Evaluation of Clustering Algorithms

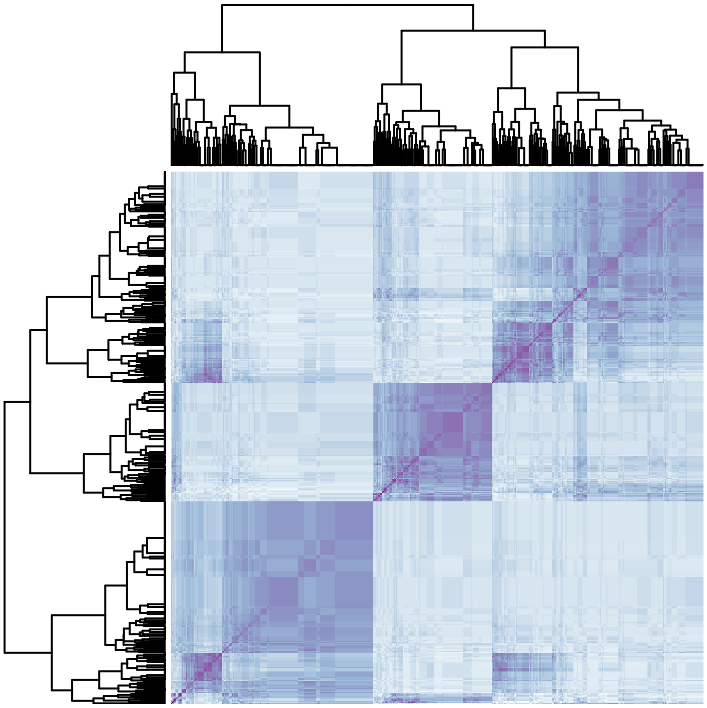
Derek Chiu

August 18, 2015

Contents

1 Introduction

Cluster analysis is the unsupervised learning method of assigning entities into different groups based on one or more of their attributes. The goal is to place similar objects together and separate dissimilar objects. For example, in genomics studies, we frequently try and cluster patient samples measured on a large number of molecular features. When we get a clustering assignment from an algorithm, we often want to evaluate its performance. Ideally, a good clustering algorithm is able to differentiate entities with no knowledge of the true class labels. In addition, we want the algorithm to arrive at a stable and optimal number of clusters. There are two main categories of clustering evaluation: **external evaluation** and **internal evaluation**.



The proportion of cases with at least 0.6 agreement is 0.2170958.

The confusion matrix is shown below, as well as different metrics for each class.

	C1	C2	C4	C5
C1	104	10	18	7
C2	1	37	10	1
C4	1	60	91	21
C5	3	0	16	109

	Sensitivity	Specificity	Pos Pred Value	Neg Pred Value	Prevalence	Detection Rate	Detection Prevalence	Balanced Accuracy
Class: C1	0.9541	0.9079	0.7482	0.9857	0.2229	0.2127	0.2843	0.931
Class: C2	0.3458	0.9686	0.7551	0.8409	0.2188	0.07566	0.1002	0.6572
Class: C4	0.6741	0.7684	0.526	0.8608	0.2761	0.1861	0.3538	0.7212
Class: C5	0.7899	0.9459	0.8516	0.9197	0.2822	0.2229	0.2618	0.8679

2 External Evaluation

External evaluation usually refers to the case when we compare our clustering assignments to true class labels, or have some gold standard to compare to. In applications, this might be the published clustering result. The downside of using external evaluation is that the reference classes may not be correctly clustered themselves, and we are treating these as the norm. None the less, we can explore a few metrics.

2.1 Kappa Statistic

The unadjusted kappa statistic is 0.5927477 and the weighted kappa statistic is 0.7717546 for the final meta consensus cluster.

2.2 Adjusted Rand Index

The larger the better.

Algorithms	ARI
NMF (Divergence)	0.4799
NMF (Euclidean)	0.4435
PAM (Spearman)	0.427
HC (Diana)	0.4221
KM (Spearman)	0.4049
PAM (Euclidean)	0.3434
HC (Euclidean)	0.3275
KM (Euclidean)	0.2559
PAM (MI)	0.07465
KM (MI)	0.07369

