week7

January 26, 2018

1 MIS 492 - Data Analysis and Visualization

1.1 Week 7

1.2 Multivariate Visualization

1.2.1 Dr. Mohammad AlMarzouq

2 Multivariate Visualization

- Used to compare univariate distributions between groups
- Used to find relations between 2 variables
- Used to see how a relationship changes between 2 variables across groups

3 Plots

- Best at representing two variables on the X and Y axes
- Three variables possible with 3D plots but should be avoided if possible.
- Third variable usually represented as color, shape, or another plot
  - Most suitable for comparison of relationship or distributions across groups

In [82]:
```
# Setup the libraries
%matplotlib inline
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

sns.set(color_codes=True)


# lets load the data again
weather_df = pd.read_csv("https://raw.githubusercontent.com/vega/vega-datasets/gh-pages
cars_df = pd.read_json("https://github.com/vega/vega-datasets/raw/gh-pages/data/cars.js
```

4 Plotting Two Variables

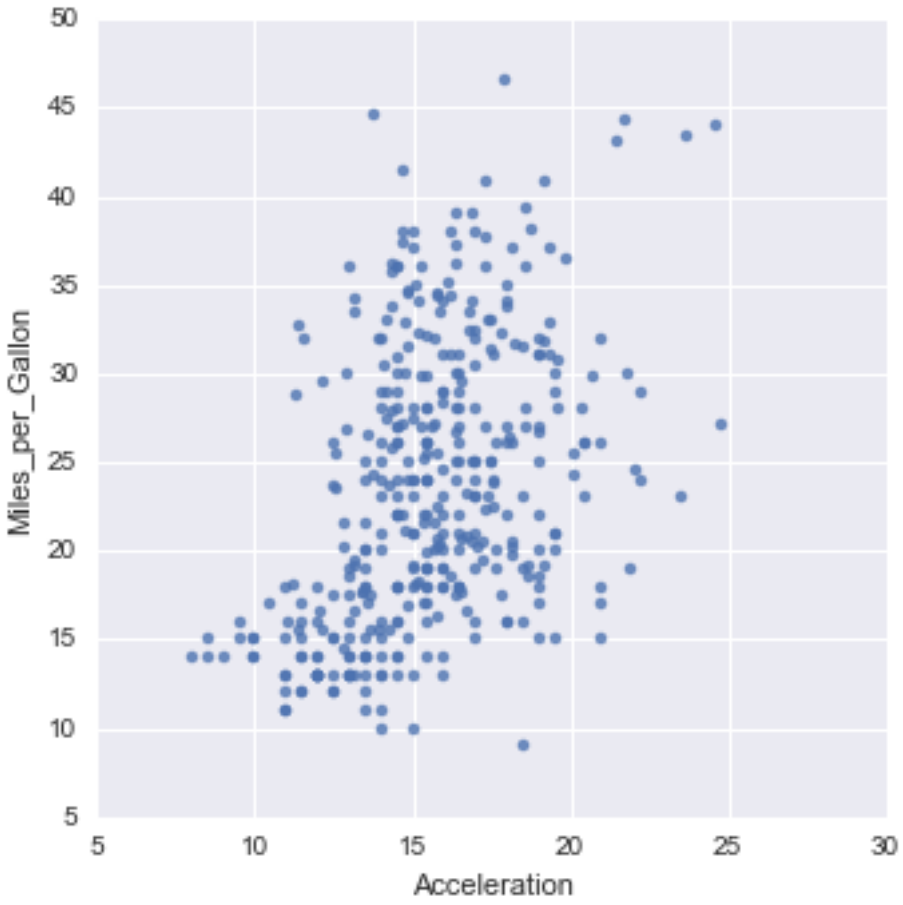
- Both matplot and seaborn could be used
- Scatter plot typically used
- Can help detect relations
  - Time searies plots are a special form of these plots showing relationship to time
  - line plots possible with time series

```
In [5]: # Scatter plot for Acceleration X Miles per gallon
        plt.scatter(cars_df.Acceleration, cars_df.Miles_per_Gallon)
```

```
Out[5]: <matplotlib.collections.PathCollection at 0x1190e8358>
```

