti-correlation feat7(year) &
feat8(origin)
```

```
[[1.        0.9329944]
 [0.9329944 1.       ]]
[[1.         0.18152772]
 [0.18152772 1.        ]]
```

Highly correlated variables are weight v/s displacement and anti-correlated variables are year v/s origin. np.corrcoef(displacement, weight) = 0.933 np.corrcoef(year, origin) = 0.18 It is justified to use the np.corrcoef() function here as it returns Pearson correlation coefficients which is good indicator for calculating correlation between two variables. In our case, the function returns 2x2 matrix where (1,1) value indicates correlation between var1 & var1 and (2,2) value indicates correlation between var2 & var2 and both (1,2) (2,1) value indicates correlation between var1 & var2. The results from np.corrcoef() function are similar to our observations.

3. Linear Regression

```
from sklearn.linear_model import LinearRegression

linear_model = LinearRegression()

y = np.array(data['mpg']).reshape(-1, 1)
x = np.array(data['cylinders']).reshape(-1, 1)
print(x.shape, y.shape)
# Use the fit function and the score function in linear regression
module of sklearn to  fit the data and observe the fit
linear_model.fit(x,y)
r_sq = linear_model.score(x, y)
print("R square (mpg, cylinders): ", r_sq)
```

```
(392, 1) (392, 1)
R square (mpg, cylinders):  0.6046889889441246
```

```
# Fit the linear regression on mpg for the other features: cylinder,
horsepower, year and displacement

y = np.array(data['mpg']).reshape(-1, 1)
x = np.array(data['displacement']).reshape(-1, 1)
linear_model.fit(x,y)
r_sq = linear_model.score(x, y)
```

```
print("R square (mpg, displacement): ", r_sq)

y = np.array(data['mpg']).reshape(-1, 1)
x = np.array(data['horsepower']).reshape(-1, 1)
linear_model.fit(x,y)
r_sq = linear_model.score(x, y)
print("R square (mpg, horsepower): ", r_sq)

y = np.array(data['mpg']).reshape(-1, 1)
x = np.array(data['year']).reshape(-1, 1)
linear_model.fit(x,y)
r_sq = linear_model.score(x, y)
print("R square (mpg, year): ", r_sq)
```

```
R square (mpg, displacement):  0.6482294003193044
R square (mpg, horsepower):  0.6059482578894348
R square (mpg, year):  0.33702781330962295
```

We used R2 value (score) as a measurement to observe the fit and found that predictors (cylinders, displacement, horsepower) have score > 60 which indicates good fit and good relationship of the predictor to the outcome (mpg) whereas predictor (year) have a score=0.33 which is not as good as other variables and does not have a strong relationship with the outcome (mpg). Following values are arranged from good (high R2) to bad (low R2) models: R2 value (mpg, displacement) = 0.6482 R2 value (mpg, horsepower) = 0.6059 R2 value (mpg, cylinders) = 0.6046 R2 value (mpg, year) = 0.337

```
#Multiple Linear Regression
#Create a 2D matrix with each column representing all features
#Look up np.concatenate or np.stack

#Print the predictions of the your fitted model and the weights of
Linear Regression (look at the function get_params())

y = data['mpg']
x = data[['weight', 'cylinders', 'displacement', 'horsepower',
'acceleration', 'year', 'origin']]
linear_model.fit(x,y)
y_predict = linear_model.predict(x)
r_sq = linear_model.score(x, y)
print("Predicted y is: ", y_predict)
print("R square is: ", r_sq)
print("Coeffecients: ", linear_model.coef_)
print(linear_model.get_params())
```

