6-Seaborn (visualization)

September 13, 2019

___ ## Pedram Jahangiry (Fall 2019)

1 Seaborn visualization

Topics to be covered: 1. Distribution Plots * distplot * jointplot * pairplot 2. Categorical plots * barplot * countplot * boxplot 3. Heatmaps
4. Facet grids 5. Regression plots

1.1 Imports

[36]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

[31]:
```
# Seaborn comes with built-in data sets!
sns.get_dataset_names()
```

```
C:\Users\jahan\Anaconda3\lib\site-packages\seaborn\utils.py:376: UserWarning: No
parser was explicitly specified, so I'm using the best available HTML parser for
this system ("lxml"). This usually isn't a problem, but if you run this code on
another system, or in a different virtual environment, it may use a different
parser and behave differently.

The code that caused this warning is on line 376 of the file
C:\Users\jahan\Anaconda3\lib\site-packages\seaborn\utils.py. To get rid of this
warning, pass the additional argument 'features="lxml"' to the BeautifulSoup
constructor.

  gh_list = BeautifulSoup(http)
```

[31]:
```
['anscombe',
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'exercise',
 'flights',
```

```
 'fmri',
 'gammas',
 'iris',
 'mpg',
 'planets',
 'tips',
 'titanic']
```

[4]:
```
mpg = sns.load_dataset('mpg')
```

[5]:
```
mpg.head()
```

[5]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model_year	origin	name
0	18.0	8	307.0	130.0	3504	12.0	70	usa	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	3693	11.5	70	usa	buick skylark 320
2	18.0	8	318.0	150.0	3436	11.0	70	usa	plymouth satellite
3	16.0	8	304.0	150.0	3433	12.0	70	usa	amc rebel sst
4	17.0	8	302.0	140.0	3449	10.5	70	usa	ford torino

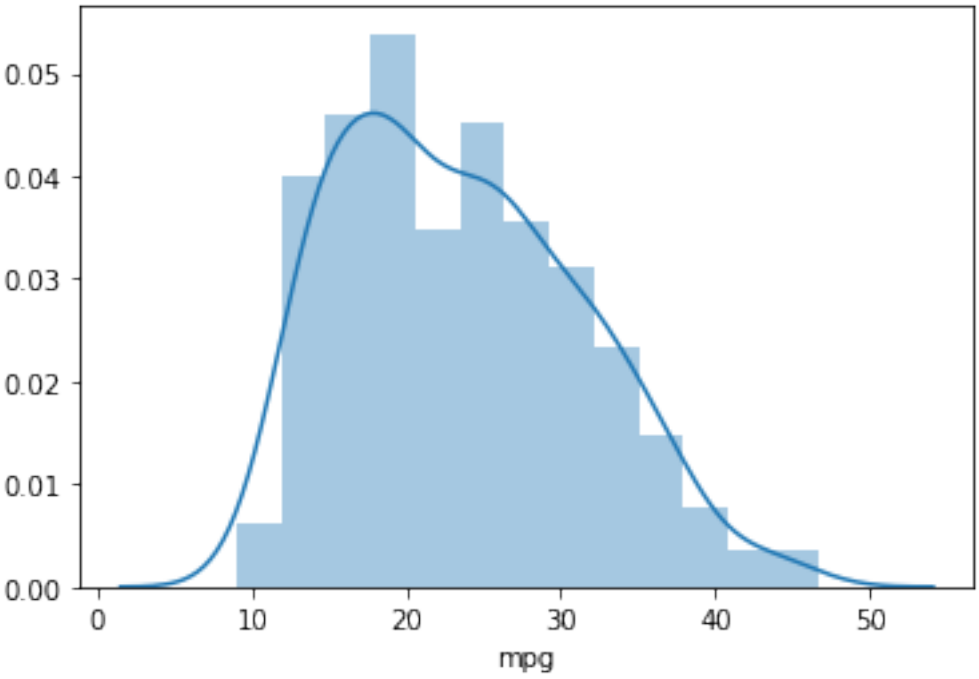
[6]:
```
mpg.info()    # df.info() is equivalent to str(df) in R
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
mpg             398 non-null float64
cylinders       398 non-null int64
displacement    398 non-null float64
horsepower      392 non-null float64
weight          398 non-null int64
acceleration    398 non-null float64
model_year      398 non-null int64
origin          398 non-null object
name            398 non-null object
dtypes: float64(4), int64(3), object(2)
memory usage: 28.1+ KB
```

