NYC DATA SCIENCE ACADEMY

Introduction to Python: Seaborn

NYC Data Science Academy

OVERVIEW

- Seaborn overview
- Univariate plots
- Bivariate plots
- Multivariate plots
- Advanced plots

Visualization in Python

- This course covers two parts: matplotlib and seaborn. This slide covers the package Seaborn.

OVERVIEW

- Seaborn overview
  - Univariate plots
  - Bivariate plots
  - Multivariate plots
  - Advanced plots

Seaborn overview

- Seaborn is a Python visualization library based on matplotlib. It provides a high-level interface for drawing attractive statistical graphics. Here is the documentation.

Seaborn overview

- We first import the packages matplotlib.pyplot and visualize with it.

```
%pylab inline
import matplotlib.pyplot as plt
```

- The package Pandas is imported for data manipulation, Numpy and SciPy is imported to generate random numbers and distribution.

```
from scipy import stats
import pandas as pd
import numpy as np
```

Seaborn overview

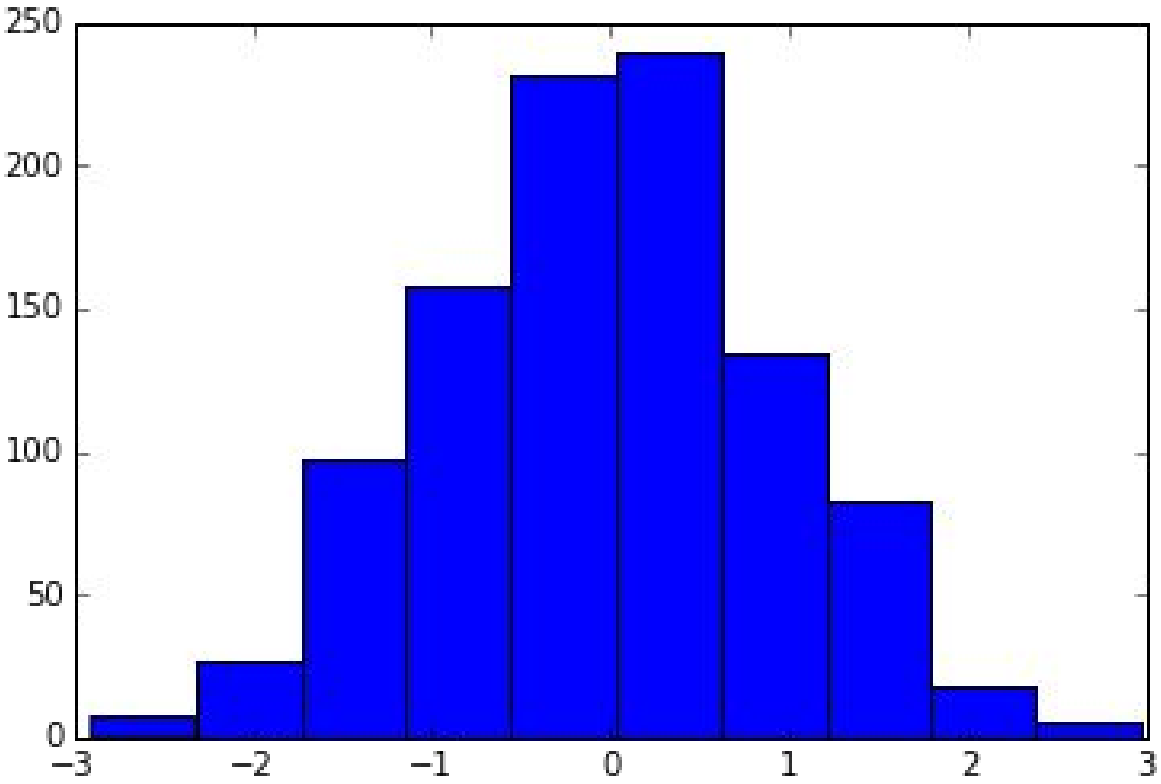
- The code below visualize a normal distribution.

```
n = 1e3
df = pd.DataFrame({'norm': stats.norm.rvs(size = n),
                   'binomial': stats.binom.rvs(10, 0.5, size = n),
                   'poisson': stats.poisson.rvs(5, size = n),
                   't': stats.t.rvs(30, size = n)})

plt.hist(df.norm)
```

Seaborn overview

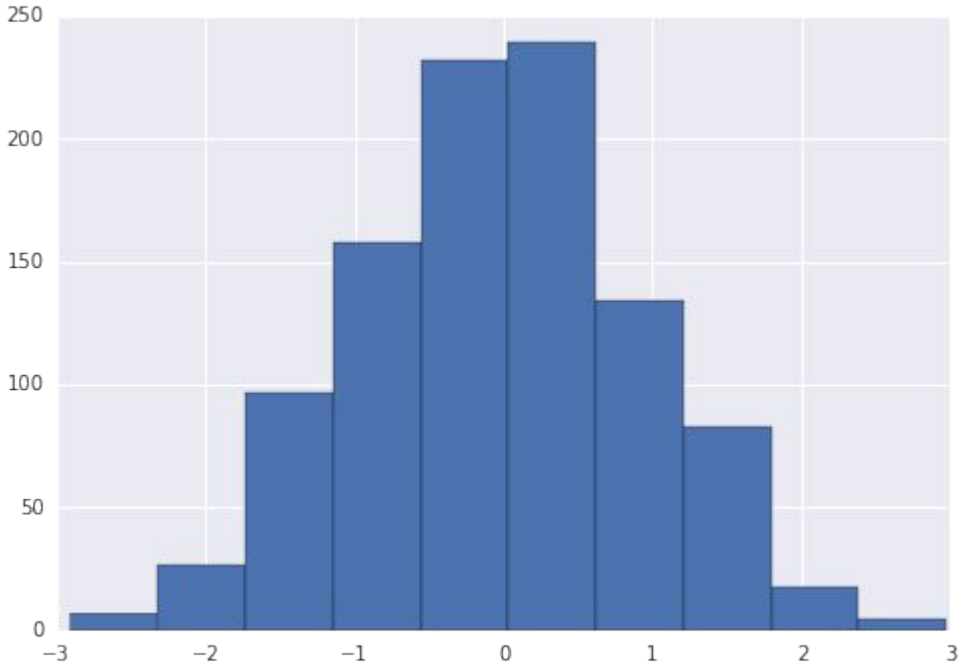
- The plot obtained:



Seaborn overview

- Import seaborn and visualize again:

```
import seaborn as sns
plt.hist(df.norm)
```



OVERVIEW

- Seaborn overview
- Univariate plots
- Bivariate plots
- Multivariate plots
- Advanced plots

Univariate plot

- We generate some arrays that we will visualize.

