Chapter 9

Rough Mereological Calculus of Granules in Decision and Classification Problems

The idea of mereological granulation of knowledge, proposed and presented in detail in Ch. 7, sect. 3, finds an effective application in problems of synthesis of classifiers from data tables. This application consists in granulation of data at preprocessing stage in the process of synthesis: after granulation, a new data set is constructed, called a granular reflection, to which various strategies for rule synthesis can be applied. This application can be regarded as a process of *filtration* of data, aimed at reducing noise immanent to data. This chapter presents this application.

0.1 On decision rules

In Ch. 4, we have given an introduction to the problem of decision rule synthesis from decision systems, so basic notions and results are known to the reader. Now, we comment on a more specific problem of quality of decision rules and classifiers. We recall that decision rules are formed in the frame of a *decision system* (U, A, d), as implications of the form

$$\bigwedge_i (a_i = v_i) \Rightarrow (d = v) \tag{0.1}$$

where a_i are *conditional attributes*, v_i are their values, and, v is a value of the decision.

A *decision algorithm, classifier* is a judiciously chosen set of decision rules, approximating possibly most closely the real decision function, which, by necessity, is not known to us. This comes down to a search in the space of possible descriptors in order to find their successful combinations. In order to judge the quality, or, degree of approximation, decision rules are learned on a part of the decision system, the *training set* and then the decision algorithm is *tested* on the remaining part of the decision system, called the *test set*. Degree of approximation is measured by some coefficients of varied character. Simple measures of statistical character are found from the *contingency table*, see Arkin and Colton [5]. This table is built for each decision rule r and a decision value v, by counting the number n_t of training objects, the number n_r of objects satisfying the premise of the rule r (caught by the rule), $n_r(v)$ is the number of objects counted in n_r and with the decision v, and $n_r(\neg v)$ is the number of objects counted in n_r but with decision value distinct from

v. To these factors, we add n_v, the number of training objects with decision v and $n_{\neg v}$, the number of remaining objects, i.e, $n_{\neg v} = n_t - n_v$.

For these values, *accuracy of the rule r relative to v* is the quotient

$$acc(r, v) = \frac{n_r(v)}{n_r} \tag{0.2}$$

and *coverage of the rule r relative to v* is

$$cov(r, v) = \frac{n_r(v)}{n_v} \tag{0.3}$$

These values are useful as indicators of a *rule strength* which is taken into account when classification of a test object is under way: to assign the value of decision, a rule pointing to a decision with a maximal value of accuracy, or coverage, or combination of both can be taken; methods for combining accuracy and coverage into a single criterion are discussed, e.g., in Michalski [10]. Accuracy and coverage can, however, be defined in other ways; for a decision algorithm D, trained on a training set Tr, and a test set Tst, the *accuracy* of D is measured by its efficiency on the test set and it is defined as the quotient

$$accuracy(D) = \frac{n_{corr}}{n_{caught}} \tag{0.4}$$

where n_{corr} is the number of test objects correctly classified by D and n_{caught} is the number of test objects classified.

Similarly, *coverage* of D is defined as

$$coverage(D) = \frac{n_{caught}}{n_{test}} \tag{0.5}$$

where n_{test} is the number of test objects. Thus, the product $accuracy(D) \cdot coverage(D)$ gives the measure of the fraction of test objects correctly classified by D.

We have already mentioned that accuracy and coverage are often advised to be combined in order to better express the trade–off between the two: one may have a high accuracy on a relatively small set of caught objects, or a lesser accuracy on a larger set of caught by the classifier objects. Michalski [10] proposes a combination rule of the form

$$MI \ = \ \frac{1}{2} \cdot A + \frac{1}{4} \cdot A^2 + \frac{1}{2} \cdot C - \frac{1}{4} \cdot A \cdot C \tag{0.6}$$

where A stands for accuracy and C for coverage. With the symbol MI, we denote the *Michalski index* as defined in (0.6).

Statistical measures of correlation between the rule r and a decision class v are expressed, e.g., by χ^2 statistic

$$\chi^2 = \frac{n_t \cdot (n_r(v) \cdot n_{\neg r}(\neg v) - n_r(\neg v) \cdot n_{\neg r}(v))^2}{n(v) \cdot n(\neg v) \cdot n_r \cdot n_{\neg r}} \tag{0.7}$$

where $n_{\neg r}$ is the number of objects not caught by the rule r, see Bruning and Kintz [8].

