Improved prediction of breast cancer outcome by identifying heterogeneous biomarkers (version 0.1)

Manual

April 25, 2017

Index_Toc480901230

1. Installation 3

2. Quick start 3

1) Run 3

2) Result 4

3. Method description of CPR.py 4

1) class CPR 4

2) method CPR.fit() 5

3) method CPR.validate() 5

4) method CPR.setParam() 5

5) method CPR.getParam() 6

6) method CPR.getRankedGenes() 6

7) method CPR.getBiomarkers() 6

4. Contact 6

5. Reference 6

1. Installation

CPR requires:
Python (>=2.6 or >=3.3),
NumPy (>=1.6.1),
SciPy (>=0.9),
scikit-learn (>=0.18)

If you already have a working installation of numpy, scipy, and scikit-learn:

(1) Download the CPR module from https://github.com/mathcom/CPR

(2) If successfully downloaded, user can find 'CPR.py'.

2. Quick start

1) Run

$ **python CPR.py -g** *NETWORK_FILE* **-r** *TRAINING_FILE* **[-e** *TEST_FILE***] [-o** *RESULT_FILE***] [***optional parameters***]**

Option	Name	Description
-g	*NETWORK_FILE* (required data)	Gene interactions in tab-delimited format. Its format is like below. ex) GENE1 GENE2 RPL37A RPS27A MRPL1 MRPS36 RFC3 SPRTN …
-r	*TRAINING_FILE* (required data)	Gene expressions dataset for fitting model. Tab-delimited format as below. (OSEVENT = 1:occurred and 0:censored. OSDURATION = years.) ex) PATIENT GSM110630 GSM110631 GSM110632 … OSEVENT 0 1 0 … OSDURATION 11.75 1.17 11.33 … DDR1 9.44016332 9.120552894 9.520718877 … RFC2 6.040801386 6.125876465 6.178279602 … HSPA6 5.987424508 6.456271426 6.725197108 … …
-e	*TEST_FILE* (optional data)	Gene expressions dataset to validate a fitted model. If *TEST_FILE* is not given, a model is 10 fold cross-validated with *TRAING_FILE*. Tab-delimited format as below. (OSEVENT = 1:occurred and 0:censored. OSDURATION = years.) ex) PATIENT GSM110635 GSM110636 GSM110637 … OSEVENT 1 1 0 … OSDURATION 0.92 3.08 11.17 … DDR1 9.246254311 9.421580476 9.255679766 … RFC2 5.40074029 6.214693638 6.24958879 … HSPA6 5.960663941 5.919019798 6.260029918 … …
-o	*RESULT_FILE* (optional data)	Results of model are saved in *RESULT_FILE*. If *RESULT_FILE* is not given, a summary of the results is put to stdout. A detail format of results is described in the 2) Result section.
-d	*Damping factor* (optional parameter)	Hyper-parameter of PageRank algorithm. A damping factor decides an influence of network data. Range = 0.0 ~ 1.0. (default = 0.7)
-n	*Number of biomarkers* (optional parameter)	The number of biomarkers used for outcome prediction. (default = 70)

2) Result

```
=====   RESULT   =================
Accuracy(=AUC): 0.677

=====   Biomarkers (70)   ========
ZNF681   RAC2      ZNF672   ARRB1    GNAI1    AURKB    EGFR      STAT5A   PTPN11   PIK3CA
RNPS1    CDK1      ZNF253   ZNF257   E2F1     UPF3B    SOS1      ITGB1    ZNF431   HSP90AA1
EED      CBL       HDAC3    TAF1     UBE2I    ZNF879   ZNF718    ACTG1    ZNF25    PPP1CC
BRCA1    SPI1      ZNF473   GNA11    CREBBP   RBM8A    RPS27     ZNF273   ZNF35    PLK1
POLR2L   ZNF626    ZNF430   POLR2J   POLR2I   YWHAG    ZNF92     GSK3B    TP53     ZNF572
CALM1    ZNF501    ZNF829   STAT3    POLR2C   HRAS     RAD21     ZNF519   LCK      FOS
GNAL     UBC       JUN      PRKACG   MAPK14   ZNF429   POLR2B    GNGT1    POLR2D   MAGOH

=====   Subnetwork (499)   ========
PTPN11   STAT3
JUN      POLR2D
LCK      STAT5A
HSP90AA1         PLK1
POLR2D   POLR2I
ZNF429   ZNF626
CALM1    EGFR
RAD21    SPI1
AURKB    PLK1
JUN      STAT3
MAGOH    UPF3B
PTPN11   SOS1
EED      POLR2B
HSP90AA1         SOS1
ZNF473   ZNF92
LCK      SOS1
ZNF572   ZNF829
ZNF501   ZNF92
...
```

* Each row of Subnetwork is an edge of the network

3. Method description of CPR.py

A user can import and utilize CPR.py in python. CPR.py contains one class and its seven methods.

