Jiang2013_solution

January 16, 2016

1 Solution of Jiang et al. 2013

1.0.1 Write a function that takes as input the desired `Taxon`, and returns the mean value of `r`.

First, we're going to import the csv module, and read the data. We store the taxon name in the list `Taxa`, and the corresponding `r` value in the list `r_values`. Note that we need to convert the values to `float` (we need numbers, and they are read as strings).

```
In [1]: import csv
```

```
In [2]: with open('../data/Jiang2013_data.csv') as csvfile:
            reader = csv.DictReader(csvfile, delimiter = '\t')
            taxa = []
            r_values = []
            for row in reader:
                taxa.append(row['Taxon'])
                r_values.append(float(row['r']))
```

Make sure that everything went well:

```
In [3]: taxa[:5]
```

```
Out[3]: ['Fish', 'Fish', 'Fish', 'Amphibian', 'Amphibian']
```

```
In [4]: r_values[:5]
```

```
Out[4]: [-0.11, 0.38, 0.51, 0.868, 0.297]
```

Now we write a function that, given a list of taxa names and corresponding r values, calculates the mean r for a given category of taxa:

```
In [5]: def get_mean_r(names, values, target_taxon = 'Fish'):
            n = len(names)
            mean_r = 0.0
            sample_size = 0
            for i in range(n):
                if names[i] == target_taxon:
                    mean_r = mean_r + values[i]
                    sample_size = sample_size + 1
            return mean_r / sample_size
```

Testing using `Fish`:

```
In [6]: get_mean_r(taxa, r_values, target_taxon = 'Fish')
```

Out[6]: 0.39719005173783783

Let's try to run this on all taxa. We can write a little function that returns the set of unique taxa in the database:

```
In [7]: def get_taxa_list(names):
            return(set(names))
```

```
In [8]: get_taxa_list(taxa)
```

```
Out[8]: {'Amphibian',
         'Annelids',
         'Bird',
         'Chelicerate',
         'Crustacean',
         'Fish',
         'Gastropod',
         'Insect',
         'Mammal',
         'Protist',
         'Reptile'}
```

Calculate the mean r for each taxon:

```
In [9]: for t in get_taxa_list(taxa):
            print(t, get_mean_r(taxa, r_values, target_taxon = t))
```

```
Protist 0.61402
Amphibian 0.18552824175524468
Gastropod 0.40099999999999997
Mammal 0.009
Chelicerate 0.49113529650000004
Reptile 0.11750000000000002
Bird 0.13175671104423078
Crustacean 0.40302827731946345
Fish 0.39719005173783783
Insect 0.19664531553867934
Annelids 0.2
```

1.0.2 You should see that fish have a positive value of `r`, but that this is also true for other taxa. Is the mean value of `r` especially high for fish? To test this, compute a p-value by repeatedly sampling 37 values of `r` (37 experiments on fish are reported in the database) at random, and calculating the probability of observing a higher mean value of `r`. To get an accurate estimate of the p-value, use 50,000 randomizations.

Are these values of assortative mating high, compared to what expected at random? We can try associating a p-value to each `r` value by repeatedly computing the mean `r` value for the taxa, once we scrambled the taxa names! (There are many other ways of doing the same thing, for example counting how many times a certain taxon is represented, and sampling the values at random).

```
In [10]: import scipy

         def get_p_value_for_mean_r(names,
                                    values,
                                    target_taxon = 'Fish',
                                    num_simulations = 1000):
```

```
    # first, compute the observed mean
    observed = get_mean_r(names, values, target_taxon)
    # now create a copy of the names, to be randomized
    rnd_names = names[:]
    p_value = 0.0
    for i in range(num_simulations):
        # shuffle the fake names
        scipy.random.shuffle(rnd_names)
        tmp = get_mean_r(rnd_names, values, target_taxon)
        if tmp >= observed:
            p_value = p_value + 1.0
    p_value = p_value / num_simulations
    return [target_taxon, round(observed, 3), round(p_value, 5)]
```

Let's try the function on `Fish`:

```
In [11]: get_p_value_for_mean_r(taxa, r_values, 'Fish', 50000)
```

```
Out[11]: ['Fish', 0.397, 0.00364]
```

A very small p-value: this means that the observed value (0.397) is larger than what we would expect by chance.

1.0.3 Repeat the procedure for all taxa.

```
In [12]: for t in get_taxa_list(taxa):
             print(get_p_value_for_mean_r(taxa, r_values, t, 50000))
```

```
['Protist', 0.614, 0.0033]
['Amphibian', 0.186, 1.0]
['Gastropod', 0.401, 0.07886]
['Mammal', 0.009, 0.84372]
['Chelicerate', 0.491, 0.01108]
['Reptile', 0.118, 0.93094]
['Bird', 0.132, 0.99988]
['Crustacean', 0.403, 0.0]
['Fish', 0.397, 0.00356]
['Insect', 0.197, 0.99844]
['Annelids', 0.2, 0.5948]
```

Meaning that Fish, Protists and Crustaceans have high values, while Amphibians and Birds lower values than expected by chance.