Z Number: 23226341

Name: Carlos Gross-Martinez

Professor: Dr. Taghi M. Khoshgoftaar

Course: CAP6778 – Advanced Data Mining & Machine Learning

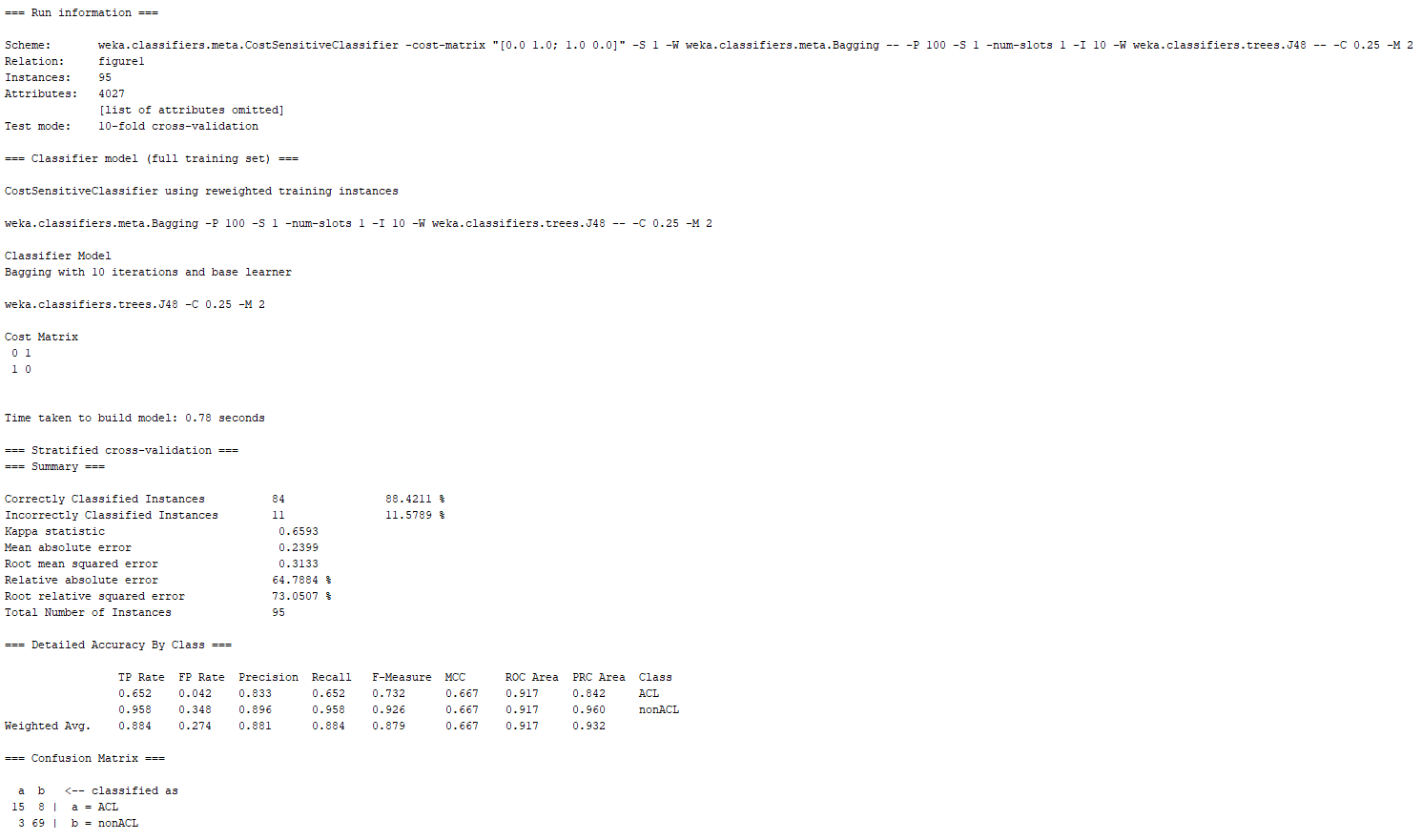
Assignment 2: Modeling Assignment: Using Meta Learning Schemes with a Strong and a Weak Learner for Classification

Dataset Analysis

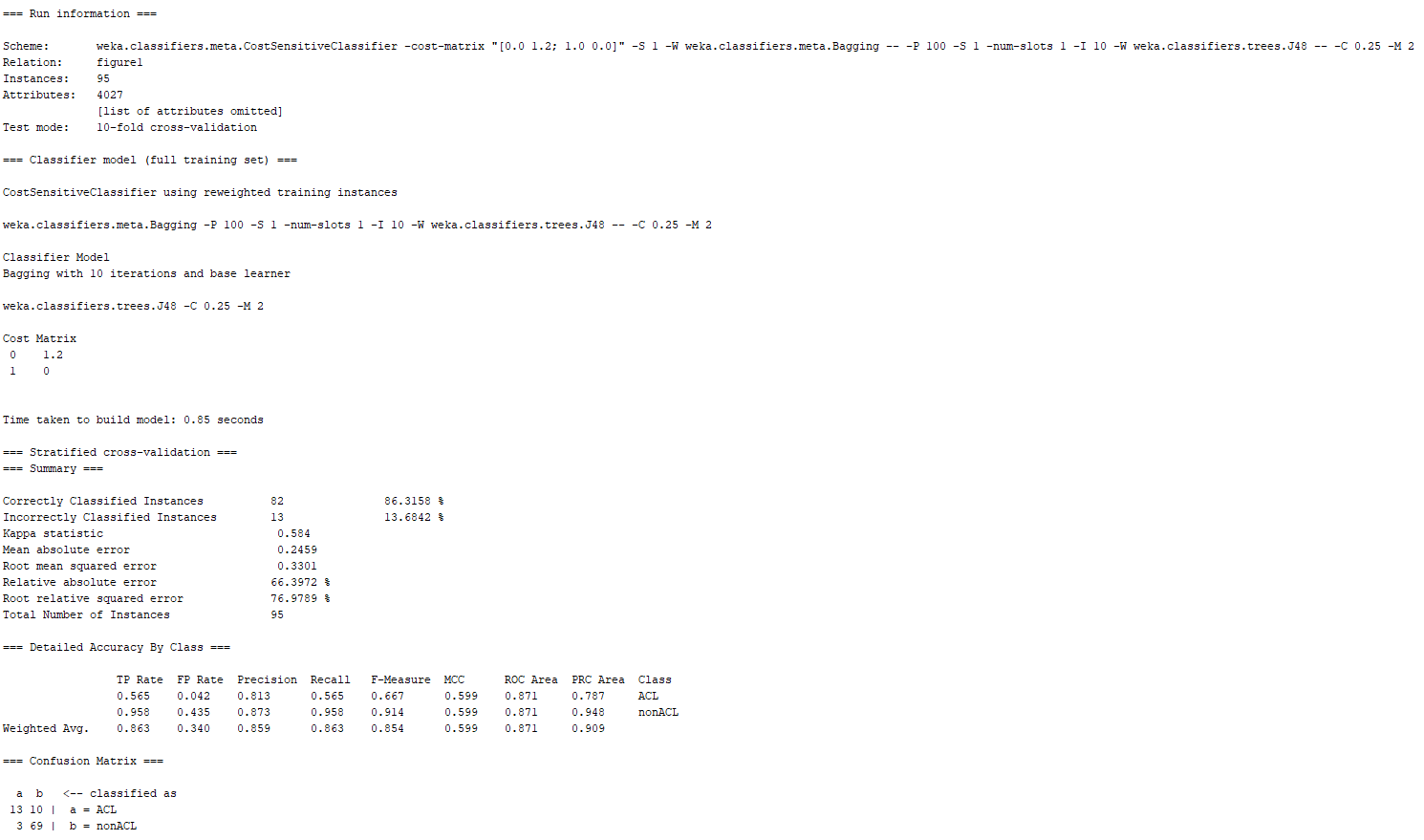
The dataset used in this assignment consisted of 95 instances/samples and 4027 attributes. From the 4027 attributes 4026 were input attributes to the model while 1 attribute was the class label. Furthermore, out of the 95 samples provided in the dataset, 23 were from the minority class labeled “ACL” while 72 were from the majority class labeled “nonACL”. With this information, it can be noted that percentage wise, the minority class represents of the full dataset, while the majority class represents of the full dataset. With this data distribution in mind is important to note that the data set contains contain high dimensionality due to the high number of attributes serving as an input as well as class imbalance, as the data contains a great number of samples for the secondary class “nonACL” while containing a smaller number of samples for the primary class “ACL”.

Cost Sensitive Classifier Combined with Bagging and J48

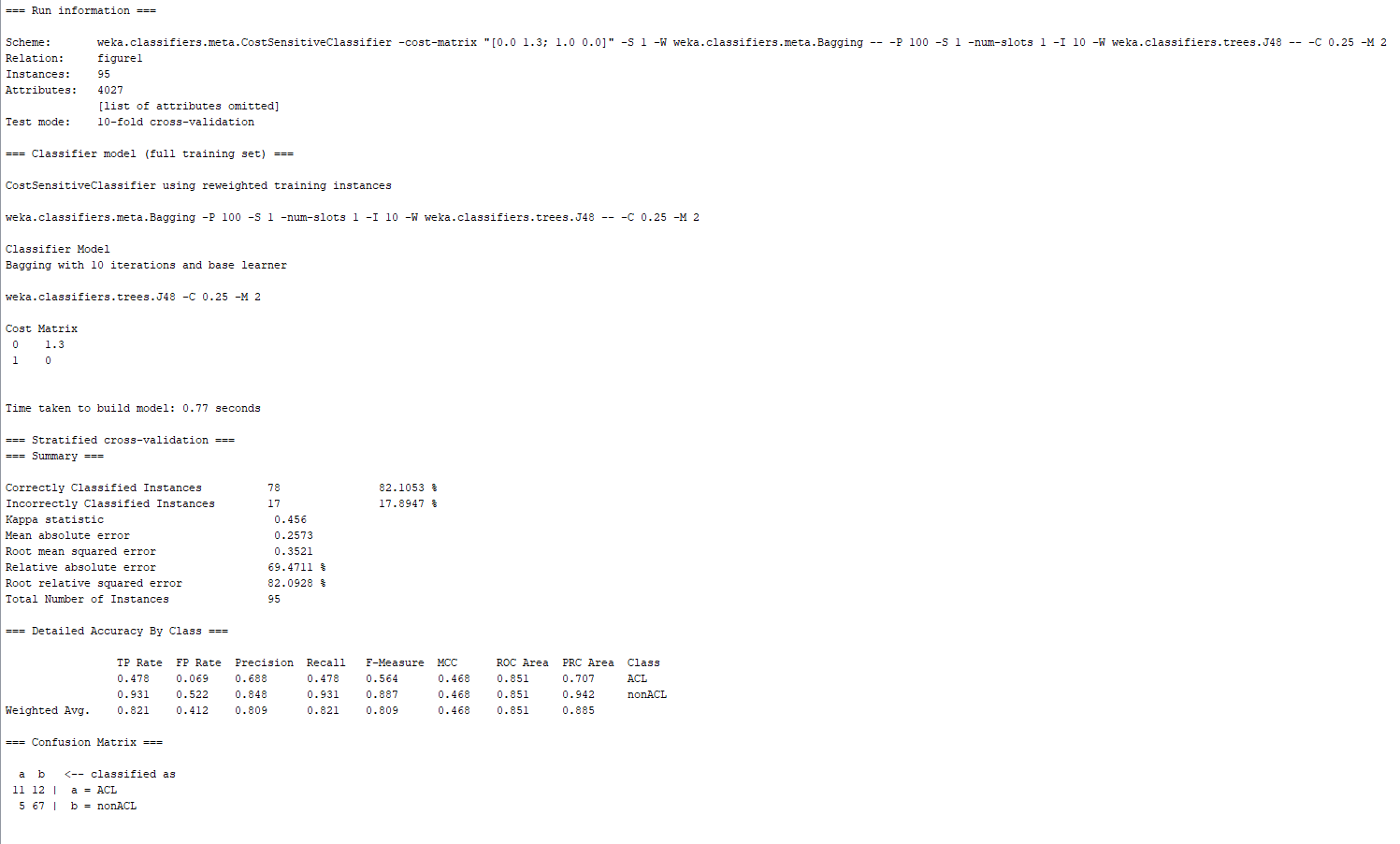
The Cost Sensitive Classifier Combined with Bagging and J48 ML model was trained using the following values for the cost sensitive matrix: 0.5, 0.6, 1, 1.01, 1.1, 1.2, 1.3, 1.4, 1.401, 1.5 and 2. The parameters for the classifier were the default settings for the classifier while only changing the cost sensitive matrix value to procure results. The most relevant matrix values were 1, 1.2, 1.3, and 1.4. A screenshot for these classifiers with their respective cost matrix can be seen below:



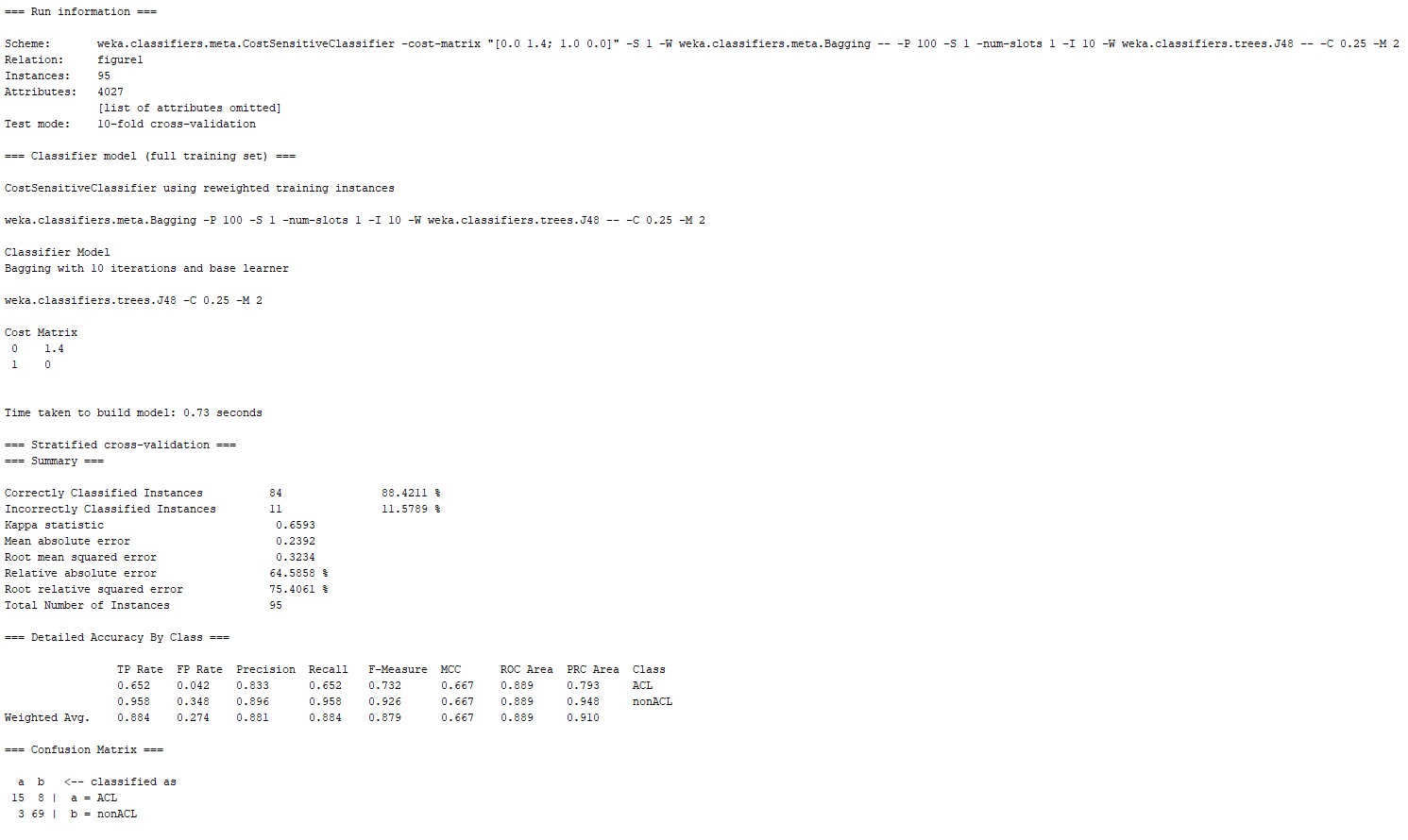
Picture 1: Cost Sensitive Classifier Combined with Bagging and J48 – Cost Matrix 1 – 10-Fold Cross Validation



Picture 2: Cost Sensitive Classifier Combined with Bagging and J48 – Cost Matrix 1.2 – 10-Fold Cross Validation

