DSO 545: Statistical Computing and Data Visualization

Abbass Al Sharif

Fall 2019

Lab 8: Data Visualization Using Matplotlib (Part2)

```
#import necessary packages
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings;
```

```
#supress warnings
warnings.filterwarnings(action='once')
```

Dot Plot

1. Load the dataset `mpg.csv`, and create the following dot plot using matplotlib. This dotplot looks at the average miles per gallon for each car manufacturer.

```
# Set Parameters for plots in matplotlib

large = 22; med = 16; small = 12

params = {"figure.figsize": (16,10),
          "figure.dpi": 200,
          "figure.titlesize": large,
          "axes.titlesize": med,
          "axes.labelsize": med,
          "legend.fontsize":med,
          "xtick.labelsize": med,
          "ytick.labelsize":med
          }

plt.rcParams.update(params)
plt.style.use("seaborn-whitegrid")
sns.set_style("white")
```

```
### Dot plot

# read data
data = pd.read_csv("mpg.csv")
data.head()

# clean data
```

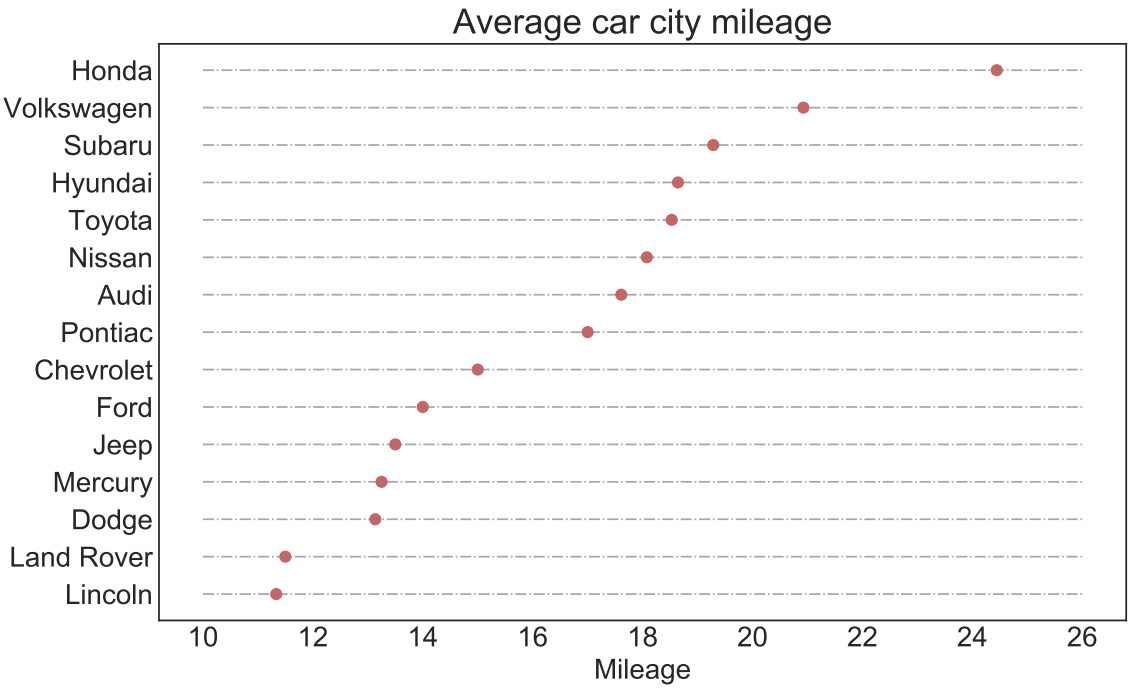
```
##   manufacturer model  displ  year  cyl       trans drv  cty  hwy fl    class
## 0         audi    a4    1.8  1999    4    auto(l5)   f   18   29  p  compact
## 1         audi    a4    1.8  1999    4  manual(m5)   f   21   29  p  compact
## 2         audi    a4    2.0  2008    4  manual(m6)   f   20   31  p  compact
## 3         audi    a4    2.0  2008    4    auto(av)   f   21   30  p  compact
```

```
## 4         audi    a4    2.8  1999    6    auto(l5)   f   16   26  p  compact
```

```
dataviz = data[['cty','manufacturer']].groupby('manufacturer').mean()
dataviz.sort_values('cty', inplace= True)
dataviz.reset_index(inplace = True)
dataviz

## OR
##dataviz =data[['cty', 'manufacturer']].groupby('manufacturer').apply(lambda x: x.mean())
```

```
##    manufacturer        cty
## 0       lincoln  11.333333
## 1    land rover  11.500000
## 2         dodge  13.135135
## 3       mercury  13.250000
## 4          jeep  13.500000
## 5          ford  14.000000
## 6     chevrolet  15.000000
## 7       pontiac  17.000000
## 8          audi  17.611111
## 9        nissan  18.076923
## 10       toyota  18.529412
## 11      hyundai  18.642857
## 12       subaru  19.285714
## 13   volkswagen  20.925926
## 14        honda  24.444444
```

```
#visualize data

plt.figure(figsize=(10, 6))

# draw plot
plt.scatter(x = dataviz.cty,
            y = dataviz.manufacturer.str.title(),
            color = "firebrick",
            alpha = 0.7
            )
ax = plt.gca()
ax.hlines(y = dataviz.index,
          xmin = 10,
          xmax = 26,
          color = "grey",
          alpha = 0.7,
          linewidth = 1,
          linestyle = "-.")

# Title, axis, labels, and ylim
plt.title("Average car city mileage", fontsize = 20)
plt.xlabel("Mileage")
plt.show()
```



