CS 529 CLASS PROJECT: IMPLEMENTATION OF ID3 DECISION TREE

Farhan Asif Chowdhury[1]
Md Amanul Hasan[2]
September 16, 2018

Abstract

In this class project, we used the machine learning algorithms to develop a way to recognize the boundary types (EI, IE or N = neither) using a given sequence of DNA. We have a training data set that has a total of two-thousand DNA sequences with known boundary types. Each DNA sequence has a total of sixty attributes with any of the eight unique values (A, G, T, C, D, N, S, and R). Since later four are very rare (less than 1%) in our data set, we eliminated the sequences which have those values from the training data to make our analysis simple. Here, the ID3 algorithm was used to develop the decision tree by splitting the data at the most suitable attribute in the sequence of the data by computing entropy (or Gini-index) criteria. Chi-square test was used to evaluate the potentiality of further splitting decision each time by comparing the test-statistics with critical values for given confidence levels (99%, 95% or 0%). Finally, the efficiency of our developed algorithm was evaluated by uploading the outputs for the testing dataset in Kaggle for each specified condition. The best accuracy we got using the entropy criterion for 95% confidence level is 91.81%.

Keywords: *Machine learning, ID3 algorithm, Decision tree, Entropy, Gini-index, Chi-square.*

[1]Graduate student, Department of Computer Science, University of New Mexico, Albuquerque, email: asifchowdhury@unm.edu
[2]Graduate student, Department of Civil Engineering, University of New Mexico, Albuquerque, email: amanulhasan@unm.edu
Both authors have contributed equally.

1 INTRODUCTION

Machine learning is a computer-based statistical approach that has the ability to extract a generalized knowledge from a large set of known data (training data) and to apply this learned knowledge for making predictions on an unknown data (testing data) [1]. In this class project, we used the machine learning algorithms to develop a way to recognize the boundaries between exon/intron (EI) and intron/exon (IE) using a given sequence of DNA. The exons are the parts of the DNA sequence retained after splicing during the process of protein creation in higher organisms, where the introns are the parts of the DNA sequence that are spliced out. In our problem, we have a training data set that has a total of two-thousand DNA sequences with known boundary types. The boundary types can be divided into three different types (EI, IE, or N = neither). On the other hand, each DNA sequence has a total sixty attributes that can be divided into eight unique types (A, G, T, C, D, N, S, and R). Since the variables in the data set are categorical, decision tree based on the ID3 algorithm is viable to solve this problem. Here, two different approaches: entropy, and Gini-index, were used to select the most appropriate attribute in the sequence of the attributes for splitting the data at each level in the decision tree. Again, for being the categorical data set, Chi-square test was used to evaluate the potentiality of further splitting decision each time by comparing the test-statistics with critical values for given confidence levels (99%, 95%, and 0%). Later, the developed decision tree was used to predict the boundary types for a testing dataset that has a total of eleven hundred and ninety DNA sequences. Finally, the efficiency of our developed algorithm was evaluated by uploading the outputs in Kaggle for each specified condition. Detailed ID3 algorithm description, our implementation description, accuracies of results for the all specified conditions, and explanations of the results are given in following sections.

2 DESCRIPTION OF ALGORITHM

2.1 ID3 Algorithm

For creating the decision tree, we have used the ID3 algorithm. It takes input of all the training data in its root node and from there it recursively creates children node and subtree. In the beginning of the subtree creation method it checks whether all the current data has same class label, and if all of them has same class then the node will be marked as a leaf node and will be given a class label. Then the algorithm checks if there is any attribute left to be used as the splitting attribute. If there is no attribute left then the algorithm will mark the node as a leaf node and will be given a class label, the class which occurs the most among the given set of training data given to that node. If the set of data has different classes and there are attributes left to continue splitting the data, the algorithm selects one attribute as the splitting attribute which gives the maximum information gain after performing the splitting using corresponding attribute. Information gain can be calculated using either ‘Entropy’, ‘Gini-Index’ or ‘Misclassification Error’. In our

implementation we have used both ‘Entropy’ and ‘Gini-Index’ for information gain calculation and the user can chose to use any of these tow for the information gain calculation.