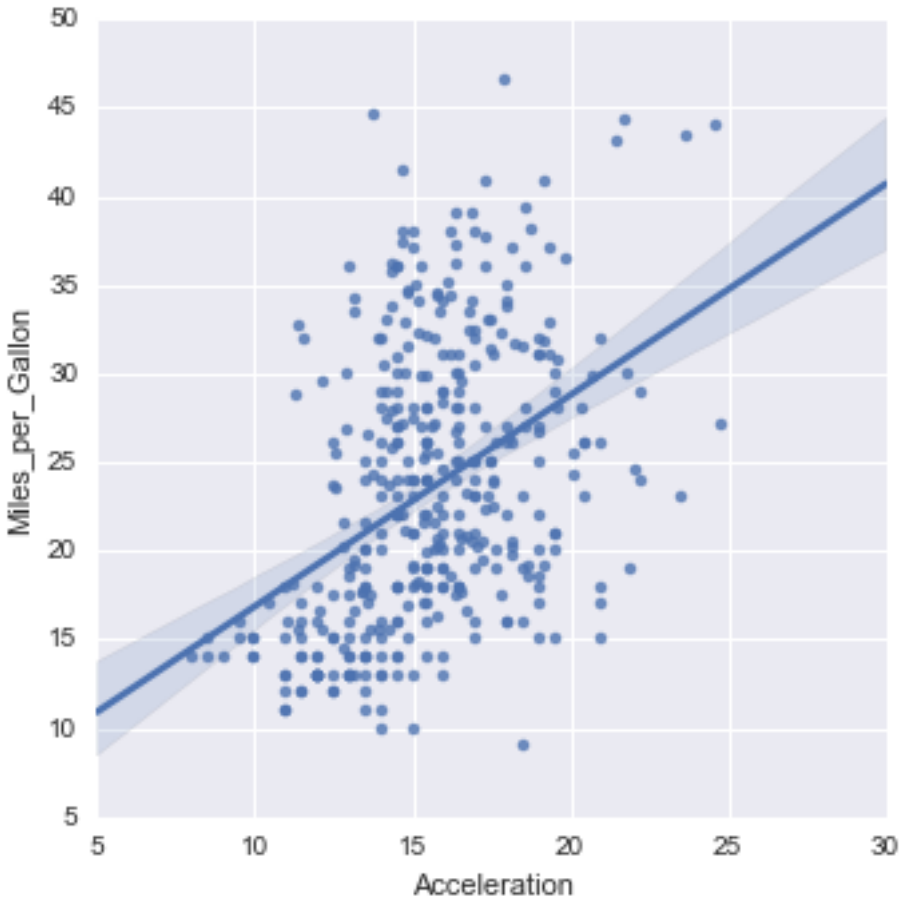

```
In [17]: # Scatter plot with Seaborn
         sns.lmplot(x='Acceleration', y='Miles_per_Gallon', data=cars_df, fit_reg=False)
```

```
Out[17]: <seaborn.axisgrid.FacetGrid at 0x1194161d0>
```




```
In [18]: # Seaborn can also fit a regression line to show the direction of the relation
         sns.lmplot(x='Acceleration', y='Miles_per_Gallon', data=cars_df)
```

```
Out[18]: <seaborn.axisgrid.FacetGrid at 0x118eccb00>
```

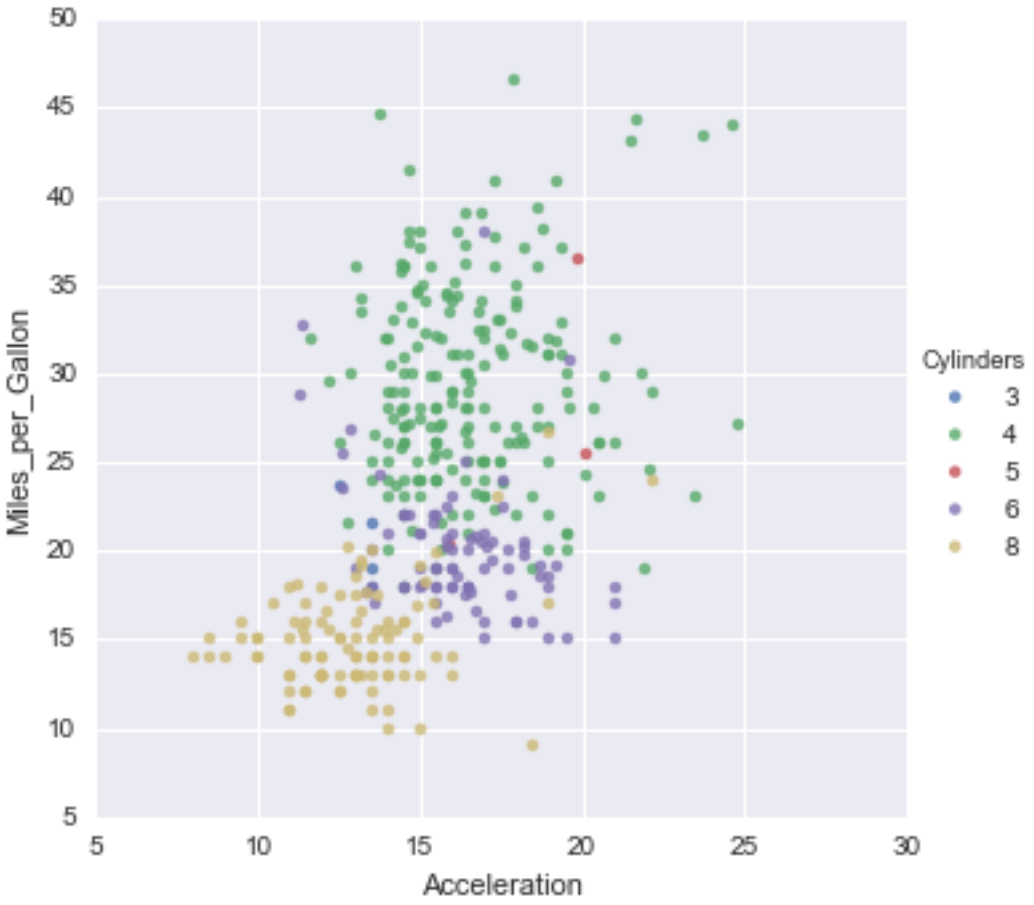


5 Plotting Three or More Variables

- This typically means we are comparing subgroups in our data
- You can use colors (hue) to represent different groups
- You can also plot different groups in different plot side by side
  - Organized in rows, or columns
- The groups are typically categorical variables
  - Consider Subdividing continuous variables if you want to use as groups

