UJAT
Universidad Juárez Autónoma de Tabasco
"Estudio en la duda. Acción en la fe"

DAIS 11111000011
División Académica de Informática y Sistemas

Universidad Juárez Autónoma de Tabasco

Tesis Doctoral

Descriptive and predictive models for Guillain-Barre syndrome based on clinical data using machine learning algorithms

Que presenta

José Hernández Torruco

Para obtener el grado de

Doctor en Ciencias de la Computación

Directores

Dra. Juana Canul Reich
Dr. Juan Frausto Solís

Cunduacán, Tabasco, México **Febrero 2016**

UJAT
Universidad Juárez Autónoma de Tabasco
"Estudio en la duda. Acción en la fe"

DAIS 11111000011
División Académica de Informática y Sistemas

Universidad Juárez Autónoma de Tabasco

Tesis Doctoral

Descriptive and predictive models for Guillain-Barre syndrome based on clinical data using machine learning algorithms

Que presenta

José Hernández Torruco

Para obtener el grado de

Doctor en Ciencias de la Computación

Comité Tutoral: **Dr. Alejandro Padilla Díaz**
Dr. Juan Frausto Solís
Dra. Juana Canul Reich

Jurado:	**Dr. Alejandro Padilla Díaz**	Presidente
	Dr. Carlos Alberto Ochoa Ortíz	Secretario
	Dr. Juan Paulo Sánchez Hernández	Vocal
	Dra. Juana Canul Reich	Vocal
	Dr. Juan Frausto Solís	Vocal

Cunduacán, Tabasco, México **Febrero 2016**

Universidad Autónoma de Aguascalientes

Universidad Juárez Autónoma de Tabasco
"Estudio en la duda. Acción en la fe"

Universidad Veracruzana

UNIVERSIDAD JUÁREZ AUTÓNOMA DE TABASCO

"ESTUDIO EN LA DUDA. ACCIÓN EN LA FE"

DIVISIÓN ACADÉMICA DE INFORMÁTICA Y SISTEMAS

Cunduacán, Tab. Enero 11 de 2016.

En la Universidad Juárez Autónoma de Tabasco, de acuerdo al Reglamento de Estudios de Posgrado vigente, se revisó el trabajo de investigación titulado "**Descriptive and predictive models of Guillain-Barré Syndrome based on clinical data using machine learning algorithms**", realizado por el C. **José Hernández Torruco**, estudiante del Doctorado Interinstitucional en Ciencias de la Computación para obtener el Grado de Doctor en Ciencias de la Computación bajo la modalidad de Tesis.

Los integrantes del jurado, después de revisar el trabajo, y en virtud de que se han atendido satisfactoriamente las observaciones y recomendaciones, otorgamos nuestra aprobación para que continúe los trámites correspondientes a la obtención del grado.

Dra. Juana Canul Reich
Profesora-Investigadora

Dr. Juan Frausto Solís
Profesor-Investigador

Dr. Alejandro Padilla Diaz
Profesor-Investigador

Dr. Carlos Alberto Ochoa Ortiz
Profesor-Investigador

Dr. Juan Paulo Sánchez Hernández
Profesor-Investigador

Carretera Cunduacán-Jalpa Km. 1, Colonia Esmeralda, C.P. 86690. Cunduacán, Tabasco, México
E-mail: direccion.dais@uat.mx
Teléfonos: (993) 358 1500 ext. 6727; (914) 336 0616. Fax: (914) 336 0870

UNIVERSIDAD JUÁREZ AUTÓNOMA DE TABASCO
"ESTUDIO EN LA DUDA. ACCIÓN EN LA FE"

DAIS 11111000011

DIVISIÓN ACADÉMICA DE INFORMÁTICA Y SISTEMAS

Oficio No.562/2016/DAIS-D
10 de febrero de 2016.

MIS. José Hernández Torruco
Estudiante del DICC
Cunduacán, Tabasco.

Por este medio me permito hacer de su conocimiento que con base el acta de evaluación de examen predoctoral, acta de examen de grado y evaluaciones del comité tutoral, se determina la Validación de las Figuras Académicas en sus distintas responsabilidades, Director, Co-director y Jurado de examen de grado quedando de la siguiente manera:

Directores:

Dra. Juana Canul Reich
Dr. Juan Frausto Solís

Comité Tutoral:
Dr. Alejandro Padilla Díaz
Dra. Juana Canul Reich
Dr. Juan Frausto Solís

Jurado:

Dr. Alejandro Padilla Díaz	Presidente
Dr. Carlos Alberto Ochoa Ortiz	Secretario
Dr. Juan Paulo Sánchez Hernández.	Vocal
Dra. Juana Canul Reich	Vocal
Dr. Juan Frausto Solís	Vocal

Sin otro particular reciba usted un cordial saludo.

A t e n t a m e n t e

MATI. Eduardo Cruces Gutiérrez
Director

C.c.p. Dr. Jesús Hernández del Real. Responsable del Área de Posgrado.
Archivo.
Consecutivo.

Carretera Cunduacán-Jalpa Km. 1, Colonia Esmeralda, C.P. 86690, Cunduacán, Tabasco, México.
E-mail: direccion.dais@ujat.mx
Teléfonos: (993) 358 1500 ext. 6727; (914) 336 0616; Fax: (914) 336 0870

UNIVERSIDAD JUÁREZ AUTÓNOMA DE TABASCO
"ESTUDIO EN LA DUDA. ACCIÓN EN LA FE"

DAIS 11111000011

DIVISIÓN ACADÉMICA DE INFORMÁTICA Y SISTEMAS

Oficio No.1089/2016/DAIS-D
08 de febrero de 2016.

MIS. José Hernández Torruco
Estudiante del DICC
Cunduacán, Tabasco.

Por este medio me permito hacer de su conocimiento que con base en el Dictamen del Consejo Académico del Doctorado Interinstitucional en Ciencias de la Computación (DICC), dirigido por el Dr. Francisco Diego Acosta Escalante en su carácter de Secretario Técnico del Consejo Académico, se determina en Anexo 1, que las publicaciones sometidas a revisión cumplen con los requisitos de obtención del grado previstos en el programa.

Sin otro particular reciba usted un cordial saludo.

A t e n t a m e n t e

MATI. Eduardo Cruces Gutiérrez
Director

C.c.p. Dr. Francisco Diego Acosta Escalante.- Secretario Técnico del DICC
Dr. Jesús Hernández del Real. Responsable del Área de Posgrado.
Archivo.

ANEXO 1

Producto 1

Predictores de falla respiratoria y de la necesidad de ventilación mecánica en el Síndrome de Guillain-Barré

Evidencias del arbitraje

http://revmexneuroci.com/acerca-de-rev-mex-neuroci/

Evidencias del índice

http://revmexneuroci.com/acerca-de-rev-mex-neuroci/
http://www.imbiomed.com.mx/1/1/articulos.php?method=showIndex&id_revista=91

Estado

Publicado en el número correspondiente al periodo septiembre – octubre de 2013. Vol. 14, Num. 5.

Relación con el trabajo doctoral

Es una revisión del estado del arte de los predictores de falla respiratoria y de la necesidad de ventilación mecánica en el síndrome estudiado.

Producto 2

El aprendizaje computacional en el diagnóstico de enfermedades

Evidencias del arbitraje

http://komputersapiens.smia.mx/index.php?option=com_content&view=article&id=67&Itemid=96

Evidencias del índice

http://komputersapiens.smia.mx/index.php?option=com_content&view=article&id=73&Itemid=93

Estado

Publicado en el número correspondiente al periodo septiembre – diciembre de 2014. Año VI, Vol. III.

Relación con el trabajo doctoral

Es una revisión del estado del arte de los métodos de aprendizaje computacional que se aplican en el diagnóstico de enfermedades.

Producto 3

Towards a predictive model for Guillain-Barré Syndrome

Evidencias del arbitraje
http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7318240

Evidencias del índice
http://www.ieee.org/conferences_events/conferences/conferencedetails/index.html?Conf_ID=1817

Estado
Publicado. La conferencia se realizó del 25 al 29 de agosto de 2015 en la ciudad de Milán, Italia.

Relación con el trabajo doctoral
Describe los resultados de experimentos exploratorios para la construcción de un modelo predictivo para el síndrome investigado, usando algoritmos de aprendizaje automático.

Producto 4
A kernel-based predictive model for Guillain-Barré Syndrome.

Evidencias del arbitraje
http://link.springer.com/chapter/10.1007/978-3-319-27101-9_20

Evidencias del índice

Estado
Publicado. Proceedings of the 14th Mexican International Conference on Artificial Intelligence, MICAI 2015, held in Cuernavaca, Proceedings, Part II. Morelos, Mexico, in October 2015. La conferencia se realizó del 25 al 31 de octubre de 2015 en la ciudad de Cuernavaca, Morelos.

Relación con el trabajo doctoral
Describe los resultados de experimentos para la construcción de un modelo predictivo para el síndrome investigado, usando máquinas de vectores soporte (SVM) y el clasificador C4.5.

Producto 5
Feature selection for better identification of subtypes of Guillain-Barré Syndrome

Evidencias del arbitraje
http://www.hindawi.com/journals/cmmm/reviewers/

Evidencias del índice
http://ip-science.thomsonreuters.com/cgi-bin/jrnlst/jlresults.cgi?PC=MASTER&ISSN=1748-670X

Estado
Publicado el 15 de septiembre de 2014.

Relación con el trabajo doctoral
Describe los resultados de experimentos para obtener los atributos relevantes para identificar los cuatro subtipos principales del síndrome estudiado, usando métodos filtro.

Producto 6

Finding relevant features for identifying subtypes of Guillain-Barré Syndrome using Quenching Simulated Annealing and Partitions Around Medoids

Evidencias del arbitraje
http://ijcopi.org/ojs/index.php?journal=ijcopi&page=about&op=editorialTeam

Evidencias del índice
http://ijcopi.org/ojs/index.php?journal=ijcopi&page=about&op=editorialPolicies#custom0

Estado
Publicado en el número correspondiente al periodo mayo – agosto de 2015. Vol. 6, Num. 2.

Relación con el trabajo doctoral
Describe los resultados de experimentos para obtener los atributos relevantes para identificar los cuatro subtipos principales del síndrome estudiado, usando un método que combina un algoritmo de optimización y un algoritmo de clustering, denominado QSA-PAM.

UNIVERSIDAD JUÁREZ AUTÓNOMA DE TABASCO
"ESTUDIO EN LA DUDA. ACCIÓN EN LA FE"

DAIS 11111000011

DIVISIÓN ACADÉMICA DE INFORMÁTICA Y SISTEMAS

Oficio Núm.351/2016-DAIS-D
09 de Febrero de 2016

MIS. José Hernández Torruco

En virtud de que cumple satisfactoriamente los requisitos establecidos en el Reglamento de General Estudio de Posgrado vigente en la Universidad, informo a Usted que se autoriza la impresión del trabajo recepcional **"Descriptive and predictive models for Guillain-Barre syndrome based on clinical data using machine learning algorithms"**, para presentar Examen Grado y obtener el Grado de Doctor en Ciencias de la Computación bajo la modalidad de Tesis.

Sin otro particular aprovecho la oportunidad para saludarle.

Atentamente

M.A.T.I. Eduardo Cruces Gutiérrez
Director

C.c.p. Director de Tesis 1: Dra. Juana Canul Reich.
C.c.p. Director de Tesis 2: Dr. Juan Frausto Solís.

Archivo
Consecutivo

M.A.T.I.*ECG/LC*CGZ

Carretera Cunduacán-Jalpa Km. 1, Colonia Esmeralda, C.P. 86690. Cunduacán, Tabasco, México.
E-mail: direccion.dais@ujat.mx
Teléfonos: (993) 358 1500 ext. 6727; (914) 336 0616; Fax: (914) 336 0870

iThenticate Página 13 de 168 - Engrega de integridad Identificador de la entrega trn:oid:::21:202706933

[redacted]

Cesión de Derechos

Cunduacán, Tabasco, a 8 de febrero de 2016

A quien corresponda:

Los abajo firmantes, declaramos que el trabajo de tesis doctoral titulado, **"Descriptive and predictive models for Guillain-Barre syndrome based on clinical data using machine learning algorithms"** es de nuestra autoría intelectual y por lo tanto cedemos los derechos de comunicación pública, reproducción, distribución, difusión en general y puesta a disposición electrónica de la citada tesis doctoral, de forma gratuita y no exclusiva, a la Universidad Juárez Autónoma de Tabasco, a la cual relevamos de cualquier sanción y asumimos responder a cualquier reclamo de derechos de autor ante las autoridades competentes.

Atentamente

Autores:

José Hernández Torruco	Pedro Gutiérrez Cortés 406 col. Gaviotas Norte C.P. 86090, Villahermosa, Centro, Tabasco
Dra. Juana Canul Reich	Revolución 361-1 col. Atasta C.P. 86100, Villahermosa, Centro, Tabasco
Dr. Juan Frausto Solís	Acacias 306 C Fracc. Lomas de Cuernavaca C.P. 62584, Temixco, Morelos

Carta de Autorización

A quien corresponda:

El que suscribe, autoriza por medio del presente escrito a la Universidad Juárez Autónoma de Tabasco para que utilice tanto física como digitalmente la tesis de grado denominada, **"Descriptive and predictive models for Guillain-Barre syndrome based on clinical data using machine learning algorithms"**, de la cual soy autor y titular de los Derechos de Autor.

La finalidad del uso por parte de la Universidad Juárez Autónoma de Tabasco de la tesis antes, mencionada, será única y exclusivamente para difusión, educación y sin fines de lucro; autorización que se hace de manera enunciativa más no limitativa para subirla a la Red Abierta de Bibliotecas Digitales (RABID) y a cualquier otra red académica con las que la Universidad tenga relación institucional.

Por lo antes manifestado, libero a la Universidad Juárez Autónoma de Tabasco de cualquier reclamación legal que pudiera ejercer respecto al uso y manipulación de la tesis mencionada y para los fines estipulados en éste documento.

Se firma la presente autorización en la ciudad de Villahermosa, Tabasco a los 9 días del mes de febrero del año 2016.

Autorizo

José Hernández Torruco

Abstract

Guillain-Barré Syndrome (GBS) is an autoimmune neuropathy of fast evolution and potentially fatal. The exact cause of GBS is unknown; however, it is frequently preceded either by a respiratory or a gastrointestinal infection. The diagnosis of GBS includes clinical, serological and electrophysiological criteria (nerve conduction or electrodiagnostic test) [67]. The severity of GBS varies among subtypes, which can be mainly Acute Inflammatory Demyelinating Polyneuropathy (AIDP), Acute Motor Axonal Neuropathy (AMAN), Acute Motor Sensory Axonal Neuropathy (AMSAN) and Miller-Fisher Syndrome [59].

A better understanding of the differences in the GBS subtypes is critical for the implementation of appropriate treatments for total recovery and, in certain cases for the survival of patients. Hospitalization time and the cost of treatments vary according to the severity of the specific subtype. Finding a minimum feature subset to accurately identify GBS subtypes could lead to a simplified and cheaper process of diagnosis and treatment of the GBS case. The ultimate goal of a physician is to get patients to a full recovery. This can be more effectively achieved when an early diagnosis of the case is performed using a minimum number of medical features.

There are many successful applications of machine learning in medicine, biology and genetics [82, 7, 10]. Also, the machine learning techniques have been applied successfully in the early diagnosis of diseases, including neurological disorders [5, 93]. Besides, the identification of a reduced number of relevant features to predict GBS subtypes could guide physicians to design a faster, simpler and cheaper diagnosis of the case.

This dissertation applies machine learning techniques to create two models for GBS, a descriptive model and a predictive model. The former aimed at identifying a minimum set of relevant features that builds four clusters, each corresponding to a specific GBS subtype. This model uses two approaches. In the first approach, the problem is tackled by using filter methods along with a clustering algorithm called Partitions Around Medoids (PAM). In the second approach, a novel method termed QSA-PAM (Quenching Simulated Annealing-Partitions Around Medoids) is introduced in order to solve the same problem. Several experiments are conducted with a real dataset. Results from descriptive model allowed the identification of 16 relevant features selected out of an original 356-feature dataset.

The goal of the predictive model was to apply the 16 relevant features to classify GBS subtypes. Again, two approaches were used. In the first approach, classification experiments were conducted using single classifiers. The second approach applied ensemble methods. Three types of experiments were performed in both approaches: four classes classification, One versus All classification and One

vs One classification. A comparison of performance between both approaches was made. Successful results in experiments in both approaches resulted in creating the first predictive model for GBS using machine learning techniques. Besides, the analysis performed in this work provides insight about the best methods for each classification case.

Agradecimientos

Agradezco a Dios por la vida, la buena salud, el trabajo y por todas las buenas experiencias.

Gracias a mi compañera en las buenas y en las malas, mi mejor amiga, la persona que más me ha querido en esta vida y a quien más quiero, en quien confío plenamente, quien me ha hecho feliz. Gracias Norma.

Gracias a la persona que fue mi padre y madre a la vez, quien cuidó de mí y me dió su amor hasta su muerte, mi abuelita, la Sra. Lucinda Sánchez Alfonso.

Gracias a mis directores de tesis, Dra. Juana Canul Reich y Dr. Juan Frausto Solís por sus conocimientos impartidos, por la dedicación, el esfuerzo y dirección. Sin ustedes no hubiera podido conseguir esto.

Gracias a los miembros de mi Comité Tutoral, Dra. Juana Canul Reich, Dr. Juan Frausto Solís y Dr. Alejandro Padilla Díaz por su acompañamiento a lo largo de todo el trayecto de mis estudios y por el cabal cumplimiento de sus funciones.

Gracias a los revisores y miembros de mi Jurado de Examen de Grado, Dr. Alejandro Padilla Díaz, Dr. Carlos Alberto Ochoa Ortíz, Dr. Juan Paulo Sánchez Hernández, Dra. Juana Canul Reich, Dr. Juan Frausto Solís por sus comentarios y sugerencias que permitieron mejorar el presente trabajo.

Gracias a mis compañeros de doctorado por sus consejos, apoyo y amistad.

Gracias a la Universidad Juárez Autónoma de Tabasco por su confianza, apoyo y respaldo.

Gracias al CONACyT por el apoyo otorgado durante mis estudios.

Publications

1. Juana Canul-Reich, José Hernández-Torruco, Juan Frausto-Solís, and Juan José Méndez-Castillo. Finding relevant features for identifying subtypes of guillain-barré syndrome using quenching simulated annealing and partitions around medoids. International Journal of Combinatorial Optimization Problems and Informatics, 6(2):11-27, May-August 2015.

2. José Hernández-Torruco, Juana Canul-Reich, and Juan Frausto-Solís. El aprendizaje computacional en el diagnóstico de enfermedades. Komputer Sapiens, 6(3):20-24, 2014.

3. José Hernández-Torruco, Juana Canul-Reich, Juan Frausto-Solís, and Juan José Méndez-Castillo. Predictores de falla respiratoria y de la necesidad de ventilación mecánica en el síndrome de guillain-barré: una revisión de la literatura. Revista Mexicana de Neurociencia, 14(5):162-170, 2013.

4. José Hernández-Torruco, Juana Canul-Reich, Juan Frausto-Solís, and Juan José Méndez-Castillo. Feature selection for better identification of subtypes of guillain-barré syndrome. Computational and mathematical methods in medicine, 2014:1-9, 2014.

5. José Hernández-Torruco, Juana Canul-Reich, Juan Frausto-Solís, and Juan José Méndez-Castillo. A kernel-based predictive model for guillain-barré syndrome. In Obdulia Pichardo Lagunas, Oscar Herrera Alcántara, and Gustavo Arroyo Figueroa, editors, Advances in Artificial Intelligence and Its Applications 14th Mexican International Conference on Artificial Intelligence MICAI 2015 Proceedings Part II , volume 9414 of Lecture Notes in Artificial Intelligence, pages 270-281, Cuernavaca, Morelos, Mexico, October 2015. Springer International Publishing.

6. José Hernández-Torruco, Juana Canul-Reich, Juan Frausto-Solís, and Juan José Méndez-Castillo. Towards a predictive model for guillain-barré syndrome. In Proceedings of the 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pages 7234-7237, Milano, Italy, august 2015.

Contents

List of figures

List of tables

Chapter 1

Introduction

Guillain-Barré Syndrome (GBS) is an autoimmune neurological disorder characterized by a fast evolution, usually it goes from a few days up to four weeks. The severity of GBS varies among subtypes, which can be mainly Acute Inflammatory Demyelinating Polyneuropathy (AIDP), Acute Motor Axonal Neuropathy (AMAN), Acute Motor Sensory Axonal Neuropathy (AMSAN) and Miller-Fisher Syndrome (MF). Currently, the identification of AIDP, AMAN, and AMSAN subtypes is made according to electrodiagnostic criteria applied after nerve conduction studies [83]. These three subtypes are known as electrophysiological subtypes. On the other hand, Miller-Fisher (MF) subtype is characterized by the clinical triad: ophthalmoplegia, ataxia and areflexia [59]. Reason why MF is considered a clinical subtype.

Hospitalization time and the cost of treatments vary according to the severity of the specific GBS subtype. The ultimate goal of a physician is to get patients to a full recovery. This can be more effectively achieved when an early diagnosis of the case is performed using a minimum number of medical features. This kind of problem can be successfully solved using machine learning techniques.

In this work, we applied machine learning techniques to build a descriptive and predictive model for GBS. The descriptive model allowed us to identify a minimum relevant feature subset from an original 365-feature dataset. Then, we analyzed these relevant features using different classifiers to create the predictive model. These classifiers include single methods like decision trees, kNN, SVM, among others. Ensemble methods like boosting and bagging were also included in this work.

1.1 Motivation

There are many successful applications of machine learning in medicine, biology and genetics [82, 7, 10]. Also, the machine learning techniques have been successfully applied in the early diagnosis of diseases, including neurological disorders [5, 93].

On the other hand, the studies published to date on this syndrome have been conducted from a medical perspective (etiology, pathogenesis, treatment, relationship with other diseases) using traditional statistical methods. In addition, studies have become successful in diagnosing the need for mechanical ventilation, as well as to determine the predictors of the risk of respiratory failure.

Besides, the identification of a reduced number of relevant features to predict GBS subtypes could guide physicians to design a faster, simpler and cheaper diagnosis of the case.

Finally, the availability of a real dataset for experimentation makes this work feasible.

1.2 Research objectives

a) General

To build a predictive/classifier model for GBS that is able to recognize clinical subtypes of this syndrome through its predictor variables.

b) Specific

- To build a machine learning model to identify predictive variables of GBS.
- To build a machine learning model that recognizes clinical patterns of GBS subtypes.
- To build a machine learning model that applies the predictor variables of GBS in a predictive model.

1.3 Hypothesis

Determining predictor variables, subtypes and classification of patients with GBS through computational machine learning models allows the modeling of this syndrome with at least 90% of accuracy.

Research questions arising from this thesis are:

- What are the predictor variables of GBS?
- What machine learning model will recognize clinical patterns of subtypes of GBS with at least 90% of accuracy?
- What model will allow to emit predictions on the GBS with at least 90% of accuracy?

1.4 Contributions

This work contributes in two disciplines, medicine and computer science. Figure 1.1 shows the contributions of this doctoral research. From the medical point of view, the contribution of this work is the identification of a minimum relevant feature subset, 16 features out of an original dataset of 365 features. This 16-feature subset allows to create the first predictive model for GBS using machine learning methods. This finding could guide physicians to design a faster, simpler and cheaper diagnosis of the GBS case.

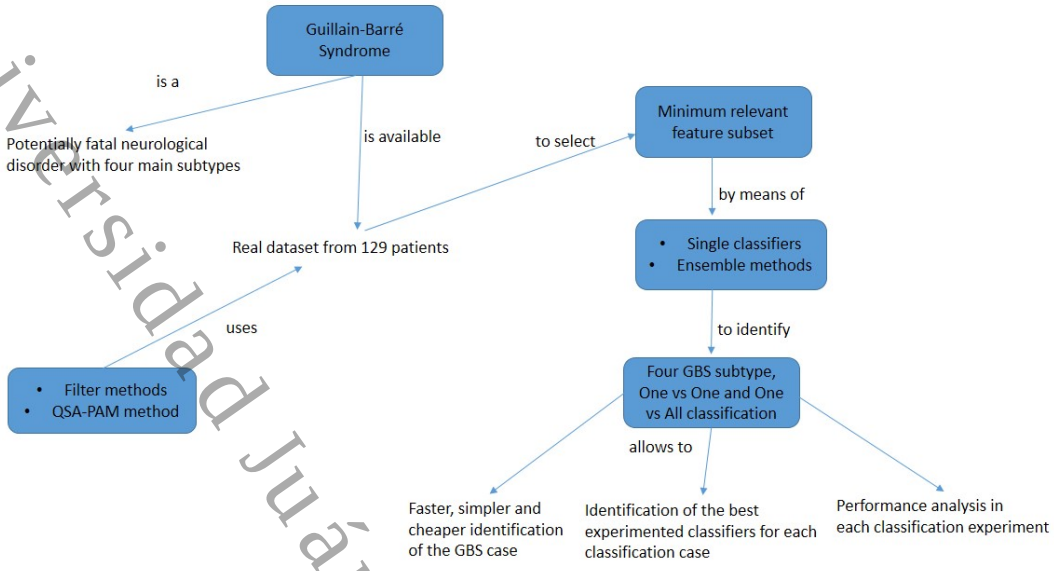

Figure 1.1: Contributions of the doctoral research.

As to computer science, this work contributes in the creation of a novel method termed QSA-PAM (Quenching Simulated Annealing-Partitions Around Medoids) method. PAM (Partitions Around Medoids) is a clustering algorithm and is thoroughly described in section 3.1.3.1. Simulated Annealing (SA) is a general purpose randomized metaheuristic that finds good approximations to the optimal solution for large combinatorial problems. QSA (Quenching Simulated Annealing), described in detail in section 4.3.1, is a version of SA consisting of two phases: Quenching and Annealing [35]. QSA-PAM was used in the first stage of this work, where the identification of the minimum relevant features were performed. The details of QSA-PAM are described in section 4.3.2. Results from descriptive model allowed the identification of 16 relevant features selected out of an original 356-feature dataset. In addition, a predictive model was created using these 16 relevant features. To build this model, single classifiers and ensemble methods were applied. Successful results in experiments in both methods resulted in creating the first predictive model for GBS using machine learning techniques. Besides, the analysis performed in this work provides insight about the best methods for each classification case.

Some of our findings are published in prestigious journals and international conferences as described in the list of publications section.

1.5 Organization

The problem of this dissertation focuses on knowledge discovery and is located within the Research Line called Intelligent Systems belonging to the area of Artificial Intelligence of the Interinstitutional

Doctorate in Computer Science.

This dissertation is organized as follows:

In chapter 3, some background is provided. This chapter defines the concept of data mining, its general classification, problems that it solves, data mining metrics and methods used along this work, and finally it outlines some recent applications in the medical area. It also describes the general characteristics of GBS, morbidity and mortality, diagnosis, pathogenesis and treatment. Chapter 4 fully describes the building process of the descriptive model. It starts with a description of the data used in the experiments performed in this chapter. The two approaches used for descriptive model are thoroughly explained, including experimental design, results, and discussion. The second approach of this model also includes a statistical analysis. In chapter 5, the building process of the predictive model is detailed. This chapter begins with a specification of the data used in the predictive model. As in the previous model, two approaches were used for the predictive model. These two approaches are explained in this chapter with full details. These include classification scenarios, experimental design, parameter optimization/setting for each method, results, and discussion. In this case, a statistical analysis is performed in the first approach. Finally, in chapter 6 conclusions and future works are provided.

Chapter 2

Background

2.1 Data mining: concepts, methods and metrics

2.1.1 Definition

Data mining enables the detection of patterns and associations that are hidden in datasets. This work relies on sophisticated statistical and computational techniques such as clustering, decision trees, neural networks, support vector machines, decision rules, regression, among others. Thereafter, patterns and associations found through these techniques will serve, using as a framework the field of medicine, to answer questions like: What set of symptoms are more related to each other in the onset of a disease?, What variables predict the course of a disease?, What is the combination of drugs or which treatments are most effective and in what order should be applied?, Will the patient survive a disease? Of course, all of this with a certain degree of accuracy.

According to Fayyad et al [33], as mentioned in [94], data mining is one of the phases (although one of the most important) of a larger process known as KDD (Knowledge Discovery in Databases). This process begins with the definition of a problem within a specific domain of knowledge, medicine, for example. The second step is to obtain and prepare the data for analysis, that is, integrate the data in a specific format and eliminate noise and inconsistencies. Then, data mining algorithms are applied to detect patterns and associations. The next phase is the interpretation of the results and their implications, this is where the discovery of knowledge is produced, this can lead to some of the previous phases. Finally, the new knowledge is used either to describe, predict or intervene in the modeling problem.

There are other authors -Shearer [75] for example- mentioned in [94] as well, who claim that data mining is a well-defined, independent of KDD, which has its own process, as established by the Cross-Industry Standard Process for Data Mining (CRISP-DM). This process consists of six steps: business understanding, data understanding, data preparation, modeling, evaluation and deployment. Because of the similarity between the two processes, KDD and data mining, the two terms have come to be used interchangeably. In this work, the term data mining will be used.

Data mining is divided into two main areas [12]: supervised learning and unsupervised learning (see Figure 2.1). This division is based on the data type that each one uses: labeled data (supervised

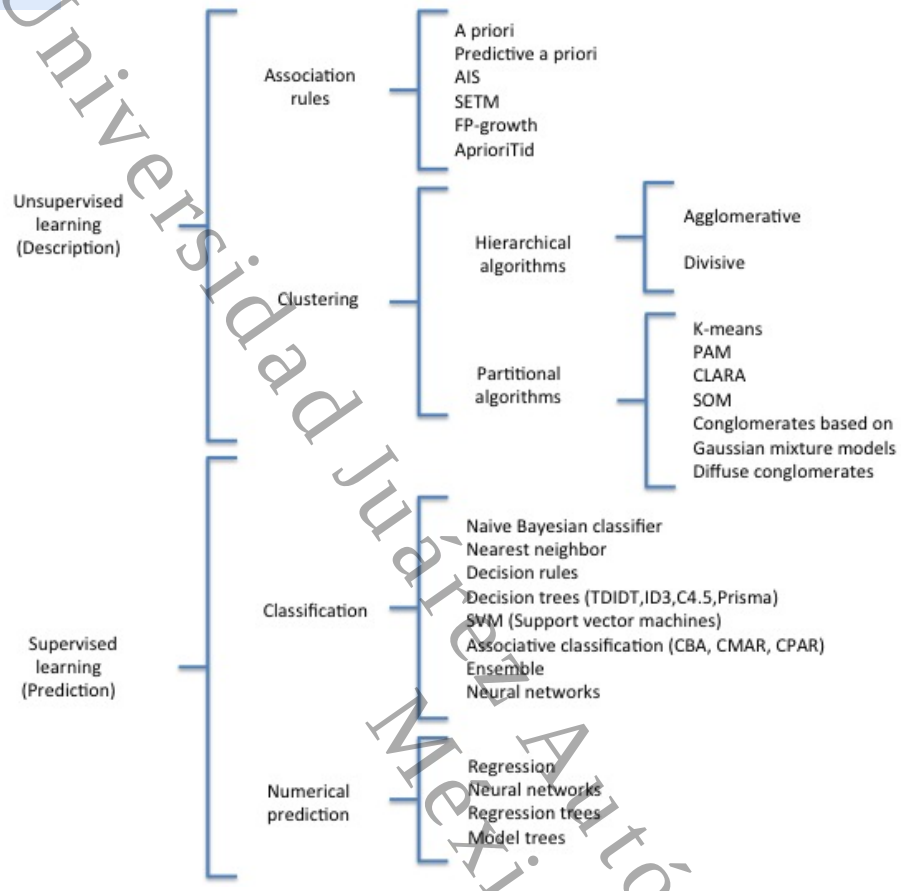

Figure 2.1: Learning approaches in data mining and some of their algorithms. [52]

learning) and unlabeled data (unsupervised learning). Labeled data have a special attribute called class or label. The unlabeled data lack this one. Suppose that an insurance company has a table where each row represents a customer's information, each row (or also called instance) has customer attributes such as age, number of children, sport practiced and salary. Using a technique of unsupervised learning, clustering, for example, we could get two well differentiated groups: the group of good clients and the group of overdue clients.

By applying this technique what we are doing is to describe data, to obtain as much information about them as possible. Once we find a label to classify data, in this case, the client type (good or overdue), we proceed to apply a supervised learning technique to predict the value of the class for a new instance, i.e. if a new client is good or overdue. For this reason, the term supervised learning is often used to refer to the predictive techniques and unsupervised learning to the descriptive ones.

Another research problem in data mining is that of feature selection in problems where it is considered that the number of attributes is high (high dimensionality), and consists of identifying from among the available variables, those directly related to the identification of the class (relevant attributes). The quantity and quality of the selected attributes impact the performance of the algorithms used

in subsequent steps to pre-processing, whether classification, association or clustering, therefore, is a very important process in data mining. One case where this occurs is in the analysis of microarray data. A microarray, also called DNA chip or biochip is a solid surface which binds to a collection of DNA fragments. The surfaces used to fix DNA are quite variable and can be glass, plastic and even silicon. Microarrays have been applied to the study of almost any biological problem: the study of genes that are differentially expressed between various conditions (healthy / diseased, mutant / wild, treated / untreated), identification of characteristic genes of a disease, predicting response to treatment, detection of mutations and polymorphisms in a single gene, among others. A microarray forms a matrix in which rows are genes and experimental conditions columns. The number of rows and columns used is such that the total attributes are measured in thousands. This makes the analysis of microarray data a challenge to find the most efficient algorithms for feature selection.

2.1.2 Filter methods for feature selection

Datasets might contain a mixture of "bad" and "good" features. "Bad" features are redundant or noisy features and make algorithms slow and inaccurate. Feature selection techniques allow to reduce the dimensionality of a dataset such that it only contains "good" features which would maximize the performance of the algorithms and thus enabling the possibility of reaching a higher accuracy [27]. For feature selection, several machine learning methods are available, which are usually classified as filter [45, 97, 73, 62, 28, 72], wrapper [58, 79, 56], embedded [14, 40, 16], and hybrid [76, 26, 23, 20]. From the machine learning point of view it is interesting to analyze the performance of feature selection methods in diverse scenarios with real data, as this case is.

For an initial exploratory study, we chose filter methods as they are the simplest and lowest computational demanding methods available in the literature and as they work independently of the clustering algorithms. We focus on five filter methods: Correlation-based Feature Selection (CFS), chi squared, information gain, consistency and symmetrical uncertainty.

2.1.2.1 Correlation-Based Feature Selection (CFS)

CFS [45] evaluates two aspects of a feature subset: its capacity to predict the class and the correlation between the features of the subset. This method seeks to maximize the first aspect and minimize the second one. This method results in a feature subset with the highest capacity to predict the class and the least correlation between features of the subset.

Given a feature subset S containing k features, CFS finds the goodness of S denoted (M_S) as follows:

$$M_S = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}} \tag{2.1}$$

where $\overline{r_{ff}}$ is the average correlation of all feature-feature pairs, and $k\overline{r_{cf}}$ is the average correlation of all feature-class pairs.

2.1.2.2 Chi squared

This method evaluates the chi-square statistic of each feature taken individually with respect to the class [97] and provides a feature ranking as a result.

The chi square test for a feature f and the class c is defined as follows:

$$x^2(f,c) = \frac{N[P(f,c)P(\overline{f},\overline{c}) - P(f,\overline{c})P(\overline{f},c)]^2}{(P(f)P(\overline{f})P(c)P(\overline{c}))} \tag{2.2}$$

where N is the number of observations in the dataset, $P(x,y)$ is the joint probability of x and y, and $P(x)$ is the marginal probability of x.

2.1.2.3 Information gain

Information gain measures the goodness of a feature to predict the class given that the presence or absence of the feature in the dataset is known. This method delivers a ranking according to the goodness of each feature.

Information gain [73] of a feature fk and a class ci is defined as follows:

$$IG(f_k, c_i) = \sum_{c \in \{c_i, \overline{c}_i\}} \sum_{f \in \{f_i, \overline{f}_i\}} P(f,c) log_2 \frac{P(f,c)}{P(f)P(c)} \tag{2.3}$$

where cc_i, c_i is the set of all classes, ff_k, f_k is the set of all features, $P(f,c)$ is the joint probability of feature f and class c, and $P(f)$ and $P(c)$ are the marginal probabilities of f and c respectively.

2.1.2.4 Consistency

This method finds the smallest feature subset that presumably improves the discriminatory power of the original feature subset. This subset has the highest consistency. The consistency for a given feature subset S is computed as follows [62]:

$$Consistency = 1 - inconsistencyrate \tag{2.4}$$

Let us define a pattern as a set of values for S. An inconsistency arises when two patterns match exactly all attributes except for the class. The inconsistency count for a pattern is the number of times it appears in the dataset minus the number of times it appears in the majority class. The inconsistency rate is the sum of all the inconsistency counts for all possible patterns of S divided by the total number of patterns [28].

2.1.2.5 Symmetrical uncertainty

This method measures the correlation between pairs of attributes using normalization of information gain. The normalization is performed to compensate for the bias of information gain to benefit attributes with more values and to ensure that they are comparable [62]. This method results in a feature ranking.

Symmetrical uncertainty is computed as follows [72]:

$$U(A,B) = 2 * (MI(A,B))/(Entropy(A) + Entropy(B)) \tag{2.5}$$

$$MI(A,B) = P(A,B) log_2 \frac{P(A,B)}{P(A)P(B)} \tag{2.6}$$

where $P(X)$ is the marginal probability of feature X and $P(A,B)$ is the joint probability of features A and B. Entropy is computed using the classical equation discussed in [62].

2.1.3 Cluster analysis

Cluster analysis or clustering is a machine learning technique that allows to group objects into clusters. Objects in the same cluster are similar and dissimilar from objects in other clusters. Cluster analysis has applications in many disciplines [9, 3, 6]. Particularly, cluster analysis has been used in medicine to identify disease subtypes [17, 91, 31], among other applications.

2.1.3.1 Partitions Around Medoids (PAM)

As stated before, the dataset used in this work combines categorical and numeric data. PAM is a clustering algorithm capable of handling such situations. It receives a distance matrix between instances as input. The distance matrix was computed using Gower's coefficient, described in section 3.1.3.2.

PAM, introduced by Kaufman and Rousseeuw [57], aims to group data around the most central item of each group, known as medoid, which has the minimum sum of dissimilarities with respect to all data points. PAM forms clusters that minimizes the total cost E of the configuration, that is, the sum of the distances of each data point with respect to the medoids. E is formally defined as:

$$E = \sum_{i=1}^{k} \sum_{o \in C_i} dist(o,m) \tag{2.7}$$

where k is the number of clusters, $o \in C_i$ is the set of objects in cluster C_i, $dist(o,m)$ is the distance between an object o and a medoid m.

PAM works as shown in **Algorithm 1** [46]:

2.1.3.2 Gower's similarity coefficient

Distance metrics are used in clustering tasks to compute the distance between objects. The distance computed is used by clustering algorithms to determine how similar or dissimilar the objects are. Based on it, a clustering algorithm can determine which cluster the objects could be thrown in. There are many distance metrics. Some of them deal with numeric data, such as Euclidean, Manhattan and Minkowski [46]. To deal with binary data, the Jaccard coefficent and Hamming are

Algorithm 1 Partitions Around Medoids (PAM).

```
k objects are arbitrarily selected as the initial medoids m
repeat
    The distance is computed between each remaining object o and the medoids m
    Each object o is assigned to the cluster with the nearest medoid m
    An initial total cost Eini is calculated
    for each m do
        A random o is selected
        A total cost Efin is calculated as a result of swapping an arbitrary m with the o randomly
        selected
        if Efin − Eini < 0 then
            m is replaced with o
        end if
    end for
until Efin − Eini = 0
```

often used [46]. For categorical data, some distance metrics are Overlap, Goodall and Gambaryan [11].

In this work we used for experimentation a dataset that contains mixed data, that is, both categorical and numeric data. To deal with this context we selected the Gower's coefficient. It is a robust and widely used distance metric for mixed data. We used this coefficient to obtain a matrix of distances between instances as required by PAM. It was introduced by Gower in 1971 [42]. Gower's coefficient is defined as follows [19]:

$$d^2{}_{ij} = 1 - S_{ij} \tag{2.8}$$

$$S_{ij} = \frac{\sum_{h=1}^{p_1}(1 - \frac{|x_{ih} - x_{jh}|}{G_h}) + a + \alpha}{p_1 + p_2 - d + p_3} \tag{2.9}$$

where p_1 is the number of quantitative variables, p_2 is the number of binary variables, p_3 is the number of qualitative variables, α is the number of coincidences for qualitative variables, a is the number of coincidences in 1 (feature presence) for binary variables, d is the number of coincidences in 0 (feature absence) for binary variables, G_h is the range of the h-th quantitative variable.

Gower's coefficient lies in the range $[0, 1]$. A value near to 1 indicates strong similarity between items and a value near to 0 indicates weak similarity.

2.1.3.3 Purity

The quality of a clustering process can be evaluated using two types of metrics: internal and external. Internal metrics evaluate the quality of a clustering process based on some intrinsec characteristics, regularly, intra and inter cluster distances. Internal metrics assign high scores to clusters with

largest distances among them (separability) and shortest distances among members of the same cluster (compactness). These metrics are very useful when the number of clusters is not known at all. Examples of internal metrics are Q-modularity [13], Davies-Bouldin index, Dunn index and silhouette [4].

External metrics evaluate the quality of clusters based on data not used during the clustering process, such as the ground truth; that is, the real classes of the instances. The larger the number of instances correctly located according to the ground truth, the higher the index. Some examples of external metrics are Rand index, Folkes and Mallows index, Hubert's T statistic [44] and purity [63].

The dataset used in this work provides the ground truth. We know there are four classes in the dataset. The goal of this study was to find a minimum set of features that as purely as possible identify four clusters, each corresponding to one class. To achieve this goal we selected purity as the metric to evaluate the quality of the clustering process.

Purity validates the quality of a clustering process based on the location of data in each cluster with respect to the true classes. The more objects in each resultant cluster belong to the true class, the higher the purity. Formally [63] :

$$purity(\Omega, C) = \frac{1}{N} \sum_{k} \max_{j} |w_k \cap c_j| \tag{2.10}$$

where N is the number of labeled samples, $\Omega = \{w_1, w_2, ..., w_K\}$ is the set of clusters found by the clustering algorithm, $C = \{c_1, c_2, ..., c_J\}$ is the set of classes of the objects, K is the number of clusters, J is the number of classes, $|w_k \cap c_j|$ is the number of objects of cluster k belonging to class j, w_k is the set of objects in cluster k, c_j is the set of objects in class j.

Purity ranges in [0,1]. A purity value of 1 indicates that all objects in each cluster belong to the same class.

2.1.4 Performance metrics for classification

2.1.4.1 Confusion matrix

A confusion matrix, also known as contingency table, is a table that shows the performance of a predictive model. This table is derived of the comparison of the result of a predictive model with the actual values (ground truth). Each row contains the values predicted by the model. Each column represents the actual values. A confusion matrix for a binary classification model is shown in Table 2.1.

Table 2.1: Confusion matrix for a binary classification.

	Actual value	
Predicted value	TP	FP
	FN	TN

where:
TP = True Positives, FP = False Positives, TN = True Negatives, and FN = False Negatives.

A confusion matrix is the starting point for the calculation of the measures of performance of a predictive model.

2.1.4.2 Accuracy

Accuracy is a primary measure to evaluate the performance of a predictive model. For binary classifications, accuracy is computed dividing the number of instances correctly identified by the total number of instances. Formally:

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{2.11}$$

where TP = True Positive, TN = True Negative, FP = False Positive and FN = False Negative.

2.1.4.3 Average accuracy

A more suitable measure for multiclass classification problems is the average accuracy [77]. It assesses the accuracy individually for each class without distinguishing between the other classes. The original n x n confusion matrix obtained from the multiclass problem is transformed in n 2 x 2 confusion matrices. From each binary confusion matrix, a per-class accuracy is computed. All individual per-class accuracies are averaged to give a final accuracy. Formally:

$$AverageAccuracy = \frac{\sum_{i=1}^{l} \frac{TP_i + TN_i}{TP_i + FN_i + FP_i + TN_i}}{l} \tag{2.12}$$

where TP_i = True Positive for class i, TN_i = True Negative for class i, FP_i = False Positive for class i and FN_i = False Negative for class i.

2.1.4.4 Balanced accuracy

The balanced accuracy is a better estimate of a classifier performance when a unequal distribution of two classes is present in a dataset. We used it in this work in OVO and OVA classifications. Balanced accuracy is defined as:

$$BalancedAccuracy = \left(\frac{TP}{TP + FN} + \frac{TN}{FP + TN} \right) /2 \tag{2.13}$$

where TP = True Positive, TN = True Negative, FP = False Positive and FN = False Negative.

2.1.4.5 Error rate

It measures the proportion of errors made by a classifier over a whole set of instances. It is the complement of accuracy, that is 100 % - accuracy.

2.1.4.6 Sensitivity

Sensitivity measures the proportion of true positives, which were correctly identified by a predictive model. For a diagnostic test, sensitivity measures the ability of a test to detect ill subjects.

2.1.4.7 Specificity

Specificity measures the proportion of true negatives, which were correctly identified by a predictive model. For a diagnostic test, specificity measures the ability of a test to detect healthy subjects.

2.1.4.8 Receiver Operating Characteristic Curves (ROC)

A ROC (Receiver Operating Characteristic) curve measures the performance of a classifier based on how well it separates the group being tested into those belonging to one class and another. ROC curves apply only for binary classifications. A ROC curve is a graphical representation of the True Positive Rate (TPR) in y axis, and the False Positive Rate (FPR). These measures are defined by:

$$TPR = \frac{TP}{TP + FN} \tag{2.14}$$

$$FPR = \frac{FP}{FP + TN} \tag{2.15}$$

Performance is measured by the AUC. AUC from a ROC curve ranges [0,1] where 1 is a perfect classification. An AUC of 0.5 represents a random classification.

