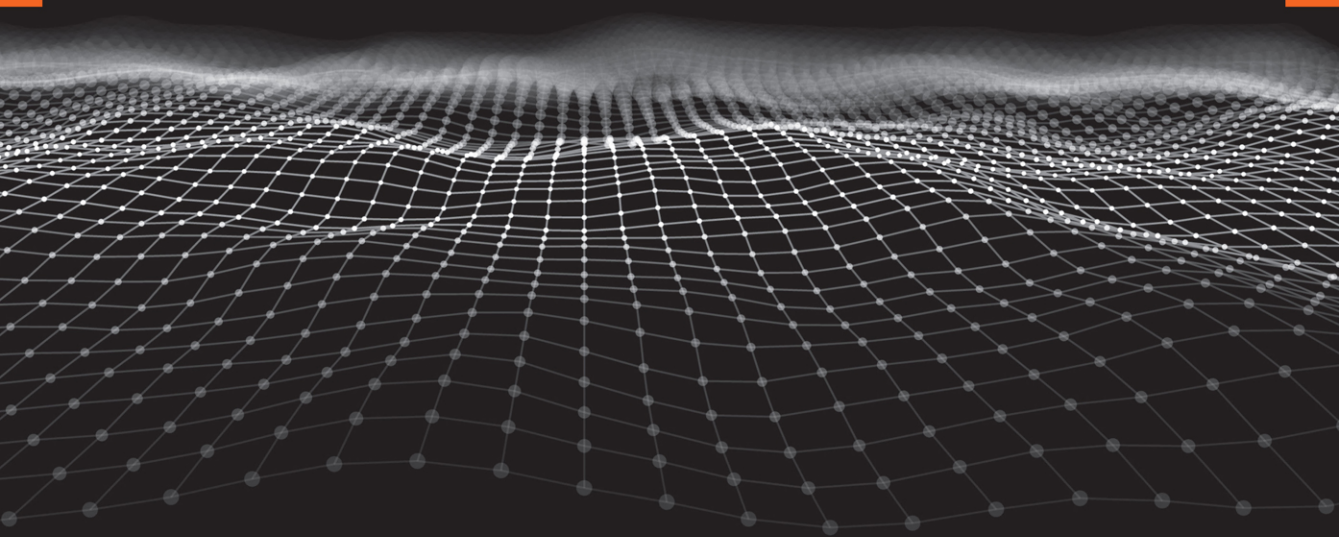
ADDISON-WESLEY DATA & ANALYTICS SERIES

PANDAS FOR EVERYONE

PYTHON DATA ANALYSIS

SECOND EDITION



DANIEL Y. CHEN

Pandas for Everyone

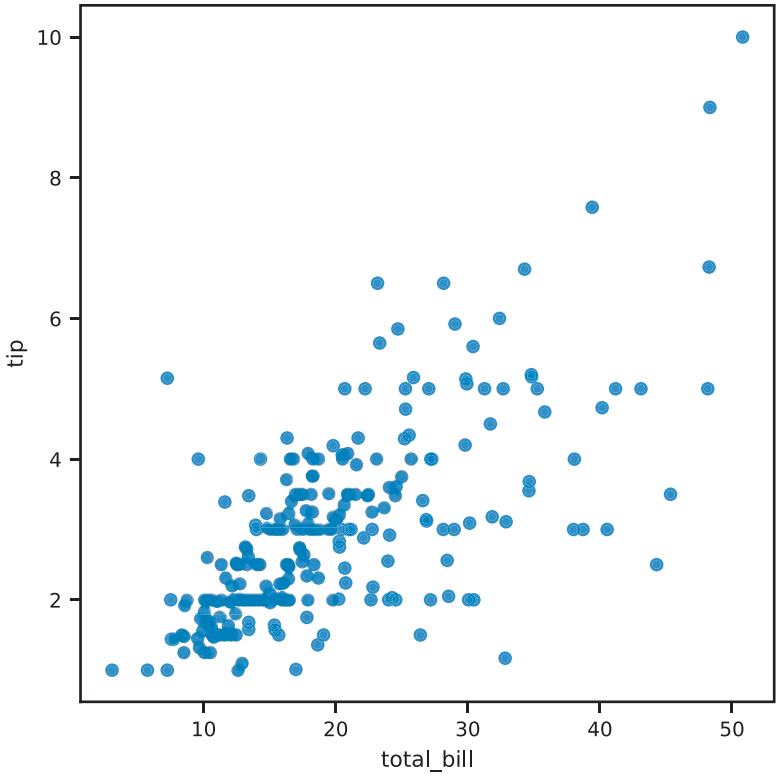


Figure 3.47 Pandas scatter plot

```
fig, ax = plt.subplots()
tips.plot.scatter(x='total_bill', y='tip', ax=ax)
plt.show()
```

