Regional Classifiers: A Novel Framework for Pattern Classification

Rômulo B. P. Drumond, Renan F. Albuquerque, Guilherme A. Barreto

Graduate Program in Teleinformatics Engineering (PPGETI)
Federal University of Ceará (UFC), Center of Technology
CEP 60455-970, Campus of Pici, Fortaleza, Ceará, Brazil.
Emails: romulo.drumond@alu.ufc.br, renanfonteles@gmail.com, gbarreto@ufc.br

Abstract: The global classification paradigm uses the entire training set for building a single discriminating model for distinct classes. Alternatively, the local classification approach builds multiple discriminating models using smaller subsets of the training data. By considering these two paradigms as the extremes of a spectrum of possibilities, in this paper we introduce a novel two-stage framework for building pattern classification models based on the clustering of the self-organizing map (SOM) approach Vesanto and Alhoniemi (2001). According to this technique, data samples are submitted to the SOM as a preprocessing stage. Then, clustering algorithms (e.g. the K-means) are applied to the prototype vectors of the SOM aiming at organizing them in well-defined groups. By applying this two-stage strategy to labelled data, we show how to build accurate classifying models, henceforth referred to as regional classifiers, using the subset of samples mapped to a specific cluster of SOM prototypes. A comprehensive comparative study is carried out in order to evaluating the effectiveness of the proposed approach on several benchmarking data sets.

Keywords: Pattern classification; Global and local models; Self-organizing maps; Clustering of the SOM; Regional models.

1. INTRODUCTION

Pattern classification refers to the construction of models that build decision surfaces separating data from different classes. The classification model (or, simply, classifier) must provide a function that maps data sample from a continuous input space to a discrete output space coding the finite classes of the problem. Classification problems can be roughly categorized into linearly and nonlinearly separable ones. This categorization determines if a particular problem will need either a linear or a nonlinear decision surface to perform the data separation. Often, real-world applications produce nonlinear separable data and, therefore, require nonlinear classifiers to achieve high performance.

Artificial neural networks and kernel methods - e.g. the least squares SVM (Suykens and Vandewalle, 1999) - are examples of widely used paradigms for solving complex classification problems. These techniques are usually based on the global modeling approach, which consists in building a single model that represents the available data set of input-output observations. In contrast to this approach, local modeling requires the segmentation of the input space into several smaller partitions, where for each partition a specific local model is constructed. As such, local modeling is a viable alternative to the global method for modeling nonlinear data (Wang and Syrmos, 2007; Souza Júnior et al., 2015; Di Marzio et al., 2019).

Local models have already been presented in classic works in the field of artificial neural networks, such as local linear perceptrons (Alpaydin and Jordan, 1996) and adaptive mixtures of local experts (Jacobs et al., 1991). More recently, several works have proposed methods of local discrimination in the construction of standard classifiers (Bevilacqua and Marini, 2014; Song et al., 2017; Peng and Bhamt, 2001). In Bevilacqua and Marini (2014), a study on locally weighted partial least squares (LW-PLS) for nonlinear classification is performed. In this study, a classification approach is proposed by extending the local weighting approach of partial least squares (PLS) in a discriminant form (PLS-DA). Moreover, a similar work presented by Song et al. (2017) combines local PLS-DA with non-Euclidean distance measures (i.e. fractional distance) to classify data with different dimensionalities. In Gonen and Alpaydin (2010), a localized and supervised dimensionality reduction method is proposed. This method consists in partitioning the input space into regions which will be processed for dimensionality reduction separately in each region, from projection matrices using kernel functions. A performance comparison study and evaluation of several local models can also be found in Bischl et al. (2013).

