Mushroom Plant Analysis through Reduct Technique

Ayesha Butalia Divya Shah Dr. R.V Dharaskar

ABSTRACT

The issues of Real World are Very large data sets, Mixed types of data (continuous valued, symbolic data), Uncertainty (noisy data), Incompleteness (missing, incomplete data), Data change, Use of background knowledge etc. Lot of knowledge related to the application can be generated through these large data sets.
Rough set is the methodology which can be used to deduce rules from these data sets.

The main goal of the rough set analysis is induction of approximations of concepts [4]. Rough sets constitute a sound basis for KDD. It offers mathematical tools to discover patterns hidden in data [4] and hence used in the field of data mining.

Rough Sets does not require any preliminary information as Fuzzy sets require membership values or probability is required in statistics. Hence this is its specialty.

Two novel algorithms to find optimal Reducts of condition attributes based on the relative attribute dependency, out of which the first algorithms gives simple Reduct whereas the second one gives the Reduct with minimum attributes,

This project highlights on the case study of mushroom which consists of twenty two attributes depending on which the decision is taken whether the mushroom plant is edible or poisonous. The technique of Reduct is very useful as when tested, through the algorithms, the twenty one attributes, excluding the decision attribute gets reduced to two to three attributes.

1. INTRODUCTION

Usually the primary considerations of traditional computing are precision, certainty, and rigor. We distinguish this as "hard" computing. In contrast, the principal notion in soft computing [6] is that precision and certainty carry a cost and that computation, reasoning, and partial truth for obtaining low-cost solutions. This leads to the remarkable human ability of understanding distorted speech, deciphering sloppy handwriting, comprehending the nuances of natural language, summarizing text, recognizing and classifying images, driving a vehicle in dense traffic, and, more generally, making rational decisions in an environment of uncertainty and imprecision. The challenge, then, is to exploit the tolerance for imprecision by devising methods of computation that lead to an acceptable solution at low cost.
The example taken here is of mushroom plant which has twenty two attributes and 8000 records. The data set is taken from UCI Machine learning site. Mushroom records drawn from The Audubon Society Field Guide to North American Mushrooms (1981). G. H. Lincoff (Pres.), New York: Alfred A.Knopf. The Donor: Jeff Schlimmer

Machine learning overlaps heavily with statistics. In fact, many machine learning algorithms have been found to have direct counterparts with statistics. As a broad subfield of artificial intelligence, Machine learning is concerned with the development of algorithms and techniques that allow computers to "learn". At a general level, there are two types of learning:, inductive and deductive. Inductive machine learning methods create computer programs by extracting rules and patterns out of massive data sets. It should be noted that although pattern identification is important to Machine Learning, without rule extraction a process falls more accurately in the field of data mining.

Hence Rough Sets can be used as a framework for datamining especially in the areas of soft computing where exact data is not required and in some areas where approximate data can be of great help.

2. PRESENT THEORY AND PRACTICES

Decision Tree Induction, which is a flow-chart-like tree structure, can do classification, in which internal node denotes a test on an attribute, branch represents an outcome of the test and leaf nodes represent class labels or class distribution. This requires usage of heavy data structures to construct trees and the induction algorithm has extensive calculations on each node which makes it time consuming. In contrast, Bayesian Theorem can also be used for Classification, but the practical difficulty is that it requires initial knowledge of many probabilities and has significant computational cost.
Neural Networks have been used successfully for Classification but suffer somewhat in that the resulting network is viewed as a black box and no explanation of the results is given. This lack of explanation inhibits confidence, acceptance and application of results. It also noted the problem that neural network suffered from long training time, difficult to understand the learned function (weights), and not easy to incorporate domain knowledge.