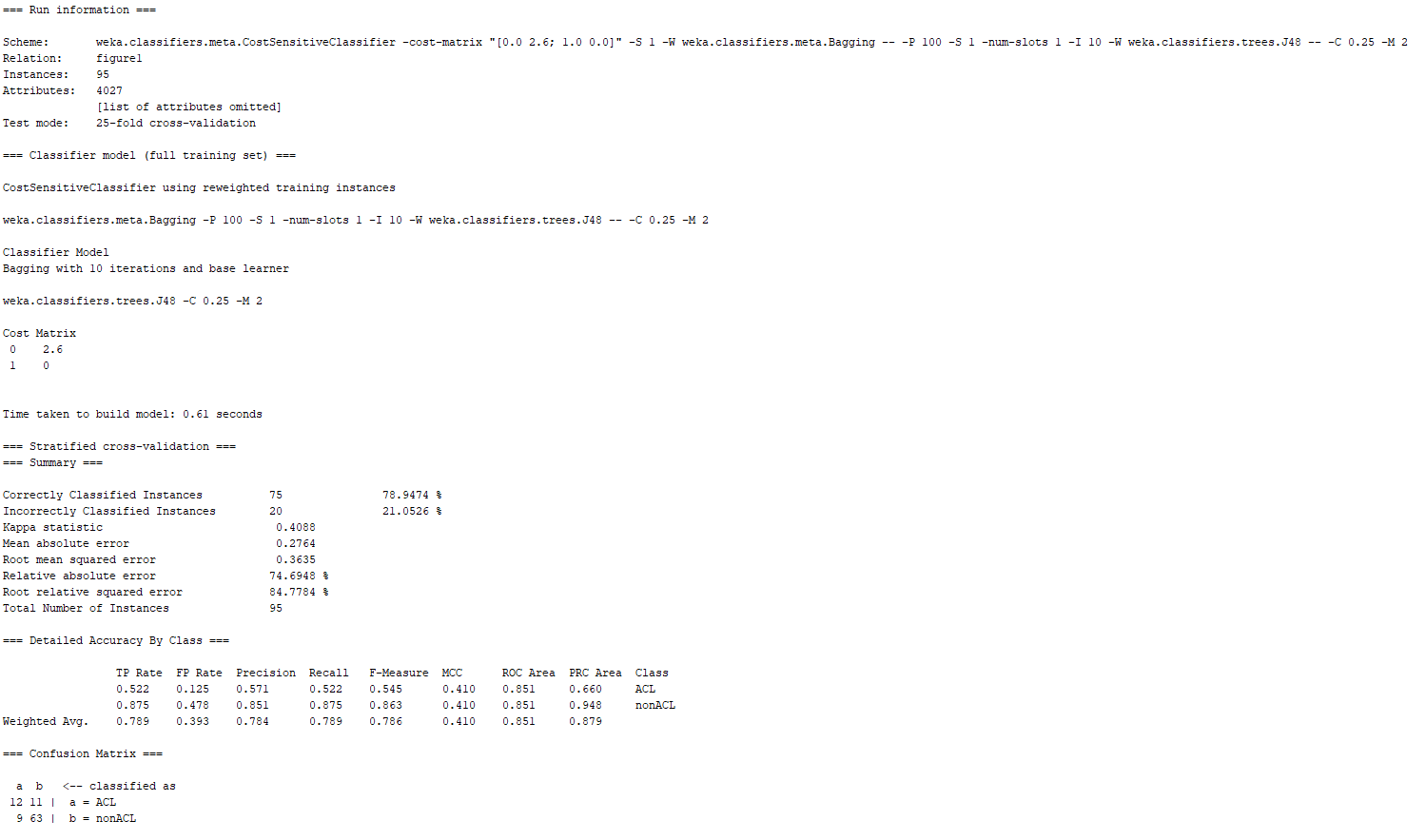


Picture 3: Cost Sensitive Classifier Combined with Bagging and J48 – Cost Matrix 1.3 – 10-Fold Cross Validation

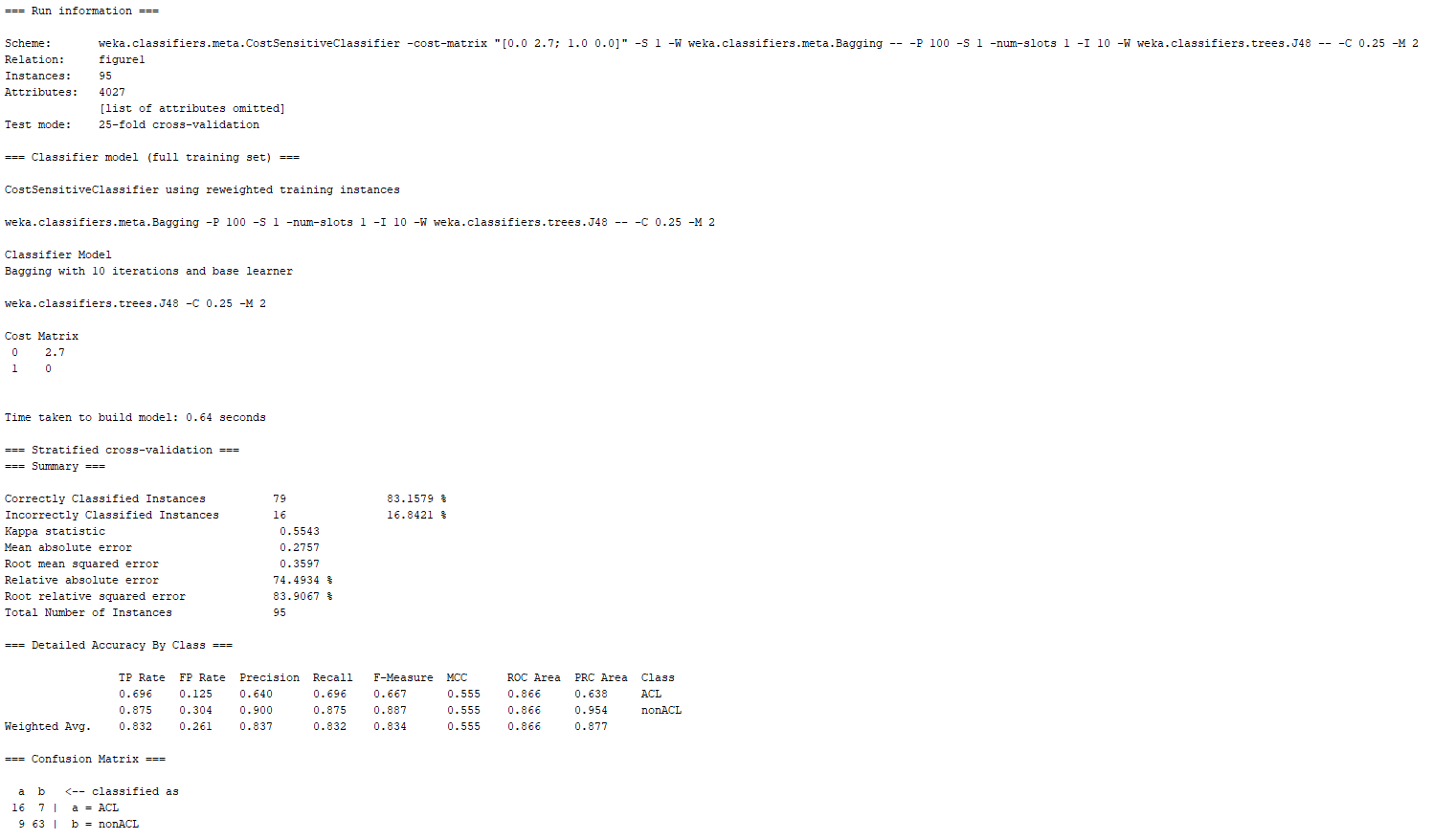


Picture 4: Cost Sensitive Classifier Combined with Bagging and J48 – Cost Matrix 1.4 – 10-Fold Cross Validation

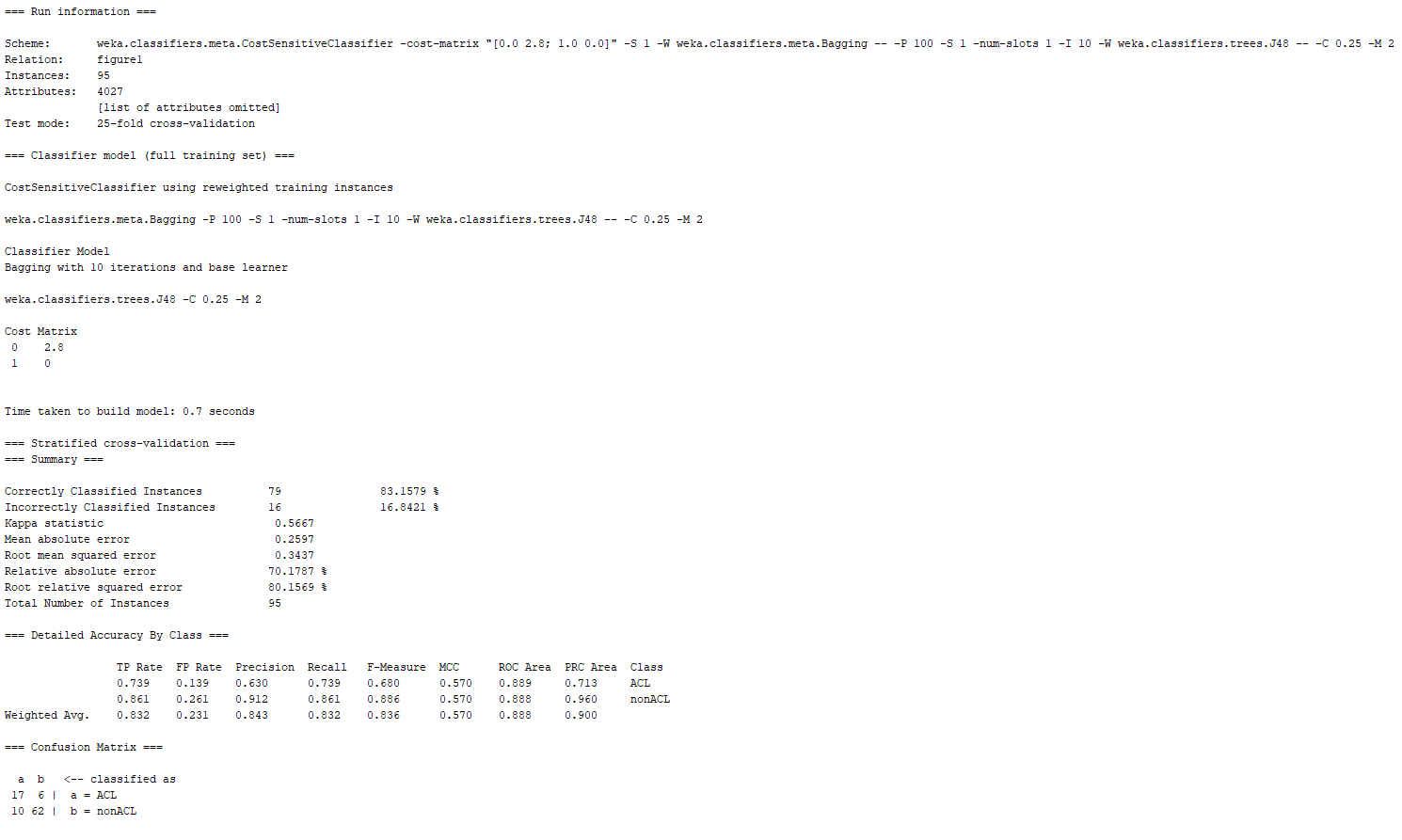
From these results, it can be noted that when the cost sensitive matrix values were 1 and 1.4, the model had the best performance with the lowest misclassification of the minority class. Nevertheless, none of the models trained were able to match the same efficiency as the cost sensitive tree obtained in Assignment 1 part 4. With this information in mind, the next step consists in training the model with 25-fold cross validation. The cost sensitive matrix: 0.5, 0.6, 1, 1.1, 1.2, 1.3, 1.4, 1.5, 1.9, 2, 2.1, 2.7, 2.8, 2.9, 3, 3.1, and 4. The most relevant matrix values were 2.6, 2.7, 2.8, and 2.9. A screenshot these classifiers with their respective cost matrix can be seen below:



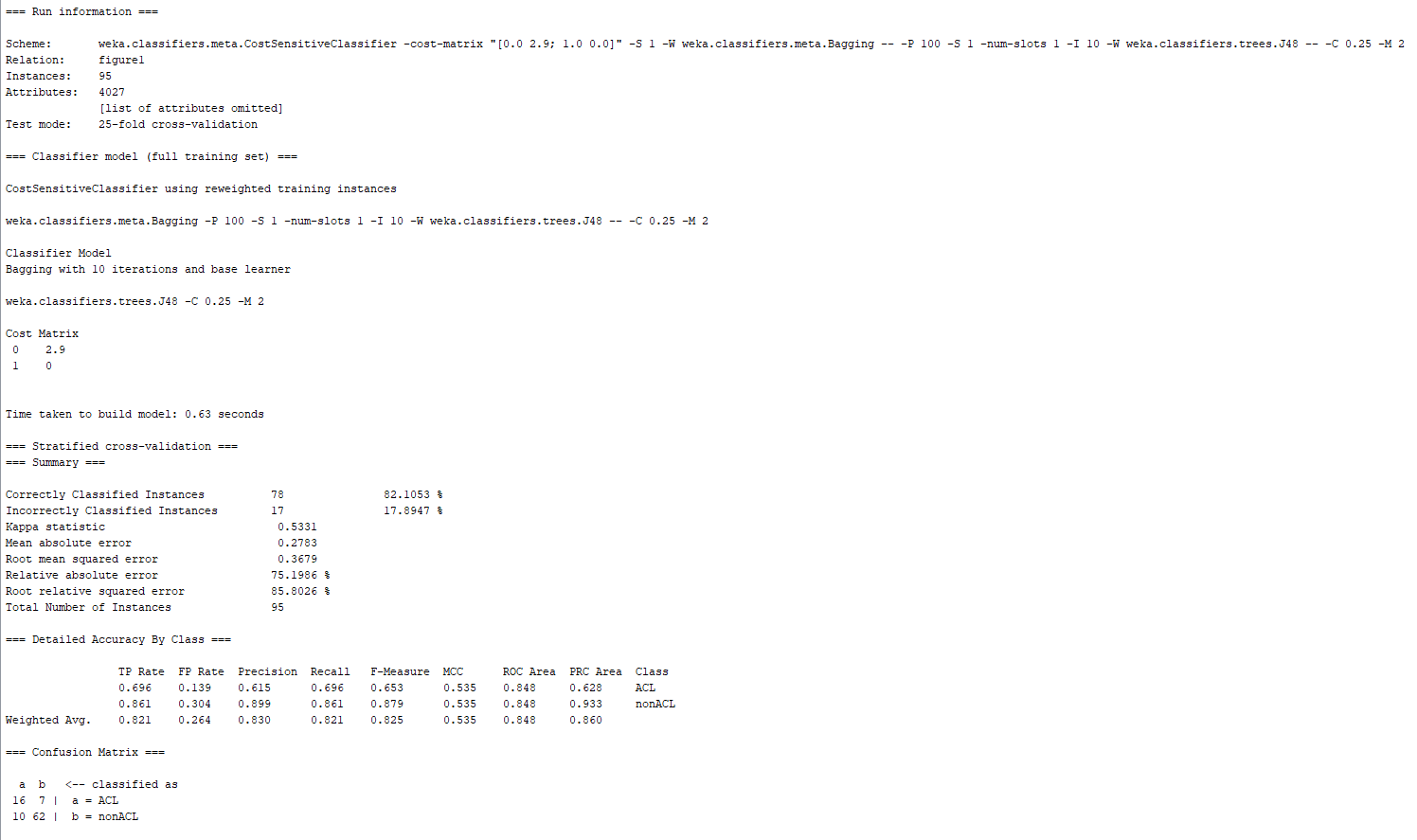
Picture 5: Cost Sensitive Classifier Combined with Bagging and J48 – Cost Matrix 2.6 – 25-Fold Cross Validation



Picture 6: Cost Sensitive Classifier Combined with Bagging and J48 – Cost Matrix 2.7 – 25-Fold Cross Validation



Picture 7: Cost Sensitive Classifier Combined with Bagging and J48 – Cost Matrix 2.8 – 25-Fold Cross Validation

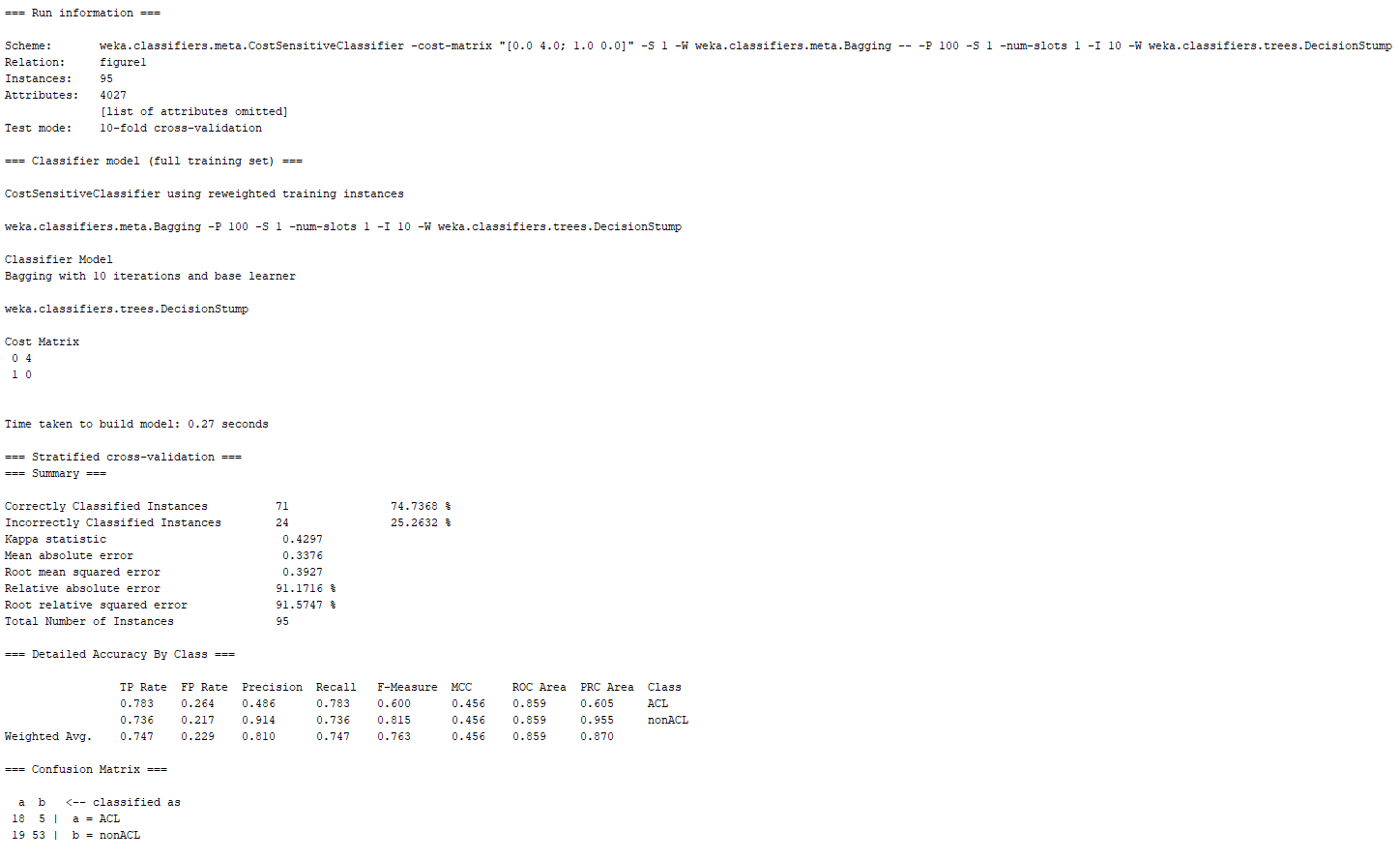


Picture 8: Cost Sensitive Classifier Combined with Bagging and J48 – Cost Matrix 2.9 – 25-Fold Cross Validation

