```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.formula.api as smf
```

```
dataset=pd.read_csv('car performance (1).csv')
dataset
```

```
      mpg  cylinders  displacement  horsepower  weight  acceleration  model  origin        car name
                                                                       year
  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
...   ...        ...           ...         ...     ...           ...    ...     ...             ...
393  27.0          4         140.0          86    2790          15.6     82       1            ford
                                                                                          mustang gl
394  44.0          4          97.0          52    2130          24.6     82       2       vw pickup
395  32.0          4         135.0          84    2295          11.6     82       1           dodge
```

Next steps: Generate code with dataset   View recommended plots   New interactive sheet

```
dataset.isnull().any()
```

```
                  0
         mpg  False
   cylinders  False
displacement  False
  horsepower  False
      weight  False
acceleration  False
  model year  False
      origin  False
    car name  False
```

```
dataset['horsepower']=dataset['horsepower'].replace('?',np.nan)
```

```
dataset['horsepower'].isnull().sum()
```

```
6
```

```
dataset['horsepower']=dataset['horsepower'].astype('float64')
```

```
dataset['horsepower'].fillna((dataset['horsepower'].mean()),inplace=True)
```

```
<ipython-input-9-cc4e4918054d>:1: FutureWarning: A value is trying to be set on a copy of a DataFrame
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate c

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inpl


  dataset['horsepower'].fillna((dataset['horsepower'].mean()),inplace=True)
```

```
dataset.isnull().any()
```

	0
mpg	False
cylinders	False
displacement	False
horsepower	False
weight	False
acceleration	False
model year	False
origin	False
car name	False

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   mpg           398 non-null    float64
 1   cylinders     398 non-null    int64  
 2   displacement  398 non-null    float64
 3   horsepower    398 non-null    float64
 4   weight        398 non-null    int64  
 5   acceleration  398 non-null    float64
 6   model year    398 non-null    int64  
 7   origin        398 non-null    int64  
 8   car name      398 non-null    object 
dtypes: float64(4), int64(4), object(1)
memory usage: 28.1+ KB
```

```
dataset.describe()
```

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	orig
count	398.000000	398.000000	398.000000	398.000000	398.000000	398.000000	398.000000	398.0000
mean	23.514573	5.454774	193.425879	104.469388	2970.424623	15.568090	76.010050	1.5728
std	7.815984	1.701004	104.269838	38.199187	846.841774	2.757689	3.697627	0.8020
min	9.000000	3.000000	68.000000	46.000000	1613.000000	8.000000	70.000000	1.0000
25%	17.500000	4.000000	104.250000	76.000000	2223.750000	13.825000	73.000000	1.0000
50%	23.000000	4.000000	148.500000	95.000000	2803.500000	15.500000	76.000000	1.0000
75%	29.000000	8.000000	262.000000	125.000000	3608.000000	17.175000	79.000000	2.0000
max	46.600000	8.000000	455.000000	230.000000	5140.000000	24.800000	82.000000	3.0000

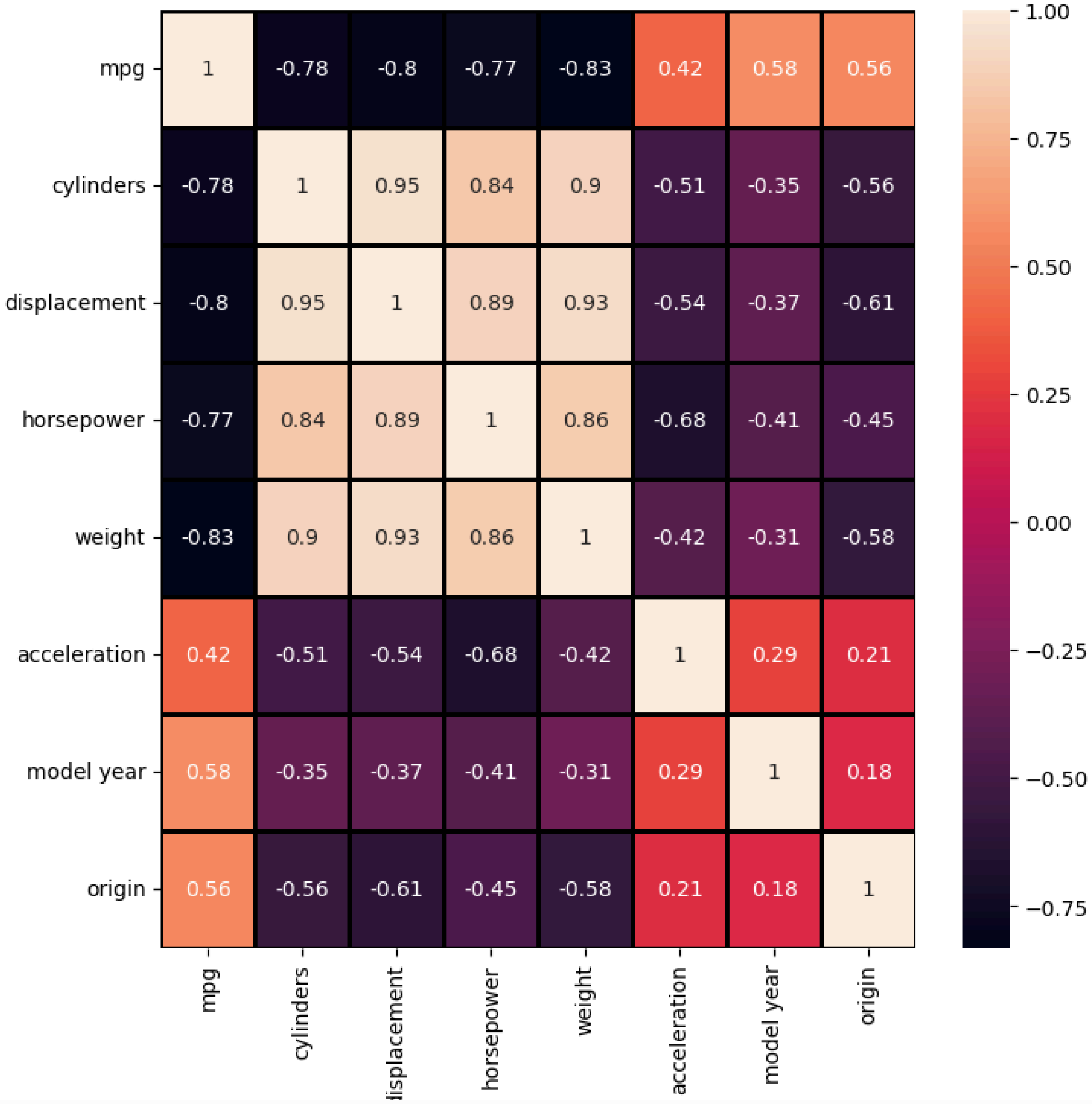
```
dataset=dataset.drop('car name',axis=1)
```

