Learning and Optimizing the Features with Genetic Algorithms

Dr.E.Chandra
Research Supervisor & Director
Department of Computer Applications
D, J. Academy for Managerial
Excellence Coimbatore-641 032
Tamil Nadu, India

K. Nandhini
Research Scholar & Head
Department of Computer Science
Dr.N.G.P. Arts and Science College
Coimbatore -641 048, Tamil Nadu,
India

ABSTRACT
The quality of the data being analyzed is a critical factor that affects the accuracy of data mining algorithms. There are two important aspects of the data quality, one is relevance and the other is data redundancy. The inclusion of irrelevant and redundant features in the data mining model results in poor predictions and high computational overhead. Feature extraction aims to reduce the computational cost of feature measurement, increase classifier efficiency, and allow greater classification accuracy based on the process of deriving new features from the original features. This paper represents an approach for classifying students in order to predict their final grades based on features extracted from logged data in an educational web-based system. A combination of multiple classifiers leads to a significant improvement in classification performance. By weighing feature vectors representing feature importance using a Genetic Algorithm (GA), we can optimize the prediction accuracy and obtain a marked improvement over raw classification. We further show that when the number of features is few, feature weighting and transformation into a new space works efficiently compared to the feature subset selection. This approach is easily adaptable to different types of courses, different population sizes, and allows for different features to be analyzed.

General Terms
Data Mining, Machine Learning, Genetic Algorithms.

Keywords
Feature Subset Selection, Student repository, Classification, Rule Generations.

1. INTRODUCTION
Feature extraction in knowledge and date engineering is the process of identifying and removing as much of the irrelevant and redundant information as possible. Regardless of whether a machine learning algorithm attempts to select features itself or ignores the issue, feature extraction prior to learning can be beneficial. Reducing the dimensionality of the data reduces the size of the hypothesis space and allows algorithms to operate faster. In some cases, accuracy on future classification can be improved; in others, the result needs to be more compact and can be interpreted more easily. Dash and Liu [6], Blum and Langley [4], and Hall and Holmes [8] presented a survey of the research on machine learning for feature extraction. In essence, many feature extraction methods model the task as a search problem, where each state in the search space specifies a distinct subset of the possible features. Dash and Liu categorize feature extraction into two major steps: generation procedure and evaluation function [6]. In the generation procedure, complete, heuristic, and random are different approaches for space searching. The searching space is to use a heuristic search procedure for an even medium number of features. Another important step in the feature selection is the evaluation function, which serves as the criterion in evaluating the relative merit of alternative feature subsets.

2 .FEATURE EXTRACTION TECHNIQUES
Feature extraction techniques can be categorized according to a number of criteria. One popular categorization consists of "filter" and "wrapper" to quantify the worth of features [5], [11]. Filters use general characteristics of the training data to evaluate attributes and operate independently of any learning algorithm. Wrappers, on the other hand, evaluate attributes by using accuracy estimates provided by the actual target learning algorithm. Due to the fact that the wrapper model is computationally expensive [13], the filter model is usually a good choice when the number of features becomes very large. Das [5] combined both models into a hybrid one to improve the performance of a particular learning algorithm. In this paper, we focus on the filter model and present a novel feature extraction algorithm which can effectively remove both irrelevant and redundant information.

Evaluation of individual feature emphasizes the relevance of the feature to the final decision. There are two typical individual feature-based evaluation approaches. The first one is information-based feature ranking. In this approach, the mutual information between decision and feature is used to evaluate the importance of the feature with respect to the decision under consideration. This method is independent of the underlying distribution and especially efficient when the data sets have a sheer dimensionality. The second type of algorithms relies on the relevance evaluation of features such as Relief which is an instance-based feature ranking scheme introduced by Kira and Rendell [10], and ReliefF, which can handle multiple class data, is enhanced by Kononenko [12] from Relief. The rationale of Relief and ReliefF is that a useful feature should differentiate between instances from different classes and have the same value for instances from the same class. The Relief approach is based

1. Calculates the mutual information between the feature X_i and the decision Y, $I(Y; X_i)$.

2. Generating relevant features set R by comparing the mutual information $I(Y; X_i)$

   if $I(Y; X_i) \geq \delta_1$ then $R \leftarrow R + \{X_i\}$.

