Universidad Politécnica
de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Máster Universitario en Inteligencia Artificial

Trabajo Fin de Máster

Mejora de Métodos para Búsqueda de Patrones en Conjuntos de Datos

Autor(a): Aarón García Muñiz
Tutor(a): Emilio Serrano Fernández
Co-Tutor(a): Guillermo Vigueras González

Madrid, 09/07/2024

Este Trabajo Fin de Máster se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Trabajo Fin de Máster
Máster Universitario en Inteligencia Artificial

Mejora de Métodos para Búsqueda de Patrones en Conjuntos de Datos
09/07/2024

Autor(a): Aarón García Muñiz
Tutor(a): Emilio Serrano Fernández
Grupo de Ingeniería Ontológica
ETSI Informáticos
Universidad Politécnica de Madrid

Co-Tutor(a): Guillermo Vigueras González
Departamento LSIIS
ETSI Informáticos
Universidad Politécnica de Madrid

Resumen

Tradicionalmente, ha existido una dificultad a la hora de ser capaz de extraer información en forma de patrones o subgrupos de un conjunto de datos, y en particular, encontrar un set de subgrupos (*i.e* conjunto de subgrupos no ordenados) de alta calidad, que sirvan para representar la mayor parte del conjunto de datos, pero que no presenten redundancia entre ellos (*i.e* ser capaz de representar distintas partes del conjunto de datos). En la literatura se han propuesto varias soluciones para poder lograr esta tarea, sin embargo, en algunas soluciones es necesario definir valores de manera manual para los distintos parámetros utilizados en la ejecución, lo que puede provocar altos tiempos de ejecución o una búsqueda ineficiente. Por otra parte, en la literatura se suele emplear una única métrica de evaluación de los subgrupos en la fase de búsqueda, lo que provoca que no podemos asegurar la total calidad de los subgrupos, ya que se hace necesario analizar distintas cualidades para ver la importancia del subgrupo. Finalmente, es necesario evaluar no solo la calidad de un subgrupo, sino también el impacto que tiene en el set de subgrupos añadir un subgrupo, en términos de redundancia. Por lo tanto, se propone DASSD (Descubrimiento de Set de Subgrupos Dinámico y Adaptable), un algoritmo heurístico que descubre sets de subgrupos maximizando la calidad de los mismos y minimizando la redundancia presente en el set. Además, la característica principal de este algoritmo es la capacidad adaptativa y dinámica que posee, siendo capaz de establecer para cualquier conjunto de datos y en cualquier fase de la búsqueda, los valores óptimos a emplear en los parámetros requeridos. Por otra parte, se ha propuesto emplear la técnica de *Feature Selection* como paso previo a la búsqueda de subgrupos, con el objetivo de reducir o eliminar variables redundantes o irrelevantes del conjunto de datos permitiendo así optimizar la búsqueda. Finalmente, se han estudiado los resultados obtenidos por DASSD en diferentes conjuntos de datos, así como la comparación con diferentes algoritmos presentes en el estado del arte.

Abstract

Traditionally, there has been a difficulty in being able to extract information in the form of patterns or subgroups from a dataset, and more specifically, finding a set of subgroups (i.e. a set of unordered subgroups) of high quality, which serve to represent most of the dataset, but which do not present redundancy between them (i.e. being able to representing different parts of the dataset). In the literature, several approaches have been made to achieve this task, however, in some solutions it is necessary to set manual values for the different parameters employed in the execution, which may lead to high execution times or an inefficient search. On the other hand, in the literature a single evaluation metric for the subgroups is usually used during the search phase, which means that we cannot ensure the total quality of the subgroups, since it is necessary to analyse different qualities to see the importance of the subgroup. Finally, it is necessary to evaluate not only the quality of a subgroup, but also the impact that adding a subgroup has on the set of subgroups, in terms of redundancy. Therefore, we proposed DASSD (Dynamic and Adaptive Subgroup Set Discovery), a heuristic algorithm that discovers subgroup sets maximizing their quality and minimizing the redundancy present in the set. Moreover, the main characteristic of this algorithm is the adaptative and dynamic capacity it has, being able to determine for any dataset and in any phase of the search, the optimal values to employ in the required parameters. On the other hand, we propose to use the Feature Selection technique as a prior step to searching for subgroups, with the aim of reducing or eliminating redundant or irrelevant variables from the dataset, thus allowing the search to be optimized. Furthermore, the results obtained by DASSD in different datasets have been studied, as well as the comparison with State-of-the-Art algorithms.

Contents

Chapter 1

Introduction

Exploratory Data Analysis [39] is a data analysis approach whose objective is to find general patterns in datasets. These patterns may contain relationships between unexpected variables that shed new information or knowledge about the domain of the data considered. Among the possible techniques included in Exploratory Data Analysis, there is Subgroup Discovery, a data mining technique responsible for discovering interpretable descriptions of subsets of the data that are related to a variable of interest (i.e. subgroups) [34, 13]. In SD, as with the majority of local pattern mining techniques, an important limitation is the number of patterns that can be generated. Numerous patterns can lead to a very high degree of redundancy, meaning that the same information is described in the data set; as well as making the subsequent analysis of the generated subgroups difficult. This challenge results in the definition of the pattern set mining problem [6, 14, 32]. In its classic form, subgroup discovery is also referred to as top-k subgroup mining, which means extracting the top k subgroups with the highest rank in accordance with a local quality measure and a number k of subgroups defined by the user. Although this approach allows controlling the number of patterns returned, it does not solve the problem of redundancy, since the k best subgroups could only describe a small part of the dataset. On the other hand, the discovery of the top k subgroups can be performed following an exhaustive or heuristic strategy. The exhaustive strategy will lead to discovering the best possible solution with the penalty of a high computational cost or even being an unfeasible task to complete, depending on the dimensionality of the data set. On the other hand, the heuristic strategy allows obtaining high quality subgroups without the need to perform a complete exploration of the search space. However, the heuristic strategy may present limitations due to the possible need to specify values for the required parameters, such as the number of subgroups to consider in the exploration or a threshold value for a chosen measure to discard patterns (i.e. BeamSearch). Furthermore, the SD technique can be employed to discover sets of ordered subgroups (list model) or sets of unordered subgroups (set model). The main difference lies in how the subgroups obtained have to be interpreted. While in a list model, subgroups describe a part of the dataset in an exclusive way (i.e. subgroups cannot cover the same instance of the dataset), in set models, subgroups can intersect similar regions of the dataset described (subgroups can cover the same instances). Therefore, the main limitation to consider in set models is to be able to guarantee a set of subgroups with high quality, but minimizing the redundancy that may be found in the set of subgroups, while in list models, the challenge comes in determining which

subgroup most accurately represents a particular region of the dataset, as well as avoiding obtaining subgroups that represent a tiny part of the dataset. In this work, a heuristic algorithm is proposed to obtain a set of non-redundant and high quality non-ordered subgroups (set model). Furthermore, the use of a dynamic and adaptive subgroup selection technique is proposed, which allows finding the optimal number of subgroups to be considered at any time during the exploration. Besides, the problem of having a high dimension search space is addressed, reducing the number of variables in the dataset using the Feature Selection (FS) technique. Thus, the impact of using FS techniques has been analysed as a prior step to using the proposed Subgroup Discovery algorithm, comparing the results obtained without employing Feature Selection (i.e. considering all the variables presented in the datasets) and employing Feature Selection to remove irrelevant variables. Additionally, a comparison is made with two list model algorithms and a set model algorithm present in the State-of-the-Art (SotA). Finally, a discussion of the results obtained using various quality measures is carried out.

Structure of the paper Besides the introduction, this work contains four additional chapters:

- Chapter 2 (Materials and methods): This chapter includes the following sections: definition and descriptions of datasets, State-of-the-Art of the Feature Selection and Subgroup discovery techniques. Regarding the Subgroup Discovery SotA, we introduce a subgroup definition, the different cultures and models, and a summary of different algorithms, measures and pruning methods.
- Chapter 3 (Algorithm): In this chapter, the algorithm DASSD is introduced and explained in detail. Furthermore, besides the explanation of the different phases of the algorithm, we propose a solution to the problem of guided search when manual values are defined.
- Chapter 4 (Results): In this section, we show the performance of different SotA and DASSD algorithms. We analyse the relation of different quality measures when subgroups are being compared. Moreover, we provide an individual comparison for each dataset, as well as a global comparison for a better understanding of algorithms performance.
- Chapter 5 (Conclusions and Future work): This section provides a summary of the different tasks and goals to achieve in this works. Furthermore, besides algorithm analysis and conclusions obtained, we propose several search lines to follow in order to better understand how to improve the quality and performance of DASSD algorithm.

Chapter 2

Materials and methods

2.1 Datasets

A dataset can be modelized as D(X,Y) = $\{a^1, a^2, a^n\}$, where n is the number of instances in the dataset, X = $\{x^1, x^2, x^i\}$ represents the set of explanatory variables (i.e. features) and Y = $\{y^1, y^2, y^j\}$ represents the target variable in case of an unitarget problem or a set of target variables in the case of a multitarget problem. The domain for variables presented in the sets X and Y can be nominal or numerical. In this work, seven datasets available in the UCI and KEEL repositories have been used to analyse the performance of the proposed algorithm, as well as the algorithms presented in the State-of-the-Art.

Dataset	Rows	Variables	Targets	Datatype (nom/num)
Mushrooms	8124	21	2	(22-0)
Splice	3190	60	3	(60-0)
Tuandromd	4464	241	2	(241-0)
Coil2000	9822	85	2	(0-85)
Chess	3196	36	2	(36-0)
Connect4	67557	42	3	(42-0)
Adult	30162	14	2	(8-6)

Figure 2.1: Description of datasets

2.2 Feature Selection

Feature Selection (FS) is a dimensionality reduction method employed with the objective of eliminating those variables/features that are irrelevant or redundant. By reducing the number of variables in the dataset, we can increase the precision of the algorithm employed and reduce the execution time due to the reduced number of relevant features.

Historically, three *Feature Selection* methods can be distinguished in the State-of-the-Art: filter, wrapper and hybrid methods. The filter methods use statistical metrics to evaluate the dependence between features with the target variables. On the other

hand, wrapper methods depend on the evaluation of a classifier to select the features, since it selects those that satisfy the best results based on the value of a metric (i.e. accuracy) for a classifier. Finally, hybrid methods emerge as an alternative that combines a two-phase filter-wrapper scheme. The first step consists of using a filter method to reduce the dimensional space of variables, thus improving the effectiveness and efficiency of the wrapper method used in the second phase, obtaining as a result the most relevant variables of the subset. In the context of this work, it has been decided to use the filter methods, explained in more details in the following section.

2.2.1 Filter methods

Filter methods are employed to evaluate the relevance of features based on certain statistical properties, being independent of learning algorithms. Those features that score below a certain threshold are removed, while features that score above it are selected. The main advantage of filter methods over other feature selection methods is that they are generally less computationally demanding, and thus can easily be scaled to very high dimensional data. Two types of filter can be distinguished: the *univariate* filters evaluate each features individually without considering other features, meanwhile *multivariate* filters consider the relevance between features subsets in each iteration. Thus, since each feature is considered separately, univariate methods only focus on feature relevance and cannot detect feature redundancy. However, univariate methods are less computationally heavy than multivariate methods and thus can as effectively be scaled to very high dimensional data. Some univariate filter methods presented in the State-of-the-Art are:

- Pearson coefficient: is a statistical measure that evaluates the linear relationship between two variables. It gives information about the magnitude of the association, or correlation, as well as the direction of the relationship. It ranges from -1 to 1, where -1 indicates a perfect negative linear relationship, a value of 0 implies that there is no linear dependency between the variables, and 1 indicates a perfect positive linear relationship.

$$\rho(X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{2.1}$$

  where cov(X,Y) represents the covariance and σ_X, σ_Y the standard deviation, respectively.

- Fisher score [37]: It selects those features, such that in a data space spanned by the selected features, and maximize the distances between data points in different classes while minimizing the distances between data points in the same class.

$$Fs(f_i) = \frac{S_b(f_i)}{\sum_{k=1}^{c} S_t^k(f_i)} \tag{2.2}$$

  where $S_b(f_i) = \sum_{k=1}^{c} n_k (m_i^k - m_i)^2$, n_k is the number of samples in the k-th class, m_i^k is the mean of the i-th feature in the k-th class m_i is the mean of the ith feature in dataset X, and $S_t^k(f_i) = \sum_{j=1}^{n_k} (x_{ij}^k - m_i^k)^2$ with x_{ij}^k denoting the value of the i-th feature for the j-th sample in the k-th class.

