UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA DE TELEINFORMÁTICA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE TELEINFORMÁTICA

RÔMULO BANDEIRA PIMENTEL DRUMOND

REGIONAL MODELS FOR PROTOTYPE-BASED CLASSIFICATION: A NOVEL PARADIGM

FORTALEZA

2020

RÔMULO BANDEIRA PIMENTEL DRUMOND

REGIONAL MODELS FOR PROTOTYPE-BASED CLASSIFICATION: A NOVEL PARADIGM

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia de Teleinformática do Centro de Tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Engenharia Engenharia de Teleinformática. Área de Concentração: Sinais e Sistemas

Orientador: Prof. Dr. Guilherme de Alencar Barreto

FORTALEZA

2020

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

D858r Drumond, Rômulo Bandeira Pimentel.
Regional Models for Prototype-based Classification: A Novel Paradigm / Rômulo Bandeira Pimentel Drumond. – 2020.
72 f. : il. color.

Dissertação (mestrado) – Universidade Federal do Ceará, Centro de Tecnologia, Programa de Pós-Graduação em Engenharia de Teleinformática, Fortaleza, 2020.
Orientação: Prof. Dr. Guilherme de Alencar Barreto.

1. Reconhecimento de padrões. 2. Modelos globais e locais. 3. Mapas auto-organizáveis. 4. Clusterização do SOM. 5. Modelos locais. I. Título.

CDD 621.38

RÔMULO BANDEIRA PIMENTEL DRUMOND

REGIONAL MODELS FOR PROTOTYPE-BASED CLASSIFICATION: A NOVEL PARADIGM

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia de Teleinformática do Centro de Tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Engenharia Engenharia de Teleinformática. Área de Concentração: Sinais e Sistemas

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. Guilherme de Alencar Barreto (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Victor Hugo Albuquerque Costa
Universidade de Fortaleza (UNIFOR)

Prof. Dr. Amauri Holanda de Souza Júnior
Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE)

Prof. Dr. Aluizio Fausto Ribeiro Araújo
Universidade Federal de Pernambuco (UFPE)

À mim, por ter tido resiliência durante esses últimos anos e lembrado que a peça mais importante da minha vida sou eu.

ACKNOWLEDGEMENTS

I say my thanks to my family, not only for their support but also because they gave me the foundation of my character and work ethic. I also want to say thanks to Marcia Galeno for being a supportive and kind girlfriend.

I would further say thanks to my supervisor, Prof. Dr. Guilherme de Alencar Barreto for his support and guidance throughout the research process. I also give my thanks to all members of "Centaurinho", David, Haroldo, Júlio Peixoto, Michael Duarte, Nonato, Polycarpo, Allan Kelvin, Jackson Uchoa and Renan Bessa. My special thanks go to Diego Perdigão and Renan Fonteles for being partners on my research, resulting in them being co-authors of the publications made during my masters.

Finally, I would like to acknowledge the financial support of FUNCAP (process 88887.177150/2018-00), of Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) (financial code 001), CNPq (PROEX) and CNPq (process 309451/2015-9).

“ But swear if you call me
I won’t resist you
I want to say no but I wanna see you
And if call me,
You know where I’m heading
Out the door, in your bed is where it’s ending ”

(Barons of the Little Step)

RESUMO

O paradigma de classificação global utiliza todo o conjunto de treinamento para produzir um único modelo discriminante para as diversas classes. Alternativamente, a abordagem de classificação local baseada em *clusters* constrói múltiplos modelos discriminantes usando subconjuntos dos dados de treinamento. Ao considerar esses dois paradigmas como extremos de um espectro de possibilidades, nesta dissertação, é introduzido um novo paradigma de dois estágios para a construção de modelos de classificação de padrões baseados no método da *clusterização* dos mapas auto-organizáveis (SOM, *self-organizing maps*) (VESANTO *et al.*, 2000). De acordo com essa técnica, amostras são submetidas ao SOM em um estágio de pré-processamento. Posteriormente, algoritmos de *clusterização* (e.g. K-médias) são aplicados nos vetores protótipos do SOM com o objetivo de organizá-los em regiões bem definidas. Ao aplicar essa estratégia de dois estágios em dados rotulados, é mostrado como construir modelos de classificação precisos, doravante referidos como *classificadores regionais*, usando um subconjunto de amostras mapeados a um *cluster* específico de unidades do SOM. Um abrangente estudo comparativo é realizado para avaliar a eficácia da abordagem proposta em diversos bancos de dados de *benchmarking*, usando modelos lineares, i.e. classificador de mínimos quadrados com função de base linear (LSC-LBF, *least squares classifier with linear basis functions*), e não lineares, i.e. máquinas de vetores-suporte de mínimos quadrados (LSSVM, *least squares support vector machine*) com *kernels* não lineares. Como passo adicional no treinamento de modelos locais baseados em *cluster* e regionais, durante a fase de validação do modelo, um conjunto de doze métricas de validação de *clusters* foi empregado para avaliar suas competências em prever a melhor quantidade de protótipos dado uma função objetivo bem definida. A capacidade das abordagens local e regional de construir funções de decisão não lineares com um conjunto de classificadores lineares é avaliada e o paradigma regional se apresentou como uma alternativa mais esparsa do que a abordagem local, tendo desempenho semelhante enquanto utilizando menos protótipos/modelos.

Palavras-chave: Reconhecimento de padrões. Modelos globais e locais. Mapas auto-organizáveis. Clusterização do SOM. Modelos locais. Modelos regionais. Máquinas de vetores-suporte de mínimos quadrados. K-médias.

ABSTRACT

The global classification paradigm uses the entire training set for producing a single discriminating model for distinct classes. Alternatively, the cluster-based 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 thesis, it is introduced a novel two-stage framework for building pattern classification models based on the clustering of the self-organizing map (SOM) method (VESANTO *et al.*, 2000). 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 regions. By applying this two-stage strategy to labeled data, it is shown how to build accurate classifying models, henceforth referred to as *regional classifiers*, using the subset of samples mapped to a specific cluster of SOM units. A comprehensive comparative study is carried out to evaluate the effectiveness of the proposed approach on several benchmarking data sets, using linear models, i.e. least squares classifier with linear basis functions (LSC-LBF), and nonlinear ones, i.e. least squares support vector machine (LSSVM) with nonlinear kernel functions. As an additional step on the training of cluster-based local and regional models, during the model validation phase, a set of twelve cluster validation metrics was used to assess their ability to predict the best number of prototypes given a well-defined objective function. The capability of the local and regional approaches to building nonlinear decision functions with a set of linear classifiers is assessed and the regional paradigm presented itself as a sparser alternative than the local approach, having similar performance while using fewer prototypes/models.

Keywords: Pattern recognition. Global and local models. Self-organizing maps. Clustering of the SOM. Local Models. Regional models. Least Squares Support Vector Machine. K-means.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

1-NN	1-Nearest Neighbor classifier
AMI	Adjusted Mutual Information
ARI	Adjusted Rand Index
CH	Calinski-Harabasz
CLHP	Cluster-based Local Modeling with Hard Partitioning
DB	Davies-Bouldin
DL	Deep Learning
DU	Dunn
FM	Fowlkes-Mallows
G-LSSVM	Global LSSVM
KKT	Karush-Khun-Tucker
L-LSSVM	CLHP version of the LSSVM
LSC-LBF	Least squares classifier with linear basis functions
LSSVM	Least squares support vector machine
mAIC	Modified Akaike Information Criteria
mBIC	Modified Bayesian Information Criteria
mFPE	Modified Final Prediction Error
MI	Mutual Information
ML	Machine Learning
mMDL	Modified Minimum Description Length
MSQE	Mean Squared Quantization Error
R-LSSVM	Regional LSSVM
RC	Regional Classifiers
RI	Rand index
SI	Silhouette
SIC	Schwarz Information Criterion
SOM	Self-Organizing Map
SSD	Sum of the Squared Distances
SVM	Support Vector Machine
VM	V-measure
VQ	Vector Quantization

LIST OF SYMBOLS

$\mathbf{W}$	Parameters of a model
$f_o(\cdot)$	Objective function
$g(\cdot)$	Classifier discriminative function
$\mathbf{x}$	Input vector of a model
y	A class label
N	Number of samples in a data set
N_{tr}	Number of samples in the training data set
b	A model bias
ε	A white Gaussian noise
$\mathcal{X}$	A data set
$\mathbb{R}^d$	A d dimensional real vector space
$h(\cdot)$	A classifier input space mapping function
$\|\|\cdot\|\|_2$	Euclidean norm
K	Number of prototypes in a local or regional modeling
$\psi(\cdot,\cdot)$	A function that measures dissimilarity between two vectors
$\mathbf{p}$	A prototype weight vector
$\emptyset$	Empty set
$\alpha(t)$	SOM's learning rate at iteration t
$h_{i,i^*(t)}$	SOM's neighborhood function at iteration t
$\mathbf{r}_i$	Coordinate of the SOM's i-th neuron in the output grid neighborhood function at iteration t
$\mathbf{r}_{i^*(t)}$	Coordinates of the SOM's winning neuron $i^*(t)$ in the output grid neighborhood function at iteration t
$\sigma(t)$	Radius or width of the Gaussian neighborhood function at iteration t
α_0	The initial learning rate of the SOM
σ_0	Initial radius of the Gaussian neighborhood function of the SOM

τ_1	User-defined decay parameter of SOM's Gaussian neighborhood function radius
τ_2	User-defined decay parameter of SOM's learning rate
$\mathbf{X}^{\dagger}$	The pseudo-inverse of the matrix $\mathbf{X}$
C	The variable that controls the regularization of an SVM classifier
$\boldsymbol{\xi}$	Vector of the flexible margins in an SVM/LSSVM optimization problem
$\phi(\cdot)$	High feature space mapping used in an SVM/LSSVM classifier
$K(\cdot,\cdot)$	A kernel function
γ	The variable that controls the regularization of an LSSVM classifier
$L(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha})$	The Lagrangian function
$\boldsymbol{\Omega}$	The kernel matrix
$\mathbf{I}$	Identity matrix
$\mathbf{1}$	The vector which all dimensions are equal to '1'
$\mathbf{u}$	A SOM's unit weight vector
$\mathcal{R}$	A data region
K_{opt}	The optimum number of prototypes K given a criteria
$\ln(\cdot)$	Natural logarithm

CONTENTS

1 INTRODUCTION

The research area of Machine Learning (ML) has gained a lot of attention in recent years given the success of Deep Learning (DL) (LECUN *et al.*, 2015), or the use of deep neural networks (DNN), in beating state-of-the-art models in some tasks (VOULODIMOS *et al.*, 2018) and in industry adoption (DUTTA, 2018). Although the fancy name, the 'learning' in machine learning is often a mathematical optimization of the kind

$$\begin{aligned} &\text{minimize} \quad f_o(\mathbf{W}) \\ &\text{subject to} \quad f_i(\mathbf{W}) \leq 0, \;\; i = 1, \ldots, m \end{aligned} \tag{1.1}$$

where $\mathbf{W}$ is the optimization variable, in ML the model parameters, $\{f_i\}_{i=1}^{m}$ the set of constraint functions and f_o is the objective function, usually referred to as cost function when the problem is of minimization and utility function when it is of maximization. It should be noted that any maximization problem can be rewritten as a minimization problem, and vice versa, either by changing the sign of the objective function or by using the reciprocal function..

The objective of the optimization is to find an $\mathbf{W}^*$, such as $f_i(\mathbf{W}^*) \leq 0, \forall i \in \{j\}_{j=1}^{m}$, and $f_o(\mathbf{W}) \geq f_o(\mathbf{W}^*), \forall\, \mathbf{W}$ which satisfies the set of constraint functions. When the optimization is done, or the 'machine has learned', the final product is a function $g(\mathbf{x}, \mathbf{W}^*)$ known as regressor function in regression problems or discriminative function in classification problems. As we focus on classification in this thesis, we will not discuss in details regression problems.

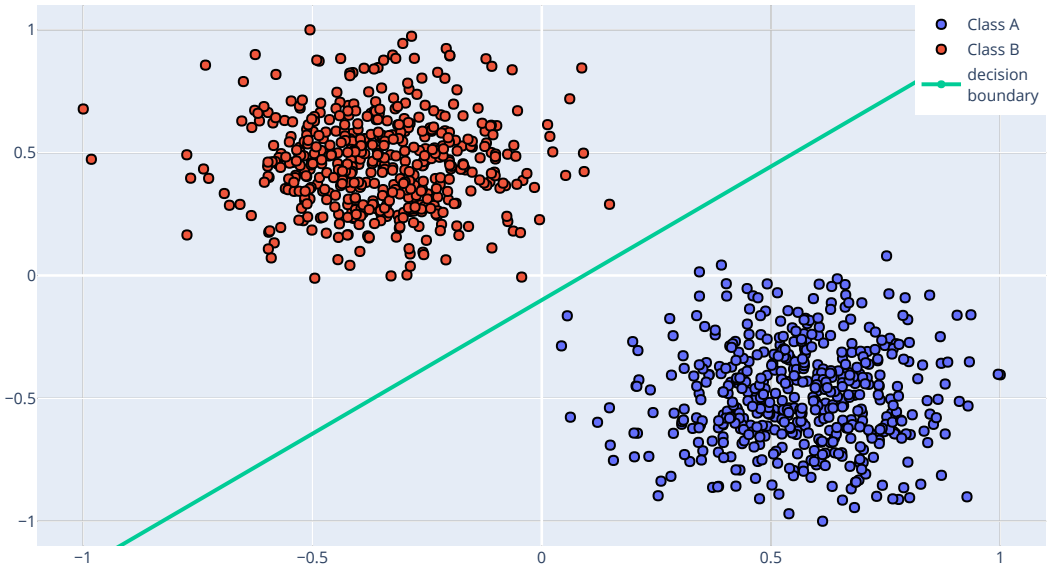
In classification problems we have access to a data set $\mathcal{X} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$; where $\mathbf{x}_i \in \mathbb{R}^d$ are the input vectors or attributes with dimensionality d, $y_i \in \{L_j\}_{j=1}^{c}$ are the labels or classes in a problem with c distinct classes, and N is the number of samples. In this type of problems we want to find $g(\mathbf{x}, \mathbf{W})$ that is able to predict the class $\hat{y}_{new}$ of a new sample $\mathbf{x}_{new}$ that was not present in the training data set $\mathcal{X}$. To do so an optimization problem of the type found in Eq. (1.1) is usually formulated using a predefined discriminative function form, for example an affine function

$$g(\mathbf{x}, \mathbf{w}) = [\mathbf{x}^T \;\; 1]\mathbf{w}, \tag{1.2}$$

where we concatenate $\mathbf{x}$ with '1' to encapsulate the model bias in the $\mathbf{w}$ variable, which is a vectorial representation of the model parameters $\mathbf{W}$.

After the mathematical optimization, we can find a function similar to the one shown in Figure 1 for a binary classification problem.

Figure 1 – Hypothetical example of a binary classification problem with a possible decision boundary.



Source: the author (2019).

In Figure 1, our function $g(\mathbf{x}, \mathbf{W})$ creates a decision boundary that separates two classes, reaching the objective to find a suitable discriminative function for our training data set.

The process of building a classifier described above is known as global modeling (WANG; SYRMOS, 2007; BISCHL *et al.*, 2013), where all the training data set is used to build a single discriminative function $g(\mathbf{x}, \mathbf{W})$. Another approach, that will be the focus of this thesis, is local modeling, which uses a set of models, or discriminative functions, $\{g_i(\mathbf{x}, \mathbf{W}_i)\}_{i=1}^{K}$ to form a global discriminative function. Drawing inspiration from the mathematical formulations of Alpaydin & Jordan (1996) the global function can be represented by the form

$$g(\mathbf{x}) = \sum_{i=1}^{K} h_i(\mathbf{x}, \mathbf{U}_i) g_i(\mathbf{x}, \mathbf{W}_i), \tag{1.3}$$

where function $h_i(\mathbf{x}, \mathbf{U}_i)$ denotes the weight of i-th classifier, or $g_i(\mathbf{x}, \mathbf{W}_i)$, with respect to the class prediction of $\mathbf{x}$. Analogous to $\mathbf{W}_i$, the $\mathbf{U}_i$ variable controls how the h_i function behaves.

Given that h_i depends on the input vector $\mathbf{x}$ it adds a notion of locality, depending on the position of the sample in the input space we have different importance for each classifier g_i. To further simplify notation we can rewrite it in a vectorial form while suppressing extra parameters other than the input vector $\mathbf{x}$

$$\mathbf{g}(\mathbf{x}) = [g_1(\mathbf{x}), g_2(\mathbf{x}), \cdots, g_K(\mathbf{x})]^T, \tag{1.4}$$

$$\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), h_2(\mathbf{x}), \cdots, h_K(\mathbf{x})]^T, \tag{1.5}$$

so we have

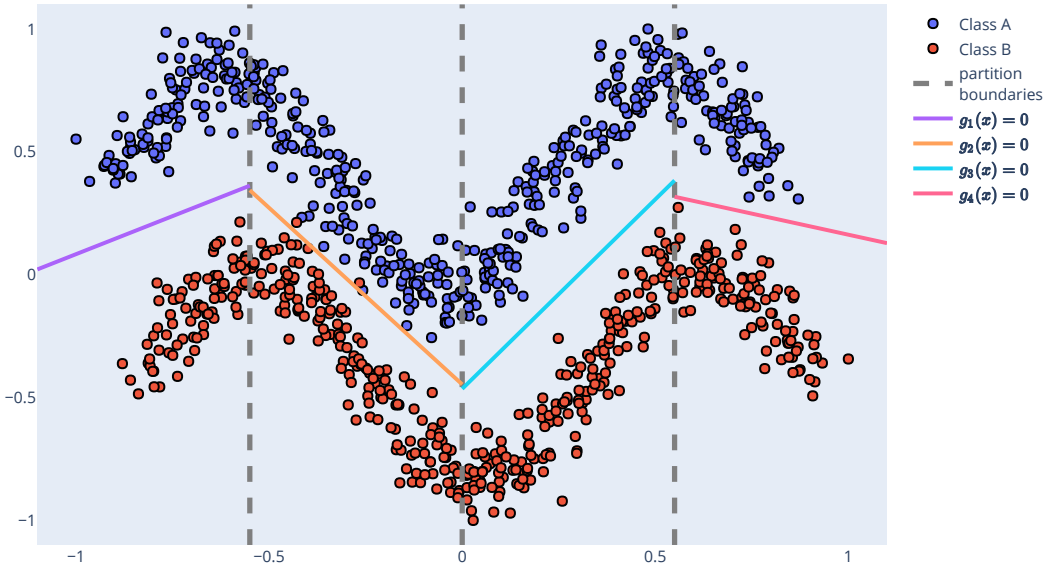
$$g(\mathbf{x}) = \mathbf{h}(\mathbf{x})^T \mathbf{g}(\mathbf{x}). \tag{1.6}$$

One can think of the $\mathbf{h}(\mathbf{x})$ as another classifier that for a given input it assigns a weight to the models as proposed by Alpaydin & Jordan (1996).

In some cases, we may know a priori how to divide the data, for example, in building a model for each client in an internet service or each product in a supermarket. In other cases, as it is not easy to identify groups, we may use local modeling, which, in the process, divides the task of modeling into subtasks by partitioning the input space between models. The partitioning can be hard, each subset is only visible for a specific model ($\mathbf{h}(\mathbf{x})$ has only a dimension equals to 1 and the others 0), or soft, $\mathbf{h}(\mathbf{x})$ may have continuous values in each dimension, them soft partitioning the data set between models $\{g_i(\cdot)\}_{i=1}^{K}$.

In Figure 2 we can see a local modeling case with a hard partitioning function $\mathbf{h}(\mathbf{x})$. In this case, we had made explicit the fact that the global decision function may not be either smooth or continuous in cases of hard partitioned input space.

Figure 2 – A global discriminative function formed by a set of local models with hard input space partitioning.



Source: the author (2019).

In local modeling, we have two known approaches in learning the classifier mapping function $\mathbf{h}(\cdot)$ and the set of discriminating functions $\mathbf{g}(\cdot)$:

- Coupled: during training, both $\mathbf{h}(\cdot)$ and $\mathbf{g}(\cdot)$ are optimized, or learned, at the same time (ALPAYDIN; JORDAN, 1996; JACOBS *et al.*, 1991);

- Uncoupled: the mapping $\mathbf{h}(\cdot)$ and models $\mathbf{g}(\cdot)$ are learned in a decoupled manner (BOTTOU; VAPNIK, 1992; MARTINETZ *et al.*, 1993);

One simple type of uncoupled local modeling approach is using a clustering algorithm to divide the input space, consequently the data set, and build a model for each cluster. During inference the classifier mapping function becomes

$$h_i(\mathbf{x}) = \begin{cases} 1, & \text{if } \|\mathbf{c}_i - \mathbf{x}\|_2^2 = \min\limits_{1 \leq j \leq K} \|\mathbf{c}_j - \mathbf{x}\|_2^2 \\ 0, & \text{otherwise} \end{cases} \tag{1.7}$$

where $\|\cdot\|_2$ denotes the Euclidian norm and $\mathbf{c}_i$ the cluster center of the i-th cluster.

