IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-5, Issue-12 E-ISSN: 2347-2693

Evaluation of a NeuroFuzzy Unsupervised Feature Selection Approach

Bacharaju Vishnu Swathi

Dept. of CSE, Geetanjali College of Enggineering and Tech., Hyderabad, India

*Corresponding Author: swathiveldanda@yahoo.com

Tel.: +91-9885096286

Available online at: www.ijcseonline.org

Received: 25/Nov/2017, Revised: 02/Dec/2017, Accepted: 18/Dec/2017, Published: 31/Dec/2017

***Abstract*—**Dimensionality reduction is a commonly used step in machine learning, especially when dealing with a high dimensional space of features. The original feature space is mapped onto a new, reduced dimensionality space and the examples to be used by machine learning algorithms are represented in that new space. The mapping is usually performed either by feature extraction or feature selection. Feature extraction involves constructing some new features from original feature set. Feature selection involves selecting a subset of the original features from original feature set without transformation. Feature selection can be implemented either by feature ranking or subset selection. Feature ranking is an approach in which all the features are ranked based on some criteria. In this project, Feature ranking algorithm has been implemented. Work presented here includes the implementation of UFSNF for ranking different features using the fuzzy evaluation index with neural networks. The results (ranks) obtained from UFSNF have been compared with the ranks obtained by Relief-F evaluator using four clustering techniques EM, k-Means, Farthest First and Hierarchical. For the experimental study, benchmark datasets from the UCI Machine Learning Repository have been used. From the study, it is found that the newly proposed algorithm, UFSNF in some cases exceeds the performance of Relief-F.

***Keywords*—**Dimensionality reduction,feature selection,unsupervised, Relief-F,clustering

I. INTRODUCTION

Across a wide variety of fields, data are being collected and accumulated at a dramatic pace. There is an urgent need for a new generation of computational theories and tools to assist humans in extracting useful information (knowledge) from the rapidly growing volumes of digital data. These theories and tools are the subject of the emerging field of knowledge discovery in databases (KDD). Data sets for analysis may contain hundreds of attribute, many of which may be irrelevant to the mining task, or redundant.

For example, if task is to classify customers as to whether or not they are likely to purchase a popular new CD when notified of a sale, attribute such as the customers telephone no are likely to be irrelevant, unlike attributes such as age or music taste. All though it may be possible for a domain expert to pick out some of the useful attributes, this can be a difficult and time-consuming task, especially when the behaviour of the data is not well known. Leaving out relevant attributes or keeping irrelevant attributes may be detrimental, causing confusion for the mining algorithm employed. This can result in discovered patterns of poor quality. In addition, the added volume of irrelevant or redundant attributes can slow down the mining process.

As a last paragraph of the introduction should provide In machine learning, feature selection, also known as variable selection, feature reduction, attribute selection or variable subset selection, is the technique of selecting a subset of relevant features for building robust learning models.

Feature selection is a must for any data mining product. That is because, when you build a data mining model, the dataset frequently contains more information than is needed to build the model. For example, a dataset may contain 500 columns that describe characteristics of customers, but perhaps only 50 of those columns are used to build a particular model. If you keep the unneeded columns while building the model, more CPU and memory are required during the training process, and more storage space is required for the completed model.

Even if resources are not an issue, you typically want to remove unneeded columns because they might degrade the quality of discovered patterns, for the following reasons:

1. Some columns are noisy or redundant. This noise makes it more difficult to discover meaningful patterns from the data.

© 2017, IJCSE All Rights Reserved

2. To discover quality patterns, most data mining algorithms require much larger training data set on high-dimensional data set. But the training data is very small in some data mining applications.

Feature selection helps solve this problem, of having too much data that is of little value, or of having too little data that is of high value. Feature selection works by calculating a score for each attribute, and then selecting only the attributes that have the best scores. You can adjust the threshold for the top scores. Feature selection is always performed before the model is trained, to automatically choose the attributes in a dataset that are most likely to be used in the model.