We now restrict ourselves to rough set framework of decision systems and we denote a rule r by a convenient shortcut

$$r : \phi \Rightarrow (d = v) \tag{0.8}$$

where ϕ is a conjunct of descriptors, see Ch. 4, sect. 2. We recall that $[[\phi]]$ denotes the meaning of a rule ϕ, i. e., the set of objects satisfying ϕ. An object $u \in U$, is caught by r (or, *matches* r) in case $u \in \phi$; $match(r)$ is the number of objects matching r.

Support, $supp(r)$, of r is the number of objects in $[[\phi]] \cap [[(d = v)]]$; the fraction

$$cons(r) = \frac{supp(r)}{match(r)} \tag{0.9}$$

is the *consistency degree* of r; $cons(r) = 1$ means that the rule is *certain*, or *true*.

Strength, $strength(r)$, of the rule r is defined, see, e. g., Bazan [6], and Grzymala–Busse and Ming Hu [9], as the number of objects correctly classified by the rule in the training phase; *relative strength* is defined as the fraction

$$rel - strength(r) = \frac{supp(r)}{|[[(d = v)]]|} \tag{0.10}$$

Specificity of the rule r, $spec(r)$, is the number of descriptors in the premise ϕ of the rule r, Grzymala–Busse and Ming Hu [9].

In the testing phase, rules vie among themselves for object classification when they point to distinct decision classes; in such case, negotiations among rules or their sets are necessary. In these negotiations rules with better characteristics are privileged.

For a given decision class $c : d = v$, and an object u in the test set, the set $Rule(c, u)$ of all rules matched by u and pointing to the decision v, is characterized globally by

$$Support(Rule(c, u)) = \sum_{r \in Rule(c,u)} strength(r) \cdot \ spec(r) \tag{0.11}$$

The class c for which $Support(Rule(c, u))$ is the largest wins the competition and the object u is classified into the class $c : d = v$, see, e.g., Grzymala–Busse and Ming Hu [9].

It may happen that no rule in the available set of rules is matched by the test object u and *partial matching* is necessary, i.e., for a rule r, the *matching factor* $match - fact(r, u)$ is defined as the fraction of descriptors in the

premise ϕ of r matched by u to the number $spec(r)$ of descriptors in ϕ. The rule for which the *partial support*

$$Part{-}Support(Rule(c,u)) = \sum_{r \in Rule(c,u)} match{-}fact(r,u) \cdot strength(r) \cdot spec(r) \tag{0.12}$$

is the largest wins the competition and it does assign the value of decision to u, see Grzymala–Busse and Ming Hu [9].

In a similar way, notions based on relative strength can be defined for sets of rules and applied in negotiations among them, see Bazan et al. [7].

A combination of rough set methods with k-nearest neighbor idea, is a further refinement of the classification based on similarity or analogy, cf., an implementation in RSES [21]. In this approach, training set objects are endowed with a metric, and the test objects are classified by voting by k nearest training objects for some k that is subject to optimization, cf., Polkowski [18].

Our idea of augmenting existing strategies for rule induction consists in using granules of knowledge. The principal assumption we can make is that the nature acts in a continuous way: if objects are similar with respect to judiciously and correctly chosen attributes, then decisions on them should also be similar. A granule collecting similar objects should then expose the most typical decision value for objects in it while suppressing outlying values of decision, reducing noise in data, hence, leading to a better classifier.

These ideas were developed and proposed in Polkowski [12] – [15] see also surveys Polkowski [16] – [18]. In Polkowski and Artiemjew [19], [20] and in Artiemjew [1] – [4] the theoretical analysis was confirmed as to its application merits. We proceed with a summary of methods and results of these verification.

0.2 The idea of granular rough mereological classifiers

We assume that we are given a decision system (U, A, d) from which a classifier is to be constructed; on the universe U, a rough inclusion μ is given, and a radius $r \in [0,1]$ is chosen, see Polkowski [12] – [15].

We can find granules $g_\mu(u,r)$ defined as in Ch. 7, sect. 6, for all $u \in U$, and make them into the set $G(\mu,r)$.

From this set, a covering $Cov(\mu,r)$ of the universe U can be selected by means of a chosen strategy $\mathcal{G}$, i.e.,

$$Cov(\mu,r) = \mathcal{G}(G(\mu,r)) \tag{0.13}$$

We intend that $Cov(\mu, r)$ becomes a new universe of the decision system whose name will be the *granular reflection* of the original decision system. It remains to define new attributes for this decision system.