2.1.4.9 Multiclass Area Under the Curve (AUC)

Hand et al. [47] derived an AUC for multiclass problems, which measures the pairwise discriminability of classes. Formally:

$$AUC_{total} = \frac{2}{|C|\,(|C|-1)} \sum_{\{C_i, C_j\} \in C} AUC(C_i, C_j) \tag{2.16}$$

where $AUC(C_i, C_j)$ is the area under the two-class ROC curve involving classes C_i and C_j. The summation is calculated over all pairs of distinct classes, regardless of order [32].

2.1.4.10 Kappa statistic

Kappa statistic, introduced by Cohen [21], measures the agreement between the classifier itself and the ground truth corrected by the effect of the agreement between them by chance. Formally:

$$kappa = \frac{P(A) - P(E)}{1 - P(E)} \tag{2.17}$$

where $P(A)$ is the proportion of agreement between the classifier and the ground truth, $P(E)$ is the proportion of agreement expected between the classifier and the ground truth by chance.

2.1.5 Single classifiers

2.1.5.1 k Nearest Neighbors (kNN)

kNN [34] assigns the class for a new instance based on the majority class in the k closest neighbors to this new instance. The k neighbors are selected according to some distance metric: Minkowski, Jaccard, Manhattan, Chebychev, among others. In this study we use Minkowski distance. Minkowski is a generic distance measure suitable for use with any data type as feature vector [24]. Additionally, Minkowski distance allows to explore various distance configurations by setting an exponential parameter in its original formula. The version of kNN used in this study requires the tuning of two parameters: k and d. k represents the number of neighbors and d the exponent in the Minkowski distance formula.

2.1.5.2 Support Vector Machines (SVM)

SVM was first introduced by Vapnik and colleagues [86]. Given a set of training instances (input space), where each instance belongs to class A or class B, SVM uses a mapping function (kernel) to transform the input space into a higher dimension space (feature space), that is, if input space is 2-D, then it is mapped into a 3-D space. In the feature space, SVM finds a hyperplane that gives the largest separation between classes, named maximum marginal hyperplane. The maximum margin hyperplane has the largest distance from the hyperplane to the closest training instances. The instances located in the boundaries of the hyperplane are called support vectors. However, the largest margin is not always the best solution since it can compromise the generalization of the model to new instances. For the sake of flexibility, SVM introduces a parameter C that creates a soft margin that allows for some errors in classification but at the same time it penalizes them. A tuning procedure is necessary for finding the best value of C.

In this study, we use four different kernels: linear (SVMLin), Gaussian (SVMGaus), polynomial (SVMPoly) and Laplacian (SVMLap). Each of these kernels has particular parameters and they must be tuned in order to achieve the best performance. Table 2.2 shows the parameters of each kernel used in this study.

Table 2.2: Kernel parameters.

Kernel	Parameter
Linear (SVMLin)	C
Polynomial (SVMPoly)	C, degree, σ (γ), coef
Gaussian (SVMGaus)	C, σ (γ)
Laplacian (SVMLap)	C, σ (γ)

Below is shown the details of each of these kernels' functions.

Linear kernel function:

$$k(x, y) = x^T y + C \tag{2.18}$$

where x and y are vectors in the input space, and C is the penalty parameter.

Polynomial kernel function:

$$k(x, y) = (x^T y + coef)^d \tag{2.19}$$

where x and y are vectors in the input space, d is the degree of the polynomial, and $coef >= 0$ is a free parameter trading off the influence of higher-order versus lower-order terms in the polynomial. When $coef = 0$, the kernel is called homogeneous. When $coef > 0$, the kernel is called non-homogeneous.

Laplacian kernel function:

$$k(x, y) = exp(-\sigma \|x - y\|) \tag{2.20}$$

where x and y are vectors in the input space, and σ controls the standard deviation of the Gaussian function.

Gaussian kernel function:

$$k(x, y) = exp(-\gamma \|x - y\|^2) \tag{2.21}$$

where x and y are vectors in the input space, and γ controls the standard deviation of the Gaussian function.

2.1.5.3 C4.5

C4.5, introduced by Quinlan [70], builds a decision tree from training data using recursive partitions. In each iteration, C4.5 selects the attribute with the highest gain ratio as the attribute from which the tree branching (splitting attribute) is performed. This results in a more simplified tree (fewer subtrees). C4.5 is widely used in classification problems of diverse nature. In this work, C4.5 was applied with pruning.

2.1.5.4 Artificial Neural Networks (ANN)

An Artificial Neural Network (ANN) is a machine learning algorithm that emulates the functioning of biological neural systems, like the human brain. It is composed by a number of neurons interconnected, which together parallely process input data and produce an output [51]. Figure 2.2 shows a diagram of a neuron, the basic processing unit of an ANN. Each neuron receives a set of inputs through connections with other neurons. Each input has a weight, its magnitude represents the efficiency with which the output signal of a unit is transmitted to the other. If the weight is positive, an excitation effect is produced. Otherwise, if the weight is negative, an inhibition effect is produced. The sum of the products of the inputs and their respective weights are computed by a propagation function. An activation function will activate the neuron if the result is greater than a threshold, meaning that the neuron will send a signal to communicate with another neuron. Finally,

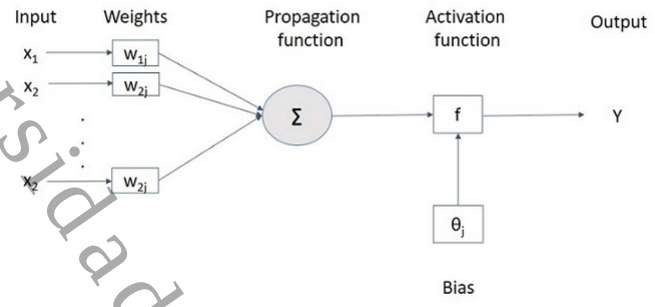

Figure 2.2: Diagram of an artificial neuron.

a transference or output function transforms the result of the activation function into a formatted output, e.g. a Sigmoid function produces values in the range $[0, 1]$. Formally, the output of an ANN is given by:

$$Y = f(\sum_i w_{ij} x_i) \tag{2.22}$$

Figure 2.3 shows a complete ANN with all the neurons interconnected. Typically, an ANN has three layers: the input layers, the hidden layers, and the output layer. Nevertheless, there are many variations of this model. There are for example, ANN's with a single hidden layer. The inputs represent the set of features used in the training phase. These inputs are processed by the neurons present in this layer and passed on to the hidden layers. The data processing continues through one or more hidden layers. Finally, the output layer emits a result for every instance received as input. In some cases, multiple outputs can exist in an ANN.

2.1.5.4.1 Single Layer Perceptron (SLP)

As indicated by its name, it uses only one hidden layer. This is the simplest kind of feed-forward ANN. In feed-forward ANNs, the neurons are connected in only one direction, from the input units passing through the hidden layer, and to the output layer [51]. This type of ANN requires the tuning of two parameters: size, representing the number of units in the hidden layer, and decay, a parameter that exponentially declines the weights to zero. Multiclass classification is available in the implementation used in this study by using a softmax parameter.

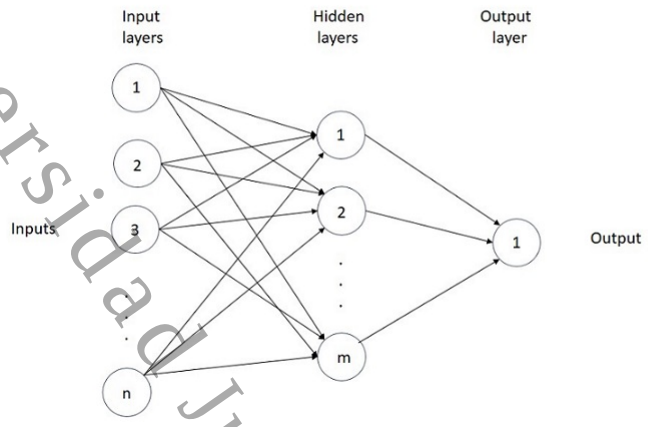

Figure 2.3: Diagram of an Artificial Neural Network (ANN).

2.1.5.4.2 Multilayer Perceptron (MLP)

This is another feed-forward ANN. This kind of ANN is composed by multiple layers. This characteristic allows this model to solve problems that are not linearly separable. The training phase is usually performed by an algorithm called back-propagation. This algorithm executes a large number of training cycles in order to improve the classification task. In each cycle, the classification error is computed by comparing the output values with the correct answers. This information is introduced back into the network to adjust the weights of each connection, and so reduce the error in each iteration until convergence [46]. The implementation used in this study uses only one tuning parameter: size, indicating the number of units in the hidden layer.

2.1.5.4.3 Radial Basis Function ANN (RBF)

This type of ANN uses a Radial Basis Function as activation function. In this study, we use a version named RBF with Dynamic Decay Adjustment (RBF-DDA) [66]. This ANN uses a single hidden layer and its number of units is determined automatically during the training phase. The implementation used in this study uses a negative threshold parameter that controls the units added to the network, and this parameter requires optimization.

2.1.5.5 JRip

JRip is an implementation of the RIPPER (Repeated Incremental Pruning to Produce Error Reduction) algorithm [22]. It belongs to the rule induction learning methods. JRip builds a set of rules that identify the classes while minimizing the amount of error. A rule has the form:

if attribute1 <relational operator> value1 <logical operator> attribute2 <relational operator> value2 < ... > **then** decision-value

The rule set for each class is optimized using a pruning method. JRip requires optimization of the NumOpt parameter, representing the number of optimization runs.

2.1.5.6 OneR

OneR stands for One Rule. This algorithm, as JRip does, belongs to the rule induction learning methods. It generates one rule for each predictor in the dataset, and among all selects the rule with the smallest total error as the one rule [92]. One advantage of rules is that are simple for humans to interpret.

2.1.5.7 Naive Bayes

Naive Bayes is a probabilistic classifier based on Bayes' theorem. This classifier assumes that the value of a particular feature is independent of the value of any other feature, given the class variable. That is, it considers each feature contributes independently to the probability of an instance belongs to a class, regardless of any possible correlations between features. Bayes' theorem is formally defined as:

$$P(H \mid X) = \frac{P(X \mid H)P(H)}{P(X)} \tag{2.23}$$

where X = $\{x_1, x_2, ..., x_n\}$ is an instance composed of n attributes, H is some hypothesis such as X belongs to a specified class C, $P(H \mid X)$ is the a posteriori probability of H conditioned on X, $P(X \mid H)$ is the a posteriori probability of X conditioned on H, P(H) is the a priori probability of H, and P(X) is the a priori probability of X.

Suppose that there are m classes, C_1,C_2,...,C_m. Given an instance X, this classifier will predict that X belongs to the class having the highest a posteriori probability, conditioned on X [46].

2.1.5.8 Binary Logistic Regression (BLR)

It is a regression model where the dependent variable or outcome is categorical [51]. It is used to predict a binary response based on one or more predictors (independent or explanatory variables). Logistic regression belongs to the probabilistic models. The probabilities of the possible outcomes are modeled using the logistic function:

$$F(x) = \frac{1}{1 + \epsilon^{-(\beta_0 + \beta_1 x)}} \tag{2.24}$$

A graph of the logistic function is shown in Figure figBinLogReg. The values of the outcome are coded 0 or 1.

2.1.5.9 Multinomial Logistic Regression (MLR)

This type of logistic regression deal with situations where the outcome can have three or more values that are not ordered. Multinomial logistic regression is a simple extension of binary logistic regression that allows for more than two categories of the dependent or outcome variable. Like binary

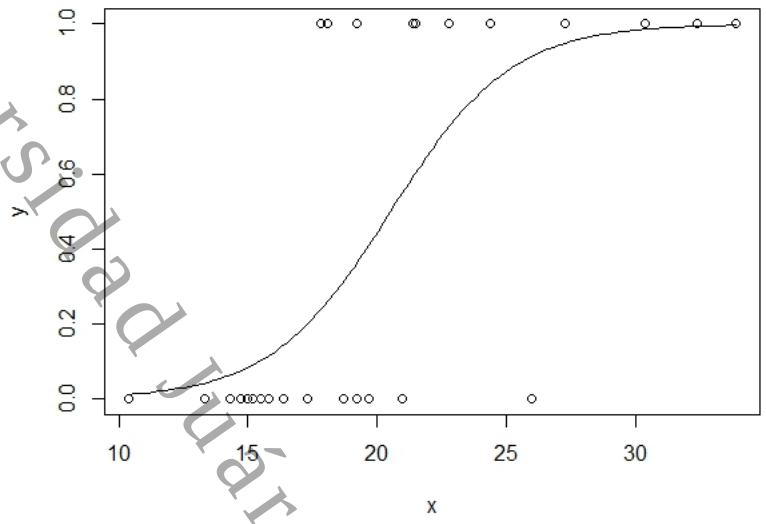

Figure 2.4: Logistic regression function.

logistic regression, multinomial logistic regression uses maximum likelihood estimation to evaluate the probability of categorical membership [49].

2.1.5.10 Linear Discriminant Analysis (LDA)

The aim of LDA is to find rules to assign objects into one of two or more classes. To solve this problem, a sample of n objects is given, from which the classes they belong is known. In addition, measures of a set of p predictor variables is available. It it assumed that each class is defined by a different probability distribution [51].

Let q be the number of classes and C_1,C_2,...,C_q each of the classes. For each class there is a specific probability density function $f_k(x) = f(x \mid C_k)$, where x represents the p predictor variables. On the other hand, each class may have a different a priori probability $P(C_k)$. From these two probabilities, rules to assign objects into one of two or more classes are determined.

LDA aims at maximizing the a posteriori probability, previously defined in Naive Bayes classifier.

2.1.6 Ensemble methods

2.1.6.1 Boosting

It is a type of ensemble method that combines multiple homogeneous classifiers by voting [92]. Boosting aims at turning a set of weak learners into a strong learner. A weak learner is defined as a classifier which is only slightly correlated with the true classification (it can label examples better than random guessing). In contrast, a strong learner is a classifier that is arbitrarily well-correlated

with the true classification.

Boosting is applied iteratively to the data, so that a sequence of weak classifiers is produced. Boosting assigns weights to the instances, initially all instances have the same weight. In each iteration the weights are modified, increasing the weight of the misclassified instances and thus the weak learners focus more on them. Finally, all the weak classifiers are combined by a weighted voting where the weight assigned to each classifier depends on its error rate. [51]

In this work, we implemented AdaBoost (Adaptive Boosting algorithm) [39], which uses decision trees as weak learners.

2.1.6.2 Bagging

Bagging, meaning bootstrap aggregating, was introduced by Breiman. Bagging is a method for generating multiple versions of a predictor and using these to get an aggregated predictor [15]. Bagging generates m new training sets by making bootstrap replicates from the original training set. The m models are fitted using the above m bootstrap (random sampling with replacement) samples and combined by averaging the output (for regression) or voting (for classification). In this work, bagging was implemented using decision trees as single classifiers.

2.1.6.3 C5.0

C5.0 is a improved version of C4.5, introduced by Quinlan [70]. Its major improvement is the implementation of boosting. Boosting improves trees and gives them higher precision.

2.1.6.4 Random forest

Random Forest, introduced by Breiman and Adele Cutler [16], is a predictive algorithm built by a bootstrap ensemble of CART trees. Given N number of training data points and M number of predictor variables, this algorithm generates many bootstrap samples by selecting N data points with replacement from the training dataset. Then, a CART tree is trained on each bootstrap sample using m randomly chosen predictors out of the original M predictors ($m \ll M$ if M is large). The trees are fully grown without pruning. Random forest is robust against overfitting.

2.1.6.5 Random subspace

Random subspace, introduced by Ho [54], consists of several base classifiers each operating in randomly chosen subspaces of the original feature space. These classifiers are usually combined by simple majority voting to generate the final class.

2.2 Medical applications

Medical data mining is a research area that has gained prominence in recent years. There are extensive databases in Mexico and worldwide of public and private institutions with a tremendous amount of clinical information of patients, containing hidden patterns of diverse diseases. An analysis of these databases, using appropriate techniques of data mining, could lead to identify these

patterns and get knowledge that can help doctors in predicting, preventing and treating diseases.

Asha, Natarajan and Murthy [7] carried out a research related to the application of data mining techniques for the diagnosis of tuberculosis. In this research, a dataset of 700 records of real patients of an urban hospital was used and 12 attributes (symptoms) were considered: age, chronic cough, weight loss, intermittent fever, night sweats, sputum, blood cough, chest pain, HIV, radiographic findings, wheezing and type of tuberculosis. Two techniques were applied: Association Rules and Associative Classification. By applying the former technique on the dataset, it was found a significant association between symptoms of tuberculosis; however, this technique generates a large number of repetitive rules. The latter, Associative Classification, integrates Association Rules and Classification. It uses Association Rules algorithms such as Apriori or Frequent Pattern Growth to generate the complete set of association rules. Then, it selects a small set of high quality rules to use for prediction. This technique results in a lower amount of rules compared to the former.

Barati et. al. [10] published a survey on data mining approaches used to predict skin diseases. In this survey, it was reviewed, among others, a study to find significant risk factors for certain cancers, in which three algorithms of association were used: A priori, A priori predictive and Tertius. It was concluded that A priori is the best algorithm for generating association rules for the data used in this particular study. In another research, Support Vector Machines (SVM) were applied to the problem of differential diagnosis of erythemato-squamous diseases (psoriasis, seborrheic dermatitis, among others.).

Arisi et. al. [5] applied data mining techniques to identify biomarkers of microarray gene expression in the brain of a mouse model for Alzheimer's disease. The identification of biomarkers for Alzheimer's disease is critical because the development of therapies for disease modification may depend on the discovery and validation of such markers. This study identified certain genes as potential biomarkers for early and late stages of the disease and concluded that the software used is a powerful and reliable tool for analyzing gene expression data of a large scale. This software is called Microarray Logic Mining (MLM) and it implements methods such as clustering, iterative feature selection, logic formulas identification and ranking by correlation.

Wu and Guo [93] applied decision trees and neural networks to analyze a database of Parkinson's disease and to explore whether the variables measuring the voice can be a diagnostic tool for this disease. Parkinson's disease is a neurological disorder and, as many diseases like this, affects phonation of patients; hence the voice can be a valuable tool in the diagnosis of neurological illnesses. By studying the voice measurements of patients, medical researchers can create a mechanism for distinguishing the sick from the healthy people and this way alerting physicians and people of symptoms timely, so that they can take appropriate treatments.

Tsai [82] applied an Artificial Neural Network (ANN) model along with with a statistical method called Agglomerative Hierarchical Clustering Techniques (AHCT) to study the possibility that DNA viruses are involved in the development of breast cancer. ANNs are systems that try to mimic brain function. They consist of an interconnection of neurons in a network which works to produce output stimulation. There are a variety of models of ANNs, but the most used ANN is the multilayer perceptron with back propagation because their performance is higher than other neural

network algorithms. Hence, a back propagation model was selected for this study. The conclusion of this study was that the combination of the techniques used, ANN and the AHCT method, contributes to some extent in the detection of aggressive viral factors, however it is not the best method.

2.3 Guillain-Barré Syndrome (GBS)

The Ministry of Health of Mexico in its Clinical Practice Guideline for the Diagnosis and Management of Guillain-Barré syndrome in the acute stage, in primary care [29] defines it as "...an acute inflammatory demyelinating polyradiculoneuropathy of autoimmune origin, characterized by a progressive motor deficit symmetrical, ascending, and hyporeflexia or areflexia generalized; in its classic form is accompanied by sensory symptoms of cranial nerve involvement and dysautonomic disorders"(p. 9). In other words, GBS is an autoimmune neurological disorder that attacks the peripheral nervous system so that the nerves can not transmit signals effectively from the encephalon and muscles lose their ability to react to commands from the brain resulting in loss of sensation and mobility. GBS can be mainly of two types, although there are various subtypes, which are axonal or demyelinating. GBS is called axonal when it attacks the nerve fibers, that is the axons, and demyelinating when it damages the sheath that covers the axons, that is the myelin.

Although the exact cause of the syndrome is not yet determined, it is believed it is developed from an allergy or hypersensitivity to any type of antigen, such as a virus or bacterium. Lestayo and Hernandez [60] mention that

> " in over 60% of patients, onset of symptoms is preceded, in a period of 1-4 weeks, by a digestive or respiratory event, usually of viral or bacterial infectious nature. It is suggested that both bacterial infections (Campylobacter jejuni and Mycoplasma pneumoniae, etc.) as viral (cytomegalovirus and Epstein-Barr virus, among others) can trigger the syndrome. Acute enteritis caused by C. jejuni (gram-negative enterobacteria) is the most referred and studied. It has been reported the occurrence of GBS after noninfectious events such as vaccination, surgical instrumentation, immediate postpartum and, unusually, pregnancy, among various other factors."(p. 1)

Axonal forms of GBS are Acute Motor Axonal Neuropathy (AMAN) and Acute Motor Sensory Axonal Neuropathy (AMSAN). Demyelinating forms are subdivided into Acute Inflammatory Demyelinating Polyradiculoneuropathy (AIDP) and Miller-Fisher syndrome.

Zuniga et al [98] found that the most common electrophysiological subtype among Mexican adults is AMAN, with high presentation of mixed type neuropathy and AMSAN. The same study found that there is a predominance of axonal forms of GBS among Mexican adults.

GBS is the most common cause of acute paralysis of the Peripheral Nervous System (PNS) in developed countries. Its annual incidence is 1.1 to 1.8 per 100,000 inhabitants. In contrast, most studies show a linear increase in incidence in relation to age and people aged 50-80 years, the incidence is between 1.7 and 3.3 per 100,000 inhabitants. [29].

In Mexico, national statistics contemplate acute flaccid paralysis as an obligatory reportable disease,

which can be related to the incidence of GBS, since it is not reported any cases of polio since at least 2000. For the year 2008, an annual incidence rate per 100,000 inhabitants of 0.4180 is reported [29].

2.4 State of the art

This section is organized into three subsections: diagnosis, prediction and prognosis. The diagnosis subsection reviews the recent and classic literature about methods for diagnosis of the disorder. In the prediction subsection, publications whose aim is to find variables that predict GBS associated ailments such as the need for mechanical ventilation or variable that seek to predict the course of the syndrome are outlined. In the last section, corresponding to prognosis, results of research concerning variables that emit a prognosis of the syndrome are presented.

2.4.1 Diagnostic

Emitting a timely and accurate diagnosis is vital in the field of medicine, among other reasons because the appropriate choice of treatment and therefore the patient relief, depends on this. Diagnosis in GBS is especially important given the rapid evolution of the disease, on average eight days.

It is presented in [41] universally accepted diagnostic criteria established by Asbury and Cornblath, which are summarized in table 2.3.

Several disorders have symptoms similar to those found in GBS, so doctors have to make a differential diagnosis in order to rule out other conditions and emit a precise diagnosis. In [41] disorders are mentioned to be considered in the differential diagnosis, which are shown in table 2.4.

One of the current methods for the diagnosis of GBS is to conduct a neurophysiological study, specifically tests of nerve conduction velocity (NCV), wave F (OF) and electromyography, since one of the signs of GBS is the loss of the velocity of the signals traveling through the nerves. Neurophysiology studies have been considered the main instrument to help differentiate between axonal and demyelinating GBS [41], the criteria for such differentiation are shown in table 2.5.

Hernández [50] conducted a study to compare the pattern of disturbance of neurophysiological studies in patients with GBS and chronic inflammatory demyelinating polyneuroradiculopathy (CIDP) to find differences between them, in order to contribute to the differential diagnosis of both disorders. GBS and CIDP are autoimmune demyelinating neuropathies that share many clinical and laboratory similarities. Eighteen patients with GBS and twenty seven with CIDP were assessed who underwent studies of nerve conduction velocity (NCV) and F wave (OF). This research concludes that through neurophysiological studies, the differential diagnosis between GBS and CIDP can be made. Likewise, it concludes that different electrophysiological behavior exists between GBS and CIDP, more sensory involvement in CIDP. The pattern of involvement is different in both diseases: in GBS is predominantly myelinated in the sensory study, axono-myelinated in the motor study; while in CIDP is axono-myelinated in the sensitive study, and myelinated in the motor study.

The study of cerebrospinal fluid (CSF) is another of the classical methods for the diagnosis of GBS, it consists in the analysis of the concentration of protein in the fluid that bathes the brain and spinal

Table 2.3: Diagnostic criteria for typical GBS proposed by Asbury and Cornblath. [8]

Features required for diagnosis:
o Progressive weakness in both arms and legs
o Areflexia

Features strongly supporting diagnosis:
o Progression of symptoms over days, up to four weeks
o Relative symmetry of symptoms
o Mild sensory symptoms or signs
o Cranial nerve involvement, especially bilateral weakness of facial muscles
o Recovery beginning two to four weeks after progression ceases
o Autonomic dysfunction
o Absence of fever at onset
o High concentration of protein in cerebrospinal fluid, with fewer than 10 cells per cubic millimeter
o Typical electrodiagnostic features

Dubious diagnostic features:
o Presence of a sensory level
o Marked or persistent asymmetry on symptoms or signs
o Severe and persistent sphincter dysfunction
o More than 50 cells / mm 3 in the CSF

Features excluding diagnosis:
o Diagnosis of botulism, myasthenia, poliomyelitis, or toxic neuropathy
o Abnormal porphyrin metabolism
o Purely sensory syndrome, without weakness
o Recent diphtheria

Table 2.4: Differential diagnosis of Acute Flaccid Paralysis Syndrome. Adapted from Peter van Doorn [85]

Motor Neuron Disease

o Acute form of progressive spinal muscular atrophy (amyotrophic lateral sclerosis)
o Bulbar form (dysarthria, dysphagia and tongue denervation)
o Acute poliomielitis
o Other neurotropic viruses (Coxsackie, echovirus, enterovirus, West Nile Virus)

Polyneuropathy

o Guillain-Barre Syndrome: Acute Inflammatory Demyelinating Polyradiculoneuropathy (AIDP)
o Other forms of Guillain-Barre syndrome: acute motor axonal neuropathy, acute motor sensory axonal neuropathy, acute pandysautonomia, Miller-Fisher syndrome
o Intoxication by Karwinskia humboldtiana (tullidora)
o Polyneuropathy of the seriously ill in intensive care
o Porphyria
o Vasculitis, paraproteinemia
o Neurotoxicity of metals (arsenic, lead, thallium), abuse or intoxication by substances and drugs, biological toxins, drug effects
o Borreliosis (Lyme disease)

Disorders of neuromuscular transmission

o Myasthenia gravis
o Paraneoplastic myasthenic sndrome
o Botulism
o Hypermagnesemia
o Aminoglycosides
o Neuromuscular blocking agents (pancuronium or vecuronium)

Muscle and metabolic disorders

o Acute hypokalaemic paralysis
    o Chronic depletion of potassium
    o Periodic thyrotoxic paralysis
o Familial hypokalemic periodic paralysis
o Familial hyperkalemic periodic paralysis
o Necrotizing myopathies
o Acid maltase deficiency
o Mitochondrial myopathy

Table 2.5: Criteria for distinguishing between axonal and demyelinating neuropathy. [41]

	Axonal	Demyelinating
Motor sensory nerve conduction velocity (m/s)	Normal or decreased	Significant decrease
Amplitude of motor and sensory unit compound potential	Significant decrease	Normal or decreased
Electromyography	Signs of denervation	Nonspecific
Proximal latencies	Normal	Conduction block
F wave and H reflex	Diminished or absent	Diminished or absent

cord. This analysis is performed via a lumbar puncture that involves inserting a spinal needle into the lumbar area between two vertebrae in order to extract a sample. The finding of an increase in proteins with less than 10 cells/mm3 is a criterion that strongly support the diagnosis of GBS.

Ramos and Cacho [41] show in table 2.6 the differences among GBS subtypes.

2.4.2 Prediction

Prediction consists in applying either statistical or data mining methods to determine in advance the diagnosis or prognosis of a disease or syndrome, in order to take measures to alleviate the condition. It is useful for clinicians to have an automated tool to assist them in decision making.

In GBS, studies have been conducted primarily focused to identify variables that determine in advance the appearance of certain syndrome-related consequences, such as the need for mechanical ventilation.

This phenomenon occurs in approximately 30% of cases of GBS [68], making it one of the most important and recurring research topics regarding this syndrome. In patients who develop respiratory paralysis, early detection (ideal therapeutic window less than three weeks) allows a diagnosis that significantly influences decision making, such as the need for assisted ventilation or the type of treatment to use, either G-iv Ig or plasmapheresis [41].

Paul et al [68] conducted a study to identify clinical variables often associated with the need for mechanical ventilation in patients with GBS. One hundred and thirty-eight GBS patients studied were divided into two groups ventilated (Group 1) and non-ventilated (Group 2). There were 53 patients in Group 1 and 85 in Group 2. Parameters assessed included age, gender, associated illness(es), antecedent events, first symptom at onset, time from onset to bulbar involvement, confinement to bed and peak disability, upper limb power and reflexes at nadir, presence of facial weakness, neck muscle weakness and autonomic dysfunction. Differences between groups were examined by Fisher and t tests. Predictor variables were assessed using logistic regression with backward elimination

Table 2.6: Clinical, pathological and serological features of GBS forms (Adapted from Griffin JW). [41]

Syndrome	Pathology	Previous disease	Related antigen
Acute Inflammatory Demyelinating Polyneuropathy (AIDP)	Demyelination with variable axonal degeneration and lymphocytic inflammation	Herpesvirus, *Campylobacter jejuni*. *Mycoplasma pneumoniae*	14 to 25% have AcIgG Angi GM_1
Fisher syndrome	No data	Occasionally *Campylobacter jejuni*, the majority of cases with unknown history or *Mycoplasma pneumoniae*	$GQ_{1b} and related$ gangliosides
Axonal forms			
Acute Motor Axonal Neuropathy (AMAN)	Motor axonal degeneration with small swelling or demyelination	*Campylobacter jejuni*	$GD_{1a}, GM_1Ga1NAc$, GD_{1a}, GM_{1b}
Acute Motor Sensory Axonal Neuropathy (AMSAN)	Similar with sensitive involucre	*Campylobacter jejuni*	Probably gangliosides
Acute pandysautonomia	Axonal changes and autonomic ganglia	None	Antireceptor acetylcholine in autonomic ganglia

method. Three variables were identified that can independently predict the need for mechanical ventilation: simultaneous onset of motor weakness of upper and lower limbs as the initial symptom; upper limb power grade <3/5 at nadir; and bulbar weakness, which is directly related to the requirement for intubation due to the lack of oral and pharyngeal protection mechanisms in these patients and qualifies as one of the indicators for early intubation in patients with GBS. Furthermore, it was found that the preservation of the reflexes of the upper limbs at the nadir of the condition means a lower probability of the need for assisted ventilation. This study suggests that focusing on clinical signs and history can predict the course of GBS within the first week of onset.

Sundar et al [80] carried out a research to identify clinical and electrodiagnostic predictors of neuromuscular respiratory paralysis in GBS. Forty six patients with GBS were studied for a period of six years. Clinical and electrodiagnostic data were compared between ventilated (28) and non-ventilated (18) patients. The clinical parameters assessed were median age, gender, antecedent infection, prior lung disease, time to peak disability, bifacial weakness, upper limb weakness, bulbar paralysis, neck weakness and autonomic dysfunction. Electrodiagnostic studies included motor nerve conduction studies in 11 ventilated and 13 non-ventilated patients, done prior to maximum disability in each group. Multiple logistic regression analysis was used to compare the two groups. Early progression to peak disability, bulbar dysfunction and autonomic instability predicted the development of neuromuscular respiratory paralysis in GBS. In the case of bulbar dysfunction, it is confirmed the finding made by Paul et al [68] that identifies it as a predictor of the need for assisted ventilation.

Hasan [48] developed a system based on a combined score for predicting respiratory failure in patients with GBS. Eight clinical parameters on fifty GBS patients were assessed: progression of patients to maximum weakness, respiratory rate/minute, breath holding count (the number of digits the patient can count in holding his breath), presence of facial muscle weakness (unilateral or bilateral), presence of weakness of the bulbar muscle, weakness of the neck flexor muscle, and limbs weakness. Each parameter was assigned a certain score depending on their grades, for example, in the case of progression to maximum weakness: if less than three days was assigned 3 points, if it was between three and five days 2 points and if greater than five days 1 point. Each parameter was rated according to clinical standards, for example, in the case of limb weakness was assessed according to the MRC (Medical Research Council) scale. Considering all this a combined scoring system was built. Data were analyzed using statistical techniques such as measures of central tendency and dispersion, as well as statistical independence tests and P values. There was a highly significant statistical association between the development of respiratory failure and the lower grades of (bulbar muscle weakness score, breath holding count scores, neck muscle weakness score, lower limbs and upper limbs weakness score and respiratory rate score) and above 16 total Combined score(p value=0.000). It was concluded in this study that patients with a combined score greater than 16 to 24 points are at serious risk of respiratory failure.

Rajabally and Uncini [71] published a survey on research related to the identification of predictors in GBS. In a study conducted by the French Cooperative Group on Plasma Exchange in GBS [74], by multivariate statistical analysis, five predictors of mechanical ventilation within 30 days of admission were found: time from onset of symptoms to admission less than 7 days, inability to cough, inability to stand, inability to lift the elbows and inability to lift the head. Walgaard et al. [89] used a multivariate logistic regression model for the early prediction of respiratory failure in GBS. They

concluded that Medical Research Council (MRC) Sum Score at admission, number of days between onset of weakness and admission and facial and/or bulbar weakness at admission are all strong predictors of mechanical ventilation in the first week of hospital stay. Hadden et al [43] found, in a test conducted by The Plasma Exchange/Sandoglobulin Guillain-Barre syndrome Trial Group that death or inability to walk at 48 weeks was associated with preceding diarrhoea, severe arm weakness and age >50 years. Visser and coworkers [87] used data of 147 patients who had participated in the Dutch GBS trial comparing intravenous immunoglobulin (IVIG) and plasma exchange (PE) in a study. They found, by multivariate logistic regression analysis, that a previous gastrointestinal illness, age >50 years and MRC Sum Score <40 pretreatment were predictors of a poor outcome. Rajabally and Uncini summarizes the main likely predictors of prognosis in GBS in table 2.7.

Table 2.7: Main likely predictors of prognosis in GBS.

Category	Predictor
Clinical	Age >40 or 50 years
	Reduced vital capacity
	Need for mechanical ventilation
	Preceding diarrhoea
	Low MRC Sum Score at admission
	Low MRC Sum Score at day 7 postadmission
	Short interval between weakness onset and admission
	Facial and/or bulbar weakness
Electrophysiological	Inexcitable nerves
	Low summated distal compound muscle action potential <20% of lower limit of normal
Biological	None of definite value
	More confirmatory studies required

2.4.3 Prognosis

Akbayram et al [1] retrospectively assessed the clinical and laboratory features of 36 children diagnosed with GBS to determine the prognostic factors. The patients were examined for the following parameters: age, sex, personal history, seasonal preponderance, physical examination and laboratory findings, including cerebrospinal fluid (CSF) and electromyography (EMG), need for mechanical ventilation, specific treatment applied and prognosis. Statistical analysis was performed using the commercial program SPSS 17. Mann-Whitney U test was used to compare the data of the groups. Correlation analysis was performed using the Spearman correlation test. The results showed that the most common initial symptoms were limb weakness, which was documented in 34 (94.4%) patients. In this study, 18 patients (51.4%) showed albuminocytological dissociation (raised protein concentration without pleocytosis) on cerebrospinal fluid (CSF) examination. Three patients (8.3%) required mechanical ventilation therapy during hospitalization. In conclusion, researchers' findings showed that cases of GBS was not uncommon in children younger than 2 years of age, and CSF protein level might be found high in the first week of the disease in about one half of the patients.

Patients with axonal involvement showed more severe clinical progression than those with AIDP.

Soysal et al [78] reviewed the medical records of 104 GBS patients obtained in a 10 years period to assess the correlation between the prognosis and epidemiological, clinical, laboratory and electrophysiological findings. All statistical analyses were performed using SPSS. Parametric and non-parametric data were evaluated by one-way ANOVA and chi-square tests, respectively. Pearson correlation test was used for correlation analysis between outcome and epidemiological, clinical and electrophysiological findings. P-values < 0.05 were considered statistically significant. As a result, it was obtained that fifty-five patients reported a preceding infection in a mean period of 13.5 days prior to the disease. The most frequent infection was respiratory infection. The most frequent initial symptom was weakness of legs and/or arms. The total MRC sum scores at admission, during discharge and at first, third, and sixth months were strongly correlated with prognosis. Moreover, first year prognosis was strongly correlated with total MRC sum score during discharge. There was a strong correlation between the clinical subtypes and first month prognosis, and also between respiratory distress and third month prognosis. Researchers concluded that their demographic, clinical and electrophysiological findings were highly similar to those of the previous reports. Two of their cases were presented with preceding tuberculosis infection, which was not reported before in the literature.

Walgaard et al [90] developed a clinical prognostic model for early prediction of outcome in GBS. Data collected prospectively from a derivation cohort of 397 patients with GBS were used to identify risk factors of being unable to walk at 4 weeks, 3 months, and 6 months. Potential predictors of poor outcome (unable to walk unaided) were considered in univariable and multivariable logistic regression models. The clinical model was based on the multivariable logistic regression coefficients of selected predictors and externally validated in an independent cohort of 158 patients with GBS. The result was that high age, preceding diarrhea, and low Medical Research Council sumscore at hospital admission and at 1 week were independently associated with being unable to walk at 4 weeks, 3 months, and 6 months (all p 0.05–0.001). The conclusion of the authors is that a clinical prediction model applicable early in the course of disease accurately predicts the first 6 months outcome in GBS.

Netto et al [65] studied the prognostic factors in a selected cohort of mechanically ventilated GBS patients. A retrospective review of case records of consecutive patients of GBS requiring mechanical ventilation admitted to neurology ICU of a tertiary care center for neurosciences between July 1997 and June 2007 was made. Statistical analysis was performed using SPSS software. Univariate analysis was made using analysis of variance (ANOVA) for continuous variables and Chi-square test for categorical variables. Chi-square test or Fischer's exact test was made to compare 2 groups (dead vs alive and Hughes scale 3 vs >3). The variables which emerged as significant in univariate analysis were used for multivariate analysis using logistic regression analysis to find out which variables were helpful in predicting the outcomes. During the study period, 173 GBS patients were mechanically ventilated. A history of antecedent events was present in 83 (48%) patients: fever (68), diarrhea (16), respiratory infection (13), recent vaccination (2), surgery (1), and others (14 specific infections: varicella zoster, hepatitis, enteric fever). There were 18 deaths (10.4%). The number of days from the onset of the disease to death ranged from 5 to 196 (mean 39 .40.3) days. Logistic regression analysis showed significant regression coefficients for age, pneumonia, and dysautonomia. Researchers concluded that early identification of modifiable risk factors, such as pulmonary involvement, autonomic dysfunction, hypokalemia, sepsis, bleeding, and nutritional complications, may reduce the

mortality and morbidity associated with GBS.

Chapter 3

Descriptive model

The descriptive model constitutes an initial exploratory analysis of our dataset using machine learning techniques. We started with a clustering technique as this method is useful to study the internal structure of data to disclose the existence of groups of homogeneous data. We know in advance of the existence of four GBS subtypes or classes in our dataset. It is our interest to investigate the capability of computational algorithms such as clustering techniques to identify these four real groups using a minimum number of features.

Usually in cluster analysis there is no prior information as a guide to find the exact number of clusters and therefore different partitions with different number of clusters are explored. The ultimate goal is to select the highest quality partition, based on an internal or external metric. In this work, we perform a cluster analysis in a real GBS dataset to identify relevant features that allow to build four clusters, each corresponding to a GBS subtype. The quality of the clusters is measured with an external metric called purity. Purity is an external clustering validation metric that evaluates the quality of a clustering based on the grouping of objects into clusters and comparing this grouping with the ground truth. The highest purity is reached when clusters contain the largest number of elements of the same type and the fewest number of elements of a different type. This metric is fully explained in section 3.1.3.3. Although there are several clustering validation metrics, both internal and external [44], we selected purity since our interest was to find "pure" groups and to take advantage of the available prior knowledge of the true labels.

In this work, we used two different approaches for this model. Both approaches combine a method to select different feature subsets from the original dataset, and a clustering algorithm to build the four homogeneous groups. In both approaches, we used a clustering algorithm called PAM (see section 3.1.3.1 for details). In the first approach, we used filter methods for feature selection. For the second approach, we created a novel method termed QSA-PAM (Quenching Simulated Annealing - Partitions Around Medoids), where QSA is a search algorithm (see section 4.3.1 for details). Details of this method can be found in section 4.3.2.

3.1 Data

The dataset used in this work comprises 129 cases of patients seen at Instituto Nacional de Neurología y Neurocirugía located in Mexico City. Data were collected from 1993 through 2002. There are 20 AIDP cases, 37 AMAN, 59 AMSAN and 13 Miller-Fisher cases. The identification of subtypes was made by a group of neurophysiologists based on the clinical and electrophysiological criteria established in the literature [67, 59]. In this study, each subtype represents a class. Therefore, there are four classes in the dataset. This dataset is not yet publicly available and this is the first time it is used in an experimental study. No public dataset was found to be used as a benchmark.

Originally, the dataset consisted of 365 attributes corresponding to epidemiological data, clinical data, results from two nerve conduction tests, and results from two Cerebrospinal Fluid (CSF) analyses. The complete list of attributes as well as an exploratory study of the dataset can be consulted in [2]. The second nerve conduction test was conducted in 22 patients and the second CSF analysis was conducted in 47 patients only. Therefore, data from these two tests were excluded from the dataset.

The diagnostic criteria for GBS are established in the literature [67, 59]. These formal criteria were considered to determine which variables from the original dataset could be important in the characterization of the four subtypes of GBS. We made a pre-selection of variables based on these criteria. After pre-selection, the dataset was left with 156 variables: 121 variables from the nerve conduction test, 4 variables from the CSF analysis and 31 clinical variables. As for the type of attributes, these are 28 categorical and 128 numeric attributes. The situation of dealing with mixed data types was solved using Gower's similarity coefficient, as previously explained in section 3.1.3.

3.2 Approach 1: Filter methods

We selected filter methods for the initial exploratory study as they are in computational terms the fastest and simplest methods available in the literature for feature selection. Filters work independently from any clustering algorithm and base their decision solely on characteristics of data.

We chose five particular methods based on their performance reported in the literature [97, 62, 69, 61]. Chosen filters apply diverse criteria to evaluate feature relevance. Filters investigated are: CFS, chi-squared, information gain, symmetrical uncertainty and consistency.

3.2.1 Experimental design

We used the 156-feature GBS dataset, described earlier, for experiments. This dataset contains a combination of categorical and numeric features. The Gower's coefficient is able to deal with both types of features when present in the same dataset. We used this method to compute the distance matrix among instances, which is required as input to the PAM algorithm.

As we know beforehand, there are four GBS subtypes present in our dataset. This is why the number of clusters requested to PAM algorithm in our experiments was k = 4. We expected the clustering

algorithm would identify each subtype as a cluster, with the highest purity possible. Five filter methods were used for feature selection, as clustering algorithms perform more efficiently when they work only with relevant attributes [27].

The class attribute was not used when the clustering algorithm was executed. We used it to compute the purity of the clusters obtained with PAM.

A baseline purity using all the 156 features included in the dataset was computed. This value was compared with the purity obtained using only the relevant features as determined by each filter method. Such comparison would allow for a clear view of the benefits of the feature selection process over using the entire dataset, in terms of purity.

Each of the five filter methods selected for experiments in this work was applied to the 156-feature dataset. Along with the features, the class attribute was included in the dataset during the filtering process.

As previously described, CFS and consistency methods include in their output the subset with the most relevant features found. In contrast, chi-squared, information gain and symmetrical uncertainty output a feature ranking.

In all scenarios, new datasets were created with the best feature subsets. The distance matrix of the new datasets was calculated and used as input to the PAM algorithm. Finally, purity of clusters was computed.

In both CFS and consistency methods, the new datasets were created with the resultant most relevant features.

For chi-squared, information gain and symmetrical uncertainty, feature rankings they produced were used to create the new datasets. Datasets with dimension from 2 through 156 were created, with the best two features, the best three features, and so on. The reason for a dataset of dimension 2 is that the calculation of the distance matrix requires at least 2 attributes. The best feature subset was the set of features conforming the dataset which led to the highest purity in the clustering process.

3.2.2 Results

3.2.2.1 Identification of the four GBS subtypes

The baseline purity of the four clusters obtained using all the 156 features included in the dataset was 0.6899. After experimentation, four filter methods found feature subsets which increased the baseline purity after the clustering process. Only the feature subset selected by the consistency method as the most relevant obtained a lower purity of 0.6589 than that of the baseline experiment.

Table 3.1 shows the results of purity of the five methods. Three methods tied with the highest purity (0.7984): information gain, symmetrical uncertainty and CFS. Both information gain and symmetrical uncertainty selected seven relevant features while CFS selected 16 relevant features.

Chi squared chose 41 nerve conduction test variables as the most relevant and reached 0.7829 of purity. The consistency method showed the worst performance, which reached a purity of 0.6589. The six relevant features selected by consistency method were two clinical and four corresponding to the nerve conduction test.

Table 3.1: Results of filter methods ranked on purity.

Method	Number of features	Purity
Information gain	7	0.7984
Symmetrical uncertainty	7	0.7984
CFS	16	0.7984
Chi-squared	41	0.7829
Consistency	6	0.6589

Table 3.2 shows the list of the variables selected by both information gain and symmetrical uncertainty. These variables conformed as a dataset were able to identify the four subtypes of GBS with a purity of 0.7984. All these variables are related to the nerve conduction test.

Table 3.2: List of variables with the highest purity (0.7984) selected by information gain and symmetrical uncertainty.

Feature	Meaning
v105	Amplitude of left ulnar motor nerve
v106 *	Area under the curve of left ulnar motor nerve
v116	Amplitude of right ulnar motor nerve
v172 *	Amplitude of left median sensory nerve
v177 *	Amplitude of right median sensory nerve
v182 *	Amplitude of left ulnar sensory nerve
v187	Amplitude of right ulnar sensory nerve

CFS picked out 16 relevant variables, three of them clinical and 13 corresponding to the nerve conduction test, which reached a purity of 0.7984 as well. The list of 16 variables is shown in Table 3.3.

