|  |  |  |
| --- | --- | --- |
| *Darko* | *Ivanovski* | *1243085* |

## Midterm test No. 3

### 22 / 12 / 2020

## Questions

1. Paste below your assignment ID.

*22*

1. DownloadhumanGO annotations (GAF format) from GOA at EBI ([ftp.ebi.ac.uk/pub/databases/GO/goa/HUMAN/](http://ftp.ebi.ac.uk/pub/databases/GO/goa/HUMAN/)).

*For these answers I used the code provided in “example.py”.*

*For every accession if it was present in my dataset, I counted the number of annotations and the terms as belonging to my set. In the other case the annotations/proteins/terms were part of the rest.*

*To calculate the leaf terms associated with my list of proteins I just extracted all the leaf terms and filtered out the ones that weren’t inside my terms set.*

* 1. How many annotations are associated with your list of proteins?

*8212*

* 1. How many annotations are associated with the rest of human proteins?

*283098*

* 1. How many different (unique) terms are associated with your proteins?

*5348*

* 1. How many different (unique) terms are associated with the rest of human proteins?

*21868*

* 1. How many leaf terms are associated with your proteins?

*1151*

1. Which are the most abundant GO terms considering your list of proteins? Provide no more than 10 terms separately for each namespace (sub-ontology).

*For this answer I used the code provided in the script “parse\_goa,py”: basically the code builds a map where for every term is associated the count of numbers of times is repeated.*

*Then from this map is extracted an array, sorted by the value (decreasing) and foreach root the code loops over the first 10 elements.*

* *molecular\_function*
  + *GO:0005515 197 protein binding*
  + *GO:0005262 52 calcium channel activity*
  + *GO:0042802 44 identical protein binding*
  + *GO:0005509 43 calcium ion binding*
  + *GO:0005245 38 voltage-gated calcium channel activity*
  + *GO:0046872 36 metal ion binding*
  + *GO:0005524 27 ATP binding*
  + *GO:0005516 22 calmodulin binding*
  + *GO:0005385 21 zinc ion transmembrane transporter activity*
  + *GO:0005102 20 signaling receptor binding*
* *biological\_process*
  + *GO:0070588 115 calcium ion transmembrane transport*
  + *GO:0006816 75 calcium ion transport*
  + *GO:0006874 44 cellular calcium ion homeostasis*
  + *GO:0051209 35 release of sequestered calcium ion into cytosol*
  + *GO:0034220 30 ion transmembrane transport*
  + *GO:0007186 28 G protein-coupled receptor signaling pathway*
  + *GO:0070374 26 positive regulation of ERK1 and ERK2 cascade*
  + *GO:0019722 24 calcium-mediated signaling*
  + *GO:0034765 23 regulation of ion transmembrane transport*
  + *GO:0006851 23 mitochondrial calcium ion transmembrane transport*
* *cellular\_component*
  + *GO:0005886 183 plasma membrane*
  + *GO:0005887 104 integral component of plasma membrane*
  + *GO:0016021 91 integral component of membrane*
  + *GO:0016020 83 membrane*
  + *GO:0005737 60 cytoplasm*
  + *GO:0005739 36 mitochondrion*
  + *GO:0005783 36 endoplasmic reticulum*
  + *GO:0005829 35 cytosol*
  + *GO:0009986 33 cell surface*
  + *GO:0070062 31 extracellular exosome*

1. Which are the most abundant GO terms considering your list of proteins after integrating ancestors terms? Provide no more than 10 terms separately for each namespace (sub-ontology).

*This answer follows the same procedure of the 3th with the difference than this time it sums also the number of ancestors inside the terms count map.*

