Class Aware Exemplar Discovery from Microarray Gene Expression Data

Shivani Sharma, Abhinna Agrawal, and Dhaval Patel(✉)

Department of Computer Science and Engineering,
Indian Institute of Technology-Roorkee, Roorkee, India
{pannicle,abhinnaagrawal}@gmail.com,
patelfec@iitr.ac.in

Abstract. Given a dataset, exemplars are subset of data points that can represent a set of data points without significance loss of information. Affinity propagation is an exemplar discovery technique that, unlike k–centres clustering, gives uniform preference to all data points. The data points iteratively exchange real–valued messages, until clusters with their representative exemplar become apparent.

In this paper, we propose a Class Aware Exemplar Discovery (CAED) algorithm, which assigns preference value to data points based on their ability to differentiate samples of one class from others. To aid this, CAED performs class wise ranking of data points, assigning preference value to each data point based on its class wise rank. While exchanging messages, data points with better representative ability are more favored for being chosen as exemplar over other data points.

The proposed method is evaluated over 18 gene expression datasets to check its efficacy for selection of relevant exemplars from large datasets. Experimental evaluation exhibits improvement in classification accuracy over affinity propagation and other state-of-art feature selection techniques. Class Aware Exemplar Discovery converges in lesser iterations as compared to affinity propagation thereby dropping the execution time significantly.

Keywords: Gene · Exemplar and clustering

1 Introduction

With the advent of microarray technology, simultaneous profiling of thousands of gene expression across multiple samples in a single experiment was made possible. The microarray technology generates huge amount of gene expression data whose competitive analysis is challenging. Moreover, it has been observed that gene expression datasets has large number of uninformative and redundant features which increases complexity of classification algorithms [1, 11–13].

To circumvent these problems many feature selection techniques are being proposed. The purpose of feature selection is extracting relevant features from the observed data which improves results of machine learning models. Compared with the dimensionality reduction techniques like Principal Component Analysis (PCA) and

N. Kumar and V. Bhatnagar (Eds.): BDA 2015, LNCS 9498, pp. 244–257, 2015.
DOI: 10.1007/978-3-319-27057-9_17

Linear Discriminate Analysis (LDA), feature selection algorithms only select the relevant subset of features instead of altering the original features. The selected relevant genes, also known as "biomarkers", find their application in medicine for discovery of new diseases, development of new pharmaceuticals [2, 12] etc. Existing work on feature subset selection from gene expression data can be categorized as (i) classification and (ii) clustering based approaches. The classification based approaches like ReliefF [3] and Correlation based Feature Selection [4], ranks features based on their intrinsic properties and select subset of top ranked features. The clustering based approaches like k-medoids [5] and affinity propagation [6] clusters the features based on their similarity with each other. The feature set is reduced to the representative of each cluster.

Affinity propagation proposed by Frey and Dueck in [6] for feature subset selection identifies subset of features which can represent the dataset. Such features are called exemplars. Affinity propagation takes similarity between features as input. Instead of specifying the number of clusters, a real number called preference value for all features is also passed as input to affinity propagation. The number of identified exemplars is influenced by the preference value. Larger the preference value more the clusters are formed. Features exchange real- valued messages until clusters with their representative exemplars emerge. Affinity propagation has found its application in the machine learning community, computational biology and computer vision. However, aforementioned approach gives uniform preference to all features and messages are exchanged iteratively between features irrespective of the capability of features to differentiate samples of one class from samples of other classes.

In this paper, we propose a Class Aware Exemplar Discovery (CAED) algorithm which calculates class wise ranking for all features and incorporates this information while assigning preference value to features. The features are clustered by exchanging two types of messages viz. responsibility and availability iteratively. The messages are exchanged in a way that the features ranked higher in class wise ranking are favored over the feature ranked lower which leads to better exemplar discovery.

We evaluated correctness of our approach by conducting experiments on 18 publicly available microarray gene expression datasets. We recorded classification accuracy as our performance metric. Improvement in classification accuracy over affinity propagation of three classifiers namely Support Vector Machine, Naive Bayes and C4.5 Decision Tree is achieved for 16, 17 and 13 datasets respectively.

2 Overview of Our Approach

The workflow of our approach is shown in Fig. 1. Gene expression matrix is transformed into similarity matrix using a distance measure. The diagonal values of similarity matrix also called preferences are updated using class aware ranking of features. The updated matrix is used for class aware clustering. The representative from each cluster called exemplar is extracted and the reduced set of features is evaluated over classifiers.