We have also used ‘Chi-square’ test after selecting the splitting attribute to check whether there is any significance of performing the split. If for a selected confidence level of chi-square test, the chi-square value is appropriate the algorithm continues to split the current node, otherwise, stops splitting the current node and marks the node as a leaf node and gives it a class label. If the chi-square test value is positive for splitting, the algorithm creates new child node from the current node for each of possible attribute values of the selected attribute. When creating a new child of the current node the algorithm partitions the current data set based on their matched attribute values. If for a particular attribute value, there is no data available, the algorithm labels the corresponding node as a leaf node and gives it a class. On the contrary, where there are matched data set for a particular attribute value, the algorithm recursively creates subtree from that node and the whole process mentioned above starts from the beginning for that node. Fig. 1 shows the flow chart of used develop algorithm for this project.

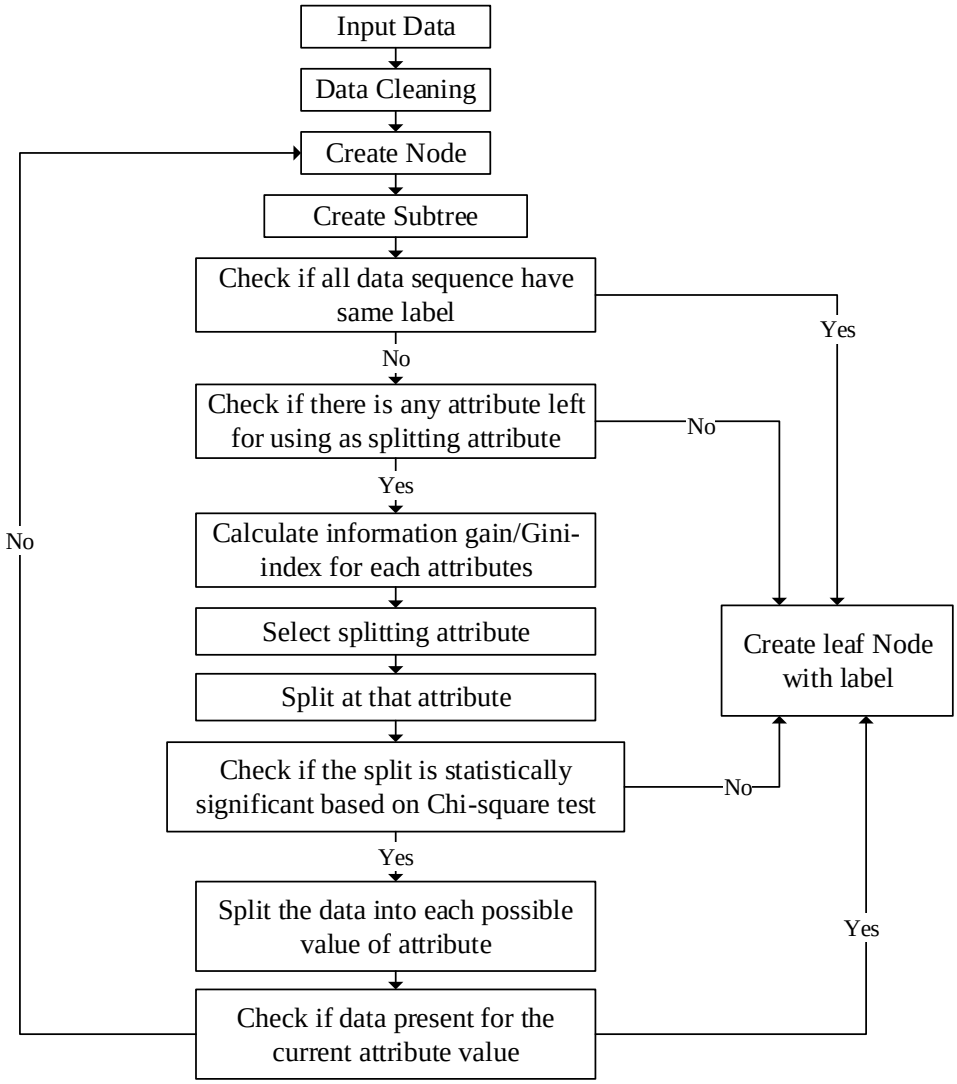

Fig. 1: Flow chart of the developed algorithm for decision tree

2.2 Data Structure for Implementation of ID3

To implement the ID3 algorithm we have taken an object-oriented programming method and used python (version 3.6.2) as our programming language. Our whole program is built using two class, one is the 'decision_tree' class and the other is 'tree_node' class. The 'decision_tree' class is used to create a decision tree object which will store our created decision tree model and can be used for classification of new test data set.

We have created a 'tree_node' class which is used to store the information of nodes and how they are connected to other nodes in order to form the whole decision tree. Each 'tree_node' object stores whether the node is leaf node, if it is a leaf node then what is class label of that node and what is the attribute value of that node. If the node is not a leaf node then it stores which attribute is used for splitting that node, what are the children nodes of that node and what is the attribute