3.5.4 Hexbin Plot

Hexbin plots are created using the `Dataframe.plt.hexbin()` function (Figure 3.48).

```
fig, ax = plt.subplots()
tips.plot.hexbin(x='total_bill', y='tip', ax=ax)
plt.show()
```

Grid size can be adjusted with the `gridsize` parameter (Figure 3.49).

```
fig, ax = plt.subplots()
tips.plot.hexbin(x='total_bill', y='tip', gridsize=10, ax=ax)
plt.show()
```

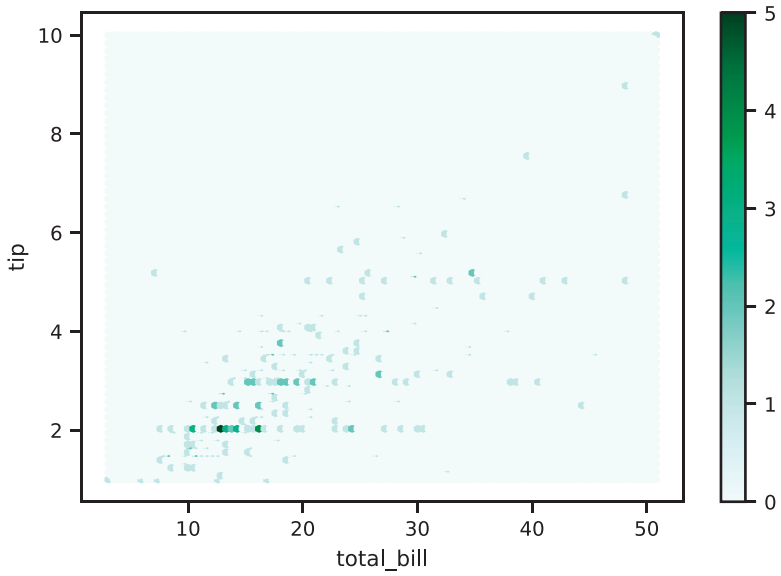


Figure 3.48 Pandas hexbin plot

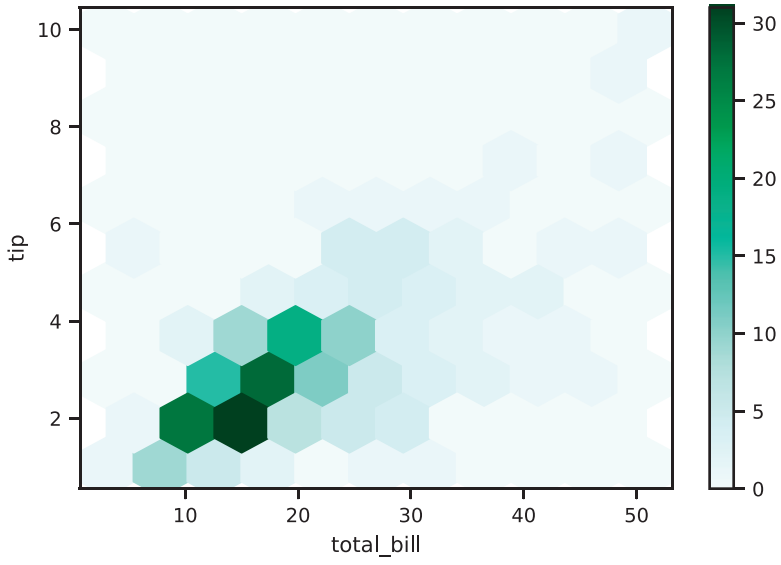


Figure 3.49 Pandas hexbin plot with modified grid size

3.5.5 Box Plot

Box plots are created with the `DataFrame.plot.box()` function (Figure 3.50).

```
fig, ax = plt.subplots()
ax = tips.plot.box(ax=ax)
plt.show()
```