There are various methods for feature selection[2][3]. The exact method for selecting the attributes with the highest value depends on the algorithm used in your model, and any parameters that you may have set on your model. Feature selection is applied to inputs, predictable attributes, or to states in a column. Only the attributes and states that the algorithm selects are included in the model-building process and can be used for prediction. Predictable columns that are ignored by feature selection are used for prediction, but the predictions are based only on the global statistics that exist in the model.

The paper is organized as follows. Section II presents the neuro-fuzzy feature selection algorithm which is implemented as part of the work presented herein. The experimental methodology adopted is presented in section III. Results and discussion are presented in section IV. Conclusion and future work are presented in section V.

II. RELATED WORK

The feature selection algorithm that has been implemented is based on the paper in [1]. The network shown in Fig.1 results in an optimal order of importance of individual features in the feature space. The network consists of an input layer, a hidden and an output layer. The input layer consists of a pair of nodes corresponding to each feature, i.e., the number of nodes in the input layer is '2n', for 'n' dimensional feature space. The hidden layer consists of 'n' number of nodes. The output layer consists of two nodes µO and µT which determine how similar a pair of patterns is in the original and transformed feature spaces respectively.

UFSNF algorithm involves four stages:Initialization of weights, Feed forward,Back propagation of errors and Updation of weights

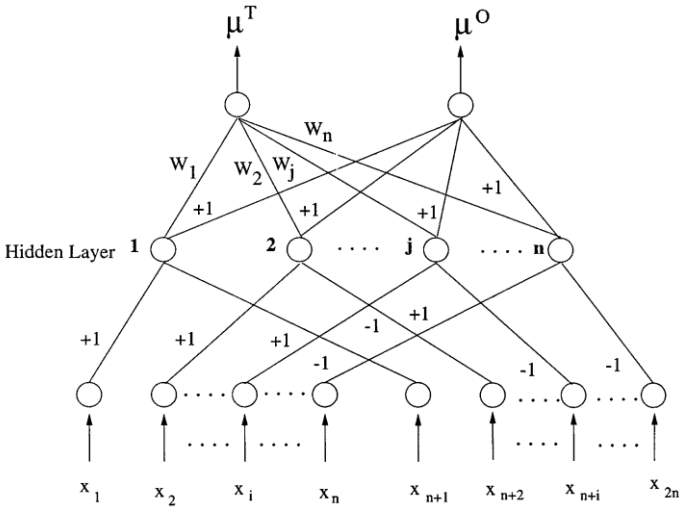

Figure 1. Neural network model for feature selection

A. *Initialization of weights*

It will influence whether the net reaches a global (or a local) minima of the error and if so how rapidly it converges. If the initial weight is too large, the initial input signals to each hidden or output unit will fall in the saturation region. If the initial weights are too small, the net input to a hidden or output unit will approach to zero which then causes extremely slow learning. To get the best results, the initial weights are set to the numbers between -0.5 and 0.5 or between -1 and 1.

Step1: Initialize weights to small random values.

Step2: While stopping condition is false, do steps 3-10.

Stopping condition is the minimization of fuzzy evaluation index i.e., when the change in evaluation index gets reduced to a small value.

Step3: For 'n' patterns, there are nc2 combinations of pair of patterns.

For each pair of patterns, similarity is measured by performing steps 4-10.

B. *Feed forward*

During this stage, each input unit receives an input signal and transmits this signal to each of the hidden units. Each hidden unit then calculates the activation function and sends its signal to each output unit. The output unit calculates the activation function to form the response of the net for the given input pattern.

Step4: Each input unit receives activations Xi corresponding to the feature values of each pair of patterns 'p' and 'q'.

© 2017, IJCSE All Rights Reserved

These signals are transmitted to the layer above i.e. hidden units. Xi is the ith feature value of a particular pattern.