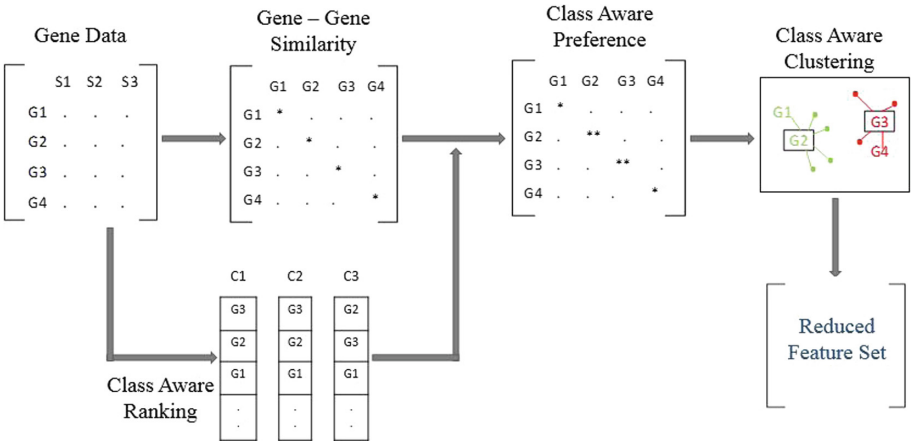


Fig. 1. Workflow of Class Aware Exemplar Discovery (CAED)

2.1 Gene Data

We selected 18 publicly available microarray datasets (available at http://faculty.iitr.ac.in/∼patelfec/index.html) for experimental evaluation of Class Aware Exemplar Discovery algorithm. Microarray datasets are gene expression matrices, where expression value for each gene is measured over different samples. Generally total numbers of samples are very few compared to the features.

2.2 Gene-Gene Similarity

For a gene expression matrix $D_{n\times m}$, with n features and m samples, similarity between every two features is calculated and stored in similarity matrix $S_{n\times n}$. The similarity $s(i, k)$ indicates how well the feature with index k is suited to be the exemplar for feature$i$. The aim is to maximize the similarity, so we take negative of the distance between each feature. We used negative of Euclidean distance as the similarity measure for experimental evaluation.

2.3 Class Aware Preference

The affinity propagation algorithm takes similarity matrix and a real number called preference for each feature as input. For a similarity matrix $S_{n\times n}$, the value $s(i, i)$ where $i = \{1, 2, 3\ldots, n\}$ is the preference value. These values are called preferences since the feature i with larger values $s(i, i)$ is more likely to be chosen as exemplar. The number of identified exemplars (number of clusters) is influenced by the values of input preference. Larger the value more the clusters are formed. The preference value can be uniform or non-uniform. If all features are equally suitable as exemplars, the preference value is set to a common value. The preference value can be set as any number in the range of $\min_{js.t.j\neq i} s(i,j)$ to $\max_{js.t.j\neq i} s(i,j) : i, j = \{1, 2, 3\ldots, n\}$.

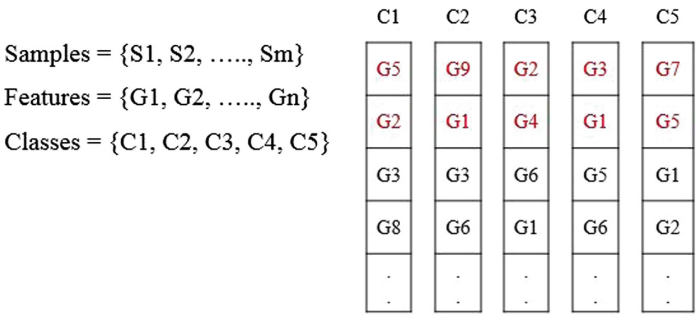


Fig. 2. Depiction of class wise ranking of features

We propose assignment of preference value to a feature based on its ability to distinguish samples of one class from other classes. To aid this, we do class wise ranking of features using p-metric [1] by one versus all strategy. Top 0.015 % of features from each class is selected and are assigned preference value zero thereby increasing their probability of being chosen as exemplar.

Figure 2 depicts class wise ranking of features where each feature is assigned multiple ranks, one for each class. The high ranked features of each class are highlighted.

The other features are assigned uniform preference value i.e. $\underset{js.t.j\neq i}{median} s(i,j) : i.j = \{1, 2, 3 \ldots, n\}$. Figure 3 shows how accuracy of classifiers namely support vector machine (SVM), Naïve Bayes (NB) and C4.5 decision tree (DT) varied by changing the count of features which are assigned high value.

We observed that by selecting more than 0.015 % of features from each class no further improvement in classification accuracy of the classifiers was observed.