- Symmetrical Uncertainty (SU) [15] is a normalized version in the range [0,1] of Mutual Information (MI), where the objective is to measure the level of similarity and correlation between two variables. The concept of symmetric refers to the

equality between comparing the mutual information between X given Y, and Y given X, thus reducing the number of comparisons to be made.

$$SU(X,Y) = 2\frac{H(X) - H(X|Y)}{H(X) + H(Y)} \tag{2.3}$$

where H(X) and H(Y) represents the entropies of variables X and Y, and H(X|Y) represents the conditional entropy of X when Y is known.

The entropy of a dataset can be understood in terms of the probability distribution of observations within the dataset belonging to different classes. Thus, the entropy measures the level of uncertainty or randomness in the distribution of classes within the dataset.

$$H(X) = -\sum_{x \in X} p(x) log_2 p(x) \tag{2.4}$$

$$H(X|Y) = -\sum_{y \in Y} p(y) \sum_{x \in X} p(x|y) log_2 p(x|y) \tag{2.5}$$

On the other hand, some multivariate filter methods presented in the State-of-the-Art are:

- Minimum redundancy maximum relevance (mrMR) [17, 33, 7]: is a method employed to measure the importance of a set of features using mutual information (MI). To achieve this task, the relevance of features with respect to the target variable will be measures, but at the same time, redundancy between features will be penalized. Regarding the task of finding most relevant features, the criterion of maximum relevance is used:

$$maxD(X,c) = \frac{1}{|X|} \sum_{X_i \in X} MI(X_i, c) \tag{2.6}$$

  On the other hand, the set of features obtained employing the maximum criterion may bring high relevance among the selected features. To solve this issue, the criterion of minimum redundancy is considered:

$$minR(X) = \frac{1}{|X|} \sum_{X_i, X_j \in X} MI(X_i, X_j) \tag{2.7}$$

  The combination and optimization of both criterion results in the mrMR criterion. Furthermore, the optimization of the mrMR method can be performed in two ways:

  - MID (mutual information difference criterion):

$$max_{X_i \notin X}[D(X_i, c) - R(X)] \tag{2.8}$$

  - MIQ (mutual information quotient criterion):

$$max_{X_i \notin X}[D(X_i, c)/R(X)] \tag{2.9}$$

- Conditional Mutual Information Maximization (CMIM) [12]: selects features that maximize their mutual information with the target variable to be predicted, conditional on any features already selected.

$$CMIM(X_n) = min_{j \in S} MI(X_i, Y|X_j) \tag{2.10}$$

- Fast correlation-based filter (FCBF) [42]: It is an algorithm which, using Symmetric Uncertainty, identifies irrelevant and redundant features. To perform this task, identify a set of features that are highly correlated with the target variable, to later be ranked using the predominant correlation. Consequently, redundant features will be eliminated using a search algorithm, thus obtaining the set of most relevant features.

2.3 Subgroup Discovery

Subgroup Discovery (SD) is a data mining technique that intends to reveal significant associations between variables in relation to a particular target property [34, 13]. In the case of structured data, these associations generally are named as subgroups and takes the form of an association rule [1]. Furthermore, the significant value of a subgroup, also known as interestingness, will be quantified by a given quality measure [41].

2.3.1 Subgroups

A subgroup, denoted by s, consists of a combination of conditions that defines a cover (i.e. a subset of dataset D), that describe a distribution with respect to a target value in the dataset D.

$$s : Comb \rightarrow Target_{value} \tag{2.11}$$

Conditions can be formally represented as triplets with the following format: Variable - Selector - Value. The possible selectors depend on the variable type: numeric variables support $\{\geq, \leq\}$ operators (i.e. $\{attr1 \geq 5\}$) while binary and nominal support {==} operator, i.e. $\{attr1 == "dog"\}$.

2.3.2 Cultures in Subgroup Discovery

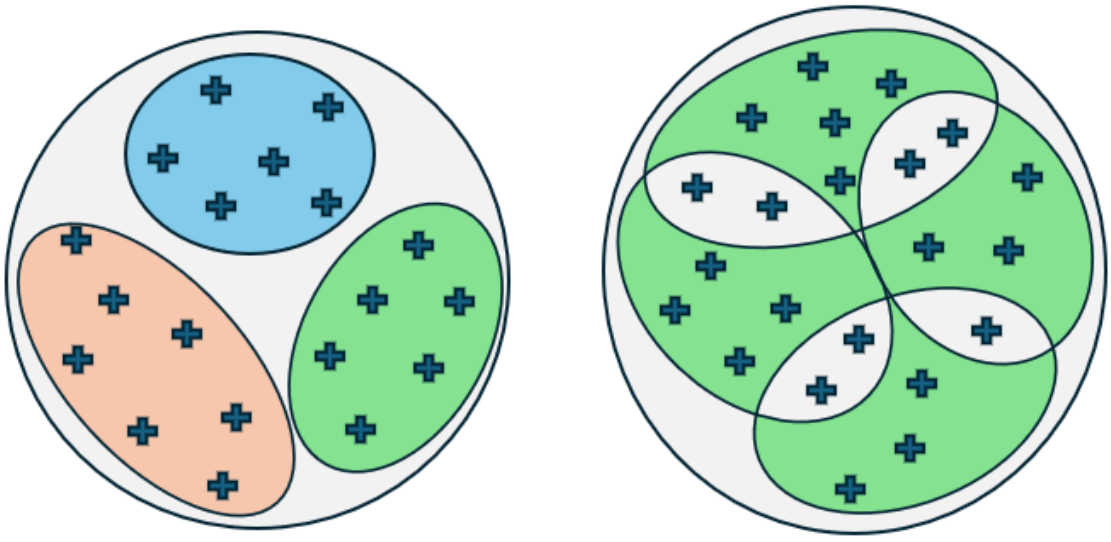
In the literature, two cultures related to subgroup discovery are distinguished (SI and KDD). The first is closely related to biostatistics and medical data analysis, particularly in the context of drug discovery, where analysis relies heavily on both treatments and outcomes. On the other hand, the second culture has its roots in the Data Mining and KDD communities and applies to all types of data. Association rules, set mining, contrast sets, and emergent patterns are all connected to the concept of descriptive induction [13]. Although both cultures share characteristics and limitations, they are employed in different areas and contexts, therefore, can be considered exclusive. It is important to notice that the goal of SD is not to find associations that best predict the value of the variable of interest given an unknown observation, but rather to obtain the most significant groups of observations that meet the subgroup condition and the variable of interest.

2.3.3 Subgroup lists and sets

Subgroup discovery algorithms are employed to discover a set of high-quality, statistically significant and non-redundant subgroups that together can describe a given dataset. More specifically, this process of discovering can be considered as an application of the LeGo (from Local Patterns to Global Models) framework, which detailed the procedures involved in moving from local data descriptions to a global model [21]. This global model can be divided in three types: top-k subgroups (i.e. best k ranked subgroups according to a quality measure), subgroup list (i.e. sequentially ordered set of subgroups), and subgroup set (i.e. unordered set of subgroups). In particular, we are focusing on the difference between subgroup lists and sets, which both are considered global model as they contemplate the subgroups' local coverage and their global coverage as sets.

On one hand, subgroup list models intends to be interpreted sequentially, due to the subgroups found are ordered in decreasing importance. This is because each subgroup in the model represents a unique partition of the dataset, hence each instance of the dataset is covered by only one subgroup.

On the other hand, subgroup sets allow for a semi-independent interpretation of each subgroup, due to each instance can be covered by one or more subgroup, as different subgroups may cover a conjunction of partitions of the dataset.



(a) Example of SD List model

(b) Example of SD Set model

Figure 2.2: Types of SD models

So, considering these properties, subgroups sets can be considered as more interpretable when examining all the subgroups, being subgroup lists more interpretable considering the instances.

Nevertheless, both types of models are more difficult to interpret when the number of subgroups within a model increases. Regarding subgroup lists, it is necessary to study both the covering subgroup and all the preceding ones for each instance. In the case of subgroup sets, it could be difficult to assess the individual contribution of

each subgroup given that there may be subgroups which cover similar instances (i.e. overlapping subgroups), as can be seen in Figure 2.2b.

2.3.4 Algorithms presented in SotA

Considering the global scope of this work, because neither the domain of the data used is associated with a specific domain, nor will the results obtained be analysed in a medical context, the proposed algorithm will fit within the KDD culture. Consequently, in SD-KDD culture, the workflow followed by the algorithms during the SD task, can be defined in three phases: subgroup generation, subgroup evaluation and ranking, and subgroup pruning [18].

2.3.4.1 Subgroup generation

In order to discover interesting subgroups, exhaustive and heuristic approach are used to perform this task. Exhaustive algorithms are commonly employed to maximize the quality of the subgroups found (i.e., obtain the highest quality subgroups possible). However, to guarantee this result it is necessary to analyse the entire search space, which, if this search space is too large, the application of these algorithms is not possible due to the high execution times. On the other hand, heuristic algorithms are those that use a heuristic function in order to not analyse the whole search space in each step, thus being more efficient in terms of execution times by reducing the potential number of subgroups that must be explored, but without being able to guarantee that the best subgroups will be found. Furthermore, Beam search is perhaps the most popular heuristic in subgroup discovery [23, 40, 28], having been empirically proved to be competitive in terms of quality compared to a complete search while providing a significant speed-up [28]. Moreover, it can be defined as a greedy hill-climbing strategy with three parameters: the beamwidth bw (i.e. number of subgroups stored), the maximum search depth d_{max}, and a quality measure to analyse the subgroups. At the beginning, it starts by discovering the top bw subgroups of length one (i.e. subgroups with one condition such as attr1 == "dog") according to the selected quality measure. Then, after the process of adding one more condition to the considered bw subgroups (i.e. candidates), only the wb best candidates (i.e. the highest quality value) are selected and discarding the rest. This process iteratively continues until d_{max} is reached.

Considering the exhaustive approach, some algorithms presented in the State-of-the-Art are:

- Apriori-SD [20]: An exhaustive search algorithm adapted from association rule learning to SD. Compared to APRIORI it only considers subgroups that contain the target in the right-hand side. Like CN2-SD, it also uses the weighted covering method.
- SD-Map [2]: SD-Map is an exhaustive subgroup discovery algorithm that uses the well-known FP-growth method [16] for mining association rules with adaptations for the subgroup discovery task. This algorithm uses the FPTree data structure to represent the entire dataset and extract subgroups in two steps; a complete FPTree is first built from the input dataset, after which successive conditional FPTrees are constructed recursively to mine the subgroups. Furthermore, SD-Map uses a modified FP-growth step that can compute the subgroup

quality directly without referring to other intermediate results.

- BSD [24]: BSD is a subgroup discovery algorithm that introduces the concept of dominance relation between subgroups. To handle the dominances between subgroups, BSD uses a bitset that stores for each discovered pattern and each row of the dataset whether the pattern appears in the row or not. Regarding the dominance relation, a subgroup S makes another subgroup S1 irrelevant by dominance if and only if the positive instances of the bitset of subgroup S1 are included in the positive instances of subgroup S and the negative instances of subgroup S are included in the negative instances of subgroup S1. This algorithm also uses a list of the best subgroups along with an optimistic estimation to prune the search space.

- : QFinder [11]: A subgroup discovery algorithm that aims to generate statistically credible subgroups and combines an exhaustive search with a cascade of filters based on metrics assessing key credibility criteria, including relative risk reduction assessment, adjustment on confounding factors, individual feature's contribution to the subgroup's effect, interaction tests for assessing between-subgroup treatment effect interactions and tests adjustment (multiple testing).

- VLSD [26]: An efficient Subgroup Discovery algorithm that combines an equivalence class exploration strategy and a pruning strategy based on optimistic estimate. The implementation of this proposal is based on vertical list data structure, which makes it easily parallelizable. The pruning based on optimistic estimate implies that, for all the subgroups generated, their optimistic estimate values are computed and compared with the threshold in order to discover whether they must be pruned (and it is, therefore, not necessary to explore their refinements), or whether their refinements (i.e., the next depth level) must also be explored. Additionally, the vertical list data structure is used in order to compute the subgroup refinements easily and efficiently by making list concatenations and set intersections. Moreover, it stores all the elements required, with the objective of avoiding multiple and unnecessary recalculations.

On the other hand, algorithms presented in the State-of-the-Art employing an heuristic strategy are:

- CN2-SD [23]: A beam search algorithm developed by adapting the standard classification rule learning approach CN2 to the subgroup discovery task. It introduces a weighted covering method, in which examples covered by a subgroup are not removed from the training set but their weights are decreased. These subgroups are obtained in the form of rules using a modified unusualness as the quality measure for rule selection.