Step5: A jth hidden node is connected only to an ith and (i+n)th input nodes via weights +1 and -1 respectively where j, i = 1,2, …, n. Each hidden unit (Zj) sums its weighted input signals. The activation function is

$$Zj = (Xpi - Xqi)2 \text{ , } 1 \le i \le n \text{ , } 1 \le j \le n$$

where j is the number of hidden units and i is the number of features. This signal is sent to all units in the layer above i.e. output units.

Step6: The output node computing μO (i.e., μ value in the original feature space) is connected to jth hidden node via weight +1 each whereas that computing μT(i.e., μ value in the transformed feature space) is connected to the hidden nodes via weights Wj(=wj2). Each output unit $u_T^{(2)}$ and $u_O^{(2)}$ sums its weighted input signals i.e., $u_T^{(2)}$ and $u_O^{(2)}$ are the activations received by the output node which computes μ^{T}_{pq} and μ^{O}_{pq} respectively.

$$u_T^{(2)} = \sum_{j=1}^{n} Z_j$$

$u_T^{(2)}$ is the distance measure which provides the similarity between the pth and qth patterns in the original feature space. The higher the value of $u_T^{(2)}$, the lower is the similarity between pth and qth patterns, and vice versa.

$$u_O^{(2)} = \sum_{j=1}^{n} Z_j W_j$$

This gives the similarity between pth and qth patterns in the transformed feature space.

Step7: Computation of membership function μOpq and μTpq.

μOpq and μTpq are the activations of the output node. μOpq is the degree that both pth and qth patterns belong to the same cluster in the n-dimensional original feature space and μTpq be that in the n1 dimensional transformed feature space.

$$\mu^{O}_{pq} = 1 - \frac{u_O^{(2)}}{D} \quad \text{if } u_O^{(2)} \le D$$
$$= 0, \quad \text{otherwise}$$

$$\mu^{T}_{pq} = 1 - \frac{u_T^{(2)}}{D} \quad \text{if } u_T^{(2)} \le D$$
$$= 0, \quad \text{otherwise}$$

D is a parameter which reflects the minimum separation between a pair of patterns belonging to two different clusters.

The term D is given as D = β dmax, where dmax is the maximum separation between a pair of patterns in the entire feature space and $0 < \beta \le 1$ is a user defined constant. β determines the degree of flattening of membership function. The higher the value of β , more will be the degree and vice versa

$$d_{\max} = \sqrt{\sum_i (x_{\max i} - x_{\min i})^2}$$

where xmaxi and xmini are the maximum and minimum values of the ith feature in the corresponding feature space

C. Back propagation of errors

During back propagation of errors, each output unit determines the associated error for that pattern with that unit. The error is distributed back to all units in the previous layer.

Step8: Calculate fuzzy feature evaluation index

The fuzzy evaluation index for a set of features is defined in terms of membership values denoting the degree of similarity between two patterns both in the original and the transformed spaces. The evaluation index is such that, for a set of features, the lower is its value, the higher is the importance of that set in characterizing/discriminating various clusters.

$$E = \frac{2}{S(S-1)} \sum_{p} \sum_{p \ne q} \frac{1}{2} [\mu^{T}_{pq}(1 - \mu^{O}_{pq}) + \mu^{O}_{pq}(1 - \mu^{T}_{pq})] \quad (1)$$

Here, S is the number of samples on which the feature evaluation index is calculated.

It has the following cases:

Case 1: For μOpq <0.5 as μTpq → 0, E decreases. For μOpq > 0.5 as μTpq → 1, E decreases. In both the cases, the contribution of the pair of patterns to the evaluation index E becomes minimum (=0) when μOpq= μTpq =0 or 1.