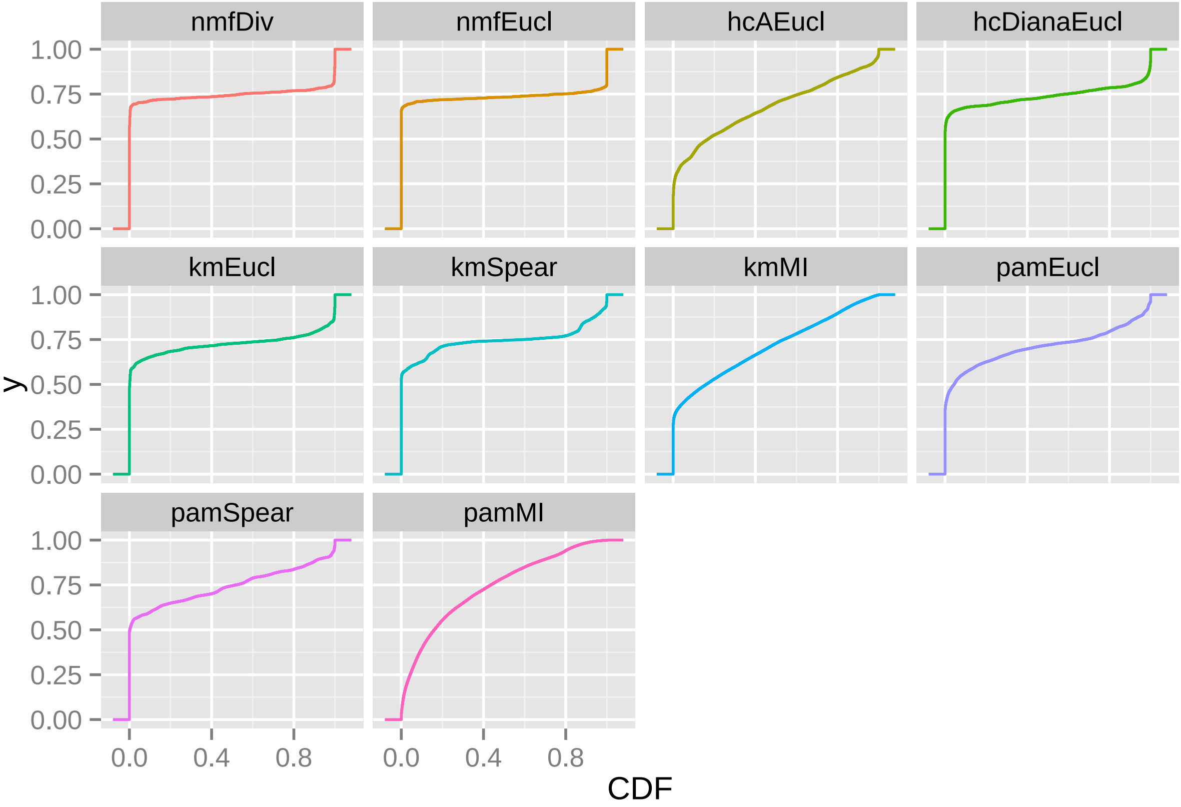
2.3 Mutual Information

The larger the better.

Algorithms	MI
NMF (Divergence)	0.6723
NMF (Euclidean)	0.6416
PAM (Spearman)	0.621
HC (Diana)	0.5976
KM (Spearman)	0.5889
PAM (Euclidean)	0.546
HC (Euclidean)	0.481
KM (Euclidean)	0.4598
PAM (MI)	0.1652
KM (MI)	0.1169

The mutual information for the meta consensus clustering is 0.5987537.

2.4 Cumulative Distribution Function



2.5 Proportion of Ambiguous Clusters

The lower the better.

Algorithms	PAC
NMF (Euclidean)	0.1406
NMF (Divergence)	0.3267
HC (Diana)	0.3829
KM (Spearman)	0.425
KM (Euclidean)	0.4398
PAM (Spearman)	0.4748
PAM (Euclidean)	0.5984
KM (MI)	0.7167
HC (Euclidean)	0.7836
PAM (MI)	0.9638

The PAC for the meta consensus matrix is 0.7114804.

3 Internal Evaluation

3.1 Davies-Bouldin Index

For DBI, the lower the better.

Algorithms	DBI
HC (Euclidean)	1.689
NMF (Euclidean)	1.702
NMF (Divergence)	1.702
PAM (Spearman)	1.72
PAM (Euclidean)	1.725
KM (Spearman)	1.725
HC (Diana)	1.731
KM (Euclidean)	1.741
PAM (MI)	1.935
KM (MI)	1.972

3.2 Dunn Index

For DI, the larger the better.

Algorithms	DI
HC (Euclidean)	1.04
PAM (Euclidean)	1.028
NMF (Euclidean)	1.003
NMF (Divergence)	0.999
KM (Spearman)	0.9926
HC (Diana)	0.9866
PAM (MI)	0.9821
PAM (Spearman)	0.9492
KM (MI)	0.9262
KM (Euclidean)	0.9238

3.3 Silhouette Average Width

For SAW, the larger the better.

