# Decision trees, Random forests and Gradient boosting for cancer type classification

**Michael Cornelisse s1059020, Nienke Helmers s1016904**
January 11, 2021

## Abstract

Classification of cancer types is a greatly investigated research area and in this project we compare the classification of the three classifiers decision tree, random forest and gradient boosting. The models are made based on the gene expression of two cell types: AML and ALL. This led to the conclusion that a random forest model has the best performance using the setup, with an accuracy of 0.853 and an AUC value of 0.821. The standard decision tree scores the lowest with an accuracy of 0.794 and an AUC of 0.75.
*The complete code and toolbox are available at https://github.com/Michael-Cornelisse/Datamining-project*

## Introduction

There are many different types of cancer currently known. The differences between all those types of the same tissue are sometimes very minimal, but someone with enough experience is able to semi-accurately determine the type. There have already been many attempts at creating a classification model that can classify the cancer type based on pictures of the cells and their surrounding tissues.

However, all of the differences of these cancer types are at their core based on the gene expression of the cells. The gene expression patterns of these cells can be determined by measuring the amount of RNA present per cell. The expression pattern can still vary slightly between cancer cells of a similar type because of the environment or the stage in the life cycle of the cell

With the help of classification algorithms a cancer type can be determined based on the gene expression. They allow for a more flexible determination of the type by using existing data to build a model, making use of the values of the attributes of already existing data. This model can then be used to assign a type to an unknown cell with new data.

For classification with gene expression many models have been made as well, with pictures of the cells. One of those models is for classifying two specific types of cancer: acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL), made by Golub et al (1999). They sorted the genes based on correlation and made predictions per gene, which were then assigned a weight and summed. As an accuracy test they used cross validation.

This is but one method for classification, and there are many more, among which decision trees. These trees can be utilised in many ways, either on their own or in combination with other algorithms, also in combination with themselves. This is called ensemble method, of which there are multiple types, two of which are bagging and boosting. Bagging involves training individual models parallel to each other and combining them later down the line, while boosting trains the models sequentially and takes the mistakes of the previous model into account (Chen, 2019).

Random forest is one of the bagging methods of assembling using decision trees. Using small sample fractions of the data with replacement, done with bootstrap, multiple decision trees are grown. After this they are combined into one by averaging. It is a versatile method that has a good performance for training data with a small sample size, but with a large number of attributes (Biau and Scornet, 2016).

Gradient boosting machines are relatively simple, and thus highly flexible algorithms that learn by making consecutive models based on the data and previously made models. The new model makes use of a loss function, which gives a cost to the inaccuracies in a model. This loss function, of which there are many types, should be minimized as much as possible to create the best classifier (Natekin and Knoll, 2013).

In this project the performance of a standard decision tree, a random forest build tree and a gradient boosting tree are compared to each other on the classification of the gene expression data of AML and ALL cancer cells. Since the ensemble methods allow for flexibility and learning based on multiple models, we expect those two to outperform a standard decision tree. However, whether random forest or gradient boosting is better is more difficult to predict, as it is also dependent on the amount of data samples and its attributes. Since gradient boosting relies on a more coordinated way of training, we predict that model to be better than the random forest one.

## Methods

For the preprocessing, first the unnecessary columns were removed. The call columns in de gene data each sample which genes are present (P), absent (A) or marginal (M). Since the raw data of the gene expression will be used, and not its interpretation, these columns can be removed. Additionally, to prevent the inclusion of the wrong data, the columns for gene description and gene accession number are removed from the data set.

Then, the data is transposed for the correct orientation, as before the attributes were displayed in the collums and the samples over the rows. Since the indices in the attribute data did not line up with the class data, the data was sorted to prevent making a wrong model.

No standardization was performed on the data because only decision trees are used. Decision trees use comparisons of data and thus scaling has no effect.

After sorting, principal component analysis was performed on the training data and then applied to the test data to reduce the amount of attributes from seven thousand to a more reasonable amount. This is necessary as decision trees have a tendency to overfit when the ratio between samples and attributes is high (Scikit-learn, 2020). The number of attributes was selected on explanation of 90% of the data. The train and test data sets were merged to be able to perform k-fold calculations.

