EVOLUTIONARY ALGORITHM FOR CONSTRUCTING A DECISION TREE

Sultangaliev Amirkhan Manasovich
2nd year of Master's degree,Tashkent University of Information Technologies named after Muhammad Al-Khwarizmi,
amir_xan@list.ru

ABSTRACT

A new approach is proposed to solve the problem of data classification based on decision trees, using the genetic method of combining heuristics. This approach made it possible to improve the accuracy of classification, while retaining all the advantages of the decision tree method.

Keywords: forecasting, data classification, decision trees, genetic algorithms, heuristics combination method.

INTRODUCTION

Forecasting is used to support decision making in various information systems, such as business applications, workflow systems, e-learning systems with a high degree of adaptability, etc.. Its particular cases are the problems of data classification and regression recovery. The main difference between them is that in the first case the prediction result belongs to the set of non-overlapping classes, and in the second it is a real number. One way or another, in both cases, for successful prediction, the system must be trained using a test sample of data. This article proposes an efficient data classification algorithm using decision trees and genetic algorithms.

Decision trees are a way of representing rules in a hierarchical, sequential structure, where each object has a single node that gives a decision. At the edges of the tree, the attributes on which the objective function depends are written, in the leaves - the values of the objective function, and in the remaining nodes - the attributes by which the cases are distinguished.

The MAIN IDEA of constructing decision trees from some training set **X**, formulated in the interpretation of R. Quinlan [1], is as follows.

Let a set of examples **X*, X*** ^ **X** be concentrated at some node of the tree. In this case, three situations are possible.

1. The set **X*** contains one or more examples belonging to the same class $\mathbf{y_k}$. Then the decision tree for **X*** is a leaf that defines the class $\mathbf{y_k}$.

2. The set **X*** does not contain a single example, that is, it is an empty set. Then it is again a leaf, and the class associated with the leaf is chosen from another set than **X*** .

3. The set **X*** contains examples belonging to different classes. In this case, the set **X*** should be divided into some subsets. To do this, one of the features **j** is selected, which has two or more different values, and **X*** is divided into new subsets, each of which contains all examples that have a certain range of values of the selected feature. The procedure will continue recursively until any subset **X*** consists of examples belonging to the same class.

In general, the heuristic algorithm for constructing a decision tree will be as follows.

1. Selection of a splitting criterion in order to find the most appropriate attribute to check at each node of the tree.

2. Dividing the sample into two or more parts according to the values of the attribute selected based on the split criterion.

3. Recursion starting from step 2. The resulting subsets are divided into smaller subsets in accordance with the selected criterion until each of them contains objects of the same class or attributes that allow these objects to be distinguished.

These steps are common to most existing decision tree building algorithms. These algorithms differ from each other in the choice of the separation criterion, as well as in the mechanism for pruning tree nodes and other parameters.

Some methods use the so-called information content measure of attribute subspaces to select the splitting attribute, which is based on the entropy approach and is known as the information gain measure, or entropy measure. A similar approach is applied in such algorithms as ID3 and C4.5. In accordance with this criterion, the feature that gives the maximum information about the classes is considered the best for separation. This value is determined by the formula for the amount of information:

Gain(Y | X) = H(Y) - H(Y | X), (1)

where **H(Y)** is the entropy of the set **Y**; **H(Y|X)** - the average conditional entropy of the set **Y** with known set **X**. The quantities **H(Y)** and **H(Y|X)** are determined by the formulas:

$$\mathbf{H(Y)} = -\sum_i \mathbf{p(y_i)} \cdot \mathbf{\log_2 p(y_i)} \quad (2)$$

$$\mathbf{H(Y \mid X)} = -\sum_i \mathbf{p(x_i)} \cdot \mathbf{H(Y|x_i)} \quad (3)$$

where $\mathbf{p(x_i)}$ and $\mathbf{p(y_i)}$ are the probabilities of choosing one or another value from the sets **X** and **Y**, respectively; $\mathbf{H(Y|x_j)}$ is the conditional entropy if it is known that the value $\mathbf{x_j}$ is chosen from **X**. Conditional entropy is determined by the formula

$$\mathbf{H(Y \mid x_i)} = -\sum_j \mathbf{p(y_j)} \cdot \mathbf{\log_2 p(y_j \mid x_i)} \quad (4)$$

When dividing the sample, an attribute is always selected that gives the maximum gain of information for the target attribute, that is, for which the value of **Gain(Y|X)** is the maximum among all **X**. Here **X** is the set of attribute values of the classified objects, **Y** is the set of target attribute values.