Finding the minimal subsets (Reducts) of attributes (for feature Reduction) is NP-hard but discernibility matrix is used to reduce the computation intensity. An attribute-oriented rough sets technique reduces the computational complexity of learning processes and eliminates the unimportant or irrelevant attributes so that the knowledge discovery in databases or in experimental data sets can be efficiently learned. The theory of Rough sets has been studied in the

context of expert systems, decision support systems, inductive reasoning [11], pattern recognition, and machine learning [13]. Using rough sets has been shown to be very effective for revealing relationships within imprecise data, discovering dependencies among objects and attributes, evaluating the classificatory importance of attributes, removing data redundancies (and thus reducing the size of information systems), and generating decision rules.

To sum up, the exhaustive search approach is infeasible in practice; and the heuristic search approach can reduce the search time significantly, but will fail on hard problems or cannot find the best subset of features.

The author has compared this approach with the Rough Set for data mining by testing the datasets of Cars, Medical, and Mushroom. Each method has some advantages. In Rough Set method, one of the major problems in data mining i.e. noise is handled well. The accuracy of the datasets mentioned above is measured. The Accuracy of the three datasets is much better in Rough Set compared to the Decision tree. So the conclusion is that Rough Set Approach is a good one for Data mining.

4. METHODOLOGIES USED

4.1 Algorithm to calculate Decision Rules

Step1: Find out the decision attribute

Step2: Create the Concept class according to the decision attribute, i.e. find out all the tuples related to that attribute, say concept Y1.

Step3: Select the Condition attributes

Step4: Find out the DES and the Equivalence classes according to the decision and condition attributes, i.e. the indiscernibility relation, say X_1, $X_2 \ldots X_n$.

Step5: If X_1 Union $Y_1 = X_1$, then we say that the concept definitely holds. Similarly for $X_2 \ldots X_n$.

Recently, rough set theory has been employed to select feature subset. In the rough set community, feature selection algorithms are attribute-Reduct oriented, that is, finding optimal Reduct of condition attributes of a given data set. Two main approaches to finding attribute Reducts are recognized as discernibility function-based and attribute dependency-based. These algorithms, however, suffer from intensive computations of either discernibility functions for the former or positive regions for the latter, although some computation efficiency improvement has been made in some new developments.

In rough set theory, the data is collected in a table, called decision table. Rows of the decision table correspond to instances, and columns correspond to features (or attributes). All attributes are recognized into two groups: conditional attributes set C as input and decision attributes set D as output.

Assume P ⊂ CUD and Q ⊂ CUD, the positive region of Q with respect to P, denoted POS_p (Q), and is defined as

$$def \bigcup_{x \in U / IND(Q)} \underline{P}X ,$$

where $\underline{P}X$ is the lower approximation of X and U/IND(Q) is the equivalent partition induced by Q. The positive region of Q with respect to P contains all objects in U that can be classified using the information contained in P. With this definition, the degree of dependency of Q from P, denoted y_p (Q), is defined as

$$\gamma_P(Q) = def \frac{POS_P(Q)}{U} ,$$

where |X| denotes the cardinality of the set X.

The degree of attribute dependency provides a measure how an attributes subset is dependent on another attributes subset. y_p (Q) =1 means that Q totally depends on P, y_p(Q) =0 indicates that Q is totally independent from P, while O<y_p (Q) <I denotes a partially dependency of Q from P. Particularly, assume P ⊂ C, then y_p (D) can be used to measure the dependency of the decision attributes from a conditional attributes subset. The task of rough set attribute Reduction is to find a subset of the conditional attributes set, which functions as the original conditional attributes set without loss of classification capability. This subset of the conditional attributes set is called Reduct, and defined as follows [10].

R ⊂ C is called a Reduct of C, if and only if POS_R (D) = POS_C (D), or, Y_R (D) = Y_c (D). A Reduct R of C is called a minimum Reduct of C if ∀Q C R, Q is not a Reduct of C.

A Reduct R of C has the same expressiveness of instances as C with respect to D. A decision table may have more than one Reduct. Anyone of them can be used to replace the original condition attributes set. Finding all the Reducts from a decision table, however, is NP-hard. Thus, a natural question is which Reduct is the best. Without domain knowledge, the only source of information to select the Reduct is the contents of the decision table. For example, the number of attributes can be used as the criteria and the best Reduct is the one with the smallest number of attributes. Unfortunately, finding the Reduct with the smallest number of attributes is also NP-hard.

