**Patterns over time**

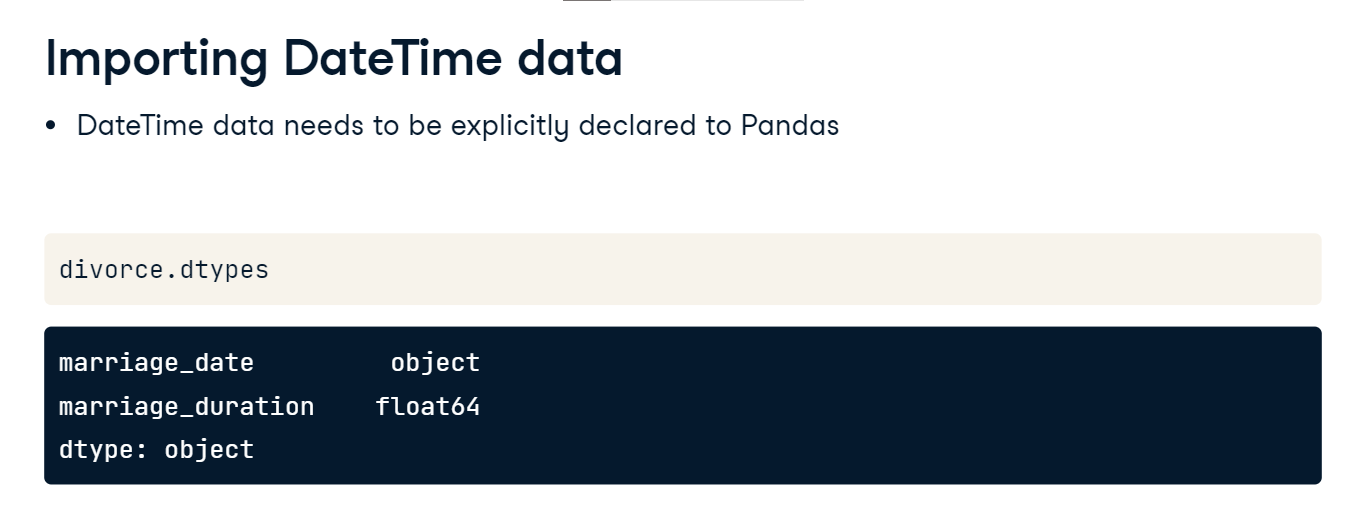
When data includes dates or time values, we'll want to examine whether there might be patterns over time.

**Patterns over time**

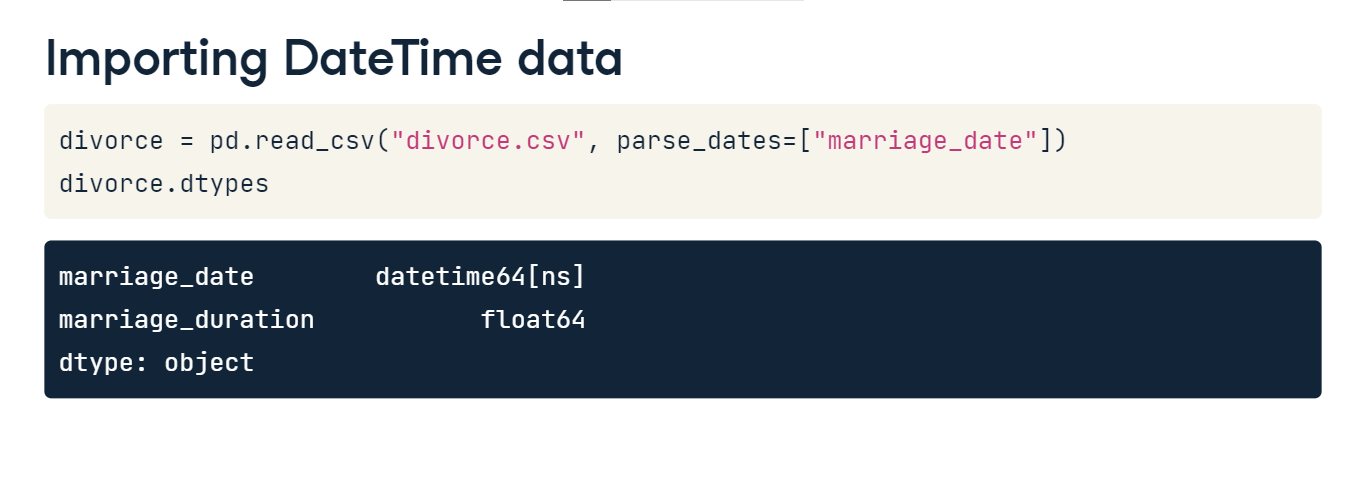
To illustrate, we'll be working with a subset of a dataset about divorce filings taking place in Mexico from 2000 until 2015. This data contains columns for marriage date and marriage duration in years.

**Importing DateTime data**

Before we can begin to look at potential patterns over time, we need to help pandas understand that data in a given column is in fact date or time data. When a CSV file is imported into pandas, date and time data are typically interpreted as strings, as we see here.

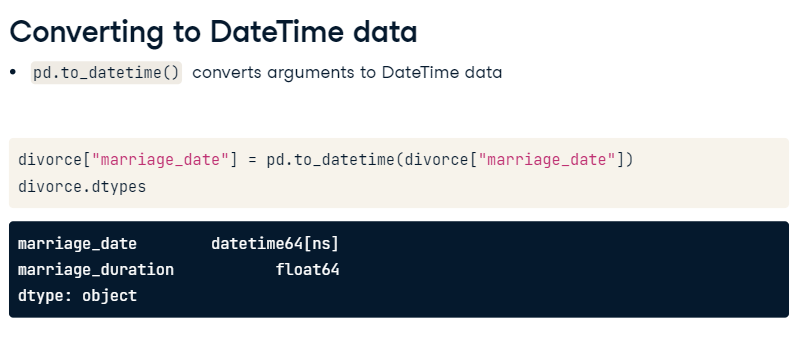


**Importing DateTime data**

We can fix that by adding the parse\_dates keyword argument to the CSV import and setting it equal to a list of column names that should be interpreted as DateTime data. Now, when we check the data types of the imported CSV, the indicated column is a DateTime object. This data type opens up many possibilities for analysis, such as looking at patterns over years, months, or even days of the week. 