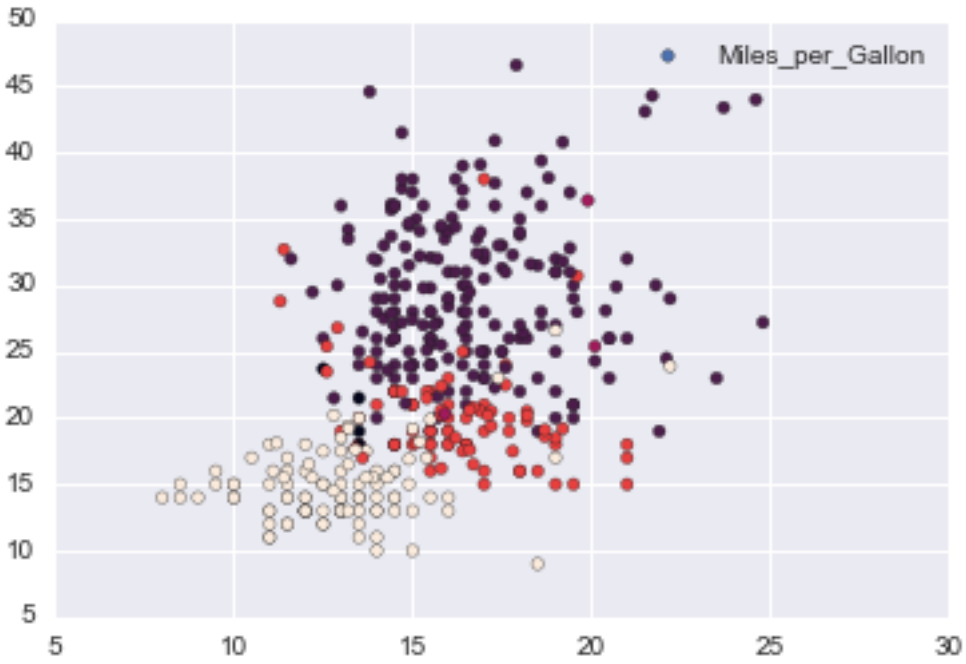
```
In [19]: # Plotting 3 variables, using jue
         sns.lmplot(x='Acceleration', y='Miles_per_Gallon', hue='Cylinders',data=cars_df, fit_re
```

```
Out[19]: <seaborn.axisgrid.FacetGrid at 0x118c5cb70>
```




```
In [69]: # Possible on matplotlib as well
         # Showing legend properly is not easy
         plt.scatter(cars_df.Acceleration, cars_df.Miles_per_Gallon, c=cars_df.Cylinders)
         plt.legend()
```

```
Out[69]: <matplotlib.legend.Legend at 0x11be56780>
```



In [67]:
```
# Same thing with matplotlib
# Notice the legend is missing
colors = {
    2:'r',
    3:'g',
    4:'b',
    5:'y',
    6:'w',
    8:'k',
}
# We draw each cylinder plot separatly
for x in sorted(set(cars_df.Cylinders)):
    d = cars_df[cars_df.Cylinders == x]
    plt.scatter(d.Acceleration, d.Miles_per_Gallon, c=colors.get(x), label=x)
plt.legend(title="Cylinders")
```

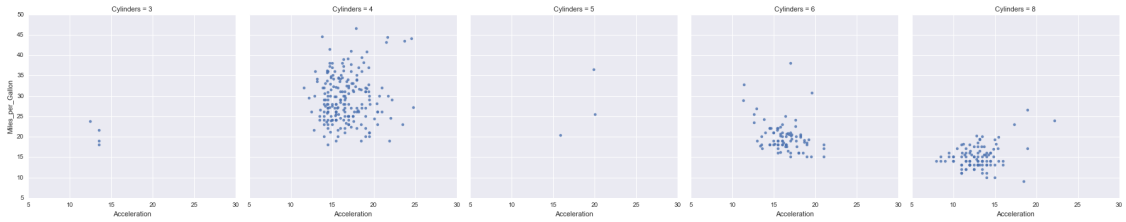
Out[67]:
```
<matplotlib.legend.Legend at 0x11bb848d0>
```

