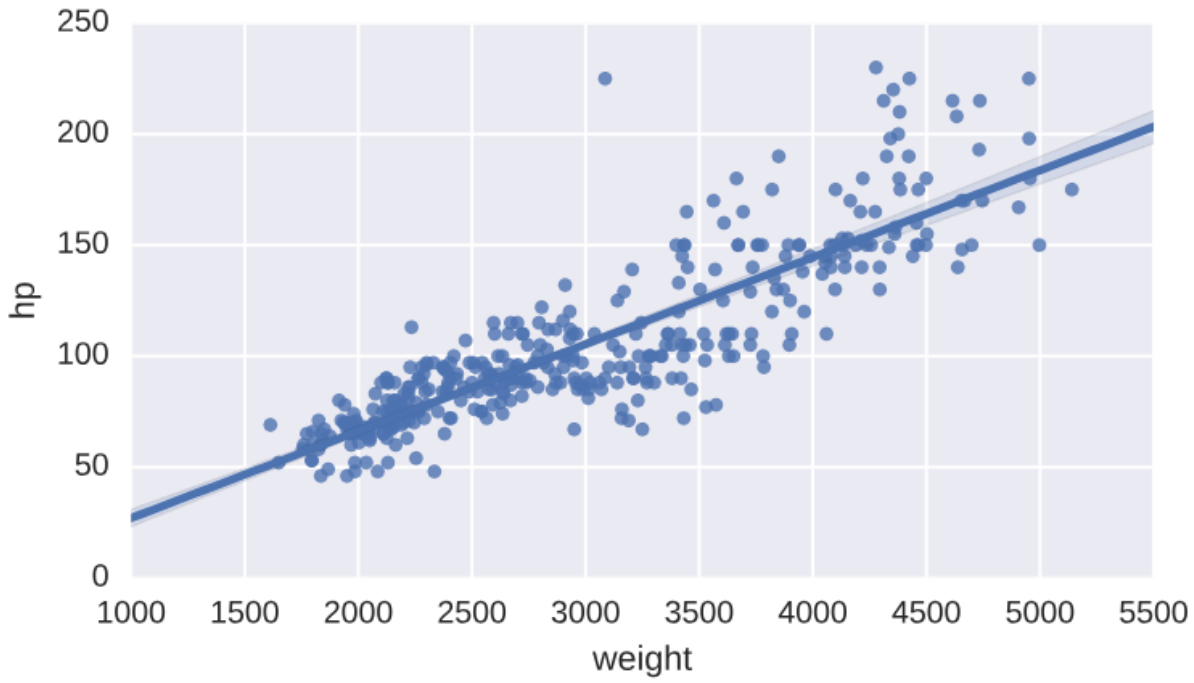
Introduction to Data Visualization with Python

3). Statistical plots with Seaborn

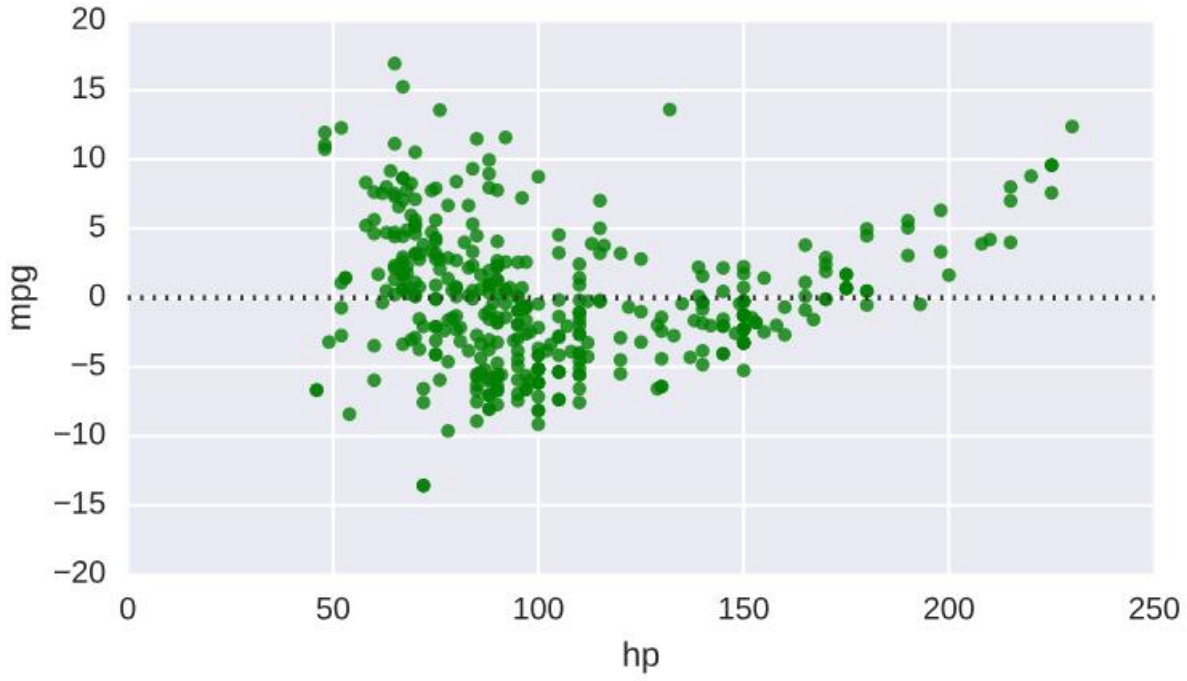
a).

```
# Import plotting modules
import matplotlib.pyplot as plt
import seaborn as sns


# Plot a linear regression between 'weight' and 'hp'
sns.lmplot(x='weight', y='hp', data=auto)


# Display the plot
plt.show()
```



