Instituto Tecnológico y de Estudios Superiores de Occidente

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Departamento de Electrónica, Sistemas e Informática

Maestría en Sistemas Computacionales

SELECCIÓN DE CARACTERÍSTICAS PARA CLASIFICACIÓN DE PRODUCTOS ALIMENTICIOS USANDO IMÁGENES HIPERESPECTRALES

TRABAJO RECEPCIONAL que para obtener el **GRADO** de
MAESTRO EN SISTEMAS COMPUTACIONALES

Presenta: Salvador Romo Macias

Dr. Miguel Ángel de la Torre Gómora y Dr. Himer Avila George

Tlaquepaque, Jalisco. August 18, 2025

Instituto Tecnológico
y de Estudios Superiores de Occidente

Recognition of official validity of higher education studies according to secretarial agreement 15018, published in the Official Gazette of the Federation on November 29, 1976.

Department of Electronics, Systems, and Computer Science

Master of Science in Computer Systems

FEATURE SELECTION FOR THE CLASSIFICATION OF FOODSTUFFS USING HYPERSPECTRAL IMAGES.

RECEPTIONAL WORK to obtain the **DEGREE** of **MASTER OF SCIENCE IN COMPUTER SYSTEMS**

Presents: Salvador Romo Macias

Dr. Miguel Ángel de la Torre Gómora and Dr. Himer Avila George

Tlaquepaque, Jalisco. August 18, 2025

AGRADECIMIENTOS

Quiero dar un agradecimiento especial al ITESO junto con el Dr. Iván Esteban, quien confió en mí y me dio la oportunidad de estudiar la maestría, brindándome apoyo a través de una beca completa, porque sin su ayuda, no habría podido siquiera concluir este viaje. A mis asesores, Miguel de la Torre, Himer Avila-George y Wilson Castro, quienes me dieron la oportunidad y la confianza para desarrollar un trabajo tan maravilloso y asombroso con temas que nunca antes había visto; debo decir que aprendí mucho de ellos. También por proporcionarme todas las herramientas, el conocimiento y el material para llevar a cabo esta investigación, así como la guía y los recursos para poder difundir en revistas importantes los artículos derivados de este trabajo.

ACKNOWLEDGMENTS

I want to give special thanks to ITESO along with Ivan Esteban, who trusted in me and gave me the opportunity to study for my master's degree, providing me with support through a full scholarship, because if it wasn't for his help, I wouldn't have been able to even conclude this journey. To my advisors, Miguel de la Torre, Himer Avila-George, and Wilson Castro, who gave me the opportunity and trust to develop such wonderful and amazing work with topics I had never seen before; I have to say that I learned a lot from them. Also, by providing me with all the tools, knowledge, and material to carry out this research, as well as the guidance and resources to be able to disseminate papers derived from this work in important journals.

DEDICATORIA

Dedico esta tesis a mi esposa Soemi Roxana Santillan Camacho, quien me motivó a continuar mis estudios de posgrados . Quien siempre me ha acompañado en cada etapa de mi vida desde que me gradué en la Licenciatura, hasta ahora con la maestría, y es un soporte muy importante para mí, ya que sin sus consejos, ánimo y amor no hubiera podido tomar este camino de las Ciencias Computacionales que me apasiona mucho.

DEDICATION

I dedicate this thesis to my wife, Soemi Roxana, who motivated me to continue my postgraduate studies. She has always accompanied me in every stage of my life, from when I graduated with my bachelor's degree until now with my master's, and she is a very important support for me. Without her advice, encouragement, and love, I would not have been able to take this path of Computer Science that I am so passionate about.

ABSTRACT

In this groundbreaking work, we unveil Iterative Covering Array Feature Selection (ICAFS), an innovative wavelength selection algorithm that seamlessly integrates the powerful architecture of covering arrays (CA). This method emerges as a pioneering extension of covering array Feature Selection (CAFS), designed to revolutionize and expand upon an indispensable tool for high-dimensional chemometrics applications. Consequently, in **chapter 1**, we will meticulously outline both the overarching and specific objectives associated with ICAFS, alongside the pivotal hypotheses we aim to validate and the pressing need for this novel approach in analyzing foodstuff hyperspectral images. Moving forward, in **chapter 2**, we delve into the fundamental theory essential for grasping this forward-thinking proposal, with a comprehensive review of concepts such as feature selection (FS), covering arrays, In-Parameter-Order (IPO), Binary Bat Algorithm (BBA), and In-Parameter-Order General (IPOG). This robust framework serves as the cornerstone for the ambitious goals we intend to achieve. In **chapter 3**, we conduct a thorough examination of the current state of the art concerning covering arrays and feature selection. Subsequently, in **chapter 4**, we detail the meticulous methodology employed to validate ICAFS, which includes an exhaustive comparative analysis among BBA and CAFS. Following this, **chapter 5** will showcase the results obtained in the preceding chapter, while also expanding on the current work through rigorous performance testing and introducing ICAFS to public benchmark challenge to discover the optimal interaction parameter t. Finally, in **chapter 6**, we offer our concluding insights along with potential future directions this transformative methodology might pursue.

RESUMEN

En este trabajo innovador, se presenta Iterative Covering Array Feature Selection, un algoritmo innovador de selección de bandas que integra la poderosa arquitectura de covering arrays. Este método surge como una extensión pionera de covering array Feature Selection, diseñado para revolucionar y expandir una herramienta indispensable para aplicaciones de quimiometría de alta dimensión. En consecuencia, en **capítulo 1**, se detalla meticulosamente tanto los objetivos generales como los específicos asociados con Iterative Covering Array Feature Selection, junto con las hipótesis clave que se busca validar y la justificacion en el análisis de imágenes hiperespectrales de alimentos. Avanzando, en **capítulo 2**, se expande en la teoría fundamental esencial para comprender esta propuesta innovadora, con una revisión exhaustiva de conceptos como feature selection, covering arrays, In-Parameter-Order, Binary Bat Algorithm y In-Parameter-Order General. Este marco sólido sirve como la piedra angular para los ambiciosos objetivos que se buscan alcanzar. En **capítulo 3**, realizamos un análisis exhaustivo del estado del arte actual en relación con covering arrays y feature selection. Posteriormente, **para el capítulo 4**, detallamos la metodología meticulosa empleada para validar Iterative Covering Array Feature Selection, que incluye un análisis comparativo exhaustivo entre Binary Bat Algorithm y covering array Feature Selection. A continuación, **capítulo 5** mostrará los resultados obtenidos en el capítulo anterior, al mismo tiempo que amplía el trabajo actual mediante pruebas de rendimiento rigurosas e introduce Iterative Covering Array Feature Selection a problemas fuera del campo de quimiometría para descubrir el parámetro de interacción óptimo t. Finalmente, en **capítulo 6**, se ofrecen los puntos de vista concluyentes junto con las posibles direcciones futuras que esta metodología podría seguir.

Contents

List of Figures

List of Tables

LIST OF ACRONYMS AND ABBREVIATIONS

BBA Binary Bat Algorithm. v, vi, 31, 35, 40, 46, 48, 58, 59

CA covering arrays. v, vi, 2–6, 8–11, 21, 24, 25, 27, 28, 39, 45, 58

CAFS covering array Feature Selection. v, vi, 3, 4, 15, 21, 23, 25–27, 29, 31, 35, 37, 39, 40, 42, 43, 46, 48, 58, 59

DL deep learning. 22

FS feature selection. v, vi, 2, 5, 14, 15, 21–23, 25, 29

GA genetic algorithm. 21

ICAFS Iterative Covering Array Feature Selection. v, vi, viii, ix, 2–4, 6, 10, 11, 14, 15, 20, 25–27, 29–35, 37, 39, 40, 42–48, 57–59

IPO In-Parameter-Order. v, vi, 6, 10, 11

IPOG In-Parameter-Order General. v, vi, 6, 10–12, 27, 39, 45, 47

KNN k-Nearest Neighbors. 43

MCA mixed covering array. 6, 9, 11

MI mutual information. 22, 23

ML machine learning. 5, 21, 22

NIR near-infrared. 21

OA orthogonal array. 21

OPF optimum path forest. 35

PSO particle swarm Optimization. 21

SA simulated annealing. 21

SVM Support Vector Machine. 43

1. INTRODUCTION

Overview: This chapter presents the objective that establishes the foundational basis for the ICAFS method, outlines the primary focus of the research, and introduces its intended application.

1.1 Background

Chemometric methods are typically employed to analyze food composition, aiming to automate and improve critical operations that include food authenticity [1], agriculture processing [2], ripeness classification [3], [4] and other related applications. In this context, hyperspectral imagery is commonly used to extract food properties that are undetectable within the visible spectrum, offering multiple measurements per sample. However, this comes at the cost of increased computational complexity during analysis. In order to reduce such computational complexity, feature selection methods are commonly applied to select relevant wavelengths for the application.

The role of wavelengths selection in chemometrics is substantiated: Anzanello and Fogliatto [5] posit that the presence of numerous predictors in chemometrics and industrial measurement may significantly complicate inferential analysis. As an illustration, wavelength selection has been applied for the detection of honey adulteration [6], the quality evaluation of *camellia oleifera seeds* [7], the classification of cacao varieties [8], among others [9]. Thus, the selection of wavelengths involves identifying the most relevant wavelengths for a particular application.

Wavelength selection represents a specialization within the domain of Feature Selection, specifically tailored to spectral data. Despite the absence of foundational literature explicitly addressing this concept; for example, Leardi, Boggia, and Terrile [10] in 1992 on the Journal of Chemometrics proposed an innovative Genetic Algorithm for feature selection, and Araújo, Saldanha, Galvão, *et al.* [11] Araújo and Saldanha articulated in their work from the early 2000's that the selection of spectrum wavelengths from chemometric measurements is really important, especially when the pronounced spectral data exhibit overlapping wavelengths or lack distinctive **features**, thereby underscoring the necessity for robust wavelength selection methods.

Therefore, the need for wavelength selection techniques is crucial in this domain. For example, nature-inspired algorithms, particularly those that emphasize wrapper methods for feature selection, are explored in [12], or even covering arrays Wrapper Methods in [8], [13], [14]. In filters, one can find recent methods such as the maximum dual interaction and maximum feature relevance [15] or the repeated elastic net technique [16], whereas for embedded methods, a decision trees algorithm that embeds feature selection on the model construction utilizing statistical concepts such as Gini-Index [17].

Although feature selection is extensively addressed within this study, the primary focus is not to provide a generalized framework, but rather a specialized tool for wavelength selection substantiated on the extensive analysis of high-dimensional chemometrics Applications through the extension of work done by Castro *et al.* [8]; proposing an enhancement of CAFS while concurrently extending its applicability to the broader chemometrics community. Such enhancement will be called ICAFS.

1.2 Justification

The study of feature selection algorithms has been extensively explored, as evidenced by previous research [18], which expanded the understanding of feature relevance and introduced several prominent categories for feature selection. Furthermore, Sklearn [19] provides a comprehensive evaluation of widely recognized algorithms for generic problems, alongside the many methodologies proposed in the ***state-of-the-art*** by various researchers. However, wavelength selection methods rigorously tested on high-dimensional chemometrics applications are not sufficiently explored. Consequently, practitioners and researchers remain uncertain and overwhelmed by the many existing tools on different categories that can be applied for chemometrics. Therefore, the primary rationale for extending CAFS to ICAFS is to present a well-validated tool for wavelength selection.

1.3 Problem Statement

CAFS was introduced to decrease the number of wavelengths in a high-dimensional task that involves the classification of six Amazonian cacao varieties using a range of wavelengths within the NIR spectrum. The study thoroughly elucidates the methodologies used for the extraction of each observation. CAFS is based on covering arrays [20] to model the interactions between the available wavelengths in the dataset. The result of this interaction is a binary matrix, where each row represents a subset of selected wavelengths. Once a subset of features, deemed optimal, was selected, the matrix was halved to correspond to the new subset of wavelengths. However, the interaction among the new wavelengths was compromised, preventing the assurance of a novel interaction upon each selection of a new subset of wavelengths.

1.4 Hypothesis

In order to address the limitations discussed in the preceding section, we have proposed an analysis that encompasses six high-dimensional Chemometrics tasks, to be executed in accordance with ICAFS. This approach integrates the construction of covering arrays within CAFS. The central thesis of this work is to demonstrate that ICAFS can generalize wavelength selection by processing a broader selection of chemometrics tasks and exceed the performance observed with cafs in the Amazonian cacao nibs.

1.5 Objectives

1.5.1 General Objective

The overarching aim of this research is to extend and generalize the framework established in CAFS to ICAFS, with the purpose of developing an innovative tool for wavelength selection specifically designed to enhance inferential tasks within the domain of chemometrics.

1.5.2 Specific Objectives

To substantiate the present study, we will conduct a comprehensive set of experiments aimed at achieving the following objectives:

1. Extends CAFS to ICAFS by incorporating the construction of covering arrays on each iteration, and generalizing ICAFS as specialized method that can be used on any classification chememotrics application demonstrating its effectiveness by trialling it against 6 well known chemometric applications.
2. Provide a deep comparison against CAFS and a well established meta-heuristic method for feature selection.
3. Validate its credibility by testing the proposed algorithm against very-high dimensional synthetic datasets. Additionally, try out ICAFS in applications out of chemometrics to demonstrate its applicability in different domains.
4. Release the proposed algorithm as an open-source library, thereby facilitating its integration and utilization by researchers and practitioners specializing in hyperspectral feature selection.

1.6 Contribution

The groundbreaking innovation of this research is embodied in ICAFS, an algorithm designed for the strategic selection of relevant wavelengths within the complex realm of high-dimensional chemometrics applications. This tool serves as a game-changer for practitioners and researchers alike, empowering them to significantly reduce the wavelength search space, thereby enabling the training of sophisticated models with reduced cost and maximal efficiency.

2. THEORETICAL / CONCEPTUAL FRAMEWORK

***Overview:** The proposed tool is grounded in several key concepts—most notably covering arrays, feature selection, and machine learning, among others. Accordingly, a clear understanding of this theoretical framework is essential for comprehending the proposed methodology.*

2.1 Covering Arrays

Before getting into detail in covering arrays, we will explain T-way interaction, the foundational principle on which covering arrays it is built. **T-way** testing [21], also known as interaction testing, emphasizes the examination of combinations of inputs rather than evaluating every conceivable combination. For example, consider the task of testing a system composed of 20 parameters, each having 10 distinct values. Employing a brute force methodology necessitates exploring 10^{20} different combinations. However, as indicated by *t-way* testing, software faults typically arise from the interaction of a limited subset of inputs. Consequently, examining all conceivable combinations involving t parameters is considerably more efficient. In view of this, covering arrays constitutes a robust combinatorial data structure that models input interaction across t parameters.

Thereby, covering arrays underpins the fundamental tenets of ICAFS. Consequently, this section elucidates covering arrays and presents diverse methodologies for the construction of covering arrays across various categories. Moreover, an exhaustive analysis of IPO/IPOG will be undertaken, as they are relevant algorithms to construct ICAFS. A covering array is a combinatorial data structure defined by the function $\mathrm{CA}(N, k, t, v)$, where N and k stand for the number of rows and columns, t the strengths or interactions between the columns, and v is the size of the alphabet defined by $\mathbb{Z}_v$. This can be seen as test suites for software and hardware components where each column represents a **paramater**. In addition, a significant challenge referenced in covering arrays pertains to the construction of CAN (k, t, v). CAN can be defined as: given a set of k parameters, a specified strength t, and an alphabet v, the objective is to construct a covering array that minimizes the number of rows. This problem has been proven to be **NP**-hard in the context of pairwise testing [22]. Thus, the development of optimal methodologies for constructing covering arrays constitutes a significant area of research. In the next section, we will review two methods and review some essential categories.

An example of a covering array is depicted in Figure 2.1 with the following properties: $\mathrm{CA}(12, 7, 2, 3)$. The alphabet is $Z_v = \{0, 1, 2\}$ because each cell of the matrix can take three distinct values $\{0, 1, 2\}$, $N = 12$ and each row represents a different test for the system, and $k = 7$ are the different parameters to test in the system. Finally, t represents the interaction between two parameters such that each of the $\binom{7}{2}$ sub-arrays formed by $t = 2$ columns encompasses v^t tuples that extend over $\mathbb{Z}_v$; this means, for the matrix to meet the condition to be a covering array, each sub-array composed of 2 columns has to cover $3^2 = 9$ tuples at least once. To give an example, for every sub-array of length two it has to cover the following $v^t = [(0,0), (0,1), (0,2), (1,0), (1,1), (1,2), (2,0), (2,1), (2,2)]$ tuples.

Conversely, mixed covering array (MCA) are predominantly observed in practical applications due to the variability in parameter values among software and hardware components. Thereby, defined as $\mathrm{MCA}(N, k, t, (v_0, v_1, \ldots, v_{k-1}))$, where each $j \in \{0, \ldots, k-1\}$ comes from an alphabet with a cardinality of v_j. Consequently, for any combination of t distinct

$$\begin{pmatrix} 1 & 2 & 0 & 0 & 2 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 2 & 2 & 2 & 1 & 2 & 1 & 2 \\ 2 & 1 & 0 & 1 & 0 & 2 & 1 \\ 0 & 0 & 2 & 1 & 1 & 0 & 1 \\ 0 & 2 & 0 & 2 & 1 & 2 & 2 \\ 2 & 2 & 1 & 2 & 0 & 0 & 0 \\ 1 & 1 & 2 & 0 & 0 & 0 & 2 \\ 1 & 0 & 2 & 2 & 2 & 2 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 2 & 0 & 1 & 0 & 1 & 2 & 2 \\ 0 & 1 & 1 & 2 & 2 & 1 & 1 \end{pmatrix}$$

Figure 2.1: **CA**$(12, 7, 2, 3)$ **with** $t = 2$**,** $Z_v = \{0, 1, 2\}$**,** $k = 7$ **and** $N = 12$ **because every sub array of length** 2 **covers at least once all possible tuples formed from** $Z_v = \{0, 1, 2\}$

columns; these columns must cover all t-tuples that can be formed from the Cartesian product of its respective alphabet at least once. Sometimes the exponential notation $s_0^{u_0}, \ldots, s_l^{u_i}$ can be used to indicate that the MCA has u_i columns with cardinality s_l, an example of this can be observed on Figure 2.2 and such forms of Covering Arrays are more prevalent in hardware and software components, as each input may possess distinct values.

$$\begin{pmatrix} 0 & 0 & 3 & 6 \\ 0 & 1 & 1 & 1 \\ 0 & 2 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 3 & 6 \\ 1 & 2 & 3 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 6 \\ 0 & 2 & 1 & 6 \\ 0 & 1 & 3 & 0 \end{pmatrix}$$

Figure 2.2: **MCA**$(10, 4, 2, (\{0, 1\}, \{0, 1, 2\}, \{0, 1, 3\}, \{0, 1, 6\}))$ **because every sub array of length** 2 **covers at least once all possible** t**-tuples formed by the Cartesian products of its respective alphabet**

2.1.1 Categories on Covering Array Construction

Therefore, the broad researchers and practitioners community has been fully advocated for investigating diverse methods for constructing optimum CAN, as noted in the investigation driven by Torres-Jiménez [23], who compiles diverse methodologies to generate covering arrays from influential authors and classifies them into six main categories. So, in this section, we will only review algebraic, recursive, exact, and greedy approaches to provide the reader with an understanding of significant construction of covering arrays methods.