1.2 1. distribution plots

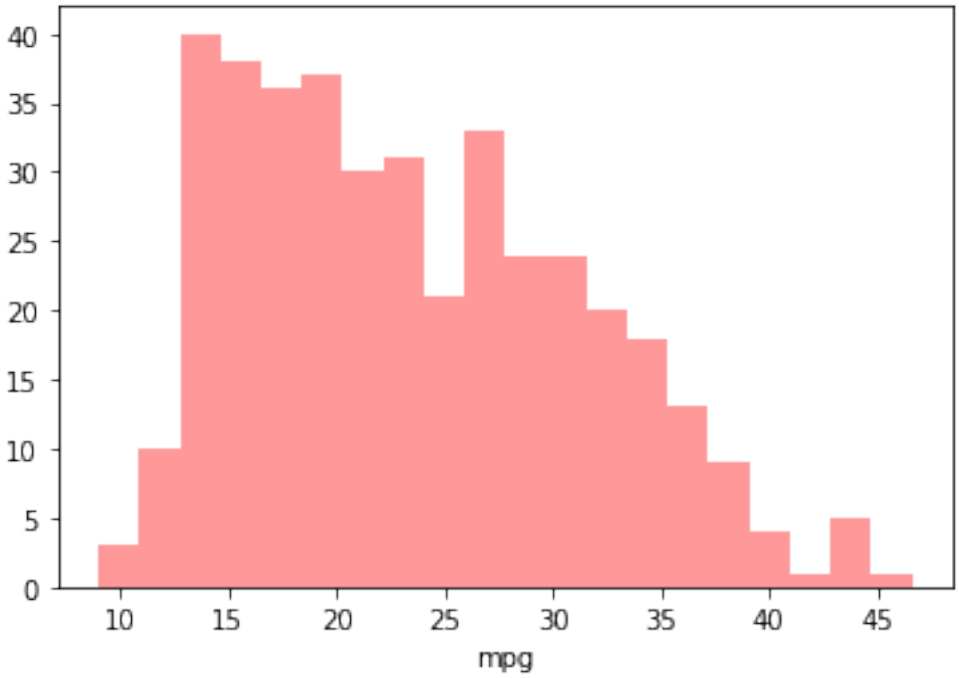
[7]:
```
sns.distplot(mpg['mpg'])
```

[7]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x230aac3d358>
```



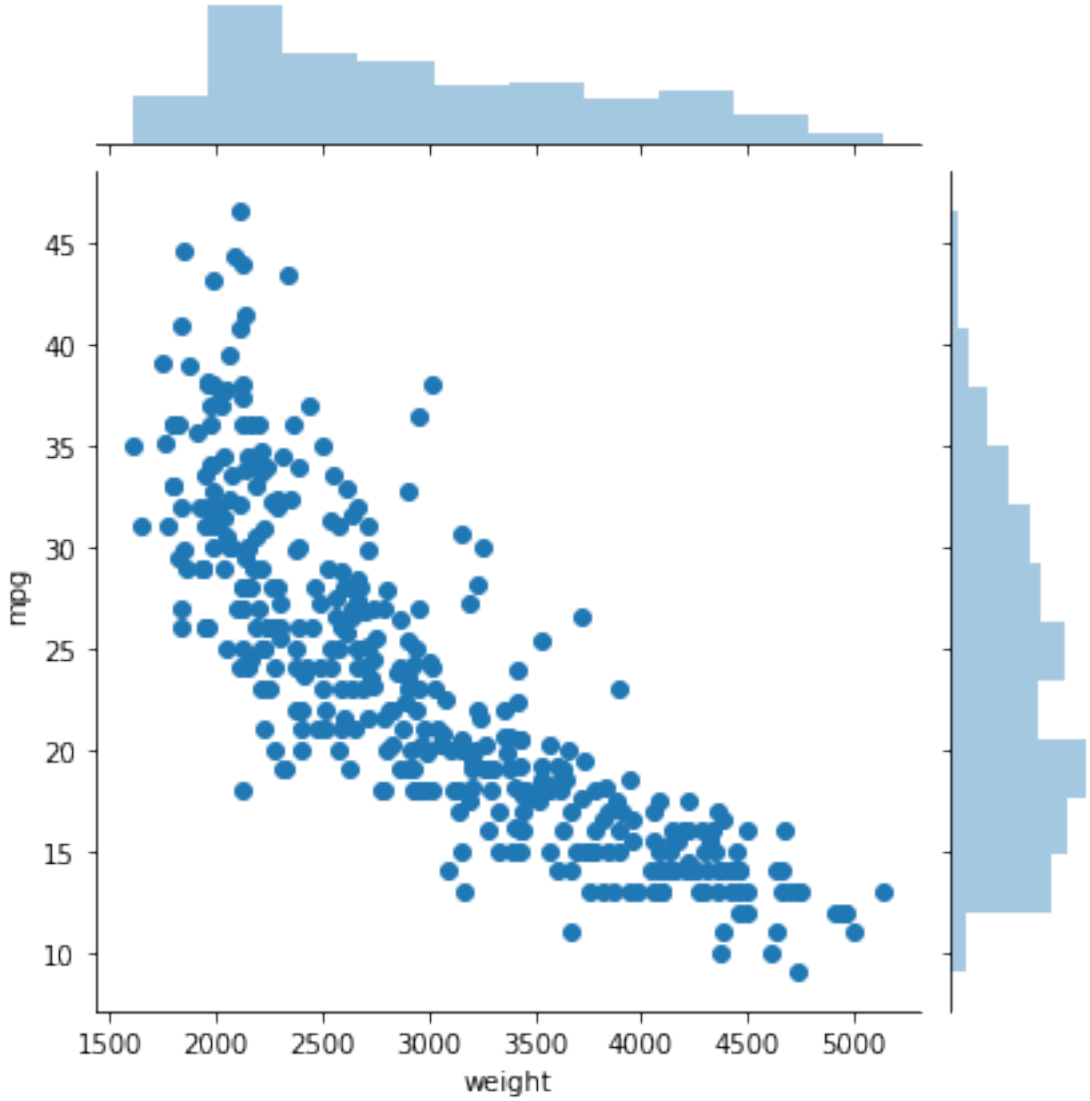
[8]:
```
sns.distplot(mpg['mpg'],kde=False,bins=20, color="red")
```

[8]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x230aaf77dd8>
```



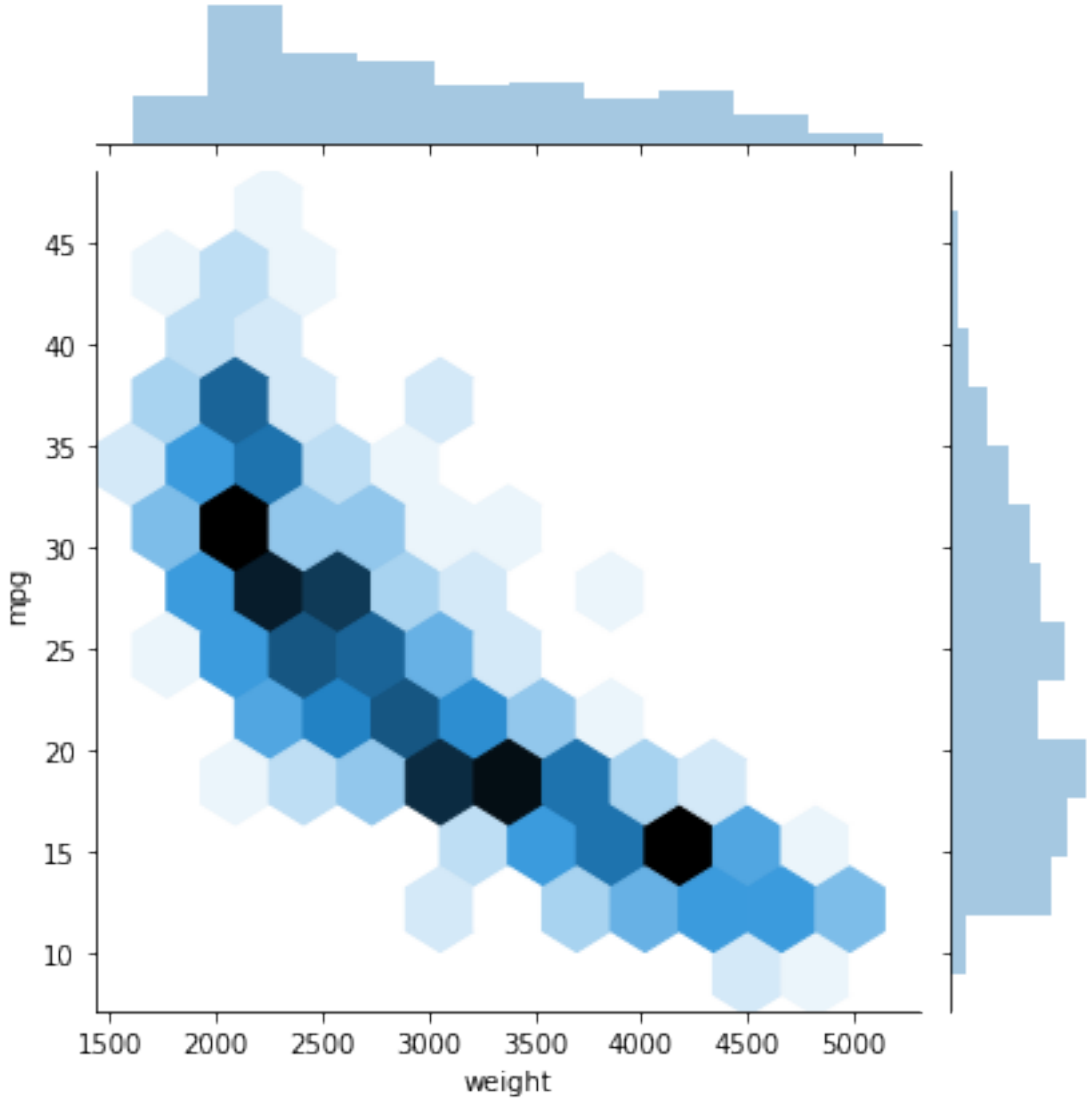
[9]:
```
sns.jointplot(x='weight',y='mpg',data=mpg)
```

