DATA MINING

Assignment 1

Classification Trees and Random Forests

Ziv Hochman, student number 8454434

Stylianos Psara, student number 2140527

Panagiotis Andrikopoulos, student number 1780743

The data:

For this assignment the Eclipse bug data set was used to predict whether any post-release bugs have been reported. The data consists of one file for each Eclipse release 2.0, 2.1, 3.0 for the package levels. However, for the purpose of this assignment, only two files were used. The 2.0 for training and the 3.0 for testing.
The purpose of the data set is to be able to give insight into software defects. according to the [paper].

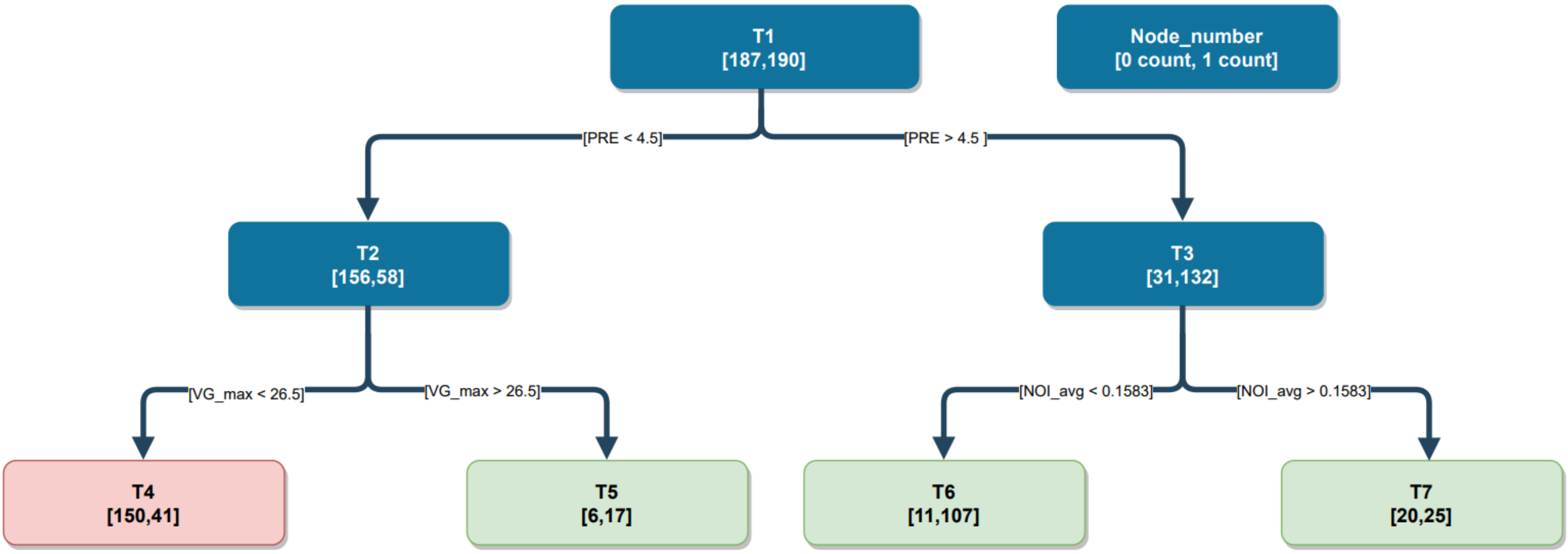
Each record in the data consists of the following information:

- Name: The name of the package for the corresponding case.
- Pre-release defects: The number of non-trivial defects that were reported in the last six months before release.
- Post-release defects: The number of non-trivial defects that were reported in the first six months after release.
- Complexity metrics: Metrics that are computed for classes or methods are aggregated by using average (AVG), maximum (MAX), and accumulation (SUM) to package level.
- Structure of abstract syntax trees: For each entry, the size (=number of nodes) of the abstract syntax tree(s) of the file or package is listed. Furthermore, frequency is listed for each of the different types of nodes.

To use the data for the training, some pre-processing had to be performed.
First, the predictor variables had to be extracted consisting of 41 features of the metrics and the number of pre-release defects for every package.

