Modeling Interestingness of Streaming Classification Rules as a Classification Problem

Tolga Aydın and Halil Altay Güvenir

Department of Computer Engineering, Bilkent University,
06800 Ankara, Turkey
{atolga, guvenir}@cs.bilkent.edu.tr

Abstract. Inducing classification rules on domains from which information is gathered at regular periods lead the number of such classification rules to be generally so huge that selection of interesting ones among all discovered rules becomes an important task. At each period, using the newly gathered information from the domain, the new classification rules are induced. Therefore, these rules stream through time and are so called streaming classification rules. In this paper, an interactive classification rules' interestingness learning algorithm (ICRIL) is developed to automatically label the classification rules either as "interesting" or "uninteresting" with limited user interaction. In our study, VFFP (Voting Fuzzified Feature Projections), a feature projection based incremental classification algorithm, is also developed in the framework of ICRIL. The concept description learned by the VFFP is the interestingness concept of streaming classification rules.

1 Introduction

Data mining is the efficient discovery of patterns, as opposed to data itself, in large databases. Patterns in the data can be represented in many different forms, including classification rules, association rules, clusters, sequential patterns, time series, contingency tables, and others. However, for example, inducing classification rules on domains from which information is gathered at regular periods lead the number of such classification rules to be generally so huge that selection of interesting ones among all discovered rules becomes an important task. At each period, using the newly gathered information from the domain, the new classification rules are induced. Therefore, these rules stream through time and are so called streaming classification rules.

In this paper, an interactive classification rules' interestingness-learning algorithm (ICRIL) is developed to automatically label the classification rules either as "interesting" or "uninteresting" with limited user interaction. In our study, VFFP (Voting Fuzzified Feature Projections), a feature projection based incremental classification learning algorithm, is also developed in the framework of ICRIL. The concept description learned by the VFFP is the interestingness concept of streaming classification rules. Being specific to our concerns, VFFP takes the rule interestingness factors as features and is used to learn the rule interestingness concept and to classify the newly learned classification rules.

Section 2 describes the interestingness issue of rules. Section 3 is devoted to the knowledge representation used in this study. Section 4 and 5 are related to the training and classifying phases of the VFFP algorithm. ICRIL is explained in Section 6. Giving the experimental results in Section 7, we conclude in Section 8.

2 Interestingness Issue of Rules

There are factors contributing to the interestingness of a discovered rule such as coverage, confidence, completeness, actionability and unexpectedness. The first three factors are objective, actionability is subjective and unexpectedness is regarded both as subjective [2, 3, 4] and objective [5, 6]. Objective interestingness factors can be measured independently of the user and domain knowledge. But, subjective ones are user and/or domain knowledge dependent.

In this paper, different from the existing approaches in the literature, we learn the interestingness concept of the classification rules (rather than giving the interestingness concept as an input) and develop ICRIL algorithm in this respect. ICRIL tries to automatically label the rules with sufficient certainty factor. If it fails, user is requested to label the rule himself. Each rule labeled either as “interesting” or “uninteresting” by the user is treated as a training instance. And some interestingness factors that have the capability to determine the interestingness of rules are regarded as features, and interestingness labels (“interesting” or “uninteresting”) of the rules are regarded as the classes of the training instances. Any classification algorithm can be used in the training and the querying phases. However, we also developed VFFP in the framework of ICRIL. VFFP is an incremental feature projection based classification algorithm. It represents each different nominal feature value as a point in the associated feature projection. In the case of numeric features, it fuzzifies the feature and always uses three linguistic terms: low, medium and high. The shape of the membership function is given in Figure 1. The user supplies parameters p1, p2, p3 and p4 for each different feature.

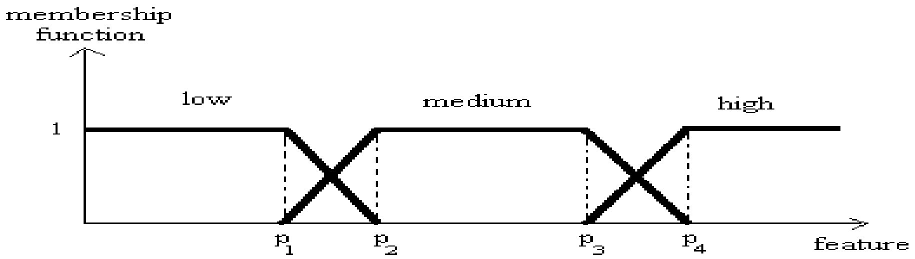

Fig. 1. Shape of the membership functions used for numeric features

3 Knowledge Representation

We think of a domain from which information is gathered at regular periods. For each period *p*, classification rules are induced from the gathered information and these rules’ interestingness labeling seems to be an important problem. This labeling problem is modeled as a new classification problem and a *rule set* is produced for these rules. Each instance of the rule set is represented by a vector whose components are the interestingness factors having the potential to determine the interestingness of the corresponding rule and the interestingness label of the same rule. Rules used in this study are probabilistic and have the following general structure:

If (A_1 *op value*$_1$) AND (A_2 *op value*$_2$) AND …AND (A_n *op value*$_n$) THEN
($Class_1$: $vote_1$, $Class_2$: $vote_2$,…,$Class_k$: $vote_k$)

A_i's are the features, $Class_i$'s are the classes and $op \in \{=, \leq, \geq\}$.

The instances of the rule set have either "interesting" or "uninteresting" as the interestingness label, and have the interestingness factors shown in Table 1. In this new classification problem, these factors are treated as determining features, and interestingness label is treated as the target feature (class) of the rule set.

Table 1. Features of the rule set

Feature	**Short description and/or formula**
Major Class	$Class_i$ that takes the highest vote
Major Class Frequency	Ratio of the instances having $Class_i$ as the class label in the data set
Rule Size	Number of conditions in the antecedent part of the rule
Confidence with respect to Major Class	$\lvert Antecedent\ \&\ Class_i \rvert / \lvert Antecedent \rvert$
Coverage	$\lvert Antecedent \rvert / \lvert N \rvert$
Completeness with respect to Major Class	$\lvert Antecedent\ \&\ Class_i \rvert / \lvert Class_i \rvert$
Standard Deviation of Class Votes	Standard deviation of the votes of the classes
Decisive	True if Std.Dev.of Class.Votes > s_{min}

Each feature carries information of a specific property of the corresponding rule. For instance, letting $Class_i$ to take the highest vote makes it the *Major Class* of that rule. If we shorten the representation of any rule as "If *Antecedent* THEN $Class_i$" and assume the data set to consist of N instances, we can define *Confidence*, *Coverage* and *Completeness* as in Table 1. Furthermore, a rule is decisive if the standard deviation of the votes is greater than s_{min}, whose definition is given as follows:

$$s_{min} = \frac{1}{(Class\ Count - 1)\sqrt{Class\ Count}} \tag{1}$$

4 Training in the VFFP Algorithm

VFFP (Voting Fuzzified Feature Projections) is a feature projection based classification algorithm developed in this study. It is used to learn the rule interestingness concept and to classify the unlabeled rules in the context of modeling rule interestingness problem as a new classification problem.

The training phase, given in Figure 4, is achieved incrementally. On a nominal feature, concept description is shown as the set of points along with the numbers of instances of each class falling into those points. On the other hand, on a numeric feature, concept description is shown as three linguistic terms (low, medium and high) along with the numbers of instances of each class falling into those linguistic terms. The user gives the parameters of the membership functions of each numeric feature as inputs. Training can better be explained by looking at the sample data set in Figure 2, and the associated learned concept description in Figure 3.

