Environmental Data Exploration with Data-Mining Techniques: A New Challenge

Jan Žižka

Institute of Biostatistics and Analyses
Masaryk University
Brno, Czech Republic (EU)

Email: zizka@iba.muni.cz
Contents

✔ Raw data and knowledge hidden inside
✔ Large data (pre)processing
✔ Machine learning
✔ Clustering algorithms
✔ WEKA system
✔ Environmental data and clustering
✔ Adaptive resonance theory (ART)
✔ Conclusions
✔ References
Raw data and knowledge hidden inside

Solving real problems involves very often coping with various complexities associated with the original raw data acquired from the primary measurements and observations, e.g.:

- missing sufficient understanding of the problem,
- missing acquaintance with relevant features to create good models or to select the right hypothesis,
- the necessity to process huge volumes of data to mine satisfactory knowledge as the solution,
- the necessity to carry out more processing steps to achieve desired targets satisfactorily,
- changing conditions during getting the raw data,
- uncertainty what and how to do,
- and many other problems.
**Large data (pre)processing**

When obtaining *huge* or *high-dimensional data* related to the solved problem, one usually has to somehow preprocess them before starting looking for a good generalization to mine suitable knowledge.

*Preprocessing* is almost always inevitable—it can, for example, remove irrelevant features or entries with missing values, significantly decrease the number of dimensions or the data volume, eliminate noise, and so like.

Preprocessing can naturally involve several steps before approaching the main processing phase (i.e., the knowledge mining) itself.
Large data (pre)processing (continued)

Typically, the data measurements can provide various entries, some of them might be more, some of them might be less similar.

The similarity among data items allows creating certain groups of the entries that belong to individual different classes—they make clusters containing objects that have something common.

As a result, each group of many similar objects can be profitably substituted by just one representative object, thus significantly decreasing the data volume as well as representing those many substituted objects in each group only by their common features, i.e., reducing the number of dimensions, together with lowering the computational complexity.
Large data (pre)processing (continued)

The fact that we can generalize groups of specific objects so that they can be represented by new individual entities means that certain knowledge was mined from the raw data. Of course, this knowledge does not have to be the final result. However, it can support the next steps.

Large data need (pre)processing by machines. Machines need exact procedures how to mine the knowledge. But what if people, as the machine users, do not know how to mine it? How people obtain knowledge? How, e.g., small children learn the mother language or recognize similar and different things when no sufficiently exact description is possible?

The mystery is in the possibility of learning.
Machine learning

The rules that can tell us what and when to do under certain conditions can be obtained by various ways, for instance, by deduction (when we know the exact and proved knowledge given by a procedure or formula) or by induction (when we can induce the knowledge, e.g., from specific examples by generalization).

Small children learn to differentiate chairs from tables—they are able to create a general image of chair and table, and these images represent groups of particular chairs and tables. Even if a child has never seen a certain particular chair, he or she can recognize it as chair—it is pattern recognition learned from a couple of specific examples.
Machine learning (continued)

So as for machines: Can they learn as well? Yes, they can.

The artificial–intelligence branch called machine learning deals with getting knowledge automatically by machines.

One possibility is also employing particular examples for generalization of data or information. Examples have to be correctly selected and described using their feature values, and the features have to be relevant for getting the desired knowledge.

Unfortunately, not always people can provide ideal training examples—sometimes it is necessary just to use the obtained data without detailed information on them.
Machine learning (continued)

People can often learn using only a low number of carefully selected training examples. Machines usually need much more examples.

On the other hand, when there are data enough (without having any specific knowledge about them), machines are able to reveal some knowledge that can be very useful.

This process may be successfully applied also in such cases when the data are described by a very high number of features (dimensions)—people mostly cannot mine knowledge when there are tens (or even hundreds of thousands) features of examples, esp. when the dimension number is much greater than the example number.
Machine learning (continued)

Machine–learning algorithms can acquire knowledge by two principal ways:

1) _supervised_ learning ("with a teacher") or
2) _unsupervised_ learning ("without a teacher").

Supervised learning needs to know where each example belongs to. The learning procedure uses a _feedback_ to check whether its answer to a question is correct or not.

Unsupervised learning does not use such a feedback; belonging to a certain cluster is formed by "natural groupings" of the input patterns without any explicit supervision.
Machine learning (continued)

Unsupervised learning is used when we do not know what clusters (classes) should be used for the correct categorization (classification) of individual data items. We rely on finding some similarities among the data items—the similarities are given by values of features in some dimensions.

Algorithms based on available training examples applied to the clustering process should find appropriate representations as well as correct clusters.