[9]:
```
<seaborn.axisgrid.JointGrid at 0x230aafcb320>
```



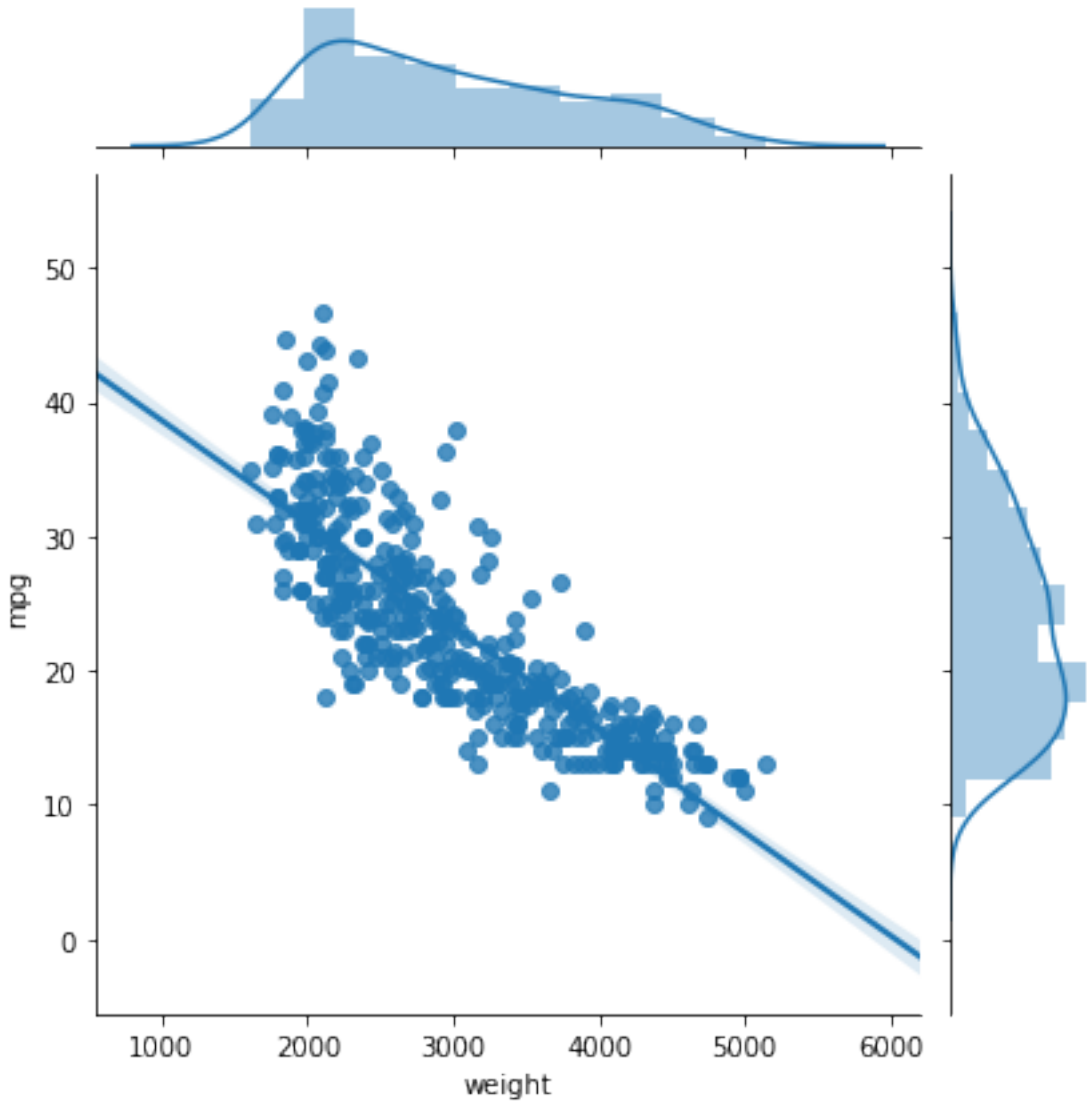
[10]:
```
sns.jointplot(x='weight',y='mpg',data=mpg,kind='hex')
```

[10]:
```
<seaborn.axisgrid.JointGrid at 0x230ab1358d0>
```



[11]:
```
sns.jointplot(x='weight',y='mpg',data=mpg ,kind='reg')
```

[11]:
```
<seaborn.axisgrid.JointGrid at 0x230ab24a0b8>
```



1.3 pairplot

pairplot will plot pairwise relationships across an entire dataframe (for the numerical columns) and supports a color hue argument (for categorical columns).

[12]:
```
mpg.head(5)
```

[12]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	\
0	18.0	8	307.0	130.0	3504	12.0	
1	15.0	8	350.0	165.0	3693	11.5	
2	18.0	8	318.0	150.0	3436	11.0	
3	16.0	8	304.0	150.0	3433	12.0	
4	17.0	8	302.0	140.0	3449	10.5	

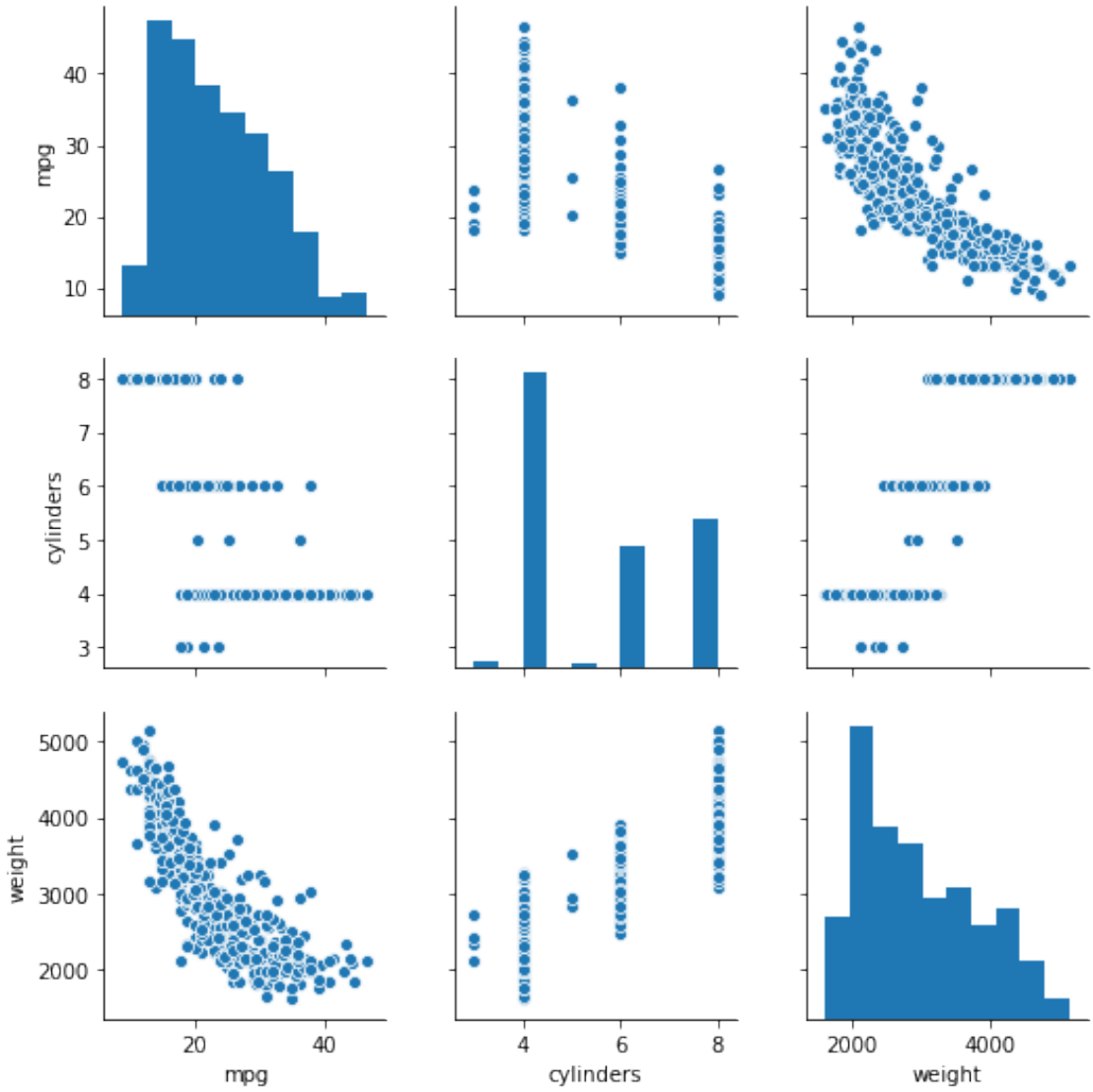
```
   model_year origin                       name
0          70    usa  chevrolet chevelle malibu
1          70    usa          buick skylark 320
2          70    usa         plymouth satellite
3          70    usa              amc rebel sst
4          70    usa                ford torino
```

