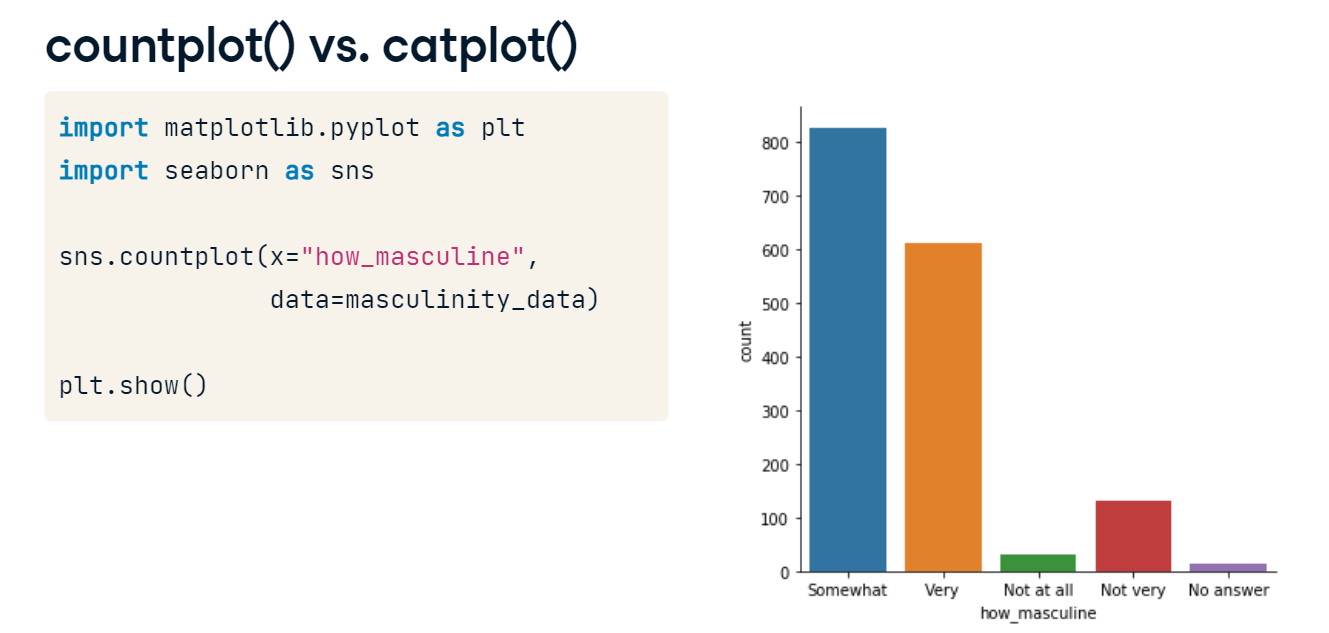
**Count plots and bar plots**

Welcome to Chapter 3! In this chapter, we'll focus on visualizations that involve categorical variables. The first two plots we'll look at are count plots and bar plots.

**Categorical plots**

Count plots and bar plots are two types of visualizations that Seaborn calls "categorical plots". Categorical plots involve a categorical variable, which is a variable that consists of a fixed, typically small number of possible values, or categories. These types of plots are commonly used when we want to make comparisons between different groups. We began to explore categorical plots in Chapter 1 with count plots. As a reminder, a count plot displays the number of observations in each category. We saw several examples of count plots in earlier chapters, like the number of men reporting that they feel masculine. Most men surveyed here feel "somewhat" or "very" masculine.



**catplot()**

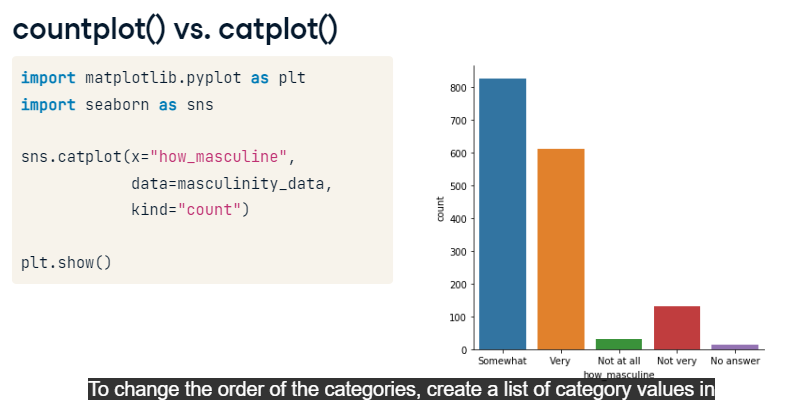
Just like we used "relplot()" to create different types of relational plots, in this chapter we'll be using "catplot()" to create different types of categorical plots. "catplot()" offers the same flexibility that "relplot()" does, which means it will be easy to create subplots if we need to using the same "col" and "row" parameters.

**countplot() vs. catplot()**

To see how "catplot()" works, let's return to the masculinity count plot. On the left, we see how we originally created a count plot with the "countplot()" function.

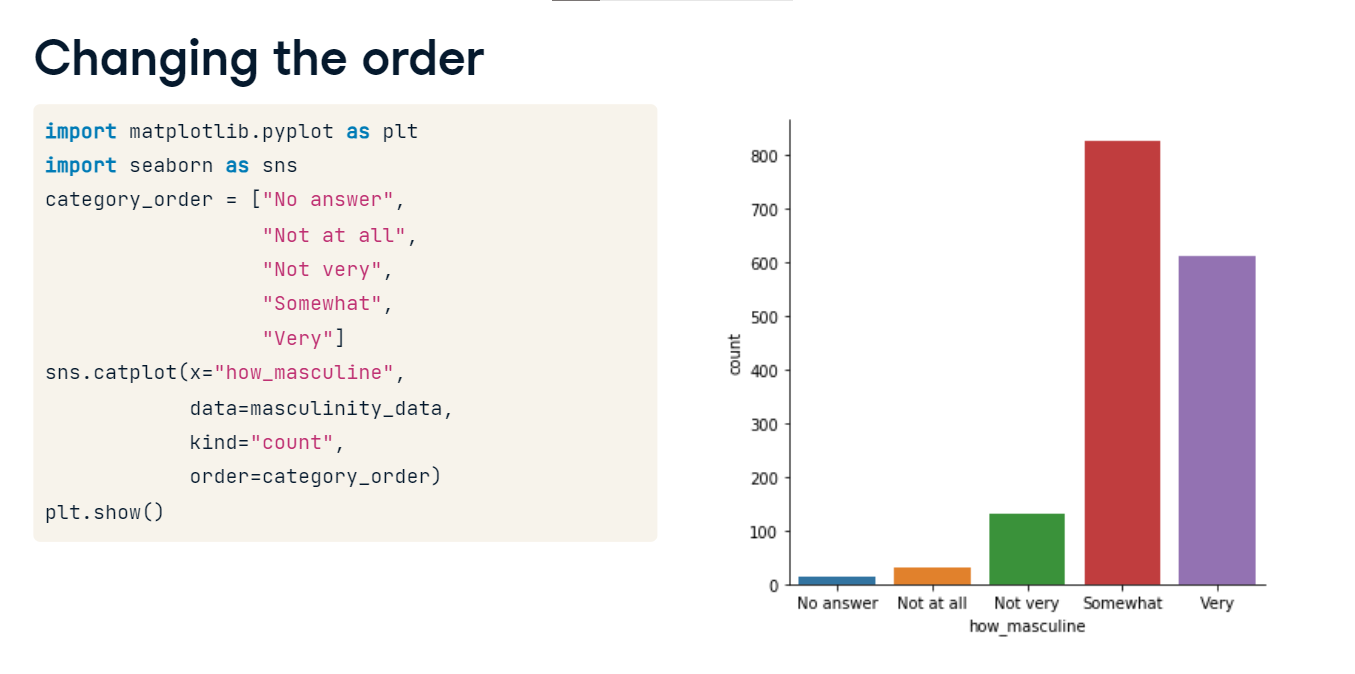
**countplot() vs. catplot()**

To make this plot with "catplot()" instead, we change the function name to "catplot()" and use the "kind" parameter to specify what kind of categorical plot to use. In this case, we'll set kind equal to the word "count".