A distinct modeling paradigm, regional modeling, was first proposed by Souza Jr. *et al.* (2015) and it was based on the idea of the clustering of the self-organizing map (VESANTO *et al.*, 2000). In the paper, the regional approach was used in a regression problem for dynamic system identification while in this thesis we will extend this approach for classification problems. By considering global and local modeling as the extremes of a spectrum of possibilities, the regional models sit between the two, where the two-stage strategy organizes the data in well-defined regions.

1.1 Motivation

The primary motivation of this work is to extend the regional modeling paradigm for pattern classification and compare it with other global and local modeling approaches. Due to the specificities that arise from classification tasks, when compared with regression, the regional approach demanded some adaptation.

Local modeling is still an important topic in the ML community as we see some combinations of it emerging with deep learning (ZHANG *et al.*, 2018; MANSANET *et al.*, 2016). We can also find industry adoption of techniques that rely on local modeling giving privacy concerns as in Konečnỳ *et al.* (2016), where a global model is estimated using local models that only have access to local data.

1.2 Objectives

The general objective of this thesis is to evaluate the viability and investigate the properties of the regional modeling paradigm as an alternative to the global and local paradigms in the design of pattern classifiers. The specific objectives can be enumerated as follows.

1. Develop global classifiers based on kernel machines, e.g. Least squares support vector machine (LSSVM), and evaluate them on benchmarking classification tasks;
2. Implement local classifiers based on prototypes, e.g. cluster-based local modeling, in benchmark problems of item 1;
3. Implement regional classifiers in the same benchmark problems of items 1 and 2, comparing the results with the ones of local and global classifiers.

1.3 Scientific production

During the development of this work the following articles were published:

- R. B. P. DRUMOND, R. F. ALBUQUERQUE, G. A. BARRETO, (2019). **Regional Classifiers: A Novel Framework for Pattern Classification**. In: ANAIS DO 14º SIMPÓSIO BRASILEIRO DE AUTOMAÇÃO INTELIGENTE, 2019, Ouro Preto, Anais eletrônicos Campinas, GALOÁ, 2020. Disponível em: <https://proceedings.science/sbai-2019/papers/regional-classifiers–a-novel-framework-for-pattern-classification> Acesso em: 04 fev. 2020. DOI: 10.17648/sbai-2019-111444.
- R. B. P. DRUMOND, R. F. ALBUQUERQUE, D. P. SOUSA, G. A. BARRETO, (2019). **Classificação Local utilizando Least Squares Support Vector Machine (LSSVM)**. Brazilian Computational Intelligence Meeting (CBIC'2019), Belém, Pará, p. 1-8.

1.4 Organization of the thesis

The rest of this document is organized as follows. **Chapter 2** presents the local classification paradigm in detail. Additionally, we describe briefly the Vector Quantization (VQ) algorithms used as building blocks of the proposed local and regional classifiers. We also present global classification models whose performances will serve as baseline of comparison for proposed approaches. **Chapter 3** introduces the regional modeling paradigm for pattern classifiers and discuss its main features. **Chapter 4** describes the simulation methodology used such as the choice of data sets, training and testing procedures, hyperparameter optimization, and cluster validation metrics. **Chapter 5** comprises a comprehensive evaluation comparison of global, local and regional paradigms for classification. There the reader can find graphs and

tables showing diverse performance metrics of the simulation results of the three paradigms. **Chapter 6** closes this thesis with conclusions that can be drawn from this work and future works proposals.

2 FUNDAMENTALS OF LOCAL CLASSIFICATION

In this chapter, we will discuss local modeling for classification and review some algorithms and models used in this thesis. As shown in Chapter 1, local modeling is a multimodel approach which creates a global discriminative function of the form

$$g(\mathbf{x}) = \sum_{i=1}^{K} h_i(\mathbf{x}) g_i(\mathbf{x}), \tag{2.1}$$

where $h_i(\mathbf{x})$ gives a notion of locality for the i-th model, represented by the function $g_i(\mathbf{x})$.

The most studied type of local modeling used in this thesis is the Cluster-based Local Modeling with Hard Partitioning (CLHP). For a better understanding of this approach, its fundamentals are discussed in the following section.

2.1 Cluster-based local modeling with hard partitioning (CLHP)

To explain this paradigm we had split the two main steps of the modeling process: training and inference.

2.1.1 Training in CLHP

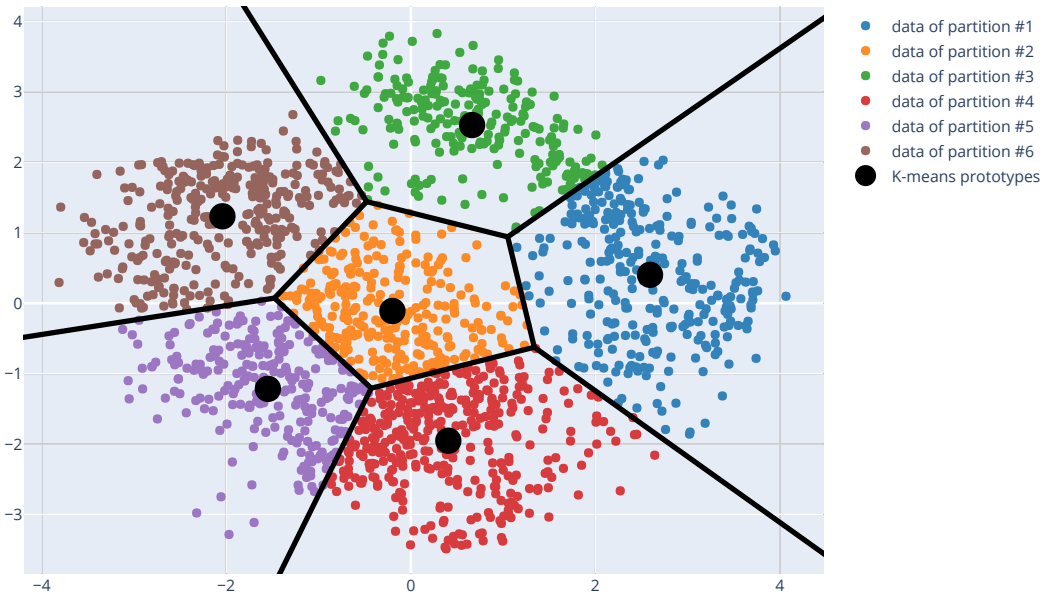
During the training phase, the whole data set $\mathcal{X}$ is divided into K partitions by some clustering or vector quantization algorithm (e.g. K-means). After that, for each data partition $V_i \subset \mathcal{X}$ we build a classification model (e.g. LSSVM), so that each model in the set $\{g_i\}_{i=1}^{K}$ has only seen the data of its associated partition. We can say that CLHP does a hard partitioning of the data set.

As illustrated in Figure 3, after the convergence of K-means algorithm, we had 6 prototypes defining 6 data partitions. For each data partition, highlighted with a unique color, we train a model.

2.1.2 Inference in CLHP

For each new sample $\mathbf{x}$ we estimate its partition by searching for the closest prototype using some dissimilarity measure (e.g. Euclidean distance to the K-means prototypes). As $\mathbf{x}$

Figure 3 – A hypothetical example of data partitioning into $K = 6$ non-overlapping partitions using the K-means algorithm.



Source: the author (2019).

belongs to partition V_{i^*} the most suitable model to make the prediction is the one portrayed by the function g_{i^*}. In this case, the classifier mapping function has the form

$$h_i(\mathbf{x}) = \begin{cases} 1, & \text{if } \psi(\mathbf{x}, \mathbf{p}_i) = \min\limits_{1 \leq j \leq K} \psi(\mathbf{x}, \mathbf{p}_j) \\ 0, & \text{otherwise,} \end{cases} \tag{2.2}$$

as $\psi(\cdot)$ is a function that measures dissimilarity between the sample $\mathbf{x}$ and the prototype $\mathbf{p}_i$. In the Eq. (2.2) the i index is the same index of h_i, g_i, and prototype $\mathbf{p}_i$.

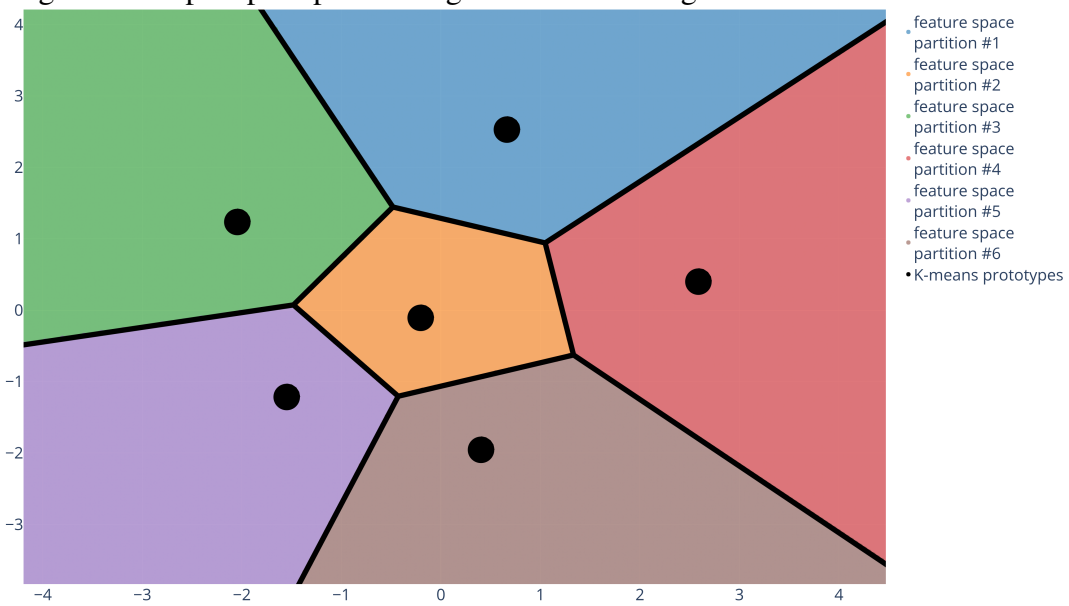
In Figure 4 becomes clear, using the same data of Figure 3, why the CLHP approach leads to a hard partitioning of the input space. In each input space partition, highlighted with a unique color, a model is responsible for making the predictions.

2.1.3 Major Drawbacks of the CLHP Approach

The application of CLHP in classification may give rise to two important phenomena: empty and homogenous partitions.

Empty partitions arise from a prototype $\mathbf{p}_i$ that has not any samples in its data partition, i.e. $V_i \subset \emptyset$. Although it may be a sign of bad clustering one may deal with this problem

Figure 4 – Input space partitioning with K-means algorithm.



Source: the author (2019).

by redefining Eq. (2.2) as follows

$$h_i(\mathbf{x}) = \begin{cases} 1, & \text{if } \psi(\mathbf{x}, \mathbf{p}_i) = \min\limits_{1 \leq j \leq K \mid V_j \not\subset \emptyset} \psi(\mathbf{x}, \mathbf{p}_j) \\ 0, & \text{otherwise,} \end{cases} \tag{2.3}$$

and not build the classifier $g_i(\mathbf{x})$ since there are no data samples to do so.

Even when the number of samples per partition is considered adequate, it should be noted that vector quantization is an unsupervised task. Thus, it is common to have samples with different labels within the same partition. In this case, we refer to this type of partition as an *heterogeneous partition*. On the other side, *homogenous partitions* happen when a prototype $\mathbf{p}_i$ maps only data points of a single class. One logical approach to deal with this phenomenon is to assume that any new data that arrive in the input space partition of $\mathbf{p}_i$ should have predicted label L_i. One may say that the g_i model is a "bias model"as its output is a constant (e.g. $+1$ or -1 in a binary classification problem with label encoding).

To further enhance our understanding of the CLHP approach in the following section we will review some vector quantization algorithms utilized in this thesis.

2.2 Vector Quantization algorithms

Essentially, Vector Quantization (VQ) algorithms are lossy data compression algorithms, since they have the goal of finding a simplified representation of the data set by using

prototypes. In this work, we use its capacity of finding partitions of interest in the data and input space to properly separate them for the task of local modeling.

In the next subsections, one may find the VQ algorithms used in this work.

2.2.1 The K-means algorithm

K-means is a well-known clustering algorithm that separates the data set into partitions based on the 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 prototypes/clusters/partitions used to separate the data samples;
2. Initialize the position of the K prototypes. The i-th prototype is described by the vector $\mathbf{p}_i$;
3. Split the data set into K partitions. Each partition V_i is defined as follows

$$V_i = \left\{ \mathbf{x} \in \mathbb{R}^d \;\middle|\; \|\mathbf{x} - \mathbf{p}_i\|_2^2 < \|\mathbf{x} - \mathbf{p}_j\|_2^2, \quad \forall j \neq i \right\}, \tag{2.4}$$

   where $\|\cdot\|_2$ represents the Euclidean norm;
4. Evaluate $\mathbf{p}_i(t+1)$ by the arithmetic mean of the $N_i(t)$ vectors of partition $V_i(t)$

$$\mathbf{p}_i(t+1) = \frac{1}{N_i(t)} \sum_{\forall \mathbf{x} \in V_i(t)} \mathbf{x}\,; \tag{2.5}$$

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

$$SSD = \sum_{i=1}^{K} \sum_{\forall \mathbf{x} \in V_i} \|\mathbf{x} - \mathbf{p}_i\|_2^2. \tag{2.6}$$

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

Examples of the result of K-kmeans algorithm can be seen in Figures 3 and 4.

2.2.2 The Self-Organizing Map

The Self-Organizing Map (SOM) is an unsupervised competitive neural network introduced by Kohonen (1982) which is usually applied to data visualization, clustering, and

vector quantization tasks. The main goal of the SOM consists of learning a mapping from a continuous high-dimensional space to a discrete space. This mapping, or projection, is realized by N_p neurons (or prototypes) arranged in an S-dimensional space, typically represented as a two-dimensional grid. Formally, for a continuous space $\mathcal{X} \subset \mathbb{R}^d$ and a discrete space $\mathcal{Y} \subset \mathbb{R}^S$, composed by N_p prototypes, a vector $\mathbf{x} \in \mathcal{X}$ will be represented by a prototype vector $\mathbf{p}_{i^*} \in \mathcal{Y}$ by the mapping $i^*(\mathbf{x}) : \mathcal{X} \to \mathcal{Y}$. For the training of a SOM network, firstly the N_p prototypes are randomly initialized. After proper initialization, the algorithm proceeds with two essential stages (HAYKIN, 2009):

1. **Competition**: For an input pattern $\mathbf{x}(t)$, the SOM network searches for the nearest prototype (a.k.a. the winning neuron/prototype) based on a dissimilarity measure, for example, the Euclidean distance:

$$i^*(t) = \arg \min_{1 \leq i \leq N_p} \|\mathbf{x}(t) - \mathbf{p}_i(t)\|_2^2 \,; \tag{2.7}$$

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

$$\mathbf{p}_i(t+1) = \mathbf{p}_i(t) + \alpha(t) h_{i,i^*(t)}(t)[\mathbf{x}(t) - \mathbf{p}_i(t)], \tag{2.8}$$

where $0 < \alpha(t) < 1$ denotes the learning rate at iteration t and $h_{i,i^*(t)}$ is referred to as the 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:

$$h_{i,i^*(t)}(t) = \exp\left[-\frac{\|\mathbf{r}_i - \mathbf{r}_{i^*(t)}\|_2^2}{2\sigma^2(t)}\right], \tag{2.9}$$

where $\mathbf{r}_i$ and $\mathbf{r}_{i^*(t)}$ are, respectively, the coordinates of the i-th neuron and the winning neuron $i^*(t)$ in the output grid. The parameter $\sigma(t) > 0$ 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 ordered states during the training process, it is necessary to decrease the values of the radius and the learning rate. Considering σ_0 and α_0 their initial values, the neighborhood radius σ and the learning rate α can be, for

example, reduced exponentially over time as follows:

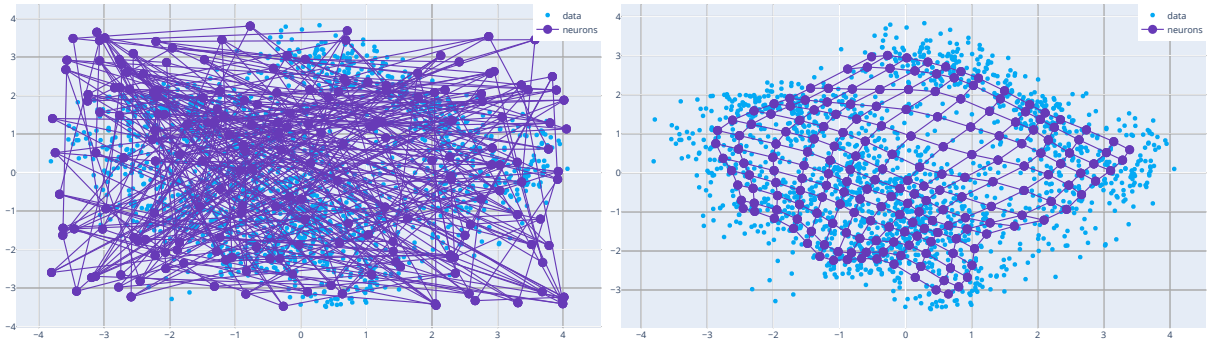
$$\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\tau_1}\right), \tag{2.10}$$

$$\alpha(t) = \alpha_0 \exp\left(-\frac{t}{\tau_2}\right), \tag{2.11}$$

where τ_1 and τ_2 are user-defined decay parameters.

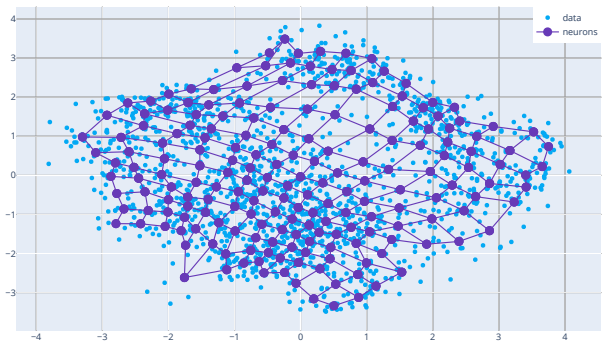
An example of the training of the SOM can be view in Figure 5. In the next section, we review some classification models used in this work.

Figure 5 – Example of the evolution of the training of a SOM network along several epochs and the ordering of the prototype vectors.

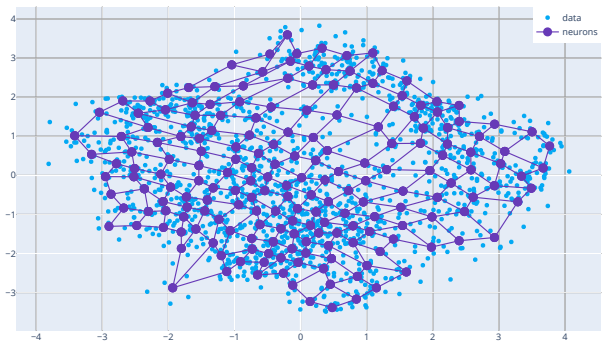


(a) Random initialization

(b) Epoch #3



(c) Epoch #6



(d) Epoch #10

Source: the author (2019).

2.3 Classification models

As mentioned before, in a classification task we have access to a data set $\mathcal{X} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$; where $\mathbf{x}_i \in \mathbb{R}^d$ are the input vectors with dimensionality d, $y_i \in \{L_j\}_{j=1}^{c}$ are the labels in a problem with c distinct classes, and N is the number of samples. In this type of problems we want to find $g(\mathbf{x}, \mathbf{W})$ that is able to predict the class y_{new} of a new sample $\mathbf{x}_{new}$ that was not present in the training data set $\mathcal{X}$.

Giving the categorical nature of the labels and the numerical nature of ML models, one commom aproach to deal with the target value, y_i, is to encode it into the set $\{-1,1\}$ in case of a binary problem ($c=2$) or in the set $\{\mathbf{o}_j\}_{j=1}^{c}$ in a multiclass classification ($c>2$). We define the $\mathbf{o}_j$ i-th dimension, o_{ji}, as follows

$$o_{ji} = \begin{cases} +1, & \text{if } j=i, \\ -1, & \text{otherwise,} \end{cases} \tag{2.12}$$

for example $\mathbf{o}_2 = [-1,+1,-1,\ldots,-1]^T$. This type of encoding was used in simulations with the classifiers to be described in the subsections 2.3.1 and 2.3.2.