[13]:
```
df = mpg[['mpg','cylinders','weight']]
df.head(5)
```

[13]:
```
    mpg  cylinders  weight
0  18.0          8    3504
1  15.0          8    3693
2  18.0          8    3436
3  16.0          8    3433
4  17.0          8    3449
```

[14]:
```
sns.pairplot(df) # pairplot look at the joint relationship between numerical␣
↪variables in the dataframe.
```

[14]:
```
<seaborn.axisgrid.PairGrid at 0x230aab57278>
```



[15]:
```
df['cylinders'].value_counts()
```

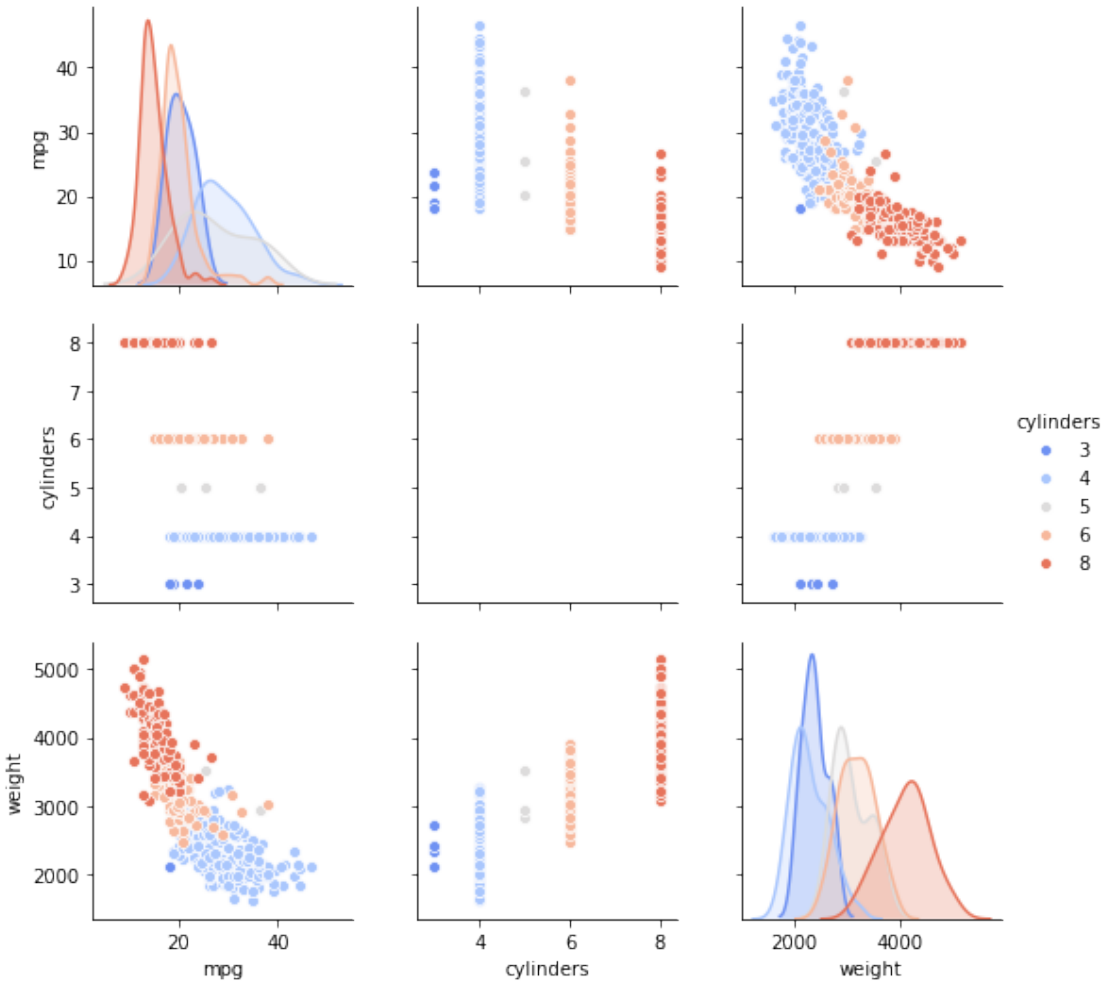
[15]:
```
4    204
8    103
6     84
3      4
5      3
Name: cylinders, dtype: int64
```

[16]:
```
sns.pairplot(df,hue='cylinders',palette='coolwarm')
# hue is used for categorical variables
# palette specifies the colors
```

```
C:\Users\jahan\Anaconda3\lib\site-packages\statsmodels\nonparametric\kde.py:487:
RuntimeWarning: invalid value encountered in true_divide
```

```
  binned = fast_linbin(X, a, b, gridsize) / (delta * nobs)
C:\Users\jahan\Anaconda3\lib\site-
packages\statsmodels\nonparametric\kdetools.py:34: RuntimeWarning: invalid value
encountered in double_scalars
  FAC1 = 2*(np.pi*bw/RANGE)**2
```

