0.1_Pandas_and_Seaborn

January 15, 2018

1 Exploring Data with Python

GOALS

- Introduce basic functionality of Pandas DataFrame
- Use Seaborn to explore datasets
- Understand how to produce different forms of reports with Jupyter notebook
- Investigate World Development Indicators with Pandas, Seaborn, and StatsModels packages

First, we investigate `Pandas` and `Seaborn` using some built in datasets. Next, we move to external data with an existing API for `Pandas`. We complete the work by preparing a brief presentation with the Jupyter notebook.

1.0.1 Libraries and Jupyter Notebook

To begin, it's important to understand how the notebook and Python works. We will not be writing all our functions from scratch. There are powerful existing libraries that we can make use of with ready made functions that can accomplish most everything we'd want to do. When using a library with Python, we will first import it. Many times, there are also standard ways of abbreviating the functions.

When we want to use one of the libraries, we call refer to it based on the abbreviation we've offered. For example, to load a dataset called "tips" from the Seaborn package, we write

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

Here, we are calling something from the Seaborn package (`sns`), it is the `load_dataset` function, and the dataset we want it to load is contained in the parenthesis.`("tips")`

```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np
        import pandas as pd
        import seaborn as sns

In [2]: tips = sns.load_dataset("tips")

In [3]: tips.to_csv('data/tips.csv')
```

1.0.2 Pandas Dataframe

The Pandas library is the standard Python data structure library. A `DataFrame` is an object similar to that of an excel spreadsheet, where there is a collection of data arranged in rows and columns. The datasets from the `Seaborn` package are loaded as `Pandas DataFrame` objects. We can see this by calling the `type` function. Further, we can investigate the data by looking at the first few rows with the `head()` function.

This is an application of a function to a pandas object, so we will write

```
tips.head()
```

If we wanted a different number of rows displayed, we could input this in the `()`. Further, there is a similar function `tail()` to display the end of the `DataFrame`.

```
In [4]: type(tips)

Out[4]: pandas.core.frame.DataFrame

In [5]: tips.head()

Out[5]:    total_bill   tip     sex smoker  day    time  size
        0       16.99  1.01  Female     No  Sun  Dinner     2
        1       10.34  1.66    Male     No  Sun  Dinner     3
        2       21.01  3.50    Male     No  Sun  Dinner     3
        3       23.68  3.31    Male     No  Sun  Dinner     2
        4       24.59  3.61  Female     No  Sun  Dinner     4

In [6]: tips["total_bill"].head()

Out[6]: 0    16.99
        1    10.34
        2    21.01
        3    23.68
        4    24.59
        Name: total_bill, dtype: float64

In [7]: tips["tip"].mean()

Out[7]: 2.9982786885245902
```

As shown above, we can refer to specific elements of a DataFrame in a variety of ways. For more information on this, please consult the Pandas Cheatsheet here. Use this to answer the following questions.

PROBLEMS: SLICE AND DICE DATAFRAME

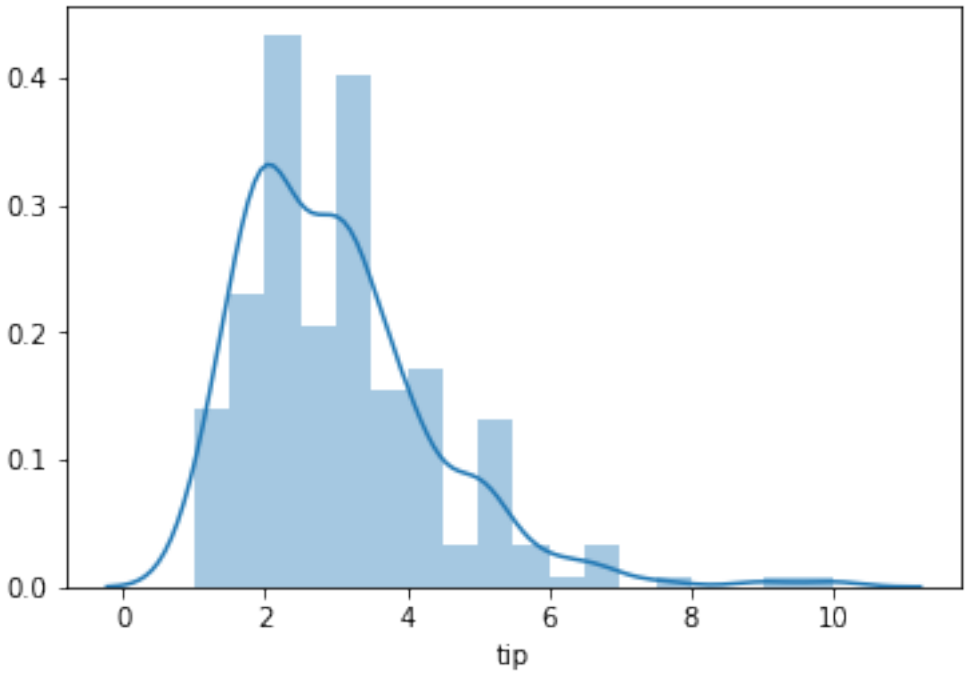
1. Select Column
2. Select Row
3. Boolean Select
4. Groupby
5. Pivot