The global and local paradigms can be seen as the extremes of a spectrum of possibilities for pattern classification. By doing so, in this paper we introduce a novel two-stage framework for building pattern classification models based on the clustering of the self-organizing map (SOM) approach (Vesanto and Alhoniemi, 2001). According to
In this technique, data samples are submitted firstly to the SOM as a preprocessing stage. Then, clustering algorithms (e.g., the K-means) are applied to the prototype vectors of the SOM aiming at organizing them in well-defined groups. By applying this two-stage strategy to labelled data, we show how to build accurate classifying models, henceforth referred to as regional classifiers, using the subset of samples mapped to a specific cluster of SOM prototypes. A comprehensive comparative study is carried out in order to evaluating the effectiveness of the proposed approach on several benchmarking data sets.

The remainder of the paper are organized as follows. In Section 2, the fundamentals of the ordinary least squares (OLS) method, the K-means and the SOM algorithms are briefly described. In Section 3, we introduce the basics of local classification and proposes a framework of regional modelling for pattern classification. In Section 4, we report the results of computer experiments, and present the classification performance metrics used in the evaluation the proposed approach. The obtained results are discussed in Section 5. The paper is concluded in Section 6.

2. METHODS

In this section we describe briefly the algorithms and estimation techniques over which the proposed approach will be built.

The Self-Organizing Map (SOM) is an unsupervised competitive neural network introduced by Kohonen (1997) which is usually applied to clustering and vector quantization (VQ) tasks. The main goal of the SOM network consists in learning a mapping from a continuous high-dimensional space to a discrete space. This mapping, or projection, is realized by \( N \) neurons (or prototypes) arranged in an \( S \)-dimensional space, typically represented as a two-dimensional grid. Formally, for a continuous space \( \mathcal{X} \subseteq \mathbb{R}^D \) and a discrete space \( \mathcal{Y} \subseteq \mathbb{R}^s \), composed by \( N_w \) prototypes, a vector \( \mathbf{x} \in \mathcal{X} \) will be represented by a prototype vector \( \mathbf{w}_i \in \mathcal{Y} \) by the mapping \( i^*(\mathbf{x}) : \mathcal{X} \rightarrow \mathcal{Y} \). For training a SOM network, firstly the \( N_w \) neurons’ weights are randomly initialized. After proper initialization, the algorithm proceeds with two essential stages (Haykin, 2009):

1. **Competition:** For an input pattern \( \mathbf{x}(n) \), the SOM network searches for the nearest prototype (a.k.a. the winning neuron/prototype) to \( \mathbf{x}(n) \), based on a similarity measure:

   \[
   i^*(n) = \arg \min_{i \in \mathcal{Y}} \| \mathbf{x}(n) - \mathbf{w}_i(n) \|^2, \tag{1}
   \]

   where \( \| \cdot \| \) denotes the Euclidean norm.

2. **Cooperation:** All the weight vectors of the network are updated based on the following learning rule:

   \[
   \mathbf{w}_i(n+1) = \mathbf{w}_i(n) + \alpha(n) h_{i,i^*(n)}(n) [\mathbf{x}(n) - \mathbf{w}_i(n)] \tag{2}
   \]

   where \( 0 < \alpha(n) < 1 \) denotes the learning rate at iteration \( n \) and \( h_{i,i^*(n)} \) is referred to as neighborhood function. Since this function defines a neighbourhood around the winning prototype, the prototypes that will be mostly adjusted are the winning prototype and its immediate neighbours. A common choice for the neighborhood function is the Gaussian function:

   \[
   h_{i,i^*(n)}(n) = \exp \left[ - \frac{\| \mathbf{r}_i - \mathbf{r}_{i^*}(n) \|^2}{2\sigma^2(n)} \right] \tag{3}
   \]

   where \( \mathbf{r}_i \) and \( \mathbf{r}_{i^*}(n) \) are, respectively, the coordinates of the \( i \)-th neuron and the winning neuron \( i^*(n) \) in the output grid. The parameter \( \sigma(n) > 0 \) denotes the radius or width of the neighborhood function. The larger the radius the higher the number of neurons updated around the winning neuron. To ensure convergence of the SOM network to stable values during the training process, it is necessary to decrease the values of the radius and the learning rate. Considering \( \alpha_0 \) and \( \sigma_0 \) their initial values, the neighborhood radius \( \sigma \) and the learning rate \( \alpha \) can be, for example, reduced exponentially over time as follows:

   \[
   \sigma(n) = \sigma_0 \exp \left( - \frac{n}{\tau_1} \right) \quad \text{and} \quad \alpha(n) = \alpha_0 \exp \left( - \frac{n}{\tau_2} \right) \tag{4}
   \]

   where \( \tau_1 \) and \( \tau_2 \) are user-defined decay parameters.

The **K-Means Algorithm** is a well-known clustering algorithm that separates the data set in partitions based on distances to the nearest centroid (i.e. the mean vector of a cluster). The algorithm can be summarized in five main steps:

1. Define \( K > 1 \) in advance. This is the number of clusters used to separate the data samples.
2. Initialize the position of the \( K \) prototypes. Prototype \( i \) is in position \( \mathbf{w}_i \).
3. Divide the data set into the existing \( K \) partitions. Each partition \( V_i \) is defined as follows:

   \[
   V_i = \{ \mathbf{x} \in \mathbb{R}^P : \| \mathbf{x} - \mathbf{w}_i \|^2 < \| \mathbf{x} - \mathbf{w}_j \|^2, \quad \forall j \neq i \} \tag{5}
   \]

4. Calculate the new position of \( \mathbf{w}_i \) by taking the average of the data points in partition \( V_i \):

   \[
   \mathbf{w}_i = \frac{1}{N_i} \sum_{\mathbf{x} \in V_i} \mathbf{x} \tag{6}
   \]

5. Repeat steps 3 and 4 until convergence is achieved.

A common choice to evaluate the convergence of the algorithm is via the sum of the squared distances (SSD), defined as

\[
SSD(K) = \sum_{i=1}^{K} \sum_{\mathbf{x} \in V_i} \| \mathbf{x} - \mathbf{w}_i \|^2 \tag{7}
\]

As the convergence of the K-means algorithm strongly depends on the initial position of the prototypes, it is usual to run the algorithm several times, for different initial positions, and choose as the best final positions of the prototypes the one producing the smallest SSD value. Further details about K-means clustering algorithm can be found in Kanungo et al. (2002).

The **OLS Method** is a method for estimating the parameters of a linear model. This can be accomplished by finding the optimum value of the parameter vector \( \mathbf{\theta} \) that minimizes the following cost function:

\[
J(\mathbf{\theta}) = \| \mathbf{y} - \mathbf{\theta}^T \mathbf{X} \|^2 \tag{8}
\]

where \( \mathbf{y} \) is a vector with all observed outputs and \( \mathbf{X} \) is a matrix whose columns are the corresponding input vectors \( \mathbf{x} \). This problem is convex and has an analytical solution:

\[
\mathbf{\theta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \tag{9}
\]

The OLS is commonly used in linear regression problems, but it can also be used for estimating the parameters of a
linear classifier (Barreto and Barros, 2015). Further details about OLS algorithm can be found in Fomby et al. (1984).

3. PROPOSED APPROACH

As mentioned in the introduction, regional classification (RC) is an alternative approach that can be positioned between the global and local classification paradigms. A single global classifier may not be able to model faithfully highly nonlinear decision boundaries between classes, while multiple local classifiers may not handle well classes with strong overlapping (it would be necessary too many local models in order to achieve a high accuracy!). The regional classification comes up as an alternative that can capture enough details of the decision boundaries without suffering from an excessive number of local models (high granularization) due to class overlapping.