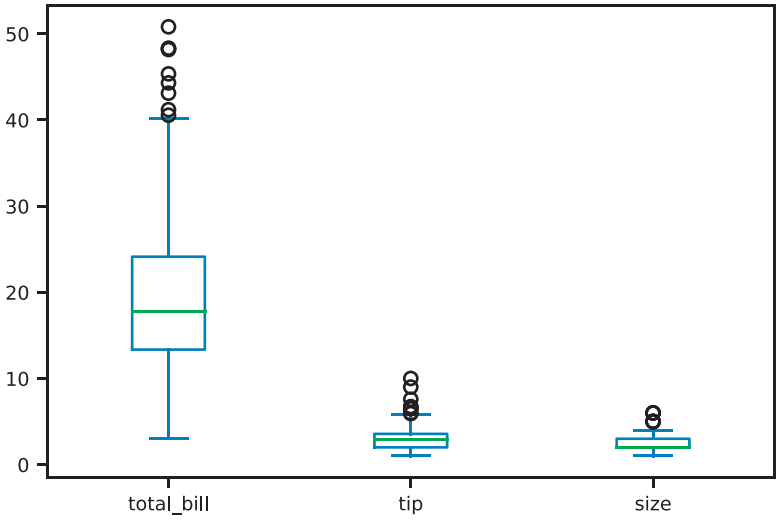


Figure 3.50 Pandas box plot

Conclusion

Data visualization is an integral part of exploratory data analysis and data presentation. This chapter provided an introduction to the various ways to explore and present your data. As we continue through the book, we will learn about more complex visualizations.

There are myriad plotting and visualization resources available on the Internet. The `seaborn` documentation, Pandas visualization documentation, and `matplotlib` documentation all provide ways to further tweak your plots (e.g., colors, line thickness, legend placement, figure annotations). Other resources include `colorbrewer` to help pick good color schemes. The plotting libraries mentioned in this chapter also have various color schemes that can be used to highlight the content of your visualizations.

This page intentionally left blank

4

Tidy Data

Hadley Wickham, PhD,[1] one of the more prominent members of the R community, introduced the concept of **tidy data** in a *Journal of Statistical Software* paper.[2] Tidy data is a framework to structure data sets so they can be easily analyzed and visualized. It can be thought of as a goal one should aim for when cleaning data. Once you understand what tidy data is, that knowledge will make your data analysis, visualization, and collection much easier.

What is tidy data? Hadley Wickham's paper defines it as meeting the following criteria: (1) Each row is an observation, (2) Each column is a variable, and (3) Each type of observational unit forms a table.

The newer definition from the R4DS book[3] focuses on an individual data set (i.e., table):

1. Each variable must have its own column.
2. Each observation must have its own row.
3. Each value must have its own cell.

This chapter goes through the various ways to tidy data using examples from Wickham's paper.

Learning Objectives

The concept map for this chapter can be found in Figure A.4.

- Identify the components of tidy data
- Identify common data errors
- Use functions and methods to process and tidy data

Note About This Chapter

Data used in this chapter will have `NaN` missing values when they are loaded into Pandas (Chapter 9). In the raw CSV files, they will appear as empty values. I typically try to avoid

1. Hadley Wickham, PhD: `http://hadley.nz`
2. Tidy Data paper: `http://vita.had.co.nz/papers/tidy-data.pdf`
3. R For Data Science Book: `https://r4ds.had.co.nz/tidy-data.html`

forward referencing in the book, but I felt that the concept of tidy data warranted a much earlier place in the book because it is so fundamental to how we should be thinking about data technically (as opposed to ethically), that the chapter was moved toward the front of the book without having to cover more detailed data processing steps first. I could have changed the data sets such that there were no missing values, but opted not to do so because (1) it would no longer follow the data used in Wickam's "Tidy Data" paper, and (2) it would be a less realistic data set.

4.1 Columns Contain Values, Not Variables

Data can have columns that contain values instead of variables. This is usually a convenient format for data collection and presentation.

4.1.1 Keep One Column Fixed

We'll use data on income and religion in the United States from the Pew Research Center to illustrate how to work with columns that contain values, rather than variables.

```
import pandas as pd
pew = pd.read_csv('data/pew.csv')
```

When we look at this data set, we can see that not every column is a variable. The values that relate to income are spread across multiple columns. The format shown is a great choice when presenting data in a table, but for data analytics, the table should be reshaped so that we have `religion`, `income`, and `count` variables.

```
# show only the first few columns
print(pew.iloc[:, 0:5])
```

```
                  religion  <$10k  $10-20k  $20-30k  $30-40k
0                 Agnostic     27       34       60       81
1                  Atheist     12       27       37       52
2                 Buddhist     27       21       30       34
3                 Catholic    418      617      732      670
4      Don't know/refused     15       14       15       11
..                    ...    ...      ...      ...      ...
13                Orthodox     13       17       23       32
14        Other Christian      9        7       11       13
15           Other Faiths     20       33       40       46
16  Other World Religions      5        2        3        4
17            Unaffiliated    217      299      374      365

[18 rows x 5 columns]
```

This view of the data is also known as "wide" data. To turn it into the "long" tidy data format, we will have to unpivot/melt/gather (depending on which statistical programming language we use) our dataframe.