Four variables from Tables 3.2 and 3.3, denoted (*), were selected by all methods.

Purity results of the clustering process using the datasets formed with the most relevant features as ranked by chi-squared, information gain and symmetrical uncertainty, as described in methodology section, are shown in Figure 3.1. The three methods behave similarly. Both Information gain and symmetrical uncertainty methods reached a maximum value with seven relevant variables, while chi-squared reached its maximum with 41 variables. All three methods kept a purity in the range of 0.7 and 0.8 for feature subsets of sizes between 2 and 102. For bigger subsets, purity lies in the range of 0.65 through 0.7.

Table 3.3: List of variables with the highest purity (0.7984) selected by CFS.

Feature	Meaning
v29	Extraocular muscles involvement
v30	Ptosis
v40	Karnofsky at discharge
v105	Distal amplitude of left ulnar motor nerve
v106 *	Area under the curve of left ulnar motor nerve
v108	Proximal amplitude of left ulnar motor nerve
v111	Average F-wave latency of left ulnar motor nerve
v116	Distal amplitude of right ulnar motor nerve
v134	F-wave amplitude of left tibial motor nerve
v172 *	Amplitude of left median sensory nerve
v173	Area under the curve of left median sensory nerve
v177 *	Amplitude of right median sensory nerve
v182 *	Amplitude of left ulnar sensory nerve
v185	Conduction velocity of right ulnar sensory nerve
v187	Amplitude of right ulnar sensory nerve
v192	Amplitude of left sural sensory nerve

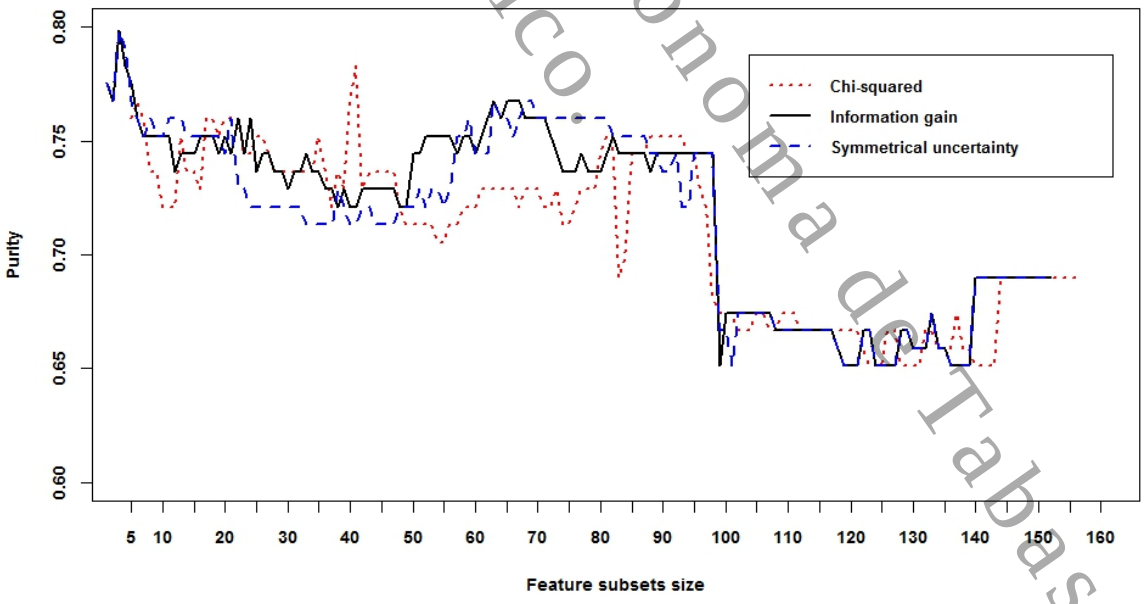

Figure 3.1: Purity reached by the best feature subsets as ranked by chi-squared, information gain and symmetrical uncertainty.

3.2.2.2 Pairwise exploration of the GBS subtypes

In order to investigate if any two pair of GBS subtypes were distinguishable we conducted an additional experiment. We created six new datasets, each one contained instances of only two GBS subtypes. We calculated a baseline purity of each pair of GBS subtypes using all the 156 features. Our goal was to determined a feature subset capable of identifying each pair of GBS subtypes with a higher purity than that of the baseline. We used the five filter methods investigated all along this work to determine the most relevant features for each pair of GBS subtypes. For all scenarios we used $k = 2$, as there are only two GBS subtypes in each dataset. Finally, we applied PAM to form the clusters using only the relevant features determined with each filter method and calculated their purity.

Table 3.4 shows the results of this experiment. Each row represents a pair of GBS subtypes. Columns 2 to 6 represent a filter method. The right-most column indicates the purity achieved using all the features in the dataset, that is, doing no feature selection at all. Table entries indicate the purity obtained in each case. Numbers in bold show the highest purity obtained for each pair of GBS subtype. Based on the purity obtained, it was found that any filter method is better than using all the features. The highest purity for all pairs of GBS subtypes was superior to 0.9. This result demonstrates the effectiveness of filter methods and highlights the importance of feature selection.

Table 3.4: Purity of pairwise clustering of GBS subtypes.

GBS subtypes	IG	SU	CFS	Consistency	Chi-squared	All features
AIDP and AMAN	0.9649 (2)	0.9649 (2)	*	**0.9824 (2)**	0.9649 (2)	0.8771
AMAN and AMSAN	0.927 (3)	0.9375 (2)	0.875 (7)	0.927 (3)	**0.9687 (4)**	0.7395
AIDP and AMSAN	**0.962 (3)**	0.9367 (2)	0.9113 (9)	0.7468 (5)	**0.962 (3)**	0.8354
AIDP and MF	0.8787 (3)	0.8787 (4)	0.8787 (4)	0.8787 (3)	**0.909 (3)**	0.6666
AMAN and MF	**0.98 (6)**	0.96 (3)	**0.98 (4)**	0.74 (2)	**0.98 (6)**	0.96
AMSAN and MF	0.9305 (13)	0.9444 (13)	**0.9583 (14)**	0.8194 (5)	0.9444 (14)	0.8333

IG = Information gain, SU = Symmetrical uncertainty.
* = One feature selected, therefore purity was not computed.
The number of features selected in each case is shown in parenthesis.

3.2.2.3 Exploring different values of k

As explained at the beginning of section 3.1., we performed the clustering process requesting k = 4 clusters as we know this is the number of existing GBS subtypes in the dataset. However, we wanted to explore the clustering process with different values of k. Purity results were analyzed and shown

in Table 3.5.

Table 3.5: Purity for different numbers of clusters using the four GBS subtypes.

k	IG	SU	CFS	Consistency	Chi-squared	All features
2	0.6434 (9)	0.6434 (10)	0.6511 (16)	0.4883 (6)	0.6124 (48)	0.5038
3	0.7906 (7)	0.7906 (7)	0.7286 (16)	0.6666 (6)	0.7829 (6)	0.5813
4	0.7984 (7)	0.7984 (7)	0.7984 (16)	0.6589 (6)	0.7829 (41)	0.6899
5	0.7984 (5)	0.7984 (5)	0.7906 (16)	0.7286 (6)	0.7829 (91)	0.6821
6	0.7751 (4)	0.7751 (7)	0.8139 (16)	0.7596 (6)	0.7751 (38)	0.6666
10	0.8139 (5)	0.8139 (5)	0.8217 (16)	0.7596 (6)	0.8062 (38)	0.6976
20	0.8449 (38)	0.8372 (31)	0.8527 (16)	0.8294 (6)	0.8294 (53)	0.7596

k = Number of clusters, IG = Information gain, SU = Symmetrical uncertainty.
The number of features selected in each case is shown in parenthesis.

The results of this experiment are shown in Table 3.5. The first column represents the different values of k analyzed. Each remaining column represents a filter method. The right-most column represents the purity obtained using all the features, that is, doing no feature selection at all. Each row represents the results obtained for each value of k. Table entries indicate the purity obtained in each case. The results indicate that, in general, purity keeps an ascending pattern as k increases. Purities for k = 4 and k = 5 are very close. In all cases, purity is low for k = 2 and very high for k = 20. The highest purity values were found for k = 20 in all cases; however, these numbers do not indicate that the real number of clusters in the dataset is 20, in fact this number of cluster does not correspond with the nature of GBS subtypes in real life. This result confirms what is reported in literature, higher values of purity are easily obtained for higher values of k [63]. Purity is a good evaluation metric for clustering when the number of clusters is known, as in this case.

3.2.3 Discussion

Our objective in this work was to find the best feature subset to identify four GBS subtypes with the highest purity. We did not find any similar work in the literature, therefore this one represents the first effort in this direction. In order to achieve our purpose, we applied machine learning techniques. We used five filter methods for feature selection and compared their performance.

3.2.3.1 Importance of feature selection to identify GBS subtypes

The clustering of the four GBS subtypes using all the 156 features in the dataset reached a purity of 0.6899. This means that many cases were mislocated in the clustering process. Table 3.1 shows that four of the five feature selection methods used in this work obtained a small feature subset that led to the identification of the four groups with a higher purity than that of the baseline.

The identification of GBS subtypes pairwise was achieved with a high purity. The initial baseline purity was improved in all cases (Table 3.4) when the algorithm used only the relevant features.

These results demonstrate that the clustering algorithm underperforms in the presence of redundant and irrelevant features and highlight the importance of feature selection methods.

3.2.3.2 Analysis of different numbers of clusters

Purity is a good evaluation metric for clustering when the number of clusters is known, as in this case. Higher purity is easily achieved as the number of clusters increases [63] and that is demonstrated with the results shown in Table 6.

3.2.3.3 Identification of four GBS subtypes

The main contribution of this work is the identification of a subset of seven relevant features from a dataset of 156 variables which identified four GBS subtypes with a purity of 0.7984. Another contribution is the analysis of the performance of five filter methods for feature selection. Finally, this work contributes with the feature rankings produced by chi-squared, information gain and symmetrical uncertainty.

A remarkable finding is that all five methods coincided in four variables. It is also noteworthy that only two of the five methods selected clinical variables. It is important to highlight the fact that the consistency method was not able to select a feature subset to improve the baseline purity (0.6899), but instead the six features selected by this method achieved a worse purity (0.6589).

Information gain, symmetrical uncertainty and CFS showed to be highly efficient as they could obtain a reduced subset of relevant features that allow identifying four subtypes of GBS with high purity (0.7984). The first two methods coincided in the same seven variables. CSF selected 16 variables. Further studies are needed to evaluate other methods of feature selection, such as wrapper, embedded and hybrid methods.

3.3 Approach 2: QSA-PAM method

With this approach, we investigate the performance of a more sophisticated method to identify a minimum subset of relevant features to build four clusters, each corresponding to a specific GBS subtype. This novel method consists of a combination of Quenching Simulated Annealing (QSA) and Partitions Around Medoids (PAM), which we named as QSA-PAM method (described in full detail in section 4.3.2). PAM algorithm is used to perform the clustering process and is thoroughly described in section 3.1.3.1. Simulated Annealing (SA) is a general purpose randomized metaheuristic that finds good approximations to the optimal solution for large combinatorial problems. QSA, described in detail below, is a version of SA consisting of two phases: Quenching and Annealing.

3.3.1 Quenching Simulated Annealing (QSA)

In this work, we aim to find a minimum feature subset from a 156-feature real dataset to build four clusters, each corresponding to a specific GBS subtype. This represents a combinatorial problem and to find a good approximation of the best solution the QSA metaheuristic was selected. QSA, shown in **Algorithm 2**, is a version of the SA algorithm which has two phases: Quenching and Annealing. Quenching phase is applied at a very high temperature which is rapidly reduced using an exponential function until a quasi-thermal equilibrium is reached. Quenching phase has shown to allow the exploration of a wider portion of the solution space than previous SA approaches did [36].

Algorithm 2 Quenching Simulated Annealing (QSA).

Require:

Examples $X =< x_1 >, ... < x_m >$
T_i, T_f, $\alpha Quenching$, $\alpha Annealing$, αVLT, τ
Energy function $Energy$ $(.,.)$
Neighbor function $Neighbor$ $(.,.)$

$T \leftarrow T_i$
$Sbest \leftarrow$ random feature subset
while $T > T_f$ **do**
  {Metropolis cycle}
  repeat
    $S_i \leftarrow Neighbor(Sbest, T)$
    $\Delta E \leftarrow Energy(Sbest, X) - Energy(S_i, X)$
    if $\Delta E < 0$ **then**
      $Sbest \leftarrow S_i$
    else
      if $random[0, 1) < \exp(\frac{-\Delta E}{T})$ **then**
        $Sbest \leftarrow S_i$
      end if
    end if
  until thermal equilibrium is reached
  if $\tau > 0$ **then**
    $\gamma \leftarrow 1 - \tau$
    $T_{+1} \leftarrow \gamma * \alpha Quenching * T$ {Quenching phase}
    $\tau \leftarrow \tau^2$
  else
    $T_{+1} \leftarrow \alpha Annealing * T$ {Annealing phase}
  end if
  if $T/Ti < 0.1$ **then**
    $T_{+1} \leftarrow \alpha VLT * T$ {Very low temperatures phase}
  end if
end while
return (Sbest)

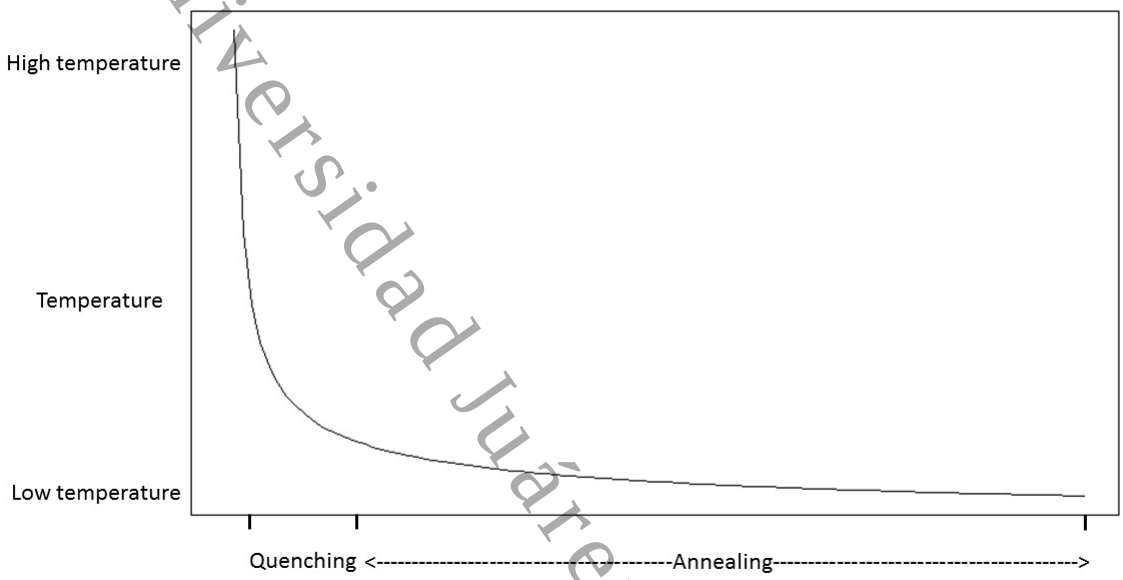

Figure 3.2: Temperature behavior in Quenching and Annealing phases of QSA.

In Annealing phase the temperature is gradually reduced and therefore the exploration is performed at a slower pace, and so the solution space is scrutinized aiming to find the best solution. Figure 3.2 shows the temperature behavior in both Quenching and Annealing phases.

The classical SA algorithm receives as input a set X of m examples with n features. Classical SA requires a triplet composed of an Annealing schedule, an Energy function and a Neighbor function. The Annealing schedule is defined by the initial temperature T_i, the final temperature T_f and the Annealing constant $\alpha Annealing$. These values control the pace and duration of the search. The Energy function evaluates the quality of a solution. The goal of SA is to optimize this function. In this particular problem we used the purity measure as the Energy Function. The Neighbor function receives both the current solution and temperature as input and returns a new candidate solution that is nearby to the current solution, in terms of the variation of features between both solutions.For high values of T a broader search in the dataset is performed to generate S_i. For low values of T the search becomes almost local.

QSA also behaves different with respect to the energy function in Quenching and Annealing phases (Figure 3.3). In Quenching phase, the energy function undergoes high fluctuations as QSA is still exploring the search space. As Annealing phase advances, the fluctuations in the energy function decrease as in this phase QSA is now looking for the best solution. At the end of Annealing, the energy function in fact remains stable.

QSA uses two extra parameters: a Quenching constant $\alpha Quenching$ and τ. These parameters work

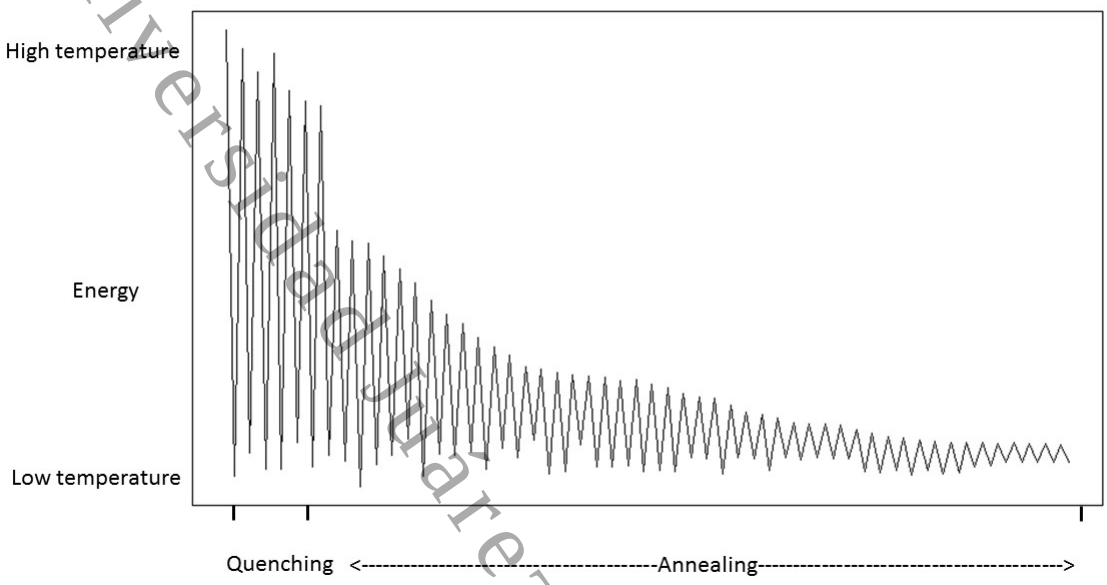

Figure 3.3: Energy behavior in Quenching and Annealing phases of QSA.

together to reduce the temperature at a rapid pace until τ converges to 0.

In addition, this particular implementation uses a cooling constant αVLT at very low temperatures, which enables the search space to be meticulously traversed, which corresponds to performing a local search.

QSA, shown in **algorithm 2**, begins initializing the value of T (line 1). A random feature subset is taken as an initial best solution *Sbest* (line 2). An external cycle is performed for controlling temperature variations until T_f (line 3) is reached. The Metropolys cycle (nested loop of lines 5-15) is performed until "thermal equilibrium" is reached. In this cycle, a neighbor solution S_i is generated using *Sbest* and the value of T (line 6). The Neighbor function draws a new state S_i based on the previous state *Sbest* where a number of features in *Sbest* is replaced with new features randomly taken from the original dataset depending on the current temperature T. For high values of T, a maximum of 80% of the features are replaced. For low values of T, a maximum of 20% of them are replaced. The difference in energy ΔE between *Sbest* and S_i is calculated (line 7). If $\Delta E < 0$, that is S_i is a better solution than *Sbest* (line 8), then S_i becomes the best solution (line 9). If $\Delta E >= 0$, it means S_i is a worse solution than *Sbest*, then the best solution *Sbest* is replaced with S_i only if a random number generated in the range $[0, 1)$ is less than $\exp(\frac{-\Delta E}{T}))$ (lines 11-12). This step is fundamental in SA since it allows to escape from local minima by accepting worse solutions than the current one. When the Metropolis cycle is finished, the temperature new T is updated (lines 16-22). If $\tau > 0$ the Quenching phase is performed; on this phase T will be rapidly reduced due to the use of $\alpha Quenching$ constant. After a few iterations of the external cycle (line 3), τ converges

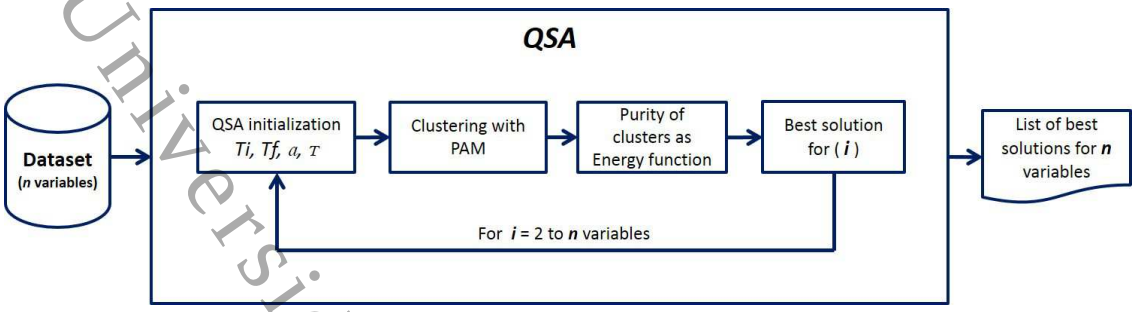

Figure 3.4: QSA-PAM Method.

to 0 (lines 19-21). Right after that, the Annealing phase is performed; on this phase T is reduced at a slower pace (line 17). As stated before, this particular implementation of QSA uses a αVLT constant to perform local searches at very low temperatures (for $T/Ti < 0.1$) (lines 23-25). When T reaches T_f, QSA algorithm returns the feature subset that optimizes the energy function (line 27).

3.3.2 Description of QSA-PAM method

The method we introduced in this work is a combination of two methods: Quenching Simulated Annealing (QSA) and Partitions Around Medoids (PAM), named QSA-PAM method (Figure 4.3). This is a novel method in machine learning implemented to explore a large search space for a feature subset that leads to a good solution for a clustering problem. In our work, QSA-PAM was used to solve the problem of building four clusters with the highest purity possible, each cluster corresponding to a specific GBS subtype - AIDP, AMAN, AMSAN and Miller-Fisher. The method is described below.

QSA-PAM receives a dataset with n features as input. In this particular implementation $n = 156$. The class attribute was excluded as required in a clustering problem. We used it as the ground truth to compute the purity of the clusters obtained with PAM. For all experiments, the number of clusters requested to PAM algorithm was $k = 4$, as there are four GBS subtypes present in the dataset.

The method works as follows (see Figure 4.3): a cycle from $i = 2$ to n is performed. For each value of i, QSA finds the best feature subset containing up to i features that identify four clusters with the highest purity (e.g. for i equals 25 the best feature subset could be of size 18). At each iteration, different feature subsets (size varying from 2 to i) from the entire dataset are chosen by QSA algorithm. New datasets are created with each of these feature subsets. A distance matrix between instances from these new datasets is computed using Gower's similarity coefficient which is used as input to PAM to build four clusters. The calculation of the distance matrix requires at least 2 attributes, and this is the reason for using a minimum value of $i = 2$ in the main loop (For $i = 2$ to n variables). Finally, the purity of the clusters is computed as it is used as the energy function in QSA. QSA finds the feature subset that optimizes the energy function for each value of i. The aim of having QSA selecting the best features for each value of i is to look into the number of times such variables get selected, this would give us insight on their relevance. That is, the more times a variable is selected the more relevant it is.

As a result of the method we obtained a list of feature subsets that build four clusters with the highest purity possible for each value of i, each corresponding to a GBS subtype. In other words, the list contains the best 2-feature subset, the best 3-feature subset, and so on up to the best 156-feature subset.

In this study we perform a number of runs of QSA-PAM. For each run, we set a different seed. These seeds were generated using Mersenne-Twister pseudo random number generator [64]. Using a different seed for each run of QSA-PAM allows for QSA part of the method to generate different combinations of feature subsets from the search space to choose from.

3.3.3 Results

3.3.3.1 Determination of QSA parameters

For tuning the proposed algorithm, we started determining the initial and final temperatures (T_i) and (T_f) respectively used by QSA part of our method. These temperatures are computed as follows [35]:

$$T_i = \frac{-\Delta Zmax}{ln(P(\Delta Zmax))} \tag{3.1}$$

$$T_f = \frac{-\Delta Zmin}{ln(P(\Delta Zmin))} \tag{3.2}$$

where

$\Delta Zmax$ and $\Delta Zmin$ are the maximal and the minimal deterioration respectively, both of the energy function. Deterioration refers to changes in the Energy function, that is, the changes in purity at a extremely high temperatures and extremely low temperatures for $\Delta Zmax$ and $\Delta Zmin$ respectively.

$P(\Delta Zmax)$ and $P(\Delta Zmin)$ are the probabilities of accepting a new solution with $\Delta Zmax$ and $\Delta Zmin$ respectively. The probability of accepting a new solution is close to 1 at extremely high temperatures and close to 0 at extremely low temperatures [35].

Table 3.6: Average purity over 30 runs of QSA-PAM using three different temperature configurations.

T_i	T_f	**Average best solution (purity) after 30 runs of QSA-PAM**
1.5×10^{6}	1×10^{-3}	0.7509
1.376999×10^{6}	7×10^{-4}	0.7564
1×10^{6}	1×10^{-5}	0.7881

To determine $\Delta Zmax$ and $\Delta Zmin$, 30 runs of QSA-PAM were performed using random samples from the 156-feature dataset. Different arbitrary temperature values, in the range [7.674419×10^{6}, 1×10^{-5}] were used. An average value of 0.1372 was obtained for $\Delta Zmax$ and 0.0088 for $\Delta Zmin$. Using both equations in 3.1 and 3.2 led us to temperatures of 1.376999×10^{6} for T_i and 7×10^{-4} for T_f. We used these values as approximations to the final temperatures. To verify the quality of these

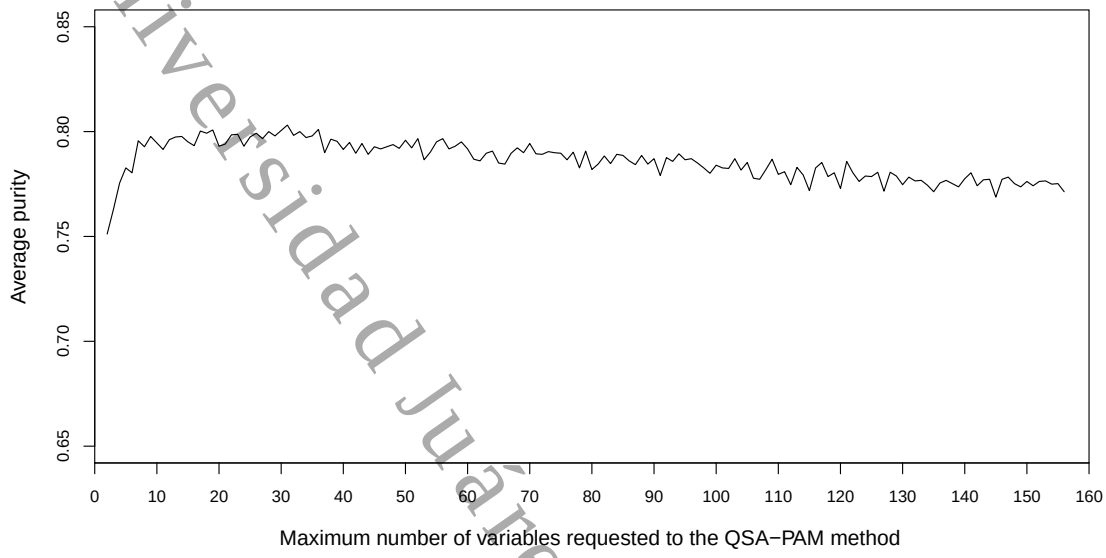

Figure 3.5: Average purity for each maximum number of variables requested to the method across 30 runs.

values, three different combinations of T_i and T_f were used and compared in 30 runs of QSA-PAM: the values obtained with equations 3.1 and 3.2 ($T_i = 1.376999\times10^6$ and $T_f = 7\times10^{-4}$), an arbitrary higher temperature combination ($T_i = 1.5\times10^6$ and $T_f = 1\times10^{-3}$) and another arbitrary lower temperature combination ($T_i = 1\times10^6$ and $T_f = 1\times10^{-5}$). The purity values obtained across 30 runs were averaged for every temperature combination. Results are shown in Table 3.6.

Values of α and τ were assigned according to values typically used in literature [35, 37, 38]. Values for QSA parameters are shown in Table 3.7.

Table 3.7: Final values for QSA parameters.

Parameter	Value
T_i	1×10^6
T_f	1×10^{-5}
αQuenching	0.7
αAnnealing	0.985
αVLT	0.99
τ	0.9

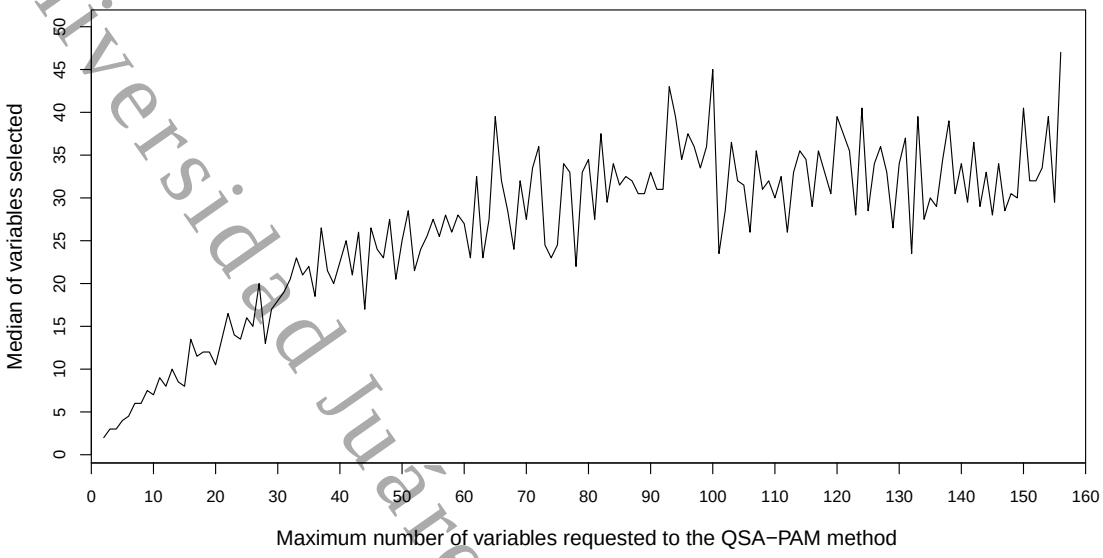

Figure 3.6: Median of the number of features selected by the method across 30 runs.

3.3.4 Experimentation

3.3.4.1 Finding clusters using the 156-feature dataset

The aim of the initial experiments was to analyze the behavior of clusters using the whole 156-feature dataset. In order to achieve our goal, QSA-PAM was run 30 times using the 156-feature dataset, each run with a different seed. In each run, a list of the best feature subset for i equals 2 to 156 was obtained. Then, the purity of clusters obtained using each of the feature subsets was calculated. Finally, the average purity of clusters over 30 runs for each value of i was computed. The results are shown in Figure 3.5. The chart shows an initial ascending behavior, a peak is reached approximately at i equals 30 and then the chart begins to slowly descend. The average purity values ranged in [0.75, 0.80]. We want to highlight that two feature subsets in the feature space over 30 runs reached a maximum purity of 0.8527. Also, by looking at Figure 3.5 we identified the need of determining a proper number of features to be used to create clusters of high purity.

3.3.4.2 Analysis of the size of feature subsets selected by QSA-PAM

With the aim of determining the proper number of features selected by QSA-PAM, we further analyzed the size of the resultant feature subsets. QSA-PAM selects the best feature subset that maximizes the purity of four clusters for each value of i ranging in [2,156]. The size of these feature subsets would not necessarily match the value of i. That is, for i equals 38 the size of the best feature subset could be 12 in one run, and 23 in another run. We were interested in investigating the median of the best feature subsets for each value of i over 30 runs. To achieve this, we counted the number of features selected by QSA-PAM in each run for every value of i and then we calculated

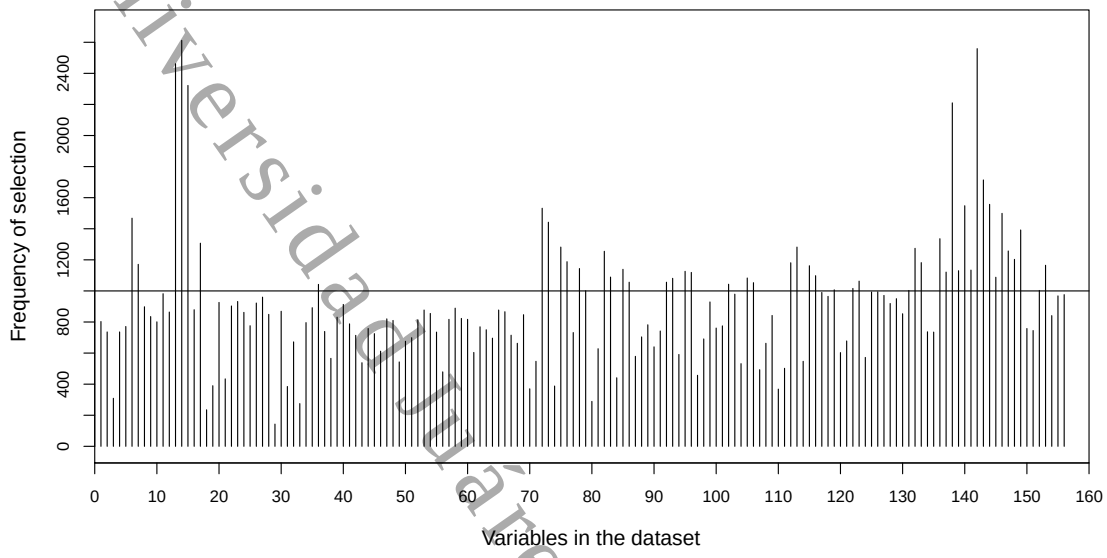

Figure 3.7: Total frequencies of selection per feature across 30 runs.

its median. The results are shown in Figure 3.6. Overall, the size median of feature subsets selected by QSA-PAM was less than 50 even for cases where a number greater than 100 was requested.

3.3.4.3 Analysis of the frequencies of features selected by QSA-PAM

We also investigated the total frequency of selection by QSA-PAM for each feature in the dataset along the 30 runs, as shown in Figure 3.7. A ranking for all the 156 features was created based on these frequencies. That is, the feature on top of this ranking is the most frequently selected feature. The highest frequency possible for any feature over 30 runs was 4650, that is 155 x 30 (155 total features, 30 runs). The highest frequency obtained for a feature was 2612. Five features stood out of all the 156 features reaching a frequency greater than 2000. These features were 14, 142, 13, 15 and 138 as shown in the chart. We considered to use for further experiments only features with a frequency over 1000. This number of features counted to fifty. Coincidentally, fifty was the threshold derived from the analysis described in section 3.2.2.

3.3.4.4 Quality of clusters formed using feature subsets based on the ranking of frequencies

Based on the ranking of features according to frequencies for all 156 features, we created new datasets using the best 2 features, the best 3 features, and so on, up to 156 features. For each of these datasets, a distance matrix was computed using Gower's similarity coefficient. Each matrix was used as input to PAM to build four clusters. Finally, the purity of clusters was computed. The results of this experiment are shown in Figure 3.8. The highest purity obtained with these clusters

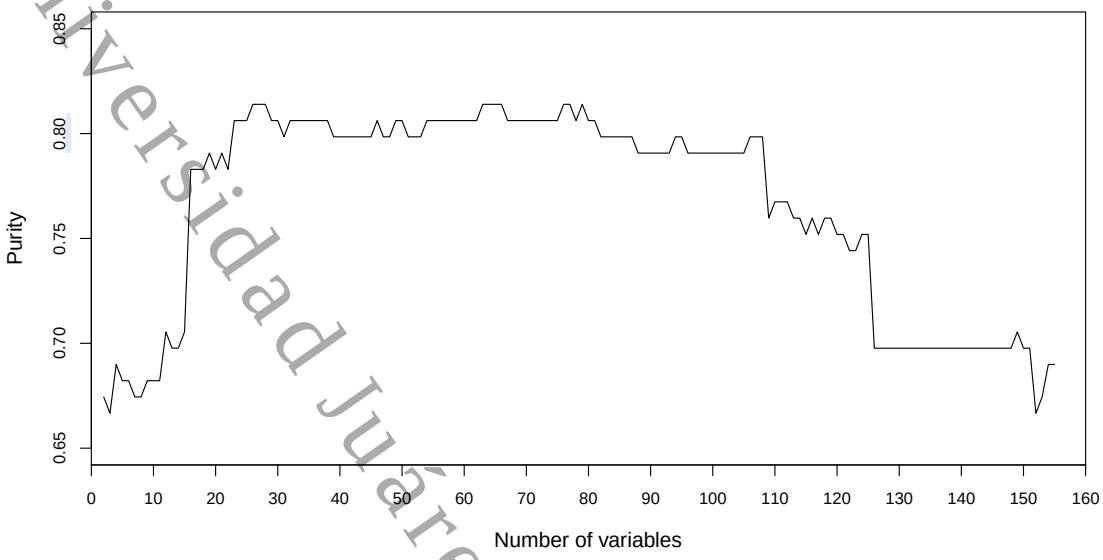

Figure 3.8: Purity of clusters built using variables as ranked on the frequencies.

was 0.81. This showed that not necessarily a sequential grouping of features based on the ranking of features would lead us to the desired purity of 1. Two other facts observed is that a low purity of 0.7054 is reached with few features (<=15) which is similar to that obtained (0.6976) with a large number of features (>=125).

3.3.4.5 Finding clusters using the 50 most frequently selected features

We were interested in searching for a feature subset that improved the highest purity obtained (0.8527) when QSA-PAM was ran using all the 156 features. To achieve this, we created a dataset using the 50 most frequently selected features. We performed a second experiment of 30 runs of QSA-PAM, for i ranging in [2,50]. Again, purity of clusters obtained using each feature subset was calculated. Then, we calculated the average purity for each value of i. The results are shown in figure 3.9. Again, we can observe that with a few features (<=10) the lowest average purity is obtained. This chart shows an initial ascending behaviour until i equals 10 approximately and then it stabilizes at approximately a value of purity equals 0.85 and it remains stable until the end.

3.3.4.6 Our best 16-feature subset

In the second experiment of 30 runs, described in section 3.2.5, we found a feature subset that built four clusters with a purity of 0.8992. This subset contains 16 features: four clinical features and 12 features from the nerve conduction test. These features are listed in table 3.8. We will further investigate this feature subset to create a predictive model.

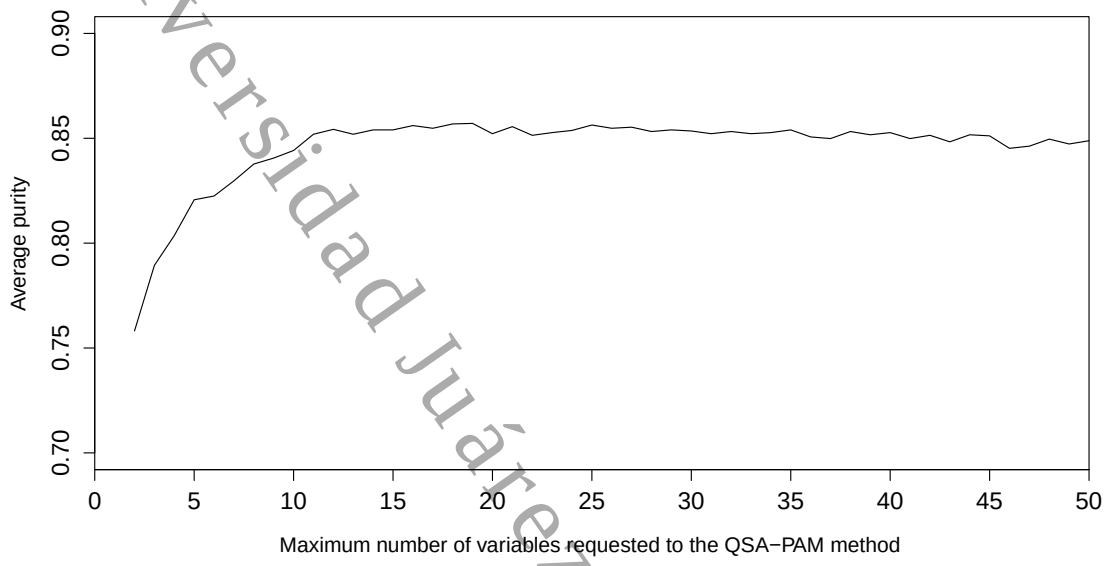

Figure 3.9: Average purity reached with the best 50 features according to frequency.

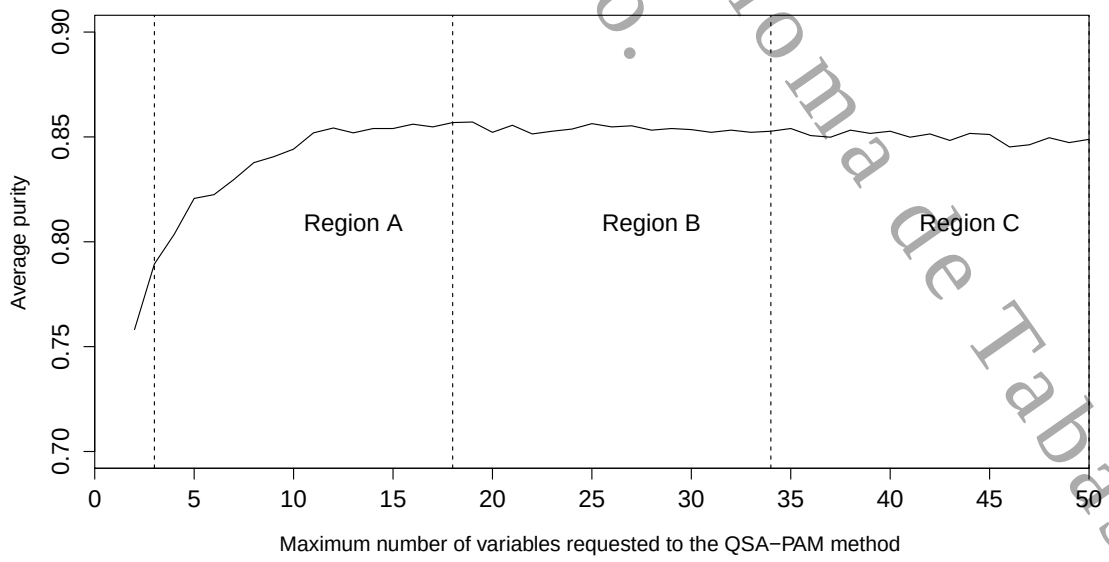

Figure 3.10: Regions of average purity reached with the best 50 features.

Table 3.8: List of features that built 4 clusters with a purity of 0.8992.

Feature label	Feature name
v22	Symmetry (in weakness)
v29	Extraocular muscles involvement
v30	Ptosis
v31	Cerebellar involvement
v63	Amplitude of left median motor nerve
v106	Area under the curve of left ulnar motor nerve
v120	Area under the curve of right ulnar motor nerve
v130	Amplitude of left tibial motor nerve
v141	Amplitude of right tibial motor nerve
v161	Area under the curve of right peroneal motor nerve
v172	Amplitude of left median sensory nerve
v177	Amplitude of right median sensory nerve
v178	Area under the curve of right median sensory nerve
v186	Latency of right ulnar sensory nerve
v187	Amplitude of right ulnar sensory nerve
v198	Area under the curve of right sural sensory nerve

3.3.4.7 Statistical analysis

We were interested in finding a region in the chart of Figure 3.9 where QSA-PAM could have achieved the highest average purity values. That is, we looked at diverse ranges of variables requested to QSA-PAM aiming to find a group of four clusters with the highest quality. We divided the plot into three regions (Figure 3.10). The first region (group A) for values of i from 3 to 18, the second region for values from 19 to 34 (Group B), and for values from 35 to 50 (Group C). The first value of i was not taken into account. These 3 groups consisted of 16 values of purity each one. We applied the non-parametric Friedman test using Holms correction to find statistical significant differences between purities of the three groups. The test was applied pair-wise. We took as the null hypothesis (Ho) = purity levels between two regions are the same, and as the alternative hypothesis (Ha) = purity levels between two regions are different. The results are shown in Table 3.9.

Table 3.9: Friedman-Holm test for regions A, B and C.

Regions compared	**p Value**	**Ho**
A - B	0.317	Accepted
B - C	0.000	Rejected
A - C	1.000	Accepted

According to the results shown in Table 3.9, we found statistical significant differences between purities achieved across the three regions shown in Figure 3.10. It was found a statistical significant difference between regions B and C. A more extensive search in these regions could lead to a better result. That is, requesting a number of variables to QSA-PAM between 19 and 34, or between 35

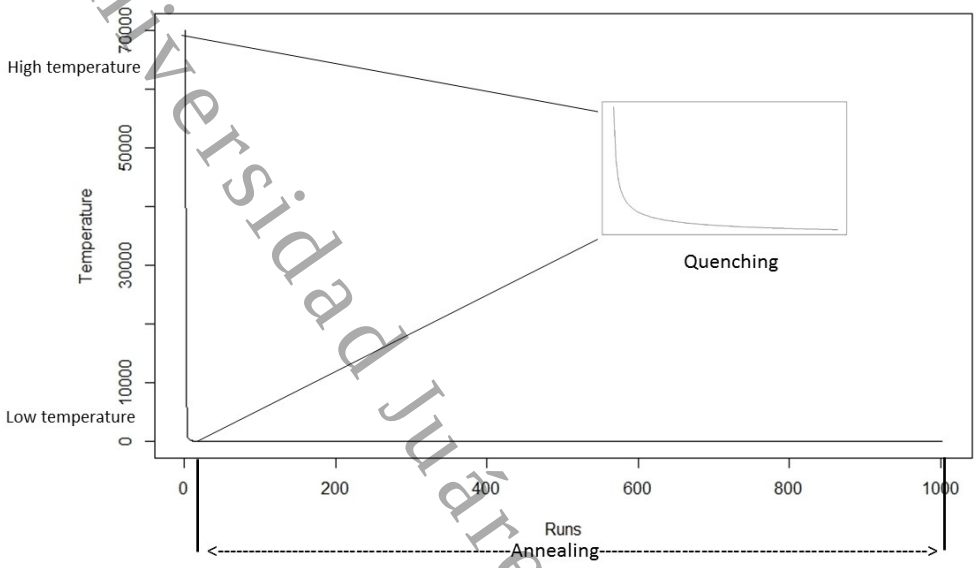

Figure 3.11: Temperature behavior in Quenching and Annealing phases of QSA-PAM for i = 50.

and 50.