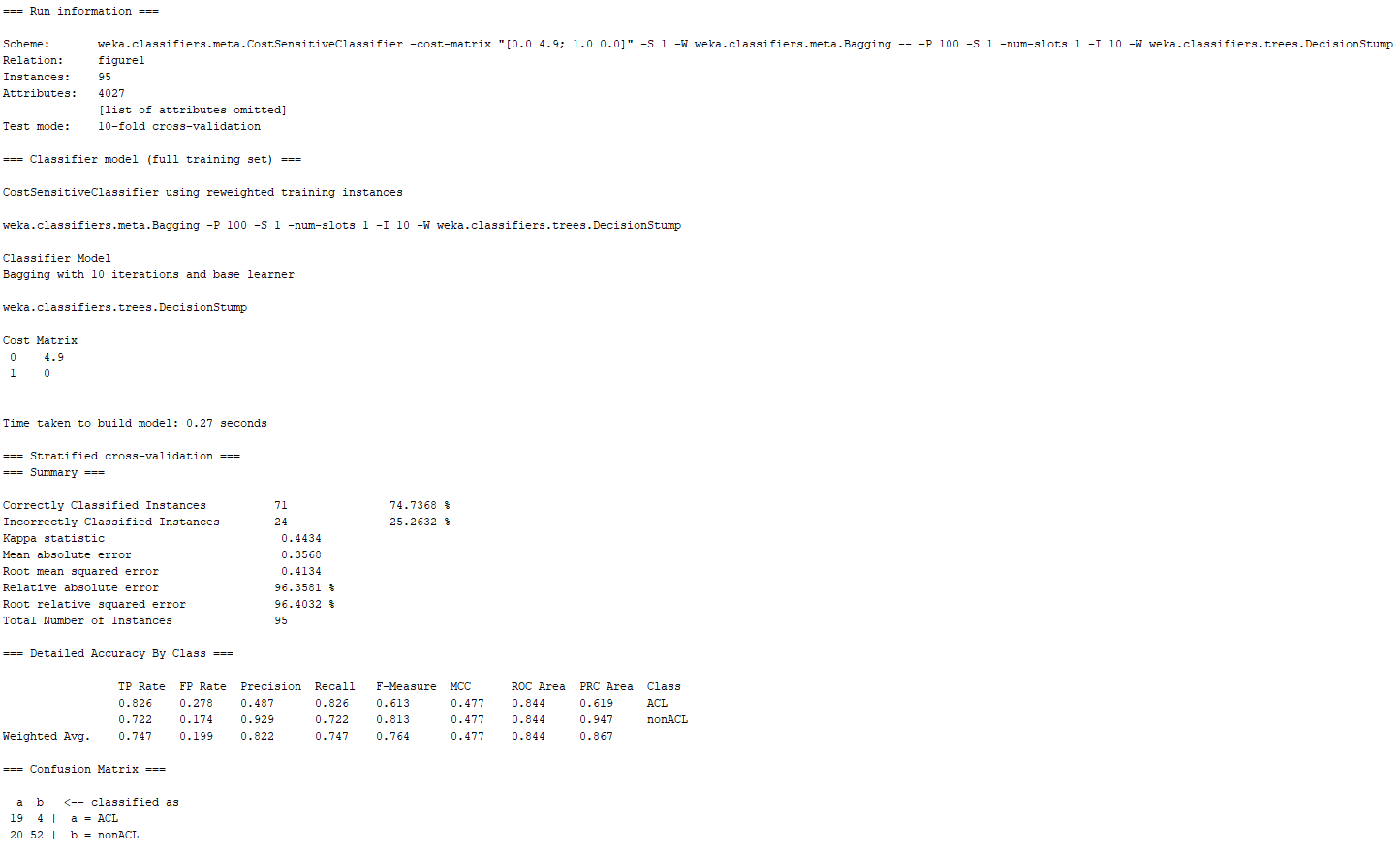
From these results, it can be noted that when the cost sensitive matrix value was 2.8, the model had the best performance with the lowest misclassification of the minority class and lowest type II error ratio. Moreover, when compared to the cost sensitive tree obtained in Assignment 1 part 4, it can be noted that the misclassification of the minority class was the same. Nevertheless, the misclassification of the majority class in the model in picture 7 had a higher misclassification rate.

Cost Sensitive Classifier Combined with Bagging and Decision Stump

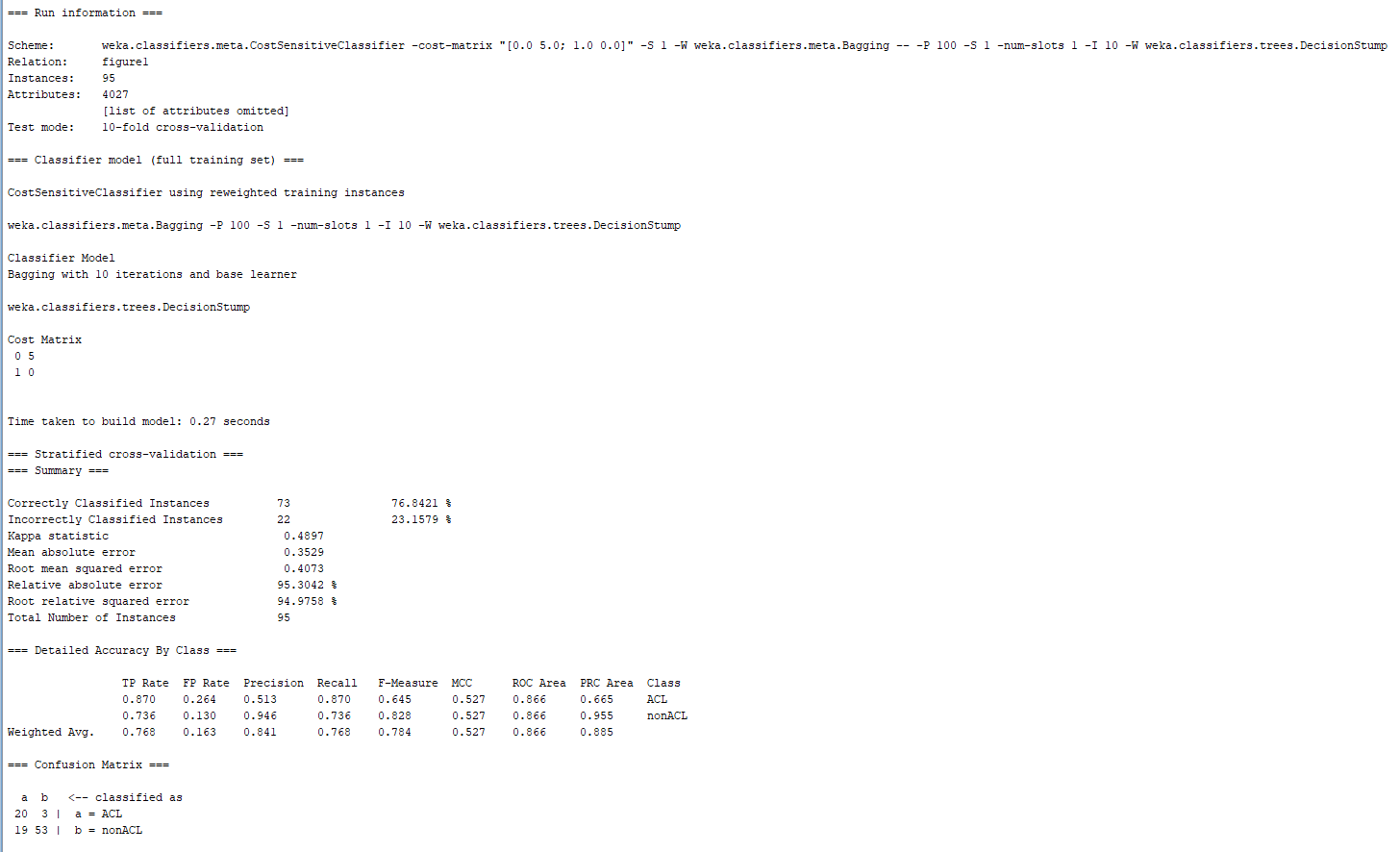
The Cost Sensitive Classifier Combined with Bagging and Decision Stump ML model was trained using the following values for the cost sensitive matrix: 0.5, 0.6, 1, 2, 3, 4, 4.9, 5, 5.1, and 6. The parameters for the classifier were the default settings for the classifier while only changing the cost sensitive matrix value to procure results. The most relevant matrix values were 4, 4.9, 5, and 5.1. A screenshot these classifiers with their respective cost matrix can be seen below:



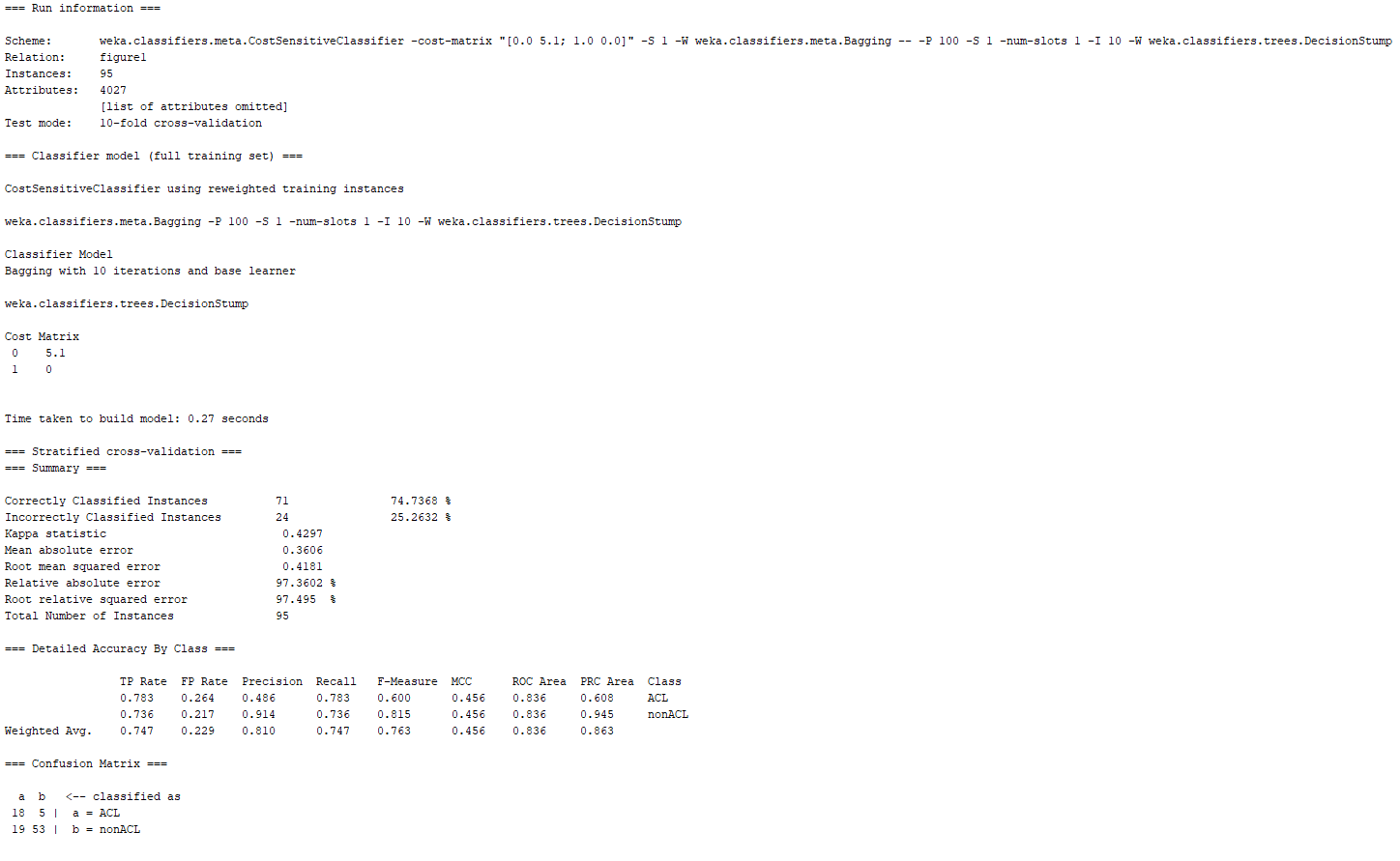
Picture 9: Cost Sensitive Classifier Combined with Bagging and Decision Stump – Cost Matrix 4 – 10-Fold Cross Validation



Picture 10: Cost Sensitive Classifier Combined with Bagging and Decision Stump – Cost Matrix 4.9 – 10-Fold Cross Validation

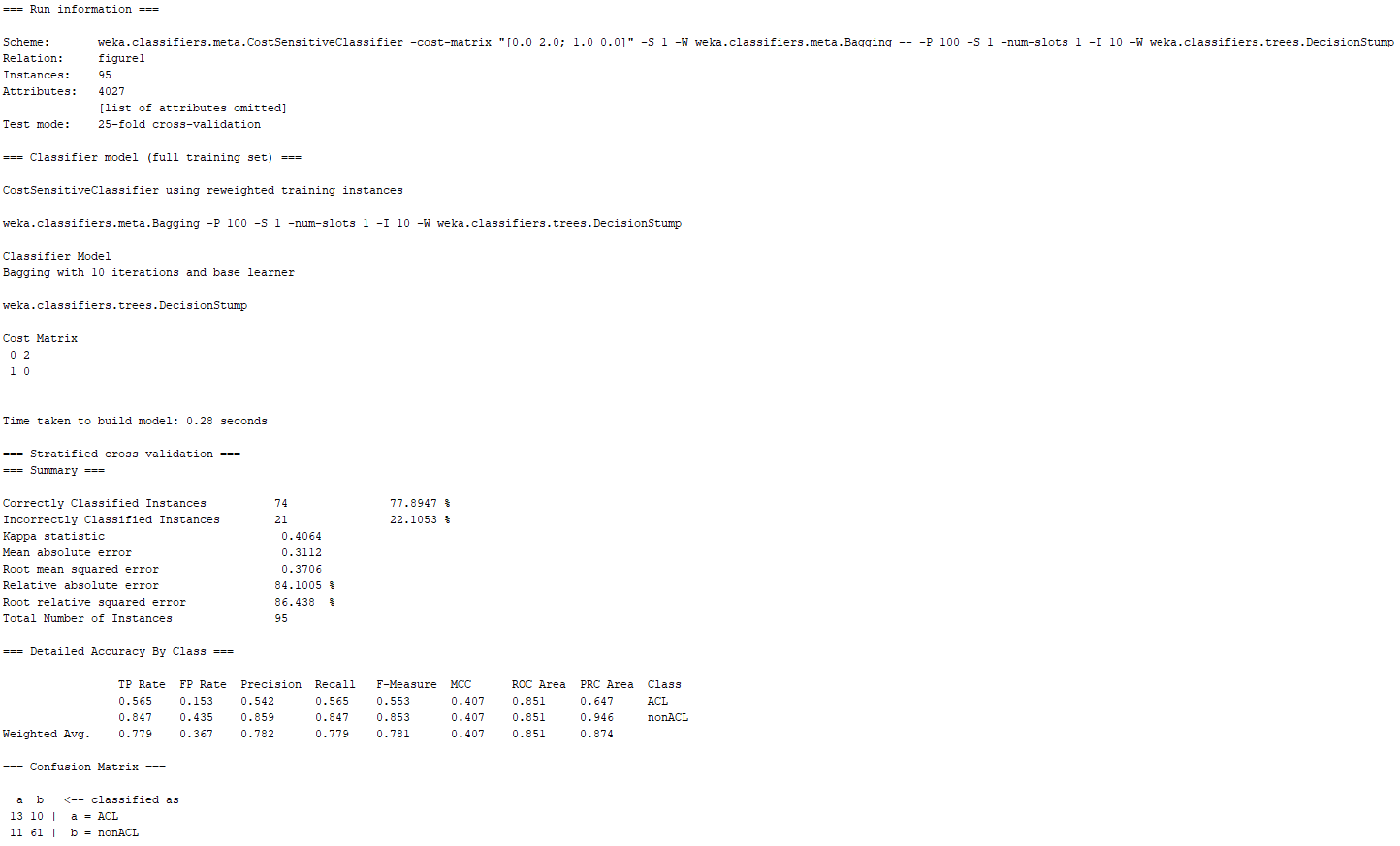


Picture 11: Cost Sensitive Classifier Combined with Bagging and Decision Stump – Cost Matrix 5 – 10-Fold Cross Validation

