mason-sp21-cds101 / **assignment-4-LiamWhitenack** Private

assignment-4-LiamWhitenack created by GitHub Classroom

0 stars 0 forks

Star Watch

Code Issues Pull requests Actions Projects Wiki Security

master

github-classroom Initial commit … yesterday 1

View code

README.md

Assignment 4: Exploratory Data Analysis of global development

It seems like the world is getting worse and worse (2020, am I right?). But is it?

We can use data to explore actual trends in the world over time. A big proponent of this approach was Hans Rosling of the Gapminder dataset. You can watch a short TED talk about his work here (highly recommended!):

About the data

We will be using the `gapminder` dataset from the `dslabs` package. This dataset contains the following 9 variables (columns):

variable	description
`country`	factor (categorical variable) with 185 levels (values)
`year`	ranges from 1960 to 2016
`infant_mortality`	infant deaths per 1000
`life_expectancy`	life expectancy in years
`fertility`	average number of children per woman
`population`	population of the country
`gdp`	GDP according to World Bankdev
`continent`	factor with 5 levels
`region`	geographical region: factor with 22 levels

Tips and Reminders

- Big tables will overflow the page in this (and future) assignments. Make sure you reduce the size of any tables you print in your PDF to fit within the width of a page, and not to be longer than ~1 page (but it is OK if a smaller table wraps over two pages).
    - You can use the `select()` function to reduce the number of rows, and the `head()` function to show just the first 6 rows.
- You should use the `labs()` function to add titles and axis labels to all your plots in this (and future) assignments.
- If you don't remember how to use a function, you can look up its help page in RStudio by typing `??function_name` in the *Console* (or by Googling the function!)
- Remember to commit your work after each question.

Exercises

1. Open the `gapminder.Rmd` file in RStudio and run the set-up chunk to load the required packages.

   A good first step when working with a new dataset is to look at the actual dataset.

   Take a look at the `gapminder` dataset (e.g. by running `View(gapminder)` in the Console) and answer the following questions:

   i. Which variables in the dataset are categorical?

   ii. Which variables in the dataset are continuous?

   iii. What does each row in the dataset represent?

   Save your answer file, and commit your changes with an appropriate commit message.

2. Another useful step is to calculate some quick statistical summaries of the data. As you learned in the interactive tutorial, we can easily do this with the `group_by` and `summarize` functions. It is helpful to do this because it can warn of us of potential issues with the data such as missing data, or strange values.

   i. Calculate summary statistics (the number of observations, mean, median, standard deviation, interquartile range, minimum value, and maximum value) of the `fertility` variable for each continent (i.e. `group_by` the `continent` variable first).

- Because the column contains missing data, you will need to provide the `na.rm = TRUE` argument to the summary functions, e.g. `mean = mean(fertility, na.rm = TRUE)`.
- Remmember to use the `IQR()` function for the inter-quartile range.
- You can calculate the number of observations using `n()`, and the number of missing observations using `sum(is.na(fertility))`. The fraction missing is the latter divided by the former.

  > **How does `is.na()` work?**
  >
  > `is.na()` is a function that returns a Boolean value (`TRUE` / `FALSE`) depending on whether the input argument is missing (i.e. NA). If the input is a vector or a column, then the output will be a vector of `TRUE` s and `FALSE` s for each item in the input.
  >
  > However, in most programming languages, the Boolean value `TRUE` is equivalent to 1, and `FALSE` to 0. Therefore, we can add up a vector of Boolean values (using R's sum() function) to calculate the number of TRUEs in the vector. As this vector is the output of `is.na()` in our code, this will give us the number of missing values in that column!

- Make sure this table doesn't overrun the right margin when you knit to a PDF! (You might have to shorten any long columns names.)

ii. We cannot calculate the mean, etc., of a categorical variable, but we can still calculate the number of observations in each group and how many are missing.

First `group_by()` the `continent` variable, and then use `summarize()` to calculate the number of observations, and the fraction of the `region` column that is missing. (You should find that there are no missing values for this variable!)

Save and commit your answer to this question.