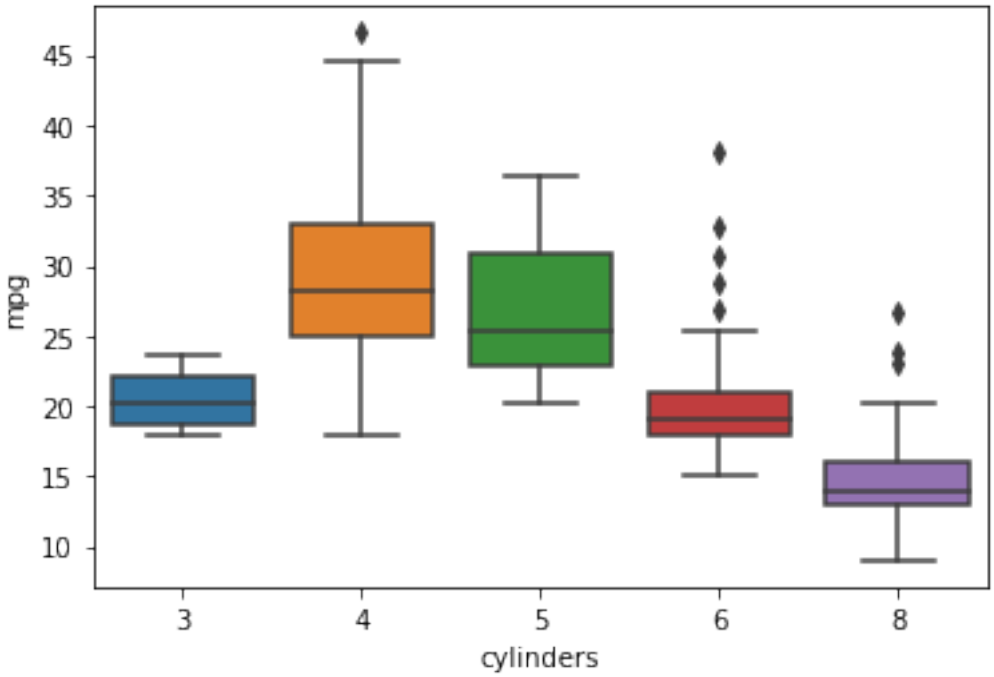
[16]: `<seaborn.axisgrid.PairGrid at 0x230ab8077f0>`



1.4 2. Categorical plots

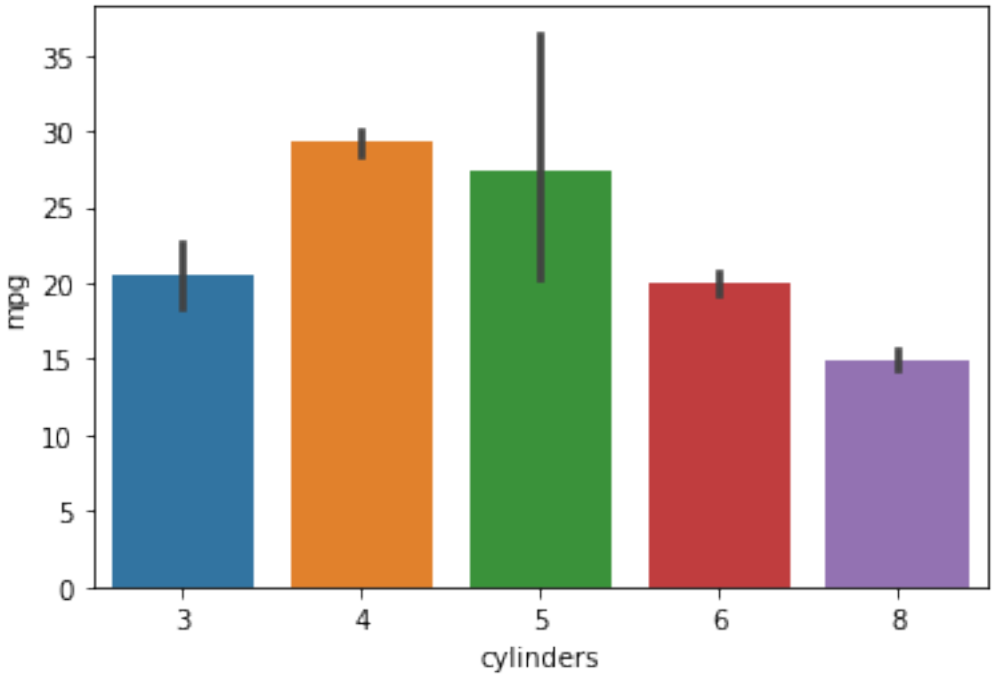
[17]:
```
sns.boxplot(x='cylinders', y='mpg', data=mpg)
```

[17]: `<matplotlib.axes._subplots.AxesSubplot at 0x230abe5d6d8>`



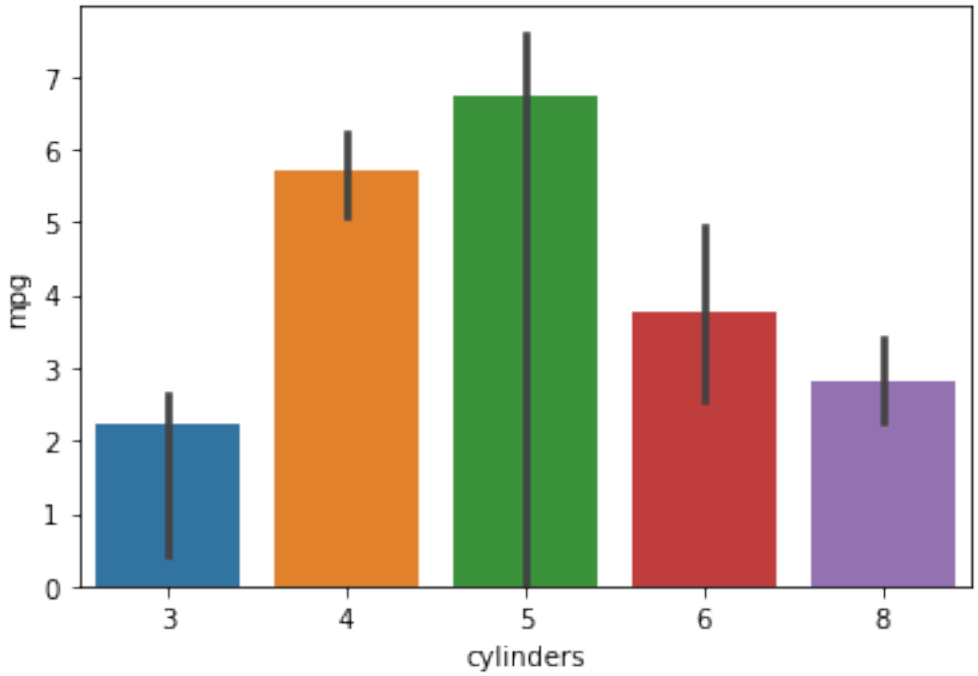
[18]:
```
sns.barplot(x='cylinders', y='mpg', data=mpg)
# barplot is a general plot that allows you to aggregate the categorical data␣
 ↪based off some function, by default the mean.
```