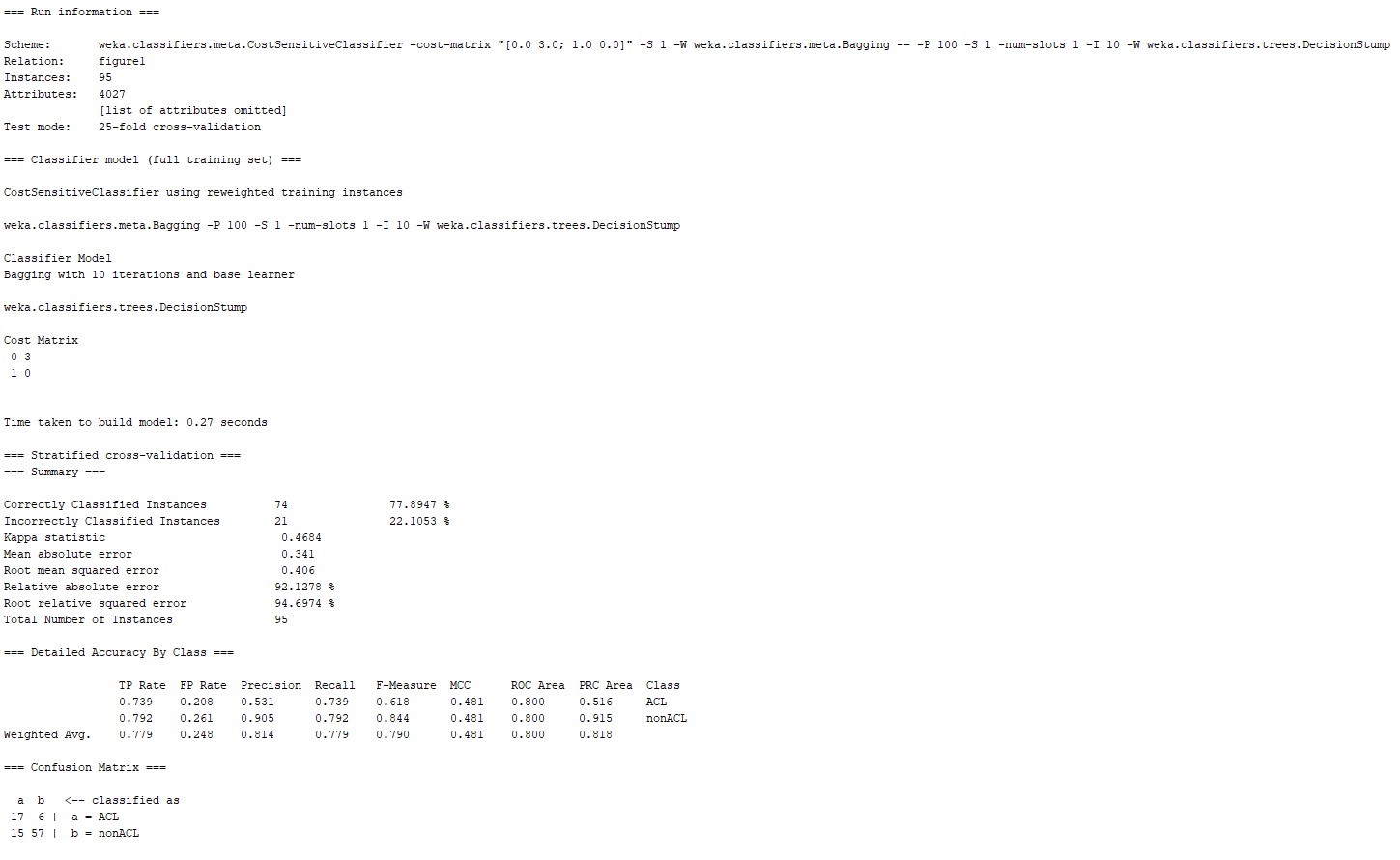


Picture 12: Cost Sensitive Classifier Combined with Bagging and Decision Stump – Cost Matrix 5.1 – 10-Fold Cross Validation

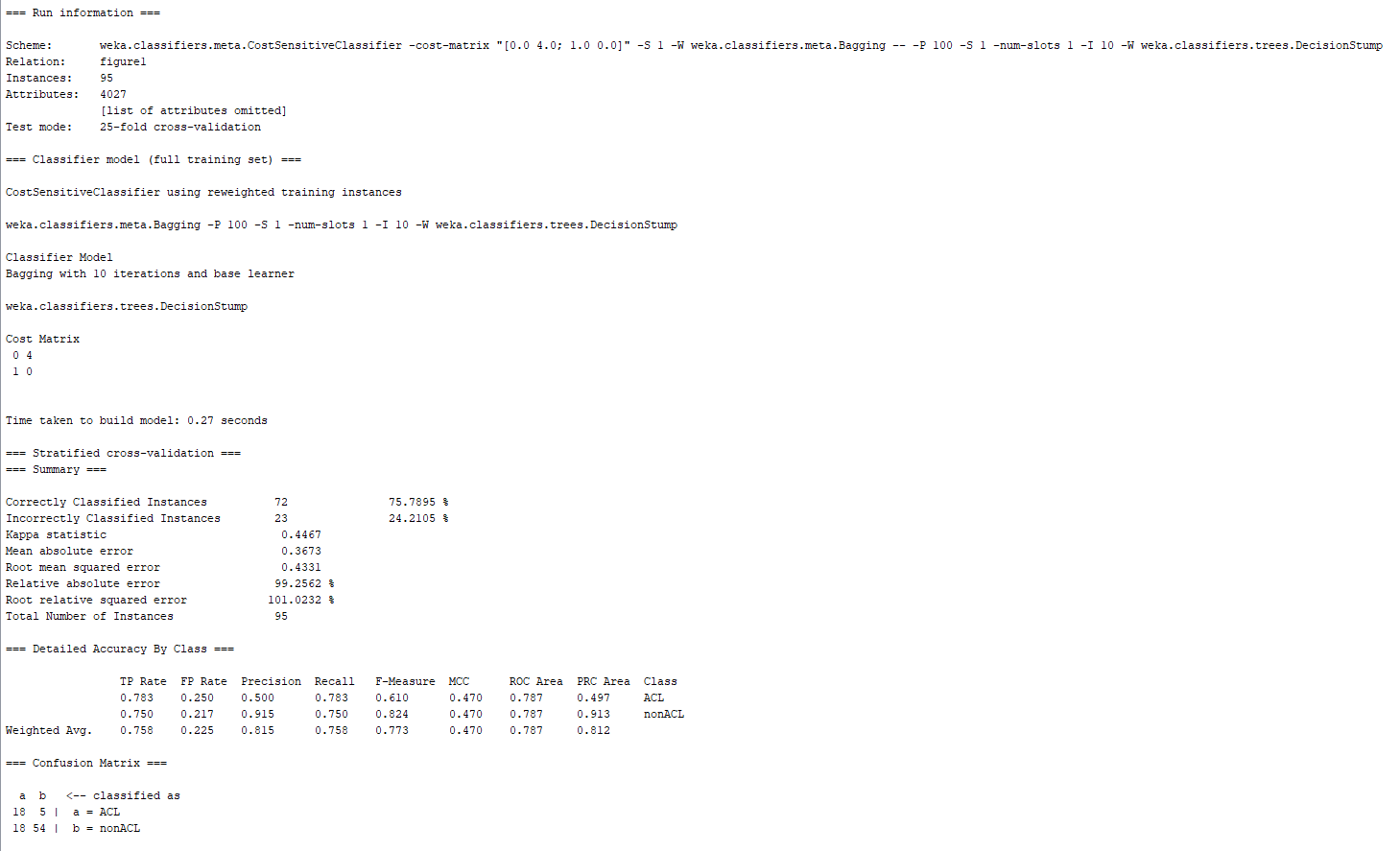
From these results, it can be noted that when the cost sensitive matrix value was 5, the model had the best performance with the lowest misclassification of the minority class and lowest type II error ratio. Moreover, when compared to the cost sensitive tree obtained in Assignment 1 part 4, it can be noted that the misclassification of the minority class is lower from the model in picture 11. Nevertheless, the misclassification of the majority class in the model in picture 11 had a higher misclassification rate. Therefore, this model might be the best model for classifying the minority class, keeping in mind that the classification for the majority class will be heavily affected in regards to classification performance. However, since the minority class is the class of interest, the tradeoff between how much misclassification from the majority class can be allowed would need to be analyzed. With this information in mind, the next step consists in training the model with 25-fold cross validation. The cost sensitive matrix: 0.5, 0.6, 1, 2, 3, 4, 5, and 6. The most relevant matrix values were 2, 3, 4, and 5. A screenshot these classifiers with their respective cost matrix can be seen below:



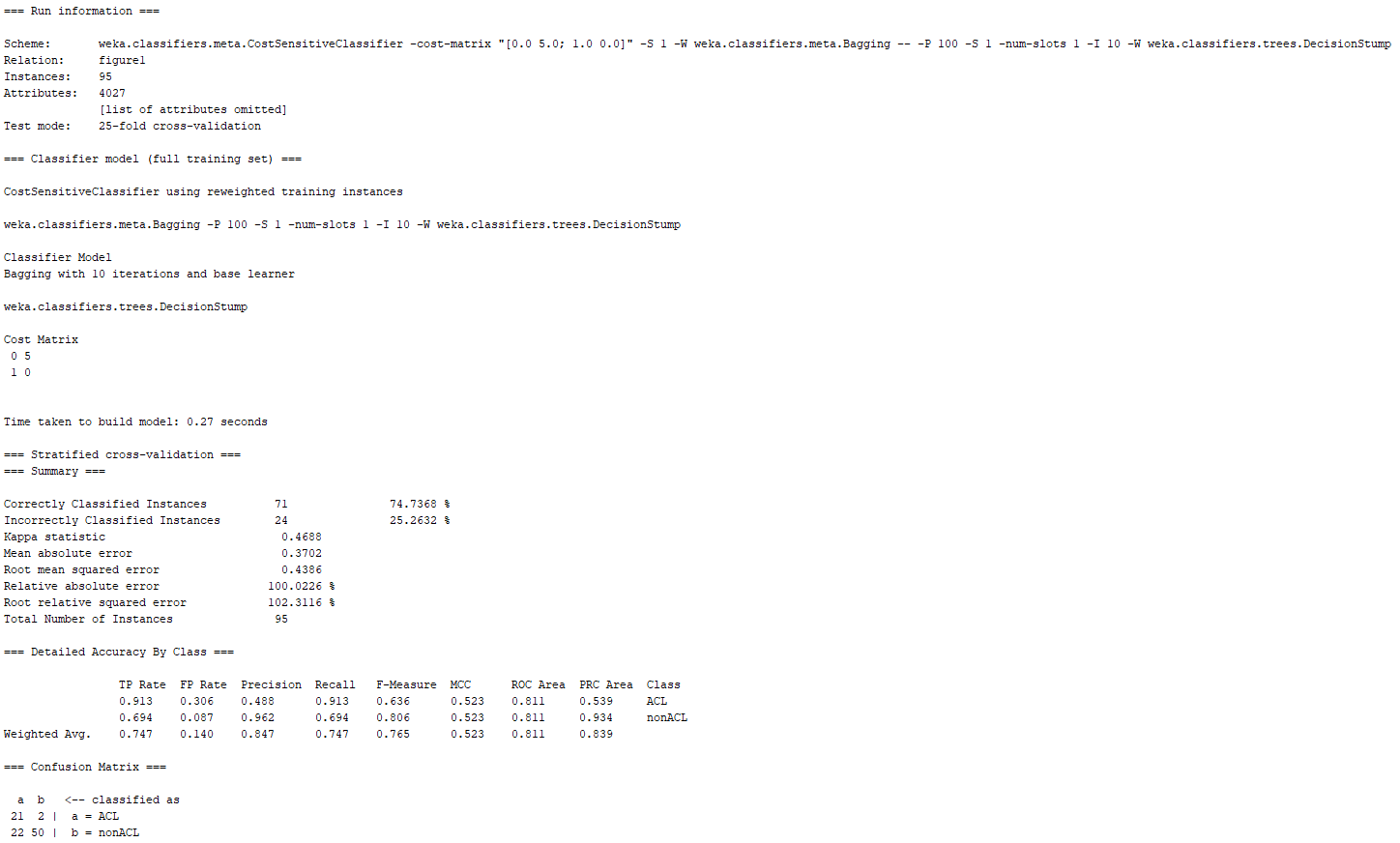
Picture 13: Cost Sensitive Classifier Combined with Bagging and Decision Stump – Cost Matrix 2 – 25-Fold Cross Validation



Picture 14: Cost Sensitive Classifier Combined with Bagging and Decision Stump – Cost Matrix 3 – 25-Fold Cross Validation



Picture 15: Cost Sensitive Classifier Combined with Bagging and Decision Stump – Cost Matrix 4 – 25-Fold Cross Validation

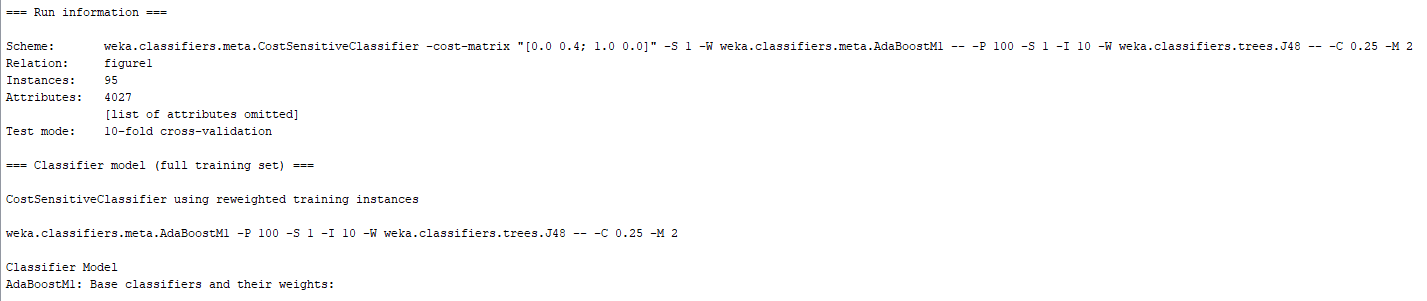


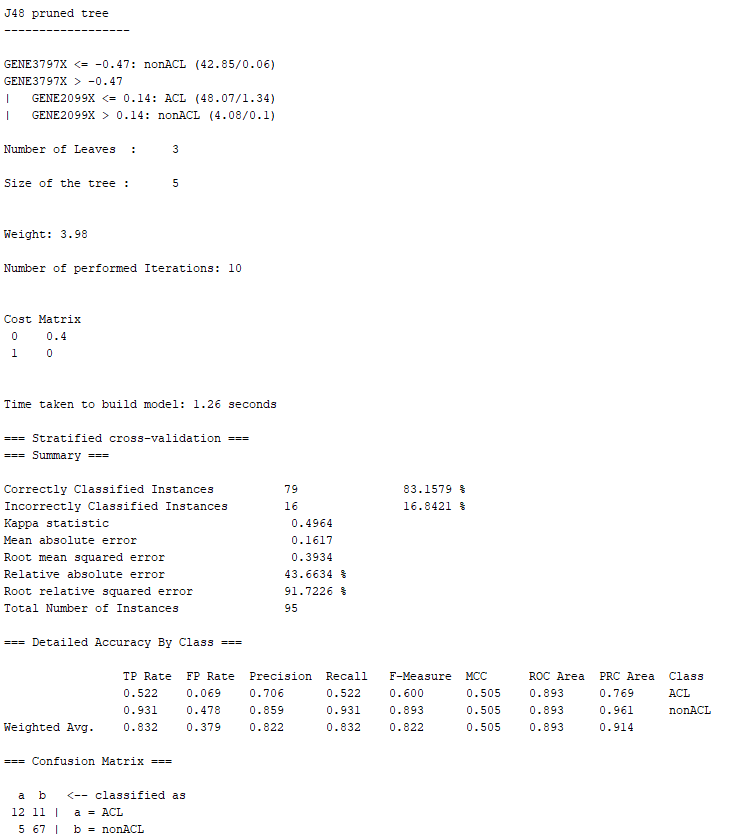
Picture 16: Cost Sensitive Classifier Combined with Bagging and Decision Stump – Cost Matrix 5 – 25-Fold Cross Validation

From these results, it can be noted that when the cost sensitive matrix value was 5, the model had the best performance with the lowest misclassification of the minority class and lowest type II error ratio. Moreover, when compared to the cost sensitive tree obtained in Assignment 1 part 4, it can be noted that the misclassification of the minority class is lower from the model in picture 15. Nevertheless, the misclassification of the majority class in the model in picture 11 had a higher misclassification rate. Therefore, this model might be the best model for classifying the minority class, keeping in mind that the classification for the majority class will be heavily affected in regards to classification performance. However, since the minority class is the class of interest, the tradeoff between how much misclassification from the majority class can be allowed would need to be analyzed. From a different perspective, it is important to note that cost ratio of 4 delivers a more balance misclassification from the two classes.

