Documentation of the code for the project
“Clustering Alzheimer’s Disease Diagnoses using Genomic Information”

Anjanet Loon
BTEC 699-51- B-2020/Fall
Harrisburg University of Science and Technology
*Note: This is a midterm submission. Paper is still in progress.

Command lines

Command lines define the main steps of the pipelines and are to be entered in a prompt/terminal window. To be launched, each command line requires input files to be present in the working directory. Command lines may produce output files, which are stored in the working directory after the completion of the run of the command line.

>python Final_code.py recdis	
FUNCTIONS USED	• normalization_Text • nonrepeating_Append • record_Diseases
INPUT	• part-ii-dependency-paths-gene-disease-sorted-with-themes.txt.gz
OUTPUT	• PA_diseases.txt

>python Final_code.py recword	

FUNCTIONS USED	• get_Exclusions • counting_Append
INPUT	• Alzheimer_dict_nz_for_icd10.corrected.modular.txt • excluded_words.txt
OUTPUT	• wordlist.txt

>python Final_code.py disdict	
FUNCTIONS USED	• list_Diseases • get_Exclusions • match_String • disease_Dictionary
INPUT	• Alzheimer_dict_nz_for_icd10.corrected.modular.txt • excluded_words.txt • PA_diseases.txt
OUTPUT	• DICT_icd10_PA_diseases.txt

>python Final_code.py gendict	
FUNCTIONS USED	• get_Genes • gene_Dictionary • nonrepeating_Append
INPUT	• DICT_icd10_PA_diseases.txt • part-ii-dependency-paths-gene-disease-sorted-with-themes.txt.gz
OUTPUT	• DICT_icd10_PA_genes.txt

>python Final_code.py gendict2 filename	
FUNCTIONS USED	• get_Genes • gene_Dictionary • nonrepeating_Append

INPUT	• DICT_icd10_PA_diseases.txt • part-ii-dependency-paths-gene-disease-sorted-with-themes.txt.gz • filename.txt (containing a list of ICD10 codes)
OUTPUT	• DICT_icd10_PA_genes_filename.txt

>python Final_code.py orgloc	
FUNCTIONS USED	• organize_GeneLocations
INPUT	• BIOMART_gene_name_location.txt
OUTPUT	• DICT_gene_locations.txt

>python Final_code.py locdict	
FUNCTIONS USED	• get_Location • location_Dictionary
INPUT	• DICT_idc10_PA_genes.txt • DICT_gene_locations.txt
OUTPUT	• DICT_idc10_ENSBL_locations.txt

>python Final_code.py intersect	
FUNCTIONS USED	• get_Locations • nearby_Locations • intersect_Locations
INPUT	• Alzheimer_dict_nz_for_icd10.corrected.modular.txt • DICT_icd10_GRCH38_locations.txt

OUTPUT	• NETW_icd10_GRCH38_locations.txt

>python Final_code.py showtable	
FUNCTIONS USED	• get_Locations • threshold_Network • present_Network
INPUT	• NETW_icd10_GRCH38_locations.txt • DICT_icd10_GRCH38_locations.txt • Alzheimer_dict_nz_for_icd10.corrected.modular.txt
OUTPUT	• An image

>python Final_code.py displaygen icd10_code	
FUNCTIONS USED	• dispay_locations (from displaygenelocations.py)
INPUT	• DICT_icd10_GRCH38_locations.txt
OUTPUT	• An image

>python Final_code.py displaynet icd10_code1 icd10_code2	
FUNCTIONS USED	• dispay_locations (from displaygenelocations.py)
INPUT	• DICT_icd10_GRCH38_locations.txt
OUTPUT	• An image

Functions

Below is the list of functions belonging to the code of the pipeline.

normalization_Text(the_string)	
INPUT	• a string the_string
OUTPUT	• a lower-case version of the input string in which dashes, underscores and apostrophes were replaced with spaces

DESCRIPTION: Turn a string into a lower-case string in which dashes underscores and apostrophes where replaced with spaces.

`nonrepeating_Append(the_list,element)`	
INPUT	• a list `the_list` • an item `element` to be added to `the_list`
OUTPUT	• a Boolean value (true or false) indicating whether the item `element` needed to be added to the list or not

DESCRIPTION: adds the item `element` to the input list `the_list` if the item `element` is not already present in the list.

`record_Diseases(filename)`	
INPUT	• a string `filename` referring to a gene-disease network from the Percha and Altman database
OUTPUT	• the list of diseases contained in the network

DESCRIPTION: stores in a list all the disease names contained in the network called [filename]. The list does not contain repetitions and all its elements were normalized using the function `normalization_Text`.