But how to know in advance the number of clusters and what are the relevant features when we have just the raw data without any specific guiding information?
Clustering

Clustering as the unsupervised learning is useful from several reasons, as the experience with solving various real problems demonstrates:

- Collecting and labeling a large set of sample patterns can be very costly.
- When there are a lot of data, it can be useful firstly to train an algorithm with unlabeled data and only then to use supervision to label the groupings found.
- Patterns can slowly change with time—a classifier should track the changes in an unsupervised mode.
- Clustering can find relevant features for categorization.
- Found clusters can provide valuable initial knowledge for the future data processing.
Clustering (continued)

When a machine should look for clusters, the first question arises: How many clusters are hidden in the data?

If we know it, we can correctly provide this parameter to a selected clustering algorithm and the task is easier.

Unfortunately, we often even do not know what clusters are inside, i.e., how to automatically categorize the available data.

How a cluster is defined? How many members it should contain? How similar the members should be? How to compute the similarity? What features are relevant for setting the similarity? How many clusters could be useful for solving the specific problem?
Clustering (continued)

The membership in a cluster is given by a certain similarity among the members—the future prediction where a new data entry, not used for training, belongs to.

Generally, there are two extreme solutions:

1) Each cluster contains just one data item, so there are as many clusters as the number of the training examples; too specific solution.

2) Only one cluster containing all training examples; too general solution.

Evidently, the correct solution is somewhere between.
Which of the shown hierarchical clustering up–to–bottom is correct? Is the similarity given by color or shape?

And how the situation would be for the bottom–to–up hierarchical clustering?
Clustering (continued)

Clusters can be created by *splitting* the complete data set into subsets, or by *unifying* individual data items and consecutively their representatives.

A clustering algorithm must know when to stop the splitting or unifying procedure—however, it is not easy to define this parameter, so we usually have to carry out some experiments to obtain a satisfactory result.

In addition, the clustering process can be even more difficult when the data items come sequentially, on–line, and we do not know in advance when there will be data entries enough to stop learning the clusters. And what if, after stopping, some other entries would come to make another new cluster?
**Clustering (continued)**

One of elementary but very popular clustering methods is **k–means**. Its goal is to find \( k \) cluster centers (or \( k \) mean vectors \( \mu_1, \mu_2, \ldots, \mu_k \)) using available training data.

**k–means** is a form of stochastic hill–climbing in the log–likelihood function. All data items in each cluster are represented by the cluster’s center. Before starting, the iterative algorithm needs to know how many clusters we want—we have to “guess” \( k \) in advance.

Of course, various \( k \)’s can give different results, so it is useful to try several \( k \)’s and then to select the most convenient result. A new unknown data entry belongs to a cluster with the nearest center to the categorized item.
**Clustering (continued)**

The $k$–means algorithm can be described by a pseudo–code:

```plaintext
begin initialize $n$, $k$, $\mu_1$, $\mu_2$, …, $\mu_k$
    do classify $n$ samples according to nearest $\mu_i$
        recompute $\mu_i$
    until no change in $\mu_i$
    return $\mu_1$, $\mu_2$, …, $\mu_k$
end
```

In practice, the number of iterations is usually much less than the number of training data items. At the beginning, centers are chosen randomly. All instances are assigned to their closest cluster center according to the ordinary Euclidean distance metric. Then, the centroids (means) of the instances in each cluster are recalculated—new centers.
Clustering (continued)

$k$-means clustering:

the dots indicate parameter values after different iterations;

6 starting points lead to local maxima;

2 lead to a saddle point near $\mu = 0$. 

equal log-likelihood values
Clustering (continued)

Clustering methods can or cannot provide good results—one has to be careful when evaluating and interpreting outcomes.

We use machine–learning methods if there are missing possibilities of applying conventional mathematical procedures (e.g., modeling, statistical processing, and so like).

Insufficient knowledge about the solved problem may easily mislead a researcher. For example, the following figure shows how the real clusters look like—but mechanically applying available algorithms (e.g., the popular $k$–means method) based on the Euclidean (or other) distance could provide wrong results.
Centroid–based methods, like $k$–means clustering, would evidently give unnatural results.
Clustering (continued)

Similarity measures are important for clustering as a cluster is defined by similar members.

How to measure the similarity between data samples?

We have more possibilities, which is good (as we can select a convenient method) as well as bad (as we often do not know which method would be the best one).

Here we can mention briefly just some of the similarity measures; much more detailed information is available in literature.
Clustering (continued)

The most obvious measure of the (dis)similarity between two data items is the distance between them. We have to define a suitable metric. Close items (the nearest neighbours) form a cluster, distant points form different clusters.

The elementary metric is the *Euclidean* one. Another possibility is the *Manhattan* (city block) metric—the sum of the absolute distances along each of the coordinate axes, or an alternative called the *Mahalanobis* distance based on the data itself.

Other similarity can be based on the *cosine* between two vectors, or *Tanimoto* distance for binary–valued features (based on the number of shared attributes).
Clustering (continued)

Sometimes it is not necessary to insist on assigning each data point to only one cluster.

We can accept a data entry belonging to more clusters with a graded membership in a cluster. This method is called the fuzzy $k$–means clustering. The memberships in clusters are given by normalized probabilities for each point (for more details see literature in the References section).

Many clustering algorithms, including the $k$–means with various modifications, are now available as software. For example, the free machine–learning set of tools WEKA offers some testified methods.
**WEKA system**

At this moment (September, 2006), WEKA includes the following clustering methods trained by data examples:

*Cobweb, DBScan (a density based algorithm), EM (expectation maximization), FarthestFirst, OPTICS, SimpleKMeans, XMeans, MakeDensityBasedClusterer.*

Because of the famous *no–free–lunch* theorem, we usually do not know in advance which algorithm would provide the best expected results, so WEKA conveniently supports experiments with data and algorithms.
**WEKA system** (continued)

*WEKA* is in the Java language and can be freely downloaded from the following URL:

http://www.cs.waikato.ac.nz/ml/weka/

This software is also used in some *bioinformatics–oriented subjects* taught at the *Institute of Biostatistics and Analyses* (IBA), Masaryk University, Brno, Czech Republic.

*WEKA* is available both for Windows and Linux operating systems. It can be recommended for introduction into machine–learning methods applied to biomedical data as well as for advanced research.
Environmental data and clustering

Environmental data are taken from *environment*—we can informally understand something like:

*Environment, thin layer of life and life-supports called the biosphere, including the Earth’s air, soil, water, and living organisms.*

These data can be processed by standard clustering methods, especially when there are large volumes of measured or observed samples during a longer time period, which is typical for the environmental data.

Also, such data can vary during the time, e.g., if we investigate a certain area through seasons in a year.
Environmental data... (continued)

Typically, researchers can measure what occurs or does not occur in a certain area (binary yes/no values), or the frequency of measured items (numeric values, generally real numbers). If we look for similar areas, we can apply a clustering method.

If similar areas are defined only by occurrence of certain features, we cannot always say that two areas are almost identical when there are no certain features:

```
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
```

Can we say these two 25–dimensional data items are practically identical? Not always! But they look similar, don’t they?
Environmental data... (continued)

Let us look again at the items with the 25 monitored features:

```
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
```

The problem is that missing features do not always define the identity: What if the both items contain only zeros? Can we say two areas without any fauna and flora are identical? Sometimes yes, but often no (yes, we work also with *vague information*).

From the other point of view, ‘1’ and ‘0’ does not have to represent the *symmetrical information*.

How to build clusters in such situations?
In addition, monitoring certain areas involves often obtaining the *on–line data*. Thus, how long should we wait for accumulating the data before starting building the clusters?

Can we simply somehow create the clusters “dynamically” as the data come?

These above–mentioned problems represent typically (but not always) the environmental data (but not only them).

Fortunately, there are methods that enable also processing of such data types. One of those algorithms is a modern method called the *Adaptive Resonance Theory* (ART).
Adaptive Resonance Theory (ART)

The algorithm employs a set of initialized prototype vectors, $P_i$, $i = 1, \ldots, n$, as the representatives of individual clusters. The symbol $n$ means the maximal number of clusters.

If an incoming data item cannot be categorized into an already existing cluster, it starts a new cluster. The categorization depends strongly on a defined similarity coefficient. The similarity is given by the number of 1’s in the categorized item and in the cluster representative (two items with only 0’s are considered as undefined).

The main advantage of ART: It is not necessary to guess in advance the number of clusters. Moreover, the sizes of created clusters are also automatically modified.
Adaptive Resonance Theory (continued)

The main disadvantage of ART: The result can strongly depend on the order of the submitted training samples. Therefore, it is convenient to try to generate the clusters using different orders of samples, if possible.

ART–1 parameters:

- \( n \) … the number of prototypes \( P \) (clusters)
- \( l \) … the vector length (all vectors have the same length)
- \( \rho \) … vigilance, \( 0.0 < \rho \leq 1.0 \) (the similarity threshold of two binary vectors)
- \( \beta \) … a small natural number (for the same similarity of an item against more clusters, a cluster with more 1’s is preferred)
- \( P \) … a prototype vector (it represents a “center” of a cluster)
- \( E \) … a vector of a training sample
Adaptive Resonance Theory (continued)

The different vigilance parameter can, of course, bring different number of different clusters.

At the beginning, when there is no cluster \( P \), the first incoming sample, \( E_0 \), creates the initial cluster, \( P_0 \). In the following steps, every submitted sample is compared, using the vigilance \( \rho \), with all existing clusters how many 1’s are at the same positions in the pair of vectors (a sample and a cluster). The similarity between two vectors is defined as:

\[
\frac{||P_i \cap E||}{\beta + ||P_i||} > \frac{||E||}{\beta + l}
\]

where \( ||X|| \) means the number of 1’s in a vector and \( \cap \) means the binary logical multiplication.
Adaptive Resonance Theory (continued)

When the condition given by Eq. 1 is fulfilled, then the parameter vigilance $\rho$ determines if $E$ should really be a member of $P_i$:

$$\frac{\|P_i \cap E\|}{\|E\|} \geq \rho$$  \hspace{1cm} (2)

The boundary $\rho$ also determines the size of a cluster.

If the criterion (2) is not fulfilled for any prototype, then the sample starts a new cluster. If Eq. 2 is fulfilled, the sample is included into the corresponding cluster by the bit–like logical multiplication $P_i \cap E \rightarrow P_i$. 
Adaptive Resonance Theory (continued)

Larger clusters lead to a smaller number of them and vice versa.

The vigilance parameter $\rho$ determines the generalization degree of clustering.

The right values of $\rho$ and $\beta$ should usually be found experimentally which is a typical feature of machine–learning algorithms. The goal is to find the optimal combination of both parameters.

When all the incoming samples were processed, we must test them towards the generated clusters—after processing of a certain sample, new clusters might be generated and what if a processed sample would better belong to a new cluster?
Adaptive Resonance Theory (continued)

If the testing results into no changes, the process is finished (usually it is good to define the maximal (but large enough) number of iterations to avoid the endless looping).

An example:

Let there are the following prototypes $P_1$ and $P_2$ plus the instance $E$, and let the parameters be $\beta = 1$, $\rho = 0.3$, $l = 7$:

$P_1 = \{1,0,0,1,1,0,1\}$  $P_2 = \{1,1,0,0,0,1,0\}$  $E = \{1,1,1,0,0,1,0\}$. 
Adaptive Resonance Theory (continued)

Then the Eq. 1 and 2 provide the following results:

\[ \frac{||P_0 \cap E||}{\beta+||P_0||} = \frac{1}{1+4} = \frac{1}{5}, \]
\[ \frac{||E||}{(\beta+1)} = \frac{4}{1+7} = \frac{4}{8}; \quad 1/5 > 4/8 \rightarrow false; \]

\[ \frac{||P_1 \cap E||}{\beta+||P_1||} = \frac{3}{1+3} = \frac{3}{4}, \]
\[ \frac{||E||}{(\beta+1)} = \frac{4}{1+7} = \frac{4}{8}; \quad 3/4 > 4/8 \rightarrow true \]

\[ \frac{||P_1 \cap E||}{||E||} = \frac{3}{4}, \quad 3/4 \geq 0.3 \rightarrow true \]

Therefore \( E \) belongs to \( P_1 \):

\[ P_1 \cap E = \{1,1,0,0,0,1,0\} \cap \{1,1,1,0,0,1,0\} = \{1,1,0,0,0,1,0\} = P_1 \]
Adaptive Resonance Theory (continued)

If, for example, $\rho$ would be 0.8, then $E$ should be more similar to be accepted into the cluster $P_1$; $E$ would be rejected and it would create a new cluster, say $P_2$.

For a very high vigilance parameter $\rho = 1.0$, the agreement in 1’s should be perfect.

The higher vigilance means the higher “granularity” of clusters.

The selected cluster must “resonate” with the included sample.
Adaptive Resonance Theory (continued)

An example with real data describing occurrence of fauna in certain areas in the Czech Republic. It is up to a user what number of clusters is finally selected—experts in biology should say the last word. Of course, the cluster contents is known.
Adaptive Resonance Theory (continued)

It is also possible to make ART–1 behave symmetrically with respect to 0 and 1 inputs: Simply duplicate the inputs and switch 0’s and 1’s in the second copy—it makes all patterns have the same number of 0’s and 1’s.

If a user needs to cluster data described numerically, then the method called ART–2 is available. The input data are normalized and selected values are suppressed below a certain threshold. This scales the data and eliminates small noise signals. Prototypes are updated by moving them a fraction $\beta$ toward a new training example falling in that prototype’s class. It is a modification of ART–1 (details can be seen in literature).
Conclusions

Modern machine–learning methods as the ART can help also in difficult cases that are typical also for environmental data.

ART algorithms support also the decision how many clusters one should use as well as they help avoid situations when the missing occurrence of features brings actually no information.

During the present time, ART is successfully applied to the environmental data preprocessing and processing.
References