3.3.4.8 Temperature and energy behavior in QSA

We show in this section the temperature and energy behavior of QSA part of the QSA-PAM method for a single arbitrary value of i, that is for $i = 50$. For this value of i, the number of runs was 1002. Figure 3.11 shows the temperature behavior of QSA in both Quenching and Annealing phase. The quenching phase showed a rapid exponential decrease in temperature as described in the QSA section (3.3.1.). In the annealing phase, the temperature decreases slowly, as QSA searches locally in the dataset.

Figure 3.12 shows the energy behavior of QSA, the purity values in this case. Being this a maximization problem, the energy behaves in an opposite way of that of the typical QSA problems. In the quenching phase, the energy had high fluctuations as it was expected. In this experiment, the fluctuations in energy in the quenching phase were between 0.48 and 0.75 approximately. These fluctuations continued in a number of runs of the annealing phase. From run 400 approximately, these changes in energy began to decrease to levels of purity of 0.62 to 0.72. From run 500 to the final run, there were three long phases of stability in the energy function. This stability are the result of QSA accepting best solutions than the current one with higher probabilities as the temperature reaches T_f. The final value of energy (purity) was 0.81.

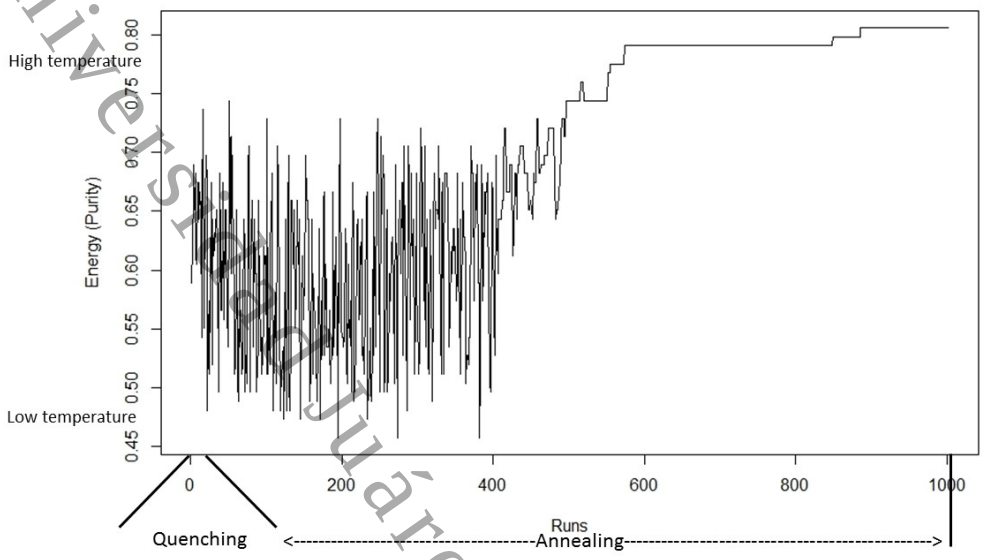

Figure 3.12: Energy behavior in Quenching and Annealing phases of QSA-PAM for i = 50.

3.3.5 Discussion

In this study we aimed to find a minimum feature subset to build four clusters, each corresponding to a GBS subtype, with the highest purity possible. In order to achieve our goal we introduced a combination of a metaheuristic (QSA) and a clustering algorithm (PAM), which we named QSA-PAM method.

As we explained in section 3.3.4, we created a ranking of all the 156 features based on the frequency of selection of each feature by QSA-PAM across 30 runs performed in an initial experiment. Using the 50 most frequently selected features, we created a new dataset to perform a final experiment consisting of 30 runs of QSA-PAM. We obtained a 16-feature subset able to build four clusters with a purity of 0.8992. In further studies, a more formal method could be used to decide on the number of most frequently selected features and use it as threshold instead of using 50. This could lead to obtain a feature subset that build clusters from GBS data with a higher purity.

The final feature subset identified in this study includes a combination of clinical and nerve conduction features. As to the four clinical features, they are reported in the literature as common clinical characteristics of GBS subtypes. This confirm the ability of QSA-PAM to select relevant clinical features for the identification of clusters. Concerning the electrophysiological features, further studies are required to confirm these findings. The identification of a minimum nerve conduction feature subset would be very useful to design a fast electrodiagnostic test.

On the other hand, this finding leads to the research question of the possibility of distinguishing the GBS subtypes investigated in this study with only one type of feature by itself, either clinical or nerve conduction feature. Specifically, a study to investigate the ability of clinical features to

identify GBS subtypes is recommended. Wakerley et. al. [88] identified a set of clinical diagnostic features to classify GBS subtypes that could be used in the proposed research.

An important contribution of this study is a ranking of 156 features related with GBS. This ranking was created according to the frequency of selection of each feature by QSA-PAM across 30 runs. This ranking may be used in further studies.

With QSA-PAM, we found a 16-feature subset able to build four clusters present in our dataset with a purity of 0.8992. This finding, although able to be improved, shows the effectiveness of the method to find good solutions to clustering tasks involving feature selection.

Chapter 4

Predictive model

The aim of this phase is to create the highest accurate predictive model for GBS possible, using the 16 relevant features described in chapter 4. Again, to create this model we used two approaches: single classifiers and ensemble methods. We expect to reach the highest classification results with the latter.

4.1 Data

In the predictive model, we investigate the 16 relevant features, identified in the previous phase, to classify GBS subtypes. The features are listed in Table 4.1. The first four features are all clinical and the remaining features come from a nerve conduction test.

Table 4.1: List of features used in this study.

Feature label	Feature name
v22	Symmetry (in weakness)
v29	Extraocular muscles involvement
v30	Ptosis
v31	Cerebellar involvement
v63	Amplitude of left median motor nerve
v106	Area under the curve of left ulnar motor nerve
v120	Area under the curve of right ulnar motor nerve
v130	Amplitude of left tibial motor nerve
v141	Amplitude of right tibial motor nerve
v161	Area under the curve of right peroneal motor nerve
v172	Amplitude of left median sensory nerve
v177	Amplitude of right median sensory nerve
v178	Area under the curve of right median sensory nerve
v186	Latency of right ulnar sensory nerve
v187	Amplitude of right ulnar sensory nerve
v198	Area under the curve of right sural sensory nerve

4.2 Classification scenarios

4.2.1 Multiclass classification

In this scenario, we classified the four GBS subtypes at the same time, that is AIDP, AMAN, AMSAN and MF. Each classifier tackles the multiclass classification problem particularly. Some of them, like decision trees, intrinsically support multiclass classification. Others, like SVM, tackle multiclass classification using either OVA (One vs All) or OVO (One vs One) strategy. The implementation of these strategies were out-of-the-box in the R versions of the classifiers we used in this study.

4.2.2 One vs All classification (OVA)

In OVA classification, we classified each class against the rest of the classes. That is, AIDP vs ALL, and so on. We aimed to investigate how well classifiers distinguish each class with respect to the other classes. OVA strategy consists of building n binary classifiers. In this particular work, we made 4 different OVA classifications.

4.2.3 One vs One classification (OVO)

In OVO classification, we classified one class against another class, as many combinations as possible. That is, AIDP vs AMAN, and so on. We aimed to investigate how well classifiers distinguish each pair of classes. OVO strategy consists of building $n(n-1)/2$ binary classifiers. In this particular work, we made 6 different OVO classifications.

4.3 Model evaluation scenarios

4.3.1 Train-Test

5 Train-test consists of separating the original complete dataset into two independent new datasets. These two datasets contain both the predictor features and the outcome variable. The first dataset, named train, used for a classification algorithm to discover relationships or patterns among data, that is, to "train" or to fit a model. The second dataset, named test, used for the model fitted to estimate its performance on completely unseen data. In this study we used two-thirds of the original dataset for train and one-third for test.

4.3.2 Cross-validation

5 Cross-validation divides the original complete dataset into k independent new datasets (k folds). Then, it performs k loops in which $k-1$ partitions of the original dataset is used for training and the rest for testing. For each fold, the measure of evaluation of the model obtained from the confusion matrix is calculated and summed. When all the k loops have ended, the cross-validation accuracy is obtained. In this study, we use a 10-fold cross-validation (10-FCV).

4.4 Approach 1: Single classifiers

4 In this first approach, we used single classifiers. We selected 14 single classifiers from diverse nature: decision trees (C4.5), instance-based learners (kNN), kernel-based (SVM), neural networks (SLP,

MLP, RBF-DDA), rule induction learners (OneR, JRip), among others. The complete list is in section 3.1.5. From each type, we selected the best classifiers and compared their performance in three types of experiments: four GBS subtype classification, OVA classification and OVO classification.

4.4.1 Experimental design

We used the 16-feature subset, described in section 5.1, for experiments. We added the class variable to this subset, that is, the GBS subtype. Finally, we created a dataset containing the 129 instances and 17 features. As mentioned in section 4.1, our dataset has 4 classes, identified with numbers 1 to 4, where 1 is AIDP, 2 is AMAN, 3 is AMSAN, and 4 is MF.

We used two model evaluation schemes: train-test and 10-FCV. For each of these schemes, we performed 30 runs where we applied each of the classifiers described in section 3.1.5. In each run, we set a different seed. This warranties to have each classifier train on the same train set at the same run number. The same seeds were used for each classifier. These seeds were generated using Mersenne-Twister pseudo-random number generator [64]. The use of a different seed for each run ensures producing different splits of train and test sets in both evaluation schemes.

4.4.1.1 Four GBS subtype classification

In this classification, the four GBS subtypes were included in the dataset, that is AIDP, AMAN, AMSAN, and MF. In this scenario, the base metric was the average accuracy.

4.4.1.2 OVA classification

For OVA classification, we derived as many new datasets as there are GBS subtypes in the original dataset. That is, four datasets were derived. In each one, the instances of one class were marked as the positive cases and the instances of the remaining classes were marked as the negative cases. In this scenario, the base metric was the balanced accuracy.

4.4.1.3 OVO classification

For OVO classification we created six new datasets, as many combinations of pairs of GBS subtypes. Each dataset contained instances of only two GBS subtypes, one class marked as the positive case and the other class as the negative case. In this scenario, the base metric was the balanced accuracy.

4.4.1.4 Train-test

For each run, we computed accuracy, sensitivity, specificity, Kappa statistic and multiclass AUC. Finally, we averaged each of these quantities across the 30 runs.

4.4.1.5 10-FCV

For each run, we performed a 10-FCV. For each fold, we computed accuracy, sensitivity, specificity, Kappa statistic and multiclass AUC. After the 10 folds, we calculated 10-FCV accuracy and the average of each of the other measures. Finally, we averaged each of these quantities across the 30 runs.

4.4.2 Parameter optimization

4.4.2.1 kNN parameter optimization

kNN parameter optimization was performed using the four GBS subtypes. kNN requires optimization for d and k values. We tried values for d in the range 1 to 3 inclusive. For each d, we used values of k from 5 to 35. We analyzed the performance of kNN for each value of d across 30 runs of both 10-FCV and train-test processes with different seeds each. We calculated the accuracy of each run and the average accuracy over 30 runs. We selected the combination of d and k that obtained the highest accuracy.

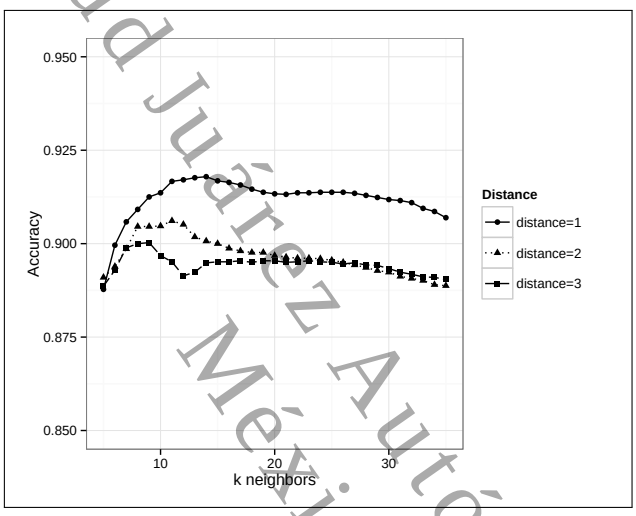

Figure 4.1: Tuning process of number of neighbors k and distance d for kNN using 10-FCV.

Figure 4.1 shows the results of kNN parameter optimization for different values of d and k, using 10-FCV. The chart shows the accuracy for $d = 1$, $d = 2$, and $d = 3$. Overall, kNN achieved the best results with $d = 1$. The highest accuracy obtained was 0.9179 for $k = 14$ and $d = 1$. These values of k and d were used for the classification experiments with kNN, using 10-FCV, including four GBS subtype classification as well as OVO and OVA classification.

Figure 4.2 shows the results of kNN parameter optimization for different values of d and k, using train-test. The chart shows the accuracy for $d = 1$, $d = 2$, and $d = 3$. Overall, kNN achieved the best results with $d = 1$. The highest accuracy obtained was 0.9268 for $k = 18$ and $d = 1$. These values of k and d were used for the classification experiments with kNN, using train-test, including four GBS subtype classification as well as OVO and OVA classification.

4.4.2.2 SVM parameter optimization

SVM parameter optimization was performed using the four GBS subtypes. An effective and simple method of tuning C and σ simultaneously is given by Hsu et al [55]. This method consists of an exhaustive grid search using growing values for C and σ. The C values used in this work for all kernels were 1, 10, 50, 80 and 100. The σ (γ in Laplacian kernel implementation) values were

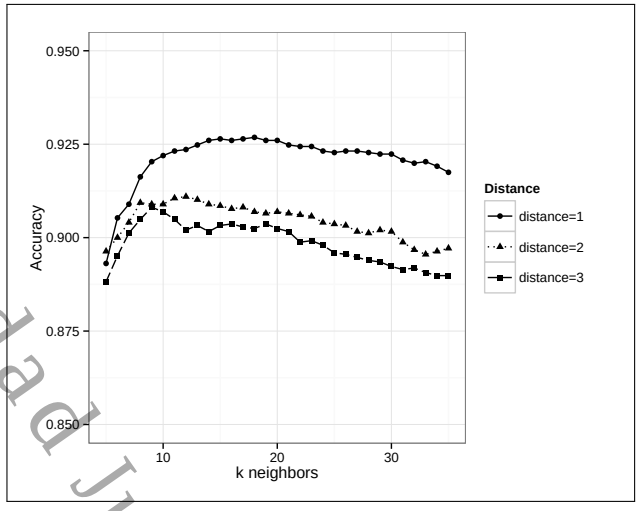

Figure 4.2: Tuning process of number of neighbors k and distance d for kNN using train-test.

exponentially growing values from 0.001 to 100. The tuning procedure for the rest of Polynomial parameters was to try different values for degree (from 2 to 10), and 0-1 for coef parameter. We analyzed the behavior of accuracies across polynomial degrees. All these parameter values are found typical in the literature.

For each combination of parameter values, we performed 30 SVM runs of both 10-FCV and train-test processes with different seeds each. We calculated the accuracy of each run and the average accuracy over 30 runs. We selected the combination of parameters that obtained the highest accuracy.

The results of the linear kernel optimization for both train-test and 10-FCV scenarios are shown in Table 4.2.

Table 4.2: Linear kernel optimization. The units of the results are shown in average accuracy over 30 runs. The highest value is shown in bold.

C	train-test	10-FCV
1	**0.9175**	**0.9069**
10	0.8965	0.8890
50	0.8969	0.8890
80	0.8969	0.8890
100	0.8969	0.8890

The results of the polynomial kernel optimization for both train-test and 10-FCV scenarios are shown in Table 4.3. We show only the best results obtained for each degree.

In Figure 4.3, we show the behavior of accuracies for each degree in polynomial kernel for both 10-FCV and train-test scenarios. As shown in Figure 4.3, accuracies grow as degrees increase until reaching a maximum accuracy of 0.9235 (train-test) and 0.9175 (10-FCV) at degree equal 6. For

Table 4.3: Polynomial kernel optimization. The units of the results are shown in average accuracy over 30 runs. The highest value is shown in bold.

		train-test			10-FCV		
degree	coef	γ	C	accuracy	γ	C	accuracy
2	1	0.001	50	0.9110	0.01	10	0.9178
3	1	0.001	100	0.9110	0.01	10	0.9176
4	1	0.001	80	0.9126	0.001	100	0.9175
5	1	0.01	1	0.9118	0.01	1	0.9213
6	1	0.01	1	**0.9142**	0.01	1	**0.9235**
7	1	0.01	1	0.9126	0.01	1	0.9232
8	1	0.001	50	0.9122	0.001	80	0.9218
9	1	0.001	10	0.9114	0.001	80	0.9214
10	1	0.001	10	0.9110	0.001	50	0.9194

degrees 6 and higher, accuracies decrease. The optimal parameters for both train-test and 10-FCV scenarios were degree = 6, coef = 1, C = 1 and $\gamma = 0.01$.

The results of SVM parameter optimization for Gaussian kernel are shown in Table 4.4. The first row shows C values. The first column represents γ values. Each table entry is the accuracy averaged across 30 runs using both train-test and 10-FCV, as described in section 5.4.1. The optimal parameters for both train-test and 10-FCV scenarios were C = 10 and $\gamma = 0.001$.

Table 4.4: Gaussian kernel optimization. The units of the results are shown in average accuracy over 30 runs. The highest value is shown in bold.

	train-test					**10-FCV**				
	C					**C**				
γ	**1**	**10**	**50**	**80**	**100**	**1**	**10**	**50**	**80**	**100**
0.001	0.733	0.887	0.911	0.910	0.909	0.736	0.885	0.913	0.915	0.915
0.01	0.885	**0.913**	0.904	0.902	0.903	0.882	**0.919**	0.915	0.910	0.912
0.1	0.910	0.900	0.896	0.896	0.896	0.910	0.895	0.893	0.893	0.893
1	0.809	0.814	0.814	0.814	0.814	0.799	0.809	0.809	0.809	0.809
10	0.732	0.734	0.734	0.734	0.734	0.726	0.729	0.729	0.729	0.729
100	0.732	0.732	0.732	0.732	0.732	0.727	0.727	0.727	0.727	0.727

The results of SVM parameter optimization for Laplacian kernel are shown in Table 4.5. The optimal parameters for both train-test and 10-FCV scenarios were C = 50 and $\sigma = 0.01$.

The optimal parameters for each kernel in train-test and 10-FCV scenarios were used for further classification experiments with SVM, including four GBS subtype classification as well as OVO and OVA classification.

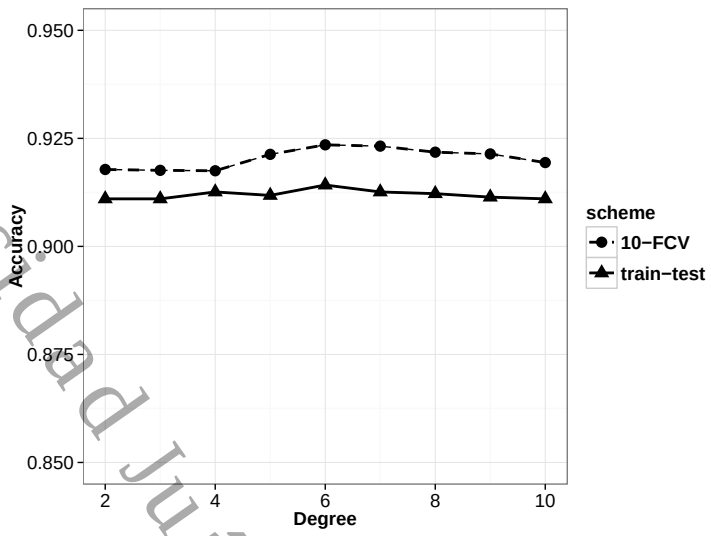

Figure 4.3: Accuracies vs degrees in polynomial kernel.

Table 4.5: Laplacian kernel optimization. The units of the results are shown in average accuracy over 30 runs. The highest value is shown in bold.

	train-test					10-FCV				
			C					C		
σ	**1**	**10**	**50**	**80**	**100**	**1**	**10**	**50**	**80**	**100**
0.001	0.732	0.732	0.893	0.898	0.905	0.727	0.752	0.878	0.895	0.903
0.01	0.732	0.905	**0.916**	0.912	0.913	0.745	0.901	**0.920**	0.911	0.913
0.1	0.898	0.913	0.913	0.913	0.913	0.891	0.913	0.913	0.913	0.913
1	0.880	0.884	0.884	0.884	0.884	0.875	0.875	0.875	0.875	0.875
10	0.732	0.732	0.732	0.732	0.732	0.727	0.727	0.727	0.727	0.727
100	0.732	0.732	0.732	0.732	0.732	0.727	0.727	0.727	0.727	0.727

4.4.2.3 Other classifiers parameter optimization

MLP, SLP, RBF-DDA and JRip require each a particular parameter optimization, as mentioned in section 3.1.5. These parameters were optimized for each classifier in each of the 30 runs in train-test and in 10-FCV scenarios, therefore the best parameters for each run were used for classification.

4.4.3 Results

4.4.3.1 Four GBS subtype classification

In this section, we show the results of the classification of the four GBS subtypes. Table 4.6 shows the average prediction results of each classifier across 30 runs in train-test scenario. The standard deviation of each metric is also shown. Seven classifiers, half of all the classifiers, obtained an average accuracy above 0.90. The best seven classifiers were: kNN, SVMLap, SVMPoly, SVMGaus, C4.5, SVMLin and Naive Bayes. Six of the remaining classifiers obtained an average accuracy around 0.88.

kNN obtained the best performance with 0.9268 of average accuracy, it slightly outperformed SVM in all kernels and C4.5. The worst performance was obtained by OneR classifier, with an average accuracy under 0.80 and with poor results in all remaining metrics. ANN classifiers, that is MLP, SLP and RBF-DDA, obtained similar results in all metrics.

Multiclass AUC ranged 0.70 - 0.84. Six classifiers obtained a multiclass AUC above 0.80. Specificity was higher than sensitivity. Specificity ranged 0.85 - 0.95, while sensitivity ranged 0.43 - 0.83. Kappa values obtained showed large variation, in the range of 0.37 - 0.77. Overall, four GBS subtype classification using train-test obtained high results in average accuracy. The remaining metrics showed varied results. However, half of classifiers stood out from the rest, obtaining on or above 0.80 in most metrics.

Figure 4.4 shows the average accuracy across 30 runs for each classifier in four GBS subtype classification using train-test. Also the standard deviation for each classifier is shown. Most of the classifiers obtained an average accuracy around 0.90, seven of them surpassed this quantity. In average, the standard deviation was around 0.03.

Table 4.7 shows the average prediction results across 30 runs of each classifier in four GBS subtype classification using 10-FCV. The standard deviation of each metric is also shown. Six classifiers, almost half of all the classifiers, obtained an average accuracy above 0.90. Six of them repeated as the best classifiers: kNN, SVMLap, SVMPoly, SVM SVMGaus, C4.5 and SVMLin. Five of the remaining classifiers obtained an average accuracy around 0.89. Two around 0.88, and again, OneR showed the worst performance with an average accuracy under 0.80 and overall very low values in all metrics.

Multiclass AUC ranged 0.75 - 0.89. A little higher than in train-test. Thirteen classifiers obtained a multiclass AUC above 0.82. Sensitivity ranged 0.39 - 0.83. Similar results to those obtained from train-test. Specificity ranged 0.85 - 0.95, similarly as in train-test. Again, Specificity was higher than sensitivity. Only OneR obtained a specificity under 0.90. Kappa value obtained showed large variation, in the range of 0.30 - 0.76. Similar situation for Kappa values occurred in train-test scenario. In short, four GBS subtype classification using 10-FCV obtained close-to-expected results in average accuracy. The remaining metrics behaved differently. As in train-test, half of classifiers stood out from the rest, obtaining comparable results to those obtained in train-test in most metrics.

Figure 4.5 shows the average accuracy across 30 runs for each classifier in four GBS subtype classification using 10-FCV. Also the standard deviation for each classifier is shown. Most of the classifiers obtained an average accuracy around 0.90, six of them surpassed this number. In average, the standard deviation was around 0.01.

Overall, similar results were obtained in both train-test and 10-FCV scenarios. Only two differences were observed. The first difference was in multiclass AUC metric, which was a little higher in 10-FCV. The second difference was in the standard deviation of the average accuracy, which was a little lower in 10-FCV.

Table 4.6: Four GBS subtype classification using train-test. The units of the results are shown in average over 30 runs.

Classifier	Optimal parameters	Average Accuracy	Multiclass AUC	Sensitivity	Specificity	Kappa
kNN	k=18 d=1	**0.9268**	**0.8408**	**0.8323**	**0.9503**	**0.7798**
		0.0224	0.0605	0.0678	0.0158	0.0678
SVMLap	s=0.01 C=50	**0.9163**	**0.7913**	**0.7753**	**0.9432**	**0.7523**
		0.0233	0.0844	0.0764	0.0159	0.0675
SVMPoly	d=6 coef=1	**0.9142**	**0.8361**	**0.7858**	**0.9390**	**0.7452**
	g=0.01 C=1	0.0230	0.0717	0.0674	0.0163	0.0677
SVMGaus	g=0.01 C=10	**0.9126**	**0.8210**	**0.7884**	**0.9397**	**0.7427**
		0.0292	0.0672	0.0704	0.0200	0.0827
C4.5		**0.9114**	**0.8085**	**0.7727**	**0.9356**	**0.7348**
		0.0283	0.0827	0.0727	0.0218	0.0850
SVMLin	C=1	**0.9069**	**0.7751**	**0.7572**	**0.9355**	**0.7246**
		0.0282	0.0830	0.0822	0.0198	0.0810
Naive Bayes		**0.9008**	**0.8113**	**0.7762**	**0.9324**	**0.7105**
		0.0233	0.0760	0.0669	0.0152	0.0650
MLP		**0.8915**	**0.7852**	**0.7273**	**0.9239**	**0.6795**
		0.0319	0.0768	0.0941	0.0225	0.0887
RBF-DDA		**0.8882**	**0.7542**	**0.6696**	**0.9202**	**0.6633**
		0.0246	0.0535	0.0836	0.0217	0.0786
SLP		**0.8850**	**0.7558**	**0.6972**	**0.9209**	**0.6608**
		0.0334	0.0795	0.1142	0.0229	0.0932
JRip		**0.8837**	**0.8083**	**0.7279**	**0.9137**	**0.6526**
		0.0326	0.0801	0.0899	0.0238	0.0944
LDA		**0.8825**	**0.7439**	**0.7200**	**0.9181**	**0.6547**
		0.0291	0.0789	0.0658	0.0205	0.0804
MLR		**0.8793**	**0.7125**	**0.7029**	**0.9156**	**0.6444**
		0.0329	0.0850	0.0772	0.0232	0.0930
OneR		**0.7972**	**0.7039**	**0.4365**	**0.8511**	**0.3771**
		0.0326	0.0451	0.0597	0.0243	0.0936

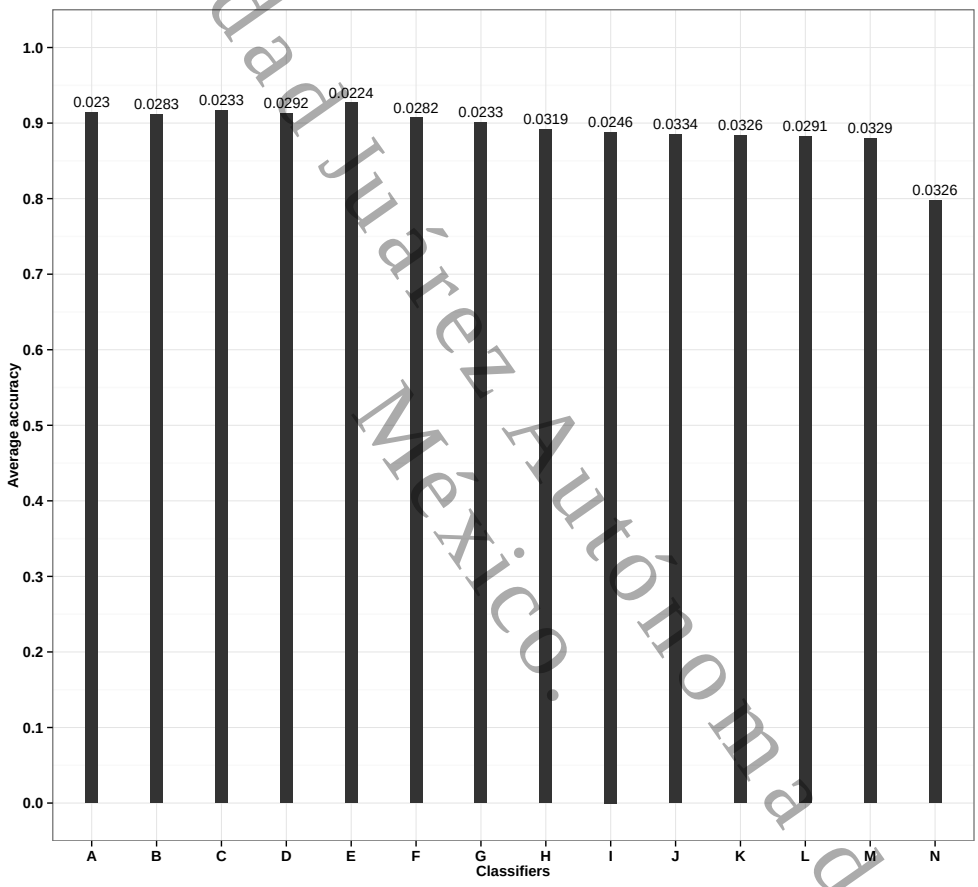

Figure 4.4: Average accuracy in four GBS subtype classification using train-test. A=SVMPoly, B=C4.5, C=SVMLap, D=SVMGaus, E=kNN, F=SVMLin, G=Naive Bayes, H=MLP, I=RBF-DDA, J=SLP, K=JRip, L=LDA, M=MLR, N=OneR.

Table 4.7: Four GBS subtype classification using 10-FCV. The units of the results are shown in average over 30 runs.

Classifier	Optimal parameters	Average Accuracy	Multiclass AUC	Sensitivity	Specificity	Kappa
SVMPoly	**d=6 coef=1 g=0.01 C=1**	**0.9235** 0.0080	**0.8985** 0.0199	**0.8308** 0.0287	**0.9481** 0.0058	**0.7648** 0.0265
C4.5		**0.9211** 0.0109	**0.8857** 0.0242	**0.8136** 0.0421	**0.9446** 0.0081	**0.7567** 0.0321
SVMLap	**s=0.01 C=50**	**0.9201** 0.0072	**0.8712** 0.0240	**0.7952** 0.0457	**0.9466** 0.0051	**0.7554** 0.0195
SVMGaus	**g=0.01 C=10**	**0.9193** 0.0067	**0.8897** 0.0221	**0.8212** 0.0316	**0.9448** 0.0049	**0.7515** 0.0214
***k*NN**	**k=14 d=1**	**0.9179** 0.0041	**0.8783** 0.0188	**0.7868** 0.0425	**0.9437** 0.0036	**0.7467** 0.0132
SVMLin	**C=1**	**0.9175** 0.0096	**0.8632** 0.0232	**0.7831** 0.0461	**0.9415** 0.0076	**0.7423** 0.0317
JRip		**0.8999** 0.0143	**0.8729** 0.0291	**0.7808** 0.0552	**0.9267** 0.0113	**0.6880** 0.0427
Naive Bayes		**0.8986** 0.0079	**0.8632** 0.0244	**0.7936** 0.0472	**0.9325** 0.0063	**0.6928** 0.0247
MLP		**0.8974** 0.0122	**0.8514** 0.0257	**0.7438** 0.0592	**0.9289** 0.0089	**0.6842** 0.0383
SLP		**0.8972** 0.0147	**0.8452** 0.0230	**0.7281** 0.0536	**0.9300** 0.0117	**0.6851** 0.0468
MLR		**0.8926** 0.0082	**0.8405** 0.0279	**0.7575** 0.0445	**0.9264** 0.0069	**0.6721** 0.0245
LDA		**0.8806** 0.0083	**0.8256** 0.0223	**0.7251** 0.0369	**0.9150** 0.0064	**0.6366** 0.0249
RBF-DDA		**0.8797** 0.0079	**0.8249** 0.0287	**0.6606** 0.0451	**0.9148** 0.0060	**0.6273** 0.0263
OneR		**0.7744** 0.0164	**0.7528** 0.0249	**0.3928** 0.0343	**0.8374** 0.0119	**0.3039** 0.0453

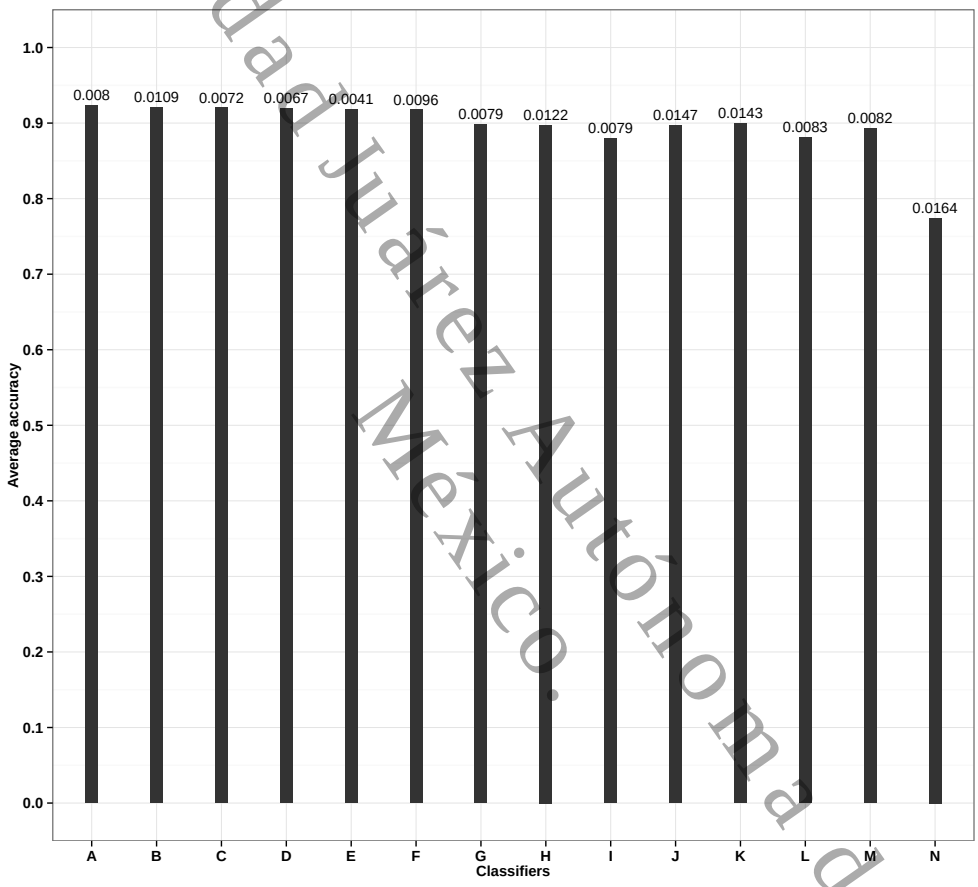

Figure 4.5: Average accuracy in four GBS subtype classification using 10-FCV. A=SVMPoly, B=C4.5, C=SVMLap, D=SVMGaus, E=kNN, F=SVMLin, G=Naive Bayes, H=MLP, I=RBF-DDA, J=SLP, K=JRip, L=LDA, M=MLR, N=OneR.

4.4.3.2 OVA classification

In this section, we describe the results of OVA classification using train-test and 10-FCV scenarios. That is, AIDP vs ALL, AMAN vs ALL, AMSAN vs ALL and MF vs ALL. As mentioned before, we used the balanced accuracy as our base metric in OVA classification scenario.

Table 4.8 shows the average prediction results of each classifier across 30 runs in AIDP vs ALL classification using train-test. The standard deviation of each metric is also shown. The balanced accuracy ranged 0.62 - 0.82, this contrasts with the results obtained in accuracy which ranged 0.81 - 0.93. However, balanced accuracy is a more reliable metric in this case since it is a two-class classification with imbalanced data. While specificity obtained high results, ranging 0.85 - 0.98; sensitivity obtained poor results, in the range of 0.35 - 0.67. Kappa statistic also showed a large variation among all classifiers, ranging 0.24 - 0.71. Four classifiers obtained a balanced accuracy above 0.80: kNN, C4.5, MLP, and SVMLap.

Table 4.9 shows the average prediction results across 30 runs of each classifier in AMAN vs ALL classification using train-test. The standard deviation of each metric is also shown. The balanced accuracy ranged 0.63 - 0.93, while accuracy ranged 0.71 - 0.92. Overall, balanced accuracy in AMAN vs ALL was a little higher than in AIDP vs ALL. Both specificity and sensitivity obtained superior results, the former in the range of 0.82 - 0.94, the latter in the range of 0.75 - 0.97 with one outlier obtained by OneR (0.4472). Kappa statistic ranged 0.62 - 0.81. Again, OneR obtained an atypical result (0.2735). Five classifiers obtained a balanced accuracy above 0.90: kNN, SVMGaus, SVMPoly, SVMLap, and C4.5.

Table 4.10 shows the average prediction results of each classifier across 30 runs in AMSAN vs ALL classification using train-test. The standard deviation of each metric is also shown. Both balanced accuracy and accuracy obtained similar results. Balanced accuracy ranged 0.79 - 0.89, while accuracy ranged 0.80 - 0.89. Shortly, AMSAN vs ALL obtained the lower variation in balanced accuracy in OVA classification using train-test. Both specificity and sensitivity obtained high values, the former in the range of 0.80 - 0.94, the latter in the range of 0.76 - 0.87. Kappa statistic ranged 0.59 - 0.79. Six classifiers obtained a balanced accuracy above 0.85: kNN, C4.5, SVMLap, RBF-DDA, SLP, and SVMPoly.

Table 4.11 shows the average prediction results of each classifier across 30 runs in MF vs ALL classification using train-test. The standard deviation of each metric is also shown. In this case, balanced accuracy and accuracy obtained very dissimilar results. Balanced accuracy ranged 0.50 - 0.89, while accuracy ranged 0.88 - 0.94. Overall, MF vs ALL obtained the worst results in balanced accuracy in OVA classification using train-test, with only one classifier above 0.80. Also specificity and sensitivity obtained very different results, the former in the range of 0.92 - 0.98, the latter in the range of 0.33 - 0.83 (with an outlier of 0.0250). Kappa statistic showed a large variation, in the range of 0.33 - 0.70 (with an outlier of 0.0089). Four classifiers obtained a balanced accuracy above 0.75: LDA, C4.5, JRip, and SVMGaus. One classifier obtained a balanced accuracy above 0.80, Naive Bayes.

