Częstochowa University of Technology
Faculty of Mechanical Engineering and Computer Science

Ph.D. Thesis

MSc, Eng. Krzysztof Ropiak

Adaptation of mereological granulation techniques in selected data analysis issues

Under supervision of:

Ph.D., D.Sc. Piotr Artiemjew, Assoc. prof. UWM

Częstochowa 2025

Acknowledgments

I would like to thank my promoter, dr hab. Piotr Artiemjew, prof. UWM for his great contribution to my scientific work and motivation which he is giving me.

Last, but not least, I would like to thank my immediate family, especially my wife, who constantly motivated me on the way to complete my dissertation.

Contents

Abstract 5

Streszczenie 6

I. Introduction

1. Research background 8

2. Granular computing and it's applications based on rough sets 9

2.1. Granular computing overview 9

2.2. Polkowski's standard and concept dependent granulation 10

2.2.1. Concept dependent granulation 10

2.2.2. Standard granulation 19

2.3. Layered granulation 26

2.4. Experimental part 27

2.4.1. Introduction 27

2.4.2. Datasets description 27

2.4.3. Methodology 34

2.4.4. Concept dependent granulation experiments 34

2.4.5. Standard granulation experiments 43

2.4.6. Results comparison for standard and concept dependent granulation 51

2.4.7. Classification results 57

2.5. Conclusions 80

II. In search of knowledge granulation techniques with an adaptive mechanism for determining the granulation radius

3. Homogeneous granulation 82

3.1. Method description 82

3.2. Simple example of homogeneous granulation 84

3.3. Experimental session 89

3.3.1. Methodology 89

3.3.2. Results . . . 89
3.4. Conclusions . . . 116

4. Epsilon homogeneous granulation . . . 117

4.1. Motivation . . . 117
4.2. Relevant definitions . . . 118
4.2.1. ε–modification of the standard rough inclusion . . . 118
4.2.2. Definition of homogeneous epsilon granules and granulation steps . 118
4.2.3. Metrics for granulation and classification . . . 119
4.2.4. k-NN method for evaluation of epsilon homogeneous granulation . . 119
4.2.5. Parameter estimation in kNN classifier . . . 121
4.2.6. Toy example of epsilon homogeneous granulation . . . 121
4.3. Experimental Session . . . 123
4.4. Section summary . . . 123

III. Selected applications of knowledge granulation techniques in data analysis problems

5. A Novel Ensemble Model - The Random Granular Reflections . . . 126

5.1. Selected Ensemble models - in a nut shell . . . 128
5.1.1. Bagging - ensemble of bootstraps . . . 128
5.1.2. Arcing - ensemble of bootstraps . . . 129
5.1.3. Ada-Boost with random split . . . 129
5.2. Ensemble of Random Granular Reflections . . . 129
5.3. Experimental Session . . . 130
5.3.1. Comparison with selected other methods . . . 130
5.3.2. Results for selected other decision-making systems - multiple runs . 134
5.3.3. Testing the performance of other popular classification techniques on selected data . . . 139
5.3.4. The operation of the technique on unbalanced data. . . . 143
5.3.5. A few words about the degree of homogeneity . . . 143
5.4. A few final words . . . 144

6. Missing values handling based on homogeneous granulation . . . 146

6.0.1. A set of basic strategies . . . 146
6.1. Homogeneous granulation in $* = *$ and $* = don't\ care$ cases . . . 149

6.2. The experimental session - procedures and model settings 149
6.2.1. Pseudo-code of experiments design 150
6.2.2. The results evaluation 150
6.2.3. The results discussion 150
6.3. Section summary 154

IV. Study of the influence of over and undersampling techniques on the quality of the granulation process

7. Impact of oversampling and undersampling on data granulation. 157
7.1. Introduction 157
7.2. Methodology 159
7.3. Experiments and results 162
7.3.1. Scenario 1 162
7.3.2. Scenario 2 and scenario 3 177
7.4. Conclusion 181

V. Summary

List of Figures 185

List of Tables 189

List of Algorithms 193

Bibliography 194

Abstract

The aim of the dissertation is to introduce the reader to the world of granular computing derived from Polkowski's methods in terms of rough set theory [19]. Our goal was to expand our knowledge of this particular niche of data analysis. To present our new approximation techniques for decision systems in the area of classification. In particular, we plan to show exemplary new results using known granulation methods, but also a new method that does not require parameter estimation - homogeneous granulation. Its use for missing values handling, its use in an ensemble model (a competitive technique to other ensemble models including boosting and bagging) and the epsilon variant applied to numerical data. An additional aim is to test the performance of the aforementioned granulation methods in combination with oversampling. Our methods are dedicated to reducing the size of decision-making systems while extracting the most important information - maintaining classification efficiency. In the dissertation, we will present, among others, results that were accepted or recognized in the competition - PP-RAI Contest for the Most Influential Article on Rough Sets co-authored by Polish Researchers in 2020-2021 papers [36] and [4]. In order to achieve the aim of the dissertation, the following theses have been formulated:

(i) It is possible to design a knowledge granulation method that does not require the estimation of the optimal parameter value for the granulation radius,

(ii) The use of knowledge granules can have an effective application in reinforcing classification processes in Ensemble models,

(iii) The use of knowledge granules can have wide application in various data analysis processes - including the absorption of missing values,

(iv) Oversampling and undersampling techniques affect the process of creating granular reflections of decision-making systems.

Streszczenie

Celem rozprawy jest wprowadzenie czytelnika w świat obliczeń granularnych wywodzących się z metod Polkowskiego w terminach teorii zbiorów przybliżonych [19]. Jednym z celów było poszerzenie wiedzy na temat tej szczególnej niszy analizy danych i zaprezentowanie nowych technik aproksymacji dla systemów decyzyjnych w obszarze klasyfikacji. Praca w szczególności koncentruje się na pokazaniu przykładowych nowych wyników granulacji przy użyciu znanych metod, ale także nowej metody, która nie wymaga estymacji parametrów - granulację jednorodną. Opisano wyniki jej zastosowania do obsługi brakujących wartości w danych, wykorzystania w modelu zespołowym (technika konkurencyjna do innych modeli zespołowych, w tym boosting i bagging) oraz użycia w wariancie epsilon zastosowanym do danych numerycznych. Dodatkowym celem jest przetestowanie wydajności wyżej wymienionych metod granulacji w połączeniu z nadpróbkowaniem (ang. oversampling). Przedstawione metody mają na celu zmniejszenie rozmiaru systemów decyzyjnych przy jednoczesnym wydobyciu najważniejszych informacji - zachowaniu skuteczności klasyfikacji. W rozprawie zaprezentujemy m.in. wyniki, które zostały zaakceptowane lub wyróżnione w konkursie PP-RAI Contest for the Most Influential Article on Rough Sets co-authored by Polish Researchers in 2020-2021 w publikacjach [36] i [4]. Aby osiągnąć cel dysertacji, sformułowano następujące tezy:

(i) Możliwe jest zaprojektowanie metody granulacji wiedzy nie wymagającej szacowania optymalnej wartości parametru promienia granulacji,

(ii) Zastosowanie granul wiedzy może mieć skuteczne zastosowanie we wzmacnianiu procesów klasyfikacji w modelach zespołowych (ang. ensemble),

(iii) Zastosowanie granul wiedzy może mieć szerokie zastosowanie w różnych procesach analizy danych - w tym w obsłudze wartości brakujących,

(iv) Techniki oversamplingu i undersamplingu wpływają na proces tworzenia granularnych refleksji systemów decyzyjnych.

Part I

Introduction

1. Research background

The theoretical foundations of this dissertation are in the area of rough set theory, proposed by Z. Pawlak [19]. The considerations that form the foundation of the theory of granular computing - which forms the basis of the methods used in the dissertation - refer to works devoted to rough mereology, see [31, 32].
The theoretical part, closely related to the application layer, then introduces the issues of granular computing, framed in terms of the granulation scheme proposed by L. Polkowski, see [24, 25].
The next part of the dissertation introduces the concept of granular computing and techniques for its use in the context of rough set theory.

2. Granular computing and it's applications based on rough sets

2.1. Granular computing overview

Granular computing is not a specific method of processing data, it is a set of paradigms that define what a granule can be in terms of structures, similarities and patterns found in the data for different levels of granularity.
Granular computing can be based on various strategies, including the already mentioned fuzzy sets [46] and rough sets [44],[21], [22] , but also based on shadowed sets [20] or techniques related to data clustering [23], [42].
Lotfi A. Zadeh defined granules and granulation with these words:

> *Informally, granulation of an object A results in a collection of granules of A, with a granule being a clump of objects (or points) which are drawn together by indiscernibility, similarity, proximity or functionality. In this sense, the granules of a human body are the head, neck, arms, chest, etc. In turn, the granules of a head are the forehead, cheeks, nose, ears, eyes, hair, etc. In general, granulation is hierarchical in nature. A familiar example is granulation of time into years, years in months, months into days and so on.*
>
> *L.A. Zadeh in [47]*

In this dissertation, the concept of granularity will refer to granularity based on rough inclusions [24, 25, 26].

2.2. Polkowski's standard and concept dependent granulation

2.2.1. Concept dependent granulation

The concept dependent granulation method described below proceeds analogously to standard granulation [24, 25] except that granules are formed within the same decision class.

The following steps describe the concept dependent granulation algorithm.

1. Loading the original decision system (U - universe of objects, A - non decision attributes, d - decision attribute).
2. Specifying the radius of granulation r_{gran}. Let the $u, v \epsilon U$.
3. For each object u we analyze all objects v, looking at the attributes from A, we form a granule g^{cd} with center with object u, assuming that

$$v \epsilon g^{cd}_{r_{gran}}(u) \Leftrightarrow \mu(v, u, r_{gran}) \wedge (d(u) = d(v)), \tag{2.1}$$

   i.e.

$$v \epsilon g^{cd}_{r_{gran}} \leftrightarrow \left(\frac{|IND(u,v)|}{|A|} \geq r_{gran} \right) \wedge (d(u) = d(v)), \tag{2.2}$$

   then

$$g^{cd}_{r_{gran}}(u) = \left\{ v \ : \ \left(\frac{|IND(u,v)|}{|A|} \geq r_{gran} \right) \wedge (d(u) = d(v)) \right\} \tag{2.3}$$

   μ is a rough inclusion, formally derived from Lukasiewicz's t-norm [51].
4. We create granular coverage of the original decision system in one of the ways:
   - hierarchical coverage (granules are selected by sequence)
   - random selection of granules,
   - selecting granules with minimal, mean or maximal length,
   - selecting granules that convey the least, most or average number of new objects, respectively,
   - random selection of granules depending on concept size.

   Whether a granule is in the coverage set depends on whether it passes at least one new object.

   The original decision system is considered covered when the unique set of objects derived from the coverage granules overlaps with the entire original set

Table 2.1: The dataset used to demonstrate an example of how the concept-dependent granulation algorithm works.

	sepal_length	sepal_width	petal_length	petal_width	iris class
1	5.1	3.5	1.4	0.2	1
2	4.9	3.0	1.4	0.2	1
3	4.7	3.2	1.3	0.2	1
4	4.6	3.1	1.5	0.2	1
5	5.0	3.6	1.4	0.2	1
6	7.0	3.2	4.7	1.4	2
7	6.4	3.2	4.5	1.5	2
8	6.9	3.1	4.9	1.5	2
9	5.5	2.3	4.0	1.3	2
10	6.5	2.8	4.6	1.5	2
11	6.3	3.3	6.0	2.5	3
12	5.8	2.7	5.1	1.9	3
13	7.1	3.0	5.9	2.1	3
14	6.3	2.9	5.6	1.8	3
15	6.5	3.0	5.8	2.2	3

of objects. It means that granules with center of u meet the condition:

$$\bigcup\{g^{cd}_{r_{gran}}(u) \ : \ g^{cd}_{r_{gran}}(u) \in U_{cover}\} = U. \tag{2.4}$$

where U_{cover} denotes the set of granular coverage.

5. All objects in each granule are voting through a *majority voting* function which is used to select a representative new object. All ties are resolved by random choice. After all granules are processed, a new granular decision system is formed.

A toy example of the concept-dependent granulation

The iris dataset was chosen as the base for this example. To make the whole algorithm more comprehensible and easier to present in this document, only the first five objects from each of the three decision-making classes were selected. This slice of the dataset is shown in table 2.1.

Step 1: forming granules.

As you can see, the selected dataset contains three decision-making classes and each object contains four descriptive attributes. This means that the granulation process will be carried out for four granulation radii. In order to present a special

case of granulation, it will also be carried out for radius zero, which means that we treat all objects as indiscernible objects, which in practice will result in the creation of a single object in the reflection set for a given decision class, which will contain the most frequent attribute value at a given position (according to the principle of majority voting).

An auxiliary step in the granulation process (especially in the implementation phase) can be the creation of so-called indiscernibility matrices, which will be created for each granulation radius and, in the case of concept dependent granulation, for each decision class separately. Considering our case, we will obtain 15 such matrices (4 + 1 granulation radii * 3 unique decision classes). Due to the rather large number of them, only the matrices for the decision class with a value of 1 are presented below.

Table 2.2: Indiscernibility matrix for radius 0/4(special case) and radius 1/4, concept dependent granulation.

	u_1	u_2	u_3	u_4	u_5
u_1	1	1	1	1	1
u_2	1	1	1	1	1
u_3	1	1	1	1	1
u_4	1	1	1	1	1
u_5	1	1	1	1	1

Table 2.3: Indiscernibility matrix for radius 2/4, concept dependent granulation.

	u_1	u_2	u_3	u_4	u_5
u_1	1	1	0	0	1
u_2	1	1	0	0	1
u_3	0	0	1	0	0
u_4	0	0	0	1	0
u_5	1	1	0	0	1

Table 2.4: Indiscernibility matrix for radius 3/4 and 4/4, concept dependent granulation.

	u_1	u_2	u_3	u_4	u_5
u_1	1	0	0	0	0
u_2	0	1	0	0	0
u_3	0	0	1	0	0
u_4	0	0	0	1	0
u_5	0	0	0	0	1

Each indiscernibility matrix contains information about objects similar to each other to a given degree and in a given decision class. Considering the case of a 1/4 granularity radius and its indiscernibility matrix, we will find values of 1 at the intersection of objects that have at least one attribute with the same value. This will allow us in subsequent granulation steps to find objects similar to each other and form granules.

Similarly, in the matrix for radius 2/4, we will find values of 1 at the intersections of objects that have two attributes with the same values. This means that they are indiscernible from each other in degree 2. In this case, the objects are u_1, u_2 and u_5. We can also note that within this decision class (concept) there are no objects similar to each other in degree 3 and degree 4.

Granules are collections of objects similar to each other, which will be formed for each granulation radius separately in the granulation process. Granules are formed around a given central object, which we can, by analogy, imagine as a process similar to the well-known clustering methods. Each row from the indiscernibility matrix above represents a single granule, around a given object for a given radius of granulation. Formally, we will write the granules for a radius of 2/4 as follows:

g(u_1): {u_1, u_2, u_5}, g(u_2): {u_1, u_2, u_5}, g(u_3): {u_3}, g(u_4): {u_4}, g(u_5): {u_1, u_2, u_5}

Since listing all granules for each granulation radius and decision concept would take up too much space, the view for a single concept (decision class = 1), for each of the five granulation radii, is presented below.

Granules for radius 0/4 and 1/4

g(u_i): {u_1, u_2, u_3, u_4, u_5}, $for\ i = 1, ..., 5$

Granules for radius 2/4

g(u_1): {u_1, u_2, u_5}, g(u_2): {u_1, u_2, u_5}, g(u_3): {u_3}, g(u_4): {u_4}, g(u_5): {u_1, u_2, u_5}

Granules for radius 3/4 and 4/4

g(u_1): {u_1}, $for i = 1, ..., 5$

As can be observed, as the radius of granulation increases, the number of objects of individual granules decreases, which means that the set is moderately diverse and contains objects similar to each other (or otherwise indiscernible) in the case of selected 1 or 2 attributes, but comparing already 3 or 4 attributes objects are unique in the set.

Step 2: covering original dataset with granules.

Granular coverage is a set of granules whose unique set of objects will cover 100% of the set from which the granules originated. There are many strategies for covering sets, which are listed above in this subsection on page 10 in point 4.
To better compare the results with other granulation methods, a hierarchical coverage method was used, which means going through the granules in search of new objects in the order of their origin (that is, also according to the original order of objects in the initial set). Once again, we will use granules for radius 2/4.
We iterate through the granules and add them to the coverage set (the granules, not the objects themselves) insofar as they provide objects that are not already in the set:

Initialization

coverage = $\emptyset$

Iteration 1

We consider the granule **g(**u_1**): {**u_1, u_2, u_5**}**, which provides new objects, so it will be added to the coverage.

So currently **coverage = {g(**u_1**)} : {**u_1, u_2, u_5**}**

Iteration 2

We consider the granule **g(**u_2**): {**u_1, u_2, u_5**}**, which does not provide any new object, we skip it.

Iteration 3

We consider granule **g(**u_3**): {**u_3**}**, which provides a new object {u3}.

So currently **coverage = {g(**u_1**), g(**u_3**)} : {**u_1, u_2, u_3, u_5**}**

Iteration 4

We consider the granule **g(u_4): {u_4}**, which provides a new object {u4}.
So currently **coverage = {g(u_1), g(u_3), g(u_4)} : {u_1, u_2, u_3, u_4, u_5}**
We have reached the convergence of the initial set of objects with the set of coverage, so we abort the possible further iteration.

Step 3: creating reflection dataset from coverage.

This stage involves selecting representatives of the coverage set that will form the reflection set. This involves collecting all the occurrences of objects from the previously selected granules, and then selecting one representative (object) for each granule. Let's consider an example using the set created in the previous step.
So we have **coverage = {g(u_1), g(u_3), g(u_4)}**.

Initialization

reflection = $\emptyset$

Outer iteration 1

We take the granule g(u_1) containing the objects {u_1, u_2, u_5}, which when put together look as follows:

	sepal_length	sepal_width	petal_length	petal_width	iris class
1	5.1	3.5	1.4	0.2	1
2	4.9	3.0	1.4	0.2	1
5	5.0	3.6	1.4	0.2	1

At this stage, through the majority voting mechanism, the value of each attribute will be determined. If there are multiple equal most frequent attribute values, such a tie will be resolved randomly. Also taking into account the fact that the value of each attribute is determined independently, there is often a situation where a new object is created, which in the exact combination of attributes may not exist in the original data.

Inner iteration 1

In our case, it looks as follows: sepal_length = {5.1, 4.9, 5.0}
The cardinality is identical for each value, so one of the values is selected by a draw. Let's assume that the value 5.1 is drawn.

new_object = {5.1, , , , 1}

Inner iteration 2

The situation with the next attribute is similar, let's assume that the value 3.6 was drawn new_object = {5.1, 3.6, , , 1}

Inner iteration 2 and 3

For the next two, the situation is obvious, since in both cases the attribute values are identical for each object.

So in the end we have new_object = {5.1, 3.6, 1.4, 0.2, 1}, which is a single representation of this granule.

reflection = {

{5.1, 3.6, 1.4, 0.2, 1}

}

Outer iteration 2 and 3

The next two granules contain only one object each, so they go directly into the reflection set.

reflection = {

{5.1, 3.6, 1.4, 0.2, 1},

{4.7, 3.2, 1.3, 0.2, 1},

{4.6, 3.1, 1.5, 0.2, 1}

}

This is, of course, only a slice of the reflection set for one decision concept. We can see that from the five objects of the original set in the process of granulation we obtained only three objects in the reflective set, which, however, according to the theory of rough sets, is a set representing internal knowledge to a degree similar to the knowledge of the original set. In the following chapters, this will be thoroughly presented and determined by comparing the classification measures on the original and granulated sets using selected classification algorithms.

By running the experiment to the end already without dissecting each step, we can obtain the following reflective sets for each degree of granulation and all decision classes.

Table 2.5: Reflection dataset for radius 0/4 (0.0), concept dependent granulation.

	sepal_length	sepal_width	petal_length	petal_width	iris class
1	5.0	3.0	1.4	0.2	1.0
2	5.5	3.2	4.9	1.5	2.0
3	6.3	3.0	5.8	2.1	3.0

Table 2.6: Reflection dataset for radius 1/4 (0.25), concept dependent granulation.

	sepal_length	sepal_width	petal_length	petal_width	iris class
1	5.1	3.1	1.4	0.2	1.0
2	7.0	3.2	4.5	1.5	2.0
3	6.4	3.2	4.7	1.5	2.0
4	5.5	2.3	4.0	1.3	2.0
5	6.3	3.3	6.0	1.8	3.0
6	5.8	2.7	5.1	1.9	3.0
7	7.1	3.0	5.9	2.1	3.0

Table 2.7: Reflection dataset for radius 2/4 (0.5), concept dependent granulation.

sepal_length	sepal_width	petal_length	petal_width	iris class
5.1	3.6	1.4	0.2	1.0
4.7	3.2	1.3	0.2	1.0
4.6	3.1	1.5	0.2	1.0
7.0	3.2	4.7	1.4	2.0
6.4	3.2	4.5	1.5	2.0
6.9	3.1	4.9	1.5	2.0
5.5	2.3	4.0	1.3	2.0
6.5	2.8	4.6	1.5	2.0
6.3	3.3	6.0	2.5	3.0
5.8	2.7	5.1	1.9	3.0
7.1	3.0	5.9	2.1	3.0
6.3	2.9	5.6	1.8	3.0
6.5	3.0	5.8	2.2	3.0

Table 2.8: Reflection dataset for radius 3/4 (0.75), concept dependent granulation.

sepal_length	sepal_width	petal_length	petal_width	iris class
5.1	3.5	1.4	0.2	1.0
4.9	3.0	1.4	0.2	1.0
4.7	3.2	1.3	0.2	1.0
4.6	3.1	1.5	0.2	1.0
5.0	3.6	1.4	0.2	1.0
7.0	3.2	4.7	1.4	2.0
6.4	3.2	4.5	1.5	2.0
6.9	3.1	4.9	1.5	2.0
5.5	2.3	4.0	1.3	2.0
6.5	2.8	4.6	1.5	2.0
6.3	3.3	6.0	2.5	3.0
5.8	2.7	5.1	1.9	3.0
7.1	3.0	5.9	2.1	3.0
6.3	2.9	5.6	1.8	3.0
6.5	3.0	5.8	2.2	3.0

Table 2.9: Reflection dataset for radius 4/4 (1.0), concept dependent granulation.

sepal_length	sepal_width	petal_length	petal_width	iris class
5.1	3.5	1.4	0.2	1.0
4.9	3.0	1.4	0.2	1.0
4.7	3.2	1.3	0.2	1.0
4.6	3.1	1.5	0.2	1.0
5.0	3.6	1.4	0.2	1.0
7.0	3.2	4.7	1.4	2.0
6.4	3.2	4.5	1.5	2.0
6.9	3.1	4.9	1.5	2.0
5.5	2.3	4.0	1.3	2.0
6.5	2.8	4.6	1.5	2.0
6.3	3.3	6.0	2.5	3.0
5.8	2.7	5.1	1.9	3.0
7.1	3.0	5.9	2.1	3.0
6.3	2.9	5.6	1.8	3.0
6.5	3.0	5.8	2.2	3.0

Summary for concept dependent granulation

This detailed example of how concept dependent granulation works was intended to convey as fully as possible the next steps of the entire algorithm. In the case of the standard granulation described in the next subsection, some of the details in the toy example section identical to those described here will be intentionally omitted.

The granulation itself aims to find objects in the data that, for the adopted degree of granulation, as closely as possible represent the internal knowledge while covering the entire dataset with knowledge granules. Concept dependent granulation here introduces the nuance of operating on subsets within the same decision class.

The number of degrees of granulation is determined by the number of features of the objects, and the choice of the optimal radius for solving a given problem can be driven by the desire to maximize the quality of classification, maximize the reduction in the number of objects, or find the optimal compromise between these measures in a given case.

2.2.2. Standard granulation

Standard granulation consists in creating groups of objects indiscernible to a fixed degree in terms of similarity relations and then covering the entire original universe of objects with these groups. In other words, coverage involves selecting granules with a specific selection strategy until the set of unique objects from the granules overlaps with the set of unique objects of the granular dataset. The component variables of granulation are how to determine the similarity of objects, the method of covering the universe with granules (groups), and the methods of creating granular representatives.
The following is the standard granulation procedure proposed by Polkowski in [24] and [25].

1. Loading the original decision system (U - universe of objects, A - non decision attributes, d - decision attribute).
2. Specifying the radius of granulation r_{gran}. Let the $u, v \epsilon U$.
3. For each object u we analyze all objects v, looking at the attributes from A, we create a set $IND(u, v) = \{a \epsilon A : a(u) = a(v)\}$ and we form a granule g with center with object u, assuming that

$$v \epsilon g_{r_{gran}}(u) \Leftrightarrow \mu(v, u, r_{gran}) \Leftrightarrow \frac{|IND(u, v)|}{|A|} \geq r_{gran} \tag{2.5}$$

then

$$g_{r_{gran}}(u) = \left\{ v : \frac{|IND(u,v)|}{|A|} \geq r_{gran} \right\}. \tag{2.6}$$

μ is a rough inclusion, formally derived from Lukasiewicz's t-norm.

4. We create granular coverage of the original decision system in one of the ways:
   - hierarchical coverage (granules are selected by sequence)
   - random selection of granules,
   - selecting granules with minimal, mean or maximal length,
   - selecting granules that convey the least, most or average number of new objects, respectively,
   - random selection of granules depending on concept size.

   Whether a granule is in the coverage set depends on whether it passes at least one new object.

   The original decision system is considered covered when the unique set of objects derived from the coverage granules overlaps with the entire original set of objects. It means that granules with center of u meet the condition:

$$\bigcup \{ g_{r_{gran}}(u) \ : \ g_{r_{gran}}(u) \in U_{cover} \} = U. \tag{2.7}$$

5. All objects in each granule are voting through a *majority voting* function which is used to select a representative new object. All ties are resolved by random choice. After all granules are processed, a new granular decision system is formed.

A toy example of the standard granulation.

As mentioned, this example will not be as detailed as for concept-dependent granulation, but the same data will be used to make it easier to compare results. The data is presented in the table 2.1.

Step 1: forming granules.

As in the case of concept-dependent granulation, we will start by generating indiscernibility matrices, which in the case of standard granulation will be of size $|U|x|U|$ i.e. here 15x15 since there is no division of the matrix into decision classes.

Table 2.10: Indiscernibility matrix for radius 0/4 (0.0)), standard granulation.

	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9	u_{10}	u_{11}	u_{12}	u_{13}	u_{14}	u_{15}
u_1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
u_2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
u_{15}	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 2.11: Indiscernibility matrix for radius 1/4 (0.25), standard granulation.

	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9	u_{10}	u_{11}	u_{12}	u_{13}	u_{14}	u_{15}
u_1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
u_2	1	1	1	1	1	0	0	0	0	0	0	0	1	0	1
u_3	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
u_4	1	1	1	1	1	0	0	1	0	0	0	0	0	0	0
u_5	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
u_6	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0
u_7	0	0	1	0	0	1	1	1	0	1	0	0	0	0	0
u_8	0	0	0	1	0	0	1	1	0	1	0	0	0	0	0
u_9	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
u_{10}	0	0	0	0	0	0	1	1	0	1	0	0	0	0	1
u_{11}	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
u_{12}	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
u_{13}	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1
u_{14}	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
u_{15}	0	1	0	0	0	0	0	0	0	1	0	0	1	0	1

Table 2.12: Indiscernibility matrix for radius 2/4 (0.5), standard granulation.

	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9	u_{10}	u_{11}	u_{12}	u_{13}	u_{14}	u_{15}
u_1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0
u_2	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0
u_3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
u_4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
u_5	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0
u_6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
u_7	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
u_8	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
u_9	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
u_{10}	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
u_{11}	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
u_{12}	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
u_{13}	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
u_{14}	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
u_{15}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 2.13: Indiscernibility matrix for radius 3/4 (0.75) and 4/4 (1.0), standard granulation.

	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9	u_{10}	u_{11}	u_{12}	u_{13}	u_{14}	u_{15}
u_1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
u_2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
u_3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
u_4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
u_5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
u_6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
u_7	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
u_8	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
u_9	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
u_{10}	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
u_{11}	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
u_{12}	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
u_{13}	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
u_{14}	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
u_{15}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Analyzing the above matrices, we can quickly deduce what the granules will look like, which we will formally write as follows.

Granules for radius 0/4

g(u_i):{$u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8, u_9, u_{10}, u_{11}, u_{12}, u_{13}, u_{14}, u_{15}$}, $for\ i = 1, ..., 15$

Granules for radius 1/4

g(u_1): {u_1, u_2, u_3, u_4, u_5}, g(u_2): {$u_1, u_2, u_3, u_4, u_5, u_{13}, u_{15}$}, g($u_3$): {$u_1, u_2, u_3, u_4, u_5, u_6, u_7$}, g($u_4$): {$u_1, u_2, u_3, u_4, u_5, u_8$}, g($u_5$): {$u_1, u_2, u_3, u_4, u_5$}, g($u_6$): {$u_3, u_6, u_7$}, g($u_7$): {$u_3, u_6, u_7, u_8, u_{10}$}, g($u_8$): {$u_4, u_7, u_8, u_{10}$}, g($u_9$): {$u_9$}, g($u_{10}$): {$u_7, u_8, u_{10}, u_{15}$}, g($u_{11}$): {$u_{11}, u_{14}$}, g($u_{12}$): {$u_{12}$}, g($u_{13}$): {$u_2, u_{13}, u_{15}$}, g($u_{14}$): {$u_{11}, u_{14}$}, g($u_{15}$): {$u_2, u_{10}, u_{13}, u_{15}$}

Granules for radius 2/4

g(u_1): {u_1, u_2, u_5}, g(u_2): {u_1, u_2, u_5}, g(u_3): {u_3}, g(u_4): {u_4}, g(u_5): {u_1, u_2, u_5},g(u_6): {u_6}, g(u_7): {u_7}, g(u_8): {u_8}, g(u_9): {u_9}, g(u_{10}): {u_{10}}, g(u_{11}): {u_{11}}, g(u_{12}): {u_{12}}, g(u_{13}): {u_{13}}, g(u_{14}): {u_{14}}, g(u_{15}): {u_{15}},

Granules for radius 3/4 and 4/410,1

g(u_i): {u_i}, $fori = 1, ...15$

Step 2: covering original dataset.

To find granular coverage, the same strategy will be used as in the example for concept-dependent granulation, i.e. hierarchical coverage. In the case of standard granulation, coverage is not done in individual decision concepts but for the entire data set.

In the case of the 0/4 radius, the situation is obvious and the very first granule contains all the objects and the selection process ends there.

So we have:

coverage for radii $\frac{0}{4}$ = {g(u_1)}

Other coverage collections look as follows.

coverage for radii $\frac{1}{4}$ = {g(u_1), g(u_2), g(u_3), g(u_4), g(u_7), g(u_9), g(u_{11}), g(u_{12})}

coverage for radii $\frac{2}{4}$ = {g(u_1), g(u_3), g(u_4), g(u_6), g(u_7), g(u_8), g(u_9), g(u_{10}), g(u_{11}), g(u_{12}), g(u_{13}), g(u_{14}), g(u_{15})}

coverage for radii $\frac{3}{4}$ = {g(u_1), g(u_2), g(u_3), g(u_4), g(u_5), g(u_6), g(u_7), g(u_8), g(u_9), g(u_{10}), g(u_{11}), g(u_{12}), g(u_{13}), g(u_{14}), g(u_{15})}

coverage for radii $\frac{4}{4}$ = {g(u_1), g(u_2), g(u_3), g(u_4), g(u_5), g(u_6), g(u_7), g(u_8), g(u_9), g(u_{10}), g(u_{11}), g(u_{12}), g(u_{13}), g(u_{14}), g(u_{15})}

Step 3: creating reflection dataset from coverage.

The main difference between standard granularity and concept-dependent granularity is that for the former, for a granularity radius of 0, only one object will be obtained, whose value for the decision class will take the value of the most numerous class or, in the case of a tie, a randomly selected class value. Therefore, we treat this case as an extreme case and in practice it is difficult to use its effect in a real scenario. Since in the case studied each of the three classes is equidistant, for each run of standard granulation we can get a similar value of attributes with one of the three values of the decision class.

Table 2.14: Reflection dataset for radius 0/4, standard granulation.

sepal_length	sepal_width	petal_length	petal_width	iris class
6.5	3.0	1.4	0.2	1.0

Table 2.15: Reflection dataset for radius 1/4, standard granulation.

sepal_length	sepal_width	petal_length	petal_width	iris class
4.6	3.5	1.4	0.2	1.0
7.1	3.0	1.4	0.2	1.0
7.0	3.2	1.4	0.2	1.0
5.0	3.1	1.4	0.2	1.0
7.0	3.2	1.3	1.5	2.0
5.5	2.3	4.0	1.3	2.0
6.3	3.3	5.6	2.5	3.0
5.8	2.7	5.1	1.9	3.0

Table 2.16: Reflection dataset for radius 2/4, standard granulation.

sepal_length	sepal_width	petal_length	petal_width	iris class
4.9	3.0	1.4	0.2	1.0
4.7	3.2	1.3	0.2	1.0
4.6	3.1	1.5	0.2	1.0
7.0	3.2	4.7	1.4	2.0
6.4	3.2	4.5	1.5	2.0
6.9	3.1	4.9	1.5	2.0
5.5	2.3	4.0	1.3	2.0
6.5	2.8	4.6	1.5	2.0
6.3	3.3	6.0	2.5	3.0
5.8	2.7	5.1	1.9	3.0
7.1	3.0	5.9	2.1	3.0
6.3	2.9	5.6	1.8	3.0
6.5	3.0	5.8	2.2	3.0

Table 2.17: Reflection dataset for radius 3/4, standard granulation.

sepal_length	sepal_width	petal_length	petal_width	iris class
5.1	3.5	1.4	0.2	1.0
4.9	3.0	1.4	0.2	1.0
4.7	3.2	1.3	0.2	1.0
4.6	3.1	1.5	0.2	1.0
5.0	3.6	1.4	0.2	1.0
7.0	3.2	4.7	1.4	2.0
6.4	3.2	4.5	1.5	2.0
6.9	3.1	4.9	1.5	2.0
5.5	2.3	4.0	1.3	2.0
6.5	2.8	4.6	1.5	2.0
6.3	3.3	6.0	2.5	3.0
5.8	2.7	5.1	1.9	3.0
7.1	3.0	5.9	2.1	3.0
6.3	2.9	5.6	1.8	3.0
6.5	3.0	5.8	2.2	3.0

Table 2.18: Reflection dataset for radius 4/4, standard granulation.

sepal_length	sepal_width	petal_length	petal_width	iris class
5.1	3.5	1.4	0.2	1.0
4.9	3.0	1.4	0.2	1.0
4.7	3.2	1.3	0.2	1.0
4.6	3.1	1.5	0.2	1.0
5.0	3.6	1.4	0.2	1.0
7.0	3.2	4.7	1.4	2.0
6.4	3.2	4.5	1.5	2.0
6.9	3.1	4.9	1.5	2.0
5.5	2.3	4.0	1.3	2.0
6.5	2.8	4.6	1.5	2.0
6.3	3.3	6.0	2.5	3.0
5.8	2.7	5.1	1.9	3.0
7.1	3.0	5.9	2.1	3.0
6.3	2.9	5.6	1.8	3.0
6.5	3.0	5.8	2.2	3.0

Summary for standard granulation

Comparing the reflective datasets of concept-dependent and standard granulation, several differences can be observed.

Although both techniques ultimately yielded a very similar number of objects in each reflection set (ignoring the aforementioned 0 radius), one can observe

significant differences in the number of objects in each decision class at a granulation radius of 1/4.
The largest of these will be in the pair with smaller granulation radii, where indiscernibility in the context of rough mereology will be greater. In addition, if this indiscernibility between decision classes is small then standard granulation will also yield a greater reduction in the number of objects relative to concept-dependent granulation, however, the consequence is the possibility of losing information about the actual decision class of a given object through majority voting taking place between decision concepts. This typically leads to lower classification accuracy than concept-dependent granulation.
Both granulation techniques, however, require the identification of multiple reflective systems depending on the number of attributes and, in the case of concept dependent granulation, also the cardinality of each decision class. Then, in the process of evaluating the classification on each reflection set, a trade-off can be determined between the reduction of its size and the accuracy of the classification, that is, the choice of the optimal granulation radius at a given time.
In the next chapter, a new granulation method named homogeneous granulation will be presented that does not require estimation of the optimal radius and the results of the experiments that have been carried out will make it possible to compare its main features and effectiveness compared to the methods presented in this chapter.

2.3. Layered granulation

Layered granulation [1] is an extension of previously presented approaches as granulation repeated recursively on previously generated reflection dataset from the original data. This idea was introduced in order to investigate the impact on the data by multiple granulation of it. Results of a different granulation techniques used in this thesis are presented in the experimental part.

2.4. Experimental part

2.4.1. Introduction

This section first describes all the datasets that were used in all the granulation and classification experiments described in this dissertation. Then there is the methodology of the research carried out in this chapter, as well as a collection and description of the results obtained in standard granulation, concept dependent granulation and layered granulation.

2.4.2. Datasets description

The following datasets were selected for experiments conducted with the methods presented in this dissertation the following datasets selected from the UCI repository [16] and from kaggle.com website.
Originally, some of the sets contained missing values, but due to the nature of the algorithms presented here, these rows were removed from the sets in the preprocessing phase. For this reason, the counts of individual observations for the *mushroom* and *adult* datasets differ from those presented on the official UCI repository website.
The preprocessing phase also included the removal of features that contained observation identifiers, such as in the breast cancer collection. Some of the data, although presented as integer values in the table below, was described as a categorical feature in the source, hence in the process of experimental research it was decided to also use those sets with features coded using the one-hot technique, which resulted in an increase in the number of features. This was done with the collection of australian credit (statlog) and that dataset was named *australian dummy*. Also for the *wine quality* dataset, a different approach was used, where experiments were conducted on three sets: only observations for red wine, observations for white wine and both sets combined into one.

Table 2.19: List of used datasets in the experimental sessions.

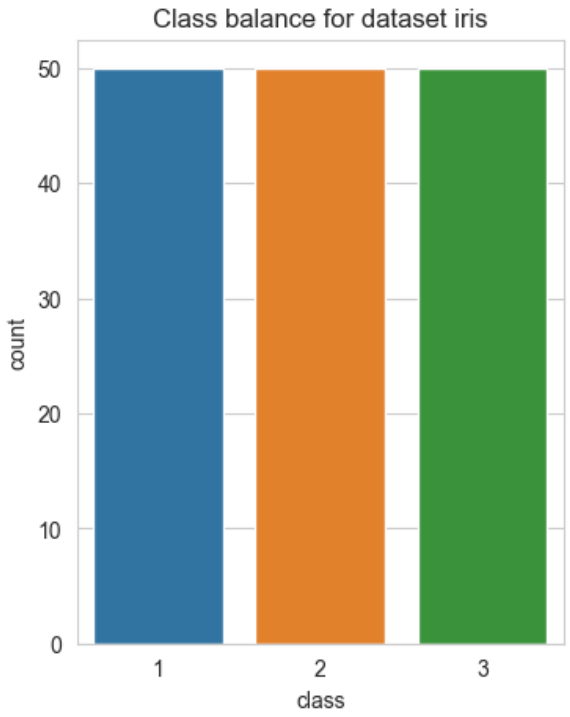
	name	# objects	# features	# classes	data types	missing values
1	iris	150	4	3	float	False
2	australian credit	690	14	2	int, float	False
3	australian dummy	689	38	2	int, float	False
4	heart	303	13	2	int, float	False
5	pima	768	8	2	int, float	False
6	breast	569	29	2	float	False
7	mushroom	5644	22	2	categorical	False
8	red wine	1599	11	6	float	False
9	white wine	4898	11	7	float	False
10	wine merged	6496	12	7	float, int	False
11	adult	45222	14	2	int, categorical	False

Some features of the datasets have a direct impact on the final classification results, aside from the variety of feature values, and one of them is undoubtedly decision class balance. The analysis of the distribution of decision classes will be repeated after granulation and compared with the baseline distribution, which will also provide a better understanding of the internal structure of each set's data and a better understanding of the possible difference in classification results before and after data granulation.

Below the distribution of these values for each dataset used is placed.

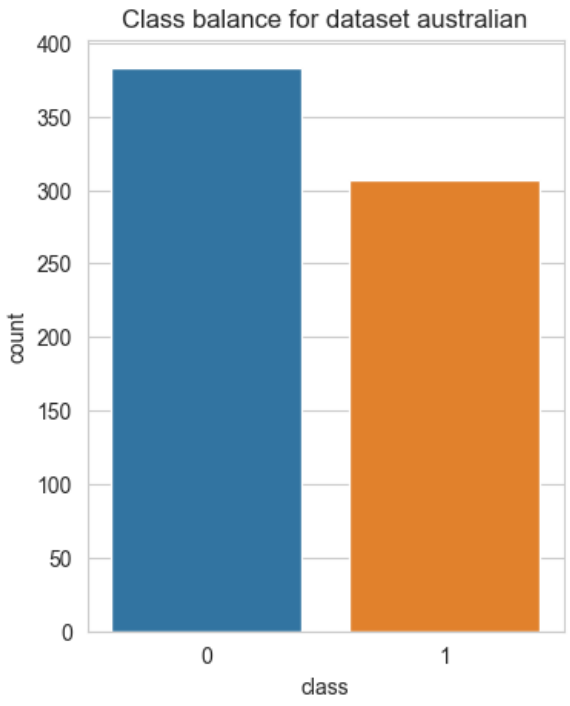
Dataset: iris

class	count
1	50
2	50
3	50



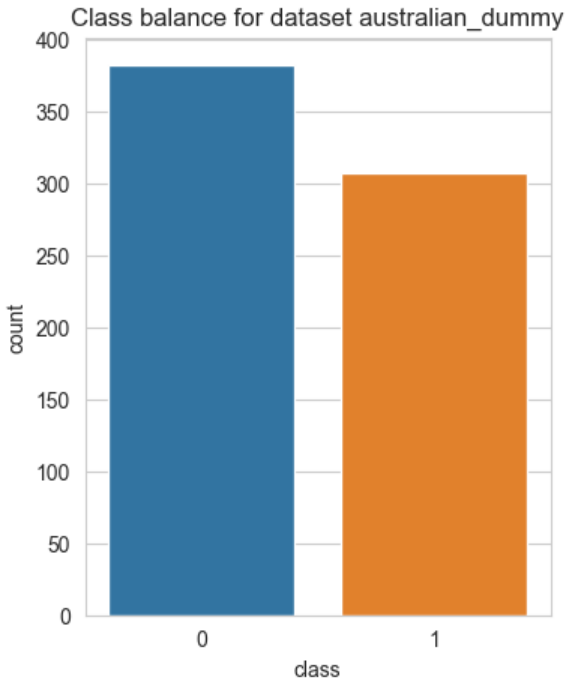
Dataset: australian credit

class	count
0	383
1	307



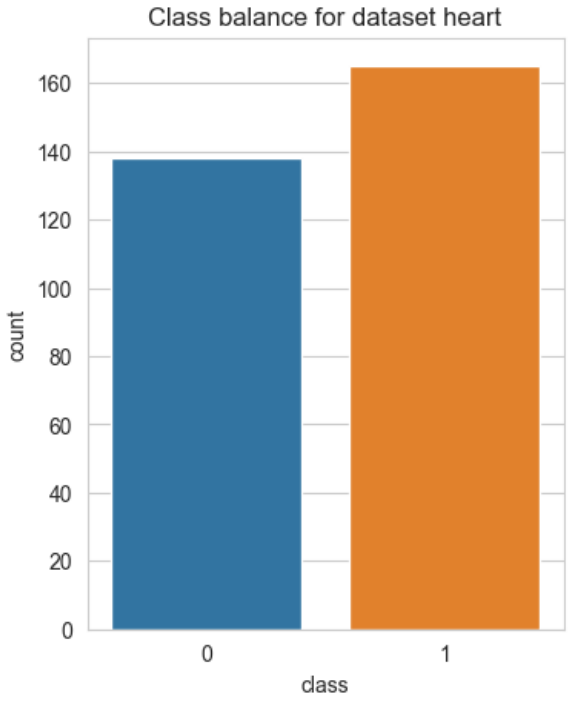
Dataset: australian credit dummy

class	count
0	382
1	307



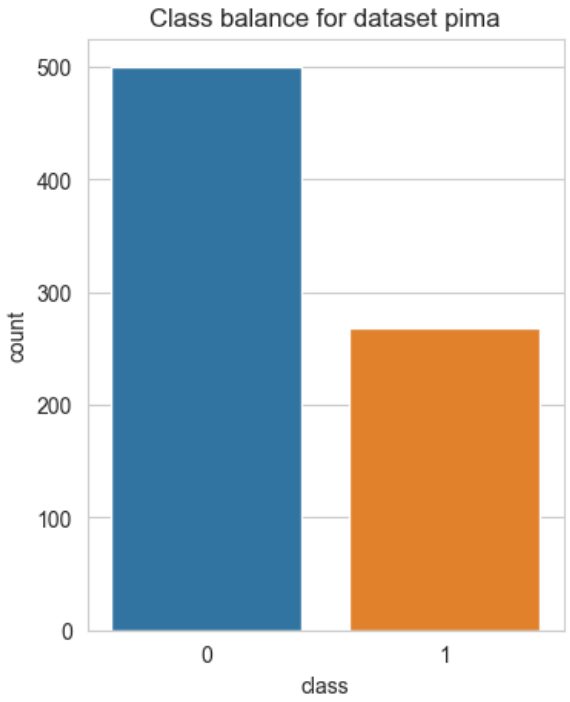
Dataset: heart disease

class	count
1	165
0	138



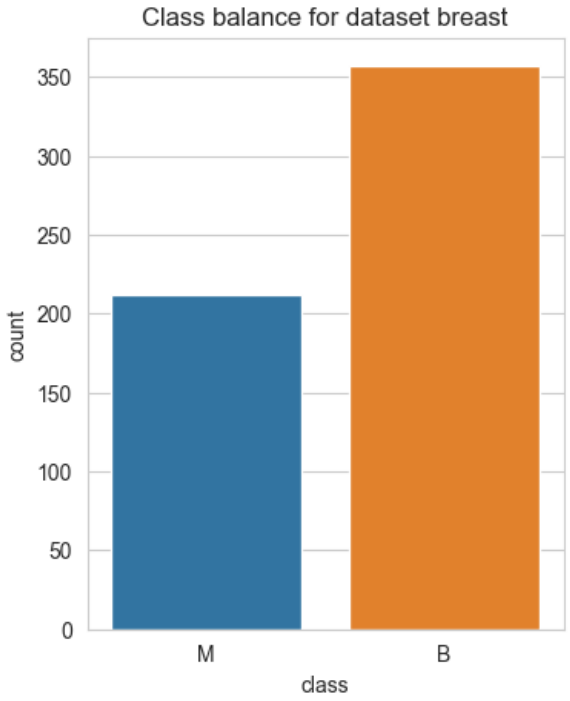
Dataset: pima

class	count
0	500
1	268



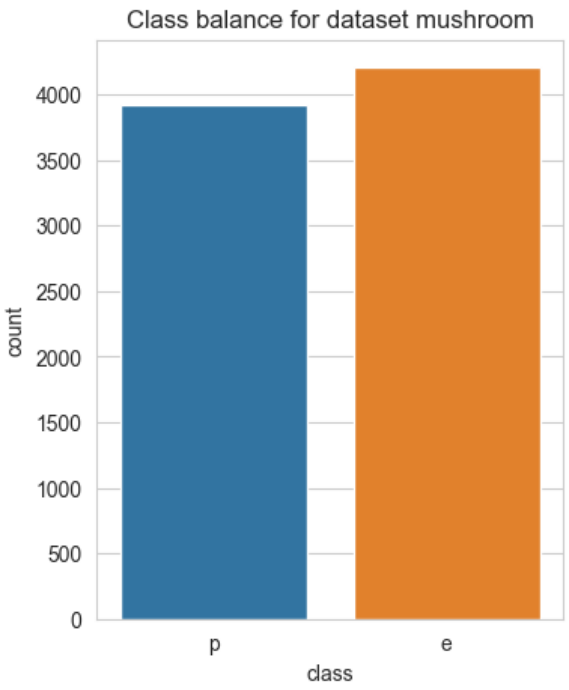
Dataset: breast

class	count
B (0)	357
M (1)	212



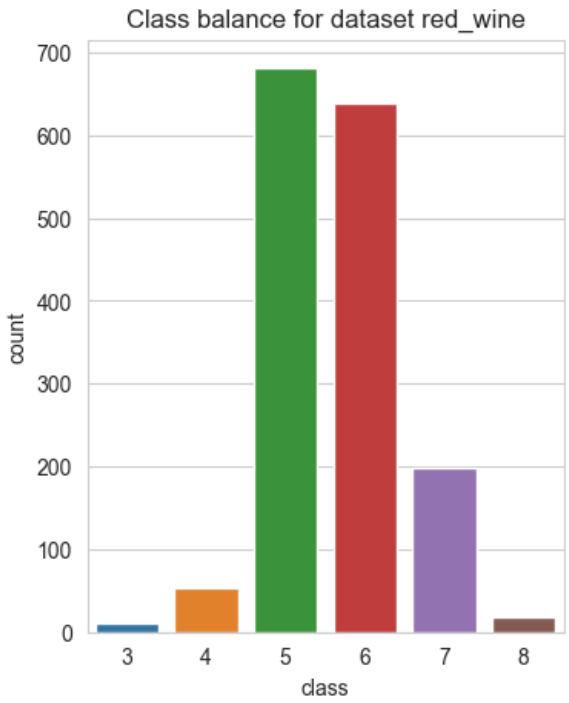
Dataset: mushroom

class	count
e (0)	4208
p (1)	3916



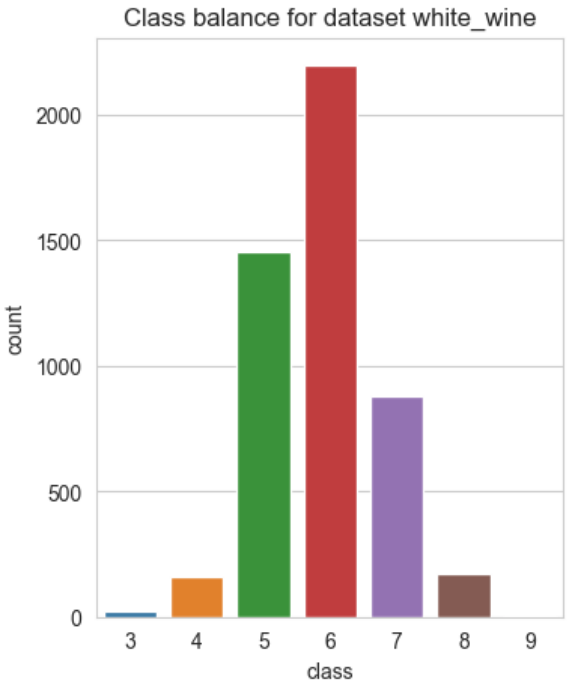
Dataset: red wine

class	count
5	681
6	638
7	199
4	53
8	18
3	10



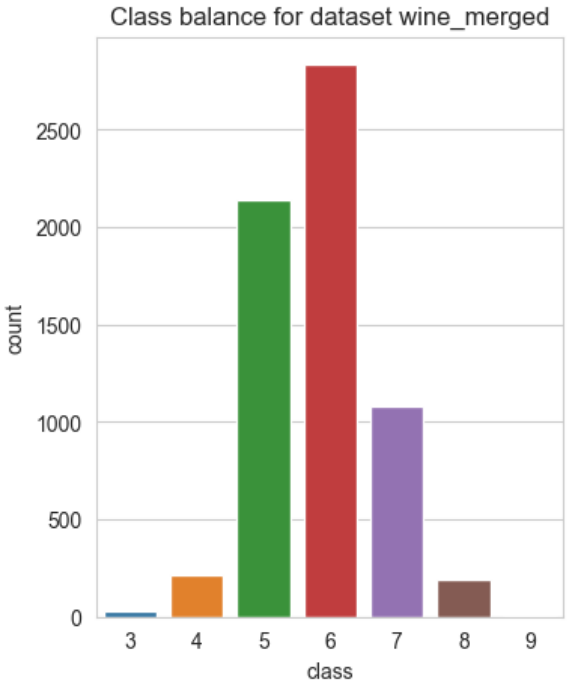
Dataset: white wine

class	count
6	2198
5	1457
7	880
8	175
4	163
3	20
9	5



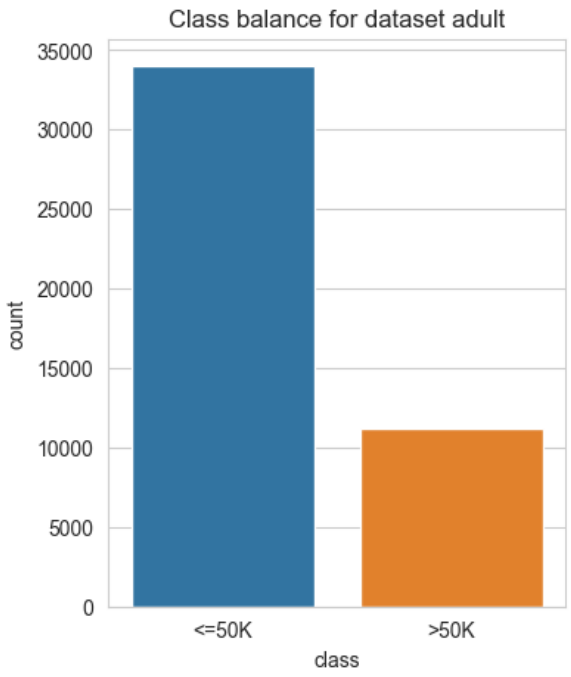
Dataset: wine merged

class	count
6	2836
5	2137
7	1079
4	216
8	193
3	30
9	5



Dataset: adult

class	count
$<= 50K$ (0)	34014
$> 50K$ (1)	11208



The datasets with categorical labels (breast, mushroom, adult) were number encoded to speed up computations and simplify code developed during experiments. Those values are placed in brackets near their categorical label values.