1. **Algebraic Methods**: These methods employ mathematical formulas and algebraic frameworks such as groups and finite fields, as well as operations involving one-dimensional entities such as polynomials, vectors, and sequences. They typically do not utilize two-dimensional configurations, such as orthogonal arrays or difference matrices.

   - **Case** $t = v = 2$ [24]–[26]: Constructs a $\mathrm{CA}(N; 2, k, 2)$ by placing as columns distinct binary vectors of length N that begin with 0 and have $[\frac{N}{2}]$ 1's.
   - **Cyclotomy** [27]: Uses cyclotomic vectors derived from a prime-power $q \equiv 1 \pmod{v}$ and a primitive element ω of $\mathbb{F}_q$. Then it generates a $q \times q$ cyclotomic matrix $A_{q,v}$ which is a CA under certain conditions. If bigger strengths are needed, a $vq \times v$ matrix $B_{q,v}$ is constructed by vertically juxtaposing $A_{q,v} + c$ for $0 \leq c \leq v$ which is a CA for $t \geq 3$ on certain conditions.

2. **Recursive Methods**: These methods construct new CAs from smaller ones.

   - **Product of CAs** [28]: Generates a $\mathrm{CA}(N_1+N_2; 2, k, v)$ from $X = \mathrm{CA}(N_1, 2, l, v)$ and $Y = \mathrm{CA}(N_2; 2, kl, v)$ by taking the first input array, X, and place l copies of it side-by-side. This creates the top block of the new array with N_1 rows and $k \times l$ columns, then underneath the top block, For each of the l copies of X a block of size $N_2 \times k$ is placed below it by taking the $i - th$ column of the second input array, Y, and replicating that single column k times.
   - **Duplication** $2k$ [23]: This approach generates a strength-3 binary covering arrays by utilizing both a strength-3 and a strength-2 binary array. Specifically, given a $X = \mathrm{CA}(N_3; 3, k, 2)$ of $t = 3$ with k columns and a $Y = \mathrm{CA}(N_2; 2, k, 2)$ of $t = 2$, two replicas of X are arranged adjacently for the upper portion. Below the first instance of X, the array Y is positioned. Subsequently, beneath the second instance of X, the bitwise complement of Y is arranged. The result of the aforementioned procedure is illustrated on Eq. 2.1.

$$Z = \begin{pmatrix} X & X \\ Y & \overline{Y} \end{pmatrix} \tag{2.1}$$

3. **Exacts Methods**: These construct CAs by exhaustive search, typically for small CAs. They often try to break symmetries (row/column permutation, symbol relabeling).

- **The Automatic Generator Exact** [29], [30]: EXACT builds the matrix one single cell at a time. It very carefully places a value in a cell and then checks if it's on the right track. After it places a value, it checks whether the new assignment implies a specific value for another cell or if it leads to a contradiction with the covering array requirements. This is called "constraint propagation". If it finds a problem or a dead end, it backtracks and tries a different value. This means it looks at all possibilities to find the smallest possible chart that works. It is best suited for small covering arrays.
- **New Backtracking Algorithm** [31]: It is a search method for constructing binary covering arrays ($v = 2$) of a specific strength and size.The algorithm builds the covering array one column at a time. It uses a backtracking search, which explores potential solutions and "backtracks" when it hits a dead end, to speed up the search, the algorithm imposes a lexicographic (dictionary-style) order on the columns, helping the algorithm to avoids re-exploring solutions, also it only considers candidate columns that are "balanced" in their symbols, so it must have a specific numbers of 0 and 1.

4. **Greedy Methods**: These generate good solutions in a short time, often constructing arrays one row or one parameter at a time. Most of the time the covering arrays generated are deterministic, meaning the output covering arrays is the same on every run.

    - **Test Case Generator (TCG)** [32]: is a greedy algorithm designed for constructing strength-two covering arrays mixed covering array. A notable characteristic of TCG is its incremental construction of the array, adding one row at a time. The process begins by permuting the k parameters (columns) of the covering array MCA, arranging them in a non-increasing order according to the cardinality of their respective alphabets. The number of candidate rows considered can be up to M, where M represents the cardinality of the largest alphabet among the parameters. Each candidate row is developed sequentially, element by element. The construction procedure involves counting the number of new pairs covered by adding an element to the current partial row. In the final step, the algorithm selects the candidate that covers the most new pairs and repeats the procedure until the covering array MCA is completely constructed.

Furthermore, two additional categories are identified: metaheuristics and transformation. The focus of metaheuristics is on utilizing nature-inspired and swarm-optimization heuristics to guide the search toward an optimum covering arrays. Conversely, transformation is concerned with altering the structure of pre-existing covering arrays.

2.1.2 In-Parameter-Order: A Greedy deterministic approach to construct CA of t=2

Even though IPO will not be a functional part for ICAFS is important to be detailed in order to understand the core behavior of IPOG. IPO strategy serves as a method for generating pairwise test sets. This technique incrementally constructs a comprehensive test suite by incorporating one parameter at a time into an already established covering array comprised of the v^t tuples from the first t parameters. A notable feature of this algorithm is its construction of covering arrays through two distinct phases: horizontal and vertical. The overall pseudocode is presented in Algorithm 18.

Algorithm 1 In-Parameter-Order (IPO)

Initialize with the first two parameters p_1 and p_2
$\mathcal{T} \leftarrow \{(v_1, v_2) \mid v_1$ is a value of p_1 and v_2 is a value of $p_2\}$
if $n = 2$ **then**
    return $\mathcal{T}$
end if
Process the remaining parameters
for parameter $p_i, i \leftarrow 3$ to n **do**
    Horizontal Growth: Extend existing tests
    for each test $(v_1, \ldots, v_{i-1}) \in \mathcal{T}$ **do**
        Choose a value v_i for parameter p_i
        Replace test with $(v_1, \ldots, v_{i-1}, v_i)$
    end for
    Vertical Growth: Add new tests for uncovered pairs
    while $\mathcal{T}$ does not cover all pairs between p_i and each of $p_1, \ldots, p_{i-1}$ **do**
        Add a new test for $(p_1, \ldots, p_i)$ to $\mathcal{T}$
    end while
end for
return $\mathcal{T}$

Consider an initial pairwise test set for parameters A and B, represented as $\{(A_1, B_1)(A_1, B_2)(A_2, B_1)(A_2, B_2)\}$. Upon the introduction of a new parameter C, horizontal growth extends these four tests by adding values from the C parameter to existing sets, resulting in tests denoted by $X = \{(A_1, B_1, C_1), (A_1, B_2, C_1), A_2, B_1, C_2), (A_2, B_2, C_2)\}$. Following this sample, the test set still comprises only four tests, and each is insufficient to cover all interactions with C. Consequently, vertical growth is executed. This step introduces entirely new tests into the set if necessary to address pairwise combinations unaddressed during the horizontal phase. For example, suppose the addition of parameter C to X reveals six uncovered tuples $\{(A_1, C_2), (A_1, C_3), (A_2, C_1), (A_2, C_3), (B_1, C_3), (B_2, C_3)\}$. To cover these tuples, the vertical growth algorithm generates new tests. To illustrate, in order to address the missing pair (A_1, C_2), a test such as $(A_1, *, C_2)$ is appended, wherein $*$ signifies a don't care value; these don't care values will have the function to serve as wildcards, so uncovered

tuples can use it to avoid adding more tests. If it's not possible, the algorithm will attempt to cover remaining tuples by adding new tests; as an example, let's suppose we want to add another tuple (B_1, C_3) by modifying previously added tests with don't care values $(A_1, *, C_3)$, obtaining (A_1, B_1, C_3); hadn't the tuple been covered using a wildcard, it will need to append a new test with the conditions previously stated. This is one of the weaknesses of In-Parameter-Order, as it will add unnecessary tests.

2.1.3 In-Parameter-Order General: A Greedy deterministic approach to construct CA with greatest strength

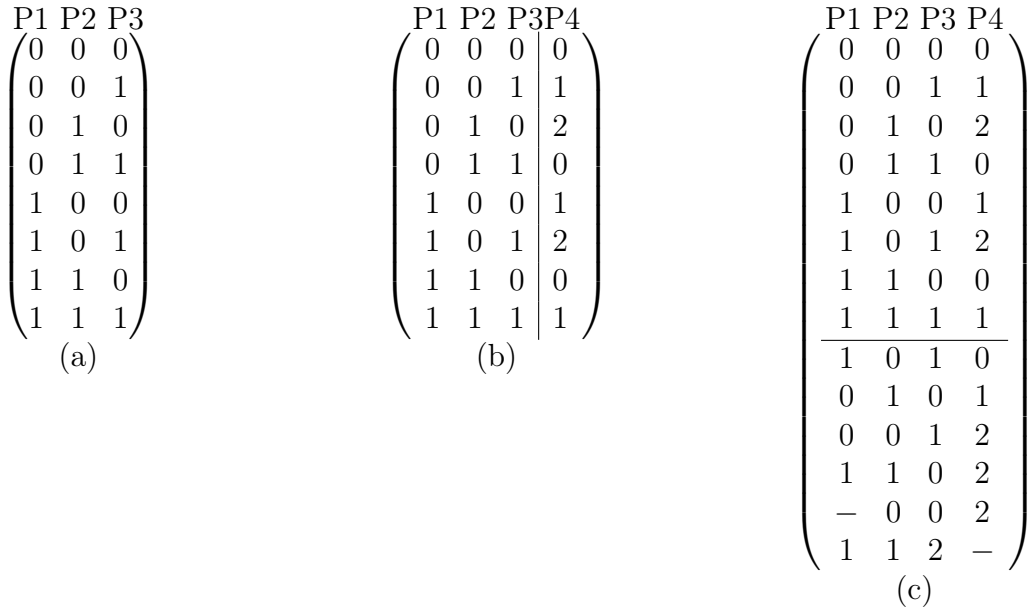

Figure 2.3: Vertical and horizontal growth extensions.

The selection of IPOG to be part of ICAFS is substantiated by the fact that IPOG builds faster covering arrays and allows the construction of greater t. Such a feature will be relevant because in Section 5.3 we will make a deep discussion on picking the best t for feature interaction. IPOG adopts a methodology akin to that of IPO with the difference that it could work on MCA and $t \geq 2$. Hence, to ensure that all t values' tuples across the interaction of every t parameter have been covered at least once, it incorporates an optimized access data structure. This structure is employed to check whether the test τ_i has adequately encompassed the most t-tuples possible, bearing in mind current P_i. It is noteworthy that both algorithms exhibit deterministic characteristics, thereby consistently producing the same covering array upon each execution. The Algorithm 2 elaborates on the IPOG introduced by Lei, Kacker, Kuhn, *et al.* [33]. This methodology follows the very same procedure from IPO, vertical and horizontal growth depicted in Figure 2.3, meaning that just after considering P_i, it has to calculate all T-way parameters' subsets, taking into account the new P_i in order to check if test $\tau_i + P_j.\alpha$ has covered the most values' tuples for each of the parameter interactions of a defined t, bearing in mind

P_i. For this reason, when $t \geq 2$, generating all t-way subsets containing P_i with nested loops would be overwhelming; instead, an approach working with a binary representation of parameter tuples is proposed next. Let P_4 be a parameter being processed and $t = 3$ with the initial tuple consisting of the P_{i-2}, P_{i-1}, P_i resulting in the tuple $\{P_2, P_3, P_4\}$. Later, we will start by creating a binary vector 0111 by placing 1's in the rightmost (least significant) position. Now, in order to generate the next permutation containing P_4 we follow the below two rules.

- First, check if the last digit of the vector is 1. If true, it looks for the least significant (rightmost) digit g that is a 0 and is immediately followed by a 1. Then, change g to 1 and the digit following it to 0. Let's illustrate this step; assume we have a combination of this parameters P_2, P_4, P_5 represented in binary format as 01011 confirming that the last digit is 1. From digits 4_{th} and 5_{th} we notice this sequence 11, while 3_{th} and 4_{th} g is 01; thus, 0 at 3_{rd} position is changed to a 1, and 1 at position 4 to 0. This will generate the next tuple 01101, representing parameters interaction of P_2, P_3, P_5.
- In case the last digit of the vector is 0; find the least significant (rightmost) digit g that is 0 and is followed by a 1; then, count the number of 1's, c, that appear before digit g and change digit g from 0 to 1, then flip all digits after g to 0 except for the last $n - c - 1$ digits, which are set to 1. Here, n is the total number of 1's in the vector. Following that rule, we can start with the next binary vector 10110, the rightmost 0 followed by a 1 is at 2_{nd} position, so, the number of 1 before the second position is $c = 1$, change the 0 at 2_{nd} position to 1, getting the next partial vector 11110. Now, digits after 2_{nd} position are flipped from 1 to 0, except for last $n - c - 1 = 3 - 1 - 1 = 1$ digits which are set to 1, resulting in vector 11001.

Now, in order to get all tuple values $(P_i.\alpha, P_j.\beta, P_k.\gamma)$, the algorithm repeatedly adds 1 to the rightmost digit in tuples. This is accomplished by finding the least significant (rightmost) digit that is not at its maximum value in its corresponding base - 1 and then incrementing it by one. When the max base - 1 has been reached , it increments the leftmost character which is not at its max base - 1, adds 1, and all digits to its right back to 0. This process of adding 1 is repeated until a combination where every value is at its maximum base - 1 has been reached.

By applying these two rules, we can generate all t-tuples involving parameter P_i. But now we need to make sure that adding $P_i.\alpha$ to τ_i covers the most t-tuples containing P_i. For that reason, IPOG introduces a bidimensional datastructure. The first level are the indexes that correspond to all t-tuples involving P_i; the second level is a bit map where each bit represents a value's $t-$tuple to be covered. The indexes can be calculated as follows; from the next tuple, $\{P_0, P_2, P_3\}$ we can derive the next variables, $i = 0$, $j = 2$, $k = 3$ and by applying the next formula $3 * i + 2 * (j - i - 1) + (k - j - 1)$ and substituting each variable, we will have $3 * 0 + 2 * (2 - 0 - 1) + (3 - 2 - 1) = 2$ which is the index to access the tuple position into the array. For accessing the value's tuple bitmap position,

we can calculate the next formula; assume we have the following tuples with a valid value combination $\{P_{0.1}, P_{2.0}, P_{3.1}\}$, then, by computing Eq 2.2, the values' tuple index is obtained. Here, v_k is the zero-based index of the value for the k-th parameter in the tuple; n the number of parameters in the combination. Next, d_j represents the domain size for the j-th parameter and Π the product of a sequence of terms.

$$\sum_{k=1}^{n} \left(v_k \prod_{j=k+1}^{n} d_j \right) \tag{2.2}$$

With the incorporation of these additional procedures, growth and vertical extension, is now capable of generating all t-tuples to generate a pseudo-optimum covering array on greater strengths with multiple vocabulary cardinalities.

Algorithm 2 IPOG (t,P)

```
1: ts ← {}
2: denote the parameters in P, in an arbitrary order, as P_1, P_2, ..., P_n
3: add into test set ts a test for each combination of values of the first t parameters
4: for i ← t + 1 to P_n do
5:     π ← t way combination of values involving the Parameter P_i and t − 1 parameter among the first i − 1 parameter
6:     for τ = (v_1, v_2, ..., v_{i−1}) ∈ ts do
7:         for v_i ∈ P_i do
8:             τ' = (v_1, v_2, ..., v_{i−1}, v_i)
9:             if τ' covers the most t tuples from π then
10:                τ = τ'
11:            end if
12:        end for
13:    end for
14:    //vertical extension for parameter P_i
15:    for σ ∈ π do
16:        if σ ∈ ts then
17:            remove σ from π
18:        else
19:            change an existing test to cover σ if possible, otherwise add a new test to cover σ
20:        end if
21:    end for
22: end for
```

2.2 Feature Selection

Feature Selection is a key concept for ICAFS, even though the main objective is to select wavelengths and not features. The methodology in the study incorporates key concepts from feature selection. So, foundational concepts should be addressed to comprehend real ICAFS objectives.

Feature Selection is the process of selecting a subset of features that best represents the problem, but sometimes, the removal of non-relevant and noisy information may have different implications and not always lead to greater performance; for example, there are cases in which high correlation may exist between two features; in other words, these features may be relevant for the task, and removing one of the features will not negatively affect the learning performance. However, other features may not be relevant for classification, even if they are strongly independent from the rest. Even more, a noisy and non-relevant feature alone can represent a decrease in classification accuracy, but two noisy features can complement each other to distinguish samples from different classes; for that reason, the broad scientific community has developed diverse feature selection algorithms on different categories to tackle different objectives. In general, one thing that is irrefutable for feature selection is that it could help overcome (1) models that get overfitted (2) degeneration of algorithms' performance due to highly dimensional data sets, and (3) usage of a lot of computational resources [34], [35]. Furthermore, Huang, Samuel [36], detailed an extensive taxonomy for supervised feature selection methods based on the output of each algorithm. Figure 2.4 depicts such taxonomy, broken down in three main categories: Feature Ranking, Subset Selection, and Embedded methods.

1. Feature Ranking Methods: Feature ranking methods assess individual features and rank them according to their relevance to the target concept. The output is a prioritized list of features. These are further divided into Pair-wise Ranking and Simultaneous Ranking.

    - **Pair-wise Ranking**: These methods evaluate the dependency between each single feature and the target concept, one at a time. The criteria for this evaluation fall into four categories.
        - **Correlation**: Measures the statistical correlation between a feature and the target, such as using Pearson's correlation coefficient [37].
        - **Uncertainty**: Uses information theory concepts like entropy to measure the reduction in uncertainty about the target when a feature's value is known. Information gain [38] is a common criteria used under this category.
        - **Hypothesis Test**: Employs statistical tests to determine if a feature is independent of the target class. The p-values from tests like the Chi-squared test or t-test are used for ranking.

- **Discriminative Power**: Evaluates how well a single feature can discriminate between different class labels, for instance, by measuring the classification error or the area under the ROC curve for a model built with that single feature.

- **Simultaneous Ranking**: Calculate relevance weights for all features at the same time considering their joint relationship with the target concept; this definition is completely driven by the Relief Algorithm [39].

2. Subset Selection Methods: The goal of these methods is to produce an optimal subset of features that should be collectively relevant to the target. They require a search strategy to navigate the space of possible feature subsets.

   - **Subset Configuration Methods**: Evaluate relevant feature subsets by analyzing both the correlation between features and targets selected, as well as the correlation among the features to configure an optimum candidate.
   - **Subset Evaluation Methods**: These methods determine the quality of a feature subsets by obtaining the best subsets with higher score or applying statistical analysis.
     - Model-Specific (Wrappers): Implements cross-validation on candidate subsets as the criterion to guide the search toward the best feature subset. These methods generally rely on heuristics search to navigate the feature space.
     - Model-Independent (Filters): They assess subsets based on statistical insights from data relationship, making candidates subsets unbiased toward any particular algorithm. Examples of criteria include inconsistency rate[40] and inference correlation [41].

3. Embedded Methods: Incorporate the feature selection process directly into the construction of the model. Features that remain in the final model are a byproduct of the modeling process itself.

