MetaAnalyst v1.0

User Manual

Table of Contents

Introduction

MetaAnalyst is a free, user-friendly software package for metagenomic biomarker detection and phenotype classification. MetaAnalyst provides the following functionalities as an efficient analysis tool for metagenomic biomarker detection and phenotype classification:

1. Accepts seven different types of input files.
2. Supports multilevel labeling that enables researchers to analyze the data from different perspectives (i.e., under various conditions) smoothly.
3. Provides a variety of pre-processing procedures before downstream statistical analysis.
4. Includes about 28 biomarker detection algorithms and 4 classifiers
5. Provides three criteria for evaluating the performance of biomarker detection algorithms:
    a. Supervised classification performance: MetaAnalyst computes the overall classification accuracy (ACC), balanced accuracy (BACC), sensitivity (SEN), specificity (SPC), miss classification rate (MCR), receiver operation curve (ROC), and area under the curve (AUC).
    b. Unsupervised clustering performance
    c. Consensus performance
6. Generate the output in tab-delimited files and publication quality plots with various formatting capabilities.

The MetaAnalyst software is implemented using Matlab R2021a and it is available freely as a stand-alone package for Windows and Linux operating systems at:
https://github.com/mshawaqfeh/MetaAnalystV1

Requirements

MetaAnalyst is a standalone desktop application, written in MATLAB and does not require any additional packages to install. MetaAnalyst runs on Microsoft Windows and Linux.

Installation

- Windows version

In order to install MetaAnalyst, follow the below steps:

1. Run the MetaAnalyst.exe file (Fig. 1).

Fig.1 Preparing for installation

2. Press next to continue (Fig. 2).

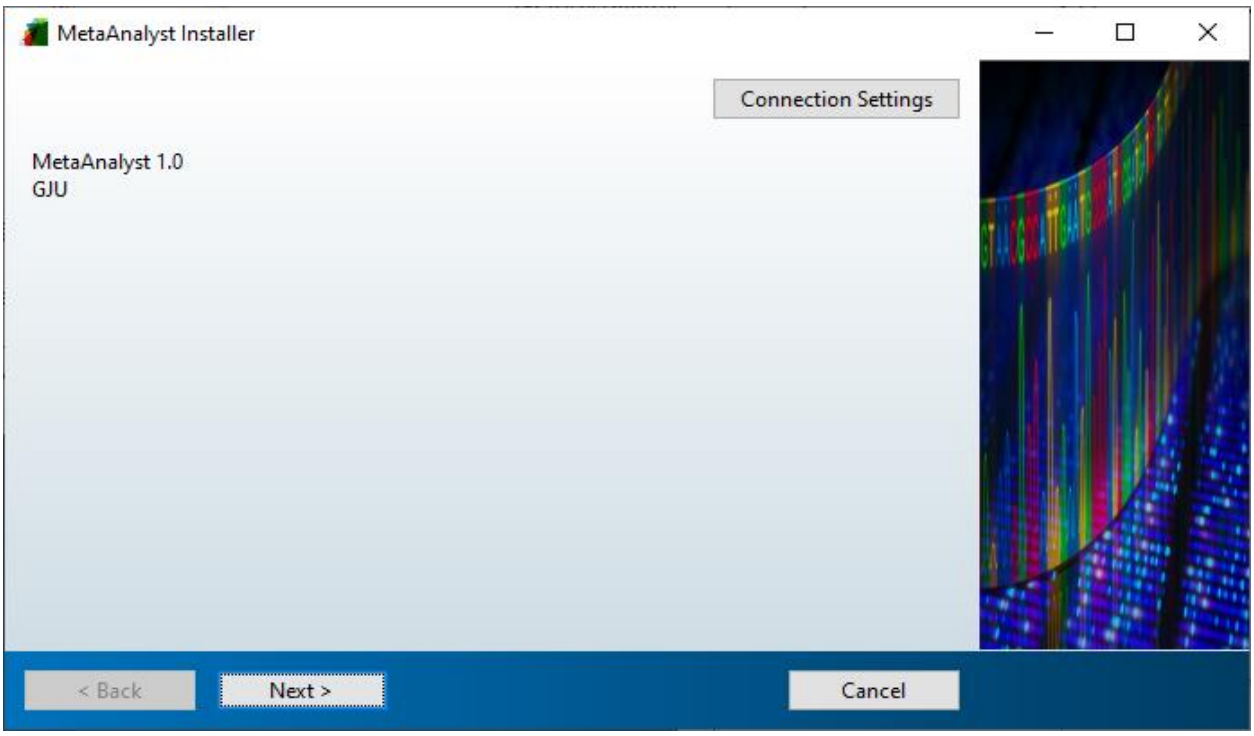


Fig.2 Preparing for installation continue

3. Choose the installation folder for MetaAnalyst, then press next (Fig. 3).

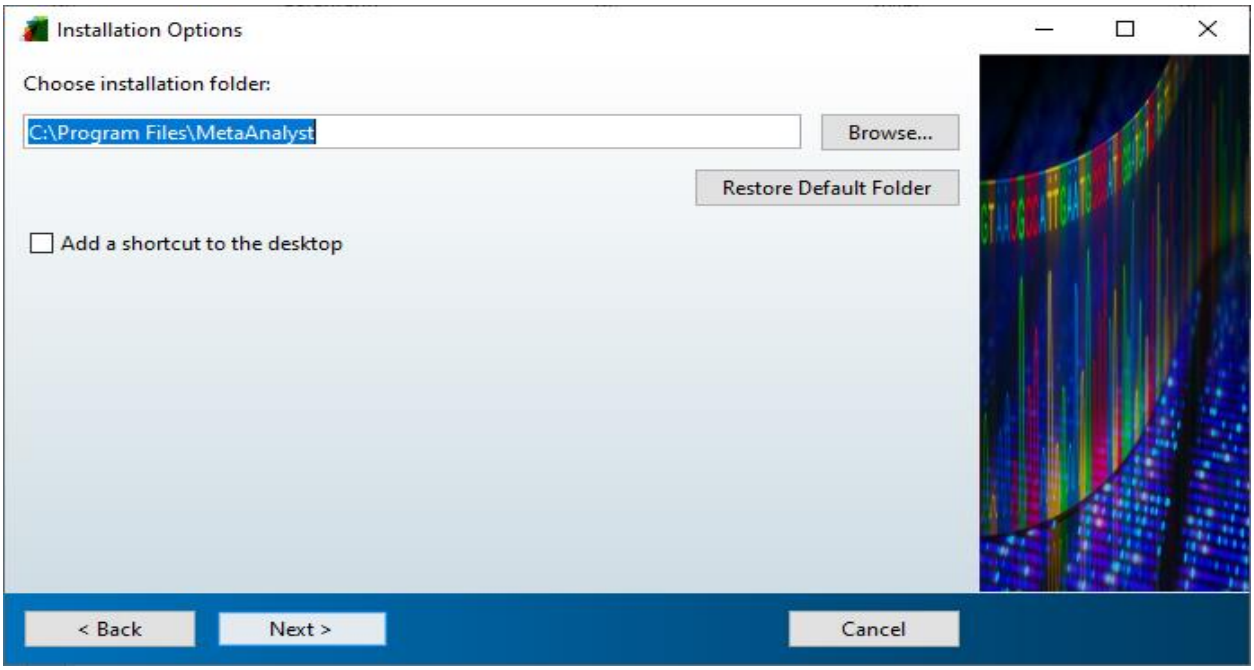


Fig.3 Installation folder for MetaAnalyst

4. Choose the installation path for MATLAB Runtime, then press next (Fig. 4)

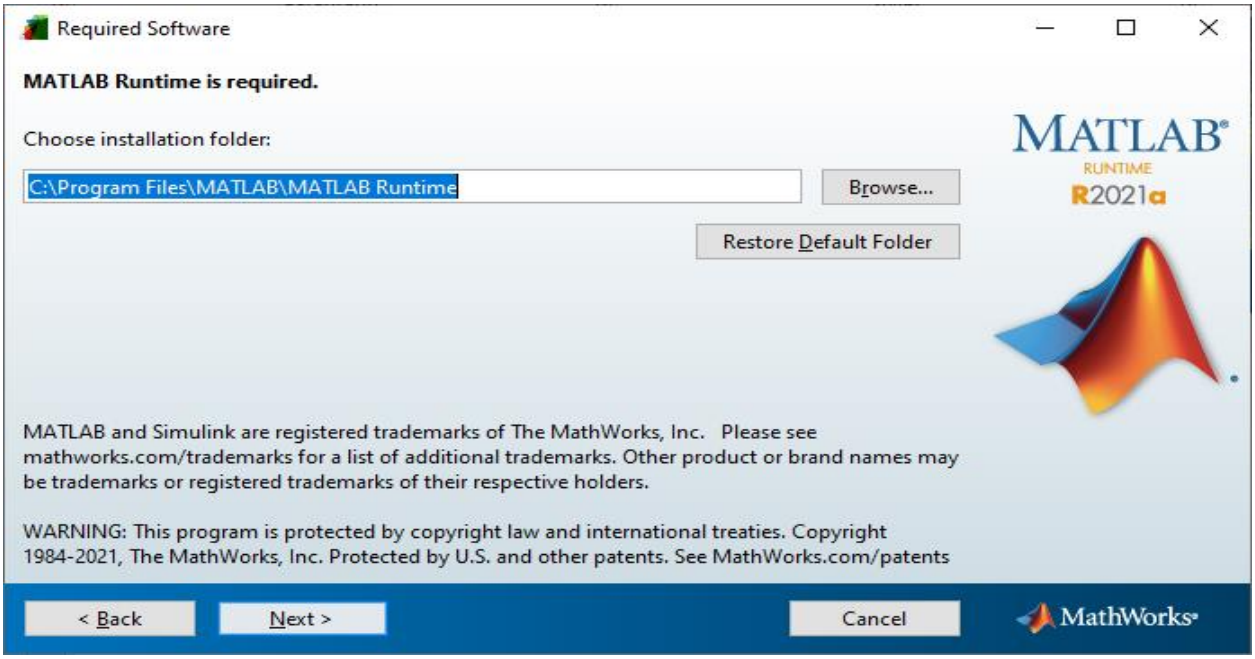


Fig.4 Installation folder for MATLAB Runtime

5. Accept the terms of the license agreement, then press next (Fig. 5)

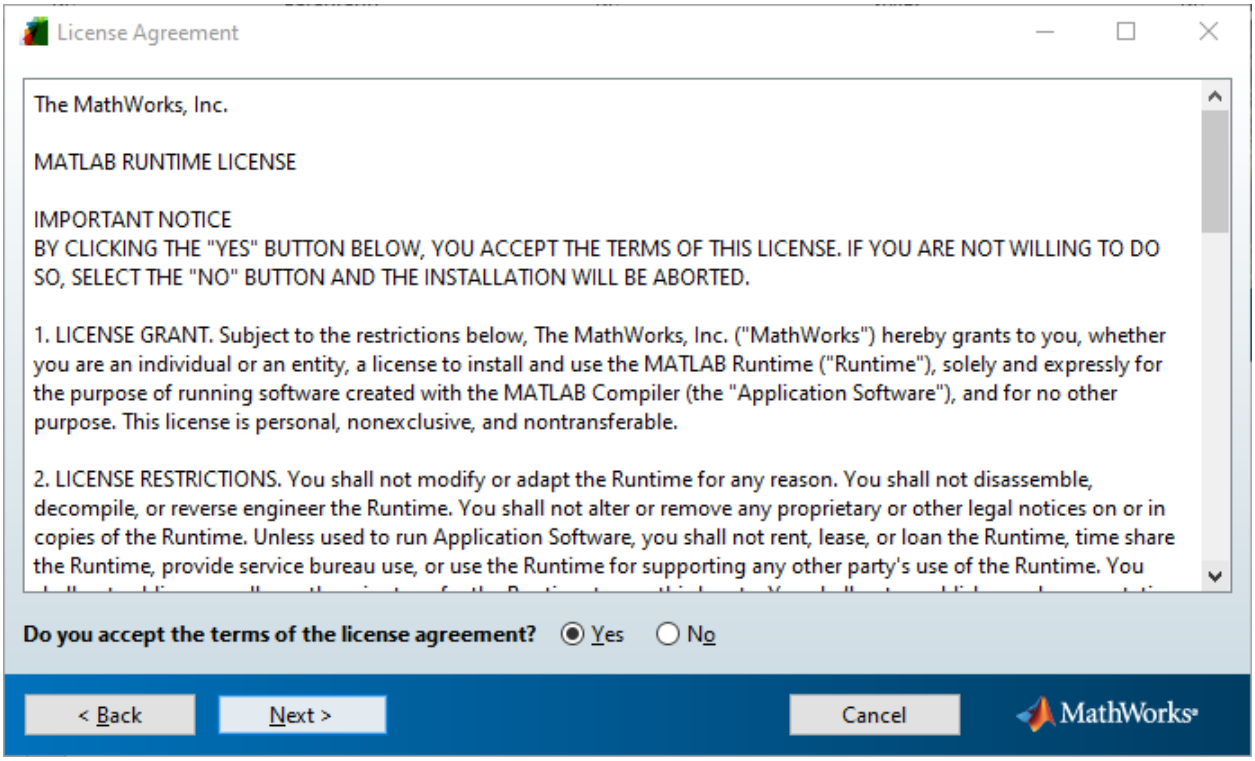
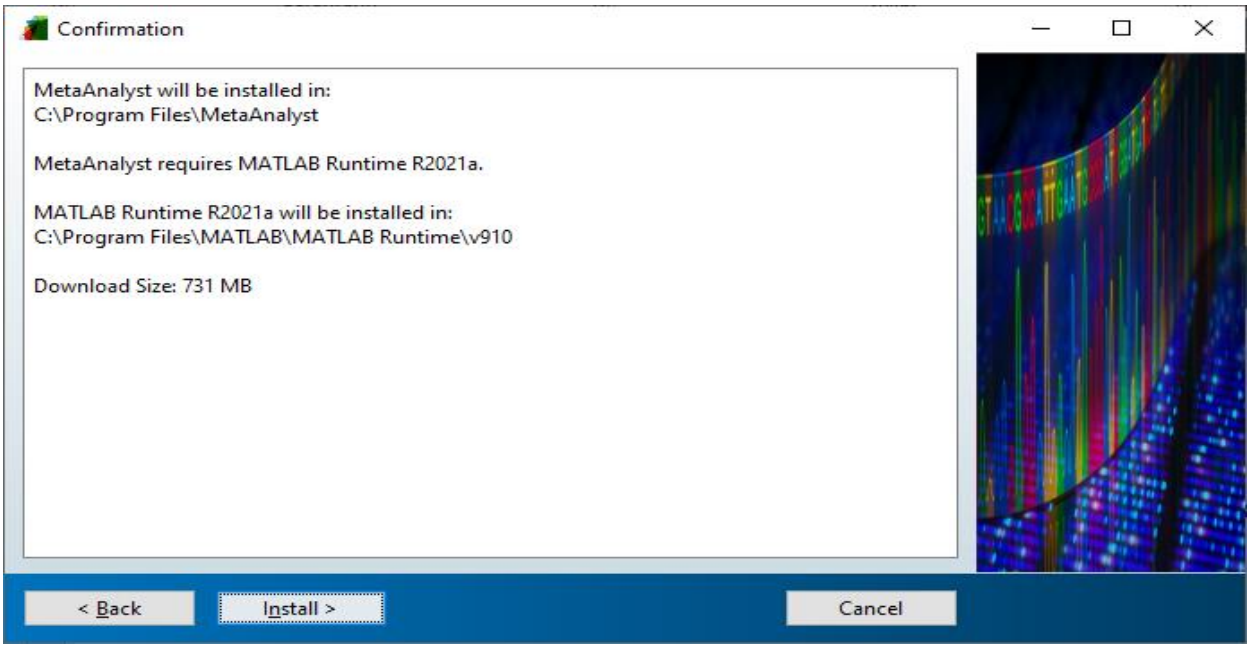
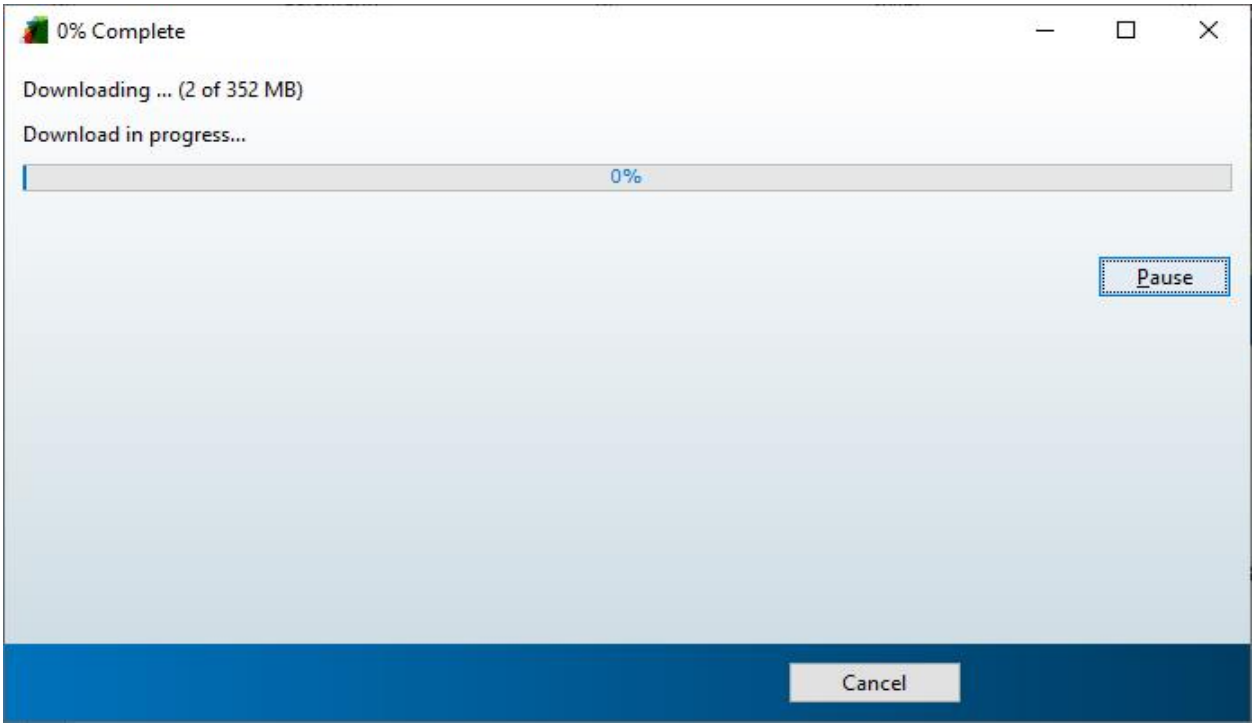


Fig.5 License agreement

6. Press next to confirm installation, then the installer will automatically install MATLAB Runtime version 9.10 (Fig. 6).



(a) Confirm installation



(b) Begin installation of MATLAB Runtime

Fig.6 MATLAB Runtime installation

7. Finally, the installer will begin installing MetaAnalyst (Fig. 7).

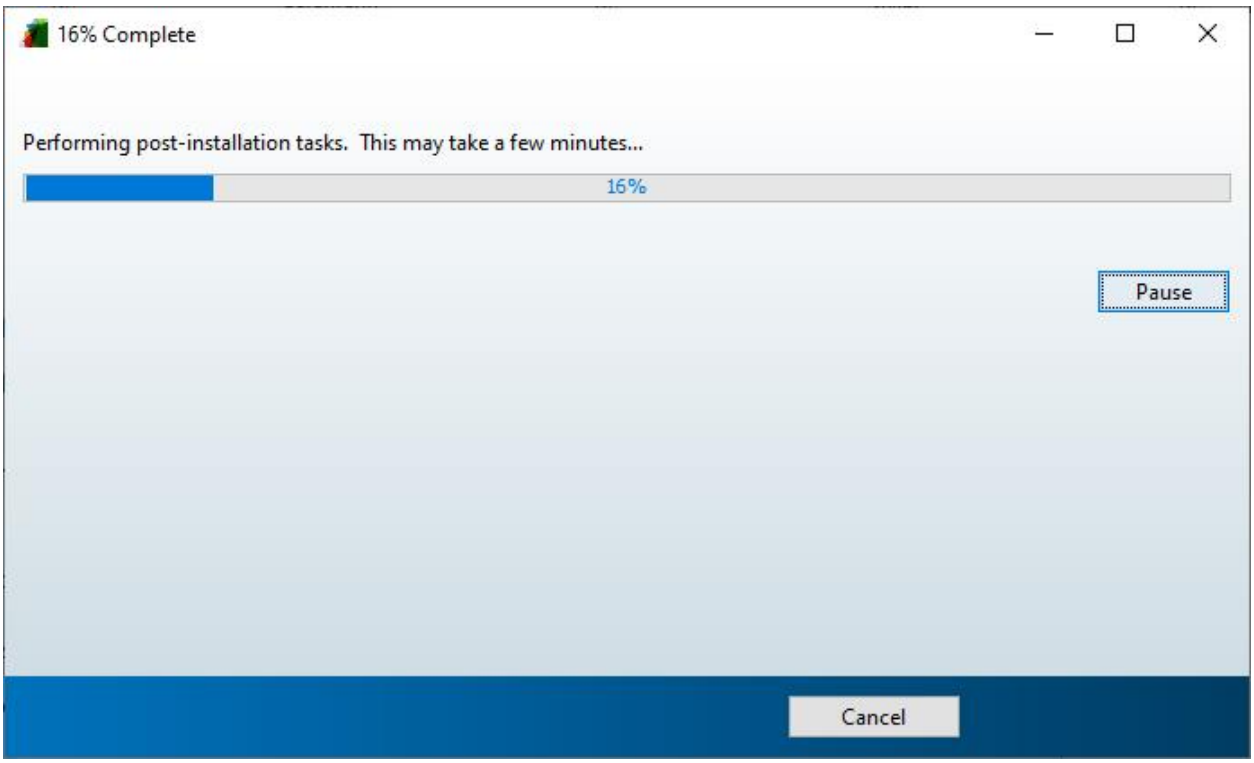


Fig.7 Installing MetaAnalyst

After installation, it is recommended to run the software as administrator. Furthermore, the user should run the software in the installation directory for the first time.

In order to confirm that the installation directory has been added to Windows environment variables properly, the user should follow the below steps:

1. Press start menu and type edit environment variables
2. If the value of the environment variable 'MetaAnalystPath' represents the installation directory of the software, then the path has been added properly to Windows environment variables. Otherwise, the user should manually set the value 'MetaAnalystPath' variable to the directory where 'MetaAnalyst.exe.' file resides.

This step is essential to ensure that MetaAnalyst runs properly on Windows.

MetaAnalyst Tabs

The MetaAnalyst software consists of five tabs, as shown in Fig. 8, that will guide the user smoothly through the analysis pipeline:

1. Input Data
2. Study Design
3. Preprocessing
4. Statistical Analysis
5. Results and Plots

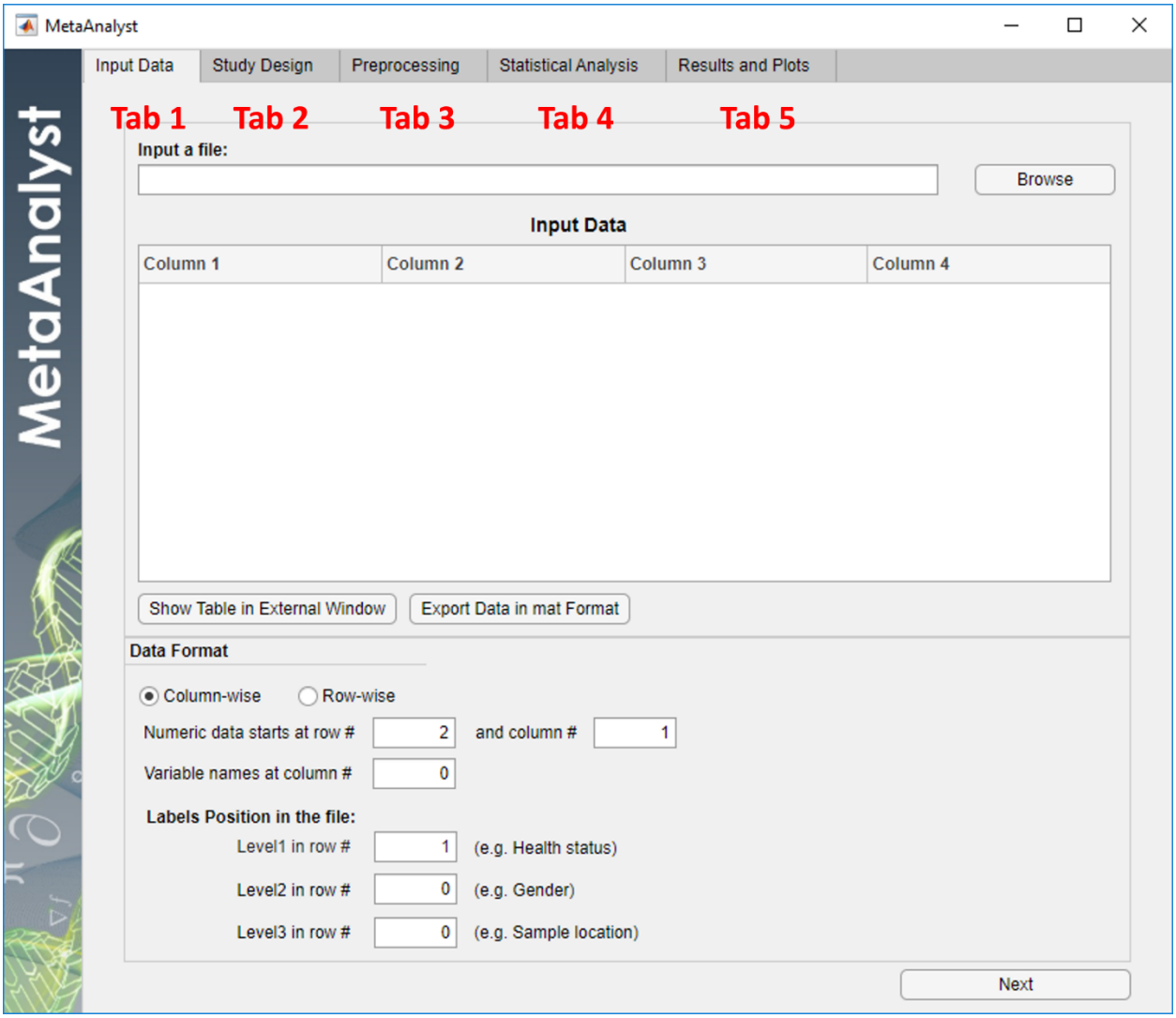


Fig.8 MetaAnalyst tabs

Input Data Tab

This tab aims at loading the input metagenomic data file. In particular, the user is required to upload the input file and fill important parameters related to the input data. As shown in Fig. 9, the input data tab consists of the following:

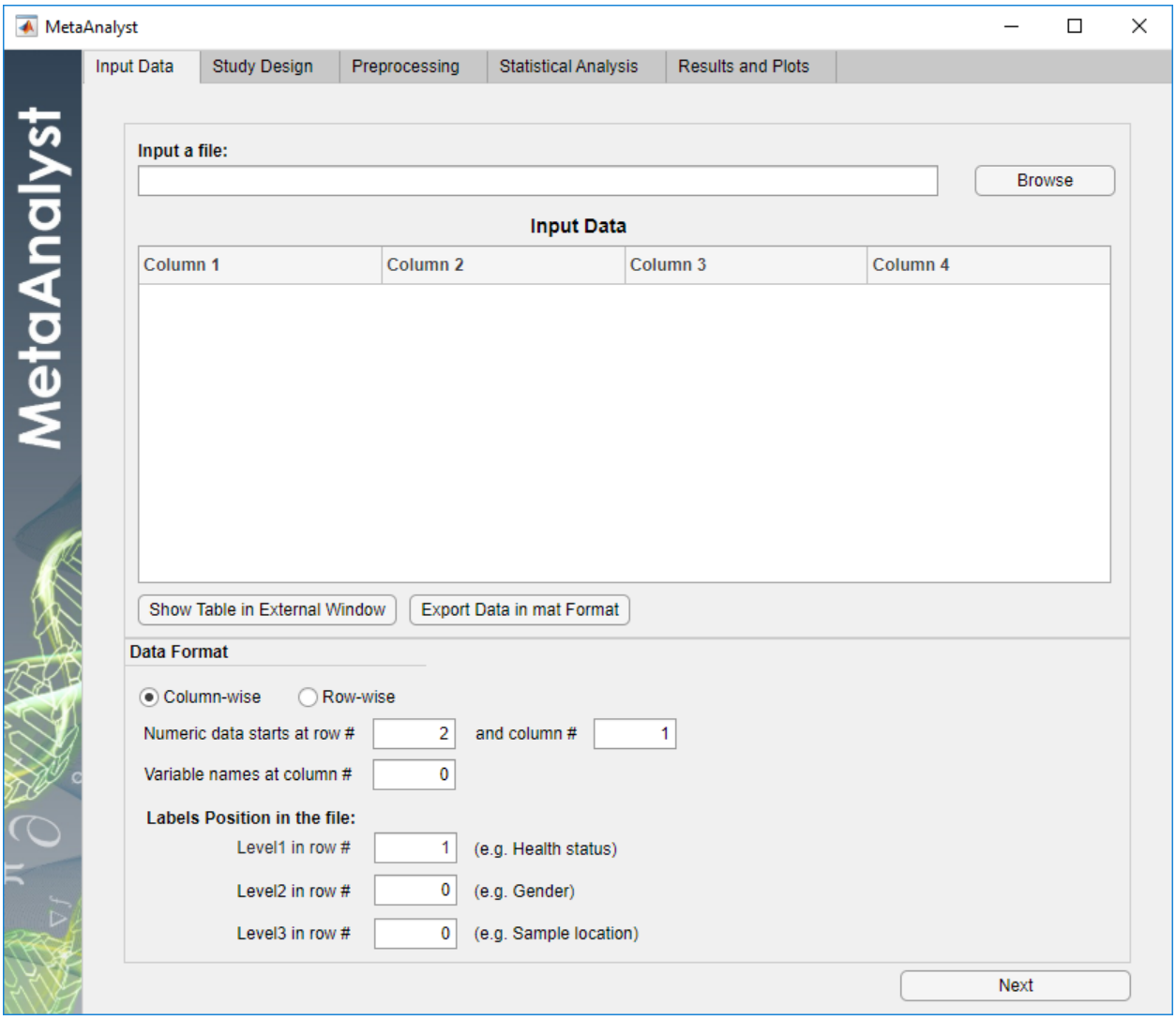


Fig.9 Input data tab

- **Browsing Input File**: To input a file, you need to browse its location by clicking the 'Browse' button, and select the desired file, the software then will automatically read the input data and display the data in the 'Input Data' table (directly below the "Input file" field). To provide the user with a better view of the data (i.e., larger screen), the 'Show in External Window' button displays the data to the

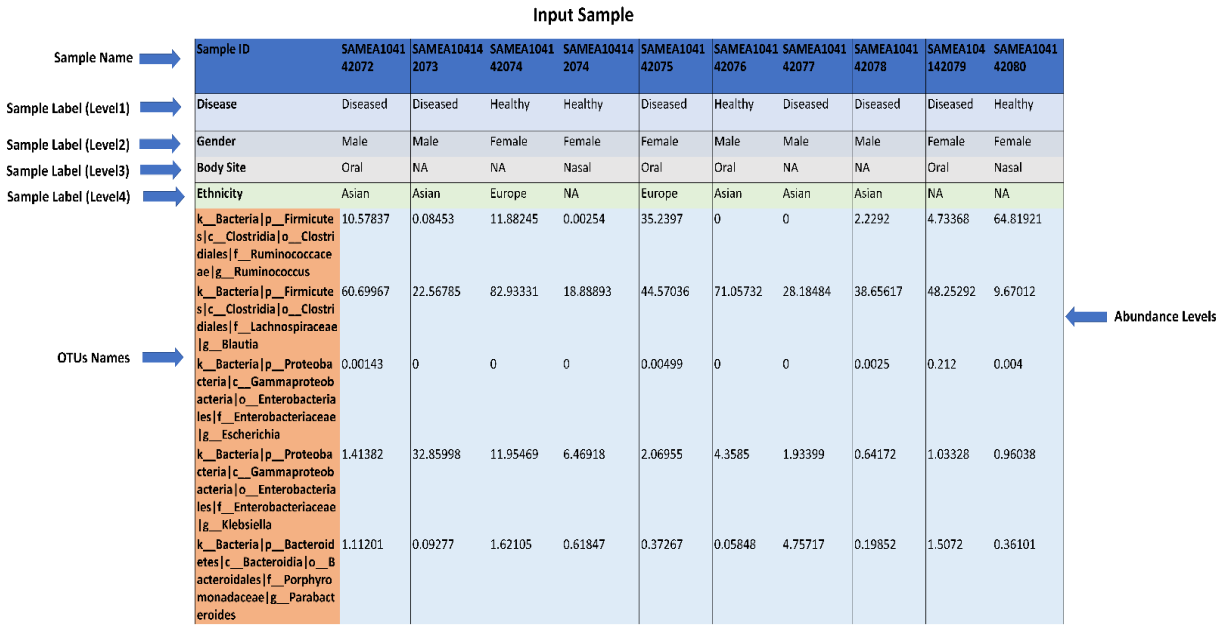
user in an external window. Furthermore, the 'Export Data in mat Format' button allows the user to export a 'mat' (Matlab file) version of the input file. Saving the data as a "mat" file is useful to speed up the process of reading the input data, especially for large input files, for future analysis. This is because MetaAnalyst reads the input data in mat format more rapidly compared to the other available formats.

The software supports seven different types of input files: mat[1], csv, tsv, xls, xlsx, biom[2] and json [3]files. MetaAnalyst reads biom files whether the map file that contains the metadata of the samples is integrated within the biom file or not. However, if no metadata is provided in the biom file, then the first row or column of the biom_data should contain the labels of the samples.

- **Parameters Related to the Input Data**: Input data are expected to have both:
    - Numeric data: represents the abundance levels (count data or ratios) of the operational taxonomic units (OTUs).
    - Metadata: data that provides information about the input file (e.g. samples' IDs, OTUs names, classes/labels of the samples).

  It is important to extract the parameters related to the data (numeric and metadata) accurately from the input file; since all subsequent processing is dependent on proper reading of the data. It is required to determine the locations of the: (1) numeric data, (2) OTUs names, and (3) class labels for each sample. To facilitate this task, the MetaAnalyst software displays the input file automatically in the 'Input Data' table. Hence, the user can extract the required information about the input file directly from the table without referring to the original file. It is worth to mention that this feature is extremely useful when dealing with "biom" files since such files are not human readable and they require other utilities to convert it into readable format.

[1] https://www.mathworks.com/help/matlab/import_export/mat-file-versions.html
[2] https://biom-format.org/
[3] https://docs.fileformat.com/web/json/



Sample ID	SAMEA104142072	SAMEA10414 2073	SAMEA104142074	SAMEA10414 2074	SAMEA104142075	SAMEA104142076	SAMEA104142077	SAMEA104142078	SAMEA104142079	SAMEA104142080
Disease	Diseased	Diseased	Healthy	Healthy	Diseased	Healthy	Diseased	Diseased	Diseased	Healthy
Gender	Male	Male	Female	Female	Female	Male	Male	Male	Female	Female
Body Site	Oral	NA	NA	Nasal	Oral	Oral	NA	NA	Oral	Nasal
Ethnicity	Asian	Asian	Europe	NA	Europe	Asian	Asian	Asian	NA	NA
k__Bacteria|p__Firmicutes|c__Clostridia|o__Clostridiales|f__Ruminococcaceae|g__Ruminococcus	10.57837	0.08453	11.88245	0.00254	35.2397	0	0	2.2292	4.73368	64.81921
k__Bacteria|p__Firmicutes|c__Clostridia|o__Clostridiales|f__Lachnospiraceae|g__Blautia	60.69967	22.56785	82.93331	18.88893	44.57036	71.05732	28.18484	38.65617	48.25292	9.67012
k__Bacteria|p__Proteobacteria|c__Gammaproteobacteria|o__Enterobacteriales|f__Enterobacteriaceae|g__Escherichia	0.00143	0	0	0	0.00499	0	0	0.0025	0.212	0.004
k__Bacteria|p__Proteobacteria|c__Gammaproteobacteria|o__Enterobacteriales|f__Enterobacteriaceae|g__Klebsiella	1.41382	32.85998	11.95469	6.46918	2.06955	4.3585	1.93399	0.64172	1.03328	0.96038
k__Bacteria|p__Bacteroidetes|c__Bacteroidia|o__Bacteroidales|f__Porphyromonadaceae|g__Parabacteroides	1.11201	0.09277	1.62105	0.61847	0.37267	0.05848	4.75717	0.19852	1.5072	0.36101

Fig.10 A sample of input file format

Fig. 10 shows a sample of input file format. Extracting the parameters related to the numeric and metadata from the original input file involves specifying the following fields:

(1) Data Format: Input data can be arranged either in a column-wise or a row-wise format. The input data is considered row-wise if each row represents one sample (i.e., contains the abundance levels of all OTUs for one sample). If the samples are stored in columns (each column represents one sample), then the data is considered column-wise as shown in Fig. 10. MetaAnalyst supports both formats. By default, the input data is considered as column-wise. If not, the user should select the 'Row-wise' radio button.

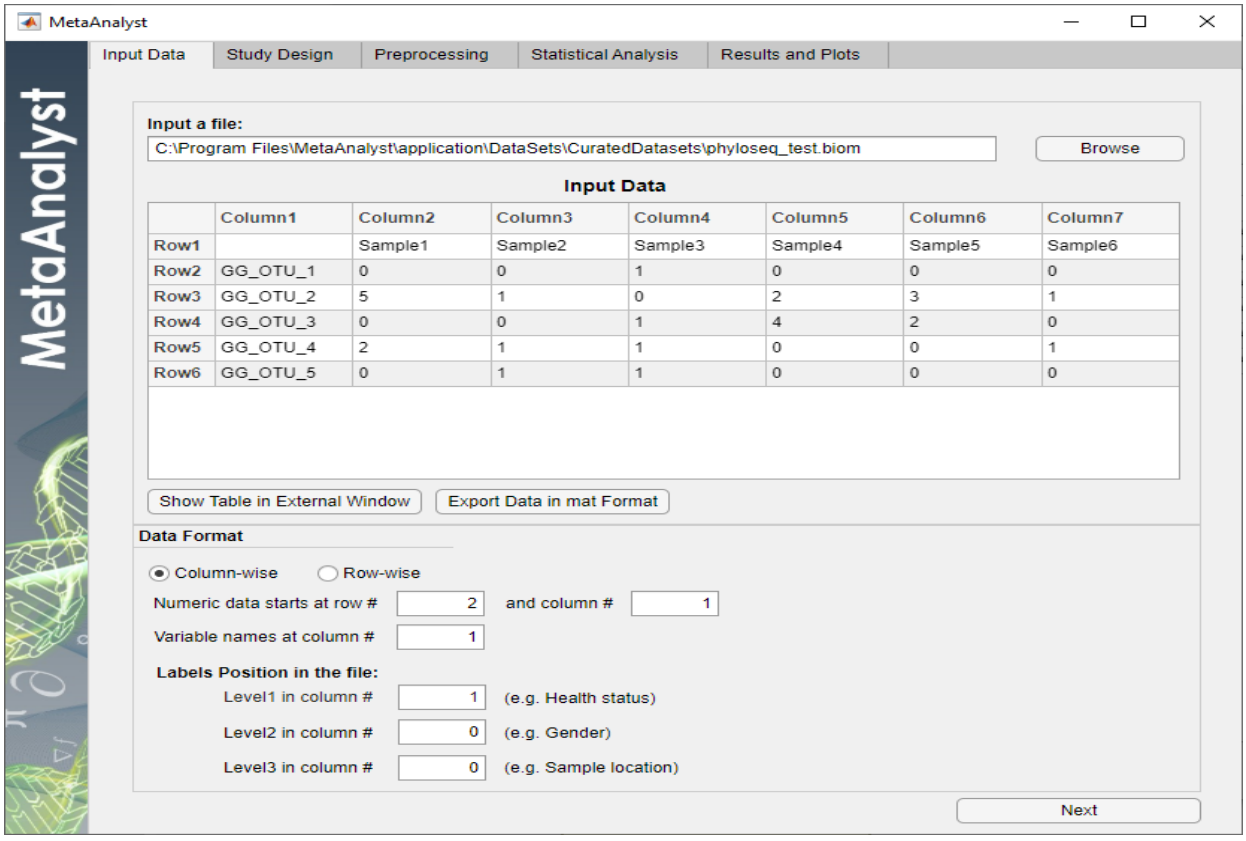
(2) Numeric Data Location in the File: The user is required to determine the location of the top left cell (i.e., the row# and column#) at which the numeric data starts. Then, all data that reside to the right and below *the top-left corner cell* will be extracted as the input OTUs abundance levels. For example, the numeric data in the sample input file (Fig.10) starts at row6 and column2.

(3) Variable Names Location in the File: This field specifies the column/row number at which the variable names (i.e., OTUs names) reside. In case the variable names are not present in the input file, leave this filed empty. *The software will assign the variables automatic dummy names: var1, var2, var3, and so on.* According to the sample input file shown in Fig. 10, the variable names reside in the first column.

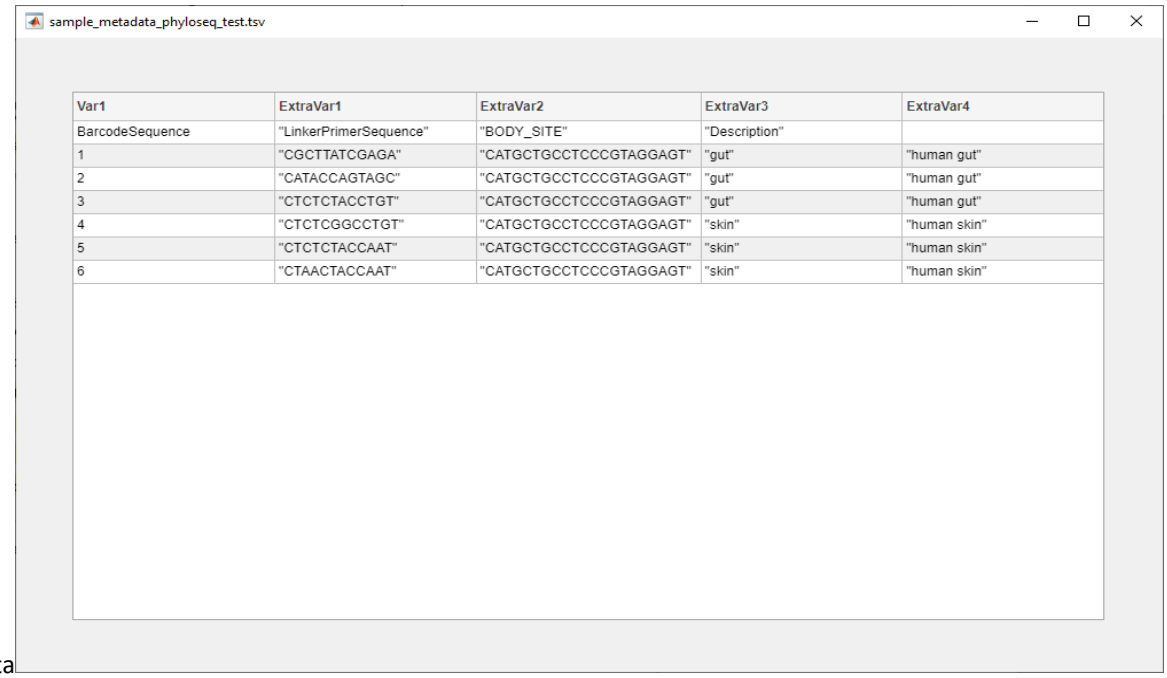
(4) Labels Location in the File: The software supports up to three levels of labeling scheme with AND/OR operations to combine the final representation of the classes. For instance, Level1, Level2 and Level3 may represent the samples health status, gender, and body site location, respectively, as shown in Fig.10 The user is required to input the location of each level, in terms of row# if the data is in column-wise format, and in terms of column# otherwise. In our sample input file, the locations of the levels are at row2, row3 and row4, respectively. Undesired levels can be left zeros. Furthermore, order of the levels in the input file is not important, for instance, Level1 can be in row#4 and Level2 can be in row#1.

In case the input file is in biom format and contains metadata of the samples, a new window displaying the metadata in a tabular form will pop up, as shown in Fig. 11. Thus, allowing the user to input the location of the levels accordingly.

Finally, by clicking the 'Next' button, the software automatically detects the labels at each level and switches to the next tab.



(a) Input data



Var1	ExtraVar1	ExtraVar2	ExtraVar3	ExtraVar4
BarcodeSequence	"LinkerPrimerSequence"	"BODY_SITE"	"Description"	
1	"CGCTTATCGAGA"	"CATGCTGCCTCCCGTAGGAGT"	"gut"	"human gut"
2	"CATACCAGTAGC"	"CATGCTGCCTCCCGTAGGAGT"	"gut"	"human gut"
3	"CTCTCTACCTGT"	"CATGCTGCCTCCCGTAGGAGT"	"gut"	"human gut"
4	"CTCTCGGCCTGT"	"CATGCTGCCTCCCGTAGGAGT"	"skin"	"human skin"
5	"CTCTCTACCAAT"	"CATGCTGCCTCCCGTAGGAGT"	"skin"	"human skin"
6	"CTAACTACCAAT"	"CATGCTGCCTCCCGTAGGAGT"	"skin"	"human skin"

(b) Metadata

Fig.11 Reading biom file with metadata

Study Design Tab

Fig. 12 shows the 'Study Design' tab. This tab allows the user to specify the setup of the study (i.e., construct the positive and negative cohorts). According to the number of selected levels, the associated labels will appear in the respective drop-down menus. Subsequently, the user is required to choose the label representation of the negative and positive classes at each inputted level and the desired operation to combine the final representation of the classes. For example, the negative class may represent healthy females who have normal body mass index, on the other hand, the positive class may represent diseased females who have normal body mass index.

Finally, by pressing on the 'Create Negative and Positive Classes' button, the software will group the samples into positive and negative classes. In addition, MetaAnalyst will display the included samples with their associated labels according to the user designed criteria in the "Data corresponding to the created classes" table. Besides, it will display the number of samples in each class.

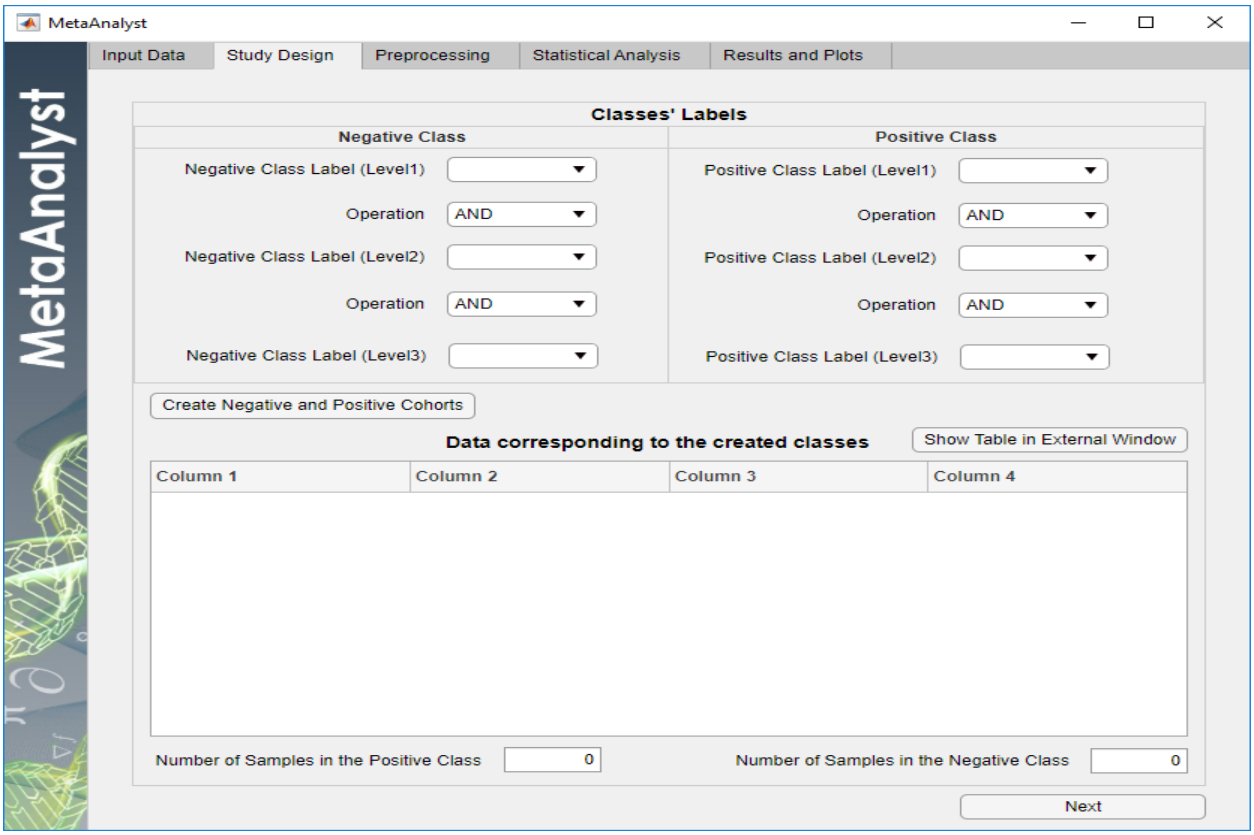


Fig.12 Study design tab

Preprocessing Tab

Data preprocessing is a supplementary step to prepare the data for analysis. MetaAnalyst provides a variety of preprocessing procedures before downstream analysis. These options are included in the 'Preprocessing Tab'. In addition, a descriptive summary statistic before and after preprocessing is displayed, as shown in Fig. 13. The preprocessing techniques can be categorized into *filtering*, *normalization*, and *centering* operations.

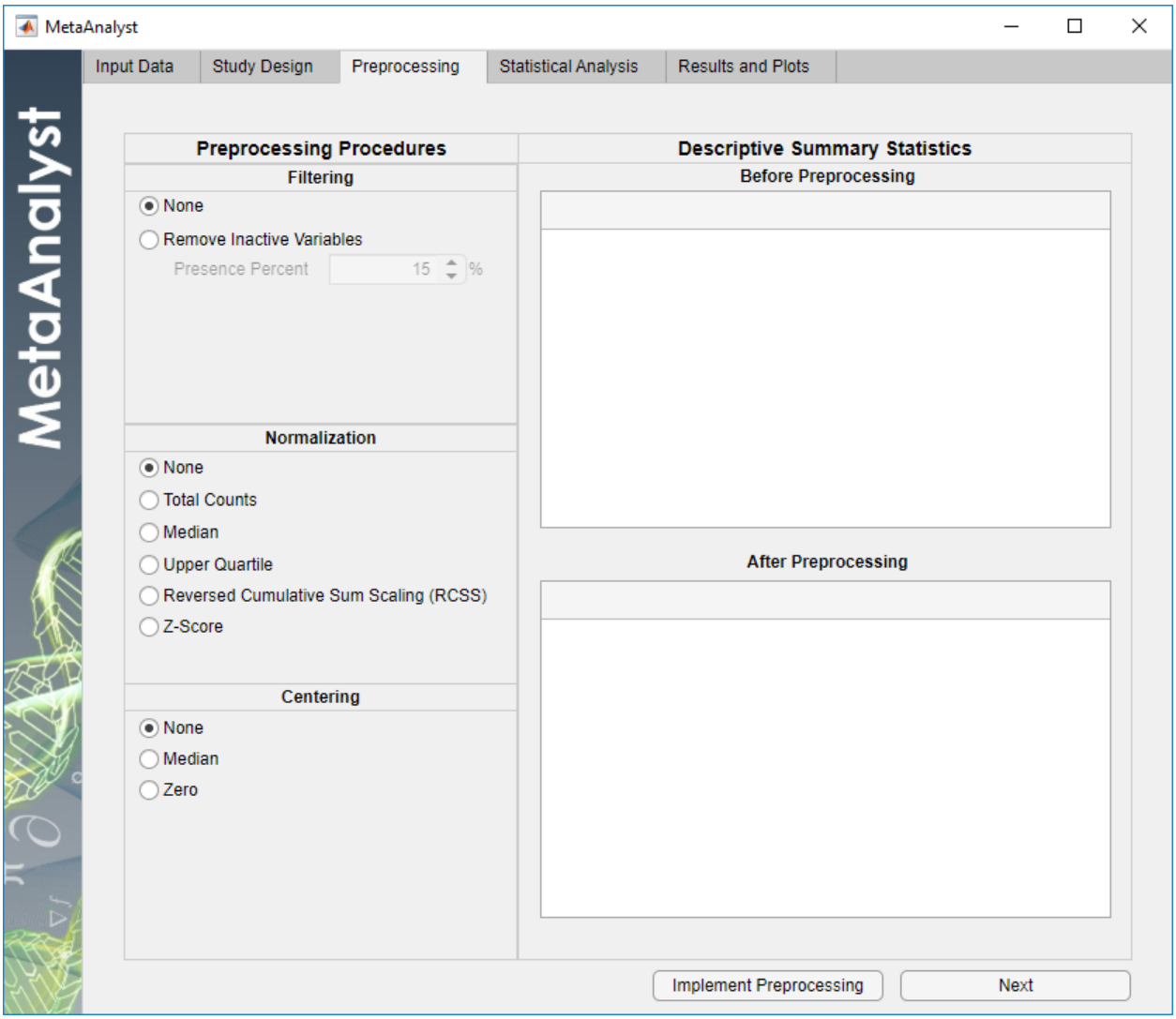


Fig.13 Preprocessing tab

1. **Filtering**: The 'Remove Inactive Variables' option aims at removing the variables/features that are present (i.e., have zero abundance level) in at least a certain percentage of samples. The 'Presence Percent' parameter determines the predefined percentage that a variable must be present in at least "Presence Percent * Number of samples of either class".
2. **Normalization**: Normalization seeks converting the samples to be comparable by removing the systematic variability due to differences in sequence depth. In total, MetaAnlyst provides five normalization techniques:

a. Total Counts: This technique adjusts the abundance level of each variable according to the sum of all abundance levels in each sample. More information about this technique is provided in the following link: https://academic.oup.com/bioinformatics/article/25/15/1849/212949.
b. Median Normalization: The median normalization technique calculates the normalization factor as the median of the abundance levels of the variables in each sample. More information in the following link: https://academic.oup.com/bioinformatics/article/26/1/139/182458.
c. Upper Quartile Normalization: This technique estimates the normalization factor as the sample upper quartile or 75th percentile of the abundance levels of the variables in each sample. The following link provides more information: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-11-94.
d. Reversed Cumulative Sum Scaling (RCSS): This normalization method calculates the normalization factor as the sum of all variables abundance levels that are larger than the median abundance level in each sample. More information is provided in the following link: https://www.liebertpub.com/doi/10.1089/cmb.2016.0180.
e. Z-Score Normalization: This method transforms the abundance levels of the variables according to their distance from their mean. The following link provides more information: https://en.wikipedia.org/wiki/Standard_score.

3. **Centering**: Centering aims to convert the abundance levels of the variables to fluctuations around a specific value, such as zero or median. MetaAnalyst provides two centering methods: Median Centering and Zero Centering.

After running the selected preprocessing procedures, MetaAnalyst will update the 'After Preprocessing' table to allow the user to compare the summary statistics before and after preprocessing the input data. The content of the two tables provides information about the number of samples, number of features, mean, standard deviation, minimum, 25th percentile, 50th percentile, 75th percentile and the maximum abundance value.