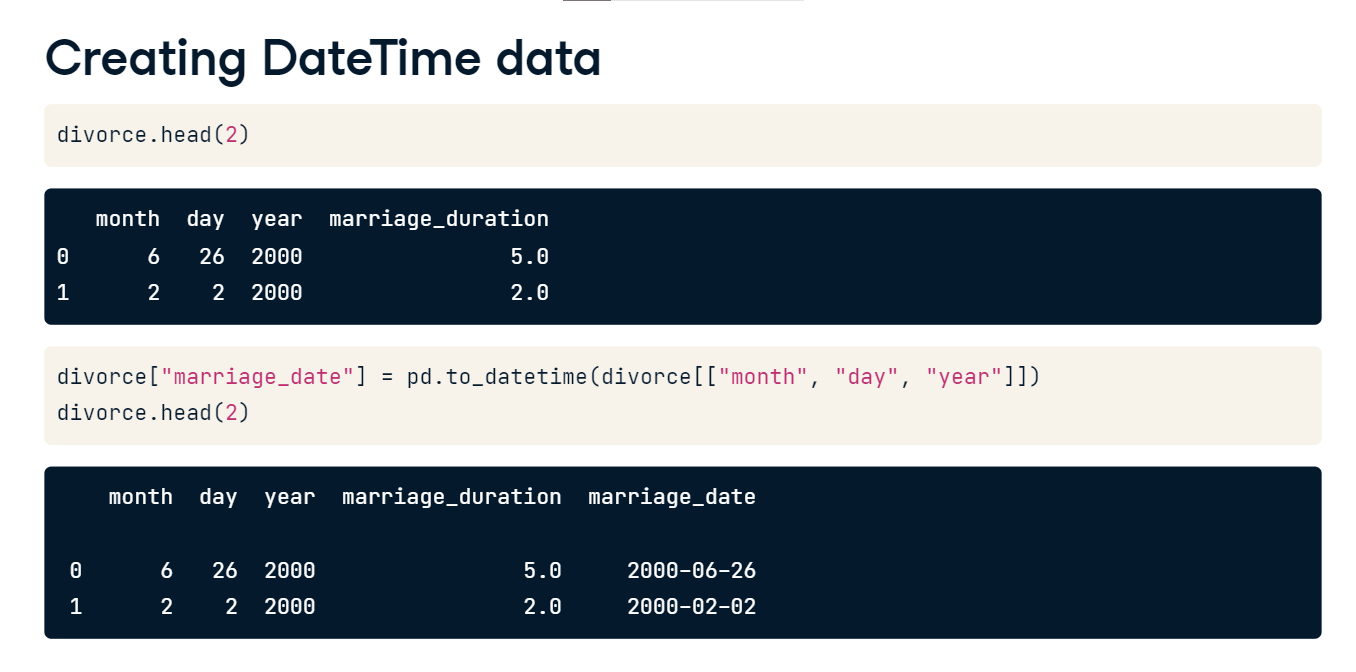
**Converting to DateTime data**

Of course, we may wish to update data types to DateTime data after we import the data. This is possible with pd-dot-to\_datetime, which converts the argument passed to it to DateTime data. Here, we pass the marriage\_date column with values stored as strings to pd-dot-to\_datetime. This returns DateTime data which we save as the new marriage\_date column.



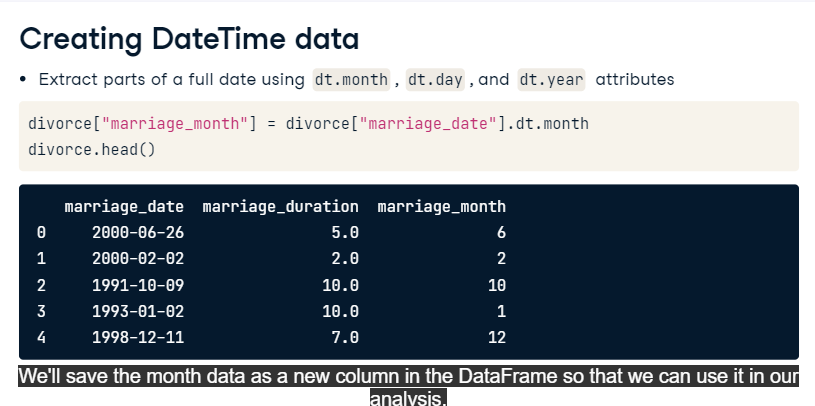
**Creating DateTime data**

pd-dot-to\_datetime has lots of other useful functionality. For example, if a DataFrame has month, day, and year data stored in three different columns, as this one does, we can combine these columns into a single DateTime value by passing them to pd-dot-to\_datetime. Note that for this trick to work, columns must be named "month", "day", and "year", but can appear in any order in the DataFrame.



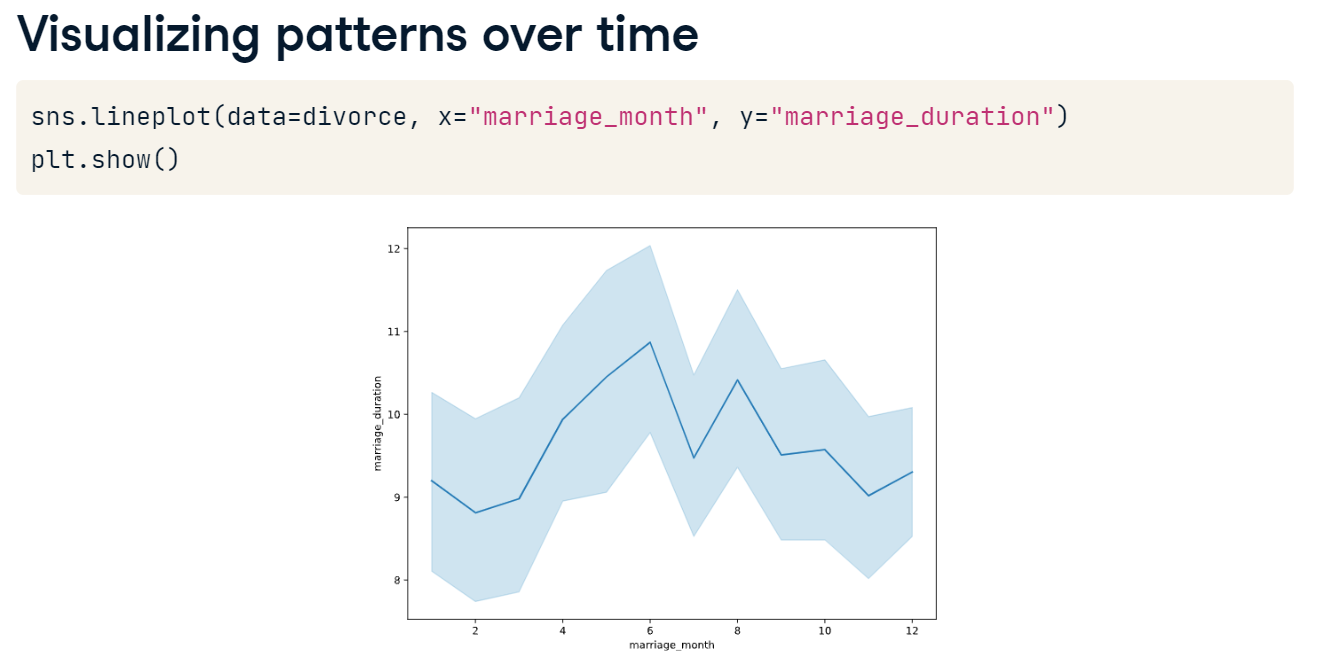
**Creating DateTime data**

Conversely, we might want to extract just the month, day, or year from a column containing a full date. If data is already stored in DateTime format, as marriage\_date is, we can append dot-dt-dot-month to extract the month attribute, for example. We'll save the month data as a new column in the DataFrame so that we can use it in our analysis.



**Visualizing patterns over time**

Line plots are a great way to examine relationships between variables. In Seaborn, line plots aggregate y values at each value of x and show the estimated mean and a confidence interval for that estimate. Perhaps we'd like to check whether there is any relationship between the month that a now-divorced couple got married and the length of their marriage. We can set x equal to the marriage\_month column and y equal to marriage\_duration. The results show some variation in mean marriage duration between months. The blue line represents the mean marriage duration for our dataset, while the confidence intervals in the lighter blue shading indicate the area that, with 95% probability, the population mean duration could fall between. The wide confidence intervals suggest that further analysis is needed!