2.4.3. Methodology

The experimental part using different granulation techniques described in the following subsections was performed on the same datasets. The granulation process was performed 10 times for each training data set to reduce the impact of randomness on the final results and enable better comparison of the results. The sizes of granulated sets were compared and, in subsequent steps, the impact of the granularity of each set on the quality of classification measures compared to non-granulated sets.
The experimental part in this chapter will serve as reference results for the author's homogeneous granulation method described in section 3.1.

2.4.4. Concept dependent granulation experiments

The table below presents the averaged harvest size after 10 times concept dependent granulation, which will also serve to compare the differences in these sizes for other granulation techniques.

Table 2.20: Detailed information about datasets sizes after 10-times concept dependent granulation.

	dataset	radius	obj. # min	obj. # max	obj. # mean	% of all objects
0	adult	0.070	2	3	2.100	0.005
1	adult	0.140	3	6	4.400	0.010
2	adult	0.210	7	11	9.100	0.020
3	adult	0.290	18	29	21.700	0.048
4	adult	0.360	36	53	45.200	0.100
5	adult	0.430	104	127	117.600	0.260
6	adult	0.500	277	301	289.100	0.639
7	adult	0.570	689	728	707.200	1.564
8	adult	0.640	1707	1811	1752.300	3.875
9	adult	0.710	4258	4369	4333.800	9.583
10	adult	0.790	10572	10711	10624.200	23.493
11	adult	0.860	23353	23407	23381.500	51.704
12	adult	0.930	39163	39182	39175.500	86.629
13	adult	1.000	45175	45175	45175.000	99.896
14	australian	0.070	2	3	2.300	0.333
15	australian	0.140	2	4	3.500	0.507
16	australian	0.210	4	7	5.200	0.754
17	australian	0.290	8	13	9.900	1.435
18	australian	0.360	15	22	17.700	2.565
19	australian	0.430	32	39	35.400	5.130
20	australian	0.500	75	85	79.700	11.551
21	australian	0.570	169	185	178.300	25.841
22	australian	0.640	368	382	376.700	54.594
23	australian	0.710	561	577	570.000	82.609
24	australian	0.790	664	667	666.000	96.522
25	australian	0.860	683	683	683.000	98.986
26	australian	0.930	685	685	685.000	99.275
27	australian	1.000	690	690	690.000	100.000
28 - 47	australian_dummy	0.03 - 0.53	2	2	2.000	0.290
48	australian_dummy	0.550	2	3	2.300	0.334
49	australian_dummy	0.580	2	5	3.000	0.435
50	australian_dummy	0.610	3	6	4.200	0.610
51	australian_dummy	0.630	4	7	5.600	0.813

Continued on next page

	dataset	radius	obj. # min	obj. # max	obj. # mean	% of all objects
52	australian_dummy	0.660	6	11	8.900	1.292
53	australian_dummy	0.680	12	17	13.500	1.959
54	australian_dummy	0.710	18	22	20.800	3.019
55	australian_dummy	0.740	31	39	34.400	4.993
56	australian_dummy	0.760	53	61	58.900	8.549
57	australian_dummy	0.790	78	100	92.500	13.425
58	australian_dummy	0.820	150	178	164.200	23.832
59	australian_dummy	0.840	260	273	266.300	38.650
60	australian_dummy	0.870	430	440	434.800	63.106
61	australian_dummy	0.890	585	592	588.900	85.472
62	australian_dummy	0.920	666	667	666.800	96.778
63	australian_dummy	0.950	682	682	682.000	98.984
64	australian_dummy	0.970	684	684	684.000	99.274
65	australian_dummy	1.000	689	689	689.000	100.000
66	breast	0.030	237	251	242.300	42.583
67	breast	0.070	544	544	544.000	95.606
68	breast	0.100	557	557	557.000	97.891
69	breast	0.130	557	557	557.000	97.891
70	breast	0.170	557	557	557.000	97.891
71	breast	0.200	557	557	557.000	97.891
72	breast	0.230	568	568	568.000	99.824
73 - 95	breast	0.27 - 1.	569	569	569.000	100.000
96	heart	0.080	2	3	2.400	0.792
97	heart	0.150	2	4	2.900	0.957
98	heart	0.230	4	8	5.500	1.815
99	heart	0.310	8	13	10.300	3.399
100	heart	0.380	17	21	18.800	6.205
101	heart	0.460	38	46	41.400	13.663
102	heart	0.540	84	96	88.900	29.340
103	heart	0.620	155	168	161.200	53.201
104	heart	0.690	240	250	244.000	80.528
105	heart	0.770	287	290	288.600	95.248
106 - 108	heart	0.850 - 1.000	302	302	302.000	99.670
109	iris	0.250	22	27	24.600	16.400
110	iris	0.500	68	75	70.300	46.867
111	iris	0.750	131	133	131.900	87.933

Continued on next page

	dataset	radius	obj. # min	obj. # max	obj. # mean	% of all objects
112	iris	1.000	147	147	147.000	98.000
113 - 115	mushroom	0.050 - 0.140	2	2	2.000	0.035
116	mushroom	0.180	2	3	2.200	0.039
117	mushroom	0.230	2	3	2.100	0.037
118	mushroom	0.270	2	5	3.000	0.053
119	mushroom	0.320	3	7	4.900	0.087
120	mushroom	0.360	5	9	6.700	0.119
121	mushroom	0.410	6	11	8.300	0.147
122	mushroom	0.450	7	16	11.100	0.197
123	mushroom	0.500	12	24	16.500	0.292
124	mushroom	0.550	13	31	22.000	0.390
125	mushroom	0.590	20	33	23.600	0.418
126	mushroom	0.640	24	35	29.800	0.528
127	mushroom	0.680	33	45	38.900	0.689
128	mushroom	0.730	29	40	35.000	0.620
129	mushroom	0.770	34	51	42.200	0.748
130	mushroom	0.820	59	75	67.500	1.196
131	mushroom	0.860	130	148	140.900	2.496
132	mushroom	0.910	355	388	371.900	6.589
133	mushroom	0.950	1299	1337	1320.000	23.388
134	mushroom	1.000	5644	5644	5644.000	100.000
135	pima	0.120	30	41	35.700	4.648
136	pima	0.250	165	183	176.600	22.995
137	pima	0.380	430	452	441.800	57.526
138	pima	0.500	657	672	663.400	86.380
139	pima	0.620	761	762	761.400	99.141
140 - 142	pima	0.75 - 1.	768	768	768.000	100.000
143	red_wine	0.090	82	92	86.100	5.385
144	red_wine	0.180	355	385	367.700	22.996
145	red_wine	0.270	928	951	938.300	58.680
146	red_wine	0.360	1268	1272	1269.900	79.418
147	red_wine	0.450	1324	1324	1324.000	82.802
148	red_wine	0.550	1339	1339	1339.000	83.740
149	red_wine	0.640	1345	1345	1345.000	84.115
150	red_wine	0.730	1350	1350	1350.000	84.428
151	red_wine	0.820	1351	1351	1351.000	84.490

Continued on next page

	dataset	radius	obj. # min	obj. # max	obj. # mean	% of all objects
152	red_wine	0.910	1351	1351	1351.000	84.490
153	red_wine	1.000	1359	1359	1359.000	84.991
154	white_wine	0.090	118	130	123.700	2.526
155	white_wine	0.180	645	673	659.100	13.457
156	white_wine	0.270	2247	2300	2271.400	46.374
157	white_wine	0.360	3581	3596	3589.100	73.277
158	white_wine	0.450	3842	3843	3842.500	78.450
159	white_wine	0.550	3889	3889	3889.000	79.400
160	white_wine	0.640	3907	3908	3907.800	79.784
161	white_wine	0.730	3917	3917	3917.000	79.971
162	white_wine	0.820	3920	3920	3920.000	80.033
163	white_wine	0.910	3922	3922	3922.000	80.073
164	white_wine	1.000	3961	3961	3961.000	80.870
165	wine_merged	0.080	18	50	29.400	0.453
166	wine_merged	0.170	196	220	209.700	3.228
167	wine_merged	0.250	997	1044	1021.700	15.728
168	wine_merged	0.330	3182	3231	3206.300	49.358
169	wine_merged	0.420	4849	4863	4855.400	74.744
170	wine_merged	0.500	5164	5165	5164.600	79.504
171	wine_merged	0.580	5226	5226	5226.000	80.450
172	wine_merged	0.670	5250	5251	5250.500	80.827
173	wine_merged	0.750	5265	5265	5265.000	81.050
174	wine_merged	0.830	5269	5269	5269.000	81.111
175	wine_merged	0.920	5271	5271	5271.000	81.142
176	wine_merged	1.000	5320	5320	5320.000	81.897

In the table above columns have the following meaning:

- **dataset** - a dataset name
- **radius** - granulation radius in the range [0,1]
- **obj. # min** - lowest number of objects that were included in the final granular dataset
- **obj. # max** - highest number of objects that were included in the final granular dataset
- **obj. # mean** - mean number of objects that were included in the final granular dataset
- **% of all objects** - the mean percentage value of objects in the granuled dataset vs the original dataset size.

From observing this table, we can draw the first conclusions about the data sets themselves. The smaller the size of the granulated set for a given granulation radius, the

smaller the diversity of the set of objects within the decision classes and the better the effects of reducing the size of sets are achieved. We can also note that the non-deterministic nature of the granulation algorithm can have a significant impact on the size of the granulated set depending on the diversity of its objects. The minimum and maximum size of some sets with smaller granulation radii differs quite significantly. The impact of this granularity on the data can be better explored by using classification models using granular and original data.

We can also immediately notice sets with a high diversity of observations where, with low granulation radii, the number of objects in the coverage set is close to or equal 100% as we can observe in the *breast* dataset. In such a case, the effect of granulation is negligible or non-existent, but in such a case better granulation effects can be achieved by using epsilon granulation, in which we define a certain threshold of similarity (epsilon) between attribute values, thanks to which we can achieve a higher degree of indiscernibility of objects.

Another important factor which can be influenced by a data granulation is class balance in each granulation radii. Depending on data diversity in each decision class this initial balance can be disturbed or reversed. A good example is an australian dataset shown in the table 2.21 where the class balance for lower granulation radiuses changes so that the dominance between classes rotates with the next granulation radius.

Table 2.21: Detailed information about datasets decision class balance (average value) after 10-times concept dependent granulation.

	dataset	radius	class_balance
0	iris	0.250	1: 6.0, 2: 9.0, 3: 10.0
1	iris	0.500	1: 17.0, 2: 25.0, 3: 28.0
2	iris	0.750	1: 39.0, 2: 46.0, 3: 47.0
3	iris	1.000	1: 48.0, 2: 50.0, 3: 49.0
4	australian	0.071	0: 1.0, 1: 1.0
5	australian	0.143	0: 2.0, 1: 2.0
6	australian	0.214	0: 3.0, 1: 3.0
7	australian	0.286	0: 5.0, 1: 4.0
8	australian	0.357	0: 10.0, 1: 8.0
9	australian	0.429	0: 18.0, 1: 18.0
10	australian	0.500	0: 39.0, 1: 41.0
11	australian	0.571	0: 84.0, 1: 94.0
12	australian	0.643	0: 167.0, 1: 210.0

Continued on next page

	dataset	radius	class_balance
13	australian	0.714	0: 290.0, 1: 280.0
14	australian	0.786	0: 363.0, 1: 303.0
15	australian	0.857	0: 379.0, 1: 304.0
16	australian	0.929	0: 380.0, 1: 305.0
17	australian	1.000	0: 383.0, 1: 307.0
18 - 38	australian_dummy	0.026 - 0.553	0: 1.0, 1: 1.0
39	australian_dummy	0.579	0: 1.0, 1: 2.0
40	australian_dummy	0.605	0: 2.0, 1: 2.0
41	australian_dummy	0.632	0: 2.0, 1: 3.0
42	australian_dummy	0.658	0: 4.0, 1: 5.0
43	australian_dummy	0.684	0: 6.0, 1: 7.0
44	australian_dummy	0.711	0: 11.0, 1: 10.0
45	australian_dummy	0.737	0: 16.0, 1: 18.0
46	australian_dummy	0.763	0: 29.0, 1: 30.0
47	australian_dummy	0.789	0: 48.0, 1: 45.0
48	australian_dummy	0.816	0: 81.0, 1: 83.0
49	australian_dummy	0.842	0: 135.0, 1: 131.0
50	australian_dummy	0.868	0: 206.0, 1: 229.0
51	australian_dummy	0.895	0: 303.0, 1: 286.0
52	australian_dummy	0.921	0: 364.0, 1: 303.0
53	australian_dummy	0.947	0: 378.0, 1: 304.0
54	australian_dummy	0.974	0: 379.0, 1: 305.0
55	australian_dummy	1.000	0: 382.0, 1: 307.0
56	heart	0.077	0: 1.0, 1: 1.0
57	heart	0.154	0: 2.0, 1: 1.0
58	heart	0.231	0: 3.0, 1: 2.0
59	heart	0.308	0: 6.0, 1: 5.0
60	heart	0.385	0: 10.0, 1: 9.0
61	heart	0.462	0: 22.0, 1: 20.0
62	heart	0.538	0: 48.0, 1: 41.0
63	heart	0.615	0: 85.0, 1: 76.0
64	heart	0.692	0: 123.0, 1: 121.0
65	heart	0.769	0: 136.0, 1: 153.0
66	heart	0.846	0: 138.0, 1: 164.0
67	heart	0.923	0: 138.0, 1: 164.0
68	heart	1.000	0: 138.0, 1: 164.0

Continued on next page

	dataset	radius	class_balance
69	pima	0.125	0: 18.0, 1: 17.0
70	pima	0.250	0: 93.0, 1: 83.0
71	pima	0.375	0: 270.0, 1: 172.0
72	pima	0.500	0: 427.0, 1: 236.0
73	pima	0.625	0: 494.0, 1: 267.0
74	pima	0.750	0: 500.0, 1: 268.0
75	pima	0.875	0: 500.0, 1: 268.0
76	pima	1.000	0: 500.0, 1: 268.0
77	breast	0.033	'B': 139.0, 'M': 104.0
78	breast	0.067	'B': 334.0, 'M': 210.0
79	breast	0.100	'B': 345.0, 'M': 212.0
80	breast	0.133	'B': 345.0, 'M': 212.0
81	breast	0.167	'B': 345.0, 'M': 212.0
82	breast	0.200	'B': 345.0, 'M': 212.0
83	breast	0.233	'B': 356.0, 'M': 212.0
84 - 106	breast	0.267 - 1.	'B': 357.0, 'M': 212.0
107 - 111	mushroom	0.045 - 0.227	'e': 1.0, 'p': 1.0
112	mushroom	0.273	'e': 1.0, 'p': 2.0
113	mushroom	0.318	'e': 2.0, 'p': 3.0
114	mushroom	0.364	'e': 2.0, 'p': 4.0
115	mushroom	0.409	'e': 3.0, 'p': 5.0
116	mushroom	0.455	'e': 5.0, 'p': 6.0
117	mushroom	0.500	'e': 9.0, 'p': 8.0
118	mushroom	0.545	'e': 14.0, 'p': 8.0
119	mushroom	0.591	'e': 15.0, 'p': 9.0
120	mushroom	0.636	'e': 18.0, 'p': 11.0
121	mushroom	0.682	'e': 26.0, 'p': 13.0
122	mushroom	0.727	'e': 22.0, 'p': 13.0
123	mushroom	0.773	'e': 26.0, 'p': 16.0
124	mushroom	0.818	'e': 39.0, 'p': 29.0
125	mushroom	0.864	'e': 82.0, 'p': 59.0
126	mushroom	0.909	'e': 222.0, 'p': 150.0
127	mushroom	0.955	'e': 805.0, 'p': 515.0
128	mushroom	1.000	'e': 3488.0, 'p': 2156.0
129	red_wine	0.091	3: 5.0, 4: 11.0, 5: 21.0, 6: 24.0, 7: 17.0, 8: 7.0
130	red_wine	0.182	3: 8.0, 4: 34.0, 5: 116.0, 6: 125.0, 7: 71.0, 8: 14.0

Continued on next page

	dataset	radius	class_balance
131	red_wine	0.273	3: 10.0, 4: 49.0, 5: 362.0, 6: 363.0, 7: 137.0, 8: 17.0
132	red_wine	0.364	3: 10.0, 4: 53.0, 5: 526.0, 6: 504.0, 7: 160.0, 8: 17.0
133	red_wine	0.455	3: 10.0, 4: 53.0, 5: 561.0, 6: 520.0, 7: 163.0, 8: 17.0
134	red_wine	0.545	3: 10.0, 4: 53.0, 5: 567.0, 6: 527.0, 7: 165.0, 8: 17.0
135	red_wine	0.636	3: 10.0, 4: 53.0, 5: 571.0, 6: 529.0, 7: 165.0, 8: 17.0
136	red_wine	0.727	3: 10.0, 4: 53.0, 5: 574.0, 6: 531.0, 7: 165.0, 8: 17.0
137	red_wine	0.818	3: 10.0, 4: 53.0, 5: 575.0, 6: 531.0, 7: 165.0, 8: 17.0
138	red_wine	0.909	3: 10.0, 4: 53.0, 5: 575.0, 6: 531.0, 7: 165.0, 8: 17.0
139	red_wine	1.000	3: 10.0, 4: 53.0, 5: 577.0, 6: 535.0, 7: 167.0, 8: 17.0
140	white_wine	0.091	3: 8.0, 4: 18.0, 5: 27.0, 6: 28.0, 7: 23.0, 8: 16.0, 9: 4.0
141	white_wine	0.182	3: 16.0, 4: 74.0, 5: 176.0, 6: 200.0, 7: 130.0, 8: 58.0, 9: 5.0
142	white_wine	0.273	3: 20.0, 4: 139.0, 5: 672.0, 6: 897.0, 7: 426.0, 8: 114.0, 9: 5.0
143	white_wine	0.364	3: 20.0, 4: 150.0, 5: 1050.0, 6: 1601.0, 7: 634.0, 8: 129.0, 9: 5.0
144	white_wine	0.455	3: 20.0, 4: 150.0, 5: 1133.0, 6: 1736.0, 7: 668.0, 8: 130.0, 9: 5.0
145	white_wine	0.545	3: 20.0, 4: 150.0, 5: 1156.0, 6: 1756.0, 7: 672.0, 8: 130.0, 9: 5.0
146	white_wine	0.636	3: 20.0, 4: 151.0, 5: 1160.0, 6: 1766.0, 7: 675.0, 8: 131.0, 9: 5.0
147	white_wine	0.727	3: 20.0, 4: 152.0, 5: 1163.0, 6: 1767.0, 7: 679.0, 8: 131.0, 9: 5.0
148	white_wine	0.818	3: 20.0, 4: 153.0, 5: 1163.0, 6: 1768.0, 7: 680.0, 8: 131.0, 9: 5.0
149	white_wine	0.909	3: 20.0, 4: 153.0, 5: 1164.0, 6: 1769.0, 7: 680.0, 8: 131.0, 9: 5.0
150	white_wine	1.000	3: 20.0, 4: 153.0, 5: 1175.0, 6: 1788.0, 7: 689.0, 8: 131.0, 9: 5.0
151	wine_merged	0.083	3: 2.0, 4: 5.0, 5: 3.0, 6: 4.0, 7: 7.0, 8: 8.0, 9: 1.0
152	wine_merged	0.167	3: 13.0, 4: 30.0, 5: 49.0, 6: 58.0, 7: 42.0, 8: 24.0, 9: 4.0
153	wine_merged	0.250	3: 23.0, 4: 109.0, 5: 290.0, 6: 318.0, 7: 203.0, 8: 73.0, 9: 5.0
154	wine_merged	0.333	3: 30.0, 4: 188.0, 5: 1028.0, 6: 1256.0, 7: 560.0, 8: 131.0, 9: 5.0
155	wine_merged	0.417	3: 30.0, 4: 203.0, 5: 1575.0, 6: 2104.0, 7: 793.0, 8: 146.0, 9: 5.0
156	wine_merged	0.500	3: 30.0, 4: 203.0, 5: 1693.0, 6: 2257.0, 7: 830.0, 8: 147.0, 9: 5.0
157	wine_merged	0.583	3: 30.0, 4: 203.0, 5: 1722.0, 6: 2283.0, 7: 836.0, 8: 147.0, 9: 5.0
158	wine_merged	0.667	3: 30.0, 4: 204.0, 5: 1730.0, 6: 2295.0, 7: 838.0, 8: 148.0, 9: 5.0
159	wine_merged	0.750	3: 30.0, 4: 205.0, 5: 1736.0, 6: 2298.0, 7: 843.0, 8: 148.0, 9: 5.0
160	wine_merged	0.833	3: 30.0, 4: 206.0, 5: 1737.0, 6: 2299.0, 7: 844.0, 8: 148.0, 9: 5.0
161	wine_merged	0.917	3: 30.0, 4: 206.0, 5: 1738.0, 6: 2300.0, 7: 844.0, 8: 148.0, 9: 5.0
162	wine_merged	1.000	3: 30.0, 4: 206.0, 5: 1752.0, 6: 2323.0, 7: 856.0, 8: 148.0, 9: 5.0
163	adult	0.071	'$<= 50K$': 1.0, '$> 50K$': 1.0
164	adult	0.143	'$<= 50K$': 2.0, '$> 50K$': 2.0
165	adult	0.214	'$<= 50K$': 4.0, '$> 50K$': 5.0
166	adult	0.286	'$<= 50K$': 12.0, '$> 50K$': 10.0

Continued on next page

	dataset	radius	class_balance
167	adult	0.357	'$<= 50K$': 25.0, '$> 50K$': 20.0
168	adult	0.429	'$<= 50K$': 65.0, '$> 50K$': 52.0
169	adult	0.500	'$<= 50K$': 177.0, '$> 50K$': 112.0
170	adult	0.571	'$<= 50K$': 446.0, '$> 50K$': 261.0
171	adult	0.643	'$<= 50K$': 1159.0, '$> 50K$': 593.0
172	adult	0.714	'$<= 50K$': 3016.0, '$> 50K$': 1317.0
173	adult	0.786	'$<= 50K$': 7677.0, '$> 50K$': 2947.0
174	adult	0.857	'$<= 50K$': 17277.0, '$> 50K$': 6104.0
175	adult	0.929	'$<= 50K$': 29208.0, '$> 50K$': 9968.0
176	adult	1.000	'$<= 50K$': 33973.0, '$> 50K$': 11202.0

Class balance sumary and differences between achived results in this chapter can be found in section 2.4.6.

2.4.5. Standard granulation experiments

Below is a detailed table showing each crop and its averaged size after ten times the standard granulation.

Table 2.22: Detailed information about datasets sizes after 10-times standard granulation.

	dataset	radius	obj. # min	obj. # max	obj. # mean	% of all objects
0	australian	0.070	1	2	1.500	0.217
1	australian	0.140	1	3	2.100	0.304
2	australian	0.210	2	4	2.800	0.406
3	australian	0.290	3	8	5.300	0.768
4	australian	0.360	6	15	11.100	1.609
5	australian	0.430	18	27	22.500	3.261
6	australian	0.500	54	69	61.200	8.870
7	australian	0.570	143	156	149.800	21.710
8	australian	0.640	338	356	346.800	50.261
9	australian	0.710	547	562	554.000	80.290
10	australian	0.790	663	664	663.500	96.159

Continued on next page

	dataset	radius	obj. # min	obj. # max	obj. # mean	% of all objects
11	australian	0.860	682	682	682.000	98.841
12	australian	0.930	684	684	684.000	99.130
13	australian	1.000	690	690	690.000	100.000
14 - 33	australian_dummy	0.030 - 0.530	1	1	1.000	0.145
34	australian_dummy	0.550	1	2	1.200	0.174
35	australian_dummy	0.580	1	3	1.600	0.232
36	australian_dummy	0.610	2	4	2.500	0.363
37	australian_dummy	0.630	2	5	2.700	0.392
38	australian_dummy	0.660	4	8	5.400	0.784
39	australian_dummy	0.680	6	10	8.100	1.176
40	australian_dummy	0.710	8	18	13.100	1.901
41	australian_dummy	0.740	17	26	22.000	3.193
42	australian_dummy	0.760	38	43	41.100	5.965
43	australian_dummy	0.790	64	78	70.900	10.290
44	australian_dummy	0.820	126	141	132.900	19.289
45	australian_dummy	0.840	222	240	232.000	33.672
46	australian_dummy	0.870	399	420	406.600	59.013
47	australian_dummy	0.890	569	577	572.900	83.149
48	australian_dummy	0.920	663	664	663.600	96.313
49	australian_dummy	0.950	681	681	681.000	98.839
50	australian_dummy	0.970	683	683	683.000	99.129
51	australian_dummy	1.000	689	689	689.000	100.000
52	breast	0.030	181	202	191.900	33.726
53	breast	0.070	544	544	544.000	95.606
54	breast	0.100	557	557	557.000	97.891
55	breast	0.130	557	557	557.000	97.891
56	breast	0.170	557	557	557.000	97.891
57	breast	0.200	557	557	557.000	97.891
58	breast	0.230	568	568	568.000	99.824
59 - 81	breast	0.270 - 1.000	569	569	569.000	100.000
82	heart	0.080	1	2	1.500	0.495
83	heart	0.150	1	3	2.300	0.759

Continued on next page

	dataset	radius	obj. # min	obj. # max	obj. # mean	% of all objects
84	heart	0.230	2	6	3.500	1.155
85	heart	0.310	5	9	6.600	2.178
86	heart	0.380	10	17	13.600	4.488
87	heart	0.460	26	34	30.200	9.967
88	heart	0.540	67	78	71.800	23.696
89	heart	0.620	140	150	144.400	47.657
90	heart	0.690	235	242	238.300	78.647
91	heart	0.770	287	290	288.900	95.347
92 - 94	heart	0.850 - 1.000	302	302	302.000	99.670
95	iris	0.250	15	23	19.300	12.867
96	iris	0.500	63	72	66.800	44.533
97	iris	0.750	131	132	131.500	87.667
98	iris	1.000	147	147	147.000	98.000
99 - 101	mushroom	0.050 - 0.140	1	1	1.000	0.018
102	mushroom	0.180	1	2	1.200	0.021
103	mushroom	0.230	1	3	1.700	0.030
104	mushroom	0.270	1	4	1.900	0.034
105	mushroom	0.320	3	8	3.900	0.069
106	mushroom	0.360	5	8	6.300	0.112
107	mushroom	0.410	5	15	8.300	0.147
108	mushroom	0.450	8	14	10.300	0.182
109	mushroom	0.500	13	24	15.900	0.282
110	mushroom	0.550	12	26	20.400	0.361
111	mushroom	0.590	15	38	25.700	0.455
112	mushroom	0.640	22	39	29.100	0.516
113	mushroom	0.680	24	44	35.800	0.634
114	mushroom	0.730	26	48	37.800	0.670
115	mushroom	0.770	36	47	42.400	0.751
116	mushroom	0.820	60	76	67.700	1.200
117	mushroom	0.860	135	156	144.600	2.562
118	mushroom	0.910	367	406	381.700	6.763
119	mushroom	0.950	1313	1350	1328.200	23.533

Continued on next page

	dataset	radius	obj. # min	obj. # max	obj. # mean	% of all objects
120	mushroom	1.000	5644	5644	5644.000	100.000
121	pima	0.120	18	25	21.300	2.773
122	pima	0.250	117	135	126.800	16.510
123	pima	0.380	375	398	389.700	50.742
124	pima	0.500	629	641	635.800	82.786
125	pima	0.620	757	758	757.300	98.607
126	pima	0.750	768	768	768.000	100.000
127	pima	0.880	768	768	768.000	100.000
128	pima	1.000	768	768	768.000	100.000
129	red_wine	0.090	25	32	28.000	1.751
130	red_wine	0.180	178	193	187.500	11.726
131	red_wine	0.270	716	732	722.400	45.178
132	red_wine	0.360	1218	1229	1223.300	76.504
133	red_wine	0.450	1316	1317	1316.800	82.351
134	red_wine	0.550	1337	1337	1337.000	83.615
135	red_wine	0.640	1345	1345	1345.000	84.115
136	red_wine	0.730	1350	1350	1350.000	84.428
137	red_wine	0.820	1351	1351	1351.000	84.490
138	red_wine	0.910	1351	1351	1351.000	84.490
139	red_wine	1.000	1359	1359	1359.000	84.991
140	white_wine	0.090	33	37	34.900	0.713
141	white_wine	0.180	260	284	269.000	5.492
142	white_wine	0.270	1464	1517	1496.000	30.543
143	white_wine	0.360	3303	3323	3311.000	67.599
144	white_wine	0.450	3811	3814	3812.000	77.828
145	white_wine	0.550	3874	3876	3875.100	79.116
146	white_wine	0.640	3900	3901	3900.800	79.641
147	white_wine	0.730	3916	3916	3916.000	79.951
148	white_wine	0.820	3920	3920	3920.000	80.033
149	white_wine	0.910	3922	3922	3922.000	80.073
150	white_wine	1.000	3961	3961	3961.000	80.870
151	wine_merged	0.080	2	8	4.000	0.062

Continued on next page

	dataset	radius	obj. # min	obj. # max	obj. # mean	% of all objects
152	wine_merged	0.170	59	75	67.300	1.036
153	wine_merged	0.250	428	462	445.900	6.863
154	wine_merged	0.330	2186	2247	2214.400	34.083
155	wine_merged	0.420	4517	4542	4530.800	69.737
156	wine_merged	0.500	5123	5129	5126.900	78.912
157	wine_merged	0.580	5209	5211	5210.500	80.199
158	wine_merged	0.670	5243	5244	5243.700	80.710
159	wine_merged	0.750	5264	5264	5264.000	81.022
160	wine_merged	0.830	5269	5269	5269.000	81.099
161	wine_merged	0.920	5271	5271	5271.000	81.130
162	wine_merged	1.000	5320	5320	5320.000	81.884

During granulation, the original balance of classes in the set may change, which is also important information in the context of the classification results presented later in this chapter. Detailed information about the balance of classes after granulation using the standard method can be found in the table 2.23.

Table 2.23: Detailed information about datasets decision class balance (average value) after 10-times standard granulation.

	dataset	radius	class_balance
0	iris	0.250	3: 8.0, 2: 6.0, 1: 5.0
1	iris	0.500	3: 27.0, 1: 18.0, 2: 21.0
2	iris	0.750	3: 47.0, 1: 38.0, 2: 46.0
3	iris	1.000	1: 48.0, 3: 49.0, 2: 50.0
4	australian	0.071	0: 2.0
5	australian	0.143	0: 2.0
6	australian	0.214	0: 3.0, 1: 0.0
7	australian	0.286	0: 3.0, 1: 2.0
8	australian	0.357	0: 8.0, 1: 4.0
9	australian	0.429	0: 13.0, 1: 10.0
10	australian	0.500	0: 33.0, 1: 28.0
11	australian	0.571	0: 75.0, 1: 75.0
12	australian	0.643	1: 186.0, 0: 161.0

Continued on next page

	dataset	radius	class_balance
13	australian	0.714	0: 285.0, 1: 269.0
14	australian	0.786	0: 363.0, 1: 301.0
15	australian	0.857	1: 304.0, 0: 378.0
16	australian	0.929	1: 305.0, 0: 379.0
17	australian	1.000	0: 383.0, 1: 307.0
18 - 38	australian_dummy	0.026 - 0.553	0: 1.0
39	australian_dummy	0.579	0: 2.0
40	australian_dummy	0.605	0: 2.0
41	australian_dummy	0.632	0: 3.0
42	australian_dummy	0.658	0: 5.0, 1: 1.0
43	australian_dummy	0.684	1: 3.0, 0: 5.0
44	australian_dummy	0.711	1: 4.0, 0: 9.0
45	australian_dummy	0.737	1: 8.0, 0: 14.0
46	australian_dummy	0.763	0: 25.0, 1: 16.0
47	australian_dummy	0.789	0: 42.0, 1: 29.0
48	australian_dummy	0.816	1: 60.0, 0: 73.0
49	australian_dummy	0.842	0: 120.0, 1: 112.0
50	australian_dummy	0.868	1: 212.0, 0: 195.0
51	australian_dummy	0.895	0: 299.0, 1: 274.0
52	australian_dummy	0.921	0: 362.0, 1: 302.0
53	australian_dummy	0.947	0: 377.0, 1: 304.0
54	australian_dummy	0.974	1: 305.0, 0: 378.0
55	australian_dummy	1.000	1: 307.0, 0: 382.0
56	heart	0.077	1: 2.0
57	heart	0.154	1: 2.0, 0: 0.0
58	heart	0.231	1: 3.0, 0: 1.0
59	heart	0.308	0: 3.0, 1: 4.0
60	heart	0.385	1: 7.0, 0: 6.0
61	heart	0.462	1: 15.0, 0: 15.0
62	heart	0.538	0: 35.0, 1: 37.0
63	heart	0.615	0: 72.0, 1: 72.0
64	heart	0.692	1: 120.0, 0: 119.0
65	heart	0.769	0: 135.0, 1: 154.0
66	heart	0.846	1: 164.0, 0: 138.0
67	heart	0.923	0: 138.0, 1: 164.0
68	heart	1.000	1: 164.0, 0: 138.0

Continued on next page

	dataset	radius	class_balance
69	pima	0.125	0: 21.0, 1: 0.0
70	pima	0.250	0: 108.0, 1: 18.0
71	pima	0.375	0: 268.0, 1: 122.0
72	pima	0.500	0: 417.0, 1: 219.0
73	pima	0.625	0: 493.0, 1: 265.0
74	pima	0.750	1: 268.0, 0: 500.0
75	pima	0.875	0: 500.0, 1: 268.0
77	breast	0.033	'B': 139.0, 'M': 53.0
78	breast	0.067	'B': 334.0, 'M': 210.0
79	breast	0.100	'B': 345.0, 'M': 212.0
80	breast	0.133	'M': 212.0, 'B': 345.0
81	breast	0.167	'B': 345.0, 'M': 212.0
82	breast	0.200	'M': 212.0, 'B': 345.0
83	breast	0.233	'M': 212.0, 'B': 356.0
84 - 88	breast	0.267 - 0.400	'B': 357.0, 'M': 212.0
89	breast	0.433	'M': 212.0, 'B': 357.0
90	breast	0.467	'B': 357.0, 'M': 212.0
91	breast	0.500	'M': 212.0, 'B': 357.0
92 - 95	breast	0.533 - 0.633	'B': 357.0, 'M': 212.0
96 - 98	breast	0.667 - 0.700	'M': 212.0, 'B': 357.0
99 - 101	breast	0.767 - 0.833	'B': 357.0, 'M': 212.0
102	breast	0.867	'M': 212.0, 'B': 357.0
103	breast	0.900	'B': 357.0, 'M': 212.0
104	breast	0.933	'B': 357.0, 'M': 212.0
105	breast	0.967	'M': 212.0, 'B': 357.0
106	breast	1.000	'M': 212.0, 'B': 357.0
107 - 110	mushroom	0.045 - 0.182	'e': 1.0
111	mushroom	0.227	'e': 2.0
112	mushroom	0.273	'e': 2.0
113	mushroom	0.318	'e': 4.0
114	mushroom	0.364	'e': 6.0, 'p': 0.0
115	mushroom	0.409	'e': 7.0, 'p': 1.0
116	mushroom	0.455	'e': 6.0, 'p': 4.0
117	mushroom	0.500	'p': 4.0, 'e': 11.0
118	mushroom	0.545	'e': 16.0, 'p': 4.0
119	mushroom	0.591	'p': 4.0, 'e': 22.0

Continued on next page

	dataset	radius	class_balance
120	mushroom	0.636	’e’: 25.0, ’p’: 4.0
121	mushroom	0.682	’p’: 10.0, ’e’: 26.0
122	mushroom	0.727	’e’: 23.0, ’p’: 15.0
123	mushroom	0.773	’p’: 17.0, ’e’: 25.0
124	mushroom	0.818	’e’: 40.0, ’p’: 28.0
125	mushroom	0.864	’e’: 84.0, ’p’: 60.0
126	mushroom	0.909	’p’: 151.0, ’e’: 231.0
127	mushroom	0.955	’p’: 515.0, ’e’: 814.0
128	mushroom	1.000	’e’: 3488.0, ’p’: 2156.0
129	red_wine	0.091	6: 9.0, 5: 19.0
130	red_wine	0.182	6: 79.0, 5: 106.0, 7: 3.0
131	red_wine	0.273	6: 286.0, 5: 325.0, 7: 86.0, 4: 16.0, 8: 7.0, 3: 2.0
132	red_wine	0.364	8: 15.0, 5: 511.0, 6: 484.0, 7: 156.0, 4: 48.0, 3: 10.0
133	red_wine	0.455	6: 516.0, 7: 164.0, 5: 558.0, 4: 53.0, 3: 10.0, 8: 16.0
134	red_wine	0.545	6: 526.0, 5: 566.0, 4: 53.0, 7: 165.0, 8: 17.0, 3: 10.0
135	red_wine	0.636	6: 529.0, 5: 571.0, 7: 165.0, 4: 53.0, 8: 17.0, 3: 10.0
136	red_wine	0.727	5: 574.0, 7: 165.0, 6: 531.0, 4: 53.0, 8: 17.0, 3: 10.0
137	red_wine	0.818	5: 575.0, 6: 531.0, 7: 165.0, 3: 10.0, 8: 17.0, 4: 53.0
138	red_wine	0.909	6: 531.0, 7: 165.0, 5: 575.0, 4: 53.0, 8: 17.0, 3: 10.0
139	red_wine	1.000	5: 577.0, 7: 167.0, 6: 535.0, 3: 10.0, 4: 53.0, 8: 17.0
140	white_wine	0.091	6: 35.0
141	white_wine	0.182	6: 233.0, 5: 33.0, 7: 3.0
142	white_wine	0.273	6: 830.0, 5: 411.0, 7: 200.0, 4: 24.0, 8: 26.0, 3: 4.0, 9: 0.0
143	white_wine	0.364	8: 112.0, 7: 570.0, 6: 1496.0, 5: 973.0, 4: 138.0, 3: 18.0, 9: 4.0
144	white_wine	0.455	6: 1722.0, 8: 130.0, 5: 1123.0, 7: 663.0, 4: 149.0, 3: 20.0, 9: 5.0
145	white_wine	0.545	8: 130.0, 5: 1152.0, 7: 670.0, 6: 1748.0, 4: 150.0, 3: 20.0, 9: 5.0
146	white_wine	0.636	5: 1158.0, 6: 1762.0, 8: 131.0, 7: 674.0, 4: 151.0, 9: 5.0, 3: 20.0
147	white_wine	0.727	6: 1766.0, 5: 1163.0, 7: 678.0, 8: 131.0, 9: 5.0, 4: 152.0, 3: 20.0
148	white_wine	0.818	7: 680.0, 5: 1163.0, 6: 1768.0, 4: 153.0, 8: 131.0, 3: 20.0, 9: 5.0
149	white_wine	0.909	6: 1769.0, 7: 680.0, 5: 1164.0, 8: 131.0, 4: 153.0, 3: 20.0, 9: 5.0
150	white_wine	1.000	7: 689.0, 6: 1788.0, 5: 1175.0, 8: 131.0, 4: 153.0, 3: 20.0, 9: 5.0
151	wine_merged	0.083	6: 4.0, 5: 0.0
152	wine_merged	0.167	6: 51.0, 5: 16.0
153	wine_merged	0.250	6: 310.0, 5: 130.0, 7: 6.0
154	wine_merged	0.333	6: 1122.0, 5: 734.0, 7: 281.0, 8: 32.0, 4: 38.0, 3: 7.0, 9: 0.0
155	wine_merged	0.417	6: 1979.0, 5: 1477.0, 3: 28.0, 7: 730.0, 4: 185.0, 8: 127.0, 9: 4.0

Continued on next page

	dataset	radius	class_balance
156	wine_merged	0.500	5: 1680.0, 6: 2239.0, 4: 202.0, 7: 825.0, 8: 145.0, 3: 30.0, 9: 5.0
157	wine_merged	0.583	6: 2276.0, 5: 1717.0, 7: 834.0, 8: 147.0, 4: 202.0, 9: 5.0, 3: 30.0
158	wine_merged	0.667	7: 838.0, 6: 2291.0, 5: 1728.0, 8: 148.0, 3: 30.0, 4: 204.0, 9: 5.0
159	wine_merged	0.750	5: 1736.0, 8: 148.0, 7: 843.0, 6: 2297.0, 4: 205.0, 9: 5.0, 3: 30.0
160	wine_merged	0.833	7: 844.0, 5: 1737.0, 6: 2299.0, 8: 148.0, 4: 206.0, 3: 30.0, 9: 5.0
161	wine_merged	0.917	5: 1738.0, 6: 2300.0, 7: 844.0, 8: 148.0, 4: 206.0, 3: 30.0, 9: 5.0
162	wine_merged	1.000	6: 2323.0, 5: 1752.0, 7: 856.0, 4: 206.0, 8: 148.0, 3: 30.0, 9: 5.0

2.4.6. Results comparison for standard and concept dependent granulation

Graph showing the difference in the size of granular reflections of training systems calculated by concept dependent and standard techniques is shown in figure 2.1. The y-axis labeled $percent\_diff$ denotes the size of the standard granular system relative to the concept dependent granular system. This can be written as an expression:

$$percent\_diff = \frac{cd\_size\_for\_radius - standard\_size\_for\_radius}{cd\_size\_for\_radius} \tag{2.8}$$

That is, a value of 0.5 on the y-axis means that the set after standard granulation for a given radius is 50% of the size of the set after concept dependent granulation for the same radius. A value of 0 means there is no difference in size, and value < 0 means that the set after standard granulation contains more objects than that after concept dependent granulation. Standard granulation, as proven by the experiments, generally results in more approximated data sets.

From observing the above graph, we can draw several conclusions. A high value on the y-axis means that the dataset is differentiated between decision classes, which does not allow standard granulation to "merge" similar objects between classes. Such a situation is particularly evident for the sets $red\ wine$, $white\ wine$ and $wine\ merged$ and, to a lesser extent, for the sets $australian\ dummy$, $adult$ and $australian$.

Comparison of the class balance of the two presented granulation methods for radius 1 and the original datasets can be found in the table 2.24.

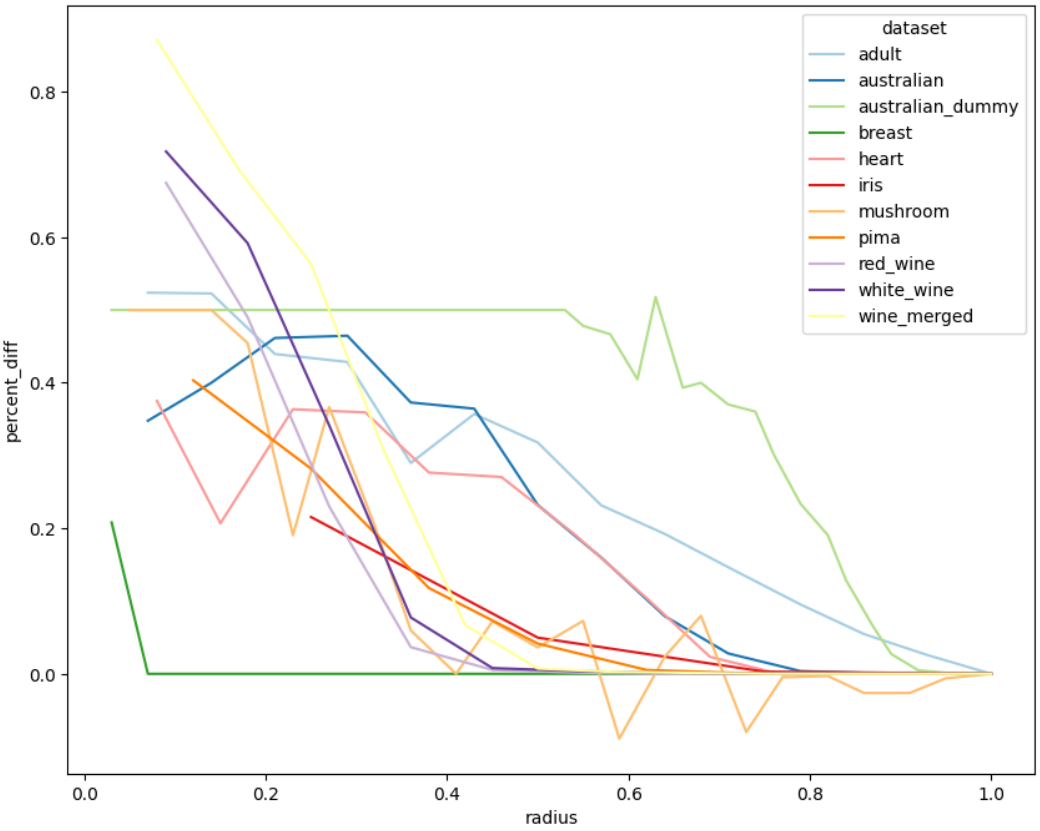


Figure 2.1: Reflection dataset sizes comparison for each granulation radius.

Table 2.24: Comparison of class balance between concept dependent granulation radius of 1, standard granulation of radius 1 and original datasets.

dataset	class balance	cdgran class balance	standard class balance
iris	1: 50, 2: 50, 3: 50	1: 48, 2: 50, 3: 49	1: 48, 3: 49, 2: 50
australian	0: 383, 1: 307	0: 383, 1: 307	0: 383, 1: 307
australian dummy	0: 382, 1: 307	0: 382, 1: 307	1: 307, 0: 382
heart	1: 165, 0: 138	0: 138, 1: 164	1: 164, 0: 138
pima	1: 268, 0: 500	0: 500, 1: 268	0: 500, 1: 268
breast	'M': 212, 'B': 357	'B': 357, 'M': 212	'M': 212, 'B': 357
mushroom	'p': 2156, 'e': 3488	'e': 3488, 'p': 2156	'e': 3488, 'p': 2156
red wine	5: 681, 6: 638, 7: 199, 4: 53, 8: 18, 3: 10	3: 10, 4: 53, 5: 577, 6: 535, 7: 167, 8: 17	5: 577, 7: 167, 6: 535, 3: 10, 4: 53, 8: 17
white wine	6: 2198, 5: 1457, 7: 880, 8: 175, 4: 163, 3: 20, 9: 5	3: 20, 4: 153, 5: 1175, 6: 1788, 7: 689, 8: 131, 9: 5	7: 689, 6: 1788, 5: 1175, 8: 131, 4: 153, 3: 20, 9: 5
wine merged	5: 2138, 6: 2836, 7: 1079, 4: 216, 8: 193, 3: 30, 9: 5	3: 30, 4: 206, 5: 1752, 6: 2323, 7: 856, 8: 148, 9: 5	6: 2323, 5: 1752, 7: 856, 4: 206, 8: 148, 3: 30, 9: 5
adult	'<=50K': 34014, '>50K': 11208	'<=50K': 33973, '>50K': 11202	0: 33970, 1: 11200

Comparison of granulation for radii close to or equal to 0.25, 0.5, 0.75, 1.0 for concept dependent and standard granulation is shown in the table 2.25.