6 Tip

Use seaborn whenever possible

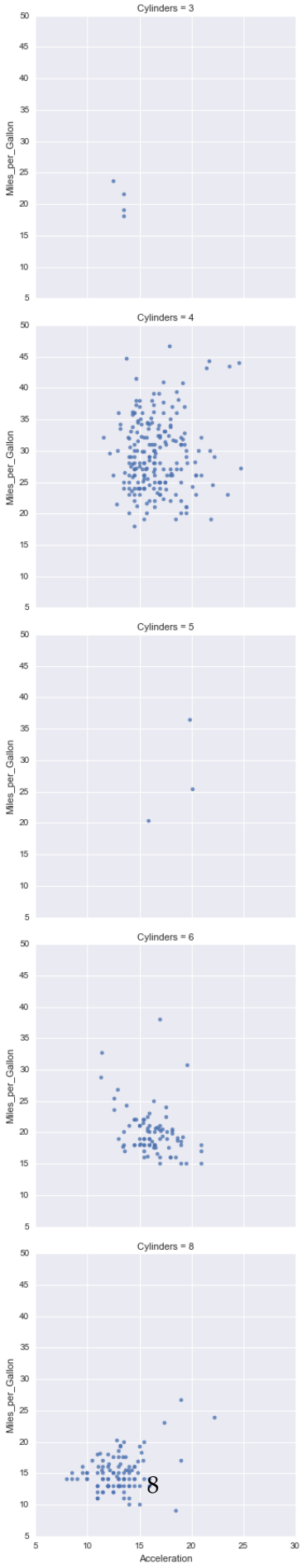
```
In [72]: # See what happens when I replace hue with col
         sns.lmplot(x='Acceleration', y='Miles_per_Gallon', col='Cylinders',data=cars_df, fit_re
```

```
Out[72]: <seaborn.axisgrid.FacetGrid at 0x11c8a5d30>
```




```
In [73]: # now row
         sns.lmplot(x='Acceleration', y='Miles_per_Gallon', row='Cylinders',data=cars_df, fit_re
```

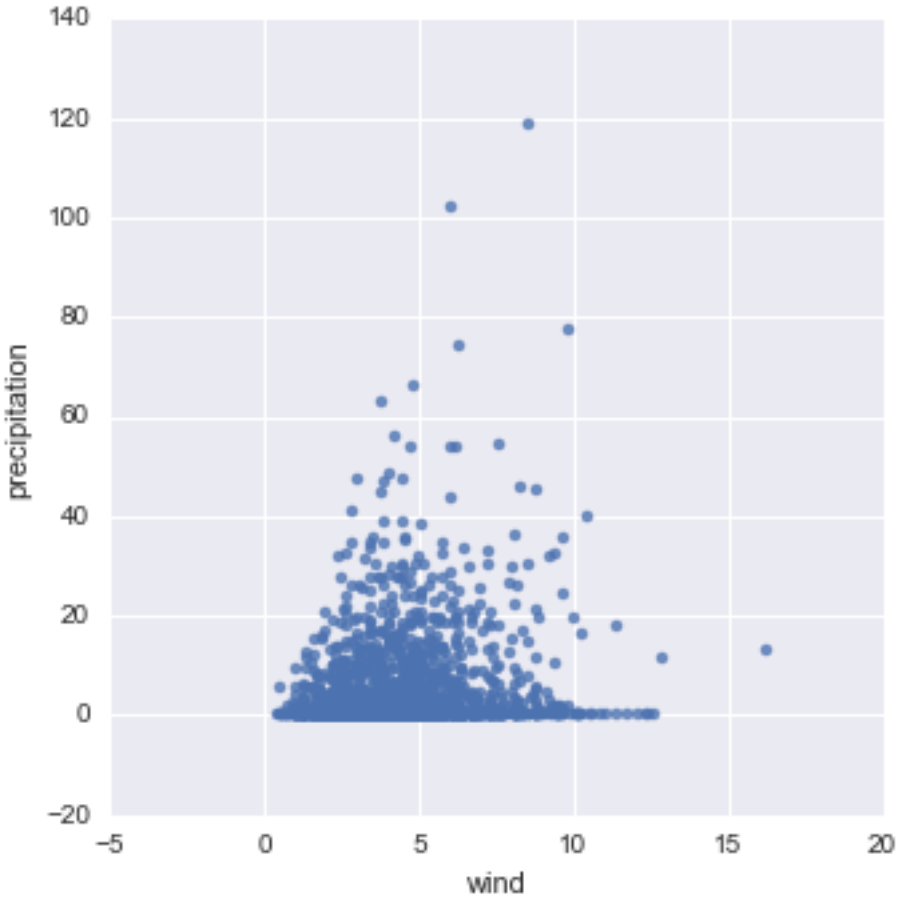
```
Out[73]: <seaborn.axisgrid.FacetGrid at 0x11ce27d30>
```


Cylinders = 3
Cylinders = 4
Cylinders = 5
Cylinders = 6
Cylinders = 8
Miles_per_Gallon
Acceleration