Each of the models for the decision tree, random forest and gradient boosting forest has many parameters that can be altered and they are important for the model's performance. Tree depth that partly determines the complexity. The number of estimates in ensemble methods dictates how many trees are built to determine the best model and the minimum samples of a leaf makes sure that there are enough samples in a node to make an accurate split, though this parameter is not present in the gradient boosting algorithm. The evaluation metric/criterion is of essence, as that determines the quality of a split in the tree. Random state keeps the selection of features and samples steady across multiple runs of the algorithm.

For estimates of these five parameters, first different values were tried iteratively, testing the accuracy with k-fold loop (see Appendix, S1-S5). Then using grid search and the estimates, the optimal parameter composition for each model was found.

Using the parameters, the models were fitted and the accuracy was measured based on the prediction versus the true classes. In addition to that, a confusion matrix and an ROC curve were plotted for each model.

## Results

The pca ran over the test data resulted in 38 components. The 90% of the summed variance is explained by the first 21 attributes, as can be seen in figure 1.

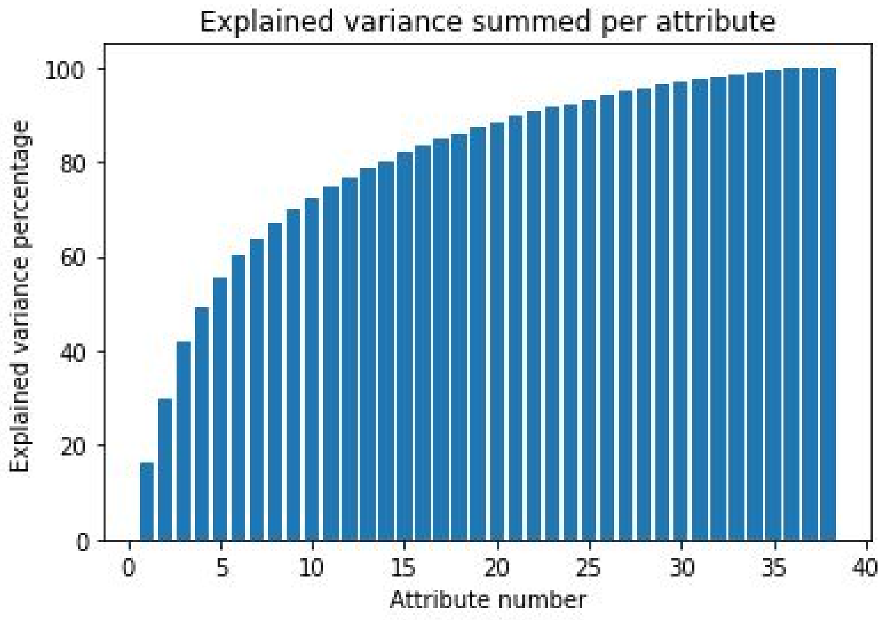


*Figure 1: A plot of the explained variance. For every attribute number, the variance from attribute 1 up to and including that attribute is summed.*

### Decision tree

The standard decision tree has an accuracy of 0.794 with an error rate of 0.206. Of the 34 samples in the test set, 7 were predicted wrongly. All the wrongly predicted samples were predicted to be of the ALL class instead of the AML as can be seen in figure 2. The model results in an AUC of 0.75 (figure 3).