Each granule g in $Cov(\mu, r)$ is a collection of objects; attributes in the set $A \cup \{d\}$ can be factored through the granule g by means of a chosen strategy $\mathcal{S}$, i.e., for each attribute $q \in A \cup \{d\}$, the new factored attribute $\overline{q}$ is defined by means of the formula

$$\overline{q}(g) = \mathcal{S}(\{a(v) : ingr(v, g_\mu(u, r))\}) \tag{0.14}$$

In effect, a new decision system $(Cov(\mu, r), \{\overline{a} : a \in A\}, \overline{d})$ is defined. The object v with

$$Inf(v) = \{(\overline{a} = \overline{a}(g)) : a \in A\} \tag{0.15}$$

is called the *granular reflection of g*. Granular reflections of granules need not be objects found in data set; yet, the results show that they mediate very well between the training and test sets.

The procedure just described for forming a granular reflection of a decision system can be modified as proposed in Artiemjew [1] with help of the procedure of *concept dependent granulation*. In this procedure, the granule $g_{mu}(u, r)$ is modified to the granule

$$g^c_\mu(u, r) = g_\mu(u, r) \cap [u]_d$$

i.e., it is computed relative to the decision class of u.

We collect best results for an exemplary data set, the Australian credit data set, see [23], by various rough set based methods in the table of Fig. 0.1. For a comparison we include in the table of Fig. 0.2 results obtained by some other methods, as given in Statlog. In the table of Fig. 0.3, we give a comparison of performance of rough set classifiers, exhaustive, covering and LEM implemented in RSES [21] system. We begin in the next section with granular classifiers in which granules are induced from the training set.

Fig. 0.1 Best results for Australian credit by some rough set based algorithms

source	*method*	*accuracy*	*coverage*	*MI*
Bazan [6])	$SNAPM(0.9)$	$error = 0.130$	$--$	$--$
Nguyen SH [11]	*simple.templates*	0.929	0.623	0.847
Nguyen SH [11]	*general.templates*	0.886	0.905	0.891
Nguyen SH [11]	*tolerance.gen.templ.*	0.875	1.0	0.891
Wroblewski [25]	*adaptive.classifier*	0.863	$-$	$--$

Fig. 0.2 A comparison of errors in classification by rough set and other paradigms

paradigm	*system/method*	*Austr.credit*
Stat.Methods	*Logdisc*	0.141
Stat.Methods	*SMART*	0.158
Neural Nets	*Backpropagation2*	0.154
Neural Networks	*RBF*	0.145
Decision Trees	*CART*	0.145
Decision Trees	*C4.5*	0.155
Decision Trees	*ITrule*	0.137
Decision Rules	*CN2*	0.204

Fig. 0.3 Train and Test (trn=345 objects, tst=345 objects) ; Australian Credit; Comparison of RSES implemented algorithms exhaustive, covering and LEM

algorytm	*accuracy*	*coverage*	*rule number*	*MI*
$covering(p = 0.1)$	0.670	0.783	589	0.707
$covering(p = 0.5)$	0.670	0.783	589	0.707
$covering(p = 1.0)$	0.670	0.783	589	0.707
$LEM2(p = 0.1)$	0.810	0.061	6	0.587
$LEM2(p = 0.5)$	0.906	0.368	39	0.759
$LEM2(p = 1.0)$	0.869	0.643	126	0.804

0.3 Classification by granules of training objects

We begin with a classifier in which granules computed by means of the rough inclusion μ_L form a granular reflection of the data set and then to this new data set the exhaustive classifier, see [21], is applied.

Procedure of the test

1. *The data set* (U, A, d) *is input*;
2. *The training set is chosen at random. On the training set, decision rules are induced by means of exhaustive, covering and LEM algorithms implemented in the RSES system*;
3. *Classification is performed on the test set by means of classifiers of pt. 2*;
4. *For consecutive granulation radii* r, *granule sets* $G(\mu, r)$ *are found*;
5. *Coverings* $Cov(\mu, r)$ *are found by a random irreducible choice*;
6. *For granules in* $Cov(\mu, r)$, *for each* r, *we determine the granular reflection by factoring attributes on granules by means of majority voting with random resolution of ties*;
7. *For found granular reflections, classifiers are induced by means of algorithms in pt. 2*;

8. *Classifiers found in pt. 7, are applied to the test set*;
9. *Quality measures: accuracy and coverage for classifiers are applied in order to compare results obtained, respectively, in pts. 3 and 8.*

In the table of Fig. 0.4, the results are collected of results obtained after the procedure described above is applied. The classifier applied was exhaustive one; the method was train–and–test. The rough inclusion applied was the Łukasiewicz t–norm induced μ_L of Ch. 6, sect. 4.