1.0.3 Plotting with Seaborn

To begin, we will explore the distribution of one variable. For the tips example, this allows us to both look at the distribution of **quantitative** variables like total bill and tip, but also to break these apart based on **categories** like smoker or sex. These are fundamental processes for exploring a dataset, and we aim to use these plotting strategies to understand how different categories compare within a quantitative variable.

```
In [8]: sns.distplot(tips["tip"])
```

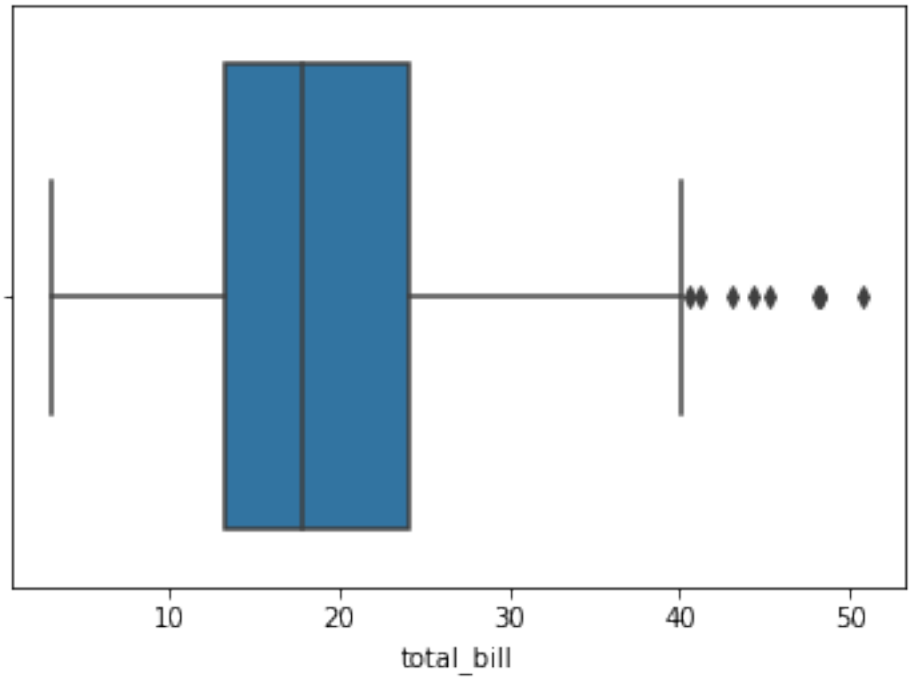
```
Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0x1a0c801fd0>
```




```
In [9]: sns.distplot?
```