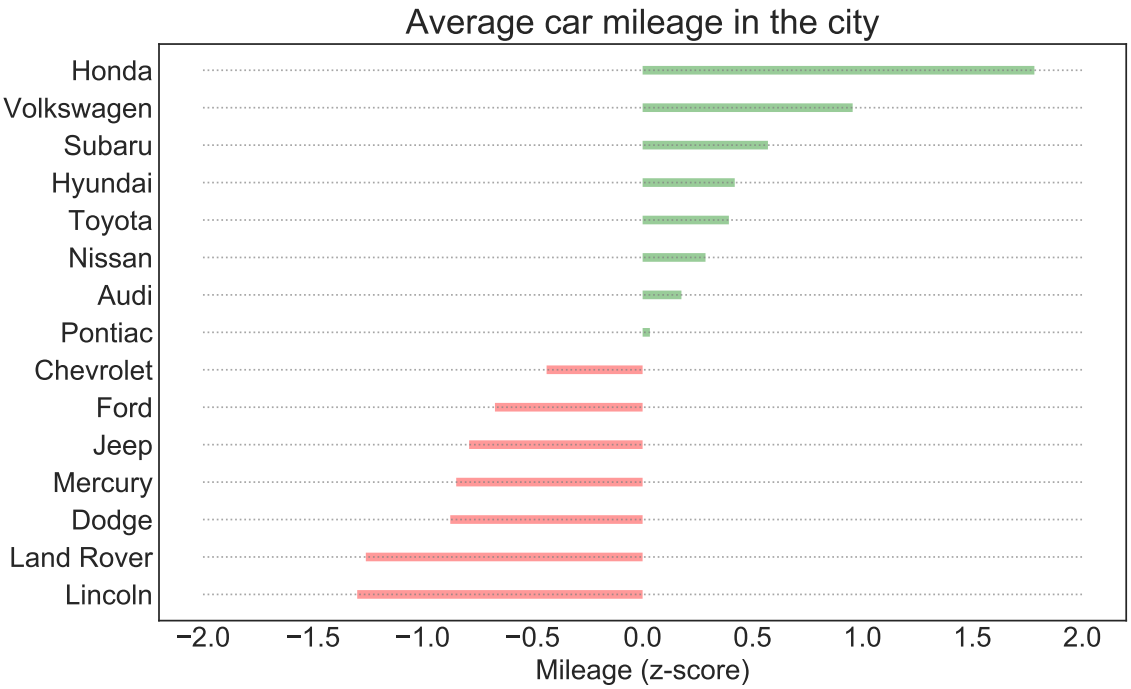
Diverging Bars

2. Create the following diverging bar plot using matplotlib. This plot looks at the average miles per gallon (z-scores) for each car manufacturer.

```
# read the data
data = pd.read_csv("mpg.csv")

# clean and prepare data

# get the standarized scores for city mileage

data['cty_z'] = (data.cty - data.cty.mean())/data.cty.std()

dataviz = data[['manufacturer', 'cty_z']].\
            groupby('manufacturer').\
            mean().\
            sort_values("cty_z").\
            reset_index()

dataviz['col'] = ["red" if x<0 else "green" for x in dataviz.cty_z]
```

```
plt.figure(figsize=(10, 6))

ax = plt.gca()
ax.hlines(y = dataviz.manufacturer.str.title(),
          xmin = 0,
```

```
           xmax = dataviz.cty_z,
           color = dataviz.col,
           linewidth = 5,
           alpha = 0.4)

ax.hlines(y = dataviz.index,
          xmin = -2,
          xmax = 2,
          color = "grey",
          alpha = 0.7,