`get_Exclusions(filename)`	
INPUT	• a string `filename` referring to the name of a file containing a single word at each of its lines
OUTPUT	• the list of words contained in the input file

DESCRIPTION: The functions loops over the elements of the list and checking whether there are elements of this list that are substrings of the string.

`counting_Append(the_list,element)`	
INPUT	• a list `the_list` • an item `element` to be added to `the_list`

OUTPUT	• a Boolean value (true or false) indicating whether the item `element` was found in the list before being added and how many times

DESCRIPTION: The function adds the item `element` to the input list `the_list` with a count of 1 if the item [element] is not already present in the list and increments the count of item if by one if the item was found in the list

`record_Words(medinfofile,exclusionfile)`	
INPUT	• a string `medinfofile` referring to the name of a file containing a table of 3 columns whose • 1st column contains ICD10 codes, • 2nd column contains numerical values • 3rd column contains descriptions of ICD10 codes. • a string `exclusionfile` referring to the name of a file whose lines contain a single word
OUTPUT	a list containing lists of the form `[w,c]` where • `w` is a word extracted from one of the ICD10 descriptions contained in the file `medinfofile` and not present in the text of `exclusionfile` • `c` is an integer counting the number of occurences of that word in `medinfofile`

DESCRIPTION: For each word of every ICD10 description in the file `medinfofile`, the function will count the number of occurrence of that word in the ICD10 descriptions of `medinfofile` if the lower-case version of that word is not in the list of words contained in `exclusionfile`.

`list_Diseases(diseasefile)`	
INPUT	• a string `diseasefile` referring to the name of a file whose lines are numbered and contain the name of a disease. In this project, this is the file named `PA_diseases.txt` created with the option `recdis` passed to the command line

OUTPUT	• a list containing all the disease names of the file

DESCRIPTION: The function puts in a list the disease names contained in the input file

`match_String(word,list_of_diseases,exclusions)`	
INPUT	• a string `word` containing a word from an ICD10 description, • a list `list_of_diseases` of strings containing disease names, • a list `exclusions` of strings that contains words to not consider
OUTPUT	• a list of strings belonging to the list `list_of_diseases`

DESCRIPTION: The function returns a list containing disease names from `list_of_diseases` such that these names are longer than 3 letters, are not in the list `exclusions` of excluded words and

- either contain the string `word`,
- or is contained in the string `word`.

`disease_Dictionary(medinfofile,diseasefile)`	
INPUT	• a string `medinfofile` referring to the name of a file containing a table of 3 columns whose • 1st column contains ICD10 codes, • 2nd column contains numerical values • 3rd column contains descriptions of ICD10 codes. • a string `diseasefile` referring to a file whose lines each contain a disease name
OUTPUT	• no output

DESCRIPTION: The function creates a file "DICT_icd10_PA_diseases.txt containing a table of 2 columns whose 1st column contains ICD10 codes and whose 2nd column contains disease names related to the ICD10 code diagnosis (see `match_String` above).

`get_Genes(disgenfile,disease)`	
INPUT	• a string `disgenfile` referring to the name of a gene-disease network from the Percha and Altman database • a string `disease` referring to the name of a disease
OUTPUT	• a list of gene names associated with the input disease in the gene-disease network passed in the first argument.

DESCRIPTION: the function returns the list of genes associated with a specific disease in the Percha and Altman network passed in the first argument.

`gene_Dictionary(dictionary,disgenfile)`	
INPUT	• a string `dictionary` referring to the name of a file containing a table of 2 columns whose • 1st column contains ICD10 codes, • 2nd column contains lists of diseases associated with the ICD10 code. • a string `disgenfile` referring to the name of a gene-disease network from the Percha and Altman database
OUTPUT	• no output

DESCRIPTION: The function creates a file `DICT_icd10_PA_genes.txt` containing a table of 2 columns whose 1st column contains ICD10 codes and whose 2nd column contains gene names related to the disease names associated with code diagnosis in `dictionary`.

`gene_Dictionary2(dictionary,disgenfile,codes,name)`

INPUT	• a string `dictionary` referring to the name of a file containing a table of 2 columns whose • 1^{st} column contains ICD10 codes, • 2^{nd} column contains lists of diseases associated with the ICD10 code. • a string `disgenfile` referring to the name of a gene-disease network from the Percha and Altman database • a list `codes` containing strings referring to ICD10 codes • a string `name` appended to the name of the output file created by the function
OUTPUT	• no output

DESCRIPTION: The function creates a file named `"DICT_icd10_PA_genes_"+name+".txt"` containing a table of 2 columns whose 1^{st} column contains ICD10 codes contained in codes and whose 2^{nd} column contains gene names related to the disease names associated with corresponding ICD10 codes in `dictionary`.