```
corr_table=dataset.corr()
corr_table
```

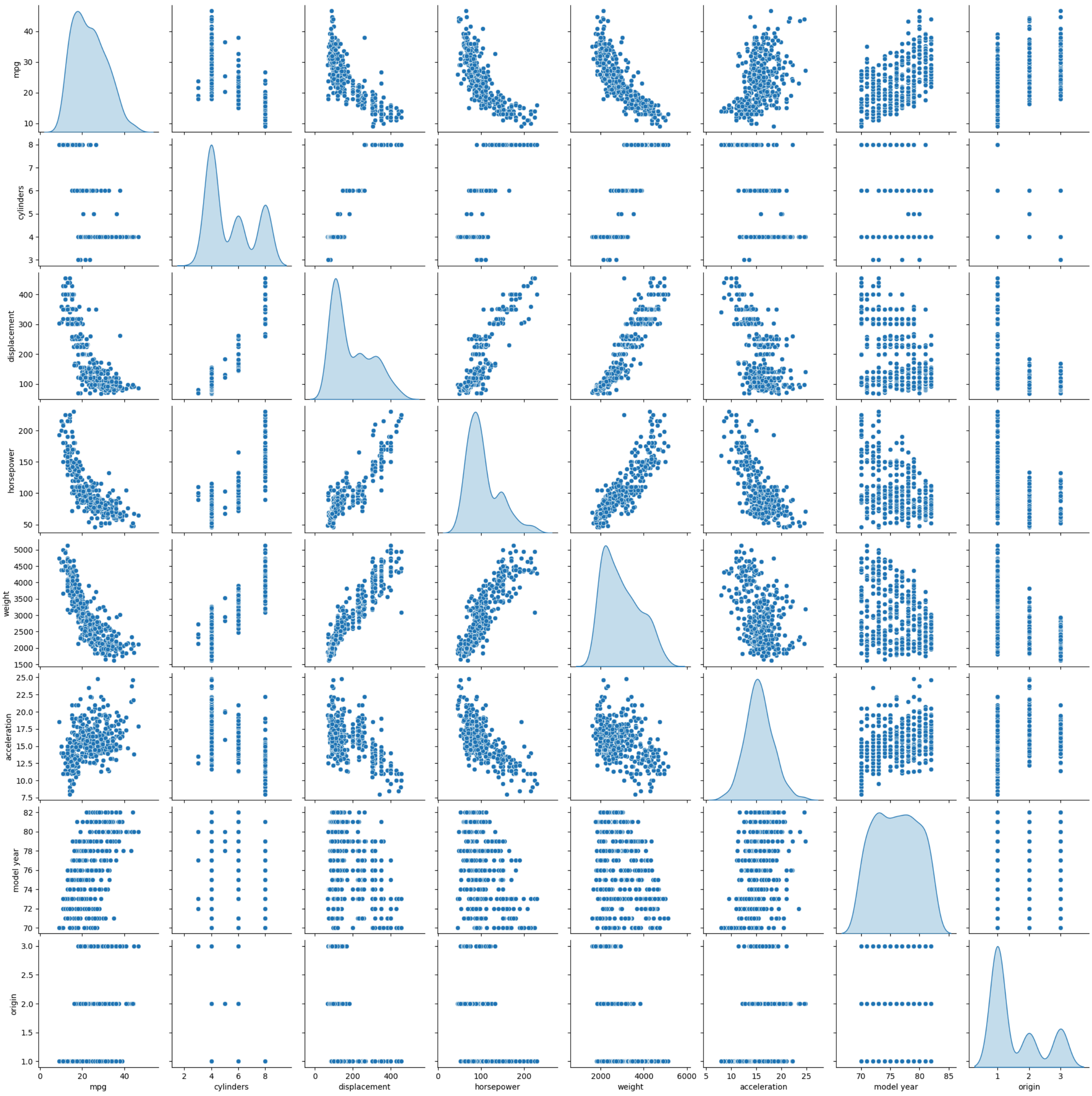
	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origi
mpg	1.000000	-0.775396	-0.804203	-0.771437	-0.831741	0.420289	0.579267	0.56345
cylinders	-0.775396	1.000000	0.950721	0.838939	0.896017	-0.505419	-0.348746	-0.56254
displacement	-0.804203	0.950721	1.000000	0.893646	0.932824	-0.543684	-0.370164	-0.60940
horsepower	-0.771437	0.838939	0.893646	1.000000	0.860574	-0.684259	-0.411651	-0.45366
weight	-0.831741	0.896017	0.932824	0.860574	1.000000	-0.417457	-0.306564	-0.58102
acceleration	0.420289	-0.505419	-0.543684	-0.684259	-0.417457	1.000000	0.288137	0.20587
model year	0.579267	-0.348746	-0.370164	-0.411651	-0.306564	0.288137	1.000000	0.18066
origin	0.563450	-0.562543	-0.609409	-0.453669	-0.581024	0.205873	0.180662	1.00000

Next steps: Generate code with `corr_table` View recommended plots New interactive sheet