          linewidth = 1,
          linestyle = "dotted")

ax.set_title('Average car mileage in the city', fontdict={"size": 20})
ax.set_xlabel('Mileage (z-score)')
plt.show()
```



Correllogram

3. Create a correllogram that shows the correlation between all numerical variables in the `mpg.csv` dataset. (**Please note that the aspect ratio of the figure will look differently in Jupter Notebook**)

```
# import seaborn
import seaborn as sns


# read the data
data = pd.read_csv("mpg.csv")
```

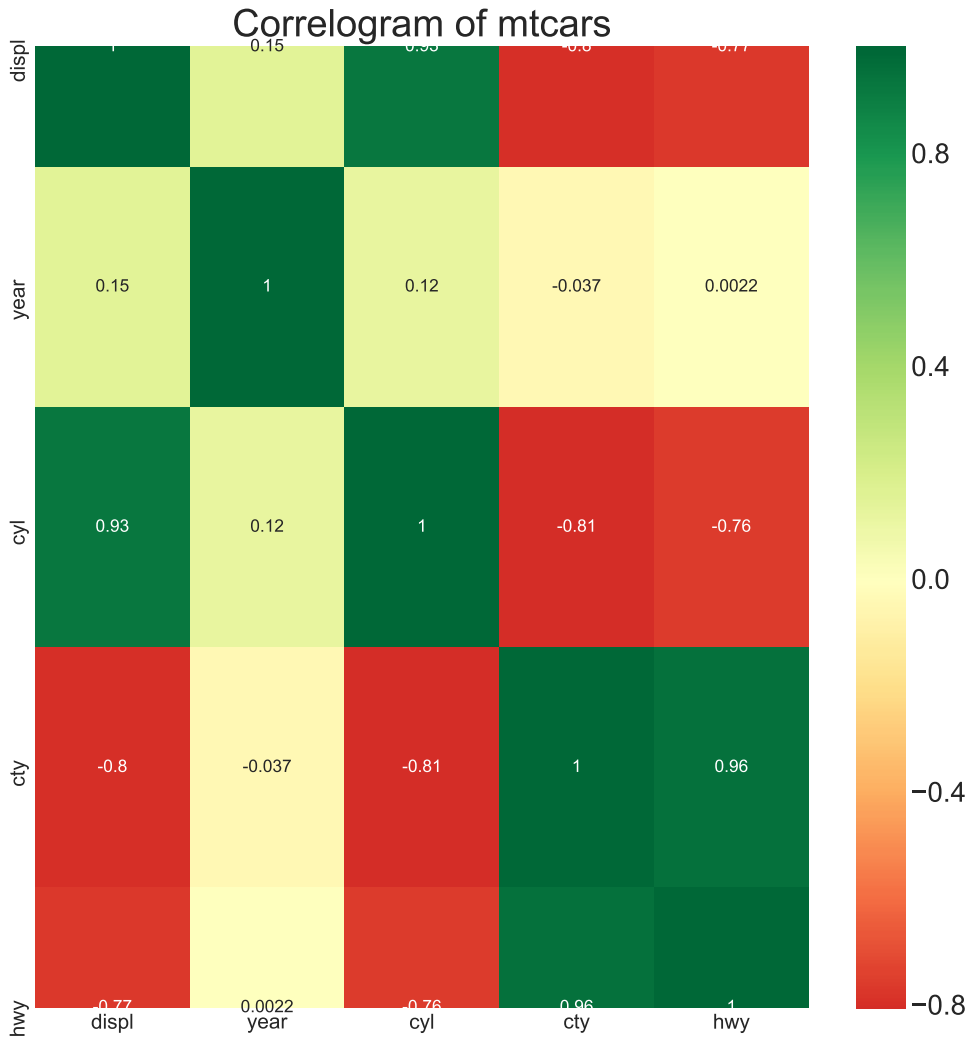
```
# draw plot

plt.figure()

sns.heatmap(data.corr(),
            cmap='RdYlGn',
            center = 0,
           annot=True)

plt.title('Correlogram of mtcars', fontsize=22)
plt.xticks(fontsize=12)
```

```
## (array([0.5, 1.5, 2.5, 3.5, 4.5]), <a list of 5 Text xticklabel objects>)
```

```
plt.yticks(fontsize=12)
```

```
## (array([0.5, 1.5, 2.5, 3.5, 4.5]), <a list of 5 Text yticklabel objects>)
```

```
plt.show()
```



