An Ensemble Based Approach for Feature Selection

Behrouz Minaei-Bidgoli, Maryam Asadi, Hamid Parvin
School of Computer Engineering, Iran University of Science and Technology (IUST), Tehran, Iran
{ b_minaei, parvin, asadi }@iust.ac.ir

Abstract. This paper proposes an ensemble based approach for feature selection. We aim at overcoming the problem of parameter sensitivity of feature selection approaches. To do this we employ ensemble method. We get the results per different possible threshold values automatically in our algorithm. For each threshold value, we get a subset of features. We give a score to each feature in these subsets. Finally by use of ensemble method, we select the features which have the highest scores. This method is not a parameter sensitive one, and also it has been shown that using the method based on the fuzzy entropy results in more reliable selected features than the previous methods'. Empirical results show that although the efficacy of the method is not considerably decreased in most of cases, the method becomes free from setting of any parameter.

Keywords. Feature Selection, Ensemble Methods, Fuzzy Entropy

1 Introduction

We have to use features of a dataset to classify data points in pattern recognition and data mining. Some datasets have a large number of features. Processing these datasets is not possible or is very difficult. To solve this problem, the dimensionalities of these datasets should be reduced. To do this, some of the redundant or irrelevant features should be eliminated. By eliminating the redundant and irrelevant features, the classification performance over them will be improved. Three different approaches are available for feature selection mechanism [1]. The first ones are embedded approaches. In these algorithms, feature selection is done as a part of the data algorithm. The second ones are filter approaches. These algorithm selected features before the data mining algorithm is run. The last ones are wrapper approaches. In these algorithms the target data mining algorithm is used to get the best subset of features.

A lot of methods for feature subset selection have been presented, such as similarity measures [2], gainentropies [3], the relevance of features [4], the overall feature evaluation index (OFEI) [5], the feature quality index (FQI) [5], the mutual information-based feature selector (MIFS) [6], classifiability measures [7], neuro-fuzzy approaches [8, 9], fuzzy entropy measures[10], etc.

In this paper we try to improve Shie-and-Chen's method. We try to solve the drawback of parameter sensitivity. To do this we use ensemble method. We get the results for different threshold values. For each threshold values, we get a subset of

features. We give a score to each feature in these subsets. Finally by use of ensemble concept, we select the features which have the highest scores. This method is not a parameter sensitive one, and also it has been shown that using the method based on the fuzzy entropy results in more reliable selected features than the previous methods'.

2 Proposed Algorithm

Shie-and-Chen's Algorithm which is presented in [10] is parameter sensitive. So if these parameters change, the result of algorithm can be changed significantly. When these parameters are given by the user, the quality of algorithm results will be even weaker. Because user selects the parameters randomly and experimentally, so it is possible that they are not proper values for an exemplary dataset. So the result of algorithm is not trustable. Also the proper values are not available for some datasets which are not used in this algorithm. So to find the best result we need to test the algorithm for a lot of possible threshold values. Then we must select the threshold values which cause the best results. To solve this problem we use ensemble method.

We do not select threshold values experimentally in our algorithm. Our algorithm test different possible values for thresholds and then by doing some steps, it selects the subset of features. This algorithm has 5 steps. We employ Shie-and-Chen's method by a little change in our algorithm. The result of their algorithm is a subset of features. But we get a sequence of features instead of a subset. Actually the order of feature appearance is important in our algorithm. First step runs Shie-and-Chen's method for each pair of (T_r , T_c). The result of algorithm at this step is a table of feature sequences which are selected for each pair of threshold values. For example the result of our algorithm for Iris is shown in Table 1. We obtained this result for 5 different values for T_c and T_r. Each element in this table is a feature sequence selected by the algorithm of Fig. 1 with a different pair of threshold values.

```
For   Tr = base_tr: step_tr :1
      For     Tc = base_tc: step_tc :1
              AllFSeq(Tr, Tc) = Shie-and-Chen's algorithm(Tr, Tc);
```