2.3 BBA (Binary Bat Algorithm)

This algorithm will be applied to the same experimental data utilized by ICAFS and CAFS. The primary objective is to facilitate a comparative analysis of the quality and quantity of wavelengths selected. The inclusion of this algorithm in the current study is primarily justified by belonging within the **wrapper methods**, offering a valuable opportunity to evaluate a ***state-of-the-art*** procedure in comparison to ICAFS and CAFS, given their shared objectives on feature selection discussed in Subsection 4.2.

The Bat algorithm is a bioinspired metaheuristic technique described in [42], and inspired by the behavior of bats on how a swarm of micro-bats locates their prey/food using echolocation. The behavior of micro-bats is defined as follows:

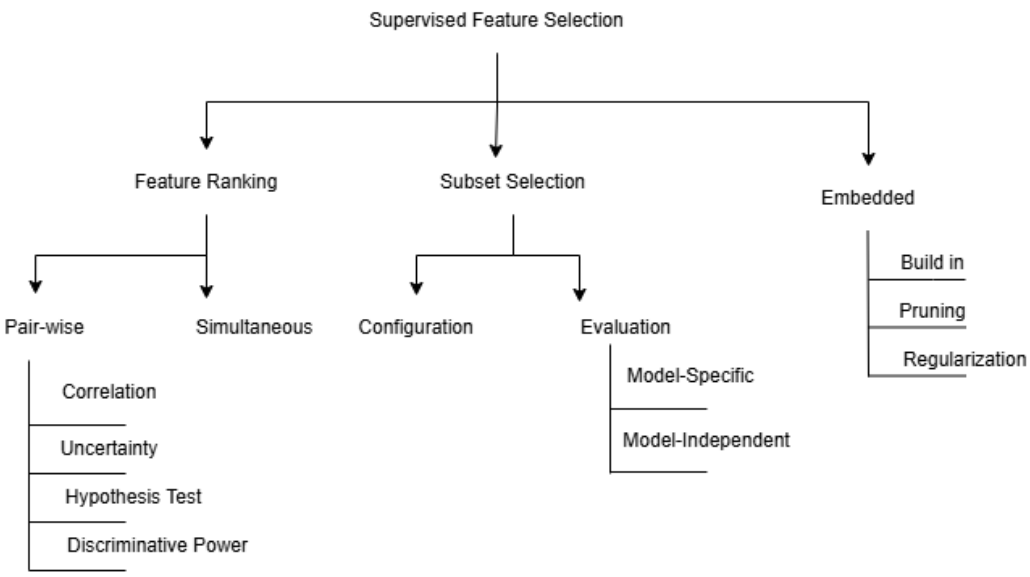

Figure 2.4: New taxonomy which main objective is to classify correctly FS algorithms based on outputs.

- All bats use echolocation to sense distance, and they know the difference between prey/food.
- A bat b_i flies randomly with velocity v_i at position x_i with a fixed frequency f_{min}, varying wavelength λ, and loudness A_0 to search for a prey/food. They can adjust the wavelength of their emitted pulses and the rate of pulse emission on behalf of the proximity of a target.

These behaviors are formally expressed in Eqs. 2.3 to 2.5, as defined below.

$$f_i = f_{min} + (f_{max} - f_{min})\beta \tag{2.3}$$

$$v_j^i(t) = v_i^j(t-1) + [\hat{x}^j - \hat{x}_i^j(t-1)]f_i \tag{2.4}$$

$$x_i^j(t) = x_i^j(t-1) + v_j^i(t) \tag{2.5}$$

This algorithm belongs to the multi-objective optimization category. This means that to get the best possible solution for a problem, the algorithm needs to optimize a set of values $M_i \in i = 0, 1, 2, \ldots, N$; therefore, one of the most important elements in the design of each microbat is a vector $\vec{v}$ that will hold each value to be optimized.

The Eq. 2.3 represents the frequency on which each microbat will emit echolocation, β is a randomly generated number between the range $[0, 1]$; then, the result from Eq. 2.3 controls the pace and range of the movement of the bats and the variation in the wavelengths' frequency that each bat is going to have at each iteration. $x_i^j(t)$ represents the best partial

solution for microbat x_i. $\hat{x}^j$ is the maximum solution globally found so far among the swarm of microbats. Eq. 2.4 adjusts the velocity of each microbat to reduce the distance to the best possible solution found on each v_i for each microbat. Eq. 2.5 just updates the new position with the velocity calculated in Eq. 2.5. An additional step was proposed to improve the variability in each microbat x_i^j consisting of including random walks depicted in Eq. 2.6.

$$x_{new} = x_{old} + \epsilon \bar{A(t)} \tag{2.6}$$

where $\bar{A(t)}$ stands for the average loudness of all bats at iteration t and ϵ it is a random number between the range $[1, -1]$. The authors in [43] developed a Binary Bat algorithm variation for feature selection that aims to produce an optimum subset, resulting from applying Eqs. 2.3 and 2.5 to each $\{b_i : i = 0, 1, 2, \ldots, N\}$ where N represents the features available. However, the resultant i did not fall within the domain of $\{0, 1\}$. Consequently, Eqs. 2.7 and 2.8 were employed to transform each i within the interval of $[0, 1]$; this transformation was subsequently evaluated against a threshold σ to obtain a $\{0, 1\}$. In other words, a feature is selected or not selected.

$$S(v_j^i) = \frac{1}{1 + e^{-v_j^i}} \tag{2.7}$$

$$S(v_j^i) = \begin{cases} 1, & \text{if } S(v_j^i) > \delta. \\ 0, & \text{otherwise.} \end{cases} \tag{2.8}$$

The selected features are systematically evaluated using a designated learning algorithm. Subsequently, the fitness criterion is defined as selecting the optimal feature subset that maximizes classification accuracy across all bats in T iterations. In addition, the learning algorithm used was the optimum path tree forest, as elaborated in [44]. Lastly, Algorithm 3 details the binary bat procedure.

From the Algorithm 3, it can be seen that Equation 2.9 and 2.10 are used to control the pulse rate and the loudness at each iteration.

$$r_0(t + 1) = r_0(t) + [1 - e^{\gamma t}] \tag{2.9}$$

$$A_i(t + 1) = \alpha A_i(t) \tag{2.10}$$

where α and γ are constants purposely added and initially initialized between the range $[1, 2]$ for loudness and $[0, 1]$ for the pulse ratio.

Algorithm 3 Binary Bat Algorithm

for $b_i(\forall i = 0, 1, 2 \ldots, m)$ **do**
  for $j(\forall j = 0, 1, 2 \ldots, n)$ **do**
    $x_i^j \leftarrow Random(0, 1)$
  end for
  $v_j^i(t) \leftarrow 0$
  $A_i \leftarrow [1, 2]$
  $r_0 \leftarrow [0, 1]$
end for
for $t(\forall t = 1, 2, \ldots, T)$ **do**
  for $b_i(\forall i = 0, 1, 2 \ldots, m)$ **do**
    Create Z_1' and Z_2' from Z_1 and Z_1 such that both contain only wavelengths on b_i in which $x_i^j \neq 0$, $\forall i = 0, 1, 2, \ldots, n$
    Create model with Z_1' and test over Z_2', then get accuracy and store it on acc
    $rand \leftarrow Random[0, 1]$
    if $rand > A_0$ and $acc > fit_i$ **then**
      $fit_i \leftarrow acc$
      $A_i \leftarrow \alpha A_i$
      $r_i \leftarrow r_i^0[1 - e^{\gamma t}]$
    end if
  end for
  $[maxfit, maxindex] \leftarrow max(fit)$
  if $maxfit > globalfit$ **then**
    $globalfit \leftarrow maxfit$
    for $b_{maxfit}(\forall j = 0, 1, 2, \ldots, n)$ **do**
      $x^j \leftarrow x_{maxfit}^j$
    end for
  end if

Algorithm 4 Binary Bat Algorithm PT 2

for $b_i(\forall i = 0, 1, 2 \ldots, m)$ **do**
  $\beta \leftarrow Random[0, 1]$
  $rand \leftarrow Random[0, 1]$
  if $rand > r_i$ **then**
    for $j(\forall j = 0, 1, 2 \ldots, n)$ **do**
      $x_i^j \leftarrow x_i^j + \epsilon \bar{A}$
      $\sigma \leftarrow Rando[0, 1]$
      if $\sigma < 1(\frac{1}{1+e^{-x_i^j}})$ **then**
        $-x_i^j \leftarrow 1$
      else
        $-x_i^j \leftarrow 0$
      end if
    end for
  end if
  $rand \leftarrow Random[0, 1]$
  if $rand > A_i$ and $fit_i < globalfit$ **then**
    for $j(\forall j = 0, 1, 2 \ldots, n)$ **do**
      $f_i \leftarrow f_{min} + (f_{max} - f_{min})\beta$
      $v_j^i \leftarrow v_i^j + [\hat{x}^j - \hat{x}_i^j]f_i$
      $\sigma \leftarrow Rando[0, 1]$
      if $\sigma < 1(\frac{1}{1+e^{-v_j^i}})$ **then**
        $-x_i^j \leftarrow 1$
      else
        $-x_i^j \leftarrow 0$
      end if
    end for
  end if
end for
end for
return x^j

3. STATE OF THE ART

***Overview:** ICAFS/ICAFS represents the integration of two significant concepts. In this section, we will examine some critical related work pertinent to Covering Arrays and Feature Selection.*

3.1 Application of Covering Arrays and the introduction in Feature Selection

A clear application of covering arrays used to model parameter interaction is thoroughly examined in [45], which details the construction of static malware detection ML models to analyze malicious files. This study utilizes covering arrays as a means to shape interactions between different ML hyper-parameters' learning algorithms to discover optimum ML models, serving as an alternative to Grid Search, which applies exhaustive search to get the most powerful combination of hyper-parameters. This technique allowed the authors to explore interactions among 2, 3, and 4 classification algorithms' hyper-parameters efficiently without extending the computational time, as it would have resulted from Grid Search.

But the emphasis of covering arrays is fully advocated toward the development of optimal methodologies for producing accurate covering arrays rather than on the application field. Such a fact is evident in recent ***state-of-the-art***; to illustrate this, in [46], authors combined meta-heuristic approaches, such as particle swarm Optimization (PSO) and simulated annealing (SA), to create minor perturbations and enable PSO to escape local optima, thereby achieving an enhanced solution for generated covering arrays. Similarly, in [47], **APPTS** was introduced; a Tabu Search approach featuring an adaptive penalty system that imposes penalties for attempting disallowed settings of combinations in covering arrays.

Although covering arrays have been the subject of study for many years, the integration to Feature Selection is not as comprehensive as might be expected. In [48], an orthogonal array (OA) was embedded for the evaluation of candidate genes within a genetic algorithm to select the optimal subsets of genes for the classification of cancer. It is crucial to distinguish between an orthogonal array and covering arrays, with covering arrays being a flexible structure allowing for the inclusion of every possible combination of parameter values at least once, whereas orthogonal array permits only one. Subsequently, Vivas, Sebastián [13] introduced an innovative Random Forest Classifier employing a covering arrays, which was utilized to identify distinct feature interactions for the purpose of guiding the construction of various decision tree classifiers. In contrast, Dorado, Hugo [14] provided a detailed account of a feature selection wrapper method that leverages a covering arrays to model interactions between various parameters. For each test, this method generates a model and evaluates it, ultimately identifying the set of features that best represent the problem based on the model's performanceF_1-score. Then, Castro, De-la-Torre, Avila-George, *et al.* [8] adapted this methodology to the chemometrics domain, by collecting seeds of various cacao species, which were then processed using specialized spectrometers to obtain samples across different wavelengths on the near-infrared (NIR) spectrum. Subsequently, all observations were preprocessed through CAFS, a similar approach leveraging covering arrays for feature selection, but with the inclusion of additional steps that reshape the covering array to the length of optimally selected wavelengths during T iterations. This caused to progressively halve the feature space at each

$i \in T$, demonstrating a remarkable correlation between reduced subsets and increasing F_1-scores.

3.2 Feature Selection; a well known studied approach

Feature selection constitutes an integral component of the preprocessing stage on ML and deep learning (DL). It involves the identification of the optimal subset of features that most accurately encapsulates the problem under consideration. Section 2.2 detailed each specific sub-specialization for feature selection based on the output. Conversely, this section will focus on discussing noteworthy algorithms pertaining to the aforementioned categories.

- **Embeded Methods**: Liu, Zhou, and Liu [49] came up with an efficient decision tree algorithm based on Gini Index as a split criterion in order to create efficient classification trees. Gini [50] and Ceriani and Verme [51] was mainly used to measure the impurity of a model. The main contribution in feature selection consist as a first step in measure the Gini Index if the dataset were to split on feature X_i. After sounding out all possible X_i features, it chooses the best feature that has the lowest Gini Index possible, because the lowest Gini Index, the less the model is impure. This feature is later used as the root of the tree; the children of the tree are recursively constructed with the before-mentioned steps along with its respective portion dataset. The final output results on a Classification Tree with the most relevant features. Moreover, it became too attached to the definition of Embedded feature selection for the merely reason that the feature selection happened always at the training phase.

- **Subset Evaluation Methods-Model Specific**: Pham, Ghanbarzadeh, Koç, *et al.* proposed a variation of a Bee Algorithm for feature selection, deeply studied in [52]. The overall heuristic of BA, consist on creating random population of bees, assigning each one to neighborhoods made of random solutions (search spaces), the bees with the highest fit got recruited. The process is repeated until no bees are left for allocation; the sites or neighborhoods with highest score are returned; aslo called elite sites. Guha, Chatterjee, Khalid Hassan, *et al.* [12], on the other side, proposed a compilation of well known meta-heuristics methods into PyFS, an open source python feature selection algorithms. Methods as BBA and more meta-heuristics based algorithms, can be found into this library [43].

- **Subset Evaluation Methods-Model Independent**: Anitha and Sherly [15] presented MDIMFR algorithm, a mutual information (MI) based approach for subset selection, considering three key factors: redundancy, relevance [53] and feature interaction. In the first step it creates a set S_{rel} containing the features with the maximum MI between the input features and target classes, with the elements present it S_{rel}, it calculates the interacting features between each pair by calculating the

positive difference between the unconditional and conditional MI; the features that met this criteria should be included on the S_{int}. Then, another set S_{red} is created to group the redundant features by calculating the differences between the unconditional dependence and the conditional dependence with respect to the target class between any pair of nodes. As a last step the algorithm output the final set S by removing al elements from S_{red} on S_{int}.Moreover, others influential authors have been researching feature selection algorithms around Mutal Information [54], [55].

- **Simultaneous Ranking**: In [56], authors presented a graph approach like algorithm with Page Rank relaying on the statement that the covariance between two features and labels should be as small as possible. So, as a first step, it randomly deletes label samples in order to convert the data partially present; after this, it calculates the covariance correlation distance between the x features and the y labels. The result of the previous computation is stored on a table. The previous step is repeated to now compute the correlation covariance distance between two features. As a third step, it generates the mean of the already computed correlation covariance distance table for every features or column and map it to the its already computed label covariance correlation distance on an third matrix. Then, it computes and adjacency matrix with the distance for every two feature based on the mean correlation covariance distance for every two nodes. Finally it creates the complete graph and apply Page Rank to get the ranking of the most relevant features.

It is important to emphasize that Sklearn also implements well-tested algorithms for feature selection, as evidenced by Table 3.1.

Algorithm	**Description**
SelectKBest	Select features according to the k highest scores.
SelectPercentile	Select features according to a percentile of the highest scores.
RFE (Recursive Feature Elimination)	Recursively eliminate features and build a model.
RFECV	Recursive feature elimination with cross-validation for optimal number of features.
SelectFromModel	Select features based on importance weights.

Table 3.1: **Scikit-learn Feature Selection Algorithms**

3.3 Covering Array Feature Selection

CAFS is an algorithm proposed to tackle the problem of high dimensionality on a dataset with more than a thousand wavelengths. This algorithm was designed to be coupled to the Naive Bayes classifier and assumes that the samples follow a normal distribution in the feature space and each feature is statistically independent from each other.

One requirement for this algorithm to work is a covering array with the following properties: CAN(k,t,v). The number of k rows is equal to the number of wavelengths, and the

alphabet $\mathbb{Z}_v$ must be binary. The rationale for this design is that the model is required to have an **interaction** between every t wavelength that is available on the feature space, and each value on $\mathbb{Z}_v$ means whether a feature will be selected or not (*e.g.* if $\mathbb{Z}_{\gtrapprox} = 1$, the feature is selected, otherwise discarded).

Algorithm 5 CAFS(CA, X, y)

```
f_max ← X : {X_i | i = 0, 1, 2..., n}
for j(∀j = 0, 1, 2..., T) do
    x̂' = 0
    f'_max = {}
    for k in (∀k = 0, 1, 2..., R) do
        for l in (∀l = 0, 1, 2..., C) do
            if CA(k, l) = 1 then
                f' ∪ {f^l_max}
            end if
        end for
        X' ← X_i : i ∈ f'
        x̄ ← 5crossvalidation(X', y)
        if x̄' ≥ x̂' then
            x̂' ← x̄
            f'_max ← f'
        end if
    end for
    C = |f_max|
    x̂_j = x̂'
    f_max = f'_max
end for
return (x̂, f_max)
```

This algorithm starts by setting f_{max} with the current labels present on X. Secondly, for each k within the current l, the covering array is examined to determine whether wavelength $(k, l) = 1$, inquiring the selection of the l wavelength on f_{max} , which is then appended to f'. Later, a subset X' is generated, containing candidates all wavelengths available on f'. Next, X' is evaluated against a 5-cross-validation function. Then, the result is assigned to $\bar{x}$, which represents the mean across the 5 folds. Following this, $\bar{x}$ is compared against $\hat{x}'$ to validate if the current sub-selection of bands is greater. Finally, if it turns out to be true, it will assign f' to f_{max}.

The cardinality of f_{max} is assigned to C, the column counts; this step is critical as it allows for the adaptation of the covering array to align with the elements chosen in f_{max}. This procedure is reiterated to further reduce the feature space. Adjusting the columns to correspond to the size of the newly selected wavelength subsets is essential, thereby reflecting **wavlengths reduction**, as depicted in Figure 3.1. It is important to emphasize that this reduction process is inherent to covering arrays, indicating that each test within the covering arrays will comprise approximately half of the parameters activated with

a value of 1. Consequently, when CAFS identifies a significant subset and adjusts the covering arrays accordingly, it effectively reduces the wavelengths.

$$\text{a)}\begin{pmatrix}0&0&0&0&0&0&0\\0&0&0&1&1&1&1\\0&1&1&0&0&1&1\\0&1&1&1&1&0&0\\1&0&1&0&1&0&1\\1&0&1&1&0&1&0\\1&1&0&0&1&1&0\\1&1&0&1&0&0&1\end{pmatrix}\quad\text{b)}\begin{pmatrix}0&0&0&0\\0&0&0&1\\0&1&1&0\\0&1&1&1\\1&0&1&0\\1&0&1&1\\1&1&0&0\\1&1&0&1\end{pmatrix}\quad\text{c)}\begin{pmatrix}0&0\\0&0\\0&1\\0&1\\1&0\\1&0\\1&1\\1&1\end{pmatrix}$$

Figure 3.1: Iterative shrinking of the covering array based on the best subset of wavelengths selected in each iteration. a) **Supose best set of wavelengths is found on the second row with a subset of 4 wavelengths for first iteration; b) in second iteration, the CA is shrinked to match now those 4 selected wavelengths. Next, it is assumed that the best subset of wavelengths is located at row three; then, for the third iteration c) it again shrink the array to match the previously selected wavelengths**

3.4 Research Guidance

Upon revisiting these two concepts, it is imperative to underscore that this degree obtention work does not intend to introduce a generic feature selection algorithm. Rather, it seeks to extend CAFS, proposing a *state-of-the-art* ICAFS algorithm specifically tailored for wavelength selection within the field of chemometrics. Further research is necessary to expand these methods into new domains.