- FSSD [4]: A subgroup discovery algorithm that exploits closure operators and tight optimistic estimates on the marginal contributions of SD-compatible measures to efficiently explore and prune unpromising areas of the search space, thereby providing non-redundant, diverse and high quality subgroups.

- SSD++ [31]: A heuristic algorithm with two main components: a SaC iterative loop that adds the best-found subgroup at the end of the subgroup list; and a beam search to find high-quality subgroups at each SaC iteration based on the compression gain measure.

- MCT4SD [5]: An exploration strategy leading to *any-time* pattern mining that can be adapted with different measures and policies. MCTS explores a search space by building a game tree incrementally and asymmetrically: the tree construction is driven by random simulations and an exploration/exploitation trade-off provided by the so called upper confidence bounds.

2.3.4.2 Subgroup evaluation

In order to perform the task of evaluating subgroups, several quality measures have been studied in the State-of-the-Art. In this section, we present the measures employed in this work, according to the classification made in [19].

- Measures of complexity: These measures are intended to describe the interpretability of the subgroups obtained.
  - Size: The subgroup set size is computed as the number of subgroups in the induced subgroup set.
  - Complexity: It measures the level of information presented in patterns. It is determined as the number of variables contained in the pattern.
  - Redundancy: It measures the level of repeated information among the induced pattern set. Subgroups can shared attributes and triplets. Considering share attributes, we refers to this type of redundancy as diversity. So, more diversity are presented in a subgroup set when lower attributes are shared among subgroups. On the other hand, when referring to shared triplets, we refers to this type of redundancy as overlapping. Subgroups sharing same triplets may lead to only be able to represent a small region of the data, due to same instances are being representing by different subgroups. Thus, for the purpose of maximizing diversity and minimizing overlapping, each subgroup compares its antecedent in terms of attributes and triplets with the rest of subgroups´ antecedents presented in the induced set. It can be computed as:

$$Redundancy(S_i) = \frac{1}{nS-1} \frac{\sum_{\forall j \neq i}^{nS-1} nA(S_i, S_j) + nT(S_i, S_j)}{Complexity(S_i)} \tag{2.12}$$

    where nS refers to the number of subgroups in the set, $nA(S_i, S_j)$ and $nT(S_i, S_j)$ refers to the number of attributes (diversity) and the number of triplets (overlapping) shared between considered subgroups. The average redundancy of a pattern set can be computed as:

$$REDUNDANCY = \frac{1}{nS} \sum_{i=0}^{nS} Redundancy(S_i) \tag{2.13}$$

    where nS is the number of induced subgroups and $Redundancy(S_i))$ refers to the redundancy value of subgroup S_i.

- Measures of generality and precision: These measures are employed to evaluate the percentage of instances of dataset D covered, and the precision of the subgroups in relation to the target variable.

- Coverage [19]: It measures the percentage of examples covered by a subgroup. This can be computed as:

$$Cov(R) = \frac{IS}{ID} \tag{2.14}$$

where ID refers to the number of total examples or rows present in the dataset and $IS = (TP + FP)$ refers to the number of examples that satisfy the conditions determined by the antecedent (i.e. examples covered by the antecedent). The average coverage of a subgroup set is computed as:

$$COV = \frac{1}{nS} \sum_{i=1}^{nS} Cov(S_i) \tag{2.15}$$

where nS is the number of induced subgroups and $Cov(S_i)$ refers to the coverage value of the subgroup i.

- Confidence [19]: It measures the relative frequency of examples satisfying the antecedent and the target value defined in the consequent among those satisfying only the antecedent. This can be computed as:

$$Cnf(R) = \frac{PS)}{IS} \tag{2.16}$$

where $PS = TP$ and refers to the number of examples that satisfy the conditions of the antecedent and also belong to the value for the target variable presented in the consequent. The average confidence of a pattern set is computed as:

$$CNF = \frac{1}{nS} \sum_{i=1}^{nS} Cnf(S_i) \tag{2.17}$$

where nS is the number of induced subgroups and $Cnf(S_i)$ refers to the confidence value of the subgroup i.

- Hybrid measures: These measures attempts to obtain a tradeoff between interest and precision in the results obtained.

  - Unusualness [19]: This measure is described as the weighted relative accuracy of a pattern (WRAcc). It can be calculated as:

$$WRAcc(R) = Cov(R) * (Cnf(R) - \frac{PD}{ID}) \tag{2.18}$$

The unusualness of a pattern can be described as the balance between its coverage, represented by $Cov(R)$, and its accuracy gain, denoted by $Cnf(R) - \frac{PD}{ID}$ where $PD = (TP + FN)$ refers to the examples that satisfy the target variable. The average unusualness of a pattern set can be computed as:

$$WRAcc = \frac{1}{nS} \sum_{i=1}^{nS} WRAcc(Si), \tag{2.19}$$

where nS is the number of induced subgroups and $WRAcc(S_i)$ refers to the unusualness value of the subgroup i.

- Measures of interest: These measures are intended for maximizing their potential interest to the user.
  - Information gain [29, 30]: It quantifies the reduction in entropy or surprise by splitting a dataset based on a specific value of a random variable. In the subgroup discovery context, we can measure how much information a subgroup gains by adding a new condition, thus allowing us to discriminate against those candidates with the least contribution. It is calculated as follows:

$$IG(D, v) = H(D) - H(D|v) \tag{2.20}$$

    Here, $IG(D, v)$ represents the information gain for the dataset D with respect to the variable v, $H(D)$ is the entropy of the dataset before any change, and $H(D|v)$ is the conditional entropy of the dataset when the variable v is added.

    An example of a IG calculation can be seen in Figure 2.3.

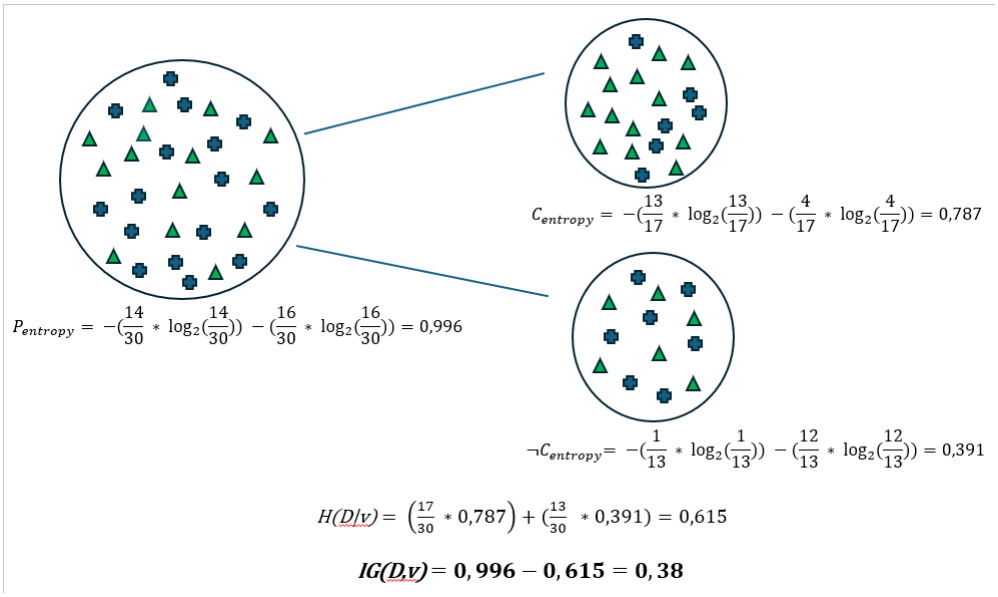


Figure 2.3: Information gain computation

  - Odds ratio OR [38, 10]: In the context of association rules, Odds ratio is a measure employed to quantify the statistical difference of two subgroups considering an outcome. Thus, the OR represents the ratio of the odds of the outcome occurring given a specific antecedent, compared to the odds of the outcome occurring in the absence of that antecedent. This measure enables us to evaluate the strength of the association between the antecedent (pattern) and the outcome of interest. For example, considering the subgroup P: ("Gender == Male and Smoker == True $\rightarrow$ Develop_cancer == True") with OR(P) = 4, people covered by this subgroup are 4 times more likely to suffer from cancer than others. Consequently, odds ratios (ORs) may be employed to compare subgroups in order to discover statistically relevant subgroups.

The odds ratio (OR) can be calculated using the following formula:

$$OR = \frac{TP \cdot TN}{FP \cdot FN} \tag{2.21}$$

Subgroup/Target	+	-
+	TP	FP
-	FN	TN

Figure 2.4: Matrix OR

Furthermore, this comparison can be extended to assess whether adding a new triplet to a subgroup improves the odds of the outcome occurring. This comparison helps determine the effectiveness of adding more information to a pattern. For instance, consider the subgroup P: ("Gender == Male and Smoker == True $\rightarrow$ Develop_cancer == True") and the subgroup P1: ("Gender == Male and Smoker == True and Age == [45,60] $\rightarrow$ Develop_cancer == True") an extended subgroup of P. If OR(P1) > OR(P), we could consider that the new information added to subgroup P is relevant. However, the difference between OR(P1) and OR(P) can be considered significant?. To address this topic, we must first recognize that a measure of quality is an estimation, because the value obtained may suffer a degree of variation or error. So, when we state that an estimation of the quality of a subgroup is 4, we mean that the quality is in an interval (i.e. confidence interval).

$$CI = [OR_-, OR_+] = [OR * exp(-w), OR * exp(w)] \tag{2.22}$$

where OR_- and OR_+ refer to the left bound and the right bound of the confidence interval of OR(P), and

$$w = \alpha * \sqrt{1/TP + 1/FP + 1/FN + 1/TN} \tag{2.23}$$

where α = 1.96 considering a 95% confidence interval (CI).

In [25] a proposal of the use of OR confidence intervals are presented. They stand that, considering subgroup P1 and its extended subgroup P2, P2 is statistically not significant if OR(P2) < OR(P1) or OR(P2) > OR(P1) and OR_-(P2) < OR_+(P1), which is defined as overlapping subgroups.

Nevertheless, the fact that two 95% confidence intervals overlap does not necessarily imply that the two subgroups are not significantly different from one another.

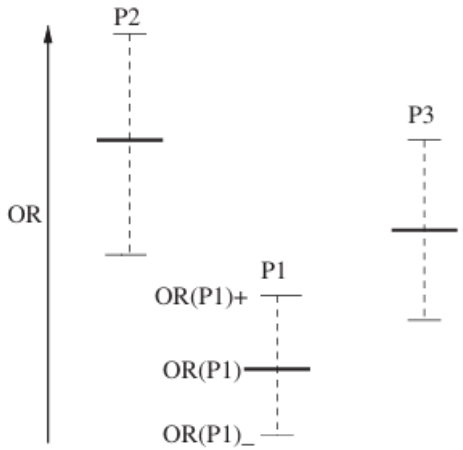


Figure 2.5: Example of OR confidence intervals overlapping [25]

In [3, 22, 8], this quandary is stated and resolved. When considering the statistical difference between 2 subgroups by looking at the overlap of the 95% CI in the subgroups, a type 1 error probability of 0.05 is often erroneously assumed, when its real value is 0.0056. Therefore, if the 95% confidence intervals overlap, the difference between the two subgroups can be statistically significantly. In order to solve this issue, 83.4% confidence intervals should be calculated instead to obtain a type 1 error probability of 0.05. Thus, to obtain a 83.4% CI, the value of α in Equation 2.23 has to be 1.39.

In addition, in order to interpret the OR as an effect size, a previous work [10] proposed the transformation of OR into Cohen's d. This transformation makes the interpretation of the OR easier, as it allows for considering an $OR > 6.71$ to have a similar effect size, regardless of the actual magnitude of the OR.

In cases where Cohen's d is not obtained, comparing subgroup sets based solely on mean values of the OR may lead to distorted results. Higher OR values can disproportionately influence the mean, while lower OR values may not receive due consideration. To address this, four intervals are defined:

* $OR < 1.68$ represents a very low effect.
* $1.68 < OR < 3.47$ represents a low effect.
* $3.47 < OR < 6.71$ represents a moderate effect.
* $OR > 6.71$ represents a high effect.

For ease of numerical representation, a value is assigned to each interval, resulting in the odds ratio range (ORR) being defined from 1 to 4. This allows for a more balanced comparison between subgroups in the subgroup set and avoids overemphasizing the impact of extremely high OR values.