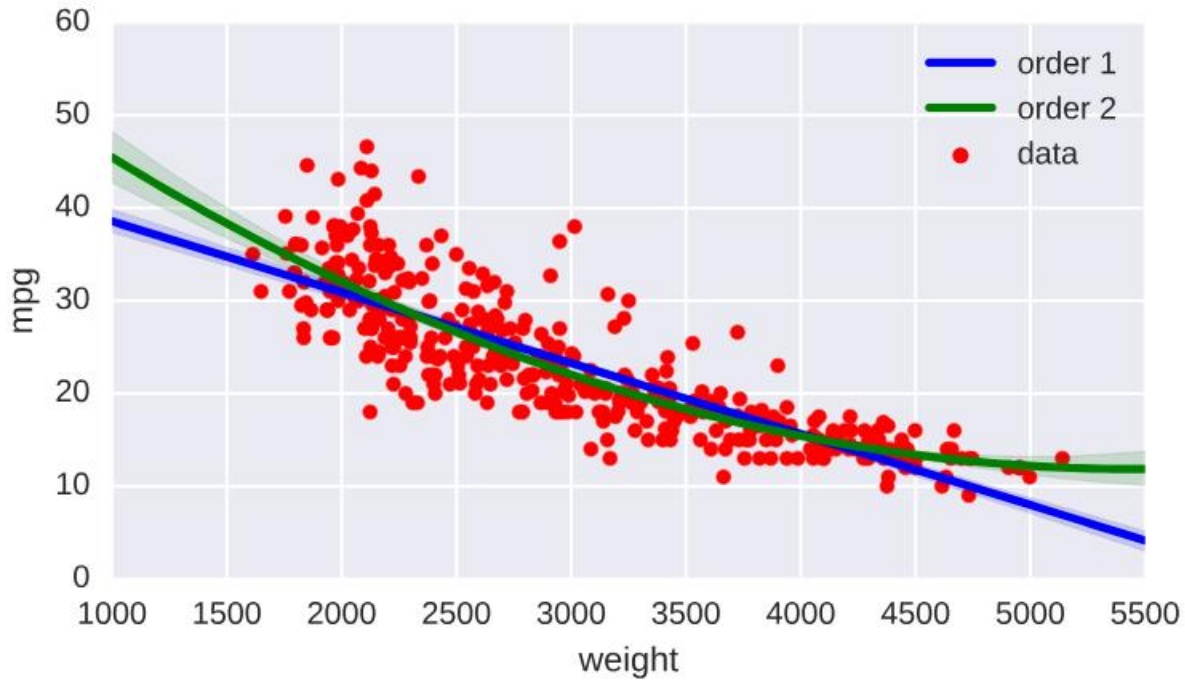
b).

```
# Import plotting modules
import matplotlib.pyplot as plt
import seaborn as sns

# Generate a green residual plot of the regression between 'hp' and 'mpg'
sns.residplot(x='hp', y='mpg', data=auto, color='green')

# Display the plot
plt.show()
```