Table 2.25: Comparison of class balance between concept dependent granulation radius and standard granulation for chosen radiuses.

dataset	radius	cdgran_class_balance	standard_class_balance
iris	0.25	{1: 6, 2: 9, 3: 10}	{3: 8, 2: 6, 1: 5}
iris	0.50	{1: 17, 2: 25, 3: 28}	{3: 27, 1: 18, 2: 21}
iris	0.75	{1: 39, 2: 46, 3: 47}	{3: 47, 1: 38, 2: 46}
iris	1.00	{1: 48, 2: 50, 3: 49}	{1: 48, 3: 49, 2: 50}
australian	0.29	{0: 5, 1: 4}	{0: 3, 1: 2}
australian	0.50	{0: 39, 1: 41}	{0: 33, 1: 28}
australian	0.79	{0: 363, 1: 303}	{0: 363, 1: 301}
australian	1.00	{0: 383, 1: 307}	{0: 383, 1: 307}
australian_dummy	0.26	{0: 1, 1: 1}	{0: 1}
australian_dummy	0.50	{0: 1, 1: 1}	{0: 1}
australian_dummy	0.76	{0: 29, 1: 30}	{0: 25, 1: 16}
australian_dummy	1.00	{0: 382, 1: 307}	{1: 307, 0: 382}
heart	0.23	{0: 3, 1: 2}	{1: 3, 0: 1}
heart	0.54	{0: 48, 1: 41}	{0: 35, 1: 37}
heart	0.77	{0: 136, 1: 153}	{0: 135, 1: 154}
heart	1.00	{0: 138, 1: 164}	{1: 164, 0: 138}
pima	0.25	{0: 93, 1: 83}	{0: 108, 1: 18}
pima	0.50	{0: 427, 1: 236}	{0: 417, 1: 219}
pima	0.75	{0: 500, 1: 268}	{1: 268, 0: 500}
pima	1.00	{0: 500, 1: 268}	{0: 500, 1: 268}
breast	0.27	{'B': 357, 'M': 212}	{'B': 357, 'M': 212}
breast	0.50	{'B': 357, 'M': 212}	{'M': 212, 'B': 357}
breast	0.73	{'B': 357, 'M': 212}	{'M': 212, 'B': 357}
breast	1.00	{'B': 357, 'M': 212}	{'M': 212, 'B': 357}
mushroom	0.27	{'e': 1, 'p': 2}	{'e': 2}
mushroom	0.50	{'e': 9, 'p': 8}	{'p': 4, 'e': 11}
mushroom	0.77	{'e': 26, 'p': 16}	{'p': 17, 'e': 25}
mushroom	1.00	{'e': 3488, 'p': 2156}	{'e': 3488, 'p': 2156}
red_wine	0.27	{3: 10, 4: 49, 5: 362, 6: 363, 7: 137, 8: 17}	{6: 286, 5: 325, 7: 86, 4: 16, 8: 7, 3: 2}
red_wine	0.55	{3: 10, 4: 53, 5: 567, 6: 527, 7: 165, 8: 17}	{6: 526, 5: 566, 4: 53, 7: 165, 8: 17, 3: 10}
red_wine	0.73	{3: 10, 4: 53, 5: 574, 6: 531, 7: 165, 8: 17}	{5: 574, 7: 165, 6: 531, 4: 53, 8: 17, 3: 10}
red_wine	1.00	{3: 10, 4: 53, 5: 577, 6: 535, 7: 167, 8: 17}	{5: 577, 7: 167, 6: 535, 3: 10, 4: 53, 8: 17}
white_wine	0.27	{3: 20, 4: 139, 5: 672, 6: 897, 7: 426, 8: 114, 9: 5}	{6: 830, 5: 411, 7: 200, 4: 24, 8: 26, 3: 4, 9: 0}
white_wine	0.55	{3: 20, 4: 150, 5: 1156, 6: 1756, 7: 672, 8: 130, 9: 5}	{8: 130, 5: 1152, 7: 670, 6: 1748, 4: 150, 3: 20, 9: 5}
white_wine	0.73	{3: 20, 4: 152, 5: 1163, 6: 1767, 7: 679, 8: 131, 9: 5}	{6: 1766, 5: 1163, 7: 678, 8: 131, 9: 5, 4: 152, 3: 20}
white_wine	1.00	{3: 20, 4: 153, 5: 1175, 6: 1788, 7: 689, 8: 131, 9: 5}	{7: 689, 6: 1788, 5: 1175, 8: 131, 4: 153, 3: 20, 9: 5}
wine_merged	0.25	{3: 23, 4: 109, 5: 290, 6: 318, 7: 203, 8: 73, 9: 5}	{6: 310, 5: 130, 7: 6}
wine_merged	0.50	{3: 30, 4: 203, 5: 1693, 6: 2257, 7: 830, 8: 147, 9: 5}	{5: 1680, 6: 2239, 4: 202, 7: 825, 8: 145, 3: 30, 9: 5}
wine_merged	0.75	{3: 30, 4: 205, 5: 1736, 6: 2298, 7: 843, 8: 148, 9: 5}	{5: 1736, 8: 148, 7: 843, 6: 2297, 4: 205, 9: 5, 3: 30}
wine_merged	1.00	{3: 30, 4: 206, 5: 1752, 6: 2323, 7: 856, 8: 148, 9: 5}	{6: 2323, 5: 1752, 7: 856, 4: 206, 8: 148, 3: 30, 9: 5}
adult	0.29	{'<=50K': 12, '>50K': 10}	{'<=50K': 12}
adult	0.50	{'<=50K': 177, '>50K': 112}	{'<=50K': 192, '>50K': 5}
adult	0.79	{'<=50K': 7677, '>50K': 2947}	{'<=50K': 7700, '>50K': 1912}
adult	1.00	{'<=50K': 33973, '>50K': 11202}	{'<=50K': 33970, '>50K': 11200}

The largest changes in balance occur for low-value radii, but for datasets with low heterogeneity, that is, for which granulation strongly reduces the original size. For standard granulation at low granulation radii, we may also notice a lack of observations for classes that originally existed in the dataset. In such cases, we cannot reliably compare the balance for the two techniques. In the table, we can highlight several examples where the balance

changes quite significantly when comparing the same radius for both granulation methods. Selected examples are presented in pie charts below.

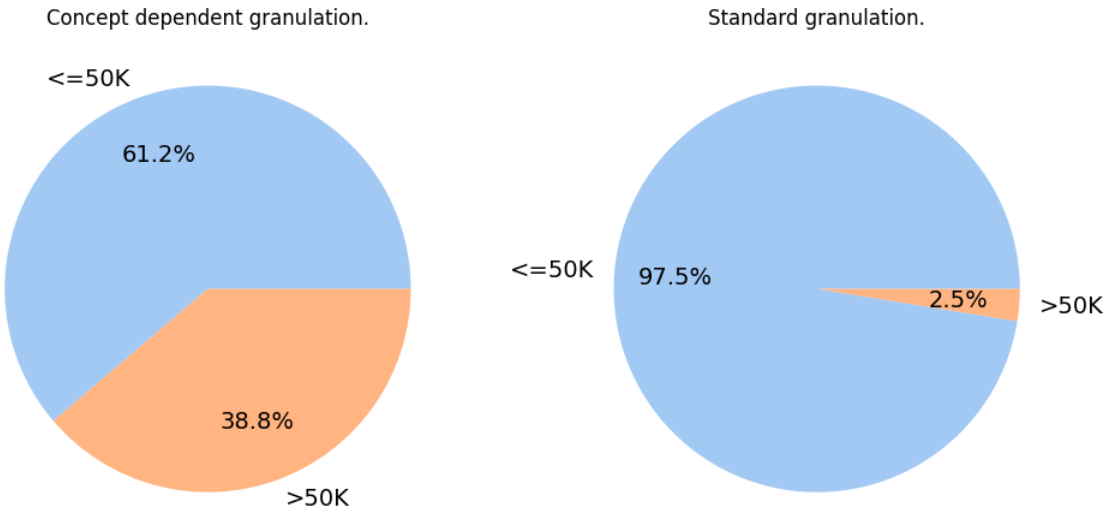


Figure 2.2: Class balance comparison between concept dependent and standard granulation. Adult dataset, radius 0.5.

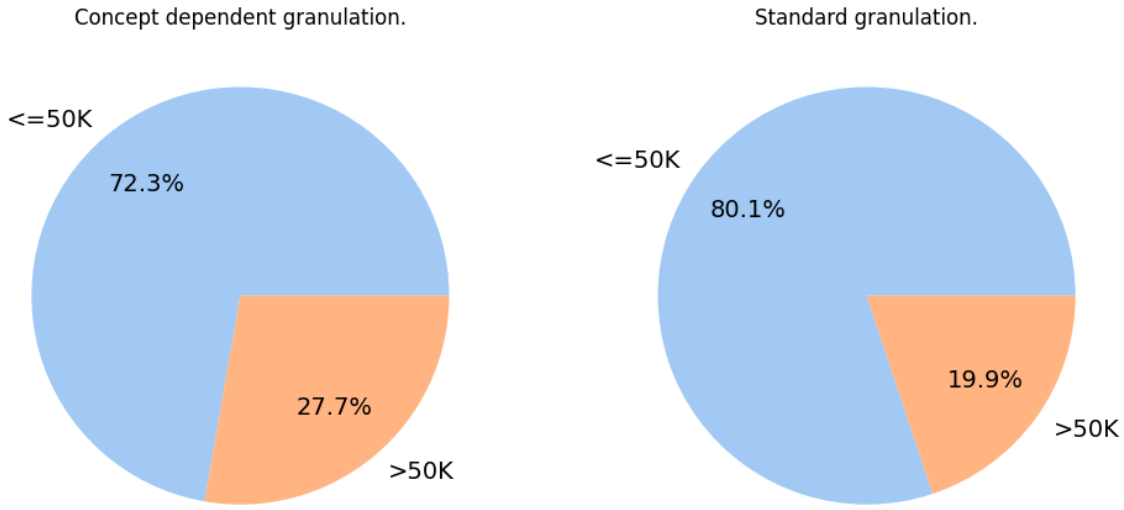


Figure 2.3: Class balance comparison between concept dependent and standard granulation. Adult dataset, radius 0.79.

In each of the above examples, we can see a change in the distribution of classes for the indicated radii. Most of the selected observations do not change the original distribution to a significant degree, but the case for the adult (radius 0.5), pima (radius 0.25), mushroom (radius 0.5) datasets differ in balance between the indicated granulation methods quite significantly. On top of that the standard granulation resulted in large or even extreme

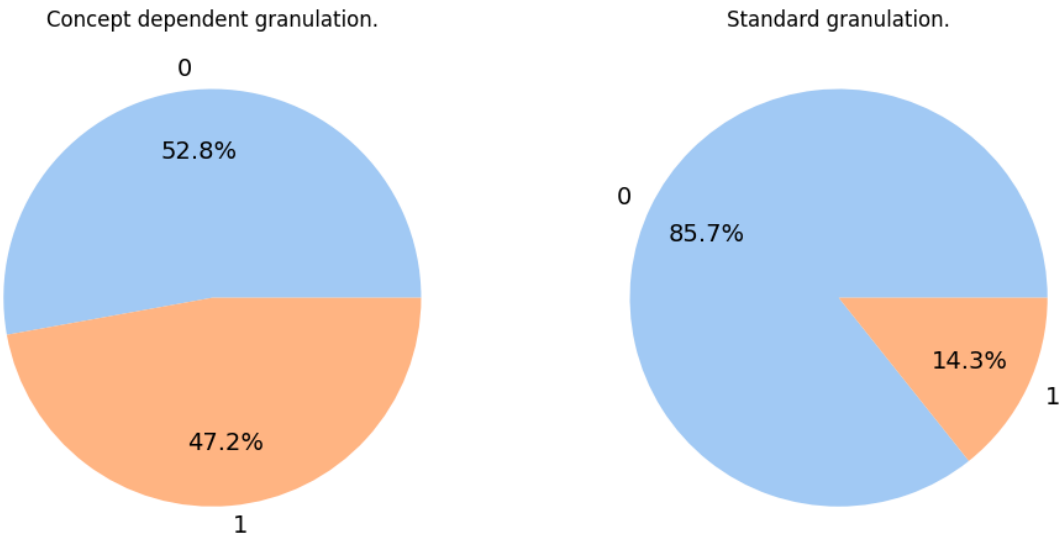


Figure 2.4: Class balance comparison between concept dependent and standard granulation. Pima dataset, radius 0.25.

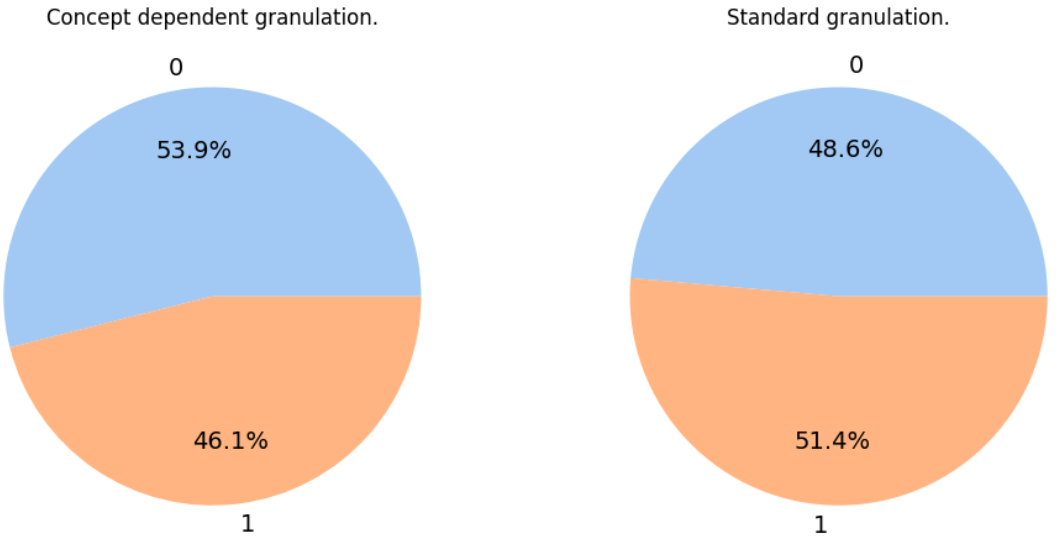


Figure 2.5: Class balance comparison between concept dependent and standard granulation. Heart dataset, radius 0.54.

imbalanced collections, which can have a significant impact on the classification results, which are presented in the next section.

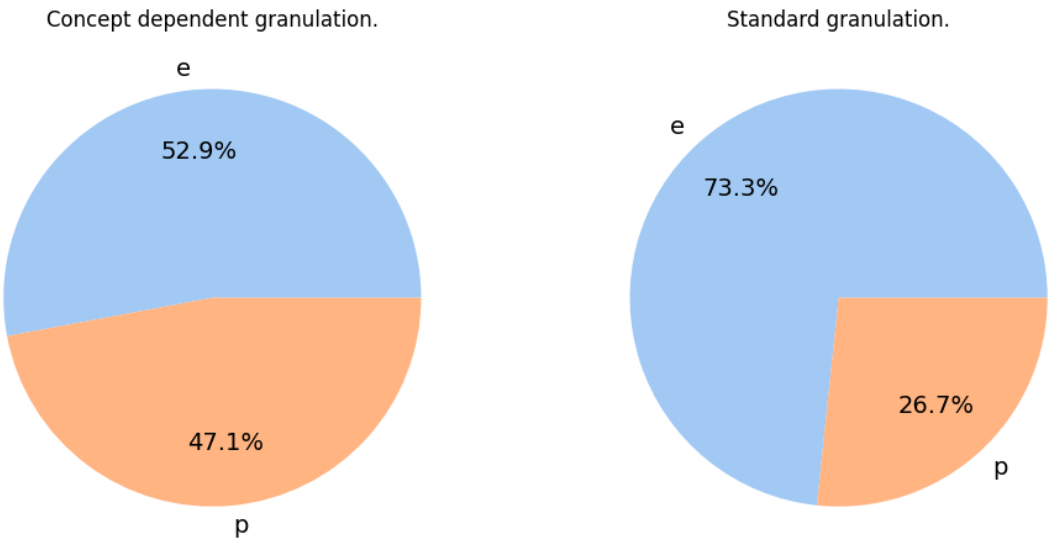


Figure 2.6: Class balance comparison between concept dependent and standard granulation. Mushroom dataset, radius 0.5.

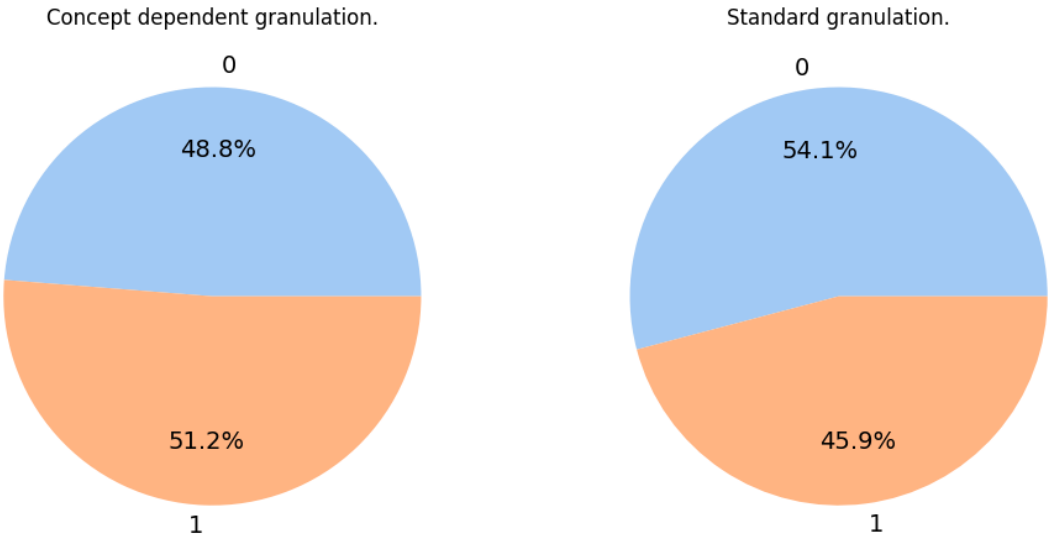


Figure 2.7: Class balance comparison between concept dependent and standard granulation. Australian dataset, radius 0.5.

2.4.7. Classification results

Methodology

The first phase of the experiments in this section consisted of performing classification with the selected algorithms on all the previously described datasets using the permutation test, here tenfold.

Eight popular algorithms were selected to perform the classification:

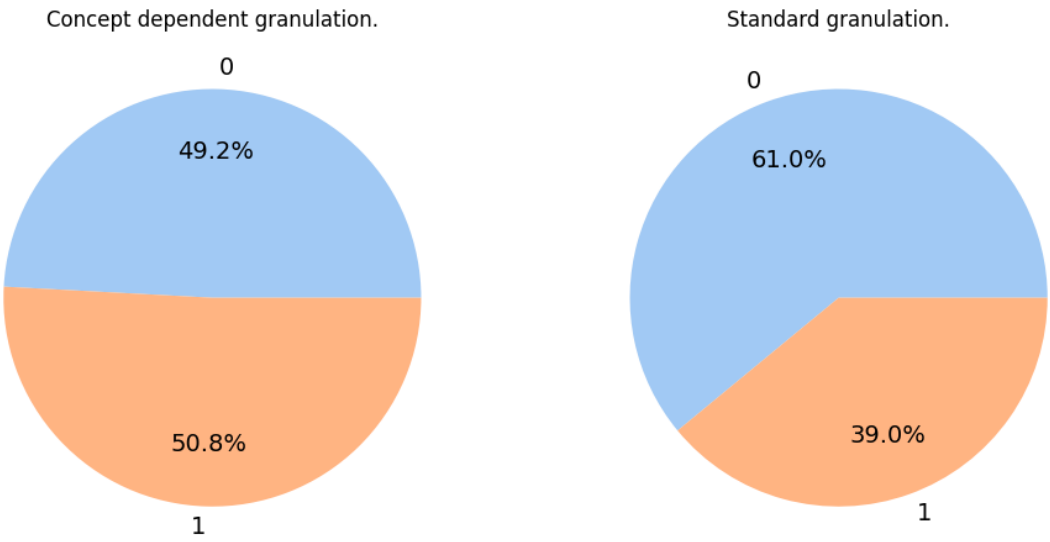


Figure 2.8: Class balance comparison between concept dependent and standard granulation. Australian dummy dataset, radius 0.76.

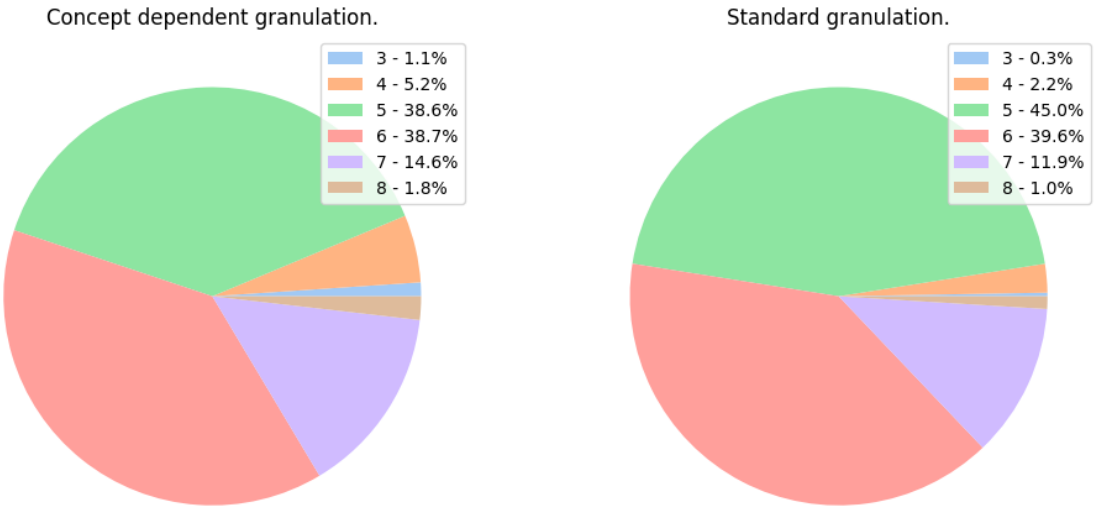


Figure 2.9: Class balance comparison between concept dependent and standard granulation. Red wine dataset, radius 0.27.

- K-nearest neighbor classifier (KNN)
- Gradient Boosting Classifier (XGBoost)
- Decision tree
- Logistic Regression
- Naive Bayes Classifier
- Support Vector Machines (SVM)
- Random Forest

— Multi Layer Perceptron (MLP)

The entire solution was implemented using the Python language and the popular machine learning library sklearn (urlhttps://scikit-learn.org/), which includes, among other things, an implementation of the aforementioned classifiers.
Diagram showing the flow of the entire experiment.

Figure 2.10: Classification nil-case experiment pipeline.

Pseudocode for the presented pipeline.

Algorithm 1 Nil case classification pipeline.

$dataset \leftarrow [...]$
$classifiers \leftarrow [...]$
$results \leftarrow []$

for all $ds \in datasets$ **do**
  for all $n \in \{1, \ldots, 10\}$ **do**
    $trn, tst \leftarrow splitdata(ds)$
    for all $classifier \in classifiers$ **do**
      $classifier.fit(trn)$
      $results \leftarrow results + classifier.predict(tst)$

Commonly used metrics shown below were used as measures of classification.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.9}$$

$$Precision = \frac{TP}{TP + FP} \tag{2.10}$$

$$Recall = Sensitivity = \frac{TP}{TP + FN} \tag{2.11}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{2.12}$$

$$Specificity = \frac{TN}{FP + TN} \tag{2.13}$$

$$Balanced\ accuracy = \frac{Sensitivity + Specificity}{2} \tag{2.14}$$

Balanced accuracy was used as the main measure when comparing results for original and granulated datasets.

Nil-case

A total of 11 datasets, 8 classifiers and 10 permutations of the random division of the data were selected giving a total of 880 classification results. This table is too large to be presented here in full, so the data was grouped and averaged for each dataset and classifier.

Table 2.26: Classification results for nil case (original data) for all selected datasets.

dataset	classifier	acc	balanced acc	precision	recall	f1
adult	decision_tree	0.81	0.75	0.74	0.75	0.74
	knn	0.77	0.62	0.68	0.62	0.63
	logistic_regression	0.79	0.63	0.73	0.63	0.64
	mlp	0.75	0.50	0.62	0.50	0.43
	naive_bayes	0.79	0.63	0.73	0.63	0.64
	random_forest	0.84	0.72	0.83	0.72	0.75
	svm	0.75	0.50	0.68	0.50	0.43
	xgboost	0.86	0.79	0.81	0.79	0.80
australian	decision_tree	0.82	0.81	0.81	0.81	0.81
	knn	0.68	0.67	0.68	0.67	0.67
	logistic_regression	0.86	0.86	0.86	0.86	0.86
	mlp	0.66	0.65	0.67	0.65	0.64
	naive_bayes	0.79	0.78	0.81	0.78	0.78
	random_forest	0.86	0.86	0.86	0.86	0.86
	svm	0.56	0.50	0.54	0.50	0.38
	xgboost	0.85	0.85	0.85	0.85	0.85

Continued on next page

dataset	classifier	acc	balanced acc	precision	recall	f1
australian_dummy	decision_tree	0.81	0.81	0.81	0.81	0.81
	knn	0.66	0.65	0.66	0.65	0.65
	logistic_regression	0.86	0.86	0.86	0.86	0.86
	mlp	0.69	0.68	0.68	0.68	0.68
	naive_bayes	0.81	0.81	0.82	0.81	0.81
	random_forest	0.87	0.86	0.87	0.86	0.86
	svm	0.54	0.52	0.52	0.52	0.45
	xgboost	0.86	0.86	0.86	0.86	0.86
breast	decision_tree	0.93	0.92	0.92	0.92	0.92
	knn	0.93	0.92	0.93	0.92	0.93
	logistic_regression	0.95	0.95	0.95	0.95	0.95
	mlp	0.61	0.58	0.45	0.58	0.46
	naive_bayes	0.94	0.93	0.94	0.93	0.93
	random_forest	0.95	0.95	0.96	0.95	0.95
	svm	0.63	0.50	0.32	0.50	0.39
	xgboost	0.96	0.95	0.96	0.95	0.95
heart	decision_tree	0.75	0.74	0.75	0.74	0.74
	knn	0.64	0.63	0.64	0.63	0.63
	logistic_regression	0.83	0.83	0.84	0.83	0.83
	mlp	0.65	0.64	0.68	0.64	0.62
	naive_bayes	0.82	0.81	0.82	0.81	0.81
	random_forest	0.83	0.83	0.84	0.83	0.83
	svm	0.54	0.50	0.45	0.50	0.36
	xgboost	0.78	0.78	0.78	0.78	0.78
iris	decision_tree	0.96	0.96	0.96	0.96	0.96
	knn	0.96	0.96	0.96	0.96	0.96
	logistic_regression	0.97	0.97	0.97	0.97	0.97
	mlp	0.97	0.97	0.97	0.97	0.97
	naive_bayes	0.95	0.96	0.96	0.96	0.95
	random_forest	0.96	0.96	0.96	0.96	0.96
	svm	0.97	0.97	0.97	0.97	0.97
	xgboost	0.95	0.96	0.95	0.96	0.95

Continued on next page

dataset	classifier	acc	balanced acc	precision	recall	f1
mushroom_num	decision_tree	1.00	1.00	1.00	1.00	1.00
	knn	1.00	1.00	1.00	1.00	1.00
	logistic_regression	0.96	0.96	0.97	0.96	0.96
	mlp	1.00	1.00	1.00	1.00	1.00
	naive_bayes	0.71	0.62	0.80	0.62	0.61
	random_forest	0.99	0.98	0.99	0.98	0.99
	svm	1.00	1.00	1.00	1.00	1.00
	xgboost	1.00	1.00	1.00	1.00	1.00
pima	decision_tree	0.69	0.66	0.66	0.66	0.66
	knn	0.73	0.68	0.70	0.68	0.68
	logistic_regression	0.76	0.72	0.75	0.72	0.73
	mlp	0.68	0.64	0.66	0.64	0.63
	naive_bayes	0.75	0.71	0.73	0.71	0.72
	random_forest	0.76	0.71	0.75	0.71	0.72
	svm	0.64	0.50	0.32	0.50	0.39
	xgboost	0.72	0.69	0.70	0.69	0.69
red_wine	decision_tree	0.58	0.33	0.32	0.33	0.32
	knn	0.50	0.23	0.25	0.23	0.23
	logistic_regression	0.59	0.27	0.29	0.27	0.27
	mlp	0.52	0.22	0.24	0.22	0.21
	naive_bayes	0.54	0.31	0.30	0.31	0.30
	random_forest	0.60	0.26	0.32	0.26	0.26
	svm	0.55	0.25	0.29	0.25	0.25
	xgboost	0.55	0.30	0.32	0.30	0.30
white_wine	decision_tree	0.57	0.33	0.33	0.33	0.33
	knn	0.47	0.23	0.30	0.23	0.24
	logistic_regression	0.49	0.20	0.24	0.20	0.20
	mlp	0.47	0.18	0.20	0.18	0.16
	naive_bayes	0.44	0.29	0.29	0.29	0.26
	random_forest	0.54	0.21	0.24	0.21	0.20
	svm	0.55	0.24	0.47	0.24	0.26
	xgboost	0.49	0.29	0.31	0.29	0.29

Continued on next page

dataset	classifier	acc	balanced acc	precision	recall	f1
wine_merged	decision_tree	0.57	0.33	0.32	0.33	0.32
	knn	0.47	0.21	0.27	0.21	0.22
	logistic_regression	0.51	0.19	0.25	0.19	0.18
	mlp	0.47	0.19	0.25	0.19	0.17
	naive_bayes	0.42	0.26	0.24	0.26	0.22
	random_forest	0.55	0.21	0.25	0.21	0.19
	svm	0.54	0.24	0.48	0.24	0.26
	xgboost	0.46	0.28	0.27	0.27	0.26

The poor quality of the classifiers for some of the granulated sets may be due to the strong imbalance of the original sets, which were further reduced in size after granulation. Decision class weighting techniques were not used for the classifiers used here.

Concept dependent granulation

Flow diagram for the granulation experiment case.

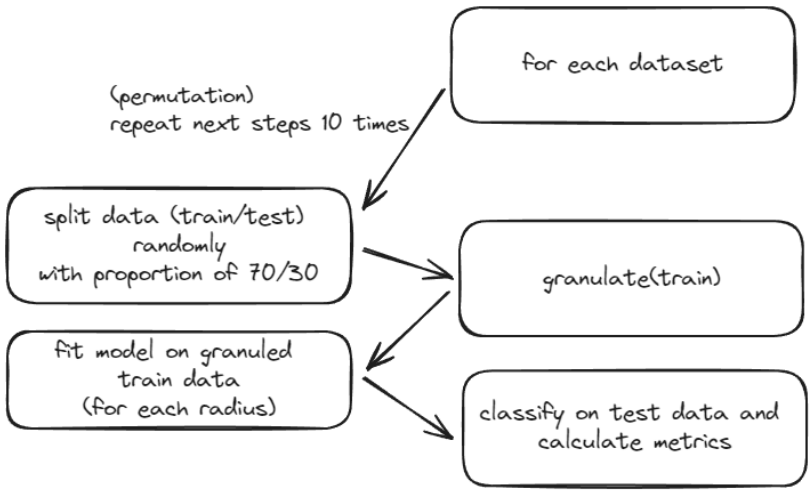


Figure 2.11: Classification of granuled datasets experiment pipeline.

As a result of classification experiments for each dataset, radius and classifier, but taking into account the rules limiting the selection of a specific reflective set, 14214 unit classification results were obtained for a tenfold permutation. The grouped and averaged results resulted in 1424 records. Rejected results that did not pass the predefined rules described below, gave a final set of results of 1419.

At the experimental stage, the following rules were added, the occurrence of which omitted a given reflection set from the classification process. This was especially true at low granulation radii, where the number of objects after granulation can be very low.

Rule 1.

For the classifier **knn**, if the training set size is $< k$ (default is 5) $\rightarrow k = |train|$.

Rule 2.

The classifier **mlp**, which in the experimental session uses the early stopping technique, the activation of which causes the separation of additional 10% of the training set into a validation set (the division is done according to the distribution of classes), with too few (<6) observations in a single class, resulted in no objects being assigned to this set. Thus, sets whose least numerous class had less than 6 objects were rejected for this classifier.

Table comparing the results of nil-case classification and classification of granular collections using the concept dependent method.

Table 2.27: Classification results comparison between nil-case and concept dependent granulation with radius equal to 1.

dataset	classifier	nil balanced acc.	cd. gran. balanced acc.	% point diff.
adult	decision_tree	0.747	0.746	-0.001
adult	knn	0.616	0.620	0.004
adult	logistic_regression	0.627	0.616	-0.011
adult	mlp	0.502	0.502	0.000
adult	naive_bayes	0.627	0.625	-0.002
adult	random_forest	0.717	0.714	-0.003
adult	svm	0.502	0.502	0.000
adult	xgboost	0.790	0.786	-0.004
australian	decision_tree	0.813	0.820	0.007
australian	knn	0.667	0.674	0.007
australian	logistic_regression	0.859	0.856	-0.003
australian	mlp	0.654	0.510	-0.144
australian	naive_bayes	0.777	0.780	0.003
australian	random_forest	0.857	0.862	0.005
australian	svm	0.502	0.499	-0.003
australian	xgboost	0.850	0.857	0.007

Continued on next page

dataset	classifier	nil balanced acc.	cd. gran. balanced acc.	% point diff.
australian_dummy	decision_tree	0.808	0.804	-0.004
australian_dummy	knn	0.651	0.660	0.009
australian_dummy	logistic_regression	0.861	0.857	-0.004
australian_dummy	mlp	0.679	0.506	-0.173
australian_dummy	naive_bayes	0.805	0.798	-0.007
australian_dummy	random_forest	0.860	0.865	0.005
australian_dummy	svm	0.521	0.515	-0.006
australian_dummy	xgboost	0.861	0.860	-0.001
breast	decision_tree	0.921	0.910	-0.011
breast	knn	0.922	0.917	-0.005
breast	logistic_regression	0.948	0.946	-0.002
breast	mlp	0.578	0.500	-0.078
breast	naive_bayes	0.927	0.927	0.000
breast	random_forest	0.948	0.938	-0.010
breast	svm	0.500	0.500	0.000
breast	xgboost	0.953	0.945	-0.008
heart	decision_tree	0.744	0.753	0.009
heart	knn	0.634	0.649	0.015
heart	logistic_regression	0.828	0.845	0.017
heart	mlp	0.638	0.500	-0.138
heart	naive_bayes	0.812	0.831	0.019
heart	random_forest	0.831	0.845	0.014
heart	svm	0.504	0.508	0.004
heart	xgboost	0.779	0.791	0.012
iris	decision_tree	0.957	0.947	-0.010
iris	knn	0.959	0.962	0.003
iris	logistic_regression	0.968	0.958	-0.010
iris	mlp	0.969	0.918	-0.051
iris	naive_bayes	0.955	0.944	-0.011
iris	random_forest	0.957	0.949	-0.008
iris	svm	0.969	0.973	0.004
iris	xgboost	0.955	0.964	0.009

Continued on next page

dataset	classifier	nil balanced acc.	cd. gran. balanced acc.	% point diff.
mushroom_num	decision_tree	1.000	1.000	0.000
mushroom_num	knn	0.999	1.000	0.001
mushroom_num	logistic_regression	0.960	0.963	0.003
mushroom_num	mlp	0.999	0.998	-0.001
mushroom_num	naive_bayes	0.625	0.618	-0.007
mushroom_num	random_forest	0.983	0.981	-0.002
mushroom_num	svm	1.000	1.000	0.000
mushroom_num	xgboost	1.000	1.000	0.000
pima	decision_tree	0.663	0.665	0.002
pima	knn	0.680	0.669	-0.011
pima	logistic_regression	0.718	0.712	-0.006
pima	mlp	0.636	0.500	-0.136
pima	naive_bayes	0.711	0.711	0.000
pima	random_forest	0.712	0.703	-0.009
pima	svm	0.500	0.500	0.000
pima	xgboost	0.688	0.694	0.006
red_wine	decision_tree	0.328	0.341	0.013
red_wine	knn	0.230	0.228	-0.002
red_wine	logistic_regression	0.271	0.267	-0.004
red_wine	mlp	0.221	0.167	-0.054
red_wine	naive_bayes	0.311	0.326	0.015
red_wine	random_forest	0.260	0.263	0.003
red_wine	svm	0.248	0.256	0.008
red_wine	xgboost	0.303	0.351	0.048
white_wine	decision_tree	0.331	0.361	0.030
white_wine	knn	0.229	0.210	-0.019
white_wine	logistic_regression	0.200	0.192	-0.008
white_wine	naive_bayes	0.287	0.299	0.012
white_wine	random_forest	0.212	0.211	-0.001
white_wine	svm	0.243	0.230	-0.013
white_wine	xgboost	0.286	0.311	0.025
wine_merged	decision_tree	0.333	0.346	0.013

Continued on next page

Table 2.28: Mean classification change for concept dependent granulation classification vs. nil-case.

classifier	perc. point diff. mean
decision_tree	0.00436
knn	-0.00027
logistic_regression	-0.00218
mlp	-0.08611
naive_bayes	0.00536
random_forest	-0.00073
svm	-0.00218
xgboost	0.01000

dataset	classifier	nil balanced acc.	cd. gran. balanced acc.	% point diff.
wine_merged	knn	0.214	0.209	-0.005
wine_merged	logistic_regression	0.191	0.195	0.004
wine_merged	naive_bayes	0.264	0.301	0.037
wine_merged	random_forest	0.208	0.206	-0.002
wine_merged	svm	0.244	0.226	-0.018
wine_merged	xgboost	0.275	0.291	0.016

The results for the compared cases do not differ significantly except in a few cases for the mlp and svm classifiers. However, it should be noted that the measures for these classifiers for nil-case are also low and often at the level of random results given the number of classes in a given set. 43 balanced accuracy values were lower than for nil-case, 9 did not change, and 34 were higher.

When we compare the average change for the classifier, we can see that most of the changes are not statistically significant. The only significant difference is an 8 percentage point decrease in the accuracy of the mlp model, which tends to bring the final result closer to a result equal to the random selection of the label.

Due to the different number of attributes in the test sets, it is not always possible to compare results for the same radii, but it was assumed that a comparison of results for a radius as close to 0.5 as possible (that is, granulation for objects similar in about half of their attributes) would be used to compare results for the two granulation methods presented - standard and concept dependent. However, it will not be possible to compare this to the homogeneous granulation presented in 3.11, since there the granulation radius is not chosen a priori, but for each object separately.

For the calculation of the results shown below, it was assumed that the data was selected for radii equal to 0.5, and if a given set did not have an even number of attributes then the value of the metrics was taken for the two radii closest to 0.5 and averaged.

Table 2.29: Classification results comparison between nil-case and concept dependent granulation with radius equal to 0.5.

dataset	classifier	nil balanced acc	cdgran balanced acc	perc. point diff.
adult	decision_tree	0.747	0.681	-0.066
adult	knn	0.616	0.503	-0.113
adult	logistic_regression	0.627	0.609	-0.018
adult	mlp	0.502	0.520	0.018
adult	naive_bayes	0.627	0.577	-0.050
adult	random_forest	0.717	0.751	0.034
adult	svm	0.502	0.500	-0.002
adult	xgboost	0.790	0.727	-0.063
australian	decision_tree	0.813	0.667	-0.146
australian	knn	0.667	0.568	-0.099
australian	logistic_regression	0.859	0.771	-0.088
australian	mlp	0.654	0.500	-0.154
australian	naive_bayes	0.777	0.720	-0.057
australian	random_forest	0.857	0.779	-0.078
australian	svm	0.502	0.513	0.011
australian	xgboost	0.850	0.672	-0.178
australian_dummy	decision_tree	0.808	0.557	-0.251
australian_dummy	knn	0.651	0.500	-0.151
australian_dummy	logistic_regression	0.861	0.483	-0.378
australian_dummy	naive_bayes	0.805	0.483	-0.322
australian_dummy	random_forest	0.860	0.748	-0.112
australian_dummy	svm	0.521	0.530	0.009
australian_dummy	xgboost	0.861	0.745	-0.116
breast	decision_tree	0.921	0.911	-0.010
breast	knn	0.922	0.917	-0.005
breast	logistic_regression	0.948	0.945	-0.003

Continued on next page

dataset	classifier	nil balanced acc	cdgran balanced acc	perc. point diff.
breast	mlp	0.578	0.502	-0.076
breast	naive_bayes	0.927	0.927	0.000
breast	random_forest	0.948	0.939	-0.009
breast	svm	0.500	0.500	0.000
breast	xgboost	0.953	0.941	-0.012
heart	decision_tree	0.744	0.680	-0.064
heart	knn	0.634	0.621	-0.013
heart	logistic_regression	0.828	0.752	-0.076
heart	mlp	0.638	0.498	-0.140
heart	naive_bayes	0.812	0.756	-0.056
heart	random_forest	0.831	0.768	-0.063
heart	svm	0.504	0.502	-0.002
heart	xgboost	0.779	0.689	-0.090
iris	decision_tree	0.957	0.938	-0.019
iris	knn	0.959	0.962	0.003
iris	logistic_regression	0.968	0.964	-0.004
iris	mlp	0.969	0.922	-0.047
iris	naive_bayes	0.955	0.949	-0.006
iris	random_forest	0.957	0.947	-0.010
iris	svm	0.969	0.976	0.007
iris	xgboost	0.955	0.940	-0.015
mushroom_num	decision_tree	1.000	0.805	-0.195
mushroom_num	knn	0.999	0.786	-0.213
mushroom_num	logistic_regression	0.960	0.748	-0.212
mushroom_num	mlp	0.999	0.601	-0.398
mushroom_num	naive_bayes	0.625	0.589	-0.036
mushroom_num	random_forest	0.983	0.828	-0.155
mushroom_num	svm	1.000	0.762	-0.238
mushroom_num	xgboost	1.000	0.805	-0.195
pima	decision_tree	0.663	0.656	-0.007
pima	knn	0.680	0.670	-0.010
pima	logistic_regression	0.718	0.711	-0.007

Continued on next page

dataset	classifier	nil balanced acc	cdgran balanced acc	perc. point diff.
pima	mlp	0.636	0.500	-0.136
pima	naive_bayes	0.711	0.707	-0.004
pima	random_forest	0.712	0.697	-0.015
pima	svm	0.500	0.500	0.000
pima	xgboost	0.688	0.697	0.009
red_wine	decision_tree	0.328	0.346	0.018
red_wine	knn	0.230	0.227	-0.003
red_wine	logistic_regression	0.271	0.267	-0.004
red_wine	mlp	0.221	0.166	-0.055
red_wine	naive_bayes	0.311	0.325	0.014
red_wine	random_forest	0.260	0.261	0.001
red_wine	svm	0.248	0.256	0.008
red_wine	xgboost	0.303	0.342	0.039
white_wine	decision_tree	0.331	0.356	0.025
white_wine	knn	0.229	0.210	-0.019
white_wine	logistic_regression	0.200	0.194	-0.006
white_wine	naive_bayes	0.287	0.298	0.011
white_wine	random_forest	0.212	0.209	-0.003
white_wine	svm	0.243	0.229	-0.014
white_wine	xgboost	0.286	0.323	0.037
wine_merged	decision_tree	0.333	0.345	0.012
wine_merged	knn	0.214	0.209	-0.005
wine_merged	logistic_regression	0.191	0.195	0.004
wine_merged	naive_bayes	0.264	0.301	0.037
wine_merged	random_forest	0.208	0.205	-0.003
wine_merged	svm	0.244	0.225	-0.019
wine_merged	xgboost	0.275	0.261	-0.014

The results of classifying granulated data with the concept dependent method yielded a reduction in balanced accuracy values. If we compare this with the results for standard granulation, the results are worse. A special case of deterioration in results is the australian dummy set (this is the australian set that was encoded using the one hot method), for which the results dropped for some classifiers by more than 30 percentage points.

An explanation can be found in the table 2.30, which shows the percentage of dataset reduction after concept dependent granulation for a radius of 0.5, and notes that for the australian dummy set, the average dataset size after granulation is 2, so one object per decision class. In practice, using any complex classifier on such a number of observations does not make much sense, but it shows that even for such a drastic reduction of the dataaset size with the presented granulation methods, classifiers such as random forest and xgboost can achieve about 75% classification efficiency.

Table 2.30: Granuled dataset sizes for concept dependent granulation for radius 0.5.

dataset	radius	% of all objects	objects_total_mean
adult	0.5	0.639	289.100
australian	0.5	11.551	79.700
australian_dummy	0.5	0.290	2.000
breast	0.5	100.000	569.000
iris	0.5	46.867	70.300
mushroom	0.5	0.292	16.500
pima	0.5	86.380	663.400
wine_merged	0.5	79.495	5164.800
heart	0.5	21.502	65.150
mushroom	0.5	0.293	16.533
red_wine	0.5	83.271	1331.500
white_wine	0.5	78.925	3865.750

The column $\%\ of\ all\ objects$ represents the % of objects that are in the granular set for a given dataset and radius of 0.5, and the $objects\_total\_mean$ column is the average number of objects that were there after a 10-fold granulation.
If we average the results for each classifier we get the results presented in table 2.31.

Table 2.31: Mean percent point balanced accuracy bias for concept dependent granuled data for radius 0.5 vs nil-case classification.

classifier	perc point diff mean
decision_tree	-0.06391
knn	-0.05709
logistic_regression	-0.07200
mlp	-0.12350
naive_bayes	-0.04264
random_forest	-0.03755
svm	-0.02182
xgboost	-0.05436

We see that the results for concept dependent granulation for radius 1.0 and 0.5 are lower than for nil-case classification. The average difference for radius 1.0 is negligible, but given that concept-dependent granulation for this radius in most of the tested datasets did not reduce their size significantly (see 2.20) this is not unusual. The situation is different for radius 0.5, where the decrease in the size of granular sets is not uncommon, and the decrease in the classification measure is is disproportionately smaller.

Standard granulation

Algorithm 2 Standard granulation classification pipeline.

$dataset \leftarrow [...]$
$classifiers \leftarrow [...]$
$results \leftarrow []$

for all $ds \in datasets$ **do**
  for all $n \in \{1, \ldots, 10\}$ **do**
    $trn, tst \leftarrow splitdata(ds)$
    for all $classifier \in classifiers$ **do**
      $classifier.fit(trn)$
      $results \leftarrow results + classifier.predict(tst)$

Taking into account the fact that the standard granulation process can lead, for low granulation radii, to a situation in which no new objects are created for the originally existing class in the reflection set, part of these sets could not be used to perform meaningful classification. For this reason, in addition to the rules presented in 2.4.7 and 2.4.7, one more rule was added.

Rule 3.

Reflective sets (i.e., after granulation) that contain less than two different decision classes are ignored.

Taking into account the above limitations, a total of 10177 individual results were collected in the classification process of sets granulated using the standard method. The granulation process was carried out 10 times for each set, so in order to average the results, the partial results should be grouped. After grouping, a set of 1089 results was obtained.

Due to the nondeterminism of the granulation process, part of the sets with the same radius were rejected by the above-mentioned rules, and part were subjected to classification. As a result, some of the results after grouping do not consist of 10 permutations of classification for a given radius, and these results were discarded from the final summary. The final grouped and filtered set has 973 classification results. This is too many to present in full here, so below is a summary of these results and a visualization of a selection of them.

Comparison of classification results by balanced accuracy measure for nil-case classification and standard granulation for radius equal to 1. It should be noted here that nominally after grouping the data there should be 88 results (11 sets * 8 classifiers), but two of them were rejected by rule 3 under 2.4.7 and are not present in the table below.

Table 2.32: Classification results comparison between nil-case and standard granulation with radius equal to 1.

dataset	classifier	nil balanced acc.	std. gran. balanced acc.	% point diff.
adult	decision_tree	0.747	0.748	0.001
adult	knn	0.616	0.622	0.006
adult	logistic_regression	0.627	0.622	-0.005
adult	mlp	0.502	0.501	-0.001
adult	naive_bayes	0.627	0.628	0.001
adult	random_forest	0.717	0.719	0.002
adult	svm	0.502	0.502	0.000
adult	xgboost	0.790	0.789	-0.001
australian	decision_tree	0.813	0.803	-0.010
australian	knn	0.667	0.656	-0.011
australian	logistic_regression	0.859	0.844	-0.015
australian	mlp	0.654	0.502	-0.152
australian	naive_bayes	0.777	0.752	-0.025
australian	random_forest	0.857	0.848	-0.009
australian	svm	0.502	0.504	0.002
australian	xgboost	0.850	0.850	0.000
australian_dummy	decision_tree	0.808	0.807	-0.001
australian_dummy	knn	0.651	0.663	0.012
australian_dummy	logistic_regression	0.861	0.858	-0.003
australian_dummy	mlp	0.679	0.517	-0.162
australian_dummy	naive_bayes	0.805	0.804	-0.001
australian_dummy	random_forest	0.860	0.857	-0.003
australian_dummy	svm	0.521	0.516	-0.005
australian_dummy	xgboost	0.861	0.838	-0.023
breast	decision_tree	0.921	0.919	-0.002

Continued on next page

dataset	classifier	nil balanced acc.	std. gran. balanced acc.	% point diff.
breast	knn	0.922	0.918	-0.004
breast	logistic_regression	0.948	0.936	-0.012
breast	mlp	0.578	0.502	-0.076
breast	naive_bayes	0.927	0.923	-0.004
breast	random_forest	0.948	0.945	-0.003
breast	svm	0.500	0.500	0.000
breast	xgboost	0.953	0.949	-0.004
heart	decision_tree	0.744	0.735	-0.009
heart	knn	0.634	0.620	-0.014
heart	logistic_regression	0.828	0.815	-0.013
heart	mlp	0.638	0.500	-0.138
heart	naive_bayes	0.812	0.810	-0.002
heart	random_forest	0.831	0.821	-0.010
heart	svm	0.504	0.504	0.000
heart	xgboost	0.779	0.783	0.004
iris	decision_tree	0.957	0.929	-0.028
iris	knn	0.959	0.962	0.003
iris	logistic_regression	0.968	0.956	-0.012
iris	mlp	0.969	0.931	-0.038
iris	naive_bayes	0.955	0.956	0.001
iris	random_forest	0.957	0.947	-0.010
iris	svm	0.969	0.960	-0.009
iris	xgboost	0.955	0.940	-0.015
mushroom_num	decision_tree	1.000	1.000	0.000
mushroom_num	knn	0.999	0.999	0.000
mushroom_num	logistic_regression	0.960	0.962	0.002
mushroom_num	mlp	0.999	0.999	0.000
mushroom_num	naive_bayes	0.625	0.622	-0.003
mushroom_num	random_forest	0.983	0.981	-0.002
mushroom_num	svm	1.000	1.000	0.000
mushroom_num	xgboost	1.000	1.000	0.000
pima	decision_tree	0.663	0.693	0.030

Continued on next page

dataset	classifier	nil balanced acc.	std. gran. balanced acc.	% point diff.
pima	knn	0.680	0.687	0.007
pima	logistic_regression	0.718	0.718	0.000
pima	mlp	0.636	0.500	-0.136
pima	naive_bayes	0.711	0.717	0.006
pima	random_forest	0.712	0.698	-0.014
pima	svm	0.500	0.500	0.000
pima	xgboost	0.688	0.702	0.014
red_wine	decision_tree	0.328	0.345	0.017
red_wine	knn	0.230	0.225	-0.005
red_wine	logistic_regression	0.271	0.259	-0.012
red_wine	mlp	0.221	0.167	-0.054
red_wine	naive_bayes	0.311	0.321	0.010
red_wine	random_forest	0.260	0.256	-0.004
red_wine	svm	0.248	0.257	0.009
red_wine	xgboost	0.303	0.344	0.041
white_wine	decision_tree	0.331	0.357	0.026
white_wine	knn	0.229	0.215	-0.014
white_wine	logistic_regression	0.200	0.192	-0.008
white_wine	naive_bayes	0.287	0.285	-0.002
white_wine	random_forest	0.212	0.209	-0.003
white_wine	svm	0.243	0.232	-0.011
white_wine	xgboost	0.286	0.292	0.006
wine_merged	decision_tree	0.333	0.344	0.011
wine_merged	knn	0.214	0.204	-0.010
wine_merged	logistic_regression	0.191	0.195	0.004
wine_merged	naive_bayes	0.264	0.294	0.030
wine_merged	random_forest	0.208	0.205	-0.003
wine_merged	svm	0.244	0.225	-0.019
wine_merged	xgboost	0.275	0.288	0.013

Analyzing the table, we can draw a clear conclusion - the classification of original and granular data using the standard method for the radius of granulation 1 does not bring significant changes in the measure of balanced accuracy, and in most cases it is lower for

granular data, but this difference is within the range of statistical insignificance. The exception is the results of the mlp classifier for the sets of australian, australian dummy, breast, heart and pima where the results are already clearly worse. However, this fits in with the earlier and later presented results for the mlp classifier, which are among the lowest in the experiments conducted. This result is not surprising, as a comparison of the size of the original and standard-granulated harvests presented in the table 2.22 shows that for radius 1 the australian, australian dummy, breast, mushroom and pima datasets did not change their size after granulation. Slightly smaller yields for radius 1 arose for iris dataset (about 98% of the original size), red wine (about 84%), white wine (about 80%) and merged wine (about 80%). Considering the results of just these last three collections, we can conclude that standard granulation has effectively reduced the noise in these datasets, as the classification results are almost identical, although globally very low.
The results for the 0.5 radius were calculated on the same basis as the results presented in the previous subsection for the 0.5 radius of concept dependent granulation.