Some heuristic approaches for finding a good enough Reduct have been proposed. A recent algorithm, called QuickReduct, was developed by Shen and Chouchoulas in 2002. QuickReduct is a filter approach of feature selection and a forward searching hill climber. QuickReduct initializes the candidate Reduct R as an empty set, and attributes are added to R incrementally using the following heuristic: the next attribute to be added to R is the one with the highest significance to R with respect to the decision attributes. R is increased until R becomes a Reduct. The basic idea behind this algorithm is that the degree of attribute dependency is monotonically increasing. There are two

problems with this algorithm, however. First, it is not guaranteed to yield the best Reduct with the smallest number of attributes. Second, to calculate the significance of attributes, the discernibility function and positive regions must be computed, which is inefficient and time-consuming. A variant of QuickReduct, called QuickReduct II is also a filter algorithm, but performs the backward elimination using the same heuristic [16].

Problems in Discernibility Matrix and Discernibility Functions:
Excessive computations are required to implement Reduct from Discernibility functions.

4.2 Relative Attribute Dependency Based on Rough Set Theory (Used to calculate Reducts)

In order to improve the efficiency of algorithms for finding optimal Reducts of condition attributes, we proposed a new definition of attribute dependency, called relative attribute dependency, with which we showed a sufficient and necessary condition of the optimal Reduct of conditional attributes [4]. The relative attribute dependency degree can be calculated by counting the distinct instances of the subset of the data set, instead of generating discernibility functions or positive regions. Thus the computation efficiency of finding minimum Reduct is highly improved.

Let $Q \subseteq C$. The degree of relative dependency, denoted K_Q (D), of Q on D over U is defined as

$$K_Q(D) = \frac{|\Pi_Q(U)|}{|\Pi_{Q \cup D}(U)|}$$

where $|\Pi_x|$ (U) is actually the number of equivalence classes in U/IND(X).

Algorithm 1 - Brute-force backward elimination

The first algorithm assumes the entire condition attribute set as the Reduct, and then eliminates the redundant attributes until the remaining attributes form a Reduct. The algorithm is described as follows.

Input: Consistent decision table U, condition attributes set C, decision attributes set D

Output: R - a minimum Reduct of condition attributes set C with respect to D in U Procedure:

1. R ← C
2. For each attribute q in C Do
3. If $K_{R-\{q\}}$ (D) = 1 Then R ← R – {q}
//remove if the relative dependency is 1
4. Return R

One may note that the outcome of Algorithm 1 is an arbitrary Reduct of the condition attributes set C. Which Reduct is generated depends on the order of attributes that are checked for dispensability in Step 2 of the algorithm. Some authors propose algorithms for constructing the best Reduct, but what is the best depends on how to define the criteria, such as the number of attributes in the Reduct. In the absence of criteria, the only source of information to select the Reduct is the content of the data table. A common metric of data content is information entropy contained in the data items. The following Algorithm 2 utilizes the information entropy conveyed in the attributes as a heuristic of selecting attributes to be eliminated.

B. Algorithm 2 -Attribute information entropy based backward elimination

Given the partition by D, U/IND(D), of U, the entropy, or expected information based on the partition by q Є C, U/ q, of U, is given by

$$E(q) = \bigcup_{Y \in U/q} \frac{|Y|}{|U|} I(q/Y)$$

where

$$I(q/Y) = -\sum_{x \in U/IND(D)} \frac{|Y \cap X|}{|Y|} \log_2 \frac{|Y \cap X|}{|Y|}$$

Thus the entropy E(q) can be represented as

$$E(q) = -\frac{1}{|U|} \sum_{X \in U/IND(D)} \sum_{Y \in U/q} |X \cap Y| \log_2 \frac{|X \cap Y|}{|Y|}$$