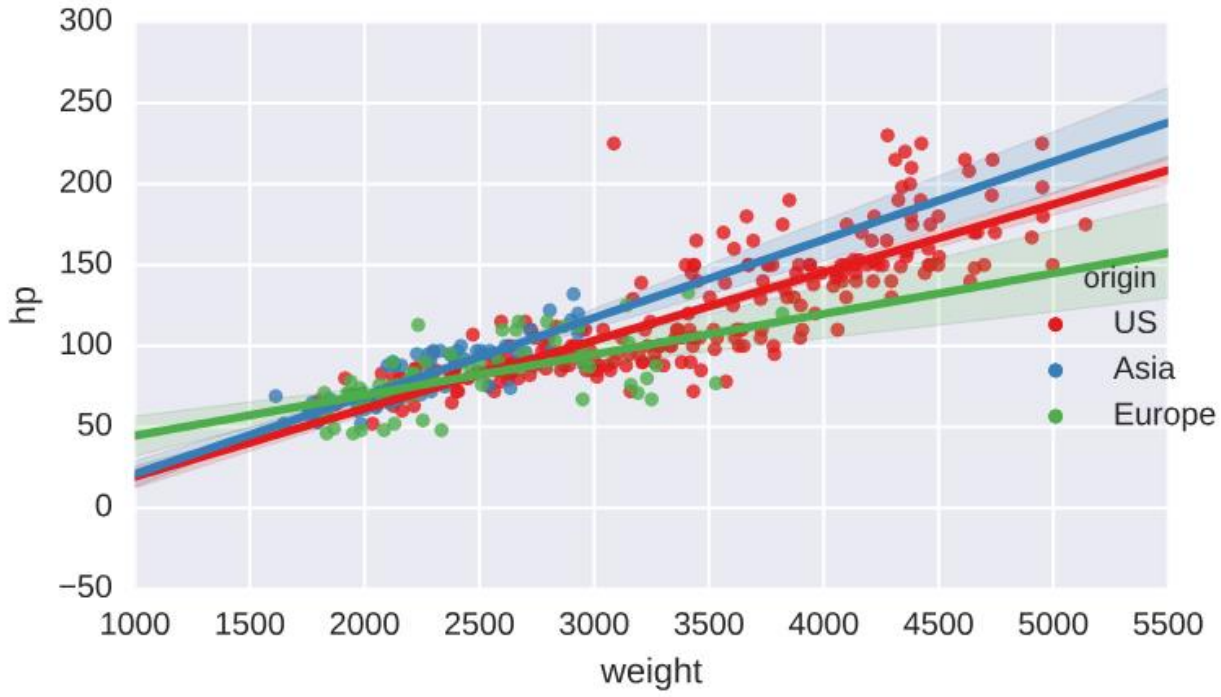
c).

```
# Generate a scatter plot of 'weight' and 'mpg' using red circles
plt.scatter(auto['weight'], auto['mpg'], label='data', color='red', marker='o')

# Plot in blue a linear regression of order 1 between 'weight' and 'mpg'
sns.regplot(x='weight', y='mpg', data=auto, scatter=None, color='blue', label='order 1')

# Plot in green a linear regression of order 2 between 'weight' and 'mpg'
sns.regplot(x='weight', y='mpg', data=auto, scatter=None, color='green', order=2,label='order 2')

# Add a legend and display the plot
plt.legend(loc='upper right')
plt.show()
```



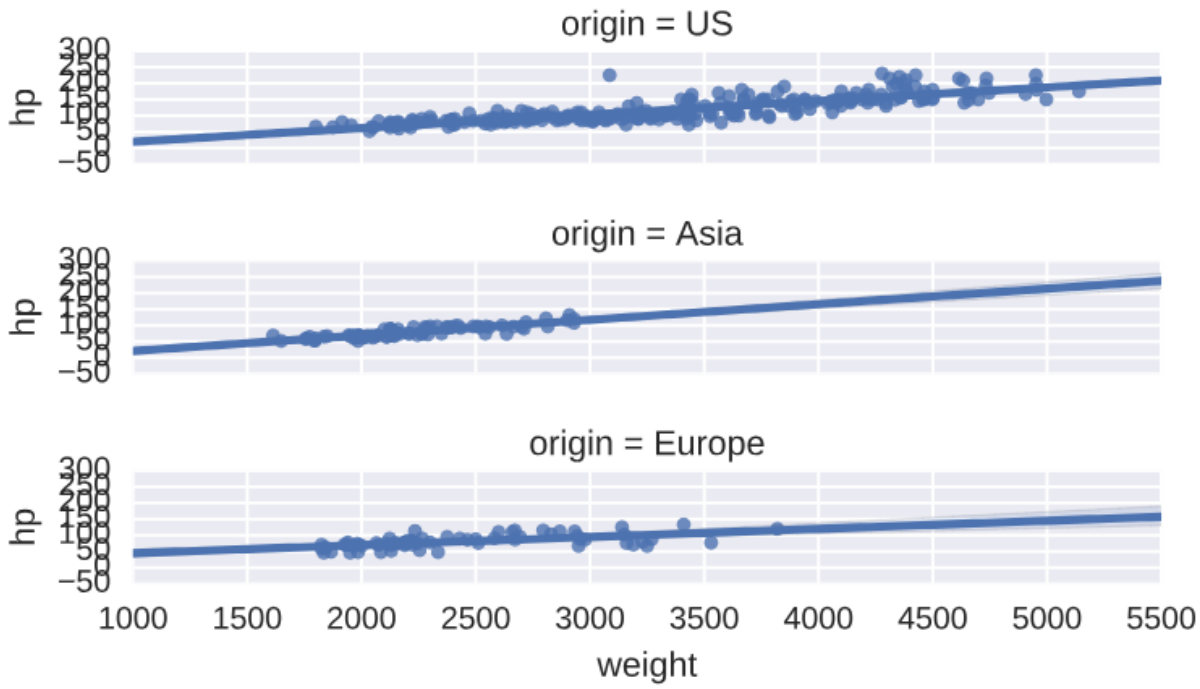
d).

```
# Plot a linear regression between 'weight' and 'hp', with a hue of 'origin' and palette of 'Set1'
sns.lmplot(data=auto,x='weight',y='hp',hue='origin',palette='Set1')

# Display the plot
plt.show()
```



