Semi-supervised Clustering: A Case Study

Andreia Silva and Cláudia Antunes

Department of Computer Science and Engineering
Instituto Superior Técnico – Technical University of Lisbon
Lisbon, Portugal
{andreia.silva,claudia.antunes}@ist.utl.pt

Abstract. The exploration of domain knowledge to improve the mining process begins to give its first results. For example, the use of domain-driven constraints allows the focusing of the discovery process on more useful patterns, from the user's point of view. Semi-supervised clustering is a technique that partitions unlabeled data by making use of domain knowledge, usually expressed as pairwise constraints among instances or just as an additional set of labeled instances. This work aims for studying the efficacy of semi-supervised clustering, on the problem of determining if some movie will achieve or not an award, just based on the movies characteristics and on ratings given by spectators. Experimental results show that, in general, semi-supervised clustering achieves better accuracy than unsupervised methods.

1 Introduction

Both clustering and classification aim for creating a model able to distinguish instances from different categories. The main difference between them is that, in the first case, the categories are not known in advance, neither their number in most cases. In unsupervised clustering, an unlabeled dataset is partitioned into groups of similar examples, typically by optimizing an objective function that characterizes good partitions. On the opposite side, in supervised classification there is a known, fixed set of categories, and labeled training data is used to induce a classification model.

Semi-supervised learning combines labeled and unlabeled data to improve performance, and is applicable to both clustering and classification. In semi-supervised clustering, some labeled data or other types of constraints are used along with the unlabeled data to obtain better clustering. Semi-supervised classification uses some unlabeled data to train the classifier. The main difference between these two approaches is that, unlike semi-supervised classification, in semi-supervised clustering the data categories can be extended and modified as needed to reflect other regularities in the data. This is very important in domains where knowledge of the relevant categories is incomplete or where labeled data does not contain examples of all categories.

In this work, we will focus on semi-supervised clustering, and apply it to a movies domain. We first collected and joined two different movie datasets

P. Perner (Ed.): MLDM 2012, LNAI 7376, pp. 252–263, 2012.

(described in section 3), in order to gather information not only about movies' characteristics, like genre, director, awards, etc., but also about the ratings given by costumers.

Our goal is to improve clustering prediction accuracy in the absence of enough labeled data, and predict which movies have or do not have at least one award. In this sense, we run some existing semi-supervised clustering algorithms[1] with several fractions of labeled data, and compare their performance with unsupervised clustering, to study the effects of seeding, and with supervised classification, to have an upper bound corresponding to the complete knowledge of the categories and all labeled data for training.

Our goal is also to analyze the efficacy of the algorithms, not only in terms of their final accuracy, but also in terms of other measures such as precision, recall and specificity.

The algorithms used are described next, and a case study on the movies domain is presented in section 3. Section 4 concludes the work, by discussing the advantages and disadvantages of semi-supervised clustering.

2 Semi-supervised Clustering

Semi-supervised clustering and its applications have been the focus of several recent projects. The most common and simplest semi-supervised clustering algorithms studied nowadays are modifications of the K-Means algorithm[7] to incorporate domain knowledge, typically through a set of labeled data points (called seeds) or pairwise constraints between the instances. This knowledge is being used to change the objective function so that it includes the satisfaction of constraints, to force some assignments to clusters[14,1] or to initialize cluster centroids[1] There are also some metric-based algorithms that employ a distance metric, trained in the beginning of the algorithm or in each iteration to satisfy the existing labels or constraints.[6,4]

There has been an effort to incorporate constraints into a more complex algorithm, Expectation Maximization (EM),[5,12] and also an effort to use other types of constraints.[3]

The used algorithms are described below.

2.1 Unsupervised Clustering with K-Means

K-Means[7] is the simplest and best known unsupervised clustering algorithm, commonly used as a baseline to compare with other clustering algorithms. It partitions a dataset into K clusters, locally minimizing the squared Euclidean distance between the data points and cluster centroids. It starts with random initial centroids and iteratively refines the clustering by assigning each instance to the nearest centroid and then recomputing each centroid as the mean of the instances of each cluster, until convergence.