7 Using row/col In Seaborn Plots

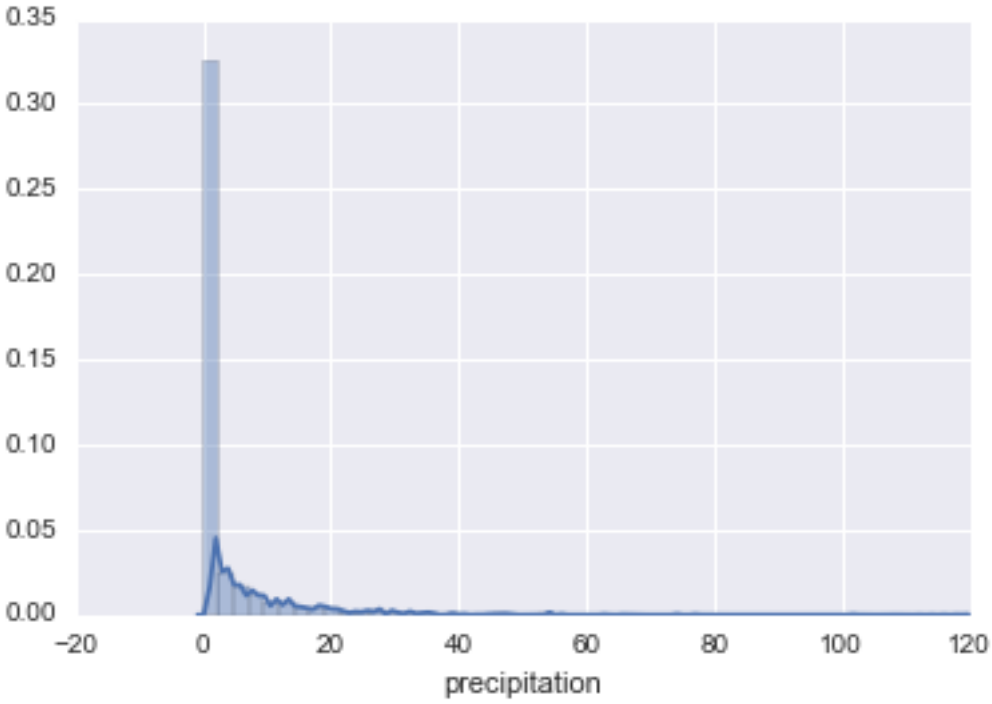
- Avoid using it with variables that have many values
  - Will create many plots
  - Difficult to compare
- Use when variable has few values

In [119]:
```
# Let's examine relationship between wind and percipitation
g = sns.lmplot(x='wind', y='precipitation', data=weather_df, fit_reg=False)
```



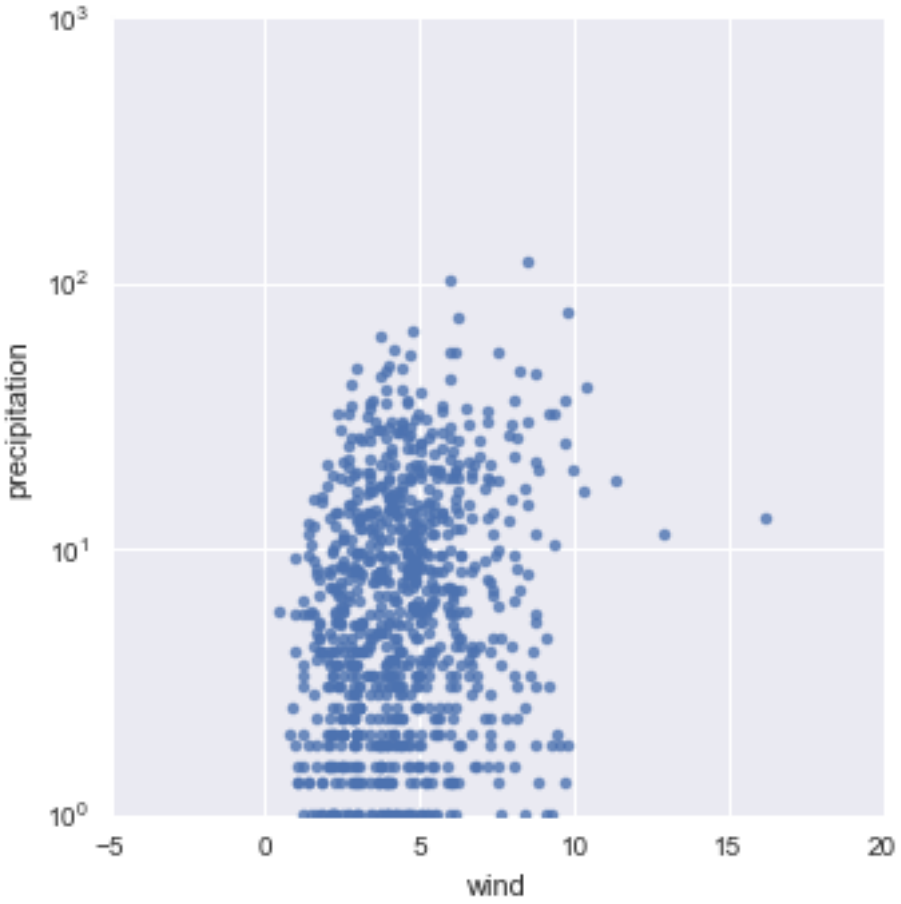
In [120]:
```
# Distribution of percipitation
sns.distplot(weather_df.precipitation)

# The values are bunched up close to zero
```