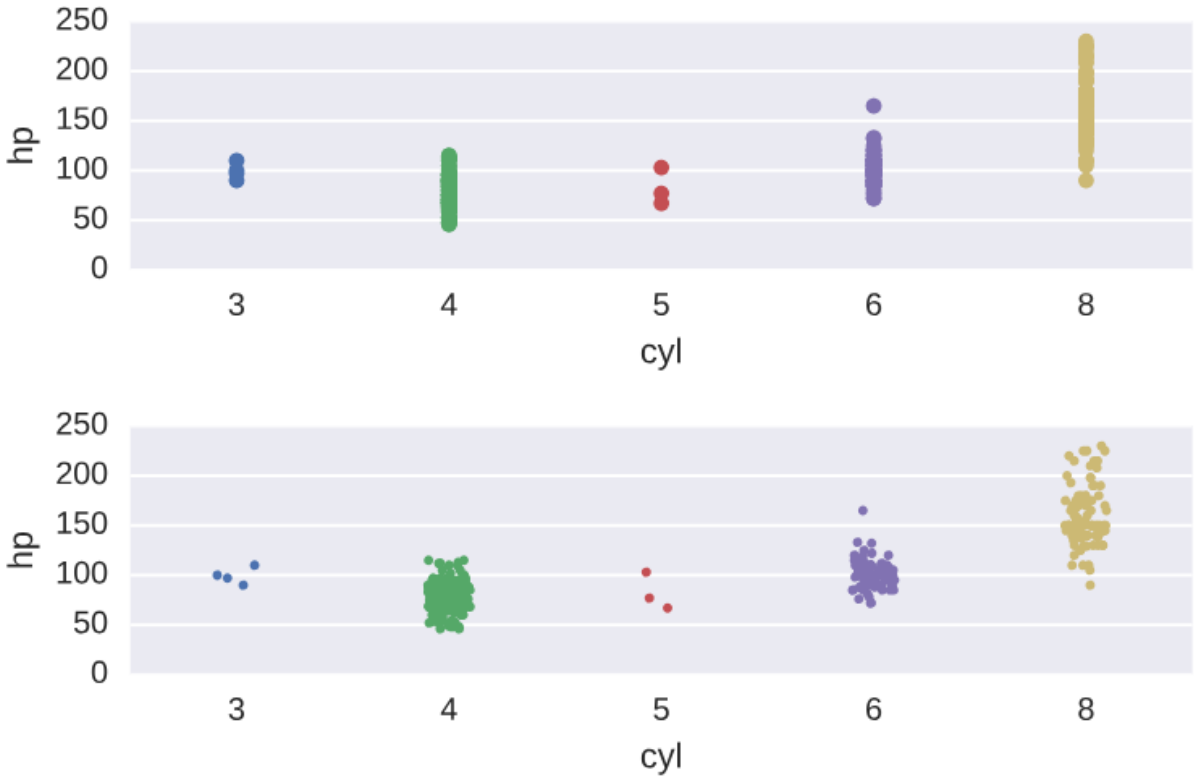
e). Grouping linear regressions by row or column

```
# Plot linear regressions between 'weight' and 'hp' grouped row-wise by 'origin'
sns.lmplot(data=auto,x='weight',y='hp',row='origin')

# Display the plot
plt.show()
```