Fig. 1. Pseudo code of the first step of algorithm

It has two loops. One of them slides over T_r and the other one slides over T_c. Two parameters $base\_t_r$ and $base\_t_c$ are the minimum values used for T_r and T_c respectively. Two parameters $step\_t_r$ and $step\_t_c$ determine the distance between two consecutive threshold values of parameters T_r and T_c respectively. *FSeq* is a two dimensional matrix whose elements are features sequences obtained by the algorithm of Fig. 2 with each possible tested pair of threshold values. As it is inferred from Table 1, at the first and the last rows of each column we have some similar results for some threshold values. There is a similar discussion about the first and the last columns of each row. The results of algorithm for the first and the last columns of each row and the first and the last rows of each column are not trustable to reach some proper threshold values. Since these results have strongly negative effect on the final evaluation, at the second step we have to remove these repetitions. This step has two parts. The first

part removes the repetitions of columns and the second part removes the repetitions of rows. First part keeps only the results at the beginning and ending of each column to reach a dissimilar result at the beginning and ending of each column. And the second part keeps only the results at the beginning and ending of a row to reach a dissimilar result at the beginning or ending of each row. In other words, we use only one of the same results at the beginning and ending parts of each row and each column in final evaluation. The following pseudo code is the first part of second step of the algorithm.

```
New_AllFSeq = AllFSeq
For     Tr = base_tr: step_tr :1
        q = base_ tc
        While (true)
                q = q + step_ tc
                if      is_same ( AllFSeq ( Tr , base_tc ) , AllFSeq ( Tr, q ))
                        New_AllFseq ( Tr , q ) = EmptySeq
                else
                        break
        q = last_ tc
        While (true)
                q = q - step_ tc;
                if      is_same ( AllFSeq ( Tr , last_tc ) , AllFSeq ( Tr, q ))
                        New_AllFseq ( Tr , q ) = EmptySeq
                else
                        break
```

Fig. 2. Pseudo code of the first part of the second step of the algorithm

Table 1. Feature subsets selected for some pairs of threshold values over Iris dataset.

Tc, Tr	0.01	0.21	0.41	0.61	0.81
0.01	4, 3	3, 4	3, 4	3, 4	3, 4
0.21	4, 3	4, 3	3, 4	3, 4	3, 4
0.41	3, 4	4, 3	3, 4	3, 4	3, 4
0.61	4, 1	4, 2	3, 1	3, 1	3, 1
0.81	3, 1	4, 3, 1	3, 4	3, 4	3, 4

Equation 1 is a function that checks the similarity of its inputs. It has two input parameters which can be two sequences of features. If they are similar, the output will be 1 and if they are not similar the output is 0.

$$is\_same\ (a\ ,b) = \begin{cases} 1, & if\ x = y \\ 0, & otherwise \end{cases} \quad (1)$$

It checks the similarity between the first sequence of a column and the consecutive sequences of that column. By reaching the first dissimilar sequence at the beginning or ending of a column, this part of algorithm is done for each column. Output for Iris example of doing the first part of the second step of the algorithm is available in Table 2 by horizontal shading (+ sings). The second part of the second step of the algorithm is like Fig. 2. It is like the first part of the second step. It checks the similarity

between the first sequence of a row and the other sequences in that column. By reaching the first dissimilar sequence at the beginning or ending of a row, this part of algorithm is done for each row. Result of doing the second part of the second step of the algorithm over the Iris dataset which is obtained from the first step is shown in Table 2 by vertical shading (* sings).

Table 2. Delete repetitions in columns of Table 1 then delete repetitions in rows of Table 1.

Tc, Tr	0.01	0.21	0.41	0.61	0.81
0.01	4, 3	*	*	*	3, 4
0.21	+	+*	+*	+*	+
0.41	3, 4	4, 3	+*	+*	+
0.61	4, 1	4, 2	*	*	3, 1
0.81	3, 1	4, 3, 1	*	*	3, 4

Third step uses majority voting to reach the best subset of features. We have to give a score to each feature. There is a subset of selected features for each pair of T_r and T_c. We change this subset to a sequence of features by their ranks of appearing at the first step. In other words each feature that appears sooner has more effect on output, so it is given a higher score. Then we sum all given scores to features for each pair of threshold values. We define the score of each feature as equation 2.