Table 2.33: Classification results comparison between nil-case and standard granulation with radius equal to 0.5.

dataset	classifier	nil balanced acc.	std. gran. balanced acc.	% point diff.
adult	decision_tree	0.747	0.574	-0.173
adult	knn	0.616	0.500	-0.116
adult	logistic_regression	0.627	0.509	-0.118
adult	naive_bayes	0.627	0.527	-0.100
adult	random_forest	0.717	0.520	-0.197
adult	svm	0.502	0.500	-0.002
adult	xgboost	0.790	0.576	-0.214
australian	decision_tree	0.813	0.767	-0.046
australian	knn	0.667	0.586	-0.081
australian	logistic_regression	0.859	0.689	-0.170
australian	mlp	0.654	0.501	-0.153
australian	naive_bayes	0.777	0.694	-0.083
australian	random_forest	0.857	0.816	-0.041
australian	svm	0.502	0.509	0.007
australian	xgboost	0.850	0.774	-0.076

Continued on next page

dataset	classifier	nil balanced acc.	std. gran. balanced acc.	% point diff.
breast	decision_tree	0.921	0.913	-0.008
breast	knn	0.922	0.918	-0.004
breast	logistic_regression	0.948	0.937	-0.011
breast	mlp	0.578	0.502	-0.076
breast	naive_bayes	0.927	0.923	-0.004
breast	random_forest	0.948	0.945	-0.003
breast	svm	0.500	0.500	0.000
breast	xgboost	0.953	0.950	-0.003
heart	decision_tree	0.744	0.705	-0.039
heart	knn	0.634	0.627	-0.007
heart	logistic_regression	0.828	0.744	-0.084
heart	mlp	0.638	0.499	-0.139
heart	naive_bayes	0.812	0.750	-0.062
heart	random_forest	0.831	0.780	-0.051
heart	svm	0.504	0.502	-0.002
heart	xgboost	0.779	0.712	-0.067
iris	decision_tree	0.957	0.918	-0.039
iris	knn	0.959	0.938	-0.021
iris	logistic_regression	0.968	0.947	-0.021
iris	mlp	0.969	0.847	-0.122
iris	naive_bayes	0.955	0.936	-0.019
iris	random_forest	0.957	0.947	-0.010
iris	svm	0.969	0.951	-0.018
iris	xgboost	0.955	0.938	-0.017
mushroom_num	decision_tree	1.000	0.771	-0.229
mushroom_num	knn	0.999	0.641	-0.358
mushroom_num	logistic_regression	0.960	0.791	-0.169
mushroom_num	naive_bayes	0.625	0.698	0.073
mushroom_num	random_forest	0.983	0.815	-0.168
mushroom_num	svm	1.000	0.726	-0.274
mushroom_num	xgboost	1.000	0.787	-0.213
pima	decision_tree	0.663	0.654	-0.009

Continued on next page

dataset	classifier	nil balanced acc.	std. gran. balanced acc.	% point diff.
pima	knn	0.680	0.668	-0.012
pima	logistic_regression	0.718	0.711	-0.007
pima	mlp	0.636	0.501	-0.135
pima	naive_bayes	0.711	0.702	-0.009
pima	random_forest	0.712	0.696	-0.016
pima	svm	0.500	0.500	0.000
pima	xgboost	0.688	0.697	0.009
red_wine	decision_tree	0.328	0.344	0.016
red_wine	knn	0.230	0.224	-0.006
red_wine	logistic_regression	0.271	0.257	-0.014
red_wine	mlp	0.221	0.168	-0.053
red_wine	naive_bayes	0.311	0.323	0.012
red_wine	random_forest	0.260	0.256	-0.004
red_wine	svm	0.248	0.255	0.007
red_wine	xgboost	0.303	0.318	0.015
white_wine	decision_tree	0.331	0.353	0.022
white_wine	knn	0.229	0.214	-0.015
white_wine	logistic_regression	0.200	0.194	-0.006
white_wine	naive_bayes	0.287	0.286	-0.001
white_wine	random_forest	0.212	0.207	-0.005
white_wine	svm	0.243	0.231	-0.012
white_wine	xgboost	0.286	0.284	-0.002
wine_merged	decision_tree	0.333	0.339	0.006
wine_merged	knn	0.214	0.203	-0.011
wine_merged	logistic_regression	0.191	0.195	0.004
wine_merged	naive_bayes	0.264	0.298	0.034
wine_merged	random_forest	0.208	0.204	-0.004
wine_merged	svm	0.244	0.224	-0.020
wine_merged	xgboost	0.275	0.281	0.006

Classification results for data granulated at radius 0.5 are usually worse than those for radius 1, but quite often on par. When we again take a closer look at the sizes of these sets after granulation then we can draw further conclusions. It will be helpful to summarize the

changes in balance accuracy for radii 1 and 0.5 and the change in size of the original dataset vs. the nil-case classification results.

Table 2.34: Balanced accuracy change for radius 1 and 0.5 along with granular sets sizes change.

dataset	nil acc.	% size r=1.0	acc diff. r=1.0	% size r=0.5	acc diff. r=0.5
adult	0.6609	99.8850	0.0004	0.4361	-0.1314
australian	0.7474	100.0000	-0.0275	8.8696	-0.0804
australian_dummy	0.7558	100.0000	-0.0233	—	0.1451
breast	0.8371	100.0000	-0.0131	100.0000	-0.0136
heart	0.7212	99.6700	-0.0228	16.8317	-0.0564
iris	0.9611	98.0000	-0.0135	44.5333	-0.0334
mushroom_num	0.9381	100.0177	-0.0004	0.2818	-0.1911
pima	0.6635	100.0000	-0.0116	82.7865	-0.0224
red_wine	0.2715	84.9906	0.0002	82.9831	-0.0034
white_wine	0.2554	80.8697	-0.0009	78.4718	-0.0027
wine_merged	0.2470	81.8839	0.0037	78.9118	0.0021

In addition to the obvious $dataset$ column, the $nil\ acc.$ column states for the balanced accuracy for the nil case as a base for comparison to radius 1 and 0.5. The $acc\ diff\ r = 1.0$ column indicates the averaged difference in balanced accuracy compared to nil-case, the $\%size\ r = 1.0$ column indicates what % of the size of the original set is the set after granulation for radius 1. The $acc\ diff.\ r = 0.5$ signifies the averaged difference in balanced accuracy compared to nil-case, and column $\%size\ r = 0.5$ indicates what % of the size of the original set is the set after granulation for radius 0.5. The missing value for the 0.5 radius of the Australian dummy dataset is due to the fact that, due to its very large approximation and the small size of the granulated training set, it was not possible to perform classification with all the selected classifiers. The result was therefore discarded. In the above table, we can see a correlation between the decrease in the balanced accuracy metric and the decrease in the size of the granulated set, but it is not large. This is best seen for the sets whose reduction in the number of objects for granulation with a radius of 0.5 is large, i.e. the adult set (reduction of 99.6 percent), australian (reduction of 91.1 percent), australian dummy (reduction of 99.85 percent), heart (reduction of 83.2 percent) and mushroom (reduction of 99.7 percent), and the decrease in the classification measure is not proportionally big to the decrease in the size of this set. This also confirms the effectiveness of granulation in the process of preserving the information contained in the original dataset allowing low diverse datasets to be well aproximated and used in Machine Learning algorithms with final good results.

2.5. Conclusions

Both of the granularity techniques presented, are effective methods for approximating decision sets, although it is also necessary to properly understand the data in order to effectively select the best classifier for a given dataset. The granularity and size of the final granular set also gives us information about how diverse the set is. The greater the reduction in the original size of the set, the less diverse it is, and vice versa. Proper selection of the granularity radius can effectively help get rid of noise from the data, keeping the quality of the classification at a comparable level.

It should also not be underestimated that granulation can contribute to changing the original balance of classes in the set and may require additional set preparation through, for example, oversampling or undersampling to improve the effectiveness of the Machine Learning model built on the set.

Concept dependent granulation also undoubtedly has the advantage that it takes place within the boundaries of the decision class and objects between classes are not "mixed" between concepts by the majority voting mechanism, and no class will disappear from the dataset.

Standard granularity, on the other hand, can be used on sets whose labels we would like to assign in a way that results from the greatest possible similarity of objects between each other, in cases where they may have originally been assigned arbitrarily or in a random manner not based on observation.

Part II

In search of knowledge granulation techniques with an adaptive mechanism for determining the granulation radius

3. Homogeneous granulation

3.1. Method description

The concept of homogeneous granulation was first proposed in [34], and was used in subsequent studies of its effect on data in the context of its use in the ensemble model [4], in the epsilon variant of [35], and as part of the missing value absorption model [5].
Homogeneous granulation is an approach based on concept dependent granulation, see. 2.2.1, but with a significant difference where it comes to granulation radius selection. In fact the granulation radius is selected in an automatic way. The granulation radius, i.e. indiscernibility level, is extended until formed granule consists only of objects within the same decision class. This approach can be compared to clustering methods where similar objects are located around their centroids - here it is done based on rough set theory, which means that the object belongs to the set (granule) to a certain degree (radius). Taking this into account, it is necessary to get rid of ambiguity in the preprocessing phase, that is, to solve the problem of identical objects within different decision classes.
For the sake of formality, the steps of homogeneous granulation are described below.

1. Input of the Decision-Making System (U - Collection of objects, A - non decision attributes, d - decision),
2. Elimination of completely contradictory objects (identical attribute values with different decision attribute value),
3. The granules are defined as,

$$g_{r_u}^{homogeneous} = \{v \in U : |g_{r_u}^{cd}| - |g_{r_u}| = 0,\ for\ minimal\ r_u\ fulfills\ the\ equation\}$$

where

$$g_{r_u}^{cd} = \{v \in U : \frac{|IND(u,v)|}{|A|} \leq r_u \ \wedge d(u) = d(v)\}$$

and

$$g_{r_u} = \{v \in U : \frac{|IND(u,v)|}{|A|} \leq r_u\}$$

$$r_u \in \{\frac{0}{|A|}, \frac{1}{|A|}, ..., 1\}$$

μ is an rough inclusion,

4. We create granular coverage using the chosen strategy:

   - Selection of granules in a fixed order (hierarchical)

   - random selection,

   - selection by length,

   - selection according to coverage rate,

   - random selection depending on class size.

   Granules that convey at least one new object go into coverage.

   A training system is considered covered when it satisfies the following equation:

$$\bigcup\{g_{r_{gran}}^{cd}(u) \ : \ g_{r_{gran}}^{cd}(u) \in U_{cover}\} = U \tag{3.1}$$

   where U_{cover} denotes the set of granular coverage.

   We form new objects from the granules of cover - using a strategy of our choice, such as *Majority Voting*.

   The granular reflection of the original decision system $D = (U, A, d)$ is the new decision system $(COV(U, \mu, r)$, the set of objects formed from granules.

$$v \in g_r^{cd}(u) \text{ if and only if } \mu(v, u, r) \text{ and } (d(u) = d(v)) \tag{3.2}$$

   for a given rough (weak) inclusion μ.

   The effect of *Majority Voting* can be recorded as:.

$$\{MV(\{a(u) : u \in g\}) : a \in A \cup \{d\}\} \tag{3.3}$$

3.2. Simple example of homogeneous granulation

In our example, we will use the data from Tab. 2.1.

This granulation works analogously to concept-dependent, except that the granules are formed in the following way.

Step 1: forming granules.

Since the radius of granulation is not predetermined, we initialize its value as 4/4 (1.0) (we are looking for objects that are 100% similar to it) and then it will be reduced as defined in the section 3.1. We first take the first object as the central object, which in our case will be an object marked with blue in the table.

Table 3.1: Homogeneous granulation toy example - objects in granule for u_1 and radius of 4/4 (1.0).

	sepal_length	sepal_width	petal_length	petal_width	iris class
1	5.1	3.5	1.4	0.2	1
2	4.9	3.0	1.4	0.2	1
3	4.7	3.2	1.3	0.2	1
4	4.6	3.1	1.5	0.2	1
5	5.0	3.6	1.4	0.2	1
6	7.0	3.2	4.7	1.4	2
7	6.4	3.2	4.5	1.5	2
8	6.9	3.1	4.9	1.5	2
9	5.5	2.3	4.0	1.3	2
10	6.5	2.8	4.6	1.5	2
11	6.3	3.3	6.0	2.5	3
12	5.8	2.7	5.1	1.9	3
13	7.1	3.0	5.9	2.1	3
14	6.3	2.9	5.6	1.8	3
15	6.5	3.0	5.8	2.2	3

Granule g(u_1) = {u_1}

There are no identical objects in this dataset, so we can decrease the granulation radius in order to catch more objects in one granule.

Granulation radius is being reduced to 3/4 (0.75). In order to better demonstrate the effect of selecting objects for granulation in this type of granulation, the entire dataset will be presented with the objects that qualify for this granulation marked with green background.

Table 3.2: Homogeneous granulation toy example - objects in granule for u_1 and radius of 3/4 (0.75).

	sepal_length	sepal_width	petal_length	petal_width	iris class
1	5.1	3.5	1.4	0.2	1
2	4.9	3.0	1.4	0.2	1
3	4.7	3.2	1.3	0.2	1
4	4.6	3.1	1.5	0.2	1
5	5.0	3.6	1.4	0.2	1
6	7.0	3.2	4.7	1.4	2
7	6.4	3.2	4.5	1.5	2
8	6.9	3.1	4.9	1.5	2
9	5.5	2.3	4.0	1.3	2
10	6.5	2.8	4.6	1.5	2
11	6.3	3.3	6.0	2.5	3
12	5.8	2.7	5.1	1.9	3
13	7.1	3.0	5.9	2.1	3
14	6.3	2.9	5.6	1.8	3
15	6.5	3.0	5.8	2.2	3

Granule g(u_1) = {u_1}

In the case of object u1, also for a radius of 3/4, there are no other objects indiscernible in this degree.

Granulation radius is being reduced to 2/4 (0.5).

Table 3.3: Homogeneous granulation toy example - objects in granule for u_1 and radius of 2/4 (0.5).

	sepal_length	sepal_width	petal_length	petal_width	iris class
1	5.1	3.5	1.4	0.2	1
2	4.9	3.0	1.4	0.2	1
3	4.7	3.2	1.3	0.2	1
4	4.6	3.1	1.5	0.2	1
5	5.0	3.6	1.4	0.2	1
6	7.0	3.2	4.7	1.4	2
7	6.4	3.2	4.5	1.5	2
8	6.9	3.1	4.9	1.5	2
9	5.5	2.3	4.0	1.3	2
10	6.5	2.8	4.6	1.5	2
11	6.3	3.3	6.0	2.5	3
12	5.8	2.7	5.1	1.9	3
13	7.1	3.0	5.9	2.1	3
14	6.3	2.9	5.6	1.8	3
15	6.5	3.0	5.8	2.2	3

Granule g(u_1) = {u_1, u_2, u_5}

This granule is still homogeneous, so we will **reduce the radius to 1/4 (0.25).**

Table 3.4: Homogeneous granulation toy example - objects in granule for u_1 and radius of 1/4 (0.25).

	sepal_length	sepal_width	petal_length	petal_width	iris class
1	5.1	3.5	1.4	0.2	1
2	4.9	3.0	1.4	0.2	1
3	4.7	3.2	1.3	0.2	1
4	4.6	3.1	1.5	0.2	1
5	5.0	3.6	1.4	0.2	1
6	7.0	3.2	4.7	1.4	2
7	6.4	3.2	4.5	1.5	2
8	6.9	3.1	4.9	1.5	2
9	5.5	2.3	4.0	1.3	2
10	6.5	2.8	4.6	1.5	2
11	6.3	3.3	6.0	2.5	3
12	5.8	2.7	5.1	1.9	3
13	7.1	3.0	5.9	2.1	3
14	6.3	2.9	5.6	1.8	3
15	6.5	3.0	5.8	2.2	3

Granule g(u_1) = {u_1, u_2, u_3, u_4, u_5}

Now our granule already contains all the objects for the decision class equal to 1, so intuitively, according to the definition of homogeneous granulation, we can assume that the algorithm should terminate its operation since only objects from other decision classes remain. However, from a formal point of view, we continue the algorithm until the granule contains an object that is not homogeneous in terms of the decision class, and then we take the radius of granulation as that of the previous iteration.

Granulation radius is being reduced to 0/4 (0.0).

Analyzing the example for concept dependent granulation, we know that radius 0 is a special case and means that all objects are indiscernible in this degree, and this means that all dataset objects will go into the granule. This in turn means that the granule will be of the form:

Granule g(u_1) = {u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8, u_9, u_{10}, u_{11}, u_{12}, u_{13}, u_{14}, u_{15}}

Table 3.5: Homogeneous granulation toy example - objects in granule for u_1 and radius of 0/4 (0.0).

	sepal_length	sepal_width	petal_length	petal_width	iris class
1	5.1	3.5	1.4	0.2	1
2	4.9	3.0	1.4	0.2	1
3	4.7	3.2	1.3	0.2	1
4	4.6	3.1	1.5	0.2	1
5	5.0	3.6	1.4	0.2	1
6	7.0	3.2	4.7	1.4	2
7	6.4	3.2	4.5	1.5	2
8	6.9	3.1	4.9	1.5	2
9	5.5	2.3	4.0	1.3	2
10	6.5	2.8	4.6	1.5	2
11	6.3	3.3	6.0	2.5	3
12	5.8	2.7	5.1	1.9	3
13	7.1	3.0	5.9	2.1	3
14	6.3	2.9	5.6	1.8	3
15	6.5	3.0	5.8	2.2	3

So it gets a non-uniform granule, and this means that the radius from the previous step is our final radius of granulation for this central object u_1.

Continuing the process of forming granules will yield the following set of granules.

$g_{0.25}(u_1) = (u_1, u_2, u_3, u_4, u_5)$

$g_{0.5}(u_2) = (u_1, u_2, u_5)$

$g_{0.5}(u_3) = (u_3)$

$g_{0.5}(u_4) = (u_4)$

$g_{0.25}(u_5) = (u_1, u_2, u_3, u_4, u_5)$

$g_{0.5}(u_6) = (u_6)$

$g_{0.5}(u_7) = (u_7)$

$g_{0.5}(u_8) = (u_8)$

$g_{0.25}(u_9) = (u_9)$

$g_{0.5}(u_{10}) = (u_{10})$

$g_{0.25}(u_{11}) = (u_{11}, u_{14})$

$g_{0.25}(u_{12}) = (u_{12})$

$g_{0.5}(u_{13}) = (u_{13})$

$g_{0.25}(u_{14}) = (u_{11}, u_{14})$

$g_{0.5}(u_{15}) = (u_{15})$

Step 2: covering original dataset.

As in the previous two cases, the hierarchical granule coverage method will also be used here for a more meaningful comparison of the final granulation effect.
The granules that make up the coverage of the original collection have been marked with underline.

$\underline{g_{0.25}(u_1) = (u_1, u_2, u_3, u_4, u_5)}$

$g_{0.5}(u_2) = (u_1, u_2, u_5)$

$g_{0.5}(u_3) = (u_3)$

$g_{0.5}(u_4) = (u_4)$

$g_{0.25}(u_5) = (u_1, u_2, u_3, u_4, u_5)$

$\underline{g_{0.5}(u_6) = (u_6)}$

$\underline{g_{0.5}(u_7) = (u_7)}$

$\underline{g_{0.5}(u_8) = (u_8)}$

$\underline{g_{0.25}(u_9) = (u_9)}$

$\underline{g_{0.5}(u_{10}) = (u_{10})}$

$\underline{g_{0.25}(u_{11}) = (u_{11}, u_{14})}$

$\underline{g_{0.25}(u_{12}) = (u_{12})}$

$\underline{g_{0.5}(u_{13}) = (u_{13})}$

$g_{0.25}(u_{14}) = (u_{11}, u_{14})$

$\underline{g_{0.5}(u_{15}) = (u_{15})}$

Step 3: creating reflection dataset from coverage.

Within each granule as before, a majority vote is conducted for the selection of the value of each attribute and ties are decided at random. This means that the solution is not deterministic and subsequent runs of granulation may yield slightly different objects in the coverage set.

Table 3.6: Reflection dataset for radius homogeneous granulation.

sepal_length	sepal_width	petal_length	petal_width	iris class
4.6	3.2	1.4	0.2	1
7.0	3.2	4.7	1.4	2
6.4	3.2	4.5	1.5	2
6.9	3.1	4.9	1.5	2
5.5	2.3	4.0	1.3	2
6.5	2.8	4.6	1.5	2
6.3	2.9	5.6	2.5	3
5.8	2.7	5.1	1.9	3
7.1	3.0	5.9	2.1	3
6.5	3.0	5.8	2.2	3

3.3. Experimental session

3.3.1. Methodology

The research carried out within the framework of this chapter is aimed at indicating the differences that occur in collections with original data and data granulated using the homogeneous method, as well as the differences in the data comparing other granulation techniques.

To indicate the differences, three areas were adopted within which these differences were studied. The first is the change in dataset size after granulation and its effect on the change in class balance. The next is to indicate how entropy changed in these datasets, and the last is the effect of granulation on classification measures for the original and granulated datasets. Common techniques were used to reduce the effects of randomness in the results obtained in the form of cross-validation and permutation tests.

3.3.2. Results

Impact of homogeneous granulation on new dataset sizes

Due to the use of a random granule selection mechanism in the process of covering the original harvest, 10 passes of homogeneous granulation were carried out through each dataset. The averaged results are presented in the table 3.7.

Table 3.7: Aggregated reflection sizes after 10 times homogeneous granulation process.

dataset	objects total			original size	% of size reduction
	mean	min	max		
adult	23513.00	23434	23652	45222	48.01
australian	350.80	340	368	690	49.16
australian dummy	355.00	335	372	689	48.48
breast	445.80	436	462	569	21.65
heart	164.60	157	170	303	45.68
iris	79.70	74	84	150	46.87
mushroom num. encoded	53.90	42	62	5643	99.04
pima	624.20	615	637	768	18.72
red wine	1161.60	1150	1171	1599	27.35
white wine	3509.30	3492	3526	4898	28.35
wine merged	4661.60	4643	4685	6497	28.25

As can be seen in the table 3.7 the size of the collections was reduced in each case, starting at 18.72% reduction for the pima indian diabetes set, through a reduction of nearly 50% for the adult, australian, heart and iris sets, until a reduction of about 99% for the mushroom set, which, as can be seen, is characterized by a lower diversity of features among the individual observations.

Presented below are box plots for each dataset and 10 results of homogeneous granulation in the context of the size of the reflection set. Four of them, i.e. mushroom, wine merged, adult and breast, have visible outliers, which can testify to quite a high diversity of objects. This works well for the latter three sets, while the outliers in the mushroom set are due to the very small size of the granuled set and hence the large spread in the data.

As a result of homogeneous granulation, the balance of classes in each dataset has also changed and is as follows.

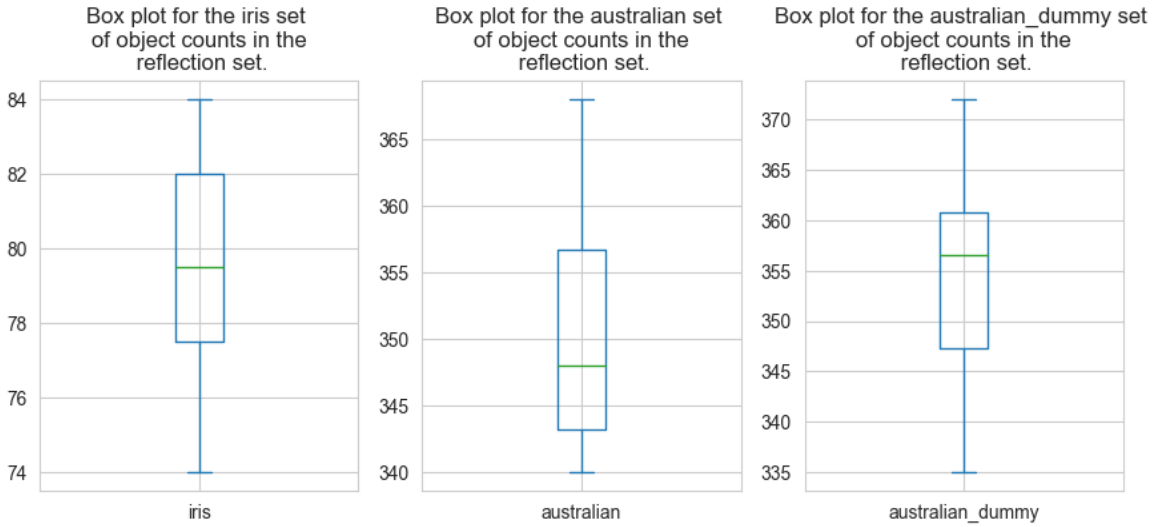


Figure 3.1: Box plots presenting a distribution of reflection set sizes for the iris, australian and australian_dummy datasets.

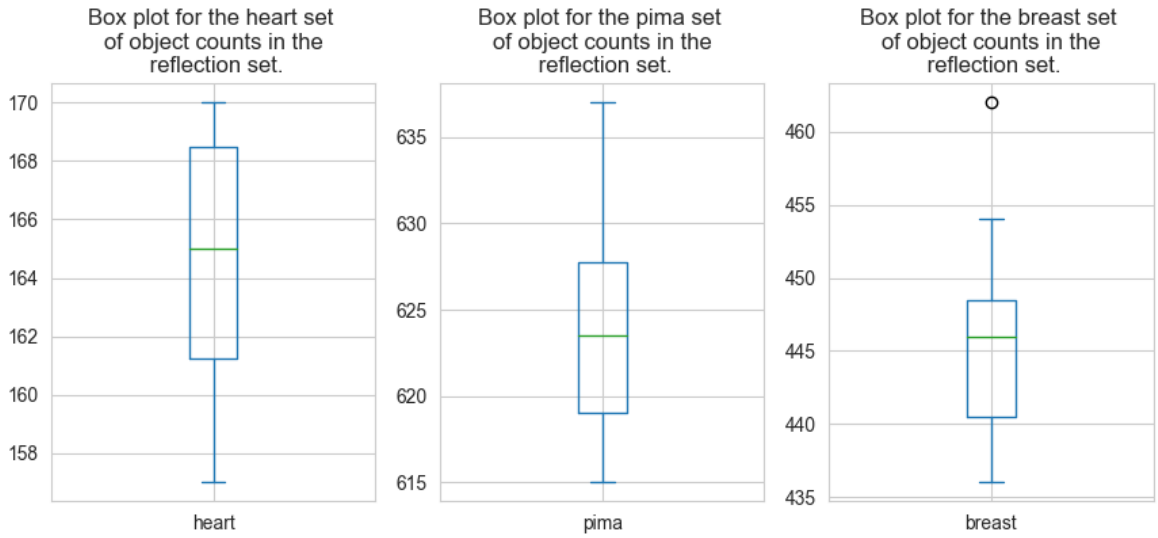


Figure 3.2: Box plots presenting a distribution of reflection set sizes for the heart desease, pima diabetes and breast cancer datasets.

Table 3.8: Class balance after 10 times homogeneous granulation.

dataset	**cover**	**class balance (mean)**
adult	random	1: 9076, 0: 14437
australian	random	1.0: 170, 0.0: 181
australian dummy	random	0.0: 190, 1.0: 165
breast	random	0: 252, 1: 194
heart	random	1.0: 86, 0.0: 79
iris	random	2.0: 30, 1.0: 16, 3.0: 33
mushroom	random	1: 22, 0: 32
pima	random	0.0: 379, 1.0: 245
red wine	random	5.0: 465, 6.0: 462, 3.0: 10, 7.0: 154, 8.0: 17, 4.0: 53
white wine	random	7.0: 635, 5.0: 1037, 4.0: 150, 6.0: 1533, 8.0: 130, 3.0: 20, 9.0: 5
wine merged	random	7.0: 788, 6.0: 1996, 5.0: 1502, 4.0: 203, 8.0: 146, 3.0: 30, 9.0: 5

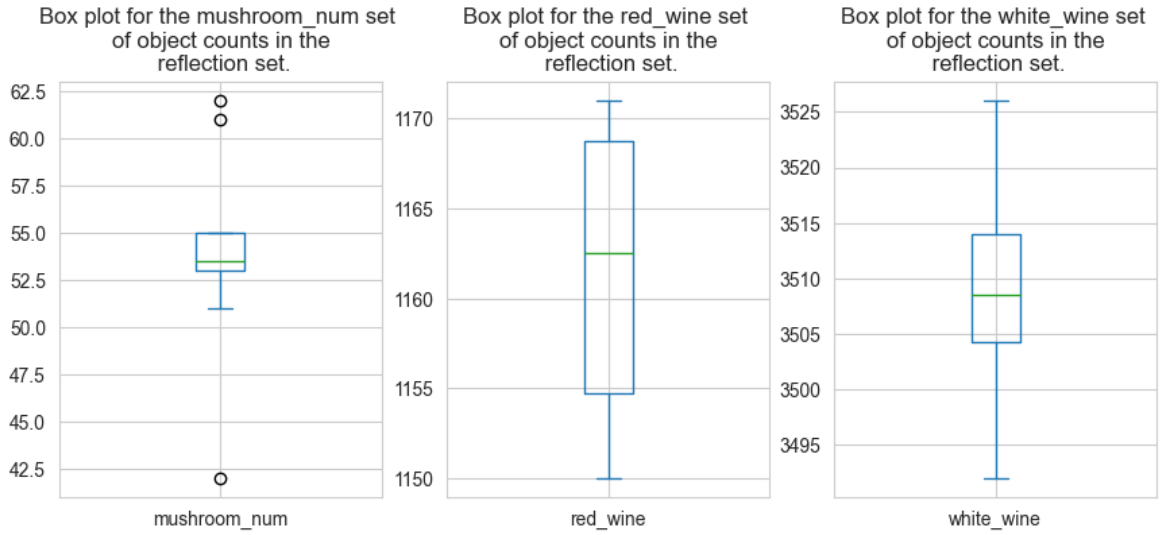


Figure 3.3: Box plots presenting a distribution of reflection set sizes for the mushroom numeric encoded, red wine and white wine datasets.

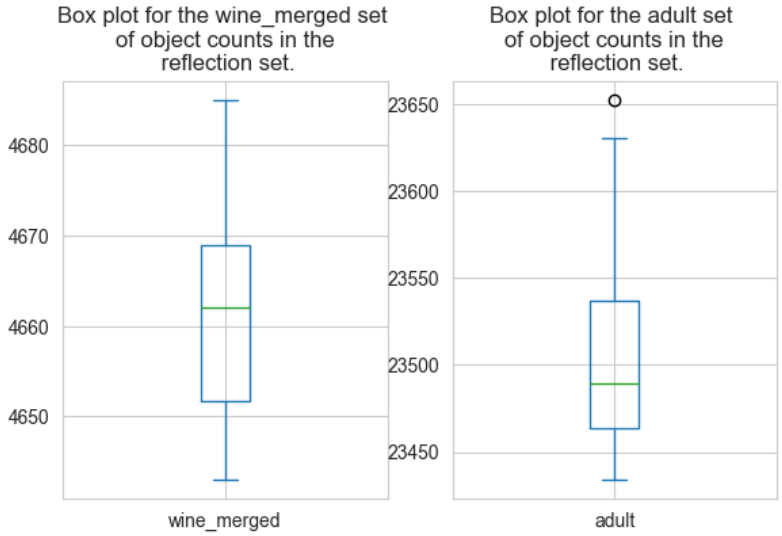


Figure 3.4: Box plots presenting a distribution of reflection set sizes for the wine merged and adult datasets.

Comparing this table with the original balance of classes presented in the 2.4.2, we can see:

- adult dataset: the original balance, which was about 3:1 (34014:11208 for 0:1 labels), has changed to about 5:3 (14437:9076), so the imbalance has decreased,
- australian dataset: from the original balance about 4:3 (383:307 for 0:1 labels), there was a change close to 1:1 (17:18),

- australian dummy dataset: originally about 4:3 (382:307 for 0:1 labels), and after granulation the imbalance slightly decreased to 190:165,
- breast dataset: originally about 7:4 (357:212, for 0:1 labels), and after granulation the imbalance decreased to about 5:4 (252:194),
- heart dataset: originally around 14:16 (138:165 for 0:1 labels), after granulation almost perfectly balanced 79:86,
- iris dataset: for labels 1,2,3 the balance was perfect in the original dataset and was 50:50:50, and after granulation the balance was disturbed to 16:30:33 respectively,
- red wine dataset: the original dataset is strongly imbalanced, with the dominance of two classes (5 and 6), and the original balance for labels 3,4,5,6,7,8 is 10:53:681:638:177:18, and after granulation is 10:53:465:462:154:17 respectively, and has improved, although the collection is still strongly imbalanced,
- white wine dataset: from the original balance for labels 3:4:5:6:7:8:9 of 20:163:1457:2198:880:175:5 was obtained, respectively, 20:150:1037:1533:635:130:5, that is, the balance also improved,
- wine merged dataset: from the merged collection of both types of wines from the original balance for classes 3,4,5,6,7,8,9 of 30:216:2137:2836:1079:193:5 was obtained respectively 30:203:1502:1996:788:146:5, which also improved balance, but the abundance of the two extreme classes is originally so low that the collection is still strongly imbalanced.

Entropy

Entropy is a measure derived from information theory and indicates the uncertainty that exists in the data. It is closely related to the distribution of values in a given sample and is interpreted somewhat differently for binomial and polynomial distributions. Here the measure was used for multiclass data. The base of the logarithm that was adopted in the research conducted is 2. That is, the entropy score is expressed in bits (also called 'shannons') of information, according to the formula presented by Claud Shannon in [40]:

$$H = -\sum_{x=1}^{n} P(x)\, log_r\, P(x) \tag{3.4}$$

where n is space of possible values of x, $P(x)$ is a probability of getting x, and r is a logarithm base.

Below is a detailed table with the calculated entropy for the original datasets and the homogenized datasets.

Table 3.9: Entropy for original dataset and homogeneously granuled datasets for every feature.

dataset	feature	entropy	homgran entropy	entropy diff
iris	sepal_length	4.82	4.52	-0.30
iris	sepal_width	4.01	3.78	-0.23
iris	petal_length	5.03	4.82	-0.21
iris	petal_width	4.07	4.10	0.03
australian	0	0.91	0.90	-0.00
australian	1	8.18	7.54	-0.65
australian	2	7.07	6.54	-0.52
australian	3	0.82	0.83	0.01
australian	4	3.50	3.44	-0.06
australian	5	1.78	1.72	-0.07
australian	6	5.90	5.54	-0.35
australian	7	1.00	0.98	-0.02
australian	8	0.98	0.97	-0.01
australian	9	2.53	2.34	-0.19
australian	10	0.99	1.00	0.00
australian	11	0.50	0.60	0.10
australian	12	5.74	5.20	-0.54
australian	13	5.19	3.68	-1.50
australian dummy	a1	0.91	0.90	-0.00
australian dummy	a2	8.19	7.57	-0.62
australian dummy	a3	7.06	6.69	-0.37
australian dummy	a7	5.90	5.58	-0.32
australian dummy	a8	1.00	0.99	-0.00

Continued on next page

dataset	feature	entropy	homgran entropy	entropy diff
australian dummy	a9	0.99	0.98	-0.01
australian dummy	a10	2.53	2.37	-0.16
australian dummy	a11	0.99	1.00	0.00
australian dummy	a13	5.74	5.17	-0.57
australian dummy	a14	5.18	3.77	-1.41
australian dummy	a4_1	0.79	0.79	0.00
australian dummy	a4_2	0.79	0.80	0.01
australian dummy	a4_3	0.03	0.05	0.02
australian dummy	a5_1	0.39	0.27	-0.12
australian dummy	a5_10	0.22	0.21	-0.01
australian dummy	a5_11	0.51	0.53	0.02
australian dummy	a5_12	0.04	0.03	-0.01
australian dummy	a5_13	0.33	0.30	-0.03
australian dummy	a5_14	0.31	0.27	-0.04
australian dummy	a5_2	0.26	0.21	-0.05
australian dummy	a5_3	0.42	0.44	0.02
australian dummy	a5_4	0.38	0.40	0.02
australian dummy	a5_5	0.11	0.13	0.02
australian dummy	a5_6	0.40	0.32	-0.07
australian dummy	a5_7	0.31	0.28	-0.03
australian dummy	a5_8	0.75	0.76	0.02
australian dummy	a5_9	0.45	0.49	0.04
australian dummy	a6_1	0.41	0.28	-0.13
australian dummy	a6_2	0.07	0.08	0.01
australian dummy	a6_3	0.09	0.13	0.04
australian dummy	a6_4	0.98	0.97	-0.00
australian dummy	a6_5	0.42	0.45	0.03
australian dummy	a6_7	0.07	0.05	-0.02
australian dummy	a6_8	0.72	0.73	0.01
australian dummy	a6_9	0.09	0.10	0.01

Continued on next page

dataset	feature	entropy	homgran entropy	entropy diff
australian dummy	a12_1	0.41	0.46	0.05
australian dummy	a12_2	0.45	0.52	0.07
australian dummy	a12_3	0.09	0.14	0.05
heart	age	5.08	4.91	-0.16
heart	sex	0.90	0.87	-0.03
heart	cp	1.74	1.80	0.06
heart	trestbps	4.70	4.30	-0.40
heart	chol	7.04	6.48	-0.56
heart	fbs	0.61	0.54	-0.07
heart	restecg	1.09	1.14	0.05
heart	thalach	6.17	5.79	-0.37
heart	exang	0.91	0.89	-0.02
heart	oldpeak	4.13	3.77	-0.36
heart	slope	1.29	1.27	-0.02
heart	ca	1.67	1.59	-0.08
heart	thal	1.30	1.28	-0.02
pima	0	3.48	3.55	0.06
pima	1	6.75	6.72	-0.03
pima	2	4.79	4.73	-0.06
pima	3	4.59	4.55	-0.03
pima	4	4.68	4.49	-0.19
pima	5	7.59	7.49	-0.11
pima	6	8.83	8.58	-0.25
pima	7	5.03	5.12	0.09
breast	0	8.73	8.46	-0.27
breast	1	8.82	8.50	-0.33
breast	2	8.98	8.58	-0.40
breast	3	9.05	8.66	-0.39
breast	4	8.78	8.40	-0.38
breast	5	9.04	8.63	-0.40

Continued on next page

dataset	feature	entropy	homgran entropy	entropy diff
breast	6	9.00	8.65	-0.35
breast	7	9.01	8.66	-0.35
breast	8	8.62	8.28	-0.34
breast	9	8.90	8.55	-0.35
breast	10	9.05	8.66	-0.39
breast	11	8.97	8.59	-0.38
breast	12	9.02	8.64	-0.38
breast	13	9.00	8.63	-0.38
breast	14	9.07	8.69	-0.38
breast	15	9.05	8.67	-0.38
breast	16	8.98	8.63	-0.36
breast	17	8.89	8.55	-0.34
breast	18	8.89	8.53	-0.36
breast	19	9.07	8.67	-0.40
breast	20	8.72	8.45	-0.27
breast	21	8.95	8.56	-0.39
breast	22	8.95	8.57	-0.38
breast	23	9.06	8.66	-0.41
breast	24	8.54	8.25	-0.29
breast	25	9.01	8.63	-0.38
breast	26	9.00	8.66	-0.35
breast	27	8.83	8.52	-0.31
breast	28	8.90	8.55	-0.36
breast	29	9.03	8.65	-0.39
mushroom num	1	1.34	1.61	0.26
mushroom num	2	1.55	1.62	0.07
mushroom num	3	2.47	2.58	0.10
mushroom num	4	0.99	0.88	-0.11
mushroom num	5	1.97	2.20	0.23
mushroom num	6	0.03	0.07	0.04

Continued on next page

dataset	feature	entropy	homgran entropy	entropy diff
mushroom num	7	0.68	0.75	0.07
mushroom num	8	0.54	0.91	0.37
mushroom num	9	2.75	2.58	-0.17
mushroom num	10	1.00	0.95	-0.05
mushroom num	11	1.35	1.57	0.23
mushroom num	12	1.25	0.74	-0.51
mushroom num	13	1.44	1.07	-0.37
mushroom num	14	1.88	1.15	-0.74
mushroom num	15	1.91	1.13	-0.78
mushroom num	16	0.00	0.00	0.00
mushroom num	17	0.02	0.13	0.12
mushroom num	18	0.20	0.67	0.47
mushroom num	19	1.37	0.90	-0.47
mushroom num	20	1.85	2.13	0.28
mushroom num	21	2.04	2.09	0.05
mushroom num	22	1.93	2.19	0.26
red wine	fixed acidity	5.94	5.97	0.03
red wine	volatile acidity	6.39	6.38	-0.01
red wine	citric acid	5.87	5.87	-0.01
red wine	residual sugar	4.78	4.77	-0.01
red wine	chlorides	6.22	6.26	0.04
red wine	free sulfur dioxide	5.08	5.04	-0.04
red wine	total sulfur dioxide	6.60	6.54	-0.07
red wine	density	7.96	7.92	-0.04
red wine	pH	5.91	5.91	0.01
red wine	sulphates	5.73	5.71	-0.02
red wine	alcohol	5.19	5.24	0.05
white wine	fixed acidity	5.06	5.10	0.03
white wine	volatile acidity	5.34	5.37	0.03
white wine	citric acid	5.35	5.35	0.00

Continued on next page

dataset	feature	entropy	homgran entropy	entropy diff
white wine	residual sugar	7.19	7.07	-0.12
white wine	chlorides	5.63	5.64	0.01
white wine	free sulfur dioxide	6.06	6.05	-0.01
white wine	total sulfur dioxide	7.39	7.39	0.00
white wine	density	8.85	8.78	-0.07
white wine	pH	5.91	5.92	0.02
white wine	sulphates	5.42	5.44	0.02
white wine	alcohol	5.56	5.58	0.02
wine merged	fixed acidity	5.47	5.51	0.03
wine merged	volatile acidity	5.96	5.98	0.02
wine merged	citric acid	5.64	5.65	0.01
wine merged	residual sugar	6.92	6.82	-0.11
wine merged	chlorides	6.31	6.32	0.01
wine merged	free sulfur dioxide	6.06	6.03	-0.02
wine merged	total sulfur dioxide	7.72	7.71	-0.01
wine merged	density	8.95	8.91	-0.04
wine merged	pH	6.01	6.01	0.00
wine merged	sulphates	5.72	5.73	0.01
wine merged	alcohol	5.51	5.53	0.02
wine merged	wine color	0.81	0.81	0.00
adult	age	5.65	5.59	-0.06
adult	workclass	1.42	1.58	0.16
adult	fnlwgt	14.40	13.58	-0.82
adult	education	2.92	2.85	-0.06
adult	education-num	2.92	2.85	-0.06
adult	marital-status	1.82	1.45	-0.37
adult	occupation	3.40	3.31	-0.09
adult	relationship	2.14	1.72	-0.42
adult	race	0.77	0.65	-0.13
adult	sex	0.91	0.76	-0.15

Continued on next page

dataset	feature	entropy	homgran entropy	entropy diff
adult	capital-gain	0.87	0.98	0.12
adult	capital-loss	0.52	0.57	0.05
adult	hours-per-week	3.44	3.17	-0.27
adult	native-country	0.82	0.70	-0.13

The decrease in entropy for individual features of the given sets after granulation seems natural because of the way the mechanism of creating a new reflective set through majority voting is able to remove noise from the data in the form of feature values that occur infrequently in similar objects. The increase in entropy after the granulation process is the result of the same mechanism, but where the most frequent feature values in the voted granule are equidistant and randomly selected, which can generate entirely new observations in the set, or affect the change in the distribution for a given feature so that uncertainty increases.
Below is a table with the summed nominal and percentage difference for each dataset.

Table 3.10: Entropy change after homogeneous granulation - summary for each dataset.

dataset	**entropy**	**homgran. entropy**	**entropy change**	**percent. change**
adult	41.99	39.76	-2.23	-5.32 %
australian	45.08	41.27	-3.80	-8.44 %
australian dummy	48.76	45.23	-3.53	-7.23 %
breast	267.93	257.10	-10.83	-4.04 %
heart	36.62	34.64	-1.98	-5.40 %
iris	17.93	17.22	-0.71	-3.97 %
mushroom num	28.57	27.93	-0.64	-2.25 %
pima	45.74	45.22	-0.52	-1.14 %
red wine	65.66	65.59	-0.07	-0.10 %
white wine	67.75	67.68	-0.07	-0.10 %
wine merged	71.08	71.00	-0.08	-0.11 %

The $entropy$ column represents the summed entropy for a given original dataset, $homgran.\ entropy$ is the summed entropy for a set granulated by the homogeneous method, $entropy\ change$ is the nominal change as $homgran.\ entropy\ -\ entropy$ and $percent.\ change$ expresses this change in percentage points.

It can be seen that for each set there was a reduction in entropy, although these changes are small, and in the case of 7 datasets even very small, below the threshold of statistical significance.

The classification results will give us an additional answer to the question "How does homogeneous granulation affect the preservation of intrinsic knowledge of individual data sets?", also in the context of comparing it with the standard and concept dependent granulation techniques presented earlier.

Table with classification results after homogeneous granulation is presented below.

Table 3.11: Classification results for homogeneous granulation.

dataset	classifier	acc	bal. acc	prec. macro	recall macro	f1 macro
adult	decision_tree	0.93	0.94	0.89	0.94	0.91
adult	knn	0.75	0.70	0.68	0.70	0.68
adult	log_regression	0.78	0.63	0.71	0.63	0.64
adult	mlp	0.75	0.50	0.81	0.50	0.64
adult	naive_bayes	0.79	0.63	0.73	0.63	0.65
adult	random_forest	0.84	0.72	0.84	0.72	0.75
adult	svm	0.93	0.87	0.96	0.87	0.90
adult	xgboost	0.88	0.84	0.85	0.84	0.84
australian	decision_tree	0.89	0.89	0.89	0.89	0.89
australian	knn	0.71	0.71	0.71	0.71	0.71
australian	log_regression	0.86	0.85	0.85	0.85	0.85
australian	mlp	0.52	0.50	0.57	0.50	0.62
australian	naive_bayes	0.77	0.75	0.79	0.75	0.75
australian	random_forest	0.90	0.90	0.90	0.90	0.90
australian	svm	0.68	0.68	0.76	0.68	0.66
australian	xgboost	0.93	0.94	0.93	0.94	0.93
australian_dummy	decision_tree	0.88	0.87	0.87	0.87	0.87
australian_dummy	knn	0.71	0.71	0.71	0.71	0.71
australian_dummy	log_regression	0.85	0.86	0.85	0.86	0.85
australian_dummy	mlp	0.56	0.50	0.56	0.50	0.71

Continued on next page

dataset	classifier	acc	bal. acc	prec. macro	recall macro	f1 macro
australian_dummy	naive_bayes	0.77	0.76	0.79	0.76	0.76
australian_dummy	random_forest	0.90	0.89	0.90	0.89	0.89
australian_dummy	svm	0.70	0.70	0.74	0.70	0.68
australian_dummy	xgboost	0.94	0.94	0.94	0.94	0.94
breast	decision_tree	0.98	0.98	0.97	0.98	0.98
breast	knn	0.93	0.93	0.93	0.93	0.93
breast	log_regression	0.96	0.95	0.96	0.95	0.95
breast	mlp	0.63	0.50	0.63	0.50	0.77
breast	naive_bayes	0.94	0.93	0.95	0.93	0.94
breast	random_forest	0.99	0.98	0.99	0.98	0.98
breast	svm	0.95	0.94	0.97	0.94	0.95
breast	xgboost	0.99	0.99	0.99	0.99	0.99
heart	decision_tree	0.85	0.86	0.86	0.86	0.85
heart	knn	0.70	0.71	0.71	0.71	0.70
heart	log_regression	0.85	0.85	0.85	0.85	0.85
heart	mlp	0.52	0.50	0.52	0.50	0.68
heart	naive_bayes	0.84	0.83	0.84	0.83	0.83
heart	random_forest	0.91	0.91	0.92	0.91	0.91
heart	svm	0.73	0.70	0.83	0.70	0.69
heart	xgboost	0.91	0.92	0.91	0.92	0.91
iris	decision_tree	0.99	0.99	0.99	0.99	0.99
iris	knn	0.97	0.97	0.97	0.97	0.97
iris	log_regression	0.97	0.97	0.97	0.97	0.97
iris	mlp	0.84	0.84	0.88	0.84	0.86
iris	naive_bayes	0.95	0.95	0.95	0.95	0.95
iris	random_forest	0.99	0.99	0.99	0.99	0.99
iris	svm	0.98	0.98	0.99	0.98	0.98
iris	xgboost	0.99	0.99	0.99	0.99	0.99
mushroom	decision_tree	0.91	0.91	0.91	0.91	0.91
mushroom	knn	0.82	0.78	0.83	0.78	0.78

Continued on next page

dataset	classifier	acc	bal. acc	prec. macro	recall macro	f1 macro
mushroom	log_regression	0.77	0.75	0.75	0.75	0.75
mushroom	mlp	0.69	0.63	0.71	0.63	0.69
mushroom	naive_bayes	0.71	0.64	0.78	0.64	0.63
mushroom	random_forest	0.97	0.96	0.97	0.96	0.96
mushroom	svm	0.90	0.88	0.92	0.88	0.89
mushroom	xgboost	0.95	0.95	0.95	0.95	0.95
pima	decision_tree	0.93	0.94	0.92	0.94	0.93
pima	knn	0.79	0.76	0.77	0.76	0.77
pima	log_regression	0.77	0.72	0.75	0.72	0.73
pima	mlp	0.65	0.50	0.67	0.50	0.75
pima	naive_bayes	0.77	0.73	0.75	0.73	0.74
pima	random_forest	0.81	0.77	0.81	0.77	0.78
pima	svm	0.93	0.90	0.95	0.90	0.92
pima	xgboost	0.97	0.97	0.96	0.97	0.96
red_wine	decision_tree	0.92	0.96	0.90	0.96	0.92
red_wine	knn	0.59	0.32	0.50	0.32	0.47
red_wine	log_regression	0.59	0.27	0.56	0.27	0.53
red_wine	mlp	0.41	0.17	0.42	0.17	0.55
red_wine	naive_bayes	0.57	0.37	0.36	0.37	0.43
red_wine	random_forest	0.64	0.28	0.72	0.28	0.57
red_wine	svm	0.70	0.35	0.81	0.35	0.56
red_wine	xgboost	0.94	0.96	0.94	0.96	0.94
white_wine	decision_tree	0.94	0.97	0.92	0.97	0.95
white_wine	knn	0.57	0.31	0.49	0.31	0.41
white_wine	log_regression	0.50	0.21	0.41	0.21	0.43
white_wine	mlp	0.45	0.14	0.45	0.14	0.62
white_wine	naive_bayes	0.45	0.31	0.35	0.31	0.33
white_wine	random_forest	0.57	0.23	0.67	0.23	0.42
white_wine	svm	0.75	0.43	0.87	0.43	0.59
white_wine	xgboost	0.76	0.69	0.74	0.69	0.71

Continued on next page

dataset	classifier	acc	bal. acc	prec. macro	recall macro	f1 macro
wine_merged	decision_tree	0.93	0.97	0.92	0.97	0.94
wine_merged	knn	0.58	0.31	0.59	0.31	0.40
wine_merged	log_regression	0.52	0.20	0.41	0.20	0.39
wine_merged	mlp	0.44	0.14	0.42	0.14	0.58
wine_merged	naive_bayes	0.40	0.37	0.25	0.37	0.24
wine_merged	random_forest	0.56	0.21	0.61	0.21	0.47
wine_merged	svm	0.71	0.37	0.85	0.37	0.51
wine_merged	xgboost	0.70	0.65	0.67	0.65	0.63

Comparison of balanced accuracy results for nil-case and homogeneous granulation is presented in the table3.12.