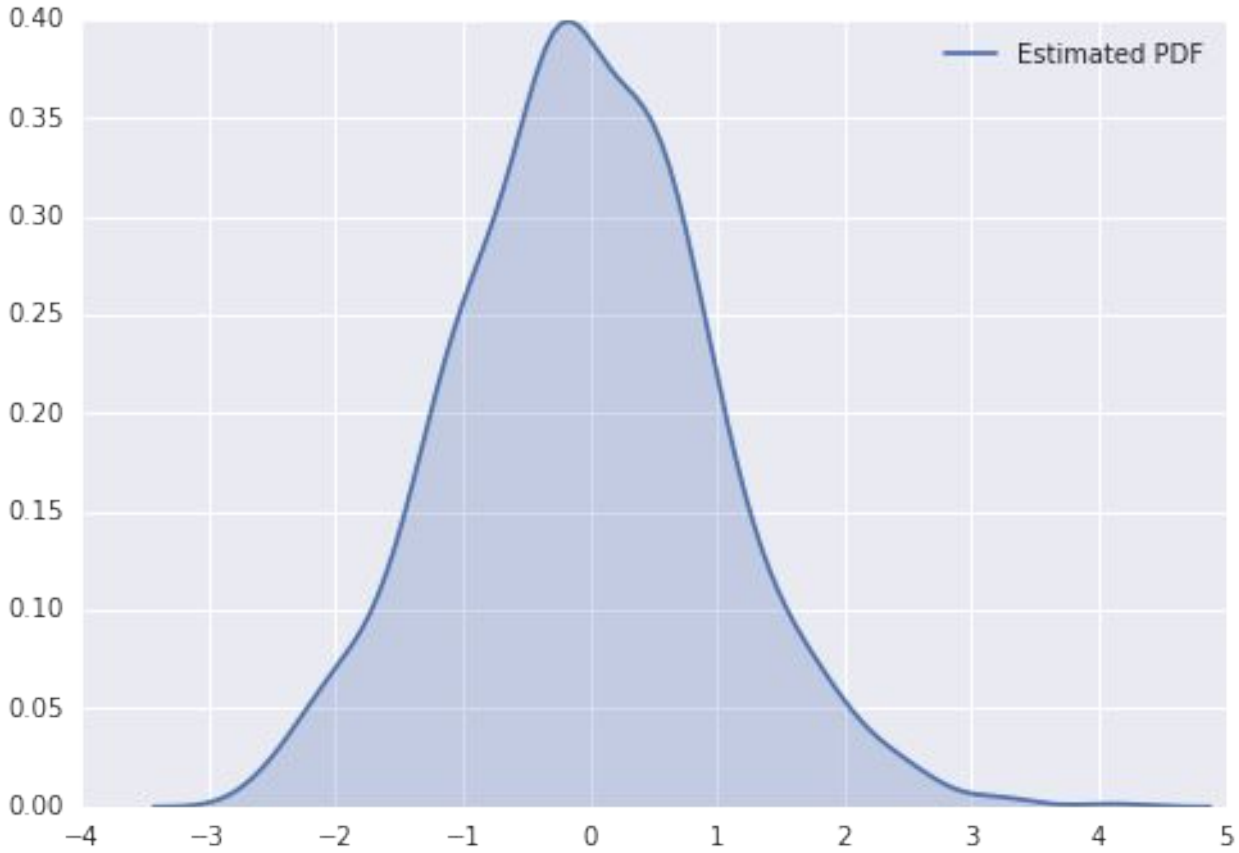
```
np.random.seed(2)
x = np.random.randn(1e3)
y = 2*x + np.random.rand(1e3)
```

- We can of course visualize the distribution of x with histogram. However, seaborn provides a nice function that smooths out the histogram to estimate the distribution.

```
sns.kdeplot(x, shade=True, label='Estimated PDF')
```

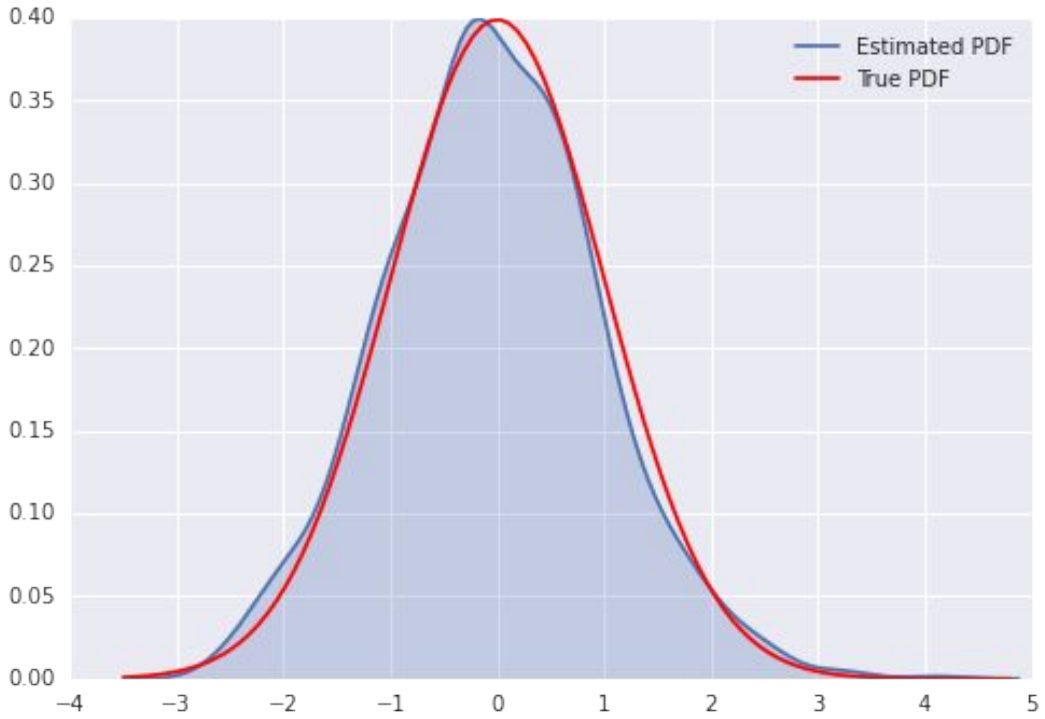
Univariate plot

- The plot obtained:



Exercise

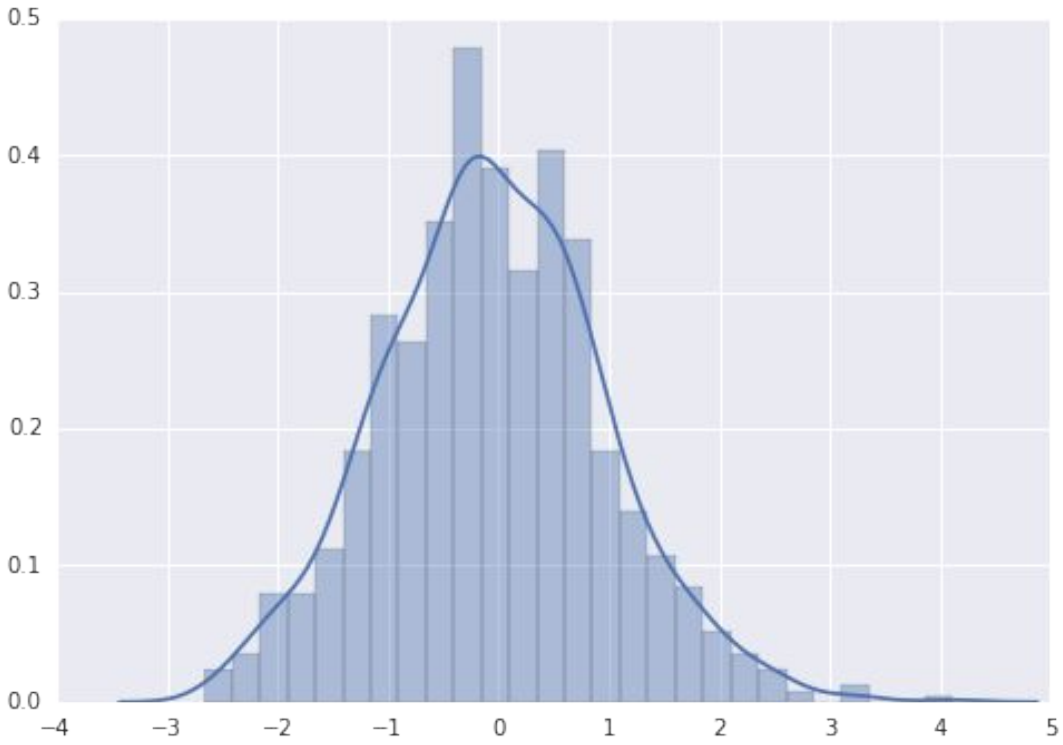
- Modify the code above to obtain the plot below:



Univariate plot

- It is possible to combine histogram and the distribution estimate plot:

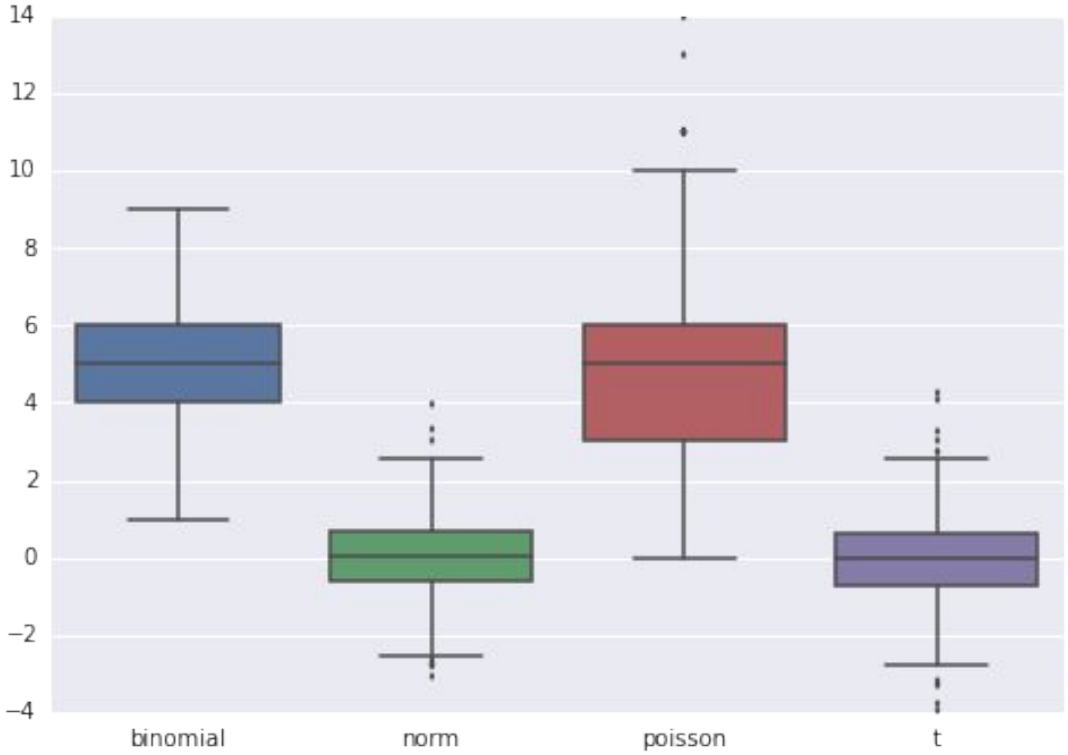
```
sns.distplot(x)
```



Univariate plot

- We may also visualize the distribution of multiple features in one plot:

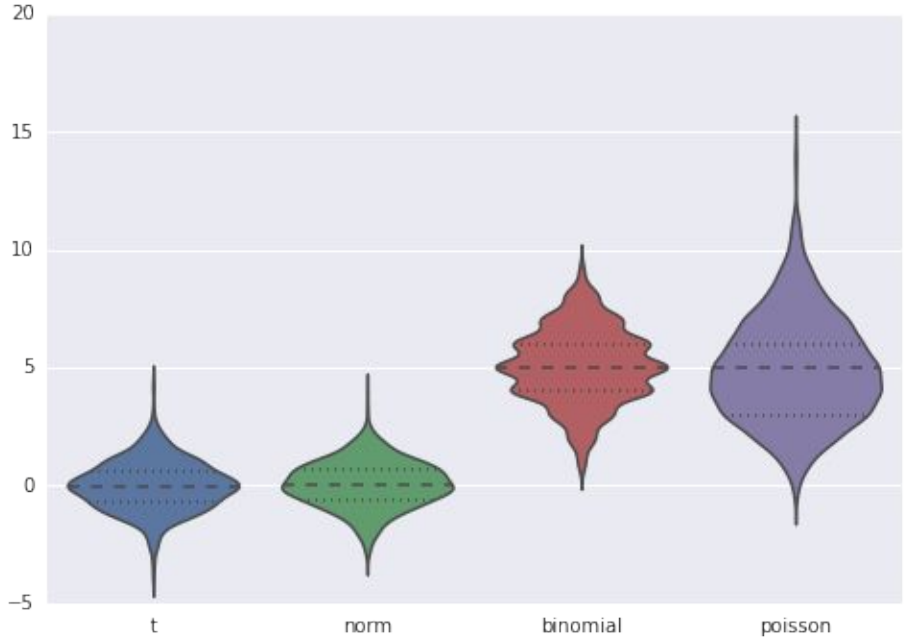
```
sns.boxplot(df)
```



Univariate plot

- We see that seaborn understand data frames. Here we also demonstrate how to order columns in a data frame.

```
df = pd.DataFrame(df, columns=['t', 'norm',\
                               'binomial', 'poisson'])
sns.violinplot(df)
```