After obtaining Table 2 for each dataset, we give a score to each of its features according to equation 2. In the equation 2, we give the higher weight to the first feature which appears sooner, and we give the lower weight to the last feature which appears at the end of the sequence. For example if there are 10 features, the weight of the first feature is considered 10, and the weight of the last feature is considered 1.

$$Score = \sum_{T_r} \sum_{T_c} \sum_{i=1}^{MaxSF} isequal(\ AllFSeq\ (T_r, T_c)(i), f) * (\ |AllFSeq| - i + 1) \quad 2$$

where MaxSF is obtained by equation 3.

$$MaxSF = max_{T_r, T_c, i}(|AllFSeq\ (T_r, T_c)(i)|) \quad 3$$

Finally we sum all the weighted scores obtained by the algorithm for different pairs of threshold values. For example, in the Iris example the *MaxFS* is 3. In the example we get these results: Score (3) = 21, Score (4) = 21, Score (1) = 7 and Score (2) = 2.

Table 3. Comparison between feature subsets selected by our and Shie-and-Chen's methods.

Data sets	Feature subsets selected by two methods	
	Shie-and-Chen's method	**Our method**
Iris	{4,3}	{4,3}
Breast cancer data set	{6, 2, 1, 8, 5, 3}	{6, 2, 3, 1, 9, 5}
Pima	{2, 6, 8, 7}	{2, 4, 6, 3}
MPG data set	{4, 6, 3}	{2, 4, 1}
Cleve data set	{13, 3, 12, 11, 1, 10, 2, 5, 6}	{13, 1, 12, 3, 9}
Crx data set	{9}	{9}
Monk-1 data set	{5, 1, 2}	{5, 1, 2}
Monk-2 data set	{5}	{5}
Monk-3 data set	{5, 2, 4}	{2,5,1}

Then we sort all features by their scores. After that we select the features with maximum scores. We select the same number of features as the Shie-and-Chen's method. In Iris example the subset of {3, 4} features is selected as final selected sub-

set, because these features have the highest scores, and Shie-and-Chen's method selected two features for this example.

3 Experimental Results

We have implemented our feature selection algorithm in Matlab. We use Weka software to evaluate the mapped datasets into the selected features obtained by our feature selection algorithms. We compare the feature subsets selected by our method with those selected by Shie-and-Chen's method in Table 3 for all of datasets which are used to compare in [10]. Also Table 4 shows that the obtained accuracies of different classifiers on the selected features obtained by proposed method are better than the obtained accuracies of the same classifiers on the selected features obtained by Shie-and-Chen's algorithms the most datasets.

Table 4. Comparing classification accuracy rates of our and Shie-and-Chen's methods

Data sets	**Classifiers**	**Average classification accuracy rates of different methods**	
		Our method	**Shie-and-Chen's method**
Pima diabetes data set	LMT	76.30 ±4.84%	77.22 ± 4.52%
	Naive Bayes	76.30 ±4.84%	77.47 ± 4.93%
	SMO	75.65 ±5.61%	77.08 ± 5.06%
	C4.5	94.62 ±2.12%	74.88 ± 5.89%
Cleve data set	LMT	82.42 ± 5.34%	82.87 ± 6.23%
	Naive Bayes	80.41 ± 3.95%	84.48 ± 3.93%
	SMO	80.00 ± 5.99%	83.51 ± 6.09%
	C4.5	76.90 ± 8.40%	76.90 ± 8.40%
Correlated data set	LMT	100.00 ± 0.00%	100.00 ± 0.00%
	Naive Bayes	86.03 ± 3.75%	86.03 ± 3.75%
	SMO	89.87 ± 6.88%	89.87 ± 6.88%
	C4.5	94.62 ± 4.54%	94.62 ± 4.54%
M of N-3-7-10 data set	LMT	100.00 ± 0.00%	100.00 ± 0.00%
	Naive Bayes	89.33 ± 1.56%	89.33 ± 1.56%
	SMO	100.00 ± 0.00%	100.00 ± 0.00%
	C4.5	100.00 ± 0.00%	100.00 ± 0.00%
Crx data set	LMT	86.53 ±3.87%	85.22 ± 4.04%
	Naive Bayes	86.53 ±3.87%	85.51 ± 4.25%
	SMO	86.53 ±3.87%	85.80 ± 3.71%
	C4.5	85.36 ± 4.12%	85.51 ± 4.25%
Monk-1 data set	LMT	100 ± 0.00%	100.00 ± 0.00%
	Naive Bayes	72.22 ± 6.33%	74.97 ± 1.95%
	SMO	72.22 ± 6.33%	75.02 ± 5.66%
	C4.5	100.00 ± 0.00%	100.00 ± 0.00%
Monk-2 data set	LMT	67.14 ± 0.61%	67.36 ± 1.17%
	Naive Bayes	67.14 ± 0.61%	67.14 ± 0.61%
	SMO	67.14 ± 0.61%	67.14 ± 0.61%