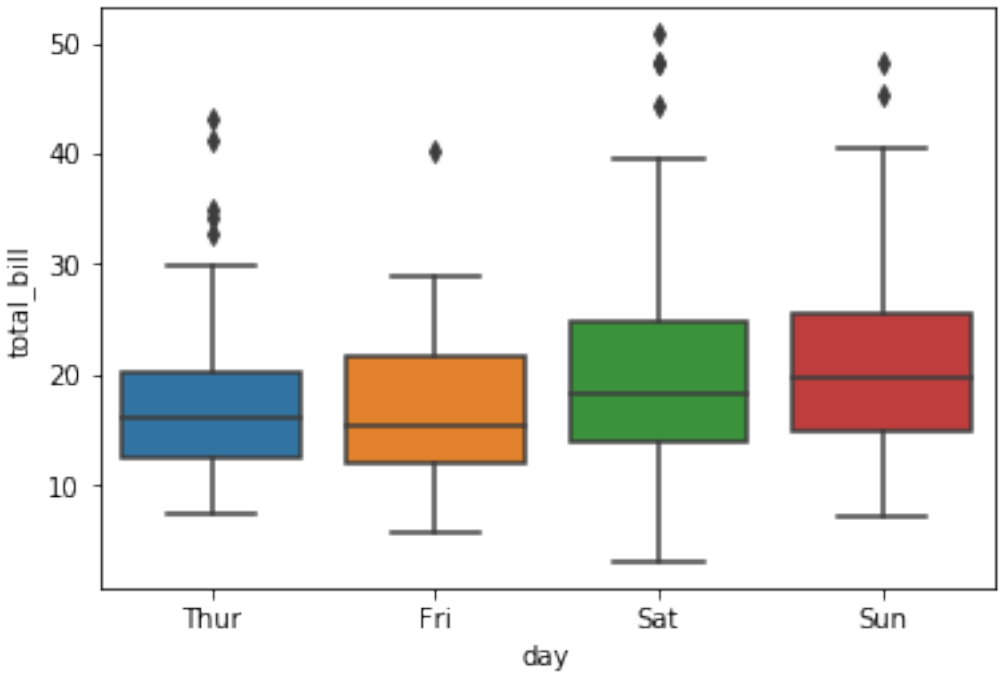
```
In [10]: sns.boxplot(x = tips["total_bill"])
```

```
Out[10]: <matplotlib.axes._subplots.AxesSubplot at 0x1a14d42a90>
```



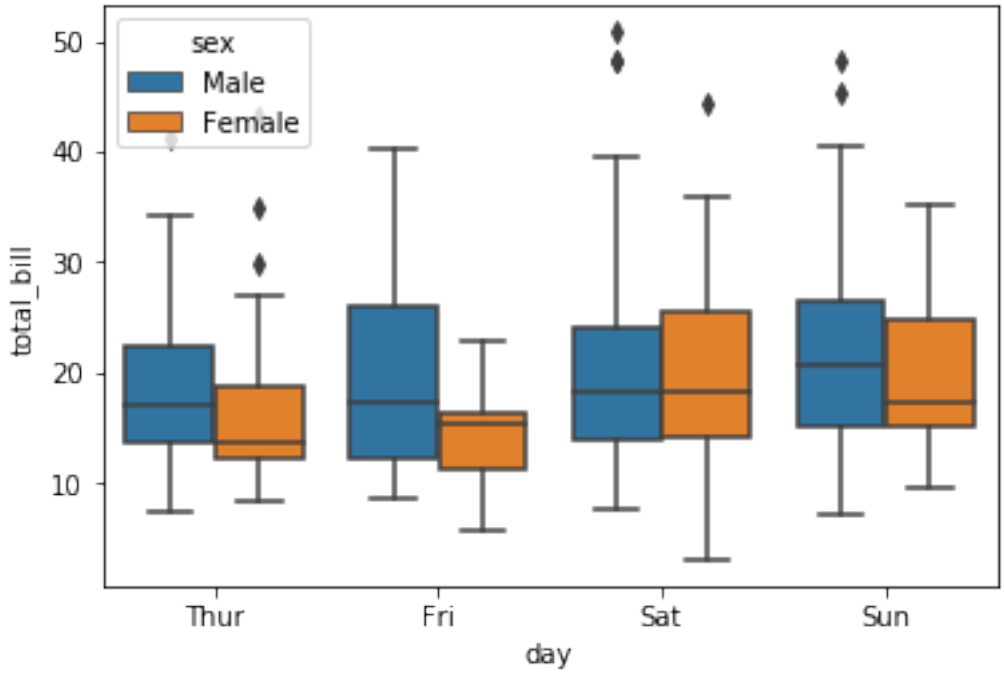
In [11]:
```
sns.boxplot(x = "day", y = "total_bill", data = tips)
```