**Changing the order**

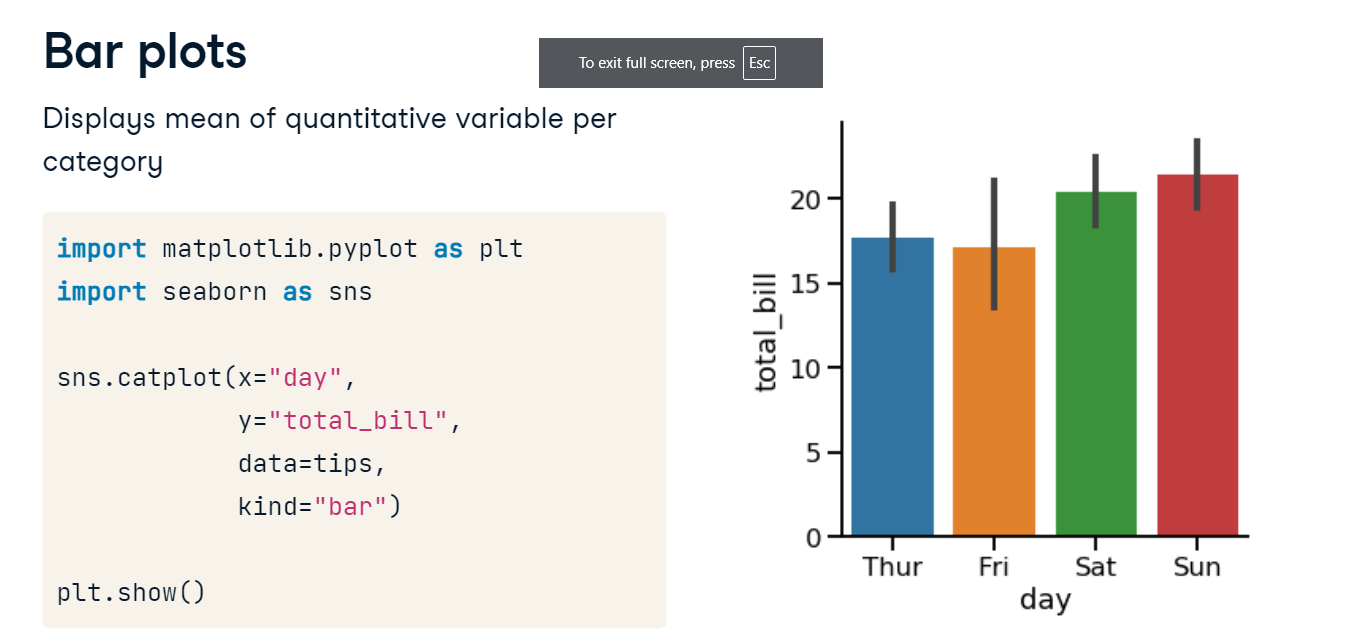
Sometimes there is a specific ordering of categories that makes sense for these plots. In this case, it makes more sense for the categories to be in order from not masculine to very masculine. To change the order of the categories, create a list of category values in the order that you want them to appear, and then use the "order" parameter. This works for all types of categorical plots, not just count plots.



**Bar plots**

Bar plots look similar to count plots, but instead of the count of observations in each category, they show the mean of a quantitative variable among observations in each category. This bar plot uses the tips dataset and shows the average bill paid among people who visited the restaurant on each day of the week. From this, we can see that the average bill is slightly higher on the weekends. To create this bar plot, we use "catplot". Specify the categorical variable "day" on the x-axis, the quantitative variable "total bill" on the y-axis, and set the "kind" parameter equal to "bar".

1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. <https://seaborn.pydata.org/>



**Confidence intervals**

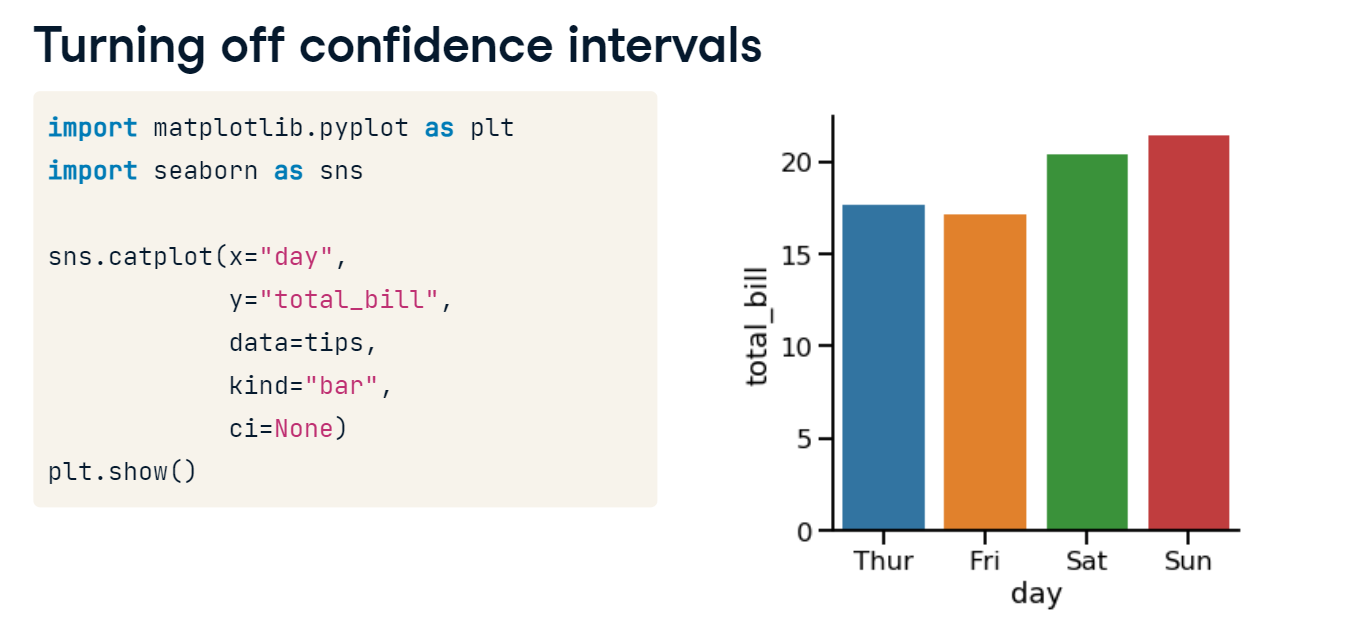
Notice also that Seaborn automatically shows 95% confidence intervals for these means. Just like with line plots, these confidence intervals show us the level of uncertainty we have about these estimates. Assuming our data is a random sample of some population, we can be 95% sure that the true population mean in each group lies within the confidence interval shown.