2.3.4.3 Subgroup pruning

After the generation of the subgroups and their subsequent evaluation with the metrics presented, those subgroups that do not meet a certain condition are discarded. The commonly used criteria are: define a threshold for the measure used to evaluate the subgroups, save the top-k subgroups based on the chosen measure, add subgroups until a condition is met (i.e. a certain value is not improved, adding a subgroup worsens the quality of the subgroups obtained so far, etc.) or discard those subgroups that do not contain some previously defined variables. In particular, we focus on the limitations of defining a threshold value for the metric used to evaluate the subgroups or staying with the top-k subgroups. Both approaches require defining a predefined value in order to be used (i.e. threshold = 0.4, k = 10), which means that different values will give different results in the same dataset or different behaviour when sharing threshold values for different datasets.

Furthermore, ignorance or lack of understanding of the use of the chosen quality measure can lead to defining an incorrect value of the threshold. For instance, in Figure 2.6, considering a threshold value for a given quality measure of 0.1, it can be seen that we would be accepting all the subgroups presented.

Subgroups	Target	Quality
Gender = Female and Smoker = True and Has_previous_cancer = True	Develop_Cancer = Yes	0,41
Smoker = True and Age = [60,70] and Has_previous_cancer = False	Develop_Cancer = Yes	0,37
Gender = Female and Smoker = True and Weight = [40,50]	Develop_Cancer = Yes	0,32
Smoker = True and Age = [20,30] and Has_previous_cancer = False	Develop_Cancer = Yes	0,24
Gender = Male and Smoker = True and Age = [45,60]	Develop_Cancer = Yes	0,21
Gender = Female and Smoker = False and Has_previous_cancer = False	Develop_Cancer = Yes	0,18
Gender = Male and Smoker = False and Age = [30,40]	Develop_Cancer = Yes	0,11

Figure 2.6: Pruning subgroups with static threshold value of 0.1

However, in Figure 2.7, considering a threshold value for a given quality measure of 0.4, it can be seen that we would be discarding almost the whole subgroups.

Subgroups	Target	Quality
Gender = Female and Smoker = True and Has_previous_cancer = True	Develop_Cancer = Yes	0,41
Smoker = True and Age = [60,70] and Has_previous_cancer = False	Develop_Cancer = Yes	0,37
Gender = Female and Smoker = True and Weight = [40,50]	Develop_Cancer = Yes	0,32
Smoker = True and Age = [20,30] and Has_previous_cancer = False	Develop_Cancer = Yes	0,24
Gender = Male and Smoker = True and Age = [45,60]	Develop_Cancer = Yes	0,21
Gender = Female and Smoker = False and Has_previous_cancer = False	Develop_Cancer = Yes	0,18
Gender = Male and Smoker = False and Age = [30,40]	Develop_Cancer = Yes	0,11

Figure 2.7: Pruning subgroups with static threshold value of 0.4

Finally, considering another dataset but the same threshold value of 0.4, it can be seen in Figure 2.8 that only 1 subgroup is discarded.

Subgroups	Target	Quality
Location = Boadilla and Size = 180m² and Garden = True and NºFloors = 3	Prize = [800k, 900k]	0,84
Location = Vallecas and Size = 60m² and Garden = False	Prize = [150k, 250k]	0,71
Location = Coslada and Size = 50m² and Garden = False and NºFloors = 1	Prize = [100k, 200k]	0,66
Location = N.Ministerios and Size = 70m² and Garage = True	Prize = [200k, 300k]	0,61
Location = Coslada and Size = 80m² and Garden = True	Prize = [200k, 300k]	0,5
Location = Boadilla and Size = 90m² and Garden = True and NºFloors = 2	Prize = [300k, 400k]	0,42
Location = Vallecas and Size = 70m² and Garage = False and NºRooms = 4	Prize = [400k, 500k]	0,37

Figure 2.8: Pruning subgroups with static threshold value of 0.4

Therefore, with the examples exposed, it is not clear what would be the best value to use as a threshold. Considering Figure 2.6, the threshold value has been useful since it has not been able to filter any subgroup, however in Figure 2.7, we would practically not obtain any subgroup. In summary, both the lack of knowledge of the measure to use to evaluate, and the lack of knowledge of the subgroups that are going to be generated in the process, means that establishing a value manually can result in undesirable results and therefore, needing to perform several runs with different threshold values.

Chapter 3

Algorithm

This section proposes DASSD (Dynamic and Adaptive Subgroup Set Discovery), a heuristic free parameter algorithm to discover statistically significant and non-redundant subgroups sets (The implementation of the algorithm can be found on: `https://github.com/aarongitrepos/TFM`). The workflow of the algorithm can be divided in three phases, explained in each following sections: preprocessing and transformation the data (Section 3.1), reducing the dimension complexity of the data (Section 3.2), and discover significant and non-redundant subgroups (Section 3.3).

3.1 Preprocessing and transformation of data

In this phase, the data is validated by removing those instances that have missing values in some column. After that, two transformation processes are applied in order to unify the data.

First, the type of the data is analysed. According to Section 2.1, data type can be nominal, numeric or mixed (i.e. the data presents some nominal and numeric values). In the case of numeric data, a discretization of the numeric values is performed in order to reduce and unite the number of possible values a variable can accept. Supposing a numeric variable X^i and a number of cutpoints or groups n_{cut}. The groups generated from this numeric variable are all valid subsets (they must cover at least one instance) given by equal frequency discretization with open and closed intervals for cut points. On the other hand, considering mixed type data, the same numeric discretization is carried out for the numeric data, being the value of n_{cut} computed considering the among of unique values of nominal data (i.e. supposing 3, 4 and 5 the number of unique values of three nominal variables, the mean value = 4, will be considered the value of n_{cut}).

The second transformation is related to the type of the problem presented (unitarget or multitarget). This transformation will be applied in the case of multitarget problem. For this scenario, all possible combinations are generated considering the different values of target variables. For instance, considering two target variables $Y^1 : \{a, b\}$ and $Y^2 : \{c, d\}$, the resulted target variable will be $Y^{1,2} : \{ac, ad, bc, bd\}$.

3.2 Dimension complexity reduction

The purpose of this phase is to reduce the dimension of the search space. In order to achieve that, we analyse the relevance between variables or features and the target variables. Many researches have used symmetric uncertainty (SU) to measure the correlation between features and target variables because it can accurately describe the nonlinear relationship between random variables. In [35], some definitions and concepts regarding the relevance between variables are presented and employed in this work.

- C-Relevance: It expresses the relevance between a variable X^n and the target variable Y^n, which is computed by SU(X^n,Y^n).
- S-Relevance: Considering a variable X^n and the target variable Y^n, if the value of SU(X^n,Y^n) is greater than a defined threshold value, X^n can be considered strong relevant with respect to Y^n.

So, according to the definitions showed, for a dataset S with X = $\{X^1, X^2, X^i\}$ features, the C-Relevance value of each feature SU(X^n,Y) is calculated. Subsequently, according to the definition of S-Relevance, the features whose C-Relevance are higher than a predefined threshold value are selected. In [36] different options are analysed regarding the calculation of the threshold value, setting the calculation of the threshold as follows:

$$p = min(0.1 * SU_{max}, SU_x) \tag{3.1}$$

being $x = [X/logX]$ the th ranked feature, and SU_{max} the maximal C-Relevance value among all features.

3.3 Discovering subgroups

As it was explained in Section 2.3.4, the exploration of the search space can be performed using an exhaustive or heuristic search. Considering that DASSD employs a heuristic strategy, and more specifically, the beam search strategy, three parameters are required:

- Quality measure: In order to determine the quality of the generated subgroups, DASSD relies on the Information gain measure. On the other hand, OR and OR intervals are used to detect statistically non-redundant subgroups. Finally, relevance and redundancy are also analysed in order to maximize and minimize, respectively, using the mrMR measure. Moreover, other measures such as coverage, confidence, complexity are used as auxiliary measures to help describe the subgroups obtained at the end.
- Maximum depth d_{max}: The maximum number of iterations (i.e. maximum conditions a subgroup can have) are defined as the number of features of the dataset. However, it can be possible to define a specific value for this parameter.
- Beam width bw: As it was explained in Section 2.3.4.3, this parameter can be really sensitive to manual values, resulting in different results depending on the chosen value. Furthermore, consider the fact that this specific threshold value will be employed not only in the final results to discard some solutions, but will also be employed in the generation phase, when several solutions are being

considered. In each iteration of the algorithm, this value could be too high or too low, depending on the solutions considered. This problem has similarity to that of genetic algorithms when defining a value for their crossover and mutation operators. In order to solve this issue, different works propose to employ a dynamic threshold value that is computed regarding the solutions considered at a specific iteration [27, 9].

Thus, extrapolating the idea of obtaining a dynamic value for each iteration considering different solution, we propose the use of a dynamic threshold, which can determine, regarding a set of considered subgroups, whose are the best subgroups to be considered in the following iteration. This approach allows to not discover which value provides the best results for a dataset (fine-tuning), and that it can be used with any quality measure. It can be computed as:

$$T = \sqrt{\frac{\sum_{i=1}^{n}(x_i^2 - \mu)^2}{n}} \tag{3.2}$$

where n represents the number of solutions considered, x_i represents the quality value of the solution i and μ represents the mean value of considered solutions.

Thus, considering the example presented in Figure 2.6, after the calculation of the threshold value employing Equation 3.2, resulting in a value of 0.215, it can be seen in Figure 3.1 the accepting subgroups.

Subgroups	Target	Quality
Gender = Female and Smoker = True and Has_previous_cancer = True	Develop_Cancer = Yes	0,41
Smoker = True and Age = [60,70] and Has_previous_cancer = False	Develop_Cancer = Yes	0,37
Gender = Female and Smoker = True and Weight = [40,50]	Develop_Cancer = Yes	0,32
Smoker = True and Age = [20,30] and Has_previous_cancer = False	Develop_Cancer = Yes	0,24
Gender = Male and Smoker = True and Age = [45,60]	Develop_Cancer = Yes	0,21
Gender = Female and Smoker = False and Has_previous_cancer = False	Develop_Cancer = Yes	0,18
Gender = Male and Smoker = False and Age = [30,40]	Develop_Cancer = Yes	0,11

Figure 3.1: Pruning subgroups with dynamic threshold value of 0.215

The phase of discovering subgroups can be described in two steps during each iteration of the algorithm (generation and evaluation of new subgroups and determine which generated subgroups should continue in the next iteration). Moreover, during the first step, if a subgroup s has not generated any children subgroup (i.e. finished subgroup), the algorithm will determine if this finished subgroup has to be stored as a final result.

Algorithm 1: Discovering subgroups

```
   Data: Data d, Number of iterations iter
 1 Sf ← [];
 2 B ← [filterByThreshold(Subgroups with one condition in D)];
 3 for i ← 1 to iter do
 4 |   B_aux ← [];
 5 |   for s ∈ B do
 6 |   |   S ← [Extended subgroups of s];
 7 |   |   S ← Evaluation(S, s);
 8 |   |   if size(S) == 0 then
 9 |   |   |   Sf ← Analyze(Sf, s);
10 |   |   end
11 |   |   else
12 |   |   |   B_aux.extend(S)
13 |   |   end
14 |   end
15 |   B_aux ← mrMR(B_aux);
16 |   B ← B_aux;
17 |   i ← i + 1;
18 end
19 R ← merge(Sf,B);
20 R ← mrMR(R);
21 return R;
```

Algorithm 1 presents the pseudocode of DASSD. In line 2, all the subgroups with one condition are evaluated and filtered by using the dynamic threshold with the IG measure (parent subgroups). Lines 3-17 perform the heuristic search for the required iterations. Lines 5-14 are related to the first step of the algorithm (Generation and evaluation of new subgroups). In line 6, a parent subgroup s is extended by adding new conditions. In line 7, the evaluation of these extended subgroups is performed (Section 3.3.1.1). Lines 8-13 analysed the results obtained in the evaluation phase. If all extended subgroups are discarded, subgroup s is considered a finished subgroup, and therefore it must be determined if it can be part of the final result (lines 8-10, Section 3.3.1.2). On the other hand, all extended subgroups will be considered in the next phase (Line 15, Section 3.3.2). Finally, in line 19, the merge is performed between the list of finished subgroups and the subgroups obtained in the last iteration to constitute the final result set.

