Predictive K-means with local models

Vincent Lemaire[1], Oumaima Alaoui Ismaili[1],
Antoine Cornuéjols[2], Dominique Gay[3]

[1] Orange Labs, Lannion, France
[2] AgroParisTech, Université Paris-Saclay, Paris, France
[3] LIM-EA2525, Université de La Réunion

Abstract. Supervised classification can be effective for prediction but sometimes weak on interpretability or explainability (XAI). Clustering, on the other hand, tends to isolate categories or profiles that can be meaningful but there is no guarantee that they are useful for labels prediction. Predictive clustering seeks to obtain the best of the two worlds. Starting from labeled data, it looks for clusters that are as pure as possible with regards to the class labels. One technique consists in tweaking a clustering algorithm so that data points sharing the same label tend to aggregate together. With distance-based algorithms, such as k-means, a solution is to modify the distance used by the algorithm so that it incorporates information about the labels of the data points. In this paper, we propose another method which relies on a change of representation guided by class densities and then carries out clustering in this new representation space. We present two new algorithms using this technique and show on a variety of data sets that they are competitive for prediction performance with pure supervised classifiers while offering interpretability of the clusters discovered.

1 Introduction

While the power of predictive classifiers can sometimes be awesome on given learning tasks, their actual usability might be severely limited by the lack of interpretability of the hypothesis learned. The opacity of many powerful supervised learning algorithms has indeed become a major issue in recent years. This is why, in addition to good predictive performance as standard goal, many learning methods have been devised to provide readable decision rules [3], degrees of beliefs, or other easy to interpret visualizations. This paper presents a predictive technique which promotes interpretability, explainability as well, in its core design.

The idea is to combine the predictive power brought by supervised learning with the interpretability that can come from the descriptions of categories, profiles, and discovered using unsupervised clustering. The resulting family of techniques is variously called *supervised clustering* or *predictive clustering.* In the literature, there are two categories of predictive clustering. The first family of algorithms aims at optimizing the trade-off between description and prediction, i.e., aiming at detecting sub-groups in each target class. By contrast, the algorithms in the second category favor the prediction performance over the discovery of all underlying clusters, still using clusters as the basis of the decision function. The hope is that the predictive performance of predictive clustering methods can approximate the performances of supervised classifiers while their descriptive capability remains close to the one of pure clustering algorithms.

Several predictive clustering algorithms have been presented over the years, for instance [1,4,10,11,23]. However, the majority of these algorithms require (*i*) a considerable execution time, and (*ii*) that numerous user parameters be set. In addition, some algorithms are very sensitive to the presence of noisy data and consequently their outputs are not easily interpretable (see [5] for a survey). This paper presents a new predictive clustering algorithm. The underlying idea is to use any existing distance-based clustering algorithms, e.g. k-means, but on a redescription space where the target class is integrated. The resulting algorithm has several desirable properties: there are few parameters to set, its computational complexity is almost linear in m, the number of instances, it is robust to noise, its predictive performance is comparable to the one obtained with classical supervised classification techniques and it tends to produce groups of data that are easy to interpret for the experts.

The remainder of this paper is organized as follows: Section II introduces the basis of the new algorithm, the computation of the clusters, the initialization step and the classification that is realized within each cluster. The main computation steps of the resulting predictive clustering algorithms are described in Algorithm 1. We then report experiments that deal with the predictive performance in Sections 3. We focus on the supervised classification performance to assess if predictive clustering could reach the performances of algorithms dedicated to supervised classification. Our algorithm is compared using a variety of data sets with powerful supervised classification algorithms in order to assess its value as a predictive technique. And an analysis of the results is carried out. Conclusion and perspectives are discussed in Section 4.

2 Turning the K-means algorithm predictive

The k-means algorithm is one of the simplest yet most commonly used clustering algorithms. It seeks to partition m instances $(X_1, \dots X_m)$ into K groups $(B_1, \dots, B_K)$ so that instances which are close are assigned to the same cluster while clusters are as dissimilar as possible. The objective function can be defined as:

$$\mathcal{G} \; = \underset{B_i}{\text{Argmin}} \sum_{i=1}^{K} \sum_{X_j \in B_i} \|X_j - \mu_i\|^2 \tag{1}$$

where μ_i are the centers of clusters B_i and we consider the Euclidean distance.

Predictive clustering adds the constrain of maximizing clusters purity (i.e. instances in a cluster should share the same label). In addition, the goal is to provide results that are easy to interpret by the end users. The objective function of Equation (1) needs to be modified accordingly.

One approach is to modify the distance used in conventional clustering algorithm in order to incorporate information about the class of the instances.

This modified distance should make points differently labelled appear as more distant than in the original input space. Rather than modifying the distance, one can instead alter the input space. This is the approach taken in this paper, where the input space is partitioned according to class probabilities prior to the clustering step, thus favoring clusters of high purity. Besides the introduction of a technique for computing a new feature space, we propose as well an adapted initialization method for the modified k-means algorithm. We also show the advantage of using a specific classification method within each discovered cluster in order to improve the classification performance. The main steps of the resulting algorithm are described in Algorithm 1. In the remaining of this section II we show how each step of the usual K-means is modified to yield a predictive clustering algorithm.

Algorithm 1 Predictive K-means algorithm

```
Input:
- $D$: a data set which contains $m$ instances. Each one $(X_i)$ $i \in \{1, \dots, m\}$ is described by $d$ descriptive features and a label $C_i \in \{1, \dots, J\}$.
- $K$: number of clusters .
Start:
1) Supervised preprocessing of data to represent each $X_i$ as $\widehat{X}_i$ in a new feature space $\Phi(\mathcal{X})$.
2) Supervised initialization of centers.
repeat
  3) Assignment: generate a new partition by assigning each instance $\widehat{X}_i$ to the nearest cluster.
  4) Representation: calculate the centers of the new partition.
until the convergence of the algorithm
5) Assignment classes to the obtained clusters:
        - method 1: majority vote.
        - method 2: local models.
6) Prediction the class of the new instances in the deployment phase:
        $\rightarrow$ the closest cluster class (if method 1 is used).
        $\rightarrow$ local models (if method 2 is used).
End
```

A modified input space for predictive clustering - The principle of the proposed approach is to partition the input space according to the class probabilities $P(C_j|X)$. More precisely, let the input space $\mathcal{X}$ be of dimension d, with numerical descriptors as well as categorical ones. An example $X_i \in \mathcal{X}$ ($X_i = [X_i^{(1)}, \dots, X_i^{(d)}]^\top$) will be described in the new feature space $\Phi(\mathcal{X})$ by $d \times J$ components, with J being the number of classes. Each component $X_i^{(n)}$ of $X_i \in \mathcal{X}$ will give J components $X_i^{(n,j)}$, for $j \in \{1, \dots, J\}$, of the new description $\widehat{X}_i$ in $\Phi(\mathcal{X})$, where $\widehat{X}_i^{(n,j)} = \log P(X^{(n)} = X_i^{(n)}|C_j)$, i.e., the log-likelihood values. Therefore, an example X is redescribed according to the (log)-probabilities of observing the values of original input variables given each of the J possible classes (see Figure 1). Below, we describe a method for computing these values.

But first, we analyze one property of this redescription in $\Phi(\mathcal{X})$ and the distance this can provide.

$\mathcal{X}$	$X^{(1)}$	$\ldots$	$X^{(d)}$	Y
X_1		$\ldots$		
$\ldots$		$\ldots$		
X_m		$\ldots$		

$$\overset{\Phi}{\Longrightarrow}$$

$\Phi(\mathcal{X})$	$X^{(1,1)}$	$\ldots$	$X^{(1,J)}$	$\ldots$	$X^{(d,1)}$	$\ldots$	$X^{(d,J)}$	Y
X_1				$\ldots$				
$\ldots$				$\ldots$				
X_m				$\ldots$				

Fig. 1. Φ redescription scheme from d variables to $d \times J$ variables, with log-likelihood values: $\log P(X^{(n)}|C_j)$

Property of the modified distance - Let us denote $dist_B^p$ the new distance defined over $\Phi(\mathcal{X})$. For the two recoded instances $\hat{X}_1$ and $\hat{X}_2 \in \mathbb{R}^{d\times J}$, the formula of $dist_B^p$ is (in the following we omit $\widehat{X} = \widehat{X}_i$ in the probability terms for notation simplification):

$$dist_B^p(\hat{X}_1, \hat{X}_2) = \sum_{j=1}^{J} \| \log(P(\hat{X}_1|C_j)) - \log(P(\hat{X}_2|C_j)) \|_p \tag{2}$$

where $\| \ . \ \|_p$ is a Minkowski distance. Let us denote now, $\Delta^p(\hat{X}_1, \hat{X}_2)$ the distance between the (log)-posterior probabilities of two instances $\hat{X}_1$ and $\hat{X}_2$. The formula of this distance as follow:

$$\Delta^p(\hat{X}_1, \hat{X}_2) = \sum_{j=1}^{J} \| \log(P(C_j|\hat{X}_1)) - \log(P(C_j|\hat{X}_2)) \|_p \tag{3}$$

where $\forall i \in \{1, \ldots, m\}$, $P(C_j|\hat{X}_i) = \frac{P(C_j)\prod_{n=1}^{d} P(X_i^{(n)}|C_j)}{P(\hat{X}_i)}$ (using the hypothesis of features independence conditionally to the target class). From the distance given in equation 3, we find the following inequality:

$$\Delta^p(\hat{X}_1, \hat{X}_2) \leq \{dist_B^p(\hat{X}_1, \hat{X}_2) + J \| \log(P(\hat{X}_2)) - \log(P(\hat{X}_1)) \|\} \tag{4}$$

Proof.

$$\begin{aligned}
\Delta^p &= \sum_{j=1}^{J} \| \log(P(C_j|\hat{X}_1)) - \log(P(C_j|\hat{X}_2)) \|_p \\
&= \sum_{j=1}^{J} \| \log(\frac{P(\hat{X}_1|C_j)P(C_j)}{P(\hat{X}_1)}) - \log(\frac{P(\hat{X}_2|C_j)P(C_j)}{P(\hat{X}_2)}) \|_p \\
&= \sum_{j=1}^{J} \| \log(P(\hat{X}_1|C_j)) - \log(P(\hat{X}_1)) - \log(P(\hat{X}_2|C_j)) + \log(P(\hat{X}_2)) \|_p \\
&\leq \sum_{j=1}^{J} [A + B]
\end{aligned}$$

with $\Delta^p = \Delta^p(\hat{X}_1, \hat{X}_2)$ and $A = \| \log(P(\hat{X}_1|C_j)) - \log(P(\hat{X}_2|C_j)) \|_p$
and $B = \| \log(P(\hat{X}_2)) - \log(P(\hat{X}_1)) \|_p$
then $\Delta^p \leq dist_B^p(\hat{X}_1, \hat{X}_2) + J \| \log(P(\hat{X}_2)) - \log(P(\hat{X}_1)) \|_p$.

This above inequality expresses that two instances that are close in terms of distance $dist_B^p$ will also be close in terms of their probabilities of belonging to the same class. Note that the distance presented above can be integrated into any distance-based clustering algorithms.

Building the log-likelihood redescription $\Phi(\mathcal{X})$ - Many methods can estimate the new descriptors $X_i^{(n,j)} = \log P(X^{(n)} = X_i^{(n)}|C_j)$ from a set of examples. In our work, we use a supervised discretization method for numerical attributes and a supervised grouping values for categorical attributes to obtain respectively intervals and group values in which $P(X^{(n)} = X_i^{(n)}|C_j)$ could be measured. The used supervised discretization method is described in [8] and the grouping method in [7]. The two methods have been compared with extensive experiments to corresponding state of the art algorithms. These methods computes univariate partitions of the input space using supervised information. It determines the partition of the input space to optimize the prediction of the labels of the examples given the intervals in which they fall using the computed partition. The method finds the best partition (number of intervals and thresholds) using a Bayes estimate. An additional bonus of the method is that outliers are automatically eliminated and missing values can be imputed.

Initialisation of centers - Because clustering is a NP-hard problem, heuristics are needed to solve it, and the search procedure is often iterative, starting from an initialized set of prototypes. One foremost example of many such distance-based methods is the k-means algorithm. It is known that the initialization step can have a significant impact both on the number of iterations and, more importantly, on the results which correspond to local minima of the optimization criterion (such as Equation 1 in [20]). However, by contrast to the classical clustering methods, predictive clustering can use supervised information for the choice of the initial prototypes. In this study, we chose to use the `K++R` method. Described in [17], it follows an "exploit and explore" strategy where the class labels are first exploited before the input distribution is used for exploration in order to get the apparent best initial centers. The main idea of this method is to dedicate one center per class (comparable to a *"Rocchio"* [19] solution). Each center is defined as the average vector of instances which have the same class label. If the predefined number of clusters (K) exceeds the number of classes (J), the initialization continues using the K-means++ algorithm [2] for the $K - J$ remaining centers in such a way to add diversity. This method can only be used when $K \geq J$, but this is fine since in the context of supervised clustering[4] we do not look for clusters where $K < J$. The complexity of this scheme is $\mathcal{O}(m + (K - J)m) < \mathcal{O}(mK)$, where m is the number of examples. When $K = J$, this method is deterministic.

Instance assignment and centers update - Considering the Euclidean distance and the original K-means procedure for updating centers, at each iteration, each instance is assigned to the nearest cluster (j) using the ℓ_2 metric ($p = 2$) in

[4] In the context of supervised clustering, it does not make sense to cluster instances in K clusters where $K < J$

the redescription space $\Phi(\mathcal{X})$. The K centers are then updated according to the K-Means procedure. This choice of distance (Euclidean) in the adopted k-means strategy could have an influence on the (predictive) relevance of the clusters but has not been studied in this paper.

Label prediction in predictive clustering - Unlike classical clustering which aims only at providing a description of the available data, predictive clustering can also be used in order to make prediction about new incoming examples that are unlabeled.

The commonest method used for prediction in predictive K-means is the majority vote. A new example is first assigned to the cluster of its nearest prototype, and the predicted label is the one shared by the majority of the examples of this cluster. This method is not optimal. Let us call P_M the frequency of the majority class in a given cluster. The true probability μ of this class obeys the Hoeffding inequality: $P\big(|P_M - \mu| \geq \varepsilon\big) \leq 2\exp(-2\, m_k\, \varepsilon^2)$ with m_k the number of instances assigned to the cluster k. If there are only 2 classes, the error rate is $1 - \mu$ if P_M and μ both are > 0.5. But the error rate can even exceed 0.5 if $P_M > 0.5$ while actually $\mu < 0.5$. The analysis is more complex in case of more than two classes. It is not the object of this paper to investigate this further. But it is apparent that the majority rule can often be improved upon, as is the case in classical supervised learning.

Another evidence of the limits of the majority rule is provided by the examination of the ROC curve [12]. Using the majority vote to assign classes for the discovered clusters generates a ROC curve where instances are ranked depending on the clusters. Consequently, the ROC curve presents a sequence of steps. The area under the ROC curve is therefore suboptimal compared to a ROC curve that is obtained from a more refined ranking of the examples, e.g., when class probabilities are dependent upon each example, rather than groups of examples.

One way to overcome these limits is to use local prediction models in each cluster, hoping to get better prediction rules than the majority one. However, it is necessary that these local models: 1) can be trained with few instances, 2) do not overfit, 3) ideally, would not imply any user parameters to avoid the need for local cross-validation, 4) have a linear algorithmic complexity $O(m)$ in a learning phase, where m is the number of examples, 5) are not used in the case where the information is insufficient and the majority rule is the best model we can hope for, 6) keep (or even improve) the initial interpretation qualities of the global model. Regarding item (1), a large study has been conducted in [22] in order to test the prediction performances in function of the number of training instance of the most commonly classifiers. One prominent finding was that the Naive Bayes (NB) classifier often reaches good prediction performances using only few examples (Bouchard & Triggs's study [6] confirms this result). This fact remains valid even when features receive weights (*e.g.,* Averaging Naive Bayes (ANB) and Selective Naive Bayes (SNB) [16]). We defer discussion of the other items to the Section 3 on experimental results.

In our experiments, we used the following procedure to label each incoming data point X: i) X is redescribed in the space $\Phi(\mathcal{X})$ using the method described in Section II, ii) X is assigned to the cluster k corresponding to the nearest center iii) the local model, l, in the corresponding cluster is used to predict the class of X (and the probability memberships) if a local model exits: $P(j|X) = \text{argmax}_{1\leq j\leq J}(P_{SNB_l}(C_j|X))$ otherwise the majority vote is used (Note: $P_{SNB_l}(C_j|X))$ is described in the next section).

3 Comparison with supervised Algorithm

3.1 The chosen set of classifiers

To test the ability of our algorithm to exhibit high predictive performance while at the same time being able to uncover interesting clusters in the different data sets, we have compared it with three powerful classifiers (in the spirit of, or close to our algorithm) from the state of the art: Logistic Model Tree (`LMT`) [15], Naives Bayes Tree (`NBT`) [14] and Selective Naive Bayes (SNB) [9]. This section briefly described these classifiers.

• **Logistic Model Tree (`LMT`)** [15] combines logistic regression and decision trees. It seeks to improve the performance of decision trees. Instead of associating each leaf of the tree to a single label and a single probability vector (piecewise constant model), a logistic regression model is trained on the instances assigned to each leaf to estimate an appropriate vector of probabilities for each test instance (piecewise linear regression model). The logit-Boost algorithm is used to fit a logistic regression model at each node, and then it is partitioned using information gain as a function of impurity.

• **Naives Bayes Tree (`NBT`)** [14] is a hybrid algorithm, which deploys a naive Bayes classifier on each leaf of the built decision tree. `NBT` is a classifier which has often exhibited good performance compared to the standard decision trees and naive Bayes classifier.

• **Selective Naive Bayes (`SNB`)** is a variant of `NB`. One way to average a large number of selective naive Bayes classifiers obtained with different subsets of features is to use one model only, but with features weighting [9]. The Bayes formula under the hypothesis of features independence conditionally to classes becomes: $P(j|X) = \frac{P(j)\prod_f P(X^f|j)^{W_f}}{\sum_{j=1}^{K}\left[P(j)\prod_f P(X^f|j)^{W_f}\right]}$, where W_f represents the weight of the feature f, X^f is component f of X, j is the class labels. The predicted class j is the one that maximizes the conditional probability $P(j|X)$. The probabilities $P(X_i|j)$ can be estimated by interval using a discretization for continuous features. For categorical features, this estimation can be done if the feature has few different modalities. Otherwise, grouping into modalities is used. The resulting algorithm proves to be quite efficient on many real data sets [13].

• **Predictive K-Means ($\texttt{PKM}_{\texttt{MV}}$, $\texttt{PKM}_{\texttt{SNB}}$):** (i) $\texttt{PKM}_{\texttt{VM}}$ corresponds to the Predictive K-Means described in Algorithm 1 where prediction is done according to the Majority Vote; (ii) $\texttt{PKM}_{\texttt{SNB}}$ corresponds to the Predictive K-Means described in Algorithm 1 where prediction is done according to a local classification model.

• **Unsupervised K-Means ($\texttt{KM}_{\texttt{MV}}$)** is the usual unsupervised K-Means with prediction done using the Majority Vote in each cluster. This classifier is given for comparison as a baseline method. The pre-processing is not supervised and the initialization used is k-means++ [2] (in this case since the initialization is not deterministic we run k-means 25 times and we keep the best initialization according to the Mean Squared Error). Among the existing unsupervised pre-processing approaches [21], depending on the nature of the features, continuous or categorical, we used:

- for Numerical attribute: Rank Normalization (RN). The purpose of rank normalization is to rank continuous feature values and then scale the feature into $[0, 1]$. The different steps of this approach are: (i) rank feature values u from lowest to highest values and then divide the resulting vector into H intervals, where H is the number of intervals, (ii) assign for each interval a label $r \in \{1, ..., H\}$ in increasing order, (iii) if X_{iu} belongs to the interval r, then $X'_{iu} = \frac{r}{H}$. In our experiments, we use $H = 100$.
- for Categorical attribute: we chose to use a Basic Grouping Approach (BGB). It aims at transforming feature values into a vector of Boolean values. The different steps of this approach are: (i) group feature values into g groups with as equal frequencies as possible, where g is a parameter given by the user, (ii) assign for each group a label $r \in \{1, ..., g\}$, (iii) use a full disjunctive coding. In our experiments, we use $g = 10$.

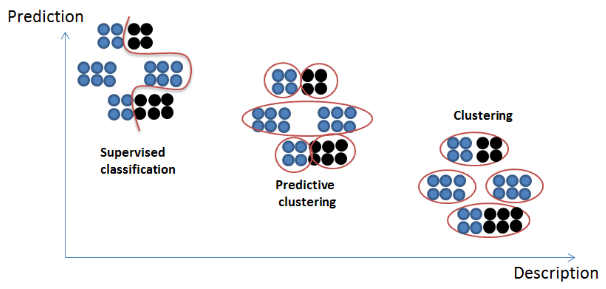

Fig. 2. Differences between the three types of "classification"

In the Figure 2 we suggest a two axis figure to situate the algorithms described above: a vertical axis for their ability to describe (explain) the data (from low to high) and horizontal axis for their ability to predict the labels (from low to high). In this case the selected classifiers exemplify various trade-offs between prediction performance and explanatory power: (i) $\texttt{KM}_{\texttt{MV}}$ more dedicated to description would appear in the bottom right corner; (ii) `LMT`, `NBT` and `SNB` dedicated to prediction would go on the top left corner; and (iii) $\texttt{PKM}_{\texttt{VM}}$, $\texttt{PKM}_{\texttt{SNB}}$ would lie in between. Ideally, our algorithm, $\texttt{PKM}_{\texttt{SNB}}$ should place itself on the top right quadrant of this kind of figure with both good prediction and description performance.

Note that in the reported experiments, $K = J$ (*i.e,* number of clusters = number of classes). This choice which biases the algorithm to find one cluster per class, is detrimental for predictive clustering, thus setting a lower bound on the performance that can be expected of such an approach.

3.2 Experimental protocol

The comparison of the algorithms have been performed on 8 different datasets of the UCI repository [18]. These datasets were chosen for their diversity in terms of classes, features (categorical and numerical) and instances number (see Table 1).

Datasets	Instances	$\#V_n$	$\#V_c$	# Classes	**Datasets**	Instances	$\#V_n$	$\#V_c$	# Classes
Glass	214	10	0	6	Waveform	5000	40	0	3
Pima	768	8	0	2	Mushroom	8416	0	22	2
Vehicle	846	18	0	4	Pendigits	10992	16	0	10
Segmentation	2310	19	0	7	Adult	48842	7	8	2

Table 1. The used datasets, V_n: numerical features, V_c: categorical features.

Evaluation of the performance: In order to compare the performance of the algorithms presented above, the same folds in the train/test have been used. The results presented in Section 3.3 are those obtained in the test phase using a 10×10 folds cross validation (stratified). The predictive performance of the algorithms are evaluated using the AUC (area under the ROC's curve). It is computed as follows: $\texttt{AUC} = \sum_i^C P(C_i)\texttt{AUC}(C_i)$, where $\texttt{AUC}(i)$ denotes the AUC's value in the class i against all the others classes and $P(Ci)$ denotes the prior on the class i (the elements frequency in the class i). $\texttt{AUC}(i)$ is calculated using the probability vector $P(C_i|X)\ \forall i$.

average results (in the test phase) using **ACC**						
Data	$\texttt{KM}_{\texttt{MV}}$	$\texttt{PKM}_{\texttt{MV}}$	$\texttt{PKM}_{\texttt{SNB}}$	LMT	NBT	SNB
Glass	70.34 ± 8.00	89.32 ± 6.09	95.38 ± 4.66	97.48 ± 2.68	94.63 ± 4.39	**97.75** ±3.33
Pima	65.11 ± 4.17	66.90 ± 4.87	73.72 ± 4.37	**76.85** ±4.70	75.38 ± 4.71	75.41 ± 4.75
Vehicle	37.60 ± 4.10	47.35 ± 5.62	72.21 ± 4.13	**82.52** ± 3.64	70.46 ± 5.17	64.26 ± 4.39
Segment	67.50 ± 2.35	80.94 ± 1.93	96.18 ± 1.26	**96.30** ± 1.15	95.17 ± 1.29	94.44 ± 1.48
Waveform	50.05 ± 1.05	49.72 ± 3.39	84.04 ± 1.63	**86.94** ± 1.69	79.87 ± 2.32	83.14 ± 1.49
Mushroom	89.26 ± 0.97	98.57 ± 3.60	**99.94** ± 0.09	98.06 ± 4.13	95.69 ± 6.73	99.38 ± 0.27
PenDigits	73.65 ± 2.09	76.82 ± 1.33	97.35 ± 1.36	**98.50** ± 0.35	95.29 ± 0.76	89.92 ± 1.33
Adult	76.07 ± 0.14	77.96 ± 0.41	**86.81** ± 0.39	83.22 ± 1.80	79.41 ± 7.34	86.63 ± 0.40
Average	66.19	73.44	88.20	**89.98**	85.73	86.36
average results (in the test phase) using 100 x **AUC**						
Data	$\texttt{KM}_{\texttt{MV}}$	$\texttt{PKM}_{\texttt{MV}}$	$\texttt{PKM}_{\texttt{SNB}}$	LMT	NBT	SNB
Glass	85.72 ± 5.69	96.93 ± 2.84	98.27 ± 2.50	97.94 ± 0.19	98.67 ± 2.05	**99.77** ± 0.54
Pima	65.36 ± 5.21	65.81 ± 6.37	78.44 ± 5.35	**83.05** ± 4.61	80.33 ± 5.21	80.59 ± 4.78
Vehicle	65.80 ± 3.36	74.77 ± 3.14	91.15 ± 1.75	**95.77** ± 1.44	88.07 ± 3.04	87.19 ± 1.97
Segment	91.96 ± 0.75	95.24 ± 0.75	99.51 ± 0.32	**99.65** ± 0.23	98.86 ± 0.51	99.52 ± 0.19
Waveform	75.58 ± 0.58	69.21 ± 3.17	96.16 ± 0.58	**97.10** ± 0.53	93.47 ± 1.41	95.81 ± 0.57
Mushroom	88.63 ± 1.03	98.47 ± 0.38	**99.99** ± 0.00	99.89 ± 0.69	99.08 ± 2.29	99.97 ± 0.02
Pendigits	95.34 ± 0.45	95.84 ± 0.29	99.66 ± 0.11	**99.81** ± 0.10	99.22 ± 1.78	99.19 ± 1.14
Adult	73.33 ± 0.65	59.42 ± 3.70	**92.37** ± 0.34	77.32 ± 10.93	84.25 ± 5.66	92.32 ± 0.34
Average	80.21	81.96	**94.44**	93.81	92.74	94.29

Table 2. Mean performance and standard deviation for the TEST set using a 10x10 folds cross-validation process

3.3 Results

Performance evaluation: Table 2 presents the predictive performance of LMT, NBT, SNB, our algorithm $\texttt{PKM}_{\texttt{MV}}$, $\texttt{PKM}_{\texttt{SNB}}$ and the baseline $\texttt{KM}_{\texttt{MV}}$ using the ACC (accuracy) and the AUC criteria (presented as a %). These results show the very good prediction performance of the $\texttt{PKM}_{\texttt{SNB}}$ algorithm. Its performance is indeed comparable to those of LMT and SNB which are the strongest ones. In addition, the use of local classifiers (algorithm $\texttt{PKM}_{\texttt{SNB}}$) provides a clear advantage over the use of the majority vote in each cluster as done in $\texttt{PKM}_{\texttt{MV}}$. Surprisingly, $\texttt{PKM}_{\texttt{SNB}}$ exhibits slightly better results than SNB while both use naive Bayes classifiers locally and $\texttt{PKM}_{\texttt{SNB}}$ is hampered by the fact that $K = J$, the number of classes. Better performance are expected when $K \geq J$. Finally, $\texttt{PKM}_{\texttt{SNB}}$ appears to be slightly superior to SNB, particularly for the datasets which contain highly correlated features, for instance the PenDigits database.

Discussion about local models, complexity and others factors: In Section II in the paragraph about label prediction in predictive clustering, we proposed a list of desirable properties for the local prediction models used in each cluster. We come back to these items denoted from (i) to (vi) in discussing Tables 2 and 3:

i) The performance in prediction are good even for the dataset Glass which contains only 214 instances (90% for training in the 10x10 cross validation (therefore 193 instances)).
ii) The robustness (ratio between the performance in test and training) is given in Table 3 for the Accuracy (ACC) and the AUC. This ratio indicates that there is no significant overfitting. Moreover, by contrast to methods described in [14, 15] (about LMT and NBT) our algorithm does not require any cross validation for setting parameters.
iii) The only user parameter is the number of cluster (in this paper $K = J$). This point is crucial to help a non-expert to use the proposed method.
iv) The preprocessing complexity (step 1 of Algorithm 1) is $O(d\,m \log m)$, the k-means has the usual complexity $O(d\,m\,J\,t)$ and the complexity for the creation of the local models is $O(d\,m^* \log m*) + O(K(d\,m^* \log dm^*))$ where d is the number of variables, m the number of instances in the training dataset, m^* is the average number of instances belonging to a cluster. Therefore a fast training time is possible as indicated in Table 3 with time given in seconds (for a PC with Windows 7 enterprise and a CPU : Intel Core I7 6820-HQ 2.70 GHz).
v) Only clusters where the information is sufficient to beat the majority vote contain local model. Table 3 gives the percentage of pure clusters obtained at the end of the convergence the K-Means and the percentage of clusters with a local model (if not pure) when performing the 10x10 cross validation (so over 100 results).
vi) Finally, the interpretation of the $\texttt{PKM}_{\texttt{SNB}}$ model is based on a two-level analysis. The first analysis consists in analyzing the profile of each cluster using histograms. A visualisation of the average profile of the overall population

(each bar representing the percentage of instances having a value of the corresponding interval) and the average profile of a given cluster allows to understand why a given instance belongs to a cluster. Then locally to a cluster the variable importance of the local classifier (the weights, W_f, in the SNB classifier) gives a local interpretation.

Datasets	Robustness		Training	Local models			**Datasets**	Robustness		Training	Local models		
(#intances)	ACC	AUC	Time (s)	(1)	(2)	(3)	(#intances)	ACC	AUC	Time (s)	(1)	(2)	(3)
Glass	0.98	0.99	0.07	40.83	23.17	36.0	Waveform	0.97	0.99	0.73	0.00	98.00	2.00
Pima	0.95	0.94	0.05	00.00	88.50	11.5	Mushroom	1.00	1.00	0.53	50.00	50.00	0
Vehicle	0.95	0.97	0.14	07.50	92.25	0.25	Pendigits	0.99	1.00	1.81	0.00	100.00	0
Segment.	0.98	1.00	0.85	28.28	66.28	5.44	Adult	1.00	1.00	3.57	0.00	100.00	0

Table 3. Elements for discussion about local models. (1) Percentage of pure clusters; (2) Percentage of non-pure clusters with a local model; (3) Percentage of non-pure clusters without a local model.

The results of our experiments and the elements (i) to (vi) show that the algorithm $\texttt{PKM}_{\texttt{SNB}}$ is interesting with regards to several aspects. (1) Its predictive performance are comparable to those of the best competing supervised classification methods, (2) it doesn't require cross validation, (3) it deals with the missing values, (4) it operates a features selection both in the clustering step and during the building of the local models. Finally, (5) it groups the categorical features into modalities, thus allowing one to avoid using a complete disjunctive coding which involves the creation of large vectors. Otherwise this disjunctive coding could complicate the interpretation of the obtained model.

The reader may find a supplementary material here: https://bit.ly/2T4VhQw or here: https://bit.ly/3a7xmFF. It gives a detailed example about the interpretation of the results and some comparisons to others predictive clustering algorithms as COBRA or MPCKmeans.

4 Conclusion and perspectives

We have shown how to modify a distance-based clustering technique, such as k-means, into a predictive clustering algorithm. Moreover the learned representation could be used by other clustering algorithms. The resulting algorithm $\texttt{PKM}_{\texttt{SNB}}$ exhibits strong predictive performances most of the time as the state of the art but with the benefit of not having any parameters to adjust and therefore no cross validation to compute. The suggested algorithm is also a good support for interpretation of the data. Better performances can still be expected when the number of clusters is higher than the number of classes. One goal of a work in progress it to find a method that would automatically discover the optimal number of clusters. In addition, we are developing a tool to help visualize the results allowing the navigation between clusters in order to view easily the average profiles and the importance of the variables locally for each cluster.

References

1. Al-Harbi, S.H., Rayward-Smith, V.J.: Adapting k-means for supervised clustering. Applied Intelligence **24**(3), 219–226 (2006)

2. Arthur, D., Vassilvitskii, S.: K-means++: The advantages of careful seeding. In: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 1027–1035 (2007)
3. Been Kim, Kush R. Varshney, A.W.: Workshop on human interpretability in machine learning (whi 2018). In: Proceedings of the 2018 ICML Workshop (2018)
4. Bilenko, M., Basu, S., Mooney, R.J.: Integrating constraints and metric learning in semi-supervised clustering. In: Proceedings of the Twenty-first International Conference on Machine Learning (ICML) (2004)
5. Blockeel, H., Dzeroski, S., Struyf, J., Zenko, B.: Predictive Clustering. Springer-Verlag New York (2019)
6. Bouchard, G., Triggs, B.: The tradeoff between generative and discriminative classifiers. In: IASC International Symposium on Computational Statistics (COMPSTAT). pp. 721–728 (2004)
7. Boullé, M.: A Bayes optimal approach for partitioning the values of categorical attributes. Journal of Machine Learning Research **6**, 1431–1452 (2005)
8. Boullé, M.: MODL: a Bayes optimal discretization method for continuous attributes. Machine Learning **65**(1), 131–165 (2006)
9. Boullé, M.: Compression-based averaging of selective naive Bayes classifiers. Journal of Machine Learning Research **8**, 1659–1685 (2007)
10. Cevikalp, H., Larlus, D., Jurie, F.: A supervised clustering algorithm for the initialization of rbf neural network classifiers. In: Signal Processing and Communication Applications Conference (June 2007), http://lear.inrialpes.fr/pubs/2007/CLJ07
11. Eick, C.F., Zeidat, N., Zhao, Z.: Supervised clustering - algorithms and benefits. In: International Conference on Tools with Artificial Intelligence. pp. 774–776 (2004)
12. Flach, P.: Machine learning: the art and science of algorithms that make sense of data. Cambridge University Press (2012)
13. Hand, D.J., Yu, K.: Idiot's bayes-not so stupid after all? International Statistical Review **69**(3), 385–398 (2001)
14. Kohavi, R.: Scaling up the accuracy of naive-bayes classifiers: a decision-tree hybrid. In: International Conference on Data Mining. pp. 202–207. AAAI Press (1996)
15. Landwehr, N., Hall, M., Frank, E.: Logistic model trees. Mach. Learn. **59**(1-2) (2005)
16. Langley, P., Sage, S.: Induction of selective bayesian classifiers. In: Proceedings of the Tenth International Conference on Uncertainty in Artificial Intelligence. pp. 399–406. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1994)
17. Lemaire, V., Alaoui Ismaili, O., Cornuéjols, A.: An initialization scheme for supervized k-means. In: International Joint Conference on Neural Networks (2015)
18. Lichman, M.: UCI machine learning repository (2013)
19. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, New York (2008)
20. Meilă, M., Heckerman, D.: An experimental comparison of several clustering and initialization methods. In: Conference on Uncertainty in Artificial Intelligence. pp. 386–395. Morgan Kaufmann Publishers Inc. (1998)
21. Milligan, G.W., Cooper, M.C.: A study of standardization of variables in cluster analysis. Journal of Classification **5**(2), 181–204 (1988)
22. Salperwyck, C., Lemaire, V.: Learning with few examples: An empirical study on leading classifiers. In: International Joint Conference on Neural Networks (2011)
23. Van Craenendonck, T., Dumancic, S., Van Wolputte, E., Blockeel, H.: COBRAS: fast, iterative, active clustering with pairwise constraints. In: Proceedings of Intelligent Data Analysis (2018)