1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. https://seaborn.pydata.org/

**Turning off confidence intervals**

If we want to turn off these confidence intervals, we can do this by setting the "ci" parameter equal to "None" - just like we did with line plots.

1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. <https://seaborn.pydata.org/>



**Changing the orientation**

Finally, you can also change the orientation of the bars in bar plots and count plots by switching the x and y parameters. However, it is fairly common practice to put the categorical variable on the x-axis.

1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. https://seaborn.pydata.org/

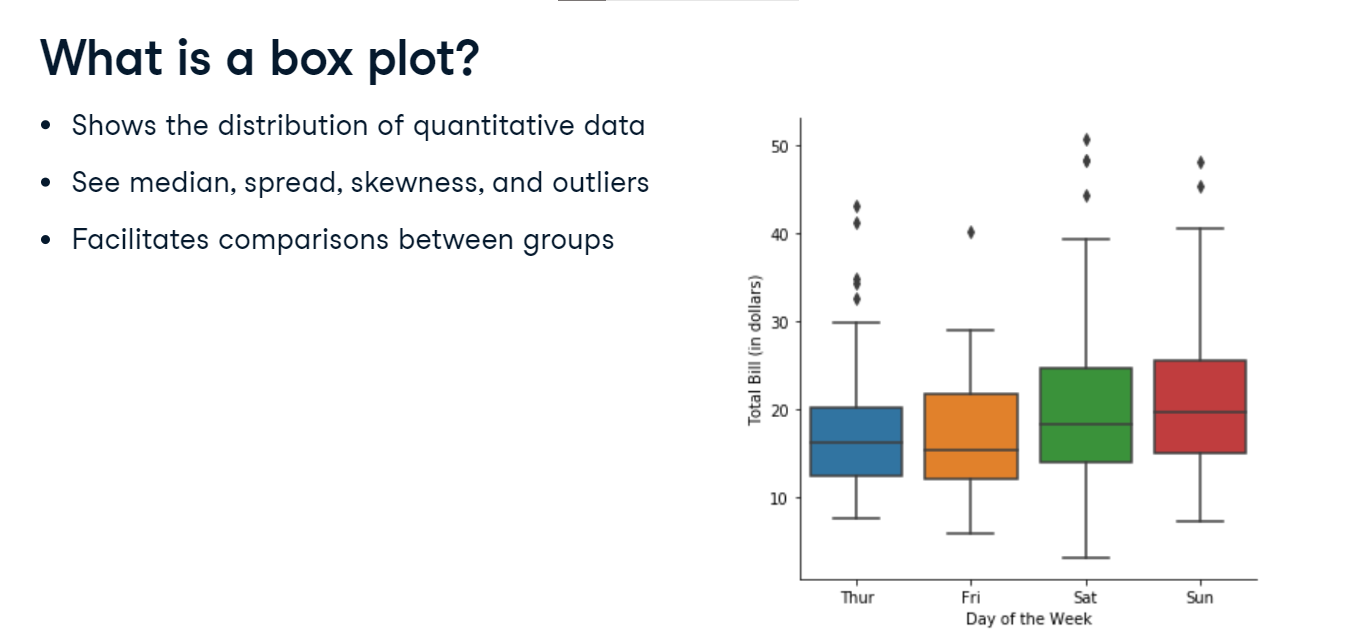
**Creating a box plot**

Hello! In this video we'll learn how to create a new type of categorical plot: the box plot.

**What is a box plot?**

A box plot shows the distribution of quantitative data. The colored box represents the 25th to 75th percentile, and the line in the middle of the box represents the median. The whiskers give a sense of the spread of the distribution, and the floating points represent outliers. Box plots are commonly used as a way to compare the distribution of a quantitative variable across different groups of a categorical variable. To see this, let's look at this example. The box plot shown here uses the tips dataset and compares the distribution of the total bill paid per table across the different days of the week. From this box plot we can quickly see that the median bill is higher on Saturday and Sunday, but the spread of the distribution is also larger. This comparison would be much harder to do with other types of visualizations.

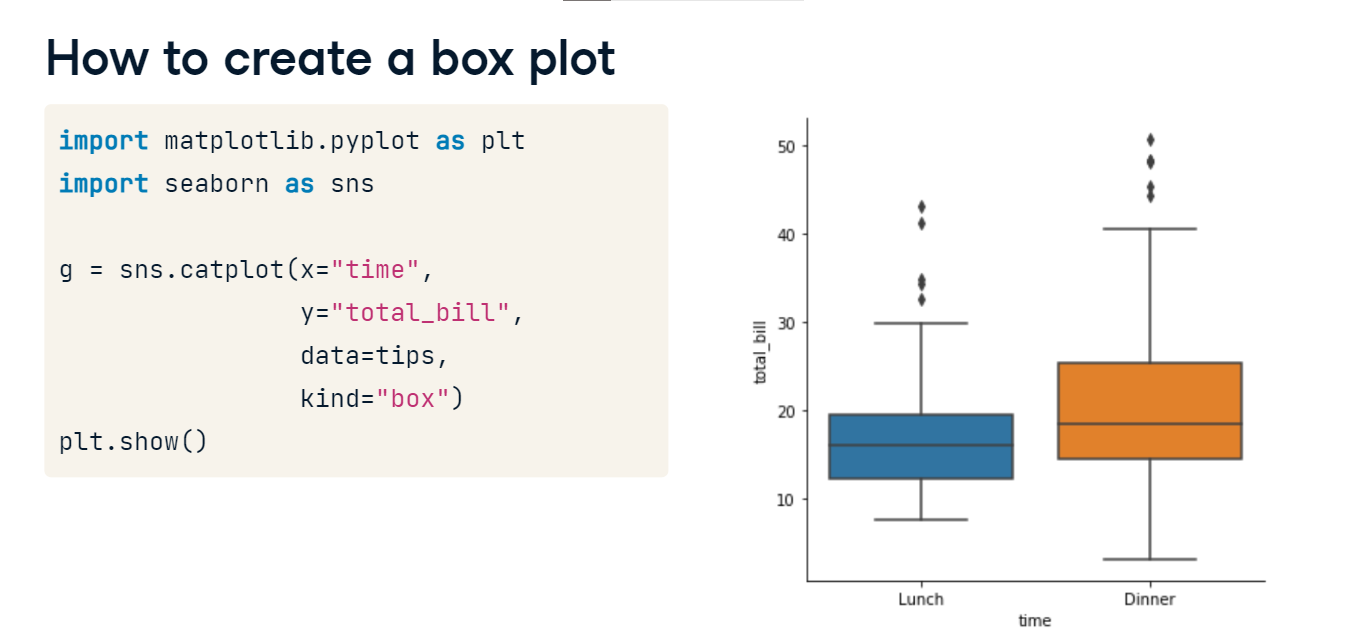
1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. https://seaborn.pydata.org/

****

**How to create a box plot**

Now let's look at how to create a box plot in Seaborn. While Seaborn does have a "boxplot()" function, we'll be using the "catplot()" function that we introduced in an earlier lesson because it makes it easy to create subplots using the "col" and "row" parameters. We'll put the categorical variable "time" on the x-axis and the quantitative variable "total bill" on the y-axis. Here, we want box plots, so we'll specify kind="box". That's it! We have a nice looking box plot. Next, we'll look at different ways to customize this plot.

