ECP 3004: Python for Business Analytics
Department of Economics
College of Business
University of Central Florida
Spring 2021

Final Examination

Due Monday, May 3, 2021 at 6:59 PM
in your GitHub repository

Instructions:

Complete this examination within the space on your *private* GitHub repo in a folder called `final_exam`. In this folder, save your answers for Questions 1 and 2 in a script `my_final_module.py`, following the sample script in the folder `final_exam` in the course repository. Save your answers for Question 3 in a script `my_final_Q3.py`, following the sample script in the same folder. When you are finished, submit it by uploading your files to your GitHub repo using any one of the approaches outlined in Question 4. As this is an examination, you are NOT free to discuss your approach to each question with your classmates and you must upload your own work.

Question 1:

Follow the function design recipe to define functions for all of the following Exercises. For each function, create three examples to test your functions. Record the definitions in the sample script `my_final_module.py`. Together, the function definitions will form a module called `my_final_module` that you can read in and test using another script.

Under time constraints, the emphasis is on creating function definitions and testing with examples. You are not responsible for error handling and can assume that the user will follow the preconditions and type contract. Refer to the partial docstrings that are already included in `my_final_module.py`.

Exercise 1 Write a function `ln_check(x, a)` that calculates the difference between $ln(x)$ and some candidate value a, which is a guess of the value of $ln(x)$. The function $\ln(x)$ is calculated with the predefined logarithmic function `math.log()`.

Exercise 2 Calculate the number e, the base of the natural logarithm, using the function `ln_check(x, a)`. Note that $e^1 = e$ so that $\ln(e) = 1$. Calculate the number e by finding the root of the function $f(x) = \ln(x) - 1$, by writing a function `calc_e(x_0, max_iter, tol)` Your function can use any algorithm, as long as you solve for the value using the `ln_check` function: Newton's method, the secant method and the bisection method are all possible choices but you may not use a built-in function to calculate e directly.

Exercise 3 Write a function `SSR_conc(beta_1, y, x)` that calculates the sum of squared residuals

$$SSR\_conc(y, x, \beta_1) = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2.$$

Note that there is only one regression parameter β_1, as the intercept β_0 will be concentrated out. That is, given a candidate estimate β_1, use the relationship $\beta_0 = \bar{y} - \bar{x}\beta_1$ to calculate β_0 within your function.

Exercise 4 Write a function `ols_slope_conc(y, x)` that calculates the estimated slope coefficient $\hat{\beta}_1$ of the bivariate linear regression model by minimizing the concentrated sum of squared residuals, `SSR_conc()`. You may use any method to perform the optimization, including built-in optimization algorithms, but you may not use a formula to calculate β_1 directly.

Question 2:

For all of the Exercises in Question 1, use your examples to test the functions you defined. Since the examples are all contained within the docstrings of your functions, use the `doctest.testmod()` function within the `doctest` module to test your functions automatically at the bottom of your module.

Don't worry about false alarms: if there are some "failures" that are only different in the smaller decimal places, then your function is good enough. It is much more important that your function runs without throwing an error.

Question 3:

Complete the following exercises by modifying the script `my_final_Q3.py`, within the folder `final_exam` in the course repository. Save your answers in a script `my_final_Q3.py`, in a folder called `final_exam` in your GitHub repository.

Your solutions can be obtained using any method of your choosing, as long as it is coded in Python. In particular, you may use Python modules for interacting with databases or data frames to obtain your solutions.

a) Read in the data from the file `US_state_roads.csv`, which contains two numeric variables, along with the names of the US states. The first variable is the total distance of all roads in each state, measured as lane-miles, i.e., a four-lane highway six miles long contributes 24 lane-miles. The second variable is the maximum posted speed limit, in miles per hour, on any of the roads in each state.

b) List the states and speed limits for the states with speed limits greater than or equal to 75 mph.

c) i. Calculate the average lane-miles across all states.

ii. Calculate the average lane-miles for the states with speed limits greater than or equal to 75 mph.

d) Read in the data from the file `US_state_pop_area.csv`, which is a listing of population figures and land area for US states and territories. It is one of the datasets we used for Assignment 8.

e) i. Calculate a list of the state names and the number of lane miles per person. List them in increasing order from highest to lowest lane miles per person.

ii. Calculate a list of the state names and the number of lane miles per square mile of land area. List them in increasing order from highest to lowest lane miles per square mile.

Question 4:

Push your completed files to your GitHub repository following one of these three methods.

Method 1: In a Browser

Upload your code to your GitHub repo using the interface in a browser.

1. Browse to your `assignment_0X` folder in your repository ("X" corresponds to Assignment X.).
2. Click on the "Add file" button and select "Upload files" from the drop-down menu.
3. Revise the generic message "Added files via upload" to leave a more specific message. You can also add a description of what you are uploading in the field marked "Add an optional extended description..."
4. Press "'Commit changes," leaving the button set to "Commit directly to the `main` branch."

Method 2: With GitHub Desktop

Upload your code to your GitHub repo using the interface in GitHub Desktop.

1. Save your file within the folder in your repository in GitHub Desktop.
2. When you see the changes in GitHub Desktop, add a description of the changes you are making in the bottom left panel.
3. Press the button "Commit to main" to commit those changes.
4. Press the button "Push origin" to push the changes to the online repository. After this step, the changes should be visible on a browser, after refreshing the page.

Method 3: At the Command Line

Push your code directly to the repository from the command line in a terminal window, such as GitBash on a Windows machine or Terminal on a Mac.

1. Open GitBash or Terminal and navigate to the folder inside your local copy of your git repo containing your assignments. Any easy way to do this is to right-click and open GitBash within the folder in Explorer. A better way is to navigate with UNIX commands, such as `cd`.
2. Enter `git add .` to stage all of your files to commit to your repo. You can enter `git add my_filename.ext` to add files one at a time, such as `my_functions.py` in this Assignment.
3. Enter `git commit -m "Describe your changes here"`, with an appropriate description, to commit the changes. This packages all the `add`ed changes into a single unit and stages them to `push` to your online repo.
4. Enter `git push origin main` to push the changes to the online repository. After this step, the changes should be visible on a browser, after refreshing the page.