2.4 Class Aware Message Passing

The similarity matrix with class aware preference values is passed to affinity propagation for exemplar discovery. Affinity propagation is a message passing based clustering algorithm. Affinity propagation iteratively transmits real-valued messages among features until a good set of clusters with their representative exemplar emerge. The messages exchanged are of two kinds viz.

Responsibility message denoted as $r(i,k)$ is sent from feature i to candidate exemplar point k. It indicates the collected evidence for how appropriate feature k is to be chosen as exemplar for feature i.

Availability message denoted as $a(i,k)$ is sent from candidate exemplar point k to feature i.. It indicates the collected evidence of how appropriate it would be, for feature i to choose feature k as its exemplar.

We propose class aware message passing algorithm where strength of message exchanged between two features depends on their ability to discriminate samples among different classes.

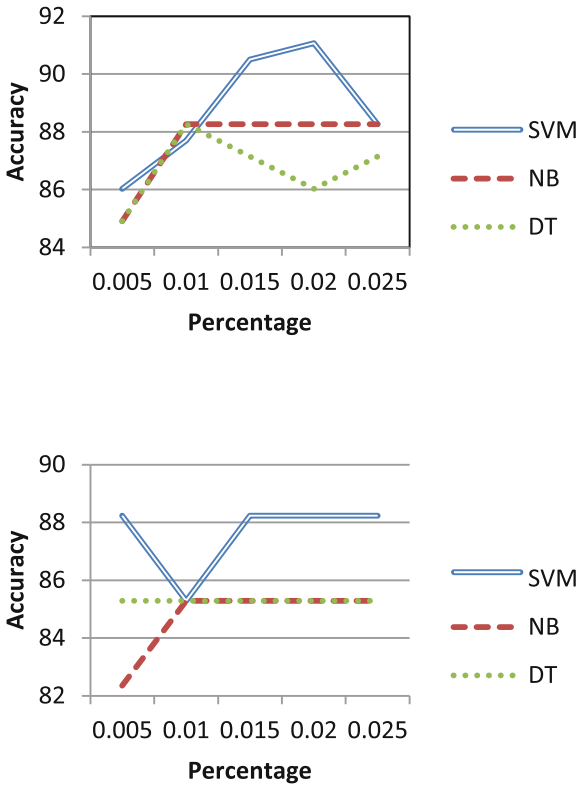


Fig. 3. Change in accuracy of classifiers by changing the percentage of high preference features on 2 datasets

Each iteration of affinity propagation has three steps:

Step 1. Calculating responsibilities r(i, k) given the availabilities
Step 2. Updating all availabilities a(i, k) given the responsibilities
Step 3. Combining the responsibilities and availabilities to generate clusters

Calculating Responsibilities *r(i,k)* Given the Availabilities: For a dataset $D_{n\times m}$ with n features, m samples and p classes we calculate class wise rank for each feature using p–metric. Class wise rank of each feature i is denoted as $R_i = \{C_1, C_2, C_3 \ldots ..., C_p\}$ where $i = \{1, 2, 3..n\}$ and C_1 is rank of feature i for class 1. To calculate $r(i,k)$ we evaluate R_i and R_k. If rank of a feature i for class j is less than $\frac{n}{2}$ i.e. $C_j \leq \frac{n}{2}$, it lies in the upper half of the ranking and we denote it as H, else it lies in lower half and denoted as L. Hence, the ranking of feature i is changed to suppose $R_i = \{H, H, L, \ldots L\}$. Similarly ranking of feature k is changed to suppose $R_k = \{L, H, L \ldots, L\}$. The strength of responsibility message sent from i to k is governed by the occurrence of H in R_i and R_k.

Further calculation of responsibility $r(i,k)$ can be divided into two sections depending on the count of class labels of the dataset.

Two Class Label Dataset. For datasets with 2 classes, the class wise p-metric ranking of features is identical for both classes. The values in R_i and R_k can be of four kinds as listed below

$R_i = \{\text{HH}\} = R_k = (\text{HH})$ – feature k should be assigned responsibility of serving as exemplar for feature i. Set $s(i,k) = \max_{vs.t.v \neq u} s(u,v)$ where $u, v \in \{1, 2, 3 \ldots n\}$
$R_i = (LL) R_k = (LL)$ – No change in $s(i,k)$
$R_i = (\text{HH}) R_k = (\text{LL})$ - No change in $s(i,k)$
$R_i = (\text{LL}) R_k = (\text{HH})$ – feature k should be assigned responsibility of serving as exemplar for feature i. Set $s(i,k) = \max_{vs.t.v \neq u} s(u,v)$ where $u, v \in \{1, 2, 3 \ldots n\}$.