4. An Iterative Covering Array Feature Selection Algorithm

***Overview:** In this section, we introduce ICAFS, an innovative expansion of covering array Feature Selection designed for the purpose of wavelength selection.*

4.1 Iterative Covering Array Feature Selection

Algorithm 5 has been empirically validated and exhibits robust performance in classifying cacao nibs. However, a notable limitation remains: it inadequately captures the interactions among selected wavelengths within the covering array. This is due to its strategy of utilizing a consistently halved CA to fit the number of chosen wavelengths. Consequently, even though the coverage property is maintained, CAFS does not construct a test matrix for the chosen wavelengths and, as a result, it maintains the interaction order of the original wavelengths with a larger number of test cases. To address this deficiency, we introduce ICAFS. This algorithm is closely related to Algorithm 5, with the primary distinction being that it generates a new array based on the selected wavelengths to accurately simulate interactions among them.

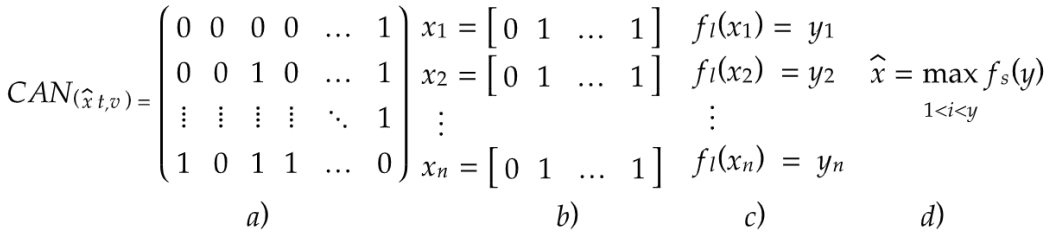

Figure 4.1: **Main framework for wavelength selection in ICAFS *a)* Create CAN from the best possible subset; initially the best subset is n. *b)* from each test, *c)* we construct a model and *c)* evaluate it. Finally, *d)* select de best subset and repeat until T iterations are reached**

The algorithm used to create covering arrays is IPOG, and it is important to mention that since IPOG is deterministic, it produces the same covering arrays. However, it stands out among other construction methods due to its greedy strategy, avoiding exhaustive search even though the number of tests resulting is greater compared to other methods, and the ability to increase t, allowing the generation of new wavelength interactions. For this reason, when γ randomness seed is activated, the candidates' feature subsets undergo a shuffle to induce variability on subsets generated before constructing covering arrays. ICAFS is described in Algorithm 6. Moreover, in Figure 4.1 it can be appreciated the strategy followed. Another important piece for ICAFS is the use of classification algorithms available on sklearn; Table 4.1 lists a few common classifiers.

This algorithm first creates p with a list of ordered pairs $(i, \{1, 0\})$ such that each i is a prospect wavelength from f_{max} assigned with a $\mathbb{Z}_v = \{0, 1\}$. Secondly, a new covering array is built from p and strength t using IPOG. Then, for each l on the current row k, within the covering array, it selects the current wavelength if the (k, l) wavelength

Algorithm	Description
Logistic Regression	Linear model for classification utilizing a logistic function
Support Vector Machine	Classifies data by finding the best hyperplane
Decision Tree	A non-parametric model that splits the data into subsets
Random Forest	Ensemble method using multiple decision trees
Naive Bayes	Probabilistic classifiers based on Bayes' theorem
K-Nearest Neighbors	Classifies based on the majority class of nearest neighbors

Table 4.1: **Supervised classification algorithms in scikit-learn**

Algorithm 6 ICAFS(X, y ,t,γ ,T,)

$f_{max} \leftarrow X : \{X_i \mid i = 0, 1, 2 \ldots, n\}$
for $j(\forall j = 0, 1, 2 \ldots, T)$ **do**
    $\hat{x}' = 0$
    if γ activated **then**
        shuffle f_{max} that contains the best subset of wavelengths base on γ seed
    end if
    $p = \{(i, \{1, 0\}) \mid i \in f_{max}\}$
    CA $\leftarrow$ IPOG(t, p)
    for k in $(\forall k = 0, 1, 2 \ldots, R)$ **do**
        for l in $(\forall l = 0, 1, 2 \ldots, C)$ **do**
            if CA$(k, l) = 1$ **then**
                $f' \cup \{f^l_{max}\}$
            end if
        end for
        $X' \leftarrow X_i : i \in f'$
        $\bar{x} \leftarrow 5crossvalidation(X', y)$
        if $\bar{x} \geq \hat{x}'$ **then**
            $\hat{x}' \leftarrow \bar{x}$
            $f_{max} \leftarrow f'$
        end if
    end for
    $\hat{x}_j = \hat{x}'$
end for
return $(\hat{x}, f_{max})$

on covering arrays is 1, appending it to f'; otherwise, it is ignored. With the selected wavelengths, it creates X' to later evaluate it using 5-fold cross-validation. The result is assigned to $\bar{x}$. Should $\bar{x}$ be greater than $\hat{x}$, f_{max} is replaced with f', and this process is reiterated until either there are no further wavelengths to diminish or the maximum permissible number of iterations has been achieved. It is crucial to note that the reduction procedure executed during each iteration is implicitly applied due to the process of constructing covering arrays repeatedly with the best subset of features selected from the previous iteration. Such reduction strategy is described in Figure 3.1.

4.2 ICAFS Feature Selection Objective and Category

Although in Section 2.2 a concise purpose for feature selection was given, Saeys, Yvan et al. [57] expanded the definition to be a manifold objective. Such objectives are (a) finding a subset with the highest accuracy possible, (b) selecting a feature subset that does not degrade the performance of the system [58], (c) finding the relevant features that best represent the data, and (d) discarding irrelevant or redundant features.

The multitude of these objectives arises from the fact that each category encompasses certain aims; for instance, **Filter Methods** proves more applicable when relevance [53], [59] is a crucial factor in the application under consideration. This indicates that while the selected variables may have low predictive scores for a critical model to predict an important disease, their selection holds greater significance than merely achieving the highest accuracy subset. Conversely, **wrapper method** prioritizes subsets that attain high scores, employing k-cross validation at each subset evaluation to ensure that the selected features are not attributable to chance, thereby disregarding relevance between features selected.

Consequently, the applicability of each category is dictated by the specific challenge it seeks to address. Accordingly, the primary objective of Iterative Covering Array Feature Selection is the implementation of a wrapper strategy tailored for high-dimensional chemometric applications, an objective strongly supported in [11], who emphasized the challenges associated with the redundant, overlapped, and high-dimensional data derived from chemometric measurements.

Additionally, from Fig. 2.4, we can correctly categorize ICAFS/CAFS as follows: each test derived from a covering array signifies an interaction between features, selecting the subset selection category. Next, each feature subset is assessed by developing a model and choosing the maximum model score in each iteration. Hence, we can go deeper into the taxonomy and classify it under the model-specific subcategory. Overall, ICAFS represents a wrapper method for subset selection based on model-specific evaluation.

5. Experimental Methodology

Overview:In this chapter, the steps and methodology proposed to examine the ICAFS will be meticulously outlined.

5.1 Methodology

To assess the effectiveness of ICAFS in wavelength selection, we will tackle six high-dimensional chemometric problems. Each dataset will subsequently be analyzed using ICAFS, CAFS, and BBA, with associated hyper-parameters explored in this section. The resultant findings will be presented, leading to the conclusion of the identified outcomes. This methodology is depicted in Fig. 5.1, and each phase is thoroughly explained in the following subsections. It's important to emphasize that throughout all the experimentation conducted, γ was deactivated in ICAFS in order to induce reproducibility.

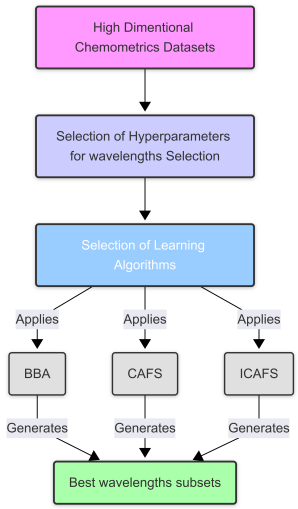

Figure 5.1: **Experimental methodology to compare best subsets of wavelengths across different FS strategies such as BBA, CAFS and ICAFS**

5.2 Chemometrics dataset selection

These datasets constitute a fundamental basis to substantiate the validity of ICAFS in the field of chemometrics. Furthermore, they pertain to classification tasks, with their properties delineated in Table 5.1.

The Cacao Nibs were extracted and processed using methodologies explained in [8]. However, samples from Non-Plus/Plus Algarrobo trees [60] were generated by taking hyperspectral photos of Algarrobo trees that belonged to two main species, Non-Plus/Plus; each sample was processed with specialized hyperspectral tools and techniques, transforming the images into indices.

The "Fresh Meats" dataset [61] was created by purchasing 20 distinct meat samples from chicken, pork, and turkey. Each sample was minced and subsequently divided into four portions, with three portions frozen at varying temperatures and one kept refrigerated.

Dataset	No. wavelengths	wavelengths	Descriptions	No. Samples
Circle In Square	10	2 relevant wavelengths and 8 noisy wavelengths in acending order by noisy perecentage	Data that explain if a point is inside or outside a cricle	10000
Non-plus/plus Algarrobo	24	24 hypersepctral wavelength including R,G,B	Non Plus / Plus *Neltuma pallida* trees samples	1000
Fruit Puree	235	Spectral Range: 899.327-1802.564 cm-1	Strawberry Puree sample explaning whether is adulterated or not.	983
Fresh Meats	448	Spectral Range:1005.3495-1867.864 cm-1	Describes 3 different meat gotten from Pork, Turkey and Chciken	120
Olive Oils Quadrum	570	Spectral Range: 798.892-1896.8085 cm-1	Olive oils from different eurupe regions	120
Cacao Nibs	1401	NIR Wavelengths from 1100 to 2500	6 different Cacao amazonian species through NIR wavelengths	990

Table 5.1: **High dimensional chemometrics datasets mainly focused on food with its respective number of bands and description**

These samples were then analyzed using an on-site fourier-transform infrared (FTIR) spectrometer system.

The "Fruit Purée" dataset [62] encompasses a two-class categorization, comprising samples of strawberry and non-strawberry purées. The samples were obtained by processing various fruits on a metal sieve or blending them in a mixer. The non-strawberry samples were composed of fruit purée with adulterants added. Finally, Olive Oils Quadrum [63] centers on classifying European olive oil obtained from different regions such as Greece, Italy, Portugal, and Spain; the data were obtained by processing each sample using a MonitIR FTIR spectrometer system.

The CIS dataset is a synthetic construct designed to examine the classification of points as either inside or outside a circular boundary, utilizing two primary wavelengths. Additionally, the dataset incorporates 8 supplementary wavelengths to enhance its dimensionality and introduce noise into the classification challenges. Figure 5.2 illustrates the visual depiction of samples *inside* and *outside* within the two-dimensional feature space.

5.3 Discovering the best strength and open ICAFS to generic applications

ICAFS will be conducted against a brief experimentation in order to open ICAFS out of the wavelength selection to general classification applications. It is essential to note that

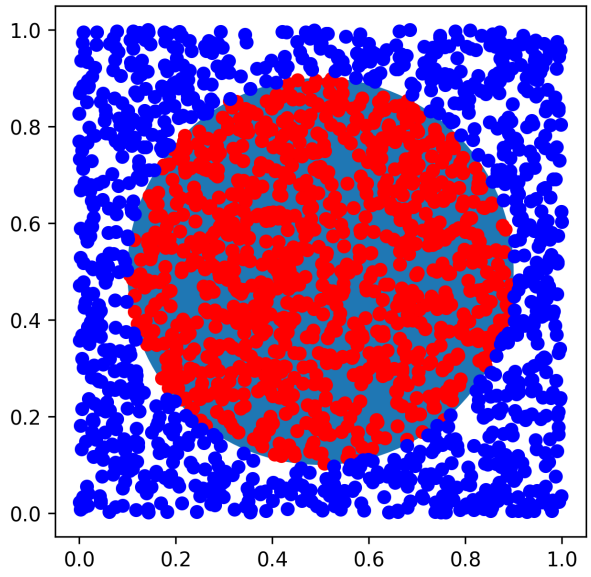

Figure 5.2: **Visual representation of the feature space of the original samples corresponding to the circle-in-the-square synthetic dataset**

the primary objective of ICAFS is to select subsets that achieve the optimal F_1-score rather than simply identifying relevant features. In this experimentation, we will evaluate three datasets available in the UCI repository [64]. Our motivation extends beyond providing evidence that ICAFS is applicable to generalized classification tasks, such as those found within the UCI repository. Moreover, as a secondary goal, we aim to demonstrate the optimal parameter t, which governs the interaction among the various t features within the system; as per Kuhn, D. Richard et al. [21] the optimal interaction for failure detection within the system is described by $t = 6$. Therefore, during this experiment, we will adhere to the same rationale, applied to the **feature interaction**. The methodology we will employ is as follows: We will select three classic classification tasks from the UCI repository, and for each task, we will execute ICAFS against a $t = \{2, 3, 4, 5, 6\}$. Each instance of ICAFS will be configured with $T = 10$. Ultimately, we will present the results accompanied by a brief conclusion. This facet of the work is crucial as it seeks to identify the most effective t concerning feature interaction in generic classification problems. The results will be further discussed in Section 6.2

5.3.1 Three well-known datasets

In this part, we will give a brief explanation of each dataset used, and each of these datasets aligns with the wrapper methods' objective, which consists of getting the subset that attains the highest possible accuracy:

Breast Cancer Wisconsin [65]; authors' main contribution was to propose a new mechanism to detect breast cancer by images as an alternative to conventional methods that

required very critical and hard procedures that led to inaccurate results, such as fine needle aspiration (FNA); therefore, the authors developed a system that uses a computer to analyze images of cell nuclei from an FNA slide, gathering a total of exact measurements, such as radius, area, and more from each sampled cell. These measurements were decomposed into the mean, standard error, and "worst" (largest); worst does not mean the features are not good enough, they are indeed one of the most powerful as they represent the highest or largest measurements obtained from each cell nuclei, resulting in a total of 30 features from 10 initially collected. It is interesting to emphasize that during the model creation for predicting cancer, they use exhaustive search to get features that attained satisfactory results and remark on the use of heuristic-based search as an alternative.

Wine Quality[66]: This dataset consists of a large collection of wines, including 1,599 red and 4,898 white wine samples. The feature space was obtained from physicochemical measurements, which consisted of 11 different laboratory tests measuring things like alcohol content, acidity, residual sugar, pH, and sulfates; and sensory data, scores rated on a scale from 0 (very bad) to 10 (excellent) by a panel of expert human tasters that dictated the quality of wine samples. The main goal was to predict how well a wine was tested on the basis of the features discussed above without testing each observation by creating regression models to approximate sensory scores. But since this study centers on classification, we are going to take as reference the work from Er and Atasoy [67] who used wine quality data for classifying correctly wine types. It is important to mention that even though there exist impactful features to approximate wine scores, on classification this may vary and may not be relevant.

Letter Recognition [68]: It is a collection of 20,000 images generated from 20 different commercial fonts, including script, italic, and Gothic styles, ensuring a wide variety of shapes. Each letter was randomly distorted using transformations such as changing magnification, aspect ratio, and applying horizontal and vertical warps, resulting in a collection of misshapen but human-recognizable characters. Then, from each letter 16 primitive numerical attributes were extracted, consisting of statistical properties of the pixel distribution.

In Table 5.2 is listed a summary of datasets that are going to be used as part of this sub-experimentation.

5.3.2 Preprocessing stage and hyperparameter selection for ICAFS in choosing the best t

Wine Quality, Letter Recognition, and Breast Cancer Wisconsin were passed through a process of Min-Max scaling in order to improve performance in classifiers used. For ICAFS, the selected algorithms will be KNN and SVM; only KNN is configured with a total of neighbors of 3. SVM is used with default hyperparameters. With this configuration in mind, the test will be running ICAFS 6 time starting at $t = 2$ and ending in $t = 6$ for each of the datasets proposed in two variants.

Dataset	No. wavelengths	wavelengths	Descriptions	No. Samples
Wine Quality	11	Physicochemical measurments as alochol, pH and more	classifying red and white wines	6,497
Letter Recognition	11	statistical properties of the pixel distribution	recognizing the 26 letter uppercase letters	20,000
Breast Cancer Wisconsin	30	Breast Cancer Ceil measurments as raidus, area and more.	Classifying Malignus and Benignus cancer	568

Table 5.2: **Brief Summary of the generic dataset to be used**

5.4 Classification Algorithms

To ensure variability in the results obtained, this document will delineate four iterations of ICAFS and CAFS, employing four distinct learning algorithms: MLP, optimum path forest (OPF), kNN, and SVM. The reason for covering-array-based methods to use OPF is to have a fair comparison between the results obtained by BBA and ICAFS / CAFS. Subsequently, a review of the foundational concepts pertaining to the implemented learning algorithms will be provided.

The BBA employs the **OPF** learning algorithm [44]. The OPF algorithm functions as a graph-based forest classification technique that initially involves the computation of a minimum spanning tree (MST) on a fully connected weighted graph. The weights of the edges are determined based on the distances between each pair of samples within the training set. After the construction of the MST, the algorithm identifies the prototype nodes. These prototype nodes are defined as adjacent nodes within the MST that belong to distinct classes. From these prototypes, the algorithm initiates the training phase or competition process by continually determining the minimum path from each prototype to every sample t, ultimately assigning each t to a prototype s such that the path between them is minimal, assigning t with the label $L(s)$. It is called competition because each sample t can be claimed by any s that belongs to a different class as long as the path π_t that is rooted in a new candidate prototype s is minimized, thus changing the label of t to $L(s)$.

SVM or Support Vector Machine [69] is a learning algorithm that is used for classification problems. The main goal consists of correctly classifying a training sample on the correct side of a plane delimited by a decision boundary. The training phase consists of minimizing this decision boundary in order to increase the chance of a correct classification. The decision boundary is delimited and supported by vectors whose loss is greater than 0; meaning that they are points that remain classified incorrectly and correctly but close to the decision boundary. The penalty for each misclassified vector is given by Equation 5.1.

$$\min_{w,b} \left(\frac{1}{2}\|w\|^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i(w \cdot x_i + b)) \right) \tag{5.1}$$

This process facilitates the minimization of $\frac{1}{2}\|w\|^2$ subject to $y_i(w \cdot x_i + b) \geq 1$. Specifically, if a sample incurs a negative penalty, it signifies an incorrect classification. Consequently, this penalty is disseminated to update b and w in order to minimize by maximally enhancing $\|w\|$.