```
Predicted y is:  [15.00095865 13.99929917 15.24044696 15.06190592
14.96717762 10.69562338
 10.6553509  10.69318951 10.21140958 13.11319398 15.29186157
14.14690266
 14.64696189 18.88015369 24.13062472 19.03847473 19.40643664
20.88068005
```

 25.45738302 27.13648841 21.03963303 22.25408365 22.7074409
23.25050717
 20.3005362   7.58306997 8.41074023 8.30037285 6.44696729
26.20815569
 23.39053912 25.73308381 21.46783861 16.23364804 17.52793963
17.90615253
 17.43525382 11.44975337 10.58385513 12.14999889 11.87950268
6.98211816
  8.88321853 6.24768971 19.73241559 23.0454129 17.79193175
18.88084577
 23.26421626 25.13116053 25.49443983 25.22718096 28.75457004
29.66193221
 27.53555535 25.15803406 26.28101808 24.47954013 26.03152794
23.5313529
 24.17758427 11.77971323 11.88636515 12.41807558 13.02233473
14.97586698
 10.21781246 10.59400653 10.80705681 11.4328463 25.38717749
13.63215327
 12.81807391 11.44132342 12.83385802 20.40479902 24.02910196
20.91375517
 25.73501085 23.04318302 26.06966453 24.82972756 24.08232262
27.31430135
 13.56803385 15.72663966 14.80166104 13.75354471 15.40597996
9.11316466
 12.54946701 12.17342553 12.58907167 10.40842679 9.13895663
15.41209672
 19.87451504 19.5611008 21.19768398 21.30747992 20.8714236
28.68457937
  9.25995356 9.44005829 10.26512597 10.8598469 22.12705892
27.10765986
 24.63256448 26.37596114 27.59026112 24.56264013 22.44813226
25.47613389
 14.19310096 12.19610795 28.47323412 26.78124528 23.60437753
21.65696035
 17.7323046 22.91659991 22.85694927 16.14480938 20.3806236
22.23331457
 19.85580312 29.98624525 24.32417673 30.76017502 24.21820401
16.97485381
 18.00971582 17.44005836 13.77224574 10.9854969 11.83497896
10.71579784
 13.01240031 26.72394641 28.19392676 26.1032573 31.92807295
29.82441039
 25.72165832 27.33843339 26.84143838 26.48161957 26.9854206
27.99467507
 20.57949571 19.6449369 20.78200302 22.43502716 12.35099671
13.45751462
 12.4008917 11.98140149 16.6846717 17.01074549 18.20660808
17.44824662
 21.82068634 20.151559   20.84109951 29.28703926 24.20538532
22.89992468

 24.71528484 26.02804741 27.4930715 27.01110186 21.16309705
29.16015946
 21.22722108 24.41349693 23.09011142 22.81674455 24.22188715
32.09569464
 26.68698266 28.64327193 25.06814935 26.73078129 28.29854643
14.77382297
 15.02480599 16.80340145 15.46949369 21.39786251 20.96109442
22.83057359
 22.84206873 28.71669964 27.99652467 29.9270473 32.83840974
18.8770368
 20.51175467 19.12289695 22.59151822 30.61330685 31.13619763
30.17362697
 25.34774195 22.44828555 16.659432   22.20933012 24.71142766
17.65541354
 13.85034576 16.53964233 17.27834413 17.9215907 31.94430733
28.1369806
 31.74200818 27.20228117 32.16218709 17.56835655 16.62475777
16.24043964
 15.26959078 20.69904022 21.06110075 19.76376439 21.07958092
16.55117173
 16.10614612 15.79051915 15.63121352 30.68623095 25.10657282
30.35729138
 24.75754377 29.01707404 28.4340407 32.16059618 29.03548977
26.11364564
 26.21144222 26.59475552 32.07896737 31.132481   33.0293463
32.24874085
 34.10835103 21.59297517 19.6576677 20.39227537 21.28505309
23.21736023
 24.4235579 25.71962878 21.76905783 23.50212832 22.01457196
23.8163016
 20.4903419 22.04686279 21.32122206 20.38330848 22.62637987
17.46441064
 28.96950257 29.27303384 30.66423609 28.34340913 29.61271129
25.69268713
 25.29260528 30.0005767 25.54720814 23.12528937 25.84869065
21.41895534
 31.10502466 31.95932201 23.43759872 25.20586151 25.52290817
23.97740445
 22.7551566 19.81680074 20.35742826 19.67460936 20.03931365
16.88153561
 19.05450791 20.63469186 19.73125191 32.2040919 33.44524117
30.90142267
 26.51946521 23.57987219 20.5695695 25.91921953 22.86655646
29.26841024
 29.72909134 33.45631692 31.00700967 27.02739892 26.16141866
25.61056544
 27.53982752 31.68775723 34.67584782 30.58316132 34.00784694
27.76666717
 26.46762239 25.95267609 23.87820651 31.43773481 29.98650295
31.21749611