Using above representation, we have the following algorithm:

Input: Consistent decision table U, condition attributes set C, decision attributes set D

Output: R -- a minimum Reduct of condition attributes set C with respect to D in U Procedure:

1. R ← C, Q ← empty
2. For each attribute q Є C Do
3. Compute the entropy E(q) of q
4. Q ← Q ∪ {<q,E(q)>}
5. While Q ≠ Φ do // select attribute with maximum entropy
6. q ← arg max{E(p)| <p,E(p)> Є Q}
7. Q ← Q- {<q,E(q)>} // test if redundant
8. If $K_{R-\{q\}}$ (D) = 1 Then
9. R ← R-(q} //remove q
10. Return R

The outcome of Algorithm I and Algorithm 2 is a minimum Reduct of C with respect to D in U. The time complexity of Algorithm I is O (|C||U|), while the time complexity of Algorithm 2 is O (|C||U|$\log_2$|U|), where |C| is the number of condition attributes, and |U| is the number of tuples in the decision table.

5. CASE STUDY

From Audobon Society Field Guide; mushrooms described in terms of physical characteristics; classification: poisonous or edible [10].

Data Set Character-istics	Multivariate	No. of Instances	8124	Area:	Life
Attribute	Categorical	No. of Attributes	22	Date Donat-ed	1987-04-27
Associated Tasks:	Classification	Missing Values	Yes	No. of Web Hits	12845

5.1 Data Set Information: [10]

This data set includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family (pp. 500-525). Each species is identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. This latter class was combined with the poisonous one. The Guide clearly states that there is no simple rule for determining the edibility of a mushroom; no rule like ``leaflets three, let it be'' for Poisonous Oak and Ivy.

5.2 Attribute Information

1. cap-shape: bell=b, conical= c, convex=x, flat=f, knobbed=k, sunken=s
2. cap-surface: fibrous=f, grooves=g, scaly=y, smooth=s
3. cap-color: brown=n, buff=b, cinnamon=c, gray-g, green=r, pink=p, purple=u, red=e, white=w, yellow=y
4. bruises?: bruises=t, no=f,
5. odor: almond=a, anise=l, creosote=c, fishy=y, foul=f, musty=m, none=n, pungent=p, spicy=s
6. gill-attachment: attached=a, descending=d, free=f, notched=n
7. gill-spacing; close=c, crowded=w, distant=d
8. gill-size: broad=b, narrow=n
9. gill-color: black=k, brown=n, buff=b, chocolate=h, gray=g, green=r, o, pink=p, purple=u, red=e, white=w, yellow=y.
10. stalk-shape: enlarging=e, tapering=t
11. stalk-root: bulbous=b, club=c, cup=u, equal=e, rhizomorphs=z, rooted=r, missing=?
12. stalk-surface-above-ring: fibrous=f, scaly=y, silky=k, smooth=s
13. stalk-surface-below-ring: fibrous=f, scaly=y, silky=k, smooth=s
14. stalk-color-above-ring: brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p ,red=e, white=w, yellow=y
15. stalk-color-below-ring: brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, red=e, white=w, yellow=y
16. veil-type: partial=p, universal=u
17. veil-color: brown=n, orange=o, white=w, yellow=y
18. ring-number: none=n, one=o, two=t
19. ring-type: cobwebby=c, evanescent=e, flaring=f, large=l, none=n, pendant=p, sheathing=s, zone=z
20. spore-print-color: black=k, brown=n, buff=b, chocolate=h, green=r, orange=o, purple=u, white=w, yellow=y
21. population: abundant=a, clustered=c, numerous=n, scattered=s, several=v, solitary=y
22. habitat: grasses=g, leaves=l, meadows=m, paths=p, urban=u, waste=w, woods=d

