Studio 4 Covariance and correlation
18.05, Spring 2022

This image is in the public domain.

Overview of the studio

This studio explores covariance and correlation via simulation.

R introduced in this studio

The R needed is introduced in `mit18_05_s22_studio4-samplecode.r`. We will use

functions `var()`, `cov()`, `cor()`, `sample()`, `hist()`

Download the zip file

- You should have downloaded the file `mit18_05_s22_studio4.zip` from our MITx site.
- Unzip it in your 18.05 studio folder.
- You should see the following R files
  `mit18_05_s22_studio4.r`
  `mit18_05_s22_studio4-samplecode.r`
  `mit18_05_s22_studio4-test.r`
  and the following other files
  `mit18_05_s22_studio4-test-answers.html`

Prepping R Studio

- In R studio, open `mit18_05_s22_studio4-samplecode.r` and `mit18_05_s22_studio4.r`

- Using the Session menu, set the working directory to source file location. (This is a good habit to develop!)
- Answer the questions in the detailed instructions just below. Your answers should be put in `mit18_05_s22_studio4.r`
- Solution code will be posted tomorrow at 10 pm

Detailed instructions for the studio

• Go through **mit18__05__s22__studio4-samplecode.r** as a tutorial.

Pay special attention to the sections on `var(), cov(), cor()` and histograms. The section on sampling from an arbitrary discrete distribution will also help.

Problem 1. This problem examines covariance and correlation. We will simulate a gambling scenario.

Setup

- Axel and Barto are dealers at a casino in Las Vegas.
- Every day during their break, Axel and Barto play roulette `n_together` times together, each betting one dollar on red each time.
- Barto then plays `n_Barto_alone` more times alone, betting one dollar each time.

Note: In Las Vegas the roulette wheel has 18 red, 18 black and 2 green slots. So there is an 18/38 chance of winning a bet on red. If you bet one dollar, you either win or lose one dollar.

Problem 1a. Here you will finish the code for the function

```
studio4_problem_1a(n_together, n_Barto_alone, ntrials)
```

This function should do a simulation to estimate the means and variances of both Axel and Barto's daily winnings and the covariance and correlation between Axel and Barto's daily winnings.

The arguments to this function are:
`n_together` = number of games Barto and Axel play together.
`n_Barto_alone` = number of games Barto plays alone after `n_together` games he plays with Axel
`ntrials` = number of trials to run in one simulation

- One trial consists of a simulation of one day of gambling.
- Run `ntrials` trials.
- Use the cat statements provided to print the sample means, variances, covariance and correlation.
- In the cat statements, you will need to provide the variables using the names you chose for the various quantities.

Problem 1b. Here you will finish the code for the function

```
studio4_problem_1b()
```

Run your code from problem 1a using `n_together = 10` and various values of `n_Barto_alone`. Then use the cat statements to say what happens to the sample covariance and correlation as the number of games Barto plays alone increases.

Your solution should just print out the descriptions asked for. Do not include the calls to `studio4_problem_1a()` in you submitted code.

Problem 2. Simulated central limit theorem.
This is cool and shouldn't take a lot of code.

Here you will finish the code for the function

```
studio4_problem_2(n_bets_per_trial, ntrials)
```

Arguments:
`n_bets_per_trial` = the number of bets in each trial
`n_trials` = number of trials

For this problem, one trial should simulate a player's winnings over `n_bets_per_trial` bets. As with Axel and Barto, the player bets on red each time. Run `ntrials` trials and plot a density histogram of the results. It is okay to let R pick the bins for your histogram.

One trial represents one draw from a random variable, call it X. Because X is the sum of many i.i.d. random variables, the central limit theorem says it should be approximately normal.

Use linearity of expectation and variance to compute the theoretical expected value and standard deviation of X. Use these in `dnorm()` and add a plot of the approximating normal distribution to your histogram.

Testing your code

For each problem, we ran the problem function with certain parameters. You can see the function call and the output in `mit18_05_s22_studio4-test-answers.html`. If you call the same function with the same parameters, you should get the same results as in `mit18_05_s22_studio4-test-answers.html` – if there is randomness involved the answers should be close but not identical.

For your convenience, the file `mit18_05_s22_studio4-test.r` contains all the function calls used to make `mit18_05_s22_studio4-test-answers.html`.

Before uploading your code

1. Make sure all your code is in `mit18_05_s22_studio4.r`. Also make sure it is all inside the functions for the problems.
2. Clean the environment and plots window.
3. Source the file.

4. Call each of the problem functions with the same parameters as the test file `mit18_05_s22_studio4-test-answers.html`.
5. Make sure it runs without error and outputs just the answers asked for in the questions.
6. Compare the output to the answers given in `mit18_05_s22_studio4-test-answers.html`.

Upload your code

Use the upload link on our MITx site to upload your code for grading.

Leave the file name as `mit18_05_s22_studio4.r`. (The upload script will automatically add your name and a timestamp to the file.)

You can upload more than once. We will grade the last file you upload.

Due date

Due date: The goal is to upload your work by the end of class.

If you need extra time, you can upload your work any time before 10 PM ET the day after the studio.

Solutions uploaded: Solution code will be posted on MITx at 10 PM the day after the studio.