3. Creates working set W by copying R.

4. Creates goal set $G = null$.

5. While $e(G) < \delta_2$ do

6. if $W = null$ then break.

7. choose $X_k \in W$ that subjects to

8. (a) $I(Y; X_k) \geq I(Y; X_l) \quad \forall l \neq k, X_l \in W$

9. (b) $Q_Y(X_k, X_n) \leq Q_Y(X_m, X_n) \quad \forall m \neq k, X_m \in W, \forall n, X_n \in G$

10. remove X_k from the working set $W \leftarrow W - \{X_k\}$ and put X_k into the target set $G \leftarrow G + \{X_k\}$

11. End loop

Figure 1: Feature Extraction Algorithm

on randomly sampling a number ðmÞ of instances from the training data set and then locating each feature's nearest neighbor from the same and opposite class. The values of the features of the nearest neighbors are compared to the sampled instance and used to update relevant scores for each feature. Although feature extraction techniques that focus only on relevance can significantly reduce the number of features to be considered, it could not help remove the redundant information existing among multiple features. Hall [7] and Kohavi and John [11] show that redundant features, along with irrelevant features, severely affect the accuracy of the learning algorithms. The reason is that if we do not consider the dependency among features, the feature selection algorithm will select multiple highly correlated features. Our results show that the linear summation of the individual mutual information values with respect to a particular decision will not linearly decrease the uncertainty in the decision because of the dependency that exists between features. Subset searching algorithms search through candidate feature subsets guided by a certain evaluation measure which captures the goodness of each subset. Some evaluation measures that have been effective in removing both irrelevance and redundancy include consistency measure [5], [2], [14] and correlation measure [7], [8]. The consistency method looks for the minimum combinations of features that could divide the training data into subsets containing a strong single class majority. This separation is hoped to be as consistent as the whole set of features. Correlation-based feature selection evaluates subsets of features rather than individual features. The ideal subsets should contain features that are highly correlated with the decision and have low level inter correlation with each other.

2.1 Feature Extraction Algorithm

The algorithm is based on the decision dependent correlation (DDC) measure discussed in the previous section. The goal of the feature selection algorithm is to select the minimum set of features that are strongly related to the desired decision variable and have the least redundancy among them. The algorithm shown in Fig. 1 consists of two functional modules. The first one focuses on removing irrelevance. We use a user defined threshold to determine which feature is relevant to the final decision (lines 1 and 2). In this part of the algorithm, irrelevant features are removed from the original feature set. The second part focuses on eliminating =redundancy from the features to be selected (line 3). We quantify a final state criterion as the distance of subset evaluation metric eðSÞ from the user defined threshold (line 5). For each pass, the feature Xk is chosen which satisfies two conditions simultaneously. The first one is that feature Xk should be the most relevant one compared with the rest of features in the working set (line 8). The second one is that feature Xk should have the least correlation with all the features in goal set G when compared with the other features in the working set W (line 9).

3. METHODOLOGY

3.1 GA for Feature Extraction

Genetic Algorithms (GA) have been shown to be an effective tool to use in data analysis and attern recognition [1], [2], [3]. An important aspect of GAs in a learning context is their use in pattern recognition. There are two different approaches to applying GA in pattern recognition:

1. Apply a GA directly as a classifier. Bandyopadhyay and Murthy in [4] applied GA to find the decision boundary in N dimensional feature space.
2. Use a GA as an optimization tool for resetting the parameters in other classifiers.

Most applications of GAs in pattern recognition optimize some parameters in the classification process. Many researchers have used GAs in feature selection [5], [6], [7], [8]. GAs have been applied to find an optimal set of feature weights that improve classification accuracy. First, a traditional feature extraction method such as Principal Component Analysis (PCA) is applied, and then a classifier such as k- NN is used to calculate the fitness function for GA [9], [10]. Combination of classifiers is another area that GAs have been used to optimize. Kuncheva and Jain in [11] used a GA for selecting the features as well as selecting the types of individual classifiers in their design of a Classifier Fusion System. GA is also used in selecting the prototypes in the case-based classification [12].