2.3.1 Least squares classifier with linear basis functions

In the Least squares classifier with linear basis functions (LSC-LBF), after properly encoding the classes, in a binary classification problem, we have $\forall y_i \in \{-1,1\}$ and we assume that the generator process of our data $\mathcal{X}$ has the form

$$y_i = \mathbf{x}_i^T \mathbf{w} + b + \varepsilon_i, \tag{2.13}$$

where b is the bias and ε_i a white gaussian noise.

To simplify notation, it's usually rewritten

$$\mathbf{x}_i \Leftarrow \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix}, \tag{2.14}$$

$$\mathbf{w} \Leftarrow \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}, \tag{2.15}$$

so we can have in a more compact form

$$y_i = \mathbf{x}_i^T \mathbf{w} + \varepsilon_i. \tag{2.16}$$

To estimate the optimum value of the model parameters $\mathbf{w}$ we minimize the square of the error between the target value, y_i, and the prediction of our model, $g(\mathbf{x}_i, \mathbf{w})$, as presented below

$$\text{minimize} \quad f_o(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N} |y_i - g(\mathbf{x}_i, \mathbf{w})|^2 = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2, \tag{2.17}$$

where $\mathbf{y} = [y_1, y_2, \cdots, y_N]^T$ and $\mathbf{X}$ is the matrix whose rows are the corresponding input vectors $\mathbf{x}_i$.

The problem in Eq. (2.17) is convex and has an analytical solution (BOYD; VANDENBERGHE, 2018)

$$\mathbf{w}_o = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{X}^{\dagger}\mathbf{y}, \tag{2.18}$$

where $\mathbf{X}^{\dagger}$ is known as the pseudo-inverse of $\mathbf{X}$.

Given the continuous nature of the output of the $g(\cdot)$ and the target values encoded in the set $\{-1, 1\}$, it becomes necessary to use the sign function to encode the model's prediction. The discriminating function of the model becomes

$$g(\mathbf{x}) = \operatorname{sign}\left(\mathbf{x}^T\mathbf{w}_o\right). \tag{2.19}$$

Multiclass classification problems with $c > 2$ classes can be seen as a set of distinct c binary classification problems, one may build a set of $\{g_j\}_{j=1}^{c}$ classifiers where each g_i evaluates if $\mathbf{x}$ belongs to i-th class or not. The optimum parameter's value can be obtained by constructing an optimization problem similar to Eq. (2.17):

$$\text{minimize} \quad f_o(\mathbf{W}) = \frac{1}{2}\sum_{j=1}^{c}\|\mathbf{y} - g_j(\mathbf{X}, \mathbf{w}_j)\|_2^2 = \frac{1}{2}\|\mathbf{Y} - \mathbf{X}\mathbf{W}\|_2^2, \tag{2.20}$$

where each column of the $\mathbf{W}$ matrix is the weight vector $\mathbf{w}_j$, or parameters of the j-th classifier, and each row of $\mathbf{Y}$ matrix is equal to $\mathbf{y}_i^T$. One may find the optimum value of $\mathbf{W}$ with

$$\mathbf{W}_o = \mathbf{X}^{\dagger}\mathbf{Y}. \tag{2.21}$$

The discriminating function then becomes

$$g(\mathbf{x}) = \arg\max\left(\mathbf{x}^T\mathbf{W}_o\right). \tag{2.22}$$

Further details about the least squares classifier can be found on Boyd & Vandenberghe (2018).

2.3.2 Least squares support vector machine

The LSSVM (SUYKENS; VANDEWALLE, 1999) is a variation of the original Support Vector Machine (SVM) in which a slight change in the optimization problem results in a big simplification of finding the optimum parameters.

First, let us consider the optimization problem of an SVM

$$\text{minimize} \qquad f_o(\mathbf{w},b,\boldsymbol{\xi}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i \tag{2.23}$$

$$\text{s.t.} \qquad y_i(\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i)+b) \geq 1-\xi_i, \qquad i=1,...,N \tag{2.24}$$

$$\xi_i \geq 0, \qquad i=1,...,N. \tag{2.25}$$

In the problem defined above our discriminating function has the form

$$g(\mathbf{x}) = \text{sign}\big(\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x})+b\big), \tag{2.26}$$

and $\{\xi_i\}_{i=1}^N$ are the flexible margins of the data samples. It can be zero, in case of a data sample outside the margin of the proposed separating hyperplane or bigger them zero, meaning that the sample can be found within the margin. Finally, the variable C controls the regularization of the classifier, the smaller C is (closer to zero) stronger is the regularization.

One may see the high similarity between Eq. (2.26) and (2.19) as both classifiers are, in a sense, linear. What differentiates them, besides the LSSVM margin, is that LSC-LBF is a linear classifier in input space, the space of the samples $\{\mathbf{x}_i\}_{i=1}^N$, while the SVM/LSSVM are linear classifiers in the **feature space**, the space of the mapped samples $\{\boldsymbol{\phi}(\mathbf{x}_i)\}_{i=1}^N$. The idea of the mapping $\boldsymbol{\phi}(\mathbf{x}_i)$ is that a nonlinear classification problem, such as the one in Figure 2, may become linearly separable, e.g. as Figure 1, in a high dimensional feature space. Instead of handcrafted feature engineering, we utilize a mapping $\boldsymbol{\phi}(\cdot)$ such as

$$\boldsymbol{\phi}\colon \mathbb{R}^d \to \mathbb{R}^q \tag{2.27}$$

$$\mathbf{x} \mapsto \boldsymbol{\phi}(\mathbf{x}), \tag{2.28}$$

where $q > d$. The interesting part is that the mapping $\boldsymbol{\phi}(\mathbf{x})$ does not need to be known for the optimization process and the final discriminating function of a SVM/LSSVM classifier, which can be rewritten in function of the kernel function, defined as

$$K(\mathbf{x}_i,\mathbf{x}_j) = \boldsymbol{\phi}(\mathbf{x}_i)^T\boldsymbol{\phi}(\mathbf{x}_j), \tag{2.29}$$

so, to find the optimum parameters of the model and make new predictions we just need to know how to efficiently compute the dot product of Eq. (2.29), it is not necessary to evaluate, and even know, the mapping $\boldsymbol{\phi}(\mathbf{x})$ of any sample $\mathbf{x}$. Such a feature of the SVM/LSSVM classifiers is known as the "kernel trick"(HOFMANN *et al.*, 2008). Some examples of kernels can be seen in Table 1.

Table 1 – A short list of widely used kernel functions.

Kernel	Description
Linear	$K(\mathbf{x},\mathbf{x}_i) = \mathbf{x}_i^T\mathbf{x} + c$, where c is a constant.
Polynomial	$K(\mathbf{x},\mathbf{x}_i) = \left(\alpha\mathbf{x}_i^T\mathbf{x} + c\right)^d$, where α and c are constants and d is the degree of the polynomial.
Gaussian (RBF)	$K(\mathbf{x},\mathbf{x}_i) = \exp\left\{-\|\mathbf{x}-\mathbf{x}_i\|^2/\sigma^2\right\}$, where σ is a constant.
Cauchy	$K(\mathbf{x},\mathbf{x}_i) = 1/\left(1+\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{\sigma^2}\right)$, where σ is a constant.
Log	$K(\mathbf{x},\mathbf{x}_i) = -\ln(\|\mathbf{x}-\mathbf{x}_i\|^d + 1)$, where d is a constant.

Source: adapted from Souza (2019).

In the LSSVM problem formulation, we change the inequality restrictions to equality ones and in the objective function the flexible margin variables have their values squared

$$\textit{minimize} \qquad f_o(\mathbf{w},b,\boldsymbol{\xi}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \gamma\frac{1}{2}\sum_{i=1}^{N}\xi_i^2 \tag{2.30}$$

$$s.t. \qquad y_i(\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i) + b) = 1 - \xi_i, \qquad i = 1,...,N\,, \tag{2.31}$$

where the γ variable has the same meaning of C.

To find the optimum for the problem formulated by the LSSVM we find its dual representation (BOYD; VANDENBERGHE, 2004). First, we construct its Lagrangian function

$$L(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \gamma\frac{1}{2}\sum_{i=1}^{N}\xi_i^2 - \sum_{i=1}^{N}\alpha_i(y_i(\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i) + b) - 1 + \xi_i), \tag{2.32}$$

where the set $\{\alpha_i\}_{i=1}^N$ are the dual variables associated with the set of equality restrictions of Eq. (2.31), we may represent them as $\boldsymbol{\alpha}$ for compactness. To satisfy the Karush-Khun-Tucker (KKT) conditions we evaluate $\nabla L = 0$ which can be translated in the set of equations below (Rocha Neto, 2017)

$$\frac{\partial L(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha})}{\partial \mathbf{w}} = 0 \quad \Longleftrightarrow \quad \mathbf{w} = \sum_{i=1}^{N}\alpha_i y_i \boldsymbol{\phi}(\mathbf{x}_i), \tag{2.33}$$

$$\frac{\partial L(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha})}{\partial b} = 0 \quad \Longleftrightarrow \quad \sum_{i=1}^{N}\alpha_i y_i = 0, \tag{2.34}$$

$$\frac{\partial L(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha})}{\partial \alpha_i} = 0 \quad \Longleftrightarrow \quad y_i(\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i) + b) - 1 + \xi_i = 0, \tag{2.35}$$

$$\frac{\partial L(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha})}{\partial \xi_i} = 0 \quad \Longleftrightarrow \quad \alpha_i = \gamma\xi_i. \tag{2.36}$$

This set of linear equations drawn from KKT conditions can be beautifully rewritten in a matrix form

$$\begin{bmatrix} 0 & \mathbf{y}^T \\ \mathbf{y} & \boldsymbol{\Omega} + \gamma^{-1}\mathbf{I} \end{bmatrix} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{1} \end{bmatrix}, \tag{2.37}$$

where $\Omega_{i,j} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$, $\mathbf{I}$ is the N x N identity matrix and $\mathbf{1} = [1, 1, \cdots, 1]^T$ with dimensionality N.

As the LSSVM formulation can be recast as a linear system, one way of solving it is by using ordinary least squares

$$\text{minimize} \quad f_o(\boldsymbol{\upsilon}) = \frac{1}{2}\|\mathbf{r} - \mathbf{A}\boldsymbol{\upsilon}\|_2^2, \tag{2.38}$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & \mathbf{y}^T \\ \mathbf{y} & \boldsymbol{\Omega} + \gamma^{-1}\mathbf{I} \end{bmatrix}, \tag{2.39}$$

$$\boldsymbol{\upsilon} = \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}, \tag{2.40}$$

$$\mathbf{r} = \begin{bmatrix} 0 \\ \mathbf{1} \end{bmatrix}. \tag{2.41}$$

We use Eqs. (2.34) and (2.26) to obtain the discriminative function $g(\mathbf{x})$ in terms of the dual optimum variable, α^o, and the kernel function $K(\cdot,\cdot)$

$$g(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i^o y_i K(\mathbf{x}_i, \mathbf{x}) + b_o\right), \tag{2.42}$$

where α_i^o and b_o denote optimum values obtained from solving the linear system in Eq. (2.37).

One may be charmed by the simplified optimization that arises from the LSSVM formulation but it has a big disadvantage compared to Vapnik's SVM: it yields a non-sparse model. One may see with Eqs. (2.30) and (2.36) that α_i is proportional to ξ_i, which is squared in the objective function of the LSSVM problem formulation. As in a case of l_2-regularization the optimization process does not find advantageous to nullify the ξ_i variables, mathematically we can show that

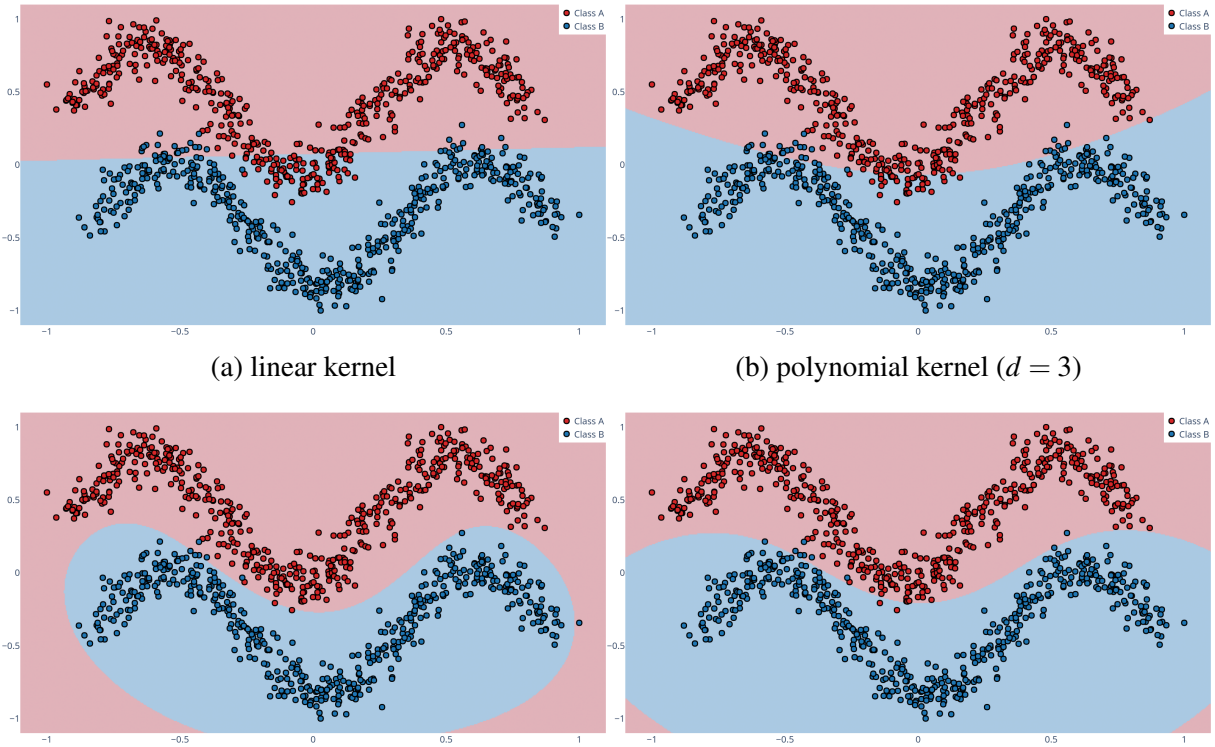
$$\frac{\partial f_o(\mathbf{w}, \boldsymbol{\xi})}{\partial \xi_i} = \gamma \xi_i, \tag{2.43}$$

so, smaller is the ξ_i variable, smaller is the incentive to make it even smaller. The optimization process find its optimum solution with the set $\{\alpha_i\}_{i=1}^N$ containing, usually, no null value, resulting in the need to save all the data set $\mathcal{X}$ as one may see in the discriminating function of Eq. (2.42).

Summing up, the LSSVM classifier is a variation of the Vapnik's SVM that can be optimized by ordinary least squares but results in a non-sparse model and, like SVM, can behave

as a linear or nonlinear classifier depending on the kernel choice. In Figure 6 one may see the different decision boundaries generated by different kernels, from there becomes clear how the kernel trick can introduce nonlinearity to the decision boundaries of a kernel-based classifier.

Figure 6 – Illustration of the decision boundaries of the LSSVM classifier for different kernel functions ($\gamma = 1$ for all simulations).



(a) linear kernel

(b) polynomial kernel ($d = 3$)

(c) polynomial kernel ($d = 5$)

(d) rbf kernel ($\sigma = 1$)

Source: the author (2019).

2.4 Concluding remarks

In this chapter, we talked about local modeling, with a focus on the CLHP approach. We also attached a review of some vector quantization algorithms and classification models, empowering the reader with tools to understand and apply the CLHP.

Now, one may be asking why to use local modeling and we may enumerate its advantages:

1. A nonlinear decision boundary can be generated with a set of linear ones, as can be seen in Figure 2;
2. One may find that a local modeling results in a more interpretable model than a black-box one, e.g., a CLHP with LSC-LBF has the ability to be nonlinear

and have the clear interpretability of a regression model when compared to a multilayer perceptron;

3. On a hard partitioning case, depending on the ML model, one may reduce its requirements for training (e.g. time complexity for SVM and memory requirement for LSSVM) and/or inference (e.g. in LSSVM case fewer data points used in the modeling result in a smaller and faster model during prediction).

As any paradigm in modeling, local classification has its disadvantages, more prominent the addition of a new hyperparameter, i.e. the number of models, and the challenge to properly construct the classifier mapping function, i.e. $\mathbf{h}(\mathbf{x})$.

In the next chapter, we will discuss a novel framework for pattern classification that has a resemblance to CLHP: regional classifiers.

3 REGIONAL CLASSIFIERS: A NOVEL PARADIGM FOR PATTERN RECOGNITION

In this chapter, we present and discuss in depth a new paradigm in pattern recognition: Regional Classifiers (RC). The idea of regional modeling was introduced by Souza Jr. *et al.* (2015) in a task of system identification, a regression problem, and this thesis will extend the idea of regional modeling for classification.

The regional proposal is inspired by the idea of the clustering of the SOM (VESANTO *et al.*, 2000), where a SOM is trained in the data set $\mathcal{X}$ and then the set of SOM units/neurons $\{\mathbf{u}_i\}_{i=1}^{N_u}$ are clustered by some VQ algorithm (e.g. K-means). From the Vesanto *et al.* (2000) paper we can highlight important properties of this two-step approach:

1. The data set can be represented using fewer prototype vectors when comparing to a direct clustering of the samples, which allows efficient use of clustering algorithms to divide the prototypes into groups;
2. The experiments showed that clustering the SOM instead of directly clustering the data is a computationally effective data representation approach;
3. The two-step approach reduces the occurrence of prototypes with no data samples associated to it, a common occurrence in local classifiers.

Bearing these properties in mind, the regional modeling approach is proposed. To better understand the regional classifier we had split the discussion between the training and inference stages.

3.1 Related Works

Although we use the term "regional classifier" to denote the natural extension of Souza Jr. *et al.* (2015) to classification problems, such a term was used before on the literature.

In Spreeuwers *et al.* (2014) paper, the authors aiming to enhance face recognition robustness to variations like illumination, face expression, and hair occlusion use a set of thirty classifiers that operate on overlapping regions of the face. Each classifier has access to a specific hand-crafted partition of 2D images of faces and their outputs are combined by voting. In Lee *et al.* (2013) the authors proposed a tree-based hierarchical multi-model approach. At the first level, a classifier has access to the entire data and is trained if all the training samples, at the second level two classifiers are trained on two non-overlapping partitions of the input space arranged in such a way that each sibling classifier has access to the same number of samples, at the third

level four classifiers are trained on four partitions dividing the input space as aforementioned and so on. During prediction, all the classifiers that have access to the input space where the sample lies made a vote and their votes are combined by a weighted sum.

To avoid confusion on the reader, henceforward, if we used the term "regional"we are referring to the stated new approach of regional modeling based on prototypes. So, when we use "regional classifiers", "regional modeling", "data regions" we would be talking about the new paradigm introduced by this thesis.

3.2 Training in RC

Given a data set $\mathcal{X}$, the training of a regional classifier can be divided into the steps:

1. The SOM is trained using the whole data set $\mathcal{X}$, in order to build a compact representation of $\mathcal{X}$ by means of the prototype vectors;
2. The set of SOM's units $\{\mathbf{u}_i\}_{i=1}^{N_u}$ are split into K partitions using a suitable clustering algorithm (e.g. K-means);
3. A set of K regional classifiers are built. For this purpose, each k-th regional classifier, $k = 1,\ldots,K$, is built using the data samples mapped to the SOM prototypes within the k-th partition.

For a more specific explanation, if using K-means as the clustering algorithm, the RC training process becomes as follows.