```
 31.40718811 32.58673398 33.53957275 26.65283567 33.6244257
32.94922355
 31.49186438 27.0054564  26.20710489 34.95803903 33.50598861
33.73009694
 27.34555857 30.67926574 29.68930984 32.60087719 29.4534759  28.889473
 28.78930104 27.01084268 29.90763811 36.4874435  32.85555202
36.39650161
 34.75856959 35.2653094  34.74098289 34.99513155 30.92848188
31.96064606
 30.15443279 32.27360779 33.60922273 32.91613431 30.83443461
31.44759457
 26.65470916 26.20406605 28.52898099 27.92594196 23.95554276
23.67201668
 26.11473697 24.000706   29.24857498 28.94140762 30.47920275
29.21969186
 30.00182981 29.02005581 27.79675446 34.472563   35.66420933
35.97193622
 32.2458033  32.19737296 34.73003366 34.40042321 34.43949485
35.8117604
 35.86816348 35.7142301  26.80267915 28.43316736 29.77472282
28.35862054
 31.75418253 30.76357903 27.5717334  28.31955391 34.46457181
31.13632611
 29.35024372 28.72892119]
R square is:  0.8214780764810599
Coeffecients:  [-0.00647404 -0.49337632  0.01989564 -0.01695114
0.08057584  0.75077268
  1.4261405 ]
{'copy_X': True, 'fit_intercept': True, 'n_jobs': None, 'normalize':
False, 'positive': False}
```

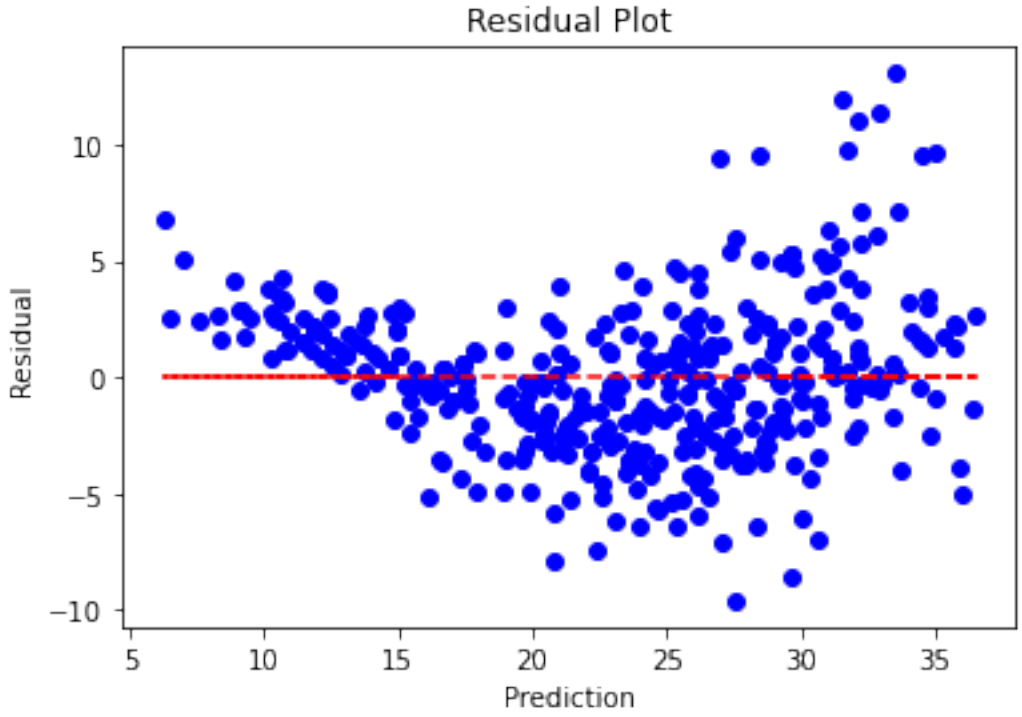
(a) R2 value (mpg, all variables except name) = 0.8215 is high and so this model is considered good fit.

(b) Multiple linear regression model is good fit and shows that combination of predictors(displacement, horsepower, cylinders, year) have strong relationship with response(mpg) as compared to simple linear regression model we have seen in previous part. The sign of the coefficient (i.e. of the estimate) tells us about the direct relationship between the predictor and the response. If the coefficient is positive, then increase in predictor will lead to increase in response. On the other hand, if coefficient is negative, then increase in predictor will lead to decrease in response. In model (mpg, weight) there is negative coefficient, if we increase the value of cylinder variable then the value of mpg decreases. We can confirm this by observing the downward graph.

```
#Make a scatter plot of the residual vs the predictions of your linear
regression model
residuals = y-y_predict
plt.plot(y_predict, residuals, 'o', color='blue')
plt.title("Residual Plot")
plt.xlabel("Prediction")
```

```
plt.ylabel("Residual")

# residual = 0 line corresponds to the estimated regression line.
plt.plot(y_predict, np.zeros(len(y_predict)), '--', color='red')
```

```
[<matplotlib.lines.Line2D at 0x179b0483850>]
```



Yes, residual plot does suggest non linearity in the data. By observing the data, we can say that regression line (residual=0) will not be best fit model but it will be some non-linear plot as data first decreases and after a point increases causing an upward concave.