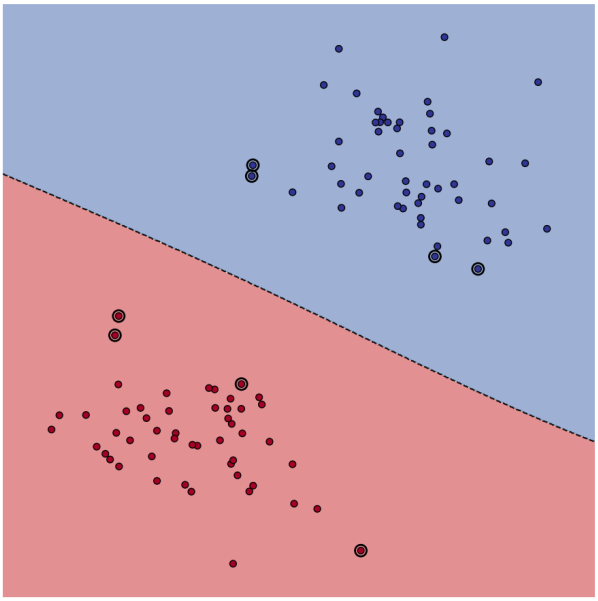

Figure 5.3: **SVM classifier in action, where the dotted line represent the decision boundary, and the highlighted points, the support vectors.**

kNN or k-Nearest Neighbors [70] is a nonparametric learning algorithm; this means that the only hyperparameter to configure is the number of closest neighbors that each training sample will be compared to assign the correct label. During the classification phase, each sample will be compared against all available points, assigning the label with the highest frequency to the incoming sample. The distances used are the Euclidean and Cosine distances. One major drawback of kNN is that it has to keep in memory all the training data for the evaluation phase.

MLP [71] is a network because it can be represented by composing many different linear functions. For instance, a model might consist of three $f^{(1)}, f^{(2)}, f^{(3)}$ connected in a chain to form $f(x) = f^{(3)}(f^{(2)}(f^{(1)}(x)))$. In this chained structure, each function is a layer of the network. The first layer is $f^{(1)}$, the second is $f^{(2)}$, and so on, with the overall length of the chain defining the model's depth. MLPs then are called "feedforward" because information flows directly from the input x, through the intermediate computations that define the function, to the output y. Thus, there are no feedback connections where the outputs of the model are fed back into itself. In addition, the layers between the input

and the output are called hidden layers. In order to induce non-linearity, MLPs apply activation functions to hidden layers. This way, the model learns to identify complex patterns. A common activation function is ReLU $g(z) = \{0, z\}$, with the main purpose of deactivating neurons in order to break linearity. This way, each neuron learns to identify different aspects of the data. Finally, all this knowledge is combined in the output layers for final prediction.

5.5 Performance Evaluation

ICAFS/CAFS will use F_1-macro to evaluate the performance of created models by each of the subsets generated. This decision comes from the fact that F_1-macro deals greatly with **class imbalance** cases by giving equal weight to each class's F_1-score. To illustrate, cacao nibs is a 7-class classification with 998 samples and 1499 features. First and foremost, it's essential to understand fundamental concepts such *precision*, *recall* and F_1-score. To begin with, Table 5.5 shows the intuition where elements $N_{i,j}$ correspond to C_i classified under j-class. Thus, this matrix, commonly called confusion matrix, depicts the classes(*rows*) being classified correctly or incorrectly (columns). Now, Table 5.5 shows the possible values that each cell might take in terms of C_i, where $C_{i,i}$ indicates *positive* (*number of sample correctly classified*) values and the rest to *false*.

		Predicted class (C_j)				
		$\hat{C}_1$	...	$\hat{C}_i$	...	$\hat{C}_n$
Actual class (C_i)	C_1	$N_{1,1}$	...	$N_{1,j}$	...	$N_{1,n}$
	⋮	⋮	⋮	⋮	⋮	⋮
	C_i	$N_{i,1}$	...	$N_{i,j}$	...	$N_{i,n}$
	⋮	⋮	⋮	⋮	⋮	⋮
	C_n	$N_{n,1}$	...	$N_{n,j}$	...	$N_{n,n}$

Table 5.3: **Confusion Matrix intuition where diagonal represents a correct classification**

The following listing will elucidate in detail each possible value from Table 5.5.

1. $\mathbf{TP}_i$ (True Positive): The amounts of samples in class i correctly classified as positives (*i.e.*, i classified. as i). See 5.2.

$$TP_i = N_{ii} \tag{5.2}$$

2. $\mathbf{TN}_i$(True Negative): Samples from class not i correctly classified as negatives (*i.e.*,

		Predicted class (C_j)				
		$\hat{C}_1$	$\dots$	$\hat{C}_i$	$\dots$	$\hat{C}_n$
Actual class (C_i)	C_1	TN_i	TN_i	$\mathbf{FP}_i$	TN_i	TN_i
	$\vdots$	TN_i	TN_i	$\mathbf{FP}_i$	TN_i	TN_i
	C_i	$\mathbf{FN}_i$	$\mathbf{FN}_i$	$\mathbf{TP}_i$	$\mathbf{FN}_i$	$\mathbf{FN}_i$
	$\vdots$	TN_i	TN_i	$\mathbf{FP}_i$	TN_i	TN_i
	C_n	TN_i	TN_i	$\mathbf{FP}_i$	TN_i	TN_i

Table 5.4: **Confusion Matrix where C_i is correctly classified**

correctly classified as not i). See 5.3.

$$TN_i = \sum_{j,k \neq i} N_{jk} = \sum_{j=1}^{n} \sum_{k=1}^{n} N_{jk} - (TP_i + FP_i + FN_i). \tag{5.3}$$

3. $\mathbf{FP}_i$ (False Positive): Samples from class not-i wrongly classified as positives (i.e., incorrectly classified as class i). See 5.4.

$$FP_i = \sum_{k \neq i} N_{ki} = \sum_{k=1}^{n} N_{ki} - \mathrm{TP}_i. \tag{5.4}$$

4. $\mathbf{FN}_i$ (False Negative): Samples from class i wrongly classified as negatives (i.e., classified as any class except class i). See 5.5

$$FN_i = \sum_{k \neq i} N_{ik} = \sum_{k=1}^{n} N_{ik} - \mathrm{TP}_i \tag{5.5}$$

In [72], *accuracy*, *recall*, *recall* and F - *measures* are defined accordingly for multiclass applications. For instance, *accuracy* can be defined as the number of correct classifications, as clearly explained in Eq 5.6. While *precision* refers to sample C_i classified correctly with respect to the samples that got predicted as positives by classifiers, such behavior is shown in Eq 5.7. On the other hand, *recall* is a measure of correctly selecting instances of the target class C_i related to the positive samples and formally defined in Eq 5.8. As a result, f - measure is the harmonic mean of the precision and recall, shown in Eq 5.9.

$$\text{Accuracy} = \frac{\sum_{i=1}^{n} \mathrm{TP}_i}{\sum_{i=1}^{n} \sum_{j=1}^{n} N_{ij}} \tag{5.6}$$

$$\text{Precision}_i = \frac{\mathrm{TP}_i}{TP + \mathrm{FP}_i} \tag{5.7}$$

$$\text{Recall}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FN}_i} \tag{5.8}$$

$$\text{F-Measure}_i = 2 \times \frac{\text{Precision}_i \times \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i} \tag{5.9}$$

Finally, the F_1-macro is the unweighted average of the F_1-Scores for each individual class defined in Eq 5.10.

$$F_1 - \text{Macro} = \frac{1}{N} \sum_{i=1}^{N} {F_1}^i \tag{5.10}$$

5.6 Hyperparameter selection

The selection of hyper-parameters for each of the wavelength selection algorithms is detailed in the subsequent listing. ICAFS and CAFS employ SVM with *C = 1*,an *rbf* kernel and a *ovr* decision function shape. For ICAFS kNN, in Algarrobo, Fresh Meats, CIS, Pureé fruits, the number of neighbors is set to 3 and 2 neighbors for Cacao Nibs and Olive Oils; whereas for CAFS kNN, Algarrobo is only experimented with 3, and the other tasks with 2; that is to say, a short experimentation was conducted between 2 and 3 neighbors, for the main reason of not having a feature selection bias toward hyper-parameter optimization. In the case of OPF, the existing implementation available in [73] is utilized, specifically focusing on the **SupervisedOPF** class and among many hyper-parameters available, only the distance metric is set to Log Squared Euclidean. The MLP is configured to operate with stochastic gradient descent and a maximum of 7000 iterations.

1. **BBA**: The best hyper-parameters that showed good performance with respect to the selected wavelengths and accuracy for all task were the following: $f_{min} = 1$, $f_{max} = 15$, $A = 1.0$, $r = 0.7$, $\alpha = 0.95$, $\gamma = 0.6$.
2. **CAFS**: Only a CA that matches the number of existing wavelengths for each of the datasets.
3. **ICAFS**: CIS were configured using **strength** $t = 3$ for the construction of covering arrays. Non-plus/plus Algarrobo, Cacao nibs, Fruit Purée, Fresh Meats and Olive Oils were utilized with $t = 2$. The alphabet specific to each covering array is $Z_v = \{0, 1\}$. Given the vast array of wavelengths associated with Cacao Nibs, a practical level of interaction was selected for each column $t = 2$ to avoid exponential increases in computational time due to the complexity of IPOG as discussed in [33].

5.7 Umap visualization

Datasets presented in Section 5.2 show high dimensionality, meaning the number of features is really high for each application. As an example, non-plus/plus algarrobo , cacao nibs, fresh meats, and fruit purée exhibit such properties as listed in Table 5.1. Then, a visualization method is strictly needed in order to compare features selected from ICAFS/CAFS/BBA against the complete feature space for each task. For this reason, UMAP[74], a dimensionality reduction algorithm, was chosen. UMAP's main objective is to convert high dimensional problems into typically 2 or 3 dimensions to get a space that accurately represents data points connected and clustered together. Hence, main methodology for constructing such a reduced space is constituted from two main phases. First, is to construct a weighted graph from the original data to reveal how points are close to each other. Second, during an iterative process, it pulls connected points and pushes away unconnected points until it can resemble a graph that looks as much as possible like the original one.

For the first phase, it assumes the data lies on a hidden, curved surface, called a manifold, assuming each sample is evenly spread out. For each point, it finds its k nearest neighbors and the distance to these neighbors is adjusted so that dense and sparse areas of data are treated equally. Secondly, it calculates the strength of the connection from point i to its neighbor j. But sometimes, calculating the strength bidirectionally won't result in the same values. Hence, UMAP will combine them to get a fair strength. Once the symmetrization process successfully balanced strengths, as the second phase, it starts arranging(pull and push) points in a low dimension by iteratively measuring cross-entropy against the original graph, in order to construct a graph in lower dimension very likely to the complete.

6. RESULTS AND DISCUSSION

Overview: In this chapter, a comprehensive discussion of the obtained results will be presented.

6.1 CAFS Results

Table 6.2 presents the wavelengths resulting from CAFS KNN, CAFS SVM, CAFS OPF, and CAFS MLP. The wavelengths selected by the KNN variant demonstrated superior performance compared to the ones on SVM, MLP, and OPF; however, these results can be attributed to the absence of comprehensive hyperparameter optimization for each implemented learning algorithm employed. It is advisable to conduct a thorough analysis to identify the optimal set of hyper-parameters to enhance the F_1 score for each of the proposed CAFS variations.

Figure 6.4.*a* illustrates the number of optimal wavelengths selected in each iteration along with their corresponding scores for the Cacao Nib problem. In this case, a smaller subset of selected wavelengths resulted in higher accuracy. Conversely, as depicted in Figure 6.4.*b*, the non-plus/plus Algarrobo exhibited an opposing trend; a reduction in wavelengths led to a lower F_1-score. The UMAP projection in Figure 6.1 indicates that the two classes are in close proximity to each other; hence, it can be inferred that CAFS eliminated significant wavelengths throughout the 10 iterations for non-plus/plus algarrobo.

Additionally, CAFS MLP showed a similar pattern; however, as the wavelengths were reduced by half in each iteration, the score was markedly affected, as observed in Figure 6.4.*c*. Furthermore, a considerable amount of time was required to achieve the results previously noted.

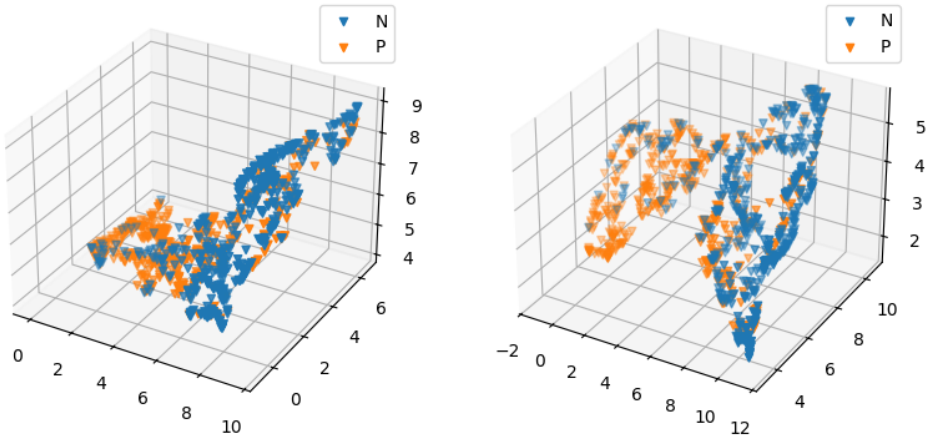

Figure 6.1: **UMAP Representation for Algarrobo dataset with CAFS KNN**

6.2 ICAFS Results

Table 6.2 displays the results for all ICAFS versions. KNN version surpassed the SVM implementation in terms of F_1-score; however, the SVM utilized fewer wavelengths while

still maintaining an acceptable F_1-score. In addition, a similar pattern was observed with ICAFS OPF, albeit only in the cases of Fresh Meat, Fruit Purée, and Olive Oilis Quadrum. Figure 6.5.*c* illustrates a clear correlation between feature reduction and improved scoring for ICAFS OPF in the Cacao task.

In Figure 6.5.*a*, ICAFS effectively selected 5 wavelengths from a total of 1401 across 9 iterations, achieving commendable performance with the feature space being reduced by half in each cycle. However, during the final iteration, the F_1-score experienced a significant decline attributable to the reduction of wavelengths. This underscores two insights: that an increased number of iterations does not necessarily enhance performance and that removing wavelengths may lead to the loss of critical information about the chemometric systems, thereby degrading performance. Figure 6.5.*d* corroborates the aforementioned statement. Figure 6.2 depicts two UMAP representations of the Algarrabo dataset, where on the left it is shown a UMAP plot with all available wavelengths, while on the right, it displays its UMAP representation with wavelengths selected through ICAFS KNN and stated on Table 6.2. This observation stands in contrast to CAFS KNN algarrobo as illustrated in Figure 6.1, owing to the fact that CAFS KNN selected a total of six wavelengths.

The two plots clearly demonstrate that the wavelength subselection significantly influenced the structure of the data, thereby explaining the previously discussed reduction in accuracy, as it caused the classes to become more proximate. However, the performance achieved with only three wavelengths is deemed quite satisfactory.

Conversely, the Multilayer Perceptron (MLP) demonstrated incredible performance across all presented tasks. For instance, Figure 6.5.*e* illustrates the reduction in wavelength while preserving an exceptional F_1-score. Furthermore, as shown in Figure 6.5.*b*, the MLP effectively minimized wavelengths in datasets with limited observations, such as Fresh Meats with 120. Nevertheless, a notable limitation is the prolonged computational time required for high-dimensional problems, which is nearly two hours, with a clear inability to reduce the feature space extensively within fewer iterations.

The Figure 6.3 shows Fresh Meat, Fruit Purée, and Olive Oils Quadrum reduced with ICAFS KNN and projected using UMAP.

Results for the best strength

From Table 6.2, it can be seen the best results obtained according to our tie breaker criteria during each t for KNN and SVM. Let's recap that we don't seek the small subset, neither the one that has a maximum F_1-score; only those that don't degrade system performance significantly. From Breast Cancer results on KNN, we could notice that most of the subsets include the most informative features. This is positive because even though we don't focus on getting the most significative features, we could find subsets in which these features appeared; moreover, the F_1-score obtained at each t shows an

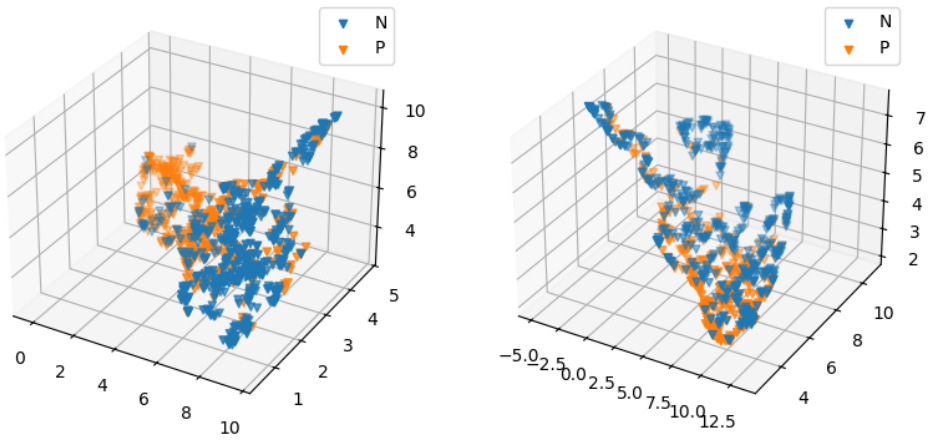

Figure 6.2: **UMAP Representation for Algarrobo Dataset with ICFAS KNN**

incremental tendency in relation to higher strengths. Other interesting facts noted are the number of features selected as the interaction incremented; this phenomenon happened due to the feature interaction being stronger. The same thing can be said for Wine, with the only distinction that some of the important features were not included as Alcohol, pH and Sulphates in lower strength; but only in greater strength. On Letter Recognition, the F_1-scores weren't increasing as the two other datasets; instead, it presents a notable fluctuation on each iteration.

On the other hand, for SVM, the very same behavior as KNN was observed, and such results are displayed in Table 6.2. One thing to notice is the difference between the features selected, confirming that the use of different learning algorithms may lead to different sets of features. Another thing that is very visible is that the more strength, the less feature reduction; on the other hand, lower strength leads to smaller interactions, allowing smaller subsets.