Fig. 0.4 Train–and–test; Australian Credit; Granulation for radii r; RSES exhaustive classifier; r=granule radius, tst=test set size, trn=train set size, rulex=rule number, aex=accuracy, cex=coverage

r	tst	trn	$rulex$	aex	cex	MI
nil	345	345	5597	0.872	0.994	0.907
0.0	345	1	0	0.0	0.0	0.0
0.0714286	345	1	0	0.0	0.0	0.0
0.142857	345	2	0	0.0	0.0	0.0
0.214286	345	3	7	0.641	1.0	0.762
0.285714	345	4	10	0.812	1.0	0.867
0.357143	345	8	23	0.786	1.0	0.849
0.428571	345	20	96	0.791	1.0	0.850
0.5	345	51	293	0.838	1.0	0.915
0.571429	345	105	933	0.855	1.0	0.896
0.642857	345	205	3157	0.867	1.0	0.904
0.714286	345	309	5271	0.875	1.0	0.891
0.785714	345	340	5563	0.870	1.0	0.890
0.857143	345	340	5574	0.864	1.0	0.902
0.928571	345	342	5595	0.867	1.0	0.904

We can compare results expressed in terms of the Michalski index MI as a measure of the trade–off between accuracy and coverage; for template based methods, the best MI is 0.891, for covering or LEM algorithms the best value of MI is 0.804, for exhaustive classifier (r=nil) MI is equal to 0.907 and for granular reflections, the best MI value is 0.915 with few other values exceeding 0.900.

What seems worthy of a moment's reflection is the number of rules in the classifier. Whereas for the exhaustive classifier (r=nil) in non–granular case, the number of rules is equal to 5597, in granular case the number of rules can be surprisingly small with a good MI value, e.g., at $r = 0.5$, the number of rules is 293, i.e., 5 percent of the exhaustive classifier size, with the best MI at all of 0.915. This compression of classifier seems to be the most impressive feature of granular classifiers.

It is an obvious idea that this procedure can be repeated until a stable system is obtained to which further granulation causes no change; it is the

procedure of *layered granulation*, see Artiemjew [1]. The table of Fig. 0.5 shows some best results of this procedure for selected granulation radii. As coverage in all reported cases is equal to 1.0, the Michalski index MI is equal to accuracy.

Fig. 0.5 Train–and–test; Australian Credit;(layered–granulation)

r	acc	cov
0.500000	0.436	1.000
0.571429	0.783	1.000
0.642857	0.894	1.000
0.714286	0.957	1.000

This initial, simple granulation, suggests further ramifications. For instance, one can consider, for a chosen value of $\varepsilon \in [0, 1]$, granules of the form

$$g_\mu(u, r, \varepsilon) = \{v \in U : \forall a \in A.|a(u) - a(v)| \leq \varepsilon\} \tag{0.16}$$

and repeat with these granules the procedure of creating a granular reflection and building from it a classifier.

Another yet variation consists in mimicking the performance of the Łukasiewicz based rough inclusion and introducing a counterpart of the granulation radius in the form of the *catch radius*, r_{catch}. The granule is then dependent on two parameters: ε and r_{catch}, and its form is

$$g_\mu(u, \varepsilon, r_{catch}) = \{v \in U : \frac{|\{a \in A : |a(u) - a(v)| \leq \varepsilon}{|A|} \geq r_{catch}\} \tag{0.17}$$

Results of classification by granular classifier induced from the granular reflection obtained by means of granules (0.17) are shown in the table of Fig. 0.6.

0.4 A treatment of missing values

A particular but important problem in data analysis is the treatment of missing values. In many data, some values of some attributes are not recorded due to many factors, like omissions, inability to take them, loss due to some events etc.

Fig. 0.6 ε_{opt}=optimal value of ε, acc=accuracy, cov=coverage. Best $r_{catch} = 0.1428$, $\varepsilon_{opt} = 0.35$: accuracy= 0.8681, coverage=1.0

r_catch	*optimal eps*	*acc*	*cov*
nil	*nil*	0.845	1.0
0	0	0.555073	1.0
0.071428	0	0.83913	1.0
0.142857	0.35	0.868116	1.0
0.214286	0.5	0.863768	1.0
0.285714	0.52	0.831884	1.0
0.357143	0.93	0.801449	1.0
0.428571	1.0	0.514493	1.0
0.500000	1.0	0.465217	1.0
0.571429	1.0	0.115942	1.0

Analysis of systems with missing values requires a decision on how to treat missing values; Grzymala–Busse and Ming Hu [9] analyze nine such methods, among them, *1. most common attribute value*, *2. concept restricted most common attribute value*, *3. assigning all possible values to the missing location*, *4. treating the unknown value as a new valid value*, etc. etc. Their results indicate that methods *3,4* perform very well and in a sense stand out among all nine methods.