Case 2: For μOpq <0.5 as μTpq → 1, E increases. For μOpq > 0.5 as μTpq → 0, E increases. In both the cases, the contribution of the pair of patterns to the evaluation index E becomes maximum (=0.5) when μOpq= 0 and μTpq =1 or μOpq= 1 and μTpq =0

© 2017, IJCSE All Rights Reserved

Case 3: If μOpq =0.5, the contribution of the pair of patterns to E becomes constant (=0.5), i.e., independent of μTpq .

The cases 1 and 2 can be verified as follows. From Equation (1) we have

$$\frac{\partial E}{\partial \mu^{T}_{pq}} = \frac{1}{S(S-1)}[1 - 2\mu^{O}_{pq}]$$

For μOpq < 0.5, (∂E/∂μTpq) > 0. This signifies that E decreases (increases) with decrease (increase) in μTpq. For μOpq > 0.5, (∂E/∂μTpq) < 0. This signifies that E decreases (increases) with increase (decrease) in μTpq. Since μTpq ϵ [0, 1], E decreases (increases) as μTpq → 0(1) in the former case, and μTpq → 0(1) in the latter.

Therefore, the feature evaluation index decreases as the membership value representing the degree of belonging of pth and qth patterns to the same cluster in the transformed feature space tends to either zero (when μOpq < 0.5) or one (when μOpq > 0.5), and becomes minimum for μOpq = μTpq = 0 or 1. In other words, the index decreases as the similarity (dissimilarity) between two patterns belonging to the same cluster (different clusters) in the original feature space increases, thereby making the decision regarding belongingness of patterns to a cluster more crisp. This means, if the intercluster (intracluster) distances in the transformed space increase (decrease), the feature evaluation index of the corresponding set of features decreases. Therefore, our objective is to select/extract those features for which the evaluation index becomes minimum, thereby optimizing the decision on the similarity of a pair of patterns with respect to their belonging to a cluster.

Case 2 implies that E increases when similar (dissimilar) patterns in the original space becomes dissimilar (similar) in the transformed space. That is, any occurrence of such a situation will be automatically protected by the process of minimizing E. Similarly in case 3, when μOpq = 0.5, decision regarding the similarity between a pair of patterns whether they lie in the same cluster or not, is most ambiguous, the contribution of the pattern pair to E does not have any impact on the minimization process.

Step9: Error information is calculated

D. Updation of weights

We usually end up with an error in each of the output units. We have to minimize the error. The simplest method to do this is the greedy method which changes the connections in the neural network in such a way that, next time around, the error will be reduced for this particular pattern.

Step 10: The weights for each feature is updated with respect to the error information calculated in step 9.

The weight correction term is given by

$$\Delta Wj = -\eta \partial E(W)/\partial Wj , \quad \forall j$$

Where η is the learning rate. A high learning rate leads to rapid learning but the weights may oscillate, while a lower learning rate leads to slower learning. Therefore, the new weights are Wj (new) = Wj (old) + ΔWj .

Step 11**:** Go to step 2 and repeat the whole process with the new weights obtained in step 10.

The stopping condition is minimization of fuzzy feature evaluation index E.

III. METHODOLOGY

UFSNF has been compared against Relief-F evaluator using EM, k-Means, Farthest First and Hierarchical clustering techniques. Initially the algorithm has been implemented in MATLAB and later revised to client-server system using Java Server Pages technology for access to clients over the web.Seven benchmark data sets Iris, Balloons, Balance scale, BUPA, Lenses, Hayes-Roth and Monks from the UCI Machine Learning Repository [4] have been used for finding the effectiveness of UFSNF algorithm. All these data sets are numerical datasets.