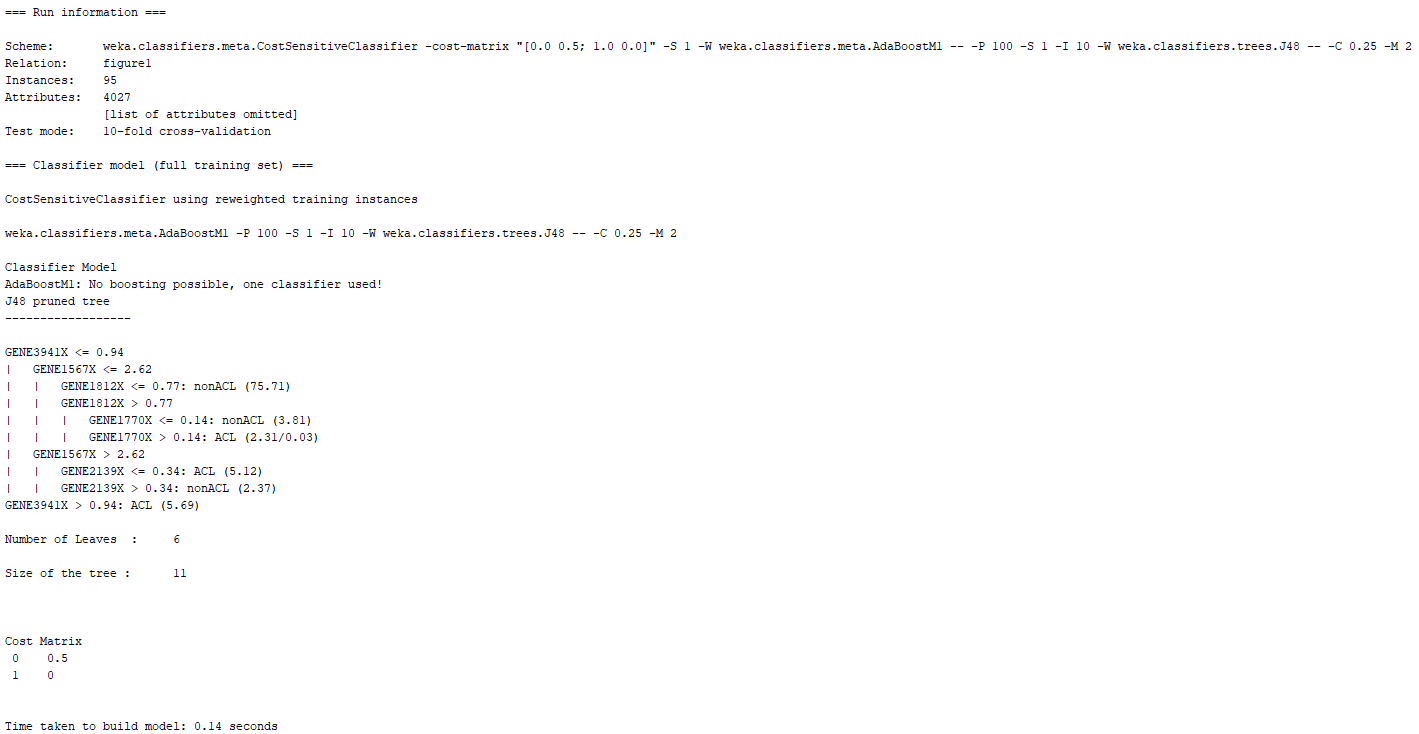
Cost Sensitive Classifier Combined with Boosting and J48

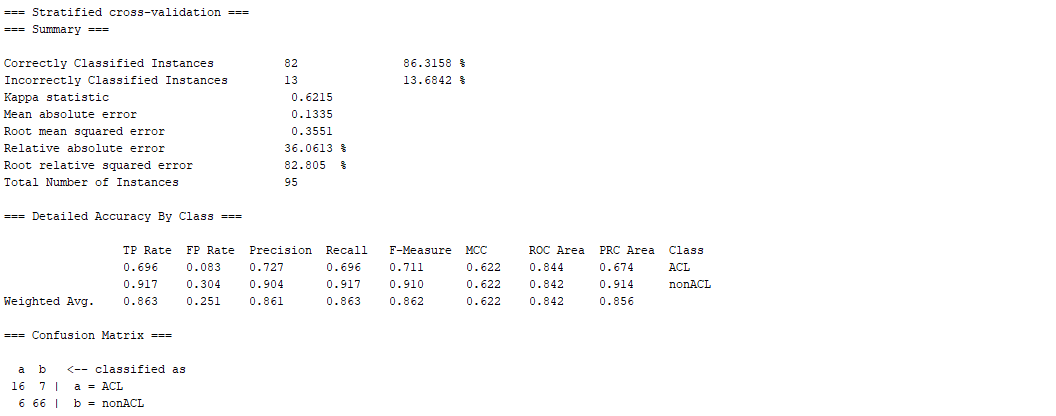
The Cost Sensitive Classifier Combined with Boosting and J48 ML model was trained using the following values for the cost sensitive matrix: 0.4, 0.5, 0.6, 0.9, 1, 1.1, 2, 3, and 4. The parameters for the classifier were the default settings for the classifier while only changing the cost sensitive matrix value to procure results. The most relevant matrix values were 0.5, 0.6, 0.9, and 1. A screenshot these classifiers with their respective cost matrix can be seen below:



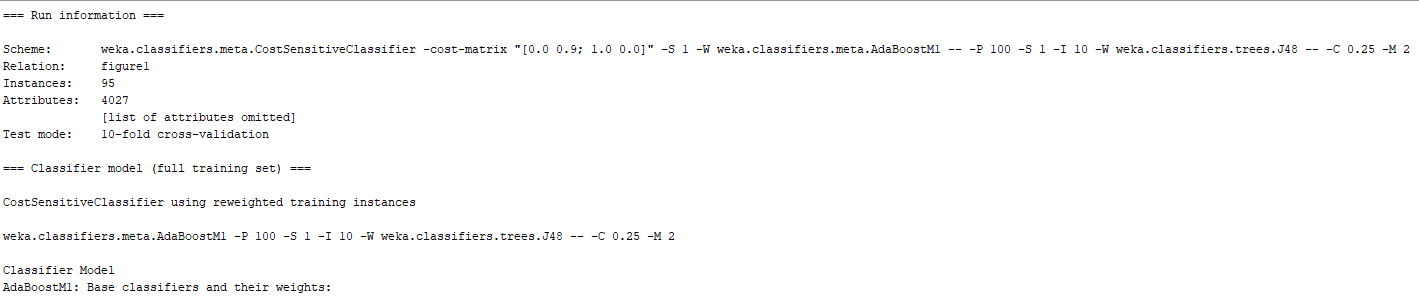


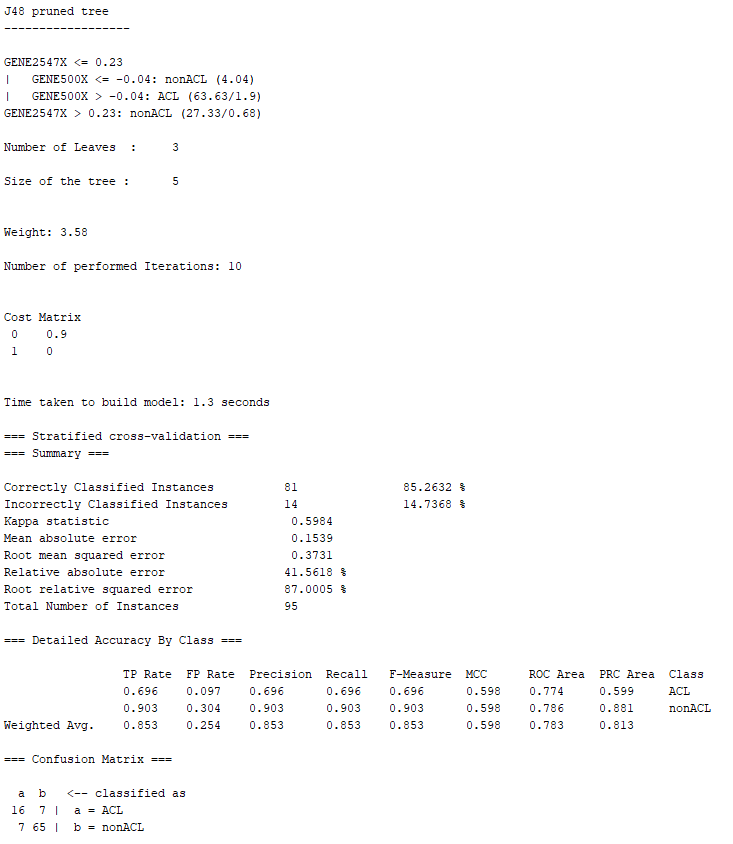
Picture 17: Cost Sensitive Classifier Combined with Boosting and J48 – Cost Matrix 0.4 – 10-Fold Cross Validation



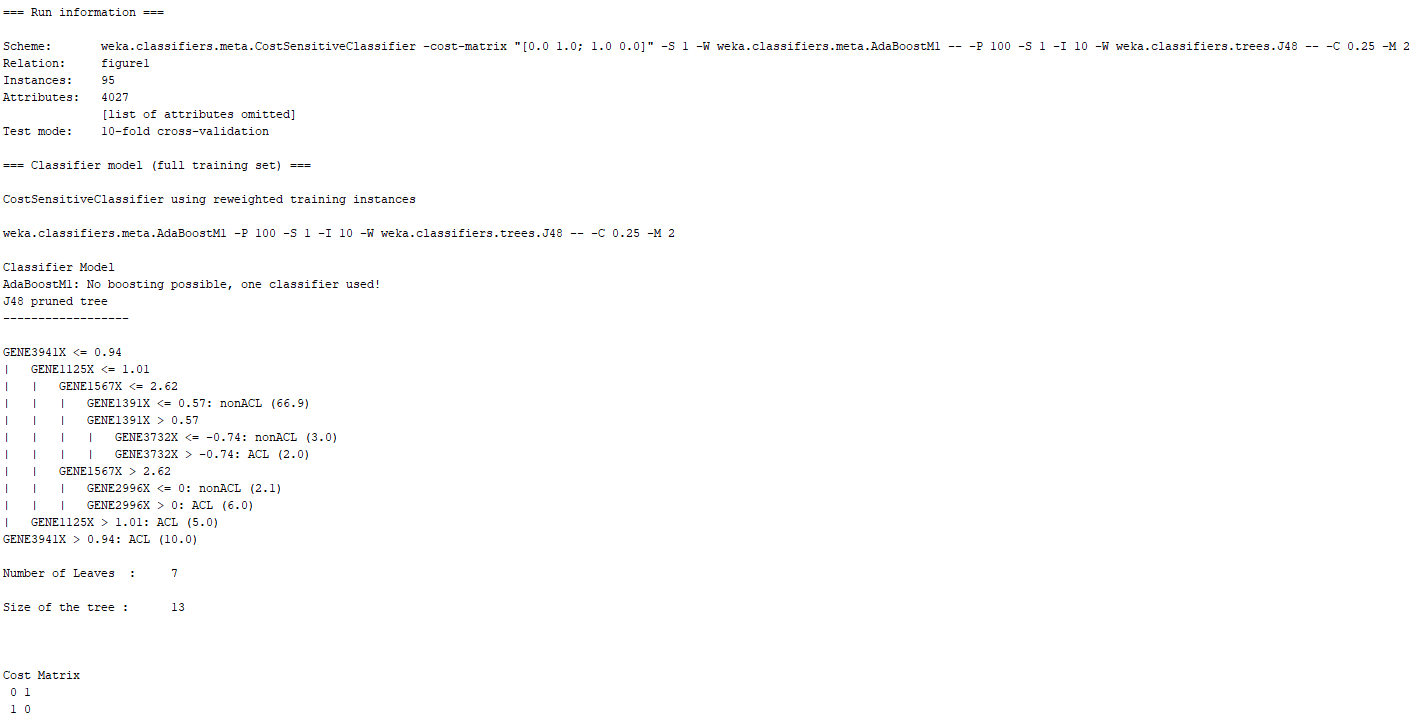


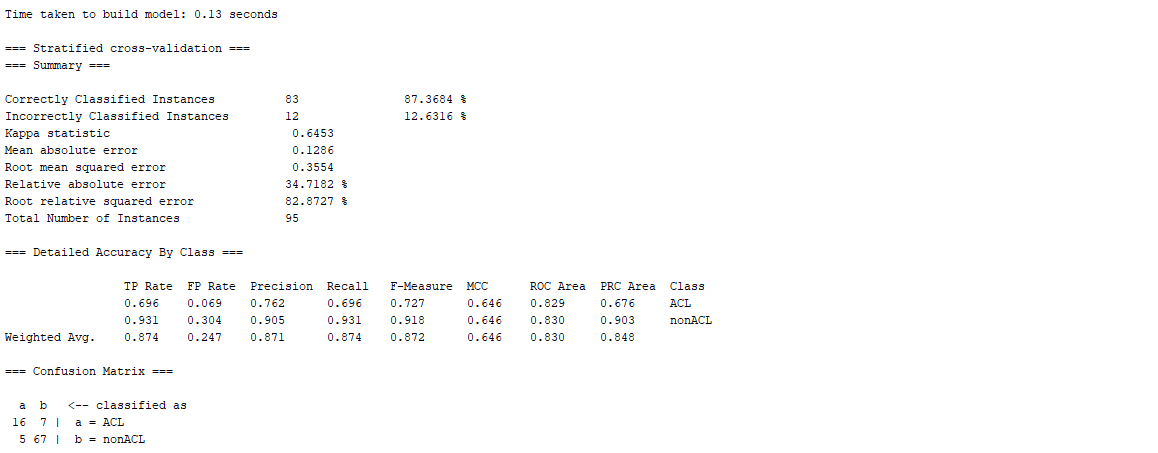
Picture 18: Cost Sensitive Classifier Combined with Boosting and J48 – Cost Matrix 0.5 – 10-Fold Cross Validation





Picture 19: Cost Sensitive Classifier Combined with Boosting and J48 – Cost Matrix 0.9 – 10-Fold Cross Validation

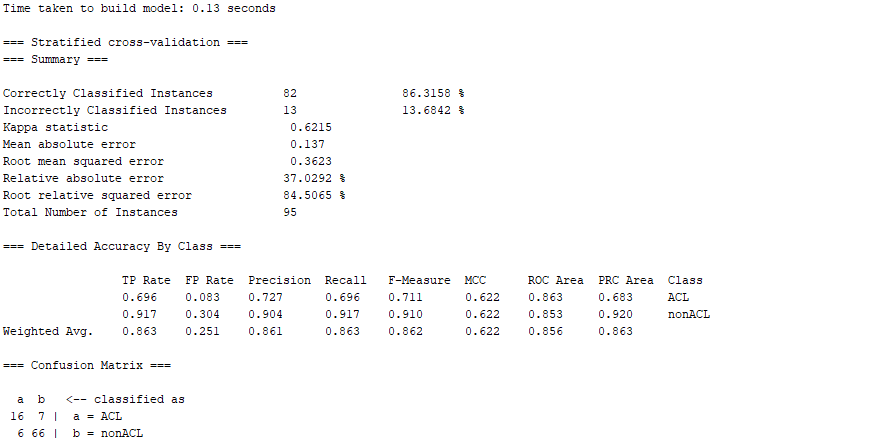




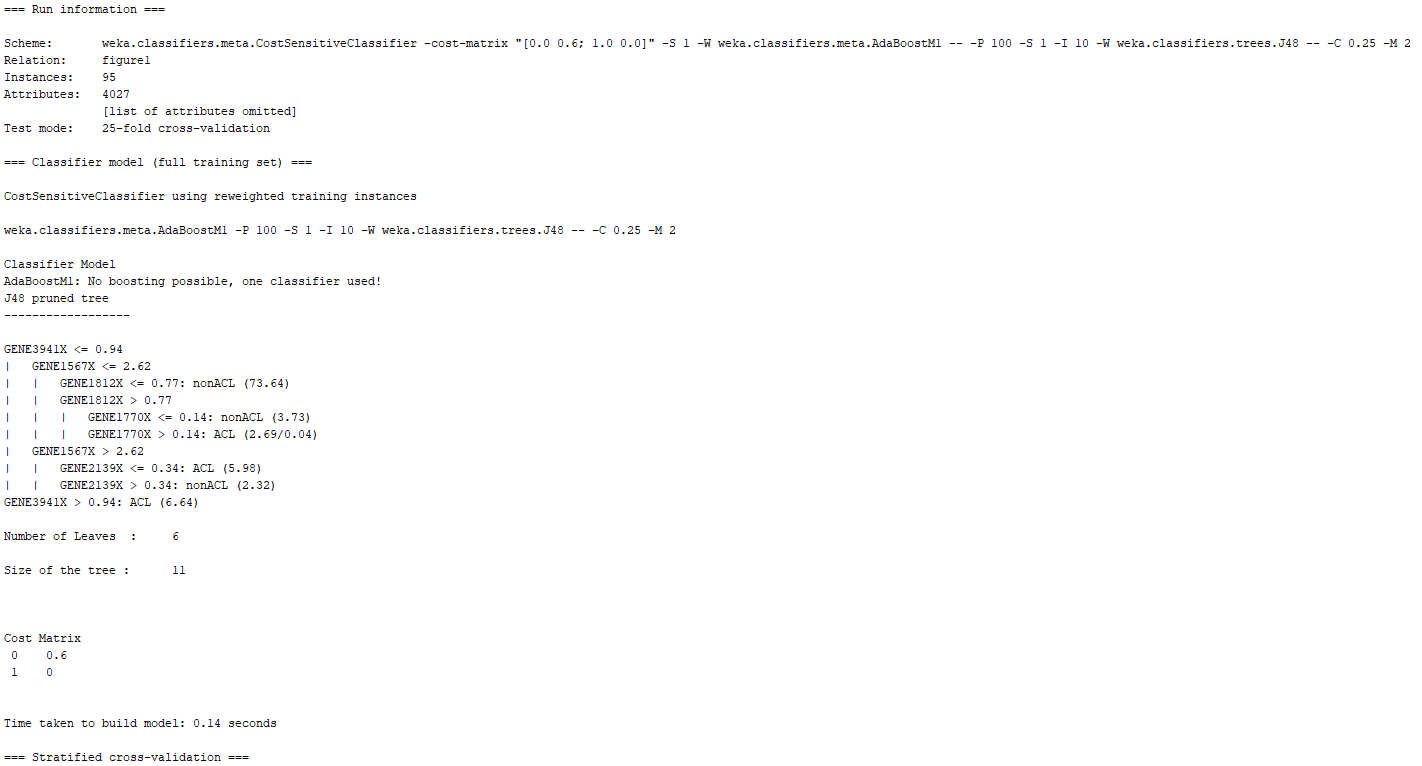
Picture 20: Cost Sensitive Classifier Combined with Boosting and J48 – Cost Matrix 1 – 10-Fold Cross Validation