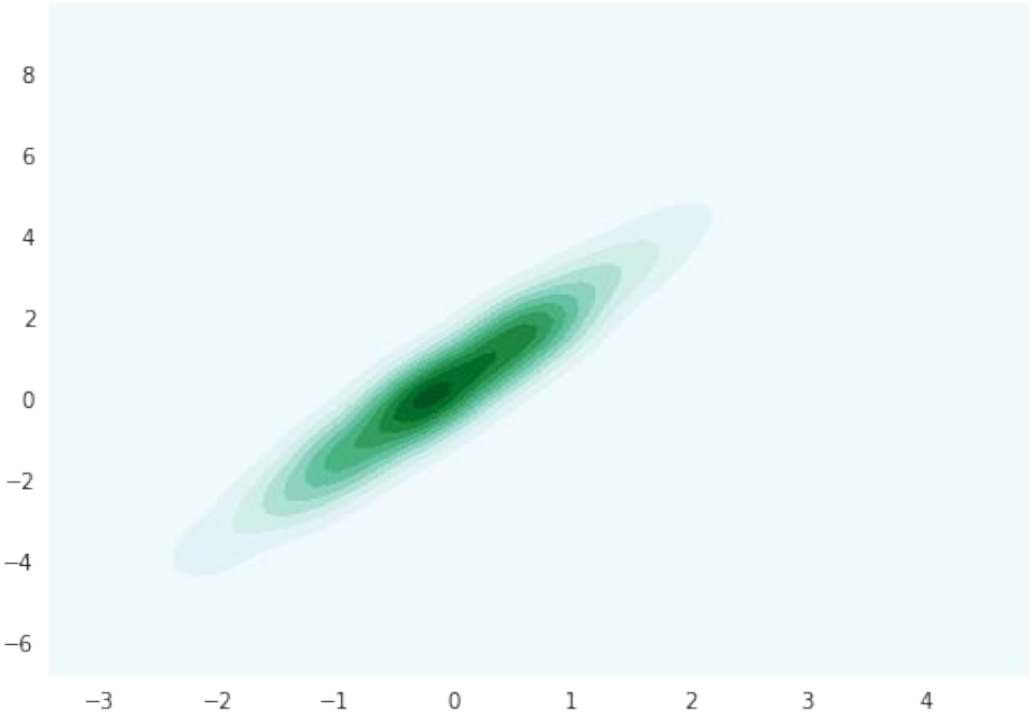
OVERVIEW

- Seaborn overview
- Univariate plots
- **Bivariate plots**
- Multivariate plots
- Advanced plots

Bivariate plot

- The `kdeplot()` function can be used to plot 2D density.

```
sns.kdeplot(x, y, shade=True)
```



Exercise

- Sketch the 2D density plot for the normal random sample and the binomial random sample in our df.

Bivariate plot

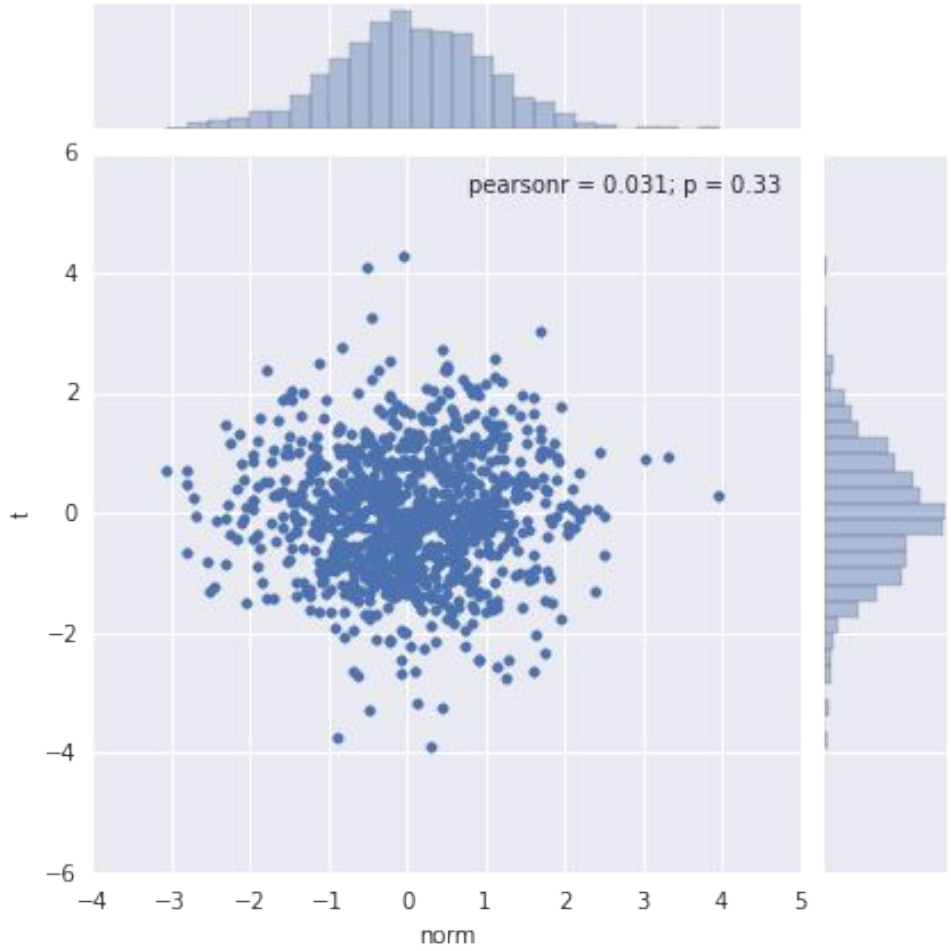
- The `joinplot()` function combines histogram and scatter plot.

```
sns.jointplot(df.norm, df.t)
```

- Below we see "pearsonr=0.031" which indicates the pearson correlation of these two variables. We also see "p=0.33", which indicates that there is no significant linear relation between the two variables.

Bivariate plot

- The plot obtained:



Exercise

- Visualize the variables x and y we created in the previous slide. Explain the pearsonr and p for this case.
- Visualize with the same function as above. This time specify kind=" hex", kind="reg" and kind="kde", respectively.