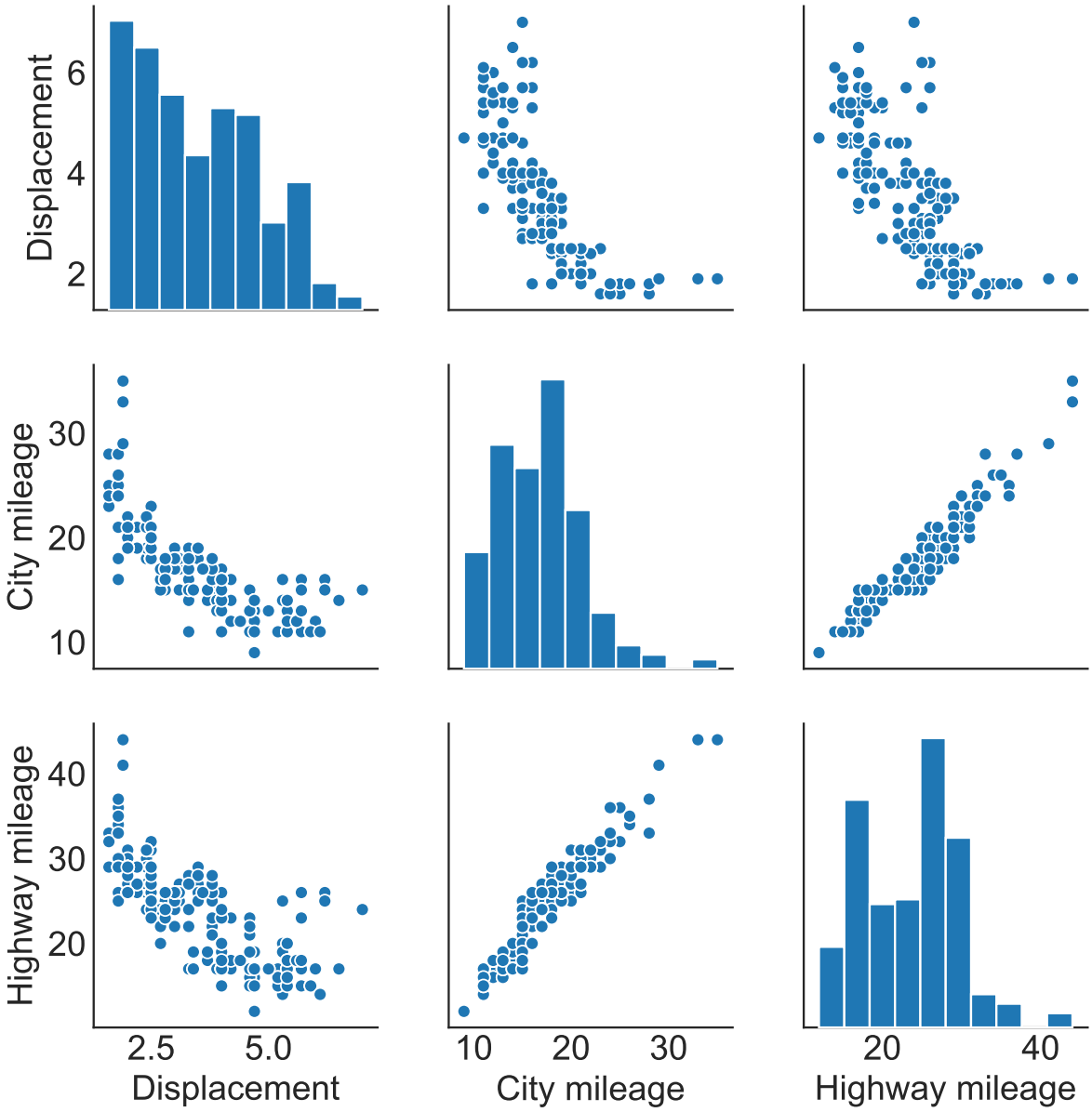
Pariwise Plots

4. Create a pairwise plot to show the relationship between the following numerical variables in the `mpg.csv` dataset.

```
# prepare data
dataviz = data[['displ', 'cty','hwy']]
dataviz.columns = ['Displacement', 'City mileage', 'Highway mileage']

# Plot
plt.figure(figsize=(10,8), dpi= 80)
sns.pairplot(dataviz)
```

```
## <seaborn.axisgrid.PairGrid object at 0x1265ce390>
```

```
plt.show()
```