3.2 GA Classification Ensembles

Pattern recognition has a wide variety of applications in many different fields, such that it is not possible to come up with a single classifier that can give good results in all cases. The optimal classifier in every case is highly dependent upon the problem domain. In practice, one might come across a case where no single classifier can achieve an acceptable level of accuracy. In such cases it would be better to pool the results of different classifiers to achieve the optimal accuracy. Every classifier operates well on different aspects of the training or test feature vector. As a result, assuming appropriate conditions, combining multiple classifiers may improve classification performance when compared with any single classifier.

We used the simple genetic algorithm (SGA), which is described by Goldberg in [14]. The SGA uses common GA operators to find a population of solutions which optimize the fitness values. We used the *Stochastic Universal Sampling* [14] as our selection method, mainly due to its popularity and functionality. A form of

stochastic universal sampling is implemented by obtaining a cumulative sum of the fitness vector, *FitnV*, and generating *N* equally spaced numbers between 0 and sum(FitnV). Thus, only one random number is generated, all the others used being equally spaced from that point. The index of the individuals selected is determined by comparing the generated numbers with the cumulative sum vector. The probability of an individual being selected is then given by

$$F(x_i) = \frac{f(x_i)}{\sum_{i=1}^{N_{ind}} f(x_i)}$$

where *f(xi)* is the fitness of individual *xi* and *F(xi)* is the probability of that individual being selected.

The operation of *crossover* is not necessarily performed on all strings in the population. Instead, it is applied with a probability *Px* when the pairs are chosen for breeding. We selected *Px* = 0.7 since this would preserve a reasonably high level of the original population. Intermediate recombination combines parent values using the following formula [15]:

Offspring = parent1 + Alpha × (*parent2 – parent1*)

where Alpha is a scaling factor chosen uniformly in the interval [-0.25, 1.25].

A further genetic operator, *mutation* is applied to the new chromosomes, with a set probability *Pm* as the rate of mutation. Mutation causes the individual genetic representation to be changed according to some probabilistic rule. Mutation is generally considered to be a background operator that ensures that the probability of searching a particular subspace of the problem space is never zero. This has the effect of tending to inhibit the possibility of converging to a local optimum, rather than the global optimum. We considered *Pm* = 1/800 as our mutation rate, due to its small value with respect to the population. The mutation of each variable is calculated as follows:

Mutated Var = Var + MutMx × *range* × *MutOpt(2)* × *delta*

where delta is an internal matrix which specifies the normalized mutation step size; MutMx is an internal mask table; and MutOpt specifies the mutation rate and its shrinkage during the run.

During the reproduction phase, each individual is assigned a *fitness value* derived from its raw performance measure given by the objective function. This value is used in the selection to bias towards more fit individuals. Highly fit individuals, relative to the whole population, have a high probability of being selected for mating whereas less fit individuals have a correspondingly low probability of being selected. The error rate is measured in each round of cross validation by dividing "the total number of misclassified examples" into "total number of test examples". Therefore, our *fitness function* measures the accuracy rate achieved by classification fusion and our objective would be to maximize this performance (minimize the error rate).

4. RESULTS

Without using GA, the overall results of classification performance on our dataset for four classifiers and lassification fusion are shown in the Table 1. Regarding individual classifiers, 1NN and *k*NN have the best performance in the case of 2-, 3-, and 9- Classes, of approximately 62%, 50% and 35% accuracy, respectively. However, the classification fusion improved the classification accuracy significantly in all three cases. That is, it achieved 72% accuracy in the case of 2-Classes, 59% in the case of 3-Classes, and 43% in the case of 9-Classes

For GA optimization, we used 200 individuals (weight ectors) in our population, running the GA over 500 generations. We ran the program 10 times and obtained the averages, which are shown, in Table 2.

95% confidence interval. For the improvement of GA over non-GA result, a Pvalue indicating the probability of the Null-Hypothesis (There is no improvement) is also given, showing the significance of the GA optimization. All have $p<0.001$, indicating significant improvement. Therefore, using GA, in all the cases, we got more than a 10% mean individual performance improvement and about 11 to 16% best individual performance improvement.