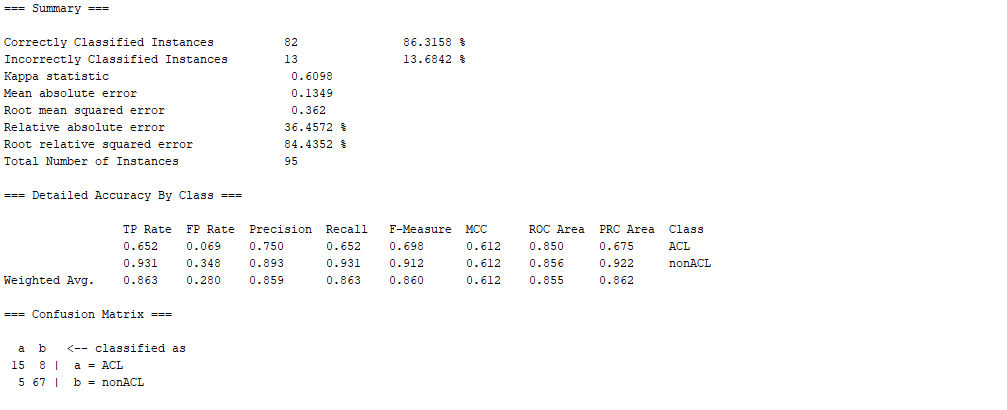
From these results, it can be noted that when the cost sensitive matrix values were 0.5 and 1, the model had the best performance with the lowest misclassification of the minority class. Nevertheless, none of the models trained were able to match the same efficiency as the cost sensitive tree obtained in Assignment 1 part 4. With this information in mind, the next step consists in training the model with 25-fold cross validation. The cost sensitive matrix: 0.4, 0.5, 0.6, 0.9, 1, 1.1, 2, 3, and 3. The most relevant matrix values were 0.5, 0.6, 1, and 1.1. A screenshot these classifiers with their respective cost matrix can be seen below:



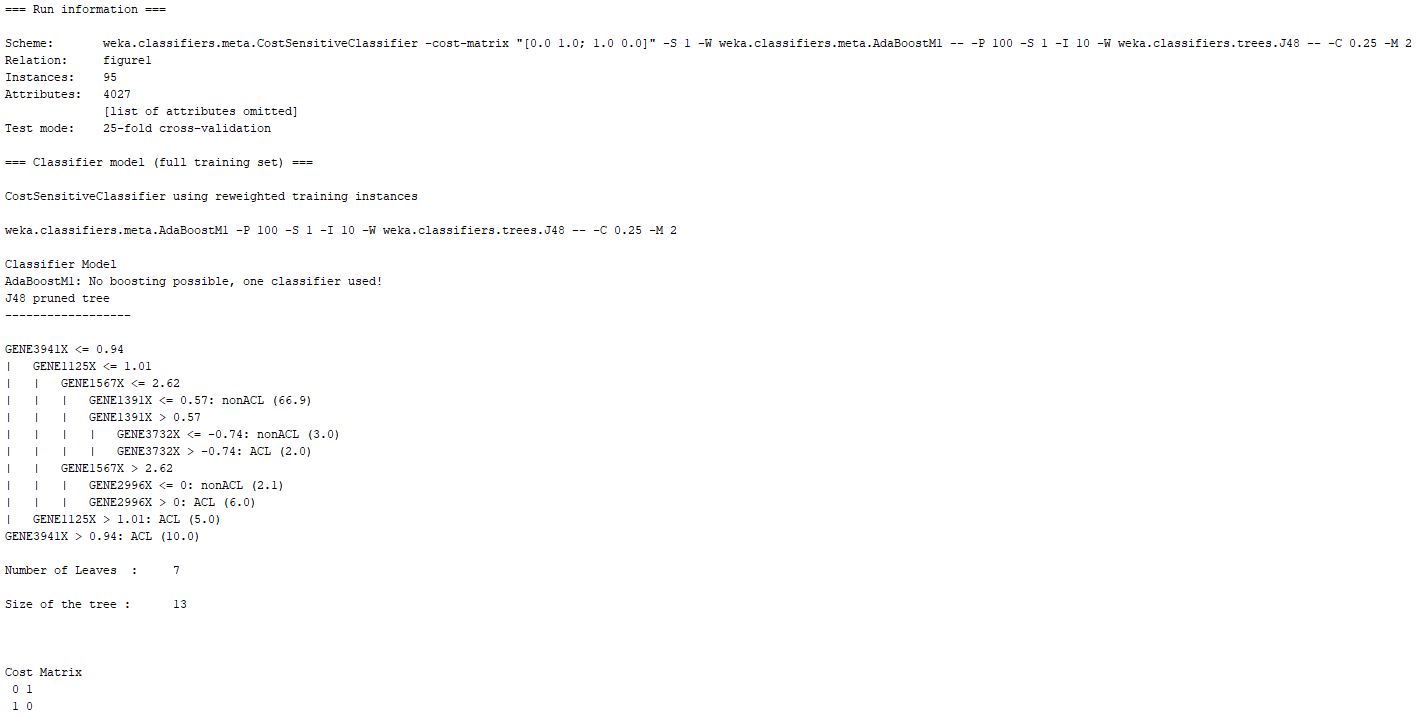


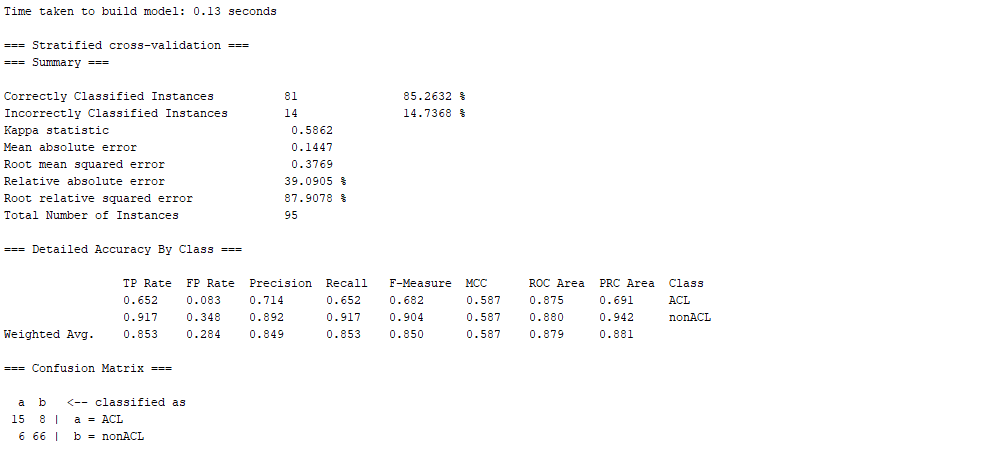
Picture 21: Cost Sensitive Classifier Combined with Boosting and J48 – Cost Matrix 0.5 – 25-Fold Cross Validation



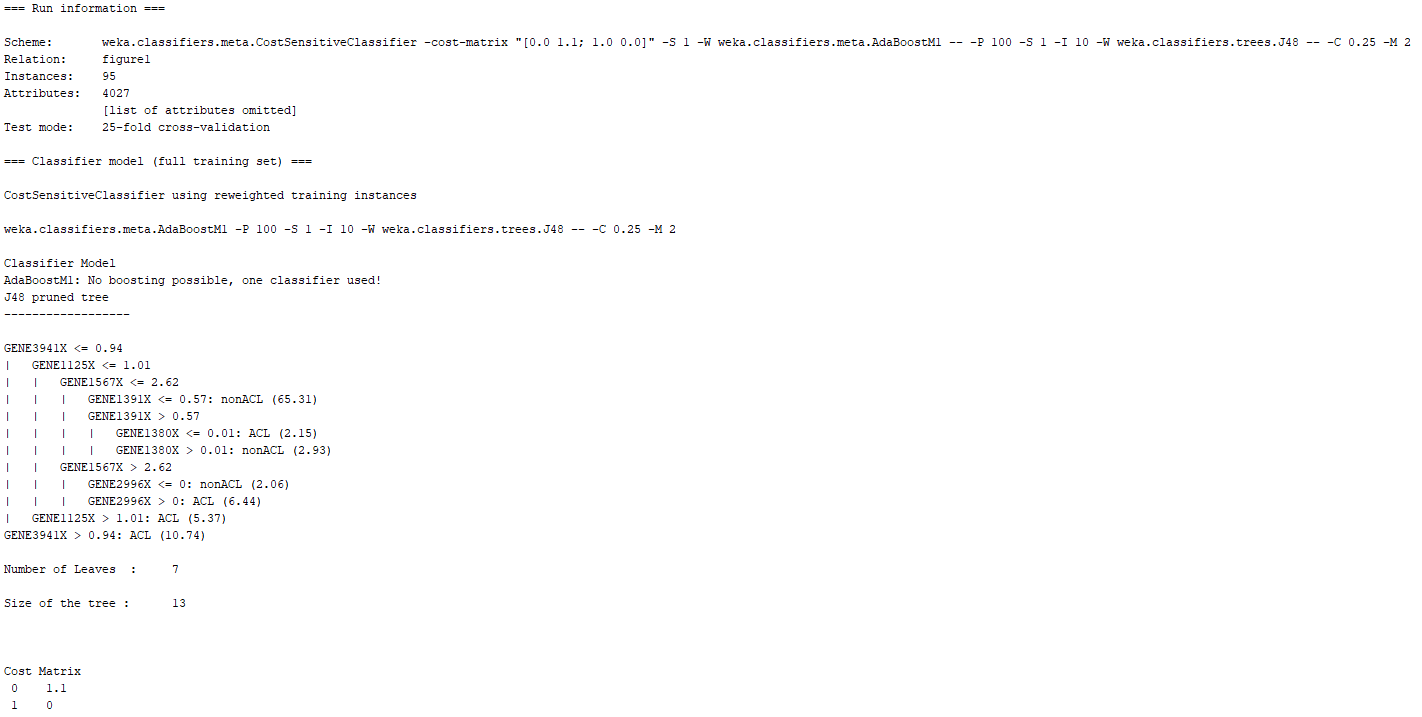


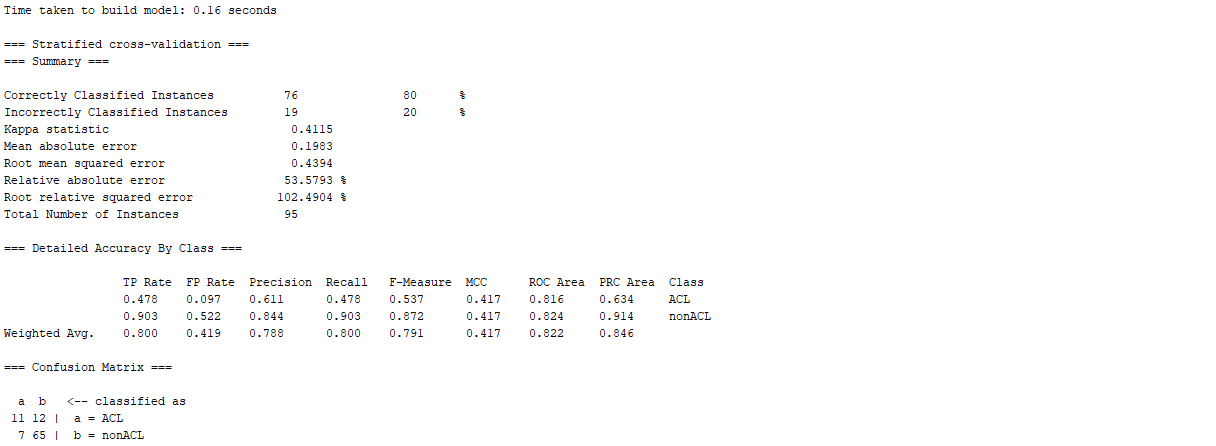
Picture 22: Cost Sensitive Classifier Combined with Boosting and J48 – Cost Matrix 0.6 – 25-Fold Cross Validation





Picture 23: Cost Sensitive Classifier Combined with Boosting and J48 – Cost Matrix 1 – 25-Fold Cross Validation



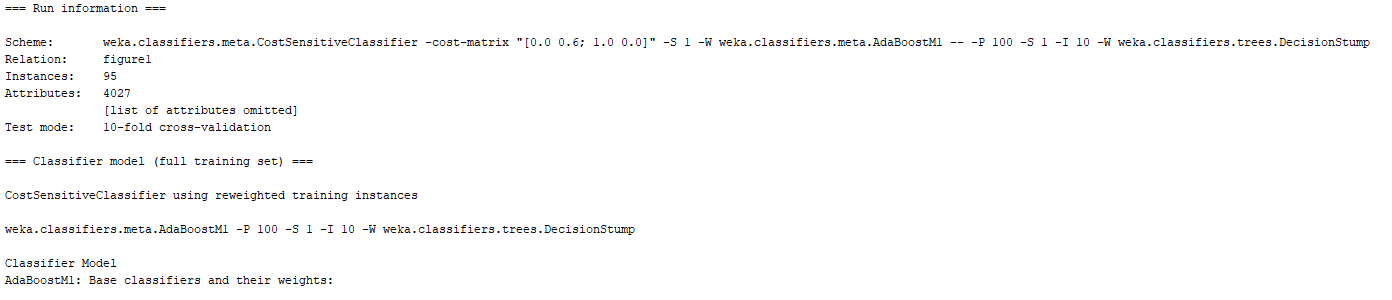


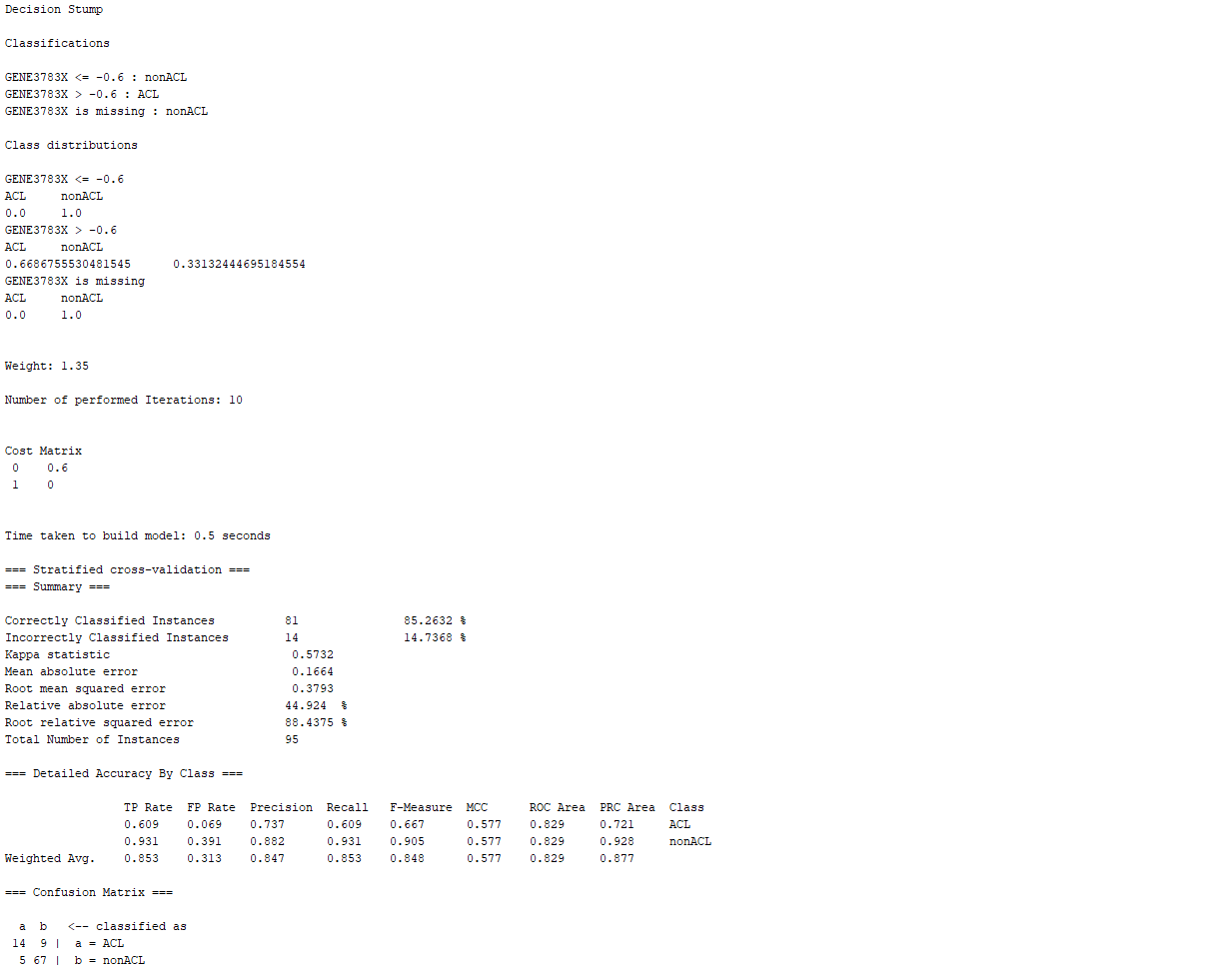
Picture 24: Cost Sensitive Classifier Combined with Boosting and J48 – Cost Matrix 1.1 – 25-Fold Cross Validation

From these results, it can be noted that when the cost sensitive matrix value was 0.5, the model had the best performance with the lowest misclassification of the minority class and lowest type II error ratio. Moreover, when compared to the cost sensitive tree obtained in Assignment 1 part 4, it can be noted that the misclassification of the minority class was slightly higher.