Then, for the labeling purposes the post-release defects value for each package was converted to binary variables -> (if post value is equal to 0 then it stays as 0, but if the post value is greater than 0, we assign 1) for each value in the post-release column.

Visualization of the single decision tree:



- At the root node (T1) the best split point according to our code is on the "Pre" column with the value 4.5.
- At the second node (T2) the best split point according to our code is on the "NOI_avg" column with the value ~0.183.
- At the third node (T3) the best split point according to our code is on the "VG_max" column with the value 26.5.

According to this heavily simplified tree above with considering the classification method as the majority class:

- T5, T6 and T7 are classifying the result as positive (1) - has a bug - (green).
- T4 classifying the result as negative (0) – doesn't have a bug - (red)

According to [paper] the high correlation between the number of pre-release and post- release shows that the packages with the most pre-release defects are more likely to also have the most post-release defects and vice versa. In our case this correlation indicates that the packages with pre-release value more than 4.5 (which means that they have at least five defect reports) are most likely to be defected after the release.

Furthermore, VG_max denotes the complexity of the packages; therefore, it makes sense that packages that have higher complexity result in more bugs than simpler packages.

Finally, NOI_avg denotes the number of interfaces. According to [paper] there is no correlation between the post-released bugs and the number of interfaces, that’s supported by the tree above (the split doesn’t give us new information about the post-release bugs).

Why ? becase both in left and right we have the same majority class (1) ?

Confusion Matrix and Quality Measures:

1. Single classification tree (nmin = 15, minleaf = 5, nfeat = 41):

Class\ Prediction	0	1
0	255	93
1	84	229

The accuracy score: 0.7322239031770046
The recall score: 0.731629392971246
The precision score: 0.7111801242236024

2. Classification trees with bagging method
(nmin = 15, minleaf = 5, nfeat = 41, m = 100):

Class\ Prediction	0	1
0	301	47
1	95	218

The accuracy score: 0.7851739788199698
The recall score: 0.6964856230031949
The precision score: 0.8226415094339623

3. Classification trees with random forest method:
(nmin = 15, minleaf = 5, nfeat = 6, m = 100):

Class\ Prediction	0	1
0	299	49
1	90	223

The accuracy score: 0.789712556732224
The recall score: 0.7124600638977636
The precision score: 0.8198529411764706

As we can observed the accuracy of the single classification tree is lower in comparison with the accuracy of the classification trees with the bagging or random forest methods.
This occurs because in the case of the single classification tree only one tree was used to classify the test set. On the other hand, Bagging and Random Forest combines the prediction of multiple trees (in our case m=100) and improves the overall accuracy.
Moreover, as we can see that the accuracies for Bagging and Random Forest trees are slightly different. (With the better results with the random forest method rather than the bagging method).
In addition, the bagging method has higher time complexity than the random forest method, given that the random forest chooses only six random attributes instead of all the 41 attribute (bagging) for each possible split point.

p-value < 0.05: This suggests that the difference between the two groups is statistically significant, meaning the difference is likely real and not just due to chance. In other words, there's less than a 5% probability that the observed difference happened by random variation. So, you can confidently say the two groups are different.

Statistically significant difference:

In our assignment we chose the McNemar's test. We chose this statistical function to distinguish whether there is a statistically significant difference between the accuracies of the different models (single tree, bagging trees and random forest trees). McNemar's test is a statistical technique utilized to assess changes or differences when comparing two models with the same test data with binary categorial variables. This technique was very useful for our case because we wanted to compare our three models' prediction accuracies pairwise using the eclipse-bug-3.0 as the test set.

Results:

After using the McNemar's test (with alpha = 0.05) we got the following results:

1. Between the single tree classification and the bagging classification trees we found that there is a statistically significant difference in proportions between the models with P value of 0.0138.
2. Between the single tree classification and the random forest classification trees we found that there is an extremely statistically significant difference in proportions between the models with P value of 0.0002.
3. Between the bagging classification trees and the random forest classification trees we found that there isn't a statistically significant in proportions between the models with P value of 0.0545.

Bibliography:

Thomas Zimmermann, Rahul Premraj, Andreas Zeller, "Predicting Defects for Eclipse"