Statistical Analysis Tab

Fig. 14 shows the 'Statistical Analysis' tab. The functionalities of this tab can be divided into the following:

1. Biomarker detection.
2. Phenotype classification.

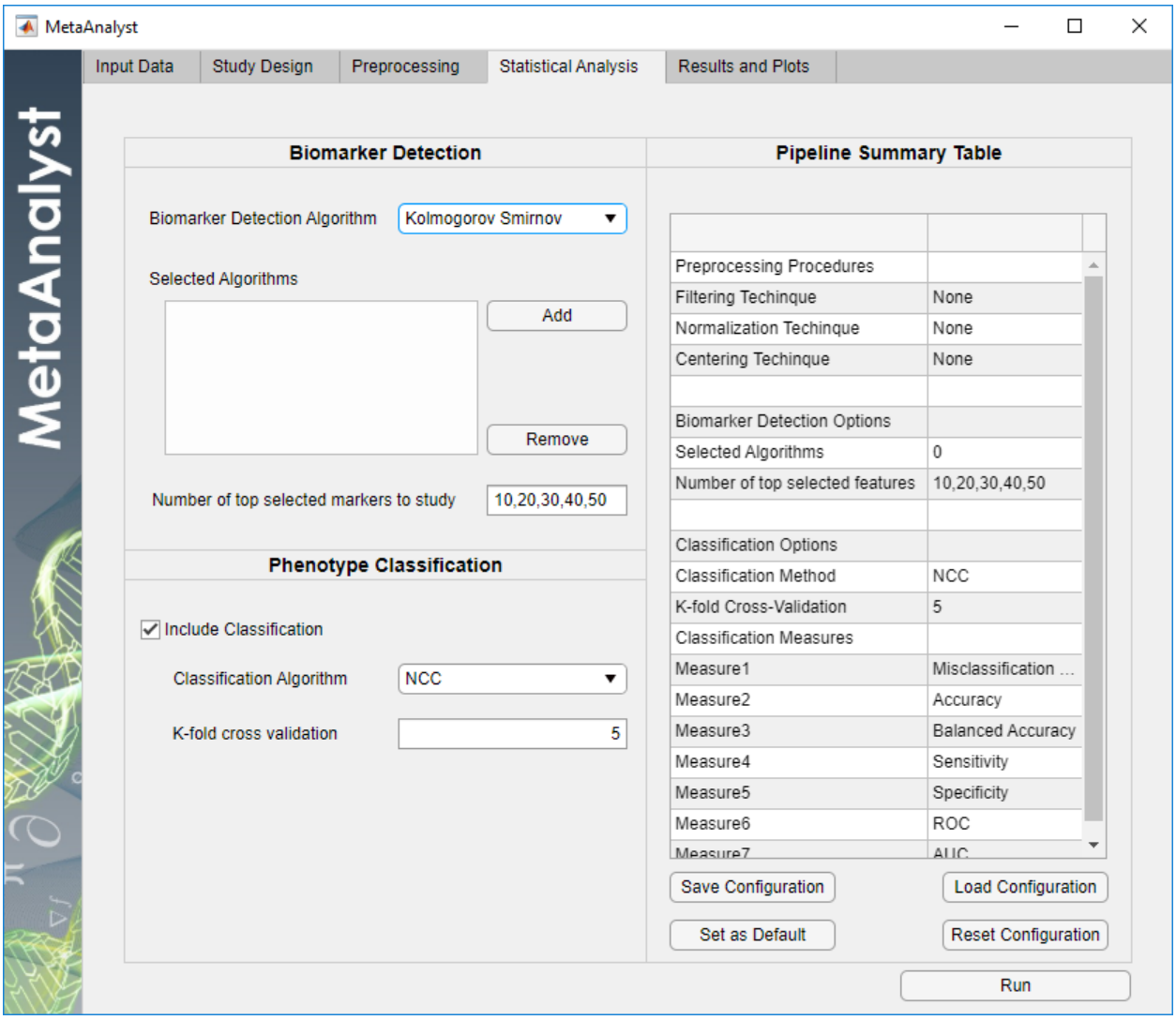


Fig.14 Statistical analysis tab

(1) Biomarker Detection: The options included in this section are the 'Biomarker Detection Algorithm' drop down menu, the 'Selected Algorithms' box, the 'Add' and 'Remove' buttons, and the 'Number of top selected markers to study' edit field box. The user can include multiple biomarker detection

algorithms in the pipeline analysis by pressing on the 'Add' button to add the method to the 'Selected Algorithms' box. The software packs 28 biomarker detection algorithms. Table 1 lists all the available algorithms and presents a brief description for each algorithm.

In addition to the capability of MetaAnalyst to run several biomarker detection algorithms simultaneously, it also enables the user to run the selected algorithms at various number of top features (i.e., biomarkers) simultaneously. In particular, after adding the selected algorithms(s) to the pipeline analysis, the user can input the number of top selected markers to study. By default, the number of top selected markers is set to a range of values equal to 10,20,30,40,50. However, the software accepts three different forms of values as an input:

(a) Single value (e.g. 100)

(b) Range in the form of *Start:Step:End* (e.g. 15:10:100)

(c) Multiple values separated by a comma (e.g. 10,20,40,80)

(2) Classification: To construct a prediction model for the input data, the 'Include Classification' checkbox should be enabled, as shown in Fig. 15.

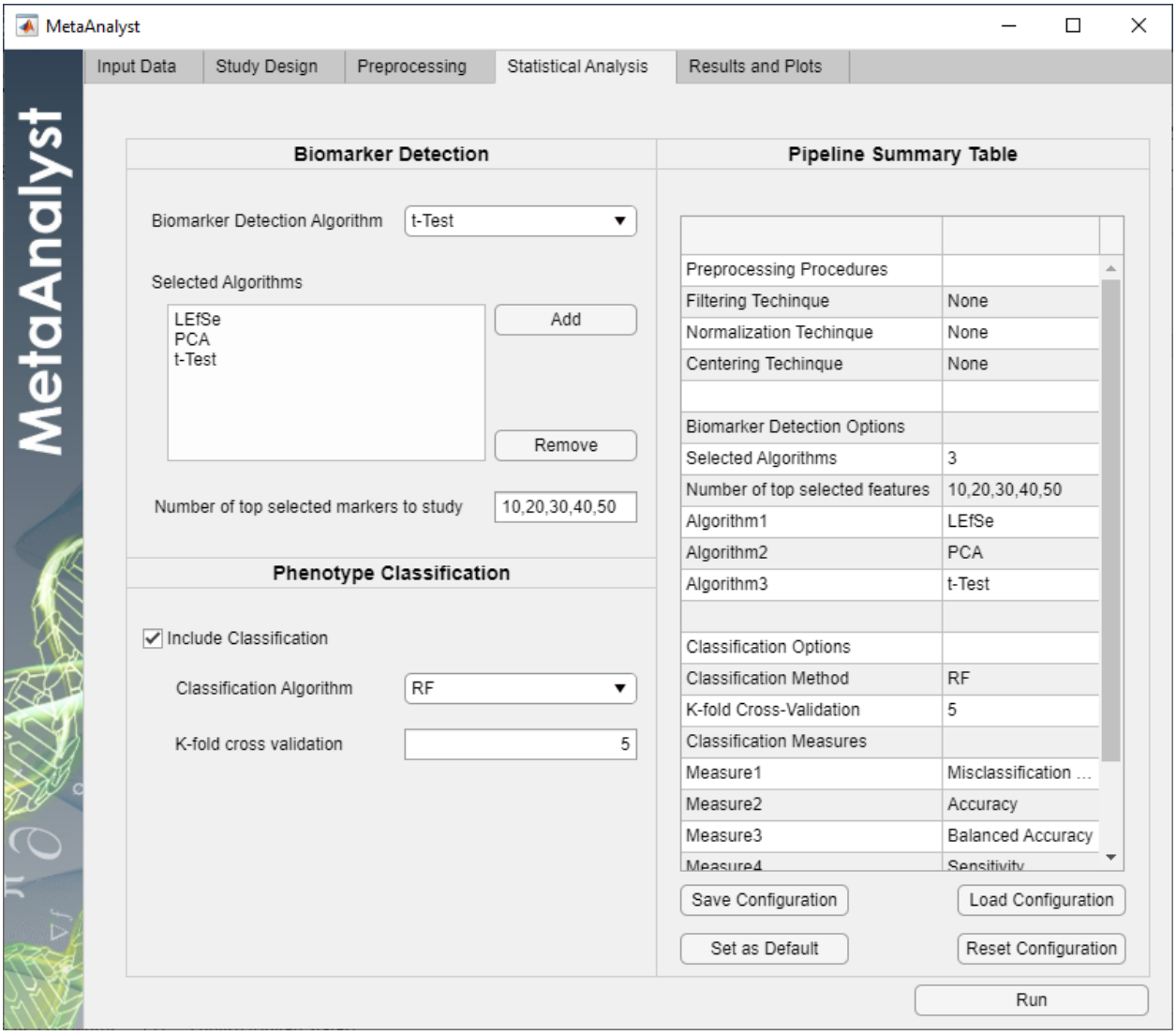


Fig.15 Biomarker detection and phenotype classification

The software provides four different classifiers:

- NCC (Nearest Centroid Classifier): NCC is a classification model that labels new data points based on the nearest centroid. MetaAnalyst provides two versions of NCC, NCC_L1 which uses L1 norm (Manhattan distance) as a measure of distance and NCC_L2 which uses L2 norm (Euclidean distance) as a measure of distance. https://en.wikipedia.org/wiki/Nearest_centroid_classifier.
- SVM (Support Vector Machine): SVM is a supervised learning algorithm that constructs a hyperplane in space to separate between the classes of the input data. The software

allows the user to change the kernel function of the SVM classifier, which include linear, gaussian, polynomial and rbf. https://en.wikipedia.org/wiki/Support-vector_machine.

- kNN (k-Nearest Neighbor): kNN is a simple machine learning algorithm that relies on the k-nearest neighbors to classify new data points. The user can change the number of neighbors parameter of the kNN model. https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm.
- RF (Random Forest): RF is an ensemble learning algorithm that builds multiple decision trees to improve prediction. The software allows the user to change the number of trees parameter for the RF model. https://en.wikipedia.org/wiki/Random_forest#:~:text=Random%20forests%20or%20random%20decision,average%20prediction%20(regression)%20of%20the.

After choosing the classifier, there is an option for the user to alter the number of folds for k-fold cross-validation via the 'K-fold cross-validation' edit box. To measure the performance of the model, the software provides seven different classification metrics:

- Misclassification Rate: the proportion of misclassified samples or observations.https://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/ .
- Accuracy: accuracy is defined as the number of correct predictions over the total number of predictions. https://www.jeremyjordan.me/evaluating-a-machine-learning-model/#:~:text=The%20three%20main%20metrics%20used,the%20number%20of%20total%20predictions.
- Balanced Accuracy: it is a classification metric used to evaluate the performance of a model in case of imbalanced classes. https://statisticaloddsandends.wordpress.com/2020/01/23/what-is-balanced-accuracy/
- Sensitivity: also known as true positive rate or recall. https://en.wikipedia.org/wiki/Sensitivity_and_specificity.
- Specificity: also known as true negative rate. https://en.wikipedia.org/wiki/Sensitivity_and_specificity.

- ROC (Receiver Operating Characteristics): ROC curve is a graph that illustrates the performance of a classification model at various threshold settings. https://en.wikipedia.org/wiki/Receiver_operating_characteristic.
- AUC (Area Under the Curve): AUC is defined as the area under the ROC curve, the larger the area the better the prediction of a model. https://en.wikipedia.org/wiki/Receiver_operating_characteristic#Area_under_the_curve.

(3) Pipeline Summary Table: This table contains the summary of the constructed analysis from the available options in the evaluation tab, which are the preprocessing procedures, the biomarker detection options and the classification options. As shown in the Fig. 14, the table initially indicates that the pipeline analysis is empty. However, as the user selects from the available options, the table instantly updates its status (Fig. 15).

(4) Configuration Settings: To allow reusability of the configuration settings/pipeline analysis, there are four available options for the user. Firstly, the user can save the current configuration into a new file, by pressing on the 'Save Configuration' button, and load this file whenever needed by pressing on the 'Load Configuration' button. Furthermore, the 'Set as Default' button allows the user to set the current configurations as the default configurations. Then, whenever the user opens the application again, the software will automatically load the selected default configurations. Finally, the 'Reset Configuration' button empties the pipeline analysis settings and restore the default configurations of the software.

Finally, by pressing on the 'Run' button, the software implements the selected pipeline analysis, then, a new window will appear indicating that the analysis has been executed successfully.

Table 1 Biomarker Detection Algorithms

Method	Description/Implementation
Kolmogorov Smirnov Test	**Description:** It is a nonparametric test used to compare a sample with a reference probability distribution. It quantifies the distance between the empirical cumulative distribution function of the sample and the cumulative distribution function of the reference distribution, or between empirical cumulative distribution functions of two samples. **Implementation:** Useful information about Kolmogorov Smirnov test is available in: itl.nist.gov/div898/handbook/eda/section3/eda35g.htm
Wilcoxon Rank Sum Test	**Description:** The Wilcoxon test is a nonparametric statistical hypothesis test that compares the means of two dependent samples and assesses the statistical significance. It is used as an alternative of the t-Test when the population does not follow a normal distribution. **Implementation:** Original research article can be accessed in: https://www.jstor.org/stable/2236101?seq=1
Levene Absolute Test	**Description:** Levene Absolute test is an inferential test used to test the null hypothesis that two groups of data have equal variances. It computes the between all data points and their mean to perform analysis of difference. **Implementation:** More information about Levene test algorithms is available: https://www.itl.nist.gov/div898/handbook/eda/section3/eda35a.htm
Levene Quadratic Test	**Description:** This version of Levene's test uses the squared deviations of the data points from their group means to perform the analysis of variances. **Implementation:** More information about Levene test algorithms is available: https://www.itl.nist.gov/div898/handbook/eda/section3/eda35a.htm
Brown Forsythe Test	**Description:** Brown Forsythe test is a statistical hypothesis test used for testing the equality of group variances in Analysis of Variance (ANOVA). It is a modification of the Levene Test. **Implementation:**

	Original paper of Brown Forsythe test can be found in: **https://www.tandfonline.com/doi/abs/10.1080/01621459.1974.10482955**
LEfSe	**Description:** LEfSe (Linear discriminate analysis effect size) is a software tool used to identify biomarkers between two or more samples of data using relative abundances. LEfSe utilizes the factorial Kruskal-Wallis (KW) rank sum test to detect the most informative feature and measures the consistency of the selected features using Wilcoxon rank sum test. Furthermore, it utilizes LDA with effect size estimation to perform dimension reduction if desired.. **Implementation:** LEfSe paper can be found in: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3218848/
RPCA	**Description:** Robust Principal Component Analysis (RPCA) is a consistency-classification framework that enables the assessment of consistency and classification performance of a biomarker discovery algorithm. This evaluation protocol is based on random resampling to mimic the variation in the experiment size. **Implementation:** Original paper can be accessed in: https://biologydirect.biomedcentral.com/articles/10.1186/s13062-017-0175-4
RegLRSD	**Description:** RegLRSD (Regularized Low Rank-Sparse Decomposition) is a reliable biomarker detection algorithm. RegLRSD models the bacterial abundance data as a superposition between a sparse matrix and a low-rank matrix, which account for the differentially and non-differentially abundant microbes, respectively **Implementation:** Original paper can be accessed in: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-017-1738-1
Pearson Correlation	**Description:** Pearson Correlation is a dimensionless measure of how two quantities vary together. It is the ratio between the covariances of the two groups and the product of their standard deviations. **Implementation:** Useful information about Pearson Correlation is available in: **https://link.springer.com/chapter/10.1007%2F978-3-642-00296-0_5**
BSS/WSS	**Description:**

	BSS/WSS (Between Sum of Squares over Within Sum of Squares) measures the strength of the relationship between the independent and dependent variable. It is often used to determine the best fitted statistical model that represents the data. **Implementation:** More information can be found in: **http://janda.org/c10/Lectures/topic07/L20-ANOVAintro.htm**
Relief	**Description:** Relief is a filter-based feature selection approach that takes into account feature interactions. It calculates a score for each feature to determine the most informative features. The scoring is computed based on identification of feature value differences between nearest-neighbor instance pairs. **Implementation:** Useful information about Relief feature selection can be found in: https://www.sciencedirect.com/science/article/pii/S1532046418301400
ReliefF	**Description:** ReliefF is a variation of the Relief feature selection Algorithm that uses other metrics to calculate the nearest neighbors and differences between nearest neighbor instance pair. It extends Relief capabilities to handle noisy and missing data problems. **Implementation:** Useful information about ReliefF variation can be found in: https://www.sciencedirect.com/science/article/pii/S1532046418301400
Lasso	**Description:** Lasso (Least Absolute Shrinkage and Selection Operator) is a powerful method that performs prediction and feature selection. It applies a shrinkage operator to penalize the coefficients of the model, shrinking some of them to zero. After the model is trained, the features that have a non-zero coefficients are selected to be the most informative features. **Implementation:** Lasso paper can be found in: https://statweb.stanford.edu/~tibs/lasso/lasso.pdf
RSPCA	**Description:** sparse PCA via regularized Singular Value Decomposition (sPCA-rSVD) method was proposed to address the interpretation problem of PCA. It also accounts for any sparsity in the data. **Implementation:** sPCA-rSVD paper is accessible in: **https://www.sciencedirect.com/science/article/pii/S0047259X07000887**
Boruta	**Description:**

	The Boruta algorithm is based on two main ideas, shadowing the features and attach them to the original data. Then, a random forest model is constructed to measure the importance of each original feature and compares it to a threshold value, which is defined as the highest feature importance recorded among the shadow features. **Implementation:** R-Package: https://cran.r-project.org/web/packages/Boruta/index.html
t-Test	**Description:** A t-Test is an inferential statistics test aims to determine the statistical significance between the means of two groups of data. The test is performed when the test statistic follows the student's t-distribution under the null hypothesis. **Implementation:** More information can be found in: **https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4667138/**
Square t-Test	**Description:** This is basically a square root transformation of the data followed by a t-test. **Implementation:** More information can be found in: **https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4667138/**
Log t-Test	**Description:** The Log t-Test is basically a t-Test applied on the transformed version of the data, which is preprocessed using log-transformation **Implementation:** Useful information about Log transformation is available in: https://www.bmj.com/content/345/bmj.e6727
Welch's Test	**Description:** The Welch's test increases the test power for samples of unequal variance compared to the t-Test. It accomplishes this by modifying the degrees of freedom. Therefore, the Welch's test is more reliable than the t-Test when the two groups of samples have different sizes and variances. **Implementation:** The Welch's test paper can be found in: https://academic.oup.com/biomet/article-abstract/34/1-2/28/210174?redirectedFrom=fulltext
Chi-square Test	**Description:** A Chi-square test is a statistical hypothesis test used to test the independence of two events. In machine learning, the Chi-square test is to determine whether there is a

	statistically significant difference between the observed frequencies (data) and the expected frequencies, also known as the contingency table, under the null hypothesis. **Implementation:** More information can be found in: https://link.springer.com/chapter/10.1007/978-1-4612-4974-0_43
edgeR Exact Test	**Description:** edgeR is a software package designed to perform differential expression analysis of count-based expression data. It implements various statistical methodologies to detect the most informative features, which include empirical Bayes estimation, exact tests, generalized linear models and quasi-likelihood tests. **Implementation:** Bioconductor Package: https://bioconductor.org/packages/release/bioc/html/edgeR.html
DESeq2	**Description:** DESeq2 is a software package designed to analyze high-dimensional count data for differentially expressed genes. It utilizes negative binomial (NB) distributions mode to test for differential expression in count data and evaluate mean-variance dependence from high-throughput sequencing assays. **Implementation:** Bioconductor Package: https://bioconductor.org/packages/release/bioc/html/DESeq2.html
MetagenomeSeq	**Description:** MetagenomeSeq is a software package designed to detect differentially abundant features between two or more groups of microbial conditions. This package utilizes a zero-inflated Gaussian mixture model with normalization to account for measurements across taxonomic features. **Implementation:** Bioconductor Package: https://bioconductor.org/packages/release/bioc/html/metagenomeSeq.html
MetaStats	**Description:** MetaStats is a statistical approach relying on the frequency data to identify differentially abundant features in clinical metagenomic data. It improves the specificity in high-complexity environments and addresses sparsely-sampled features by utilizing the false discovery rate and Fisher's exact test. **Implementation:** Original paper is available in: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2661018/
ENNB	**Description:**

	ENNB a two-stage statistical approach for detecting differentially abundant features between two or more groups of microbial conditions. The first stage reduces the dimension of the metagenomic data by using elastic net to select the informative features. Then, the differentially abundant features are identified in the second stage using generalized linear models with a negative binomial distribution **Implementation:** ENNB paper can be found in: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4287949/
RAIDA	**Description:** RAIDA (Ratio Approach for Identifying Differential Abundance) is a robust method for identifying differentially abundant features. It utilizes the ratio between features in a modified zero-inflated lognormal model. **Implementation:** The following is a link to the original paper: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4495302/
MicrobiomeDDA	**Description:** MicrobimeDDA is a novel method for testing the abundance, prevalence and dispersion of metagenomic datasets. The method is based on the zero inflated negative binomial regression model and data winsorization. **Implementation:** The paper is available in: https://academic.oup.com/bioinformatics/article/34/4/643/4470360
ShotgunFunctionalizeR	**Description:** ShotgunFunctionalizeR is a software package designed to analyze and compare metagenomic data. It introduced a novel approach based on Passion linear model that offers higher flexibility in experimental design compared to previous approaches. **Implementation:** R-package: http://shotgun.math.chalmers.se/

Results and Plots Tab

Fig. 16 shows the 'Results and Plots' tab. The functionalities provided in this tab can be divided into the following:

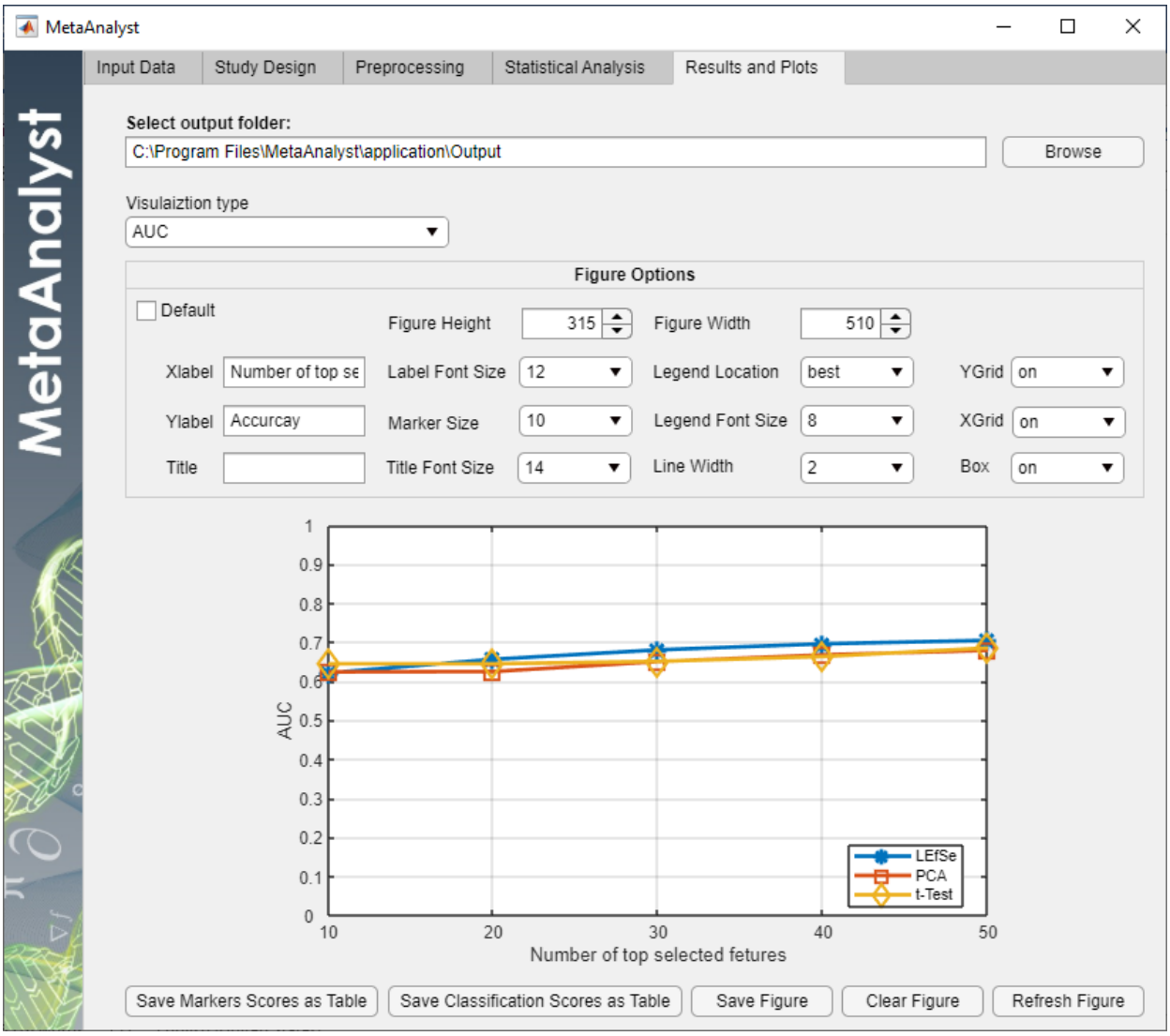


Fig. 16 Results and plots tab

(1) **Specifying Output Path**: By default, the software considers the "Output' folder in the installation directory as the output path. However, the user can change the output path by pressing on the 'Browse' button.

(2) **Plots**: The MetaAnalyst software provides several plots to present the obtained results. These plots can be classified as:

Detected Biomarkers

This figure displays the top identified markers by each biomarker detection algorithm and their scores as a horizontal bar graph. The options available for this plot are the 'Number of top selected features' and the 'FS Method' drop down menus, allowing the user to choose the number of top selected markers and the biomarker detection algorithm. As shown in Fig., the blue and red bars represent the markers that are enriched in healthy and diseased samples, respectively.

Unsupervised clustering performance

Clustermap of Detected Biomarkers: This figure displays the hierarchical clustering analysis on the significantly identified differential markers as a heatmap and dendrogram.

Supervised classification results

The MetaAnalyst provides nine metrics to measure the classification performance. The 'Visualization type' drop down menu lists all of the available plots, which include:

- **a.** Misclassification Rate: The x-axis represents the number of top selected features, and the y-axis represents the value of the misclassification rate for each feature selection method.
- **b.** Sensitivity: The x-axis represents the number of top selected features, and the y-axis represents the value of the sensitivity measure for each feature selection method.
- **c.** Specificity: The x-axis represents the number of top selected features, and the y-axis represents the value of the specificity measure for each feature selection method.
- **d.** Accuracy: The x-axis represents the number of top selected features, and the y-axis represents the accuracy of each feature selection method.
- **e.** Balanced Accuracy: The x-axis represents the number of top selected features, and the y-axis represents the balanced accuracy value of each feature selection method.
- **f.** AUC: The x-axis represents the number of top selected features, and the y-axis represents the corresponding area under the curve for each feature selection method.
- **g.** ROC: The x-axis represents the false positive rate, and the y-axis represents the false positive rate. By selecting this plot, the 'Number of top selected features' drop-down

menu will appear, allowing the user to plot the ROC curves for all selected algorithms at the selected number of top features.

Consensus performance

This figure shows the agreement among different biomrker detection algorithms as a matrix of bars. Each row corresponds to a set (algorithm), and each column represents the cardinality of the intersection between the respective sets. This plot shows the suggested overlapped markers by the different algorithms included in the analysis.

(3) **Figure Options**: there are a wide variety of options provided in the software to manipulate the displayed figure. These include the following:

- Figure Height: to adjust the height of the figure.
- Figure Width: to adjust the width of the figure.
- Xlabel: the x-label of the figure.
- Ylabel: the y-label of the figure.
- Title: the title of the figure.
- Label Font Size: the font size of the x-label and y-label.
- Marker Size: the size of the markers.
- Title Font Size: the size the figure's title.
- Legend Location: the location of the legend in the figure.
- Legend Font Size: the font size of the legend.
- Line Width: the width of the line(s) in the figure.
- XGrid: to display grid lines perpendicular to the x-axis.
- YGrid: to display grid lines perpendicular to the y-axis.
- Box: to display the box outline around the axes.
- Default Check Box: enable the default options of the figure.
- Save Figure Button: save the current figure in one of the available formats (jpg, png, tif, pdf, fig, eps, bmp, emf, pcx, pbm, pgm, ppm, svg).
- Clear Figure Button: clear the current figure.
- Refresh Figure: to refresh the figure at any time.