The above formulas allow you to calculate the amount of information for attributes representing discrete (or rather, nominal) values. In practice, integer and even real attributes are often encountered. It would be most acceptable to treat any attribute as nominal and calculate the relative frequencies (as probability estimates) for each value encountered, however, such an approach will lead to uncontrolled growth of the tree and the loss of its generalizing ability. One of the solutions to this problem is the use of the so-called threshold entropy, which is determined by the formula

$$\mathbf{H(Y|X:t)} = \mathbf{p(X < t)} \bullet \mathbf{H(Y \mid X < t)} + \mathbf{p(X \geq t)} \bullet \mathbf{H(Y \mid X \geq t)}. \quad (5)$$

Then the formula for the amount of information will take the form

$$\mathbf{Gain^*(Y \mid X) = \max, (H(Y) - H(Y \mid X: t))}. \quad (6)$$

Thus, a slightly modified criterion for the amount of information can be used for any attributes for which a comparison operation makes sense, in particular, for real and integer.

There is one significant problem with the criterion of the amount of information: the algorithm often selects the attributes that have the largest number of different values. For example, if one of the attributes in the data sample represents the date of an event, then it is likely that this attribute will be selected for splitting. In this case, the resulting decision tree will be completely useless, since dates that have already passed will never be repeated.

Another splitting criterion proposed by L. Breiman et al. [2] is implemented in the CART algorithm and is called the Gini index. With help this index attribute is selected based on distances between class distributions:

Gini(T) = 1 -$^{n}\Sigma_{i=1}p^{2}_{i}$, (7)

where $\mathbf{p_i}$ is the probability (relative frequency) of class **i** in sample **T**.

In addition to the above criteria, it is sometimes effective to select an attribute based on the number of different values in the sample. Splitting on the attribute that contains the maximum number of values in the sample is often effective for numeric attributes. In this case, the expectation of the set **X** can be used as the split value. Splitting by an attribute that contains the minimum number of values in the sample is also more efficient than the proposed criteria, especially when it comes to discrete (nominal) attributes. The split value in this case can also be chosen based on the distribution of values in the sample.

Despite their wide application, heuristic methods for constructing decision trees are fundamentally unable to find the most complete and accurate data classification rules, since they are based on the use of greedy algorithms. If an attribute was selected once and subsetting was performed on it, the algorithm cannot return and select another attribute that would give a better partition. Therefore, at the construction stage, it is impossible to say whether the selected attribute will give the optimal partition in the end.

Another significant drawback of heuristic algorithms is that the same criterion is used when splitting the sample, and this often leads to an incorrect choice of attribute and split value.

A significant increase in classification accuracy is possible if the process of constructing a decision tree is transformed into the problem of finding and using the optimal sequence of local objective functions used in the subproblems of splitting the data sample. This implies that the transformed problem will be solved by evolutionary methods. In particular, this paper considers the application of a genetic algorithm.

A similar approach has already been successfully applied to scheduling problems and is called the Heuristics Combination Method (HCM). This article illustrates the possibilities and features of using NSM for the optimal selection of heuristics when building a decision tree.

In genetic methods, any solution to the problem of synthesis is represented by a chromosome consisting of genes. Alleles are the values of design parameters .

In accordance with the proposed algorithm, in each subtask of the original problem associated with the selection of the sample, a separation criterion is selected according to one of the following heuristics:

S1: the attribute with the highest **Gain(Y|X)** index value is selected;

S2: the attribute with the highest **Gini(Y)** index value is selected;

S3: the attribute containing the largest number of values in the data sample is selected;

S4: the attribute with the fewest values in the data sample is selected;

S5: No selection split.

The chromosome in this case has a tree structure. The number of genes in it coincides with the number of tree nodes; their values can be heuristic numbers in the range **[1, 5]**.

The unusual structure of the chromosome affects the implementation of the genetic algorithm. The stage of evolution in this case is to choose the most appropriate heuristic for the corresponding tree node. In this case, individuals with different heuristics in child nodes are used. Next, the sample is split depending on the chosen heuristic and a new stage of evolution is performed for the slave nodes. A new population is formed from the child nodes of the tree, which reduces the size of both the chromosome and the data sample.

