# Final Project

**Foundations of Data Science**

Jeremy Teitelbaum

## Instructions

You may do this final homework set in either R or Python (your choice). You may use a different language in each of the major parts, but don't change languages in the middle of the parts.

### Preliminaries

Create a repository on `github.com` that will hold your solutions to this final exercise. If you would prefer that this repository not be public, make it private and invite me as a collaborator.

You will use HuskyCT to submit a *link* to this github repo, but your answers should be located in the repository.

### Part One

In this part of the final project, you will generate a data set that has certain statistical properties. "Synthetic" or "Fake" data can be useful for developing and testing algorithms and models.

Your solution to this part will consist of:

- a jupyter notebook or qmd file called `fake_data_builder.ipynb` or `fake_data_builder.qmd` that:

a. carries out the construction of the desired dataset
b. answers the questions and provides the plots requested below

- a copy of the `fake_data.csv` file that you construct.

I should be able to work through your ipynb or qmd file and build the desired dataset.

This jupyter notebook and associated `fake_data.csv` file should be included in your Git repository.

Your dataset has the following properties:

- There are 800 samples.
- Each sample from the dataset has five features called `ID`, `Group`, `Feature_1`, `Feature_2` and `Feature_3`.
- The `ID` feature is a unique identifier for each sample. The identifier has the form `IDxxx` where `x` is a three digit number.
- The `Group` feature is either `A`, `B`, or `C`. Roughly a third of each sample lies in each group.
- `Feature_1` is sampled randomly from a normal distribution with mean 2500. About 66% of the data falls between 2200 and 2800.
- `Feature_2` is sampled randomly from a normal distribution with mean 400 and standard deviation 15.
- `Feature_3` depends on the `Group` feature. For `A`, `Feature_3` is approximately `3(Feature_1)-5(Feature_2)`. For `B`, `Feature_3` is approximately `2(Feature_1)`. For `C`, `Feature_3` is approximately `-(Feature_1)+(Feature_2)`. In each of cases `A`, `B`, and `C` the difference between `Feature_3` and the value given by the formula is a normally distributed random variable with mean 0 and standard deviation 12.

1. Create a dataframe or tibble representing this dataset that has the desired properties.
2. Answer the following questions:

a. Based on the theoretical distribution of `Feature_1`, how many samples do you expect to find where the value of the feature is greater than 3000?

b. How many samples did you actually find where `Feature_1` is greater than 3000?
c. Group the dataset by the `Group` field and compute the mean and standard deviation of `Feature_3` within each group.

3. Make a scatterplot of `Feature_3` vs `Feature_1`.
4. Write your constructed dataframe/tibble to a csv file called `fake_data.csv`.

Hint: You build a pandas dataframe from lists or vectors from a dictionary: `df = pd.DataFrame({'column_name_one': data 1, 'column_name_two: data 2...})`. You build a tibble from named lists `df = tibble(column_name_one = data 1, ...)`

## Part Two

Your answer to this part should be a jupyter notebook or qmd file called `file_fun.ipynb` or `file_fun.qmd`. This notebook should include the function asked for in part one together with the code to create the files described in part 2. You will include this file in your submitted github repository.

The file codes.txt consists of 50 lines. Each line is a code of the following form:

xxxxx_nnnnn_extension

where xxxxx is a 5 digit number, nnnnn is a five digit alphabetic code, and extension is one of 'py','txt', or 'csv'.

1. Write a function that takes a string of this form, extracts the alphabetic code nnnnn and the extension and returns a string of the form nnnnn.extension. So, for example, given '35538_YLTRR_csv' it returns 'YLTRR.csv'.
2. Using your function, write code that reads in the entries from `codes.txt` and for each code writes the 5 digit number xxxxx into a file named nnnnn.extension constructed by your function in part 1. All of these files should be contained in a subdirectory of your main project directory called 'expanded'. So for example, you would create a file called expanded/YLTRR.csv containing (just) 35538.

## Part Three

### The topChef database

TopChef is a reality TV show in which contestants, who are professional chefs, compete in a variety of cooking contests over the course of a season. Each episode, one or more chefs are kicked off the show, until, at the end the winner is crowned "Top Chef".

The topChef database contains information about (most of) the top Chef seasons.

You can access the top chef database using the same credentials we used in class for the sakila database. The only difference is that the database name (`dbname`) is `topChef` instead of `sakila`.

If you would prefer to work in R (and even avoid SQL), you can load the topChef data by `install.packages("topChef")` and `library("topChef")`.

There are 6 tables in the `topChef` database but we will focus our attention on two of them: `chefdetails` and `judges`. The `chefdetails` tells us information about the contestants, and the `judges` table tells us information about the judges.

Of particular interest in the `chefdetails` table are the fields:

- `season` which is the name of the season (often a city where the competition took place, but not always)
- `seasonNumber` which is the number of the season
- `name` which is the chef's name

- `placement` which tells where the chef finished in the competition
- `gender` which tells the chef's gender.

In the `judges` table we are interested in:

- `season` and `seasonNumber` which are the same as in the `chefdetails` table.
- `episode` which gives the episode where the judge appeared; the show uses many guest judges varying episode by episode.
- `challengeType` describes the type of challenge where the judge played a role.
- `guestJudge` is the judge's name
- `competedOnTC` is 'Yes' if the judge was a previous top chef contestant.

## Top Chef problems

1. There are two types of top Chef programs; the 'Masters' programs and the regular series. The database contains information on both. We are *not* interested in the 'Masters'. There is also a season called "Canada 6" which is part of the Masters series. Create versions of the `chefdetails` and `judges` tables that exclude any records coming from seasons that include the word `Masters` or `Canada`.
2. Further trim your `chefdetails` and `judges` tables by including only the columns of interest from the table descriptions above.
3. Your tables should have information from 20 remaining different seasons, numbered from 1 to 20. Make a table that has two columns, one called `season` and one called `seasonNumber` showing the number associated with each season. (For example, `San Francisco` is season `1`.)
4. Answer the following questions using your tables so far.

a. Among all chef contestants, how many are male and how many are female?
b. Among all winners (`placement`==1) how many are male and how many are female?
c. Among all sets of top 3 finalists (`placement`=1,2,3) how many are male and how many are female?
d. Compare the number of times a female was placed in the top 3 with the number of times a female won. Does this number seem unusual? That is, are women who place in the top 3 less likely than expected to ultimately win the competition? Why or why not?

4. Make a table with one row for each first place winner containing the winner's name, the season they won, and a column containing 'Yes' or 'No' depending on whether they served as a guest judge or not.
5. Find all contestants who were *not* first place winners but *did* serve as a guest judge at some point. Make sure to only list each name once.