3.3.1 First step

3.3.1.1 Generation and evaluation of new subgroups

Considering k number of subgroups (i.e. parents subgroups), for each parent subgroup, n number of extended subgroups (children subgroups) are obtained by adding one condition to the parent subgroup. Each children subgroup has to meet two conditions in order to be considered later:

- Information gain gradient: This condition attempts to measure the difference of Information gained by a children subgroup with respect to his parent subgroup. Considering the parent subgroup S, children subgroup s and their respective IG

values IG_S and IG_s, the information gain gradient value is calculated as:

$$g = \frac{IG_s}{IG_S} \tag{3.3}$$

thus, the Information gain gradient value of s has to be equals or higher than the Information gain gradient value of S.

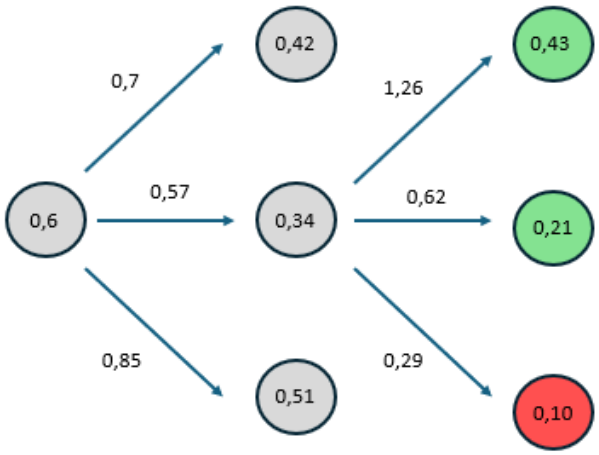


Figure 3.2: Example of IGG condition

As it can be seen in Figure3.2, two subgroups (green circles) have an IGG (1.26 and 0.62) higher than the IGG of their parent (0.57).

- Odd ratio intervals: Consider the parent subgroup S, children subgroup s, their respective OR values $OR(S)$ and $OR(s)$, and their respective OR intervals $[OR(S)_-, OR(S)_+]$ and $[OR(s)_-, OR(s)_+]$. According to the explanation showed in Section2, the $OR(s)$ has to be equals or higher than $OR(S)$ ($OR(s) \geq OR(S)$) and the lower bound $OR(s)_-$ has to be equals or higher than the upper bound $OR(S)_+$ ($OR(s)_- \geq OR(S)_+$).

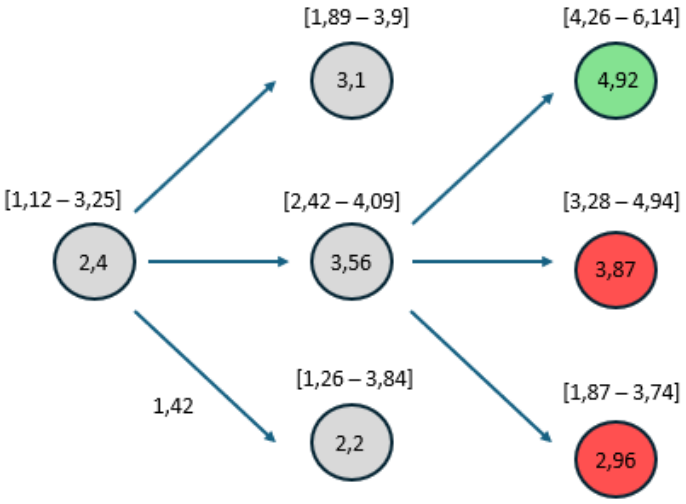


Figure 3.3: Example of ORC condition

Regarding Figure3.3, two subgroups have a OR value (4.92 and 3.87) higher than the OR value of the corresponding parent (3.56). However, only the first subgroup (green circle) has a lower bound value (4.26) higher than the upper bound value of the parent subgroup (4.09).
Consequently, by combining both conditions, only one subgroup is stored for the following step.

Algorithm 2: Evaluation

```
   Data: Generated subgroup list S, Pattern p, Iteration iter
1  R ← [];
2  S ← [∀ s ∈ S, WRAcc(s) > 0];
3  for s ∈ S do
4  |  if IGG condition == True ∧ ORC condition == True then
5  |  |  R.append(s);
6  |  end
7  end
8  return R;
```

Algorithm 2 presents the pseudocode of the evaluation phase. In line 2, subgroups with a WRAcc value lower than zero are discarded. Lines 3-7 will determine which subgroup is able to be considered a new candidate.

3.3.1.2 Analysing finished subgroups

Considering a subgroup s (i.e. parent subgroup) employed in the first task, if the subgroups generated do not meet the two conditions presented above, we call the subgroup s as a finished subgroup. Thus, this subgroup s will not be used in the following iteration, but may be a potential subgroup presented in the final results. In order to determine this task, we compare the subgroup s with other finished subgroups stored in past iterations $Sf = \{sf_1, sf_2, sf_n\}$. This comparison will be done analysing the level of redundancy and relevance between the considered subgroup s and the stored finished subgroups Sf. The procedure employed to achieve this task is the following: Order all finishes subgroups presented in Sf. Considering n the number of conditions presented in s, for each finished subgroup $sf \in Sf$, if $n-1$ conditions are shared between s and sf, the subgroup sf will be substituted by s whether $WRAcc(s) > WRAcc(sf)$. In other case, subgroup s will be discarded.

Finished subgroups	Quality
Gender = Female and Smoker = True and Has_previous_cancer = True	0,41
Smoker = True and Age = [60,70] and Has_previous_cancer = False	0,37
Gender = Female and Smoker = True and Weight = [40,50]	0,32
Smoker = True and Age = [20,30] and Has_previous_cancer = False	0,24
Gender = Male and Smoker = True and Age = [45,60]	0,21
Gender = Female and Smoker = False and Has_previous_cancer = False	0,18
Gender = Male and Smoker = False and Age = [30,40]	0,11

Figure 3.4: Example of finished subgroups comparison

For instance, consider the finished subgroups presented in Figure 3.4 and the fin-

ished subgroup s : (Smoker == True and Age == [20,30] and Location == Madrid), with $WRAcc(s) == 0.35$ and $n == 3$ (number of conditions). As it can be noticed, only the coloured subgroup sf showed in Figure3.4 shared $n-1$ conditions with s, and after the Wracc values comparison ($WRAcc(s) > WRAcc(sf)$), it is determined that s will substitute sf.

On the other hand, considering the finished subgroup s : (Gender == Female and Smoker == True and Location == Madrid), with $WRAcc(s) == 0.38$ and $n == 3$ (number of conditions), two subgroups ($sf1$ and $sf3$) presented in Figure 3.5 shared $n-1$ conditions with s. The first subgroup is not substituted because $WRAcc(sf1) > WRAcc(s)$, however, third subgroup $sf3$ will be replaced by s due to $WRAcc(sf3) < WRAcc(s)$.

Finished subgroups	Quality
Gender = Female and Smoker = True and Has_previous_cancer = True	0,41
Smoker = True and Age = [60,70] and Has_previous_cancer = False	0,37
Gender = Female and Smoker = True and Weight = [40,50]	0,32
Smoker = True and Age = [20,30] and Has_previous_cancer = False	0,24
Gender = Male and Smoker = True and Age = [45,60]	0,21
Gender = Female and Smoker = False and Has_previous_cancer = False	0,18
Gender = Male and Smoker = False and Age = [30,40]	0,11

Figure 3.5: Example of finished subgroups comparison

Algorithm 3: Analysing finished subgroups

```
   Data: Finished subgroup list Sf, Finished subgroup s, Complexity of finished
         subgroup n
 1 if size(Sf) == 0 then
 2 |  Sf ← s;
 3 end
 4 else
 5 |  for sf ∈ sortedByWracc(Sf) do
 6 |  |  if n-1 Conditions are shared between s and sf then
 7 |  |  |  if WRAcc(s) > WRAcc(sf) then
 8 |  |  |  |  Sf.remove(sf);
 9 |  |  |  |  Sf.append(s);
10 |  |  |  |  break;
11 |  |  |  end
12 |  |  end
13 |  end
14 end
15 return Sf;
```

Algorithm 3 presents the pseudocode for the analysis of finished subgroups. In lines 5-13, each subgroup sf contained in the finished subgroup list is compared with the considered subgroup s until the condition is reached. When a subgroup sf shared $n-1$ conditions with subgroup s (line 6), the WRAcc comparison is performed between them (line 7). If $WRAcc(s) > WRAcc(sf)$, subgroup s will substitute subgroup sf (lines

8-9).

3.3.2 Second step

Considering k number of subgroup generated in the first step, the algorithm evaluates which subgroups will pass to the following iteration (i.e. will be the parents subgroups for the following iteration). To perform the task of evaluating the subgroups, we rely on the mrMR filter method (Section 2.2.1) due to its ability to measure both relevance and redundancy characteristics. Considering the context of SD, it was decided to employ the WRAcc measure to define the relevance of each subgroup, meanwhile the redundancy of the subgroups set is measure using the redundancy measure showed in Section. Furthermore, for discarding subgroups, the dynamic threshold will be used employing the computed mrMR values (An example of how dynamic threshold works is shown in Section 3.3).

Algorithm 4: Determine which subgroups pass to next iteration

```
   Data: Subgroup list S
1  L ← [];
2  for s ∈ (S) do
3  |  mrMR(s) ← WRAcc(s) / Redundancy(s);
4  |  L.append(mrMR(s));
5  end
6  R ← [filterByThreshold(L)];
7  return R;
```

Algorithm 4 showed the pseudocode of the second step. Computation of the mrMR value (line 3) is done for each subgroup obtained during the first step (lines 2-5). After that, employing the dynamic threshold, the algorithm will determine which subgroups must be considered the new parent subgroups for the next iteration (line 6).

Chapter 4

Results

This section presents the performance of the DASSD algorithm proposed in Chapter 3, employing the seven datasets shown in Section 2.1. Furthermore, we analyse the difference between using all the features presented in the datasets and reducing the number of features in the datasets with the Feature Selection technique (Section 4.4). Finally, we have compared DASSD with three algorithms presented in the State-of-the-Art. To compare the differences between subgroup lists and sets, we have decided to use the FSSD and SSD++ algorithms. Moreover, the MCTS4DM (i.e MCT) algorithm is used to compare different heuristics, such as beam and MCTS in the subgroup set approach. In order to describe the performance of each algorithm, seven measures presented in Section 2.3.4.2 are used, additionally considering the execution time (seconds format).

4.1 Feature Selection vs non-Feature Selection

In this section, we provide an analysis of both approaches of the DASSD algorithm. In the results displayed below, DASSD refers to non-Feature Selection (i.e. non-FS) approach, while DASSD_FS refers to the Feature Selection (i.e. FS) approach.

	time	size	length	redundancy	wracc	OR	coverage	confidence
DASSD	68.13	20.85	2.19	**0.13**	0.04	3.31	**0.23**	0.75
DASSD_FS	**15.48**	**22.81**	**2.53**	0.17	**0.05**	**3.5**	0.19	**0.8**

Figure 4.1: Statistical comparison between FS and non-FS approaches.

Table 4.1 shows the obtained values for each measure, using a color range (green, yellow, red) to indicate the best and worst value for each column.

- The highest difference between both approaches is the execution time. Meanwhile, DASSD requires around 68 seconds to finish the execution, DASSD_FS needs around 15 seconds to finish the execution. This represents a decrease of $\sim$ 77.27%.
- Regarding the number of obtained subgroups and their complexity (i.e. length), DASSD_FS provides slightly more subgroups with more information than the

non-FS approach. However, this leads to slightly more redundancy (0.17 vs 0.13) in the result set.

- Regarding the number of instances covered by the subgroups, due to the lower number of conditions, DASSD returns a higher coverage value than DASSD_FS (0.23 - 0.19). However, regarding the confidence and WRAcc values, the FS approach returns slightly better results (0.8 - 0.75) and (0.05 - 0.04) respectively; as well as a higher OR value (3.5 - 3.31), which indicates a higher dependence between the subgroups and the target.