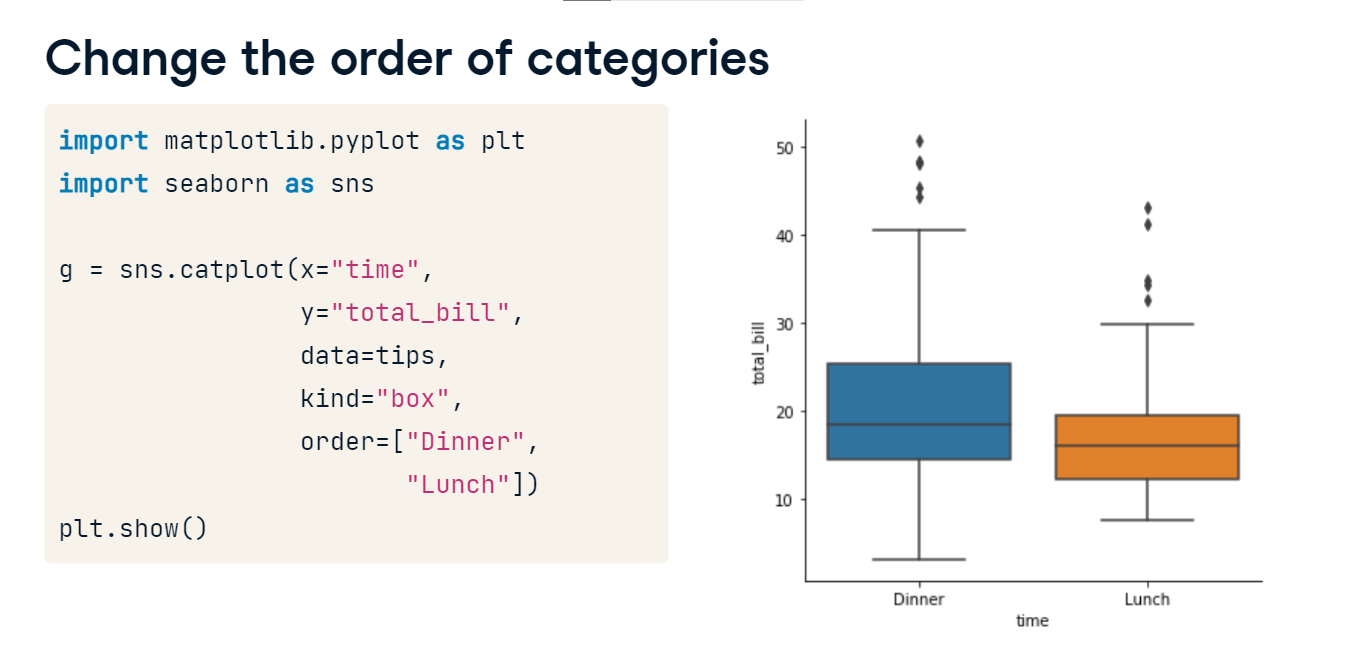
1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. https://seaborn.pydata.org/

****

**Change the order of categories**

As a reminder, "catplot" allows you to change the order of the categories using the "order" parameter. Here, we specified that "dinner" should be shown before "lunch".

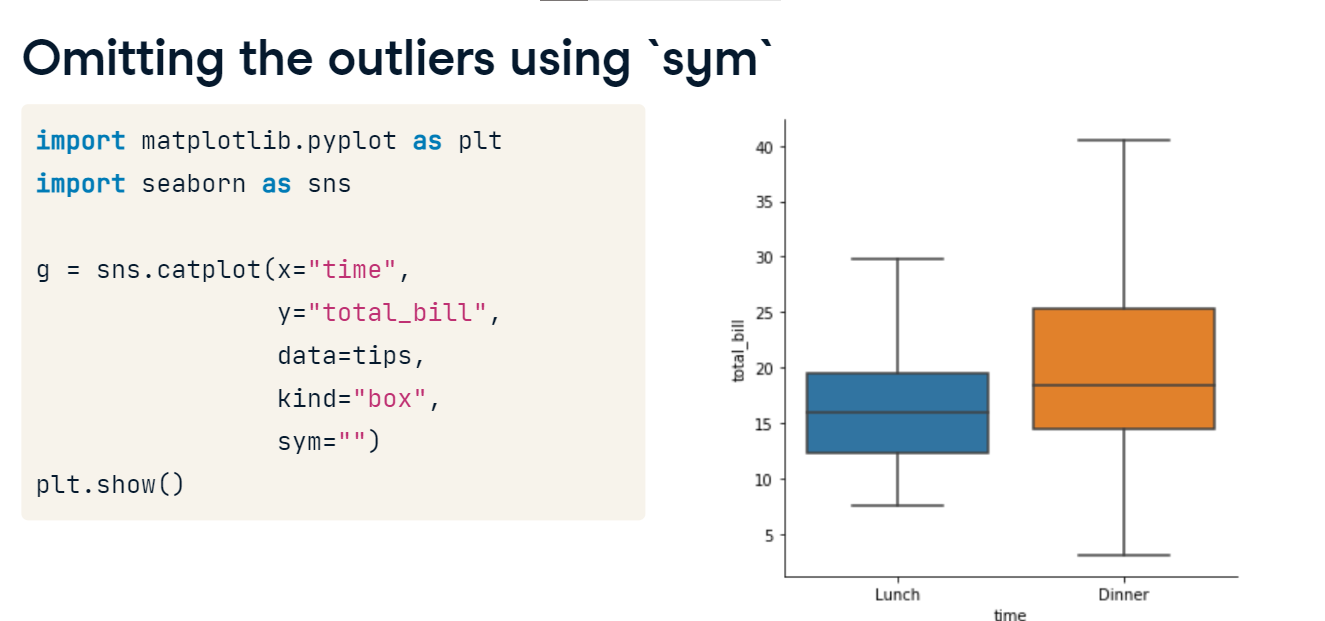
1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. https://seaborn.pydata.org/

****

**Omitting the outliers using `sym`**

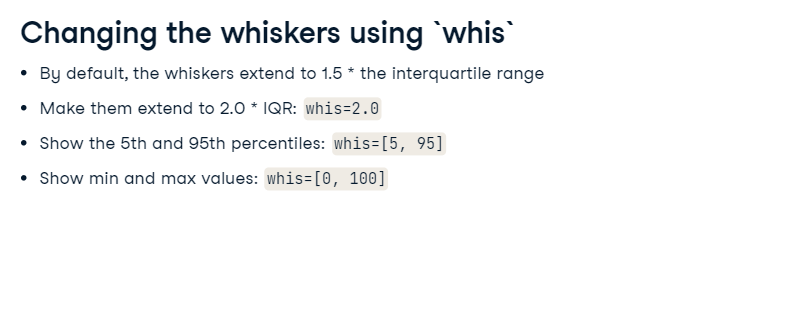
Occasionally, you may want to omit the outliers from your box plot. You can do this using the "sym" parameter. If you pass an empty string into "sym", it will omit the outliers from your plot altogether. "Sym" can also be used to change the appearance of the outliers instead of omitting them.