[18]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x230abf66160>
```



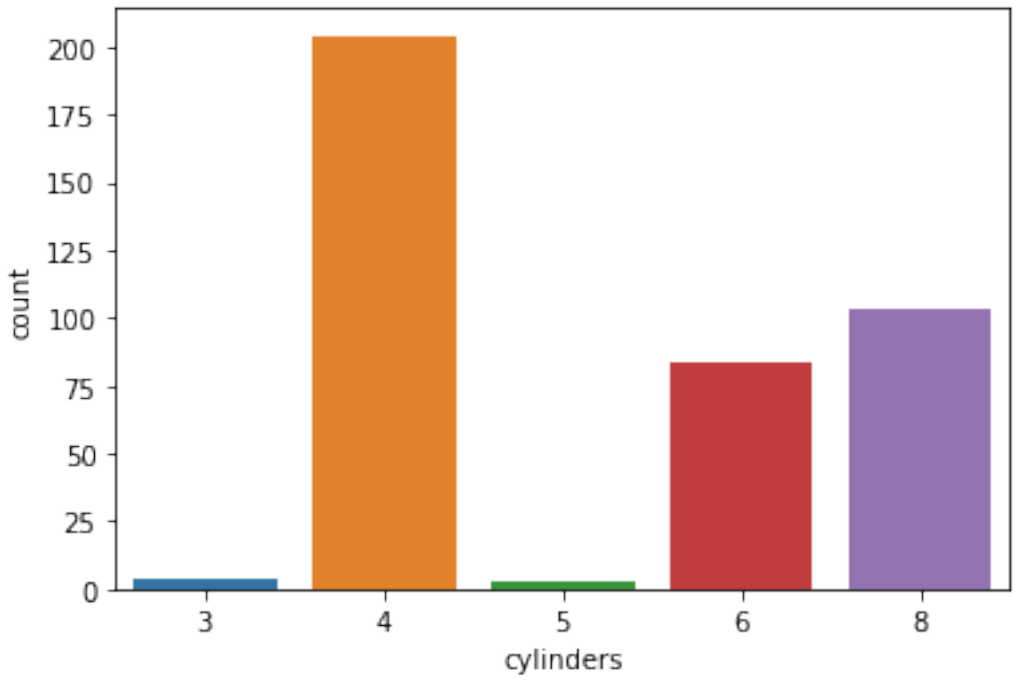
[19]:
```
import numpy as np
sns.barplot(x='cylinders', y='mpg', data=mpg, estimator=np.std)
# barplot is a general plot that allows you to aggregate the categorical data␣
 ↪based off some function, by default the mean.
```

[19]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x230abfc9518>
```



[20]:
```
sns.countplot(x='cylinders',data=mpg)
```

[20]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x230ad181828>
```



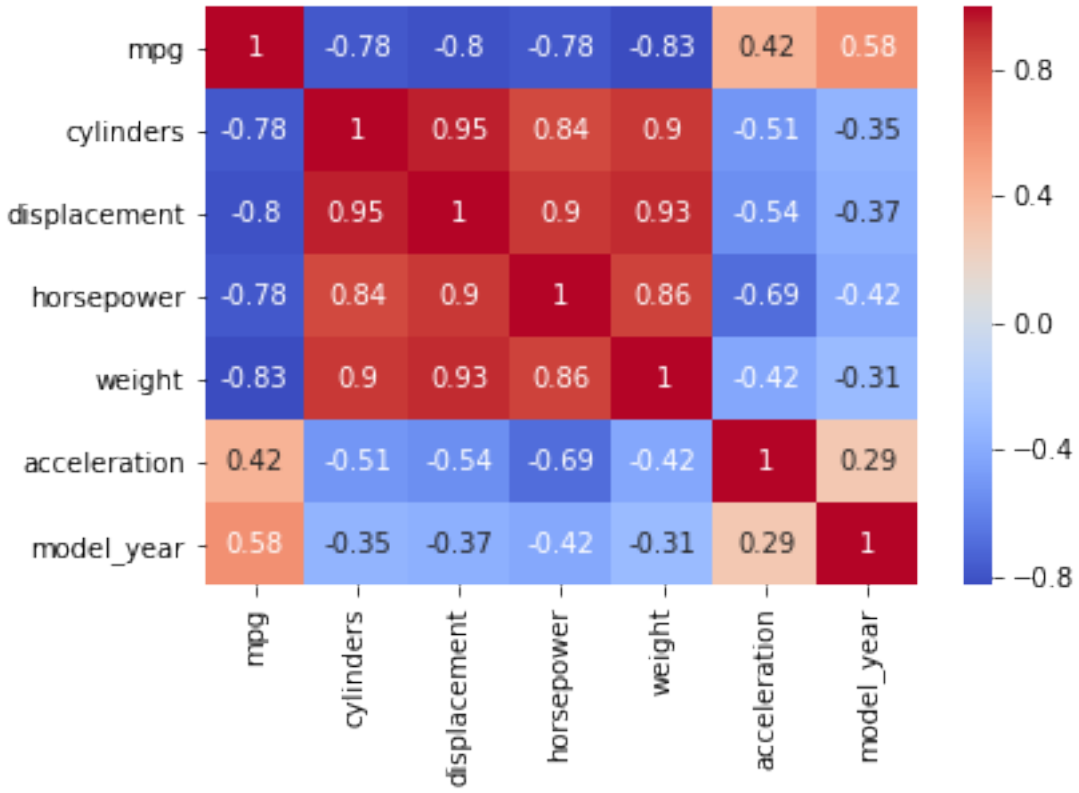
1.5 3. Heatmap

[21]:
```
mpg.corr()
```

[21]:
```
                    mpg  cylinders  displacement  horsepower    weight  \
mpg            1.000000  -0.775396     -0.804203   -0.778427 -0.831741
cylinders     -0.775396   1.000000      0.950721    0.842983  0.896017
displacement  -0.804203   0.950721      1.000000    0.897257  0.932824
horsepower    -0.778427   0.842983      0.897257    1.000000  0.864538
weight        -0.831741   0.896017      0.932824    0.864538  1.000000
acceleration   0.420289  -0.505419     -0.543684   -0.689196 -0.417457
model_year     0.579267  -0.348746     -0.370164   -0.416361 -0.306564

              acceleration  model_year
mpg               0.420289    0.579267
cylinders        -0.505419   -0.348746
displacement     -0.543684   -0.370164
horsepower       -0.689196   -0.416361
weight           -0.417457   -0.306564
acceleration      1.000000    0.288137
model_year        0.288137    1.000000
```

[22]:
```
sns.heatmap(mpg.corr(), cmap='coolwarm',annot=True)
```

[22]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x230ad21d160>
```