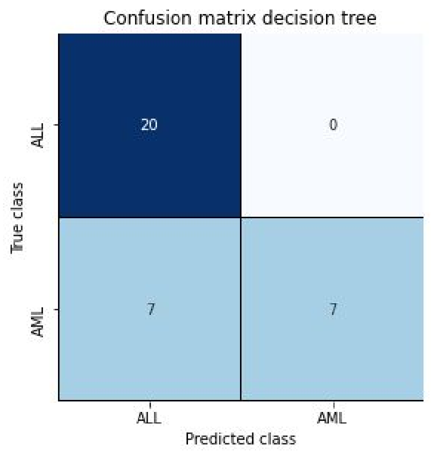


*Figure 2: A confusion matrix of the model made with a decision tree.*

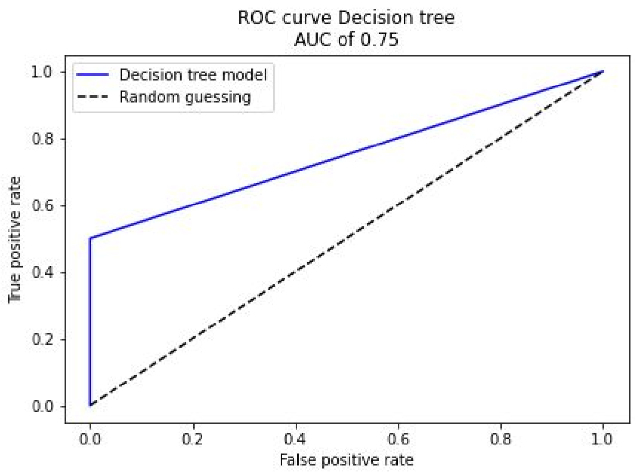


*Figure 3: An ROC curve of the model made with a decision tree. It also shows the line of random guessing in black.*

The resulting decision tree is visible in figure 4 and was made with the parameter tree depth at 2, minimum sample leaf at 2 and random state 10.

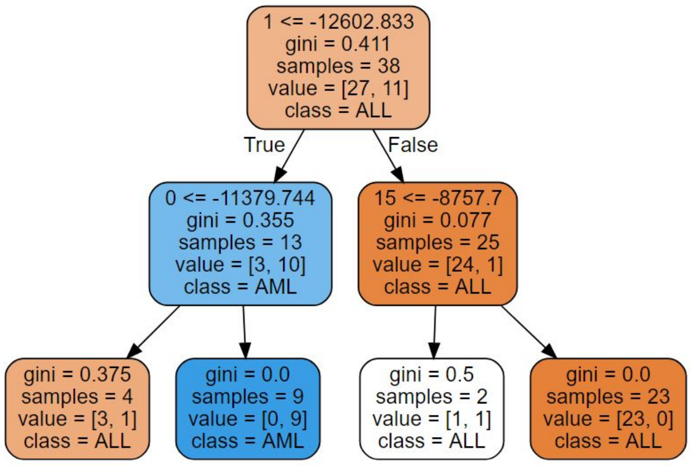
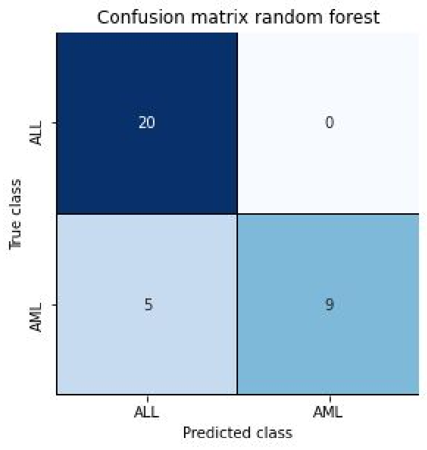


*Figure 4: The model made with a standard decision tree algorithm. The first line for a node split indicates which attribute of the PCA is used for the split. Gini indicates the impurity of the node, with the value of 0 being completely accurate. Samples gives the amount of samples in a node and value gives the class distribution of the samples.*

## Random forest

A random forest algorithm results in an accuracy of 0.853 and an error rate of 0.147. Its confusion matrix in figure 5 shows that 5 samples were wrongly put in the ALL class instead of AML. The ROC curve shows an AUC of 0.821 (figure 6).



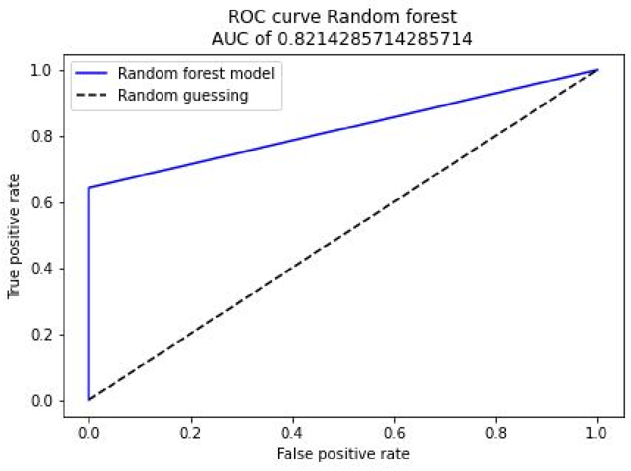


*Figure 5: A confusion matrix of the model made with a random forest algorithm.*

*Figure 6: An ROC curve of the model made with a random forest algorithm. It also shows the line of random guessing in black.*

One of the estimators for the random forest algorithm is shown in figure 7, made with the parameter for tree depth set at 4, the number of estimators at 250, the minimum samples per leaf at 2 and the random state at 42.