1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. https://seaborn.pydata.org/

****

**Changing the whiskers using `whis`**

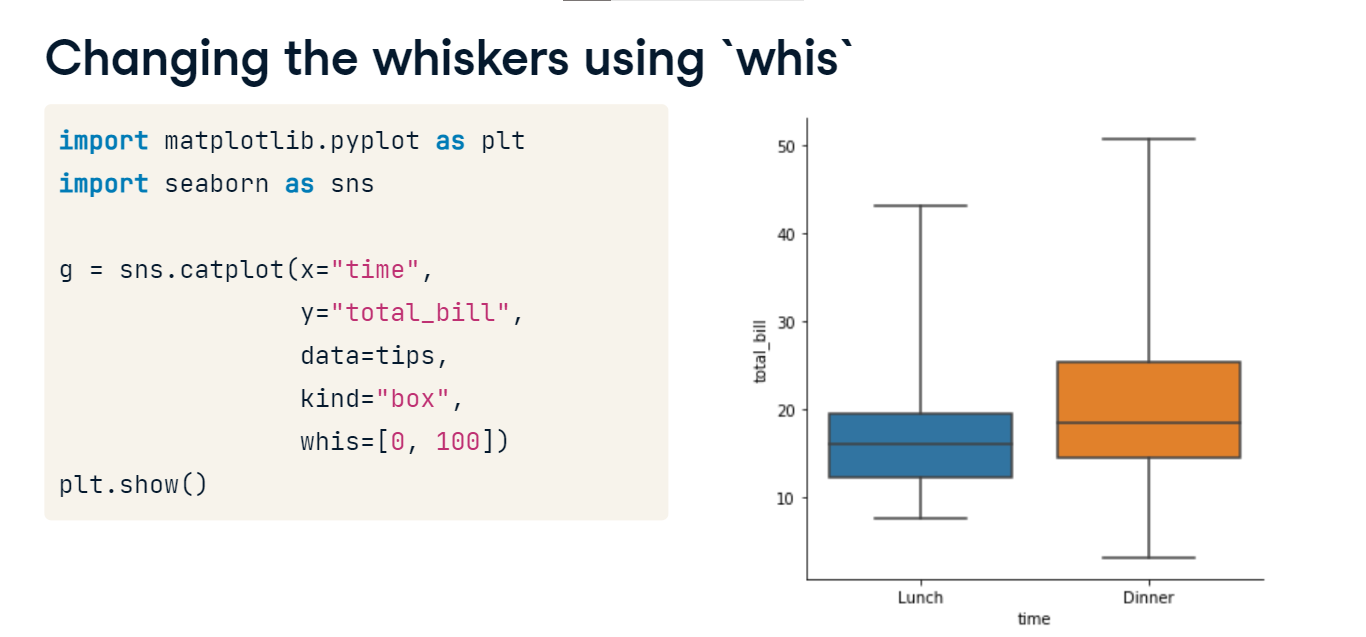
By default, the whiskers extend to 1 point 5 times the interquartile range, or "IQR". The IQR is the 25th to the 75th percentile of a distribution of data. If you want to change the way the whiskers in your box plot are defined, you can do this using the "whis" parameter. There are several options for changing the whiskers. You can change the range of the whiskers from 1 point 5 times the IQR (which is the default) to 2 times the IQR by setting "whis" equal to 2 point 0. Alternatively, you can have the whiskers define specific lower and upper percentiles by passing in a list of the lower and upper values. In this example, passing in "[5, 95]" will result in the lower whisker being drawn at the 5th percentile and the upper whisker being drawn at the 95th percentile. Finally, you may just want to draw the whiskers at the min and max values. You can do this by specifying the lower percentile as 0 and the upper percentile as 100.



**Changing the whiskers using `whis`**

Here's an example where the whiskers are set to the min and max values. Note that there are no outliers, because the box and whiskers cover the entire range of the data.

1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. https://seaborn.pydata.org/



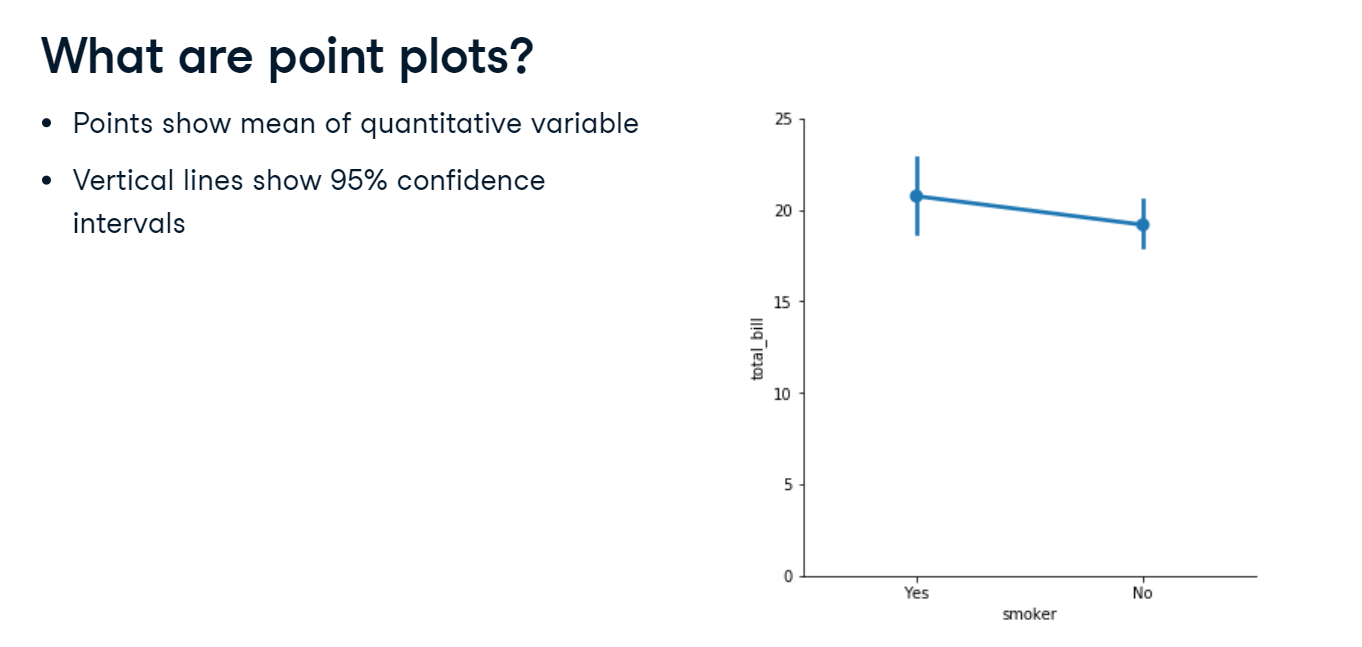
**Point plots**

Welcome! So far we've seen several types of categorical plots including count plots, bar plots, and box plots. In this lesson, we'll see one final categorical plot: point plots.