The figure is a series box plot describing the progression of the F_1 scores during each t. This trend observed on the plots confirms that with greater strength, the F_1 scores increased, but this is something that needs to be analyzed with care. All subsets generated at $t \geq 2$ include relevant features; as an example, let us take from Figure 6.2 the letter recognition best subset on all strengths. Hence, one can notice that the intersection between all subsets is high. In other words, the feature versatility expected in higher strength resulted low. Moreover, for all experiments with $t = 6$ at every $i \in T$, the subsets selection remains the same and that could be confirmed on Figure 6.6 with F_1-scores plotted on the same coordinate. Then, this only confirms that interaction in greater strengths is stricter and may include a few additional features, but let us remember that ICAFS method's main priority is to select the best accuracy possible for classification problems and the exclusion of a few features is not a problem. Moreover, according to one of the many objectives for feature selection, is the selection of a subset well enough that does not degrade performance, and this is a fundamental part for ICAFS; so given these two statements, we can say that the best t for ICAFS is $t = 2$ and $t = 3$, because

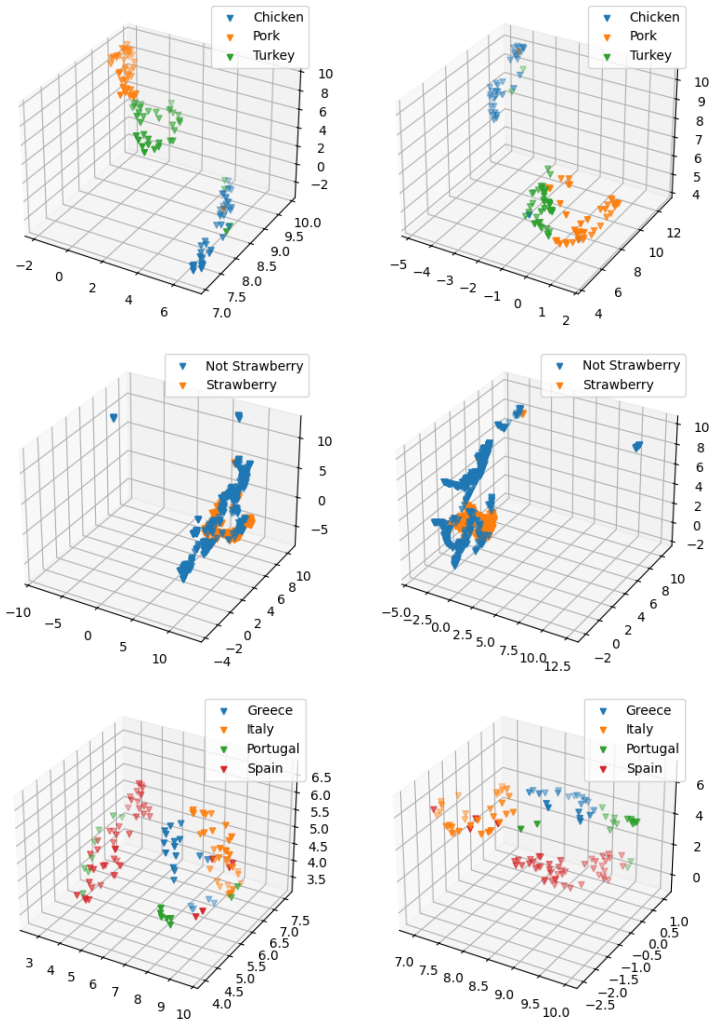

Figure 6.3: **UMAP Representation for Fresh Meats,Fruits Purée and Olive Oils using ICAFS KNN**

it allows smaller interaction resulting in greater feature reduction at lower computational time. Moreover, subsets selected at $t \geq 4$ seemed to get to the point of generating the same subsets among all iterations and features picked can be also seen on subsets generated in lower t. Secondly, the time for computing covering arrays with $t \geq 4$ consumed high computational resources and takes longer in finishing. Instead, future work in ICAFS must be focused on designing faster binary non-deterministic covering arrays algorithms as an alternative to IPOG for $t = 2$; by bringing to life one of the many mathematical approaches discussed on Section 2.1.1. This way, different subsets can be generated during multiple runs, enabling a possible diversity in subsets generated.

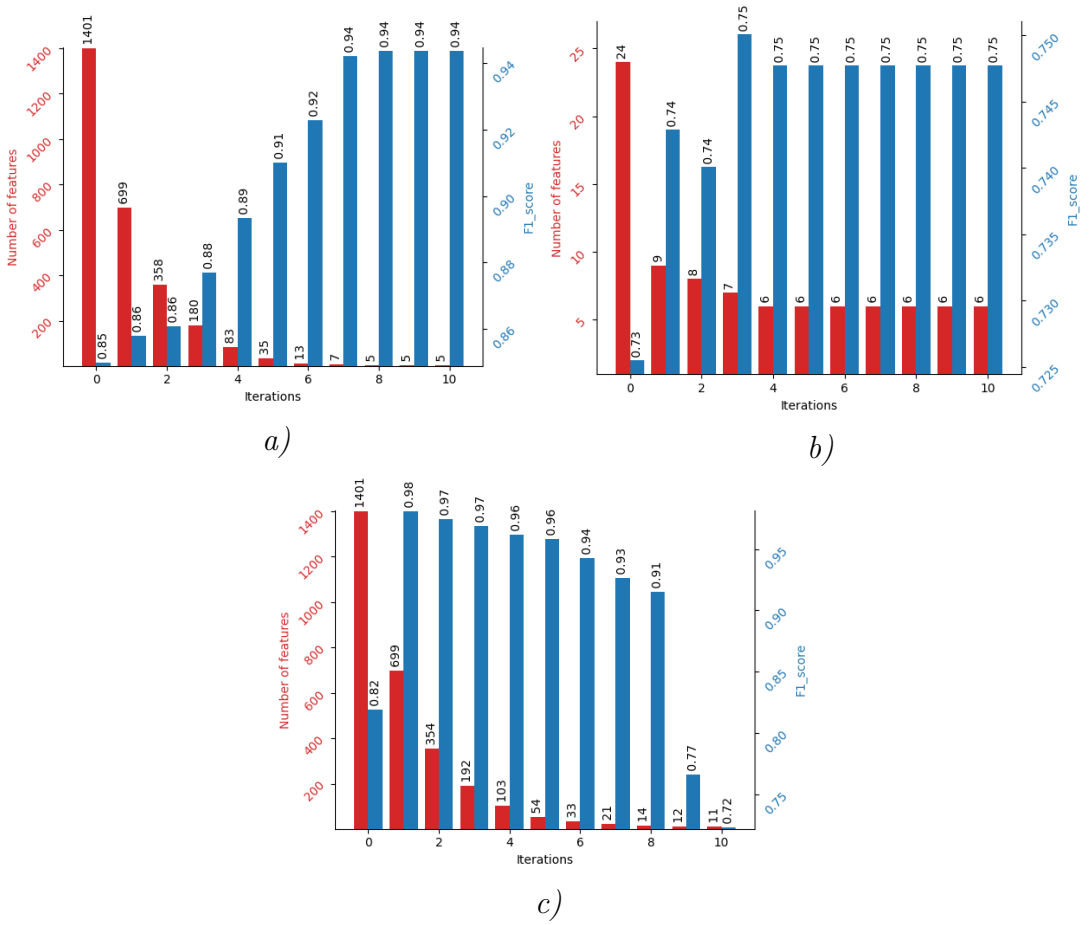

Figure 6.4: **Best sub-selection of wavelengths obtained along 10 iterations with CAFS with a) Cacao KNN, b) Algarrobo KNN, c) Cacao MLP**

6.3 Performance Test

Although this experiment primarily aims to identify the optimal subsets of wavelengths for high-dimensional chemometrics applications using ICAFS, CAFS, and BBA; ICAFS will be evaluated under three distinct scenarios: (1) a dataset consisting of 3000 samples with 10,000 total features employing KNN, (2) a dataset with 3000 samples and 10,000 total features using SVM, and (3) a dataset containing 10,000 observations and 11 classes with KNN and SVM. The datasets were synthetically generated using the make classification function available in sklearn. Below, hyper-parameters for each generated dataset are detailed in the subsequent listing.

1. A four-dimensional hypercube is constructed using the most informative features, encompassing a total of 3000 observations. Each class is represented by four Gaussian clusters situated at the vertices of the hypercube, implying that samples for each class are generated from four distinct distributions. This arrangement intentionally induces overlap between clusters, with a separation magnitude of 5.

2. We utilize the same parameters as employed in the prior dataset, while decreasing the separation to 2, which allows the clusters to assume a closer proximity.

3. A nine-dimensional hypercube characterized by 1000 distinct features, comprising 11 classes, each containing 11 Gaussian clusters, with a separation of 2 with KNN and SVM.

As illustrated in Figure 6.7.*a*, the results demonstrate a remarkable advancement in the KNN algorithm, where a reduction in the number of features is associated with an increase in F_1-score. A similar enhancement is observed for SVM, as shown in Figure 6.7.*b*; particularly in the context of more closely aligned Gaussian clusters. In the case of KNN with eleven classes, as the features are progressively reduced, there is a significant impact on the F_1-scores; however, this trend becomes evident starting at iteration 7, as depicted in Figure 6.7.*c*. In contrast, on Figure 6.7.*d* where SVM yields lower F_1-scores. Even though there is a noticeable F_1-score improvement $\approx 9\%$ from iteration 0 to 8. The lower F_1-scores yielded may be attributable to the number of clusters per class, suggesting that SVM may not be the most suitable algorithm for addressing this problem.

6.4 BBA Results

The concluding three rows of Table 6.2 elucidate the outcomes of the BBA algorithm after 10 iterations. In the context of the cacao problem, this algorithm successfully identified 441 wavelengths from a total of 1401, achieving an accuracy $\gtrapprox 85\%$. Similarly, in the case of non-plus/plus Algarrobo, the algorithm selected 7 wavelengths and was proficient in identifying the two most pertinent wavelengths within the circle in a square. For Fruit Purée, Fresh Meats, and Olive Oils, it also selected a vast number of wavelengths compared to ICAFS and CAFS.

The substantial quantity of selected wavelengths results from the limited ability to further condense the feature space within the optimal subset at each iteration. To replicate the same performance observed with CAFS and ICAFS, it is necessary to conduct multiple runs using the most advantageous feature subset, which necessitates conducting numerous experiments and consistently optimizing hyperparameter selection.

6.5 Overall Results and Complexity

The complexity of ICAFS is mainly governed by the construction of covering arrays using the IPOG algorithm. The overall complexity of IPOG is $O(d^{t-1} \times n^{t-1} \times \log n)$, because from each $\log n$ test or row generated, it takes $d^{t-1} \times n^{t-1}$ to find whether each possible combination of parameters covers the most combinations of parameters' values. Then, it

can be said that the overall complexity of ICAFS is $O(kd^{t-1}n^{t-1}\log n)$, where k is the number of iterations passed as a hyperparameter to ICAFS. On the other hand, CAFS shows a better complexity because the construction of the covering array is extracted, taking into account the best case $O(n \times \frac{m \times lr}{2})$, because for each test, it has to wait for the selected learning algorithm or lr to process each subset of wavelengths, which means that the time complexity is fully driven by the learning algorithm used. Moreover, each time the algorithm selects a subset, the columns are reduced by half at each iteration.

Before concluding the results obtained, our tie-breaking criterion involves prioritizing a small number of wavelengths that achieve a F_1-score sufficiently comparable to that achieved with an extensive feature space, so that performance is not degraded [75].

ICAFS kNN/SVM algorithm demonstrated superior performance compared to BBA and other variants of CAFS and ICAFS, as evidenced by its higher F_1-score on the wavelengths selected. Nonetheless, it is imperative to acknowledge that this should not be regarded as the sole determining factor, as the outcomes of the experiments demonstrate that both the number of wavelengths selected and the achieved F_1-score are contingent upon the choice of learning algorithms employed. On the other hand, non-plus/plus algorrobo performed really well on ICAFS OPF, but the number of wavelengths selected really contrasted with those selected for CAFS and ICAFS, with only 3 wavelengths and a decent F_1-score on SVM. For CIS, despite being a synthetic dataset, only ICAFS approximates the primal characteristics X and Y.

The Olive Oils dataset exhibited outstanding performance using the ICAFS OPF algorithm with just three wavelengths; however, it remains questionable whether the same can be asserted for CAFS, which consistently obtains F_1-scores $\gtrapprox 90\%$ across all variations. Furthermore, both the Fruit Purée and Fresh Meats datasets achieved remarkable outcomes when evaluated with CAFS/ICAFS. Finally, MLP version showed great performance regarding the F_1-score obtained, but according to our tiebreaker factor, it did not select a small subset of wavelengths compared to the other variants and took a lot of computation resources. However, the practitioner should choose between fewer selected wavelengths and minor variations in accuracy based on specific application needs. As shown in Table 6.6, the execution time is provided in seconds for each variation of the algorithms.

	Dataset	*Number*	*wavelengths selected*	*Max No. Iterations*	*F_1-Score*
ICAFS kNN	Cacao Nibs	5	1739, 1848, 1914, 2106, 2362	9	93.02
	Non-Plus/Plus Algarrobo	3	NGRDI, NDVI, RVI	10	70.92
	Fresh Meat	6	11634.3875, 1713.499, 1719.2875, 1746.3015, 1836.991, 1842.7805	10	94.89
	Fruit Purée	8	1404.985, 1443.585, 1474.465, 1516.925, 1609.565, 1679.044, 1771.684, 1802.564	10	95.05
	Fresh Meat	6	11634.3875, 1713.499, 1719.2875, 1746.3015, 1836.991, 1842.7805	10	94.89
	Olive Oils Quadrum	4	974.4775, 1032.3625, 1130.77, 1617.021	10	88.40
	Circle In Square	3	X, Y, Y40	2	99.02
ICAFS SVM	Cacao Nibs	2	1219, 1938	10	73.34
	Non-Plus/Plus Algarrobo	3	Edge Red, CIVE, VEG	3	74.85
	Fresh Meat	3	1622.8095, 1719.2875, 1750.1605	9	92.38
	Fruit Purée	5	1088.467, 1188.826, 1250.586, 1377.966, 1559.385	10	93.69
	Olive Oils Quadrum	4	968.689, 1126.911, 1250.404, 1603.5135	10	90.69
	Circle In Square	3	X, Y, X10	3	97.49
ICAFS OPF	Cacao Nibs	6	1737, 1812, 1929, 1983, 2111, 2300	10	95.94
	Non-Plus/Plus Algarrobo	9	N GRDI, NDVI, RVI, DVI, EVI, REVI, NDRE, RERVI, REDVI	2	77.45
	Fresh Meat	3	1539.838, 1580.3595, 1684.5555	10	90.74
	Fruit Purée	6	1277.606, 1296.906, 1366.386, 1451.305, 1609.565, 1690.624	10	93.47
	Olive Oils Quadrum	3	964.83, 1130.77, 1265.84	10	94.20
	Circle In Square	3	X, Y, Y40	2	99.24
ICAFS MLP	Cacao Nibs	133	1656, 1662, 1663, ... 2290, 2396, 2418	10	98.26
	Non-Plus/Plus Algarrobo	3	R, MGRVI, REDVI	10	73.22
	Fresh Meat	11	1377.755, 1416.346, 1429.8525, 1520.543, 1561.0635, 1615.0915 ...,1800.3295, 1808.0475	10	90.74
	Fruit Purée	66	1289.186, 1304.626, 1308.486, ...,1787.124, 1790.984	10	93.69
	Olive Oils Quadrum	7	986.0545, 1092.179, 1215.671, 1339.163, 1468.4445, 1601.584, 1730.8655	9	86.77
	Circle In Square	3	X, Y, Y40	3	95.60

Table 6.1: **Wavelengths resulted from executing BBA and CAFS on digh dimensional chemometrics datasets**

	Dataset	*Number*	*wavelengths selected*	*Max No. Iterations*	F_1-*Score*
CAFS kNN	Cacao Nibs	5	1725, 1830, 1929, 2075, 2117	8	94.38
	Non-Plus/Plus Algarrobo	6	Edge Red, EXGR, VEG, RGBVI, NDVI, DVI	10	74.77
	Fresh Meat	4	1128.8405, 1537.9085, 1715.4285, 1755.949	9	97.49
	Fruit Purée	5	1165.666, 1289.186, 1339.366, 1420.425, 1447.445	10	95.14
	Olive Oils Quadrum	5	972.548, 1317.9385, 1408.628, 1607.3735, 1730.8655	10	91.40
	Circle In Square	4	X, Y, X10, Y40	3	98.34
CAFS SVM	Cacao Nibs	4	1219, 1937, 2102, 2312	10	83.61
	Non-Plus/Plus Algarrobo	3	NIR, NGBDI, VEG	10	75.72
	Fresh Meat	6	1022.715, 1080.602, 1377.755, 1485.81, 1622.8095, 1755.949	10	97.49
	Fruit Purée	5	1157.946, 1381.826, 1408.845, 1443.585, 1682.904	10	94.95
	Olive Oils Quadrum	6	970.6185, 1132.7005, 1225.3195, 1254.263, 1645.9645, 1838.9215	6	94.60
	Circle In Square	4	X, Y, X10, Y40	4	97.75

Table 6.2: **Wavelengths resulted from executing BBA and CAFS on digh dimensional chemometrics datasets**

	Dataset	*Number*	*wavelengths selected*	*Max No. Iterations*	*F_1-Score*
CAFS OPF	Cacao Nibs	4	1389, 1511, 1940, 2140	10	94.14
	Non-Plus/Plus Algarrobo	5	NGRDI, GBRI.1, MGRVI, DVI, REVI, REDVI	10	77.01
	Fresh Meat	6	1121.1225, 1179.0095, 1453.0075, 1613.162, 1755.949, 1771.386	10	96.33
	Fruit Purée	6	1146.366, 1347.086, 1447.445, 1455.165, 1466.745, 1597.985	10	96.43
	Olive Oils Quadrum	7	914.662, 1001.4905, 1144.2775, 1209.8825, 1572.6415, 1661.4005, 1727.0065	7	91.24
	Circle In Square	2	Y, Y40	1	98.45
CAFS MLP	Cacao Nibs	14	1124, 1213, 1310, 1339, 1399 ,. . . , 2477	7	91.49
	Non-Plus/Plus Algarrobo	2	G, MGRVI	3	74.01
	Fresh Meat	5	1084.461, 1250.404, 1686.485, 1725.077, 1752.09	10	97.49
	Fruit Purée	7	1011.267, 1049.867, 1161.806, 1304.626, 1424.285, 1443.585, 1679.044	18	91.39
	Olive Oils Quadrum	15	930.098, 949.394, 976.407, 1020.7855, 1034.292, . . . 1746.3015, 1887.16	10	90.02
	Circle In Square	3	X, Y10, Y20	4	93.42
BBA	Cacao Nibs	441	1100,1101, 1102 . . .	20	93.09
	Non-Plus/Plus Algarrobo	7	G, NIR, EXG, NGBDI, GBRI.1, MGRVI, RERVI	20	72.16
	Fresh Meat	140	1009.2085, 1018.856, 1022.715, . . . ,1856.287, 1858.2165, 1860.146	20	94.37
	Fruit Purée	84	899.327, 907.047, 914.767, 918.627, 937.927, . . . ,1760.104, 1763.964	20	91.54
	Olive Oils Quadrum	183	810.469, 812.3985, 818.187, 835.5525, 837.482, . . . , 1885.2305, 1887.16, 1894.878	20	86.78
	Circle In Square	2	X,Y	20	98.74

Table 6.3: **Wavelengths resulted from executing BBA and CAFS on digh dimensional chemometrics datasets**