In a sum, the RC paradigm aims at building a multiple model approach that can handle highly nonlinear separable problems with highly overlapping classes. The regional classifiers can be linear (e.g. simple perceptron) or nonlinear (e.g. LSSVM), a choice that will depend on the complexity of the classification task. The main principle underlying the RC approach is the concept of regionalization, which regards the process of finding suitable regions in the SOM network. In the design of a regional classifier, two steps are essential, namely, the definition of the regions for training the K regional classifiers; and the selection of a region for class label prediction.

1) Regions definition for models’ training: In general terms, it is the process of finding the regions of the SOM that represent the data properly. This step occurs within the classifier training process, where the regions are defined and the input patterns are clustered into each region. This process is called regionalization. A region is not the same as a single data cluster. In fact, a region covers several data clusters, whose associated data samples are concatenated and used to building the corresponding regional classifier.

(2) Region selection for class prediction: It is the process of searching and selecting a single region that best describes a particular input pattern \( x \). This step defines which regional classifier will be used for class label estimation for a new data point.

Bearing in mind these two main steps, the regional classifier will be presented in two stages: training and prediction. For this sake, consider a training set \( \mathcal{X} = \{ \mathbf{x}_i \}_{i=1}^N \), \( \mathbf{x}_i \in \mathbb{R}^d \), where \( N \) is the number of training samples, and \( d \) is the input vector dimensionality. The training set \( \mathcal{X} \) will be partitioned in a set of \( K \) non-overlapping regions \( \mathcal{R} = \{ R_1, R_2, \ldots, R_k \} \). Therefore, \( \forall R_i, R_j \in \mathcal{R}, R_i \cap R_j = \emptyset \). For each region \( R_k \) a respective model \( M_k \) will be trained.

Figure 1 illustrates the training process of a regional classifier. The regionalization process is performed by the SOM and K-means algorithm applied in this order. At the end of regionalization, \( K \) regions will be defined, and then \( K \) regional classifiers will be trained. For the experiments performed in this work, the OLS linear classifier was used. Therefore, in this case \( K \) OLS classifiers need to be trained using data samples mapped to each of the \( K \) regions.

Figure 1. Flowchart describing the training process using regional classification models

In summary, the process of finding the regions and then training the regional classifiers involves two steps:

- **Regionalization**: A set of regions are identified based on sequential application of the SOM to the data, followed by application of the \( K \)-means to the SOM prototypes. Firstly, apply the SOM to the observed samples without considering their labels. Then, apply the \( K \)-means to cluster the \( N_w \) prototypes of the SOM. A region is defined as a valid cluster of SOM prototypes. It should be noted that the number of regions will define the number of classifiers that will compose the final regional classifier.

- **Training the regional classifiers**: After identifying the regions, the subset of data samples mapped to the prototypes of the region \( R_i \) will be used to train the corresponding regional classifier \( M_i \).

Figure 2 illustrates the prediction process of a regional classifier. The first step of the prediction is to identify which regional model should be used to classify a particular input pattern \( x \). This choice will depend on the similarity that the input vector \( x \) has with the SOM prototypes comprising the regions. Once the region is selected, the specific classifier of this region is used, obtaining the predicted class label. In summary, the prediction of unlabeled data samples using a regional classifier can be described into two steps:

- **Region Search**: In this step, a similarity criterion is used to determine the corresponding region of the input data point \( x \). The most similar one, \( R_{i^*} \), is selected as follows:

  (1) Find the closest SOM prototype to the input vector:
  \[
  j^* = \arg\min_{v_j} \| \mathbf{x} - \mathbf{w}_j \|^2 \quad (10)
  \]

  (2) Find the closest K-means prototype from the winning neuron:
  \[
  i^* = \arg\min_{v_l} \| \mathbf{w}_{l^*} - \mathbf{p}_j \|^2 \quad (11)
  \]

(3) Select the classifier \( M_{i^*} \) that represents region \( R_{i^*} \). This classifier represents the model generated by training data points from \( R_{i^*} \). Thus, it will be used to predict the class label of \( x \).