We adopt and consider two methods, i.e., *3, 4* from the above mentioned. As usual, the question on how to use granular structures in analysis of incomplete systems, should be answered first.

The idea is to embed the missing value into a granule: by averaging the attribute value over the granule in the way already explained, it is hoped the the average value would fit in a satisfactory way into the position of the missing value.

We will use the symbol $*$, commonly used for denoting the missing value; we will use two methods *3, 4* for treating $*$, i.e, either $*$ is a *don't care* symbol meaning that any value of the respective attribute can be substituted for $*$, hence, $* = v$ for each value v of the attribute, or $*$ is a new value on its own, i.e., if $* = v$ then v can only be $*$.

Our procedure for treating missing values is based on the granular structure $(G(\mu, r), \mathcal{G}, \mathcal{S}, \{a* : a \in A\})$; the strategy $\mathcal{S}$ is the majority voting, i.e., for each attribute a, the value $a^*(g)$ is the most frequent of values in $\{a(u) : u \in g\}$. The strategy $\mathcal{G}$ consists in random selection of granules for a covering.

For an object u with the value of $*$ at an attribute a,, and a granule $g = g(v, r) \in G(\mu, r)$, the question whether u is included in g is resolved according to the adopted strategy of treating $*$: in case $* = don't\ care$, the value of $*$ is regarded as identical with any value of a hence $|IND(u, v)|$ is automatically increased by 1, which increases the granule; in case $* = *$, the granule size is decreased. Assuming that $*$ is sparse in data, majority voting

on g would produce values of a^* distinct from $*$ in most cases; nevertheless the value of $*$ may appear in new objects g^*, and then in the process of classification, such value is repaired by means of the granule closest to g^* with respect to the rough inclusion μ_L, in accordance with the chosen method for treating $*$.

In plain words, objects with missing values are in a sense absorbed by close to them granules and missing values are replaced with most frequent values in objects collected in the granule; in this way the method *3* or *4* in [9] is combined with the idea of a frequent value, in a novel way.

We have thus four possible strategies:

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

We show how effective are these strategies, see Polkowski and Artiemjew [20] by perturbing the data set Pima Indians Diabetes, from UCI Repository [23]. First, in the table of Fig. 0.7 we show results of granular classifier on the non–perturbed (i.e., without missing values) Pima Indians Diabetes data set.

Fig. 0.7 10-fold CV; Pima; exhaustive algorithm, r=radius,macc=mean accuracy,mcov=mean coverage

r	*macc*	*mcov*
0.0	0.0	0.0
0.125	0.0	0.0
0.250	0.6835	0.9956
0.375	0.7953	0.9997
0.500	0.9265	1.0
0.625	0.9940	1.0
0.750	1.0	1.0
0.875	1.0	1.0

We now perturb this data set by randomly replacing 10 percent of attribute values in the data set with missing $*$ values. Results of granular treatment in case of Strategies A,B,C,D in terms of accuracy are reported in the table of Fig. 0.8. As algorithm for rule induction, the exhaustive algorithm of the RSES system has been selected. 10–fold cross validation (CV–10) has been applied.

Fig. 0.8 Accuracies of strategies A, B, C, D. 10-fold CV; Pima Indians; exhaustive algorithm; r=radius,maccA=mean accuracy of A,maccB=mean accuracy of B, maccC=mean accuracy of C, maccD=mean accuracy of D

r	*maccA*	*maccB*	*maccC*	*maccD*
0.250	0.0	0.0	0.0	0.645
0.375	0.0	0.0	0.0	0.7779
0.500	0.0	0.0	0.0	0.9215
0.625	0.5211	0.5831	0.5211	0.9444
0.750	0.7705	0.7769	0.7705	0.9994
0.875	0.9407	0.9407	0.9407	0.9987

Strategy A reaches the accuracy value for data with missing values within 94 percent of the value of accuracy without missing values (0.9407 to 1.0) at the radius of .875. With Strategy B, accuracy is within 94 percent from the radius of .875 on. Strategy C is much better: accuracy with missing values reaches 99 percent of accuracy in no missing values case from the radius of .625 on. Strategy D gives results slightly better than C with the same radii.

We conclude that the essential for results of classification is the strategy of treating the missing value of $*$ as $* = *$ in both strategies C and D; the repairing strategy has almost no effect: C and D differ very slightly with respect to this strategy.

0.5 Granular rough mereological classifiers using residuals

Rough inclusions used in sects. 0.2 – 0.4 in order to build classifiers do take, to a certain degree, into account the distribution of values of attributes among objects, by means of the parameters ε and the catch radius r_{catch}.