(4) **Save the Results in a Tabular Format**: the 'Save Markers Scores as table' button enables the user to save the scores of the markers as an excel file. There is an option for the user to save the scores in a descending order. Furthermore, the 'Save Classification Scores as Table' writes the classification scores of each biomarker detection algorithm in an excel sheet.

A Step-by-Step Example

In this section, we present a step-by-step example to illustrate the workflow of MetaAnalyst. The dataset used in this example studies the relationship between the human gut microbiota and the Acute Cardiovascular Disease (ACVD). This dataset is composed of metagenomic stool samples from 214 ACVD patients and 171 healthy subjects. In addition to the disease status, the dataset provides information about gender and BMI values for the majority of samples. Fig. 17 shows the ACVD dataset. The following demonstrates the steps of analysis:

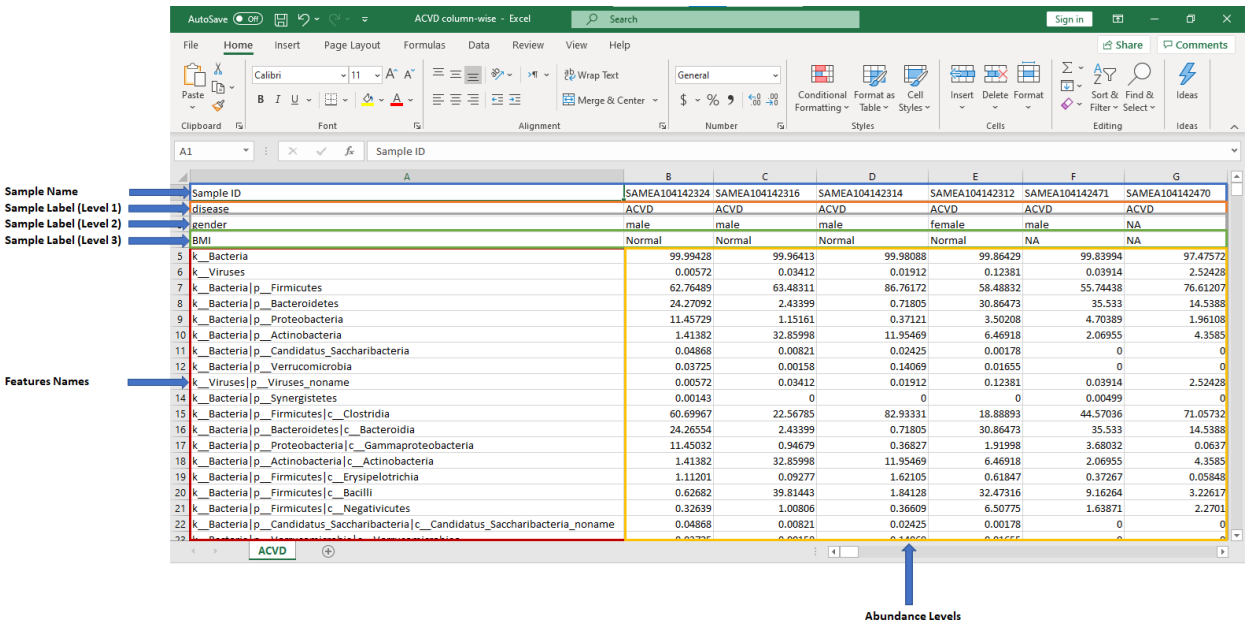


Fig.17 ACVD dataset

Step 1. Input Data Upload

1) By clicking the 'Browse' button, the user can upload the input data in the supported file formats. As can be seen in Fig. 18, we selected the 'ACVD.csv' file. Then, the software will display a progress window indicating that the software is reading the input data.

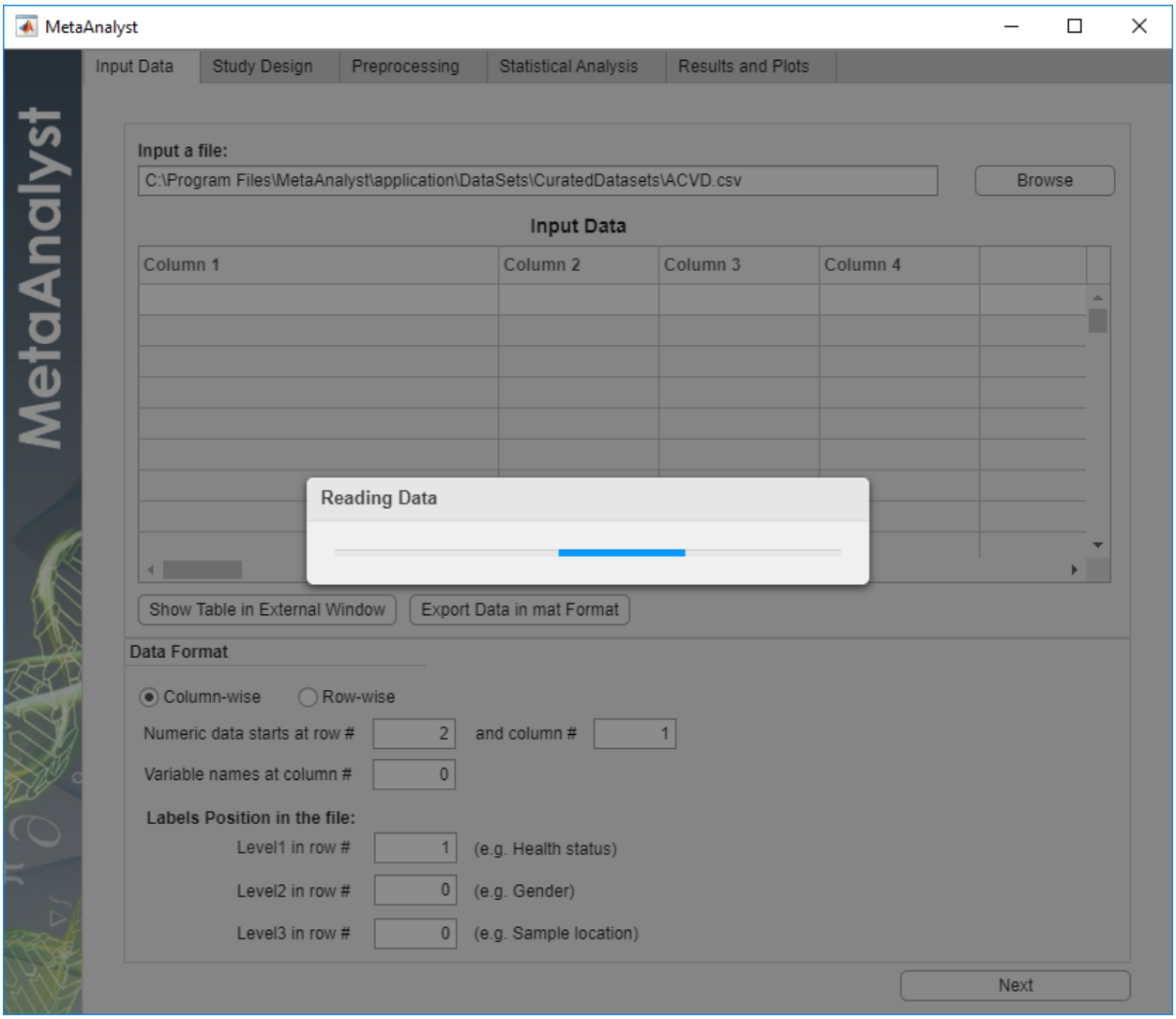


Fig.18 Reading Input Data

2) Then, the software will display the input data in the below table, as show in Fig. 19.

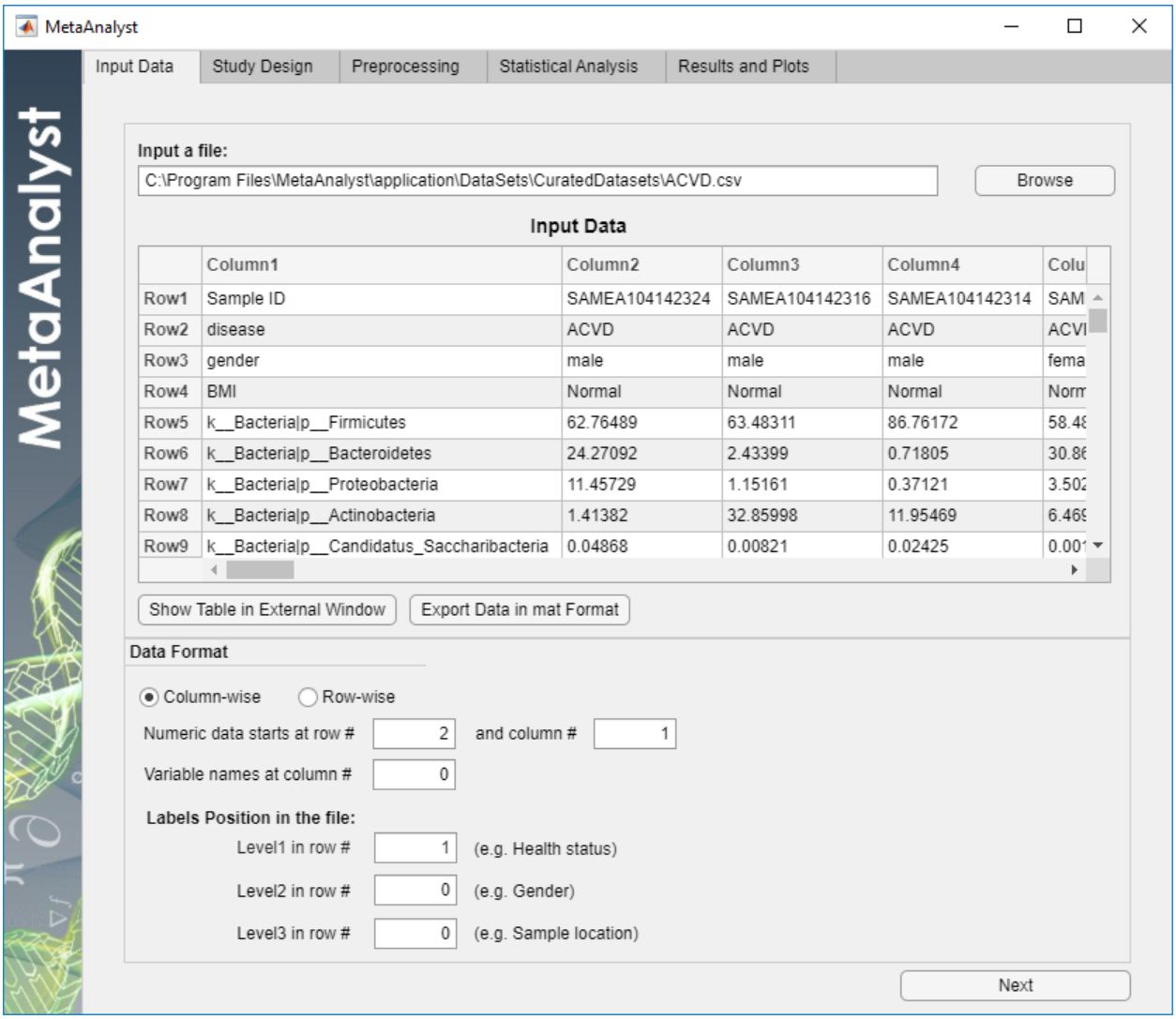


Fig. 19 Displaying Input Data

3) By looking at the table in Fig. 20, the user can now fill the parameters related to the input data. Additionally, the data is in column-wise format, the numeric data starts at row 5 and column 2, the variable names reside at column 1. The labels of the samples in this particular example are set to be the ‘disease’, ‘gender’ and ‘BMI’ rows, which reside in row 2, row 3 and row 4, respectively, as show in Fig. 20.

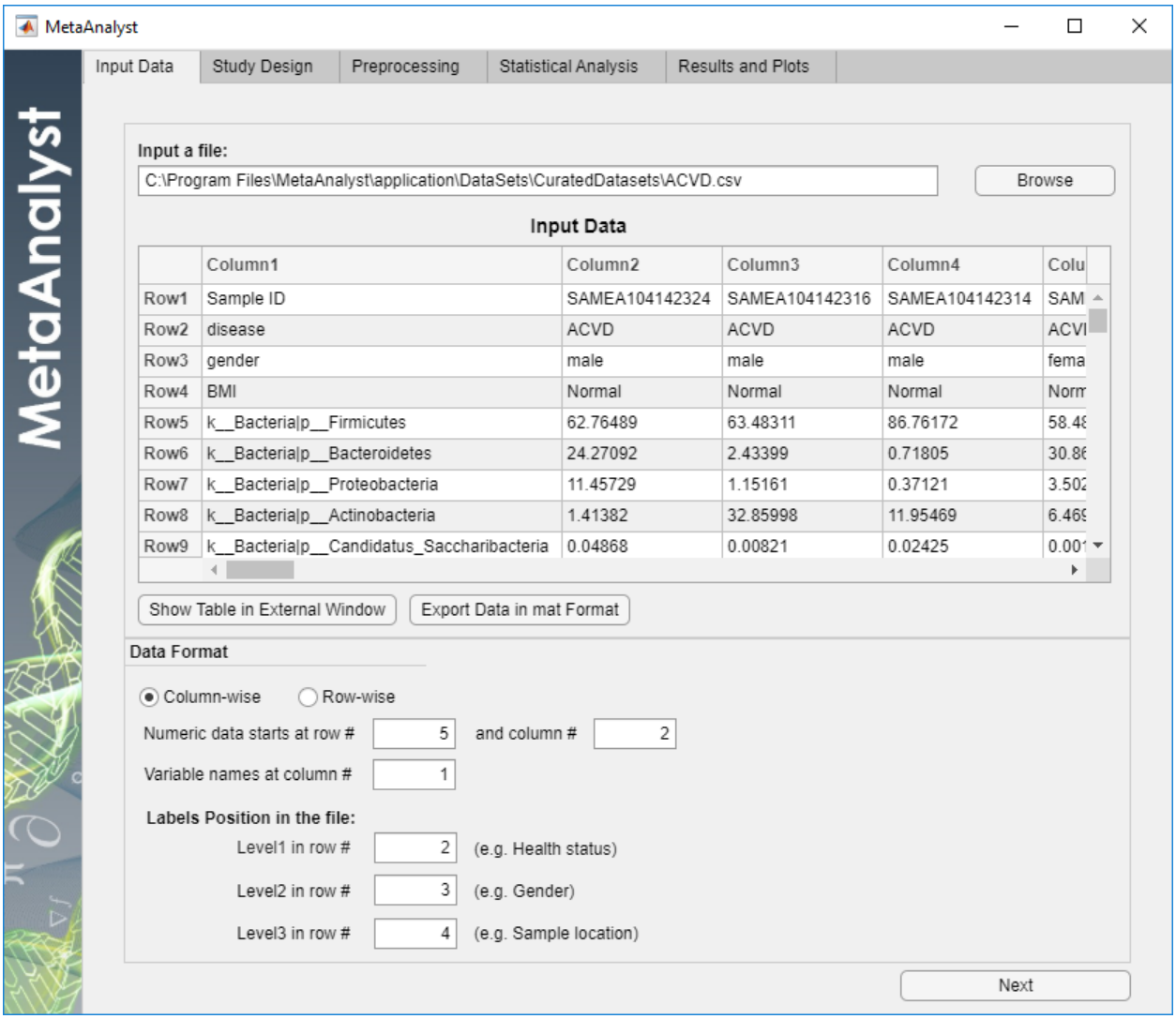


Fig.20 Inputting data parameters

4) Finally, the user should click the 'Next' button to confirm the inputted values, then the software will automatically read the values and switch to the next tab.

Step 2. Study Design

1) After confirming the data parameters, the software automatically detects the unique labels at each level. As shown in Fig. 21, the negative class representation is set to 'healthy' and 'female' and 'Normal', and the positive class representation is set to 'ACVD' and 'female' and 'Normal'.

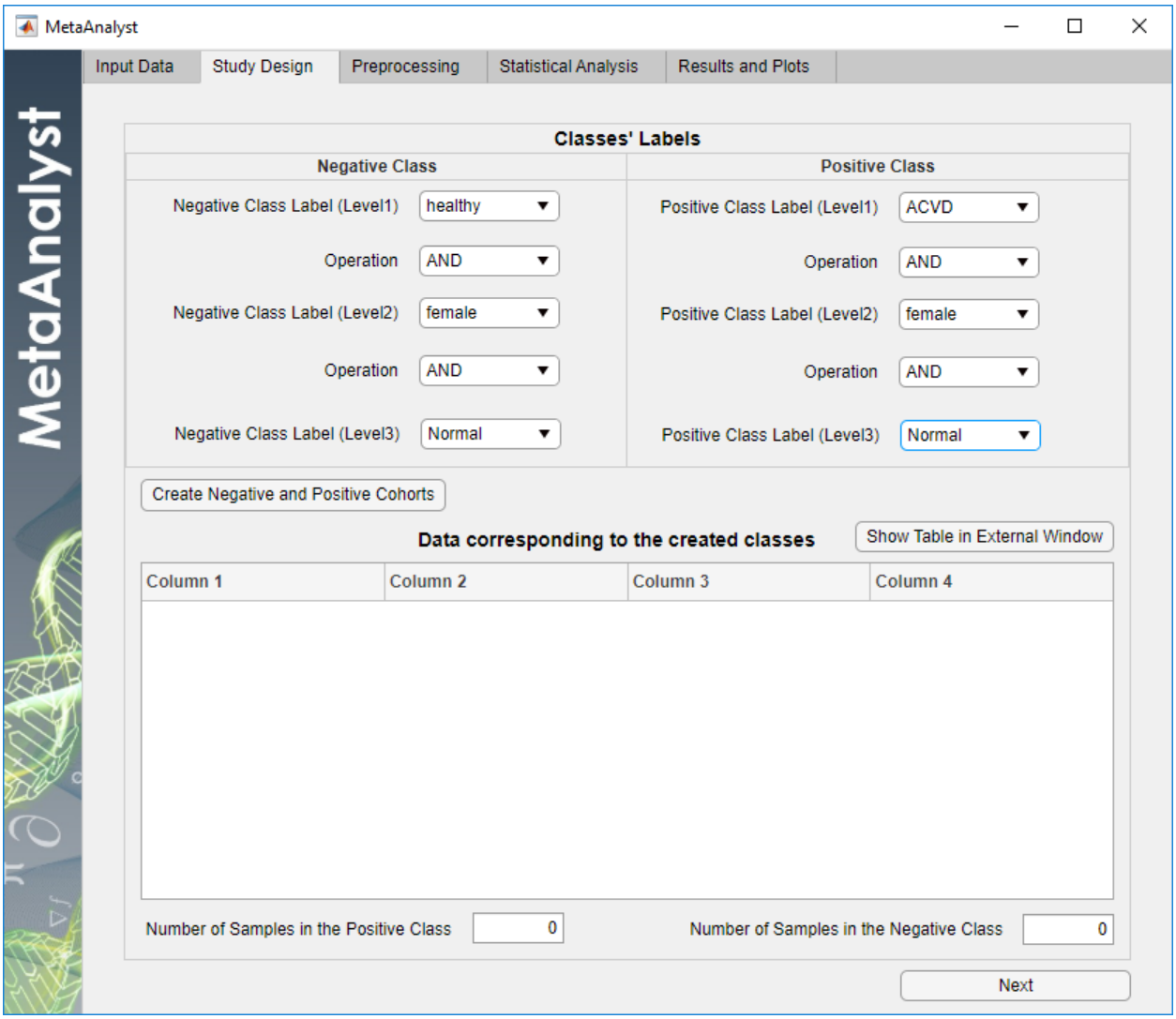


Fig.21 Selecting negative and positive classes

2) By clicking the 'Create Negative and Positive Classes' button, the software begins to assign a label to each sample based on the selected representation of the classes (Fig. 22).

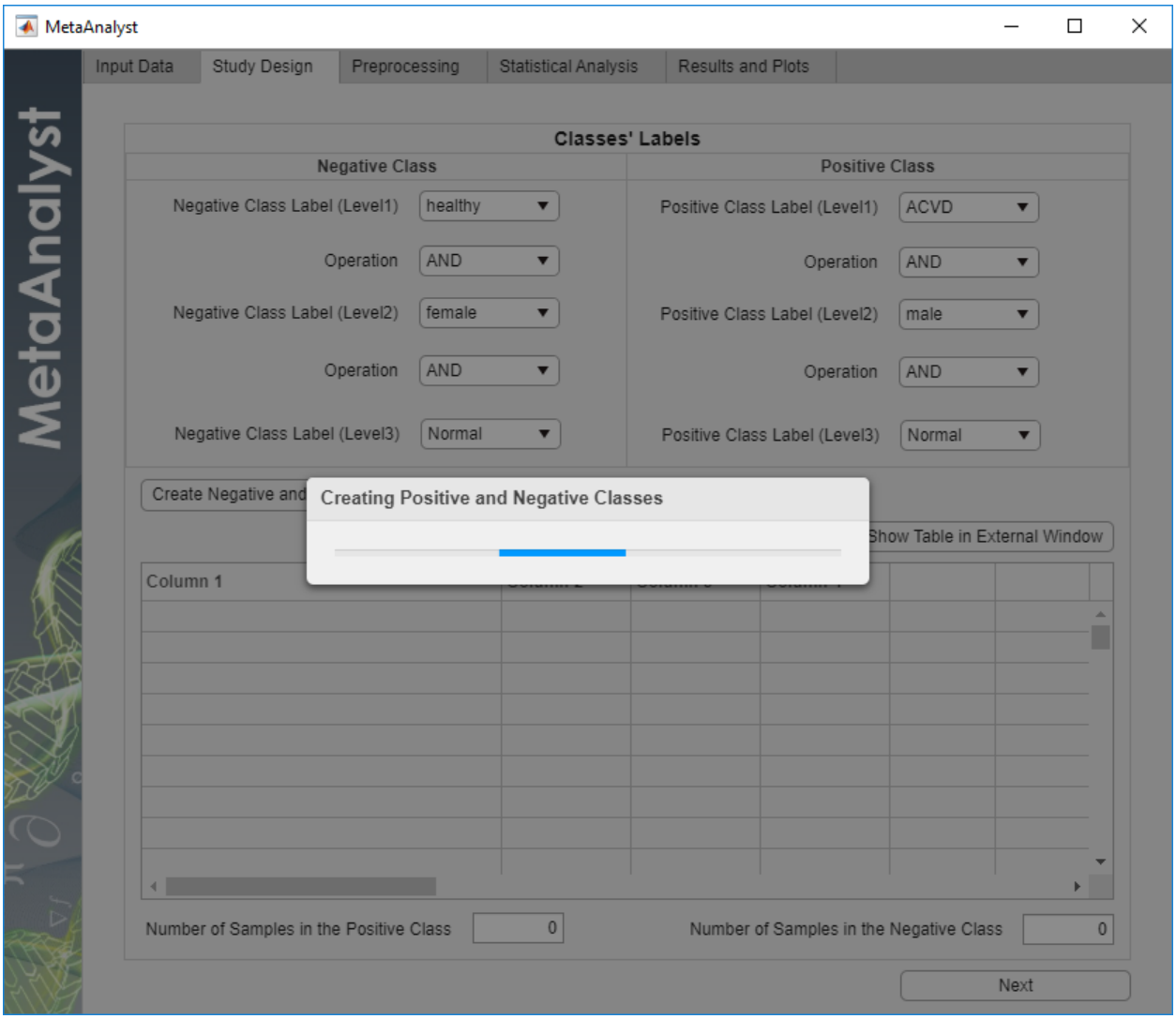


Fig.22 Creating negative and positive classes

3) Finally, the software displays the data corresponding to the created classes, as well as the labels of the samples in the table, as shown in Fig. 23.

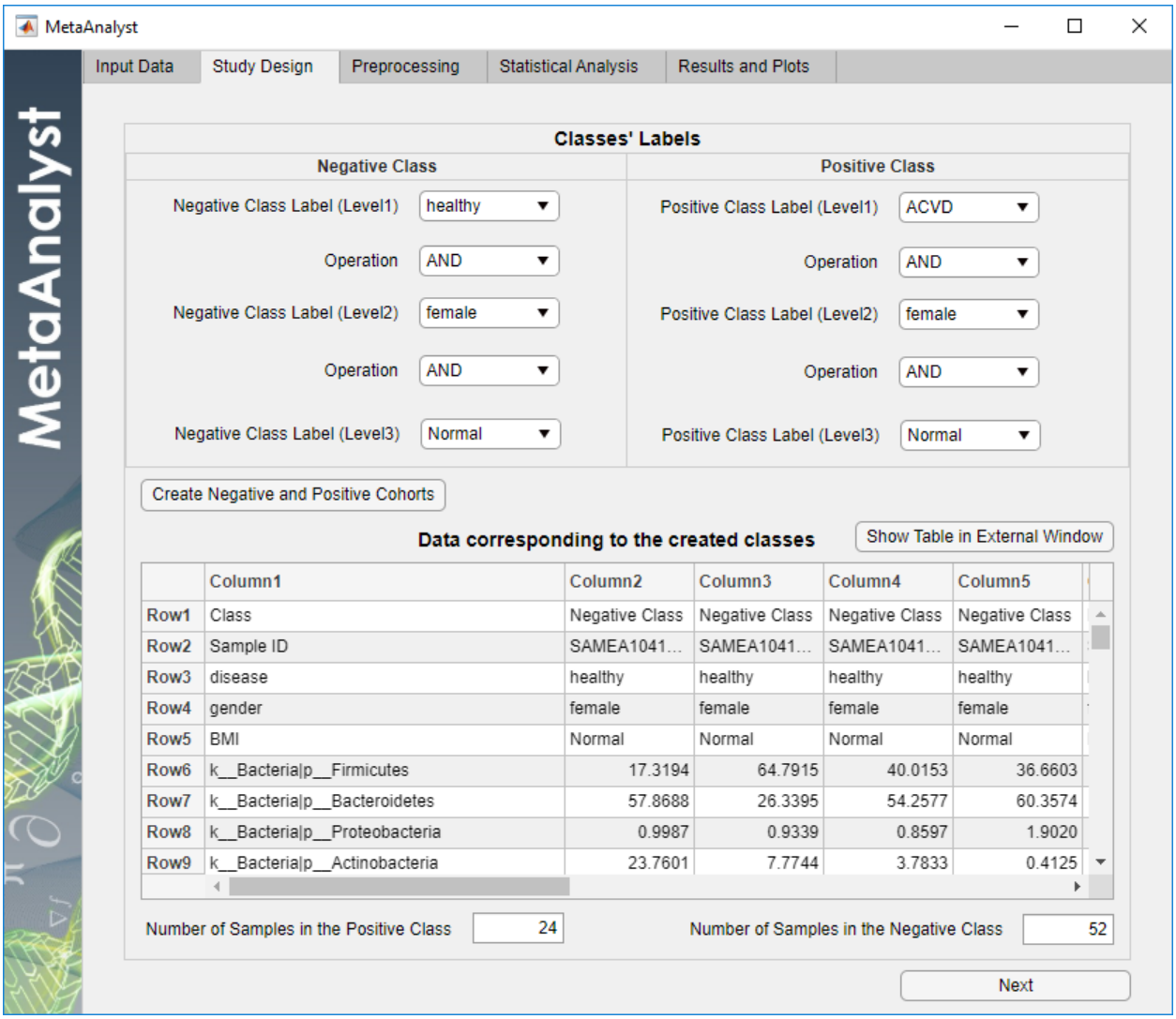


Fig.23 Data corresponding to the created classes

Step 3. Preprocessing

1) MetaAnalyst provides three themes of preprocessing procedures. Fig. 24 shows the selected preprocessing procedures in this example. Furthermore, the 'Before Preprocessing' table shows the descriptive summary statistics of the input data.