```
sns.heatmap(dataset.corr(),annot=True,linecolor ='black', linewidths = 1)#Heatmap is a way to show some sc
fig=plt.gcf()
fig.set_size_inches(8,8)
```

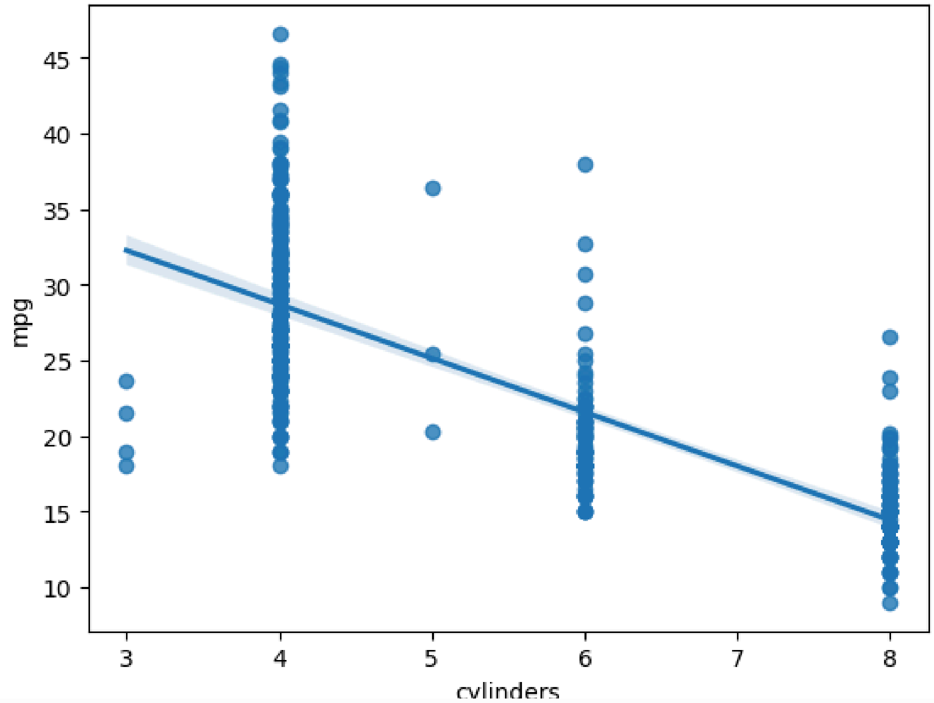