3. We usually want to look at the variation within each variable by itself before we look at covariation between variables.

   Let's start by looking at the variation within the continuous variables that you identified in the previous exercise.

   Using the cheatsheet to help you, create the following graphs:

i. Create a histogram of the `population` variable. What is the shape of the distribution? Why do you think that most of the data points occur where they do (i.e. what is the real-world interpretation of this graph)?

ii. Create a box plot of the `fertility` variable (using the `geom_boxplot` geom function - note that `fertility` will need to go on the y-axis). What is the shape and where is center of this distribution?

iii. Create a violin plot of the `life_expectancy` variable (using the `geom_violin` function - note that `life_expectancy` will need to go on the y-axis, and use `x = ""` to create a single violin). What is the shape and where is the center of this distribution?

For each graph, make sure to add a title and axis labels using the `labs()` function.

Commit your work.

> **Note**
>
> You may get a warning when creating these graphs that says something like: `Removed 185 rows containing non-finite values`. This just means there were missing values in one or more of the columns you were trying to plot, and so those rows were ommitted before the graph was created.
>
> You can ignore these warnings - ggplot shows them as a helpful hint in case you (the programmer) hadn't realised there were missing data in your dataset.
>
> If they really bother you, you can omit them by adding this option to the code chunk `warning = FALSE` (inside the `{r}`, e.g. `{r, warning = FALSE}`). Note, however, that it is often helpful to leave warnings in, and they may tell us about useful assumptions that we hadn't realzed we are making.

4. Let's also look at the values in the categorical columns `continent` and `region`.

i. Create a bar graph (using the `geom_bar()` geom function) of the `continent` variable. (Don't forget to add a title and axis labels!)

Why are the bars so high? I.e. how can the height of the Americas bar be >2000, when there are only about 195 countries in the world in total? (Hint: think back to your answer for Exercise 1, part iii, and about what each row in the dataset represents.)

ii. Create another bar graph of the `region` variable.

Because there are so many categories in this variable, the x axis labels overlap. We can rotate and reposition them by adjusting the plot's *theme*. Add this line of code to the end of your ggplot code to fix the labels: `+ theme(axis.text.x = element_text(angle = 45, hjust = 1))`, e.g.

```
ggplot() +
  geom_bar(...) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

Commit your work.

> **More on themes**
>
> You can do some pretty cool stuff with themes, and preexisiting themes are often a quick way to make your graphs pop for little effort. If you're interested, check out examples here and here

5. We have looked at the variation within many of these columns individually. However, in exploratory data analysis, we are often interested in exploring the *covariation* between two or more variables.

   For example, the measurements in this dataset are taken at yearly intervals. Comparing the life expectancy of all countries in all years (as we did in Exercise 2, part iii) is not especially useful because we are not comparing similar numbers. For example, comparing the Algerian life expectancy in 1963 with the US life expectancy in 2007 does not make much sense in most scenarios!

   To make these numbers more comparable (so that our graphs are informative rather than confusing), we need to either:

   - compare multiple times within a single country
   - compare multiple countries at a single time

   We will add time into our analyses as a second variable to study the covariation of life expectancy over time.

   i. Although year is technically a continuous number, we may wish to use it as a categorical variable in some situations. Unfortunately R does not know this, and may insist on treating it as a continuous number. Therefore, we will create a new column that holds the year as a categorical variable.

   We can convert a continuous variable into a categorical variable using the `as.factor(...)` function (a *factor* is R's name for a categorical variable).

Use the `mutate()` function to create a new variable called `year_cat`, which contains the `year` variable as a categorical variable (converted using `as.factor(year)`). Store the output dataframe in a new variable, `gapminder_cat`.

ii. Using the `gapminder_cat` dataframe, create a violin plot of the `life_expectancy` variable as you did in Exercise 3 part iii, but now use `x = year_cat` to create a separate violin for every value of `year_cat`.

You should use the `fig.width` option for the code chunk to make this figure bigger and easier to interpret. Inside the `{r}` in the opening line of the code chunk, add this argument: `fig.width = 9` to change the size of any graphs created by this code chunk. I.e. your opening line for this code chunk should look like:

~~~
```{r, fig.width = 9}
~~~

iii. The yearly violins look very different to the violin plot from Exercise 3! What does your new graph show about the *global trend* of life expectancy over time?

iv. Compare the distribution of life expectancy in each violin. (The width of a violin corresponds to the number of observations at that point, just like the height of a histogram bin.)