Out[11]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a14e73a58>
```



```
In [12]: sns.boxplot(x = "day", y = "total_bill", hue = "sex", data = tips)
```

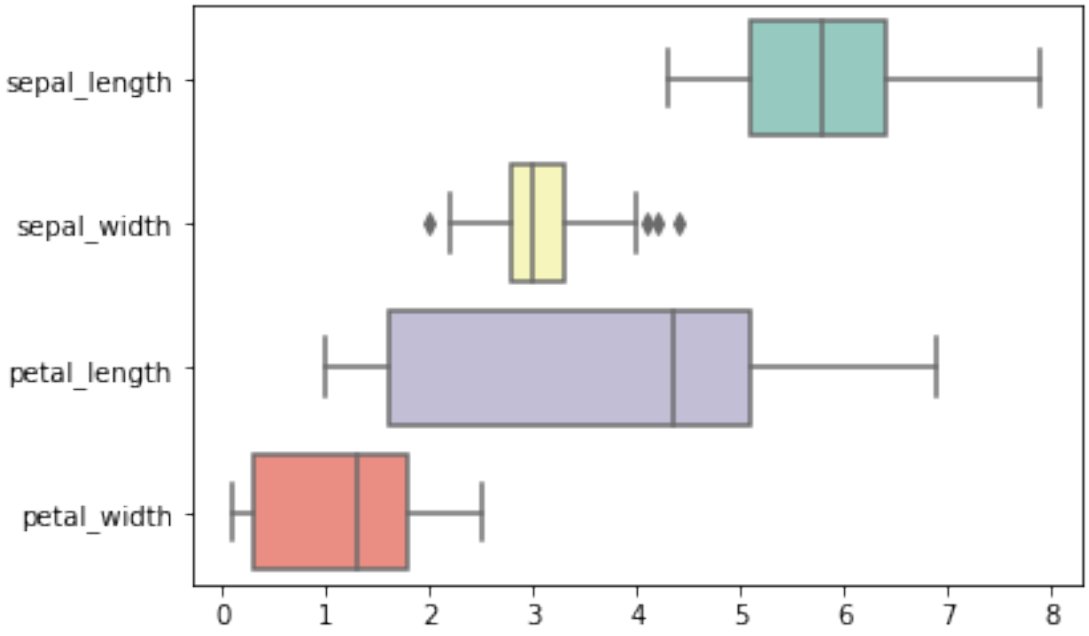
```
Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x1a14e84160>
```




```
In [13]: iris = sns.load_dataset("iris")
```

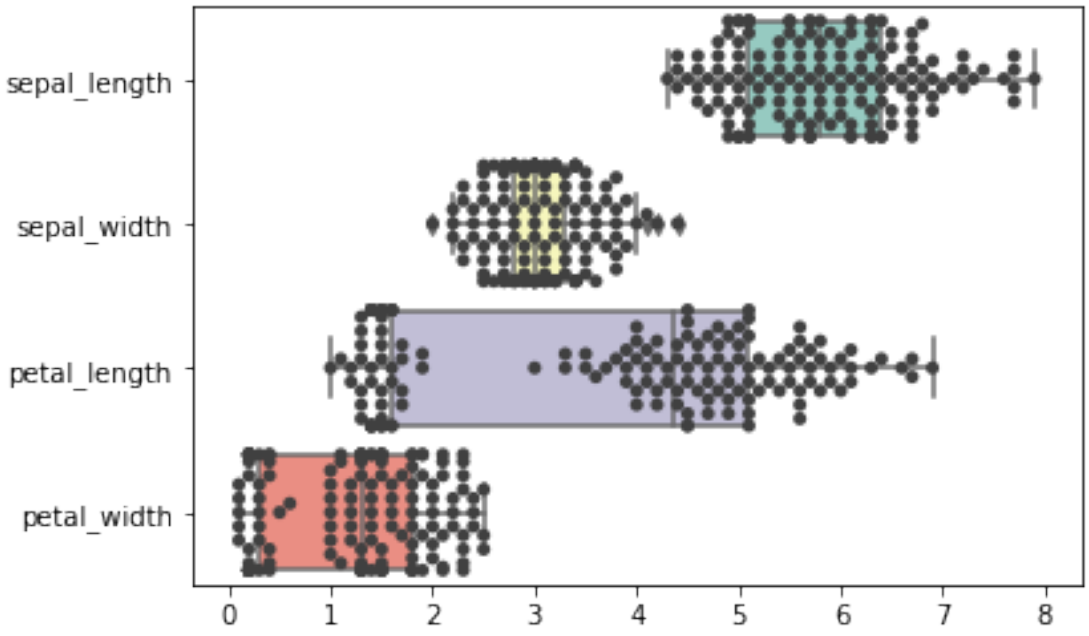
```
In [14]: sns.boxplot(data = iris, orient = "h", palette = "Set3")
```