Algorithms	SAW
HC (Euclidean)	0.1226
PAM (Euclidean)	0.1171
NMF (Divergence)	0.1111
NMF (Euclidean)	0.1084
PAM (Spearman)	0.1069
KM (Spearman)	0.1059
HC (Diana)	0.09062
KM (Euclidean)	0.08707
PAM (MI)	-0.0061
KM (MI)	-0.007504

3.4 Rousseeuw's Silhouette

For Rousseuw's Silhouette internal cluster quality index (RS), the larger the better.

Algorithms	RS
HC (Euclidean)	0.1226
PAM (Euclidean)	0.1171
NMF (Divergence)	0.1111
NMF (Euclidean)	0.1084
PAM (Spearman)	0.1069
KM (Spearman)	0.1059
HC (Diana)	0.09062
KM (Euclidean)	0.08707
PAM (MI)	-0.0061
KM (MI)	-0.007504

3.5 C-Index

For CI, the lower the better.

Algorithms	CI
KM (Euclidean)	0.2816
NMF (Divergence)	0.3023
PAM (Euclidean)	0.31
HC (Euclidean)	0.3129
HC (Diana)	0.3212
NMF (Euclidean)	0.3369
PAM (MI)	0.3376
PAM (Spearman)	0.3489
KM (MI)	0.3529
KM (Spearman)	0.356

3.6 Baker and Hubert Index

For BHI, the larger the better.

Algorithms	BHI
HC (Euclidean)	2.051
PAM (Euclidean)	1.754
HC (Diana)	1.746
PAM (Spearman)	1.715
KM (Euclidean)	1.676
NMF (Euclidean)	1.666
KM (Spearman)	1.627
NMF (Divergence)	1.617
KM (MI)	-3.142
PAM (MI)	-3.358

3.7 Calinski-Harabasz Index

For CHI, the larger the better.

Algorithms	CHI
NMF (Divergence)	80.36
NMF (Euclidean)	79.26
KM (Spearman)	78.71
PAM (Spearman)	77.19
PAM (Euclidean)	75.71
KM (Euclidean)	75.38
HC (Diana)	74.95
HC (Euclidean)	66.49
PAM (MI)	13.23
KM (MI)	7.322

3.8 Summary

Here is a summary of all the indices for each algorithm, in unsorted order.

Algorithms	ARI	MI	PAC	DBI	DI	SAW	RS	CI	BHI	CHI
HC (Diana)	0.4221	0.5976	0.3829	1.731	0.9866	0.09062	0.09062	0.3212	1.746	74.95
HC (Euclidean)	0.3275	0.481	0.7836	1.689	1.04	0.1226	0.1226	0.3129	2.051	66.49
KM (Euclidean)	0.2559	0.4598	0.4398	1.741	0.9238	0.08707	0.08707	0.2816	1.676	75.38
KM (MI)	0.07369	0.1169	0.7167	1.972	0.9262	-0.007504	-0.007504	0.3529	-3.142	7.322
KM (Spearman)	0.4049	0.5889	0.425	1.725	0.9926	0.1059	0.1059	0.356	1.627	78.71
NMF (Divergence)	0.4799	0.6723	0.3267	1.702	0.999	0.1111	0.1111	0.3023	1.617	80.36
NMF (Euclidean)	0.4435	0.6416	0.1406	1.702	1.003	0.1084	0.1084	0.3369	1.666	79.26
PAM (Euclidean)	0.3434	0.546	0.5984	1.725	1.028	0.1171	0.1171	0.31	1.754	75.71
PAM (MI)	0.07465	0.1652	0.9638	1.935	0.9821	-0.0061	-0.0061	0.3376	-3.358	13.23
PAM (Spearman)	0.427	0.621	0.4748	1.72	0.9492	0.1069	0.1069	0.3489	1.715	77.19

4 Ranked Indices

The table below shows the ranking of algorithms for performance on a clustering index, for each index. There is an additional column that shows the propoportion of indices where an algorithm was ranked **first or second**.