Multi Class Label Dataset. Suppose $R_i = \{H, H, H, \ldots L\}$ and $R_k = \{H, H, L, \ldots L\}$ is class wise ranking for features i and k respectively. Count of occurrences of H in both sets is stored as H_i and H_k. If $H_k \geq H_i$ then, feature k should be assigned high responsibility of serving as exemplar for feature i. Set $s(i,k) = \max_{vs.t.v \neq u} s(u,v)$ where $u, v \in \{1, 2, 3 \ldots n\}$.

Then, the value of responsibilities is calculated using equation:

$$r(i,k) \leftarrow s(i,k) - \max_{k' s.t. k' \neq k} \{a(i,k') + s(i,k')\}$$

Setting $s(i,k)$ as maximum of all the similarities increases the strength of responsibility message sent from feature i to candidate exemplar point k.

Initially availabilities are set to zero. For the first iteration $r(i,k)$ is similarity between feature i and k as its exemplar, reduced by the maximum similarity between i and other features. Later, when features highly similar to i are assigned to some other exemplar, their availabilities as a candidate exemplar for i falls below zero. Such negative value will affect the similarity value $s(i,k')$ in the above equation.

Updating all availabilities $a(i, k)$ given the responsibilities. Availabilities are Calculated as:

$$a(i,k) \leftarrow min\{0, r(k,k) + \sum_{i' s.t. i' \notin \{i,k\}} \max\{0, r(i',k)\}\}$$

The value of availability can be zero or negative. Zero value indicates that k is available and k can be assigned as exemplar to i. If the value of $a(i,k)$ is negative, it indicates that k belongs to some other exemplar and it is not available to become exemplar for i.

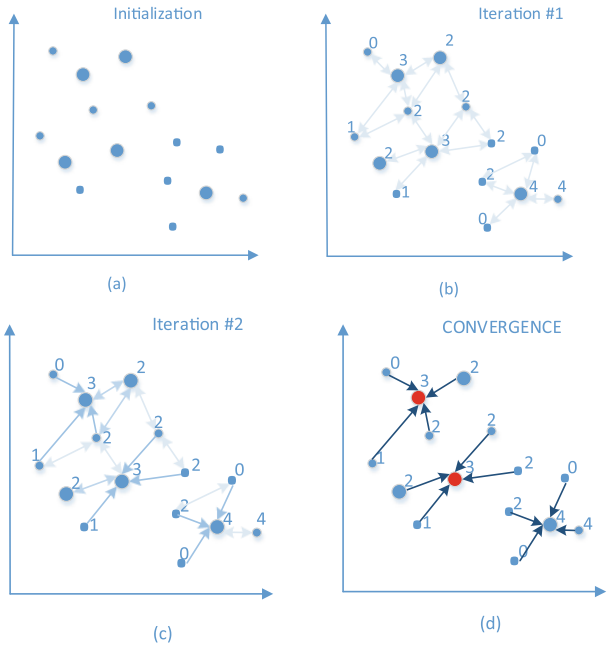


Fig. 4. (a) Two –dimensional features, sized according to their class aware preference value. (b) and (c) Messages are exchanged between features. The number associated with each feature corresponds to its count of occurrence in upper half of class wise ranking. The darkness of arrow directed from point i to point k corresponds to the strength of message that point i belongs to exemplar point k. (d) Clusters formed with their representative exemplar.

Combining the Responsibilities and Availabilities to Generate Clusters. For every iteration, the point j that maximizes $r(i,j) + a(i,j)$ is regarded as exemplar for point i. The algorithm terminate when changes in the message falls previously set threshold. Final result is list of all the exemplars which is an optimal subset of features. Figure 4 shows dynamics of Class Aware Exemplar Discovery algorithm when applied on 15 two-dimensional features with 4 class labels. Initially, features are sized according to their class aware preference. Each feature is numbered according to the count of classes in which it is high ranked. Class aware messages are exchanged between features. When convergence condition is satisfied, features are clustered with each cluster represented by red colored exemplar.

Algorithm 1 presents the steps followed by Class Aware Exemplar Discovery to select optimal set of features. The input to the algorithm is gene expression matrix with n specifying number of genes, m specifying number of samples and c classes. Three matrices namely responsibility, availability and similarity are initialized with zero. First we calculate negative of Euclidean distance between every pair of feature and store it in similarity matrix (Line 1). Next, we calculate class wise rank of all features (Line 2). We use N_c to denote ranked list of genes that is obtained for class c (Lines 3–11).

