World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:8, No:10, 2014

Frequent Itemset Mining Using Rough-Sets

Usman Qamar, Younus Javed

***Abstract*—**Frequent pattern mining is the process of finding a pattern (a set of items, subsequences, substructures, etc.) that occurs frequently in a data set. It was proposed in the context of frequent itemsets and association rule mining. Frequent pattern mining is used to find inherent regularities in data. What products were often purchased together? Its applications include basket data analysis, cross-marketing, catalog design, sale campaign analysis, Web log (click stream) analysis, and DNA sequence analysis. However, one of the bottlenecks of frequent itemset mining is that as the data increase the amount of time and resources required to mining the data increases at an exponential rate. In this investigation a new algorithm is proposed which can be uses as a pre-processor for frequent itemset mining. FASTER (FeAture SelecTion using Entropy and Rough sets) is a hybrid pre-processor algorithm which utilizes entropy and rough-sets to carry out record reduction and feature (attribute) selection respectively. FASTER for frequent itemset mining can produce a speed up of 3.1 times when compared to original algorithm while maintaining an accuracy of 71%.

***Keywords*—**Rough-sets, Classification, Feature Selection, Entropy, Outliers, Frequent itemset mining.

I. Introduction

ITEMSET mining is the process of determining which groups of items appear together [1]. Itemset mining can be divided into two main categories: Frequent Itemset Mining and Rare Itemset Mining.

1. Frequent itemset mining: This type of itemset mining is focused on determining which groups of items frequently appear together in transactions.
2. Rare Itemset mining: In some situations it may be interesting to search for "rare" itemsets, i.e. itemsets that do not occur frequently in the data (contrasting frequent itemsets). These correspond to unexpected phenomena, possibly contradicting beliefs in the domain.

This investigation focuses on only frequent itemset mining. For frequent ietmset set mining, the most common algorithm used is Apriori. The Apriori algorithm [2] is a popular and foundational member of the correlation-based data mining methods. The Apriori algorithm operates by progressively building frequent sets over multiple generations. Each generation is composed of three sections: candidate generation, candidate pruning, and candidate support steps. Each generation provides a set of candidates that is expanded in the next generation. The support information is fed back into the candidate generator and the cycle continues until the final candidate set is determined. Candidate generation is the process in which one generation of candidates is built into the next generation. The main problem with Apriori is its complexity. Each candidate must be compared against every transaction and candidate generation must see the entire transaction set. Therefore, Apriori it is a computationally expensive algorithm and the running times can stretch up to days for large datasets.

The aim of this investigation is to propose a new algorithm which may be used as a pre-processor for Apriori algorithm thus speeding up the frequent itemset mining.

Usman Qamar and Younus Javed are with the Computer Engineering Department, College of Electrical and Mechanical Engineering, National University of Sciences and Technology. Islamabad, Pakistan (e-mail: usmanq@ceme.nut.edu.pk, myjaved@ceme.nust.edu.pk).

II. Apriori

The aim of Apriori algorithm is to generate itemsets which have the highest frequency together. It does this by initially, scanning the database DB once to get frequent 1-itemset. It then generates length (k+1) candidate itemsets from length k frequent itemsets and tests the candidates against DB. It will then terminate when no frequent or candidate set can be generated.

The code for Apriori is given below.

Step 1: self-joining Lk

Step 2: pruning

- Suppose the items in Lk-1 are listed in an order
- Step 1: self-joining Lk-1
- insert into Ck
- select p.item1, p.item2, …, p.itemk-1, q.itemk-1
- from Lk-1 p, Lk-1 q
- where p.item1=q.item1, …, p.itemk-2=q.itemk-2, p.itemk-1 < q.itemk-1
- Step 2: pruning
- for all item sets c in Ck do
- for all (k-1)-subsets s of c do
- if (s is not in Lk-1) then delete c from Ck

The Apriori algorithm suffers from the following bottlenecks.