1. The SOM is trained in the data set $\mathcal{X}$, as in Figure 7a;
2. The set SOM's units $\{\mathbf{u}_i\}_{i=1}^{N_u}$ are clustered, or split, into K partitions using K-means, each partition V_j defined as

$$V_j = \left\{\mathbf{u} \in \{\mathbf{u}_i\}_{i=1}^{N_u} \,\middle|\, \|\mathbf{u}-\mathbf{p}_j\|_2^2 = \min_{1\leq l\leq K} \|\mathbf{u}-\mathbf{p}_l\|_2^2\right\}, \tag{3.1}$$

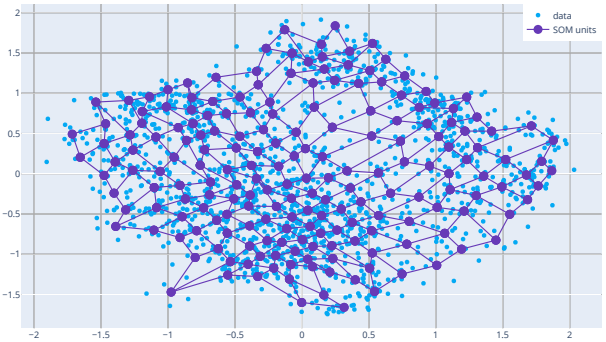
   where $\{\mathbf{p}_j\}_{j=1}^{K}$ is the set of K-means prototypes. An example of this clustering of the SOM can be view in Figure 7b;
3. Each prototype $\mathbf{p}_i$ define a **region**, $\mathcal{R}_i$, in input space of the form

$$\mathcal{R}_i = \left\{\mathbf{x} \in \mathbb{R}^d \middle| t^* = \arg\min_{1\leq t\leq N_u} \|\mathbf{x}-\mathbf{u}_t\|_2^2 \implies \|\mathbf{u}_{t^*}-\mathbf{p}_i\|_2^2 = \min_{1\leq j\leq K} \|\mathbf{u}_{t^*}-\mathbf{p}_j\|_2^2\right\}, \tag{3.2}$$

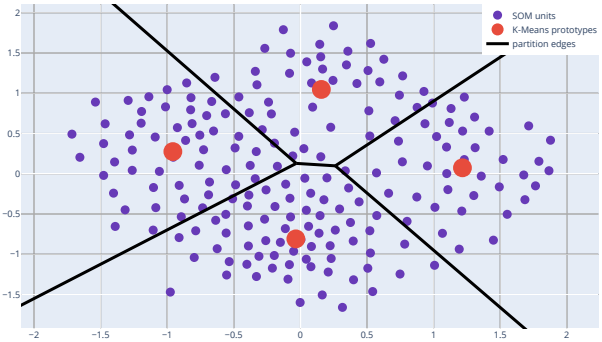
   as one may see an example of data regions in Figure 7c;
4. As in the case of CLHP, one may construct a regional classifier for each data region $\mathcal{R}_i$ with linear models (e.g. LSC-LBF or LSSVM with the linear kernel) or

a nonlinear one (e.g. LSSVM with the gaussian or polynomial kernel), resulting in a set of $\{g_i(\cdot)\}_{i=1}^{K}$ discriminative functions.

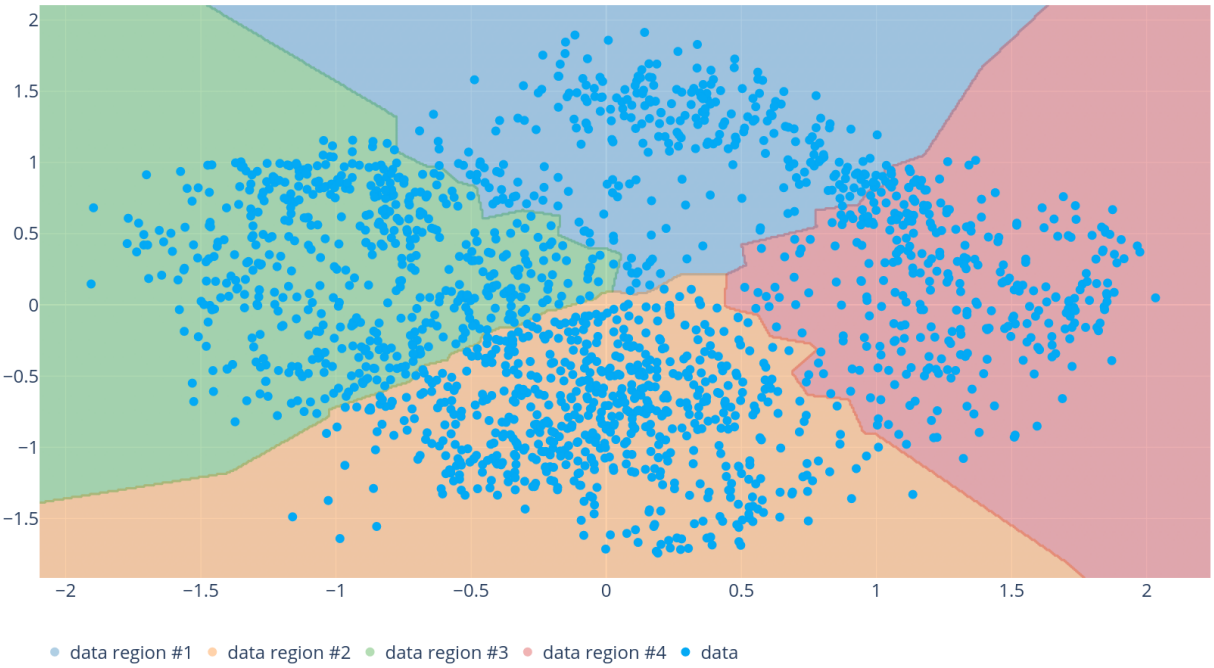
Figure 7 – Example of the training process of a regional classifier.



(a) A SOM is trained in the data set



(b) Clustering of the SOM with K-means



(c) Data regions for RC

Source: the author (2019).

More caution needs to be given for the use of the terms data cluster and data region. A **data cluster** is a set of data samples mapped to a specific prototype, e.g. data samples mapped to a prototype of the K-means algorithm in the CLHP approach, while a **data region** is a set of data samples mapped to a cluster of SOM units, and this cluster of SOM units are then mapped to a specific prototype on the regional modeling approach. For a graphical representation of the data regions, the reader can examine Figures 7c and 8.

3.3 Inference in RC

Analogous to the training step, for each new sample $\mathbf{x}$ we should proceed as follows.

1. We estimate the winning neuron using some similarity measure, e.g. by computing the minimum Euclidean distance from $\mathbf{x}$ to the SOM units $\{\mathbf{u}_j\}_{j=1}^{N_u}$:

$$j^* = \arg\min_{1 \leq j \leq N_u} \|\mathbf{x} - \mathbf{u}_j\|_2^2 \tag{3.3}$$

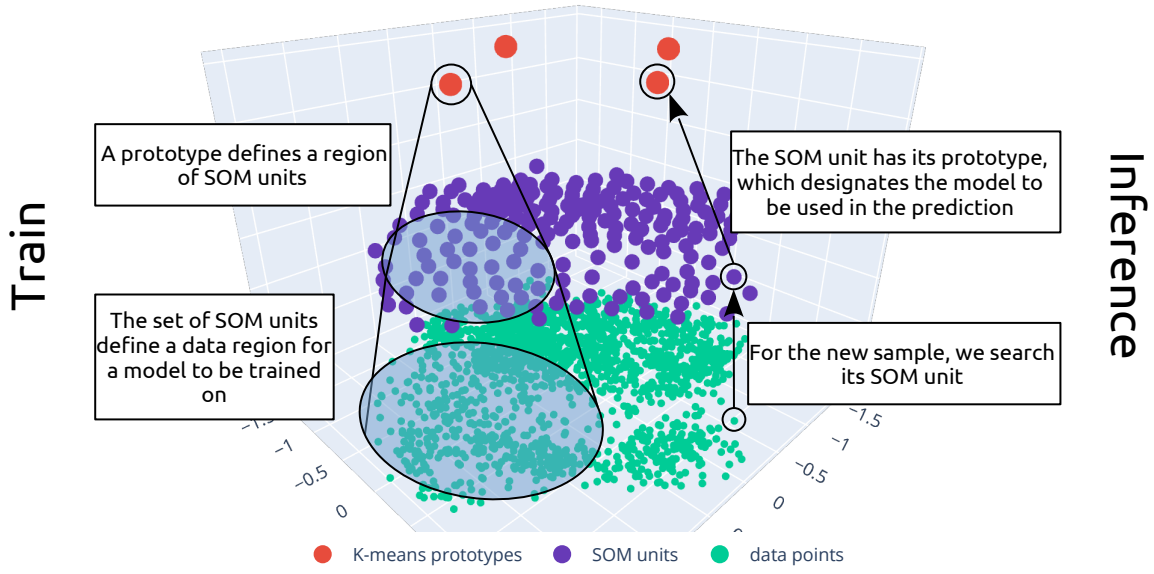
2. Find the most similar clustering prototype to the winning neuron, e.g. by computing the minimum Euclidean distance from $\mathbf{u}_{j^*}$ to the K-means prototypes $\{\mathbf{p}_i\}_{i=1}^{K}$:

$$i^* = \arg\min_{1 \leq i \leq K} \|\mathbf{u}_{j^*} - \mathbf{p}_i\|_2^2 \tag{3.4}$$

3. As $\mathbf{x}$ belongs to region $\mathcal{R}_{i^*}$ the most suitable model to make the prediction is the one with function $g_{i^*}(\cdot)$.

For a graphical representation of the search of the region of a new sample, the reader can examine Figure 8.

Figure 8 – Training and inference in RC.



Source: the author (2019).

3.3.1 *Major Drawbacks of the RC Approach*

Analogous phenomena to the presented in CLHP (Section 2.1.3) can be encountered in the RC approach, namely, empty and homogeneous regions.

Unlike the local classification approach, empty regions are of rare occurrence in the RC approach. Nevertheless, it may happen that a region end up with no associated data samples after training the SOM and the subsequent K-means algorithm. Since this occurrence makes unfeasible the construction of a model for that region, as a solution, we may suggest tagging the prototype and, if any, its associated SOM units as dead prototypes/units and then searching for the next similar prototype/unit.

Homogeneous regions are also rarer in regional modeling given the sparse representation of the data set by fewer prototypes. In a similar fashion as in CLHP, we suggest the use of a bias model, if any new sample seems to lie in a homogeneous region its label should be equal to its fellow data points.

3.4 Concluding remarks

In this chapter, we presented regional classifiers with a bit of theory and visualizations to ground this new paradigm. As noticeable differences from the CLHP paradigm, we may cite the sparse representation of RC that gives rise to fewer homogenous regions and may reduce the search space for the hyperparameter K, or the number of models. In the next chapter, we present simulations that evaluate the properties of these paradigms.

4 SIMULATION METHODOLOGY

In this chapter, we present the simulation methodology used in our experiments. This part of the thesis essentially contains information about the data sets used, the design of the experiments with its objectives and a summary of some cluster validation metrics, that were applied to reduce the search space of hyperparameter K (number of partitions/models).

The data sets used were selected from the UCI Machine Learning Repository (DUA; GRAFF, 2017) and they are described in Table 2. 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 *wall-following* data set provides three variations, with 2, 4 and 24 input features. A summary of the data sets can be found below.

- **Parkinson**: biomedical voice measurements from people with and without Parkinson's disease (LITTLE, 2008);
- **Vertebral Column**: six biomechanical attributes derived from the shape and orientation of the pelvis and lumbar spine. The patient's condition was classified as Normal, Disk Hernia or Spondylolisthesis (BARRETO *et al.*, 2011);
- **Wall-following Robot**: choose between 4 actions (Move-Forward, Slight-Right-Turn, Sharp-Right-Turn, Slight-Left-Turn) as the robot navigates the room following the wall, using ultrasound sensors arranged circularly around its 'waist' (FREIRE *et al.*, 2010).

An important note to add is that the *wall-following* and *vertebral column* were generated by our research group and the reader may find previous works using these data sets (FREIRE, 2009; NETO, 2011).

Table 2 – Summary of data sets

Data set	**# of classes**	**# of samples**	**# of features**
Parkinson	2	195	22
Vertebral Column	2 or 3	310	6
Wall-Following	4	5456	2, 4 or 24

Source: the author (2019).

A total of 4 experiments have been performed, which can be listed as follows.

1. Global vs. Local least squares classifier with linear basis functions (LSC-LBF).
2. Global vs. Local least squares support vector machine (LSSVM).
3. Global vs Regional LSC-LBF

4. Global vs Local vs Regional LSSVM

Each one of the experiments will get a separated section moving forward in this chapter.

4.1 Global vs. Local least squares classifier with linear basis functions (LSC-LBF)

The objective of this experiment is to compare the global and the CLHP approaches using a linear model and evaluate the influence of the number of partitions, K, in model performance. During training, we executed 100 independent hold-out runs, splitting the avaliable data samples into 80% for training and 20% for testing. Features are re-scaled to the range [0,1] and the K-means was re-initialized randomly 10 times.

Results of this experiment can be seen in Section 5.1.

4.2 Global vs. Local least squares SVM (LSSVM)

The objective of this experiment is to compare the global versus the CLHP approach using a nonlinear model, i.e. LSSVM with gaussian kernel, using a committee of cluster validation metrics to reduce the search space of the K hyperparameter. As a training method we executed 50 independent stratified hold-out runs, that is, maintaining the same proportion of each class in every hold-out run. All runs had 50% of the data for training and 50% for testing. Features were re-scaled to the range [0, 1] and the K-means was re-initialized randomly 10 times.

During hyperparameter optimization a grid search was made for the σ constant of the LSSVM gaussian kernel on the set $\{10^{-0.5}, 10^{-0.375}, 10^{1.25}, 10^{2.125}, 10^{3}\}$ and for the regularization variable γ on the set $\{10^{-6}, 10^{-4}, 10^{-2}, 10^{0}, 10^{2}, 10^{4}, 10^{6}\}$. For the number of partitions, the committee of cluster validation metrics analyzed the set $K \in \{2, 3, ..., \lfloor\sqrt{N}\rfloor\}$ and each one made a proposal for the optimum value of K, subsequently this set of proposals was added to the grid search. The 5-fold stratified cross-validation was the methodology adopted for model selection during hyperparameter optimization, and its objective function had the form

$$f_o(a) = mean(a) - 2std(a), \tag{4.1}$$

where a represents the accuracy during validation, which is a random variable. The reader may find below the list of the 12 validation metrics used (BEZDEK; PAL, 1998):

1. Adjusted Rand Index (ARI);
2. Adjusted Mutual Information (AMI);

3. V-measure (VM);
4. Fowlkes-Mallows (FM);
5. Silhouette (SI);
6. Calinski-Harabasz (CH);
7. Davies-Bouldin (DB);
8. Dunn (DU);
9. Modified Final Prediction Error (mFPE);
10. Modified Akaike Information Criteria (mAIC);
11. Modified Bayesian Information Criteria (mBIC);
12. Modified Minimum Description Length (mMDL);

A detail discussion of the metrics can be found in Section 4.5 while the results of this experiment are reported in Section 5.2.

4.3 Global vs Regional LSC-LBF

The objective of this experiment is to compare global versus regional approaches using a linear model, i.e. the LSC-LBF. As a training method, we executed 100 independent hold-out experiments, with 80% of the data for training and 20% for testing and features were re-scaled to the range [0, 1]. The number of SOM units was set to $N_{SOM} = 5\sqrt{N_{tr}}$, where N_{tr} is the number of samples in train set and the K-means algorithm was re-initialized randomly 10 times.

For choosing the value of K a search was made for $K \in \{2, 3, ..., \lfloor\sqrt{N_{SOM}}\rfloor\}$ and the K_{opt} was the one who showed the lowest Davies-Bouldin index (see Section 4.5.2.3 for more about this metric). The results of this experiment can be found in Section 5.3.

4.4 Global vs Local vs Regional LSSVM

The objective of this experiment is to compare the three approaches with a nonlinear model, i.e. LSSVM with rbf kernel. For the Regional LSSVM (R-LSSVM), we used as a training method 50 independent stratified hold-out experiments, with 50% of the data for training and the other 50% for testing. Features were re-scaled to the range [0, 1] and the K-means algorithm was re-initialized randomly 10 times. During hyperparameter optimization using grid search the same sets used in the CLHP version of the LSSVM (L-LSSVM) was used for the R-LSSVM, were considered the set $\{10^{-0.5}, 10^{-0.375}, 10^{1.25}, 10^{2.125}, 10^{3}\}$ for the σ variable and

the set $\{10^{-6}, 10^{-4}, 10^{-2}, 10^{0}, 10^{2}, 10^{4}, 10^{6}\}$ for γ variable. The set of possible values for K_{opt}, the optimum number of partitions, was drawn from the proposals of the committee of cluster validation metrics after searching $K \in \{2, 3, ..., \lfloor\sqrt{N_{SOM}}\rfloor\}$. The procedure adopted for model selection was 5-fold stratified cross-validation with the objective function of Eq. (4.1).

The methodology applied to the Global LSSVM (G-LSSVM) and its local counterpart (L-LSSVM), was presented in Section 4.2. What differs local and regional approach is the committee of metrics, as Adjusted Rand Index, Adjusted Mutual Information, V-measure, and Fowlkes-Mallows are supervised metrics and the SOM units are unlabeled, they were removed for the committee for the R-LSSVM case. The results of this experiment are reported in Section 5.4.

In Table 3 the reader can find a compact summary of the characteristics of the experiments proposed in this thesis.

Table 3 – Summary of the experiments.

Experiment	**# of runs**	**Train/test split**	**Hyperparam. opt.**	**# of metrics**
G. vs. L. LSC-LBF	100	80%/20%	–	–
G. vs. L. LSSVM	50	50%/50%	σ and γ	12
G. vs. R. LSC-LBF	100	80%/20%	–	1
G. vs. L. vs. R. LSSVM	50	50%/50%	σ and γ	12/8

Source: the author (2019).

4.5 Cluster validation metrics

In this section, we will describe briefly the cluster validation metrics used in the experiments carried out in this thesis. For each train/test split a search was executed for $K \in \{2, 3, ..., \lfloor\sqrt{N_{SOM}}\rfloor\}$ for the regional case and $K \in \{2, 3, ..., \lfloor\sqrt{N_{tr}}\rfloor\}$ for the CLHP case and all members of the comitte $\{C_i\}_{i=1}^{M}$ suggested an optimal K. After the commitee evaluation we took the set of optimal sugestions $\{K_{opt}^{(C_i)}\}_{i=1}^{M}$ and optimize K like the other hyperparameters.

We had divided the metrics into three categories, namely, supervised, unsupervised, and information criteria-based metrics. **Supervised metrics** requires labeled data and as the SOM units are unlabeled they were not used in regional modeling. **Unsupervised metrics** evaluate the goodness of a clustering result on unlabeled data, those indices were used both in local and regional approaches. **Information criteria-based metrics** are variations of indices used in system identification and time series, they were obtained from the work of Sousa (2019).

As the data does not need to be labeled for these metrics, they were used both in regional modeling and CLHP.

4.5.1 Supervised metrics

As mentioned previously, this type of metric requires labeled data. A total of four supervised metrics will be described in the following.

4.5.1.1 Adjusted Rand Index (ARI)

The original Rand index (RI) is then given by the following equation:

$$RI(K) = (a+b)/\binom{N}{2}, \tag{4.2}$$

where a is the number of pairs of elements that have the same label L_i and are in the same partition V_i and b the number of pairs of elements that have different labels $L_i \neq L_j$ and are different partitions $V_i \neq V_j$.

The RI metric is vulnerable to a random assignment of labels so, we discount the expected RI, $E[RI]$, of random labeling and define the adjusted Rand index as

$$ARI(K) = \frac{RI - E[RI]}{\max(RI) - E[RI]}. \tag{4.3}$$

Higher values on these metrics represent better clustering. More about this index can be found in Hubert & Arabie (1985).

4.5.1.2 Adjusted Mutual Information (AMI)

In a nutshell, the mutual information is a measurement of the agreement between two labels assignments. Let U and V be two ways of partioning the data set $\mathcal{X}$ of N samples into non-overlap subsets

$$U = \{U_1, U_2, \cdots, U_R\}, \tag{4.4}$$

$$V = \{V_1, V_2, \cdots, V_C\}, \tag{4.5}$$

and consider $a_i = |U_i|$ as the number of samples in U_i, $b_j = |V_j|$ the number of samples in V_j, and $n_{ij} = |U_i \cap V_j|$ the number of samples that are present in both U_i and V_j. The entropy of U can be evaluated by

$$H(U) = -\sum_{i=1}^{R} \frac{a_i}{N} \ln\left(\frac{a_i}{N}\right), \tag{4.6}$$

The Mutual Information (MI) between U and V is measured by the equation

$$MI(U,V) = \sum_{i=1}^{R}\sum_{j=1}^{C}\frac{n_{ij}}{N}\ln\left(\frac{n_{ij}}{N}\right). \tag{4.7}$$

As in the case of the Rand index, the mutual information is vulnerable to random labeling with an increase in number of partitions. To fight this phenomenon we use an adjusted version of it, with the expected value for the mutual information been calculated by (VINH *et al.*, 2009)

$$E[MI(U,V)] = \sum_{i=1}^{R}\sum_{j=1}^{C}\sum_{n_{ij}=(a_i+b_j-N)^+}^{\min(a_i,b_j)}\frac{n_{ij}}{N}\ln\left(\frac{Nn_{ij}}{a_i b_j}\right)\frac{a_i!b_j!(N-a_i)!(N-b_j)!}{N!n_{ij}!(a_i-n_{ij})!(b_j-n_{ij})!(N-a_i-b_j+n_{ij})!}. \tag{4.8}$$

Now the adjusted mutual information can be evaluated as

$$AMI(K) = \frac{MI - E[MI]}{\text{mean}(H(U),H(V)) - E[MI]}. \tag{4.9}$$

Higher values of these metrics indicate better clustering. More about this metric can be found in Vinh *et al.* (2009) and Vinh *et al.* (2010).