Table 3.12: Balanced accuracy comparison between nil-case classification and homogeneous granulation classification.

dataset	classifier	nil balanced_accuracy	homgran balanced_accuracy
adult	decision_tree	0.75	0.94
adult	knn	0.62	0.70
adult	logistic_regression	0.63	0.63
adult	mlp	0.50	0.50
adult	naive_bayes	0.63	0.63
adult	random_forest	0.72	0.72
adult	svm	0.50	0.87
adult	xgboost	0.79	0.84
australian	decision_tree	0.81	0.89
australian	knn	0.67	0.71
australian	logistic_regression	0.86	0.85
australian	mlp	0.65	0.50
australian	naive_bayes	0.78	0.75
australian	random_forest	0.86	0.90

Continued on next page

dataset	classifier	nil balanced_accuracy	homgran balanced_accuracy
australian	svm	0.50	0.68
australian	xgboost	0.85	0.94
australian_dummy	decision_tree	0.81	0.87
australian_dummy	knn	0.65	0.71
australian_dummy	logistic_regression	0.86	0.86
australian_dummy	mlp	0.68	0.50
australian_dummy	naive_bayes	0.81	0.76
australian_dummy	random_forest	0.86	0.89
australian_dummy	svm	0.52	0.70
australian_dummy	xgboost	0.86	0.94
breast	decision_tree	0.92	0.98
breast	knn	0.92	0.93
breast	logistic_regression	0.95	0.95
breast	mlp	0.58	0.50
breast	naive_bayes	0.93	0.93
breast	random_forest	0.95	0.98
breast	svm	0.50	0.94
breast	xgboost	0.95	0.99
heart	decision_tree	0.74	0.86
heart	knn	0.63	0.71
heart	logistic_regression	0.83	0.85
heart	mlp	0.64	0.50
heart	naive_bayes	0.81	0.83
heart	random_forest	0.83	0.91
heart	svm	0.50	0.70
heart	xgboost	0.78	0.92
iris	decision_tree	0.96	0.99
iris	knn	0.96	0.97
iris	logistic_regression	0.97	0.97
iris	mlp	0.97	0.84

Continued on next page

dataset	classifier	nil balanced_accuracy	homgran balanced_accuracy
iris	naive_bayes	0.96	0.95
iris	random_forest	0.96	0.99
iris	svm	0.97	0.98
iris	xgboost	0.96	0.99
mushroom_num	decision_tree	1.00	0.91
mushroom_num	knn	1.00	0.78
mushroom_num	logistic_regression	0.96	0.75
mushroom_num	mlp	1.00	0.63
mushroom_num	naive_bayes	0.62	0.64
mushroom_num	random_forest	0.98	0.96
mushroom_num	svm	1.00	0.88
mushroom_num	xgboost	1.00	0.95
pima	decision_tree	0.66	0.94
pima	knn	0.68	0.76
pima	logistic_regression	0.72	0.72
pima	mlp	0.64	0.50
pima	naive_bayes	0.71	0.73
pima	random_forest	0.71	0.77
pima	svm	0.50	0.90
pima	xgboost	0.69	0.97
red_wine	decision_tree	0.33	0.96
red_wine	knn	0.23	0.32
red_wine	logistic_regression	0.27	0.27
red_wine	mlp	0.22	0.17
red_wine	naive_bayes	0.31	0.37
red_wine	random_forest	0.26	0.28
red_wine	svm	0.25	0.35
red_wine	xgboost	0.30	0.96
white_wine	decision_tree	0.33	0.97
white_wine	knn	0.23	0.31

Continued on next page

dataset	classifier	nil balanced_accuracy	homgran balanced_accuracy
white_wine	logistic_regression	0.20	0.21
white_wine	mlp	0.18	0.14
white_wine	naive_bayes	0.29	0.31
white_wine	random_forest	0.21	0.23
white_wine	svm	0.24	0.43
white_wine	xgboost	0.29	0.69
wine_merged	decision_tree	0.33	0.97
wine_merged	knn	0.21	0.31
wine_merged	logistic_regression	0.19	0.20
wine_merged	mlp	0.19	0.14
wine_merged	naive_bayes	0.26	0.37
wine_merged	random_forest	0.21	0.21
wine_merged	svm	0.24	0.37
wine_merged	xgboost	0.28	0.65

The above table is presented below in a series of charts.

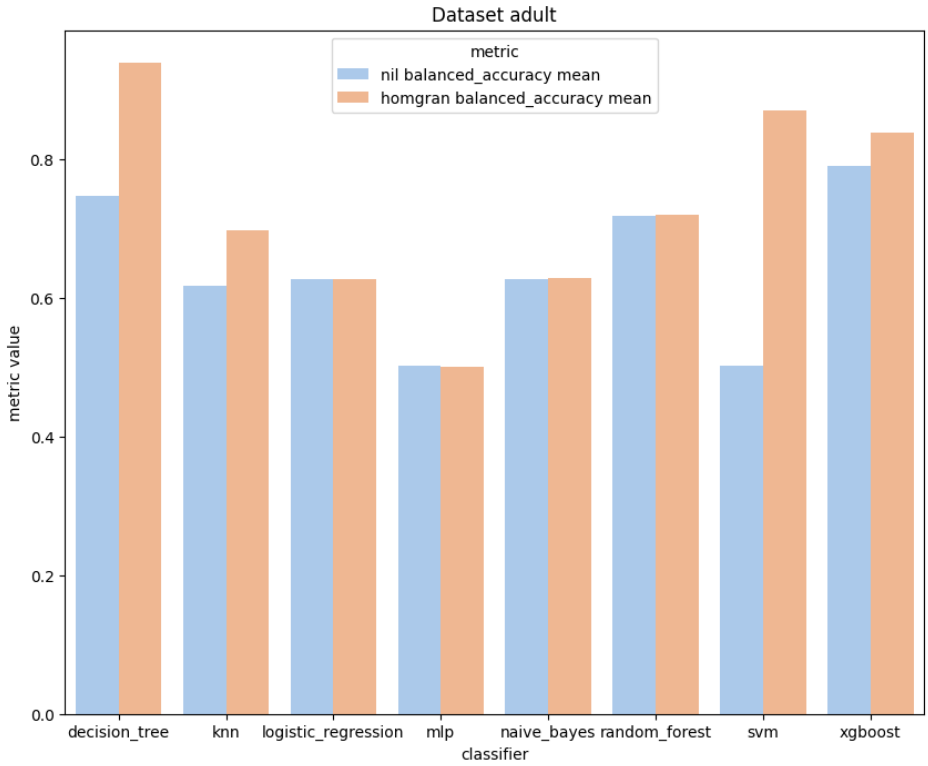


Figure 3.5: Bar plot nil-case vs homogeneous granulation balanced accuracy for adult dataset.

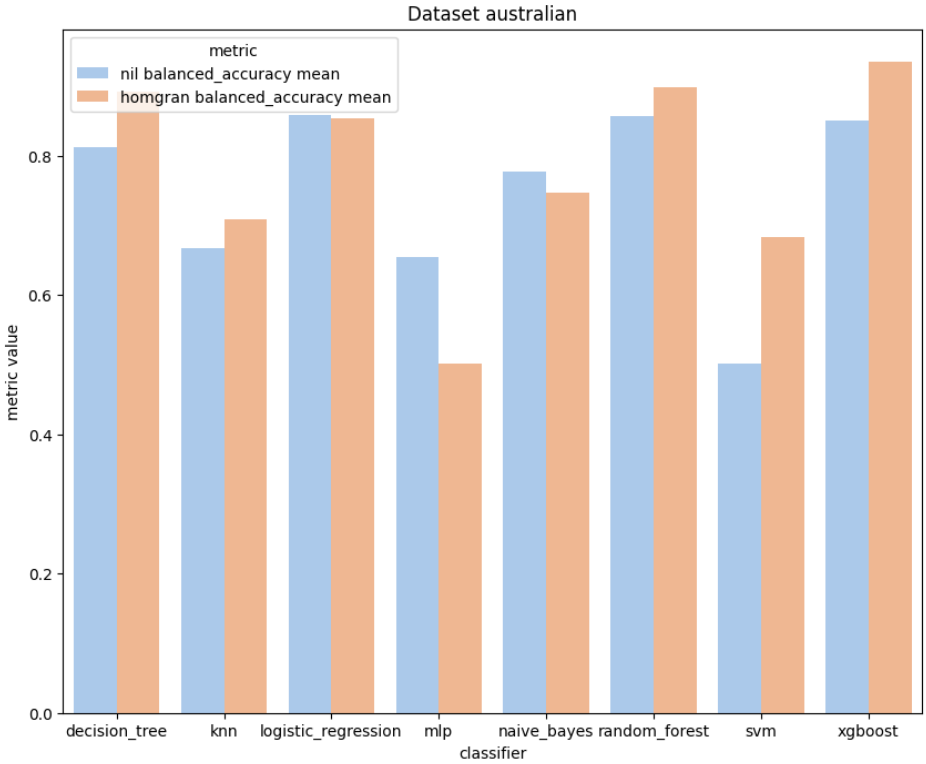


Figure 3.6: Bar plot nil-case vs homogeneous granulation balanced accuracy for australian dataset.

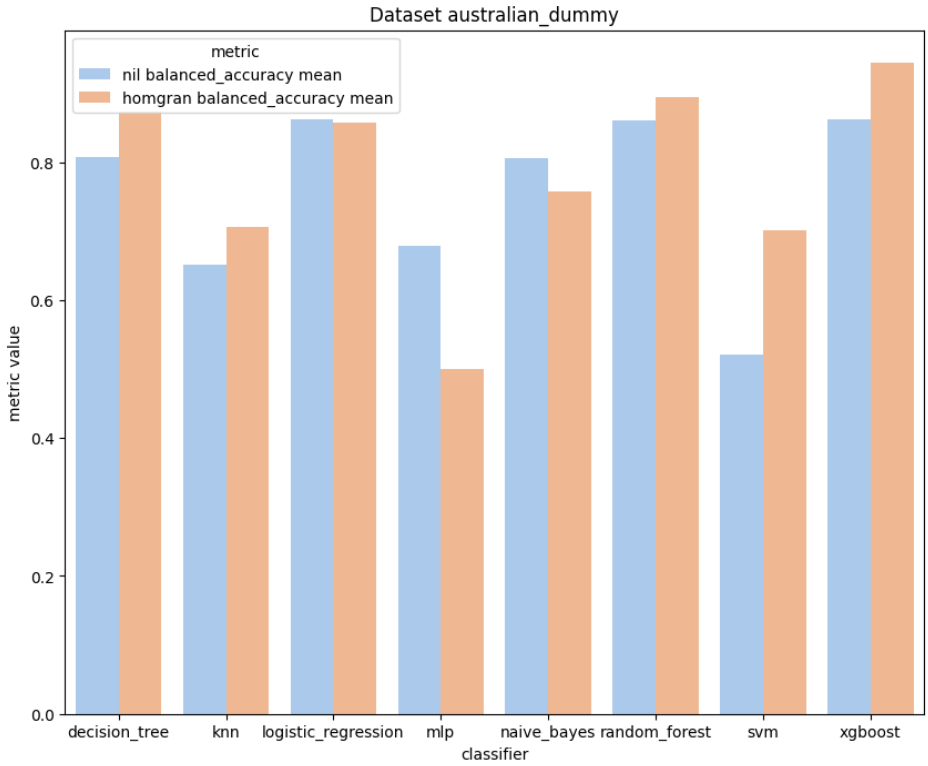


Figure 3.7: Bar plot nil-case vs homogeneous granulation balanced accuracy for australian dummy dataset.

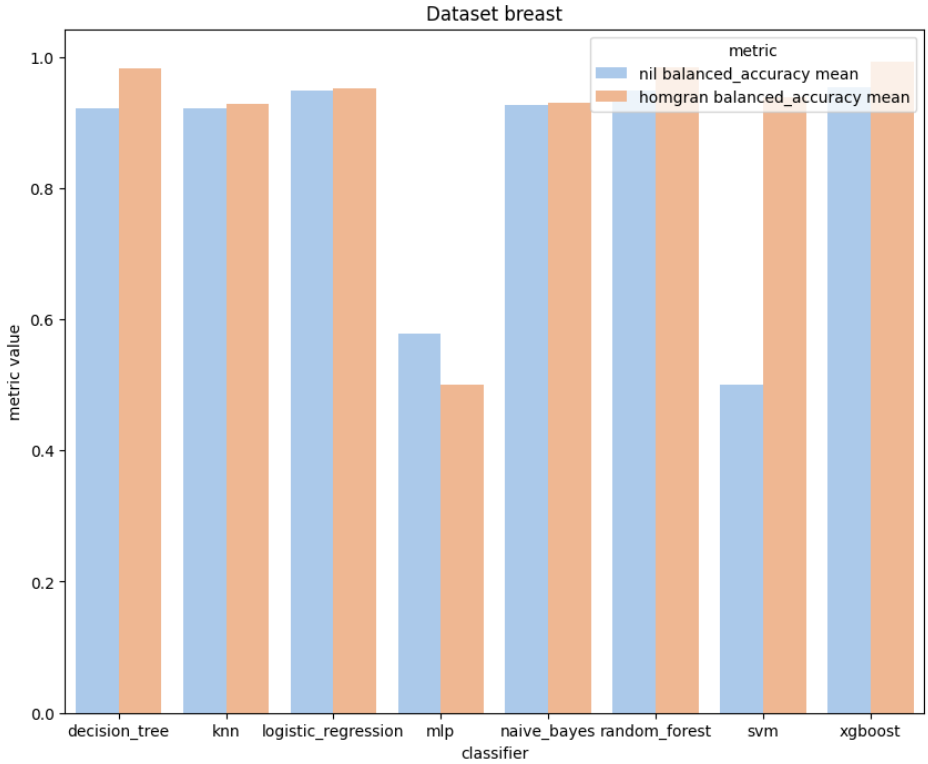


Figure 3.8: Bar plot nil-case vs homogeneous granulation balanced accuracy for breast dataset.

A comparison of the nil-case classification results for the experiments performed gives good results. It can be seen in many cases that the balanced accuracy increases after granulation of the data, but there are also cases where it is lower than with the original data, but they are few.

Looking at the results at the level of individual classifiers, we can see a rule of thumb, such that the results for decision tree, xgboost and svm improve for granulated data. The only exception to it is the mushroom num set, whose size after granulation is small, which can affect the selection of samples for train and test sets even with many permutations and it can be more imbalanced than the original data. The balanced accuracy metric for nil-case is equal to 1.0, so it is difficult to improve the results here. This may mean that homogeneous granulation improves the linear separability of observations of a given set. A special case of much better results for the aforementioned algorithms are wine-related data sets, i.e. red wine, white wine and the merged set (enriched with an additional column with wine color) - wine merged. Here the improvement in balanced accuracy is 2-3 times. A detailed analysis of the classification of individual classes and

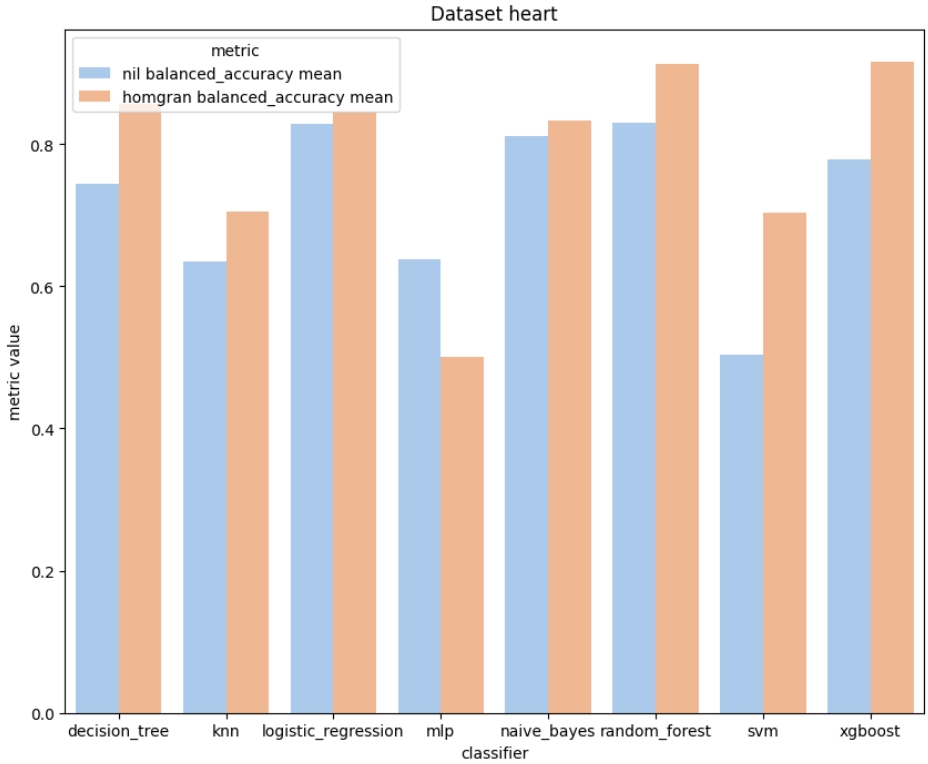


Figure 3.9: Bar plot nil-case vs homogeneous granulation balanced accuracy for heart dataset.

observations could give a definitive answer about the reasons for such improvement. Another classifier whose results are the same or slightly better than for nil-case is random forest and knn. The results for the naive-bayes, logistic regression classifiers oscillate around the results for the nil-case.
Noteworthy are the outlier results of the mlp classifier, which are at best comparable to those of nil-case, but usually significantly worse than them. It should be noted, however, that for nil-case of the mlp results are the lowest achieved by the classifiers tested, which may indicate an inappropriate choice of hyperparameters. To improve the comparability of the results no preprocessing such as normalization/standardization of the data was carried out, and the classifier parameters for all the experiments are the same.

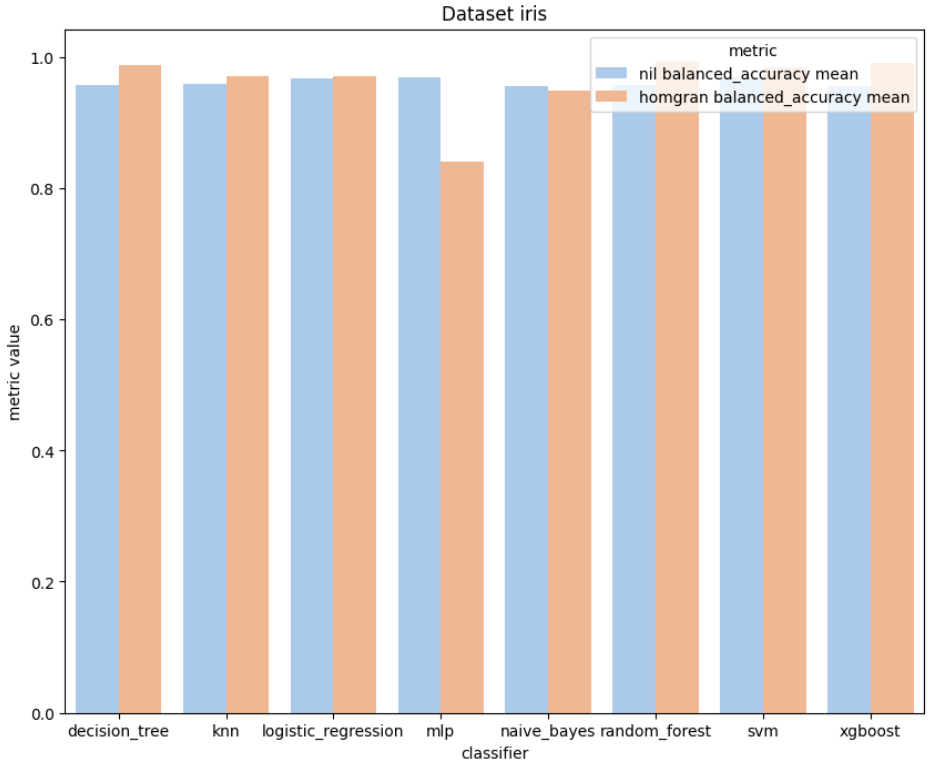


Figure 3.10: Bar plot nil-case vs homogeneous granulation balanced accuracy for iris dataset.

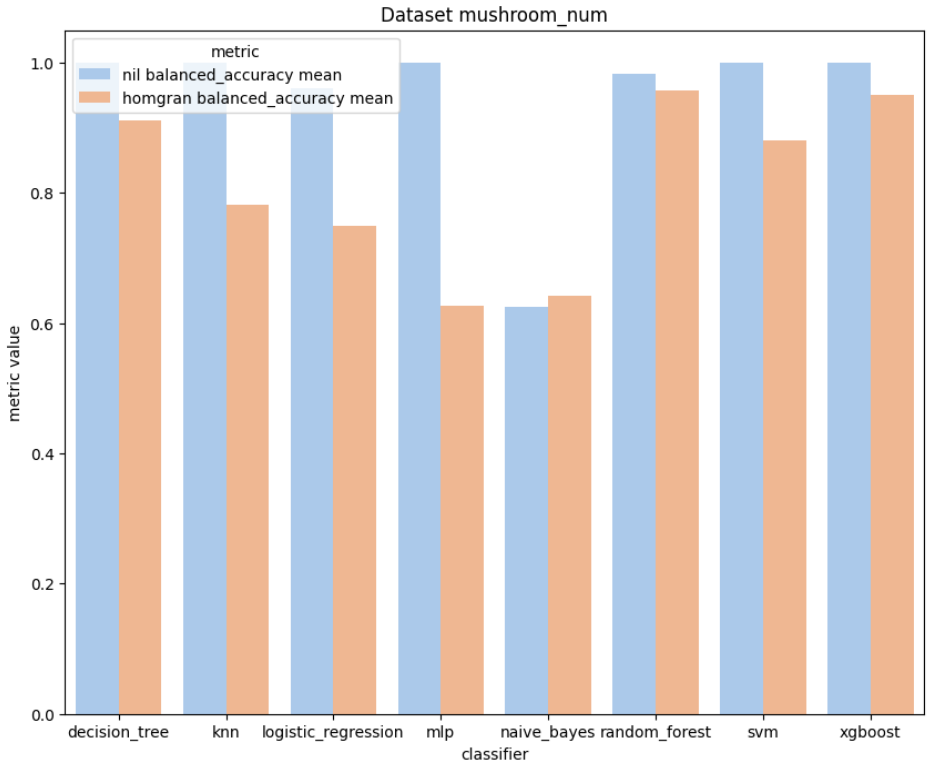


Figure 3.11: Bar plot nil-case vs homogeneous granulation balanced accuracy for mushroom num dataset.

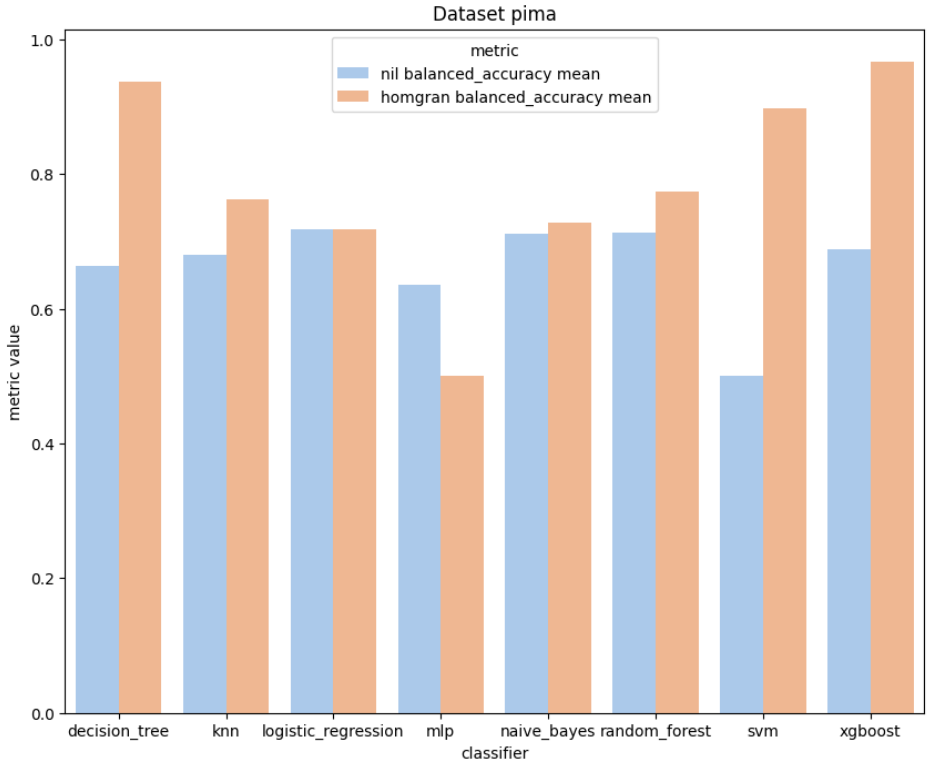


Figure 3.12: Bar plot nil-case vs homogeneous granulation balanced accuracy for pima dataset.

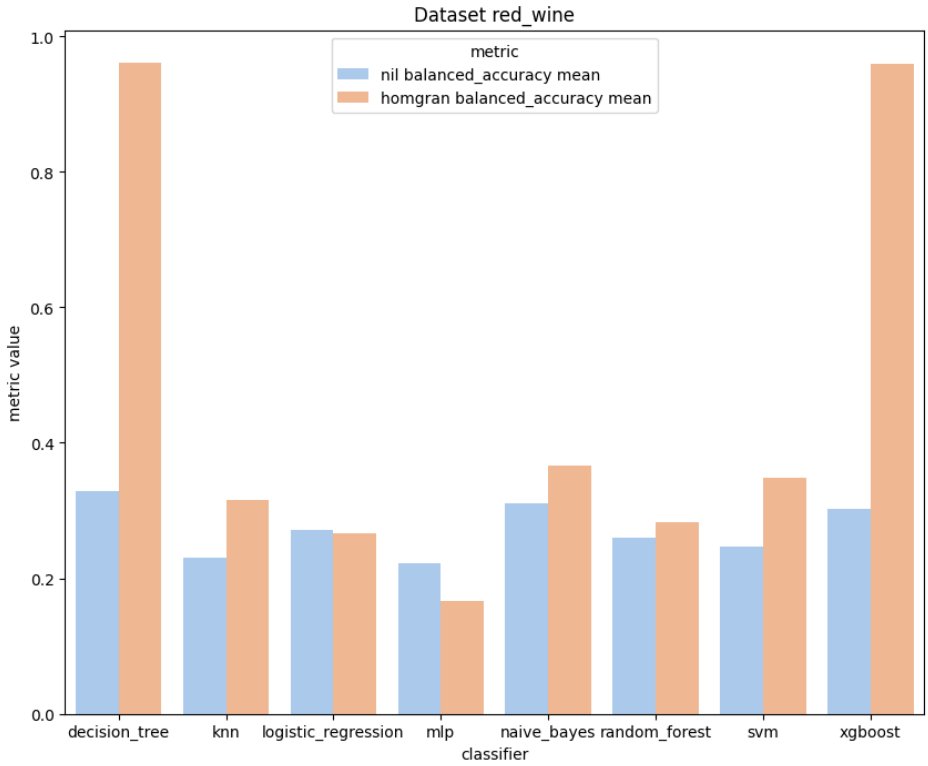


Figure 3.13: Bar plot nil-case vs homogeneous granulation balanced accuracy for red wine dataset.

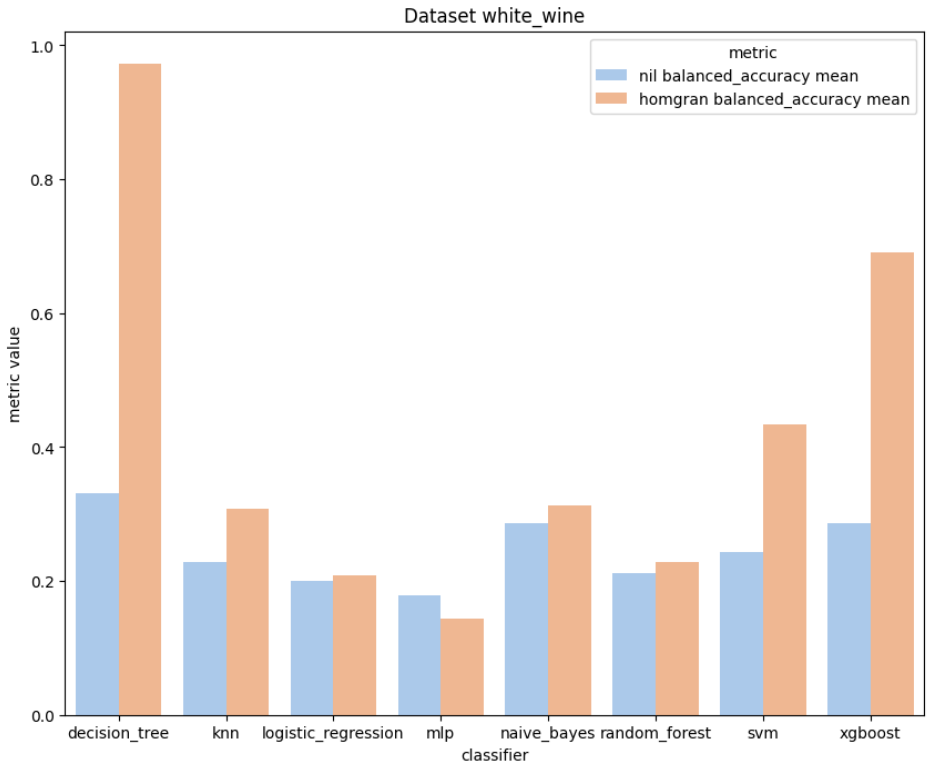


Figure 3.14: Bar plot nil-case vs homogeneous granulation balanced accuracy for white wine dataset.

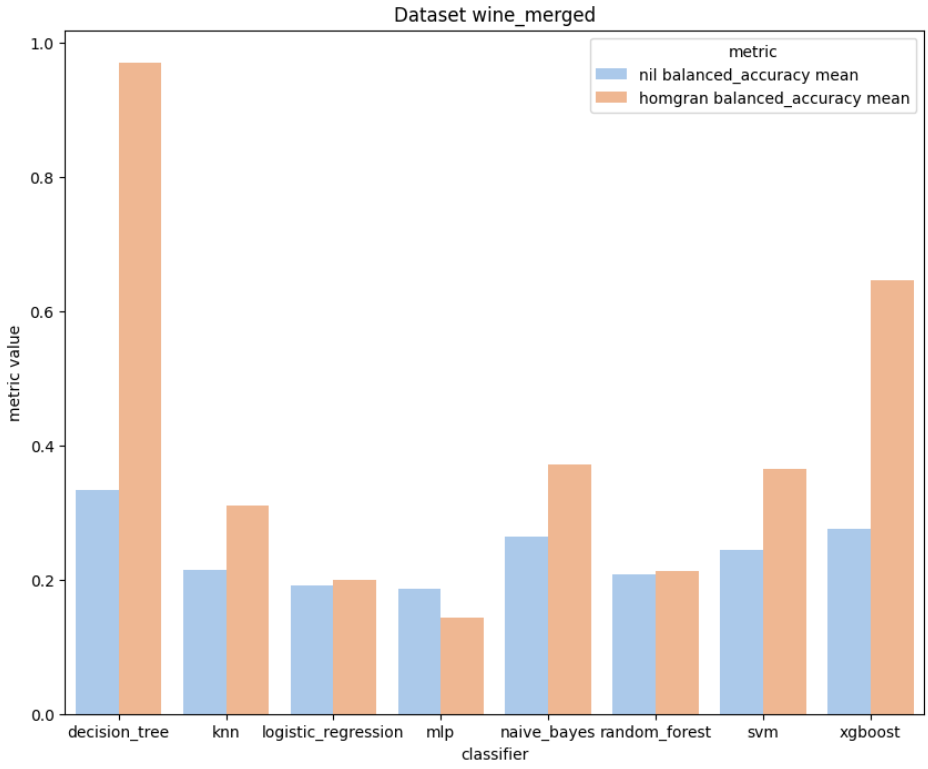


Figure 3.15: Bar plot nil-case vs homogeneous granulation balanced accuracy for wine merged dataset.

3.4. Conclusions

Homogeneous granulation is an effective method of reducing the size of a dataset to preserve the information contained in it, as proven by the classification results for the original and granulated data using this method. The additional computational overhead of granulation may be worth taking in many cases to reduce the number of parameters of the machine learning model both at the learning stage and the subsequent utilization of less memory of the learned model. The greater the homogeneity of the set, the better the effect of granulation and thus the approximation of the granulated set.

4. Epsilon homogeneous granulation

In this section, we present an extension of the homogeneous granulation technique to a variant that works on numerical data - the epsilon homogeneous granulation method. In this variant, by creating homogeneous granules by randomising objects and creating a group of training objects around them that are discernible in the lowest possible degree with no contradiction, we use the degree of indiscernibility of descriptors when comparing $varepsilon$ descriptors. As in homogeneous ordinary granulation, the radius of granulation is set adaptively. Once the granules are formed, a similar procedure of covering and generation of new objects follows. The final result is a granular decision system that is a reflection of the original training one. A characteristic feature of this method is the need to apply a degree of indiscernibility of the descriptors in the Majority Voting procedure. In order to test the effectiveness of this method, we assumed its use in a classification scenario where the granular training systems were the new training systems. We used selected decision systems from the UCI repository, and an example kNN classifier. The method demonstrated its classification performance with a significant redaction of the number of objects relative to the training system.

4.1. Motivation

The main purpose of creating this method was to adapt homogeneous granulation for application to numerical data. We introduced the r-indiscernibility factor during the granulation procedure. And we have designed the mechanisms used in granularity applying this additional parameter in relevant other places, e.g. in Majority Voting.

4.2. Relevant definitions

Let us start by recalling the definition of standard rough inclusion.

$$\mu(v,u,r) \Leftrightarrow \frac{|Ind(u,v)|}{|A|} \geq r \tag{4.1}$$

where

$$IND(u,v) = \{a \in A : a(u) = a(v)\}, \tag{4.2}$$

It follows that this rough inclusion extends the indiscernibility relation to a degree of r.

4.2.1. ε–modification of the standard rough inclusion

Given a parameter ε valued in the unit interval $[0,1]$, we define the set

$$Ind_{\varepsilon}(u,v) = \{a \in A : dist(a(u),a(v)) \leq \varepsilon\}, \tag{4.3}$$

and, we set

$$\mu_{\varepsilon}(v,u,r) \Leftrightarrow \frac{|Ind_{\varepsilon}(u,v)|}{|A|} \geq r \tag{4.4}$$

The rough inclusion extends the indiscernibility relation to a degree of r.

4.2.2. Definition of homogeneous epsilon granules and granulation steps

Granules can be represented as follows:

$$g_{r_u}^{\varepsilon,homogenous} = \{v \in U : |g_{r_u}^{\varepsilon-cd}| - |g_{r_u}^{\varepsilon}| == 0,\ for\ minimal\ r_u\ fulfills\ the\ equation\}$$

where

$$g_{r_u}^{\varepsilon,cd}(u) = \{v \in U : \frac{IND_{\varepsilon}(u,v)}{|A|} \leq r_u\ AND\ d(u) == d(v)\}$$

and

$$g_{r_u}^{\varepsilon}(u) = \{v \in U : \frac{IND_{\varepsilon}(u,v)}{|A|} \leq r_u\}$$

$$r_u = \{\frac{0}{|A|}, \frac{1}{|A|}, ..., \frac{|A|}{|A|}\}$$

$$IND_\varepsilon(u,v) = \{a \in A : \frac{|a(u) - a(v)|}{max_a - min_a} \leq \varepsilon\}$$

where max_a, min_a are the maximal and minimal attribute values for $a \in A$ in the original data set.

Coverage of the original training system is analogous to that of normal homogeneous granulation. That is, we form random granules until all objects have been used.

In forming the final form of the objects, we apply a degree of indiscernibility in the Majority Voting procedure of the form $\frac{|a(u_i)-a(u_j)|}{max_a-min_a} \leq \varepsilon$, i, j are the numbers of objects in granule.

4.2.3. Metrics for granulation and classification

The Hamming metric - for symbolic data is defined as

$$d_H(u,v) = |\{a \in A : a(u) \neq a(v)\}|. \tag{4.5}$$

ε-normalized Hamming metric is a modification for numerical, for given ε, is defined as

$$d_{H,\varepsilon}(u,v) = |\{a \in A : \frac{|a(u) - a(v)|}{max_a - min_a} > \varepsilon\}|. \tag{4.6}$$

4.2.4. k-NN method for evaluation of epsilon homogeneous granulation

The k-NN classifier use modified epsilon Hamming metric, where the descriptors are treated as indiscernible in case $\frac{|a(u)-a(v)|}{max_a-min_a} \leq \varepsilon$. The similar form of this classification was proposed in [30].

Procedure

Step 1. Granulated training data set $(G^{trn}_{r_{gran}}, A, d)$ and the test decision set (U_{tst}, A, d) have been chosen, where A is a set of conditional attributes, d the decision attribute, and, r_{gran} a granulation radius.

Step 2. Classification of test objects by means of granules of training objects is performed as follows.

For all conditional attributes $a \in A$, training objects $v \in G^{trn}$, and test objects $u \in U_{tst}$, we compute weights $w(u,v)$ based on the ε-normalized Hamming metric.

In the voting procedure of the kNN classifier, we use optimal k estimated by CV5 (Cross Validation with 5 folds), details of the procedure are highlighted in next section.

If the cardinality of the smallest training decision class is less than k, we apply the value for $k = |the\ smallest\ training\ decision\ class|$.

The test object u is classified by means of weights computed for all training objects v. Weights are sorted in increasing order as,

$$w_1^{c_1}(u, v_1^{c_1}) \leq w_2^{c_1}(u, v_2^{c_1}) \leq \ldots \leq w_{|C_1|}^{c_1}(u, v_{|C_1|}^{c_1});$$

$$w_1^{c_2}(u, v_1^{c_2}) \leq w_2^{c_2}(u, v_2^{c_2}) \leq \ldots \leq w_{|C_2|}^{c_2}(u, v_{|C_2|}^{c_2});$$

$$\ldots$$

$$w_1^{c_m}(u, v_1^{c_m}) \leq w_2^{c_m}(u, v_2^{c_m}) \leq \ldots \leq w_{|C_m|}^{c_m}(u, v_{|C_m|}^{c_m}),$$

where $C_1, C_2, ..., C_m$ are all decision classes in the training set.

Based on computed and sorted weights, training decision classes vote by means of the following parameter, where c runs over decision classes in the training set,

$$Concept\_weight_c(u) = \sum_{i=1}^{k} w_i^c(u, v_i^c). \tag{4.7}$$

Finally, the test object u is classified into the class c with a minimal value of $Concept\_weight_c(u)$.

After all test objects u are classified, the quality parameter of *accuracy, acc* is computed, according to the formula

$$acc = \frac{number\ of\ correctly\ classified\ objects}{number\ of\ classified\ objects}.$$

Table 4.1: Estimated parameters for kNN based on 5×CV5

$name$	$optimal\ k$
$Australian-credit$	5
$German-credit$	18
$Heartdisease$	19
$Hepatitis$	3

4.2.5. Parameter estimation in kNN classifier

The parameter for experiments were estimated in [30]. The optimal k is presented in Table 4.1.

4.2.6. Toy example of epsilon homogeneous granulation

Considering training decision system

Table 4.2: Training data system (U_{trn}, A, d), (a sample from australian credit data set), for $varepsilon = 0.05$

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	d
u_1	1	20.17	8.17	2	6	4	1.96	1	1	14	0	2	60	159	1
u_2	1	34.92	5	2	14	8	7.5	1	1	6	1	2	0	1001	1
u_3	1	58.58	2.71	2	8	4	2.415	0	0	0	1	2	320	1	0
u_4	1	29.58	4.5	2	9	4	7.5	1	1	2	1	2	330	1	1
u_5	0	19.17	0.58	1	6	4	0.585	1	0	0	1	2	160	1	0
u_6	1	23.08	2.5	2	8	4	1.085	1	1	11	1	2	60	2185	1
u_7	0	21.67	11.5	1	5	3	0	1	1	11	1	2	0	1	1
u_8	1	27.83	1	1	2	8	3	0	0	0	0	2	176	538	0
u_9	1	41.17	1.33	2	2	4	0.165	0	0	0	0	2	168	1	0
u_{10}	1	41.58	1.75	2	4	4	0.21	1	0	0	0	2	160	1	0
u_{11}	1	22.5	0.12	1	4	4	0.125	0	0	0	0	2	200	71	0
u_{12}	1	33.17	3.04	1	8	8	2.04	1	1	1	1	2	180	18028	1
u_{13}	1.234	22.08	11.46	2	4	4	1.585	0	0	0	1	2	100	1213	0
u_{14}	0	58.67	4.46	2	11	8	3.04	1	1	6	0	2	43	561	1
u_{15}	1	33.5	1.75	2	14	8	4.5	1	1	4	1	2	253	858	1
u_{16}	0	18.92	9	2	6	4	0.75	1	1	2	0	2	88	592	1
u_{17}	1	20	1.25	1	4	4	0.125	0	0	0	0	2	140	5	0
u_{18}	1	19.5	9.58	2	6	4	0.79	0	0	0	0	2	80	351	0
u_{19}	0	22.67	3.8	2	8	4	0.165	0	0	0	0	2	160	1	0
u_{20}	1	17.42	6.5	2	3	4	0.125	0	0	0	0	2	60	101	0
u_{21}	1	41.42	5	2	11	8	5	1	1	6	1	2	470	1	1
u_{22}	1	20.67	1.25	1	8	8	1.375	1	1	3	1	2	140	211	0
u_{23}	1	48.08	6.04	2	4	4	0.04	0	0	0	0	2	0	2691	1
u_{24}	0	28.17	0.58	2	6	4	0.04	0	0	0	0	2	260	1005	0

Table 4.3: Granular decision system formed from Covering granules

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	d
$g_{0.5}(u_2)$	1	34.92	5	2	14	8	7.5	1	1	6	1	2	0	1001	1
$g_{0.571429}(u_3)$	1	58.58	2.71	2	8	4	0.165	0	0	0	0	2	320	1	0
$g_{0.5}(u_5)$	0	19.17	0.58	2	6	4	0.21	1	0	0	0	2	160	1	0
$g_{0.5}(u_6)$	1	20.17	8.17	2	6	4	1.96	1	1	14	1	2	60	159	1
$g_{0.5}(u_7)$	0	21.67	11.5	1	5	3	0	1	1	11	1	2	0	1	1
$g_{0.5}(u_8)$	1	27.83	1.33	1	2	4	0.165	0	0	0	0	2	176	1	0
$g_{0.642857}(u_{12})$	1	33.17	3.04	1	8	8	2.04	1	1	1	1	2	180	18028	1
$g_{0.571429}(u_{13})$	1.234	22.08	11.46	2	4	4	1.585	0	0	0	1	2	100	1213	0
$g_{0.5}(u_{16})$	0	20.17	8.17	2	6	4	1.96	1	1	14	0	2	60	561	1
$g_{0.642857}(u_{18})$	1	19.5	9.58	2	6	4	0.79	0	0	0	0	2	80	351	0
$g_{0.642857}(u_{20})$	1	22.5	1.33	2	4	4	0.165	0	0	0	0	2	168	1	0
$g_{0.5}(u_{21})$	1	34.92	5	2	14	8	7.5	1	1	6	1	2	0	1001	1
$g_{0.642857}(u_{22})$	1	20.67	1.25	1	8	8	1.375	1	1	3	1	2	140	211	0
$g_{0.642857}(u_{23})$	1	48.08	6.04	2	4	4	0.04	0	0	0	0	2	0	2691	1

Epsilon Homogeneous granules for all training objects:

$g_{0.571429}(u_1) = (u_1)$, $g_{0.5}(u_2) = (u_2, u_4, u_{15}, u_{21})$, $g_{0.571429}(u_3) = (u_3, u_9, u_{19}, u_{20})$,

$g_{0.5}(u_4) = (u_1, u_2, u_4, u_6, u_{21})$, $g_{0.5}(u_5) = (u_5, u_{10}, u_{19}, u_{24})$, $g_{0.5}(u_6) = (u_1, u_4, u_6)$,

$g_{0.5}(u_7) = (u_7)$, $g_{0.5}(u_8) = (u_8, u_9, u_{11}, u_{17})$, $g_{0.642857}(u_9) = (u_9, u_{10}, u_{11}, u_{17}, u_{19}, u_{20})$,

$g_{0.642857}(u_{10}) = (u_9, u_{10}, u_{19})$, $g_{0.642857}(u_{11}) = (u_9, u_{11}, u_{17}, u_{19}, u_{20})$,

$g_{0.642857}(u_{12}) = (u_{12})$, $g_{0.571429}(u_{13}) = (u_{13})$, $g_{0.428571}(u_{14}) = (u_2, u_{14}, u_{16}, u_{21})$,

$g_{0.5}(u_{15}) = (u_2, u_{12}, u_{15}, u_{21})$, $g_{0.5}(u_{16}) = (u_1, u_{14}, u_{16})$, $g_{0.642857}(u_{17}) = (u_9, u_{11}, u_{17}, u_{20})$,

$g_{0.642857}(u_{18}) = (u_{18})$, $g_{0.571429}(u_{19}) = (u_3, u_9, u_{10}, u_{11}, u_{17}, u_{19}, u_{20}, u_{24})$,

$g_{0.642857}(u_{20}) = (u_9, u_{11}, u_{17}, u_{19}, u_{20})$, $g_{0.5}(u_{21}) = (u_2, u_4, u_{14}, u_{15}, u_{21})$,

$g_{0.642857}(u_{22}) = (u_{22})$, $g_{0.642857}(u_{23}) = (u_{23})$, $g_{0.642857}(u_{24}) = (u_{24})$,

Granules covering training system by random choice:

Covering granules: $g_{0.5}(u_2) = (u_2, u_4, u_{15}, u_{21})$, $g_{0.571429}(u_3) = (u_3, u_9, u_{19}, u_{20})$,

$g_{0.5}(u_5) = (u_5, u_{10}, u_{19}, u_{24})$, $g_{0.5}(u_6) = (u_1, u_4, u_6)$, $g_{0.5}(u_7) = (u_7)$,

$g_{0.5}(u_8) = (u_8, u_9, u_{11}, u_{17})$, $g_{0.642857}(u_{12}) = (u_{12})$, $g_{0.571429}(u_{13}) = (u_{13})$,

$g_{0.5}(u_{16}) = (u_1, u_{14}, u_{16})$, $g_{0.642857}(u_{18}) = (u_{18})$, $g_{0.642857}(u_{20}) = (u_9, u_{11}, u_{17}, u_{19}, u_{20})$,

$g_{0.5}(u_{21}) = (u_2, u_4, u_{14}, u_{15}, u_{21})$, $g_{0.642857}(u_{22}) = (u_{22})$, $g_{0.642857}(u_{23}) = (u_{23})$,

Granular decision system from above granules is as follows:

Table 4.4: Data Sets description

$name$	$attr\ type$	$attr\ no.$	$obj\ no.$	$class\ no.$
$Australian-credit$	$categorical,\ integer,\ real$	15	690	2
$German-credit$	$categorical,\ integer$	21	1000	2
$Heartdisease$	$categorical,\ real$	14	270	2
$Hepatitis$	$categorical,\ integer,\ real$	20	155	2

4.3. Experimental Session

To verify effectiveness and to obtain first sight on behaviour of epsilon homogeneous granulation we have performed a series of experiments with data from UCI Repository [16] - see Tab. 4.4. We have implemented the tests in C++. The model we used is multiple cross validation 5. The main classifier used to verify the protection of internal knowledge in the process of granulation was k-NN with modified epsilon hamming metric. The optimal values of k that were used in this research where the ones identified in [30] and presented in Table 4.1. Seeing the results for considered data in [30] we used $\varepsilon = 0.05$ in granulation and classification.

The result of experiments is presented in Table 4.5. The approximation quality seems to be comparable with our best previous methods. To show the difference we published the result for concept dependent granulation in Table 4.6. In Table 4.5 we can see also the result for homogeneous granulation dedicated to symbolic data. We observed a slight lowering of granular decision system size for $varepsilon$-homogeneous granulation in comparison with homogeneous granulation with similar result of classification. Due to the lack of space we have shown only exemplary results.

4.4. Section summary

In this section, we have presented our new technique of homogeneous granulation in a variant for numerical data.

We have defined the relevant elements necessary to implement this technique in practice. In this, we have modified the granularity peaks to take into account

Table 4.5: The result for homogeneous granulation (HG) and for epsilon homogeneous granulation ($\varepsilon - HGS$) - 5 times CV5 method; HG_acc = average accuracy for HG, $\varepsilon - HG\_acc$ average accuracy for $\varepsilon - HGS$, $HGS\_size = HG$ decision system size, $\varepsilon - HGS\_size = \varepsilon - HGS$ decision system size, $TRN\_size = training\ set\ size$, HG_TRN_red =reduction in object number in training set for HG, $\varepsilon - HGS\_size$ =reduction in object number in training set for $\varepsilon - HGS$, HG_r_range =spectrum of radii for HG, $\varepsilon - HG\_r\_range$ = spectrum of radii for $\varepsilon - HGS$

$results$	$Australian - credit$	$German - credit$	$Heartdisease$	$Hepatitis$
HG_acc	0.835	0.725	0.833	0.88
$\varepsilon - HG\_acc$	0.842	0.725	0.831	0.87
HGS_size	286.52	513.3	120.5	46.16
$\varepsilon - HGS\_size$	274.52	503	109.4	46.2
TRN_size	552	800	216	124
HG_TRN_red	48.1%	35.8%	44.2%	62.8%
εHG_TRN_red	50.3%	37.1%	49.4%	62.7%
HG_r_range	$r_u \geq 0.5$	$r_u \geq 0.6$	$r_u \geq 0.461$	$r_u \geq 0.579$
$\varepsilon - HG\_r\_range$	$r_u \geq 0.571$	$r_u \geq 0.65$	$r_u \geq 0.615$	$r_u \geq 0.579$

Table 4.6: Summary of results, k-NN vs Naive Bayes Classifier, granular and non granular case, acc=accuracy of classification, red=percentage reduction in object number, r=granulation radius, $method$=variant of Naive Bayes classifier

$name$	$k - NN(acc, red, r)$	$k - NN.nil(acc)$
$Australian - credit$	0.851, 71.86, 0.571	0.855
$Car\ Evaluation$	0.865, 73.23, 0.833	0.944
$Diabetes$	0.616, 74.74, 0.25	0.631
$German - credit$	0.724, 59.85, 0.65	0.73
$Heartdisease$	0.83, 67.69, 0.538	0.837
$Hepatitis$	0.884, 60, 0.632	0.89
$Nursery$	0.696, 77.09, 0.875	0.578
$SPECTF\ Heart$	0.802, 60.3, 0.114	0.779

the distance between descriptors. We defined a matching variant of the Majority Voting method and an appropriate kNN classifier procedure, a dedicated classifier for our problem. In the experimental section, we see that the epsilon variant of homogeneous granulation offers the possibility to effectively reduce the number of objects in the training systems - up to approximately 50 per cent - while preserving the internal knowledge of the training systems - as measured by the quality of the classifier. The radii for homogeneous ε granulation are, in many cases, larger than those for homogeneous ordinary granulation, because the granules transition to homogeneous form more quickly. Classification results for both methods are comparable, but for homogeneous ε granulation we obtained a better reduction in training set size. The results are promising the method works effectively on numerical data. The designed methodology offers the possibility of future methods working on data with mixed values. In our variants, data is treated either symbolically or numerically.