- Multiple scans of transaction database
- Huge number of candidates
- Tedious workload of support counting for candidates

These bottlenecks make it simply useless on very large datasets.

III. FASTER

FASTER is a pre-processor algorithm with the aim of reducing the number of attributes. The FASTER algorithm can be divided into two main phases. The first phase uses entropy to generate an initial set of outliers and grade each record in the dataset in terms of its likeliness or unlikeliness of being a frequent record and the second phase carries out feature selection and attributes reduction using rough-sets.

Phase 1: Generating outliers using an Entropy Outlier Algorithm

- An x% sample is taken from a dataset. The process of selecting the parameter x is discussed in detail in section IV.
- The x% sample is used generate outliers using entropy [3] as well as a grading of each record in terms of its likeliness or unlikeliness of being a rare record. Using this information each record in the dataset is binary classified as either outlier or frequent.

Phase 2: Generating Reduct Attributes using Rough-Sets

- Rough-sets [4] are applied to the x% sample of the dataset. As a result, reduct attributes are generated. The Rosetta system developed by Øhrn [5] is utilised for this purpose; Rosetta is a publicly available platform for data mining with Rough Sets. It is one of the few freeware platforms that allow for generation of reducts using genetic algorithms (G.A) without any restrictions. G.A.'s have been used here to speed up the process of generating reducts.
- Phase 1 is repeated z times to obtain more consistent reduct attributes i.e. reducts that appear more than once. Again the process of selection of z in discussed in section IV.

Rough set theory (RST) [4] determines the degree of attributes dependency and their significance. "An information system (IS) is basically a flat table or view. An IS is defined by a pair (U,A), where U is a non-empty, finite set of objects and A is a non-empty, finite set of attributes". [6].

Decision systems (DS) [4] are a special kind of IS. By labeling the objects of A, it is possible to construct classes of objects. These classes can then be modeled using rough set analysis. The labels are the target attribute of which to obtain knowledge.

A decision system (i.e. a decision table) expresses all the knowledge about the model. This table may be unnecessarily large, in part because it is redundant in at least two ways: the same or indiscernible objects may be represented several times, or some of the attributes may be superfluous.

In practice most sets cannot be determined unambiguously and hence have to be approximated [7]. This is the basic idea of rough sets. If IS = (U,A) and then it is possible to approximate decision class X using the information contained by the attribute set of B. The lower and upper approximations are defined as follows [7]:

$$\underline{IND(B)}(X) = \left\{ x \in U : [x]_B \subseteq X \right\},$$

$$\overline{IND(B)}(X) = \left\{ x \in U : [x]_B \cap X \neq \varnothing \right\}.$$

The lower approximation [7] contains all objects that are certainly members of X. The objects in the set of the upper approximation [7] are possible members of X. The boundary region [7] is defined as the difference between the upper and the lower approximation.

One natural dimension of reducing data is to identify classes, i.e. objects that are indiscernible using the available attributes. Savings are to be made as only one element of the equivalence class is needed to represent the entire class. The other dimension of reduction is to keep only those attributes that preserve the indiscernibility relation and, consequently, set approximation. The remaining attributes are redundant as their removal should not worsen the classification. There are usually several such subsets of attributes and those which are minimal are called reducts.

Formally, if p is a random variable, and S the set of values that p can take, and the probability function of p, then entropy E (S) is defined [8] is shown as follows:

$$E(S) = -\sum_i p_i \ln p_i$$

The entropy of a multivariable vector x = {X1,….,Xm} can be computed as:

$$\sum_x p(x) = 1$$

Given a dataset D of n points p1…..pn where each point is a multidimensional vector, the aim is to find a subset O D, in such a way that we minimize the entropy of D – O.

IV. Experimentation

FASTER was applied as a pre-processor to four datasets. The details of the datasets are given in Table I.

