Improved prediction of breast cancer outcome by identifying heterogeneous biomarkers

User Manual
(version 2.0)

April 12, 2018

Index

1. Installation

This library 'CPR' requires:

Python 3,
Numpy,(≥1.6.1)
Scipy, (≥0.9)
Scikit-learn (≥0.18)

To download or update those libraries, use 'pip' in command line.

```
pip install numpy scipy scikit-learn
```

Download the CPR module from https://github.com/mathcom/CPR.
If successfully downloaded, user can find the following 3 files:

CPR.py
ex_EXPRESSION.txt
ex_CLINICAL.txt
ex_NETWORK.txt

2. Quick start

1) Run

```
$ python CPR.py [-h] [-m NUMCLUSTERS] [-d DAMPINGFACTOR]
                [-n NUMBIOMARKERS] [-c CONDITIONHUBGENE] [-v]
                EXPRESSION_FILE CLINICAL_FILE NETWORK_FILE
                RESULT_FILE
```

<table>
<tr><th>Option</th><th>Name</th><th>Description</th></tr>
<tr><td></td><td><i>EXPRESSION_FILE</i>
<i>(positional argument)</i></td><td>Tab-delimited file for gene expression profiles as below:
<table>
<tr><td>PATIENT</td><td>TCGA-AR-A24H</td><td>TCGA-AR-A24L</td><td>TCGA-AR-A24M</td></tr>
<tr><td>A1CF</td><td>-0.436158</td><td>-0.276784</td><td>-0.309453</td></tr>
<tr><td>A2M</td><td>1.90128</td><td>2.72735</td><td>4.03939</td></tr>
<tr><td>A4GALT</td><td>-0.408337</td><td>-0.247608</td><td>-0.260444</td></tr>
</table></td></tr>
<tr><td></td><td><i>CLINICAL_FILE</i>
<i>(positional argument)</i></td><td>Tab-delimited file for patient's clinical data as below (LABEL= <b>0</b>:<i>good</i> prognosis and <b>1</b>:<i>poor</i> prognosis):
<table>
<tr><td>PATIENT</td><td>LABEL</td></tr>
<tr><td>TCGA-AR-A24H</td><td>0</td></tr>
<tr><td>TCGA-AR-A24L</td><td>0</td></tr>
<tr><td>TCGA-AR-A2LH</td><td>1</td></tr>
</table></td></tr>
<tr><td></td><td><i>NETWORK_FILE</i>
<i>(positional argument)</i></td><td>Tab-delimited file for gene interaction network as below:
<table>
<tr><td>GENE1</td><td>GENE2</td></tr>
<tr><td>RPL37A</td><td>RPS27A</td></tr>
<tr><td>MRPL1</td><td>MRPS36</td></tr>
<tr><td>RFC3</td><td>SPRTN</td></tr>
</table></td></tr>
<tr><td></td><td><i>RESULT_FILE</i>
<i>(positional argument)</i></td><td>The results of CPR are saved with the following names:
<i>1) RESULT_FILE_biomarker.txt</i>
<i>2) RESULT_FILE_subnetwork.txt</i>
<i>3) RESULT_FILE_score.txt</i></td></tr>
</table>

Option	Name	Description
-h	*Help message* *(optional argument)*	Show this help message and exit
-m	*Number of sample clusters* *(optional argument)*	A parameter of *K-means clustering* algorithm. This parameter decides number of sample clusters to handle the heterogeneity of patients. If the default value is given, the number of clusters is determined by the *silhouette score*. If a specific integer is given, the K-means clustering is conducted with the given number. (default=0)
-d	*Damping factor* *(optional argument)*	A parameter of *PageRank* algorithm. This parameter decides an influence of network information on prediction. The value must be between 0 and 1. (default=0.7)
-n	*Number of markers* *(optional argument)*	This parameter decides number of biomarkers to use in prediction. (default=70)
-c	*Condition of hub-gene* *(optional argument)*	This parameter is used to identify a hub-gene. When *c* is a given parameter and *x* is the total of genes, we define top *cx* genes with high degree as hub-genes. The value must be between 0 and 1. (default=0.02)
-v	*Flag of Cross validation* *(optional argument)*	When this option is given, CPR.py will conduct 10-fold cross-validation with the given data. The result of cross validation is provided in 4) *RESULT_FILE*_accuracy.txt