**Correlation**

Getting a sense of relationships between variables is important for evaluating how data should be used. That's where correlation comes in!

**Correlation**

Correlation describes the direction of the relationship between two variables as well as its strength. Understanding this relationship can help us use variables to predict future outcomes. A quick way to see the pairwise correlation of numeric columns in a DataFrame is to use pandas' dot-corr method. A negative correlation coefficient indicates that as one variable increases, the other decreases. A value closer to zero is indicative of a weak relationship, while values closer to one or negative one indicate stronger relationships. Note that dot-corr calculates the Pearson correlation coefficient, measuring the linear relationship between two variables.

**Correlation heatmaps**

Let's wrap our divorce-dot-corr results in a Seaborn heatmap for quick visual interpretation. A heatmap has the benefit of color coding so that strong positive and negative correlations, represented in deep purple and beige respectively, are easier to spot. Setting the annot argument to True labels the correlation coefficient inside each cell. Here, we can see that marriage year and marriage duration are strongly negatively correlated; in our dataset, marriages in later years are typically shorter.

**Correlation in context**

However, this highlights an important point about correlations: we must always interpret them within the context of our data! Since our dataset is about marriages that ended between 2000 to 2015, marriages that started in earlier years will by definition have a longer duration than those that started in later ones.

**Visualizing relationships**

We also need to be careful to remember that the Pearson coefficient we've been looking at only describes the linear correlation between variables. Variables can have a strong non-linear relationship and a Pearson correlation coefficient of close to zero. Alternatively, data might have a correlation coefficient indicating a strong linear relationship when another relationship, such as quadratic, is actually a better fit for the data. This is why it's important to complement our correlation calculations with scatter plots!

**Scatter plots**

For example, the monthly income of the female partner and the male partner at the time of divorce showed a correlation coefficient of zero-point-three-two in our heatmap. Let's check that this correctly indicates a small positive relationship between the two variables by passing them as x and y arguments to Seaborn's scatterplot function. It looks like the relationship exists but is not particularly strong, just as our heatmap suggested.

**Pairplots**

We can take our scatterplots to the next level with Seaborn's pairplot. When passed a DataFrame, pairplot plots all pairwise relationships between numerical variables in one visualization. On the diagonal from upper left to lower right, we see the distribution of each variable's observations. This is useful for a quick overview of relationships within the dataset. However, having this much information in one visual can be difficult to interpret, especially with big datasets which lead to very small plot labels like the ones we see here.

**Pairplots**

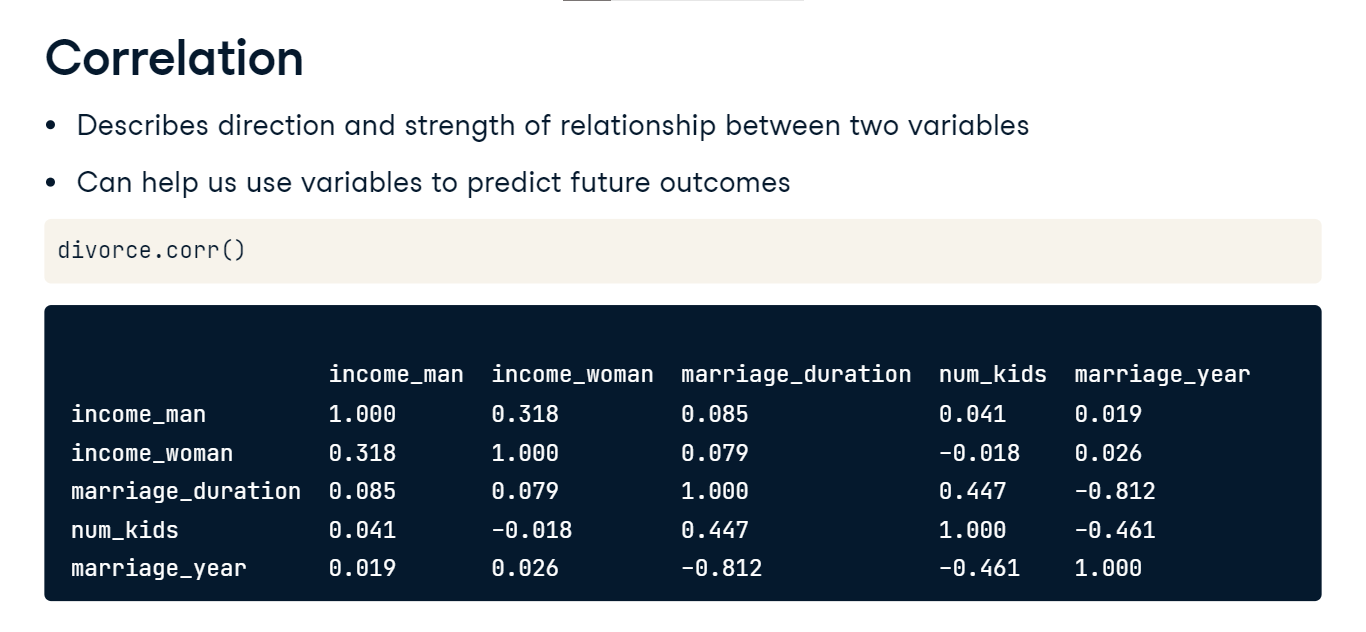
We can limit the number of plotted relationships by setting the vars argument equal to the variables of interest. This visual reassures us that what our correlation coefficients told us was true: variables representing the income of each partner as well as the marriage duration variable all have fairly weak relationships with each other. We also notice in the lower right plot that the distribution of marriage durations includes many shorter marriages and fewer longer marriages.

**Correlation**

Getting a sense of relationships between variables is important for evaluating how data should be used. That's where correlation comes in!

