A Bayes Network Based Classification Approach for Evaluation of Success of Software Reuse

Amanpreet Singh, Prof. Amanpreet Singh Brar, Dr. Parvinder Singh Sandhu

Abstract—This paper presents bayes network based classification method to evaluate the reusable software components and in identification of reusable components from existing legacy systems. The bayes network classifier algorithm takes a database and an attributes ordering as input and constructs a belief network structure as output. So many discrepancies exist between expert opinion and empirical data reported in Morisio et.al.’s recent TSE article. But the result of this evaluation depends on the probability of different instances. We find some difference related to factors that makes the success of software reuse. This implementation describes how those differences are detected and how the instances give true positive value and accuracy.

Keywords—Bayes Network, Software reuse, TP, Accuracy.

I. INTRODUCTION

Classification is a basic method in software reusability and data analysis that requires the construction of classifiers, i.e. a function that assigns a class to instances described by a set of attributes. In the recent years the researchers of software reusability gives Success and Failure Factors in Software Reuse sought key factors that predicted for successful software reuse. Their data comes from the high level management of a 24 department of a company. Bayes networks (BNs), also known as belief networks belong to the family of probabilistic graphical models (GMs). These graphical structures are used to represent knowledge about an uncertain domain. In particular, each node in the graph represents a random variable, while the edges between the nodes represent probabilistic dependencies among the corresponding random variables. These conditional dependencies in the graph are often estimated by using known statistical and computational methods. Hence, BNs combine principles from graph theory, probability theory, computer science, and statistics [6,7]. A Bayesian network $B = \langle N, A, \theta \rangle$ is a directed acyclic graph (DAG) $\langle N, A \rangle$ where each node $\langle n \in N \rangle$ represents a domain variable (eg, a dataset attribute), and each arc $a \in A$ between nodes represents a probabilistic dependency, quantified using a conditional probability distribution $\theta_{n}$ for each node $n$. A BN can be used to compute the conditional probability of one node, given values assigned to the other nodes; hence, a BN can be used as a classifier that gives the posterior probability distribution of the class node given the values of other attributes. A major advantage of BNs over many other types of predictive models, such as neural networks, is that the Bayesian network structure represents the inter-relationships among the dataset attributes. Human experts can easily understand the network structures and if necessary modify them to obtain better predictive models. By adding decision nodes and utility nodes, BN models can also be extended to decision networks for decision analysis [11]. In case of the two-cluster based problem, the confusion matrix has four categories: True positives (TP) are Projects correctly classified as Successful cases of software Reuse. False positives (FP) refer to Unsuccessful or Failure projects incorrectly labeled as Success. True negatives (TN) correspond to Unsuccessful or Failure projects correctly classified as such. Finally, false negatives (FN) refer to Successful Projects incorrectly classified as failure as shown in table 4.1.

<table>
<thead>
<tr>
<th>TABLE I</th>
<th>CONFUSION MATRIX OF PREDICTION OUTCOMES.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predicted Project</td>
<td>Real Data Value of Project Status</td>
</tr>
<tr>
<td>Success</td>
<td>failure</td>
</tr>
<tr>
<td>Success</td>
<td>TP</td>
</tr>
<tr>
<td>failure</td>
<td>FN</td>
</tr>
</tbody>
</table>

With help of the confusion matrix values the precision and recall values are calculated described below:

A. Precision

The Precision is the proportion of the examples which truly have class x among all those which were classified as class x. The technique having maximum value of probability of detection and lower value of probability of false alarms is chosen as the best prediction technique.
Precision for a class is the number of true positives (i.e. the number of items correctly labeled as belonging to the positive class) divided by the total number of elements labeled as belonging to the positive class (i.e. the sum of true positives and false positives, which are items which were not labeled as belonging to the positive class but should have been). The equation is:

\[ \text{Precision} = \frac{TP}{(TP + FP)} \]  

(1)

B. Recall

Recall in this context is defined as the number of true positives divided by the total number of elements that actually belong to the positive class (i.e. the sum of true positives and false negatives, which are items which were not labeled as belonging to the positive class but should have been). The recall can be calculated as follows:

\[ \text{Recall} = \frac{TP}{(TP + FN)} \]  

(2)

C. Accuracy

The percentage of the predicted values that match with the expected values for the given data. The best system is that having the high Accuracy, High Precision and High Recall value.

II. BAYES NETWORKS AS CLASSIFIERS

Probabilistic models have become increasingly popular in the last decade because of their ability to capture non-deterministic relationships among variables describing many real world domains. Among these models, graphical models have received significant attention because of their ability to compactly encode conditional independence assumptions over random variables and because of the development of effective algorithms for inference and learning based on these representations [3].

A. Complement Naive Bayes

This refers to class for building and using a Complement class Naive Bayes classifier. Naive Bayes is often used as a baseline in text classification because it is fast and easy to implement. It not only gives us better efficiency but also adversely affects the quality of its results. Naive Bayes classifiers addressing, both systemic issues as well as problems that arise because text is not actually generated according to a multinomial model. Reason behind Naive Bayes poor performance is that for every problem there is simple heuristic solution.