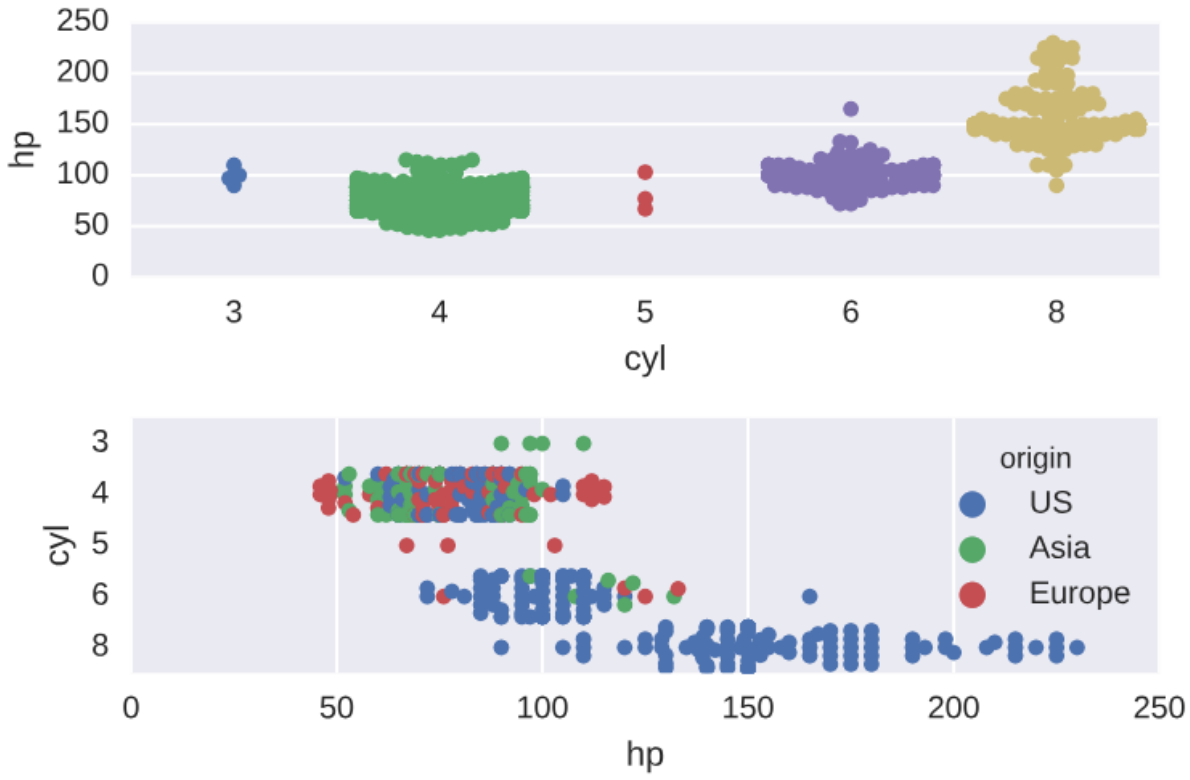
f). constructing strip plots

```
# Make a strip plot of 'hp' grouped by 'cyl'
plt.subplot(2,1,1)
sns.stripplot(x='cyl', y='hp', data=auto)

# Make the strip plot again using jitter and a smaller point size
plt.subplot(2,1,2)
sns.stripplot(x='cyl', y='hp', data=auto, jitter=True, size=3)

# Display the plot
plt.show()
```