How has the *distribution* of life expectancy changed over time (e.g. modality and skewness)? What does this mean for the lives of real people in countries around the world?

Commit your work.

6. So far we have looked at the complete dataset for all years. However, sometimes we want to plot only part of the dataset, or otherwise wrangle the dataset somehow before plotting it. In these cases we often want to pipe the dataset through one or more wrangling functions first.

   For example, rather than plotting every single year, we might prefer to look at a single value per decade. One way to do this is to use the `filter()` function to extract just rows for certain years.

   i. Let's look at the distribution of fertility rates in each decade, by creating histograms of the `fertility` variable where the bars are colored by the `year_cat` variable. To prevent having too many years to interpret we will first `filter` just rows from the first year of that decade (e.g. 1960, 1970, 1980, etc.).

   As this is a complicated piece of code, here is a template to get you started:

```
gapminder_cat %>%
  filter(...) %>%
  ggplot() +
    geom_histogram(...)
```

When filling in the ellipses:

- when writing chains of multiple functions like this, it is helpful to get each function working correctly before moving onto the next, e.g. get the `filter` function working correctly before adding in the code for the graph.
- remember that the remainder of 1960 divided by 10 is 0 (as is the remainder of 1970, 1980, etc.).
- set either the `bins` or `binwidth` parameter to a low enough value to smooth any excessive jaggedness of the distribution.
- because we want each year's distribution to be comparable, we need to provide the `position = "identity"` parameter to `geom_histogram` to start each variable from the x axis, and set the `alpha` parameter to a value less that 1 to make each series transparent (a value in the range of 0.2 - 0.5 often works well).
- you will need to `filter` the `year` variable, but use the `year_cat` variable to `fill` (i.e. color) the histogram's bars.

ii. Unfortunately it is quite hard to interpret the histogram you just created! When we have so many overlapping bars of different colors it is almost impossible to compare the different distributions.

For multiple distributions it is better to use a single line to indicate the height of the histogram's bins rather than bars. This type of graph is called a *frequency polygon*, and you should generally use it instead of a histogram when you have more than 2 distributions in the same plot.

Copy your code from part (i) to a new code chunk, but replace `geom_histogram` with `geom_freqpoly`. You should also:

- use `color` instead of `fill` (`color` changes the color of lines/outlines, whereas `fill` sets the internal color of shapes such as a histogram's bars).
- get rid of the `position` and `alpha` parameters.

iii. How has the shape of the fertility distribution changed over time?

Commit your answers.

7. When one of the variables whose covariance we are interested in is categorical, we have two main approaches:

   - we can break the graph down by color, as we did in the previous exercise
   - or we can *facet* over that categorical variable to create subplots

   When we have two categorical variables, we can color by one and facet by the second.

   We facet a ggplot graph by adding either the `facet_wrap()` or the `facet_grid()` function. For example, to facet a histogram by a categorical variable called `example_variable`, we would write code like this:

   ```
   ggplot() +
     geom_histogram(...) +
     facet_wrap(~ example_variable)
   ```

   A few things to note:

   - `facet_wrap()` is added as a *separate*, third function using the `+` operator.
   - we use the *tilde* symbol `~` in front of the variable we want to facet over.

   After examining our graph from the previous question, we might be interested whether the pattern of changing fertility rate is the same in all parts of the world (e.g. is Europe the same as Africa?).

   i. **Your turn:** copy your frequency polygon code from the previous exercise to a new code code chunk and facet it over the `continent` variable. Now we are able to explore the covariance between 3 variables: `fertility`, `year`, and `continent`.

