Practical Computing for Bioinformatics
Assingnment II

Alexandra Pančíková
Ivana Janíčková

December 7, 2020

Exercise 1.

To find species to which the genes refer to we go the NCBI Gene database. The result of querying the database with one of the genes is the page belonging to the geneID (e.g: for gene ID: ENSG00000182197 it is `https://www.ncbi.nlm.nih.gov/gene/?term=ENSG00000182197`).

- The corresponding organism is *Homo Sapiens*
- The type of ID is an Ensemble ID

Exercise 2.

In order to calculate the A, B, C, D values of contingency table for a given GO category, the script in the file **table.py** executes the following steps:

Function `parseFile`

- This function takes a string of two file names as an input - for it to run it is expected that the files containing GO annotations and IDs are in the working directory.
- The extracted data is stored in HashMap and Set data structures in order to enhance performance and facilitate operations on the data.
- The set of all genes comes from the GO annotations file and correspond to the genes for which the GO terms were given.

Function `makeTableDict`

- For each GO term, this function calculates the A, B, C, D values and stores them in an array. (the GOterm *key* : [A , B, C , D] *value* of a returned dictionary.)
- A,B,C,D values are stored in an array to enable accessing the values by their indexes.

Function `makeTable`

- For a formatted contingency table for a given GO term a *makeTable* function can be used.
- For a given GO term in returns a data frame of A,B,C,D values along with the marginal values.

Exercise 3.

Calculating a Chi-squared statistic and saving the results in a form of a data frame is done in **chistat.py** file. The file consists of two functions:

Function `calculateChiStat`

- As an input this function takes a GO term and the HashMap of contingency tables (output of *makeTableDict*).
- For a given GO term it calculates a Chi-squared statistic - based on the corresponding A,B,C,D values.

Function `makeChiDf`

- This function returns a dataframe consisting of GO name, GO term and the calculated Chi-squared statistic.
- For each GO term it calls *calculateChiStat* function.
- For outputting a GO name in the returned data frame we use the names_dict which stores a name (*value*) for each GO term (*key*).
- The entries of the data frame are sorted in an descending order based on the Chi Statistic value.

Exercise 4.

The output of the results is stored in the chi_df data frame. In order to print the top 5 GO categories with the strongest over-representation we use the command:

```
print(chi_df.head())
```

The first and second GO terms have the same Chi-squared statistic value. Additionally, the fourth and fifth GO terms belong to a large group of GO terms with the same Chi-squared statistic value - hence the actual output is only a sample of all possible GO terms.