Figure 4.6 shows the balanced accuracy for each classifier in OVA classification using train-test. The standard deviation is also shown. AMSAN vs ALL showed the most stable performance both in

Table 4.8: AIDP vs ALL classification using train-test. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
*k*NN	**0.8227**	**0.9381**	**0.6611**	**0.9843**	**0.7146**	**0.8227**
	0.0774	0.0291	0.1546	0.0238	0.1367	0.0774
C4.5	**0.8042**	**0.8984**	**0.6722**	**0.9361**	**0.6009**	**0.8042**
	0.0902	0.0496	0.1932	0.0611	0.1545	0.0902
MLP	**0.8014**	**0.9056**	**0.6556**	**0.9472**	**0.6087**	**0.8014**
	0.0784	0.0334	0.1691	0.0408	0.1265	0.0784
SVMLap	**0.8009**	**0.9246**	**0.6278**	**0.9741**	**0.6566**	**0.8009**
	0.0773	0.0266	0.1619	0.0272	0.1234	0.0773
RBF-DDA	**0.7648**	**0.8984**	**0.5778**	**0.9519**	**0.5547**	**0.7648**
	0.0824	0.0299	0.1736	0.0326	0.1424	0.0824
SLP	**0.7551**	**0.8738**	**0.5889**	**0.9213**	**0.4981**	**0.7551**
	0.0751	0.0401	0.1680	0.0531	0.1244	0.0751
SVMLin	0.7542	0.8802	0.5778	0.9306	0.5028	0.7542
	0.1040	0.0453	0.2132	0.0477	0.1947	0.1040
LDA	**0.7537**	**0.8714**	**0.5889**	**0.9185**	**0.4917**	**0.7537**
	0.0737	0.0408	0.1562	0.0494	0.1336	0.0737
SVMGaus	**0.7491**	**0.9032**	**0.5333**	**0.9648**	**0.5522**	**0.7491**
	0.0878	0.0353	0.1773	0.0334	0.1586	0.0878
JRip	**0.7407**	**0.8849**	**0.5389**	**0.9426**	**0.5126**	**0.7407**
	0.0661	0.0429	0.1431	0.0541	0.1391	0.0661
SVMPoly	**0.7389**	**0.9056**	**0.5056**	**0.9722**	**0.5462**	**0.7389**
	0.0903	0.0351	0.1830	0.0326	0.1637	0.0903
Naive Bayes	**0.7231**	**0.8151**	**0.5944**	**0.8519**	**0.3691**	**0.7231**
	0.1057	0.0514	0.2130	0.0543	0.1624	0.1057
BLR	**0.7181**	**0.8341**	**0.5556**	**0.8806**	**0.3995**	**0.7181**
	0.0851	0.0601	0.1714	0.0708	0.1755	0.0851
OneR	**0.6222**	**0.8167**	**0.3500**	**0.8944**	**0.2450**	**0.6222**
	0.0846	0.0497	0.1770	0.0594	0.1705	0.0846

Table 4.9: AMAN vs ALL classification using train-test. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
*k*NN	**0.9356**	**0.9198**	**0.9722**	**0.8989**	**0.8186**	**0.9356**
	0.0294	0.0372	0.0400	0.0536	0.0779	0.0294
SVMGaus	**0.9256**	**0.9198**	**0.9389**	**0.9122**	**0.8136**	**0.9256**
	0.0437	0.0388	0.0756	0.0459	0.0878	0.0437
SVMPoly	**0.9203**	**0.9111**	**0.9417**	**0.8989**	**0.7959**	**0.9203**
	0.0403	0.0385	0.0794	0.0550	0.0825	0.0403
SVMLap	**0.9181**	**0.9175**	**0.9194**	**0.9167**	**0.8069**	**0.9181**
	0.0404	0.0384	0.0742	0.0516	0.0857	0.0404
C4.5	**0.9067**	**0.9238**	**0.8667**	**0.9467**	**0.8124**	**0.9067**
	0.0548	0.0377	0.1129	0.0416	0.0933	0.0548
MLP	**0.8978**	**0.9111**	**0.8667**	**0.9289**	**0.7853**	**0.8978**
	0.0489	0.0390	0.0864	0.0399	0.0934	0.0489
SVMLin	**0.8969**	**0.9016**	**0.8861**	**0.9078**	**0.7678**	**0.8969**
	0.0486	0.0409	0.0861	0.0461	0.0957	0.0486
SLP	**0.8778**	**0.8944**	**0.8389**	**0.9167**	**0.7450**	**0.8778**
	0.0581	0.0454	0.1093	0.0501	0.1107	0.0581
Naive Bayes	**0.8700**	**0.8833**	**0.8389**	**0.9011**	**0.7206**	**0.8700**
	0.0754	0.0584	0.1348	0.0571	0.1414	0.0754
RBF-DDA	**0.8689**	**0.9103**	**0.7722**	**0.9656**	**0.7677**	**0.8689**
	0.0566	0.0323	0.1217	0.0283	0.0913	0.0566
LDA	**0.8594**	**0.8635**	**0.8500**	**0.8689**	**0.6848**	**0.8594**
	0.0549	0.0508	0.1081	0.0711	0.1122	0.0549
JRip	**0.8472**	**0.8889**	**0.7500**	**0.9444**	**0.7164**	**0.8472**
	0.0689	0.0478	0.1435	0.0549	0.1255	0.0689
BLR	**0.8214**	**0.8437**	**0.7694**	**0.8733**	**0.6292**	**0.8214**
	0.0566	0.0533	0.1087	0.0740	0.1158	0.0566
OneR	**0.6336**	**0.7135**	**0.4472**	**0.8200**	**0.2735**	**0.6336**
	0.0673	0.0581	0.1702	0.1027	0.1336	0.0673

Table 4.10: AMSAN vs ALL classification using train-test. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
kNN	**0.8953**	**0.8992**	**0.8544**	**0.9362**	**0.7953**	**0.8953**
	0.0447	0.0441	0.0687	0.0568	0.0894	0.0447
C4.5	**0.8852**	**0.8865**	**0.8719**	**0.8986**	**0.7703**	**0.8852**
	0.0486	0.0450	0.1031	0.0515	0.0929	0.0486
SVMLap	**0.8756**	**0.8817**	**0.8105**	**0.9406**	**0.7587**	**0.8756**
	0.0477	0.0462	0.0848	0.0599	0.0946	0.0477
RBF-DDA	**0.8695**	**0.8746**	**0.8158**	**0.9232**	**0.7452**	**0.8695**
	0.0476	0.0488	0.0630	0.0763	0.0980	0.0476
SLP	**0.8644**	**0.8667**	**0.8404**	**0.8884**	**0.7306**	**0.8644**
	0.0545	0.0551	0.0788	0.0829	0.1101	0.0545
SVMPoly	**0.8504**	**0.8587**	**0.7632**	**0.9377**	**0.7104**	**0.8504**
	0.0486	0.0459	0.0985	0.0590	0.0955	0.0486
SVMGaus	**0.8492**	**0.8563**	**0.7737**	**0.9246**	**0.7064**	**0.8492**
	0.0447	0.0431	0.0918	0.0675	0.0886	0.0447
MLP	**0.8490**	**0.8524**	**0.8140**	**0.8841**	**0.7008**	**0.8490**
	0.0429	0.0419	0.0801	0.0632	0.0853	0.0429
JRip	**0.8449**	**0.8444**	**0.8491**	**0.8406**	**0.6874**	**0.8449**
	0.0473	0.0479	0.0994	0.0984	0.0953	0.0473
OneR	**0.8219**	**0.8238**	**0.8018**	**0.8420**	**0.6441**	**0.8219**
	0.0446	0.0436	0.0766	0.0588	0.0883	0.0446
Naive Bayes	**0.8166**	**0.8159**	**0.8246**	**0.8087**	**0.6306**	**0.8166**
	0.0532	0.0545	0.0653	0.0812	0.1077	0.0532
SVMLin	**0.8101**	**0.8127**	**0.7825**	**0.8377**	**0.6213**	**0.8101**
	0.0426	0.0423	0.0860	0.0765	0.0851	0.0426
LDA	**0.8072**	**0.8095**	**0.7825**	**0.8319**	**0.6150**	**0.8072**
	0.0422	0.0415	0.0838	0.0701	0.0845	0.0422
BLR	**0.7971**	**0.8000**	**0.7667**	**0.8275**	**0.5955**	**0.7971**
	0.0588	0.0582	0.0914	0.0787	0.1179	0.0588

Table 4.11: MF vs ALL classification using train-test. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
Naive Bayes	**0.8939**	**0.9429**	**0.8333**	**0.9544**	**0.7052**	**0.8939**
	0.0914	0.0284	0.1895	0.0309	0.1401	0.0914
LDA	**0.7818**	**0.9087**	**0.6250**	**0.9386**	**0.5025**	**0.7818**
	0.1252	0.0343	0.2687	0.0428	0.1964	0.1252
C4.5	**0.7691**	**0.9127**	**0.5917**	**0.9465**	**0.5002**	**0.7691**
	0.1481	0.0416	0.3043	0.0423	0.2489	0.1481
JRip	**0.7678**	**0.9103**	**0.5917**	**0.9439**	**0.4662**	**0.7678**
	0.1701	0.0389	0.3625	0.0477	0.2788	0.1701
SVMGaus	**0.7581**	**0.9198**	**0.5583**	**0.9579**	**0.5072**	**0.7581**
	0.1310	0.0351	0.2682	0.0343	0.2370	0.1310
BLR	**0.7316**	**0.8921**	**0.5333**	**0.9298**	**0.4105**	**0.7316**
	0.1280	0.0389	0.2765	0.0490	0.1986	0.1280
SVMLin	**0.7221**	**0.9087**	**0.4917**	**0.9526**	**0.4298**	**0.7221**
	0.1272	0.0273	0.2665	0.0296	0.2223	0.1272
MLP	**0.7193**	**0.8968**	**0.5000**	**0.9386**	**0.4060**	**0.7193**
	0.1353	0.0444	0.2862	0.0519	0.2394	0.1353
SVMPoly	**0.7086**	**0.9111**	**0.4583**	**0.9588**	**0.4189**	**0.7086**
	0.1323	0.0279	0.2792	0.0322	0.2085	0.1323
SLP	**0.6875**	**0.9000**	**0.4250**	**0.9500**	**0.3624**	**0.6875**
	0.1404	0.0356	0.2947	0.0399	0.2463	0.1404
OneR	**0.6853**	**0.8825**	**0.4417**	**0.9289**	**0.3315**	**0.6853**
	0.1396	0.0451	0.2913	0.0504	0.2296	0.1396
SVMLap	**0.6664**	**0.9024**	**0.3750**	**0.9579**	**0.3354**	**0.6664**
	0.1326	0.0302	0.2766	0.0321	0.2427	0.1326
kNN	**0.6539**	**0.9135**	**0.3333**	**0.9746**	**0.3531**	**0.6539**
	0.1117	0.0246	0.2306	0.0234	0.2136	0.1117
RBF-DDA	**0.5042**	**0.8921**	**0.0250**	**0.9833**	**0.0089**	**0.5042**
	0.0343	0.0223	0.0763	0.0272	0.0882	0.0343

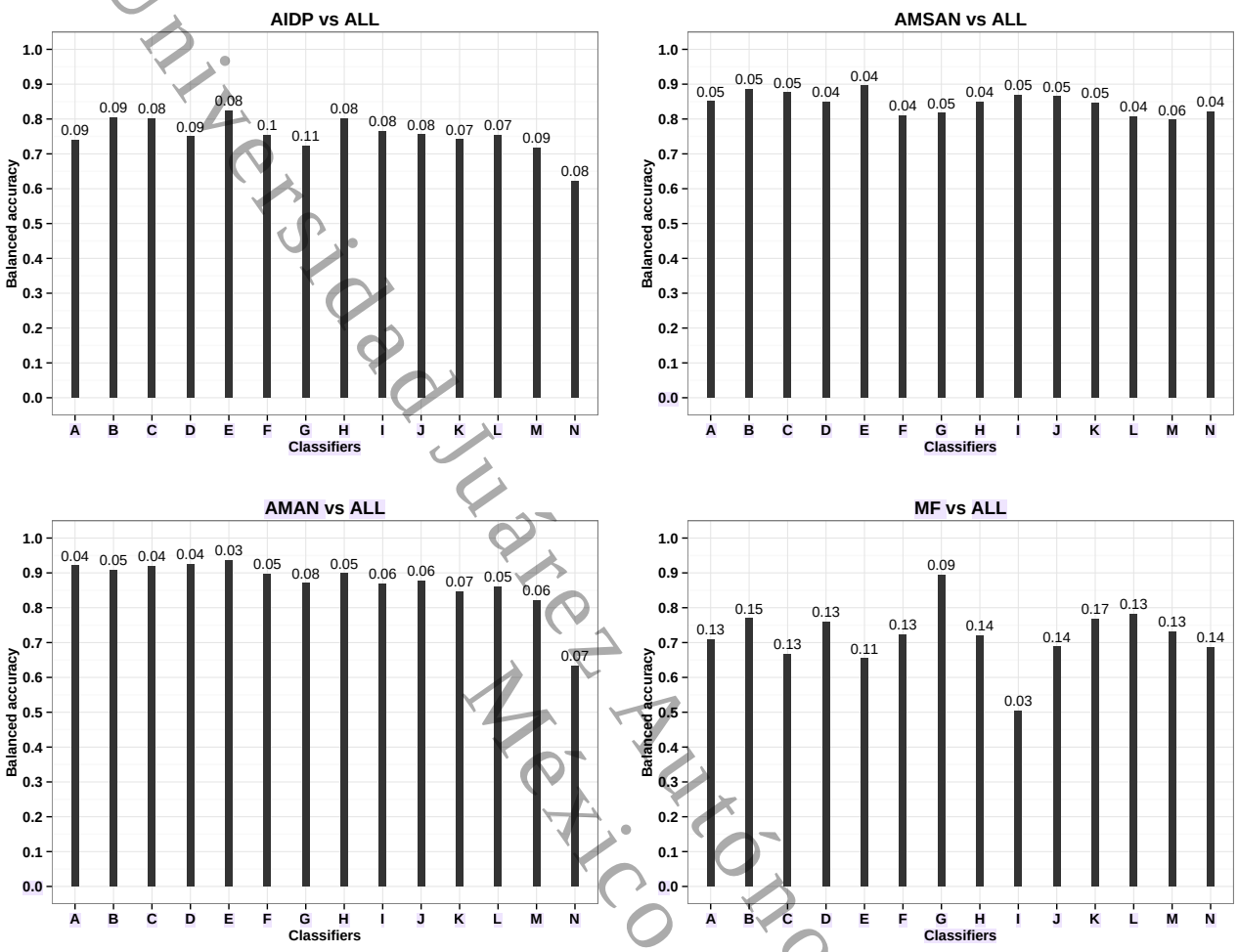

Figure 4.6: Balanced accuracy in OVA classification using train-test. A=SVMPoly, B=C4.5, C=SVMLap, D=SVMGaus, E=kNN, F=SVMLin, G=Naive Bayes, H=MLP, I=RBF-DDA, J=SLP, K=JRip, L=LDA, M=BLR, N=OneR.

balanced accuracy and in standard deviation across 30 runs. The opposite case was MF vs ALL. AMAN vs ALL obtained the highest classification performance, AMSAN vs ALL was the second best.

Table 4.12 shows the average prediction results of each classifier across 30 runs in AIDP vs ALL classification using 10-FCV. The standard deviation of each metric is also shown. Balanced accuracy ranged 0.64 - 0.81, while accuracy ranged 0.79 - 0.92. Sensitivity showed a large variation, in the range of 0.43 - 0.69. Values of specificity were much higher than those of sensitivity, in the range of 0.83 - 0.98. Kappa statistic showed also a large variation, in the range of 0.26 - 0.66. Four classifiers obtained a balanced accuracy above 0.80: MLP, SVMLap, C4.5, and kNN.

Table 4.13 shows the average prediction results of each classifier across 30 runs in AMAN vs ALL classification using 10-FCV. The standard deviation of each metric is also shown. Values obtained of

Table 4.12: AIDP vs ALL classification using 10-FCV. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
MLP	**0.8183**	**0.9039**	**0.6900**	**0.9467**	**0.6349**	**0.8183**
	0.0204	0.0104	0.0403	0.0103	0.0390	0.0204
SVMLap	**0.8158**	**0.9231**	**0.6550**	**0.9767**	**0.6612**	**0.8158**
	0.0214	0.0111	0.0402	0.0092	0.0474	0.0214
C4.5	**0.8083**	**0.9039**	**0.6650**	**0.9517**	**0.6119**	**0.8083**
	0.0226	0.0111	0.0458	0.0115	0.0451	0.0226
***k*NN**	**0.8012**	**0.9219**	**0.6200**	**0.9823**	**0.6442**	**0.8012**
	0.0135	0.0056	0.0282	0.0057	0.0316	0.0135
LDA	**0.7928**	**0.8869**	**0.6517**	**0.9340**	**0.5841**	**0.7928**
	0.0138	0.0115	0.0245	0.0130	0.0327	0.0138
SVMGaus	**0.7807**	**0.9044**	**0.5950**	**0.9663**	**0.5858**	**0.7807**
	0.0222	0.0111	0.0442	0.0110	0.0491	0.0222
JRip	**0.7800**	**0.8989**	**0.6017**	**0.9583**	**0.5753**	**0.7800**
	0.0403	0.0180	0.0815	0.0170	0.0833	0.0403
SLP	**0.7753**	**0.8767**	**0.6233**	**0.9273**	**0.5365**	**0.7753**
	0.0323	0.0190	0.0640	0.0207	0.0681	0.0323
RBF-DDA	**0.7715**	**0.8947**	**0.5867**	**0.9563**	**0.5578**	**0.7715**
	0.0254	0.0136	0.0472	0.0116	0.0595	0.0254
BLR	**0.7588**	**0.8603**	**0.6067**	**0.9110**	**0.5038**	**0.7588**
	0.0233	0.0150	0.0430	0.0152	0.0494	0.0233
SVMPoly	**0.7578**	**0.9042**	**0.5383**	**0.9773**	**0.5485**	**0.7578**
	0.0228	0.0100	0.0449	0.0078	0.0413	0.0228
SVMLin	**0.7552**	**0.8819**	**0.5650**	**0.9453**	**0.5188**	**0.7552**
	0.0204	0.0153	0.0326	0.0148	0.0520	0.0204
Naive Bayes	**0.7432**	**0.8075**	**0.6467**	**0.8397**	**0.4144**	**0.7465**
	0.0100	0.0132	0.0127	0.0156	0.0327	0.0155
OneR	**0.6497**	**0.7961**	**0.4300**	**0.8693**	**0.2663**	**0.6517**
	0.0486	0.0213	0.1005	0.0218	0.0835	0.0489

balanced accuracy and accuracy were similar, the former ranged 0.63 - 0.94, while the latter ranged 0.79 - 0.97. Values of sensitivity and specificity were very different, the former in the range of 0.46 - 0.97, the latter in the range of 0.83 - 0.98. Values obtained of Kappa statistic were very varied, in the range of 0.25 - 0.84. Nine classifiers obtained a balanced accuracy above 0.90, four of them were SVM with all different kernels. As a conclusion, this classification scenario obtained very high results with most classifiers.

Table 4.14 shows the average prediction results of each classifier across 30 runs in AMSAN vs ALL classification using 10-FCV. The standard deviation of each metric is also shown. Results obtained of balanced accuracy and accuracy were very similar. Balanced accuracy ranged 0.79 - 0.89, while accuracy ranged 0.80 - 0.90. Sensitivity ranged 0.74 - 0.88, specificity ranged 0.81 - 0.95. Kappa ranged 0.58 - 0.79. The best five classifiers were: kNN, C4.5, SVMLap, SLP and RBF-DDA. They obtained a balanced accuracy above 0.86. Overall, AMSAN vs ALL obtained high values with most classifiers.

Table 4.15 shows the average prediction results of each classifier across 30 runs in MF vs ALL classification using 10-FCV. The standard deviation of each metric is also shown. Results obtained in balanced accuracy were a little lower than those obtained in accuracy. Balanced accuracy ranged 0.50 - 0.89, and accuracy ranged 0.90 - 0.93. Very dissimilar results were obtained in sensitivity and in specificity. Sensitivity ranged 0.37 - 0.84 (with an outlier of 0.03), while specificity ranged 0.94 - 0.98. Kappa ranged 0.28 - 0.67, with an outlier of 0.01. In short, MF vs ALL obtained showed the lowest and more varied results.

Figure 4.7 shows the average balanced accuracy across 30 runs for each classifier in OVA classification using 10-FCV. The standard deviation is also shown. Again, AMSAN vs ALL showed the most stable performance both in balanced accuracy and in standard deviation across 30 runs. The opposite case was MF vs ALL, as in train-test. AMAN vs ALL obtained the highest classification performance, AMSAN vs ALL was the second best.

Figure 4.8 shows the median ROC curve values of the top five classifiers in OVA classification using train-test. That is, from each OVA classification average results table, the top five classifiers were selected. Then, across 30 ROC curve values, the median value was selected of each of the top five classifiers. The median value was selected to illustrate the fairest result across 30 runs. The highest median ROC curve values using train-test were obtained in AMAN vs ALL classification, ranging 0.91 - 0.93. Second best was AMSAN vs ALL, ranging 0.85 - 0.89. The worst median ROC curve values were obtained in MF vs ALL, ranging 0.73 - 0.83, with one median ROC curve above 0.94. Overall, most of the top five classifiers in each OVA classification scenario using train-test, obtained median ROC curve values on 0.80 or above.

4.4.3.3 OVO classification

In this section, we describe the results of OVO classification using train-test and 10-FCV scenarios. That is, AIDP vs AMAN, AIDP vs AMSAN, AIDP vs MF, AMAN vs AMSAN, AMAN vs MF and AMSAN vs MF. As mentioned before, we used the balanced accuracy as our base metric in OVO classification scenario.

Table 4.13: AMAN vs ALL classification using 10-FCV. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
SVMPoly	**0.9498**	**0.9353**	**0.9789**	**0.9207**	**0.8491**	**0.9498**
	0.0135	0.0113	0.0223	0.0119	0.0281	0.0135
SVMLap	**0.9459**	**0.9356**	**0.9667**	**0.9252**	**0.8465**	**0.9459**
	0.0173	0.0124	0.0303	0.0109	0.0349	0.0173
kNN	**0.9441**	**0.9311**	**0.9700**	**0.9181**	**0.8395**	**0.9441**
	0.0067	0.0076	0.0102	0.0099	0.0164	0.0067
SVMGaus	**0.9400**	**0.9272**	**0.9656**	**0.9144**	**0.8279**	**0.9400**
	0.0177	0.0114	0.0333	0.0093	0.0336	0.0177
MLP	**0.9256**	**0.9361**	**0.9044**	**0.9467**	**0.8351**	**0.9256**
	0.0180	0.0119	0.0347	0.0115	0.0299	0.0180
SLP	**0.9244**	**0.9322**	**0.9089**	**0.9400**	**0.8278**	**0.9244**
	0.0193	0.0141	0.0338	0.0129	0.0362	0.0193
C4.5	**0.9224**	**0.9403**	**0.8867**	**0.9581**	**0.8397**	**0.9224**
	0.0199	0.0140	0.0357	0.0126	0.0381	0.0199
SVMLin	**0.9046**	**0.9086**	**0.8967**	**0.9126**	**0.7737**	**0.9046**
	0.0244	0.0173	0.0458	0.0174	0.0461	0.0244
RBF-DDA	**0.9033**	**0.9367**	**0.8367**	**0.9700**	**0.8206**	**0.9033**
	0.0194	0.0104	0.0385	0.0052	0.0324	0.0194
LDA	**0.8902**	**0.8886**	**0.8933**	**0.8870**	**0.7302**	**0.8902**
	0.0125	0.0094	0.0254	0.0117	0.0230	0.0125
Naive Bayes	**0.8794**	**0.8836**	**0.8711**	**0.8878**	**0.7135**	**0.8794**
	0.0182	0.0099	0.0389	0.0103	0.0330	0.0182
BLR	**0.8556**	**0.8839**	**0.7989**	**0.9122**	**0.6985**	**0.8556**
	0.0197	0.0131	0.0396	0.0144	0.0344	0.0197
JRip	**0.8454**	**0.8925**	**0.7511**	**0.9396**	**0.6965**	**0.8454**
	0.0312	0.0215	0.0605	0.0225	0.0610	0.0312
OneR	**0.6313**	**0.7147**	**0.4644**	**0.7981**	**0.2504**	**0.6339**
	0.0404	0.0321	0.0732	0.0356	0.0782	0.0413

Table 4.14: AMSAN vs ALL classification using 10-FCV. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
kNN	**0.8951**	**0.9028**	**0.8493**	**0.9410**	**0.7969**	**0.8951**
	0.0124	0.0108	0.0227	0.0062	0.0227	0.0124
C4.5	**0.8860**	**0.8861**	**0.8853**	**0.8867**	**0.7672**	**0.8860**
	0.0163	0.0144	0.0310	0.0130	0.0304	0.0163
SVMLap	**0.8767**	**0.8900**	**0.7967**	**0.9567**	**0.7670**	**0.8767**
	0.0189	0.0180	0.0268	0.0157	0.0372	0.0189
SLP	**0.8647**	**0.8703**	**0.8313**	**0.8981**	**0.7321**	**0.8647**
	0.0229	0.0226	0.0343	0.0290	0.0460	0.0229
RBF-DDA	**0.8629**	**0.8711**	**0.8133**	**0.9124**	**0.7318**	**0.8629**
	0.0138	0.0136	0.0212	0.0183	0.0278	0.0138
MLP	**0.8527**	**0.8594**	**0.8120**	**0.8933**	**0.7090**	**0.8527**
	0.0180	0.0173	0.0313	0.0233	0.0356	0.0180
SVMPoly	**0.8454**	**0.8611**	**0.7513**	**0.9395**	**0.7049**	**0.8454**
	0.0183	0.0167	0.0305	0.0134	0.0351	0.0183
JRip	**0.8420**	**0.8411**	**0.8473**	**0.8367**	**0.6782**	**0.8420**
	0.0212	0.0202	0.0431	0.0320	0.0407	0.0212
SVMGaus	**0.8403**	**0.8528**	**0.7653**	**0.9152**	**0.6905**	**0.8403**
	0.0184	0.0174	0.0297	0.0191	0.0359	0.0184
Naive Bayes	**0.8112**	**0.8125**	**0.8033**	**0.8190**	**0.6177**	**0.8112**
	0.0140	0.0127	0.0241	0.0115	0.0270	0.0140
BLR	**0.7969**	**0.8028**	**0.7613**	**0.8324**	**0.5936**	**0.7969**
	0.0188	0.0178	0.0287	0.0180	0.0375	0.0188
LDA	**0.7963**	**0.8039**	**0.7507**	**0.8419**	**0.5941**	**0.7963**
	0.0152	0.0141	0.0282	0.0176	0.0289	0.0152
OneR	**0.7925**	**0.8006**	**0.7440**	**0.8410**	**0.5878**	**0.7925**
	0.0191	0.0175	0.0342	0.0179	0.0375	0.0191
SVMLin	**0.7916**	**0.8000**	**0.7413**	**0.8419**	**0.5860**	**0.7922**
	0.0192	0.0182	0.0306	0.0195	0.0379	0.0195

Table 4.15: MF vs ALL classification using 10-FCV. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
Naive Bayes	**0.8956**	**0.9392**	**0.8433**	**0.9479**	**0.6770**	**0.8956**
	0.0252	0.0059	0.0504	0.0047	0.0482	0.0252
JRip	**0.8395**	**0.9392**	**0.7200**	**0.9591**	**0.6101**	**0.8395**
	0.0424	0.0130	0.0847	0.0123	0.0879	0.0424
LDA	**0.8218**	**0.9233**	**0.7000**	**0.9436**	**0.5418**	**0.8218**
	0.0371	0.0091	0.0743	0.0077	0.0665	0.0371
SVMGaus	**0.8168**	**0.9392**	**0.6700**	**0.9636**	**0.5665**	**0.8168**
	0.0397	0.0096	0.0794	0.0079	0.0792	0.0397
SVMLin	**0.8150**	**0.9247**	**0.6833**	**0.9467**	**0.5287**	**0.8150**
	0.0438	0.0123	0.0874	0.0111	0.0854	0.0438
C4.5	**0.7971**	**0.9169**	**0.6533**	**0.9409**	**0.4676**	**0.7971**
	0.0446	0.0151	0.0900	0.0154	0.0830	0.0446
SVMPoly	**0.7711**	**0.9247**	**0.5867**	**0.9555**	**0.4740**	**0.7711**
	0.0420	0.0086	0.0860	0.0080	0.0746	0.0420
kNN	**0.7609**	**0.9311**	**0.5567**	**0.9652**	**0.4691**	**0.7609**
	0.0426	0.0076	0.0858	0.0042	0.0748	0.0426
MLP	**0.7579**	**0.9117**	**0.5733**	**0.9424**	**0.4197**	**0.7579**
	0.0695	0.0149	0.1413	0.0131	0.1227	0.0695
SVMLap	**0.7556**	**0.9214**	**0.5567**	**0.9545**	**0.4435**	**0.7556**
	0.0422	0.0087	0.0858	0.0075	0.0756	0.0422
SLP	**0.7211**	**0.9108**	**0.4933**	**0.9488**	**0.3750**	**0.7211**
	0.0659	0.0179	0.1258	0.0111	0.1263	0.0659
BLR	**0.7211**	**0.9108**	**0.4933**	**0.9488**	**0.3750**	**0.7211**
	0.0659	0.0179	0.1258	0.0111	0.1263	0.0659
OneR	**0.6641**	**0.9092**	**0.3700**	**0.9582**	**0.2873**	**0.6641**
	0.0403	0.0110	0.0837	0.0121	0.0711	0.0403
RBF-DDA	**0.5071**	**0.9047**	**0.0300**	**0.9842**	**0.0117**	**0.5071**
	0.0288	0.0089	0.0596	0.0098	0.0519	0.0288

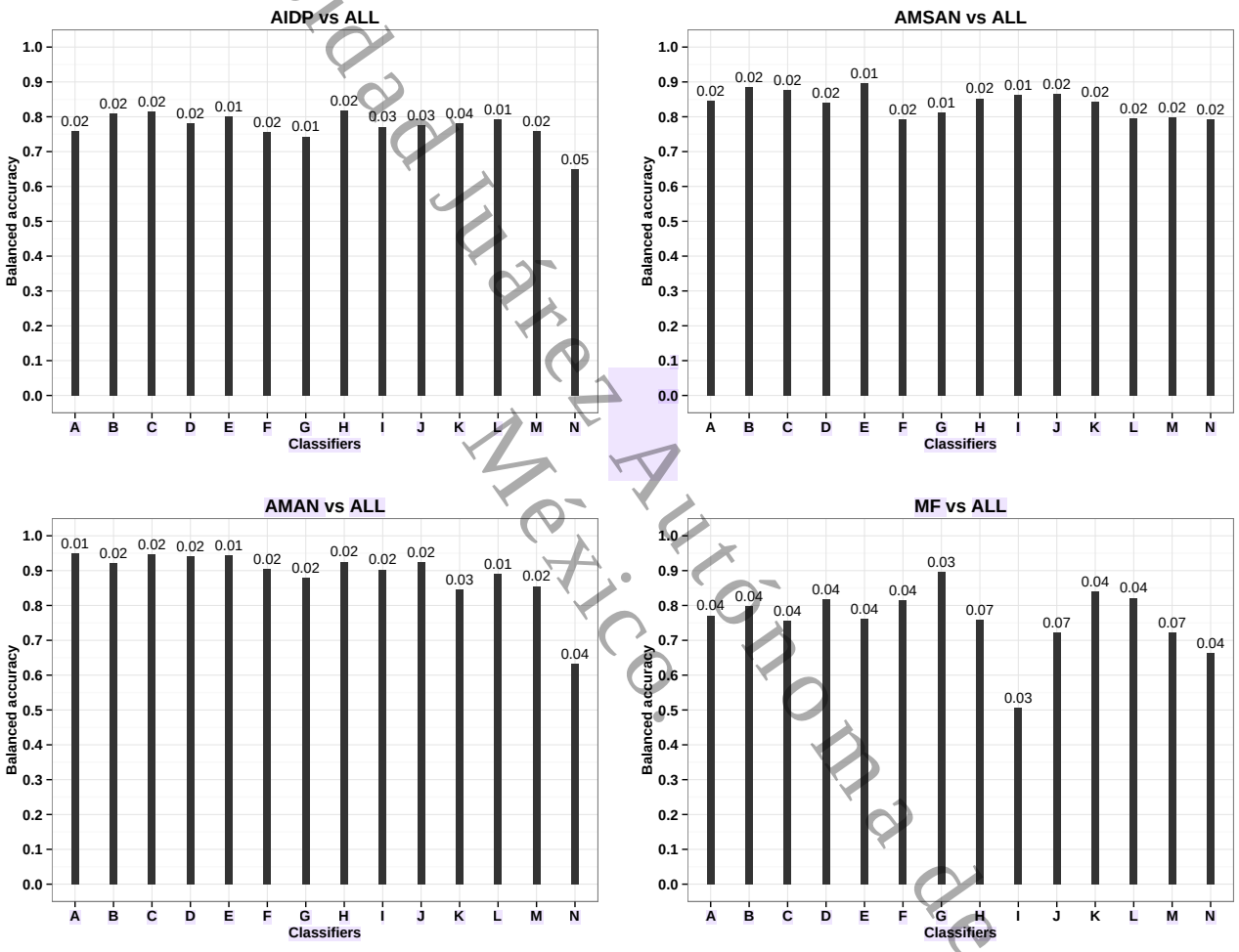

Figure 4.7: Balanced accuracy in OVA classification using 10-FCV. A=SVMPoly, B=C4.5, C=SVMLap, D=SVMGaus, E=kNN, F=SVMLin, G=Naive Bayes, H=MLP, I=RBF-DDA, J=SLP, K=JRip, L=LDA, M=BLR, N=OneR.

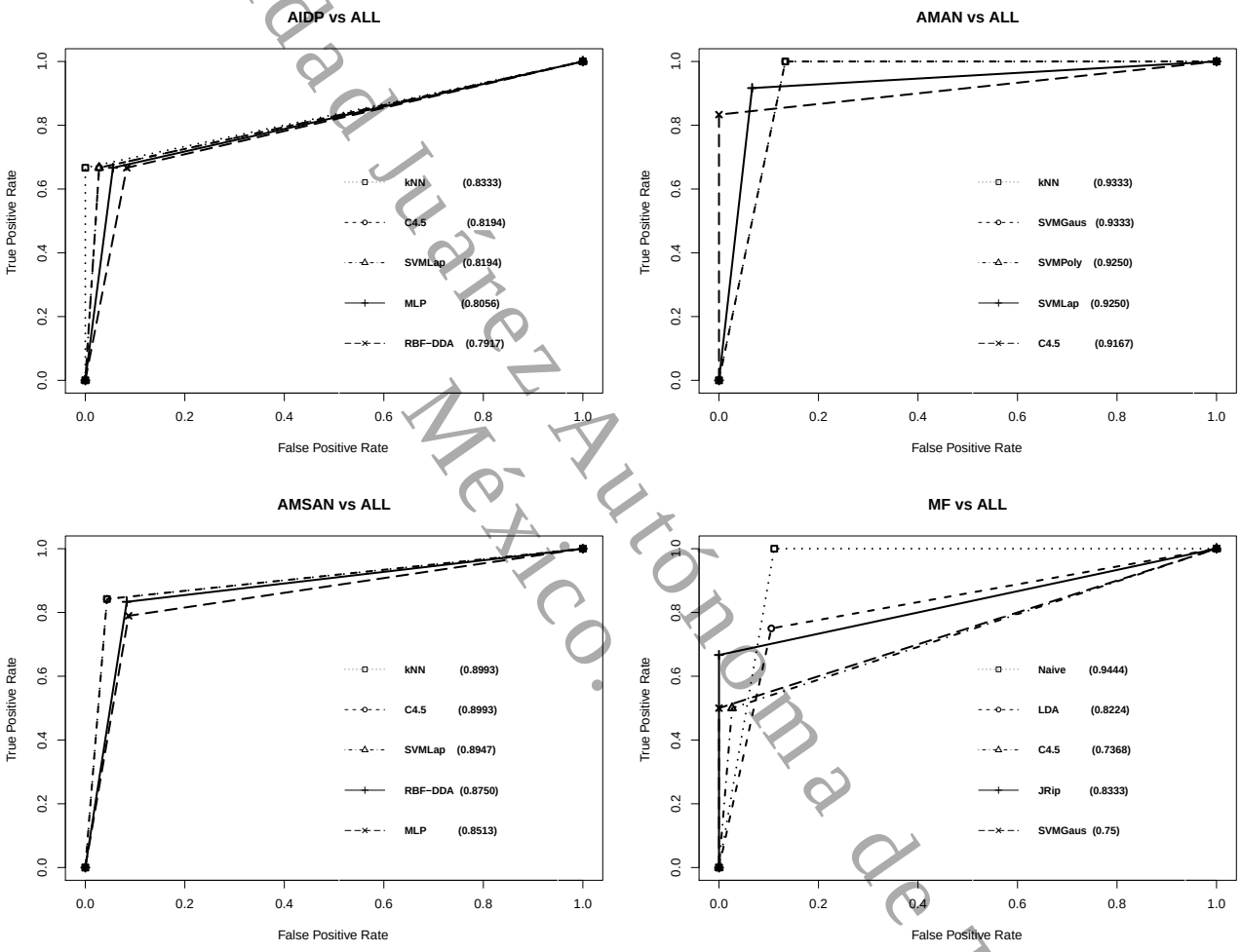

Figure 4.8: Median ROC curves of the top five classifiers in OVA classification using train-test.

Table 4.16 shows the average prediction results of each classifier across 30 runs in AIDP vs AMAN classification using train-test. The standard deviation of each metric is also shown. Very similar results were obtained in both balanced accuracy and accuracy. The balanced accuracy ranged 0.83 - 0.96, and accuracy ranged 0.87 - 0.97. High values were obtained in both specificity and sensitivity, the former ranging 0.89 - 1.00; the latter in the range of 0.72 - 0.95. Also, high values were obtained in Kappa statistic, ranging 0.70 - 0.93. ROC curves ranged 0.83 - 0.96, a very high values. Nine classifiers obtained a balanced accuracy above 0.90. Overall, AIDP vs AMAN classification using train-test had a very high performance with most classifiers.

Table 4.17 shows the average prediction results of each classifier across 30 runs in AIDP vs AMSAN classification using train-test. The standard deviation of each metric is also shown. Both balanced accuracy and accuracy behaved similarly. The balanced accuracy ranged 0.78 - 0.90, and accuracy ranged 0.83 - 0.93. Specificity values were a little higher than those of sensitivity, the former ranging 0.84 - 0.97; the latter in the range of 0.67 - 0.90. Kappa statistic behaved dissimilarly, ranging 0.56 - 0.81. ROC curves ranged 0.78 - 0.90, a high results. Only one classifier obtained a balanced accuracy above 0.90, SVMLap. Overall, AIDP vs AMSAN classification using train-test obtained a high performance with most classifiers.

Table 4.18 shows the average prediction results of each classifier across 30 runs in AIDP vs MF classification using train-test. The standard deviation of each metric is also shown. Similar results were obtained in both balanced accuracy and accuracy. The balanced accuracy ranged 0.61 - 0.86, and accuracy ranged 0.62 - 0.85. Very wide values were obtained in specificity, ranging 0.54 - 0.98; on the other hand, sensitivity ranged 0.67 - 0.91. Kappa statistic showed a large variation, ranging 0.21 - 0.69. ROC curves ranged 0.62 - 0.87. Only two classifiers obtained a balanced accuracy above 0.85. An outstanding fact was that OneR obtained the best performance in AIDP vs MF classification using train-test. Shortly, AIDP vs MF classification using train-test had acceptable results.

Table 4.19 shows the average prediction results of each classifier across 30 runs in AMAN vs AMSAN classification using train-test. The standard deviation of each metric is also shown. Both balanced accuracy and accuracy behaved very similar. The balanced accuracy ranged 0.84 - 0.95, and accuracy ranged 0.83 - 0.95. Specificity showed a little more variation than sensitivity, ranging 0.79 - 0.98; on the other hand, sensitivity ranged 0.89 - 0.96. Also Kappa statistic showed a little variation, ranging 0.66 - 0.91. High ROC curves values were obtained, ranging 0.84 - 0.95. Ten classifiers obtained a balanced accuracy above 0.90. Overall, most classifiers had a high performance in AMAN vs AMSAN classification using train-test.

Table 4.20 shows the average prediction results of each classifier across 30 runs in AMAN vs MF classification using train-test. The standard deviation of each metric is also shown. Again, both balanced accuracy and accuracy behaved very similar. The balanced accuracy ranged 0.87 - 0.95, and accuracy ranged 0.89 - 0.97. Very high values were obtained in both specificity and sensitivity. Specificity ranging 0.81 - 0.98; while sensitivity ranged 0.83 - 1.00. Kappa statistic ranged 0.73 - 0.93. High values were obtained in ROC curves, ranging 0.87 - 0.95. Ten classifiers obtained a balanced accuracy above 0.90, and all classifiers obtained a balanced accuracy above 0.85. In summary, results in AMAN vs MF classification using train-test were remarkable with most classifiers.

Table 4.16: AIDP vs AMAN classification using train-test. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
JRip	**0.9639**	**0.9704**	**0.9444**	**0.9833**	**0.9309**	**0.9639**
	0.0576	0.0406	0.1185	0.0339	0.0993	0.0576
OneR	**0.9583**	**0.9593**	**0.9556**	**0.9611**	**0.9095**	**0.9583**
	0.0438	0.0384	0.0868	0.0524	0.0855	0.0438
C4.5	**0.9458**	**0.9481**	**0.9389**	**0.9528**	**0.8841**	**0.9458**
	0.0723	0.0624	0.1177	0.0596	0.1427	0.0723
SVMLin	**0.9347**	**0.9500**	**0.8889**	**0.9806**	**0.8841**	**0.9347**
	0.0634	0.0504	0.1165	0.0466	0.1176	0.0634
SLP	**0.9333**	**0.9463**	**0.8944**	**0.9722**	**0.8767**	**0.9333**
	0.0644	0.0515	0.1275	0.0593	0.1168	0.0644
MLP	**0.9333**	**0.9500**	**0.8833**	**0.9833**	**0.8825**	**0.9333**
	0.0714	0.0533	0.1394	0.0459	0.1264	0.0714
Naive Bayes	**0.9278**	**0.9167**	**0.9611**	**0.8944**	**0.8227**	**0.9278**
	0.0526	0.0551	0.0826	0.0774	0.1150	0.0526
SVMLap	**0.9125**	**0.9370**	**0.8389**	**0.9861**	**0.8502**	**0.9125**
	0.0704	0.0500	0.1417	0.0384	0.1223	0.0704
SVMGaus	**0.9014**	**0.9278**	**0.8222**	**0.9806**	**0.8289**	**0.9014**
	0.0746	0.0568	0.1379	0.0420	0.1370	0.0746
RBF-DDA	**0.8958**	**0.9074**	**0.8611**	**0.9306**	**0.7943**	**0.8958**
	0.0724	0.0705	0.1165	0.0905	0.1481	0.0724
SVMPoly	**0.8722**	**0.9148**	**0.7444**	**1.0000**	**0.7870**	**0.8722**
	0.0947	0.0631	0.1894	0.0000	0.1681	0.0947
kNN	**0.8639**	**0.9093**	**0.7278**	**1.0000**	**0.7760**	**0.8639**
	0.0729	0.0486	0.1458	0.0000	0.1259	0.0729
BLR	**0.8542**	**0.8778**	**0.7833**	**0.9250**	**0.7203**	**0.8542**
	0.0874	0.0764	0.1646	0.0963	0.1709	0.0874
LDA	**0.8389**	**0.8759**	**0.7278**	**0.9500**	**0.7071**	**0.8389**
	0.0672	0.0540	0.1417	0.0713	0.1287	0.0672

Table 4.17: AIDP vs AMSAN classification using train-test. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
SVMLap	**0.9061**	**0.9267**	**0.8667**	**0.9456**	**0.8065**	**0.9061**
	0.0666	0.0576	0.1191	0.0684	0.1433	0.0666
SVMPoly	**0.8990**	**0.9360**	**0.8278**	**0.9702**	**0.8199**	**0.8990**
	0.0763	0.0542	0.1348	0.0492	0.1477	0.0763
MLP	**0.8949**	**0.9067**	**0.8722**	**0.9175**	**0.7619**	**0.8949**
	0.0647	0.0616	0.1043	0.0727	0.1457	0.0647
SVMGaus	**0.8931**	**0.9213**	**0.8389**	**0.9474**	**0.7869**	**0.8931**
	0.0728	0.0551	0.1348	0.0602	0.1442	0.0728
SVMLin	**0.8928**	**0.9093**	**0.8611**	**0.9246**	**0.7635**	**0.8928**
	0.0874	0.0700	0.1366	0.0647	0.1828	0.0874
***k*NN**	**0.8899**	**0.9280**	**0.8167**	**0.9632**	**0.7957**	**0.8899**
	0.0854	0.0560	0.1512	0.0411	0.1648	0.0854
SLP	**0.8841**	**0.8960**	**0.8611**	**0.9070**	**0.7336**	**0.8841**
	0.0669	0.0581	0.1080	0.0629	0.1424	0.0669
Naive Bayes	**0.8728**	**0.8587**	**0.9000**	**0.8456**	**0.6658**	**0.8728**
	0.0642	0.0675	0.1018	0.0848	0.1401	0.0642
C4.5	**0.8692**	**0.8907**	**0.8278**	**0.9105**	**0.7136**	**0.8692**
	0.0705	0.0546	0.1325	0.0625	0.1383	0.0705
JRip	**0.8588**	**0.8893**	**0.8000**	**0.9175**	**0.7059**	**0.8588**
	0.0895	0.0694	0.1541	0.0700	0.1764	0.0895
BLR	**0.8423**	**0.8613**	**0.8056**	**0.8789**	**0.6450**	**0.8423**
	0.0915	0.0703	0.1643	0.0746	0.1753	0.0915
RBF-DDA	**0.8339**	**0.8747**	**0.7556**	**0.9123**	**0.6572**	**0.8339**
	0.0910	0.0563	0.1894	0.0638	0.1599	0.0910
OneR	**0.8316**	**0.8653**	**0.7667**	**0.8965**	**0.6425**	**0.8316**
	0.0839	0.0599	0.1614	0.0655	0.1587	0.0839
LDA	**0.7836**	**0.8387**	**0.6778**	**0.8895**	**0.5623**	**0.7836**
	0.0882	0.0652	0.1691	0.0736	0.1753	0.0882

Table 4.18: AIDP vs MF classification using train-test. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
OneR	**0.8667**	**0.8433**	**0.7500**	**0.9833**	**0.6983**	**0.8778**
	0.1191	0.1278	0.1681	0.0913	0.2364	0.0750
JRip	**0.8528**	**0.8333**	**0.7556**	**0.9500**	**0.6715**	**0.8750**
	0.1483	0.1446	0.1499	0.1903	0.2970	0.0726
kNN	**0.8417**	**0.8567**	**0.9167**	**0.7667**	**0.6935**	**0.8417**
	0.0946	0.0844	0.0833	0.1700	0.1864	0.0946
MLP	**0.8347**	**0.8267**	**0.7944**	**0.8750**	**0.6485**	**0.8347**
	0.1216	0.1143	0.1362	0.2050	0.2404	0.1216
Naive Bayes	**0.8319**	**0.8367**	**0.8556**	**0.8083**	**0.6585**	**0.8319**
	0.1089	0.0983	0.1117	0.2009	0.2128	0.1089
C4.5	**0.8236**	**0.8133**	**0.7722**	**0.8750**	**0.6252**	**0.8236**
	0.0634	0.0618	0.1253	0.1548	0.1233	0.0634
SVMGaus	**0.8222**	**0.8200**	**0.8111**	**0.8333**	**0.6290**	**0.8222**
	0.1147	0.1031	0.1366	0.2306	0.2218	0.1147
SVMPoly	**0.8069**	**0.8067**	**0.8056**	**0.8083**	**0.6020**	**0.8069**
	0.1134	0.1048	0.1390	0.2146	0.2225	0.1134
SVMLap	**0.8028**	**0.8033**	**0.8056**	**0.8000**	**0.5942**	**0.8028**
	0.1132	0.1033	0.1390	0.2217	0.2202	0.1132
SLP	**0.7708**	**0.7700**	**0.7667**	**0.7750**	**0.5269**	**0.7708**
	0.1491	0.1368	0.1491	0.2572	0.2894	0.1491
SVMLin	**0.7542**	**0.7567**	**0.7667**	**0.7417**	**0.4948**	**0.7708**
	0.1557	0.1407	0.1587	0.2849	0.3119	0.1244
RBF-DDA	**0.7042**	**0.7367**	**0.8667**	**0.5417**	**0.4198**	**0.7042**
	0.1359	0.1189	0.1269	0.2633	0.2710	0.1359
LDA	**0.6250**	**0.6367**	**0.6833**	**0.5667**	**0.2426**	**0.6472**
	0.1649	0.1520	0.1875	0.3004	0.3234	0.1446
BLR	**0.6111**	**0.6233**	**0.6722**	**0.5500**	**0.2172**	**0.6278**
	0.1403	0.1305	0.1607	0.2491	0.2786	0.1247

Table 4.19: AMAN vs AMSAN classification using train-test. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
kNN	**0.9588**	**0.9570**	**0.9667**	**0.9509**	**0.9101**	**0.9588**
	0.0316	0.0293	0.0553	0.0358	0.0616	0.0316
SVMLap	**0.9406**	**0.9366**	**0.9583**	**0.9228**	**0.8685**	**0.9406**
	0.0347	0.0333	0.0647	0.0493	0.0690	0.0347
MLP	**0.9344**	**0.9333**	**0.9389**	**0.9298**	**0.8608**	**0.9344**
	0.0458	0.0431	0.0654	0.0422	0.0903	0.0458
RBF-DDA	**0.9265**	**0.9387**	**0.8722**	**0.9807**	**0.8674**	**0.9265**
	0.0425	0.0321	0.0972	0.0324	0.0712	0.0425
SVMGaus	**0.9236**	**0.9183**	**0.9472**	**0.9000**	**0.8316**	**0.9236**
	0.0324	0.0336	0.0557	0.0542	0.0682	0.0324
SLP	**0.9202**	**0.9204**	**0.9194**	**0.9211**	**0.8342**	**0.9202**
	0.0404	0.0404	0.0709	0.0614	0.0836	0.0404
SVMPoly	**0.9189**	**0.9118**	**0.9500**	**0.8877**	**0.8191**	**0.9189**
	0.0510	0.0508	0.0864	0.0741	0.1051	0.0510
C4.5	**0.9138**	**0.9151**	**0.9083**	**0.9193**	**0.8233**	**0.9138**
	0.0687	0.0684	0.0994	0.0878	0.1395	0.0687
SVMLin	**0.9125**	**0.9097**	**0.9250**	**0.9000**	**0.8134**	**0.9125**
	0.0445	0.0459	0.0692	0.0684	0.0930	0.0445
Naive Bayes	**0.9091**	**0.9043**	**0.9306**	**0.8877**	**0.8029**	**0.9091**
	0.0434	0.0437	0.0683	0.0619	0.0887	0.0434
LDA	**0.8838**	**0.8839**	**0.8833**	**0.8842**	**0.7577**	**0.8838**
	0.0552	0.0491	0.1150	0.0697	0.1051	0.0552
JRip	**0.8827**	**0.8882**	**0.8583**	**0.9070**	**0.7660**	**0.8827**
	0.0574	0.0597	0.0906	0.0914	0.1209	0.0574
BLR	**0.8559**	**0.8591**	**0.8417**	**0.8702**	**0.7064**	**0.8559**
	0.0780	0.0760	0.1011	0.0803	0.1578	0.0780
OneR	**0.8437**	**0.8323**	**0.8944**	**0.7930**	**0.6615**	**0.8437**
	0.0485	0.0504	0.1048	0.0957	0.0988	0.0485

Table 4.20: AMAN vs MF classification using train-test. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
SVMGaus	**0.9542**	**0.9771**	**1.0000**	**0.9083**	**0.9321**	**0.9542**
	0.0695	0.0348	0.0000	0.1390	0.1049	0.0695
SVMLap	**0.9542**	**0.9771**	**1.0000**	**0.9083**	**0.9321**	**0.9542**
	0.0695	0.0348	0.0000	0.1390	0.1049	0.0695
RBF-DDA	**0.9528**	**0.9542**	**0.9556**	**0.9500**	**0.8867**	**0.9528**
	0.0522	0.0517	0.0717	0.1017	0.1221	0.0522
Naive Bayes	**0.9486**	**0.9479**	**0.9472**	**0.9500**	**0.8709**	**0.9486**
	0.0634	0.0584	0.0663	0.1000	0.1416	0.0634
SVMLin	**0.9431**	**0.9688**	**0.9944**	**0.8917**	**0.9073**	**0.9431**
	0.0823	0.0419	0.0208	0.1669	0.1292	0.0823
kNN	**0.9417**	**0.9708**	**1.0000**	**0.8833**	**0.9152**	**0.9417**
	0.0624	0.0312	0.0000	0.1247	0.0907	0.0624
SVMPoly	**0.9375**	**0.9688**	**1.0000**	**0.8750**	**0.9067**	**0.9375**
	0.0787	0.0394	0.0000	0.1574	0.1207	0.0787
MLP	**0.9125**	**0.9370**	**0.8389**	**0.9861**	**0.8502**	**0.9125**
	0.0704	0.0500	0.1417	0.0384	0.1223	0.0704
SLP	**0.9056**	**0.9250**	**0.9444**	**0.8667**	**0.8040**	**0.9056**
	0.0709	0.0503	0.0799	0.1704	0.1260	0.0709
LDA	**0.9056**	**0.9125**	**0.9194**	**0.8917**	**0.7821**	**0.9056**
	0.0966	0.0797	0.0966	0.1820	0.1925	0.0966
C4.5	**0.8889**	**0.9125**	**0.9361**	**0.8417**	**0.7615**	**0.8889**
	0.1115	0.0696	0.0735	0.2281	0.2121	0.1115
BLR	**0.8833**	**0.8917**	**0.9000**	**0.8667**	**0.7322**	**0.8833**
	0.1080	0.0898	0.1036	0.1940	0.2118	0.1080
oneR	**0.8806**	**0.9042**	**0.9278**	**0.8333**	**0.7357**	**0.8806**
	0.1146	0.0608	0.0647	0.2486	0.2065	0.1146
JRip	**0.8792**	**0.9104**	**0.9417**	**0.8167**	**0.7581**	**0.8792**
	0.1068	0.0746	0.0958	0.2268	0.1980	0.1068

Table 4.21 shows the average prediction results of each classifier across 30 runs in AMSAN vs MF classification using train-test. The standard deviation of each metric is also shown. Balanced accuracy was lower than accuracy. The balanced accuracy ranged 0.67 - 0.89, and accuracy ranged 0.86 - 0.92. High values were obtained in sensitivity, ranging 0.88 - 0.96. In contrast, specificity showed a large variation, ranging 0.39 - 0.90. This also was the case in Kappa statistic, ranging 0.39 - 0.73. ROC curves ranged 0.67 - 0.89. Only three classifiers obtained a balanced accuracy above 0.85. In short, most classifiers in AMSAN vs MF classification using train-test had a poor performance.

