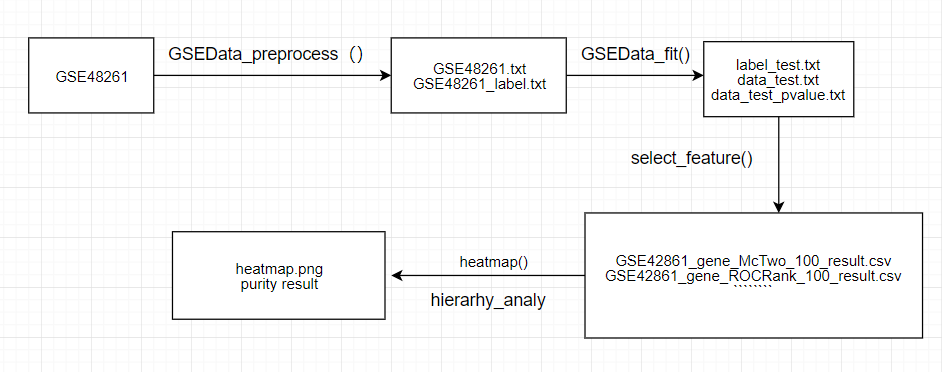
The Mege program is divided into three parts



Part1: process the methylated data set through Mege to generate the newly constructed data set.All of the functions in this step are in the mege.py file, using two main functions

Function1：GSEData\_preprocess(code,pname,nname,rowhead) (preprocessing the data set)

For examples：GSEData\_preprocess('GSE42861','\"Patient genomic DNA from sample1\"','\"Normal genomic DNA from sample 52\"','!Sample\_title')

code: the number of the data set of methylation group

pname: original file positive sample label

nname: negative sample label of original file

rowhead: sample label line start flag

Function2：GSEData\_fit(code,model\_type,data\_type) (feature construction of the data set)

For examples：GSEData\_fit('GSE42861','LinearRegression','origin\_data')

code：'LinearRegression','LogisticRegression','L1','L2'，four types of regression

model\_type:'origin\_data'(original data),'square\_data'(original data square root),'log\_data'(original data take logarithm)，'radical\_data'(original data square),'cube\_data' (original data cube)

Part2：Through the feature selection algorithm of the generated data set, the results of seven feature selection are obtained.feature\_selection.py中

Function：select\_feature(filename,labelname,pvaluename,r,times,fold,cnt,datasetName,geneOrSite)

parameter description:

filename: feature data filename for feature selection, labelname: tag filename, pvaluename: TRank pvalue filename, all three are TXT files

r: threshold for McTwo (McOne) feature selection

times: times of cross validation, fold: fold of cross validation points

CNT: number of features to perform IFS

datasetName: the datasetName (such as GSE66695) is used to generate the result write file name

geneOrSite: indicate whether it is a gene or a site. If it is a site, SVM\_RFE,LR\_RFE select the first 20,000 with the smallest pvalue for feature selection.

Part3：Through the feature selection algorithm of the data set, the clustering analysis is carried out. The two files mainly focus on two functions.

Heatmap(heatmap mapping)

Hierarhy\_analys.py (hierarchical clustering purity calculation)

heatmap.py：Function：heatmap(fn\_name,data\_type,rank\_num,data\_name)

The result returns the generated heat map image

heatmap(['McTwo'],'gene',[10,25,35],'GSE42861')

fn\_name: ['SVM\_RFE'.’McTwo’,’TRank’, ’WRank’, ’ROCRank’, ’RF’,’LR\_RFE’]

data\_type： ‘gene’ or ‘site’

rank\_num: [10,20,30] an int list

data\_name: ‘GSE42861’

hierarhy\_analys.py：Function：hierarhy\_analy(function\_name,rank\_numlist,data\_name,data\_type)

For examples：hierarhy\_analy(['McTwo','TRank'],[10,20,30],"GSE42861",'gene')

Function\_name: ['SVM\_RFE'.’McTwo’,’TRank’, ’WRank’, ’ROCRank’, ’RF’,’LR\_RFE’]

rank\_numlist: [10,20,30,40,50] An integer list, meaning to select the number of features ranked first

data\_name: ‘GSE42861’

data\_type：‘gene’or’site’