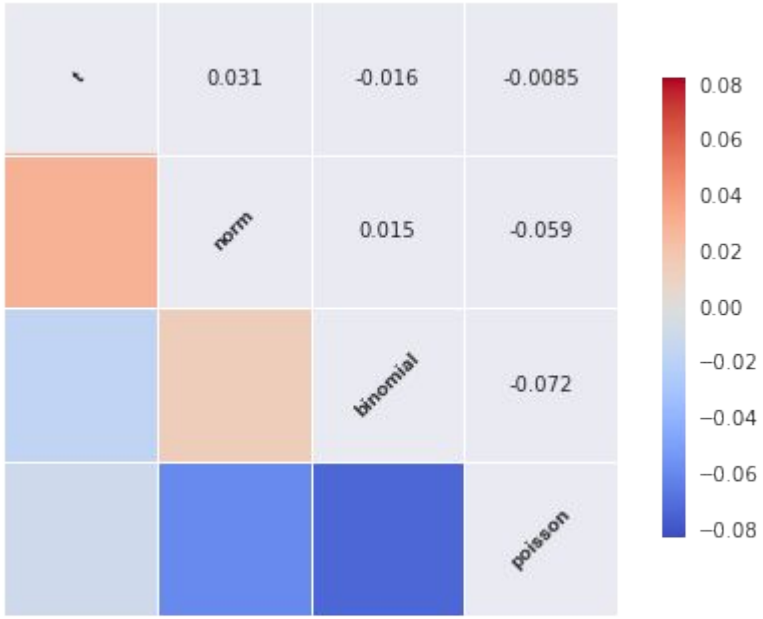
OVERVIEW

- Seaborn overview
- Univariate plots
- Bivariate plots
- **Multivariate plots**
- Advanced plots

Multivariate plot

- A scatter plot or a joint plot just show the correlation of two variables, corrplot is similar to scatter-matrix, it shows the correlations between multiple variables instead of scatters.

```
sns.corrplot(df)
```



Multivariate plot

- Seaborn also provides sample datasets.

```
titanic = sns.load_dataset("titanic").dropna()
```

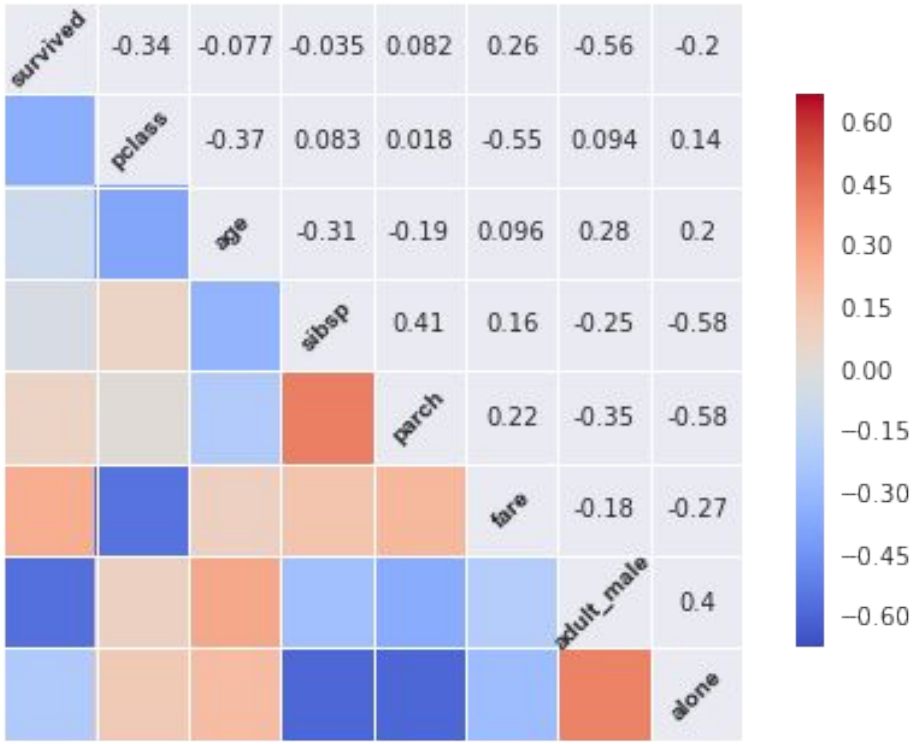
- This is a famous dataset which was originally used to demonstrate a classification problem: using the record of each passengers in the ship to predict the probability that he or she survived. Run the code below to see the data frame.

```
titanic.head()
```

Multivariate plot

- Plot the correlation:

```
sns.corrplot(titanic)
```



OVERVIEW

- Seaborn overview
- Univariate plots
- Bivariate plots
- Multivariate plots
- Advanced plots

Advanced plot

- Seaborn also provides another famous sample dataset:

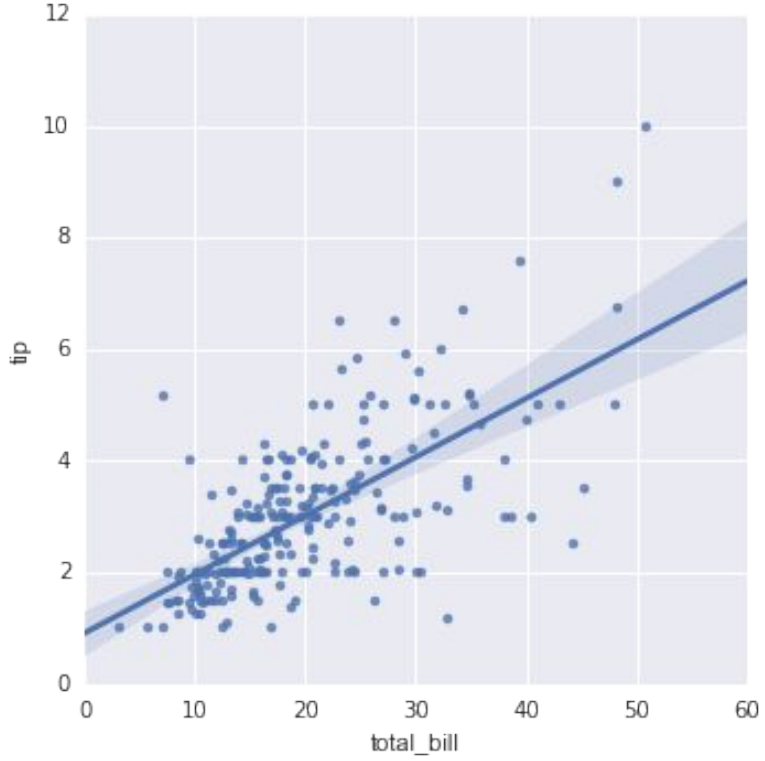
```
tips = sns.load_dataset("tips")
tips.head()
```

- This is a famous dataset record the information of people of different sex, being a smoker or not, visiting at different day or for different meal, the size of parties they joined, and the tip and the total bill they pay.