Class Aware Preference (Line 4) - We select top 0.015 % of features from N_c and assign them high preference value which is zero. To rest of the features median of similarities is assigned as preference value.

ALGORITHM 1. Class Aware Exemplar Discovery

Input: Gene Expression Matrix D_{nxm}
C = set of class labels $\{c_1, c_2, c_3, \dots c_p\}$
n = Number of genes
m = Number of samples
Initialize: Similarity matrix $s[n][n]$
Responsibility matrix $r[n][n]$
Availability matrix $a[n][n]$
Output: ESet = List of exemplars

1. **Calculate** S_{nxn} using D_{nxm}
2. Let N_c, be ranked list of genes from D_{nxm} w.r.t class label $c \in C$;
3. **for** each N_c
4. **Select** 0.015% of top ranked genes; store in N_c';
5. **end** for
6. **for** each gene $i \in N_c'$
7. **set** $s(i,i) \leftarrow 0$;
8. **end** for
9. **for** each gene $i \notin N_c'$
10. **set** $s(i,i) \leftarrow \underset{js.t.j \neq i}{median} s(i,j)$;
11. **end** for
12. **for** all $i, j \in \{1,2,3 \dots, n\}$
13. **set** $a[i][j] = 0$;
14. **end** for
15. **while** the convergence conditions are not satisfied
16. **for** each i, k
17. **Obtain** class wise rank R_i and R_k of gene i and $k$for p classes represented as $\{C_1, C_2, C_3 \dots \dots, C_p\}$;
18. **for** each class wise rank C_j
19. **if** $C_j \leq \frac{n}{2}$
20. Set $C_j = H$;
21. **else** $C_j = L$;
22. **end** if
23. **Let** H_i and H_k be count of H in R_i and R_k respectively;
24. **if** c == 2 and $R_k == HH$
25. **Set** $s(i,k) \leftarrow \underset{vs.t.v \neq u}{max} s(u,v)$ where $u, v \in \{1,2,3 \dots n\}$;
26. **Else** If $c > 2$ and $H_k \geq H_i$
27. **then,** Set $s(i,k) \leftarrow \underset{vs.t.v \neq u}{max} s(u,v)$ where $u, v \in \{1,2,3 \dots n\}$;
28. **end** for
29. find the $k' \neq k$ that maximizes $a[i][k'] + s[i][k']$
30. $r(i,k) \leftarrow s(i,k) - \{a(i,k') + s(i,k')\}$.
31. $sum = 0$;
32. **for** each $i' \notin \{i,k\}$
33. **if** $r[i'][k] > 0$
34. **do** $sum \leftarrow sum + r[i'][k]$;
35. **end** if
36. **end** for
37. **if** $i == k$
38. **then** $a[i][k] \leftarrow sum$;
39. **else if** $r[k][k] + sum < 0$
40. **then** $a[i][k] \leftarrow r[k][k] + sum$;
41. **else** $a[i][k] \leftarrow 0$
42. **end** if
43. **end** for
44. **for** each i
45. Find the k that maximizes $a[i][k] + r[i][k]$;
46. Set k as the exemplar of i;
47. Put point k in Eset;
48. **end** for
49. **end** while
50. **return** Eset

Class Aware Message Passing (Lines 15–43). Then features are clustered by passing real valued class aware messages. The algorithm terminates when change in messages falls below previously set threshold. Exemplars are returned and can be used to accurately predict the class label of unlabeled samples.

3 Experimental Evaluation

We performed three set of experiments to evaluate the performance of our approach. In the first experiment we compared the performance of three different classifiers using features selected by CAED against the features selected by affinity propagation. In second experiment, we compared performance of CAED with CFS [4] for feature selection with greedy search strategy. WEKA [7] provided us with implementations of CFS. The third experiment compared the performance of classifiers using all the features versus features extracted by CAED. Details of all the three experiments are discussed in subsequent chapters. The experiments were carried out on 3.4 GHz Intel i7 CPU with 8 GB RAM machine running Windows-based operating system.