```
sns.pairplot(dataset,diag_kind='kde') #pairplot represents pairwise relation across the entire dataframe.
plt.show()
```



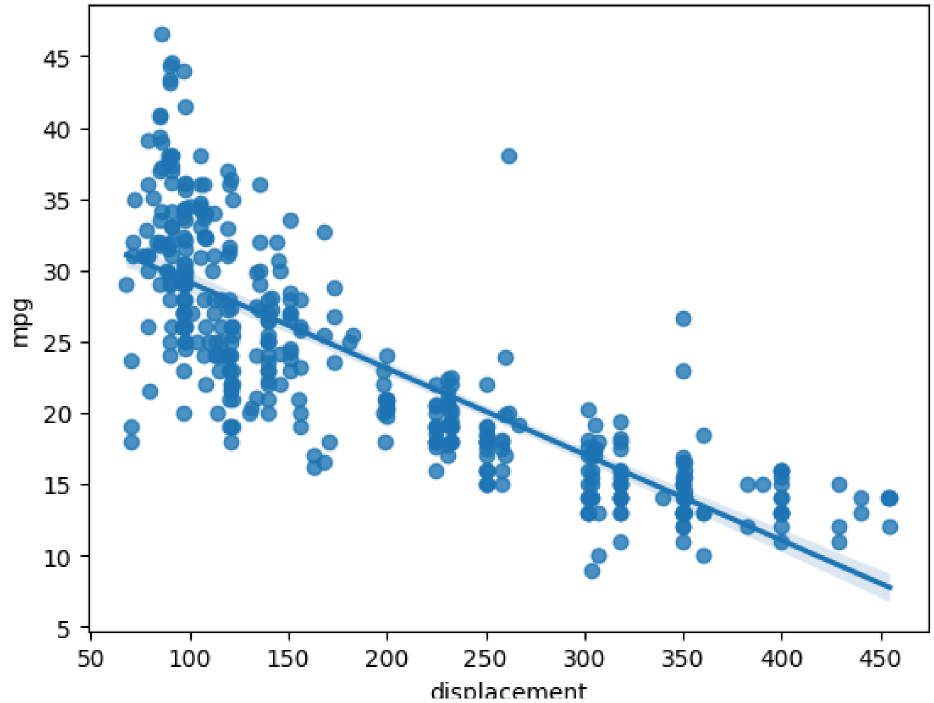
```
sns.regplot(x="cylinders", y="mpg", data=dataset)
```

```
<Axes: xlabel='cylinders', ylabel='mpg'>
```



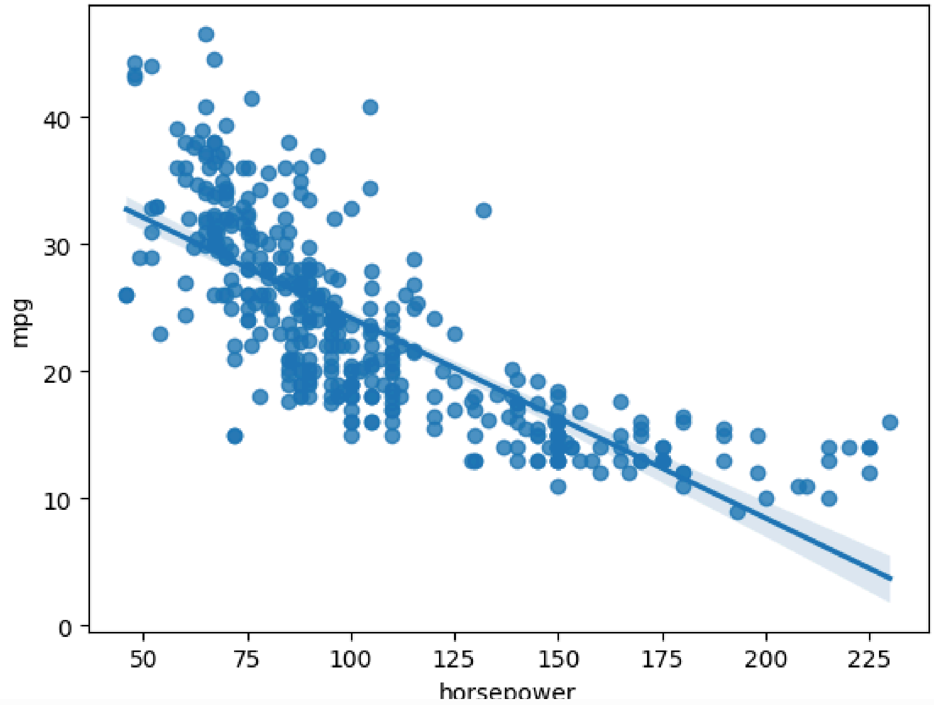
```
sns.regplot(x="displacement", y="mpg", data=dataset)
```

```
<Axes: xlabel='displacement', ylabel='mpg'>
```



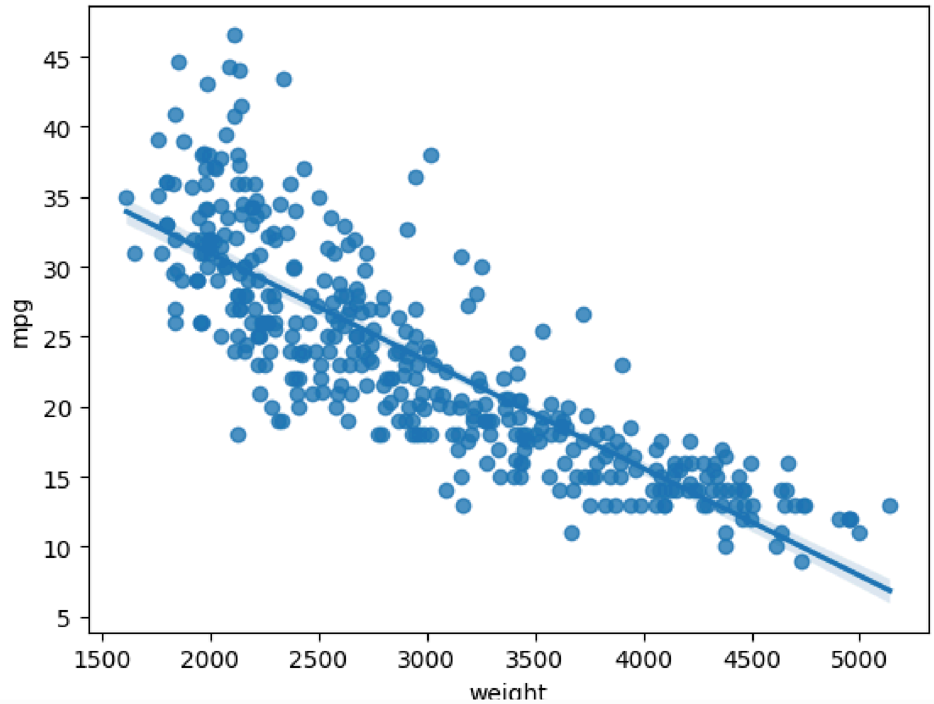
```
sns.regplot(x="horsepower", y="mpg", data=dataset)
```