Part III

Selected applications of knowledge granulation techniques in data analysis problems

5. A Novel Ensemble Model - The Random Granular Reflections

Ensemble methods are a family of techniques that are very important in the field of data science, often ranking at the forefront of data analysis methods. The effectiveness of ensemble methods stems from the fact that they use a mechanism for tuning the classification in successive learning iterations, applying the available knowledge in a random manner or using a mechanism to focus on problems that have not yet been learned correctly. In this chapter, we present our new technique, which is the result of many years of work on approximation techniques for decision-making systems - the Ensemble of Random Granules. We use the previously discussed homogeneous granules - (see Chapter 3) - as learning components in successive iterations. At the start of the algorithm, we cover the training decision system with homogeneous granules - groups of objects from the same class for each central object that are irreducible from each other to the maximum possible degree. Starting with the inclusion of completely indiscernible objects in the granule, we descend with our indiscernibility requirements until we encounter the first object that contradicts the class of the object on which we create the granule. Then, for each granule of homogeneous coverage, we produce a new object using the chosen strategy - e.g. majority voting. The produced objects take part in the classification process of the test systems. Our results from this chapter show the positive effect of enhancing classification with random granules. The results are comparable to other popular ensemble methods. A very important advantage of this technique is that there is no need for parameter estimation - granulation radii (the degree to which objects are to be indiscernible in order to form granules). Because the formed granule around the selected object spans the data and its size depends on the internal degree of indiscernibility.
In the following sections, we introduce the details needed to understand our Ensemble model. First of all, our technique is derived from the underlying

granulation scheme introduced by Polkowski [25]. In the standard method, a set of objects indiscernible to a fixed degree (radius of granulation) is calculated around each object of the training system, using objects from all available decision classes. The chosen strategy (e.g. random selection) is then applied to cover the calculated granules of the training system, finally creating new objects from the selected granules using the chosen method, e.g. Majority Voting (MV). A characteristic feature of this method, is that it works effectively for medium-sized radii. The advantage is the simplicity of implementation and speed of operation. The disadvantage is that for small radii close to 0, certain decision classes may not be included in the produced granular reflection, so that classification efficiency may be reduced. The described method has become the foundation for a number of further methods for approximation of decision systems, classification, absorption of missing values, creation of novel ensemble models, application in steganography, in the area of deep learning, and many other data science problems. Examples of works from the area of approximation of decision systems are [1]–[2], [24]–[27], [30]. They present a concept-dependent method (forming granules in the area of decision classes) securing the participation of all classes in the classification process. And the layered granulation method, showing successive degrees of approximation of decision systems, by repeating the granulation process. In the papers [33] and [34]) we developed a new granulation method, a homogeneous variant (see details Section 3) Considering a training system (U, A, d), where U is a universe of objects, A a set of conditional attributes, and d a decision attribute not belonging to A. The homogeneous method is based on the creation of granules from r indiscernible objects on at least $r * |A|$ attributes, where $|A|$. The degree of granulation is determined adaptively depending on the degree of indiscernibility of the decision system.

The Ensemble methods are a family of techniques that are among the best for operating with tabular data, and their effectiveness has been verified in many contexts, including in the area of rough sets (see [15, 17, 37, 41]). The main motivation for the creation of the custom ensemble model was the creation of a homogeneous granulation technique. The method is based on the use of random homogeneous granules in the learning process, finally forming the coverage of the original training system and creating its granular reflection. The aim of this

operation is to reduce the size of the training system while preserving the knowledge contained within it, as measured by the effectiveness of the classifier. In the case of homogeneous granulation, the level of reduction in the number of objects depends on the decision-making system used and is close to 50 per cent. Let us present the method in detail, for simplicity we will treat attributes as symbolic. In the experiential session, we used a CSG classifier based on simple knowledge granules (see [3]). We perform 50 learning iterations in each compositional exponent. We make a comparison of our own method with selected ensemble models. The computational complexity of the homogeneous technique is $|U|^2|A|$, U is the universe of objects, A is the set of conditional attributes.
The method can be applied to large data sets, examples of the effectiveness of similar models can be seen in the works, [10] in the context of data streaming, in in [8] the context of data decomposition, [9] in the context of random sampling.
Let us proceed with a brief introduction to the ins and outs of the selected ensemble methods.

5.1. Selected Ensemble models - in a nut shell

We can find an interesting overview of Ensemble methods in the paper [49]). One of the most popular are the Bagging and Boosting methods (see [45]) and Random Forest (see [14]). What our model is comparable to is boosting and multiple bootstrap, hence we will focus on example models of this type.

5.1.1. Bagging - ensemble of bootstraps

. This involves randomly selecting a committee of bootstraps [49]. After dividing the original decision system into $train$ and $valid$ systems, we determine the number of learning iterations in which we randomly select $|train|$ objects with returns from the $train$ system forming $NewTrain_i$ systems. In the ith learning iteration, we classify the $valid$ system in two ways, by classifying it with the current $NewTrain_i$ system and by applying the cumulative committee of all previous classifications - using the Majority Voting method to determine decisions.

5.1.2. Arcing - ensemble of bootstraps

The model we use here is to divide the $train$ system into the $NewTrain$, $NewTest$ system (see [6] and [38]). The partitioning is implemented on a Bootstrap basis, weights are used to assign objects to the $NewTrain$ set. The weights are initialised with equal values, when classifying $NewTest$ with $NewTrain$ the weights for which the classification was correct are lowered. Finally, the weights are appropriately normalised to a representation that allows the objects to be drawn. We will call the Bootstrap formation algorithm Arcing. The final step of this technique is to classify the $valid$ system using $NewTrain$ from a single iteration and using the committee of classifiers that have been formed up to that point. In arcing, the weights are modified using the $\frac{1-acc}{acc}$ factor. Where acc is the percentage of correctly classified objects.

5.1.3. Ada-Boost with random split

A similar technique to Arcing, $NewTrain$ and $NewTest$ are created differently (see [18], [39] and [50]). The objects for $NewTrain$ are selected based on weights, and a fixed factor is used to divide $Train$. The splitting factor is chosen experimentally - in our case it is equal to 0.6 (close to the expected degree of splitting when using Bootstrap). The rest of the algorithm works analogously to Arcing.

5.2. Ensemble of Random Granular Reflections

Each learning step of the Ensemble model uses a different granular reflection formed from random homogeneous granules covering the entire training system in each iteration. The random overlap results in granular reflections formed from other subgroups of objects - intuitively the problem is solved applying slightly different knowledge. You can see the concept of our model in figure 5.1.

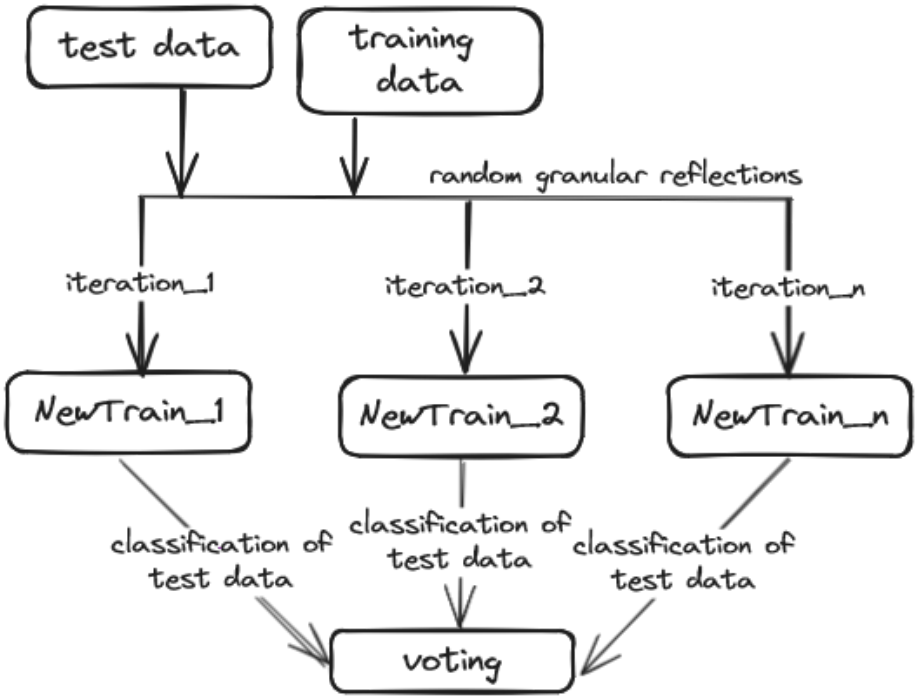


Figure 5.1: Ensemble of Random Granular Reflections

5.3. Experimental Session

Our main goal is to test the extent to which homogeneous granules forming granular reflections of training systems extract knowledge from these systems. To this end, we will recalculate our model for selected decision systems from the UCI repository and contrast the classification with an ensemble model formed from subsets of random objects of objects without the contribution of homogeneous granulation. As a complementary element, we will present a sample of how our method works, compared to several other ensemble models - pure bagging, bootstrap ensemble, ada-boost. We omit the random forest, because in its case we would have to use subsets of attributes, before forming granular systems and methods work differently, incomparably.

5.3.1. Comparison with selected other methods

In figures 5.2, 5.3, 5.4 and 5.5 we have results comparing the effectiveness of single classifiers vs those formed from their cumulative committee. The results show on the exemplary data - Australian credit data set - the process of

classification amplification for the methods: Ensemble of Random Granules, Arcing - ensemble of bootstraps, Ada-Boost and Bagging - ensemble of bootstraps.
The results of Ensemble of random granules are comparable to the other methods presented. Of course, its effectiveness depends on the data and classifiers used. We will verify this in example experiments.

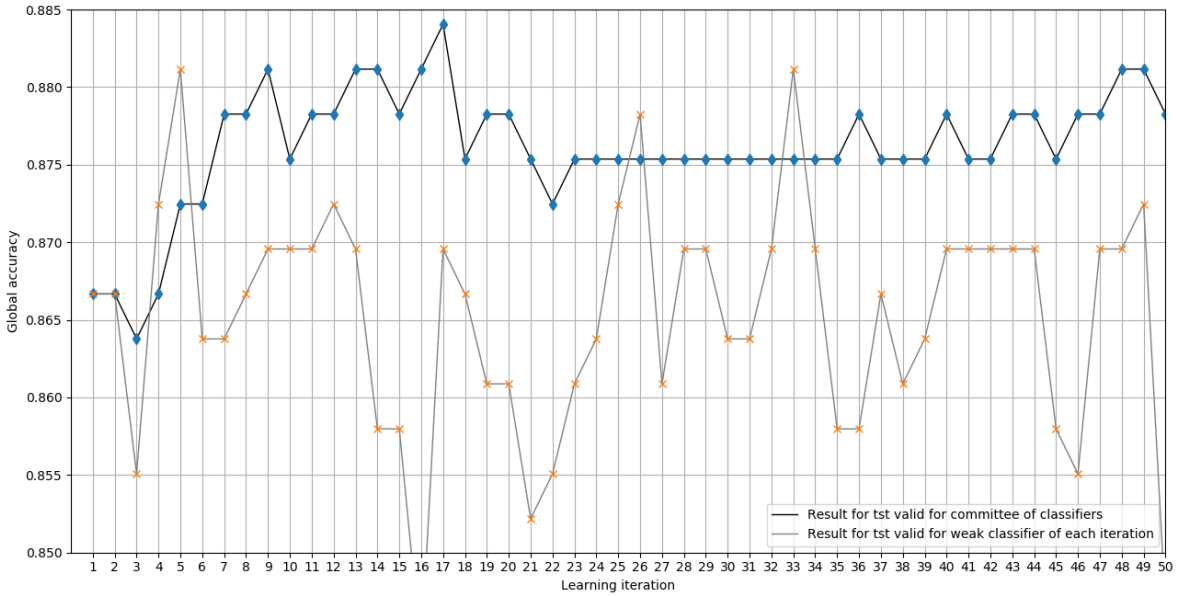


Figure 5.2: Ensemble of Random Granular Reflections for the Australian credit data set - the accuracy of classification - 50 iterations of learning - exemplary run

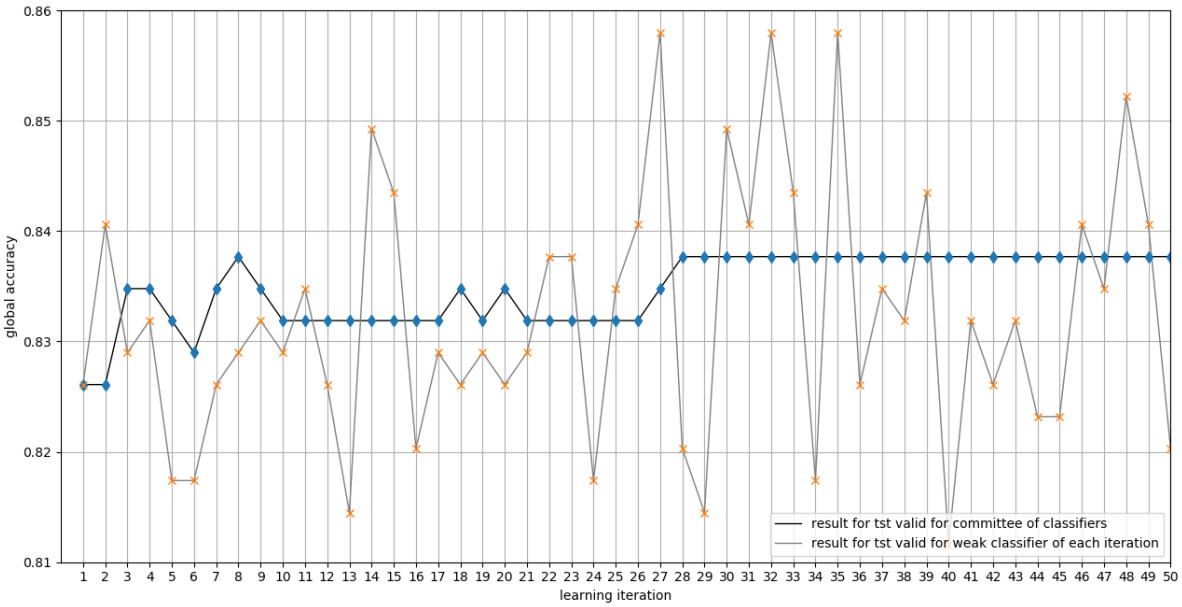


Figure 5.3: Bagging Ensemble model for the Australian credit data set - the accuracy of classification - 50 iterations of learning - exemplary run

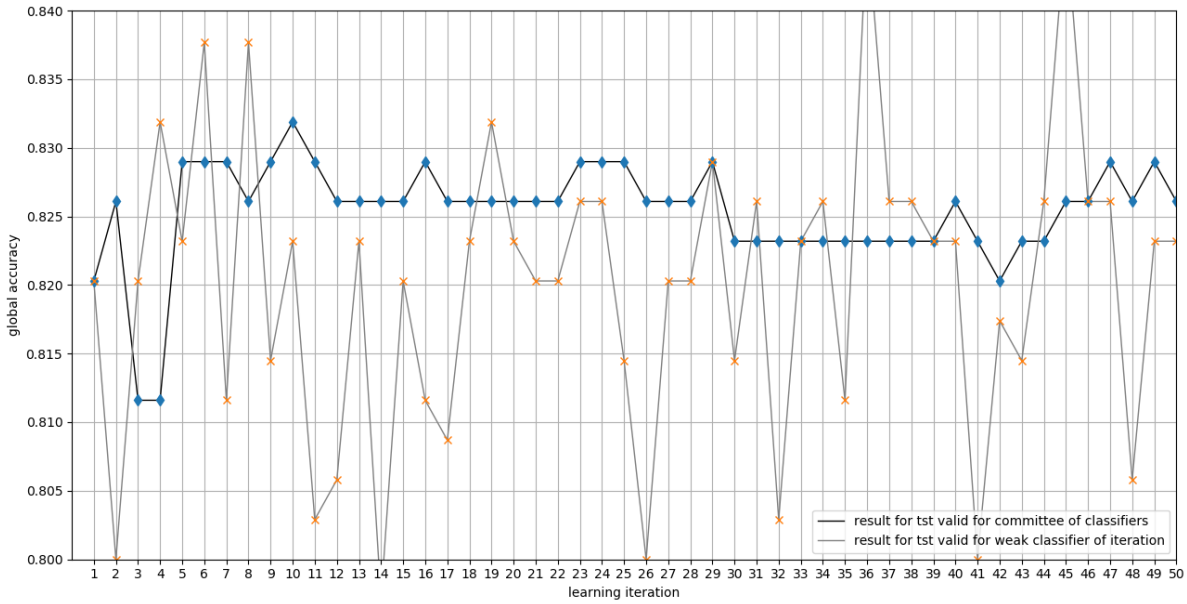


Figure 5.4: Ada-Boost Ensemble model for the Australian credit data set - the accuracy of classification - 50 iterations of learning - exemplary run

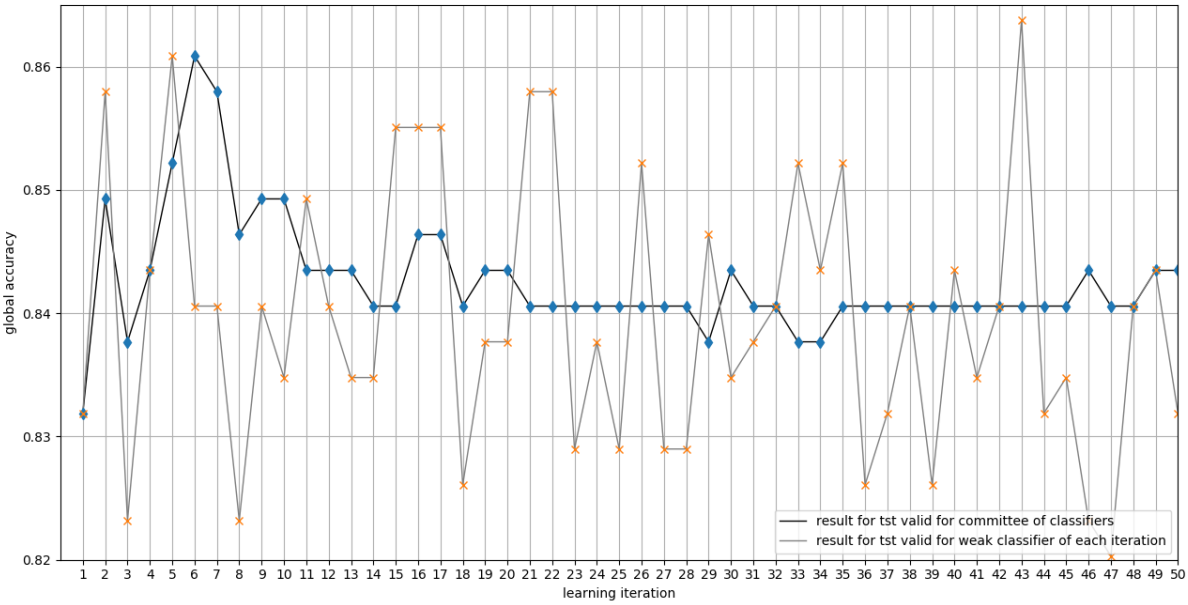
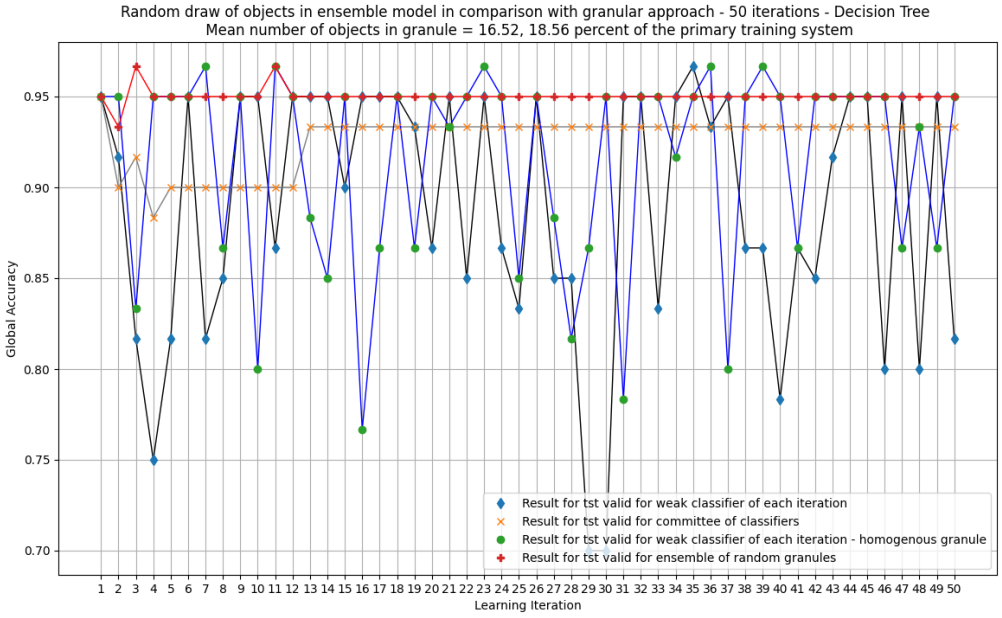


Figure 5.5: Pure Bagging Ensemble model for the Australian credit data set - the accuracy of classification - 50 iterations of learning - exemplary run

5.3.2. Results for selected other decision-making systems - multiple runs

In this section we give a sample of the results, how the ensemble of random granules method behaves on different data. We chose Decision Tree as the reference classifier. In the variant, in order to achieve minimally valuable granular systems, we used two new elements. The first is to give the possibility to use the same objects multiple times within a single granule and we have introduced the possibility to accept objects from other classes.



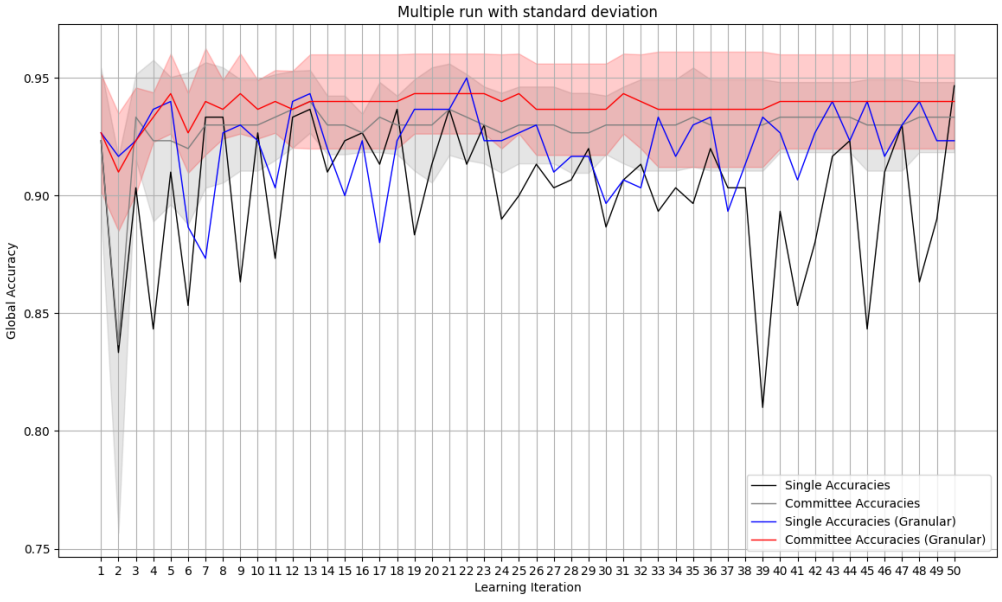
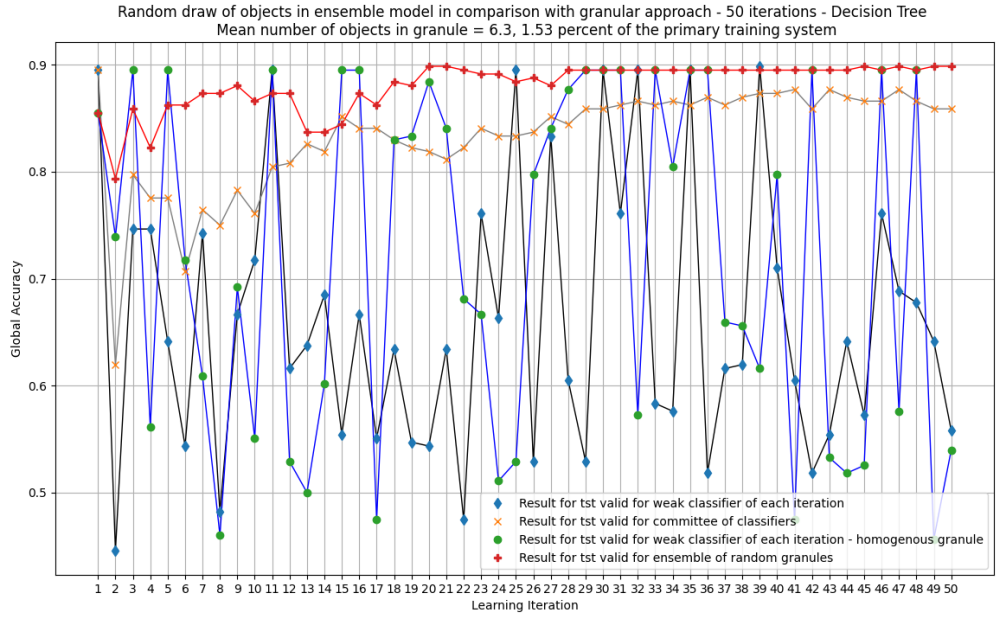


Figure 5.6: Ensemble of Random Granular Reflections for the **Iris data set** - the accuracy of classification - 50 iterations of learning - exemplary run; Decision Tree classifier



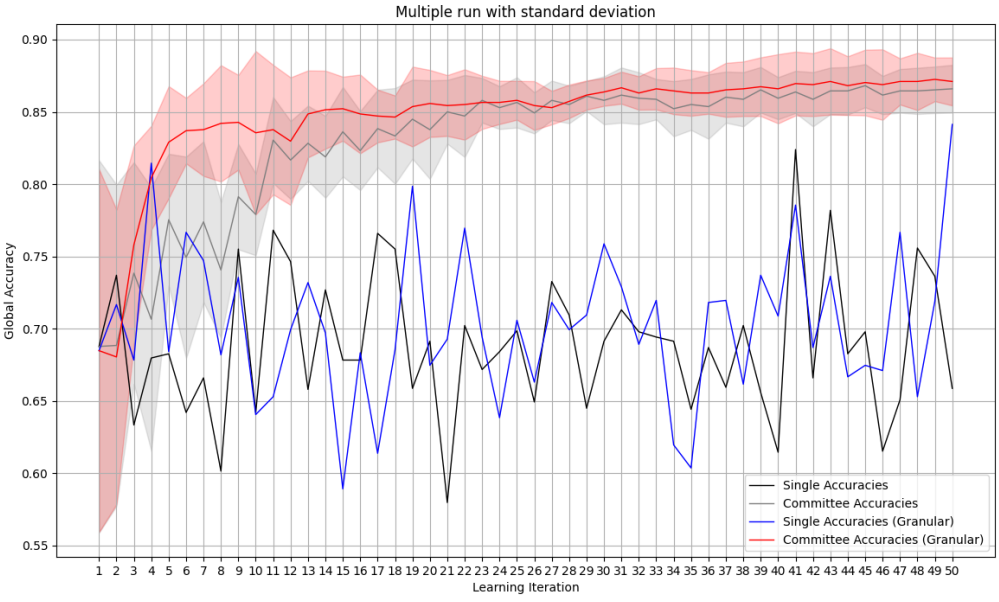
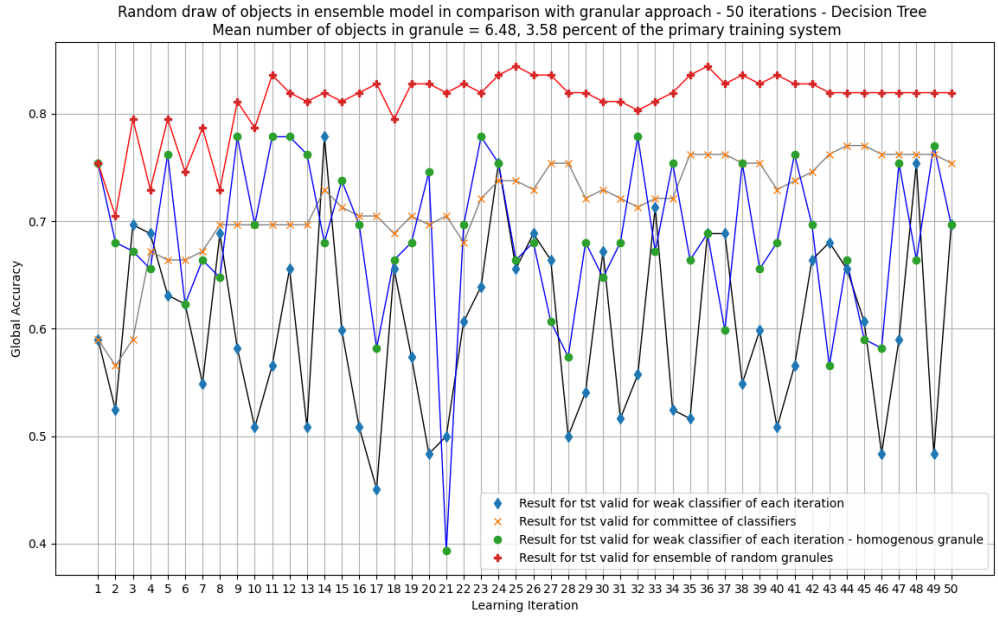


Figure 5.7: Ensemble of Random Granular Reflections for the **Australian credit** data set - the accuracy of classification - 50 iterations of learning - exemplary run; Decision Tree classifier



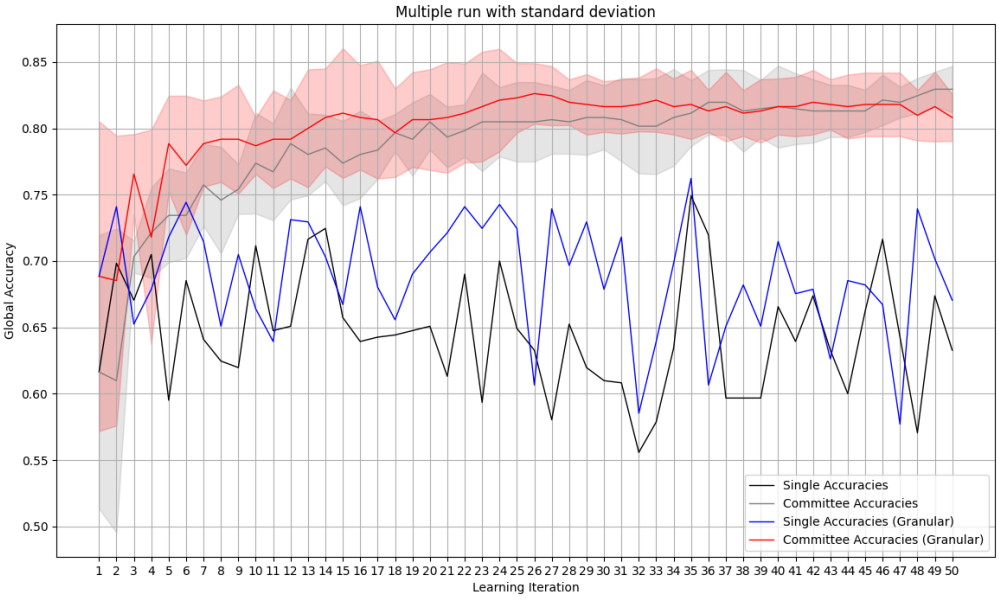
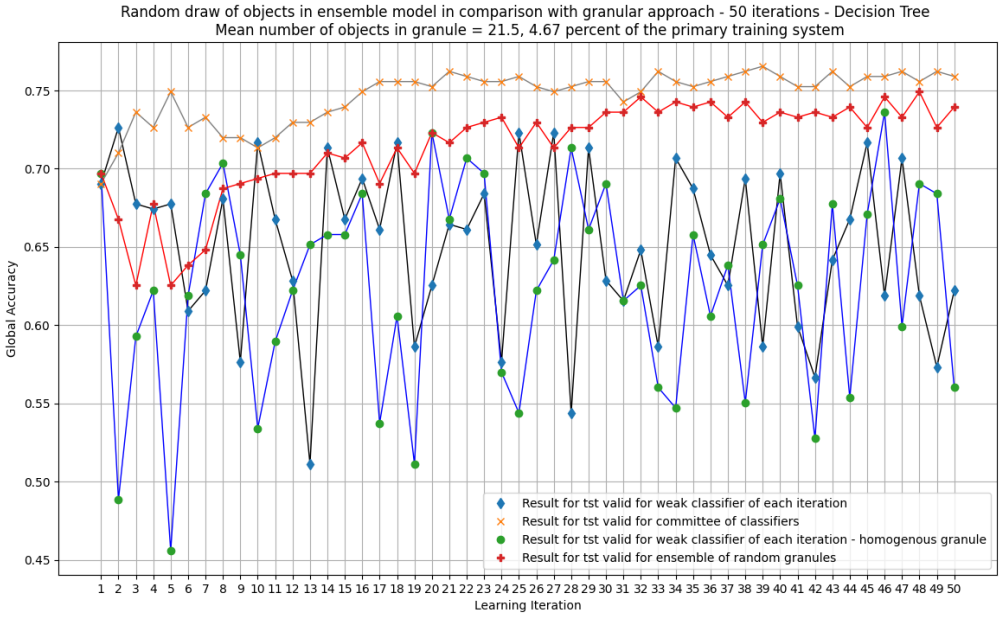


Figure 5.8: Ensemble of Random Granular Reflections for the **Heart Disease data set** - the accuracy of classification - 50 iterations of learning - exemplary run; Decision Tree classifier



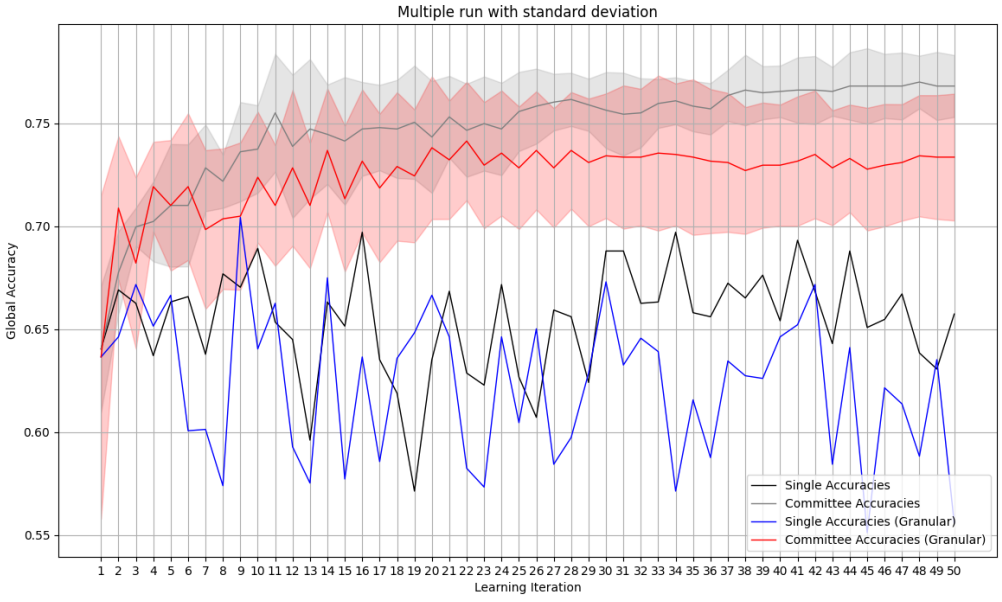
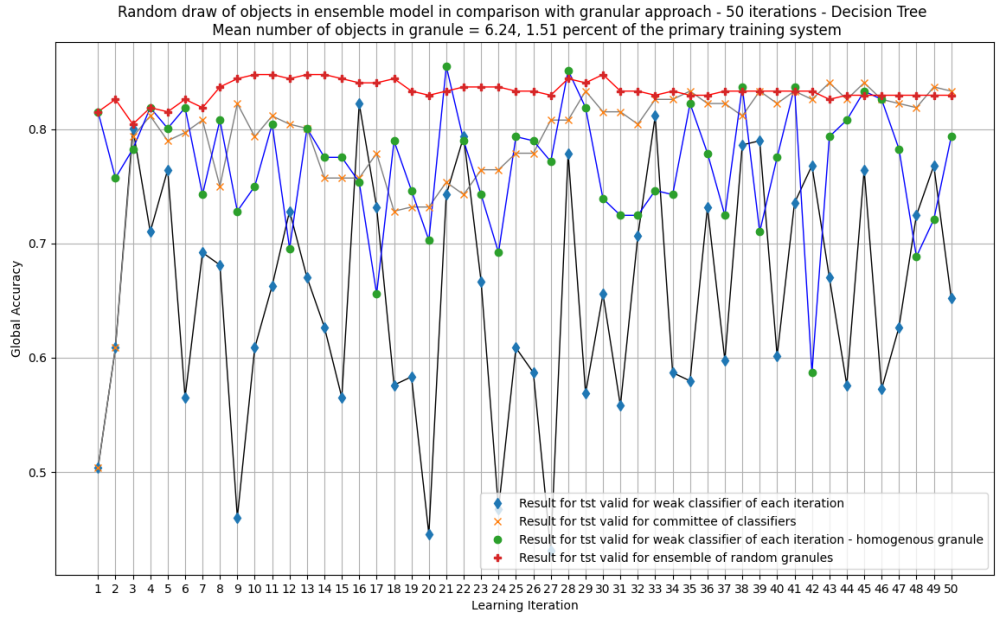


Figure 5.9: Ensemble of Random Granular Reflections for the **Pima indians diabetes data set - dummy version** - the accuracy of classification - 50 iterations of learning - exemplary run; Decision Tree classifier

5.3.3. Testing the performance of other popular classification techniques on selected data

In the second example experiment, we demonstrate the operation of several other classifiers for the selected system, including, in addition to Decision Tree, the operation of Random Forest, kNN and Naive Bayes methods. As can be seen, Random Forests and Decision Tree work well with our method on the selected data, while kNN and Naive Bayes work less well. We can see the results in Figures 5.10, 5.11 and 5.12 respectively.



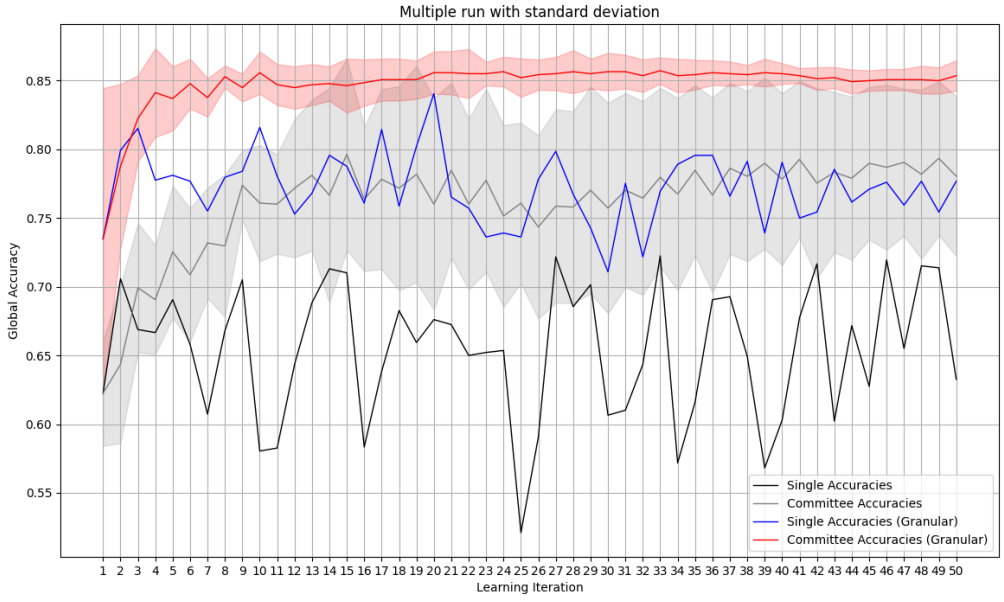
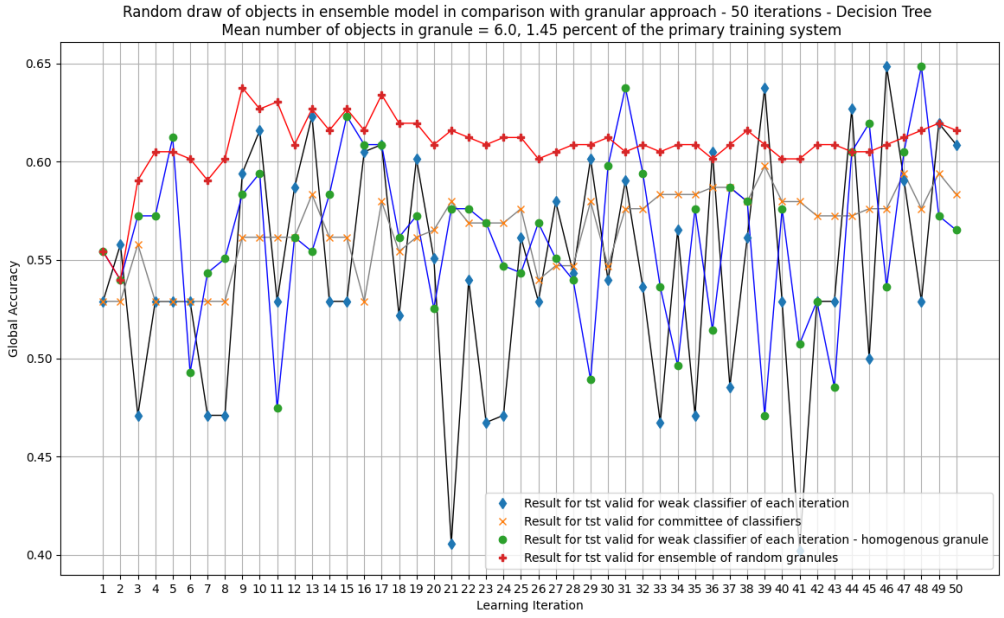


Figure 5.10: Ensemble of Random Granular Reflections for the **Australian credit** data set - the accuracy of classification - 50 iterations of learning - exemplary run; **Random Forest** classifier



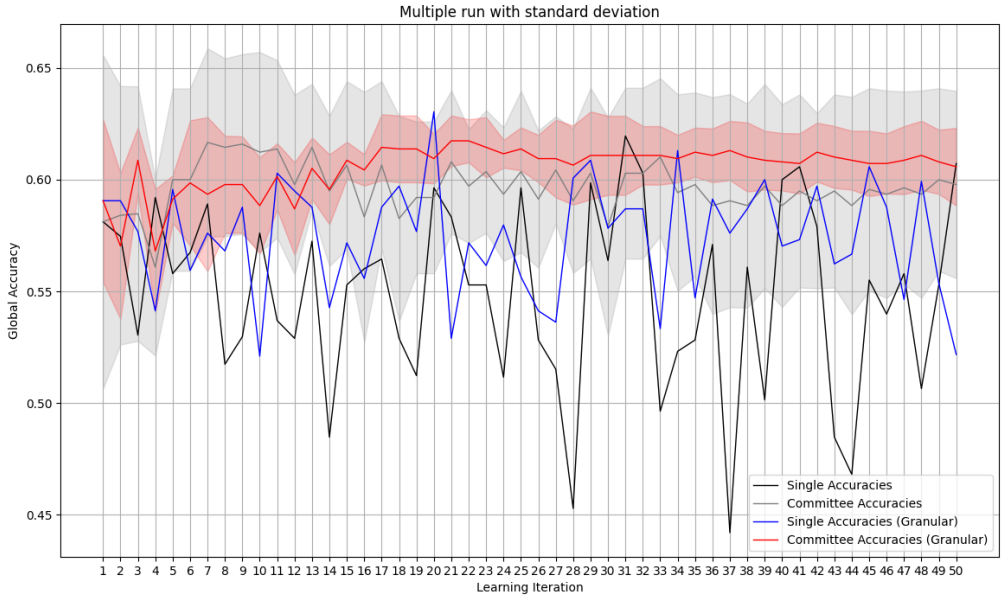
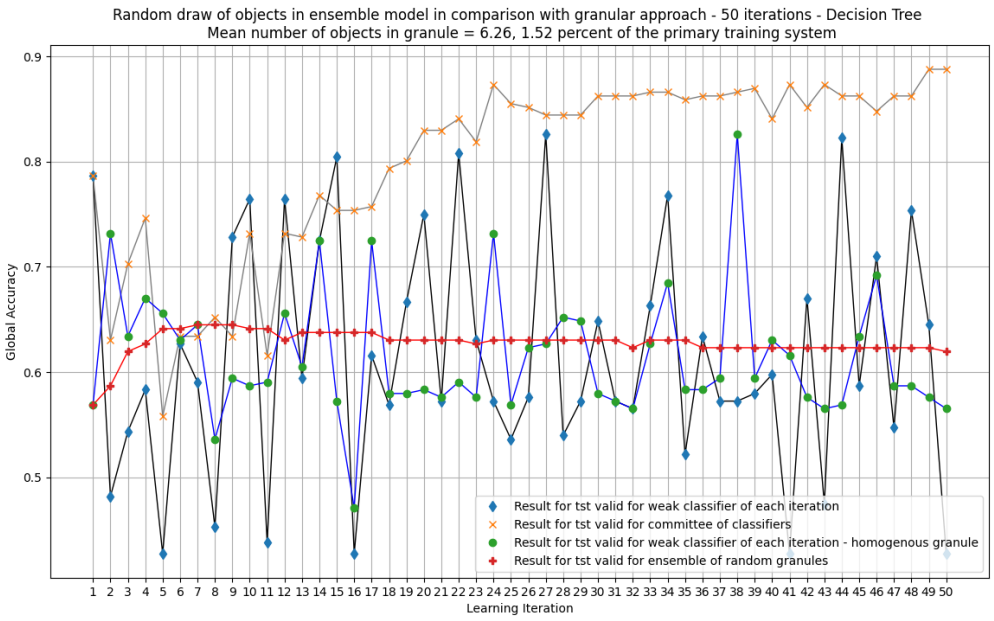


Figure 5.11: Ensemble of Random Granular Reflections for the **Australian credit** data set - the accuracy of classification - 50 iterations of learning - exemplary run; **k Nearest Neighbor** classifier



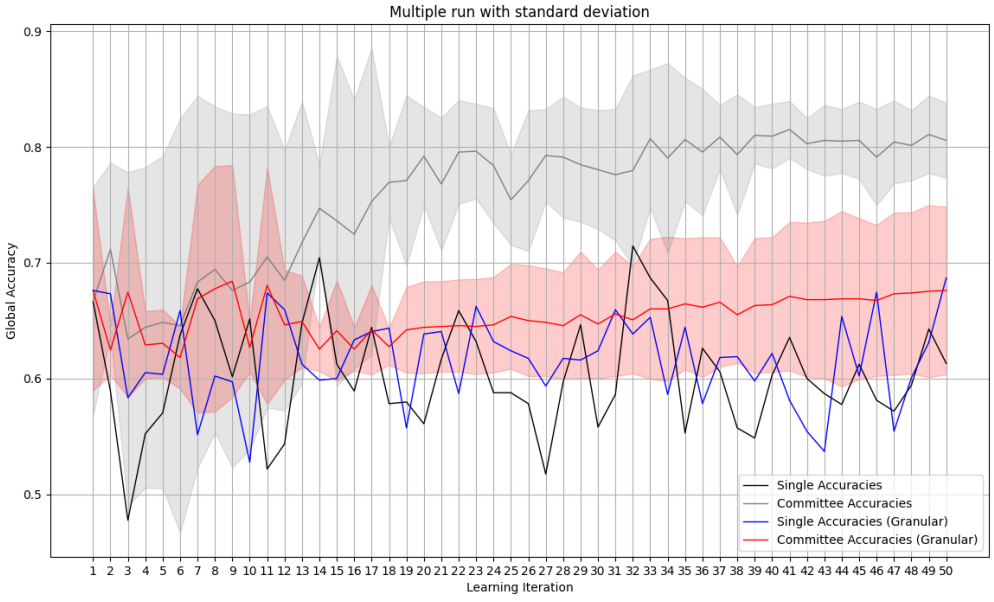


Figure 5.12: Ensemble of Random Granular Reflections for the **Australian credit** data set - the accuracy of classification - 50 iterations of learning - exemplary run; **Naive Bayes** classifier

5.3.4. The operation of the technique on unbalanced data.

The granulation technique we use generally produces representatives from all decision classes of the training system and is partially robust to unbalanced data [13], [11]. We can see research results on this problem in the chapter on the application of SMOTE in 7.3. It is natural that a suitable classifier has to be fitted to the unbalanced data and the evaluation of the classification quality should be computed using balanced parameters - equally representing each decision class of the test system.

5.3.5. A few words about the degree of homogeneity

An interesting question may be the problem of whether there is a minimum radius r_u for which the standard and concept dependent granule are equal defining the homogeneous granule used in our model. For standard granules of radius 1 the granules contain their central objects or objects completely indiscernible from them. In the case of concept-dependent granules, they also contain their centra but cannot contain objects completely indiscernible from other classes. That is, in this variant, the existence of conflicting objects does not stop the granulation process. In the process of learning homogeneous granules, a concept-dependent granule is generated which does not contain contradictory objects in the given degree of indiscernibility to which we have descended. That is, a homogeneous granule is a special case of a concept-dependent granule, but this rule does not necessarily work in reverse. In the case of a 1-radius, it may happen that a homogeneous granule does not contain any object, this situation occurs when the central object of the granule has a completely indiscernible object in another class. In our experiments, we assume that the systems used do not contain completely contradictory objects, to avoid the impossibility of calculating homogeneous granules. The discussion will be summarised in Tab. 5.1, which shows for the selected systems the relevant degrees of indiscernibility (homogeneous) of the radii_range. The degree of homogeneity is greater the lower the minimum value is. This is shown by the smallest radius encountered among all granules.

Table 5.1: The result for homogeneous granulation, $5 \times CV5$, $GS\_size = granular\ decision\ system\ size$, $TRN\_size = training\ set\ size$, $TRN\_reduction = reduction\ in\ object\ number\ in\ training\ size$, $radii\_range = spectrum\ of\ radii$.

$\boldsymbol{name}$	$\boldsymbol{GS\_size}$	$\boldsymbol{TRN\_size}$	$\boldsymbol{TRN\_reduction}$	$\boldsymbol{radii\_range}$
$Australian\ credit$	286.52	552	48.1%	$r_u \geq 0.5$
$Car\ Evaluation$	728.5	1382	47.3%	$r_u \geq 0.667$
$Diabetes$	488.9	614	20.4%	$r_u \geq 0.25$
$German - credit$	513.3	800	35.8%	$r_u \geq 0.6$
$Heartdisease$	120.5	216	44.2%	$r_u \geq 0.461$
$Hepatitis$	46.16	124	62.8%	$r_u \geq 0.579$
$Nursery$	9009.1	10368	13.1%	$r_u \geq 0.875$
$SPECTF\ Heart$	138.75	214	35.2%	$r_u \geq 0.068$

5.4. A few final words

Our new method has demonstrated effectiveness in enhancement of classification over successive learning iterations. Ensemble of Random Granular Reflections effectiveness is comparable to methods such as Bagging, Boosting. In Figs. 5.6, 5.7, 5.8 and 5.9 we have the results for the systems, iris, australian, heart and pima indian diabetes, respectively. As can be seen, effectiveness depends on the data used. It is worth noting the average size of the granules used for the classification. They represent, respectively, for Iris 18.56 per cent of the original training system, for Australian 1.53, for Heart 3.58 and for pima 4.64 per cent. That is, with a large reduction in the size of granular systems, efficiency is maintained and can be effectively enhanced. An example in which the method works poorly is the Pima indians diabetes system - see Fig. 5.9. The use of even small-scale granular systems in the Ensemble model, as can be seen, yields very promising results. **This research opens up a wide horizon for the application of granular methods in Ensemble models.**

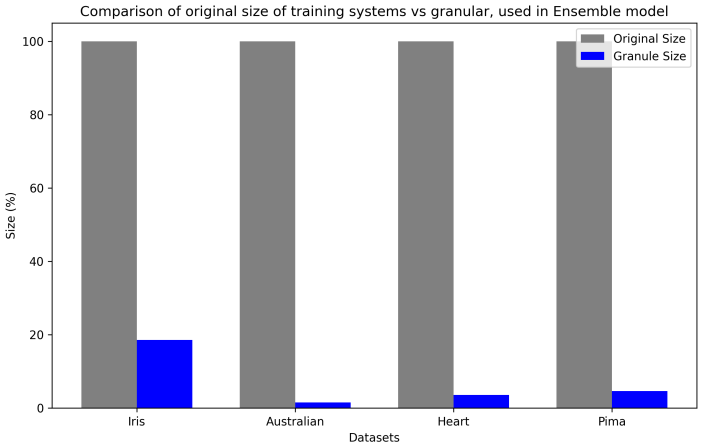


Figure 5.13: Average size of granular systems used in the classification. Despite the use of such small representations, drawing knowledge from the entire original training system, granular systems generated on random central objects show the ability to reinforce classifiers at the level of using the original training data process

6. Missing values handling based on homogeneous granulation

Granularity techniques from Polkowski's family of methods, further developed by Artiemjew and Ropiak, among others, have found application in the process of absorbing missing values. It turns out that the process of creating granular reflections naturally eliminates these values. In this section, we will explore the applicability of our new granulation technique, the homogeneous method [33], in the context mentioned above. Four strategies of missing values absorption were considered - A, B, C and D strategy. Initial experiments, in which we use selected strategies (A - D) are available in Polkowski and Artiemjew [29], [28] and [29] - what was extensively checked on data from UCI Repository in [30]. We begin with a details of strategies chosen by us.