3.1 Description of Experimental Datasets

We performed experimental evaluation of Class Aware Exemplar Discovery over 18 publicly available microarray gene expression datasets [8–11]. Table 1 describes the

Table 1. Description of datasets

S.no.	Data set name	Attributes	Samples	Classes
1	chowdary-2006_database1	183	104	2
2	alizadeh-2000-v1	1096	42	2
3	nutt-2003-v3_database1	1153	22	2
4	pomeroy-2002-v1_database1	858	34	2
5	nutt-2003-v2_database1	1071	28	2
6	west-2001_database1	1199	49	2
7	meduloblastomiGSE468	1466	23	2
8	chen-2002	86	179	2
9	breast_A	1214	98	3
10	DLBCL_B	662	180	3
11	golub-1999-v2_database1	1869	72	3
12	liang-2005	1412	37	3
13	dyrskjot-2003_database1	1204	40	3
14	DLBCL_A	662	141	3
15	bredel-2005	1740	50	3
16	risinger-2003	1772	42	4
17	tomlins-2006-v2	1289	92	4
18	Breast_B	1214	49	4

datasets used for experimental study. The datasets used for experimental evaluation are picked from various cancer related research work.

3.2 Comparison of Class Aware Exemplar Discovery with Affinity Propagation

We evaluated the classification accuracy of three state-of- art classifiers namely support vector machine (SVM), Naïve Bayes (NB) and C4.5 decision tree (DT) using the exemplars generated by Affinity Propagation (AP) and the exemplars generated by Class Aware Exemplar Discovery. The classification accuracy is calculated using 10 fold cross validation approach.

To visualize the results we performed "Win-Loss Experiment". If the classification accuracy of CAED is better than baseline approach the result is declared as win, if the classification accuracy has degraded the result is declared as loss otherwise declared as draw. Figure 5 shows results of "Win-Loss Experiment", obtained when classification accuracy of three classifiers using exemplars generated by affinity propagation is compared against Class Aware Exemplar Discovery.

We observed that Class Aware exemplar discovery generates less exemplar in comparison to Affinity propagation. Figure 6 shows drop in count of clusters (i.e., number of examples) from affinity propagation to CAED.

We also observed that Class Aware Exemplar Discovery converges in lesser message passing iteration in comparison to affinity propagation. This reduces the execution time tremendously. Figure 7 shows drop in execution time, Y axis is measured in seconds.

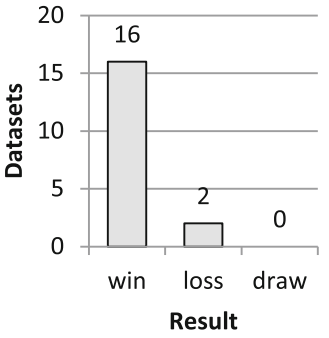
3.3 Comparison of Class Aware Exemplar Discovery with Standard Feature Subset Selection Techniques

We compare the effectiveness of CAED with features generated using CFS (Correlation based Feature Selection). The maximum achievable 10 fold cross validation classification accuracy is recorded as performance metric. Figure 8 shows results of "Win-Loss Experiment", obtained when classification accuracy of three classifiers using feature subset generated by CFS is compared against exemplars generated by Class Aware Exemplar Discovery.

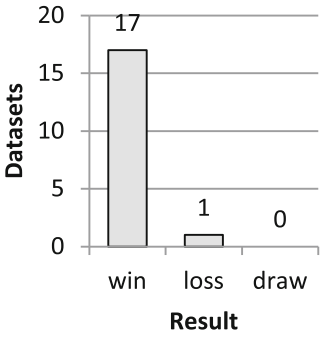
3.4 Comparison of Class Aware Exemplar Discovery with All Features

We evaluated the performance of three classifiers support vector machine (SVM), Naïve Bayes (NB) and C4.5 decision tree (DT) using all features of the 18 datasets. We compared these results with the classification accuracy obtained using features produced by Class Aware Exemplar Discovery.

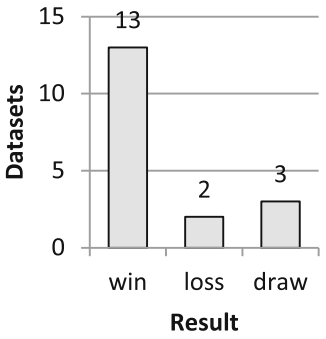
Figure 9 shows results of "Win-Loss Experiment" obtained when classification accuracy of three classifiers using all features is compared against Class Aware Exemplar Discovery.