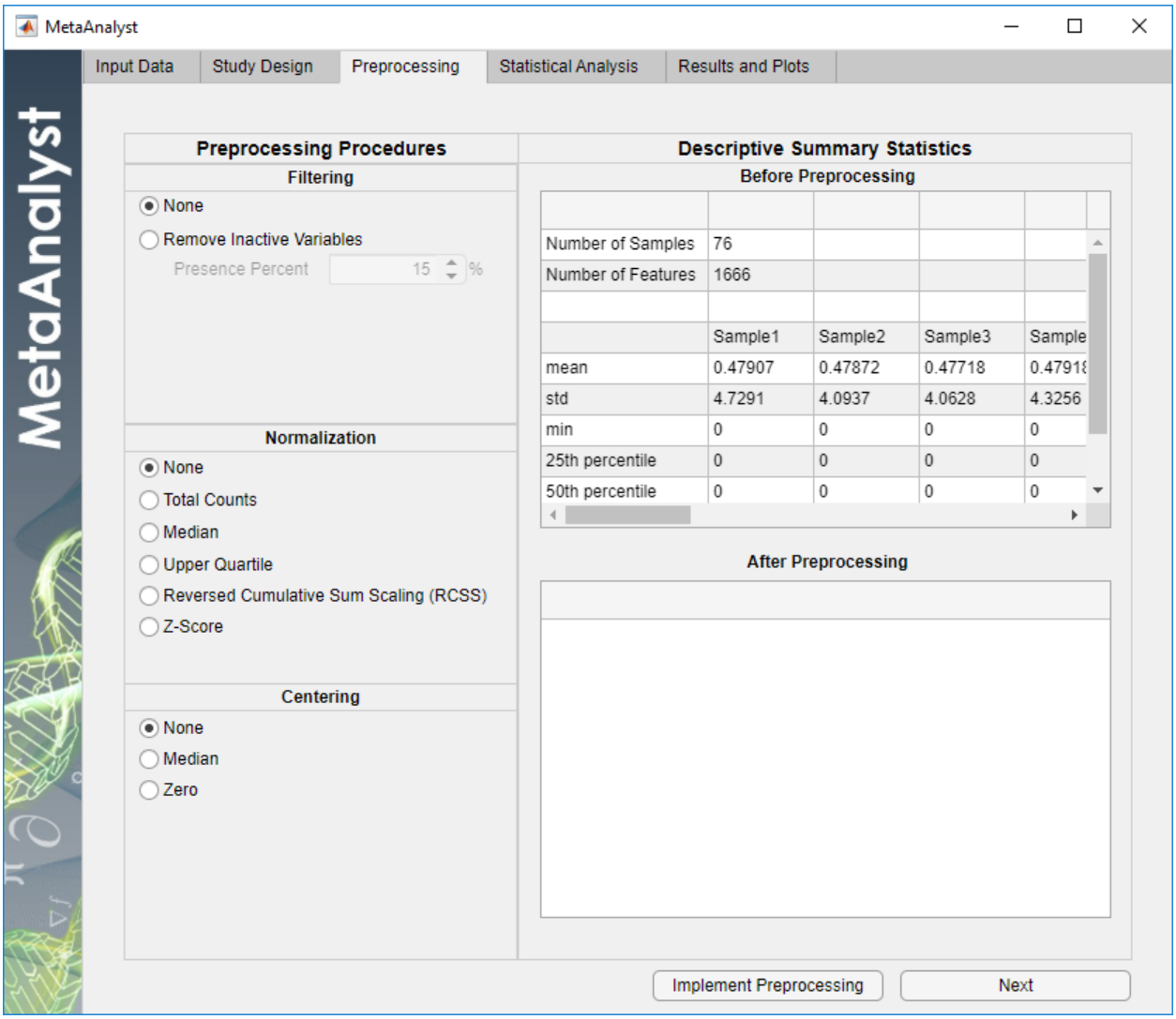


Fig.24 Preprocessing procedures

2) By clicking the 'Implement Preprocessing' button, the selected preprocessing procedures are applied on the input data. Furthermore, the software updates the 'After Preprocessing' table to display the descriptive summary statistics of the processed data, as shown in Fig. 25.

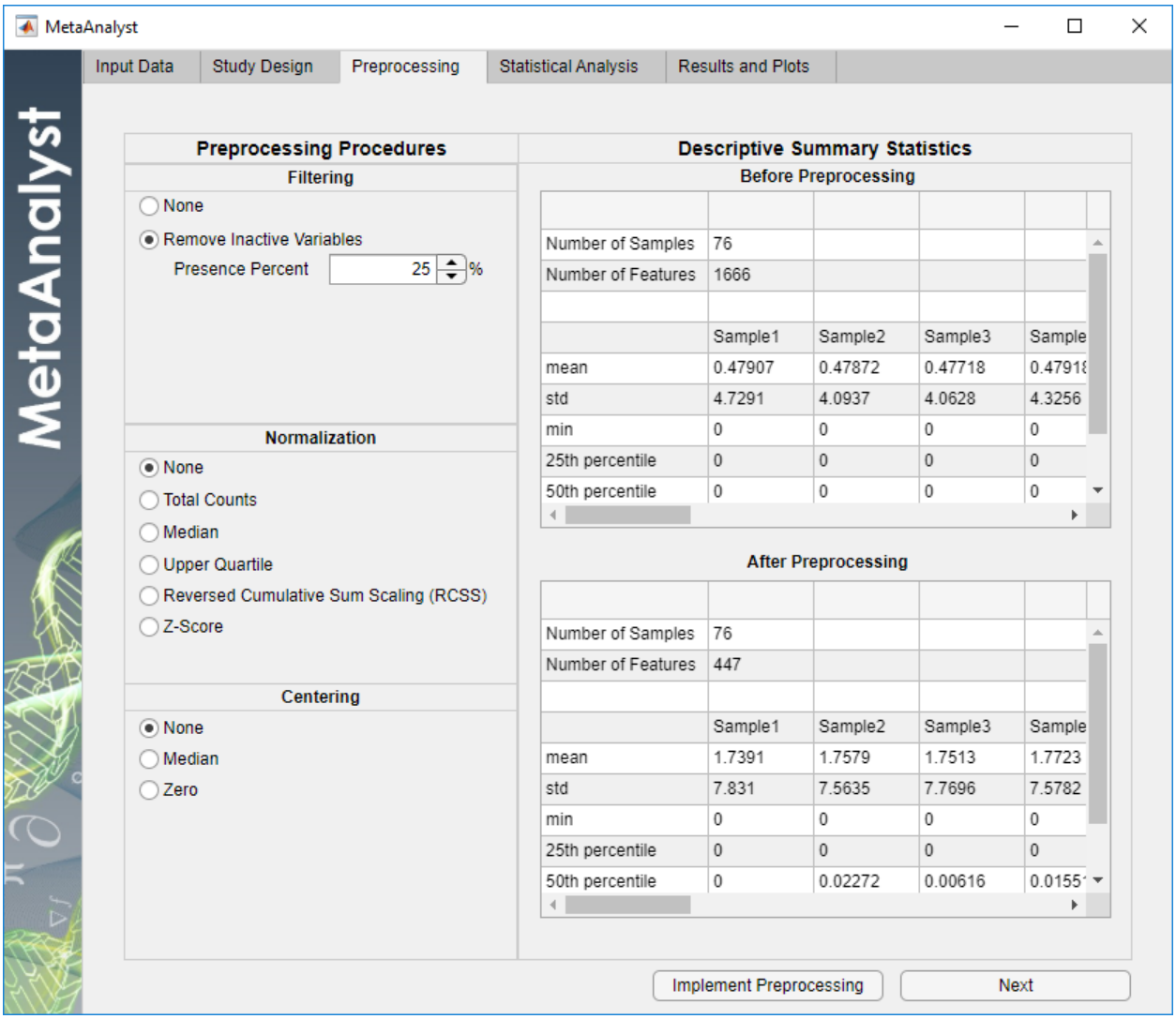


Fig.25 After implementing preprocessing

3) Finally, by clicking the 'Next' button, the software switches to the next tab.

Step 4. Statistical Analysis

1) In total, MetaAnalyst provides 28 biomarker detection algorithms (Fig. 26). By clicking the 'Add' button, the software adds the selected algorithm to the 'Selected Algorithms' box. To remove the selected algorithm from the box, the user should click the 'Remove' button. As shown in Fig. 27, we selected three algorithms, which are LEfSe, MicrobiomeDDA and Chi-square test. Furthermore, the number of top selected markers to study is set to 10, 20, 30, 40 and 50.

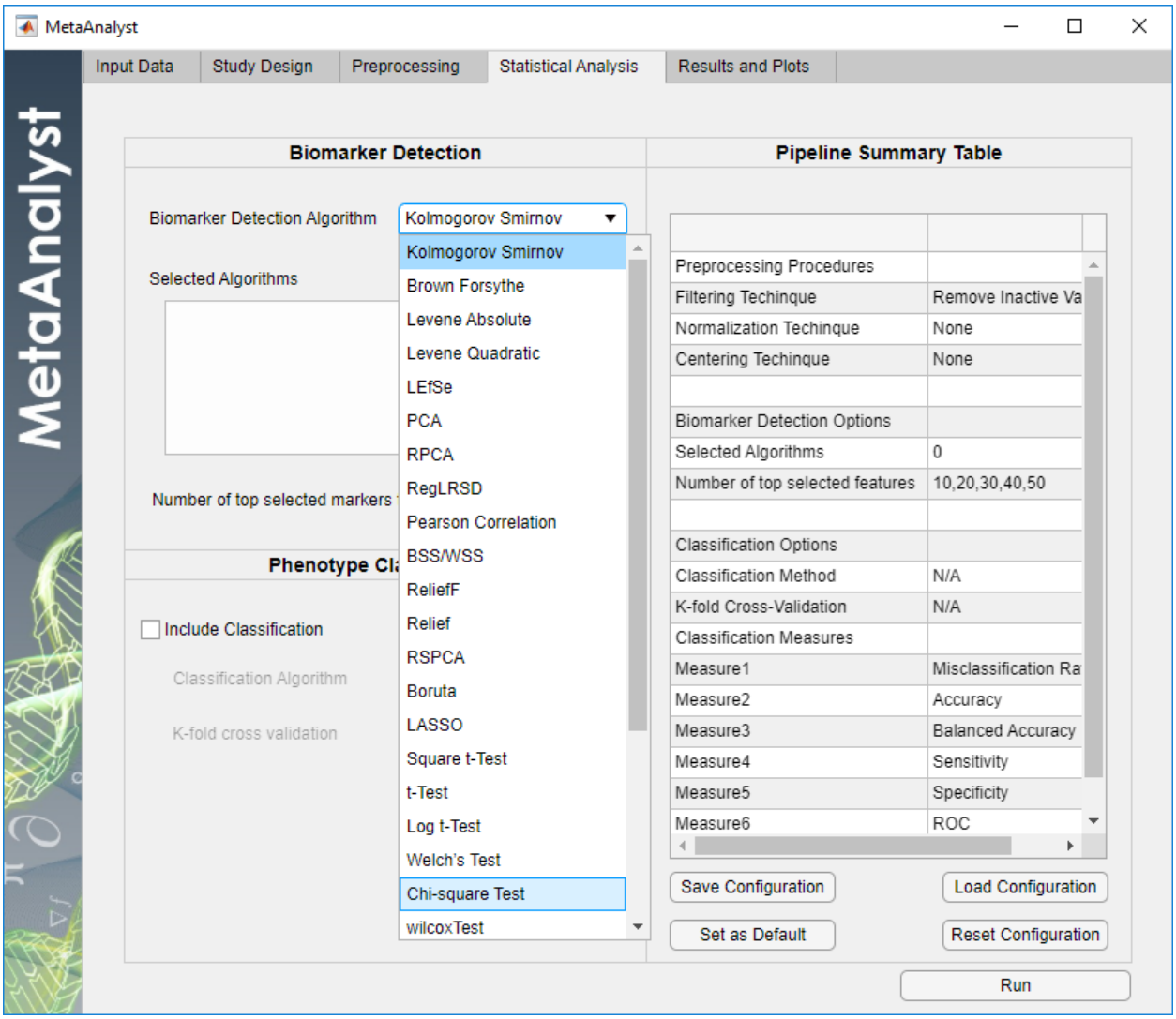


Fig.26 Biomarker detection algorithms

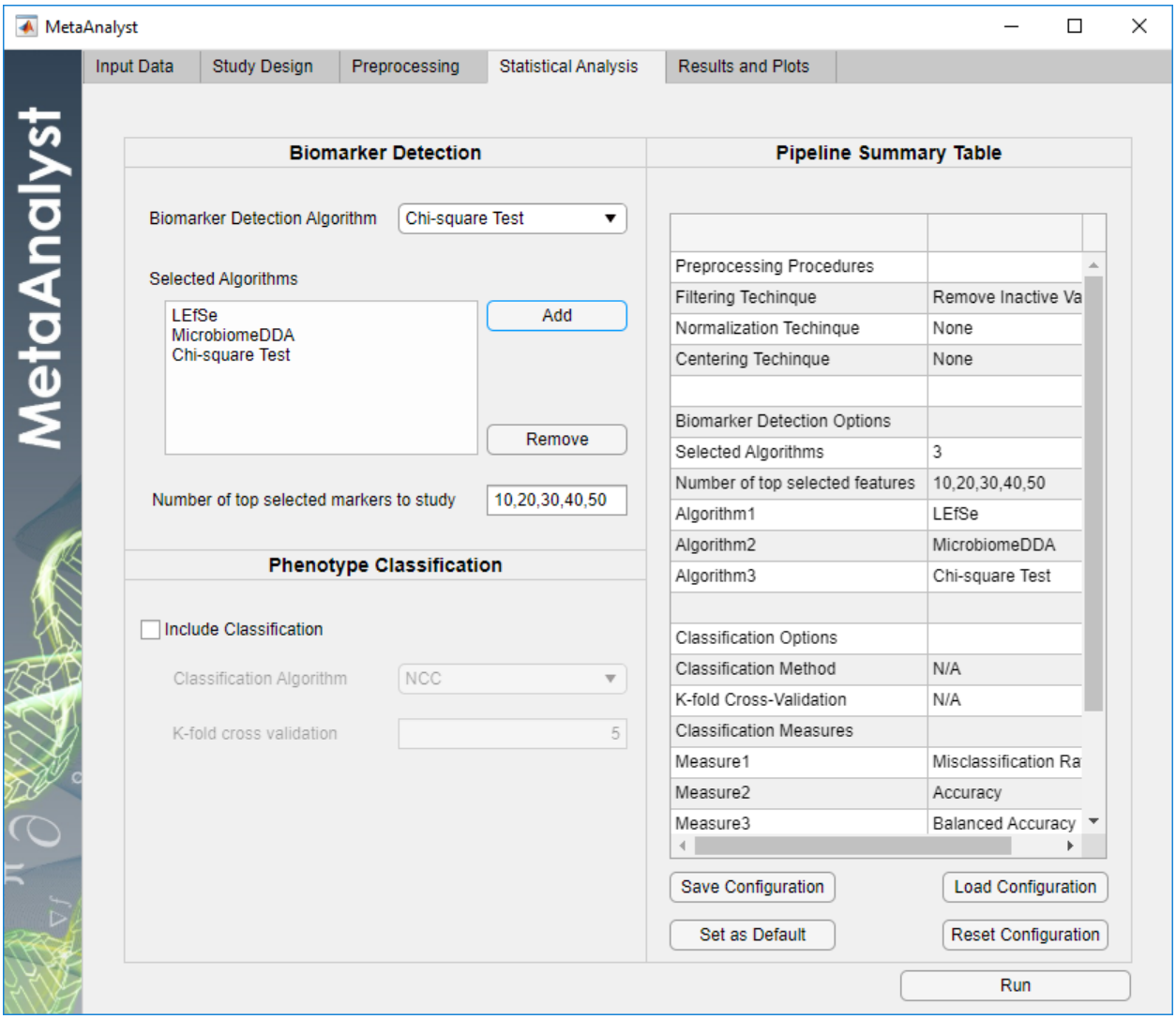


Fig.27 Selected Algorithms

2) To build a classification model, the user should enable the 'Include Classification' check box. As shown in Fig. 28, we selected the NCC classifier, and we set the K-fold cross validation value to 5.

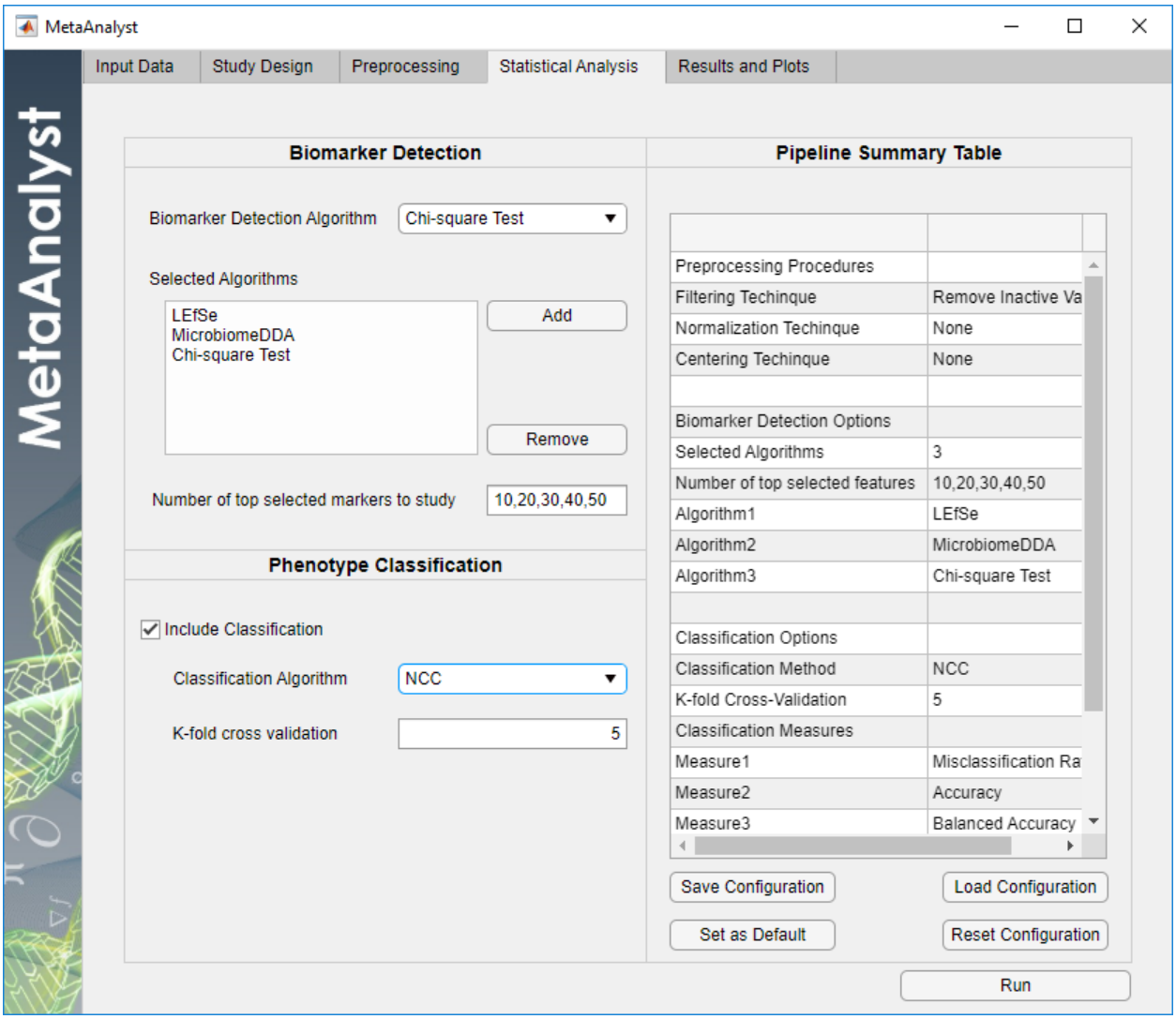


Fig.28 Pipeline analysis

3) After building the pipeline analysis, the user should click the 'Run' button to implement the selected analysis. Then, a progress window will appear indicating that the analysis is under implementation, as shown in Fig. 29. Finally, the software will notify the user once the implementation has been completed (Fig. 30).

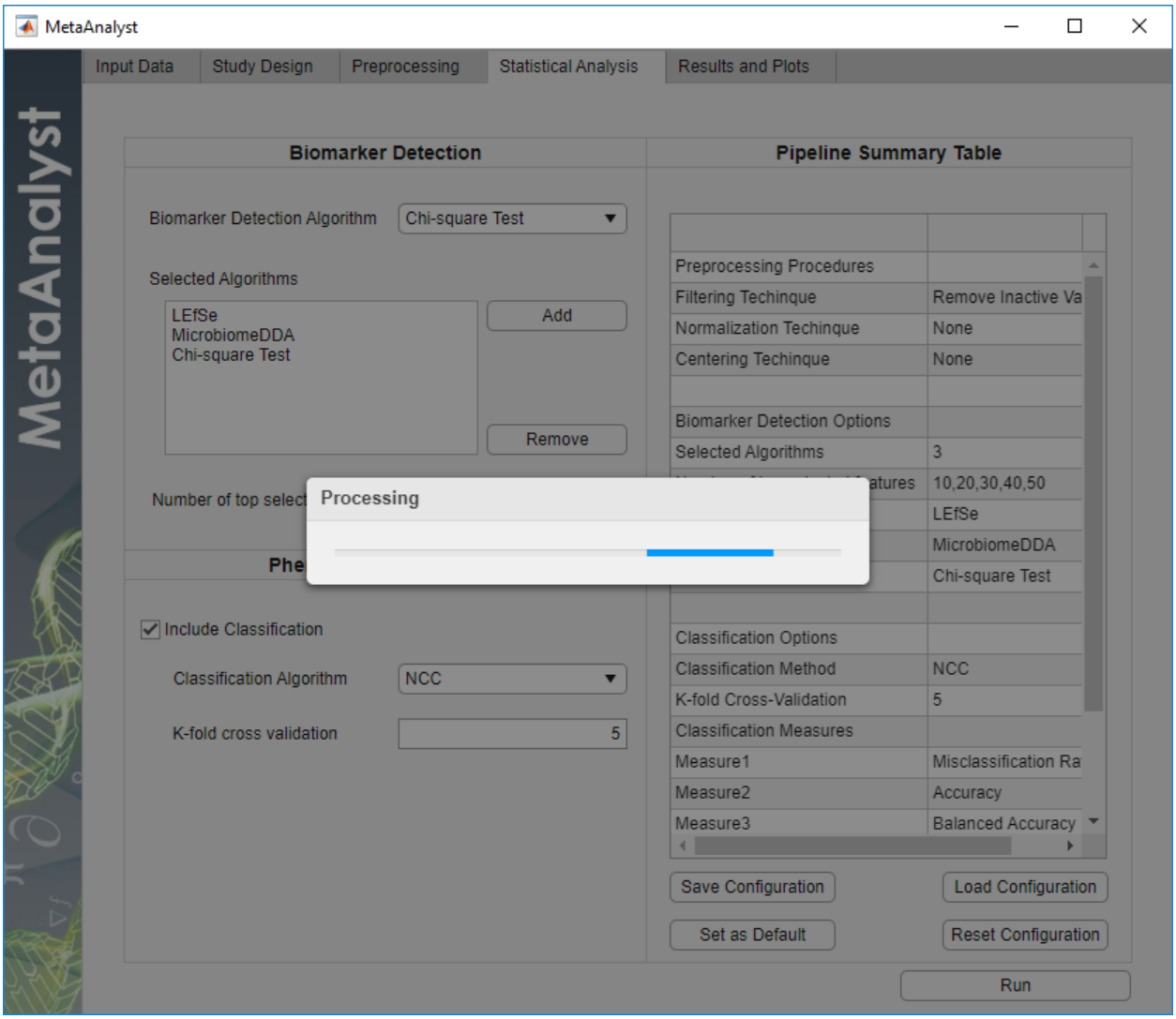


Fig.29 Implementing the pipeline analysis

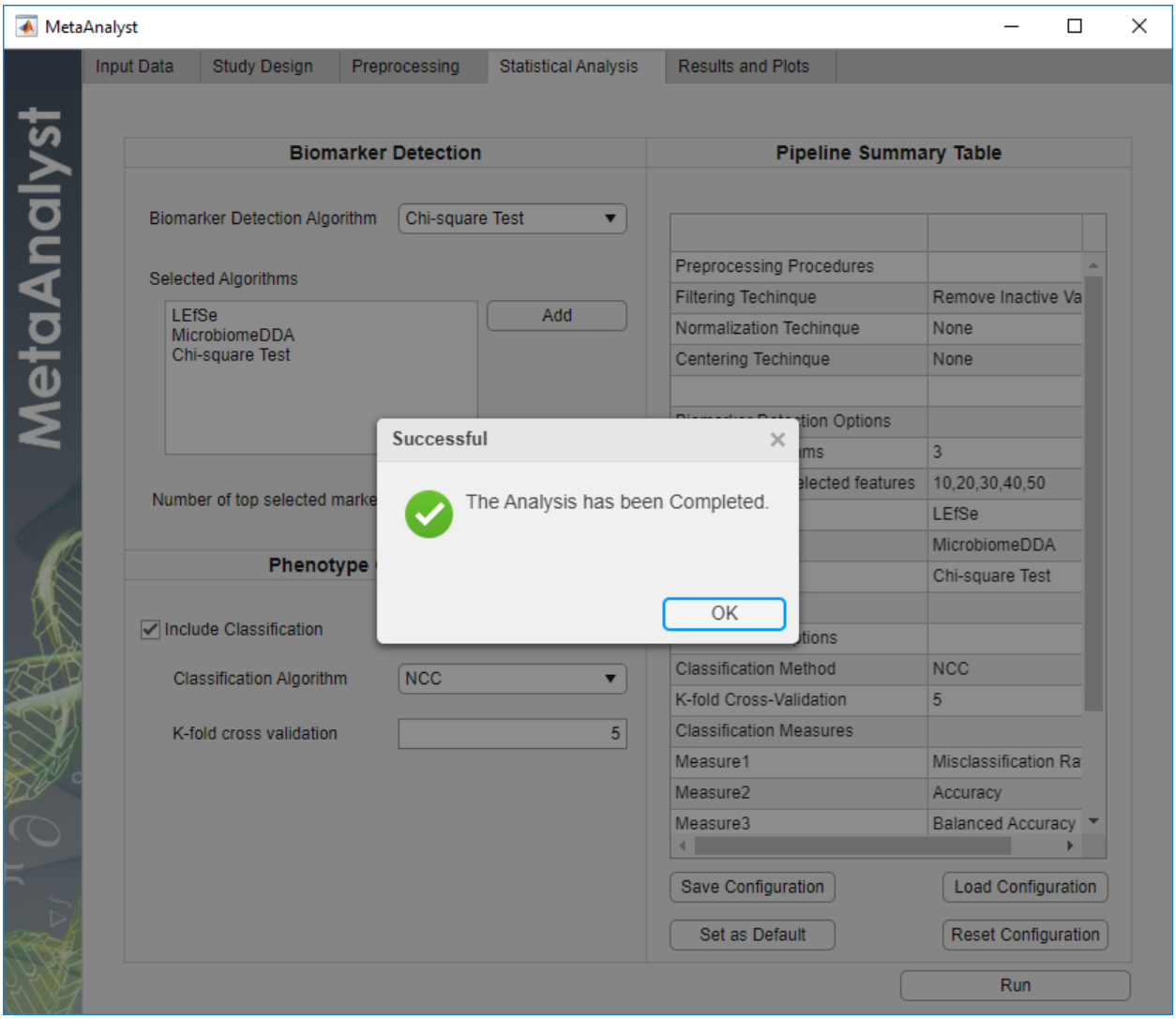


Fig.30 Execution completed

As the user selects from the available options, the software will update the pipeline summary table. To allow reusability of the configuration settings, there are four available options for the user. Firstly, the user can save the current configuration into a new file, by pressing on the 'Save Configuration' button (Fig. 31), and load this file whenever needed by pressing on the 'Load Configuration' button (Fig. 32). Furthermore, the 'Set as Default' button allows the user to set the current configurations as the default configurations (Fig. 33). Then, whenever the user opens the application again, the software will automatically load the selected default configurations. Finally, the 'Reset Configuration' button empties the pipeline analysis settings and restore the default configurations of the software, as shown in Fig. 34.