* *molecular\_function*
  + *GO:0003674 281 molecular\_function*
  + *GO:0005488 241 binding*
  + *GO:0005515 216 protein binding*
  + *GO:0022857 191 transmembrane transporter activity*
  + *GO:0005215 191 transporter activity*
  + *GO:0015075 190 ion transmembrane transporter activity*
  + *GO:0015318 186 inorganic molecular entity transmembrane transporter activity*
  + *GO:0008324 184 cation transmembrane transporter activity*
  + *GO:0022890 182 inorganic cation transmembrane transporter activity*
  + *GO:0046873 177 metal ion transmembrane transporter activity*
* *biological\_process*
  + *GO:0051234 281 establishment of localization*
  + *GO:0072511 281 divalent inorganic cation transport*
  + *GO:0006812 281 cation transport*
  + *GO:0006810 281 transport*
  + *GO:0030001 281 metal ion transport*
  + *GO:0008150 281 biological\_process*
  + *GO:0006811 281 ion transport*
  + *GO:0070838 281 divalent metal ion transport*
  + *GO:0051179 281 localization*
  + *GO:0009987 277 cellular process*
* *cellular\_component*
  + *GO:0005575 281 cellular\_component*
  + *GO:0110165 280 cellular anatomical entity*
  + *GO:0016020 248 membrane*
  + *GO:0031224 198 intrinsic component of membrane*
  + *GO:0016021 197 integral component of membrane*
  + *GO:0005886 183 plasma membrane*
  + *GO:0043226 166 organelle*
  + *GO:0043227 155 membrane-bounded organelle*
  + *GO:0043229 127 intracellular organelle*
  + *GO:0005622 127 intracellular anatomical structure*

1. For each term *GOi* build a confusion matrix as defined below and calculate the “fold increase” within your set of proteins compared to the rest of human proteins.

**Hint**: The fold increase can be calculated dividing the ratio *having-/not-having the property* of the *selected* with the ratio *having-/not-having* of the *not selected*.

|  |  |  |
| --- | --- | --- |
|  | **Having the property** | **Not having the property** |
| **Selected** | No. proteins with *GOi* in your set | No. proteins without *GOi* in your set |
| **Not selected** | No. proteins with *GOi* in the rest of human proteins | No. proteins without *GOi* in the rest of human proteins |

*For these answers I used the code at the end of “exercise.py” where it calculates exactly the confusion matrix reported above. Than in order to answer question “a” I just counted the terms with fold-increase greater than 1.*

*For the second I just loop over the first 10 elements ordered decreasingly by fold-increase (for each subontology).*

* 1. How many terms have a fold increase larger than 1.

*4990*

* 1. Report terms with the largest fold increase, 10 terms for each namespace (sub-ontology).
* *cellular\_component*
  + *GO:1990454 fold increase 824.7544483985765*
  + *GO:1990246 fold increase 481.10676156583634*
  + *GO:0030314 fold increase 412.37722419928826*
  + *GO:1903439 fold increase 343.64768683274025*
  + *GO:1903440 fold increase 274.9181494661922*
  + *GO:1903143 fold increase 274.9181494661922*
  + *GO:0014701 fold increase 274.9181494661922*
  + *GO:0098665 fold increase 274.9181494661922*
  + *GO:0098666 fold increase 274.9181494661922*
  + *GO:0005891 fold increase 206.18861209964413*
* *molecular\_function*
  + *GO:0015085 fold increase 3024.0996441281136*
  + *GO:0005262 fold increase 2657.542111506524*
  + *GO:0099604 fold increase 1855.6975088967972*
  + *GO:0072509 fold increase 1798.4228944246738*
  + *GO:0005385 fold increase 1512.049822064057*
  + *GO:0015278 fold increase 1237.131672597865*
  + *GO:0015095 fold increase 1168.4021352313168*
  + *GO:0005245 fold increase 962.2135231316726*
  + *GO:0015368 fold increase 824.7544483985765*
  + *GO:0008331 fold increase 756.0249110320285*
* *biological\_process*
  + *GO:0070838 fold increase 19381.72953736655*
  + *GO:0006816 fold increase 16288.900355871885*
  + *GO:0070588 fold increase 12783.693950177936*
  + *GO:0060401 fold increase 6185.658362989325*
  + *GO:0060402 fold increase 5017.256227758007*
  + *GO:0097553 fold increase 4398.690391459075*
  + *GO:0070509 fold increase 3299.017793594306*
  + *GO:0051283 fold increase 3161.5587188612103*
  + *GO:0051209 fold increase 3092.8291814946624*
  + *GO:0072511 fold increase 2768.8185053380785*

1. For each confusion matrix generated above, calculate the enrichment with Fisher’s exact test.

**Hint:** You can use <https://pypi.org/project/fisher/>

*For these answers I just install the library and passed the same data as the confusion matrix defined in the previous question. The value we are interested in is the right-tail p-value that identifies the terms with extraordinary presence of the property over the entire collection of terms. For question “b” was just about printing out also the p-values for each of the best terms (ordered by fold-increase) of the previous question.*

* 1. Which P-value between the left- and right-tail tells which are the enriched terms in your set of proteins?