This data set includes the decision attribute which says whether the plant is edible or poisonous. The number of records is 8000. Analysis told that more the number of records better is the result. But the variation is too less. The motive is to found out whether these 22 attributes are indispensable or not. Using the algorithm I that is, Brute-force backward elimination, using 1000 records, as the time taken for 8000 would take half a day, the Reduct was {cap-shape, cap-surface, cap-color, cap-bruises, cap-odor}. The result with the Algorithm 2, that is, Attribute information entropy based backward elimination is {habitat, ring-type, population, cap-color}. The time taken for Brute-force backward elimination (Algorithm 1) is less as compared to Attribute information entropy based backward elimination (Algorithm 2), but the results are more accurate in Algorithm 2 as information gain and entropy is considered. So in case of Algorithm 1, the output is of 5 attributes, and in case of Algorithm 2, the output is of 4 attributes. If we take 8000 records into consideration, the output is of 3 and 2 attributes in Algorithm I and Algorithm II respectively.

Once the attributes are selected, the decision as which value of that attribute is important to make that decision. For example, if we consider Reduct set through first technique, and apply the algorithm of Rough set as designed in section 4.1, we will get the answer as "IF (C_CAPSHAPE==b) AND (C_CAPCURFACE==f) AND (C_CAPCOLOR==g) THEN DEFINITELY THE DECISION IS e" where e is edible, and "IF (C_CAPSHAPE==b) AND (C_CAPCURFACE==f) AND (C_CAPCOLOR==g) THEN DEFINITELY THE DECISION IS p" where p is poisonous. Similarly, if we consider the Reduct set through second technique, the decision rule will be "IF (C_CAPSHAPE==b) AND (C_CAPCURFACE==f) AND (C_CAPCOLOR==g) THEN DEFINITELY THE DECISION IS e" for edible and "IF (C_CAPSHAPE==b) AND (C_CAPCURFACE==f) AND (C_CAPCOLOR==g) THEN DEFINITELY THE DECISION IS p" for poisonous.

5.2 Example of MUSHROOM

The csv file consists of 10 records as follows:

5.2.1 Reduct technique Through Algorithm I

Consider C={CAPSHAPE, CAPSURFACE,,CAPCOLOR, BRUISES, ODOR, GILLATTACHMENT, GILLSPACING, GILLSIZE, GILLCOLOR, STALKSHAPE, STALKROOT, STALKSURFACEABOVERING,_STALKSURFACEBELOWRING, STALKCOLORABOVERING, STALKCOLORBELOWRING, VEILTYPE, VEILCOLOR, RINGNUMBER, RINGTYPE, SPOREPRINTCOLOR, POPULATION, HABITAT,DECISION}

-D={decision d}
-card (POSC(D)) = 9 = card (U)
-Removing attribute CAPSHAPE
-card (POS_{T1}(D)) = 9
where
T1 = {CAPSURFACE,,CAPCOLOR, BRUISES, ODOR, GILLATTACHMENT, GILLSPACING, GILLSIZE, GILLCOLOR, STALKSHAPE, STALKROOT, STALKSURFACEABOVERING,_STALKSURFACEBELOWRING, STALKCOLORABOVERING, STALKCOLORBELOWRING, VEILTYPE, VEILCOLOR, RINGNUMBER, RINGTYPE, SPOREPRINTCOLOR, POPULATION, HABITAT}
-Since dependency = card (POS_{T1}(D))/card(U)= 9/9=1, CAPSURFACE can be removed.
-card (POS_{T2}(D)) = 9
where T2 = {BRUISES, ODOR, GILLATTACHMENT, GILLSPACING, GILLSIZE, GILLCOLOR, STALKSHAPE, STALKROOT, STALKSURFACEABOVERING,_STALKSURFACEBELOWRING, STALKCOLORABOVERING, STALKCOLORBELOWRING, VEILTYPE, VEILCOLOR, RINGNUMBER, RINGTYPE, SPOREPRINTCOLOR, POPULATION, HABITAT}