Out[120]: <matplotlib.axes._subplots.AxesSubplot at 0x120b8b978>



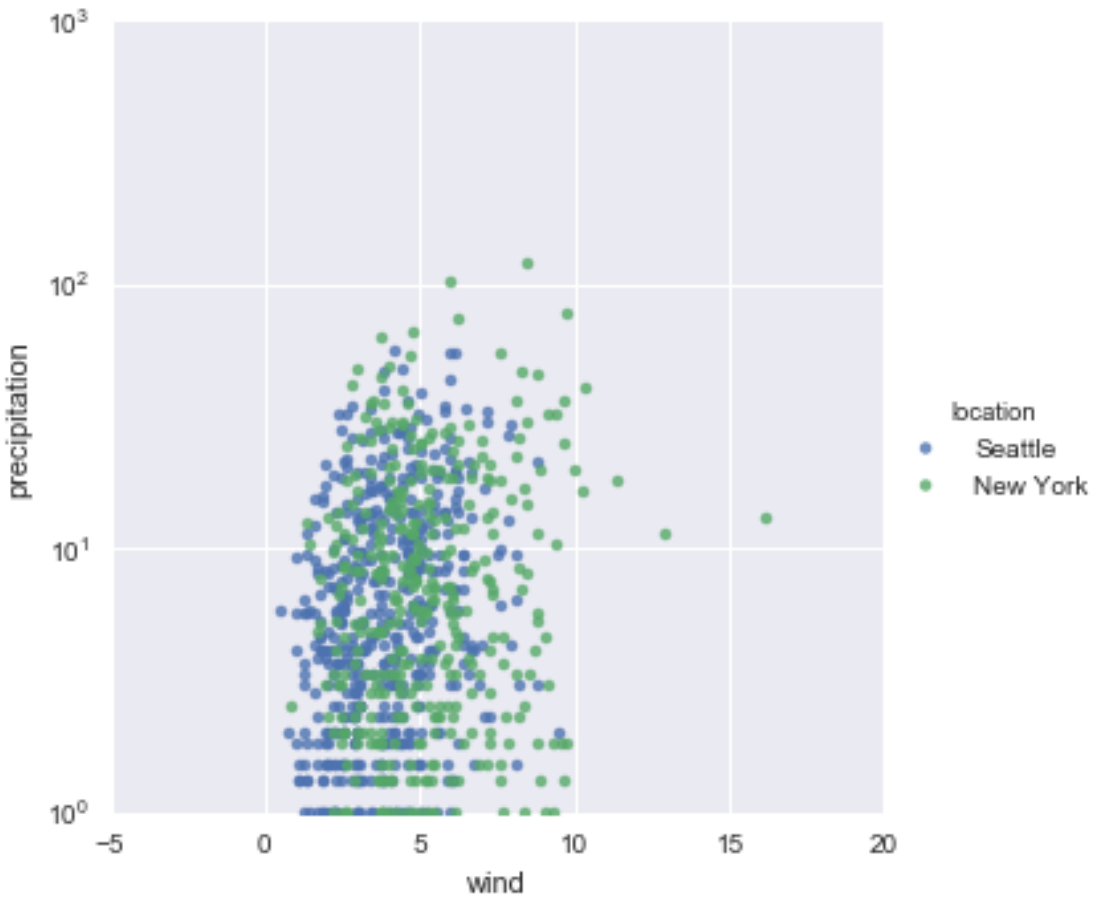
In [121]:
```
# Using log scale on the y access will make the plot clearer
g = sns.lmplot(x='wind', y='precipitation', data=weather_df, fit_reg=False)
g.set(yscale="log")

# No clear relationship
```

Out[121]: <seaborn.axisgrid.FacetGrid at 0x120e752e8>



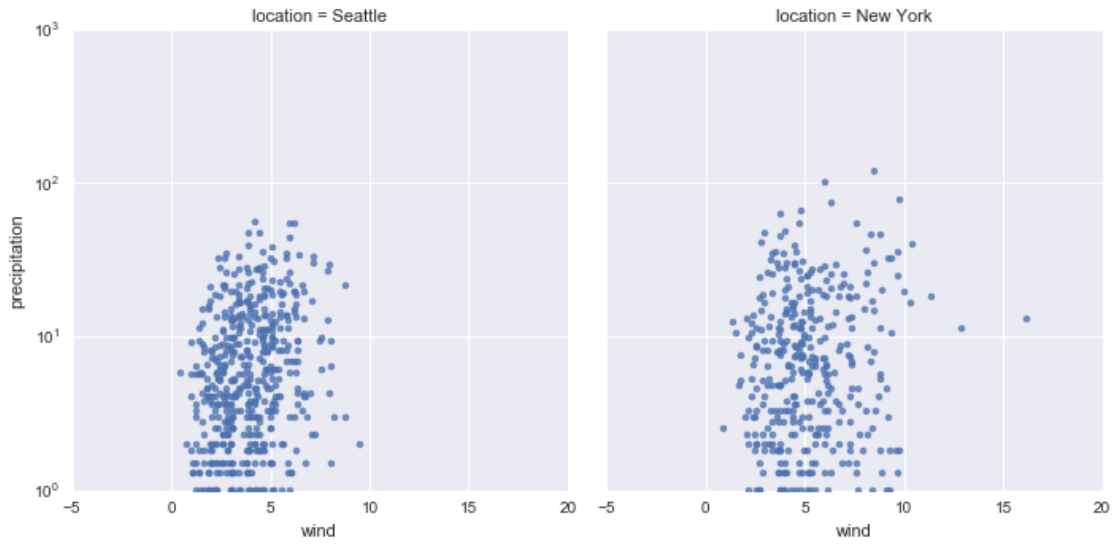
In [123]:
```
# Let's see if it is the same in all locations
g = sns.lmplot(x='wind', y='precipitation', hue='location', data=weather_df, fit_reg=F
g.set(yscale="log")

# Hue not making comparison easy
```