```
Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x1a1509fc88>
```



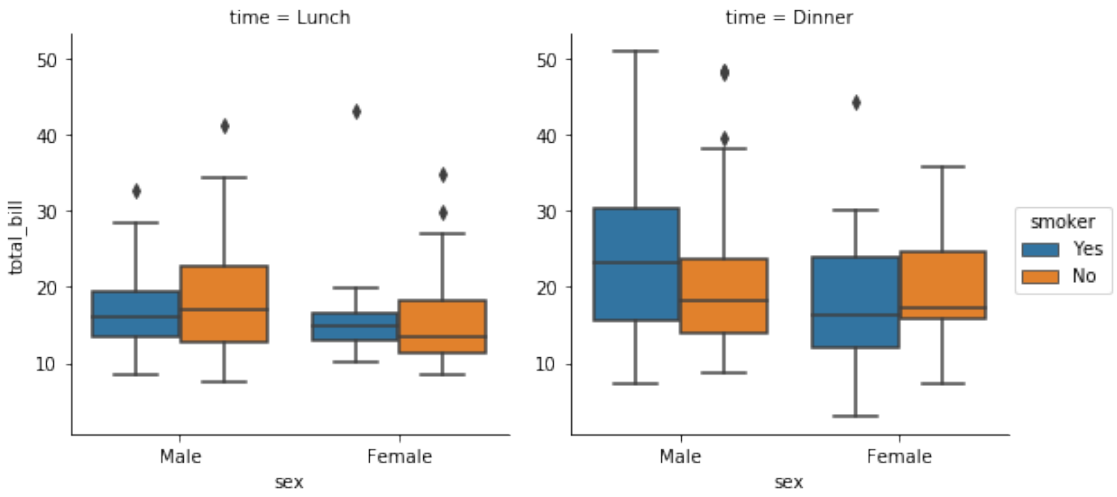

```
In [15]: sns.boxplot(data = iris, orient = "h", palette = "Set3")
         sns.swarmplot(data = iris, orient = "h",  color = ".25")
```

```
Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x1a151af390>
```



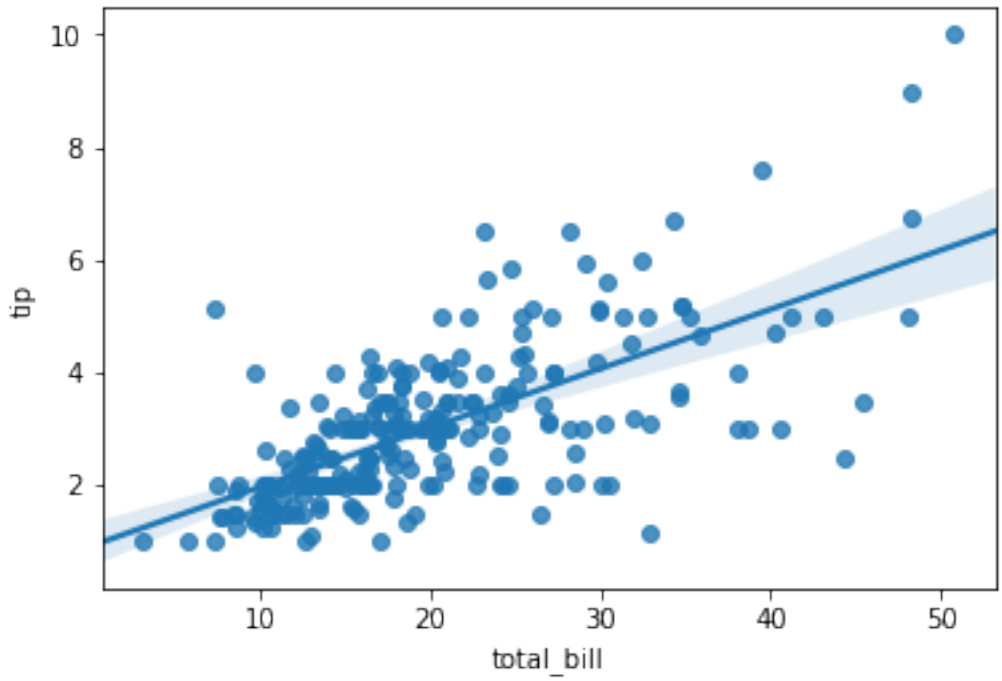
```
In [16]: sns.factorplot(x="sex", y="total_bill",
                        hue="smoker", col="time",
                        data=tips, kind="box")
```