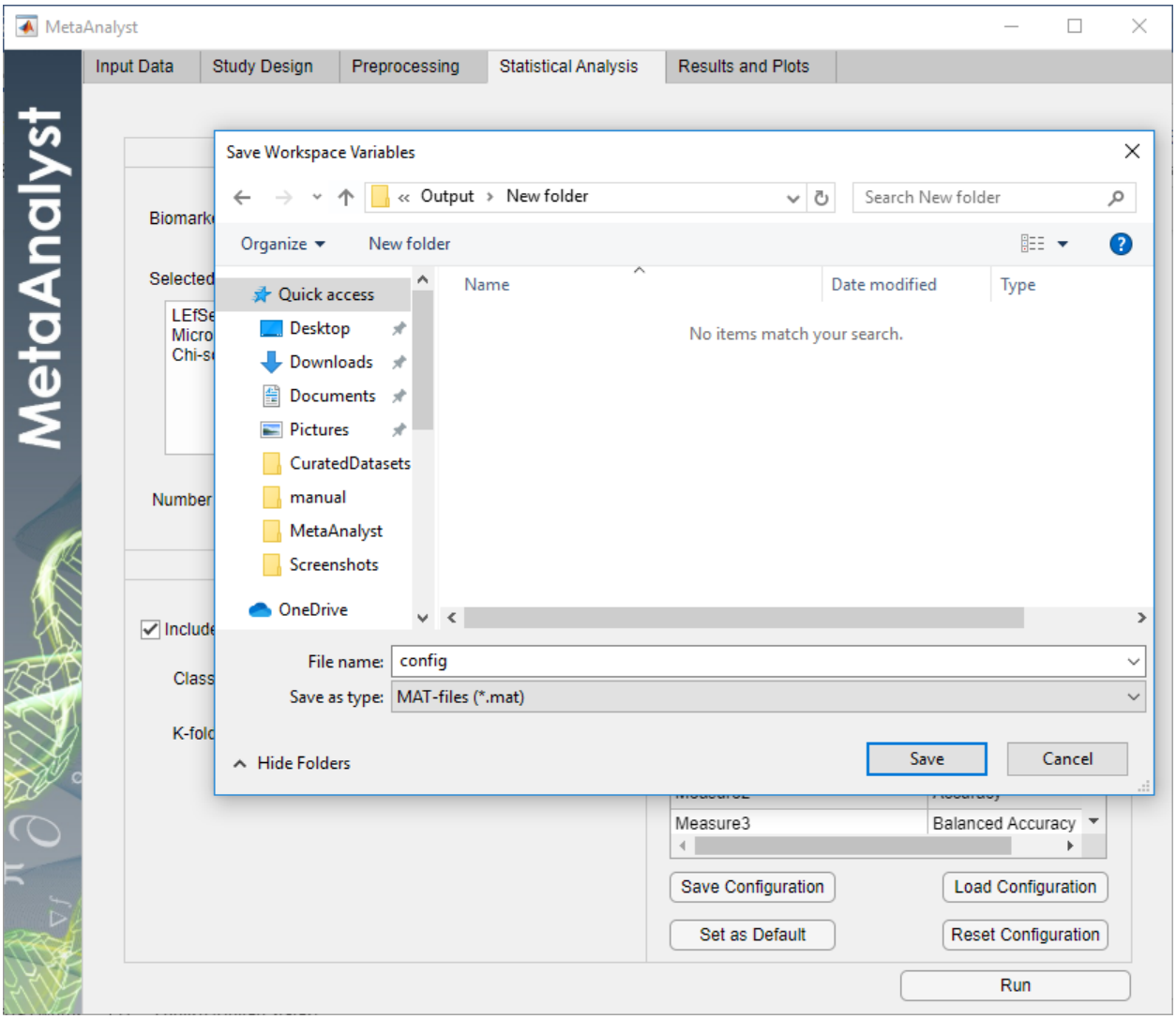


Fig.31 Saving current configuration

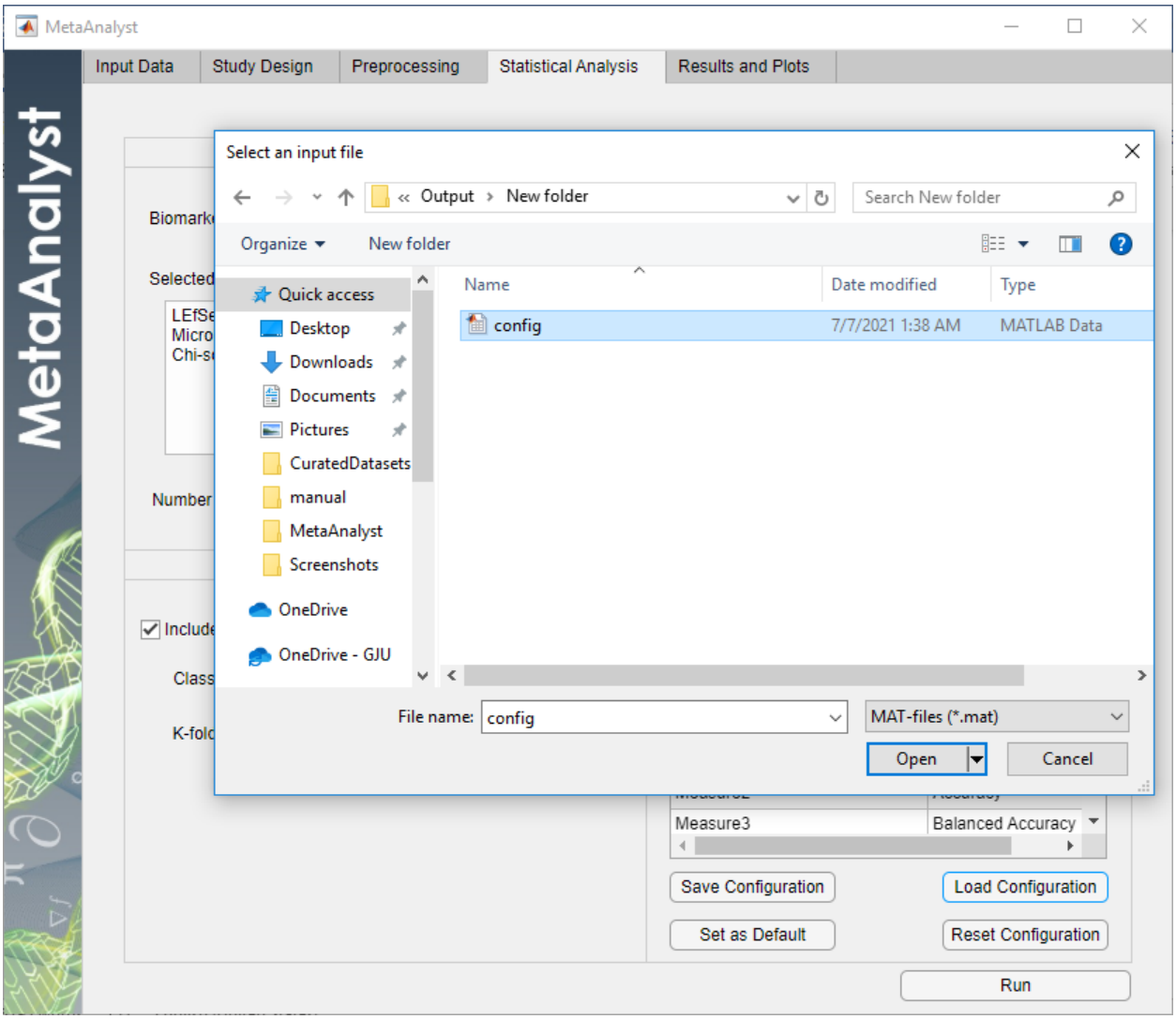


Fig.32 Loading custom configuration

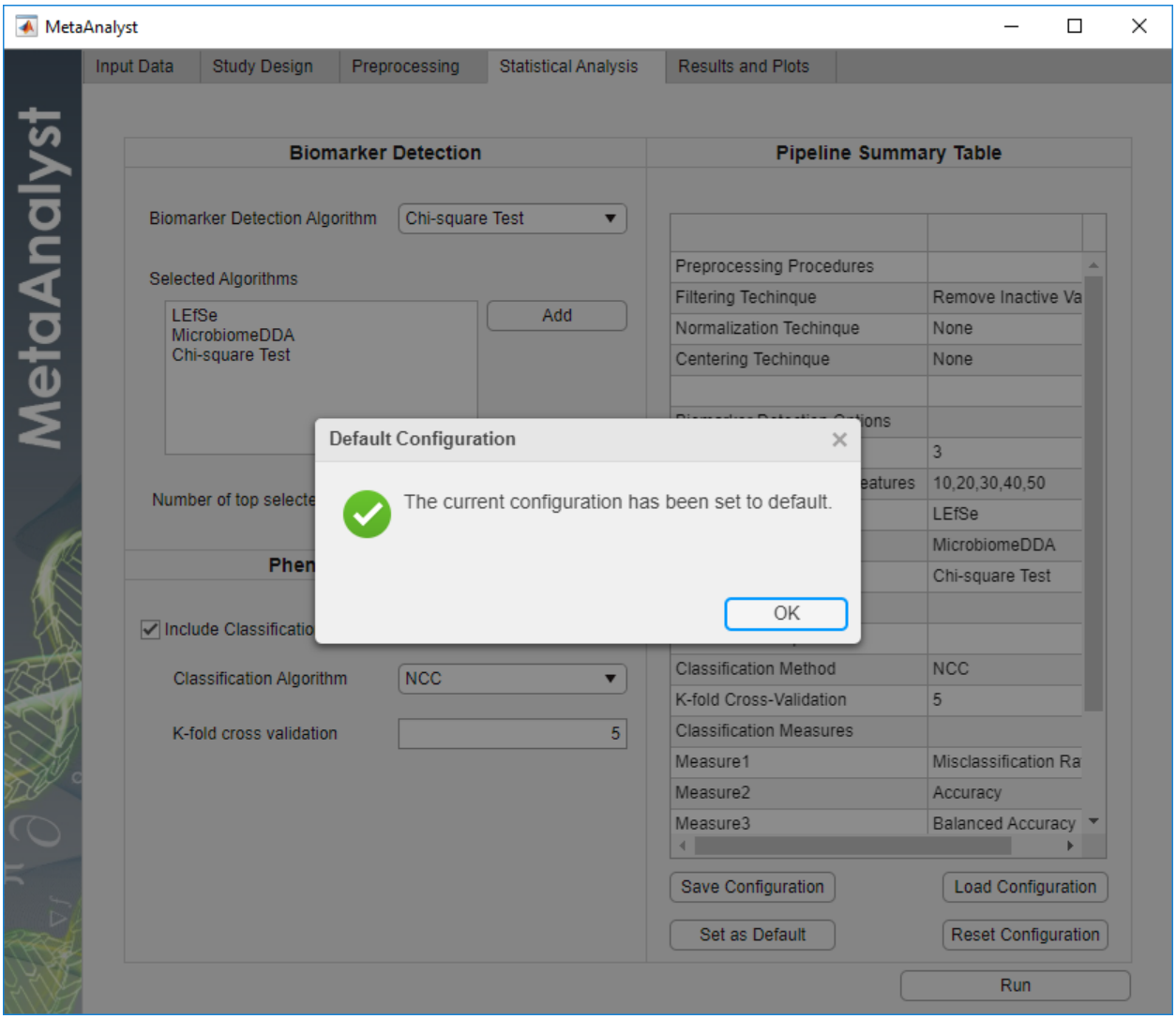


Fig.33 Saving the current configuration as default configuration

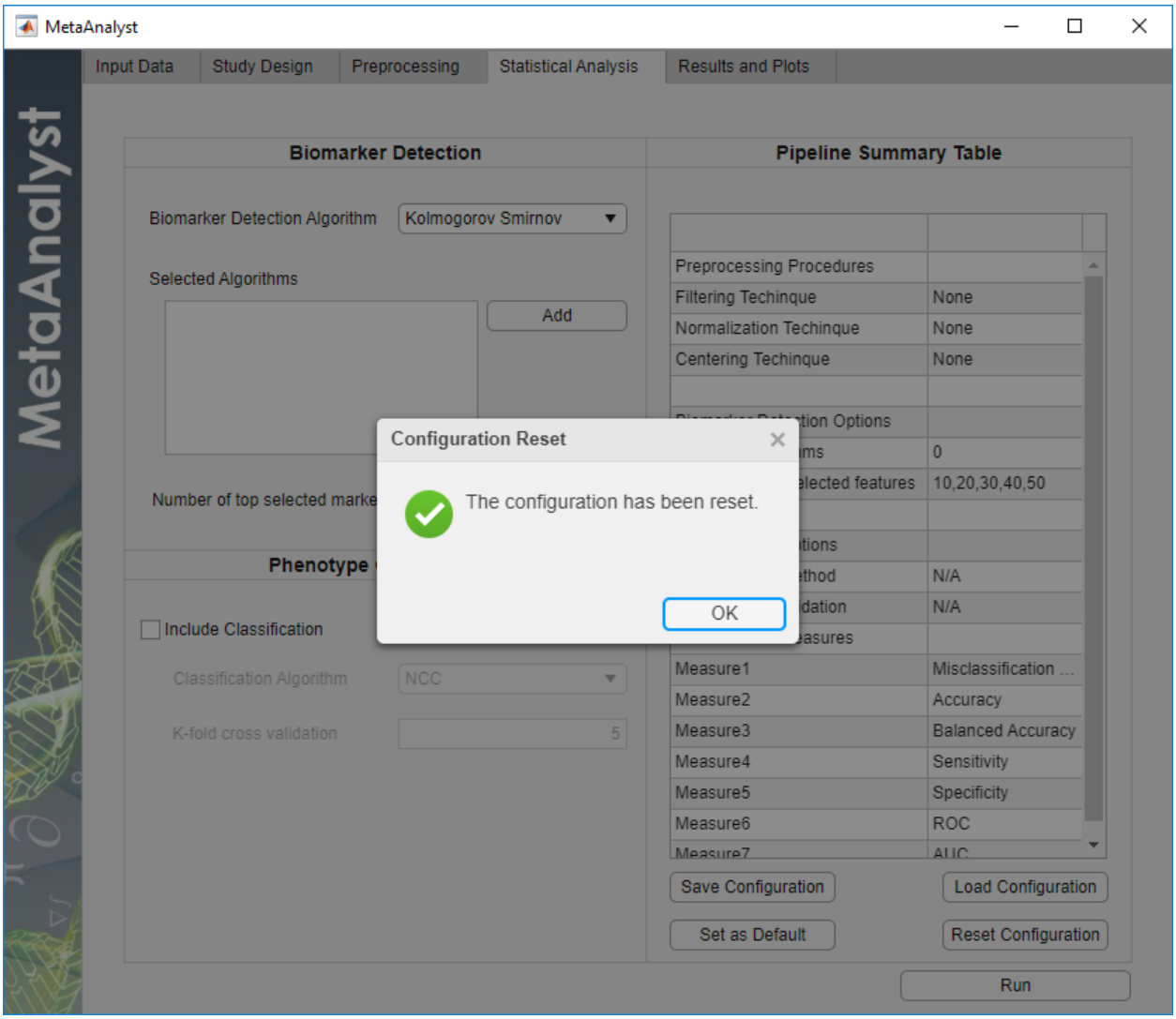


Fig.34 Resetting configuration

Step 5. Results and Plots

1) By default, the software considers the “Output” folder in the installation directory as the output path. However, the user can change the output path by pressing on the ‘Browse’ button, as shown in Fig. 35. The user should specify the output folder before running the analysis, changing the output folder after running the pipeline analysis may affect the results.

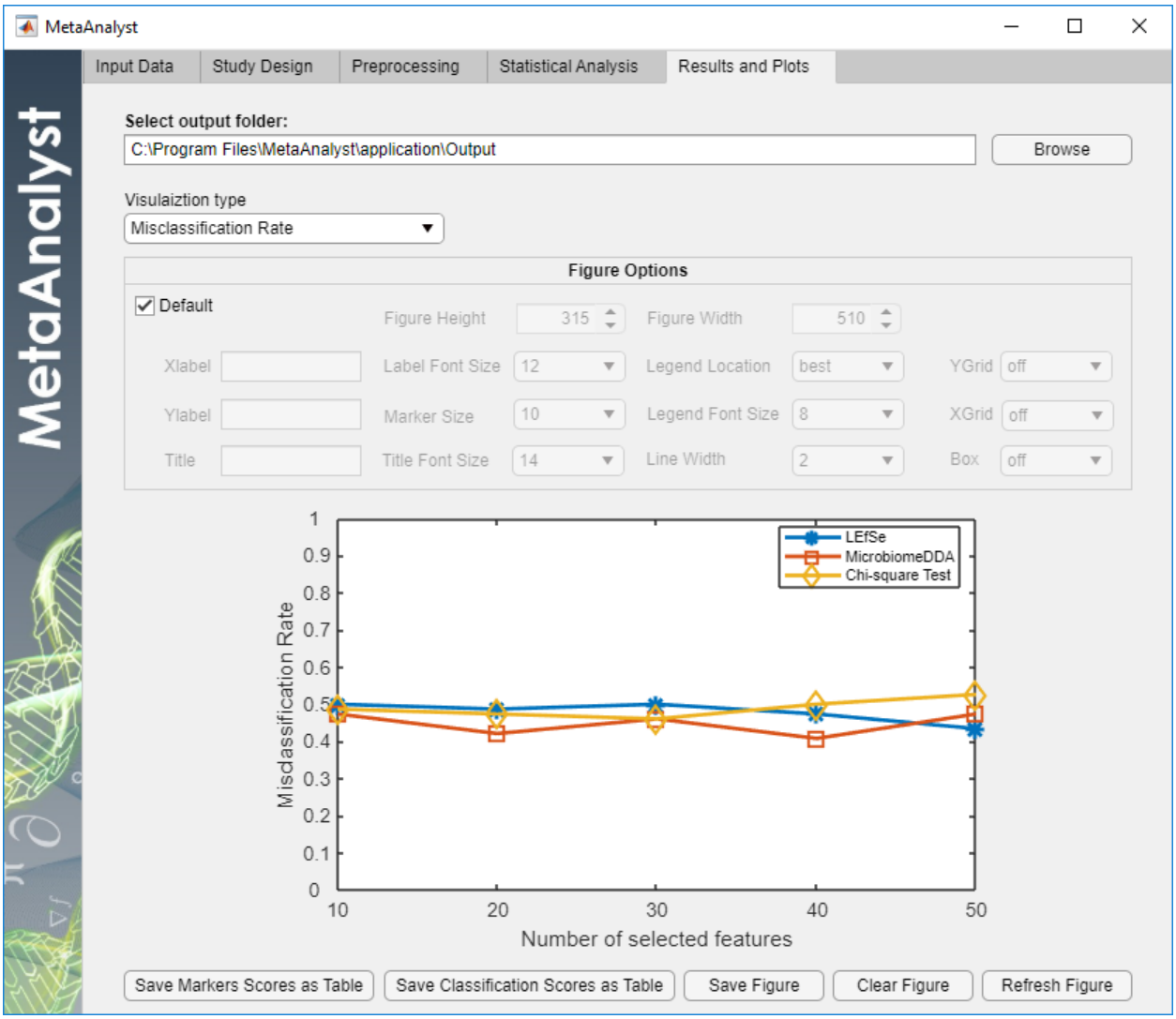


Fig.35 Selecting output folder

2) MetaAnalyst provides nine different types of interactive plots as described in the Results and Plots section, all listed in the 'Visualization type' menu (Fig. 36). As the user selects from the menu, the software will automatically display the plot in the figure below. Fig. 37 – Fig. 49 show the results of this example.