TABLE I
DATASET DESCRIPTIONS FOR FREQUENT ITEMSET MINING

Dataset No	Dataset Name	No of Attributes	Source
D1	Letter Recognition	16	[9]
D2	Aaricus-Lpiota	22	[9]
D3	CEDAR	14	[9]
D4	Pima Indians Diabetes	8	[9]

The percentage accuracy between FASTER Pre-Processed Apriori and Apriori is shown in Table II.

TABLE II
% ACCURACY BETWEEN FASTER PRE-PROCESSED APRIORI AND APRIORI

Dataset	% Similarity Between FASTER Pre-Processed Apriori and Apriori
D1	73
D2	71
D3	70
D4	71

The time comparison Between FASTER Pre-Processed Apriori and Apriori is shown in Table III.

TABLE III
TIME COMPARISON BETWEEN FASTER PRE-PROCESSED APRIORI AND APRIORI

Dataset	Time Comparison Between FASTER Pre-Processed Apriori and Apriori
D1	0.8x faster
D2	2.2x faster
D3	2.4x faster
D4	3.1x faster

For frequent itemset mining such as Apriori, the best result that was achieved using FASTER (v1) was an accuracy of 71% at 3.1 times faster than Apriori. This can be explained by revisiting how entropy works. Entropy is essentially an outlier detection algorithm and even though by inverting the outliers, frequent records are generated, this is an overly simplistic view. There are specialized algorithms for the purposes of generating frequent records just as there are specialized records for the purpose of outlier generation. Therefore entropy will tend to produce better results for rare itemset mining than frequent itemset mining. However, this does not mean that entropy cannot be applied to frequent itemset mining as the above results have shown that by using FASTER, it is still possible to get an accuracy of around 70% and still be approximately 3.1 times faster than original Apriori.

The important observations that can be made are:

- Size of the percentage sample. The size of the sample taken from the dataset has a critical impact on FASTER. If the percentage sample taken is less than one-quarter of the dataset size, FASTER tends to produce poor results. This is be explained as a small sample size may completely miss all the outliers present in the dataset or each record in the sample may be considered as an outlier On the other hand, if the size of the percentage sample taken is more than 60% then, even though FASTER may produce highly accurate results, it will be achieved only by using more computation power.
- Number of times phase 1 is repeated. As with the size of the sample taken from the dataset, the number of times Phase 1 is repeated is also a critical factor. If Phase 1 is only repeated a few times, i.e. less than 5 times, then again FASTER tends to produce poor results. This is because FASTER will have not generated a comprehensive list of significant attributes in such few passes.

From the above results and analysis it is possible to define the optimal output parameters for FASTER.

- Optimal output: This generates the best possible outputs in terms of accuracy vs. speed for frequent itemset mining. In this case, the percentage similarity is the range of 70% to 75% between the FASTER pre-processed algorithm and the original algorithm and up to 3.2 times faster than original algorithm. For this option, the percentage sample taken from the dataset is set to 70% and the number of times Phase 1 is repeated is set to 7.

V. Limitations of Faster

This investigation has focused on itemset mining and FASTER has been shown to produce good results. For frequent itemset mining it can produce decent results with up to 71% accuracy and a total speed up 1.2 times when compared to the original algorithm.

It is interesting to see if FASTER can be applied to datasets which do not necessarily have a notion of rare/frequent records. For this purpose three datasets are specifically selected that have an equal distribution of decision attribute. Thus these dataset have no notion of outliers or non-outliers. This first dataset used is “Australian Credit Card” [9] where the decision attribute or the class attribute is the outcome of an application for a credit card, i.e. “rejected” or “accepted”. The class distribution in this case is 45.5% and 55.5% for “rejected” and “accepted” respectively. The second dataset used is “Iris” [9], where each class refers to a type of iris plant. It has a class distribution of 33.3% for each of 3 classes. The final dataset used is the “Heart Disease”, [9] where the patient is either diagnosed with the heart disease or not. Unlike the other two datasets which have an equal or near equal distribution of decision/class attribute, “heart disease” dataset is modified so that it has equal class distribution i.e 50% for each of the two classes.