**Hint**: Make a comparison with terms with high “fold increase” calculated above if you are not sure.

*The enriched terms are the ones with low right tail values (usually is <0.05)*

* 1. Report enriched terms in the selected set and the corresponding P-values (left-/right-/two-tails).
* *cellular\_component*
  + *GO:1990454 p-values: left 0.9999999999701241 - right: 1.2704656987876292e-21 - two tail: 8.402903246046793e-07*
  + *GO:1990246 p-values: left 0.9999999999770811 - right: 1.0859450159686965e-12 - two tail: 3.5722506318852844e-08*
  + *GO:0030314 p-values: left 0.9999999999762951 - right: 6.4689887907396e-11 - two tail: 1.332814189406564e-08*
  + *GO:1903439 p-values: left 0.9999999999562331 - right: 3.78822743457413e-09 - two tail: 6.508188963479884e-07*
  + *GO:1903440 p-values: left 0.999999999374903 - right: 2.1644865708142907e-07 - two tail: 2.1644865708142907e-07*
  + *GO:1903143 p-values: left 0.999999999374903 - right: 2.1644865708142907e-07 - two tail: 2.1644865708142907e-07*
  + *GO:0014701 p-values: left 1.0 - right: 8.885266755656699e-14 - two tail: 1.595573857504464e-07*
  + *GO:0098665 p-values: left 0.999999999374903 - right: 2.1644865708142907e-07 - two tail: 2.1644865708142907e-07*
  + *GO:0098666 p-values: left 0.999999999374903 - right: 2.1644865708142907e-07 - two tail: 2.1644865708142907e-07*
  + *GO:0005891 p-values: left 0.9999999999637103 - right: 6.296622909446578e-51 - two tail: 4.1990239828477916e-07*
* *molecular\_function*
  + *GO:0015085 p-values: left 1.0 - right: 7.567404111380646e-227 - two tail: 9.529460008821687e-07*
  + *GO:0005262 p-values: left 0.9999999999773461 - right: 5.069034829037317e-200 - two tail: 8.360146144406171e-07*
  + *GO:0099604 p-values: left 1.0 - right: 1.682360834086012e-48 - two tail: 1.9569958130173463e-07*
  + *GO:0072509 p-values: left 1.0 - right: 9.35343501559914e-264 - two tail: 7.308251034924861e-07*
  + *GO:0005385 p-values: left 0.999999999982562 - right: 1.49778353131573e-39 - two tail: 3.881448544568358e-08*
  + *GO:0015278 p-values: left 1.0 - right: 2.2171384721933432e-32 - two tail: 2.741394466475552e-07*
  + *GO:0015095 p-values: left 0.9999999999690412 - right: 1.3778534339611633e-30 - two tail: 1.8633760452950203e-07*
  + *GO:0005245 p-values: left 1.0 - right: 1.0409910634697264e-72 - two tail: 6.232197502258238e-07*
  + *GO:0015368 p-values: left 0.9999999999701241 - right: 1.2704656987876292e-21 - two tail: 8.402903246046793e-07*
  + *GO:0008331 p-values: left 1.0 - right: 7.848466750524858e-20 - two tail: 5.14847550766814e-07*
* *biological\_process*
  + *GO:0070838 p-values: left 0.9999999999785004 - right: 0.0 - two tail: 5.580546179300185e-07*
  + *GO:0006816 p-values: left 1.0 - right: 0.0 - two tail: 1.2768826259491692e-06*
  + *GO:0070588 p-values: left 0.9999999999903403 - right: 2.28214e-319 - two tail: 5.015071425308015e-07*
  + *GO:0060401 p-values: left 0.9999999999553137 - right: 4.945237523208421e-159 - two tail: 1.1300675110319672e-06*
  + *GO:0060402 p-values: left 0.9999999999744371 - right: 1.21326062021888e-129 - two tail: 8.122884573568352e-07*
  + *GO:0097553 p-values: left 1.0 - right: 5.9454985658356496e-114 - two tail: 1.849621802022718e-07*
  + *GO:0070509 p-values: left 0.9999999999946474 - right: 7.836130756050137e-86 - two tail: 1.0599719054071254e-07*
  + *GO:0051283 p-values: left 0.9999999999809881 - right: 2.687402515444658e-82 - two tail: 9.683826784499964e-07*
  + *GO:0051209 p-values: left 0.9999999999883292 - right: 1.5798340056378495e-80 - two tail: 7.969984897460825e-07*
  + *GO:0072511 p-values: left 0.9999999999907728 - right: 0.0 - two tail: 8.194694492491608e-07*