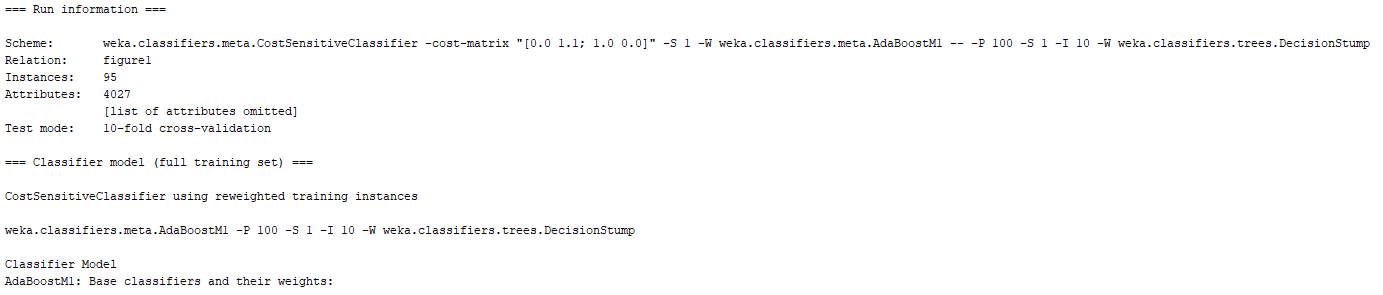
Cost Sensitive Classifier Combined with Boosting and Decision Stump

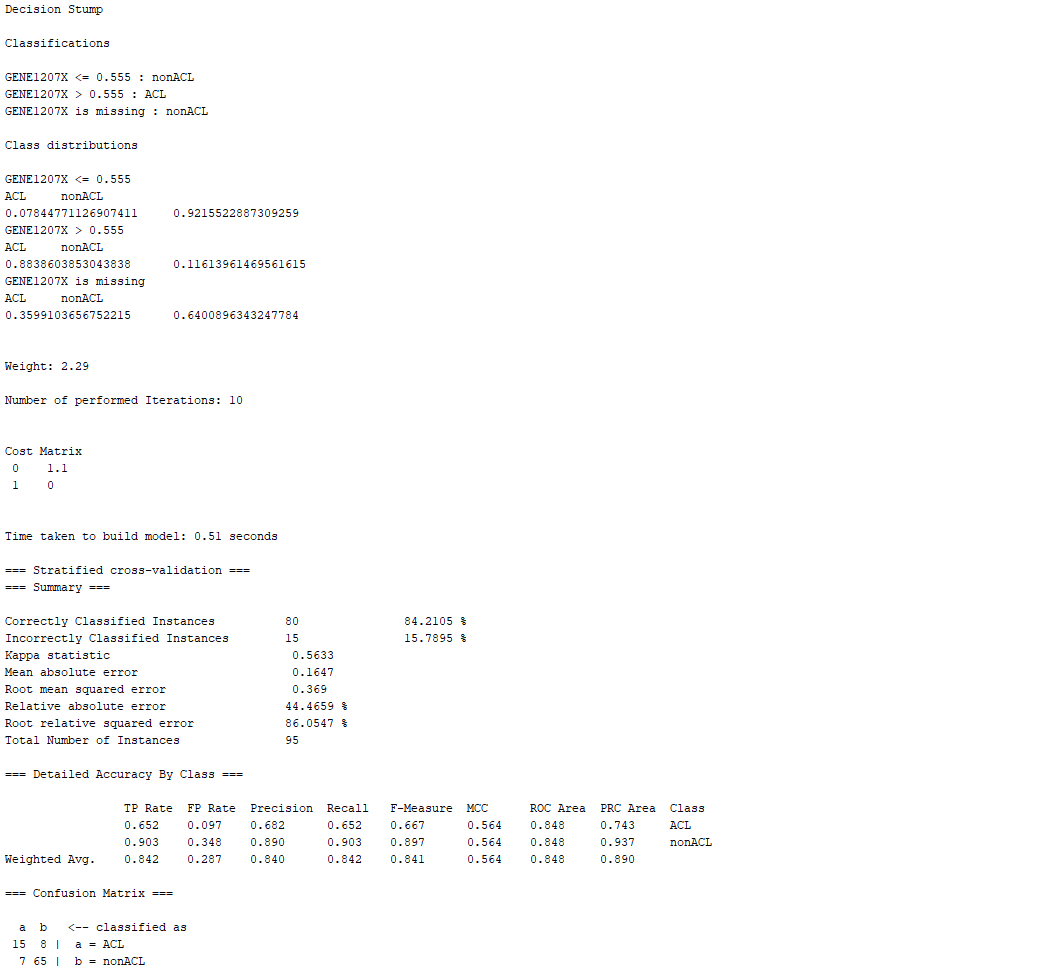
The Cost Sensitive Classifier Combined with Boosting and Decision Stump ML model was trained using the following values for the cost sensitive matrix: 0.5, 0.6, 0.7, 0.9, 1, 1.1, 1.2, 1.3, and 2. The parameters for the classifier were the default settings for the classifier while only changing the cost sensitive matrix value to procure results. The most relevant matrix values were 0.6, 1.1, 1.2, and 1.3. A screenshot these classifiers with their respective cost matrix can be seen below:



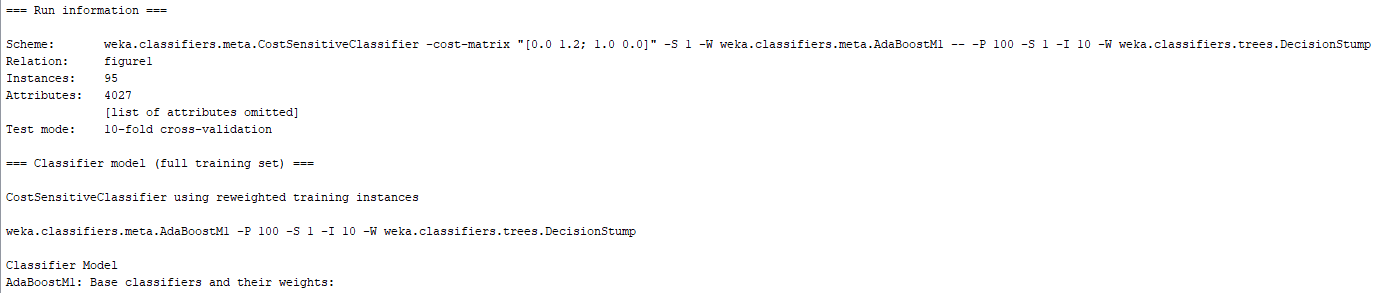


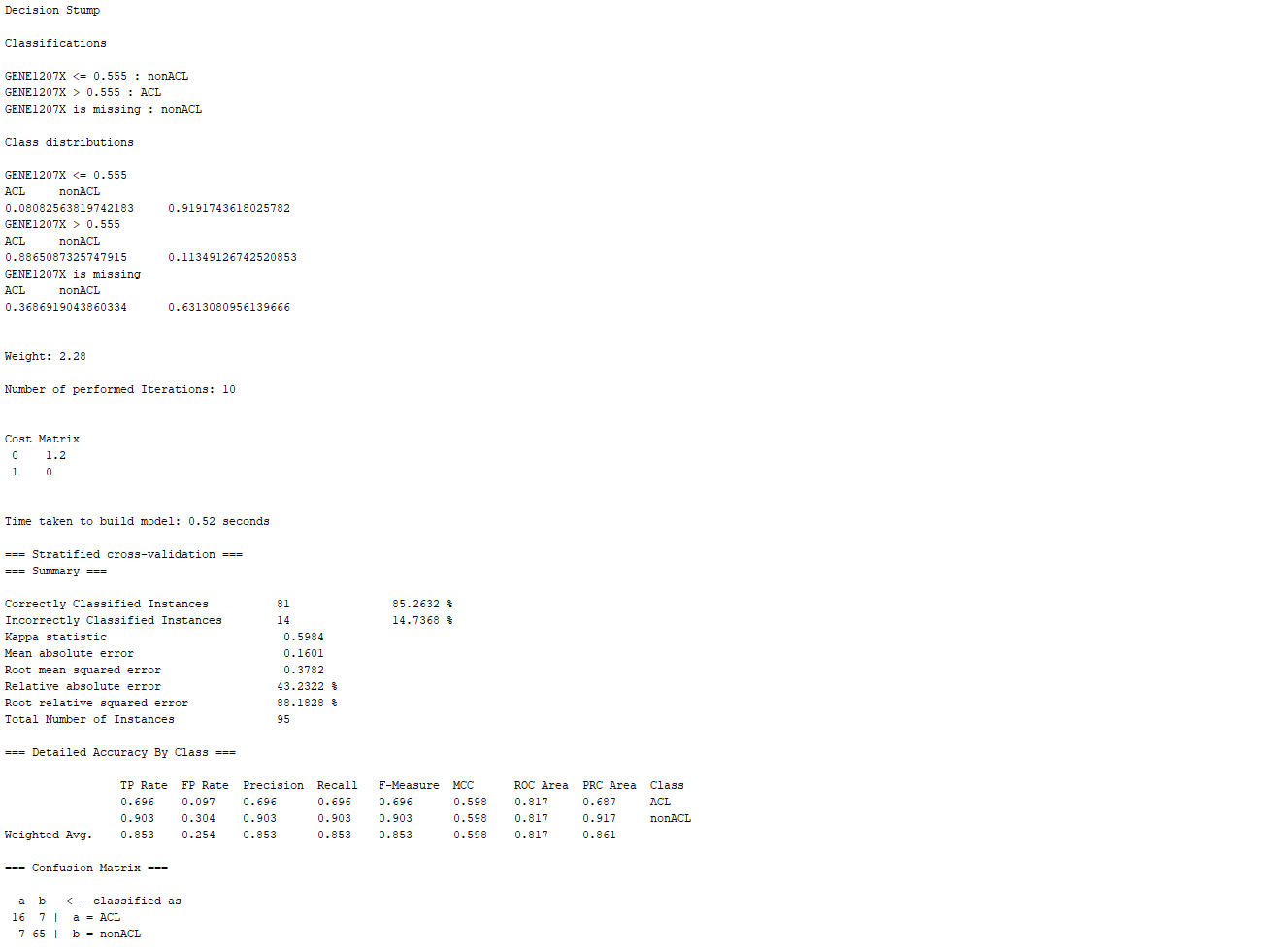
Picture 25: Cost Sensitive Classifier Combined with Boosting and Decision Stump – Cost Matrix 0.6 – 10-Fold Cross Validation



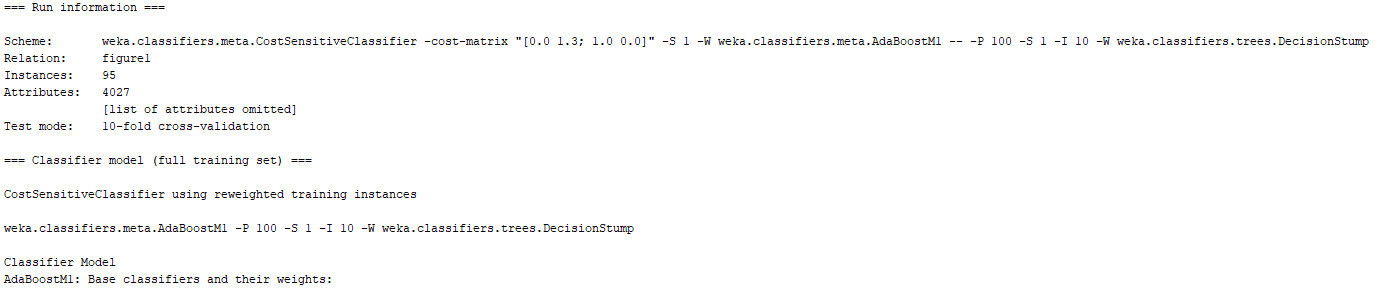


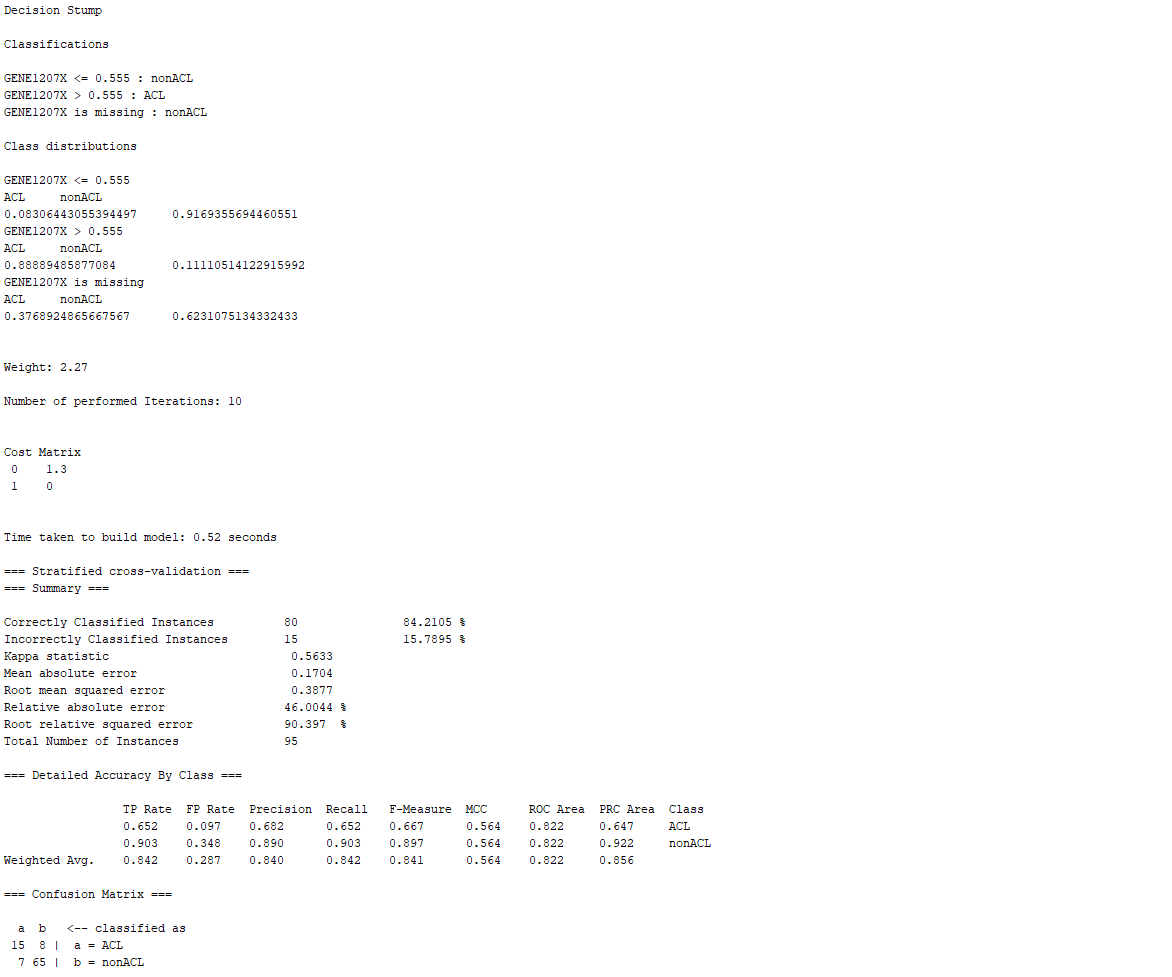
Picture 26: Cost Sensitive Classifier Combined with Boosting and Decision Stump – Cost Matrix 1.1 – 10-Fold Cross Validation





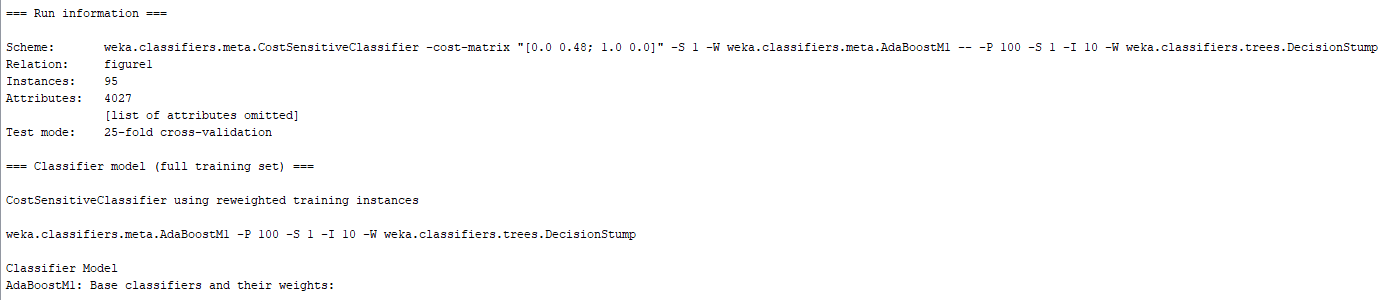
Picture 27: Cost Sensitive Classifier Combined with Boosting and Decision Stump – Cost Matrix 1.2 – 10-Fold Cross Validation