`organize_GeneLocations(filename)`	
INPUT	• a string `filename` referring to the name of a file extracted from BioMart and containing a table whose columns give the following information: • Gene stable ID, • Gene start (bp), • Gene end (bp), • Chromosome/scaffold name, • Gene description, • Gene name

OUTPUT	• a list containing lists of the form `[g,l]` where • `g` is a string referring to a gene name from the input file [filename] • `l` is a concatenation of strings of the form `c + "_" + s + "_" + e` where • `c` is the chromosome number of the gene g • `s` is the gene start location of the gene `g` (in basepair) • `e` is the gene end location of the gene `g` (in basepair)

DESCRIPTION: Reorganize a dataset extracted from BioMart into a new file containing a table of 2 columns whose first column contains gene names and whose second column contains information regarding the location of this gene in the genome. The gene locations are formatted to be compatible with the graphic library `displaygenelocations.py`.

`get_Location(genlocfile,gene)`	
INPUT	• a string `genlocfile` referring to a file containing a table of two columns such that its first column contains gene names and its second column contains a chromosomal location in the format `chromosome_start_end`, • a string `gene` referring to the name of a gene
OUTPUT	• a string containing the chromosomal location of the input gene

DESCRIPTION: the function returns the genomic location of the gene `gene` passed in the second argument by suing the information contained in the file `genlocfile`.

`location_Dictionary(dictionary,genlocfile)`

INPUT	• a string `dictionary` referring to a file containing table of two columns such that the first column contains ICD10 codes and the second column contains a list of associated genes • a string `genlocfile` referring to a file containing a table of two columns such that its first column contains gene names and its second column contains a chromosomal location in the format `chromosome_start_end`
OUTPUT	• no output

DESCRIPTION: The function creates a file `DICT_idc10_ENSBL_locations.txt` containing a table of 2 columns such that its first column contains ICD10 codes and its second column contains chromosomal locations related to the gene names associated with the corresponding ICD10 code in `dictionary`.

`get_Locations(ukbgenfile,icd10_code)`	
INPUT	• a string `ukbgenfile` referring to the name of file containing a table of 2 columns such that its first column contains ICD10 codes and its second column contains chromosomal locations and a string `icd10_code` containing an ICD10 code, • and a string `icd10_code` containing an ICD10 code
OUTPUT	• a list of strings that contain the chromosomal locations associated with the input ICD10 code

DESCRIPTION: The function returns the chromosomal locations associated with an ICD10 code.

`nearby_Locations(location1,location2)`	
INPUT	• two strings representing gene locations in the format `chromosome_start_end`
OUTPUT	• a Boolean value (True or False) indicating whether location1 is near location2 within a 100,000 bp interval

DESCRIPTION: the function indicates whether two locations are within an shared region of at most 100,000 bp.

`intersect_Locations(locations1,locations2)`	
INPUT	• two lists whose elements are strings representing gene locations in the format `chromosome_start_end`
OUTPUT	• a list of locations from the first input `locations1`that are nearby locations from the second input `locations2`

DESCRIPTION: the function gathers in a list the locations of `locations1` that are proximal to the locations of `locations2`.

`threshold_Network(netwfile,icd10_codes,lengths,threshold)`	
INPUT	• a weighted network file `netwfile` between icd10 codes, • a list `icd10_codes` of all ICD10 codes in the network, • a list `lengths` containing the numbers of locations associated with each ICD10 code, • a float number `threshold`
OUTPUT	• a list consisting of lists of the form `[a,b,w]` where • `a` and `b` are two strings referring to ICD10 codes • `w` is a weight of the network between each pair of ICD10 codes whose numbers of locations are not 0. The weights are set to 0 if they are less than the value `threshold`.

DESCRIPTION: The function truncates the weights of the network below the input threshold value.

`present_Network(network)`

INPUT	• a list consisting of lists of the form `[a,b,w]` where • `a` and `b` are two strings referring to ICD10 codes • `w` is a float value.
OUTPUT	• a table containing the weights of the network between each pair of ICD10 codes whose numbers of locations are not 0. • the list of averages weights for every row of the output table

DESCRIPTION: The function returns the heat map of the weighted network and the average scores of each row of the table.