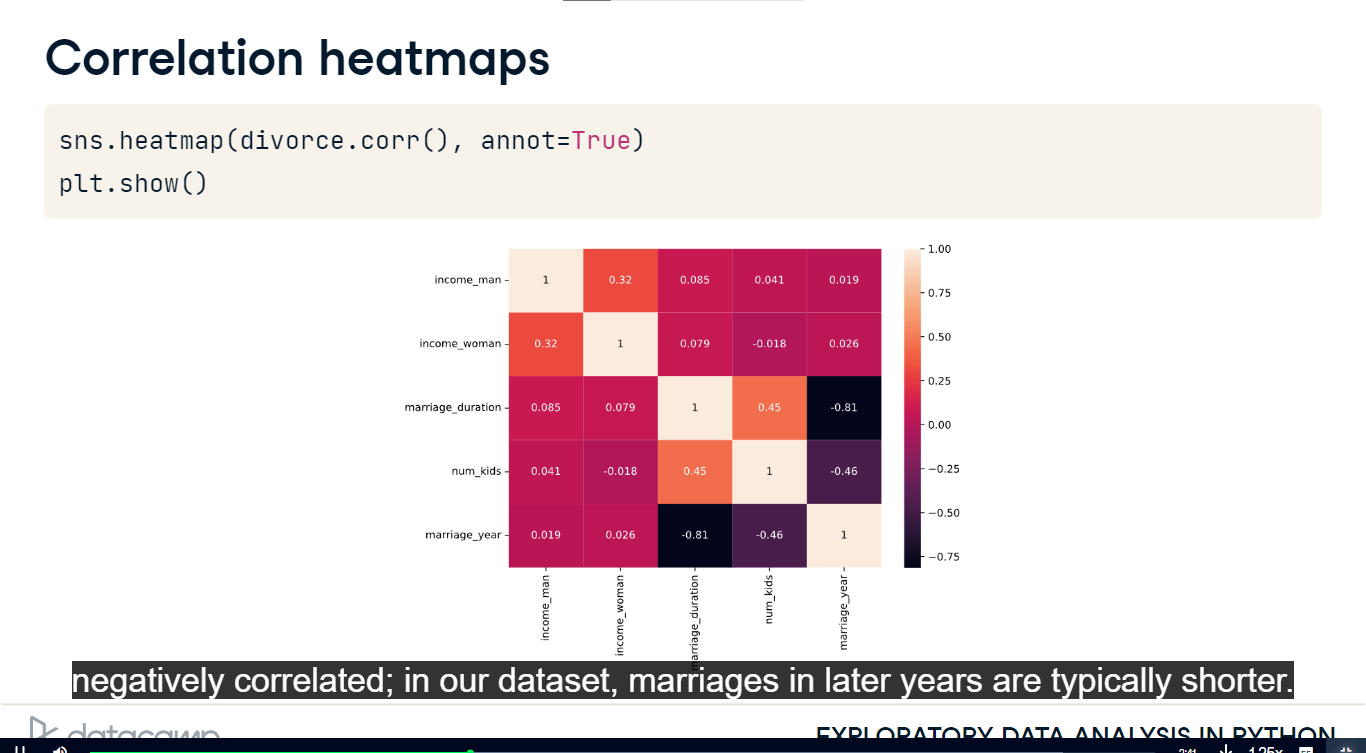
**Correlation**

Correlation describes the direction of the relationship between two variables as well as its strength. Understanding this relationship can help us use variables to predict future outcomes. A quick way to see the pairwise correlation of numeric columns in a DataFrame is to use pandas' dot-corr method. A negative correlation coefficient indicates that as one variable increases, the other decreases. A value closer to zero is indicative of a weak relationship, while values closer to one or negative one indicate stronger relationships. Note that dot-corr calculates the Pearson correlation coefficient, measuring the linear relationship between two variables.



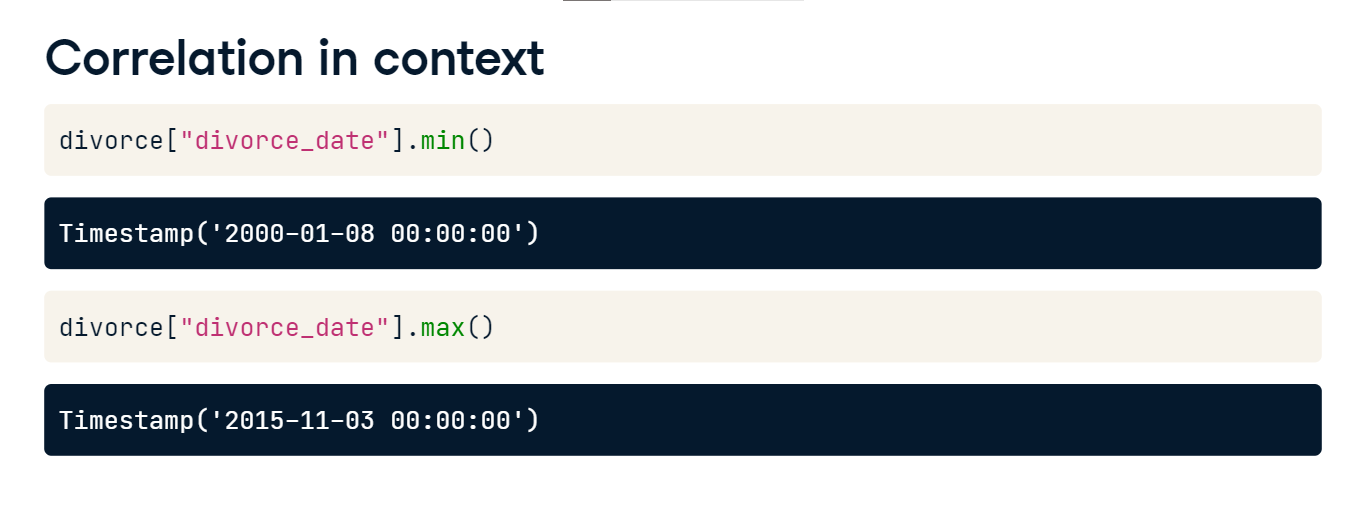
**Correlation heatmaps**

Let's wrap our divorce-dot-corr results in a Seaborn heatmap for quick visual interpretation. A heatmap has the benefit of color coding so that strong positive and negative correlations, represented in deep purple and beige respectively, are easier to spot. Setting the annot argument to True labels the correlation coefficient inside each cell. Here, we can see that marriage year and marriage duration are strongly negatively correlated; in our dataset, marriages in later years are typically shorter.



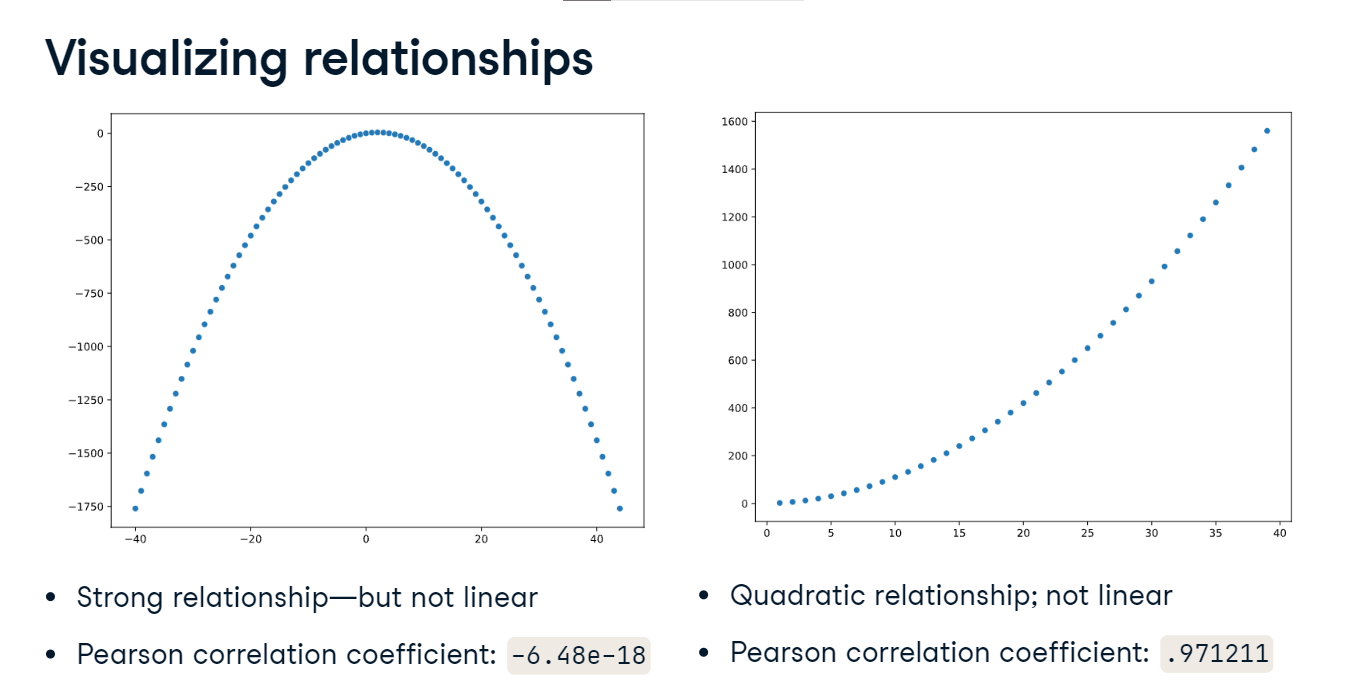
**Correlation in context**

However, this highlights an important point about correlations: we must always interpret them within the context of our data! Since our dataset is about marriages that ended between 2000 to 2015, marriages that started in earlier years will by definition have a longer duration than those that started in later ones.