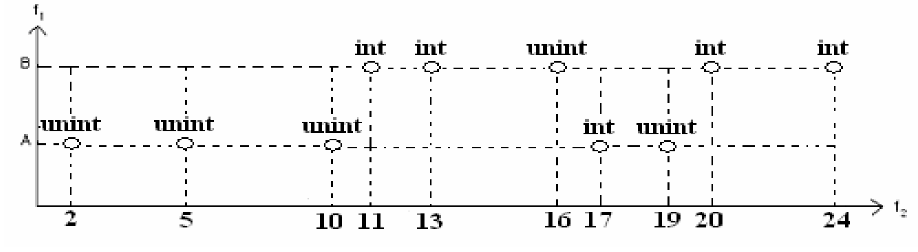

Fig. 2. Sample data set

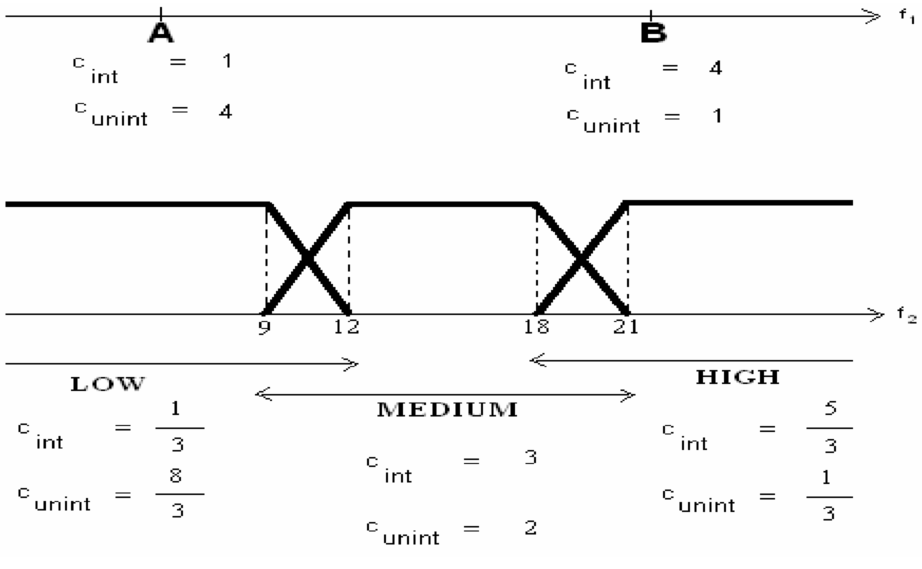

Fig. 3. Concept description learned for the sample data set

The example data set consists of 10 training instances, having nominal f_1 and numeric f_2 features. f_1 takes two values: 'A' and 'B', whereas f_2 takes some integer values. There are two possible classes: "int" and "unint". f_2 is assumed to have the following given parameter values: $p_1 = 9$, $p_2 = 12$, $p_3 = 18$ and $p_4 = 21$.

```
VFFP_train (t)      /* t: newly added training instance */
   let s be the class of t
   let others be the remaining classes
   class_count[s]++

   for each feature f

      if f is nominal
         p = find_point(f,t_f)
         if such a p exists
            point_class_count [f,p,s] ++
         else  /* add new point for f */
            add a new p' point
            point_class_count [f,p',s] = 1
            point_class_count [f,p',others] = 0

      else if f is numeric
        if membership value of t_f = 1
           Let l be the linguistic term that t_f falls in
           linguistic_term_class_count [f,l,s] ++
        else
           Let t_f be between parameters p_left and p_right
           Let left be the lefthandside linguistic term
           Let right be the righthandside linguistic
           term
           linguistic_term_class_count [f,left,s] +=
           (p_right - t_f) / (p_right - p_left)
           linguistic_term_class_count [f,right,s] +=
           (t_f - p_left) / (p_right - p_left)

            On numeric features (∀f, l, c)
   return      linguistic_term_class_count [f, l, c]
            On nominal features (∀f, p, c)
               point_class_count[f, p, c]
```

Fig. 4. Incremental train in VFFP

In Figure 4 for a nominal feature *f*, *find_point* (*f*, t_f) searches t_f, the new training instance's value at feature *f*, in the *f* projection. If t_f is found at a point *p*, then *point_class_count* [*f*, *p*, *s*] is incremented, assuming that the training instance is of class *s*. If t_f is not found, then a new point *p'* is constructed and *point_class_count* [*f*, *p'*, *class*] is initialized to 1 for *class* = *s*, and to 0 for *class* = *others*.

For a numeric feature *f*, if the membership value of t_f is 1 then the new training instance falls in a linguistic term *l* with full membership. That is, t_f lies in one of the following three intervals: $[0, p_1]$, $[p_2, p_3]$ or $[p_4, \infty)$. As a consequence, *linguistic_term_class_count* [*f*, *l*, *s*] is incremented. If the membership value of t_f is not 1, then the new training instance falls in the region shared by the linguistic terms *left* and *right*. That is, t_f lies in either of the two intervals (p_1, p_2) or (p_3, p_4). As a consequence, *linguistic_term_class_count* [*f*, *left*, *s*] and *linguistic_term_class_count* [*f*, *right*, *s*] are increased by amounts inverse proportional to the distance between t_f and the shared parameters p_{left} or p_{right}. It is apparent that the total increase of class counts, after arrival of a new training instance, is always 1.

5 Classification in the VFFP Algorithm

Classification in VFFP is shown in Figure 5. The query instance is projected on all features, and each feature gives normalized votes for the query instance. Normalization ensures each feature to have equal power in classification.