To see the impact of FASTER on such datasets, a set of experiments have been carried out. The aim of these experiments is to compare the reducts generated by using rough-sets and FASTER. Both methods are applied to the same dataset with a decision attribute, the difference being when using rough-sets the decision attribute is used, and with FASTER the decision attribute is not used.

Thus, the experiment steps are

Step 1. Generate reduct attributes R using the dataset D with a decision attribute DS.

Step 2. Using the same dataset D but without the decision attribute DS, generate reduct attributes R’ by applying FASTER.

Step 3. Carry out a comparison between R and R’.

If the reducts generated are similar then this implies that FASTER can be used for reduct generation in datasets which are not limited to rare or frequent distribution, i.e. “well balanced datasets”.

Firstly we generate reduct attributes R using the dataset D with a decision attribute DS. Reduct attributes, R are generated using rough-sets for each dataset. For the purpose of generating reducts a G.A. is used. As s G.A. will produce as set of reducts, the reduct that produces the best results in terms of classification accuracy is selected as R. The classification accuracy of each reduct can be calculated as following:

1. Classifying data using the complete set of attributes: The dataset is divided into two sets, one for training and one for testing. Using the training dataset, decision rules are produced by C4.5/See5 [9]. The rest of the dataset is then classified using these rules with the full set of attributes.
2. Reduce: The G.A. is used to compute the reducts for the dataset. Reducts are generated using Rosetta.
3. Classifying using reducts: The training set is again classified by using the rules which were generated in step 1, but instead of using all the attributes this time only those attributes which are reducts are used. Thus for each reduct generated, the classification model is run. For example, if during step two, five sets of reducts were generated, the dataset will be classified five times, each time with a different set of reducts.

The classifiers obtained from the training sets of step1 and steps 3 are compared. As standard, any difference in the classification is called “percentage error in classification”.

As an example consider "Heart Disease Diagnosis" dataset. The dataset was divided into 2 equal sets, one for training and one for the testing. The dataset has in all 10 attributes i.e. age, chol, thalach, trestbps, oldpeak, which are numerical while cp, fbs, restcgs, thal, restecg are categorical. See5 was able to generate 7 rules for classifier i.e. heart-disease detected (yes) or not detected (no). The next step was to generate reducts using rough-sets. As already mentioned, Rosetta was used to generate the reducts.

Once the reducts have been obtained, the training dataset is again classified by using the rules generated, but this time instead of using all the attributes only the reducts are used. Both the classifications use same set of rules. The "Predicated Initial" indicates the class to which the instance was initially classified using the complete set of attributes, while "predicated reduct" indicates the class to which the same instance has been classified now by using only 3 attributes i.e. {age, trestbps, thalach}.

For this current dataset, out of 155 instances, eight [Yes] were wrongly classified as [No], while thirteen [No] were wrongly classified as [Yes]. The total percentage error for reduct1 i.e. {age, trestbps, thalach} was 13.7%. Now FASTER is used to generate reduct attributes R' for the dataset. Before the dataset is loaded into FASTER the decision attribute for the dataset is removed, as Phase 1 of FASTER i.e. OutlierAlg will generate the decision attribute. In order to get high quality reducts parameter P3 i.e. "Throughout Parameter" is used. As the aim of the experiment is to compare the reduct attributes generated by FASTER and rough-sets, there is no need to reduce the number of records. Thus only Phases 1 and 2 of FASTER are used.

For Heart Disease Dataset, using FASTER the reduct R' generated is {age, cp ©, trestbps, chol, thal©, restecg}. As can be seen, reduct R' generated by FASTER has identified three attributes out of five reduct attributes generated by rough-sets. Thus it can be said that percentage similarity between R and R' is 60%. This low level of percentage similarity can be explained by focusing on Phase 1 of FASTER. FASTER generates decision attribute using the OutlierAlg algorithm which is based on the concept of outlier detection using entropy. As already discussed, entropy outlier detection is based on the assumption that outliers are a minority i.e. the dataset is clearly distributed into outliers which are in minority and non-outliers which are frequent or in majority. However, if the dataset is not distributed in terms of outliers and non-outliers, OutlierAlg may not be a suitable method to generate decision attribute.