value of that node. This tree_node class helps to store all the connectivity information among the nodes and their class label to be used in the new test data classification ion stage.

2.3 Selection of the most appropriate attribute for splitting

To create the decision tree, it is essential to select the most appropriate attribute for splitting the data. The ID3 algorithm uses some statistical properties utilizing information theory to measure how well a given attribute separates the training examples according to their target classification [2]. The most suitable attribute to split the data is the attribute that gives the highest information gain. There are different approaches used in ID3 algorithms, however, we used only the entropy and the Gini-index as per our project requirement.

2.3.1 Entropy/Information gain

The entropy also known as information gain is an impurity-based criterion that uses the entropy of the data as the primary criterion [3]. If S is a dataset of any node that has c no of different attributes, then, the entropy of S can be calculated using Eq. 1.

$$Entropy(S) = -\sum_{i=1}^{c} p_i \log_2 p_i \tag{1}$$

Where p_i is the proportion of S that belongs to i^{th} attribute. Finally, the information gain, $IG_{(S,A)}$ of an attribute A can be computed using Eq. 2.

$$IG_{(S,A)} = Entropy(S) - \sum_{\upsilon \in values(A)} \frac{S_\upsilon}{S} Entropy(S_\upsilon) \tag{2}$$

Where S_υ is the subset of S for which attribute A has value of ν. The best attribute for splitting is the one with the greatest $IG_{(S,A)}$.

2.3.2 Gini-index

The Gini-index is also an impurity-based criterion, but it uses the divergences between the probability distributions of the different attributes in a dataset [4]. If S is a dataset of any node that has c no of different attributes, then, the Gini-index of S can be calculated using Eq. 3.

$$Gini(S) = 1 - \sum_{i=1}^{c} p_i^2 \tag{3}$$

Where p_i is the proportion of S that belongs to i^{th} attribute. Finally, the Gini gain, $GG_{(S,A)}$ of an attribute A can be computed using Eq. 4.

$$GG_{(S,A)} = Gini(S) - \sum_{\upsilon \in values(A)} \frac{S_\upsilon}{S} Gini(S_\upsilon) \quad (4)$$

Where S_υ is the subset of S for which attribute A has value of ν. The best attribute for splitting is the one with the greatest $GG_{(S,A)}$.