**What are point plots?**

Point plots show the mean of a quantitative variable for the observations in each category, plotted as a single point. This point plot uses the tips dataset and shows the average bill among smokers versus non-smokers. The vertical bars extending above and below the mean represent the 95% confidence intervals for that mean. Just like the confidence intervals we saw in line plots and bar plots, these confidence intervals show us the level of uncertainty we have about these mean estimates. Assuming our data is a random sample of some population, we can be 95% sure that the true population mean in each group lies within the confidence interval shown.

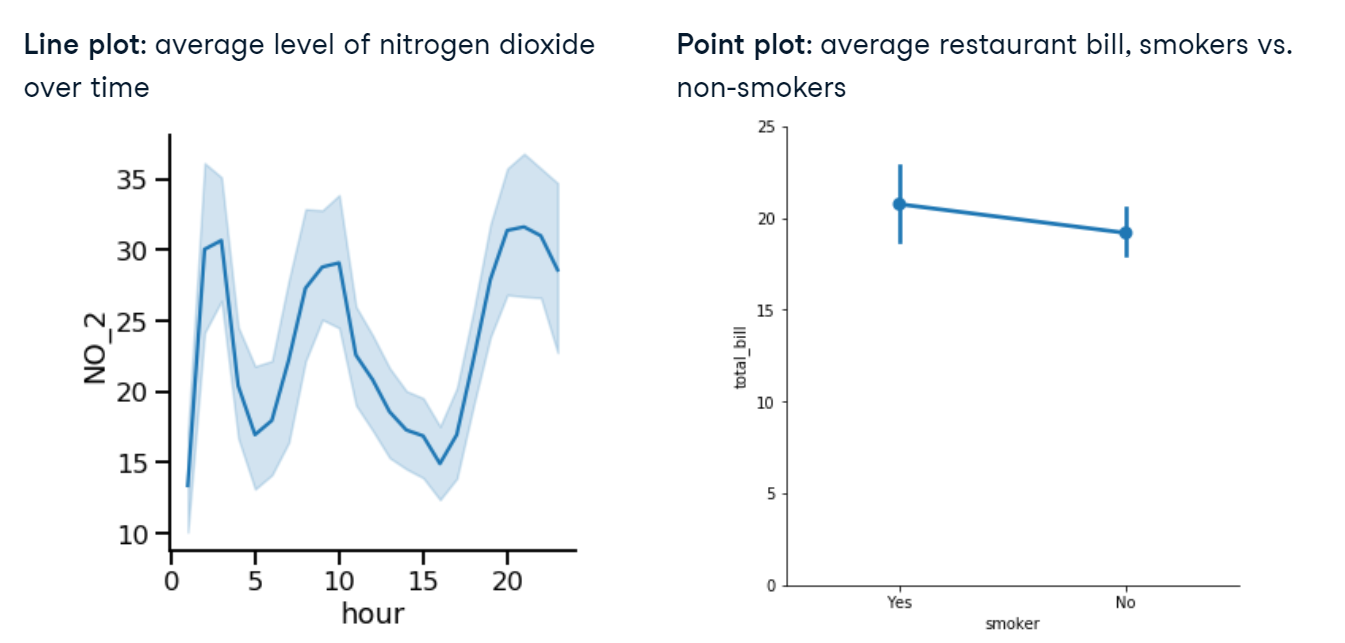
1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. https://seaborn.pydata.org/

****

**Point plots vs. line plots**

You may be thinking: point plots look a lot like line plots. What's the difference?

1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. https://seaborn.pydata.org/

****

**Point plots vs. line plots**

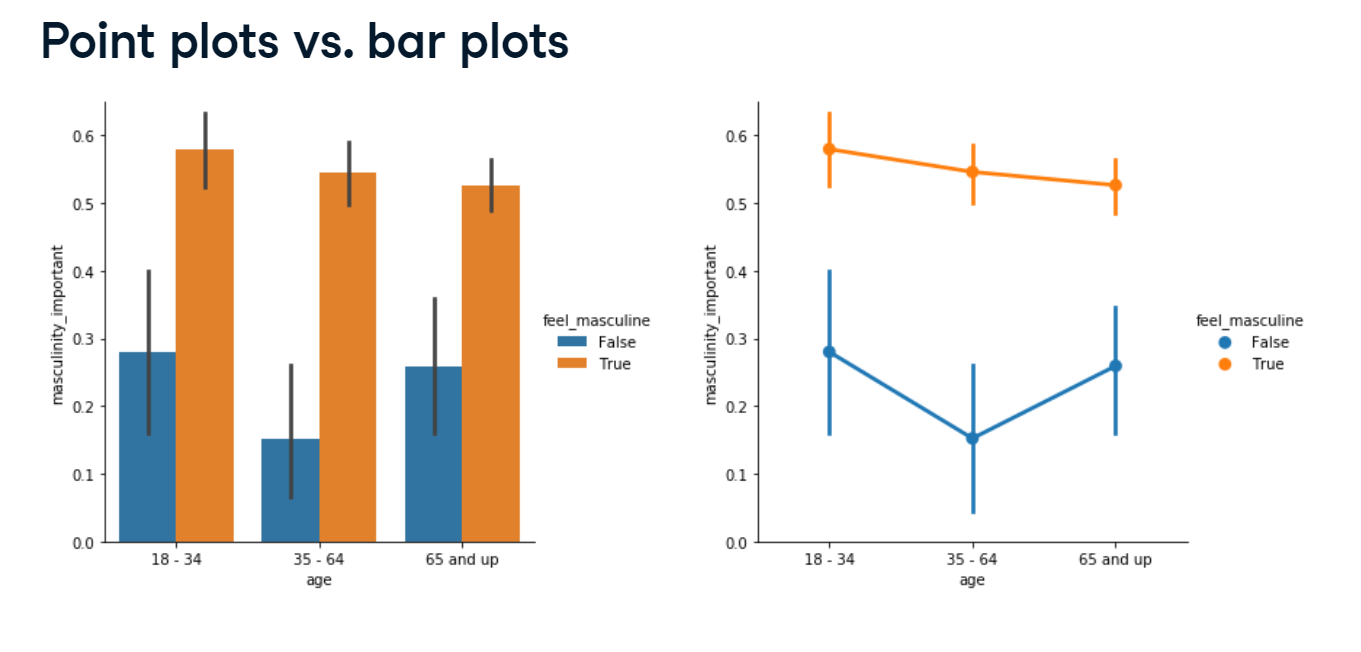
Both line plots and point plots show the mean of a quantitative variable and 95% confidence intervals for the mean. However, there is a key difference. Line plots are relational plots, so both the x- and y-axis are quantitative variables. In a point plot, one axis - usually the x-axis - is a categorical variable, making it a categorical plot.