1) class CPR

Clustering and PageRank-based classifier. For more detail, please refer to "Improved prediction for breast cancer outcome by identifying heterogeneous biomarkers" (under review)

Parameters	**dampingFactor** : float, optional (default=None) Damping factor is the hyper-parameter of PageRank algorithm. Range = 0.0 ~ 1.0 **n_biomarkers** : int, optional (default=70) User can controls the number of prognostic biomarkers. **n_clusters** : int, optional (default=2) The number of sample-clusters. To cluster samples, K-Means algorithm is applied on principal components of expression data. **n_pc** : int, optional (default=2) The number of principal components. To effectively cluster, PCA reduce the expression data. **t_degree** : float, optional (default=0.02) Threshold for degrees of biomarkers. To guarantee stable accuracy, CPR selects hub genes whose degrees are on the top (100* t_degree)% as biomarkers.

2) method CPR.fit()

Build a CPR classifier from the training dataset (data, label)

Parameters	**geneList** : list The list of all gene symbols in expression data. The order of genes must be equal to one for each sample in training data. **edgeList** : list The list of edges in Functional Interaction Networks, and all edges have tuple type. **data** : numpy.array, shape= [n_samples, n_genes] The training input samples. The order of genes for each sample must be equal to one of geneList. **label** : numpy.array, shape=[n_samples] The target values (class labels) as 1:poor prognosis and 0:good prognosis. **randomState** : int or None, optional (default=None) This parameter is used for scikit-learn functions. If int, random_state is the seed used by the random number generator. If None, the random number generator is the RandomState instance used by np.random.

3) method CPR.validate()

Predict class and return an accuracy, area under ROC curve (AUC), for fitted model.

Parameters	**geneList** : list The list of all gene symbols in expression data. The order of genes must be equal to one for each sample in test data. **data** : numpy.array, shape= [n_samples, n_genes] The training input samples. The order of genes for each sample must be equal to one of geneList. **label** : numpy.array, shape=[n_samples] The target values (class labels) as 1:poor prognosis and 0:good prognosis. **randomState** : int or None, optional (default=None) This parameter is used for scikit-learn functions. If int, random_state is the seed used by the random number generator. If None, the random number generator is the RandomState instance used by np.random.
Returns	**AUC** : float The accuracy is validated by AUC.

4) method CPR.setParam()

Set the parameters of model

Parameters	**dampingFactor** : float or None, optional (default=None) Damping factor is the hyper-parameter of PageRank algorithm. Range = 0.0 ~ 1.0. If None is given, this method skips this parameter. **n_biomarkers** : int or None, optional (default=None) The number of biomarkers for classifier. If None is given, this method skips this parameter. **n_clusters** : int or None, optional (default= None) The number of sample-clusters. If None is given, this method skips this parameter.

	n_pc : int or None, optional (default= None) The number of principal components. If None is given, this method skips this parameter. **t_degree** : float or None, optional (default= None) Threshold for degrees of biomarkers. If None is given, this method skips this parameter.

5) method CPR.getParam()

The parameters used in a model are provided via dictionary

Returns	**parameters** : dictionary The key is a name of parameter, and the value of key is a value of the parameter.

6) method CPR.getRankedGenes()

The genes prioritized by the modified PageRank in a model are provided via list

Returns	**genes** : list The genes are sorted in the order of ranking.

7) method CPR.getBiomarkers()

The biomarkers used in prediction are provided via list

Returns	**biomarkers** : list The biomarkers are sorted in the order of ranking.

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

J. Choi, S. Park, Y. Yoon and J. Ahn, Improved prediction of breast cancer outcome by identifying heterogeneous biomarkers, under review, 2017