2.4 Split stopping criterion (Chi-square test)

To prevent overfitting, Chi-square test based split stopping criterion was employed in our algorithm. Overfitting means that the model performs well when classifying examples that it has seen before however, but may perform poorly for new or unseen examples. Since the variables in our dataset are categorical, Chi-square test is the most suitable and easiest statistical method to determine the statistically significance of further splitting of the data at any node [5]. Our program calculates the Chi-square statistic using the dataset at each node and compares with the statistical value with the critical value for given the confidence level. If the calculated value is greater than the critical value, then the program chooses to split the data, otherwise, it stops further splitting. If a node has *N* no of samples and *n* no of classes, the Chi-square statistic for further splitting into *m* no of attributes can be calculated using Eq. 5.

$$\chi^2 = \sum_{j}^{m} \sum_{i}^{n} \frac{(N_{i,j}^{observed} - N_{i,j}^{expected})^2}{N_{i,j}^{expected}} \quad (5)$$

Where $N_{i,j}^{observed}$ and $N_{i,j}^{expected}$ are the observed and the expected no of samples who are in i^{th} class and j^{th} attribute. The $N_{i,j}^{expected}$ can be calculated using Eq. 6.

$$N_{i,j}^{expected} = \frac{N_i^{total} * N_j^{total}}{N} \quad (6)$$

Where N_i^{total} and N_i^{total} is the no of samples at i^{th} class and at j^{th} attribute respectively. As there is a chance to have zero in denominator ($N = 0$), we always add 0.0001 with N to avoid infinite value (or error in result). Since we have three classes and four attributes, we used the degree of freedom, dof as (3-1)*(4-1) or 6 for our program. Based on this dof, the critical values were selected as per the given confidence limits. For example, for 95% confidence level, the critical value is 12.592 for $dof = 6$.

3 RESULTS AND ACCURACIES

The developed decision tree based on the training data was used to predict the boundary classification of testing data. As per the project requirement, the accuracy of the outputs was evaluated using both entropy and Gini-index at three different confidence levels (99%, 95% and

0%). The results are summarized in Table 1. The accuracies for both methods are almost same at each confidence level, however, entropy is slightly more accurate than Gini-index. This dissimilarity may have occurred due to applying different equations for these two cases. While comparing the accuracy levels among the confidence level, both 95% and 99% confidence levels provide more accurate results compared to 0% confidence level. This is due to overfitting the model at 0% confidence level. However, we are not sure why 95% confidence level is more accurate than 99% confidence level as this situation is opposite to the expectation based on the theory.

Table 1: Summary of accuracy for different conditions.

Confidence levels	Entropy	Gini-index
99%	91.39%	90.97%
95%	91.81%	91.81%
0%	89.71%	89.50%

4 DISCUSSIONS

In this project, we implemented the ID3 Decision Tree algorithm to create a decision tree which can successfully determine the boundary classes using the DNA sequence data. Both the entropy and the Gini-index criteria were used to find the most appropriate attribute to split dataset. Also, the chi-square test was used to evaluate the necessity of further splitting requirement. Three different confidence levels were employed to see the accuracy of the developed algorithm by uploading the outputs into Kaggle. Findings are given as follows:

- Both entropy and the Gini-index criteria show almost similar accuracy at each confidence level. However, the entropy criterion performs slightly better than the Gini-index criterion.
- The best accuracy we found using entropy approach with the 95% confidence level as 91.81%.

REFERRENCES

[1] T. M. Mitchell et al., Machine Learning. wcb, 1997.

[2] Maimon O., Rokach L., “Data Mining and Knowledge Discovery Handbook”, vol. 2, Springer, 2010.

[3] J. R. Quinlan, “Induction of decision trees,” Machine Learning, vol. 1, no. 1, pp. 81–106, 1986.

[4] Gelfand S. B., Ravishankar C. S., and Delp E. J., “An Iterative Growing and Pruning Agorithm for Classification Tree Design”. IEEE Transaction on Pattern Analysis and Machine Intelligence, 13(2):163-174, 1991.

[5] James, G., Witten, D., Hastie, T., & Tibshirani, R., An Introduction to Statistical Learning, vol. 112, New York, Springer, 2013.