Algorithms	ARI	MI	PAC	DBI	DI	SAW	RS	CI	BHI	CHI	Top
HC (Euclidean)	7	7	9	1	1	1	1	4	1	8	50%
NMF (Divergence)	1	1	2	3	4	3	3	2	8	1	50%
NMF (Euclidean)	2	2	1	2	3	4	4	6	6	2	50%
PAM (Euclidean)	6	6	7	5	2	2	2	3	2	5	40%
KM (Euclidean)	8	8	5	8	10	8	8	1	5	6	10%
HC (Diana)	4	4	3	7	6	7	7	5	3	7	0%
KM (MI)	10	10	8	10	9	10	10	9	9	10	0%
KM (Spearman)	5	5	4	6	5	6	6	10	7	3	0%
PAM (MI)	9	9	10	9	7	9	9	7	10	9	0%
PAM (Spearman)	3	3	6	4	8	5	5	8	4	4	0%

If we were to conduct a meta consensus clustering, a weight for each algorithm needs to be assigned. One such way of doing so is using the inverse rank sums. We can sum the ranks for each algorithm, and assign higher weight for consistently high ranked methods (1st or 2nd) and vice versa. This is shown below:

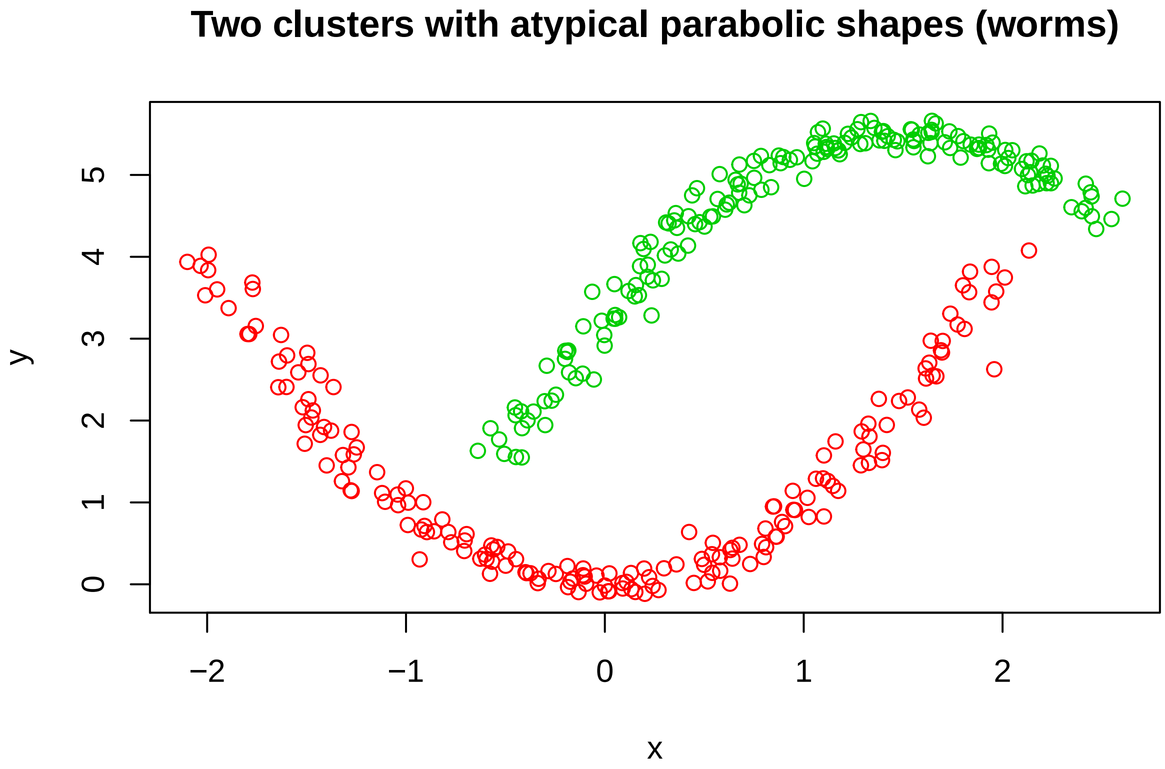
Algorithms	Top	Sum	Weight
NMF (Divergence)	50%	28	16.99%
NMF (Euclidean)	50%	32	14.87%
HC (Euclidean)	50%	40	11.89%
PAM (Euclidean)	40%	40	11.89%
PAM (Spearman)	0%	50	9.52%
HC (Diana)	0%	53	8.98%
KM (Spearman)	0%	57	8.35%
KM (Euclidean)	10%	67	7.1%
PAM (MI)	0%	88	5.41%
KM (MI)	0%	95	5.01%

5 Simulations

To confirm the clustering results from using the TCGA dataset, we need to try out the algorithms on a few simulated datasets, designed to test the robustness of each method. The `clusterSim` package provides very good built-in examples to use.

5.1 Worms Dataset

The following two-cluster dataset is our first simulation:



Two clusters with atypical parabolic shapes (worms)