6.0.1. A set of basic strategies

We consider missing values in four various cases,

1. *Strategy A: during building granules* $*$*=don't care*, *in repairing of not absorbed values* $*$, $*$*=don't care*.
2. *Strategy B: during building granules* $*$*=don't care*, *in repairing of not absorbed values* $*$, $* = *$.
3. *Strategy C: during building granules* $* = *$, *in repairing of not absorbed values* $*$, $*$*=don't care*.
4. *Strategy D: during building granules* $* = *$, *in repairing of not absorbed values* $*$, $* = *$.

Considering granulation process - in case of A and B strategy, stars are treated as all possible values. For C and D strategy stars are treated as new values in the system. Granules in the context of our strategies may be defined as follows:

In case of $* = don't\ care$ - granule building phase is as follows

Considering $i-th$ training data set TRN_i - and the phase of granulation, the granules can be defined as:

$$g_{r_{gran}}^{cd,*=don't\ care}(u) = \{v \in TRN_i : \frac{|IND^{*=don't\ care}(u,v)|}{|A|} \leq r_{gran}\ AND\ d(u) = d(v)\},$$

where

$$IND^{*=don't\ care}(u,v) = \{a \in A : a(u) = a(v)\ OR\ a(u) = *\ OR\ a(v) = *\}.$$

In case of $* = *$ - granule building phase is as follows

Granules used in C and D strategies have the form: ,

$$g_{r_{gran}}^{cd,*=*}(u) = \{v \in TRN_i : \frac{|IND^{*=*}(u,v)|}{|A|} \leq r_{gran}\ AND\ d(u) = d(v)\},$$

where

$$IND^{*=*}(u,v) = \{a \in A : a(u) = a(v)\}.$$

In case of $* = don't\ care$ in the repairing phase

In case of A and C strategies, in order to repair objects containing missing values after granulation, we immerse objects with stars on specific positions j into original disturbed training set. We fill the value for the star by means of majority voting on non missing values of the attribute j.

In case of the strategy A, the granule around the disturbed object $MV(g_{r_{gran}}^{cd,*=don't\ care}(u))$ can be defined as follows,

$$\text{if } a_j(MV(g_{r_{gran}}^{cd,*=don't\ care}(u))) = *,$$

then the missing value could be repaired by the granule,

$$g_{r_{gran},a_j}^{cd,*=don't\ care}(MV(g_{r_{gran}}^{cd,*=don't\ care}(u))) =$$

$$\{v \in TRN_i : \frac{|IND_{a_j}^{*=don't\ care}(MV(g_{r_{gran}}^{cd,*=don't\ care}(u)),v)|}{|A|} \leq r_{gran}\ AND\ d(MV(g_{r_{gran}}^{cd,*=don't\ care}(u))) = d(v)\},$$

where

$$IND_{a_j}^{*=don't\ care}(MV(g_{r_{gran}}^{cd,*=don't\ care}(u)),v) =$$

$$\{a \in A : (a(MV(g_{r_{gran}}^{cd,*=don't\ care}(u))) = a(v)\ OR\ a(MV(g_{r_{gran}}^{cd,*=don't\ care}(u))) = *\ OR\ a(v) = *)\ AND\ a_j(v) != *\}.$$

In case of the strategy C, the granule around the disturbed object $MV(g^{cd,*=*}_{r_{gran}}(u))$ can be defined as follows,

$$\text{if } a_j(MV(g^{cd,*=*}_{r_{gran}}(u))) = *,$$

then the missing value could be repaired by the granule,

$$g^{cd,*=don't\ care}_{r_{gran},a_j}(MV(g^{cd,*=*}_{r_{gran}}(u))) =$$

$$\{v \in TRN_i : \frac{|IND^{*=don't\ care}_{a_j}(MV(g^{cd,*=*}_{r_{gran}}(u)), v)|}{|A|} \leq r_{gran}\ AND\ d(MV(g^{cd,*=*}_{r_{gran}}(u))) = d(v)\},$$

where

$$IND^{*=don't\ care}_{a_j}(MV(g^{cd,*=*}_{r_{gran}}(u)), v) =$$

$$\{a \in A : (a(MV(g^{cd,*=*}_{r_{gran}}(u))) = a(v)\ OR\ a(MV(g^{cd,*=*}_{r_{gran}}(u))) = *\ OR\ a(v) = *)\ AND\ a_j(v)! = *\}.$$

In case of $* = *$ in repairing phase

As above, also in case of B and D strategies, in order to repair objects containing missing values after granulation, we immerse objects with stars on specific positions j into original disturbed training data set. We fill the star based on majority voting from non missing values of attribute number j.
In case of the strategy B, the granule around the disturbed object $MV(g^{cd,*=don't\ care}_{r_{gran}}(u))$ can be defined as follows,

$$g^{cd,*=*}_{r_{gran},a_j}(MV(g^{cd,*=don't\ care}_{r_{gran}}(u))) =$$

$$\{v \in TRN_i : \frac{|IND^{*=*}_{a_j}(MV(g^{cd,*=don't\ care}_{r_{gran}}(u)), v)|}{|A|} \leq r_{gran}\ AND\ d(MV(g^{cd,*=don't\ care}_{r_{gran}}(u))) = d(v)\},$$

where

$$IND^{*=*}_{a_j}(MV(g^{cd,*=don't\ care}_{r_{gran}}(u)), v) = \{a \in A : a(MV(g^{cd,*=don't\ care}_{r_{gran}}(u))) = a(v)\ AND\ a_j(v)! = *\}.$$

In case of the strategy D, the granule around the disturbed object $MV(g^{cd,*=*}_{r_{gran}}(u))$ can be defined as follows,

$$g^{cd,*=*}_{r_{gran},a_j}(MV(g^{cd,*=*}_{r_{gran}}(u))) =$$

$$\{v \in TRN_i : \frac{|IND^{*=*}_{a_j}(MV(g^{cd,*=*}_{r_{gran}}(u)), v)|}{|A|} \leq r_{gran}\ AND\ d(MV(g^{cd,*=*}_{r_{gran}}(u))) = d(v)\},$$

where

$$IND^{*=*}_{a_j}(MV(g^{cd,*=*}_{r_{gran}}(u)), v) = \{a \in A : a(MV(g^{cd,*=*}_{r_{gran}}(u))) = a(v)\ AND\ a_j(v)! = *\}.$$

6.1. Homogeneous granulation in $* = *$ and $* = don't\ care$ cases

Considering previously defined $IND^{*=don't\ care}(u, v)$, in case of $* = don't\ care$ the granules are formed as follows,

$$g_{r_u}^{homogenous,*=don't\ care} = \{v \in U : |g_{r_u}^{cd,*=don't\ care}| - |g_{r_u}^{*=don't\ care}| == 0,$$

$$for\ minimal\ r_u\ fulfills\ the\ equation\}$$

where

$$g_{r_u}^{cd,*=don't\ care} = \{v \in U : \frac{IND^{*=don't\ care}(u, v)}{|A|} \leq r_u\ AND\ d(u) == d(v)\}$$

and

$$g_{r_u}^{*=don't\ care} = \{v \in U : \frac{IND^{*=don't\ care}(u, v)}{|A|} \leq r_u\}$$

$$r_u = \{\frac{0}{|A|}, \frac{1}{|A|}, ..., \frac{|A|}{|A|}\}$$

for $* = *$ variant and previously defined $IND^{*=*}(u, v)$ we have:

$$g_{r_u}^{homogenous,*=*} = \{v \in U : |g_{r_u}^{cd,*=*}| - |g_{r_u}^{*=*}| == 0,$$

$$for\ minimal\ r_u\ fulfills\ the\ equation\}$$

where

$$g_{r_u}^{cd,*=*} = \{v \in U : \frac{IND^{*=*}(u, v)}{|A|} \leq r_u\ AND\ d(u) == d(v)\}$$

and

$$g_{r_u}^{*=*} = \{v \in U : \frac{IND^{*=*}(u, v)}{|A|} \leq r_u\}$$

$$r_u = \{\frac{0}{|A|}, \frac{1}{|A|}, ..., \frac{|A|}{|A|}\}$$

6.2. The experimental session - procedures and model settings

In the section we have described the experimental part with results presentation. We have checked the effectiveness of out techniques on artificially damaged (filled with 10 percent of missing values) selected data from UCI Repository [16].

6.2.1. Pseudo-code of experiments design

(i) We uploaded selected data set,
(ii) Data was split according to Cross Validation 5 model,
(iii) Training decision systems $TRN_i^{complete}$ are granulated with use of the selected method,
(iv) The TST_i are classified using set $TRN_i^{complete}$ by kNN classifier (its the nil result),
(v) $TRN_i^{complete}$ is filled with ten percent of randomly located stars,
(vi) TRN_i is treated by selected missing values handling strategy - A, B, C or D in granulation process.
(vii) The TST_i systems are classified by repaired granular systems using kNN classifier,
(viii) The final result is an average from all five tests.
The above CV5 procedure is repeated 5 times, and our result is the average value from all tests.

6.2.2. The results evaluation

To evaluate our results we have proposed to compute bias of accuracy from 5 $\times$ CV-5, based on the formula,

$$AccBias = \frac{\sum_{i=1}^{5}(max(acc_1^{CV5}, acc_2^{CV5}, ..., acc_5^{CV5}) - acc_i^{CV5})}{5}, \tag{6.1}$$

where

$$Acc = \frac{\sum_{i=1}^{5} acc_i^{CV5}}{5}.$$

As a reference classifier we use kNN in decision classes, where a class is winning if the summary distance of k-nearest objects from the class is the smallest.
Parameter k is estimated on the sample of data based on Cross Validation five method.
We use $k = 5$ for Australian Credit data set and $k = 3$ for Pima Indians Diabetes.

6.2.3. The results discussion

The results for classic parameterized concept-dependent granulation are in Tabs. 6.1 and 6.2. For homogeneous granulation are in Tabs. 6.3 and 6.4. As we

previously checked in [30], granulation is effective in missing values absorption. For all examined techniques the quality of classification is preserved on damaged data in comparison with original one - without missing values. In case of granulation techniques we have additional reduction in object numbers, even up to 80 percent of original training data size for concept dependent method. Seeing the results in [30] - the effectiveness of methods and their behavior depends strictly on the type of data set. For instance for typical data sets with high diversity of attribute values, like Australian Credit, Pima Indians Diabetes, the result is predictable. And in case of A and B strategies approximation is faster for lower values of granulation radii, its because for $* = don't\ care$ the granules contain more objects. In case of $* = *$, the approximation is similar to the nil result, but is slightly slower, because the stars could increase diversity of the data, and the granules could contain a smaller number of objects which in consequence gives a larger number of granules in coverings.

The interpretation of missing values absorption for homogeneous granulation is significantly different - see Tab 6.3 and 6.4. In case of this technique, damage of the data increase the number of granules in coverings, the granules became smaller because the indiscernibility level in decision classes is lowering. It is higher probability to find the objects, which brake homogeneity of granules. In case of A and B strategies, the granules are smaller than for C and D, thus in the first one we have bigger granular decision systems

The methods work in a stable way and results are fully comparable with nil results. The most advantage of homogeneous granulation is its single run behaviour, where granulation radius is fixed individually (automatically) for each granule - depends on the indiscernibility level in each decision class around central objects of granules.

Table 6.1: Missing values absorption based on **Concept dependent granulation**; 5 x CV-5; A,B,C,D strategies vs complete data classification; $\mathbf{Australian\ Credit}$; ; $\mathbf{10\ percent\ of\ missing\ values}$; r_{gran} = Granulation radius; nil = result for data without missing values; Acc = Accuracy of classification; $AccBias$ = Accuracy bias defined based on equation efAcccBiasEquation; $GranSize$ = The size of data set after granulation in the fixed r

	Acc					AccBias				
r_{gran}	nil	A	B	C	D	nil	A	B	C	D
0	0.772	0.77	0.77	0.77	0.77	0.009	0.006	0.006	0.006	0.006
0.0714286	0.772	0.77	0.77	0.772	0.772	0.01	0.006	0.006	0.008	0.008
0.142857	0.77	0.77	0.771	0.773	0.773	0.006	0.006	0.007	0.011	0.011
0.214286	0.781	0.766	0.767	0.786	0.785	0.008	0.01	0.012	0.02	0.018
0.285714	0.799	0.775	0.777	0.811	0.81	0.014	0.012	0.007	0.015	0.009
0.357143	0.82	0.786	0.786	0.826	0.832	0.01	0.014	0.014	0.015	0.004
0.428571	0.841	0.806	0.8	0.838	0.838	0.007	0.032	0.012	0.009	0.002
0.5	0.838	0.817	0.818	0.84	0.847	0.005	0.012	0.012	0.008	0.004
0.571429	0.839	0.828	0.826	0.847	0.844	0.006	0.019	0.021	0.007	0.01
0.642857	0.848	0.832	0.826	0.847	0.839	0.007	0.007	0.017	0.007	0.008
0.714286	0.853	0.833	0.841	0.844	0.843	0.009	0.019	0.007	0.011	0.012
0.785714	0.857	0.843	0.843	0.847	0.843	0.007	0.01	0.012	0.008	0.014
0.857143	0.86	0.838	0.838	0.845	0.844	0.007	0.01	0.014	0.01	0.008
0.928571	0.862	0.842	0.841	0.844	0.843	0.005	0.005	0.017	0.014	0.013
1	0.861	0.843	0.843	0.843	0.843	0.004	0.014	0.013	0.014	0.014

	GranSize				
r_{gran}	nil	A	B	C	D
0	2	2	2	2	2
0.0714286	2.32	2	2	3	2.96
0.142857	3.24	2.16	2.16	4.64	4.68
0.214286	5.16	2.52	2.52	8.68	8.4
0.285714	8.4	4.04	3.84	16.2	16.32
0.357143	16.08	7.12	6.96	32.44	31.92
0.428571	32	10.08	9.76	72.04	72.24
0.5	70.8	18.28	18	150.04	149.6
0.571429	156.6	34.6	34.72	286.24	284.8
0.642857	318.12	73.44	73.32	438.08	438.28
0.714286	467.6	164.2	164.44	524.64	525.08
0.785714	536.12	325.92	328.04	547	546.96
0.857143	547.16	476.76	476.76	551.28	551.28
0.928571	548.84	537.8	537.36	551.88	551.88
1	552	550.84	550.8	552	552

Table 6.2: Missing values absorption based on **Concept dependent granulation**; 5 x CV-5; A,B,C,D strategies vs complete data classification; $\mathbf{Pima\ Indians\ Diabetes}$; Concept dependent granulation; $\mathbf{10\ percent\ of\ missing\ values}$; $r_{gran} =$ Granulation radius; nil = result for data without missing values; Acc = Accuracy of classification; $AccBias$ = Accuracy bias defined based on equation efAcccBiasEquation; $GranSize$ = The size of data set after granulation in the fixed r

	Acc					AccBias				
r_{gran}	nil	A	B	C	D	nil	A	B	C	D
0	0.605	0.609	0.609	0.609	0.609	0.009	0.012	0.012	0.012	0.012
0.125	0.608	0.615	0.61	0.609	0.617	0.006	0.009	0.027	0.011	0.019
0.25	0.632	0.624	0.61	0.634	0.62	0.013	0.013	0.015	0.018	0.024
0.375	0.639	0.6	0.602	0.636	0.641	0.009	0.018	0.017	0.02	0.015
0.5	0.649	0.602	0.618	0.647	0.648	0.017	0.02	0.021	0.018	0.02
0.625	0.647	0.614	0.61	0.645	0.646	0.009	0.013	0.026	0.019	0.019
0.75	0.648	0.637	0.639	0.647	0.647	0.009	0.012	0.013	0.029	0.023
0.875	0.648	0.639	0.645	0.65	0.647	0.009	0.015	0.017	0.021	0.023
1	0.648	0.647	0.647	0.647	0.647	0.009	0.023	0.023	0.023	0.023

	GranSize				
r_{gran}	nil	A	B	C	D
0	2	2	2	2	2
0.125	35.2	3.16	3.2	33.16	31.68
0.25	155.88	8.96	8.8	145.96	145.44
0.375	365.52	29.04	26.72	364.84	363.6
0.5	540.28	87	84.24	546.72	546.48
0.625	609.72	282.04	282	609.24	609.16
0.75	614.4	491.2	488.04	614.24	614.24
0.875	614.4	593.64	593.6	614.4	614.4
1	614.4	613.64	613.6	614.4	614.4

Table 6.3: Missing values absorption based on **Homogeneous granulation**; 5 x CV-5; A,B,C,D strategies vs complete data classification; $\mathbf{Australian\ Credit}$; Homogeneous granulation; $\mathbf{10\ percent\ of\ missing\ values}$; $r_{gran} =$ Granulation radius; nil = result for data without missing values; Acc = Accuracy of classification; $AccBias$ = Accuracy bias defined based on equation efAcccBiasEquation; $GranSize$ = The size of data set after granulation in the fixed r

Acc					AccBias				
nil	A	B	C	D	nil	A	B	C	D
0.843	0.841	0.843	0.838	0.841	0.012	0.008	0.015	0.021	0.014

GranSize				
nil	A	B	C	D
283.64	426.4	424.16	311.48	313.08

Table 6.4: Missing values absorption based on **Homogeneous granulation**; 5 x CV-5; A,B,C,D strategies vs complete data classification; $\mathbf{Pima\ Indians\ Diabetes}$; Homogeneous granulation; $\mathbf{10\ percent\ of\ missing\ values}$; $r_{gran} =$ Granulation radius; nil = result for data without missing values; Acc = Accuracy of classification; $AccBias$ = Accuracy bias defined based on equation efAcccBiasEquation; $GranSize$ = The size of data set after granulation in the fixed r

	Acc				AccBias				
nil	A	B	C	D	nil	A	B	C	D
0.646	0.644	0.646	0.636	0.642	0.026	0.015	0.015	0.02	0.021

	GranSize			
nil	A	B	C	D
490.88	578	577.12	490	492.12

6.3. Section summary

The performance of homogeneous granulation in missing values absorption, compared to our other granulation techniques is different. Using the variant $* = don'tcare$ - in the concept-dependent method the granulation process reduces the diversity of the data, for $* = *$ the diversity can be increased. The granularity is smaller for strategies C and D than for A and B. Ultimately, the size of the granular reflections of the training systems is smaller for strategies A and B - the level of approximation is higher. In addition, the reduction in the size of training decision systems - in the case of granularity of corrupted systems - is significant compared to granularity of intact datasets. Missing values increase the level of approximation in many cases. The absorption of missing values for the homogeneous variant, in which the granulation radius increases dynamically until the objects in the granules belong only to the central object class, behave differently. The number of objects in the homogeneous granulation process increases compared to the null case - where the granulation process is performed on intact data. For the A and B strategies, the granules are smaller than for C and D - this is because $* = don'tcare$ breaks the homogeneity of the decision classes at a higher level than in the $* = *$ case.The level of approximation decreases in the case of corrupted datasets, as class homogeneity is affected in all variants. As we intuitively predicted, the homogeneous granulation

process effectively absorbs the missing values and the completed data retain their effectiveness in terms of classification process.

Part IV

Study of the influence of over and undersampling techniques on the quality of the granulation process

7. Impact of oversampling and undersampling on data granulation.

7.1. Introduction

Unbalanced data are those in which the number of objects in each class varies. The degree of imbalance can vary, and so a set with 120 black balls and 80 white balls will be called slightly imbalanced, and a set with 190 black balls and 10 white balls can be called strongly or even extremely imbalanced.
The imbalance in the balance of the set has a direct impact on the quality of most classifier machine learning models hence there is a need to solve such problems. It is also worth mentioning here the necessity of correct interpretation of classification measures if model training is carried out on significantly unbalanced data, where at least accuracy alone will not reflect the true effectiveness of the model when there is a significant discrepancy in the number of objects in each class.
Oversampling and undersampling are one way of dealing with unbalanced data sets, especially in the context of classification problems.

Oversampling

Oversampling involves generating new observations in classes of smaller size, e.g., through random drawing with return or more complex mechanisms that add observations with random noise.
One of the more popular oversampling mechanisms is the SMOTE (Synthetic Minority Over-sampling Technique) algorithm [7]. It was chosen as the oversampling algorithm in the experimental session of this work.
The operation of the SMOTE algorithm is to select, using the KNN algorithm, the k nearest neighbors and, through interpolation, generate new observations "on the lines" that connect these points, that is, in the feature space of these neighbors.

A visualization of the performance of the SMOTE algorithm for the selected attribute of the pima set is shown below. In the graph on the left, you can see the points in blue represent the values present in the set for the minority class attribute, and the orange color is the new points generated by the SMOTE algorithm. They are arranged in the order of their occurrence in the set (y-axis) so that they can be observed without overlapping with existing observations. In the graph on the right, these new observations are superimposed on the graph of existing ones, which illustrates the final set of values of this attribute for the minority class.

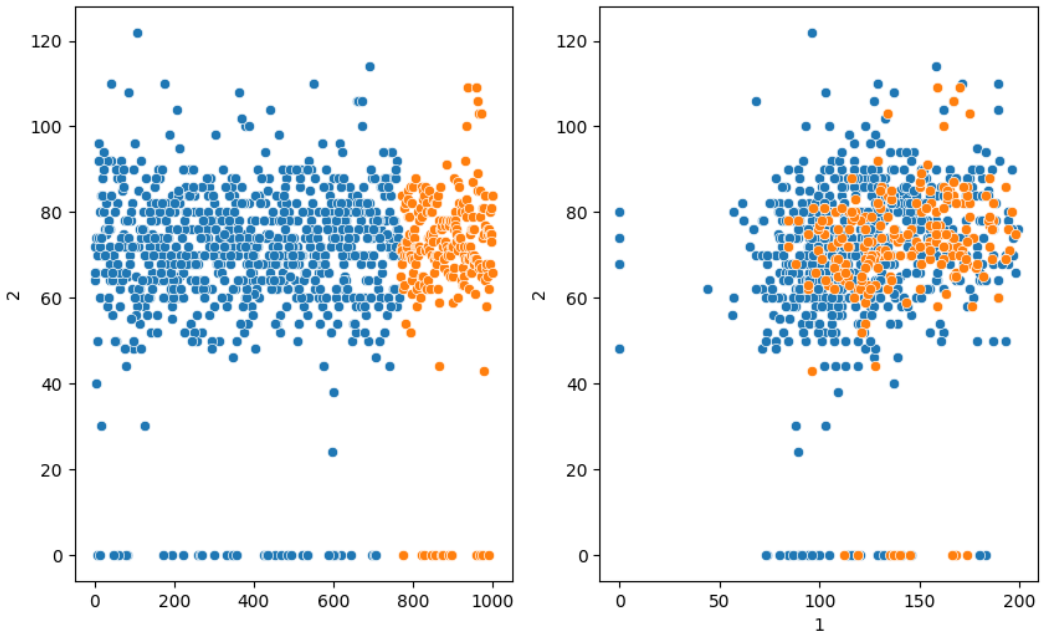


Figure 7.1: Visualization of the effects of the SMOTE algorithm on a selected attribute of the pima set.

Undersampling

Undersampling is a mechanism for removing objects from dominant classes until data balance is achieved.

Among the algorithms, we can distinguish those that indicate objects that, from the point of view of the information they carry, are worth keeping, and those that indicate objects to be removed. The first two below (Near Miss and CNN) are those that indicate objects to keep, and the last one (Tom Links) indicates objects to remove.

Near Miss is a collection of several slightly different oversampling algorithms proposed by Jianping Zhang in Inderjeet Mani in [48], whose main operation is to select objects using the distance between majority and minority classes.
In the framework of this algorithm, we distinguish three methods:

— NearMiss-1: observations from the majority class with the smallest average distance to the three nearest minority class objects are selected,
— NearMiss-2: observations from the majority class with the smallest average distance to the three most distant minority class objects are selected,
— NearMiss-3: observations from the majority class with the shortest distance to each minority class object are selected.

Another undersampling algorithm is **Condensed Nearest Neighbors** (with a rather confusing acronym today - CNN) proposed by Peter Hart in [12]. The principle of this algorithm is to build subsets using a random selection of an object, then selecting k of its nearest neighbors and checking whether the selected object is classified the same as with the original set. If not, the object is a candidate for removal. This approach minimizes the removal of objects relevant to the information carried from the set.
Another undersampling algorithm is the one called **Tomek Links** first proposed by Ivan Tomek in [43] as a proposal to modify the Condensed Nearest Neighbors algorithm. In the CNN algorithm, the points around which neighbors are then searched for a subset are chosen at random. Ivan Tomek, in proposal two, presented an approach, for a binary problem, to find two points of different classes with the smallest Euclidean distance. The undoubted advantage of this approach is the possibility of determining the boundaries between classes or finding noise, since only such objects will have opposite classes among their neighbors. The Tomek links algorithm was also chosen for the experimental part of this chapter.

7.2. Methodology

Sets from the 2.19 table were used for experiments, but only those with two decision classes. These are:

— australian credit

— australian credit dummy data (called also australian dummy)
— heart
— pima
— breast
— mushroom
— adult

Initially, the same set of classifiers was chosen as presented in the 2.4.7, but after analyzing the results from previous experimental sessions for MLP and SVM classifiers, it was decided to drop them from the final experiments in this part of the dissertation.
Below is a flow diagram of experiments examining the effects of over and undersampling on classification results for granular data.

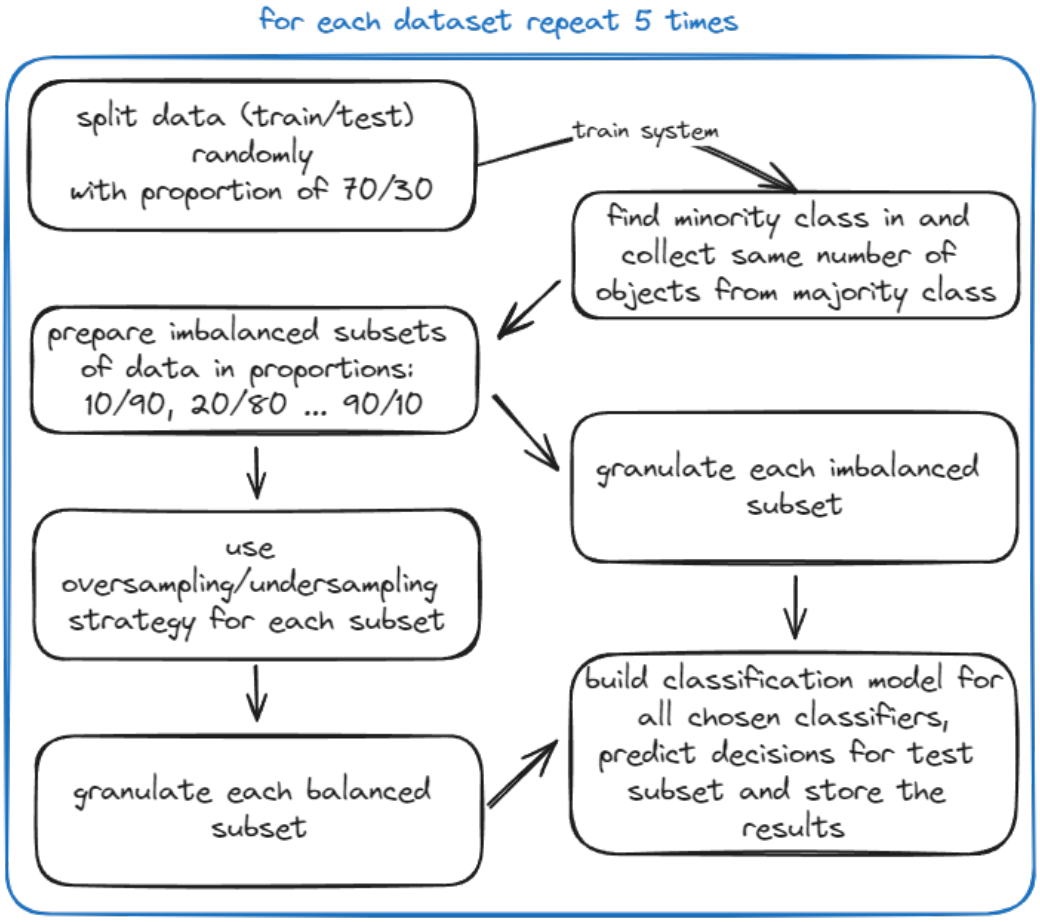


Figure 7.2: Pipeline for over and undersampling experiment pipeline.

A more detailed description of the above scheme is provided below.

Step 1.

For each dataset, the input data is divided into a training dataset and a test dataset in the ratio of 70%/30%. This and subsequent steps are repeated 5 times for each dataset.

Step 2.

The minority class in the training set is searched. Then the same number of observations as in the minority class are randomly selected from the majority class. This forms our target training set.

Step 3.

The training data is subjected to a split of 10%/90% to 90%/10% for class c1 and c2, respectively.

Step 4.

Each unbalanced training set is used as learning data for the selected classifiers, and then evaluated with a test set - this is the result denoted as imbalanced nil case. Each of these imbalanced sets is also subjected to granulation, concept dependent or homogeneous (in two separate experiments), and this case is marked as imbalanced gran case ($imbalanced\_hom\_gran$ for homogeneous granulation and $imbalanced\_cdgran\_radius$ for concept dependent granulation, separately for two selected radii of this granulation - 1.0 and 0.5).

Step 5.

Each imbalanced collection is subjected to:

- **in scenario 1:**
  - balanced with the SMOTE algorithm, then training the model for each classifier and finally testing with a test set. This case is denoted as $balanced\_nil$.
  - then each balanced set resulting from the balancing (previous point in the list) is also subjected to granulation, and each of the selected classifiers is trained on this data and finally evaluated on the test data. This case is marked in the results as $balanced\ gran$ (annotated with hom_gran or $cdgran$ for the given granulation method).
- **in scenario 2:**
  - balanced with the SMOTE algorithm, and then using the Tomek algorithm, the so-called Tomek Links are removed, a selected classifier is trained on the resulting set and evaluated with a test set. This case is denoted as

$balanced\_nil$ (the three scenarios are carried out independently, hence the same name in each scenarios results).

- each balanced collection is also subjected to granulation, and each of the selected classifiers is trained on this data and finally evaluated on the test data. This case is marked in the results as $balanced\ gran$ (annotated with hom_gran or $cdgran$ for the given granulation method).

- **in scenario 3:**
  - first Tomek Links undersampling is applied, and then the training set is balanced with the SMOTE algorithm. Each classifier is trained on the resulting set and evaluated with a test set. This case is denoted as $balanced\_nil$.
  - each balanced collection is also subjected to granulation, and each of the selected classifiers is trained on this data and finally evaluated on the test data. This case is marked in the results as $balanced\ gran$ (annotated with hom_gran or $cdgran$ for the given granulation method).

7.3. Experiments and results

7.3.1. Scenario 1

Below are graphs with classification results for the balanced accuracy metric for each dataset and classifiers with data series for each case described above (see legend of graphs). On the vertical axis is the value of the balanced accuracy metric, and on the horizontal axis is the distribution of data for which this averaged value was achieved by the test set for the five train/test permutation.

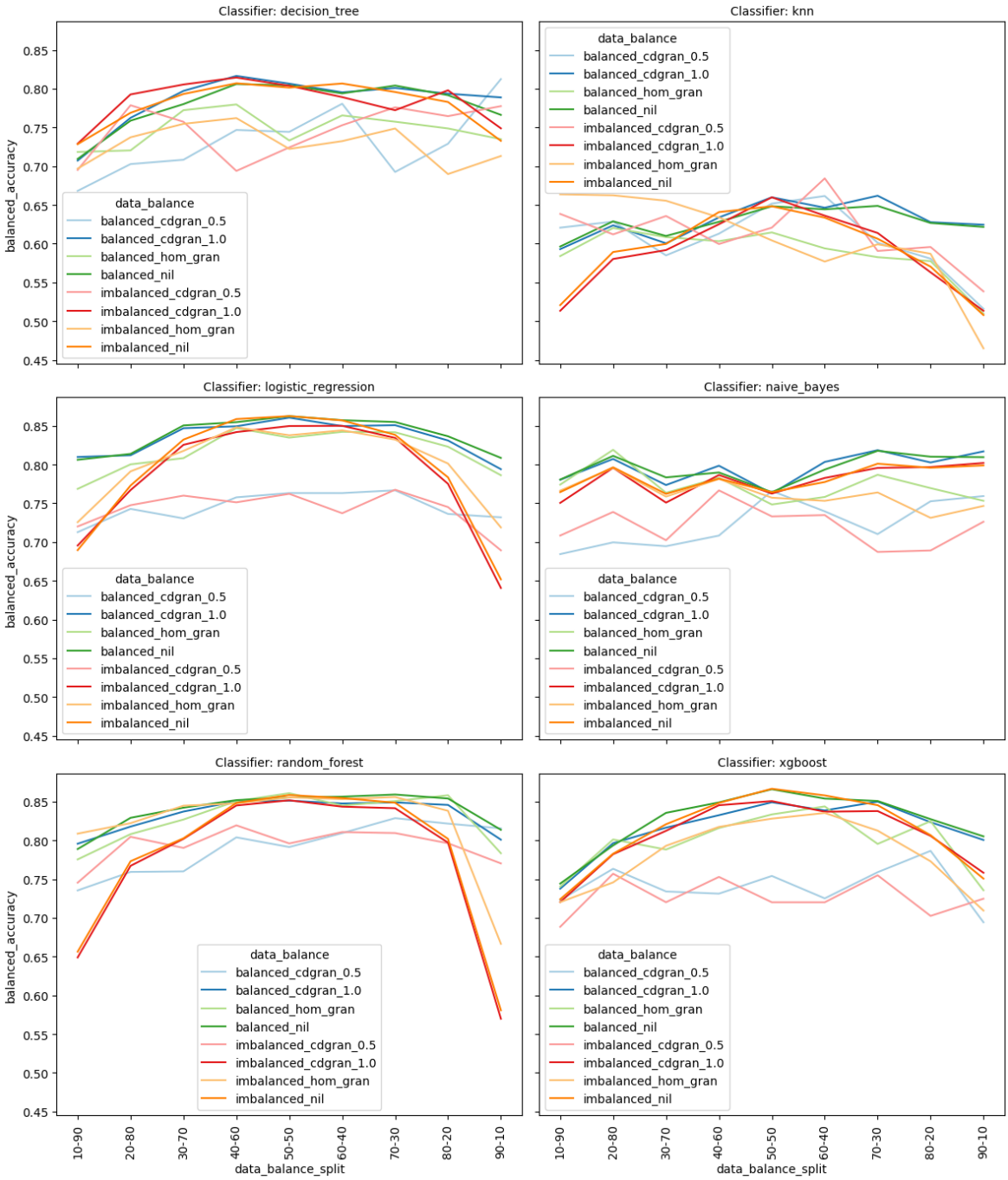


Figure 7.3: Balanced accuracy plot for SMOTE oversampling experiments for the australian dataset.

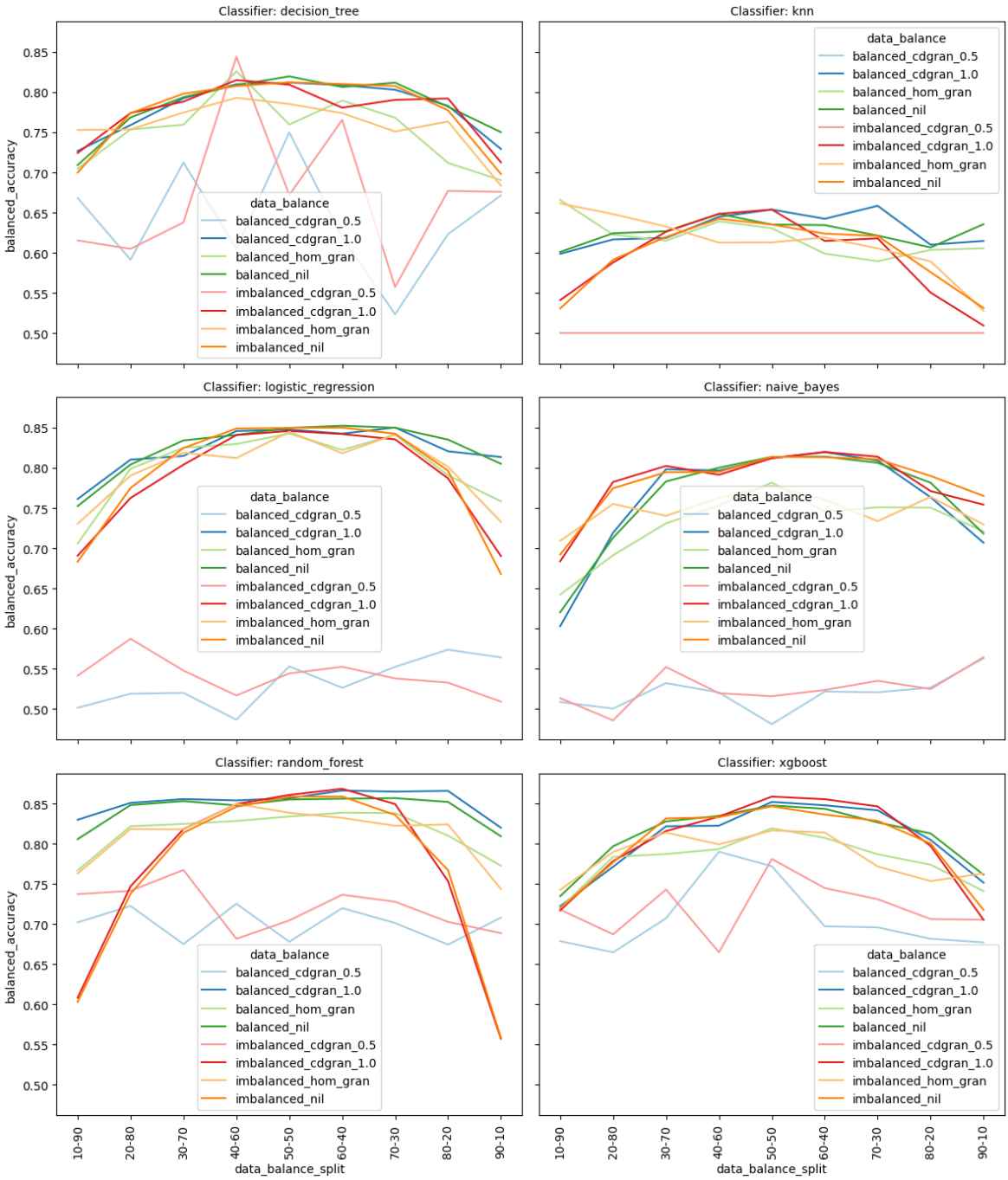


Figure 7.4: Balanced accuracy plot for SMOTE oversampling experiments for the australian dummy dataset.

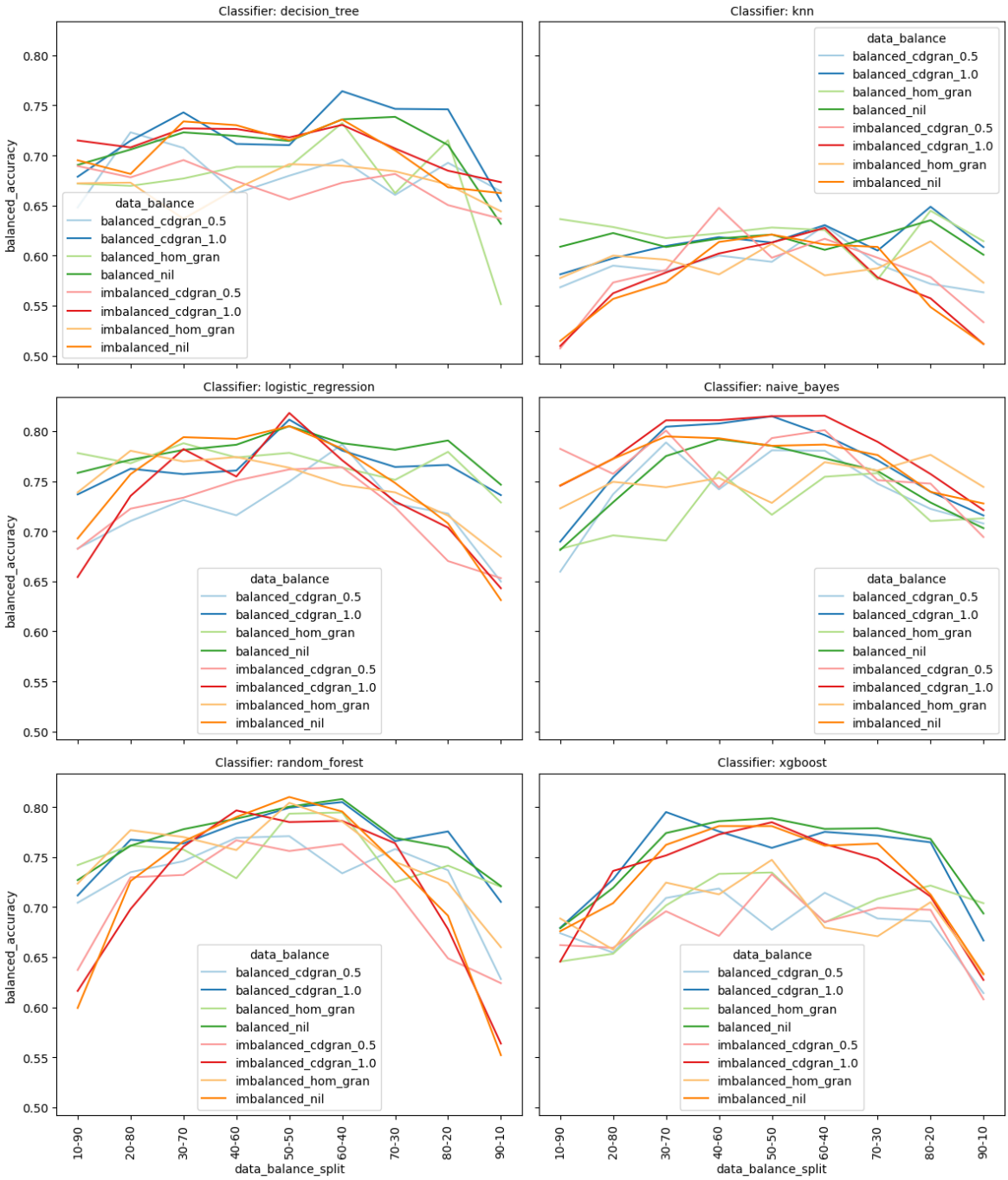


Figure 7.5: Balanced accuracy plot for SMOTE oversampling experiments for the heart dataset.

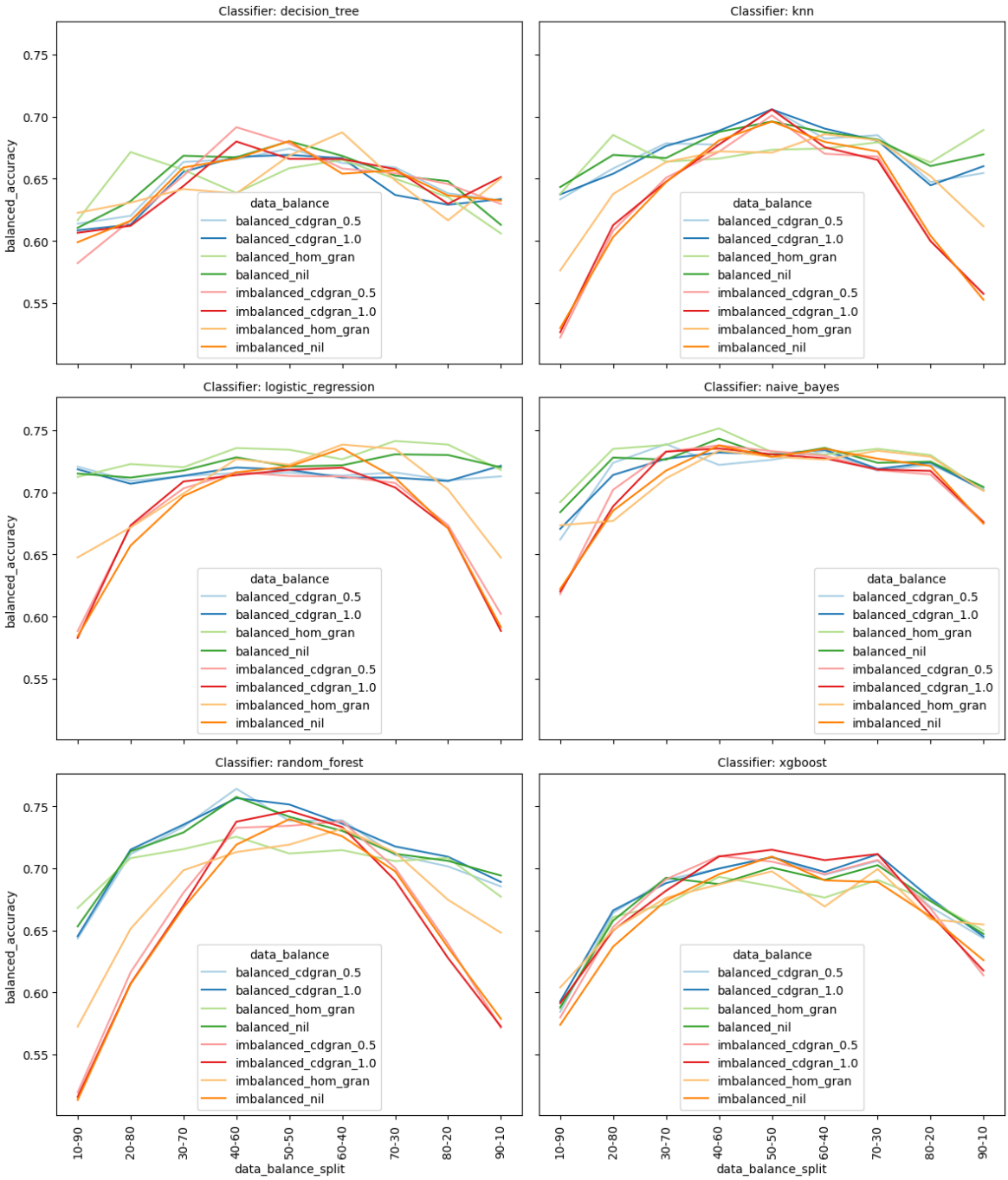


Figure 7.6: Balanced accuracy plot for SMOTE oversampling experiments for the pima dataset.

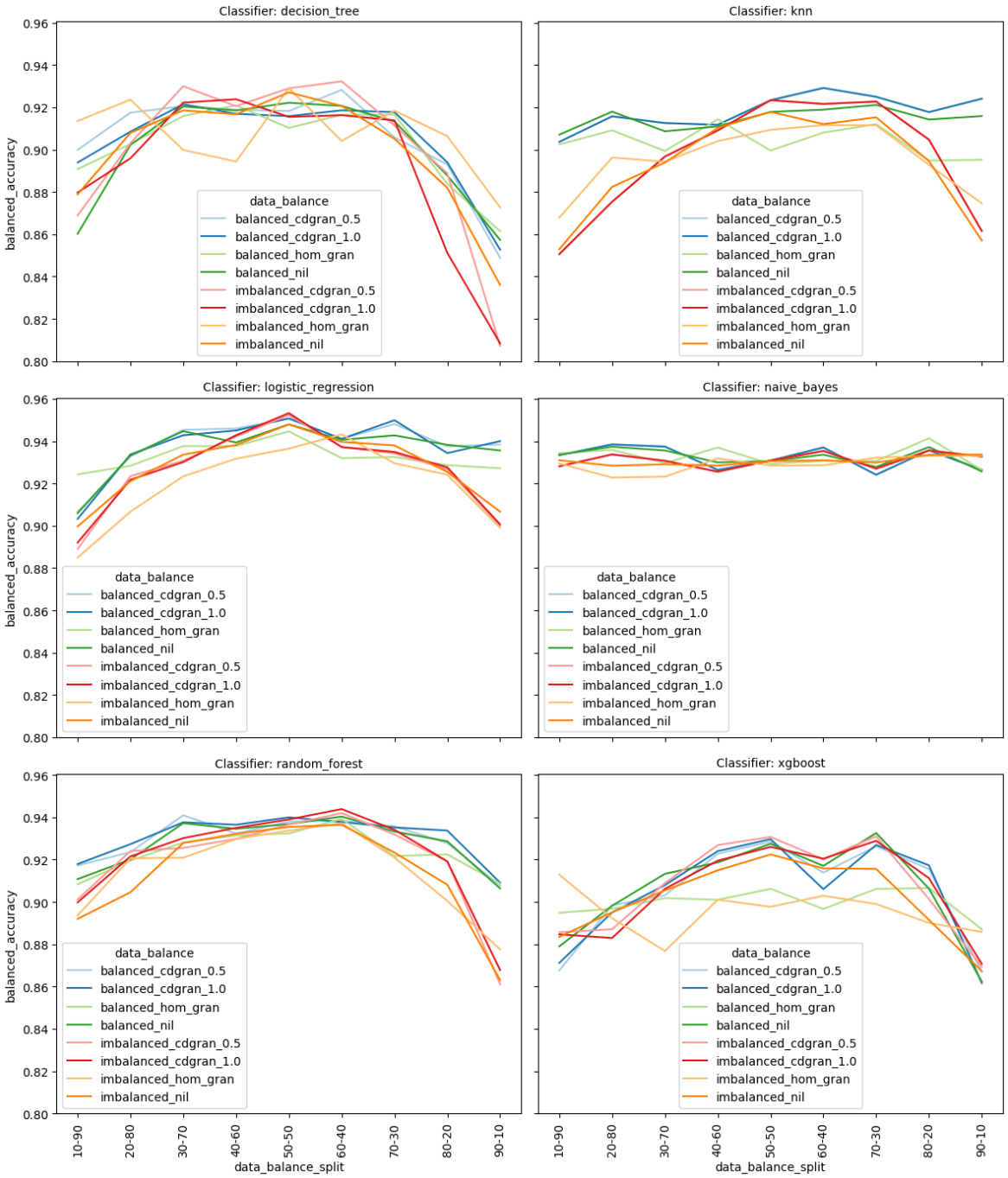


Figure 7.7: Balanced accuracy plot for SMOTE oversampling experiments for the breast dataset.