4.5.1.3 V-measure (VM)

The VM was proposed by Rosenberg & Hirschberg (2007). Let $L = \{l_i\}_{i=1}^{n}$ be a set of labels, $V = \{v_i\}_{i=1}^{m}$ a set of clusters and a_{ij} the number of samples that are members of class l_i and partition v_j. Then, we can define two objectives of a clustering task: homogeneity and completeness. **Homogeneity** is high when each partition contains only vectors of a single class, we define as

$$h = 1 - \frac{H(L|V)}{H(L)}, \tag{4.10}$$

where

$$H(L|V) = -\sum_{v=1}^{m}\sum_{l=1}^{n}\frac{a_{lv}}{N}\ln\left(\frac{a_{lv}}{\sum_{l=1}^{n}a_{lv}}\right), \tag{4.11}$$

$$H(L) = -\sum_{l=1}^{n}\frac{\sum_{v=1}^{m}a_{lv}}{n}\ln\left(\frac{\sum_{v=1}^{m}a_{lv}}{n}\right). \tag{4.12}$$

Completeness has it's maximum value when all samples of a class can be found in the same partition, it can be define as

$$c = 1 - \frac{H(V|L)}{H(V)}, \tag{4.13}$$

where

$$H(V|L) = -\sum_{l=1}^{n}\sum_{v=1}^{m}\frac{a_{lv}}{N}\ln\left(\frac{a_{lv}}{\sum_{v=1}^{m}a_{lv}}\right), \tag{4.14}$$

$$H(V) = -\sum_{v=1}^{m}\frac{\sum_{l=1}^{n}a_{lv}}{n}\ln\left(\frac{\sum_{l=1}^{n}a_{lv}}{n}\right). \tag{4.15}$$

The V-measure criteria balances homogeneity and completeness in a single equation as

$$VM(K) = 2\frac{hc}{h+c}. \tag{4.16}$$

Higher values on this metric mean better clustering. More about this metric can be found in Rosenberg & Hirschberg (2007).

4.5.1.4 Fowlkes-Mallows (FM)

First proposed by Fowlkes & Mallows (1983), the FM index can be easy computed as follows:

$$FM(K) = \frac{TP}{\sqrt{(TP+FP)(TP+FN)}}, \tag{4.17}$$

where TP is the number of true positives, the number of pair of points that have the same partition and class. FP is the number of false positives, the number of pair of points that have the same class but are in different partitions. FN is the number of false negatives, the number of pair of points that have the same partition but have different classes. Higher values of FM indicates better clustering.

4.5.2 Unsupervised metrics

As mentioned previously, this type of metric uses unlabeled data. In this scenario, the four metrics discussed below were members of the committee of validation metrics for the local and regional approaches.

4.5.2.1 Silhouette (SI)

The SI metric was proposed by Rousseeuw (1987) and measures the effectiveness of a cluster proposal based on the proximity within the cluster and the distance of patterns from adjacent clusters.

To evaluate this metric we define $a(\mathbf{x}_i)$ as the mean dissimilarity of vector $\mathbf{x}_i$ over all other vectors of partition V_i and $d(\mathbf{x}_i, V_j)$ as the mean dissimilarity of the pattern $\mathbf{x}_i$ relative to V_j partition members. Let $b(\mathbf{x}_i)$ be the smallest mean dissimilarity of $\mathbf{x}_i$ to all other clusters, it can be evaluate with

$$b(\mathbf{x}_i) = \min_{1 \leq j \leq K \mid V_i \neq V_j} [d(\mathbf{x}_i, V_j)]. \tag{4.18}$$

where $\{V_i\}_{i=1}^{K}$ represents the set of partitions/clusters.

Therefore, the silhouette of vector $\mathbf{x}_i$ is represented by $\delta(\mathbf{x}_i)$ and is calculated with

$$\delta(\mathbf{x}_i) = \begin{cases} 1 - a(\mathbf{x}_i)/b(\mathbf{x}_i), & a(\mathbf{x}_i) < b(\mathbf{x}_i) \\ 0, & a(\mathbf{x}_i) = b(\mathbf{x}_i) \\ b(\mathbf{x}_i)/a(\mathbf{x}_i) - 1, & a(\mathbf{x}_i) > b(\mathbf{x}_i) \end{cases} \tag{4.19}$$

Finally, the silhouette value of a clustering is given by

$$SI(K) = \frac{1}{N} \sum_{i=1}^{N} \delta(\mathbf{x}_i). \tag{4.20}$$

This validation metric can be applied to both hyperspherical and arbitrary shaped clusters and its main disadvantage is the high computational cost. For this index, the higher the value the better is the clustering.

4.5.2.2 Calinski-Harabasz (CH)

The CH index, proposed by Caliński & Harabasz (1974), can be evaluated by the following expression

$$CH(K) = \frac{\mathbf{B}_K/(K-1)}{\mathbf{W}_K/(N-K)}, \tag{4.21}$$

where

$$\mathbf{B}_K = \sum_{i=1}^{K} N_i (\mathbf{p}_i - \mathbf{x}_\mu)(\mathbf{p}_i - \mathbf{x}_\mu)^T, \tag{4.22}$$

$$\mathbf{W}_K = \sum_{i=1}^{K} \sum_{\forall \mathbf{x} \in V_i} (\mathbf{p}_i - \mathbf{x})(\mathbf{p}_i - \mathbf{x})^T, \tag{4.23}$$

are respectively the dispersion matrix between clusters ($\mathbf{B}_K$) and the intragroup dispersion matrix ($\mathbf{W}_K$) with $\mathbf{x}_\mu$ being the mean vector of all samples and N_i and $\mathbf{p}_i$ been respectively the number of samples and the prototype of partition V_i.

The optimum K, or the number of clusters, for this metric is the one that maximizes its value.

4.5.2.3 Davies-Bouldin (DB)

The DB metric, proposed by Davies & Bouldin (1979), is defined by the ratio of the sum of dispersion within to between clusters. This index can be evaluated with following expression:

$$DB(K) = \frac{1}{K}\sum_{i=1}^{K} R_{i,qt}, \tag{4.24}$$

where $R_{i,qt}$ is the ratio between dispersion within and between clusters that can be define as

$$R_{i,qt} = \max_{1\leq j\leq K | j\neq i}\left[\frac{S_{i,q}+S_{j,q}}{d_{ij,t}}\right], \tag{4.25}$$

with

$$S_{i,q} = \left[\frac{1}{N_i}\sum_{\forall \mathbf{x}\in V_i} \|\mathbf{p}_i - \mathbf{x}\|^q\right]^{1/q}, \tag{4.26}$$

representing the internal dispersion of the i-th cluster and

$$d_{ij,t} = \|\mathbf{p}_i - \mathbf{p}_j\|_t, \tag{4.27}$$

being the distance between partitions V_i and V_j using the t-norm and q a constant usually set to 2.

For this metric values closer to zero indicate a more suitable clustering result.

4.5.2.4 Dunn (DU)

The Dunn index, proposed by Dunn (1973), is represented by the following expression:

$$DU(K) = \frac{\min_{i\neq j}[\delta(V_i,V_j)]}{\max_{1\leq l\leq K}[\Delta(V_l)]}, \tag{4.28}$$

where $\delta(V_i,V_j)$ denotes the dissimilarity between partitions V_i and V_j and $\Delta(V_l)$ measures the dispersion within partition V_l. We can define these functions as follows

$$\delta(V_i,V_j) = \min_{\forall \mathbf{x}_i\in V_i,\ \forall \mathbf{x}_j\in V_j}[d(\mathbf{x}_i,\mathbf{x}_j)], \tag{4.29}$$

$$\Delta(V_l) = \max_{\forall \mathbf{x}_i,\mathbf{x}_j\in V_l}[d(\mathbf{x}_i,\mathbf{x}_j)], \tag{4.30}$$

where $d(\mathbf{x}_i,\mathbf{x}_j)$ can be any dissimilarity measure between vectors, e.g. euclidean distance.

The main limitations of this index are its complexity and noise sensitivity. High values in this metric mean better clustering.

4.5.3 Information criteria-based metrics

This section contains modified information criteria metrics so that they can be used to evaluate partitions formed through clustering algorithms. First proposed by Sousa (2019) the base for these metric is the Mean Squared Quantization Error (MSQE), which has the form

$$MSQE(K) = \frac{1}{N}\sum_{i=1}^{K}\sum_{\forall \mathbf{x}\in V_i} \|\mathbf{x}-\mathbf{p}_i\|_2^2. \tag{4.31}$$

4.5.3.1 Modified Final Prediction Error (mFPE)

The Final Prediction Error criteria, proposed by Akaike (1969), selects K that minimizes the variance of the mean prediction error while penalizing the excess of model parameters. The modified version of the metric can be evaluated with the following expression

$$mFPE(K) = N\ln\left(\frac{MSQE(K)}{N}\right) + N\ln\left(\frac{N+P}{N-P}\right), \tag{4.32}$$

with the model's order defined as $P = Kd$, where K is the number of prototypes and d the dimensionality of the input vectors.

The first member of the right-hand side of Eq. (4.32) has an exponential decay as K increments and the second member penalizes the excess of parameters, rising its value as K increases. For this criterion the smallest value represents the best K, trying to strike a balance between explaining the data and model complexity.

4.5.3.2 Modified Akaike Information Criteria (mAIC)

The Akaike Information Criteria, proposed by Akaike (1974), determines the order P of the model that minimizes a cost function obtained from concepts of information theory. The cost function associated with the modified version of this criterion is modeled as

$$mAIC(K) = N\ln\left(\frac{MSQE(K)}{N}\right) + 2P, \tag{4.33}$$

where, different from $mFPE$, $2P$ is a linear penalization for the number of parameters in the model.

4.5.3.3 Modified Bayesian Information Criteria (mBIC)

The Bayesian Information Criteria is another metric for model selection, also called the Schwarz Information Criterion (SIC), by the Bayesian interpretation given by Schwarz *et al.*

(1978). The modified version of this metric can be evaluated with the following expression:

$$mBIC(K) = N\ln\left(\frac{MSQE(K)}{N}\right) + P\ln N. \tag{4.34}$$

Similar to *mAIC*, the *mBIC* criterion is a decreasing function of the *MSQE* with the added increasing function of P, however, the *mBIC* metric has a strong penalization for the model's order. The model with the smallest *mBIC* value is the one regarded as optimum.

4.5.3.4 Modified Minimum Description Length (mMDL)

The Minimum Description Length criteria, proposed by Rissanen (1978), is a variant of the AIC metric and its modified version can be evaluated with the following equation:

$$mMDL(K) = N\ln\left(\frac{MSQE(K)}{N}\right) + \frac{P}{2}\ln N. \tag{4.35}$$

In a nutshell, all the 4 informational criteria-based metrics are similar, they try to find a balance between data explanation and model complexity, what differentiates one from another is the strength of the penalization term that controls the number of model parameters.

4.6 Concluding remarks

In this chapter, we presented all the information needed to perform the simulations of Chapter 5 and showed all the 12 metrics used to help finding K_{opt}, the optimal number of clusters (for the local approach) or of regions (for the regional approach). The idea of using several metrics was to investigate through simulations if any of them exhibited a relationship with the objective function of the model selection phase (see Eq. (4.1)), that means, chooses K_{opt} in a manner that the classifier generalizes well (high accuracy on validation set) and consistently (low variance).

In the next chapter, the reader will find all the simulation results and some discussions that can be drawn from them.

5 SIMULATION RESULTS AND DISCUSSION

Throughout this chapter, the reader will find the results of the computer simulations whose methodologies were described in Chapter 4, with charts and tables facilitating the understanding. The results follow the same order of Chapter 4, starting with the CLHP approach and finishing with regional classifiers.

To reduce repetitiveness, the data sets were referred by acronyms as shown in Table 4, models used beside CLHP approach will have "L-" as prefix (e.g. L-LSC and L-LSSVM), used with regional approach an "R-" (e.g. R-LSC and R-LSSVM) and models under the global method will have "G-" (e.g. G-LSC and G-LSSVM).

Table 4 – Data sets acronyms

Data set variation	**Acronym**
Parkinson	*pk*
Vertebral Column with 2 classes	*vc2c*
Vertebral Column with 3 classes	*vc3c*
Wall-Following with 2 features	*wf2f*
Wall-Following with 4 features	*wf4f*
Wall-Following with 24 features	*wf24f*

Source: the author (2019).

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

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN}, \tag{5.1}$$

$$sensitivity = \frac{TP}{TP+FN}, \tag{5.2}$$

$$specificity = \frac{TN}{TN+FP}, \tag{5.3}$$

$$F_1 = 2 \times \frac{precision \times recall}{precision+recall}, \tag{5.4}$$

where TP, TN, FP and FN stand for True-Positive, True-Negative, False-Positive and False-Negative values, respectively and

$$precision = \frac{TP}{TP+FP}, \tag{5.5}$$

$$recall = \frac{TP}{TP+FN}. \tag{5.6}$$

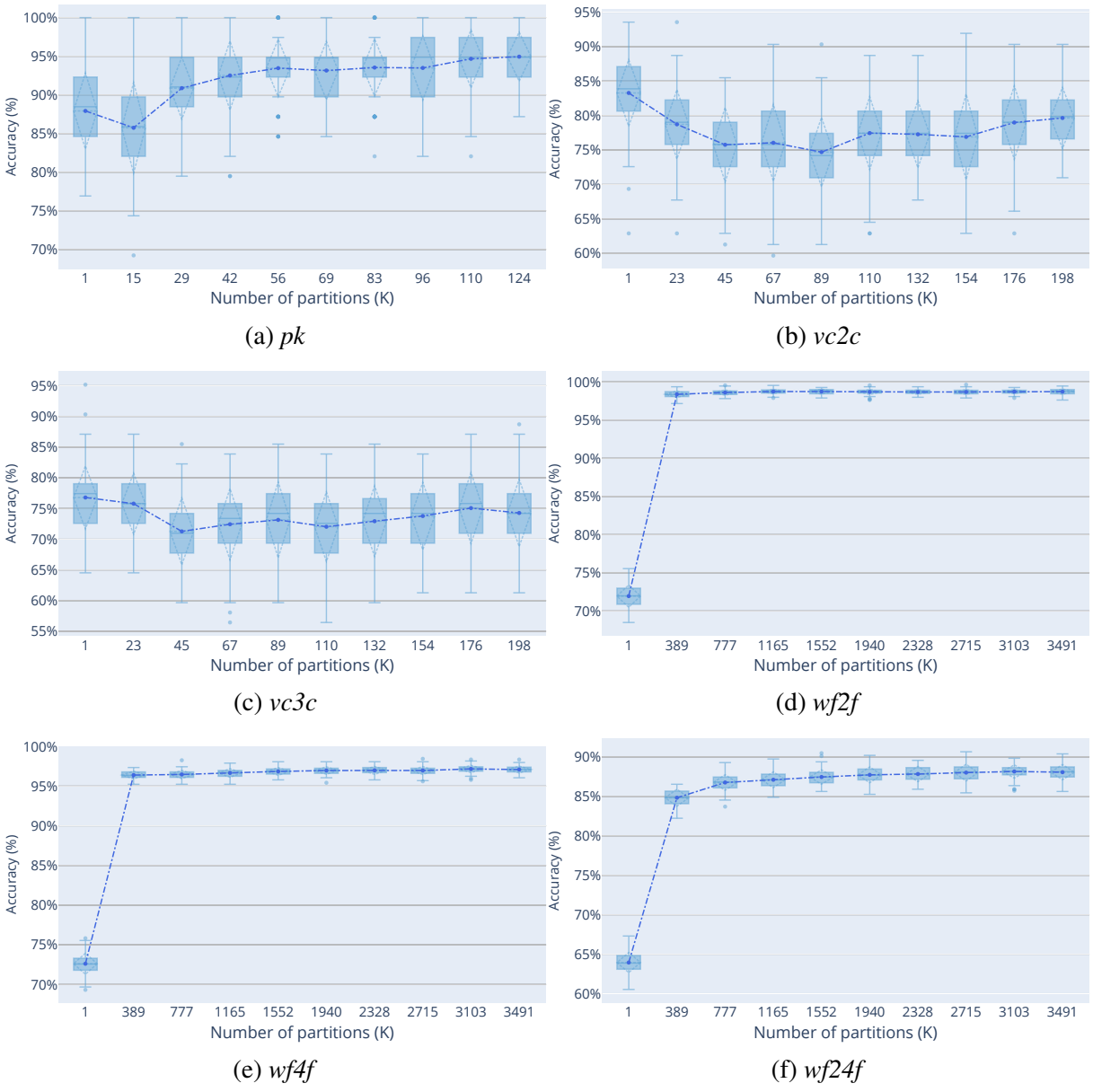
5.1 Global vs Local LSC-LBF results

The idea of this experiment was to verify the capability of a local model, a CLHP with LSC-LBF, to be a good alternative to tackle nonlinear problems and investigate the correlation between model performance and the number of partitions/models. Charts of the six data set variations can be found in Figure 9, where one can find the boxplots of the accuracy in the test set for many values of K, i.e. the number of partitions. One may find ten results ranging from $K = 1$, which means the global model (G-LSC), until $K \approx 0.8N_{tr}$, where N_{tr} symbolizes the number of samples in the train set. To act as a guide, we also added a dark blue dash-dot line connecting the mean value of the distributions.

Based on Figure 9 we can say that the *pk* data set had a slight increase in accuracy (higher mean with lower variance) while the vertebral column data set variations (*vc2c*, *vc3c*) showed a drop in performance and it is not incorrect to state that the G-LSC was better than L-LSC for the values of K. This serves as evidence that the classification problems put by *vc2c* and *vc3c* data sets do not benefit from the local linearization established by L-LSC. On all wall-following data set variations, we found a significant increase in model performance wich give us a sign that the data sets *wf2f*, *wf4f*, and *wf24f* benefit from the local linearization property of the L-LSC. A closer look at the wall-following results can be seen in Figure 10, it shows us that exists a slight trend that as K rises the models' performance rises too. As the wall-following data set has the biggest number of samples, 5456, the second being vertebral column with 310 data points, we noticed fewer empty partitions.

As one may have already realized, as K increases, getting closer to the number of samples, the CLHP approach starts to manifest more homogeneous data partitions (see Section 2.1.3) and is less prone to be comprised of a set of the base model (e.g. the LSC in the L-LSC case) and more on bias models. The biggest value that the number of partitions may have is $K = N_{tr}$ where the CLHP classifier becomes the 1-Nearest Neighbor classifier (1-NN) as each partition has only one sample. So, in Figures 9 and 10 we are not just seeing a correlation between K and model performance but also the gradual transformation of the L-LSC classifier in the 1-NN, this gives us a hint that the 1-NN classifier may be a good proposal for the wall-following data set variations.

Figure 9 – Performance with increasing number of partitions/models in G/L-LSC experiment.



(a) *pk* (b) *vc2c*

(c) *vc3c* (d) *wf2f*

(e) *wf4f* (f) *wf24f*

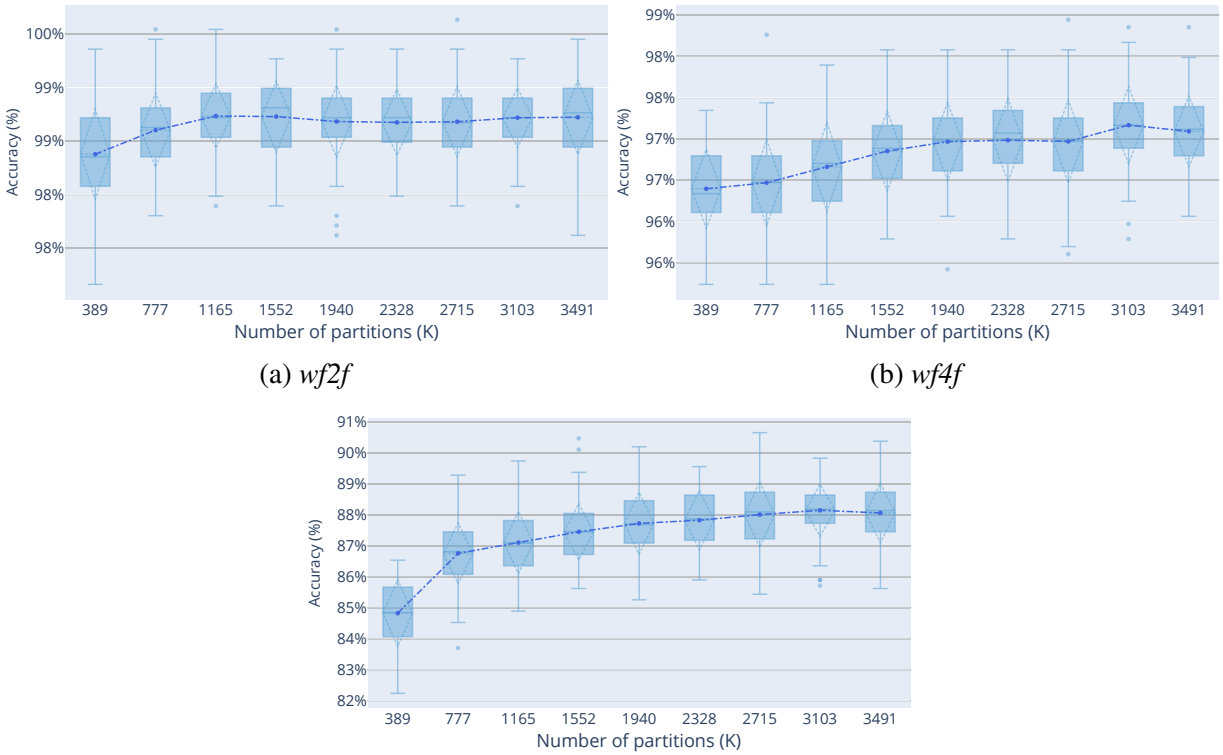
Source: the author (2019).

