#########################

#This script is designed to impute missing values in Metabolon HD4 datasets using KNN and MICE imputations. It is technically possible to use it with any metabolomic dataset.

The main function takes the following arguments:

#The script works as follows: 1- The variables in group are supplied by the user. These variables will be imputed using MICE or KNN

#2- Variables in group 2 will be imputed with zero values

#3- A Pearson’s correlation matrix is created using all the group 1 variables

#4a- Group 1 variables will be split into completed cases (no missing values), incomplete cases, and invalid cases where the number of missing is too high. These variables are named *ccm*, *icm*/*icm\_names*, and *invalids* respectively.

#4b- The cutoff for this is set at 90% missingness OR if the number of non-missing values in less than the number of predictor variables + 20. This is done to because of two reasons: 1- to eliminate possibly mis-annotated metabolites or unannotated metabolites that are xenobiotic in nature, 2- To ensure the availability of enough cases to perform the imputation. This issue prevents the MICE package from performing the imputation all together. This can be manually altered by changing the *miss90* and *maxpreds* variables.

#4c- The invalid cases will be imputed to zero.

#5- The group 1 variables will be log transformed and scaled and centered on their mean. This information will be stored in separate variable to un-scale the values later.

#6- The imputation function will loop through the icm variables one by one.

#7- In both imputation methods the auxiliary variables and/or predictors will be selected. In MICE these will include the variables specified in the covar argument.

#7- A subset dataframe (*minidf*)will be created that includes the incomplete variable and the predictor variables ONLY.

8- For kNN, the kNN function from the VIM package is used with 10 k neighbours.

9a- For MICE we use the mice package. first we explicitly specify which variable to impute using the make.where function.

#9b- The mice function will use the following arguments: defaultMethod = Predictive mean matching (pmm), m= 5 datasets. The output object is stored in “IMP” variable. Note: you can adjust how many datasets are created by providing the value in number\_m argument in Main.

#9c- We then convert the object to a single dataframe that contains all 5 generated datasets. If this is the first metabolite we run the “complete” function with the include argument set to TRUE. This is done to include the original, pre-imputation data in the “long” output where the imputed data sets are stacked vertically (see the mice package documentation, “complete” section, for details).

#9d- For subsequent variables the include argument will be omitted but it will still be in the “long” format.

#9e- It is crucial to convert to the long format for three reasons: 1- to enable merging the outputs together, which would be impossible otherwise. 2- To easily write the merged output to a sheet file for reference, future reuse, usage in multivariate analysis such PCA etc (see 9f below). 3- To enable reconverting the merged output to a “mids” object (see the mice package documentation “as.mids” section for details). The “mids” object is needed to complete the mice process of applying the analysis model on each of the datasets and pooling the estimates. This is covered in more details in section\_\_\_\_.

#9f- To write the merged output from MICE or the output from kNN to an external csv file, simply provide a file name in the filename arguments in the MAIN function.

#10- The imputation output for each variable (that had missing values) will be unscaled and exponentiated then merged into a single dataframe with the group2 variables. In the case of kNN all other variables will be included as well. In MICE the covar variables will be included if specified. If more variables need to be added to the dataframe this can be done after the imputation as shown in section ###.

Examples of running the scripts:

source('Master\_Script.r')

#mydata: your data table in dataframe class format.

#endoids: a user created vector containing the column names of the endogenous metabolites

#unknowns: a user created vector containing the column names of the unannotated metabolites

#xeono: a user created vector containing the column names of the xenobiotic metabolites

#covars: a user created vector containing the column names of the outcome and covariates to be used in the analysis model. This argument is only used in MICE. For example if you are studying the association between metabolites with bmi and adjusting for age and sex, then you can add these variables in covars (covars = c(‘bmi’,’age’,’sex’) )

Running default MICE:

miceimp <- Main(DataFrame = mydata, group1 = c( endoids , unknowns ) , group2= xeono , covars = covars, use\_knn = FALSE )

Running default KNN:

miceimp <- Main(DataFrame = mydata, group1 = c( endoids , unknowns ) , group2= xeono, use\_knn = TRUE )

Running default KNN with only the endogenous metabolites:

miceimp <- Main(DataFrame = mydata, group1 = endoids , use\_knn = TRUE )

Running MICE with single imputation:

miceimp <- Main(DataFrame = mydata, group1 = c( endoids , unknowns ) , group2= xeono, use\_knn = FALSE, covars = covars, m=1)

Running MICE with 10 imputations:

miceimp <- Main(DataFrame = mydata, group1 = c( endoids , unknowns ) , group2= xeono, use\_knn = FALSE, covars = covars, m=10)

About the “mids” object:

The mice package stores the output from the imputation step into the object class mids by default. This stores information about the imputation process used and the imputation datasets created. The with() and pool() need the object class mids as input to run the analysis on the datasets, calculate the estimate for each dataset then pools the estimates and standard errors using Rubin’s Rules.

To convert the object class mids to a “long” format:

require(‘mice’)

IMP <- miceimp <- Main(DataFrame = mydata, group1 = c( endoids , unknowns ) , group2= xeono , covars = covars, use\_knn = FALSE )

Longformat =<-complete(IMP , action = 'long' , include = TRUE)

To convert the Longformat back to mids class:

Mymids <- as.mids(Longformat)

To run the analysis on the mids class and pool:

Mysummary <- summary(pool( with(data = Mymids, expr = lm( formula = as.formula(X~Y1+Y2+Y3) ) )) ,conf.int = TRUE)

To add variables to the long format output (assuming that the imputation used the default number of datasets i.e. m=5). In this example we are adding the variables bmim and sexe from the original dataframe. :

m=5

Longformat [c('bmim','sexe')] <- as.data.frame(lapply(Original\_DataFrame[c('bmim','sexe')] ,function(x) rep(x,times = m+1) ))

Convert the Longformat to a mids class:

Mymids <- as.mids(Longformat)