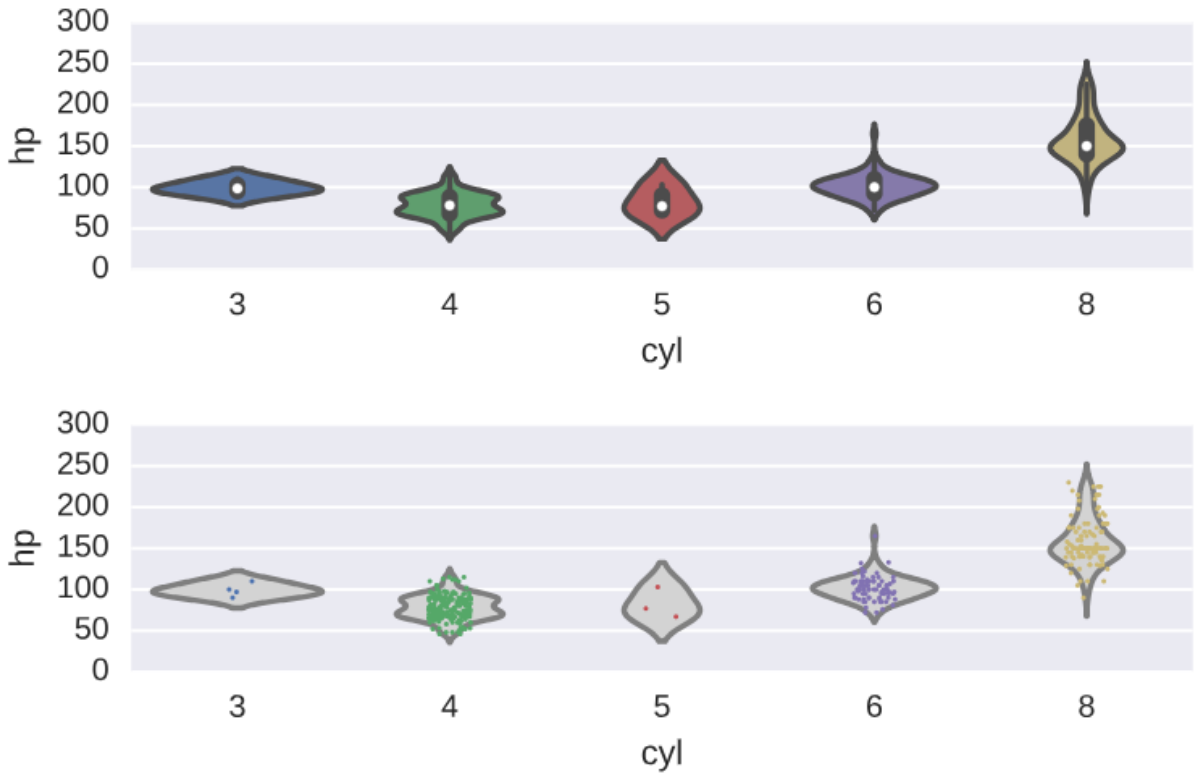
g). Constructing swarm plots

```
# Generate a swarm plot of 'hp' grouped horizontally by 'cyl'
plt.subplot(2,1,1)
sns.swarmplot(x='cyl',y='hp',data=auto)

# Generate a swarm plot of 'hp' grouped vertically by 'cyl' with a hue of 'origin'
plt.subplot(2,1,2)
sns.swarmplot(x='hp',y='cyl',data=auto,hue='origin',orient='h')

# Display the plot
plt.show()
```



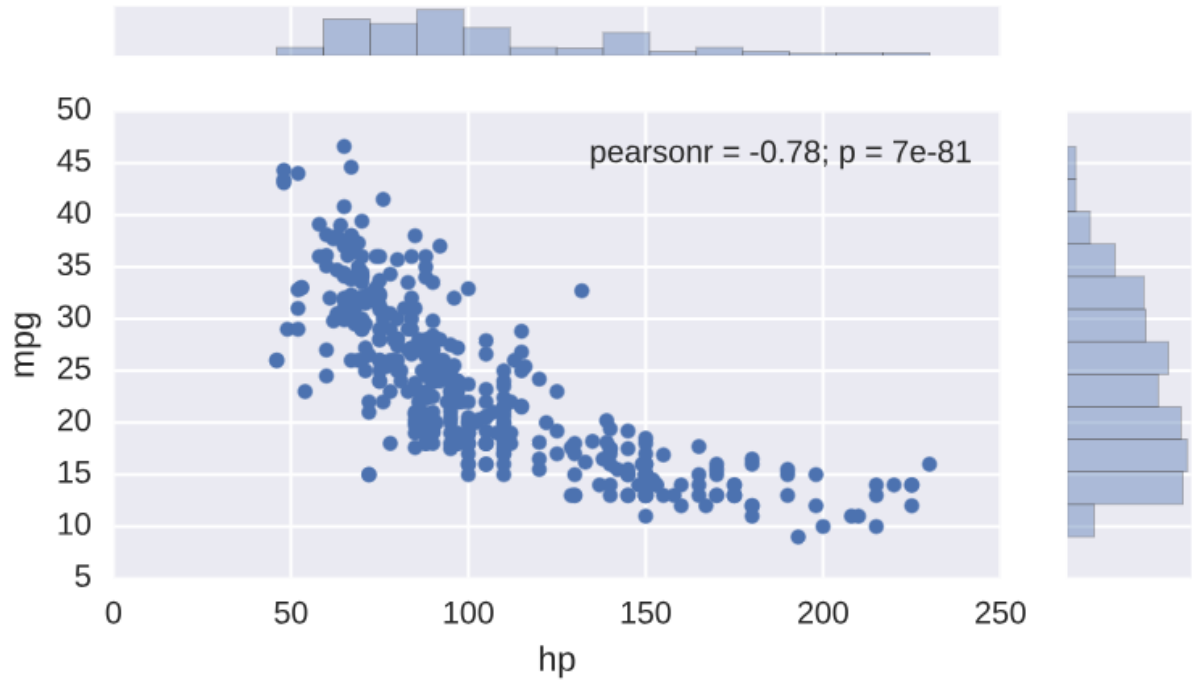
h). Constructing Violon plots and overlaying plots

```
# Generate a violin plot of 'hp' grouped horizontally by 'cyl'
plt.subplot(2,1,1)
sns.violinplot(x='cyl', y='hp', data=auto)

# Generate the same violin plot again with a color of 'lightgray' and without inner annotations
plt.subplot(2,1,2)
sns.violinplot(x='cyl', y='hp', data=auto,inner=None,color='lightgray')

# Overlay a strip plot on the violin plot
sns.stripplot(x='cyl',y='hp',data=auto,size=1.5,jitter=True)

# Display the plot
plt.show()
```



i). Plotting joint distributions (1)