Out[123]: <seaborn.axisgrid.FacetGrid at 0x120e17f28>



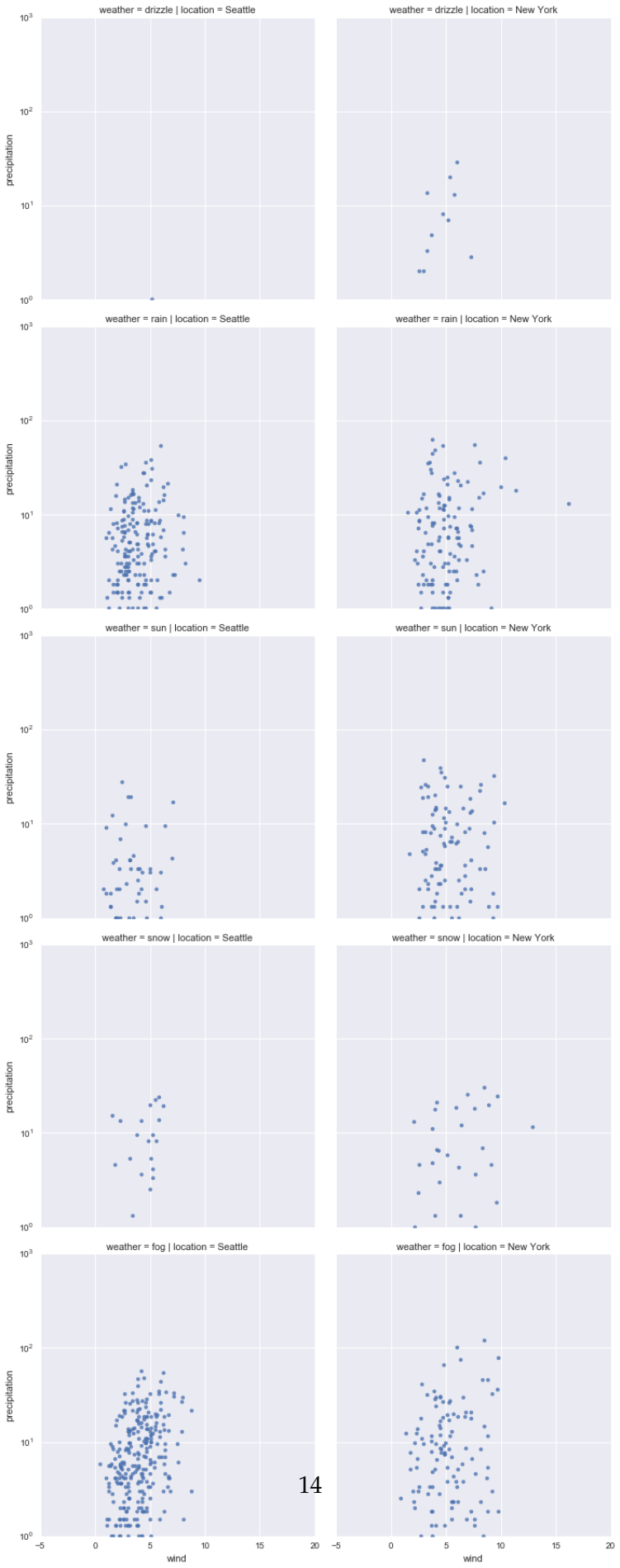
In [124]:
```
# two plots are better for comparison
g = sns.lmplot(x='wind', y='precipitation', col='location', data=weather_df, fit_reg=F
g.set(yscale="log")

# Still no clear relation
```

Out[124]: <seaborn.axisgrid.FacetGrid at 0x1215d0ac8>



In [125]:
```
# Let's also break it based on weather types
g = sns.lmplot(x='wind', y='precipitation', col='location', row='weather', data=weathe
g.set(yscale="log")
```

Out[125]:
```
<seaborn.axisgrid.FacetGrid at 0x11e558860>
```


weather = drizzle | location = Seattle
weather = drizzle | location = New York
weather = rain | location = Seattle
weather = rain | location = New York
weather = sun | location = Seattle
weather = sun | location = New York
weather = snow | location = Seattle
weather = snow | location = New York
weather = fog | location = Seattle
weather = fog | location = New York
precipitation
wind

