Metabolomics_analysis_tools Tutorials

Introduction to the Metabolomics analysis tools

The goal of this project is to implement a Python based pipeline or package related to metabolomics data analysis. I am currently working with targeted metabolomics data in my lab, and it will be helpful with my work to develop a package that contains some very common metabolomics data analysis tools, including:

data transformation data normalization data scaling common statistical analyses including PCA, MA plot and Volcano plot. Even though there are lots of packages available for the functions mentioned above, implementing them myself will help me understand those functions better and help me do a better analysis job hopefully.

Goal of the tutorial

The goal of this tutorial is to show step by step how to install and use this package to perform data processing and statistical analyses functions in this tool, and also a bit on how to read the results generated from the analyses functions.

Example data description

Targeted metabolomics data Retrieve metabolites concentration data from (https://www.metaboanalyst.ca/Meta
or from https://www.ebi.ac.uk/metabolights/search?) (Metabolights)

A data file with metabolites concentration profile will contain a matrix with metabolites names and metabolites concentrations on each row for each sample. This file will be provided by the user, I will have data for testing

Step by step installation and running

Let's get started! First, we need to install the package.
Steps: 1. Git clone or download the github folder;
2. Open the terminal, and go to this folder;
3. Enter
`pip install dist/metabolomics_analysis_tools-0.1.0.tar.gz` to install the package locally;

Then, we can import functions we will use for this demo from the package metabolomics_analysis_tools@import_

```
import metabolomics_analysis_tools.data_preprocessing.data_reading as dr
import metabolomics_analysis_tools.data_preprocessing.normalization as dn
import metabolomics_analysis_tools.stats_analyses.analyses as sa
import metabolomics_analysis_tools.data_preprocessing.data_check as dc
import warnings
warnings.filterwarnings('ignore')
```

1. Then we can use the data_reading module to read in the data, by default it will read in the data from the resources/test_dataset folder in the package.
   We can also use the data_reading module to read in the data from a custom path, by passing the path as an argument to the read_data_file function (file_path='path/to/file.csv').
   The read_data_file function will return a pandas dataframe.

```
df=dr.read_data_file()
df.head()
```

```
data read successfully
the shape of the dataframe is:  (77, 65)
```

	Patient ID	Muscle loss	1,6-Anhydro-beta-D-glucose	1-Methylnicotinamide	2-Aminobutyrate
0	PIF_178	cachexic	40.85	65.37	18.73
1	PIF_087	cachexic	62.18	340.36	24.29
2	PIF_090	cachexic	270.43	64.72	12.18
3	NETL_005_V1	cachexic	154.47	52.98	172.43
4	PIF_115	cachexic	22.2	73.7	15.64

We can use the data_check module to check if the data is normally distributed.
The normal_dist_check function will return true if the data is normally distributed, false otherwise

```
dc.normal_dist_check(df)
```

```
True
```

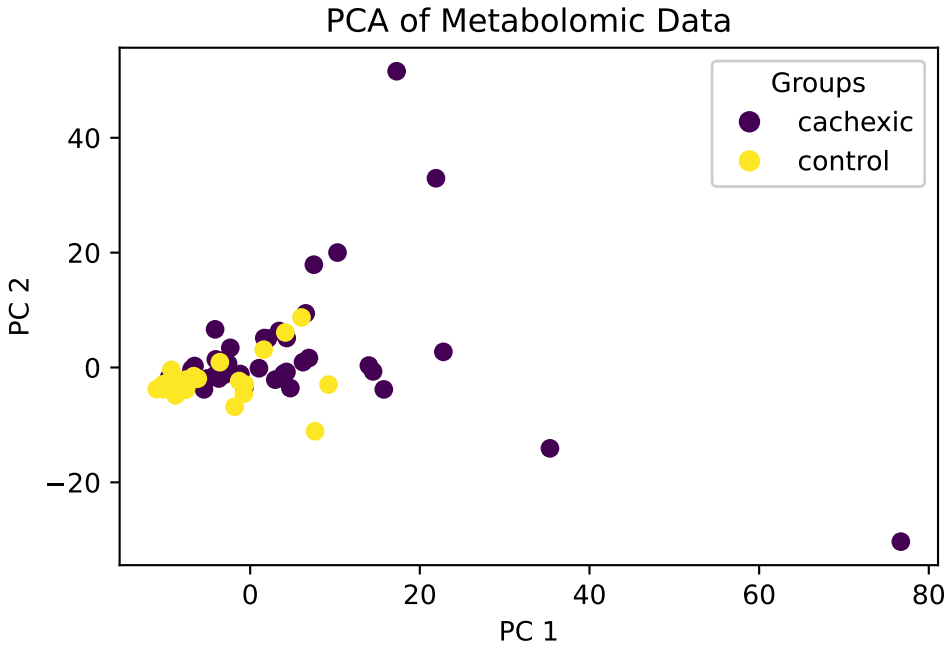
2. Next we can use the normalization module to normalize the data, here we will use the median normalization method normalized_data=dn.normalize_by_median(df).
   We can have a look at the first 5 rows of the normalized data normalized_data.head().

```
normalized_data=dn.normalize_by_median(df)
normalized_data.head()
```