5.2 Global vs Local LSSVM results

The first results of this test can be found in Figure 11 where the L-LSSVM showed better performance in the train set when compared to G-LSSVM. This outcome is expected as a set of nonlinear models producing a global decision function is more capable of explaining the train set. In the test set, the results were very similar, making it difficult to tell which one is the better model, the exception being the results in *wf2f* data set where the L-LSSVM shows a true increase on accuracy.

To deepen the investigation, the Tables 5 to 10 were built. On Table 5, even if the

Figure 10 – Zooming in the results of *wall-following* data set variations in L-LSC experiment.



(a) *wf2f* (b) *wf4f*

(c) *wf24f*

Source: the author (2019).

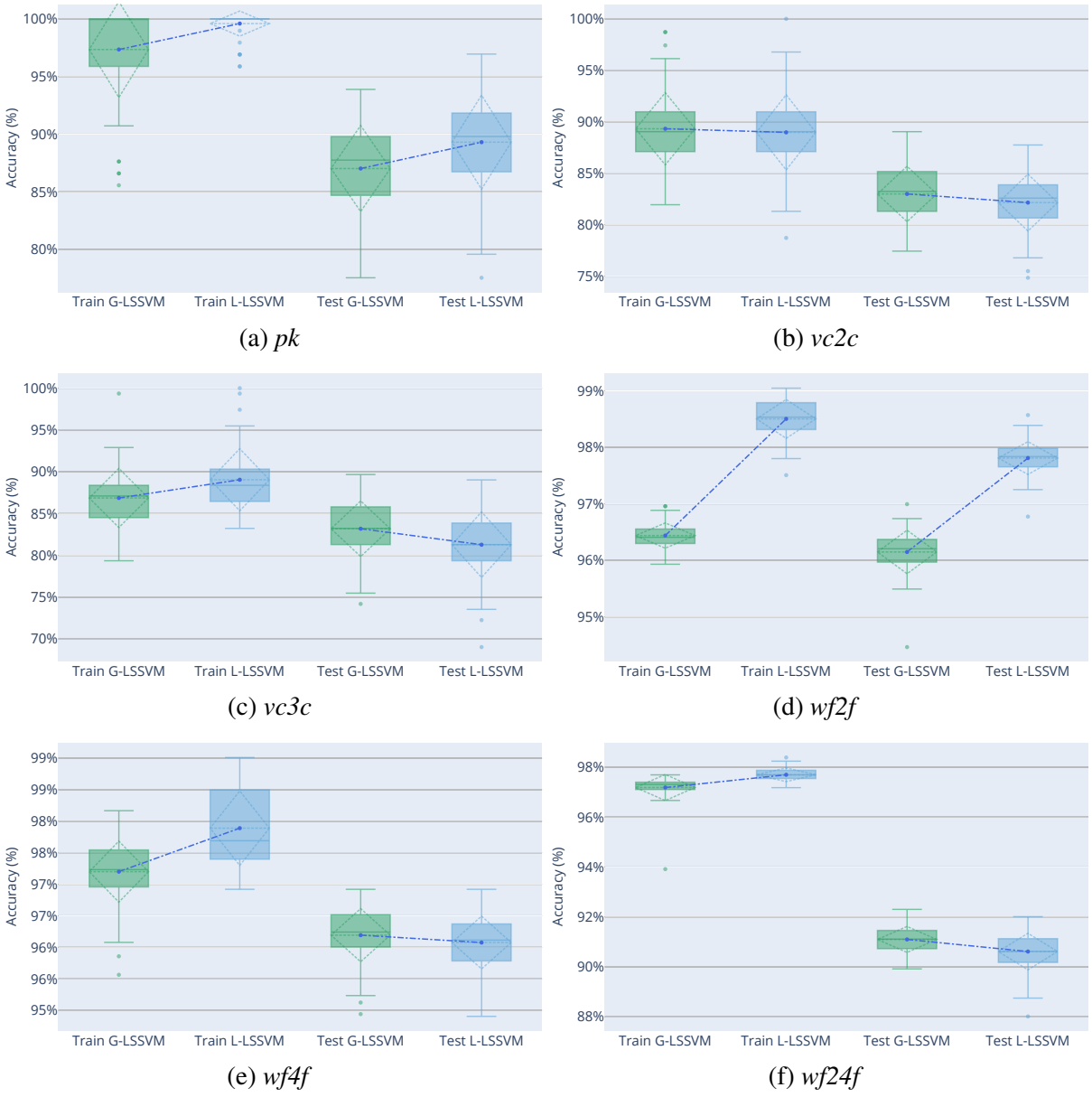
accuracy results being very close, the CLHP version of the LSSVM classifier applied to *pk* data set displayed a much higher specificity, both in train and test sets. This phenomenon may be evidence that the partitioning of the data set favored the emergence of balanced partitions (partitions with roughly the same number of samples per class), leading the LSSVM classifier to not benefit the class with more samples. Tables 6 and 7 confirm that the global and local approach applied with LSSVM has little difference in the vertebral column data set variations. Table 8 reveals a slight but clear superiority of the L-LSSVM classifier in all the four performance metrics used, while in Tables 9 and 10 corroborate the little difference saw between the global and local classifiers applied to the *wf4f* and *wf24f* data sets.

Table 5 – Performance metrics of *pk* data set during G/L-LSSVM experiment

Set	**Model**	**Accuracy (%)**	**Sens. (%)**	**Spec. (%)**	**F_1(%)**
Train	G-LSSVM	97.34 ± 4.16	99.67	90.25	98.32
	L-LSSVM	99.59 ± 1.09	99.92	98.58	99.73
Test	G-LSSVM	87.02 ± 3.73	95.49	60.92	91.72
	L-LSSVM	89.31 ± 4.08	94.30	73.92	93.00

Source: the author (2019).

Figure 11 – Train and test accuracy distributions in G/L-LSSVM experiment.



(a) *pk* (b) *vc2c*

(c) *vc3c* (d) *wf2f*

(e) *wf4f* (f) *wf24f*

Source: the author (2019).

Table 6 – Performance metrics of *vc2c* data set during G/L-LSSVM experiment

Set	**Model**	**Accuracy (%)**	**Sens. (%)**	**Spec. (%)**	**F_1 (%)**
Train	G-LSSVM	89.32 ± 3.51	81.04	93.26	82.88
	L-LSSVM	88.97 ± 3.65	79.28	93.58	81.98
Test	G-LSSVM	82.98 ± 2.69	70.52	88.91	72.53
	L-LSSVM	82.14 ± 2.77	69.68	88.08	71.33

Source: the author (2019).

In Figure 12 one will find the distribution of the optimum number of partitions, K_{opt},

Table 7 – Performance metrics of *vc3c* data set during G/L-LSSVM experiment

Set	Model	Accuracy (%)	Sens (%).	Spec. (%)	F_1 (%)
Train	G-LSSVM	86.86 ± 3.53	93.21	82.45	93.24
	L-LSSVM	89.05 ± 3.71	94.36	85.59	94.47
Test	G-LSSVM	83.19 ± 3.33	91.00	78.52	90.95
	L-LSSVM	81.28 ± 3.92	89.82	76.75	89.81

Source: the author (2019).

Table 8 – Performance metrics of *wf2f* data set during G/L-LSSVM experiment

Set	Model	Accuracy (%)	Sens. (%)	Spec. (%)	F_1 (%)
Train	G-LSSVM	96.44 ± 0.22	98.68	95.76	98.65
	L-LSSVM	98.50 ± 0.34	99.45	98.50	99.44
Test	G-LSSVM	96.15 ± 0.39	98.56	95.06	98.54
	L-LSSVM	97.81 ± 0.29	99.19	97.32	99.19

Source: the author (2019).

Table 9 – Performance metrics of *wf4f* data set during G/L-LSSVM experiment

Set	Model	Accuracy (%)	Sens. (%)	Spec. (%)	F_1 (%)
Train	G-LSSVM	97.20 ± 0.48	98.94	97.27	98.92
	L-LSSVM	97.89 ± 0.59	99.20	97.98	99.19
Test	G-LSSVM	96.19 ± 0.42	98.54	95.64	98.53
	L-LSSVM	96.08 ± 0.42	98.49	95.48	98.49

Source: the author (2019).

Table 10 – Performance metrics of *wf24f* data set during G/L-LSSVM experiment

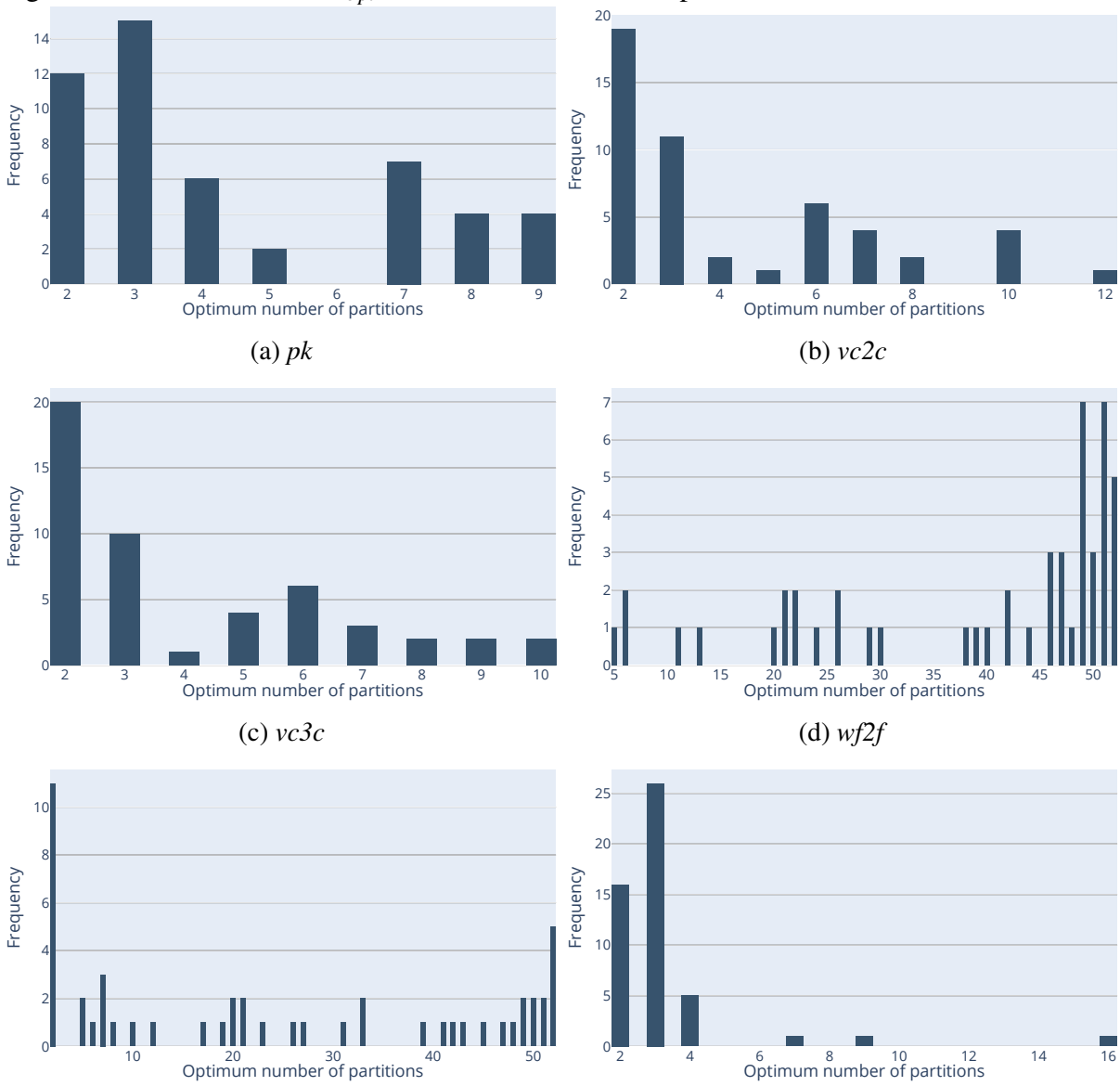
Set	Model	Accuracy (%)	Sens. (%)	Spec. (%)	F_1 (%)
Train	G-LSSVM	97.18 ± 0.52	98.89	96.78	98.92
	L-LSSVM	97.69 ± 0.27	99.09	97.35	99.12
Test	G-LSSVM	91.10 ± 0.52	96.42	89.84	96.47
	L-LSSVM	90.61 ± 0.73	96.23	89.50	96.27

Source: the author (2019).

in the 50 independent runs of the experiment. Albeit we used a committee of 12 cluster validation metrics to build a set of proposals for K_{opt}, we can see a dispersion in the values of K_{opt}. This outcome may indicate that the committee of metrics is unreliable in the task of suggesting the optimal number of partitions when submitted to our model selection objective function (Eq. (4.1)) or that the clustering phase of CLHP had too much randomness (maybe 10 independent

initializations of K-means algorithm was too little to obtain, consistently, the same clustering results). Overall we see a trend in needing fewer partitions in the data sets, usually finding 2 or 3 partitions to be the optimum during simulations, the outlier is in *wf2f* simulation where the optimum number of partitions was around 50, and this granularity may justify the superiority of the local approach when compared to G-LSSVM.

Figure 12 – Distribution of K_{opt} in L-LSSVM for 50 independent runs.



(a) *pk* (b) *vc2c*

(c) *vc3c* (d) *wf2f*

(e) *wf4f* (f) *wf24f*

Source: the author (2019).

The last examination of this experiment lies in the committee of validation metrics hit-rate of K_{opt}; i.e. how many times in the 50 independent runs a metric suggested the true

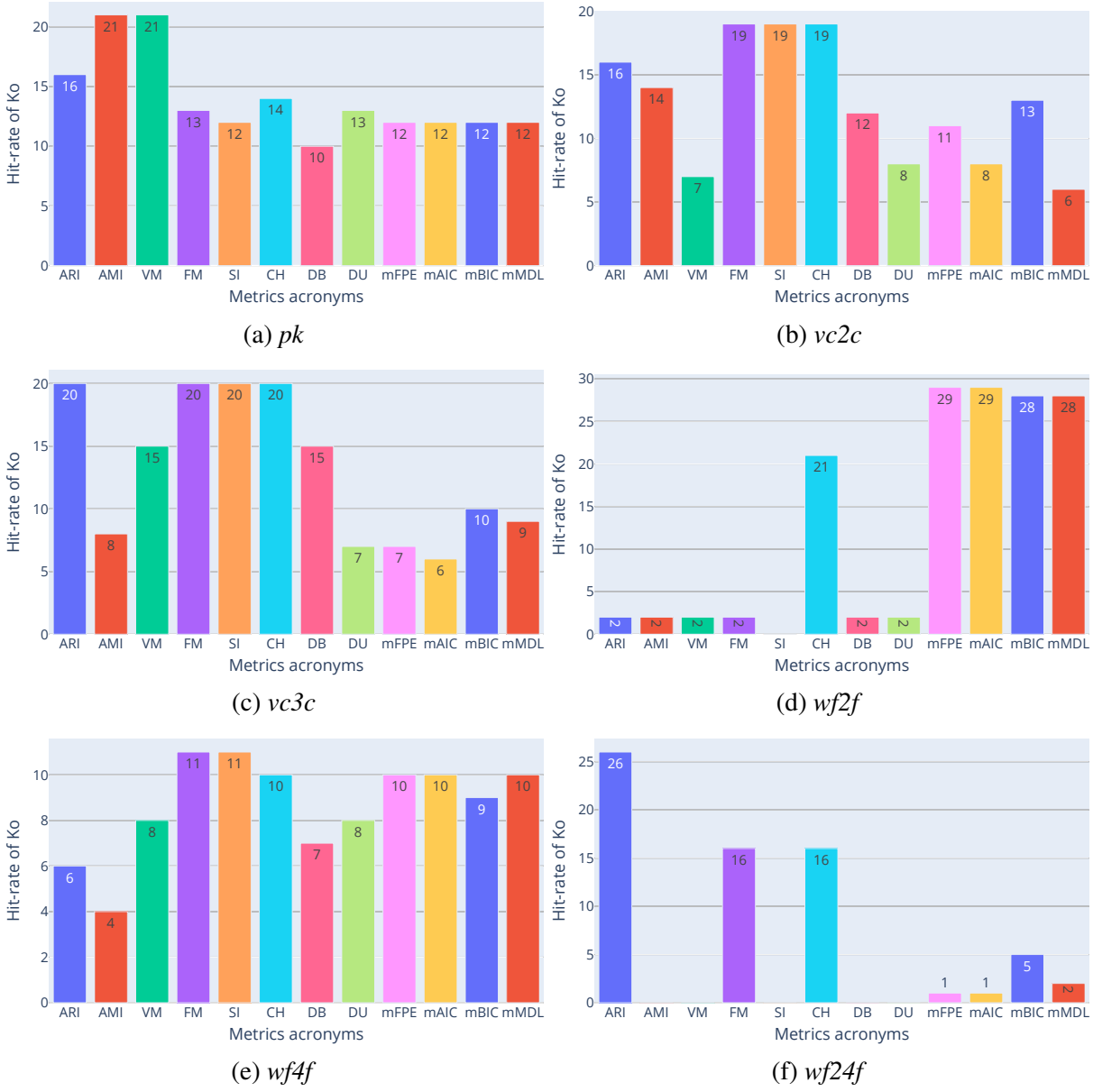
K_{opt}, that was verified in the cross-validation phase. A visual representation can be view in Figure 13. Overall no validation metric seems to be reliable in suggesting the K_{opt} for all the six data sets variations; especially in data sets *pk*, *vc2c*, *vc3c*, and *wf4f*. In the *wf2f* data set we see a prevalence of the information criteria-based metrics, this outcome is due to these metrics bias in suggesting high values of K_{opt} and as the reader can verify in Figure 12 the problem contemplated by *wf2f* seems to need a high quantity of partitions to be better modeled. In the same manner, the results of *wf24f* suggest that the adjusted Rand index was a good metric but this might have happened because of its tendency to suggest small values of K_{opt} (with Figure 12 we can see that this problem requires a fewer number of prototypes to be solved with CLHP). After this investigation, we may conclude that given the inconsistency of the validation metrics on finding K_{opt} before the cross-validation phase, the Figures 12 and 13 tell us more about how these metrics behave rather than its ability to predict K_{opt}.

5.3 Global vs Regional LSC-LBF results

In Figure 14 we can see the boxplot of the accuracy distributions on the train and test sets for the G/R-LSC performance comparison. From this chart, we are able to notice that the regional model has a consistent inclination to perform better than the global one on the training set, as a set of models specialized in regions of the input space should do. On *pk*, *vc2c*, and *vc3c* data sets we perceive similar accuracy distributions between the global and regional approach and on the wall-following data sets variations a substantial increase in performance. These results on *wf2f*, *wf4f*, and *wf24f* can be explained by the aptitude of the R-LSC on building a nonlinear decision boundary by doing a regional linearization, as the variations of wall-following data set appear to pose a problem of this nature.

With Figure 15 we can inspect the distribution of K_{opt} on the 100 independent runs of the experiment. Generally speaking, there was a tendency on using fewer partitions, the exception being the *wf24f* data set that had a mode of 19 prototypes. As on this experiment, the K_{opt} was determined by the Davies-Bouldin index (see Section 4.5.2.3) the charts of Figure 15 tell us more about the index behavior than its ability to choose a proper value for the number of regions. Given the dispersion seen on K_{opt} value, we can conclude that the DB index is sensitive to the train/test split process and/or to slight changes on prototypes locations, becoming an unreliable validation metric to choosing a proper value of K.

Figure 13 – Metrics K_{opt} hit-rate in L-LSSVM simulation.