Advanced plot: fitting

- To visualize the relation between total_bill and tip, of course we can use scatter plot. However, seaborn allows fitting with linear model as well.

```
sns.lmplot("total_bill", "tip", tips)
```



Advanced plot: fitting

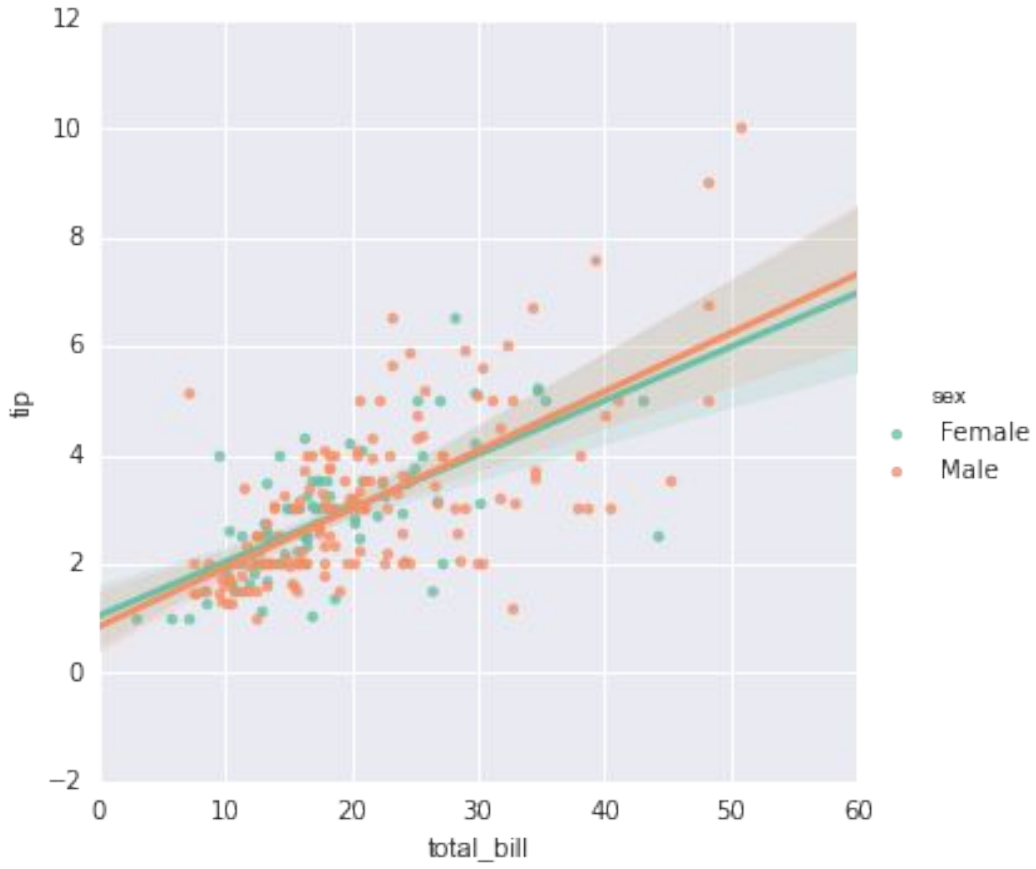
- We can further split the data into "Male" and "Female" parts and visualize them.

```
sns.lmplot("total_bill", "tip", tips,\
           hue="sex", palette="Set2");
```

- Here we used:
  - hue indicates according to which column we group our data.
  - palette simply specifies the color we want to use.

Advanced plot: fitting

- The plot obtained:



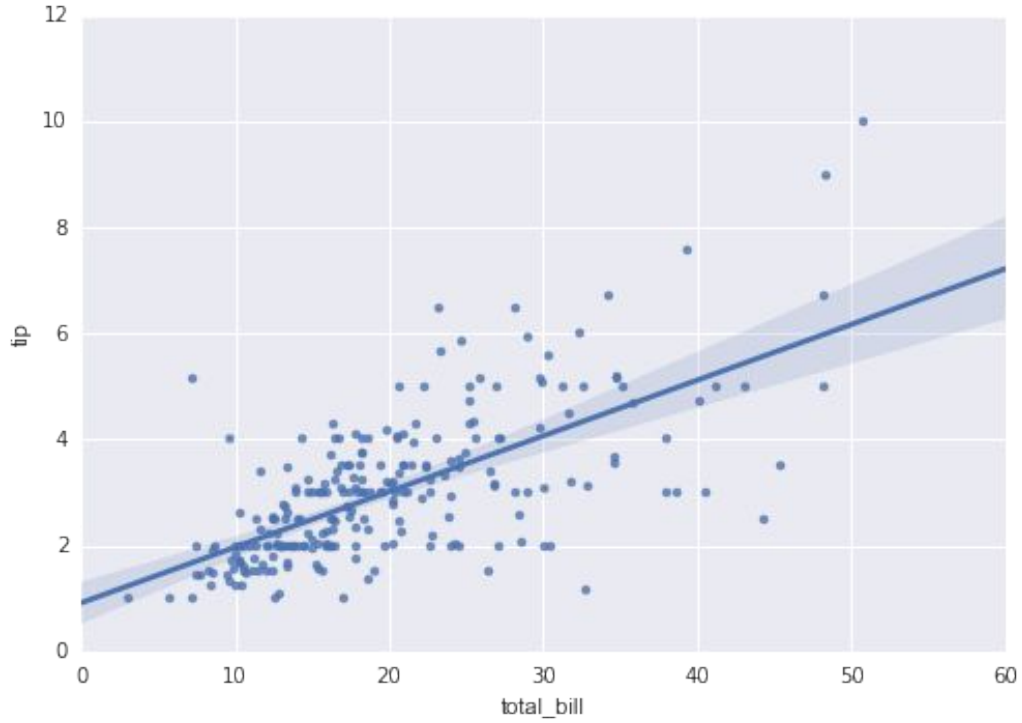
Exercise

- Make the same visualization as in the previous slide. This time group the data according to that if the observation is a smoker or not.

