STAT 19000 Project 1

Topics: jupyter notebooks

Motivation: RStudio is only one very popular tool to do data analysis and create associated reports. Another very popular tool is called Jupyter Notebook. Similarly to how RStudio provides an interface to edit and knit R Markdown files (which we will cover at a later date), Jupter Notebook provides an environment to edit Jupyter notebooks. The use of Jupter notebooks has grown more than 100% year-over-year for the last three years (as measured by the number of repositories with Jupyter identified as a primary language between 2016 and 2019). source

Context: We have used and become very familiar with RStudio through completing projects last semester. We'd be remiss to not explore other popular tools, namely, Jupyter Notebook.

Scope: Jupyter Notebook is a powerful IDE for working with and doing analysis with Jupyter notebooks. Knowing how to use these tools will enable quick adaptation to the many work environments that use these tools for reproducible analysis.

Don't forget the very useful documentation shortcut `?`. To use, simply type `?` in the console, followed by the name of the function you are interested in.

You can also look for package documentation by using `help(package=PACKAGENAME)`.

You can find some examples that may be useful provided in `/class/datamine/data/spring2020/stat19000project01exampl`

Use the template found here or on scholar: `/class/datamine/data/spring2020/stat19000project01template.ipynb` to submit your solutions.

A good resource for a summary of useful techniques for working in Jupyter Notebooks can be found here.

Question 1: Jupyter Notebook

This semester, for the first few weeks of the semester, we will work in Jupyter Notebooks. To prepare to do this, please open a connection to Scholar using ThinLinc. One you are logged onto Scholar, open a terminal, and type:

```
source /class/datamine/data/examples/setup.sh
```

Now you are ready to work in Jupyter Notebooks. Open a web browser of your choice (it can even be on your computer, rather than on Scholar), and go to this URL:

https://notebook.scholar.rcac.purdue.edu

The first time that you do this, after you login (using your Career username and Career password, without BoilerKey), please click "New" and make sure that the option "R 3.6 (Scholar)" appears there.

1a. *(1 pt)* Open a new notebook using our "R 3.6 (Scholar)" option. Call this notebook `project1.ipynb`. Install the popular R package `ggplot2`. Open a new Jupyter notebook, create a Code cell, and use the `library()` function to load `ggplot2`. Use `ggplot2` to create a density plot of the column `Petal.Length` from the built-in dataset `iris` with a light blue fill color. Make sure to run the cell to see the output.

Hint: You can find some basic ggplot examples here.

1b. *(1 pt)* Insert a new cell before the cell that creates our density plot. This time, instead of a Code cell, make a Markdown cell. Give the cell an H1 title, "Project 1 Solutions", and some subtext, "by Firstname Lastname". Run the cell, what happens?

1c. *(1 pt)* One cool feature of Jupyter Notebook is that you can export your work in various formats. List the file extensions we can export our notebook to.

1d. *(1 pt)* Sometimes it can be useful to know what line number you are on in a cell. Use the menu and toggle the line numbers. Once complete, take a screenshot that includes the line numbers and attach it to your project submission as `linenumbers.(png/jpeg/etc)`.

Note: You may find it useful to enable scrolling for outputs as well. To do so, select Cell > All Output > Toggle Scrolling.

Question 2: exploring notebooks

Answer the following questions, 1 answer per cell. These questions should look familiar, we just want to get the hang of the notebook interface. Take the time to look through the menu's and click around.

2a. Read in the `5000_transactions.csv` data (from 8451) into a data frame to be called `myDF`.

2b. *(.5 pts)* Split the data frame `myDF`, using the `STORE_R` column, and store the results of the split into a new variable called `myresults`. Use the `split` command to achieve this. Remember that we can read about the split command using: `?split`

2c. *(.5 pts)* What is the class of `myresults`? What is the length of `myresults`? What are the names of `myresults`? (Use `class`, `length`, and `names` on `myresults`.)

2d. *(.5 pts)* Check the dimensions (`dim`) and the head of `myresults[["CENTRAL"]]`.

2e. *(.5 pts)* Now manually make a data frame that has all of the same columns as `myDF` but only has rows for which `myDF$STORE_R` is equal to `"CENTRAL"`:

```
centralresults <- myDF[myDF$STORE_R == "CENTRAL", ]
```

Verify that the `dim` and `head` of `myresults[["CENTRAL"]]` and `centralresults` look the same.

Question 3: student loans

3a. Use the `read.csv()` function to directly read in the data found at https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2019/2019-11-26/loans.csv. Call this data.frame `loans`. Use the `head()` function to disply the first 3 lines of the new data.

3b. *(1 pt)* Use the `subset()` function in R to get the data for 2017 and 2018, and only include agencies where either the total is greater than or equal to 1 million. Keep only the first 2 and last 2 columns. Call this data.frame `sloans`.

3c. *(1 pt)* Write two lines of code. The first line of code should find the row where `wage_garnishments` is at its minimum from our subset, and should print this minimum value of `wage_garnishments`. (Hint: For the row where `wage_garnishments` is at its minimum, the associated value for `voluntary_payments` is 238059.5.)

The second line of code should find the row where `voluntary_payments` is at its minimum from our subset, and should print this value of `voluntary_payments`. (Hint: For the row where `voluntary_payments` is at its minimum, the associated value for `wage_garnishments` should be 387079.8.)

3d. *(1 pt)* Wage garnishments are when a loaning agency gets permission to force a company to remove part of a paycheck automatically to pay back owed debts. Let's create a new column called `ratio_forced` that shows the ratio of `wage_garnishments/voluntary_payments`. What agencies are responsible for the lowest and highest ratios?

3e. *(1 pt)* Create a pie chart using the `pie()` function that shows the sum of wage garnishments vs the sum of voluntary payments. Make the colors "tomato" and "lightblue" respectively.

Project Submission:

Submit your solutions for the project at this URL: https://classroom.github.com/a/Mx-30vo8 using the instructions found in the GitHub Classroom instructions folder on Blackboard.

Important note: Make sure you submit your solutions in both .ipynb and .html formats.