(a) *pk* (b) *vc2c*

(c) *vc3c* (d) *wf2f*

(e) *wf4f* (f) *wf24f*

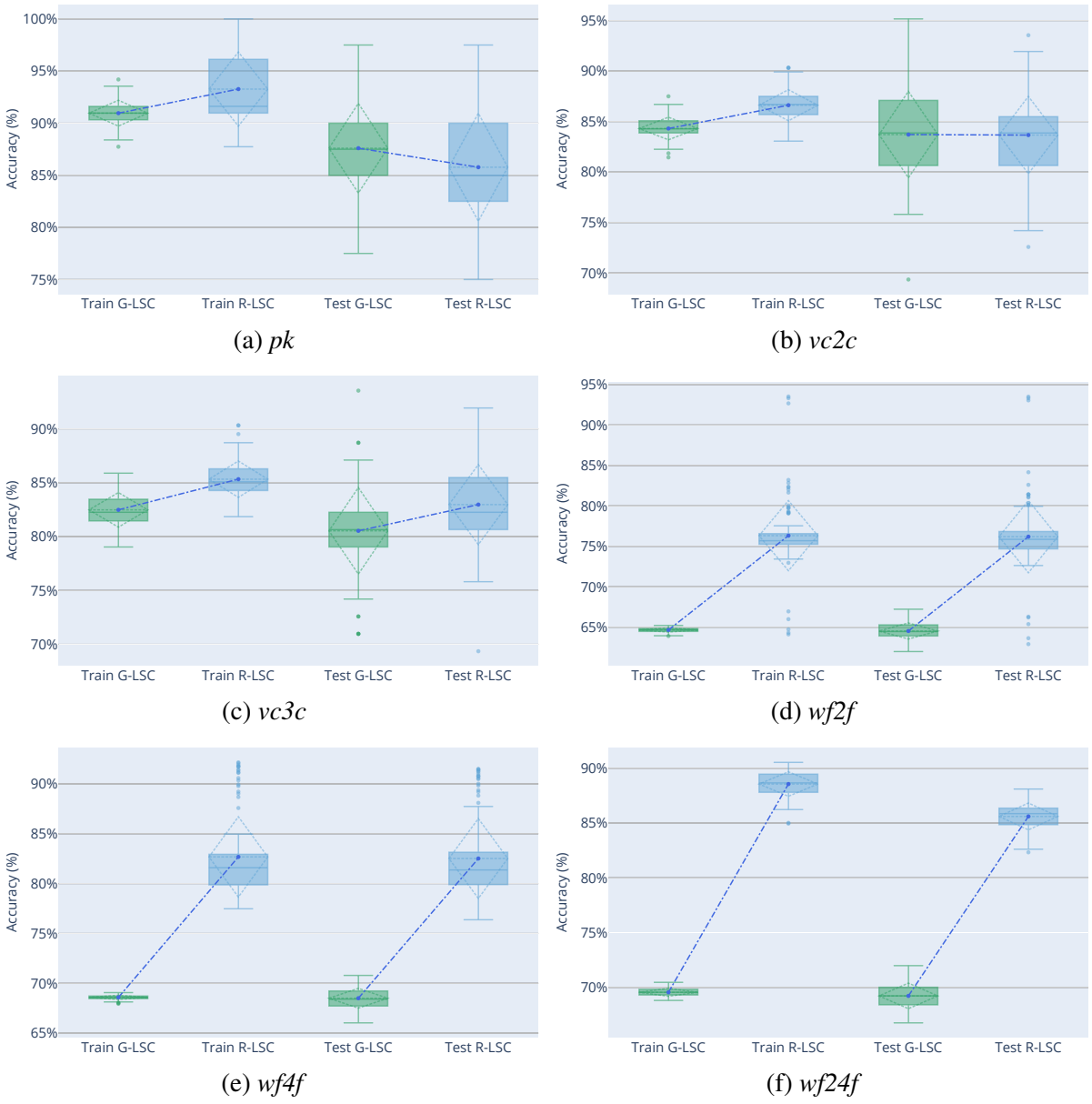
Source: the author (2019).

5.4 Global vs Local vs Regional LSSVM results

In Figure 16 we can see the train and test accuracy distributions for the three modeling approaches addressed in this thesis. The regional classifier showed similar results to CLHP, being superior to G-LSSVM in the *wf2f* and equivalent in the rest of the data sets. To serve as a guide a dash-dot line was made connecting the mean values of accuracy in the train and test set of the models.

On Tables 11-16 we find more precise numerical results with the four performance metrics used. Similar to L-LSSVM the R-LSSVM showed superior specificity in *pk* data set (see

Figure 14 – Train and test accuracy distributions in G/R-LSC performance comparison.



(a) *pk* (b) *vc2c*

(c) *vc3c* (d) *wf2f*

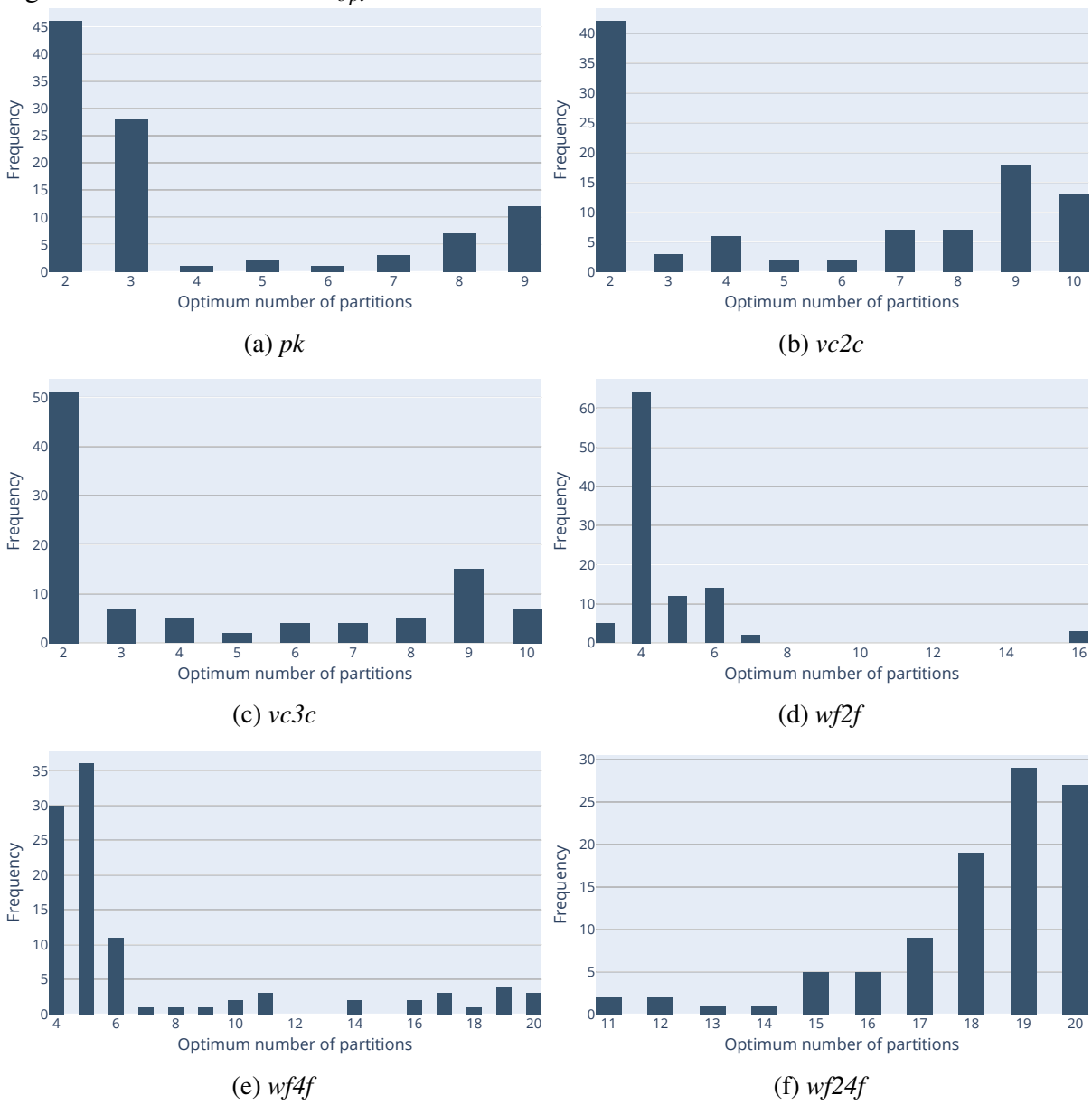
(e) *wf4f* (f) *wf24f*

Source: the author (2019).

Table 11) and overall better performance in *wf2f* (see Table 14), while been equivalent on the other data sets results.

Figure 17 shows the distribution of K_{opt} in the 50 independent runs of the simulations with L/R-LSSVM. This graph exposes the propensity of the regional classifier for requiring fewer prototypes to achieve an equivalent performance to CLHP, with the biggest difference being on the *wf2f* data set where the mode of L-LSSVM lies about 50 partitions while the R-LSSVM lies about 15. This is a consequence of the clustering of the SOM, as we cluster a simplified representation instead of the whole data set the search space of K is also reduced, being for

Figure 15 – Distribution of K_{opt} in R-LSC.



(a) *pk* (b) *vc2c*

(c) *vc3c* (d) *wf2f*

(e) *wf4f* (f) *wf24f*

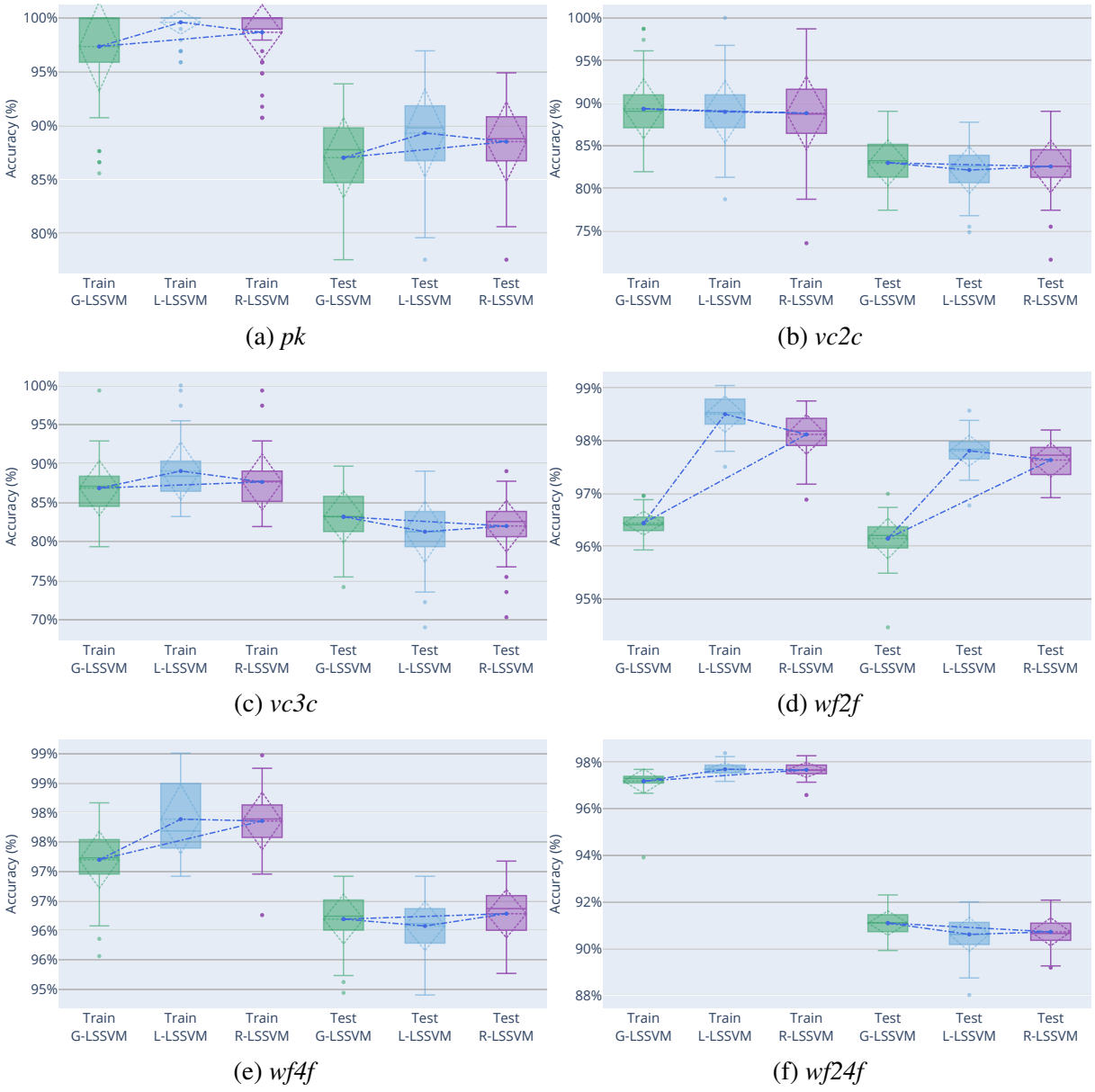
Source: the author (2019).

Table 11 – Performance in *pk* of G/L/R-LSSVM experiment

Set	Model	Accuracy (%)	Sens. (%)	Spec. (%)	F_1 (%)
Train	G-LSSVM	97.34 ± 4.16	99.67	90.25	98.32
	L-LSSVM	99.59 ± 1.09	99.92	98.58	99.73
	R-LSSVM	98.66 ± 2.56	99.64	95.67	99.13
Test	G-LSSVM	87.02 ± 3.73	95.49	60.92	91.72
	L-LSSVM	89.31 ± 4.08	94.30	73.92	93.00
	R-LSSVM	88.51 ± 3.71	94.32	70.58	92.50

Source: the author (2019).

Figure 16 – Train and test accuracy distributions in G/L/R-LSSVM experiment.



(a) *pk* (b) *vc2c*

(c) *vc3c* (d) *wf2f*

(e) *wf4f* (f) *wf24f*

Source: the author (2019).

Table 12 – Performance in *vc2c* of G/L/R-LSSVM experiment

Set	Model	Accuracy (%)	Sens. (%)	Spec. (%)	F_1 (%)
Train	G-LSSVM	89.32 ± 3.51	81.04	93.26	82.88
	L-LSSVM	88.97 ± 3.65	79.28	93.58	81.98
	R-LSSVM	88.81 ± 4.33	79.36	93.31	81.78
Test	G-LSSVM	82.98 ± 2.69	70.52	88.91	72.53
	L-LSSVM	82.14 ± 2.77	69.68	88.08	71.33
	R-LSSVM	82.55 ± 3.08	70.12	88.48	71.90

Source: the author (2019).

Table 13 – Performance in *vc3c* of G/L/R-LSSVM experiment

Set	Model	Accuracy (%)	Sens. (%)	Spec. (%)	F_1 (%)
Train	G-LSSVM	86.86 ± 3.53	93.21	82.45	93.24
	L-LSSVM	89.05 ± 3.71	94.36	85.59	94.47
	R-LSSVM	87.65 ± 3.59	93.54	83.85	93.65
Test	G-LSSVM	83.19 ± 3.33	91.00	78.52	90.95
	L-LSSVM	81.28 ± 3.92	89.82	76.75	89.81
	R-LSSVM	82.00 ± 3.31	90.21	77.24	90.23

Source: the author (2019).

Table 14 – Performance in *wf2f* of G/L/R-LSSVM experiment

Set	Model	Accuracy (%)	Sens. (%)	Spec. (%)	F_1 (%)
Train	G-LSSVM	96.44 ± 0.22	98.68	95.76	98.65
	L-LSSVM	98.50 ± 0.34	99.45	98.50	99.44
	R-LSSVM	98.12 ± 0.38	99.31	98.13	99.29
Test	G-LSSVM	96.15 ± 0.39	98.56	95.06	98.54
	L-LSSVM	97.81 ± 0.29	99.19	97.32	99.19
	R-LSSVM	97.63 ± 0.32	99.12	97.27	99.12

Source: the author (2019).

Table 15 – Performance in *wf4f* of G/L/R-LSSVM experiment

Set	Model	Accuracy (%)	Sens. (%)	Spec. (%)	F_1 (%)
Train	G-LSSVM	97.20 ± 0.48	98.94	97.27	98.92
	L-LSSVM	97.89 ± 0.59	99.20	97.98	99.19
	R-LSSVM	97.86 ± 0.49	99.19	98.01	99.18
Test	G-LSSVM	96.19 ± 0.42	98.54	95.64	98.53
	L-LSSVM	96.08 ± 0.42	98.49	95.48	98.49
	R-LSSVM	96.28 ± 0.41	98.57	95.75	98.57

Source: the author (2019).

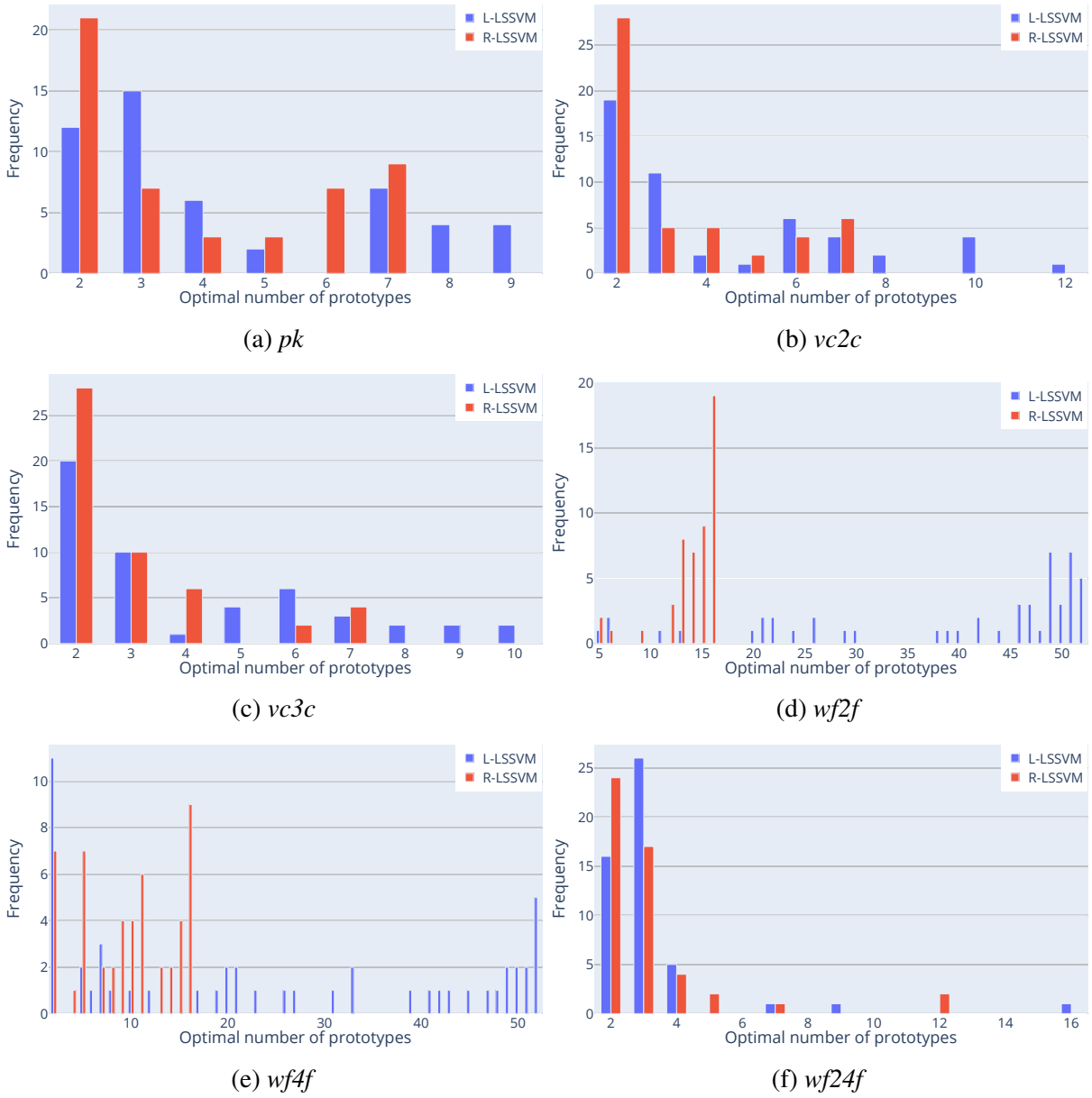
Table 16 – Performance in *wf24f* of G/L/R-LSSVM experiment

Set	Model	Accuracy (%)	Sens. (%)	Spec. (%)	F_1 (%)
Train	G-LSSVM	97.18 ± 0.52	98.89	96.78	98.92
	L-LSSVM	97.69 ± 0.27	99.09	97.35	99.12
	R-LSSVM	97.67 ± 0.32	99.08	97.33	99.11
Test	G-LSSVM	91.10 ± 0.52	96.42	89.84	96.47
	L-LSSVM	90.61 ± 0.73	96.23	89.50	96.27
	R-LSSVM	90.72 ± 0.60	96.28	89.60	96.32

Source: the author (2019).

the CLHP the set $SS_{CLHP} = \{2,3,...,\lfloor\sqrt{N_{tr}}\rfloor\}$ and for the RC $SS_{RC} = \{2,3,...,\lfloor\sqrt{5\sqrt{N_{tr}}}\rfloor\}$, as $N_{SOM} = 5\sqrt{N_{tr}}$.