[1] Available in the WEKAUT machine learning toolkit, an open source tool: `http://www.cs.utexas.edu/users/ml/risc/code`

Let $\chi = \{x_i\}_{i=1}^{N}$, with $x_i \in \mathbb{R}^d$ be a set of N data points, where the i^{th} data point is a vector represented by x_i with d dimensions. $\{\mu_h\}_{h=1}^{K}$ represents the K cluster centroids and $l_i \in \{1, ..., K\}$ is the cluster label of the point x_i. K-Means iteratively creates a K-partitioning $\{\chi_h\}_{h=1}^{K}$ of χ so that the objective function $J_{kmeans} = \sum_{x_i \in \chi} \|x_i - \mu_{l_i}\|^2$ is locally minimized.

2.2 Unsupervised Clustering with Expectation Maximization

The EM algorithm[5] is not specific for clustering, being first proposed for parameter estimation with missing data. EM is an iterative procedure that estimates the Maximum Likelihood: $\hat{\Theta}_{ML} = \arg\max_{\Theta} L(\chi|\Theta) = \arg\max_{\Theta} P(\chi, Y|\Theta)$ of the missing data Y (cluster assignments, in this case) for which the observed data χ is the most likely.

Each iteration of EM consists of two steps, repeated until convergence – Expectation (E-step) and Maximization (M-step):

$$\text{E-step:} \qquad Q(\Theta, \hat{\Theta}^{(t)}) = \mathbb{E}_{Y|\chi}[logP(\chi, Y|\Theta)|\chi, \hat{\Theta}^{(t)}]$$

$$\text{M-step:} \qquad \hat{\Theta}^{(t+1)} = \arg\max_{\Theta} Q(\Theta, \hat{\Theta}^{(t)})$$

In the E-step, given the observed data and current estimate of the model parameters, the missing data is estimated using its conditional expectation. In the M-step, under the assumption that the missing data is known, the likelihood is maximized.

It was shown in [1] that the K-Means algorithm is essentially an EM algorithm on a mixture of K Gaussians under assumptions of identity covariance of the Gaussians, uniform mixture component priors and expectation under a particular type of conditional distribution (which can be provided by semi-supervision). Under the assumption of identity covariance, the parameters to estimate are just the means of the Gaussians, i.e. the centroids of the K clusters, and the squared Euclidean distance between a point x_i and its corresponding cluster centroid μ_l is $\|x_i - \mu_{l_i}\|^2 = (x_i - \mu_{l_i})^T(x_i - \mu_{l_i})$.

2.3 Semi-supervised Clustering by Seeding

Let $S \subseteq \chi$ be a subset of data points, called the *seed set*, where for each $x_i \in S$ we have the label l of the partition χ_l to which it belongs. It is assumed that there is, at least, one seed point for each partition. The algorithms receive, therefore, a disjoint K-partition $\{S_l\}_{l=1}^{K}$ (*the seed clustering*) of the seed set S, so that all $x_i \in S_l$ belongs to χ_l, according to the supervision.

Seeded KMeans[1]. In this algorithm, the seed clustering is only used to initialize the KMeans algorithm, and it is not used in the following steps. Instead of initializing KMeans from K random means, the mean of the l^{th} cluster is initialized with the mean of the l^{th} partition of S_l of the seed set. In Seeded KMeans, the labels specified in the seed data may change in the course of the algorithm. Therefore, it is appropriate in the presence of noisy seeds.

Constrained KMeans[1]. Like Seeded KMeans, it uses the seed clustering to initialize the centroids in KMeans. However, in subsequent steps, cluster labels of seed data are kept unchanged in the cluster assignment steps, and only the labels of the non-seed data are re-estimated. It is appropriate when the initial seed labeling is noise-free, or if the user does not want the labels of the seed data to change.

2.4 Semi-supervised Clustering with Pairwise Constraints

Let us define two types of pairwise constraints, introduced by [14], that provide *a priori* knowledge about which instances should be grouped or not. Let M be a set of must-link pairs, where $(x_i, x_j) \in M$ implies x_i and x_j should be in the same cluster, and C be a set of cannot-link pairs where $(x_i, x_j) \in C$ implies x_i and x_j should be in different clusters. Let also $W = \{w_{ij}\}$ and $\overline{W} = \{\overline{w}_{ij}\}$ be penalty costs for violating the constraints in M and C respectively.