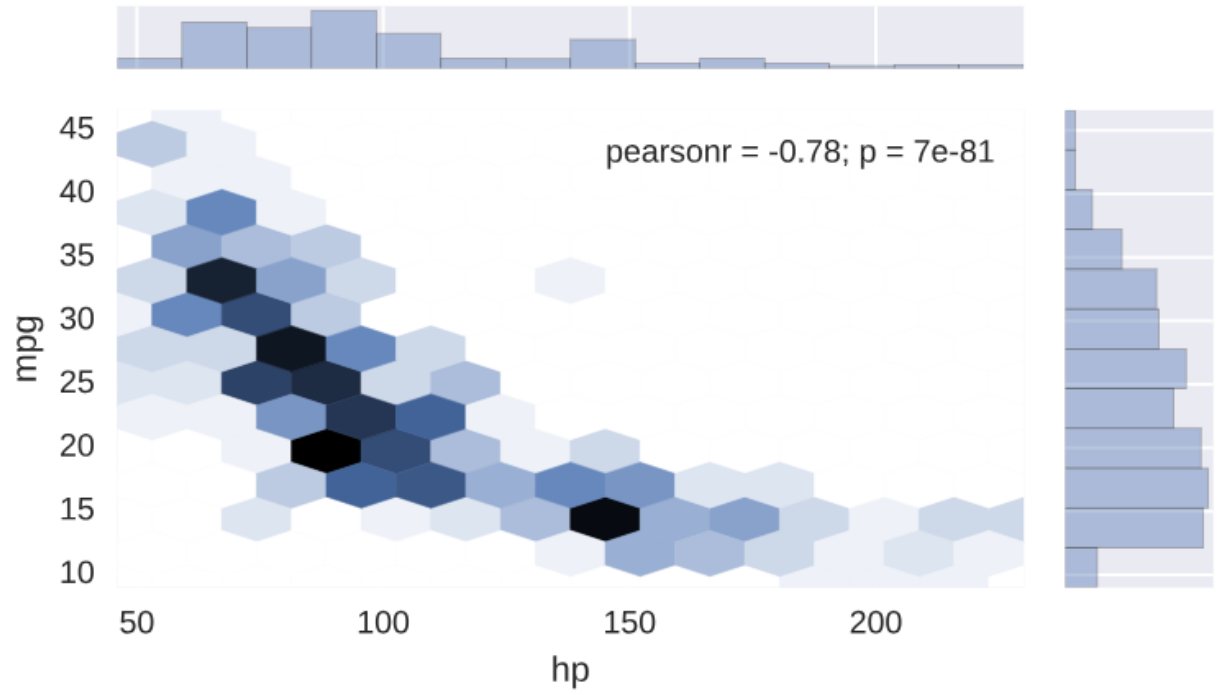
```
# Generate a joint plot of 'hp' and 'mpg'
sns.jointplot(x='hp',y='mpg', data=auto)

# Display the plot
plt.show()
```



j). Plotting joint distributions (2)

- `kind='scatter'` uses a scatter plot of the data points
- `kind='reg'` uses a regression plot (default order 1)
- `kind='resid'` uses a residual plot
- `kind='kde'` uses a *kernel density estimate* of the joint distribution
- `kind='hex'` uses a hexbin plot of the joint distribution

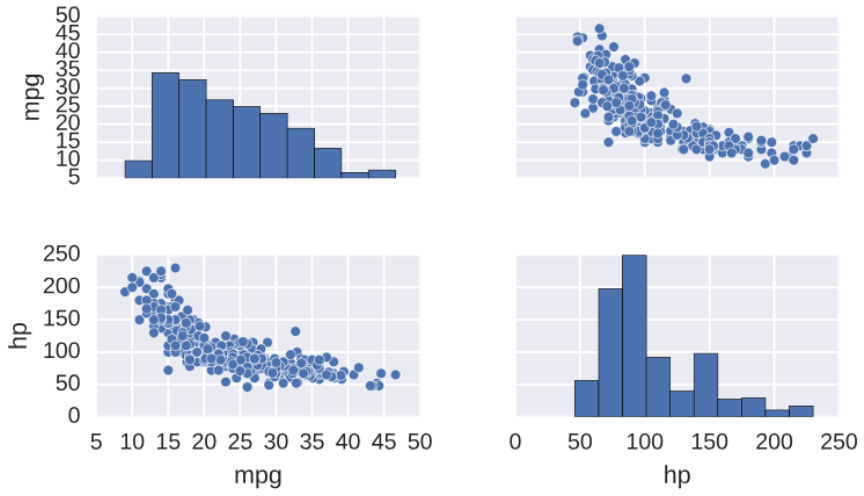
```
# Generate a joint plot of 'hp' and 'mpg' using a hexbin plot

sns.jointplot(x='hp',y='mpg',data=auto,kind='hex')

# Display the plot

plt.show()
```



k). Plotting distributions pairwise (1)