The classification starts by giving zero votes to classes on each feature projection. For a nominal feature *f*, *find_point* (*f*, q_f) searchs whether q_f exists in the *f* projection. If q_f is found at a point *p*, feature *f* gives votes for each class *c* as given in Equation 2, and then these votes are normalized to ensure equal voting power among features.

$$feature\_vote\,[f, c] = \frac{point\_class\_count[f, p, c]}{class\_count[c]} \tag{2}$$

In Equation 2, the number of class *c* instances on point *p* of feature projection *f* is divided by the total number of class *c* instances to avoid favoring major classes. For a numeric feature *f*, each class gets the vote given in Equation 3 given that the query instance falls in a linguistic term *l* with full membership. If the query instance falls in the region shared by the linguistic terms *left* and *right* (q_f lies in either of the two intervals (p_1, p_2) or (p_3, p_4)), each class gets the vote given in Equation 6. We note that *left_vote* [*f*, *c*] and *right_vote* [*f*, *c*] are increased by amounts inverse proportional to the distance between q_f and the shared parameters p_{left} or p_{right} of *left* and *right* linguistic terms. In both cases, votes of classes are again normalized.

$$feature\_vote\,[f, c] = \frac{linguistic\_term\_class\_count[f, l, c]}{class\_count[c]} \tag{3}$$

$$left\_vote\,[f, c] = \frac{linguistic\_term\_class\_count[f, left, c]}{class\_count[c]} \tag{4}$$

$$right\_vote\,[f, c] = \frac{linguistic\_term\_class\_count[f, right, c]}{class\_count[c]} \tag{5}$$

$$\begin{aligned} feature\_vote\,[f, c] = {} & left\_vote\,[f, c] * ((p_{right} - q_f) / (p_{right} - p_{left})) + \\ & right\_vote\,[f, c] * ((q_f - p_{left}) / (p_{right} - p_{left})) \end{aligned} \tag{6}$$

Final vote for any class *c* is the sum of all votes given by the features. If there exists a class *i* that uniquely gets the highest vote, then it is predicted to be the class of the query instance. The certainty factor of the classification is computed as follows:

$$C_f = \frac{final\ vote[i]}{\sum_{c=1}^{\#Classes} final\ vote[c]} \tag{7}$$

```
VFFP_query(q)          /* q: query instance*/
   feature_vote[f,c] = 0 (∀f, c)
   for each feature f
      if f is nominal
         p = find_point(f,$q_f$)
         if such a p exists
            for each class c
               Equation 2
            normalize_feature_votes (f)

      else if f is numeric
         if membership value of $t_f$ = 1
            Let l be the linguistic term that $q_f$ falls in
            for each class c
               Equation 3
            normalize_feature_votes (f)
         else
            Let $q_f$ be between parameters $p_{left}$ and $p_{right}$
            Let left be the lefthandside linguistic term
            Let right be the righthandside linguistic term
            for each class c
               Equation 4
               Equation 5
               Equation 6
            normalize_feature_votes (f)

   for each class c
      final_vote [c] = $\sum_{f=1}^{\#Features} feature\_vote[f,c]$

   if $\min_{c=1}^{\#Classes} final\_vote[c]$ < final_vote [k] = $\max_{c=1}^{\#Classes} final\_vote[c]$
      classify q as "k" with a certainty factor $C_f$
      return $C_f$
   else return -1
```

Fig. 5. Classification in VFFP

6 ICRIL Algorithm

ICRIL takes two input parameters: R_p (The set of streaming classification rules of period p and $MinC_t$ (Minimum Certainty Threshold)). It tries to classify the rules in R_p. If $C_f \geq MinC_t$ for a query rule r, this rule is inserted into the successfully classified rules set (R_s). Otherwise, two situations are possible: either the concept description is not able to classify r (C_f = -1), or the concept description's classification (prediction of r's interestingness label) is not of sufficient strength. If $C_f < MinC_t$, rule r is presented, along with its interestingness factor values such as *Coverage*, *Rule Size*, *Decisive* etc., to the user for classification. This rule is then inserted into the training rule set R_t and the concept description is reconstructed incrementally.