1.6 4. Facet grids

[23]:
```
mpg.head()
```

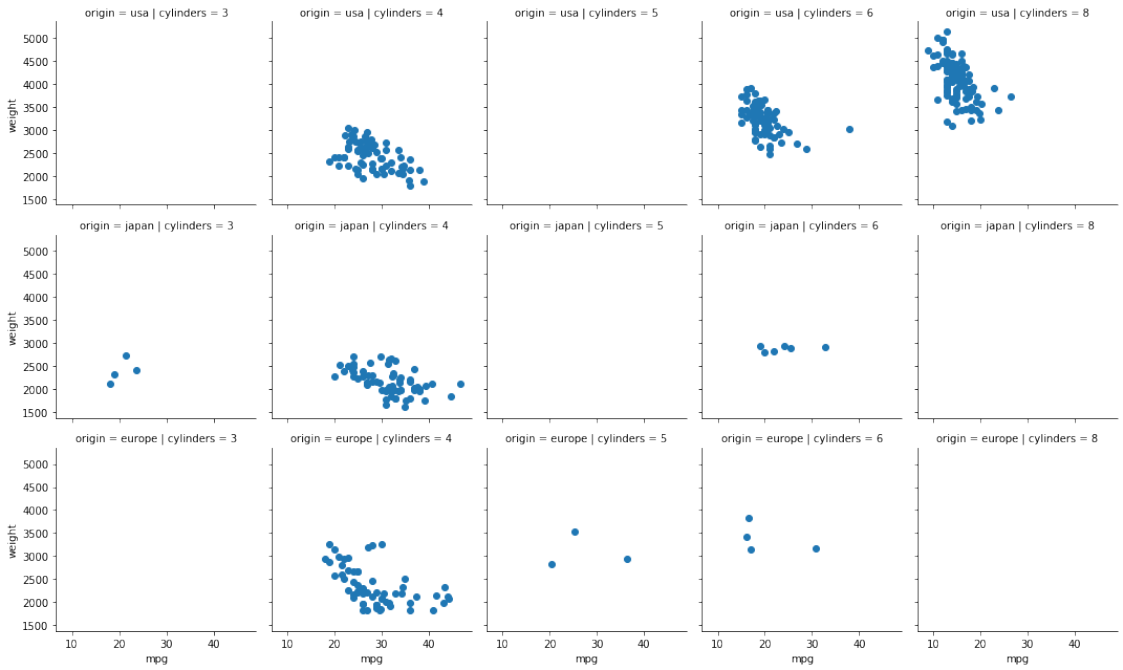
[23]:
```
    mpg  cylinders  displacement  horsepower  weight  acceleration  \
0  18.0          8         307.0       130.0    3504          12.0
1  15.0          8         350.0       165.0    3693          11.5
2  18.0          8         318.0       150.0    3436          11.0
3  16.0          8         304.0       150.0    3433          12.0
4  17.0          8         302.0       140.0    3449          10.5

   model_year origin                       name
0          70    usa  chevrolet chevelle malibu
1          70    usa          buick skylark 320
2          70    usa         plymouth satellite
3          70    usa              amc rebel sst
4          70    usa                ford torino
```

[24]:
```
mpg['origin'].value_counts()
```

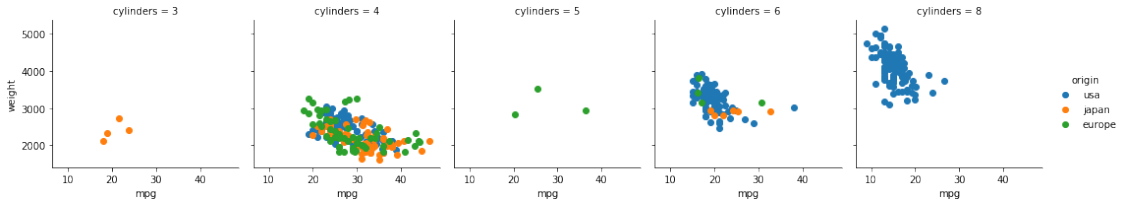
[24]:
```
usa       249
japan      79
```

```
europe     70
Name: origin, dtype: int64
```

[25]:
```
g = sns.FacetGrid(mpg, col="cylinders", row='origin')
g = g.map(plt.scatter, "mpg", "weight")
```



[26]:
```
g = sns.FacetGrid(mpg, col="cylinders", hue='origin')
g = g.map(plt.scatter, "mpg", "weight").add_legend()
```



1.7 5. Regression plots

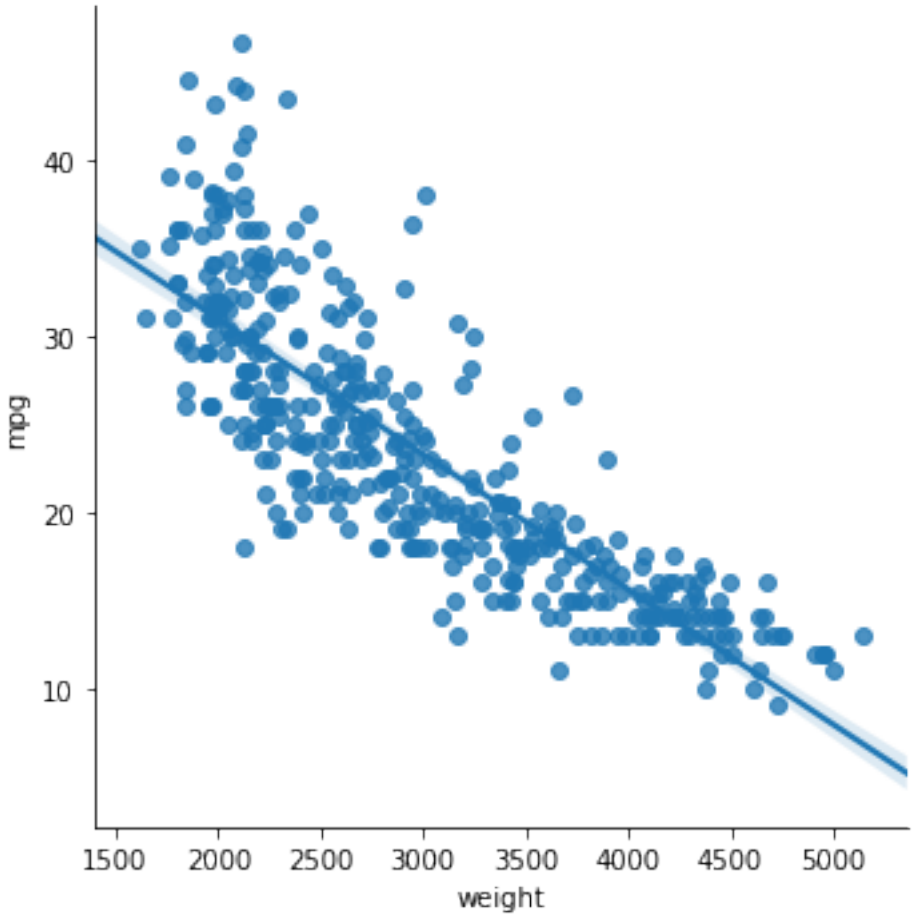
[27]:
```
mpg.head()
```

[27]:
```
    mpg  cylinders  displacement  horsepower  weight  acceleration  \
0  18.0          8         307.0       130.0    3504          12.0
1  15.0          8         350.0       165.0    3693          11.5
2  18.0          8         318.0       150.0    3436          11.0
```

```
3  16.0          8         304.0       150.0    3433          12.0
4  17.0          8         302.0       140.0    3449          10.5

   model_year origin                       name
0          70    usa  chevrolet chevelle malibu
1          70    usa          buick skylark 320
2          70    usa         plymouth satellite
3          70    usa              amc rebel sst
4          70    usa                ford torino
```