**Point plots vs. bar plots**

You may also be thinking: point plots seem to show the same information as bar plots. For each category, both show the mean of a quantitative variable and the confidence intervals for those means. When should we use one over the other? Let's look at an example using data from the masculinity survey that we've seen in prior lessons.

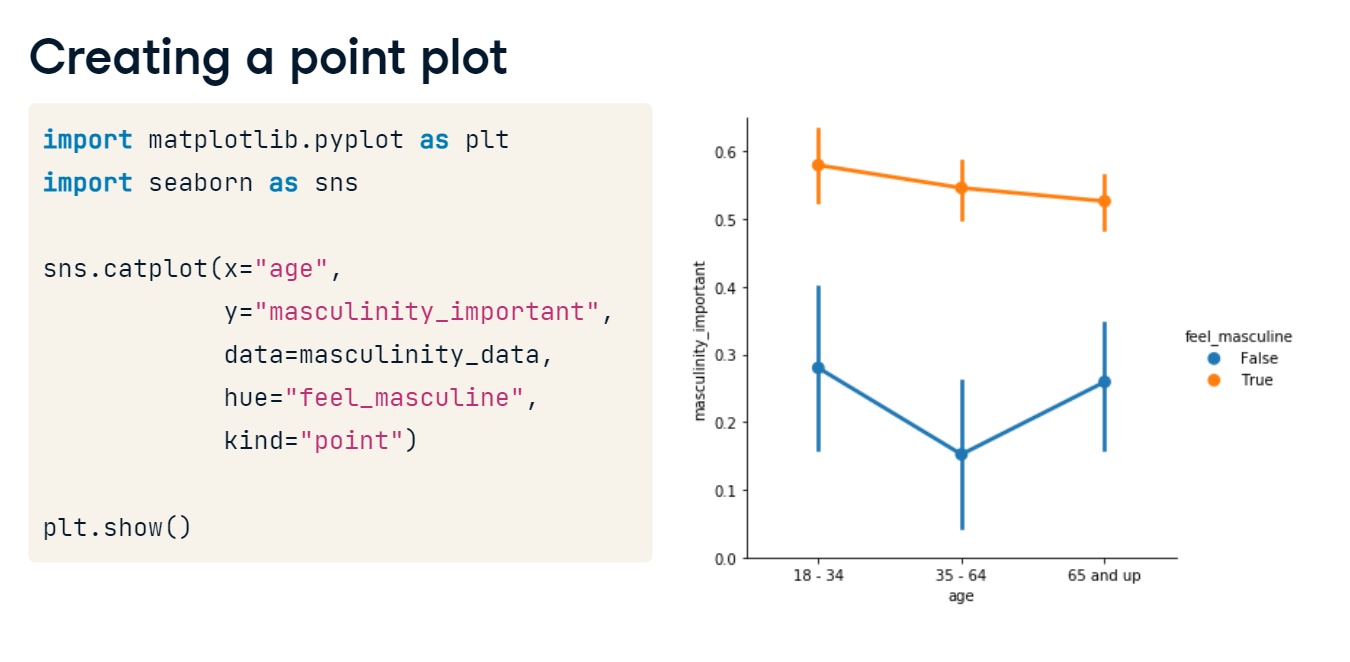
**Point plots vs. bar plots**

This is a bar plot of the percent of men per age group surveyed who report thinking that it's important that others see them as masculine, with subgroups based on whether they report feeling masculine or not. This is the same information, represented as a point plot. In the point plot, it's easier to compare the heights of the subgroup points when they're stacked above each other. In the point plot, it's also easier to look at the differences in slope between the categories than it is to compare the heights of the bars between them.

****

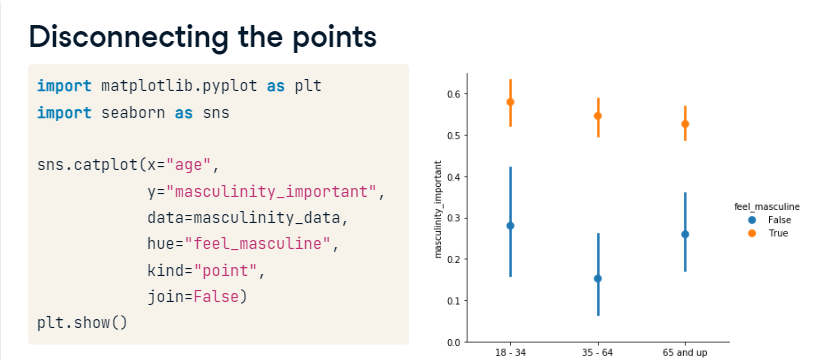
**Creating a point plot**

Here's the code to create the point plot we just saw. Since this is a categorical plot, we use "catplot" and set "kind" equal to "point".



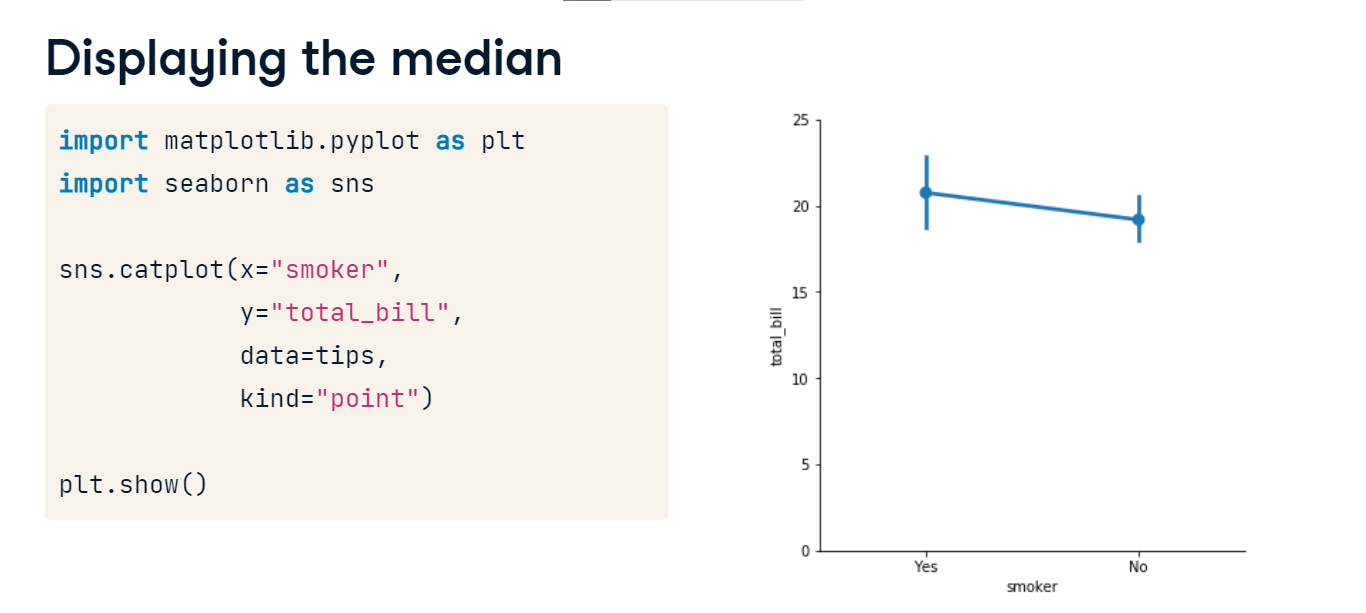
**Disconnecting the points**

Sometimes we may want to remove the lines connecting each point, perhaps because we only wish to compare within a category group and not between them. To do this, set the "join" parameter equal to False.