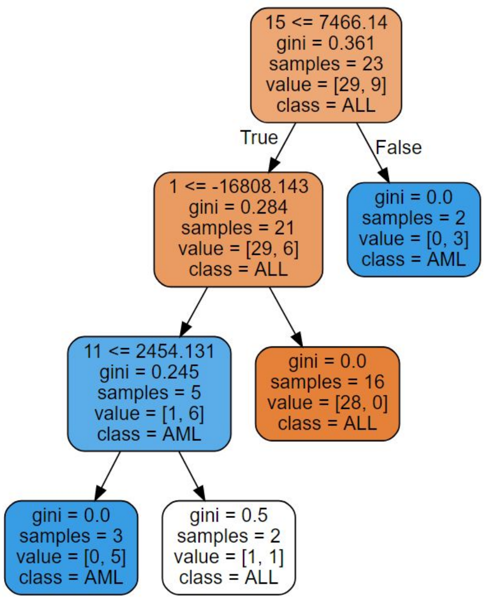


*Figure 7: The model made with the random forest algorithm. The first line for a node split indicates which attribute of the PCA is used for the split. Gini indicates the impurity of the node, with the value of 0 being completely accurate. Samples gives the amount of samples in a node and value gives the class distribution of the samples.*

## Gradient boosting

The model that resulted from the XGBoost algorithm gives an accuracy of 0.824 and an error of 0.176. It misclassified six samples into the ALL class instead of the AML class, as can be seen in the confusion matrix in figure 8. In figure 9 the ROC curve is depicted, which shows an AUC value of 0.786.

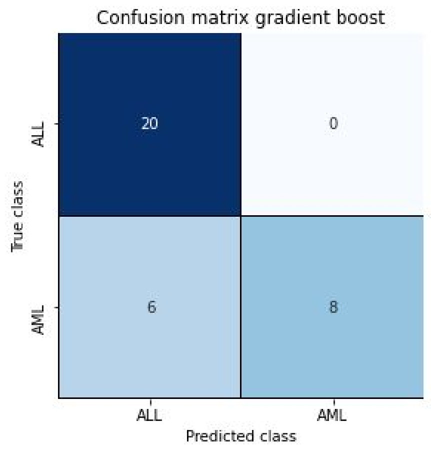


*Figure 8: A confusion matrix of the model made with the XGBoost algorithm.*

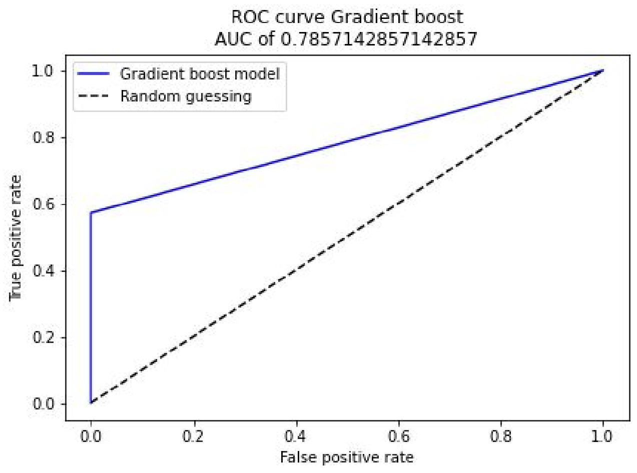


*Figure 9: An ROC curve of the model made with the XGBoosts algorithm. It also shows the line of random guessing in black.*

One of the estimators for the random forest algorithm is shown in figure 7, made with the parameter for tree depth set at 4, the number of estimators at 250, the minimum samples per leaf at 2 and the random state at 42.