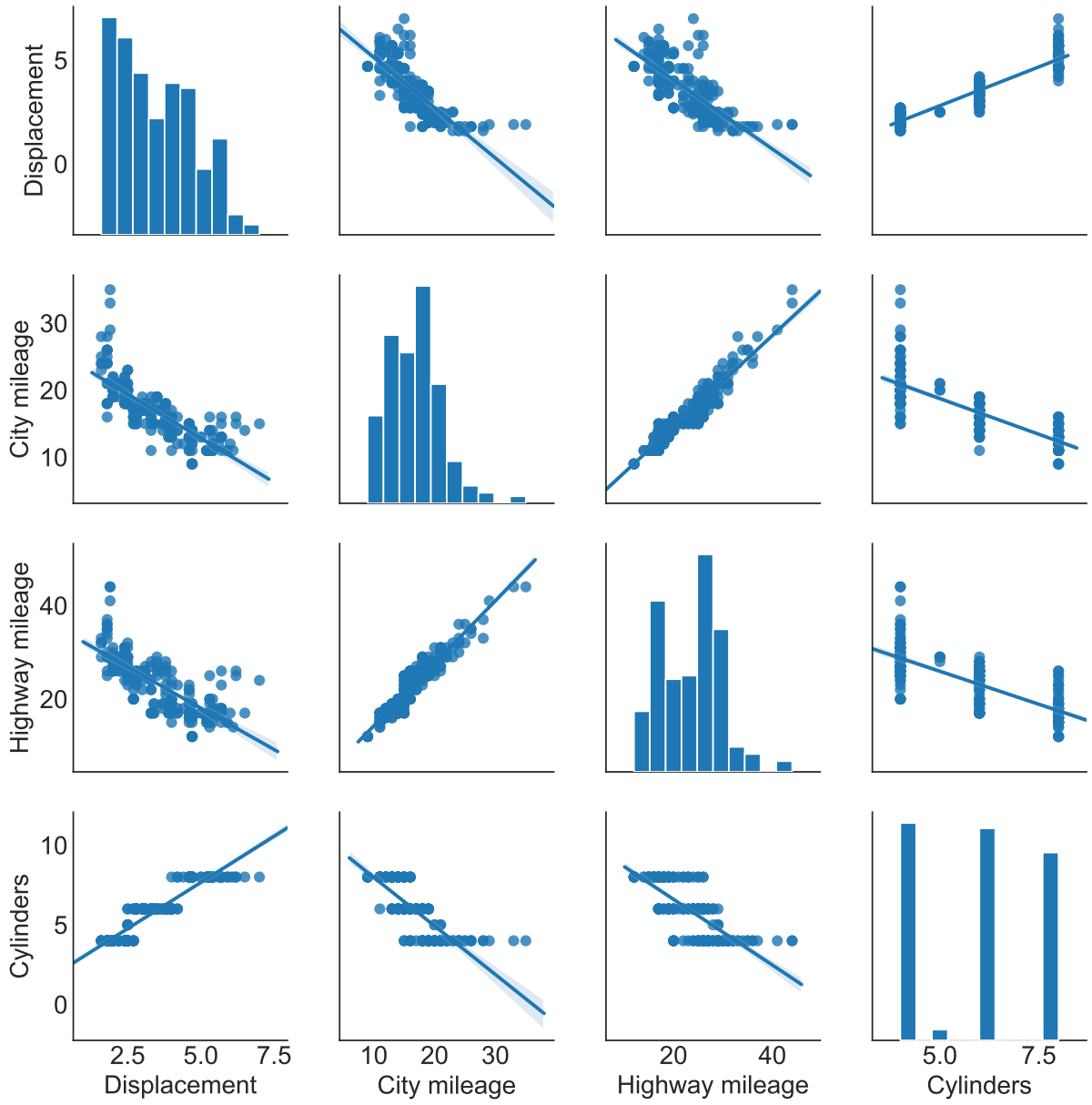


5. Update the previous pairwise plot to the regression lines.

```
# prepare data
dataviz = data[['displ', 'cty','hwy', 'cyl']]
dataviz.columns = ['Displacement', 'City mileage', 'Highway mileage', 'Cylinders']

# Plot
plt.figure(figsize=(10,8), dpi= 80)
sns.pairplot(dataviz,
             kind = 'reg')
```

```
## <seaborn.axisgrid.PairGrid object at 0x127ff3610>
```

```
plt.show()
```