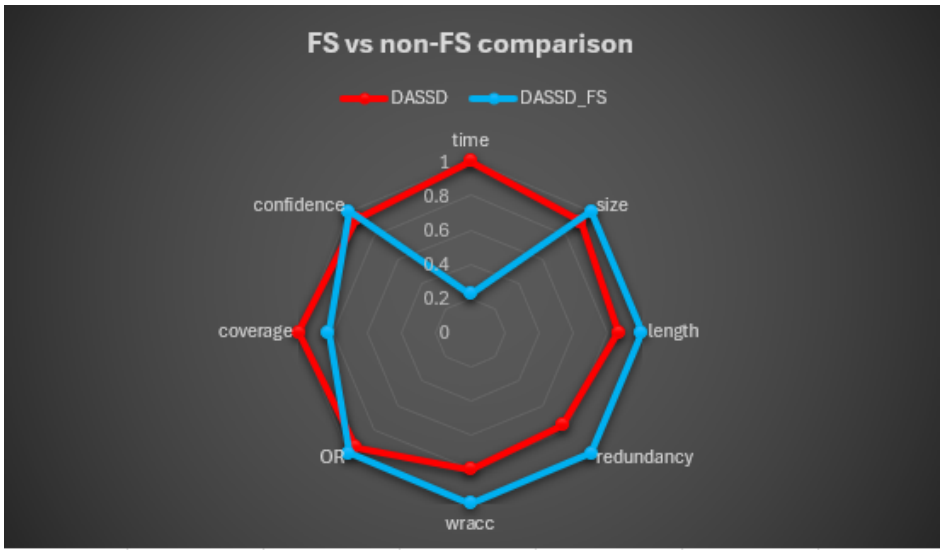


Figure 4.2: Performace summary

It can be seen in Figure 4.2 a summary of the performance of both approaches. As it has been analysed above, DASSD_FS not only provides better results in statistical terms, but also achieves a significant reduction in execution time. This is because the dimensionality of the search space is reduced by having detected and eliminated those redundant or irrelevant variables.

4.2 Comparison with SotA algorithms

In this section we provide a comparison and analysis of the DASSD_FS algorithm with three algorithms presented in the State-of-the-Art (FSSD, SSD++ and MCTS4DM (i.e. MCT)). The execution of these three algorithms was done with the default parameters presented in their respective implementations. Furthermore, one hour was considered as the maximum time to complete the execution. Regarding DASSD_FS algorithm, the FS approach is used to perform this comparison due to the better performance shown in the previous section. Firstly, in Section 4.2.1 we discuss the performance of each algorithm in each dataset, considering the different quality measures. Subsequently, in Section 4.2.2, we analyse the performance of each algorithm from a generic point of view, encompassing the results seen previously.

4.2.1 Individual comparison

Figure 4.3 shows the execution time of each algorithm using a colour range (green, yellow, red) to indicate the best and worst value. It can be seen that DASSD_FS is the algorithm which the lowest execution time. However, Coil2000 dataset shows that DASSD_FS required more time than the others. On the other hand, it can be seen that FSSD required much more time to finish the execution, specially considering the Chess dataset, 36 minutes was required to finish the execution. Furthermore, it is important to notice that DASSD_FS, SSD++ and MCTS4DM were able to finish the execution of the seven datasets considering the time condition (one hour), however, FFSD was only able to finish the execution of four datasets (Mushrooms, Splice, Chess and Adult datasets) in one hour.

	DASSD_FS	SSD++	MCT	FSSD
Mushrooms	2.84	2.56	8.217	3.589
Splice	1.23	29.82	4.62	271.36
Tuandromd	5.9	21.31	10.77	-
Coil2000	40.05	13.75	9.05	-
Chess	0.197	4.88	7.04	2253.51
Connect4	30.38	396	55.35	-
Adult	27.76	44.6	55.35	117

Figure 4.3: Time comparison between algorithms.

Figure 4.4 shows the redundancy level of the obtained subgroups. FSSD reports the highest redundancy in comparison with the other algorithms. On the other hand, DASSD_FS is able to discover subgroups with an acceptable redundancy level, although considering Tuandromd and Adult datasets, SSD++ and MCTS4DM algorithms report considerably lower redundancy between subgroups.

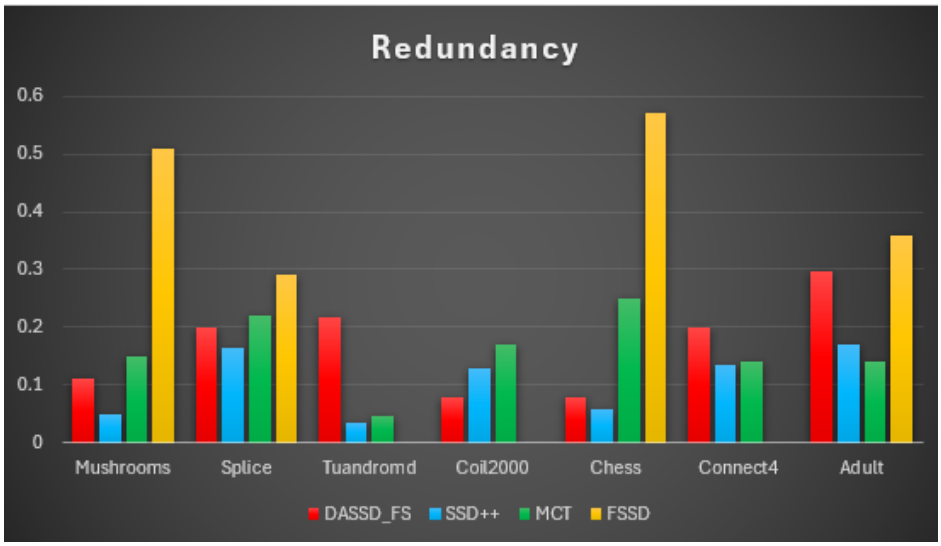


Figure 4.4: Redundancy comparison between algorithms.

Figure 4.5 shows the average coverage value of the obtained subgroups. It can be seen that DASSD_FS and MCTS4DM discover subgroups with similar amount of individuals. However, regarding Tuamdromd dataset, DASSD_FS discovered subgroups with

a more considerable representation than MCTS4DM subgroups, meanwhile, concerning Connect4 dataset, MCTS4DM discovered subgroups can covered more instances than DASSD_FS. Furthermore, SSD++ was able to discover subgroups with high representation in Mushrooms and Chess datasets, but, with respect to the other five datasets, discovered subgroups cover few instances of the datasets. On the other hand, FSSD subgroups covered few instances compared to the other algorithms.

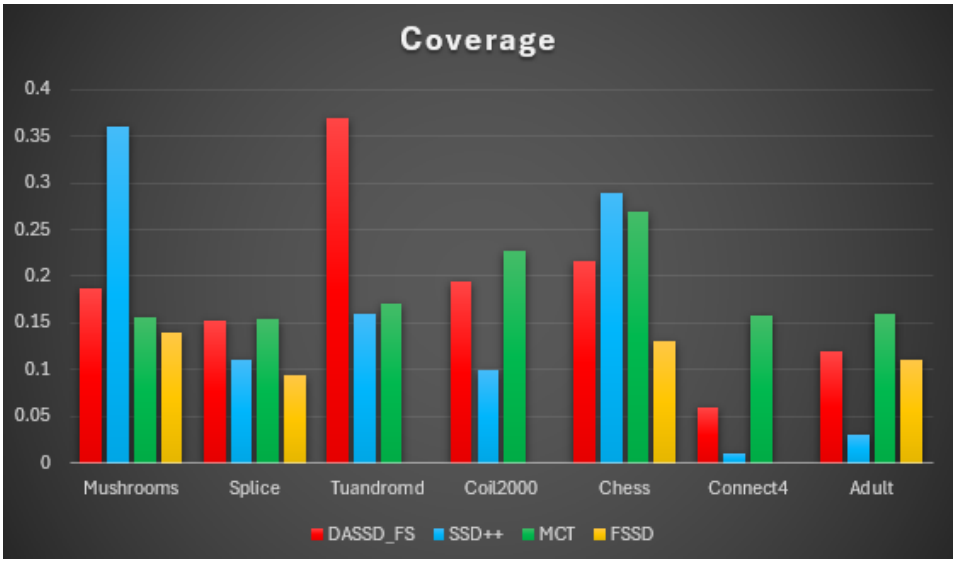


Figure 4.5: Coverage comparison between algorithms.

Figure 4.6 shows the average confidence value of the obtained subgroups. DASSD_FS and FSSD are able to discover subgroups with high number of positives individuals. However, as it was explained before, FSSD could only report results on four datasets. Furthermore, regarding the Connect4 dataset, although the three algorithms (i.e. DASSD_FS, MCTS4DM and SSD++) had problems to discover subgroups with high number of positives, where the individuals present in the subgroups have around a 50% chance of presenting the specified target, SSD++ has shown slightly worse performance than DASSD_FS and MCTS4DM algorithms.

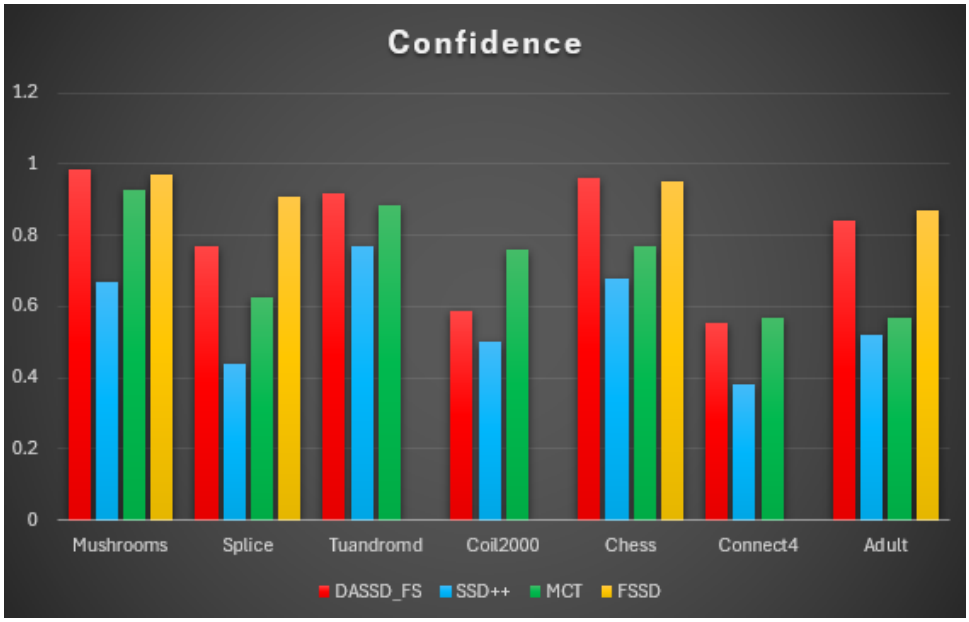


Figure 4.6: Confidence comparison between algorithms.

Figure 4.7 shows the average WRAcc value of the obtained subgroups. It can be seen that DASSD_FS is able to discover subgroups with more relevance than the other approaches. On the other hand, not only SSD++ seems to have problems obtaining subgroups with an acceptable WRAcc value, but it also shows a WRAcc value of zero in the Coil2000 dataset.

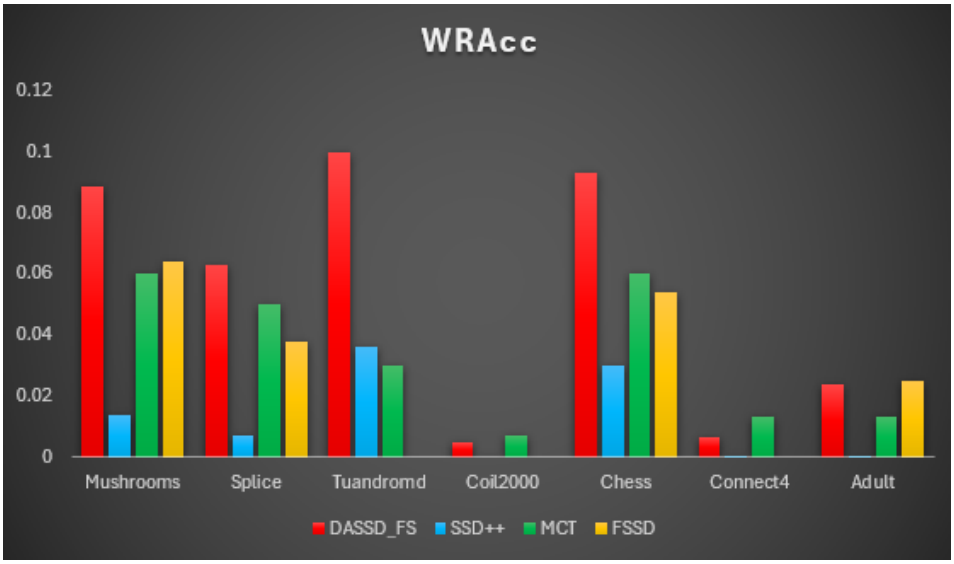


Figure 4.7: WRAcc comparison between algorithms.