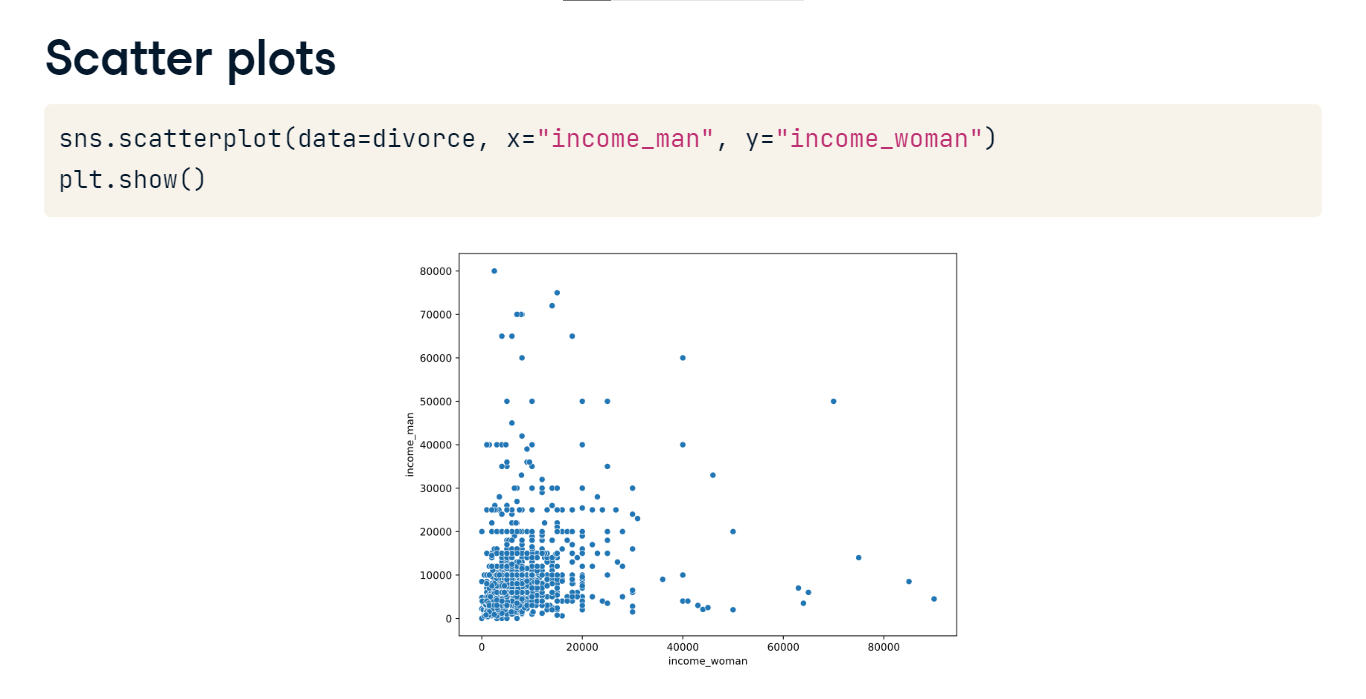
**Visualizing relationships**

We also need to be careful to remember that the Pearson coefficient we've been looking at only describes the linear correlation between variables. Variables can have a strong non-linear relationship and a Pearson correlation coefficient of close to zero. Alternatively, data might have a correlation coefficient indicating a strong linear relationship when another relationship, such as quadratic, is actually a better fit for the data. This is why it's important to complement our correlation calculations with scatter plots!



**Scatter plots**

For example, the monthly income of the female partner and the male partner at the time of divorce showed a correlation coefficient of zero-point-three-two in our heatmap. Let's check that this correctly indicates a small positive relationship between the two variables by passing them as x and y arguments to Seaborn's scatterplot function. It looks like the relationship exists but is not particularly strong, just as our heatmap suggested.



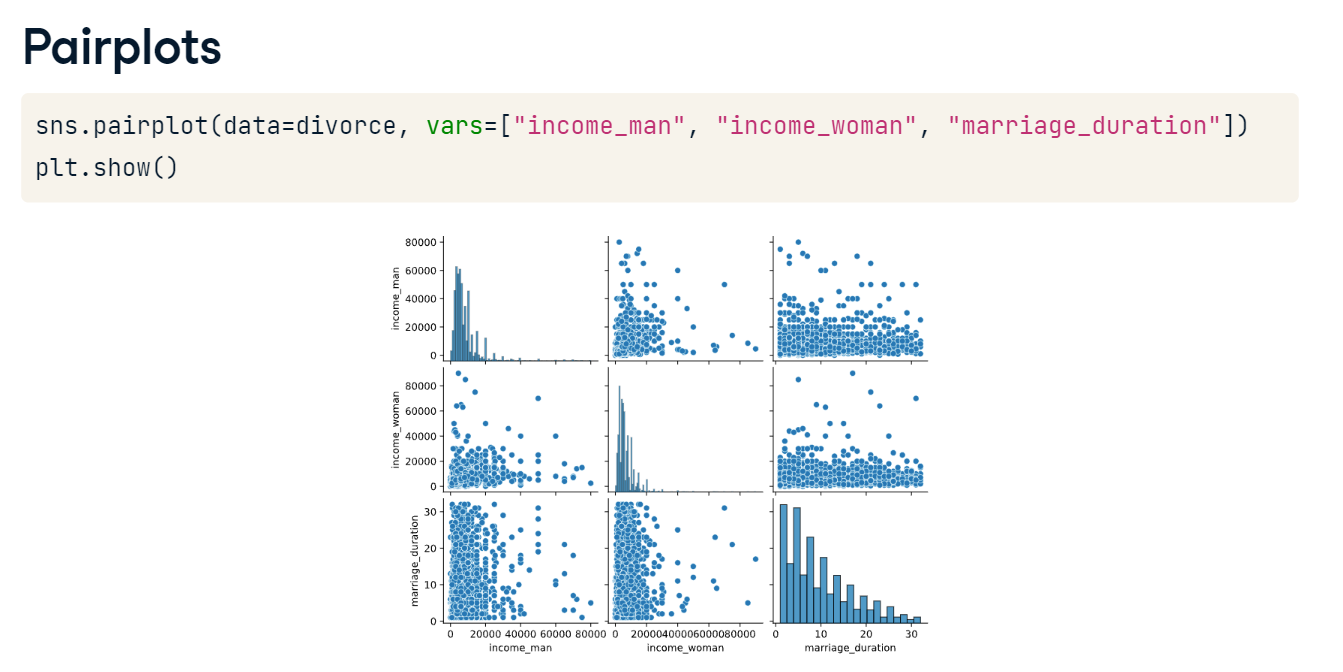
**Pairplots**

We can take our scatterplots to the next level with Seaborn's pairplot. When passed a DataFrame, pairplot plots all pairwise relationships between numerical variables in one visualization. On the diagonal from upper left to lower right, we see the distribution of each variable's observations. This is useful for a quick overview of relationships within the dataset. However, having this much information in one visual can be difficult to interpret, especially with big datasets which lead to very small plot labels like the ones we see here.