The idea that metrics used in classifier construction should depend locally on the training set is, e.g., present in classifiers based on the idea of nearest neighbor, see, e.g., a survey in Polkowski [18]: for nominal values, the metric VDM (Value Difference Metric) in Stanfill and Waltz [22] takes into account conditional probabilities $P(d = v|a_i = v_i)$ of decision value given the attribute value, estimated over the training set Trn, and on this basis constructs in the value set V_i of the attribute a_i a metric $\rho_i(v_i, v_i') = \sum_{v \in V_d} |P(d = v|a_i = v_i) - P(d = v|a_i = v_i')|$. The global metric is obtained by combining metrics ρ_i for all attributes $a_i \in A$ according to one of many-dimensional metrics, e.g., Minkowski metrics, see Ch. 2.

This idea was also applied to numerical attributes in Wilson and Martinez [24] in metrics $IVDM$ (Interpolated VDM) and $WVDM$ (Windowed VDM).

A modification of the $WVDM$ metric based again on the idea of using probability densities in determining the window size was proposed as $DBVDM$ metric.

In order to construct a measure of similarity based on distribution of attribute values among objects, we resort to residual implications, see Ch.4, sect. 3. As shown in Polkowski [14], $\Rightarrow_T$ does induce a rough inclusion on the interval $[0,1]$

$$\mu_{\rightarrow_T}(u,v,r) \text{ if and only if } x \Rightarrow_T y \geq r \tag{0.18}$$

This rough inclusion can be transferred to the universe U of an information system; to this end, first, for given objects u, v, and $\varepsilon \in [0,1]$, factors

$$dis_\varepsilon(u,v) = \frac{|\{a \in A : |a(u) - a(v)| \geq \varepsilon\}|}{|A|} \tag{0.19}$$

and

$$ind_\varepsilon(u,v) = \frac{|\{a \in A : |a(u) - a(v)| < \varepsilon\}|}{|A|} \tag{0.20}$$

are introduced.

The weak variant of rough inclusion $\mu_{\rightarrow_T}$ is defined, see Polkowski [14], as

$$\mu^*_T(u,v,r) \text{ if and only if } dis_\varepsilon(u,v) \rightarrow_T ind_\varepsilon(u,v) \geq r \tag{0.21}$$

Particular cases of this similarity measure induced by, respectively, t–norm min, t–norm $P(x,y)$, and t–norm L are, see Ch. 6, sect. 7

1. *For* $T = M(x,y) = min(x,y)$, $x \Rightarrow_{min} y$ *is* y *in case* $x > y$ *and* 1 *otherwise, hence,* $\mu^*_{min}(u,v,r)$ *if and only if* $dis_\varepsilon(u,v) > ind_\varepsilon(u,v) \geq r$ *with* $r < 1$ *and* 1 *otherwise*;
2. *For* $t = P$, *where* $P(x,y) = x \cdot y$, $x \Rightarrow_P y = \frac{y}{x}$ *when* $x \neq 0$ *and* 1 *when* $x = 0$, *hence,* $\mu^*_P(u,v,r)$ *if and only if* $\frac{ind_\varepsilon(u,v)}{dis_\varepsilon(u,v)} \geq r$ *with* $r < 1$ *and* 1 *otherwise*;
3. *For* $t = L$, $x \Rightarrow_L y = min\{1, 1 - x + y\}$, *hence,* $\mu^*_L(u,v,r)$ *if and only if* $1 - dis_\varepsilon(u,v) + ind_\varepsilon(u,v) \geq r$ *with* $r < 1$ *and* 1 *otherwise.*

These similarity measures will be applied in building granules and then in data classification. Tests are done with the Australian credit data set; the results are validated by means of the 5–fold cross validation (CV–5). For each of t–norms: M, P, L, three cases of granulation are considered, viz.,

1. *Granules of training objects* (GT);
2. *Granules of rules induced from the training set* (GRT);
3. *Granules of granular objects induced from the training set* (GGT).

In this approach, training objects are made into granules for a given ε. Objects in each granule g about a test object u, vote for decision value at u as follows:

for each decision class c, the value

$$p(c) = \frac{\sum_{\text{training object v in g falling in c}} w(u,v)}{\text{size of c in training set}} \tag{0.22}$$

is computed where the weight $w(u,v)$ is computed for a given t–norm T as

$$w(u,v) = dis_{\varepsilon}(u,v) \rightarrow_T \ ind_{\varepsilon}(u,v) \tag{0.23}$$

The class c* assigned to u is the one with the largest value of p.