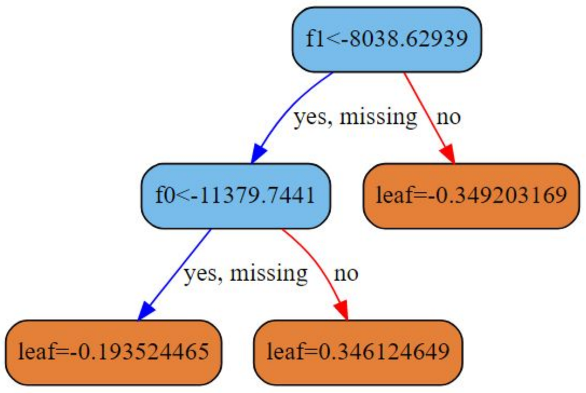


*Figure 10: The decision tree resulting from the XGBoost algorithm. Each node split shows which attribute is used for the splitting and the value of the attribute that determines the direction. The final nodes show the leaf value, which represents the probability of the prediction being of class -1 or 1 in binary classification.*

## Discussion

In general, the accuracies and AUC values for all of our models is decent. The standard decision tree has an accuracy of 0.794 and the random forest tree has one of 0.853 while the gradient boosted tree has an accuracy of 0.824. This means that the standard decision tree algorithm yields the worst classifier of the three, while the random forest algorithm gives the best result. That the standard tree would be the worst was expected, since it is the simplest algorithm we used. It still has an accuracy close to the other two models and the accuracy and AUC values are nonetheless high. However, that the random forest model was better than the boosted model was not according to the hypothesis.

Why this is the case is difficult to determine. It could be that the random forest algorithm is somewhat less sensitive to overfitting, which is an issue we will also discuss. Another possibility is a sensitivity of the gradient boosting algorithm to noise. We did not find any missing values, though it is very much possible that the gene expression data itself contains noise. Additionally, as mentioned before, genetic expression can vary a lot within the cell cycle and under various circumstances, which could make it difficult for a boosting algorithm to make an accurate classifier.

During exploration of the data we noticed an imbalance: the complete data set contained 47 counts of the class ALL, and 25 counts of AML under the patients inside the set. This resulted in a training set that contained 27 counts of ALL and 11 counts of AML, and a test set that contained 20 counts of ALL and 14 cases of AML. Due to the low sample size we determined that we could not cut back on the amount of cases of ALL that were present in the dataset. This imbalance was clearly visible during classification in the confusion matrix which displayed all misclassifications of ALL as AML, no matter the type of algorithm used. This indicates overfitting on the data, resulting in a bias towards the ALL class.

## Conclusion

To conclude, on this data set, the best algorithm is that of the random forest classifier with an accuracy of 0.853. However, we have worked with a very limited and unbalanced dataset. In addition to the parameters we tested, there are many more that might improve the accuracy of the models even more. To improve the results of the models, and potentially change which algorithm is the optimal one, many more data is required. Testing over more parameters and their ranges is also an option.

## References

[1] Biau, G. and Scornet, E., 2016. Rejoinder on: A random forest guided tour. TEST, 25(2), pp.264-268.

[2] Chen, L., 2019. Basic Ensemble Learning (Random Forest, Adaboost, Gradient Boosting)- Step By Step Explained. [online] Medium. Available at: <https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725> [Accessed 17 December 2020]

[3] Golub, T., Slonim, D., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J., Coller, H., Loh, M., Downing, J., Caligiuri, M., Bloomfield, C. and Lander, E., 1999. Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. Science, 286(5439), pp.531-537.

Data: 2017. Gene Expression Dataset (Golub Et Al.). [online] Kaggle.com. Available at: <https://www.kaggle.com/crawford/gene-expression> [Accessed 10 January 2021].

[4] Lemaitre, G., 2020. Sklearn.Ensemble.Randomforestclassifier. [online] Scikit-learn.org. Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html> [Accessed 10 January 2021].

[5] Lemaitre, G., 2020. Sklearn.Tree.Decisiontreeclassifier. [online] Scikit-learn.org. Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html> [Accessed 10 January 2021].

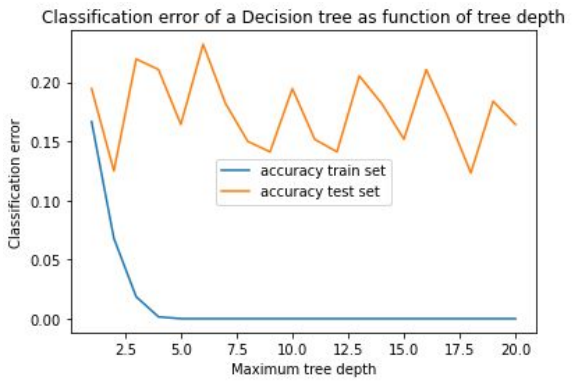
[6] Natekin, A. and Knoll, A., 2013. Gradient Boosting Machines, A Tutorial. [online] Frontiers in neurorobotics. Available at:

<https://www.frontiersin.org/articles/10.3389/fnbot.2013.00021/full> [Accessed 10 January 2021].