Table 1. Data sets

S. No.	Data Set	Instances	Features	classes
1	Balance scale	625	4	3
2	Balloons	16	4	2
3	Hayes-Roth	160	5	3
4	Iris	150	4	3
5	Lenses	24	4	3
6	BUPA	345	6	2
7	Monks	432	7	2

Table 2. Feature ranking obtained by UFSNF and Relief-F

Data Set	Relief-F Ranking	UFSNF Ranking
Balance scale	1,3,4,2	1,4,2,3
Balloons	3,4,1,2	4,2,3,1
Hayes-Roth	3,5,4,1,2	1,2,4,5,3
Iris	4,3,1,2	4,3,2,1
Lenses	4,3,2,1	1,4,3,2
BUPA	3,5,6,4,1,2	6,1,4,3,5,2
Monks	2,5,7,1,4,3,6	1,2,3,6,4,5,7

UFSNF algorithm is evaluated using EM, k-Means, Farthest First and Hierarchical clustering techniques on various feature subsets of a data set and the clustering error rate is

© 2017, IJCSE All Rights Reserved

used to measure the quality of the feature set. For a data set of size 'n' the feature subset size may range from 1 to n. For instance, the various feature subsets possible for Iris data set with size '4' are {4}, {4,3}, {4,3,1} and {4,3,1,2} for the standard ranking {4,3,1,2} obtained by Relief-F and {4},{4,3},{4,3,2},{4,3,2,1} for the ranking {4,3,2,1} obtained by UFSNF algorithm.

Graphs are plotted with error rates on the Y-axis and the number of significant features used for clustering on X-axis. The values on the X-axis can be interpreted as follows. When x is equal to 2 for a given data set, say Iris, it indicates that clustering is done with the two most important features 4th and 3rd of the Iris data set and the corresponding value on the Y-axis depicts the clustering error rate. The graph shows the error rates produced by clustering with the feature subsets obtained from the ranking of our algorithm as well as that obtained from the Relief-F evaluator method. The performance of the algorithm, UFSNF is close to and sometimes even better than that of Relief- F.

IV. RESULTS AND DISCUSSION

This section presents the results obtained in terms of comparing the performance of UFSNF with that of Relief-F. Specifically, the results for a single dataset, namely, balance scale, are presented. Clustering error rates for the selected features are evaluated with respect to three well known clustering algorithms.

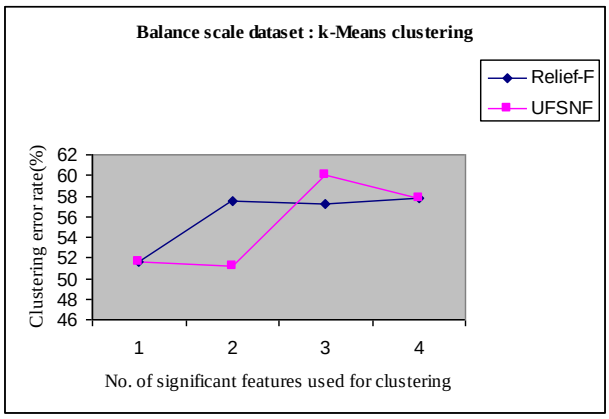

Figure 2.Comparison using k-Means clustering

In case of k-Means clustering, Relief-F produces same error rate as that of UFSNF with their respective feature subsets of size '1' and '4'. When the feature subset size is 2, error rate is low when clustered with UFSNF subsets compared to Relief-F and vice versa when the feature subset size is '3'. A lowest error rate of 51.2% is obtained when clustered with UFSNF feature subset of size '2'. A lowest error rate of 51.68% is obtained when clustered with Relief-F feature subset of size '1'. When the subset size is '3' or more, the error rate has increased. Hence, UFSNF feature subset of size '2' can be considered for feature selection.