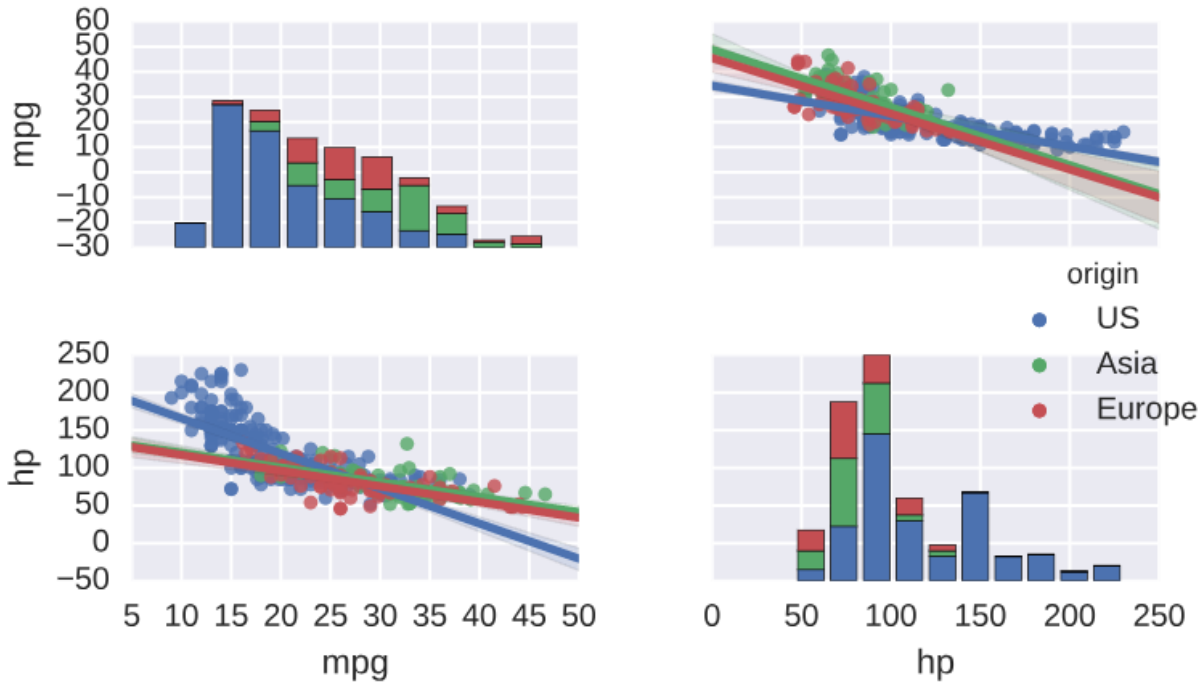
```
# Print the first 5 rows of the DataFrame
print(auto.head())


# Plot the pairwise joint distributions from the DataFrame
sns.pairplot(auto)


# Display the plot
plt.show()
```

```
mpg   hp  origin
0  18.0   88     US
1   9.0  193     US
2  36.1   60    Asia
3  18.5   98     US
4  34.3   78  Europe
```



l). Plotting distributions pairwise (2)

```
# Print the first 5 rows of the DataFrame
print(auto.head())

# Plot the pairwise joint distributions grouped by 'origin' along with regression lines
sns.pairplot(auto,kind='reg',hue='origin')

# Display the plot
plt.show()
```



m). Visualizing correlations with heatmap

```
# Print the covariance matrix
print(cov_matrix)

# Visualize the covariance matrix using a heatmap
sns.heatmap(cov_matrix)

# Display the heatmap
plt.show()
```

```
              mpg        hp    weight     accel     displ
mpg      1.000000 -0.778427 -0.832244  0.423329 -0.805127
hp      -0.778427  1.000000  0.864538 -0.689196  0.897257
weight  -0.832244  0.864538  1.000000 -0.416839  0.932994
accel    0.423329 -0.689196 -0.416839  1.000000 -0.543800
displ   -0.805127  0.897257  0.932994 -0.543800  1.000000
```