All the rules in R_p are labeled either automatically by the classification algorithm, or manually by the user. User participation leads learning process to be interactive. When the number of instances in the training rule set increases, the concept description learned tends to be more powerful and reliable. ICRIL executes on classification rules of all the periods and finally concludes by presenting the labeled rules in R_s. VFFP, the classification algorithm used in ICRIL, is more predictive than VFP, the classification algorithm used in IRIL [7]. For numeric features, VFP learns gaussian probability distribution functions for each class *c*, where as VFFP learns linguistic term class counts for each class *c*. The user gives the parameters of the membership functions of each numeric feature as inputs in the case of VFFP. However, there is no such a situation in VFP.

```
ICRIL ($R_p$, $MinC_t$)
   if p is the 1st period          //Warm-up Period
      $R_t \leftarrow \emptyset$,   $R_s \leftarrow \emptyset$
      for each rule $r \in R_p$
               ask the user to classify r
               set $C_f$ of this classification to 1
               insert r into $R_t$
               $VFFP_{train}$ (r)
   else
      for each rule $r \in R_p$
         $C_f \leftarrow VFFP_{query}$ (r)
            if $C_f < MinC_t$
               ask the user to classify r
               set $C_f$ of this classification to 1
               insert r into $R_t$
               $VFFP_{train}$ (r) //Update Concept Description
            else
               insert r into $R_s$
   return rules in $R_s$
```

Fig. 6. ICRIL algorithm

7 Experimental Results

ICRIL was tested to classify 1555 streaming classification rules induced from a financial distress domain between years 1989 and 1998. Each year has its own data and classification rules induced by using a benefit maximizing rule learner proposed in [1]. The data set of the financial distress domain is a comprehensive set consisting of 25632 data instances and 164 determining features (159 numeric, 5 nominal). There are two classes: "Succeed" and "Fail". The data set includes some financial information about 3000 companies collected during 13 years and the class feature states whether the company succeeded for the following three years. Domain expert previously labeled all the 1555 induced rules by an automated process to make accuracy measurement possible. Rules of the first year are selected as the warm-up rules to construct the initial concept description.

Table 2. Results for ICRIL

	$MinC_t$ 51%	$MinC_t$ 53%	$MinC_t$ 55%	$MinC_t$ 57%
Number of rules	1555	1555	1555	1555
Number of rules classified automatically with high certainty	1359	1294	1202	1048
User participation	13%	17%	23%	33%
Overall Accuracy	90%	93%	94%	97%

Table 3. Results for IRIL

	$MinC_t$ 51%	$MinC_t$ 53%	$MinC_t$ 55%	$MinC_t$ 57%
Number of rules	1555	1555	1555	1555
Number of rules classified automatically with high certainty	1344	1286	1196	1096
User participation	13%	17%	23%	29%
Overall Accuracy	80%	82%	86%	88%

In Table 2, results for $MinC_t = 51\%$ show that 1359 rules are classified automatically with $C_f > MinC_t$. User participation is 13% in the classification process. In the classification process, it is always desired that rules are classified automatically, and user participation is low.

The accuracy values generally increase in proportion to the $MinC_t$. Because higher the $MinC_t$, higher the user participation is. And higher user participation leads to learn a more powerful and predictive concept description. ICRIL achieves better accuracy values than IRIL, whose results are shown in Table 3.

8 Conclusion

ICRIL feature projection based, interactive classification rules' interestingness learning algorithm was developed and gave promising experimental results on streaming classification rules induced on a financial distress domain, when compared to our previous study in [7]. VFFP, the concept description learner developed in the course of ICRIL, makes use of less but more meaningful interestingness factors when compared to our previous study in [7].

References

1. Güvenir, H.A., "Benefit Maximization in Classification on Feature Projections" *Proceedings of the 3rd IASTED International Conference on Artificial Intelligence and Applications* (AIA'03), 2003, 424-429.
2. Klemettinen, M., Mannila, H., Ronkainen, P., Toivonen, H., and Verkamo, A.I., "Finding interesting rules from large sets of discovered association rules" *Proceedings of the 3rd Int. Conf. on Information and Knowledge Management*, 1994, 401-407.
3. Liu, B., Hsu, W., and Chen, S., "Using general impressions to analyze discovered classification rules" *Proceedings of the 3rd Int. Conf. on KDD*, 1997, 31-36.
4. Liu, B., and Hsu, W., "Post-analysis of learned rules", *AAAI*, 1996, 828-834.
5. Hussain, F., Liu, H., Suzuki, E., and Lu, H., "Exception rule mining with a relative interestingness measure" *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2000, 86-97.
6. Dong, G., and Li, J., "Interestingness of discovered association rules in terms of neighborhood-based unexpectedness" *Proceedings of the 2nd Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 1998, 72-86.
7. Aydın, T., and Güvenir, H.A., "Learning Interestingness of Streaming Classification Rules" *Proceedings of 19th International Symposium on Computer and Information Sciences* (ISCIS 2004), Antalya, Turkey (Oct. 27-29, 2004), LNCS 3280, Springer-Verlag, Cevdet Aykanat, Tugrul Dayar and Ibrahim Korpeoglu (Eds.), 62-71.