Figure 4.9 shows the median ROC curve values of the top five classifiers in OVO classification using train-test. That is, from each OVO classification average results table, the top five classifiers were selected. Then, across 30 ROC curve values, the median value was selected of each of the top five classifiers. The median value was selected to illustrate the fairest result across 30 runs. The highest median ROC curve values using train-test were obtained in AMAN vs MF classification, ranging 0.95 - 1.00. Second best was AIDP vs AMAN, ranging 0.91 - 1.00. The worst median ROC curve values were obtained in AIDP vs MF, ranging 0.83 - 0.87. Overall, with the top five classifiers in each OVO classification scenario using train-test, median ROC curve values on or above 0.80 were obtained.

Figure 4.10 shows the average balanced accuracy across 30 runs for each classifier in OVO classification using train-test. The standard deviation is also shown. AMAN vs MF classification showed the highest performance in balanced accuracy across 30 runs. The opposite case was AIDP vs MF. Regarding the variations in performance across 30 runs, AMSAN vs MF obtained the highest variation in balanced accuracy; on the other hand, AIDP vs AMSAN was the most stable.

Table 4.22 shows the average prediction results of each classifier across 30 runs in AIDP vs AMAN classification using 10-FCV. The standard deviation of each metric is also shown. Balanced accuracy behaved very similar to accuracy. The balanced accuracy ranged 0.87 - 0.95, and accuracy ranged 0.89 - 0.96. Sensitivity showed some variation, ranging 0.75 - 0.97. Also did specificity, which ranged 0.87 - 1.00. Kappa statistic ranged 0.76 - 0.91. High results were obtained in ROC curves, ranging 0.87 - 0.95. Twelve classifiers obtained a balanced accuracy ≥ 0.90. In summary, AIDP vs AMAN classification using 10-FCV had a high performance with most classifiers.

Table 4.23 shows the average prediction results of each classifier across 30 runs in AIDP vs AMSAN classification using 10-FCV. The standard deviation of each metric is also shown. Both balanced accuracy and accuracy behaved similarly. The balanced accuracy ranged 0.82 - 0.92, and accuracy ranged 0.84 - 0.94. Sensitivity values were a little lower than specificity values. Sensitivity ranged 0.73 - 0.88. On the other hand, specificity ranged 0.85 - 0.98. Kappa statistic ranged 0.63 - 0.86. High values were obtained in ROC curves, ranging 0.82 - 0.92. Four classifiers obtained a balanced accuracy above 0.90. Overall, most classifiers had a high performance in AIDP vs AMSAN classification using 10-FCV.

Table 4.24 shows the average prediction results of each classifier across 30 runs in AIDP vs MF classification using 10-FCV. The standard deviation of each metric is also shown. Balanced accuracy values were very similar to those of accuracy. The balanced accuracy ranged 0.65 - 0.87, and accuracy ranged 0.65 - 0.85. Values obtained in sensitivity were a little lower than those obtained in

Table 4.21: AMSAN vs MF classification using train-test. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
Naive Bayes	**0.8980**	**0.8913**	**0.8877**	**0.9083**	**0.6846**	**0.8980**
	0.0752	0.0547	0.0588	0.1367	0.1499	0.0752
LDA	**0.8656**	**0.9246**	**0.9561**	**0.7750**	**0.7372**	**0.8656**
	0.0954	0.0522	0.0619	0.2008	0.1655	0.0954
MLP	**0.8588**	**0.8971**	**0.9175**	**0.8000**	**0.6636**	**0.8588**
	0.1053	0.0529	0.0530	0.2117	0.1794	0.1053
SLP	**0.8493**	**0.8870**	**0.9070**	**0.7917**	**0.6369**	**0.8493**
	0.1084	0.0555	0.0644	0.2282	0.1792	0.1084
SVMGaus	**0.8480**	**0.9174**	**0.9544**	**0.7417**	**0.6996**	**0.8480**
	0.0921	0.0349	0.0301	0.1912	0.1420	0.0921
SVMLap	**0.8316**	**0.9174**	**0.9632**	**0.7000**	**0.6880**	**0.8316**
	0.1016	0.0385	0.0343	0.2117	0.1585	0.1016
JRip	**0.8248**	**0.8899**	**0.9246**	**0.7250**	**0.6232**	**0.8248**
	0.0982	0.0438	0.0597	0.2212	0.1528	0.0982
kNN	**0.8099**	**0.9087**	**0.9614**	**0.6583**	**0.6476**	**0.8099**
	0.1023	0.0343	0.0331	0.2187	0.1580	0.1023
BLR	**0.8042**	**0.8667**	**0.9000**	**0.7083**	**0.5575**	**0.8042**
	0.1291	0.0655	0.0624	0.2550	0.2240	0.1291
C4.5	**0.8026**	**0.8696**	**0.9053**	**0.7000**	**0.5615**	**0.8026**
	0.1259	0.0615	0.0671	0.2614	0.2120	0.1259
SVMLin	**0.8013**	**0.9000**	**0.9526**	**0.6500**	**0.6201**	**0.8013**
	0.1219	0.0516	0.0437	0.2466	0.2069	0.1219
SVMPoly	**0.7974**	**0.9043**	**0.9614**	**0.6333**	**0.6255**	**0.7974**
	0.1112	0.0386	0.0337	0.2343	0.1761	0.1112
OneR	**0.7811**	**0.8667**	**0.9123**	**0.6500**	**0.5363**	**0.7811**
	0.1662	0.0877	0.0845	0.3256	0.2982	0.1662
RBF-DDA	**0.6757**	**0.8609**	**0.9596**	**0.3917**	**0.3903**	**0.6757**
	0.1310	0.0490	0.0357	0.2682	0.2640	0.1310

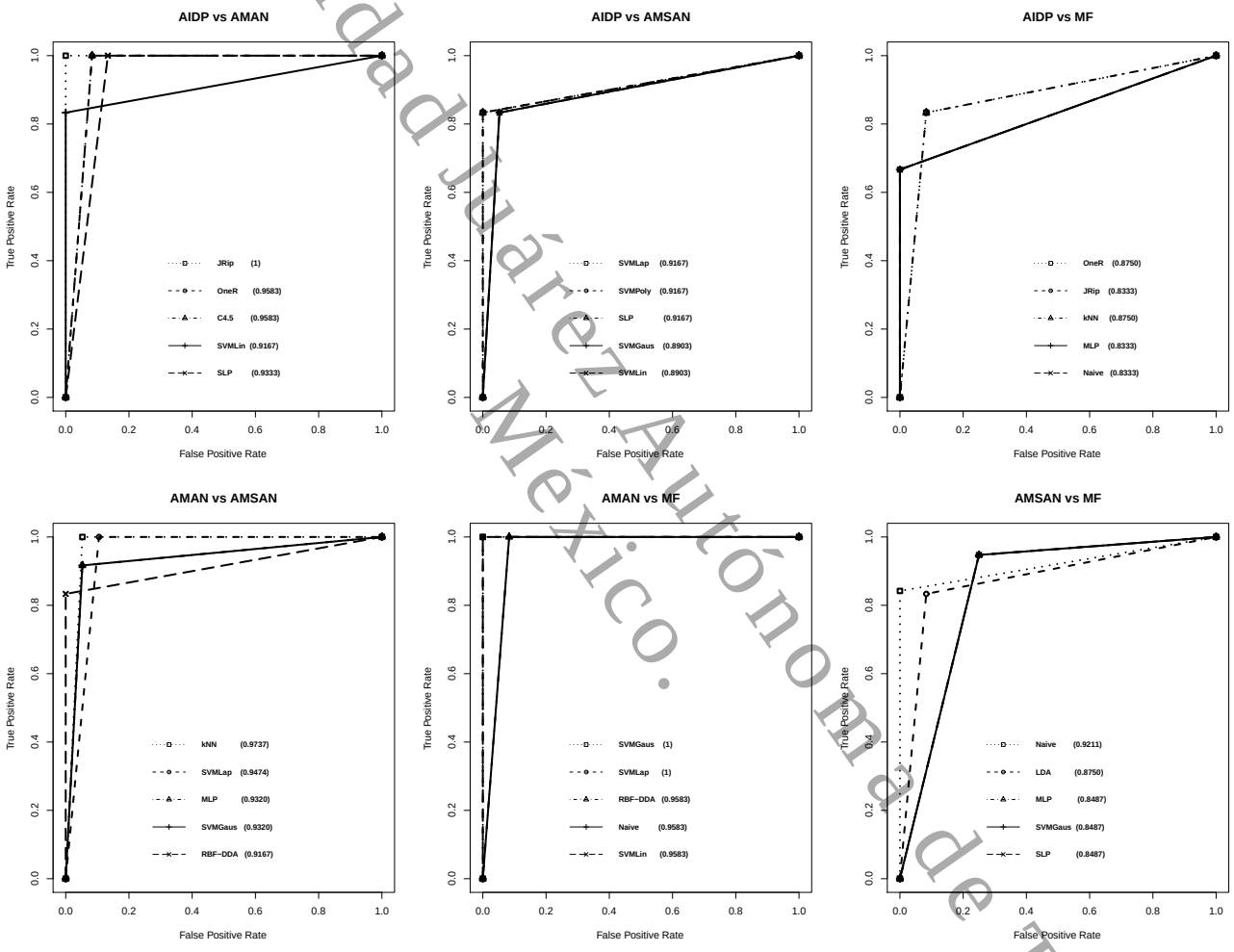

Figure 4.9: ROC curves of the best five classifiers in OVO classification using train-test.

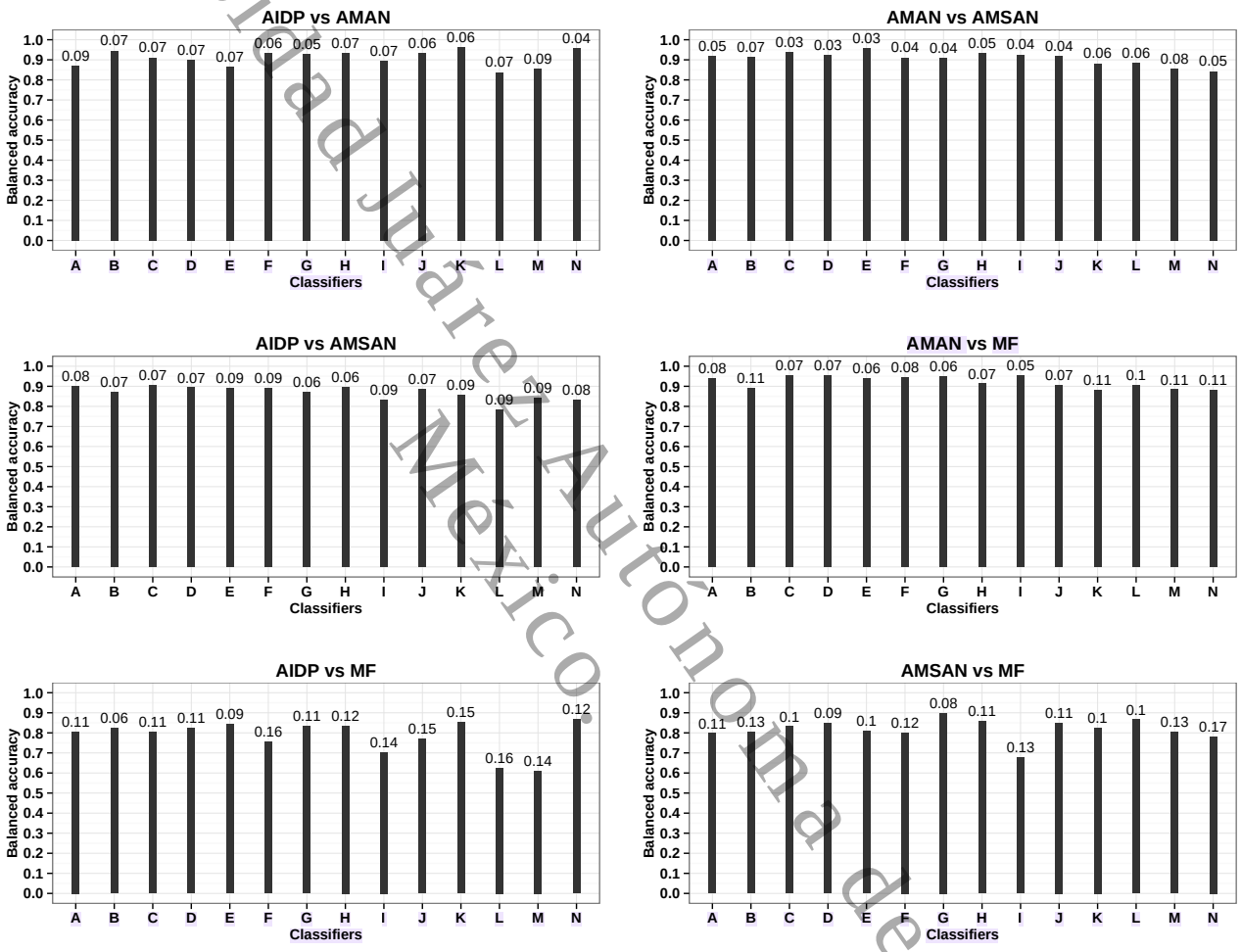

Figure 4.10: Balanced accuracy in OVO classification using train-test. A=SVMPoly, B=C4.5, C=SVMLap, D=SVMGaus, E=kNN, F=SVMLin, G=Naive Bayes, H=MLP, I=RBF-DDA, J=SLP, K=JRip, L=LDA, M=BLR, N=OneR.

Table 4.22: AIDP vs AMAN classification using 10-FCV. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
JRip	**0.9578**	**0.9613**	**0.9400**	**0.9756**	**0.9161**	**0.9578**
	0.0150	0.0148	0.0203	0.0174	0.0322	0.0150
SVMLin	**0.9536**	**0.9593**	**0.9250**	**0.9822**	**0.9112**	**0.9536**
	0.0187	0.0186	0.0254	0.0227	0.0383	0.0187
MLP	**0.9536**	**0.9627**	**0.9083**	**0.9989**	**0.9139**	**0.9536**
	0.0175	0.0146	0.0324	0.0061	0.0342	0.0175
C4.5	**0.9508**	**0.9500**	**0.9550**	**0.9467**	**0.8985**	**0.9508**
	0.0134	0.0146	0.0153	0.0225	0.0282	0.0134
OneR	**0.9464**	**0.9467**	**0.9450**	**0.9478**	**0.8896**	**0.9464**
	0.0133	0.0142	0.0153	0.0209	0.0298	0.0133
SLP	**0.9447**	**0.9527**	**0.9050**	**0.9844**	**0.8936**	**0.9447**
	0.0218	0.0193	0.0402	0.0190	0.0437	0.0218
SVMGaus	**0.9272**	**0.9407**	**0.8600**	**0.9944**	**0.8642**	**0.9272**
	0.0153	0.0144	0.0242	0.0154	0.0323	0.0153
SVMLap	**0.9239**	**0.9380**	**0.8533**	**0.9944**	**0.8575**	**0.9239**
	0.0115	0.0110	0.0183	0.0126	0.0250	0.0115
Naive Bayes	**0.9228**	**0.9133**	**0.9700**	**0.8756**	**0.8319**	**0.9228**
	0.0204	0.0212	0.0249	0.0289	0.0418	0.0204
SVMPoly	**0.9108**	**0.9287**	**0.8217**	**1.0000**	**0.8345**	**0.9108**
	0.0126	0.0101	0.0252	0.0000	0.0238	0.0126
RBF-DDA	**0.9050**	**0.9127**	**0.8667**	**0.9433**	**0.8121**	**0.9050**
	0.0233	0.0232	0.0330	0.0292	0.0487	0.0233
LDA	**0.9000**	**0.9176**	**0.8120**	**0.9880**	**0.8127**	**0.9007**
	0.0201	0.0176	0.0389	0.0190	0.0403	0.0192
BLR	**0.8894**	**0.8987**	**0.8433**	**0.9356**	**0.7836**	**0.8922**
	0.0342	0.0340	0.0504	0.0446	0.0694	0.0318
***k*NN**	**0.8750**	**0.9000**	**0.7500**	**1.0000**	**0.7667**	**0.8750**
	0.0000	0.0000	0.0000	0.0000	0.0050	0.0000

Table 4.23: AIDP vs AMSAN classification using 10-FCV. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
SVMPoly	**0.9217**	**0.9495**	**0.8567**	**0.9867**	**0.8622**	**0.9217**
	0.0172	0.0129	0.0314	0.0121	0.0321	0.0172
SVMGaus	**0.9217**	**0.9381**	**0.8833**	**0.9600**	**0.8433**	**0.9217**
	0.0173	0.0165	0.0273	0.0197	0.0387	0.0173
SVMLap	**0.9057**	**0.9281**	**0.8533**	**0.9580**	**0.8177**	**0.9057**
	0.0110	0.0133	0.0127	0.0177	0.0266	0.0110
kNN	**0.9007**	**0.9224**	**0.8500**	**0.9513**	**0.8035**	**0.9007**
	0.0057	0.0081	0.0000	0.0114	0.0190	0.0057
OneR	**0.8873**	**0.9105**	**0.8333**	**0.9413**	**0.7756**	**0.8873**
	0.0182	0.0154	0.0330	0.0181	0.0365	0.0182
C4.5	**0.8792**	**0.9038**	**0.8217**	**0.9367**	**0.7600**	**0.8792**
	0.0235	0.0212	0.0409	0.0258	0.0472	0.0235
MLP	**0.8695**	**0.8829**	**0.8383**	**0.9007**	**0.7247**	**0.8698**
	0.0256	0.0269	0.0313	0.0317	0.0588	0.0257
Naive Bayes	**0.8680**	**0.8614**	**0.8833**	**0.8527**	**0.6958**	**0.8687**
	0.0182	0.0195	0.0240	0.0243	0.0404	0.0185
SVM Linear	**0.8677**	**0.8810**	**0.8367**	**0.8987**	**0.7189**	**0.8690**
	0.0237	0.0207	0.0414	0.0240	0.0466	0.0227
SLP	**0.8630**	**0.8786**	**0.8267**	**0.8993**	**0.7119**	**0.8630**
	0.0313	0.0267	0.0487	0.0260	0.0637	0.0313
JRip	**0.8372**	**0.8824**	**0.7317**	**0.9427**	**0.6851**	**0.8372**
	0.0365	0.0275	0.0636	0.0233	0.0759	0.0365
BLR	**0.8303**	**0.8476**	**0.7900**	**0.8707**	**0.6427**	**0.8320**
	0.0308	0.0271	0.0481	0.0282	0.0623	0.0307
LDA	**0.8232**	**0.8510**	**0.7583**	**0.8880**	**0.6367**	**0.8238**
	0.0183	0.0166	0.0373	0.0238	0.0422	0.0185
RBF-DDA	**0.8232**	**0.8481**	**0.7650**	**0.8813**	**0.6306**	**0.8232**
	0.0279	0.0207	0.0494	0.0174	0.0579	0.0279

specificity. Sensitivity ranged 0.67 - 0.86. On the other hand, specificity ranged 0.55 - 1.00. Values obtained in Kappa statistic were low, ranging 0.28 - 0.71. In contrast, high values were obtained in ROC curves, ranging 0.75 - 0.87. Five classifiers obtained a balanced accuracy above 0.85. In short, in AIDP vs MF classification using 10-FCV were obtained high results in half the metrics and low values in the remaining metrics.

Table 4.25 shows the average prediction results of each classifier across 30 runs in AMAN vs AMSAN classification using 10-FCV. The standard deviation of each metric is also shown. Both balanced accuracy and accuracy behaved similarly. The balanced accuracy ranged 0.81 - 0.95, and accuracy ranged 0.79 - 0.95. Sensitivity values were very similar to those of specificity. Sensitivity ranged 0.86 - 0.96. Specificity ranged 0.75 - 0.98, with most cases above 0.87. High values were obtained in Kappa statistic, ranging 0.77 - 0.89, with one case under 0.60. Also, high values were obtained in ROC curves obtained, ranging 0.81 - 0.95. Twelve classifiers obtained a balanced accuracy above 0.90. Overall, most classifiers highly performed in AMAN vs AMSAN classification using 10-FCV.

Table 4.26 shows the average prediction results of each classifier across 30 runs in AMAN vs MF classification using 10-FCV. The standard deviation of each metric is also shown. Values obtained in balanced accuracy were very similar to those obtained in accuracy. The balanced accuracy ranged 0.88 - 0.96, and accuracy ranged 0.88 - 0.98. Sensitivity values were a little higher than those of specificity. Sensitivity ranged 0.88 - 1.00. Specificity ranged 0.84 - 0.92. High values were obtained in Kappa statistic, ranging 0.71 - 0.92. High values were obtained in ROC curves also, ranging 0.88 - 0.96. Twelve classifiers obtained a balanced accuracy above 0.90. In summary, most classifiers obtained values on or above 0.80 in most of the metrics in AMAN vs MF classification using 10-FCV.

Table 4.27 shows the average prediction results of each classifier across 30 runs in AMSAN vs MF classification using 10-FCV. The standard deviation of each metric is also shown. Balanced accuracy values were a little lower than those of accuracy. The balanced accuracy ranged 0.67 - 0.90, while accuracy ranged 0.85 - 0.94. Sensitivity values were higher than those of specificity. Sensitivity ranged 0.89 - 0.97. Specificity ranged 0.40 - 0.92. Kappa statistic showed a large variation, ranging 0.31 - 0.78. ROC curves ranged 0.67 - 0.90. Five classifiers obtained a balanced accuracy above 0.85. In summary, AMSAN vs MF classification using 10-FCV obtained high values with five classifiers only in four metrics.

Figure 4.11 shows the average balanced accuracy across 30 runs for each classifier in OVO classification using 10-FCV. The standard deviation is also shown. As in train-test, AMAN vs MF classification showed the highest performance in balanced accuracy across 30 runs. The opposite case was AIDP vs MF, again as in train-test. Regarding the variations in performance across 30 runs, AIDP vs MF obtained the highest variation in balanced accuracy; on the other hand, all classifications with AMAN present were the most stable.

4.4.3.4 Statistical analysis

We investigated if there was any statistically significant difference in average accuracy among the top five classifiers in average accuracy (balanced accuracy in OVO and OVA scenarios) across 30 runs, using train-test and 10-FCV in all classification scenarios. For this analysis, we used the Friedman

Table 4.24: AIDP vs MF classification using 10-FCV. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
OneR	**0.8758**	**0.8344**	**0.7517**	**1.0000**	**0.7113**	**0.8758**
	0.0046	0.0061	0.0091	0.0000	0.0136	0.0046
SVMPoly	**0.8675**	**0.8467**	**0.8050**	**0.9300**	**0.7020**	**0.8775**
	0.0247	0.0188	0.0153	0.0466	0.0480	0.0257
SVMLap	**0.8625**	**0.8422**	**0.8017**	**0.9233**	**0.6920**	**0.8725**
	0.0284	0.0194	0.0091	0.0568	0.0547	0.0310
JRip	**0.8600**	**0.8244**	**0.7533**	**0.9667**	**0.6817**	**0.8683**
	0.0251	0.0194	0.0183	0.0479	0.0491	0.0173
SVMGaus	**0.8525**	**0.8322**	**0.7917**	**0.9133**	**0.6727**	**0.8625**
	0.0337	0.0255	0.0190	0.0629	0.0661	0.0340
MLP	**0.8475**	**0.8300**	**0.7950**	**0.9000**	**0.6640**	**0.8608**
	0.0412	0.0282	0.0153	0.0830	0.0822	0.0375
Naive Bayes	**0.8458**	**0.8500**	**0.8583**	**0.8333**	**0.6683**	**0.8642**
	0.0455	0.0325	0.0190	0.0884	0.0916	0.0463
kNN	**0.8275**	**0.8411**	**0.8683**	**0.7867**	**0.6357**	**0.8525**
	0.0343	0.0243	0.0245	0.0730	0.0670	0.0356
SLP	**0.8208**	**0.8033**	**0.7683**	**0.8733**	**0.6080**	**0.8408**
	0.0492	0.0385	0.0382	0.0944	0.0982	0.0407
C4.5	**0.7742**	**0.7667**	**0.7517**	**0.7967**	**0.5140**	**0.8042**
	0.0418	0.0277	0.0091	0.0850	0.0812	0.0416
LDA	**0.7417**	**0.7400**	**0.7367**	**0.7467**	**0.4557**	**0.8067**
	0.0614	0.0441	0.0292	0.1196	0.1173	0.0517
RBF-DDA	**0.7017**	**0.7522**	**0.8533**	**0.5500**	**0.3863**	**0.7533**
	0.0633	0.0469	0.0260	0.1167	0.1238	0.0512
SVMLin	**0.6600**	**0.6911**	**0.7533**	**0.5667**	**0.2973**	**0.7633**
	0.0586	0.0471	0.0346	0.0994	0.1143	0.0370
BLR	**0.6508**	**0.6578**	**0.6717**	**0.6300**	**0.2800**	**0.7842**
	0.0589	0.0479	0.0449	0.1055	0.1170	0.0527

Table 4.25: AMAN vs AMSAN classification using 10-FCV. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
kNN	**0.9504**	**0.9458**	**0.9689**	**0.9320**	**0.8892**	**0.9504**
	0.0079	0.0083	0.0122	0.0124	0.0163	0.0079
SVMLap	**0.9472**	**0.9421**	**0.9678**	**0.9267**	**0.8816**	**0.9472**
	0.0164	0.0169	0.0239	0.0237	0.0333	0.0164
RBF-DDA	**0.9461**	**0.9546**	**0.9122**	**0.9800**	**0.8992**	**0.9461**
	0.0197	0.0156	0.0396	0.0129	0.0354	0.0197
C4.5	**0.9359**	**0.9354**	**0.9378**	**0.9340**	**0.8645**	**0.9359**
	0.0171	0.0143	0.0324	0.0140	0.0320	0.0171
SVMPoly	**0.9284**	**0.9200**	**0.9622**	**0.8947**	**0.8399**	**0.9284**
	0.0172	0.0182	0.0259	0.0273	0.0346	0.0172
Naive Bayes	**0.9261**	**0.9171**	**0.9622**	**0.8900**	**0.8334**	**0.9261**
	0.0176	0.0166	0.0273	0.0188	0.0350	0.0176
MLP	**0.9259**	**0.9254**	**0.9278**	**0.9240**	**0.8439**	**0.9259**
	0.0155	0.0152	0.0264	0.0213	0.0322	0.0155
SLP	**0.9233**	**0.9217**	**0.9300**	**0.9167**	**0.8377**	**0.9233**
	0.0243	0.0220	0.0395	0.0223	0.0473	0.0243
SVMGaus	**0.9201**	**0.9113**	**0.9556**	**0.8847**	**0.8218**	**0.9201**
	0.0186	0.0190	0.0267	0.0256	0.0369	0.0186
SVMLin	**0.9184**	**0.9200**	**0.9122**	**0.9247**	**0.8320**	**0.9184**
	0.0187	0.0185	0.0321	0.0266	0.0379	0.0187
JRip	**0.9041**	**0.9138**	**0.8656**	**0.9427**	**0.8118**	**0.9041**
	0.0275	0.0242	0.0521	0.0291	0.0544	0.0275
LDA	**0.9020**	**0.8942**	**0.9333**	**0.8707**	**0.7848**	**0.9020**
	0.0195	0.0182	0.0328	0.0221	0.0389	0.0195
BLR	**0.8903**	**0.8896**	**0.8933**	**0.8873**	**0.7703**	**0.8903**
	0.0242	0.0235	0.0386	0.0299	0.0494	0.0242
OneR	**0.8100**	**0.7950**	**0.8700**	**0.7500**	**0.5893**	**0.8109**
	0.0312	0.0272	0.0609	0.0335	0.0575	0.0297

Table 4.26: AMAN vs MF classification using 10-FCV. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
*k*NN	**0.9600**	**0.9800**	**1.0000**	**0.9200**	**0.9200**	**0.9600**
	0.0203	0.0102	0.0000	0.0407	0.0407	0.0203
SVMGaus	**0.9600**	**0.9800**	**1.0000**	**0.9200**	**0.9200**	**0.9600**
	0.0203	0.0102	0.0000	0.0407	0.0407	0.0203
SVMLap	**0.9600**	**0.9800**	**1.0000**	**0.9200**	**0.9200**	**0.9600**
	0.0203	0.0102	0.0000	0.0407	0.0407	0.0203
SVMPoly	**0.9567**	**0.9783**	**1.0000**	**0.9133**	**0.9133**	**0.9567**
	0.0173	0.0086	0.0000	0.0346	0.0346	0.0173
SVMLin	**0.9517**	**0.9692**	**0.9867**	**0.9167**	**0.8979**	**0.9517**
	0.0225	0.0182	0.0188	0.0379	0.0491	0.0225
RBF-DDA	**0.9506**	**0.9658**	**0.9811**	**0.9200**	**0.8923**	**0.9506**
	0.0203	0.0154	0.0189	0.0407	0.0424	0.0203
Naive Bayes	**0.9333**	**0.9400**	**0.9467**	**0.9200**	**0.8424**	**0.9333**
	0.0191	0.0169	0.0241	0.0407	0.0404	0.0191
BLR	**0.9228**	**0.9292**	**0.9356**	**0.9100**	**0.8188**	**0.9228**
	0.0351	0.0246	0.0247	0.0662	0.0742	0.0351
JRip	**0.9167**	**0.9233**	**0.9300**	**0.9033**	**0.8048**	**0.9167**
	0.0356	0.0278	0.0354	0.0718	0.0750	0.0356
LDA	**0.9106**	**0.9392**	**0.9678**	**0.8533**	**0.8080**	**0.9106**
	0.0275	0.0193	0.0185	0.0507	0.0573	0.0275
SLP	**0.9089**	**0.9417**	**0.9744**	**0.8433**	**0.8099**	**0.9111**
	0.0309	0.0249	0.0286	0.0568	0.0638	0.0301
MLP	**0.9072**	**0.9375**	**0.9678**	**0.8467**	**0.8020**	**0.9094**
	0.0309	0.0269	0.0297	0.0507	0.0682	0.0302
OneR	**0.8833**	**0.8850**	**0.8867**	**0.8800**	**0.7200**	**0.8833**
	0.0280	0.0193	0.0207	0.0551	0.0605	0.0280
C4.5	**0.8800**	**0.8883**	**0.8967**	**0.8633**	**0.7141**	**0.8800**
	0.0271	0.0215	0.0320	0.0615	0.0574	0.0271

Table 4.27: AMSAN vs MF classification using 10-FCV. The units of the results are shown in average over 30 runs.

Classifier	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
Naive Bayes	**0.9073**	**0.8967**	**0.8913**	**0.9233**	**0.7161**	**0.9073**
	0.0216	0.0172	0.0208	0.0430	0.0506	0.0216
LDA	**0.9067**	**0.9489**	**0.9700**	**0.8433**	**0.7887**	**0.9067**
	0.0351	0.0195	0.0155	0.0626	0.0721	0.0351
MLP	**0.8750**	**0.9117**	**0.9300**	**0.8200**	**0.6912**	**0.8750**
	0.0459	0.0215	0.0172	0.0887	0.0883	0.0459
BLR	**0.8673**	**0.9056**	**0.9247**	**0.8100**	**0.6764**	**0.8673**
	0.0552	0.0278	0.0245	0.1062	0.1046	0.0552
SVMGaus	**0.8657**	**0.9206**	**0.9480**	**0.7833**	**0.6884**	**0.8657**
	0.0453	0.0173	0.0124	0.0913	0.0862	0.0453
kNN	**0.8473**	**0.9144**	**0.9480**	**0.7467**	**0.6495**	**0.8473**
	0.0372	0.0137	0.0124	0.0776	0.0728	0.0372
SVMLap	**0.8460**	**0.9144**	**0.9487**	**0.7433**	**0.6483**	**0.8460**
	0.0352	0.0162	0.0125	0.0679	0.0742	0.0352
SLP	**0.8447**	**0.8989**	**0.9260**	**0.7633**	**0.6279**	**0.8447**
	0.0441	0.0219	0.0224	0.0890	0.0825	0.0441
C4.5	**0.8413**	**0.8911**	**0.9160**	**0.7667**	**0.6114**	**0.8413**
	0.0432	0.0209	0.0185	0.0844	0.0820	0.0432
JRip	**0.8410**	**0.9017**	**0.9320**	**0.7500**	**0.6304**	**0.8410**
	0.0502	0.0260	0.0214	0.0938	0.0981	0.0502
SVMLin	**0.8043**	**0.9006**	**0.9487**	**0.6600**	**0.5679**	**0.8043**
	0.0396	0.0178	0.0180	0.0814	0.0720	0.0396
SVMPoly	**0.7960**	**0.8978**	**0.9487**	**0.6433**	**0.5514**	**0.7960**
	0.0455	0.0162	0.0125	0.0935	0.0865	0.0455
OneR	**0.7577**	**0.8539**	**0.9020**	**0.6133**	**0.4609**	**0.7597**
	0.0413	0.0302	0.0358	0.0819	0.0871	0.0443
RBF-DDA	**0.6737**	**0.8561**	**0.9473**	**0.4000**	**0.3151**	**0.6737**
	0.0456	0.0183	0.0134	0.0910	0.0861	0.0456

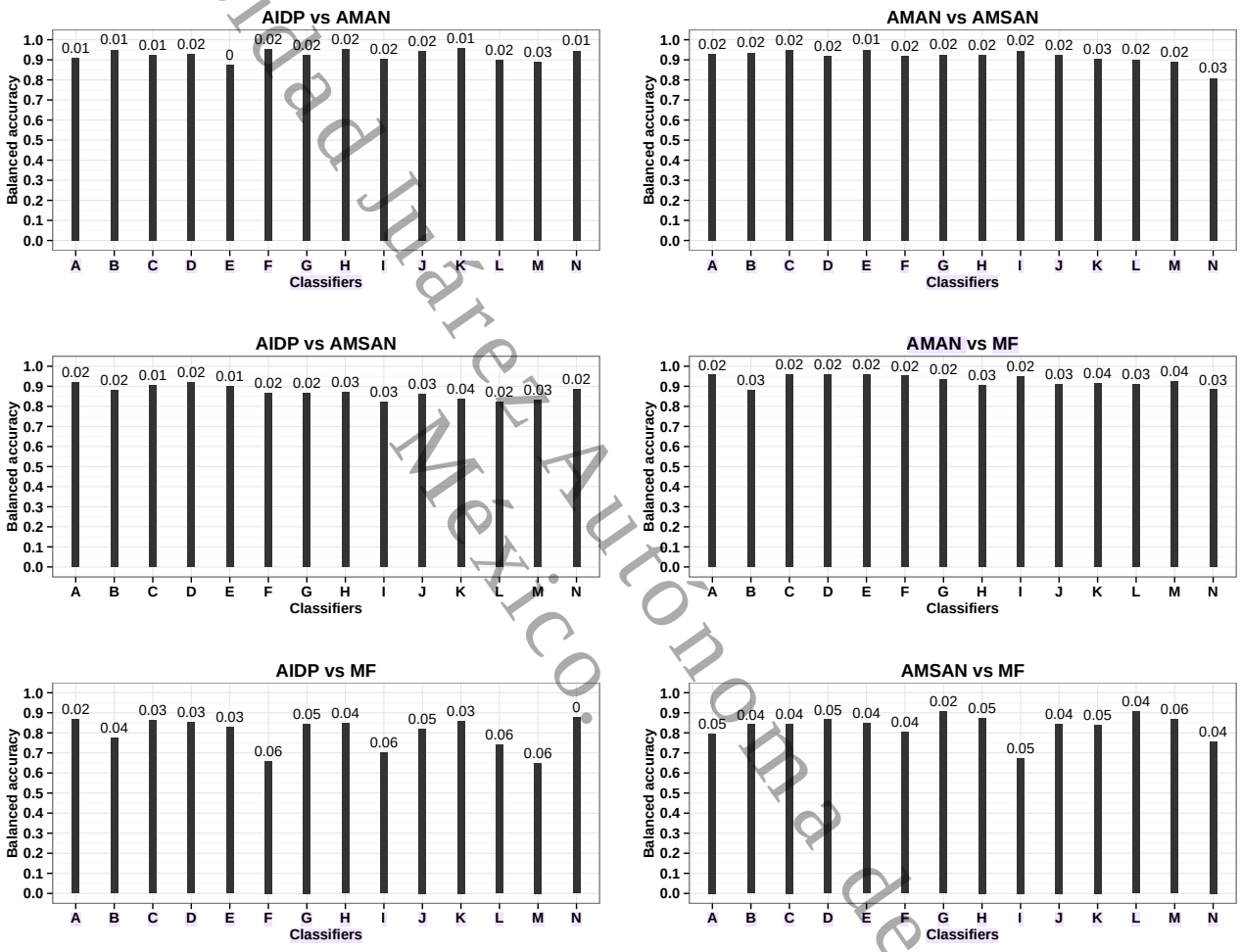

Figure 4.11: Balanced accuracy in OVO classification using 10-FCV. A=SVMPoly, B=C4.5, C=SVMLap, D=SVMGaus, E=kNN, F=SVMLin, G=Naive Bayes, H=MLP, I=RBF-DDA, J=SLP, K=JRip, L=LDA, M=BLR, N=OneR.

test. An additional post hoc analysis using Holms' correction was performed in cases where null hypothesis was rejected. We selected Friedman test since it is suitable for the type of analysis we performed and also because it is a non parametric test, that is, no assumption about data distribution is needed. Holm's correction is used for controlling the family-wise error in multiple hypothesis testing. Despite another correction procedures, we selected Holm's because it is a powerful method and it makes no additional assumption about the hypotheses tested. More details about these tests can be found in [30].

Post hoc analysis uses an alpha* parameter, which is the modified alpha value equal to alpha/(k-i), where alpha is the significance level, k is the number of classifiers and i is the rank. In all tests, we used an alpha = 0.05.

4.4.3.4.1 Four GBS subtype classification

In table 4.28 we show the Friedman test results of the comparison among the top five classifiers in average accuracy across 30 runs, using train-test and 10-FCV in four GBS subtype classification. The complete list of the top five classifiers for each case is shown in Table 4.29. In the train-test scenario, a statistically significant difference among the top five classifiers in average accuracy across 30 runs was found. On the other hand, in the 10-FCV scenario, the opposite result was found.

In all cases, we used as our null hypothesis Ho: There is no statistically significant difference in the average accuracy among the top five classifiers across 30 runs, and our alternative hypothesis H_1: There is a statistically significant difference in the average accuracy among the top five classifiers across 30 runs.

Table 4.28: Friedman test results of the comparison among top five classifiers in average accuracy across 30 runs in four GBS subtype classification using train-test and 10-FCV.

Friedman Statistic	**Critical value**	**Ho**	**Scenario**
2.707	2.45	rejected	Train-test
1.985	2.45	accepted	10-FCV

In table 4.29 we show the average ranks for the top five classifiers using train-test and 10-FCV in four GBS subtype classification. On average, in train-test scenario, kNN was the ranked first (2.20).

Table 4.29: Average ranks of the top five classifiers using train-test and 10-FCV in 4 classes classification.

	Classifiers compared					**Scenario**
	kNN	SVMLap	SVMPoly	SVMGaus	C4.5	Train-test
Average ranks	2.20	3.05	3.35	3.27	3.13	
	SVMPoly	C4.5	SVMLap	SVMGaus	kNN	10-FCV
Average ranks	2.47	2.87	2.87	3.38	3.42	

In Table 4.30, the results of the post-hoc test with Holm's correction of the top five classifiers

in four GBS subtype classification using train-test are shown. A statistically significant difference between kNN and the other four classifiers was found.

Table 4.30: Post-hoc test with Holm's correction of the top five classifiers in four GBS subtype classification using train-test.

Classifiers compared		p value	alpha*	Ho	Scenario
kNN vs	SVMPoly	0.005	0.013	rejected	Train-test
kNN vs	SVMGaus	0.009	0.017	rejected	Train-test
kNN vs	C4.5	0.022	0.025	rejected	Train-test
kNN vs	SVMLap	0.037	0.050	rejected	Train-test

4.4.3.4.2 OVA classification

In Table 4.31 we show the Friedman test results of the comparison among the top five classifiers in balanced accuracy across 30 runs, using train-test and 10-FCV in OVA classification. The complete list of the top five classifiers for each case is shown in Table 4.32. In train-test scenario, a statistically significant difference among the top five classifiers in balanced accuracy across 30 runs was found in half OVA classifications. Theses cases were AIDP vs ALL and MF vs ALL. In 10-FCV, a statistically significant difference among the top five classifiers in balanced accuracy across 30 runs was found in all OVA classifications.

In all cases, we used as our null hypothesis Ho: There is no statistically significant difference in the balanced accuracy among the top five classifiers across 30 runs, and our alternative hypothesis H_1: There is a statistically significant difference in the balanced accuracy among the top five classifiers across 30 runs.

Table 4.31: Friedman test results of the comparison among top five classifiers in balanced accuracy across 30 runs in OVA classification using train-test and 10-FCV.

Classes	Friedman Statistic	Critical value	Ho	Scheme
AIDP vs ALL	5.41	2.45	rejected	Train-test
	8.989	2.45	rejected	10-FCV
AMAN vs ALL	1.533	2.45	accepted	Train-test
	8.651	2.45	rejected	10-FCV
AMSAN vs ALL	1.746	2.45	accepted	Train-test
	35.869	2.45	rejected	10-FCV
MF vs ALL	9.935	2.45	rejected	Train-test
	25.591	2.45	rejected	10-FCV

In table 4.32 we show the average ranks for the top five classifiers using train-test and 10-FCV in OVA classification. In both train-test and 10-FCV, we highlight the ranked first classifiers only in cases where a statistically significant difference was found. On average, in train-test scenario, the ranked first classifiers were kNN (2.18) for AIDP vs ALL classification and Naive Bayes (1.63) for

MF vs ALL. In 10-FCV, the ranked first classifiers were MLP (2.17) for AIDP vs ALL, SVMPoly (2.37) for AMAN vs ALL, kNN (1.40) for AMSAN vs ALL, and Naive Bayes (1.20) for MF vs ALL.

Table 4.32: Average ranks of the top five classifiers using train-test and 10-FCV in OVA classification.

Classes	Classifiers compared					Scenario
AIDP vs ALL	kNN	C4.5	MLP	SVMLap	RBF-DDA	Train-test
Average ranks	2.18	2.98	3.13	2.78	3.92	
	MLP	SVMLap	C4.5	kNN	LDA	10-FCV
Average ranks	2.17	2.38	2.92	3.53	4.00	
AMAN vs ALL	kNN	SVMGaus	SVMPoly	SVMLap	C4.5	Train-test
Average ranks	2.55	2.68	3.25	3.22	3.3	
	SVMPoly	SVMLap	kNN	SVMGaus	MLP	10-FCV
Average ranks	2.37	2.55	2.78	3.02	4.28	
AMSAN vs ALL	kNN	C4.5	SVMLap	RBF-DDA	SLP	Train-test
Average ranks	2.48	2.72	3.25	3.33	3.22	
	kNN	C4.5	SVMLap	SLP	RBF-DDA	10-FCV
Average ranks	1.40	2.23	3.22	3.97	4.18	
MF vs ALL	Naive Bayes	LDA	C4.5	JRip	SVMGaus	Train-test
Average ranks	1.63	3.28	3.63	3.45	3.00	
	Naive Bayes	JRip	LDA	SVMGaus	SVMLin	10-FCV
Average ranks	1.20	2.92	3.78	3.22	3.88	

In Table 4.33, the results of the post-hoc test with Holm's correction of the top five classifiers in AIDP vs ALL classification using train-test and 10-FCV are shown. In train-test, a statistically significant difference between kNN and RBF-DDA was found. In 10-FCV, a statistically significant difference between MLP and LDA was found, as well as between MLP and kNN.