- **Local Prediction**: After finding the region \( R_{i^*} \), the classification model \( M_{i^*} \) will be used to predict the class of the input vector \( x \).
4. METHODOLOGY

The data sets used in the experiments were selected from the UCI Machine Learning Repository (Dua and Graff, 2017) and they are described in Table 1. The 2nd column of the table indicates the number of classes for each data set and its variations. That is, the vertebral column data set has two variations, for 2 classes and 3 classes; while the data set wall-following provides three variations, with 2, 4 and 24 input attributes. For all data sets the input attributes were rescaled to the range [0, 1].

Table 1. Summary of data sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>Classes</th>
<th>Samples</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parkinson</td>
<td>2</td>
<td>195</td>
<td>22</td>
</tr>
<tr>
<td>Vertebral Column</td>
<td>[2, 3]</td>
<td>310</td>
<td>6</td>
</tr>
<tr>
<td>Wall-Following</td>
<td>5</td>
<td>5456</td>
<td>[2, 4, 24]</td>
</tr>
</tbody>
</table>

Experiments were designed to compare the overall performance between a global OLS classifier and a regional OLS classifier (ROLS). They were evaluated using confusion matrix based metrics: accuracy, specificity and sensitivity. The holdout cross-validation method was used to train and validate the classifiers. Each experiment involved 100 independent training and prediction runs, where 80% of the data was selected for training and 20% for testing. It should be noted that in each run of the holdout cross validation, a new set of regions is found, since the training data is different for each run.

The SOM hyperparameters \( a_0 \) and \( \sigma_0 \) were optimized using grid search for each data set, with SSD as the metric for evaluating convergence. The SOM was trained for 100 epochs, and the number of prototypes was defined as \( N_{SOM} = 5 \sqrt{N} \), where \( N \) is the number of samples of the data set. In the clustering step, carried out by the \( K \)-means, a search for a optimal \( k^* \) was performed. In this case, the Davies-Bouldin (DB) index (Davies and Bouldin, 1979) is used to define the optimal number of regions for a particular SOM representation. The search starts with \( K = 2 \) to a \( K_{max} = \lfloor \sqrt{N_{SOM}} \rfloor \).

In order to evaluate the classifiers, the following metrics based on confusion matrices were used:

\[
\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN},
\]

\[
\text{sensitivity} = \frac{TP}{TP + FN}, \quad \text{specificity} = \frac{TN}{TN+FP},
\]

\[
\text{precision} = \frac{TP}{TP + FP}, \quad \text{recall} = \frac{TP}{TP + FN},
\]

where \( TP, TN, FP \) and \( FN \) stand for True-Positive, True-Negative, False-Positive and False-Negative values, respectively. The metric \( F_1 \)-score was chosen for determining the best performing model: \( F_1 = 2 \times \text{precision} \times \text{recall} / (\text{precision} + \text{recall}) \).

5. RESULTS

In this section, the performance results of the proposed ROLS classifier is evaluated and compared to the standard global OLS classifier. In Tables 2, 3 and 4 we present the results for the Parkinson, Vertebral Column-2C and Vertebral Column-3C data sets, respectively. Regarding accuracy, the results show that the ROLS classifier has a reduced generalization capacity (i.e. test accuracy) compared to OLS in Parkinson, and practically equivalent performance considering both 2C and 3C variations of vertebral column. This shows that the regional classifier approach does not guarantee higher performance in terms of accuracy, especially for unbalanced data sets, such as Parkinson. Unbalanced data sets tend to cause a bias in the separation surface in a way the class with more samples will be prioritized in relation to the other classes.