	C4.5	67.14± 0.61 %	67.14 ± 0.61%
Monk-3 data set	LMT	97.22 ± 0.47%	99.77 ± 0.10%
	Naive Bayes	97.21 ± 2.71%	97.21 ± 2.71%
	SMO	97.22 ± 0.47%	100.00 ± 0.00%
	C4.5	100.00 ± 0.00%	100.00 ± 0.00%

4 Conclusion

This paper improves one of the existing feature selection algorithms, Shie-and-Chen's method. This feature selection algorithm uses fuzzy entropy concept. The problem of Shie-and-Chen's method is that it is a parameter sensitive algorithm. User should select threshold values in that algorithm experimentally. The result of algorithm for some threshold values is very weak and it is not trustable. To solve this problem we use ensemble method. Our paper runs Shie-and-Chen's algorithm for different values as thresholds and then gives a weight to each selected features according its rank. Finally by using one of the ensemble methods, majority voting, it selects the best features which have the highest scores. So this algorithm does not need any input parameter. Also the obtained accuracies of different classifiers on the selected features obtained by proposed method are better that the obtained accuracies of the same classifiers on the selected features obtained by Shie-and-Chen's algorithms.

References

1. Tan, P. N., Steinbach, M., and Kumar V.: Introduction to Data Mining, first ed. Ad-dison-Wesley Longman Publishing Co. Inc., (2005)
2. Tsang, E.C.C., Yeung, D.S., and Wang, X.Z.: OFFSS: optimal fuzzyvalued feature subset selection. IEEE Trans Fuzzy Syst 11(2):202–213, (2003)
3. Caruana, R and Freitag, D.: Greedy attribute selection. In: Proceedings of international conference on machine learning, New Brunswick, NJ, pp 28–33, (1994)
4. Baim, P.W.: A method for attribute selection in inductive learning systems. IEEE Trans Pattern Anal Mach Intell 10(6):888–896, (1988)
5. De, R.K., Basak, J., and Pal, S.K.: Neuro-fuzzy feature evaluation with theoretical analysis. Neural Netw 12(10):1429–1455, (1999)
6. Battiti, R.: Using mutual information for selecting features in supervised neural net learning. IEEE Trans Neural Netw5(4):537–550, (1994)
7. Dong, M. and Kothari, R.: Feature subset selection using a new definition of classi-fiability. Pattern Recognit Lett 24(9):1215–1225, (2003)
8. De, R.K., Pal, N.R., and Pal, S.K.: Feature analysis: neural network and fuzzy set theoretic approaches. Pattern Recognit 30(10):1579– 1590, (1997)
9. Platt, J.C.: Using analytic QP and sparseness to speed training of support vector ma-chines. In: Proceedings of the thirteenth annual conference on neural information processing systems, Denver, CO, pp 557–563, (1999)
10. Shie, J.D. and Chen S.M.: Feature subset selection based on fuzzy entropy measures for handling classification problems, Springer Science+Business Media, (2007)