STAT/CS 94 Fall 2015 Adhikari HW11, Due: 11/18/15

Problem 1 Comprehension Comprehension

In this problem, we'll introduce you to a new piece of Python syntax, *list comprehensions*. Here is a piece of code you might find familiar by now:

```
results = []
for element in some_list:
    value = do_computation(element)
    results.append(value)
```

That's a lot of code to express something fairly simple: Make a list of the things you get when you do some computation (for example, calling `do_computation`) on each element of `some_list`. In fact, you have seen a simpler way to do that when your list is a column in a table: `table.apply(do_computation, 'column_name')`. But what if you don't want to use tables? And what if you don't want to define a function that does everything you want to do with each element? That's where list comprehensions are useful. We can rewrite our code above as: `results = [do_computation(element) for element in some_list]`. The syntax for a basic list comprehension is:

```
[<expression possibly involving varname> for <varname> in <some_list>]
```

The value of the expression is the list you get when you evaluate the expression `len(<some_list>)` times, each time defining `<varname>` to be the next element of `<some_list>`. `<some_list>` can also be a Numpy array or another sequence-like thing. For example, here is a list comprehension whose value is a list (note: not an array) of the first 5 perfect squares (including 0): `[x*x for x in np.arange(5)]`.

Here are some basic exercises to get familiar with list comprehensions.

(a) Write a list comprehension whose value is a list of the first 10 even numbers (including 0).

(b) Write a list comprehension whose value is a list of the square roots of the first 7 whole numbers (including 0).

(c) Suppose someone has put the English names of the 12 months of the year in a list called `month_names`. Write a list comprehension whose value is a list of the lengths of the names of the months.

(d) Suppose you have performed a 3-nearest-neighbors regression on a dataset with 2 columns and defined a function called `regression_prediction` that takes a single number as its argument and returns your regression prediction for that number. Write a list comprehension whose value is a list of your regression predictions for the numbers 0.5, -1.0, and 5.0, in that order.

Problem 2 Data-Driven Design

For this problem **and the remainder of the problems in this assignment**, here are four useful functions, which you may assume have already been defined for you:

1. `bootstrap_mean(table, column_label)`:
   - Bootstraps `table[column_label]` using 10,000 repetitions
   - Returns an approximate 95% confidence interval for population mean of `column_label`
2. `bootstrap_slope(table, column_x_label, column_y_label)`:

- Bootstraps rows of `table` using 10,000 repetitions
- For each bootstrap sample, performs a regression of `column_label_y` on `column_label_x`
- Returns an approximate 95% bootstrap confidence interval for the true slope

3. `perm_test_means(table, column_label, group_label)`:
   - Randomly permutes `table[column_label]` 10,000 times to test equality of distributions by comparing sample means
   - `group_label` denotes Group A by 0 and Group B by 1
   - Returns the approximate P-value of the test
4. `bootstrap_AB_test_means(table, column_label, group_label)`:
   - Bootstraps `table[column_label]` to test equality of distributions by comparing sample means, using 10,000 repetitions
   - `group_label` denotes Group A by 0 and Group B by 1
   - Returns the approximate P-value of the test

In this problem and each one below, **give a precise statement of the null hypothesis or explain why no null hypothesis is needed**. Say which (if any) of the above functions should be used, and, as appropriate, write the code you would run to call the function. (You may include code to augment tables if needed.) Please explain how you would use the output of the function to answer the question. You may assume that all samples are large, as are the numbers of repetitions of the sampling procedures. If you use regression, you may assume the regression model is justified.

Please note that the questions are stated rather loosely; this is how questions arise in practice. You, as the data scientist, have to decide how to make the question precise.

Now, for the question: Researchers would like to study the effect of a website's design on the amount of money people spend at that site. Each of 500 visitors to the site is randomly assigned to view one of two versions of the site. The table `site` contains the data. There is one row for each of the 500 visitors. The column `'version'` contains the code 1 or 2 depending on which version the visitor viewed. The column `'money'` contains the amount of money, in dollars, spent by the visitor on the site. Explain how you would test whether the version makes a difference to the amount of money spent.

Problem 3 Population Puzzler

The table `usa` contains one row for each of the years 1970 through 2010. In each row, the column `'century'` contains a 0 if the year is in the 20th century and 1 if it is in the 21st. The column `'population'` contains the US population for that year. How would you go about deciding whether the US population is on average higher in the 21st century than in the 20th?

Problem 4 Education Elucidation

The table `couples` consists of one row for each of 600 couples chosen randomly from all the married couples in a city. The column `'educ_h'` contains the husband's educational level in years, and `'educ_w'` contains the wife's educational level in years. Explain how you would decide whether or not the husbands in the city are about as educated as their wives.

Problem 5 Prolificacy Problem

(continuing the previous problem) The column `'children'` of the table `'couples'` contains the number of children in the couple's household. Explain how you would decide whether the wife's educational level makes a difference to the number of children, among families in the city.

STAT/CS 94 Fall 2015 Adhikari HW11, Due: 11/18/15

NAME: SID:

Problem 1 Comprehension Comprehension

(a)

(b)

(c)

(d)

Problem 2 Data-Driven Design

Problem 3 Population Puzzler

Problem 4 Education Elucidation

Problem 5 Prolificacy Problem