Table 2. Results for the Parkinson data set.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>Sens.</th>
<th>Spec.</th>
<th>( F_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train OLS</td>
<td>90.96 ( \pm ) 1.25</td>
<td>70.71</td>
<td>97.54</td>
<td>79.28</td>
</tr>
<tr>
<td>Train ROLS</td>
<td>93.27 ( \pm ) 1.57</td>
<td>82.24</td>
<td>96.85</td>
<td>85.44</td>
</tr>
<tr>
<td>Test OLS</td>
<td>87.60 ( \pm ) 4.32</td>
<td>63.70</td>
<td>95.57</td>
<td>71.39</td>
</tr>
<tr>
<td>Test ROLS</td>
<td>85.77 ( \pm ) 5.21</td>
<td>70.40</td>
<td>90.90</td>
<td>70.97</td>
</tr>
</tbody>
</table>

Table 3. Results for the Vertebral Column-2C data set.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>Sens.</th>
<th>Spec.</th>
<th>( F_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train OLS</td>
<td>84.31 ( \pm ) 1.12</td>
<td>92.10</td>
<td>67.95</td>
<td>88.83</td>
</tr>
<tr>
<td>Train ROLS</td>
<td>86.62 ( \pm ) 1.55</td>
<td>92.01</td>
<td>75.29</td>
<td>90.31</td>
</tr>
<tr>
<td>Test OLS</td>
<td>83.71 ( \pm ) 4.30</td>
<td>91.64</td>
<td>67.05</td>
<td>88.39</td>
</tr>
<tr>
<td>Test ROLS</td>
<td>83.66 ( \pm ) 3.85</td>
<td>90.33</td>
<td>69.65</td>
<td>88.20</td>
</tr>
</tbody>
</table>

Table 4. Results for the Vertebral Column-3C data set.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>Sens.</th>
<th>Spec.</th>
<th>( F_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train OLS</td>
<td>82.48 ( \pm ) 1.62</td>
<td>77.35</td>
<td>90.99</td>
<td>78.21</td>
</tr>
<tr>
<td>Train ROLS</td>
<td>85.33 ( \pm ) 1.70</td>
<td>80.74</td>
<td>91.73</td>
<td>81.44</td>
</tr>
<tr>
<td>Test OLS</td>
<td>80.53 ( \pm ) 4.04</td>
<td>75.26</td>
<td>89.03</td>
<td>75.76</td>
</tr>
<tr>
<td>Test ROLS</td>
<td>82.97 ( \pm ) 3.73</td>
<td>77.82</td>
<td>90.38</td>
<td>78.32</td>
</tr>
</tbody>
</table>

One of the benefits of the ROLS classifier is that it is able to balance the values of sensitivity and specificity, such that the model reduces the bias in increasing the performance of the classes containing the majority of samples. Note that in both ROLS results in Tables 2, 3 and 4, sensitivity and specificity values tend to be more balanced than those achieved by the OLS classifier. Hence, evaluating OLS and ROLS results considering \( F_1 \)-score, it can
be concluded that the ROLS classifier achieved equivalent performance in Parkinson and Vertebral Column-2C, and higher performance with respect to Vertebral Column-3C.

In Tables 5, 6 and 7 we report the results for the Wall-Following data set, considering 2, 4 and 24 attributes in the input pattern. This data set consists of data with high overlapping between classes. For this scenario, global classifiers tend to achieve poor performance. The ROLS approach defines regions of higher granularity compared to a single region covering the entire training data. It can be verified in the results among all variations of the wall-following data set, where the ROLS classifier performs significantly better than the OLS.

Table 5. Results for the Wall-Following-2a data set.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>Sens.</th>
<th>Spec.</th>
<th>F₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>OLS 64.66 ± 0.26</td>
<td>50.91</td>
<td>78.10</td>
<td>46.95</td>
</tr>
<tr>
<td></td>
<td>ROLS 76.32 ± 4.36</td>
<td>76.85</td>
<td>85.93</td>
<td>76.61</td>
</tr>
<tr>
<td>Test</td>
<td>OLS 64.53 ± 1.00</td>
<td>50.85</td>
<td>78.02</td>
<td>46.88</td>
</tr>
<tr>
<td></td>
<td>ROLS 76.19 ± 4.49</td>
<td>76.72</td>
<td>85.86</td>
<td>76.49</td>
</tr>
</tbody>
</table>