	Patient ID	Muscle loss	1,6-Anhydro-beta-D-glucose	1-Methylnicotinamide	2-Aminobutyrate
0	PIF_178	cachexic	0.895833	1.786066	1.78551
1	PIF_087	cachexic	1.363596	9.299454	2.315539
2	PIF_090	cachexic	5.930482	1.768306	1.161106
3	NETL_005_V1	cachexic	3.3875	1.447541	16.43756
4	PIF_115	cachexic	0.486842	2.013661	1.490944

3. (a)We can use the analyses module to perform statistical analyses on the data.
   Here we will first perform a PCA analysis on the data to see if there are any patterns in the data.
   The PCA_analysis function will return a pandas dataframe containing the principal components principal_components=sa.PCA_analysis(normalized_data).

```
principal_components=sa.PCA_analysis(normalized_data)
```

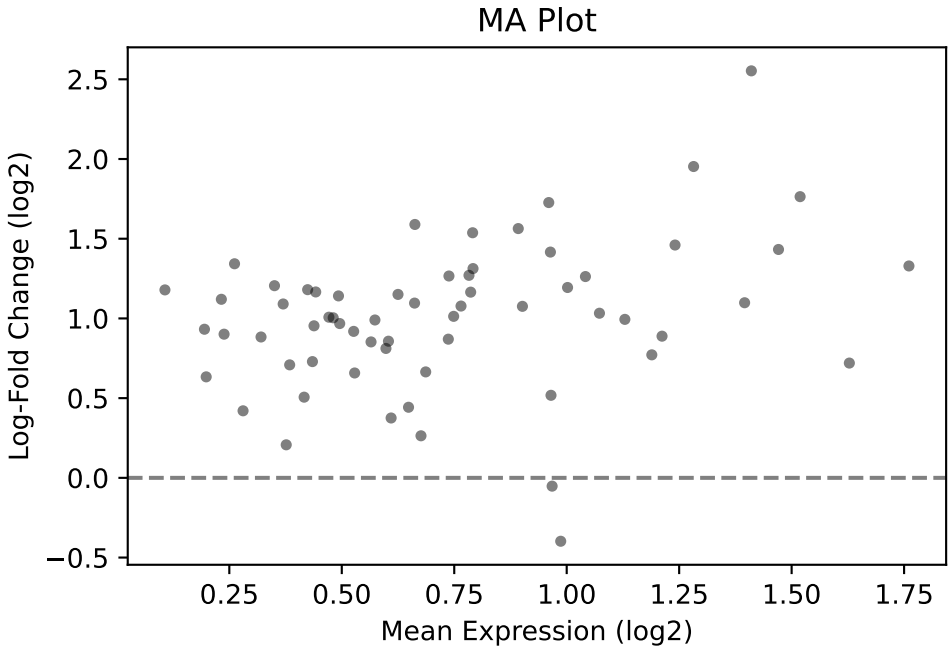


- Interpretation of the PCA results:
  From the PCA results, we can see the first two PCs could explain about 41% of the variance in our data, which is not very high. We can also see that the samples

are not well separated in the PCA plot, so we can't explain the group difference in dataset very well with the top 2 features we have, which might be due to there is a non-linear relationship here in the features that couldn't be all explained by simply PCA analysis. Therefore, further non-linear analysis might be needed to better understand the relationship between the features and the samples. (I will add more functions to the package in the future to perform non-linear analysis, such as t-SNE, UMAP, etc.)

3. (b)Next, we can do the same for the MA plot.
   The MA_plot function will return a pandas dataframe containing the log2 fold change and the -log10 p-value

```
MA_plot=sa.ma_plot(normalized_data)
```

