Jiang2013_solution

January 23, 2017

1 Solution of Jiang *et al.* 2013

1.1 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 [4]: import csv

In [5]: with open('../data/Jiang2013_data.csv') as csvfile:
            # set up csv reader and specify correct delimiter
            reader = csv.DictReader(csvfile, delimiter = '\t')
            taxa = []
            r_values = []
            for row in reader:
                taxa.append(row['Taxon'])
                r_values.append(float(row['r']))
```

We check the first five entries to make sure that everything went well:

```
In [6]: taxa[:5]

Out[6]: ['Fish', 'Fish', 'Fish', 'Amphibian', 'Amphibian']

In [7]: r_values[:5]

Out[7]: [-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 [8]: 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
```

Test the function using `Fish` as target taxon:

```
In [9]: get_mean_r(taxa, r_values, target_taxon = 'Fish')
```

```
Out[9]: 0.39719005173783783
```

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

```
In [10]: def get_taxa_list(names):
             return(set(names))
```

```
In [11]: get_taxa_list(taxa)
```

```
Out[11]: {'Amphibian',
          'Annelids',
          'Bird',
          'Chelicerate',
          'Crustacean',
          'Fish',
          'Gastropod',
          'Insect',
          'Mammal',
          'Protist',
          'Reptile'}
```

Calculate the mean r for each taxon:

```
In [12]: for t in get_taxa_list(taxa):
             print(t, get_mean_r(taxa, r_values, target_taxon = t))
```

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

1.1.1 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` at random (37 experiments on fish are reported in the database), 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 is expected by chance? We can try associating a *p-value* to each `r` value by repeatedly computing the mean `r` of randomized taxa and observing how often we obtain a mean `r` larger than the observed value. There are many other ways of obtaining such an emperical *p-value*, for example counting how many times a certain taxon is represented, and sampling the values at random.

In [ ]:

In [30]:
```python
import scipy # scipy for random shuffle

def get_p_value_for_mean_r(names,
                           values,
                           target_taxon = 'Fish',
                           num_simulations = 1000):
    # compute the (observed) mean_r
    obs_mean_r = get_mean_r(names, values, target_taxon)
    # create a copy of the names, to be randomized
    rnd_names = names[:]
    # create counter for observations that are higher than obs_mean_r
    count_mean_r = 0.0
    for i in range(num_simulations):
        # shuffle the taxa names
        scipy.random.shuffle(rnd_names)
        # calculate mean r value of randomized data
        rnd_mean_r = get_mean_r(rnd_names, values, target_taxon)
        # count number of rdn_mean_r that are larger or equal to obs_mean_r
        if rnd_mean_r >= obs_mean_r:
            count_mean_r = count_mean_r + 1.0
    # calculate p_value: chance of receiving rnd_r_mean larger than r_mean
    p_value = count_mean_r / num_simulations
    return [target_taxon, round(obs_mean_r, 3), round(p_value, 5)]
```

Let's try the function on `Fish`:

In [24]:
```python
get_p_value_for_mean_r(taxa, r_values, 'Fish', 50000)
```

Out[24]:
```
['Fish', 0.397, 0.0033]
```

A very small *p-value*: this means that the observed mean `r` value (0.397) is larger than what we would expect by chance. Note that your calculated *p-value* might deviate slightly from ours given the randomness in a simulation.

1.1.2 Repeat the procedure for all taxa.

```
In [31]: for t in get_taxa_list(taxa):
             print(get_p_value_for_mean_r(taxa, r_values, t, 50000))
```

```
['Insect', 0.197, 0.9986]
['Gastropod', 0.401, 0.0796]
['Fish', 0.397, 0.0033]
['Chelicerate', 0.491, 0.01136]
['Protist', 0.614, 0.00348]
['Bird', 0.132, 0.99984]
['Amphibian', 0.186, 1.0]
['Reptile', 0.118, 0.93122]
['Annelids', 0.2, 0.5906]
['Crustacean', 0.403, 0.0]
['Mammal', 0.009, 0.84158]
```

Fish, Protists and Crustaceans have higher mean `r` values than expected by chance (*p-value* $\leq$ 0.01). Insects, Amphibians and Birds have lower values than expected by chance (*p-value* $\geq$ 0.99).

```
In [ ]:
```