**Pairplots**

We can limit the number of plotted relationships by setting the vars argument equal to the variables of interest. This visual reassures us that what our correlation coefficients told us was true: variables representing the income of each partner as well as the marriage duration variable all have fairly weak relationships with each other. We also notice in the lower right plot that the distribution of marriage durations includes many shorter marriages and fewer longer marriages.

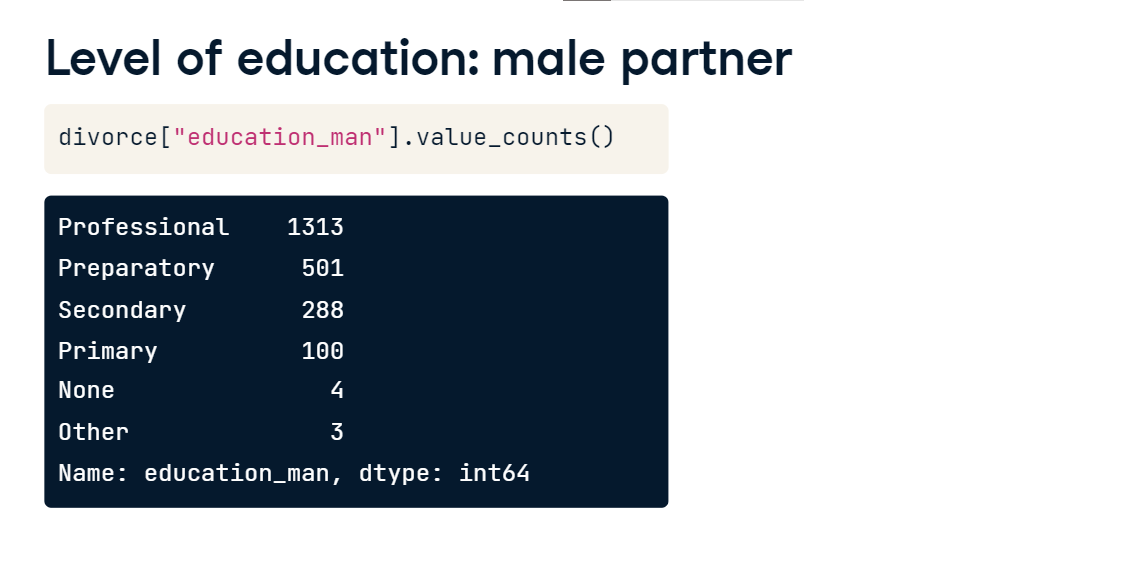


**Factor relationships and distributions**

We previously looked at relationships between numerical variables. Of course, categorical variables, or factors, also have relationships.

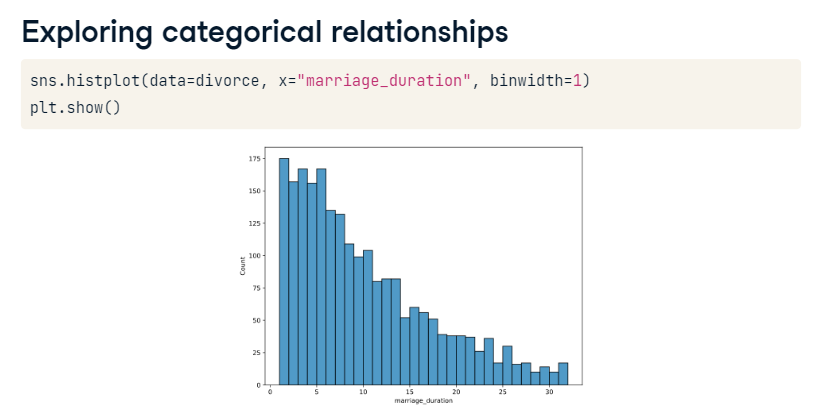
**Level of education: male partner**

We haven't explored the categorical variables related to education level yet. Let's fix that! Checking the value\_counts for education\_man, we see that most men have an education level between primary and professional, with a few men in the "None" or "Other" categories.

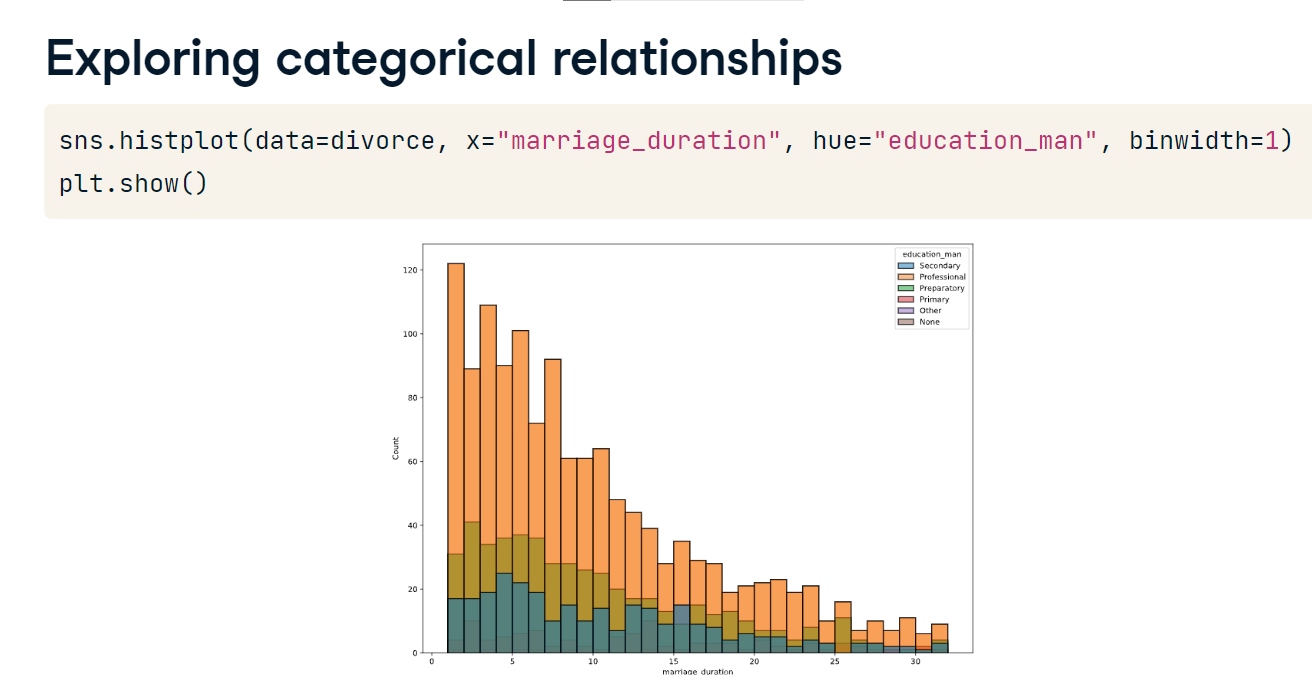


**Exploring categorical relationships**

Categorical variables are harder to summarize numerically, so we often rely on visualizations to explore their relationships. Perhaps we are interested in the relationship between marriage duration and the education level of the man in the dissolved marriage. We could begin by making a histogram of the distribution of marriage duration