(a) SVM



(b) Naïve Bayes



(c) Decision Tree

Fig. 5. Win – loss depiction of CAED versus affinity propagation carried over classifiers (a) support vector machine (b) Naïve Bayes (c) C4.5 decision tree

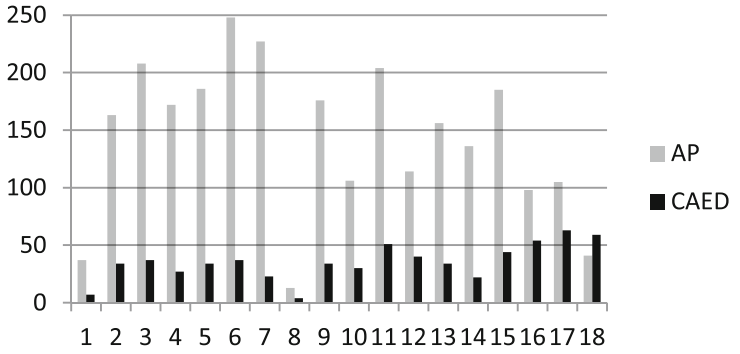


Fig. 6. Drop in count of clusters in 18 dataset

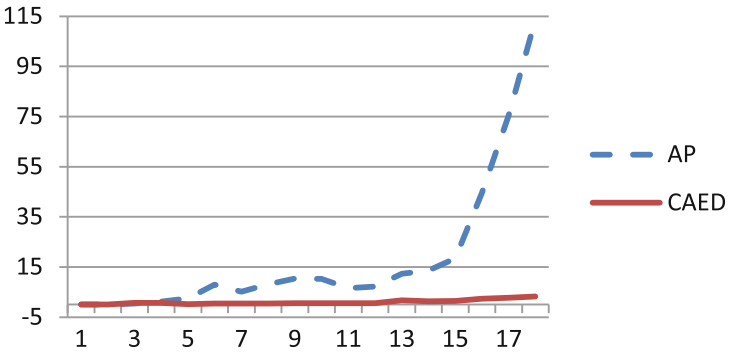


Fig. 7. Drop in execution time

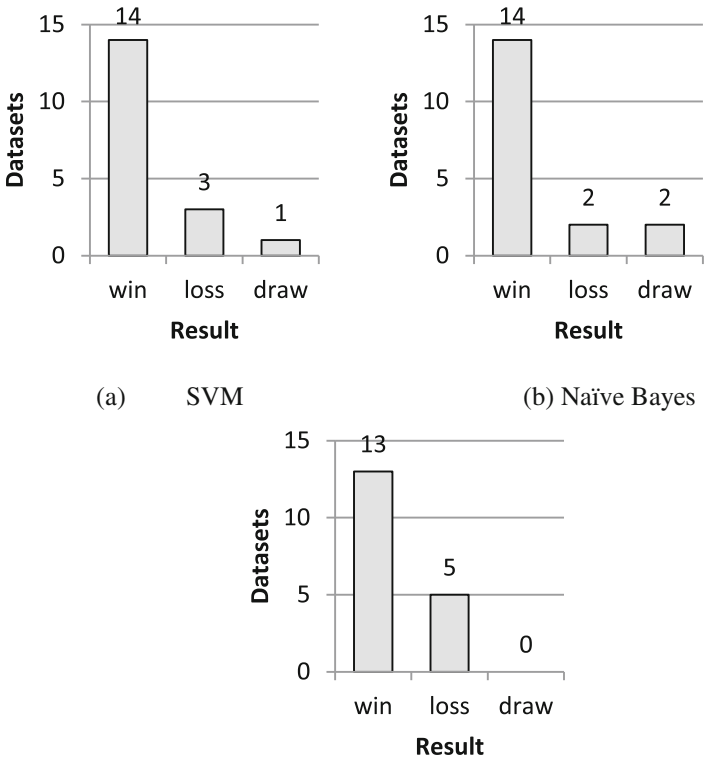
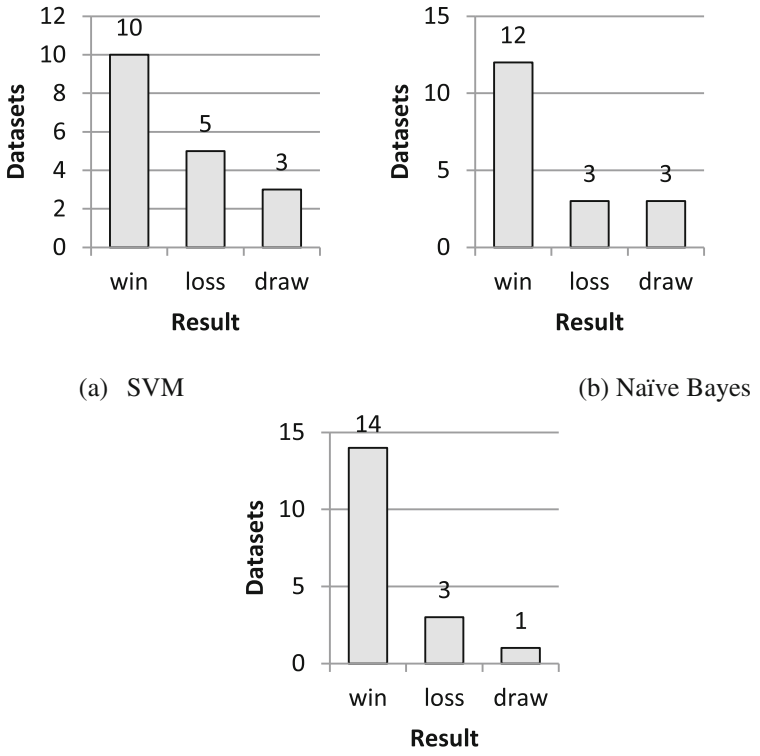