Figure 17 – Optimal number of prototypes in L/R-LSSVM.



(a) *pk* (b) *vc2c*

(c) *vc3c* (d) *wf2f*

(e) *wf4f* (f) *wf24f*

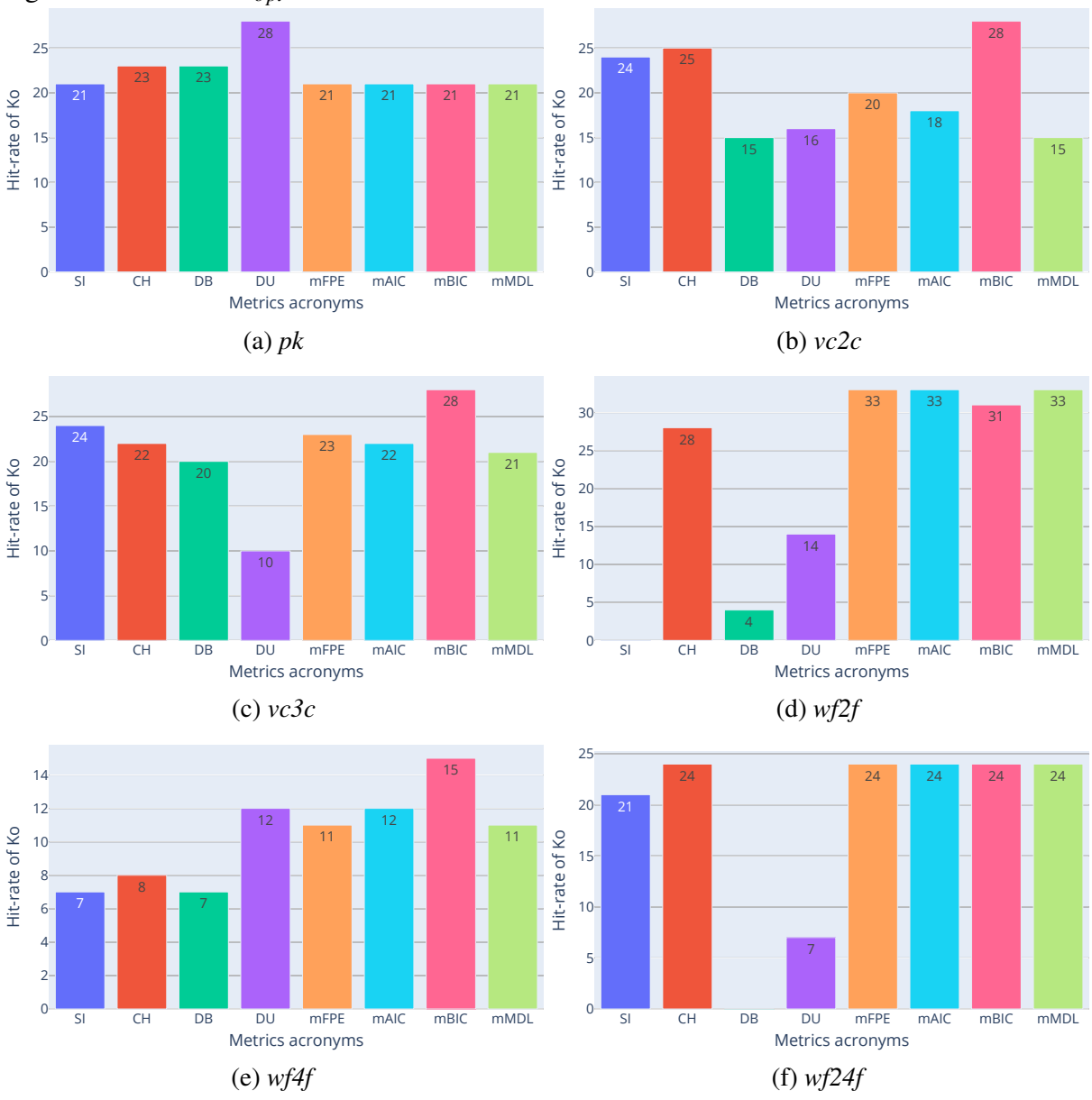
Source: the author (2019).

In Figure 18 we can see the metrics hit-rate of K_{opt} in the R-LSSVM. Unlike the L-LSSVM instance, the regional approach had only 8 validation metrics in the committee as 4 were supervised metrics (since we clustered the SOM units and they were unlabeled).

Analogous to the CLHP case, no validation metric seems to be reliable in predicting the K_{opt}. This time we can also see that the hit-rates were higher as the search space was smaller.

As regional modeling relies on fewer partitions we can also recognize, with Figure 19, that the R-LSSVM presented a trend to build fewer homogeneous regions (regions containing only points of a single class). This property points out that RC relies more on the base model (the

Figure 18 – Metrics K_{opt} hit-rate in R-LSSVM.



(a) *pk* (b) *vc2c*

(c) *vc3c* (d) *wf2f*

(e) *wf4f* (f) *wf24f*

Source: the author (2019).

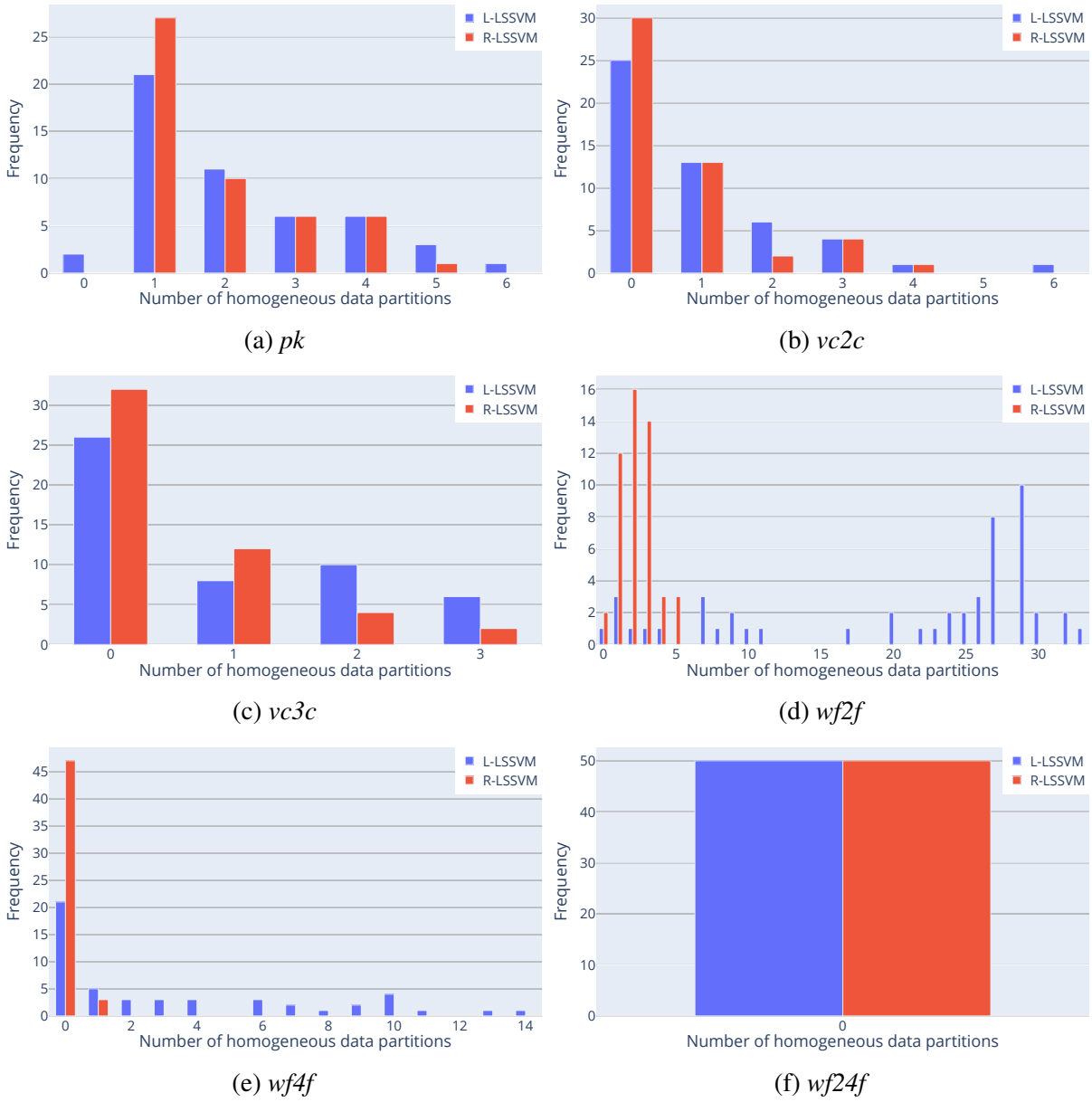
LSSVM in the R-LSSVM) than the CLHP approach, meaning that the local approach presents a trend of having bias models into its set of models.

5.5 Concluding remarks

In this chapter, we presented an comprehensive set of results of the simulations proposed in Chapter 4. Some important concluding remarks are given next.

- Both CLHP and RC were able to approximate nonlinear decision boundaries by means of a set of local/regional models;

Figure 19 – Number of homogeneous data partitions in L/R-LSSVM.



(a) *pk* (b) *vc2c*

(c) *vc3c* (d) *wf2f*

(e) *wf4f* (f) *wf24f*

Source: the author (2019).

- There are classification problems that CLHP and RC with a nonlinear base model performs better than a global nonlinear one (e.g. *wf2f* data set case);
- RC have similar performance to CLHP ones but using fewer prototypes, consequently showing fewer homogenous data partitions;
- Any of the 12 validation metrics used seems to be a reliable choice to predict the K_{opt} value, being more prudent to optimize K as other hyperparameters (e.g. with grid search and random search).

The next chapter contains the conclusion of this work as future works proposals.

6 CONCLUSION AND FUTURE WORKS

Throughout the course of this thesis, the author was able to develop *python* versions of the LSSVM classifier (DRUMOND, 2019c), of the CLHP approach (DRUMOND, 2019b), and propose the first implementation of regional classifiers (DRUMOND, 2019a). The simplicity of the LSSVM optimization problem was noted during code development as the fitness of such classifier when applied to nonlinear data sets, e.g. wall-following variations. During the development of CLHP and regional approaches, unpredicted phenomenons occur (see sections 2.1.3 and 3.3.1) that required the author to revisit and update the theory of such paradigms.

It became clear the capability of local modeling approaches to build nonlinear decision boundaries with a set of linear classifiers. In nonlinear problems as the one presented by the wall-following data set, the use of a set of linear models in the local and regional approaches presented a large performance improvement. Even in the realm of nonlinear classifiers, the local methods seem to outperform global ones in some use cases. With similar results to CLHP, the regional approach poses itself as a sparser modeling paradigm, using fewer prototypes/models.

As future works, one could develop any one of the following ideas.

- Development of specific cluster validation metrics for CLHP and regional paradigms since the traditional ones seems to be unsuitable for choosing the best K;
- Apply other classification models and compare their performances with CLHP and regional approaches;
- Use other clustering algorithms both in CLHP and regional models and investigate their properties;
- Extend the local and regional approaches for an online/growing/adaptive case, where the partitions/regions are build in an online fashion.

REFERENCES

AKAIKE, H. Fitting autoregressive models for prediction. **Annals of the institute of Statistical Mathematics**, Springer, v. 21, n. 1, p. 243–247, 1969.

AKAIKE, H. A new look at the statistical model identification. **IEEE transactions on automatic control**, Ieee, v. 19, n. 6, p. 716–723, 1974.

ALPAYDIN, E.; JORDAN, M. I. Local linear perceptrons for classification. **IEEE Transactions on Neural Networks**, IEEE, v. 7, n. 3, p. 788–794, 1996.

BARRETO, G. A.; R., N. A. R.; F., M. H. A. **Vertebral Column Data Set**. 2011. Disponível em: <http://archive.ics.uci.edu/ml/datasets/vertebral+column>. Acesso em: 28 jul. 2020.

BEZDEK, J. C.; PAL, N. R. Some new indexes of cluster validity. **IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)**, IEEE, v. 28, n. 3, p. 301–315, 1998.

BISCHL, B.; SCHIFFNER, J.; WEIHS, C. Benchmarking local classification methods. **Computational Statistics**, Springer, v. 28, n. 6, p. 2599–2619, 2013.

BOTTOU, L.; VAPNIK, V. Local learning algorithms. **Neural computation**, MIT Press, v. 4, n. 6, p. 888–900, 1992.

BOYD, S.; VANDENBERGHE, L. **Convex optimization**. [S.l.]: Cambridge university press, 2004.

BOYD, S.; VANDENBERGHE, L. **Introduction to applied linear algebra: vectors, matrices, and least squares**. [S.l.]: Cambridge university press, 2018.

CALIŃSKI, T.; HARABASZ, J. A dendrite method for cluster analysis. **Communications in Statistics-theory and Methods**, Taylor & Francis, v. 3, n. 1, p. 1–27, 1974.

DAVIES, D. L.; BOULDIN, D. W. A cluster separation measure. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, n. 2, p. 224–227, 1979.

DRUMOND, R. B. P. **Classification by regional modeling.** 2019. Disponível em: <https://github.com/RomuloDrumond/Classification-by-regional-modeling>.

DRUMOND, R. B. P. **A python implementation of the cluster-based local modeling with hard partitioning.** 2019. Disponível em: <https://github.com/RomuloDrumond/Classification-by-local-modeling>.

DRUMOND, R. B. P. **A python implementation of the Least Squares Support Vector Machine for classification on CPU (NumPy) and GPU (PyTorch).** 2019. Disponível em: <https://github.com/RomuloDrumond/LSSVM>.

DUA, D.; GRAFF, C. **UCI Machine Learning Repository**. 2017.

DUNN, J. C. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. Taylor & Francis, 1973.

DUTTA, S. An overview on the evolution and adoption of deep learning applications used in the industry. **Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery**, Wiley Online Library, v. 8, n. 4, p. e1257, 2018.

FOWLKES, E. B.; MALLOWS, C. L. A method for comparing two hierarchical clusterings. **Journal of the American statistical association**, Taylor & Francis Group, v. 78, n. 383, p. 553–569, 1983.

FREIRE, A.; VELOSO, M.; BARRETO, G. **Wall-Following Robot Navigation Data Data Set**. 2010. Disponível em: <https://archive.ics.uci.edu/ml/datasets/Wall-Following+Robot+Navigation+Data>. Acesso em: 28 jul. 2020.

FREIRE, A. L. **A DIMENSÃO TEMPORAL NO PROJETO DE CLASSIFICADORES DE PADRÕES PARA NAVEGAÇÃO DE ROBÔS MÓVEIS: UM ESTUDO DE CASO**. Dissertação (Mestrado) — Universidade Federal Do Ceará, Fortaleza, 2009.

HAYKIN, S. **Neural networks and learning machines**. 3rd. ed. [S.l.]: Prentice Hall, 2009.

HOFMANN, T.; SCHÖLKOPF, B.; SMOLA, A. J. Kernel methods in machine learning. **The annals of statistics**, JSTOR, p. 1171–1220, 2008.

HUBERT, L.; ARABIE, P. Comparing partitions. **Journal of classification**, Springer, v. 2, n. 1, p. 193–218, 1985.

JACOBS, R. A.; JORDAN, M. I.; NOWLAN, S. J.; HINTON, G. E. *et al.* Adaptive mixtures of local experts. **Neural computation**, v. 3, n. 1, p. 79–87, 1991.

KANUNGO, T.; MOUNT, D. M.; NETANYAHU, N. S.; PIATKO, C. D.; SILVERMAN, R.; WU, A. Y. An efficient k-means clustering algorithm: analysis and implementation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 24, n. 7, p. 881–892, 2002.

KOHONEN, T. Self-organized formation of topologically correct feature maps. **Biological cybernetics**, Springer, v. 43, n. 1, p. 59–69, 1982.

KONEČNỲ, J.; MCMAHAN, H. B.; RAMAGE, D.; RICHTÁRIK, P. Federated optimization: Distributed machine learning for on-device intelligence. **arXiv preprint arXiv:1610.02527**, 2016.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **nature**, Nature Publishing Group, v. 521, n. 7553, p. 436–444, 2015.

LEE, B.; CHOI, S.; OH, B.; YANG, J.; PARK, S. A new ensemble learning algorithm using regional classifiers. **International Journal on Artificial Intelligence Tools**, World Scientific, v. 22, n. 04, p. 1350025, 2013.

LITTLE, M. **Parkinsons Data Set**. 2008. Disponível em: <https://archive.ics.uci.edu/ml/datasets/parkinsons>. Acesso em: 28 jul. 2020.

MANSANET, J.; ALBIOL, A.; PAREDES, R. Local deep neural networks for gender recognition. **Pattern Recognition Letters**, Elsevier, v. 70, p. 80–86, 2016.

MARTINETZ, T. M.; BERKOVICH, S. G.; SCHULTEN, K. J. 'neural-gas' network for vector quantization and its application to time-series prediction. **IEEE transactions on neural networks**, IEEE, v. 4, n. 4, p. 558–569, 1993.

NETO, A. R. D. R. **SINPATCO II: NOVAS ESTRATÉGIAS DE APRENDIZADO DE MÁQUINA PARA CLASSIFICAÇÃO DE PATOLOGIAS DA COLUNA VERTEBRAL**. Tese (Doutorado) — UNIVERSIDADE FEDERAL DO CEARÁ, 2011.

RISSANEN, J. Modeling by shortest data description. **Automatica**, Elsevier, v. 14, n. 5, p. 465–471, 1978.

Rocha Neto, A. R. Máquinas de vetores-suporte: uma revisão. **Learning & Nonlinear Models**, ABRICOM, v. 15, n. 1, p. 16–4, 2017.

ROSENBERG, A.; HIRSCHBERG, J. V-measure: A conditional entropy-based external cluster evaluation measure. In: **Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)**. [S.l.: s.n.], 2007. p. 410–420.

ROUSSEEUW, P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. **Journal of computational and applied mathematics**, Elsevier, v. 20, p. 53–65, 1987.

SCHWARZ, G. *et al.* Estimating the dimension of a model. **The annals of statistics**, Institute of Mathematical Statistics, v. 6, n. 2, p. 461–464, 1978.

SOUSA, D. P. **DETECÇÃO DE FALHAS DE CURTO-CIRCUITO EM MOTORES DE INDUÇÃO TRIFÁSICOS USANDO CLASSIFICADORES BASEADOS EM PROTÓTIPOS**. Dissertação (Mestrado) — Universidade Federal Do Ceará, Fortaleza, 2019.

SOUZA, C. R. de. **Kernel Functions for Machine Learning Applications.** 2019. Disponível em: <http://crsouza.com/2010/03/17/kernel-functions-for-machine-learning-applications/>.

Souza Jr., A. H. S.; BARRETO, G. A.; CORONA, F. Regional models: A new approach for nonlinear system identification via clustering of the self-organizing map. **Neurocomputing**, Elsevier, v. 147, p. 31–46, 2015.

SPREEUWERS, L. J.; VELDHUIS, R.; SULTANALI, S.; DIEPHUIS, J. Fixed far vote fusion of regional facial classifiers. In: IEEE. **2014 International Conference of the Biometrics Special Interest Group (BIOSIG)**. [S.l.], 2014. p. 1–4.

SUYKENS, J. A.; VANDEWALLE, J. Least squares support vector machine classifiers. **Neural processing letters**, Springer, v. 9, n. 3, p. 293–300, 1999.

VESANTO, J.; ALHONIEMI, E. *et al.* Clustering of the self-organizing map. **IEEE Transactions on neural networks**, Citeseer, v. 11, n. 3, p. 586–600, 2000.

VINH, N. X.; EPPS, J.; BAILEY, J. Information theoretic measures for clusterings comparison: Is a correction for chance necessary? In: **Proceedings of the 26th Annual International Conference on Machine Learning**. New York, NY, USA: ACM, 2009. (ICML '09), p. 1073–1080. ISBN 978-1-60558-516-1. Disponível em: <http://doi.acm.org/10.1145/1553374.1553511>.

VINH, N. X.; EPPS, J.; BAILEY, J. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. **Journal of Machine Learning Research**, v. 11, n. Oct, p. 2837–2854, 2010.

VOULODIMOS, A.; DOULAMIS, N.; DOULAMIS, A.; PROTOPAPADAKIS, E. Deep learning for computer vision: A brief review. **Computational intelligence and neuroscience**, Hindawi, v. 2018, 2018.

WANG, X.; SYRMOS, V. L. Nonlinear system identification and fault detection using hierarchical clustering analysis and local linear models. In: IEEE. **2007 Mediterranean Conference on Control & Automation**. [S.l.], 2007. p. 1–6.

ZHANG, J.; YU, J.; TAO, D. Local deep-feature alignment for unsupervised dimension reduction. **IEEE transactions on image processing**, IEEE, v. 27, n. 5, p. 2420–2432, 2018.