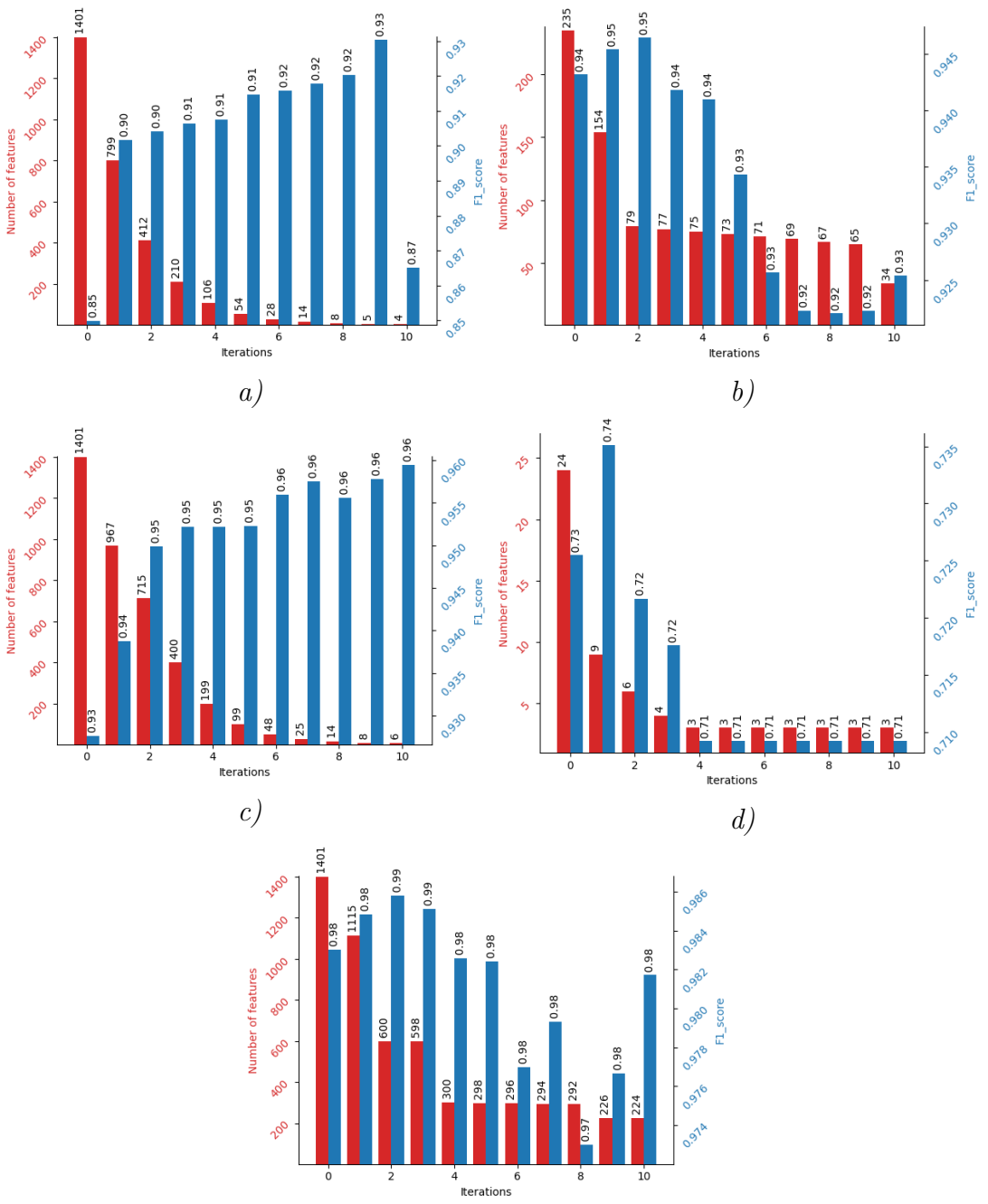

Figure 6.5: **Best sub-selection of wavelengths obtained along 10 iterations with CIAFS with a) Cacao KNN, b) Fresh Meat MLP, c) Cacao OPF d) Algarrobo KNN, and e) Cacao MLP**

	Dataset	*Number*	*features selected*	*Max No. Iterations*	F_1-score
$t = 2$	Breast Cancer	3	perimeter worst, smoothness worst, concavity worst	10	94.19
	Letter Recognition	8	y2bar, xybar, x2ybr, xy2br, x-ege, xegvy, y-ege, yegvx	5	92.66
	Wine Quality	3	chlorides, free sulfur dioxide, total sulfur dioxide	10	96.01
$t = 3$	Breast Cancer	3	concave points mean, radius worst, texture worst	10	96.41
	Letter Recognition	10	y-bar, x2bar, y2bar, xybar, x2ybr, xy2br, x-ege, xegvy, y-ege, yegvx	2	95.73
	Wine Quality	3	residual sugar, total sulfur dioxide, density	10	97.72
$t = 4$	Breast Cancer	5	area mean, texture worst, perimeter worst, area worst, smoothness worst	10	96.59
	Letter Recognition	10	onpix, x-bar, y-bar, x2bar, y2bar, xybar, xy2br, xegvy, y-ege, yegvx	2	93.42
	Wine Quality	7	fixed acidity, volatile acidity, chlorides, total sulfur dioxide, pH, sulphates, alcohol	2	98.74
$t = 5$	Breast Cancer	5	symmetry se, radius worst, texture worst, smoothness worst, concavity worst	10	96.41
	Letter Recognition	8	y2bar, xybar, x2ybr, xy2br, x-ege, xegvy, y-ege, yegvx	4	92.66
	Wine Quality	5	residual sugar, chlorides, total sulfur dioxide, density, alcohol	10	98.95
$t = 6$	Breast Cancer	13	..., radius worst, texture worst, concavity worst, symmetry worst, fractal dimension worst	10	97.16
	Letter Recognition	10	y-bar, x2bar, y2bar, xybar, x2ybr, xy2br, x-ege, xegvy, y-ege, yegvx	10	95.73
	Wine Quality	7	volatile acidity, residual sugar, chlorides, total sulfur dioxide, density, sulphates, alcohol	10	99.04

Table 6.4: **Features resulted from executing ICAFS KNN on Breast Cancer, Letter Recognition and Wine Quality**

	Dataset	*Number*	*features selected*	*Max No. Iterations*	*F_1-Score*
$t = 2$	Breast Cancer	2	area worst, concave points worst	10	94.46
	Letter Recognition	12	onpix, x-bar, y-bar, x2bar, y2bar, xybar, x2ybr, xy2br, x-ege, xegvy, y-ege, yegvx	2	90.87
	Wine Quality	3	total sulfur dioxide, density, sulphates	10	96.32
$t = 3$	Breast Cancer	3	concave points mean, texture worst, perimeter worst	10	96.23
	Letter Recognition	13	height, onpix, x-bar, y-bar, x2bar, y2bar, xybar, x2ybr, xy2br, x-ege, xegvy, y-ege, yegvx	1	91.55
	Wine Quality	3	residual sugar, total sulfur dioxide, density	10	97.83
$t = 4$	Breast Cancer	7	concave points mean, fractal dimension se, ..., oncave points worst, symmetry worst	10	97.54
	Letter Recognition	12	onpix, x-bar, y-bar, x2bar, y2bar, xybar, x2ybr, xy2br, x-ege, xegvy, y-ege, yegvx	1	90.87
	Wine Quality	7	fixed acidity, volatile acidity, chlorides, total sulfur dioxide, pH, sulfates, alcohol	2	99.02
$t = 5$	Breast Cancer	5	perimeter mean, radius se, compactness se, texture worst, concavity worst	10	96.26
	Letter Recognition	11	x-bar, y-bar, x2bar, y2bar, xybar, x2ybr, xy2br, x-ege, xegvy, y-ege, yegvx	1	90.86
	Wine Quality	5	volatile acidity, residual sugar, total sulfur dioxide, density, alcohol	10	99.10
$t = 6$	Breast Cancer	13	area mean, smoothness mean, compactness mean, concavity mean ..., concavity worst, concave points worst	10	97.53
	Letter Recognition	13	y-box, height, onpix, x-bar, y-bar, x2bar, y2bar, xybar, x2ybr, xy2br, xegvy, y-ege, yegvx	1	90.77
	Wine Quality	8	volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, alcohol	10	99.43

Table 6.5: **Features resulted from executing ICAFS SVM on Breast Cancer and Wine Quality**

Dataset	ICAFS KNN	ICAFS SVM	ICAFS MLP	ICAFS OPF	CAFS KNN	CAFS SVM	CAFS MLP	CAFS OPF	BBA
Cacao Nibs	189.3021	247.33	29516.288	2982.830	25.159	130.694	24466.6029	4091.31	172.100
Algarrobo	4.322	9.258	195.1395	524.643	28.59	42.56	1399.90	3251.520	80.520
Fruit Purée	12.76	18.8976	4750.85	1643.96	24.027	33.44	5165.767	3426.186	128.44
Fresh Meats	24.913	17.4978	926.422	74.269	11.22	7.49	1571.18	119.485	99.244
Olive Oils	32.814	35.801	1587.82	85.098	11.55	8.220	1890.65	121.45	58.682
CIS	54.425	391.064	4532.609	35323.352	198.633	2462.36	3444.15	3444.157	686.53

Table 6.6: **Execution time taken for each of the algorithms addressed in seconds**

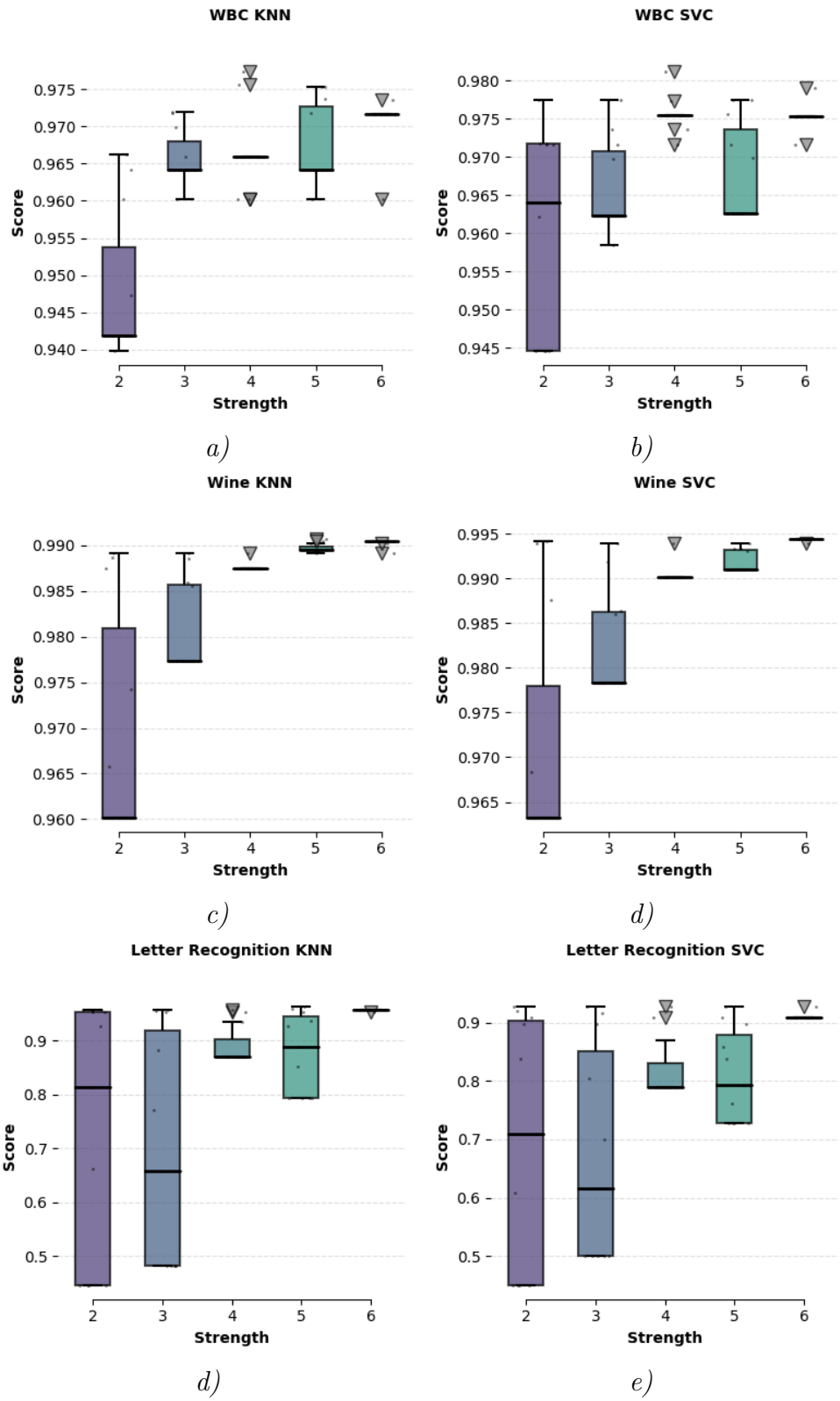

Figure 6.6: F_1-**Score progession within 10 iterations across different** t **values.**

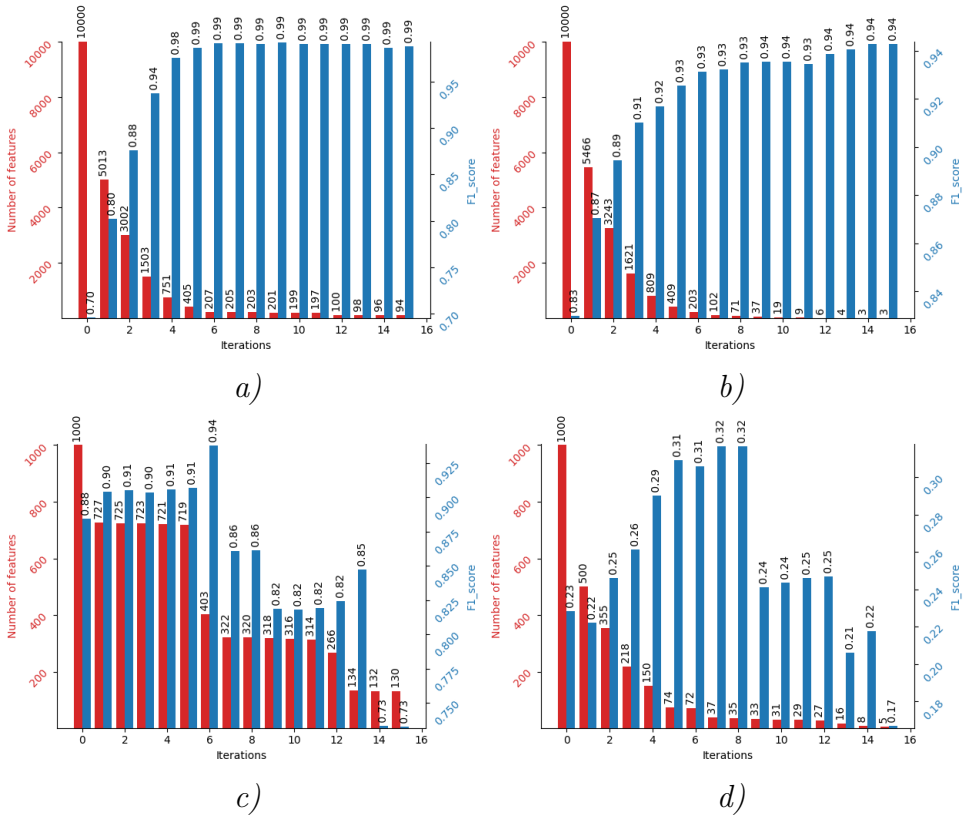

Figure 6.7: **Performance test experimented with with a) 10,000 features with KNN, b) 10,000 features with SVM, c) 11 classes KNN, and d) 11 classes SVM**

7. CONCLUSIONS

***Overview:** This final chapter serves as the culmination of this extensive work. Moreover, in this chapter we will discuss our findings along with the future direction that ICAFS can take.*

7.1 Conclusions

During this work, a new wavelength selection algorithm was proposed for chemometric applications called ICAFS that relied on binary covering arrays, a strong combinatorial data structure that modeled interaction between t parameters; allowing the simulation of feature interaction across wavelengths and iteratively reducing the feature space by taking advantage of its intrinsic characteristic of generating new covering arrays in previously selected subsets. This new approach was tested on high-dimensional chemometric applications that showed exceptional results. The method was also validated to show its applicability in the field of chemometrics.

In addition, the results showed that the algorithms based on covering arrays can significantly outperform other metaheuristic methods such as the binary bat algorithm in high-dimensionality chemometrics applications in reducing considerably wavelength space and maintaining competitive performance, as evidenced by superior F_1 scores obtained by using different classifiers. This assertion stems from the deep analysis carried out with BBA against CAFS/ ICAFS, because covering-array like algorithms, upon selecting a candidate subset of wavelengths, model new interactions between these parameters, thereby iteratively leading to a reduced set of wavelengths. In contrast, BBA did not exhibit such a feature, as its search strategy is predicated on swarm optimization. Then, the Bat population is updated according to a specific heuristic, without consideration of interactions among multiple system wavelengths.

Notably, a primary distinction between CAFS and ICAFS is that it generates new covering arrays with each iteration, ensuring that subsequent test sets are minimized more effectively compared to CAFS. This is because CAFS, while adapting the covering array to the selected wavelengths, typically results in a larger number of test cases, owing to its inability to generate new interactions for features selected. For this reason, ICAFS showed better impactful results than CAFS, as even though CAFS showcases F_1-scores significantly greater with some of the learning algorithms addressed, it shows that the feature reduction is not smaller and such characteristics are important due to the overlapping and noise in features available for high-dimensional chemometrics applications. Hence, ICAFS successfully extended CAFS and by the results demonstrated, it could generalize a powerful and well-validating tool for chemometrics.

Additionally, ICAFS was validated in very high-dimensional synthetic datasets with around 10,000 features and 11 classes, showing outstanding results but at the same time, reinforcing the idea that the exclusion of features in further iterations may lead to decreasing scores. Moreover, in rigorous experiments conducted, the best t was selected. But not only that, this experimentation was also applied with a non-chemometrics dataset, implying its applicability for generic applications. As a result, a preliminary version of ICAFS can be found in the next github repository.

Finally, all these results were obtained with near-minimum hyper-parameter optimization

in the learning algorithms utilized or default configuration, and even though they demonstrated outstanding results, they didn't demonstrate an optimum set of hyperparameters for getting the best F_1-scores in the problems addressed. Therefore, this methodology only elucidates the proposal of ICAFS in comparison with CAFS and BBA.

7.2 Future Work

Future research directions may include the use of deep techniques for classification, and the incorporation of an embedded covering array-guided search of hyper-parameters of neural classifiers. The application of the algorithm to other chemometric problems is also an issue to be addressed. Similarly, different algorithms to generate covering arrays may provide a trade-off between complexity and a more accurate exploration of the feature space in terms of the interactions between variables. Moreover, ICAFS was fully tested on chemometrics and partially on public benchmark datasets showing impactful results; but regression tasks remain unexplored, therefore, this represents an area where ICAFS can address by implementing a filter/ranking variation. Furthermore, the methodology delineated in ICAFS within the scope of this research can be extrapolated to bioinformatics domains. For example, there exists evidence supporting the applicability of feature selection techniques in genetic data for the classification of cancer variations [76]. This represents a challenge that ICAFS is suitably equipped to address.