PCKMeans[4]. Pairwise Constrained KMeans uses soft pairwise constraints, i.e. it allows violation of constraints if it leads to a more cohesive clustering, and uses them to seed the initial cluster centroids and to influence the clustering.

The goal of PCKMeans is to minimize a combined objective function, defined as the sum of the total squared distances between the points and their cluster centroids, and the cost incurred by violating any pairwise constraint:

$$J_{pckmeans} = \sum_{x_i \in \chi} \|x_i - \mu_{l_i}\|^2 + \sum_{(x_i,x_j)\in M} w_{i,j} \mathbb{1}_{l_i \neq l_j} + \sum_{(x_i,x_j)\in C} \overline{w}_{i,j} \mathbb{1}_{l_i = l_j}$$

where point x_i is assigned to the partition χ_l with centroid μ_{l_i}, and where $\mathbb{1}$ is the indicator function: $\mathbb{1}_{true} = 1$ and $\mathbb{1}_{false} = 0$.

MPCKMeans[4]. Metric PCKMeans is like PCKMeans, with a metric learner component. It performs distance-metric training in each clustering iteration, making use of both unlabeled data and pairwise constraints. The Euclidean distance between two points is defined using a symmetric positive-definite weigh matrix A: $\|x_i - \mu_{l_i}\|_A = \sqrt{(x_i - \mu_{l_i})^T A (x_i - \mu_{l_i})}$. And by using a separate weight matrix A_l for each cluster l, the algorithm is capable of learning individual metrics for each cluster, which allows clusters of different shapes.

The goal of MPCKMeans is also to minimize a combined objective function like PCKMeans, but now including the weight matrices A_h and allowing different costs of constraint violations, via a function f, so that *distant* and *nearby* points can be treated differently:

$$J_{mpckmeans} = \sum_{x_i \in \chi} (\|x_i - \mu_{l_i}\|^2_{A_{l_i}} - log(det(A_{l_l}))) + \sum_{(x_i,x_j)\in M} w_{i,j} f_M(x_i, x_j) \mathbb{1}_{l_i \neq l_j} + \sum_{(x_i,x_j)\in C} \overline{w}_{i,j} f_C(x_i, x_j) \mathbb{1}_{l_i = l_j}$$

where $f_M(x_i, x_j) = \frac{1}{2}\|x_i - x_j\|^2_{A_{l_i}} + \frac{1}{2}\|x_i - x_j\|^2_{A_{l_j}}$ is the cost violation of a must-link (i.e. if $i \neq j$), and $f_C(x_i, x_j) = \|x'_{l_i} - x''_{l_i}\|^2_{A_{l_i}} - \|x_i - x_j\|^2_{A_{l_i}}$, with (x'_{l_i}, x''_{l_i}) the maximally separated pair of points in the dataset according to l_i^{th} metric, is the cost violation of a cannot-link (i.e. if $i = j$).

MPCKMeans uses EM: the E-step consists of assigning each point to the cluster that minimizes $J_{mpckmeans}$, based on the previous assignments of points to clusters; the M-step consists of two parts: re-estimating the cluster centroids given the E-step cluster assignments, like KMeans; and re-learning the metric matrices $\{A_h\}_{h=1}^{K}$ to decrease $J_{mpckmeans}$. Each updated matrix is obtained by taking the partial derivative $\frac{\partial Jmpckmeans}{\partial A_h}$ and setting it to zero, resulting in:

$$A_h = |\chi_h|\Big(\sum_{x_i \in \chi_h} (x_i - \mu_h)(x_i - \mu_h)^T + \sum_{(x_i,x_j)\in M_h} \frac{1}{2} w_{ij}(x_i - x_j)(x_i - x_j)^T \mathbb{1}_{l_i \neq l_j} + \sum_{(x_i,x_j)\in C_h} \overline{w}_{ij}((x'_h - x''_h)(x'_h - x''_h)^T - (x_i - x_j)(x_i - x_j)^T \mathbb{1}_{l_i = l_j})^{-1}$$

3 Case Study

In this case study, we decided to join two movie datasets, in order to predict whether a movie has (or will have) an award, based not only on the ratings given by customers, but also based on other characteristics of the movie, such as its genre, studio, director, etc.

This study compares the efficacy obtained by applying semi-supervised clustering techniques versus unsupervised clustering. We also run a supervised classification algorithm just to have an upper bound, corresponding to the complete knowledge of the categories and all labeled data for training.

All the experiments were conducted using implementations incorporated into the WEKAUT machine learning toolkit.[2] The applied algorithms were: KMeans [7] and EM [5] for unsupervised clustering; Seeded-KMeans and Constrained-KMeans [1], PCKMeans and MPCKMeans [4] for semi-supervised clustering; and C4.5 [9] for classification.

For all algorithms, we have generated learning curves with 10-fold cross-validation and, in each fold, 10% of the dataset is set aside as the test set, and the remaining is used as the training set. Since the results on semi-supervised clustering can diverge with the size of the given seeds, we experimented with different fractions of seeds, generated randomly from the training set. For constrained-based algorithms, we used the selected fraction of seeds to build the respective must-links and cannot-links randomly (50% of each pairwise constraints). Unit constraint costs were used for both constraints, since the dataset did not provide individual weights.

[2] WEKAUT is a Data Mining open source tool available online at `http://www.cs.utexas.edu/users/ml/risc/code`

For each algorithm, we evaluate the accuracy of the results using the *Rand Index* metric,[10] which compares the resulted clusters with the true labels available, calculating the fraction of correctly predicted movies. We also computed the *F-measure* of the results, which makes a balance between their precision (the fraction of truly awarded movies among those that the algorithm believes to have an award) and recall (the fraction of positive awarded movies correctly predicted).[8]

Clustering algorithms were run on the whole dataset, but the measures were calculated only on the test set. After running, each category ("have" or "do not have" an award) is assigned to the cluster with more instances of that category, so that we can evaluate the efficacy of the results. Results were averaged over the 10 folds.

We first describe the datasets and the preprocessing needed. Then, the experimental results are presented and discussed.

3.1 Data Description

Netflix Dataset. The Netflix dataset used in these experiments was constructed specially for the Netflix Prize.[3]

It contains over 100 million ratings from 480 thousand randomly-chosen, anonymous Netflix customers, over 17 thousand movie titles. The data were collected between October, 1998 and December, 2005 and reflect the distribution of all ratings received during this period. The ratings are on a scale from 1 to 5 (integral) stars. The year of release (from 1890 to 2005) and title of each movie are also provided.

It is known that, in most cases, movies with awards are those that many people like, and therefore those with higher ratings. Since this dataset has no other information about the movies, except their year of release (not interesting when we want to create a model to predict the future) and their title (also not interesting because in most cases, each movie has a different one), we decided to use another dataset, described below.

Gio Dataset. This dataset was extracted from a movies database donated by Gio Widerhold,[15] which collects data about more than 10 thousand movies, released between 1891 and 1999. It contains characteristics such as the genre, directors, actors, studios and awards received for each movie.

All this extra information may help predicting if a movie has an award. For example, we know some movie directors, like James Cameron and others, who have already received several awards for most of their work. We also know some actors that tend to receive awards.

Final Dataset. To have both the ratings and other characteristics of the movies in the same dataset, we joined the movies of those two datasets with the equivalent title and the same year of release. There are 2800 movies in common. Of these, 34.5% have an award and the remaining 65.5% do not have. Only these

[3] See `http://www.netflixprize.com` for details.

common movies were used in the mining process, in order to evaluate the accuracy of the clustering algorithms.

Table 1 shows the attributes chosen for the final table, as well as their meaning. The attribute year was discarded, since we want to build a model that predicts if new movies will have awards (indeed, using this attribute, C4.5 produces a tree which decides first by the year of the movie, and therefore, based on the past, which is what we want to avoid). And since the total number of ratings per movie has a wide range (from one to more than 200 thousand, with a median of just 561 ratings per movie), we used the percentage of customers that voted in a movie and gave each rating.

Table 1. Attributes in the final dataset

Attribute	**Type**	**Missing**	**Description**
Rate 1 Rate 2 Rate 3 Rate 4 Rate 5	Real: [0 ; 1]	0%	Percentage of customers that voted in the movie and gave the star 1, 2, 3, 4 or 5, respectively
Med Rate	Integer: 1 to 5	0%	The medium rate of the movie
Director	Nominal: 1190 values	0%	Director's name
Actor 1	Nominal: 1808 values	10%	First main actor's name
Actor 2	Nominal: 1772 values	13%	Second main actor's name
Studio Name	Nominal: 314 values	35%	Studio's name
Studio Country	Nominal: 10 values	57%	Country where the movie was made
Genre	Nominal: 12 values	0%	Genre of the movie
Awards	Boolean	0%	True if the movie has an award; False otherwise

To deal with the missing values of an attribute, the implemented algorithms adopt different strategies. K-Means replaces them by the mode or mean of the attribute, while EM ignores them, and the semi-supervised algorithms available are not able to deal with them. In the context of actors and studios, which has many possible values, with completely different meanings, it makes no sense to replace missing values of an attribute by its mode. Therefore, they were replaced by a non-existing value, "zero", representing the lack of value.

3.2 Experimental Results

We first run the classification algorithm to observe its performance, and which attributes were used to build the model. Indeed, the final tree has attributes from both datasets: in a first level of decision nodes, it uses the attribute "studio name"; In the second and third levels, it uses the ratings or the genres of the movies. As an example, for a movie produced in the studio *Gaumont*:

if $StudioName = Gaumont$
  if $Med <= 3 : false$
  or if $Med > 3$
    if $Rate3 <= 0.25 : false$
    or if $Rate3 > 0.25 : true$

if the average ratings ("*Med*") is less than or equal to 3, it means the movie has no award. Otherwise, even with an average higher than 3, it only has an award if more than 25% of customers rated the movie with 3.

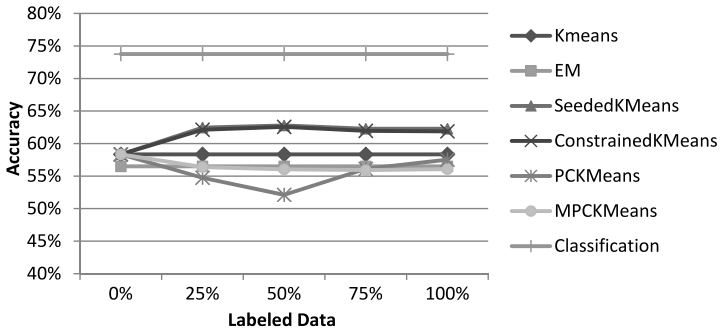

Fig. 1. Accuracy results

Figure 1 shows the accuracy (or Rand Index) of the algorithms, for different fractions of labeled instances incorporated. As we can see, semi-supervised clustering by seeding (Seeded and Constrained KMeans) always performed better than unsupervised clustering (KMeans and EM), even with a small number of seeds (% of labeled data). On the other hand, semi-supervised clustering by constraints (PC and MPC KMeans) seems have worst accuracy results. PCKMeans only started getting improvements with more than 50% of constraints, but it does not achieve better accuracy results than unsupervised clustering algorithms. Note that the labeled data is selected only from the train set, and the results are computed from the test set.

Although seeding algorithms achieve a better accuracy, it does not show where the correct decisions come from (positive awarded – also called true positives, or negative awarded movies – called true negatives). At figure 2 we analyze just that, by showing the recall and the specificity of the algorithms. The recall is the fraction of positive awarded movies correctly predicted (also known as True Positive Rate or sensitivity) and specificity is the fraction of non awarded movies correctly predicted as such (True Negative Rate).

In the chart we can see that semi-supervised algorithms were much better to correctly predict which movies do not have an award, than predicting which movies have (very high specificity versus very low recall). Only MPCKMeans had the recall and specificity at the same level as the unsupervised EM and KMeans. Seeded and Constrained KMeans are very similar to each other, and have a very low recall. However, their specificity is much higher than all other algorithms, even classification.

Finally, figure 3 presents the precision and the f-measure of the models built. Precision is the fraction of truly awarded movies among those that the algorithm

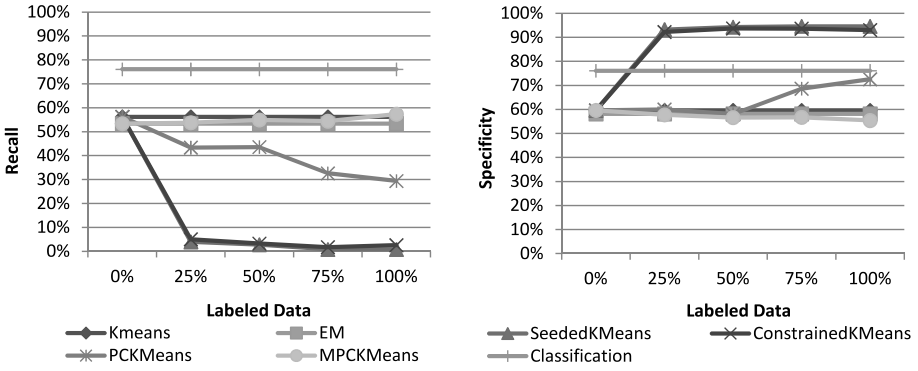

Fig. 2. Recall (TP Rate) and Specificity (TN Rate) results

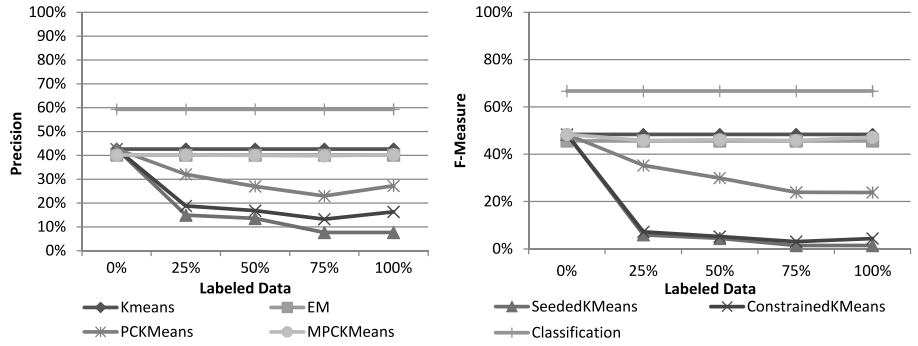

Fig. 3. Precision and F-Measure results

believes to have an award, and f-measure is a balance between precision and recall.

By analyzing the precision, we confirm that semi-supervised methods have difficulties in creating the cluster for positive awarded movies, since a lower precision indicates the presence of a lot of non-awarded movies in the cluster that represents the awarded ones. Although similar, Seeded KMeans proved to be worse than Constrained KMeans. Only MPCKMeans behaves like unsupervised clustering, maintaining its behavior. The F-measure culminates these results, showing that the best balance between the precision and recall is MPCKMeans, and that seeded algorithms are not good in none of them.

3.3 Discussion

As results showed, semi-supervised clustering does not always bring improvements over unsupervised clustering accuracy. However, the seeding algorithms can achieve a better overall accuracy.

The bad results in recall and very good ones in specificity can be explained by the fact that the dataset is unbalanced, with 34.5% of awarded movies, versus 65.5% of non awarded movies. This problem, also known as skewed class distribution, may cause the learning algorithms have difficulty in learning the concepts related to the minority class. In this case, seeded clustering has difficulties in clustering awarded movies (low recall), but can cluster non awarded movies very well (high specificity). This problem can be addressed by forms of sampling and other techniques that transform the dataset into a more balanced one.[11] However, any real dataset for this kind of analysis, used to predict if a movie has (or will have) an award, will always have much more movies without an award than with one. Therefore, the results obtained for an extracted balanced dataset may not be significant for a real dataset.

Another problem that can also explain the low results in recall is related to the choice of seeds and constraints, and is the fact that we do not really know how they will influence the results. Researchers have already studied this problem, and explain it with two properties of seed or constraint sets: *informativeness* and *coherence*.[13] Informativeness refers to "the amount of information in the set that the algorithm cannot determine on its own". Coherence is "the amount of agreement between the elements in the set, given a distance metric". They recommend the selection of sets with higher informativeness and coherence values, to avoid situations where the constraints negatively impact the performance. There are already some good works that already exploit this combination.[2]

Other interesting fact we can observe is the constant behavior of Seeded and Constrained KMeans. We see that more seeds do not change their results, meaning that new seeds do not bring new relevant information for cluster initialization.They had also very similar results. This may indicate that the initial seeds are not noisy, and that the clusters have a wide overlapping (and in fact the accuracy is above 50%, which indicates that the dataset is not easily separable).

In global terms, we can conclude that semi-supervised clustering based on seeds achieves better accuracy than unsupervised clustering, even with few fractions of seeds. They may have problems predicting which movies have an award, but they can predict, better than the others, which movies do not have an award, and therefore they can help, for example, limiting the number of candidates to awards, or eliminating a larger set of non awarded movies before another algorithm run.

4 Conclusions

Unlike unsupervised clustering, semi-supervised clustering uses some labeled data to aid the search and bias the partitioning of unlabeled data, and therefore, they can generally learn better models and achieve better accuracy results. However, the good accuracy may stem from just one or some correct decisions and not from all.

With our case study, we can conclude that semi-supervised clustering by seeding proved to be better in accuracy than approaches that use pairwise constraints and unsupervised ones, with good results even for a few fraction of seeds.

However, there are some questions we have to consider, for example:

Is the dataset unbalanced? Many real datasets are. If so, it is expected the algorithm to learn better how to cluster the instances of the most common class, and have problems with the minority class, as shown here;

What is the goal of the learning task? We cannot just look to the accuracy of an algorithm, we should analyze other measures, like recall, specificity, precision and/or f-measure, accordingly to our goal;

To what extent the seeds help? As explained before, seeds not always help. We should try to select seed sets or constraints with higher informativeness and coherence values. Future work could go through a study testing several sets of seeds and constraints.

Acknowledgment. This work is partially supported by FCT – Fundação para a Ciência e a Tecnologia, under research project D2PM (PTDC/EIA-EIA/110074/2009) and PhD grant SFRH/BD/64108/2009.

References

1. Basu, S., Banerjee, A., Mooney, R.: Semi–supervised clustering by seeding. In: Proceedings of 19th International Conference on Machine Learning, ICML (2002)
2. Basu, S., Banerjee, A., Mooney, R.J.: Active semi-supervision for pairwise constrained clustering. In: Proceedings of the 2004 SIAM International Conference on Data Mining (SDM), pp. 333–344 (2004)
3. Basu, S., Davidson, I., Wagstaff, K. (eds.): Constrained Clustering: Advances in Algorithms, Theory, and Applications. Chapman and Hall/CRC (2008)
4. Bilenko, M., Basu, S., Mooney, R.J.: Integrating constraints and metric learning in semi-supervised clustering. In: Proceedings of 21st International Conference on Machine Learning, ICML (2004)
5. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B (Methodological) 39(1), 1–38 (1977)
6. Klein, D., Kamvar, S.D., Manning, C.D.: From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In: Proceedings of 19th International Conference on Machine Learning (ICML), pp. 307–314 (2002)
7. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297 (1967)
8. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to information retrieval. Cambridge University Press (2008)
9. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco (1993)
10. Rand, W.M.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association 66(336), 846–850 (1971)
11. Scholz, M.: Sampling-based sequential subgroup mining. In: Knowledge Discovery and Data Mining, pp. 265–274 (2005)

12. Shental, N., Bar-Hillel, A., Hertz, T., Weinshall, D.: Computing gaussian mixture models with em using equivalence constraints. In: Advances in Neural Information Processing Systems (NIPS) 16 (2003)
13. Wagstaff, K., Basu, S., Davidson, I.: When is constrained clustering beneficial, and why. In: AAAI (2006)
14. Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S.: Constrained k-means clustering with background knowledge. In: Proceedings of 18th International Conference on Machine Learning (ICML), pp. 577–584 (2001)
15. Wiederhold, G.: Movies database documentation (1989)