```
<Axes: xlabel='horsepower', ylabel='mpg'>
```



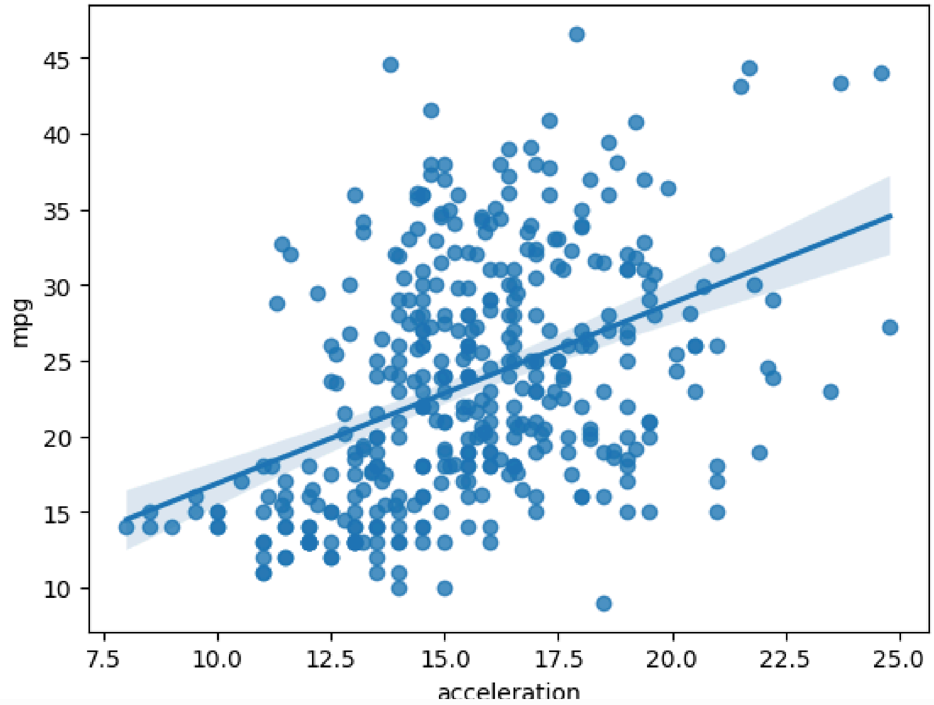
```
sns.regplot(x="weight", y="mpg", data=dataset)
```

```
<Axes: xlabel='weight', ylabel='mpg'>
```



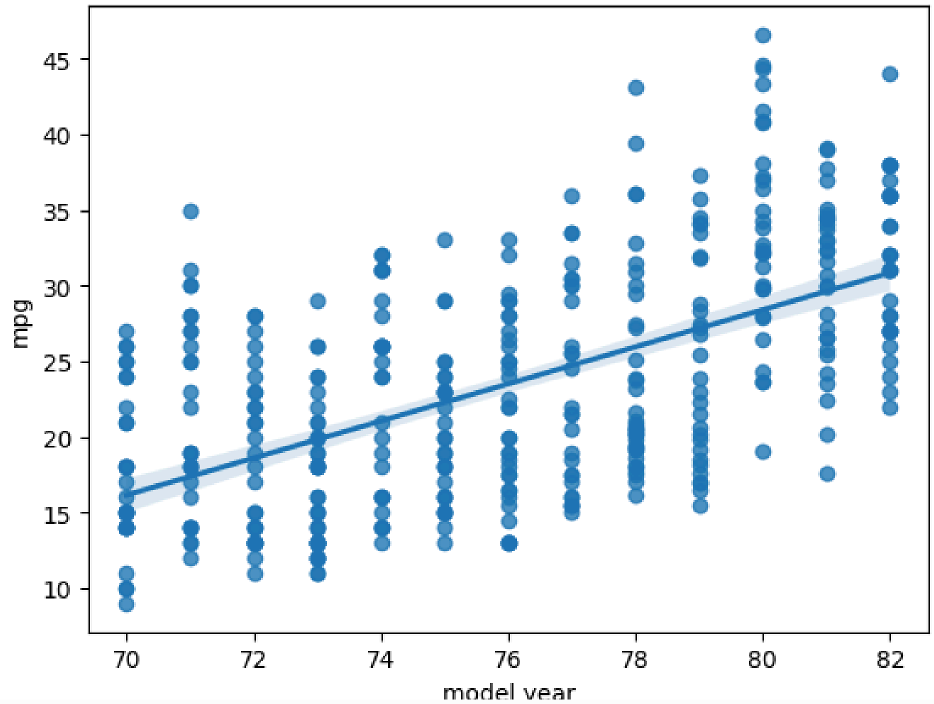
```
sns.regplot(x="acceleration", y="mpg", data=dataset)
```