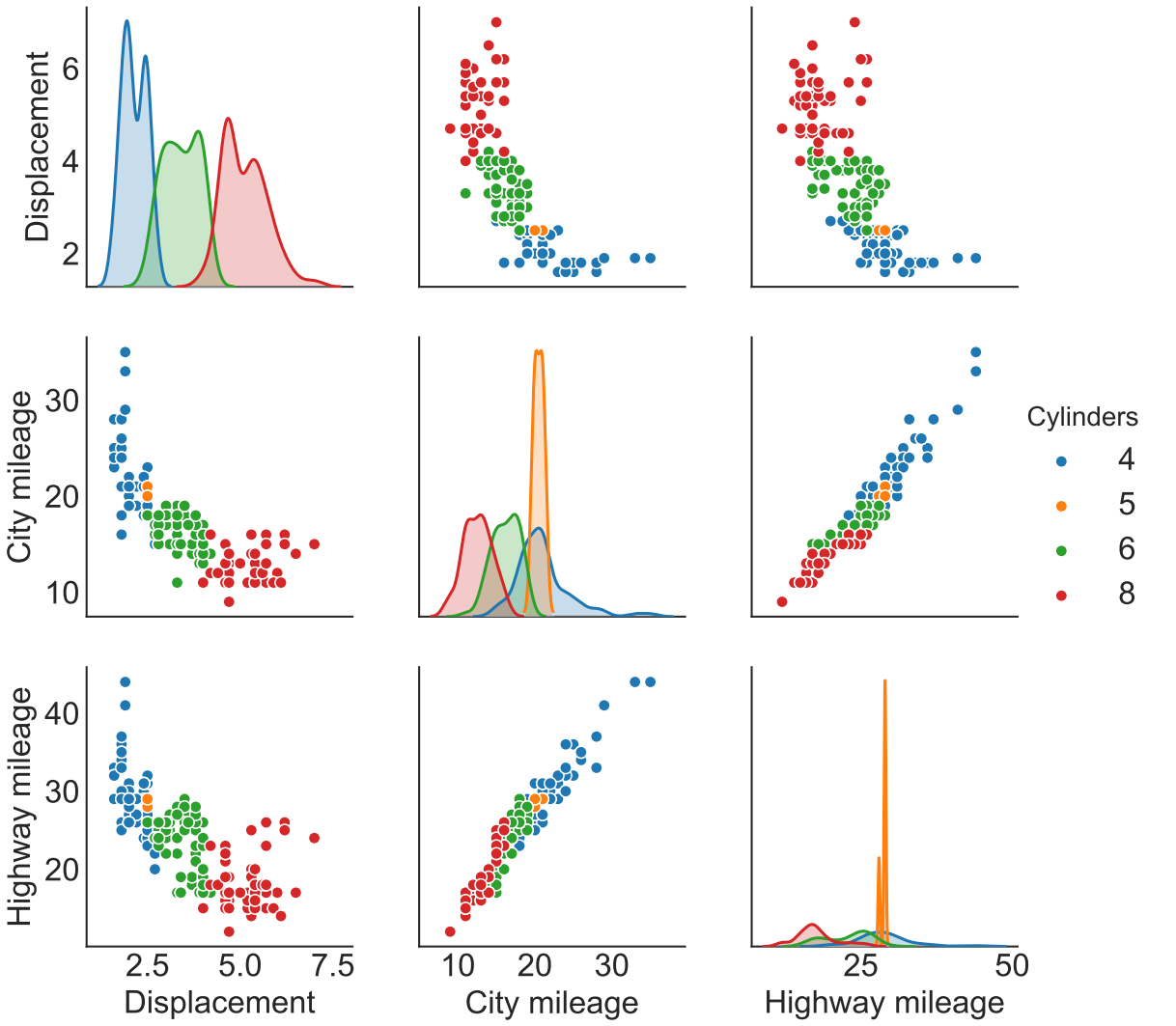


6. Update the initial pairwise plot to show the relationships among the different car engine sizes (cyl).

```
# prepare data
dataviz = data[['displ', 'cty','hwy', 'cyl']]
dataviz.columns = ['Displacement', 'City mileage', 'Highway mileage', 'Cylinders']

# Plot
plt.figure(figsize=(10,8), dpi= 80)
sns.pairplot(dataviz,
             x_vars = dataviz.columns[0:3],
             y_vars = dataviz.columns[0:3],
             hue = 'Cylinders')
```

```
## <seaborn.axisgrid.PairGrid object at 0x129ff3290>
##
## /anaconda3/envs/r-reticulate/lib/python3.7/site-packages/statsmodels/nonparametric/kde.py:487: Runtin
##   binned = fast_linbin(X, a, b, gridsize) / (delta * nobs)
## /anaconda3/envs/r-reticulate/lib/python3.7/site-packages/statsmodels/nonparametric/kdetools.py:34: Ru
##   FAC1 = 2*(np.pi*bw/RANGE)**2
```

```
plt.show()
```



Density Plots

7. Create the following density plot to describe how the distribution of city mileage varies with respect the number of cylinders.

```
import seaborn as sns

data = pd.read_csv('mpg.csv')
```

```
plt.figure()

sns.kdeplot(data.loc[data.cyl == 4, 'cty'],
            shade = True,
            color = 'green',
            label = "Cyl = 4",
            alpha = 0.4)

sns.kdeplot(data.loc[data.cyl == 5, 'cty'],
            shade = True,
            color = 'deeppink',
            label = "Cyl = 5",
            alpha = 0.4)
```

```
## <matplotlib.axes._subplots.AxesSubplot object at 0x1265c6c50>
```