Table 6. Results for the Wall-Following-4a data set.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>Sens.</th>
<th>Spec.</th>
<th>F₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>OLS 68.59 ± 0.22</td>
<td>53.44</td>
<td>78.72</td>
<td>48.90</td>
</tr>
<tr>
<td></td>
<td>ROLS 82.68 ± 4.05</td>
<td>80.23</td>
<td>89.49</td>
<td>81.32</td>
</tr>
<tr>
<td>Test</td>
<td>OLS 68.50 ± 1.00</td>
<td>53.42</td>
<td>78.67</td>
<td>48.86</td>
</tr>
<tr>
<td></td>
<td>ROLS 82.51 ± 4.03</td>
<td>79.97</td>
<td>89.39</td>
<td>81.09</td>
</tr>
</tbody>
</table>

Table 7. Results for the Wall-Following-24a data set.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>Sens.</th>
<th>Spec.</th>
<th>F₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>OLS 69.55 ± 0.36</td>
<td>58.84</td>
<td>80.13</td>
<td>59.49</td>
</tr>
<tr>
<td></td>
<td>ROLS 88.56 ± 1.12</td>
<td>87.04</td>
<td>93.47</td>
<td>87.29</td>
</tr>
<tr>
<td>Test</td>
<td>OLS 69.22 ± 1.17</td>
<td>58.62</td>
<td>79.93</td>
<td>59.26</td>
</tr>
<tr>
<td></td>
<td>ROLS 85.60 ± 1.23</td>
<td>84.06</td>
<td>91.70</td>
<td>84.22</td>
</tr>
</tbody>
</table>

Evaluating the model considering the distribution along the 100 runs of the holdout cross-validation experiment, it can be observed the stability of a classifier by analyzing the dispersion of the accuracy results in each run, for each classifier. Figure 3 illustrates the accuracy distributions considering all data sets. In general, regional classifiers tend to be more dispersed with respect to classification performance. This is due to the process of regionalization. In the proposed method, the SOM and the K-means algorithms are used to define the model’s regions. Both algorithms depend on hyperparameters which are crucial for algorithm convergence. Consequently, the regional classifier will be affected by the performance of regionalization.

Hence, if the training of the SOM or the K-means is inadequate, this will result in a performance degradation of the regional classifier. It should be noted that, even with the variability caused by the process of regionalization, the ROLS classifier still presents improved performance when compared the standard OLS model. For example, in Figure 3d, even the lower outliers presented by the ROLS boxplot are varying around the average values of accuracy of classic OLS distribution.

At this point, it is clear that the regionalization stage is essential to ensure high classification performance for the proposed approach. However, the DB cluster validation index used in this paper to determine the optimal number of regions is highly sensitive to initialization. Figure 4 shows the histogram of the number of regions identified per training/testing run of the holdout cross validation process. One can observe that the number of regions varies considerably in different runs as the SOM and K-Means algorithms process distinct training sets. Therefore, this variability in the number of regions per training/testing run depends directly on the organization of the data in the input space and, hence, of the SOM prototypes. Thus, other more stable strategies for cluster validation must be evaluated, such as the silhouette index.

6. CONCLUDING REMARKS

In this paper, a regional classification framework was presented as a new approach for the classification of non-linearly separable data. Using the proposed framework, we performed a comprehensive evaluation of the performance of the proposed approach on three benchmarking data sets. The results obtained showed that the proposed...
regional classification approach presents itself as a promising alternative to standard local and global classification paradigms. Currently, the authors are working on methods for reducing the variation in the regionalization process, since it plays an essential role in the classification performance of the regional model.

**Acknowledgements** - This study was financed by the following Brazilian research funding agencies: CAPES (Finance Code 001), FUNCAP (88887.177150/2018-00) and CNPq (309451/2015-9).

**REFERENCES**