```
<Axes: xlabel='acceleration', ylabel='mpg'>
```



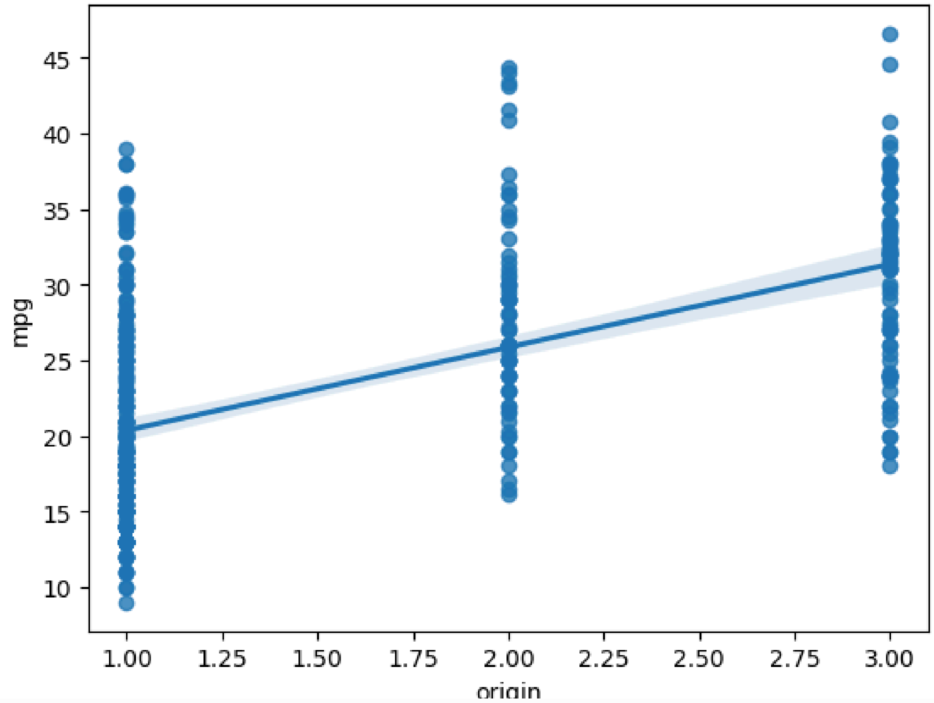
```
sns.regplot(x="model year", y="mpg", data=dataset)
```

```
<Axes: xlabel='model year', ylabel='mpg'>
```



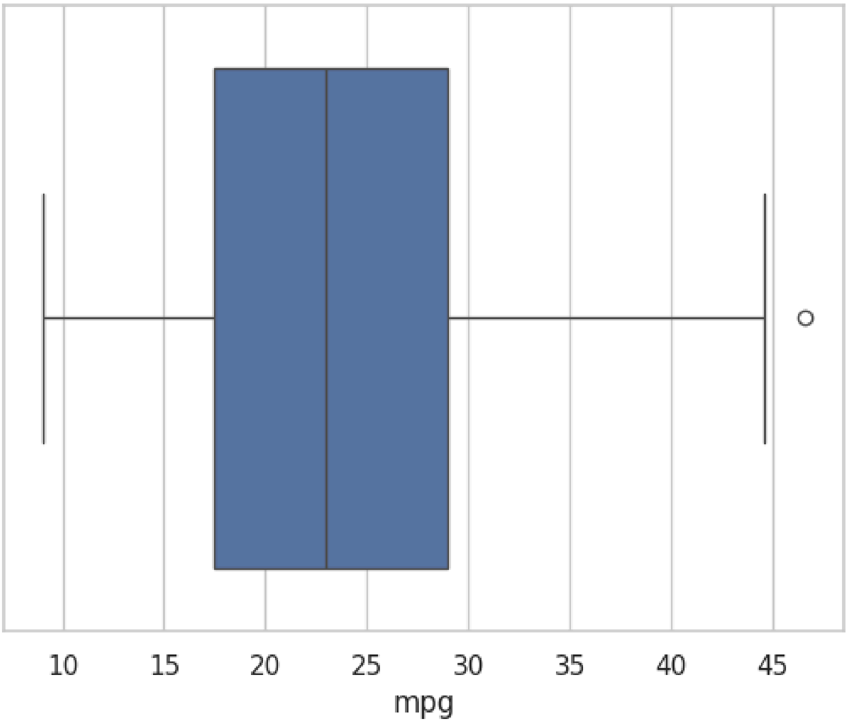
```
sns.regplot(x="origin", y="mpg", data=dataset)
```