2) Example

$ **python CPR.py** ex_EXPRESSION.txt ex_CLINICAL.txt ex_NETWORK.txt ex_RESULT

(1) Log in command line

```
>>> 0. Arguments
Namespace(CLINICAL_FILE='ex_CLINICAL.txt',
EXPRESSION_FILE='ex_EXPRESSION.txt', NETWORK_FILE='ex_NETWORK.txt',
RESULT_FILE='ex_RESULT', conditionHubgene=0.02,
crossvalidation=False, dampingFactor=0.7, numBiomarkers=70,
numClusters=0)
>>> 1. Load data
>>> 2. Preprocess data
   n_samples: 189
   n_genes  : 8819     (common genes in both EXPRESSION and NETWORK)
   n_edges  : 150168   (edges with the common genes)
>>> 3. Conduct CPR
   K-means clustering
   -> n_clusters: 2
      In cluster[0], n_samples:85, n_goods:51, n_poors:34
      In cluster[1], n_samples:104, n_goods:48, n_poors:56
   Modified PageRank
>>> 4. Save results
   ex_RESULT_biomarker.txt
   ex_RESULT_score.txt
   ex_RESULT_subnetwork.txt
```

(2) ex_RESULT_biomarker.txt

- A list of biomarkers identified by CPR.py
- The PRscore is the mean of scores computed in each sample cluster.

```
GeneSymbol    PRscore
CREBBP        1.920744
RNPS1         1.888685
HSPA8         1.819867
CALM1         1.761524
CCNB1         1.741875
PIK3CA        1.725398
```

(2) ex_RESULT_score.txt

- A list of whole genes with their PRscores
- The PRscore_*i* is the gene score computed by Modified PageRank in *i*-th cluster

```
GeneSymbol    PRScore_0     PRScore_1
A1CF          0.915375      1.058961
A2M           2.024162      1.752490
A4GNT         0.930889      1.123994
AAAS          0.416830      1.248592
AARS          1.958362      0.901890
```

(3) ex_RESULT_subnetwork.txt

- This subnetwork is an undirected network
- Each edge has at least one biomarker gene.

```
source        target
CREBBP        EP400
RNPS1         RPSA
EGFR          TLR2
EGFR          WASL
GTF2H5        POLR2B
```

3. Description of class CPR

A user can import and utilize CPR.py in python. CPR.py provides one class and its three functions.

1) class CPR

Clustering and PageRank-based gene selection method. For more detail, please refer to “Improved prediction for breast cancer outcome by identifying heterogeneous biomarkers”.

Parameters	**dampingFactor**: float, optional (default=0.7) A parameter of *PageRank* algorithm. This parameter decides an influence of network information on prediction Range = 0.0 ~ 1.0

	n_biomarkers: int, optional (default=70) User can control the number of prognostic biomarkers. **n_clusters**: int, optional (default=0) A parameter of *K-means clustering* algorithm. This parameter decides number of sample clusters to handle the heterogeneity of patients. If the default value is given, the number of clusters is determined by the *silhouette score*. If a specific integer is given, the K-means clustering is conducted with the given number. **c_hubgene**: float, optional (default=0.02) This parameter is used to identify a hub-gene. When *c* is a given parameter and *x* is the total of genes, we define top *cx* genes with high degree as hub-genes. **logshow**: bool, optional (default=False) If *logshow* is *False*, any log in class CPR is not shown in command line.

2) method CPR.fit()

Build a CPR gene selection model.

Parameters	**expr**: numpy.array(shape=[n_samples,n_genes], dtype=numpy.float32) A gene expression dataset without any header. The order of sample must be equal to one of *labels*. The order of gene must be equal to one of *genes*. **labels**: numpy.array(shape=[n_samples], dtype=numpy.int32) A value of vector represents a label of sample. (0: *good* prognosis and 1: *poor* prognosis) The order of label must be equal to one of sample in *expr*. **genes**: list A list of genes in expression data. The order of gene must be equal to one of gene in *expr*. **edges**: list A list of edges, and all edges have *tuple* type. **random_state**: int or None, optional (default=None) This parameter is used in scikit-learn functions. If *int*, the results are always same. If *None*, a result can be different each time.

3) method CPR.get_biomarkers()

A list of biomarkers identified by built model is provided as list type.

Returns	**biomarkers**: list Each element is *tuple* type and has two values. (0:gene symbol and 1:PRscore) The list is sorted by the descending order of PRscore.

3) method CPR.get_PRscores()

A list of the PRscores of whole genes computed in each cluster is provided as list type.

Returns	**PRscores**: list Each element is *tuple* type and has two or more than two values. (0:gene symbol and 1,2,3,…:PRscores)

4) method CPR.get_subnetwork()

A list of edges containing at least one biomarker is provided as list type.

Returns	**edges**: list Each element is *tuple* type and has two gene symbols.

4. Contact

Bug reporting, questions or any suggestions are highly appreciated.

Jonghwan Choi (mathcom@inu.ac.kr)

Jaegyoon Ahn (jgahn@inu.ac.kr)

5. Reference

Choi, Jonghwan, et al. "Improved prediction of breast cancer outcome by identifying heterogeneous biomarkers." *Bioinformatics* **33.22** (2017): 3619-3626.