**Displaying the median**

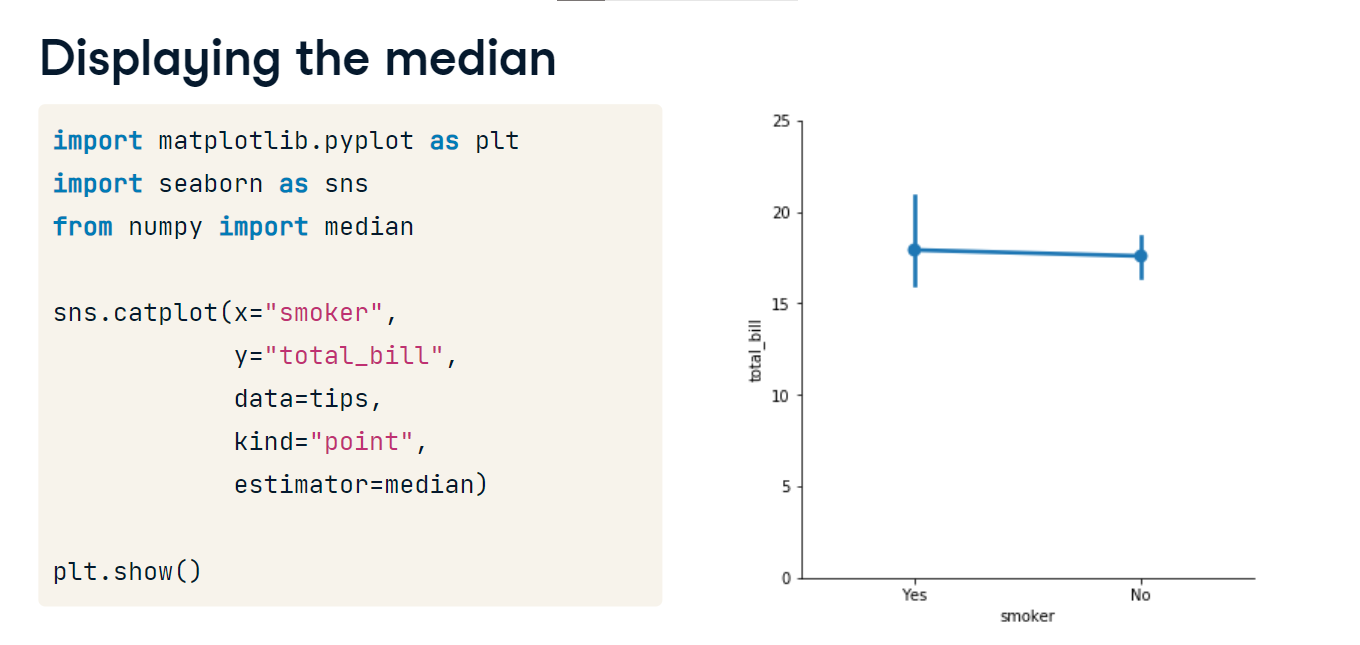
Let's return to the point plot using the tips dataset and go over a few more ways to customize your point plots. Here is the point plot of average bill comparing smokers to non-smokers.

1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. <https://seaborn.pydata.org/>

**Displaying the median**

To have the points and confidence intervals be calculated for the median instead of the mean, import the median function from the numpy library and set "estimator" equal to the numpy median function. Why might you want to use the median instead of the mean? The median is more robust to outliers, so if your dataset has a lot of outliers, the median may be a better statistic to use.

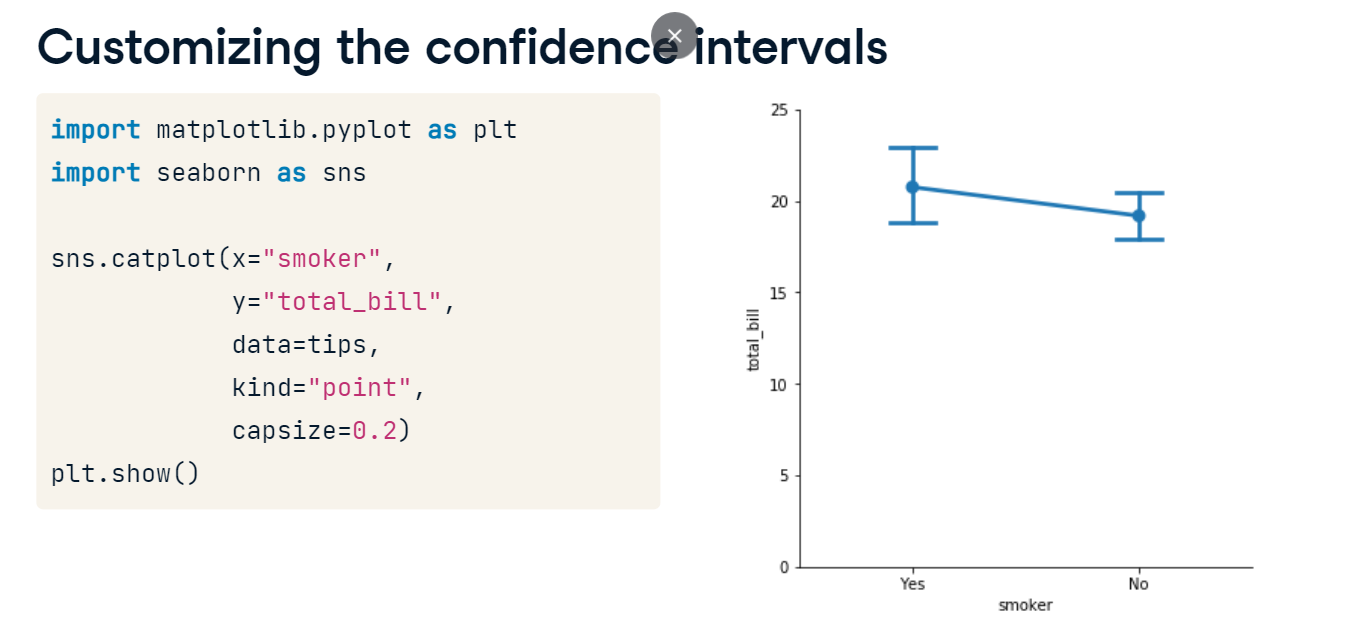
1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. https://seaborn.pydata.org/

****

**Customizing the confidence intervals**

You can also customize the way that the confidence intervals are displayed. To add “caps” to the end of the confidence intervals, set the “capsize” parameter equal to the desired width of the caps. In this case, we chose a width of 0.2.

1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. <https://seaborn.pydata.org/>



**Turning off confidence intervals**

Finally, like we saw with line plots and bar plots, you can turn the confidence intervals off by setting the "ci" parameter equal to None.

1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. https://seaborn.pydata.org/