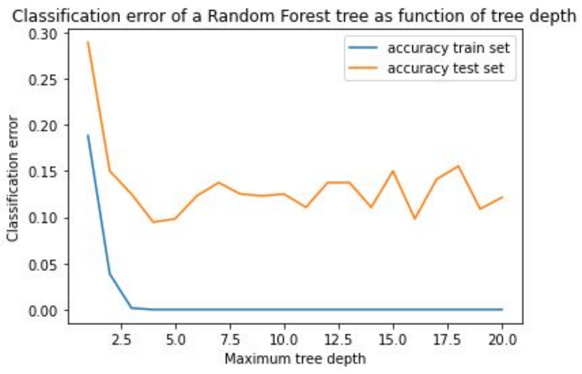
[7] Scikit-learn. 2020. Decision Trees. [online] Available at: <https://scikit-learn.org/stable/modules/tree.html#tree> [Accessed 23 December 2020].

[8] Xgboost.readthedocs.io. 2020. Xgboost Documentation. [online] Available at: <https://xgboost.readthedocs.io/en/latest/index.html> [Accessed 10 January 2021].

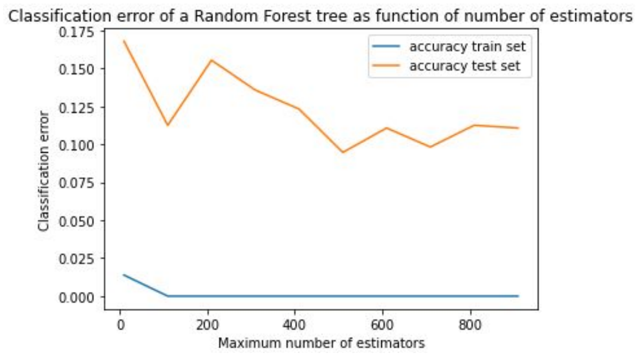
## Appendix



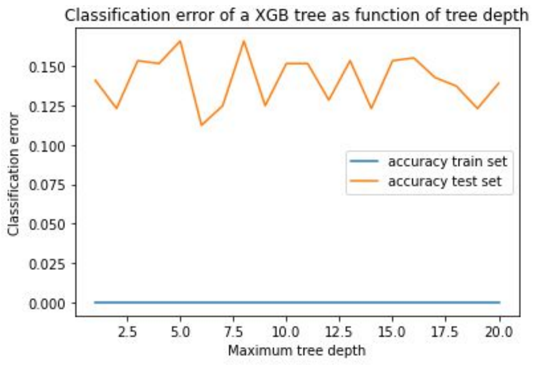
*S1: A plot of the error of a decision tree as a function of the maximum tree depth, validated with ten fold cross validation.*



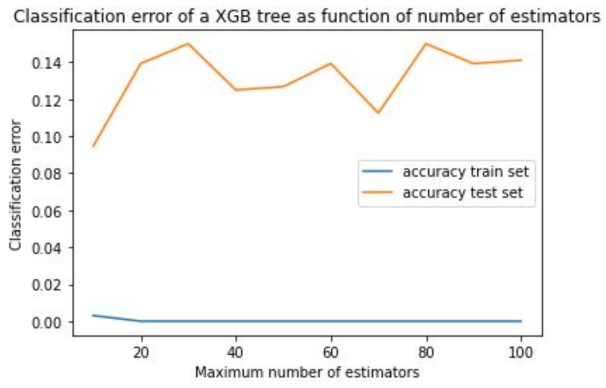
*S2: A plot of the error of a random forest decision tree algorithm as a function of the maximum tree depth, validated with ten fold cross validation.*



*S3: A plot of the error of a random forest decision tree as function of the number of estimators, examined with ten fold cross validation.*



*S4: A plot of the error of an XGB boosted tree as a function of tree depth and examined with ten fold cross validation.*



*S5: A plot of the error of a XGB boosted tree as a function of the maximum number of estimators. It was validated using ten fold cross validation.*