Table 4.33: Post-hoc test with Holm's correction of the top five classifiers in AIDP vs ALL classification using train-test and 10-FCV.

Classes	Classifiers compared		p value	alpha*	Ho	Scenario
AIDP vs ALL	kNN vs	RBF-DDA	0.000	0.013	rejected	Train-test
	kNN vs	MLP	0.020	0.017	accepted	Train-test
	kNN vs	C4.5	0.050	0.025	accepted	Train-test
	kNN vs	SVMLap	0.142	0.050	accepted	Train-test
AIDP vs ALL	MLP vs	LDA	0.000	0.013	rejected	10-FCV
	MLP vs	kNN	0.001	0.017	rejected	10-FCV
	MLP vs	C4.5	0.066	0.025	accepted	10-FCV
	MLP vs	SVMLap	0.596	0.050	accepted	10-FCV

In Table 4.34, the results of the post-hoc test with Holm's correction of the top five classifiers in AMAN vs ALL classification using 10-FCV are shown. A statistically significant difference between SVMPoly and MLP was found.

Table 4.34: Post-hoc test with Holm's correction of the top five classifiers in AMAN vs ALL classification using 10-FCV.

Classes	Classifiers compared		p value	alpha*	Ho	scheme
AMAN vs ALL	SVMPoly vs	MLP	0.000	0.013	rejected	10-FCV
	SVMPoly vs	SVMGaus	0.111	0.017	accepted	10-FCV
	SVMPoly vs	kNN	0.307	0.025	accepted	10-FCV
	SVMPoly vs	SVMLap	0.653	0.050	accepted	10-FCV

In Table 4.35, the results of the post-hoc test with Holm's correction of the top five classifiers in AMSAN vs ALL classification using 10-FCV are shown. A statistically significant difference between kNN and the rest of classifiers was found.

Table 4.35: Post-hoc test with Holm's correction of the top five classifiers in AMSAN vs ALL classification using 10-FCV.

Classes	Classifiers compared		p value	alpha*	Ho	scheme
AMSAN vs ALL	kNN vs	RBF-DDA	0.000	0.013	rejected	10-FCV
	kNN vs	SLP	0.000	0.017	rejected	10-FCV
	kNN vs	SMLap	0.000	0.025	rejected	10-FCV
	kNN vs	C4.5	0.041	0.050	rejected	10-FCV

In Table 4.36, the results of the post-hoc test with Holm's correction of the top five classifiers in MF vs ALL classification using train-test and 10-FCV are shown. Both in train-test and 10-FCV, a statistically significant difference between Naive Bayes and the rest of classifiers was found.

Table 4.36: Post-hoc test with Holm's correction of the top five classifiers in MF vs ALL classification using train-test and 10-FCV.

Classes	Classifiers compared		p value	alpha*	Ho	scheme
MF vs ALL	Naive Bayes vs	C4.5	0.000	0.013	rejected	Train-test
	Naive Bayes vs	JRip	0.000	0.017	rejected	Train-test
	Naive Bayes vs	LDA	0.000	0.025	rejected	Train-test
	Naive Bayes vs	SVMGaus	0.001	0.050	rejected	Train-test
MF vs ALL	Naive Bayes vs	SVMLin	0.000	0.013	rejected	10-FCV
	Naive Bayes vs	LDA	0.000	0.017	rejected	10-FCV
	Naive Bayes vs	SVMGaus	0.000	0.025	rejected	10-FCV
	Naive Bayes vs	JRip	0.000	0.050	rejected	10-FCV

4.4.3.4.3 OVO classification

In Table 4.37 we show the Friedman test results of the comparison among the top five classifiers in balanced accuracy across 30 runs, using train-test and 10-FCV in OVO classification. The complete list of the top five classifiers for each case is shown in Table 4.38. In train-test scenario, a statistically significant difference among the top five classifiers in balanced accuracy across 30 runs was

found only in AMAN vs AMSAN classification. Theses cases were AIDP vs ALL and MF vs ALL. In 10-FCV, a statistically significant difference among the top five classifiers in balanced accuracy across 30 runs was found in all OVA classifications with the exception of AMAN vs MF classification.

In all cases, we used as our null hypothesis Ho: There is no statistically significant difference in the balanced accuracy among the top five classifiers across 30 runs, and our alternative hypothesis H_1: There is a statistically significant difference in the balanced accuracy among the top five classifiers across 30 runs.

Table 4.37: Friedman test results of the comparison among top five classifiers in balanced accuracy across 30 runs in OVO classification using train-test and 10-FCV.

Classes	Friedman Statistic	Critical value	Ho	Scenario
AIDP vs AMAN	1.512	2.45	accepted	Train-test
AIDP vs AMSAN	1.074	2.45	accepted	Train-test
AIDP vs MF	0.820	2.45	accepted	Train-test
AMAN vs AMSAN	5.162	2.45	rejected	Train-test
AMAN vs MF	0.508	2.45	accepted	Train-test
AMSAN vs MF	0.432	2.45	accepted	Train-test
AIDP vs AMAN	3.940	2.45	rejected	10-FCV
AIDP vs AMSAN	32.246	2.45	rejected	10-FCV
AIDP vs MF	3.340	2.45	rejected	10-FCV
AMAN vs AMSAN	10.887	2.45	rejected	10-FCV
AMAN vs MF	2.046	2.45	accepted	10-FCV
AMSAN vs MF	12.727	2.45	rejected	10-FCV

In table 4.38 we show the average ranks for the top five classifiers using train-test and 10-FCV in OVO classification. In both train-test and 10-FCV, we highlight the ranked first classifiers only in cases where a statistically significant difference was found. On average, in train-test scenario, the ranked first classifier was kNN (2.08) for AMAN vs AMSAN classification. In 10-FCV, the ranked first classifiers were JRip (2.18) for AIDP vs AMAN, SVMPoly (1.78) for AIDP vs AMAN, OneR (2.25) for AIDP vs AMSAN, kNN (2.12) for AMAN vs AMSAN, and Naive Bayes (2.07) for AMSAN vs MF.

In Table 4.39, the results of the post-hoc test with Holm's correction of the top five classifiers in AIDP vs AMAN classification using 10-FCV are shown. A statistically significant difference between JRip and OneR, as well as between JRip and C4.5 was found.

In Table 4.40, the results of the post-hoc test with Holm's correction of the top five classifiers in AIDP vs AMSAN classification using 10-FCV are shown. A statistically significant difference between SVMPoly and three out of four classifiers was found, these were: OneR, kNN and SVMLap.

In Table 4.41, the results of the post-hoc test with Holm's correction of the top five classifiers in AIDP vs MF classification using 10-FCV are shown. A statistically significant difference between OneR and SVMGaus was found.

Table 4.38: Average ranks of the top five classifiers using train-test and 10-FCV in OVO classification.

Classes	Classifiers compared					Scenario
AIDP vs AMAN	JRip	OneR	C4.5	SVMLin	SLP	Train-test
Average ranks	2.53	2.87	2.90	3.40	3.30	
	JRip	SVMLin	MLP	C4.5	OneR	10-FCV
Average ranks	2.18	3.07	2.82	3.30	3.63	
AIDP vs AMSAN	SVMLap	SVMPoly	MLP	SVMGaus	SVMLin	Train-test
Average ranks	2.70	2.77	3.43	3.15	2.95	
	SVMPoly	SVMGaus	SVMLap	kNN	OneR	10-FCV
Average ranks	1.78	1.88	3.20	3.73	4.40	
AIDP vs MF	OneR	JRip	kNN	MLP	Naive Bayes	Train-test
Average ranks	2.67	2.80	3.32	3.12	3.10	
	OneR	SVMPoly	SVMLap	JRip	SVMGaus	10-FCV
Average ranks	2.25	2.83	3.12	3.17	3.63	
AMAN vs AMSAN	kNN	SVMLap	MLP	RBF-DDA	SVMGaus	Train-test
Average ranks	2.08	2.77	3.15	3.28	3.72	
	kNN	SVMLap	RBF-DDA	C4.5	SVMPoly	10-FCV
Average ranks	2.12	2.47	2.70	3.63	4.08	
AMAN vs MF	SVMGaus	SVMLap	RBF-DDA	Naive Bayes	SVMLin	Train-test
Average ranks	2.82	2.82	2.98	3.32	3.07	
	kNN	SVMGaus	SVMLap	SVMPoly	SVMLin	10-FCV
Average ranks	2.78	2.78	2.78	2.93	3.72	
AMSAN vs MF	Naive Bayes	LDA	MLP	SLP	SVMGaus	Train-test
Average ranks	2.77	2.90	3.10	3.27	2.97	
	Naive Bayes	LDA	MLP	BLR	SVMGaus	10-FCV
Average ranks	2.07	2.05	3.42	3.62	3.85	

Table 4.39: Post-hoc test with Holm's correction of the top five classifiers in AIDP vs AMAN classification using 10-FCV.

Classes	Classifiers compared		p value	alpha*	Ho	Scenario
AIDP vs AMAN	JRip vs	OneR	0.000	0.013	rejected	10-FCV
	JRip vs	C4.5	0.006	0.017	rejected	10-FCV
	JRip vs	SVMLin	0.030	0.025	accepted	10-FCV
	JRip vs	MLP	0.121	0.050	accepted	10-FCV

Table 4.40: Post-hoc test with Holm's correction of the top five classifiers in AIDP vs AMSAN classification using 10-FCV.

Classes	Classifiers compared		p value	alpha*	Ho	Scenario
AIDP vs AMSAN	SVMPoly vs	OneR	0.000	0.013	rejected	10-FCV
	SVMPoly vs	kNN	0.000	0.017	rejected	10-FCV
	SVMPoly vs	SVMLap	0.001	0.025	rejected	10-FCV
	SVMPoly vs	SVMGaus	0.806	0.050	accepted	10-FCV

Table 4.41: Post-hoc test with Holm's correction of the top five classifiers in AIDP vs MF classification using 10-FCV.

Classes	Classifiers compared		p value	alpha*	Ho	Scenario
AIDP vs MF	OneR vs	SVMGaus	0.001	0.013	rejected	10-FCV
	OneR vs	JRip	0.025	0.017	accepted	10-FCV
	OneR vs	SVMLap	0.034	0.025	accepted	10-FCV
	OneR vs	SVMPoly	0.153	0.050	accepted	10-FCV

In Table 4.42, the results of the post-hoc test with Holm's correction of the top five classifiers in AMAN vs AMSAN classification using train-test and 10-FCV are shown. In train-test, a statistically significant difference between kNN and three other classifiers was found: SVMGaus, RBF-DDA and MLP. In 10-FCV, a statistically significant difference between kNN and two other classifiers was found: SVMPoly and C4.5.

Table 4.42: Post-hoc test with Holm's correction of the top five classifiers in AMAN vs AMSAN classification using train-test and 10-FCV.

Classes	Classifiers compared		p value	alpha*	Ho	Scenario
AMAN vs AMSAN	kNN vs	SVMGaus	0.000	0.013	rejected	Train-test
	kNN vs	RBF-DDA	0.003	0.017	rejected	Train-test
	kNN vs	MLP	0.009	0.025	rejected	Train-test
	kNN vs	SVMLap	0.094	0.050	accepted	Train-test
AMAN vs AMSAN	kNN vs	SVMPoly	0.000	0.013	rejected	10-FCV
	kNN vs	C4.5	0.000	0.017	rejected	10-FCV
	kNN vs	RBF-DDA	0.153	0.025	accepted	10-FCV
	kNN vs	SVMLap	0.391	0.050	accepted	10-FCV

In Table 4.43, the results of the post-hoc test with Holm's correction of the top five classifiers in AMSAN vs MF classification using 10-FCV are shown. In 10-FCV, a statistically significant difference between Naive Bayes and three other classifiers was found: SVMGaus, BLR and MLP.

Table 4.43: Post-hoc test with Holm's correction of the top five classifiers in AMSAN vs MF classification using 10-FCV.

Classes	Classifiers compared		p value	alpha*	Ho	Scenario
AMSAN vs MF	Naive Bayes vs	SVMGaus	0.000	0.013	rejected	10-FCV
	Naive Bayes vs	BLR	0.000	0.017	rejected	10-FCV
	Naive Bayes vs	MLP	0.001	0.025	rejected	10-FCV
	Naive Bayes vs	LDA	0.967	0.050	accepted	10-FCV

4.4.4 Discussion

Our objective in this work was to create the highest accurate predictive model for GBS possible, using the 16 relevant features identified with QSA-PAM method. This work constitutes the first effort on this topic using machine learning methods. For this first approach, we used single classifiers. We selected 14 single classifiers from diverse types: decision trees (C4.5), instance-based learners (kNN), kernel-based (SVM), neural networks (SLP, MLP, RBF-DDA), rule induction learners (OneR, JRip), among others. The complete list is in section 4.6.1. We compared their performance in three types of experiments: four GBS subtype classification, OVA classification and OVO classification.

4.4.4.1 Four GBS subtype classification

In both train-test and 10-FCV scenarios, the best classifiers were the same, although not in the same order: kNN, SVM with all different kernels and C4.5. This result confirms them as a good single classifiers.

The standard deviation of the average accuracy was a little lower in 10-FCV, this could a consequence of the cross-validation characteristic in the sense of reducing the variance by averaging over k different partitions.

The multiclass AUC in 10-FCV scenario was a little higher than in train-test. This requires a further investigation.

In both train-test and 10-FCV scenarios, OneR obtained the worst performance. One possible explanation of this situation is that, since OneR generates one single rule to make the classification, maybe that single rule is not enough to classify the four GBS subtypes in this particular problem.

Values of specificity and sensitivity were similar in train-test and in 10-FCV. Sensitivity a little lower than specificity. This result could be a consequence of the imbalanced data problem.

In the statistical analysis, opposite results were found. In train-test, a statistically significant difference in average accuracy among the top five classifiers across 30 runs was found. On average, kNN was ranked first. On the contrary, in 10-FCV, no statistical significant difference in average accuracy among the top five classifiers across 30 runs was found. One possible explanation for this last result could be the stability in average accuracy achieved by the classifiers in 10-FCV.

4.4.4.2 OVA classification

Both in train-test and 10-FCV, AMAN vs ALL showed the highest performance in balanced accuracy across 30 runs. The opposite case was MF vs ALL. AMSAN vs ALL was the second best. In OVA classification, the two classes with the largest number of instances resulted in the best classification results, that is AMAN vs ALL and AMSAN vs ALL.

kNN, C4.5 and SVMLap appear in the top five classifiers in most cases both in train-test and in 10-FCV. Naive Bayes appears as the top classifier for MF vs ALL both in train-test and 10-FCV.

In most cases, OneR obtained the worst performance, both in train test and in 10-FCV.

Specificity and sensitivity show that in most cases, single classifiers were better at identifying negative instances, both in train-test and 10-FCV scenarios.

Overall, the highest and more stable average results across 30 runs were obtained in 10-FCV scenario.

After the statistical analysis, two classifiers stood out from the rest. Naive Bayes resulted as the best classifier for the minority class vs ALL, that is MF vs ALL, both in train-test and 10-FCV. kNN was the best classifier for AIDP vs ALL in train-test and for AMSAN vs ALL in 10-FCV.

4.4.4.3 OVO classification

Both in train-test and 10-FCV, all classifications with AMAN subtype present resulted with the highest performance in balanced accuracy and most stable standard deviation across 30 runs. AIDP vs MF showed the lowest balanced accuracy levels and highest standard deviation across 30 runs both in train-test and 10-FCV.

Although in OVO classification, the top five classifiers were more diverse than in four GBS subtype classification and in OVA classification, three classifiers stood out: SVM with at least one kernel, kNN and MLP. In addition, induction ruler learners, that is JRip and OneR, had a good performance in OVO classification in contrast with OVA classification.

Again, in most cases, specificity was a little higher than sensitivity, both in train-test and 10-FCV scenarios.

After the statistical analysis, kNN was the best classifier for AMAN vs AMSAN in both train-test and 10-FCV. In the rest of the cases, there were different best classifiers: JRip, SVMPoly, OneR and Naive Bayes for AIDP vs AMAN, AIDP vs AMSAN, AIDP vs MF and AMSAN vs MF respectively.

4.5 Approach 2: Ensemble methods

In this second approach, we selected five ensemble methods: boosting, bagging, C5.0, random forest and random subspace. In principle, ensemble learning combines multiple classifiers to obtain better predictive performance that could be obtained from any of the constituent classifiers. We applied these five methods in four GBS subtype classification, as well as in OVO and OVA classification and compare their performance.

4.5.1 Experimental design

We used the 16-feature subset, described in section 5.1, for experiments. We added the class variable to this subset, that is, the GBS subtype. Finally, we created a dataset containing the 129 instances and 17 features. As mentioned in section 4.1, our dataset has 4 classes, identified with numbers 1 to 4, where 1 is AIDP, 2 is AMAN, 3 is AMSAN, and 4 is MF.

We used train-test evaluation scheme in all cases. We performed 30 runs where we applied each of the methods described in section 4.7.1. In each run, we set a different seed. The same seeds were used for each classifier. These seeds were generated using Mersenne-Twister pseudo-random number generator [64]. The use of a different seed for each run ensures producing different splits of train and test sets.

The base classifier in random subspace method used was the best single classifier for each case using train-test, the complete list is in table 4.44. Experiments of random subspace were performed in Weka 3.6.12. SVMLap is not implemented in Weka 3.6.12, therefore we used SVMPoly (second best) instead in AIDP vs AMSAN classification.

4.5.1.1 Four GBS subtype classification

In this classification, the four GBS subtypes were included in the dataset, that is AIDP, AMAN, AMSAN and MF. In this scenario, the base metric was the average accuracy.

4.5.1.2 OVA classification

For OVA classification we created four new datasets, as the number of GBS subtypes in the dataset. In each one, the instances of one class were marked as the positive cases and the instances of the remaining classes were marked as the negative cases. In this scenario, the base metric was the balanced accuracy.

4.5.1.3 OVO classification

For OVO classification we created six new datasets, as many combinations of pairs of GBS subtypes. Each dataset contained instances of only two GBS subtypes, one class marked as the positive case and the other class as the negative case. In this scenario, the base metric was the balanced accuracy.

4.5.1.4 Train-test

For each run, we computed accuracy, sensitivity, specificity, Kappa statistic and multiclass AUC. Finally, we averaged each of these quantities across the 30 runs.

4.5.2 Parameter optimization/setting

4.5.2.1 Boosting

The number of boosting iterations for all cases was 50.

4.5.2.2 Bagging

The number of trees used for all cases was 100.

4.5.2.3 C5.0

C5.0 requires the optimization of the number of trials. The tuning of this parameter was performed by 30 training-test runs using different numbers of trials from 5 to 100.

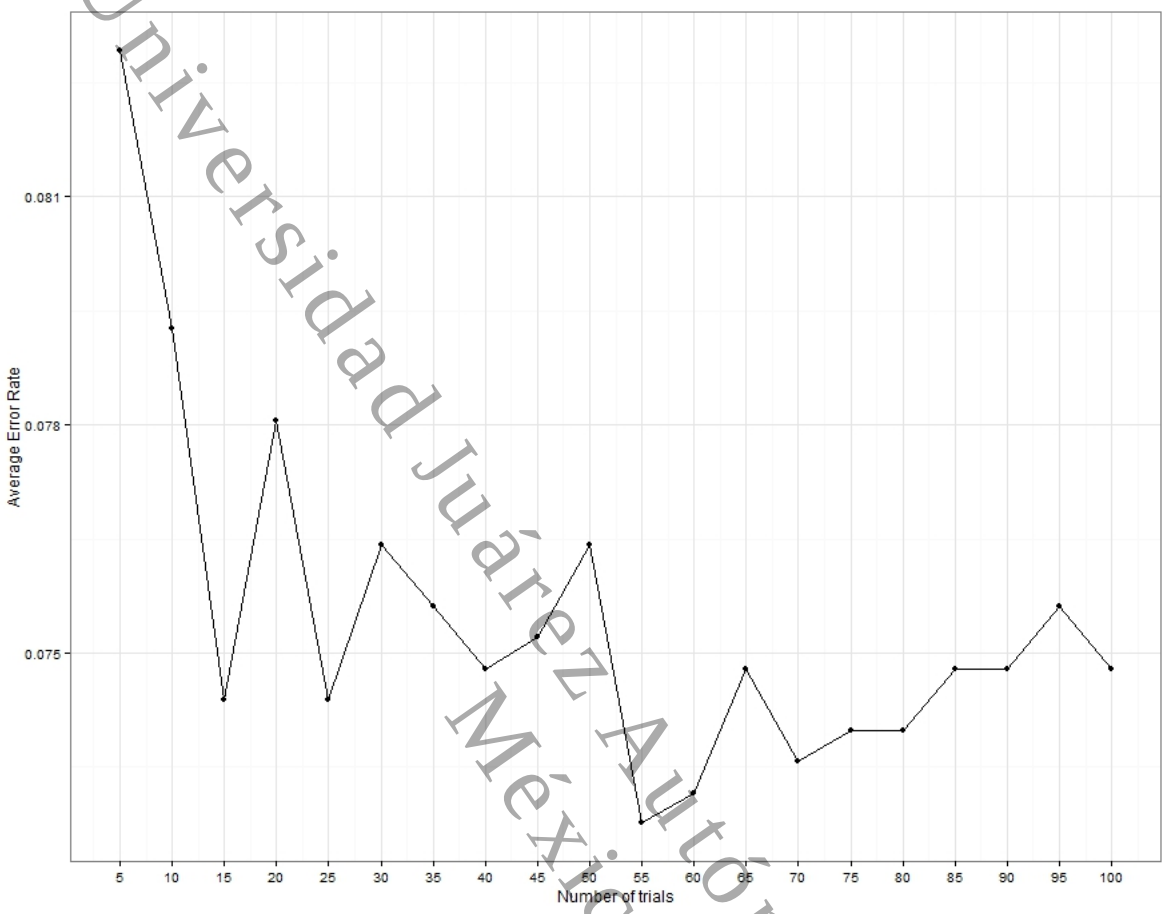

Figure 4.12: Number of trials optimization in C5.0.

Figure 4.12 shows the results of C5.0 optimization. The lowest average error rate across 30 train-test runs was obtained with number of trials = 55. Experiments for all cases in C5.0, including OVO and OVA classification, were performed using this number of trials.

4.5.2.4 Random forest

Random forest has only two tuning parameters: the number of variables in the random subset at each node and the number of trees in the forest. In this work, we let random forest to automatically tune the first parameter. In order to tune the second one, we performed 30 training-test runs using different numbers of trees from 100 to 1000.

Figure 4.13 shows the results of random forest optimization. The lowest average error rate across 30 train-test runs was obtained with number of trees = 700. Experiments for all cases in ramdom forest, including OVO and OVA classification, were performed using this number of trees.

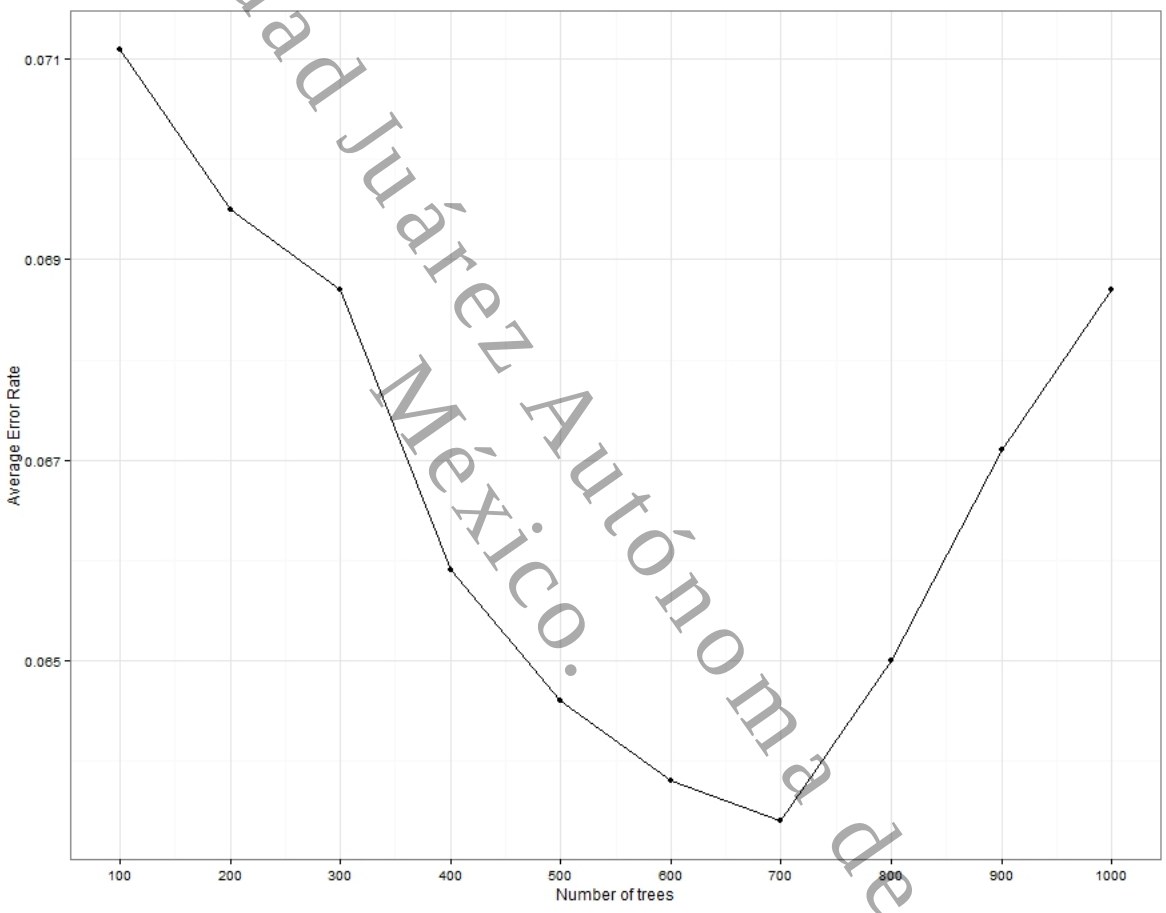

Figure 4.13: Number of trees optimization in random forest.

4.5.2.5 Random subspace

In this work, the subspace sample size was set to 0.5. The number of iterations for random subspace was set to 50. The same optimal parameter setting obtained in first approach classification for the base classifiers were used in random subspace. Table 4.44 shows the complete list of base classifiers configuration.

Table 4.44: Base classifiers used in random subspace for each classification case.

Base classifier	Parameter setting	Classes
*k*NN	k=18 d=1	4 classes
*k*NN	k=18 d=1	AIDP vs ALL
*k*NN	k=18 d=1	AMAN vs ALL
*k*NN	k=18 d=1	AMSAN vs ALL
Naive Bayes		MF vs ALL
JRip	NumOpt = 3	AIDP vs AMAN
SVMGaus	s=0.01 C=10	AIDP vs AMSAN
OneR		AIDP vs MF
*k*NN	k=18 d=1	AMAN vs AMSAN
SVMGaus	s=0.01 C=10	AMAN vs MF
Naive Bayes		AMSAN vs MF

4.5.3 Results

4.5.3.1 Four GBS subtype classification

In this section, we show the results of ensemble methods in four GBS subtype classification. Table 4.45 shows the average results across 30 runs along with the standard deviation (sd). Four of the five ensemble methods obtained an average accuracy above 0.90. Random forest outperformed the rest of the methods in most of the metrics. The worst performance was obtained by bagging, with an average accuracy of 0.89 and poor results in sensitivity and in Kappa statistic.

Multiclass AUC ranged 0.78 - 0.83. Specificity values were higher than those of sensitivity. Specificity ranged 0.92 - 0.95, while sensitivity ranged 0.66 - 0.81. Kappa ranged 0.69 - 0.80. Overall, four GBS subtype classification using ensemble methods obtained high values in average accuracy. The remaining metrics showed a large variation.

Figure 4.14 shows the average accuracy across 30 runs for each ensemble method in four GBS subtype classification. Also the average error rate for each method is shown. Most of the methods obtained an average accuracy above 0.90. Random forest obtained the lowest average error rate across 30 train-test runs.

Table 4.46 shows the average accuracy of single classifiers and ensemble methods across 30 runs of four GBS subtype classification. Only two ensemble methods, random forest and C5.0, outperformed all single classifiers in average accuracy. Boosting resulted better than 13 of 14 single classifiers. Random subspace had a performance comparable to that of SVMLin and Naive Bayes.

Table 4.45: Average results of ensemble methods across 30 runs in four GBS subtype classification.

Ensemble method	Average Accuracy	Multiclass AUC	Sensitivity	Specificity	Kappa
Random forest	**0.9366**	**0.8390**	**0.8120**	**0.9544**	**0.8090**
	0.0245	0.0803	0.0812	0.0178	0.0748
C5.0	**0.9272**	**0.8398**	**0.8126**	**0.9476**	**0.7825**
	0.0251	0.0789	0.0749	0.0191	0.0746
Boosting	**0.9195**	**0.8099**	**0.7906**	**0.9422**	**0.7596**
	0.0202	0.0578	0.0648	0.0158	0.0610
Random subspace	**0.9016**	**0.7871**	**0.6607**	**0.9251**	**0.6960**
	0.0216	0.0592	0.0691	0.0169	0.0682
Bagging	**0.8980**	**0.7895**	**0.6936**	**0.9251**	**0.6923**
	0.0284	0.0484	0.0622	0.0206	0.0831

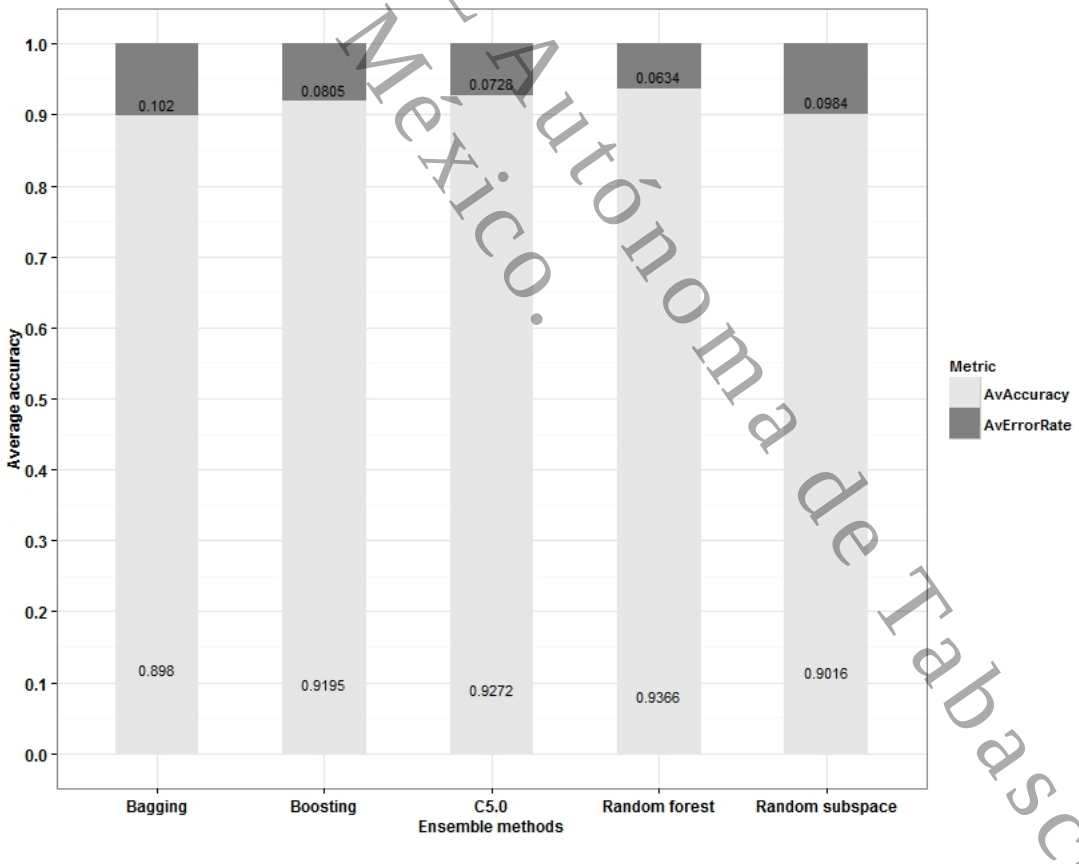

Figure 4.14: Average accuracy and average error rate of ensemble methods across 30 runs in four GBS subtype classification.

However, random subspace failed at improving kNN as single classifier. As shown in Table 4.46, kNN when used as single classifier obtained a higher average accuracy (0.9268) that when used in random subspace as a base classifier (0.9016). The worst performance of ensemble methods was obtained by Bagging, however it was better than half of the single classifiers.

Table 4.46: Average accuracy of single classifiers and ensemble methods across 30 runs in four GBS subtype classification. Ensemble methods are shown in bold.

Method	Average Accuracy
Random forest	**0.9366**
C5.0	**0.9272**
kNN	0.9268
Boosting	**0.9195**
SVMLap	0.9163
SVMPoly	0.9142
SVMGaus	0.9126
C4.5	0.9114
SVMLin	0.9069
Random subspace	**0.9016**
Naive Bayes	0.9008
Bagging	**0.8980**
MLP	0.8915
RBF-DDA	0.8882
SLP	0.8850
JRip	0.8837
LDA	0.8825
MLR	0.8793
OneR	0.7972

4.5.3.2 OVA classification

In this section, we show the results of ensemble methods across 30 runs in OVA classification, that is AIDP vs ALL, AMAN vs ALL, and so on. Table 4.47 shows the average results of ensemble methods across 30 runs along with the standard deviation (sd) in AIDP vs ALL classification. Only two of the five ensemble methods obtained an balanced accuracy above 0.80, these were C5.0 and boosting. The worst performance was obtained by random subspace, with an balanced accuracy of 0.68 and poor results in most metrics. However, this same method obtained an unusual high specificity value.

AUC ranged 0.68 - 0.81. Specificity was higher than sensitivity. Specificity ranged 0.92 - 0.99, while sensitivity ranged 0.36 - 0.69. Kappa ranged 0.46 - 0.64. In summary, ensemble methods obtained a low performance in most metrics in AIDP vs ALL classification.

Table 4.48 shows the balanced accuracy of single classifiers and ensemble methods across 30 runs in AIDP vs ALL classification. No ensemble method was able to improve kNN, a single classifier, in

Table 4.47: Average results of ensemble methods across 30 runs in AIDP vs ALL classification.

Method	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
C5.0	**0.8171**	**0.9048**	**0.6944**	**0.9398**	**0.6266**	**0.8171**
	0.0976	0.0556	0.1861	0.0565	0.1806	0.0976
Boosting	**0.8069**	**0.8992**	**0.6907**	**0.9231**	**0.6046**	**0.8069**
	0.0827	0.0479	0.1750	0.0732	0.1670	0.0827
Random forest	**0.7769**	**0.9270**	**0.5667**	**0.9870**	**0.6407**	**0.7769**
	0.0888	0.0306	0.1836	0.0289	0.1645	0.0888
Bagging	**0.7681**	**0.8960**	**0.6093**	**0.9269**	**0.5629**	**0.7681**
	0.0818	0.0466	0.1829	0.1091	0.1682	0.0818
Random subspace	**0.6815**	**0.9056**	**0.3666**	**0.9953**	**0.4661**	**0.6810**
	0.1240	0.0351	0.2453	0.0129	0.2408	0.1218

balanced accuracy. Only two ensemble methods, C5.0 and boosting, outperformed the rest of single classifiers in balanced accuracy. Random subspace had a poor performance, only being better than the worst single classifier, OneR. Again, random subspace was not able of improving kNN as single classifier.

Table 4.49 shows the average results of ensemble methods across 30 runs in AMAN vs ALL classification. The standard deviation is also shown. Four of the five ensemble methods obtained a balanced accuracy above 0.90, only bagging was under this value. AUC ranged 0.85 - 0.92. Values obtained in specificity were higher than those obtained in sensitivity. Specificity ranged 0.92 - 0.94, while sensitivity ranged 0.78 - 0.91. Kappa ranged 0.73 - 0.83. In short, ensemble methods obtained values on or above 0.85 in most metrics in AMAN vs ALL classification.

Table 4.50 shows the balanced accuracy of single classifiers and ensemble methods across 30 runs in AMAN vs ALL classification. Two single classifiers outperformed all the ensemble methods, kNN and SVMGaus. Boosting resulted better than 12 single classifiers and 4 ensemble methods. Like in the former cases, random subspace was not able of improving kNN as single classifier. Bagging was the worst ensemble method in AMAN vs ALL classification.

Table 4.51 shows the average results of ensemble methods across 30 runs in AMSAN vs ALL classification. The standard deviation is also shown. All five ensemble methods obtained a balanced accuracy above 0.85. AUC ranged 0.85 - 0.89. Like all other cases, specificity was higher than sensitivity. Specificity ranged 0.87 - 0.93, while sensitivity ranged 0.83 - 0.86. Kappa ranged 0.71 - 0.78. Overall, ensemble methods had a high performance in AMSAN vs ALL classification in most metrics.

Table 4.52 shows the balanced accuracy of single classifiers and ensemble methods across 30 runs in AMSAN vs ALL classification. Random forest was the best ensemble method with a balanced

Table 4.48: Balanced accuracy of single classifiers and ensemble methods across 30 runs in AIDP vs ALL classification. Ensemble methods are shown in bold.

Method	Balanced Accuracy
kNN	0.8227
C5.0	**0.8171**
Boosting	**0.8069**
C4.5	0.8042
MLP	0.8014
SVMLap	0.8009
Random forest	**0.7769**
Bagging	**0.7681**
RBF-DDA	0.7648
SLP	0.7551
SVMLin	0.7542
LDA	0.7537
SVMGaus	0.7491
JRip	0.7407
SVMPoly	0.7389
Naive Bayes	0.7231
BLR	0.7181
Random subspace	**0.6815**
OneR	0.6222

Table 4.49: Average results of ensemble methods across 30 runs in AMAN vs ALL classification.

Method	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
Boosting	**0.9214**	**0.9341**	**0.8989**	**0.9439**	**0.8384**	**0.9214**
	0.0599	0.0441	0.1045	0.0591	0.1098	0.0599
Random subspace	**0.9138**	**0.9143**	**0.9112**	**0.9155**	**0.7992**	**0.9133**
	0.0449	0.0418	0.0788	0.0545	0.0926	0.0442
C5.0	**0.9136**	**0.9302**	**0.8750**	**0.9522**	**0.8290**	**0.9136**
	0.0487	0.0385	0.0948	0.0461	0.0911	0.0487
Random forest	**0.9033**	**0.9214**	**0.8611**	**0.9456**	**0.8069**	**0.9033**
	0.0550	0.0406	0.1057	0.0424	0.1006	0.0550
Bagging	**0.8542**	**0.8952**	**0.7872**	**0.9211**	**0.7303**	**0.8542**
	0.0727	0.0462	0.1624	0.0929	0.1313	0.0727

Table 4.50: Balanced accuracy of single classifiers and ensemble methods across 30 runs in AMAN vs ALL classification. Ensemble methods are shown in bold.

Method	Balanced Accuracy
kNN	0.9356
SVMGaus	0.9256
Boosting	**0.9214**
SVMPoly	0.9203
SVMLap	0.9181
Random subspace	**0.9138**
C5.0	**0.9136**
C4.5	0.9067
Random forest	**0.9033**
MLP	0.8978
SVMLin	0.8969
SLP	0.8778
Naive Bayes	0.8700
RBF-DDA	0.8689
LDA	0.8594
Bagging	**0.8542**
JRip	0.8472
BLR	0.8214
OneR	0.6336

Table 4.51: Average results of ensemble methods across 30 runs in AMSAN vs ALL classification.

Method	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
Random forest	**0.8924**	**0.8960**	**0.8544**	**0.9304**	**0.7889**	**0.8924**
	0.0429	0.0417	0.0814	0.0621	0.0849	0.0429
Boosting	**0.8786**	**0.8810**	**0.8671**	**0.8902**	**0.7591**	**0.8786**
	0.0386	0.0375	0.0663	0.0796	0.0762	0.0386
C5.0	**0.8782**	**0.8810**	**0.8491**	**0.9072**	**0.7588**	**0.8782**
	0.0494	0.0480	0.0779	0.0532	0.0978	0.0494
Random subspace	**0.8688**	**0.8722**	**0.8617**	**0.8798**	**0.7425**	**0.8708**
	0.0410	0.0440	0.0550	0.0664	0.0878	0.0416
Bagging	**0.8555**	**0.8587**	**0.8365**	**0.8744**	**0.7137**	**0.8555**
	0.0526	0.0515	0.0960	0.1012	0.1047	0.0526

accuracy of 0.8924. However, it was not able of outperforming kNN, the best single classifier with 0.8953 of balanced accuracy. C4.5, a single classifier was the third best method, and it had a highest balanced accuracy than 4 ensemble methods and 12 single classifiers. Neither in this case, random subspace was able of improving kNN as single classifier.

Table 4.52: Balanced accuracy of single classifiers and ensemble methods across 30 runs in AMSAN vs ALL classification. Ensemble methods are shown in bold.

Method	Balanced accuracy
kNN	0.8953
Random forest	**0.8924**
C4.5	0.8852
Boosting	**0.8786**
C5.0	**0.8782**
SVMLap	0.8756
RBF-DDA	0.8695
Random subspace	**0.8688**
SLP	0.8644
Bagging	**0.8555**
SVMPoly	0.8504
SVMGaus	0.8492
MLP	0.8490
JRip	0.8449
OneR	0.8219
Naive Bayes	0.8166
SVMLin	0.8101
LDA	0.8072
BLR	0.7971

Table 4.53 shows the average results of ensemble methods across 30 runs in MF vs ALL classification. The standard deviation is also shown. Three ensemble methods obtained a balanced accuracy above 0.80. AUC ranged 0.74 - 0.85. Sensitivity was much lower than specificity than in previous cases. Sensitivity ranged 0.52 - 0.76. Specificity ranged 0.90 - 0.96. Kappa ranged 0.49 - 0.64. In summary, ensemble methods had a poor performance in MF vs ALL classification in most metrics.

Table 4.54 shows the balanced accuracy of single classifiers and ensemble methods across 30 runs in MF vs ALL classification. Naive Bayes, a single classifier, obtained the highest balanced accuracy outperforming all ensemble methods. Random subspace was the best ensemble method. However, it was not able of improving Naive Bayes as single classifier. Three ensemble methods were better than most of single classifiers, these were random subspace, bagging and C5.0.

4.5.3.3 OVO classification

Table 4.55 shows the average results of ensemble methods across 30 runs in AIDP vs AMAN classification. The standard deviation is also shown. All ensemble methods obtained a balanced accuracy

Table 4.53: Average results of ensemble methods across 30 train-test runs in MF vs ALL classification.

Method	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
Random subspace	**0.8471**	**0.9333**	**0.7500**	**0.9525**	**0.6413**	**0.8513**
	0.1067	0.0296	0.2274	0.0349	0.1509	0.1074
Bagging	**0.8342**	**0.9024**	**0.7675**	**0.9009**	**0.5498**	**0.8342**
	0.1298	0.0513	0.2460	0.1297	0.2213	0.1298
C5.0	**0.8048**	**0.9167**	**0.6667**	**0.9430**	**0.5601**	**0.8048**
	0.1329	0.0487	0.2653	0.0484	0.2145	0.1329
Boosting	**0.7572**	**0.9183**	**0.5697**	**0.9447**	**0.5022**	**0.7572**
	0.1346	0.0323	0.2802	0.0881	0.2085	0.1346
Random forest	**0.7463**	**0.9254**	**0.5250**	**0.9675**	**0.4973**	**0.7463**
	0.1519	0.0311	0.3104	0.0246	0.2599	0.1519

Table 4.54: Balanced accuracy of single classifiers and ensemble methods across 30 train-test runs in MF vs ALL classification. Ensemble methods are shown in bold.

Method	Balanced Accuracy
Naive Bayes	0.8939
Random subspace	**0.8471**
Bagging	**0.8342**
C5.0	**0.8048**
LDA	0.7818
C4.5	0.7691
Boosting	**0.7572**
JRip	0.7678
SVMGaus	0.7581
Random forest	**0.7463**
BLR	0.7316
SVMLin	0.7221
MLP	0.7193
SVMPoly	0.7086
SLP	0.6875
OneR	0.6853
SVMLap	0.6664
*k*NN	0.6539
RBF-DDA	0.5042

above 0.90. Also, AUC surpassed this value. Sensitivity values were lower than those of specificity. Sensitivity ranged 0.84 - 0.95. Specificity ranged 0.96 - 0.97. Kappa ranged 0.83 - 0.91. Overall, ensemble methods obtained values on or above 0.90 in most metrics in AIDP vs AMAN classification.

Table 4.55: Average results of ensemble methods across 30 runs in AIDP vs AMAN classification.

Method	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
Bagging	**0.9597**	**0.9630**	**0.9472**	**0.9722**	**0.9171**	**0.9597**
	0.0458	0.0395	0.0861	0.0551	0.0882	0.0458
C5.0	**0.9597**	**0.9611**	**0.9556**	**0.9639**	**0.9139**	**0.9597**
	0.0530	0.0487	0.0868	0.0566	0.1078	0.0530
Boosting	**0.9486**	**0.9556**	**0.9278**	**0.9694**	**0.8997**	**0.9486**
	0.0473	0.0397	0.0895	0.0599	0.0889	0.0473
Random subspace	**0.9325**	**0.9444**	**0.8833**	**0.9751**	**0.8683**	**0.9292**
	0.0887	0.0652	0.1703	0.0445	0.1589	0.0890
Random forest	**0.9097**	**0.9315**	**0.8444**	**0.9750**	**0.8368**	**0.9097**
	0.0805	0.0540	0.1747	0.0497	0.1348	0.0805

Table 4.56 shows the balanced accuracy of single classifiers and ensemble methods across 30 runs in AIDP vs AMAN classification. JRip slightly outperformed Bagging and C5.0, as the best classifier in this case. Two rule induction learners, JRip and OneR, were at the top 4 classifiers in AIDP vs AMAN classification. The performance of JRip as a single classifier was not improved by random subspace when used as base classifier.