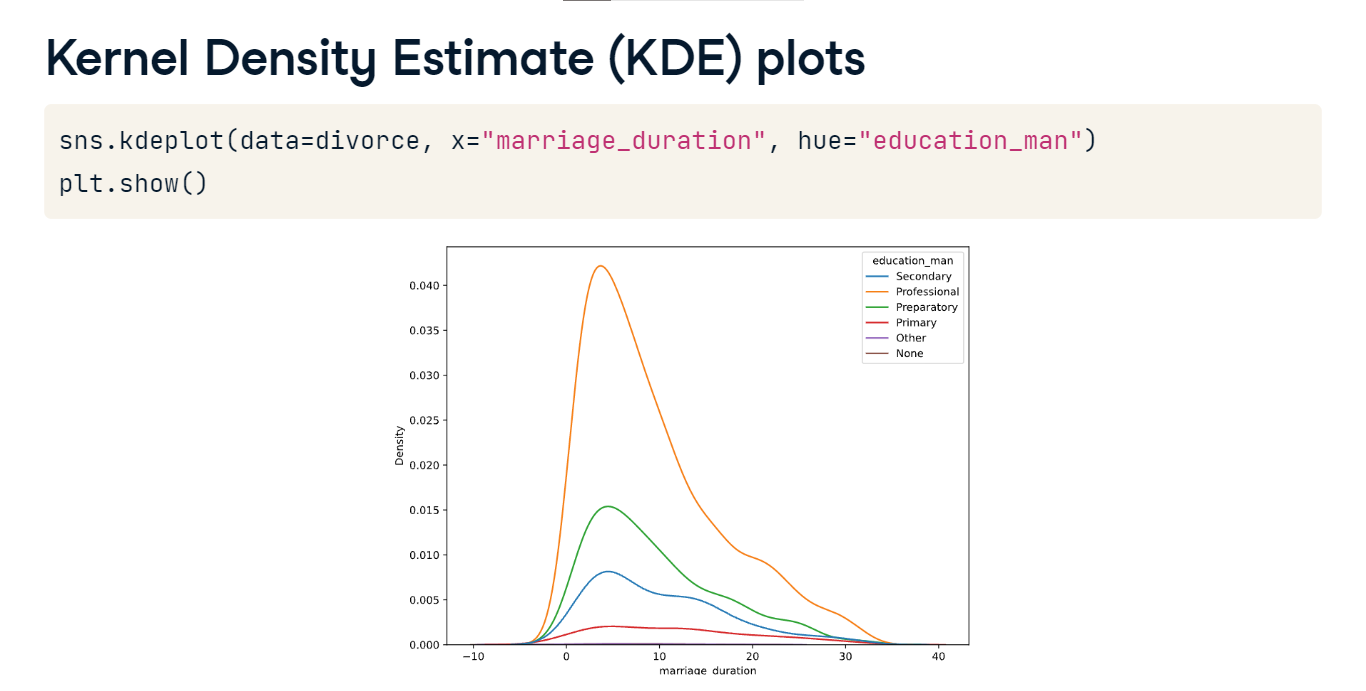
**Exploring categorical relationships**

and then layer in the information we have on male education level by setting education\_man as the hue argument. The resulting histogram reinforces what we saw in value\_counts: we have a lot of information on males with professional-level education. However, because the education levels are stacked on top of each other, the relationship between marriage duration and male education level isn't super clear.



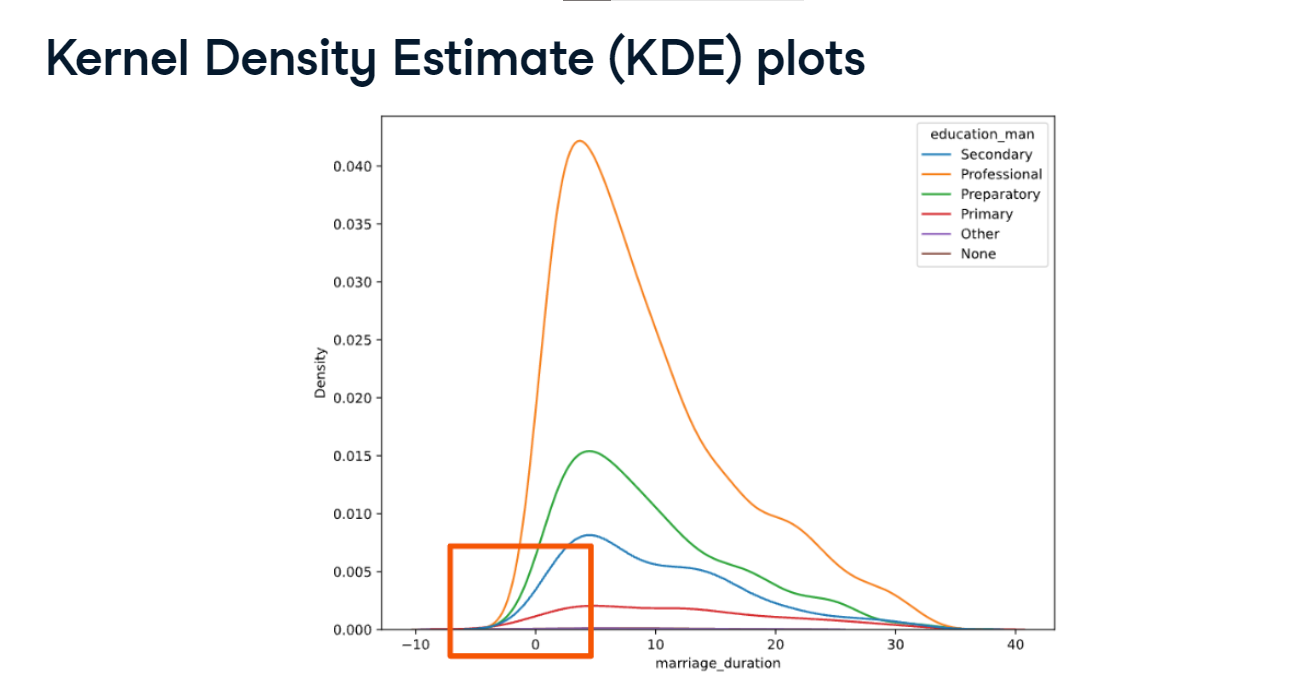
**Kernel Density Estimate (KDE) plots**

Seaborn's Kernel Density Estimate or KDE plots address this issue. Similar to histograms, KDEs allow us to visualize distributions. KDEs are considered more interpretable, though, especially when multiple distributions are shown as they are here. Notice that the location of the peak marriage duration for each level of the male partner's education is more identifiable in this KDE plot than it was in the histogram. However, due to the smoothing algorithm used in KDE plots, the curve can include values that don't make sense, so it's important to set good smoothing parameters.