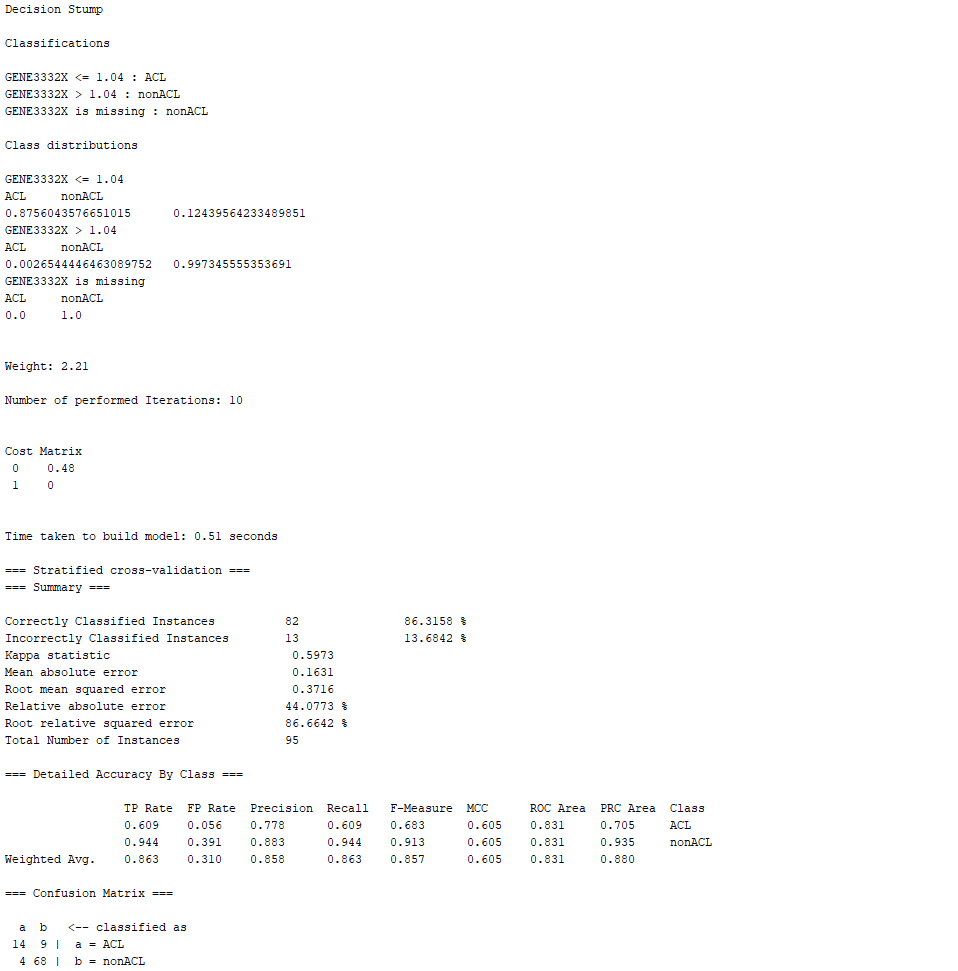




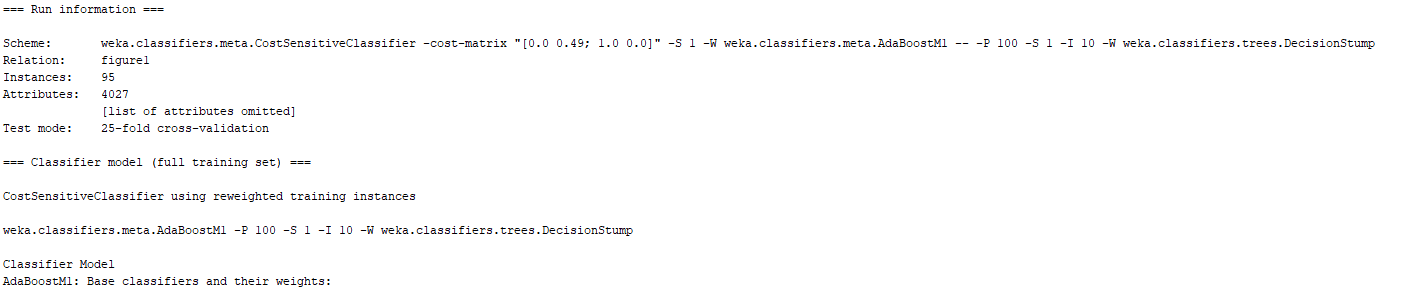
Picture 28: Cost Sensitive Classifier Combined with Boosting and Decision Stump – Cost Matrix 1.3 – 10-Fold Cross Validation

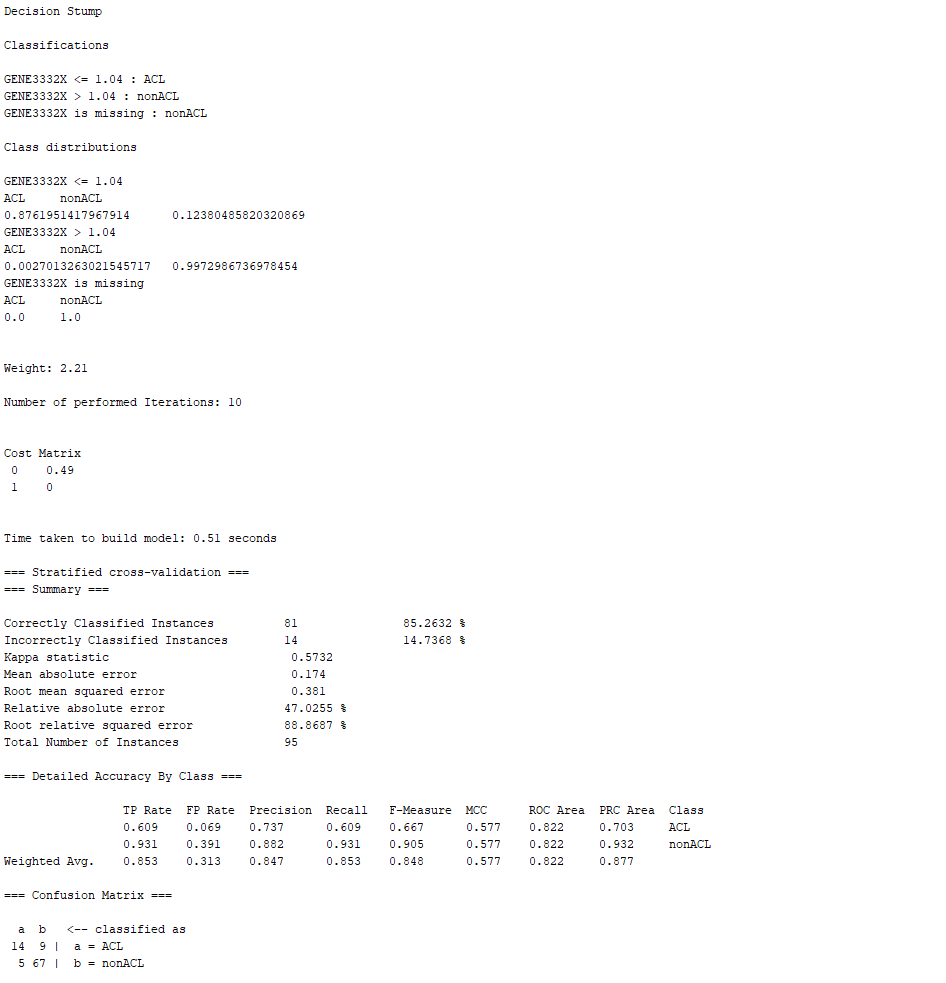
From these results, it can be noted that when the cost sensitive matrix value was 1.2, the model had the best performance with the lowest misclassification of the minority class and lowest type II error ratio. Moreover, when compared to the cost sensitive tree obtained in Assignment 1 part 4, it can be noted that the misclassification of the minority class was higher from the model in picture 27. With this information in mind, the next step consists in training the model with 25-fold cross validation. The cost sensitive matrix: 0.4, 0.47, 0.48, 0.49, 0.5, 0.51, 0.52, 0.6, 1, and 2. The most relevant matrix values were 0.48, 0.49, 0.5, and 0.51. A screenshot these classifiers with their respective cost matrix can be seen below:



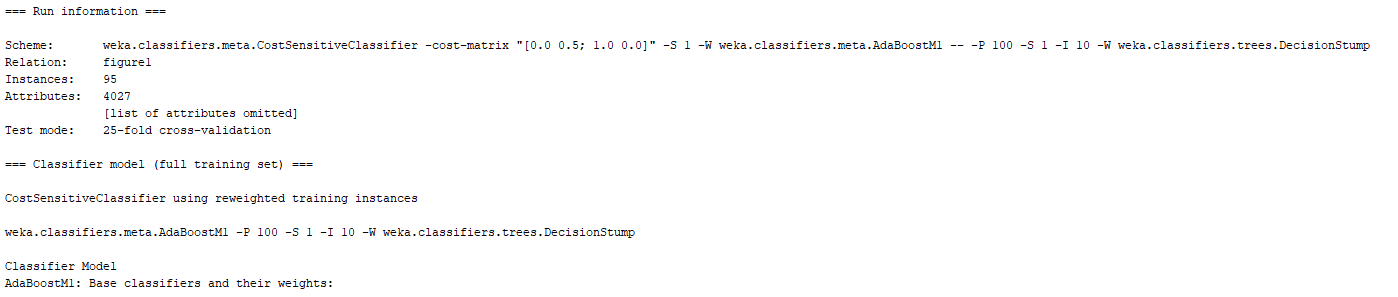


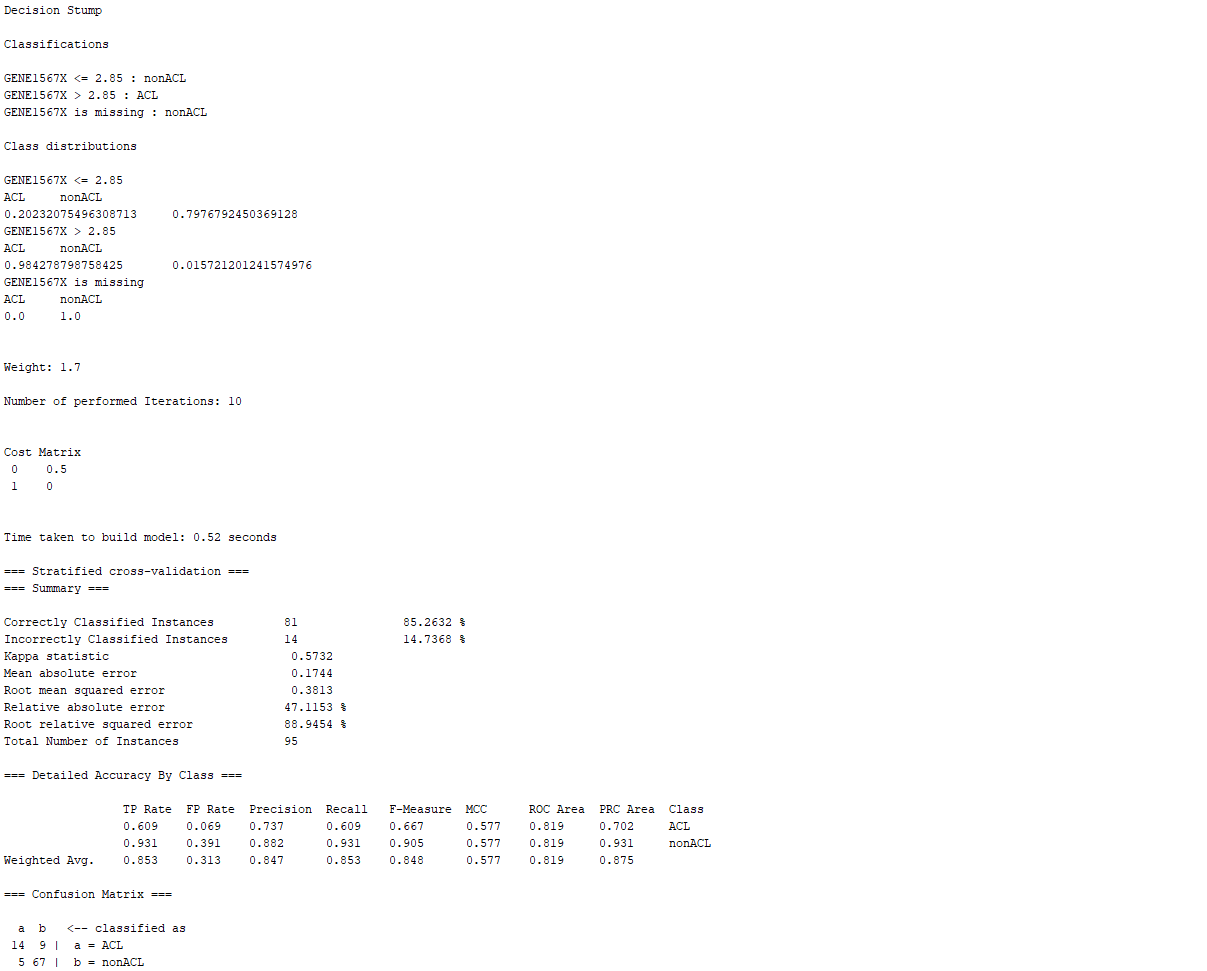
Picture 29: Cost Sensitive Classifier Combined with Boosting and Decision Stump – Cost Matrix 0.48 – 25-Fold Cross Validation



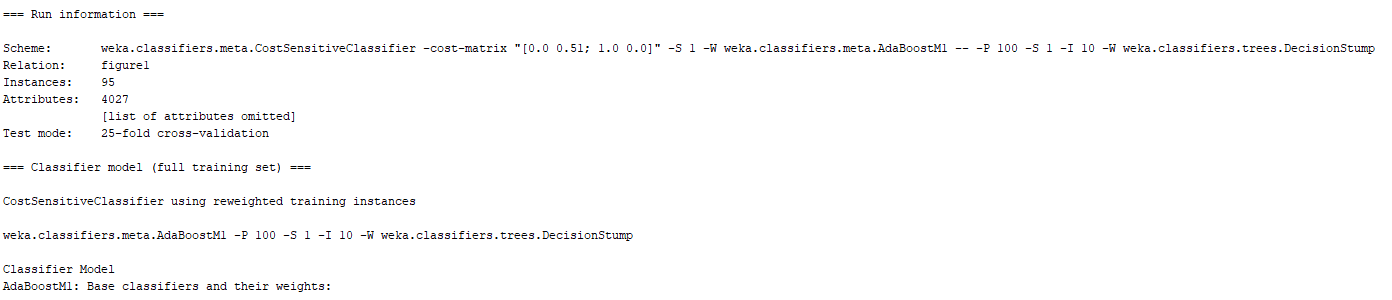


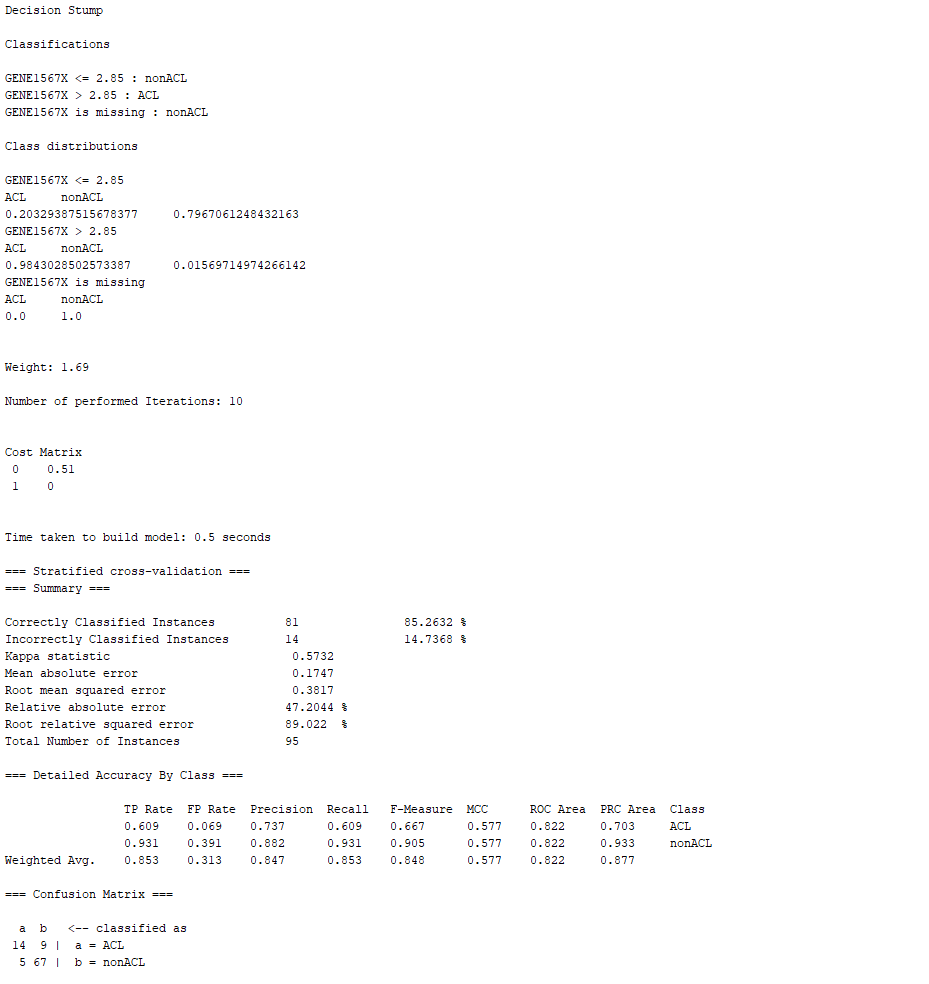
Picture 30: Cost Sensitive Classifier Combined with Boosting and Decision Stump – Cost Matrix 0.49 – 25-Fold Cross Validation





Picture 31: Cost Sensitive Classifier Combined with Boosting and Decision Stump – Cost Matrix 0.5 – 25-Fold Cross Validation





Picture 32: Cost Sensitive Classifier Combined with Boosting and Decision Stump – Cost Matrix 0.51 – 25-Fold Cross Validation

From these results, it can be noted that when the cost sensitive matrix value was 0.48, 0.49, 0.5, and 0.51, the model had the best performance with the lowest misclassification of the minority class and lowest type II error ratio. Moreover, when compared to the cost sensitive tree obtained in Assignment 1 part 4, it can be noted that the misclassification of the minority class is higher from the model in picture 32. It is important to note that the cost sensitive classifier combined with bagging and decision stump with a cost matrix 5 and 25-fold cross validation provided the highest performance in properly classifying the samples from the minority class. Nevertheless, it is worth considering the fact that this high level of accuracy in the classification of the minority class resulted in the biggest misclassification rate of the majority class. Therefore, although this model might be the best model for classifying the minority class, it is not a balanced model that could be used for the proper classification of both classes combined. Nevertheless, since the minority class is the class of interest, the tradeoff between how much misclassification from the majority class can be permitted would need to be analyzed. Finally, from these experiments it can be concluded that the models built with decision stump and bagging under the 25-fold cross validation performed better at classifying the minority class of the dataset in comparison to other techniques and models used during the experimentation.