# Exploring the Kaggle Data Science Survey

**Project Description**

When beginning a career in data science, one often wonders what programming tools and languages are being used in the industry, and what skills one should learn first. By exploring the 2017 Kaggle Data Science Survey results, you can learn about the tools used by 10,000+ people in the professional data science community.

Before starting this project, you should be comfortable manipulating data frames and have some experience working with the tidyverse packages dplyr, tidyr, and ggplot2. We recommend that you complete [**Introduction to the Tidyverse**](https://www.datacamp.com/courses/introduction-to-the-tidyverse), and [**Data Visualization with ggplot2 (Part 1)**](https://www.datacamp.com/courses/data-visualization-with-ggplot2-1) before starting this project.

This project uses a subset of the [**2017 Kaggle Machine Learning and Data Science Survey**](https://www.kaggle.com/kaggle/kaggle-survey-2017?utm_medium=partner&utm_source=datacamp.com&utm_campaign=ml+survey+case+study) dataset. If you want to know more about the tools and techniques Kaggle participants use, check out the full [**report of the Kaggle 2017 survey results**](https://www.kaggle.com/amberthomas/kaggle-2017-survey-results?utm_medium=partner&utm_source=datacamp.com&utm_campaign=ml+survey+case+study).

**Task 1: Instructions**

Load the data and look at the first 10 responses.

* Load the tidyverse package.
* Using read\_csv, load datasets/kagglesurvey.csv and assign it to the variable responses.
* Print the first 10 entries of responses.

**Good to Know**

The tidyverse package automatically loads in dplyr, ggplot2, readr, tidyr, and a few other helpful packages. Learn more about the tidyverse [**here**](https://www.tidyverse.org/).

Before starting this project you should be comfortable manipulating data frames and have some experience working with the tidyverse packages dplyr, tidyr, and ggplot2. We recommend that you have completed at least one of the following courses:

* [**Introduction to the Tidyverse**](https://www.datacamp.com/courses/introduction-to-the-tidyverse)
* [**Data Visualization with ggplot2 (Part 1)**](https://www.datacamp.com/courses/data-visualization-with-ggplot2-1)

Code examples will frequently utilize the pipe operator (%>%). More information about using the pipe operator in R can be found [**here**](https://www.datacamp.com/community/tutorials/pipe-r-tutorial)(<https://www.tidyverse.org/>).

Take Hint

**Task 2: Instructions**

Split the tools each respondent uses at work into separate rows.

* Print the tools and languages used by the first respondent (found in column 2: WorkToolsSelect).
* Copy responses into a new data frame called tools.
* Add a new column to tools called work\_tools. Split the WorkToolsSelect column at the commas and unnest the new column to fill work\_tools.
* View the first 6 rows of tools.

After looking at the first respondent's tool-use, you'll see that this survey-taker listed multiple tools which are each separated by a comma. To learn how many people use each tool, we need to separate out all of the tools used by each individual. There are several ways to complete this task, but we recommend using the base R function strsplit() to separate the tools at each comma. Since that will leave you with a list inside of your data frame, we can use the tidyr function unnest() to separate each list item into a new row. You can see that solution and others in [**this Stack Overflow Answer**](https://stackoverflow.com/a/27360900/7227814).

Take Hint

**Task 3: Instructions**

Find the number of respondents that use each language or tool.

* Create a new data frame from tools and call it tool\_count.
* Within tool\_count, group the data by work\_tools, then use the summarise() function in dplyr to calculate the number of responses in each group.
* Sort tool\_count so that the most popular tools are at the top.
* Print the first 6 results of tool\_count.

Your final data frame should have only two columns: work\_tools and the count of users.

If you're unsure about how to use the group\_by(), summarise(), and arrange() functions in dplyr, check out this [**cheat sheet**](https://www.rstudio.com/wp-content/uploads/2015/02/data-wrangling-cheatsheet.pdf).

Take Hint

**Task 4: Instructions**

Create a bar chart that displays tool popularity.

* Use ggplot2 to create a bar chart of work\_tools in the tool\_count data frame. Arrange the bars so that the tallest are on the far right.
* Rotate the bar labels 90 degrees.

If you need a refresher on how to make figures in ggplot2, check out [**this cheat sheet**](https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf) or this chapter from [**the ggplot cookbook**](http://www.cookbook-r.com/Graphs/Bar_and_line_graphs_(ggplot2)/).