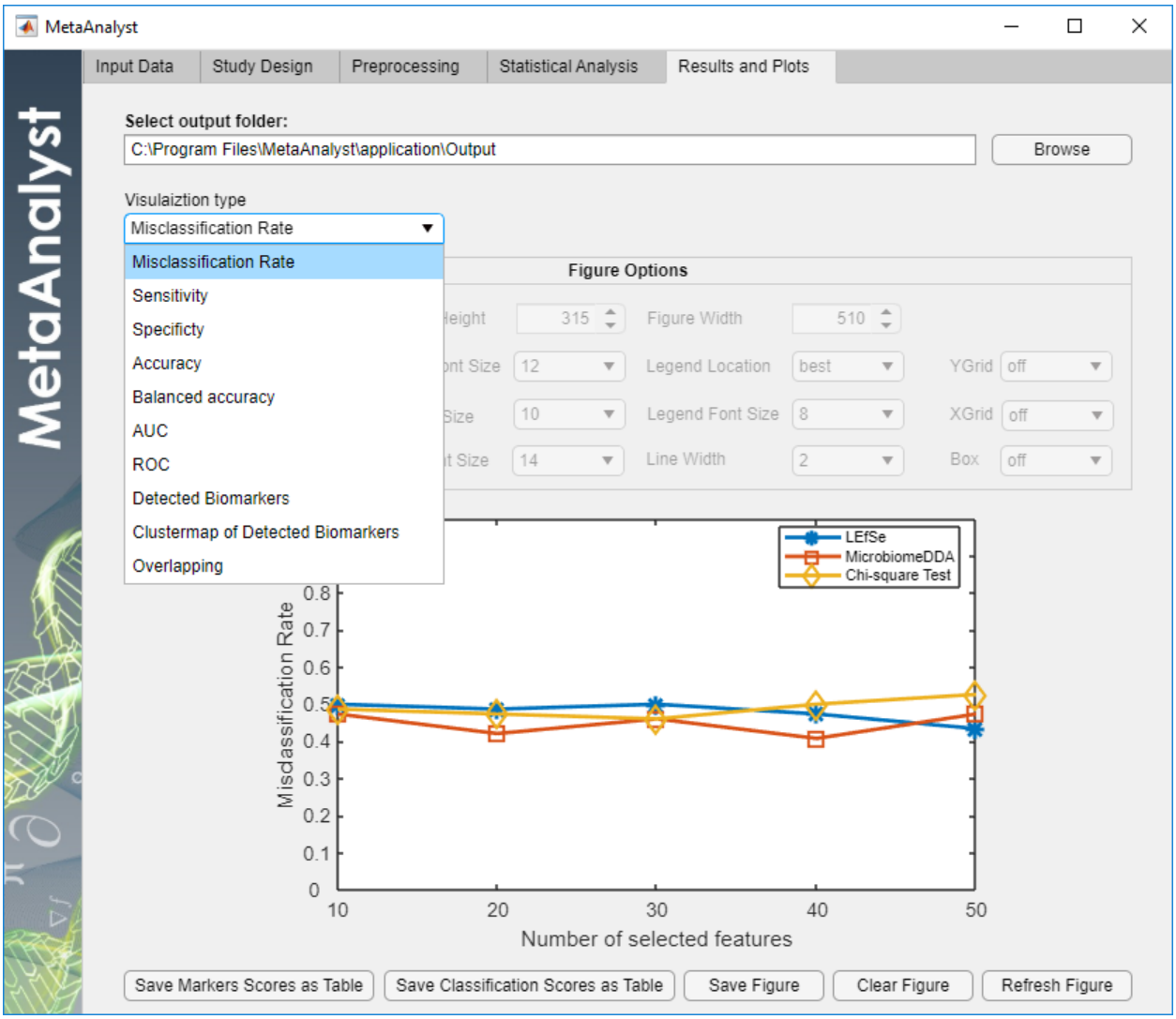


Fig.36 Available plots

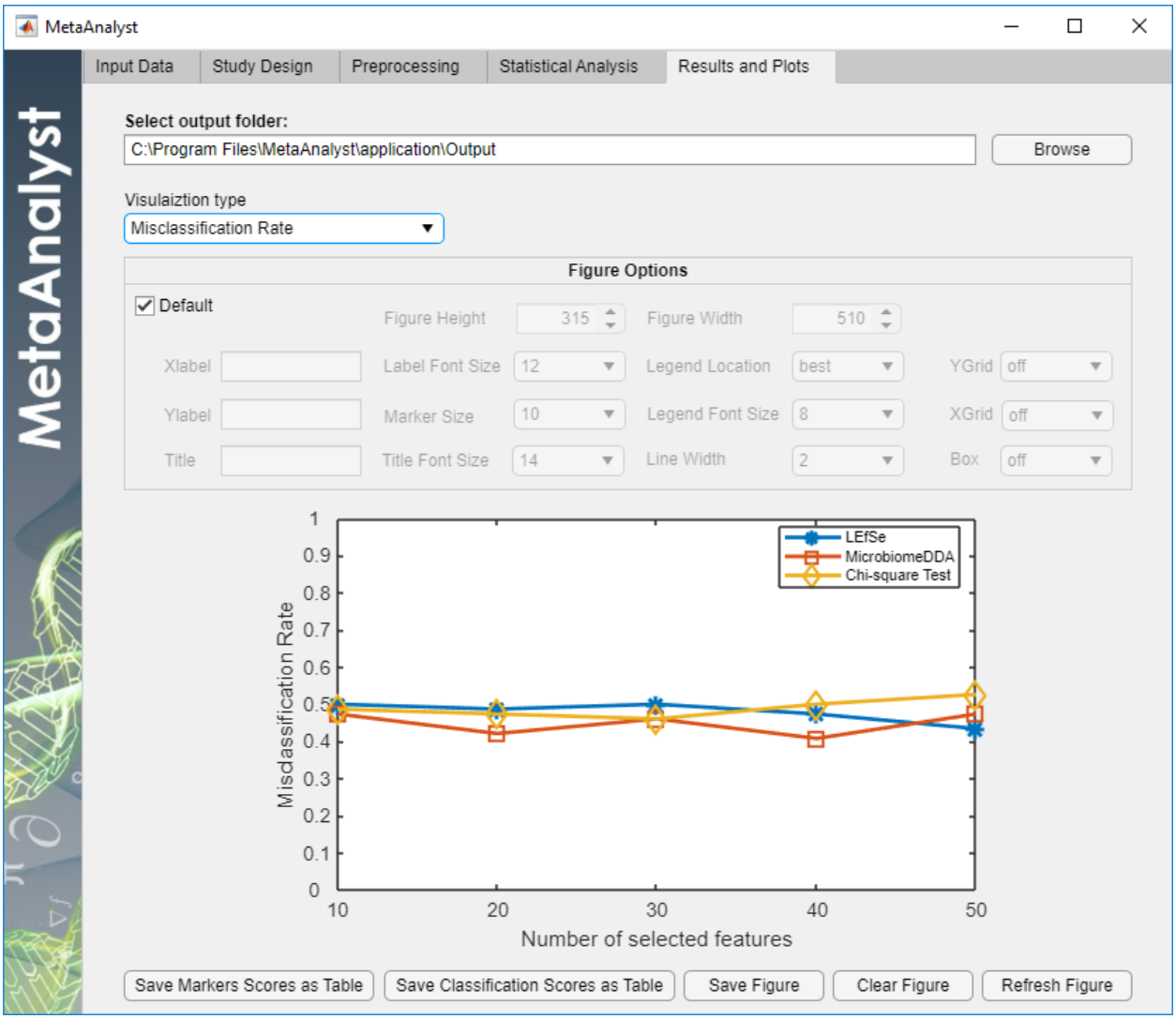


Fig.37 Misclassification rate

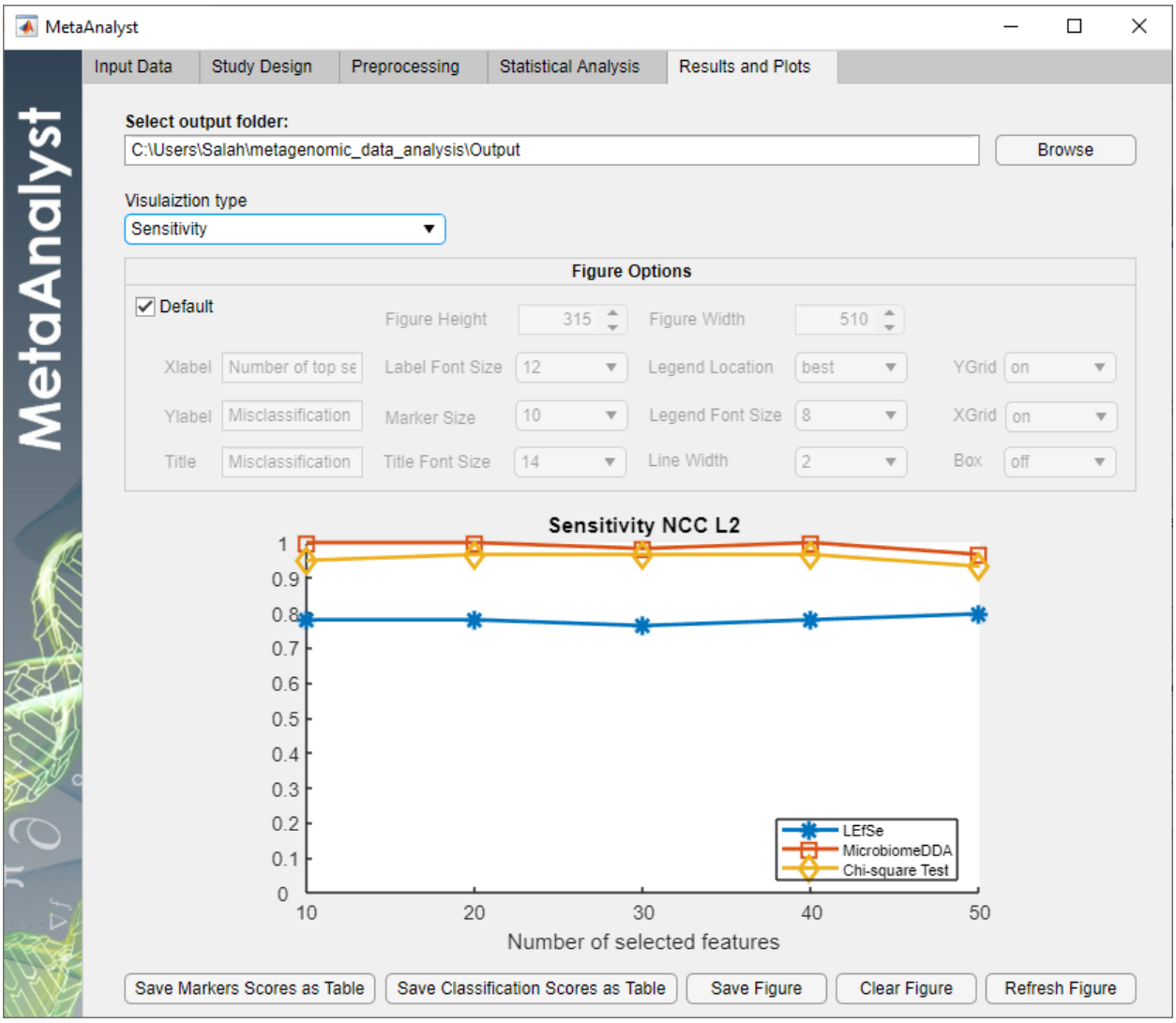


Fig.38 Sensitivity rate

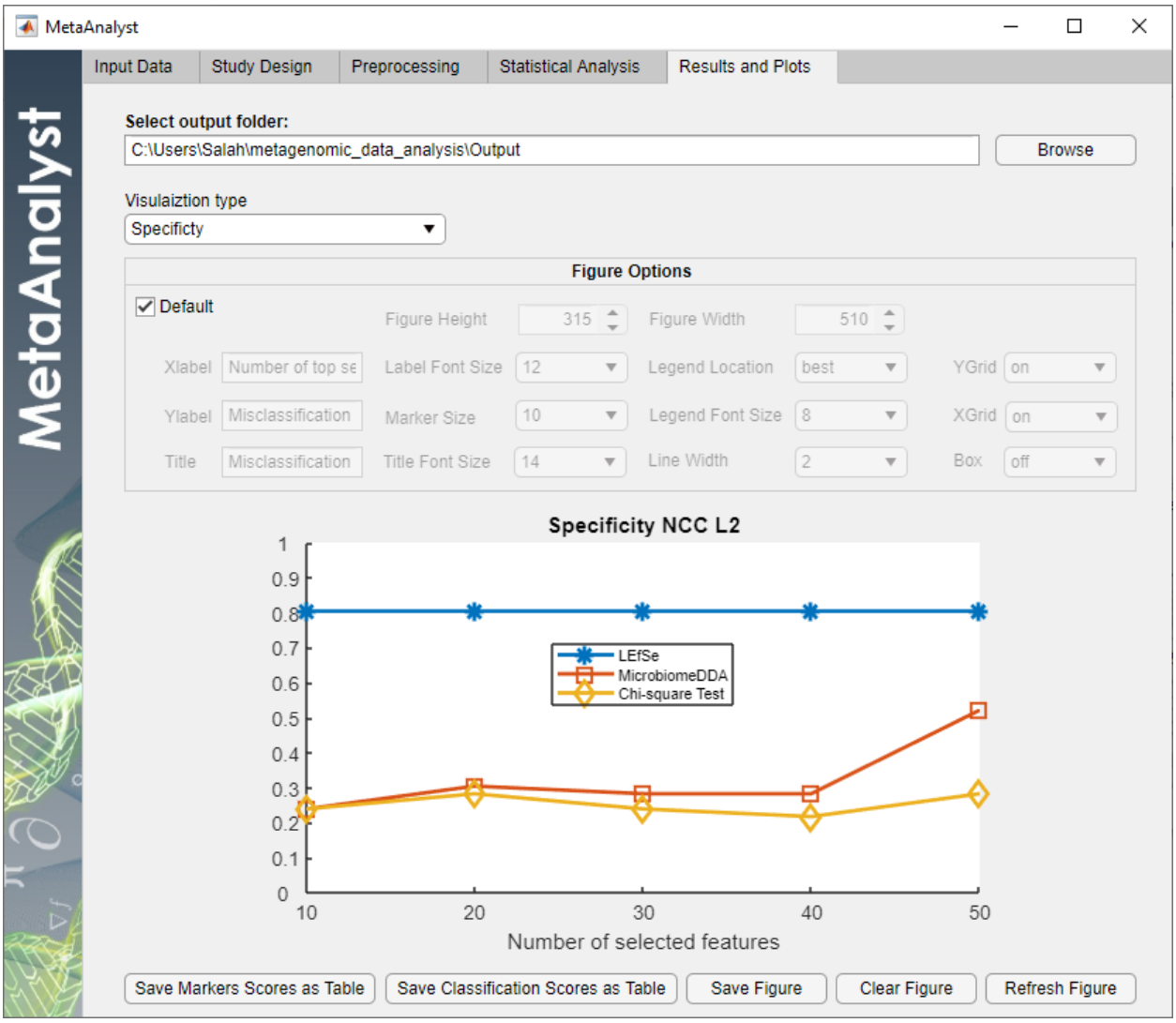


Fig.39 Specificity rate

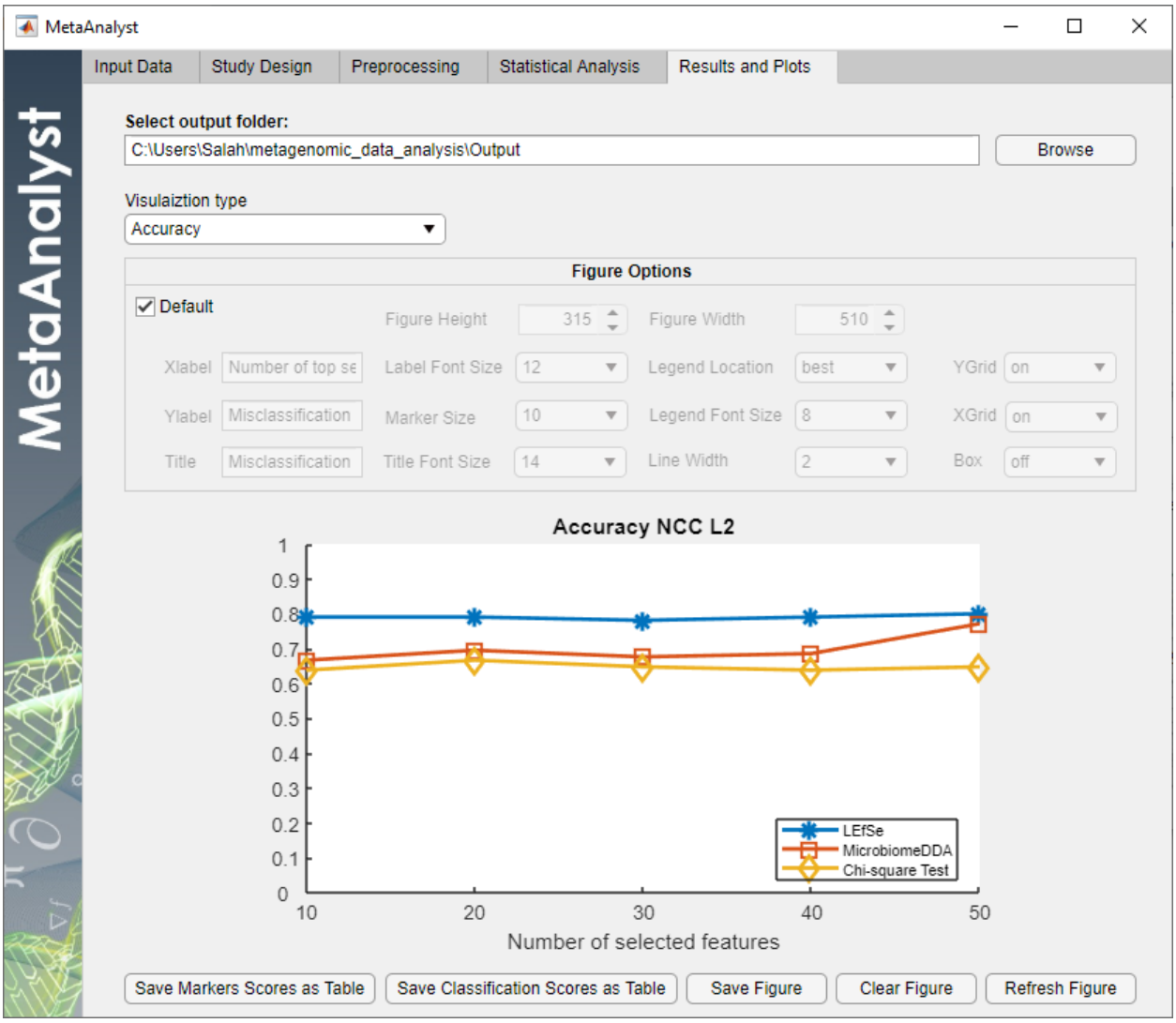


Fig.40 Accuracy

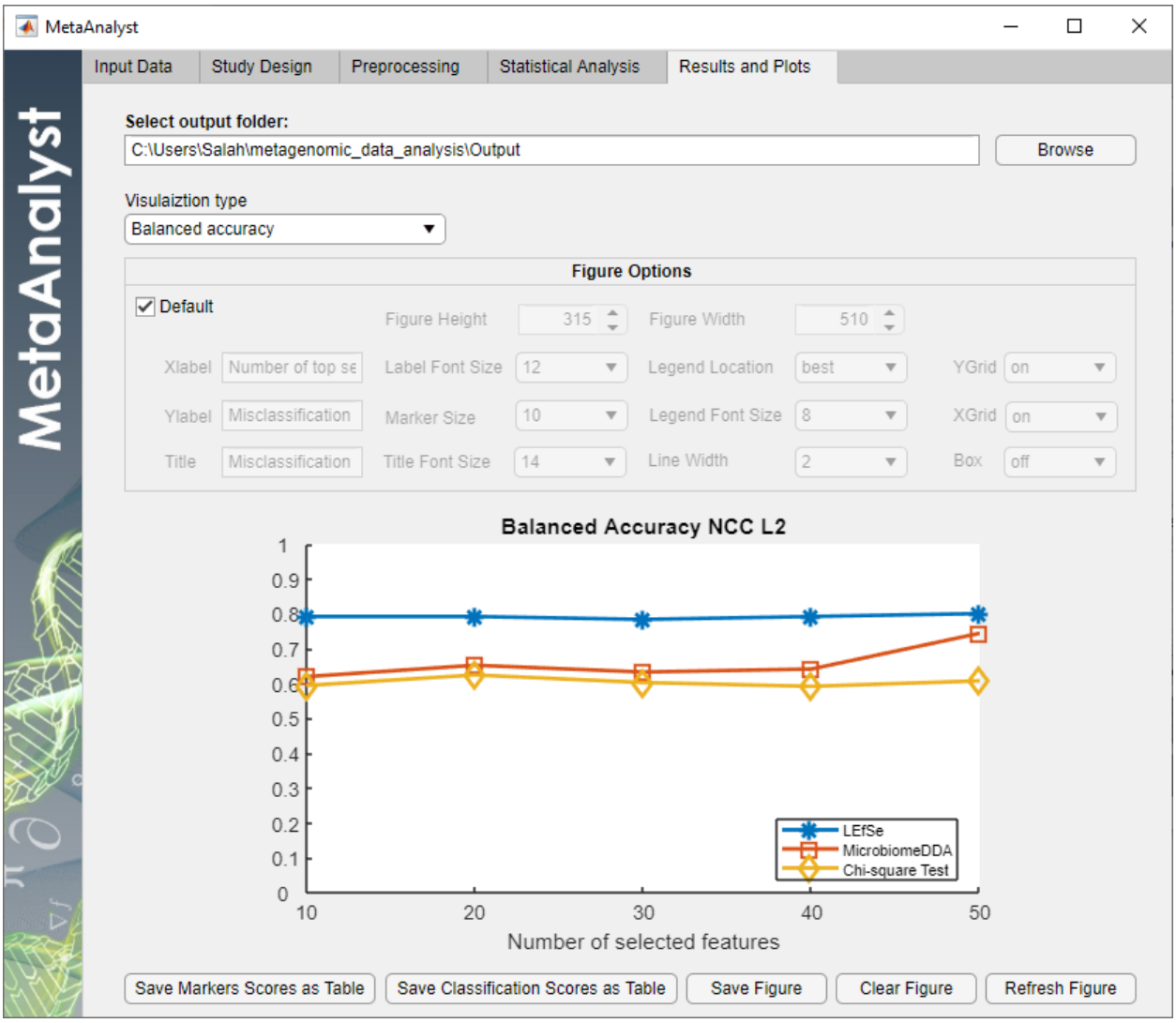


Fig.41 Balanced Accuracy

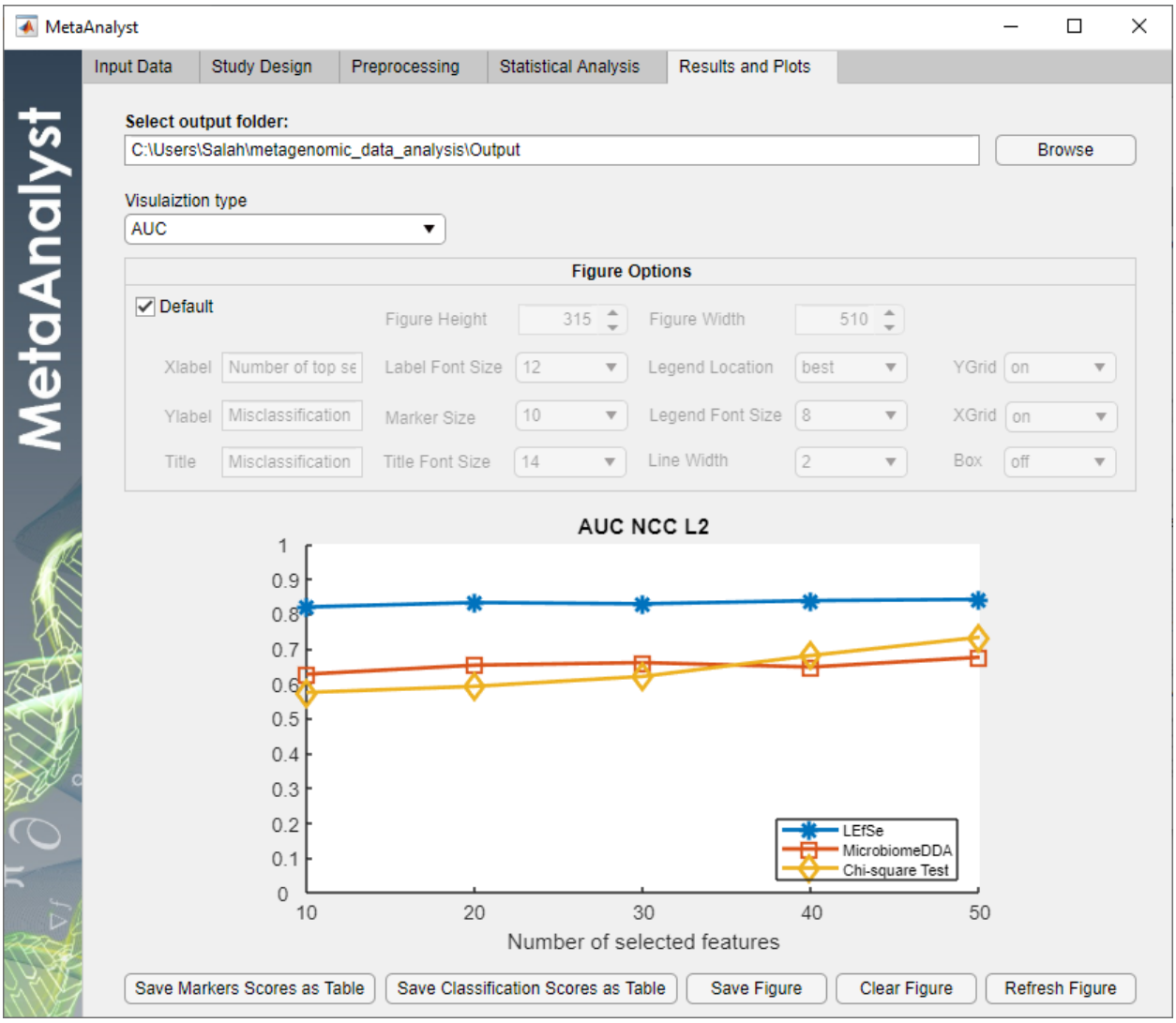

MetaAnalyst
Input Data
Study Design
Preprocessing
Statistical Analysis
Results and Plots
MetaAnalyst
Select output folder:
C:\Users\Salah\metagenomic_data_analysis\Output
Browse
Visulaiztion type
AUC
Figure Options
Default
Figure Height
315
Figure Width
510
Xlabel
Number of top se
Label Font Size
12
Legend Location
best
YGrid
on
Ylabel
Misclassification
Marker Size
10
Legend Font Size
8
XGrid
on
Title
Misclassification
Title Font Size
14
Line Width
2
Box
off
AUC NCC L2
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
10
20
30
40
50
Number of selected features
LEfSe
MicrobiomeDDA
Chi-square Test
Save Markers Scores as Table
Save Classification Scores as Table
Save Figure
Clear Figure
Refresh Figure

Fig.42 AUC

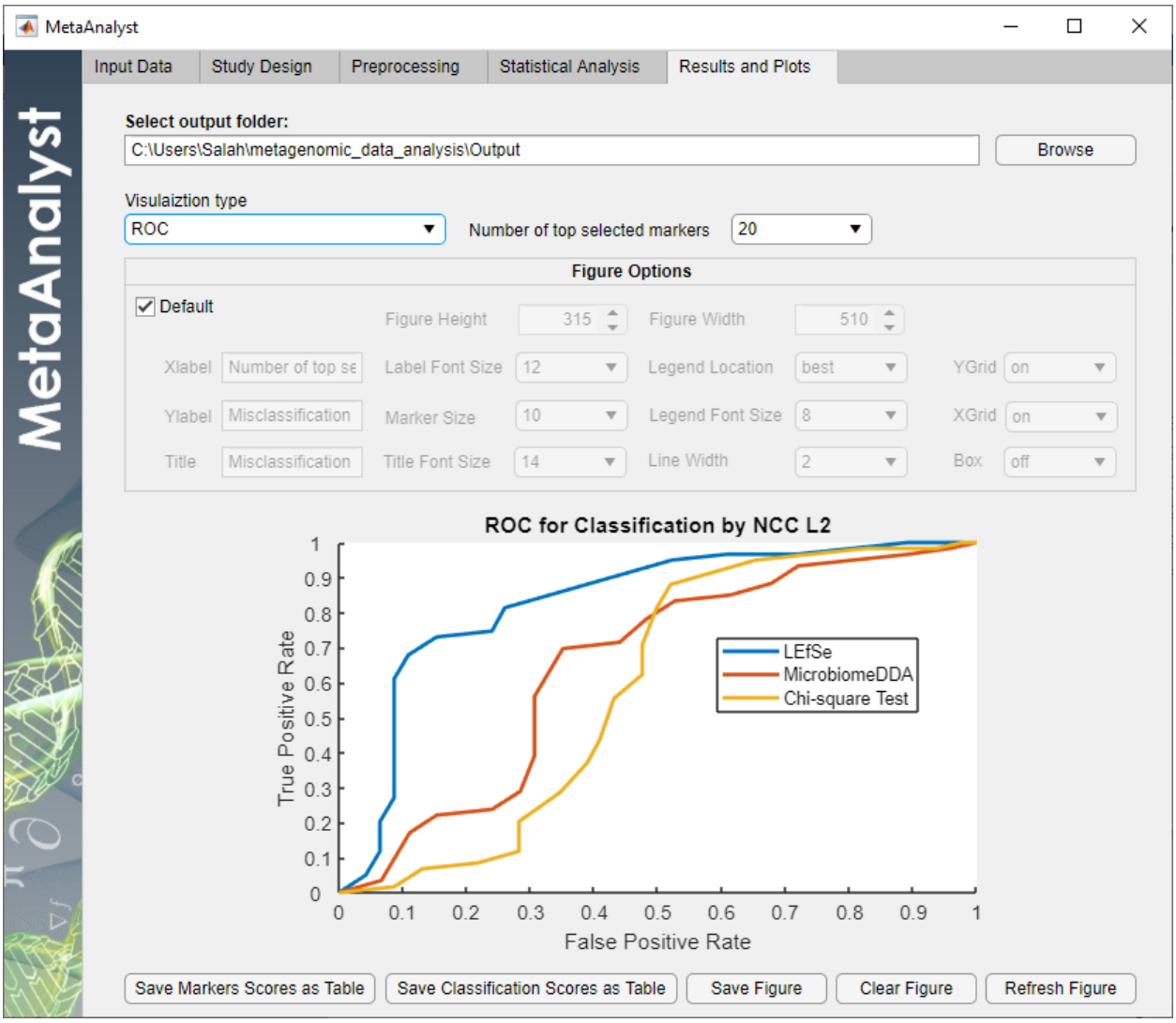

MetaAnalyst
Input Data
Study Design
Preprocessing
Statistical Analysis
Results and Plots
Select output folder:
C:\Users\Salah\metagenomic_data_analysis\Output
Browse
Visulaiztion type
ROC
Number of top selected markers
20
Figure Options
Default
Figure Height
315
Figure Width
510
Xlabel
Number of top se
Label Font Size
12
Legend Location
best
YGrid
on
Ylabel
Misclassification
Marker Size
10
Legend Font Size
8
XGrid
on
Title
Misclassification
Title Font Size
14
Line Width
2
Box
off
ROC for Classification by NCC L2
True Positive Rate
False Positive Rate
LEfSe
MicrobiomeDDA
Chi-square Test
Save Markers Scores as Table
Save Classification Scores as Table
Save Figure
Clear Figure
Refresh Figure