Advanced plot: fitting

- lmplot only supports data frames, regplot also accepts data passed directly as numpy arrays or pandas series objects.

```
sns.regplot(tips.total_bill, tips.tip)
```



Advanced plot: facet

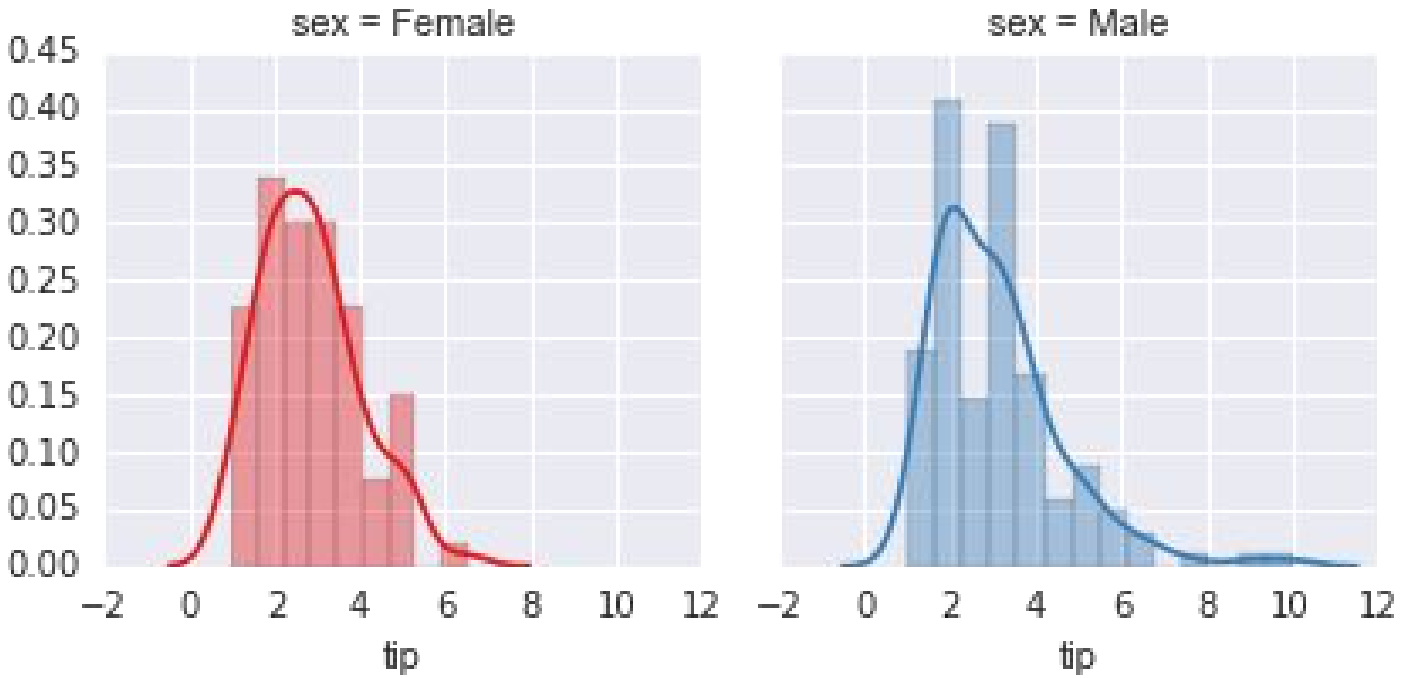
- We often need to compare the same kind of plot for different features. Functions for faceting comes in handy.

```
SexGrid = sns.FacetGrid(tips, col='sex',\
                        hue="sex", palette="Set1")
SexGrid.map(sns.distplot, "tip");
```

- Within the function `FacetGrid()`, hue indicates sketching a separate plot for each sex; col specifies that each plot is placed in a new column; palette specifies the colors. Then we use `.map()` method to specify the type of the plot and the feature we want to visualize.

Advanced plot: facet

- The plot obtained:



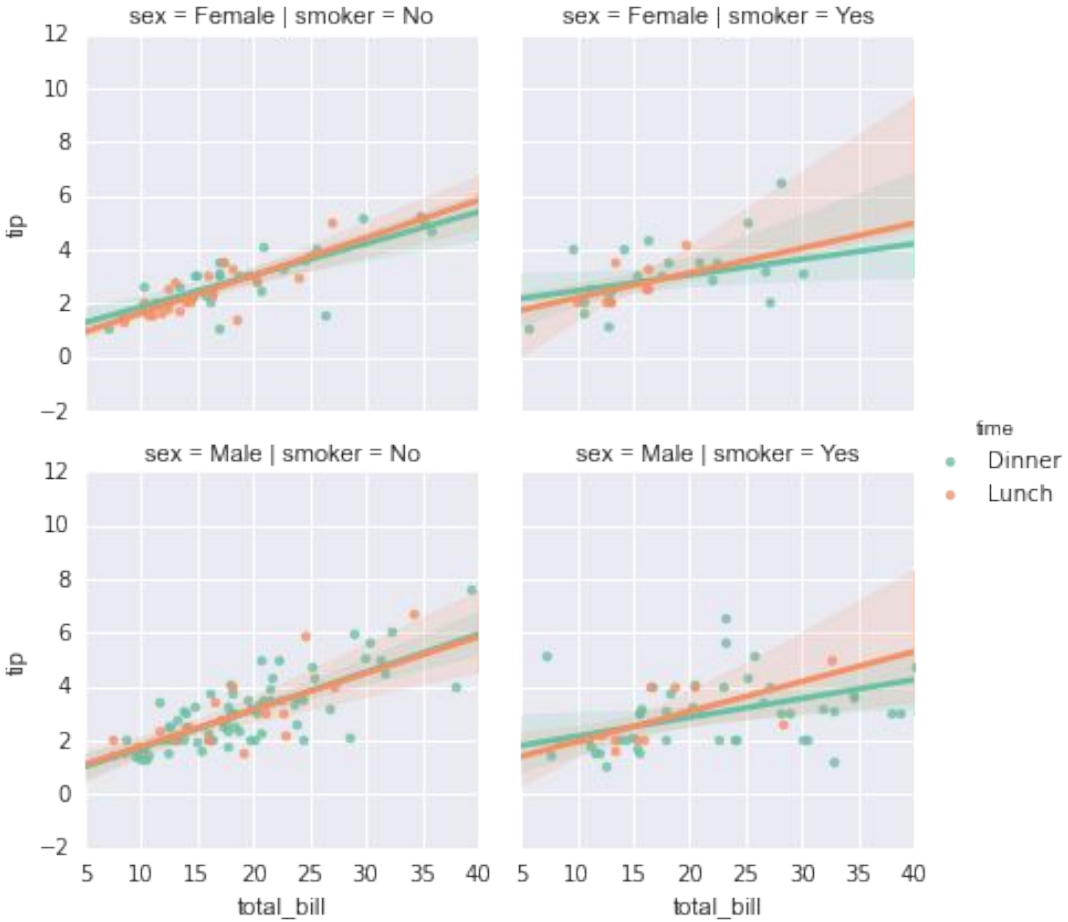
Advanced plot: facet

- Facet can be used to differentiate multiple factors as well.

```
tipsGrid = sns.FacetGrid(tips, row='sex', col='smoker',\
                                 hue='time', palette="Set2")
tipsGrid.map(sns.regplot, 'total_bill', 'tip')
tipsGrid.add_legend()
```

Advanced plot: facet

- The plot obtained:



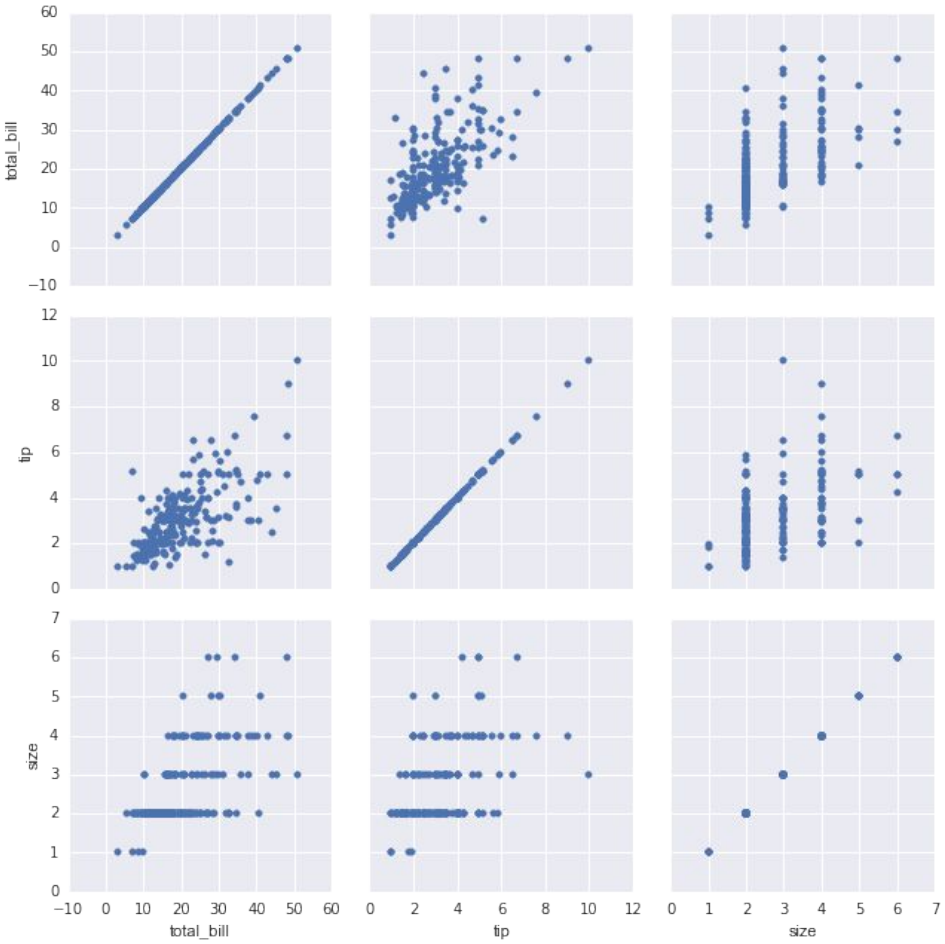
Advanced plot: pairwise plot

- The function FacetGrid help you explore the specific variables conditioned on different levels. Function PairGrid is useful to explore the relationships between pairs of variables.

```
tipGrid = sns.PairGrid(tips)
tipGrid.map(plt.scatter)
```

Advanced plot: pairwise plot

- The plot obtained:



Exercise 5

- Download the iris data frame built in the seaborn package.

```
iris = sns.load_dataset("iris")
iris.head()
```

- Then pair the four numeric features and sketch scatter plot for each pair.

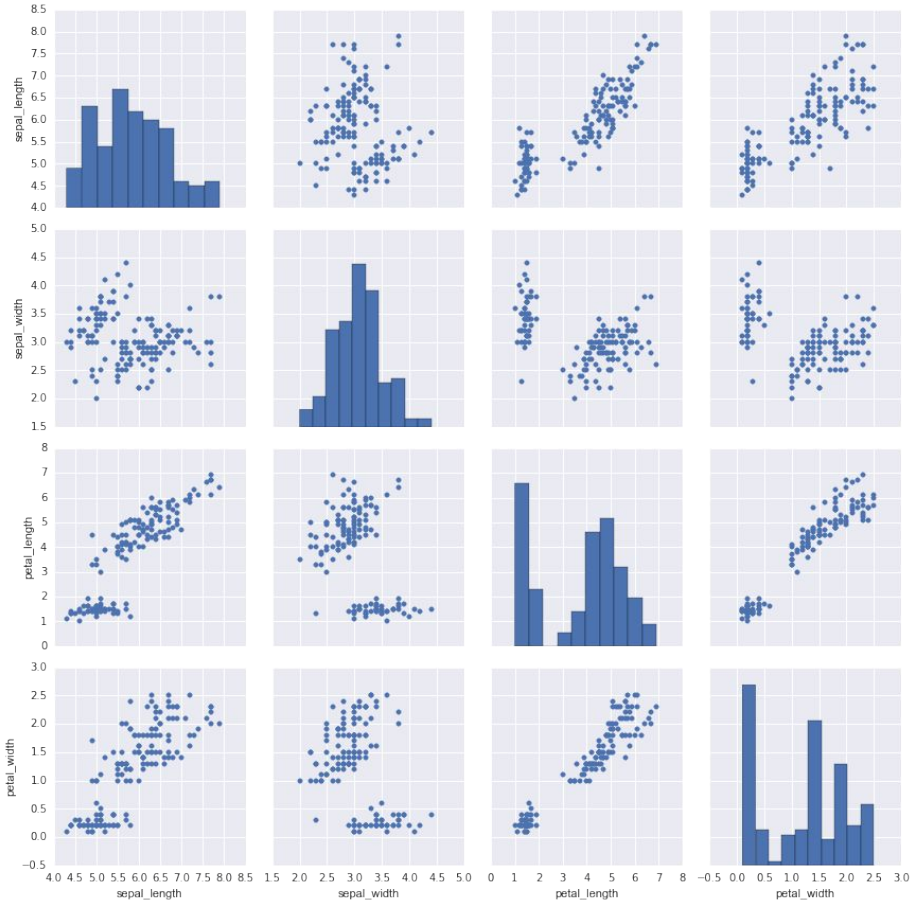
Advanced plot: More flexibility

- Some examples with different type of plot and colors.

```
g = sns.PairGrid(iris)
g.map_diag(plt.hist)
g.map_offdiag(plt.scatter)
```

Advanced plot: More flexibility

- The plot obtained:



Advanced plot: More flexibility

- Some examples with different type of plot and colors.

```
g = sns.PairGrid(iris, hue = 'species', palette='Set2',\
          hue_kws={'cmap':['Greens','Oranges','Blues']})
g.map_diag(plt.hist)
g.map_upper(plt.scatter)
g.map_lower(sns.kdeplot)
g.add_legend()
```

Advanced plot: More flexibility

- The plot obtained: