Experimental Platform for Intelligent Computing (EPIC)

Javier A. Hernández-Castaño[1], Yenny Villuendas-Rey[1], Oscar Camacho-Nieto[1], Cornelio Yáñez-Márquez[2]

[1] Instituto Politécnico Nacional,
Centro de Innovación y Desarrollo Tecnológico en Cómputo, Ciudad de México, Mexico

[2] Instituto Politécnico Nacional,
Centro de Investigación en Computación, Ciudad de México, Mexico

{javierhc92, coryanez}@gmail.com, {yvilluendasr, ocamacho}@ipn.mx

Abstract. This paper presents the architecture and user interface of a novel Experimental Platform for Intelligent Computing (EPIC). Unlike the two most popular platforms (WEKA and KEEL), the proposed EPIC tool has a very friendly user interface, and offers some advantages with respect to existing tools for Intelligent Computing experiments. In particular, EPIC handles mixed and incomplete data directly, without preprocessing, and its architecture supports multi-target supervised classification and regression. It also contains a module for two dimensional dataset visualization, which includes the visualization of the decision frontier for several supervised learning algorithms.

Keywords. Experimental tools, EPIC, WEKA, KEEL, intelligent computing, supervised classification.

1 Introduction

The expression “Computational Intelligence” was first introduced by James Bezdek in a seminal paper [1], where he enunciate the foundations of Computational Intelligence, as well as the differences between this new discipline and Artificial Intelligence. However, it should be noted that the expression coined by Bezdek only changes the qualifying adjective to the word Intelligence, which from "Artificial", it becomes "Computational". Within this context, some specialized authors have defined “Computational Intelligence” as the set of computational models and intelligent tools capable of accepting data coming from sensors, in order to process them efficiently, to generate reliable, fast, and highly failure tolerant responses [2].

Within the Alfa-Beta group (to which the authors of this work belong), we have reflected deeply on these facts. We have discussed at length the contents of scientific works within Computational Intelligence, which span a variety of subjects; for example, topics as supervised classification [3, 4], data preprocessing [5-7], data associations [11-13], time series mining [14-16] and data streaming [17-19], among others.

We have concluded that, in all cases of the sample examined, the authors agree with [2], in the sense that they develop or apply "*intelligent* tools" and "*computational* models". In this context, an interesting result that we have arrived at, is that the authors do not mention that *intelligence* is computational, but rather that the *models are computational*; and in addition, they do not mention *intelligence* as a noun, but that it is affirmed that *tools are intelligent* (here, the word intelligent is used as a qualifying adjective).

All these facts and serious discussions have led the members of the Alfa-Beta Group to conclude, in a responsible manner, that the phrase "Computational Intelligence" does not faithfully reflect the essence of the works included in this discipline. We affirm categorically that the phrase "Intelligent Computing" is much more adequate to express the essence of the mentioned contents. Thus, from here on we will use the expression Intelligent Computing (IC).

For the development of new IC models and algorithms, it is necessary to compare them with respect to existing similar models; and for this task, several researching supporting tools have been developed. Among the most popular tools are WEKA [20] and KEEL [21], which fasten the researching process, due to they include existing algorithms and procedures. However, the researchers in the IC community suffer from numerous functionality insufficiencies.

The proposal of this research consists on the creation of a software prototype to execute experiments in IC. The proposed prototype (EPIC), keeps the main functionalities and characteristics of WEKA and KEEL, and overcomes some of their insufficiencies. Besides, EPIC includes additional IC algorithms.

EPIC has a simple yet effective architecture, capable of fulfilling the needs of users; in particular, the need of directly handling mixed and incomplete data, without any data transforming or preprocessing, and the need of handling data belonging simultaneously to several decision attributes (multi-target classification).

In addition, the developed prototype has an Input/Output interface compatible with the data files managed by WEKA and KEEL. It also has a module for data transforming, a module for data partitioning by several validation techniques, and a user friendly interface to visualize the results of classification algorithms over two dimensional (2D) data. Additionally, EPIC has a user interface to develop supervised classification experiments, which is friendlier and has more functionalities than the ones by WEKA and KEEL.

2 Related Works

Table 1 summarizes some of the characteristics of existing tools to support research in IC. The characteristics considered are the presence of a user interface (UI), if the tool has not cost (Free) and if the tool requires internet access in order to be executed (Internet).

In this research, we focuses on tools having user interfaces. This is due to we consider that the researchers must have access to a friendly environment, which allows the effortless design and execution of experiments, and also to be easy to explain to undergraduate and postgraduate students in fields related to IC, in a way such that the fast and reliable knowledge acquisition is guaranteed.

Table 1. Most used tools for IC

Tool	UI	Free	Internet
Accord [22]	-	X	-
Amazon - Machine Learning [23]	X	-	X
Apache Singa [24]	-	X	X
Azure ML Studio [25]	X	-	X
Caffe [26]	-	X	X
KEEL [21]	X	X	-
Mahout [27]	-	X	-
ML Pack [28]	-	X	-
Oryx [29]	-	X	-
Pattern [30]	-	X	-
Scikit-learn [31]	-	X	X
Shogun [32]	X	X	X
Spark MLib [33]	-	X	-
Tensor Flow [34]	-	X	-
Theano [35]	-	X	-
WEKA [20]	X	X	-

On the other hand, we want the tools to be free of charges, and to do not require an internet connection, in order to execute experiments off-line, and without depending on external networks. In addition, we want the data to be protected.

According to the previous analysis (Table 1), the tools fulfilling the three main requirements (UI, Free and no Internet) are WEKA and KEEL. However, both tools had some lack of functionalities, and an architecture design which difficult their daily use for handling mixed data directly, and to include algorithms such as associative memories. With this research, we aim at successfully solve such problems.

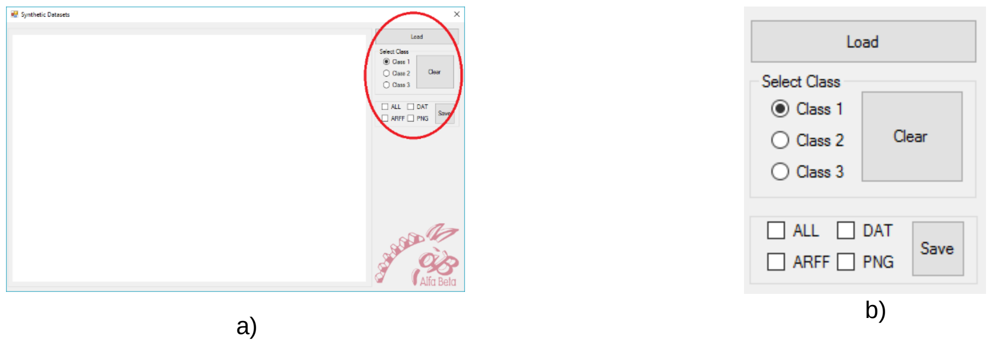


Fig. 1. User Interface of the Synthetic Datasets module of EPIC. In a) is shown the overall view, while in b) is shown a zoom of the highlighted area

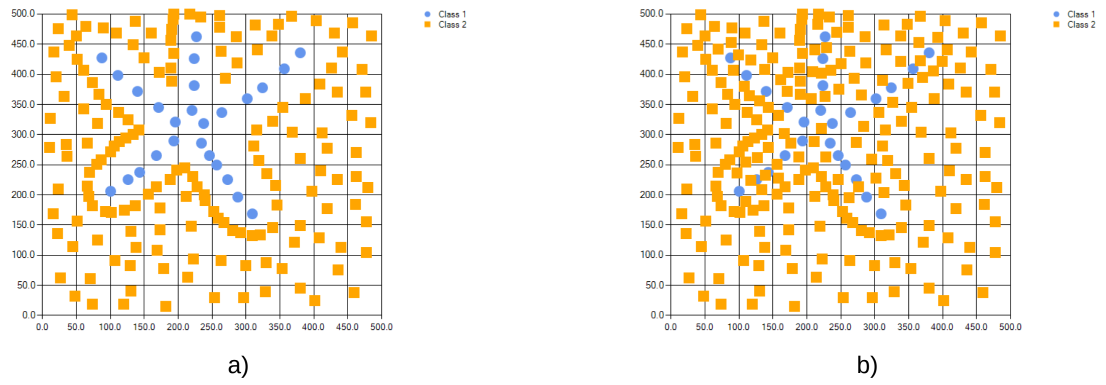


Fig. 2. Example of datasets obtained with the Synthetic Datasets module of EPIC. In a) is shown a clover dataset, while in b) is shown an overlapped version of the dataset in a)

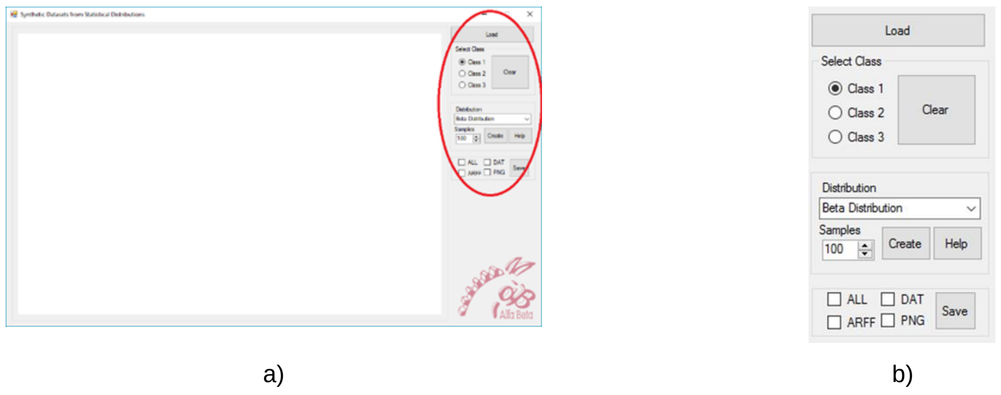


Fig. 3. User Interface of the Synthetic Datasets from Statistical Distributions module of EPIC. In a) is shown the overall view, while in b) is shown a zoom of the highlighted area

Considering the above, and carrying out a deep analysis of both tools, we found out that some of the drawbacks of WEKA are:

1. It does not allow the visualization of classification results of the instances in two dimensions (2D).

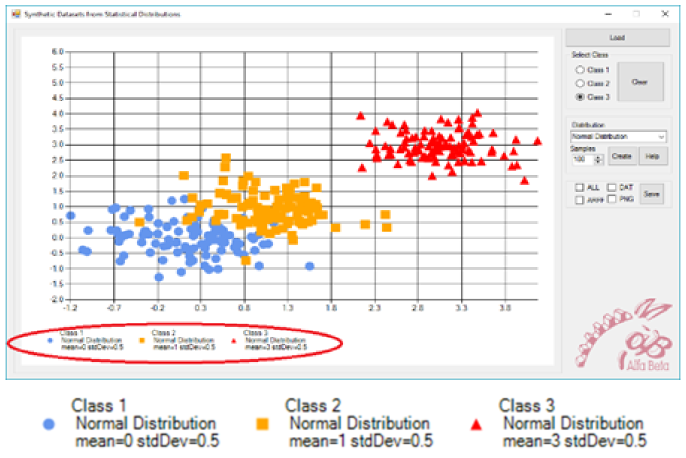
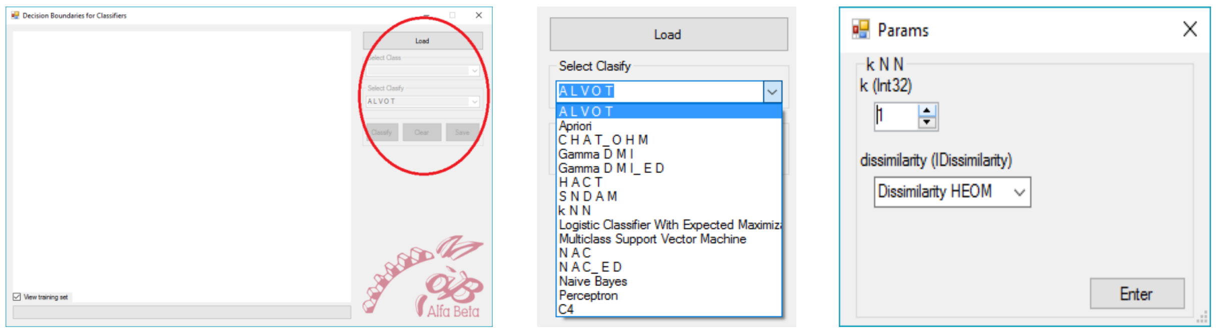


Fig. 4. Example of dataset obtained with the Synthetic Datasets from Statistical Distribution module of EPIC. In a) is shown the dataset, while in b) is zoomed the highlighted legend



a) b) c)

Fig. 5. User Interface of the Decision Boundaries for Classifiers module of EPIC. In a) is shown the overall view, while in b) is shown a zoom of the highlighted area. Note that the menu displays all available learning algorithms. In c) is shown another Windows form displayed to enter the parameters of the selected classifier, in this case, for kNN classifier

2. It does not allow the use of dissimilarity functions for mixed data descriptions (it only have distances, to be computed over real data).
3. It arbitrarily handles mixed and incomplete data (the architecture assume that the feature values of instances are an array of doubles, and it converts the data to fulfill the architecture requirement).
4. It does not include any associative supervised classifier.
5. Its architecture does not allow associating an instance with something apart from a single class label.
6. It does not have the functionality to create synthetic data.
7. It does not allow serializing data partitions.

On the other hand, KEEL tool also have some drawbacks, as follows:

1. It does not allow the visualization of instance classification results in two dimensions (2D).
2. It does not include any associative supervised classifier.
3. It does not have the functionality to create synthetic data.

4. It converts mixed and incomplete data (the architecture assume that the feature values of instances are an array of doubles).
5. It does not allow changing the dissimilarity function in the Distribution optimally balanced stratified cross validation (Dob-SCV), data partitioning procedure.

The new EPIC platform overcomes these drawbacks of WEKA and KEEL.

3 Methods and Materials

In this research, we have decided to start the creation of EPIC from scratch, in order to develop an effective solution to the architectural problems shown by WEKA and KEEL.

We decide to use C# programming language, and the Integrated Development Environment (IDE) *Visual Studio Community* 2017, due to the facilities they offer to create a tool with a very user-friendly interface. Despite C# is not a multiplatform language, we consider that its use will no represent a difficult, due to the widely extension of Windows operating system in Mexico and the rest of the world.

On the other hand, the IDE used is free, and its license allows the development of academic and researching software.

3.1 Core Architecture of the Tool

In order to make experiments with IC algorithms, one of the main users' requirements is the capability of directly handling mixed as well as incomplete data; besides, users also want to handle multiple target attributes (many decisions attributes) and also to associate an instance with some else than a decision class label (for instance, to create an auto-associative memory).

To offer a satisfactory response to such user requirements, we have designed a software core architecture, which includes the classes to handle mixed as well as incomplete attribute values, and to handle several decision attributes.

In addition, we consider the existence of several kinds of classes. Thus, we can directly model supervised classification problems (where the decision attribute has nominal labels), regression problems (where the decision attribute has numeric labels), among others.

Additionally, we consider the possibility that a dataset has more than one decision attribute (as in multi-target classification problems).

It allows us to directly implement multi-target classification algorithms (such as ALVOT [36-40]), which does not require to convert the multiclass problem into several single class problems. This is a clear architectural advantage over some existing tools, such as MEKA [41].

4 Results and Discussion

In this section, we offer a general description of the user interface of the proposed Experimental Platform for Intelligent Computing (EPIC), as well as its functioning. It is worth noting that EPIC is fully Input/Output compatible with both WEKA and KEEL; i.e., EPIC handles. ARFF (native file format of WEKA) and DAT (native file format of KEEL) files; thus, EPIC is able to write and read files generated by both WEKA and KEEL.

4.1 User Interface of EPIC

EPIC has three modules at this time. In the first module, it is possible to manually create a dataset in two dimensions (x-axis and y-axis), in a way fully controlled by the user (Figure 1). It is worth mentioning that neither WEKA not KEEL have such functionality.

Having a two dimensional dataset, designed in a way such that it fully fulfils the current researchers needs, turn to be really useful, due to, in several times, it is necessary to study the behavior of algorithms under certain data configurations.

For such tasks, it is convenient to have a friendly user interface allowing to design the datasets with the desired spatial configuration, and to export such dataset to well-known and popular file formats.

The synthetic dataset created with EPIC can be exported in .ARFF format (used by WEKA) and in a .DAT format (used by KEEL). By this, EPIC guaranteed a full file compatibility.

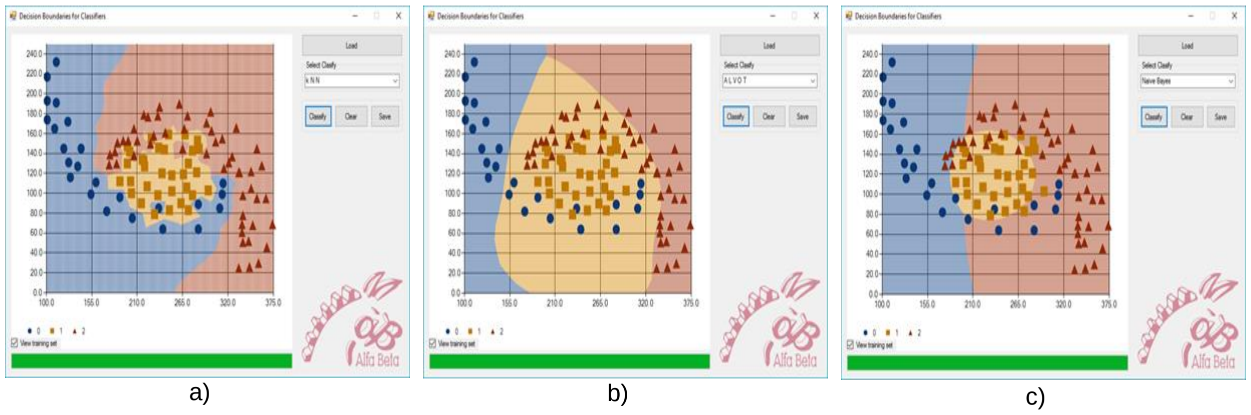


Fig. 6. Decision Boundaries for Classifiers in EPIC, showing training instances. In a) for a kNN classifier, in b) for a SNDAM classifier and in c) for the Naïve Bayes Classifier

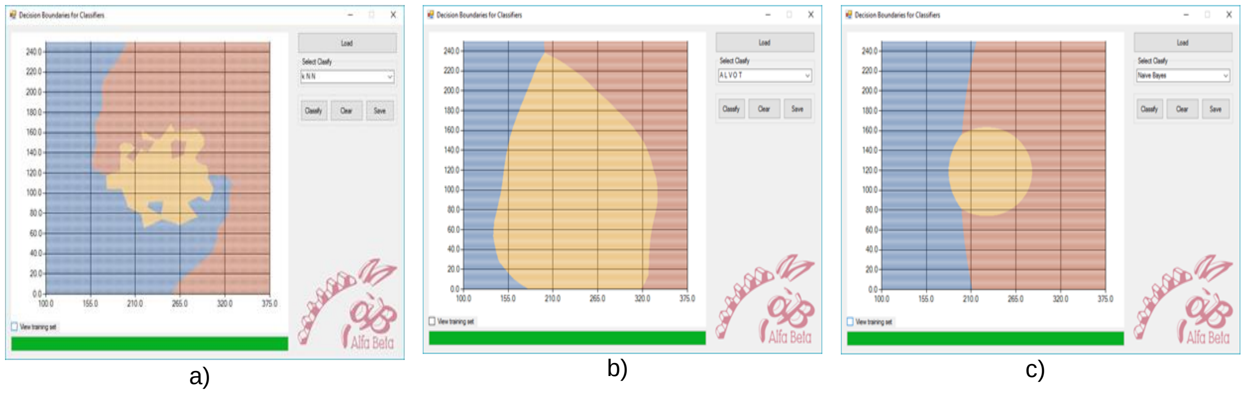


Fig. 7. Decision Boundaries for Classifiers in EPIC, without showing training instances. In a) for a kNN classifier, in b) for a SNDAM classifier and in c) for the Naïve Bayes Classifier

In addition, it is possible to export in a .PNG format, the image of the synthetic dataset created. This option of exporting a graphical representation of the 2D dataset is extremely useful, due to it allows the direct incorporation of such images in researching papers, thesis, technical reports, and other important documents.

As shown in Figure 2, EPIC shows the created datasets with class labels of different shapes and colors (in the example, there are two classes: Class 1, blue-circles, and Class 2, mustard-squares). The class legend is shown in the upper right corner of the user interface.

Using this, the user can visually appreciate the data distribution, and to carry out the desired corrections and comparisons.

The second module of EPIC allows to create a synthetic dataset from known statistical data distributions. For such task, we used some of the statistical distributions offered by Accord [22]. In this module, the user defines the desired amount of samples, and the desired distribution for each class.

It allows obtaining in a fast and extremely simple way, huge amounts of data points, following the desired data distribution. Figure 3 shows the user interface of the Synthetic Datasets from Statistical Distributions module of EPIC.

As previous module, the obtained dataset can be exported in .ARFF and .DAT file formats (Figure 3b), and the corresponding graphical representation can be exported in .PNG, making easier its further use. Figure 4 shows an example of a dataset created with EPIC, with three classes (Class 1 blue-circles, Class 2 mustard-squares, and Class 3 red-triangles), having 100 instances each, and following a Normal distribution, but with different means and standard deviations.

The legend shows the name of the classes, as well as the name of the used statistical distribution, and the corresponding parameters and values.

In the example, the parameters for a Normal distribution are mean and standard deviation.

It is important to highlight that neither WEKA nor KEEL had functionalities for creating synthetic data, designed to fit the user needs.

The third module of EPIC reads a two dimensional dataset in both. ARFF or .DAT file formats, and allows to visualize the decision boundaries of several supervised classifiers. For this, EPIC trains the corresponding classifier with the dataset read, and classifies the points in the graphic area.

In Figure 5, we show the user interface of the decision boundaries visualization module of EPIC. This functionality is extremely useful for researchers, due to it allows to visualize rapidly the uncertainty zones of the supervised classifiers, and to study interesting behaviors of the algorithms, as well as to deeply understanding their functioning.

This is very important for complex datasets, having imbalanced data, and with small disjoints in the data.

As show in figures 6 and 7, the Decision Boundaries for Classifiers module of EPIC helps the user in the analysis of the behavior of several supervised classification algorithms, and allows stablishing comparisons with the desired data. In addition, this module can be of interest for researchers and practitioners within IC, by graphically showing the functioning of the algorithms, in the scenarios desired by the user.

5 Conclusions and Future Works

In this paper, we introduce a novel tool for Intelligent Computing research experiments. The tool, named EPIC, offers several desirable functionalities. It includes three modules for data processing, data generation and for the visualization of results of supervised classification algorithms. The architecture of EPIC allows to directly handling mixed and incomplete data, also having multiple decision labels. It allows the inclusion of supervised learning algorithms for multiple target classification and regression tasks.

As future works, we want to extend the EPIC tool, by adding more functionalities, including a module for the execution of experiments to evaluate the performance of supervised and unsupervised learning algorithms over multiple datasets.

The EPIC tool is under development, and the current release can be found at the Alpha Beta Group web site, available at http://www.alfabeta.org.

Acknowledgements

The authors like to thank the Instituto Politécnico Nacional (Secretaría Académica, Comisión de Operación y Fomento de Actividades Académicas, Secretaría de Investigación y Posgrado, Centro de Investigación en Computación, and Centro de Innovación y Desarrollo Tecnológico en Cómputo), the Consejo Nacional de Ciencia y Tecnología, and Sistema Nacional de Investigadores for their financial support.

References

1. **Bezdek, J.C. (1994).** What is Computational Intelligence? *IEEE Press Computational Intelligence Imitating Life*, pp. 1–12.
2. **Konar, A. (2005).** Computational intelligence principles, Techniques and Applications. Springer Berlin Heidelberg.
3. **Wang, X., et al. (2017).** Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. *IEEE Conference on Computer Vision and Pattern Recognition.*
4. **Mondal, A., Khare, D., & Kundu, S. (2017).** Identification of Crop Types with the Fuzzy Supervised Classification Using AWiFS and LISS-III Images. *Environment and Earth Observation*, Springer, pp. 73–86. DOI: 10.1007/978-3-319-46010-9_5.
5. **Xue, B., Zhang, M., Browne, N.W., & Yao, X. (2016).** A survey on evolutionary computation approaches to feature selection. *IEEE Transactions*

on Evolutionary Computation, Vol. 20, No. 4, pp. 606–626. DOI:10.1109/TEVC. 2015.2504420.

6. **Fernández, A., et al. (2017).** A Pareto Based Ensemble with Feature and Instance Selection for Learning from Multi-Class Imbalanced Datasets. *International Journal of neural systems,* Vol. 27, No. 6, pp. 1–21. DOI: 10.1142/S0129065717500289.
7. **Rosales-Pérez, A., García, S., Gonzalez, J.A., Coello-Coello, C.A., & Herrera, F. (2017).** An Evolutionary Multi-Objective Model and Instance Selection for Support Vector Machines with Pareto-based Ensembles*. IEEE Transactions on Evolutionary Computation*, Vol. 21, No. 6, pp. 863–877. DOI: 10.1109/TEVC. 2017.2688863.
8. **Villarreal, S.E.G. & Schaeffer, S.E. (2016).** Local bilateral clustering for identifying research topics and groups from bibliographical data. *Knowledge and Information Systems*, Vol. 48, No. 1, pp. 179–199. DOI: 10.1007/s10115-015-0867-y.
9. **Hasenstab, K., Sugar, C., Telesca, D., Jeste, S., & Şentürk, D. (2016).** Robust functional clustering of ERP data with application to a study of implicit learning in autism. *Biostatistics*, Vol. 17, No. 3, pp. 484–498. DOI: 10.1093/ biostatistics/kxw002.
10. **Golman, R. & Klepper, S. (2016).** Spinoffs and clustering. *The RAND Journal of Economics*, Vol. 47, No. 2, pp. 341–365. DOI: 10.1111/ 1756-2171.12130.
11. **Ramírez-Rubio, R., Aldape-Péreza, M., Yáñez-Márquez, C., López-Yáñez, I., & Camacho, O. (2017).** Pattern classification using smallest normalized difference associative memory. *Pattern Recognition Letters*, Vol. 93, pp. 104–112. DOI: 10.1016/j. patrec.2017.02.013.
12. **Cleofas-Sánchez, L., Sánchez, S., García, V., & Valdovinos, R.M. (2016).** Associative learning on imbalanced environments: An empirical study. *Expert Systems with Applications*, Vol. 54, pp. 387–397. DOI: 10.10 16/j.eswa.2015.10.001.
13. **Uriarte-Arcia, A.V., López-Yáñez, I., & Yáñez-Márquez, C. (2014).** One-hot vector hybrid associative classifier for medical data classification*. PloS one*, Vol. 9, No. 4. DOI: 10.1371/journal.pone.0095715.
14. **Salmeron, J.L. & Froelich, W. (2016).** Dynamic optimization of fuzzy cognitive maps for time series forecasting*. Knowledge-Based Systems*, Vol. 105, pp. 29–37. DOI:10.1016/j. knosys.2016.04.023.
15. **Sheremetov, L.B., González-Sánchez, A., López-Yáñez, I., & Ponomarev, A.V. (2013).** Time series forecasting: applications to the upstream oil and gas supply chain. *IFAC Proceedings*, Vol. 46, No. 9, pp. 957–962. DOI: 10.3182/20130619-3-RU-3018.00526.
16. **Cao, L.J. & Tay, F.E.H. (2003).** Support vector machine with adaptive parameters in financial time series forecasting. *IEEE Transactions on neural networks*, Vol. 14, No. 6, pp. 1506–1518. DOI: 10.1109/TNN.2003.820556.
17. **Uriarte-Arcia, A.V., López-Yáñez, I., Yáñez-Márquez, C., Gama, J., & Camacho-Nieto, O. (2015).** Data stream classification based on the gamma classifier*. Mathematical Problems in Engineering*, Vol. 2015. DOI: 10.1155/2015/ 939175.
18. **Guha, S. & Mishra, N. (2016).** Clustering data streams. *Data Stream Management*, Springer, pp. 169–187.
19. **Baccarelli, E., Cordeschi, N., Mei, A., Panella, M., Shojafar, M., & Stefa, J. (2016).** Energy-efficient dynamic traffic offloading and reconfiguration of networked data centers for big data stream mobile computing: review, challenges, and a case study. *IEEE Network*, Vol. 30, No. 2, pp. 54–61. DOI: 10.1109/MNET.2016.7437025.
20. **Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I.H. (2009).** The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, Vol. 11, No. 1, pp. 10–18. DOI: 10.1145/1656274.1656278.
21. **Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., & Herrera, F. (2011).** KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic & Soft Computing*, Vol. 17, pp. 255–287.
22. **Souza, C.R. (2012).** A Tutorial on Principal Component Analysis with the Accord. NET Framework, *Department of Computing, Federal University of Sao Carlos*, Technical Report.
23. **Amazon. (2017).** *Amazon Machine Learning Service*. https://aws.amazon.com/es/machine-learning/.
24. **Ooi, B.C., Tan, K.L., Wang, S., Wang, W., Cai, Q., Chen, G., Gao, J., Luo, Z., Tung, A.K.H., Wang, Y., Xie, Z., Zhang, M., & Zheng, K. (2015).** SINGA: A distributed deep learning platform. *Proceedings of the 23rd ACM international conference on Multimedia,* pp. 685–688. DOI: 10.1145/2733373.2807410.
25. **Barnes, J. (2015).** *Microsoft Azure Essentials Azure Machine Learning.* Microsoft Press*.*
26. **Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., & Darrell, T. (2014).** *Caffe: Convolutional Architecture for Fast Feature Embedding. Proceedings of the 22nd ACM*

international conference on Multimedia, pp. 675–678. DOI: 10.1145/2647868.2654889.

27. **Apache Software Foundation (2014).** Mahout. https://mahout.apache.org/.
28. **Curtin, R.R., Cline, J.R., Slagle, N.P., March, W.B., Ram, P., Mehta, N.A., & Gray, A.G. (2013).** MLPACK: A scalable C++ machine learning library. *Journal of Machine Learning Research,* Vol. 14, pp. 801–805.
29. **Oryx Project (2014).** *Oryx 2: Lambda architecture on Apache Spark*. Apache Kafka for real-time large scale machine learning. http://oryx.io.
30. **Smedt, T.D. & Daelemans, W. (2012).** Pattern for Python. *Journal of Machine Learning Research*, Vol. 13, pp. 2063–2067.
31. **Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011).** Scikit-learn: Machine learning in Python. *Journal of machine learning research*, Vol. 12, pp. 2825–2830.
32. **Sonnenburg, S. et al. (2017).** *shogun-toolbox/shogun: Shogun 6.1.0* (Version shogun_6.1.0), http://doi.org/10.5281/zenodo.1067840.
33. **Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., et al. (2016).** Mllib: Machine learning in apache spark. *Journal of Machine Learning Research*, Vol. 17, No. 34, pp. 1–7.
34. **Abadi, M. et al. (2016).** TensorFlow: A System for Large-Scale Machine Learning. *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI),* Vol. 16, pp. 265–283.
35. **Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., & Bengio, Y. (2010).** Theano: A CPU and GPU math compiler in Python. *Proceeding 9th Python in Science Conference*, pp. 1–7.
36. **Carrasco-Ochoa, J.A. & Martínez-Trinidad, J.F. (2003).** Editing and training for ALVOT, an evolutionary approach. *International Conference on Intelligent Data Engineering and Automated Learning,* Springer, Vol. 2690, pp. 452–456. DOI: 10.1007/978-3-540-45080-1_61.
37. **López-Espinoza, E., Carrasco-Ochoa, J.A., & Martínez-Trinidad, J.F. (2004).** Two floating search strategies to compute the support sets system for ALVOT. *Iberoamerican Congress on Pattern Recognition,* Springer, Vol. 3287, pp. 677–684. DOI: 10.1007/978-3-540-30463-0_85.
38. **Ruiz-Shulcloper, J. & Lazo-Cortés, M. (1999).** Mathematical algorithms for the supervised classification based on fuzzy partial precedence. *Mathematical and Computer Modelling,* Vol. 29, No. 4, pp. 111–119. DOI: 10.1016/S0895-7177(99)00044-8.
39. **Medina-Pérez, M.A., García-Borroto, M., Villuendas-Rey, Y., & Ruiz-Shulcloper, J. (2006).** Selecting objects for ALVOT. *Iberoamerican Congress on Pattern Recognition,* Springer, Vol. 4225, pp. 606–613. DOI: 10.1007/11892755_63.
40. **Medina-Pérez, M.A., García-Borroto, M., & Ruiz-Shulcloper, J. (2007).** Object selection based on subclass error correcting for ALVOT. *Iberoamerican Congress on Pattern Recognition,* Springer, Vol. 4756, pp. 496–505. DOI: 10.1007/978-3-540-76725-1_52.
41. **Read, J., Pfahringer, B., & Holmes, G. (2016).** Meka: a multi-label/multi-target extension to weka. *The Journal of Machine Learning Research,* Vol. 17, No. 1, pp. 667–671.
42. **Cover, T. & Hart, P. (1967).** Nearest neighbor pattern classification. *IEEE transactions on information theory*, Vol. 13, No. 1, pp. 21–27. DOI: 10.1109/TIT.1967.1053964.
43. **John, G.H. & Langley, P. (1995).** Estimating continuous distributions in Bayesian classifiers. *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, Morgan Kaufmann Publishers Inc., pp. 338–345.

Article received on 20/12/2017; accepted on 23/01/2018. Corresponding author is Javier A. Hernández-Castaño.