Table 1: Classification Comparison without GA

Classifier	**2-Classes**	**3-Classes**	**9-Classes**
Bayes	52.6	38.8	22.1
1NN	**62.1**	45.3	29.0
*k*NN	55.0	**50.6**	**34.5**
Parzen	59.7	42.9	22.6
Classification Fusion	***72.2***	***58.8***	***43.1***

The Classifiers Bayes, 1NN, kNN and Parzen of 2,3,and 9 classes folding and its fusions compared using the feature extraction techniques.

Table 2: Optimization with GA and without GA Comparison

Classifier	2-Classes	3-Classes	9-Classes
Classification fusion of 4 Classifiers without GA optimization	71.19 ± 1.34	58.92±1.36	42.94 ±2.06
GA Optimized Classification Fusion, Mean individual (not best value)	81.09 ± 2.42	70.13 ± 0.89	55.25 ± 1.03
Improvement of Mean individual	***9.82 ± 1.33***	***11.06 ± 1.84***	***12.71 ±0.75***

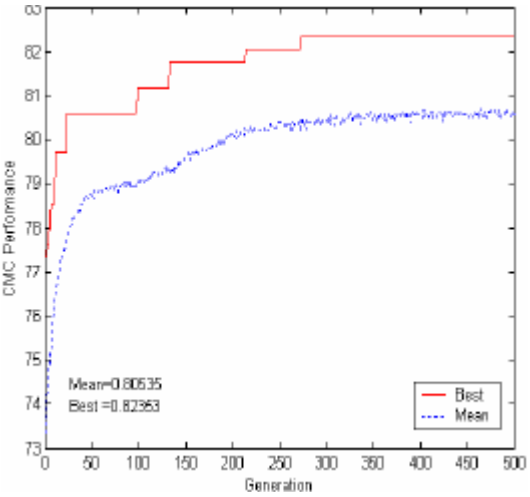

Figure-1: GA Optimized Performance for 200 samples

Finally, we can examine the individuals (weights) for features by which we obtained the improved results. This feature weighting indicates the *importance* of each feature for making the required classification. In most cases the results are similar to Multiple Linear Regressions or some tree-based software (like CART) that use statistical methods to measure feature importance. The GA feature weighting results, as shown in Table 6, state that the "Success with high number of tries" is the most important feature in all three cases. The "Total number of correct answers" feature is also important in some cases.

5. CONCLUSIONS AND FUTURE WORK

We proposed a new approach to classifying student data for any institution. Four classifiers are used in grouping the students. A combination of multiple classifiers leads to a significant accuracy improvement in the 2-, 3- and 9-Class cases. Weighing the features and using a genetic algorithm to minimize the error rate improves the prediction accuracy by at least 10% in the all three test cases. In cases where the number

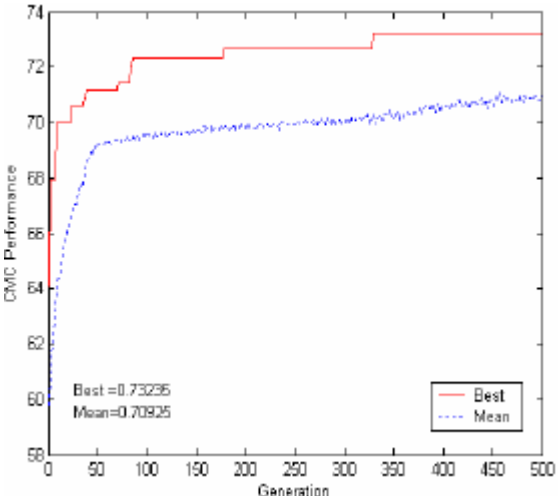

Figure-2: GA Optimized Performance for 500 Samples

of features is low, feature weighting worked much better than feature selection. The successful optimization of student classification in all three cases demonstrates the merits of using the system data to *predict* the students' final grades based on their features, which are extracted from the homework data. This approach is easily adaptable to different types of courses, different population sizes, and allows for different features to be analyzed. This work represents a rigorous application of known classifiers as a means of analyzing and comparing use and performance of students who have taken a technical course that was partially/completely administered via the system.In the present work, we propose an approach for predicting students' performance based on extracting the average of feature values over all of the problems in a course.