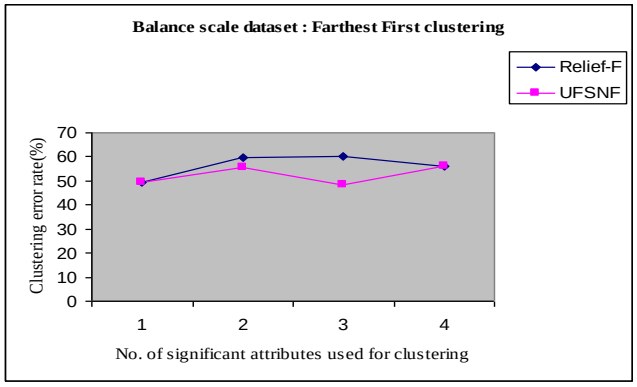

Figure 3. Comparison using Farthest First clustering

In case of Farthest First clustering, UFSNF performance is better over Relief-F with feature subsets of all sizes. Least error rate of 48.32% is obtained by clustering with UFSNF feature subset of size '3' and a least error rate of 49.6% is obtained by clustering with Relief-F subset of size '1'. Hence, UFSNF feature subset of size '3' can be considered for feature selection

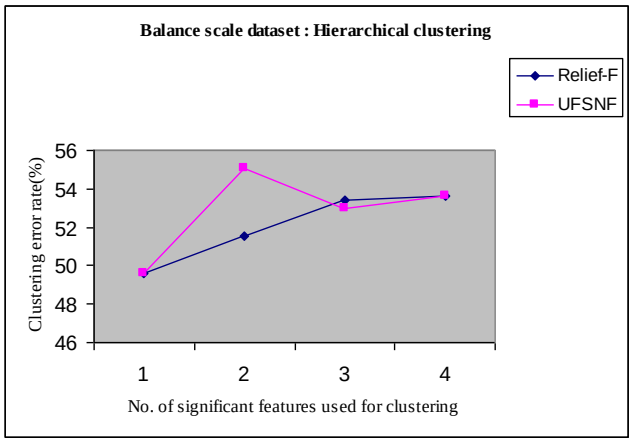

Figure 4. Comparison using Hierarchical clustering

In case of Hierarchical clustering, UFSNF performance is the same as Relief-F when the feature subset size is '1' and '4'. When clustered with a feature subset of size '2', Relief-F produces less error rate of 51.52% than UFSNF. When clustered with a feature subset of size '3', UFSNF produces less error rate of 52.96%. . A lowest error rate of 49.6% is obtained when clustered with UFSNF feature subset of size '1'. A lowest error rate of 49.6% is obtained when clustered with Relief-F feature subset of size '1'. Hence, UFSNF feature subset of size '1' can be considered for feature selection.

© 2017, IJCSE All Rights Reserved

V. CONCLUSION AND FUTURE SCOPE

An algorithm for feature selection has been implemented which involves feature ranking. That is, finding the order of importance of each feature which is useful for discriminating clusters. This algorithm has been implemented with Neuro fuzzy methodology under unsupervised mode of learning.

The results (ranks) obtained from UFSNF have been compared with the ranks obtained by Relief-F evaluator using four clustering techniques EM, k-Means, Farthest First and Hierarchical. From the experimental study, it is found that UFSNF algorithm exceeds the performance of Relief-F in some cases.

The algorithm UFSNF only works with numerical datasets. This can be further extended to work with any kind of datasets.

REFERENCES

[1] Sankar K. Pal, Fellow, IEEE, Rajat K. De, Member, IEEE, and Jayanta Basak, Senior Member, IEEE "*Unsupervised learning: Neuro fuzzy approach*", IEEE Transactions on Neural Networks, vol. 11, no. 2, MARCH 2000.

[2] E. C. C. Tsang, D. S. Yeung, and X. Z. Wang, "*Optimal Fuzzy-Valued Feature Subset Selection"*, IEEE Transactions On Fuzzy Systems, vol. 11, no. 2, APRIL 2003

[3] Hahn-Ming Lee, Chih-Ming Chen, Jyh-Ming Chen, and Yu-Lu Jou, "*An Efficient Fuzzy Classifier with Feature Selection Based on Fuzzy Entropy*", IEEE Transactions on Systems, Man, and Cybernetics—PART B: CYBERNETICS, vol. 31, no. 3, JUNE 2001

[4] http://archive.ics.uci.edu/ml/datasets.html

© 2017, IJCSE All Rights Reserved