```
<Axes: xlabel='origin', ylabel='mpg'>
```



```
sns.set(style="whitegrid")
sns.boxplot(x=dataset["mpg"])
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grour
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
<Axes: xlabel='mpg'>
```



```
from scipy import stats


pearson_coef, p_value = stats.pearsonr(dataset['cylinders'], dataset['mpg'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

```
The Pearson Correlation Coefficient is -0.7753962854205542  with a P-value of P = 4.503992246178154e-8
```

```
pearson_coef, p_value = stats.pearsonr(dataset['displacement'], dataset['mpg'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

```
The Pearson Correlation Coefficient is -0.8042028248058978  with a P-value of P = 1.6558889101929443e-
```

```
pearson_coef, p_value = stats.pearsonr(dataset['horsepower'], dataset['mpg'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

```
The Pearson Correlation Coefficient is -0.7714371350025526  with a P-value of P = 9.255477533167874e-8
```

```
pearson_coef, p_value = stats.pearsonr(dataset['weight'], dataset['mpg'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

```
The Pearson Correlation Coefficient is -0.831740933244335  with a P-value of P = 2.9727995640496354e-1
```

```
pearson_coef, p_value = stats.pearsonr(dataset['acceleration'], dataset['mpg'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

```
The Pearson Correlation Coefficient is 0.4202889121016507  with a P-value of P = 1.823091535078707e-18
```

```
test=smf.ols('mpg~cylinders+displacement+horsepower+weight+acceleration+origin',dataset).fit()
test.summary()
```

```
                         OLS Regression Results
   Dep. Variable:     mpg                  R-squared:    0.717
        Model:        OLS              Adj. R-squared:   0.713
       Method:        Least Squares        F-statistic:  165.5
         Date:        Fri, 04 Oct 2024 Prob (F-statistic): 4.84e-104
         Time:        16:49:07          Log-Likelihood:  -1131.1
No. Observations:     398                        AIC:    2276.
   Df Residuals:      391                        BIC:    2304.
     Df Model:        6
Covariance Type:      nonrobust

                coef    std err   t       P>|t|  [0.025   0.975]
  Intercept     42.7111 2.693    15.861  0.000  37.417   48.005
  cylinders     -0.5256 0.404    -1.302  0.194  -1.320   0.268
displacement    0.0106  0.009    1.133   0.258  -0.008   0.029
 horsepower     -0.0529 0.016    -3.277  0.001  -0.085   -0.021
   weight       -0.0051 0.001    -6.441  0.000  -0.007   -0.004
acceleration    0.0043  0.120    0.036   0.972  -0.232   0.241
   origin       1.4269  0.345    4.136   0.000  0.749    2.105

    Omnibus:      32.659  Durbin-Watson:    0.886
Prob(Omnibus):    0.000   Jarque-Bera (JB): 43.338
      Skew:       0.624      Prob(JB):      3.88e-10
    Kurtosis:     4.028      Cond. No.      3.99e+04


Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.99e+04. This might indicate that there are
```

```
X=dataset[['cylinders','displacement','horsepower','weight','model year','origin']].values
X
```

```
array([[8.000e+00, 3.070e+02, 1.300e+02, 3.504e+03, 7.000e+01, 1.000e+00],
       [8.000e+00, 3.500e+02, 1.650e+02, 3.693e+03, 7.000e+01, 1.000e+00],
       [8.000e+00, 3.180e+02, 1.500e+02, 3.436e+03, 7.000e+01, 1.000e+00],
       ...,
       [4.000e+00, 1.350e+02, 8.400e+01, 2.295e+03, 8.200e+01, 1.000e+00],
       [4.000e+00, 1.200e+02, 7.900e+01, 2.625e+03, 8.200e+01, 1.000e+00],
       [4.000e+00, 1.190e+02, 8.200e+01, 2.720e+03, 8.200e+01, 1.000e+00]])
```

```
y=dataset.iloc[:,0:1].values
y
```

```
        [28. ],
        [31. ]])
```

```
from sklearn.model_selection import train_test_split

X_train,X_test,y_train,y_test=train_test_split(X ,y,test_size=0.1,random_state=0)
```

```
from sklearn.tree import DecisionTreeRegressor
dt=DecisionTreeRegressor(random_state=0,criterion="squared_error")
dt.fit(X_train,y_train)
```

```
▾     DecisionTreeRegressor     i ?

DecisionTreeRegressor(random_state=0)
```