8 EDA

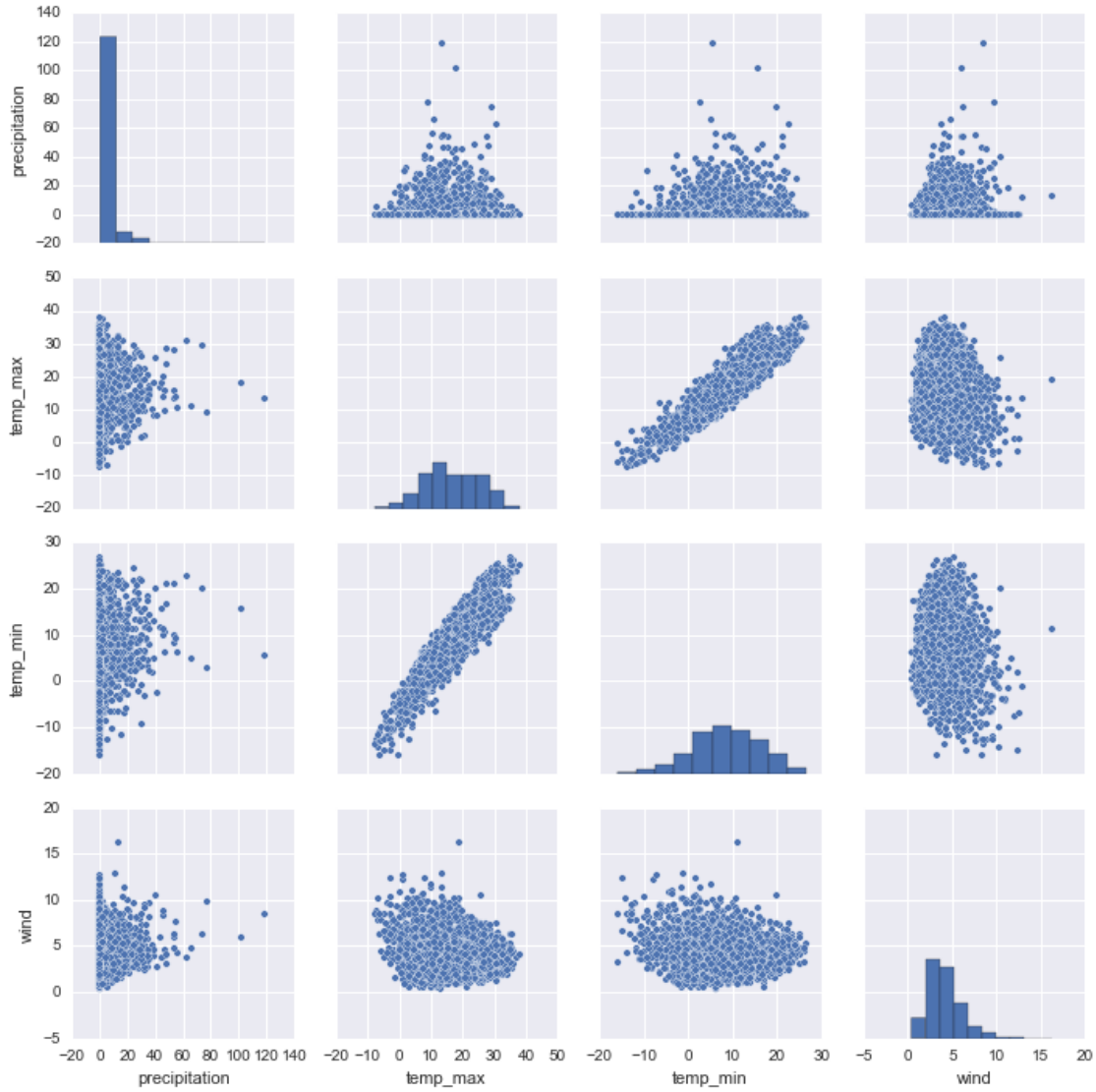
- Remember it is an investigation
- Sometimes our investigation takes as to a dead end
- We think of different ways to break our data
- Possibly rescale our axes, like the log scale
- Think about missing data that we might get
- There might be nothing interesting in the relationship
  - Examine other relationships

9 Pair Plots

- Use to get a quick overview of the numeric data that you have
- Diagonal represents the distributions
- Of diagonals gives you relationships between the variables
- Use to find insights that you can dig deeper into

```
In [2]: sns.pairplot(weather_df)
```

```
Out[2]: <seaborn.axisgrid.PairGrid at 0x105f0a2b0>
```



10 Remember The Univariate Plots in Seaborn?

- Violin, stip, swarm, count, and dist plots
- You can use them for multivariate comparison of distributions
  - You can set both the y and hue to partition your data
- Cannot place them in rows and columns
  - Use them with factor plots

```
In [3]: # Try to plot 2 different plots of each (10 plot in total)
        # where you set the y and then the hue to see how the
        # plots will behave differently
```

11 Factor Plots

- Use it to further breakdown the distributional plots
- Allows you to place them in rows and columns as well
- Examine documentation for factor plot and attempt to plot 2 different distributional plots in rows or columns for comparison

```
In [126]: # Your turn to show off FactorPlot
```