Figure 4.8 shows the average OR value of the obtained subgroups. It can be seen that DASSD_FS and FSSD are able to discover subgroups with high dependence with the target required. However, as it was explained before, FSSD could only report results on four datasets. On the other hand, SSD++ and MCTS4DM shows similar OR values, although considering Mushrooms and Splice datasets, MCTS4DM obtained

subgroups with slightly more dependence than SSD++.

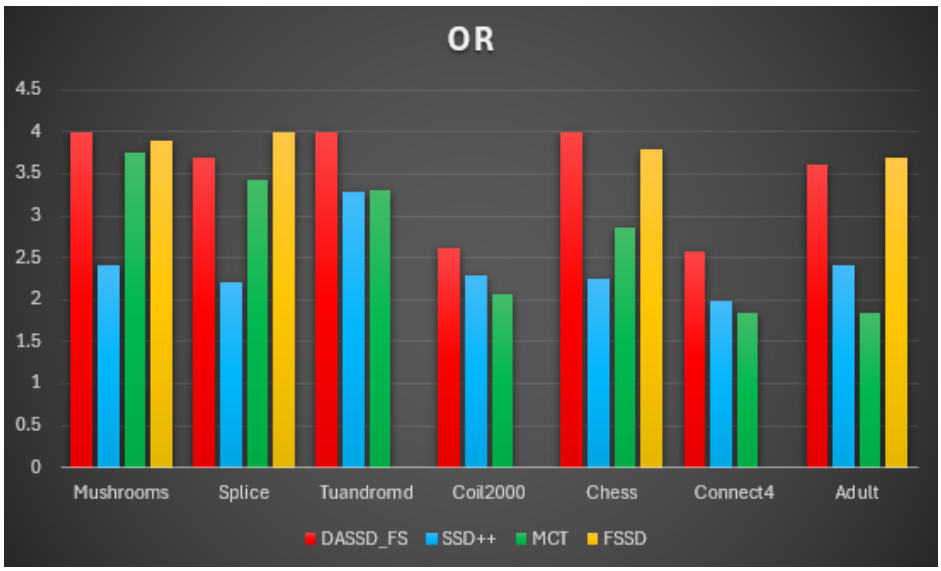


Figure 4.8: Odd ratio comparison between algorithms.

4.2.2 Global comparion

After the comparison of the performance of each algorithm in the seven datasets, in this section results are encompassing in order to analyse the average performance of each algorithm. As it was explained in the Section 4.2.1, FSSD algorithm was only able to provide results for four datasets, thus, two different results (non considering FSSD and considering FSSD in the four commented datasets) are provided in the following section.

4.2.2.1 Non considering FSSD

	time	size	length	redundancy	wracc	OR	coverage	confidence
DASSD_FS	**15.48**	22.81	2.53	0.17	**0.05**	**3.5**	**0.19**	**0.8**
SSD++	73.27	187.86	2.59	**0.11**	0.01	2.41	0.15	0.57
MCT	21.49	50	2.8	0.16	0.03	2.73	**0.19**	0.73

Figure 4.9: Statistical comparison between algorithms.

Table 4.9 shows the obtained values for each measure, using a colour range (green, yellow, red) to indicate the best and worst value for each column. It is importance to notice that size and complexity measures are not evaluated with a scale colour due to the interpretation may be subjective.

- Considering the execution time, SSD++ requires much more time (73 seconds) to finish than DASSD_FS and MCTS4DM (15 and 21 seconds).
- Regarding the number of obtained subgroups, SSD++ produce a huge amount of subgroups compared to the number of subgroups discovered by DASSD_FS (187 and 22). On the other hand, MCTS4DM has produced the number of subgroups regarding the manual set value of the beamwidth parameter (50). Furthermore,

pattern complexity (i.e. length) seems to be very similar among the algorithms. Moreover, considering the redundancy level, DASSD_FS and MCTS4DM produce subgroups sets with a similar level of redundancy. On the other hand, SSD++ produces subgroups with lower redundancy (0.11).

- Regarding the number of instances covered by the subgroups, DASSD_FS and MCTS4DM produces subgroups with more coverage than SSD++ (0.19 and 0.15). On the other hand, the subgroups discovered by DASSD_FS and MCTS4DM report a high value of confidence (0.8 and 0.73), thus leading to a high number of positives individuals, unlike SSD++ which reports a confidence value of 57%. Furthermore, the OR values show the level of dependence between subgroups and target. DASSD_FS reports an OR value of 3.5, which indicates a high level of dependence. On the other hand, MCTS4DM and SSD++ report similar OR values (2.73 and 2.41), suggesting a moderate dependence.

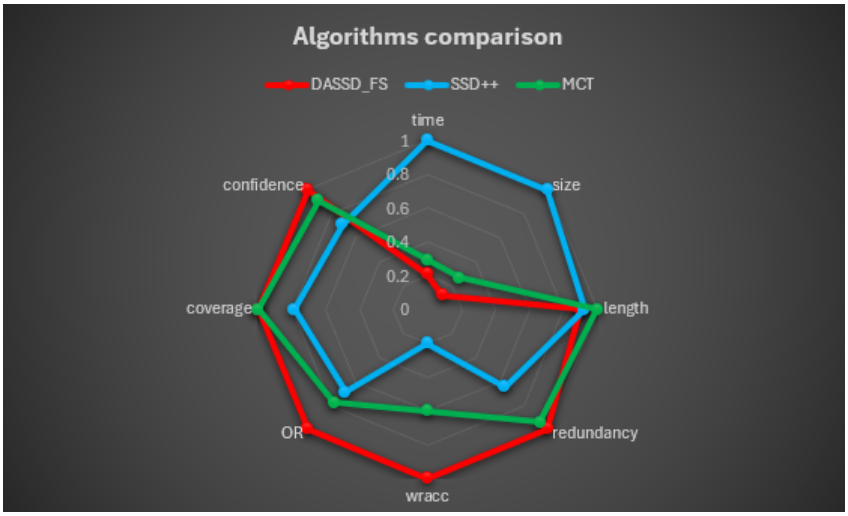


Figure 4.10: Performance summary

Figure 4.10 shows a summary of the performance of the algorithms examined. The time difference between set and list models can be related to how each approach works. In a list model, in each iteration an entire exploration is performed on the search space (perform a complete beamsearch execution) obtaining the subgroup with the best quality. Subsequently, those instances covered by said subgroup are eliminated from the dataset, repeating the process until all instances of the dataset are covered by a single subgroup. Therefore, it will be necessary to run as many beamsearch as the dataset partitions are necessary. On the other hand, in a set model, only a single beamsearch is performed, where in each iteration, the subgroups of past iterations are extended based on an optimization criterion. Regarding the redundancy among subgroups, considering that SSD++ produces subgroups list, whose main characteristic is the absence of overlapping between instances, the level of redundancy of the subgroups sets obtained with DASSD_FS and MCTS4DM are quite similar to that obtained with SSD++. On the other hand, considering the number of covered and positives instances, list models tends to have fewer covered instances as dataset dimension is reducing in each iteration. However, not only generated subgroups have less instances with the passing of iterations, but they also show that the number of positives instances presented in subgroups are lower than

the results provided by set models. Moreover, WRAcc measure provides interesting information of how the tradeoff works between coverage and confidence. Considering set models (DASSD_FS and MCTS4DM), coverage and confidence values are quite similar. However, WRAcc values show a difference of 50%. This is caused because WRAcc measure can be negative when the number of positives instances presented in a subgroup are lower than the number of positives instances presented in the dataset (confidence ratio). Thus, subgroups with negative WRAcc values negatively affects the model results. Due to DASSD_FS is able to discard subgroups with WRAcc ≤ 0, it can be assured that generated subgroups have higher proportion of positives instances than the dataset. This assumption is also supported by the OR values. It shows that the use of the OR condition to determine the relevance of the subgroups obtained provides statistically relevant subgroups with strong dependence between antecedents and target value.

4.2.2.2 Considering FSSD

	time	size	length	redundancy	wracc	OR	coverage	confidence
DASSD_FS	**8.01**	13.92	2.29	0.17	**0.07**	3.83	0.17	0.89
SSD++	20.47	65.75	2.14	**0.11**	0.01	2.33	**0.2**	0.58
MCT	18.81	50	2.86	0.19	0.05	2.97	0.19	0.72
FSSD	661.36	4.46	10.92	0.43	0.05	**3.85**	0.12	**0.93**

Figure 4.11: Statistical comparison between algorithms.

Table 4.11 shows the obtained values for each measure, using a colour range (green, yellow, red) to indicate the best and worst value for each column. It is importance to notice that size and complexity measures are not evaluated with a scale colour due to the interpretation may be subjective.

- Considering the execution time, FSSD requires around 11 minutes to finish the subgroup discovery task, meanwhile the other algorithms required less than a minute. Furthermore, DASSD_FS could finish the execution in half of the time than SSD++ and MCTS4DM.
- Regarding the number of obtained subgroups and their complexity (i.e. length), SSD++ and MCTS4DM produce more amount of subgroups (65 and 50) compare to the number of subgroups discovered by DASSD_FS (14). Moreover, considering the redundancy level, DASSD_FS and MCTS4DM produce subgroups sets with a similar level of redundancy (0.17 and 0.19), being slightly higher than the redundancy level reported by SSD++ (0.11). On the other hand, although FSSD is a list model as well as SSD++, high redundancy is presented in the resulting list. This is due to the fact that FSSD subgroups have much more variables or information (10.92) than other algorithms ($\sim$ 2), thus having more shared variables and triplets.
- Regarding the number of instances covered by the subgroups, FSSD produces subgroups with less coverage (0.12) than the other algorithms, being quite similar between them. Furthermore, the subgroups discovered by DASSD_FS and FSSD report a great value of confidence (0.89 and 0.93), thus having a high number of positives individuals. This is also supported by the high OR val-

ues (3.83 and 3.85 respectively), which indicates a strong dependence between the subgroup and the target. On the other hand, MCTS4DM reports slightly lower confidence and OR values (0.72 and 2.97), which indicates a moderate dependence. Finally, SSD++ reports a confidence values less than 60%, which, combined with an OR value of 2.33, indicates a lower dependence with the target.

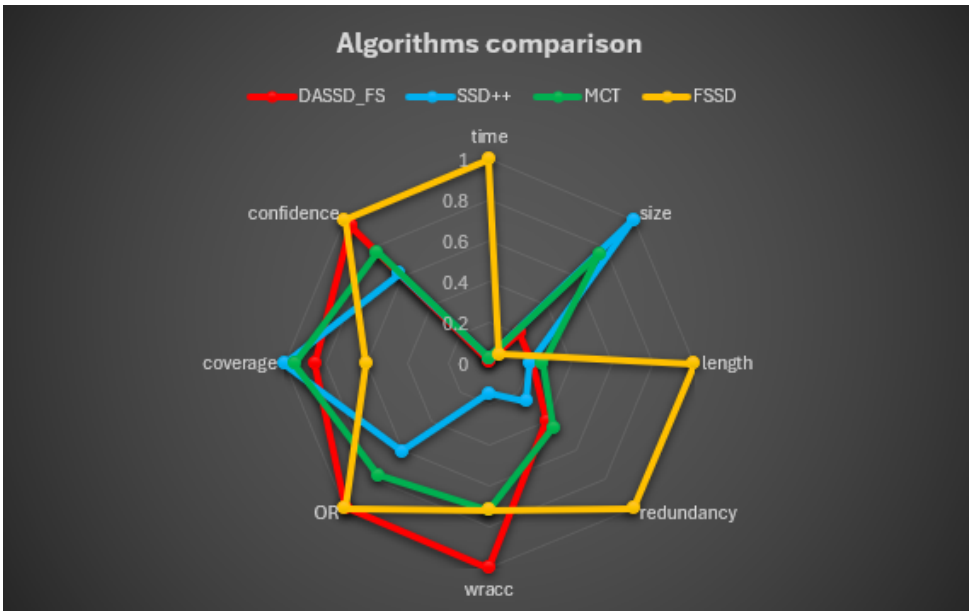


Figure 4.12: Performance summary

Figure 4.12 shows a summary of the performance of the algorithms examined, considering four datasets. The time difference between FSSD and the other algorithms is not only related to the fact that it is a list model. Due to the heuristic employed in FSSD, whose objective is to find in each iteration the subgroup that maximizes a measure (WRAcc) using an optimistic estimate in a branch-and-bound scheme, it will be necessary to expand the considered subgroups until the metric is no longer improved. This will allow obtaining subgroups with many variables (i.e. length of subgroups), so shared information (i.e. variables or triplets) may appear, causing much redundancy among the subgroup list. Moreover, regarding the DASSD_FS results, it can be seen that the performance is similar as the previous analysis (Section 4.2.2.1). However, it is important to notice that although DASSD_FS is not using the WRAcc measure as an optimization criterion during the exploration, the proposed heuristic allows to not only discover high quality subgroups, but it also provides the best WRAcc values in comparison with all the algorithms, even considering FSSD goal is to maximize the WRAcc measure. This behaviour can be understood analysing the coverage and confidence measures. Considering FSSD, the maximization of WRAcc allows obtaining subgroups with a high rate of positive instances and dependence on the target (i.e. high confidence and OR values), but with a lower coverage value because in each iteration the number of instances in the dataset are reduced and subgroups with a greater number of variables imply a lower number of instances covered. Consequently, by comparing the confidence and coverage values reported, it can be seen that although FSSD and DASSD_FS confidence values are similar, FSSD coverage value is slightly lower than DASSD_FS coverage value. Therefore, because

there is no trade-off between confidence and coverage, DASSD_FS is able to obtain better WRAcc values than FSSD.