Weighted voting of rules in a given granule g for decision at test object u goes according to the formula $d(u) = argmaxp(c)$, where

$$p(c) = \frac{\sum_{\text{rule in g pointing to c}} w(u,r) \cdot support(r)}{\text{size of c in training set}} \tag{0.24}$$

where weight $w(u,r)$ is computed as

$$dis_{\varepsilon}(u,r) \rightarrow_T \ ind_{\varepsilon}(u,r) \tag{0.25}$$

The optimal (best) results in terms of accuracy of classification are collected in the table of Fig. 0.9.

Fig. 0.9 5-fold CV; Australian; residual metrics. met=method of granulation, T=t–norm, ε_{opt}=optimal ε, macc=mean accuracy,mcov=mean coverage

met	*T*	ε_{opt}	*macc*	*mcov*
GT	*M*	0.04	0.848	1.0
GT	*P*	0.06	0.848	1.0
GT	*L*	0.05	0.846	1.0
GRT	*M*	0.02	0.861	1.0
GRT	*P*	0.01	0.851	1.0
GGT	*M*	0.05	0.855	1.0
GRT	*P*	0.01	0.852	1.0

0.6 Granular rough mereological classifiers with modified voting parameters

An interesting modification of voting schemes of above sections was proposed and tested in Artiemjew [4]. It consists in weighted voting with modified weight computing scheme, viz., the procedure is now as follows. It is considered in [4] in five cases, of which we include here two with best results, i.

e., cases 4 and 5. For each attribute a, each training object v, and each test object u, we denote with the symbol $\rho_{trn}(u,v)$ the quotient

$$\frac{||a(u)-a(v)||}{|max_{training\ set}a - min_{training\ set}a|} \tag{0.26}$$

where $max_{training\ set}a$, $min_{training\ set}a$ are, respectively, the maximal and the minimal values of the attribute a over the training set; the symbol $||.||$ stands for the Euclidean distance in attribute value spaces. Augmented values of weights are computed in cases 4, 5 in two variants: (a) when $\rho_{trn}(u,v) \geq \varepsilon$ and (b) when $\rho_{trn}(u,v) \leq \varepsilon$.

We have in Case 4

$$w(u,v) = \begin{cases} (a)\ w(u,v) + \rho_{trn}(u,v)\cdot\varepsilon + ||a(u)-a(v)|| \\ (b)\ w(u,v) + rho_{trn}(u,v)\cdot\varepsilon \end{cases} \tag{0.27}$$

and in Case 5

$$w(u,v) = \begin{cases} (a)\ w(u,v) + \rho_{trn}(u,v) \\ (b)\ w(u,v) + \rho_{trn(u,v)}\cdot\varepsilon \end{cases} \tag{0.28}$$

Voting procedure consists in computing values of parameters

$$p_1 = \frac{\sum v \text{ in positive class } w(u,v)}{\text{cardinality of positive class}} \tag{0.29}$$

and respectively, p_2 by means of (0.29) with 'positive' replaced by 'negative'; when $p_1 < p_2$, the test object u is classified into the positive class, otherwise it is classified into the negative class. Optimal results are shown in the table of Fig. 0.10. As coverage is 1.0 in each case, we do not show it in the table. For comparison, we insert also results by RSES exhaustive algorithm and by RSES implemented k–NN method.

Fig. 0.10 Parameterized voting. CV–5. ε_{opt}=optimal ε for maxacc, maxacc=max max fold accuracy, minacc=min max fold accuracy

$Case$	ε_{opt}	$maxacc$	$minacc$
$Case4$	0.62	0.905	0.861
$Case5$	$0.35-0.37$	0.906	0.880
$RSESexh$	–	0.862	0.819
$RSESk-NN$	–	0.884	0.841