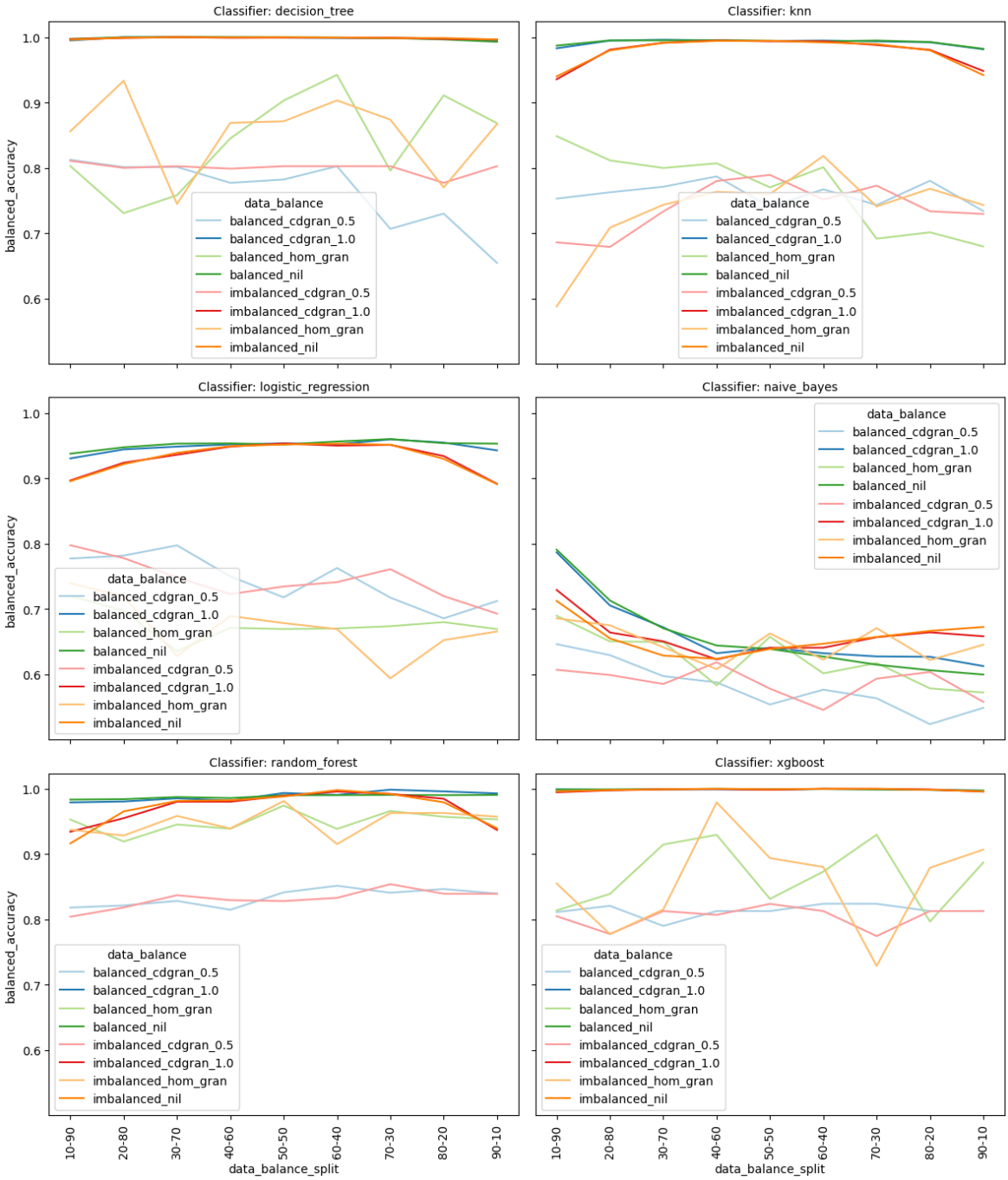


Figure 7.8: Balanced accuracy plot for SMOTE oversampling experiments for the mushroom dataset.

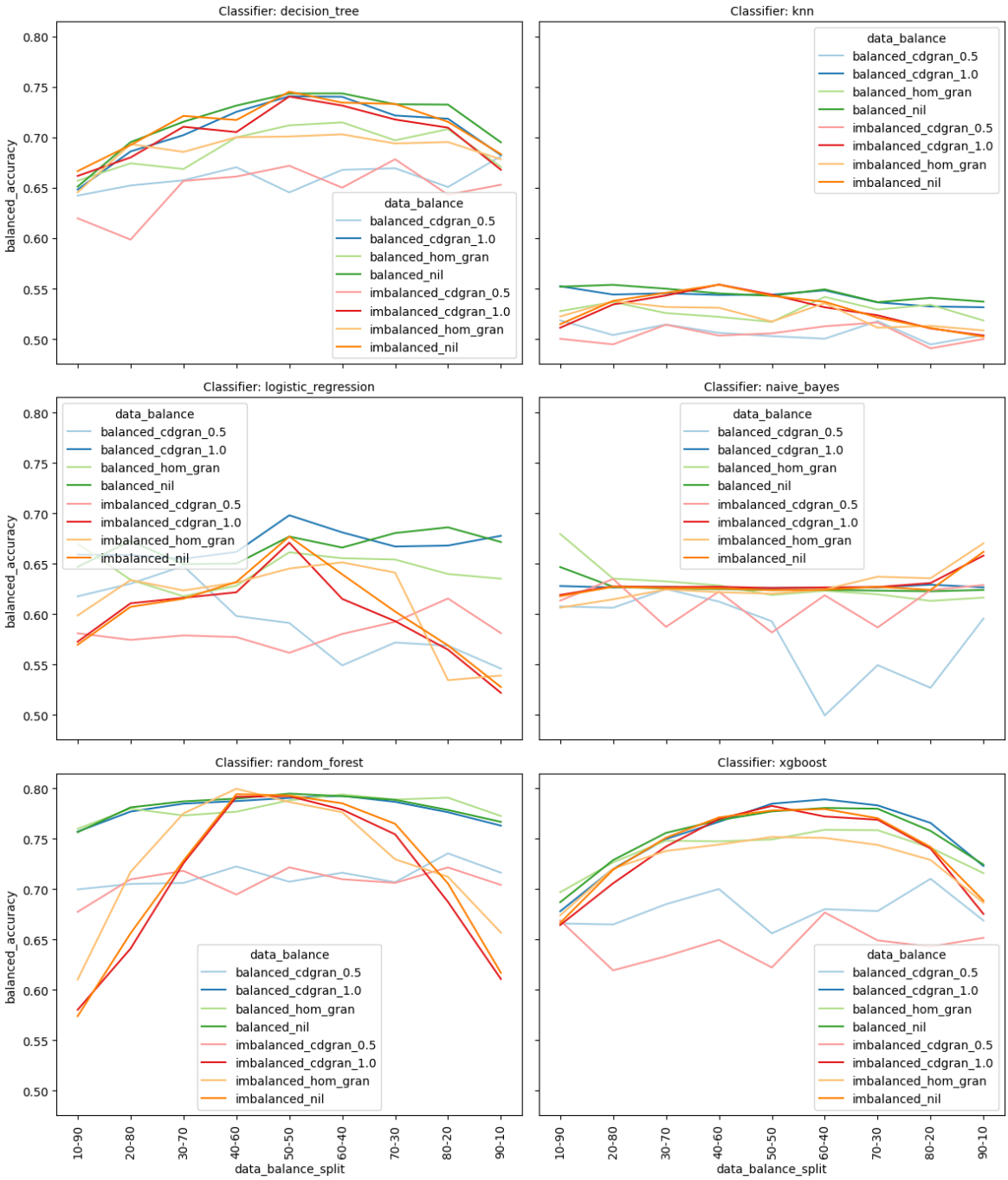


Figure 7.9: Balanced accuracy plot for SMOTE oversampling experiments for the adult dataset.

From observing the charts themselves, we can already draw several conclusions.

— The lowest results are usually achieved for training sets resulting from concept dependent granulation with a radius of 0.5, with no clear advantage for the imbalanced and imbalanced case. The reason for this is usually the rather large approximation of the sets at this radius of granulation, which in effect reduces the size of the feature domain and oversampling does not improve the results here. It can also be noted that changing the $data\_balance\_split$ feature does not change the value of the metric as it usually does for the other cases. Here the trend for the results is rather sideways. In the other cases, we can usually see that they converge to a normal distribution. The lateral trend is due to the fact that the sets granulated within a radius of 0.5 can be very small, and over and undersampling do not change their size to any significant degree. Small sets can also cause a slightly larger bias in classification results for decision tree-based classifiers due to the random selection of objects to build them (via bootstrapping) which can result in the selection of a very small number of objects of a given class.
— Not surprisingly, on average, the highest classification metrics are typically achieved for nil-case balanced data and granularity balanced concept dependent data with a radius of 1.0. This also coincides with the largest number of observations in these sets. The advantage in balanced accuracy for these two mentioned cases can be seen especially with the largest base imbalance of the training sets i.e. extreme values of $data\_balance\_split$ such as 10/90, 20/80 or 80/20 and 90/10.
— We can usually see that the best results are achieved for an initial class balance close to or equal to 50/50 which is common knowledge in supervised learning issues. This is also the parity one tries to aim for when preprocessing and preparing data for models, although there are techniques for tuning models that are trained on strongly or extremely imbalanced data (e.g., class weighting). Here these techniques have not been applied. Given this fact, we realize that the case of a 50/50 split of classes is a rather special case, since the application of the SMOTE algorithm on an imbalanced set does not change its objects to any extent. This means that the imbalanced nil and balanced nil sets for this split

are usually identical, and this can be seen in the graph where the balanced accuracy value converges for these two cases for the 50/50 split.

The following table summarizes all the cases for the way the training set is prepared with the averaged balanced accuracy value. For an explanation of how each set was created, see smote-pipeline-description.

Table 7.1: Training dataset ranking for scenario 1.

data balance	balanced accuracy mean
balanced_nil	0.7872
balanced_cdgran_1.0	0.7866
imbalanced_nil	0.7664
imbalanced_cdgran_1.0	0.7658
balanced_hom_gran	0.7535
imbalanced_hom_gran	0.7464
balanced_cdgran_0.5	0.7136
imbalanced_cdgran_0.5	0.7098

Below is a table showing the average balanced accuracy score for each level of dataset division and for each classifier separately.

Table 7.2: Classifiers ranking - mean balanced accuracy for each tested case and scenario 1.

data_balance_split	classifier	balanced accuracy mean
90-10	xgboost	0.750
90-10	random_forest	0.748
90-10	decision_tree	0.735
90-10	logistic_regression	0.725
90-10	naive_bayes	0.721
90-10	knn	0.646
80-20	random_forest	0.801
80-20	xgboost	0.785
80-20	logistic_regression	0.759
80-20	decision_tree	0.758
80-20	naive_bayes	0.729
80-20	knn	0.668
70-30	random_forest	0.820
70-30	xgboost	0.801
70-30	logistic_regression	0.778
70-30	decision_tree	0.766

Continued on next page

data_balance_split	classifier	balanced accuracy mean
70-30	naive_bayes	0.736
70-30	knn	0.683
60-40	random_forest	0.832
60-40	xgboost	0.803
60-40	decision_tree	0.785
60-40	logistic_regression	0.783
60-40	naive_bayes	0.738
60-40	knn	0.695
50-50	random_forest	0.833
50-50	xgboost	0.811
50-50	logistic_regression	0.790
50-50	decision_tree	0.781
50-50	naive_bayes	0.737
50-50	knn	0.696
40-60	random_forest	0.828
40-60	xgboost	0.807
40-60	logistic_regression	0.780
40-60	decision_tree	0.780
40-60	naive_bayes	0.739
40-60	knn	0.693
30-70	random_forest	0.814
30-70	xgboost	0.793
30-70	logistic_regression	0.774
30-70	decision_tree	0.769
30-70	naive_bayes	0.737
30-70	knn	0.682
20-80	random_forest	0.794
20-80	xgboost	0.767
20-80	logistic_regression	0.764
20-80	decision_tree	0.755

Continued on next page

data_balance_split	classifier	balanced accuracy mean
20-80	naive_bayes	0.733
20-80	knn	0.676
10-90	random_forest	0.749
10-90	xgboost	0.738
10-90	decision_tree	0.734
10-90	logistic_regression	0.734
10-90	naive_bayes	0.715
10-90	knn	0.652

As we can observe from the above table, the highest value of the balanced accuracy metric is most often achieved by the random forest algorithm (8 out of 9 splits). Xgboost is the second most effective algorithm in this list where it settles for the highest measure once and occurs 8 times in second place. The next place alternates between decision tree or logistic regression. We can also quickly deduce from the above table that as the difference in class distribution increases, the effectiveness of classification decreases.

Looking for differences in the results of individual subsets, we can still point to the balanced accuracy results for homogeneous granulation, the scatter of which is the smallest among the tested sets, which we can find confirmation in the following summary in the form of the lowest value of the standard deviation.

Table 7.3: Distribution statistics for each training subset for scenario 1.

data_balance	balanced_accuracy					
	std	min	25%	50%	75%	max
balanced_cdgran_0.5	0.1187	0.4811	0.6466	0.7095	0.7796	0.9507
balanced_cdgran_1.0	0.1218	0.5315	0.6920	0.7829	0.8940	1.0000
balanced_hom_gran	0.1073	0.5096	0.6734	0.7463	0.8266	0.9743
balanced_nil	0.1213	0.5365	0.6937	0.7816	0.8856	1.0000
imbalanced_cdgran_0.5	0.1181	0.4859	0.6222	0.7053	0.7784	0.9523
imbalanced_cdgran_1.0	0.1330	0.5034	0.6620	0.7625	0.8682	1.0000
imbalanced_hom_gran	0.1097	0.4650	0.6673	0.7387	0.8223	0.9813
imbalanced_nil	0.1325	0.5023	0.6619	0.7663	0.8671	1.0000

We can also see what the average balanced accuracy value looks like along with the 95% confidence interval for each dataset when averaging balanced accuracy for each classifier in the figure 7.10.

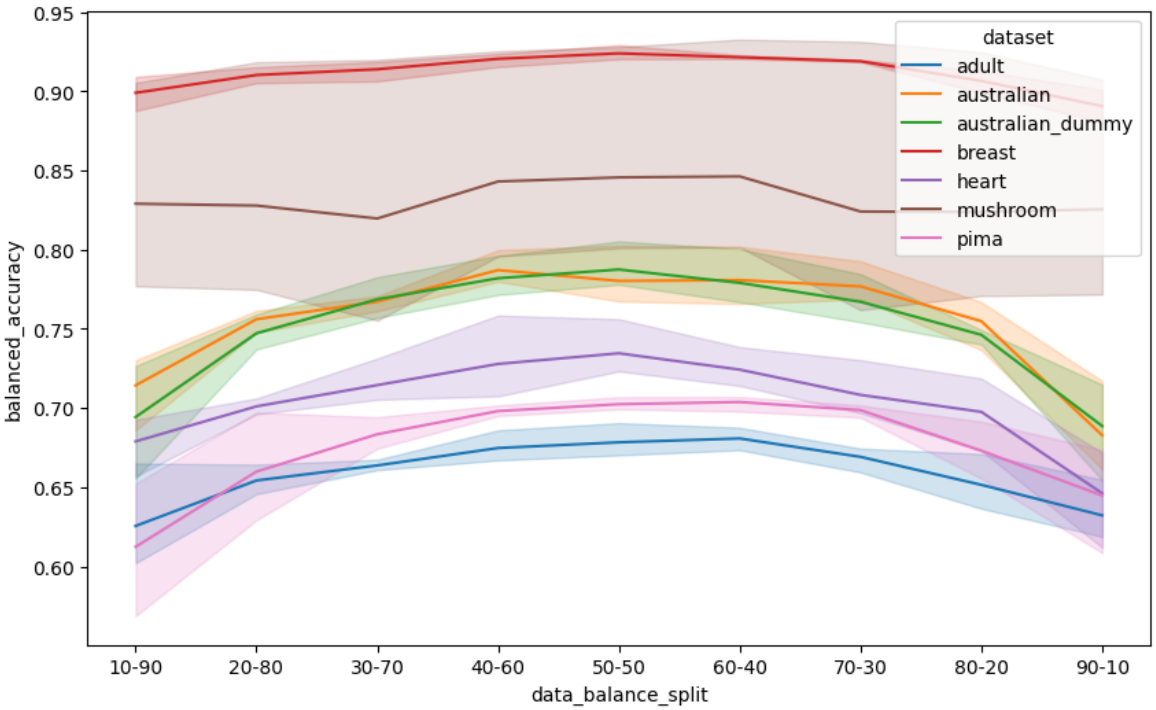


Figure 7.10: Balanced accuracy mean and 95% confidence interval for each dataset.

The graph shows how the confidence interval for each data split (x-axis) for balanced accuracy (y-axis) runs. We can see that for the $mushroom$ set this interval is the widest, which is due to the extremely high approximation of this set by data granularity methods, which then results in a small number of objects in the sets that are used to train the models and the fluctuation of the accuracy metric is much higher. This can also be seen in the graph 7.8 where the differences in balanced accuracy for granulated sets are the largest among the sets tested.

7.3.2. Scenario 2 and scenario 3

The results achieved for scenario 2 and scenario 3 do not differ significantly from those of scenario 1. Therefore, graphs for each dataset will not be included here, only some summaries for the scenario and, in particular, summaries for all three scenarios.

In the table below for scenario 2, the random forest is again the most effective classifier, coming in first place in 7 out of 8 cases. In second place is, as in scenario 1, the xgboost classifier once in first place and seven times in second place. The next places go to the classifiers decision tree, logistic regression, naive bayes and knn, so here there are no significant changes from the results presented in scenario 1.

For scenario 3, the situation is almost identical, except that the random forest algorithm achieved the highest accuracy for all 8 data splits. The second most effective classifier is again xgboost. The next places are again occupied by the logistic regression and decision tree algorithms, with very little difference, although for the case of the 90/10 data split the naive bayes classifier also reached the 3rd result.

It can also be noted that the results for scenario 3 are the best of all those tested, which may indicate that the removal of objects that lie near the "boundaries" of the two classes, followed by oversampling, introduces fewer new objects that may be on that boundary, reducing the uncertainty arising from the data.

Table 7.4: Classifiers ranking - mean balanced accuracy for each tested case and all scenarios.

balance split	classifier	acc sc. 1	acc sc. 2	acc sc. 3
90-10	xgboost	0.7497	0.7480	0.7540
90-10	random_forest	0.7483	0.7431	0.7624
90-10	decision_tree	0.7354	0.7367	0.7343
90-10	logistic_regression	0.7249	0.7216	0.7353
90-10	naive_bayes	0.7212	0.7187	0.7447
90-10	knn	0.6455	0.6436	0.6456

Continued on next page

balance split	classifier	acc sc. 1	acc sc. 2	acc sc. 3
80-20	random_forest	0.8008	0.7979	0.8208
80-20	xgboost	0.7848	0.7805	0.7972
80-20	logistic_regression	0.7594	0.7561	0.7783
80-20	decision_tree	0.7583	0.7622	0.7748
80-20	naive_bayes	0.7287	0.7336	0.7435
80-20	knn	0.6684	0.6635	0.6744
70-30	random_forest	0.8202	0.8184	0.8454
70-30	xgboost	0.8005	0.7988	0.8152
70-30	logistic_regression	0.7775	0.7752	0.7962
70-30	decision_tree	0.7657	0.7728	0.7791
70-30	naive_bayes	0.7355	0.7403	0.7473
70-30	knn	0.6832	0.6790	0.6905
60-40	random_forest	0.8319	0.8343	0.8580
60-40	xgboost	0.8033	0.8068	0.8199
60-40	decision_tree	0.7848	0.7800	0.7836
60-40	logistic_regression	0.7835	0.7870	0.7939
60-40	naive_bayes	0.7378	0.7389	0.7491
60-40	knn	0.6947	0.6896	0.6953
50-50	random_forest	0.8331	0.8348	0.8578
50-50	xgboost	0.8111	0.8068	0.8154
50-50	logistic_regression	0.7901	0.7847	0.7930
50-50	decision_tree	0.7810	0.7762	0.7712
50-50	naive_bayes	0.7373	0.7391	0.7456
50-50	knn	0.6960	0.6882	0.7019
40-60	random_forest	0.8277	0.8296	0.8564
40-60	xgboost	0.8065	0.8051	0.8190
40-60	logistic_regression	0.7798	0.7832	0.7937
40-60	decision_tree	0.7796	0.7759	0.7741
40-60	naive_bayes	0.7387	0.7424	0.7533
40-60	knn	0.6928	0.6909	0.7031

Continued on next page

balance split	classifier	acc sc. 1	acc sc. 2	acc sc. 3
30-70	random_forest	0.8140	0.8186	0.8394
30-70	xgboost	0.7929	0.7921	0.8095
30-70	logistic_regression	0.7741	0.7697	0.7817
30-70	decision_tree	0.7694	0.7660	0.7782
30-70	naive_bayes	0.7372	0.7330	0.7449
30-70	knn	0.6819	0.6802	0.6962
20-80	random_forest	0.7938	0.7910	0.8120
20-80	xgboost	0.7675	0.7790	0.7910
20-80	logistic_regression	0.7636	0.7619	0.7710
20-80	decision_tree	0.7547	0.7523	0.7684
20-80	naive_bayes	0.7333	0.7296	0.7335
20-80	knn	0.6757	0.6673	0.6905
10-90	random_forest	0.7490	0.7529	0.7634
10-90	xgboost	0.7385	0.7365	0.7484
10-90	decision_tree	0.7343	0.7313	0.7259
10-90	logistic_regression	0.7340	0.7265	0.7394
10-90	naive_bayes	0.7152	0.7283	0.7189
10-90	knn	0.6518	0.6483	0.6614

There were no significant changes in the summary of averaged balanced accuracy results for the training sets.

Table 7.5: Training dataset ranking for all scenarios.

balance split	acc sc. 1	acc sc. 2	acc sc. 3
balanced_nil	0.7872	0.7872	0.7880
balanced_cdgran_1.0	0.7866	0.7856	0.7863
imbalanced_nil	0.7664	0.7663	0.7663
imbalanced_cdgran_1.0	0.7658	0.7644	0.7658
balanced_hom_gran	0.7535	0.7544	0.7552
imbalanced_hom_gran	0.7464	0.7488	0.7470
balanced_cdgran_0.5	0.7136	0.7117	0.7133
imbalanced_cdgran_0.5	0.7098	0.7035	0.7045

Finally, for consistency in data presentation, tables are presented with the main statistics of the distribution of results for Scenarios 2 and 3 separately.

Table 7.6: Distribution statistics for each training subset for scenario 2.

data_balance	balanced_accuracy					
	std	min	25%	50%	75%	max
balanced_cdgran_0.5	0.1178	0.4699	0.6254	0.7127	0.7746	0.9474
balanced_cdgran_1.0	0.1214	0.5353	0.6881	0.7774	0.8869	1.0000
balanced_hom_gran	0.1064	0.5140	0.6709	0.7446	0.8290	0.9642
balanced_nil	0.1209	0.5404	0.6940	0.7821	0.8880	1.0000
imbalanced_cdgran_0.5	0.1196	0.4764	0.6185	0.6997	0.7744	0.9609
imbalanced_cdgran_1.0	0.1348	0.5028	0.6588	0.7659	0.8700	1.0000
imbalanced_hom_gran	0.1099	0.5038	0.6698	0.7420	0.8278	0.9560
imbalanced_nil	0.1336	0.5016	0.6604	0.7663	0.8707	0.9999

Table 7.7: Distribution statistics for each training subset for scenario 3.

data_balance	balanced_accuracy					
	std	min	25%	50%	75%	max
balanced_cdgran_0.5	0.1161	0.4757	0.6487	0.7111	0.7716	0.9493
balanced_cdgran_1.0	0.1210	0.5294	0.6930	0.7815	0.8894	1.0000
balanced_hom_gran	0.1070	0.5169	0.6751	0.7470	0.8276	0.9688
balanced_nil	0.1204	0.5328	0.6913	0.7844	0.8916	0.9999
imbalanced_cdgran_0.5	0.1176	0.4551	0.6185	0.6990	0.7667	0.9475
imbalanced_cdgran_1.0	0.1330	0.4961	0.6563	0.7680	0.8675	1.0000
imbalanced_hom_gran	0.1096	0.4906	0.6669	0.7465	0.8193	0.9680
imbalanced_nil	0.1327	0.5002	0.6621	0.7736	0.8742	0.9999

7.4. Conclusion

As a result of our experiments, we cannot conclusively state that oversampling and undersampling have a significant effect on classification results for granular sets. However, we can conclude with little confidence that the use of undersampling methods before oversampling methods can have a more positive effect on classification results than the reverse order of applying data balancing algorithms. We can also learn from these experiments that training sets subjected to homogeneous granulation and then classification models built on them achieve accuracy values with less scatter. We can also determine that the original sets, whose class distribution will be balanced, allow to achieve the highest values of classification efficiency.

Part V

Summary

The aim of this study was to develop and test algorithms: develop granulation techniques - derived from the methods discovered by Prof. Polkowski and apply them to selected data analysis problems. The main original achievements presented in the dissertation are:

(1) development of homogeneous granulation and its epsilon variant.

(2) Application of the granulation method in the creation of an ensemble model - the Ensemble of Random Granules model is presented.

(3) Applications of knowledge granulation techniques in absorbing missing values were tested.

(4) Finally, the effect of oversampling and undersampling on the granulation process was investigated.

These results, together with the theoretical considerations carried out, make it possible to confirm the assumed theses:

(i) Finally, knowledge granulation methods that do not require estimation of optimal parameters were designed. This included homogeneous granulation and an epsilon variant of homogeneous granulation. The effectiveness of this method was verified through the lens of data classification.

(ii) It was verified experientially that granular reflections of decision systems, even when they are in the form of a few percentages of the original training system retain classification efficiency and work well as a committee of classifiers - in the Ensemble model. It was verifiable to test the Ensemble of Random Granules model and some variant dedicated to the largest possible approximation of decision systems.

(iii) It has been confirmed in studies that homogeneous granulation variants effectively absorb missing values while maintaining classification efficiency.

(iv) It has been confirmed in studies that oversampling and undersampling techniques affect the process of creating granular reflections of decision-making systems. In particular, the combination of undersampling methods followed by oversampling has a positive effect on the homogeneous granulation process - in the sense of improving the quality of classification.

In the course of the research carried out, a number of new themes emerged concerning application of our granular computing techniques. The author therefore plans to continue research in this area. In the near future, work is planned to

include: - Develop other variants of ensemble models, using a selection of other granulation models - in particular combining multiple granulation models with each other, - Develop new classification techniques based on homogeneous granulation - by immersing test objects in granular systems, - Applying the concept of granularity of our techniques to other data types including images.

List of Figures

2.1 Reflection dataset sizes comparison for each granulation radius. 52
2.2 Class balance comparison between concept dependent and standard granulation. Adult dataset, radius 0.5. 55
2.3 Class balance comparison between concept dependent and standard granulation. Adult dataset, radius 0.79. 55
2.4 Class balance comparison between concept dependent and standard granulation. Pima dataset, radius 0.25. 56
2.5 Class balance comparison between concept dependent and standard granulation. Heart dataset, radius 0.54. 56
2.6 Class balance comparison between concept dependent and standard granulation. Mushroom dataset, radius 0.5. 57
2.7 Class balance comparison between concept dependent and standard granulation. Australian dataset, radius 0.5. 57
2.8 Class balance comparison between concept dependent and standard granulation. Australian dummy dataset, radius 0.76. 58
2.9 Class balance comparison between concept dependent and standard granulation. Red wine dataset, radius 0.27. 58
2.10 Classification nil-case experiment pipeline. 59
2.11 Classification of granuled datasets experiment pipeline. 63

3.1 Box plots presenting a distribution of reflection set sizes for the iris, australian and australian_dummy datasets. 91
3.2 Box plots presenting a distribution of reflection set sizes for the heart desease, pima diabetes and breast cancer datasets. 91
3.3 Box plots presenting a distribution of reflection set sizes for the mushroom numeric encoded, red wine and white wine datasets. 92
3.4 Box plots presenting a distribution of reflection set sizes for the wine merged and adult datasets. 92
3.5 Bar plot nil-case vs homogeneous granulation balanced accuracy for adult dataset. 108

3.6 Bar plot nil-case vs homogeneous granulation balanced accuracy for australian dataset. . . . 109
3.7 Bar plot nil-case vs homogeneous granulation balanced accuracy for australian dummy dataset. . . . 110
3.8 Bar plot nil-case vs homogeneous granulation balanced accuracy for breast dataset. . . . 111
3.9 Bar plot nil-case vs homogeneous granulation balanced accuracy for heart dataset. . . . 112
3.10 Bar plot nil-case vs homogeneous granulation balanced accuracy for iris dataset. 113
3.11 Bar plot nil-case vs homogeneous granulation balanced accuracy for mushroom num dataset. . . . 113
3.12 Bar plot nil-case vs homogeneous granulation balanced accuracy for pima dataset. . . . 114
3.13 Bar plot nil-case vs homogeneous granulation balanced accuracy for red wine dataset. . . . 114
3.14 Bar plot nil-case vs homogeneous granulation balanced accuracy for white wine dataset. . . . 115
3.15 Bar plot nil-case vs homogeneous granulation balanced accuracy for wine merged dataset. . . . 115

5.1 Ensemble of Random Granular Reflections . . . 130
5.2 Ensemble of Random Granular Reflections for the Australian credit data set - the accuracy of classification - 50 iterations of learning - exemplary run . . . 132
5.3 Bagging Ensemble model for the Australian credit data set - the accuracy of classification - 50 iterations of learning - exemplary run . . . 132
5.4 Ada-Boost Ensemble model for the Australian credit data set - the accuracy of classification - 50 iterations of learning - exemplary run . . . 133
5.5 Pure Bagging Ensemble model for the Australian credit data set - the accuracy of classification - 50 iterations of learning - exemplary run . . . 133
5.6 Ensemble of Random Granular Reflections for the **Iris data set** - the accuracy of classification - 50 iterations of learning - exemplary run; Decision Tree classifier 135
5.7 Ensemble of Random Granular Reflections for the **Australian credit** data set - the accuracy of classification - 50 iterations of learning - exemplary run; Decision Tree classifier . . . 136

5.8 Ensemble of Random Granular Reflections for the **Heart Disease data set** - the accuracy of classification - 50 iterations of learning - exemplary run; Decision Tree classifier . . . 137
5.9 Ensemble of Random Granular Reflections for the **Pima indians diabetes data set - dummy version** - the accuracy of classification - 50 iterations of learning - exemplary run; Decision Tree classifier . . . 138
5.10 Ensemble of Random Granular Reflections for the **Australian credit** data set - the accuracy of classification - 50 iterations of learning - exemplary run; **Random Forest** classifier . . . 140
5.11 Ensemble of Random Granular Reflections for the **Australian credit** data set - the accuracy of classification - 50 iterations of learning - exemplary run; **k Nearest Neighbor** classifier . . . 141
5.12 Ensemble of Random Granular Reflections for the **Australian credit** data set - the accuracy of classification - 50 iterations of learning - exemplary run; **Naive Bayes** classifier . . . 142
5.13 Average size of granular systems used in the classification. Despite the use of such small representations, drawing knowledge from the entire original training system, granular systems generated on random central objects show the ability to reinforce classifiers at the level of using the original training data process . . 145

7.1 Visualization of the effects of the SMOTE algorithm on a selected attribute of the pima set. . . . 158
7.2 Pipeline for over and undersampling experiment pipeline. . . . 160
7.3 Balanced accuracy plot for SMOTE oversampling experiments for the australian dataset. . . . 163
7.4 Balanced accuracy plot for SMOTE oversampling experiments for the australian dummy dataset. . . . 164
7.5 Balanced accuracy plot for SMOTE oversampling experiments for the heart dataset. . . . 165
7.6 Balanced accuracy plot for SMOTE oversampling experiments for the pima dataset. . . . 166
7.7 Balanced accuracy plot for SMOTE oversampling experiments for the breast dataset. . . . 167
7.8 Balanced accuracy plot for SMOTE oversampling experiments for the mushroom dataset. . . . 168

7.9 Balanced accuracy plot for SMOTE oversampling experiments for the adult dataset. 169
7.10 Balanced accuracy mean and 95% confidence interval for each dataset. 176

List of Tables

2.1 The dataset used to demonstrate an example of how the concept-dependent granulation algorithm works. . . . 11
2.2 Indiscernibility matrix for radius 0/4(special case) and radius 1/4, concept dependent granulation. . . . 12
2.3 Indiscernibility matrix for radius 2/4, concept dependent granulation. . . . 12
2.4 Indiscernibility matrix for radius 3/4 and 4/4, concept dependent granulation. . . . 13
2.5 Reflection dataset for radius 0/4 (0.0), concept dependent granulation. . . . 17
2.6 Reflection dataset for radius 1/4 (0.25), concept dependent granulation. . . . 17
2.7 Reflection dataset for radius 2/4 (0.5), concept dependent granulation. . . . 17
2.8 Reflection dataset for radius 3/4 (0.75), concept dependent granulation. . . . 18
2.9 Reflection dataset for radius 4/4 (1.0), concept dependent granulation. . . . 18
2.10 Indiscernibility matrix for radius 0/4 (0.0)), standard granulation. . . . 21
2.11 Indiscernibility matrix for radius 1/4 (0.25), standard granulation. . . . 21
2.12 Indiscernibility matrix for radius 2/4 (0.5), standard granulation. . . . 21
2.13 Indiscernibility matrix for radius 3/4 (0.75) and 4/4 (1.0), standard granulation. . . . 22
2.14 Reflection dataset for radius 0/4, standard granulation. . . . 23
2.15 Reflection dataset for radius 1/4, standard granulation. . . . 24
2.16 Reflection dataset for radius 2/4, standard granulation. . . . 24
2.17 Reflection dataset for radius 3/4, standard granulation. . . . 25
2.18 Reflection dataset for radius 4/4, standard granulation. . . . 25
2.19 List of used datasets in the experimental sessions. . . . 28
2.20 Detailed information about datasets sizes after 10-times concept dependent granulation. . . . 35
2.21 Detailed information about datasets decision class balance (average value) after 10-times concept dependent granulation. . . . 39
2.22 Detailed information about datasets sizes after 10-times standard granulation. . . . 43
2.23 Detailed information about datasets decision class balance (average value) after 10-times standard granulation. . . . 47

2.24 Comparison of class balance between concept dependent granulation radius of 1, standard granulation of radius 1 and original datasets. 53
2.25 Comparison of class balance between concept dependent granulation radius and standard granulation for chosen radiuses. 54
2.26 Classification results for nil case (original data) for all selected datasets. 60
2.27 Classification results comparison between nil-case and concept dependent granulation with radius equal to 1. 64
2.28 Mean classification change for concept dependent granulation classification vs. nil-case. 67
2.29 Classification results comparison between nil-case and concept dependent granulation with radius equal to 0.5. 68
2.30 Granuled dataset sizes for concept dependent granulation for radius 0.5. 71
2.31 Mean percent point balanced accuracy bias for concept dependent granuled data for radius 0.5 vs nil-case classification. 71
2.32 Classification results comparison between nil-case and standard granulation with radius equal to 1. 73
2.33 Classification results comparison between nil-case and standard granulation with radius equal to 0.5. 76
2.34 Balanced accuracy change for radius 1 and 0.5 along with granular sets sizes change. 79

3.1 Homogeneous granulation toy example - objects in granule for u_1 and radius of 4/4 (1.0). 84
3.2 Homogeneous granulation toy example - objects in granule for u_1 and radius of 3/4 (0.75). 85
3.3 Homogeneous granulation toy example - objects in granule for u_1 and radius of 2/4 (0.5). 85
3.4 Homogeneous granulation toy example - objects in granule for u_1 and radius of 1/4 (0.25). 86
3.5 Homogeneous granulation toy example - objects in granule for u_1 and radius of 0/4 (0.0). 87
3.6 Reflection dataset for radius homogeneous granulation. 89
3.7 Aggregated reflection sizes after 10 times homogeneous granulation process. . 90
3.8 Class balance after 10 times homogeneous granulation. 91
3.9 Entropy for original dataset and homogeneously granuled datasets for every feature. 94

3.10 Entropy change after homogeneous granulation - summary for each dataset. . . 100
3.11 Classification results for homogeneous granulation. 101
3.12 Balanced accuracy comparison between nil-case classification and homogeneous granulation classification. 104

4.1 Estimated parameters for kNN based on 5×CV5 121
4.2 Training data system (U_{trn}, A, d), (a sample from australian credit data set), for $varepsilon = 0.05$ 121
4.3 Granular decision system formed from Covering granules 122
4.4 Data Sets description 123
4.5 The result for homogeneous granulation (HG) and for epsilon homogeneous granulation ($\varepsilon - HGS$) - 5 times CV5 method; $HG\_acc =$ average accuracy for HG, $\varepsilon - HG\_acc$ average accuracy for $\varepsilon - HGS$, $HGS\_size = HG$ decision system size, $\varepsilon - HGS\_size = \varepsilon - HGS$ decision system size, $TRN\_size = training\ set\ size$, $HG_TRN_red =$reduction in object number in training set for HG, $\varepsilon - HGS_size =$reduction in object number in training set for $\varepsilon - HGS$, $HG_r_range =$spectrum of radii for HG, $\varepsilon - HG\_r\_range =$ spectrum of radii for $\varepsilon - HGS$ 124
4.6 Summary of results, k-NN vs Naive Bayes Classifier, granular and non granular case, acc=accuracy of classification, red=percentage reduction in object number, r=granulation radius, $method$=variant of Naive Bayes classifier 124

5.1 The result for homogeneous granulation, $5 \times CV5$, $GS\_size = granular\ decision\ system\ size$, $TRN\_size = training\ set\ size$, $TRN\_reduction = reduction\ in\ object\ number\ in\ training\ size$, $radii\_range = spectrum\ of\ radii$. 144

6.1 Missing values absorption based on **Concept dependent granulation**; 5 x CV-5; A,B,C,D strategies vs complete data classification; $\mathbf{Australian\ Credit}$; ; $\mathbf{10\ percent\ of\ missing\ values}$; $r_{gran} =$ Granulation radius; nil = result for data without missing values; Acc = Accuracy of classification; $AccBias$ = Accuracy bias defined based on equation efAcccBiasEquation; $GranSize$ = The size of data set after granulation in the fixed r 152

6.2 Missing values absorption based on **Concept dependent granulation**; 5 x CV-5; A,B,C,D strategies vs complete data classification; $\mathbf{Pima\ Indians\ Diabetes}$; Concept dependent granulation; $\mathbf{10\ percent\ of\ missing\ values}$; $r_{gran} =$ Granulation radius; nil = result for data without missing values; Acc = Accuracy of classification; $AccBias$ = Accuracy bias defined based on equation efAcccBiasEquation; $GranSize$ = The size of data set after granulation in the fixed r 153

6.3 Missing values absorption based on **Homogeneous granulation**; 5 x CV-5; A,B,C,D strategies vs complete data classification; $\mathbf{Australian\ Credit}$; Homogeneous granulation; $\mathbf{10\ percent\ of\ missing\ values}$; $r_{gran} =$ Granulation radius; nil = result for data without missing values; Acc = Accuracy of classification; $AccBias$ = Accuracy bias defined based on equation efAcccBiasEquation; $GranSize$ = The size of data set after granulation in the fixed r 153

6.4 Missing values absorption based on **Homogeneous granulation**; 5 x CV-5; A,B,C,D strategies vs complete data classification; $\mathbf{Pima\ Indians\ Diabetes}$; Homogeneous granulation; $\mathbf{10\ percent\ of\ missing\ values}$; $r_{gran} =$ Granulation radius; nil = result for data without missing values; Acc = Accuracy of classification; $AccBias$ = Accuracy bias defined based on equation efAcccBiasEquation; $GranSize$ = The size of data set after granulation in the fixed r 154

7.1 Training dataset ranking for scenario 1. 172

7.2 Classifiers ranking - mean balanced accuracy for each tested case and scenario 1. 172

7.3 Distribution statistics for each training subset for scenario 1. 175

7.4 Classifiers ranking - mean balanced accuracy for each tested case and all scenarios. 177

7.5 Training dataset ranking for all scenarios. 179

7.6 Distribution statistics for each training subset for scenario 2. 180

7.7 Distribution statistics for each training subset for scenario 3. 181

List of Algorithms

1 Nil case classification pipeline. 59
2 Standard granulation classification pipeline. 72

Bibliography

[1] P. Artiemjew. Classifiers from granulated data sets: Concept dependent and layered granulation. *In Proceedings RSKD'07. The Workshops at ECML/PKDD'07*,, pages 1–9, 2007. *(Cited on pages 26 and 127.)*

[2] P. Artiemjew. A review of the knowledge granulation methods: Discrete vs. continuous algorithms. *In Skowron A., Suraj Z. (eds.)(2013): Rough Sets and Intelligent Systems.*, (ISRL 43):41–59, 2013. *(Cited on page 127.)*

[3] P. Artiemjew. Boosting effect of classifier based on simple granules of knowledgeg. *In: Information technolojy and control*, 47(2), 2018. *(Cited on page 128.)*

[4] P. Artiemjew and K. Ropiak. A novel ensemble model - the random granular reflections. In Bernd-Holger Schlingloff and Samira Akili, editors, *Proceedings of the 27th International Workshop on Concurrency, Specification and Programming, Berlin, Germany, September 24-26, 2018*, volume 2240 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2018. *(Cited on pages 5, 6, and 82.)*

[5] P. Artiemjew and K. Ropiak. On granular rough computing: Handling missing values by means of homogeneous granulation. *Comput.*, 9(1):13, 2020. *(Cited on page 82.)*

[6] L. Breiman. Arcing classifier (with discussion and a rejoinder by the author). *Ann. Statist.*, 26(3):801–849, 1998. *(Cited on page 129.)*

[7] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, June 2002. *(Cited on page 157.)*

[8] R. Cybulski and P. Artiemjew. Accelerating concept-dependent granulation technique using data decomposition. In Shusaku Tsumoto, Yukio Ohsawa, Lei Chen, Dirk Van den Poel, Xiaohua Hu, Yoichi Motomura, Takuya Takagi, Lingfei Wu, Ying Xie, Akihiro Abe, and Vijay Raghavan, editors, *IEEE International Conference on Big Data, Big Data 2022, Osaka, Japan, December 17-20, 2022*, pages 6195–6201. IEEE, 2022. *(Cited on page 128.)*

[9] R. Cybulski and P. Artiemjew. Application of random sampling in the concept-dependent granulation method. In Maria Ganzha, Leszek A. Maciaszek, Marcin Paprzycki, and Dominik Slezak, editors, *Position Papers of the 17th Conference on Computer Science and Intelligence Systems, FedCSIS 2022, Sofia, Bulgaria, September 4-7, 2022*, volume 31

of *Annals of Computer Science and Information Systems*, pages 3–11, 2022. *(Cited on page 128.)*

[10] R. Cybulski and P. Artiemjew. Data streaming in concept-dependent granulation. In Jingrui He, Themis Palpanas, Xiaohua Hu, Alfredo Cuzzocrea, Dejing Dou, Dominik Slezak, Wei Wang, Aleksandra Gruca, Jerry Chun-Wei Lin, and Rakesh Agrawal, editors, *IEEE International Conference on Big Data, BigData 2023, Sorrento, Italy, December 15-18, 2023*, pages 6033–6039. IEEE, 2023. *(Cited on page 128.)*

[11] A. Fernández, S. García, M. Galar, R.C. Prati, B. Krawczyk, and F. Herrera. *Learning from Imbalanced Data Sets*. Springer, 2018. *(Cited on page 143.)*

[12] P. Hart. The condensed nearest neighbor rule (corresp.). *IEEE Transactions on Information Theory*, 14(3):515–516, 1968. *(Cited on page 159.)*

[13] H. He and Y. Ma. *Imbalanced Learning: Foundations, Algorithms, and Applications*. Wiley-IEEE Press, 1st edition, 2013. *(Cited on page 143.)*

[14] T. K. Ho. Random decision forests. *Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC,*, page 278–282, 1995. *(Cited on page 128.)*

[15] X. Hu. Ensembles of classifiers based on rough sets theory and set-oriented database operations. *Presented at the 2006 IEEE International Conference on Granular Computing, Atlanta, GA*, 2006. *(Cited on page 127.)*

[16] M. Kelly and K. Longjohn, R. nad Nottingham. Uci machine learning repository, 2017. *(Cited on pages 27, 123, and 149.)*

[17] C. Murthy, S. Saha, and S.K. Pal. Rough set based ensemble classifier. *In: Rough Sets, Fuzzy Sets, Data Mining and Granular Computing Lecture Notes in Computer Science*, 6743, 2001. *(Cited on page 127.)*

[18] L. Ohno-Machado. Cross-validation and bootstrap ensembles, bagging, boosting. *Medical Decision Support*, pages 1–422, 2005. *(Cited on page 129.)*

[19] Z. Pawlak. Rough sets. *International Journal of Computer and Information Sciences*, 11:341–356, 1982. *(Cited on pages 5, 6, and 8.)*

[20] W. Pedrycz. Shadowed sets: representing and processing fuzzy sets. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 28(1):103–109, 1998. *(Cited on page 9.)*

[21] W. Pedrycz. *Granular Computing: An Introduction. Kluwer Academic, Dordrecht*. 01 2003. *(Cited on page 9.)*

[22] W. Pedrycz, A. Skowron, and V. Kreinovich. *Handbook of Granular Computing*. John Wiley & Sons, New York, NY, 2008. *(Cited on page 9.)*

[23] G. Peters and R. Weber. Dcc: a framework for dynamic granular clustering. *Granular Computing*, 1(1):1–11, 2016. *(Cited on page 9.)*

[24] L. Polkowski. Formal granular calculi based on rough inclusions. *Proceedings of IEEE 2005 International Conference on Granular Computing GrC05, Tsinghua Univ., Beijing, China*, 2005:57–62, 2005. *(Cited on pages 8, 9, 10, 19, and 127.)*

[25] L. Polkowski. A model of granular computing with applications. *Proceedings of IEEE 2006 International Conference on Granular Computing GrC06, Atlanta, USA*, 2006:9–16, 2006. *(Cited on pages 8, 9, 10, 19, and 127.)*

[26] L. Polkowski. Granulation of knowledge in decision systems: The approach based on rough inclusions. the method and its applications. *LNAI,volume 4585, proceedings for RSEISP 2007: Rough Sets and Intelligent Systems Paradigms*, pages 69–79, 2007. *(Cited on page 9.)*

[27] L. Polkowski. *Approximate Reasoning by Parts. An Introduction to Rough Mereology.* Springer Verlag, Berlin, 2011. *(Cited on page 127.)*

[28] L. Polkowski and P. Artiemjew. Granular computing: Granular classifiers and missing values. *In Proceedings of the 6th IEEE International Conference on Cognitive Informatics ICCI'07*, pages 186–194, 2007. *(Cited on page 146.)*

[29] L. Polkowski and P. Artiemjew. On granular rough computing with missing values. *In Proceedings RSEISP'07, Lecture Notes in Artificial Intelligence*, 4585:271–279, 2007. *(Cited on page 146.)*

[30] L. Polkowski and P. Artiemjew. Granular computing in decision approximation - an application of rough mereology. *in: Intelligent Systems Reference Library 77*, 5390:1–422, 2015. *(Cited on pages 119, 121, 123, 127, 146, and 151.)*

[31] L. Polkowski and A. Skowron. Rough mereology. In *Methodologies for Intelligent Systems*, pages 85–94, Berlin, Heidelberg, 1994. Springer Berlin Heidelberg. *(Cited on page 8.)*

[32] L. Polkowski and A. Skowron. Rough mereology: A new paradigm for approximate reasoning. *International Journal of Approximate Reasoning*, 15(4):333–365, 1996. Rough Sets. *(Cited on page 8.)*

[33] K. Ropiak and P. Artiemjew. On granular rough computing: Epsilon homogenous granulation. In Hung Son Nguyen, Quang-Thuy Ha, Tianrui Li, and Malgorzata Przybyla-Kasperek, editors, *Rough Sets - International Joint Conference, IJCRS 2018, Quy Nhon, Vietnam, August 20-24, 2018, Proceedings*, volume 11103 of *Lecture Notes in Computer Science*, pages 546–558. Springer, 2018. *(Cited on pages 127 and 146.)*

[34] K. Ropiak and P. Artiemjew. A study in granular computing: Homogenous granulation. In Robertas Damasevicius and Giedre Vasiljeviene, editors, *Information and Software Technologies - 24th International Conference, ICIST 2018, Vilnius, Lithuania, October 4-6, 2018, Proceedings*, volume 920 of *Communications in Computer and Information Science*, pages 336–346. Springer, 2018. *(Cited on pages 82 and 127.)*

[35] K. Ropiak and P. Artiemjew. Homogenous granulation and its epsilon variant. *Comput.*, 8(2):36, 2019. *(Cited on page 82.)*

[36] K. Ropiak and P. Artiemjew. On a hybridization of deep learning and rough set based granular computing. *Algorithms*, 13(3), 2020. *(Cited on pages 5 and 6.)*

[37] S. Saha, C.A. Murthy, and S.K. Pal. Rough set based ensemble classifier for web page classification. *Fundamenta Informaticae*, 76:171–187, 2007. *(Cited on page 127.)*

[38] R.E. Schapire. A short introduction to boosting. 1999. *(Cited on page 129.)*

[39] R.E. Schapire. The boosting approach to machine learning: An overview. *MSRI (Mathematical Sciences Research Institute) Workshop on Nonlinear Estimation and Classification*, 2003. *(Cited on page 129.)*

[40] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948. *(Cited on page 93.)*

[41] L. Shi, M. Weng, X. Ma, and L. Xi. Rough set based decision tree ensemble algorithm for text classification. *Journal of Computational Information Systems*, 1(6):89–95, 2010. *(Cited on page 127.)*

[42] D. Shifei, D. Mingjing, and Z. Hong. Survey on granularity clustering. *Cognitive Neurodynamics*, 9, 07 2015. *(Cited on page 9.)*

[43] I. Tomek. Two modifications of cnn. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(11):769–772, 1976. *(Cited on page 159.)*

[44] L. Tsau Young and L. Churn-jung. *Granular Computing and Rough Sets*, pages 535–561. 01 2005. *(Cited on page 9.)*

[45] P. Yang, Y.H. Yang, B.B. Zhou, and A.Y. Zomaya. A review of ensemble methods in bioinformatics: Including stability of feature selection and ensemble feature selection methods. *Current Bioinformatics*, 4(5):296–308, 2010, updated on 2016. *(Cited on page 128.)*

[46] L.A. Zadeh. Fuzzy sets and information granularity. Technical Report UCB/ERL M79/45, EECS Department, University of California, Berkeley, Jun 1979. *(Cited on page 9.)*

[47] L.A. Zadeh. Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic. *Fuzzy Sets and Systems*, 90(2):111–127, 1997. Fuzzy Sets: Where Do We Stand? Where Do We Go? *(Cited on page 9.)*

[48] J. Zhang and I. Mani. KNN Approach to Unbalanced Data Distributions: A Case Study Involving Information Extraction. In *Proceedings of the ICML'2003 Workshop on Learning from Imbalanced Datasets*, 2003. *(Cited on page 159.)*

[49] Z.-H. Zhou. Ensemble methods: Foundations and algorithms. *Chapman and Hall/CRC.*, 2012. *(Cited on page 128.)*

[50] Z.-H. Zhou. Boosting 25 years. *CCL 2014 Keynote*, 2014. *(Cited on page 129.)*

[51] J. Łukasiewicz. O logice trojwartosciowej. *Ruch Filozoficny, 5*, pages 170–171, 1920. *(Cited on page 10.)*