If my\_data has columns a and b where b is a number, this is how to make a bar chart where the bar height corresponds to the numbers in column b:

ggplot(my\_data, aes(x = a, y = b)) +

geom\_line(stat = "identity")

Take Hint

**Task 5: Instructions**

Calculate the number of respondents that use R, Python, and both tools.

* Create a new data frame called debate\_tools from responses.
* Create a new column called language\_preference which should be set to
  + **"R"** if WorkToolsSelect contains "R" but not "Python".
  + **"Python"** if WorkToolsSelect contains "Python" but not "R".
  + **"both"** if WorkToolsSelect contains both "R" and "Python".
  + **"neither"** if WorkToolsSelect contains neither "R" nor "Python".
* Print the first 6 rows of debate\_tools.

While you can use nested ifelse() statements to accomplish this task, we recommend using the case\_when() function from dplyr. You can find more information about case\_when() [**here**](https://www.rdocumentation.org/packages/dplyr/versions/0.7.3/topics/case_when).

To determine if the comma-separated string in the WorkToolsSelect column contains R or Python, we recommend using the base R grepl() function. More information on grepl() can be found [**here**](https://www.rdocumentation.org/packages/base/versions/3.4.3/topics/grep).

Take Hint

**Task 6: Instructions**

Calculate total number of users that use R, Python, or both, and plot the results.

* Create new data frame called debate\_plot from debate\_tools.
* Group by language\_preference and then use the summarise() function to calculate the number of each response.
* Remove the row for respondents that use "neither" R nor Python.
* Create a bar chart of language preference counts using ggplot.

To remove a row from a data frame, you can use dplyr's filter() function.

Take Hint

**Task 7: Instructions**

Find language recommendations for users that use R, Python, or both languages.

* Create a new data frame named recommendations from debate\_tools.
* Group recommendations by language\_preference and then LanguageRecommendationSelect. Count the number of recommendations for each language within the group.
* Filter the data (using dplyr's filter) to exclude any empty responses in LanguageRecommendationSelect and include only the top 4 most common recommendations for each language preference.

To filter the data in this way, it is probably helpful to arrange() the data from most to least popular. Add a column which counts the row number within each group (using the row\_number() function) and then filtering the row numbers to be less than or equal to 4. Learn more about row\_number() and the other dplyr ranking functions [**here**](http://dplyr.tidyverse.org/reference/ranking.html).

**HINT**

Here is a bit more code scaffold to help get you going:

# Grouping by language\_preference and then LanguageRecommendationSelect

recommendations <- recommendations %>%

group\_by(....) %>%

summarise(....) %>%

# Removing empty responses and include the top recommendations

filter(....) %>%

arrange(...., desc(....)) %>%

mutate(....) %>%

filter(....)

**Task 8: Instructions**

Create a faceted plot showing the top 4 language recommendations from users of R, Python, both, and neither.

* Use the ggplot function facet\_wrap() to create a faceted plot of recommendation frequency. You should get four sub-plots, one for each type of value in the language\_preference column.

The facet\_wrap() function creates a series of plots; you just need to tell it how it should split them. For instance, if you had data about different countries and you wanted a separate plot for each country, you could do something like this:

ggplot(my\_data, aes(x = x, y = y)) +

geom\_bar(stat = "identity") +

facet\_wrap(~country)

Take Hint

**Task 9: Instructions**

Would R-users find the following statement TRUE or FALSE?

* R is the language I recommend for new data scientists.

Congratulations! You've made it to the end of this project!

If you haven't already, try to *check* your project by clicking the **Check Project** button.

If you're looking to learn more from this dataset, you can find the questions we explored here and many others on [**Kaggle**](https://www.kaggle.com/kaggle/kaggle-survey-2017?utm_medium=partner&utm_source=datacamp.com&utm_campaign=ml+survey+case+study). You can also explore [**my analysis**](https://www.kaggle.com/amberthomas/kaggle-2017-survey-results?utm_medium=partner&utm_source=datacamp.com&utm_campaign=ml+survey+case+study) of the full dataset as well as the analysis by [**many other**](https://www.kaggle.com/kaggle/kaggle-survey-2017/kernels?utm_medium=partner&utm_source=datacamp.com&utm_campaign=ml+survey+case+study) talented data lovers.

Good luck! :)

Take Hint