```
Out[16]: <seaborn.axisgrid.FacetGrid at 0x1a151fb5c0>
```



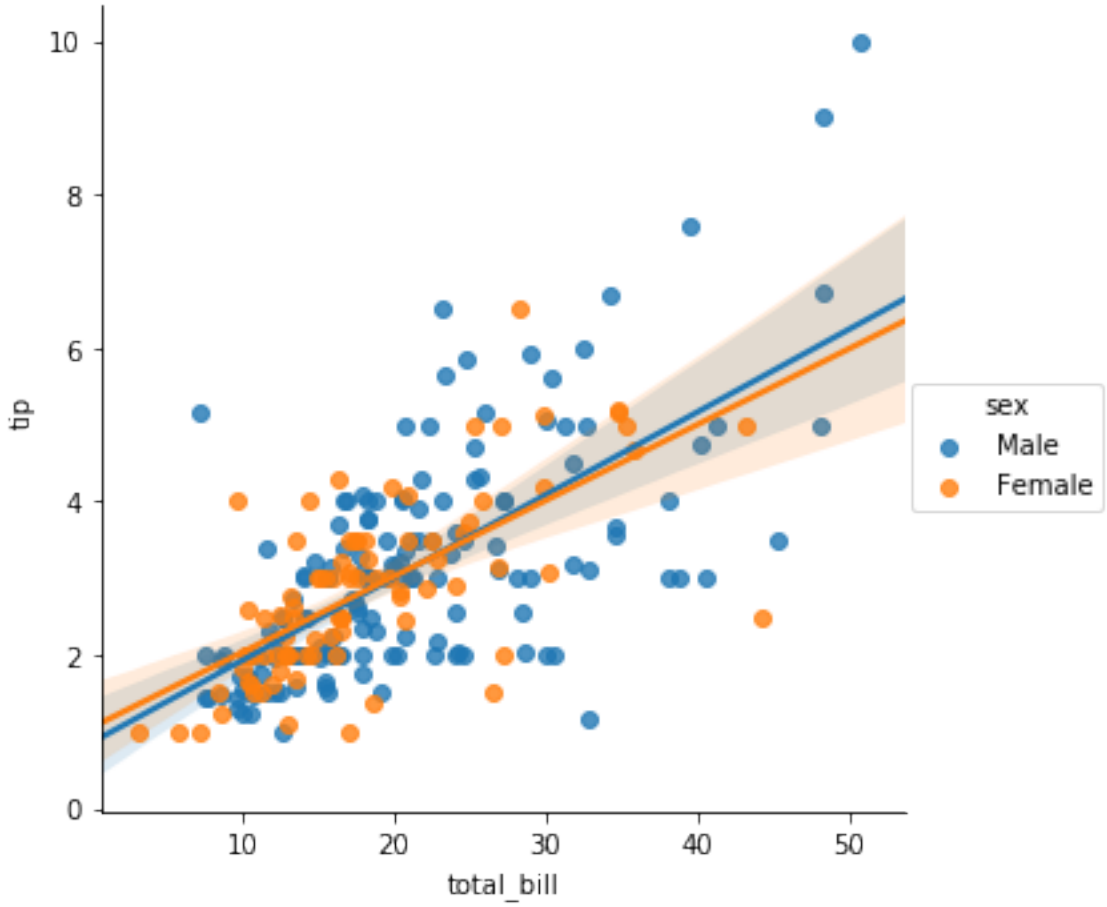

```
In [17]: sns.regplot(x = "total_bill", y = "tip", data = tips)
```

```
Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0x1a1548af28>
```



```
In [18]: sns.lmplot(x = "total_bill", y = "tip", hue = "sex", data = tips)
```

```
Out[18]: <seaborn.axisgrid.FacetGrid at 0x1a154bc588>
```



1.0.4 Playing with More Data

Below, we examine some other datasets available from seaborn. We can use these to play a little more with some of the plotting capabilities, and see how they help us to understand the structure of a dataset.

```
In [19]: sns.get_dataset_names()
```

```
/Users/NYCMath/anaconda3/lib/python3.6/site-packages/bs4/__init__.py:181: UserWarning: No parser

The code that caused this warning is on line 193 of the file /Users/NYCMath/anaconda3/lib/python
```

```
 BeautifulSoup(YOUR_MARKUP})

to this:

 BeautifulSoup(YOUR_MARKUP, "lxml")

  markup_type=markup_type))
```

```
Out[19]: ['anscombe',
          'attention',
          'brain_networks',
          'car_crashes',
          'dots',
          'exercise',
          'flights',
          'fmri',
          'gammas',
          'iris',
          'planets',
          'tips',
          'titanic']
```

```
In [20]: data = sns.load_dataset('attention')
```

```
In [21]: data.head()
```

```
Out[21]:    Unnamed: 0  subject attention  solutions  score
         0           0        1   divided          1    2.0
         1           1        2   divided          1    3.0
         2           2        3   divided          1    3.0
         3           3        4   divided          1    5.0
         4           4        5   divided          1    4.0
```