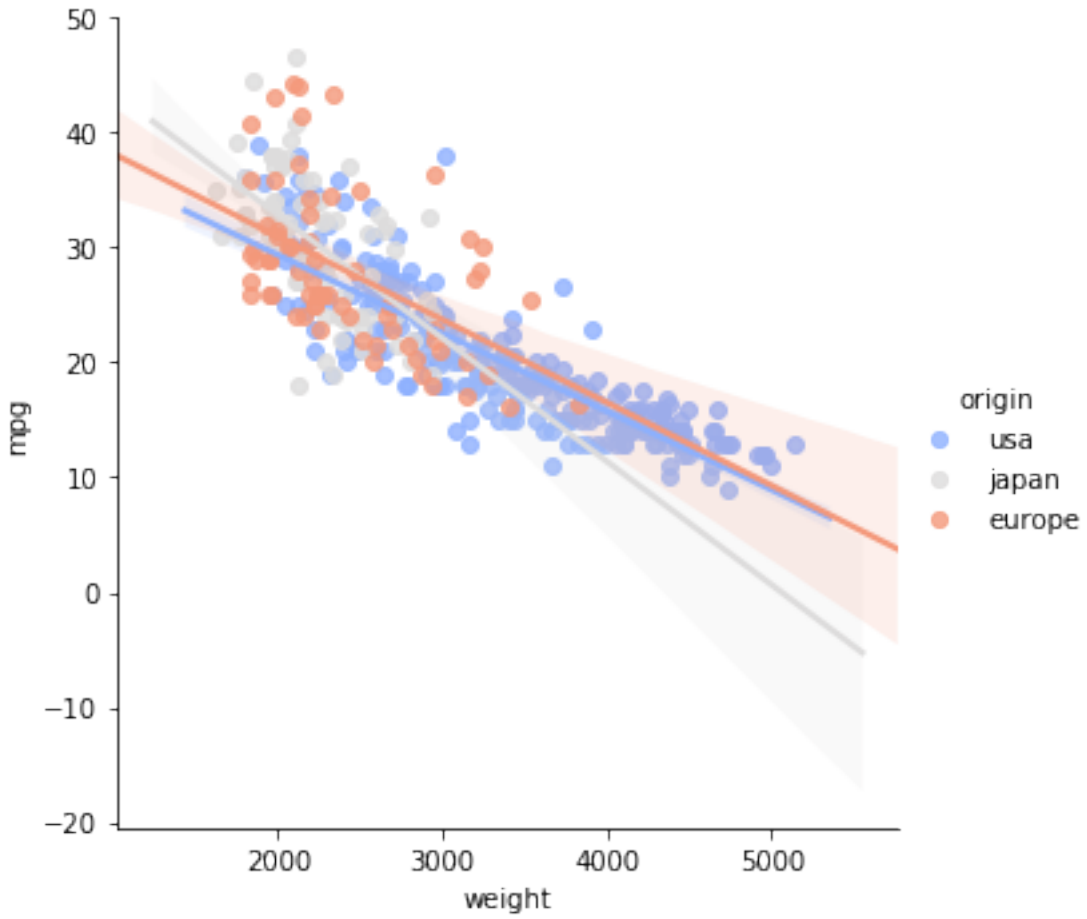
[28]:
```
sns.lmplot(x='weight',y='mpg',data=mpg)
```

[28]:
```
<seaborn.axisgrid.FacetGrid at 0x230ae0641d0>
```



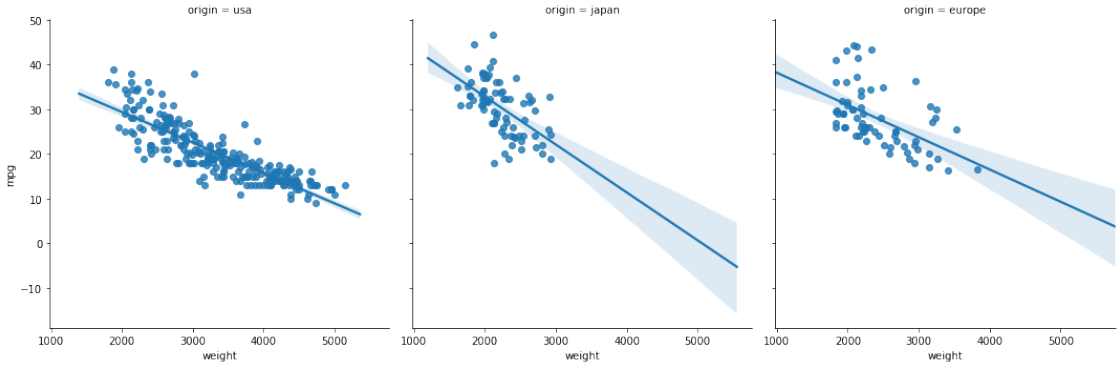
[29]:
```
sns.lmplot(x='weight',y='mpg',data=mpg, hue='origin', palette='coolwarm')
```

[29]:
```
<seaborn.axisgrid.FacetGrid at 0x230ae0c9438>
```



[30]:
```
sns.lmplot(x='weight',y='mpg',data=mpg, col='origin')
```

[30]:
```
<seaborn.axisgrid.FacetGrid at 0x230ae16c6a0>
```



[41]:
```
import pandas as pd
df = pd.read_csv('GDP.csv')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 290 entries, 0 to 289
Data columns (total 2 columns):
DATE    290 non-null object
GDP     290 non-null float64
dtypes: float64(1), object(1)
memory usage: 4.6+ KB
```

[45]:
```
df.tail(10)
```

[45]:

	DATE	GDP
280	2017-01-01	19190.431
281	2017-04-01	19356.649
282	2017-07-01	19611.704
283	2017-10-01	19918.910
284	2018-01-01	20163.159
285	2018-04-01	20510.177
286	2018-07-01	20749.752
287	2018-10-01	20897.804
288	2019-01-01	21098.827
289	2019-04-01	21339.121

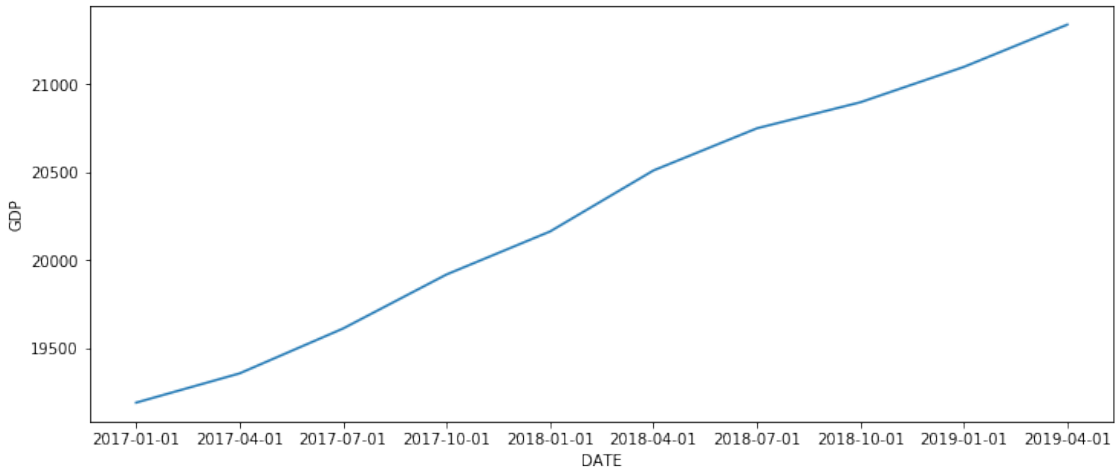
[49]:
```
df.iloc[-10:]
```

[49]:

	DATE	GDP
280	2017-01-01	19190.431
281	2017-04-01	19356.649
282	2017-07-01	19611.704
283	2017-10-01	19918.910
284	2018-01-01	20163.159
285	2018-04-01	20510.177
286	2018-07-01	20749.752
287	2018-10-01	20897.804
288	2019-01-01	21098.827
289	2019-04-01	21339.121

[51]:
```
plt.figure(figsize=(12,5))
sns.lineplot(x="DATE", y="GDP", data=df.iloc[-10:])
```

[51]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1aed3de4f28>
```


21000
20500
20000
19500
GDP
2017-01-01
2017-04-01
2017-07-01
2017-10-01
2018-01-01
2018-04-01
2018-07-01
2018-10-01
2019-01-01
2019-04-01
DATE