B. Naive Bayes

It is a class for a Naive Bayes classifier using estimator classes. Numeric estimator precision values are chosen based on analysis of the training data. For this reason, the classifier is not an Updateable Classifier (which in typical usage is initialized with zero training instances), if needed the Updateable Classifier functionality, use the Naive Bayes Updateable classifier. The Naive Bayes Updateable classifier will use a default precision of 0.1 for numeric attributes when “build Classifier” is called with zero training instances.

C. Naive Bayes Multinomial

This is a class for building and using a Multinomial Naive Bayes classifier. This is fast, easy to implement and relatively effective. Multinomial Naive Bayes models the distribution of words in a document as a multinomial. A document is treated as a sequence of words and it is assumed that each word position is generated independently of every other. Over simplification McCallum and Nigam (1998) posit a Multinomial Naive Bayes model for text classification and show improved performance compared to the multivariate Bernoulli model due to the incorporation of frequency information.

D. Naive Bayes Updateable

Bayesian Networks face the problem of how to handle continuous variables most previous work has either solved the problem by discretizing or assumed that the data are generated by a single Gaussian. It is observed with large reductions in
error on several natural and artificial data sets which suggests that kernel estimation is a useful tool for learning Bayesian models the data come from is much simpler and much older approach probabilistic induction known as Naive Bayes classifier, describe Flexible Bayes (Buntine, 1996), class for a Naive Bayes classifier using estimator classes. This is the updateable version of Naive Bayes. This classifier will use a default precision of 0.1 for numeric attributes when “build Classifier” is called with zero training instances

IV. METHODOLOGY

The main objective of the thesis is:

1. Perform the literature survey of the factors for the success of software reuse proposed by different researchers.
2. Select the attributes needed for evaluation successfulness of software reuse.

The dataset used in the paper is as follows;

- There are three types are factors that are considered here:
  - a) high-level control variables
  - b) state variables
  - c) low-level control variables

Here are the "high-level control variables" i.e. key high-level management decisions about a reuse program

- Top management commitment for the top management reuse program? Expressed in :(Yes, No)
- Whether Key reuse roles are introduced?
- Whether Reuse process Expressed in :(Yes, No)
- If Non-reuse process modified? Expressed in :(Yes, No)
- Whether Human factors handled; e.g. via awareness, training, and motivation programs? Expressed in :(Yes, No)
- If Repository have assets? Expressed in :(Yes, No)

V. RESULTS

The following are details of the "high-level control variables" i.e. key high-level management decisions about a reuse program for Top management commitment, Key reuse roles, Reuse process, Non reuse process modified, Human factors and Repository assets respectively. When 10 fold cross validation is performed Correctly Classified Instances are found to be 24 instances i.e. 100% and incorrectly Classified Instances are 24 i.e. 100%. The Mean absolute error and Root mean squared error are found to be 0.0245 and 0.066 respectively.

VII. CONCLUSION

Reuse based approaches emphasize cost reduction as a means of increasing productivity. From an accounting perspective there are different ways of achieving this. One way is the amortization of the development and maintenance cost of assets over multiple projects. Another way is the avoidance of cost in later projects through the use of results of earlier projects. As evidenced by the results, Bayes Network Based Classification algorithm is proved to be best as compared to the earlier algorithms used in the literature for evaluating the success of software reuse in an organization. The MAE and RMSE values calculated are 0.0245 and 0.066 respectively, those are also satisfactorily low. Hence, it is concluded that for non linear and complex engineering applications involving decision and analysis, the Bayes Network Based Classification algorithm is an efficient technique.

VI. DISCUSSIONS

Bayes network analysis of software reusability is different then others due to so many reasons. Firstly, our evaluated data is published data and not the managers interviews in their analysis. This data is drawn after the discussion of different project managers rather than conclusions drawn from their automatic. Secondly, our analysis of the data uses different machine learner techniques. In this analysis we use bayes network classification to find out the root mean square error. However, there are major differences in the other machine learners we used. For example, TAR2 is a recently invented learner by Menzies & Hu [17]. TAR2’s report of differences between classes is a novel and succinct method of isolating the key factors that can most change a situation.

TABLE II

CONFUSION MATRIX

<table>
<thead>
<tr>
<th>Predicted</th>
<th>Real Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Success</td>
</tr>
<tr>
<td>Success</td>
<td>TP=15</td>
</tr>
<tr>
<td>Failure</td>
<td>FN=0</td>
</tr>
</tbody>
</table>

REFERENCES

[10] Tim Menzies, Justin S. Di Stefano, More Success and Failure Factors in Software Reuse, SUBMITTED TO IEEE TRANS. SOFT. ENG. VOL. XX, NO. Y, MONTH.