Fig. 8. Win – loss depiction of CAED over CFS carried over classifiers (a) support vector machine (b) Naïve Bayes (c) C4.5 decision tree



(a) SVM

(b) Naïve Bayes

(c) C4.5 Decision Tree

Fig. 9. Win- Loss depiction of CAED over all features carried over classifiers (a) support vector machine (b) Naïve Bayes (c) C4.5 decision tree

4 Conclusions

Gene expression datasets have large number of features. For effective application of any learning algorithm on gene expression datasets, feature subset selection is required. In this paper, we proposed a class aware clustering based feature subset selection technique. Our approach quantifies the ability of a feature to distinguish samples of one class from other classes. We use this value to influence the message passing procedure of affinity propagation. We observed that our approach leads to more relevant selection of features in less time in comparison to existing approach using the similar strategy for feature selection. We evaluated the effectiveness of our approach on 18 real world cancer datasets.

We evaluated Class Aware Exemplar Discovery against affinity propagation. Experiments have shown have shown that our technique outruns Affinity propagation in terms of classification accuracy. In comparison to affinity propagation, CAED converges in less number of iteration leading to huge drop in execution time.

We also evaluated the feature set generated by CAED against state-of-art feature selection technique. Experimental results have shown CAED gives better classification accuracy for all the classifiers used. Motivated by recent growth in parallel computing [13] and NVIDIA CUDA Research Center Support, we are developing a GPU based parallel algorithm for CAED. We are also working on improving the readability of mathematical symbol in the printed version of the submitted paper.

References

1. Inza, I., Larrañaga, P., Blanco, R., Cerrolaza, A.J.: Filter versus wrapper gene selection approaches in DNA microarray domains. Artif. Intell. Med. **31**(2), 91–103 (2004)
2. De Abreu, F.B., Wells, W.A., Tsongalis, G.J.: The emerging role of the molecular diagnostics laboratory in breast cancer personalized medicine. Am. J. Pathol. **183**(4), 1075–1083 (2013)
3. Kononenko, I., Šimec, E., Robnik-Šikonja, M.: Overcoming the myopia of inductive learning algorithms with RELIEFF. Appl. Intell. **7**(1), 39–55 (1997)
4. Hall, M.A.: Correlation-based feature selection for machine learning. Doctoral dissertation, The University of Waikato (1999)
5. Kashef, R., Kamel, M.S.: Efficient bisecting *k*-medoids and its application in gene expression analysis. In: Campilho, A., Kamel, M. (eds.) ICIAR 2008. LNCS, vol. 5112, pp. 423–434. Springer, Heidelberg (2008)
6. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. Science **315** (5814), 972–976 (2007)
7. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. ACM SIGKDD Explor. Newsl. **11**(1), 10–18 (2009)
8. De Souto, M.C., Costa, I.G., de Araujo, D.S., Ludermir, T.B., Schliep, A.: Clustering cancer gene expression data: a comparative study. BMC Bioinf. **9**(1), 497 (2008)
9. Foithong, S., Pinngern, O., Attachoo, B.: Feature subset selection wrapper based on mutual information and rough sets. Expert Syst. Appl. **39**(1), 574–584 (2012)
10. Mramor, M., Leban, G., Demšar, J., Zupan, B.: Visualization-based cancer microarray data classification analysis. Bioinformatics **23**(16), 2147–2154 (2007)
11. Blum, A., Langley, P.: Selection of relevant features and examples in machine learning. Artif. Intell. **97**(1/2), 245–271 (1997)
12. Chandrashekar, G., Sahin, F.: A survey on feature selection methods. Comput. Electr. Eng. **40**, 16–28 (2014)
13. Soufan, O., Kleftogiannis, D., Kalnis, P., Kalnis, B.: Bajic DWFS: a wrapper feature selection tool based on a parallel genetic algorithm. PLoS ONE **10**, e0117988 (2015). doi:10.1371/journal.pone.0117988