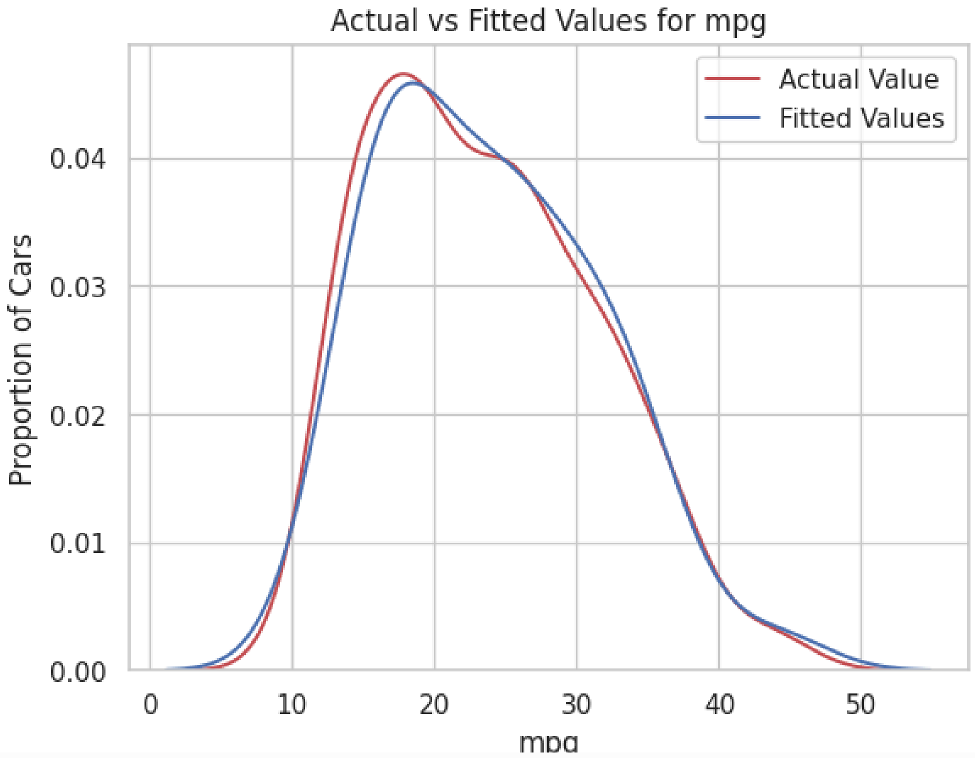
```
import pickle
pickle.dump(dt,open('decision_model.pkl','wb'))
```

```
y_pred=dt.predict(x_test)
y_pred
```

```
array([15. , 26.5, 14. , 19. , 18. , 31. , 31.8, 22. , 15. , 24.2, 36. ,
       31.8, 18. , 26. , 15.5, 29. , 27. , 26. , 16. , 44. , 16. , 23. ,
       25. , 19. , 34.2, 24.2, 29.8, 36. , 34.5, 15. , 19.2, 23.7, 18. ,
       32. , 19.1, 23. , 19.4, 16. , 29. , 12. ])
```

```
ax1 = sns.distplot(dataset['mpg'], hist=False, color="r", label="Actual Value")
sns.distplot(y_pred, hist=False, color="b", label="Fitted Values" , ax=ax1)


plt.title('Actual vs Fitted Values for mpg')
plt.xlabel('mpg')
plt.ylabel('Proportion of Cars')
plt.legend()
plt.show()
plt.close()
```

```
<ipython-input-43-ec6d5054b950>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  ax1 = sns.distplot(dataset['mpg'], hist=False, color="r", label="Actual Value")
<ipython-input-43-ec6d5054b950>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(y_pred, hist=False, color="b", label="Fitted Values" , ax=ax1)
```



```
from sklearn.metrics import r2_score,mean_squared_error

r2_score(y_test,y_pred)
```

```
0.8244088578636598
```

```
mean_squared_error(y_test,y_pred)
np.sqrt(mean_squared_error(y_test,y_pred))
```

```
3.4837838624116744
```

```
from sklearn.linear_model import LinearRegression
mr=LinearRegression()
mr.fit(x_train,y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
y_pred2=mr.predict(x_test)
y_pred2
```

```
array([[13.20818031],
       [24.27993342],
       [11.61339788],
       [20.96914745],
       [17.7247275 ],
       [29.44595217],
       [33.47372984],
       [23.1855594 ],
       [15.045202  ],
       [26.79998444],
       [32.32754229],
       [33.93400668],
       [21.48572281],
       [25.80404696],
       [16.32002867],
       [30.62069212],
       [28.3611479 ],
       [28.68598061],
       [17.66367225],
       [31.02921296],
       [15.54781059],
       [24.61489613],
       [26.90655487],
       [20.51716586],
       [29.66216351],
       [28.48379869],
       [31.00137585],
       [29.9752557 ],
       [29.90123742],
       [18.07465439],
       [20.36226872],
       [31.32907003],
       [20.95979818],
       [32.03796407],
       [23.8731354 ],
       [26.30724058],
       [21.37158555],
       [16.80870416],
       [32.14991802],
       [ 9.27600756]])
```

```
ax1 = sns.distplot(dataset['mpg'], hist=False, color="r", label="Actual Value")
sns.distplot(y_pred2, hist=False, color="b", label="Fitted Values" , ax=ax1)


plt.title('Actual vs Fitted Values for mpg')
plt.xlabel('mpg')
plt.ylabel('Proportion of Cars')
plt.legend()
plt.show()
plt.close()
```

```
<ipython-input-54-4c0275dbc6f8>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  ax1 = sns.distplot(dataset['mpg'], hist=False, color="r", label="Actual Value")
<ipython-input-54-4c0275dbc6f8>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density plots).
```