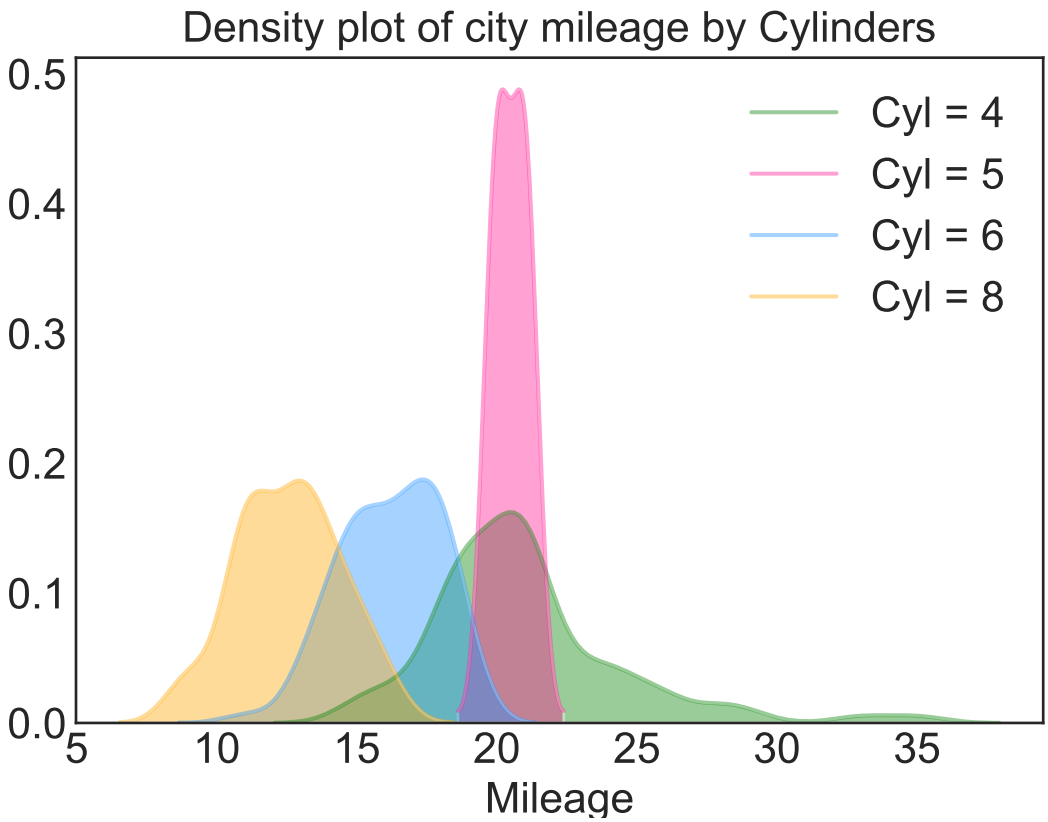
```
sns.kdeplot(data.loc[data.cyl == 6, 'cty'],
            shade = True,
            color = 'dodgerblue',
            label = "Cyl = 6",
            alpha = 0.4)
```

```
## <matplotlib.axes._subplots.AxesSubplot object at 0x1265c6c50>
```

```
sns.kdeplot(data.loc[data.cyl == 8, 'cty'],
            shade = True,
            color = 'orange',
            label = "Cyl = 8",
            alpha = 0.4)
```

```
## <matplotlib.axes._subplots.AxesSubplot object at 0x1265c6c50>
```

```
plt.xlabel('Mileage')
plt.title("Density plot of city mileage by Cylinders")
plt.show()
```



Plotting with different scales using secondary Y axis

8. Use the `economics.csv` dataset to create the following plot which shows the relationship between personal savings rate and # of unemployed from 1967 to 2012.

```
data = pd.read_csv("economics.csv")
data.head()
```

```
##          date    pce     pop  psavert  uempmed  unemploy
## 0  1967-07-01  507.4  198712     12.5      4.5      2944
## 1  1967-08-01  510.5  198911     12.5      4.7      2945
## 2  1967-09-01  516.3  199113     11.7      4.6      2958
## 3  1967-10-01  512.9  199311     12.5      4.9      3143
## 4  1967-11-01  518.1  199498     12.5      4.7      3066
```