Chapter 5

Conclusions and Future work

State-of-the-Art subgroup discovery algorithms propose different ways of discovering the search space (exhaustive search or heuristic search), to evaluate and discard the subgroups that are discovered. On the other hand, as it have been explained, these algorithms can focus on the task of discovering ordered subgroups (lists) or unordered subgroups (sets). Considering the approach of subgroups sets, one of the main limitations that exists is the degree of overlapping that can exist between the resulting subgroups. Considering the heuristic search, it has been shown the limitations that this search may have, regarding the parameters necessary for its operation. The impact that manual assignment of values for these parameters can have has been analysed, as well as the use of a single measure to evaluate subgroups may not be very effective.

Thus, in this work, the DASSD (Dynamic and Adaptable Subgroup Set Discovery) algorithm has been proposed, which does not require fine-tuning of parameters, and can be employed for both nominal and numerical variables.

Among the main characteristics of DASSD it stands out: in each iteration, through the use of two proposed filters based on the IG and the OR measures, DASSD is able to detect when it is not necessary to continue exploring a subgroup (i.e. continue expanding the subgroup by adding new features), or on the other hand, which features improve the characteristics of a subgroup. Furthermore, DASSD is capable of filtering those subgroups which have a lower impact (relevance) and/or present a high degree of repeated information dynamically through a threshold adaptable to the set of subgroups considered. In this way, DASSD is able to adapt to any dataset and discover an optimal set of subgroups maximizing impact and relevance, but minimizing redundancy. Furthermore, before using DASSD to discover relevant subgroups, it has been analysed the impact that reducing the dimensionality of features by using the Feature Selection technique can have on the search, since a reduction of features would limit the search space, allowing for an execution time reduction due to only those features that have greater relevance in the dataset will be considered. In Section 4 we have analysed the performance of the proposed algorithm together with three algorithms present in the State-of-the-Art, using seven datasets with different characteristics. Firstly, it is observed that the use of Feature Selection to reduce the number of features to be considered in the explanatory phase allows reducing considerably the execution time, as well as the resulting subgroups are slightly better considering the different metrics. On the other hand, when analysing the performance of each algo-

rithm, we observe that DASSD is the algorithm that needs the least time to complete the Subgroup Discovery task, being able to discover not only subgroups with better statistical values, but also with reduced redundancy in the resulting set. With the results obtained, we see that the use of the dynamic threshold and the combination of several filters, without the need to specify default thresholds, yields promising results. The main lines of research for future work could be: 1) regarding the Feature Selection method, exploring the possibility of combining different filter methods or using other options such as wrapper or hybrid methods, studying the impact they may have on both the relevance of the features such as the time needed in the task of exploring the search space; 2) study the possibility of combining other measures in the proposed filters to consider the extension of subgroups; and 3) analyse the possibility of generating a search space adapted to each subgroup, thus eliminating the need to observe the whole search space in each considered subgroup.

Bibliography

[1] Martin Atzmueller. Subgroup discovery. *WIREs Data Mining and Knowledge Discovery*, 5(1):35–49, January 2015.

[2] Martin Atzmueller and Frank Puppe. SD-Map – A Fast Algorithm for Exhaustive Subgroup Discovery. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Dough Tygar, Moshe Y. Vardi, Gerhard Weikum, Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *Knowledge Discovery in Databases: PKDD 2006*, volume 4213, pages 6–17. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[3] Peter C. Austin and Janet E. Hux. A brief note on overlapping confidence intervals. *Journal of Vascular Surgery*, 36(1):194–195, July 2002.

[4] Adnene Belfodil, Aimene Belfodil, Anes Bendimerad, Philippe Lamarre, Céline Robardet, Mehdi Kaytoue, and Marc Plantevit. Fssd-a fast and efficient algorithm for subgroup set discovery. In *2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 91–99. IEEE, 2019.

[5] Guillaume Bosc, Jean-François Boulicaut, Chedy Raïssi, and Mehdi Kaytoue. Anytime discovery of a diverse set of patterns with Monte Carlo tree search. *Data Mining and Knowledge Discovery*, 32(3):604–650, May 2018.

[6] Bjorn Bringmann and Albrecht Zimmermann. The chosen few: On identifying valuable patterns. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 63–72, 2007.

[7] Peter Bugata and Peter Drotar. On some aspects of minimum redundancy maximum relevance feature selection. *Science China Information Sciences*, 63(1):112103, January 2020.

[8] Jooyoung Cho, Dong Min Seo, and Young Uh. Clinical Application of Overlapping Confidence Intervals for Monitoring Changes in Serial Clinical Chemistry Test Results. *Annals of Laboratory Medicine*, 40(3):201–208, May 2020.

[9] M. Dashtban and Mohammadali Balafar. Gene selection for microarray cancer classification using a new evolutionary method employing artificial intelligence concepts. *Genomics*, 109(2):91–107, March 2017.

[10] Sergio Alexis Dominguez-Lara. El odds ratio y su interpretación como magnitud del efecto en investigación. *Educación Médica*, 19(1):65–66, January 2018.

[11] Cyril Esnault, May-Line Gadonna, Maxence Queyrel, Alexandre Templier, and Jean-Daniel Zucker. Q-Finder: An Algorithm for Credible Subgroup Discovery in Clinical Data Analysis — An Application to the International Diabetes Management Practice Study. *Frontiers in Artificial Intelligence*, 3:559927, December 2020.

[12] François Fleuret. Fast binary feature selection with conditional mutual information. *Journal of Machine learning research*, 5(9), 2004.

[13] Johannes Fürnkranz, Dragan Gamberger, and Nada Lavrač. *Foundations of rule learning*. Springer Science & Business Media, 2012.

[14] Tias Guns, Siegfried Nijssen, and Luc De Raedt. Evaluating pattern set mining strategies in a constraint programming framework. In Joshua Zhexue Huang, Longbing Cao, and Jaideep Srivastava, editors, *Advances in Knowledge Discovery and Data Mining*, pages 382–394, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[15] Mark Hall. Correlation-based feature selection for machine learning. *Department of Computer Science*, 19, 06 2000.

[16] Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. *Data Mining and Knowledge Discovery*, 8(1):53–87, January 2004.

[17] Hanchuan Peng, Fuhui 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*, 27(8):1226–1238, August 2005.

[18] Sumyea Helal. Subgroup discovery algorithms: a survey and empirical evaluation. *Journal of computer science and technology*, 31:561–576, 2016.

[19] Franciso Herrera, Cristóbal José Carmona, Pedro González, and María José Del Jesus. An overview on subgroup discovery: foundations and applications. *Knowledge and Information Systems*, 29(3):495–525, December 2011.

[20] Branko Kavšek and Nada Lavrač. Apriori-sd: Adapting association rule learning to subgroup discovery. *Applied Artificial Intelligence*, 20(7):543–583, 2006.

[21] Arno J. Knobbe, Bruno Crémilleux, Johannes Fürnkranz, and Martin Scholz. From local patterns to global models: The lego approach to data mining. 2008.

[22] Mirjam J. Knol, Wiebe R. Pestman, and Diederick E. Grobbee. The (mis)use of overlap of confidence intervals to assess effect modification. *European Journal of Epidemiology*, 26(4):253–254, April 2011.

[23] Nada Lavrac, Branko Kavsek, Peter Flach, and Ljupco Todorovski. Subgroup discovery with cn2-sd. *J. Mach. Learn. Res.*, 5(2):153–188, 2004.

[24] Florian Lemmerich, Mathias Rohlfs, and Martin Atzmüller. Fast discovery of relevant subgroup patterns. 01 2010.

[25] Jiuyong Li, Jixue Liu, Hannu Toivonen, Kenji Satou, Youqiang Sun, and Bingyu Sun. Discovering statistically non-redundant subgroups. *Knowledge-Based Systems*, 67:315–327, September 2014.

[26] Antonio Lopez-Martinez-Carrasco, Jose M. Juarez, Manuel Campos, and Bernardo Canovas-Segura. VLSD—An Efficient Subgroup Discovery Algorithm Based on Equivalence Classes and Optimistic Estimate. *Algorithms*, 16(6):274, May 2023.

[27] Huijuan Lu, Junying Chen, Ke Yan, Qun Jin, Yu Xue, and Zhigang Gao. A hybrid feature selection algorithm for gene expression data classification. *Neurocomputing*, 256:56–62, September 2017.

[28] Marvin Meeng and Arno Knobbe. For real: a thorough look at numeric attributes in subgroup discovery. *Data Mining and Knowledge Discovery*, 35(1):158–212, 2021.

[29] E. Noda, A.A. Freitas, and H.S. Lopes. Discovering interesting prediction rules with a genetic algorithm. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, pages 1322–1329, Washington, DC, USA, 1999. IEEE.

[30] Maria Irmina Prasetiyowati, Nur Ulfa Maulidevi, and Kridanto Surendro. Determining threshold value on information gain feature selection to increase speed and prediction accuracy of random forest. *Journal of Big Data*, 8(1):84, December 2021.

[31] Hugo M Proença, Peter Grünwald, Thomas Bäck, and Matthijs van Leeuwen. Robust subgroup discovery: Discovering subgroup lists using mdl. *Data Mining and Knowledge Discovery*, 36(5):1885–1970, 2022.

[32] Luc De Raedt and Albrecht Zimmermann. *Constraint-Based Pattern Set Mining*, pages 237–248.

[33] Sergio Ramírez-Gallego, Iago Lastra, David Martínez-Rego, Verónica Bolón-Canedo, José Manuel Benítez, Francisco Herrera, and Amparo Alonso-Betanzos. Fast-mRMR: Fast Minimum Redundancy Maximum Relevance Algorithm for High-Dimensional Big Data: FAST-mRMR ALGORITHM FOR BIG DATA. *International Journal of Intelligent Systems*, 32(2):134–152, February 2017.

[34] Claude Sammut and Geoffrey I. Webb, editors. *Encyclopedia of Machine Learning and Data Mining*. Springer US, Boston, MA, 2017.

[35] Qinbao Song, Jun Ni, and Guangtao Wang. A fast clustering-based feature subset selection algorithm for high dimensional data. *Knowledge and Data Engineering, IEEE Transactions on*, 25:1 – 1, 01 2011.

[36] Xian-Fang Song, Yong Zhang, Dun-Wei Gong, and Xiao-Zhi Gao. A Fast Hybrid Feature Selection Based on Correlation-Guided Clustering and Particle Swarm Optimization for High-Dimensional Data. *IEEE Transactions on Cybernetics*, 52(9):9573–9586, September 2022.

[37] Lin Sun, Tianxiang Wang, Weiping Ding, Jiucheng Xu, and Yaojin Lin. Feature selection using Fisher score and multilabel neighborhood rough sets for multilabel classification. *Information Sciences*, 578:887–912, November 2021.

[38] Magdalena Szumilas. Explaining odds ratios. *Journal of the Canadian Academy of Child and Adolescent Psychiatry = Journal De l'Academie Canadienne De Psychiatrie De L'enfant Et De L'adolescent*, 19(3):227–229, August 2010.

[39] J.W. Tukey. *Exploratory Data Analysis*. Number v. 2 in Addison-Wesley series in behavioral science. Addison-Wesley Publishing Company, 1977.

[40] Matthijs Van Leeuwen and Arno Knobbe. Diverse subgroup set discovery. *Data Mining and Knowledge Discovery*, 25:208–242, 2012.

[41] Geoffrey I Webb. Opus: An efficient admissible algorithm for unordered search. *Journal of Artificial Intelligence Research*, 3:431–465, 1995.

[42] Lei Yu and Huan Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. volume 2, pages 856–863, 01 2003.