To ensure a sufficient diversity of individuals, when forming the initial population, decision trees are built for each of the heuristics in the range **[S1:S4]**. This uses different heuristics in the child nodes. Thus, the initial population size is 16 individuals.

The crossover operator for tree chromosomes is executed for a specific data subset corresponding to a given tree node and consists of the following steps.

1. Two chromosomes with the same genes are selected corresponding to the root node of the tree of the considered subset of data.

2. In each of the parental chromosomes, the values of the genes corresponding to the first and second subordinate nodes of the tree are swapped.

One of the main limitations of the considered crossover operator is the need to match the heuristics for the considered tree node. An equally important condition is the number of child nodes. According to the algorithm, they change places only the first two child nodes in the parent chromosomes. The rest remain unchanged.

A chromosome mutation is a random change in the heuristic for the tree node in question. Since the main purpose of the mutation is to ensure the diversity of the population, any of the chromosomes can be changed, provided that there is a similar one in the population. The optimal number of individuals to perform mutation **k** is calculated based on the diversity of the population and the number of individuals in it:

$$\mathbf{k = max}\ (\mathbf{1}, \frac{n}{4}),$$

where **l** is the number of pairs of identical chromosomes; **n** is the size of the population.

As can be seen from formula (8), the number of individuals for mutation should not exceed 25% of the total population size. Otherwise, this may violate the convergence of the genetic algorithm and lead to the loss of the local extremum of the objective function **f**.

The mutation algorithm consists of the following steps.

1. Search in the population for a pair of chromosomes with the same heuristic value in all child nodes.

2. Selection from a pair of one chromosome (randomly) for mutation.

3. Replacing the heuristic at the root node of the selected chromosome (a new kind of heuristic is also chosen randomly).

4. If the number of individuals for mutation is less than **k** and the population has the same chromosomes, return to step 1.

In the process of selecting individuals, the strategy of elitism is used. At the same time, the population size remains unchanged, that is, 16 individuals with the highest classification accuracy are selected from the intermediate population. The work of the algorithm stops after all stages of evolution have been completed. Additionally, a restriction on the maximum number of generated generations can be introduced.

To test the algorithm, a training sample of 3212 lines was used, containing information about the executive discipline of employees, collected on the basis of the organization's workflow data. The elements of the object set **X** include the following attributes.

1. Type of control - can take one of two values: "Control over the department" or "Control over the performer".

2. Contractor - contains the name of the department or full name. the executor to whom the resolution on the document is assigned.

3. The month in which the document was sent to the executor.

4. Day of the week on which the document was sent to the executor.

The set of responses **Y** is information about the violation of the deadline for the implementation of the resolution. To increase the accuracy of classification, the values of the **Y** set are represented not by a number, but by an interval of numbers, which makes it possible to attribute different values of the target attribute to the same class.

RESULTS AND DISCUSSION

The result of the algorithm execution is presented in the table. For convenience of perception, the error rate is expressed as a percentage. As can be seen, the proposed genetic algorithm demonstrates a higher accuracy of data classification, unlike any of the heuristic algorithms.

Classification results for a sample of 3212 rows

Classification algorithm	Error rate, %
Criterion Gain	24.12
Gini criterion	25.01
Split by maximum number of values	26.53
Separation by the minimum number of values	25.36
Algorithm built into the program 1C: Enterprise	24.84
Genetic Algorithm	19.46

CONCLUSION

The use of evolutionary methods eliminates the disadvantage of heuristic algorithms associated with the wrong choice of the separation criterion when constructing a decision tree, which, in turn, improves the accuracy of data classification and the quality of forecasting in general.

REFERENCES

[1] Ross J. Quinlan. C4.5: Programs for Machine learning. Morgan Kaufmann Publishers, 1993.

[2] Breiman L., Friedman J.H., Olshen R.A. and Stone C.T. Classification and Regression Trees. Wadsworth, Belmont, California, 1984.

[3] Norenkov I.P. Genetic methods of structural synthesis of design solutions // Information technologies. 1998. No. 1. S. 9-13.

[4] Goldberg D. Genetic Algorithms in Search, Optimization, and Machine Learning // Adison-Wesley Publ., 1989..