- Hence CAPCOLOR can also be removed.
- Similarly all dependency is 1 until we reach SPOREPRINTCOLOR in the set.
- So, card (POS_{T3}(D)) = 6
  - Where T3 = { SPOREPRINTCOLOR, POPULATION, HABITAT}
  - k= 0.6666667
- Now we remove POPULATION from the set
- card (POS_{T4}(D)) = 9
  - where T4 = {SPOREPRINTCOLOR, HABITAT}
  - k=1
- Hence POPULATION is extraneous
- Now lastly, we remove HABITAT from the set
- card (POS_{T5}(D)) = 7
  - where T5= {SPOREPRINTCOLOR}
  - k= 0.7777778 which is not 1
  - Hence HABITAT is not extraneous.
- So finally the Reduct is {SPOREPRINTCOLOR, HABITAT}

The above algorithm fails as all combinations are not tried,
Hence may not give correct solution.

6 Reduct technique Through Algorithm II
7 Here the entropy method is used.
8 The entropy of each attribute is calculated which is as follows:
8.2 C_HABITAT 0.7449736116185914
8.3 C_CAPSURFACE 0.7142300931534918
8.4 C_CAPCOLOR 0.6960736118322672
8.5 C_STALKSURFACEABOVERING 0.6869431338648536
8.6 C_STALKSURFACEBELOWRING 0.6869431338648536
8.7 C_STALKSURFACEABOVERING 0.6869431338648536
8.8 C_SPOREPRINTCOLOR 0.5283208335737187
8.9 C_VEILCOLOR 0.5080479168135689
8.10 C_POPULATION 0.5066555557692314
8.11 C_CAPSHAPE 0.4755500001068379
8.12 C_GILLATTACHMENT 0.46683379664351965
8.13 C_STALKCOLORBELOWRING 0.46683379664351965
8.14 C_GILLSPACING 0.42071851894587037
8.15 C_RINGNUMBER 0.42071851894587037
8.16 C_GILLCOLORCAPCOLOR 0.31027569448450865
8.17 C_ODOR 0.1453634260594734
8.18 C_BRUISES 0.7142300931534918
8.19 C_GILLSIZE 0.0
8.20 C_STALKSHAPE 0.0
8.21 C_STALKROOT 0.0
8.22 C_VEILTYPE 0.0
8.23 C_RINGTYPE 0.0

The maximum entropy is of C_HABITAT, so it is removed first and then dependency is calculated,
It is 1.0
Hence it is removed.
Likewise all are removed until C_RINGTYPE.

Hence the reduct is C_RINGTYPE

REFERENCES

[1] Z.Pawlak, "Rough Sets", International Journal of Computer and Information Sciences, Vol.11, 341-356(1982).
[2] Z.Pawlak, Rough Sets – Theoretical Aspect of Reasoning about Data, Kluwer Academic Publishers (1991).
[3] A.Skowron, N. Zhong, and N. Cercone Computational Intelligence, An International Journal, Special Issue on Rough Sets, Data Mining, and Granular Computing.
[4] J. Grzymala-Busse, R. Swiniarski, N. Zhong, and Z. Ziarko International Journal of Applied Mathematics and Computer Science, Special Issue on Rough Sets and Its Applications.
[5] H. S Nguyen and S.H Nguyen, "Discretization Methods in Data Mining", Vol.1, 451-482, Physica-Verlag (1998).
[6] T.Y Lin, Journal of Intelligent Automation and Soft Computing, Vol.2, No. 2, Special Issue on Rough Sets (1996).
[7] T.Y Lin, International Journal of Approximate Reasoning, Vol.15, No. 4, Special Issue n Rough Sets (1996).
[8] Z. Ziarko Computational Intelligence, An International Journal, Vol.11, No.2, Special Issue on Rough Sets (1995).
[9] Z. Ziarko Fundamental Informaticae, An International Journal, Vol.27, No. 2-3, Special Issue on Rough Sets (1996).
[10] Mushroom\UCI Machine Learning Repository Mushroom Data Set.htm