**Categorical Plot Types**

**Got It!**

## 1. Categorical Plot Types

In the first two chapters of this course, we covered the basics of how to use the Seaborn API for creating and customizing plots using different Seaborn and matplotlib approaches. These chapters provide the foundation for exploring additional plot types. This lesson will focus on the many different categorical plots that Seaborn supports.

## 2. Categorical Data

In our earlier exercises we looked at distribution and linear regression plots, used on numerical values. Seaborn also supports many plot types with categorical data. Categorical data is data which includes a limited or fixed number of values and is most useful when combined with numeric data. For the rest of this lesson, we will be looking at US Healthcare reimbursement data related to Renal Failure category codes and their associated reimbursement values. In these examples, the codes are the categorical variables and the average hospital charge is the numerical value we will analyze in our plots.

## 3. Plot types - show each observation

Seaborn breaks categorical data plots into three groups. The first group includes the stripplot() and swarmplot(), which show all the individual observations on the plot.

## 4. Plot types - abstract representations

The second category contains the familiar boxplot(), as well as the violinplot() and lvplot(). These plots show an abstract representation of the categorical data.

## 5. Plot types - statistical estimates

The final group of plots show statistical estimates of the categorical variables. The barplot() and pointplot() contain useful summaries of data. The countplot() shows the number of instances of each observation.

## 6. Plots of each observation - stripplot

Seaborn's stripplot() shows every observation in the dataset. In some cases, it can be difficult to see individual data points. We can use the jitter parameter in order to more easily see how the Average Covered Charges vary by Diagnostic Reimbursement Code.

## 7. Plots of each observation - swarmplot

We can plot a more sophisticated visualization of all the data using a swarmplot(). This plot uses a complex algorithm to place the observations in a manner where they do not overlap. **The downside of this approach is that the swarmplot() does not scale well to large datasets**.

## 8. Abstract representations - boxplot

The next category of plots show abstract representations of the data. A boxplot() is the most common of this type. This plot is used to show several measures related to the distribution of data, including the median, upper and lower quartiles, as well as outliers.

## 9. Abstract representation - violinplot

The violinplot() is a combination of a kernel density plot and a box plot and can be suitable for providing an alternative view of the distribution of data. Because the plot uses a kernel density calculation it does not show all data points. **This can be useful for displaying large datasets but it can be computationally intensive to create.**

## 10. Abstract representation - lvplot

The final plot in this grouping is the lvplot(), which stands for Letter Value plot. The API is the same as the boxplot() and violinplot() but can scale more effectively to large datasets. The lvplot() is a hybrid between a boxplot() and violinplot() and is relatively quick to render and easy to interpret.

## 11. Statistical estimates - barplot

The final category of plots are statistical estimates of the data. The barplot() shows an estimate of the value as well as a confidence interval. In this example, we include the hue parameter described in Chapter 1, which provides another useful way for us to look at this categorical data.

## 12. Statistical estimates - pointplot

The pointplot() is similar to the barplot() in that it shows a summary measure and confidence interval. A pointplot() can be very useful for observing how values change across categorical values.

## 13. Statistical estimates - countplot

The final categorical plot is the countplot(), which displays the number of instances of each variable.

## 14. Let's practice!

Now that we have gone through all of the categorical plots available in Seaborn, let's practice making some of our own.

# Selecting Seaborn Plots

**Got It!**

## 1. Selecting Seaborn Plots

We have covered a lot of different plots in Seaborn. The final section of this course will bring all of the concepts together and give you a framework for deciding when to use each Seaborn plot.

## 2. Seaborn plot map

We will reinforce the previous lessons by showing how these plot types fit together. The power of Seaborn is the way that the different plots build on each other. For instance, a kdeplot can be used on its own or it can be generated from a distplot(). In addition, the PairGrid() and JointGrid() plots build on top of the regression and distribution plots. Let's explore this in more detail and discuss guidelines on how to approach using Seaborn in your daily data science workflow.

## 3. Univariate Distribution Analysis

One of the first steps in analyzing numerical data is looking at its distribution. Seaborn's distplot() combines many of the features of the rugplot(), kdeplot(), and matplotlib histogram into a single function. The distplot() function is the best place to start when trying to do distribution analysis with Seaborn.

## 4. Regression Analysis

A regression plot is an example of a plot that shows the relationship between two variables. matplotlib's scatter() plot is a very simple method to compare the interaction of two variables on the x and y-axis. The lmplot() combines many of these features of the underlying regplot() and residplot() in addition to the ability to plot the data on a FacetGrid(). In many instances, lmplot() is the best function to use for determining linear relationships between data.

## 5. Categorical Plots

Seaborn has many types of categorical plots as well. In most scenarios, it makes sense to use one of the categorical plots such as the boxplot() or violinplot() to examine the distribution of the variables. Then, follow up with the statistical estimation plots such as the point, bar, or countplot. If you need to facet the data across rows or columns, use a factorplot().

## 6. pairplot() and jointplot()

The pairplot() and jointplot() visualizations are going to be most useful after you have done some preliminary analysis of regressions or distributions of the data. Once you are familiar with the data, the pairplot() and jointplot() can be very useful in understanding how two or more variables interact with each other.

## 7. Thank You!

Congratulations on completing the course. You are now familiar with the Seaborn library and can start to incorporate it into your own data analysis tasks!