This shows the limitation of FASTER that it may be only applied to datasets which have a notion of rare/frequent records.

VI. Applications of Faster

FASTER can be used as a pre-processor for clustering and classification. By pre-processing the data, the time required to classify or cluster the data should be reduced. In order to see how effective FASTER can be as a pre-processor for classification and clustering a set of experiments can be done. Two types of experiments can be carried out: the first set will target classification while the second set will focus on clustering. For classification the experiments can be devised as following.

1. Classifying data using the complete set of attributes: The dataset is divided into two sets, training and testing. Using the training dataset, decision rules are produced by C4.5 (or another classifier). The rest of the dataset is then classified using these rules with the full set of attributes.
2. Applying FASTER: FASTER is used to generate reducts and reduce records.
3. Classifying using reducts: The training set is again classified by using the rules which were generated in step 1, but instead of using all the attributes this time only those attributes which are part of reducts are used.
4. Comparison: Comparison is carried out between the classifiers of the training sets of step1 and step 3.
5. The classifiers will indicate the accuracy of FASTER in terms of classification. If FASTER does a good job of classification then the majority of the outcomes in step1 and step 3 should be same.

Similar steps can be carried out for clustering

1. Clustering using complete set of attributes: The dataset is clustered with the complete set of attributes.
2. Reduce using FASTER: FASTER is used to compute the reducts for the dataset as reduce the number of records.
3. Clustering using Reducts: The dataset is again clustered but instead of using all the attributes this time only those attributes which are part of reducts are used.
4. Comparison: Comparison is carried out between the clusters of step1 and step 3.

If FASTER does a good job of clustering then majority of the clusters in step1 and step 3 should be the same or identical.

VII. Conclusion

FASTER is a hybrid pre-processor algorithm which utilizes both entropy and rough-sets to carry out feature selection. The aim of FASTER is to reduce dimensions both horizontally and vertically i.e. columns corresponding to attributes and rows corresponding to number of distinct samples or records. FASTER can be divided into two main phases: The first phase uses entropy to generate an initial set of outliers and frequent itemsets. The second phase carries out feature selection and attributes reduction using rough-sets. FASTER for frequent itemset mining can produce a speed up of 3.1 times when compared to original algorithm while maintaining an accuracy of 71%.

References

[1] R. Agrawal, T. Imielinski, Mining Association Rules between Sets of Items in Large Databases. SIGMOD 1993, pp. 207-216.
[2] S. Chai, J. Yang, Y. Cheng, The Research of Improved Apriori Algorithm for Mining Association Rules, International Conference on Service Systems and Service Management, 2007, pp. 1-4.
[3] J. Liang, Y. Qian, Information granules and entropy theory in information systems, Science in China Series F: Information Sciences, Vol. 51, 2008, pp. 1427-1444.
[4] Pawlak. Rough Sets: Theoretical Aspects of Reasoning About Data. Dordrecht: Kluwer Academic. 1991.

[5] Li-Juan, L. Zhou-Jun, A novel rough set approach for classification, IEEE International Conference on Granular Computing, 2006, pp. 349-352.
[6] C. Hung, H. Purnawan, B,Kuo, Multispectral image classification using rough set theory and the comparison with parallelepiped classifier, Geoscience and Remote Sensing Symposium, 2007. IGARSS 2007. IEEE International, pp. 2052-2055.
[7] R. Jensen and Q. Shen. Fuzzy-rough data reduction with ant colony optimization. Fuzzy Sets Systems, vol. 149, Issue No. 1, 2005, pp. 5–20.
[8] Zengyou H, Xiaofei Xu, An Optimization Model for Outlier Detection in Categorical Data, Lecture Notes in Computer Science, Volume 3644, 2005, pp. 400-409.
[9] UCL Machine Learning Group.