References

1. Artiemjew P. (2007) Classifiers from granulated data sets: Concept dependent and layered granulation. In: Proceedings RSKD'07. Workshop at ECML/ PKDD'07, Warsaw University Press, Warsaw, pp 1–9
2. Artiemjew P. (2008) On classification of data by means of rough mereological granules of objects and rules. Lecture Notes in Artificial Intelligence 5009, Springer Verlag, Berlin, pp 221–228
3. Artiemjew P. (2008) Rough mereological classifiers obtained from weak rough set inclusions. Lecture Notes in Artificial Intelligence 5009, Springer Verlag, Berlin, pp 229–236
4. Artiemjew P.(2009) On Strategies of Knowledge Granulation with Applications to Decision Systems. L.Polkowski (supervisor). PhD Dissertation. Polish–Japanese Institute of Information Technology, Warszawa, 355 pages
5. Arkin H., Colton R.R. (1970) Statistical Methods. Barnes and Noble, New York
6. Bazan J. G. (1998) A comparison of dynamic and non–dynamic rough set methods for extracting laws from decision tables. In: Polkowski L., Skowron A. (Eds.)(1998) Rough Sets in Knowledge Discovery 1. Physica Verlag, Heidelberg, pp 321–365
7. Bazan J. G., Nguyen H. S., Nguyen S. H., Synak, P., Wróblewski J. (2000) Rough set algorithms in classification problems. In: Polkowski L., Tsumoto S., Lin T. Y. (Eds.)(2000) Rough Set Methods and Applications. New Developments in Knowledge Discovery in Information Systems. Physica Verlag, Heidelberg, pp 49–88
8. Bruning J.L., Kintz B.L. (1997) Computational Handbook of Statistics. 4th ed. Allyn and Bacon, Columbus, OH
9. Grzymala–Busse J. W., Ming Hu (2000) A comparison of several approaches to missing attribute values in data mining. Lecture Notes in Artificial Intelligence 2005, Springer Verlag, Berlin, pp 378–385
10. Michalski R. (1990) Pattern recognition as rule–guided inductive inference. IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 2 (4), pp 349–361
11. Nguyen Sinh Hoa (2000) Regularity analysis and its applications in Data Mining. In: Polkowski L., Tsumoto S., Lin T. Y. (Eds.)(2000) Rough Set Methods and Applications. New Developments in Knowledge Discovery in Information Systems. Physica Verlag, Heidelberg, pp 289–378
12. Polkowski L. (2005) Formal granular calculi based on rough inclusions (a feature talk). In: Proceedings of IEEE 2005 Conference on Granular Computing,GrC05, Beijing, China, July 2005. IEEE Press, pp 57–62
13. Polkowski L. (2006) A model of granular computing with applications (a feature talk). In: Proceedings of IEEE 2006 Conference on Granular Computing, GrC06, Atlanta, USA, May 2006. IEEE Press, pp 9–16
14. Polkowski L. (2007) Granulation of knowledge in decision systems: The approach based on rough inclusions. The method and its applications (a plenary talk). In: Proceedings

RSEISP 07, Warsaw, Poland, June 2007. Lecture Notes in Artificial Intelligence 4585. Springer Verlag, Berlin, pp 271–279
15. Polkowski L. (2008) On the idea of using granular rough mereological structures in classification of data. Lecture Notes in Artificial Intelligence 5009. Springer Verlag, Berlin, pp 213–220
16. Polkowski L. (2008) A Unified approach to granulation of knowledge and granular computing based on rough mereology: A Survey. In: Pedrycz W., Skowron A., Kreinovich V. (Eds.)(2008) Handbook of Granular Computing. John Wiley and Sons Ltd., Chichester, UK, Chapter 16
17. Polkowski L. (2009) Granulation of Knowledge: Similarity Based Approach in Information and Decision Systems. In: Meyers R. A. (Ed.) (2009) Encyclopedia of Complexity and System Sciences, Springer Verlag, Berlin, Article 00 788
18. Polkowski L. (2009) Data-mining and Knowledge Discovery: Case Based Reasoning, Nearest Neighbor and Rough Sets. In: Meyers R. A. (Ed.) (2009) Encyclopedia of Complexity and System Sciences, Springer Verlag, Berlin, Article 00 391
19. Polkowski L., Artiemjew P. (2007) On granular rough computing: Factoring classifiers through granular structures. Lecture Notes in Artificial Intelligence 4585. Springer Verlag, Berlin, pp 280–290
20. Polkowski L., Artiemjew P. (2007) On granular rough computing with missing values. Lecture Notes in Artificial Intelligence 4585, Springer Verlag, Berlin, pp 271–279
21. RSES. available at: http://mimuw.edu.pl/logic/ rses/; last entered 01. 04. 2011
22. Stanfill C., Waltz D. (1986) Toward memory–based reasoning. Communications of the ACM 29, pp 1213–1228
23. UCI (University of California at Irvine) Repository. Available at: http://archive.ics.uci.edu./ml/; last entered 01. 04. 2011
24. Wilson D. R., Martinez T. R. (1997) Improved heterogeneous distance functions. Journal of Artificial Intelligence Research 6, pp 1–34
25. Wróblewski J. (2004) Adaptive aspects of combining approximation spaces. In: Pal S.K., Polkowski L., Skowron A. (Eds.) (2004) Rough Neural Computing. Techniques for Computing with Words, Springer Verlag, Berlin, pp 139–156