   You should also add the `fig.width` parameter to this code chunk's option inside the `{r}` (try a value between 8 - 10) - it is usually a good idea to do this whenever you create faceted subplots, otherwise the plots will be too small to interpret. I.e. the first line of your code chunk should look like:

   ~~~
    ```{r, fig.width = 9}
   ~~~

   ii. After creating your faceted graph, write a paragraph answering this question: *Are there differences in how fertility has changed over time in different continents, and if so, what are they?* (Here are some things to consider in your answer: Are there any continents in which fertility rates seem to have been more stable? Are there any continents in which fertility rates have not declined as much as other continents? By 2010, are most continents more similar to each other or less?)

Once you have answered this exercise, commit your work.

> **More information on faceting: `facet_wrap` vs `facet_grid`**
>
> What is the difference between the two faceting functions? `facet_wrap` facets over a single variable, and it will automatically decide how many plots to put on each row (although this can be customized).
>
> In contrast, `facet_grid` allows you facet over 2 categorical variables (1 in the column axis, and 1 in the row axis), e.g.: `facet_grid(variable_1 ~ variable_2)`.

8. When we have two continuous variables, the most common way to interpret their covariation is using a scatter plot using the `geom_point` geom function.

   After our analysis in the previous exercises, we might interested in how fertility rates are related to infant mortality. For example, you may have heard people claim that as more children survive, families have fewer children on average (i.e. large families were historically common because there was a high chance of some/all of the children dying before reaching adulthood).

   i. A natural next step for us to take as data scientists is to examine this hypothesis. *Create a scatter plot of `fertility` (y axis) versus `infant_mortality` (x axis).*

   ii. Whoah - that scatter plot has a lot of points! In fact there are so many points that they are obscuring the trend between the two variables, although there definitely looks to be patterns in there. This problem is called *overplotting*.

   We have a number of options to improve this graph. To reduce the density of points, we could use `filter` to reduce the number of years or countries, or we could facet over the region or the continent variables.

   Another option is to make the density of points part of the graph itself. We can do this on the scatter plot by setting a low `alpha`. Copy your code from part i to create a new graph and set `alpha = 0.1`.

   iii. Alternatively, we could use a geom function that colors the regions of the graph by density, such as `geom_bin2d`, which creates a type of plot called a *heatmap*.

   Copy your code from part i, and modify it to use the `geom_bin2d` function instead of `geom_point` (you should be able to leave the parameters inside the function unchanged).

iv. What can you deduce about the relationship between infant mortality and fertility from these two graphs? Are there any patterns that are apparent in the scatter plot that are not obvious in the heatmap, or vice versa?

Commit your work.

9. Your go! Create a new graph exploring this dataset, and write a paragraph interpreting any patterns that you see.

   A few tips:

   - Try a few different graphs before settling on one that you think is impressive or informative. EDA is not an exact science, and we often have to play around with different graphs to find the one that works best. Use the "What graph" guide to give you some ideas.
   - You might want to refer back to the "What graph?" cheatsheet.
   - When interpreting the graph, think about both:
     - the statistical pattern in the data (centrality, modality, variance, covariance, density, etc.)
     - but also what those patterns mean for the actual world around us. After all, that's the whole reason for doing these analyses in the first place!
   - There are a lot of points in this dataset - depending on the type of graph that you create, you might want to `filter` the data to subset for a particular year(s) and/or country(s) until a clear pattern emerges.

   Once you are happy with your visualization, commit your work.

How to submit

To submit your assignment, follow the two steps below. Your assignment will be graded for credit **after** you've completed both steps!

1. Save, commit, and push your completed R Markdown file so that everything is synchronized to GitHub. If you do this right, then you will be able to view your completed file on the GitHub website.
2. Knit your R Markdown document to the PDF format, export (download) the PDF file from RStudio Server, and then upload it to *Assignment 4* posting on Blackboard.

Cheatsheets

You are encouraged to review and keep the following cheatsheets handy while working on this assignment:

- What graph should I make?
- dplyr cheatsheet
- ggplot2 cheatsheet
- RStudio cheatsheet
- R Markdown cheatsheet
- R Markdown reference

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package