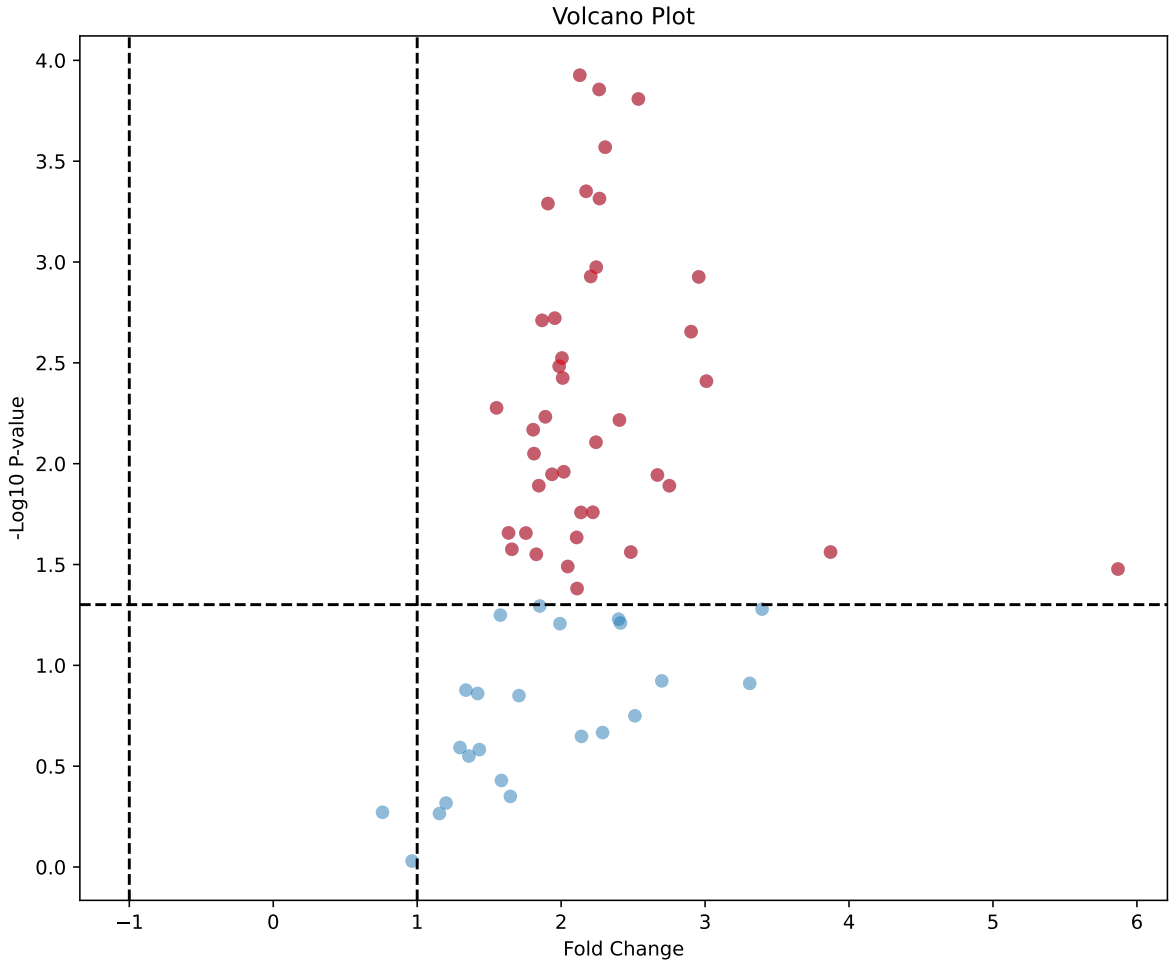


- Interpretation of the MA plot results:
  The MA plot shows that for most metabolites identified here, they were upregulated in the non-control group, which is an interesting observation here. It is about the idea that, for patients with disease, their metabolites profile might be different from the healthy people, and this difference might be due to they are unable to metabolize some metabolites (decreased liver function), or they are producing more metabolites than the healthy people. This is a very interesting observation, and we can further explore this

in the future.

3. (c)We can also do a volcano plot, which can show us the significantly differentially expressed metabolites in the data.
   The volcano_plot function will return a pandas dataframe containing the log2 fold change and the -log10 p-value volcano_plot=sa.volcano_plot(normalized_data)

```
volcano_plot=sa.volcano_plot(normalized_data)
```



- Interpretation of the volcano plot results:
  For the volcano plot here, every point represents a metabolite. The x axis represents the fold change of different metabolites, and y axis represents the adjusted p values using Benjamini-Hochberg method. The red points are the metabolites that are

significantly different between the two groups, and the blue points are the metabolites that are not significantly different between the two groups. We can see that there are a lot of metabolites that are significantly different between the two groups, which is consistent with the MA plot results. We can also see that there are some metabolites that are not significantly different between the two groups, but they have a very high fold change, which is also interesting. We can further explore this in the future.

Conclusion about applications of the tool

- As we can see, using metabolomics analysis tools can help us process the metabolites data and to better visually explore the metabolites as features and metabolite expressions we have in different groups of subjects, and help us to generate some interesting hypothesis that we can further explore in the future.