Fig.43 ROC

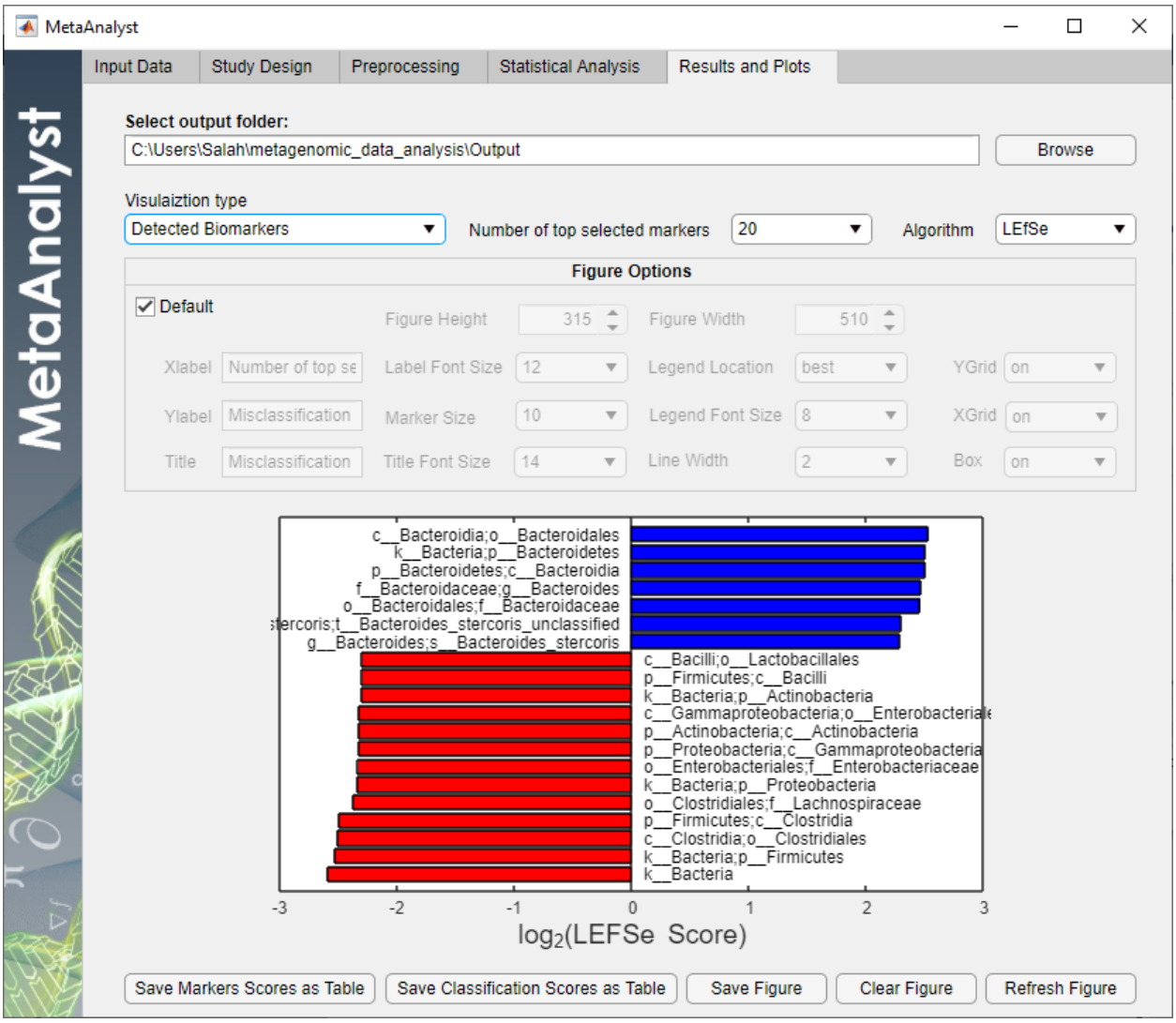


Fig.44 Top 20 markers detected by LEfSe algorithm

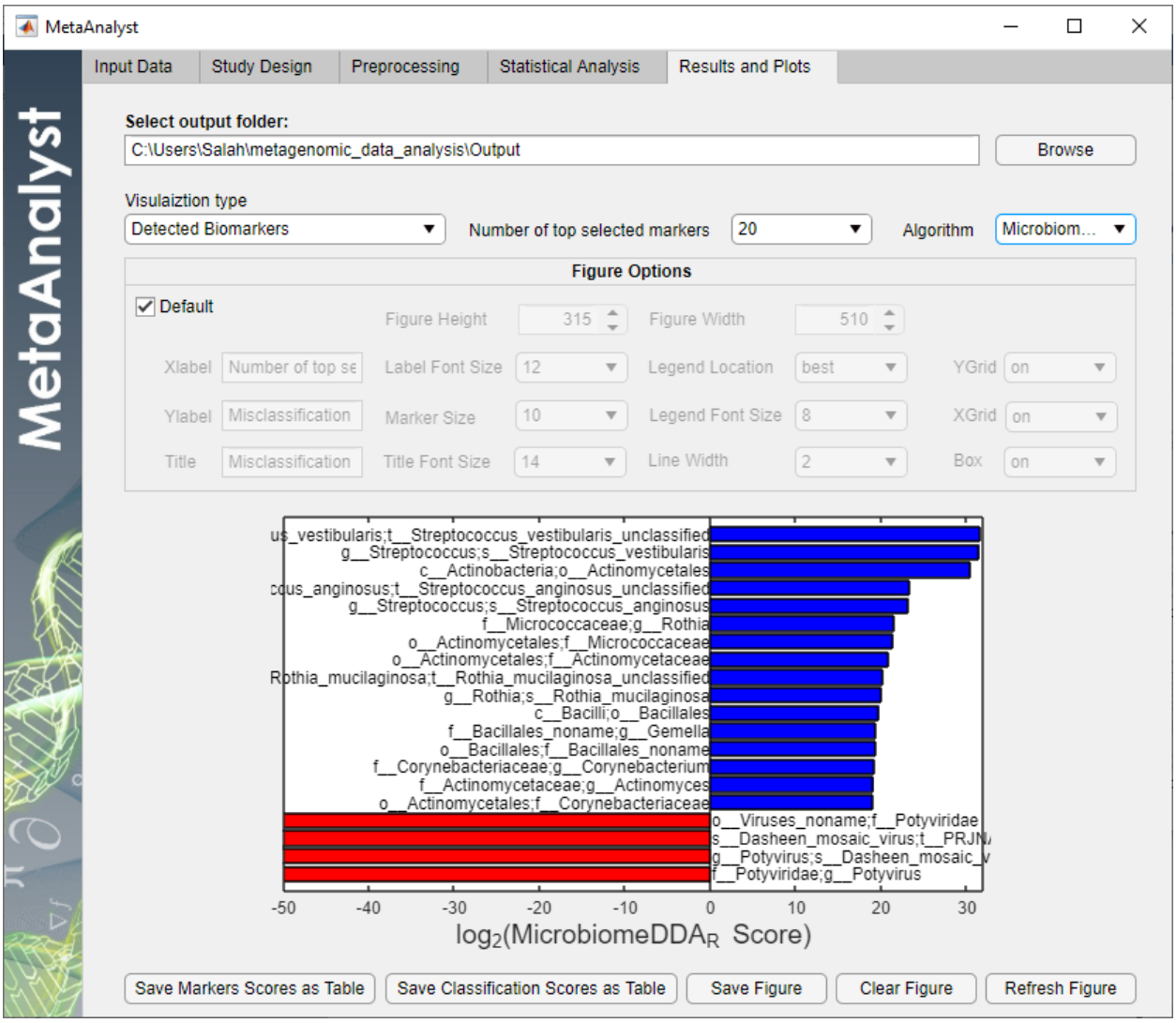


Fig.45 Top 20 markers detected by MicrobiomeDDA algorithm

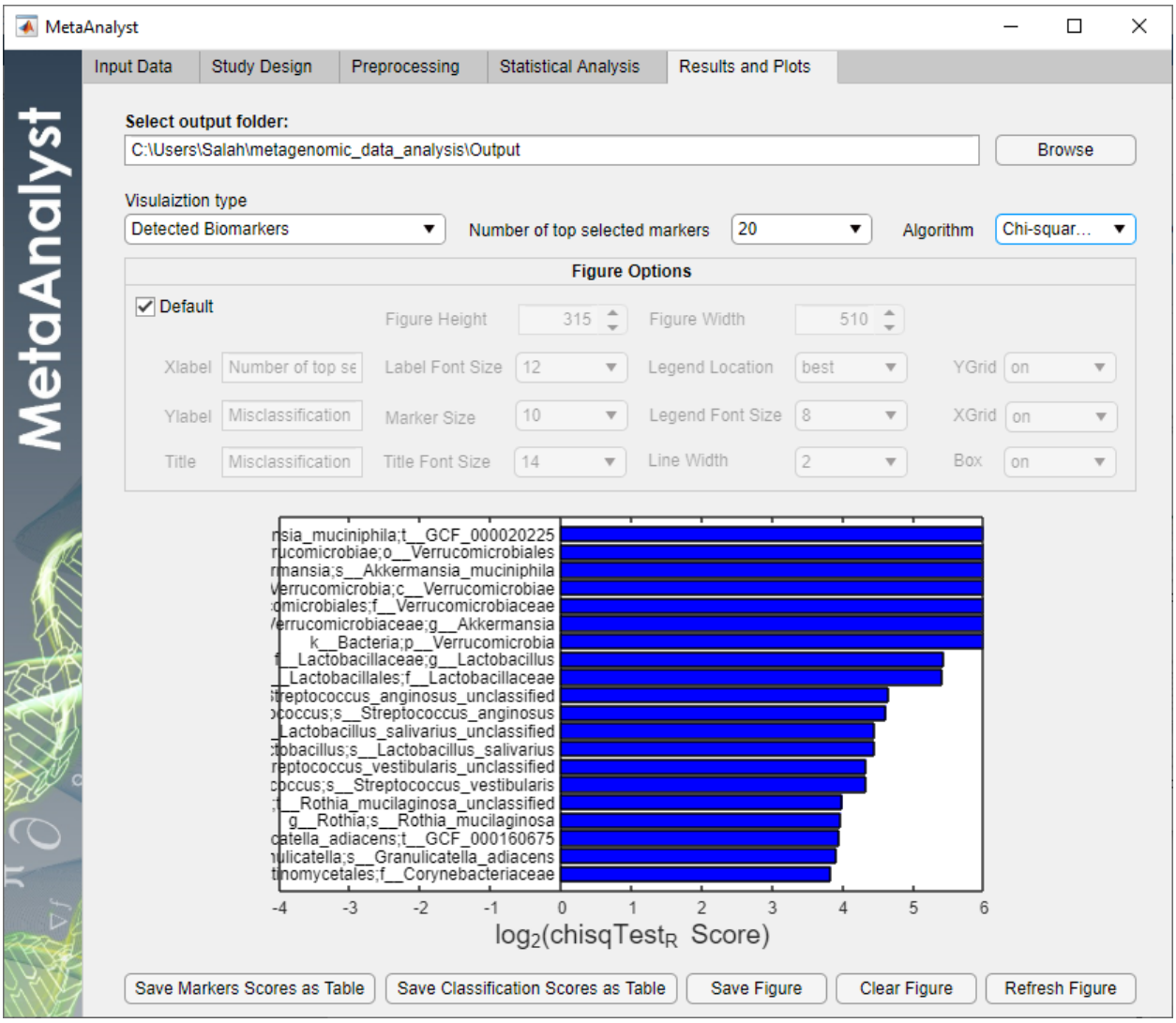


Fig.46 Top 20 markers detected by Chi-square test

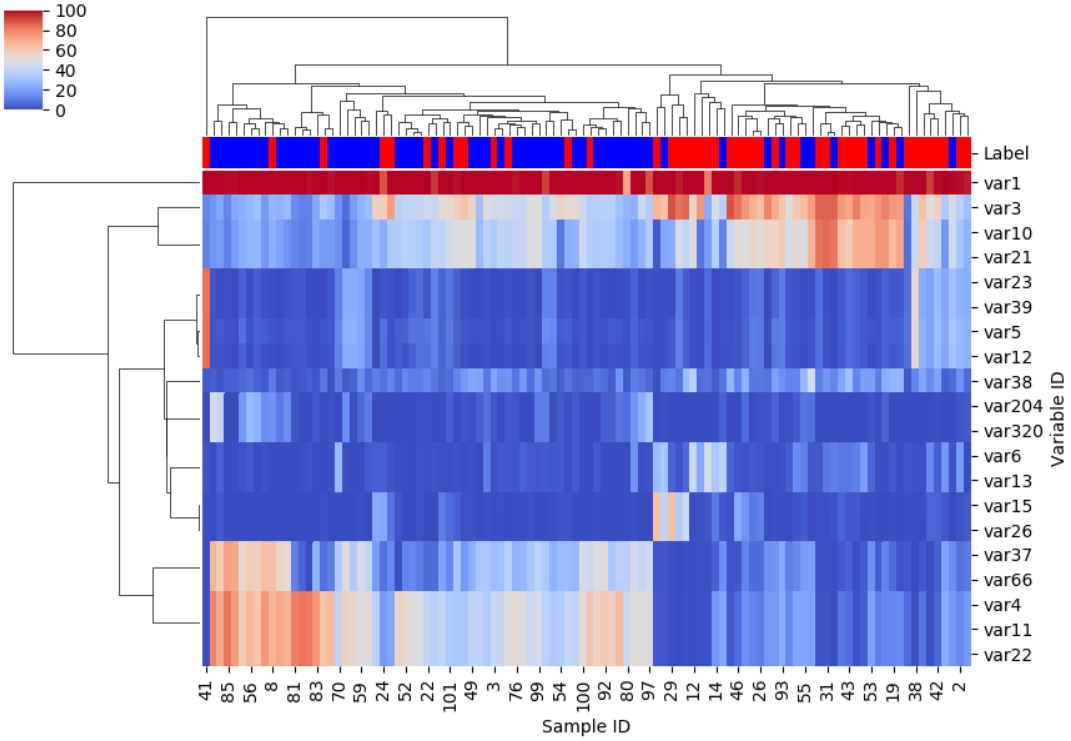


Fig.47 LEfSe clustering performance

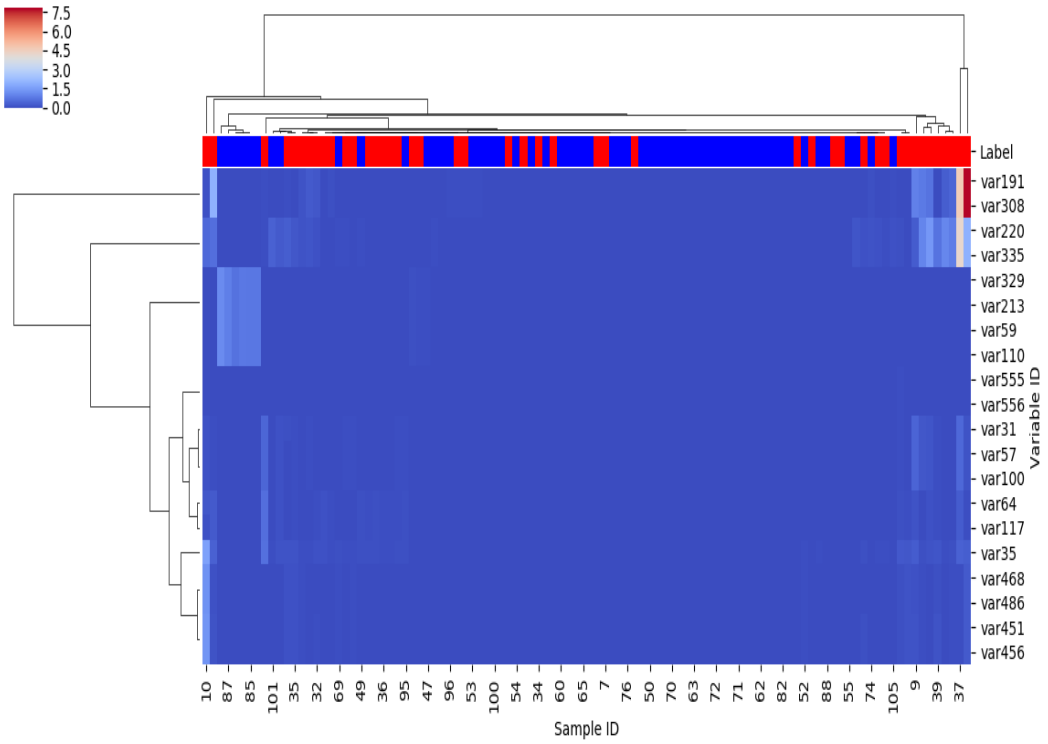


Fig.48 MicrobiomeDDA clustering performance

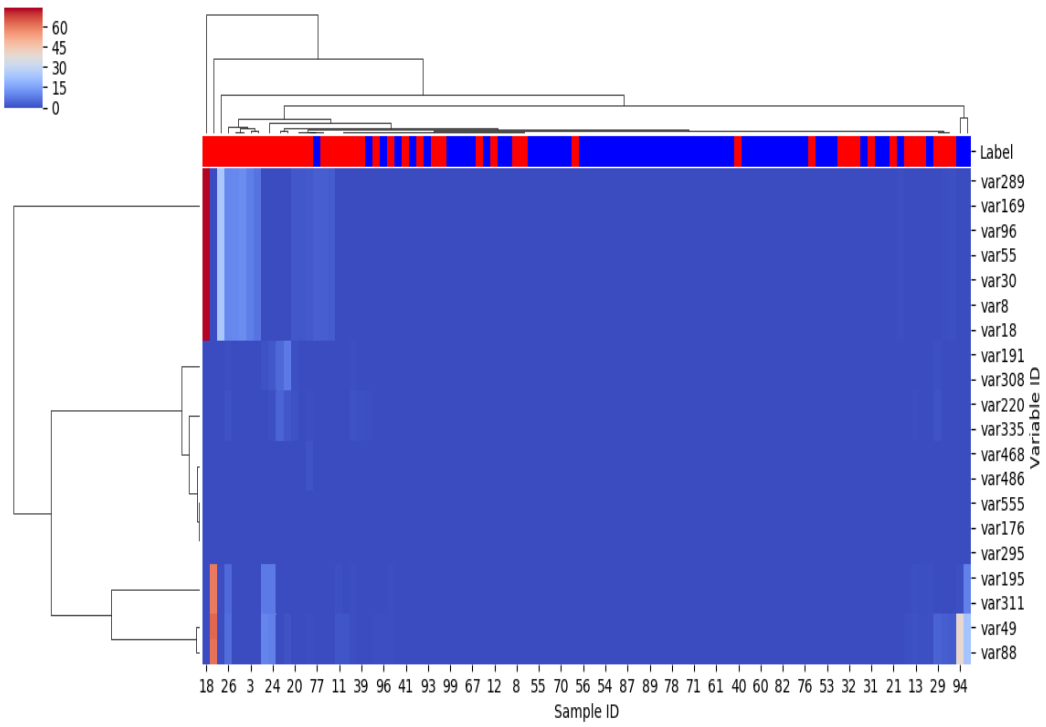


Fig.49 Chi-square clustering performance

3) The software provides a utility to change the options of the figure. Firstly, the user should uncheck the 'Default' checkbox. As the user change the figure options, the software will automatically adjust the figure accordingly, as shown in Fig. 50. Furthermore, the 'Clear Figure' and 'Refresh Figure' flush and refresh the figure, respectively.

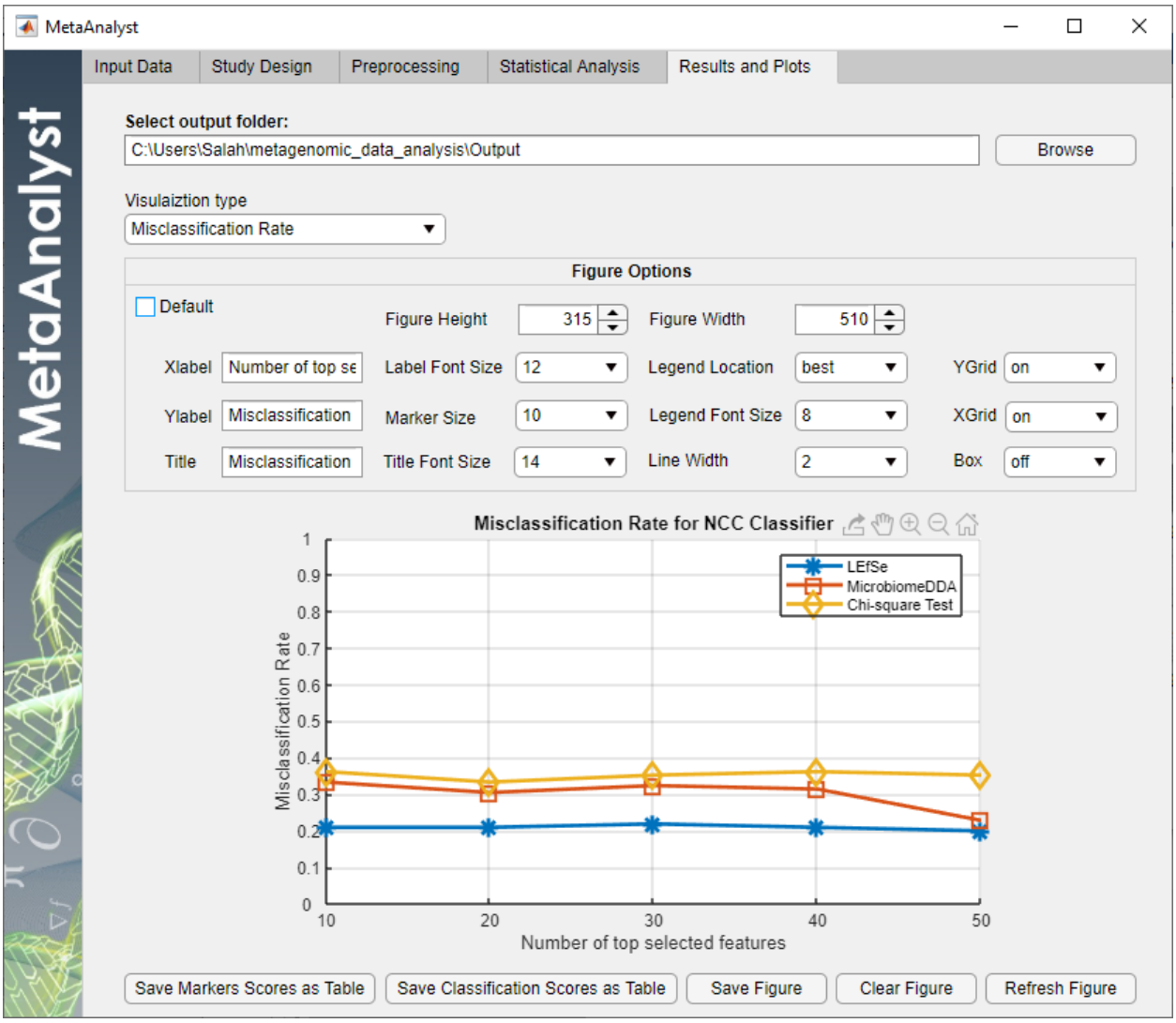
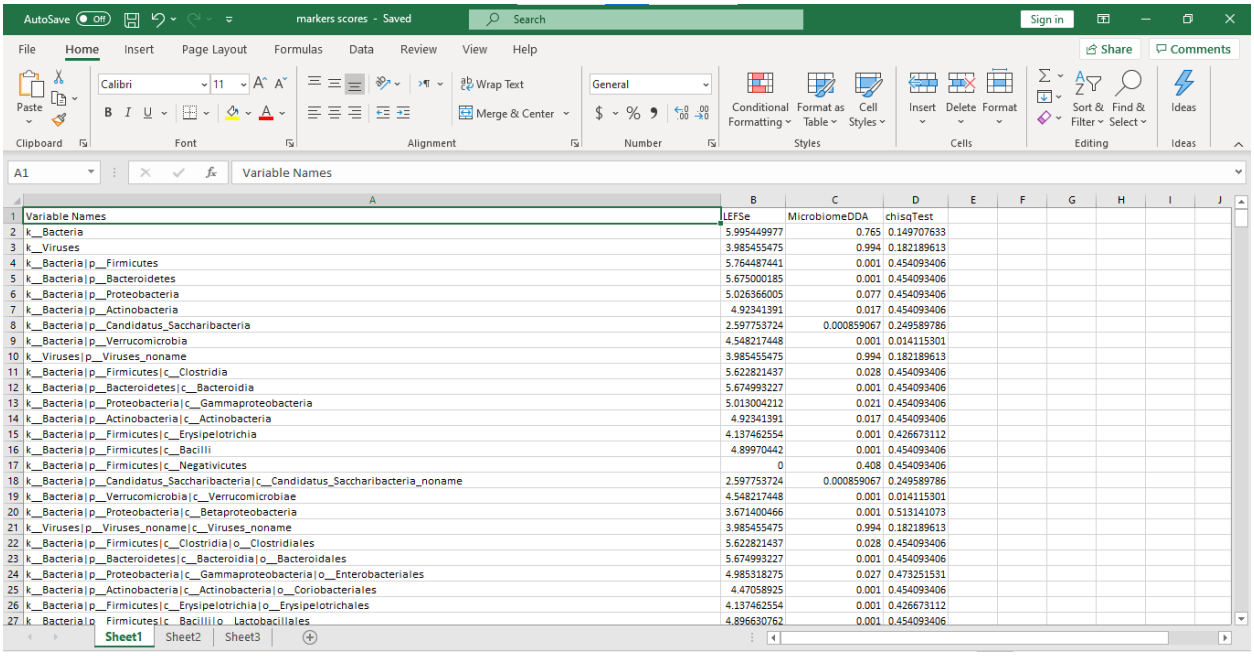


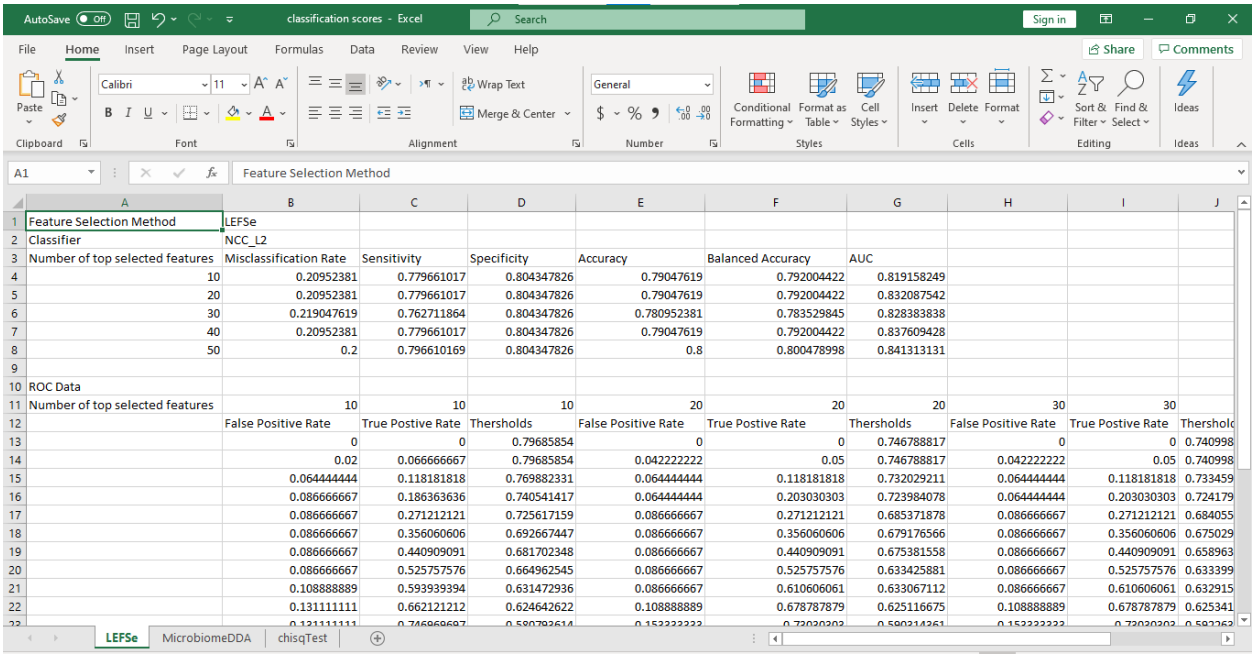
Fig.50 Adjusting figure's options

4) The two buttons, 'Save Marker Scores as Table' and 'Save Classification Scores as Table' allow the user to save the marker scores and the classification scores as an excel file. Fig. 51. shows the markers scores for each selected algorithm and Fig. 52 shows the classification scores of this example. The 'Save Figure' button allows the user to save the current plot. The software provides 13 different formats to save the current plot as described in Results and Plots section.



Variable Names	LEFSe	MicrobiomeDDA	chisqTest
k__Bacteria	5.995449977	0.765	0.149707633
k__Viruses	3.985455475	0.994	0.182189613
k__Bacteria|p__Firmicutes	5.764487441	0.001	0.454093406
k__Bacteria|p__Bacteroidetes	5.675000185	0.001	0.454093406
k__Bacteria|p__Proteobacteria	5.026366005	0.077	0.454093406
k__Bacteria|p__Actinobacteria	4.92341391	0.017	0.454093406
k__Bacteria|p__Candidatus_Saccharibacteria	2.597753724	0.000859067	0.249589786
k__Bacteria|p__Verrucomicrobia	4.548217448	0.001	0.014115301
k__Viruses|p__Viruses_noname	3.985455475	0.994	0.182189613
k__Bacteria|p__Firmicutes|c__Clostridia	5.622821437	0.028	0.454093406
k__Bacteria|p__Bacteroidetes|c__Bacteroidia	5.674993227	0.001	0.454093406
k__Bacteria|p__Proteobacteria|c__Gammaproteobacteria	5.013004212	0.021	0.454093406
k__Bacteria|p__Actinobacteria|c__Actinobacteria	4.92341391	0.017	0.454093406
k__Bacteria|p__Firmicutes|c__Erysipelotrichia	4.137462554	0.001	0.426673112
k__Bacteria|p__Firmicutes|c__Bacilli	4.89970442	0.001	0.454093406
k__Bacteria|p__Firmicutes|c__Negativicutes	0	0.408	0.454093406
k__Bacteria|p__Candidatus_Saccharibacteria|c__Candidatus_Saccharibacteria_noname	2.597753724	0.000859067	0.249589786
k__Bacteria|p__Verrucomicrobia|c__Verrucomicrobiae	4.548217448	0.001	0.014115301
k__Bacteria|p__Proteobacteria|c__Betaproteobacteria	3.671400466	0.001	0.513141073
k__Viruses|p__Viruses_noname|c__Viruses_noname	3.985455475	0.994	0.182189613
k__Bacteria|p__Firmicutes|c__Clostridia|o__Clostridiales	5.622821437	0.028	0.454093406
k__Bacteria|p__Bacteroidetes|c__Bacteroidia|o__Bacteroidales	5.674993227	0.001	0.454093406
k__Bacteria|p__Proteobacteria|c__Gammaproteobacteria|o__Enterobacteriales	4.985318275	0.027	0.473251531
k__Bacteria|p__Actinobacteria|c__Actinobacteria|o__Coriobacteriales	4.47058925	0.001	0.454093406
k__Bacteria|p__Firmicutes|c__Erysipelotrichia|o__Erysipelotrichales	4.137462554	0.001	0.426673112
k__Bacteria|p__Firmicutes|c__Bacilli|o__Lactobacillales	4.896630762	0.001	0.454093406

Fig.51 Markers scores



Feature Selection Method	LEFSe								
Classifier	NCC_L2								
Number of top selected features	Misclassification Rate	Sensitivity	Specificity	Accuracy	Balanced Accuracy	AUC			
10	0.20952381	0.779661017	0.804347826	0.79047619	0.792004422	0.819158249			
20	0.20952381	0.779661017	0.804347826	0.79047619	0.792004422	0.832087542			
30	0.219047619	0.762711864	0.804347826	0.780952381	0.783529845	0.828383838			
40	0.20952381	0.779661017	0.804347826	0.79047619	0.792004422	0.837609428			
50	0.2	0.796610169	0.804347826	0.8	0.800478998	0.841313131			
ROC Data									
Number of top selected features	10	10	10	20	20	20	30	30	
	False Positive Rate	True Postive Rate	Thersholds	False Positive Rate	True Postive Rate	Thersholds	False Positive Rate	True Postive Rate	Thersholc
	0	0	0.79685854	0	0	0.746788817	0	0	0.740998
	0.02	0.066666667	0.79685854	0.042222222	0.05	0.746788817	0.042222222	0.05	0.740998
	0.064444444	0.118181818	0.769882331	0.064444444	0.118181818	0.732029211	0.064444444	0.118181818	0.733459
	0.086666667	0.186363636	0.740541417	0.064444444	0.203030303	0.723984078	0.064444444	0.203030303	0.724179
	0.086666667	0.271212121	0.725617159	0.086666667	0.271212121	0.685371878	0.086666667	0.271212121	0.684055
	0.086666667	0.356060606	0.692667447	0.086666667	0.356060606	0.679176566	0.086666667	0.356060606	0.675029
	0.086666667	0.440909091	0.681702348	0.086666667	0.440909091	0.675381558	0.086666667	0.440909091	0.658963
	0.086666667	0.525757576	0.664962545	0.086666667	0.525757576	0.633425881	0.086666667	0.525757576	0.633399
	0.108888889	0.593939394	0.631472936	0.086666667	0.610606061	0.633067112	0.086666667	0.610606061	0.632915
	0.131111111	0.662121212	0.624642622	0.108888889	0.678787879	0.625116675	0.108888889	0.678787879	0.625341

Fig.52 Classification scores