For future work, we plan to implement such an optimized assessment tool for every student on any particular problem. Therefore, we can track students' behaviors on a particular problem over several semesters.

6. REFERENCES

[1] A. Al-Ani and M. Deriche, "Feature Selection Using a Mutual Information Based Measure," Proc. 16th Int'l Conf. Pattern Recognition, vol. 4, pp. 82-85, 2002.

[2] H. Almuallim and T.G. Dietterich, "Learning with Many Irrelevant Features," Proc. Ninth Nat'l Conf. Artificial Intelligence, pp. 547-552, 1991.

[3] R. Battiti, "Using Mutual Information for Selecting Features in Supervised Neural Net Learning," IEEE Trans. Neural Networks, vol. 5, pp. 537-550, 1994.

[4] A. Blum and P. Langley, "Selection of Relevant Features and Examples in Machine Learning," Artificial Intelligence, vol. 97, nos. 1-2, pp. 245-271, 1997.

[5] S. Das, "Filters, Wrappers and a Boosting-Based Hybrid for Feature Selection," Proc. 18th Int'l Conf. Machine Learning, pp. 74-81, 2001.

[6] M. Dash and H. Liu, "Feature Selection for Classification," Intelligent Data Analysis: An Int'l J., vol. 1, pp. 131-156, 1997.

[7] M.A. Hall, "Correlation-Based Feature Selection for Discrete and Numeric Class Machine Learning," Proc. 17th Int'l Conf. Machine Learning, pp. 359-366, 2000.

[8] M.A. Hall and G. Holmes, "Benchmarking Attribute Selection Techniques for Discrete Class Data Mining," IEEE Trans. Knowledge and Data Eng., 2002.

[9] K.A. De Jong, "An Analysis of Behavior of a Class of Genetic Adaptive Systems," PhD Dissertation, Dept. of Computer and Comm. Sciences, Univ. of Michigan, 1975.

[10] K. Kira and L.A. Rendell, "A Practical Approach to Feature Selection," Proc. Ninth Int'l Workshop Machine Intelligence, 1992.

[11] R. Kohavi and G. John, "Wrappers for Feature Subset Selection," Artificial Intelligence, pp. 273-324, 1997.

[12] I. Kononenko, "Estimating Attributes: Analysis and Extensions of Relief," Proc. Seventh European Conf. Machine Learning, pp. 171-182, 1994.

[13] P. Langley, "Selection of Relevant Features in Machine Learning," Proc. AAAI Fall Symp. Relevance, 1994.

[14] H. Liu and R. Setiono, "A Probabilistic Approach to Feature Selection: A Filter Solution," Proc. 13th Int'l Conf. Machine Learning, pp. 319-327, 1996.

[15] J.A. Miller, W.D. Potter, R.V. Grandham, and C.N. Lapena, "An Evaluation of Local Improvement Operators for Genetic Algorithms," IEEE Trans. Systems, Man, and Cybernetics, vol. 23,pp. 1340-1351, Sept./Oct. 1993.

[16] Y. Peng and J.A. Reggia, "A Connectionist Model for Diagnostic Problem Solving," IEEE Trans. Systems, Man, and Cybernetics, vol. 19, pp. 285-298, Mar./Apr. 1989.

[17] W.H. Press, B.P. Flannery, S.A. Teukolski, and W.T. Vetterling,Numerical Recipes in C. Cambridge Univ. Press, http:// www.library.cornell.edu/nr/bookcpdf.html, 2005.

[18] J.R. Quinlan, C4.5: Programs for Machine Learning. San Mateo, Calif.: Morgan Kaufmann, 1993.

[19] L. Yu and H. Liu, "Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution," Proc. 20th Int'l Conf. Machine Learning (ICML-2003), 2003.

[20] L. Yu and H. Liu, "Efficient Feature Selection via Analysis of Relevance and Redundancy," J. Machine Learning Research, vol. 5, pp. 1205-1224, Oct. 2004.

[21] http://kdd.ics.uci.edu/databases/kddcup99/task.html, 2005.