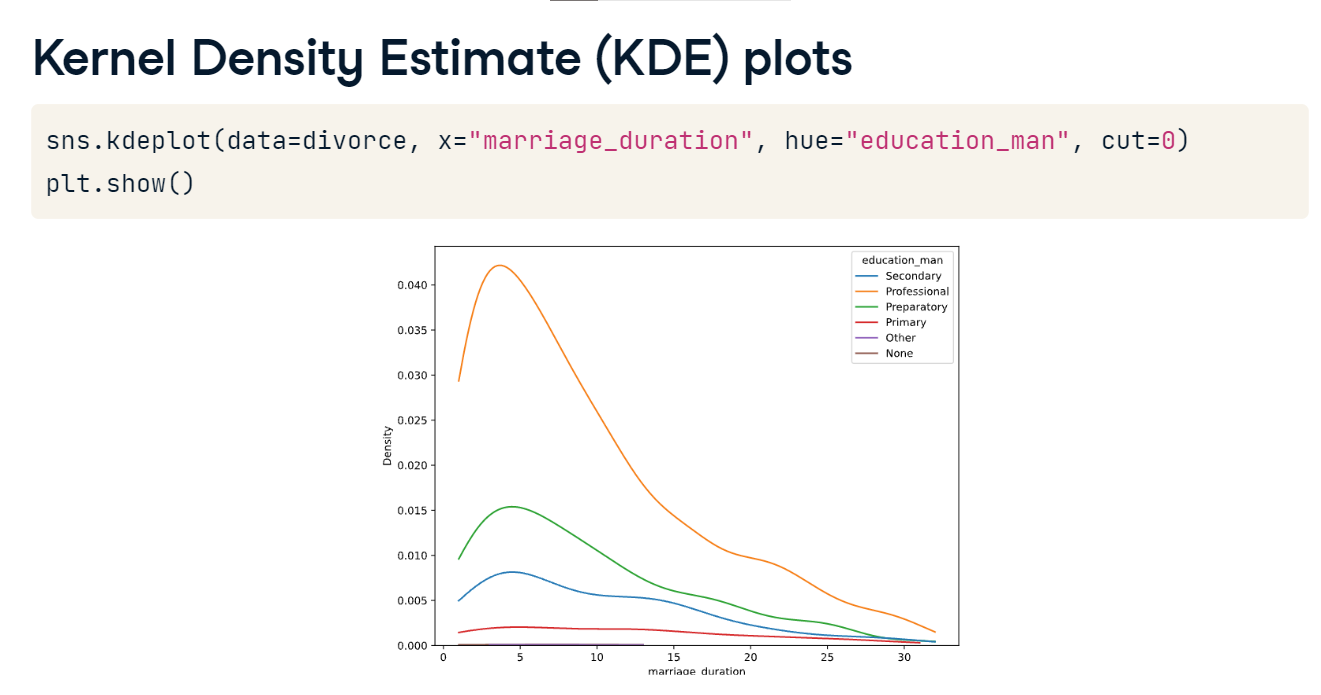
**Kernel Density Estimate (KDE) plots**

Here's an example: zooming in on the KDE plot showing the distribution of male education levels, we can see that the distribution seems to suggest that some couples had marriage durations of less than zero. That's impossible!



**Kernel Density Estimate (KDE) plots**

To fix this, we can use the cut keyword argument. cut tells Seaborn how far past the minimum and maximum data values the curve should go when smoothing is applied. When we set cut equal to zero, the curve will be limited to values between the minimum and maximum x values, here, the minimum and maximum values for marriage duration. The plot now shows only marriage durations greater than or equal to one year, the shortest marriage duration in the dataset.



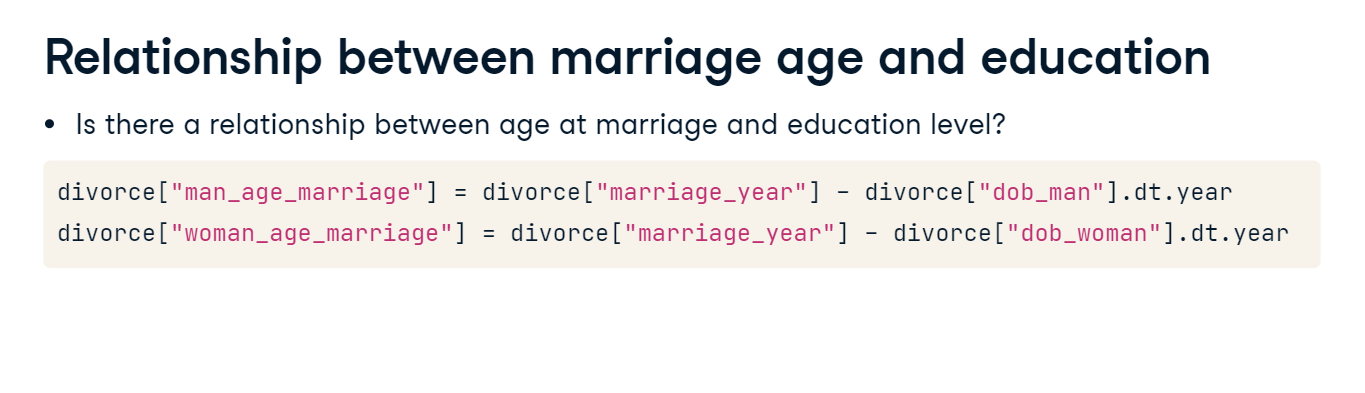
**Cumulative KDE plots**

If we're interested in the cumulative distribution function, we can set the cumulative keyword argument to True. This graph describes the probability that marriage duration is less than or equal to the value on the x-axis for each level of male partner education.



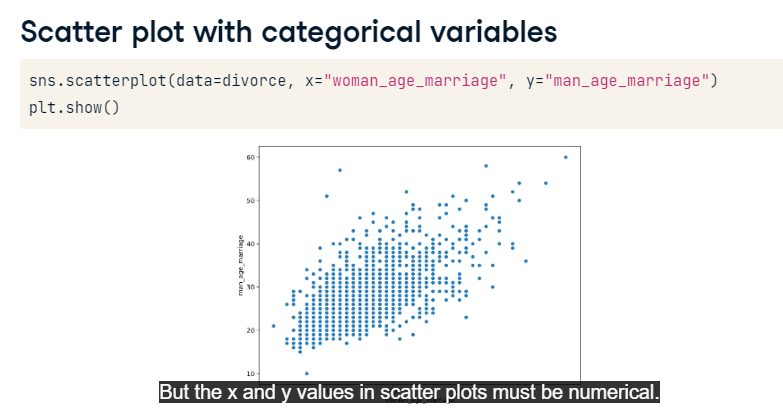
**Relationship between marriage age and education**

Perhaps we are interested in whether divorced couples who got married when they were older typically have higher levels of education. We can create columns representing the approximate age at marriage for men and women by subtracting each partner's birth year from the marriage year.



**Scatter plot with categorical variables**

Then, we create a scatterplot using these variables on the x and y-axis. It looks like there is a positive correlation between them! Indeed, the Pearson correlation coefficient is 0.69. But the x and y values in scatter plots must be numerical. How do we introduce education level into our visual?



**Scatter plot with categorical variables**

One way to do this is to set the hue argument, which assigns a color to each data point based on values in a given column. Here, we set hue equal to education\_man. The results suggest that men with a professional education level, represented with orange dots, may tend to get married later.