```
fig = plt.figure()

plt.plot(data.date, data.psavert, color = 'tab:red') # Plot line 1

ax_left = plt.gca()

# set left y-axis
ax_left.tick_params(axis = 'y', labelcolor = "tab:red")
ax_left.set_ylabel("Personal Savings Rate", color = "tab:red", fontsize = 16)
ax_left.grid(alpha = 0.4)
```

```
# set secondary y-axis (right)
ax_right = ax_left.twinx() # inistantiate a second axes that share the same x-axis

ax_right.plot(data.date, data.unemploy, color = 'tab:blue') # Plot line 2
ax_right.tick_params(axis = 'y', labelcolor = "tab:blue")
ax_right.set_ylabel("# Unemployed (1000's)", color = 'tab:blue', fontsize = 16)

# set x-axis
ax_left.set_xlabel('Year', fontsize = 16)
ax_left.set_xticks(np.arange(0, len(data.date), 60))
```

```
## [<matplotlib.axis.XTick object at 0x12b4e1d90>, <matplotlib.axis.XTick object at 0x12b4e1510>, <matpl
```

```
ax_left.set_xticklabels(data.date[::60], rotation = 0, fontsize = 10)

#set title
```

```
## [Text(0, 0, '1967-07-01'), Text(0, 0, '1972-07-01'), Text(0, 0, '1977-07-01'), Text(0, 0, '1982-07-0
```

```
plt.title("Personal Savings Rate vs. Unemployed", fontsize = 20)

plt.show()
```