BIBLIOGRAPHY

[1] R. González-Domínguez, A. Sayago, and Á. Fernández-Recamales, "An overview on the application of chemometrics tools in food authenticity and traceability," *Foods*, vol. 11, no. 23, p. 3940, Dec. 2022, ISSN: 2304-8158. DOI: 10.3390/foods11233940.

[2] D. Granato, P. Putnik, D. B. Kovacevic, *et al.*, "Trends in chemometrics: Food authentication, microbiology, and effects of processing," *Comprehensive Reviews in Food Science and Food Safety*, vol. 17, no. 3, pp. 663–677, Mar. 2018, ISSN: 1541-4337. DOI: 10.1111/1541-4337.12341.

[3] W. Castro, J. Oblitas, M. De-La-Torre, C. Cotrina, K. Bazan, and H. Avila-George, "Classification of cape gooseberry fruit according to its level of ripeness using machine learning techniques and different color spaces," *IEEE Access*, vol. 7, pp. 27 389–27 400, 2019, ISSN: 2169-3536. DOI: 10.1109/access.2019.2898223.

[4] M. De-la-Torre, O. Zatarain, H. Avila-George, *et al.*, "Multivariate analysis and machine learning for ripeness classification of cape gooseberry fruits," *Processes*, vol. 7, no. 12, p. 928, Dec. 2019, ISSN: 2227-9717. DOI: 10.3390/pr7120928.

[5] M. J. Anzanello and F. S. Fogliatto, "A review of recent variable selection methods in industrial and chemometrics applications," *European Journal of Industrial Engineering*, vol. 8, no. 5, pp. 619–645, 2014. DOI: 10.1504/EJIE.2014.065731. [Online]. Available: https://www.inderscienceonline.com/doi/abs/10.1504/EJIE.2014.065731.

[6] M. Al-Awadhi and R. Deshmukh, "Enhancing honey adulteration detection with optimal subspace wavelength reduction in vis-nir reflection spectroscopy," *IEEE Access*, vol. 11, pp. 144 226–144 243, 2023, ISSN: 2169-3536. DOI: 10.1109/access.2023.3343731.

[7] A. Puttipipatkajorn and A. Puttipipatkajorn, "Rapid quality evaluation of camellia oleifera seed kernel using a developed portable nir with optimal wavelength selection," *IEEE Access*, vol. 10, pp. 8317–8327, 2022, ISSN: 2169-3536. DOI: 10.1109/access.2022.3143818.

[8] W. Castro, M. De-la-Torre, H. Avila-George, J. Torres-Jimenez, A. Guivin, and B. Acevedo-Juárez, "Amazonian cacao-clone nibs discrimination using nir spectroscopy coupled to naïve bayes classifier and a new waveband selection approach,"

Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy, vol. 270, p. 120 815, Apr. 2022, ISSN: 1386-1425. DOI: 10.1016/j.saa.2021.120815.

[9] C. Kumaravelu and A. Gopal, "A review on the applications of near-infrared spectrometer and chemometrics for the agro-food processing industries," in *2015 IEEE Technological Innovation in ICT for Agriculture and Rural Development (TIAR)*, IEEE, Jul. 2015, pp. 8–12. DOI: 10.1109/tiar.2015.7358523.

[10] R. Leardi, R. Boggia, and M. Terrile, "Genetic algorithms as a strategy for feature selection," *Journal of Chemometrics*, vol. 6, no. 5, pp. 267–281, 1992. DOI: https://doi.org/10.1002/cem.1180060506. [Online]. Available: https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/abs/10.1002/cem.1180060506.

[11] M. C. U. Araújo, T. C. B. Saldanha, R. K. H. Galvão, T. Yoneyama, H. C. Chame, and V. Visani, "The successive projections algorithm for variable selection in spectroscopic multicomponent analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 57, no. 2, pp. 65–73, 2001, ISSN: 0169-7439. DOI: https://doi.org/10.1016/S0169-7439(01)00119-8. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169743901001198.

[12] R. Guha, B. Chatterjee, S. K. Khalid Hassan, S. Ahmed, T. Bhattacharyya, and R. Sarkar, "Py_fs: A python package for feature selection using meta-heuristic optimization algorithms," in *Computational Intelligence in Pattern Recognition*. Springer Singapore, Sep. 2021, pp. 495–504, ISBN: 9789811625435. DOI: 10.1007/978-981-16-2543-5_42.

[13] S. Vivas, C. Cobos, and M. Mendoza, "Covering arrays to support the process of feature selection in the random forest classifier," in *Machine Learning, Optimization, and Data Science*, G. Nicosia, P. Pardalos, G. Giuffrida, R. Umeton, and V. Sciacca, Eds., Cham: Springer International Publishing, 2019, pp. 64–76, ISBN: 978-3-030-13709-0.

[14] H. Dorado, C. Cobos, J. Torres-Jimenez, D. D. Burra, M. Mendoza, and D. Jiménez, "Wrapper for building classification models using covering arrays," *IEEE Access*, vol. 7, pp. 148 297–148 312, 2019. DOI: 10.1109/ACCESS.2019.2944641.

[15] M. A. Anitha and K. K. Sherly, "A novel forward filter feature selection algorithm based on maximum dual interaction and maximum feature relevance(mdimfr) for machine learning," in *2021 International Conference on Advances in Computing and Communications (ICACC)*, 2021, pp. 1–7. DOI: 10.1109/ICACC-202152719.2021.9708300.

[16] A. Jenul, S. Schrunner, K. H. Liland, U. G. Indahl, C. M. Futsæther, and O. Tomic, "Rent—repeated elastic net technique for feature selection," *IEEE Access*, vol. 9, pp. 152 333–152 346, 2021.

[17] H. Liu, M. Zhou, and Q. Liu, "An embedded feature selection method for imbalanced data classification," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 3, pp. 703–715, 2019. DOI: 10.1109/JAS.2019.1911447.

[18] "Irrelevant features and the subset selection problem," in *Machine Learning Proceedings 1994*, W. W. Cohen and H. Hirsh, Eds., San Francisco (CA): Morgan Kaufmann, 1994, pp. 121–129, ISBN: 978-1-55860-335-6. DOI: https://doi.org/10.1016/B978-1-55860-335-6.50023-4. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9781558603356500234.

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[20] G. B. Sherwood, S. S. Martirosyan, and C. J. Colbourn, "Covering arrays of higher strength from permutation vectors," *Journal of Combinatorial Designs*, vol. 14, no. 3, pp. 202–213, 2006. DOI: 10.1002/jcd.20067.

[21] D. R. Kuhn and V. Okun, "Pseudo-exhaustive testing for software," in *2006 30th Annual IEEE/NASA Software Engineering Workshop*, 2006, pp. 153–158. DOI: 10.1109/SEW.2006.26.

[22] Y. Lei and K. Tai, "In-parameter-order: A test generation strategy for pairwise testing," in *Proceedings Third IEEE International High-Assurance Systems Engineering Symposium (Cat. No.98EX231)*, 1998, pp. 254–261. DOI: 10.1109/HASE.1998.731623.

[23] J. Torres-Jimenez, I. Izquierdo-Marquez, and H. Avila-George, "Methods to construct uniform covering arrays," *IEEE Access*, vol. 7, pp. 42 774–42 797, 2019, ISSN: 2169-3536. DOI: 10.1109/access.2019.2907057.

[24] G. O. H. Katona, "Two applications (for search theory and truth functions) of sperner type theorems," *Periodica Mathematica Hungarica*, vol. 3, no. 1-2, pp. 19–26, 1973.

[25] D. J. Kleitman and J. Spencer, "Families of k-independent sets," *Discrete Mathematics*, vol. 6, no. 3, pp. 255–262, 1973.

[26] A. Rényi, *Foundations of Probability*. Hoboken, NJ, USA: Wiley, 1971.

[27] C. J. Colbourn, "Covering arrays from cyclotomy," *Designs, Codes and Cryptography*, vol. 55, no. 2-3, pp. 201–219, 2010.

[28] C. J. Colbourn, S. S. Martirosyan, G. L. Mullen, D. Shasha, G. B. Sherwood, and J. L. Yucas, "Products of mixed covering arrays of strength two," *Journal of Combinatorial Designs*, vol. 14, no. 2, pp. 124–138, 2006.

[29] J. Yan and J. Zhang, "Backtracking algorithms and search heuristics to generate test suites for combinatorial testing," in *Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC)*, vol. 1, Washington, DC, USA: IEEE Computer Society, 2006, pp. 385–394.

[30] J. Yan and J. Zhang, "A backtracking search tool for constructing combinatorial test suites," *Journal of Systems and Software*, vol. 81, no. 10, pp. 1681–1693, 2008.

[31] J. Bracho-Rios, J. Torres-Jimenez, and E. Rodriguez-Tello, "A new backtracking algorithm for constructing binary covering arrays of variable strength," in *Advances in Artificial Intelligence (Lecture Notes in Computer Science)*, A. H. Aguirre, R. M. Borja, and C. A. R. García, Eds., vol. 5845, Berlin, Germany: Springer-Verlag, 2009, pp. 397–407.

[32] Y.-W. Tung and W. S. Aldiwan, "Automating test case generation for the new generation mission software system," in *Proceedings of the IEEE Aerospace Conference*, vol. 1, Big Sky, MT, USA: IEEE Computer Society, Mar. 2000, pp. 431–437.

[33] Y. Lei, R. Kacker, D. R. Kuhn, V. Okun, and J. Lawrence, "Ipog: A general strategy for t-way software testing," in *14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'07)*, IEEE, Mar. 2007. DOI: 10.1109/ecbs.2007.47.

[34] S. Wang, J. Tang, and H. Liu, "Feature selection," in *Encyclopedia of Machine Learning and Data Mining*. Springer US, 2016, pp. 1–9, ISBN: 9781489975027. DOI: 10.1007/978-1-4899-7502-7_101-1.

[35] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, no. null, pp. 1157–1182, Mar. 2003, ISSN: 1532-4435.

[36] S. Huang, "Supervised feature selection: A tutorial," *Artificial Intelligence Research*, vol. 4, Mar. 2015. DOI: 10.5430/air.v4n2p22.

[37] K. Pearson, "Notes on the history of correlation," *Biometrika*, vol. 13, no. 1, pp. 25–45, 1920. DOI: 10.1093/biomet/13.1.25.

[38] R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986. DOI: 10.1007/BF00116251.

[39] K. Kira and L. A. Rendell, "A practical approach to feature selection," in *Proceedings of the 9th International Workshop for Machine Learning*, San Francisco: Morgan Kaufmann, 1992, pp. 249–259.

[40] M. Dash, H. Liu, and H. Motoda, "Consistency based feature selection," in *Proceedings of the Forth Pacific Asia Conference on Knowledge Discovery and Data Mining*, London: Springer-Verlag, 2000, pp. 98–109.

[41] D. Mo and S. H. Huang, "Feature selection based on inference correlation," *Intelligent Data Analysis*, vol. 15, no. 3, pp. 375–398, 2011.

[42] X. S. Yang, "Bat algorithm for multi-objective optimisation," *International Journal of Bio-Inspired Computation*, vol. 3, no. 5, p. 267, 2011, ISSN: 1758-0374. DOI: 10.1504/ijbic.2011.042259.

[43] R. Y. M. Nakamura, L. A. M. Pereira, D. Rodrigues, K. A. P. Costa, J. P. Papa, and X.-S. Yang, "Binary bat algorithm for feature selection," in *Swarm Intelligence and Bio-Inspired Computation*. Elsevier, 2013, pp. 225–237, ISBN: 9780124051638. DOI: 10.1016/b978-0-12-405163-8.00009-0.

[44] J. P. Papa, A. X. Falcão, and C. T. N. Suzuki, "Supervised pattern classification based on optimum-path forest," *International Journal of Imaging Systems and Technology*, vol. 19, no. 2, pp. 120–131, May 2009, ISSN: 1098-1098. DOI: 10.1002/ima.20188.

[45] F. T. ALGorain and J. A. Clark, "Covering arrays ml hpo for static malware detection," *Eng*, vol. 4, no. 1, pp. 543–554, 2023, ISSN: 2673-4117. DOI: 10.3390/eng4010032. [Online]. Available: https://www.mdpi.com/2673-4117/4/1/32.

[46] Z. Li, Y. Chen, Y. Song, K. Lu, and J. Shen, "Effective covering array generation using an improved particle swarm optimization," *IEEE Transactions on Reliability*, vol. 71, no. 1, pp. 284–294, 2022. DOI: 10.1109/TR.2021.3132147.

[47] Y. Wang, H. Wu, X. Niu, C. Nie, and J. Xu, "A constrained covering array generator using adaptive penalty based parallel tabu search," in *2022 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2022, pp. 82–86. DOI: 10.1109/ICSTW55395.2022.00028.

[48] "A hybrid feature selection method for dna microarray data," *Computers in Biology and Medicine*, vol. 41, no. 4, pp. 228–237, 2011, ISSN: 0010-4825. DOI: https://doi.org/10.1016/j.compbiomed.2011.02.004.

[49] H. Liu, M. Zhou, and Q. Liu, "An embedded feature selection method for imbalanced data classification," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 3, pp. 703–715, 2019. DOI: 10.1109/JAS.2019.1911447.

[50] C. Gini, *Variabilità e mutabilità: contributo allo studio delle distribuzioni e delle relazioni statistiche*. Cagliari, Italy: Tipogr. di P. Cuppini, 1912.

[51] L. Ceriani and P. Verme, "The origins of the gini index: Extracts from variabilità e mutabilità (1912) by corrado gini," *The Journal of Economic Inequality*, vol. 10, no. 3, pp. 421–443, 2012. DOI: 10.1007/s10888-011-9188-x.

[52] D. Pham, A. Ghanbarzadeh, E. Koç, S. Otri, S. Rahim, and M. Zaidi, "The bees algorithm — a novel tool for complex optimisation problems," in *Intelligent Production Machines and Systems*. Elsevier, 2006, pp. 454–459, ISBN: 9780080451572. DOI: 10.1016/b978-008045157-2/50081-x.

[53] P. Gärdenfors, "On the logic of relevance," *Synthese*, vol. 37, pp. 351–367, Mar. 1978. DOI: 10.1007/BF00873245.

[54] R. Battiti, "Using mutual information for selecting features in supervised neural net learning," *IEEE Transactions on Neural Networks*, vol. 5, no. 4, pp. 537–550, 1994. DOI: 10.1109/72.298224.

[55] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005. DOI: 10.1109/TPAMI.2005.159.

[56] A. Dalvand, M. B. Dowlatshahi, and A. Hashemi, "Sgfs: A semi-supervised graph-based feature selection algorithm based on the pagerank algorithm," in *2022 27th International Computer Conference, Computer Society of Iran (CSICC)*, 2022, pp. 1–6. DOI: 10.1109/CSICC55295.2022.9780486.

[57] Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, vol. 23, no. 19, pp. 2507–2517, Aug. 2007.

[58] P. Pudil, J. Novovičová, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognition Letters*, vol. 15, no. 11, pp. 1119–1125, 1994, ISSN: 0167-8655. DOI: https://doi.org/10.1016/0167-8655(94)90127-9. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0167865594901279.

[59] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1, pp. 273–324, 1997, Relevance, ISSN: 0004-3702. DOI: https://doi.org/10.1016/S0004-3702(97)00043-X. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S000437029700043X.

[60] W. Castro, R. Seminario, W. Nauray, B. Acevedo-Juárez, M. De-la-Torre, and H. Avila-George, "Multispectral drone imagery dataset for plus and non-plus neltuma pallida trees in northern peru," *Data in Brief*, vol. 60, p. 111 645, 2025, ISSN: 2352-3409. DOI: https://doi.org/10.1016/j.dib.2025.111645. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352340925003750.

[61] O. Al-Jowder, E. Kemsley, and R. Wilson, "Mid-infrared spectroscopy and authenticity problems in selected meats: A feasibility study," *Food Chemistry*, vol. 59, no. 2, pp. 195–201, 1997, ISSN: 0308-8146. DOI: https://doi.org/10.1016/S0308-8146(96)00289-0. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0308814696002890.

[62] J. K. Holland, E. K. Kemsley, and R. H. Wilson, "Use of fourier transform infrared spectroscopy and partial least squares regression for the detection of adulteration of strawberry purées," *Journal of the Science of Food and Agriculture*, vol. 76, no. 2, pp. 263–269, 1998. DOI: https://doi.org/10.1002/(SICI)1097-0010(199802)76:2<263::AID-JSFA943>3.0.CO;2-F. eprint: https://scijournals.onlinelibrary.wiley.com/doi/pdf/10.1002/%28SICI%291097-0010%28199802%2976%3A2%3C263%3A%3AAID-JSFA943%3E3.0.CO%3B2-F. [Online]. Available: https://scijournals.onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-0010%28199802%2976%3A2%3C263%3A%3AAID-JSFA943%3E3.0.CO%3B2-F.

[63] H. S. Tapp, M. Defernez, and E. K. Kemsley, "Ftir spectroscopy and multivariate analysis can distinguish the geographic origin of extra virgin olive oils," *Journal of Agricultural and Food Chemistry*, vol. 51, no. 21, pp. 6110–6115, 2003, PMID: 14518931. DOI: 10.1021/jf030232s.

[64] D. Dua and C. Graff, *UCI machine learning repository*, 2019. [Online]. Available: http://archive.ics.uci.edu/ml.

[65] W. N. Street, W. H. Wolberg, and O. L. Mangasarian, "Nuclear feature extraction for breast tumor diagnosis," in *IS&T/SPIE 1993 International Symposium on Electronic Imaging: Science and Technology*, SPIE, vol. 1905, 1993, pp. 861–870.

[66] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," *Decision Support Systems*, vol. 47, no. 4, pp. 547–553, 2009. DOI: 10.1016/j.dss.2009.05.016.

[67] Y. Er and A. Atasoy, "The classification of white wine and red wine according to their physicochemical qualities," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 4, no. Special Issue, pp. 23–26, 2016, ISSN: 2147-6799.

[68] P. W. Frey and D. J. Slate, "Letter recognition using holland-style adaptive classifiers," *Machine Learning*, vol. 6, pp. 161–182, 1991.

[69] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge, UK: Cambridge University Press, 2000.

[70] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in R*. New York, NY: Springer, 2013.

[71] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, urlhttp://www.deeplearningbook.org.

[72] X. Deng, Q. Liu, Y. Deng, and S. Mahadevan, "An improved method to construct basic probability assignment based on the confusion matrix for classification problem," *Information Sciences*, vol. 340-341, pp. 250–261, 2016, ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2016.01.033. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S002002551600044X.

[73] G. H. de Rosa and J. P. Papa, "Opfython: A python implementation for optimum-path forest," *Software Impacts*, p. 100 113, 2021, ISSN: 2665-9638. DOI: 10.1016/j.simpa.2021.100113.

[74] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction," *arXiv preprint arXiv:1802.03426v3*, Sep. 2020. arXiv: 1802.03426 [stat.ML].

[75] P. Pudil, J. Novovičová, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognition Letters*, vol. 15, no. 11, pp. 1119–1125, 1994, ISSN: 0167-8655. DOI: 10.1016/0167-8655(94)90127-9.

[76] S. H. Bouazza, N. Hamdi, A. Zeroual, and K. Auhmani, "Gene-expression-based cancer classification through feature selection with knn and svm classifiers," 2015. DOI: 10.1109/ACS.ISALY.2015.7306368.