Table 4.57 shows the average results of ensemble methods across 30 runs in AIDP vs AMSAN classification. The standard deviation is also shown. Random forest obtained a balanced accuracy above 0.90, the rest of ensemble methods above 0.85. Values obtained in sensitivity were lower than those obtained in specificity. Sensitivity ranged 0.78 - 0.85. Specificity ranged 0.90 - 0.96. Kappa ranged 0.70 - 0.83. In short, ensemble methods had a high performance in AIDP vs AMSAN classification in most metrics.

Table 4.58 shows the balanced accuracy of single classifiers and ensemble methods across 30 runs in AIDP vs AMSAN classification. Random forest obtained the highest balanced accuracy. The second best ensemble method was boosting, only under SVMLap and SVMPoly. In this case, SVMGaus was implemented as base classifier in random subspace instead of SVMLap, as mentioned in section 4.7.3. As in previous cases, random subspace did not improve the performance of its base classifier.

Table 4.59 shows the average results of ensemble methods across 30 runs in AIDP vs MF classification. The standard deviation is also shown. Random subspace and bagging obtained a balanced accuracy above 0.85, the rest of ensemble methods ranged 0.76 - 0.83. Sensitivity was lower than

Table 4.56: Balanced accuracy of single classifiers and ensemble methods across 30 runs in AIDP vs AMAN classification. Ensemble methods are shown in bold.

Method	Balanced Accuracy
JRip	0.9639
Bagging	**0.9597**
C5.0	**0.9597**
OneR	0.9583
Boosting	**0.9486**
C4.5	0.9458
SVMLin	0.9347
SLP	0.9333
MLP	0.9333
Random subspace	**0.9325**
Naive Bayes	0.9278
SVMLap	0.9125
Random forest	**0.9097**
SVMGaus	0.9014
RBF-DDA	0.8958
SVMPoly	0.8722
kNN	0.8639
BLR	0.8542
LDA	0.8389

Table 4.57: Average results of ensemble methods across 30 runs in AIDP vs AMSAN classification.

Method	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
Random forest	**0.9102**	**0.9387**	**0.8556**	**0.9649**	**0.8312**	**0.9102**
	0.0588	0.0430	0.1136	0.0485	0.1137	0.0588
Boosting	**0.8972**	**0.9160**	**0.8532**	**0.9412**	**0.7797**	**0.8972**
	0.0609	0.0539	0.1032	0.0623	0.1308	0.0609
Bagging	**0.8854**	**0.9067**	**0.8480**	**0.9228**	**0.7565**	**0.8854**
	0.0774	0.0675	0.1211	0.0798	0.1635	0.0774
C5.0	**0.8683**	**0.8893**	**0.8278**	**0.9088**	**0.7095**	**0.8683**
	0.0755	0.0563	0.1348	0.0569	0.1434	0.0755
Random subspace	**0.8674**	**0.9093**	**0.7834**	**0.9491**	**0.7459**	**0.8662**
	0.0755	0.0481	0.1524	0.0544	0.1324	0.0744

Table 4.58: Balanced accuracy of single classifiers and ensemble methods across 30 runs in AIDP vs AMSAN classification. Ensemble methods are shown in bold.

Method	Balanced Accuracy
Random forest	**0.9102**
SVMLap	0.9061
SVMPoly	0.8990
Boosting	**0.8972**
MLP	0.8949
SVMGaus	0.8931
SVMLin	0.8928
*k*NN	0.8899
Bagging	**0.8854**
SLP	0.8841
Naive Bayes	0.8728
C4.5	0.8692
C5.0	**0.8683**
Random subspace	**0.8674**
JRip	0.8588
BLR	0.8423
RBF-DDA	0.8339
OneR	0.8316
LDA	0.7836

specificity. Sensitivity ranged 0.71 - 0.83. Specificity ranged 0.82 - 0.99. Kappa ranged 0.53 - 0.75. In summary, only random subspace and bagging highly performed in AIDP vs MF classification in most metrics. The rest of ensemble methods had a low performance.

Table 4.60 shows the balanced accuracy of single classifiers and ensemble methods across 30 runs in AIDP vs MF classification. Two ensemble methods, random subspace and bagging, outperformed all the remaining methods, including single classifiers. In this case, random subspace was able to improve the performance of its base classifier, OneR.

Table 4.61 shows the average results of ensemble methods across 30 runs in AMAN vs AMSAN classification. The standard deviation is also shown. All ensemble methods obtained a balanced accuracy around 0.90 and above. As in previous cases, sensitivity was lower than specificity. Sensitivity ranged 0.88 - 0.93. Specificity ranged 0.90 - 0.97. Kappa ranged 0.78 - 0.89. Overall, ensemble methods obtained values on or above 0.85 in most metrics in AMAN vs AMSAN classification.

Table 4.62 shows the balanced accuracy of single classifiers and ensemble methods across 30 runs in AMAN vs AMSAN classification. *k*NN was the best classifier, above Random forest, second best. In this case, random subspace was not able to improve the performance of its base classifier, *k*NN.

Table 4.63 shows the average results of ensemble methods across 30 runs in AMAN vs MF classification. The standard deviation is also shown. Four of five ensemble methods obtained a balanced

Table 4.59: Average results of ensemble methods across 30 runs in AIDP vs MF classification.

Method	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
Random subspace	**0.8951**	**0.8733**	**0.7944**	**0.9917**	**0.7527**	**0.8930**
	0.0673	0.0785	0.1288	0.0456	0.1494	0.0671
Bagging	**0.8750**	**0.8533**	**0.8333**	**0.9167**	**0.7158**	**0.8806**
	0.1089	0.1167	0.1578	0.1681	0.2223	0.0867
Random forest	**0.8306**	**0.8267**	**0.8111**	**0.8500**	**0.6460**	**0.8306**
	0.0847	0.0785	0.1366	0.1925	0.1626	0.0847
C5.0	**0.8014**	**0.7967**	**0.7778**	**0.8250**	**0.5820**	**0.8014**
	0.1167	0.0999	0.1337	0.2555	0.2221	0.1167
Adaboost	**0.7694**	**0.7767**	**0.7167**	**0.8222**	**0.5333**	**0.7806**
	0.1275	0.1165	0.2037	0.2303	0.2337	0.0996

Table 4.60: Balanced accuracy of single classifiers and ensemble methods across 30 runs in AIDP vs MF classification. Ensemble methods are shown in bold.

Method	Balanced Accuracy
Random subspace	**0.8951**
Bagging	**0.8750**
OneR	0.8667
JRip	0.8528
*k*NN	0.8417
MLP	0.8347
Naive Bayes	0.8319
Random forest	**0.8306**
C4.5	0.8236
SVMGaus	0.8222
SVMPoly	0.8069
SVMLap	0.8028
C5.0	**0.8014**
SLP	0.7708
Boosting	**0.7694**
SVMLin	0.7542
RBF-DDA	0.7042
LDA	0.6250
BLR	0.6111

Table 4.61: Average results of ensemble methods across 30 runs in AMAN vs AMSAN classification.

Method	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
Random forest	**0.9474**	**0.9505**	**0.9333**	**0.9614**	**0.8950**	**0.9474**
	0.0415	0.0347	0.0859	0.0389	0.0752	0.0415
Random subspace	**0.9390**	**0.9473**	**0.9057**	**0.9736**	**0.8872**	**0.9396**
	0.0420	0.0344	0.0840	0.0359	0.0752	0.0414
Boosting	**0.9284**	**0.9355**	**0.9216**	**0.9352**	**0.8625**	**0.9284**
	0.0464	0.0397	0.0689	0.0960	0.0865	0.0464
C5.0	**0.9242**	**0.9290**	**0.9028**	**0.9456**	**0.8505**	**0.9242**
	0.0651	0.0625	0.0981	0.0725	0.1293	0.0651
Bagging	**0.8940**	**0.8989**	**0.8877**	**0.9003**	**0.7879**	**0.8940**
	0.0648	0.0632	0.0768	0.1065	0.1327	0.0648

Table 4.62: Balanced accuracy of single classifiers and ensemble methods across 30 runs in AMAN vs AMSAN classification. Ensemble methods are shown in bold.

Method	Balanced accuracy
kNN	0.9588
Random forest	**0.9474**
SVMLap	0.9406
Random subspace	**0.9390**
MLP	0.9344
Boosting	**0.9284**
RBF-DDA	0.9265
C5.0	**0.9242**
SVMGaus	0.9236
SLP	0.9202
SVMPoly	0.9189
C4.5	0.9138
SVMLin	0.9125
Naive Bayes	0.9091
Bagging	**0.8940**
LDA	0.8838
JRip	0.8827
BLR	0.8559
OneR	0.8437

accuracy above 0.90, only random subspace had a poor result. In this case, being AMAN the majority class, sensitivity was higher than specificity. Sensitivity ranged 0.95 - 0.99. Specificity ranged 0.50 - 0.87. Kappa showed a large variation, ranging 0.57 - 0.89. Shortly, almost all ensemble methods obtained a remarkable performance in AMAN vs MF classification in most metrics.

Table 4.63: Average results of ensemble methods across 30 runs in AMAN vs MF classification.

Method	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
Random forest	**0.9333**	**0.9625**	**0.9917**	**0.8750**	**0.8908**	**0.9333**
	0.0838	0.0452	0.0335	0.1706	0.1346	0.0838
Bagging	**0.9181**	**0.9313**	**0.9611**	**0.8750**	**0.8214**	**0.9181**
	0.0705	0.0502	0.0811	0.1431	0.1271	0.0705
Boosting	**0.9167**	**0.9417**	**0.9639**	**0.8694**	**0.8449**	**0.9201**
	0.0922	0.0655	0.0779	0.1662	0.1888	0.0967
C5.0	**0.9069**	**0.9313**	**0.9556**	**0.8583**	**0.8201**	**0.9069**
	0.0802	0.0664	0.0750	0.1421	0.1619	0.0802
Random subspace	**0.7543**	**0.8708**	**0.9917**	**0.5083**	**0.5749**	**0.7500**
	0.1227	0.0567	0.0253	0.2583	0.2123	0.1228

Table 4.64 shows the balanced accuracy of single classifiers and ensemble methods across 30 runs in AMAN vs MF classification. Half of single classifiers outperformed ensemble methods, even though these last obtained a high performance. Random subspace was the worst method, including single classifiers and ensemble methods.

Table 4.65 shows the average results of ensemble methods across 30 runs in AMSAN vs MF classification. The standard deviation is also shown. Four of five ensemble methods obtained a balanced accuracy above 0.85. Like the previous case, sensitivity was higher than specificity, because of the majority class effect. Sensitivity ranged 0.89 - 0.95. Specificity ranged 0.71 - 0.87. Low values were obtained in Kappa, which ranged 0.65 - 0.71. Overall, almost all ensemble methods obtained values on or above 0.85 in most metrics in AMSAN vs MF classification.

Table 4.66 shows the balanced accuracy of single classifiers and ensemble methods across 30 runs in AMSAN vs MF classification. Naive Bayes resulted better than all ensemble methods and the rest of single classifiers. Random subspace was the second best, and it almost reaches Naive Bayes performance, its base classifier.

4.5.4 Discussion

Our objective in this work was to investigate if ensemble methods were able to improve single classifiers in building a predictive model for GBS. We used the 16 relevant features identified with

Table 4.64: Balanced accuracy of single classifiers and ensemble methods across 30 in AMAN vs MF classification. Ensemble methods are shown in bold.

Method	Balanced Accuracy
SVMGaus	0.9542
SVMLap	0.9542
RBF-DDA	0.9528
Naive Bayes	0.9486
SVMLin	0.9431
*k*NN	0.9417
SVMPoly	0.9375
Random forest	**0.9333**
Bagging	**0.9181**
Boosting	**0.9167**
MLP	0.9125
C5.0	**0.9069**
SLP	0.9056
LDA	0.9056
C4.5	0.8889
BLR	0.8833
oneR	0.8806
JRip	0.8792
Random subspace	**0.7543**

Table 4.65: Average results of ensemble methods across 30 runs in AMSAN vs MF classification.

Method	Balanced Accuracy	Accuracy	Sensitivity	Specificity	Kappa	AUC
Random subspace	**0.8879**	**0.8928**	**0.8964**	**0.8750**	**0.6786**	**0.8857**
	0.0862	0.0556	0.0561	0.1574	0.1590	0.0855
C5.0	**0.8651**	**0.9130**	**0.9386**	**0.7917**	**0.7032**	**0.8651**
	0.0898	0.0379	0.0480	0.1979	0.1272	0.0898
Bagging	**0.8645**	**0.8957**	**0.8961**	**0.8329**	**0.7195**	**0.8978**
	0.1789	0.1749	0.1869	0.2177	0.2482	0.0757
Random forest	**0.8539**	**0.9217**	**0.9579**	**0.7500**	**0.7113**	**0.8539**
	0.1061	0.0418	0.0350	0.2177	0.1727	0.1061
Boosting	**0.8246**	**0.9058**	**0.9364**	**0.7127**	**0.6519**	**0.8246**
	0.1155	0.0458	0.0717	0.2463	0.1888	0.1155

Table 4.66: Balanced accuracy of single classifiers and ensemble methods across 30 runs in AMSAN vs MF classification. Ensemble methods are shown in bold.

Method	Balanced Accuracy
Naive Bayes	0.8980
Random subspace	**0.8879**
LDA	0.8656
C5.0	**0.8651**
Bagging	**0.8645**
MLP	0.8588
Random forest	**0.8539**
SLP	0.8493
SVMGaus	0.8480
SVMLap	0.8316
JRip	0.8248
Boosting	**0.8246**
*k*NN	0.8099
BLR	0.8042
C4.5	0.8026
SVMLin	0.8013
SVMPoly	0.7974
OneR	0.7811
RBF-DDA	0.6757

QSA-PAM method as predictors. Also, we applied five ensemble methods: boosting, bagging, C5.0, random forest and random subspace. We conducted three types of experiments: four GBS subtype classification, OVA classification and OVO classification. We compared the performance of both single classifiers and ensemble methods.

In four GBS subtype classification, two ensemble methods succeeded at improving average accuracy of all single classifiers: random forest and C5.0. Random forest surpassed *k*NN by almost one percentage point. C5.0 barely made it.

The best performance of ensemble methods was in AIDP vs AMAN, where all of them obtained a balanced accuracy above 0.90 and excellent results in all metrics. Overall, classifications with AMAN subtype present obtained the highest results.

Only in AIDP vs MF classification, random subspace was able to improve the performance of its base classifier, OneR in this case. This requires further investigation, in order to determine the cause of this result.

*k*NN was the best classifier in four cases, including single classifiers and ensemble methods.

Ensemble methods were outperformed by single classifiers in most cases. Three cases were the exception: four GBS subtype classification, AIDP vs AMSAN and AIDP vs MF. This result requires

further investigation.

Chapter 5

Conclusions and future work

In this work, we aimed at building a descriptive and a predictive model for GBS based on a real dataset using machine learning techniques. Our work was motivated by the hypothesis that an identification of the syndrome case is possible using a minimum number of medical features with at least 90% of accuracy.

In order to achieve our goal, we followed two phases. In the first phase, we built a descriptive model, which aimed at identifying a minimum relevant feature subset from an original 156-feature dataset. As showed in Figure 5.1, the descriptive model consists of two approaches. In a first approach of this phase, we applied filter methods and PAM clustering algorithm to fulfill our goal. Filter methods investigated included: CFS, chi-squared, information gain, symmetrical uncertainty and consistency.

Conclusions from first approach are [53]:

- PAM underperforms in the presence of redundant and irrelevant features and highlight the importance of feature selection methods.
- Purity is a good evaluation metric for clustering when the number of clusters is known, as in this case. Higher purity is easily achieved as the number of clusters increases.
- Information gain, symmetrical uncertainty and CFS showed to be highly efficient as they could obtain a reduced subset of relevant features that allow identifying four subtypes of GBS with high purity (0.7984). The first two methods coincided in the same seven variables. CSF selected 16 variables.

Our second approach of the descriptive model introduced a novel method called QSA-PAM. Different feature subsets were randomly selected from the dataset using the QSA metaheuristic. New datasets created using the feature subsets selected were used as input to PAM algorithm to build four clusters, corresponding to a specific GBS subtype each. Finally, purity of clusters was measured. An initial experiment consisting of 30 runs of QSA-PAM was performed. The frequency of selection of each feature across the 30 runs was computed. A new dataset was created using the resultant top fifty features from the initial experiment and used as input to a final experiment of 30 runs of QSA-PAM. Finally, a 16-feature subset was identified as the most relevant. Conclusions from this

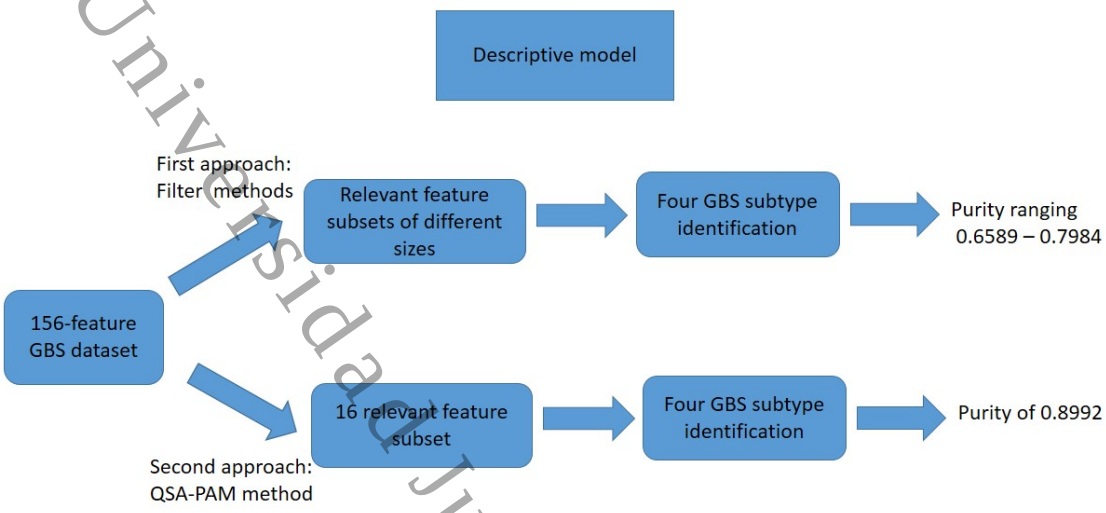

Figure 5.1: Approaches in the descriptive model.

approach are [18]:

- QSA-PAM was able to find a 16-feature subset that could build four clusters, each corresponding to a GBS subtype, with a purity of 0.8992.
- The ability of metaheuristics to find good solutions to combinatorial problems in large search spaces was confirmed.

In the second phase, we built a predictive model, which used the 16 relevant features identified with QSA-PAM method to classify GBS subtypes. Again, two approaches were used, as shown in Figure 5.2. In the first approach, classification experiments were conducted using single classifiers. Three types of experiments were performed: four GBS subtype classification, OVA classification and OVO classification.

Conclusions from first approach are:

- In four GBS subtype classification, we obtained an average accuracy >= 0.90 with half of classifiers investigated.
- In OVA classification, the two classes with the largest number of instances resulted in the best classification results, that is AMAN vs ALL and AMSAN vs ALL.

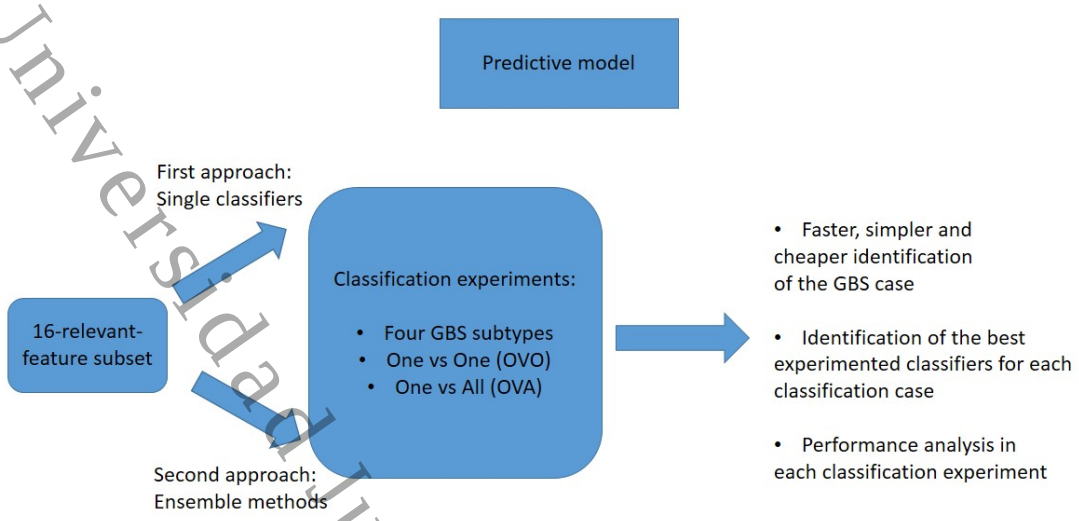

Figure 5.2: Approaches in the predictive model.

- In OVO classification, all classifications with AMAN subtype present resulted with the highest performance in balanced accuracy and most stable standard deviation across 30 runs.
- The analysis performed provides insight about the best classifiers for each classification case.

The second approach applied ensemble methods. Three types of experiments were performed: four GBS subtype classification, OVA classification and OVO classification. A comparison of performance between first and second approaches was made. Conclusions from second approach are:

- Ensemble methods outperformed single classifiers in four GBS subtype classification, AIDP vs AMSAN and AIDP vs MF. Random forest in two first cases and random subspace in the last case.
- However, in most cases, single classifiers resulted better than the ensemble methods investigated in this work.
- The analysis performed provides insight about the best classifiers for each classification case.

As to the specific objectives proposed at the beginning of this work. All of these were met. A machine learning model was built that was able to identify a minimum relevant feature subset. Also, a machine learning model that recognizes clinical patterns of Guillain-Barre syndrome subtypes was

implemented. Finally, a machine learning model that applies the predictor variables of Guillain-Barre syndrome in a predictive model was also built.

After results of all experiments and statistical analysis, we conclude that our hypothesis "determining predictor variables, subtypes and classification of patients with GBS through computational machine learning models allows the modeling of this syndrome with at least 90% of accuracy", is accepted.

5.1 Future work

In the first approach of the descriptive model, we applied five filter methods for feature selection. Three of them were able to obtain a reduced subset of relevant features that allow identifying four subtypes of GBS with high purity (0.7984). It would be interesting to investigate another filter methods like FCBF (Fast Correlation-Based Filter) [95] and INTERACT [96] in further studies. Also, more sophisticated methods of feature selection are recommended for analysis, such as those listed in [81, 84, 25].

As for the second approach of the descriptive model, a selection of the top fifty frequently selected features from the initial experiment was made. This selection can be more precisely made by using a threshold method in future studies.

Also, machine learning techniques such as neural networks or support vector machines could be used for clustering, and compared their efficiency with that of PAM. This study is planned for future research. In addition, using different metaheuristics to SA combined with other functions of energy in the process of feature selection can be further investigated for a more robust hybrid model.

Concerning the predictive model, the following topics will be further investigated:

- The causes of the lower performance of the ensemble methods investigated in this work than that of single classifiers, in most cases.
- The reason why random subspace was not able to improve the performance of its base classifier, in most cases.
- Methods to tackle the imbalanced data problem.
- The application of Case Based Reasoning (CBR) methodology.
- The building of a prognostic model for early prediction of outcome in GBS using machine learning.

References

[1] S. Akbayram, M. Dogan, C. Akgun, E. Peker, R. Sayin, F. Aktar, M.S. Bektas, and H. Caksen. Clinical features and prognosis with guillain-barre sindrome. *Annals of Indian Academy of Neurology*, 14(2), April-June 2011.

[2] E. Álvarez Cornelio and J.F. Arellano Pérez. Exploración y preprocesamiento de datos clínicos de pacientes con el síndrome de guillain-barre. Graduate College thesis, January 2013. Universidad Juárez Autónoma de Tabasco.

[3] J. Angus Webb et al. Bayesian clustering with autoclass explicitly recognizes uncertainties in landscape classification. *Ecography*, 30(4):526–536, 2007.

[4] O. Arbelaitz et al. An extensive comparative study of cluster validity indices. *Pattern Recognition*, 46(1):243–256, 2013.

[5] I. Arisi, M. D'Onofrio, R. Brandi, A. Felsani, S. Capsoni, G. Drovandi, G. Felici, E. Weitschek, P. Bertolazzi, and A. Cattaneo. Gene expression biomarkers in the brain of a mouse model for alzheimer's disease: Mining of microarray data by logic classification and feature selection. *Journal of Alzheimer's Disease*, 24:721–738, 2011.

[6] L. Arnaud et al. Cluster analysis of arterial involvement in takayasu arteritis reveals symmetric extension of the lesions in paired arterial beds. *Arthritis & Rheumatism*, 63:1136–1140, 2011.

[7] T. Asha, S. Natarajan, and K.N.B. Murthy. Data mining techniques in the diagnosis of tuberculosis. In P.J. Cardona, editor, *Understanding Tuberculosis - Global Experiences and Innovative Approaches to the Diagnosis*. InTech, Rijeka, Croacia, 2012.

[8] A.K. Ashbury and D.R. Comblath. Assessment of current diagnostic criteria for guillain-barre syndrome. *Ann Neurol*, 27(Suppl):s21– s24, 1990.

[9] A. V. Babkin, T. J. Kudryavtseva, and S. A. Utkina. Identification and analysis of industrial cluster structure. *World Applied Sciences Journal*, 28(10):1408–1413, 2013.

[10] E. Barati, M. Saraee, A. Mohammadi, N. Adibi, and M.R. Ahamadzadeh. A survey on utilization of data mining approaches for dermatological (skin) diseases prediction. *Journal of Selected Areas in Health Informatics*, may 2012.

[11] S. Boriah, V. Chandola, and V. Kumar. Similarity measures for categorical data: A comparative evaluation. In *Proceedings of the SIAM International Conference on Data Mining*, pages 243–254, 2008.

[12] M. Bramer. *Principles of Data Mining*. Springer-Verlag, London, U.K, 2007.

[13] U. Brandes et al. On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, pages 172–188, 2008.

[14] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth Inc., 1984.

[15] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[16] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[17] P. R. Burgel, J. L. Paillasseur, D. Caillaud, I. Tillie-Leblond, P. Chanez, R. Escamilla, I. Court-Fortune, T. Perez, P. Carré, and N. Roche. Clinical COPD phenotypes: a novel approach using principal component and cluster analyses. *European Respiratory Journal*, 36(3):531–539, September 2010.

[18] Juana Canul-Reich, José Hernández-Torruco, Juan Frausto-Solís, and Juan José Méndez-Castillo. Finding relevant features for identifying subtypes of guillain-barré syndrome using quenching simulated annealing and partitions around medoids. *International Journal of Combinatorial Optimization Problems and Informatics*, 6(2):11–27, May-August 2015.

[19] D. Chávez Esponda, I. Miranda Cabrera, M. Varela Nualles, and L. Fernández. Utilización del análisis de clusters con variables mixtas en la selección de genotipos de maíz. *Revista Investigación Operacional*, 30:209–216, 2010.

[20] S. Chebrolu, A. Abraham, and J. P. Thomas. Hybrid feature selection for modeling intrusion detection systems. In N. Pal, N. Kasabov, R. Mudi, S. Pal, and S. Parui, editors, *Neural Information Processing*, volume 3316 of *Lecture Notes in Computer Science*, pages 1020–1025. 2004.

[21] J. Cohen. A coefficient of agreement for nominal scales. *Educat. Psychol. Meas.*, 20(1):37–46, 1960.

[22] W.W. Cohen. Fast effective rule induction. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123. Morgan Kaufmann, 1995.

[23] Y. Couce, L. Franco, D. Urda, J.L. Subirats, and J. M. Jerez. Hybrid (generalization-correlation) method for feature selection in high dimensional dna microarray prediction problems. In J. Cabestany, I. Rojas, and G. Joya, editors, *Advances in Computational Intelligence*, volume 6692 of *Lecture Notes in Computer Science*, pages 202–209. 2011.

[24] P. Cunningham and S. J. Delany. Technical report ucd-csi-2007-4. Technical report, 2007.

[25] K. Dai, H.Y. Yu, and Q. Li. A semisupervised feature selection with support vector machine. *Journal of Applied Mathematics*, 2013, 2013.

[26] S. Das. Filters, wrappers and a boosting-based hybrid for feature selection. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 74–81, 2001.

[27] M. Dash and H. Liu. Feature selection for clustering. In T. Terano, H. Liu, and A. L. P. Chen, editors, *Knowledge Discovery and Data Mining. Current Issues and New Applications*, volume 1805 of *Lecture Notes in Computer Science*, pages 110–121. Springer Berlin Heidelberg,, 2000.

[28] M. Dash, H. Liu, and H. Motoda. Consistency based feature selection. In T. Terano, H. Liu, and A. L. P. Chen, editors, *Knowledge Discovery and Data Mining. Current Issues and New Applications*, volume 1805 of *Lecture Notes in Computer Science*, pages 98–109. Springer Berlin Heidelberg, 2000.

[29] Secretaría de Salud. *Guía de Práctica Clínica: Diagnóstico y manejo del Síndrome de Guillain Barré en la etapa aguda, en el primer nivel de atención*. México, 2008.

[30] J. Demsar. Statistical comparisons of classifiers over multiple datasets. *Journal of Machine Learning Research*, 7(2006):1–30, 2006.

[31] E. Docampo, A. Collado, G.a EscaramÃs, J. Carbonell, J. Rivera, J. Vidal, J. Alegre, R. Rabionet, and X. Estivill. Cluster analysis of clinical data identifies fibromyalgia subgroups. *PLoS ONE*, 8(9):e74873, 09 2013.

[32] T. Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[33] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. The kdd process for extracting useful knowledge from volumes of data. *Commun. ACM*, 39(11):27–34, November 1996.

[34] E. Fix and J. Hodges. Discriminatory analysis, nonparametric discrimination: Consistency properties. Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas, 1951.

[35] J. Frausto-Solis, E. Liñán García, M. Sánchez-Pérez, and J. P. Sánchez-Hernández. Chaotic multiquenching annealing applied to the protein folding problem. *The Scientific World Journal*, 2014:1–11, 2014.

[36] J. Frausto-Solis, E. F. Román, D. Romero, X. Soberon, and E. Liñán García. Analytically tuned simulated annealing applied to the protein folding problem. In *Proceedings of the 7th International Conference on Computational Science, Part II*, ICCS '07, pages 370–377, Berlin, Heidelberg, 2007. Springer-Verlag.

[37] J. Frausto-Solis, M. Sánchez-Pérez, E. Liñán García, and J. P. Sánchez-Hernández. Threshold temperature tuning simulated annealing for protein folding problem in small peptides. *Computational and Applied Mathematics*, 32(3):471–482, 2013.

[38] J. Frausto-Solis, H. Sanvicente-Sánchez, and F. Imperial-Valenzuela. Andymark: An analytical method to establish dynamically the length of the markov chain in simulated annealing for the satisfiability problem. In Tzai-Der Wang, Xiaodong Li, Shu-Heng Chen, Xufa Wang, Hussein Abbass, Hitoshi Iba, Guo-Liang Chen, and Xin Yao, editors, *Simulated Evolution and Learning*, volume 4247 of *Lecture Notes in Computer Science*, pages 269–276. Springer Berlin Heidelberg, 2006.

[39] Y. Freund and R. Schapire. Experiments with a new boosting algorithm. In *In Proceedings of the Thirteenth International Conference on Machine Learning*, pages 148–156, Bari, Italy, 1996.

[40] H. Fu, Z. Xiao, E. Dellandrea, W. Dou, and L. Chen. Image categorization using esfs: A new embedded feature selection method based on sfs. In J. Blanc-Talon, W. Philips, D. Popescu, and P. Scheunders, editors, *Advanced Concepts for Intelligent Vision Systems*, volume 5807 of *Lecture Notes in Computer Science*, pages 288–299. 2009.

[41] G.S. García Ramos and B. Cacho Díaz. Síndrome de guillain barré (sgb) diagnóstico diferencial. *Revista Mexicana de Neurociencia*, 6(5):448–454, 2005.

[42] J.C. Gower. A general coefficient of similarity and some of its properties. *Biometrics*, 27:857–871, 1971.

[43] R.D.M. Hadden, H. Karch, H.P. Hartung, et al. Preceding infections, immune factors and outcome in guillain-barre syndrome. *Neurology*, 56(6):758–65, 2001.

[44] M. Halkidi, Y. Batistakis, and M. Michalis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2-3):107–145, 2001.

[45] M. A. Hall. *Correlation-based Feature Selection for Machine Learning*. PhD thesis, University of Waikato, New Zealand, 1999.

[46] J. Han, M. Kamber, and J. Pei. *Data mining: concepts and techniques*. Morgan Kaufmann, San Francisco, USA, 2012.

[47] D.J. Hand and R.J. Hill. A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine Learning*, 45(2):171–186, 2001.

[48] Z.N. Hasan. New combined scoring system for predicting respiratory failure in iraqui patients with guillain-barre syndrome. *Broad Research in Artificial Intelligence and Neuroscience*, 1(4):5–12, 2010.

[49] T. Hastie et al. *The elements of statistical learning: Data Mining, inference and prediction*. Springer Verlag, 2008.

[50] A. Hernández Hernández. Contribución de la electrofisiología al diagnóstico diferencial entre el síndrome de guillain barré y la polineurorradiculopatía desmielinizante inflamatoria crónica. *Revista Cubana de Investigaciones Biomédicas*, 27(3), Diciembre 2008.

[51] J. Hernández Orallo et al. *Introducción a la minería de datos*. Pearson, 2005.

[52] José Hernández-Torruco, Juana Canul-Reich, and Juan Frausto-Solís. El aprendizaje computacional en el diagnóstico de enfermedades. *Komputer Sapiens*, 6(3):20–24, 2014.

[53] José Hernández-Torruco, Juana Canul-Reich, Juan Frausto-Solís, and Juan José Méndez-Castillo. Feature selection for better identification of subtypes of guillain-barré syndrome. *Computational and mathematical methods in medicine*, 2014:1–9, 2014.

[54] Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(8):832–844, August 1998.

[55] C.C. Hsu, C.W. Chang and C.J. Lin. A practical guide to support vector classification. Technical report, Department of Computer Science, National Taiwan University, 2003.

REFERENCES

[56] I. Inza, P. Larranaga, R. Etxeberria, and B. Sierra. Feature subset selection by bayesian network-based optimization. *Artificial Intelligence*, 123(1-2):157–184, 2000.

[57] L. Kaufman and P.J. Rousseeuw. Clustering by means of medoids. In Y. Dodge, editor, *Statistical Data Analysis Based on the L1–Norm and Related Methods*, page 405–416. North-Holland, 1987.

[58] R. Kohavi and G.H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[59] S. Kuwabara. Guillain-barre syndrome. *Drugs*, 64(6):597–610, 2004.

[60] Z. Lestayo-O'Farrill and J.L. Hernández-Cáceres. Análisis del comportamiento del síndrome de guillain-barré. consensos y discrepancias. *Revista de Neurología*, 46(4):230–237, 2008.

[61] H. Liu, J. Li, and L. Wong. A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns. *Genome Informatics*, 13:51–60, 2002.

[62] Y. Liu and M. Schumann. Data mining feature selection for credit scoring models. *Journal of the Operational Research Society*, 56(9):1099–1108, 2005.

[63] C. D. Manning, P. Raghavan, and H. Schutze. *An introduction to Information retrieval*. Cambridge University Press, 2009.

[64] M. Matsumoto and T. Nishimura. Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Trans. Model. Comput. Simul.*, 8(1):3–30, January 1998.

[65] A.B. Netto, A.B. Taly, G.B. Kulkarni, G.S. Uma Maheshwara Rao, and S. Rao. Prognosis of patients with guillain-barre syndrome requiring mechanical ventilation. *Neurol India*, 59(5):707–11, 2011.

[66] A. L. I. Oliveira, F. B. L. Neto, and S. R. L. Meira. A method based on rbf-dda neural networks for improving novelty detection in time series. In *In Proc. of the International FLAIRS Conference*, pages 670–675. AAAI Press, 2004.

[67] Samuel Ignacio Pascual Pascual. *Protocolos Diagnóstico Terapéuticos de la AEP: Neurología Pedi'atrica. Síndrome de Guillain Barré*. Asociación Espanola de Pediatría, Madrid, Spain, 2008.

[68] B.S. Paul, R. Bhatia, K. Prasad, M.V. Padma, M. Tripathi, and M.B. Singh. Clinical predictors of mechanical ventilation in guillain-barre syndrome. *Neurol India*, 60(2):150–3, Mar-Apr 2012.

[69] E. Pitt and R. Nayal. The use of various data mining and feature selection methods in the analysis of a population survey dataset. In *Proceedings of the 2nd International Workshop on Integrating Artificial Intelligence and Data Mining*, volume 84, pages 83–93, 2007.

[70] J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[71] A. Rajabally Yusuf and A. Uncini. Outcome and its predictors in guillain-barré syndrome. *Journal of Neurology, Neurosurgery and Psychiatry*, 83(7):711–718, may 2012.

[72] E. Sarhrouni, A. Hammouch, and D. Aboutajdine. Application of symmetric uncertainty and mutual information to dimensionality reduction of and classification hyperspectral images. *International Journal of Engineering and Technology (IJET)*, 4(5):268–276, 2012.

[73] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[74] T. Sharshar, S. Chevret, F. Bourdain, and J. Raphael. Early predictors of mechanical ventilation in guillain-barre syndrome. *Crit Care Med*, 31(1):278–83, 2003.

[75] C. Shearer. The crisp-dm model: The new blueprint for data mining. *Journal of Data Warehousing*, 5(4), 2000.

[76] Y. Shen et al. Quad-pre: A hybrid method to predict protein quaternary structure attributes. *Computational and Mathematical Methods in Medicine*, 2014, 2014.

[77] M. Solokova and G. Lapalme. A systematic analysis of performance measures for classification tasks. *Information processing and management*, 45:427–437, 2009.

[78] A. Soysal, F. Aysal, B. Caliskan, A. P. Dogan, B. Mutluay, N. Sakalli, S. Baybas, and B. Arpaci. Clinico-electrophysiological findings and prognosis of guillain-barre syndrome – 10 years of experience. *Acta Neurol Scand*, 123(3):181 – 186, 2011.

[79] D. J. Stracuzzi and P. E. Utgoff. Randomized variable elimination. *Journal of Machine Learning Research*, 5:1331–1364, 2004.

[80] U. Sundar, E. Abraham, A. Gharat, M.E. Yeolekar, T. Trivedi, and N. Dwivedi. Neuromuscular respiratory failure in guillain-barre syndrome: Evaluation of clinical and electrodiagnostic predictors. *Journal of the Association of Physicians of India*, 53(1):764–8, Sep 2005.

[81] A. M. Taha, A. Mustapha, and S.D. Chen. Naive bayes-guided bat algorithm for feature selection. *The Scientific World Journal*, 2013, 2013.

[82] J.H. Tsai. Data mining for dna viruses with breast cancer and its limitation. In Eugenia G. Giannopoulou, editor, *Data Mining in Medical and Biological Research*, pages 39–54. InTech, November 2008.

[83] A. Uncini and S. Kuwabara. Electrodiagnostic criteria for guillain-barré syndrome: A critical revision and the need for an update. *Clinical neurophysiology*, 123(8):1487–1495, 2012.

[84] M.S. Uzer, N. Yilmaz, and O. Inan. Feature selection method based on artificial bee colony algorithm and support vector machines for medical datasets classification. *The Scientific World Journal*, 2013, 2013.

[85] P. van Doorn. Acute flaccid paralysis. *Continuum*, 9(3):47–61, 2003.

[86] V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[87] L.H. Visser, P.I. Schmitz, J. Meulstee, et al. Prognostic factors of guillain-barre syndrome after intravenous immunoglobulin or plasma exchange. *Neurology*, 53(5):598–604., 1999.

REFERENCES

[88] B. R. Wakerley, A. Uncini, and N. Yuki. Guillain-barré and miller fisher syndromes - new diagnostic classification. *Nature Reviews Neurology*, 10:537–544, 2014.

[89] C. Walgaard, H.F. Lingsma, L. Ruts, et al. Prediction of respiratory insufficiency in guillain-barre syndrome. *Ann Neurol*, 1(67):781–7, 2010.

[90] C. Walgaard, H.F. Lingsma, L. Ruts, P.A. van Doorn, E.W. Steyerberg, and B.C. Jacobs. Early recognition of poor prognosis in guillain-barre syndrome. *Neurology*, 76(11):968–75, mar 2011.

[91] C. R. Williams-DeVane, D. M. Reif, E. C. Hubal, P. R. Bushel, E. E. Hudgens, J. E. Gallagher, and S. W. Edwards. Decision tree-based method for integrating gene expression, demographic, and clinical data to determine disease endotypes. *BMC Systems Biology*, 7:119, 2013.

[92] I.H. Witten, E. Frank, and M. A. Hall. *Data Mining: Practical Machine Learning Tools and techniques*. Morgan Kaufmann, third edition, 2011.

[93] S. Wu and J. Guo. A data mining analysis of the parkinson's disease. *iBusiness*, 3(1):71–75, 2011.

[94] I. Yoo, P. Alafaireet, M. Marinov, K. Pena-Hernandez, R. Gopidi, J.F. Chang, and L. Hua. Data mining in healthcare and biomedicine: A survey of the literature. *Journal of Medical Systems*, 36(4):2431–48, May 2011.

[95] L. Yu and H. Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 856–863, 2003.

[96] Z. Zhao and H. Liu. Searching for interacting features. In *Proceedings of the 20th international joint conference on Artifical intelligence*, pages 1156–1161, 2007.

[97] Z. Zheng, X. Wu, and R. Srihari. Feature selection for text categorization on imbalanced data. *ACM SIGKDD Explorations Newsletter*, 6(1):80–89, 2004.

[98] E. A. Zúniga González, A. Rodríguez de la Cruz, and J. Millán Padilla. Subtipos electrofisiológicos del síndrome de guillain-barré en adultos mexicanos. *Revista Médica del IMSS*, 45(5):463–468, 2007.

José Hernández Torruco

Descriptive and predictive models for Guillain-Barre syndrome based on clinical data using machine

Universidad Juárez Autónoma de Tabasco

Detalles del documento

Identificador de la entrega
trn:oid:::21:202706933

Fecha de entrega
26 mar 2025, 6:29 p.m. GMT-6

Fecha de descarga
18 may 2026, 4:54 p.m. GMT-6

Nombre del archivo
José Hernández Torruco.pdf

Tamaño del archivo
7.3 MB

164 páginas

43.816 palabras

254.928 caracteres

19% Similitud general

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para ca...

Filtrado desde el informe

- Bibliografía
- Texto citado
- Coincidencias menores (menos de 20 palabras)

Grupos de coincidencias

174 Sin cita o referencia 19%
Coincidencias sin una citación ni comillas en el texto

2 Faltan citas 0%
Coincidencias que siguen siendo muy similar al material fuente

0 Falta referencia 0%
Las coincidencias tienen comillas, pero no una citación correcta en el texto

0 Con comillas y referencia 0%
Coincidencias de citación en el texto, pero sin comillas

Fuentes principales

17% Fuentes de Internet
10% Publicaciones
0% Trabajos entregados (trabajos del estudiante)

Marcas de integridad

N.º de alertas de integridad para revisión

No se han detectado manipulaciones de texto sospechosas.

Los algoritmos de nuestro sistema analizan un documento en profundidad para buscar inconsistencias que permitirían distinguirlo de una entrega normal. Si advertimos algo extraño, lo marcamos como una alerta para que pueda revisarlo.

Una marca de alerta no es necesariamente un indicador de problemas. Sin embargo, recomendamos que preste atención y la revise.

Grupos de coincidencias

174 Sin cita o referencia 19%
Coincidencias sin una citación ni comillas en el texto

2 Faltan citas 0%
Coincidencias que siguen siendo muy similar al material fuente

0 Falta referencia 0%
Las coincidencias tienen comillas, pero no una citación correcta en el texto

0 Con comillas y referencia 0%
Coincidencias de citación en el texto, pero sin comillas

Fuentes principales

17% Fuentes de Internet
10% Publicaciones
0% Trabajos entregados (trabajos del estudiante)

Fuentes principales

Las fuentes con el mayor número de coincidencias dentro de la entrega. Las fuentes superpuestas no se mostrarán.

1	Internet	www.hindawi.com 4%
2	Internet	www.redalyc.org 4%
3	Internet	www.scielo.org.mx 2%
4	Internet	downloads.hindawi.com 2%
5	Publicación	Lecture Notes in Computer Science, 2015. 2%
6	Internet	www.researchgate.net 1%
7	Internet	docksci.com 1%
8	Internet	www.ncbi.nlm.nih.gov <1%
9	Publicación	Jose Hernandez-Torruco, Juana Canul-Reich, Juan Frausto-Solis, Juan Jose Mendez-... <1%
10	Internet	dokumen.pub <1%

11 Publicación
"Advances in Soft Computing", Springer Science and Business Media LLC, 2017 <1%

12 Internet
ijcopi.org <1%

13 Publicación
"Distributed Computing and Artificial Intelligence, 13th International Conference... <1%

14 Internet
m.moam.info <1%

15 Internet
scholar.archive.org <1%

16 Internet
vdoc.pub <1%

17 Publicación
Yungang Zhang, Bailing Zhang, Frans Coenen, Wenjin Lu. "Breast cancer diagnosi... <1%

18 Internet
www.slideshare.net <1%

19 Internet
pdfs.semanticscholar.org <1%

20 Internet
www.scirp.org <1%

21 Internet
doc.lagout.org <1%

22 Publicación
Chege, V., and A. S. Susuman. "Landholding and Fertility Relationship in Kenya: A ... <1%

23 Publicación
Santosh S. Rathore, Sandeep Kumar. "A decision tree logic based recommendatio... <1%